TAMYRES PEREIRA

CLUSTERIZAÇÃO INTERVALAR INCREMENTAL BOTTOM-UP A PARTIR DE FLUXOS DE DADOS INTERVALARES

LAVRAS – MG

2020

TAMYRES PEREIRA

CLUSTERIZAÇÃO INTERVALAR INCREMENTAL BOTTOM-UP A PARTIR DE FLUXOS DE DADOS INTERVALARES

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Engenharia de Sistemas e Automação, área de concentração em Inteligência Computacional, para a obtenção do título de Mestre.

Daniel Furtado Leite
Orientador

LAVRAS – MG

2020

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

Pereira, Tamyres
Clusterização Intervalar Incremental Bottom-up a partir de Fluxos de Dados Intervalares / Tamyres Pereira. - 2020.
71 p. : il.

Orientador(a): Daniel Furtado Leite.

Dissertação (mestrado acadêmico)- Universidade Federal de Lavras, 2020.
Bibliografia.

1. Clusterização Incremental. 2. Aprendizado de Máquina. 3. Fluxos de Dados Intervalares. I. Leite, Daniel Furtado. II. Título.

TAMYRES PEREIRA

CLUSTERIZAÇÃO INTERVALAR INCREMENTAL BOTTOM-UP A PARTIR DE FLUXOS DE DADOS INTERVALARES

Dissertação apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Engenharia de Sistemas e Automação, área de concentração em Inteligência Computacional, para a obtenção do título de Mestre.

APROVADA em 20 de Fevereiro de 2020.

Belisário Nina Huallpa	UFLA
Pyramo Pires da Costa Júnior	PUC-MG
Daniel Furtado Leite	UFLA

Daniel Furtado Leite
Orientador

LAVRAS – MG
2020

AGRADECIMENTOS

Agradeço a Deus por me conduzir nessa caminhada.

Aos meus pais Rosana e João Carlos, à minha irmã Bianca, e ao hoje meu esposo Rafael, pelo incentivo e suporte diante das dificuldades.

Aos meus familiares e amigos pelo apoio.

Aos professores pelo conhecimento transmitido.

À coordenação e secretaria do PPGESISA pelo suporte.

Aos meus colegas do PPGESISA pela amizade e apoio mútuo.

E à CAPES pela assistência.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

RESUMO

Esse trabalho propõe um método de clusterização intervalar incremental bottom-up a partir de fluxos de dados intervalares. O método é apoiado por conceitos, definições e ferramentas matemáticas da teoria da computação granular, em particular da álgebra de intervalos. Diferentemente de outros métodos evolutivos de processamento e modelagem de fluxos de dados numéricos, o método proposto lida com fluxos de dados que apresentam incerteza não-estruturada representados por valores intervalares, e também fluxos de dados numéricos como um caso particular. O método proposto é capaz de modelar processos complexos apresentados como um fluxo de dados e sujeitos à mudanças no ambiente. O algoritmo de aprendizado desenvolve a estrutura de um modelo de maneira bottom-up, sem conhecimento anterior a respeito do processo, e adapta os parâmetros deste modelo à medida que há necessidade, evitando assim, que o modelo seja reconstruído e retreinado quando há mudança no ambiente ou no sistema – sendo esta uma vantagem clara com relação a modelos pré-concebidos a partir de conhecimento especialista ou dados históricos. Para o desenvolvimento de grânulos (modelos locais), o algoritmo de aprendizado é equipado com fórmulas recursivas para cálculo de similaridade entre objetos intervalares, e com o índice de validação incremental Xie-Beni.

Palavras-chave: Clusterização Incremental. Aprendizado de Máquina. Fluxos de Dados. Matemática Intervalar. Computação Granular.

ABSTRACT

This work proposes a method of bottom-up incremental interval clustering from interval data streams. The method is supported by concepts, definitions and mathematical tools of the granular computation theory, in particular interval algebra. Differently from other evolutionary methods of processing and modeling numerical data flows, the proposed method deals with data streams that exhibits unstructured uncertainty represented by interval values, and also numerical data streams as a particular case. The proposed method is able to model complex processes presented as a data stream and subject to changes in the environment. The learning algorithm develop the structure of the model in a bottom-up manner, without prior knowledge about of the process, and adapts the parameters of the model as needed, thus avoiding that the model be reconstructed and retrained when there is a change in the environment or system - this being a clear advantage over pre-designed models based on specialized knowledge or historical data. For the development of granules (local models), the learning algorithm is equipped with recursive formulas to calculate the similarity between interval objects and with the Xie-Beni incremental validation index.

Keywords: Incremental Clustering. Machine Learning. Data Streams. Interval Mathematics. Granular Computing.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Sistemas físicos e virtuais estão sujeitos à dinâmicas não-lineares e não-estacionárias, ou seja, existem relações complexas e muitas vezes desconhecidas entre as variáveis envolvidas. Além disso, as características inerentes ao sistema podem variar ao longo do tempo tal que cálculos estatísticos *a priori*, e.g. média, variância, correlação, são válidos apenas temporariamente. A adaptação de modelos ao longo do tempo a partir de métodos de aprendizado recursivo e de uma sequência de amostras de dados se torna essencial nesses casos.

A capacidade de adaptação e aprendizado são comuns aos seres humanos. O aprendizado de máquina incremental para desenvolvimento de modelos computacionais adaptativos tem sido investigado ao longo das últimas duas décadas (KASABOV, 2007) (LUGHOFER, 2011) (LEITE, 2012) (ANGELOV, 2013) (SKRJANC et al., 2019). A principal inspiração para o desenvolvimento de mecanismos de aprendizado de máquina são as características de redes neurais biológicas e a capacidade do cérebro humano em lidar com incertezas e grânulos de informação (PEDRYCZ; HOMENDA, 2013) (YAO; VASILAKOS; PEDRYCZ, 2013).

Modelos computacionais visam auxiliar e dar suporte à tomada de decisões e a efetivamente prover valores de saída que atuam no mundo físico. Modelos forçam a explicitação dos objetivos; forçam a identificação das interações entre variáveis; permitem raciocínio criterioso sobre as variáveis; identificam as limitações dos valores possíveis; e podem ser melhorados com a experiência e a história. Eles podem ser usados para tarefas como reconhecimento de padrões, classificação de dados, previsão de séries temporais, controle de sistemas dinâmicos, aproximação de funções, entre outros (LEITE, 2012). Alguns exemplos de aplicações práticas seriam pesquisa e segmentação de mercado para determinar potenciais grupos homogêneos de consumidores para melhor definir a disposição de produtos como uma estratégia corporativa; reconhecimento de comunidades que compartilhem alguma preferência ou opinião em redes sociais; identificação de períodos de alta e baixa volatilidade do mercado financeiro para subsidiar a construção de estratégias de *trading* (processos de negociação no mercado, tais como a compra e venda de ações, títulos, moedas, entre outros) e de gestão de risco.

Dentre as linhas atuais de pesquisa em aprendizado de máquina incremental destacam-se métodos recursivos de mineiração de dados (DUARTE; GAMA; BIFET, 2016) (KOURTELLIS et al., 2016), extensões da teoria do aprendizado estatístico (PARISIEN; FAZLY; STEVENSON, 2008) (FRIEDMAN; HASTIE; TIBSHIRANI, 2010), aprendizado profundo (SUN et al., 2014) (BODYANSKIY et al., 2016), computação granular (CORDOVIL et al., 2019) (LEITE

et al., 2019), e inteligência computacional (ANGELOV; ZHOU, 2008a) (WEN et al., 2015). Ambientes de computação paralela têm sido desenvolvidos para processamento de grandes volumes de dados disponibilizados gradualmente (BIFET et al., 2010) (SHVACHKO et al., 2010) (MORALES; BIFET, 2015).

A inteligência computacional inclui sistemas evolucionários, adaptativos e evolutivos como linhas de pesquisa. Sistemas evolucionários são inspirados na evolução biológica. Eles fazem uso de operadores de seleção, cruzamento e mutação para a evolução de uma população de indivíduos para um melhor desempenho na resolução de um determinado problema (ZHAO et al., 2009) (ZHOU et al., 2011) (FADAEE; RADZI, 2012). Sistemas inteligentes adaptativos geralmente adaptam apenas os parâmetros de um modelo. Não há adaptação estrutural (WANG; HUANG, 2010) (LIANG et al., 2012) (XUE et al., 2018). A estrutura do modelo deve ser escolhida antes ao aprendizado de máquina.

Sistemas inteligentes evolutivos trata-se de uma linha de pesquisa em inteligência computacional que faz uso de modelos neurais, fuzzy e neuro-fuzzy munidos com algoritmos de aprendizado incrementais que buscam extrair informações significativas de fluxos de dados (LEITE, 2012) (SILVA et al., 2014) (MOSHTAGHI; LECKIE; BEZDEK, 2016). Estes sistemas lidam com os requerimentos de aplicações em tempo real a partir do uso de procedimentos recursivos rápidos e escaláveis (linearmente ou polinomialmente) de acordo com o volume de dados (LEITE, 2012). Adicionalmente, modelos ditos evolutivos apresentam a característica de auto-adaptação estrutural, além da adaptação paramétrica convencional. Isto os tornam aptos a lidar com não-linearidades fortes e diferentes tipos de não-estacionariedades (ANGELOV; FILEV; KASABOV, 2010) (RUBIO; PÉREZ-CRUZ, 2014) (GU et al., 2015).

Modelos evolutivos devem (MOSHTAGHI et al., 2015) (LUGHOFER; PRATAMA; SKRJANC, 2018) (AL-HMOUZ et al., 2018) (SKRJANC et al., 2019):

- incorporar novas informações provenientes do fluxo de dados;
- preservar as informações adquiridas anteriormente;
- identificar novos comportamentos ou classes em dados nunca vistos antes;
- acompanhar variações de conceito, i.e., variações de distribuições de probabilidade.

Algoritmos evolutivos, no sentido de processamento online de fluxos de dados, lidam com amostras de até aproximadamente cinquenta atributos em um computador comercial convencional em cerca de dez mili-segundos (LUGHOFER, 2008) (LEITE et al., 2012) (LEITE;

COSTA; GOMIDE, 2013a) (LEITE et al., 2014) (KHUAT; CHEN; GABRYS, 2019). Para fluxos de dados mais rápidos e com mais atributos tem-se considerado técnicas de granulação do tempo, processamento paralelo e super-computadores (LEITE, 2012) (BIFET; GAVALDÀ, 2009).

O fluxo de dados submetido ao processamento do algoritmo evolutivo é constituído por uma sequência ordenada de vetores, os quais são compostos por elementos (atributos) reais ou intervalares ou outros. Enquanto a grande maioria dos trabalhos ficam em valores reais, o presente trabalho considera dados intervalares.

A Tabela 1.1 apresenta um exemplo de base de dados intervalar. Cada linha da tabela refere-se à um paciente e as colunas listam medidas referentes ao coração daquele paciente. Cada linha da tabela é uma amostra, e cada coluna é um atributo.

Tabela 1.1 – Exemplo de base de dados intervalar da saúde cardíaca de pacientes

	Taxa de pulso	**Pressão sistólica**	**Pressão diastólica**
1	[60,72]	[90,130]	[70,90]
2	[70,112]	[110,142]	[80,108]
3	[54,72]	[90,100]	[50,70]
4	[70,100]	[130,160]	[80,110]
5	[63,75]	[60,100]	[140,150]
6	[44,68]	[90,100]	[50,70]

Fonte: adaptado de Jajuga, Sokolowski e Bock (2012)

O coração trabalha como uma bomba. Ele joga o sangue para frente quando se contrai (sístole), o que causa o esvaziamento de sangue, e enchimento das artérias. Do outro lado do coração, o sangue volta pelas veias, fazendo com que haja novo enchimento do coração, que até então encontrava-se relaxado (diástole). Este movimento de vai e vem, sem parar, é que nos mantém vivos e exerce uma pressão na contração, denominada sistólica ou máxima, e outra no enchimento do coração relaxado, chamada de diastólica ou mínima (SOCIEDADE BRASILEIRA DE CARDIOLOGIA, 2018). A pressão arterial pode variar durante o dia, por isso as medidas referentes ao coração de cada paciente são expressas como intervalos.

O conceito de aprendizado bottom-up e top-down vem sendo utilizado em sistemas inteligentes (SENGE; HÜLLERMEIER, 2011) (DENG et al., 2016) (CHEN et al., 2018) – embora nem sempre o termo seja usado explicitamente. Ele se refere ao modo como acontece o desenvolvimento de um modelo, usualmente a partir de um conjunto de dados. Em comum, a

ideia é organizar as informações extraídas dos dados, proporcionando processamento eficiente, memorização da essência dos dados, e garantindo um sistema com prioridades e hierarquia.

Bottom-up é uma abordagem de aprendizado em que modelos locais (partições do espaço dos dados) são incialmente tão detalhados quanto os dados. Um modelo local é, de fato, uma amostra de dados. Os modelos se expandem conforme amostras de dados semelhantes, i.e. relativamente próximas segundo uma métrica de similaridade ou distância, são apresentadas. Já o aprendizado top-down inicia-se a partir de um modelo abstrato, rudimentar, único, que se divide em fragmentos até que uma granularidade adequada seja encontrada, i.e., até que a essência dos dados tenha sido capturada de maneira concisa e explicável, sendo subconjuntos de um conjunto universal e que transmite uma representação interna (LEITE, 2012). Esses conceitos são empregados em modelagem em diversas áreas, e.g. na indústria, ciências humanas, econometria, física, neurologia, gestão e, de uma maneira teórica e mais formal, em ciência da computação com o apoio de teorias matemáticas (CASTELLA et al., 2007) (KLAPPER et al., 2014) (WANG et al., 2015).

A computação granular (PEDRYCZ; GOMIDE, 2007) (LEITE, 2012) se vale de teorias e metodologias que desenvolve a granularidade de modelos a partir de dados. Uma granularidade apropriada é encontrada por meio de algoritmos de aprendizado incrementais visando a resolução de problemas complexos a partir da resolução de problemas mais simples e da união ou agregação das soluções locais. Em particular, granularidade é a extensão em que um sistema maior e mais complexo é dividido em partes menores e mais facilmente tratáveis. Estas partes menores são denominadas grânulos, i.e., agrupamentos de dados numéricos ou incertos (informalmente, grânulos relativamente menores, porém não pontuais) que são considerados indistinguíveis de acordo com um medida de similaridade ou proximidade (LEITE, 2012). Portanto no método proposto nesse trabalho, cada hiper-caixa ou modelo local é um grânulo.

1.1 Objetivo

Esse trabalho propõe um método de clusterização intervalar incremental bottom-up a partir de fluxos de dados intervalares. O método é apoiado por conceitos, definições e ferramentas matemáticas de computação granular em geral, e em particular da teoria granular intervalar no sentido de incerteza completa sobre valores internos. Diferentemente de outros métodos evolutivos de processamento e modelagem de fluxos de dados numéricos, o método proposto lida com fluxos de dados que apresentam incerteza não-estruturada. Eles são representados por

valores intervalares – sendo fluxos de dados numéricos um caso particular. Para o desenvolvimento de grânulos (modelos locais), o algoritmo de aprendizado é equipado com fórmulas recursivas para cálculo de similaridade entre objetos intervalares, e com o índice de validação incremental Xie-Beni. Considera-se uma variedade de bases de dados *benchmark* no contexto de classificação de padrões para avaliação empírica de desempenho de modelos classificadores.

1.2 Contribuições

O algoritmo de clusterização incremental proposto foca sistemas do mundo real cujas equações determinísticas são desconhecidas. Esses sistemas são percebidos a partir de fluxos de dados numéricos e/ou intervalares, e estão sujeitos à mudanças e incerteza. O algoritmo de aprendizado proposto desenvolve a estrutura de um modelo de maneira bottom-up, sem conhecimento anterior a respeito do processo ou fenômeno físico, e adapta os parâmetros deste modelo conforme a necessidade. O modelo não precisa ser reconstruído e retreinado quando há mudança no sistema ou no ambiente – sendo esta uma vantagem clara com relação a modelos pré-concebidos a partir de conhecimento especialista ou dados históricos.

O método proposto nesse trabalho lida com fluxos de dados que apresentam incerteza não-estruturada, diferentemente da grande maioria dos métodos evolutivos encontrados na literatura. Nesse caso, assume-se um intervalo de valores possíveis em torno dos valores numéricos reais. Não se assume qualquer estrutura estatística dentro dos limites do intervalo. Sabe-se somente que o valor correto está incluso no intervalo e trabalha-se com as bordas em espaços multi-dimensionais. Incertezas nos dados são comuns. Elas podem surgir em situações de limitação dos equipamentos, faixa de erro de sensores, transformação de informação qualitativa em quantitativa, e até propositalmente, para a preservação da privacidade na comunicação de dados e sistemas de defesa (AGRAWAL; SRIKANT, 2000) (LEITE, 2012). A incerteza surge também em pré-processamento de dados experimentais, como na extração de atributos importantes a partir de atributos mensuráveis usando equações algébricas ou diferenciais aproximadas, tão bem como métodos estatísticos de previsão ou imputação de valores faltantes (GARCIA; LEITE; SKRJANC, 2019). Há imprecisões naturais associadas à própria maneira com que os dados foram coletados, questões temporais, condições ambientais, condução de experimentos por diferentes pessoas, entre outros.

O algoritmo de aprendizado iUP proposto apresenta as seguintes flexibilidades particulares:

- o parâmetro número de rótulos de classe, λ, oferece flexibilidade e um mecanismo de expressão de conhecimento especialista sobre o domínio do problema, uma vez que ele pode ser escolhido ou não;
- a indicação explícita do número de grânulos que uma base de dados contém é desnecessária no ambiente do algoritmo. O parâmetro ξ-vizinhança máxima dos grânulos, ξ_{max}, mantém um certo controle sobre a quantidade de grânulos que são mantidos em um modelo;
- lida com dados incertos e, em particular, dados numéricos;
- emprega um método baseado na vizinhança dos grânulos para rotulação (atribuição de classes) de grânulos;
- o índice de validação incremental de Xie-Beni proposto, provê uma medida do desempenho do algoritmo de aprendizado e das partições do modelo iUP proposto.

O algoritmo de clusterização intervalar incremental bottom-up para construção de modelos classificadores é de propósito geral e encontra aplicações nas mais diversas áreas do conhecimento, e.g. biologia, medicina, economia, engenharias, computação, agricultura, neurociência, entre outras. Os modelos obtidos também podem ser usados integrados à modelos preditores de séries temporais não-estacionárias, aproximadores de funções não-lineares, e controladores de sistemas dinâmicos.

1.3 Estrutura do Trabalho

O restante desse trabalho está organizado como segue. O Capítulo 2 aborda aprendizado de máquina e classificação incremental de fluxos de dados. Ele apresenta uma visão geral dos trabalhos relacionados à presente proposta, e o estado da arte. Conceitos relacionados a fluxo de dados numéricos e incertos, incerteza não-estruturada e clusterização crisp e fuzzy são discutidos. No Capítulo 3, o método proposto é apresentado. O Capítulo 4 apresenta o esquema geral do método e as bases de dados consideradas para avaliação empírica de modelos. No Capítulo 5 encontra-se os resultados e as discussões. O Capítulo 6 apresenta a conclusão do trabalho e proposta de trabalhos futuros.

2 MODELAGEM DE FLUXOS DE DADOS

2.1 Aprendizado de Máquina e Classificação Incremental

Os seres humanos granulam, agrupam e classificam dados e informações de forma natural e muitas vezes inconsciente no dia a dia. Desta forma, podem resolver problemas imediatos por aproximação e generalizar comportamentos para novas situações e ocasiões por comparação e similaridade entre domínios. Essa capacidade é importante já que permite a tomada de decisão geralmente rápida em situações novas e difíceis (PEDRYCZ, 1997).

A habilidade humana em aprender a partir de exemplos e raciocinar aproximadamente a partir do conhecimento visando tomar decisões coerentes é uma inspiração para o desenvolvimento de sistemas computacionais inteligentes. A ideia é que esses sistemas inteligentes possam proporcionar suporte à humanos na realização de suas tarefas ou mesmo atuar no mundo físico muitas vezes de maneira mais precisa e eficiente que humanos. Sistemas dinâmicos reais têm se tornado mais complexos. Além disso, o volume de dados produzidos e compartilhados por diferentes tecnologias tem aumentado. Consequentemente, sistemas computacionais inteligentes têm se tornado ferramentas essenciais em determinados contextos.

Os sistemas computacionais inteligentes são construídos a partir de algoritmos de aprendizado de máquina. A teoria do aprendizado de máquina tem por objetivo codificar mecanismos indutivos de forma que as soluções para uma variedade de problemas possam ser derivadas de exemplos, ao invés de se preocupar em gerar solução para problemas específicos através da codificação de instruções explícitas. A abordagem de aprendizado definida em aprendizado de máquina é mais flexível do que a abordagem de programação. Ela faz com que os computadores possam colaborar com a solução de problemas mais gerais (PEDRYCZ, 1997).

No campo do aprendizado de máquina existem dois paradigmas fundamentais: aprendizado supervisionado e não-supervisionado. O aprendizado supervisionado considera que as amostras disponíveis para a construção e treinamento de modelos são rotuladas. Logo, é possível calcular erros de estimação para adaptação de parâmetros de modelos. Por outro lado, o aprendizado não-supervisionado não tem acesso aos valores desejados da saída do modelo. Portanto, métodos no contexto não-supervisionado devem fazer uso de informações disponíveis apenas nos dados de entrada para adaptação de modelo (KUNCHEVA, 2014).

Algoritmos de aprendizado de máquina em geral geram modelos que podem lidar com problemas de classificação, clusterização, mineração de padrões frequentes, previsão de valores

futuros de séries temporais, regressão para aproximação de função, e controle de processos. Existem milhares de problemas possíveis de serem formulados em cada um destes contextos. Um mesmo algoritmo de aprendizado pode prover soluções interessantes em muitos deles.

Os algoritmos de aprendizado que seguem a vertente incremental são identificados por duas características principais (GIRAUD-CARRIER, 2000). São elas:

- Dispensam o reprocessamento de exemplos anteriores em um fluxo de dados.
- Como cada hipótese de solução gerada pode ser vista como a melhor aproximação até o momento, a qualquer momento um modelo munido de algoritmo de aprendizado incremental pode passar a prover uma resposta melhor.

A presença de fluxos de dados online é massiva nos sistemas atuais. Há uma clara necessidade de extração de informação de dados em tempo real, modelagem, análise e compreensão desses sistemas. Esses fluxos de dados se originam de diversas fontes, como monitoramento e controle industrial, satélites, mercado de ações, entretenimento de mídia, dispositivos móveis, multimídia, saúde, sistemas financeiros e meteorológicos, entre outros (LEITE, 2012) (SKRJANC et al., 2019).

2.2 Trabalhos Relacionados: Uma Visão Geral

Alguns trabalhos sobre clusterização incremental de fluxos de dados são brevemente discutidos nessa seção.

Em (ANGELOV; ZHOU, 2008b) apresenta-se uma abordagem chamada eClass (classificador evolutivo), fundamentada em um sistema baseado em regras fuzzy evolutivas do tipo Takagi-Sugeno. eClass se divide em eClass0, onde os termos consequentes das regras do modelo classificador representam rótulos de classe; e eClass1, que faz regressão sobre o vetor de características usando um classificador fuzzy Takagi-Sugeno evolutivo de primeira ordem. Modelos eClass podem começar a aprender “do zero”, i.e., as regras fuzzy não precisam ser pré-especificadas e o número de classes pode aumentar de acordo com novos rótulos de classe, sendo adicionado pelo processo de aprendizado online.

Um método para adaptação de modelos a partir de fluxos de dados variantes no tempo é descrito em (BIFET; GAVALDÀ, 2009). Árvores de Hoeffding trata-se de um indutor incremental de árvores de decisão usando fluxos de dados. Os modelos de árvore de decisão gerados podem lidar com distribuições de probabilidade variáveis e com *concept drift* – mudanças

graduais no ambiente. É descrito um algoritmo baseado em janela deslizante e um algoritmo adaptativo baseado em uma abordagem amostra por amostra. Os métodos são baseados em detectores de mudança e módulos estimadores.

Em (LEITE; COSTA; GOMIDE, 2010) (LEITE; COSTA; GOMIDE, 2013b) foi apresentada uma estrutura de rede neural fuzzy adaptativa para classificar fluxos de dados. A rede usa um algoritmo de aprendizado parcialmente supervisionado. A estrutura consiste em uma rede neural granular fuzzy evolutiva capaz de processar fluxos de dados não-estacionários usando um algoritmo incremental que lê e descarta amostras uma a uma. A rede evolui hiper caixas fuzzy no espaço de entrada, e usa neurônios baseados em normas triangulares nulas (*null-norms*) para agregar as contribuições locais. O algoritmo de aprendizado realiza a adaptação estrutural e paramétrica do modelo de rede sempre que as mudanças no ambiente são refletidas nos dados sequenciais de entrada. Não é necessário qualquer conhecimento prévio sobre a propriedade estatística dos dados e das classes.

Em (CARVALHO; TENÓRIO, 2010), um algoritmo de clusterização denominado fuzzy K-means para dados intervalares é proposto. Faz-se uso de distâncias quadráticas entre dados e modelos locais ao longo das decisões do processo de agrupamento. O modelo final é uma partição fuzzy do espaço dos dados que otimiza um critério de adequação. As distâncias quadráticas adaptativas mudam a cada iteração do algoritmo e podem ser a mesma, ou não, para todos os clusters gerados. Procedimentos adicionais para interpretação dos clusters fuzzy válidos em intervalos são apresentados.

Um sistema para classificação e regressão baseada em fluxos de dados (IBLStreams) é apresentado em (SHAKER; HÜLLERMEIER, 2012). Quando um novo exemplo é apresentado, ele é adicionado à uma base e é verificado se outros exemplos podem ser removidos da base, seja porque se tornaram redundantes ou porque são *outliers*. Essa verificação é feita com base em um conjunto C de exemplos que são mais próximos, vizinhos do exemplo que foi disponibilizado. A vizinhança é determinada por uma medida de distância, que é um versão simplificada da métrica de diferença de valor (VDM). Em seguida, é determinada a classe mais frequente entre os exemplos recentes. Se essa classe corresponde à classe do exemplo atual, os exemplos em C que têm rótulo de classe diferente, e não pertencem aos exemplos mais recentes, são removidos. Além disso, para garantir que um limite superior no tamanho da base não seja ultrapassado, o exemplo mais antigo é eliminado, independente da sua classe. Finalmente, no caso de uma mudança ser detectada, os exemplos atuais são descartados de acordo com a porcenta-

gem correspondente da diferença entre o menor erro de predição das últimas cem instâncias de treino e o erro de classificação para as últimas vinte instâncias.

Em (MENA-TORRES; AGUILAR-RUIZ, 2014) é descrito o Classificador de Fluxos de Dados baseado em Similaridade (SimC), que se fudamenta em uma política de inserção/-remoção que visa adaptação segundo a tendência dos dados e a manutenção de um conjunto pequeno e representativo de estimadores locais. A base de casos preserva um número limitado de exemplos para o qual cada inserção leva à eliminação de outro exemplo. Quando todas as instâncias de um grupo são removidas, esse grupo também é excluído. Quando um novo grupo é criado, é feita uma verificação ao longo da base de casos no sentido de eliminar os grupos mais antigos que contêm uma única instância, o que ajuda o controle de *outliers*. A abordagem SimC é inspirada no IBLStreams.

Em (KRAWCZYK; WOŹNIAK, 2015) é proposto uma extensão de máquina de vetores de suporte ponderada de uma classe (WOCSVM). Modifica-se os pesos atribuídos a objetos do conjunto de dados, o que permite mudar a forma da fronteira de decisão para novos dados recebidos. Os maiores pesos são atribuídos aos objetos provenientes dos novos blocos de dados, fazendo com que esses objetos tenham prioridade máxima na formação da nova fronteira de decisão. Os pesos de objetos de blocos de dados anteriores são reduzidos a cada iteração. Dessa forma a proposta lida com *concept drift*.

2.3 Classificação Incremental: Estado da Arte

Um esquema de classificação pode ser visto como uma forma de sistematizar um grande conjunto de dados de maneira compacta. A informação deve ser idealmente recuperada a partir do modelo de maneira eficiente. Caso seja possível sintetizar os dados em um pequeno número de clusters, então os rótulos atribuídos aos clusters propiciam uma descrição concisa de padrões ou classes (EVERITT et al., 2011).

Em (XU; WANG, 2016), uma abordagem baseada em máquina de aprendizagem extrema (ELM) é proposta. A abordagem utiliza redes neurais feedforward de camada oculta única. Máquinas de aprendizagem extrema decidem de forma adaptativa o número de neurônios na camada oculta da rede, além disso, funções de ativação também são selecionadas aleatoriamente de uma coleção de funções para melhorar o desempenho da abordagem. Finalmente, o algoritmo proposto treina uma série de classificadores, e os resultados da decisão para dados não-rotulados são feitos por uma estratégia de votação ponderada. Quando o conceito do

sistema que gera o fluxo de dados muda, cada classificador é atualizado de forma incremental usando novos dados ou os classificadores com piores desempenhos são eliminados.

Em (DORA et al., 2016) uma rede neural spiking para problemas de classificação de padrões, denominada classificador Neural Spiking Evolutivo Autorregulável (SRESN), gerencia o processo de aprendizagem. É proposta uma SRESN de duas camadas. A camada de entrada consiste de neurônios de campo receptivo que convertem uma entrada de valor real em *spikes* (impulsos), usando um esquema de codificação de população sem atraso. A camada de saída consiste de neurônios de integração e disparo. Durante o treinamento, o algoritmo de aprendizado de SRESN evolui os neurônios na camada de saída com base no fluxo de dados e informações armazenadas na rede. Dependendo da amostra de dados e de parâmetros da rede, o algoritmo pode optar por adicionar um neurônio, atualizar parâmetros, ou ignorar a amostra (autorregulação). No caso de adição de neurônios, os pesos para o neurônio recém-adicionado são inicializados usando um esquema de ordem de classificação modificado.

Em (SOARES et al., 2017) uma variação do método inteligente baseado em nuvens de dados, conhecido como método de tipicidade e excentricidade para análise de dados (TEDA), é aplicada à predição de séries temporais meteorológicas. A fim de predizer a temperatura média mensal, modelos não-lineares e variantes no tempo são desenvolvidos. TEDA é um algoritmo incremental que considera a densidade de dados e o espalhamento de nuvens no espaço dos dados, o qual não exige conhecimento prévio do conjunto de dados. Se algum conhecimento sobre o número de nuvens estiver disponível, ele pode ser expresso por meio de um único parâmetro. De outra forma, o processo de aprendizado e construção de modelo é totalmente autônomo. Valores passados de temperatura mensal, bem como valores prévios de variáveis exógenas, como nebulosidade, chuva e umidade, são considerados na análise. Com o objetivo de classificar e selecionar as características mais relevantes e os atrasos de tempo para uma predição mais precisa é utilizado um método não-paramétrico baseado em correlações de Spearman. Um *ensemble* de modelos de nuvens e regras fuzzy também é construído para fornecer predições numéricas e granulares das séries temporais. A previsão granular fornece possíveis valores de temperatura e dão uma ideia sobre o erro e a incerteza associados às previsões numéricas.

Um algoritmo de clusterização evolutivo para o processamento de relatórios de anomalia rodoviária é apresentado em (LI et al., 2017). A ideia é localizar anomalias isoladas e comprimir informações para anomalias densamente distribuídas. Duas categorias de clusters, principal e *outlier*, são definidas para dados das estradas. A distância de Mahalanobis é explo-

rada para quantificar a similaridade entre um novo relatório e os clusters existentes. Os clusters são mantidos em modo online. O lema inverso da matriz de Woodbury é usado para atualizações recursivas do modelo de anomalias.

O uso de aprendizado incremental para criar e melhorar modelos de análise multivariada em quimiometria de dados espectrais é proposto em (DIAZ-CHITO et al., 2017). É apresentado o uso de uma técnica de aprendizado incremental baseada em subespaços, denominada Vetores Comuns Discriminativos Generalizados Incrementais (IGDCV). Após a aplicação de IGDCV, as amostras são projetadas em um subespaço mais discriminativo que o espaço original e, então, aplica-se a técnica de k-Vizinhos Mais Próximos (kNN) para clusterizá-las e classificá-las. Como estudo de caso, classificou-se tipos de óleos vegetais.

Em (HU et al., 2018) ressalta-se que a capacidade de realizar atividades da vida diária é um indicador importante da condição de saúde dos seres humanos. Apresenta-se um método de aprendizado incremental denominado Florestas Aleatórias Incrementais de Classe (CIRF) para permitir que modelos existentes de reconhecimento de atividades humanas identifiquem novas atividades. Para isso, é projetada uma estratégia de divisão baseada no teorema de eixo de separação. Insere-se nós internos e adota-se o índice de Gini, ou ganho de informação, para dividir as folhas da árvore de decisão em florestas aleatórias.

2.4 Fluxo de Dados Numéricos e Incertos

Fluxo de dados é uma sequência ordenada de vetores $\mathbf{x}(k)$, $k = 1, ...$ Se $\mathbf{x}(k)$ é composto de elementos reais, i.e. $\mathbf{x}(k) \in \Re^n$, $\forall k$, então tem-se um fluxo de dados numéricos. n é a dimensão dos vetores do fluxo, i.e., $\mathbf{x}(k) = (x_1, ..., x_j, ..., x_n)$. De outra forma, se $\mathbf{x}(k)$ é constituído de elementos intervalares, i.e. $\mathbf{x}(k) \in \mathbb{I}^n$, $\forall k$, então tem-se um fluxo de dados intervalar. Nesse caso, $\mathbf{x}(k) = ([\underline{x}_1, \bar{x}_1], ..., [\underline{x}_j, \bar{x}_j], ..., [\underline{x}_n, \bar{x}_n])$. O acesso aos dados de um fluxo acontece ordenadamente. Um vetor, geralmente, pode ser observado apenas uma vez (GUHA et al., 2003) (ANGELOV, 2013) (LEITE; COSTA; GOMIDE, 2012).

A modelagem dirigida por fluxos de dados é motivada por aplicações que envolvem uma variedade de fontes e pela necessidade de extrair a essência da informação dos dados em tempo real. São exemplos de áreas de aplicação: controle de processos, monitoramento via satélite, mercado de ações, mídias, dispositivos móveis, sistemas de saúde, sistemas financeiros e meteorológicos, para mencionar alguns. Como o tamanho dos conjuntos de dados usualmente excede o espaço de armazenamento, isto é, a memória disponível, um algoritmo não-incremental não

consegue reter todos os comportamentos e novidades. Algoritmos incrementais mantêm um sumário dos dados passados em uma granularidade suficiente (LEITE, 2012) (LEITE et al., 2019). Seus procedimentos são geralmente simples visto as limitações de espaço e tempo (HAN; PEI; KAMBER, 2011).

Um ponto importante em fluxos de dados é a possibilidade de mudanças. Algoritmos de aprendizado convencionais assumem que os dados são gerados de forma aleatória ou seguem uma distribuição de probabilidade estacionária. Em vista disso não é preciso atualizar um modelo induzido a partir de uma quantidade satisfatória de dados históricos. No entanto, há alta possibilidade de acontecer mudanças ao longo do tempo em sistemas físicos. Dessa forma, uma abordagem natural de aprendizado consiste em adotar modelos adaptativos (GAMA; RODRIGUES, 2009).

2.4.1 Mudança de Conceito

Dados gerados de maneira contínua em ambientes dinâmicos e em larga escala estão sujeitos a mudanças ao longo do tempo. Essas alterações são denominadas mudanças de conceito (LUGHOFER; ANGELOV, 2011) (LEITE, 2012).

A mudança real de conceito ocorre na distribuição condicional da variável de saída a ser predita, $p(y|\mathrm{x})$. Por exemplo, a classificação adaptativa de documentos com respeito ao interesse particular de uma pessoa é um exemplo de aplicação em que ocorre mudança real de conceito. Um usuário de um portal da internet que estava interessado em comprar um carro inicialmente classificava todas as notícias relacionadas a esse assunto como relevantes. Ao passar do tempo, começou a descartar notícias sobre modelos de carro que não lhe interessavam mais. Em seguida, comprou o carro desejado e todas as notícias relacionadas passaram a ser classificadas como irrelevantes.

Já a mudança virtual de conceito ocorre na distribuição dos dados, i.e., $p(\mathrm{x})$ muda, e conduz mudanças na função de saída (fronteira de decisão em problemas de classificação). Um exemplo de onde pode ocorrer a mudança de conceito virtual é na categorização de *spams*. Enquanto o entendimento do que é *spam* permanece inalterado relativamente por um longo período, a frequência dos diferentes tipos de *spams* pode mudar drasticamente. Em ambos os casos, real e virtual, o modelo deve ser atualizado ou substituído (TSYMBAL, 2004). As mudanças de conceito também podem ser distinguidas entre graduais e abruptas. Nos exemplos anteriores, o leitor que queria comprar um carro passou a descartar gradualmente notícias que

não lhe interessavam, enquanto que na categorização de *spams*, um domínio que antes era classificado automaticamente como *spam* passa a ser abruptamente considerado como um email relevante, caso o usuário modifique manualmente a categoria dos emails recebidos daquele domínio.

As mudanças graduais, conhecidas como *concept drift*, geralmente são mais complicadas de serem detectadas do que as mudanças abruptas, conhecidas como *concept shift*, visto que as perturbações inicias podem ser compreendidas como ruídos. Quando um modelo se depara com esta situação, ele deve aguardar a chegada de novos casos para tentar distinguir se está acontecendo uma mudança, ou se são apenas interferências aleatórias. Em um caso extremo, as alterações podem acontecer de forma tão lenta que podem ser confundidas com estacionariedade (GAMA et al., 2014).

O esquecimento de modelos locais antigos ou inativos é uma abordagem usual para lidar com mudanças. Esses modelos locais podem caracterizar um conceito distinto em relação ao que tem acontecido recentemente (LEITE, 2012). A fim de esquecer as observações antigas em fluxos de dados, o mecanismo mais utilizado é janela deslizante. Apenas os exemplos que estão dentro de tal janela são utilizados na indução de modelos. Janelas baseadas em sequência e janelas baseadas em tempo são os dois tipos básicos de janelas deslizantes (BABCOCK et al., 2002).

2.5 Dados com Incerteza Não-estruturada

Dados com incerteza não-estruturada são representados nesse trabalho como dados intervalares. A incerteza nos dados geralmente surge em situações de (AGGARWAL; PHILIP, 2009):

- transformação de informação qualitativa em quantitativa;
- preservação de privacidade, como por exemplo, dados demográficos que normalmente é possível ter acesso apenas a conjuntos de dados parcialmente agregados;
- pré-processamento de dados experimentais, como na extração de atributos importantes a partir de atributos mensuráveis usando equações algébricas ou diferenciais aproximadas, tão bem como métodos estatísticos de previsão ou imputação de valores faltantes;
- limitação de dispositivos, como por exemplo, da saída de sensores devido a amplitude de entrada; contribuições de ruído ou erros na transmissão;

- trajetória de objetos móveis, uma vez que é possível prever o comportamento futuro do objeto apenas aproximadamente;

- imprecisões naturais associadas à própria maneira com que os dados foram coletados, questões temporais, condições ambientais, condução de experimentos por diferentes pessoas, entre outros.

2.6 Clusterização

A análise de grupos é uma ferramenta de análise de dados exploratória, a qual tem como objetivo organizar um conjunto de itens (geralmente representado como um vetor de valores quantitativos em um espaço multidimensional) em grupos. Em outras palavras, o objetivo do agrupamento é encontrar estrutura em dados (FRIEDMAN; HASTIE; TIBSHIRANI, 2001) (XU; WUNSCH, 2005). Isso permite que a compreensão desses dados seja mais clara, mais simples e que a informação possa ser recuperada de maneira mais eficiente.

As técnicas de agrupamento ou clusterização podem ser divididas em métodos hierárquicos e particionais. Métodos hierárquicos produzem uma série de partições baseadas em critérios de aglomeração e divisão de acordo com a medida de similaridade. Métodos aglomerativos iniciam com cada objeto formando um grupo (*singleton*), e interativamente os pares de grupos mais próximos são combinados até que todos os objetos estejam em apenas um grupo. Já um método divisivo vai na direção oposta do aglomerativo, isto é, inicia com todos os objetos em um único grupo, e iterativamente se divide em uma maior quantidade de grupos de acordo com o afastamento dos objetos no espaço (XU; WUNSCH, 2005).

Os métodos particionais calculam diretamente as partições ao otimizar uma função objetivo. A função objetivo tipicamente envolve minimizar a similaridade inter-grupos e maximizar a similaridade de dados alocados em um mesmo grupo. Entre os métodos particionais estão os métodos baseados em protótipos, que muitas vezes são pontos no espaço dos dados que representam os grupos. Alguns dos algoritmos mais conhecidos são *K-Means* (MACQUEEN et al., 1967) e *Fuzzy C-Means* (FCM) (BEZDEK, 1981). Eles usam uma função vizinhança para preservar as propriedades topológicas dos dados (FRIEDMAN; HASTIE; TIBSHIRANI, 2001).

Os algoritmos particionais, como, por exemplo, o *K-Means*, inicialmente necessitam da base de dados completa e que o usuário informe o número k de clusters a ser gerado. Esses

algoritmos são sensíveis à diferença de tamanho e densidade dos clusters, o que afeta significativamente a eficiência deles (ANKERST et al., 1999) (GUHA; RASTOGI; SHIM, 1998). Diferentemente, o método proposto nesse trabalho lida com os dados à medida que estes vão sendo disponibilizados; gera e adapta o número de grânulos automaticamente e com tamanho e densidade diferentes.

Algoritmos de classificação normalmente usam rótulos de categoria que marcam objetos com identificadores anteriores, ou seja, rótulos de classe, para particionar os objetos em grupos (aprendizado supervisionado), como é o caso, por exemplo, da Análise Discriminante de Fisher (FISHER, 1936). Diferentemente, algoritmos de clusterização não usam rótulos de classe (aprendizado não-supervisionado), o que lhes dão vantagem, pois em muitos casos, esses rótulos de classe podem simplesmente não existir, e tais algoritmos podem ser usados para estimá-los.

Em clusterização, o particionamento dos objetos em grupos (clusters) é baseado no princípio de que amostras dentro de um mesmo grupo (intra-clusters) são mais similares e amostras em grupos diferentes (inter-clusters) são menos similares (HAN; PEI; KAMBER, 2011). A ideia de similaridade é formulada matematicamente a partir de uma medida de distância. Cada grupo tem um protótipo que o representa, geralmente definido como o centro do grupo, então as distâncias podem ser medidas a partir desse protótipo.

2.6.1 Medidas de Distância entre Vetores Reais e Intervalares

2.6.1.1 Família de Minkowski para Vetores Reais

Considere um vetor de entrada com elementos reais, $\mathbf{x}$, e um vetor real de mesma dimensão, $\mathbf{v}^i$, que representa o centro do i-ésimo cluster de uma coleção de clusters. A partir da família de distâncias ou normas de Minkowski,

$$d_p = \|\mathbf{x} - \mathbf{v}^i\|_p = (\sum_{j=1}^{n} |x_j - v_j^i|^p)^{\frac{1}{p}} = \sqrt[p]{|x_1 - v_1^i|^p + |x_2 - v_2^i|^p + ... + |x_n - v_n^i|^p} \tag{2.1}$$

com $p \geq 1$ para convexidade. Medidas de distância específicas podem ser obtidas a partir de valores adotados para p. Se $p = 1$, tem-se a distância de Manhattan ou *city block*,

$$d_1 = \|\mathbf{x} - \mathbf{v}^i\|_1 = |x_1 - v_1^i| + |x_2 - v_2^i| + ... + |x_n - v_n^i| \tag{2.2}$$

Caso $p = 2$, tem-se a distância Euclidiana,

$$d_2 = \|\mathbf{x} - \mathbf{v}^i\|_2 = \sqrt[2]{|x_1 - v_1^i|^2 + |x_2 - v_2^i|^2 + ... + |x_n - v_n^i|^2} \tag{2.3}$$

Se $p = \infty$, tem-se a distância de Chebyshev,

$$d_\infty = \|\mathbf{x} - \mathbf{v}^i\|_\infty = max(|x_1 - v_1^i| + |x_2 - v_2^i| + ... + |x_n - v_n^i|) \tag{2.4}$$

2.6.1.2 Distância de Hausdorff para Vetores Intervalares

A distância de Hausdorff é uma métrica adequada para medir a distância entre dois vetores intervalares. Ela permite que um algoritmo de clusterização reconheça grânulos de diferentes tamanhos e formas.

Sejam x e v dois intervalos, isto é, $x = [\underline{x}, \bar{x}]$ e $v = [\underline{v}, \bar{v}]$. A distância entre esses intervalos pode ser dada por

$$d(x, v) = max(|\underline{x} - \underline{v}|, |\bar{x} - \bar{v}|) \tag{2.5}$$

Esta medida pode ser estendida para vetores de intervalos. Seja $\mathbf{x} = (x_1, ..., x_n)$ e $\mathbf{v} = (v_1, ..., v_n)$. Logo,

$$d(\mathbf{x}, \mathbf{v}) = (max(|\underline{x}_1 - \underline{v}_1|, |\bar{x}_1 - \bar{v}_1|), ..., max(|\underline{x}_n - \underline{v}_n|, |\bar{x}_n - \bar{v}_n|)) \tag{2.6}$$

Obtém-se de fato um vetor de intervalos. Caso apenas um número que represente a distância entre os vetores intervalares for requerido, então a distância total é obtida de

$$d_H(\mathbf{x}, \mathbf{v}) = max(d(\mathbf{x}, \mathbf{v})) \tag{2.7}$$

onde $d_H(\mathbf{x}, \mathbf{v})$ é a distância de Hausdorff.

2.6.2 Índice de Validação

Um índice de validação determina a qualidade do agrupamento resultante. Há índices de validação internos e externos, sendo que o primeiro baseia-se somente no resultado adquirido pelo algoritmo de clusterização para avaliar a qualidade do agrupamento. O segundo baseia-se em informações externas, geralmente o rótulo das classes. Índices internos são mais utilizados, pois em aplicações do mundo real nem sempre a informação externa está disponível.

A maioria dos índices internos de validação combinam duas propriedades, sendo elas separação e compacidade. A separação é geralmente medida usando a distância entre centros de grânulos, enquanto a compacidade é geralmente alguma medida de densidade dos pontos de dados em cada grânulo (IBRAHIM; KELLER; BEZDEK, 2018).

Nesse trabalho é utilizado um índice de validação incremental, baseado no índice de validação Xie-Beni, dado por

$$X = \sum_{i=1}^{c} \sum_{k=1}^{n} \frac{\mu_{ik}^{q} \|x_k - v_i\|^2}{n.min_{i \neq j}(\|v_i - v_j\|^2)} \tag{2.8}$$

onde c é o número de grânulos; n é o número de amostras; q é uma constante de fuzzicidade; x é a amostra; v é o centro do grânulo.

Nesta equação note que o numerador indica a compacidade da partição, i.e., a variância intra-cluster, e o denominador indica a separação dos centros de grânulos. A melhor partição minimiza X.

3 IUP: ALGORITMO INCREMENTAL BOTTOM-UP

3.1 Preliminares

O método proposto trata-se de um algoritmo de clusterização intervalar incremental bottom-up a partir de fluxos de dados. Diferentemente de outros métodos evolutivos de processamento e modelagem de fluxos de dados numéricos, o método proposto lida com fluxos de dados que apresentam incerteza não-estruturada representados por valores intervalares, e também com fluxos de dados numéricos. Fluxo de dados é uma sequência ordenada de vetores $\mathbf{x}(k)$, $k = 1, ...$ Os vetores $\mathbf{x}(k)$ de um fluxo de dados numéricos são constituídos por elementos reais, i.e. $\mathbf{x}(k) = (x_1, ..., x_j, ..., x_n)$. Já os vetores $\mathbf{x}(k)$ de um fluxo de dados intervalar são compostos de elementos intervalares, i.e. $\mathbf{x}(k) = ([\underline{x}_1, \bar{x}_1], ..., [\underline{x}_j, \bar{x}_j], ..., [\underline{x}_n, \bar{x}_n])$.

O modo como acontece o desenvolvimento dos modelos locais (grânulos) do algoritmo proposto a partir dos dados é baseado no conceito de aprendizado bottom-up. Nesse caso os grânulos de informação incialmente são tão detalhados quanto os dados. Um grânulo é, de fato, um dado. Os grânulos podem se expandir conforme dados semelhantes, pertencentes à sua ξ-vizinhança máxima, são apresentados. Geometricamente, os grânulos apresentam formato de hiper-caixas n-dimensionais no espaço dos dados de entrada, também n-dimensionais. Os dados em si também são hiper-caixas, porém de menor tamanho. A ideia é que haja compressão de hiper-caixas menores similares, ou seja, que carregam informação semelhante, em hiper-caixas maiores, mais abstratas, mas que carregam um rótulo de classe único e válido para quaisquer hiper-caixas inclusas.

Define-se protótipos de grânulos como vetores $\mathbf{v}^i = (v^i_1, ..., v^i_j, ..., v^i_n)$. Seus elementos são valores reais, i.e., $\mathbf{v}^i \in \mathbb{R}^n$. O j-ésimo elemento de $\mathbf{v}^i$ é obtido de

$$v^i_j = \frac{\underline{\xi}^i_j + \bar{\xi}^i_j}{2} \tag{3.1}$$

onde $\xi^i_j = [\underline{\xi}^i_j, \bar{\xi}^i_j]$ é a largura do i-ésimo grânulo na j-ésima dimensão. Note que $\mathbf{v}^i$ é o ponto central da hiper-caixa ξ^i. A hiper-caixa ξ^i é fechada e limitada e, portanto, é um conjunto compacto conforme o teorema de Borel-Lebesgue (RUDIN, 1953). Ela estabelece a borda do grânulo i, i.e., uma restrição generalizada no sentido da teoria geral da incerteza (ZADEH, 2006) e da computação granular de Lin e Zadeh (ZADEH, 1998).

O algoritmo iUP requer a escolha de um ou dois hiper-parâmetros. São eles: a máxima expansão ou ξ-vizinhança máxima dos grânulos, ξ_{max}; e o número de rótulos de classe, λ, que

deseja-se atribuir aos c grânulos encontrados, $\lambda \leq c$. Caso não haja qualquer conhecimento especialista sobre a aplicação, ou noção sobre o número de classes existentes em uma base de dados, faz-se $\lambda = c$. De outra forma, se um número limite de classes quiser ser imposto, o parâmetro λ permite tal expressão. Por exemplo, em um problema de detecção de faltas, queremos identificar a presença de 3 tipos de falta, logo $\lambda = 3$. Caso o algoritmo encontre $c = 4$ grânulos, dois grânulos, vizinhos mais próximos, adquirem o mesmo rótulo. Logo, λ é um parâmetro que apenas oferece flexibilidade e um mecanismo de expressão de conhecimento especialista sobre o domínio do problema. Por vezes, e diferentemente de outros algoritmos de clusterização, evolutivos ou não, apenas por meio dele é possível obter soluções racionais e intuitivas em certos problemas.

O parâmetro $\xi_{max} \in [0,1]$ determina o tamanho máximo que um grânulo pode assumir em qualquer dimensão, i.e., $\bar{\xi}_j^i - \underline{\xi}_j^i \leq \xi_{max}$, $j = 1,...,n$; $i = 1,...,c$, em qualquer iteração k. ξ_{max}, indiretamente, mantém um certo controle sobre a quantidade de grânulos que são mantidos em um modelo. A indicação explícita do número de grânulos que uma base de dados contém é desnecessária no ambiente iUP. Essa é uma característica interessante em várias aplicações e corrobora com o conceito de sistemas autônomos (*fully-autonomous systems*) (ANGELOV, 2013) (ANGELOV; FILEV; KASABOV, 2010) (LEITE, 2012). O número de grânulos existentes é encontrado automaticamente pelo algoritmo iUP a partir da percepção das diferenças entre as amostras de dados. Não somente os parâmetros dos grânulos, mas também a quantidade de grânulos no modelo é adaptativa ao longo do tempo. A capacidade de adaptação estrutural e paramétrica de modelos classificadores ao longo da operação online permite lidar com mudanças de conceito abruptas e graduais (SKRJANC et al., 2019). Essa é uma vantagem significativa do algoritmo e modelo iUP proposto, frente a grande maioria de propostas de classificadores. Além disso, em particular, iUP lida com dados incertos e dados numéricos.

O algoritmo iUP consiste ainda da mesclagem de grânulos e da deleção de grânulos inativos como procedimentos de compactação estrutural e manutenção de conhecimento atualizado do domínio do problema. Um método baseado na vizinhança dos grânulos é empregado para rotulação (atribuição de classes) de grânulos. O índice de validação incremental de Xie-Beni proposto, provê uma medida do desempenho do algoritmo de aprendizado e das partições do modelo iUP proposto. Dada esta introdução aos parâmetros-chave e conceitos gerais do algoritmo iUP, as próximas seções detalham os procedimentos do processo de aprendizado e adaptação.

3.2 Sobre a Similaridade entre Objetos Incertos

Diferentes formas para ser obter um valor quantitativo que represente a similaridade entre dois objetos, i.e., similaridade dado-dado, dado-cluster ou cluster-cluster, são descritas. A noção de similaridade entre um exemplo $\mathbf{x} \in \mathbb{I}^n$ e um cluster $\xi \in \mathbb{I}^n$, tal que $\mathbf{x}, \xi \subseteq [0,1]$, pode ser associada à noção de distância segundo

$$S(\mathbf{x}, \xi) \stackrel{\triangle}{=} 1 - d(\mathbf{x}, \xi) \tag{3.2}$$

A distância de Hausdorff, d_H, aplica-se diretamente a (3.2). Para que as distâncias de Manhattan, d_1, Euclidiana, d_2, e de Chebyshev, d_∞, possam ser empregadas, é preciso considerar pontos representativos dos objetos $\mathbf{x}$ e ξ. Note que, ao se adotar pontos representativos dos objetos, a noção de incerteza e vizinhança no entorno destes é perdida. Informação é perdida. Por exemplo, dois objetos intuitivamente afastados no espaço podem vir a apresentar uma região de interseção caso suas bordas sejam expandidas. Entretanto, os mesmos valores de distância serão obtidos usando d_1, d_2 e d_∞ – estejam os objetos intuitivamente afastados ou sobrepostos. Outros exemplos de perda de informação sobre a incerteza envolvida acontecem para vários casos de objetos não-hipercúbicos multi-dimensionais (LEITE; COSTA; GOMIDE, 2013a).

A distância/similaridade de Hausdorff é adotada em iUP. Ela pode ser empregada mesmo quando um ou mais objetos se degeneram em um ponto no espaço.

3.3 Criação de Grânulo

Caso não haja nenhum grânulo ξ^i, $i = 1, ..., c$, que inclua completamente o exemplo $\mathbf{x}(k)$ e, adicionalmente, nenhum grânulo possa expandir para incluir completamente $\mathbf{x}(k)$, segundo ξ_{max}, então um novo grânulo é criado, ξ^{c+1}. O novo grânulo é definido como

$$\begin{aligned} r_j^{c+1} &= \frac{\underline{x}_j(k) + \bar{x}_j(k)}{2} \qquad & s_j^{c+1} &= \frac{\bar{x}_j(k) - \underline{x}_j(k)}{2} \qquad & \forall j,\ j = 1, ..., n \\ \underline{\xi}_j^{c+1} &= r_j^{c+1} - s_j^{c+1} & \bar{\xi}_j^{c+1} &= r_j^{c+1} + s_j^{c+1} \end{aligned} \tag{3.3}$$

onde r_j é o ponto central; e s_j é a metade da largura.

O protótipo do grânulo, i.e., $\mathbf{v}^{c+1}$, é obtido análogo a (3.1). Sempre que um grânulo é criado, há também o ajuste do contador β do número de vezes que o grânulo foi acionado por uma amostra, sendo $\beta^{c+1} = 1$.

3.4 Adaptação de Grânulo

A adaptação dos grânulos está relacionada com o arraste e a expansão a que eles estão sujeitos.

3.4.1 Arraste

O grânulo ξ^i é deslocado, sofre arraste, quando uma amostra $\mathbf{x}(k)$ pertence completamente à sua ξ-vizinhança. Para que essa condição seja satisfeita é necessário que

$$\underline{x}_j(k) \geq \underline{\xi}^i_j \quad \text{e} \quad \bar{x}_j(k) \leq \bar{\xi}^i_j \quad \forall j, \;\; j = 1, ..., n$$

O arraste é descrito por

$$\begin{aligned} \mathbf{r}^i_{jk} &= \frac{\underline{x}_j(k) + \bar{x}_j(k)}{2} & \mathbf{s}^i_{jk} &= \frac{\bar{x}_j(k) - \underline{x}_j(k)}{2} & \forall j, \;\; j = 1, ..., n \\ \rho &= mediana(\mathbf{r}^i_{jk}) & \sigma &= mediana(\mathbf{s}^i_{jk}) & \\ \underline{\xi}^i_j &= \rho - \sigma & \bar{\xi}^i_j &= \rho + \sigma & \end{aligned} \tag{3.4}$$

O protótipo do grânulo $\mathbf{v}^i$ é obtido análogo a (3.1). Há também o ajuste do contador do número de vezes que o grânulo foi acionado por uma amostra, sendo $\beta^i(k+1) = \beta^i(k) + 1$.

No caso da amostra pertencer a mais de uma ξ-vizinhança, o grânulo que a amostra tem mais similaridade de acordo com o maior valor de $S(\mathbf{x}(k), \xi^i)$, $\forall i$, é deslocado. A similaridade é obtida conforme a Eq. (3.2).

3.4.2 Expansão

O grânulo ξ^i pode expandir quando uma amostra $\mathbf{x}(k)$ pertence à sua região de expansão. Para que essa condição seja satisfeita é necessário que

$$\begin{aligned} &\text{Região 1:} \quad (\bar{\xi}^i_j - \xi_{max}) < \underline{x}_j(k) < \underline{\xi}^i_j \quad \text{e} \quad (\bar{\xi}^i_j - \xi_{max}) < \bar{x}_j(k) < \underline{\xi}^i_j \\ \text{ou} \quad &\text{Região 2:} \quad \bar{\xi}^i_j < \underline{x}_j(k) < (\underline{\xi}^i_j + \xi_{max}) \quad \text{e} \quad \bar{\xi}^i_j < \bar{x}_j(k) < (\underline{\xi}^i_j + \xi_{max}) \quad \forall j, \;\; j = 1, ..., n \end{aligned}$$

A expansão é definida por

$$\begin{aligned} \underline{\xi}_j^i &= \underline{x}_j(k) \quad \text{se } \mathbf{x}(k) \in \text{Região 1} \\ \text{ou } \quad \bar{\xi}_j^i &= \bar{x}_j(k) \quad \text{se } \mathbf{x}(k) \in \text{Região 2} \end{aligned} \tag{3.5}$$

O protótipo do grânulo $\mathbf{v}^i$ é obtido análogo a (3.1). Há também o ajuste do contador do número de vezes que o grânulo foi acionado por uma amostra, sendo $\beta^i(k+1) = \beta^i(k) + 1$.

No caso da amostra pertencer a região de expansão de mais de um grânulo, a ξ-vizinhança que a amostra tem mais similaridade de acordo com o maior valor de $S(\mathbf{x}(k), \xi^i)$, $\forall i$, é expandida. A similaridade é obtida conforme a Eq. (3.2).

3.5 Índice de Validação Incremental Xie-Beni

O índice de validação utilizado no algoritmo é um índice incremental baseado no índice de Xie-Beni. O índice de validação incremental Xie-Beni provê uma medida do desempenho do algoritmo de aprendizado e das partições do modelo iUP proposto. O índice é dado por

$$XB \triangleq \sum_{i=1}^{c} \frac{\mu_a^{i_1}(k) \cdot D_a^{i_1}(k)}{min_{\forall i_1, i_2;\ i_1 \neq i_2}(d_H(\xi^{i_1}, \xi^{i_2}))} \tag{3.6}$$

onde $\mu_a^{i_1}(k)$ é a pertinência acumulada das amostras à um grânulo; $D_a^{i_1}(k)$ é o acúmulo das distâncias das amostras que pertenceram à um grânulo e o centro do mesmo grânulo.

A pertinência acumulada é definida por

$$\mu_a^{i_1}(k+1) = \frac{\beta^{i_1} \cdot \mu_a^{i_1}(k) + \mu(\mathbf{x}(k), \xi^{i_1})}{\beta^{i_1} + 1} \tag{3.7}$$

onde $\mu(\mathbf{x}(k), \xi^{i_1})$ é obtido de

$$\mu(\mathbf{x}(k), \xi^{i_1}) = \frac{1}{\sum_{i=1}^{c} \frac{d_H(\mathbf{x}(k), \xi^{i_1})}{d_H(\mathbf{x}(k), \xi^{i_2})}} \tag{3.8}$$

O acúmulo das distâncias é definido por

$$D_a^{i_1}(k+1) = \frac{\beta^{i_1}\ D_a^{i_1}(k) + d_H(\mathbf{x}(k), \xi^{i_1})}{\beta^{i_1} + 1} \tag{3.9}$$

onde β^{i_1} é o número de vezes que o grânulo i_1 foi escolhido para ser adaptado no passado.

A melhor partição minimiza XB, então se é detectado crescimento significativo do índice de validação incremental Xie-Beni, o processo de mesclagem é acionado para compactar a estrutura do modelo classificador.

3.6 Mesclagem de Grânulos

A mesclagem de grânulos é um procedimento de compactação estrutural e manutenção de conhecimento atualizado do domínio do problema. O procedimento de mesclagem é acionado, como mencionado anteriormente, caso seja detectado crescimento do índice de validação incremental Xie-Beni.

A menor distância de um grânulo ao outro indica uma relação de vizinhança entre eles, então é analisado se os grânulos devem ser mesclados.

Se $(\xi_{max} - d_H(\xi^{i_1}, \xi^{i_2})) \geq 0$, ξ^{i_1} e ξ^{i_2} serão mesclados segundo

$$\begin{aligned} \underline{\xi}_j^{c+1} &= \frac{\underline{\xi}_j^{i_1} + \underline{\xi}_j^{i_2}}{2} \\ \bar{\xi}_j^{c+1} &= \frac{\bar{\xi}_j^{i_1} + \bar{\xi}_j^{i_2}}{2} \end{aligned} \tag{3.10}$$

onde ξ^{c+1} é a ξ-vizinhança do grânulo gerado da combinação.

O protótipo do grânulo $\mathbf{v}^{c+1}$ é obtido análogo a (3.1). Há também o ajuste do contador do número de vezes que o grânulo foi acionado por uma amostra, sendo $\beta^{c+1} = \beta^{i_1} + \beta^{i_2}$. Os grânulos i_1 e i_2 são removidos e, consequentemente a coleção de grânulos reduz em um elemento.

3.7 Adaptação da ξ-vizinhança Máxima dos Grânulos

A adaptação da ξ-vizinhança máxima dos grânulos, ξ_{max}, é beneficial após um número fixo de iterações.

Se $(c(k) - c(k-1)) \geq \frac{\mathbf{x}(k)}{2}$, ξ_{max} é ampliado segundo

$$\xi_{max}(k+1) = 1 + \frac{c(k) - c(k-1)}{k} \cdot \xi_{max}(k) \tag{3.11}$$

Se $(c(k) - c(k-1)) < \frac{\mathbf{x}(k)}{2}$, ξ_{max} é reduzido segundo

$$\xi_{max}(k+1) = 1 - 2 \cdot \frac{c(k) - c(k-1)}{k} \cdot \xi_{max}(k) \tag{3.12}$$

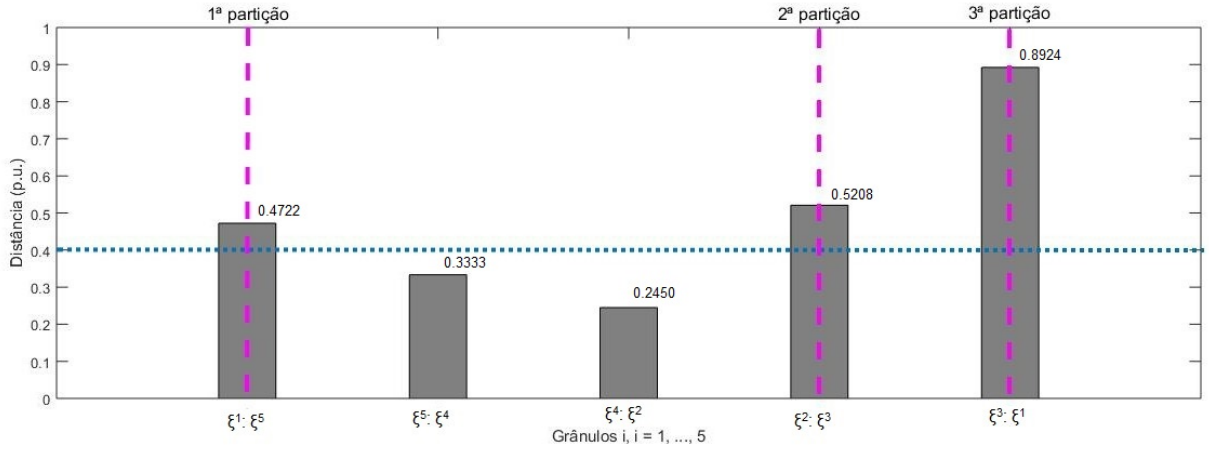
onde $c(k)$ é o número de grânulos na iteração k.

3.8 Rotulação de Grânulo

Um método baseado na vizinhança dos grânulos é empregado para rotulação (atribuição de classes) de grânulos. λ é o número de rótulos de classe que deseja-se atribuir aos c grânulos encontrados, sendo $\lambda \leq c$. Caso não haja qualquer conhecimento especialista da aplicação ou noção sobre o número de classes existentes em uma base de dados, faz-se $\lambda = c$. De outra forma, se um número limite de classes quiser ser imposto, o parâmetro λ permite tal expressão.

A vizinhança de grânulos é determinada pela menor distância. As maiores distâncias entre vizinhos simbolizam pontos de corte para rotular sequências de grânulos. Essas distâncias são identificadas de acordo com o número de rótulos de classe. Por exemplo, se $\lambda = 3$, as três maiores distâncias serão identificadas e determinadas como pontos de corte. A Figura 3.1 mostra um gráfico da relação de vizinhança entre grânulos.

Figura 3.1 – Relação de vizinhança entre grânulos



Fonte: do autor

Nesse exemplo é considerado três classes para a base de dados, i.e., $\lambda = 3$, logo as três maiores distâncias entre os grânulos vizinhos são identificadas (linha pontilhada azul e horizontal no gráfico) e simbolizam o ponto de corte para a rotulação (linhas tracejadas rosa e vertical no gráfico). Dessa forma, o grânulo à esquerda da primeira partição (localizada em 0.4722) é rotulado como classe 1, entre a primeira e segunda partição (localizada entre 0.4722 e 0.5208) os grânulos são rotulados como classe 2, entre a segunda e terceira partição (localizada

entre 0.5208 e 0.8924) como classe 3 e à direita da terceira partição (localizada em 0.8924) é uma continuação da primeira partição, sendo assim também rotulado como classe 1. Então classe 1 é definida para o grânulo ξ^1; classe 2 rotula os grânulos ξ^5, ξ^4, ξ^2; e classe 3 é atribuída ao grânulo ξ^3.

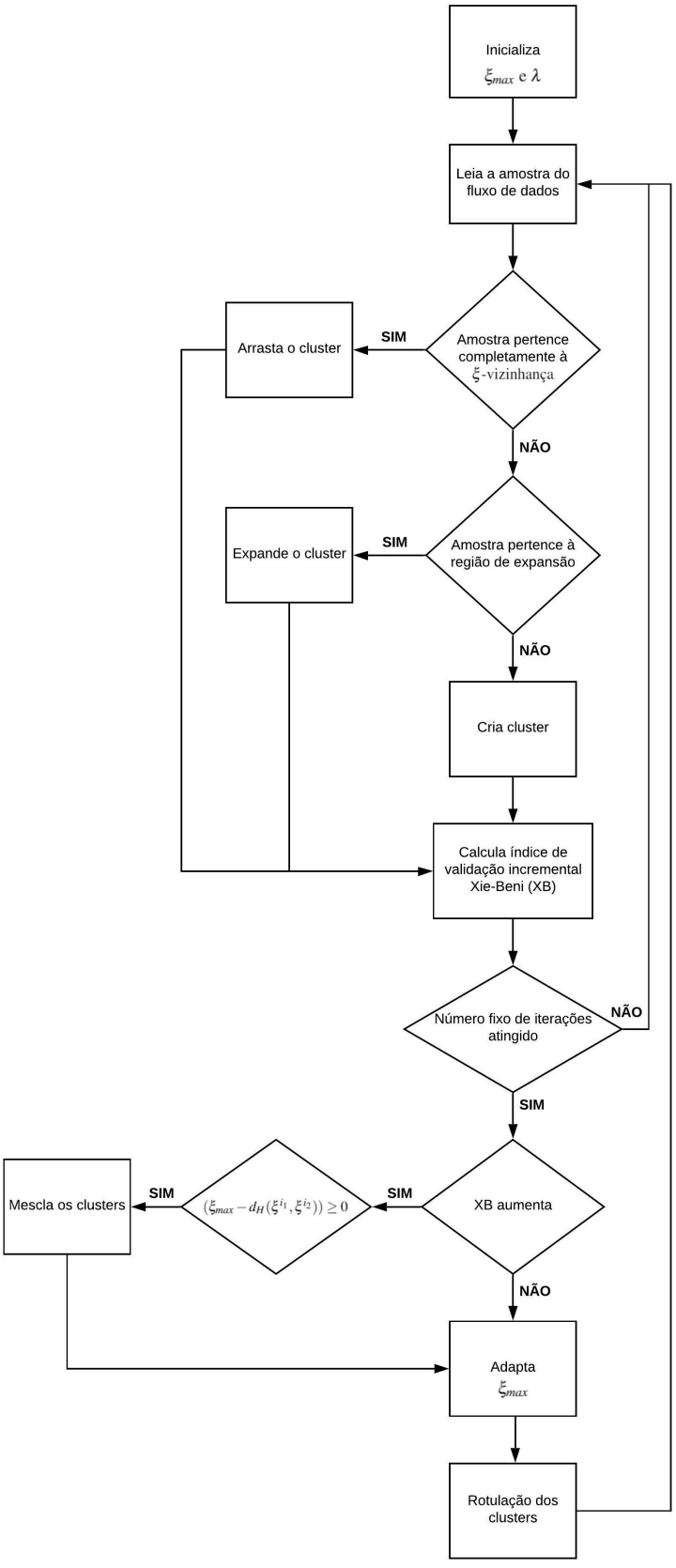
3.9 Pseudo-Código

O método proposto é apresentado no pseudo-código abaixo:

INÍCIO
Inicializa $\xi_{max} = 0.4$ e λ;
Repetir
  Leia $\mathbf{x}(k),\ k = 1,...;$
  Se $\underline{x}_j \geq \underline{\xi}^i_j$ e $\bar{x}_j \leq \bar{\xi}^i_j \ \ \forall j,\ j = 1,...,n;\ i = 1,...,c$
    Arrasta ξ^i;
    Atualiza $\mathbf{v}^i$; $\beta^i = \beta^i + 1$;
  Senão se $((\bar{\xi}^i_j - \xi_{max}) < \underline{x}_j < \underline{\xi}^i_j$ e $(\bar{\xi}^i_j - \xi_{max}) < \bar{x}_j < \underline{\xi}^i_j \ \ \forall j)$ ou $(\bar{\xi}^i_j < \underline{x}_j < (\underline{\xi}^i_j + \xi_{max})$ e $\bar{\xi}^i_j < \bar{x}_j < (\underline{\xi}^i_j + \xi_{max}) \ \ \forall j)$, $j = 1,...,n;\ i = 1,...,c$
    Expande ξ^i;
    Atualiza $\mathbf{v}^i$; $\beta^i = \beta^i + 1$;
  Senão
    Cria ξ^{c+1};
    Cria $\mathbf{v}^{c+1}$; $\beta^{c+1} = 1$;
  Fim se
  Calcula índice de validação incremental Xie-Beni;
  //Após um número fixo de iterações
  Se (XB aumenta)
    Se $(\xi_{max} - d_H(\xi^{i_1}, \xi^{i_2})) \geq 0$
      Mescla ξ^{i_1} e ξ^{i_2};
      Ajusta $\mathbf{v}^{c+1}$; $\beta^{c+1} = 1$;
      Remove clusters i_1 e i_2; $c = c - 1$;
    Fim se
  Fim se
  Adapta ξ_{max};
Rotulação dos clusters
FIM

Da perspectiva de fluxograma, o pseudo-código acima pode ser representado conforme a Figura 3.2.

Figura 3.2 – Fluxograma representativo do pseudo-código do algoritmo iUP



Fonte: do autor

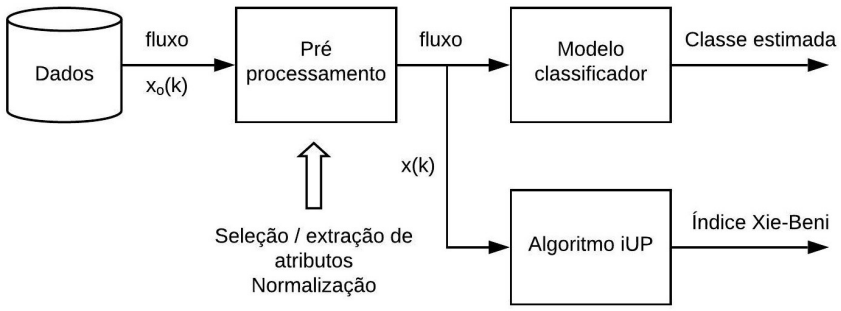
4 METODOLOGIA

4.1 Esquema Geral do Método

O iUP foi implementado no software MATLAB em um computador comercial Intel Core i3-4005U 1.70 GHz, RAM 4.00 GB, sistema operacional Windows 10 Home Single Language.

O método pode ser sintetizado conforme a Figura 4.1. O fluxo de dados passa por um pré-processamento de seleção/extração de atributos e normalização. O algoritmo iUP realiza os procedimentos do processo de aprendizado e adaptação de modelos. De acordo com o modelo classificador a classe da amostra é estimada.

Figura 4.1 – Esquema geral do iUP



Fonte: do autor

4.2 Bases de Dados

A fim dos fluxos de dados apresentarem incerteza não-estruturada, assume-se um intervalo de valores possíveis em torno dos valores numéricos reais. Não se assume qualquer estrutura estatística dentro dos limites do intervalo. Sabe-se somente que o valor correto está incluso no intervalo e trabalha-se com as bordas em espaços multi-dimensionais. Por conseguinte, os atributos das bases de dados serão abertos por um valor sorteado randomicamente dentro de uma faixa para cada amostra, ocasionando incerteza diferente para amostras diferentes. Dessa forma, o vetor de um fluxo de dados numéricos, $\mathbf{x}(k) = (x_1,...,x_j,...,x_n)$ passa pela alteração $\mathbf{x}(k) = ([(x_1 - \text{valor sorteado}),(x_1 + \text{valor sorteado})],...,[(x_j - \text{valor sorteado}),(x_j + \text{valor sorteado})],...,[(x_n - \text{valor sorteado}),(x_n + \text{valor sorteado})])$, resultando no vetor de um fluxo de dados intervalar, $\mathbf{x}(k) = ([\underline{x}_1,\bar{x}_1],...,[\underline{x}_j,\bar{x}_j],...,[\underline{x}_n,\bar{x}_n])$. As faixas consideradas para o sorteio randômico serão [0.05,0.1] e [0.1,0.2], sendo cada número com quatro casas decimais,

então os atributos podem ser abertos por cinquenta valores diferentes na primeira faixa e por um mil valores diferentes na segunda faixa.

As bases de dados consideradas para avaliação empírica de modelos iUP são descritas a seguir e podem ser encontradas no *UCI Machine Learning Repository*.

4.2.1 Flor Íris

Íris é um gênero de plantas com flor, muito apreciado pelas suas diversas espécies, que ostentam flores de cores muito vivas. Devido seu aspecto, são constantemente confundidas com orquídeas diversas, no entanto, essas plantas possuem detalhes únicos e especiais e nascem em diferentes tons, indo do branco ao azul vibrante.

A flor Íris é considerada perfeita para o clima brasileiro e por isso desperta ainda mais a atenção dos cultivadores do país. É uma espécie delicada, exuberante e de simples cuidados, o que faz com que essa flor seja vista como uma opção perfeita para decorar um jardim.

A base de dados possui 4 atributos e 150 amostras. Os atributos são: comprimento da sépala; largura da sépala; comprimento da pétala; largura da pétala. As classes estão relacionadas ao tipo da flor Íris, sendo elas: Íris Setosa; Íris Versicolour; Íris Virginica.

4.2.2 Pulsares

Os pulsares são um tipo raro de estrela de neutrons que produzem emissões de rádio detectáveis aqui na Terra. Eles são de considerável interesse científico como sondas do espaço-tempo, o meio interestelar e estados da matéria.

À medida que os pulsares giram, seu feixe de emissão varre o céu e, quando isso cruza nossa linha de visão, produz um padrão detectável de emissão de rádio banda larga. Como pulsares giram rapidamente, esse padrão se repete periodicamente. Assim, a busca por pulsares envolve procurar sinais de rádio periódicos com grandes radiotelescópios.

A base de dados possui 8 atributos e 527 amostras. Os atributos são: média do perfil integrado; desvio padrão do perfil integrado; excesso de curtose do perfil integrado; assimetria do perfil integrado; média da curva relação sinal-ruído do modulador delta (DM-SNR); desvio padrão da curva DM-SNR; excesso de curtose da curva DM-SRN; assimetria da curva DM-SNR. As classes estão relacionadas à identificação de pulsares, sendo elas: não é pulsar; é pulsar.

4.2.3 Parkinson

Parkinson é uma doença neurológica causada pela degeneração das células situadas na região do cérebro chamada substância negra. Essas células produzem a substância dopamina, que conduz as correntes nervosas (neurotransmissores) ao corpo. A redução ou falta da dopamina afeta os movimentos da pessoa, causando tremores, lentidão de movimentos, rigidez muscular, desequilíbrio, além de alterações na fala e na escrita (MINISTÉRIO DA SAÚDE, 2015).

A base de dados possui 22 atributos e 195 amostras. Os atributos são: frequência fundamental vocal média; frequência fundamental vocal máxima; frequência fundamental vocal mínima; várias medidas de variação na frequência fundamental; várias medidas de variação na amplitude; medidas de relação de ruído para componentes tonais na voz; medidas de complexidade dinâmica não linear; expoente de escala de sinal fractal; medidas não lineares de variação da frequência fundamental. As classes estão relacionadas ao diagnóstico do paciente, sendo elas: saudável; portador de Parkinson.

4.2.4 Doença Cardíaca

Doença cardíaca é um termo geral para indicar condições médicas crônicas ou aguda que afetam um ou mais componentes do coração.

A base de dados possui 7 atributos e 234 amostras. Os atributos são: idade; pressão sanguínea em repouso; colesterol sérico em mg/dl; frequência cardíaca máxima alcançada; depressão induzida pelo exercício em relação ao repouso; inclinação do pico do exercício; número de vasos principais coloridos por fluoroscopia. As classes estão relacionadas à identificação de doença cardíaca, sendo elas: ausência; presença.

4.2.5 Sumário

Um resumo das bases de dados é apresentado na Tabela 4.1.

Tabela 4.1 – Informações das bases de dados

Nome	Atributos	Amostras	Classes
Flor Íris	4	150	3
Pulsares	8	527	2
Parkinson	22	195	2
Doença Cardíaca	7	234	2

Fonte: do autor

4.3 Avaliação e Comparação de Resultados

O iUP será executado com os parâmetros $\xi_{max} = 0.4$ e λ igual ao número de classes definido na base de dados.

A acurácia será usada para avaliação de modelos iUP e também para comparação de desempenho entre os algoritmos iUP, K-means e Fuzzy C-Means. O índice de desempenho é dado por

$$Acc(\%) = \frac{acertos}{acertos + erros} \cdot 100\% \tag{4.1}$$

onde acertos é a somatória de cada dado intervalar que pertence a um cluster rotulado condizente a classe que o dado é definido na base de dados considerada para gerar o fluxo de dados, i.e., a somatória dos dados classificados corretamente; erros é a somatória de cada dado intervalar que pertence a um cluster rotulado não condizente a classe que o dado é definido na base de dados considerada para gerar o fluxo de dados, i.e., a somatória dos dados classificados erroneamente.

A evolução da acurácia, do índice de validação incremental Xie-Beni e da ξ-vizinhança máxima dos grânulos para ambas as faixas consideradas para gerar os fluxos de dados intervalares – [0.05,0.1] e [0.1,0.2], serão analisadas. A rotulação dos grânulos também será apresentada.

Matriz de Confusão pode ser definida como uma tabela que apresenta o desempenho de um modelo de classificação em um conjunto de dados de teste para os quais os valores verdadeiros são conhecidos. A fim de analisar o desempenho de classificação do método proposto em cada classe, a Matriz de Confusão será utilizada nesse trabalho.

Os algoritmos iUP, K-means e Fuzzy C-Means serão executados dez vezes cada um a título de comparação. Além da acurácia, índice de validação Xie-Beni, número de grânulos e tempo de processamento estarão presentes na comparação de resultados entre os algoritmos. O índice Xie-Beni para os algoritmos K-means e Fuzzy C-Means é obtido análogo a (2.8). Ao que

se refere ao algoritmo iUP o índice Xie-Beni é o incremental, obtido análogo a (3.6). O número de grânulos é o número médio gerado, entretanto os algoritmos K-means e Fuzzy C-Means solicitam o número de classes para serem executados, logo o número de grânulos é o mesmo do número de classes da base de dados para estes.

Os algoritmos K-means e Fuzzy C-Means serão executados com os valores numéricos reais das bases de dados, com 30% de embaralhamento dos dados e sendo 70% dos dados para treino e 30% dos dados para teste.

5 RESULTADOS E DISCUSSÃO

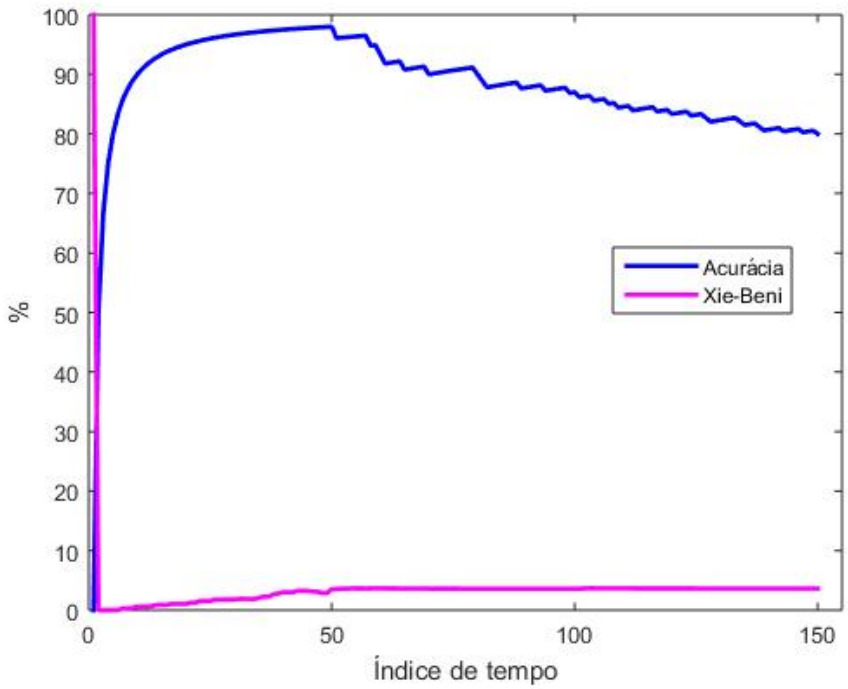
O algoritmo de clusterização intervalar incremental bottom-up, denominado iUP, proposto nesse trabalho, foi executado com fluxos de dados intervalares, obtidos das bases de dados consideradas para avaliação empírica de modelos iUP. Esses fluxos de dados intervalares foram gerados assumindo um intervalo de valores possíveis em torno dos valores numéricos reais, i.e., os atributos das bases de dados foram abertos por um valor sorteado randomicamente dentro de uma faixa para cada amostra, o que ocasionou incerteza diferente para amostras diferentes. As faixas consideradas para o sorteio randômico foram [0.05,0.1] e [0.1,0.2], sendo cada número com quatro casas decimais.

5.1 Flor Íris

5.1.1 Resultados para Faixa [0.05,0.1]

A acurácia do método proposto para o fluxo de dados Flor Íris é de 80.7% e o índice de validação incremental Xie-Beni final é 3.6268. A Figura 5.1 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Figura 5.1 – Acurácia e índice Xie-Beni para o fluxo de dados Flor Íris - primeira faixa

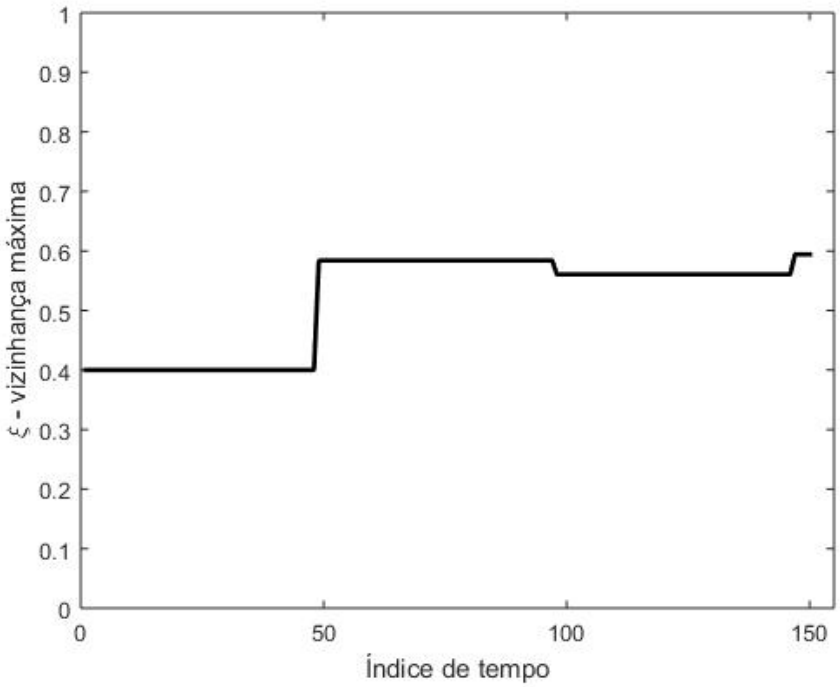


Fonte: do autor

Percebe-se pela Figura 5.1 que a acurácia do algoritmo iUP esteve entre 80% e 98% no decorrer do fluxo de dados e o índice de validação Xie-Beni evoluiu até 3.63.

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.2.

Figura 5.2 – Evolução ξ_{max} para o fluxo de dados Flor Íris - primeira faixa



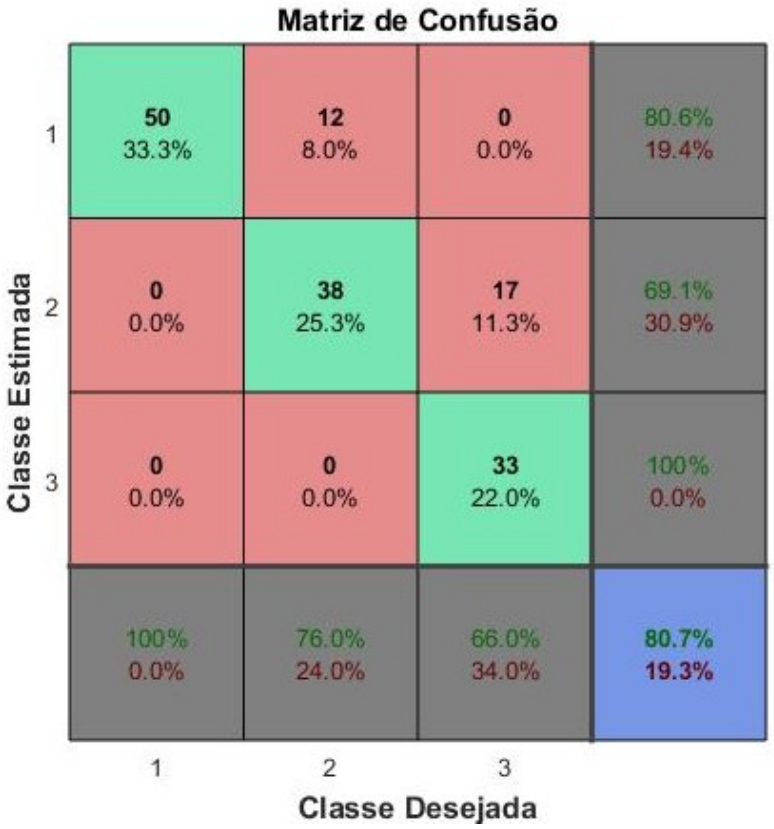
Fonte: do autor

Nota-se por meio da Figura 5.2 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 0.6. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP três vezes no decorrer do fluxo de dados.

O iUP gerou 28 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Íris Setosa é definida para os grânulos ξ^1 ao ξ^{28}, exceto ξ^{25} e ξ^{27}; classe 2 - Íris Versicolour rotula o grânulo ξ^{25}; e classe 3 - Íris Virginica é atribuída ao grânulo ξ^{27}.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Flor Íris pode ser observado na Figura 5.3.

Figura 5.3 – Matriz de Confusão para o fluxo de dados Flor Íris - primeira faixa



Fonte: do autor

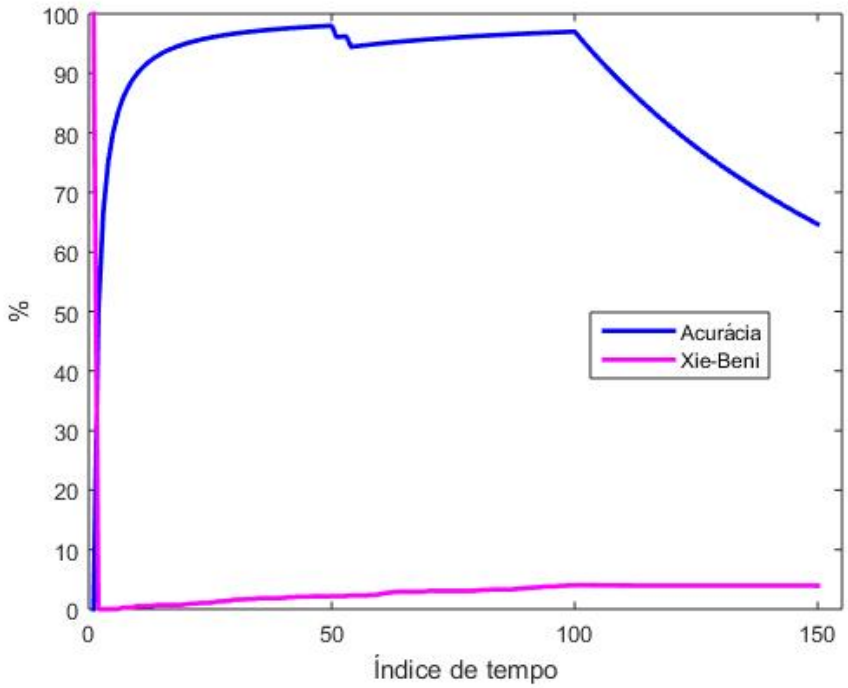
Verifica-se através da Figura 5.3 que 100% de Íris Setosa foi estimada corretamente pelo iUP; 76% de Íris Versicolour foi corretamente classificada; e 66% de Íris Virginica obteve classificação correta. Ao que se refere aos dados confundidos, 8% de Íris Versicolour foi classificada erroneamente como Íris Setosa; e 11.3% de Íris Virginica foi erroneamente estimada como Íris Versicolour.

5.1.2 Resultados para Faixa [0.1,0.2]

A acurácia do método proposto para o fluxo de dados Flor Íris é de 66% e o índice de validação incremental Xie-Beni final é 3.9399. A Figura 5.4 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Percebe-se pela Figura 5.4 que a acurácia do algoritmo iUP esteve entre 66% e 97% no decorrer do fluxo de dados e o índice de validação Xie-Beni evoluiu até 3.94.

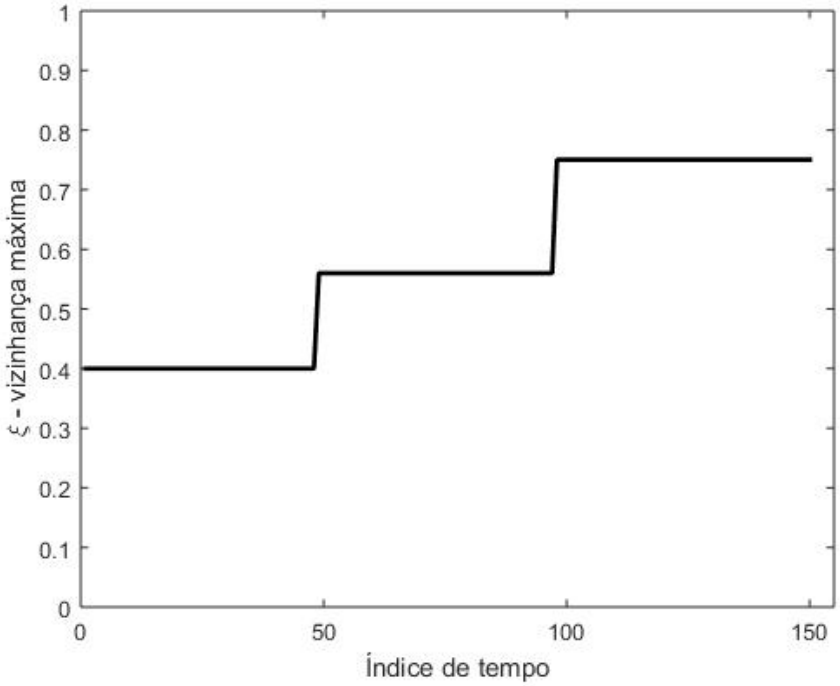
Figura 5.4 – Acurácia e índice Xie-Beni para o fluxo de dados Flor Íris - segunda faixa



Fonte: do autor

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.5.

Figura 5.5 – Evolução ξ_{max} para o fluxo de dados Flor Íris - segunda faixa



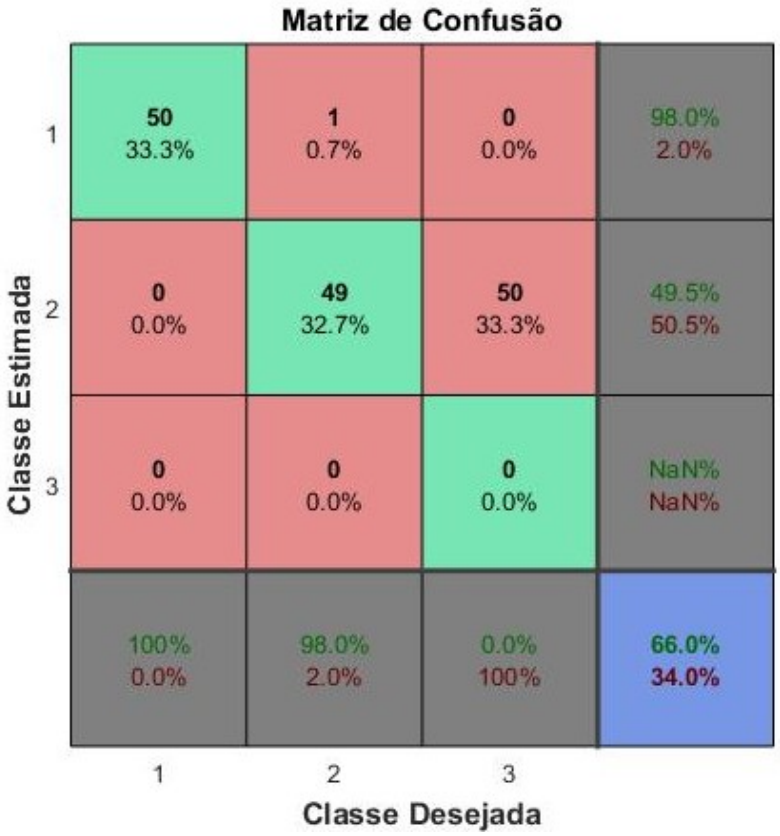
Fonte: do autor

Nota-se por meio da Figura 5.5 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 0.75. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP duas vezes no decorrer do fluxo de dados.

O iUP gerou 34 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Íris Setosa é definida para os grânulos ξ^1 ao ξ^{21}; classe 2 - Íris Versicolour rotula os grânulos ξ^{22} ao ξ^{34}; e classe 3 - Íris Virginica não foi atribuída a grânulo.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Flor Íris pode ser observado na Figura 5.6.

Figura 5.6 – Matriz de Confusão para o fluxo de dados Flor Íris - segunda faixa



Fonte: do autor

Verifica-se através da Figura 5.6 que 100% de Íris Setosa foi estimada corretamente pelo iUP; 98% de Íris Versicolour foi corretamente classificada; e 0% de Íris Virginica obteve classificação correta. Ao que se refere aos dados confundidos, 0.7% de Íris Versicolour foi classificada erroneamente como Íris Setosa; e 33.3% de Íris Virginica foi erroneamente estimada como Íris Versicolour.

5.1.3 Comparação entre Algoritmos

A comparação de desempenho entre os algoritmos iUP, K-means e Fuzzy C-Means para o fluxo de dados Flor Íris é apresentada na Tabela 5.1. Os parâmetros são acurácia em porcentagem com desvio padrão, índice de validação Xie-Beni final, número de grânulos médio e tempo de processamento em segundos.

Tabela 5.1 – Comparação dos algoritmos para o fluxo de dados Flor Íris

	Acc(%)	$\mathbf{XB_{final}}$	$\mathbf{c_{medio}}$	**t(s)**
iUP [0.05,0.1]	80.7 $\pm$0.7858	3.6268	28	0.0203
iUP [0.1,0.2]	66.0 $\pm$0.4753	3.9399	34	0.0288
K-means	48.8889 $\pm$0.9034	1.1171	3	0.3333
Fuzzy C-Means	44.4444 $\pm$0.7268	1.6257	3	0.0567

Fonte: do autor

Constata-se pela Tabela 5.1 que a maior acurácia é do algoritmo iUP para faixa [0.05,0.1]; melhor partição, ou seja, menor índice de validação Xie-Beni é do algoritmo K-means; o menor número de grânulos é do algoritmo K-means e também do algoritmo Fuzzy C-Means, que se refere ao número de classes da base de dados que os algoritmos solicitaram para execução; e o menor tempo de processamento é do algoritmo iUP para faixa [0.05,0.1].

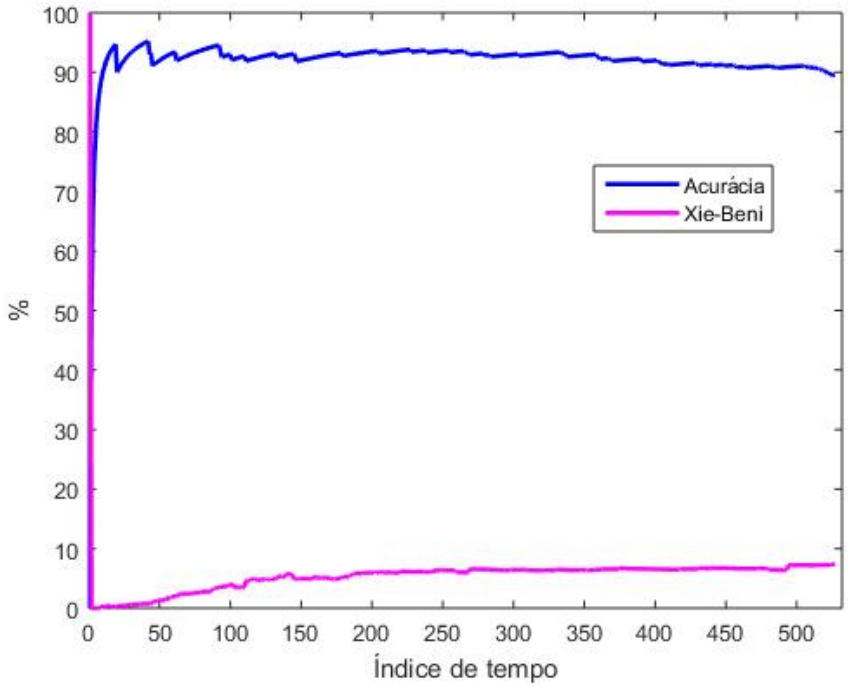
5.2 Pulsares

5.2.1 Resultados para Faixa [0.05,0.1]

A acurácia do método proposto para o fluxo de dados Pulsares é de 89.9431% e o índice de validação incremental Xie-Beni final é 3.6823. A Figura 5.7 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Percebe-se pela Figura 5.7 que a acurácia do algoritmo iUP esteve entre 89% e 95% no decorrer do fluxo de dados e o índice de validação Xie-Beni evoluiu até 3.68.

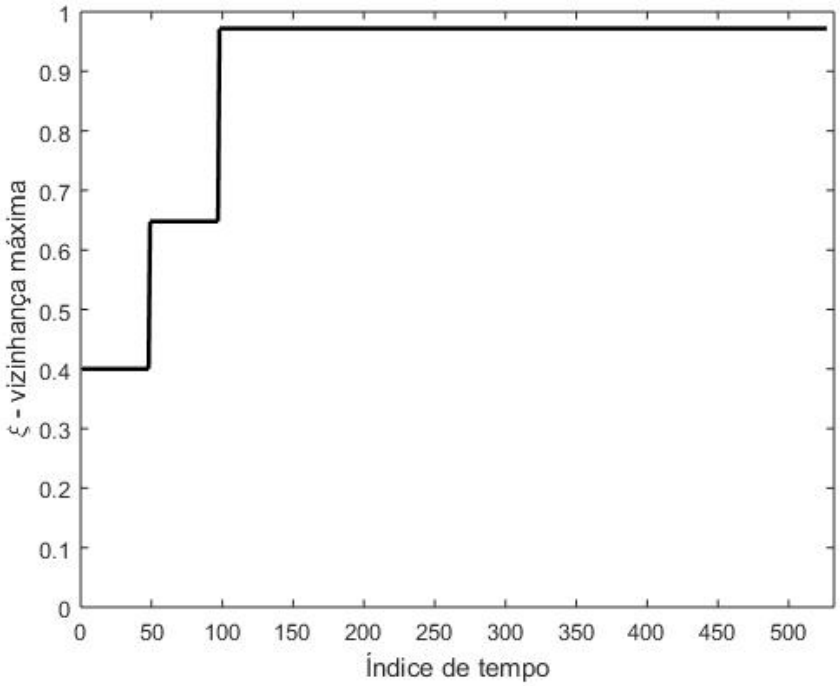
Figura 5.7 – Acurácia e índice Xie-Beni para o fluxo de dados Pulsares - primeira faixa



Fonte: do autor

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.8.

Figura 5.8 – Evolução ξ_{max} para o fluxo de dados Pulsares - primeira faixa



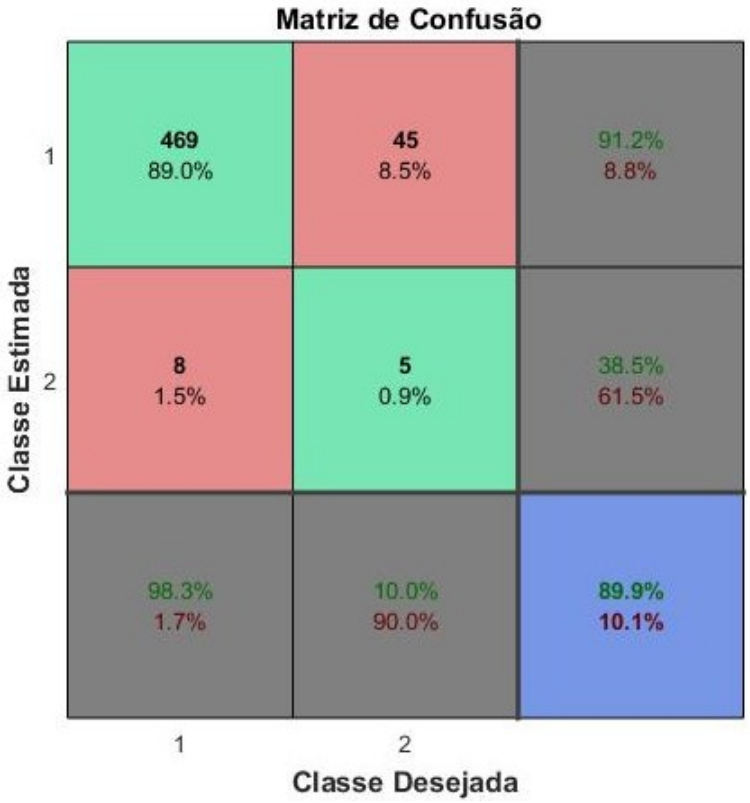
Fonte: do autor

Nota-se por meio da Figura 5.8 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 0.98. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP duas vezes no decorrer do fluxo de dados.

O iUP gerou 57 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Não é Pulsar é definida para os grânulos ξ^1 ao ξ^{57}, exceto ξ^{41}; e classe 2 - É Pulsar rotula o grânulo ξ^{41}.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Pulsares pode ser observado na Figura 5.9.

Figura 5.9 – Matriz de Confusão para o fluxo de dados Pulsares - primeira faixa



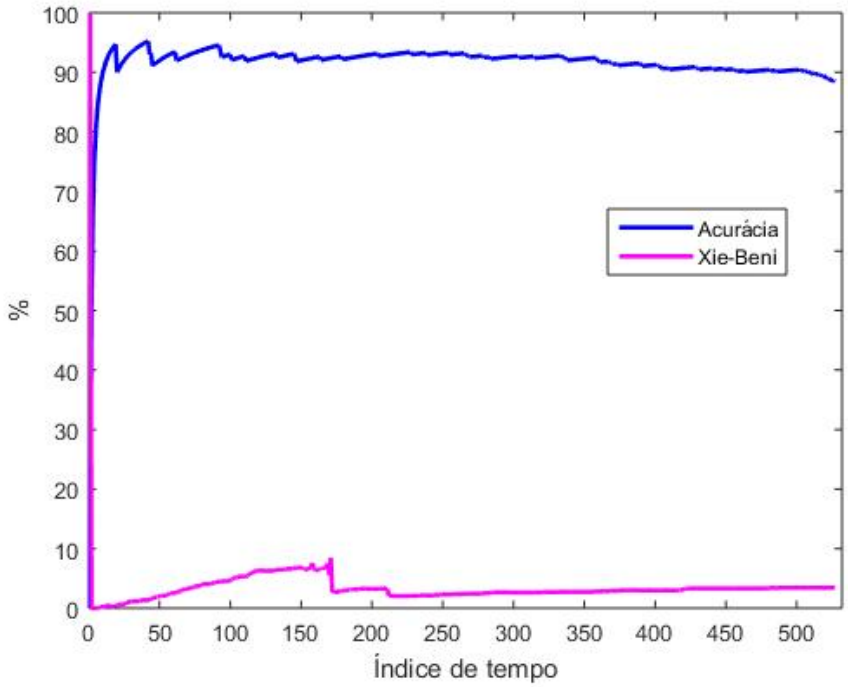
Fonte: do autor

Verifica-se através da Figura 5.9 que 98.3% de Não é Pulsar foi estimado corretamente pelo iUP; e 10% de É Pulsar foi corretamente classificado. Ao que se refere aos dados confundidos, 1.5% de Não é Pulsar foi classificado erroneamente como É Pulsar; e 8.5% de É Pulsar foi erroneamente estimado como Não é Pulsar.

5.2.2 Resultados para Faixa [0.1,0.2]

A acurácia do método proposto para o fluxo de dados Pulsares é de 88.6342% e o índice de validação incremental Xie-Beni final é 3.4214. A Figura 5.10 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Figura 5.10 – Acurácia e índice Xie-Beni para o fluxo de dados Pulsares - segunda faixa



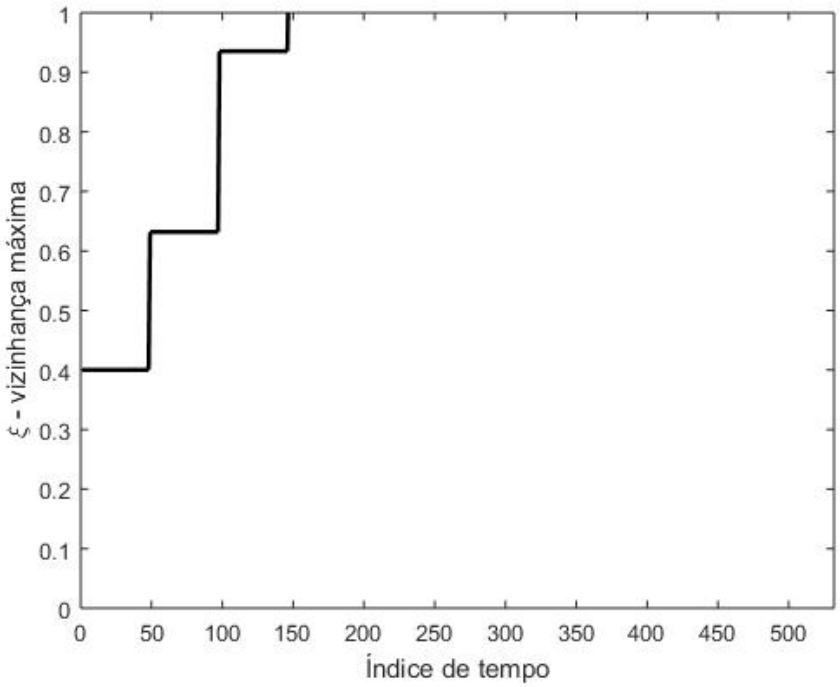
Fonte: do autor

Percebe-se pela Figura 5.10 que a acurácia do algoritmo iUP esteve entre 88% e 95% no decorrer do fluxo de dados e o índice de validação Xie-Beni foi aumentando até no decorrer de 175 amostras, então houve queda e evoluiu até 3.42.

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.11.

Nota-se por meio da Figura 5.11 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 1. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP três vezes no decorrer do fluxo de dados.

Figura 5.11 – Evolução ξ_{max} para o fluxo de dados Pulsares - segunda faixa



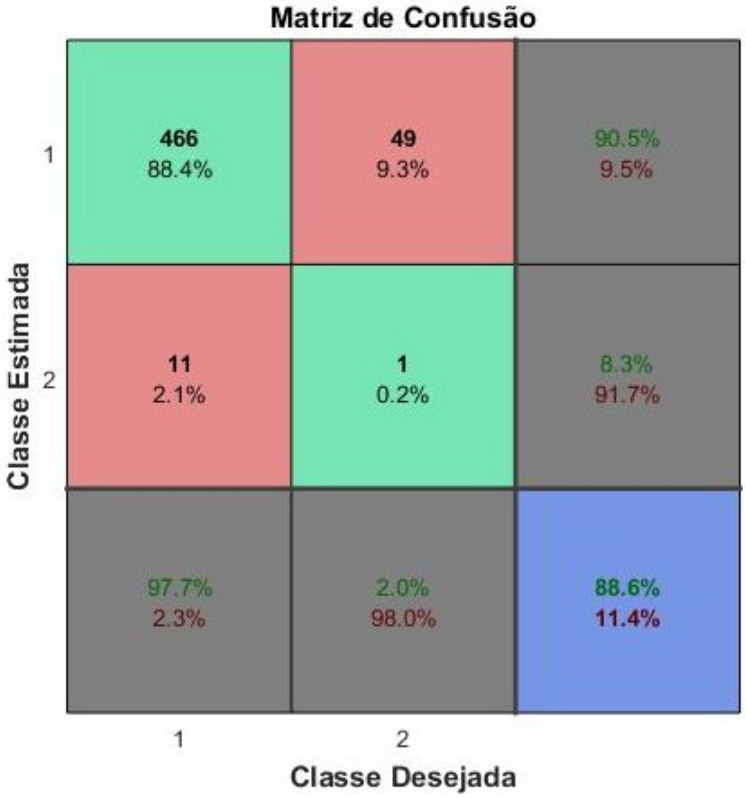
Fonte: do autor

O iUP gerou 59 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Não é Pulsar é definida para os grânulos ξ^1 ao ξ^{59}, exceto ξ^{39} e ξ^{54}; e classe 2 - É Pulsar rotula os grânulos ξ^{39} e ξ^{54}.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Pulsares pode ser observado na Figura 5.12.

Verifica-se através da Figura 5.12 que 97.7% de Não é Pulsar foi estimado corretamente pelo iUP; e 2% de É Pulsar foi corretamente classificado. Ao que se refere aos dados confundidos, 2.1% de Não é Pulsar foi classificado erroneamente como É Pulsar; e 9.3% de É Pulsar foi erroneamente estimado como Não é Pulsar.

Figura 5.12 – Matriz de Confusão para o fluxo de dados Pulsares - segunda faixa



Fonte: do autor

5.2.3 Comparação entre Algoritmos

A comparação de desempenho entre os algoritmos iUP, K-means e Fuzzy C-Means para o fluxo de dados Pulsares é apresentada na Tabela 5.2. Os parâmetros são acurácia em porcentagem com desvio padrão, índice de validação Xie-Beni final, número de grânulos médio e tempo de processamento em segundos.

Tabela 5.2 – Comparação dos algoritmos para o fluxo de dados Pulsares

	Acc(%)	**XB_{final}**	**c_{medio}**	**t(s)**
iUP [0.05,0.1]	89.9431 ±0.1228	3.6823	57	0.1277
iUP [0.1,0.2]	88.6342 ±0.1553	3.4214	59	0.1411
K-means	82.2785 ±0.3109	1.0885	2	0.3265
Fuzzy C-Means	77.8481 ±0.3601	0.3206	2	0.0629

Fonte: do autor

Constata-se pela Tabela 5.2 que a maior acurácia é do algoritmo iUP para faixa [0.05,0.1]; melhor partição, ou seja, menor índice de validação Xie-Beni é do algoritmo Fuzzy C-Means; o menor número de grânulos é do algoritmo K-means e também do algoritmo Fuzzy C-Means,

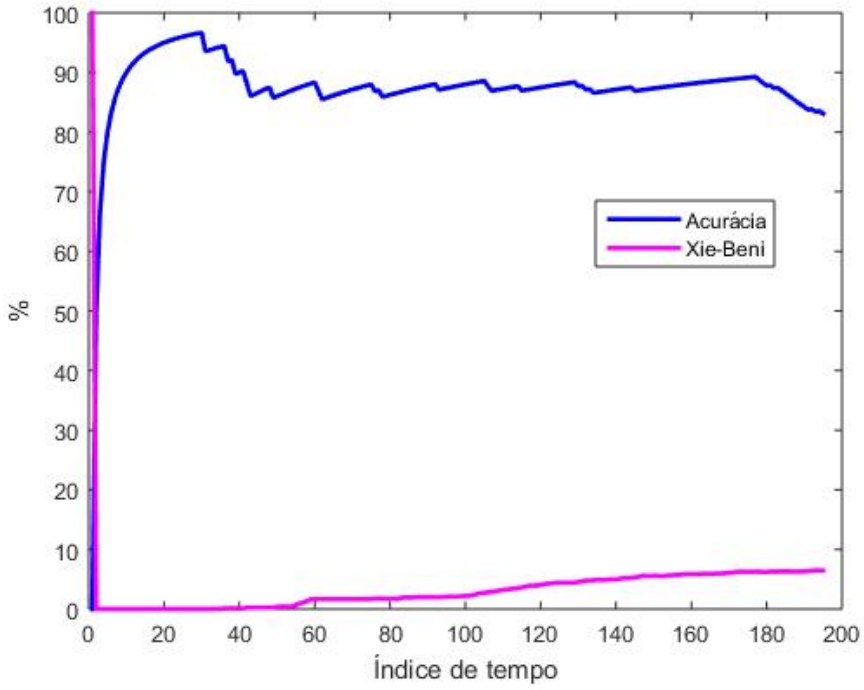
que se refere ao número de classes da base de dados que os algoritmos solicitaram para execução; e o menor tempo de processamento é do algoritmo Fuzzy C-Means.

5.3 Parkinson

5.3.1 Resultados para Faixa [0.05,0.1]

A acurácia do método proposto para o fluxo de dados Parkinson é de 83.6269% e o índice de validação incremental Xie-Beni final é 6.4650. A Figura 5.13 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Figura 5.13 – Acurácia e índice Xie-Beni para o fluxo de dados Parkinson - primeira faixa

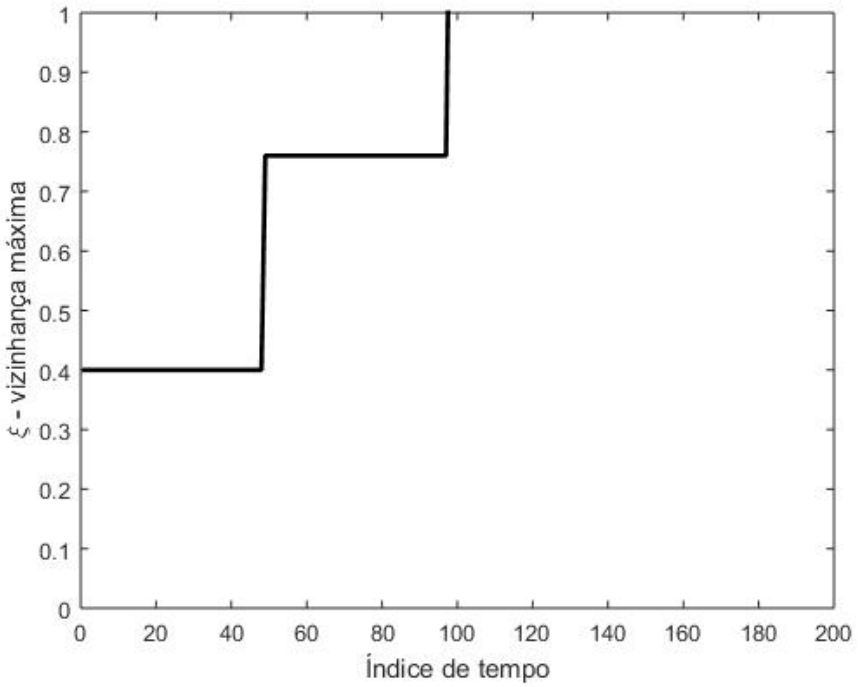


Fonte: do autor

Percebe-se pela Figura 5.13 que a acurácia do algoritmo iUP esteve entre 83% e 97% no decorrer do fluxo de dados e o índice de validação Xie-Beni evoluiu até 6.47.

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.14.

Figura 5.14 – Evolução ξ_{max} para o fluxo de dados Parkinson - primeira faixa



Fonte: do autor

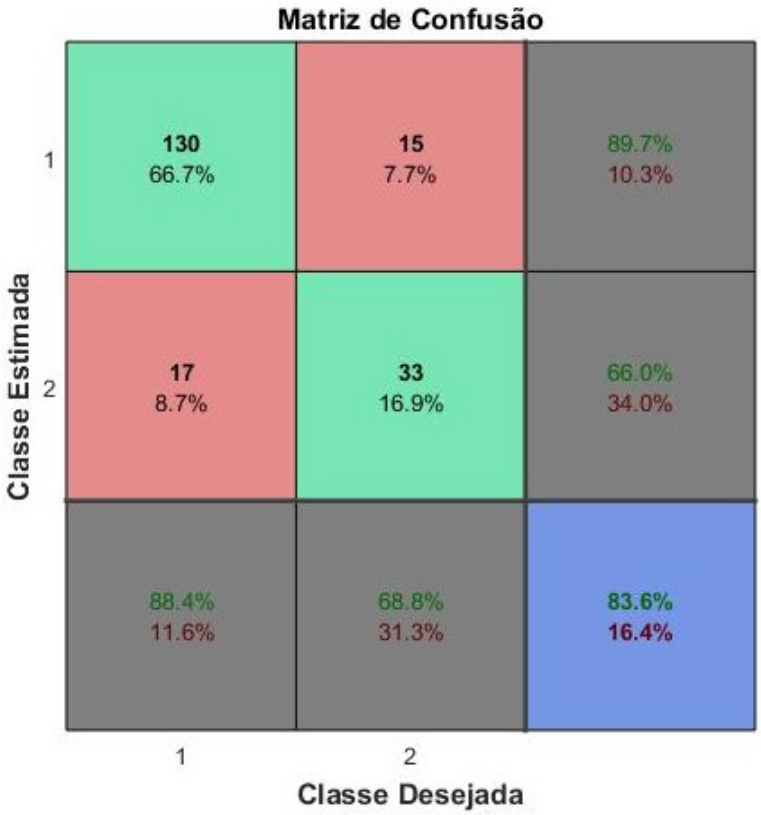
Nota-se por meio da Figura 5.14 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 1. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP duas vezes no decorrer do fluxo de dados.

O iUP gerou 77 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Saudável é definida para os grânulos ξ^1 ao ξ^{77}, exceto ξ^5, ξ^6, ξ^{19}, ξ^{21}, ξ^{52}, ξ^{69} e ξ^{71}; e classe 2 - Portador de Parkinson rotula os grânulos ξ^5, ξ^6, ξ^{19}, ξ^{21}, ξ^{52}, ξ^{69} e ξ^{71}.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Parkinson pode ser observado na Figura 5.15.

Verifica-se através da Figura 5.15 que 88.4% de Saudável foi estimado corretamente pelo iUP; e 68.8% de Portador de Parkinson foi corretamente classificado. Ao que se refere aos dados confundidos, 8.7% de Saudável foi classificado erroneamente como Portador de Parkinson; e 7.7% de Portador de Parkinson foi erroneamente estimado como Saudável.

Figura 5.15 – Matriz de Confusão para o fluxo de dados Parkinson - primeira faixa



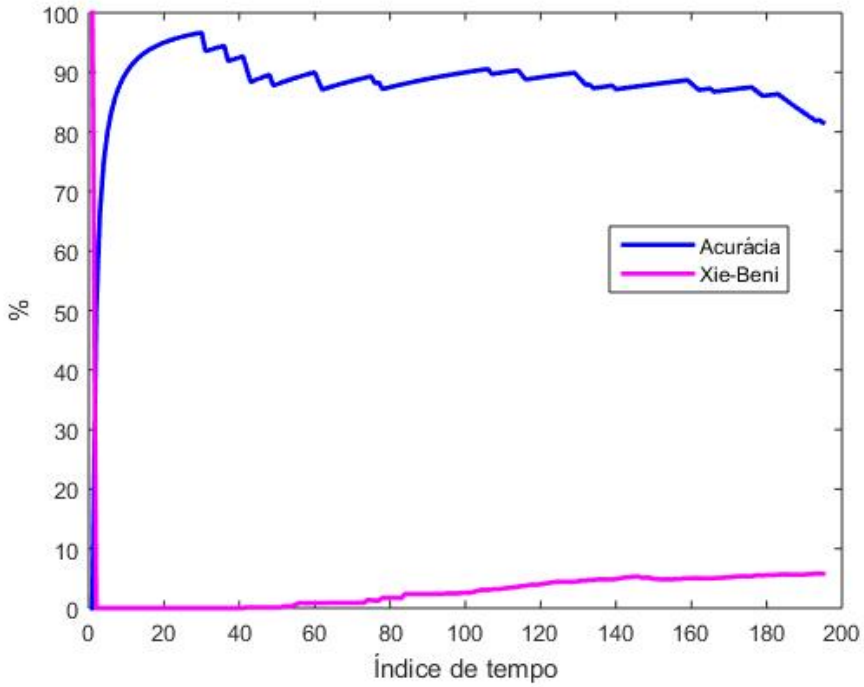
Fonte: do autor

5.3.2 Resultados para Faixa [0.1,0.2]

A acurácia do método proposto para o fluxo de dados Parkinson é de 82.1358% e o índice de validação incremental Xie-Beni final é 5.7731. A Figura 5.16 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Percebe-se pela Figura 5.16 que a acurácia do algoritmo iUP esteve entre 82% e 97% no decorrer do fluxo de dados e o índice de validação Xie-Beni evoluiu até 5.77.

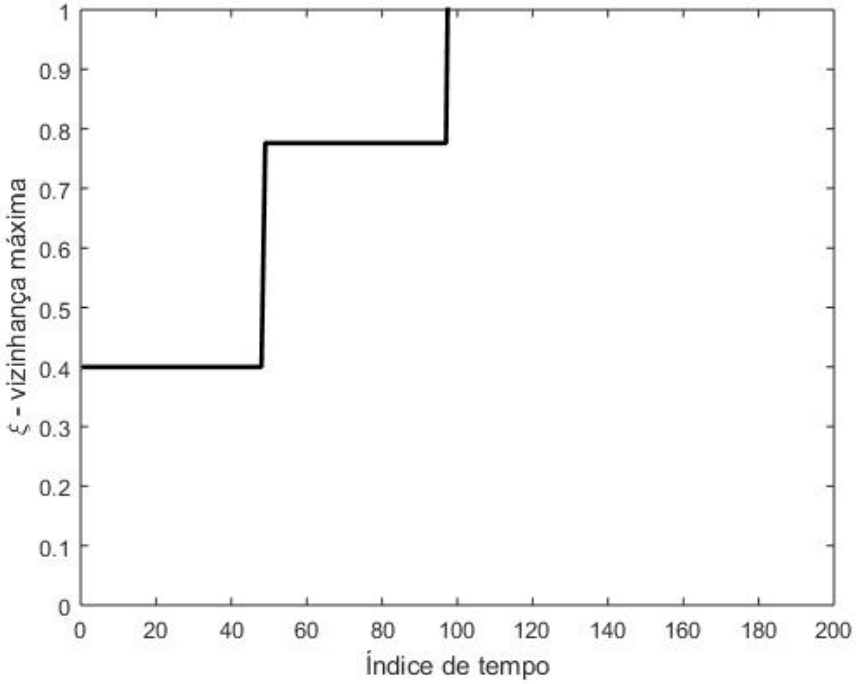
Figura 5.16 – Acurácia e índice Xie-Beni para o fluxo de dados Parkinson - segunda faixa



Fonte: do autor

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.17.

Figura 5.17 – Evolução ξ_{max} para o fluxo de dados Parkinson - segunda faixa



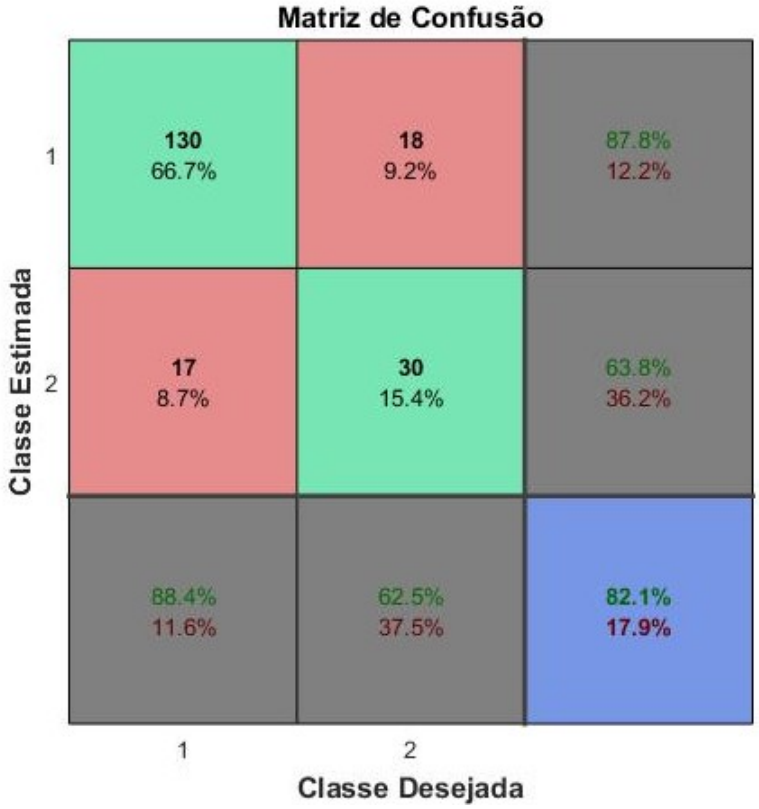
Fonte: do autor

Nota-se por meio da Figura 5.17 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 1. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP duas vezes no decorrer do fluxo de dados.

O iUP gerou 81 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Saudável é definida para os grânulos ξ^1 ao ξ^{81}, exceto ξ^{54}, ξ^{55} e ξ^{59}; e classe 2 - Portador de Parkinson rotula os grânulos ξ^{54}, ξ^{55} e ξ^{59}.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Parkinson pode ser observado na Figura 5.18.

Figura 5.18 – Matriz de Confusão para o fluxo de dados Parkinson - segunda faixa



Fonte: do autor

Verifica-se através da Figura 5.18 que 88.4% de Saudável foi estimado corretamente pelo iUP; e 62.5% de Portador de Parkinson foi corretamente classificado. Ao que se refere aos dados confundidos, 8.7% de Saudável foi classificado erroneamente como Portador de Parkinson; e 9.2% de Portador de Parkinson foi erroneamente estimado como Saudável.

5.3.3 Comparação entre Algoritmos

A comparação de desempenho entre os algoritmos iUP, K-means e Fuzzy C-Means para o fluxo de dados Parkinson é apresentada na Tabela 5.3. Os parâmetros são acurácia em porcentagem com desvio padrão, índice de validação Xie-Beni final, número de grânulos médio e tempo de processamento em segundos.

Tabela 5.3 – Comparação dos algoritmos para o fluxo de dados Parkinson

	Acc(%)	$\mathbf{XB_{final}}$	$\mathbf{c_{medio}}$	**t(s)**
iUP [0.05,0.1]	83.6269 $\pm$0.4137	6.4650	77	0.5721
iUP [0.1,0.2]	82.1358 $\pm$0.4312	5.7731	81	0.7148
K-means	62.0690 $\pm$0.3478	15.9829	2	0.3344
Fuzzy C-Means	55.1724 $\pm$0.4895	1.9672	2	0.0592

Fonte: do autor

Constata-se pela Tabela 5.3 que a maior acurácia é do algoritmo iUP para faixa [0.05,0.1]; melhor partição, ou seja, menor índice de validação Xie-Beni é do algoritmo Fuzzy C-Means; o menor número de grânulos é do algoritmo K-means e também do algoritmo Fuzzy C-Means, que se refere ao número de classes da base de dados que os algoritmos solicitaram para execução; e o menor tempo de processamento é do algoritmo Fuzzy C-Means.

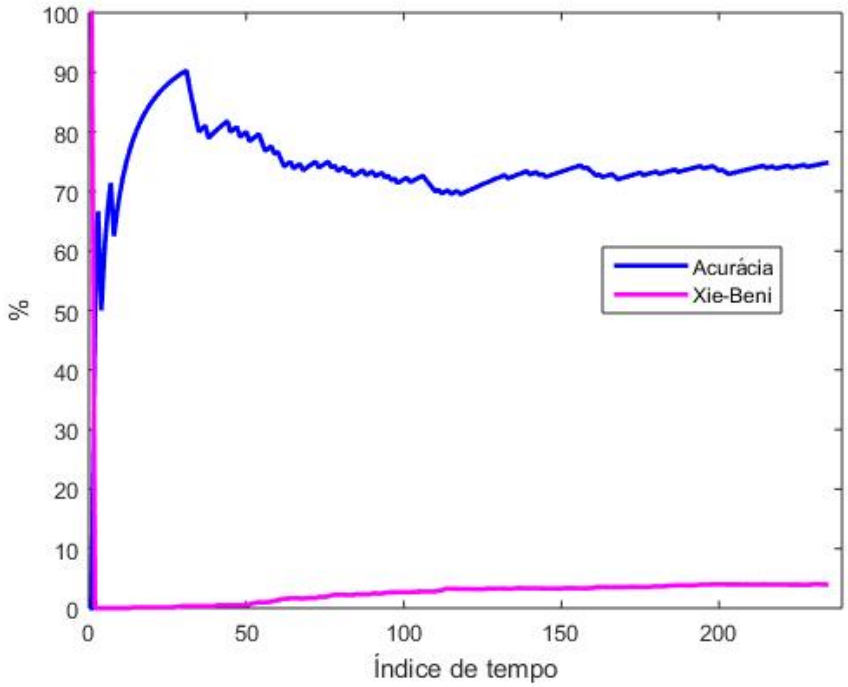
5.4 Doença Cardíaca

5.4.1 Resultados para Faixa [0.05,0.1]

A acurácia do método proposto para o fluxo de dados Doença Cardíaca é de 75.2263% e o índice de validação incremental Xie-Beni final é 3.9658. A Figura 5.19 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Percebe-se pela Figura 5.19 que a acurácia do algoritmo iUP esteve entre 75% e 90% no decorrer do fluxo de dados e o índice de validação Xie-Beni evoluiu até 3.97.

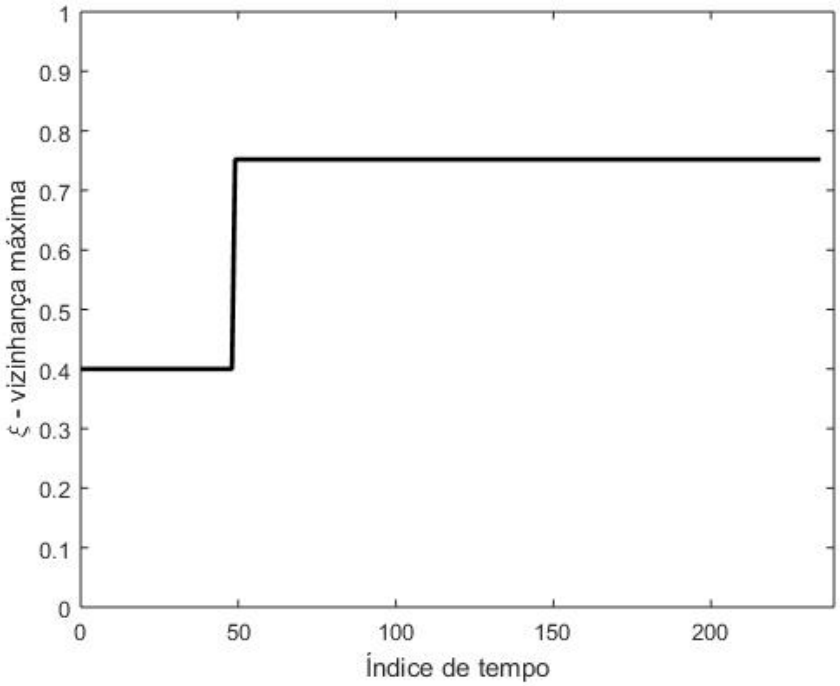
Figura 5.19 – Acurácia e índice Xie-Beni para o fluxo de dados Doença Cardíaca - primeira faixa



Fonte: do autor

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.20.

Figura 5.20 – Evolução ξ_{max} para o fluxo de dados Doença Cardíaca - primeira faixa



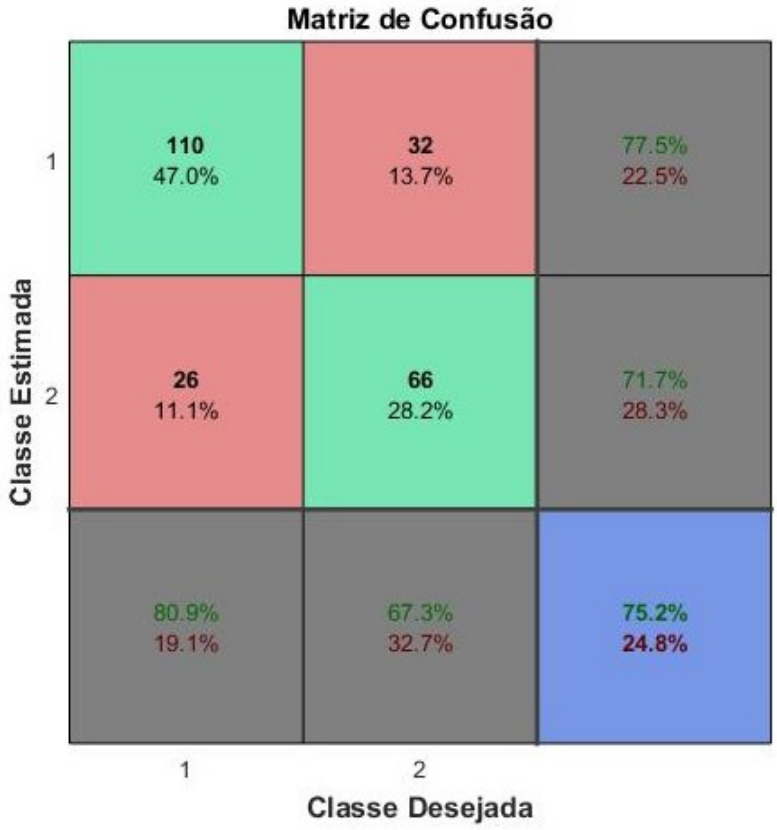
Fonte: do autor

Nota-se por meio da Figura 5.20 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 0.75. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP uma vez no decorrer do fluxo de dados.

O iUP gerou 45 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Ausência é definida para os grânulos ξ^1 ao ξ^{45}, exceto ξ^6, ξ^7, ξ^{11}, ξ^{17}, ξ^{30}, ξ^{40}, ξ^{42} e ξ^{43}; e classe 2 - Presença rotula os grânulos ξ^6, ξ^7, ξ^{11}, ξ^{17}, ξ^{30}, ξ^{40}, ξ^{42} e ξ^{43}.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Doença Cardíaca pode ser observado na Figura 5.21.

Figura 5.21 – Matriz de Confusão para o fluxo de dados Doença Cardíaca - primeira faixa



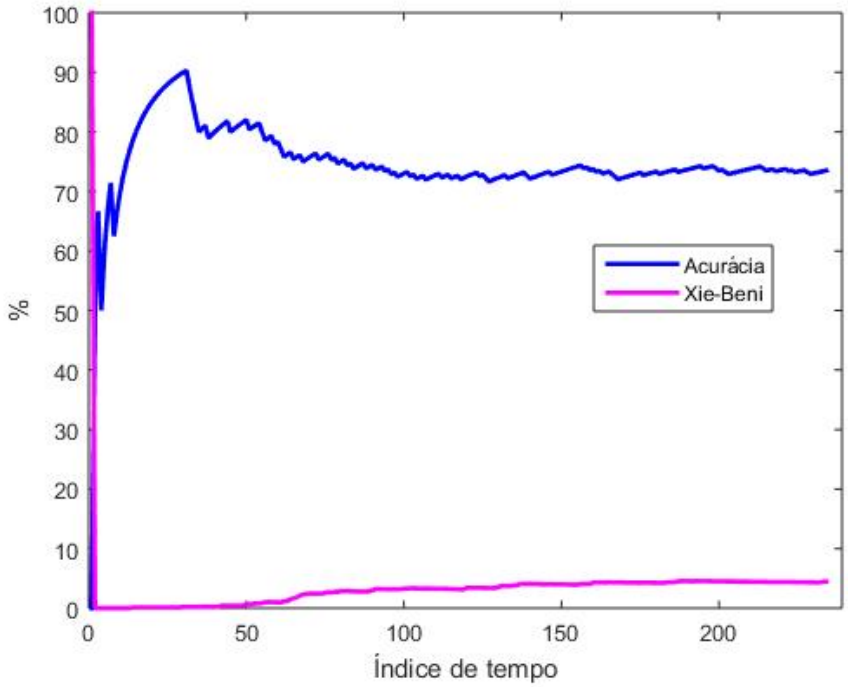
Fonte: do autor

Verifica-se através da Figura 5.21 que 80.9% de Ausência foi estimado corretamente pelo iUP; e 67.3% de Presença foi corretamente classificado. Ao que se refere aos dados confundidos, 11.1% de Ausência foi classificado erroneamente como Presença; e 13.7% de Presença foi erroneamente estimado como Ausência.

5.4.2 Resultados para Faixa [0.1,0.2]

A acurácia do método proposto para o fluxo de dados Doença Cardíaca é de 73.9134% e o índice de validação incremental Xie-Beni final é 4.4477. A Figura 5.22 permite observar esse desempenho de classificação no decorrer do fluxo de dados e a evolução do índice Xie-Beni.

Figura 5.22 – Acurácia e índice Xie-Beni para o fluxo de dados Doença Cardíaca - segunda faixa



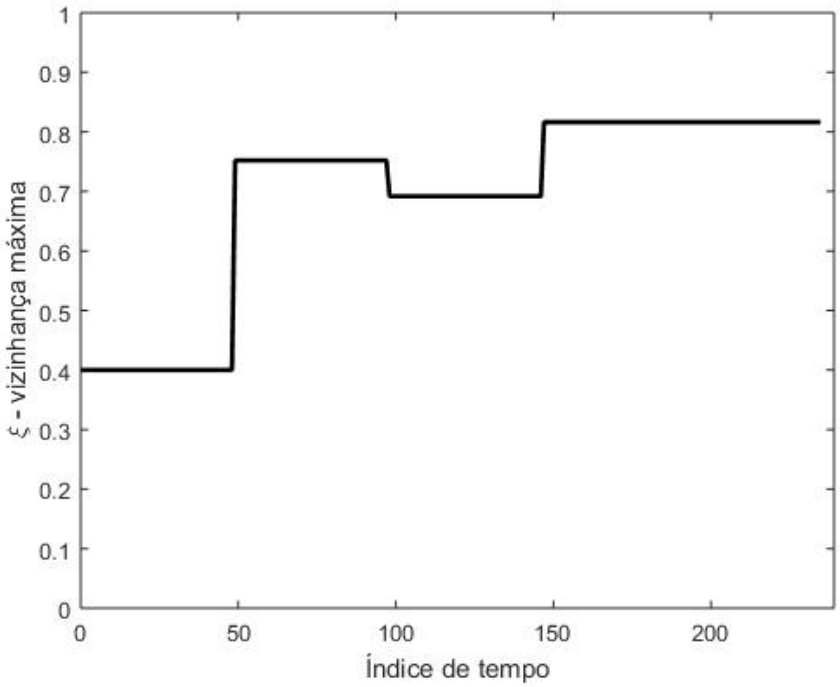
Fonte: do autor

Percebe-se pela Figura 5.22 que a acurácia do algoritmo iUP esteve entre 73% e 90% no decorrer do fluxo de dados e o índice de validação Xie-Beni evoluiu até 4.45.

A evolução referente a adaptação da ξ-vizinhança máxima dos grânulos é apresentada na Figura 5.23.

Nota-se por meio da Figura 5.23 que a ξ-vizinhança máxima dos grânulos evoluiu de 0.4 (valor *default*) até 0.82, com aumento e redução no decorrer do fluxo de dados. De acordo com os picos observados no gráfico, ξ_{max} foi adaptado pelo iUP três vezes no decorrer do fluxo de dados.

Figura 5.23 – Evolução ξ_{max} para o fluxo de dados Doença Cardíaca - segunda faixa



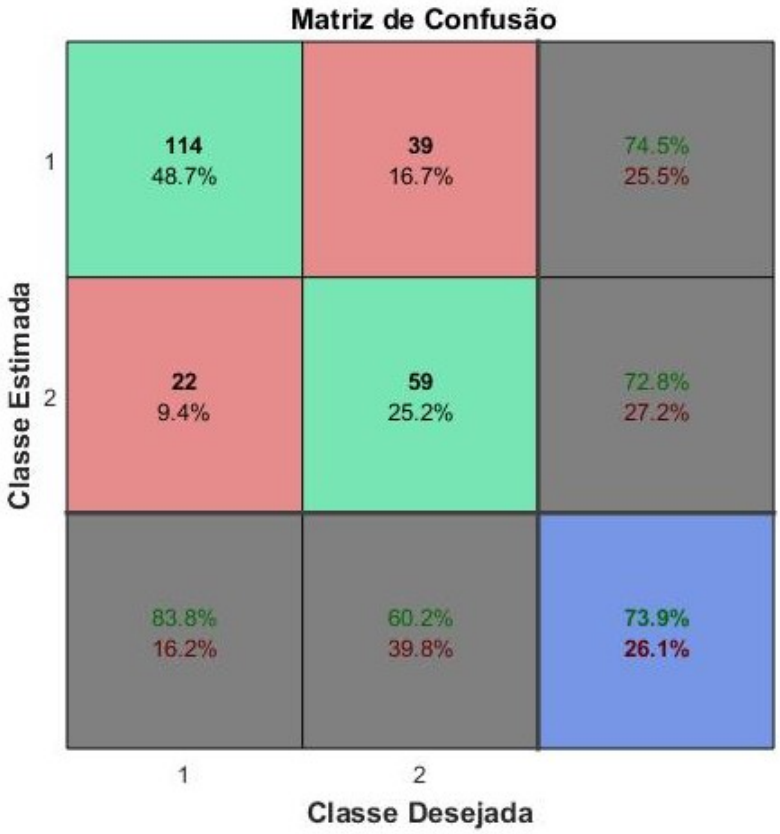
Fonte: do autor

O iUP gerou 54 grânulos. Conforme o processo de rotulação de grânulos, as classes foram atribuídas, então classe 1 - Ausência é definida para os grânulos ξ^1 ao ξ^{54}, exceto ξ^3, ξ^7, ξ^9, ξ^{14}, ξ^{28}, ξ^{29}, ξ^{33}, ξ^{46}, ξ^{47}, ξ^{48} e ξ^{50}; e classe 2 - Presença rotula os grânulos ξ^3, ξ^7, ξ^9, ξ^{14}, ξ^{28}, ξ^{29}, ξ^{33}, ξ^{46}, ξ^{47}, ξ^{48} e ξ^{50}.

O desempenho de classificação do método proposto em cada classe do fluxo de dados Doença Cardíaca pode ser observado na Figura 5.24.

Verifica-se através da Figura 5.24 que 83.8% de Ausência foi estimado corretamente pelo iUP; e 60.2% de Presença foi corretamente classificado. Ao que se refere aos dados confundidos, 9.4% de Ausência foi classificado erroneamente como Presença; e 16.7% de Presença foi erroneamente estimado como Ausência.

Figura 5.24 – Matriz de Confusão para o fluxo de dados Doença Cardíaca - segunda faixa



Fonte: do autor

5.4.3 Comparação entre Algoritmos

A comparação de desempenho entre os algoritmos iUP, K-means e Fuzzy C-Means para o fluxo de dados Doença Cardíaca é apresentada na Tabela 5.4. Os parâmetros são acurácia em porcentagem com desvio padrão, índice de validação Xie-Beni final, número de grânulos médio e tempo de processamento em segundos.

Tabela 5.4 – Comparação dos algoritmos para o fluxo de dados Doença Cardíaca

	Acc(%)	**XB_{final}**	**c_{medio}**	**t(s)**
iUP [0.05,0.1]	75.2263 ±0.4963	3.9658	45	0.0746
iUP [0.1,0.2]	73.9134 ±0.4875	4.4477	54	0.1160
K-means	50.0 ±0.4826	4.4282	2	0.3464
Fuzzy C-Means	52.8571 ±0.5036	2.4344	2	0.0564

Fonte: do autor

Constata-se pela Tabela 5.4 que a maior acurácia é do algoritmo iUP para faixa [0.05,0.1]; melhor partição, ou seja, menor índice de validação Xie-Beni é do algoritmo Fuzzy C-Means;

o menor número de grânulos é do algoritmo K-means e também do algoritmo Fuzzy C-Means, que se refere ao número de classes da base de dados que os algoritmos solicitaram para execução; e o menor tempo de processamento é do algoritmo Fuzzy C-Means.

6 CONCLUSÃO

O algoritmo de clusterização intervalar incremental bottom-up, denominado iUP, proposto nesse trabalho, apresenta flexibilidades particulares. O parâmetro número de rótulos de classe, λ, oferece flexibilidade e um mecanismo de expressão de conhecimento especialista sobre o domínio do problema, uma vez que ele pode ser escolhido ou não.

A indicação explícita do número de grânulos que uma base de dados contém é desnecessária no ambiente do algoritmo. O parâmetro ξ-vizinhança máxima dos grânulos, ξ_{max}, mantém um certo controle sobre a quantidade de grânulos que são mantidos em um modelo.

O iUP lida com dados incertos e, em particular, dados numéricos. E para rotulação (atribuição de classes) de grânulos emprega um método baseado na vizinhança dos grânulos.

O índice de validação incremental de Xie-Beni proposto, provê uma medida iterativa do desempenho do algoritmo de aprendizado e das partições do modelo iUP proposto.

O método proposto foi capaz de modelar processos complexos apresentados como um fluxo de dados e sujeitos à mudanças no ambiente ou no sistema, uma vez que lidou nos experimentos com fluxos de dados intervalares em que amostras diferentes possuíam incerteza diferente.

O algoritmo de aprendizado desenvolveu a estrutura de modelos de maneira bottom-up, sem conhecimento anterior a respeito do processo, e adaptou os parâmetros dos modelos à medida que houve necessidade, evitando assim, que o modelo fosse reconstruído e retreinado diante de mudança no ambiente ou no sistema – uma vantagem clara com relação a modelos pré-concebidos a partir de conhecimento especialista ou dados históricos.

Os experimentos mostraram que o desempenho do algoritmo é reduzido com o aumento do valor médio do tamanho de dados intervalares. A acurácia sofreu uma redução de 14.7% para o fluxo de dados Flor Íris com relação a mudança da faixa [0.05,0.1] para [0.1,0.2]; para os fluxos de dados Pulsares e Doença Cardíaca a redução foi de 1.3%; e 1.5% de redução para o fluxo de dados Parkinson.

O maior desempenho obtido pelo iUP foi a acurácia de 89.9431% do fluxo de dados Pulsares gerados com a faixa [0.05,0.1]; a melhor partição, ou seja, menor índice de validação incremental Xie-Beni foi 3.4214 desse mesmo fluxo de dados, porém com a faixa [0.1,0.2]; adicionalmente este último também foi o caso em que a ξ-vizinhança máxima dos grânulos foi adaptada mais vezes e atingiu o seu valor limite, que é 1.

Os dados mais confundidos se referem ao fluxo de dados Flor Íris com faixa [0.1,0.2] no qual 33.3% da classe Íris Virginica foi erroneamente estimada como Íris Versicolour; seguida pelo fluxo de dados Doença Cardíaca com faixa [0.1,0.2] no qual 16.7% da classe Presença foi erroneamente estimada como Ausência.

A comparação entre os algoritmos iUP, K-means e Fuzzy C-Means mostrou que o iUP gerou um número superior de grânulos, entretanto obteve melhor desempenho de classificação; Fuzzy C-Means obteve a melhor partição em três dos quatro casos, exceto para o fluxo de dados Flor Íris que foi o K-means; o mesmo é válido para o tempo de processamento, exceto para o fluxo de dados Flor Íris que foi o iUP.

O processamento de dados demais, como dados de satélites, aplicativos móveis conectados a nuvem, gps, é para o método proposto uma limitação, uma vez que nesses casos, os dados serão bufferizados e perdidos se não houver técnicas adicionais ou máquinas adicionais.

Para trabalhos futuros o objetivo é desenvolver uma técnica para tratar o desbalanceamento das amostras em conjunto com o iUP, podendo usar a incerteza para ponderar atributos mais e menos relevantes.

REFERÊNCIAS

AGGARWAL, C. C.; PHILIP, S. Y. A survey of uncertain data algorithms and applications. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 21, n. 5, p. 609–623, 2009.

AGRAWAL, R.; SRIKANT, R. Privacy-preserving data mining. In: **Proceedings of the 2000 ACM SIGMOD international conference on Management of data**. [S.l.: s.n.], 2000. p. 439–450.

AL-HMOUZ, R. et al. Granular description of data in a non-stationary environment. **Soft Computing**, Springer, v. 22, n. 2, p. 523–540, 2018.

ANGELOV, P. **Autonomous Learning Systems: From Data Streams to Knowledge in Real-time**. [S.l.]: Wiley, 2013.

ANGELOV, P.; FILEV, D. P.; KASABOV, N. **Evolving intelligent systems: methodology and applications**. [S.l.]: John Wiley & Sons, 2010. v. 12.

ANGELOV, P.; ZHOU, X. On line learning fuzzy rule-based system structure from data streams. In: IEEE. **Fuzzy Systems, 2008. FUZZ-IEEE 2008.(IEEE World Congress on Computational Intelligence). IEEE International Conference on**. [S.l.], 2008. p. 915–922.

ANGELOV, P. P.; ZHOU, X. Evolving fuzzy-rule-based classifiers from data streams. **IEEE Transactions on Fuzzy Systems**, IEEE, v. 16, n. 6, p. 1462–1475, 2008.

ANKERST, M. et al. Optics: ordering points to identify the clustering structure. In: ACM. **ACM Sigmod record**. [S.l.], 1999. v. 28, n. 2, p. 49–60.

BABCOCK, B. et al. Models and issues in data stream systems. In: ACM. **Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems**. [S.l.], 2002. p. 1–16.

BEZDEK, J. C. Objective function clustering. In: **Pattern recognition with fuzzy objective function algorithms**. [S.l.]: Springer, 1981. p. 43–93.

BIFET, A.; GAVALDÀ, R. Adaptive learning from evolving data streams. In: SPRINGER. **International Symposium on Intelligent Data Analysis**. [S.l.], 2009. p. 249–260.

BIFET, A. et al. Moa: Massive online analysis. **Journal of Machine Learning Research**, v. 11, n. May, p. 1601–1604, 2010.

BODYANSKIY, Y. et al. Fast learning algorithm for deep evolving gmdh-svm neural network in data stream mining tasks. In: IEEE. **Data Stream Mining & Processing (DSMP), IEEE First International Conference on**. [S.l.], 2016. p. 257–262.

CARVALHO, F. d. A. D.; TENÓRIO, C. P. Fuzzy k-means clustering algorithms for interval-valued data based on adaptive quadratic distances. **Fuzzy Sets and Systems**, Elsevier, v. 161, n. 23, p. 2978–2999, 2010.

CASTELLA, J.-C. et al. Combining top-down and bottom-up modelling approaches of land use/cover change to support public policies: Application to sustainable management of natural resources in northern vietnam. **Land use policy**, Elsevier, v. 24, n. 3, p. 531–545, 2007.

CHEN, C. et al. A novel bottom-up saliency detection method for video with dynamic background. **IEEE Signal Processing Letters**, IEEE, v. 25, n. 2, p. 154–158, 2018.

CORDOVIL, L. A. Q. et al. Uncertain data modeling based on evolving ellipsoidal fuzzy information granules. **IEEE Transactions on Fuzzy Systems**, IEEE, 2019.

DENG, T. et al. Where does the driver look? top-down-based saliency detection in a traffic driving environment. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 17, n. 7, p. 2051–2062, 2016.

DIAZ-CHITO, K. et al. Incremental model learning for spectroscopy-based food analysis. **Chemometrics and Intelligent Laboratory Systems**, Elsevier, v. 167, p. 123–131, 2017.

DORA, S. et al. Development of a self-regulating evolving spiking neural network for classification problem. **Neurocomputing**, Elsevier, v. 171, p. 1216–1229, 2016.

DUARTE, J.; GAMA, J.; BIFET, A. Adaptive model rules from high-speed data streams. **ACM Transactions on Knowledge Discovery from Data (TKDD)**, ACM, v. 10, n. 3, p. 30, 2016.

EVERITT, B. et al. **Cluster analysis: Wiley series in probability and statistics**. [S.l.]: Wiley Chichester, 2011.

FADAEE, M.; RADZI, M. Multi-objective optimization of a stand-alone hybrid renewable energy system by using evolutionary algorithms: A review. **Renewable and sustainable energy reviews**, Elsevier, v. 16, n. 5, p. 3364–3369, 2012.

FISHER, R. Linear discriminant analysis. **Ann. Eugenics**, v. 7, p. 179, 1936.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. **The elements of statistical learning**. [S.l.]: Springer series in statistics New York, 2001. v. 1.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. Regularization paths for generalized linear models via coordinate descent. **Journal of statistical software**, NIH Public Access, v. 33, n. 1, p. 1, 2010.

GAMA, J.; RODRIGUES, P. P. An overview on mining data streams. In: **Foundations of Computational, Intelligence Volume 6**. [S.l.]: Springer, 2009. p. 29–45.

GAMA, J. et al. A survey on concept drift adaptation. **ACM computing surveys (CSUR)**, ACM, v. 46, n. 4, p. 44, 2014.

GARCIA, C.; LEITE, D.; SKRJANC, I. Incremental missing-data imputation for evolving fuzzy granular prediction. **IEEE Transactions on Fuzzy Systems**, IEEE, 2019.

GIRAUD-CARRIER, C. A note on the utility of incremental learning. **Ai Communications**, IOS Press, v. 13, n. 4, p. 215–223, 2000.

GU, B. et al. Incremental support vector learning for ordinal regression. **IEEE Transactions on Neural networks and learning systems**, IEEE, v. 26, n. 7, p. 1403–1416, 2015.

GUHA, S. et al. Clustering data streams: Theory and practice. **IEEE transactions on knowledge and data engineering**, IEEE, v. 15, n. 3, p. 515–528, 2003.

GUHA, S.; RASTOGI, R.; SHIM, K. Cure: an efficient clustering algorithm for large databases. In: ACM. **ACM Sigmod Record**. [S.l.], 1998. v. 27, n. 2, p. 73–84.

HAN, J.; PEI, J.; KAMBER, M. **Data mining: concepts and techniques**. [S.l.]: Elsevier, 2011.

HU, C. et al. A novel random forests based class incremental learning method for activity recognition. **Pattern Recognition**, Elsevier, v. 78, p. 277–290, 2018.

IBRAHIM, O. A.; KELLER, J. M.; BEZDEK, J. C. Analysis of streaming clustering using an incremental validity index. In: IEEE. **2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)**. [S.l.], 2018. p. 1–8.

JAJUGA, K.; SOKOLOWSKI, A.; BOCK, H.-H. **Classification, clustering, and data analysis: recent advances and applications**. [S.l.]: Springer Science & Business Media, 2012.

KASABOV, N. **Evolving Connectionist Systems: The Knowledge Engineering Approach**. [S.l.]: Springer, 2 ed., 2007.

KHUAT, T. T.; CHEN, F.; GABRYS, B. An effective multi-resolution hierarchical granular representation based classifier using general fuzzy min-max neural network. **arXiv preprint arXiv:1905.12170**, 2019.

KLAPPER, A. et al. The control of automatic imitation based on bottom–up and top–down cues to animacy: Insights from brain and behavior. **Journal of Cognitive Neuroscience**, MIT Press, 2014.

KOURTELLIS, N. et al. Vht: Vertical hoeffding tree. In: IEEE. **Big Data (Big Data), 2016 IEEE International Conference on**. [S.l.], 2016. p. 915–922.

KRAWCZYK, B.; WOŹNIAK, M. One-class classifiers with incremental learning and forgetting for data streams with concept drift. **Soft Computing**, Springer, v. 19, n. 12, p. 3387–3400, 2015.

KUNCHEVA, L. I. **Combining pattern classifiers: methods and algorithms**. [S.l.]: John Wiley & Sons, 2014.

LEITE, D. Evolving granular systems= sistemas granulares evolutivos. PhD Dissertation, University of Campinas, 2012.

LEITE, D. et al. Optimal rule-based granular systems from data streams. **IEEE Transactions on Fuzzy Systems**, IEEE, 2019.

LEITE, D. et al. Evolving fuzzy granular modeling from nonstationary fuzzy data streams. **Evolving Systems**, Springer, v. 3, n. 2, p. 65–79, 2012.

LEITE, D.; COSTA, P.; GOMIDE, F. Evolving granular neural network for semi-supervised data stream classification. In: IEEE. **Neural Networks (IJCNN), The 2010 International Joint Conference on**. [S.l.], 2010. p. 1–8.

LEITE, D.; COSTA, P.; GOMIDE, F. Interval approach for evolving granular system modeling. **Learning in non-stationary environments, Springer, New York, NY**, v. 1, p. 271–300, 2012.

LEITE, D.; COSTA, P.; GOMIDE, F. Evolving granular neural networks from fuzzy data streams. **Neural Networks**, Elsevier, v. 38, p. 1–16, 2013.

LEITE, D.; COSTA, P.; GOMIDE, F. Evolving granular neural networks from fuzzy data streams. **Neural Networks**, v. 38, p. 1–16, 2013.

LEITE, D. et al. Evolving granular fuzzy model-based control of nonlinear dynamic systems. **IEEE Transactions on Fuzzy Systems**, IEEE, v. 23, n. 4, p. 923–938, 2014.

LI, Z. et al. A new clustering algorithm for processing gps-based road anomaly reports with a mahalanobis distance. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 18, n. 7, p. 1980–1988, 2017.

LIANG, K. et al. A new adaptive contrast enhancement algorithm for infrared images based on double plateaus histogram equalization. **Infrared Physics & Technology**, Elsevier, v. 55, n. 4, p. 309–315, 2012.

LUGHOFER, E. **Evolving Fuzzy Systems: Methodologies, Advanced Concepts and Applications**. [S.l.]: Springer: Verlag Berlin Heidelberg, 2011.

LUGHOFER, E.; ANGELOV, P. Handling drifts and shifts in on-line data streams with evolving fuzzy systems. **Applied Soft Computing**, v. 11, n. 2, p. 2057–2068, 2011.

LUGHOFER, E.; PRATAMA, M.; SKRJANC, I. Incremental rule splitting in generalized evolving fuzzy systems for autonomous drift compensation. **IEEE Transactions on Fuzzy Systems**, IEEE, v. 26, n. 4, p. 1854–1865, 2018.

LUGHOFER, E. D. Flexfis: A robust incremental learning approach for evolving takagi–sugeno fuzzy models. **IEEE Transactions on fuzzy systems**, IEEE, v. 16, n. 6, p. 1393–1410, 2008.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. **Proceedings of the fifth Berkeley symposium on mathematical statistics and probability**. [S.l.], 1967. v. 1, n. 14, p. 281–297.

MENA-TORRES, D.; AGUILAR-RUIZ, J. S. A similarity-based approach for data stream classification. **Expert Systems with Applications**, Elsevier, v. 41, n. 9, p. 4224–4234, 2014.

MINISTÉRIO DA SAÚDE. **Doença de Parkinson**. Brasil, 2015. Disponível em: <http://bvsms.saude.gov.br>.

MORALES, G. D. F.; BIFET, A. Samoa: scalable advanced massive online analysis. **Journal of Machine Learning Research**, v. 16, n. 1, p. 149–153, 2015.

MOSHTAGHI, M. et al. Evolving fuzzy rules for anomaly detection in data streams. **IEEE Transactions on Fuzzy Systems**, IEEE, v. 23, n. 3, p. 688–700, 2015.

MOSHTAGHI, M.; LECKIE, C.; BEZDEK, J. C. Online clustering of multivariate time-series. In: SIAM. **Proceedings of the 2016 SIAM International Conference on Data Mining**. [S.l.], 2016. p. 360–368.

PARISIEN, C.; FAZLY, A.; STEVENSON, S. An incremental bayesian model for learning syntactic categories. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the twelfth conference on computational natural language learning**. [S.l.], 2008. p. 89–96.

PEDRYCZ, W. **Computational intelligence: an introduction**. [S.l.]: CRC press, 1997.

PEDRYCZ, W.; GOMIDE, F. **Fuzzy systems engineering: toward human-centric computing**. [S.l.]: John Wiley & Sons, 2007.

PEDRYCZ, W.; HOMENDA, W. Building the fundamentals of granular computing: A principle of justifiable granularity. **Appl Soft Comput**, v. 13, n. 10, p. 4209–4218, 2013.

RUBIO, J. de J.; PÉREZ-CRUZ, J. H. Evolving intelligent system for the modelling of nonlinear systems with dead-zone input. **Applied Soft Computing**, Elsevier, v. 14, p. 289–304, 2014.

RUDIN, W. **Principles of mathematical analysis**. [S.l.]: McGraw-Hill, 1953.

SENGE, R.; HÜLLERMEIER, E. Top-down induction of fuzzy pattern trees. **IEEE Transactions on Fuzzy Systems**, IEEE, v. 19, n. 2, p. 241–252, 2011.

SHAKER, A.; HÜLLERMEIER, E. Iblstreams: a system for instance-based classification and regression on data streams. **Evolving Systems**, Springer, v. 3, n. 4, p. 235–249, 2012.

SHVACHKO, K. et al. The hadoop distributed file system. In: IEEE. **Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on**. [S.l.], 2010. p. 1–10.

SILVA, A. M. et al. A fast learning algorithm for evolving neo-fuzzy neuron. **Applied Soft Computing**, Elsevier, v. 14, p. 194–209, 2014.

SKRJANC, I. ět al. Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey. **Info Sci**, Submitted, n. 5, p. 55p, 2019.

SOARES, E. et al. Ensemble of evolving data clouds and fuzzy models for weather time series prediction. **Applied Soft Computing**, Elsevier, 2017.

SOCIEDADE BRASILEIRA DE CARDIOLOGIA. **Coração**. Brasil, 2018. Disponível em: <http://portal.cardiol.br>.

SUN, Y. et al. Deep learning face representation by joint identification-verification. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2014. p. 1988–1996.

TSYMBAL, A. The problem of concept drift: definitions and related work. **Computer Science Department, Trinity College Dublin**, Citeseer, v. 106, n. 2, 2004.

WANG, C.-M.; HUANG, Y.-F. Self-adaptive harmony search algorithm for optimization. **Expert Systems with Applications**, Elsevier, v. 37, n. 4, p. 2826–2837, 2010.

WANG, L. et al. Combining bottom-up and top-down segmentation: A way to realize high-performance organic circuit. **IEEE Electron Device Letters**, IEEE, v. 36, n. 7, p. 684–686, 2015.

WEN, X. et al. A rapid learning algorithm for vehicle classification. **Information Sciences**, Elsevier, v. 295, p. 395–406, 2015.

XU, R.; WUNSCH, D. Survey of clustering algorithms. **IEEE Transactions on neural networks**, Ieee, v. 16, n. 3, p. 645–678, 2005.

XU, S.; WANG, J. A fast incremental extreme learning machine algorithm for data streams classification. **Expert Systems with Applications**, Elsevier, v. 65, p. 332–344, 2016.

XUE, Y. et al. A self-adaptive artificial bee colony algorithm based on global best for global optimization. **Soft Computing**, Springer, p. 1–18, 2018.

YAO, J. T.; VASILAKOS, A. V.; PEDRYCZ, W. Granular computing: Perspectives and challenges. **IEEE Trans Cybern**, v. 43, n. 6, p. 1977–1989, 2013.

ZADEH, L. Some reflections on soft computing, granular computing, and their role in the conception, design, and utilization of information/intelligent systems. **Soft Computing**, v. 2, n. 1, p. 23–25, 1998.

ZADEH, L. Generalized theory of uncertainty (gtu) - principal concepts and ideas. **Comput Stat Data An**, v. 51, n. 1, p. 15–46, 2006.

ZHAO, Z. et al. Cognitive radio spectrum allocation using evolutionary algorithms. **IEEE Transactions on Wireless Communications**, IEEE, v. 8, n. 9, p. 4421–4425, 2009.

ZHOU, A. et al. Multiobjective evolutionary algorithms: A survey of the state of the art. **Swarm and Evolutionary Computation**, Elsevier, v. 1, n. 1, p. 32–49, 2011.