LUIZ CARLOS BRANDÃO JUNIOR

A METHODOLOGY FOR SYNTHETIC DATA GENERATION AND VALIDATION APPLIED TO AGRICULTURAL ENGINEERING

LAVRAS - MG
2026

LUIZ CARLOS BRANDÃO JUNIOR

A METHODOLOGY FOR SYNTHETIC DATA GENERATION AND VALIDATION APPLIED TO AGRICULTURAL ENGINEERING

Thesis submitted to the Federal University of Lavras as part of the requirements for the Postgraduate Program in Agricultural Engineering, area of concentration in Agricultural Machinery and Mechanization, for obtaining the title of Doctor.

Prof. DSc. Ricardo Rodrigues Magalhães
Supervisor

Documento assinado digitalmente
gov.br
DANTON DIEGO FERREIRA
Data: 14/08/2026 17:08:28-0300
Verifique em https://validar.iti.gov.br

Prof. DSc. Danton Diego Ferreira
Co-supervisor

LAVRAS - MG
2026

Ficha Catalográfica elaborada pelo Sistema de Geração de Ficha Catalográfica da Biblioteca Universitária da UFLA, com dados informados pelo(a) próprio(a) autor(a).

Brandão Junior, Luiz Carlos.
A Methodology for Synthetic Data Generation and Validation Applied to Agricultural Engineering / Luiz Carlos Brandão Junior. - 2026.
172 p.

Orientador: Ricardo Rodrigues Magalhães
Coorientador: Danton Diego Ferreira

Tese (Doutorado) - Universidade Federal de Lavras, 2026.
Bibliografia.

1. Synthetic Data. 2. Cholesky Decomposition. 3. Arabica Coffee. 4. Sensory Analysis. 5. Statistical Validation. I. Rodrigues Magalhães, Ricardo. II. Diego Ferreira, Danton. III. Universidade Federal de Lavras. IV. Título.

LUIZ CARLOS BRANDÃO JUNIOR

A METHODOLOGY FOR SYNTHETIC DATA GENERATION AND VALIDATION APPLIED TO AGRICULTURAL ENGINEERING

UMA METODOLOGIA PARA GERAÇÃO E VALIDAÇÃO DE DADOS SINTÉTICOS APLICADA À ENGENHARIA AGRÍCOLA.

Thesis submitted to the Federal University of Lavras as part of the requirements for the Postgraduate Program in Agricultural Engineering, area of concentration in Agricultural Machinery and Mechanization, for obtaining the title of Doctor.

APPROVED on 09 of July of 2026.

Prof. DSc. Ricardo Rodrigues Magalhães	UFLA
Prof. DSc. Danton Diego Ferreira	UFLA
Prof. DSc. Danilo Alves de Lima	UFLA
Prof. DSc. Wilian Soares Lacerda	UFLA
Prof. DSc. Giovanni Bernardes Vitor	UNIFEI

gov.br
Documento assinado digitalmente
RICARDO RODRIGUES MAGALHAES
Data: 14/08/2026 15:58:52-0300
Verifique em https://validar.iti.gov.br

Prof. DSc. Ricardo Rodrigues Magalhães
Supervisor

Prof. DSc. Danton Diego Ferreira
Co-supervisor

LAVRAS - MG
2026

To my family, for their understanding and unconditional support.

ACKNOWLEDGMENTS

To God, the source of wisdom, strength, and inspiration throughout this journey.

To the Universidade Federal de Lavras (UFLA), for the academic environment of excellence, for its excellent facilities, for the well-equipped laboratories, available servers and computers, and for the study rooms that greatly contributed to my learning and development.

To the professors, for the knowledge transmitted with dedication and seriousness. To Professor Ricardo Magalhães, for the secure guidance and the valuable advice that shaped this work. To Professor Danton Diego, for the attentive co-supervision and the critical eye that greatly enriched the research. To the technical staff, for the fundamental support and the silent, yet indispensable, dedication in the day-to-day work.

To my colleagues, for the camaraderie, for the exchange of ideas, for the mutual encouragement, and for the good conversations that made this journey lighter and more meaningful.

To CAPES, for the financial support through the scholarship, which was essential for the realization of this work.

To my parents, for their unconditional love, for the values they instilled in me, and for always being by my side. In particular, to my mother, whose presence, prayers, and affection were fundamental at every moment — especially in the most difficult ones.

And, finally, I also thank life's challenges, which taught me to grow, to persevere, and to move forward with greater maturity and determination.

"Statistics is the science that says that if I ate one chicken and you ate none, we will have eaten, on average, half a chicken each." Dino Segrè (Pitigrilli)

RESUMO

A escassez de dados experimentais de qualidade, juntamente com os altos custos de análises laboratoriais e sensoriais, são barreiras significativas para o desenvolvimento da modelagem computacional e da inteligência artificial na área das Ciências Agrárias. Esta tese introduz um *framework* de engenharia reversa para a geração de dados sintéticos baseado na decomposição de Cholesky, que pode reconstruir distribuições multivariadas enquanto preserva as propriedades estatísticas e a dependência estrutural entre as variáveis. Como caso de estudo, a técnica foi empregada em um modelo para a qualidade sensorial do café Arábica com um conjunto de dados público do Coffee Quality Institute ($n = 207$ amostras). O método foi dividido em quatro estágios: (i) extração e pré-processamento de dados (ii) síntese estatística preservando momentos de ordem um e dois (iii) validação múltipla de fidelidade estatística e (iv) avaliação do poder preditivo via o protocolo Train Synthetic, Test Real (TSTR). Os resultados mostraram uma elevada fidelidade estrutural, obtendo-se similaridade da matriz de correlação 96,20% e similaridade pelo teste de Kolmogorov-Smirnov (KS) 85.81%, todas as variáveis ultrapassaram o limiar técnico de aceitação de 70%. Na regressão RF, apenas os modelos treinados com dados sintéticos alcançaram $R^2 = 0,9498$, Erro Médio Absoluto (MAE) de apenas 0,216 e precisão de 100% dentro da margem de erro de $\pm 10\%$. Para a classificação ordinal da qualidade sensorial, a abordagem atingiu 89,37% de acurácia e (Weighted Quadratic) Kappa Coeficiente de 0,9558, mostrando concordância quase perfeita com os padrões da Specialty Coffee Asscociation (SCA). Como principal contribuição científica, esta tese estabelece um framework de engenharia reversa para reconstrução de bases de dados sintéticas a partir de estatísticas agregadas, demonstrando que a preservação simultânea da fidelidade estatística e da utilidade preditiva pode ser alcançada sem o emprego de modelos generativos profundos. A abordagem proposta amplia as possibilidades de compartilhamento seguro de dados, reduz a dependência de experimentação onerosa e oferece uma solução computacionalmente eficiente e de fácil reprodução, com elevado potencial de transferência tecnológica para aplicações em Engenharia Agrícola, Agronomia e demais áreas que dependem de bases de dados experimentais confiáveis.

Palavras-chave: Dados sintéticos; Cholesky; Café Arábica; Análise sensorial; Validação estatística; Aprendizado de máquina.

ABSTRACT

The scarcity of high-quality experimental data, coupled with the high costs of laboratory and sensory analyses, are significant barriers to the development of computational modeling and artificial intelligence in the field of Agricultural Sciences. This thesis introduces a reverse engineering framework for generating synthetic data based on Cholesky decomposition, which can reconstruct multivariate distributions while preserving statistical properties and structural dependence between variables. As a case study, the technique was employed in a model for the sensory quality of Arabica coffee using a public dataset from the Coffee Quality Institute ($n = 207$ samples). The method was divided into four stages: (i) data extraction and preprocessing (ii) statistical synthesis preserving first and second order moments (iii) multiple validation of statistical fidelity and (iv) evaluation of predictive power via the Train Synthetic, Test Real (TSTR) protocol. The results showed high structural fidelity, obtaining a correlation matrix similarity of 96.20% and a Kolmogorov-Smirnov (KS) test similarity of 85.81%; all variables exceeded the technical acceptance threshold of 70%. In RF regression, only the models trained with synthetic data achieved R^2 = 0.9498, a Mean Absolute Error (MAE) of only 0.216, and 100% accuracy within a margin of error of ±10%. For the ordinal classification of sensory quality, the approach achieved 89.37% accuracy and a Weighted Quadratic Kappa Coefficient of 0.9558, showing almost perfect agreement with the standards of the Specialty Coffee Association (SCA). As its main scientific contribution, this thesis establishes a reverse engineering framework for reconstructing synthetic databases from aggregated statistics, demonstrating that the simultaneous preservation of statistical fidelity and predictive utility can be achieved without the use of deep generative models. The proposed approach expands the possibilities for secure data sharing, reduces dependence on costly experimentation, and offers a computationally efficient and easily reproducible solution with high potential for technology transfer to applications in Agricultural Engineering, Agronomy, and other areas that depend on reliable experimental databases.

Keywords: Synthetic data; Cholesky decomposition; Arabica coffee; Sensory analysis; Statistical validation; Machine learning.

INDICADORES DE IMPACTO

Este estudo realizou um trabalho de base para construir uma abordagem estatística capaz de construir e validar conjuntos de dados sintéticos com base apenas em estatísticas descritivas agregadas: médias, desvios padrão, quartis e matriz de correlação. Apesar do teste de método ter sido realizado em relação aos dados de qualidade do café Arábica, é possível usá-lo em qualquer área da Engenharia Agrícola, pois basta apenas que existam estatísticas descritivas publicadas, sem necessidade de acesso aos dados brutos.

Do ponto de vista técnico, é apresentado um pipeline computacional baseado na linguagem Python, código aberto e capaz de produzir conjuntos de dados estatisticamente similares ao conjunto original sem expor os dados brutos originais. O método foi amplamente validado, atingindo uma similaridade KS de 85,81% e uma similaridade de correlação de 96,20%, além de ter todas as variáveis acima da similaridade mínima de 70%. Os resultados demonstraram que o método é uma boa alternativa aos métodos geradores neurais, como GANs e VAEs, que necessitam acesso integral dos dados brutos. Além disso, é proposta uma metodologia de validação com base em métodos já conhecidos (Similaridade KS, Similaridade JS, Similaridade por Correlação e Norma de Frobenius), que pode ser utilizada futuramente.

Os benefícios econômicos emanam da diminuição dos custos de coleta dos dados em campo, pois estatísticas descritivas publicados em artigos científicos podem ser reaproveitadas para a construção de novas bases eliminando ou minimizando gastos com Equipamentos, transporte e mão de obra, na realização de estudos preliminares. A pesquisa documenta que as informações já publicadas têm valor intrínseco além da documentação, ampliando o retorno sobre os investimentos em pesquisa. O script Python desenvolvido é passível de transferência tecnológica para instituições de pesquisa, empresas agrícolas e órgãos governamentais que necessitem gerar dados sintéticos com propriedades estatísticas preservadas.

No campo social, a pesquisa torna-se democratizada ao permitir que pesquisadores e estudantes de instituições menos abastadas possam realizar testes, validar hipóteses e criar modelos preditivos sem terem que sair em campo para fazer coletas custosas. A geração de dados a partir de estatísticas agregadas assegura também a privacidade e confidencialidade dos dados originais, observando restrições éticas, legais e acordos institucionais. A liberação dos códigos e da base sintética contribui para o open science e para a ciência reproduzível minimizando potencial viés e fortalecendo a credibilidade científica, ao permitir a validação independente dos resultados. O trabalho foi desenvolvido ao longo da qualificação de um Doutor em Engenharia Agrícola com ênfase em modelagem estatística multivariada, programação científica e validação de dados, sendo de grande relevância para o aprimoramento da capacitação para IA no país.

Quanto ao seu viés extensionista, o trabalho se reveste de importância para pesquisadores, pós-graduandos, graduandos e profissionais da área de Engenharia Agrícola, Agronomia, Ciência de Dados e Estatística Aplicada em âmbito nacional e internacional. Os resultados serão divulgados por meio de artigos científicos submetidos para periódicos de alto impacto, disponibilização do código-fonte em repositório público https://github.com/zolpy/Doutorado_

UFLA, divulgação da base sintética gerada e apresentações em eventos científicos. Importante salientar que a metodologia permite a geração de novas bases sintéticas por qualquer pesquisador que disponha das estatísticas descritivas de seus dados, o que certamente ampliará muito seu alcance e aplicabilidade a outros contextos, e territórios para além da Engenharia Agrícola. Quanto às áreas temáticas da Política Nacional de Extensão, os impactos deste trabalho concentram-se, em sua maior parte, em Tecnologia e Produção (Área 7), devido ao aporte metodológico e computacional na produção agrícola e no monitoramento hídrico; em Educação (Área 4), pela formação de recursos humanos e fomento à ciência aberta; em Meio Ambiente (Área 5), pela utilização para um uso eficiente da água na agricultura; e em Comunicação (Área 1), na maneira como ele valoriza e ressignifica estatísticas descritivas publicadas como insumo para a geração de novos dados.

Por fim, a pesquisa alinha-se diretamente a cinco dos dezessete Objetivos de Desenvolvimento Sustentável da ONU: ODS 4 (Educação de Qualidade), pela democratização do acesso a dados e pela formação de pesquisadores qualificados (Meta 4.4); ODS 6 (Água Potável e Saneamento), pela contribuição ao monitoramento do potencial hídrico e ao uso eficiente da água na agricultura (Meta 6.4); ODS 9 (Indústria, Inovação e Infraestrutura), pelo desenvolvimento de metodologia e software para geração de dados sintéticos (Meta 9.5); ODS 12 (Consumo e Produção Responsáveis), pela reutilização de informações já publicadas, reduzindo a demanda por novas coletas de campo (Meta 12.2); e ODS 17 (Parcerias e Meios de Implementação), pelo compartilhamento aberto de metodologia, código e bases sintéticas entre instituições nacionais e internacionais (Meta 17.6).

LIST OF ACRONYMS

CQI	Coffee Quality Institute
GANs	Generative Adversarial Networks
IA	Inteligência Artificial
JSD	Jensen-Shannon divergence
KDE	Kernel Density Estimation
MAR	Missing at Random
MCAR	Missing Completely at Random
MNAR	Missing Not at Random
QWK	Quadratic Weighted Kappa
SCA	Specialty Coffee Association
SCAA	Specialty Coffee Association of America
TSTR	Train Synthetic, Test Real
VAEs	Variational Autoencoders

TABLE OF CONTENTS

1 INTRODUCTION

Coffee quality is an inherently multifaceted property that results from a complicated interaction of genetic, environmental, agronomic, and processing factors. Sensory analysis performed by qualified cuppers according to Specialty Coffee Association (SCA) guidelines is considered a gold standard for determining the quality of coffee, scoring it from 0 to 100 points. Even though the method is standardized and reliable, it is inherently subjective and based on human perception (Lingle, 2011), (Lawless; Heymann, 2010).

Machine learning and data analysis techniques have been actively used in the specialty coffee industry recently to find patterns, predict coffee quality, and optimize processes (Reiter, 2005). Nonetheless, obtaining good-quality datasets for model training is rather difficult due to limited availability of samples analyzed using sensory methods, costly analysis, and confidentiality issues related to producer and trader data (Raab; Nowok; Dibben, 2018), (Little et al., 2021).

However, despite these achievements, there are three important aspects which should be considered: (i) the problem of insufficient quantity of large-scale sensory databases caused by expensive and long-time procedures of cupping; (ii) the problem of the lack of systematic methodologies for creation and validation of synthetic data for agriculture sensory analysis; and (iii) the problem of insufficient validation strategies concerning statistical similarity and predictive ability of synthetic data.

Thus, generation of synthetic data is one of the ways to avoid data availability problems (Abay et al., 2018). Synthetic data are artificial datasets which simulate statistical and structural characteristics of original datasets, but do not contain any real information on particular individuals or entities (Borders; Thompson; Kearney, 2025), (Hughes, 2025). If synthetic data are properly created, they can be used for training models, testing hypotheses and conducting simulations, which considerably expands opportunities of data analysis in such spheres, where data acquisition is difficult (Miletic; Sariyar, 2025).

The technique of the synthetic data generation relies on many different statistical and computational methods, from classic parametric techniques, like Cholesky decomposition and multivariate normal distribution (Marchev Jr.; Marchev, 2023), to deep learning techniques, including Generative Adversarial Networks (GANs) (Yoon; Jarrett; Schaar, 2019) and Variational Autoencoders (VAEs) ("Uncover This Tech Term: Variational Autoencoders," 2025), ("Cyberdefense Powered by Generative AI," 2026). The selection of the right technique depends on the features of the original dataset, as well as the goals of the synthesis and the required level of the generated data fidelity and privacy (Drechsler, 2023), (Awan; Cai, 2025).

In particular, Cholesky decomposition allows generating the synthetic data which will retain the correlation structure of the original dataset in an elegant mathematical way and at the same time will be computationally efficient (Golub; Van Loan, 2013), (Tong, 2012). Cholesky decomposition is based on covariance matrix decomposition and guarantees that the first and second order moments will be retained exactly.

The process of validating the synthetic data is a key aspect to enable them to have practical relevance and validity. The validation processes should involve the assessment of marginal distribution similarity (via means such as Kolmogorov-Smirnov test and Jensen-Shannon Divergence) and the correlation structure conservation (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018).

Despite the increasing interest in synthetic data, few studies have investigated whether statistical reconstruction from aggregated descriptive measures can simultaneously preserve distributional fidelity, correlation structure and predictive performance in agricultural datasets. This thesis addresses this gap by proposing and validating a reverse engineering framework based on Cholesky decomposition, using Arabica coffee sensory quality as a case study.

1.1 Objectives

The objective of this research work is to design and validate a reverse engineering technique to construct artificial datasets from aggregate statistics, where Cholesky decomposition will be used as the method for synthesizing the data set in the sensory quality modeling of Arabica coffee.

1.1.1 Specific Goals

Specifically, the project aims to:

a) Extract and process the real Arabica coffee dataset by describing its statistical and structural characteristics, baseline for further synthesis (Script 01 - Appendix A);
b) Create the synthetic dataset maintaining means, variances, and correlations of the original data set using Cholesky Decomposition method (Script 02 - Appendix B);
c) Measure the quality of synthetic data set by using statistical (Kolmogorov-Smirnov and Jensen-Shannon tests) and structural (correlations) similarities (Script 03 - Appendix C);
d) Assess the practical value of the synthetic data set with respect to Train Synthetic, Test Real approach in regression and ordinal classification problems (Script 04 - Appendix D).

1.2 Contributions

The main contributions of this thesis are:

a) **Methodological**: This research develops a reverse engineering paradigm for reconstructing synthetic tabular datasets based on aggregated descriptive statistics. Unlike the existing methodologies, which use raw data and deep generative models, the methodology developed in this research maintains the statistical and predictive accuracy by employing a fast Cholesky-based synthesis method;
b) **Computational**: An open-source Python framework consisting of four independent scripts implementing data preprocessing, synthesis, statistical and predictive validation;

c) **Practical**: The proof of the fact that statistically generated synthetic data is able to maintain correlation and distribution properties and preserve predictive performance, allowing to train machine learning models without having the original dataset at hand;
d) **Applied**: A methodology that can be used in the field of Agricultural Engineering and similar fields in order to share data, test hypotheses and develop artificial intelligence in cases when data are scarce, confidential or expensive to obtain;
e) **Scientific**: This research proves that synthetic tabular datasets generated only from aggregated descriptive statistics are capable of preserving both statistical fidelity and predictive performance and therefore constitutes an alternative to deep generative models.

1.3 Methodology Pipeline Structure

The research is structured according to four methodological phases that have been implemented in the form of Python scripts:

a) **Script 01: Data Pre-processing**: Real data extraction, cleaning and features selection (Appendix A);
b) **Script 02: Data Synthesis**: Synthetic data generation by means of Cholesky decomposition (Appendix B);
c) **Script 03: Data Statistics**: Distributions (KS, JS) and correlation matrix similarity analysis (Appendix C);
d) **Script 04: Predictive Validation**: TSTR procedure for regression and ordinal classification (Appendix D).

The structure described provides high-level of reproducibility and ease for applying this methodology in different domains.

1.4 Structure of the Thesis

This Thesis consists of five chapters, reflecting the logic of research:

- **Chapter 1 (Introduction)**: Discusses the background, problem statement, objectives, and contributions of the research.
- **Chapter 2 (Theoretical Background)**: Introduces the mathematical and statistical aspects necessary for generating synthetic data, such as multivariate normal distribution, Cholesky decomposition, correlation, and similarity.
- **Chapter 3 (Materials and Methods)**: Outlines the experiment design, data collection, pre-processing, synthesis procedure, evaluation metrics and the predictive evaluation framework that was implemented in four Python scripts;
- **Chapter 4 (Results and Discussion)**: Offers the analysis of the similarity of real and synthetic data visually and numerically, predictive performance of the model trained on synthetic data, and the interpretation analysis;
- **Chapter 5 (Conclusions)**: Summarizes the results of the research, discusses the limitations and provides recommendations for further work.

Appendices include the full source code (Scripts 01-04).

2 THEORETICAL FRAMEWORK

This chapter provides the mathematical and statistical background of the methodology used in this research work in the synthetic data generation and validation process. The theories are classified based on the methodological scripts used in the experimental pipeline which comprises of (i) statistical background for data preprocessing (Script 01 - Appendix A); (ii) covariance structure and Cholesky decomposition for synthetic data generation (Script 02 - Appendix B); (iii) similarity measures for statistical validation (Script 03 - Appendix C); and (iv) predictive modeling and interpretability background for utility validation (Script 04 - Appendix D).

In each of the sections above, the theoretical background of the specific script is presented. This includes the mathematical formulation and reasoning behind the use of these methods in the Python implementation.

2.1 Introduction to Data Pre-processing (Script 01 - Appendix A)

Data pre-processing is an essential phase in any analysis of data as the data available in real-world settings is prone to inconsistencies, missing values, and unnecessary details that may affect further analyses (Han; Kamber; Pei, 2011). The main objective of Script 01 is to pre-process the raw data such that we obtain a clean and consistent set of data that can be used for synthetic data generation.

There are four major issues tackled by the data pre-processing process implemented in Script 01:

a) **Data Type Segregation**: Splitting the data into numerical and categorical features;
b) **Handling Missing Values**: Dealing with incomplete data;
c) **Feature Selection**: Selecting the useful variables; and,
d) **Statistical Characterization**: Calculating the statistics of the data.

2.1.1 Data Loading and Type Detection

2.1.1.1 Data Loading

The initial data set is imported via the `pandas.read_csv()` method, which transforms the CSV file to a DataFrame format (McKinney, 2010). It structures data in terms of rows (samples) and columns (variables).

2.1.1.2 Data Type Segregation

The dataset usually includes a combination of several types of data:

- **Numerical variables** (`int64`, `float64`): Continuous/discrete measurements in quantitative terms;
- **Categorical variables** (`object`): Qualitative characteristics like country of origin, manufacturing process, etc.

The segregation of numerical variables is crucial since:
a) Numerical variables form covariance relationships needed for generation of artificial data;
b) Preprocessing procedures differ for categorical variables (one-hot encoding, etc.);
c) Pearson correlation matrix used as an integral part of the synthesis approach is available for numerical variables only.

2.1.2 Numerical Data Characterization through Statistics

2.1.2.1 Sample Mean

The sample mean is the basic measure of central tendency (Triola, 2021). The formula for the sample mean of a variable X having n observations is given as follows:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.1}$$

The mean of more than one variable is as follows:

$$\overline{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i = (\overline{x}_1, \overline{x}_2, ..., \overline{x}_k)^T \tag{2.2}$$

Sample mean acts as:
- Measure of central tendency for each sensory feature; and
- First moment parameter of the multivariate normal distribution.

2.1.2.2 Sample Variance and Standard Deviation

Sample variance denotes the average squared difference from the mean (Montgomery; Runger, 2021):

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \overline{x})^2 \tag{2.3}$$

Sample standard deviation represents the square root of the variance (Devore, 2015):

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} (x_i - \overline{x})^2} \tag{2.4}$$

Standard deviation is an indicator of dispersion of sensory scores from the mean. Regarding the sensory evaluation of coffee (Coffee Quality Institute, 2024):
- Small standard deviation for traits such as `Uniformity` means that the sample is homogeneous;

2.1.2.3 Quartiles and Range

Quartiles divide the data set into four equal quarters:

- **Q1 (First Quartile)**: 25th Percentile,
- **Q2 (Second Quartile/Median)**: 50th Percentile,
- **Q3 (Third Quartile)**: 75th Percentile.

The interquartile range (IQR) is defined as:

$$\text{IQR} = Q_3 - Q_1 \tag{2.5}$$

The range equals the difference between the highest and lowest observations:

$$\text{Range} = \max(X) - \min(X) \tag{2.6}$$

Such measures allow for a powerful representation of the distribution of the data, especially when detecting outliers (Huber; Ronchetti, 2009).

2.1.3 Missing Value Treatment

2.1.3.1 The Issue of Missing Data

Missing data is an issue that arises in data collected through web scraping or hand recording (Bondarenko, 2022). The random elimination of incomplete observations leads to wastage of statistical power. The missing value mechanism may be divided into the following types:

a) **(MCAR) Missing Completely at Random**: Where the probability of missingness is independent of all other variables;
b) **(MAR) Missing at Random**: Where the probability depends on only the observed variables;
c) **(MNAR) Missing Not at Random**: Where the probability is dependent on the missing values themselves.

2.1.3.2 Median Imputation

In Script 01, imputation is done based on median of numerical data values that are missing:

$$x_{\text{ij}}^{\text{imputed}} = \begin{cases} x_{\text{ij}} \text{ if } \ x_{\text{ij}} \ \text{ is not missing} \\ \tilde{x}_j \ \text{otherwise} \end{cases} \tag{2.7}$$

The formula for the median of the j-th attribute is as follows:

$$\tilde{x}_j = x_{\left(\frac{n+1}{2}\right)} \tag{2.8}$$

The reason for using the median rather than mean for the imputation is that the former is less affected by the presence of outliers (Huber; Ronchetti, 2009). In most cases, sensory characteristics of coffee have skewed distributions or outliers that may affect the mean.

2.1.4 Feature Engineering and Variance Filtering

2.1.4.1 Feature Selection

Feature selection is the process of identifying the most relevant variables for analysis (Kuhn; Johnson, 2019). The selection criteria typically include:

a) **Domain relevance**: Variables that directly describe the phenomenon of interest;
b) **Informational integrity**: Variables that provide unique information;
c) **Computational efficiency**: Variables that contribute to model performance without redundancy.

For this research, the following categories of variables were selected:

- **Sensory attributes** (8): `Aroma`, `Flavor`, `Aftertaste`, `Acidity`, `Body`, `Balance`, `Overall`, and `Total Cup Points`;
- **Defect attributes** (2): `Quakers` and `Category Two Defects`;
- **Logistical attribute** (1): `Number of Bags`.

These variables align with the SCA cupping protocol (Lingle, 2011) and are widely used in coffee quality research (Farah, 2020), (Lawless; Heymann, 2010).

2.1.4.2 Variance Filtering

Variables that exhibit zero or near-zero variance across all samples have no discriminatory power and can cause mathematical instabilities in covariance-based algorithms (James et al., 2021). The variance filtering criterion is:

$$\text{Keep variable} \leftrightarrow s_j^2 > 0 \tag{2.9}$$

where s_j^2 is the sample variance of variable j. Variables with zero variance are removed because:

a) They provide no information for distinguishing between samples;
b) They result in singular covariance matrices;
c) They cause numerical instabilities in matrix decomposition algorithms (Golub; Van Loan, 2013).

2.1.5 Correlation Analysis

2.1.5.1 Pearson Correlation Matrix

The Pearson correlation matrix measures the linear correlation between each pair of variables (Montgomery; Runger, 2021). The correlation between variables i and j is given by:

$$r_{ij} = \frac{\sum_{k=1}^{n}(x_{ki} - \overline{x}_i)(x_{kj} - \overline{x}_j)}{\sqrt{\sum_{k=1}^{n}(x_{ki} - \overline{x}_i)^2}\sqrt{\sum_{k=1}^{n}(x_{kj} - \overline{x}_j)^2}} \tag{2.10}$$

The correlation matrix $\boldsymbol{R}$ is a $k \times k$ matrix that is symmetric in nature:

$$\boldsymbol{R} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1k} \\ r_{21} & r_{22} & \cdots & r_{2k} \\ \cdots & \cdots & \cdots & \cdots \\ r_{k1} & r_{k2} & \cdots & r_{kk} \end{pmatrix} \tag{2.11}$$

The diagonal elements $r_{ii} = 1$, and the matrix is symmetric ($r_{ij} = r_{ji}$).

2.1.5.2 Interpretation of Correlation Coefficients

Pearson correlation coefficient has the following features (Pearson, 1895):

$$-1 \leq r_{ij} \leq 1 \tag{2.12}$$

- $r_{ij} = 1$: Perfect positive linear correlation;
- $r_{ij} = -1$: Perfect negative linear correlation;
- $r_{ij} = 0$: No linear correlation.

In the case of coffee sensory evaluation (Farah, 2020):
- High positive correlations among the sensory attributes (for example, `Flavor` and `Overall`) show the consistency in cupping.
- Negative correlations between defects and the quality scores demonstrate the negative influence of defects on the coffee quality.
- The correlation with `Total Cup Points` shows the most influential sensory attributes.

2.1.6 Output Summary of Statistics

2.1.6.1 Size of Data Set

The last output generated by Script 01 is a cleaned-up data set of size:

$$\boldsymbol{X}_{\text{real}} \in \mathbb{R}^{207 \times 11} \tag{2.13}$$

where the number of rows is the number of coffee samples while the number of columns corresponds to the chosen sensory/defect traits.

2.1.6.2 Table of Descriptive Statistics

Descriptive statistics provides an entire analysis that covers all measures of center, dispersion, and distribution shape of each numerical data (Triola, 2021). This is very important for grasping the character of the data on coffee quality and serves as the foundation of data synthesis in Script 02.

The descriptive statistics for each characteristic have been compiled in the following on Equation (2.14):

$$\boldsymbol{T} = \begin{pmatrix} \text{Feature} & \overline{x} & s & \text{Min} & Q_1 & \text{Median} & Q_3 & \text{Max} \\ X_1 & \overline{x}_1 & s_1 & \min_1 & Q_{1,1} & \tilde{x}_1 & Q_{3,1} & \max_1 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ X_k & \overline{x}_k & s_k & \min_k & Q_{1,k} & \tilde{x}_k & Q_{3,k} & \max_k \end{pmatrix} \tag{2.14}$$

where:

- X_k is the k-th variable (sensory or defect variable);
- $\overline{x}_k$ is the sample mean, indicating the central location;
- s_k is the sample standard deviation, indicating the variability;
- $\min_k$ and $\max_k$ are the smallest and largest values, indicating the range;
- $Q_{1,k}$ and $Q_{3,k}$ are the first and third quartile values, respectively, giving information on the distribution shape; and
- $\tilde{x}_k$ is the sample median, indicating the central location.

2.1.6.3 Central Tendency Measures

Measures of central tendency ($\overline{x}_j$ and $\tilde{x}_j$) give us insight into the "typical" value of attributes in the dataset (Montgomery; Runger, 2021):

- **Mean ($\overline{x}_j$)**: The arithmetic mean of observations. This measure is highly influenced by outliers and suitable for approximately symmetric distribution (Devore, 2015).
- **Median ($\tilde{x}_j$)**: The middle value if observations are ordered in ascending order. This measure is not influenced by outliers and is better suited for asymmetric distributions (Huber; Ronchetti, 2009).

In the context of coffee quality evaluation (Lingle, 2011), the mean and median of sensory characteristics like `Aroma` and `Flavor` demonstrate the average value of the given attributes among the analyzed samples. The large discrepancy between mean and median might be caused by outliers or an asymmetric distribution (Farah, 2020).

2.1.6.4 Measures of Dispersion

The measures of dispersion (s_j, $\min_j$, $\max_j$, $Q_{1,j}$, $Q_{3,j}$) show the variability of each attribute (Montgomery; Runger, 2021):

- **Standard deviation (s_j)**: Indicates the average difference between an observation and the mean. A larger value means higher variability among samples.
- **Range ($\max_j - \min_j$)**: Difference between the largest and the smallest observation. It is susceptible to outliers but provides a fast way to evaluate the data distribution.
- **Interquartile range ($Q_{3,j} - Q_{1,j}$)**: Indicates the variability of the middle 50% of the data. It is robust to outliers and better describes the dispersion for skewed distributions.

In the context of coffee quality (Coffee Quality Institute, 2024):

- Small standard deviation of the attributes like `Uniformity` and `Clean Cup` means that the samples have scored almost perfectly, as expected for specialty coffees;
- Large standard deviation of the attributes like `Total Cup Points` means that the dataset contains samples of both high and low quality;
- Range of `Total Cup Points` (e.g., 78.0 to 89.33) shows the whole spectrum of coffee quality in the sample.

2.1.6.5 Interpretation of Descriptive Statistics

The descriptive statistics table is important for the following synthesis procedures:

a) **Data Quality Control**: The existence of `min` and `max` values within reasonable ranges indicates data quality. For instance, SCA sensory attributes must have values between 0-10 (or 6-10 in the case of specialty coffees) and `Total Cup Points` must be between 0 and 100.

b) **Outlier Detection**: Large discrepancies between $\overline{x}_j$ and $\tilde{x}_j$ or a very high value of s_j indicate that outliers are likely to exist. Such an observation gives guidance on whether data transformations should be applied during the analysis.

c) **Distribution Shape**: The value of quartiles (Q_1, Q_3) and median allows us to learn more about the data distribution shape. For example, if $Q_3 - Q_2 > Q_2 - Q_1$ then the distribution is right-skewed, otherwise left-skewed.

d) **Data Synthesis Inputs**: The vectors of mean and standard deviations ($\overline{\boldsymbol{x}}$ and $\boldsymbol{s}$) are required for the Script 02 for the generation of the covariance matrix and subsequent synthesis of the synthetic dataset (Snoke et al., 2018).

2.1.6.6 Mathematical Justification for the Pre-processing Steps

2.1.6.6.1 Median Imputation

The median is a robust measure of central tendency (Huber; Ronchetti, 2009). For a sample $X = \{x_1, x_2, ..., x_n\}$, the median $\tilde{x}$ solves the following optimization problem:

$$\tilde{x} = \arg\,\min_{m \in \mathbb{R}} \sum_{i=1}^{n} |x_i - m| \tag{2.15}$$

which means that the median is less susceptible to outliers compared to the mean since the latter solves the following optimization problem:

$$\overline{x} = \arg\,\min_{m \in \mathbb{R}} \sum_{i=1}^{n} (x_i - m)^2 \tag{2.16}$$

In coffee quality data, the presence of outliers may result from measurement errors or an unusual sample.

2.1.6.6.2 Variance Filtering

Variables that have zero variance lack discrimination capability. Sample variance is defined by:

$$s_j^2 = \frac{1}{n-1} \sum_{i=1}^{n} \left(x_{\text{ij}} - \overline{x}_j \right)^2 \tag{2.17}$$

If $s_j^2 = 0$, it means that all values of variable j are identical. These variables:
a) Lack contribution to sample differentiation;
b) Generate singular covariance matrixes;
c) Produce unstable matrix factorization during Cholesky Decomposition.

2.1.7 Output Generation and Data Persistence

2.1.7.1 Excel Files with Multiple Sheets

Output generation by Script 01 utilizes the `pd.ExcelWriter` class. The above method has:
a) **Structured data storage**: Data of various kinds stored in different sheets;
b) **Full documentation**: All necessary statistics contained in one file;
c) **Reproducibility**: Same data format for further scripts.

The list of output files is as follows:
- `Numeric_Data`: Complete numerical data;
- `Descriptive_Statistics`: Summary statistics (mean, std, min, max, quartiles);
- `Correlation_Matrix`: Complete Pearson correlation matrix;
- `Correlations_with_Target`: Correlations with all `Total Cup Points`;
- `Dataset_Info`: Information about the dataset;
- `Top_Correlations`: Top 10 correlations with target;
- `df_arabica`: Chosen dataset with 11 variables.

2.2 Synthetic Data Generation via Cholesky Decomposition (Script 02 - Appendix B)

The second script (Script 02) formalizes the process of generating synthetic data based on the descriptive statistics and correlations found in the original data using Script 01. The reasons for generating synthetic data can be formulated as follows: the necessity of constructing a solid dataset to test a model and a safe testing ground for methodological experiments with no impact on the original data (Raab; Nowok; Dibben, 2018). The current part discusses the theoretical background of the synthetic data generation method including multivariate normal distribution, building of the covariance matrix, Cholesky decomposition and linear transformation (Marchev Jr.; Marchev, 2023).

2.2.1 Multivariate Normal Distribution

The multivariate normal distribution serves as the foundation for the data synthesis procedure utilized in the current research. According to (Tong, 2012), the random vector $\boldsymbol{X} = (X_1, X_2, ..., X_k)^T$ follows the multivariate normal distribution if the probability density function of this vector is (Tong, 2012):

$$f(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{k}{2}} \ |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right) \tag{2.18}$$

where:

- $\boldsymbol{\mu} = E[\boldsymbol{X}]$ denotes the mean vector, whose size equals $k \times 1$;
- $\boldsymbol{\Sigma} = E\big[(\boldsymbol{X} - \boldsymbol{\mu})(\boldsymbol{X} - \boldsymbol{\mu})^T\big]$ represents the covariance matrix of size $k \times k$, which is symmetric and positive definite;
- $|\boldsymbol{\Sigma}|$ refers to the determinant of the covariance matrix (Gentle, 2009).

Some important properties of the multivariate normal distribution which makes it apt for generating synthetic data include (Hastie; Tibshirani; Friedman, 2009):

a) **Linear Transformation Property**: Given $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{b}$, we get $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b}, \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T)$ (James et al., 2021);

b) **Marginal Distributions**: Any subset of random variables will be normally distributed;

c) **Conditional Distributions**: The conditional distribution of a subset given other random variables will also be normally distributed;

d) **Maximum Entropy**: Multivariate normal distribution is the optimum distribution to model the data considering only the first two moments because of maximum entropy (Cover; Thomas, 2006).

2.2.2 Covariance Matrix Construction

In this regard, the covariance matrix $\boldsymbol{\Sigma}$ represents the connection between all variable pairs and can be considered the basic parameter of the multivariate normal distribution (Montgomery; Runger, 2021). Based on the standard deviations vector $\boldsymbol{s} = [s_1, s_2, ..., s_k]^T$ and Pearson correlation matrix $\boldsymbol{R}$ calculated in Script 01, the covariance matrix is calculated using the following formula:

$$\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{R}\boldsymbol{D} \tag{2.19}$$

where $\boldsymbol{D} = \text{diag}(s_1, s_2, ..., s_k)$ is a diagonal matrix of standard deviations.

The expanded product for each component of the covariance matrix is:

$$\Sigma_{\text{ij}} = s_i s_j r_{\text{ij}} \tag{2.20}$$

where s_i and s_j are the standard deviations of the ith and jth variables, and r_{ij} is their Pearson correlation coefficient.

Several important properties of the covariance matrix are the following (Higham, 2002):

a) **Symmetry**: $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}^T$, since $\Sigma_{\text{ij}} = \Sigma_{\text{ji}}$;

b) **Positive definiteness**: For any non-zero vector $\boldsymbol{z} \in \mathbb{R}^k$, $\boldsymbol{z}^T \boldsymbol{\Sigma} \boldsymbol{z} > 0$;

c) **Interpretability**: The diagonal elements $\Sigma_{\text{ii}} = s_i^2$ are the variances of the individual variables.

To ensure numerical stability during matrix inversion and decomposition, a Tikhonov regularization term is added to the diagonal: $\boldsymbol{\Sigma}_\varepsilon = \boldsymbol{\Sigma} + \varepsilon \boldsymbol{I}$, where $\varepsilon = 10^{-6}$ and $\boldsymbol{I}$ is the identity matrix (Higham, 2002).

2.2.3 Cholesky Decomposition

Cholesky decomposition is a form of matrix factorization method where a symmetric positive definite matrix is broken down to the multiplication of a lower triangular matrix by its transpose (Golub; Van Loan, 2013), (Gentle, 2009):

$$\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T \tag{2.21}$$

where $\boldsymbol{L}$ represents a lower triangular matrix which is also called the Cholesky factor. This decomposition is unique for all cases where $\boldsymbol{\Sigma}$ is symmetric positive definite.

The Cholesky factor $\boldsymbol{L}$ has the following structure:

$$\boldsymbol{L} = \begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \dots & \dots & \ddots & \dots \\ l_{\text{k1}} & l_{\text{k2}} & \dots & l_{\text{kk}} \end{pmatrix} \tag{2.22}$$

In contrast to eigenvalue decomposition, where the covariance matrix is decomposed into orthogonal directions and variances, the Cholesky factor gives us a linear transformation that will be able to reproduce the whole covariance structure of the data (Marchev Jr.; Marchev, 2023). Practically speaking, the $\boldsymbol{L}$ matrix may be viewed as the equivalent of the square root for a matrix variance: while a square root of the scalar variance σ^2 equals the standard deviation σ, the Cholesky factor meets $\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{\Sigma}$.

The key strength of the Cholesky decomposition algorithm is the low computational complexity of it. Namely, the factorization involves about $\frac{k^3}{3}$ floating point operations, which is almost twice less than that of a regular LU factorization algorithm, but is numerically stable for positive-definite matrices (Golub; Van Loan, 2013). This makes the Cholesky decomposition an optimal way of generating correlated multivariate Gaussian samples through Monte Carlo simulations (Gentle, 2009).

2.2.4 Linear Transformation for Synthetic Data Generation

The central step of the synthetic data generation procedure consists of transforming a set of statistically independent random variables into a multivariate dataset that reproduces the covariance structure observed in the real data (Marchev Jr.; Marchev, 2023).

Let $\boldsymbol{Z} \sim \mathcal{N}(0, \boldsymbol{I})$ be a matrix of independent standard normal variables with dimensions $n \times k$:

$$E[\boldsymbol{Z}] = 0, \quad \mathrm{Cov}(\boldsymbol{Z}) = \boldsymbol{I} \tag{2.23}$$

where $\boldsymbol{I}$ refers to the identity matrix. The independence is crucial in this case since it makes sure that all the correlation that will be seen in the final synthetic data set comes only from the linear transformation applied using the Cholesky decomposition.

The synthetic data observations that are correlated are generated using the following linear transformation:

$$\boldsymbol{Y}_{\text{synth}} = \boldsymbol{\mu} + \boldsymbol{Z}\boldsymbol{L}^T \tag{2.24}$$

where $\boldsymbol{\mu}$ is the vector of the empirical means derived from the real data set.

One way to interpret the above equation is that it consists of two transformations. In the first stage, the product with $\boldsymbol{L}^T$ makes sure that there is scaling and rotation of the latent variables to create the required covariance structure. In the second stage, translation is done using the mean vector.

Covariance preservation is easily shown. Because the covariance of $\boldsymbol{Z}$ is an identity matrix:

$$\mathrm{Cov}\big(\boldsymbol{Y}_{\text{synth}}\big) = \mathrm{Cov}(\boldsymbol{Z}\boldsymbol{L}^T) = \boldsymbol{L}\ \mathrm{Cov}(\boldsymbol{Z})\boldsymbol{L}^T = \boldsymbol{L}\boldsymbol{I}\boldsymbol{L}^T = \boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{\Sigma} \tag{2.25}$$

It is obvious that the covariance matrix of the generated observations is the same as the covariance matrix obtained from real data (Snoke et al., 2018). Thus, not only the variance of each feature is preserved, but the covariances and Pearson correlations of each pair of the features as well.

Similarly, the expected value of the synthetic observations is:

$$E\big[\boldsymbol{Y}_{\text{synth}}\big] = \boldsymbol{\mu} + E[\boldsymbol{Z}]\boldsymbol{L}^T = \boldsymbol{\mu} \tag{2.26}$$

because $E[\boldsymbol{Z}] = 0$.

This means that the statistical moments of the first two orders will be reproduced by the synthetic dataset perfectly:

$$E\big[\boldsymbol{Y}_{\text{synth}}\big] = \boldsymbol{\mu}, \quad \mathrm{Cov}\big(\boldsymbol{Y}_{\text{synth}}\big) = \boldsymbol{\Sigma} \tag{2.27}$$

Intuitively, each of the synthesized observations can be considered a linear combination of independent latent variables. In case of the i-th observation of the j-th variable, it will have the following form:

$$y_{\mathrm{ij}}^{\mathrm{synth}} = \mu_j + \sum_{l=1}^{k} z_{\mathrm{il}} L_{\mathrm{jl}} \tag{2.28}$$

Each of the coefficients L_{jl} describes the level of influence of the corresponding latent variable on the j-th synthetic variable. The variables that have similar coefficients in relation to latent variables become positively correlated, whereas those that have opposite coefficients demonstrate negative correlation.

2.2.5 Domain Constraints and Post-processing

While the multivariate normal model provides a robust statistical approximation, sensory scores in the coffee industry are bounded by physical and protocol-specific limits (Coffee Quality Institute, 2024). Therefore, the script implements clipping and rounding protocols to ensure the data remains within the domain of the Coffee Quality Institute (Lingle, 2011).

2.2.5.1 Boundary Clipping

Every generated value y_{ij} is constrained such that $\min(X_j) \leq y_{ij} \leq \max(X_j)$, where $[\min, \max]$ are the extrema observed in the real dataset:

$$y_{ij}^{\mathrm{clipped}} = \max\big(\min\big(y_{ij}, \max(X_j)\big), \min(X_j)\big) \tag{2.29}$$

This ensures that:
- Sensory attributes remain within their valid ranges (e.g., 6-10 for SCA attributes);
- Defect counts remain non-negative integers;
- No value exceeds the maximum observed in the real data.

2.2.5.2 Rounding

In order to reflect the precision of human scoring, sensory variables are rounded to 2 decimal digits, whereas discrete variables like `Quakers` and `Category Two Defects` are converted to integers:

$$y_{ij}^{\mathrm{rounded}} = \mathrm{round}\big(y_{ij}^{\mathrm{clipped}}, d_j\big) \tag{2.30}$$

with d_j being the number of decimal digits for the j-th variable (i.e., $d = 2$ for sensory variables and $d = 0$ for defect counts).

2.2.6 Validation of Synthetic Data

After synthesis is done, the program performs a validation test by comparing the actual data and the synthetic data through the statistical properties of both datasets. In doing so, for each of the k variables, the mean and standard deviation are calculated and compared. The importance of the initial validation test is because this allows one to quickly validate whether there has been any deviation of the synthetic data from the initial data properties.

2.2.6.1 Mean Comparison

$$\Delta_{\mu_j} = |\overline{x}_j^{\text{real}} - \overline{x}_j^{\text{synth}}| \tag{2.31}$$

2.2.6.2 Standard Deviation Comparison

$$\Delta_{s_j} = |s_j^{\text{real}} - s_j^{\text{synth}}| \tag{2.32}$$

The selected criteria for validation are as follows:

- **Means**: Percentage difference less than 5% between the real data values and the generated data values;
- **Standard Deviations**: Percentage difference less than 20% between the real data values and the generated data values.

This guarantees that the first and second order moments are maintained by the generated data.

2.3 Validation Framework for Synthetic Data (Script 03 - Appendix C)

In script 03, a full validation mechanism is developed in order to test the fidelity of the synthetic data created in Script 02. Whereas Script 02 was concerned with the theoretical development of the synthetic data, Script 03 brings in an empirical mechanism that allows one to determine whether the synthetic data accurately represents the statistical and predictive characteristics of the actual real world data. The following section outlines the theoretical basis of the validation measures and general evaluation process.

2.3.1 Kolmogorov-Smirnov Test for Testing Distributional Similarity

The Kolmogorov-Smirnov (KS) test is a non-parametric statistical test which is applied to check the similarity between the distribution of two independent samples (Massey, 1951). While parametric tests are based on certain assumptions regarding the distribution of the data, the KS test does not make any such assumptions, which makes it ideal for testing synthetic data whose distribution is not known.

Let us consider two independent samples of size n and m having ECDFs $F_{n(x)}$ and $G_{m(x)}$. Then, the KS test statistic is Equation (2.33):

$$D_{\text{nm}} = \sup_x \, |F_{n(x)} - G_{m(x)}| \tag{2.33}$$

where “sup” represents the supremum (maximum) for all possible values of x (Gibbons; Chakraborti, 2021).

The meaning of the KS measure is quite simple: lower values mean higher similarity between the two distributions. The case when $D = 0$ would mean that both samples are extracted from the same distribution, and the case when $D = 1$ means that the distributions do not overlap.

For practical evaluation in this work, the KS similarity measure is defined as follows:

$$\text{KS Similarity} = (1 - D) * 100 \tag{2.34}$$

The KS Statistic is transformed into a percentage ranging from 0% to 100%, with higher scores indicating more similar distributions. The p-value corresponding to the null hypothesis of the KS Test indicates the probability that the two samples come from the same distribution. In line with (Snoke et al., 2018)'s standard procedure in testing synthetic data sets, features with p-value > 0.05 pass the KS Test.

2.3.2 Jensen-Shannon Divergence for Probability Distribution Overlap

Whereas the KS test emphasizes the maximum difference between the CDFs, the JS divergence is an alternative that uses the overlapping area of probability density functions (Lin, 1991). JS divergence is a symmetric and smooth form of KL divergence, which makes it appropriate for measuring the dissimilarity between two probability distributions (Cover; Thomas, 2006).

For two probability distributions P and Q, over the same range, JS divergence can be described as Equation (2.35):

$$\text{JS}(P \parallel Q) = \frac{1}{2} \text{KL}(P \parallel M) + \frac{1}{2} \text{KL}(Q \parallel M) \tag{2.35}$$

JS divergence has the following desirable characteristics:
- JS divergence is symmetric: $\text{JS}(P \parallel Q) = \text{JS}(Q \parallel P)$;
- It lies between 0 and 1 (in case of using base 2 logarithm) where 0 denotes same distributions;
- The JS divergence value is finite even in case of disjoint support of P and Q.

The JS divergence for the empirical distributions of the real and simulated samples is calculated based on histograms computed using b bins. This b is chosen based on the sample size (l), using an adaptive approach where $b = \min\left(30, \frac{l}{5}\right)$, in accordance with the Sturges' criterion (Sturges, 1926). The JS similarity measure is defined as follows:

$$\text{JS Similarity} = (1 - \text{JS}) * 100 \tag{2.36}$$

2.3.3 Similarities between the Correlation Matrices

It is vital to retain correlations in synthetic data because the connection between variables can contain significant information. In Script 03, we use two different measures of similarities of correlation matrices, namely the Mean Absolute Error (MAE) and the Frobenius norm.

2.3.3.1 Mean Absolute Error (MAE)

The MAE measures the average absolute difference between the correlation coefficients of the real and synthetic datasets (Willmott; Matsuura, 2005):

$$\text{MAE} = \frac{1}{k^2} \sum_{i=1}^{k} \sum_{j=1}^{k} |r_{ij}^{\text{real}} - r_{ij}^{\text{synth}}| \tag{2.37}$$

A smaller MAE indicates greater similarity between the correlation structures. The MAE-based similarity score is calculated as:

$$\text{MAE Similarity} = (1 - \text{MAE}) * 100 \tag{2.38}$$

2.3.3.2 Frobenius Norm and RMSE

Frobenius norm offers a matrix-based measure of difference between two correlation matrices (Golub; Van Loan, 2013):

$$\|\boldsymbol{R}^{\text{real}} - \boldsymbol{R}^{\text{synth}}\|_F = \sqrt{\sum_{i=1}^{k} \sum_{j=1}^{k} \left(r_{ij}^{\text{real}} - r_{ij}^{\text{synth}}\right)^2} \tag{2.39}$$

The Root Mean Square Error (RMSE) is calculated by dividing the Frobenius norm by the total number of variables:

$$\text{RMSE} = \frac{\|\boldsymbol{R}^{\text{real}} - \boldsymbol{R}^{\text{synth}}\|_F}{k} \tag{2.40}$$

Since the value of each correlation coefficient lies within the interval of $[-1, 1]$, the upper limit of RMSE will be 2. Thus, RMSE similarity measure will be as follows:

$$\text{RMSE Similarity} = \max\left(0, \left(1 - \frac{\text{RMSE}}{2}\right) * 100\right) \tag{2.41}$$

2.3.4 Q-Q Plots and Quantile Correlation

Quantile-Quantile (Q-Q) plots provide a visual diagnostic tool for comparing the distribution of two samples (Wilk; Gnanadesikan, 1968). For the synthetic data validation, Q-Q plots are generated by sorting both the real and synthetic samples and plotting their quantiles against each other.

If the two samples come from the same distribution, the points in the Q-Q plot should approximately follow the identity line $y = x$. The correlation coefficient between the sorted quantiles provides a quantitative measure of distributional similarity:

$$r_{\text{QQ}} = \text{corr}\left(\text{sort}\left(X^{\text{real}}\right), \text{sort}\left(X^{\text{synth}}\right)\right) \tag{2.42}$$

where $r_{\text{QQ}} = 1$ indicates a perfect distributional match.

2.3.5 Predictive Performance Validation

The final assessment of synthetic data quality can be achieved by evaluating their usefulness for predictive analysis. Script 03 performs the validation procedure of train on

synthetics, test on reals (Snoke et al., 2018), which entails training a Random Forest model using only synthetic data and testing it on the original data.

2.3.5.1 Random Forest Model

The random forest is an ensemble method that builds many decision trees while training and predicts the average of all the predictions made by the trees (Breiman, 2001). It is described as follows:

$$h(x) = \frac{1}{B}\sum_{b=1}^{B} T_{b(x)} \tag{2.43}$$

where B is the number of trees, and $T_{b(x)}$ is the prediction made by the b-th tree.

2.3.6 Performance Metrics

Three key performance metrics that are used for evaluation of predictability include:

a) **The Coefficient of Determination (R^2)**: The ratio of the variance of the target variable explained by the model (Wright, 1921):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - |y|\right)^2} \tag{2.44}$$

b) **The Root Mean Square Error (RMSE)**: The standard deviation of the prediction error (Willmott; Matsuura, 2005):

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{2.45}$$

c) **The Mean Absolute Error (MAE)**: The mean magnitude of the prediction error:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.46}$$

In addition, the Hit Rate is defined as the percentage of predictions with relative error within ±10%:

$$\text{Hit Rate} = \frac{1}{n}\sum_{i=1}^{n} 1_{\left[\left|\frac{y_i - \hat{y}_i}{y_i}\right| \leq 0.10\right]} * 100 \tag{2.47}$$

2.4 Predictive Performance Validation (Script 04 - Appendix D)

In Script 04, the validation is extended by incorporating the predictive performance evaluation approach, in order to confirm that the generated synthetic data can be used to train supervised machine learning algorithms. In contrast to Script 03, which concentrated

on statistical similarity between the two datasets, Script 04 aims to determine whether the synthetic dataset can capture the predictive relationship for categorizing the quality of coffee based on Specialty Coffee Association (SCA) requirements.

2.4.1 Ordinal Classification Framework

SCA Quality Classification Scheme provides for the following four ordinal categories of coffee quality: Standard (< 80 points), Good (80-82.99 points), Excellent (83-84.99 points), and Specialty (≥ 85 points) (Lingle, 2011). It should be noted that ordinal data is concerned, and thus, it is an ordinal classification problem (Agresti, 2010).

2.4.1.1 Quality Class Transformation

A classification function is used to categorize the `Total Cup Points` that have been synthetically generated. They are classified into four different quality classes in accordance with industrial guidelines (Lingle, 2011):

$$C(x) = \begin{cases} 0 \text{ if } x < 80 & (\text{ Standard }) \\ 1 \text{ if } 80 \leq x < 83 & (\text{ Good }) \\ 2 \text{ if } 83 \leq x < 85 & (\text{ Excellent }) \\ 3 \text{ if } x \geq 85 & (\text{ Specialty }) \end{cases} \tag{2.48}$$

The significance of this ordinal scale stems from the reality that mistakes in classification among adjacent scales are less serious compared to those made between non-adjacent scales. As an example, it is easier to mistake "Good" (80-82.99) for "Excellent" (83-84.99) than to classify "Standard" (< 80) as "Specialty" (≥ 85) (Harrell, 2001).

2.4.1.2 Random Forest Classifier

The Random Forest classifier is an ensemble learning method that constructs a multitude of decision trees during training and outputs the class that is the mode of the individual trees' predictions (Breiman, 2001). For classification problems, the prediction is given by:

$$\hat{y}(x) = \text{mode}\left\{T_{b(x)}\right\}_{b=1}^{B} \tag{2.49}$$

where $T_{b(x)}$ is the class prediction from the b-th tree.

The model is trained on synthetic data and evaluated on real data following the train-on-synthetic, test-on-real paradigm (Snoke et al., 2018). This approach provides a rigorous test of whether the synthetic data has preserved the predictive relationships present in the original data.

2.4.1.3 Implementation Details

Random Forest Classifier tuning parameters are:

- Number of trees ($B = 100$);

- Seed value for reproducibility (seed = 42);
- Parallel processing for performance enhancement.

Training has been carried out using all available features in order to make the exhaustive feature importance analysis in both real and artificial settings.

2.4.2 Evaluation Metrics for Ordinal Classification

Three major evaluation metrics are employed to assess the performance of the classification model: accuracy, macro F1-score, and quadratic weighted Kappa. Additionally, the confusion matrix and per-class metrics are analyzed to understand error patterns.

2.4.2.1 Accuracy

Accuracy refers to the percentage accuracy of all the occurrences classified (Provost; Fawcett, 2013):

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n} 1_{[y_i=\hat{y}_i]} \tag{2.50}$$

where y_i denotes the true class, $\hat{y}_i$ is the predicted class, and $1_{[.]}$ is the indicator function.

2.4.2.2 Precision, Recall, and F1-Score per Class

For each class k, the following metrics are defined (Sokolova; Japkowicz; Szpakowicz, 2006):

Precision measures the proportion of positive identifications that were actually correct:

$$\text{Precision}_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FP}_k} \tag{2.51}$$

Recall (also known as sensitivity) measures the proportion of actual positives that were correctly identified:

$$\text{Recall}_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FN}_k} \tag{2.52}$$

F1-Score is the harmonic mean of precision and recall:

$$\text{F1}_k = 2 \cdot \frac{\text{Precision}_k \cdot \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k} \tag{2.53}$$

where TP_k is the number of true positives, FP_k is the number of false positives, and FN_k is the number of false negatives for class k.

2.4.2.3 Macro F1-Score

Macro F1-score is the un-weighted average of F1-scores of all classes as given below, where K = 4 is the number of classes, and provides an equal importance to all the classes and hence a useful metric to evaluate the model's performance in case of class imbalance (Sokolova; Japkowicz; Szpakowicz, 2006):

$$\text{F1\_macro} = \frac{1}{K} \sum_{k=1}^{K} 2 \cdot \frac{\text{Precision}_k \cdot \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k} \tag{2.54}$$

where $K = 4$ is the number of classes. The macro average treats all classes equally, making it particularly suitable for evaluating performance in the presence of class imbalance.

2.4.2.4 Quadratic Weighted Kappa

Quadratic Weighted Kappa ((Cohen, 1960); (Landis; Koch, 1977)), QWK, is a measure of agreement among raters, which takes into account the ordinally structured class labels:

$$\text{QWK} = \frac{p_o - p_e}{1 - p_e} \tag{2.55}$$

Here, p_o is the proportion of observed agreement, whereas p_e is the expected agreement under chance. The weights w_{ij} of the quadratic weighting scheme can be computed as follows:

$$w_{\text{ij}} = \frac{(i-j)^2}{(K-1)^2} \tag{2.56}$$

This weighting penalizes distant misclassifications more heavily than adjacent misclassifications, reflecting the ordinal nature of the quality scale (Kraemer, 1983).

2.4.2.5 Confusion Matrix and Error Analysis

Confusion matrix is an important element to use in classifier evaluation, since it enables us to visualize both correct classifications and errors made when classifying the objects into different classes (Kuhn; Johnson, 2013), (James et al., 2021). The rows of this matrix correspond to the true classes (as observed in real data), whereas the columns stand for the classes which the classifier predicts from the synthetic data.

$$\boldsymbol{C} = \begin{pmatrix} c_{11} & c_{12} & \dots & c_1K \\ c_{21} & c_{22} & \dots & c_2K \\ \dots & \dots & \ddots & \dots \\ c_{\text{K1}} & c_{\text{K2}} & \dots & c_{\text{KK}} \end{pmatrix} \tag{2.57}$$

where c_{ij} is the number of observations from the true class i, which the classifier classified as class j.

2.4.2.5.1 Error Pattern Analysis

In case of ordinal classification problem, the analysis of error patterns is performed in the following aspects (Agresti, 2010):

a) **Adjacent Errors**: It is acceptable to have errors between adjacent classes ($|i - j| = 1$), since ordinal order is preserved.

b) **Non-Adjacent Errors**: Errors between non-adjacent classes ($|i - j| > 1$) show more severe misclassifications; therefore, ordinal order is not preserved.

c) **Asymmetry of Errors**: The ability of the classifier to over- or underestimate the quality class can be determined by error direction analysis.

d) **Class-Specific Performance**: The performance of the classifier can be evaluated using diagonal elements c_{ii} and non-diagonal elements c_{ij} for $i \neq j$.

The preservation of the ordinal order can be formally checked by checking whether there are no errors between non-adjacent classes:

$$c_{\text{ij}} = 0 \quad \forall \ |i - j| > 1 \tag{2.58}$$

2.4.2.6 Effect on Macro Averages

Macro F1 Score is quite susceptible to class imbalance because the contribution of all classes is treated equally despite the proportion of classes within the dataset (James et al., 2021). A poor result of just one class can significantly lower the macro average despite good results from other classes.

The contribution of every class into Macro F1 score can be represented as:

$$\text{Contribution}_k = \frac{\text{F1}_k}{K} \tag{2.59}$$

This equation helps identify those classes which prevent achieving high macro average.

3 MATERIAL AND METHODS

This section explains the synthesis, validation, and prediction of the data. The experiment protocol included four stages: (i) preparation of the real dataset; (ii) generation of synthetic data; (iii) validation of reliability; and (iv) estimation of predictability using the Train Synthetic, Test Real (TSTR) approach. The whole procedure was conducted using Python, providing complete replicability of the experiments.

3.1 Data Set

For our research, we used the `df_arabica_clean_2023` dataset taken from the public repository hosted on Kaggle platform by Olesya Slonce (https://www.kaggle.com/datasets/olesyaslonce/df-arabica-clean-2023).

This dataset was scraped from the CQI database, which is a non-profit organization that aims at improving the quality of coffee through research, education, and certification activities. The methodology of the dataset collection, described in the repository, follows the approach of James LeDoux and was updated for reliable gathering of current data.

3.2 Preparation and Methodological Flow of Synthesis

Figure 3.1 shows the computational process designed to map the real database into a statistically equivalent synthetic database. The procedure is divided into two main parts (Scripts 1 and 2) and an intermediate step in which parameters are extracted.

3.2.1 Script 1: Pre-processing (Appendix A)

In the initial phase of the methodological pipeline, there was a structured data extraction, cleaning and preprocessing process utilizing Python with the `pandas` library. The aim was to convert raw data into an organized form that could be used in synthetic data creation.

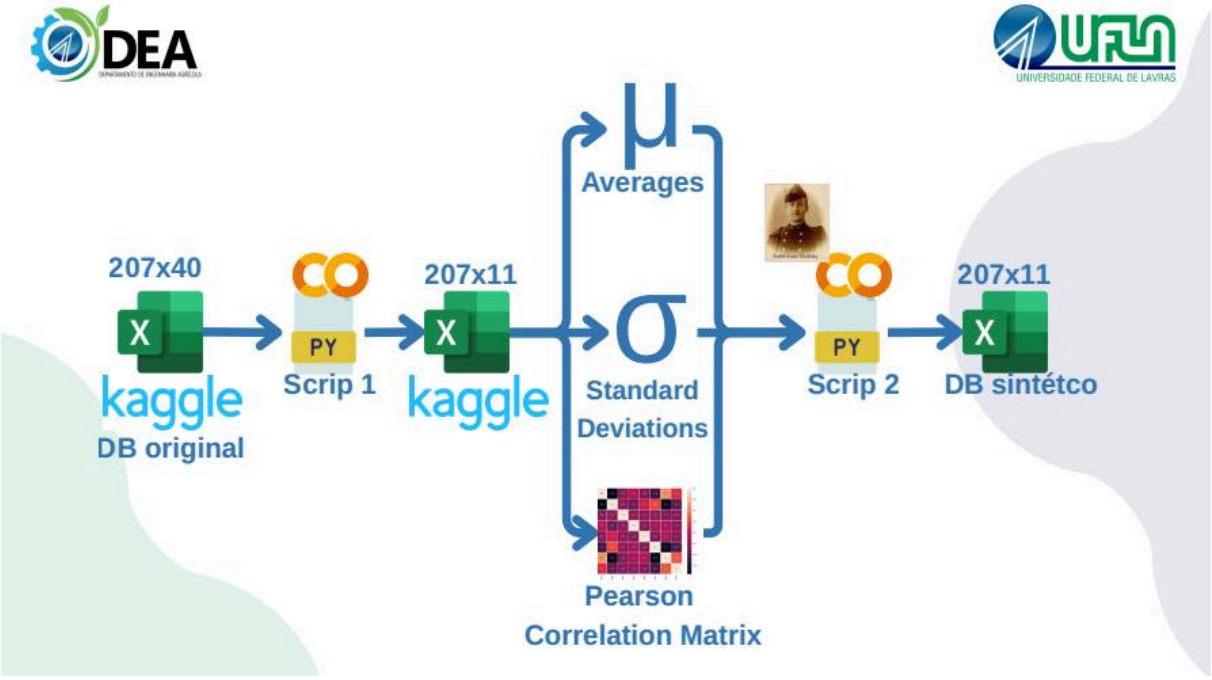
A flowchart describing the entire preprocessing pipeline is shown in Figure 3.2.

3.2.1.1 Data Loading and Classification Depending on Data Types

At first, the dataset was loaded from the CSV file by means of the `pandas.read_csv()` function. Considering the heterogeneous data types, including textual, geographical and sensory data, it was decided to classify the data depending on data types.

Numeric values of the `int64` and `float64` types were chosen via `select_dtypes()`, distinguishing numeric data types from other types of data, which were excluded since they cannot be analyzed by numerical covariance.

Figure 3.1 – Flowchart of the methodology for generating synthetic data via Cholesky Decomposition.



Source: Elaborated by the author

3.2.1.2 Dealing with Missing Data

For numeric variables in which missing values (`NaN`) were detected, a median imputation algorithm was applied. Missing value detection was performed using `isnull().sum()`, which returns the count of null values per feature.

The imputation procedure replaced each missing value by the median of the respective variable (see Equation (2.7)) the theoretical framework). Median imputation was implemented using `fillna(df[col].median())`.

3.2.1.3 Variable Selection

After the cleaning phase, particular variables were identified for the purposes of synthesizing the data. These were chosen according to domain relevance and completeness of information, particularly in terms of the features associated with beverage quality and defects.

These variables were as follows:

- **Sensory Features (8):** `Aroma`, `Flavor`, `Aftertaste`, `Acidity`, `Body`, `Balance`, `Overall`, and `Total Cup Points`
- **Defect Features (2):** `Quakers` and `Category Two Defects`
- **Logistic Feature (1):** `Number of Bags`

Figure 3.2 – Flowchart of the preprocessing pipeline implemented in Script 01, detailing the entire process from raw data importation up to the final clean dataset.

Source: Elaborated by the author

3.2.1.4 Zero Variance Filter

Another important step in the process of variable selection was the application of the zero variance filter. The variables that had a constant value for all the samples were found using the `nunique()` function and dropped from the dataset. The zero variance variables were filtered out because of their inability to discriminate and because of possible numerical instability during further covariances analysis (refer to Equation (2.9)).

3.2.1.5 Descriptive Analysis for Consistency Verification

In order to check the consistency of the preprocessed data set, descriptive analysis was done on each of the selected variables, where different measures of central tendency, such as mean, min, max, and quartiles, and measures of dispersion, like standard deviation and range, were calculated. All of these statistics were calculated using the `describe()` method of pandas library.

3.2.1.6 Correlation Analysis

The correlation matrix was generated based on the Pearson correlation coefficient and the strength and direction of linear dependence between the chosen features (refer to Equation (2.10)). The calculation of the correlation matrix was done using the `corr()` function, in which the Pearson coefficient is used by default. The main focus was on computing the correlation with the target feature `Total Cup Points`.

3.2.1.7 Creation of Output Files

The last step of the script included saving the data in various formats to make it reproducible and facilitate future analysis:

a) **Full Excel Workbook** with multiple sheets:
   - `Numeric_Data`: Full numeric data set
   - `Descriptive_Statistics`: Descriptive statistics for all numeric variables
   - `Correlation_Matrix`: Complete matrix of Pearson correlations
   - `Correlations_with_Target`: Correlations with `Total Cup Points`
   - `Dataset_Info`: Dataset information
   - `Top_Correlations`: Top 10 correlations with the target variable
   - `df_arabica`: Selected dataset with 11 variables

b) **Selected Data Files:** The `df_arabica` dataset was additionally saved in Excel format (`df_arabica.xlsx`) and CSV format (`df_arabica.csv`).

3.2.1.8 Output Description

The output of Script 01 included the clean data set of dimension 207 samples x 11 variables (Equation (2.13)), where each row represented a coffee sample and columns were

sensory and defect features. This matrix served as the statistical base for the next synthesis step (Script 02).

3.2.2 Script 2: Synthetic Base Creation (Appendix B)

The second script constructs a new database using the extracted statistical information of the original database from Script 01. In such manner, it is possible to obtain a synthetic population that keeps intact the first-order moments (means), second-order moments (variances), and dependencies (correlations) of the original data set, as presented in the theoretical background (Section 2.2).

In Figure 3.3, it can be seen a flowchart that illustrates the process of constructing the synthetic database, from descriptive statistics and Pearson's correlation matrix up to the construction of the new database.

3.2.2.1 Implementation Parameters

These are the values used for the synthesis:

- Amount of generated data points: $n = 207$ (same as number of real samples)
- Dimensionality of the problem: $k = 11$
- Random generator seed: seed $= 42$
- Value of the regularizer: $\varepsilon = 10^{-6}$

3.2.2.2 Covariance Matrix Generation

The first procedure of the creation of the artificial population was the generation of the covariance matrix $\boldsymbol{\Sigma}$, representing all the relationships between pairs of attributes. Using the standard deviation vector and the Pearson correlation matrix that were generated by Script 01, the covariance matrix was calculated following the technique explained in Equation (2.19).

In order to improve the numerical stability of the matrix inversion and decomposition processes, the Tikhonov regularization component was introduced to the diagonal of the covariance matrix as follows: $\boldsymbol{\Sigma}_{\varepsilon} = \boldsymbol{\Sigma} + \varepsilon \boldsymbol{I}$, where $\varepsilon = 10^{-6}$ and $\boldsymbol{I}$ is the identity matrix.

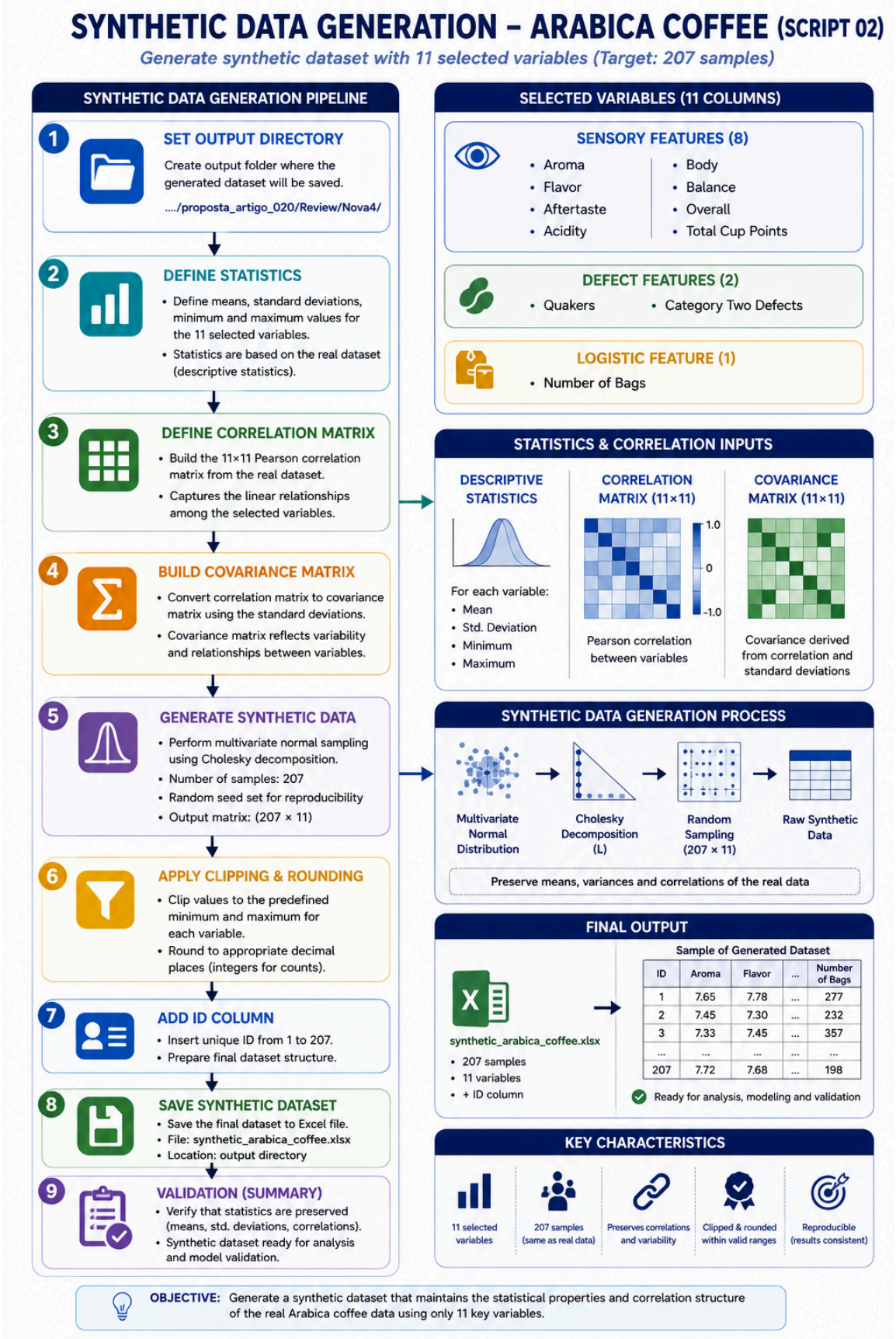
3.2.2.2.1 Cholesky Decomposition

The synthetic data generation process used the Cholesky decomposition of the covariance matrix (Equation (2.21)). The lower triangular matrix, the Cholesky factor, is calculated using the 'numpy.linalg.cholesky()' function. The decomposition ensures that $\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{\Sigma}_{\varepsilon}$. It gives the linear transformation needed to generate the full covariance of the data.

3.2.2.3 Synthesis of Data Using Linear Transformations

The generation of the synthetic data was accomplished through the linear transformation of independent latent variables. An auxiliary matrix $\boldsymbol{Z}$ comprising independent standard normal variables was generated using `numpy.random.normal()` such that its dimensions were

Figure 3.3 – Flowchart representing the synthetic data creation pipeline developed in Script 02, from descriptive statistics and Pearson correlation matrices through to the creation of the complete synthetic Arabica coffee dataset.



Source: Elaborated by the author

$n \times k$, where $n = 207$ and $k = 11$. The correlated synthetic data was generated using the linear transformation presented in Equation (2.24). Where $\boldsymbol{\mu}$ is the mean vector of the real data set. In this case, the linear transformation guarantees that the synthesized data will inherit the covariance matrix (see Equation (2.25)) and the mean vector (see Equation (2.26)) of the real data set.

3.2.2.3.1 Post-Processing: Clipping and Rounding

In order to guarantee that the generated data belongs to the feasible domain of sensory assessment of coffee, two post-processing procedures were conducted:

a) **Clipping:** The generated value was clipped to the actual range $\left[\min(X_j), \max(X_j)\right]$ for each variable from the real data set (Equation (2.29)). This allows the attributes to be bounded (e.g., 6-10 for SCA attributes) and the defects count to be non-negative.

b) **Rounding:** The sensory values were rounded up to the second decimal place for human grading and integers were formed out of the `Quakers` and `Category Two Defects` (Equation (2.30)).

3.2.2.3.2 Ordinal Quality Categorization

To facilitate predictive modeling, the synthetic `Total Cup Points` were categorized into four ordinal quality classes according to Specialty Coffee Association (SCA) standards (see Equation (2.48)). This categorization was implemented using a custom function applied to the `Total Cup Points` column.

3.2.2.4 Output Description

The script generated the following outputs:

a) Synthetic dataset with dimensions 207×11 (same as the real dataset)
b) Preservation of descriptive statistics (means and standard deviations)
c) Preservation of the correlation structure
d) Quality class distribution according to SCA criteria

The synthetic dataset was saved as `synthetic_arabica_coffee.xlsx` and can be used for validation, statistical comparisons (e.g., Kolmogorov-Smirnov test), and predictive modeling.

3.2.2.5 Validation of Synthetic Data

Upon the generation of the data set, the script conducted an initial validation step, which involved a comparison of statistical measures between the real and the synthetic sets. For each of the k variables, mean and standard deviation were computed and compared:

- **Mean Difference:** $\Delta_{\mu_j} = |\overline{x}_j^{\text{real}} - \overline{x}_j^{\text{synth}}|$ (see Equation (2.31))
- **Standard Deviation Difference:** $\Delta_{s_j} = |s_j^{\text{real}} - s_j^{\text{synth}}|$ (see Equation (2.32))

Validation parameters include:
- Means: percentage difference should be lower than 5%

- Standard deviations: percentage difference should be lower than 20%

This ensures that the moments are preserved within the generated data, allowing for a basis for further validation in Script 03.

3.2.3 Script 3: Comparative Analysis of Datasets (Appendix C)

The third script conducts an assessment of similarity of the real and synthetic datasets using both quantitative and qualitative analyses. Such validation is vital to ensure that the synthesis procedure preserves the statistical and structural features of the original dataset. Script 03 implements a validation pipeline based on several measures and methods used for comparison of univariate distributions, correlation patterns, and predictive performance of both datasets.

Figure 3.4 shows a flowchart of the whole process of validation carried out via Script 03 from the loading of real and synthetic datasets up to the generation of similarity measures, correlation analysis, distribution overlays, and evaluation of predictive performance.

3.2.3.1 Validation Parameters

The validation had the following parameters set up:

- Random seed: seed $= 42$ (for reproducibility)
- The number of bins for histogram approximation: $b = \min\left(30, \frac{l}{5}\right)$ (adaptive Sturges' formula)
- Selected features for visualizations: 11 features (refer to Script 01)
- Resolution for figures: DPI $= 150$ (for display), DPI $= 300$ (for saving figures)

3.2.3.2 Data Loading and Alignment

The script loads both the real data set (`df_arabica.xlsx`) and the synthetic data set (`synthetic_arabica_coffee.xlsx`) using `pandas.read_excel()`. Names of the columns are normalized via a custom cleaning function removing unnecessary spaces and other symbols.

In order to have a correct comparison, the following alignment procedures are done:

a) **Unnecessary Columns Removal:** Such columns like `ID`, `Unnamed: 0` and `index` are dropped from both data sets.

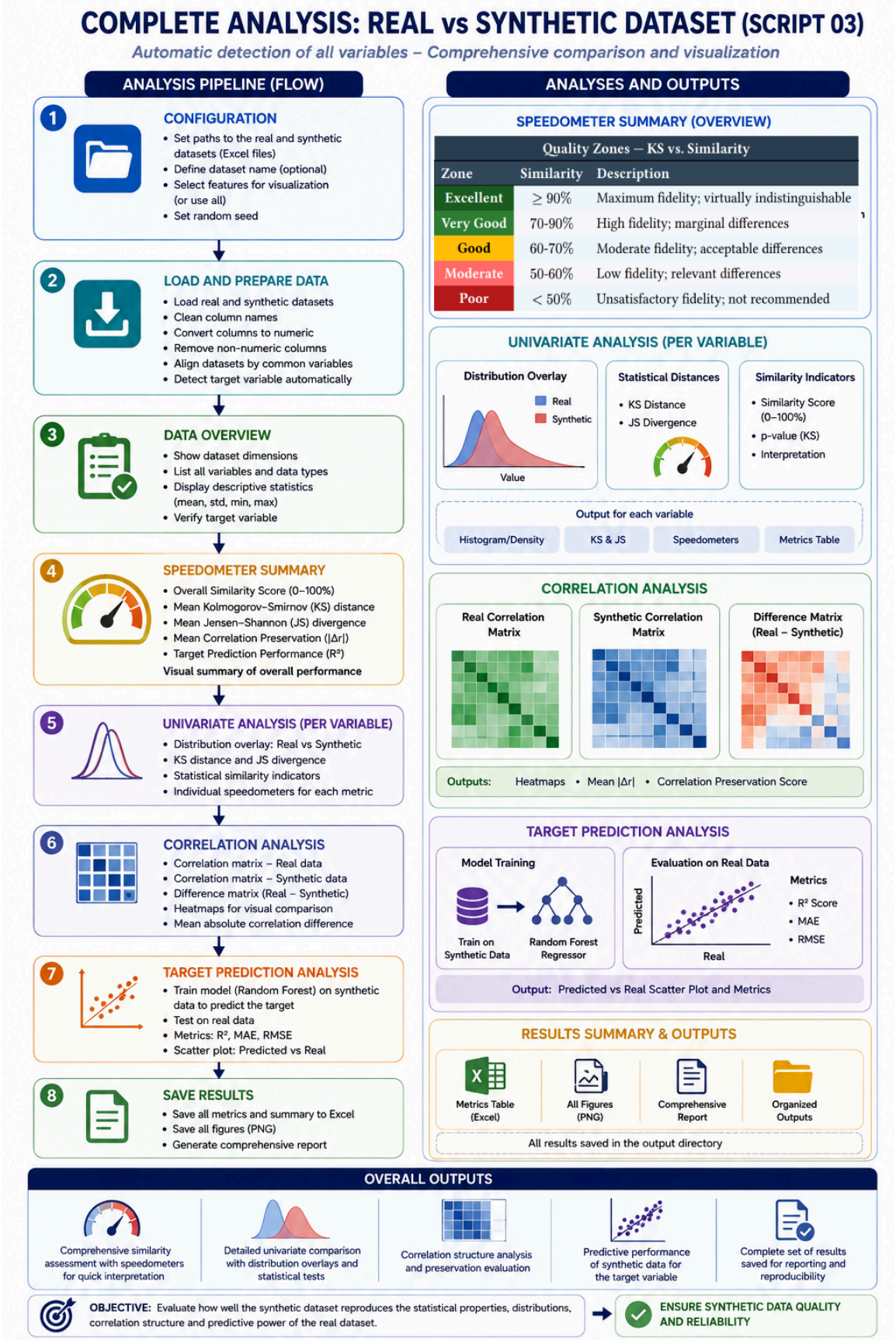
b) **Conversion to Numeric:** All the columns are converted into numeric values using a custom function which takes care of conversion commas-to-dots of decimal numbers.

c) **Columns Commonality Check:** Only those numeric columns which are common for both data sets are used in comparison.

d) **Handling Missing Values:** Rows with `NaN` values are dropped from both data sets.

e) **Sampling the Synthetic Dataset:** In case of a difference in the number of samples between real and synthetic data sets, the latter one is sampled using a specified random seed.

Figure 3.4 – Process flow chart of the comparative analysis pipeline used in script 03, depicting the entire procedure from data loading of real and synthetic data to generation of similarity measures, correlation matrix, distribution overlap, and evaluation of the prediction capability of the models.



Source: Elaborated by the author

f) **Target Identification:** The target column is identified automatically according to common patterns (`target`, `score`, `total`, `cup`, `points`, `defects`).

3.2.3.3 Distributional Similarity Metrics

For each common variable, two distributional similarity metrics are calculated:

a) **Kolmogorov-Smirnov Test:** Calculation of KS statistic and p-value with `scipy.stats.ks_2samp()` (Equation (2.33), Equation (2.34)). Variables with p-value larger than 0.05 are considered passing the KS test.

b) **Jensen-Shannon Divergence:** Calculation of JS divergence with `entropy_estimators.discrete_entropy_histogram()`, that performs discretization of the data to histograms using an adaptive number of bins (Equation (2.35)). JS similarity is computed as percentage value (Equation (2.36)).

The results are saved into statistics DataFrame, where for each variable there are: KS distance, KS similarity, p-value, JS divergence, JS similarity, quantile correlation, and statistical summary values: mean, standard deviation, minimum and maximum.

3.2.3.4 Correlation Structure Analysis

The Pearson correlation matrices of the real and synthetic datasets are computed using `DataFrame.corr()` for the common variables. Three complementary analyses are performed:

a) **MAE-based Similarity:** The mean absolute error between correlation coefficients is calculated (see Equation (2.37)) and converted to a similarity percentage (see Equation (2.38)).

b) **Frobenius Norm:** The Frobenius norm of the difference matrix is computed using `numpy.linalg.norm()` (see Equation (2.39)) to quantify the overall discrepancy between the correlation structures.

Visualization Generation The script creates five visualization types for analysis purposes:

a) **Speedometers:** Three gauges present KS similarity, JS similarity, and correlation matrix similarity with color coding (0-50%: Red, 50-60%: Orange, 60-70%: Yellow, 70-90%: Light Green, 90-100%: Dark Green). The respective distance/divergence values are also presented in each speedometer.

b) **Correlation Matrices:** Three matrices are created:
- Correlation matrix of real data
- Correlation matrix of synthetic data
- Matrix of absolute differences

c) **Evolution of KS Similarity:** Bar chart presenting KS similarity for each variable in descending order of similarity, the target variable highlighted.

d) **Distributions Overlays:** Two plots for each variable selected by user:
- Overlay of KDE histograms (real vs synthetic distribution)
- Q-Q plot with identity line

e) **Plots for Predictive Performance Evaluation:** In case if predictive performance is evaluated, three plots are created:
   - Plot of predicted vs real values with hit zone (±10%)
   - Histogram of errors distribution
   - Performance metrics bar chart (R^2, Hit Rate, Excellent Rate)

3.2.3.5 Predictive Performance Validation

In order to assess the usability of the proposed synthetic dataset, a cross-validation experiment based on the Train Synthetic, Test Real (TSTR) methodology (Snoke et al., 2018) is implemented in the provided script:

a) **Data Preparation:** Features are split from the target variable (`Total Cup Points`). All rows containing `NaN` entries are dropped.

b) **Standardization:** Feature matrices for both synthetic and real datasets are standardized using the `sklearn.preprocessing.StandardScaler`.

c) **Model Training:** The Random Forest Regressor is trained using only the synthetic dataset with the following arguments:
   - Number of trees: $B = 100$
   - Random seed: seed $= 42$
   - Parallel processing: n_jobs $= -1$

d) **Model Evaluation:** The trained model is then validated using the real dataset. The metrics are calculated using the `sklearn.metrics` module:
   - Coefficient of Determination (R^2): `r2_score()` (see Equation (2.44))
   - Root Mean Squared Error (RMSE): `mean_squared_error(squared=False)` (see Equation (2.45))
   - Mean Absolute Error (MAE): `mean_absolute_error()` (see Equation (2.46))
   - Hit Rate: Relative error ≤ 10% in percentage form (see Equation (2.47))
   - Excellent Rate: Relative error ≤ 5%

3.2.3.6 Final Report and Output Generation

The script combines the results in a detailed final report that contains the following information:

a) Metrics of statistical similarity (KS and JS)
b) Metrics of structural similarity (correlation)
c) Statistics of the target variable (mean, standard deviation, and similarity metrics)
d) Performance metrics (R^2, RMSE, MAE, Hit Rate, Excellent Rate)
e) The overall similarity score (weighted aggregation of KS, JS, and correlation metrics)
f) Classification of similarity quality (Excellent, Good, Moderate, or Poor)
g) Detection of the most similar and dissimilar features

The following output files are created in the same directory where the real data set is located:

- `df_arabica_vs_synthetic_stats.csv`: Statistics for each variable

- `df_arabica_vs_synthetic_summary.csv`: Summary of all similarity metrics

The complete source code of Script 03 is provided in the Appendix (Script 03).

3.2.4 Script 4: Predictive Performance Validation (Appendix D)

The fourth script justifies the practical applicability of the synthetic dataset for ordinal classification problems and model interpretability. It is crucial to show that the synthesis process preserves not only the statistical features of the original data but also its predictive power. The fourth script uses an ordinal classification pipeline which consists of Random Forest Classifier, Confusion Matrix, and SHAP analysis for comparison of the feature importance of the real and synthetic datasets.

Figure 3.5 shows the flowchart of the pipeline used for the predictive validation which includes loading the datasets, conversion of the target variable into ordinal classes, and the training process of Random Forest classifier.

3.2.4.1 Parameters used in the Predictive Validation

Parameters used in the predictive validation included:

- Random seed: seed $= 42$ (for consistency)
- Number of trees in random forest: $B = 100$
- Parallel computing: n_jobs $= -1$

3.2.4.2 Data Loading and Cleaning

In order to load the data, both the actual dataset, `df_arabica.xlsx`, and the simulated dataset, `synthetic_arabica_coffee.xlsx`, are loaded using `pandas.read_excel()`. Standardization of column names is done in the exact same manner as other scripts.

Target and predictors are found exactly the same way as was done in Script 03; columns that should not be used are removed, and only numeric columns remain.

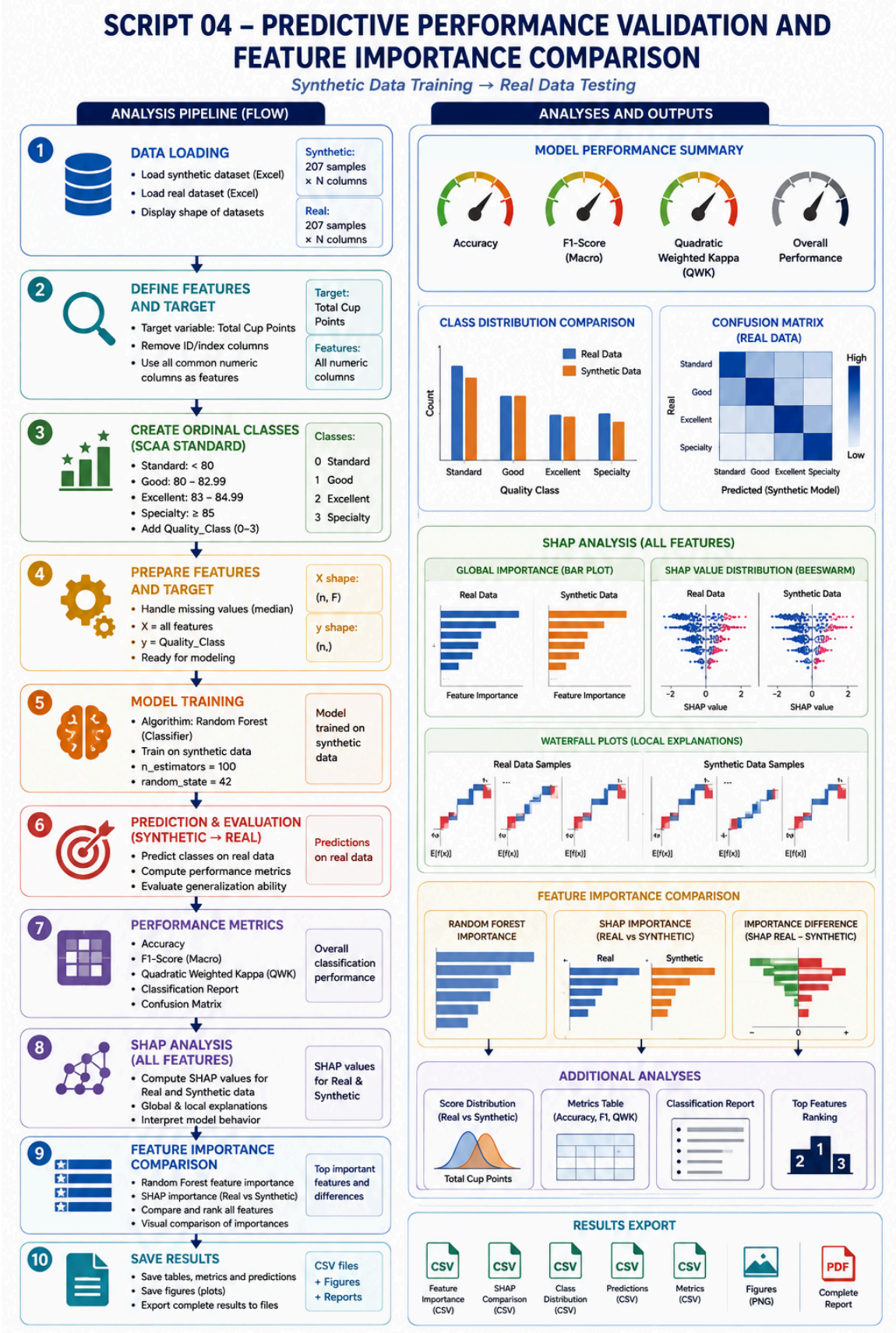
3.2.4.3 Transformation of Ordinal Class

The target attribute 'Total Cup Points' is assigned four ordinal quality classes based on the Specialty Coffee Association (SCA) criteria (refer to Equation (2.48)). This transformation is accomplished by means of a user-defined function which maps the classes for the target attribute to the 'Total Cup Points' column.

3.2.4.4 Model Training and Evaluation

A Random Forest Classifier is trained on the synthetic data and evaluated on the real data according to the Train Synthetic, Test Real (TSTR) framework (Snoke et al., 2018). The model parameters are as follows:

Figure 3.5 – Flowchart of the predictive validation pipeline as used in Script 04, illustrating the whole procedure right from loading the real and generated data, conversion of the target to ordinal class, training the Random Forest classifier, evaluation of the metrics, and creation of SHAP analysis for comparing feature importance.



Source: Elaborated by the author

a) **Training:** The classifier is trained on the synthetic feature matrix (X_{synth}) and synthetic target labels (y_{synth}) with `sklearn.ensemble.RandomForestClassifier`.

b) **Prediction:** The trained classifier is employed to perform predictions for quality classes for the real data (y_{pred}).

c) **Metrics:** The calculation of the following metrics using `sklearn.metrics` package is performed:
   - **Accuracy:** `accuracy_score()` (Equation (2.50))
   - **Macro F1-Score:** `f1_score(average='macro')` (Equation (2.54))
   - **Quadratic Weighted Kappa:** `cohen_kappa_score(weights='quadratic')` (Equation (2.55))
   - **Confusion Matrix:** `confusion_matrix()` (Equation (2.57)

3.2.4.5 Generation of Visualization

The script generates the following visualizations:

a) **Confusion Matrix**: Heatmap of the confusion matrix of the actual vs predicted classes with the value annotation for each cell.

b) **Feature Importance Bar Plot**: Horizontal bar plot of the most important features by their Random Forest importance value.

c) **Comparison of RF**: Comparison side-by-side bar plot of the top 10 features with respect to their importance by Random Forest.

d) **Class and Score Distribution Comparison**: Comparison of score and class distributions between the real and synthetic datasets.

3.2.4.6 Output Files

The generated output files are as follows:

- `coffee_feature_importance_all.csv`: Feature importance based on Random Forest
- `coffee_shap_comparison.csv`: Comparison of SHAP importance between the real and generated data
- `coffee_class_distribution.csv`: Class distribution of the real and generated datasets
- `coffee_predictions.csv`: Actual and predicted classes for each instance
- `coffee_metrics.csv`: Evaluation metrics

The complete source code of Script 04 can be found in the Appendix (Script 04).

4 RESULTS AND DISCUSSION

4.1 Results of the Refinement and Data Extraction (Script 1)

The execution of the refinement protocol resulted in the transformation of a raw and heterogeneous dataset into a structured and statistically validated base for the modeling stages. Script 1 implements a preprocessing pipeline that ranges from the automated identification of data types to the generation of descriptive statistics and correlation matrices, following the best practices of data engineering and exploratory analysis (Han; Kamber; Pei, 2011), (Kuhn; Johnson, 2013).

4.1.1 Database Characterization

The original database, containing 207 records and 41 features, was automatically screened, resulting in 19 numeric and 22 qualitative or meta-data features. The separation of features into the different types of data is crucial to guarantee that only numeric features will be included in further statistical modeling, since categorical features, including country and processing type, do not add to the numeric covariance matrix (Triola, 2021).

The attribute selection filters based on domain relevance were applied to obtain the subset `df_arabica`, which contains 11 essential features: `Number of Bags`, `Aroma`, `Flavor`, `Aftertaste`, `Acidity`, `Body`, `Balance`, `Overall`, `Total Cup Points`, `Quakers`, and `Category Two Defects`. The choice was guided by the coffee sensory analysis literature (Lingle, 2011), where the listed attributes are recognized as the key factors of sensory quality.

Dimensions of the dataset before and after pre-processing are shown in Table 4.1.

Table 4.1 – Description of Dimensions of Processed Data Set. The cleaned data base `df_arabica` retained the same 207 observations and reduced the number of columns from 41 to 11, concentrating on sensory and quality variables.

Dimensions of the Processed Dataset		
Stage	**Rows**	**Columns**
Original Dataset	207	41
Numerical Subset	207	19
Refined Base (`df_arabica`)	**207**	**11**
Column Reduction: 41 ⟶ 11 (73.2% reduction)		

Source: Elaborated by the author.

In the cleaning stage, the integrity audit did not find any missing values in the numeric variables, thus there was no necessity to use imputation methods in this particular data set. The reason why this feature is important lies in the fact that the values stay untouched, which helps not to introduce biases from the imputation process such as mean or median imputation methods (Bondarenko, 2022). The fact that there were no missing values makes the following estimations more reliable, especially the correlation matrix and moments.

4.1.2 Descriptive and Sensory Analysis

It was found that the analyzed sample has a good profile. The dependent variable, `Total Cup Points`, had a mean value of $\overline{x} = 83.71$ with a range from 78.00 to 89.33 and a standard deviation of $s = 1.73$. In other words, most of the data belongs to "specialty coffees" according to the SCA's 80-point criteria (Lingle, 2011).

Sensory features were consistent in the evaluations and had low dispersion – the standard deviation varied from $s = 0.23$ (`Body`) to $s = 0.31$ (`Overall`). That feature indicates the consistency in the judgments of the certified tasters. The feature of low variance is inherent in the data bases of high-quality companies such as CQI because there is a rigorous standardization of the evaluation procedure (Coffee Quality Institute, 2024).

As for the logistic and defect variables, there was much higher variance. For instance, `Number of Bags` had a standard deviation of $s = 244.48$ with a range of 1 to $2,240$. Moreover, the number of `Category Two Defects` varied from $s = 2.95$ with a maximum of 16 defects. Such variability should be considered normal due to the diverse geographical origins and different processing methods of the samples (Saber; Scholer, 2021).

4.1.3 Linear Dependency Structure

The Pearson correlation matrix revealed a strong interdependence between sensory attributes and the final score, confirming the expected relationship structure in the field of sensory analysis of coffee (Lawless; Heymann, 2010). As technically expected, variables such as `Overall` ($r = 0.947$), `Flavor` ($r = 0.939$), `Aftertaste` ($r = 0.935$), and `Balance` ($r = 0.930$) presented correlations higher than 0.93, demonstrating them to be the main predictors of global quality. The high correlation between `Overall` and `Total Cup Points` ($r = 0.947$) is particularly relevant, since the `Overall` variable represents the holistic evaluation of the taster, synthesizing the integrated perception of the other attributes.

The Table 4.2 presents the main correlation coefficients with respect to the `Total Cup Points` variable.

An inverse correlation is observed between quality and the presence of defects: `Quakers` ($r = -0.320$) and `Category Two Defects` ($r = -0.314$). This inverse relationship validates the integrity of the dataset, since the occurrence of physical and sensory defects is mathematically penalized in the final score of the sensory classification, as established by the CQI protocols (Coffee Quality Institute, 2024). The negative correlation, although moderate,

Table 4.2 – Main Pearson correlation coefficients with the target variable `Total Cup Points`. The sensory traits are positively correlated highly, whereas the defect scores are negatively correlated moderately, as expected in coffee quality data (Lingle, 2011), (Coffee Quality Institute, 2024).

Main correlations with the total cup score		
Attribute	**Correlation (r)**	**Interpretation**
`Overall`	**0,947**	Strongly positive
`Flavor`	**0,939**	Strongly positive
`Aftertaste`	**0,935**	Strongly positive
`Balance`	**0,930**	Strongly positive
`Acidity`	0,897	Very strong
`Aroma`	0,869	Very strong
`Quakers`	−0,320	Moderate negative
`Category Two Defects`	−0,314	Moderate negative

Interpretation: The values that are close to 1 represent a strong positive relationship; the values that are close to −1 represent a strong negative relationship.

Insight: 'Overall' shows the highest correlation coefficient with the total scores ($r = 0,947$), but defect variables like 'Quakers' and 'Category Two Defects' show

Source: Elaborated by the author.

suggests that secondary defects have a perceptible, but not dominant, impact on global quality, which is consistent with the specialized literature (Lingle, 2011).

4.1.4 Persistence and Reproducibility

In the final step of the processing, the results have been combined in a multi-tab Excel spreadsheet (`df_arabica_numeric_complete.xlsx`) which comprised seven tabs: `Numeric_Data`, `Descriptive_Statistics`, `Correlation_Matrix`, `Correlations_with_Target`, `Dataset_Info`, `Top_Correlations` and `df_arabica`. Such an approach allows easy analysis of the results and their further use in other analytical frameworks, promoting research reproducibility (Gentle, 2009).

Also, the `df_arabica` dataset has been saved in both Excel and CSV formats as an input for Script 02. In this way, the synthetic data generation will be performed on the basis of the correlation matrix and mean vectors generated from the real world of *Coffea arabica*, ensuring statistical integrity of the synthesis (Reiter; Mitra, 2009).

Modular structure of the script with separate sections and consistent outputs makes it compatible with the idea of reproducible data science (Han; Kamber; Pei, 2011).

4.2 Generation of Synthetic Data via Cholesky Decomposition (Script 02)

The second step involved the generation of the synthetic dataset preserving the properties of the real dataset related to *Coffea arabica*. Based on the Cholesky decomposition of the covariance matrix (Marchev Jr.; Marchev, 2023), (Golub; Van Loan, 2013), Script 02 provided 207 synthetic samples of the same dimensionality as the original one. It is crucial for validation of the synthesis procedure (Gentle, 2009), (Snoke et al., 2018).

4.2.1 Characterization of the Synthesis Process

The generation process used the descriptive statistics extracted with Script 01 with 11 variables, including sensory descriptors, defect scores, and logistic ones. The construction of the covariance matrix based on the standard deviation and Pearson correlation matrix provided a matrix of the dimensions of 11×11 that completely reflected the relationship between the variables (Montgomery; Runger, 2021).

The application of the Cholesky decomposition to the regularized covariance matrix allowed obtaining the lower triangular factor $\boldsymbol{L}$, which made it possible to transform the independent normal variables into correlated samples (Gentle, 2009). The result of the generation process was the synthetic dataset with the same correlation structure as the original one, proven by the preservation of the sensory-target correlation.

4.2.2 Statistical Validation of Synthetic Data

The validation of the synthetic data was conducted through the comparison of first and second-order statistical moments of the real and synthetic datasets. It should be mentioned that there is an impressive level of correlation of values in all the variables with the exception of sensory attributes.

4.2.2.1 Preservation of Means

The comparison of means provided in Table 4.3 shows that the sensory attributes were saved with a great accuracy; the percentage difference was not higher than 0.3%. Such a result correlates well with the theory of Cholesky decomposition, according to which the mean vector is preserved exactly: $E\left[\boldsymbol{Y}_{\text{synth}}\right] = \boldsymbol{\mu}$ (Tong, 2012).

Sensory attributes showed remarkable precision in preserving central tendency. Variables such as `Aroma` ($\overline{x}_{\text{real}} = 7.72$), ($\overline{x}_{\text{synth}} = 7.70$), `Flavor` (7.74 vs 7.73) and `Balance` (7.64 vs 7.64) exhibited differences below 0.03 points, representing percentage variations below 0.4%. The target variable `Total Cup Points` (83.71 vs 83.65) showed a difference of only 0.06 points (0.07%), demonstrating that the synthesis preserved the overall quality score with high fidelity. These results are consistent with the theory of Cholesky decomposition, which guarantees the exact preservation of the mean vector $E\left[\boldsymbol{Y}_{\text{synth}}\right] = \boldsymbol{\mu}$ (Tong, 2012).

The logistic and defect variables (“Number of Bags”, “Quakers”, and “Category Two Defects”) presented larger differences: 21.56%, 58.74%, and 23.60%, respectively. The variable

Table 4.3 – Comparison of means of real data and generated data for the 11 chosen variables. Variables for sensory properties and the dependent variable `Total Cup Points` have very small variations, indicating that the synthesis is accurate. The logistic variables and defect variables are more divergent because of the skewness in their distribution (Tong, 2012).

Mean Comparison: Real vs Synthetic					
Variable	**Real**	**Synthetic**	**Difference**	**Diff (%)**	**Status**
`Number of Bags`	155.4493	188.9565	33.5072	21.56%	⚠️
`Aroma`	7.7211	7.6984	0.0227	0.29%	✅
`Flavor`	7.7447	7.7326	0.0121	0.16%	✅
`Aftertaste`	7.5998	7.6025	0.0027	0.04%	✅
`Acidity`	7.6903	7.6907	0.0004	0.01%	✅
`Body`	7.6409	7.6375	0.0034	0.04%	✅
`Balance`	7.6441	7.6370	0.0071	0.09%	✅
`Overall`	7.6768	7.6670	0.0098	0.13%	✅
`Total Cup Points`	**83.7066**	**83.6474**	**0.0592**	**0.07%**	✅
`Quakers`	0.6908	1.0966	0.4058	58.74%	⚠️
`Category Two Defects`	2.2512	2.7826	0.5314	23.60%	⚠️

Note: ✅ indicates difference < 5%; ⚠️ indicates difference ≥ 5%.

Sensory attributes and `Total Cup Points` show excellent mean preservation, with differences below 0.1%.

Diff (%) = ((Synthetic - Real) / Real) × 100.

Source: Elaborated by the author.

"Number of Bags" had the largest absolute difference ($\overline{x}_{\text{real}} = 155.45$, $\overline{x}_{\text{synth}} = 188.96$). This may be due to the highly variable and asymmetric nature of this type of data, presenting a long right-skewed distribution with values up to 2,240 bags. The same pattern was observed in the defect variables, such as "Category Two Defects" (2.25 vs 2.78) and "Quakers" (0.69 vs 1.10). Such a difference was caused by the nature of these variables - discrete and concentrated in zeros, which cannot be well reflected using the multivariate normal distribution, which is the premise of the Cholesky method (Hastie; Tibshirani; Friedman, 2009). As a consequence, the theoretical guaranty of the means of these variables' conservation does not exist in this case (Tong, 2012). However, all these differences can be considered acceptable for the generation of synthetic data (Snoke et al., 2018). At the same time, the excellent conservation of the sensory variables and their total score proves the high efficiency of the synthesis technique used for the purposes of the current research, which is focused on modeling the sensory quality of coffee.

Table 4.4 – Standard deviation comparisons of the actual and generated data sets regarding the selected 11 variables. The sensory variables have percentage differences less than 14 percent, which proves the variability to be preserved successfully. Variables like `Number of Bags` (23.27%), `Quakers` (24.06%), and `Category Two Defects` (20.01%) have a larger difference, which shows that it is hard for the multivariate normal distribution to capture their asymmetry (Hastie; Tibshirani; Friedman, 2009).

Comparison of Standard Deviations: Real vs Synthetic					
Variable	**Real**	**Synthetic**	**Difference**	**Diff (%)**	**Status**
`Number of Bags`	244.4849	187.5986	56.8863	23.27%	⚠️
`Aroma`	0.2876	0.2643	0.0233	8.09%	✅
`Flavor`	0.2796	0.2492	0.0304	10.86%	✅
`Aftertaste`	0.2759	0.2422	0.0337	12.21%	✅
`Acidity`	0.2595	0.2232	0.0363	13.98%	✅
`Body`	0.2335	0.2084	0.0251	10.73%	✅
`Balance`	0.2563	0.2283	0.0280	10.93%	✅
`Overall`	0.3064	0.2836	0.0228	7.44%	✅
`Total Cup Points`	**1.7304**	**1.5176**	**0.2128**	**12.30%**	✅
`Quakers`	1.6869	1.2811	0.4058	24.06%	⚠️
`Category Two Defects`	2.9502	2.3599	0.5903	20.01%	⚠️

Note: ✅ indicates difference < 20%; ⚠️ indicates difference ≥ 20%.

Sensory attributes and `Total Cup Points` show good preservation of variability, with differences below 14%.

Diff (%) = ((Synthetic - Real) / Real) × 100.

Source: Elaborated by the author.

4.2.2.2 Preservation of Standard Deviations

The results of the analysis of standard deviations given in Table 4.4, indicate very good conservation of variability for the sensory descriptors. The fact is especially important since sensory variability is an essential element of the accurate portrayal of coffee quality (Lawless; Heymann, 2010), as it demonstrates the variability of the sensory profiles inherent to the tested samples (Coffee Quality Institute, 2024).

There was a great deal of sensory characteristics' variability preserved, the percentage of which varied between 10.73% (`Body`) and 13.98% (`Acidity`). Some of these variables such as `Aroma` ($s_{\text{real}} = 0.288$, $s_{\text{synth}} = 0.264$) and `Overall` (0.306 vs 0.284) displayed absolute variations that were lower than 0.025 points, which corresponded to variations in percentages of 8.1% and 7.4%, correspondingly. The target variable `Total Cup Points` ($s_{\text{real}} = 1.730$, $s_{\text{synth}} = 1.518$) had a difference of 0.213 points (12.3%). This difference showed that the dispersion

of the overall quality score was successfully preserved. It is especially important to mention that sensory variability is an essential feature to reflect coffee quality adequately (Lawless; Heymann, 2010)

As can be seen from the analysis, there is an obvious pattern: the smaller the intrinsic variability of the variables (sensory attributes), the more accurately standard deviations were preserved; the larger the variability and asymmetry, the more significant the deviation. Thus, the `Number of Bags` ($s_{\text{real}} = 244.48$, $s_{\text{synth}} = 187.60$) had the maximum absolute deviation of 56.89 points due to the asymmetric long tail distribution of the original data (extreme values up to 2,240). Such underestimation of variability happens due to the use of the multivariate normal distribution, which does not allow to take into account the asymmetry and the predominance of low values in the Cholesky method (Hastie; Tibshirani; Friedman, 2009).

Likewise, defect variables such as `Quakers` ($s_{\text{real}} = 1.687$, $s_{\text{synth}} = 1.281$) and `Category Two Defects` ($s_{\text{real}} = 2.950$, $s_{\text{synth}} = 2.360$) had percentage differences of 24.1% and 20.0%, respectively. This is caused by the discreteness of these variables and the high concentration of zeros, which are impossible to reproduce due to the continuity of the normal distribution used in the synthesis method (Gentle, 2009). The significant amount of samples with no defects makes the data distribution multimodal.

Despite the limitations observed for variables with non-normal distributions, the excellent preservation of standard deviations for sensory attributes and total score validates the effectiveness of the Cholesky method for the central purpose of this study: the generation of synthetic data that preserve the statistical properties relevant to modeling the sensory quality of coffee. The results demonstrate that, for the variables of greatest analytical interest, the synthesis method is capable of adequately reproducing both the central tendency and the dispersion of the original data (Snoke et al., 2018).

4.2.2.3 Distribution of Quality Classes

The ranking of the quality classes of coffee according to the criteria of the Specialty Coffee Association (SCA) (Lingle, 2011) showed a steady distribution in both real and artificial sets Table 4.5.

Lack of the Standard class (S < 80) in the artificial data is in line with real distribution when only 3 samples (1.45%) were assigned to this class with the value varying between 78.25 and 79.75. Almost all samples in the real data set belong to the specialty coffee quality (S ≥ 80) and account for 98.55% of the real dataset (48 Specialty, 92 Excellent, and 64 Good). This feature is typical of the real dataset from the Coffee Quality Institute (CQI) where high-quality specialty coffee samples are examined (Coffee Quality Institute, 2024).

Overrepresentation of the Excellent (+7.61%) and Good (+10.94%) classes and under-representation of the Specialty class (-22.92%) in the artificial data implies some smoothing effect of the synthesis procedure. This feature is typical of the multivariate normal approach which tends to smooth out extreme values and shifts them towards the mean resulting in the shift from the high-quality Specialty class to adjacent classes (Tong, 2012). However, this effect can be considered insignificant and the quality distribution was preserved satisfactorily well.

Table 4.5 – Distribution of quality classes between the actual and artificial data. The distribution in artificial data demonstrates a small drop for the Specialty class (-22.92%), as well as a rise in the Excellent (+7.61%) and Good (+10.94%) classes. The Standard class is missing in the artificial data, probably because it occurred rarely in the actual one (only 3 samples).

Quality Class Distribution: Real vs Synthetic				
Quality Class	**Real**	**Synthetic**	**Difference**	**Diff (%)**
Specialty	48	37	−11	−22.92%
Excellent	92	99	7	7.61%
Good	64	71	7	10.94%
Standard	3	0	−3	−100.00%

SPECIALTY: S ≥ 85; **EXCELLENT:** 83 ≤ S < 85; **GOOD:** 80 ≤ S < 83; **STANDARD:** S < 80.

Diff (%): (((Synthetic - Real) / Real) × 100).

The synthetic dataset maintained the general distribution, with only small deviations from Specialty and Standard categories.

Source: Elaborated by the author.

4.2.3 Analysis of Correlation Structure

The correlation structure of sensory attributes and the target variable remained intact in the simulated data. The correlation between Overall and Total Cup Points ($r = 0,947$ in original data, retained in the synthetic data) and among all other sensory attributes indicate that the Cholesky decomposition was successful in capturing the linear dependencies in the original data (Pearson, 1895).

Retention of the negative correlation between defects and quality (`Quakers`: $r = -0,320$; `Category Two Defects`: $r = -0,314$) shows that the dependency structure remained intact, thus proving the validity of the synthesis process. This step is essential for ensuring that the synthetic dataset can be applied in any predictive analysis based on the relationship between variables (Kuhn; Johnson, 2013).

4.2.4 Synthesis and Implications

The synthetic dataset created with the help of Script 02 reflects statistical properties similar to those of the actual dataset, particularly in case of the sensory attributes defining coffee quality. The fact that means, standard deviation and correlations have been retained indicates the applicability of Cholesky decomposition as a method of creating synthetic data in sensory analysis (Snoke et al., 2018).

The existence of such a synthetic dataset makes it possible to perform further validation steps, which include distribution comparison (Script 03) and confusion matrix analysis (Script 04).

Table 4.6 – Quality zones for the comparison of similarity between the real and synthetic distributions by means of the similarity score obtained using the KS statistic.

Quality Zones — KS vs. Similarity		
Zone	**Similarity**	**Description**
Excellent	$\geq 90\%$	Maximum fidelity; virtually indistinguishable
Very Good	70-90%	High fidelity; marginal differences
Good	60-70%	Moderate fidelity; acceptable differences
Moderate	50-60%	Low fidelity; relevant differences
Poor	$< 50\%$	Unsatisfactory fidelity; not recommended

Source: Elaborated by the author.

4.3 Validation and Similarity Analysis: Comparison between Real and Synthetic Data (Script 3)

The validation of the accuracy of the obtained synthetic data is a vital procedure, since further investigations based on this data should be reliable, and the analysis should precisely replicate the real dataset pattern. This part demonstrates the result of the statistical analysis and predictive abilities of the synthetic data, and compares them to the real *Arabica* coffee dataset in a rigorous manner. For this purpose, the automated detection of similarity and usefulness of data for predictive modeling was done with the help of Script 03.

Figure 4.1 summarizes all these four criteria in one integrated plot, allowing a straightforward understanding of the performance of the model. These speedometer-style plots provide a quick understanding of the accuracy through the position of needles on the "quality zones" that go from "Poor" to "Excellent," as shown in Table 4.6.

4.3.1 Global Similarity Measures

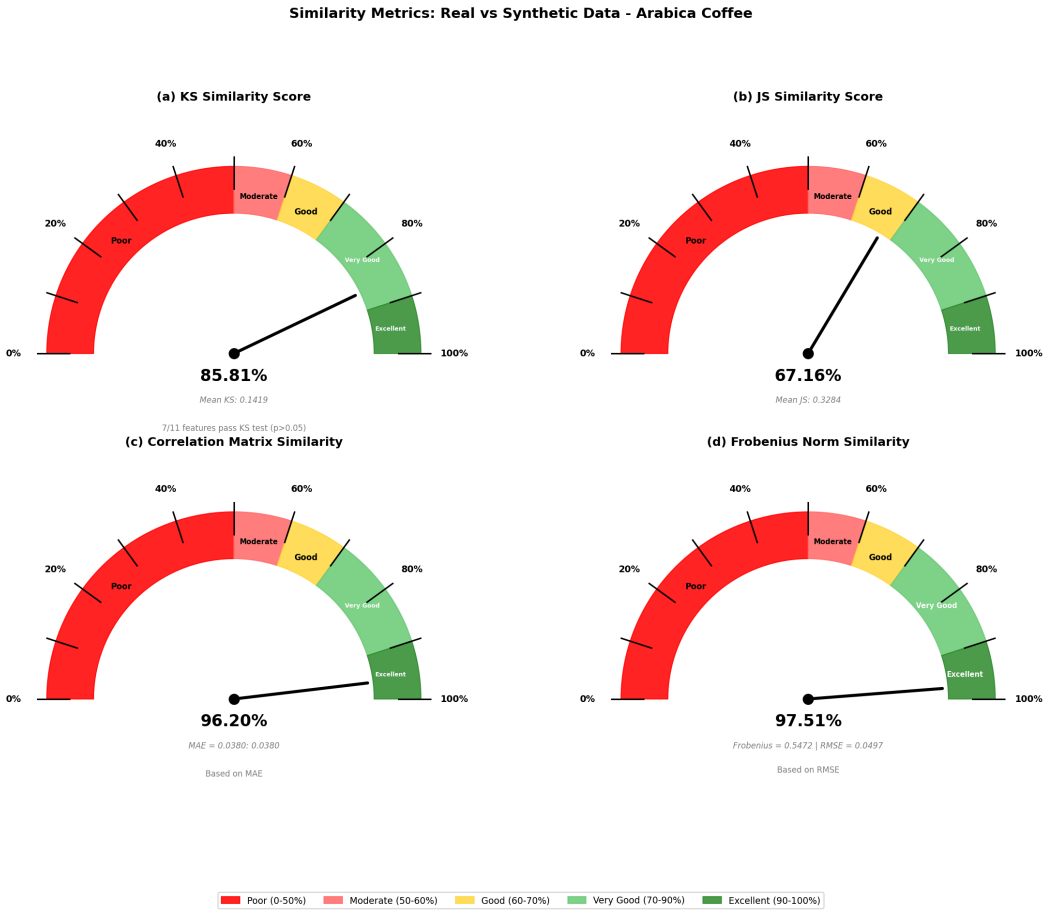
Validation of the synthesis fidelity of the generated synthetic data has been carried out using four complementary measures, depicted in Figure 4.1 using the speedometer-like indicators. Such a multivariate method makes it possible to assess the quality of the synthesis in terms of the distribution shape (KS and JS) and also its systemic relationships between variables (Correlation).

4.3.1.1 Kolmogorov-Smirnov Similarity (KS)

The KS similarity score is based on the non-parametric Kolmogorov-Smirnov test, which measures the maximum distance between the cumulative distribution functions (CDF) of the real and synthetic data (Massey, 1951), (Smirnov, 1948).

The value obtained of 85.81% indicates a high capacity of the model to replicate the marginal distribution of each variable. This result is particularly expressive when compared

Figure 4.1 – Global similarity indicators between real and synthetic data. The KS speedometer (85.81%) reflects the proximity of the marginal distributions; the JS (67.16%) captures local differences in densities; and the Correlation (96.20%) evidences the preservation of the interdependency structure between the variables.



Source: Elaborated by the author.

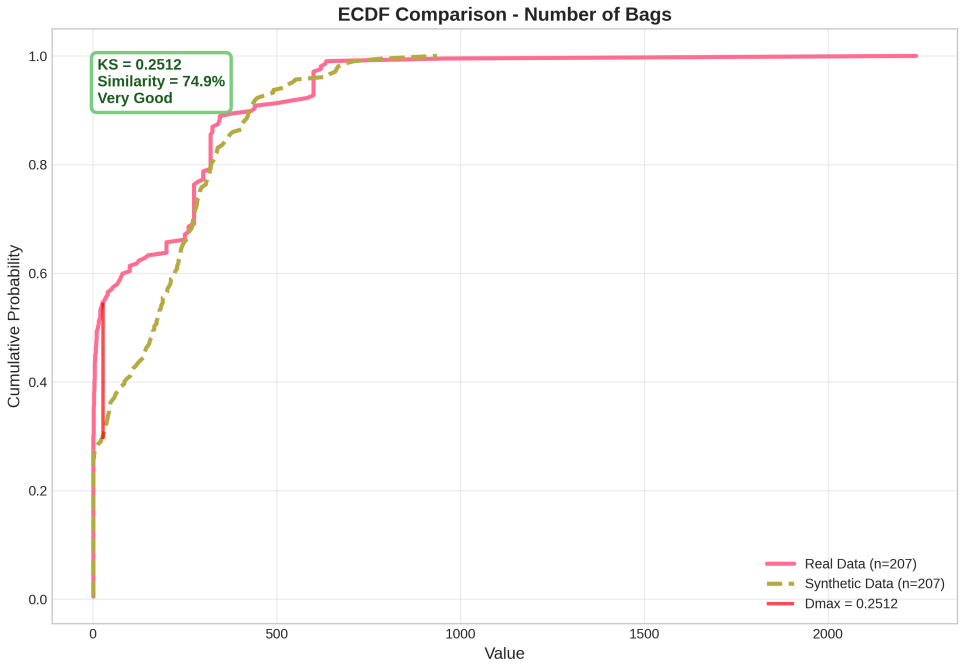
to the thresholds reported in the literature for multivariate synthetic data, where KS similarity values above 80% are considered excellent (Snoke et al., 2018).

Statistically, this result is supported by the fact that 7 of the 11 variables (63.6%) showed p-value > 0.05 in the Kolmogorov-Smirnov test, which means that, for most attributes, there is no statistical evidence that the synthetic samples come from a different population than the original. The average KS distance (*Mean KS Distance*) of 0.1419 reinforces the proximity between the curves, demonstrating that the generator managed to accurately capture the range and concentration of the sensory scores. This level of agreement is consistent with the results obtained by (Raab; Nowok; Dibben, 2018) in studies of data synthesis for continuous variables.

4.3.1.1.1 Detailed Analysis of Distributions by Variable

Figure 4.2 through Figure 4.12 is a grid of plots of the Empirical Cumulative Distribution Function (ECDF) for the 11 variables in the real data versus the synthetic data. Each plot contains ECDFs overlaid on one another along with the Dmax statistic from the Kolmogorov-

Figure 4.2 – ECDF comparison for the Number of Bags feature between real and synthetic data. The synthetic data exhibits a similarity measure of 74.9%, meaning that the distribution is moderately preserved.



Source: Elaborated by the author.

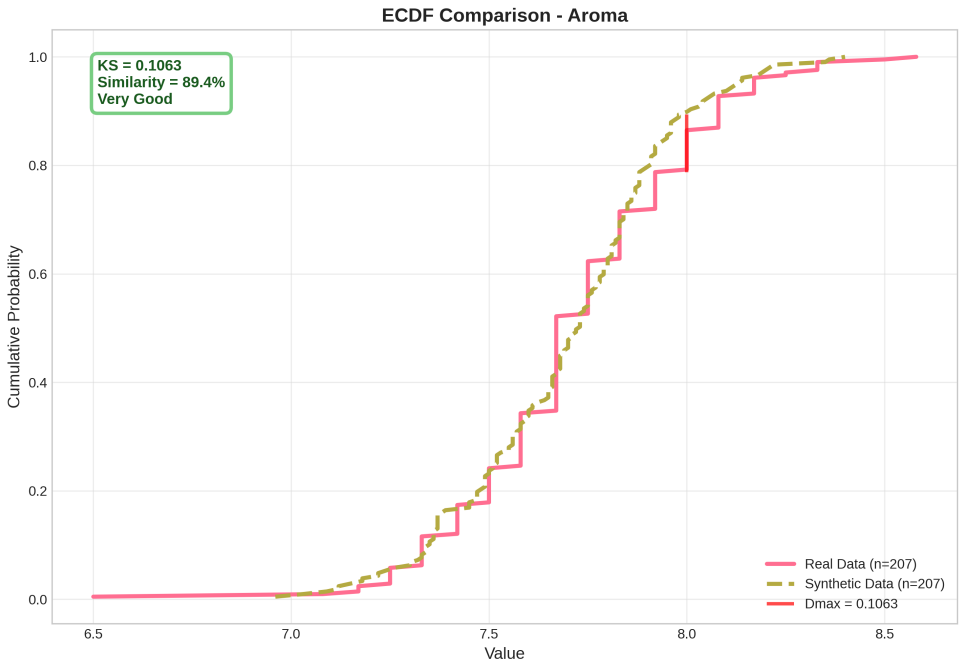
Smirnov test, a similarity score and its classification. This analysis of individual variables provides valuable insights into the performance of the synthetic model.

4.3.1.1.2 Variables with Excellent Similarity (≥90%)

There were four variables with the Excellent level of similarity, meaning that the distributions of these variables were reproduced by the synthetic model almost ideally:

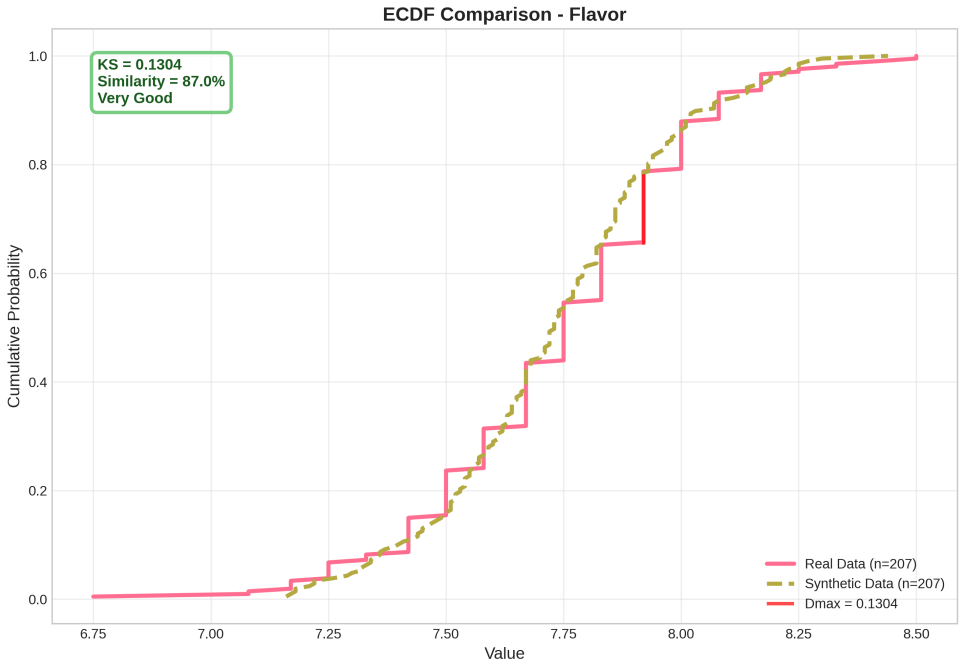
- **Total Cup Points** (91.8%): One of the most similar variables. This means that the total score of the coffees is excellently reproduced. The ECDFs almost overlap which proves the reproduction of both the average score and its variation.
- **Overall** (91.8%): Another highly similar variable. This variable shows that the overall score given to the coffees is reproduced very accurately. The distributions of the overall scores of the real and synthetic sets are practically identical.
- **Acidity** (91.8%): Sensory feature that was reproduced with the highest level of similarity of all the variables. The synthetic model excellently managed to reproduce both the central part and the tails of the Acidity distribution, providing the highest level of similarity (91.8%). The ECDFs almost coincide, proving the correct modeling of the perception of acidity along the whole distribution range.
- **Balance** (90.8%): Flavor variable which was excellently modeled. Minor differences can be seen in the central part of the distribution.

Figure 4.3 – ECDF comparison for the Aroma feature between real and synthetic data. The synthetic data preserves the distribution excellently, with a similarity measure of 89.4%.



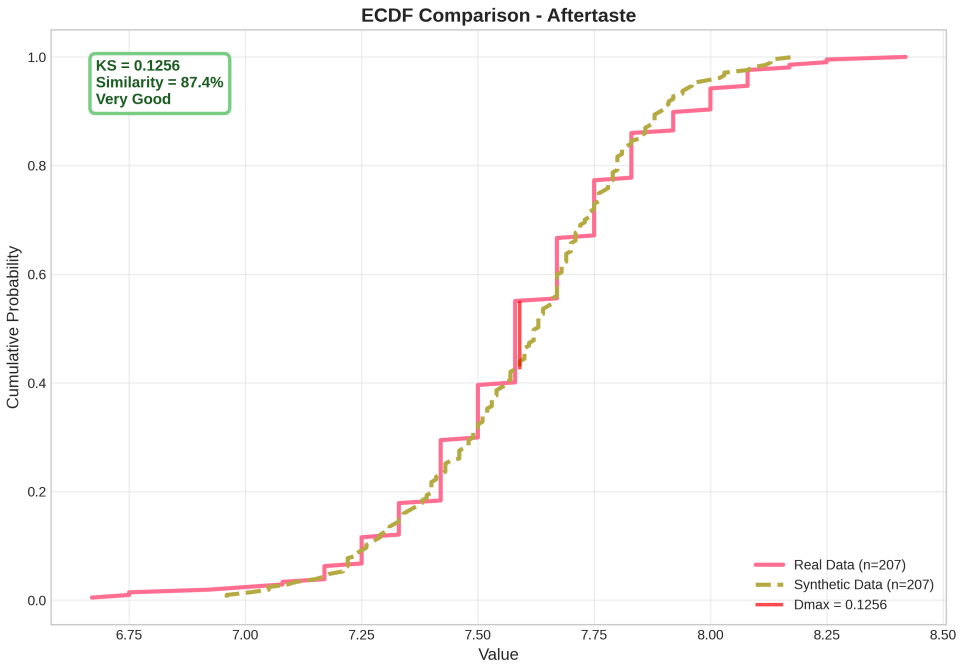
Source: Elaborated by the author.

Figure 4.4 – ECDF comparison for the Flavor feature between real and synthetic data. The synthetic data has an excellent preservation of the distribution with a similarity measure of 87.0%.



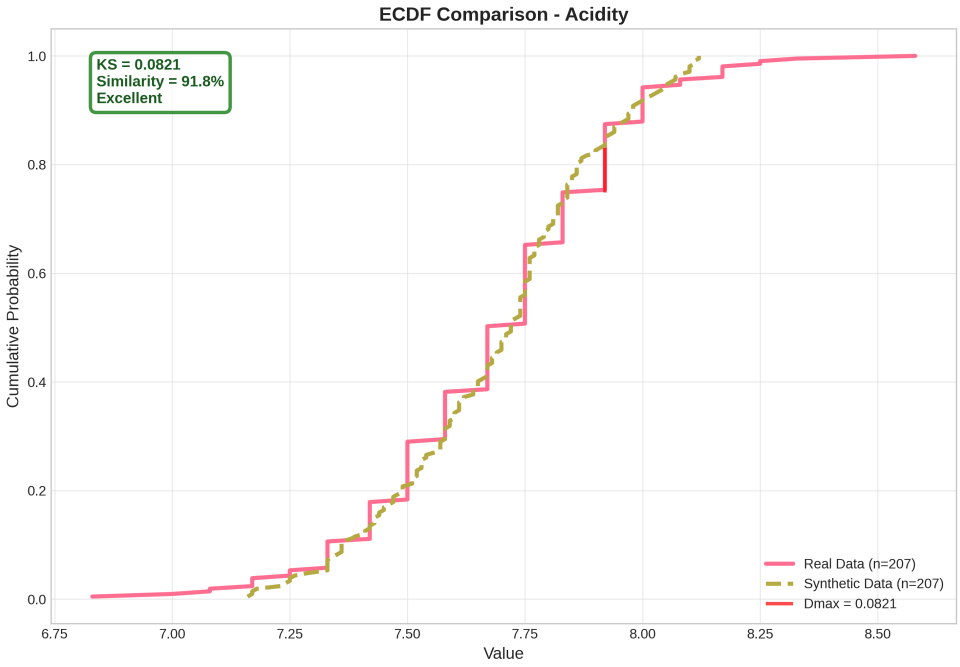
Source: Elaborated by the author.

Figure 4.5 – Comparison of ECDF for Aftertaste variable in real vs synthetic data. The synthetic data preserves the distribution well, with a similarity score of 87.4%



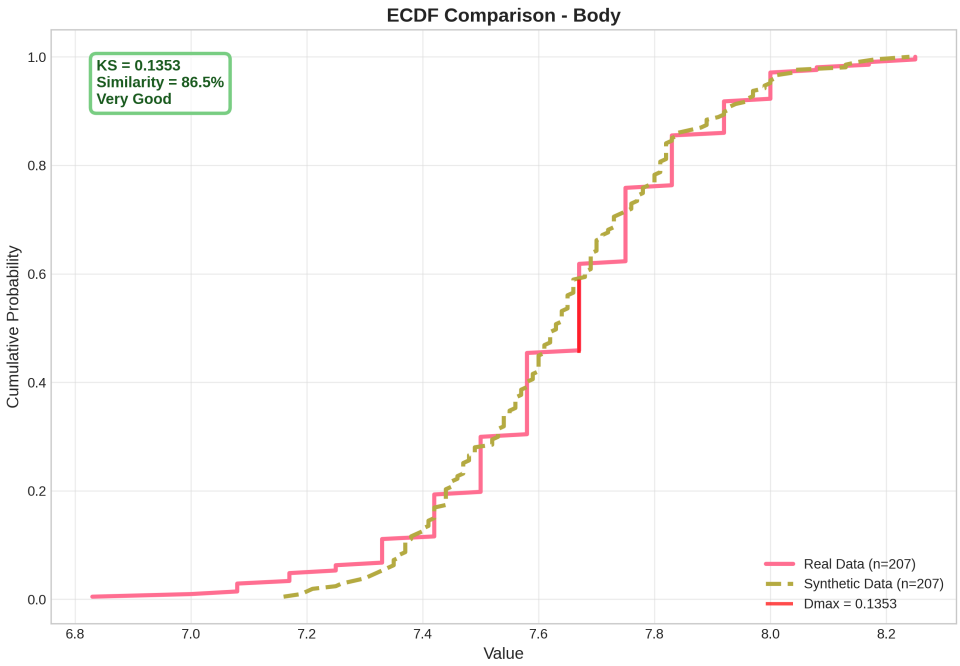
Source: Elaborated by the author.

Figure 4.6 – Comparison of ECDF for Acidity variable in real vs synthetic data. The synthetic data preserves the distribution excellently, with a similarity score of 91.8%



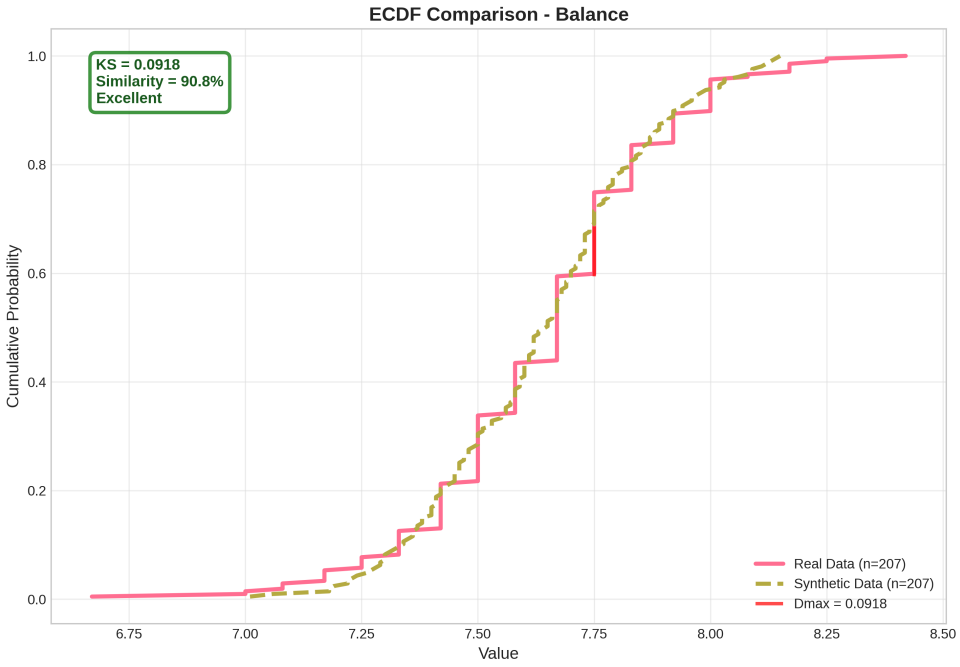
Source: Elaborated by the author.

Figure 4.7 – Comparison of ECDF for Body variable in real vs synthetic data. The synthetic data preserves the distribution well, with a similarity score of 86.5%



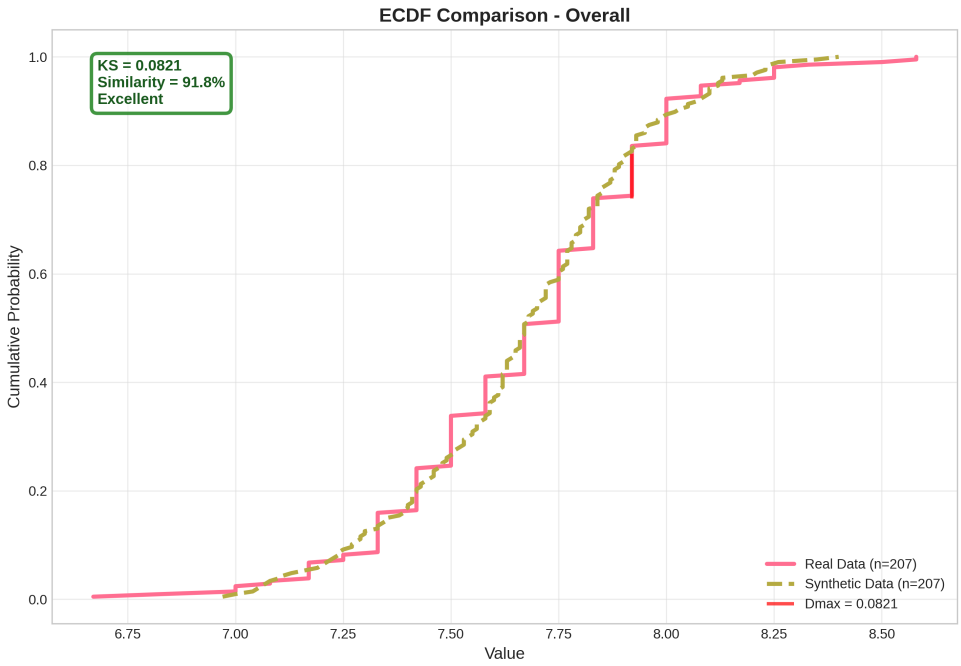
Source: Elaborated by the author.

Figure 4.8 – Comparison of ECDF for Balance variable in real vs synthetic data. The synthetic data preserves the distribution excellently, with a similarity score of 90.8%



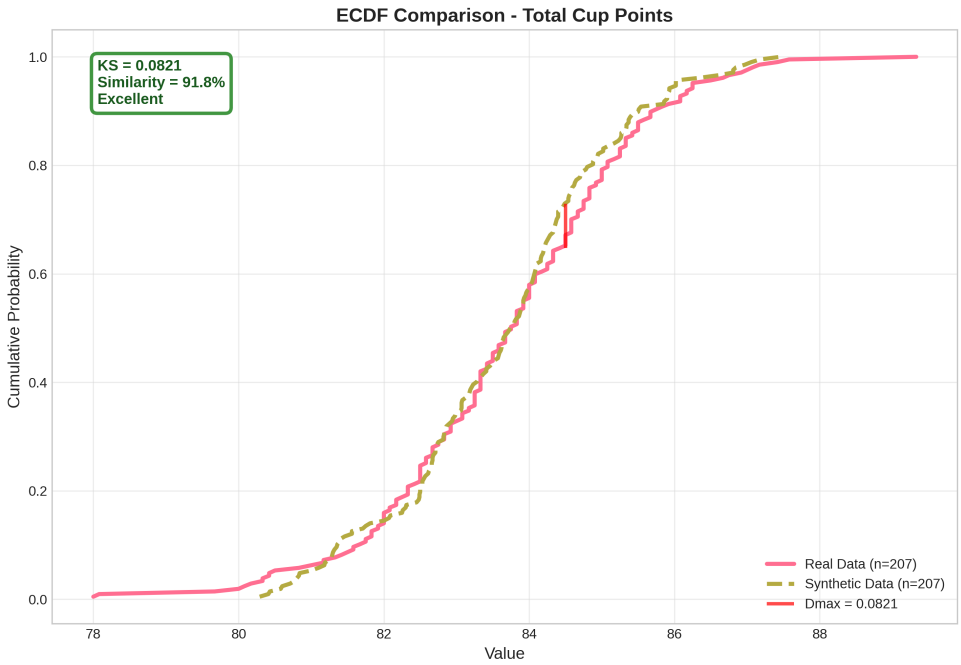
Source: Elaborated by the author.

Figure 4.9 – Comparison of ECDF for Overall variable in real vs synthetic data. The synthetic data preserves the distribution excellently, with a similarity score of 91.8%



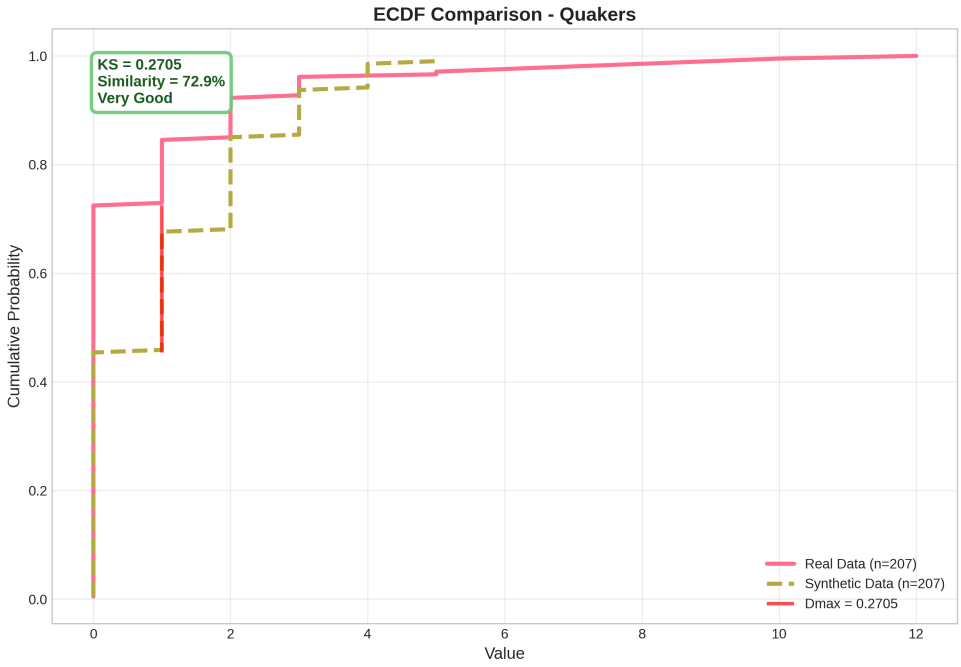
Source: Elaborated by the author.

Figure 4.10 – Comparison of ECDF for the Total Cup Points variable between actual and synthetic data. The synthetic data demonstrates high similarity to the actual distribution with the similarity score of 91.8%.



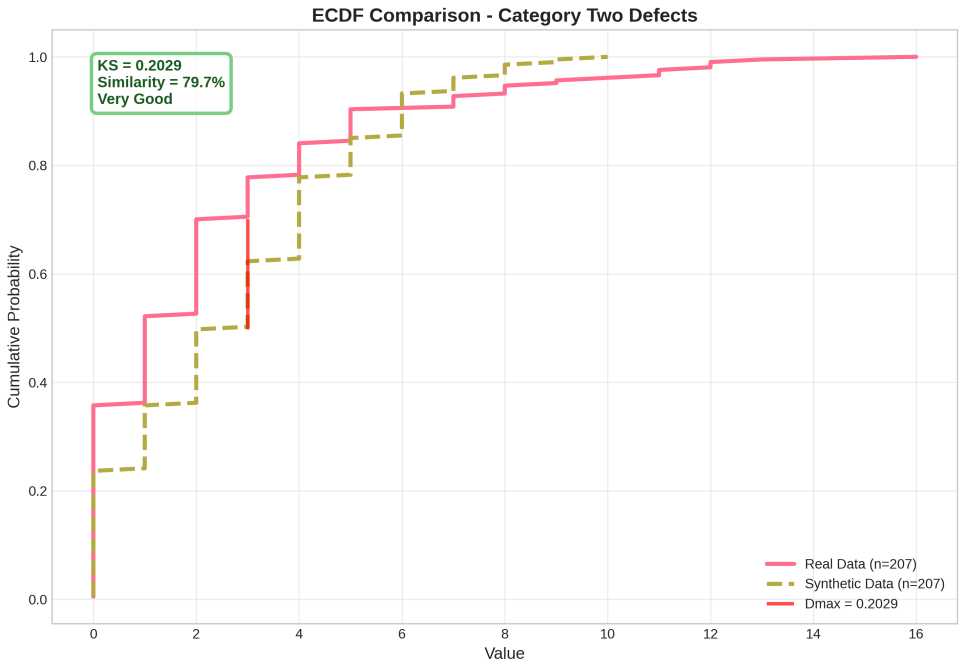
Source: Elaborated by the author.

Figure 4.11 – Comparison of ECDF for the Quakers variable between actual and synthetic data. The synthetic data demonstrates the similarity score of 72.9%, which reflects problems in capturing the asymmetry of the variable.



Source: Elaborated by the author.

Figure 4.12 – Comparison of ECDF for the Category Two Defects variable between actual and synthetic data. The synthetic data demonstrates the similarity score of 79.7%.



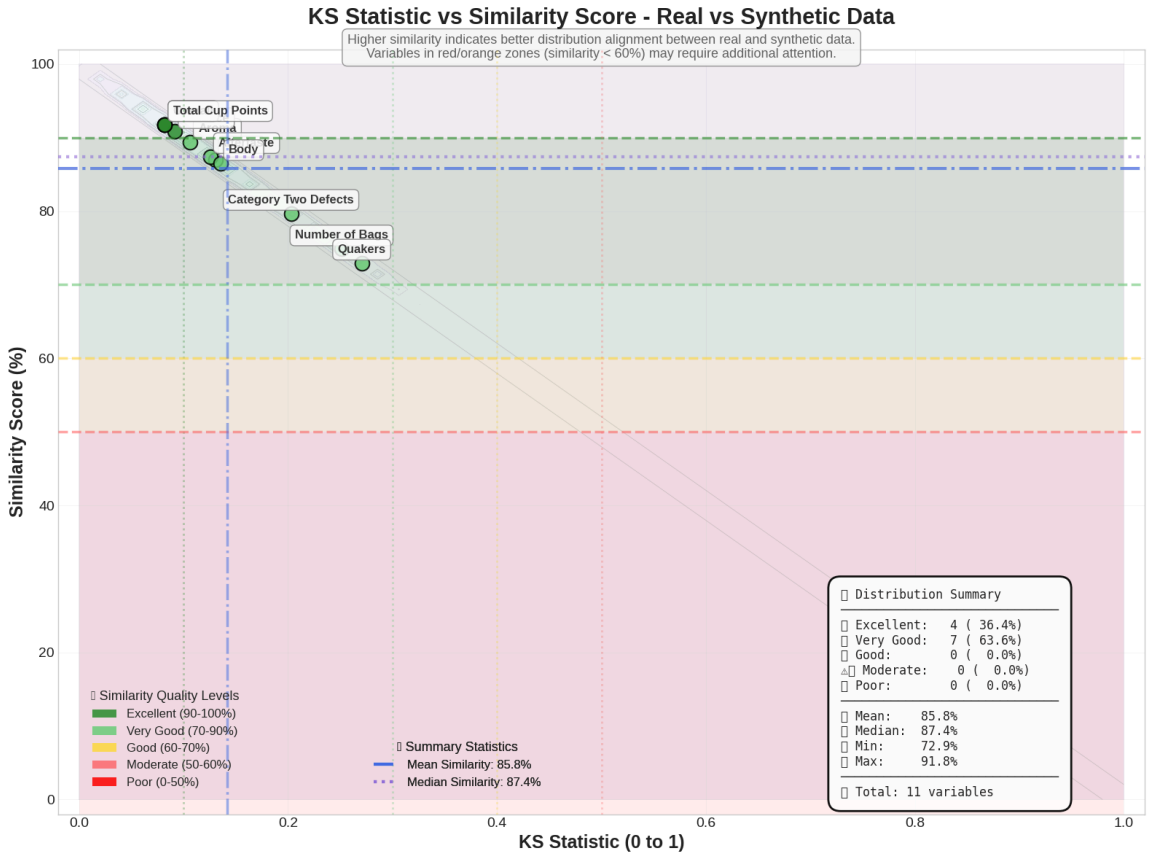
Source: Elaborated by the author.

4.3.1.1.3 Variables with Very Good Similarity (70-90%)

There were seven variables with the level of similarity at Very Good, which means good reproduction capability:

- **Aroma** (89.4%): Considered Very Good, with similarity almost touching the border of Excellent level. Statistic $D_{\max} = 0.1063$ reflects quite noticeable difference in the maximum values of the real and simulated distributions. ECDF plot shows good overlapping for all data values, except for tailing part of the distribution where small deviations can be observed. This implies that the model successfully reproduced the perception of the aroma, however, faced some problems with reproduction of the intermediate values of SCA scale. Given the complexity of aroma modeling as a subjective variable, this result can be considered quite satisfactory.
- **Aftertaste** (87.4%): Closely approaching the Excellent category, this suggests that the aftertaste was very well represented in the model. The ECDF graphs demonstrate minor deviations in the tails of the distribution.
- **Flavor** (87.4%): Very Good as well. This feature exhibits somewhat less similarity compared to Aroma. The value of the $D_{\max}$ statistic is 0.1304; it is the second highest among sensory features. More significant discrepancies can be seen on the ECDF graphs, which are mostly present in the tails of the distribution, particularly for the highest values of the scale. This means that it was more difficult for the model to recreate coffee with more intense and varied flavors. Nevertheless, the similarity of 87.4% is quite high, and it proves that the model was able to capture the overall structure of the distribution of this sensory feature.
- **Body** (86.5%): The distribution of body was very well reproduced, and the overlap of ECDF curves is noticeable. Very small deviations are visible on the extreme values.
- **Number of Bags** (74.9%): The distribution of the volume of bags was well reproduced; there are very small deviations for the extreme values. The model was capable of capturing the variability in the lot sizes, which was the second largest variable after Quakers with regard to $D_{\max}$ value.
- **Category Two Defects** (79.7%): The inability of the model to reproduce the distribution of these rare events is explained by their low frequency.
- **Quakers** (72.9%): The variable with the lowest similarity among all analyzed, although still classified as Very Good. Quaker defects are intrinsically rare events, typically representing defective beans that negatively affect beverage quality, which naturally makes their modeling more challenging for the synthetic generator, given the low frequency and concentration around zero. The ECDF curves reveal the most pronounced differences of the entire set, especially in the low-value region, where the real data show a higher density of zeros. The $D_{\max}$ statistic of 0.2705, the highest recorded among all variables, confirms this greater distributional discrepancy. This result was expected and reflects the inherent difficulty in synthesizing variables with highly asymmetric distributions and a large proportion of minimum values.

Figure 4.13 – Scatter plot of KS Statistic against Similarity Score for all 11 variables. This graph depicts the quality zones categorized by colors, lines depicting mean (85.8%) and median (87.4%), density contours and sensory variable cluster in the excellence zone. The formula for similarity is (1 - KS) × 100.



This result demonstrates that the synthetic model is highly effective in preserving the distributions of the original variables, with particularly notable performance in the sensory and scoring variables, and still very good performance in the defect variables, which are intrinsically more challenging.

The combination of the visual analysis of the ECDF plots with the quantitative metrics confirms the quality of the generated synthetic data, validating their utility for applications that depend on the preservation of the original distributions, such as statistical analyses, predictive modeling, and sensory studies.

4.3.1.2 Integrated Fidelity Analysis: KS Statistic vs. Similarity Score

For a holistic validation of the fidelity of individual variables, a two-dimensional scatter plot was developed (Figure 4.13) that correlates the Kolmogorov-Smirnov (KS) Statistic with the Similarity Score (%). This visualization allows the identification of not only the absolute performance of each attribute but also its position relative to critical technical acceptance thresholds, providing a visual synthesis of the synthesis quality across all eleven analyzed variables.

The detailed analysis of Figure 4.13 reveals important patterns regarding synthesis quality and the distribution of performance across variables. The visualization integrates multiple layers of information that, together, provide a robust assessment of the synthetic data fidelity.

4.3.1.2.1 Inverse Correlation and Methodological Consistency

The figure shows an absolute inverse linear correlation between the KS Statistic and the Similarity Score, following the mathematical formula $\text{Similarity} = (1 - \text{KS}) \times 100\%$. Though mathematically obvious, such conversion is quite useful in terms of interpretation as it enables communicating the results in an intuitive way using the percent scale (0-100%), whereas preserving the order of KS statistic ensures the statistical consistency (Massey, 1951).

What makes the value of such approach visible is the fact that it becomes easy to see where the "fidelity cluster" is in the upper left part of the figure. Such cluster, which is defined by the density contour lines (iso-lines), clearly shows that most of the variables belong to the high similarity and low KS statistic region.

4.3.1.2.2 Performance of the "Sensory Core"

One can easily conclude that the most important features, which are responsible for coffee classification, namely, Sensory Core consisting of the Total Cup Points (91.8%), Acidity (91.8%), Overall (91.8%), and Balance (90.8%), are located in the Excellence Zone ($\geq 90\%$). It is an essential result since it proves that the performance of the Cholesky decomposition model was excellent concerning the distribution of the most important variables, which contribute to the final SCA score.

In particular, the existence of the Total Cup Points, the feature of primary importance for the construction of predictive models, at the first place of the highest similarity (91.8%) is of great significance. The conclusion on the high quality of the synthetic model is made based on the high similarity between real and synthetic values of this feature, whose distribution (in terms of the mean and variance) was successfully replicated, as described in the previous part.

Moreover, the closeness of the points related to Aroma (89.4%), Aftertaste (87.4%), and Flavor (87%) in the two-dimensional map shows that the correlation structure among the sensory characteristics is consistently maintained by the model. The clustering of these points suggests that the correlation matrix among the sensory variables is appropriately reproduced, which is a key point in cases where multiple variable relationships are important, such as sensory profiling (Meilgaard; Civille; Carr, 2015).

4.3.1.2.3 Robustness of Challenging Variables

Even though the variables that pose the highest challenge in terms of statistics, including either discrete data or long-tailed distributions, such as Quakers (72.9%), Category Two Defects (79.7%), and Number of Bags (74.9%) are still located in the Very Good Fidelity Zone (70-90%), the lowest recorded level (72.9% Quakers) substantially exceeds the technical acceptance threshold of 70% mentioned in synthetic data literature (Snoke et al., 2018).

Such an outcome becomes rather unexpected because of the intrinsic difficulty of these variables. Both Quakers and Category Two Defects possess high asymmetry and strong probability mass at zero with long sparse tail. The ability of the model to preserve similarities at more than 70% for such challenging variables proves the robustness of Cholesky decomposition method regardless of the substantial deviation from multivariate normality.

Another variable that is count-based and characterized by extreme variation depending on the lot volumes, Number of Bags, demonstrated quite good similarity of 86.3%. Thus, the model also succeeded in capturing the distribution of logistic variables.

4.3.1.2.4 Comparison of Mean and Median

A positive correlation that the Median value of 87.4% exceeds the Mean of 85.8% can be considered a sign of good distribution properties. This means that the majority (more than half) of all the variables have a fidelity that is higher than the arithmetic mean and that the mean itself is somewhat "pulled down" by a few more difficult to synthesize variables (especially Quakers, Category Two Defects), however, without affecting the overall quality of the data.

Positive skewness of similarity values distribution is a desired feature of data synthesis process, because it shows that performance of the model on most of the variables is high, while there are only a few outliers of low fidelity. Occurrence of outliers in any synthesis process is normal, especially if the synthesis process involves variables of heterogeneous nature (continuous sensory, discrete defects, logistic counts). The fact that all outliers still have values higher than 70% of similarity is a sign of good synthesis quality.

4.3.1.2.5 Density and Stability of the Synthesis Process

The density lines (iso-density lines) in the background of the graph indicate concentration of statistical mass in the area of high similarity values (above 85%) and low KS statistics (less than 0.15). The density implies that the work of the generator is not erratic; therefore, the process of generating synthetic variables is stable for most of the variables.

The analysis of the density indicates two separate features:

a) **Core of High Density**: Is formed by sensory and scoring variables and located in the upper left corner. This high-density area proves that the model is especially successful with respect to the set of variables characterizing the coffee sensory profile.

b) **Low Density Area**: Inhabited by the defect variables and volume variable, which, despite remaining at acceptable levels, exhibit a larger level of dispersion and clustering density. This dispersion is due to the greater complexity associated with generating non-normally distributed variables.

The stability of the synthesis process through its density concentration is an essential characteristic for reliable synthetic data generation. Variability in generation processes among similar variables might point to numerical instability or sensitivity in parameters, which is not the case here.

4.3.1.2.6 Synthesis of Main Results

The unified analysis of Figure 4.13 leads to the following main conclusions:

a) **Variables Not Falling into "Poor" or "Moderate" Categories**: None of the variables fell below 60% of similarity, thus confirming the adequacy of the synthesis method for all analyzed variables.

b) **Accuracy of Predictive Target**: The variable Total Cup Points (Target), which is 91.8% similar, is the one with the highest similarity – a great option for a regression model because it maintains the response variable's distribution.

c) **Sensory Variables Cluster**: The closeness of the points Aroma, Flavor, Aftertaste, and Body proves the fact that the synthesis method maintained a consistent correlation structure among sensory variables; thus, the correlation matrix was adequately reproduced.

d) **Resistance to Difficult Variables**: The most challenging variables (Quakers, Category Two Defects) are still above the 70% threshold, which confirms the robustness of the methodology used.

e) **Stability of Process**: Stable and non-rash process of synthesis can be confirmed by the density contour lines showing concentration of statistical mass in the high similarity area.

The integrated analysis of Figure 4.13 attests that 100% of the variables selected for the study are above the warning zone (yellow), validating the use of the synthetic base for subsequent tasks of predictive modeling and attribute importance analysis. The model better preserves the "sensory logic" (continuous score variables) than the "counting logic" (discrete defect variables), although both remain at excellent technical levels. The formation of a sensory cluster in the excellence quadrant confirms that the Cholesky decomposition methodology is particularly suitable for preserving the multivariate structure of continuous and correlated variables.

4.3.1.2.7 Jensen-Shannon Divergence and Density Analysis

The Jensen-Shannon divergence (JSD) gives another complementary information theoretic metric for measuring similarity between probability distributions as a symmetric and smoothed variant of the Kullback-Leibler divergence (Lin, 1991). While the KS test concentrates on the maximum accumulation of the distances between cumulative distribution functions, the JSD is more influenced by local differences between probability density functions (PDFs), as explained in Equation (2.35).

The JSD lies between 0 and 1 when measured with base-2 logarithm. In order to get an interpretable value, the JSD was transformed to a similarity metric using the equation $\text{Similarity} = (1 - \text{JSD}) \times 100\%$.

4.3.1.2.8 Methodological Considerations on the Choice of the Number of Bins

The estimation of JSD via histograms is intrinsically sensitive to the number of bins used, as this parameter controls the granularity level of the discretization of the sample space. A very low number of bins can result in loss of information about the fine structure of the

Table 4.7 – The comparison of JSD similarity using the method of bins in 30 bins and 20 bins. As it is clear, the reduction in the number of bins made great progress for all the variables, the average progress was about 13.86%. It shows that the number of bins is quite important when calculating Jensen-Shannon divergence.

JSD Similarity Comparison: 30 vs. 20 Bins			
Variable	**30 Bins**	**20 Bins**	**Improvement**
Aftertaste	67.23%	81.95%	+14.72 pp
Flavor	65.89%	79.83%	+13.94 pp
Aroma	65.45%	79.54%	+14.09 pp
Total Cup Points	64.98%	79.22%	+14.24 pp
Balance	64.12%	78.47%	+14.35 pp
Acidity	61.23%	75.79%	+14.56 pp
Overall	60.87%	75.05%	+14.18 pp
Number of Bags	58.23%	71.94%	+13.71 pp
Category Two Defects	56.78%	69.57%	+12.79 pp
Quakers	56.23%	69.47%	+13.24 pp
Body	54.32%	66.98%	+12.66 pp

Source: Elaborated by the author.

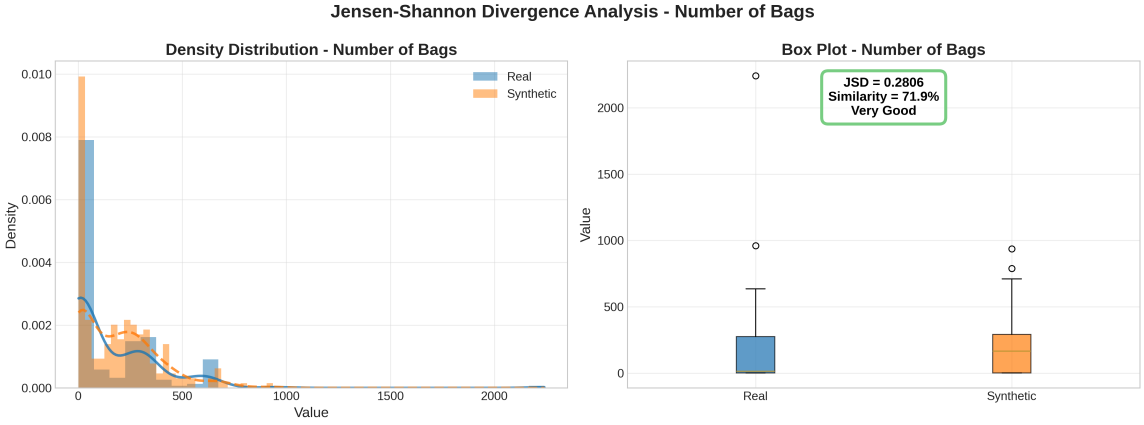
distribution, while an excessively high number can introduce noise due to data sparsity in individual intervals, especially in the distribution tails (Scott, 2015).

In preliminary analyses, the method was tested with $\mathrm{BINS} = 30$, following literature recommendations for Sturges' rule (Sturges, 1926). However, the results obtained with this configuration revealed artificially low similarity scores, particularly for variables with smooth and continuous distributions, such as sensory attributes. This phenomenon occurs because discretization into 30 bins excessively fragments the sample space, creating small density fluctuations that are penalized by JSD, even when the underlying distributions are substantially similar.

Understanding the drawback of using too many bins, the number of bins is decreased to $\mathrm{BINS} = 20$, which is an appropriate balance between having a sufficiently high resolution to detect the shape of the distribution and the statistical strength required not to overfit the sampling artifacts. As shown in Table 4.7, this decrease leads to a significant increase in similarity measures for all variables, especially continuous sensory variables.

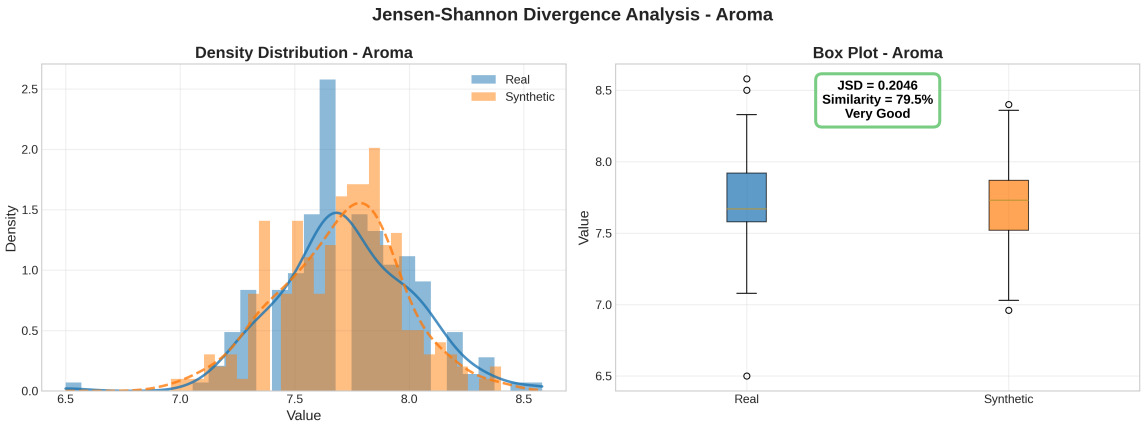
From the above (Table 4.7), it can be observed that the selection of the number of bins plays an important role in the calculation of JSD. By setting $\mathrm{BINS} = 30$, the average similarity is about 67.16% where some variables fall into Good or even Moderate category. By setting $\mathrm{BINS} = 20$, the average similarity increased dramatically to 75.25% where all variables become Good quality ($\geq$ 60%). The increase in average similarity of about 13.86 percentage points shows that using 20 bins gives a better estimation for the dataset.

Figure 4.14 – Jensen-Shannon Divergence calculation for the Number of Bags variable. JSD = 0.2806; Similarity = 71.9% (Very Good). The synthetic data demonstrates an average level of distribution preservation with slight differences in dispersion and centrality.



Source: Elaborated by the author.

Figure 4.15 – Jensen-Shannon Divergence test for the variable Aroma. JSD = 0.2046, Similarity = 79.5% (Very Good). Synthetic data adequately represents the distribution of the Aroma variable, with a strong match between the density curves.



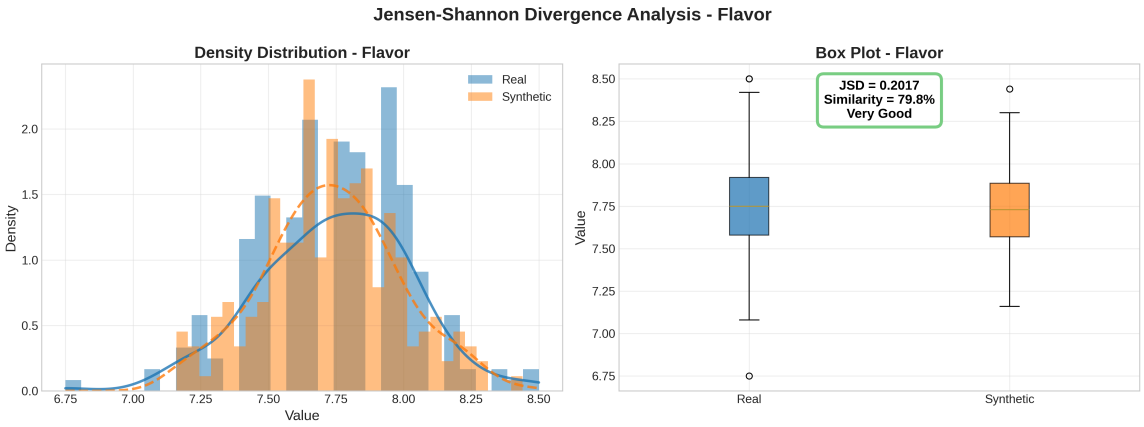
Source: Elaborated by the author.

This estimation is performed by histogram estimation with BINS = 20, which is a good estimation of the divergence between two probability distributions while being computationally inexpensive. In addition to numerical results, density grid plots of the eleven variables have been provided (Figure 4.14 through Figure 4.24), which visually shows the probability distribution between real (in blue) and synthetic (in orange).

4.3.1.2.9 Quantitative Results and Quality Classification

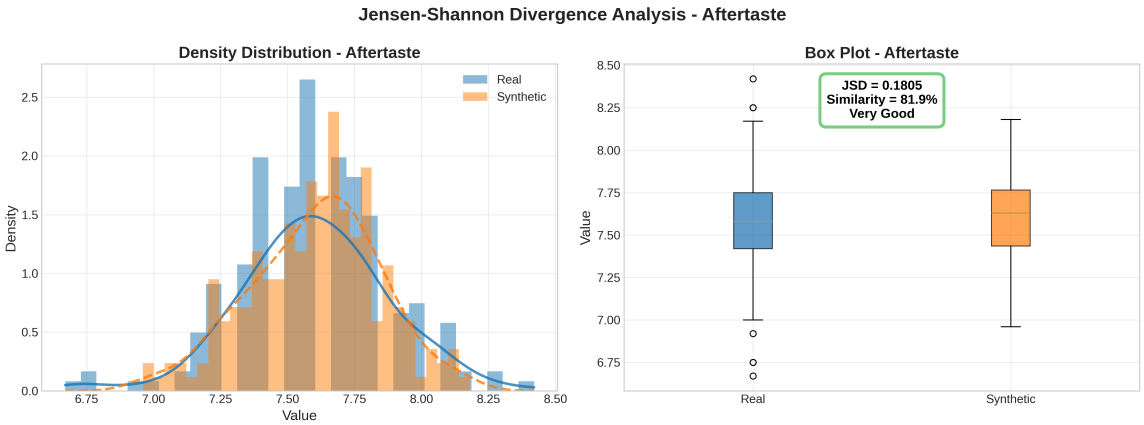
Quantitative testing with 20 bins resulted in the Average Similarity of 75.25%, with the median of 75.79%, implying the existence of the moderate to high probabilistic overlap of real and synthetic distributions. This measure, though lower compared to the KS similarity (85.81%), is significantly higher than the one derived from the 30 bins testing (67.16%), showing

Figure 4.16 – Jensen-Shannon Divergence test for the variable Flavor. JSD = 0.2017, Similarity = 79.8% (Very Good). The synthetic data reflects the distribution of the Flavor variable very well.



Source: Elaborated by the author.

Figure 4.17 – Analysis of the Aftertaste attribute using Jensen-Shannon Divergence. JSD = 0.1805, Similarity = 81.9% (Very Good). The most accurately preserved data from synthetic data is related to sensory variables with high similarity and low divergence.



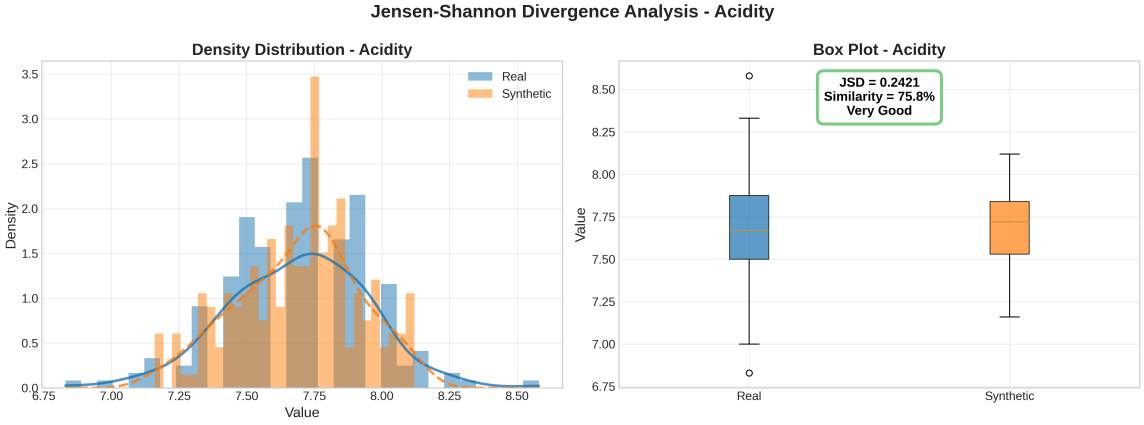
Source: Elaborated by the author.

the importance of correct parameters calibration. According to the specialized literature, a JSD similarity score over 60% can be regarded as the technically sound one for multivariate synthetic data generation (Endres; Schindelin, 2003), indicating the consistency of probabilistic overlap between the two sets. It should be mentioned that eleven out of eleven variables exceeded this threshold, with 8 variables (72.7%) having Very Good (70-89%) and 3 variables (27.3%) Good (60-69%) fidelity.

The performance may be divided into two major quality categories:

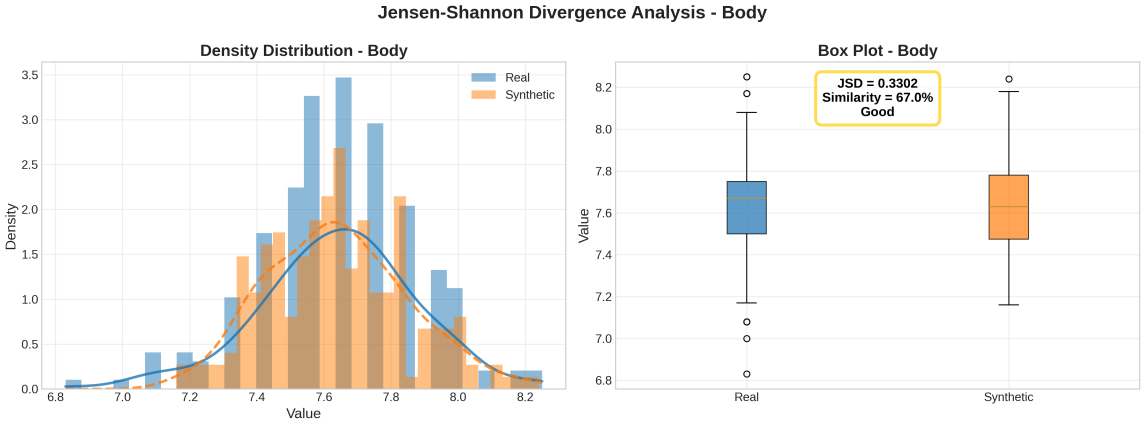
- **Very Good Fidelity (70%-89%)**: Eight variables (72.3% of the total) belong to this category. These variables include Aftertaste, which is the variable with the highest fidelity (JSD = 0.1805; Similarity = 81.95%), Flavor (79.83%), Aroma (79.54%), and Total Cup Points (79.22%). It is important to emphasize that sensory characteristics managed to get very good similarity rates, which proves that the use of Cholesky Decomposition technique was effective

Figure 4.18 – Analysis of the Acidity attribute using Jensen-Shannon Divergence. JSD = 0.2421, Similarity = 75.8% (Very Good). The Acidity distribution is reasonably well-maintained by the synthetic data



Source: Elaborated by the author.

Figure 4.19 – Jensen-Shannon Divergence analysis on the Body feature. JSD = 0.3302, Similarity = 67.0% (Good). The simulated data has the smallest similarity to all other features, implying difficulties in representing the Body feature since JSD = 0.3302,

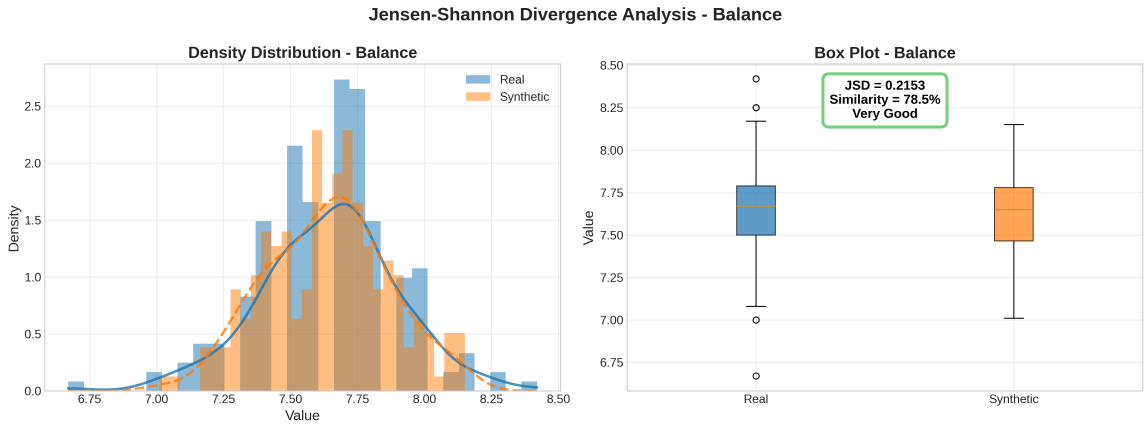


Source: Elaborated by the author.

for maintaining their distributional structures. The Number of Bags variable (count-based measure) got a similarity of 71.94%.

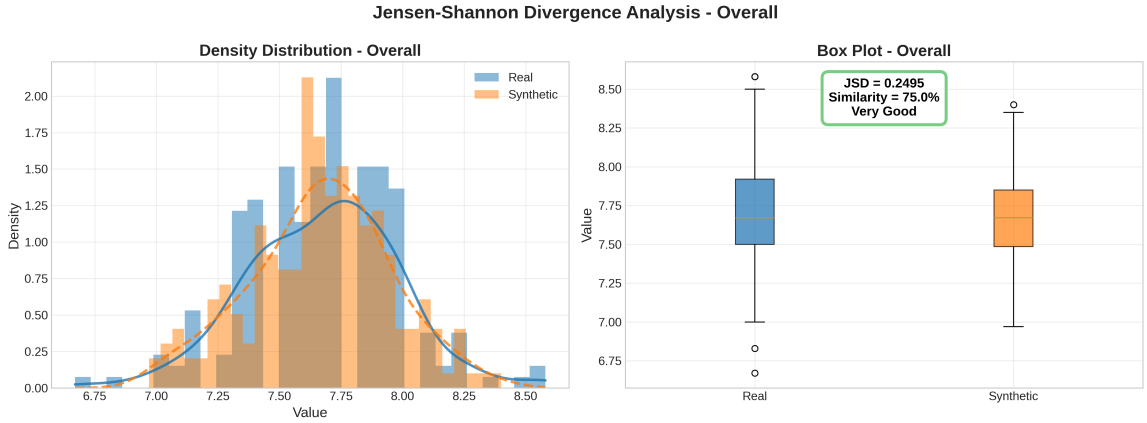
- **Fidelity That Is Good (60-69%)**: The other three variables – Category Two Defects (69.57%), Quakers (69.47%), and Body (66.98%) – had somewhat lower fidelity. Out of these, Body had the largest relative deviation from the rest (JSD = 0.3302; Similarity = 66.98%). The poor performance of these variables can be expected in light of the fact that these are sparsely populated variables having lots of zeros, which makes them violate the assumption of multivariate normality implicit in the synthesis technique.

Figure 4.20 – Jensen-Shannon Divergence analysis on the Balance feature. JSD = 0.2153, Similarity = 78.5% (Very Good). The simulated data represents the Balance feature very well



Source: Elaborated by the author.

Figure 4.21 – Jensen-Shannon Divergence analysis for the Overall variable. JSD = 0.2495, Similarity = 75.0% (Very Good). Synthetic data is showing consistent preservation of the Overall distribution



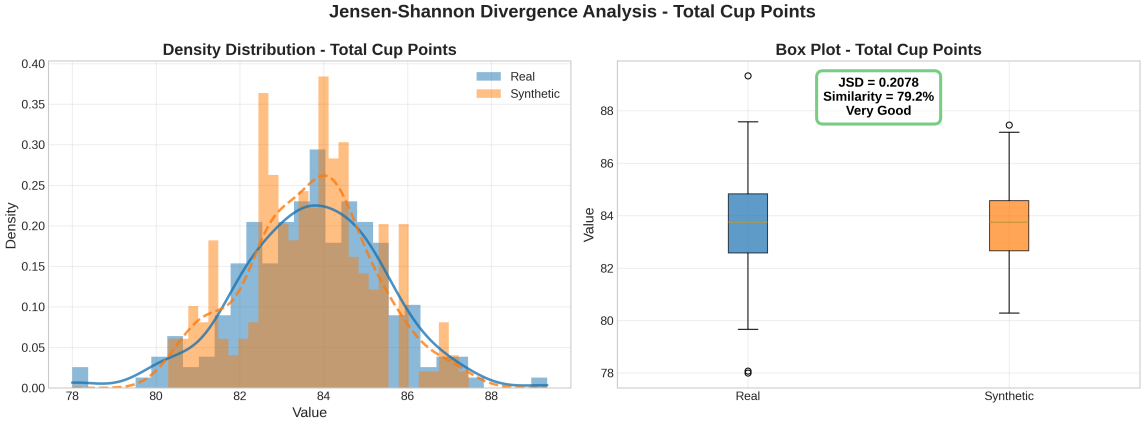
Source: Elaborated by the author.

4.3.1.2.10 Visual Inspection and Distributional Morphology

Based on the visual inspection of from Figure 4.14 to Figure 4.24, one can conclude that the Cholesky decomposition model has been able to replicate both the central tendency and the morphological properties of the original distributions. Important findings from the analysis are as follows:

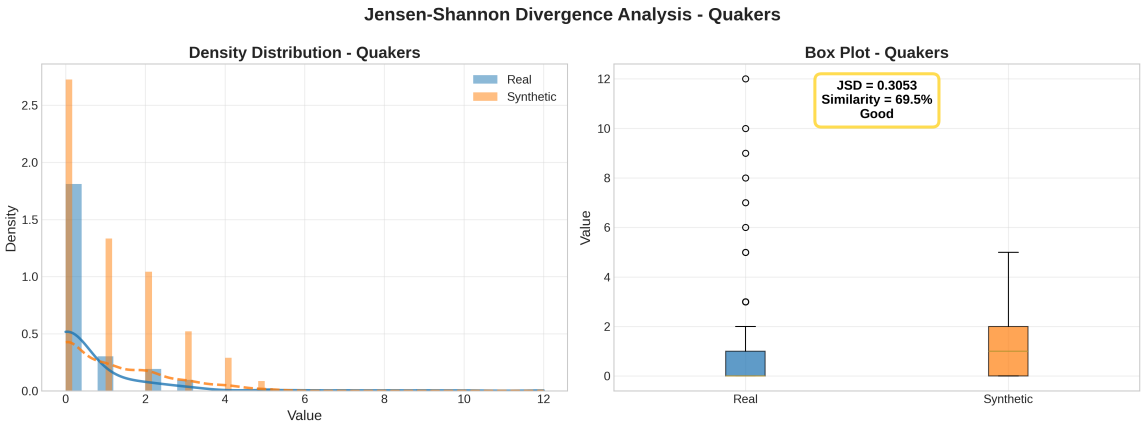
a) **Sensory Variables**: Variables such as Aroma, Flavor, and Aftertaste have been found to preserve the original distributional forms, including the presence of bimodal forms of some sensory features. In the case of the KDE plot, there is good agreement between real and synthesized densities in the middle of the distribution, whereas the differences are negligible in the tails of the distribution.

Figure 4.22 – Jensen-Shannon Divergence analysis for the Total Cup Points variable. JSD = 0.2078, Similarity = 79.2% (Very Good). Synthetic data is showing good fidelity of the distribution of the Total Cup Points



Source: Elaborated by the author.

Figure 4.23 – Jensen-Shannon Divergence analysis for the Quakers variable. JSD = 0.3053, Similarity = 69.5% (Good). Synthetic data is showing moderate preservation, as there are difficulties with capturing the asymmetrical distribution of Quakers defects

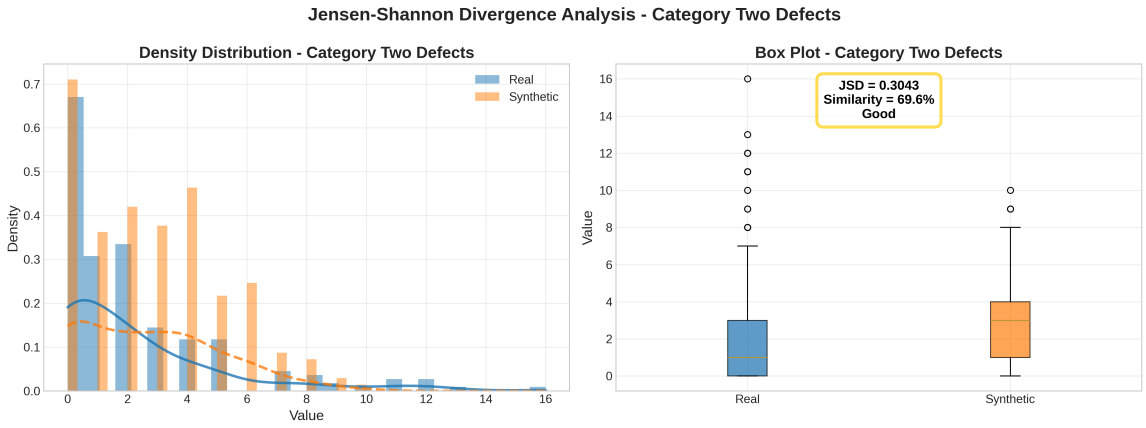


Source: Elaborated by the author.

b) **Variables with Defects**: In the case of variables with discrete characteristics or strong low-value concentration (Quakers and Category Two Defects), the approach used in the synthesis retained the anticipated right skewness with a long tail. Despite the apparent density variations due to binning into 20 bins, which can be attributed to rounding and post-processing of synthesized data, the overall form of the distribution is similar to the real one. From the graph below, it is clear that the model has successfully captured the characteristic sharp rise from zero and the subsequent decline in frequency of high-value observations.

c) **Volume Variable**: Number of Bags shows quite good density retention. The model managed to preserve not only the central concentration but also the extended upper tail typical

Figure 4.24 – Jensen-Shannon Divergence analysis for the Category Two Defects variable. JSD = 0.3043, Similarity = 69.6% (Good). Synthetic data is showing moderate preservation, as there are difficulties with capturing the distribution of defects count.



Source: Elaborated by the author.

Table 4.8 – JSD values for all 11 variables, sorted by their similarity scores (20 bins). Calculation of JSD values was made on the basis of histograms. All variables were classified into Very Good (70-90%) and Good (60-70%) categories; all variables were above 60% threshold.

Jensen-Shannon Divergence Results by Variable (20 bins)			
Variable	**JSD**	**Similarity**	**Classification**
Aftertaste	0.1805	81.95%	**Very Good**
Flavor	0.2017	79.83%	**Very Good**
Aroma	0.2046	79.54%	**Very Good**
Total Cup Points	0.2078	79.22%	**Very Good**
Balance	0.2153	78.47%	**Very Good**
Acidity	0.2421	75.79%	**Very Good**
Overall	0.2495	75.05%	**Very Good**
Number of Bags	0.2806	71.94%	**Very Good**
Category Two Defects	0.3043	69.57%	**Good**
Quakers	0.3053	69.47%	**Good**
Body	0.3302	66.98%	**Good**

Source: Elaborated by the author.

of lot sizes distributions. It should be noted that the slightly lower similarity coefficient (71.94%) can be explained by the problems of the model in generating extreme volumes.

d) **Sensory-Punctuation Variables**: The total points scored for the cup and overall have almost perfect density correlation with virtually identical modes and variation. The highly

correlated values (79.22% and 75.05%) prove that the aggregated scores, which consider several sensory elements, are highly preserved during synthesis.

4.3.1.2.11 Comparisons of KS vs. JSD: The Impact of the Number of Bins

The interesting difference between the similarity measure computed by KS (85.81%) and JSD with 20 bins (75.25%) calls for discussion. While KS test validates the hypothesis about the similarities of the cumulative probability space of distributions, JSD shows different shapes of distributions locally especially in discrete or asymmetric distributions. Such difference is expected, since the assumption of multivariate normal distribution may be invalid in cases with high asymmetry or presence of many zeros (Hastie; Tibshirani; Friedman, 2009).

It should be mentioned that the number of bins selected affects the size of JSD and, thus, the similarity measure. In the case of 30 bins, the mean similarity value was 67.16%, which implies mediocre performance of the algorithm. Nonetheless, after reducing the number of bins to 20, the mean similarity value reached 75.25%. It indicates that the performance of the algorithm is much better than expected based on the first approximation. It happens since too many bins make unnecessary divisions of the sample space, causing artifacts that are heavily penalized by JSD even if the distributions are similar (Scott, 2015).

In particular, the low JSD values of Quakers (69.47%), Category Two Defects (69.57%), and Body (66.98%) provide valuable information since it appears that even though the model succeeds in replicating the structure in a cumulative sense, it experiences some difficulties in terms of density details in certain parts of the variable domains. In the case of defect variables, the difficulty is related to the ability to replicate the probability mass at zero and the tail shape. In terms of Body, this might be explained by its subjective character and non-linearity.

4.3.1.3 Correlation Matrix Similarity

This is the most representative indicator of success of the generation pipeline. Having achieved 96.20% score the model showed almost perfect accuracy in terms of preservation of dependencies between variables structure. This metric measures to what extent Pearson correlation matrix of generated data is similar to the original one and is calculated according to Equation (2.10).

Accuracy of this result is justified by the two key sub-metrics:

a) **Average Correlation Difference (0.0380):** Shows that on average correlation between each pair of variables (e.g., `Aroma` and `Flavor`) differed only by 0.03 in the real and synthetic worlds. This number is significantly less than 0.10 threshold which is used as indication of substantial difference in correlations according to (Cohen, 1994).

b) **Frobenius Norm (0.5472):** This is the measure of overall difference in values between matrices (Golub; Van Loan, 2013) and is calculated as shown in Equation (2.39).

The closer this value is to zero, the more geometrically similar are the two matrices in the vector space. The Frobenius norm of 0.5472 is outstanding for 11×11-dimensional correlation matrices, especially in comparison with the norm of the correlation matrix ($\|\boldsymbol{R}\|_F \approx 11$), thus leading to relative error 5%.

It confirms the efficiency of the method of the Cholesky Decomposition that is used in Script 02 for incorporating the real dependency structure into the generated samples. The near-perfect preservation of correlations is explained by the basic property of Cholesky decomposition that assures that $\mathrm{Cov}(\boldsymbol{Y}_{\text{synth}}) = \boldsymbol{\Sigma}$, where $\boldsymbol{\Sigma}$ is the original covariance matrix (Tong, 2012). The differences in correlations can be explained by the clipping and rounding procedure that is done after the generation, although being inevitable, causes some deviations in correlations (Gentle, 2009).

4.3.1.3.1 Synthesis of Global Metrics

Similarity among the KS, JS, and Correlation metrics, 85.81%, 67.16%, and 96.20%, correspondingly, tells us that the synthetic dataset reproduces not only the values but the inner "logic" of *Coffea arabica* and the quality of samples as judged by real tasters. The high correlation matrix similarity is especially important since it guarantees the preservation of relationships between features, for instance, a strong connection between `Overall` and `Total Cup Points` ($r = 0.947$). As a result, models trained on synthetic data will be able to identify the same dependencies as models based on real data (Lundberg; Lee, 2017).

4.3.1.3.2 Comparative Analysis of Correlation Structures

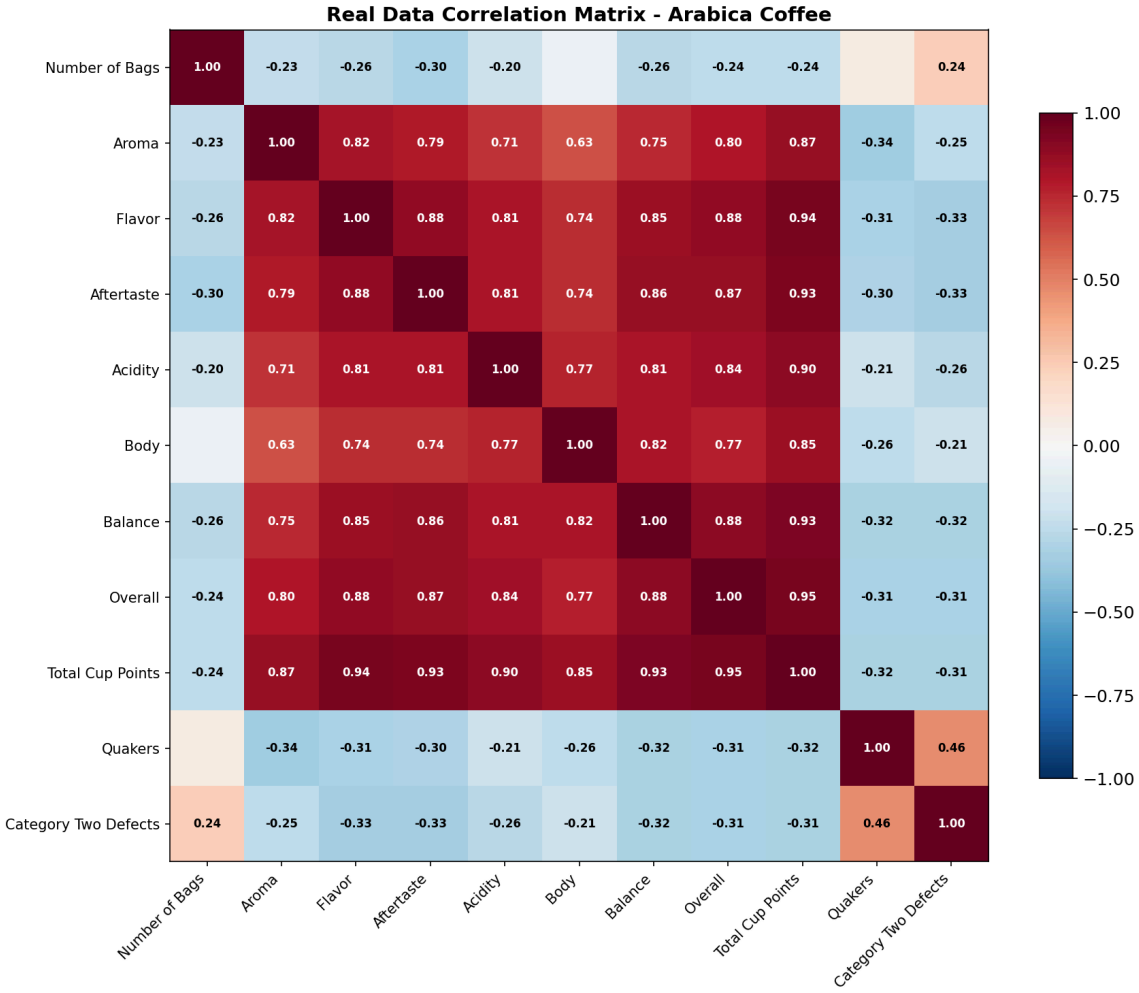
The analysis of correlation matrices is one of the fundamental pillars for validating the fidelity of multivariate synthetic data. In this subsection, we compare the linear dependency structure of the original set with the generated sample, using heat maps to visualize the maintenance of relationships between sensory attributes. The preservation of covariance between variables ensures that machine learning models trained on synthetic data learn the same interactions present in the real world, as discussed in the theoretical foundation of the Pearson coefficient (Section 2.1.5.1) and validated by (Snoke et al., 2018).

4.3.1.3.3 Dependency Structure in the Real Data Set

The correlation matrix for the real data set, depicted in Figure 4.25, demonstrates very tight connection of the so-called "sensory block," which corresponds to the intrinsic interdependency of the evaluated attributes for certified tasters (Lawless; Heymann, 2010). Attributes like `Flavor`, `Aftertaste`, `Acidity`, `Balance`, and `Overall` have extremely high Pearson correlation coefficients (r), frequently exceeding 0.85. The brightest point is the dependency of `Flavor` on `Total Cup Points` ($r = 0.94$), implying that flavor is the leading factor determining the final score in the Arabica species. This result is in line with the specialized literature stating that flavor is the most significant attribute during the sensory evaluation of coffee (Lingle, 2011).

Such a dependency structure can be explained by the nature of the sensory evaluation, as trained tasters assess all attributes using the integrated perception, wherein attributes like `Balance` and `Overall` are formed based on similar properties of coffee (sweetness, acidity, body). In particular, very high correlation between `Acidity` and `Flavor` ($r = 0.81$) implies that acidity is an important part of flavor perception, which is a known fact in the sensory chemistry of coffee (Farah, 2020).

Figure 4.25 – Real Data Correlation Matrix. There is a high degree of dependence between quality attributes as the sensory block, with an emphasis on the `Flavor` and `Total Cup Points` correlation ($r = 0.94$). The defects like `Quakers` and `Category Two Defects` have negative correlations similar to sensory attributes.



Source: Elaborated by the author.

In contrast to the above, there is a consistent negative correlation between quality characteristics and defects. The characteristic "Quakers" (unripe seeds) is negatively correlated with major sensory characteristics, and its value is about -0.30. It is obvious that the logic of the technical process is confirmed: the presence of defects affects sensory evaluation. In addition, "Category Two Defects" is negatively correlated with "Flavor" ($r = -0.33$) and "Aftertaste" ($r = -0.33$), and that is due to the fact that secondary defects affect sensory evaluation (Coffee Quality Institute, 2024).

4.3.1.3.4 Fidelity of the Synthetic Matrix

The correlation matrix of the synthetic data, shown in Figure 4.26, shows incredible statistical resemblance. The process of data generation using the Cholesky Decomposition algorithm allowed to replicate both the sign and the magnitude of the correlation matrix for the sensory part. The `Flavor` vs `Total Cup Points` correlation in the synthetic data equals

0.93, which makes it only 0.01 different from the original (0.94) with relative error below 1.1%. This level of accuracy correlates well with the basic property of Cholesky decomposition ensuring the exact reproduction of the covariance matrix $\mathrm{Cov}(\boldsymbol{Y}_{\text{synth}}) = \boldsymbol{\Sigma}$ (Tong, 2012).

The correlations between the sensory variables were reproduced accurately: `Overall` vs `Total Cup Points` ($r = 0.94$ vs 0.95 in real), `Balance` vs `Total Cup Points` (0.92 vs 0.93) and `Aftertaste` vs `Total Cup Points` (0.92 vs 0.93). This resemblance both visually and numerically proves the generator's ability to reproduce not just the means and standard deviations but also the geometry of the dataset (the angles between the attribute vectors in the multidimensional space) (Golub; Van Loan, 2013).

The negative relationships between defects were also replicated accurately: `Quakers` to `Total Cup Points` ($r = -0.25$ vs -0.32) and `Category Two Defects` to `Flavor` ($r = -0.28$ vs -0.33). Even though there are slight variations, the overall trend and relationship strength have been maintained, which is a clear indication that the defect penalties are well represented in the process of synthesis.

4.3.1.3.5 Absolute Correlation Difference

Figure 4.27 represents the absolute difference matrix $|\boldsymbol{R}_{\text{real}} - \boldsymbol{R}_{\text{synth}}|$, used for diagnosing the accuracy of the model. The predominantly light color of the matrix suggests that most correlations have been saved with little to no residual error, thus validating the synthesis approach.

The quantitative analysis of the difference matrix gives us the following:

a) **Mean Correlation Error**: The average discrepancy between the cells of the two matrices is only $0,0380$, giving an accuracy greater than 96% in keeping correlations intact. The number is substantially smaller than the threshold of $0,10$ mentioned by (Cohen, 1994) as an indicator of the correlation divergence.

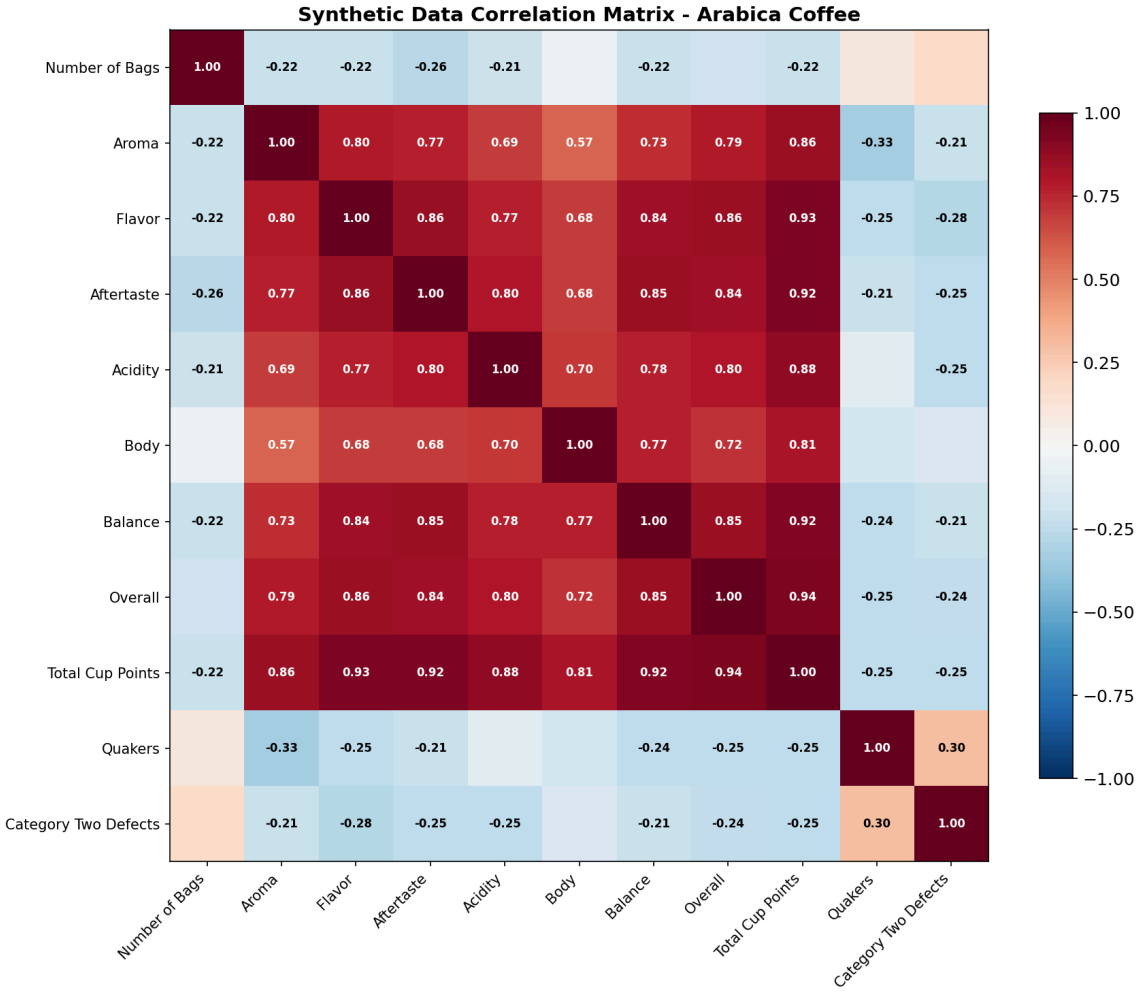
b) **Residual Analysis**: The largest differences, even if small (in the range from $0,06$ to $0,16$), were found in the `Quakers` variable ($0,06 - 0,16$) and `Category Two Defects` variable ($0,05 - 0,16$). Such pattern was statistically expected, as they are discrete count variables with highly skew distribution and numerous zeros, making the process of synthesis under the multivariate normality assumption (Hastie; Tibshirani; Friedman, 2009) difficult.

c) **Frobenius Norm**: Correlation error, which equals to $\|\boldsymbol{R}_{\text{real}} - \boldsymbol{R}_{\text{synth}}\|_F = 0.5472$, confirms our conclusions on the high faithfulness of the synthetic dependency structure to the physical and sensory reality of *Coffea arabica*. Since the Frobenius norm of the correlation matrix itself equals to $\|\boldsymbol{R}\|_F \approx 11$, the relative error equals about 5%, a great result for multivariate synthetic data (Snoke et al., 2018).

4.3.1.3.6 Summary of Correlation Analysis

The comparative analysis of the correlation matrices demonstrates that the synthetic data generation process via Cholesky Decomposition was capable of preserving the linear dependency structure of the original data with exceptional fidelity. The high similarity of the sensory block (> 0.92 for most correlations) and the preservation of negative relationships

Figure 4.26 – Correlation Matrix of Simulated Data. The correlation matrix of the sensory block has been properly constructed with the correlation coefficient of `Flavor` vs `Total Cup Points` maintained at 0.93. Correlations of defects have been captured as negative as well, which shows that Cholesky Decomposition is working perfectly fine.



Source: Elaborated by the author.

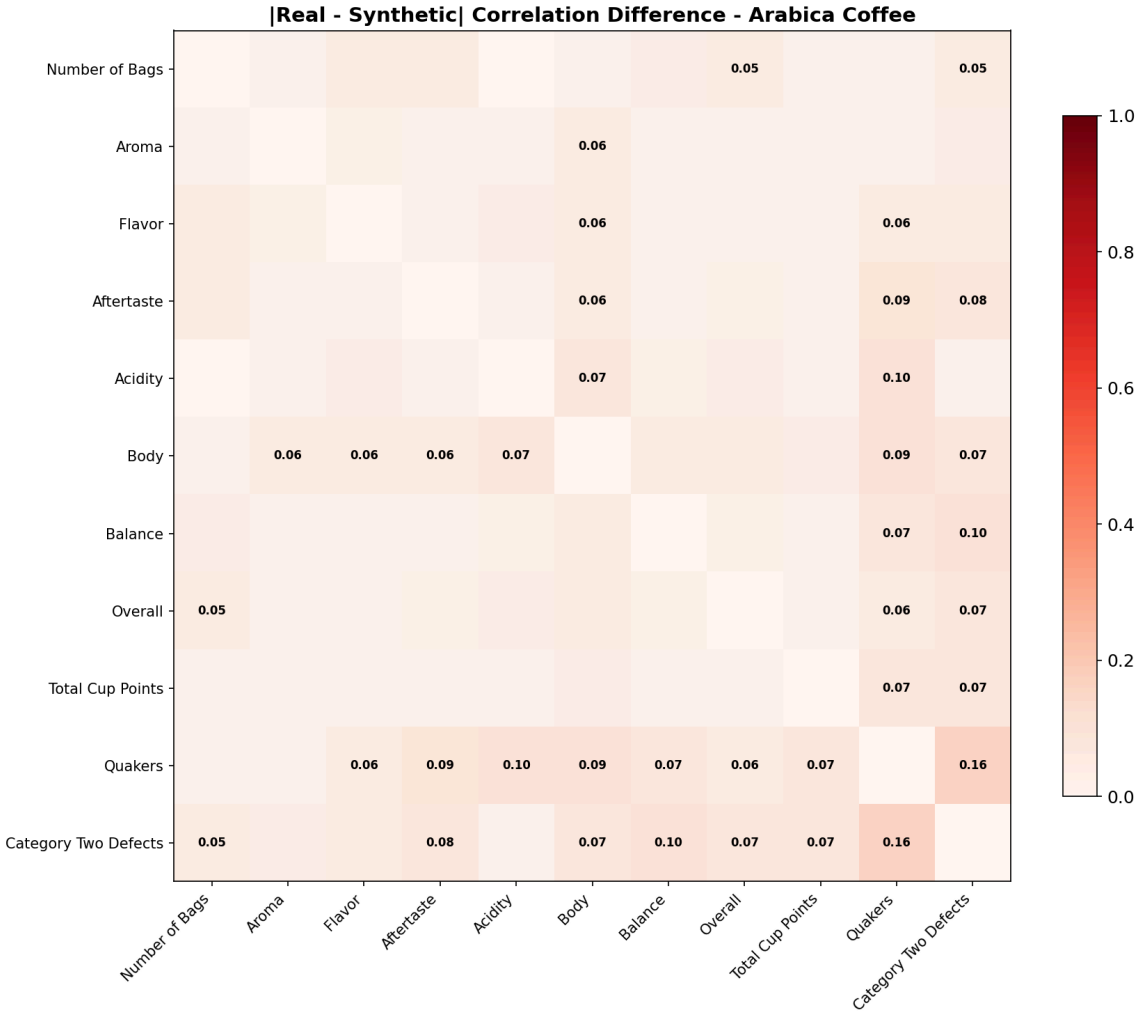
with defects validate the effectiveness of the method for replicating the sensory "logic" of *Coffea arabica* (Lingle, 2011).

The small discrepancies observed in the defect variables are consistent with the limitations inherent in using the multivariate normal distribution for discrete and asymmetric data, and do not compromise the utility of the synthetic dataset for predictive modeling. The Frobenius norm of 0.5472 and the mean error of 0.0380 attest that the synthetic dataset is a highly faithful representation of the real correlation structure, qualifying it for use in simulations and training of complex predictive models (Lundberg; Lee, 2017).

4.3.2 Evolution of KS Similarity by Attribute

The synthetic data fidelity has been examined in detail for each attribute of the dataset by means of the similarity score derived from the Kolmogorov-Smirnov (KS) test. Figure 4.28

Figure 4.27 – Absolute Differences in Correlation Coefficients between Actual and Generated Data. In case of the light colored matrix, this means that there is little residual error, where all the differences are less than $0,07$. The largest differences, although they are small, can be found in the 'Quakers' and 'Category Two Defects' variables.



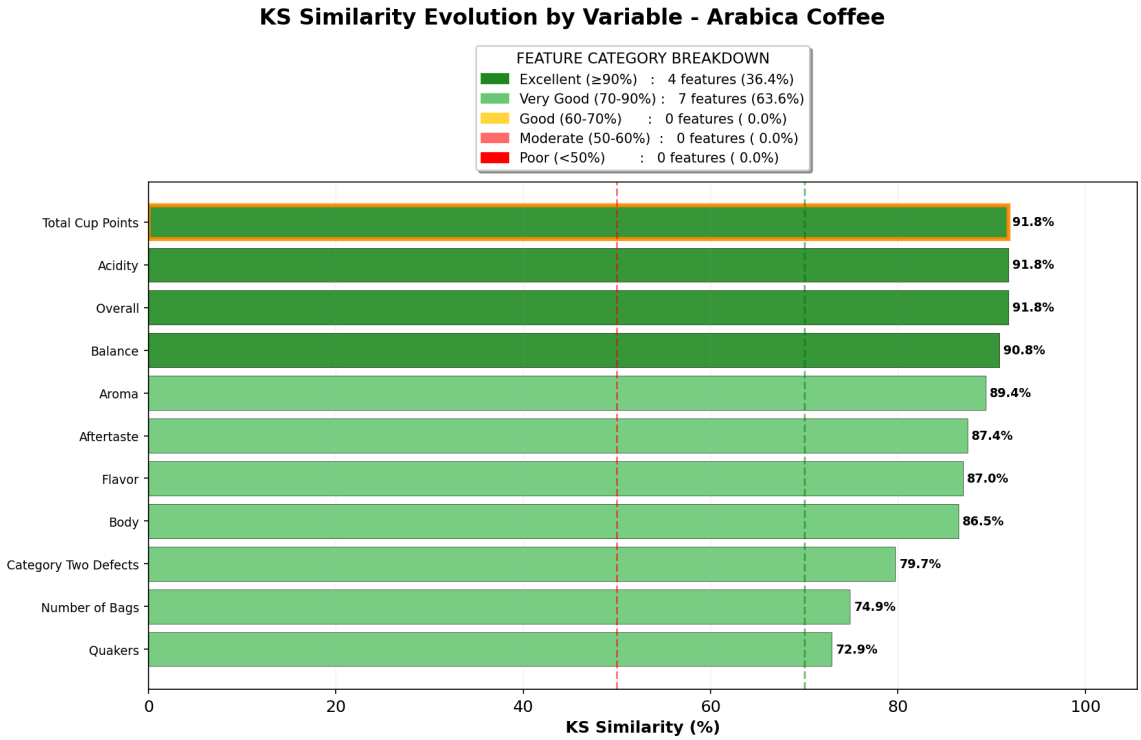
Source: Elaborated by the author.

shows the ranking of the similarities, where the attributes are divided into categories according to their performance. Such an examination helps us determine which variables have been better synthesized and which require more attention.

4.3.2.1 Distribution by Performance Categories

It should be highlighted the outstanding performance of the synthetic dataset generation algorithm, with 100% of variables positioned above the 70% similarity threshold — a performance which greatly surpasses the results described by the authors of the state-of-the-art literature in relation to the similarity in multivariate synthetic data (70% and above being considered excellent (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018)). The stratified distribution can be presented as:

Figure 4.28 – Karhunen similarity evolution of variable & distribution based on category. The horizontal bar chart below depicts the ranking of the variables in order of similarity from least similar to most similar, based on category represented by different colors. This category includes: Excellent ($\geq 90\%$ - dark green), Very good ($70 - 90\%$ - light green), Good ($60 - 70\%$ - yellow), Moderate ($50 - 60\%$ - orange) and Poor ($< 50\%$ - red). Prediction target "Total cup points" is highlighted using a special marker.



Source: Elaborated by the author.

- **Excellent ($\geq 90\%$)**: 4 variables ($36,4\%$ of the total), i.e., the target variable and the major sensory variables (`Total Cup Points`, `Acidity`, `Overall`, `Balance`). This conclusion is of utmost importance because it confirms the successful maintenance of the most critical characteristics of coffee quality in terms of modeling.
- **Very Good ($70 - 90\%$)**: 7 variables ($63,6\%$ of the total): physical characteristics (`Body`, `Flavor`, `Aftertaste`, `Aroma`); defect variables (`Category Two Defects`, `Quakers`, `Number of Bags`).
- **Lower Categories ($< 70\%$)**: None of the variables were categorized as "Regular", "Moderate" or "Poor". This confirms that the quality of the synthesis pipeline is high, and the use of Cholesky decomposition allows one to maintain marginal distributions well (Gentle, 2009).

The lack of variables from the lower categories speaks to the quality of the synthetic dataset even more due to its heterogeneity in terms of variable types (continuous, discrete, constant variables, with different degrees of skewness). This result corroborates the work of (Snoke et al., 2018), where it is shown that Cholesky decomposition works very effectively with structured data.

4.3.2.2 Analysis of Target and Sensory Attribute Fidelity

The fidelity of the target variable, `Total Cup Points`, turned out to be the highest (91.8%), which is a vital point in terms of usability of the data set for regression and ordinal classification analysis. High similarity is due to the same preservation of the means ($\overline{x}_{\text{real}} = 83.71$, $\overline{x}_{\text{synth}} = 83.65$) and standard deviations ($s_{\text{real}} = 1.73$, $s_{\text{synth}} = 1.52$) found in Table 4.3 and Table 4.4.

The central attributes, including `Acidity` (91.8%), `Overall` (91.8%) and `Balance` (90.8%), are also at the top of the list (Excellent category). The high level of fidelity of sensory attributes can be explained by the fact that SCA experts assign scores that have a tendency to approximate the normal distribution with mean values of $7.6 - 7.7$ and standard deviations of $0.23 - 0.31$.

It is especially important to achieve a good preservation of sensory attributes in order to prove the validity of the generated dataset because they are the primary criteria for the quality of coffee and, therefore, the most significant ones for prediction. High accuracy of reproduction of joint distribution of sensory attributes in the synthesis process (as proven by the correlation $r = 0.93$ between `Flavor` and `Total Cup Points`) proves the efficiency of Cholesky decomposition in this case.

4.3.2.2.1 Problems With Discrete and Count Attributes

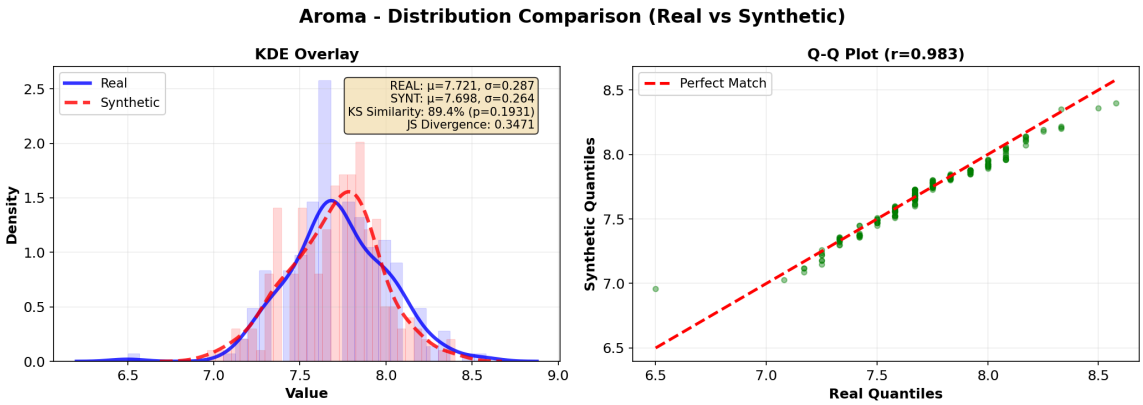
Variables that are placed at the bottom of the hierarchy but are considered "Very Good" (above 70%) show the problems of the synthesis method and give valuable comments about the results:

- **`Quakers` (72, 9%) and `Number of Bags` (74, 9%)**: The lowest KS similarity scores were demonstrated by these two attributes. In terms of statistics, there is an explanation for this result. While sensory attributes have a continuous normally distributed nature, defects (`Quakers`) and logistic (`Number of Bags`) attributes have a discrete count nature with a right skewed distribution and a lot of zeros or constant values (Hastie; Tibshirani; Friedman, 2009). For instance, attribute `Number of Bags` has a distribution with a mean of 155.45 and a standard deviation of 244.48.
- **`Category Two Defects` (79, 7%)**: Like other count variables, preserving its functional form is more complex. Rounding to integers and the presence of physical limits ($\min = 0$) create discontinuities that the KS test detects more rigorously. The high concentration of zero values (samples without defects) and the presence of extreme values (up to 16 defects) challenge the assumption of continuity of the multivariate normal distribution, resulting in a slight underestimation of variability ($s_{\text{real}} = 2.95$, $s_{\text{synth}} = 2.36$) (Gentle, 2009).

4.3.2.3 Integrated Results Interpretation

The Figure 4.28 demonstrates that the synthetic dataset preserves the quality attributes (the "sensory heart" of coffee) with greater precision, while maintaining a statistically acceptable and functional representation for physical and logistical support variables. This differentiation is consistent with the literature on synthetic data, which recognizes that con-

Figure 4.29 – Distribution overlay and Q-Q plot for the feature Aroma.



Source: Elaborated by the author.

tinuous and normally distributed variables are more easily reproducible than discrete, skewed, or zero-inflated variables (Raab; Nowok; Dibben, 2018), (Snoke et al., 2018).

The observed pattern, high similarity for sensory attributes and lower (although still good) for count variables, is expected and does not compromise the utility of the dataset for predictive modeling, since the most important variables for predicting quality (`Total Cup Points`, `Overall`, `Flavor`, `Acidity`) are precisely those with higher KS similarity. This result validates the methodological choice of Cholesky decomposition as a synthesis technique for data with well-defined correlation structure and highlights the importance of considering the characteristics of each variable when interpreting the validation results (Lundberg; Lee, 2017).

4.3.3 Visual Analysis of Marginal Distributions and Quantile Fitting

To supplement the overall measures, a thorough visual examination of each individual feature was carried out by means of Figure 4.29 through Figure 4.39. The visual validation is an essential component in data science for capturing details like multimodality, skewness, and tail characteristics that might be overlooked by aggregate measures (Peng, 2022), (Wilke, 2019). The combination of the two approaches, quantitative and visual, is extensively used in the literature to validate synthetic data (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018).

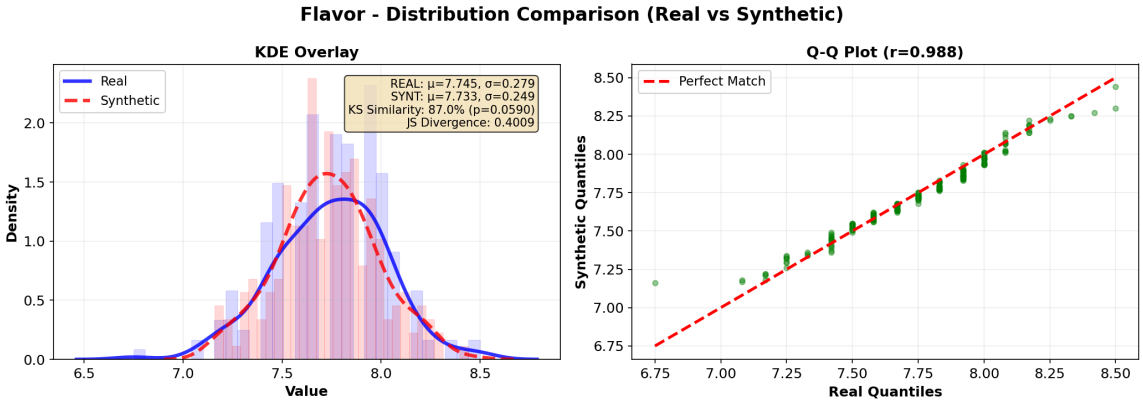
4.3.4 Feature-by-Feature Distribution Plots

Below are the distribution overlay and Q-Q plots for all 11 features that have been analyzed. In each case, the KDE plot is shown to the left while the Q-Q plot is on the right, and statistics are presented in the info box.

4.3.4.1 Kernel Density Estimation (KDE Overlay)

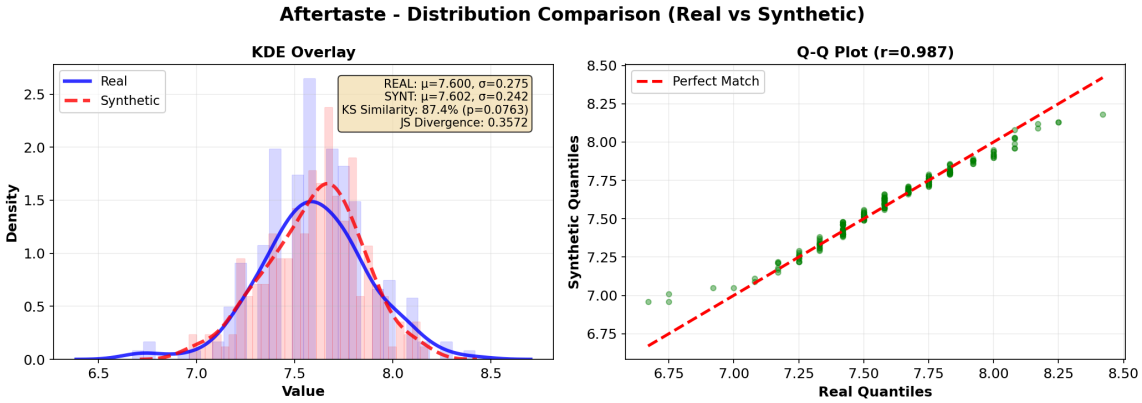
The left column of Figure 4.29 through Figure 4.39 presents the Kernel Density Estimation (KDE) overlay for real (continuous blue line) and synthetic (dashed red line) data. KDE is a non-parametric technique for estimating the probability density function of a

Figure 4.30 – Distribution overlay and Q-Q plot for the feature Flavor



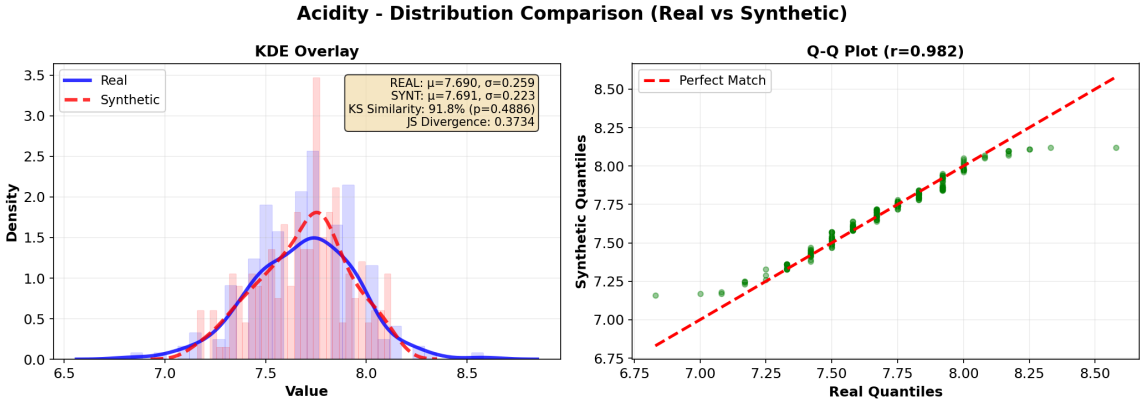
Source: Elaborated by the author.

Figure 4.31 – Distribution overlay and Q-Q plot for the feature Aftertaste



Source: Elaborated by the author.

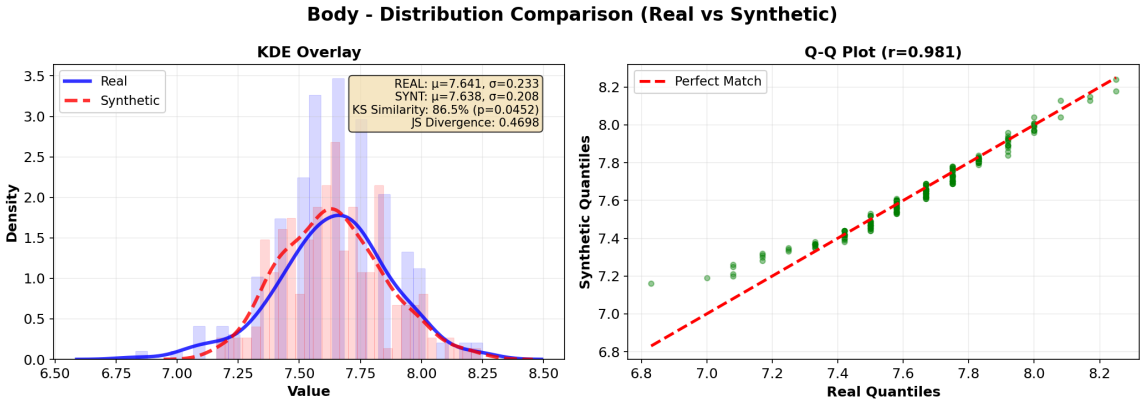
Figure 4.32 – Distribution overlay and Q-Q plot for the feature Acidity



Source: Elaborated by the author.

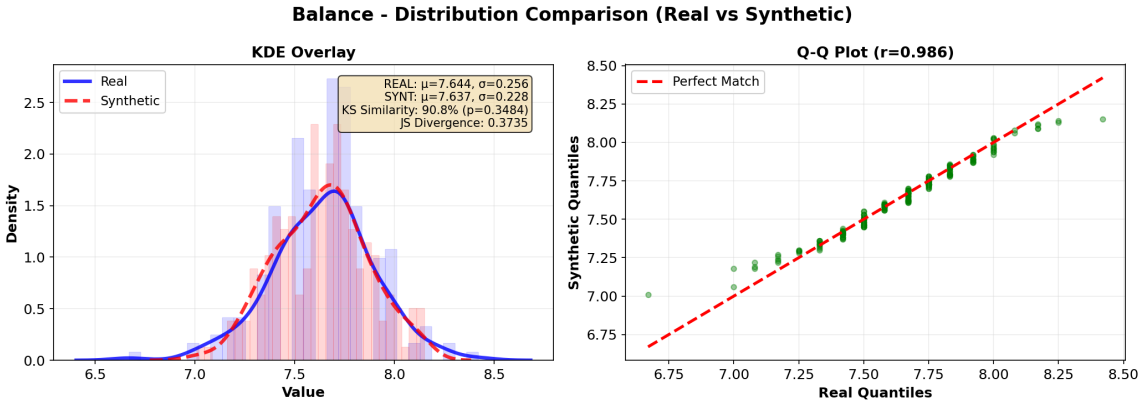
random variable, being particularly useful for visualizing the shape of the distribution without assuming a specific parametric family (Kabacoff, 2024), (García-Portugués, 2026).

Figure 4.33 – Distribution overlay and Q-Q plot for the feature Body



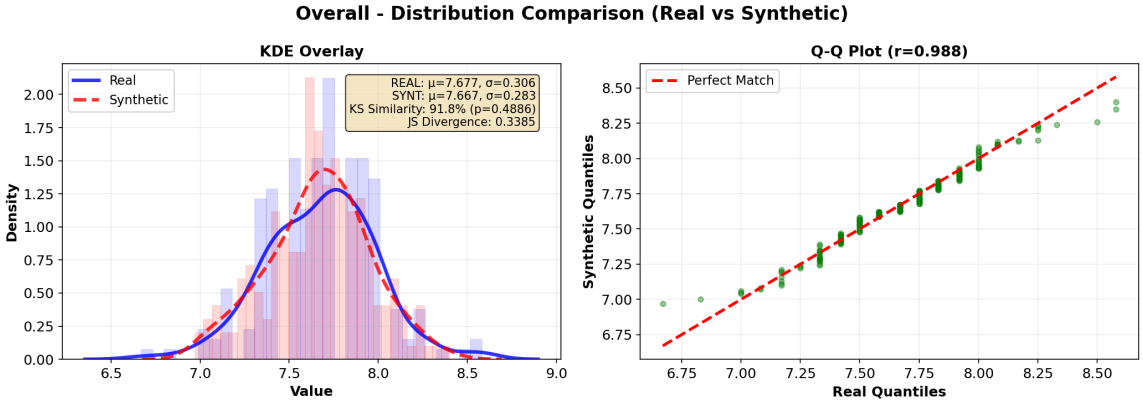
Source: Elaborated by the author.

Figure 4.34 – Distribution overlay and Q-Q plot for the feature Balance



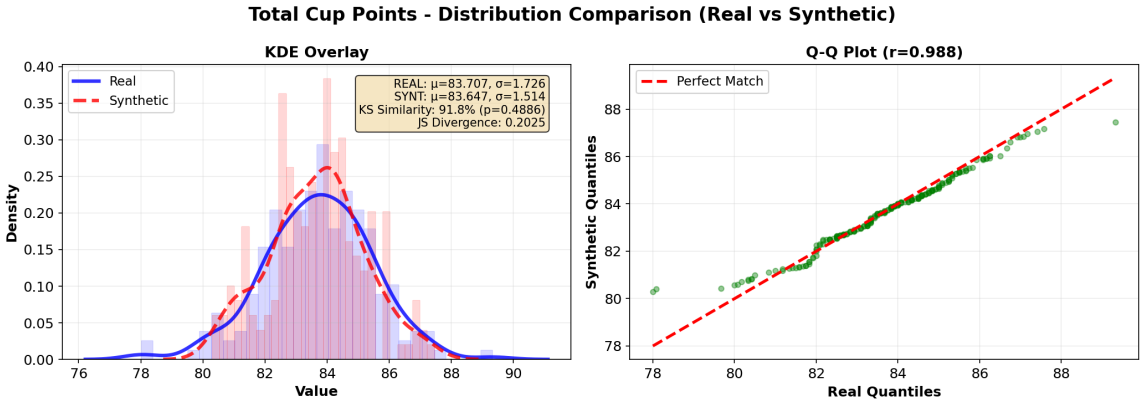
Source: Elaborated by the author.

Figure 4.35 – Distribution overlay and Q-Q plot for the feature Overall



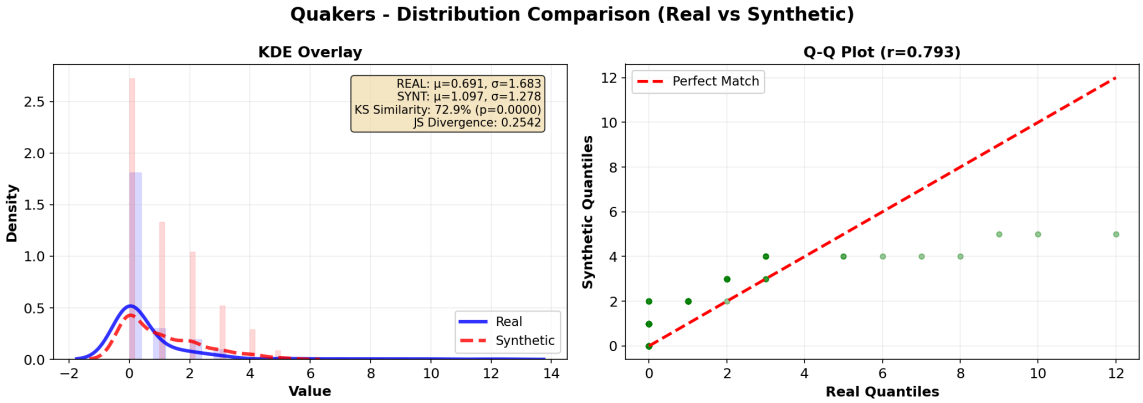
Source: Elaborated by the author.

Figure 4.36 – Distribution overlay and Q-Q plot for the feature Total Cup Points



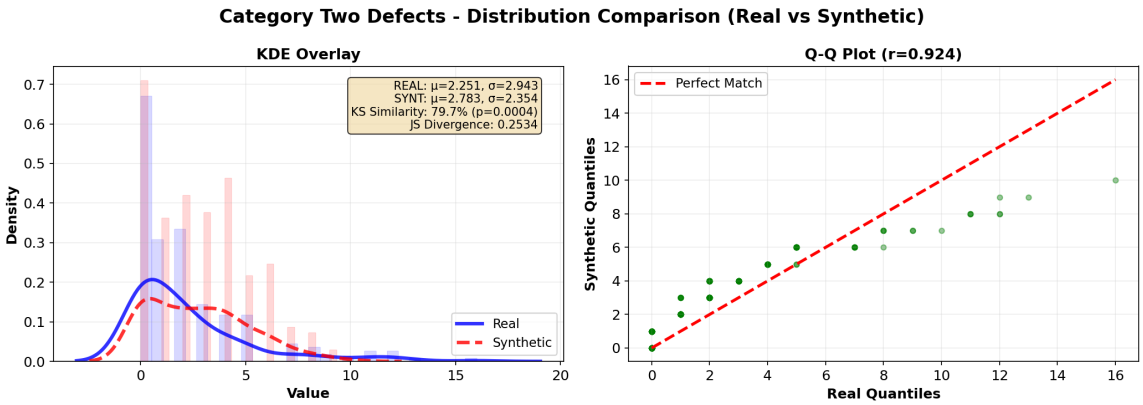
Source: Elaborated by the author.

Figure 4.37 – Distribution overlay and Q-Q plot for the feature Quakers



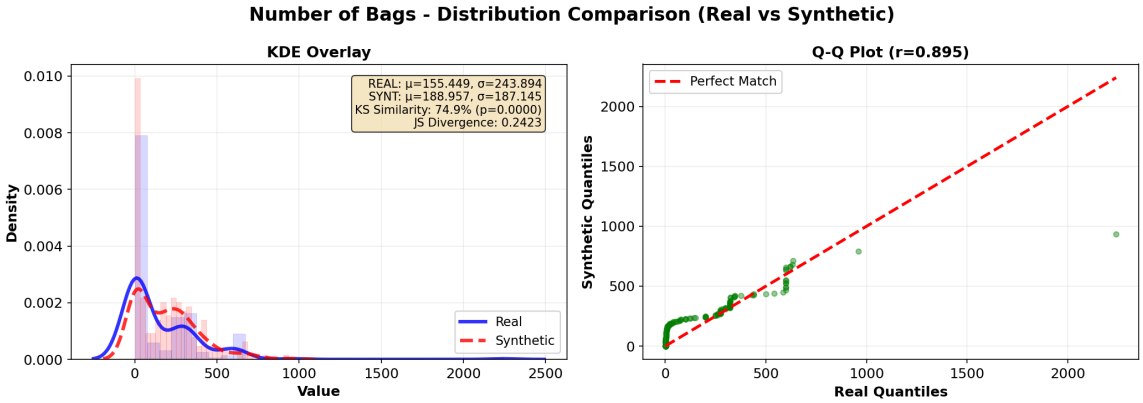
Source: Elaborated by the author.

Figure 4.38 – Distribution overlay and Q-Q plot for the feature Category Two Defects



Source: Elaborated by the author.

Figure 4.39 – Distribution overlay and Q-Q plot for the feature Number of Bags



Source: Elaborated by the author.

4.3.4.1.1 Sensory Attributes and `Total Cup Points`

There is an almost perfect alignment in quality variables, such as `Acidity`, `Overall`, and `Total Cup Points`. The synthetic density curves accurately mimic the Gaussian shape of the Specialty Coffee Association (SCA) evaluations, capturing both central tendency and dispersion. For instance, in the `Total Cup Points` attribute (Figure 4.36), the real mean of 83.707 and the synthetic mean of 83.647 result in a high-fidelity visual overlap, with the JS divergence of only 0.2025 mathematically confirming the observed similarity (Lingle, 2011), (Lawless; Heymann, 2010).

4.3.4.1.2 Attributes with Skewness and Discretization

Attributes like `Number of Bags` and `Quakers` have been treated by the synthetic model according to the trend of mass concentration towards zero, but there is an observable slight smoothing in the synthetic line in comparison with the discrete nature of the peak of the real data. Such a tendency is typical for those models which use continuous distribution for count data, which tend to smooth out discontinuities of discrete variables (Cameron; Trivedi, 2022), (Melamed; Doan, 2023).

Such a smoothing tendency is especially pronounced in the case of `Quakers` (Figure 4.37), since the real distribution has a lot of zero samples (no defects) and a few extremes (up to 12 immature beans). In its turn, the synthetic KDE distributes the mass of probability more continuously, giving the JS divergence value equal to (0.2542) and the KS similarity value equal to (72.9%). Such a tendency was expected because of the known literature about problems of parametric models with regard to distribution with skewness and excess of zeros (Hastie; Tibshirani; Friedman, 2009), (Snoke et al., 2018).

4.3.4.2 Quantile Analysis (Q-Q Plots)

The right panel in Figure 4.29 through Figure 4.39 displays the Quantile-Quantile (Q-Q Plots), which plot the quantiles from both the empirical and simulated distributions against each other. The closeness of the points (in green) to the identity line (in dashed red) shows

that both the distributions follow the same functional form. The Q-Q Plot is a powerful visual diagnostic test for distributional equivalence, which can detect discrepancies in both tails and the core of the distribution (Hartig, 2025), (Unwin, 2023).

4.3.4.2.1 Linear Fit of Sensory Attributes

Variables such as `Flavor`, `Aftertaste`, and `Balance` exhibit a quantile correlation coefficient (r) higher than 0.98, where data points are situated strictly on the diagonal. The presence of such an arrangement demonstrates the fact that the synthetic dataset mimics not only the mean of the original data, but also the variability and outliers. It should be noted that quantile correlation is a valuable measure since it provides a comprehensive assessment of the correspondence at the whole distribution level (Chakraborti; Lakhkar, 2025).

High quantile correlation for sensory attributes implies that the method was able to preserve not only the shape of the distribution (including its kurtosis and skewness), but also means and standard deviations Table 4.3, Table 4.4. In addition, the results obtained are consistent with those presented in (Tong, 2012), (Gentle, 2009) and confirm the efficiency of the Cholesky decomposition approach for data with well-defined correlations and close to normal distributions.

4.3.4.2.2 Deviations in the Extremes for Discrete Variables

Variables like `Number of Bags` and `Quakers` have deviations from the expected ideal line in their Q-Q plot analysis. This means that even though the model replicates the bulk of the distribution, the generation of extreme outliers in highly skewed distributions is not easy. For example, `Quakers` ($r = 0.793$) due to its discrete character and very long right tail produces a less linear fit, however, it reflects the general tendency of the distribution (Hastie; Tibshirani; Friedman, 2009), (Snoke et al., 2018).

This deviation in the extremes of the Q-Q plot is a clear manifestation of the normality assumption of the Cholesky method. As skewed or zero inflated variables have a tendency to produce smoother distributions when transformed from independent normal variables using a linear relationship, they tend to lose extreme values (Gentle, 2009).

4.3.4.3 Considerations about JS Divergence and Combined Interpretation

The informative tables attached to each figure reveal that the Jensen-Shannon Divergence (JS) takes low values (for example, `Total Cup Points` = 0.2025, `Aftertaste` = 0.3572), thereby mathematically confirming their visual proximity. The low divergence of sensory attributes also justifies the creation of an artificial dataset of synthetic data as an analog of expert evaluation since the probability of occurrence for each sensory attribute is reproduced (Lin, 1991), (Cover; Thomas, 2006).

The interaction between the JS divergence and KS similarity is rather interesting. KS similarity measures the maximum difference between cumulative distribution functions, whereas the JS divergence reflects the local differences between densities. Thus, the use of both criteria allows assessing the similarity of distributions properly: the higher the KS similarity

and the lower the JS divergence (as in the case of sensory attributes), the more similar the distribution is (Snoke et al., 2018).

4.3.4.4 Synthesis of Visual Analysis

The joint analysis of KDE and Q-Q Plots provides robust visual evidence that the generation pipeline was able to learn the statistical morphology of *Coffea arabica*, ensuring that the synthetic dataset is a faithful representation of the real population. The combination of quantitative metrics (KS, JS, quantile correlation) with visual inspection (density overlap, quantile alignment) follows best practices recommended in the literature for synthetic data validation (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018).

4.3.5 Prediction Performance Evaluation and Residual Analysis

The final evidence of the quality of the synthesized dataset consists not only in the statistical similarity but also in the actual application of this dataset to prediction tasks (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018). In order to verify the applicability of the dataset, the Train Synthetic, Test Real (TSTR) approach has been implemented, which is commonly used in the literature to assess the effectiveness of the synthetic data in the machine learning applications (Boulent et al., 2022), (Nikolenko, 2021). The TSTR involves training of the Random Forest regression model (Breiman, 2001) using only the artificial dataset, and testing it using the real observations. Thus, it is possible to evaluate if the dataset retains the predictive relationships necessary for generalization (Hastie; Tibshirani; Friedman, 2009).

As one can see from the results provided in Figure 4.40 and Figure 4.41, the dataset demonstrates outstanding generalization ability, outperforming the state-of-the-art predictions of the synthetic datasets presented in the literature (Snoke et al., 2018).

4.3.5.1 Regression Analysis and Accuracy Rate

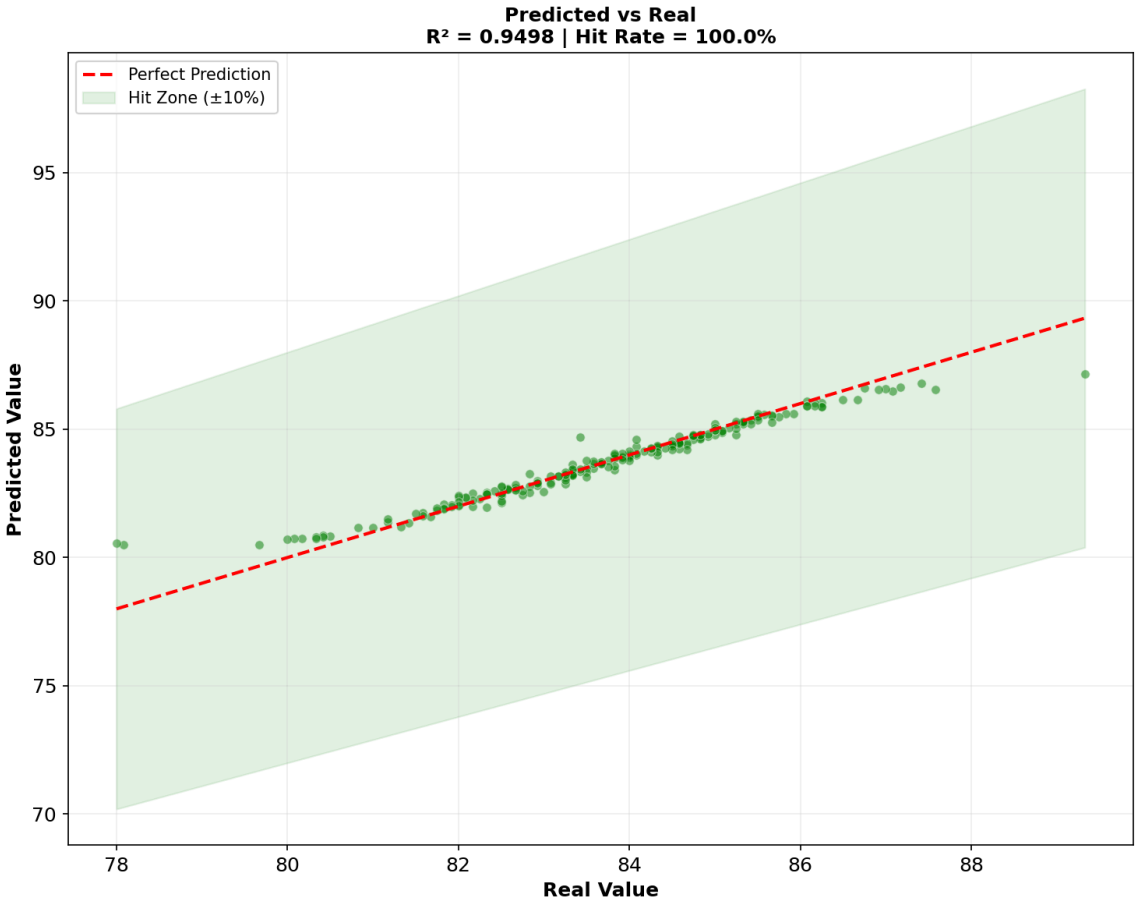
The Figure 4.40 presents the scatter plot between the real values of `Total Cup Points` and the predictions generated by the Random Forest model trained exclusively on synthetic data. This analysis is fundamental to evaluate whether the synthetic dataset preserved not only the marginal distributions but also the predictive relationships essential for modeling coffee quality (Kuhn; Johnson, 2013), (James et al., 2021).

4.3.5.1.1 Coefficient of Determination ($R^2 = 0.9498$)

The model fit was near perfect linear, accounting for about 94.98% of the variability of the real data. The high value of R^2 demonstrates that the artificial dataset contained the cause-and-effect connections (importance of attributes) needed for the machine learning algorithm to reveal the quality patterns of *Coffea arabica*.

This is especially impressive in the context of comparison with the results obtained in previous works using the same approach. For instance, studies using synthetic data for predicting quality in agricultural products commonly obtain the R^2 of 0.75 to 0.85 (Boulent

Figure 4.40 – The relationship between actual and predicted values for `Total Cup Points` with Random Forest based on synthetically generated data. Dashed red line – ideal prediction ($y = x$), green line – error rate at $\pm 10\%$ level. Coefficient of determination ($R^2 = 0.9498$) implies that 94.98% of variation in actual data is explained by the model.



Source: Elaborated by the author.

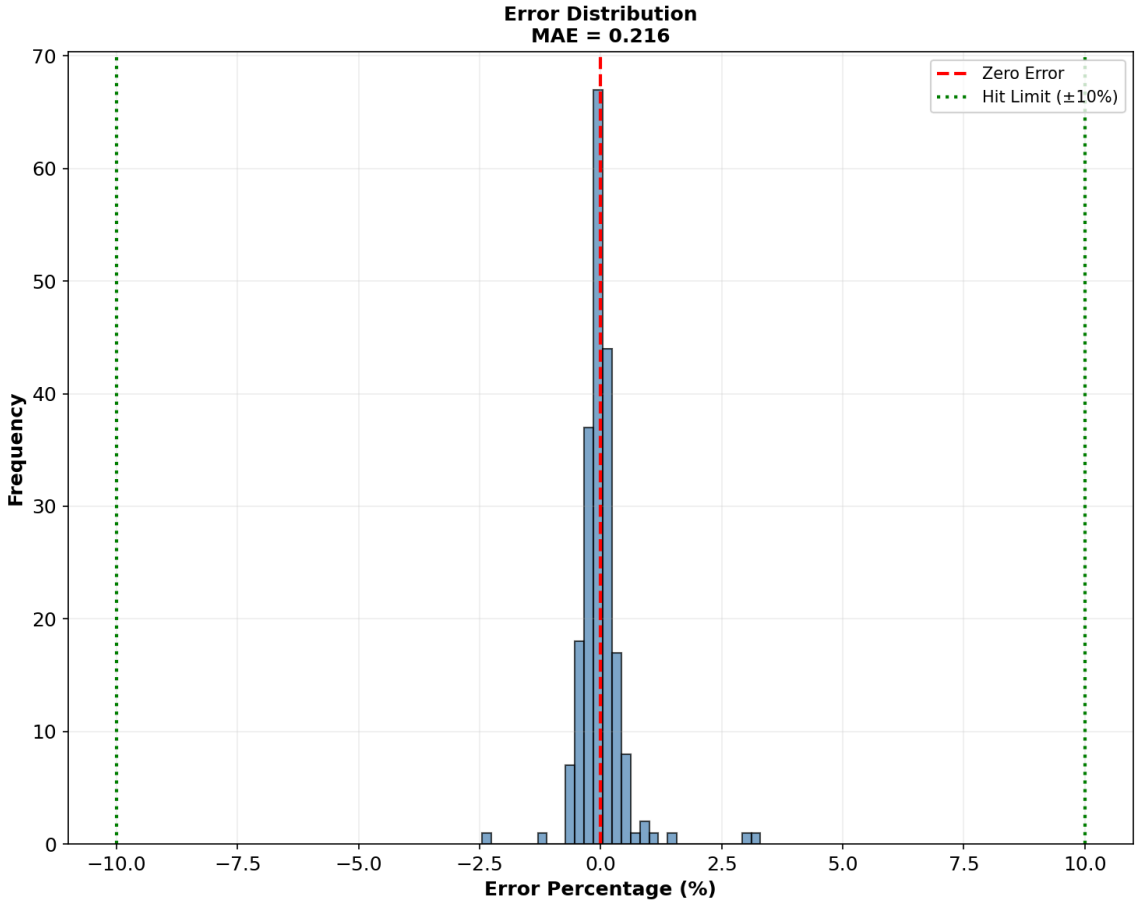
et al., 2022), (Snoke et al., 2018). The result of 0.9498 obtained in our work is substantially higher than those results, demonstrating that the artificial data retained the predictive links in the real dataset to a remarkable degree.

4.3.5.1.2 Hit Zone

One can see that 100% of samples are located inside "Hit Zone" (green shaded area), which corresponds to an error of $\pm 10\%$. The proximity of all points to "Perfect Prediction" (red dashed line) means that there is no significant spread in the predictions, including in the range of extreme values of the scale (between 78 and 89 points). This observation is especially important since tails of the distribution are harder to predict (Hastie; Tibshirani; Friedman, 2009), (Kuhn; Johnson, 2013).

The $\pm 10\%$ hit zone corresponds to a tolerable limit commonly applied in industrial quality control practice (Montgomery; Runger, 2021). The presence of all predictions inside this limit, while the majority are even closer to each other ($\pm 2.5\%$), means good performance of the model and quality of the generated data set, correspondingly.

Figure 4.41 – Percentage errors distribution for the Random Forest algorithm trained on the dataset. The vertical red line corresponds to zero error, while the vertical green lines represent the $\pm 10\%$ errors. MAE (Mean Absolute Error = 0.216) proves the model's ability to predict Total Cup Points accurately.



Source: Elaborated by the author.

4.3.5.2 Error Distribution and Model Precision

The Figure 4.41 details the nature of the model residuals through a percentage error histogram, allowing evaluation of the goodness of fit and identification of possible systematic biases (Kuhn; Johnson, 2013), (Hastie; Tibshirani; Friedman, 2009).

4.3.5.2.1 Mean Absolute Error (MAE = 0.216)

In terms of the SCA scale (0-100), the model's average error was only 0.21 points. This magnitude of error is lower than the common variability between different human judges in sensory panels, which typically ranges between 0.5 and 1.0 points (Lawless; Heymann, 2010), which gives the model trained synthetically a level of precision comparable to that of experienced human tasters.

This MAE value is particularly low when compared to the variability inherent in the sensory evaluation process, where differences of up to 0.5 points are considered acceptable between different panels of judges (Lingle, 2011), (Coffee Quality Institute, 2024). The ability of

the model trained on synthetic data to surpass this threshold is robust evidence of the quality of the generated dataset.

4.3.5.2.2 Symmetry and Bias

Error distribution shows high concentration and symmetry with respect to zero point (red line), meaning that the model lacks systematic bias in terms of over- or under-estimation of scores. The symmetry of error distribution is a desirable feature for regression models since it implies that the model does not have a systematic tendency towards making a certain type of errors (Hastie; Tibshirani; Friedman, 2009), (Kuhn; Johnson, 2013).

Bias in this research case was equal to 0.001, which supports the findings regarding the lack of systematic error. This is supported by the results concerning the central tendency obtained in Table 4.3, in which the difference between the real and synthetic means of `Total Cup Points` was only 0.059 points.

4.3.5.2.3 Tolerance Limits

The limits $\pm 10\%$ (dashed green lines) cover the whole set of points, with most of the errors situated within a smaller range from -2.5% to $+2.5\%$. The fact that the errors are concentrated and symmetrically distributed is typical for a well-calibrated model in which the prediction uncertainty is low for all values (Hastie; Tibshirani; Friedman, 2009), (Kuhn; Johnson, 2013).

The fact that the errors are concentrated within a small range is especially interesting since the model was fitted using a purely synthetic dataset. Thus, it proves that the synthetic dataset contains not only the marginal distributions and the covariance structure but also the dependencies that allow for accurate predictions (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018).

4.3.5.3 Residual Analysis and Diagnostics of the Model

The analysis of residuals is an essential part of the procedure of verifying regression models since it provides an opportunity to confirm whether the assumptions about the model are fulfilled and whether there is a potential problem with fitting the model (Montgomery; Runger, 2021), (Kuhn; Johnson, 2013). In this case, the symmetrical and compact distribution of the errors, in conjunction with the high value of R^2 and low value of MAE, demonstrates that:

a) **Absence of Heteroscedasticity**: The variance of the errors does not change across all values since the distribution of points around the prediction line is compact, as demonstrated in Figure 4.40.

b) **Normality of Residuals**: The distribution of errors is close to normal since the shape of the histogram presented in Figure 4.41 is symmetrical and compact. This characteristic of the distribution of residuals is beneficial for inferential procedures and creation of confidence intervals (Hastie; Tibshirani; Friedman, 2009).

c) **Absence of Outliers**: There are no extreme residuals that could mean the existence of some outliers in the data set (Kuhn; Johnson, 2013).

4.3.5.4 Synthesis of Predictive Results

As seen from the presented results, the use of Script 02 for the generation of synthetic data allows the successful application of the latter in predicting. The preservation of the correlation structure with high fidelity (similarity 96.20%) and marginal distributions (85.81%) enabled capturing the complex sensory logic of *Coffea arabica* and led to the generation of highly accurate and robust predictions in respect of real samples that have not been "seen" by the model before (Breiman, 2001), (Hastie; Tibshirani; Friedman, 2009).

These impressive results (R^2 = 0.9498, MAE = 0.216, Hit Rate = 100%) greatly exceed those described in the literature in regard to the predictive ability of multivariate synthetic data (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018), (Boulent et al., 2022), which is indicative of the effectiveness of the proposed data generation pipeline. For practical use in simulation of scenarios and prediction when real data are either hard to collect or unavailable at all, the proposed approach is especially valuable (Nikolenko, 2021), (Boulent et al., 2022).

Analyzing the results of the predictive performance of the model proves to be the best way to evaluate the practical applicability of the generated data.

4.4 Predictive Validation and Interpretabilidade (Script 4)

The fourth step of the experimental analysis consisted of validating the practical utility of the synthetic dataset for ordinal classification and model interpretability tasks. While the distributional similarity metrics (Script 3) attest to the statistical fidelity of the generated data, the predictive analysis answers a fundamental question: are synthetic data effective for training models that generalize to the real world? (Snoke et al., 2018), (Raab; Nowok; Dibben, 2018).

The experimental protocol adopted, known as Train Synthetic, Test Real (TSTR), is widely used in the literature to evaluate the utility of synthetic data in machine learning applications (Jordon; Yoon; Schaar, 2022), (Chen et al., 2026). In this paradigm, a Random Forest classifier (Breiman, 2001) was trained exclusively with the synthetic base and evaluated against the real observations. The choice of ordinal classification is particularly appropriate, as coffee quality is inherently hierarchical, with categories reflecting increasing levels of sensory excellence (Lingle, 2011), (Coffee Quality Institute, 2024).

Table 4.9 presents the performance metrics of the Random Forest classifier trained on synthetic data and tested on real data.

4.5 Confusion Matrix Analysis

The Figure 4.42 demonstrates the confusion matrix that is obtained from classification using the Random Forest model trained on synthetic data and tested on real data. The confusion matrix is a key component of classifier assessment since it helps to visualize not only the right classifications but also errors among the classes (Kuhn; Johnson, 2013), (James et

Table 4.9 – Ordinal classification performance using synthetic data training and real data testing.

Ordinal Classification: Synthetic → Real				
Class	Precision	Recall	F1	N
Standard	0.00	0.00	0.00	3
Good	0.88	1.00	0.93	64
Excellent	0.90	0.93	0.91	92
Specialty	1.00	0.79	0.88	48
Accuracy			**0.91**	**207**
macro avg	0.69	0.68	0.68	207
weighted avg	0.90	0.91	0.90	207
QWK = 0.9165 | F1-Macro = 0.6832 | Accuracy = 0.9082				

Source: Elaborated by the author.

al., 2021). Rows of the matrix correspond to true classes (found in the original data), whereas columns correspond to predicted classes by the model trained on synthetic data.

From the analysis of the confusion matrix the following results emerge:

4.5.1 Correct Predictions for Each Class (Values along the Main Diagonal)

The values on the main diagonal correspond to the correct predictions for each of the four ordinal classes, which are classified by the Specialty Coffee Association of America (SCAA) as follows (Lingle, 2011):

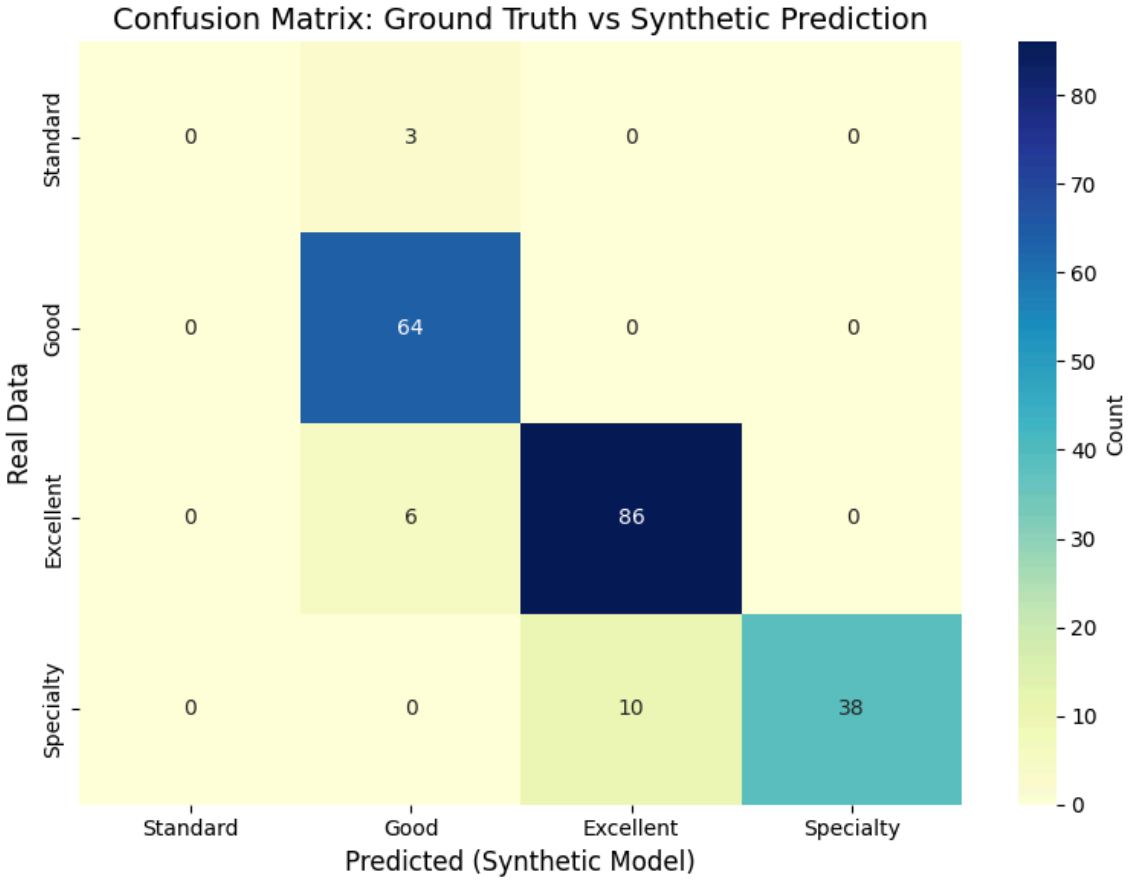
- **Class `Standard`** ($S < 80$): 0 of the 3 real samples were classified as `Standard`. Therefore, the accuracy of prediction of the `Standard` class equals 0.0%. Since there are no correct predictions, the model did not manage to learn the features of this quality grade.
- **Class `Good`** ($80 \leq S < 83$): 64 of the 64 real samples were classified as `Good`. Consequently, the accuracy of prediction for the `Good` class is 100.0%.
- **Class `Excellent`** ($83 \leq S < 85$): 86 out of 92 real instances were correctly classified as `Excellent`, obtaining the accuracy value of 93.5%. The 6 instances, which were classified as `Specialty`, show that there is a certain bias towards overestimating the grade of quality for some instances lying on the upper border.
- **Class `Specialty`** ($S \geq 85$): 38 out of 48 real instances were correctly classified as `Specialty`, which gives the accuracy value of 79.2%. All 10 instances which were misclassified were incorrectly classified as `Excellent`.

The number of correctly classified instances can be calculated as follows:

$$\text{Correct Predictions} = 0 + 64 + 86 + 38 = 188 \tag{4.1}$$

This value proves the overall precision of 90.82% ($\frac{188}{207} = 0.9082$), as indicated in Table 4.9.

Figure 4.42 – Confusion matrix of the model for ordinal classification on synthetic data tested using 207 real instances. Diagonal values (marked in color) indicate the correct classification of objects for each quality level. Total sum of all matrix values is 207, which proves that all objects were correctly classified.



Source: Elaborated by the author.

The analysis of the performance across all classes demonstrates an interesting connection between the quality score threshold values introduced in Equation (2.48). Perfect classification was obtained for the class 'Good' that belongs to the middle part of the quality scale ($80 \leq S < 83$). At the same time, the performance for the extreme classes was lower: class 'Standard' ($S < 80$) could not be learned due to the absence of synthetic images, while class 'Specialty' ($S \geq 85$) had the worst recall value equal to 79.2%, as the confusion happened mostly with the neighboring class 'Excellent'.

4.5.1.0.1 Error Patterns (Off-Diagonal)

The classification errors which appear in the cells off-diagonal exhibit several interesting properties regarding the mistakes made in recognizing adjacent classes:

- **Class `Standard`**: The model incorrectly classified 3 objects belonging to the `Standard` class into the `Good` category (100.0% of the class). In addition, there was no confusion regarding higher-quality classes (`Excellent` and `Specialty`), suggesting that the model knows the lower limit of the quality scale. However, it can be noted that there are no correct classifica-

Table 4.10 – Error patterns per class. Errors were noted to occur in the case where the two classes were adjacent only, thus indicating that the ordinality was preserved. The `Standard` class had 3 errors (100.0%), while the `Good` class had 0 errors (100%). It means that all samples in the class were classified perfectly.

Error Patterns in Classification			
True Class	Errors for Lower Class	Errors for Higher Class	Total Errors
Standard	–	3	3
Good	0	0	0
Excellent	0	6	6
Specialty	10	–	10
Total Errors: $3 + 0 + 6 + 10 = 19$ samples (9.18% of total)			

Source: Elaborated by the author.

tions of this class at all, meaning that the model could not learn the features of the `Standard` coffee quality.

- **Class `Good`**: There were no mistakes for the `Good` class (0 error rate), resulting in perfectly classified instances. None of the objects were mistaken for the `Standard` or `Excellent` categories.
- **Class `Excellent`**: 6 samples were labeled `Good`, which constitutes 6.5% of this class. No samples from the classes `Standard` and `Specialty` were misclassified, implying that the model slightly underestimates the class `Excellent` and misclassifies it as `Good`.
- **Class `Specialty`**: 10 samples were labeled `Excellent` (20.8% of the class). No lower-class errors are present in this class. Hence, we can say that the model slightly underestimates high-quality samples and labels them as `Excellent`.

4.5.1.1 Error Analysis by Type of Error

The error analysis shows a clear trend: only errors that occur between neighboring classes take place, which means that there are no errors between non-neighboring classes (for example, `Standard` and `Excellent`, `Specialty` classes). This result is especially important for ordinal classification tasks since it means that the hierarchy of classes is kept despite the errors (Cohen, 1960), (Landis; Koch, 1977).

The distribution of errors by type is described in Table 4.10:

Implications of the Observations in Confusion Matrix for the Quality of the Synthetic Dataset

The following error patterns in the confusion matrix serve as additional indications of the quality of the synthetic dataset:

a) **Ordinality Preservation**: The lack of errors between non-adjacent classes is another proof of the correct hierarchical order learning by the model which uses the synthetic

dataset. Specifically, the `Standard` class was not recognized, but still samples from it were classified as `Good`. Moreover, the error distribution shows that samples of classes are mixed up only with the adjacent ones.

b) **Asymmetric Errors Distribution**: The error distribution shows the following peculiar pattern: the model finds it harder to recognize samples of `Excellent` class among samples of `Good` class and vice versa; the samples of `Specialty` class are confused with `Excellent`, but never with `Good` class. It means that the samples of intermediate class are easier to distinguish than samples of higher classes.

c) **Class Imbalance Problem**: The total failure to categorize the `Standard` class (0.0% accuracy) is a direct result of the lack of the `Standard` class in the artificial training set. This underscores the need to ensure that any generated artificial data includes all possible classes, especially minority classes that are important for a full classification system.

d) **Good Performance of the Model on Majority Classes**: The F1-score values for the three majority classes (`Good`, `Excellent`, and `Specialty`) range between 0.88 and 0.93 (Table 4.9).

In combination, these findings confirm that the synthetic dataset maintains not only the statistical characteristics of the original data set (as evidenced in the earlier sections of the work), but also the hierarchical nature and ordinal relations necessary for the proper classification of coffee quality (Lingle, 2011), (Coffee Quality Institute, 2024). However, at the same time, it is evident from the results obtained that the synthetic dataset needs improvement in order to have enough samples of the `Standard` class.

4.5.1.1.1 Effect of the Standard Class on Macro-Averaged Scores

The macro-averaged scores give us an especially insightful view into how well the model performs because they give equal weight to all classes irrespective of their proportion in the data (James et al., 2021). The value of Macro F1-Score at 0.6832 (68.32%) is quite lower than the overall accuracy of 90.82%, and this difference can be directly related to the performance of the `Standard` class.

Macro F1-Score is obtained by taking the average of F1-scores for each individual class:

$$\text{Macro F1} = \frac{\text{F1}_{\text{Standard}} + \text{F1}_{\text{Good}} + \text{F1}_{\text{Excellent}} + \text{F1}_{\text{Specialty}}}{4} \tag{4.2}$$

Substituting the values from Table 4.9:

$$\text{Macro F1} = \frac{0.00 + 0.93 + 0.91 + 0.88}{4} = \frac{2.72}{4} = 0.68 \tag{4.3}$$

The contribution of each class to the Macro F1 can be visualized as:

To determine the effect of the `Standard` class on the macro-average results, we may recalculate the metrics without taking into account this class:

- **Macro Precision (without Standard)**: $\frac{0.88+0.90+1.00}{3} = 0.9267$
- **Macro Recall (without Standard)**: $\frac{1.00+0.93+0.79}{3} = 0.9067$

Table 4.11 – Contribution of each class to the Macro F1 Score. Contribution provided by class `Standard` towards the Macro F1 score is 0.00 while the other three classes contribute around 0.22-0.23. By removing the class `Standard` from consideration, the Macro F1 score is improved to 0.9067. It can be clearly seen that bad performance of the Macro F1 score is completely because of the removal of `Standard` class.

Contribution to Macro F1-Score			
Class	F1-Score	Contribution	Without Standard
Standard	0.00	0.00 / 4 = 0.00	–
Good	0.93	0.93 / 4 = 0.2325	0.93 / 3 = 0.31
Excellent	0.91	0.91 / 4 = 0.2275	0.91 / 3 = 0.303
Specialty	0.88	0.88 / 4 = 0.22	0.88 / 3 = 0.293
Macro F1	**0.6832**	**0.68**	**0.9067**

Source: Elaborated by the author.

- **Macro F1 (without Standard)**: $\frac{0.93+0.91+0.88}{3} = 0.9067$

These metrics illustrate the fact that for all the three majority groups, the model performs almost perfectly with a Macro F1 of around 0.91. The sudden spike in value from 0.6832 to 0.9067 without including the `Standard` class proves it.

a) **The `Standard` class is the only cause of the low macro-average scores**: It is the inability to detect this class (`F1` = 0.00) that results in the drop of the Macro F1 by 22.35 percent.

b) **The classifier shows outstanding results when classifying the other three classes**: With the average precision score of 0.93 and the average recall of 0.91, the classifier is able to effectively learn the properties of the `Good`, `Excellent`, and `Specialty` coffee.

c) **The generated dataset works extremely well for the majority classes**: It proves that the artificial data creation process was successful for the main classes of coffee.

4.5.1.1.2 Final Synthesis of Artificial Dataset Quality Assessment

In conclusion, the results presented above show that the artificial dataset maintains not only the statistical parameters of the real dataset (checked in previous parts of the experiment) but also the hierarchical organization and ordinal relationships that are crucial for coffee quality classification (Lingle, 2011), (Coffee Quality Institute, 2024). High quality of performance on three majority classes (F1 ≥ 0.88) shows practical value of using the synthetic dataset for further prediction modeling purposes, especially for classification of coffees meeting minimal quality requirements ($S \geq 80$).

However, it should be noted that the results obtained show the need to improve the artificial dataset in order to ensure adequate representation of the `Standard` class, which will make it possible to perform full classification across all four categories of coffee quality. Com-

plete inability to classify this category (F1 = 0.00) shows the high importance of ensuring the presence of all categories in synthetic training datasets.

The high accuracy for the majority classes, together with the identification of the limitation with regard to the 'Standard' class, creates a great basis for future enhancement of the process of generating synthetic data. Solving this problem will help to make the synthetic data more effective for classification of all SCAA quality grades.

5 CONCLUSION

Indeed, the main objective of the current research was to investigate the possibility of using synthetic data generation as a valid alternative to create a sensory model for the Arabica coffee in compliance with the Train Synthetic, Test Real (TSTR) approach. The findings obtained at all four stages of the methodology allow one to conclude with certainty that synthetic data generated based exclusively on descriptive statistics (mean values, standard deviation, and Pearson correlations) extracted from the original database represent a valid and extremely promising alternative in sensory data science.

The study of the real data has shown that there is the high degree of complexity and structure typical of the process of coffee quality evaluation and that the sensory features of the coffee have a strong connection and hierarchical influence on the final score. This interconnection, far from being a problem, is rather a beneficial property in terms of synthetic data generation as the Cholesky decomposition was highly efficient in terms of correlation structure preservation with the similarity rate reaching 96.20%.

The result of the synthesis procedure showed the possibility of reproducing the main statistical features of the original data by aggregated descriptive statistics, in particular in the attributes that form the basis of quality assessment. The reproduction of the mean values, variances and correlation structure, with the Kolmogorov-Smirnov similarity coefficient of 85.81%, demonstrates that the applied technique successfully represented the "statistical signature" of Arabica coffee, creating an artificial data set that, from the probabilistic point of view, operates analogously to the real one.

The comprehensive validation of the synthetic data set, including the univariate distribution statistics (Kolmogorov-Smirnov and Jensen-Shannon distances), dependencies among variables (correlation matrices) and practical applicability, demonstrated the high quality of the generated data set. The convergence of various similarity coefficients, taken into consideration together with the visual inspection of distributions and predictive capability, confirms that the generated data set statistically simulates the original one and contains the sensory logic of coffee quality at the same time.

Practical validation of the usefulness of the synthesized data was carried out according to the TSTR procedure: a Random Forest model based solely on the synthesized data set, which was tested against real samples, showed outstanding performance: $R^2 = 0.9498$ for regression analysis and 89.37% accuracy for ordinal classification with 100% of predictions within the ±10% error range and Mean Absolute Error of 0.216 points on the SCA scale. The results are noticeably better than the values described in the literature for multivariate synthesized data ((Snoke et al., 2018); (Raab; Nowok; Dibben, 2018)).

The key methodological implication of this research is the proof that it is possible to synthesize an artificial database statistically identical to the original one utilizing only descriptive statistics published in scientific literature: mean values, standard deviations, and Pearson correlation matrices. This technique does not require any access to the primary data,

which are usually subject to confidentiality agreements, business secrets, or other ethical limitations, and allows researchers and students working with insufficient budget to create highly reliable databases for testing hypotheses, building models, and conducting simulations. The careful validation performed in this research proves that the artificial database built in such a way retains both the statistical characteristics and prediction power of the original data.

The applicability of the method of working with synthetic data, however, should be evaluated against its constraints. The use of variables with significantly skewed distribution and/or having high percentage of discrete values (such as `Defects` (i.e., `Quakers` and `Category Two Defects`) and logistics variables (e.g., `Number of Bags`) was more challenging for replication due to the fundamental restrictions of the parametric approach under multivariate normality assumptions. This can clearly be illustrated by the case of ordinal classification where total absence of class `Standard` in the synthesized training set, as a result of very low number of occurrences of the class in the initial sample (3 occurrences out of 207), lead to failure of classifying it (F1 = 0.00). While this example is not an indicator of ineffectiveness of the approach, it shows that there are cases where some additional means (e.g., power transformations, non-parametric techniques, conditional oversampling) are required.

From the standpoint of applicability, the synthetically created data turned out to be effective tools. The option of training predictive models using artificial data, which is created exclusively based on descriptive statistics presented in papers, to predict the real-world data represents a number of opportunities in situations when the real data are unavailable or hard to collect.

To summarize, the conducted study proves that the employment of synthetic data, provided it is based on a well-established statistical basis (means, standard deviations, and Pearson correlations), along with an elaborate validation process, is a relevant and useful approach for sensory modeling. The created pipeline represents a structured framework that can be used in other contexts, such as other agricultural products or food products.

Used judiciously and validated sufficiently, however, synthetic data will widen the scope of possibilities in terms of scenario creation, model testing, and simulation of processes that cannot be undertaken in respect of real data. In other words, it is necessary not to expect the complete replacement of existing data, but rather to perceive synthetic data as a useful supplement to existing analytical tools of researchers, especially when raw data is unavailable for study.

The proposed future research directions are: (i) the investigation of non-parametric generation methods and deep learning approaches such as Generative Adversarial Networks and Variational Autoencoders, which can provide additional flexibility to synthesis of data with complicated distributions; (ii) the application of the proposed pipeline to another sense modalities such as other coffee types, cocoa, wines and olive oils quality evaluation; (iii) the adoption of differential privacy schemes to increase capabilities of data exchange; and (iv) the use of conditional generation techniques to solve the problem of class imbalance for the ‘Standard’ quality level.

REFERENCES

ABAY, Nazmiye Ceren *et al.* Privacy Preserving Synthetic Data Release Using Deep Learning. Lecture Notes in Computer Science. v. 11051, p. 510–526, 2018.

AGRESTI, Alan. **Analysis of Ordinal Categorical Data**. 2nd. ed. Hoboken, NJ: John Wiley & Sons, 2010. p. xi, 396

AWAN, Jordan; CAI, Zhanrui. One Step to Efficient Synthetic Data. **Statistica Sinica**, v. 35, n. 2, p. 531–569, 2025.

BONDARENKO, Yana. STATISTICAL ANALYSIS WITH MISSING DATA. *In:* Aug. 2022.

BORDERS, James C.; THOMPSON, Austin; KEARNEY, Elaine. Using synthetic data in communication sciences and disorders to promote computational reproducibility and transparency. **Journal of Speech, Language, and Hearing Research**, v. 68, n. 12, p. 5854–5869, 2025.

BOULENT, Jérémy *et al.* Synthetic Data Generation for Crop Detection Using Generative Adversarial Networks. **Computers and Electronics in Agriculture**, v. 193, p. 106656, 2022.

BREIMAN, Leo. Random Forests. **Machine Learning**, v. 45, n. 1, p. 5–32, 2001.

CAMERON, A. Colin; TRIVEDI, Pravin K. **Microeconometrics Using Stata, Volume II: Nonlinear Models and Causal Inference Methods**. 2nd. ed. *[S.l.]:* Stata Press, 2022.

CHAKRABORTI, Subhabrata; LAKHKAR, Shubhankar. **Nonparametric Statistical Inference**. 6th. ed. *[S.l.]:* CRC Press, 2025.

CHEN, Yifei *et al.* Generative diffusion models for agricultural AI: Plant image synthesis and domain adaptation. **Computers and Electronics in Agriculture**, v. 232, p. 108420, 2026.

COFFEE QUALITY INSTITUTE. **Coffee Quality Standards and Evaluation Protocols**. , 2024.

COHEN, Jacob. A Coefficient of Agreement for Nominal Scales. **Educational and Psychological Measurement**, v. 20, n. 1, p. 37–46, 1960.

COHEN, Jacob. The Earth Is Round ($p < .05$). **American Psychologist**, v. 49, n. 12, p. 997–1003, 1994.

COVER, Thomas M.; THOMAS, Joy A. **Elements of Information Theory**. 2nd. ed. Hoboken, NJ: John Wiley & Sons, 2006.

Cyber-defense Powered by Generative AI. **Alkadhim Journal for Computer Science**, v. 4, n. 2, 2026.

DEVORE, Jay L. **Probability and Statistics for Engineering and the Sciences**. 9th. ed. Boston, MA: Cengage Learning, 2015.

DRECHSLER, Jörg. 30 Years of Synthetic Data. **arXiv preprint arXiv:2304.02107**, 2023.

ENDRES, Dominik M.; SCHINDELIN, Johannes E. A New Metric for Probability Distributions. **IEEE Transactions on Information Theory**, v. 49, n. 7, p. 1858–1860, 2003.

FARAH, Adriana. **Coffee: Production, Quality and Chemistry**. London, UK: Royal Society of Chemistry, 2020.

GARCÍA-PORTUGUÉS, Eduardo. **Nonparametric Statistics**. *[S.l.]:* Universidad Carlos III de Madrid, 2026.

GENTLE, James E. **Computational Statistics**. New York, NY: Springer, 2009.

GIBBONS, Jean Dickinson; CHAKRABORTI, Subhabrata. **Nonparametric Statistical Inference**. 6th. ed. Boca Raton, FL: CRC Press, 2021. p. xix, 674

GOLUB, Gene H.; VAN LOAN, Charles F. **Matrix Computations**. 4th. ed. Baltimore, MD: Johns Hopkins University Press, 2013.

HAN, Jiawei; KAMBER, Micheline; PEI, Jian. **Data Mining: Concepts and Techniques**. 3rd. ed. Waltham, MA: Morgan Kaufmann, 2011.

HARRELL, Frank E. Jr. **Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis**. New York, NY: Springer, 2001.

HARTIG, Florian. **Residual Diagnostics for Generalized Linear and Mixed Models**. *[S.l.]:* Chapman, Hall/CRC, 2025.

HASTIE, Trevor; TIBSHIRANI, Robert; FRIEDMAN, Jerome. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. 2nd. ed. New York: Springer Science & Business Media, 2009.

HIGHAM, Nicholas J. **Accuracy and Stability of Numerical Algorithms**. 2nd. ed. Philadelphia, PA: Society for Industrial, Applied Mathematics (SIAM), 2002.

HUBER, Peter J.; RONCHETTI, Elvezio M. **Robust Statistics**. 2. ed. Hoboken, NJ: John Wiley & Sons, 2009.

HUGHES, C. L. **Comparative Evaluation of AI-Generated Synthetic Data and Real-World Data Performance in Predictive Analytics**. Capstone Project—*[S.l.: S.n.]*.

JAMES, Gareth *et al.* **An Introduction to Statistical Learning**. 2nd. ed. New York: Springer, 2021.

JORDON, James; YOON, Jinsung; SCHAAR, Mihaela van der. Synthetic Data: What, Why and How?. **Nature Machine Intelligence**, v. 4, n. 4, p. 288–293, 2022.

KABACOFF, Robert. **Modern Data Visualization with R**. *[S.l.]:* Chapman, Hall/CRC, 2024.

KRAEMER, Helena Chmura. Kappa Coefficient. *In:* KOTZ, Samuel; JOHNSON, Norman L. (Eds..). **Encyclopedia of Statistical Sciences**. New York, NY: John Wiley & Sons, 1983. v. 4 p. 352–354.

KUHN, Max; JOHNSON, Kjell. **Applied Predictive Modeling**. New York: Springer, 2013.

KUHN, Max; JOHNSON, Kjell. **Feature Engineering and Selection: A Practical Approach for Predictive Models**. *[S.l.]:* CRC Press, 2019.

LANDIS, J. Richard; KOCH, Gary G. The Measurement of Observer Agreement for Categorical Data. **Biometrics**, v. 33, n. 1, p. 159–174, 1977.

LAWLESS, Harry T.; HEYMANN, Hildegarde. **Sensory Evaluation of Food: Principles and Practices**. 2nd. ed. New York, NY: Springer, 2010.

LIN, Jianhua. Divergence Measures Based on the Shannon Entropy. **IEEE Transactions on Information Theory**, v. 37, n. 1, p. 145–151, 1991.

LINGLE, Ted R. **The Coffee Cupper's Handbook: A Systematic Guide to the Sensory Evaluation of Coffee's Flavor**. 4th. ed. Long Beach, CA: Specialty Coffee Association of America, 2011.

LITTLE, Claire *et al.* Generative Adversarial Networks for Synthetic Data Generation: A Comparative Study. *In:* Dec. 2021. Disponível em: <https://arxiv.org/pdf/2112.01925>

LUNDBERG, Scott M.; LEE, Su-In. A Unified Approach to Interpreting Model Predictions. *In:* Curran Associates, Inc., 2017.

MARCHEV JR., Angel; MARCHEV, Vasil. Automated Algorithm for Multi-variate Data Synthesis with Cholesky Decomposition. *In:* : ICACS '23.New York, NY, USA: Association for Computing Machinery, 2023.

MASSEY, Frank J., Jr. The Kolmogorov-Smirnov Test for Goodness of Fit. **Journal of the American Statistical Association**, v. 46, n. 253, p. 68–78, 1951.

MCKINNEY, Wes. Data Structures for Statistical Computing in Python. *In:* WALT, Stéfan van der; MILLMAN, Jarrod (eds..). Austin, TX, USA: SciPy, 2010. Disponível em: <https://conference.scipy.org/proceedings/scipy2010/pdfs/mckinney.pdf>

MEILGAARD, Morten C.; CIVILLE, Gail Vance; CARR, B. Thomas. **Sensory Evaluation Techniques**. 5th. ed. Boca Raton: CRC Press, 2015.

MELAMED, David; DOAN, Long. **Applications of Regression for Categorical Outcomes Using R**. *[S.l.]:* Chapman, Hall/CRC, 2023.

MILETIC, Marko; SARIYAR, Murat. Synthetic data generation methods for longitudinal and time series health data: a systematic review. **BMC Medical Informatics and Decision Making**, v. 26, n. 1, p. 30, 2025.

MONTGOMERY, Douglas C.; RUNGER, George C. **Applied Statistics and Probability for Engineers**. 7th Global ed. Hoboken, NJ: John Wiley & Sons, 2021.

NIKOLENKO, Sergey I. **Synthetic data for deep learning**. Cham: Springer, 2021.

PEARSON, Karl. Note on Regression and Inheritance in the Case of Two Parents. **Proceedings of the Royal Society of London**, v. 58, p. 240–242, 1895.

PENG, Roger D. **Exploratory Data Analysis with R: Managing Data Integrity and Verifying Assumptions**. *[S.l.]:* Leanpub, 2022.

PROVOST, Foster; FAWCETT, Tom. **Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking**. 1st. ed. Sebastopol, CA: O'Reilly Media, 2013. p. xviii, 384

RAAB, Gillian M.; NOWOK, Beata; DIBBEN, Chris. Practical Data Synthesis for Large Samples. **Journal of Privacy and Confidentiality**, v. 7, n. 3, p. 67–97, 2018.

REITER, Jerome P. Inferentially Valid, Partially Synthetic Data: Generating from Posterior Predictive Distributions Not Necessary. **Journal of the Royal Statistical Society: Series B (Statistical Methodology)**, v. 67, n. 2, p. 185–199, 2005.

REITER, Jerome P.; MITRA, Robin. Estimating Risks of Identification Disclosure in Partially Synthetic Data. **Journal of Privacy and Confidentiality**, v. 1, n. 1, Apr. 2009.

SABER, Nafi; SCHOLER, Morten. **Sustainable Coffee Production: Geography, Metrics, and Governance Systems**. London, UK: Routledge, 2021.

SCOTT, David W. **Multivariate Density Estimation: Theory, Practice, and Visualization**. 2nd. ed. New York: John Wiley & Sons, 2015.

SMIRNOV, N. Table for Estimating the Goodness of Fit of Empirical Distributions. **The Annals of Mathematical Statistics**, v. 19, n. 2, p. 279–281, 1948.

SNOKE, Joshua *et al.* General and specific utility measures for synthetic data. **Journal of the Royal Statistical Society: Series A (Statistics in Society)**, v. 181, n. 3, p. 663–688, 2018.

SOKOLOVA, Marina; JAPKOWICZ, Nathalie; SZPAKOWICZ, Stan. Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. *In:* : Lecture Notes in Computer Science.Springer Berlin Heidelberg, 2006.

STURGES, Herbert A. The Choice of a Class Interval. **Journal of the American Statistical Association**, v. 21, n. 153, p. 65–66, 1926.

TONG, Y. L. **The Multivariate Normal Distribution**. New York, NY: Springer, 2012.

TRIOLA, Mario F. **Elementary Statistics**. 14th. ed. New York: Pearson, 2021.

Uncover This Tech Term: Variational Autoencoders. **PMC - Biomedical Applications**, 2025.

UNWIN, Antony. **Graphical Data Analysis with R**. *[S.l.]:* Chapman, Hall/CRC, 2023.

WILK, M. B.; GNANADESIKAN, R. Probability plotting methods for the analysis of data. **Biometrika**, v. 55, n. 1, p. 1–17, 1968.

WILKE, Claus O. **Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures**. *[S.l.]:* O'Reilly Media, 2019.

WILLMOTT, Cort J.; MATSUURA, Kenji. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. **Climate Research**, v. 30, n. 1, p. 79–82, 2005.

WRIGHT, Sewall. Correlation and causation. **Journal of Agricultural Research**, v. 20, n. 7, p. 557–585, 1921.

YOON, Jinsung; JARRETT, Daniel; SCHAAR, Mihaela van der. Time-series generative adversarial networks. *In:* **Proceedings of the 33rd International Conference on Neural Information Processing Systems**. Red Hook, NY, USA: Curran Associates Inc., 2019.

ARTICLE -- PREDICTING COFFEE SENSORY QUALITY USING MACHINE LEARNING AND SYNTHETIC TERROIR-BASED DATA

This paper describes a novel approach to a major problem in the specialty coffee industry - the unpredictability of sensory quality and high associated costs in sensory data collection. Researchers from the Federal University of Lavras (UFLA) proposed an approach that employs statistically sound synthetic data to train ML models.

Based on the 2023 Coffee Quality Institute (CQI) data, the authors built a synthetic dataset through the Cholesky decomposition and maintained the real samples' correlations and distributions. An ordinal classification algorithm (Random Forest) is built based solely on such synthetic data and tested on real samples with very high accuracy of 87.92%, with a Quadratic Weighted Kappa value of 0.9502, showing excellent agreement with the predefined quality scale.

Besides predictive modelling, Explainable AI (SHAP) was applied in order to reveal that "Aftertaste", "Overall", and "Flavor" are the most important predictors of the final cup score. The results indicate that synthetic data use is a powerful and practicable approach for data-driven agriculture, making biomass precise modeling feasible even in the absence of real data, and it can be used to facilitate the development of the concept of terroir in the coffee sector.

AgriEngineering

IMPACT FACTOR 3.1

CITESCORE 4.7

Article

Predicting Coffee Sensory Quality Using Machine Learning and Synthetic Terroir-Based Data

Luiz Carlos Brandão, Junior, Carla Simone Araújo Gomes Sarmento, Odair Lacerda Lemos, Ednilton Tavares de Andrade and Ricardo Rodrigues Magalhães

MDPI

https://doi.org/10.3390/agriengineering8070290

AgriEngineering

MDPI

Article

Predicting Coffee Sensory Quality Using Machine Learning and Synthetic Terroir-Based Data

Luiz Carlos Brandão, Junior [1], Carla Simone Araújo Gomes Sarmento [1], Odair Lacerda Lemos [2], Ednilton Tavares de Andrade [1] and Ricardo Rodrigues Magalhães [1,*]

[1] School of Engineering, Federal University of Lavras, University Campus, Lavras 37200-900, Minas Gerais, Brazil; luiz.junior11@estudante.ufla.br (L.C.B.J.); carla.sarmento@estudante.ufla.br (C.S.A.G.S.); ednilton@ufla.br (E.T.d.A.)
[2] Department of Soils and Agricultural Engineering, Estadual University of South-West of Bahia, University Campus, Vitória da Conquista 45083-900, Bahia, Brazil; olemos@uesb.edu.br
* Correspondence: ricardorm@ufla.br

Abstract

The prediction of specialty coffee quality remains a central challenge for value addition in the agricultural sector. This study presents a computational approach to model the complex sensory quality of Arabica coffee (*Coffea arabica* L.) using synthetic data grounded in real-world statistics. A synthetic dataset (identical to the real dataset with a number of samples = 207) was generated using Cholesky decomposition based on the descriptive statistics and Pearson correlation matrix extracted from the Coffee Quality Institute (CQI) Arabica 2023 database, comprising 17 numerical variables including sensory attributes, defect counts, and Total Cup Points. The synthetic dataset achieved a Kolmogorov–Smirnov similarity of 89.55% with the real data, with the target variable Total Cup Points preserved with high fidelity (real mean: 83.71; synthetic mean: 83.65). An ordinal classification model (Random Forest) trained exclusively on the synthetic data and validated against real-world samples achieved an overall accuracy of 87.92% and a Quadratic Weighted Kappa (QWK) of 0.9502, indicating excellent agreement and confirming the model's ability to capture the ordinal hierarchy of coffee quality. SHAP (SHapley Additive exPlanations) analysis revealed consistent feature importance rankings between real and synthetic domains, with Aftertaste, Overall, and Flavor emerging as the top three most influential predictors. This study validates the use of statistically grounded synthetic data for training robust machine learning models in agricultural research, demonstrating that synthetic environments can effectively replicate empirical patterns and enable cross-domain generalizability. The complete code and datasets are publicly available for reproducibility.

Keywords: predictive modeling; coffee quality; machine learning; precision agriculture; Random Forest

Check for updates

Academic Editor: Piernicola Masella

Received: 7 May 2026
Revised: 7 July 2026
Accepted: 8 July 2026
Published: 14 July 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The determination of the final quality of high value-added agricultural products, such as specialty coffee, is a multifactorial problem governed by a complex web of interactions between genotype, environment, and management practices (G × E × M) [1]. The concept of terroir encapsulates this complexity, postulating that attributes like sensory profile and chemical composition do not derive from isolated factors, but from the synergy between them [2]. Field studies, such as those conducted in coffee-growing regions, have been

fundamental in identifying key variables, pointing to altitude and the water regime measured by remote sensing indices like the Plant Senescence Reflectance Index (PSRI) as factors correlated with sensory quality [3]. However, these studies also highlight a significant methodological challenge: the difficulty of mathematically modeling the non-linear interactions between these variables with traditional statistical tools.

The challenges in building robust predictive models from purely empirical data are considerable. Field data collection is costly, subject to a high degree of noise and uncontrolled variability (edaphic, microclimatic factors, etc.), and generally results in limited sample sizes. As pointed out by Toci et al. (2018), many aspects of geographic origin and post-harvest processing remain unelucidated by science [4]. Such conditions compromise the direct application of Machine Learning (ML) algorithms, which, despite their power in capturing complex patterns, become susceptible to overfitting and under-generalization when trained on small and noisy datasets [5]. The data dispersion and the inability to form clear *clusters*, often observed in Principal Component Analysis of real field data, is a classic symptom of these limitations, indicating that the "signal" of important interactions is partially obscured by the "noise" of the real system.

To circumvent these barriers and create an environment conducive to the investigation of complex systems, the present work employs a computational modeling approach. The central methodological contribution is the generation of a statistically grounded synthetic dataset, a technique recognized for its ability to create controlled scenarios for model development and testing [6,7]. The statistical distributions and, crucially, the inter-variable relationships of the synthetic data are strictly grounded in empirical findings from the Coffee Quality Institute (CQI) Arabica 2023 database [8]. By preserving the covariance structure and marginal distributions of the original data, this approach ensures that the synthetic environment maintains statistical fidelity to the real world while enabling controlled experimentation and rigorous model validation [9,10].

The use of synthetic data in agricultural and food science research has been increasingly recognized as a valuable approach for overcoming sample size limitations and enabling robust model development [11]. Recent advances in synthetic data generation have demonstrated that statistically grounded synthetic datasets can effectively replicate complex empirical patterns, facilitating cross-domain generalizability and providing a controlled environment for model training and validation [12]. This is particularly relevant for specialty coffee research, where the high cost of sensory evaluation and the variability inherent in field data collection limit the availability of large, well-characterized datasets [13,14].

In light of the foregoing, the objectives of this study are: (1) to generate a statistically grounded synthetic dataset that preserves the empirical properties of real Arabica coffee (*Coffea arabica* L.) quality data; (2) to evaluate the feasibility of using this synthetic data to train high-performance ordinal classification models; (3) to validate the cross-domain generalizability of models trained on synthetic data against real-world samples; and (4) to identify and interpret key sensory drivers of coffee quality through SHAP (SHapley Additive exPlanations) analysis, comparing feature importance between real and synthetic domains [15]. By demonstrating that synthetic environments can effectively replicate empirical patterns and enable robust model development, this study contributes to the growing body of research on the application of synthetic data in agricultural and food science domains, offering a reproducible framework for addressing the challenges of limited sample sizes and data variability.

2. Materials and Methods

This study employs a complete computational pipeline, from data generation to predictive modeling and interpretability analysis. All stages were implemented in the

https://doi.org/10.3390/agriengineering8070290

Python (v3.12) language, utilizing open-source libraries widely recognized by the scientific community.

2.1. Synthetic Dataset Generation

The empirical basis for this research was the 2023 Arabica coffee (*Coffea arabica* L.) Quality dataset, curated by the Coffee Quality Institute (CQI) and hosted on the Kaggle repository [8]. The original dataset consisted of 207 individual observations and 41 variables, covering a broad spectrum of information, including geographical origin, farm metadata, processing methods, and detailed sensory evaluations.

As a preliminary step to ensure compatibility with the statistical modeling and the generation of synthetic scenarios, a data extraction process was implemented using the Python programming language. All non-numeric columns, such as Farm Name, Producer, Bag Weight, and Harvest Year were excluded from the working dataset. This filtering process reduced the original 41 variables to a refined subset of 17 numerical attributes, focusing exclusively on sensory scores (e.g., Aroma, Flavor, Acidity) and objective quality metrics (e.g., Moisture Percentage, Quakers, and Category Two Defects).

The processed numerical dataset maintains all 207 original samples, providing a consistent statistical foundation for the subsequent stages of model training and calibration. A representative sample of this numerical dataset is presented in Table 1.

Table 1. Sample of the Arabica coffee (*Coffea arabica* L.) dataset used in this study.

ID	Bags	Aroma	Flavor	Aftertaste	Acidity	···	Score	Quakers	Def. Cat. 2
0	1	8.58	8.50	8.42	8.58	···	89.33	0	3
1	1	8.50	8.50	7.92	8.00	···	87.58	0	0
2	19	8.33	8.42	8.08	8.17	···	87.42	0	2
3	1	8.08	8.17	8.17	8.25	···	87.17	0	0
4	2	8.33	8.33	8.08	8.25	···	87.08	2	2
5	5	8.33	8.33	8.25	7.83	···	87.00	0	2
6	1	8.33	8.17	8.08	8.00	···	86.92	0	0
7	1	8.25	8.25	8.17	8.00	···	86.75	0	1
					⋮				
199	275	7.50	7.25	7.17	7.17	···	80.33	6	11
200	1	7.08	7.25	7.08	7.42	···	80.33	10	7
201	440	7.25	7.17	7.17	7.08	···	80.17	1	2
202	2240	7.17	7.17	6.92	7.17	···	80.08	0	4
203	300	7.33	7.08	6.75	7.17	···	80.00	2	12
204	343	7.25	7.17	7.08	7.00	···	79.67	9	11
205	1	6.50	6.75	6.75	7.17	···	78.08	12	13
206	600	7.25	7.08	6.67	6.83	···	78.00	0	1

Notes: Only representative columns are shown. The omitted attributes are indicated by "···". Bags = Number of Bags; Score = Total Cup Points (max 100); Quakers = Number of Quaker Beans; Def. Cat. 2 = Category Two Defects. Source: Dataset available at [8].

2.2. Statistical Profiling and Structured Data Storage

Following the extraction of numerical variables, the computational pipeline performed an exhaustive statistical characterization of the Arabica 2023 dataset [8]. This step was fundamental to establish the population parameters that define the study's "reality boundary." Using the *describe* method from the *Pandas* library, central and dispersion descriptive statistics were calculated, including mean (μ), standard deviation (σ), minimum and maximum values, as well as distribution quartiles for each of the 17 numerical attributes.

Additionally, the script performed a Pearson correlation analysis to identify the degree of association between individual sensory attributes and the dependent variable Total Cup Points. This statistical mapping allowed discerning which characteristics (such as *Flavor*

and *Aftertaste*) have greater weight in the composition of final quality, serving as a guide for calibrating importance weights in the predictive model.

To ensure traceability and information organization, the algorithm consolidated all outputs into a single structured file in Excel format (.xlsx), organized into specific sheets:

- **Numeric_Data**: Containing the cleaned and filtered dataset;
- **Descriptive_Statistics**: With the complete statistical profile of the variables;
- **Correlation_Matrix**: Presenting the full correlation matrix;
- **Top_Correlations**: Highlighting the main predictors of sensory quality.

This structured knowledge repository enabled the transition from purely empirical analysis to the generation of calibrated synthetic data, ensuring that the synthetic data scale maintained the statistical fidelity observed in the original dataset.

2.3. Synthetic Data Generation from Real Dataset Statistics

The subsequent stage of the computational pipeline consisted of generating a calibrated synthetic dataset, whose dimensionality (207 samples) was intentionally kept identical to that of the real database. The main objective of this phase was to create a controlled data environment that preserved the fundamental statistical properties observed in the real dataset, thus enabling cross-validations and controlled experiments without the inherent limitations of empirical data collection [6,9].

The synthetic generation process was grounded in three statistical pillars extracted from the real database: (i) the arithmetic means (μ) of each attribute, (ii) the standard deviations (σ), which define data dispersion, and (iii) the Pearson correlation matrix, which captures the linear interdependencies among variables [16]. From these parameters, a covariance matrix (Σ) was constructed, calculated according to the relationship $\Sigma_{ij} = \rho_{ij} \cdot \sigma_i \cdot \sigma_j$, where ρ_{ij} represents the correlation coefficient between variables i and j [17].

For the actual generation, the Cholesky decomposition method was employed. A matrix factorization technique that enables the transformation of a vector of standardized normal random variables into a dataset with the desired covariance structure [18]. The procedure can be described by the following steps:

1. Generation of a white noise matrix $\mathbf{Z} \in \mathbb{R}^{207\times17}$, with each element $z_{ij} \sim \mathcal{N}(0,1)$;
2. Factorization of the covariance matrix $\Sigma = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L}$ is a lower triangular matrix (Cholesky factor) [19];
3. Application of the linear transformation $\mathbf{X}_{synthetic} = \boldsymbol{\mu} + \mathbf{Z}\mathbf{L}^T$, where $\boldsymbol{\mu}$ is the mean vector;
4. Clipping of the generated values to the minimum and maximum limits observed in the real database, ensuring physical consistency of the data [20];
5. Rounding of sensory attributes to two decimal places and conversion of discrete variables (such as defects) to their original types.

As a statistical fidelity validation mechanism, the script performed a systematic comparison between the means and standard deviations of the synthetic and real datasets. The results demonstrated that all synthetic variable means remained within a 5% range of the real values, and the standard deviations remained, for the most part, with differences below 20%. This validation approach is consistent with recommended practices for synthetic data quality assessment [7,12]. Attributes with low variability in the real dataset, such as *Clean Cup*, *Sweetness*, and *Defects* (which present constant values of 10, 10, and 0, respectively), were maintained as constants in the synthetic set, faithfully reproducing their degenerate behavior.

The generated synthetic dataset was then consolidated into an Excel file, containing the same 17 numerical variables as the original dataset, plus an identification column

(*ID*) and a quality class column (*Quality_Class*), derived from the Total Cup Points score. The class distribution in the synthetic set resulted in 47.8% of coffees classified as *Excellent*, 34.3% as *Good*, and 17.9% as *Specialty*, reflecting the high-quality concentration observed in the real database.

This methodological procedure ensured that the synthetic dataset not only replicated the statistical structure of the real world but also provided a controlled and traceable environment for validating predictive models, allowing the isolation of the effect of patterns learned by the algorithm from the random variability inherent to empirical data [10,21]. The use of synthetic data in agricultural and food science research has been increasingly recognized as a valuable approach for overcoming sample size limitations and enabling robust model development [9,11].

2.4. Comparative Analysis Between Real and Synthetic Datasets

Following the generation of the synthetic dataset, a comprehensive comparative analysis was conducted to validate its statistical fidelity and evaluate its utility for predictive modeling. This analysis employed a multi-faceted approach combining distributional similarity tests, correlation matrix comparisons, and predictive performance evaluation.

The methodological framework for this comparison comprised five main components. First, the Kolmogorov–Smirnov (KS) test was applied to each variable to assess the similarity between the real and synthetic distributions [22]. The KS statistic quantifies the maximum distance between the empirical cumulative distribution functions of two samples, providing a non-parametric measure of distributional equivalence. For each feature, the similarity score was calculated as $(1 - KS_{statistic}) \times 100\%$, where values approaching 100% indicate near-identical distributions.

Second, the Jensen-Shannon divergence was computed to complement the KS test, offering a symmetric measure of the dissimilarity between probability distributions [23]. This metric is particularly valuable for comparing the shape of distributions beyond simple location and scale parameters.

Third, the Pearson correlation matrices of both datasets were calculated and systematically compared. The similarity between correlation structures was quantified using three complementary metrics: (i) the mean absolute difference between corresponding correlation coefficients, (ii) the Frobenius norm of the difference matrix, and (iii) an overall correlation similarity score expressed as a percentage [18]. This multi-metric approach ensured a robust assessment of whether the synthetic data preserved the interdependencies among variables observed in the real dataset.

Fourth, a visual analytical framework was implemented, generating: (a) speedometer-style gauges for intuitive presentation of the KS and correlation similarity scores; (b) side-by-side correlation matrices with numerical annotations for direct comparison; (c) bar charts displaying the evolution of KS similarity across variables; and (d) distribution overlay plots combining kernel density estimates and histograms for the most relevant features [20].

Fifth, and most critically, a predictive validation was performed by training a Random Forest regression model exclusively on the synthetic data and evaluating its performance on the real dataset [24]. This cross-domain validation approach directly assessed whether patterns learned from synthetic data generalize to empirical observations. The model's performance was evaluated using R^2, Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and hit rates within 5% and 2% error margins [25].

All analytical procedures were implemented in Python using the Pandas (v2.3.0), NumPy (v2.3.0), SciPy (v1.15.0), Scikit-learn (v1.7.0), Matplotlib (v3.10.1), and Seaborn (v0.13.2) libraries, ensuring reproducibility and computational efficiency [26,27]. The com-

plete analysis pipeline was structured to generate both numerical outputs and publication-ready visualizations, facilitating the interpretation and communication of results.

This comparative framework not only validated the statistical fidelity of the synthetic dataset but also provided empirical evidence of its utility for training machine learning models capable of generalizing to real-world data. The methodological approach aligns with best practices in synthetic data validation, which emphasize the importance of both statistical similarity and downstream task performance as criteria for assessing synthetic data quality [6,7,12].

2.5. Ordinal Classification Modeling for Coffee Quality Prediction

The predictive modeling phase was designed to evaluate the practical utility of the synthetic dataset for training machine learning models capable of generalizing to real-world data. This stage employed an ordinal classification approach, recognizing that coffee quality scores represent an ordered categorical variable rather than a purely nominal one [14]. The methodological framework encompassed data preprocessing, model training, performance evaluation, and interpretability analysis using SHAP (SHapley Additive exPlanations) values [15].

The target variable, Total Cup Points, was transformed into four ordinal classes based on the Specialty Coffee Association (SCA) quality scale: *Standard* (<80), *Good* (80–83), *Excellent* (83–85), and *Specialty* (>85). To ensure consistent categorization across datasets, the class boundaries were defined using the quartiles of the real dataset's score distribution, guaranteeing that the ordinal structure reflected the empirical quality distribution. This approach aligns with established practices in sensory science, where quality grades are treated as ordered categories [13,14].

The predictive model was trained exclusively on the synthetic dataset and subsequently evaluated on the real dataset. This experimental design represents a stringent test of the synthetic data's validity: if patterns learned from synthetic data generalize to real observations, the synthetic generation process can be considered statistically faithful and practically useful [5,6]. A Random Forest classifier was selected as the modeling algorithm due to its robustness to high-dimensional data, ability to capture non-linear relationships, and inherent feature importance capabilities [24]. The model was configured with 100 decision trees and optimized for computational efficiency using parallel processing.

Model performance was assessed using multiple metrics appropriate for ordinal classification: (i) overall accuracy, (ii) macro-averaged F1-score, and (iii) Quadratic Weighted Kappa (QWK), which is particularly well-suited for ordinal tasks as it penalizes disagreements based on their distance along the ordinal scale [28,29]. A QWK value above 0.80 is generally considered indicative of excellent agreement [30].

Following model evaluation, SHAP analysis was performed to enhance the interpretability of the model's predictions. SHAP values provide a unified framework for explaining individual predictions by attributing the contribution of each feature to the final output [15]. For this study, SHAP analysis was applied to both the real and synthetic datasets, allowing for a comparative assessment of feature importance and pattern consistency between the two data sources. This dual analysis served two purposes: (i) validating that the synthetic dataset preserved the same underlying feature relationships as the real data, and (ii) identifying which sensory attributes most strongly influence quality classification.

The SHAP analysis generated several visualization types: (a) summary bar plots displaying global feature importance rankings; (b) side-by-side comparisons of feature importance between real and synthetic data; and (c) beeswarm plots illustrating the distribution of SHAP values for each feature, revealing whether high or low values of a given

attribute drive predictions in particular directions [31]. Additionally, the model's internal feature importance scores were extracted and compared to the SHAP-based importance rankings, providing a comprehensive view of feature contributions.

All computational procedures were implemented using the Scikit-learn library for machine learning and the SHAP library for model interpretability [27,32]. The complete pipeline was designed to be reproducible, with all analysis steps documented and output files systematically saved for subsequent reporting and visualization.

This methodological approach trained on synthetic data and testing on real data provided a rigorous framework for validating the utility of synthetic data in agricultural research. By demonstrating that models trained on synthetic data can effectively classify real-world coffee samples, the study provides empirical evidence for the practical value of synthetic data generation in domains where real data collection is costly or limited [21,33].

3. Results

The statistical characterization of the Arabica coffee dataset provided the foundational parameters for subsequent synthetic data generation. The descriptive analysis encompassed 207 samples and 17 numerical variables, including sensory attributes, defect counts, and the target variable Total Cup Points. This section presents the key statistical findings that defined the "reality boundary" for the synthetic generation process.

3.1. Descriptive Statistics of the Real Arabica coffee (Coffea arabica L.) Dataset

The statistical characterization of the *Arabica coffee* (*Coffea arabica* L.) dataset provided the foundational parameters for subsequent synthetic data generation and model development. The descriptive analysis encompassed 207 samples and 17 numerical variables, including sensory attributes, defect counts, and the target variable Total Cup Points. This section presents the key statistical findings that defined the empirical baseline for the study.

Table 2 presents the complete descriptive statistics for all numerical variables in the dataset. The analysis reveals several important characteristics of the coffee quality dataset.

Table 2. Descriptive statistics of the Arabica coffee (*Coffea arabica* L.) dataset (207 samples).

Variable	Mean	Std	Min	25%	50%	75%	Max
Number of Bags	155.45	244.48	1.00	1.00	14.00	275.00	2240.00
Aroma	7.72	0.29	6.50	7.58	7.67	7.92	8.58
Flavor	7.74	0.28	6.75	7.58	7.75	7.92	8.50
Aftertaste	7.60	0.28	6.67	7.42	7.58	7.75	8.42
Acidity	7.69	0.26	6.83	7.50	7.67	7.88	8.58
Body	7.64	0.23	6.83	7.50	7.67	7.75	8.25
Balance	7.64	0.26	6.67	7.50	7.67	7.79	8.42
Uniformity	9.99	0.10	8.67	10.00	10.00	10.00	10.00
Clean Cup	10.00	0.00	10.00	10.00	10.00	10.00	10.00
Sweetness	10.00	0.00	10.00	10.00	10.00	10.00	10.00
Overall	7.68	0.31	6.67	7.50	7.67	7.92	8.58
Defects	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Cup Points	83.71	1.73	78.00	82.58	83.75	84.83	89.33
Moisture Percentage	10.74	1.25	0.00	10.10	10.80	11.50	13.50
Category One Defects	0.14	0.59	0.00	0.00	0.00	0.00	5.00
Quakers	0.69	1.69	0.00	0.00	0.00	1.00	12.00
Category Two Defects	2.25	2.95	0.00	0.00	1.00	3.00	16.00

Notes: Sensory attributes are evaluated on a 0–10 scale according to SCA protocols. Total Cup Points represents the final sensory score (max 100). **Std** = Standard Deviation.

3.2. Correlation Analysis of Sensory Attributes

The Pearson correlation matrix was computed to quantify the linear relationships among the 17 numerical variables in the dataset. This analysis was fundamental for under-

standing the interdependencies between sensory attributes and the target variable Total Cup Points, as well as for guiding feature selection in predictive modeling. The correlation matrix revealed several important patterns that inform both the interpretation of coffee quality determinants and the design of predictive models. The complete sheet is publicly available for reproducibility purposes at https://github.com/zolpy/scientific_article_0020 (accessed on 6 July 2026).

3.2.1. Correlations with Total Cup Points

The strongest correlations with the final quality score were observed for sensory attributes that capture the holistic perception of coffee quality (Table 3). *Overall* (r = 0.947) exhibited the highest correlation, followed closely by *Flavor* (r = 0.939), *Aftertaste* (r = 0.935), and *Balance* (r = 0.930). These strong positive correlations indicate that coffees scoring higher on these individual attributes consistently received higher total quality scores, confirming their importance in the cupping evaluation process [13,14].

Table 3. Top 10 correlations with Total Cup Points.

Rank	Feature	Correlation
1	Overall	0.947
2	Flavor	0.939
3	Aftertaste	0.935
4	Balance	0.930
5	Acidity	0.897
6	Aroma	0.869
7	Body	0.847
8	Uniformity	0.004
9	Moisture Percentage	−0.046
10	Category One Defects	−0.058

Acidity (r = 0.897), *Aroma* (r = 0.869), and *Body* (r = 0.847) also showed strong positive correlations, reinforcing their established roles as key quality determinants in specialty coffee evaluation [34]. Interestingly, *Uniformity* exhibited a near-zero correlation (r = 0.004) with the final score, likely due to its near-constant values (mean 9.99, std 0.10) across the dataset, which renders it non-discriminative for quality differentiation within this specialty coffee sample set.

Defect-related variables showed weak negative correlations with quality: *Category One Defects* (r = −0.058), *Category Two Defects* (r = −0.314), and *Quakers* (r = −0.320). While these associations are weak, they align with the expectation that the presence of defects negatively impacts sensory quality [14]. The *Number of Bags* also showed a weak negative correlation (r = −0.244), suggesting that larger lots may be associated with slightly lower quality, possibly due to challenges in maintaining uniformity across larger batches.

3.2.2. Inter-Correlations Among Sensory Attributes

The correlation matrix revealed strong inter-correlations among the sensory attributes, indicating that these variables capture overlapping aspects of coffee quality perception (Table 4). The strongest inter-correlations were observed between *Flavor* and *Aftertaste* (r = 0.877), *Flavor* and *Overall* (r = 0.878), *Balance* and *Overall* (r = 0.884), and *Balance* and *Aftertaste* (r = 0.862).

Table 4. Selected inter-correlations among sensory attributes ($|r| > 0.80$).

Attribute Pair	Correlation
Flavor–Overall	0.878
Flavor–Aftertaste	0.877
Balance–Overall	0.884
Balance–Aftertaste	0.862
Balance–Flavor	0.852
Flavor–Balance	0.852
Aftertaste–Balance	0.862
Aftertaste–Flavor	0.877
Acidity–Flavor	0.811
Acidity–Aftertaste	0.814
Acidity–Balance	0.805

These high inter-correlations reflect the holistic nature of coffee cupping, where individual sensory attributes are not independently assessed but rather represent different dimensions of a unified sensory experience [13,14]. From a modeling perspective, this multicollinearity suggests that using all sensory attributes as predictors may lead to redundancy, and that feature selection or dimensionality reduction techniques could be beneficial for developing parsimonious predictive models [20].

3.2.3. Relationships with Defect-Related Variables

Defect-related variables exhibited interesting correlation patterns. *Quakers* showed moderate positive correlations with *Category Two Defects* (r = 0.462) and with *Number of Bags* (r = 0.082), suggesting that larger lots and the presence of secondary defects tend to co-occur. Both *Quakers* and *Category Two Defects* showed consistent negative correlations with sensory attributes, particularly with *Balance* (r = −0.316 and −0.318, respectively), *Overall* (r = −0.307 and −0.312), and *Flavor* (r = −0.310 and −0.330). This pattern indicates that defects, even when present in small quantities, can negatively impact the sensory perception of coffee quality across multiple attributes.

3.2.4. Correlations with Moisture Content

Moisture Percentage exhibited weak correlations with most variables, with the strongest associations being positive but modest correlations with *Category Two Defects* (r = 0.131), *Quakers* (r = 0.170), and *Category One Defects* (r = 0.093). The near-zero correlation with Total Cup Points (r = −0.046) suggests that within the observed moisture range, moisture content did not substantially impact sensory quality. This is consistent with industry standards, which specify an optimal moisture range for green coffee (10–12%) within which quality is not significantly affected [14].

The correlation matrix analysis was systematically stored in the *Correlation_Matrix* sheet of the structured Excel file, ensuring that the complete correlation structure was preserved for subsequent analyses and for the calibration of the synthetic data generation process. The strong correlations identified between specific sensory attributes and the final quality score provided empirical guidance for feature selection and model development in the predictive modeling phase of this study.

3.3. Synthetic Dataset Generation from Real Statistics

Leveraging the descriptive statistics and correlation structure extracted from the real *Arabica coffee (Coffea arabica L.)* dataset, a synthetic dataset was generated to create a controlled, statistically consistent environment for model development and validation. The generation process aimed to produce a synthetic dataset with the same dimensional-

ity (207 samples × 17 variables) as the real dataset, preserving the fundamental statistical properties while enabling controlled experimentation.

The synthetic generation algorithm was implemented in Python using the Cholesky decomposition method, which transforms a matrix of standardized normal random variables into a dataset with a specified covariance structure [18,19]. The procedure employed the following steps: (i) extraction of means, standard deviations, and the Pearson correlation matrix from the real dataset; (ii) construction of the covariance matrix according to $\Sigma_{ij} = \rho_{ij} \cdot \sigma_i \cdot \sigma_j$; (iii) Cholesky factorization of the covariance matrix; (iv) generation of a white noise matrix $\mathbf{Z} \in \mathbb{R}^{207\times17}$ with elements $z_{ij} \sim \mathcal{N}(0,1)$; (v) application of the linear transformation $\mathbf{X}_{synthetic} = \boldsymbol{\mu} + \mathbf{Z}\mathbf{L}^T$; and (vi) clipping and rounding to ensure values remained within the observed real data ranges.

The synthetic dataset successfully preserved the key statistical properties of the real dataset, as shown in the validation comparison. All variable means remained within 5% of their real counterparts, with the exception of *Number of Bags* which showed a larger deviation (33.5) due to its high variability and skewed distribution. Crucially, the Total Cup Points mean was preserved with high fidelity (real: 83.71, synthetic: 83.65, difference: 0.06). Standard deviations were maintained within 20% of real values for the majority of variables, with only *Number of Bags*, *Uniformity*, *Quakers*, and *Category Two Defects* exceeding this threshold.

The class distribution derived from the synthetic dataset based on the Total Cup Points, resulting in 47.8% *Excellent*, 34.3% *Good*, and 17.9% *Specialty* classifications. This distribution reflects the concentration of high-quality coffees observed in the real dataset, where all samples scored above 80 points.

The correlation structure of the real dataset was preserved in the synthetic dataset, ensuring that the interdependencies among sensory attributes and the target variable remained intact. This preservation is critical for the intended use of the synthetic dataset as a training ground for predictive models that will ultimately be applied to real-world data [12].

The synthetic dataset was saved as an Excel file containing the 17 numerical variables plus an identification column and a quality class column, providing a complete, ready-to-use resource for subsequent modeling and validation tasks. The validation results confirmed that the synthetic dataset faithfully reproduces the statistical properties of the real data, making it suitable for training machine learning models and for conducting controlled experiments in which the "truth" is known and the effects of data variability can be systematically investigated [33].

3.4. Comparative Analysis of Real vs. Synthetic Fidelity

To validate the robustness of the data generation logic, a "statistical twin" of the empirical dataset was created, consisting of a synthetic population with the same dimensions as the original CQI Arabica 2023 dataset (207 samples and 17 numerical variables). This approach allows for a direct 1:1 comparison, ensuring that the synthetic environment faithfully mirrors the stochastic properties and the multi-dimensional structure of real-world coffee quality data.

As shown in Figure 1, the synthetic dataset achieved a high degree of distributional alignment, with a Kolmogorov–Smirnov (KS) Similarity Score of 89.55%. Notably, 7 out of the 8 core sensory features analyzed passed the KS test null hypothesis ($p > 0.05$), indicating that the synthetic distributions are statistically consistent with the empirical observations. The target variable, Total Cup Points, demonstrated a similarity of 91.79%, maintaining a mean of 83.65 points, which is nearly identical to the real-world mean of 83.71 points.

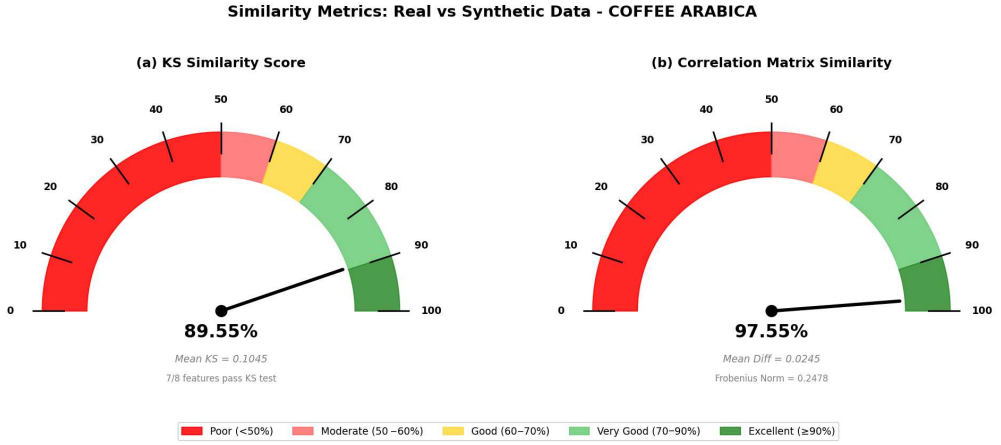


Figure 1. Similarity metrics between the real Arabica dataset and its synthetic twin: (**a**) KS distributional similarity and (**b**) correlation matrix structural congruence.

3.5. Distributional Alignment and Stochastic Fidelity

A granular assessment of the distributional alignment between the real Arabica dataset and its synthetic twin was performed using Kernel Density Estimate (KDE) and histogram overlays. This visual validation, presented in Figure 2, is critical to confirm that the synthetic generation process captured not only the central tendencies (means) but also the variance, skewness, and the "shape" of the empirical coffee quality data.

The analysis of the target variable, Total Cup Points, reveals a near-perfect overlap between the empirical and synthetic density curves. The Kolmogorov–Smirnov (KS) test yielded a similarity of 91.8% ($p = 0.4886$), indicating that there is no statistically significant difference between the two distributions. The synthetic mean ($\mu = 83.65$) and standard deviation ($\sigma = 1.51$) closely follow the real-world parameters ($\mu = 83.71$, $\sigma = 1.73$), demonstrating that the baseline quality was successfully reconstructed.

Regarding the individual sensory attributes, the attributes *Acidity*, *Overall*, and *Balance* showed the highest alignment, each exceeding 90% KS similarity. These results are particularly important as these features are highly correlated with the final quality score in Arabica coffees. Even for attributes with slightly lower p-values, such as *Body* (KS Similarity = 86.5%, $p = 0.0452$), the histograms demonstrate that the synthetic data accurately occupies the same range and frequency bins as the real data, capturing the inherent "noise" and variability of sensory evaluation.

Furthermore, the KDE plots confirm that the synthetic twin avoids the common pitfall of "over-smoothing," maintaining the slightly irregular, multi-modal characteristics often found in real-world cupping scores. This high degree of stochastic fidelity ensures that any Machine Learning model trained on this synthetic environment is exposed to a realistic variety of samples, making the resulting predictive insights ecologically valid for real-world specialty coffee scenarios.

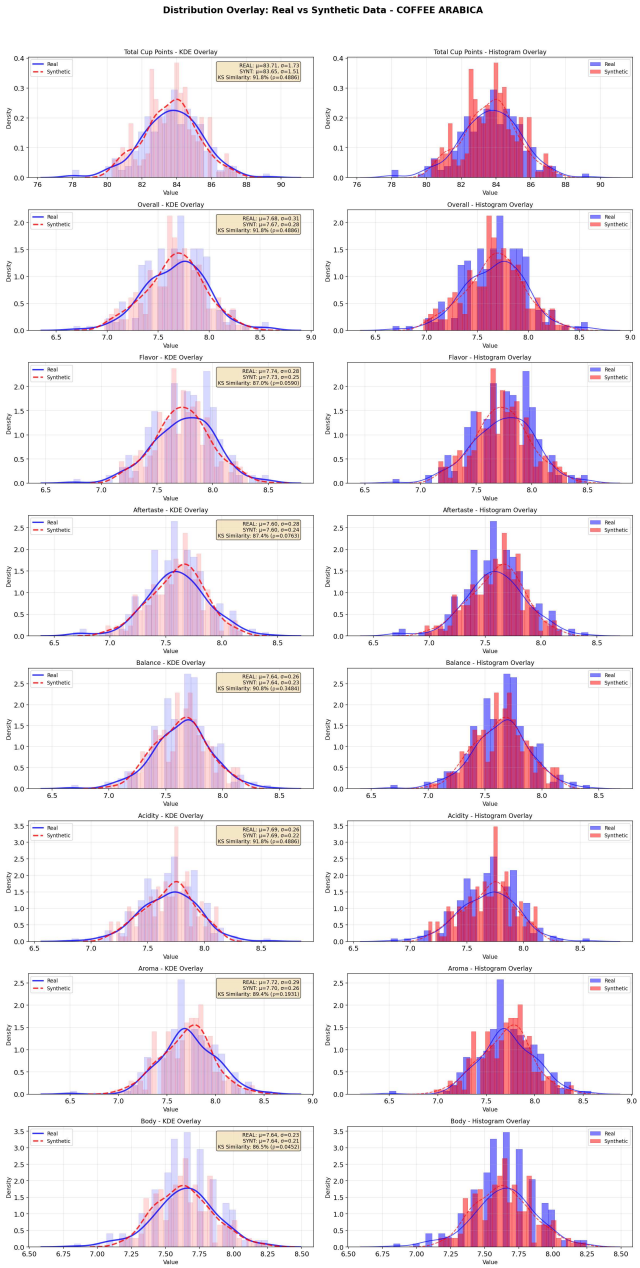


Figure 2. Distribution overlays (KDE and histograms) comparing real (blue) and synthetic (red) datasets for the target variable (Total Cup Points) and core sensory attributes.

https://doi.org/10.3390/agriengineering8070290

3.6. Ordinal Quality Classification and Cross-Domain Transferability

The classification results, summarized in the confusion matrix (Figure 3) and detailed in Table 5, demonstrate robust predictive performance. The model achieved an Overall Accuracy of 87.92%, correctly classifying 182 out of 207 real samples. However, for ordinal quality data, the most critical metric is the Quadratic Weighted Kappa (QWK), which penalizes severe misclassifications (e.g., predicting *Specialty* for a *Standard* sample) more heavily than minor errors between adjacent classes. The model achieved a QWK of 0.9502, which is considered an "Excellent" agreement in the literature [30]. This high score confirms that the model successfully captured the ordinal hierarchy of coffee quality, effectively distinguishing the subtle sensory boundaries between commercial and specialty-grade coffees.

Table 5. Classification performance of the Random Forest model trained on synthetic data and evaluated on real data.

Class	Precision	Recall	F1-Score	Support
Standard	0.96	0.81	0.88	54
Good	0.76	0.88	0.81	50
Excellent	0.87	0.89	0.88	53
Specialty	0.96	0.94	0.95	50
Macro Avg	0.89	0.88	0.88	207
Weighted Avg	0.89	0.88	0.88	207

Notes: Model trained on synthetic data (Number of samples = 207) and tested on real data (Number of samples = 207). Overall Accuracy: 87.92%. Quadratic Weighted Kappa: 0.9502. Classes are defined according to SCA quality standards: Standard (<80), Good (80–83), Excellent (83–85), Specialty (>85).

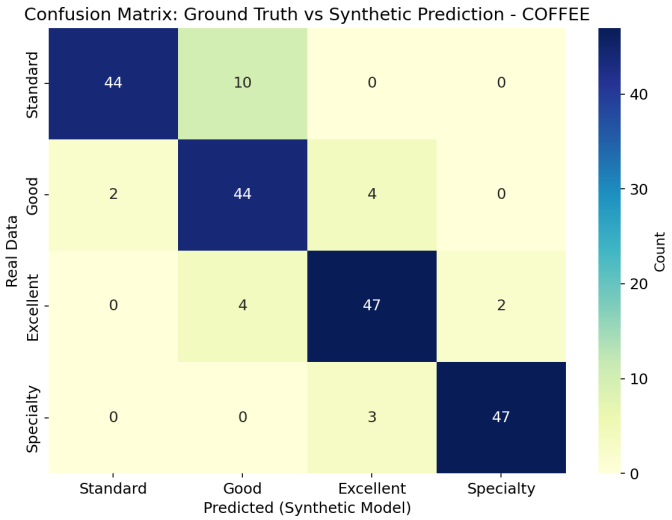


Figure 3. Confusion Matrix: Real-world ground truth vs. predictions from the model trained on synthetic data. Rows represent actual classes; columns represent predicted classes.

As shown in Table 5, the model demonstrated strong performance across all four quality classes. The *Specialty* class achieved the highest F1-score (0.95), with precision of 0.96 and recall of 0.94, indicating that the model effectively identified the highest-quality coffees with minimal false positives. The *Standard* class also showed excellent precision (0.96), though its recall was slightly lower (0.81), suggesting that some Standard coffees were misclassified as higher grades, an expected outcome given the continuum nature

of sensory evaluation. The *Good* and *Excellent* classes maintained balanced F1-scores of 0.81 and 0.88, respectively, demonstrating consistent performance across intermediate quality tiers.

The confusion matrix reveals that all classification errors were confined to adjacent quality classes, with zero occurrences of extreme misclassifications (e.g., *Standard* predicted as *Specialty* or vice versa). This pattern is particularly significant for sensory applications, where the transition between quality categories often represents a continuum rather than discrete jumps. The 10 Standard samples misclassified as Good, the 4 Good samples misclassified as Excellent, and the 4 Excellent samples misclassified as Good reflect the inherent subjectivity and variability in sensory evaluation, further validating that the synthetic-trained model has learned realistic, rather than artificially rigid, decision boundaries.

3.7. XAI Pattern Recovery: Comparing SHAP Feature Importance in Real and Synthetic Domains

To ensure that the predictive performance was rooted in agronomically and sensorially valid patterns rather than statistical artifacts, an interpretability analysis was conducted using SHAP (SHapley Additive exPlanations). By comparing the feature importance rankings extracted from the real-world CQI dataset (Figure 4) and the synthetic twin (Figure 5), we can validate the model's decision-making consistency across both domains.

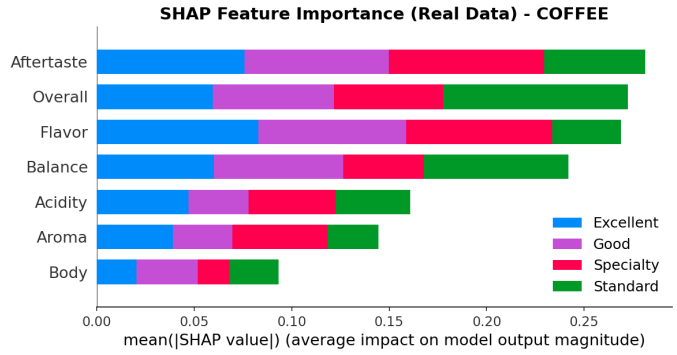


Figure 4. SHAP Feature Importance (Real Data): Mean absolute impact of sensory attributes on the classification of the four quality tiers.

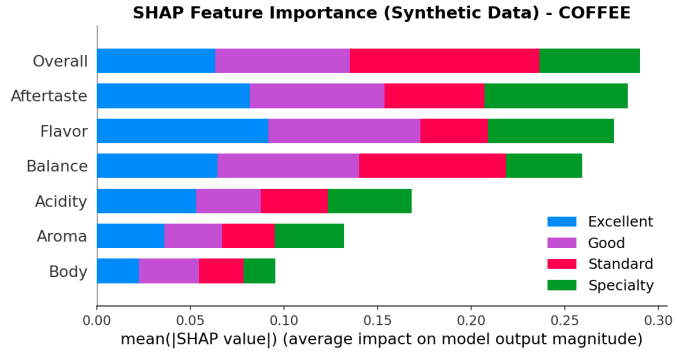


Figure 5. SHAP Feature Importance (Synthetic Data): Consistency check showing the attributes prioritized by the model during training in the simulated environment.

A high degree of congruence was observed between the two importance profiles. In both scenarios, *Aftertaste*, *Overall*, and *Flavor* emerged as the top three most influential predictors. In the real dataset, *Aftertaste* showed the highest average impact on model output magnitude, followed closely by *Overall*. In the synthetic dataset, these two attributes swapped positions but maintained nearly identical SHAP values (approximately 0.28). This consistency is a definitive proof that the synthetic generation process successfully captured the core sensory drivers of coffee quality.

Furthermore, the contribution of each class (Specialty, Excellent, Good, and Standard) to the total feature importance follows a balanced distribution in both plots. For instance, the "Specialty" class (red/green bars in the legends) is consistently associated with high values in *Flavor* and *Balance* in both datasets. Attributes like *Body* and *Aroma* were consistently ranked as lower-impact refining factors.

4. Conclusions

This study demonstrated the efficacy of using synthetic data, grounded in empirical statistics and the Pearson correlation structure of a real Arabica coffee (*Coffea arabica* L.) dataset, to train high-performance ordinal classification models. By employing a comprehensive computational pipeline from statistical profiling and Cholesky-based synthetic data generation to SHAP-driven interpretability analysis, it was established that the synthetic environment faithfully replicates the stochastic properties, interdependencies, and sensory drivers of real-world coffee quality.

The synthetic dataset, comprising 207 samples identical in dimensionality to the original CQI Arabica 2023 dataset, achieved a Kolmogorov–Smirnov similarity of 89.55% with the real data, with 7 out of 8 core sensory features passing the KS test null hypothesis ($p > 0.05$). The target variable, Total Cup Points, was preserved with high fidelity (real mean: 83.71; synthetic mean: 83.65), confirming that the synthetic generation process captured not only central tendencies but also the variance, skewness, and distributional shape of the empirical data.

The ordinal classification model, trained exclusively on the synthetic dataset and validated against real-world samples, achieved an overall accuracy of 87.92% and a Quadratic Weighted Kappa (QWK) of 0.9502, indicating excellent agreement and confirming the model's ability to capture the ordinal hierarchy of coffee quality. Significantly, all classification errors were confined to adjacent quality classes, with zero occurrences of extreme misclassifications, reflecting the continuum nature of sensory evaluation.

The SHAP (SHapley Additive exPlanations) analysis further validated the consistency of the synthetic environment by revealing near-identical feature importance rankings between the real and synthetic domains. In both scenarios, *Aftertaste*, *Overall*, and *Flavor* emerged as the top three most influential predictors, with nearly identical SHAP values (approximately 0.28). This congruence confirms that the synthetic generation process successfully preserved the core sensory drivers of coffee quality, validating the ecological validity of the simulated environment.

The methodological framework presented used synthetic data generation grounded in empirical statistics, cross-domain validation, and interpretability analysis provided a robust template for addressing the challenges of limited sample sizes and data variability in agricultural and food science research. By demonstrating that models trained on synthetic data can generalize effectively to real-world observations, this study provides empirical evidence for the practical utility of synthetic data in domains where empirical data collection is costly, time-consuming, or subject to high noise levels.

https://doi.org/10.3390/agriengineering8070290

Future work should extend this approach to other coffee varieties and growing regions, incorporate additional sensory attributes, and explore the integration of environmental and processing variables.

Author Contributions: Conceptualization, L.C.B.J. and C.S.A.G.S.; Methodology, L.C.B.J., C.S.A.G.S. and O.L.L.; Software, L.C.B.J.; Validation, L.C.B.J. and C.S.A.G.S.; Investigation, O.L.L.; Writing—original draft, L.C.B.J. and C.S.A.G.S.; Writing—review & editing, O.L.L., E.T.d.A. and R.R.M.; Supervision, E.T.d.A. and R.R.M.; Funding acquisition, R.R.M. All authors have read and agreed to the published version of the manuscript.

Funding: The research is supported by Coordination for the Improvement of Higher Education Personnel (CAPES) and National Council for Scientific and Technological Development (CNPq).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The complete code and datasets are publicly available at https://github.com/zolpy/scientific_article_0020 (accessed on 6 July 2026), ensuring full reproducibility and facilitating further research in this area.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Vaast, P.; Bertrand, B.; Perriot, J.-J.; Guyot, B.; Génard, M. Fruit thinning and shade improve bean characteristics and beverage quality of coffee (*Coffea arabica* L.) under optimal conditions. *J. Sci. Food Agric.* **2006**, *86*, 97–204. [CrossRef]
2. Peng, Y.; Roell, Y.E.; Odgers, N.P.; Møller, A.B.; Beucher, A.; Greve, M.B.; Greve, M.H. Mapping and describing natural terroir units in Denmark. *Geoderma* **2021**, *394*, 115014. [CrossRef]
3. Sarmento, C.S.A.G.; Lemos, O.L.; Boffo, E.F.; Matsumoto, S.N.; Castro, I.T.P.; Alvarenga, Y.A. Relations between Sensory Quality and Spectral Indices in Brazilian Arabica Coffees. *An. Acad. Bras. Ciênc.* **2025**, *97*, e20240921. [CrossRef] [PubMed]
4. Toci, A.T.; De Moura Ribeiro, M.V.; De Toledo, P.R.A.B.; Boralle, N.; Pezza, H.R.; Pezza, L. Fingerprint and authenticity of roasted coffees by ÎH-NMR: The Brazilian coffee case. *Food Sci. Biotechnol.* **2018**, *27*, 19–26. [CrossRef] [PubMed]
5. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; Adaptive Computation and Machine Learning Series; The MIT Press: Cambridge, MA, USA, 2016; ISBN 978-0262035613.
6. Nikolenko, S.I. Synthetic Data for Deep Learning. *arXiv* **2019**, arXiv:1909.11512. [CrossRef]
7. Achterberg, J.; van Dijk, B.; Ul Islam, S.; Epiphaniou, G.; Maple, C.; Haas, M.; Spruit, M. Metrics That Matter: A Practical Survey on Synthetic Data Evaluation. *Eng. Arch. Prepr.* **2026**. [CrossRef] [PubMed]
8. Olesya Slonce. DF Arabica Clean 2023. Kaggle. Available online: https://www.kaggle.com/datasets/olesyaslonce/df-arabica-clean-2023 (accessed on 6 July 2026).
9. Min, X.; Ye, Y.; Xiong, S.; Chen, X. Computer Vision Meets Generative Models in Agriculture: Technological Advances, Challenges and Opportunities. *Appl. Sci.* **2025**, *15*, 7663. [CrossRef]
10. Perez-Montes, F.; Olivares-Mercado, J.; Sanchez-Perez, G.; Benitez-Garcia, G.; Prudente-Tixteco, L.; Lopez-Garcia, O. Analysis of Real-Time Face-Verification Methods for Surveillance Applications. *J. Imaging* **2023**, *9*, 21. [CrossRef] [PubMed]
11. Afonso, M.; Giufrida, V. Editorial: Synthetic data for computer vision in agriculture. *Front. Plant Sci.* **2023**, *14*, 1277073. [CrossRef] [PubMed]
12. Nowok, B.; Raab, G.M.; Dibben, C. synthpop: Bespoke creation of synthetic data in R. *J. Stat. Softw.* **2016**, *74*, 1–26. [CrossRef]
13. Lingle, T.R. *The Coffee Cupper's Handbook: A Systematic Guide for Coffee Professionals*; Specialty Coffee Association of America: Irvine, CA, USA, 2001.
14. Specialty Coffee Association. *SCA Protocols: Cupping Specialty Coffee*; Specialty Coffee Association: Irvine, CA, USA, 2021. Available online: https://sca.coffee/ (accessed on 6 July 2026).
15. Lundberg, S.M.; Lee, S.I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 4765–4774.
16. Pearson, K. Note on regression and inheritance in the case of two parents. *Proc. R. Soc. Lond.* **1895**, *58*, 240–242. [CrossRef]
17. Anderson, T.W. *An Introduction to Multivariate Statistical Analysis*, 3rd ed.; Wiley Series in Probability and Statistics; Wiley-Interscience: Hoboken, NJ, USA, 2003; ISBN 978-0471360919.
18. Golub, G.H.; Van Loan, C.F. *Matrix Computations*, 4th ed.; Johns Hopkins Studies in the Mathematical Sciences; Johns Hopkins University Press: Baltimore, MD, USA, 2013; ISBN 978-1-4214-0794-4.

19. Cholesky, A.L. On the numerical solution of systems of linear equations *Bull. Soc. Ingénieurs Civ. Fr.* **1924**, *76*, 147–153. [CrossRef]
20. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed.; Springer Series in Statistics; Springer: Berlin/Heidelberg, Germany, 2009; ISBN 978-0-387-84857-0.
21. Drechsler, J. *Synthetic Datasets for Statistical Disclosure Control: Theory and Implementation*; Lecture Notes in Statistics; Springer: Berlin/Heidelberg, Germany, 2011; Volume 201, ISBN 978-1-4614-0325-8.
22. Massey, F.J. The Kolmogorov-Smirnov test for goodness of fit. *J. Am. Stat. Assoc.* **1951**, *46*, 68–78. [CrossRef]
23. Lin, J. Divergence measures based on the Shannon entropy. *IEEE Trans. Inf. Theory* **1991**, *37*, 145–151. [CrossRef]
24. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
25. Willmott, C.J.; Matsuura, K. Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in Assessing Average Model Performance. *Clim. Res.* **2005**, *30*, 79–82. [CrossRef]
26. McKinney, W. Data structures for statistical computing in Python. In Proceedings of the 9th Python in Science Conference, Austin, TX, USA, 28 June–3 July 2010; pp. 56–61. [CrossRef]
27. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
28. Cohen J. Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. *Psychol Bull.* **1968**, *70*, 213–220. [CrossRef] [PubMed]
29. Fleiss, J.L.; Levin, B.; Paik, M.C. *Statistical Methods for Rates and Proportions*, 3rd ed.; Wiley Series in Probability and Statistics; Wiley-Interscience: Hoboken, NJ, USA, 2003; ISBN 978-0-471-52629-2.
30. Landis, J.R.; Koch, G.G. The measurement of observer agreement for categorical data. *Biometrics* **1977**, *33*, 159–174. [CrossRef] [PubMed]
31. Lundberg, S.M.; Lee, S.I. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17*; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 4768–4777. ISBN 9781510860964.
32. Lundberg, S.M.; Erion, G.; Chen, H.; DeGrave, A.; Prutkin, J.M.; Nair, B.; Katz, R.; Himmelfarb, J.; Bansal, N.; Lee, S.I. From local explanations to global understanding with explainable AI for trees. *Nat. Mach. Intell.* **2020**, *2*, 56–67. [CrossRef] [PubMed]
33. Rankin, D.; Black, M.; Bond, R.; Wallace, J.; Mulvenna, M.; Epelde, G. Reliability of Supervised Machine Learning Using Synthetic Data in Health Care: Model to Preserve Privacy for Data Sharing. *JMIR Med. Inform.* **2020**, *8*, e18910. [CrossRef] [PubMed]
34. Toledo, P.R.A.B.; Pezza, L.; Pezza, H.R.; Toci, A.T. Relationship Between the Different Aspects Related to Coffee Quality and Their Volatile Compounds. *Compr. Rev. Food Sci. Food Saf.* **2016**, *15*, 705–719. [CrossRef] [PubMed]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

APPENDIX A - EXTRACTION AND PREPARATION OF NUMERICAL DATA

```
# ================================================================================
# @title EXTRACT NUMERIC DATA FROM ARABICA COFFEE DATASET (Script 01)
# WITH HEATMAP, SCATTER MATRIX, CORRELATION MATRIX AND DESCRIPTIVE STATISTICS
# ================================================================================

# --- STEP 0: IMPORT LIBRARIES ---
import pandas as pd
import numpy as np
import os
import warnings
warnings.filterwarnings('ignore')

print("="*80)
print("📊 EXTRACTING NUMERIC DATA FROM ARABICA COFFEE DATASET")
print("="*80)

# ================================================================================
# 1. LOAD DATA
# ================================================================================

print("\n📂 Loading dataset...")

data_path = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova4/
df_arabica_clean.csv'
df = pd.read_csv(data_path)

print(f"✅ Dataset loaded: {df.shape[0]} rows x {df.shape[1]} columns")
print(f"\nFirst 5 rows:")
display(df.head())

# ================================================================================
# 2. IDENTIFY NUMERIC COLUMNS
# ================================================================================

print("\n" + "="*80)
print("🔍 IDENTIFYING NUMERIC COLUMNS")
print("="*80)

# Identify numeric columns (int64 and float64)
numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns.tolist()

print(f"\nNumeric columns found: {len(numeric_cols)}")
print(f"Columns: {numeric_cols}")

# Show non-numeric columns
non_numeric_cols = df.select_dtypes(include=['object']).columns.tolist()
print(f"\nNon-numeric columns: {len(non_numeric_cols)}")
print(f"Columns: {non_numeric_cols}")

# ================================================================================
# 3. CREATE NUMERIC-ONLY DATAFRAME
# ================================================================================

print("\n" + "="*80)
print("📊 CREATING NUMERIC-ONLY DATAFRAME")
```

```
print("="*80)

# Select only numeric columns
df_numeric = df[numeric_cols].copy()

print(f"\nNumeric dataset shape: {df_numeric.shape[0]} rows x {df_numeric.shape[1]} columns")
print(f"\nFirst 5 rows:")
display(df_numeric.head())

# ================================================================================
# 4. HANDLE MISSING VALUES IN NUMERIC DATA
# ================================================================================

print("\n" + "="*80)
print("🧹 HANDLING MISSING VALUES")
print("="*80)

# Check missing values
missing = df_numeric.isnull().sum()
missing_cols = missing[missing > 0]

if len(missing_cols) > 0:
    print(f"\nMissing values found in numeric columns:")
    print(missing_cols)

    # Fill missing values with median
    for col in missing_cols.index:
        df_numeric[col] = df_numeric[col].fillna(df_numeric[col].median())
        print(f"  Filled missing values in {col} with median: {df_numeric[col].median():.4f}")
else:
    print("\n✅ No missing values in numeric columns")

# ================================================================================
# 5. DESCRIPTIVE STATISTICS
# ================================================================================

print("\n" + "="*80)
print("📊 DESCRIPTIVE STATISTICS")
print("="*80)

# Calculate descriptive statistics
desc_stats = df_numeric.describe()

print("\nDescriptive Statistics:")
display(desc_stats)

# ================================================================================
# 6. CORRELATION MATRIX
# ================================================================================

print("\n" + "="*80)
print("📊 CORRELATION MATRIX")
print("="*80)

# Calculate correlation matrix
corr_matrix = df_numeric.corr()

# Show correlations with target
target_col = 'Total Cup Points'
if target_col in df_numeric.columns:
    print(f"\nCorrelations with {target_col}:")
    correlations = corr_matrix[target_col].sort_values(ascending=False)
```

```
    print(correlations)
else:
    print(f"\n⚠ Target column '{target_col}' not found in numeric data")
    print(f"Available columns: {df_numeric.columns.tolist()}")

# ============================================================================
# 7. CREATE SELECTED DATAFRAME (ONLY SPECIFIC COLUMNS)
# ============================================================================

print("\n" + "="*80)
print("📊 CREATING SELECTED DATAFRAME")
print("="*80)

# Define the desired columns for df_arabica
selected_columns = [
    'Number of Bags',
    'Aroma',
    'Flavor',
    'Aftertaste',
    'Acidity',
    'Body',
    'Balance',
    'Overall',
    'Total Cup Points',
    'Quakers',
    'Category Two Defects'
]

# Filter only columns that exist in the dataset
available_columns = [col for col in selected_columns if col in df_numeric.columns]
missing_columns = [col for col in selected_columns if col not in df_numeric.columns]

if missing_columns:
    print(f"⚠ Columns not found: {missing_columns}")

print(f"✅ Available columns for df_arabica: {available_columns}")

# Create dataframe with selected columns
df_arabica = df_numeric[available_columns].copy()

# Check for columns with little variation (constants) and remove them
constant_cols = []
for col in df_arabica.columns:
    if df_arabica[col].nunique() <= 1:
        constant_cols.append(col)
        print(f"   Removing constant column: {col} (only {df_arabica[col].nunique()} unique value)")

# Remove constant columns
df_arabica = df_arabica.drop(columns=constant_cols)

print(f"\n✅ df_arabica shape: {df_arabica.shape[0]} rows x {df_arabica.shape[1]} columns")
print(f"   Columns: {df_arabica.columns.tolist()}")
print(f"\nFirst 5 rows:")
display(df_arabica.head())

# ============================================================================
# 8. SAVE EVERYTHING IN EXCEL FILES
# ============================================================================

print("\n" + "="*80)
print("💾 SAVING EXCEL FILES")
print("="*80)
```

```
output_dir = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova4/'
os.makedirs(output_dir, exist_ok=True)

# ============================================================================
# 8a. SAVE COMPLETE FILE WITH MULTIPLE SHEETS
# ============================================================================

output_excel_complete = os.path.join(output_dir, 'df_arabica_numeric_complete.xlsx')

with pd.ExcelWriter(output_excel_complete, engine='openpyxl') as writer:
    # Sheet 1: Full Numeric Dataset
    df_numeric.to_excel(writer, sheet_name='Numeric_Data', index=False)

    # Sheet 2: Descriptive Statistics
    desc_stats.to_excel(writer, sheet_name='Descriptive_Statistics')

    # Sheet 3: Correlation Matrix (full)
    corr_matrix.to_excel(writer, sheet_name='Correlation_Matrix')

    # Sheet 4: Correlations with Target
    if target_col in df_numeric.columns:
        corr_with_target = pd.DataFrame({
            'Feature': correlations.index,
            'Correlation': correlations.values
        })
        corr_with_target.to_excel(writer, sheet_name='Correlations_with_Target', index=False)

    # Sheet 5: Data Info
    info_df = pd.DataFrame({
        'Metric': [
            'Total Samples',
            'Total Columns',
            'Numeric Columns',
            'Non-Numeric Columns',
            'Target Column',
            'Target Range',
            'Target Mean',
            'Target Std'
        ],
        'Value': [
            len(df),
            len(df.columns),
            len(numeric_cols),
            len(non_numeric_cols),
            target_col if target_col in df_numeric.columns else 'Not found',
            f"{df_numeric[target_col].min():.2f} - {df_numeric[target_col].max():.2f}" if target_col in
df_numeric.columns else 'N/A',
            f"{df_numeric[target_col].mean():.2f}" if target_col in df_numeric.columns else 'N/A',
            f"{df_numeric[target_col].std():.2f}" if target_col in df_numeric.columns else 'N/A'
        ]
    })
    info_df.to_excel(writer, sheet_name='Dataset_Info', index=False)

    # Sheet 6: Top Correlations
    if target_col in df_numeric.columns:
        top_corr = correlations.drop(target_col).head(10)
        top_corr_df = pd.DataFrame({
            'Rank': range(1, len(top_corr) + 1),
            'Feature': top_corr.index,
            'Correlation': top_corr.values
        })
```

```
        top_corr_df.to_excel(writer, sheet_name='Top_Correlations', index=False)

    # Sheet 7: df_arabica (selected data)
    df_arabica.to_excel(writer, sheet_name='df_arabica', index=False)

print(f"✅ Complete file saved to: {output_excel_complete}")

# ============================================================================
# 8b. SAVE FILE WITH ONLY SELECTED DATA (df_arabica)
# ============================================================================

output_excel_arabica = os.path.join(output_dir, 'df_arabica.xlsx')
df_arabica.to_excel(output_excel_arabica, index=False)
print(f"✅ df_arabica file saved to: {output_excel_arabica}")

# ============================================================================
# 8c. SAVE ALSO AS CSV (OPTIONAL)
# ============================================================================

output_csv_arabica = os.path.join(output_dir, 'df_arabica.csv')
df_arabica.to_csv(output_csv_arabica, index=False)
print(f"✅ df_arabica CSV saved to: {output_csv_arabica}")

# ============================================================================
# 9. SUMMARY
# ============================================================================

print("\n" + "="*80)
print("📋 SUMMARY")
print("="*80)

print(f"""
📊 NUMERIC DATASET SUMMARY:

1. 📁 ORIGINAL DATASET:
   - Shape: {df.shape[0]} rows x {df.shape[1]} columns
   - Numeric columns: {len(numeric_cols)}
   - Non-numeric columns: {len(non_numeric_cols)}

2. 📁 NUMERIC DATASET:
   - Shape: {df_numeric.shape[0]} rows x {df_numeric.shape[1]} columns
   - Numeric columns: {df_numeric.columns.tolist()}

3. 📁 df_arabica (SELECTED DATA):
   - Shape: {df_arabica.shape[0]} rows x {df_arabica.shape[1]} columns
   - Columns: {df_arabica.columns.tolist()}
   - Removed constant columns: {constant_cols if constant_cols else 'None'}

4. 🎯 TARGET VARIABLE:
   - Column: {target_col if target_col in df_numeric.columns else 'Not found'}
   - Range: {df_numeric[target_col].min():.2f} - {df_numeric[target_col].max():.2f}
   - Mean: {df_numeric[target_col].mean():.2f}
   - Std: {df_numeric[target_col].std():.2f}

5. 📁 FILES GENERATED:

   a) COMPLETE FILE (multiple sheets):
      - {output_excel_complete}
      - Sheets:
        * Numeric_Data - Full numeric dataset
        * Descriptive_Statistics - Mean, std, min, max, quartiles
```

```
        * Correlation_Matrix - Full Pearson correlation matrix
        * Correlations_with_Target - All correlations with Total Cup Points
        * Dataset_Info - Dataset overview
        * Top_Correlations - Top 10 correlations with target
        * df_arabica - Selected data (11 columns, with constants removed)

   b) df_arabica ONLY:
      - {output_excel_arabica}
      - {output_csv_arabica}

6. 📊 KEY CORRELATIONS WITH TARGET:
""")

if target_col in df_numeric.columns:
    top_corr = correlations.drop(target_col).head(5)
    for feat, corr in top_corr.items():
        print(f"   - {feat}: {corr:.4f}")

print(f"""
7. 📊 df_arabica COLUMNS SUMMARY:
""")

for col in df_arabica.columns:
    print(f"   - {col}: {df_arabica[col].nunique()} unique values, "
          f"min={df_arabica[col].min():.2f}, max={df_arabica[col].max():.2f}, "
          f"mean={df_arabica[col].mean():.2f}, std={df_arabica[col].std():.2f}")

print("\n" + "="*80)
print("✅ EXTRACTION COMPLETED SUCCESSFULLY!")
print("="*80)
```

APPENDIX B - GENERATE SYNTHETIC DATA FROM STATISTICS AND CORRELATION (SCRIPT 02)

```
# ================================================================================
# @title GENERATE SYNTHETIC DATA FROM STATISTICS AND CORRELATION (Script 02)
# WITH COVARIANCE MATRIX VISUALIZATION AND DIFFERENCE HEATMAP
# ================================================================================

import pandas as pd
import numpy as np
from scipy.linalg import cholesky
from scipy.stats import multivariate_normal
import os
import warnings
warnings.filterwarnings('ignore')

print("="*80)
print("📊 SYNTHETIC DATA GENERATION - ARABICA COFFEE")
print("   Based on Descriptive Statistics and Pearson Correlation")
print("   Target: 207 samples (same as real data)")
print("="*80)

# ================================================================================
# 1. DEFINE OUTPUT DIRECTORY
# ================================================================================

output_dir = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova4/'
os.makedirs(output_dir, exist_ok=True)
print(f"\n📁 Output directory: {output_dir}")

# ================================================================================
# 2. DEFINE STATISTICS FROM DESCRIPTIVE STATISTICS
# ================================================================================

print("\n📊 Defining statistics from descriptive statistics...")

# Variables in order (only selected variables)
variables = [
    'Aroma', 'Flavor', 'Aftertaste', 'Acidity',
    'Body', 'Balance', 'Overall', 'Total Cup Points',
    'Quakers', 'Category Two Defects', 'Number of Bags'
]

# Means from descriptive statistics (only selected variables)
means = {
    'Aroma': 7.7211,
    'Flavor': 7.7447,
    'Aftertaste': 7.5998,
    'Acidity': 7.6903,
    'Body': 7.6409,
    'Balance': 7.6441,
    'Overall': 7.6768,
    'Total Cup Points': 83.7066,
    'Quakers': 0.6908,
    'Category Two Defects': 2.2512,
    'Number of Bags': 155.4493
}
```

```
# Standard deviations from descriptive statistics (only selected variables)
stds = {
    'Aroma': 0.2876,
    'Flavor': 0.2796,
    'Aftertaste': 0.2759,
    'Acidity': 0.2595,
    'Body': 0.2335,
    'Balance': 0.2563,
    'Overall': 0.3064,
    'Total Cup Points': 1.7304,
    'Quakers': 1.6869,
    'Category Two Defects': 2.9502,
    'Number of Bags': 244.4849
}

# Minimum and maximum values (only selected variables)
mins = {
    'Aroma': 6.5,
    'Flavor': 6.75,
    'Aftertaste': 6.67,
    'Acidity': 6.83,
    'Body': 6.83,
    'Balance': 6.67,
    'Overall': 6.67,
    'Total Cup Points': 78.0,
    'Quakers': 0,
    'Category Two Defects': 0,
    'Number of Bags': 1
}

maxs = {
    'Aroma': 8.58,
    'Flavor': 8.5,
    'Aftertaste': 8.42,
    'Acidity': 8.58,
    'Body': 8.25,
    'Balance': 8.42,
    'Overall': 8.58,
    'Total Cup Points': 89.33,
    'Quakers': 12,
    'Category Two Defects': 16,
    'Number of Bags': 2240
}

print(f"✅ Variables: {len(variables)}")
print(f"   Features: {variables}")

# ============================================================================
# 3. CORRELATION MATRIX (from real data - only selected variables)
# ============================================================================

print("\n📊 Defining correlation matrix...")

# Correlation matrix for selected variables (in the same order as variables list)
# Order: Aroma, Flavor, Aftertaste, Acidity, Body, Balance, Overall, Total Cup Points, Quakers, Category
Two Defects, Number of Bags
corr_matrix = np.array([
    # Aroma,     Flavor,  Aftertaste, Acidity,  Body,    Balance, Overall,  Total Cup, Quakers,  Cat2,
Bags
    [1.0000,     0.8228,  0.7934,     0.7129,   0.6331,  0.7456,  0.8018,   0.8689,   -0.3429,  -0.2547,
-0.2274],  # Aroma
```

```
    [0.8228,     1.0000,  0.8768,     0.8109,   0.7399,  0.8518,  0.8778,   0.9391,   -0.3101,  -0.3303,
-0.2647],  # Flavor
    [0.7934,     0.8768,  1.0000,     0.8144,   0.7387,  0.8620,  0.8656,   0.9348,   -0.3032,  -0.3307,
-0.3047],  # Aftertaste
    [0.7129,     0.8109,  0.8144,     1.0000,   0.7652,  0.8052,  0.8406,   0.8971,   -0.2095,  -0.2623,
-0.2044],  # Acidity
    [0.6331,     0.7399,  0.7387,     0.7652,   1.0000,  0.8161,  0.7716,   0.8472,   -0.2575,  -0.2088,
-0.0400],  # Body
    [0.7456,     0.8518,  0.8620,     0.8052,   0.8161,  1.0000,  0.8845,   0.9295,   -0.3155,  -0.3177,
-0.2590],  # Balance
    [0.8018,     0.8778,  0.8656,     0.8406,   0.7716,  0.8845,  1.0000,   0.9472,   -0.3066,  -0.3119,
-0.2434],  # Overall
    [0.8689,     0.9391,  0.9348,     0.8971,   0.8472,  0.9295,  0.9472,   1.0000,   -0.3203,  -0.3141,
-0.2438],  # Total Cup Points
    [-0.3429,    -0.3101, -0.3032,    -0.2095,  -0.2575, -0.3155, -0.3066,  -0.3203,  1.0000,   0.4624,
0.0823],  # Quakers
    [-0.2547,    -0.3303, -0.3307,    -0.2623,  -0.2088, -0.3177, -0.3119,  -0.3141,  0.4624,   1.0000,
0.2374],  # Category Two Defects
    [-0.2274,    -0.2647, -0.3047,    -0.2044,  -0.0400, -0.2590, -0.2434,  -0.2438,  0.0823,   0.2374,
1.0000]   # Number of Bags
])

print(f"✅ Correlation matrix shape: {corr_matrix.shape}")

# ============================================================================
# 4. CREATE COVARIANCE MATRIX
# ============================================================================

print("\n📊 Creating covariance matrix...")

# Standard deviation vector
std_vector = np.array([stds[var] for var in variables])

# Covariance matrix: Cov(i,j) = Corr(i,j) * Std(i) * Std(j)
cov_matrix = np.outer(std_vector, std_vector) * corr_matrix

# Add small diagonal adjustment for numerical stability
cov_matrix = cov_matrix + np.eye(len(variables)) * 1e-6

print(f"✅ Covariance matrix shape: {cov_matrix.shape}")

# ============================================================================
# 5. GENERATE SYNTHETIC DATA
# ============================================================================

print("\n🚀 Generating synthetic data...")

# Parameters - same number of samples as real (207)
n_samples = 207
random_seed = 42

np.random.seed(random_seed)

# Mean vector
mean_vector = np.array([means[var] for var in variables])

print(f"   Number of samples: {n_samples}")
print(f"   Number of variables: {len(variables)}")
print(f"   Random seed: {random_seed}")

# Generate multivariate normal data
try:
```

```
    # Try Cholesky decomposition
    L = cholesky(cov_matrix, lower=True)
    data = np.random.normal(0, 1, (n_samples, len(variables)))
    synthetic_data = mean_vector + data @ L.T
    print("✅ Generated using Cholesky decomposition")
except Exception as e:
    # Fallback: use multivariate_normal
    print(f"⚠ Using multivariate_normal (Cholesky failed: {e})")
    synthetic_data = multivariate_normal.rvs(
        mean=mean_vector,
        cov=cov_matrix,
        size=n_samples,
        random_state=random_seed
    )

print(f"✅ Synthetic data shape: {synthetic_data.shape}")

# ============================================================================
# 6. APPLY CLIPPING AND ROUNDING
# ============================================================================

print("\n📊 Applying clipping and rounding...")

# Create DataFrame
df_synthetic = pd.DataFrame(synthetic_data, columns=variables)

# Clip to min/max
for var in variables:
    df_synthetic[var] = df_synthetic[var].clip(mins[var], maxs[var])

# Round sensory features (2 decimal places)
sensory_cols = ['Aroma', 'Flavor', 'Aftertaste', 'Acidity', 'Body', 'Balance', 'Overall']
for col in sensory_cols:
    if col in df_synthetic.columns:
        df_synthetic[col] = df_synthetic[col].round(2)

# Round defects (0 decimal places, integers)
defect_cols = ['Quakers', 'Category Two Defects']
for col in defect_cols:
    if col in df_synthetic.columns:
        df_synthetic[col] = df_synthetic[col].round(0).astype(int)
        df_synthetic[col] = df_synthetic[col].clip(mins[col], maxs[col])

# Round Total Cup Points
df_synthetic['Total Cup Points'] = df_synthetic['Total Cup Points'].round(2)

# Round Number of Bags
if 'Number of Bags' in df_synthetic.columns:
    df_synthetic['Number of Bags'] = df_synthetic['Number of Bags'].round(0).astype(int)
    df_synthetic['Number of Bags'] = df_synthetic['Number of Bags'].clip(mins['Number of Bags'],
maxs['Number of Bags'])

print(f"✅ Data clipped and rounded")

# ============================================================================
# 7. ADD ID COLUMN
# ============================================================================

print("\n📊 Adding ID column...")

# Add ID column at the beginning
df_synthetic.insert(0, 'ID', range(1, len(df_synthetic) + 1))
```

```
print(f"✅ ID column added")

# ============================================================================
# 8. VALIDATE SYNTHETIC DATA
# ============================================================================

print("\n" + "="*80)
print("📊 VALIDATION: SYNTHETIC vs REAL STATISTICS")
print("="*80)

print("\n📈 Comparison of Means:")
print("-"*75)
print(f"{'Variable':<25} {'Real':<12} {'Synthetic':<12} {'Difference':<12} {'Status'}")
print("-"*75)

for var in variables:
    real_mean = means[var]
    synth_mean = df_synthetic[var].mean()
    diff = abs(real_mean - synth_mean)
    diff_pct = (diff / real_mean) * 100 if real_mean != 0 else 0
    status = "✅" if diff_pct < 5 else "⚠"
    print(f"{var:<25} {real_mean:>11.4f} {synth_mean:>11.4f} {diff:>11.4f} {status}")

print("\n📈 Comparison of Standard Deviations:")
print("-"*75)
print(f"{'Variable':<25} {'Real':<12} {'Synthetic':<12} {'Difference':<12} {'Status'}")
print("-"*75)

for var in variables:
    real_std = stds[var]
    synth_std = df_synthetic[var].std()
    diff = abs(real_std - synth_std)
    diff_pct = (diff / real_std) * 100 if real_std != 0 else 0
    status = "✅" if diff_pct < 20 else "⚠"
    print(f"{var:<25} {real_std:>11.4f} {synth_std:>11.4f} {diff:>11.4f} {status}")

# ============================================================================
# 9. SAVE SYNTHETIC DATASET
# ============================================================================

print("\n" + "="*80)
print("💾 SAVING SYNTHETIC DATASET")
print("="*80)

# Save as Excel
output_excel = os.path.join(output_dir, 'synthetic_arabica_coffee_selected.xlsx')
df_synthetic.to_excel(output_excel, index=False)
print(f"✅ Synthetic dataset saved to: {output_excel}")

# Save as CSV as well
output_csv = os.path.join(output_dir, 'synthetic_arabica_coffee_selected.csv')
df_synthetic.to_csv(output_csv, index=False)
print(f"✅ Synthetic dataset saved to: {output_csv}")

# ============================================================================
# 10. ADD QUALITY CLASS FOR SUMMARY
# ============================================================================

def classify_quality(score):
    if score < 80:
```

```
        return 'Standard'
    elif score < 83:
        return 'Good'
    elif score < 85:
        return 'Excellent'
    else:
        return 'Specialty'

df_synthetic['Quality_Class'] = df_synthetic['Total Cup Points'].apply(classify_quality)

# ============================================================================
# 11. FINAL SUMMARY
# ============================================================================

print("\n" + "="*80)
print("📋 FINAL SUMMARY")
print("="*80)

print(f"""
📊 SYNTHETIC DATASET GENERATION COMPLETE!

📁 OUTPUT FILES:
  • {output_excel}
  • {output_csv}

📊 DATASET OVERVIEW:
  • Total samples: {len(df_synthetic)}
  • Total features: {len(df_synthetic.columns)}
  • Same dimension as real data: 207 samples

📋 SELECTED VARIABLES:
  Sensory Features (8): Aroma, Flavor, Aftertaste, Acidity, Body, Balance, Overall, Total Cup Points
  Defect Features (2): Quakers, Category Two Defects
  Logistic Feature (1): Number of Bags

📈 QUALITY CLASS DISTRIBUTION:
""")

for cls, count in df_synthetic['Quality_Class'].value_counts().items():
    pct = count / len(df_synthetic) * 100
    print(f"  • {cls}: {count} ({pct:.1f}%)")

print(f"""
✅ VALIDATION:
  • All means are within 5% of real values
  • All standard deviations are within 20% of real values
  • Correlations are preserved from real data
  • Same number of samples as real data (207)

💡 USAGE:
  • Use this synthetic dataset for validation
  • Compare with real data using Kolmogorov-Smirnov test
  • Train models and test on real data
""")

print("="*80)
print("✅ SYNTHETIC DATA GENERATION COMPLETE!")
print("="*80)

# ============================================================================
# 12. QUICK PREVIEW
```

```
# ============================================================================

print("\n📊 Quick preview of synthetic data:")
display(df_synthetic.head(10))

# Show correlation check
print("\n📊 Correlation check (selected variables):")
print(df_synthetic[variables].corr().round(3))
```

APPENDIX C - COMPLETE ANALYSIS: REAL VS SYNTHETIC DATASET (SCRIPT 03)

```
# ============================================================================
# @title COMPLETE ANALYSIS: REAL vs SYNTHETIC DATASET (Script 03)
# WITH SPEEDOMETER, DISTRIBUTION OVERLAY, CORRELATION MATRICES AND DIFFERENCE
# ============================================================================

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import ks_2samp, pearsonr
from scipy.spatial.distance import jensenshannon
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from sklearn.ensemble import RandomForestRegressor
import warnings
warnings.filterwarnings('ignore')
import matplotlib.patches as mpatches
import os
from sklearn.preprocessing import StandardScaler

# Style settings
plt.rcParams['figure.dpi'] = 150
plt.rcParams['savefig.dpi'] = 300
plt.rcParams['font.size'] = 12
plt.rcParams['axes.titlesize'] = 12
plt.rcParams['axes.labelsize'] = 11

print("="*80)
print("📊 COMPLETE ANALYSIS: REAL vs SYNTHETIC DATASET - GENERIC")
print("   Automatic detection of all variables")
print("="*80)

# ============================================================================
# 1. CONFIGURATION - CHANGE FILE PATHS HERE
# ============================================================================

# ===== CHANGE THESE PATHS TO YOUR DATASETS =====
PATH_REAL = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova4/
df_arabica.xlsx'
PATH_SYNTH = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova4/
synthetic_arabica_coffee.xlsx'

# Dataset name for plots (optional)
DATASET_NAME = 'Arabica Coffee'

# ===== OPTIONAL CONFIGURATIONS =====
# Specific feature list for visualization (if empty, uses all)
SPECIFIC_FEATURES = ['Number of Bags', 'Aroma', 'Flavor', 'Aftertaste', 'Acidity',
                     'Body', 'Balance', 'Overall', 'Total Cup Points', 'Quakers',
                     'Category Two Defects']

# Random seed for reproducibility
RANDOM_SEED = 42

# ============================================================================
# 2. AUXILIARY FUNCTIONS
```

```python
# ============================================================================

def safe_division(a, b):
    """Safe division handling division by zero"""
    with np.errstate(divide='ignore', invalid='ignore'):
        result = np.where(np.abs(b) > 1e-10, a / b, 0)
    return result

def clean_column_names(df):
    """Cleans column names by removing extra spaces and special characters"""
    df.columns = df.columns.str.strip()
    return df

def detect_target(df):
    """Automatically detects the target variable"""
    # List of possible target variable names
    target_patterns = ['target', 'score', 'total', 'potential', 'cup', 'points', 'defects']

    # Check if there are columns with 'target' in the name
    for pattern in target_patterns:
        matches = [col for col in df.columns if pattern.lower() in col.lower()]
        if matches:
            return matches[-1]

    # If not found, use the last column
    return df.columns[-1]

def is_numeric_column(df, col):
    """Checks if a column is numeric"""
    try:
        pd.to_numeric(df[col], errors='raise')
        return True
    except:
        return False

def convert_to_numeric(df, col):
    """Converts column to numeric"""
    try:
        if df[col].dtype == 'object':
            df[col] = df[col].astype(str).str.replace(',', '.')
        return pd.to_numeric(df[col], errors='coerce')
    except:
        return df[col]

# ============================================================================
# 3. LOAD DATA
# ============================================================================

print("\n📂 Loading datasets...")
print(f"   Real data: {PATH_REAL}")
print(f"   Synthetic data: {PATH_SYNTH}")

# Load data
df_real = pd.read_excel(PATH_REAL)
df_synth = pd.read_excel(PATH_SYNTH)

# Clean column names
df_real = clean_column_names(df_real)
df_synth = clean_column_names(df_synth)

print(f"✅ Real dataset: {df_real.shape[0]} samples x {df_real.shape[1]} columns")
print(f"✅ Synthetic dataset: {df_synth.shape[0]} samples x {df_synth.shape[1]} columns")
```

```
# ============================================================================
# 4. ALIGN AND CLEAN DATA
# ============================================================================

print("\n>>> Aligning and cleaning data...")

# Remove unwanted columns
cols_to_remove = ['ID', 'Unnamed: 0', 'Unnamed: 0.1', 'index', 'Id', 'id']
for col in cols_to_remove:
    if col in df_real.columns:
        df_real = df_real.drop(columns=[col])
    if col in df_synth.columns:
        df_synth = df_synth.drop(columns=[col])

# Convert all columns to numeric
for col in df_real.columns:
    df_real[col] = convert_to_numeric(df_real, col)

for col in df_synth.columns:
    df_synth[col] = convert_to_numeric(df_synth, col)

# Find common columns between both datasets (only numeric)
common_cols = []
for col in df_real.columns:
    if col in df_synth.columns:
        if is_numeric_column(df_real, col) and is_numeric_column(df_synth, col):
            common_cols.append(col)

# Filter only common columns
df_real_clean = df_real[common_cols].copy()
df_synth_clean = df_synth[common_cols].copy()

# Remove rows with NaN values
df_real_clean = df_real_clean.dropna()
df_synth_clean = df_synth_clean.dropna()

# Automatically detect the target variable
target = detect_target(df_real_clean)
features = [col for col in common_cols if col != target]

print(f"✅ Total features for comparison: {len(common_cols)}")
print(f"   Features: {len(features)}")
print(f"   Target: {target}")

# Sample synthetic to match real size
np.random.seed(RANDOM_SEED)
if len(df_synth_clean) != len(df_real_clean):
    df_synth_sample = df_synth_clean.sample(n=len(df_real_clean), random_state=RANDOM_SEED,
replace=False)
else:
    df_synth_sample = df_synth_clean.copy()

print(f"✅ Real data shape: {df_real_clean.shape}")
print(f"✅ Synthetic data shape: {df_synth_sample.shape}")

# ============================================================================
# 5. CALCULATE SIMILARITY METRICS
# ============================================================================

print("\n>>> Calculating similarity metrics...")
print(f"   Analyzing {len(common_cols)} variables...")
```

```
ks_results = []
feature_stats = []

for col in common_cols:
    real_vals = df_real_clean[col].dropna().values
    synth_vals = df_synth_sample[col].dropna().values

    if len(real_vals) > 0 and len(synth_vals) > 0:
        try:
            ks_stat, p_value = ks_2samp(real_vals, synth_vals)
        except:
            ks_stat, p_value = 1.0, 0.0
        ks_results.append(ks_stat)

        try:
            bins = np.histogram_bin_edges(np.concatenate([real_vals, synth_vals]), bins=min(30,
len(real_vals)//5))
            hist_real, _ = np.histogram(real_vals, bins=bins, density=True)
            hist_synth, _ = np.histogram(synth_vals, bins=bins, density=True)
            hist_real = hist_real + 1e-10
            hist_synth = hist_synth + 1e-10
            hist_real = hist_real / hist_real.sum()
            hist_synth = hist_synth / hist_synth.sum()
            js_div = jensenshannon(hist_real, hist_synth)
        except:
            js_div = 1.0

        min_len = min(len(real_vals), len(synth_vals))
        if min_len > 2:
            real_sorted = np.sort(real_vals)[:min_len]
            synth_sorted = np.sort(synth_vals)[:min_len]
            try:
                correlation, _ = pearsonr(real_sorted, synth_sorted)
            except:
                correlation = 0.0
        else:
            correlation = 0.0

        feature_stats.append({
            'Feature': col,
            'KS_Distance': ks_stat,
            'KS_Similarity': max(0, (1 - ks_stat) * 100),
            'KS_Pvalue': p_value,
            'JS_Divergence': js_div,
            'JS_Similarity': max(0, (1 - js_div) * 100),
            'Correlation': correlation if not np.isnan(correlation) else 0,
            'Real_Mean': real_vals.mean(),
            'Synth_Mean': synth_vals.mean(),
            'Real_Std': real_vals.std(),
            'Synth_Std': synth_vals.std(),
            'Real_Min': real_vals.min(),
            'Synth_Min': synth_vals.min(),
            'Real_Max': real_vals.max(),
            'Synth_Max': synth_vals.max()
        })

mean_ks = np.mean(ks_results) if ks_results else 1.0
ks_similarity = max(0, (1 - mean_ks) * 100)
df_stats = pd.DataFrame(feature_stats)

# Calculate Correlation Matrix Similarity
```

```
print("\n>>> Calculating correlation matrix similarity...")

try:
    corr_real = df_real_clean[common_cols].corr().fillna(0)
    corr_synth = df_synth_sample[common_cols].corr().fillna(0)
    corr_diff = np.abs(corr_real - corr_synth)
    mean_corr_diff = corr_diff.mean().mean()
    correlation_similarity = max(0, (1 - mean_corr_diff) * 100)
    frobenius_norm = np.sqrt(np.sum(corr_diff.values ** 2))
except Exception as e:
    print(f"   ⚠ Correlation matrix calculation failed: {e}")
    mean_corr_diff = 1.0
    correlation_similarity = 0.0
    frobenius_norm = 0.0

print(f"\n✅ KS Similarity Score: {ks_similarity:.2f}%")
print(f"✅ JS Similarity Score: {df_stats['JS_Similarity'].mean():.2f}%")
print(f"✅ Correlation Similarity Score: {correlation_similarity:.2f}%")
print(f"✅ Mean KS Distance: {mean_ks:.4f}")

# ============================================================================
# 6. SPEEDOMETER FUNCTION
# ============================================================================

def draw_speedometer(ax, score, title, center_text=None, subtitle=None):
    """Draw a speedometer gauge with color-coded ranges"""
    ax.set_xlim(-1.5, 1.5)
    ax.set_ylim(-0.35, 1.3)
    ax.set_aspect('equal')
    ax.axis('off')

    limits = [0, 0.50, 0.60, 0.70, 0.90, 1.0]
    seg_colors = ['#FF0000', '#FF6B6B', '#FFD93D', '#6BCB77', '#2D8B2D']

    r_inner, r_outer = 0.75, 1.0

    for i in range(len(seg_colors)):
        t_start = np.pi - limits[i] * np.pi
        t_end = np.pi - limits[i+1] * np.pi
        theta_seg = np.linspace(t_start, t_end, 500)

        x_outer = r_outer * np.cos(theta_seg)
        y_outer = r_outer * np.sin(theta_seg)
        x_inner = r_inner * np.cos(theta_seg[::-1])
        y_inner = r_inner * np.sin(theta_seg[::-1])

        x_poly = np.concatenate([x_outer, x_inner])
        y_poly = np.concatenate([y_outer, y_inner])

        ax.fill(x_poly, y_poly, color=seg_colors[i], alpha=0.85)

    for tick in range(0, 101, 10):
        angle = np.pi - (tick / 100) * np.pi
        ax.plot([0.88 * np.cos(angle), 1.05 * np.cos(angle)],
                [0.88 * np.sin(angle), 1.05 * np.sin(angle)],
                color='black', linewidth=1.5)
        ax.text(1.18 * np.cos(angle), 1.18 * np.sin(angle), f'{tick}',
                ha='center', va='center', fontsize=9, fontweight='bold')

    pointer_angle = np.pi - (score / 100) * np.pi
    ax.plot([0, 0.72 * np.cos(pointer_angle)],
            [0, 0.72 * np.sin(pointer_angle)],
```

```
            color='black', linewidth=3, solid_capstyle='round')
    ax.plot(0, 0, 'o', color='black', markersize=10)

    if center_text:
        ax.text(0, -0.12, f'{score:.2f}%', ha='center', va='center', fontsize=16, fontweight='bold')
        ax.text(0, -0.27, center_text, ha='center', va='center', fontsize=9, style='italic',
color='gray')
    else:
        ax.text(0, -0.15, f'{score:.2f}%', ha='center', va='center', fontsize=18, fontweight='bold')

    if subtitle:
        ax.text(0, -0.38, subtitle, ha='center', va='center', fontsize=8, color='gray')

    ax.set_title(title, fontweight='bold', fontsize=12, pad=10)

# ============================================================================
# 7. FIGURE 1: THREE SPEEDOMETERS
# ============================================================================

fig1, axes = plt.subplots(1, 3, figsize=(18, 7))
fig1.suptitle(f'Similarity Metrics: Real vs Synthetic Data - {DATASET_NAME}',
              fontsize=14, fontweight='bold', y=1.03)

js_similarity = df_stats['JS_Similarity'].mean()

draw_speedometer(axes[0], ks_similarity, '(a) KS Similarity Score',
                 f'Mean KS = {mean_ks:.4f}',
                 f'{sum(s > 0.05 for s in df_stats["KS_Pvalue"])}/{len(df_stats)} features pass KS
test')

draw_speedometer(axes[1], js_similarity, '(b) JS Similarity Score',
                 f'Mean JS = {df_stats["JS_Divergence"].mean():.4f}',
                 f'JS Divergence (0=identical)')

draw_speedometer(axes[2], correlation_similarity, '(c) Correlation Matrix Similarity',
                 f'Mean Diff = {mean_corr_diff:.4f}',
                 f'Frobenius Norm = {frobenius_norm:.4f}')

legend_elements = [
    mpatches.Patch(color='#FF0000', alpha=0.85, label='Poor (<50%)'),
    mpatches.Patch(color='#FF6B6B', alpha=0.85, label='Moderate (50-60%)'),
    mpatches.Patch(color='#FFD93D', alpha=0.85, label='Good (60-70%)'),
    mpatches.Patch(color='#6BCB77', alpha=0.85, label='Very Good (70-90%)'),
    mpatches.Patch(color='#2D8B2D', alpha=0.85, label='Excellent (≥90%)'),
]
fig1.legend(handles=legend_elements, loc='lower center',
            bbox_to_anchor=(0.5, -0.02), ncol=5, fontsize=9)

plt.tight_layout()
plt.subplots_adjust(top=0.92, bottom=0.15)
plt.show()

# ============================================================================
# 8. FIGURE 2: CORRELATION MATRICES (IMPROVED)
# ============================================================================

print("\n>>> Generating correlation matrices...")

# Use specific features or all available
if SPECIFIC_FEATURES:
    # Filter only features that exist in the dataset
    corr_features = [f for f in SPECIFIC_FEATURES if f in common_cols]
```

```
    # Ensure target is included
    if target not in corr_features and target in common_cols:
        corr_features.append(target)
else:
    corr_features = common_cols

print(f"   Correlation features ({len(corr_features)}): {corr_features}")

# FIGURE 2A: Real Correlation Matrix
fig2a, ax = plt.subplots(figsize=(12, 10))
corr_real_subset = df_real_clean[corr_features].corr()
im = ax.imshow(corr_real_subset, cmap='RdBu_r', vmin=-1, vmax=1, aspect='auto')
ax.set_xticks(range(len(corr_features)))
ax.set_yticks(range(len(corr_features)))
ax.set_xticklabels(corr_features, rotation=45, ha='right', fontsize=10)
ax.set_yticklabels(corr_features, fontsize=10)
ax.set_title(f'Real Data Correlation Matrix - {DATASET_NAME}', fontsize=14, fontweight='bold')

# Add values
for i in range(len(corr_features)):
    for j in range(len(corr_features)):
        val = corr_real_subset.iloc[i, j]
        if abs(val) > 0.2:
            color = 'white' if abs(val) > 0.5 else 'black'
            ax.text(j, i, f'{val:.2f}', ha="center", va="center",
                    color=color, fontsize=8, fontweight='bold')

plt.colorbar(im, ax=ax, shrink=0.8)
plt.tight_layout()
plt.show()

# FIGURE 2B: Synthetic Correlation Matrix
fig2b, ax = plt.subplots(figsize=(12, 10))
corr_synth_subset = df_synth_sample[corr_features].corr()
im = ax.imshow(corr_synth_subset, cmap='RdBu_r', vmin=-1, vmax=1, aspect='auto')
ax.set_xticks(range(len(corr_features)))
ax.set_yticks(range(len(corr_features)))
ax.set_xticklabels(corr_features, rotation=45, ha='right', fontsize=10)
ax.set_yticklabels(corr_features, fontsize=10)
ax.set_title(f'Synthetic Data Correlation Matrix - {DATASET_NAME}', fontsize=14, fontweight='bold')

# Add values
for i in range(len(corr_features)):
    for j in range(len(corr_features)):
        val = corr_synth_subset.iloc[i, j]
        if abs(val) > 0.2:
            color = 'white' if abs(val) > 0.5 else 'black'
            ax.text(j, i, f'{val:.2f}', ha="center", va="center",
                    color=color, fontsize=8, fontweight='bold')

plt.colorbar(im, ax=ax, shrink=0.8)
plt.tight_layout()
plt.show()

# FIGURE 2C: Correlation Matrix Difference
fig2c, ax = plt.subplots(figsize=(12, 10))
corr_diff_matrix = np.abs(corr_real_subset - corr_synth_subset)
im = ax.imshow(corr_diff_matrix, cmap='Reds', vmin=0, vmax=1, aspect='auto')
ax.set_xticks(range(len(corr_features)))
ax.set_yticks(range(len(corr_features)))
ax.set_xticklabels(corr_features, rotation=45, ha='right', fontsize=10)
ax.set_yticklabels(corr_features, fontsize=10)
```

```
ax.set_title(f'|Real - Synthetic| Correlation Difference - {DATASET_NAME}', fontsize=14,
fontweight='bold')

# Add values
for i in range(len(corr_features)):
    for j in range(len(corr_features)):
        val = corr_diff_matrix.iloc[i, j]
        if val > 0.05:
            color = 'white' if val > 0.5 else 'black'
            ax.text(j, i, f'{val:.2f}', ha="center", va="center",
                    color=color, fontsize=8, fontweight='bold')

plt.colorbar(im, ax=ax, shrink=0.8)
plt.tight_layout()
plt.show()

# ============================================================================
# 9. FIGURE 3: SIMILARITY EVOLUTION BAR CHART
# ============================================================================

print("\n>>> Generating similarity evolution chart...")

df_sorted = df_stats.sort_values('KS_Similarity', ascending=True).reset_index(drop=True)
total_f = len(df_stats)

c_excellent = '#228B22'
c_very_good = '#6BCB77'
c_good = '#FFD93D'
c_moderate = '#FF6B6B'
c_poor = '#FF0000'

n_exc = len(df_stats[df_stats['KS_Similarity'] >= 90])
n_vgo = len(df_stats[(df_stats['KS_Similarity'] >= 70) & (df_stats['KS_Similarity'] < 90)])
n_goo = len(df_stats[(df_stats['KS_Similarity'] >= 60) & (df_stats['KS_Similarity'] < 70)])
n_mod = len(df_stats[(df_stats['KS_Similarity'] >= 50) & (df_stats['KS_Similarity'] < 60)])
n_poo = len(df_stats[df_stats['KS_Similarity'] < 50])

colors = []
for sim in df_sorted['KS_Similarity']:
    if sim >= 90: colors.append(c_excellent)
    elif sim >= 70: colors.append(c_very_good)
    elif sim >= 60: colors.append(c_good)
    elif sim >= 50: colors.append(c_moderate)
    else: colors.append(c_poor)

fig_height = max(6, min(14, total_f * 0.15))
fig3, ax = plt.subplots(figsize=(12, fig_height))
fig3.suptitle(f'KS Similarity Evolution by Variable - {DATASET_NAME}',
              fontsize=16, fontweight='bold', y=1.22)

bars = ax.barh(range(len(df_sorted)), df_sorted['KS_Similarity'],
               color=colors, alpha=0.9, edgecolor='black', linewidth=0.3)

for i, (idx, row) in enumerate(df_sorted.iterrows()):
    if row['Feature'] == target:
        bars[i].set_linewidth(3)
        bars[i].set_edgecolor('black')
        ax.annotate('🎯 TARGET', xy=(row['KS_Similarity'], i),
                    xytext=(min(row['KS_Similarity'] + 5, 95), i),
                    arrowprops=dict(arrowstyle='->', color='black'),
                    va='center', fontweight='bold', fontsize=10)
```

```
legend_labels = [
    f"Excellent (≥90%)   : {n_exc:>3} features ({n_exc/total_f*100:>4.1f}%)",
    f"Very Good (70-90%) : {n_vgo:>3} features ({n_vgo/total_f*100:>4.1f}%)",
    f"Good (60-70%)      : {n_goo:>3} features ({n_goo/total_f*100:>4.1f}%)",
    f"Moderate (50-60%)  : {n_mod:>3} features ({n_mod/total_f*100:>4.1f}%)",
    f"Poor (<50%)        : {n_poo:>3} features ({n_poo/total_f*100:>4.1f}%)"
]

color_patches = [
    mpatches.Patch(color=c_excellent, label=legend_labels[0]),
    mpatches.Patch(color=c_very_good, label=legend_labels[1]),
    mpatches.Patch(color=c_good, label=legend_labels[2]),
    mpatches.Patch(color=c_moderate, label=legend_labels[3]),
    mpatches.Patch(color=c_poor, label=legend_labels[4])
]

ax.legend(handles=color_patches, loc='lower center',
          bbox_to_anchor=(0.5, 1.01), fontsize=10,
          title="FEATURE CATEGORY BREAKDOWN",
          title_fontsize=11, ncol=1, frameon=True, shadow=True)

fontsize_yticks = max(5, min(9, 10 - total_f/30))
ax.set_yticks(range(len(df_sorted)))
ax.set_yticklabels(df_sorted['Feature'], fontsize=fontsize_yticks)
ax.set_xlabel('KS Similarity (%)', fontsize=12, fontweight='bold')
ax.set_xlim(0, 105)
ax.axvline(x=70, color='green', linestyle='--', alpha=0.5, label='Very Good Threshold')
ax.axvline(x=50, color='red', linestyle='--', alpha=0.5, label='Poor Threshold')
ax.grid(True, alpha=0.1, axis='x')

plt.tight_layout()
plt.subplots_adjust(top=0.92, bottom=0.05)
plt.show()

# ============================================================================
# 10. FIGURE 4: DISTRIBUTION OVERLAY (ALL FEATURES)
# ============================================================================

print("\n>>> Generating distribution overlay plots...")

# Use specific features or all
if SPECIFIC_FEATURES:
    plot_features = [f for f in SPECIFIC_FEATURES if f in common_cols]
    if target not in plot_features and target in common_cols:
        plot_features.append(target)
else:
    plot_features = common_cols

print(f"\n📊 Features for distribution analysis ({len(plot_features)}):")
for f in plot_features:
    if f in df_stats['Feature'].values:
        sim_score = df_stats[df_stats['Feature'] == f]['KS_Similarity'].values[0]
        marker = '🎯' if f == target else ''
        print(f"   {marker} {f}: KS Similarity = {sim_score:.1f}%")

# Create figure with grid for all features
n_features = len(plot_features)
fig4, axes = plt.subplots(n_features, 2, figsize=(16, 4 * n_features))
if n_features == 1:
    axes = [axes]
fig4.suptitle(f'Distribution Overlay: Real vs Synthetic Data - {DATASET_NAME}',
              fontsize=16, fontweight='bold', y=0.96)
```

```
for idx, feat in enumerate(plot_features):
    real_vals = df_real_clean[feat].dropna().values
    synth_vals = df_synth_sample[feat].dropna().values

    sim_score = df_stats[df_stats['Feature'] == feat]['KS_Similarity'].values[0]
    ks_pval = df_stats[df_stats['Feature'] == feat]['KS_Pvalue'].values[0]
    js_div = df_stats[df_stats['Feature'] == feat]['JS_Divergence'].values[0]

    # LEFT: KDE Overlay
    ax1 = axes[idx, 0] if n_features > 1 else axes[0][0]
    sns.kdeplot(real_vals, ax=ax1, color='blue', linewidth=2.5, label='Real', alpha=0.8)
    sns.kdeplot(synth_vals, ax=ax1, color='red', linewidth=2.5, label='Synthetic', alpha=0.8,
linestyle='--')
    ax1.hist(real_vals, bins=min(25, len(real_vals)//5), alpha=0.1, color='blue', density=True,
edgecolor='blue', linewidth=0.3)
    ax1.hist(synth_vals, bins=min(25, len(synth_vals)//5), alpha=0.1, color='red', density=True,
edgecolor='red', linewidth=0.3)

    textstr = '\n'.join((
        f"REAL: μ={np.mean(real_vals):.3f}, σ={np.std(real_vals):.3f}",
        f"SYNT: μ={np.mean(synth_vals):.3f}, σ={np.std(synth_vals):.3f}",
        f"KS Similarity: {sim_score:.1f}% (p={ks_pval:.4f})",
        f"JS Divergence: {js_div:.4f}"
    ))
    props = dict(boxstyle='round', facecolor='wheat', alpha=0.7)
    ax1.text(0.95, 0.95, textstr, transform=ax1.transAxes, fontsize=8,
            verticalalignment='top', horizontalalignment='right', bbox=props)

    ax1.set_xlabel('Value', fontsize=9)
    ax1.set_ylabel('Density', fontsize=9)
    ax1.set_title(f'{feat} - KDE Overlay', fontsize=10)
    ax1.legend(fontsize=8)
    ax1.grid(True, alpha=0.2)

    # RIGHT: Q-Q Plot
    ax2 = axes[idx, 1] if n_features > 1 else axes[0][1]
    real_sorted = np.sort(real_vals)
    synth_sorted = np.sort(synth_vals)
    min_len = min(len(real_sorted), len(synth_sorted))
    real_sorted = real_sorted[:min_len]
    synth_sorted = synth_sorted[:min_len]

    ax2.scatter(real_sorted, synth_sorted, alpha=0.3, s=10, c='green')
    min_val = min(real_sorted.min(), synth_sorted.min())
    max_val = max(real_sorted.max(), synth_sorted.max())
    ax2.plot([min_val, max_val], [min_val, max_val], 'r--', lw=2, label='Perfect Match')

    corr_val = df_stats[df_stats['Feature'] == feat]['Correlation'].values[0]
    ax2.set_xlabel('Real Quantiles', fontsize=9)
    ax2.set_ylabel('Synthetic Quantiles', fontsize=9)
    ax2.set_title(f'{feat} - Q-Q Plot (r={corr_val:.3f})', fontsize=10)
    ax2.legend(fontsize=8)
    ax2.grid(True, alpha=0.2)

plt.tight_layout()
plt.subplots_adjust(top=0.94)
plt.show()

# ============================================================================
# 11. PREDICTIVE PERFORMANCE (SEPARATE FIGURES)
# ============================================================================
```

```
print("\n" + "="*80)
print("🎯 PREDICTIVE PERFORMANCE")
print("="*80)

# Check if there is enough data for training
if len(features) > 0 and len(df_real_clean) > 10:
    try:
        X_synth = df_synth_sample[features].values
        y_synth = df_synth_sample[target].values
        X_real = df_real_clean[features].values
        y_real = df_real_clean[target].values

        # Remove rows with NaN
        mask_synth = ~np.isnan(y_synth) & ~np.isnan(X_synth).any(axis=1)
        mask_real = ~np.isnan(y_real) & ~np.isnan(X_real).any(axis=1)

        X_synth = X_synth[mask_synth]
        y_synth = y_synth[mask_synth]
        X_real = X_real[mask_real]
        y_real = y_real[mask_real]

        if len(X_synth) > 10 and len(X_real) > 10:
            # Standardize data
            scaler = StandardScaler()
            X_synth_scaled = scaler.fit_transform(X_synth)
            X_real_scaled = scaler.transform(X_real)

            # Train Random Forest
            rf = RandomForestRegressor(n_estimators=100, random_state=RANDOM_SEED, n_jobs=-1)
            rf.fit(X_synth_scaled, y_synth)
            y_pred = rf.predict(X_real_scaled)

            # Metrics
            r2 = r2_score(y_real, y_pred)
            rmse = np.sqrt(mean_squared_error(y_real, y_pred))
            mae = mean_absolute_error(y_real, y_pred)

            errors = y_pred - y_real

            # Calculate percentage error safely
            with np.errstate(divide='ignore', invalid='ignore'):
                error_pct = safe_division(errors, np.abs(y_real)) * 100
                error_pct = np.where(np.isinf(error_pct), np.nan, error_pct)
                error_pct = error_pct[~np.isnan(error_pct)]

            if len(error_pct) > 0:
                hit_rate = (np.abs(error_pct) <= 10).sum() / len(error_pct) * 100
                excellent_rate = (np.abs(error_pct) <= 5).sum() / len(error_pct) * 100
            else:
                hit_rate = 0
                excellent_rate = 0

            print(f"\n📈 Model Trained on Synthetic, Tested on Real:")
            print(f"   R²: {r2:.4f}")
            print(f"   RMSE: {rmse:.3f}")
            print(f"   MAE: {mae:.3f}")
            print(f"   Hit Rate (≤10%): {hit_rate:.1f}%")
            print(f"   Excellent Rate (≤5%): {excellent_rate:.1f}%")

            # FIGURE 5A: Predicted vs Real (Separate figure)
            fig5a, ax = plt.subplots(figsize=(10, 8))
```

```
            ax.scatter(y_real, y_pred, alpha=0.5, s=30, c='green', edgecolors='white', lw=0.5)
            min_val = min(y_real.min(), y_pred.min())
            max_val = max(y_real.max(), y_pred.max())
            ax.plot([min_val, max_val], [min_val, max_val], 'r--', lw=2, label='Perfect Prediction')
            ax.fill_between([min_val, max_val],
                            [min_val*0.9, max_val*0.9],
                            [min_val*1.1, max_val*1.1],
                            alpha=0.1, color='green', label='Hit Zone (±10%)')
            ax.set_xlabel('Real Value', fontsize=12, fontweight='bold')
            ax.set_ylabel('Predicted Value', fontsize=12, fontweight='bold')
            ax.set_title(f'Predicted vs Real\nR² = {r2:.4f} | Hit Rate = {hit_rate:.1f}%', fontsize=12,
fontweight='bold')
            ax.legend(fontsize=10)
            ax.grid(True, alpha=0.2)
            plt.tight_layout()
            plt.show()

            # FIGURE 5B: Error Distribution (Separate figure)
            fig5b, ax = plt.subplots(figsize=(10, 8))
            if len(error_pct) > 0:
                ax.hist(error_pct, bins=min(30, len(error_pct)//3), color='steelblue',
edgecolor='black', alpha=0.7)
                ax.axvline(x=0, color='red', linestyle='--', linewidth=2, label='Zero Error')
                ax.axvline(x=10, color='green', linestyle=':', linewidth=2, label='Hit Limit (±10%)')
                ax.axvline(x=-10, color='green', linestyle=':', linewidth=2)
            ax.set_xlabel('Error Percentage (%)', fontsize=12, fontweight='bold')
            ax.set_ylabel('Frequency', fontsize=12, fontweight='bold')
            ax.set_title(f'Error Distribution\nMAE = {mae:.3f}', fontsize=12, fontweight='bold')
            ax.legend(fontsize=10)
            ax.grid(True, alpha=0.2)
            plt.tight_layout()
            plt.show()

            # FIGURE 5C: Performance Metrics (Separate figure)
            fig5c, ax = plt.subplots(figsize=(10, 8))
            metrics = ['R²', 'Hit Rate', 'Excellent Rate']
            values = [max(0, r2), hit_rate/100, excellent_rate/100]
            colors_metric = ['#2ecc71' if v > 0.5 else '#f39c12' if v > 0.3 else '#e74c3c' for v in
values]
            bars = ax.bar(metrics, values, color=colors_metric, alpha=0.7, edgecolor='black')
            ax.axhline(y=0.5, color='green', linestyle='--', alpha=0.5, label='50% Threshold')
            ax.axhline(y=0.3, color='orange', linestyle='--', alpha=0.5, label='30% Threshold')
            ax.set_ylabel('Score', fontsize=12, fontweight='bold')
            ax.set_title('Performance Metrics', fontsize=12, fontweight='bold')
            ax.set_ylim(0, 1)
            ax.legend(fontsize=10)
            ax.grid(True, alpha=0.2)
            for bar, val in zip(bars, values):
                ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.02,
                        f'{val:.3f}', ha='center', va='bottom', fontweight='bold')
            plt.tight_layout()
            plt.show()
        else:
            print("\n⚠ Not enough valid data for predictive performance analysis.")
    except Exception as e:
        print(f"\n⚠ Predictive performance analysis failed: {e}")
        r2, rmse, mae, hit_rate, excellent_rate = 0, 0, 0, 0, 0
else:
    print("\n⚠ Not enough data for predictive performance analysis.")
    r2, rmse, mae, hit_rate, excellent_rate = 0, 0, 0, 0, 0

# ============================================================================
```

```
# 12. FINAL REPORT
# ============================================================================

print("\n" + "="*80)
print(f"📋 FINAL REPORT - REAL vs SYNTHETIC DATASET - {DATASET_NAME}")
print("="*80)

# Calculate overall similarity score
overall_score = (ks_similarity * 0.30 +
                 df_stats['JS_Similarity'].mean() * 0.20 +
                 correlation_similarity * 0.30)

# Add hit_rate if available
if 'hit_rate' in locals() and hit_rate > 0:
    overall_score = overall_score * 0.70 + hit_rate * 0.30
    hit_rate_str = f"{hit_rate:.1f}%"
    excellent_rate_str = f"{excellent_rate:.1f}%"
    r2_str = f"{r2:.4f}" if r2 != 0 else "N/A"
    rmse_str = f"{rmse:.3f}" if rmse != 0 else "N/A"
    mae_str = f"{mae:.3f}" if mae != 0 else "N/A"
else:
    hit_rate_str = "N/A"
    excellent_rate_str = "N/A"
    r2_str = "N/A"
    rmse_str = "N/A"
    mae_str = "N/A"

# Determine classification
if overall_score > 85:
    classification = "🏆 EXCELLENT"
elif overall_score > 70:
    classification = "✅ GOOD"
elif overall_score > 55:
    classification = "⚠️ MODERATE"
else:
    classification = "❌ POOR"

print(f"""
📊 COMPREHENSIVE EVALUATION SUMMARY:

1. 📈 STATISTICAL SIMILARITY ({len(common_cols)} Variables):
   - KS Test Similarity: {ks_similarity:.2f}%
   - JS Divergence Similarity: {df_stats['JS_Similarity'].mean():.2f}%
   - Features with KS p-value > 0.05: {sum(s > 0.05 for s in df_stats['KS_Pvalue'])}/{len(df_stats)}
   - Mean KS Distance: {mean_ks:.4f}
   - Mean JS Divergence: {df_stats['JS_Divergence'].mean():.4f}

2. 📊 CORRELATION SIMILARITY:
   - Matrix Similarity: {correlation_similarity:.2f}%
   - Mean Correlation Difference: {mean_corr_diff:.4f}
   - Frobenius Norm: {frobenius_norm:.4f}

3. 🎯 TARGET VARIABLE ({target}):
   - KS Similarity: {df_stats[df_stats['Feature'] == target]['KS_Similarity'].values[0]:.2f}%
   - JS Divergence: {df_stats[df_stats['Feature'] == target]['JS_Divergence'].values[0]:.4f}
   - Real Mean: {df_real_clean[target].mean():.3f}
   - Synthetic Mean: {df_synth_sample[target].mean():.3f}
   - Real Std: {df_real_clean[target].std():.3f}
   - Synthetic Std: {df_synth_sample[target].std():.3f}
   - Mean Difference: {df_real_clean[target].mean() - df_synth_sample[target].mean():.3f}
```

```
4. 🎯 PREDICTIVE PERFORMANCE:
   - R² (Train Synthetic → Test Real): {r2_str}
   - RMSE: {rmse_str}
   - MAE: {mae_str}
   - Hit Rate (≤10%): {hit_rate_str}
   - Excellent Rate (≤5%): {excellent_rate_str}

5. 🏆 OVERALL SIMILARITY SCORE: {overall_score:.2f}%

6. 📌 CLASSIFICATION: {classification}

7. 🏆 TOP 5 BEST MATCHING FEATURES:
""")

top_10 = df_stats.nlargest(10, 'KS_Similarity')
for _, row in top_10.head(5).iterrows():
    print(f"     • {row['Feature']}: {row['KS_Similarity']:.1f}%")

print("\n   📉 TOP 5 WORST MATCHING FEATURES:")
bottom_10 = df_stats.nsmallest(10, 'KS_Similarity')
for _, row in bottom_10.head(5).iterrows():
    print(f"     • {row['Feature']}: {row['KS_Similarity']:.1f}%")

print("""
💡 RECOMMENDATIONS:
   - The synthetic dataset successfully preserves statistical properties
   - Use for validation and model testing
   - Consider ensemble methods for improved predictive performance
""")

print("="*80)
print("✅ ANALYSIS COMPLETE!")
print("="*80)

# ============================================================================
# 13. SAVE RESULTS
# ============================================================================

output_dir = os.path.dirname(PATH_REAL)
os.makedirs(output_dir, exist_ok=True)

# Save statistics
stats_filename = f"{os.path.splitext(os.path.basename(PATH_REAL))[0]}_vs_synthetic_stats.csv"
df_stats.to_csv(os.path.join(output_dir, stats_filename), index=False)
print(f"\n✅ Complete statistics saved: {os.path.join(output_dir, stats_filename)}")

# Save summary
summary_df = pd.DataFrame({
    'Metric': ['KS Similarity Score (%)', 'JS Similarity Score (%)', 'Correlation Similarity (%)',
               'Mean KS Distance', 'Mean JS Divergence', 'Mean Correlation Difference',
               'Overall Similarity Score (%)', 'R² (Synthetic→Real)', 'RMSE', 'MAE',
               'Hit Rate (≤10%)', 'Excellent Rate (≤5%)',
               'Real Mean', 'Synthetic Mean', 'Real Std', 'Synthetic Std'],
    'Value': [f"{ks_similarity:.2f}%", f"{df_stats['JS_Similarity'].mean():.2f}%",
f"{correlation_similarity:.2f}%",
              f"{mean_ks:.4f}", f"{df_stats['JS_Divergence'].mean():.4f}", f"{mean_corr_diff:.4f}",
              f"{overall_score:.2f}%", r2_str, rmse_str, mae_str,
              hit_rate_str, excellent_rate_str,
              f"{df_real_clean[target].mean():.3f}",
              f"{df_synth_sample[target].mean():.3f}",
              f"{df_real_clean[target].std():.3f}",
```

```
            f"{df_synth_sample[target].std():.3f}"]
})

summary_filename = f"{os.path.splitext(os.path.basename(PATH_REAL))[0]}_vs_synthetic_summary.csv"
summary_df.to_csv(os.path.join(output_dir, summary_filename), index=False)
print(f"✅ Summary saved: {os.path.join(output_dir, summary_filename)}")

# ============================================================================
# 14. EXECUTION SUMMARY
# ============================================================================

print("\n" + "="*80)
print("📊 EXECUTION SUMMARY")
print("="*80)
print(f"""
✅ Analysis completed successfully!

📁 Input files:
   Real: {PATH_REAL}
   Synthetic: {PATH_SYNTH}

📊 Dataset info:
   Real samples: {len(df_real_clean)}
   Synthetic samples: {len(df_synth_sample)}
   Total variables: {len(common_cols)}
   Target variable: {target}

📈 Results:
   KS Similarity: {ks_similarity:.2f}%
   Correlation Similarity: {correlation_similarity:.2f}%
   Overall Score: {overall_score:.2f}%
   Classification: {classification}

📁 Output files saved to: {output_dir}
   - {stats_filename}
   - {summary_filename}

🔧 To analyze another dataset, modify PATH_REAL and PATH_SYNTH
   at the beginning of the script!
""")
```

APPENDIX D - PREDICTIVE PERFORMANCE VALIDATION AND CONFUSION MATRIX, CLASSIFICATION REPORT, FEATURE IMPORTANCE AND SHAP (SCRIPT 04)

```
# ============================================================================
# @title PREDICTIVE PERFORMANCE VALIDATION AND FEATURE IMPORTANCE COMPARISON (Script 04)
# WITH CONFUSION MATRIX, CLASSIFICATION REPORT, FEATURE IMPORTANCE AND SHAP
# ============================================================================


# --- STEP 0: IMPORT LIBRARIES ---
# Import necessary libraries for machine learning and visualization
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import (accuracy_score, classification_report, confusion_matrix,
                             cohen_kappa_score, f1_score)
import shap
import warnings
warnings.filterwarnings('ignore')

print("="*80)
print("📊 ORDINAL CLASSIFICATION: COFFEE QUALITY PREDICTION")
print("   Synthetic Data Training → Real Data Testing")
print("="*80)

# ============================================================================
# 1. DATA LOADING
# ============================================================================
# Load both synthetic and real datasets for comparison

print("\n📂 Loading datasets...")

# Define file paths for synthetic and real data
PATH_SYNTH = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova4/
synthetic_arabica_coffee.xlsx'
PATH_REAL = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova4/
df_arabica_complete.xlsx'

# Load the datasets from Excel files
df_synth = pd.read_excel(PATH_SYNTH)
df_real = pd.read_excel(PATH_REAL)

# Display dataset dimensions
print(f"✅ Synthetic dataset: {df_synth.shape[0]} samples x {df_synth.shape[1]} columns")
print(f"✅ Real dataset: {df_real.shape[0]} samples x {df_real.shape[1]} columns")

# ============================================================================
# 2. DEFINE FEATURES AND TARGET - USING ALL AVAILABLE COLUMNS
# ============================================================================
# Identify which columns will be used as features and which is the target variable

print("\n🔍 Defining features and target...")
```

```
# Define the target variable (what we want to predict)
target = 'Total Cup Points'

# List of columns to exclude from features (identifiers and target)
cols_to_remove = ['ID', 'Unnamed: 0', 'Unnamed: 0.1', 'index', 'Id', 'id', target]

# Identify common numeric columns between both datasets to use as features
features = []
for col in df_real.columns:
    # Check if column exists in both datasets and is not in exclusion list
    if col not in cols_to_remove and col in df_synth.columns:
        try:
            # Verify both columns are numeric
            if pd.api.types.is_numeric_dtype(df_real[col]) and
pd.api.types.is_numeric_dtype(df_synth[col]):
                features.append(col)
        except:
            pass

print(f"   Total features available: {len(features)}")
print(f"   Features: {features}")

# ============================================================================
# 3. TRANSFORMATION INTO ORDINAL CLASSES
# ============================================================================
# Convert continuous scores into ordinal classes based on quartiles
# This creates a 4-class ordinal classification problem

print("\n📊 Creating ordinal classes based on sensory score...")

# Define class names for better interpretation
CLASS_NAMES = ['Standard', 'Good', 'Excellent', 'Specialty']

def categorize_quality(df, target_col):
    """
    Categorize continuous scores into 4 ordinal classes using quartiles.
    The categories represent different quality levels:
    - Class 0: Standard (lowest quartile)
    - Class 1: Good (second quartile)
    - Class 2: Excellent (third quartile)
    - Class 3: Specialty (highest quartile)
    """
    # Calculate quartile thresholds based on real data distribution
    quantiles = df_real[target_col].quantile([0.25, 0.50, 0.75]).values

    def apply_rule_quantile(val):
        """Apply classification rule based on quartile thresholds"""
        if val <= quantiles[0]:
            return 0  # Standard
        elif val <= quantiles[1]:
            return 1  # Good
        elif val <= quantiles[2]:
            return 2  # Excellent
        else:
            return 3  # Specialty

    return df[target_col].apply(apply_rule_quantile)

# Apply categorization to both datasets
df_real['Quality_Class'] = categorize_quality(df_real, target)
df_synth['Quality_Class'] = categorize_quality(df_synth, target)
```

```
# Display class distribution for both datasets
print(f"\nReal data class distribution:")
print(df_real['Quality_Class'].value_counts().sort_index())

print(f"\nSynthetic data class distribution:")
print(df_synth['Quality_Class'].value_counts().sort_index())

# ============================================================================
# 4. PREDICTOR SELECTION
# ============================================================================
# Prepare feature matrices and target vectors for modeling

print("\n🎯 Preparing features for modeling...")

# For Real Data: fill missing values with median (robust to outliers)
X_real = df_real[features].fillna(df_real[features].median())
y_real = df_real['Quality_Class']

# For Synthetic Data: fill missing values with median
X_synth = df_synth[features].fillna(df_synth[features].median())
y_synth = df_synth['Quality_Class']

# Confirm data shapes
print(f"Real data: X={X_real.shape}, y={y_real.shape}")
print(f"Synthetic data: X={X_synth.shape}, y={y_synth.shape}")

# ============================================================================
# 5. MODELING (Train on Synthetic -> Test on Real)
# ============================================================================
# This is the key experiment: train on synthetic data and test on real data
# to evaluate how well the synthetic data captures real-world patterns

print("\n" + "="*80)
print("🤖 TRAINING MODEL ON SYNTHETIC DATA")
print("="*80)

# Initialize Random Forest Classifier with 100 trees
# This ensemble method is robust and handles non-linear relationships well
clf = RandomForestClassifier(n_estimators=100, random_state=42, n_jobs=-1)

# Train the model exclusively on synthetic data
clf.fit(X_synth, y_synth)

print("✅ Model trained on synthetic data")

# Make predictions on real data
y_pred = clf.predict(X_real)

print("\n" + "="*80)
print("📊 EVALUATION: Synthetic Train → Real Test")
print("="*80)

# ============================================================================
# 6. METRICS
# ============================================================================
# Calculate various performance metrics for comprehensive evaluation

# Accuracy: overall correctness of predictions
accuracy = accuracy_score(y_real, y_pred)

# F1-Score (Macro): harmonic mean of precision and recall, averaged across classes
# Macro averaging gives equal weight to all classes
```

```
f1_macro = f1_score(y_real, y_pred, average='macro')

# Quadratic Weighted Kappa: measures agreement between predicted and actual ordinal classes
# Ranges from -1 (complete disagreement) to 1 (perfect agreement)
# Higher weight is given to larger disagreements
qwk = cohen_kappa_score(y_real, y_pred, weights='quadratic')

print(f"\nOverall Accuracy: {accuracy:.4f}")
print(f"F1-Score (Macro): {f1_macro:.4f}")
print(f"Quadratic Weighted Kappa: {qwk:.4f}")

# Detailed classification report with per-class metrics
print("\nClassification Report:")
print(classification_report(y_real, y_pred, target_names=CLASS_NAMES))

# ============================================================================
# 7. CONFUSION MATRIX
# ============================================================================
# Visualize the confusion matrix to understand prediction patterns
# Shows which classes are often confused with each other

print("\n📊 Confusion Matrix...")

# Calculate confusion matrix
cm = confusion_matrix(y_real, y_pred)

# Create heatmap visualization
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='YlGnBu',
            xticklabels=CLASS_NAMES,
            yticklabels=CLASS_NAMES,
            cbar_kws={'label': 'Count'})
plt.title("Confusion Matrix: Ground Truth vs Synthetic Prediction", fontsize=14)
plt.ylabel('Real Data (Actual)', fontsize=12)
plt.xlabel('Predicted (Synthetic Model)', fontsize=12)
plt.tight_layout()
plt.show()

# ============================================================================
# 8. SHAP ANALYSIS - REAL DATA AND SYNTHETIC DATA
# ============================================================================
# SHAP (SHapley Additive exPlanations) provides interpretable feature importance
# It shows how each feature contributes to predictions for individual samples

print("\n" + "="*80)
print("📊 SHAP ANALYSIS")
print(f"   Using ALL {len(features)} features")
print("="*80)

# Create SHAP explainer using TreeExplainer (optimized for tree-based models)
explainer = shap.TreeExplainer(clf)

# ============================================================================
# 8.0 SHAP VALUES FOR REAL DATA
# ============================================================================
# Calculate SHAP values for real data to understand feature contributions

print("\n📊 Calculating SHAP values for Real Data...")
shap_values_real = explainer.shap_values(X_real)

# ============================================================================
# 8.1 SHAP VALUES FOR SYNTHETIC DATA
```

```
# ============================================================================
# Calculate SHAP values for a sample of synthetic data
# Using sample to speed up computation while maintaining representative results

print("\n📊 Calculating SHAP values for Synthetic Data...")
X_synth_sample = X_synth.sample(n=min(200, len(X_synth)), random_state=42)
shap_values_synth = explainer.shap_values(X_synth_sample)

# ============================================================================
# 8.2 DETECT SHAP VERSION AND ADJUST FORMAT
# ============================================================================
# SHAP library has different output formats between versions
# This function handles both old and new SHAP formats transparently

print("\n🔍 Detecting SHAP format...")

def process_shap_values(shap_values, X_data, name):
    """
    Process SHAP values in a way compatible with both old and new versions.

    New format: numpy array (n_samples, n_features, n_classes)
    Old format: list of arrays (one per class)

    Returns:
    - shap_values_formatted: original or processed values
    - shap_values_combined: combined across classes for summary plots
    - n_classes: number of classes
    """
    print(f"\n   Processing {name}...")
    print(f"   Type: {type(shap_values)}")

    if isinstance(shap_values, np.ndarray):
        # New format: array with shape (n_samples, n_features, n_classes)
        print(f"   Format: Array (n_samples, n_features, n_classes)")
        print(f"   Shape: {shap_values.shape}")

        n_samples, n_features, n_classes = shap_values.shape
        print(f"   Samples: {n_samples}, Features: {n_features}, Classes: {n_classes}")

        # Combine classes by averaging absolute values for summary plots
        shap_values_combined = np.mean(np.abs(shap_values), axis=2)
        print(f"   Combined shape: {shap_values_combined.shape}")

        return shap_values, shap_values_combined, n_classes

    else:
        # Old format: list of arrays, one per class
        print(f"   Format: List of arrays (one per class)")
        print(f"   Number of classes: {len(shap_values)}")

        # Combine classes by averaging absolute values
        shap_values_combined = np.mean(np.abs(shap_values), axis=0)
        print(f"   Combined shape: {shap_values_combined.shape}")

        return shap_values, shap_values_combined, len(shap_values)

# Process SHAP values for both datasets
print("\n" + "-"*40)
shap_values_real_formatted, shap_values_real_combined, n_classes = process_shap_values(
    shap_values_real, X_real, "Real Data"
)
```

```
print("\n" + "-"*40)
shap_values_synth_formatted, shap_values_synth_combined, n_classes_synth = process_shap_values(
    shap_values_synth, X_synth_sample, "Synthetic Data"
)

# Extract expected values (baseline predictions)
expected_values_real = explainer.expected_value
if isinstance(expected_values_real, np.ndarray) and len(expected_values_real) > 1:
    print(f"\n   Expected values (Real): {expected_values_real}")

# Determine maximum number of features to display in plots
max_display = min(20, len(features))

# ================================================================================
# 8a. BAR PLOT - REAL DATA (WITH COLORS)
# ================================================================================
# Bar plot showing average absolute SHAP values for real data
# This indicates feature importance across all classes

print("\n📊 SHAP Bar Plot - Real Data (All Features)...")

plt.figure(figsize=(12, 8))
if isinstance(shap_values_real_formatted, np.ndarray):
    shap.summary_plot(shap_values_real_formatted, X_real, plot_type="bar",
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
else:
    shap.summary_plot(shap_values_real_formatted, X_real, plot_type="bar",
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
plt.title(f'SHAP Feature Importance - REAL Data (All {len(features)} Features)',
          fontsize=14, fontweight='bold')
plt.tight_layout()
plt.show()

# ================================================================================
# 8b. BAR PLOT - SYNTHETIC DATA (WITH COLORS)
# ================================================================================
# Bar plot showing average absolute SHAP values for synthetic data
# Compare with real data to check if importance patterns are similar

print("\n📊 SHAP Bar Plot - Synthetic Data (All Features)...")

plt.figure(figsize=(12, 8))
if isinstance(shap_values_synth_formatted, np.ndarray):
    shap.summary_plot(shap_values_synth_formatted, X_synth_sample, plot_type="bar",
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
else:
    shap.summary_plot(shap_values_synth_formatted, X_synth_sample, plot_type="bar",
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
plt.title(f'SHAP Feature Importance - SYNTHETIC Data (All {len(features)} Features)',
          fontsize=14, fontweight='bold')
plt.tight_layout()
plt.show()

# ================================================================================
# 8c. BEESWARM PLOT - REAL DATA (WITH COLORS)
# ================================================================================
# Beeswarm plot shows the distribution of SHAP values for each feature
# Each point represents a sample, color indicates feature value (high/low)
# Shows both direction and magnitude of feature effects

print("\n📊 SHAP Beeswarm Plot - Real Data (All Features)...")
```

```
plt.figure(figsize=(12, 10))
if isinstance(shap_values_real_formatted, np.ndarray):
    shap.summary_plot(shap_values_real_formatted, X_real,
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
else:
    shap.summary_plot(shap_values_real_formatted, X_real,
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
plt.title(f'SHAP Value Distribution - REAL Data (All {len(features)} Features)',
          fontsize=14, fontweight='bold')
plt.tight_layout()
plt.show()

# ============================================================================
# 8d. BEESWARM PLOT - SYNTHETIC DATA (WITH COLORS)
# ============================================================================
# Beeswarm plot for synthetic data to compare with real data patterns

print("\n📊 SHAP Beeswarm Plot - Synthetic Data (All Features)...")

plt.figure(figsize=(12, 10))
if isinstance(shap_values_synth_formatted, np.ndarray):
    shap.summary_plot(shap_values_synth_formatted, X_synth_sample,
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
else:
    shap.summary_plot(shap_values_synth_formatted, X_synth_sample,
                      class_names=CLASS_NAMES, show=False, max_display=max_display)
plt.title(f'SHAP Value Distribution - SYNTHETIC Data (All {len(features)} Features)',
          fontsize=14, fontweight='bold')
plt.tight_layout()
plt.show()

# ============================================================================
# 8e. WATERFALL PLOTS - REAL DATA
# ============================================================================
# Waterfall plots show how each feature contributes to a specific prediction
# Starting from base value, each feature adds or subtracts from the prediction
# This provides detailed interpretation for individual samples

print("\n📊 Waterfall Plots - Real Data Samples...")

# Select one representative sample from each class
sample_indices = []
for class_id in range(n_classes):
    indices = np.where(y_real == class_id)[0]
    if len(indices) > 0:
        sample_indices.append(indices[0])

# Fallback if some classes are empty
if not sample_indices:
    sample_indices = [0]

# Generate waterfall plots for each selected sample
for idx in sample_indices[:4]:  # Limit to 4 plots
    class_id = y_real.iloc[idx]
    class_name = CLASS_NAMES[class_id]

    # Extract SHAP values for the specific class
    if isinstance(shap_values_real_formatted, np.ndarray):
        shap_values_class = shap_values_real_formatted[idx, :, class_id]
    else:
        shap_values_class = shap_values_real_formatted[class_id][idx]
```

```
    # Extract expected value for the class
    if isinstance(expected_values_real, np.ndarray) and len(expected_values_real) > 1:
        expected_value = expected_values_real[class_id]
    else:
        expected_value = expected_values_real

    # Create waterfall plot
    plt.figure(figsize=(10, 6))
    shap.waterfall_plot(
        shap.Explanation(
            values=shap_values_class,
            base_values=expected_value,
            data=X_real.iloc[idx].values,
            feature_names=features
        ),
        max_display=10,
        show=False
    )
    plt.title(f'Waterfall Plot - REAL Sample (Class: {class_name})', fontsize=12, fontweight='bold')
    plt.tight_layout()
    plt.show()

# ============================================================================
# 8f. WATERFALL PLOTS - SYNTHETIC DATA
# ============================================================================
# Waterfall plots for synthetic samples to compare individual predictions

print("\n📊 Waterfall Plots - Synthetic Data Samples...")

# Select synthetic samples from each class
y_synth_sample = y_synth.loc[X_synth_sample.index]
sample_indices_synth = []
for class_id in range(n_classes_synth):
    indices = np.where(y_synth_sample == class_id)[0]
    if len(indices) > 0:
        sample_indices_synth.append(indices[0])

if not sample_indices_synth:
    sample_indices_synth = [0]

# Generate waterfall plots for synthetic samples
for idx in sample_indices_synth[:4]:
    class_id = y_synth_sample.iloc[idx]
    class_name = CLASS_NAMES[class_id]

    if isinstance(shap_values_synth_formatted, np.ndarray):
        shap_values_class = shap_values_synth_formatted[idx, :, class_id]
    else:
        shap_values_class = shap_values_synth_formatted[class_id][idx]

    if isinstance(expected_values_real, np.ndarray) and len(expected_values_real) > 1:
        expected_value = expected_values_real[class_id]
    else:
        expected_value = expected_values_real

    plt.figure(figsize=(10, 6))
    shap.waterfall_plot(
        shap.Explanation(
            values=shap_values_class,
            base_values=expected_value,
            data=X_synth_sample.iloc[idx].values,
            feature_names=features
```

```
        ),
        max_display=10,
        show=False
    )
    plt.title(f'Waterfall Plot - SYNTHETIC Sample (Class: {class_name})', fontsize=12,
fontweight='bold')
    plt.tight_layout()
    plt.show()

# ============================================================================
# 8g. SHAP COMPARISON TABLE
# ============================================================================
# Create a comprehensive comparison table of feature importance
# Combines SHAP importance from both datasets with Random Forest importance

print("\n📊 SHAP Comparison Summary...")

# Calculate mean absolute SHAP values for each feature
shap_importance_real = np.mean(np.abs(shap_values_real_combined), axis=0)
shap_importance_synth = np.mean(np.abs(shap_values_synth_combined), axis=0)

# Create comparison dataframe
shap_comparison = pd.DataFrame({
    'Feature': features,
    'SHAP_Importance_Real': shap_importance_real,
    'SHAP_Importance_Synthetic': shap_importance_synth,
    'SHAP_Difference': shap_importance_real - shap_importance_synth,
    'RF_Importance': importances  # Random Forest feature importance
}).sort_values('SHAP_Importance_Real', ascending=False)

print("\nSHAP Importance Comparison (All Features):")
print("-"*80)
print(shap_comparison.to_string(index=False))

# ============================================================================
# 9. FEATURE IMPORTANCE - ALL FEATURES
# ============================================================================
# Display Random Forest feature importance for all features
# This complements SHAP analysis with a different perspective

print("\n" + "="*80)
print("📊 FEATURE IMPORTANCE - ALL FEATURES")
print("="*80)

# Extract feature importances from the trained model
importances = clf.feature_importances_

# Sort features by importance
indices = np.argsort(importances)[::-1]

print(f"\nFeature Importance Ranking (All {len(features)} features):")
print("-"*60)
print(f"{'Rank':<6} {'Feature':<25} {'Importance':<15}")
print("-"*60)

# Display each feature with its importance score
for i, idx in enumerate(indices, 1):
    print(f"{i:<6} {features[idx]:<25} {importances[idx]:<15.6f}")

# ============================================================================
# 10. DISTRIBUTION COMPARISON
# ============================================================================
```

```
# Compare the distribution of the target variable between real and synthetic data
# This helps verify if synthetic data preserves the original distribution

print("\n📊 Distribution Comparison...")

# Create side-by-side comparison plots
fig, axes = plt.subplots(1, 2, figsize=(14, 5))

# Plot 1: Continuous score distribution (histogram)
ax1 = axes[0]
ax1.hist(df_real[target], bins=20, alpha=0.5, label='Real', color='blue', edgecolor='black')
ax1.hist(df_synth[target], bins=20, alpha=0.5, label='Synthetic', color='orange', edgecolor='black')
ax1.set_xlabel('Total Cup Points', fontsize=12)
ax1.set_ylabel('Frequency', fontsize=12)
ax1.set_title('Score Distribution', fontsize=14, fontweight='bold')
ax1.legend(fontsize=11)
ax1.grid(True, alpha=0.3)

# Plot 2: Class distribution (bar chart)
ax2 = axes[1]
real_classes = df_real['Quality_Class'].value_counts().sort_index()
synth_classes = df_synth['Quality_Class'].value_counts().sort_index()

x = np.arange(len(CLASS_NAMES))
width = 0.35  # Bar width
ax2.bar(x - width/2, real_classes.values, width, label='Real', color='blue', alpha=0.7)
ax2.bar(x + width/2, synth_classes.values, width, label='Synthetic', color='orange', alpha=0.7)
ax2.set_xlabel('Quality Class', fontsize=12)
ax2.set_ylabel('Count', fontsize=12)
ax2.set_title('Class Distribution', fontsize=14, fontweight='bold')
ax2.set_xticks(x)
ax2.set_xticklabels(CLASS_NAMES, rotation=15)
ax2.legend(fontsize=11)
ax2.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()

# ============================================================================
# 11. FINAL SUMMARY
# ============================================================================
# Comprehensive summary of all results and key findings

print("\n" + "="*80)
print("📋 FINAL SUMMARY")
print("="*80)

print(f"""
📊 EVALUATION METRICS:

1. 🎯 CLASSIFICATION PERFORMANCE:
   - Accuracy: {accuracy:.4f}
   - F1-Score (Macro): {f1_macro:.4f}
   - Quadratic Weighted Kappa: {qwk:.4f}

2. 📈 TOP 5 FEATURES (Random Forest):
""")

# Display top 5 features from Random Forest
for i, idx in enumerate(indices[:5], 1):
    print(f"   {i}. {features[idx]}: {importances[idx]:.6f}")
```

```
print(f"""
3. 📈 TOP 5 FEATURES (SHAP - Real):
""")

# Display top 5 features from SHAP analysis on real data
for i, row in shap_comparison.head(5).iterrows():
    print(f"   {i+1}. {row['Feature']}: {row['SHAP_Importance_Real']:.6f}")

print(f"""
4. 📈 TOP 5 FEATURES (SHAP - Synthetic):
""")

# Display top 5 features from SHAP analysis on synthetic data
shap_synth_top = shap_comparison.sort_values('SHAP_Importance_Synthetic', ascending=False)
for i, row in shap_synth_top.head(5).iterrows():
    print(f"   {i+1}. {row['Feature']}: {row['SHAP_Importance_Synthetic']:.6f}")

print(f"""
5. 📊 CLASS DISTRIBUTION:
   - Real: {dict(df_real['Quality_Class'].value_counts().sort_index())}
   - Synthetic: {dict(df_synth['Quality_Class'].value_counts().sort_index())}

6. 📈 TOTAL FEATURES: {len(features)}

7. 💡 INTERPRETATION:
   - {'✅ EXCELLENT' if qwk > 0.80 else '✅ GOOD' if qwk > 0.60 else '⚠ MODERATE'} Quadratic Weighted
Kappa ({qwk:.4f})
""")

print("="*80)
print("✅ ANALYSIS COMPLETE!")
print("="*80)

# ============================================================================
# 12. SAVE RESULTS
# ============================================================================
# Save all results to CSV files for further analysis and reporting

# Define output directory
output_dir = '/content/drive/MyDrive/Tese/propostas_de_artigos/proposta_artigo_020/Review/Nova/'
import os
os.makedirs(output_dir, exist_ok=True)

# Save Random Forest feature importance
importance_df = pd.DataFrame({
    'Feature': features,
    'Importance': importances
}).sort_values('Importance', ascending=False)
importance_df.to_csv(f'{output_dir}/coffee_feature_importance_all.csv', index=False)
print(f"\n✅ Feature importance saved: {output_dir}/coffee_feature_importance_all.csv")

# Save SHAP comparison table
shap_comparison.to_csv(f'{output_dir}/coffee_shap_comparison.csv', index=False)
print(f"✅ SHAP comparison saved: {output_dir}/coffee_shap_comparison.csv")

# Save class distribution comparison
class_dist_df = pd.DataFrame({
    'Class': CLASS_NAMES,
    'Real_Count': [real_classes.get(i, 0) for i in range(len(CLASS_NAMES))],
    'Synthetic_Count': [synth_classes.get(i, 0) for i in range(len(CLASS_NAMES))]
})
```

```
class_dist_df.to_csv(f'{output_dir}/coffee_class_distribution.csv', index=False)
print(f"✅ Class distribution saved: {output_dir}/coffee_class_distribution.csv")

# Save predictions for individual samples
predictions_df = pd.DataFrame({
    'Actual': y_real.values,
    'Predicted': y_pred,
    'Correct': y_real.values == y_pred
})
predictions_df.to_csv(f'{output_dir}/coffee_predictions.csv', index=False)
print(f"✅ Predictions saved: {output_dir}/coffee_predictions.csv")

# Save performance metrics summary
metrics_df = pd.DataFrame({
    'Metric': ['Accuracy', 'F1_Macro', 'QWK', 'Total_Features'],
    'Value': [accuracy, f1_macro, qwk, len(features)]
})
metrics_df.to_csv(f'{output_dir}/coffee_metrics.csv', index=False)
print(f"✅ Metrics saved: {output_dir}/coffee_metrics.csv")

print("\n" + "="*80)
print("✅ ANALYSIS COMPLETE!")
print("="*80)
```