UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

Rodrigo Amador Coelho

Detectores de Mudança de Conceito por meio do Mapeamento Espacial do Fluxo de Dados usando Quadtree

Belo Horizonte
2022

Rodrigo Amador Coelho

Detectores de Mudança de Conceito por meio do Mapeamento Espacial do Fluxo de Dados usando Quadtree

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do Título de Doutor em Engenharia Elétrica.

Orientador: Cristiano Leite de Castro

Belo Horizonte

2022

Coelho, Rodrigo Amador.
C672d Detectores de mudança de conceito por meio do mapeamento espacial do fluxo de dados usando Quadtree [recurso eletrônico] / Rodrigo Amador Coelho. - 2022.
1 recurso online (91 f. : il., color.) : pdf.

Orientador: Cristiano Leite de Castro.

Tese (doutorado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Bibliografia: f.87-91.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Fluxo de dados (Computação) – Teses. 3. Algoritmos – Teses. I. Castro, Cristiano Leite de.
II. Universidade Federal de Minas Gerais. Escola de Engenharia.
III. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pela Bibliotecária Letícia Alves Vieira - CRB-6/2337
Biblioteca Prof. Mário Werneck - Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

FOLHA DE APROVAÇÃO

"DETECTORES DE MUDANÇA DE CONCEITO POR MEIO DO MAPEAMENTO ESPACIAL DO FLUXO DE DADOS USANDO QUADTREE"

RODRIGO AMADOR COELHO

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Aprovada em 29 de setembro de 2022. Por:

Prof. Dr. Cristiano Leite de Castro - Orientador ((UFMG))

Prof. Dr. Antônio de Pádua Braga

Prof. Dr. Roberto Souto Maior de Barros

Prof. Dr. Luis Enrique Zárate Gálvez

Prof. Dr. Luiz Carlos Bambirra Torres

Documento assinado eletronicamente por **Frederico Gadelha Guimaraes**, **Coordenador(a) de curso de pós-graduação**, em 04/11/2022, às 14:35, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1541960** e o código CRC **74F99F0F**.

Referência: Processo nº 23072.235841/2022-10 SEI nº 1541960

Dedico este trabalho aos meus pais e irmão

Agradecimentos

Agradeço a Deus por iluminar meu caminho, por reforçar minha esperança a cada dia. Aos meus pais Salustriano Gonçalves Coelho e Gilca Amador Coelho pelo apoio e acreditarem no meu esforço. Ao meu irmão Tiago Amador Coelho que esteve sempre presente nesta caminhada. Sem vocês, tudo isso não seria possível.

Agradeço ao orientador Cristiano Leite de Castro e o professor Luiz Carlos Bambirra Torres pelos ensinamentos e suporte junto ao desenvolvimento deste trabalho. Foram muitas horas em reuniões, novas ideias e soluções alcançadas. Agradeço também ao Álvaro C. Lemos Neto e a Thaís Macela de Lira Menegaldi pela amizade e esforços empregados para a realização de nossos trabalhos.

Aos professores, amigos e membros do MINDS (Machine Intelligence and Data Science Lab.) e LITC (Computational Intelligence Laboratory) pelo companheirismo e apoio. Aos funcionários e técnicos do Programa de Pós-Graduação em Engenharia Elétrica (PPGEE) que participaram de forma direta e indireta para minha formação, o meu muito obrigado.

"O que você sabe não tem valor;
o valor está no que você faz com o que sabe."
-Bruce Lee-

Resumo

O aprendizado *online* é uma tarefa complexa, especialmente quando o fluxo de dados muda sua distribuição ao longo do tempo. É um desafio monitorar e detectar estas mudanças para preservar o desempenho do algoritmo de classificação. Este trabalho apresenta dois novos métodos de detecção de mudança de conceito, o QT e o QTS, construídos a partir de uma perspectiva diferente de outros detectores já existentes. Os novos métodos de detecção analisam o espaço ocupado pelos dados a partir da premissa de que o espaço ocupado pelos dados de classes diferentes é imutável. Os dados são mapeados em estruturas de memória baseada em quadtree, a qual fornece informações sobre a classe (rótulo) dominante em uma determinada região do espaço de características. A detecção de mudança de conceito no método proposto QT acontece ao atribuir um dado a um espaço previamente ocupado na quadtree por dados de classe oposta. Já o QTS detecta a mudança de conceito quando identifica um aumento significativo na quantidade de dados da quadtree de uma das classes. Os métodos propostos foram avaliados em problemas de classificação binária. Os resultados alcançados pelos métodos de detecção propostos foram competitivos comparados aos métodos existentes na literatura.

Palavras-chave: Fluxo de dados, Mudança de conceito, Quadtree, Detector de mudança de conceito, Classificação.

Abstract

Online learning is a complex task, especially when the data stream changes its distribution over time. It has been a challenge to monitor and detect these changes to preserve the learning algorithm performance. This work presents two novels drift detection methods built from a different perspective of other preexisting detectors from literature.It analyzes the space occupied by the data, assuming that it would be immutable unless changes in this space occur among data of different classes. Data are mapped into a quadtree-based memory structure that provides knowledge about which class (label) is dominant in a given region of the feature space. The proposed method QT detects a drift by checking whether data assigned to a given class occupy spaces considered relevant to the other class. The QTS, on the other hand, detects a concept drift when it identifies a significant increase in the increment of data in one of the classes. The proposed methods were evaluated on binary classification benchmark problems. Results show that our methods were competitive with well-known drift detectors from literature.

Keywords: Data stream, Concept drift, Quadtree, Drift detector, Classification.

Lista de Ilustrações

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

IoT	*Internet of Things* – Internet das Coisas
ML	*Machine Learning* – Aprendizado de Máquina
LSTM	*Long Short Term Memory*
FIFO	*First In First Out* - Primeiro a Entrar Primeiro a Sair
RAM	*Random Access Memory* - Memória de Acesso Randômico
DDM	*Drift Detection Method*
EDDM	*Early Drift Detection Method*
PH	*Page-Hinkley test*
HDDM	*Drift Detection Method based on the Hoeffding's inequality*
ADWIN	*ADaptive WINdowing*
PDF	*Probability Density Function*
KDE	*Kernel density estimation*
RBF	*Radial Basis Function*
LDNF	Limite de Dados no Nó Folha

Sumário

Capítulo 1

Introdução

O aumento da capacidade de computação e o número de dispositivos IoT (*Internet of Things* – Internet das Coisas) conectados têm contribuído significativamente para o acréscimo da quantidade de dados em diferentes contextos. Da mesma forma, o aprendizado de máquina (*Machine Learning* - ML) está cada vez mais presente no dia a dia da indústria, permitindo a automação de tarefas e a otimização de processos. No entanto, estes avanços impuseram desafios extras à comunidade de ML, como a necessidade de processar dados de maneira distribuída, restrições na capacidade de armazenamento e os problemas inerentes ao fluxo de dados contínuos.

Uma forma de lidar com os desafios acima mencionados é permitir que o algoritmo aprenda de modo incremental (ou de modo *online*), onde os dados são apresentados em lotes de tamanho limitado e o algoritmo aprenda continuamente com os dados mais recentes. Ainda assim, a complexidade computacional de retreinar e fazer o ajuste de parâmetros nesses algoritmos, que muitas vezes requer a solução de grandes problemas de otimização, e a presença de mudanças nas distribuições das funções geradoras, fenômeno conhecido como mudança de conceito (*concept drift*), faz com que o uso de técnicas para detectar mudanças nos dados sejam necessárias [Gama et al., 2014]. Junto com os detectores de mudança de conceito, os algoritmos de ML *online* podem se adaptar rapidamente à evolução dos dados, realizando o retreinamento somente quando se fizer necessário.

Na literatura, observam-se duas vertentes de soluções para o problema de mudança de conceito: algoritmos evolutivos (*evolving algorithms*) e algoritmos baseados em gatilho (*trigger-based algorithms*) [Gama et al., 2004, Lu et al., 2018, Zliobaite, 2010, Khamassi et al., 2018]. Os algoritmos evolutivos possuem em seus algoritmos mecanismos implícitos que ajustam automaticamente os parâmetros para lidar com as mudanças de conceito. Para manter o modelo atualizado, os algoritmos evolutivos adotam diferentes estratégias para lidar com a mudança de conceito, como técnicas de aprendizado em comitê de classificadores (*ensemble learning*), redes neurais (*neural network*), agrupamento (*clustering*) e análise estatística (*statistical analysis*) [Cano and Krawczyk, 2020, van Rijn et al., 2018, Zhao

et al., 2020, Liu et al., 2020, Boracchi et al., 2018]. Para uma visão mais aprofundada sobre os algoritmos evolutivos, é recomendada a leitura dos trabalhos de Iwashita and Papa [2018], Leite et al. [2020]. Nos algoritmos baseados em gatilho, um método cujo objetivo é detectar as mudanças de conceito dos dados é integrado ao modelo de aprendizado. Esta integração faz com que o modelo de aprendizado se torne apto a atualizações ou mesmo retreino completo quando uma mudança de conceito for detectada. Os algoritmos baseados em gatilho podem trabalhar monitorando a taxa de erro do modelo, a distribuição de dados brutos (ou processados) ou até mesmo utilizando de múltiplos testes de hipótese a partir de métricas estatísticas extraídas durante o processo de aprendizado do modelo e das amostras do fluxo de dados [Lu et al., 2018, Zliobaite, 2010]. Nos detectores baseados no monitorando da taxa de erro, uma mudança de conceito é detectada quando a taxa de erro do modelo aumenta de forma significativa, o que reflete a sua perda de desempenho. Nesta categoria, os detectores comumente referenciados são o Método de Detecção de Mudança de Conceito (DDM) [Gama et al., 2004], a Máquina de Aprendizado Extremo Dinâmico (DELM) [Xu and Wang, 2017], Teste Estatístico de Detecção de Proporções Iguais (STEPD) [Nishida and Yamauchi, 2007], o Método Preciso de Detecção de Mudança de Conceito (ACDDM) [Yan, 2020], e a Medida de Diversidade como um novo Método de Detecção de Mudança de Conceito (DMDDM) [Mahdi et al., 2020]. Os detectores de mudança de conceito baseados em múltiplos testes de hipótese (ou hierárquicos) combinam monitoramento da taxa de erro com o da distribuição de dados brutos ou processados para detecção dos desvios de conceito. Dentre os métodos mais conhecidos estão a Detecção de Mudança de Conceito Linear de Quatro Taxas (LFR) [Wang and Abraham, 2015], os Testes Hierárquicos de Detecção de Mudanças (HCDTs) [Alippi et al., 2016], e as Quatro Taxas Lineares Hierárquicas (HLFR) [Yu and Abraham, 2017].

Independentemente das estratégias usadas pelos algoritmos baseados em gatilho, a ideia central de muitos dos trabalhos é a de quantificar estatisticamente a dissimilaridade entre duas distribuições de dados em instantes de tempo distintos. Por exemplo, no caso dos detectores baseados no monitoramento da taxa de erro, o detector é acionado apenas quando há uma perda significativa de desempenho do modelo, o que não se aplica a determinadas situações de mudança de conceito. Existem situações, por exemplo, em que ocorre uma mudança na distribuição dos dados e o modelo atual pode classificar os dados de maneira razoável. Tais cenários requerem soluções alternativas capazes de enxergar o problema sob uma perspectiva diferente.

Os detectores de mudança propostos nesta Tese foram construídos a partir de uma visão diferente de outros já existentes. Eles monitoram a ocupação espacial das amostras do fluxo de dados. São utilizadas quadtrees, árvores com estruturas hierárquicas espaciais baseadas no princípio da decomposição recursiva do espaço. Desta forma, os dados são mapeados em uma estrutura de memória baseada em quadtree [Mehta and Sahni, 2004], a qual fornece informações sobre a classe (rótulo) dominante em uma determinada região do

espaço de características. O primeiro método proposto baseado em quadtree é o QT, que monitora a ocupação espacial de dados de classes opostas e detecta a mudança de conceito ao constatar a sobreposição do espaço previamente ocupado por dados de uma classe por dado de classe oposta. O segundo método proposto é o QTS, o qual monitora a ocupação espacial dos dados de classes opostas, e detecta a mudança de conceito ao identificar um aumento significativo na ocupação espacial em uma das classes.

Vale ressaltar que a ideia de mapear e monitorar o espaço ocupado por dados não é nova, pois, foi relatada na literatura no contexto de aprendizagem *online* [Boracchi et al., 2018, Liu et al., 2020]. A abordagem presente neste trabalho, no entanto, é diferente de modo que considera uma estrutura distinta (quadtree) para mapear os dados e uma nova lógica na detecção de mudança de conceito.

1.1 Hipótese

A hipótese deste trabalho é a de que a análise geométrica dos dados no espaço de características por meio da decomposição espacial seja capaz de extrair informações capazes de detectar mudanças de conceito. Eventualmente, ela poderia encontrar situações de mudanças de conceito que outros detectores tradicionais (baseados na perda de desempenho do modelo) possam ter dificuldades em detectar.

1.2 Objetivos

1.2.1 Objetivos Gerais

O objetivo geral deste trabalho é avançar os estudos da detecção sobre mudança de conceito em fluxo de dados, através de uma visão diferente do problema, propondo dois novos métodos que utilizam da decomposição espacial e da geometria dos dados no espaço de características para a detecção de mudanças de conceito.

1.2.2 Objetivos Específicos

- Propor um método de detecção de mudança de conceito que utiliza da análise espacial dos dados, monitorando a sobreposição de dados de classes opostas (QT);
- Propor um segundo método de detecção de mudança de conceito que utiliza da análise espacial dos dados, monitorando o aumento da quantidade de dados em cada classe (QTS);
- Implementar um sistema de mapeamento n-dimensional usando a quadtree;

- Propor uma forma de controle automático da sensibilidade de detecção de mudança do método proposto QT.
- Demonstrar que os métodos propostos são capazes de resultados competitivos, evidenciando seu pontos fortes e fracos, e apontando em qual situação a utilização destes métodos é vantajosa frente a outros detectores.

1.3 Contribuições

Como principais contribuições, foram publicados os seguintes artigos durante o desenvolvimento deste trabalho:

- Á. C. Lemos Neto, R. A. Coelho, and C. L. d. Castro. An incremental learning approach using long short-term memory neural networks. *Journal of Control, Automation and Electrical Systems, pages* 1–9, 2022.
- T. M. L. Menegaldi, R. A. Coelho, and C. L. d. Castro. Aprendizado incremental de redes RBF via agrupamento evolutivo de fluxos de dados. In *Anais do 15 Congresso Brasileiro de Inteligência Computacional, pages* 1–8, Joinville, SC, 2021. SBIC.
- R. A. Coelho and C. L. de Castro. Abordagem espacial via quadtree para detecção de mudança de conceito em fluxos de dados contínuos. In *Congresso Brasileiro de Automática-CBA, pages* 1–6, 2020.

No trabalho Lemos Neto et al. [2022] foi desenvolvido uma variante incremental da rede neural LSTM (*Long Short Term Memory*) capaz de aprender e se adaptar em um contexto de fluxo de dados contínuos, sem a necessidade de um detector de mudança de conceito dedicado. No trabalho Menegaldi et al. [2021] foi apresentada uma maneira de definir a topologia da camada escondida da rede neural de base radial por um algoritmo de clusterização evolutiva MicroTeda [Maia et al., 2020], dando a ela a capacidade de aprender incrementalmente, aprendendo à medida que novas amostras de dados são recebidas.

Em Coelho and de Castro [2020] foram publicados os primeiros resultados do método proposto de detecção de mudança de conceito por meio da quadtree QT. Nesta publicação o método proposto ainda em estágio preliminar já apresentava resultados muito promissores. Foi submetido para a revista *Information Sciences* e se encontra em processo de avaliação o trabalho *Concept Drift Detection with Quadtree-based Spatial Mapping of Streaming Data* que contempla o método proposto acrescido do controle dinâmico do parâmetro altura. Neste trabalho, o método proposto foi avaliado em bases de dados reais e sintéticas, frente a outros detectores bem conhecidos na literatura e se mostrou bastante competitivo, demonstrando real capacidade na detecção de mudança de conceito.

O método proposto nesta Tese, chamado de QT, se apresenta como uma nova solução para o problema de detectar mudanças de conceito. Sob uma perspectiva até então não explorada, utilizando de uma quadtree para cada classe, o QT analisa geometricamente o espaço de características. A detecção de mudança de conceito acontece ao verificar a posição de um dado a um espaço previamente ocupado na quadtree de classe oposta. Este método pode detectar mudanças de conceito em situações em que outros métodos não conseguem, por exemplo, em situações em que ocorrem a mudança de conceito sem uma perda significativa de desempenho do classificador. A Figura 1 apresenta o fluxograma do funcionamento do método proposto QT, em que o x_t é a amostra de dado apresentada no instante de tempo t, o $\hat{y}_t$ é a classe estimada e o y_t e a classe esperada.

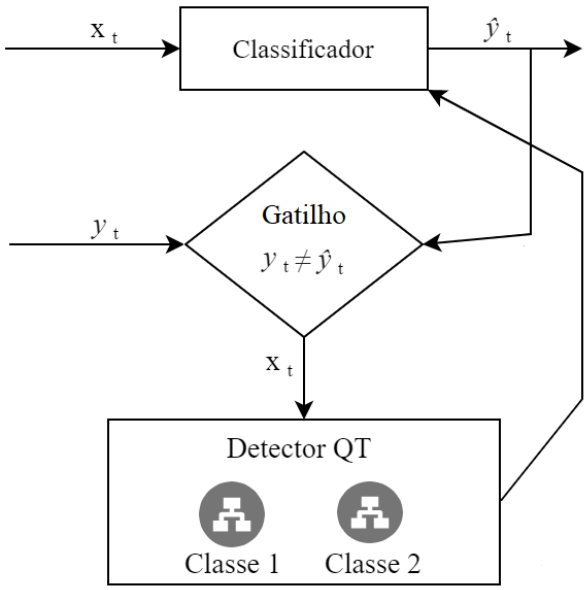


Figura 1 – Fluxograma do funcionamento do método proposto QT.

O QT é um método baseado em erro, que utiliza a amostra que causou o erro de classificação para uma análise espacial, e avalia se há uma mudança de conceito. O método QT possui apenas um parâmetro que controla a sua sensibilidade de detecção. Para estimar o valor deste parâmetro, foram propostas duas estratégias capazes de fornecer este parâmetro de forma dinâmica e automática.

O QTS utiliza de uma quadtree por classe, armazena os dados da qual a classe esperada foi informada em sua respectiva árvore. Monitora o espaço ocupado pelos dados de cada classe que aos poucos começa a saturar, caracterizado pelo menor aumento da quantidade de dados na quadtree da classe. A detecção de mudança acontece quando é constatado um aumento significativo da quantidade de dados na quadtree da classe. É esperado que os métodos propostos possuam a característica em comum de detectar mudanças de conceito em situações em que outros métodos tenham dificuldades ou não consigam.

1.4 Organização da Tese

No Capítulo 2 é apresentado o referencial teórico, as técnicas e métodos utilizados para na criação do método proposto com uma revisão bibliográfica dos trabalhos relacionados. O Capítulo 3 apresenta a metodologia utilizada no desenvolvimento dos métodos propostos e suas especificidades. No Capítulo 4 são apresentados os experimentos que avaliam a capacidade dos métodos propostos frente a outros métodos da literatura. Discussões sobre os resultados alcançados e limitações também estão presentes. Já o Capítulo 5 traz a conclusão do trabalho bem como as sugestões de trabalhos futuros.

Capítulo 2

Fundamentação Teórica e Trabalhos Correlatos

Neste capítulo será apresentada uma definição formal da mudança de conceito em fluxo de dados. Também será apresentado um esquema genérico de um algoritmo de aprendizado para o ambiente de fluxo de dados com mudança de conceito, com suas principais estruturas e a forma de avaliação para estes algoritmos. Será apresentada a estrutura de dados quadtree utilizada para a criação dos métodos propostos, e por fim, uma revisão dos detectores de mudança disponíveis na literatura que estão diretamente relacionados ao trabalho.

2.1 Mudança de Conceito no Fluxo de Dados

Neste ambiente de aprendizado, as amostras são fornecidas na forma de um fluxo contínuo de dados que nem sempre é estacionário. Um fluxo de dados pode ser formalizado como $DS \in \mathbb{R}^d$ uma sequência de dados de d dimensões ordenados no tempo $DS = (x_{i=1}, x_{i=2}, \ldots, x_{i=t}, \ldots)$, em que x_t é o mais novo exemplo de dado a ser apresentado. Cada exemplo x_i está associado a um rótulo $y_i \in Y$ em que $Y = \{0, 1\}$.

A mudança no conceito dos dados ocorre quando a relação entre os dados de entrada e a classe de saída muda ao longo do tempo. Seja DS_t um fluxo de dados com a seguinte função de distribuição de probabilidade conjunta $p_0(x, y)$, em que $\hat{y} = f_t(x)$ é o modelo atual capaz de atribuir corretamente o exemplo de entrada ao rótulo de saída ($\hat{y}_t = y_t$) até o instante t. A mudança de conceito acontece quando a atual função geradora dos dados é substituída por outra $p_1(x, y)$, que ocorre no tempo $t + \tau$ de forma que $DS_{t+\tau}$ representa o fluxo de dados após a mudança de conceito. A consequência desta mudança é que o modelo atual tenha seu desempenho afetado, de forma que não consiga classificar corretamente as amostras após o tempo $t + \tau$. A forma como o modelo classificador se adapta a uma mudança de conceito é um desafio: se ele adaptar rapidamente, informações

relevantes mais antigas podem ser perdidas; Se, por outro lado, sua adaptação for lenta, seu desempenho pode ser altamente afetado. Este *trade-off* é conhecido na literatura como *dilema da estabilidade-plasticidade* [Gama et al., 2004, Khamassi et al., 2018].

Existem dois tipos de mudança de conceito, a mudança virtual e a real. A mudança de conceito virtual não requer mudanças na superfície de decisão $p(y|x)$ (probabilidade a posteriori) para manter a precisão do classificador. Por outro lado, a mudança real de conceito requer atualizações nos limites entre as classes para manter a precisão do classificador (Figura 2). A mudança de conceito real pode se manifestar de diferentes formas: abrupta, incremental, gradual ou recorrente, como ilustra a Figura 3. Os métodos detectores de mudança devem alertar quando ocorre a mudança de conceito real.

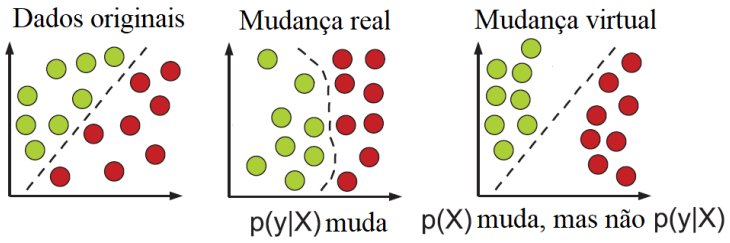


Figura 2 – Mudança de conceito real e virtual.

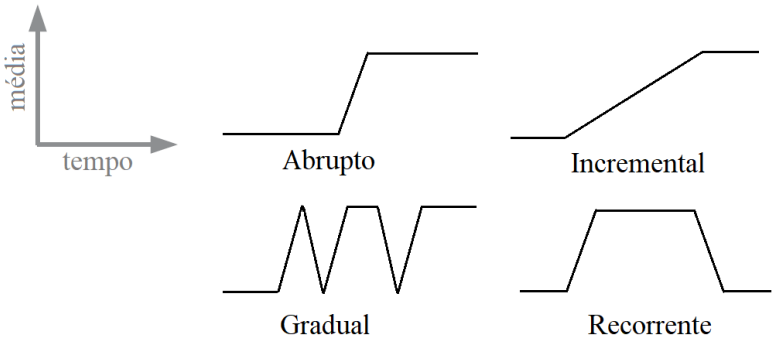


Figura 3 – Formas de mudança de conceito real.

2.2 Esquema Genérico de um Algoritmo de Aprendizado para o Fluxo de Dados

Os algoritmos de aprendizado para fluxo de dados contínuos precisam trabalhar em ambientes dinâmicos, que estão mudando a todo instante. É esperado dos algoritmos a capacidade de aprender de forma incremental com a chegada de novos dados. Se o processo de geração de dados apresentar mudanças em sua função geradora (situação comumente encontrada no mundo real), é desejável que o algoritmo tenha um mecanismo de esquecimento capaz de descartar os dados considerados irrelevantes para que possa se adaptar ao novo conceito.

A Figura 4 apresenta um esquema genérico de um algoritmo de aprendizado para o cenário de fluxo de dados contínuo, adaptado de [Gama et al., 2014]. Este esquema pode ser dividido basicamente em duas partes: A parte responsável pela predição, composta pelos módulos de memória e o módulo de aprendizado, o qual associa a cada exemplo de dado apresentado pelo fluxo uma predição (classificação); A parte responsável pelo treinamento do modelo, composta pelos módulos de estimação de erro e detecção de mudança de conceito, o qual avalia as predições realizadas e procura manter o modelo o mais preciso e atual possível.

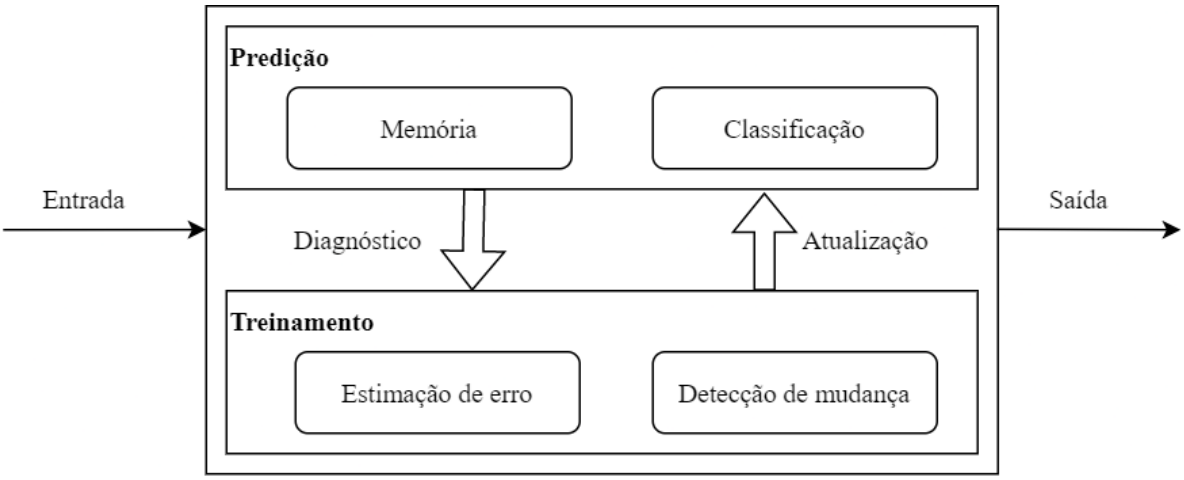


Figura 4 – Esquema genérico de um algoritmo de aprendizado para fluxo de dados, adaptado de Gama et al. [2014]

O módulo de memória é responsável pelo gerenciamento dos dados e pelo sistema de esquecimento. Os dados podem ser presentados ao algoritmo de aprendizado um por fez ou em lotes (*batch*). O gerenciamento dos dados diz respeito à forma como os dados são armazenados após apresentados ao algoritmo de aprendizado. Os dados podem ser armazenados em janelas, que podem ter tamanho fixo ou variável [Gama et al., 2014, Bouchachia, 2011, Bouchachia and Vanaret, 2013].

As janelas são estruturas de dados do tipo fila (*First In First Out* - FIFO) na qual uma certa quantidade de dados podem ser armazenados. A janela pode ser definida como $W[n] = (x_{t-n}, x_{t-(n-1)}, \ldots, x_t)$, em que n é a quantidade de dado que pode ser armazenada na estrutura. Existem diferentes variações de janela como: *landmark window, sliding window, fading window, tilted-time window* entre outras [Nguyen et al., 2015].

A *landmark window* é uma janela que procura conservar em sua estrutura os dados desde o instante de tempo $t = 1$. No entanto, há medida em que o tempo passa, a quantidade de dados e o modelo criado com os dados antigos podem se mostrar inconsistentes frente aos dados mais novos. É o tipo de janela, que ao atingir o seu limite, evolui para uma variável como *sliding window* ou *fading window*. A *sliding window* ou janela deslizante pode manter os n dados mais recentes em sua estrutura, descartando os dados mais antigos.

Na *fading window* são associados pesos w aos dados da janela, em que os dados mais recentes apresentam maior peso, maior relevância, e os dados mais antigos, menor peso. O tamanho da janela está ligado à função do peso, desta forma, o dado sai da janela assim que seu peso se torna zero. A *tilted-time window* é uma variação da *fading window* e janela deslizante. Com um tamanho fixo $W[n]$ são aplicados diferentes níveis de peso w_i para os dados de forma estratificada, ou seja, os dados novos recebem o peso w_1, dados atuais o peso w_2 e dados velhos o peso w_3. A Figura 5 ilustra as variações de janelas apresentadas.

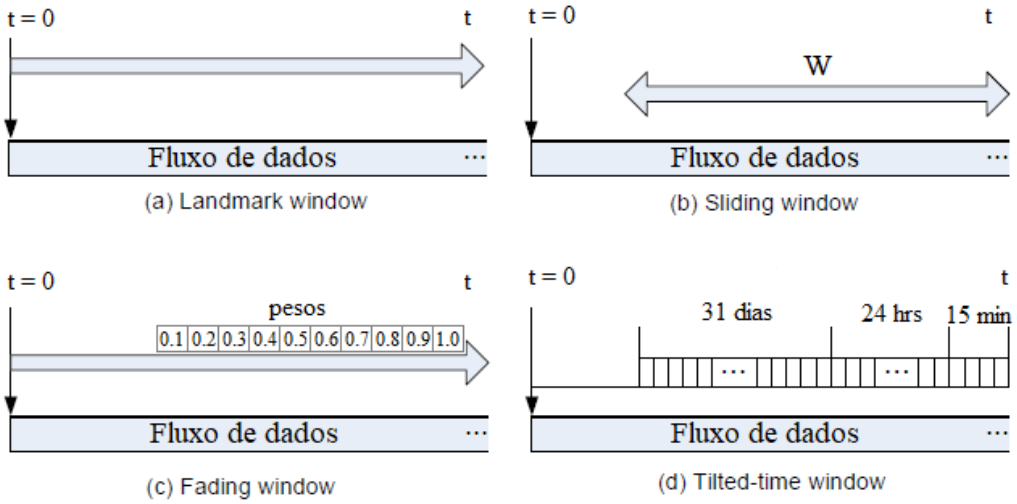


Figura 5 – Tipos de janelas [Nguyen et al., 2015]

O sistema de esquecimento é responsável pela forma como o módulo de aprendizado se adapta a uma mudança de conceito e ao *dilema da estabilidade-plasticidade*. Pertencente ao módulo de memória, o sistema de esquecimento conta basicamente com duas estratégias sobre a estrutura de armazenamento de dados que são: esquecimento gradual e esquecimento abrupto. O esquecimento gradual associa pesos aos exemplos presentes na memória. Isso acarreta em menor influência dos dados mais antigos. O esquecimento abrupto descarta os dados assim que estes são irrelevantes [Gama et al., 2014, Zhao et al., 2011, Klinkenberg, 2004].

No módulo de classificação estão presentes as técnicas e mecanismos de aprendizado para o fluxo. Este módulo é a sede do algoritmo de aprendizado, responsável por atribuir uma classe a cada exemplo de dado apresentado, e de estratégias de adaptação incremental com e sem a detecção de mudanças. O módulo de classificação também é responsável por técnicas e manutenção de modelos de classificadores em comitê [Gama et al., 2014, Ikonomovska et al., 2011, Gama et al., 2006, Carmona-Cejudo et al., 2011, Elwell and Polikar, 2011].

O módulo de estimação de erro é responsável pelo sistema de retorno entre a parte de predição e a parte de treinamento. Os algoritmos baseados em gatilho necessitam de algum retorno para acionar estratégias de adaptação necessárias para manter o módulo

de classificação atualizado, este retorno pode ser oferecido pelo módulo de estimação de erro (métodos baseados em erro). Neste contexto, somente após um erro, o módulo de classificação sofre atualização [Gama et al., 2014].

O módulo de detecção de mudança (*drift detection*) refere-se às técnicas e mecanismos para detecção de mudanças na função geradora dos dados. Este módulo é responsável por identificar as mudanças reais no conceito dos dados. A mudança de conceito real acontece quando a atual função geradora dos dados $p_0(x, y)$ é substituída por outra $p_1(x, y)$, que ocorre no tempo $t + \tau$ de forma que $(y_t \neq y_{t+\tau})$.O módulo de detecção de mudança deve ser capaz de alertar o módulo de classificação, já que não é mais possível classificar com a mesma acurácia as amostras após o tempo $t + \tau$.

2.2.1 Forma de Avaliação

A comparação entre algoritmos de aprendizado para fluxo de dados não é fácil, pois, os autores nem sempre usam os mesmos métodos de avaliação e os mesmos conjuntos de dados. Cada um dos módulos, ou conjunto deles que compõem o algoritmo de aprendizado para fluxo de dados, possuem critérios para comparação, como o custo computacional e a quantidade de memória RAM (*Random Access Memory* - Memória de Acesso Randômico) gasta por hora no processo, a probabilidade de falsa detecção, a verdadeira detecção e o atraso na detecção de mudança de conceito na distribuição dos dados [Gama et al., 2014, Bifet et al., 2010]. Já os critérios de avaliação quanto ao resultado do processo de aprendizado no fluxo de dados, são os mesmos utilizados no contexto da aprendizagem *offline*, apenas adaptados, os quais temos a acurácia, matriz de confusão, revocação, precisão e outros.

No cenário de aprendizado em fluxo de dados contínuos existem duas técnicas principais para avaliação: O *Holdout Evaluation* e o *Prequential Evaluation* [Gama et al., 2014, Lemaire et al., 2014]. O *Holdout Evaluation* requer o uso de dois conjuntos de dados, o conjunto de dados de treino e o conjunto de dados de teste (validação). O conjunto de dados de treino é usado para treinar o algoritmo de aprendizado, e o *"holdout"* de teste é então usado para avaliar o algoritmo de aprendizado atual em intervalos de tempo regulares.

O *Prequential Evaluation* também conhecido como *Test-Then-Train* ou Teste-então-Treine tem a avaliação do algoritmo de aprendizado feita a partir das próprias amostras do fluxo de dados. A mesma mostra usada para o teste do algoritmo de aprendizado também pode ser utilizada para treinar o algoritmo de aprendizado se necessário. A vantagem é que o algoritmo de aprendizado está constantemente sendo testado, ou seja, não é necessário construir os conjuntos de treino e teste, com a vantagem de que todos os dados são utilizados na avaliação do modelo.

Lemaire et al. [2014] alerta quanto a forma de avaliação e a presença ou não da mudança de conceito no fluxo de dados. Quando o fluxo de dados é estacionário, ou seja, não apresenta mudança de conceito, tanto a *Holdout Evaluation* quanto a *Prequential Evaluation* podem ser usadas. Na presença de mudança de conceito no fluxo de dados a *Prequential Evaluation* é a indicada. Caso utilize o método *Holdout Evaluation* existe a possibilidade do conjunto de dados usado para avaliação possuir dados de conceitos diferentes, ou seja, uma mesma janela de avaliação com dois conceitos diferentes.

2.3 Quadtree

A quadtree é uma estrutura de dados do tipo árvore, que possui arranjos hierárquicos espaciais que se baseiam no princípio da decomposição recursiva do espaço. Por meio da decomposição recursiva do espaço $\mathbb{R}^2$, usando de separadores paralelos aos eixos das coordenadas e abscissas, tem-se como resultado quatro regiões de mesmo tamanho [Mehta and Sahni, 2004]. É importante estabelecer uma relação entre os arranjos hierárquicos espaciais e as regiões que correspondem aos nós da quadtree.

O nó raiz da quadtree corresponde ao espaço de características que contém todos os pontos de interesse, ou seja, a base de toda a hierarquia espacial. O nó raiz da quadtree é denominado de célula raiz. Cada região de mesmo tamanho resultante da divisão da célula raiz é chamado de célula. Os termos sub-célula e super-célula são usados para definir a relação hierárquica de uma célula. Desta forma, o termo sub-célula é usado para definir uma célula que está contida em outra, já a célula que contém a sub-célula é definida como super-célula.

A Figura 6 ilustra a estrutura hierárquica de uma quadtree. Em (1) é apresentada a célula raiz. Em (2) está representada a primeira divisão recursiva do espaço na qual estão definidas as células A, B, C e D. Em (3) temos a segunda divisão recursiva na qual estão definidas as células E, F, G e H na quais são sub-células da super-célula C. Em (4) a terceira divisão recursiva na qual está definida a célula I que é uma sub-célula da super-célula E, desta forma delimitando a seguinte relação hierárquica $I \subset E \subset C$.

Do nome quadtree dado à estrutura, o termo "*quad*" se refere a divisão resultante do espaço do nó raiz, que resulta em quatro nós folha com regiões congruentes. A quadtree admite apenas 1 dado por região, ou seja, somente os nós folha possuem dados. Ao inserir um segundo dado, o nó é dividido recursivamente em 4 nós filhos, e cada dado deve ser mapeado para o correto nó folha [Mehta and Sahni, 2004]. O processo passo a passo de construção de uma quadtree, com a divisão espacial e sua representação na estrutura de árvore, está ilustrado na Figura 7.

Para uma estrutura de dados com mesmo funcionamento em $\mathbb{R}^3$, dá-se o nome de *octree*. Nesta árvore, a decomposição recursiva do espaço do nó raiz resulta em oito

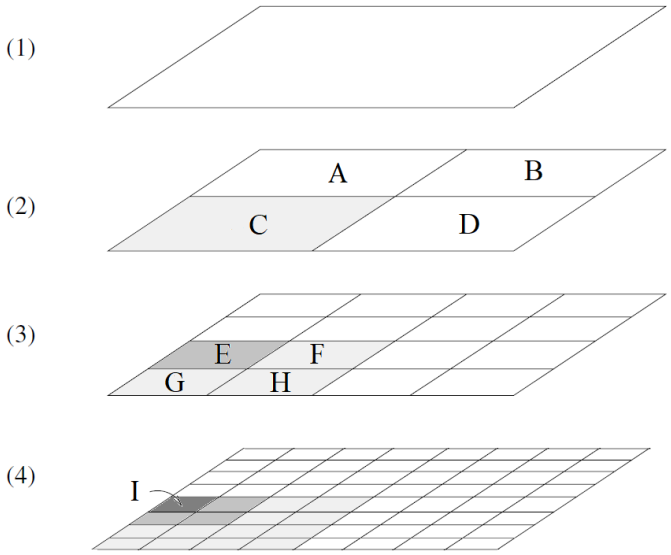


Figura 6 – Estrutura espacial hierárquica, [Mehta and Sahni, 2004]

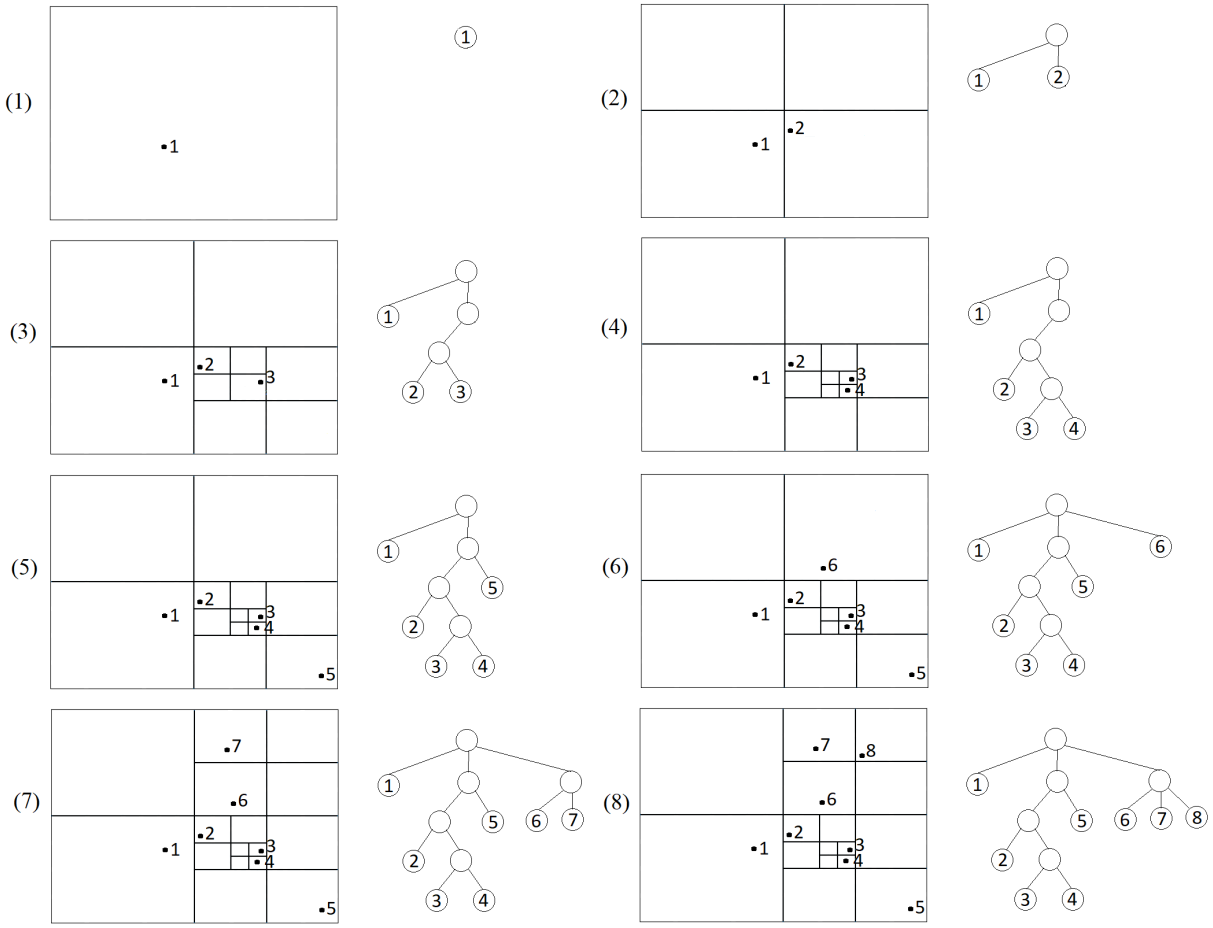


Figura 7 – Construção da estrutura de dados de uma quadtree

hipercubos congruentes. À medida que a dimensionalidade do espaço $\mathbb{R}^d$ aumenta, para um $d > 3$, cada nova recursão da estrutura de dados resulta em um aumento de 2^d

espaços congruentes. É prática usual a aplicação do nome quadtree independentemente da dimensionalidade da estrutura, com complexidade de tempo para construção da árvore de $O(dn \log n)$.

2.4 Estado da Arte

Esta seção apresenta uma breve revisão dos métodos de detecção de mudança de conceito. Conforme mencionado no Capítulo 1, o método proposto nesta Tese monitora o espaço de características e usa do particionamento espacial para detectar mudanças de conceito, mas outros métodos usam de abordagens semelhantes. No trabalho de Liu et al. [2020] é apresentado uma modificação do algoritmo de clusterização k-means, chamado EI-kmeans (*Equal Intensity k-means*). Neste trabalho o algoritmo EI-kmeans cria *clusters*, segmentando o espaço em *bins* de tamanhos diferentes, com mesma quantidade de dados. A detecção de mudança de conceito acontece uma vez que esteja finalizada a divisão do espaço via EI-kmeans das amostras de treino, teste e a tabela de contingência montada. É aplicado o teste chi-quadrado de Pearson para avaliar a discrepância entre as amostras por meio dos valores presentes na tabela de contingência.

No trabalho de [Boracchi et al., 2018] é apresentado um algoritmo de divisão do espaço chamado QuantTree (estratégia semelhante ao kd-tree). A divisão realizada pela QuantTree resulta em espaços com densidades uniformes chamados de *bins*, onde que cada *bin* apresenta uma estimativa de probabilidade de uma amostra do fluxo de dados cair naquele espaço. A detecção de mudança é feita por um teste de hipótese que avalia se o *batch* de amostras apresentado pelo fluxo de dados são consistentes com o histograma de referência construído a partir das amostras de treino. Estes dois trabalhos são exemplos que compartilham da ideia de particionamento do espaço para a criação de algoritmos para detecção de mudança de conceito, mas estes algoritmos independem de uma configuração de aprendizado ativo para esta tarefa. O detector de mudança proposto neste trabalho é um método baseado em erro, ou seja, depende de um retorno do classificador para agir.

Os métodos de detecção de mudança baseados em erro são acoplados a um classificador na estrutura de um algoritmo de aprendizado para o fluxo de dados. Ele analisa continuamente o desempenho do classificador e, a diminuição do desempenho é indício de um possível desvio. É aprensado a seguir uma revisão de detectores de mudança representativos, que podem ser classificados em três grupos gerais: métodos baseados em controle estatístico de processos, métodos baseados em análise sequencial e métodos baseados em janelas [Gama et al., 2014]. Para uma revisão mais extensa sobre algoritmos de aprendizado em fluxo de dados, são recomendados os trabalhos de [Lu et al., 2018, de Barros and de Carvalho Santos, 2019, Barros and Santos, 2018, Khamassi et al., 2018, Zliobaite, 2010].

O Método de Detecção de Mudança de Conceito (*Drift Detection Method*) (DDM) [Gama et al., 2004] monitora a taxa de erro do classificador e define zonas de tolerância. O DDM monitora a soma do erro de classificação geral e seu desvio padrão empírico dado por $s_i = \sqrt{e_i\,(1-e_i)\,/i}$, onde e_i é a taxa de erro em cada instante de tempo i e s_i é o desvio padrão. São definidos dois valores estatísticos para o limite de detecção, e_{min} e s_{min}, onde e_{min} é a taxa de erro mínima registrada e s_{min} é o desvio padrão mínimo registrado. Ambos os valores são alcançados quando $e_i + s_i$ é mínimo. As zonas de tolerância são definidas da seguinte forma: se $e_i + s_i \geq e_{min} + 2 * s_{min}$ então a zona de alerta é acionada; caso contrário, se $e_i + s_i \geq e_{min} + 3 * s_{min}$ então uma mudança de conceito é detectada. Uma ideia semelhante foi adotada e aplicada no Método de Detecção de Mudança de Conceito Antecipado (EDDM) [Baena-Garcıa et al., 2006], monitorando a distância entre dois erros de classificação consecutivos além da taxa de erro. O mesmo princípio também foi aplicado em Máquinas de Aprendizado Extremas Dinâmicas (DELM) [Xu and Wang, 2017], no Método de Detecção de Mudança de Conceito Baseado em Desigualdade de Hoeffding (HDDM) [Frias-Blanco et al., 2014], no Aprendendo com a Detecção de Mudança de Conceito Local (LLDD) [Gama and Castillo, 2006], e no Método de Detecção de Mudança de Conceito Reativo (RDDM) [Barros et al., 2017].

O Método de Detecção de Mudança de Conceito baseado na desigualdade de Hoeffding (*Drift Detection Method based on the Hoeffding's inequality* (HDDM)) [Frias-Blanco et al., 2014] é um método baseado no controle estatístico de processo que utiliza de desigualdade de probabilidades através de testes estatísticos baseados na inequação de Hoeffding [Hoeffding, 1994]. O HDDM realiza dois tipos de teste, o A-Test e o W-Test para os dois métodos propostos, HDDM$_a$ e HDDM$_w$ respectivamente. O HDDM$_a$ compara as médias móveis para a detectar a mudança de conceito, já o HDDM$_w$ utiliza de médias móveis ponderadas exponencialmente (*Exponential Weighted Moving Average* (EWMA)) para detectar a mudança de conceito. Semelhante ao DDM, utiliza de dois níveis de confiança, alerta, um novo classificador começa a ser treinado, e alarme, quando a mudança de conceito é detectada e o classificador atual é substituído pelo novo.

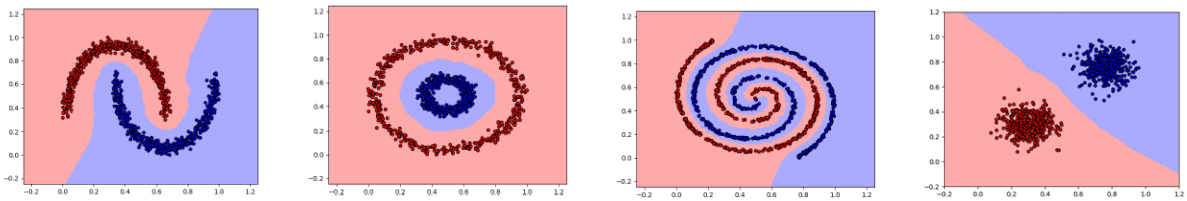
Técnicas de análise sequencial foram adaptadas para detecção de mudança de conceito. O *Page-Hinkley test* (PH) [Page, 1954], dado por $PH_t = U_t - m_t$, calcula as diferenças entre a variável cumulativa (U_t) e o valor mínimo observado (m_t). A variável cumulativa U_t é definida como a diferença cumulativa entre os valores observados e sua média até o instante de tempo atual, ou seja, $U_t = \sum_{i=1}^{t}(x_i - \bar{x} - \delta)$ onde $\bar{x}$ é a média dos dados até o momento atual e δ é a mudança de magnitude permitida. O valor mínimo m_t é definido como o valor mínimo observado de U_t, $m_t = min(U_t)$. Se essa diferença for maior que o valor limite λ, uma mudança de conceito será detectada. Como exemplo de método de detecção de mudança de conceito tem o algoritmo *CUmulative SUM* (CUSUM) [Basseville et al., 1993]. As abordagens sequenciais têm como característica não necessitar de memória de armazenamento, pois, apenas as medidas históricas são armazenadas e

comparadas com a medida atual.

O método do Janelamento Adaptativo *ADaptive WINdowing* (ADWIN) [Bifet and Gavalda, 2007] é um algoritmo baseado em janela para detecção de mudança de conceito. Os métodos baseados em janela geralmente monitoram as distribuições entre duas janelas de tempos diferentes. Uma delas resume informações sobre dados passados enquanto a outra resume informações dos dados mais recentes. O ADWIN utiliza de duas janelas ajustadas dinamicamente, representando dados mais antigos (W_{hist}) e recentes (W_{new}). As mudanças de conceito são detectadas quando a diferença média das amostras destas janelas $|\hat{\mu}_{hist} - \hat{\mu}_{new}|$ é maior que um determinado limiar δ. Implementações semelhantes foram aplicadas ao Detector de Mudança de Conceito pelo Ranqueamento do Teste de Wilcoxon (WSTD) [de Barros et al., 2018], e Janelas Independentes Adaptativas Dinâmicas Detectoras de Mudança de Conceito (DAWIDD) [Hinder et al., 2020].

2.5 Considerações Finais

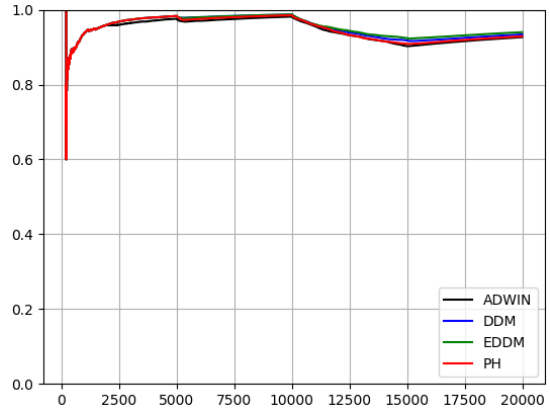
A maioria dos detectores de mudança descritos na Seção 2.4 são métodos baseados em erro associados a um teste estatístico. O detector é acionado apenas quando ocorre uma perda significativa de desempenho do classificador, o que pode não ser o caso. Existem situações em que a mudança de conceito ocorre, e o modelo classificador atual pode classificar os dados de forma razoável. Esta situação foi recriada no Capítulo 4 (Seção 4) usando um experimento controlado, por meio das bases de dados Toy e Checkerboard. A base de dados Toy apresenta três mudanças de conceito que ocorrem a cada 5000 amostras, causadas pelas transições entre quatro diferentes conceitos formados por funções bem conhecidas, como ilustra a Figura 8.



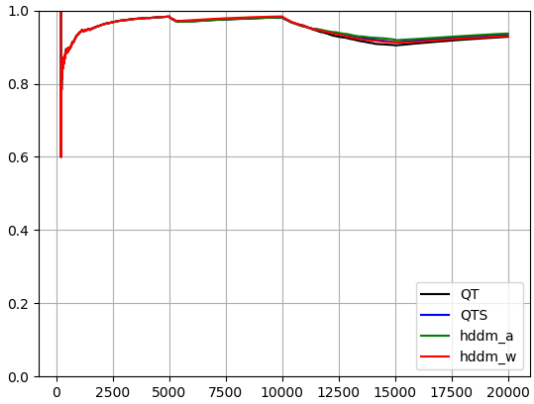
(a) Duas Meia Luas (b) Círculos (c) Espirais (d) Gaussianas

Figura 8 – Base de dados Toy

Ao observar a curva da acurácia geral (Figura 9), a terceira mudança de conceito não é percebida pelos detectores (baseado em erro) como o DDM, pois, o desempenho do atual do classificador não diminui. Isso pode ser controverso, no entanto, efetivamente aconteceu a mudança de conceito. Toda mudança de conceito real deve ser detectada, pois, é a forma de manter atualizado o modelo classificador. A Figura 9 apresenta a acurácia geral do classificador durante o fluxo de dados da base Toy, já a Figura 10 apresenta o histograma com a posição e frequência onde as detecções aconteceram.



(a) Acurácia geral dos detectores ADWIN, DDM, EDDM e PH.



(b) Acurácia geral dos detectores QT, QTS, HDDMa e HDDMw.

Figura 9 – Acurácia geral para a base Toy.

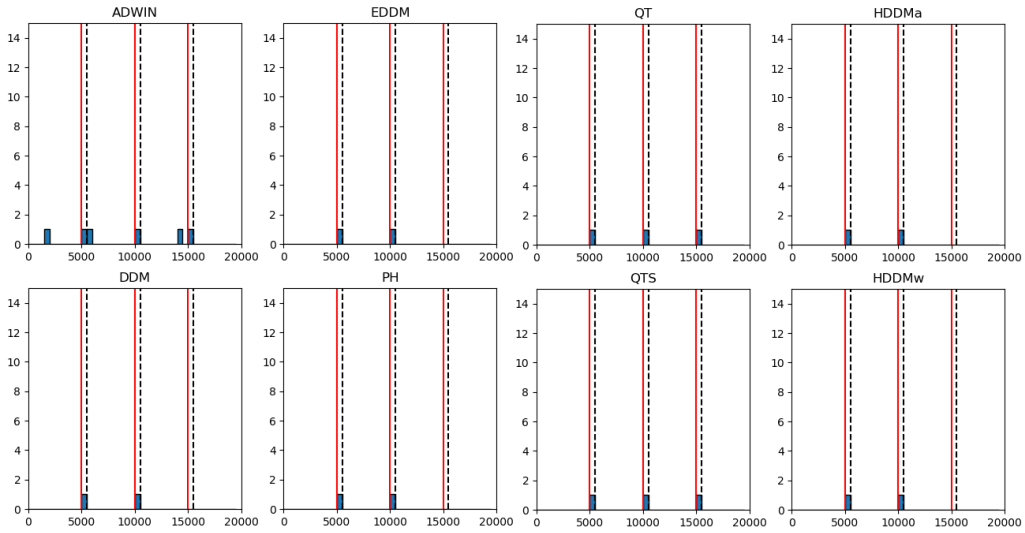


Figura 10 – Histograma da base Toy.

Os métodos propostos nesta Tese para a detecção de mudança de conceito utilizam do gatilho e dependem do erro do classificador, mas são construídos a partir de uma perspectiva diferente dos detectores baseados em erro. Eles analisam o espaço ocupado pelos dados, assumindo que este é imutável, a menos que ocorram mudanças neste espaço; o que implica uma mudança em $p(y|x)$ e não apenas em $p(x)$ ou $p(y)$. Os métodos propostos trazem propriedades interessantes, que são apresentadas no Capítulo a seguir.

Capítulo 3

Detectores de Mudança Propostos

O desenvolvimento de um novo método de detecção de mudança se justifica ao atender a algum propósito. Mesmo que seja o de resolver um problema específico que necessite de características únicas, ampliar o leque de problemas que o método solucione, sanar alguma deficiência ou ponto fraco que um método possua, ou até mesmo a necessidade de melhoria de alguma métrica, como a acurácia, a precisão ou a redução do atraso nas detecções.

A principal motivação para o desenvolvimento dos métodos desta Tese, está no fato de que grande parte dos detectores de mudança vigentes na literatura explora apenas a distribuição dos erros do classificador para alarmar uma mudança, não utilizando informação sobre os dados que são apresentados na entrada do modelo. Uma vez que estes dados de entrada e seus rótulos (classes) ficam disponíveis para o problema em questão, nosso argumento é de que outras informações poderiam ser extraídas desses dados e usadas pelo detector, para complementar a sua decisão acerca de um alarme.

Uma segunda motivação está na crença de que a análise da ocupação espacial dos dados ao longo do fluxo possa detectar mudanças que seriam imperceptíveis pelos detectores convencionais, ou até mesmo acelerar algumas detecções. A ideia não é desprezar a informação do desempenho do classificador, que se apoia nos erros e acertos, mas usá-la de maneira adicional à ocupação espacial dos dados de entrada.

A estrutura de dados quadtree permite monitorar o espaço de características. Uma característica importante da quadtree é o fato desta possibilitar a implementação de índices espaciais para otimizar consultas feitas em bancos de dados que armazenam dados referentes à localização de objetos no espaço, tais como os bancos de dados geográficos [Mehta and Sahni, 2004]. Tomando como base este conceito de índices espaciais, foi implementado nesta Tese um sistema de mapeamento binário que possibilita armazenar dados de alta dimensão na quadtree, superando uma limitação que é inerente a esta estrutura, e que usualmente é discutida na literatura de Geometria Computacional. Ademais, a quadtree possui um algoritmo eficiente (tempo polinominal) para construção e atualização de dados

(vide Seção 2.3.), o que a torna interessante para o cenário de fluxos de dados contínuos.

Outra propriedade interessante da quadtree, que é explorada nos métodos da Tese, é sua capacidade de operar sobre dados que são mapeados para o mesmo índice. Ao se impor um limite no parâmetro altura da quadtree, dados pertencentes a um mesmo nó folha (mesmo índice) são sumarizados. A sumarização permite um armazenamento mais inteligente do fluxo de dados na quadtree, economizando memória, sem a perda significativa de informação.

Como mencionado anteriormente, os dois métodos de detecção de mudança propostos nesta Tese monitoram a ocupação dos dados no espaço de entrada, com o uso de quadtrees. No caso do primeiro método, denominado QT (detecção baseada em QuadTree), uma mudança é apontada quando os dados de uma determinada classe passam a ocupar um espaço previamente ocupado por dados de outra classe. O segundo método proposto, denominado QTS (detecção baseada na saturação da QuadTree), alarma uma mudança de conceito quando ocorre um aumento significativo na quantidade de dados armazenados em uma das quadtrees.

Ambos os métodos QT e QTS são descritos em detalhes neste capítulo. No entanto, antes de descrevê-los, a Seção 3.1, a seguir, traz as ideias por trás do sistema de mapeamento binário que foi proposto para contornar o problema de se armazenar dados de dimensão maior que 3 em uma quadtree.

3.1 Mapeamento n-Dimensional para Quadtree

Como os dados a serem armazenados na quadtree nem sempre estão em $\mathbb{R}^2$, foi desenvolvido um sistema de mapeamento binário que possibilita o armazenamento de dados em $\mathbb{R}^d$, em que d é o número de dimensões. O sistema de mapeamento binário se aplica ao princípio básico da quadtree, que consiste em dividir o espaço em 2^d espaços congruentes. Tomando como base um espaço $\mathbb{R}^2$ em que uma divisão recursiva resulta em 4 espaços congruentes, têm-se os eixos divididos em seus respectivos ponto-médios por retas perpendiculares a cada um desses eixos (ou dimensões). Desta forma, foi adotado o mapeamento binário que atribui o valor "0" para o espaço com valores menores que o valor do ponto de intersecção da reta com o eixo, e atribui o valor "1" para o espaço com valores maiores, como observado na Figura 11.

Desta forma, para uma dada quadtree em $\mathbb{R}^2$, cada uma das duas dimensões será mapeada de forma binária, assumindo os valores "0" ou "1". Isso torna possível sua representação por um par de valores binários, vide Figura 12.

O nó raiz da quadtree corresponde ao espaço de características d dimensional delimitado por um hipercubo que contém o fluxo de dados. Uma função *hash* foi implemen-

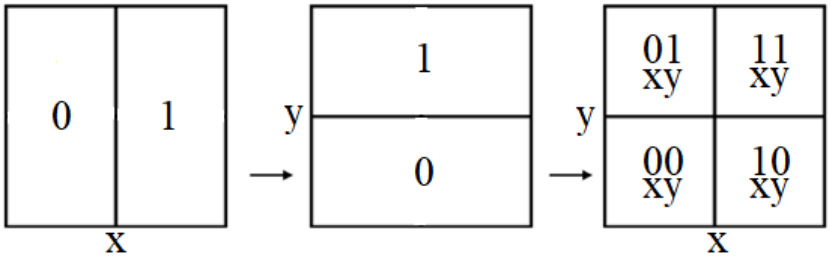


Figura 11 – Mapeamento binário por dimensão

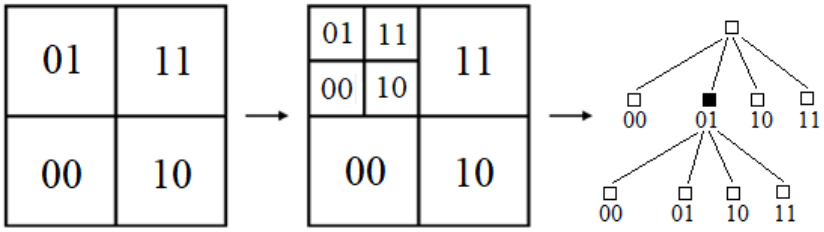


Figura 12 – Modelo de mapeamento da quadtree

tada para mapear os dados para seus espaços correspondentes. É usado o *ponto médio* do hipercubo para atribuir um valor binário "0" ou "1" para compor a chave de endereçamento para o hipercubo filho. O *ponto médio* é um vetor de características formado pelo encontro de todos os hiperplanos bissetores no processo de divisão do hipercubo atual. As posições dos dados presentes no hipercubo a ser dividido são comparadas com a respectiva posição do *ponto médio* para gerar a chave de endereçamento para o correto hipercubo filho, de maneira similar ao exemplo anterior em $\mathbb{R}^2$, ilustrado na Figura 12.

Para cada dimensão do espaço de características, o valor binário "0" é incorporado à chave do hipercubo se o valor da posição do dado for menor que o valor da posição do *ponto médio*; caso contrário, o valor binário "1" é incorporado. Esse processo produz uma chave binária (*string*) cujo tamanho é igual à dimensão dos dados. Por exemplo, para um espaço $d = 4$ a sua divisão recursiva resulta em $2^4 = 16$ novos espaços. Cada um desses espaços é endereçado de forma binária pela composição de um número de d *bits*, ou seja, composto de uma *string* de 4 *bits* capaz de endereçar de "0000" a "1111". Para um *ponto médio* do hipercubo igual a $[0.5, 0.5, 0.5, 0.5]$ e um dado $[0.3, 0.6, 0.8, 0.1]$ temos como chave a *string* "0110". O processo de mapeamento está descrito no pseudocódigo do Algoritmo 1.

3.1.1 Sumarização de Dados na Quadtree

A quadtree pode armazenar uma certa quantidade de dados dependendo de sua altura h, e dimensão d, funcionando como uma estrutura de memória. No entanto, o número de divisões recursivas e a sua altura podem ser aumentadas drasticamente dependendo dos quão próximos estão os dados. Nessa situação, as recursões tendem a continuar até que o tamanho dos hipercubos sejam muito pequenos para acomodar um único dado.

Algoritmo 1: Algoritmo da construção das chaves que mapeiam os dados em uma quadtree de d dimensões.

Entrada: ponto médio do hipercubo, amostra de dado
Saída: chave de endereço
função CHAVE HASH($pontomedio[\,], dado[\,]$)
  $chave \leftarrow$ " "
  para i **em** $length(pontomedio)$ **faça**
    se $dado[i] < pontomedio[i]$ **então**
      $chave$ + "0"
    senão
      $chave$ + "1"
 retorne $chave$

Assumindo que s é a menor distância entre dois dados e D é o comprimento do lado do hipercubo raiz (nó raiz da árvore), o comprimento do lado do menor hipercubo filho (de dimensão d) que pode conter os dois dados com distância s é dado por $s/\sqrt{d}$. Assim, o valor da altura h mínima necessária para acomodar esses dois dados mais próximos em diferentes hipercubos filhos pode ser calculada pela seguinte equação [Mehta and Sahni, 2004]

$$\frac{D}{2^h} < \frac{s}{\sqrt{d}} \quad \equiv \quad h = \lceil \log \frac{\sqrt{d}D}{s} \rceil \tag{3.1}$$

Como pode ser visto na Equação 3.1, a complexidade espacial da estrutura de memória baseada em quadtree aumenta com o número de dimensões d e também com a proximidade s dos dados. Para limitar a quantidade de memória usada, pode-se adotar um valor máximo para o parâmetro da altura h da quadtree. Tal limitação para h permite a presença de mais de um dado dentro de um hipercubo filho que não pode ser dividido recursivamente, quebrando a regra da quadtree de ter apenas um único dado por nó folha. Na presença de mais de um dado em um hipercubo de altura máxima (limitada), tomamos o vetor de características médio correspondente aos dados que se encontram no mesmo hipercubo filho, ou seja, $\bar{x} = \frac{1}{j} * \sum_{i=1}^{j} x_i$, onde j é a quantidade de dados no mesmo nó folha. Essa estratégia de limitação da altura permite gerar uma representação sumarizada dos dados no espaço de características.

Para ilustrar o processo de sumarização de dados em quadtrees, a Fig. 13 mostra o conjunto de dados Duas meia luas contendo um total de 5000 amostras. A Figura 14 mostra a quadtree de altura $h = 5$ formada a partir da base de dados, e a Figura 15 mostra a quadtree com os dados sumarizados. Os pontos dentro dos *quads* representam a posição resultante dos vetores de características médios correspondentes, resultando em um total de 360 pontos na quadtree.

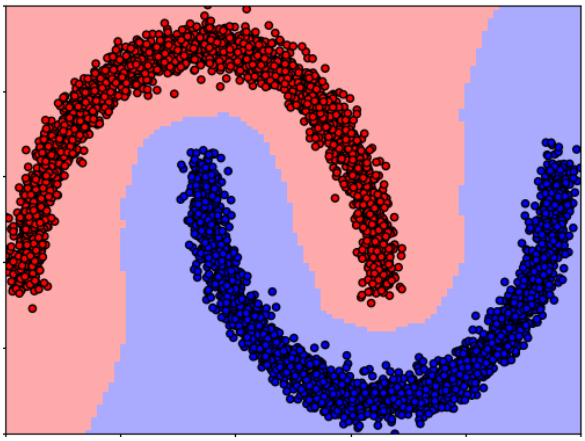


Figura 13 – Base de dados Duas meia luas.

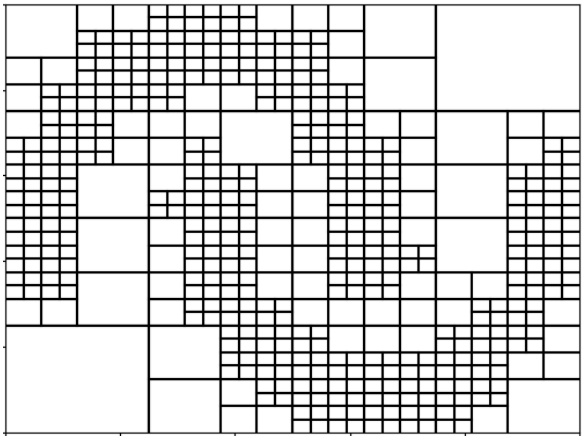


Figura 14 – Quadtree de $h = 5$ formada a partir da base de dados Duas meia luas.

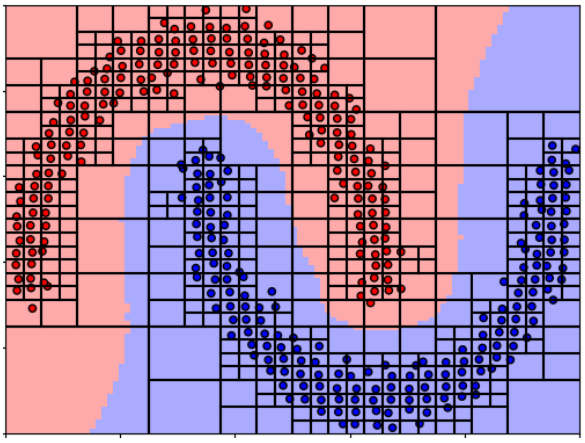


Figura 15 – Quadtree com os dados sumarizados a partir da base de dados Duas meia luas.

3.2 Método de Detecção de Mudança de Conceito Baseado em Quadtree (QT)

O detector de mudança de conceito baseado em quadtree (QT) assume que a atual geometria do fluxo de dados no espaço de características tende a ser imutável, a menos que ocorra uma mudança de conceito real. Se essa mudança na geometria afeta a capacidade da atual regra de classificação ($\hat{f}(x) \approx p(y|x)$), então a mudança de conceito deve ser detectada. Observando a estrutura formada pela quadtree, mudanças significativas em sua geometria ocorre quando novos dados são inseridos em hipercubos folha já ocupados por dados de classe oposta. Adicionalmente, pequenas mudanças nesta geometria devem ser toleradas. O método deve ser capaz de acomodar ruídos e absorver mudanças graduais na distribuição de dados; como todo o espaço é mapeado por quadtrees, inserções de dados de classes opostas em espaços vazios ou menos densos não devem acionar o detector de mudança.

O mapeamento dos dados pelas quadtrees ocorre individualmente por classe; para um problema de classificação contendo duas classes, duas quadtrees são geradas a partir dos dados da fase de pré-treino do classificador. As Figuras 17 e 18 apresentam quadtrees formadas individualmente por classe a partir dos 200 primeiros dados da base Gaussianas (Figura 16). Depois disso, as quadtrees são atualizadas de forma incremental com a chegada de novos dados. Nem todas as amostras apresentadas pelo fluxo de dados serão inseridas nas quadtrees, de modo que essa decisão depende da saída do classificador. Apenas os dados que causaram erros de classificação são inseridos na quadtree.

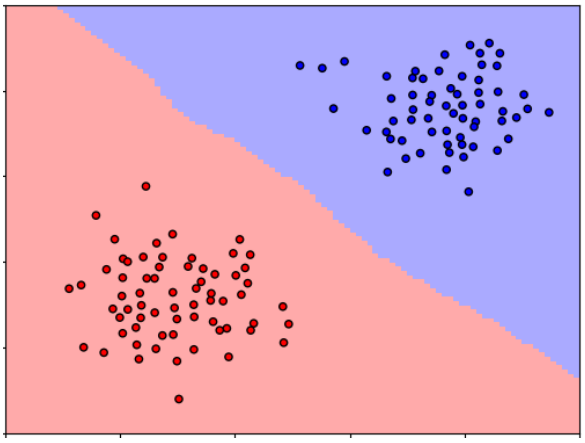


Figura 16 – Base de dados Gaussianas com 200 dados.

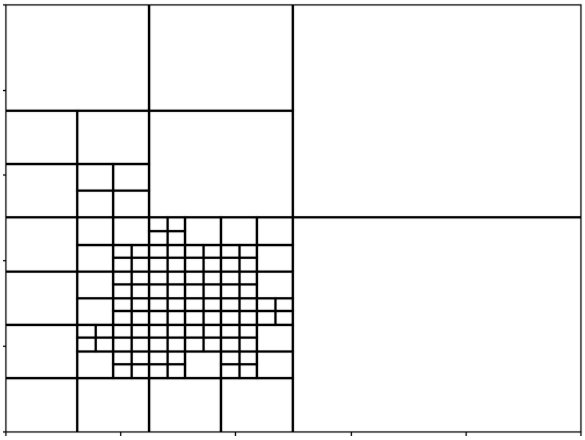


Figura 17 – Quadtree para os dados da classe "vermelha".

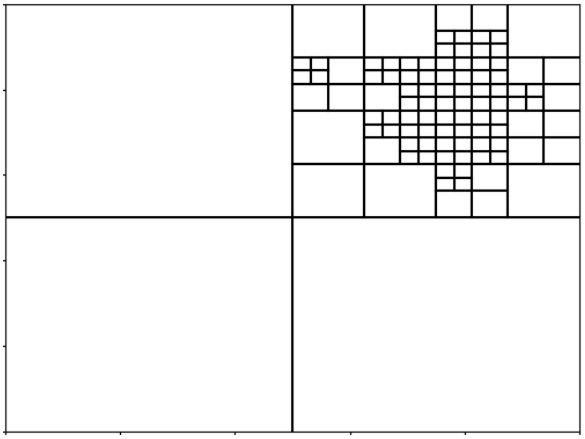


Figura 18 – Quadtree para os dados da classe "azul".

Seja DS_t o fluxo de dados no instante de tempo t, em que a função de classificação $\hat{y}_t = f_t(x_t)$ foi capaz de classificar corretamente os dados até o instante t. No instante de tempo $t + \tau$, o classificador apresentou um erro $\hat{y}_{t+\tau} \neq f_t(x_{t+\tau})$, acionando assim a verificação da posição da amostra $x_{t+\tau}$ na quadtree de classe oposta. Desta forma, dependendo do local verificado uma das duas seguintes ações será tomada:

- Ação de *atenção*; uma possível mudança gradual na distribuição dos dados (ou dado de ruído). Neste caso, a amostra $x_{t+\tau}$ teve seu mapeamento de verificação para um

hipercubo filho vazio na quadtree da classe oposta; a amostra $x_{t+\tau}$ é inserida na quadtree de sua própria classe e o QT comunica ao classificador para que use a amostra em seu treinamento incremental.

- Ação de *alarme*; uma mudança de conceito real acabou de acontecer. A amostra $x_{t+\tau}$ teve seu mapeamento de verificação para um hipercubo filho já ocupado por um dado na quadtree de classe oposta; neste caso, o classificador atual é considerado obsoleto e deve ser descartado com as respectivas quadtrees. Os dados mais recentes (pertencentes à última janela) são usados para treinar o novo classificador. Para a reconstrução das quadtrees serão usadas as próximas amostras a partir da $x_{t+\tau}$. É recomendada uma quantidade mínima de cem amostras dividida, se possível, equilibrada entre as classes.

A Figura 19 apresenta o fluxograma do funcionamento do método QT. É possível observar a estrutura do classificador na qual temos como entrada $x_{t+\tau}$ e saída $\hat{y}_{t+\tau}$. A estrutura do *trigger* (gatilho) tem como entradas $y_{t+\tau}$ representada pela seta (1) e $\hat{y}_{t+\tau}$ pela seta (2), que possui como condição de acionamento a confirmação de um erro de classificação.

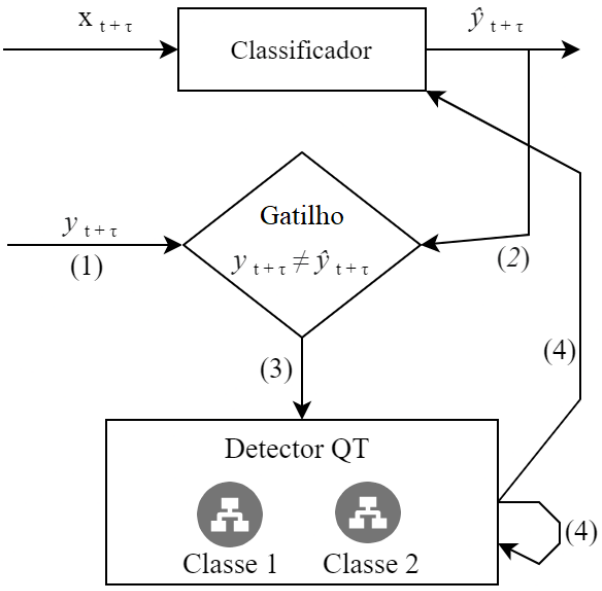


Figura 19 – Fluxograma do funcionamento do método QT.

Em (3) temos $x_{t+\tau}$ como saída do gatilho e entrada para o detector de mudança QT. O QT é responsável por verificar a posição da amostra $x_{t+\tau}$ na quadtree de classe oposta e tomar a ação de *atenção* ou de *alarme*. As ações tomadas pelo QT implicam em alterações tanto na própria estrutura do detector quanto no classificador, indicadas pelas setas com o número (4).

3.2.1 Ajuste do Parâmetro Altura para o Método QT

Determinar um bom valor para a altura da quadtree é primordial para o apropriado funcionamento do método de detecção de mudanças QT. Este problema é ainda mais difícil, visto que no ambiente de fluxo de dados contínuos, a função geradora dos dados muda com o tempo, podendo exigir uma mudança no parâmetro altura. Frente a estas condições, um novo valor para o parâmetro altura precisa ser definido sempre que uma mudança de conceito for detectada.

O parâmetro altura controla a sensibilidade do detector QT. Valores menores do parâmetro altura resultam em uma maior sensibilidade do método à detecção de mudança de conceito, o que poderia levar a um número elevado de falsos alarmes (Falsos Positivos - FP). Já valores maiores do parâmetro altura resultam em uma menor sensibilidade do método, com menor número de detecções. Esta relação que acontece entre o parâmetro altura do detector proposto e o número de detecções tem um efeito na qualidade e acurácia da classificação em fluxo de dados contínuos. A Figura 20 apresenta o número de detecções (incluindo detecções verdadeiras (TP) e falsas (FP)) feitas pelo método QT em função do valor do parâmetro altura para a base de dados SEA [Street and Kim, 2001].

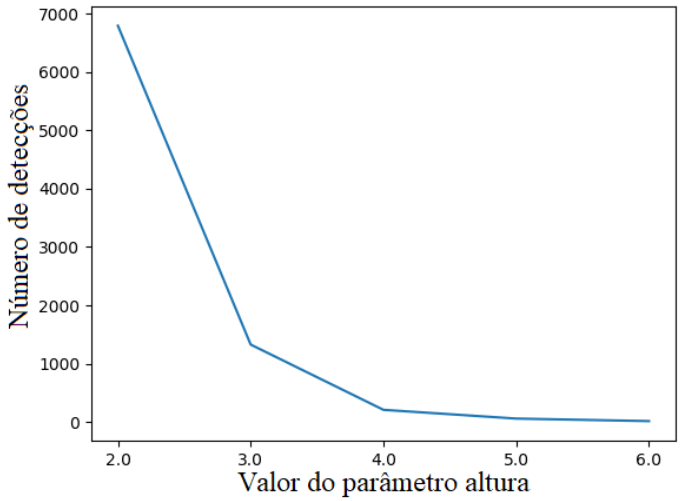


Figura 20 – A relação entre o número de detecções e o parâmetro altura.

O valor do parâmetro altura indicado no início do processo nem sempre atende de forma satisfatória todos os conceitos do fluxo de dados. O problema consiste então em estimar o valor do parâmetro altura para o método QT, usando apenas a informação dos dados mais recentes. Este parâmetro deve ser encontrado de modo a maximizar a qualidade do detector, isto é, manter a sensibilidade de detecção desejada para todos os conceitos. Isto evita que o modelo fique muito sensível em certos conceitos e pouco sensível em outros, reduzido o número de falsos alarmes sem o risco de não identificar as mudanças de conceito quando realmente acontecem.

Os trabalhos [Liu et al., 2020, Boracchi et al., 2018] apresentam em suas metodologias estratégias de divisão do espaço em *bins*, e a detecção de mudança de conceito através de histogramas. A ideia fundamental da detecção de mudança de conceito através de histogramas é converter um problema de um teste multivariado de duas amostras em um teste de adequação para distribuições multimodais, para avaliar a consistência entre os dados de treino e teste. Estes trabalhos foram as principais referências para a criação das duas estratégias, descritas a seguir, para determinar o parâmetro a altura para o método QT.

No processo de investigação da distribuição e da ocupação espacial por um conjunto de dados é comum o uso da estimativa de densidade. Nesta seção são apresentadas as duas estratégias para estimar o parâmetro altura por meio do cálculo da densidade. A Figura 21 apresenta os dois caminhos explorados.

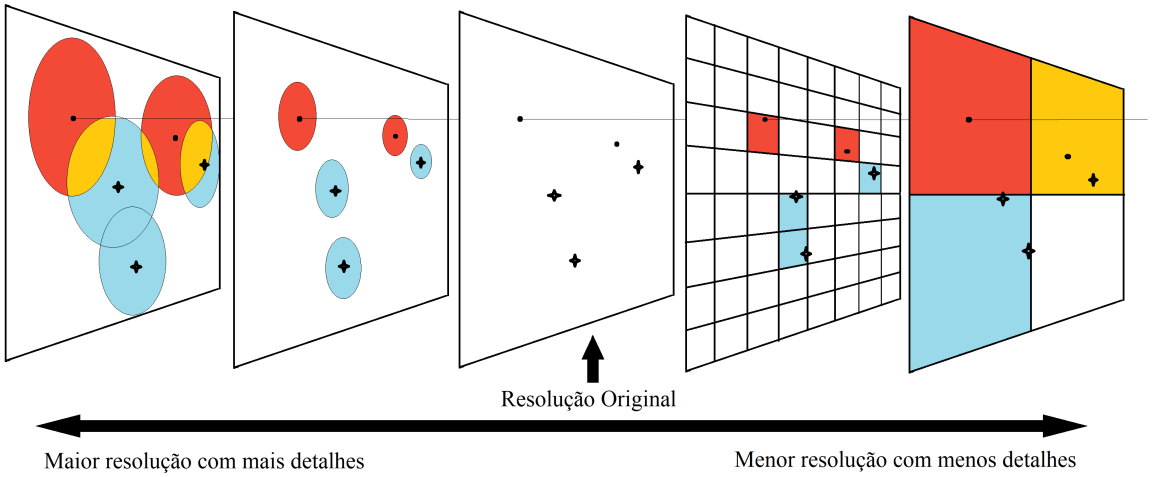


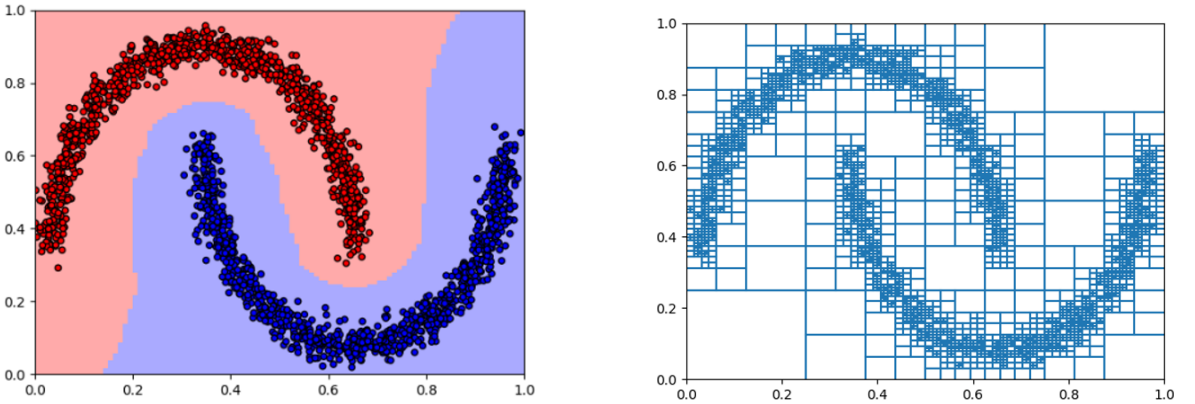
Figura 21 – Exemplo de partição espacial de maior e menor resolução [Liu et al., 2020]

Com uma menor resolução, a densidade poder ser estimada por meio da contagem dos pontos presentes em uma dada partição do espaço de características (como um *quad* da quadtree), estratégia abordada por meio do limite de dados no nó folha da quadtree. Na maior resolução, a estratégia adotada corresponde ao uso de uma função de *Kernel Density Estimation* (KDE) [Silverman, 2018] para o cálculo da densidade, para assim estimar o valor do parâmetro altura do detector.

3.2.1.1 Estratégia do Limite de Dados no Nó Folha

A quadtree fornece uma estrutura recursiva para divisão do espaço em espaços menores. Apresentado um conjunto de dados (ou lote de dados), a quadtree aumenta a resolução do espaço até que cada dado esteja em uma única partição. Uma característica importante da quadtree é a capacidade de produzir regiões com resoluções diferentes a partir de um mesmo espaço. A Figura 22a apresenta uma base de dados contendo duas

classes. A Figura 22b apresenta a quadtree correspondente, onde é possível observar as regiões de maior resolução.



(a) Base de dados com duas classes

(b) Quadtree da base de dados com duas classes

Figura 22 – Representação dos dados na quadtree

É possível observar na estrutura da quadtree (Figura 22b) que as regiões de maior altura na árvore são as regiões onde os dados estão em maior quantidade e mais próximos entre si. De forma análoga à estimativa da densidade por meio da contagem dos pontos presentes em uma partição, é possível extrair uma ideia de densidade dos dados por meio da altura de uma quadtree. A Figura 23 apresenta uma representação por níveis de saturação da Figura 22b, em que o quanto mais saturada é a cor, maior é a altura da quadtree na região.

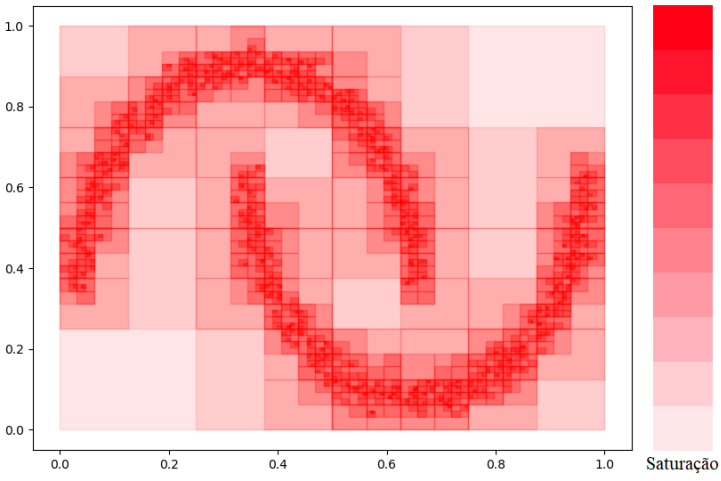


Figura 23 – Representação de uma quadtree por níveis de saturação

Como pode ser visto pela Figura 23, é possível extrair uma estimativa de densidade espacial para um conjunto de dados com o uso de uma quadtree. Regiões mais densas do espaço naturalmente ocuparão particões de maior altura na árvore, representada pelas regiões com maior saturação.

Para o problema de estimar o valor da altura ideal para o QT, o uso da altura máxima nem sempre acarreta nos melhores resultados. Geralmente o detector fica pouco sensível apresentando atrasos na detecção de mudança de conceito. A altura ideal estaria então em um meio caminho entre um valor mínimo (altura = 1) e um valor máximo (maior altura alcançada pela quadtree para um lote de dados). Com base nesta hipótese, propõe-se a criação da estratégia do Limite de Dados no Nó Folha (LDNF) que se utiliza de uma quadtree modificada para estimar o parâmetro altura h para o método QT.

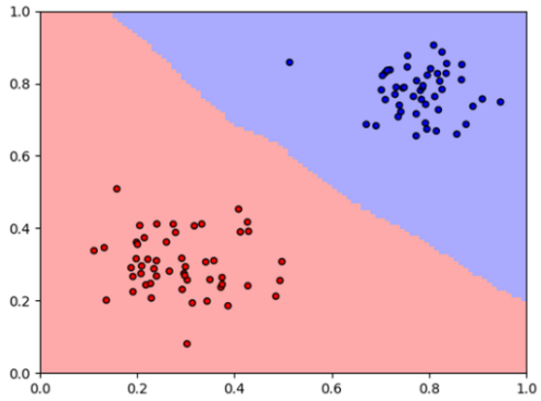
A modificação consiste na implementação de uma quadtree com o parâmetro ρ que controla o limite de dados que podem estar em um mesmo espaço para que ocorra a divisão recursiva deste espaço. Um valor de $\rho = 1$ resulta na estratégia de uma quadtree padrão. Já com um $\rho = 2$, resultaria em uma quadtree modificada que precisaria de mais de dois dados para que um espaço seja dividido, resultando em uma estrutura com menor altura. Valores baixos do parâmetro ρ implicam em um valor mais alto do parâmetro altura para o método QT, resultando em um método com baixa sensibilidade a mudanças de conceito. Já valores altos do parâmetro ρ implicam em um valor de altura menor para o método QT, resultando no aumento da sensibilidade do método QT à detecção de mudanças de conceito. Quanto à distribuição dos dados no espaço, um valor de $\rho > 1$ evita que pontos que se encontram muito próximos entre si, resulte em um valor de altura muito grande para o método QT.

É definido um valor para o parâmetro altura na fase de pré-treino e a mudança dinâmica da altura ocorre sempre que uma mudança de conceito é detectada. A quadtree modificada usa os 100 dados mais recentes para estimar um novo valor para o parâmetro altura, permitindo até ρ dados em um mesma partição sem que ocorra a divisão recursiva. Estes 100 dados são os mesmos usados nas respectivas fases de pré-treino e treino de um novo classificador. Um valor de $\rho > 1$ ajuda a mitigar o problema de dados muito próximos entre si, além de gerar resultados mais consistentes para diferentes bases de dados.

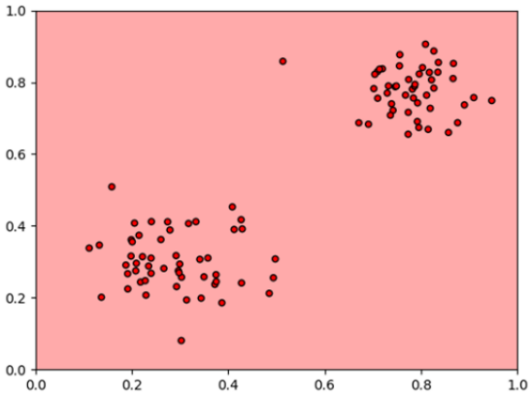
A Figura 24 exemplifica as etapas do processo para estimar o novo valor do parâmetro altura por meio da estratégia do limite de dados no nó folha. A Figura 24a apresenta os 100 dados mais recentes dispostos no espaço de características. Os mesmos dados sem a definição de suas classes são apresentados na Figura 24b. A Figura 24c apresenta os dados na quadtree modificada, onde é possível observar que existem até $\rho = 3$ dados por subespaço. Já, a Figura 24d apresenta a representação por nível de saturação, na qual a cor mais intensa representa maior altura na árvore. O valor da maior altura alcançada na quadtree modificada é adotada como parâmetro de altura do QT.

3.2.1.2 Estratégia da Densidade

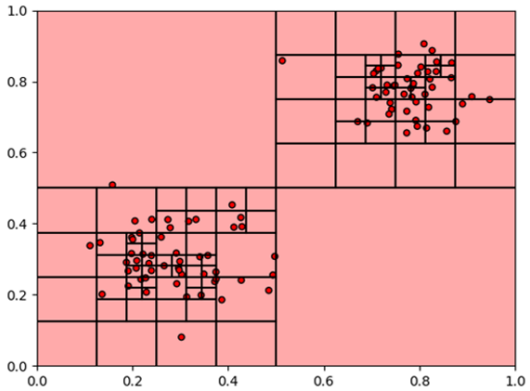
Conforme descrito em Mehta and Sahni [2004], a altura de uma quadtree pode ser calculada por meio da distância entre as duas amostras mais próximas de um conjunto



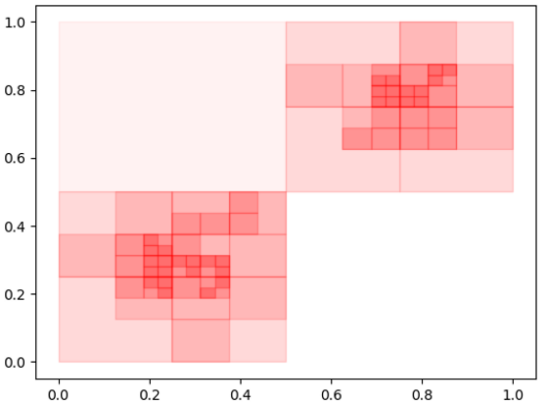
(a) Base com duas classes



(b) Eliminação das classes



(c) Construção da árvore modificada



(d) Representação da árvore por cores

Figura 24 – Representação da estratégia do limite de dados no nó folha ($\rho = 3$)

de dados. Assumindo que s é o valor da distância entre as duas amostras mais próximas, D é o comprimento de uma das d dimensões do hipercubo raiz da quadtree, a altura h é calculada pela Equação 3.2.

$$h = \left\lceil log \frac{\sqrt{d}D}{s} \right\rceil \tag{3.2}$$

Desta forma, uma estratégia para determinar uma altura plausível para o QT, com base na Equação 3.2, depende somente do valor da variável s, já que as outras duas variáveis são intrínsecas à construção da quadtree.

Por meio da Equação 3.2, é possível verificar que a altura da árvore tem uma relação inversa com a variável s, ou seja, quanto menor a distância entre os dados, maior a altura necessária para representá-los na estrutura da quadtree. No entanto, obter o s através da estimativa da distância euclidiana entre as duas amostras mais próximas do conjunto dados poderia acarretar um valor de altura muito grande para o QT, o que poderia não ser representativo ao se considerar toda a base de dados. Dessa forma, uma estimativa mais apropriada para a variável s poderia ser calculada a partir da estimativa da densidade espacial dos dados.

A ideia é mapear o valor de s em função de uma estimativa empírica da densidade dos dados através de uma PDF (*Probability Density Function*). Dado um conjunto com N dados $X \in R^d$ em que $X = (x_{i=1}, \ldots, x_{i=N})$, temos $s = g(max(pdf(X)))$, onde $max(pdf(\cdot))$ é a função responsável por extrair o valor de um ponto x_i de densidade máxima. Em seguida, a estimativa de s através da função $g(\cdot)$ poderia nos levar a um valor de altura (tomando como base a Equação 3.2) que consideraria a densidade do conjunto de dados (com foco nas regiões mais densas) e não somente a distância entre os dois pontos mais próximos. Para estimar o $max(pdf(X))$, foi adotado o método KDE (*Kernel density estimation*) [Silverman, 2018].

O cálculo do KDE se dá por meio da superposição de funções de kernel, atribuindo-se uma Gaussiana d-dimensional com centro em cada um dos elementos $x_i = (i = 1, \ldots, N)$ do conjunto de dados. Assim, podemos combinar de forma a sintetizar uma PDF para o conjunto de dados [Silverman, 2018, Liu et al., 2020]. De modo geral, um estimador de densidade por kernel pode ser descrito pela Equação 3.3.

$$\hat{f}_h(x_i) = \frac{1}{N\sigma^d} \sum_{k=1}^{N} K(x_i, x_k) \tag{3.3}$$

onde $\hat{f}_h(x_i)$ é o valor da densidade em x_i, d é o número de dimensões, σ é o prâmetro de suavização do kernel, e $K(x_i, x_k)$ é o operador do kernel Gaussiano, cuja integral $\int k(u)du$ deve ser unitária. Para a definição do parâmetro σ, que define o grau de suavização do kernel, é utilizada a Regra de Scott [Scott, 2015] conforme Equação 3.4.

$$\sigma = N^{\frac{-1}{(d+4)}} \tag{3.4}$$

desta forma, estimar a densidade sobre um conjunto de dados se torna uma tarefa que não depende de parâmetros informados.

Para determinar o parâmetro altura para o método QT a partir da densidade foi preciso encontrar uma relação entre a densidade e a distância dos dados. Foi definido uma caixa delimitadora, desta forma, os dados foram inicialmente normalizados para os limites mínimo e máximo (entre 0 e 1) com uma quantidade fixa $N = 100$ dados. Para determinar a função $g(\cdot)$ que mapeia s em função da $max(pdf(X))$, foi usada uma estratégia em que um valor de s de referência (s_{ref}) é calculado assumindo que os dados estão igualmente espaçados no espaço de dimensão d, por uma distribuição uniforme (Figura 25).

Assim, colocando N dados uniformemente distribuídos em um espaço de dimensão d, é possível calcular o valor de s_{ref}, através da distância entre dois pontos quaisquer, bem como o valor de densidade pdf_{ref}. A Figura 26 apresenta um gráfico com os valores de s_{ref} em relação ao aumento da dimensão para um conjunto de tamanho fixo $N = 100$ dados.

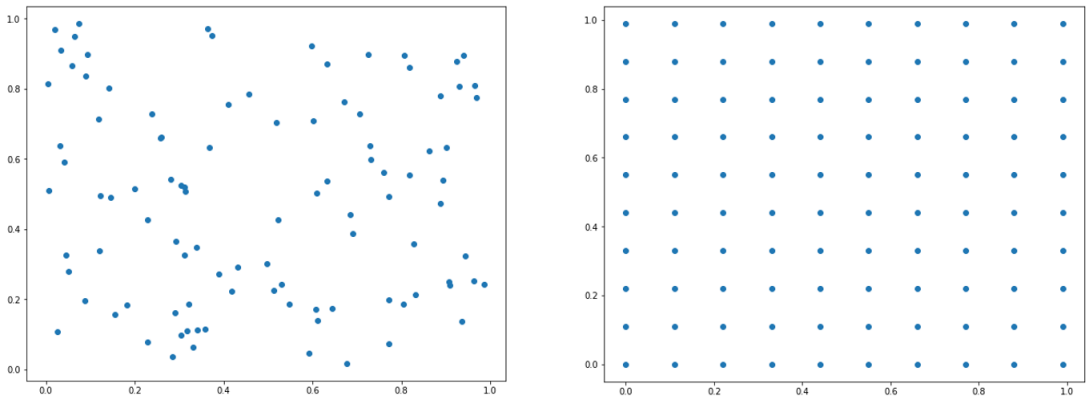


Figura 25 – Dados distribuídos de forma aleatória à esquerda, e à direita uma distribuição uniforme.

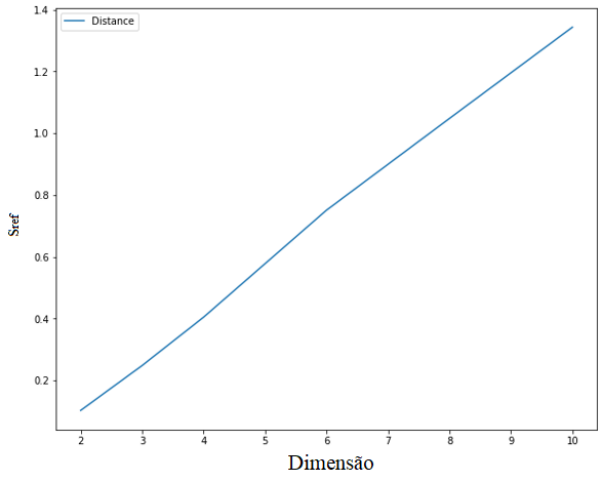


Figura 26 – Relação entre a dimensão d e s_{ref}.

Como a Figura 26 apresenta uma relação linear, através de uma equação reduzida da reta ($r(p) = mp + c$, onde m é o coeficiente angular, c é o coeficiente linear e p é a variável independente) é possível encontrar s_{ref} a partir da dimensão por meio da Equação 3.5.

$$s_{ref} = (0.16 * d) - 0.21 \tag{3.5}$$

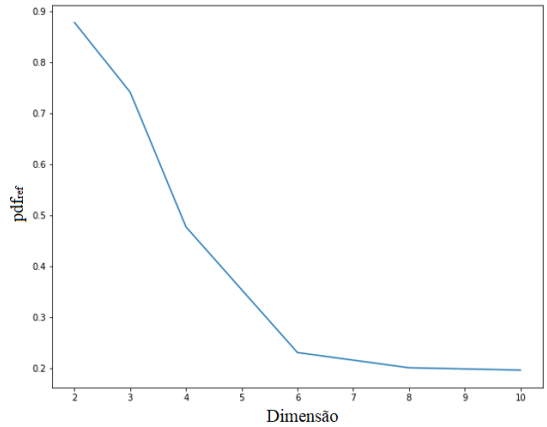
onde d é a dimensão, com $d > 1$, e s_{ref} é a distância de referência entre os dados para uma determinada dimensão. Considerando que em um lote de dados real, a distribuição desses dados será mais densa do que a distribuição de referência que é igualmente espaçada, o valor de s_{ref} é o maior valor possível de ser alcançado, o que levaria a uma quadtree com a menor altura possível (conforme Equação 3.2), para uma dada dimensão d.

A Figura 27a apresenta um gráfico com os valores de pdf_{ref} em relação ao aumento da dimensão para um conjunto de $N = 100$ dados. Com o aumento da dimensão, temos o aumento do espaçamento entre os dados e consequente diminuição do valor do pdf_{ref}. O

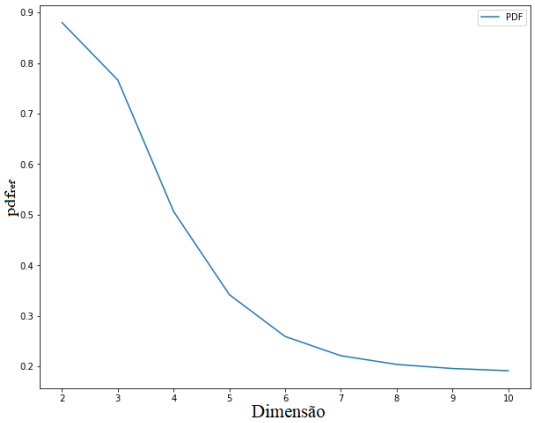
valor de pdf_{ref} e o menor valor de densidade possível para um conjunto de $N = 100$ dados de d dimensões respeitando as condições da caixa delimitadora. Observa-se que pdf_{ref} tende a zero quando a dimensão dos dados tende ao infinito, apresentando o comportamento de uma função exponencial decrescente. Por meio da equação que descreve uma curva exponencial decrescente ($e(p) = a * b^p$, onde a é a base, com o $b = (1 - v)$ em que v é a razão e p é a variável independente) é possível encontrar pdf_{ref} a partir da dimensão através da Equação 3.6.

$$Pdf_{ref} = (s_{ref}) * (1 - 0.66)^d * 53 + 0.21 * 0.99^d \tag{3.6}$$

onde d é a dimensão, com $d > 1$, e pdf_{ref} é o valor da densidade de referência para uma determinada dimensão. A Figura 27b apresenta o gráfico gerado a partir da Equação 3.6. Uma vez que a relação entre d e s_{ref} é linear, a Figura 28 apresenta a relação de proporção inversa entre s_{ref} e pdf_{ref}.



(a) Gerado a partir dos resultados obtidos da caixa delimitadora



(b) Gerado a partir da Equação 3.6.

Figura 27 – Relação entre a dimensão d e pdf_{ref}.

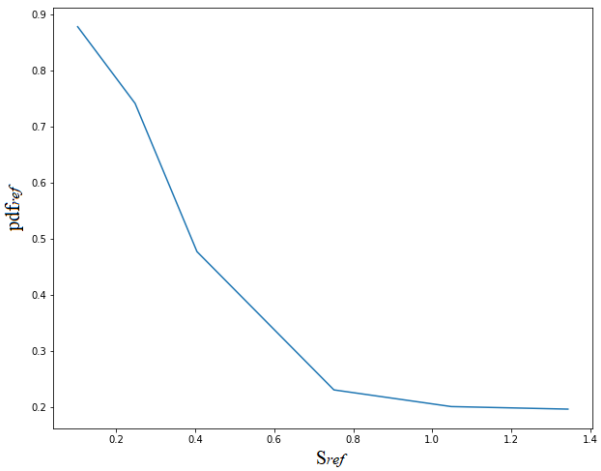


Figura 28 – Relação de proporção inversa entre s_{ref} e pdf_{ref}.

Por meio da equação que descreve uma curva exponencial decrescente, podemos representar uma fórmula geral para pdf_{ref} através da Equação 3.7.

$$pdf_{ref} = s_{ref} * b^d \tag{3.7}$$

onde s_{ref} é a base e b^d é a razão da curva exponencial para uma determinada dimensão d. Seja pdf_{calc} o valor de $max(pdf(\cdot))$ calculado para um conjunto de dados o qual se deseja descobrir o valor de s, para uma determinada dimensão d temos a Equação 3.8.

$$pdf_{calc} = s * b^d \tag{3.8}$$

Para dois pontos em uma mesma curva exponencial de mesma razão b^d é possível relacionarmos as Equações 3.7 e 3.8 como apresentado na Equação 3.9.

$$\frac{pdf_{ref}}{s_{ref}} = \frac{pdf_{calc}}{s} \Rightarrow \frac{pdf_{ref}}{pdf_{calc}} = \frac{s_{ref}}{s} \tag{3.9}$$

Movendo as variáveis de diferentes grandezas para lados opostos da equação, observa-se uma regra de três. Como a relação de proporção entre os valores de pdf e s é inversa (Figura 28), aplica-se a forma da regra de três inversa à Equação 3.9, desta forma chegamos a Equação 3.10.

$$s = \frac{pdf_{ref}}{pdf_{calc}} * s_{ref} \tag{3.10}$$

A Equação 3.10 oferece uma forma de calcular s através de valores s_{ref}, pdf_{ref} e pdf_{calc}. O valor de s é utilizado na Equação 3.2 para determinar a altura de uma quadtree.

Para criar uma função $g(\cdot)$ que determine um valor de altura para o método QT, é necessário adequar a Equação 3.10 com o propósito de gerar um valor recomendado s_{rec} para o método QT. Esta adequação consiste em modificar o valor da razão entre as densidades, sem alterar a relação que existe entre os valores de densidade e a base da curva exponencial decrescente, gerando um conjunto de possíveis valores solução S onde $s_{rec} \in S$. O valor de s_{rec} deve ser maior ou igual a s, já que s apresenta o valor de menor distância entre duas amostras para calcular a altura de uma quadtree. O valor de s_{rec} também deve ser menor ou igual que s_{ref}, já que este apresenta o maior valor possível para a menor distância entre duas amostras, desta forma, o conjunto de possíveis valores solução para s_{rec} é $S = \{s_{rec} \in R | s \leq s_{rec} \leq s_{ref}\}$.

Com objetivo de encontrar um valor de altura que gerasse resultados mais consistentes para diferentes bases de dados, chegou-se a Equação 3.11, a qual satisfaz a desigualdade $s < s_{rec} < s_{ref}$.

$$s_{rec} = \sqrt[d]{\frac{pdf_{ref}}{pdf_{calc}}} * s_{ref} \tag{3.11}$$

De forma análoga à Equação 3.2 para determinar a altura de uma quadtree [Mehta and Sahni, 2004], foram definidas duas equações (Eq. 3.12 e 3.13) para determinar a altura do detector de mudança de conceito QT.

$$h_{rt} = \left\lceil log \frac{\sqrt{d}D}{s_{rec}} \right\rceil \tag{3.12}$$

$$h_{rp} = \left\lfloor log \frac{\sqrt{d}D}{s_{rec}} \right\rfloor \tag{3.13}$$

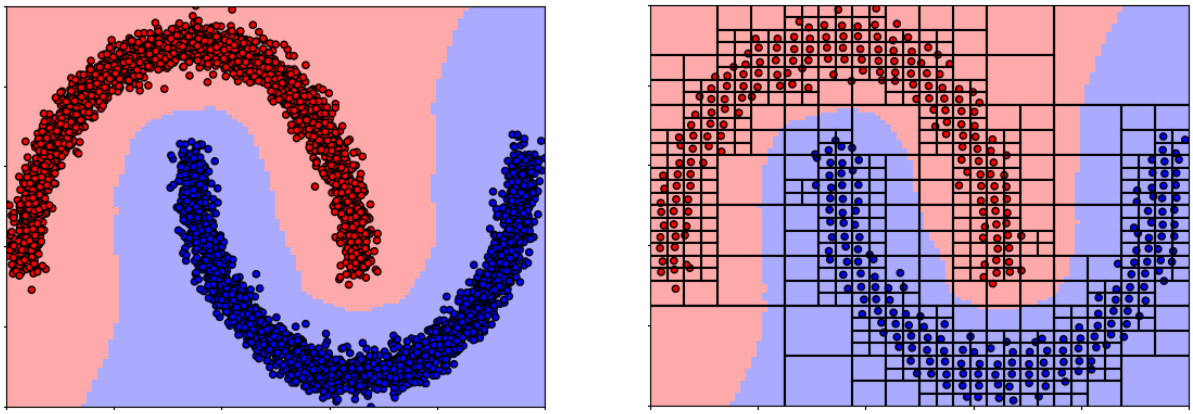
onde o valor de s_{rec} é calculado pela Equação 3.11, D é o comprimento de uma das dimensões do hipercubo raiz da quadtree, d é a dimensão dos dados, e as alturas recomendadas com valor teto h_{rt} e altura com valor piso h_{rp} para o método QT. Como o valor de s_{rec} calculado por meio da Equação 3.2 não é um número inteiro, usou-se os dois números inteiros mais próximos (teto e piso) como recomendação para o parâmetro altura.

3.3 Método de Detecção de Mudança de Conceito baseado na Saturação da Quadtree (QTS)

Além da capacidade de mapeamento do espaço que a quadtree oferece, a propriedade de sumarização de dados, a partir de um valor máximo de altura, tem o efeito de limitar a quantidade de dados que podem estar presentes na estrutura. Por exemplo, a capacidade de armazenamento de uma quadtree de altura $h = 5$ e dimensão $d = 2$ é de até 1024 amostras, i.e., $(2^d)^h$.

No contexto de fluxo de dados contínuos, a saturação na quantidade de dados acontece quando os dados oriundos de uma dada função geradora (ou conceito) não ocupam mais espaços diferentes na quadtree. Em outras palavras, a saturação ocorre quando a quantidade de dados armazenados na quadtree desacelera ou pára de crescer. Para exemplificar esse processo de saturação do espaço de uma quadtree, a Figura 29a apresenta o conjunto de dados Duas Meia Luas com 5000 amostras. Ao utilizar uma quadtree de altura $h = 5$ para mapear e armazenar os dados, a saturação se dá em torno de 360 amostras, correspondendo à quantidade máxima de dados que a estrutura foi capaz de absorver, como pode ser observado na Figura 29b.

Com base neste princípio de saturação, um segundo detector de mudança de conceito é proposto nesta Tese. QTS (Detector baseado na saturação da quadtree) assume



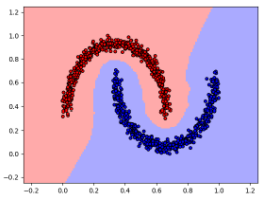
(a) Base de dados Duas meia luas. (b) Quadtree saturada

Figura 29 – Saturação do espaço feito pela base Duas meia luas

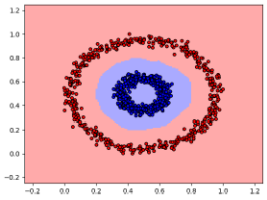
que os dados gerados a partir de uma mesma função (ou conceito) tendem a saturar o espaço mapeado pela quadtree. Dessa forma, o monitoramento de quantidade de dados presente na estrutura ao longo do tempo, pode ser um indicativo para a ocorrência de possíveis mudanças no fluxo de dados. É esperado que a chegada de dados de um novo conceito provoque uma mudança na estrutura ocupacional da quadtree, levando a um incremento da função de saturação. Caso isso ocorra e o desempenho do classificador ($\hat{f}(x) \approx p(y|x)$) seja afetado, então uma mudança de conceito deve ser alarmada.

Para ilustrar as ideias mencionadas acima, considere a base de dados Toy que é composta por quatro diferentes conceitos com mudança abrupta entre eles: Duas meia luas, Círculos, Espirais e Gaussianas (Figs. 30a, 30b, 30c e 30d, respectivamente), com cinco mil amostras em cada conceito. A Figura 30e apresenta a quantidade de dados armazenados nas quadtrees (por classe) em função do tempo. É possível observar o comportamento de saturação das quadtrees, que ocorre de maneira similar para cada um dos diferentes conceitos. Por exemplo, no intervalo de tempo $t = 0$ a $t = 4999$, a chegada de dados da função Duas meia luas, causa um aumento na quantidade de dados até um ponto de saturação que ocorre em torno de $t = 2500$ e vai até $t = 5000$. Nesse momento, a quantidade de dados das quadtrees volta a crescer de modo mais acelerada com a entrada de um novo conceito (Círculos). Pode também ser observado, para este conceito de Duas Meia Luas, que a quantidade de dados e o espaço por eles ocupado nas duas quadtrees (uma para cada classe), são semelhantes.

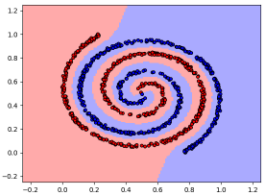
No intervalo de tempo $t = 5000$ a $t = 9999$ temos a chegada dos dados da função geradora do conceito Círculos. Pode-se observar que no instante $t = 5000$ se inicia um novo incremento na quantidade de dados, referente à chegada dos dados do novo conceito, que passam a ocupar na quadtree espaços anteriormente vazios, somando-se à quantidade de dados já existente. Para este conceito, nota-se que tanto o incremento da quantidade de dados para cada classe quanto o espaço ocupado por estes dados, são diferentes.



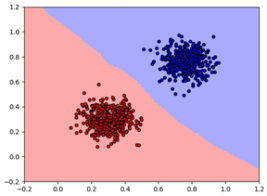
(a) Duas Meia Luas



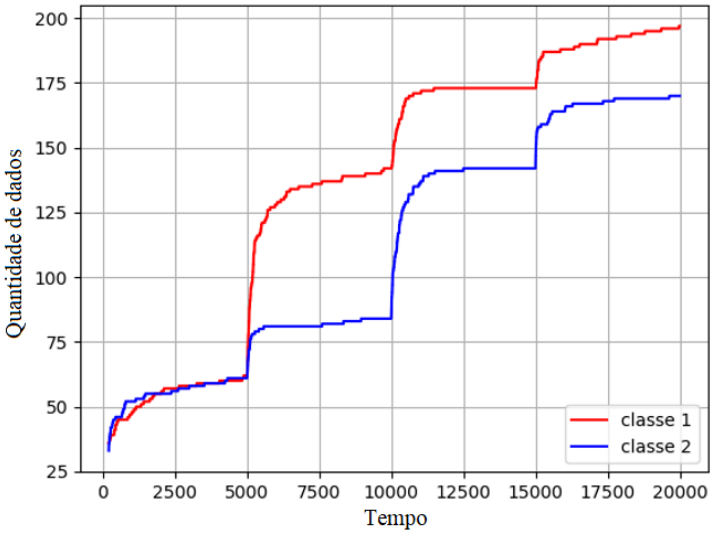
(b) Círculos



(c) Espirais



(d) Gaussianas



(e) Ocupação espacial na quadtree

Figura 30 – Ocupação espacial da base de dados Toy

Conforme observado pela Figura 30e, à medida em que os dados de um mesmo conceito se tornam disponíveis, o espaço ocupado pelas quadtrees cresce até um determinado ponto, levando à saturação da quantidade de dados presente na estrutura. As mudanças de conceito se dão justamente nos momentos em que acontecem incrementos mais acentuados nessa quantidade de dados, caracterizando o crescimento da estrutura. Este é o princípio usado pelo método (QTS) para alarmar possíveis mudanças de conceito no fluxo de dados.

No campo de processamento digital de imagens, alguns métodos de detecção de borda se valem do cálculo de derivadas da função de intensidade dos *pixels* com vistas à checagem de variações positivas ou negativas que poderiam indicar a presença de uma borda na imagem [Gonzalez, 2009]. De modo análogo, têm-se as variações da função que mede a quantidade de dados presentes nas quadtrees, que podem ser usadas para checagem de mudanças de conceito. O método QTS usa, portanto, a primeira derivada desta função para identificar variações positivas abruptas que marcam mudanças ao longo do fluxo de dados.

A primeira derivada de uma função $y = f(x)$ é a taxa de variação instantânea de y em função da variável x. A primeira derivada também pode ser interpretada como a inclinação da reta tangente ao eixo x, além de informar se a função é crescente ou decrescente [Stewart and Romo, 2017]. No contexto de uma série temporal, a primeira

derivada é definida por $f'(x) = \frac{f(x+t)-f(x)}{(x+t)-x}$, onde t é o intervalo de tempo considerado para o cálculo da derivada, como pode ser observado na Figura 31.

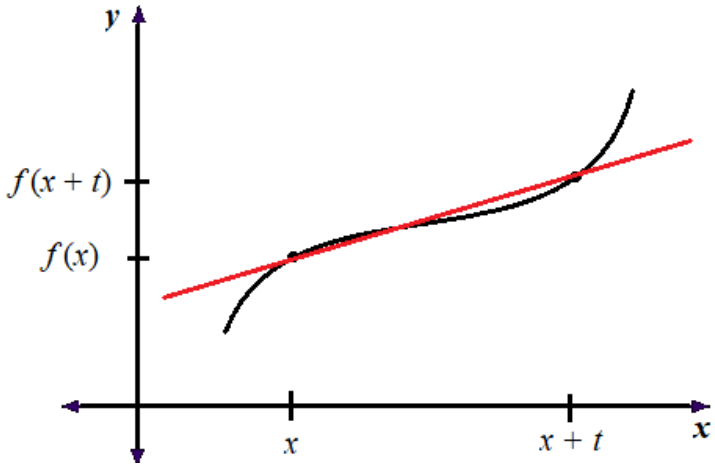


Figura 31 – Derivada de um intervalo.

O cálculo da derivada no método QTS é feito por classe, e considera duas janelas deslizantes de tamanhos diferentes. A janela grande W_g, de tamanho dinâmico, tem seu início no instante em que o primeiro dado x_g é apresentado ao classificador atual, com derivada calculada por $f'(W_g) = \frac{f(x_t)-f(x_g)}{(x_t)-x_g}$ onde o x_t é o dado mais recente apresentado ao classificador. A janela pequena W_p, possui tamanho fixo $t-p$, em que x_p é o dado mais antigo na janela e x_t é o mais recente, com derivada calculada por $f'(W_p) = \frac{f(x_t)-f(x_p)}{(x_t)-x_p} * \psi$; ψ é um parâmetro atenuador para os valores da derivada da janela pequena.

A derivada calculada para a janela grande W_g funciona como um limiar (*threshold*) para a detecção de mudanças, apresentando um comportamento mais suave devido ao tamanho da janela que é sempre crescente. A derivada da janela pequena, no que lhe concerne, é bem menos suave e tende a apresentar variações positivas abruptas toda vez que um incremento mais significativo ocorre na função que mede a quantidade de dados nas quadtrees. A Figura 32 apresenta as curvas de derivada das janelas grande e pequena para a quadtree referente a "classe 1"da Figura 30e. É possível ver que as variações de $f'(W_g)$ são mais suaves que as de $f'(W_p)$.

Com as definições das derivadas da janela pequena $f'(W_p)$ e grande $f'(W_g)$, pode-se estabelecer um limiar para checagem de uma mudança de conceito no fluxo de dados. Se em um dado instante de tempo, o valor de $f'(W_p) > f'(W_g)$, então este ponto é candidato a ser uma de mudança de conceito. O parâmetro atenuador ψ é usado para reduzir os valores de $f'(W_p)$, controlando a sensibilidade de detecção, para que incrementos pouco significativos na função que mede a quantidade de dados na quadtree não provoque valores muito elevados de derivada.

Formalmente, o detector de mudança de conceito baseado na Saturação da Quadtree (QTS) funciona da seguinte forma: seja DS_t o fluxo de dados no instante de tempo t,

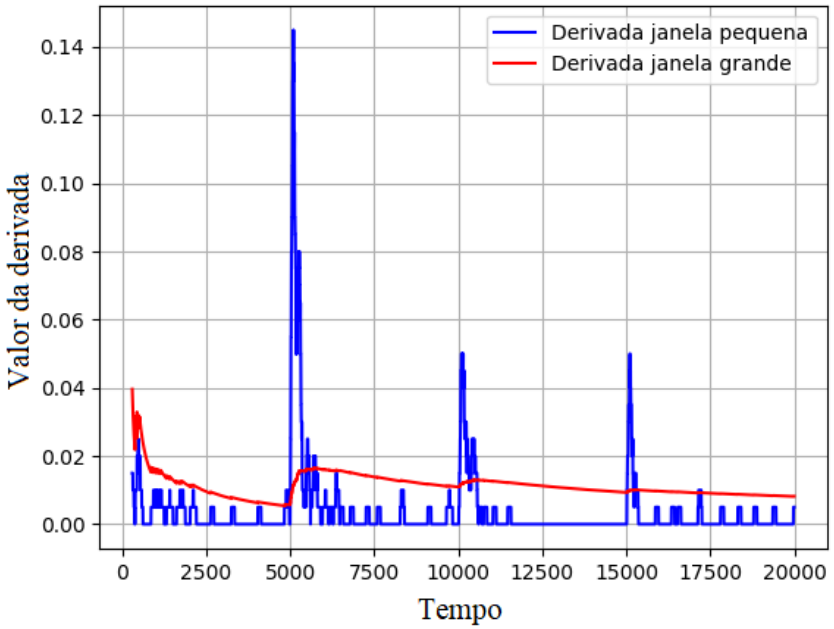


Figura 32 – curvas de derivada das janelas grande e pequena para a quadtree referente à "classe 1"da Figura 30e.

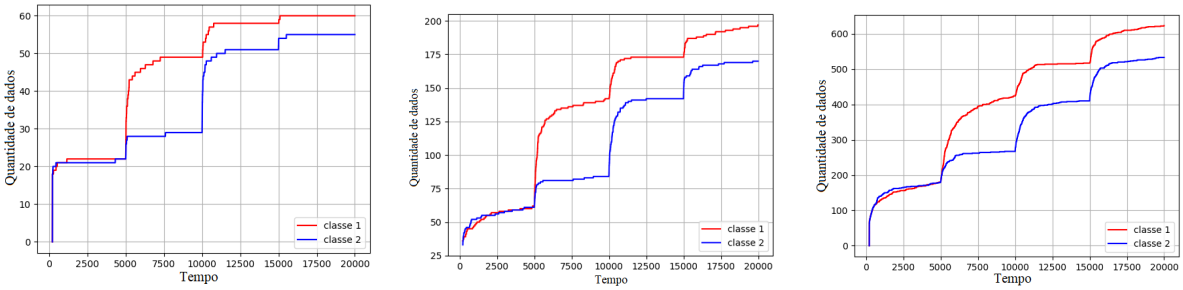
em que a função de classificação $\hat{y}_t = f_t(x_t)$ foi capaz de classificar corretamente os dados até o instante t. Se no instante de tempo $t + \tau$, o classificador apresente um erro, tal que $\hat{y}_{t+\tau} \neq f_t(x_{t+\tau})$ e, a derivada da janela pequena supere em módulo a derivada da janela grande, ou seja, $f'(W_p) > f'(W_g)$, então uma mudança de conceito é alarmada.

Detectada uma mudança de conceito, o classificador atual é considerado obsoleto e deve ser descartado com as respectivas quadtrees. As janelas grande e pequena (W_p e W_g) também são descartadas e novas serão construídas a partir de $x_{t+\tau}$. Os dados mais recentes são usados para treinar o novo classificador.

3.3.1 Ajuste dos Parâmetros do Método de Detecção QTS

O método QTS possui três parâmetros para ajuste, o parâmetro h inerente à estrutura de dados quadtree, o atenuador ψ que atua nos valores da derivada da janela pequena, e os parâmetros referentes aos tamanhos das janelas W_p e W_g. O parâmetro altura h é responsável por mapear o espaço e limitar a quantidade de dados na quadtree. Valores muito baixos de h podem causar a saturação prematura da quadtree, impedindo a detecção da mudança de conceito, já que a quadtree não permitirá mais incrementos. Valores muito altos de h dificultam a saturação do espaço onde os dados da função geradora são armazenados, deixando menos aparente o aumento no incremento de dados na quadtree onde acontece a mudança de conceito. A Figura 33, apresenta as quantidades de dados por classe para a quadtree com alturas $h = 3$, $h = 4$ e $h = 5$ para a base de dados Toy de duas dimensões.

Como dito anteriormente, a capacidade de armazenamento de uma quadtree está

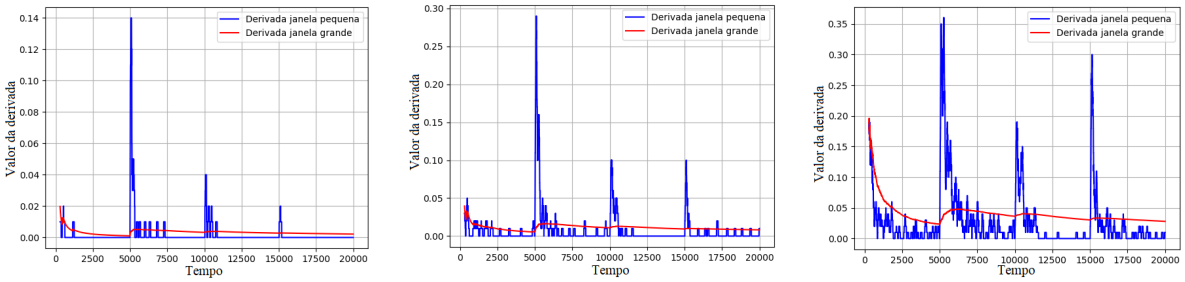


(a) Altura $h = 3$ (b) Altura $h = 4$ (c) Altura $h = 5$

Figura 33 – Quantidades de dados por classe para as quadtrees com alturas $h = 3$, $h = 4$ e $h = 5$ para a base de dados Toy.

diretamente ligada a sua dimensão d e altura h, dado por $(2^d)^h$. Podemos observar na Figura 33a, que a quadtree chega próximo da saturação com 64 amostras sumarizadas na classe 1. Já na Figura 33c, há pouco menos de um terço da classe 1 ocupando o total de dados que a quadtree oferece. Os problemas com os baixos valores de altura são o risco de saturar a estrutura da quadtree prematuramente e a diminuição do aumento no incremento da quantidade de dados na quadtree, condição chave para a detecção de mudança de conceito. Esta situação é observada entre o terceiro e quarto conceitos da Figura 33a. Para altos valores do parâmetro altura tem-se a diminuição da diferença no incremento da quantidade de dados na quadtree entre diferentes conceitos, devido entrada quase que constante de dados na quadtree. Esta situação é evidente entre o segundo e terceiro conceitos da Figura 33c. Dessa forma, as alturas mínima e máxima recomendadas de armazenamento de uma quadtree são dadas no intervalo $256 \leq (2^d)^h \leq 65536$, que compreende ao espaço mapeado capaz de atender as bases utilizadas neste trabalho (de duas a dez dimensões). Este amplo intervalo recomendado se deve a outros parâmetros que controlam a sensibilidade do método, que são discutidos a seguir.

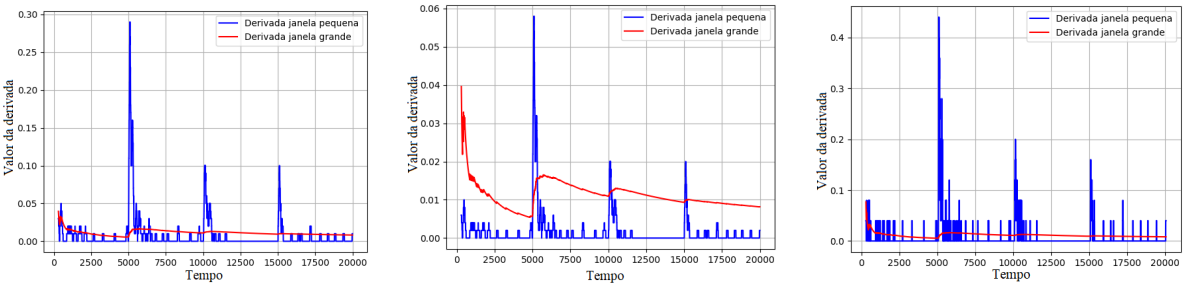
A Figura 34 apresenta os valores das derivadas referentes à Figura 33, usando os seguintes parâmetros: $\psi = 0.5$, W_g com tamanho dinâmico iniciando em x_0 e indo até x_t, e $W_g = 100$. É possível observar diferentes comportamentos dependendo do valor da altura. Como mencionado em relação aos problemas com baixos valores de altura, na Figura 34a observam-se os menores valores das derivadas onde acontecem as mudanças de conceito, em especial na última mudança. Já com altos valores de altura, como visto na Figura 34c, fica evidente a diminuição do pico nos valores da derivada na segunda mudança de conceito, devido à diminuição no incremento que acontece no início do terceiro conceito.



(a) Derivada $h = 3$ (b) Derivada $h = 4$ (c) Derivada $h = 5$

Figura 34 – Comparação das derivadas da base de dados Toy para quadtrees com diferentes alturas

O parâmetro ψ atua nos valores da derivada da janela pequena $f'(W_p)$, atenuando seu módulo, o que permite o controle da sensibilidade do detector. O parâmetro tamanho da janela pequena, $t - p$, influencia no comportamento da derivada $f'(W_p)$ e tem função semelhante ao parâmetro ψ. A Figura 35 mostra a influência destes parâmetros nos valores da derivada da janela pequena $f'(W_p)$ na base de dados Toy com altura $h = 4$. Foram alterados apenas os parâmetros ψ e W_p para demonstrar os seus efeitos.



(a) Derivada $\psi = 0.5$, $W_p = 100$ (b) Derivada $\psi = 0.1$, $W_p = 100$ (c) Derivada $\psi = 0.5$, $W_p = 25$

Figura 35 – Derivadas da base de dados Toy com diferentes valores de ψ e W_p

A Figura 35a apresenta a derivada pequena com os parâmetros $\psi = 0.5$ e $W_p = 100$, e é usada como *baseline* para esta comparação. A Figura 35b está configurada com os parâmetros $\psi = 0.1$ e $W_p = 100$, onde é possível observar que os valores da derivada $f'(W_p)$ foram reduzidos. Baixos valores de ψ diminuem a sensibilidade do QTS, já valores altos, aumentam a sensibilidade a detecção de mudança do QTS. A Figura 35c está configurada com os parâmetros $\psi = 0.5$ e $W_p = 25$, onde é possível observar altos valores da derivada $f'(W_p)$. A diminuição do tamanho da janela pequena W_p faz com que exista uma maior amplitude dos valores da derivada em suas variações, tornando o método QTS mais sensível às detecções de mudança.

3.4 Considerações Finais

Na parte inicial deste capítulo foram apresentados: (i) a metodologia criada para o mapeamento n-dimensional de dados, que possibilita que uma quadtree receba dados de mais alta dimensão; e (ii) o princípio de sumarização que limita a quantidade de dados na quadtree sem muita perda de informação. Estas duas técnicas formaram a base para a construção dos métodos QT e QTS.

O método proposto QT utiliza da análise espacial para a detecção de mudanças de conceito através de quadtrees criadas por dados de classes opostas. A utilização de uma quadtree por classe facilita a análise espacial, pois, delega ao classificador a responsabilidade de lidar com a sobreposição de dados (bases complexas com sobreposição). Caso o método QT adotasse apenas uma árvore, estratégias para lidar com dados de classes diferentes em um mesmo nó folha deveriam ser implementadas. Esta decisão também simplificou as regras criadas para a detecção de mudança de conceito, com a definição de ações simples e com escopo bem definido.

O método QT depende do valor do parâmetro altura h para o seu bom funcionamento. Foram desenvolvidas duas estratégias para o controle dinâmico de h, a estratégia do Limite de Dados no Nó Folha (LDNF) e a estratégia da Densidade (referentes as Equações 3.12 e 3.13), que apresentam caminhos distintos para tratar do problema. Por meio destas estratégias, o método QT pode se adaptar com maior precisão às mudanças de conceito.

O método QTS utiliza a capacidade de sumarização de dados da quadtree para identificar quando dados de um conceito saturam, caracterizando uma situação em que estes dados não ocupam mais espaços diferentes na estrutura, o que resulta em estagnação na quantidade de dados que é armazenada. A utilização de uma quadtree por classe facilita identificar quando esta quantidade de dados volta a crescer, o que possibilita a criação de uma nova regra para detecção de mudanças de conceito.

No próximo capítulo, tanto os métodos propostos quanto as estratégias para o controle dinâmico de h serão colocadas a prova com o intuito de validar estas ideias, explorar seus pontos fortes e fracos, e comparar os resultados alcançados frente a outros detectores.

Capítulo 4

Experimentos e Resultados

Neste capítulo são apresentadas as condições experimentais aplicadas para avaliar os métodos de detecção de mudança de conceito propostos QT e QTS, com as estratégias para determinar o valor do parâmetro altura de forma dinâmica para o QT. São apresentadas as bases de dados utilizadas nos experimentos e a forma de avaliação do desempenho. Os experimentos são realizados em duas etapas. A primeira consiste na avaliação das diferentes estratégias para determinar dinamicamente o parâmetro altura. Na segunda, etapa os métodos propostos são avaliados com outros métodos de detecção de mudança de conceito já conhecidos na literatura.

Os experimentos foram implementadas em Python usando o *Random Forest*[1] (RF) e o Naive Bayes[2] (NB) como o classificadores base. A fim de fornecer uma comparação justa entre os detectores de mudança de conceito, os parâmetros dos classificadores foram mantidos constantes em todos os experimentos. Os parâmetros do RF *number of trees in the forest* e *maximum depth of trees* foram definidos em 20 e 2 respectivamente com valor de *seed* fixo. O parâmetro do NB *nominal_ attributes* usou o valor *None*, assumindo que todos os valores de entrada são numéricos. Adotou-se a normalização das bases de dados, uma etapa de pré-treinamento utilizando as primeiras 200 amostras, e os dados são amostrados um a um ao algoritmo de aprendizado. Na ocorrência de mudança de conceito, um novo classificador é treinado usando os 100 dados mais recentes armazenados em uma janela deslizante.

A Figura 36 apresenta o fluxograma do algoritmo de aprendizado para fluxo de dados utilizado nos experimentos. O fluxograma possui em sua composição as estruturas do classificador, do detector de mudança de conceito e da janela deslizante. No classificador temos como entrada $x_{t+\tau}$ e saída $\hat{y}_{t+\tau}$. Em (1) é feito o armazenamento da mesma amostra de dado apresenta ao classificador na janela deslizante W_{100}. Em (2) é apresentado ao detector a saída do classificador. O detector é responsável por analisar $\hat{y}_{t+\tau}$ e verificar a

[1] Disponível em: https://scikit-learn.org/ .
[2] Disponível em: https://scikit-multiflow.github.io/

necessidade da ação de *atenção* (3), em que $x_{t+\tau}$ é usado para melhorar incrementalmente o classificador; ou uma ação de *alarme*, na qual foi detectada uma mudança de conceito, onde é necessário o descarte do atual detector, classificador e a redefinição de novos modelos (classificador e detector). As setas (4) e (5) indicam o uso dos dados presentes na janela para o treino do novo classificador. Em (6) é apresentada a classe esperada $y_{t+\tau}$ ao detector.

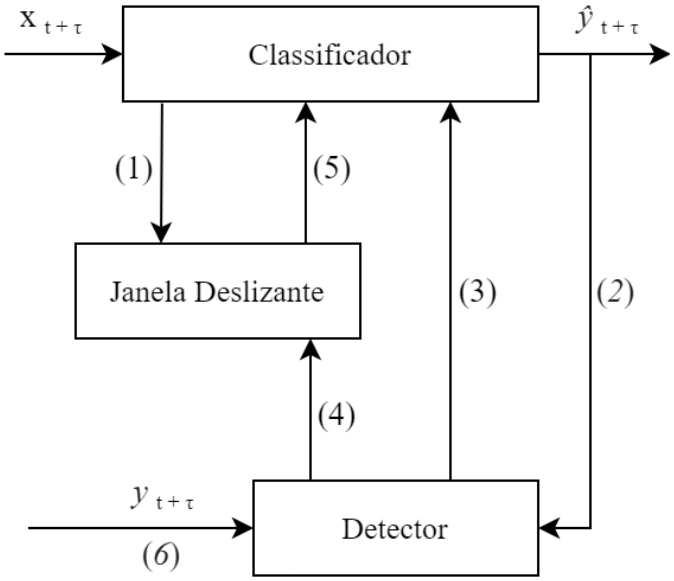


Figura 36 – Fluxograma do algoritmo de aprendizado para fluxo de dados utilizado nos experimentos.

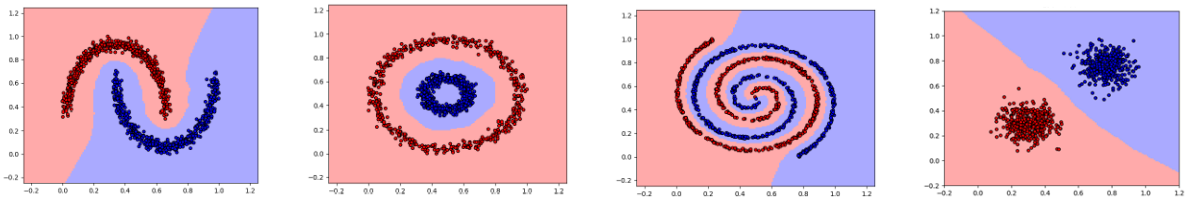
4.1 Bases de dados

Nos experimentos foram utilizadas nove bases de dados sintéticas com mudança de conceito abrupta, cinco com mudança gradual e duas bases de dados reais, todas com duas classes. Geradores de fluxo de dados[2] foram usados para criar as bases de dados com mudança de conceito abrupto SEA, SINE1, SINE2, RBF_10d, RBF_3d, MIXED e Hyperplane, e as versões com mudança de conceito gradual das bases Hyperplane_G, SINE1_G, SINE2_G, SEA_G. A mudança de conceito gradual foi feita por meio da aplicação da função sigmoidal na amostra x_t é $f(x_t) = 1/(1 + e^{-4(x_t-p)/L})$, onde $L = 1000$ é o comprimento da transição e p é a posição central da função sigmoidal. As bases de dados reais são a Electricity e a Weather que estão disponíveis no GitHub [3].

A base de dados **Toy** possui um total de 20000 amostras, duas dimensões e três mudanças de conceito abruptas causadas pelas transições entre os problemas de

[3] Disponível em https://github.com/ogozuacik/concept-drift-datasets-scikit-multiflow.

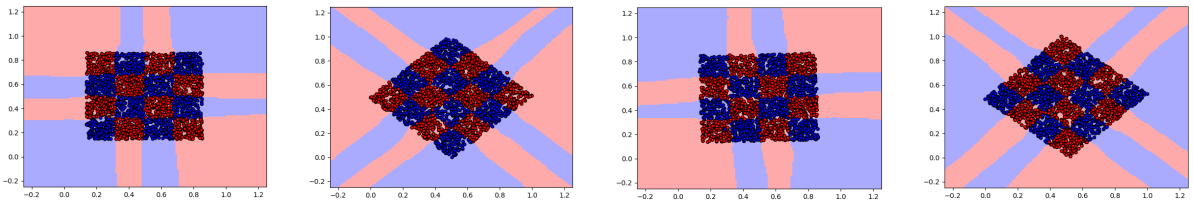
classificação bem conhecidos: (a) duas meias luas, (b) círculos, (c) espirais e (d) gaussianas, conforme ilustrado na Figura 37.



(a) Duas meia luas (b) Círculos (c) Espirais (d) Gaussianas

Figura 37 – Base de dados Toy e suas quatro diferentes funções

A base de dados **Checkerboard** [Alvarenga et al., 2021] possui 10000 amostras de duas dimensões e três mudanças de conceito abruptas com mesmo intervalo causadas pela rotação em 90° do tabuleiro no sentido anti-horário (Figura 38). A versão com mudança de conceito gradual **Checker_G** apresenta um comprimento de transição de $L = 1000$ nas mesmas posições que aconteciam as mudanças de conceito abruptas.



(a) 0° (b) 90° (c) 180° (d) 270°

Figura 38 – Base de dados Checkerboard

A base de dados **SEA** [Street and Kim, 2001] possui 60000 amostras de três dimensões. Contém duas classes desbalanceadas com 22384 e 37616 amostras cada. A base apresenta quatro conceitos diferentes, com 15000 amostras cada e 10% dos dados são ruídos. A versão com mudança de conceito gradual **SEA_G** apresenta um comprimento de transição de $L = 1000$ nas posições que aconteciam as mudanças de conceito abruptas.

A base de dados **SINE1** possui 10.000 amostras de duas dimensões e dois conceitos diferentes com 5.000 amostras cada. No primeiro conceito todos os dados abaixo da curva $y = sin(x)$ são classificados como positivos, após o desvio, as classes dos dados são invertidas. A versão com mudança de conceito gradual **SINE1_G** apresenta um comprimento de transição de $L = 1000$ na posição $t = 5000$.

A base de dados **SINE2** é composta de 10000 amostras de quatro dimensões, em que duas destas dimensões são apenas ruído. A base de dados possui dois diferentes conceitos com 5.000 amostras cada. O hiperplano de separação entre as classes é feita por $y < 0,5 + 0,3sin(3\pi x)$ em que após a mudança de conceito as classes são invertidas. A versão com mudança de conceito gradual **SINE2_G** apresenta um comprimento de transição de $L = 1000$ na posição $t = 5000$.

A base de dados **RBF_10d** foi criada usando um gerador de fluxo de dados de Função de Base Radial aleatório (RBF). Neste gerador, centroides são criados em posições aleatórias e são atribuídos a cada um deles valores de desvio padrão, peso e classe. Novas amostras são geradas em torno dos centroides, de forma que os centroides com os maiores pesos possuam mais amostras. A base de dados **RBF_10d** possui 20000 amostras de dez dimensões e dois conceitos diferentes de mesmo tamanho, criados a partir de duas funções RBF diferentes. A base de dados **RBF_3d** é composta de 10000 amostras de três dimensões e dois conceitos diferentes com 5.000 amostras cada, criada a partir de um gerador de fluxo de dados de Função de Base Radial aleatório (RBF).

A base de dados **MIXED** possui 10.000 amostras de quatro dimensões e dois conceitos diferentes com 5.000 amostras cada. Das quatro dimensões, duas são compostas por dados booleanos.

A base de dados **Hyperplane** é composta de 10000 amostras de quatro dimensões, com dois diferentes conceitos de 5.000 amostras cada. O hiperplano que separa as classes muda de posição nos diferentes conceitos. A versão com mudança de conceito gradual **Hyperplane_G** apresenta um comprimento de transição de $L = 1000$ na posição $t = 5000$.

A base de dados real **Electricity** é formada a partir de dados coletados do mercado de eletricidade da Austrália. Neste mercado, os preços flutuam de acordo com a oferta e demanda do mercado, definidos a cada cinco minutos. Esta base possui 45312 amostras de oito dimensões e exibe dependências temporais [Zliobaite, 2013]. O objetivo é prever se o preço aumentará. Não foi informado o tipo de mudança de conceito que a base apresenta.

A base de dados real de previsão de tempo de Nebraska (**Weather**) foi formada a partir de medições diárias com várias informações meteorológicas em um período de 50 anos [Elwell and Polikar, 2011]. Esta base de dados tem um total de 18159 amostras e oito dimensões, em que os atributos são medições diárias de variáveis meteorológicas como a temperatura, a pressão, a velocidade do vento, entre outras. O objetivo é prever se vai chover. Não foi informado o tipo de mudança de conceito que a base apresenta.

4.2 Avaliação da Performance

O desempenho dos detectores de mudança testados foi avaliado sob duas perspectivas distintas. A primeira considera a acurácia global medida ao longo do tempo por meio da metodologia de avaliação teste-então-treine [Gama et al., 2014]. Esta metodologia permite que todas as amostras da base de dados sejam utilizadas em um primeiro momento para testar e posteriormente para treinar o classificador. Esta primeira perspectiva busca uma avaliação da sinergia entre classificador e detector de mudança.

A segunda perspectiva busca avaliar o detector de mudança de conceito quanto

a sua capacidade de detecção. A avaliação utiliza dos instantes em que as mudanças do conceito realmente acontecem e, desta forma, mede a capacidade dos detectores de identificar esta mudança. Conforme recomendado em [Gama et al., 2014, Yu et al., 2019], definimos o Verdadeiro Positivo (TP) como uma detecção dentro do intervalo de atraso fixo após a ocorrência de uma mudança de conceito e o espaço até o fim do mesmo, o Falso Negativo (FN) como a falta da detecção dentro do intervalo de atraso. O Falso Positivo (FP) como uma detecção fora do intervalo de atraso ou uma detecção extra dentro do intervalo de atraso, e o Verdadeiro Negativo (TN) como todo o espaço fora do intervalo de atraso. A Figura 39 apresenta um fluxo de dados do instante de tempo $t = 0$ ao instante t, onde a barra vertical em vermelho indica o local onde acontece a mudança de conceito, a barra vertical tracejada indica o fim do intervalo de atraso, e as setas verticais indicam o local onde foram identificadas as mudanças de conceito por um detector qualquer. Também são indicados os intervalos onde são contabilizados os valores de TP, FN, FP e TN a cada instante de tempo para o cálculo das métricas de qualidade.

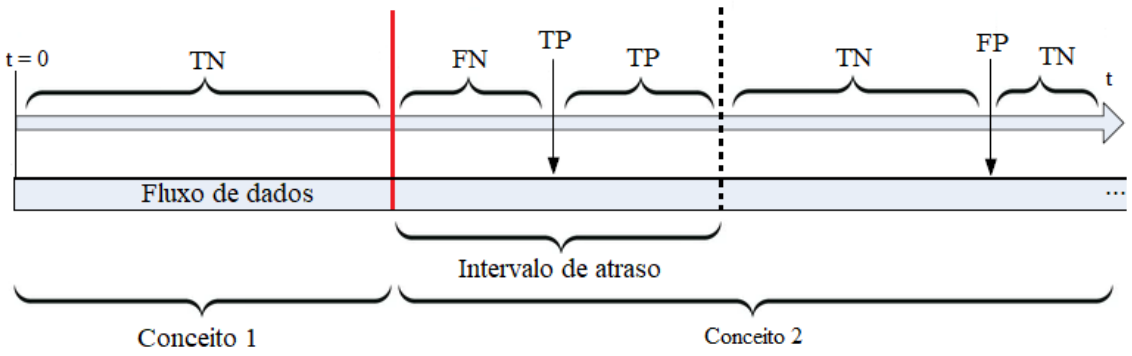


Figura 39 – Os intervalos utilizados para o cálculo das métricas de qualidade.

A qualidade da detecção é então avaliada pelas métricas de $revocação = TP/(TP + FN)$, $precisão = TP/(TP + FP)$, e $f1\text{-}score = 2 * (precisão * revocação)/(precisão + revocação)$. Neste contexto, a *revocação* se refere ao quão responsivo o detector é às mudanças de conceito, enquanto *precisão* refere-se ao quão preciso o detector é em identificar uma mudança de conceito.

Normalmente, o número de unidades de tempo dentro do intervalo de atraso é muito menor quando comparado ao número de unidades de tempo fora desse intervalo, o que tende a causar um viés nas métricas de qualidade de detecção. Para superar esse viés, adotamos uma estratégia proposta em [Swamidass et al., 2010] para compensar o desequilíbrio de unidades de tempo, intensificando o custo de um falso positivo (FP) durante o processo de detecção através de uma transformada calculada pela Equação 4.1.

$$fFP = \frac{fFPR * TN}{1 - fFPR} \tag{4.1}$$

em que fFP é a transformada do falso positivo, TN é o número de verdadeiros negativos e o $fFPR$ é a transformada da taxa de falsos positivos. O valor de $fFPR$ é calculado por meio da Equação 4.2.

$$fFPR = \frac{1 - e^{-\alpha * FPR}}{1 - e^{-\alpha}} \tag{4.2}$$

em que FPR é a taxa de falsos positivos ($FPR = FP/(FP + TN)$), e o α é razão entre o tamanho (em número de unidades de tempo) da região fora do intervalo de atraso e da região dentro do intervalo de atraso. A transformada do falso positivo (fFP) é usada para obter os valores de $Precisão = TP/(TP + fFP)$ e *f1-score*.

Nesta segunda perspectiva, além dos valores de *Revocação*, *Precisão* e *F1-score* também serão avaliados o número total de detecções feitas (*Detecções*), a quantidade de detecções feitas dentro do intervalo de atraso (*Mudanças*), e o atraso médio dessas detecções (*Atraso*). É importante ressaltar que o *Atraso* contabiliza apenas o atraso das detecções feitas dentro do espaço de atraso. Dessa forma, não se deve confundir com os valores de Falso Negativo (FN) utilizados nos cálculos de *Revocação*, onde o FN é contabilizado mesmo sem a detecção dentro do intervalo de atraso.

4.3 Experimentos 1 - Estratégias de Altura Dinâmica

Nesta seção são apresentados os resultados experimentais do método QT utilizando as estratégias de altura dinâmica. O objetivo desta primeira etapa experimental é avaliar se o QT utilizando as estratégias de altura dinâmica pode produzir melhores resultados que o QT utilizando altura fixa. Admite que um valor fixo para o parâmetro altura, indicado no início do processo, nem sempre atende de forma satisfatória a todos os conceitos. Como linha de base para as estratégias dinâmicas, o método QT foi utilizado com o valor de altura fixo (Fixo) $h = 4$. A estratégia de altura dinâmica do Limite de Dados no Nó Folha (LDNF) utiliza os valores de $\rho = 1$, $\rho = 2$ e $\rho = 3$ para LDNF1, LDNF2 e LDNF3 respectivamente. Na estratégia da Densidade são utilizadas as Equações 3.13 (Eq. Piso) e 3.12 (Eq. Teto). O intervalo de atraso adotado nas bases de dados sintéticas com mudanças de conceito abrupto Toy, SINE1 e SINE2 são de 500 amostras. Para as bases Checkerboard, Hyperplane, RBF_3d e MIXED são de 1000 amostras e para as bases SEA e RBF_10d são de 2000 amostras. Estes diferentes tamanhos para o intervalo de atraso se deve a diferenças na complexidade das bases, como a dimensionalidade, presença de ruído e a sobreposição de classes.

Para as bases sintéticas com mudanças de conceito abrupto são analisados os valores de *Acurácia*, *Revocação*, *Precisão* e *F1-score* já descritos anteriormente. Também são analisados o número total de *Detecções* que o QT realizou para cada base de dados, a média da *Altura* adotada pelas estratégias de altura dinâmica, a quantidade de *Mudanças*

de conceito que foram detectadas dentro do intervalo de atraso e o *Atraso* médio dessas detecções. Não confundir a coluna *Atraso* com os valores de Falso Negativo (FN) utilizados nos cálculos da *Revocação*. A coluna *Atraso* é contabilizada apenas quando existe a detecção dentro do intervalo de atraso.

A Tabela 1 apresenta os resultados das estratégias de altura dinâmica em todas as bases de dados com mudança de conceito abrupto usando o classificador *Random Forest* (RF). Na parte final da tabela têm-se os resultados médios para cada uma das métricas que foram avaliadas.

O valor da *Revocação* se difere bastante entre as estratégias. A *Revocação* está ligado diretamente ao tempo de resposta do detector à mudança de conceito, ou seja, o quão responsivo o detector é à mudança que aconteceu. As estratégias que obtiveram os melhores resultados foram as Eq. Piso e LDNF3, exatamente as estratégias mais sensíveis às mudanças de conceito. Dentre as estratégias não paramétricas, a Eq. Piso possui os valores de Altura mais baixos que a Eq. Teto, o que faz com que a Eq. Piso seja mais sensível à detecção de mudança de conceito, o mesmo acontece com o LDNF3 frente ao LDNF2 e LDNF1. O quanto menor o valor do parâmetro altura do método QT, mais sensível à detecção ele se torna.

O valor da *Precisão* está ligado diretamente às detecções realizadas dentro do intervalo de atraso e o total de detecções realizadas. Esta métrica é uma das mais importantes para balizar a escolha da estratégia de altura dinâmica a ser aplicada. Isso porque ela mede a capacidade do detector em lidar com falsos alarmes, que podem acarretar numa atualização desnecessária do modelo. A estratégia LDNF2 apresentou os melhores valores de *Precisão*, deixando de detectar apenas uma mudança de conceito. O LDNF1 obteve o pior valor de *Precisão*, apresentando sucesso em apenas 8 das 15 Mudanças.

O *F1-score* apresenta a média harmônica entre *Precisão* e *Revocação*. Observa-se que as estratégias LDNF2, LDNF3 e Eq. Teto obtiveram os melhores resultados para *F1-score*. Porém, entre essas 3 estratégias, destaca-se a LDFN2 com um valor de *Precisão* superior às demais, demonstrando uma melhor habilidade em lidar com os falsos positivos.

Quanto ao número de *Detecções*, o LDNF1 apresentou a menor contagem, mas infelizmente obteve os piores valores das métricas observadas. O maior número de *Detecções* foi obtido pela Eq. Piso, o que reflete diretamente o valor elevado da sua métrica *Revocação*. O que se observa diante das estratégias de altura dinâmica é que parece existir uma correlação positiva entre a número total de *Detecções* e métricas como *Revocação* e *Acurácia*, de forma que quanto mais sensível for o detector, melhor serão essas métricas. Em contrapartida, uma maior sensibilidade do detector tende a refletir negativamente na sua *Precisão* e, consequente habilidade de lidar com falsos alarmes. Com base nesses argumentos, a escolha da estratégia de altura dinâmica (ou de qualquer detector) deve considerar um compromisso entre a sua precisão e número de detecções, de forma a se

Base	Detector	Acurácia	Revocação	Precisão	F1-score	Detecções	Altura	Mudanças	Atraso
Toy	Fixo	93.36	**0.9973**	**0.9757**	**0.9864**	3	4.00	3/3	1.33
	LDNF1	92.45	0.6662	0.9642	0.7879	3	7.50	2/3	0
	LDNF2	92.81	0.7902	0.9696	0.8708	3	6.25	3/3	104.66
	LDNF3	**93.50**	0.9919	0.9756	0.9837	3	4.75	3/3	4
	Eq. Piso	93.36	**0.9973**	**0.9757**	**0.9864**	3	4.00	3/3	1.33
	Eq. Teto	**93.50**	0.9919	0.9756	0.9837	3	5.00	3/3	4
SEA	Fixo	81.16	0.8951	0.7645	**0.8247**	181	4.00	3/3	209
	LDNF1	78.41	0.2315	**0.8020**	0.3593	38	4.94	1/3	1216.5
	LDNF2	**81.27**	0.8685	0.6559	0.7473	296	3.44	3/3	261.33
	LDNF3	81.11	0.9615	0.5041	0.6615	599	3.02	3/3	76
	Eq. Piso	80.74	0.9612	0.3569	0.5206	1056	2.40	3/3	76
	Eq. Teto	81.20	**0.9618**	0.7000	0.8103	269	3.49	3/3	76
SINE1	Fixo	93.04	**0.9819**	0.4160	0.5844	35	4.00	1/1	9
	LDNF1	**93.71**	0.5531	**0.8284**	0.6633	3	7.00	1/1	223
	LDNF2	92.97	**0.9819**	0.5587	**0.7122**	20	4.76	1/1	9
	LDNF3	93.40	**0.9819**	0.5016	0.6640	35	4.15	1/1	9
	Eq. Piso	91.74	0.9818	0.2347	0.3789	78	3.26	1/1	9
	Eq. Teto	93.45	0.9478	0.5377	0.6862	21	4.50	1/1	26
SINE2	Fixo	**90.69**	0.6152	**0.8896**	**0.7274**	2	4.00	1/1	192
	LDNF1	89.86	0.6152	0.8009	0.6959	4	3.60	1/1	192
	LDNF2	89.40	0.0000	0.0000	0.0000	5	3.16	0/1	n/a
	LDNF3	88.00	**0.9592**	0.3126	0.4715	52	2.18	1/1	20
	Eq. Piso	85.39	**0.9592**	0.1665	0.2838	112	2.00	1/1	20
	Eq. Teto	89.38	0.8757	0.6186	0.7250	14	3.00	1/1	62
RBF_10d	Fixo	64.02	0.0000	0.0000	0.0000	0	4.00	0/1	n/a
	LDNF1	64.02	0.0000	0.0000	0.0000	0	5.00	0/1	n/a
	LDNF2	**66.21**	0.7908	0.9669	0.8700	6	3.14	1/1	418
	LDNF3	64.73	**0.9829**	0.7969	**0.8802**	55	1.37	1/1	34
	Eq. Piso	64.63	0.7907	**0.9358**	0.8572	12	3.00	1/1	418
	Eq. Teto	64.02	0.0000	0.0000	0.0000	0	4.00	0/1	n/a
Checkerboard	Fixo	62.78	0.9598	0.5796	0.7227	173	4.00	3/3	19.66
	LDNF1	**68.70**	0.5347	**0.9791**	0.6916	3	8.00	2/3	98.5
	LDNF2	66.22	0.8517	0.8502	0.8510	39	5.32	3/3	151.66
	LDNF3	65.00	0.9661	0.7279	0.8302	92	4.54	3/3	16.66
	Eq. Piso	62.78	0.9598	0.5796	0.7227	173	4.00	3/3	19.66
	Eq. Teto	65.06	**0.9710**	0.7763	**0.8628**	72	5.00	3/3	19.33
Hyperplane	Fixo	84.47	0.0000	0.0000	0.0000	3	4.00	0/1	n/a
	LDNF1	**84.73**	0.0000	0.0000	0.0000	4	4.20	0/1	n/a
	LDNF2	84.10	0.8184	0.7620	0.7892	28	2.82	1/1	181
	LDNF3	83.67	0.8179	0.5639	0.6676	68	2.20	1/1	181
	Eq. Piso	82.37	**0.9959**	0.4319	0.6025	136	2.00	1/1	4
	Eq. Teto	84.15	0.8184	**0.8033**	**0.8108**	22	3.00	1/1	181
RBF_3d	Fixo	85.65	0.9098	0.3050	0.4569	207	4.00	1/1	88
	LDNF1	85.17	0.0000	0.0000	0.0000	0	9.00	0/1	n/a
	LDNF2	87.24	0.7833	**0.9251**	**0.8483**	7	7.00	1/1	216
	LDNF3	87.24	0.4178	0.7302	0.5315	17	5.83	1/1	581
	Eq. Piso	87.08	**0.9112**	0.4709	0.6209	108	4.37	1/1	88
	Eq. Teto	**87.51**	0.8906	0.7647	0.8229	30	5.45	1/1	109
MIXED	Fixo	90.71	0.9919	0.8516	**0.9164**	19	4.00	1/1	8
	LDNF1	**91.33**	0.0000	0.0000	0.0000	3	6.75	1/1	54
	LDNF2	90.83	0.9969	**0.8728**	0.9308	16	4.11	1/1	3
	LDNF3	89.31	0.9979	0.7409	0.8504	38	3.10	1/1	2
	Eq. Piso	86.81	**1.0000**	0.3689	0.5390	174	2.00	1/1	0
	Eq. Teto	88.64	0.9979	0.6795	0.8085	51	3.00	1/1	2
Médias	Fixo	82.88	0.7057	0.5313	0.5799	69.22	4.00	13/15	84.56
	LDNF1	83.15	0.2890	0.4861	0.3553	6.44	6.22	8/15	297.33
	LDNF2	**83.45**	0.7646	**0.7290**	**0.7355**	46.67	4.44	14/15	168.08
	LDNF3	82.88	0.8975	0.6504	0.7267	106.56	3.46	15/15	102.63
	Eq. Piso	81.66	**0.9508**	0.5023	0.6124	205.78	3.00	15/15	77.04
	Eq. Teto	82.99	0.8283	0.6506	0.7234	53.56	4.05	14/15	59.92

Tabela 1 – Resultados das estratégias de altura dinâmica usando o classificador *Random Forest* para as bases de dados com mudanças abruptas.

obter um detector robusto a falso positivos e que ainda assim seja capaz de apontar a maior quantidade possível de mudanças de conceito existentes. Para o caso da Tabela 1, a estratégia LDNF2 é a que mais se aproxima das propriedades acima apontadas.

A coluna *Altura* apresenta os valores da média das alturas sugeridas pelas estratégias de altura dinâmica. É desejado que o método QT se adapte da melhor forma possível aos diferentes conceitos apresentados pelo fluxo, de forma que seja capaz de identificar o mais breve possível suas mudanças. O parâmetro altura é responsável por controlar a sensibilidade do método QT, de forma que, o quanto menor o valor da altura, mais sensível à detecção ele se torna. É possível observar na base MIXED para a estratégia LDNF3 com média de 3.10, que o método detectou 38 vezes, já para a LDNF2 que ele detectou 16 vezes com média de 4.11, por fim, que a estratégia LDNF1 com as médias de 6.75, detectou 3 vezes. Como mencionado anteriormente, é desejável que as alturas sugeridas pela estratégia de altura dinâmica resulte no máximo de *Mudanças* detectadas com uma quantidade mínima de falsos positivos.

A coluna *Mudanças* apresenta as detecções feitas dentro do espaço de atraso. As estratégias LDNF3 e Eq. Piso conseguiram detectar todas as mudanças de conceito, seguidas das LDNF2 e Eq. Teto que deixaram de detectar apenas uma. É importante que o método seja capaz de detectar o máximo de mudanças possíveis, pois, isso faz com que o classificador esteja treinado no conceito atual.

A coluna *Atraso* apresenta apenas os valores do atraso médio das detecções feitas dentro do intervalo de atraso, de forma que o valor *n/a* é apresentado quando não foram feitas detecções dentro do intervalo de atraso. As estratégias não paramétricas, Eqs. Teto e Piso, apresentaram os menores atrasos na detecção de mudança de conceito.

A Tabela 2 apresenta os resultados das estratégias de altura dinâmica para as bases sintéticas com mudança de conceito gradual e bases reais usando o classificador *Random Forest* (RF). É importante esclarecer que nas bases de dados com mudanças de conceito graduais e em bases reais não são especificadas ou informadas as posições onde ocorrem as mudanças de conceito. Desta forma, não é possível fazer o cálculo das métricas de qualidade como a *Revocação*, a *Precisão* e o *F1-score*.

Neste contexto limitado, as métricas utilizadas para a análise das bases graduais e das reais são a *Acurácia* e o número de *Detecções*. Parte-se do pressuposto de que um detector que possui um maior valor de acurácia com um menor número de detecções é aquele que encontra os melhores locais de detecção, os quais maximizam a manutenção/melhoria da acurácia do classificador.

Quanto aos resultados de *Acurácia* obtidos na Tabela 2, a estratégia de altura dinâmica LDNF2 obteve a melhor média entre as bases com mudanças graduais, e com a segunda menor quantidade de *Detecções*. Entre as bases reais, a estratégia da Eq. Piso

Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	Fixo	92.89	172		Fixo	**89.46**	2
	LDNF1	**93.39**	10		LDNF1	**89.46**	2
Hyperplane_G	LDNF2	93.22	78	SINE2_G	LDNF2	88.75	8
	LDNF3	93.12	148		LDNF3	86.98	48
	Eq. Piso	92.56	278		Eq. Piso	84.36	117
	Eq. Teto	93.05	132		Eq. Teto	88.38	13
	Fixo	**82.74**	142		Fixo	61.97	184
	LDNF1	79.58	20		LDNF1	**65.21**	8
SEA_G	LDNF2	82.44	269	Checker_G	LDNF2	64.59	37
	LDNF3	82.24	526		LDNF3	63.84	100
	Eq. Piso	81.58	648		Eq. Piso	61.97	184
	Eq. Teto	82.41	253		Eq. Teto	63.29	91
	Fixo	91.79	51		Fixo	83.77	110.20
	LDNF1	92.12	4		LDNF1	83.95	8.80
SINE1_G	LDNF2	**92.20**	26	Médias	LDNF2	**84.24**	83.60
	LDNF3	92.01	39		LDNF3	83.64	172.20
	Eq. Piso	90.79	95		Eq. Piso	82.25	264.40
	Eq. Teto	91.99	34		Eq. Teto	83.82	104.60

Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	Fixo	89.04	181		Fixo	**73.67**	45
	LDNF1	**89.59**	11		LDNF1	73.20	2
Electricity	LDNF2	88.68	46	Weather	LDNF2	73.31	31
	LDNF3	89.04	181		LDNF3	73.46	56
	Eq. Piso	**89.59**	5		Eq. Piso	73.00	6
	Eq. Teto	89.17	2		Eq. Teto	73.16	0

Tabela 2 – Resultados das estratégias de altura dinâmica usando o classificador *Random Forest* em bases graduais e reais.

e LDNF1 obtiveram os melhores resultados na base *Electricity* com 5 e 11 detecções respectivamente. Já na base *Weather*, as estratégias com os melhores resultados foram a Fixo e a LDNF3, com 45 e 56 detecções respectivamente. Entre as bases graduais e reais é notado os diferentes níveis de sensibilidade da estratégia de altura dinâmica LDNF pelo número de *Detecções* que cada uma apresenta, tal que o mesmo acontece ao se comparar os números de detecções obtidos pelas Eqs. Teto e Piso.

As Tabelas 3 e 4 apresentam os resultados das estratégias de altura dinâmica usando o classificador *Naive Bayes*. A Tabela 3 possui os resultados das bases de dados sintéticas com mudança de conceito abruptas, já a Tabela 4 os resultados das bases de dados graduais e reais.

Analisando a Tabela 3 com base nos principais critérios de qualidade apontados anteriormente, que são: a precisão do detector, aliado a sua capacidade de recuperar a maior quantidade possível de mudanças de conceito reais, apontam-se como estratégias mais promissoras LDNF3 e Eq. Teto, seguido da estratégia LDNF2. É importante observar que apesar de LDNF1 tenha apresentado o melhor valor de *Precissão*, ele possui uma baixa sensibilidade, sendo capaz de recuperar em média 9 das 15 mudanças de conceito existentes.

O comportamento geral dos resultados observado na Tabela 3 é similar ao da Tabela 1, o que mostra que a mudança do classificador de *Random Forest* para *Naive Bayes* não teve um efeito grande nos resultados. Em ambos os casos, as melhores estratégias

de altura dinâmica foram LDNF3, Eq. Teto e LDNF2 com uma pequena troca de ordem entre elas.

Base	Detector	Acurácia	Revocação	Precisão	F1-score	Detecções	Altura	Mudanças	Atraso
Toy	Fixo	81.52	**0.9919**	**0.9756**	**0.9837**	3	4.00	3/3	4
	LDNF1	**81.63**	0.8269	0.9709	0.8932	3	7.25	3/3	86.33
	LDNF2	81.22	0.9452	0.9744	0.9596	3	5.75	3/3	27.33
	LDNF3	81.53	0.9886	0.9755	0.9820	3	4.75	3/3	5.66
	Eq. Piso	81.52	**0.9919**	**0.9756**	**0.9837**	3	4.00	3/3	4
	Eq. Teto	81.53	0.9886	0.9755	0.9820	3	5.00	3/3	5.66
SEA	Fixo	**87.49**	0.8786	0.8581	**0.8682**	96	4.00	3/3	242.33
	LDNF1	80.52	0.5350	**0.9368**	0.6810	24	5.00	3/3	980.33
	LDNF2	**87.49**	0.8847	0.7387	0.8051	205	3.42	3/3	229.66
	LDNF3	85.25	**0.9919**	0.5394	0.6988	539	3.02	3/3	16
	Eq. Piso	85.01	0.9790	0.3875	0.5553	950	2.42	3/3	41
	Eq. Teto	86.85	0.9427	0.7629	0.8433	192	3.51	3/3	114
SINE1	Fixo	95.42	0.9959	0.6319	0.7732	15	4.00	1/1	2
	LDNF1	94.67	0.3627	0.7600	0.4910	3	7.25	1/1	318
	LDNF2	95.04	0.9679	**0.7830**	**0.8657**	7	4.75	1/1	16
	LDNF3	94.47	0.9939	0.5874	0.7384	18	4.31	1/1	3
	Eq. Piso	94.55	**1.0000**	0.4585	0.6288	30	3.38	1/1	0
	Eq. Teto	**95.44**	0.9698	0.6590	0.7847	13	4.28	1/1	15
SINE2	Fixo	**81.63**	0.6993	**0.9483**	**0.8050**	1	4.00	1/1	150
	LDNF1	**81.63**	0.6993	**0.9483**	**0.8050**	1	4.00	1/1	150
	LDNF2	81.54	0.6993	0.6693	0.6840	9	3.10	1/1	150
	LDNF3	78.24	0.9392	0.2282	0.3672	77	2.10	1/1	30
	Eq. Piso	77.07	**0.9595**	0.1522	0.2628	124	2.00	1/1	20
	Eq. Teto	79.69	0.8390	0.5597	0.6715	17	3.00	1/1	80
RBF_10d	Fixo	57.11	0.0000	0.0000	0.0000	0	4.00	0/1	n/a
	LDNF1	57.11	0.0000	0.0000	0.0000	0	5.00	0/1	n/a
	LDNF2	59.27	0.0000	0.0000	0.0000	7	3.00	0/1	n/a
	LDNF3	58.81	0.7907	0.8248	0.8074	37	1.63	1/1	418
	Eq. Piso	**59.49**	**0.7908**	**0.9359**	**0.8573**	12	3.00	1/1	418
	Eq. Teto	57.11	0.0000	0.0000	0.0000	0	4.00	0/1	n/a
checkerboard	Fixo	50.28	0.9904	0.5914	0.7406	170	4.00	3/3	4.66
	LDNF1	51.06	0.3264	**0.9346**	0.4838	6	7.28	1/3	9
	LDNF2	50.91	0.7068	0.8481	0.7710	33	5.41	3/3	146
	LDNF3	50.01	0.9555	0.7634	0.8487	76	4.61	3/3	22
	Eq. Piso	50.28	**0.9904**	0.5914	0.7406	170	4.00	3/3	4.66
	Eq. Teto	**51.28**	0.9434	0.8048	**0.8686**	59	5.00	3/3	28
Hyperplane	Fixo	91.49	0.3763	**0.9329**	0.5363	3	4.00	1/1	623
	LDNF1	89.79	0.0000	0.0000	0.0000	0	5.00	0/1	n/a
	LDNF2	92.50	0.9047	0.8541	0.8786	17	2.88	1/1	95
	LDNF3	92.32	**0.9959**	0.6874	0.8134	49	2.14	1/1	4
	Eq. Piso	91.19	0.9746	0.5312	0.6876	91	2.00	1/1	25
	Eq. Teto	**92.56**	0.9047	0.8691	**0.8865**	15	3.00	1/1	95
RBF_3d	Fixo	84.29	0.9703	0.3299	0.4923	198	4.00	1/1	29
	LDNF1	79.09	0.0000	0.0000	0.0000	1	9.00	0/1	n/a
	LDNF2	80.91	0.0000	0.0000	0.0000	5	6.66	0/1	n/a
	LDNF3	83.82	0.5455	**0.8577**	0.6669	10	6.00	1/1	454
	Eq. Piso	**84.72**	**0.9706**	0.4791	0.6415	111	4.36	1/1	29
	Eq. Teto	83.30	0.8907	0.8376	**0.8634**	19	5.55	1/1	109
MIXED	Fixo	88.34	0.9969	**0.9015**	**0.9468**	12	4.00	1/1	3
	LDNF1	**88.56**	0.0000	0.0000	0.0000	2	6.33	0/1	n/a
	LDNF2	88.34	0.9969	**0.9015**	**0.9468**	12	4.00	1/1	3
	LDNF3	87.64	0.9969	0.7840	0.8777	30	3.09	1/1	3
	Eq. Piso	86.63	**1.0000**	0.3942	0.5654	158	2.00	1/1	0
	Eq. Teto	88.18	0.9638	0.7043	0.8139	44	3.00	1/1	36
Médias	Fixo	**79.73**	0.7666	0.6855	0.6829	55.33	4.00	14/15	132.25
	LDNF1	78.23	0.3056	0.5056	0.3727	4.44	6.23	9/15	308.73
	LDNF2	79.69	0.6784	0.6410	0.6568	33.11	4.33	13/15	95.28
	LDNF3	79.12	0.9109	0.6942	**0.7556**	93.22	3.52	15/15	106.18
	Eq. Piso	78.94	**0.9619**	0.5451	0.6581	183.22	3.02	15/15	60.18
	Eq. Teto	79.55	0.8270	**0.6859**	0.7460	40.22	4.04	14/15	60.33

Tabela 3 – Resultados das estratégias de altura dinâmica usando o classificador *Naive Bayes* para as bases de dados com mudanças abruptas.

A Tabela 4 apresenta os resultados das estratégias de altura dinâmica para as

bases sintéticas com mudança de conceito gradual e bases reais usando o classificador *Naive Bayes* (NB). As métricas utilizadas para a análise das bases graduais e das reais foram as mesmas adotadas na Tabela 2, ou seja, a *Acurácia* e o número de *Detecções*.

No que se refere aos resultados de Acurácia obtidos na Tabela 4, a estratégia de altura dinâmica LDNF2 obteve a melhor média entre as bases com mudanças graduais, e com a segunda menor quantidade de *Detecções*, repetindo o comportamento visto na Tabela 2. Entre as bases reais, a estratégia Fixo e LDNF3 obtiveram os melhores resultados na base *Electricity* com 146 detecções em ambas as estratégias. Na base *Weather* as estratégias com os melhores resultados também foram LDNF3 e Fixo, com 37 e 29 detecções respectivamente. A relação de crescimento entre a sensibilidade do detector e o parâmetro ρ da estratégia LDNF também se mantém para o classificador *Naives Bayes*. O mesmo pode ser observado para as estratégias das Eqs. Teto e Piso, nessa ordem.

Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	Fixo	88,25	120		Fixo	**81,68**	2
	LDNF1	87,87	15		LDNF1	**81,68**	2
Hyperplane_G	LDNF2	88,12	47	SINE2_G	LDNF2	81,06	8
	LDNF3	**88,81**	102		LDNF3	79,23	65
	Eq. Piso	87,88	187		Eq. Piso	76,12	127
	Eq. Teto	88,41	91		Eq. Teto	79,00	20
	Fixo	89,93	91		Fixo	49,91	186
	LDNF1	84,18	34		LDNF1	50,46	5
SEA_G	LDNF2	89,93	164	Checker_G	LDNF2	50,44	46
	LDNF3	89,86	447		LDNF3	**50,83**	95
	Eq. Piso	89,57	813		Eq. Piso	49,91	186
	Eq. Teto	**89,96**	178		Eq. Teto	50,34	73
	Fixo	94,05	40		Fixo	80,76	87,80
	LDNF1	94,35	3		LDNF1	79,71	11,80
SINE1_G	LDNF2	94,78	6	Média	LDNF2	**80,87**	54,20
	LDNF3	93,40	34		LDNF3	80,43	148,60
	Eq. Piso	93,67	50		Eq. Piso	79,43	272,60
	Eq. Teto	**94,79**	19		Eq. Teto	80,50	76,20

Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	Fixo	**74,50**	146		Fixo	72,04	29
	LDNF1	69,70	16		LDNF1	71,52	3
Electricity	LDNF2	71,72	48	Weather	LDNF2	71,81	19
	LDNF3	**74,50**	146		LDNF3	**72,16**	37
	Eq. Piso	67,60	6		Eq. Piso	71,52	3
	Eq. Teto	67,35	2		Eq. Teto	71,25	1

Tabela 4 – Resultados das estratégias de altura dinâmica usando o classificador *Naive Bayes* em bases graduais e reais.

No geral, as estratégias de altura dinâmica trazem melhores resultados que a estratégia ingênua de altura fixa, salvo exceções. Na escolha das melhores estratégias, para as bases de dados sintéticas com mudanças de conceito abruptas, levou-se em consideração a precisão e sua capacidade de recuperar o maior número de mudanças de conceitos reais. No caso das bases graduais e reais foi utilizado o compromisso que existe entre a acurácia e o menor número de detecções.

As estratégias LDNF2, LDFN3 e Eq. Teto foram as que apresentaram os resultados qualitativos mais interessantes, sob o prisma de se obter detectores precisos e com uma

quantidade razoável de detecções capaz de apontar as mudanças de conceito existentes. Observou-se que o comportamento das estratégias de altura dinâmica se manteve com a mudança do classificador. Por fim, novas pesquisas com a finalidade de refinar as estratégias de altura dinâmica podem beneficiar o método QT, especialmente na diminuição de falsos positivos.

4.4 Experimentos 2 - Diferentes Detectores de Mudança de Conceito

Nesta seção os métodos propostos QT e QTS são avaliados junto a outros detectores conhecidos na literatura como: *Drift Detection Method* (DDM) [Gama et al., 2004], *Early Drift Detection Method* (EDDM) [Baena-Garcıa et al., 2006], *ADaptive WINndowing* (ADWIN) [Bifet and Gavalda, 2007], Page-Hinkley (PH) [Gama, 2010, Mahdi et al., 2020], *Drift Detection Method based on Hoeffding's bounds with Moving Average-test* (HDDMa) [Frias-Blanco et al., 2014] e *Drift Detection Method based on Hoeffding's bounds with Moving Weighted average-test* (HDDMw) [Frias-Blanco et al., 2014] sobre bases de dados sintéticas e reais na presença de mudança de conceito. Estes métodos estão disponíveis na *framework Scikit-multiflow*2.

Os parâmetros dos detectores de mudança foram mantidos constantes durante os experimentos com bases sintéticas. Os detectores foram configurados de acordo com os valores recomendados no framework Scikit-multiflow[2]. Os parâmetros do DDM $min\_num\_instances$, $warning\_level$ e $out\_control\_level$ foram definidos com os seguintes valores 30, 2.0 e 3.0, respectivamente. O EDDM possui os mesmos parâmetros do DDM, de modo que foram definidos em 30, 0.9 e 0.85, respectivamente. Para o ADWIN o parâmetro δ foi definido com o valor $0,002$. Para o PH, os parâmetros $min\_instances$, δ, $threshold$ e α foram definidos com os valores 30, 0.005, 50 e 0.9999, respectivamente. Os parâmetros do HDDMa $drift\_confidence$, $warning\_confidence$ e two_side_option foram definidos com os valores 0.001, 0.005 e $True$, respectivamente. O HDDMw que possui os mesmo parâmetros do HDDMa está com os mesmos valores definidos, o parâmetro extra $lambda\_option$ foi definido com o valor 0.050. Para o QTS, o parâmetro *altura* foi ajustado segundo a recomendação $(256 \leq (2^d)^h \leq 65536)$, para as bases de dados com dimensões $2, 3, 4, 8$ e 10 foram usadas as alturas $4, 3, 2, 2$ e 1 respectivamente. O parâmetro $janela\_maior$ recebeu o valor $tamanho\_dinâmico$, já os parâmetros $janela\_menor$ e ψ foram definidos com os valores 100 e 0.5 respectivamente. Para o QT foi adotado a estratégia do Limite de Dados no Nó Folha com o parâmetro $\rho = 2$. Os critérios utilizados para esta escolha foram o maior número de Mudanças detectadas com o menor número de Detecções feitas, condições que maximizam as métricas de qualidade do detector. O intervalo de atraso adotado nas bases de dados sintéticas com mudanças de conceito

abrupto permanece o mesmo do Experimento 1.

Nas bases sintéticas com mudanças de conceito abruptas são analisados os valores de Acurácia, *Revocação*, *Precisão* e *F1-score*. Também são analisadas a quantidade de Detecções, a quantidade de Mudanças de conceito que foram detectadas dentro do intervalo de atraso, e o Atraso médio apenas das detecções feitas dentro do intervalo de atraso.

As Tabelas 5 e 6 apresentam os resultados dos detectores de mudança de conceito sobre as bases de dados com mudança de conceito abrupto usando o classificador *Random Forest* (RF). No final da Tabela 6, têm-se os valores médios das métricas analisadas.

A partir da análise dos valores médios da *Acurácia* presentes na Tabela 6, foram observados valores muito próximos. O EDDM obteve o melhor valor e o HDDMa o valor mais baixo, mas com uma diferença de 0.76, o que faz com que uma tomada de decisão usando apenas esta métrica seja muito difícil.

O valor da *Revocação* se difere bastante entre os detectores. Os detectores que obtiveram os melhores resultados foram o QT e QTS, detectores que se mostraram mais responsivos às detecções de mudança de conceito. Normalmente, um detector com elevado valor de *Revocação* tende a apresentar uma maior quantidade de *Detecções*, como pode ser observado com QT e EDDM. No entanto, o QTS se mostra uma exceção, já que possui a menor quantidade de *Detecções*.

O valor da *Precisão* está ligado diretamente às detecções realizadas dentro do intervalo de atraso e o total de detecções realizadas. Os métodos de detecção tiveram valores de *Precisão* muito próximos, pois, os métodos que alarmaram muitas *Detecções* também conseguiram acertar muitas *Mudanças* frente aqueles que tiveram poucas *Detecções* e deixaram passar algumas *Mudanças*. O detector QTS apresentou os melhores valores de *Precisão*, com o menor número de *Detecções* e o segundo maior número de *Mudanças*. O HDDMw obteve o segundo melhor resultado, com valores de *Detecções* e *Mudanças* próximos ao QTS. Os detectores QTS e QT obtiveram os melhores resultados de *F1-score*, mas com características muito distintas entre eles.

Quanto ao número de *Detecções*, o QTS realizou o menor número de *Detecções* acompanhado pelos bons valores de *Precisão*, o que sugere que o método é acionado apenas quando uma mudança de conceito real acontece. Por outro lado, tem-se o QT com o maior número de *Detecções*, e que apesar de apresentar bons valores de *Revocação* e *Precisão*, tem o comportamento de um método com alta sensibilidade a detecções, perdendo qualidade na detecção devido ao alto número de falsos positivos.

Base	Detector	Acurácia	Revocação	Precisão	F1-score	Detecções	Mudanças	Atraso
Toy	QT	92,81	0.7902	0.9696	0.8708	3	3/3	104.66
	QTS	93,44	**0.9933**	**0.9757**	**0.9844**	3	3/3	3.33
	ADWIN	92,67	0.8236	0.9432	0.8794	6	3/3	88
	DDM	93,38	0.6114	0.9737	0.7512	2	2/3	41
	EDDM	**93,99**	0.6121	0.9737	0.7517	2	2/3	40.5
	PH	92,90	0.4432	0.9641	0.6073	2	2/3	167
	HDDMa	93,70	0.6582	0.9755	0.7860	2	2/3	6
	HDDMw	93,01	0.4799	0.9668	0.6414	2	2/3	139.5
SEA	QT	**81,27**	0.8685	0.6559	0.7473	296	3/3	186.33
	QTS	77,06	0.0000	0.0000	0.0000	0	0/3	n/a
	ADWIN	77,00	0.0000	0.0000	0.0000	2	0/3	n/a
	DDM	77,52	0.0000	0.0000	0.0000	8	0/3	n/a
	EDDM	81,18	**0.9036**	**0.7325**	**0.8091**	216	3/3	192
	PH	77,06	0.0000	0.0000	0.0000	3	0/3	n/a
	HDDMa	77,15	0.0000	0.0000	0.0000	9	0/3	n/a
	HDDMw	77,87	0.3846	0.9175	0.5420	23	2/3	845.50
SINE1	QT	92,97	0.9819	0.5587	0.7122	20	1/1	14
	QTS	94,04	**0.9919**	**0.9630**	**0.9772**	1	1/1	4
	ADWIN	**94,87**	0.8717	0.9195	0.8949	2	1/1	64
	DDM	94,46	0.9358	0.9608	0.9482	1	1/1	32
	EDDM	94,12	0.9198	0.8892	0.9042	3	1/1	42
	PH	94,73	0.8897	0.9589	0.9230	1	1/1	55
	HDDMa	94,04	**0.9919**	**0.9630**	**0.9772**	1	1/1	4
	HDDMw	93,91	0.9799	0.9625	0.9711	1	1/1	10
SINE2	QT	89,40	0.0000	0.0000	0.0000	5	0/1	n/a
	QTS	90,65	0.9739	0.9623	0.9681	1	1/1	13
	ADWIN	90,78	0.7434	0.9512	0.8346	1	1/1	128
	DDM	90,88	0.9198	0.9234	0.9216	2	1/1	40
	EDDM	90,76	0.7710	0.8009	0.7857	5	1/1	114
	PH	**90,89**	0.8837	0.9586	0.9196	1	1/1	58
	HDDMa	90,76	0.9198	0.9234	0.9216	2	1/1	40
	HDDMw	90,68	**0.9759**	**0.9624**	**0.9691**	1	1/1	12
RBF_10d	QT	**66,21**	**0.7908**	0.9669	**0.8700**	6	1/1	418
	QTS	64,02	0.0000	0.0000	0.0000	0	0/1	n/a
	ADWIN	65,52	0.2676	**0.9834**	0.4207	1	1/1	1464
	DDM	64,02	0.0000	0.0000	0.0000	0	0/1	n/a
	EDDM	64,44	0.0000	0.0000	0.0000	1	0/1	n/a
	PH	64,02	0.0000	0.0000	0.0000	0	0/1	n/a
	HDDMa	64,02	0.0000	0.0000	0.0000	0	0/1	n/a
	HDDMw	65,56	0.0000	0.0000	0.0000	10	0/1	n/a
Checkerboard	QT	66,22	0.8517	0.8502	0.8510	39	3/3	151.66
	QTS	**68,82**	**0.9285**	**0.9878**	**0.9572**	3	3/3	35.66
	ADWIN	66,89	0.1621	0.9343	0.2762	3	1/3	256
	DDM	67,01	0.1621	0.9771	0.2780	1	1/3	256
	EDDM	66,06	0.2096	0.7432	0.3269	19	1/3	184
	PH	67,41	0.1914	0.9805	0.3203	1	1/3	212
	HDDMa	67,46	0.2855	0.9741	0.4416	2	1/3	71
	HDDMw	68,78	0.5861	0.9747	0.7320	4	2/3	60
Hyperplane	QT	84,10	0.8184	0.7620	0.7892	28	1/1	181
	QTS	84,02	**0.9669**	**0.9907**	**0.9787**	1	1/1	33
	ADWIN	84,15	0.6796	0.9869	0.8049	1	1/1	320
	DDM	**84,59**	0.4804	0.9141	0.6298	5	1/1	519
	EDDM	83,27	0.7758	0.7321	0.7533	31	1/1	223
	PH	84,33	0.6086	0.9574	0.7441	3	1/1	391
	HDDMa	83,37	0.8018	0.9888	0.8855	1	1/1	198
	HDDMw	84,17	0.8158	0.9678	0.8853	3	1/1	184
RBF_3d	QT	**87,24**	0.7833	0.9251	0.8483	7	1/1	216
	QTS	85,66	0.9609	**0.9815**	**0.9711**	2	1/1	39
	ADWIN	85,99	0.0000	0.0000	0.0000	2	0/1	n/a
	DDM	86,08	0.0000	0.0000	0.0000	1	0/1	n/a
	EDDM	86,95	**0.9939**	0.8520	0.9175	19	1/1	6
	PH	85,82	0.0000	0.0000	0.0000	1	0/1	n/a
	HDDMa	85,17	0.0000	0.0000	0.0000	0	0/1	n/a
	HDDMw	85,52	0.0000	0.0000	0.0000	1	0/1	n/a

Tabela 5 – Resultados dos detectores usando o classificador *Random Forest* em bases sintéticas com mudanças abruptas (PARTE 1 de 2).

Base	Detector	Acurácia	Revocação	Precisão	F1-score	Detecções	Mudanças	Atraso
MIXED	QT	90.83	**0.9969**	0.8728	0.9308	16	1/1	3
	QTS	91.88	0.9939	**0.9910**	**0.9925**	1	1/1	4
	ADWIN	92.51	0.9039	0.9804	0.9406	2	1/1	96
	DDM	91.94	0.9629	0.9907	0.9766	1	1/1	37
	EDDM	**93.17**	0.9389	0.9719	0.9551	3	1/1	61
	PH	92.50	0.9439	0.9905	0.9666	1	1/1	56
	HDDMa	91.40	0.9809	0.9908	0.9859	1	1/1	19
	HDDMw	91.59	0.9899	0.9909	0.9904	1	1/1	10
Médias	QT	83.45	**0.7646**	0.7290	0.7355	46.67	14/15	155.47
	QTS	83.29	0.7566	**0.7613**	**0.7588**	1.33	11/15	18.86
	ADWIN	83.38	0.4947	0.7443	0.5613	2.22	9/15	345.14
	DDM	83.32	0.4525	0.6378	0.5006	2.33	7/15	154.17
	EDDM	**83.77**	0.6805	0.7439	0.6893	33.22	11/15	117.43
	PH	83.30	0.4401	0.6456	0.4979	1.44	7/15	156.50
	HDDMa	83.01	0.5153	0.6462	0.5553	2.00	7/15	56.33
	HDDMw	83.45	0.5791	0.7492	0.6368	5.11	10/15	55.20

Tabela 6 – Resultados dos detectores usando o classificador *Random Forest* em bases sintéticas com mudanças abruptas (PARTE 2 de 2).

A coluna *Mudanças* apresenta as detecções feitas dentro do espaço de atraso. Os detectores que conseguiram os melhores resultados foram o QT, QTS e EDDM com 14, 11 e 11 detecções respectivamente das 15 possíveis. Vale ressaltar que o QTS possui a menor quantidade de *Detecções* dentre estes detectores, o que sugere uma melhor qualidade das detecções realizadas.

A coluna *Atraso* apresenta apenas os valores do atraso médio das detecções feitas dentro do intervalo de atraso. O QTS é o detector com o menor atraso nas detecções, seguido pelo HDDMw e HDDMa, sugerindo que estes detectores conseguem detectar uma mudança de conceito rapidamente quando ela efetivamente acontece.

Investigando as bases de dados que trouxeram um desafio maior aos detectores, temos a base Checkerboard que apresenta uma baixa acurácia entre os detectores, o que sugere que o classificador não esteja bem ajustado. Como os detectores QT e QTS têm como princípio a análise espacial, estes foram capazes de identificar as mudanças entre os diferentes conceitos, já que a diferenciação espacial entre os conceitos são possíveis. A Figura 40 apresenta a ocupação espacial dos diferentes conceitos em relação ao tempo. É possível observar que os conceitos ocupam diferentes espaços que é perceptível a partir do incremento da quantidade de dados nas árvores de cada classe.

O método QT é susceptível a falsas detecções em duas situações, sobreposição entre as classes e presença de ruído. Estas são situações em que dados de classes opostas passam a ocupar espaços na árvore de classe oposta a ponto de eventualmente vir a provocar uma falsa detecção. Este é o caso da base de dados Checkerboard, em que dados de classes opostas estão lado a lado com uma pequena sobreposição.

A vantagem do QT e QTS frente a métodos baseados em erro, como o DDM, é que estes tem dificuldade em detectar mudanças de conceito quando a acurácia do classificador não varia ou varia positivamente. A Figura 41 apresenta a acurácia média do classificador

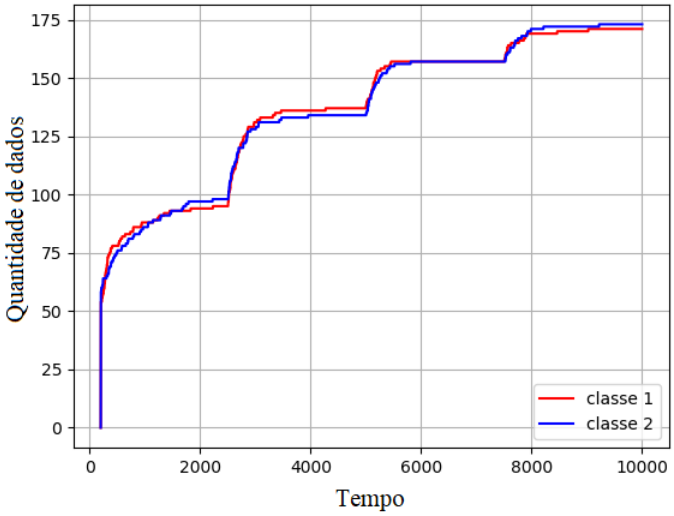
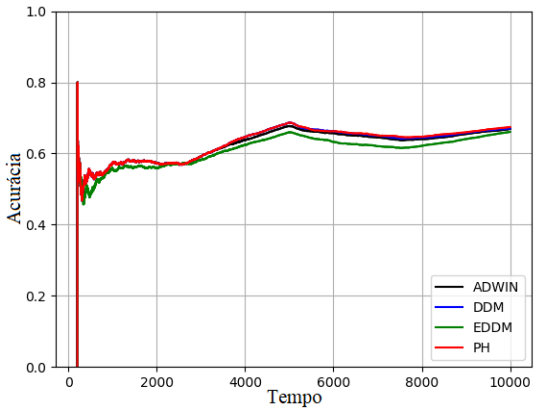
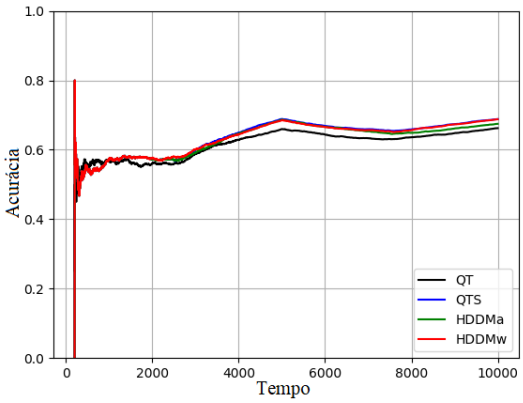


Figura 40 – Ocupação espacial da base Checkerboard nas duas árvores no método QTS.

ao longo do fluxo de dados da base Checkerboard, onde é possível observar que em dois momentos a acurácia varia positivamente. A Figura 42 apresenta o histograma com a posição e frequência das detecções feitas nas barras verticais. A linha vertical vermelha indica onde as mudanças de conceito acontecem, e a linha vertical tracejada indica o intervalo de atraso utilizado. É possível observar que no local onde aconteceu o aumento na acurácia não houveram detecções. Esta mesma característica se repete na base Toy, na mudança de conceito entre a função Espirais e a função Gaussianas.



(a) Acurácia geral dos detectores ADWIN, DDM, EDDM e PH.



(b) Acurácia geral dos detectores QT, QTS, HDDMa e HDDMw.

Figura 41 – Acurácia geral para a base Checkerboard.

As bases SEA, RBF_10d e RBF_3d também trouxeram maior desafio aos detectores. Para o QTS estas bases têm por característica pouca mudança na ocupação espacial entre os diferentes conceitos. A Figura 43 mostra essa característica, o que dificulta a detecção da mudança por parte do QTS. Para as bases SEA e RBF_10d é possível notar nas Figuras 43a e 43c pequenos incrementos no aumento da quantidade de dados nos

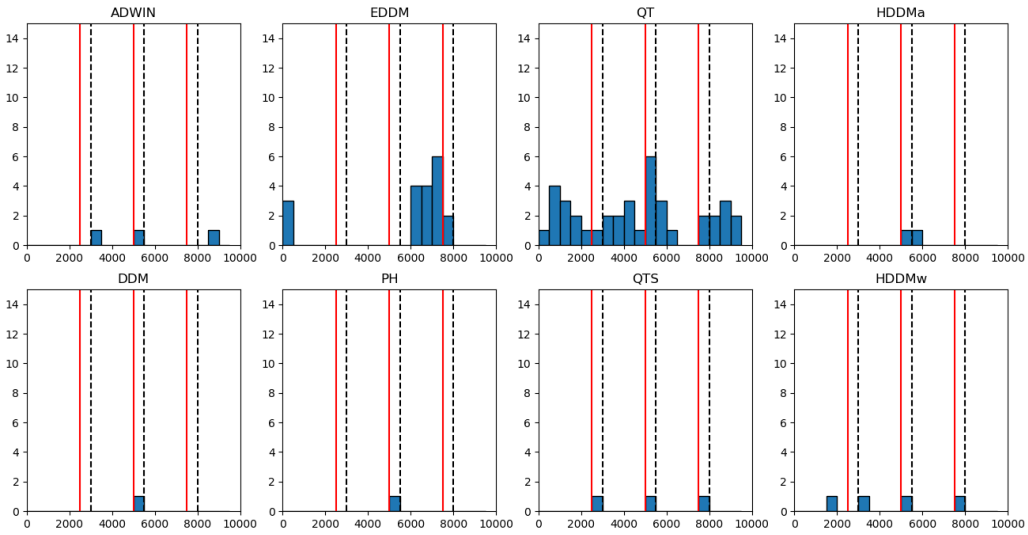


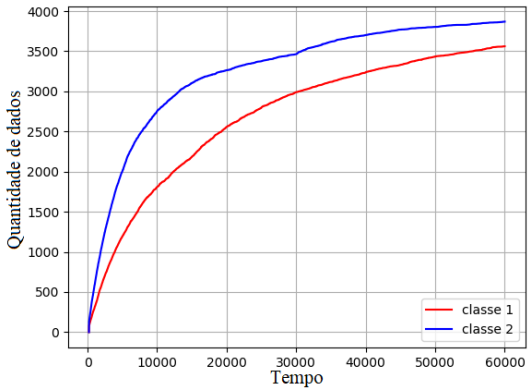
Figura 42 – Histograma da base Checkerboard.

tempos 30000 e 10000 respectivamente, onde um ajuste nos parâmetros no sentido de aumentar a sensibilidade do detector poderia evidenciar essas mudanças. Quanto a base RBF_3d, o QTS foi capaz de identificar a mudança de conceito, evidente em apenas uma das classes.

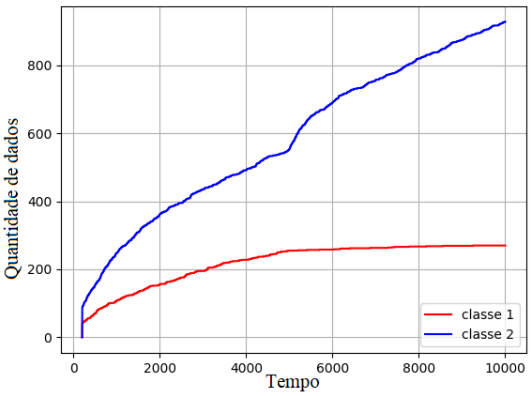
A presença de ruído dificulta a análise espacial, e no caso do QT, ela aumenta a probabilidade de falsos positivos. Para o QTS, o ruído tende a ocupar novos espaços, causando o desaparecimento do platô formado na saturação do espaço e diminuindo a velocidade do incremento de dados no espaço no momento em que acontece a mudança de conceito. As Figuras 44a e 44b exemplificam esta situação por meio das bases SINE1 que não possui ruido e a base SINE2 com ruído. Cabe ressaltar que ainda assim, o método QTS foi capaz de encontrar a única mudança de conceito que ocorre na base SINE2, apesar dos seus atributos ruidosos.

A Tabela 7 apresenta os resultados dos detectores de mudança de conceito para as bases sintéticas com mudança de conceito gradual e bases reais usando o classificador *Random Forest* (RF). É importante esclarecer que nas bases de dados com mudanças de conceito graduais e as bases reais não são especificadas ou informadas as posições onde ocorrem as mudanças de conceito. Desta forma, não é possível calcular as métricas de qualidade *Revocação*, *Precisão* e *F1-score*.

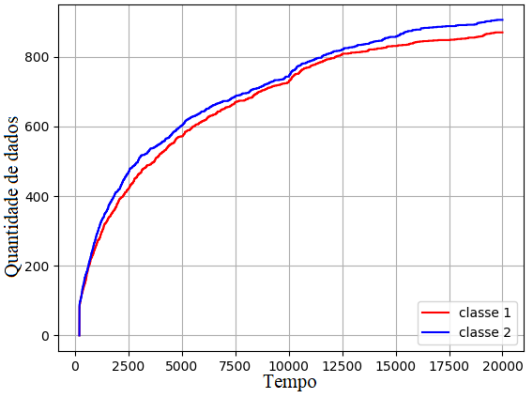
Neste contexto limitado, as métricas utilizadas para a análise das bases graduais e reais são a *Acurácia* e o número de *Detecções*. Quanto aos resultados obtidos nas bases sintéticas com mudança de conceito gradual, o detector EDDM foi o que conseguiu os melhores valores de *Acurácia*, seguido pelo QT, justamente os dois métodos que tiveram as maiores quantidades de *Detecções*. Como mencionado anteriormente, o QT apresenta um número maior de detecções devido à sobreposição espacial dos dados que geralmente



(a) Ocupação espacial da base SEA.



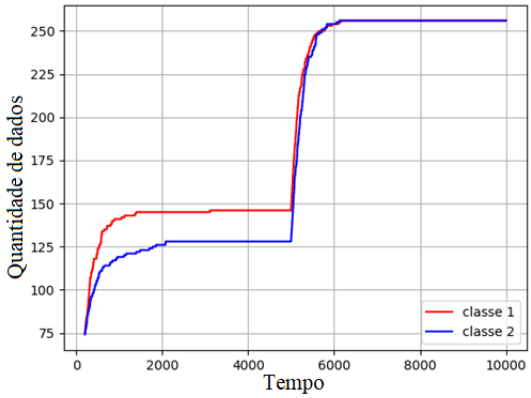
(b) Ocupação espacial da base RBF_3d.



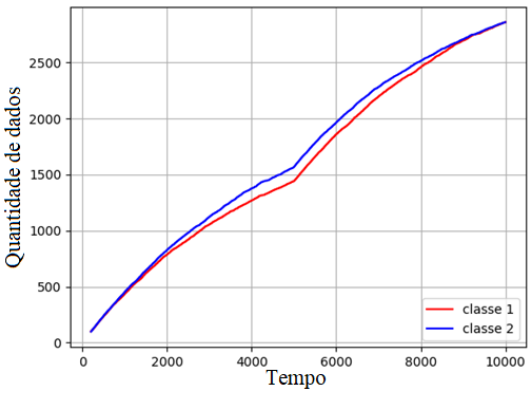
(c) Ocupação espacial da base RBF_10d.

Figura 43 – Ocupação espacial das bases SEA, RBF_10d e RBF_3d nas duas classes das árvores no método QTS.

ocorre durante mudanças graduais de funções geradoras.



(a) Ocupação espacial da base SINE1.



(b) Ocupação espacial da base SINE2.

Figura 44 – Ocupação espacial das bases SINE1 e SINE2 nas duas árvores no método QTS.

Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	QT	93.22	78		QT	88.75	8
	QTS	92.90	1		QTS	89.03	1
	ADWIN	93.26	5		ADWIN	**89.51**	2
Hyperplane_G	DDM	92.91	6	SINE2_G	DDM	89.24	2
	EDDM	**93.38**	10		EDDM	88.81	11
	PH	92.95	2		PH	89.18	1
	HDDMa	93.01	3		HDDMa	89.13	2
	HDDMw	92.74	3		HDDMw	89.02	1
	QT	82.44	269		QT	64.59	37
	QTS	79.27	0		QTS	64.31	1
	ADWIN	79.12	2		ADWIN	64.98	3
SEA_G	DDM	79.59	14	Checker_G	DDM	65.35	1
	EDDM	**82.47**	177		EDDM	65.15	34
	PH	79.06	4		PH	65.70	1
	HDDMa	79.44	2		HDDMa	65.28	2
	HDDMw	79.45	16		HDDMw	**65.80**	6
	QT	92.20	26		QT	84.24	83.60
	QTS	92.31	1		QTS	83.56	0.80
	ADWIN	**93.27**	2		ADWIN	84.03	2.80
SINE1_G	DDM	92.39	3	Média	DDM	83.90	5.20
	EDDM	92.32	6		EDDM	**84.43**	47.60
	PH	92.50	1		PH	83.88	1.80
	HDDMa	92.48	2		HDDMa	83.87	2.20
	HDDMw	92.64	2		HDDMw	83.93	5.60

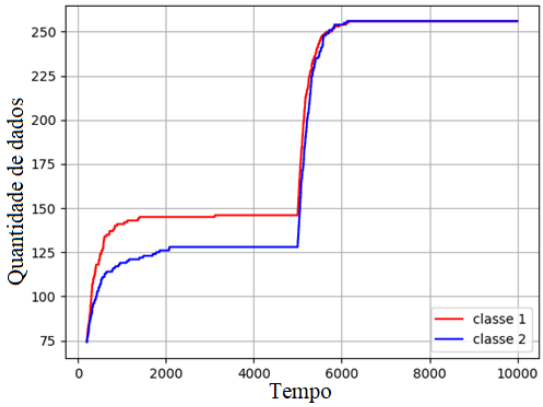
Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	QT	**89.59**	11		QT	73.20	2
	QTS	89.01	11		QTS	73.14	2
	ADWIN	89.00	6		ADWIN	73.05	1
Electricity	DDM	89.22	9	Weather	DDM	**73.59**	3
	EDDM	88.40	30		EDDM	73.43	4
	PH	88.51	3		PH	72.86	3
	HDDMa	88.58	18		HDDMa	72.67	6
	HDDMw	88.88	10		HDDMw	72.75	12

Tabela 7 – Resultados dos detectores usando o classificador Random Forest em bases graduais e reais.

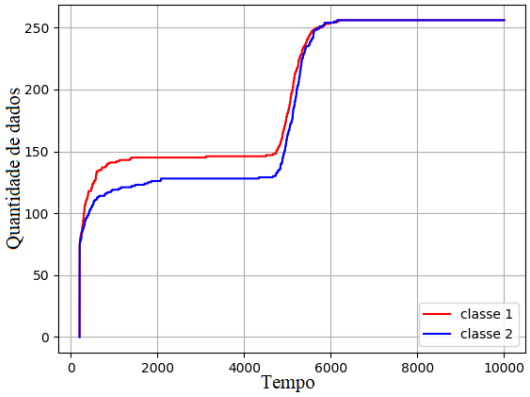
O QTS obteve o valor de acurácia mais baixo entre os detectores avaliados, apesar desses valores serem muito próximos entre si, diferenciando apenas 0.87. A Figura 45 apresenta a ocupação espacial das bases SINE1 e SINE1_G, onde é possível observar a mudança gradual que acontece na base SINE1_G. A Figura 46 apresenta os valores das derivadas correspondentes da curva de ocupação espacial das bases SINE1 e SINE1_G.

É possível observar que o comportamento do método QTS tanto para mudanças abruptas quanto para graduais é o mesmo, e isso é um ponto fraco deste método. Ele identifica a mudança de conceito logo no início da mudança gradual, treinando um novo classificador com dados de dois conceitos, o que pode levar a uma acurácia mais baixa ao longo do tempo. Propostas de melhorias na forma da detecção de mudanças de conceito para o método QTS são discutidas no capítulo de Conclusão junto às propostas de continuidade. Para a base SEA_G em que o QTS não encontrou mudanças de conceito, o motivo é similar ao discutido anteriormente para a base de dados SEA.

Quanto aos experimentos realizados nas bases de dados reais, foi adotado o mesmo método utilizado em [Yu et al., 2019], no qual os parâmetros dos detectores são ajustados para que a quantidade de detecções sejam similares. Para a base *Electricity* os parâmetros

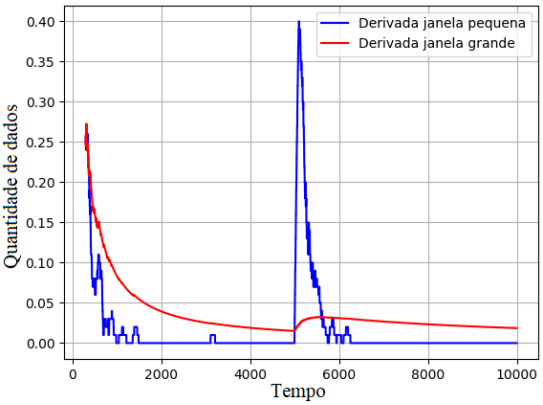


(a) Ocupação espacial da base SINE1.

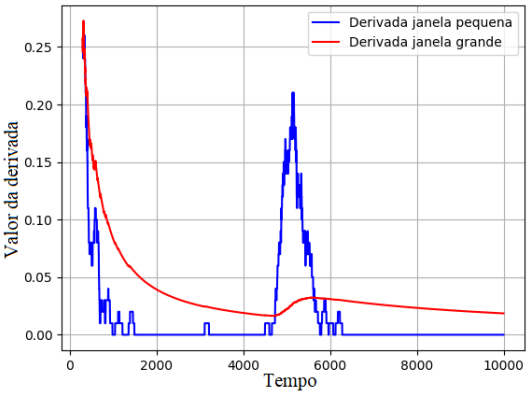


(b) Ocupação espacial da base SINE1_G.

Figura 45 – Ocupação espacial das bases SINE1 e SINE1_G nas duas classes das árvores no método QTS.



(a) Valores de derivada da base SINE1.



(b) Valores de derivada da base SINE1_G.

Figura 46 – Valores de derivada das bases SINE1 e SINE1_G no método QTS.

dos detectores EDDM *min_num_instances*, *warning_level* e *out_control_level* foram reajustados com os seguintes valores 30, 0.95 e 0.85, respectivamente. O QT ajustou o valor do parâmetro $\rho = 1$. Os demais detectores mantiveram os valores dos parâmetros. Conforme observado pela Tabela 7, os valores de *Acurácia* foram muito próximos, com o QT obtendo maior valor com 11 Detecções, e o QTS com o mesmo número de *Detecções* teve uma acurácia pouco menor.

Para a base *Wheather*, além dos detectores EDDM e QT já reajustados, o HDDMW teve seus parâmetros $drift\_confidence$, $warning\_confidence$, two_side_option e $lambda\_option$ alterados para 0.0001, 0.005, $True$ e 0.050 respectivamente. Os demais mantiveram os parâmetros. Os valores de Acurácia foram muito próximos entre os detectores avaliados, com o EDDM obtendo o maior valor com 3 *Detecções*. O QT e QTS tiveram o terceiro e quarto melhores valores de acurácia com duas detecções cada.

A Tabela 8 apresenta a *Acurácia* média e a quantidade média de *Detecções* para os métodos avaliados sobre todas as bases sintéticas abruptas, graduais e as bases reais

Detectores	QT	QTS	ADWIN	DDM	EDDM	PH	HDDMa	HDDMw
Acurácia	84.41	84.02	84.15	84.11	**84.57**	83.97	83.80	84.11
Detecções	53.19	1.81	2.56	3.69	35.69	1.75	3.31	6.00

Tabela 8 – Média da acurácia em todas as bases de dados usando o classificador *Random Forest* (RF)

usando o classificador *Random Forest* (RF). Os métodos EDDM e QT obtiveram as melhores médias de acurácia, já o QTS teve a sexta melhor média. Uma característica a ser destacada é de que os métodos que tiveram as maiores médias de acurácia são justamente aqueles que tiveram os maiores números de detecções.

Com o foco na métrica *Acurácia*, falsos positivos, dependendo da situação e quantidade, podem acarretar na melhora da métrica. Por exemplo, na situação da falta de detecção dentro do intervalo de atraso, é preferível uma detecção tardia para que tenha um classificador treinado exclusivamente com dados do conceito corrente, o que acarretará em melhores médias de acurácia a longo prazo. A respeito da quantidade de falsos positivos, caso as perdas de acurácia no treinamento de um novo classificador são menores frente as perdas no treinamento incremental de um classificador já existente, quando o mesmo pode estar com sinais de sobreajuste.

As Tabelas 9 e 10 apresentam os resultados dos detectores de mudança de conceito sobre todas as bases de dados com mudança de conceito abrupto usando o classificador *Naive Bayes* (NB). Ao fim, a Tabela 10 apresenta as médias dos resultados dos detectores. Os parâmetros dos detectores de mudança usados nas bases sintéticas foram os mesmos utilizados no experimento anterior com o classificador RF e mantidos contrantes durante todos os experimentos.

A partir da análise dos valores médios da *Acurácia* presentes na Tabela 10, o QT e o EDDM obtiveram os melhores valores. O valor de *Revocação* variou bastante entre os detectores. O EDDM conseguiu o melhor resultado, seguido pelo QTS e QT. Da mesma forma que aconteceu com o classificador RF, os detectores que tiveram os valores mais altos de *Revocação* também fizeram mais Detecções. O QTS se mostrou uma exceção, já que possui a menor quantidade de Detecções.

No que se refere ao *Precisão*, o detector EDDM apresentou o melhor resultado, seguido de perto pelo QTS. O QT com a maior quantidade de *Detecções*, ficou a frente apenas do PH. Os detectores EDDM e QTS obtiveram os melhores resultados para *F1-score*, mas com características distintas entre eles.

Base	Detector	Acurácia	Revocação	Precisão	F1-score	Detecções	Mudanças	Atraso
Toy	QT	81.22	0.9452	**0.9744**	**0.9596**	3	3/3	27.33
	QTS	81.80	0.9472	0.9663	0.9566	4	3/3	16.33
	ADWIN	80.45	0.5006	0.9100	0.6459	6	2/3	124
	DDM	81.08	0.5942	0.9230	0.7230	6	2/3	53.5
	EDDM	80.66	0.5194	0.8997	0.6586	7	2/3	109
	PH	80.89	0.5854	0.9594	0.7271	3	2/3	60.5
	HDDMa	81.60	0.6341	0.9624	0.7645	3	2/3	24
	HDDMw	**82.72**	**0.9725**	0.9363	0.9541	8	3/3	112.66
SEA	QT	**87.49**	0.8847	0.7387	0.8051	205	3/3	229.66
	QTS	74.88	0.0000	0.0000	0.0000	0	0/3	n/a
	ADWIN	81.16	0.7264	0.9252	0.8139	39	3/3	546.66
	DDM	81.99	**0.9129**	**0.9529**	**0.9324**	30	3/3	174
	EDDM	84.73	0.6599	0.8452	0.7411	80	3/3	679
	PH	79.69	0.2808	0.9077	0.4290	19	1/3	314
	HDDMa	79.66	0.2905	0.8978	0.4390	22	1/3	256
	HDDMw	80.96	0.4494	0.9228	0.6045	25	3/3	1100.33
SINE1	QT	95.04	0.9679	0.7830	0.8657	7	1/1	16
	QTS	94.87	0.9919	**0.9630**	0.9772	1	1/1	4
	ADWIN	**96.34**	0.8714	0.8505	0.8608	4	1/1	64
	DDM	95.16	0.9659	0.9620	0.9639	1	1/1	17
	EDDM	95.74	0.9698	0.9269	0.9479	2	1/1	15
	PH	95.44	0.8917	0.9590	0.9241	1	1/1	54
	HDDMa	94.75	**0.9939**	**0.9630**	**0.9782**	1	1/1	3
	HDDMw	95.04	0.9799	0.9625	0.9711	1	1/1	10
SINE2	QT	81.54	0.6993	0.6693	0.6840	9	1/1	150
	QTS	81.39	0.9739	0.9623	0.9681	1	1/1	13
	ADWIN	80.45	0.7434	0.9069	0.8171	2	1/1	128
	DDM	**81.91**	0.9078	0.8879	0.8977	3	1/1	46
	EDDM	81.28	0.9778	0.6967	0.8137	11	1/1	11
	PH	81.55	0.8957	0.9591	0.9264	1	1/1	52
	HDDMa	81.51	0.9158	0.9600	0.9374	1	1/1	42
	HDDMw	81.39	**0.9839**	**0.9627**	**0.9732**	1	1/1	8
RBF_10d	QT	59.27	0.0000	0.0000	0.0000	7	0/1	n/a
	QTS	57.11	0.0000	0.0000	0.0000	0	0/1	n/a
	ADWIN	58.10	0.0000	0.0000	0.0000	2	0/1	n/a
	DDM	58.32	0.0000	0.0000	0.0000	1	0/1	n/a
	EDDM	59.22	**0.9453**	0.8220	**0.8793**	45	1/1	109
	PH	57.11	0.0000	0.0000	0.0000	0	0/1	n/a
	HDDMa	58.25	0.0000	0.0000	0.0000	2	0/1	n/a
	HDDMw	**60.86**	0.8319	**0.9109**	0.8696	18	1/1	336
Checkerboard	QT	50.91	0.7068	0.8481	0.7710	33	3/3	146
	QTS	50.80	**0.9311**	**0.9879**	**0.9587**	3	3/3	34.33
	ADWIN	50.54	0.0000	0.0000	0.0000	0	0/3	n/a
	DDM	50.54	0.0000	0.0000	0.0000	0	0/3	n/a
	EDDM	50.02	0.0000	0.0000	0.0000	12	0/3	n/a
	PH	50.54	0.0000	0.0000	0.0000	0	0/3	n/a
	HDDMa	50.68	0.0000	0.0000	0.0000	1	0/3	n/a
	HDDMw	**50.99**	0.0000	0.0000	0.0000	7	0/3	n/a
Hyperplane	QT	**92.50**	0.9047	0.8541	0.8786	17	1/1	95
	QTS	87.94	**0.9669**	**0.9907**	**0.9787**	1	1/1	33
	ADWIN	90.40	0.8078	0.9675	0.8805	3	1/1	192
	DDM	90.53	0.9047	0.9523	0.9279	5	1/1	95
	EDDM	90.74	0.8793	0.9237	0.9010	8	1/1	120
	PH	90.51	0.7747	0.9772	0.8643	2	1/1	225
	HDDMa	89.30	0.8648	0.9795	0.9186	2	1/1	135
	HDDMw	90.02	0.8168	0.9679	0.8859	3	1/1	183
RBF_3d	QT	80.91	0.0000	0.0000	0.0000	5	0/1	n/a
	QTS	83.38	0.9620	0.9815	**0.9716**	2	1/1	38
	ADWIN	84.66	0.0000	0.0000	0.0000	2	0/1	n/a
	DDM	83.97	0.7207	0.9638	0.8247	3	1/1	279
	EDDM	**85.99**	**0.9949**	0.9014	0.9459	12	1/1	5
	PH	84.25	0.5395	0.9676	0.6927	2	1/1	460
	HDDMa	83.53	0.7377	**0.9879**	0.8446	1	1/1	262
	HDDMw	80.88	0.0000	0.0000	0.0000	2	0/1	n/a

Tabela 9 – Resultados dos detectores usando o classificador *Naive Byes* em bases sintéticas (PARTE 1 de 2).

Base	Detector	Acurácia	Revocação	Precisão	F1-score	Detecções	Mudanças	Atraso
MIXED	QT	88.34	**0.9969**	0.9015	0.7892	12	1/1	3
	QTS	87.86	0.9959	0.9822	0.9890	2	1/1	4
	ADWIN	87.94	0.9038	0.9709	0.9361	3	1/1	96
	DDM	87.39	0.9649	0.9907	0.9776	1	1/1	35
	EDDM	**88.74**	0.9248	0.9715	0.9476	3	1/1	75
	PH	87.62	0.9499	0.9906	0.9698	1	1/1	50
	HDDMa	87.44	0.9759	0.9908	0.9833	1	1/1	24
	HDDMw	87.17	0.9909	**0.9909**	**0.9909**	1	1/1	9
Médias	QT	**79.69**	0.6784	0.6410	0.6392	33.11	13/15	95.28
	QTS	77.78	0.7521	0.7593	0.7555	1.56	11/15	21.06
	ADWIN	78.89	0.5059	0.6146	0.5505	6.78	9/15	191.78
	DDM	78.99	0.6635	0.7370	0.6941	5.56	10/15	99.93
	EDDM	79.68	**0.7635**	**0.7763**	**0.7595**	20.00	11/15	140.38
	PH	78.62	0.5464	0.7467	0.6148	3.22	8/15	173.64
	HDDMa	78.52	0.6014	0.7490	0.6517	3.78	8/15	106.57
	HDDMw	78.89	0.6695	0.7393	0.6944	7.33	11/15	251.28

Tabela 10 – Resultados dos detectores usando o classificador *Naive Bayes* em bases sintéticas (PARTE 2 de 2).

Repetindo o mesmo comportamento visto com o classificador RF, o QTS realizou o menor número de *Detecções* acompanhado pelos bons valores de *Precisão*, o que sugere que o detector é acionado apenas quando uma mudança de conceito real acontece. No outro extremo, temos o QT com o maior número de detecções, demonstrando alta sensibilidade a detecções, perdendo qualidade na detecção devido ao alto número de falsos positivos. Quanto às detecções feitas dentro do espaço de atraso (coluna *Mudanças*), os detectores que conseguiram os melhores resultados foram o QT, QTS e EDDM com 13, 11 e 11 detecções respectivamente das 15 possíveis.

Em relação à coluna *Atraso*, o mesmo comportamento observado com o classificador RF, também foi observado com o NB. O QTS foi o método que teve o menor atraso médio nas detecções. O EDDM, apesar de ter apresentado os melhores valores de *Revocação*, *Precisão* e *F1-score*, foi o que apresentou um dos maiores *Atrasos* de detecção.

Observando o desempenho dos detectores QT e QTS nos diferentes classificadores RF e NB, foi percebido que o comportamento se repetiu. A boa capacidade de detecção se manteve para as bases Checkerboad e Toy. O mesmo também pode ser dito quanto às dificuldades já apontadas para a detecção de mudanças para as bases SEA, RBF_10d e RBF_3d. A mudança do classificador mostrou ter pouco efeito no comportamento dos detectores avaliados, especialmente QT e QTS.

A Tabela 11 apresenta os resultados dos detectores de mudança de conceito para as bases sintéticas com mudança de conceito gradual e bases reais usando o classificador *Naive Bayes* (NB). Como pode ser observado, o detector QT foi o que conseguiu os melhores valores de *Acurácia*, seguido pelo EDDM, justamente os dois detectores que tiveram as maiores quantidades de *Detecções*. O QTS apresentou o valor mais baixo de *Acurácia*, apesar dos resultados muito próximos entre todos os detectores avaliados, repetindo o cenário do classificador RF.

Quanto aos experimentos realizados nas bases de dados reais, os parâmetros dos detectores foram ajustados para que a quantidade de detecções sejam similares. Para a base *Electricity* os parâmetros do DDM $min\_num\_instances$, $warning\_level$ e $out\_control\_level$ foram definidos para 30, 2.0 e 2.5 respectivamente. No EDDM, os parâmetros $min\_num\_instances$, $warning\_level$ e $out\_control\_level$ foram reajustados com os seguintes valores 30, 0.95 e 0.85. O PH tive seus parâmetros $min\_instances$, δ, $threshold$ e α ajustados para 30, $0,005$, 10 e $0,9999$, respectivamente. O parâmetro δ do detector ADWIN foi ajustado com o valor 0.2. Os parâmetros do HDDMa $drift\_confidence$, $warning\_confidence$ e two_side_option foram definidos com os valores 0.0001, 0.005 e $True$, respectivamente. O QT teve o parâmetro ρ reajustado para o valor 3, e o QTS teve o parâmetro ψ reajustado para 0.9. O HDDMw manteve os valores dos parâmetros já anteriormente definidos. Conforme observado na Tabela 11, os valores de Acurácia foram novamente muito próximos entre os detectores, com o DDM apresentando o melhor resultado. O QT e o QTS, juntamente com o ADWIN, apresentaram os valores mais baixos de *Acurácia*, mas com os menores números de *Detecções* entre os métodos avaliados.

Para a base *Wheather*, os parâmetros dos detectores EDDM, HDDMw e QT tiveram que ser reajustados, os demais usaram os parâmetros aplicados para as bases sintéticas. O EDDM teve os parâmetros $min\_num\_instances$, $warning\_level$ e $out\_control\_level$ reajustados com os seguintes valores 30, 0.95 e 0.85. O HDDMW teve seus parâmetros $drift\_confidence$, $warning\_confidence$, two_side_option e $lambda\_option$ alterados para 0.0001, 0.005, $True$ e 0.050 respectivamente. O QT teve o parâmetro ρ reajustado para o valor 1.

A Tabela 12 apresenta os valores médios de *Acurácia* e número de *Detecções* obtidos pelos detectores sobre todas as bases de dados abruptas, graduais e reais, com o classificador *Naive Bayes*. Os resultados de *Acurácia* são um pouco menores quando comparados aos obtidos com o classificador RF (vide Tabela 8). No entanto, em termos do comportamento dos detectores, os resultados são similares, de forma que os detectores mais sensíveis (com maior quantidade de detecções) tendem a ter um valor maior de acurácia. O detector QT obteve a melhor média de acurácia, já o QTS teve as médias mais baixas.

Os resultados experimentais mostraram que o QT e o QTS foram competitivos frente aos outros detectores. O QT se caracterizou por bons valores de acurácia e alto número de detecções sendo capaz de identificar muitas mudanças de conceito. O QTS, no que lhe concerne, mostrou ser um detector com valores elevados e equilibrados de *Revocação* e *Precisão*, um atraso na detecção de mudança muito baixo. Estas características se mantiveram nos dois classificadores utilizados.

Uma característica que chama atenção no método QT é o número elevado de detecções, o que normalmente implica em uma alta quantidade de falso positivos, tendo efeito negativo na sua precisão (*Precisão*). Nota-se, no entanto, que este método obteve as

Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	QT	88.12	47		QT	81.06	8
	QTS	84.92	1		QTS	81.21	1
	ADWIN	**88.58**	16		ADWIN	80.54	2
	DDM	83.26	6		DDM	**81.39**	4
Hyperplane_G	EDDM	84.97	67	SINE2_G	EDDM	80.74	11
	PH	83.21	5		PH	81.38	1
	HDDMa	86.42	10		HDDMa	81.21	1
	HDDMw	86.13	9		HDDMw	80.70	2
	QT	**89.93**	164		QT	50.44	46
	QTS	76.90	0		QTS	**50.97**	1
	ADWIN	82.76	37		ADWIN	50.62	0
	DDM	82.72	22		DDM	50.62	0
SEA_G	EDDM	89.89	140	Checker_G	EDDM	50.46	44
	PH	81.90	20		PH	49.82	1
	HDDMa	82.97	22		HDDMa	50.62	0
	HDDMw	83.94	24		HDDMw	50.18	9
	QT	94.78	6		QT	**80.87**	54.20
	QTS	94.88	1		QTS	77.78	0.80
	ADWIN	**95.22**	3		ADWIN	79.54	11.60
	DDM	94.10	3		DDM	78.42	7.00
SINE1_G	EDDM	93.95	7	Média	EDDM	80.00	53.80
	PH	94.38	1		PH	78.14	5.60
	HDDMa	94.26	2		HDDMa	79.10	7.00
	HDDMw	94.24	2		HDDMw	79.04	9.20

Base	Detector	Acurácia	Detecções	Base	Detector	Acurácia	Detecções
	QT	74.50	146		QT	71.52	3
	QTS	73.17	151		QTS	69.73	4
	ADWIN	73.27	181		ADWIN	**72.36**	4
	DDM	**76.24**	171		DDM	71.68	6
Electricity	EDDM	74.87	206	Weather	EDDM	71.25	39
	PH	75.38	201		PH	71.19	4
	HDDMa	75.75	179		HDDMa	71.11	15
	HDDMw	75.45	179		HDDMw	71.93	21

Tabela 11 – Resultados dos detectores usando o classificador Naive Bayes em bases graduais e reais.

Detectores	QT	QTS	ADWIN	DDM	EDDM	PH	HDDMa	HDDMw
Acurácia	**79.22**	76.99	78.34	78.18	78.95	77.80	78.07	78.29
Detecções	44.88	10.81	19.00	16.38	43.38	16.38	16.44	19.50

Tabela 12 – Média da acurácia em todas as bases de dados usando o classificador Naive Bayes

melhores acurácias para as bases com mudança gradual, o que indica que para este tipo de mudança (gradual), retreinar o classificador várias vezes ao longo do fluxo de dados, pode ser uma boa estratégia.

O QTS tem um comportamento diferente do QT, com baixas médias de *Acurácia* e valores mais elevados para as métricas de qualidade (*Revocação*, *Precisão*e *F1-score*). Ele normalmente consegue identificar a mudança de conceito mais rapidamente entre todos os detectores avaliados, com uma quantidade menor de *Detecções* e, com um número bem pequeno de falsos positivos.

No que se refere às bases com mudança abrupta, o QTS apresentou, juntamente como o EDDM, os melhores resultados em *Revocação*, *Precisão* e *F1-score*, mas com um atraso médio de detecção muito menor que o EDDM. Em relação às bases de dados

graduais, o QTS teve desempenho inferior aos outros detectores em *Acurácia*. No entanto, sua quantidade de detecções ao longo do fluxo é bem menor, o que faz com que o número de retreinamentos do modelo também seja pequeno. Na maior parte do tempo, tem-se um modelo treinado com dados oriundos de conceitos diferentes, o que provavelmente pode ter causado uma perda de desempenho no modelo.

4.5 Considerações Finais

Este capítulo apresentou os resultados experimentais dos detectores QT e QTS propostos na Tese. Na primeira etapa dos experimentos, foram avaliadas as estratégias de altura dinâmica para ajuste automático do parâmetro altura do detector QT. Avaliaram-se as estratégias LDNF com diferentes valores do parâmetro ρ (1, 2 e 3) e as estratégias não paramétricas, denominadas Eq. Piso e Eq. Teto. Os resultados apontaram que, de modo geral, as estratégias para ajuste automático da altura trazem melhores resultados que a estratégia ingênua de altura fixa. As estratégias LDNF2, LDFN3 e Eq. Teto foram as que apresentaram os resultados qualitativos mais interessantes, sob o prisma de se obter detectores precisos e com uma quantidade razoável de detecções capaz de apontar as mudanças de conceito existentes.

Na segunda parte dos experimentos, os métodos propostos QT (utilizando o LDNF2) e QTS tiveram seus desempenhos avaliados frente a detectores conhecidos na literatura. No que se refere às características observadas, QT apresentou valores elevados de acurácia e de detecções dentro do intervalo de atraso, mas com uma quantidade razoável de falsos positivos. O QTS se destacou por obter valores elevados de *Revocação*, *Precisão* e *F1-score*, e com um número menor de detecções e atraso médio em relação aos demais detectores avaliados.

De modo geral, o comportamento dos detectores se manteve parecido ao se usar dois classificadores diferentes: *Random Forest* e *Naive Bayes*. O método QT se adaptou bem aos diferentes tipos de mudança de conceito, obtendo melhores valores de acurácia para as bases com mudança gradual e também para as bases reais. Por ser mais sensível, QT tende a retreinar o modelo mais vezes ao longo fluxo de dados, o que pode ser benéfico em cenários com mudanças mais lentas nas distribuições ou com uma maior quantidade de ruído nos dados. O método QTS se adaptou muito bem às mudanças de conceito abruptas e, por ser menos sensível, teve uma maior dificuldade nas mudanças de conceito graduais. Dessa forma, QTS seria indicado para cenários onde o custo de se obter um falso alarme é muito alto, por exemplo, na manutenção preditiva de processos industriais, em que um falso alarme causaria a paralisação de uma planta sem a necessidade, acarretando um prejuízo na produção. Outro aspecto importante observado sobre os detectores de mudança propostos na tese foi que eles foram capazes de detectar mudanças de conceito em cenários

onde outros detectores tiveram dificuldades, devido à sua capacidade de analisar o espaço ocupado pelos dados durante a passagem do fluxo.

Outro ponto que chamou a atenção nos resultados foi uma aparente correlação positiva entre a acurácia e a quantidade de detecções. Esta característica já tinha sido apontada por outros trabalhos na literatura [Yu et al., 2019, Barros and Santos, 2018]. Os resultados das Tabelas 8 e 12 mostraram que os métodos QT e EDDM tiveram as melhores médias de acurácia, e são justamente os métodos que possuem também as maiores médias de detecções. Esta situação em parte se deve ao fato de que a manutenção da acurácia está ligada diretamente ao quanto o classificador está ajustado ao conceito atual. O classificador treinado com o conceito mais atual produz os melhores valores de acurácia. É esperado que os métodos que não detectaram mudanças de conceito dentro do intervalo de atraso, a façam de forma tardia. Em geral, uma detecção tardia produz melhores resultados, no que se refere a acurácia, que uma não detecção.

Capítulo 5

Conclusão e Propostas de Continuidade

Esta tese apresentou dois novos detectores de mudanças de conceito QT e QTS, que utilizam da estrutura de divisão recursiva do espaço por meio da quadtree. O método proposto QT analisa geometricamente a ocupação espacial dos dados e a detecção de mudança de conceito acontece ao atribuir um dado a um espaço previamente ocupado por dados de classe oposta. Já o QTS detecta a mudança de conceito quando identifica um aumento significativo no incremento de dados de uma das classes.

Para o desenvolvimento dos métodos propostos, foi necessário implementar um sistema de mapeamento binário que possibilitou estender a capacidade da quadtree para suportar dados de mais alta dimensão. Também foi implementado um limite ao parâmetro altura da quadtree, condição chave para a criação do QT e QTS, pois, permitiu a existência das ações que definem o comportamento dos detectores. Devido ao limite do parâmetro altura, para que a estrutura criada ainda seja considerada uma quadtree, dados de mesma classe que estejam na altura limite são sumarizados.

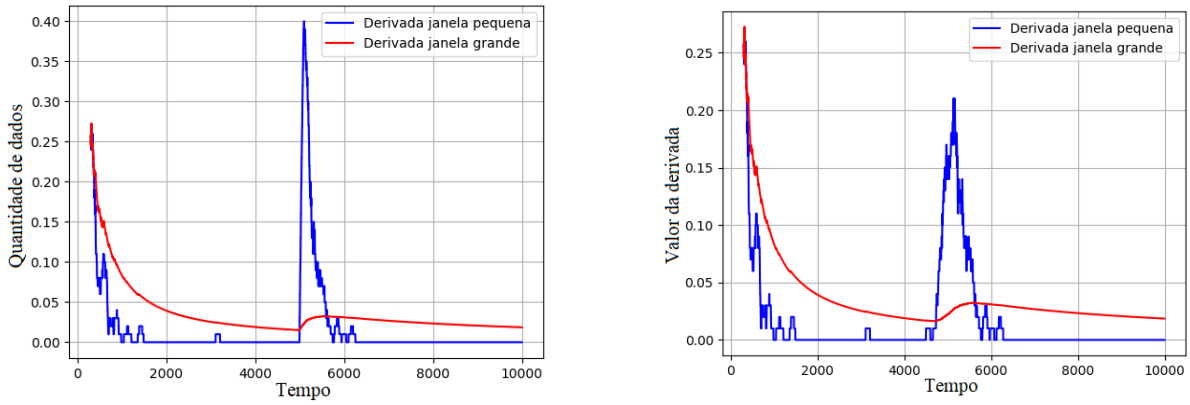
O parâmetro altura controla a sensibilidade da detecção de mudança de conceito do método QT. O valor do parâmetro altura indicado apenas no início do processo nem sempre atende de forma satisfatória todos os conceitos do fluxo de dados. Para resolver este problema foram desenvolvidas duas estratégias de altura dinâmica, estratégia do Limite de Dados no Nó Folha (LDNF) e a estratégia do cálculo da Densidade. Estas estratégias fornecem ferramentas necessárias para que o método mantenha a sensibilidade de detecção desejada para todos os conceitos. Isto evita que o método fique muito sensível em certos conceitos e pouco sensível em outros.

Os resultados experimentais foram divididos em suas etapas, a primeira apenas com o QT e as estratégias de altura dinâmica. As estratégias de altura dinâmica se mostraram eficientes, melhoraram a capacidade de adaptação do detector às variações impostas pelo fluxo de dados, e no controle e redução do número de falsos positivos. Na segunda etapa dos experimentos, os detectores QT e QTS se mostraram competentes quanto a sua tarefa, competitivos frente a outros detectores, e capazes em detectar mudanças de conceito em

cenários onde outros detectores tiveram dificuldades.

Durante os experimentos foram discutidos os pontos fortes, os fracos e as limitações dos métodos QT e QTS. Os detectores tiveram resultados e comportamentos complementares, a fraqueza de um acabou sendo o ponto forte do outro. Mantiveram o comportamento nos dois diferentes classificadores, e conseguiram evidenciar situações nas quais a sua utilização é preferencial frente a outros detectores: quando o classificador não está bem ajustado e quando a mudança de conceito é seguida do aumento da acurácia.

Como trabalhos futuros, 1) está a proposta de melhoria no método QTS, onde foi identificada uma limitação quanto a detecção em bases com mudanças graduais. A limitação consiste no treinamento de um novo classificador com dados dos dois conceitos presentes na mudança gradual, o que resulta na diminuição de acurácia. A Figura 47 apresenta os valores das derivadas correspondentes às curvas de ocupação espacial das bases SINE1 e SINE1_G. É possível observar que existem diferenças entre as curvas das derivadas (47a e 47b) da janela pequena tanto na amplitude quanto extensão. 2) A proposta de solução para esse problema consiste na utilização de um segundo classificador criado no momento em que a curva da derivada da janela menor cruzar a derivada da janela maior de forma descendente. Este novo classificador seria treinado com os dados mais atuais e seu desempenho seria comparado com o classificador já existente. Após um determinado tempo, o classificador com pior desempenho seria descartado. Para a adoção desse segundo classificador, é necessário reestruturar a forma de esquecimento das janelas deslizantes, mantendo os dados nas janelas deslizantes durante a mudança de conceito, e estipular um tamanho fixo para a janela grande W_g.



(a) Valores da derivada na base SINE1. (b) Valores da derivada na base SINE1_G.

Figura 47 – Valores das derivadas nas bases SINE1 e SINE1_G no método QTS.

2) A diminuição do número de falsos positivos no método QT. Melhorar a forma como o método interpreta a sobreposição natural que possa existir entre as classes, as identificando e desconsiderando detecções de mudança de conceito que acontecem nesses espaços.

3) Propostas de melhorias das estratégias de altura dinâmica. Os resultados experimentais (Tabelas 1, 2, 3 e 4) mostram uma grande variação da quantidade de detecções entre as estratégias. Foi observada a existência um ponto mais favorável ainda inexplorado entre o número de detecções e a altura recomendada, capaz de minimizar ainda mais a quantidade de falsos positivos sem diminuir a quantidade de mudanças detectadas.

4) Pesquisar por detectores de mudança de conceito não paramétrico de modo a trazer parâmetros de comparação para o QT não paramétrico usando a estratégia da Densidade. Explorar as equações da estratégia da Densidade em busca de melhorar suas métricas de qualidade.

5) Implementar as melhorias necessárias para que o QT e QTS possam suportar problemas de classificação com mais de duas classes.

6) Existem limitações inerentes à estrutura de dados quadtree, como a dificuldade de trabalhar com dados não reais e dados de mais alta dimensão. Para lidar com dados não reais, sejam bases categóricas ou numéricas (inteiros), é necessário um pré-processamento através de uma transformação não linear, seja ela polinomial, ordinal ou numérica, por exemplo, para uma escala real. Técnicas como *Target Encoding* também se aplicam. Já com dados de mais alta dimensão também é necessário um pré-processamento para a diminuição da dimensionalidade através de técnicas como a análise de componente principal incremental (*Incremental Principal Component Analysis*).

Referências

C. Alippi, G. Boracchi, and M. Roveri. Hierarchical change-detection tests. *IEEE transactions on neural networks and learning systems*, 28(2):246–258, 2016.

W. J. Alvarenga, F. V. Campos, V. M. Hanriot, E. B. Gonçalves, A. C. Costa, L. R. Araujo, E. Magalhães, and A. P. Braga. Online learning of neural networks using random projections and sliding window: A case study of a real industrial process. *Engineering Applications of Artificial Intelligence*, 100:104181, 2021.

M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno. Early drift detection method. In *Fourth international workshop on knowledge discovery from data streams*, volume 6, pages 77–86, 2006.

R. S. Barros, D. R. Cabral, P. M. Gonçalves Jr, and S. G. Santos. Rddm: Reactive drift detection method. *Expert Systems with Applications*, 90:344–355, 2017.

R. S. M. Barros and S. G. T. C. Santos. A large-scale comparison of concept drift detectors. *Information Sciences*, 451:348–370, 2018.

M. Basseville, I. V. Nikiforov, et al. *Detection of abrupt changes: theory and application*, volume 104. Prentice hall Englewood Cliffs, 1993.

A. Bifet and R. Gavalda. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM international conference on data mining*, pages 443–448. SIAM, 2007.

A. Bifet, G. Holmes, B. Pfahringer, and E. Frank. Fast perceptron decision tree learning from evolving data streams. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 299–310. Springer, 2010.

G. Boracchi, D. Carrera, C. Cervellera, and D. Maccio. Quanttree: Histograms for change detection in multivariate data streams. In *International Conference on Machine Learning*, pages 639–648. PMLR, 2018.

A. Bouchachia. Fuzzy classification in dynamic environments. *Soft Computing*, 15(5): 1009–1022, 2011.

A. Bouchachia and C. Vanaret. Gt2fc: An online growing interval type-2 self-learning fuzzy classifier. *IEEE Transactions on Fuzzy Systems*, 22(4):999–1018, 2013.

A. Cano and B. Krawczyk. Kappa updated ensemble for drifting data stream mining. *Machine Learning*, 109(1):175–218, 2020.

J. M. Carmona-Cejudo, M. Baena-García, J. del Campo-Avila, R. Morales-Bueno, and A. Bifet. Gnusmail: Open framework for on-line email classification. *Frontiers in Artificial Intelligence and Applications*, 2011.

R. A. Coelho and C. L. de Castro. Abordagem espacial via quadtree para detecção de mudança de conceito em fluxos de dados contínuos. In *Congresso Brasileiro de Automática-CBA*, pages 1–6, 2020.

R. S. M. de Barros and S. G. T. de Carvalho Santos. An overview and comprehensive comparison of ensembles for concept drift. *Information Fusion*, 52:213–244, 2019.

R. S. M. de Barros, J. I. G. Hidalgo, and D. R. de Lima Cabral. Wilcoxon rank sum test drift detector. *Neurocomputing*, 275:1954–1963, 2018.

R. Elwell and R. Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, 2011.

I. Frias-Blanco, J. del Campo-Ávila, G. Ramos-Jimenez, R. Morales-Bueno, A. Ortiz-Diaz, and Y. Caballero-Mota. Online and non-parametric drift detection methods based on hoeffding's bounds. *IEEE Transactions on Knowledge and Data Engineering*, 27(3): 810–823, 2014.

J. Gama. *Knowledge discovery from data streams*. Chapman and Hall/CRC, 2010.

J. Gama and G. Castillo. Learning with local drift detection. In *International Conference on Advanced Data Mining and Applications*, pages 42–55. Springer, 2006.

J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In *Brazilian symposium on artificial intelligence*, pages 286–295. Springer, 2004.

J. Gama, R. Fernandes, and R. Rocha. Decision trees for mining data streams. *Intelligent Data Analysis*, 10(1):23–45, 2006.

J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):44, 2014.

R. C. Gonzalez. *Digital image processing*. Pearson education india, 2009.

F. Hinder, A. Artelt, and B. Hammer. Towards non-parametric drift detection via dynamic adapting window independence drift detection (dawidd). In *International Conference on Machine Learning*, pages 4249–4259. PMLR, 2020.

W. Hoeffding. Probability inequalities for sums of bounded random variables. In *The collected works of Wassily Hoeffding*, pages 409–426. Springer, 1994.

E. Ikonomovska, J. Gama, and S. Džeroski. Learning model trees from evolving data streams. *Data mining and knowledge discovery*, 23(1):128–168, 2011.

A. S. Iwashita and J. P. Papa. An overview on concept drift learning. *Ieee Access*, 7: 1532–1547, 2018.

I. Khamassi, M. Sayed-Mouchaweh, M. Hammami, and K. Ghédira. Discussion and review on evolving data streams and concept drift adapting. *Evolving systems*, 9(1):1–23, 2018.

R. Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intelligent data analysis*, 8(3):281–300, 2004.

D. Leite, I. Škrjanc, and F. Gomide. An overview on evolving systems and learning from stream data. *Evolving systems*, 11(2):181–198, 2020.

V. Lemaire, C. Salperwyck, and A. Bondu. A survey on supervised classification on data streams. In *European Business Intelligence Summer School*, pages 88–125. Springer, 2014.

Á. C. Lemos Neto, R. A. Coelho, and C. L. d. Castro. An incremental learning approach using long short-term memory neural networks. *Journal of Control, Automation and Electrical Systems*, pages 1–9, 2022.

A. Liu, J. Lu, and G. Zhang. Concept drift detection via equal intensity k-means space partitioning. *IEEE transactions on cybernetics*, 51(6):3198–3211, 2020.

J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363, 2018.

O. A. Mahdi, E. Pardede, N. Ali, and J. Cao. Diversity measure as a new drift detection method in data streaming. *Knowledge-Based Systems*, 191:105227, 2020.

J. Maia, C. A. S. Junior, F. G. Guimarães, C. L. de Castro, A. P. Lemos, J. C. F. Galindo, and M. W. Cohen. Evolving clustering algorithm based on mixture of typicalities for stream data mining. *Future Generation Computer Systems*, 106:672–684, 2020.

D. P. Mehta and S. Sahni. *Handbook of data structures and applications*. Chapman and Hall/CRC, 2004.

T. M. L. Menegaldi, R. A. Coelho, and C. L. Castro. Aprendizado incremental de redes rbf via agrupamento evolutivo de fluxos de dados. In *Anais do 15 Congresso Brasileiro de Inteligência Computacional*, pages 1–8, Joinville, SC, 2021. SBIC.

H.-L. Nguyen, Y.-K. Woon, and W.-K. Ng. A survey on data stream clustering and classification. *Knowledge and information systems*, 45(3):535–569, 2015.

K. Nishida and K. Yamauchi. Detecting concept drift using statistical testing. In *International conference on discovery science*, pages 264–269. Springer, 2007.

E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

D. W. Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

B. W. Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

J. Stewart and J. H. Romo. *cálculo*. Cengage Learning, 2017.

W. N. Street and Y. Kim. A streaming ensemble algorithm (sea) for large-scale classification. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 377–382. ACM, 2001.

S. J. Swamidass, C.-A. Azencott, K. Daily, and P. Baldi. A croc stronger than roc: measuring, visualizing and optimizing early retrieval. *Bioinformatics*, 26(10):1348–1356, 2010.

J. N. van Rijn, G. Holmes, B. Pfahringer, and J. Vanschoren. The online performance estimation framework: heterogeneous ensemble learning for data streams. *Machine Learning*, 107(1):149–176, 2018.

H. Wang and Z. Abraham. Concept drift detection for streaming data. In *2015 international joint conference on neural networks (IJCNN)*, pages 1–9. IEEE, 2015.

S. Xu and J. Wang. Dynamic extreme learning machine for data stream classification. *Neurocomputing*, 238:433–449, 2017.

M. M. W. Yan. Accurate detecting concept drift in evolving data streams. *ICT Express*, 6 (4):332–338, 2020.

S. Yu and Z. Abraham. Concept drift detection with hierarchical hypothesis testing. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 768–776. SIAM, 2017.

S. Yu, Z. Abraham, H. Wang, M. Shah, Y. Wei, and J. C. Príncipe. Concept drift detection and adaptation with hierarchical hypothesis testing. *Journal of the Franklin Institute*, 356(5):3187–3215, 2019.

P. Zhao, S. C. Hoi, R. Jin, and T. YANG. Online auc maximization. *International Conference on Machine Learning ICML*, 2011.

P. Zhao, L.-W. Cai, and Z.-H. Zhou. Handling concept drift via model reuse. *Machine Learning*, 109(3):533–568, 2020.

I. Zliobaite. Learning under concept drift: an overview. *arXiv preprint arXiv:1010.4784*, 2010.

I. Zliobaite. How good is the electricity benchmark for evaluating concept drift adaptation. *arXiv preprint arXiv:1301.3524*, 2013.