UNIVERSIDADE FEDERAL DE MATO GROSSO

CAMPUS UNIVERSITÁRIO DO ARAGUAIA

INSTITUTO DE CIÊNCIAS EXATAS E DA TERRA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MÉTODOS ENSEMBLE ONLINE PARA CLASSIFICAÇÃO BINÁRIA

Arthur Gonçalves Soares

Barra do Garças-MT

2024

UNIVERSIDADE FEDERAL DE MATO GROSSO
CAMPUS UNIVERSITÁRIO DO ARAGUAIA
INSTITUTO DE CIÊNCIAS EXATAS E DA TERRA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MÉTODOS ENSEMBLE ONLINE PARA CLASSIFICAÇÃO BINÁRIA

Arthur Gonçalves Soares

Monografia apresentada ao Instituto de Ciências Exatas e da Terra do Campus Universitário do Araguaia, da Universidade Federal de Mato Grosso, como requisito parcial para a conclusão do Curso de Graduação de Ciência da Computação.

Orientador: Prof. Dr. Thiago Pereira da Silva

Barra do Garças-MT
2024

UNIVERSIDADE FEDERAL DE MATO GROSSO
CAMPUS UNIVERSITÁRIO DO ARAGUAIA
INSTITUTO DE CIÊNCIAS EXATAS E DA TERRA
BACHARELADO DE CIÊNCIA DA COMPUTAÇÃO

MÉTODOS ENSEMBLE ONLINE PARA CLASSIFICAÇÃO BINÁRIA

Arthur Gonçalves Soares

Esta monografia foi avaliada e aprovada pela banca examinadora instituída pelo Colegiado do Curso de Bacharelado em Ciência da Computação do Campus Universitário do Araguaia, da Universidade Federal de Mato Grosso.

Prof. Dr. Thiago Pereira da Silva
ICET/UFMT
(Orientador)

Prof. Dr. Rafael Teixeira Sousa
ICET/UFMT
(Membro)

Prof. Dr. Robson da Silva Lopes
ICET/UFMT
(Membro)

"Tudo tem o seu tempo determinado,
e há tempo para todo o propósito
debaixo do céu."

Eclesiastes 3:1

AGRADECIMENTOS

A Deus, por me guiar durante minha jornada, por conceder-me a força necessária para superar cada obstáculo. Em momentos de incerteza, Ele mostrou-me o caminho, sendo meu refúgio e minha fortaleza

Ao meu orientador, expresso profunda gratidão por sua orientação dedicada e maestria no ensino. Suas instruções e conselhos foram inestimáveis e contribuíram significativamente ao longo da monografia. Agradeço sinceramente por sua generosidade, comprometimento e apoio ao longo desta jornada.

Gostaria de expressar minha profunda gratidão à minha amada avó, Maria Valdereza, e ao meu querido avô, Leonel Barreto. Pelo amor, carinho e por acreditar tanto no meu potencial, foram um farol de inspiração ao longo desta jornada acadêmica. Suas palavras de encorajamento e carinho foram um conforto constante, motivando-me a persistir mesmo diante dos desafios.

A minha mãe, Ruth Gonçalves, suas orientações foram fundamentais para meu crescimento pessoal e acadêmico. Sou eternamente grato por tudo que você fez e por sua presença em minha vida.

A Rita de Kassia Ferrari, por todo seu amor e carinho durante essa jornada. Nos momentos de alegria, você esteve ao meu lado. Nos momentos de incerteza, sua presença e palavras de encorajamento foram um raio de luz. Por ser um exemplo e uma fonte de inspiração em minha vida.

RESUMO

Os métodos de *ensemble* se destacam como uma abordagem altamente eficaz para a classificação de fluxos de dados, frequentemente superando modelos individuais em termos de precisão. Ao combinar diversos modelos individuais, o *ensemble* é capaz de aprimorar as previsões, capturando diferentes perspectivas e nuances dos dados. No entanto, em cenários dinâmicos e desafiadores da vida real, onde os fluxos de dados são volumosos e em constante mudança, e muitas vezes não há acesso aos dados de treinamento com antecedência, os métodos de *ensemble* baseados em aprendizado tradicional, especialmente em lotes (*batch*), não são adequados. Nesses contextos, o aprendizado de máquina online emerge como a solução ideal. Neste paradigma, os modelos são treinados continuamente a cada nova observação, permitindo-lhes adaptar-se dinamicamente às mudanças nos dados, onde as condições e padrões podem evoluir rapidamente. Esta monografia explora a lacuna entre teoria e prática no âmbito do aprendizado de máquina online, com foco na abordagem de método ensemble para classificação binária. A monografia analisa os desafios inerentes à seleção e implementação de modelos individuais, bem como explora diversas técnicas para combinar as previsões desses modelos. Além disso, este trabalho apresenta os resultados de experimentos realizados em oito *datasets*, que simulam cenários reais. Os resultados evidenciam a superioridade do método ensemble proposto em comparação com abordagens individuais.

PALAVRAS-CHAVE: COMITÊ, APRENDIZADO DE MÁQUINA, PROCESSAMENTO ONLINE, CLASSIFICAÇÃO BINÁRIA, FLUXO DE DADOS.

ABSTRACT

Ensemble methods stand out as highly effective approaches for classifying data streams, often surpassing individual models in terms of prediction errors. By combining individual models, the ensemble can enhance predictions, capturing different perspectives and nuances of the data. However, in dynamic and challenging scenarios, where data streams are voluminous and constantly changing, and often there is no access to training data in advance, ensemble methods based on traditional, especially batch, learning are not suitable. In these contexts, online machine learning emerges as the ideal solution. In this paradigm, models are continuously trained with each new observation, allowing them to dynamically adapt to changes in the data, where conditions and patterns can evolve rapidly. This monograph explores the gap between theory and practice in the realm of online machine learning, focusing on the ensemble method approach for binary classification. The monograph examines the inherent challenges in the selection and implementation of individual models, as well as explores some techniques for combining the predictions of these base models. Additionally, this work presents the results of experiments conducted on eight datasets, which simulate real-world scenarios. The results highlight the proposed ensemble method's superiority over individual approaches considering the most common prediction metrics.

KEY-WORDS: ENSEMBLE, MACHINE LEARNING, ONLINE PROCESSING, BINARY CLASSIFICATION, DATA STREAM.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A evolução da computação tem sido notável, caracterizada por computadores mais velozes e capacidade de processamento e armazenamento cada vez maior. Com essa evolução, houve o crescimento exponencial de dados gerados. Esta era marcante na computação é frequentemente denominada "Era dos dados" (1). Os gigantescos conjuntos de dados, conhecidos como *Big Data* (6), podem apresentar estruturação variada ou serem não estruturados. Além de possuírem volumes massivos de dados, os *Big Data* (6) destacam-se pela diversidade de informações que contêm e pela velocidade dinâmica com que são atualizados (7).

Na era dos dados, as empresas e organizações são confrontadas com o desafio de utilizar métodos para compreender e analisar todos os dados que possuem, tendo o objetivo de encontrar padrões e *insights* para tomar decisões mais precisas. Neste contexto, conforme aponta (1), "dados são o novo petróleo" devido a sua importância estratégica. Os fluxos de dados, também conhecido como *data stream*[1], são uma forma de disponibilizar dados instantaneamente conforme e quando um evento ocorre, podendo ser em tempo real ou quase em tempo real. Os fluxos de dados em tempo real são uma manifestação direta da abundância de dados que caracteriza a era dos dados.

Em contraste com a abordagem tradicional de coletar e armazenar dados ao longo do tempo para posterior análise, os fluxos de dados permitem que os dados estejam prontamente disponíveis logo que os eventos ocorrem (6). Esses fluxos de dados representam desafios substanciais para as áreas de aprendizado de máquina e mineração de dados (2). Para organizações, os fluxos de dados são extremamente importantes, pois permitem o processamento contínuo e a extração de *insights* importantes de negócio a partir dos dados que surgem. Desta forma, as empresas podem tomar decisões mais ágeis, além de permitir a detecção precoce de padrões e eventos significativos que possam impactar as operações ou estratégias (6).

A agilidade na obtenção de *insights* em tempo real proporcionada pelo processamento dos data streams também levanta questões cruciais sobre a classificação e interpretação desses dados em constante fluxo. A classificação de dados é fundamental no processo de análise dos *Big Data*, independente de sua natureza, seja textual, multimídia, proveniente de redes sociais, biológicos ou médicos. A classificação visa aprender a relação entre um conjunto de características e uma variável de interesse específica, permitindo assim associar essas características à variável e prever o valor dessa variável para novos exemplos/observações (8). Principalmente na área financeira tem sido amplamente utilizada a técnica de aprendizado de máquina para classificação, especialmente a binária, nos cenários que exigem previsão de uma resposta qualitativa. Esses cenários incluem detecção

[1]Termo original em inglês para fluxo de dados.

de fraudes, previsões de inadimplência, previsão direcional (alta/baixa) de movimentos dos preços de ativos e recomendações de compra/venda (9).

Na classificação do conjunto de dados, conforme (10), existem diversos desafios, tais como, ruídos presente no fluxo de dados, viés do modelo, variância, desvio de conceitos, desvio de recursos, dependência de recursos, abundância de instâncias, valores ausentes, desequilíbrio, e *overfitting*. O método de aprendizado em lote[2] (também conhecida como aprendizado offline) não é recomendada para analisar fluxos de dados devido à sua incapacidade de lidar com desvio de conceito[3], servindo apenas em dados estacionários, uma vez que não aprende continuamente com os novos dados (10). Além disso, em métodos offline o processo de retreinamento é oneroso, a medida que os dados vão crescendo, consumindo mais recursos de processamento, memória de disco e tempo (1). Por outro lado, o aprendizado de máquina online (também conhecido como *streaming* ou aprendizado incremental) é um método de aprendizado contínuo que se adapta aos desvios de conceito, pois o modelo é treinado continuamente (a cada novo dado), tendo os parâmetros dos modelos atualizados a cada nova observação (4).

Apesar dos benefícios dos modelos online para análise de fluxos de dados, segundo (4), é importante encontrar o ponto de equilíbrio entre dois principais erros: viés e variância. É observado que o aumento na variância de um modelo leva a uma redução no viés, enquanto a redução na variância leva o aumento do viés em direção à complexidade do modelo. Isso ocorre porque, conforme mais parâmetros são adicionados ao modelo, sua complexidade aumenta, o que pode levar a uma variância maior. Em cenários online, onde os dados estão em constante mudança, aplicar um único modelo de decisão pode resultar em um desempenho inferior. Portanto, é fundamental garantir que o modelo seja capaz de generalizar de forma precisa e eficiente, sem subestimar ou superestimar a importância dos dados observados.

O método *ensemble*[4](também conhecido como comitê) no aprendizado de máquina online mostra-se promissor em relação à modelos individuais (10), solucionando uma variedade de problemas relacionados à classificação. Conforme argumenta (8), métodos *ensemble* são mais eficientes, versáteis e equilibram o viés e variância, resultando em uma maior precisão. Portanto, métodos *ensemble* são mais robustos e podem lidar com diferentes tipos de fluxos de dados, como também possíveis desafios específicos que possam ser enfrentados (10).

[2]Termo em inglês *Batch Learning*.
[3]Termo em inglês *Concept Drift*.
[4]Neste trabalho será utilizado o termo original em inglês *ensemble*.

1.1 OBJETIVOS

O objetivo principal deste trabalho é apresentar e desenvolver **métodos *ensemble* para classificação binária em aprendizado de máquina online, utilizando a combinação de vários classificadores heterogêneos**. A partir disso, os objetivos delineados incluem:

- Avaliar o desempenho em termos de **erros de predições** dos métodos *ensemble*;
- Avaliar a eficiência em termos de **tempo** necessário para predição e treinamento dos diferentes métodos *ensemble* propostos;
- Investigar a capacidade dos métodos *ensemble* em lidar de forma eficaz com o **desvio de conceitos** nos dados;

1.2 ORGANIZAÇÃO

Este trabalho está organizado da seguinte forma: A seção 2 apresenta a fundamentação teórica. A seção 3 descreve os métodos *ensemble* propostos. A seção 4 apresenta a análise e os resultados experimentais. Finalmente, na seção 5 são apresentadas as considerações finais.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão apresentados conceitos-chave relacionados ao trabalho desenvolvido para o entendimento ao decorrer da leitura, os conceitos abordados serão: Aprendizado de máquina, classificação binária, método *ensemble* e as métricas para avaliação do desempenho.

2.1 APRENDIZADO DE MÁQUINA

O aprendizado de máquina é uma subárea da inteligência artificial e tem se tornado cada vez mais relevante devido à sua capacidade de aprendizado automático a partir de dados. Os algoritmos de aprendizado de máquina podem identificar padrões nos conjuntos de dados e fazer previsões ou tomar decisões com base nos padrões identificados (9). A figura 1 ilustra os principais tipos de aprendizado de máquina.

Figura 1: Taxonomia para aprendizado de máquina baseado em (1)

O aprendizado por reforço é baseado na ideia de aprender com experiências e as recompensas ou punições associadas a elas. O sistema de aprendizado, chamado de agente, pode observar o ambiente, através dessa observação, decide o que fazer para realizar a tarefa dada, executa a ação para realizar essa tarefa e aprende se essa foi uma ação correta com base na recompensa. O algoritmo determina a chave de resposta por meio de sua experiência (9).

O aprendizado não supervisionado é empregado na extração de inferências a partir de conjuntos de dados de entradas não rotulados (9). Embora os resultados do aprendizado não supervisionado seja mais incerto, essa abordagem oferece a possibilidade de obter informações valiosas que anteriormente não estavam acessíveis apenas com a análise dos dados brutos (1).

Os métodos de aprendizado semi-supervisionado geralmente se situam entre os métodos supervisionados e não supervisionados. Esses métodos normalmente utilizam uma abundância de dados de treinamento não rotulados (formando o componente de

aprendizado não supervisionado) e uma pequena quantidade de dados pré-rotulados e anotados (formando o componente de aprendizado supervisionado)(1).

Os métodos de aprendizagem supervisionada são amplamente adotados na análise preditiva, principalmente devido ao seu objetivo fundamental de prever respostas para dados de entrada. Isso é essencialmente alcançado através do estabelecimento de uma associação entre os dados de entrada e as respostas correspondentes durante o treinamento do modelo. Há duas principais classes de problemas que o aprendizado de máquina supervisionado é aplicado: classificação e regressão. Dentro da classe problemas de Classificação, há dois tipos principais: a binária e multi-classes (1). O foco deste trabalho é problemas de classificação binária.

Para os algoritmos de aprendizado de máquina existem duas técnicas para o aprendizado, a mais tradicional que é o aprendizado em lote (do inglês *batch*) e o aprendizado online (1). O método de aprendizado em lote não é recomendado para um cenário de fluxos de dados realista que pressupõe que o conjunto de dados possa variar ao longo do tempo. Neste tipo de cenário, é comum acontecer o desvio de conceito, quanto as relações entre as variáveis de entrada e saída mudam ao longo do tempo. Com aprendizado de máquina tradicional, na presença de mudança de conceito, será necessário treinar o modelo do zero, e depois colocar novamente em produção, sendo muito oneroso fazer isso em termos de desempenho, processamento, armazenamento e tempo (1).

O método de aprendizado em lote é recomendado para cenários onde a distribuição do conjunto de dados seja estacionária (10). O método de aprendizado online funciona diferente, sendo adequado em cenários realistas, pois o processo de aprendizado é contínuo, ocorrendo tudo em produção, a cada novo dado o modelo é treinado e os parâmetros dos modelos são atualizados constantemente (4). Devido à aprendizagem ser contínua, não há necessidade de preocupar com restrição de memória, pois a cada novo dado o modelo é treinado, logo, não é necessário armazenar o histórico de dados (1).

2.2 CLASSIFICAÇÃO BINÁRIA

O principal objetivo da classificação é prever rótulos de saída ou respostas, os quais são de natureza categórica, a partir dos dados de entrada. Esses rótulos de saída denominam-se classes ou rótulos de classes, e representam categorias distintas e discretas. Dessa forma, cada resposta de saída é atribuída a uma classe ou categoria específica, significando que essas saídas têm seus valores não ordenados e discretos (1). A classificação binária identifica as instâncias como uma das duas classes predefinidas, sendo rótulos de classe positivos ou negativos (11).

A figura 2 ilustra um exemplo prático de uma tarefa de classificação binária apli-

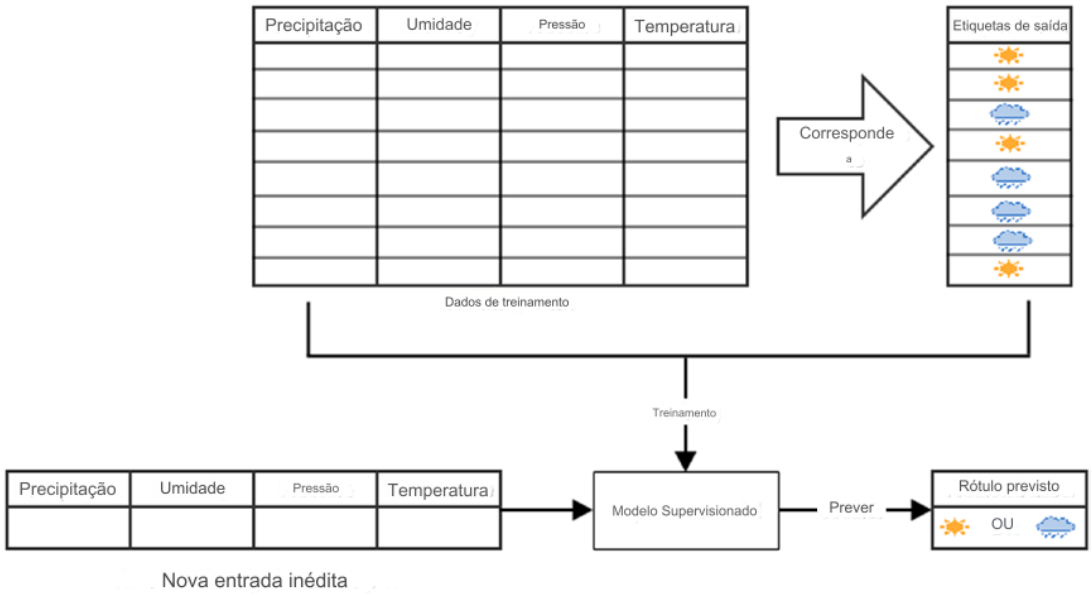

Figura 2: Exemplo de classificação binária aplicada na previsão do tempo (1).

cada à previsão do tempo. Neste caso, dispomos de um conjunto de dados de entrada composto por um vetor de atributos e características, tais como precipitação, umidade, pressão e temperatura. Com base nessas informações, duas possíveis classes podem ser identificadas: chuvoso ou ensolarado. Durante o treinamento do modelo supervisionado, são utilizadas amostras de dados de entrada contendo vetores de características (precipitação, umidade, pressão e temperatura) para cada observação, juntamente com suas respectivas etiquetas de classe, indicando se o tempo estará ensolarado ou chuvoso.

2.3 APRENDIZADO DE MÁQUINA ONLINE

O método de aprendizado de máquina online (OML[5]) representa uma abordagem inovadora em relação ao aprendizado de máquina tradicional (em lote). Ao contrário dos métodos tradicionais, em que o modelo deve ser totalmente treinado antes de ser colocado em produção, o OML pode ser colocado em produção imediatamente, sem a necessidade de treinamento e conhecimentos dos dados. Ele possui uma capacidade adaptativa e evolutiva única na sua aprendizagem, permitindo que seja treinado e ajustado continuamente à medida que novos dados são recebidos ao longo do tempo. Em essência, o modelo OML é capaz de aprender e prever dinamicamente a medida que novos dados chegam, sem a necessidade de reexecutar o treinamento completo com conjuntos de dados anteriores. Essa flexibilidade torna o OML altamente adequado para aplicações em que os dados estão em constante evolução e é crucial adaptar-se rapidamente as mudanças nas condições ou no ambiente (1).

[5]Abreviatura para o termo original em inglês: online machine learning.

2.4 ENSEMBLE

Ao longo do ciclo de vida de um ser humano, é evidente que ele se depara com diversos sistemas de decisões em consenso (*ensemble*). Esses sistemas, ou métodos de tomada de decisões, têm sido uma parte intrínseca da história humana, especialmente no contexto de convivência social. Um exemplo claro de tomada de decisões em consenso pode ser observado no funcionamento da democracia. Na democracia, fazendo analogia ao método *ensemble*, um amplo espectro de classificadores, representado pelos indivíduos, avalia potenciais candidatos por meio do processo de votação (parte posterior ao processo de treinamento, combinando as predições individuais), um desses candidatos é eleito (predição final). Este processo ilustra como o método *ensemble* funciona, ou seja, a combinação de várias opiniões individuais para alcançar uma melhor decisão coletiva (8).

Como ilustra a figura 3, o método *ensemble* consiste na técnica de combinar as previsões de vários modelos base (no contexto deste trabalho, os classificadores individuais) com base em uma estratégia de combinação para construir um modelo que seja capaz de obter um melhor desempenho na previsão geral (predição) em comparação ao modelo individual (11).

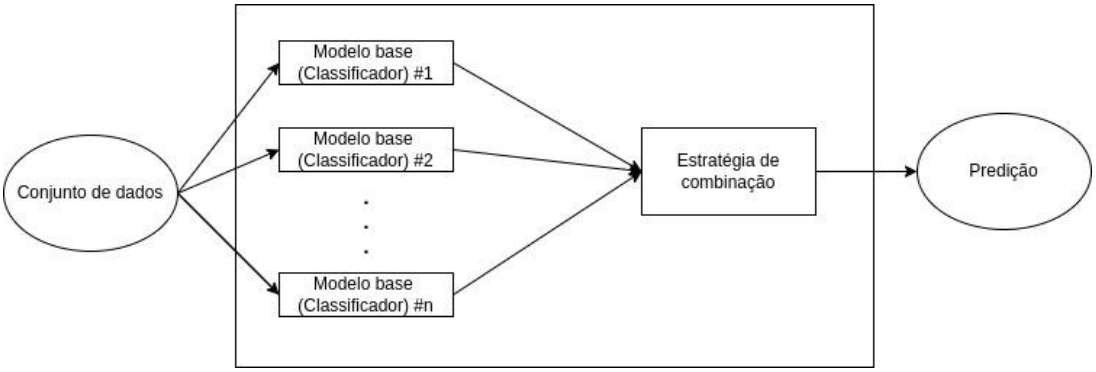

Figura 3: Diagrama de um *ensemble* (2).

Cada classificador base tem seu próprio domínio de competência, cada um com suas particularidades, e comportamento diferente principalmente em relação ao viés e variância (2). O *ensemble* visa buscar uma estratégia de combinação para representar melhor a decisão individual de cada classificador base a fim de melhorar o desempenho geral e a precisão preditiva final (2).

Segundo (4), existem três estratégias fundamentais na composição do método *ensemble*: a geração dos modelos base, a combinação das previsões individuais e a *reset*[6] ou reconfiguração dos modelos.

[6]Termo original em inglês.

A estratégia de geração determina o método pelo qual os modelos base são treinados. Nesse processo, é essencial garantir a diversidade entre os modelos base, utilizando diferentes subconjuntos de instâncias (observações), características, ou uma combinação de ambos (4).

A estratégia de combinação descreve o processo pelo qual as previsões individuais dos modelos base são combinadas para formar a previsão do método *ensemble*. É fundamental evitar previsões provenientes de um único modelo, as quais poderiam comprometer o desempenho do *ensemble*, sendo importante a diversidade da composição dos modelos base, para que ele possa escolher dentre os melhores modelos para sua previsão ou considerar todos para compor a previsão (4).

A estratégia de *reset* é essencial para determinar como e quando os modelos base são ajustados ou atualizados para refletir alterações na distribuição dos dados, especialmente em cenários de desvio de conceito. O propósito da redefinição dos modelos base é evitar que um modelo subajustado prejudique as previsões finais do *ensemble*. Existem duas abordagens para manter os modelos base atualizados diante do desvio de conceito: estratégia reativa e a estratégia ativa.

Na estratégia reativa, utilizam-se detectores de desvio para acionar ajustes nos modelos base (4). O ADWIN (*ADaptive Window*) é um exemplo que utiliza a estratégia reativa, sendo um método popular para detecção de desvio de conceito com garantias matemáticas (12) e o método mais conhecido que compara duas janelas deslizantes (2). Ele mantém uma janela de tamanho variável das observações mais recentes, de modo que se mantenha a premissa de que não houve mudança na distribuição dos dados. Essa janela é dividida em duas sub-janelas, usados para determinar se ocorreu um desvio de conceito. Ele compara o valor médio das duas sub-janelas para identificar se correspondem à mesma distribuição. Se a igualdade da distribuição não for mais válida, então é detectado o desvio de conceito. O ADWIN utiliza um valor de significância para determinar se as duas subjanelas correspondem à mesma distribuição (12).

Já os métodos ativos, eles redefinem os modelos conforme as estratégias predefinidas, como intervalos de tempo fixos. É importante ressaltar que a redefinição dos modelos base tem impacto significativo nas previsões finais do *ensemble* (4).

2.5 MÉTRICAS PARA ANÁLISE DO DESEMPENHO

A métrica de avaliação visa quantificar o desempenho de um modelo na sua predição. Essa ferramenta é essencial, permite uma avaliação sistemática e objetiva da capacidade de um modelo em fazer previsões precisas com base nos dados fornecidos, no contexto da classificação, as métricas analisam a concordância entre os rótulos de classe esperados e os

rótulos de classe previstos pelo modelo. Além disso, as métricas também permitem interpretar as probabilidades previstas para os rótulos de classe, oferecendo *insights* adicionais sobre a confiança das previsões do modelo (3).

Nos problemas de classificação binária, a avaliação da eficácia da solução ideal durante o treinamento pode ser realizada por meio da matriz de confusão tabela 1). Nesta matriz, as linhas representam as classes previstas pelo modelo, enquanto as colunas representam as classes reais dos dados. Os valores Verdadeiro Positivo (VP) e Verdadeiro Negativo (VN) indicam o número de instâncias positivas e negativas que foram corretamente classificadas. Por outro lado, os valores de Falso Positivo (FP) e Falso Negativo (FN) representam o número de instâncias que foram incorretamente classificadas. Com base nessas informações da matriz de confusão, várias métricas podem ser calculadas para avaliar o desempenho de um classificador (5).

	Classe Positiva Real	**Classe Negativa Real**
Classe Positiva Prevista	Verdadeiro Positivo	Falso Negativo
Classe Negativa Prevista	Falso Positivo	Verdadeiro Negativo

Tabela 1: Matriz de confusão para classificação binária (5).

As métricas de avaliação para classificação binária são organizadas em três grupos distintos, conforme representado na figura 4, e serão exploradas nas seções subsequentes. Ressalta-se que, neste trabalho, foram usadas as métricas[7] de desempenho para classificação disponíveis na biblioteca *River*. As próximas seções apresentam as métricas que foram usadas neste trabalho.

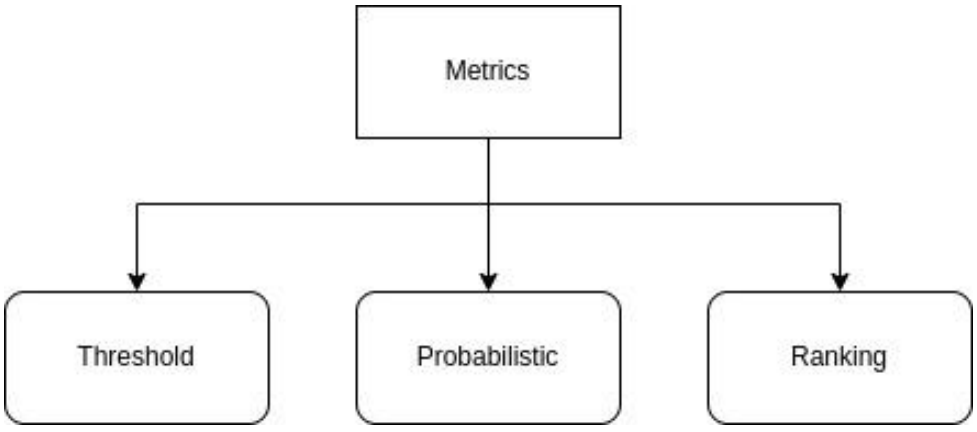

Figura 4: Taxonomia das métricas para classificação binária (3).

[7]Neste trabalho todos os nomes das métricas são originais em inglês.

2.5.1 *THRESHOLD*

Esse grupo de métricas é usado para ter uma compreensão quantitativa dos erros que um classificador teve durante suas previsões. Essas métricas são amplamente usadas nos classificadores onde o objetivo é minimizar o número de erros e itens mal classificados. Dentro desse grupo de métricas, deve compreender que cada métrica tem sua aplicabilidade, levando em conta que algumas são mais apropriadas para conjuntos de dados equilibrados e outras para desequilibrados, para detecção de falha e outras para sinal, ou para tarefas de recuperação de informações (13). Nesse grupo há dois subgrupos de métricas: *precision* e *sensitivity-specificity* (3), que serão discutidos a seguir.

- ***PRECISION*:** Métricas do tipo *precision* avaliam a proporção de verdadeiros para as classes previstas pelo modelo. No contexto deste trabalho são usados as seguintes métricas de *precision*:

  - **Accuracy**: Essa métrica avalia a porcentagem de previsões corretas em relação ao total de instâncias avaliadas. Equação da *accuracy* (5):

$$Accuracy = \frac{(VP + VN)}{(VP + FP + VN + FN)}$$

  - **Precision**: Essa métrica avalia o total de classe verdadeira positiva (VP) que são corretamente previstas em relação ao total de classes positivas (VP + FP) previstas. Equação da *precision* (5):

$$Precision = \frac{VP}{(VP + FP)}$$

  - **Weighted precision**: Essa métrica é uma variação da *precision* que calcula a média ponderada da *precision*. Equação da *weighted precision*, onde W_i é o peso associado a classe, i corresponde a classe, e n o número de classes:

$$Weighted precision = \frac{\sum_{i=1}^{n} W_i * \frac{VP_i}{(VP_i + FP_i)}}{\sum_{i=1}^{n} W_i}$$

  - **Balanced accuracy**: Essa métrica calcula a média de *recall* obtida para cada classe. Equação (14):

$$Balanced accuracy = \frac{1}{2}(\frac{VP}{VP + FN} + \frac{VN}{VN + FP})$$

  - **Cohen Kappa**: Essa métrica avalia o nível de concordância entre dois avaliadores. Equação da *Cohen Kappa*, onde P_i é a probabilidade empírica de

concordância sobre a classe atribuída a qualquer amostra, uma precisão sequencial, e P_k é a concordância esperança quando ambos avaliadores atribuem rótulos aleatórios (15):

$$CohenKappa = \frac{P_i - P_k}{1 - P_k}$$

- ***SENSITIVITY-SPECIFICITY*:** O *threshold sensitivity* avalia como a classe verdadeiro positivo (VP) são corretamente classificadas em relação ao total positivo. Por outro lado, existe *threshold specificity*, sendo o complemento ao *sensitivity*, avaliando o quão bem a classe negativa foi prevista (3). Neste trabalho não foi utilizado uma métrica específica para especificidade, mas foram utilizadas métricas híbridas, que faz a junção da *sensitivity* e *specificity*, também a junção da *sensitivity* com *precision*, entre outras. Equação da *sensitivity* e *specificity* (5):

$$Sensitivity = \frac{VP}{(VP + FN)}$$

$$Specificity = \frac{VN}{(VN + FP)}$$

  - **Recall**: Essa métrica avalia a taxa de verdadeiros positivos e mede o quão bem os verdadeiros positivos foram previstos. Equação do *Recall* (5):

$$Recall = \frac{VP}{(VP + FN)}$$

  - **Weighted Recall**: Essa métrica calcula a média ponderada do *Recall*. Na equação que representa a métrica *Weighted Recall* (5), W_i é o peso associado a classe, i corresponde a classe, e n é o número de classes. A equação da métrica *Weighted Recall* está representada abaixo:

$$WeightedRecall = \frac{\sum_{i=1}^{n} W_i * \frac{VP_i}{(VP_i + FN_i)}}{\sum_{i=1}^{n} W_i}$$

  - **Geometric mean**: Essa métrica é utilizada para otimizar tanto a taxa de verdadeiros positivos (VP) quanto a taxa de verdadeiros negativos (VN) garantindo, ao mesmo tempo, um equilíbrio entre ambas as taxas (5). Essa métrica combina a *sensitivity* e *specificity*. Segue a equação da métrica *Geometric mean* conforme apresentado por (3):

$$GeometricMean = \sqrt{\frac{VP}{(VP + FN)} * \frac{VN}{(VN + FP)}}$$

– **F1-Score**: Essa métrica calcula a média harmônica entre os valores do *recall* e *precision* (5). Segundo (13), essa métrica é amplamente usada em cenários que precisam de um *recall* mais preciso (5). Segue a equação:

$$F1 - Score = \frac{2*(\frac{VP}{(VP+FP)}*\frac{VP}{(VP+FN)})}{(\frac{VP}{(VP+FP)} + \frac{VP}{(VP+FN)})}$$

– **F-Beta**: Essa métrica é uma variação da *F1-Score*, calcula a média ponderada entre os valores de *recall* e *precision*, controlado pelo parâmetro do coeficiente β (3). Quando o β é igual a 1, a pontuação é igual à pontuação do $F1$, o que indica o equilíbrio entre a *precision* e *recall*. Contudo, quando β é maior que 1, a métrica dá mais peso para o *recall*, dando importância na capacidade do modelo em classificar positivos. Por outro lado, quando β é menor que 1, a métrica atribui mais peso a *precision*, dando importância na capacidade do modelo de evitar falsos positivos. Essa flexibilidade no ajuste do β permite adaptar conforme as necessidades e preferências específicas de avaliação do modelo. Segue a equação:

$$F - Beta = (1 + \beta^2) * \frac{(\frac{VP}{(VP+FP)} * \frac{VP}{(VP+FN)})}{(\beta^2 * \frac{VP}{(VP+FP)} + \frac{VP}{(VP+FN)})}$$

2.5.2 *PROBABILISTIC*

Esse grupo de métricas é usado para obter uma compreensão probabilística do erro. Essas métricas são utilizadas nos classificadores buscando avaliar a confiabilidade, indo além da simples detecção de falhas, mas mostrando se o classificador selecionou a classe falsas com uma probabilidade alta ou baixa. Tais métricas desempenham um papel fundamental ao mostrar a eficaz de modelos *ensemble*, pois verifica se a combinação dos modelos base foram adequadas para um melhor desempenho, se essa combinação teve uma confiança devido à seleção baixa nas classes falsas (13).

- **Log loss**: Essa métrica (também conhecida como entropia cruzada) é utilizada para avaliar a qualidade das probabilidades previstas (3). Segundo (13), essa métrica é importante quando a calibração das probabilidades é crucial, pois avalia a precisão das previsões do modelo, mas também a sua confiança nessas previsões. A equação do *log loss* (3), onde y_i são os valores previstos e y os valores verdadeiros:

$$\log loss = -((1 - y) * \log(1 - y_i) + y * \log(y_i))$$

- **MCC (Matthews Correlation Coefficient)**: Segundo (16), essa métrica é a

melhor para descrever a matriz de confusão, pois faz a correlação entre as classes observadas e as previstas. Segue a equação:

$$MCC = \frac{VP * VN \ - \ FP * FN}{\sqrt{(VP \ + \ FP)(VP \ + \ FN)(VP \ + \ FP)(VN \ + \ FN)}}$$

2.5.3 *RANKING*

Esse grupo de métricas é usado para mensurar a capacidade do modelo de classificar, ou seja, avalia a eficácia dos classificadores na separação das classes. Essa avaliação é crucial em cenários nos quais a distinção nítida entre as classes é fundamental para o desempenho do modelo (13). Segundo (3), a métrica mais utilizada desse grupo é a ROC Curve[8].

- **ROC Curve**: Essa métrica (na biblioteca *River* essa métrica chama-se ROC AUC) avalia o desempenho do modelo mediante uma representação gráfica combinado a métrica de *recall* (algumas literaturas chamam de taxa de verdadeiro positivo) sob o eixo Y e a taxa de falsos positivos sob o eixo X (sendo a probabilidade de alarmes falsos) para um conjunto de previsões pelo modelo em diferentes *threshold* (3).

$$Recall = \frac{VP}{(VP + FN)}$$

$$Taxa de falsos positivos = \frac{FP}{(FP + VN)}$$

2.5.4 *VERSÃO ROLLING WINDOW*

A técnica de janela deslizante ou rolante (do inglês *rolling window* oferece uma abordagem flexível e poderosa para análise contínua e atualizada do desempenho de modelos, enquanto elimina as flutuações de curto prazo. Essa técnica faz o cálculo da média da métrica usada ao longo de um *buffer* com o tamanho de janela das observações mais recentes. À medida que novas observações são incorporadas, o *buffer* é preenchido até atingir a sua capacidade máxima, momento que a observação mais antiga é descartada para dar lugar a mais recente. A quantidade de observações que compõem o *buffer* é determinada pelo tamanho da janela. A cada cálculo do *rolling window*, após o preenchimento do *buffer*, a janela é movida adiante para as novas observações que chegam. A equação da *rolling window* está apresentada abaixo:

[8]Neste trabalho utiliza-se o termo original em inglês, a tradução é Curva ROC. ROC é a abreviatura de *Receiver Operating Characteristic*, a tradução é: Característica de Operação do Receptor.

$$Rolling\ window_i^t = \frac{1}{WINDOWsize} \sum_{k=t-WINDOWsize}^{t} P_i^k$$

Na equação o cálculo de *rolling window* é feito através da predição i no momento t, $WINDOW_{size}$ é o tamanho da janela, e P_i^k é a métrica de performance na predição i no tempo k (4). Para todas as métricas de performance para classificação foi criado uma versão *Rolling window*, com o tamanho de janela fixa de 100.

3 MÉTODOS ENSEMBLE ONLINE PARA CLASSIFICAÇÃO BINÁRIA

Esta seção explora a arquitetura e as estratégias dos métodos *ensemble* propostos para o processamento de fluxos de dados dinâmicos. Inicialmente, são apresentados detalhes sobre a composição do *ensemble*, destacando a natureza agregativa dos modelos base e sua interação no processo de predição. Em seguida, são delineadas as estratégias adotadas para a seleção, geração, reset e combinação dos modelos base.

A figura 5 ilustra o diagrama de classes da UML que representa a composição do *ensemble*. É possível observar que o ensemble é uma agregação de modelos base de classificadores. Tanto os classificadores e métodos *ensemble* possuem várias métricas de desempenho para avaliação das predições feitas, as métricas utilizadas são discutidas na seção 2.5. A composição dos modelos base vai variar conforme a estratégia de combinação, essas estratégias de combinação são descritas na seção 3.4.

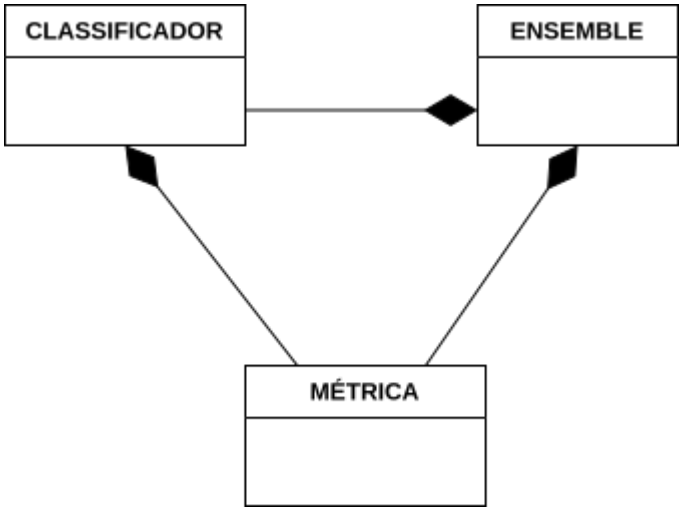

Figura 5: Representação de um *ensemble*.

A figura 6 ilustra a arquitetura do *ensemble* proposto para processar o fluxo de dados. A primeira etapa do *ensemble* envolve a construção de múltiplos modelos base heterogêneos, seguida pela avaliação de desempenho de cada um deles por meio de métricas de desempenho específicas a cada nova observação. Quando a entrada do fluxo de dados recebe uma nova observação, todos os modelos base são alimentados com essa nova observação e é posteriormente avaliado a performance para cada modelo através das métricas de performance. Uma dessas métricas de performance vai ser selecionada para fazer a estratégia de combinação nos métodos ensemble propostos *threshold*, *average* e *best model* para eleger a composição dos modelos base para a predição.

Uma das vantagens do *ensemble* proposto é a independência dos modelos base, isso permite explorar diversas estratégias de combinação dos modelos base para otimizar

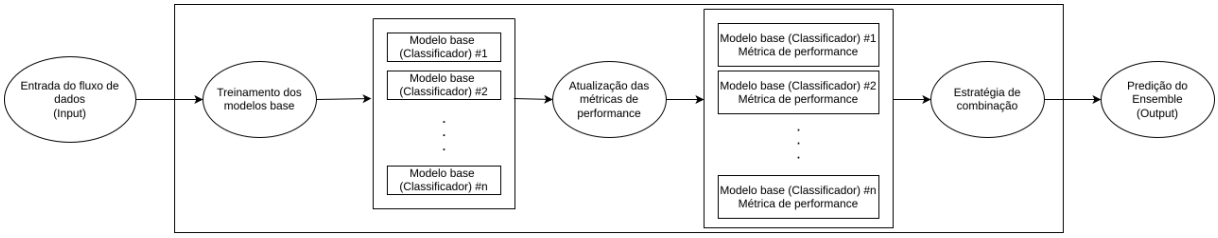

Figura 6: Arquitetura do *ensemble* para processar o fluxo de dados.

a predição do *ensemble* e fazer futuras otimizações nos modelos proposto. Conforme destacado por (4), o tempo é um fator fundamental para o processamento de fluxos de dados. Os modelos base de aprendizado de máquina online utilizados neste trabalho são leves e possuem rotinas de gerenciamento de memória eficientes, o que contribui significativamente para o desempenho em termos de tempo.

É importante ressaltar que, embora a utilização de modelos de aprendizado de máquina online na composição do *ensemble* contribua para reduzir o tempo, o processamento do ensemble foi arquitetado de forma sequencial, fazendo um aumento significativo em relação ao tempo. A avaliação do desempenho em relação ao tempo dos métodos de *ensemble* propostos são detalhadas na seção 4.3. Além disso, na seção 5, foi abordado um possível trabalho futuro para otimização dos métodos de *ensemble* propostos para melhorar o desempenho em relação ao tempo.

As próximas seções descrevem o ensemble proposto. A seção 3.1 discorre sobre a proposta de seleção dos modelos base para o *ensemble*. Na seção 3.2 é descrito como é o processo de treinamento dos modelos base e sua geração. Posteriormente é apresentado as estratégias de *reset* dos modelos na seção 3.3. Por fim, na seção 3.4 é descrito as propostas das estratégias de combinação adotada neste trabalho.

3.1 SELEÇÃO DOS MODELOS BASE

Conforme destacado em (2), um *ensemble* de classificação ideal deve possuir classificadores individuais como modelos base que sejam mutuamente complementares caracterizados por sua diversidade. Conforme sugerido, a importância da diversidade dos classificadores na composição do ensemble não pode ser subestimada, pois desempenha um papel importante no aumento da precisão do ensemble em suas predições. Assim, a seleção de classificadores para compor o método ensemble não tem que ser limitada apenas à precisão individual, mas deve ser levado em conta também a diversidade dos classificadores, pois essa diversidade é necessária para enfrentar os desafios inerentes aos fluxos de dados. Segundo Zhou (*aput* (4)), o desempenho de um *ensemble* melhora com o aumento do número de modelos base.

Neste trabalhado, foi utilizado a biblioteca *River*[9], que resulta da fusão de duas bibliotecas especializadas em fluxo de dados: Creme e scikit-multiflow. Conforme citado por (17), "*river* é uma biblioteca de aprendizado de máquina voltada para fluxos de dados dinâmicos e aprendizado contínuo", e é também de código aberto. Os resultados obtidos no trabalho (17) com os algoritmos OML do *river* demonstraram desempenho igual ou superior aos algoritmos das bibliotecas Creme e scikit-multiflow. Vários algoritmos online têm suas raízes em versões offline ou em métodos tradicionais de aprendizado de máquina.

Considerando esses fatores para a seleção dos modelos base, foram selecionados os algoritmos[10] da biblioteca *river* capazes de fazer classificação binária. São eles: *Bernoulli Naive Bayes*, *Adaptive Random Forest classifier*, *Hard sampling classifier*,*Random over-sampling*, *Random sampling*, *Random under-sampling.*, *Logistic regression*, *Perceptron*, *One Vs One classifier*, *One Vs Rest classifier*, *Output Code classifier*, *Extremely Fast Decision Tree classifier*, *Hoeffding Adaptive Tree classifier*, *Hoeffding Tree Classifier*, *ADWIN Bagging classifier*, *ADWIN Boosting Classifier*, *BOLE Classifier*, *Ada Boost Classifier*, *Bagging classifier*, *Leveraging Bagging Classifier*, *SRP Classifier*, *Stacking Classifier*, *Voting Classifier*, *AMF Classifier*, *ALMA Classifier*, *PA Classifier*, *Softmax*, *ComplementNB*, *GaussianNB*, *MultinomialNB* e *KNN Classifier*.

Vale ressaltar que não foi feita nenhuma técnica de otimização dos modelos com *tuning*[11]. Foram utilizadas para cada modelo base a forma padrão de configurações dos parâmetros e adotada uma mesma *seed*[12] para torná-los determinísticos, para possibilitar a reprodutibilidade dos resultados obtidos.

3.2 ESTRATÉGIA DE GERAÇÃO

A estratégia de geração é o processo de treinamento dos modelos base que compõem o método *ensemble*. Na abordagem proposta por este trabalho, primeiro foi construído todos os modelos base a partir dos algoritmos supracitados e depois assegurou-se que cada modelo base fosse treinado a cada nova observação.

Neste trabalho, foi adotado a abordagem de independência dos modelos. Segundo (4), a abordagem da independência dos modelos base é usada para melhorar o desempenho do *ensemble*, pois proporciona melhor capacidade de generalização, uma vez que cada modelo base apresenta comportamentos diferentes em relação aos fluxos de dados. Além disso, permite também a diversidade de técnicas de combinação na modelagem do *ensemble*. Por fim, para avaliar o desempenho dos modelos base e também do *ensemble*, são utilizadas um conjunto de métricas baseadas em erros que mensuram a "qualidade"das

[9] https://riverml.xyz/dev/

[10] Os nomes dos modelos estão conforme a biblioteca *river*

[11] Técnica de ajustar os parâmetros do algoritmo para alcançar um melhor desempenho.

[12] Termo original em inglês, a tradução é: semente.

predições das classificações.

3.3 ESTRATÉGIA DE *RESET*

O uso mais comum de métodos *ensemble* é permitir a adaptação ao desvio de conceito nos fluxos de dados (10). Segundo (4), a estratégia de *reset* procura enfrentar o desvio de conceito e manter os modelos de aprendizado base atualizados, com o propósito de evitar que modelos subajustados influenciem as previsões do *ensemble*.

Os modelos de *ensemble* podem adotar estratégias ativas ou reativas para enfrentar o desvio de conceito (10). Neste trabalho foi utilizado uma estratégia híbrida entre a ativa e reativa. Os modelos base levam em consideração a recentidade dos erros de previsão (estratégia ativa), dando maior importância às previsões corretas feitas em amostras mais recentes do fluxo de dados em detrimento das previsões corretas nas amostras mais antigas.

Na composição dos modelos base, existem modelos que adotam estratégias reativas para enfrentar o desvio de conceito. Por exemplo, os modelos *ADWINBaggingClassifier* e *ADWINBoostingClassifier* utilizam o algoritmo *ADWIN* como detector de desvio de mudança (estratégia reativa). Nesta estratégia, o pior membro do conjunto é substituído por um novo (18). Além disso, alguns modelos base também são soluções ensemble. Por exemplo, *ARFClassifier* e *SRPClassifier* são modelos ensemble que combinam estratégias reativas e ativas.

3.4 ESTRATÉGIA DE COMBINAÇÃO

De acordo com (2), *ensemble* de classificação é uma abordagem atrativa para classificar os fluxos de dados, porque tem uma maior adaptabilidade aos desvios de conceito. A adaptação do *ensemble* pode ser feita alterando a composição dos modelos base para sua predição, isso significa a adição ou remoção dos modelos base que vão ser combinados, por isso se torna um fator-chave para uma maior adaptabilidade e desempenho a seleção dos modelos de aprendizado de máquina.

A estratégia de combinação descreve as técnicas de combinação das predições dos modelos base para formar a predição do *ensemble* (4). A combinação das predições tem um impacto significativo no desempenho, podendo tanto melhorar quanto prejudicar (10). Segundo Cha Zhang (*apud* (8)), "combinar os modelos base de classificação em um *ensemble* não garante que o modelo resultante possuirá uma performance de classificação maior do que a performance do melhor modelo base presente no *ensemble* para aquela base, mas reduz drasticamente as chances do modelo resultante possuir uma performance ruim". No método *ensemble* foram utilizadas quatro estratégias de combinação. As próximas seções apresentam a descrição de cada uma delas.

3.4.1 ESTRATÉGIA DE COMBINAÇÃO *VOTING*

Essa estratégia de combinação é a da votação majoritária, sendo a mais popular técnica de combinação de *ensemble* em problemas de classificação (3). Na votação majoritária, cada classificador base tem o mesmo peso na contribuição da decisão final do *ensemble*. Cada classificador base vota em um rótulo da classe, logo a previsão final é o rótulo da classe que a maioria dos modelos base votou. Para critérios de empate na votação foi adotado duas abordagens, uma é que a preferência para uma das classes se o número de modelos base fosse um número par, foi escolhida a preferência classe positiva, outra abordagem adotada foi o número de modelos base que compõe *ensemble* é ímpar (são 29 modelos base), no caso é garantido que não vai haver empate (10).

3.4.2 ESTRATÉGIA DE COMBINAÇÃO *AVERAGE*

Essa estratégia representa uma variação da votação majoritária. Aqui, somente os modelos base cujo desempenho excedeu a média geral de uma determinada métrica de desempenho (no caso, a *accuracy* foi selecionada), são incluídos na composição de um novo subconjunto para votação. Dado que M seja o conjunto de todos os modelos base. Assim, M' vai ser o subconjunto de M definido da seguinte forma:

$$M' = \{m \mid m \in M \mid P_m > \frac{\sum_{i=1}^{M} P}{N}\}$$

Onde P_m é a métrica de desempenho do modelo base m e vai fazer parte da composição do subconjunto M' apenas os modelos base que tiver o desempenho maior do que a média geral da métrica. Na média é calculado o somatório da métrica selecionada P de cada modelo base e dividido pela quantidade de todos os modelos base disponível N do conjunto M. Com o subconjunto M' formado, é posteriormente usado a estratégia de voto majoritário para ter a previsão do *ensemble*.

3.4.3 ESTRATÉGIA DE COMBINAÇÃO *THRESHOLD*

Essa estratégia *threshold*[13] define um limiar fixo de uma métrica de desempenho para selecionar os modelos base na composição da previsão do conjunto, limitando a contribuição dos modelos base que não atendem a um certo nível de qualidade (4).

Neste trabalho, foi selecionada um limiar de 0.8 da métrica *accuracy*. Desta forma, apenas os modelos base com desempenho acima desse limiar serão considerados na predição ensemble e posteriormente usada a estratégia de votação majoritária para a

[13]Termo original em inglês.

previsão final do modelo. Dado que M seja o conjunto de todos os modelos base e τ seja o limiar de desempenho do modelo. Assim, M' vai ser o subconjunto de M definido da seguinte forma:

$$M' = \{m \mid m \in M \mid P_m > \tau\}$$

Onde P_m é a métrica de desempenho selecionada do modelo base m. Com o subconjunto M' formado, é posteriormente usado a estratégia de voto majoritário para ter a previsão do *ensemble*. É importante destacar que o desempenho de M' varia ao longo do tempo, pois os modelos base são afetados pelas etapas de treinamento.

3.4.4 ESTRATÉGIA DE COMBINAÇÃO *BEST MODEL*

Segundo (4), essa estratégia é simples, porém eficaz, pois não precisa fazer cálculos da composição dos modelos base para seleção e nem fazer combinações das previsões. Nessa estratégia, o *ensemble* mantém um registro contínuo do desempenho dos modelos base ao longo do tempo, selecionando sempre o modelo com o melhor desempenho para gerar as previsões. Embora essa estratégia adote uma dinâmica simples, ela se revela promissora em cenários caracterizados por variações significativas do desempenho dos modelos individuais (4).

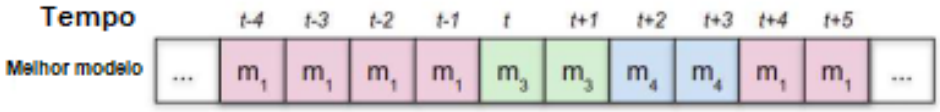

Figura 7: Exemplo da estratégia *best model* (4)

A figura 7 ilustra o funcionamento da estratégia em ação ao longo do tempo na previsão do *ensemble*. Para cada observação (momento t) a métrica de desempenho selecionada que vai eleger o melhor modelo (através de seu desempenho no treinamento) para fazer a predição do *ensemble* nesse momento t. Até o momento $t-1$, o modelo base m_1 é selecionado para realizar a predição do *ensemble* devido ao seu destaque na métrica de desempenho específica (neste trabalho foi selecionado a *accuracy*), isto é, o valor de m_1 é o mais elevado na métrica específica entre todos os modelos base no conjunto M até o momento t. Nos momentos t e $t+1$, o modelo m_3 é selecionado para realizar a predição do *ensemble*. Em seguida, nos momentos $t+2$ e $t+3$, é o modelo m_4 selecionado, pois teve o melhor desempenho nesses momentos. Por fim, a partir do momento $t+4$ o modelo m_1 volta a ser selecionado como melhor modelo para fazer a predição do *ensemble*.

Essa estratégia demonstra sua eficácia ao adaptar-se dinamicamente nas variações no desempenho dos modelos, garantindo assim a seleção do modelo mais adequado para cada momento t (4).

4 AVALIAÇÃO

Esta seção apresenta as avaliações experimentais realizadas nos métodos ensemble propostos. Esta seção está organizada da seguinte forma: a seção 4.1 descreve a abordagem GQM utilizada para estruturar a avaliação dos métodos ensemble. A seção 4.2 apresenta as métricas para responder o GQM, os conjuntos de dados utilizados e o *setup* para os experimentos. A seção 4.3 traz os resultados dos experimentos. Na seção 4.4 são feitas as análises dos resultados dos experimentos. Por fim, a seção 4.5 apresenta as respostas para as questões de pesquisas estabelecidas neste trabalho.

Todos os experimentos foram codificados utilizando a linguagem de programação Python 3.10. Os algoritmos de aprendizado de máquina foram obtidos do framework River. O código-fonte, dados e resultados deste trabalho são facilmente reproduzíveis e estão acessíveis publicamente no GitHub[14].

4.1 GQM (*Goal Question Metric*)

A abordagem GQM é empregada neste trabalho, pois ela proporciona uma estrutura metodológica sólida para análise de dados resultantes dos experimentos conduzidos. Essa abordagem visa responder a questões específicas e orientar o processo de avaliação, fornecendo *insights* valiosos (19). Conforme ilustra a figura 8, o GQM é um modelo estruturado hierárquico de três camadas (meta/questão/métrica), em que cada camada é refinado progressivamente.

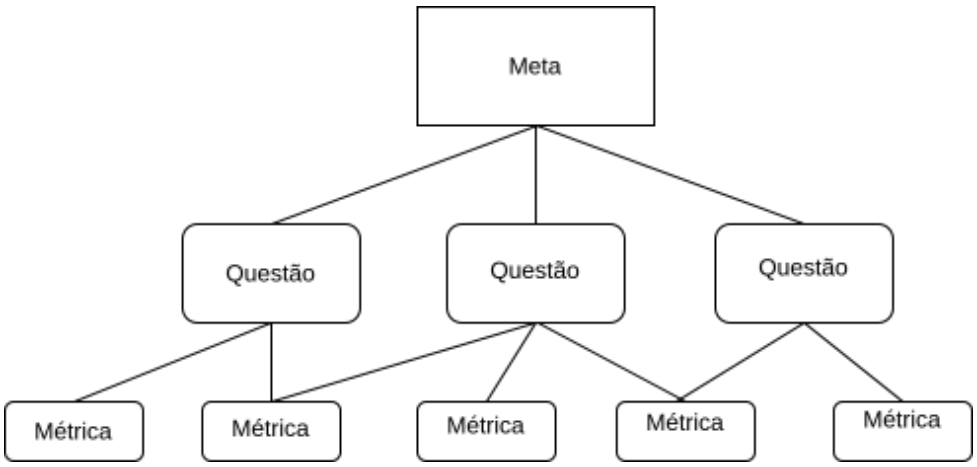

Figura 8: Estrutura do GQM.

Os *Goals*[15] representam os fenômenos a serem analisados, sendo que para cada *goal* são definidas uma ou várias questões (*Questions*). Essas questões descrevem o objeto de

[14]https://github.com/ProgrammeurArthur/OnlineEnsembleForBinaryClassifier
[15]Neste trabalho utiliza-se o termo original em inglês, a tradução desse termo é Meta ou Objetivo.

medição sob diferentes perspectivas. Cada questão está associada a um conjunto específico de métricas (*Metrics*), o que proporciona uma visão quantitativa. Assim, as questões representam um papel fundamental do detalhamento quantitativo do GQM, contribuindo para uma análise precisa e sistemática (4).

Para este trabalho, foi definida a seguinte *Meta*: **Avaliar a eficácia dos métodos *ensembles* e suas habilidades preditivas para diferentes conjuntos de dados.** Diante desta Meta, foram especificadas as seguintes *Questões*:

- **Q1** - Os métodos ensembles propostos superam os modelos individuais em termos de erro de predição?
- **Q2** - A capacidade da sensibilidade de predizer as classes verdadeiro positivo corretas dos modelos ensemble propostos em cada conjunto de dados, são melhores que os modelos base?
- **Q3** - Como se comporta a avaliação da predição dos modelos propostos e individuais na concordância das classes verdadeiras em sua predição, as verdeiras positivas e as verdadeiras negativas, em cada conjunto de dados?
- **Q4** - Os métodos ensembles se comportam melhor do que os modelos individuais em algum conjunto de dados com desvio de conceito?
- **Q5** - Em relação à eficiência do tempo de treinamento e predição, os métodos ensembles são superiores aos modelos individuais?

O desempenho da classificação no aprendizado de máquina é importantíssimo para demonstrar quão eficaz é um classificador em diferenciar corretamente as classes (20). As métricas de avaliação de software desempenham um papel fundamental na medição e melhoria na qualidade do software. No contexto do modelo GQM as métricas são selecionadas de forma criteriosa e associada a cada pergunta específica formulada. Essas métricas são projetadas para fornecer respostas mensuráveis às questões levantadas, permitindo uma avaliação objetiva do desempenho do classificador (4). A tabela 2 apresenta as métricas utilizadas para responder às perguntas do GQM relacionadas aos métodos *ensemble* propostos.

A próxima seção descreve o ambiente computacional usada para execução dos experimentos.

4.2 SETUP PARA OS EXPERIMENTOS

Os experimentos foram conduzidos na plataforma do Google Colab, na versão pro, utilizando um ambiente de execução baseado em CPU com processador Intel Xeon @

Métrica	Descrição	Questão
Accuracy	Avalia a pontuação da precisão por meio de correspondências corretas.	Q1 e Q3
Precision	Avalia o desempenho através da pontuação do modelo através do acerto das classes verdeiros positivos.	Q1 e Q2
MCC (Matthews correlation coefficient)	Avalia a qualidade das previsões de um modelo.	Q3
Balanced Accuracy	Avalia a média de *Recall* obtida em cada classe proporcionando identificar possíveis vieses.	Q3
Recall	Avalia a capacidade da sensibilidade do modelo de identificar corretamente as classes verdadeiros positivos.	Q2
Cohen Kappa	Avalia o nível de concordância entre previsões e classes verdadeiras.	Q3
ROC Curve	Avalia o desempenho do modelo em classificar, mostrando a sensibilidade dele em separar as classes corretamente.	Q1 e Q2
Rolling ROC Curve	Avalia o desempenho através das janelas deslizantes, assim sendo uma métrica para avaliar se o modelo teve desvio de conceito e como se comportou	Q1, Q2 e Q4
Tempo de Treinamento e Predição	Avalia a eficiência dos modelos através do cálculo do tempo de treinamento e predição.	Q5

Tabela 2: Métricas para avaliação dos modelos propostos.

2.20GHz (Família da CPU: 6). O sistema disponibilizou 51 GB de RAM e 225.8 GB de armazenamento em disco para os testes realizados nesta pesquisa.

Considerando os métodos *ensemble* propostos, a métrica *accuracy* foi adotada na estratégia de combinação. É importante destacar que nenhum ajuste de parâmetros dos modelos foi realizado por meio de técnicas de otimização para os modelos base, como o *tunning*. Para cada modelo base, foram utilizadas a configuração padrão dos parâmetros e para os algoritmos que possuem na sua configuração ajuste da semente (seed), foi configurada uma mesma semente para garantir que os resultados desses algoritmos fossem determinísticos, possibilitando a reprodutibilidade dos resultados obtidos (para cenários reais não deve ser fixado uma mesma semente, para uma melhor adaptabilidade dos modelos a cada nova observação).

Os métodos propostos juntamente com os modelos base processaram os diferentes *datasets* reais e sintéticos. A tabela 3 mostra os oito *datasets*, detalhando a quantidade

de amostras e características (*features*). O gerador de fluxo *ConceptDrift*, da biblioteca *River*, foi configurado para ter o desvio de conceito gradual (parâmetro: *position*) na posição especificada da largura do espaço do fluxo (parâmetro: *width*) de modo que a cada nova instância gerada nesse espaço do fluxo, pertença ao novo conceito após o desvio. A mudança de conceito ocorreu em apenas uma posição designada (posição: 250.000), com a largura do espaço de transição do conceito em duas mil posições (largura: 2.000).

Os métodos utilizados não foram previamente treinados em um conjunto de dados teste e, em vez disso, foram diretamente inseridos nos fluxos de dados de cada *dataset*. Todos os métodos implementados aprenderam continuamente conforme o fluxo de dados de cada *dataset* utilizando a essência da técnica de aprendizado online.

Nome	**Amostras**	**Features**
Fluxos de dados sintéticos		
ConceptDrift	500.000	3
Dataset		
HTTP	567.498	3
CreditCard	100.000	30
SMTP	95.156	3
Elec2	45.312	8
SMSSpam	5.574	1
Bananas	5.300	2
Phishing	1.250	9

Tabela 3: Datasets usados para a avaliação dos modelos.

O dataset HTTP é uma versão do *dataset* usado na competição KDD-1999 com apenas os dados das conexões HTTP. Os labels neste dataset especificam se uma conexão é um ataque a rede ou é uma conexão normal (21). O *dataset* CreditCard apresenta dados de transações de cartões de crédito por titulares europeus em dois dias do mês de setembro de 2013. Esse *dataset* é altamente desbalanceado (uma disparidade entre as classes, contendo mais transações legítimas do que fraudulentas) e o seu objetivo ao verificar essas transações é identificar se é uma transação fraudulenta ou se é uma transação normal (22). Já o *dataset* SMTP é uma versão da competição KDD-1999 com apenas os dados das conexões SMTP, onde o objetivo é prever se a conexão SMTP é um ataque ou dados normais(21).

O *dataset* Elec2 contém os dados do mercado de eletricidade em Nova Gales do Sul, nesse mercado o preço da eletricidade não é fixo, e varia conforme a oferta/demanda. O objetivo aqui é prever se o valor da eletricidade vai subir ou descer (23). O *dataset* SMSSpam contém dados de SMS reais, tendo como objetivo identificar se as mensagens de SMS é um spam ou não (24).

No *dataset* Bananas os dados foram gerados artificialmente, composto por instâncias distribuídas em grupos que exibem uma forma gráfica em formato de uma banana. Cada

instância é caracterizada por dois atributos, que são $At1$ e $At2$, correspondente às coordenadas nos eixos x e y de um gráfico. Já o *dataset* Phishing possui dados de características de Web sites, onde o objetivo é predizer se ele é um site phishing ou não (25).

Os *datasets* presentes na tabela 3 possuem particularidades nos dados e problemas de classificação binária. Desta forma, cada modelo ao ser experimentado para cada um desses *datasets* terá comportamento e desempenho distintos.

4.3 RESULTADOS

Os resultados foram avaliados com base nas métricas definidas para responder às questões do GQM (*vide* tabela 2). Além disso, foi computado a média ("Avg") e o ranqueamento ("Rank") do desempenho final dos modelos em todos os *datasets* experimentados.

O valor do ranqueamento (*Rank*) representa a média das posições de desempenho que os modelos obtiveram da métrica avaliada para cada um dos *datasets*. Os resultados das avaliações dos métodos *ensemble* estão presentes no apêndice 5 e para simplificar a análise, foi feito um recorte das tabelas do apêndice 5. Nessa seleção, foi considerado os três melhores e piores modelos com base em termos do ranqueamento para cada métrica, juntamente com os métodos ensemble propostos. Para facilitar a visualização e interpretação, nas tabelas contidas nesta seção, os melhores desempenhos são destacados em negrito, tanto o melhor modelo base quanto o melhor método *ensemble* proposto. Vale ressaltar que nesta seção os termos modelo *ensemble* e método *ensemble* serão usados de forma intercambiável. **Nas tabelas que representam os resultados computados para as métricas, o valor da métrica é a média do cálculo dos valores do desempenho ao longo de todas as observações**, diferente da estratégia *rolling* que é o cálculo do desempenho apenas para a janela de observações definida.

Algoritmos	**ConceptDrift**	**HTTP**	**CreditCard**	**SMTP**	**Elec2**	**SMSSpam**	**Bananas**	**Phishing**	**Avg Accuracy**	**Avg Rank**
ensembleStackingClassifier	**0.99726**	0.99997	0.99916	0.99982	**0.92377**	**0.95981**	0.88169	0.9056	**0.95838**	**5.625**
forestAMFClassifier	0.98609	**0.99998**	**0.99924**	**0.99985**	0.85848	0.86398	**0.88865**	0.8823	0.93482	11.625
ensembleLeveragingBaggingClassifier	0.9972	0.99996	0.99912	0.9998	0.85258	0.91855	0.53132	0.89600	0.89932	12.875
linearModelSoftmax	0.8608	0.78845	0.99917	0.73206	0.80911	0.63215	0.53557	0.89111	0.78105	24.375
HoeffdingAdaptiveTreeClassifier	0.97944	0.9961	0.99777	0.99968	0.78504	0.86183	0.54029	0.82146	0.8727	25.0
modelLinearALMAClassifier	0.83944	0.85159	0.99292	0.76498	0.87511	0.5696	0.50641	0.8256	0.77821	28.0
ensembleBestModel	**0.99739**	**0.99998**	0.99925	**0.99987**	**0.9241**	**0.97936**	**0.88981**	**0.91593**	**0.96321**	**1.125**
ensembleBestModelAverage	0.997208	0.99997	**0.99926**	0.99978	0.88663	0.89253	0.86566	0.9032	0.94303	7.5
ensembleBestModelThreshold	0.99711	0.99997	0.99922	0.99978	0.85266	0.89343	0.87018	0.8976	0.93874	8.5
ensembleVoting	0.997	0.99997	0.99921	0.99977	0.84041	0.89666	0.62471	0.904	0.90772	9.75

Tabela 4: Recorte dos resultados dos experimentos considerando a métrica *accuracy*.

A tabela 4 representa o valor calculado da métrica *accuracy* considerando os datasets experimentados. É possível verificar que, para todos os *datasets*, o método *ensemble* proposto com a estratégia *best model* supera os modelos base, exceto no *dataset* Credit-

Card, onde o método ensemble com estratégia *best model average* foi superior por uma pequena fração. Considerando o ranqueamento ("Rank"), o método *ensemble* com a estratégia *best model* obteve em média a melhor posição, estando praticamente em primeiro lugar para todos *datasets* (1.125). Em segundo lugar, ficou o modelo base *ensemble Stacking Classifier* e nas posições seguintes ficaram os outros três modelos *ensemble* propostos. Os três modelos base que obtiveram as piores posições no ranqueamento considerando a métrica *accuracy* foram: *Linear ALMA Classifier*, *linear Model Softmax* e *Hoeffding Adaptive Tree Classifier*.

Algoritmos	**ConceptDrift**	**HTTP**	**CreditCard**	**SMTP**	**Elec2**	**SMSSpam**	**Bananas**	**Phishing**	**Avg Precision**	**Avg Rank**
ensembleStackingClassifier	**0.99768**	**0.99954**	0.88826	**1.0**	**0.91514**	0.95796	**0.8878**	**0.88808**	**0.94181**	**5.75**
modelKNNClassifier	0.97424	0.99674	**0.95959**	0.79166	0.80972	**0.9759**	0.87767	0.88	0.90819	11.875
forestAMFClassifier	0.98147	0.99774	0.93023	0.9	0.84757	0.13333	0.89064	0.86496	0.81824	12.375
modellinearALMAClassifier	0.97875	0.0255	0.1887	0.00089	0.85492	0.19743	0.45522	0.77317	0.43432	24.375
RandomUnderSampler	0.02597	0.0	0.81067	0.0	0.60781	0.61229	0.45572	0.86371	0.42202	26.375
treeHoeffdingAdaptiveTreeClassifier	0.98049	0.0	0.0	0.0	0.75515	0.44102	0.48157	0.8253	0.43544	26.75
ensembleBestModelAverage	0.99655	**0.99954**	0.9162	**1.0**	0.87813	**1.0**	0.88697	0.87001	**0.94342**	5.375
ensembleBestModel	**0.997**	0.99864	**0.92528**	0.875	**0.91407**	0.97023	**0.88922**	**0.88636**	0.93197	**4.25**
ensembleBestModelThreshold	0.99636	**0.99954**	0.91428	**1.0**	0.85985	0.93714	0.83598	0.85958	0.92534	9.375
ensembleVoting	0.99613	**0.99954**	0.91379	**1.0**	0.85482	**1.0**	0.66782	0.87412	0.91328	6.75

Tabela 5: Recorte dos resultados dos experimentos considerando a métrica *precision*.

Ao observar a tabela 5 referente a métrica *precision*, o modelo base *ensemble Stacking Classifier* continua liderando o *rank* como melhor classificador base, porém ficou em terceiro lugar na visão geral, perdendo para dois métodos *ensemble* propostos, *ensemble best model* e *ensemble best model average*. O método proposto *ensemble best model* lidera o ranqueamento, pois praticamente ficou em quarto lugar para todos os *datasets*. Em segundo lugar ficou o modelo proposto *ensemble best model average*. Em quarto e quinto lugares ficaram os outros dois modelos *ensemble* propostos.

Algoritmos	**ConceptDrift**	**HTTP**	**CreditCard**	**SMTP**	**Elec2**	**SMSSpam**	**Bananas**	**Phishing**	**Avg MCC**	**Avg Rank**
ensembleStackingClassifier	**0.99406**	0.99704	0.79542	0.65822	**0.84379**	**0.81709**	0.76051	**0.80855**	**0.83434**	**5.75**
forestAMFClassifier	0.96992	**0.9975**	**0.81661**	**0.73478**	0.70927	-0.0001	**0.77457**	0.76097	0.72044	12.0
ensembleLeveragingBaggingClassifier	0.99394	0.99545	0.79466	0.62819	0.6968	0.60586	0.01573	0.78945	0.69001	12.25
modelLinearALMAClassifier	0.70697	0.14701	0.35028	0.01807	0.74422	0.1824	0.01408	0.6524	0.35193	24.125
treeHoeffdingAdaptiveTreeClassifier	0.95535	0.0	0.0	0.0	0.55839	0.17157	0.04443	0.63576	0.29569	26.75
treeHoeffdingTreeClassifier	0.93496	0.0	0.0	0.0	0.4865	0.28458	0.09641	0.60832	0.30135	26.875
ensembleBestModel	**0.99434**	**0.99772**	0.81698	**0.78256**	**0.84451**	**0.90884**	**0.7769**	**0.83077**	**0.86908**	**1.125**
ensembleBestModelAverage	0.99394	0.99658	**0.8205**	0.57728	0.76731	0.41982	0.72873	0.80543	0.7637	6.875
ensembleBestModelThreshold	0.99373	0.99658	0.80956	0.57728	0.69707	0.42443	0.73995	0.79487	0.75419	8.375
ensembleVoting	0.9935	0.99658	0.80681	0.54766	0.671968	0.45222	0.23274	0.80658	0.68851	9.5

Tabela 6: Recorte dos resultados dos experimentos considerando a métrica *MCC*.

Já ao observar a tabela 6 que ilustra o cômputo da métrica *MCC*, o *ensemble best model* obteve o melhor desempenho no ranqueamento e na média da métrica, com o maior desempenho geral em sete dos oito *datasets*. No *dataset* CreditCard o modelo proposto *ensemble best model average* foi o que teve maior desempenho em termo da métrica MCC e

em segundo lugar ficou o *ensemble best model*, por pouca diferença. O melhor modelo base no ranqueamento da métrica *MCC* foi o *ensemble stackingClassifier*, ficando na segunda posição na visão geral do *rank* e na média da métrica MCC. Os outros modelos propostos, como mostra a tabela, ficaram em terceiro, quarto e quinto lugar no ranqueamento.

Algoritmos	ConceptDrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg C. Kappa	Avg Rank
ensembleStackingClassifier	**0.99406**	0.99704	0.79062	0.60457	**0.84375**	**0.80771**	0.75969	**0.80852**	**0.82574**	**6.0**
ensembleLeveragingBaggingClassifier	0.99394	0.99545	0.79395	0.61215	0.69529	0.58912	0.0145	0.7893	0.68546	11.5
forestAMFClassifier	0.96969	**0.9975**	**0.80975**	**0.71992**	0.70881	-2.79341	**0.77411**	0.76097	0.71759	12.0
modelLinearALMAClassifier	0.67898	0.04246	0.29095	0.00116	0.74421	0.12614	0.01398	0.64975	0.31845	24.125
treeHoeffdingAdaptiveTreeClassifier	0.95531	0.0	0.0	0.0	0.55816	0.13456	0.04247	0.6337	0.29052	27.375
treeHoeffdingTreeClassifier	0.93429	0.0	0.0	0.0	0.48365	0.14984	0.03741	0.60006	0.27565	27.75
ensembleBestModel	**0.99434**	**0.99772**	0.81071	**0.77771**	**0.84449**	**0.90717**	**0.77658**	**0.83009**	**0.86735**	**1.125**
ensembleBestModelAverage	0.99394	0.99658	**0.81555**	0.49992	0.76703	0.29968	0.72597	0.80455	0.7379	7.0
ensembleBestModelThreshold	0.99373	0.99658	0.80363	0.49992	0.69475	0.32121	0.73897	0.79353	0.73029	8.625
ensembleVoting	0.99349	0.99658	0.80061	0.46146	0.66815	0.33957	0.20287	0.805972	0.65859	9.625

Tabela 7: Recorte dos resultados dos experimentos considerando a métrica Cohen Kappa.

Na tabela 7 (Cohen Kappa), é verificado que o modelo proposto *ensemble best model* foi o melhor no ranqueamento e na média da métrica Cohen Kappa. Em comparação aos demais modelos, tal método *ensemble* obteve desempenho nessa métrica superior em sete dos oito *datasets*. Em segundo lugar no ranqueamento, vem o modelo base *ensemble Stacking Classifier*. Em terceiro lugar ficou o modelo proposto *ensemble best model average*, que no *dataset* CreditCard teve um desempenho superior nessa métrica, superando por uma pequena fração o *ensemble best model* e *ensemble Stacking Classifier*. Em seguida no *rank*, em quarto lugar ficou o modelo proposto *ensemble best model threshold* e em quinto *ensemble voting*.

Algoritmos	ConceptDrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg B. Accuracy	Avg Rank
ensembleStackingClassifier	**0.99696**	0.99728	0.8564	0.71666	**0.92121**	0.86364	**0.87803**	**0.90474**	**0.89187**	**8.5**
ensembleLeveragingBaggingClassifier	0.99681	0.99683	**0.88099**	**0.74998**	0.84422	0.75156	0.50697	0.8956	0.82787	10.875
Perceptron	0.99187	**0.99747**	0.87177	0.74979	0.9007	**0.86386**	0.51193	0.85631	0.84296	12.75
treeExtremelyFastDecisionTreeClassifier	0.97926	0.5	0.5	0.5	0.80884	0.5501	0.60134	0.8823	0.66523	24.0
treeHoeffdingAdaptiveTreeClassifier	0.97635	0.5	0.5	0.5	0.77788	0.54627	0.52061	0.81371	0.64185	26.25
treeHoeffdingTreeClassifier	0.96226	0.5	0.5	0.5	0.73794	0.54618	0.51708	0.79403	0.63218	27.75
ensembleBestModel	**0.9968**	**0.99841**	0.86092	**0.84998**	**0.92178**	**0.93434**	**0.88716**	**0.91714**	**0.92081**	**3.125**
ensembleBestModelAverage	0.99648	0.99683	**0.86763**	0.66666	0.88196	0.59906	0.85975	0.90461	0.84662	11.0
ensembleBestModelThreshold	0.99633	0.99683	0.85866	0.66666	0.84315	0.60863	0.87146	0.89962	0.84267	12.625
ensembleVoting	0.99616	0.99683	0.85642	0.65	0.82879	0.61445	0.59654	0.90492	0.80551	14.125

Tabela 8: Recorte dos resultados dos experimentos considerando a métrica *balanced accuracy*.

A tabela 8 apresenta os resultados dos experimentos considerando a métrica *Balanced Accuracy*. É possível observar que o melhor modelo no ranqueamento foi o modelo proposto *ensemble best model*, tendo um valor de *rank* 3.125, significando que ele obteve em média o melhor desempenho para todos os *dataset*. Ele foi o modelo com melhor desempenho em seis dos oito *datasets*. Nos demais dois *datasets*, o método *ensemble best model* obteve desempenho muito próximo do melhor modelo. Em segundo lugar do

rank ficou o classificador base *ensemble Stacking Classifier*, tendo o melhor desempenho em apenas um dos oito *datasets*. O modelo proposto *ensemble best model Average* ficou na quarta posição em relação ao *rank*, o *ensemble best model threshold* em quinto, e o *ensemble Voting* não teve uma posição significativa, ficando em nono lugar.

Algoritmos	**ConceptDrift**	**HTTP**	**CreditCard**	**SMTP**	**Elec2**	**SMSSpam**	**Bananas**	**Phishing**	**Avg Recall**	**Avg Rank**
RandomOverSampler	0.99548	0.9421	**0.73991**	0.43333	0.7016	**0.87282**	**0.94318**	**0.93235**	**0.82009**	**11.75**
linearModelSoftmaxRegression	0.78213	**0.99729**	0.73543	**0.5**	0.7185	0.86881	0.54588	0.92505	0.75913	**11.75**
EnsembleStackingClassifier	**0.99803**	0.99457	0.713	0.43333	**0.90429**	0.73226	0.84259	0.89781	0.81448	12.125
TreeExtremelyFastDecisionTreeClassifier	0.98729	0.0	0.0	0.0	0.73435	0.1004	0.36279	0.86288	0.38096	25.375
TreeHoeffdingAdaptiveTreeClassifier	0.98745	0.0	0.0	0.0	0.7305	0.11512	0.32996	0.75137	0.3643	26.375
TreeHoeffdingTreeClassifier	0.99011	0.0	0.0	0.0	0.64644	0.09236	0.05092	0.68921	0.30863	30.25
ensembleBestModel	0.99892	**0.99683**	0.72197	**0.7**	**0.90642**	**0.87282**	0.86153	**0.92687**	**0.87317**	**5.25**
ensembleBestModelThreshold	0.99912	0.99366	0.71748	0.33333	0.78011	0.21954	**0.88383**	0.91605	0.73039	11.875
ensembleBestModelAverage	0.99908	0.99366	**0.73542**	0.33333	0.85106	0.19812	0.8026	0.91605	0.72867	11.125
ensembleVoting	**0.99918**	0.99366	0.71300	0.3	0.75178	0.22891	0.32407	0.9124	0.65287	15.5

Tabela 9: Recorte dos resultados dos experimentos considerando a métrica *recall*.

A tabela 9 traz os resultados dos experimentos considerando a métrica *Recall*, que é a média do valor do cálculo dessa métrica ao longo das observações em cada *dataset*. Observa-se que o modelo proposto, *ensemble best model*, ocupa a posição de liderança no *rank*. Em relação à métrica de *Recall*, este modelo foi o melhor em dois dos oito *datasets*. Nos demais *datasets*, tal modelo teve um desempenho próximo o melhor desempenho observado. No contexto do *dataset* ConceptDrift, todos os métodos ensemble propostos apresentaram um desempenho superior em comparação aos modelos base. Especificamente, o modelo *ensemble voting* foi identificado como o mais eficaz nesse *dataset*. Os modelos base *Random Over Sampler* e *linear Model Softmax Regression* ficaram empatados no *rank*, ocupando o terceiro lugar. Na segunda posição do *rank* ficou o método proposto *ensemble best model average* e em quarto o método proposto *ensemble best model threshold*. O *ensemble voting* ocupou a décima quarta posição no *rank*.

Algoritmos	**ConceptDrift**	**HTTP**	**CreditCard**	**SMTP**	**Elec2**	**SMSSpam**	**Bananas**	**Phishing**	**Avg ROC Curve**	**Avg Rank**
Ensemble StackingClassifier	**0.99696**	0.99728	0.8564	0.71666	**0.92121**	0.86364	**0.87803**	**0.90474**	**0.89187**	**8.5**
ensembleLeveragingBaggingClassifier	0.99681	0.99683	**0.88099**	**0.74998**	0.84422	0.75156	0.50697	0.8956	0.82787	10.87500
Perceptron	0.99187	**0.99747**	0.87177	0.74979	0.9007	**0.86386**	0.51193	0.85631	0.84296	12.75
treeExtremelyFastDecisionTreeClassifier	0.97926	0.5	0.5	0.5	0.80884	0.5501	0.60134	0.8823	0.66523	24.0
TreeHoeffdingAdaptiveTreeClassifier	0.97635	0.5	0.5	0.5	0.77788	0.54627	0.52061	0.81371	0.64185	26.25
TreeHoeffdingTreeClassifier	0.96226	0.5	0.5	0.5	0.73794	0.54618	0.51708	0.79403	0.63218	27.75
ensembleBestModel	**0.9968**	**0.99841**	0.86092	**0.84998**	**0.92178**	**0.93434**	**0.88716**	**0.91714**	**0.92081**	**3.125**
ensembleBestModelAverage	0.99648	0.99683	**0.86763**	0.66666	0.88196	0.59906	0.85975	0.90461	0.84662	11.0
ensembleBestModelThreshold	0.99633	0.99683	0.85866	0.66666	0.84315	0.60863	0.87146	0.89962	0.84267	12.625
ensembleVoting	0.99616	0.99683	0.85642	0.65	0.82879	0.61445	0.59654	0.90492	0.80551	14.12500

Tabela 10: Recorte dos resultados dos experimentos considerando a métrica *ROC Curve*.

A tabela 10 demonstra os resultados obtidos dos experimentos considerando a métrica ROC Curve. Observa-se que o método proposto *ensemble best model* liderou o ranqueamento, bem como a média da métrica *ROC Curve*. Tal método obteve melhor

desempenho em seis dos oito *datasets*. Nos outros dois *datasets*, tal modelo obteve desempenho próximo ao melhor desempenho observado. Em segundo lugar do *rank* ficou o modelo base *ensemble StackingClassifier*. O modelo proposto *ensemble best model Average* ocupou a quarta posição no *rank* dessa métrica, e em termos da média da métrica ele ocupou a terceira posição. Em quinto lugar ficou *ensemble best model threshold* e o *ensemble voting* em nono lugar. Nessa métrica o modelo proposto *ensemble voting* supera o modelo base *ensemble Voting Classifier*.

Algoritmos	**ConceptDrift**	**HTTP**	**CreditCard**	**SMTP**	**Elec2**	**SMSSpam**	**Bananas**	**Phishing**	**Média**	**Rank**
ensembleStackingClassifier	0.02042	0.011095	0.08119	0.01574	0.02462	0.60111	0.01856	0.02328	0.0995	5.0
modelKNNClassifier	0.0029	0.00277	0.01207	0.00364	0.00366	0.04988	0.00318	0.00492	0.01038	6.5
ARFClassifier	0.00236	0.00122	0.002	0.00166	0.002	0.05558	0.00216	0.00173	0.0086	8.25
modellinearALMAClassifier	0.00012	0.0001	0.00028	0.0001	0.00013	0.00075	0.00013	0.00012	0.00021	31.625
Perceptron	0.00014	0.00012	0.00024	0.00012	0.00014	0.00144	0.00013	0.00014	0.00031	30.62500
linearmodelPAClassifier	0.00013	0.00011	0.00027	0.00011	0.00014	0.0004	0.00012	0.00013	0.00018	31.25
ensembleBestModelThreshold	0.08727	0.05869	0.21019	0.08099	0.13725	1.62222	0.07689	0.09606	0.2962	2.37500
ensembleBestModelAverage	0.08983	0.06495	0.20097	0.079	0.13626	1.62524	0.072	0.09842	0.29583	2.0
ensembleBestModel	0.08626	0.06228	0.1964	0.07822	0.09724	0.98327	0.08793	0.08916	0.21009	3.37500
ensembleVoting	0.08919	0.06428	0.2179	0.08325	0.10363	0.9216	0.08431	0.09623	0.20755	2.25

Tabela 11: Recorte dos resultados dos experimentos considerando o tempo de treinamento e predição.

A tabela 11 apresenta o tempo médio demandado por cada modelo base e método ensemble para o treinamento e predição de cada observação. Nesta tabela, o *rank* dos registros é organizado do maior para o menor. Desta forma, o primeiro lugar é ocupado pelo modelo que registrou o maior tempo de execução (indicando um desempenho mais lento para treinamento e predição). É importante ressaltar que os métodos *ensemble* propostos foram executados de forma sequencial, o que resultou em um desempenho em relação ao tempo considerado insatisfatório. O pior modelo base foi o *ensemble Stacking Classifier* devido a sua complexidade computacional, uma vez que o processo de treinamento envolve a criação de múltiplos conjuntos de dados intermediários e a execução de vários modelos base.

Para a métrica *rolling ROC Curve* não é interessante fazer análise por meio de tabelas, pois essa métrica é calculada mediante uma janela deslizante (definida com tamanho 100). Diferentemente das outras métricas avaliadas, que acumulam o desempenho ao longo de cada observação nos conjuntos de dados executados, a Curva ROC móvel não é um valor cumulativo. Portanto, neste texto, optamos por uma análise gráfica dessa métrica, que será apresentada na seção 4.4.4.

4.4 ANÁLISES

As análises dos resultados aborda cinco aspectos importantes de métodos de aprendizado de máquina baseados em ensemble: i) a seleção de múltiplos modelos base para

predição ensemble; ii) a avaliação do desempenho do ensemble em comparação com os modelos base individuais; iii) a habilidade do método ensemble em distinguir as classes binárias em diferentes pontos de corte; iv) a análise do comportamento desses métodos ensemble diante de mudanças nos conceitos (*concept drift*) subjacentes aos dados; e v) o tempo requerido para predição e aprendizado.

4.4.1 SELEÇÃO DE MÚLTIPLOS MODELOS BASE PARA PREDIÇÃO ENSEMBLE

A figura 9 ilustra o comportamento do método *ensemble best model* no decorrer da experimentação com o fluxo de dados HTTP. Vale ressaltar que neste método *ensemble*, a estratégia de combinação utiliza apenas o melhor modelo para cada predição ensemble. Isto é, o método *ensemble* considera no instante de tempo t o modelo base que apresenta o maior valor da métrica *accuracy*. Esta dinâmica na escolha justifica a variação de modelos escolhidos para predição *ensemble*.

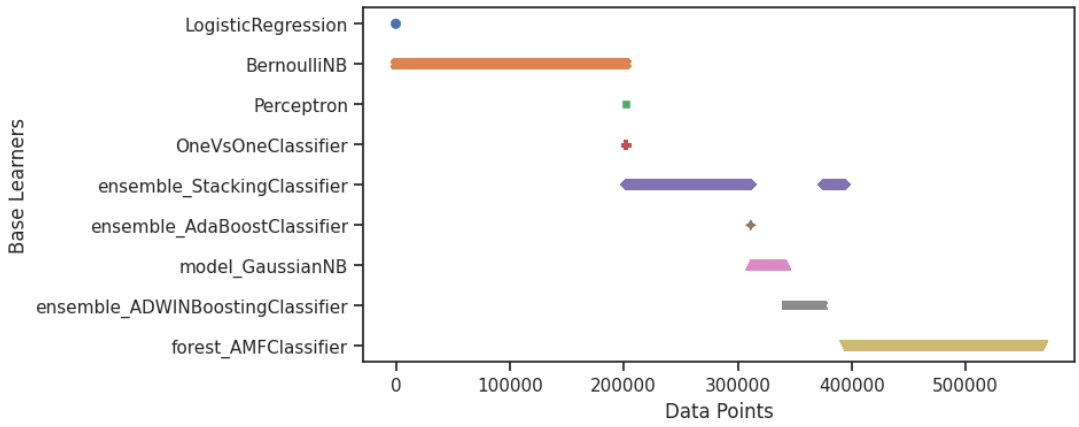

Figura 9: Modelos selecionados pelo método *ensemble best model* para o processamento do *dataset* HTTP.

Conforme ilustrado na figura 9, inicialmente, o método *ensemble* seleciona o modelo base *Logistic Regression*, chegando a ter uma pequena participação de 0.0001%. Logo em seguida, o modelo base *Bernoulli NB* é selecionado como o melhor modelo, tendo uma participação de 35.5363%. Em seguida, é selecionado o modelo base *Perceptron* (participação de 0.0001%), seguido por *One Vs One Classifier* (participação de 0.007%). O modelo base *ensemble Stacking Classifier* (participação total de 22.7093% é selecionado em dois momentos no fluxo de dados. Posteriormente, vem a escolha do *ensemble Ada Boost Classifier* (participação de 0.0003%), *model Gaussian NB* (participação de 5.4068%), e *ensemble ADWIN Boosting Classifier* (participação de 5.8049%). Logo em seguida, o *ensemble Stacking Classifier* é selecionado novamente, e por fim, *forest AMF Classifier* (participação de 30.5343%) é selecionado como o melhor modelo até o fim do processa-

mento do fluxo dos dados.

Através da seleção dos modelos base do *ensemble best model*, pode-se analisar a importância de múltiplos modelos base heterogêneos. Em cada momento do fluxo de dados, um modelo apresentou desempenho superior aos outros, evidenciando a complementaridade entre eles. Não é porque alguns modelos não figuraram entre os três melhores considerando o cálculo acumulado das métricas no *rank*, que em algum momento que não obtiveram desempenho superior aos demais. Nota-se que, entre os três melhores modelos no *rank* da métrica *accuracy*, apenas dois foram selecionados no *ensemble best model*, contribuindo para a predição dele: o *ensemble Stacking Classifier*, no meio do fluxo de dados, e o *forest AMF Classifier*, no final. Essa diversidade de modelos contribuiu para a robustez e eficácia do *ensemble best model* ao longo do processo, garantindo que o *ensemble best model* obtivesse o melhor desempenho na métrica *accuracy* em termos de precisão.

4.4.2 ANÁLISE COMPARATIVA DO DESEMPENHO ENTRE O MÉTODO ENSEMBLE E OS MODELOS BASE

Nessa seção, inicialmente, serão apresentados os modelos base selecionados pelo *ensemble best model* para um dos oito *datasets*. Em seguida, serão realizadas as seguintes análises: i) análise individual dos modelos selecionados pelo método *ensemble best model* para predição ensemble; ii) análise do próprio método *ensemble best model*; por fim a análise dos melhores modelos base conforme o ranqueamento da métrica escolhida para avaliação do desempenho no *dataset* em questão.

A figura 10 demonstra os modelos base selecionados pelo método *ensemble best model* para as predições *ensemble* durante o processamento do *dataset* CreditCard. Considerando os resultados da tabela 4, percebe-se que o método *ensemble best model* foi superior a qualquer modelo base, em termos da média do cálculo do desempenho da métrica acurácia (*accuracy*). No fluxo de dados CreditCard, nas observações finais, o modelo base *forest AMF Classifier* foi eleito como melhor modelo base através da métrica *accuracy* na seleção dos modelos base do *ensemble best model*, com uma participação de 89.09% na predição.

Entretanto, ao analisar os resultados obtidos dos experimentos considerando a métrica *precision* (tabela 5), no processamento do *dataset* CreditCard, o método *ensemble best model* ficou em terceiro lugar ao considerarmos o cálculo cumulativo da métrica, perdendo para os modelos base *KNN Classifier* e *forest AMF Classifier* por uma pequena fração no desempenho avaliado. Ressalta-se que a métrica *precision* avalia a quantidade de acertos da classe verdadeira positiva (VP). Desta forma, observa-se que o método *ensemble best model* não superou o poder de classificação de rótulos verdadeiro positivos dos modelos base *KNN Classifier* e *forest AMF Classifier*. Porém, ao usar um conjunto

heterogêneo de modelos base, o método *ensemble best model* foi capaz de mitigar sua deficiência na classificação de rótulos verdadeiros positivos.

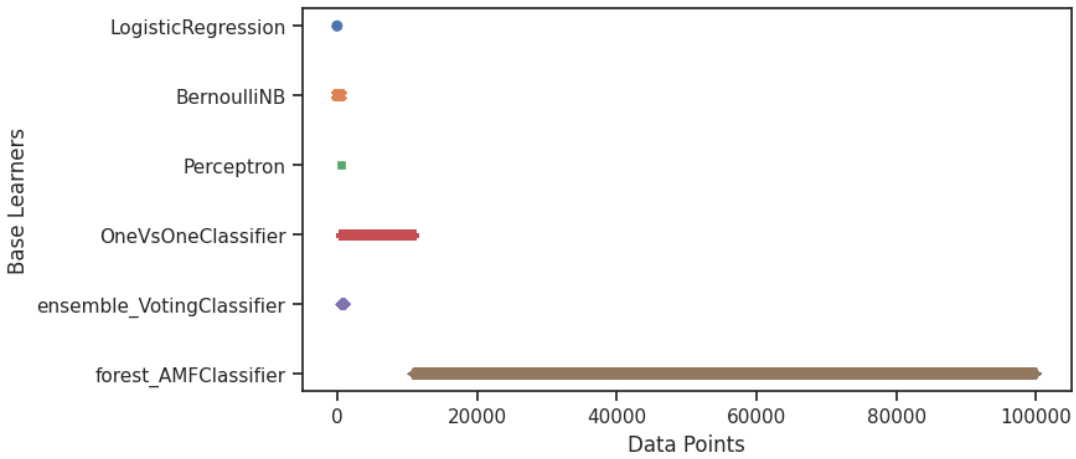

Figura 10: Modelos base selecionados pelo método *ensemble best model* para o processamento do *dataset* CreditCard.

Na Figura 11, é evidente que os modelos base *Perceptron* e *One Vs One classifier*, selecionados pelo método *ensemble best model* durante o processamento do dataset CreditCard, demonstraram desempenho superior em termos da métrica *precision* durante as primeiras observações do dataset. Tais modelos, inicialmente, superaram o desempenho do próprio método *ensemble best model* e os melhores modelos base individuais no ranqueamento (vide tabela 5), como *KNN Classifier* e *Forest AMF Classifier*. O modelo base *Forest AMF Classifier*, exibiu desempenho superior ao do método *ensemble best model*, desempenhando um papel significativo nas predições ensemble. Além disso, conforme destacado na tabela 4, observa-se que o modelo base *Forest AMF Classifier* foi classificado como o segundo melhor modelo base em termos da métrica *accuracy*, além de alcançar excelentes resultados na métrica *precision*.

Considerando o *dataset* SMSSpam (vide seção 4.3), observa-se que os valores computados das métricas apresentam variação significativa na maioria dos modelos base. Isto evidencia que alguns modelos base não foram capazes de generalizar a relação entre as entradas (variáveis de entrada) com os rótulos esperados (*underfitting*). Neste *dataset*, o método *ensemble best model* foi superior a maioria dos modelos base e conseguiu minimizar a presença de underfitting dos modelos base.

A figura 12 ilustra o resultado da seleção dos modelos base para predição *ensemble* do método *best model* durante o processamento do *dataset* SMSSpam. Observa-se que cinco modelos foram selecionados, com participação majoritária do modelo base *Random Over Sampler* (94.52%). Conforme os resultados dos experimentos com este *dataset* (vide tabela 7), o método *ensemble best model* apresentou o melhor desempenho em relação à métrica Cohen Kappa. Apesar que o modelo base *Random Over Sampler* contribuiu

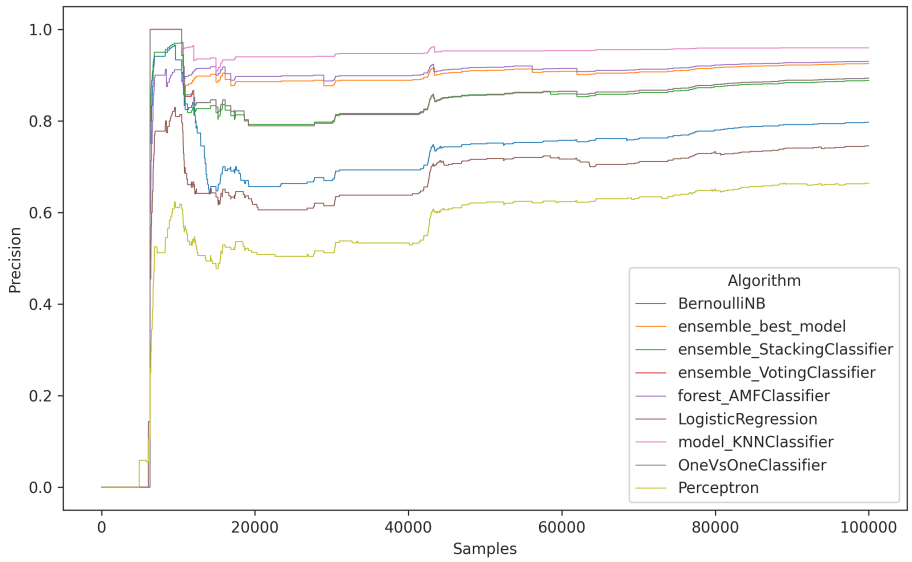

Figura 11: Desempenho dos modelos base e método *ensemble* em termos da métrica *precision* ao longo do processamento do *dataset* CreditCard.

significativamente para a predição *ensemble*, esse modelo base não configurou entre os melhores modelos base em relação ao ranqueamento da métrica Cohen Kappa.

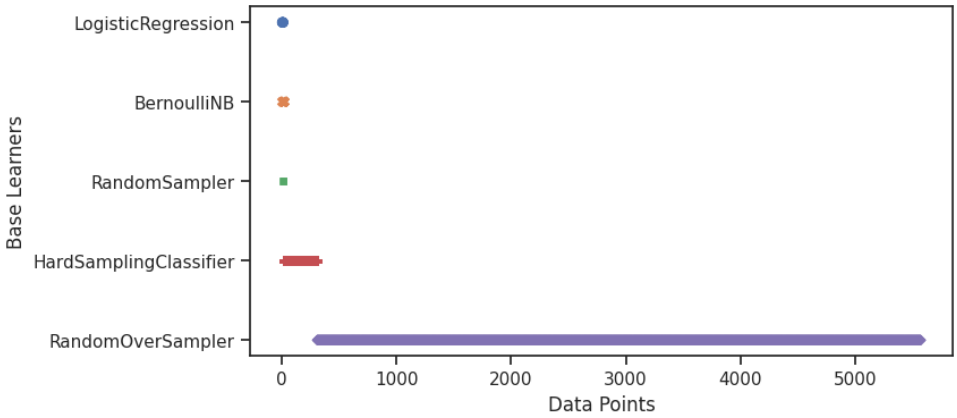

Figura 12: Modelos base selecionados pelo método *ensemble best model* durante o processamento do *dataset* SMSSpam.

A figura 13 ilustra a média do cálculo da métrica Cohen Kappa ao longo do processamento do *dataset* SMSSpam. São apresentados os três melhores modelos base considerando o ranqueamento da métrica, os modelos base selecionados pelo método *ensemble best model* para predição, e o próprio método *ensemble*. Conforme ilustra a figura 13, quase todos os modelos selecionados pelo método *ensemble best model* apresentam desempenho superior em relação aos melhores modelos base no *rank* da métrica Cohen Kappa.

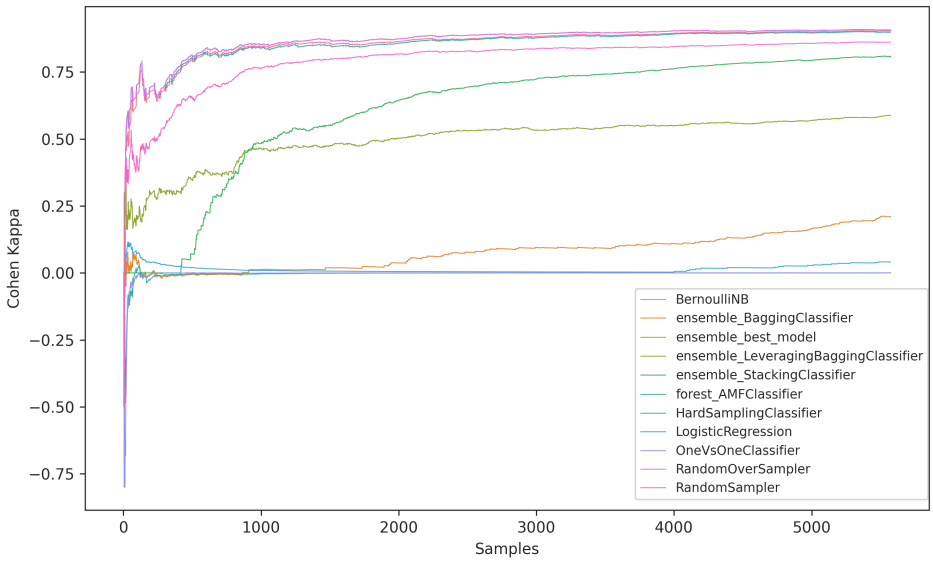

Figura 13: Desempenho dos modelos base e método *ensemble* mensurado pela métrica Cohen Kappa ao longo do processamento do *dataset* SMSSpam.

O método proposto *ensemble best model* obteve o melhor desempenho nesse *dataset*, ficou praticamente "empatado", com uma pouca diferença de fração no desempenho com os modelos *Hard Sampling Classifier*, *Bernoulli NB* e *Random Over Sampler*. Analisando os outros modelos *ensemble* propostos no contexto do *dataset* SMSSpam através tabela 7, é evidente o desempenho em relação à métrica Cohen Kappa não foi significativo.

No cômputo das métricas *accuracy, precision, MCC, Cohen Kappa e balanced accuracy*, o modelo base *ensemble Stacking Classifier* ocupava consistentemente o segundo lugar no *rank*. No entanto, em relação à métrica *recall*, tal modelo base configurou na sexta posição. Essa variação na posição do *rank* na métrica *recall* do *ensemble Stacking Classifier*, mostra que provavelmente ele não acertou com tanta precisão as classes verdadeiras positivas avaliado na métrica *recall*.

No entanto, na métrica *accuracy* o *ensemble Stacking Classifier* esteve em segundo lugar, essa métrica computa tanto os acertos da classe verdadeiras negativas quanto das positivas. Já que esse modelo teve uma variação significativa na métrica *recall*, é possível inferir que ele foi mais eficaz na classificação das classes verdadeiras negativas do que nas classes verdadeiras positivas. Essa análise indica que o desempenho de um modelo pode variar dependendo da métrica escolhida para avaliação e ressalta a importância de considerar múltiplas métricas ao avaliar o desempenho de um modelo.

4.4.3 SENSIBILIDADE DO MÉTODO ENSEMBLE EM DISTINGUIR AS CLASSES BINÁRIAS

A figura 14 ilustra os modelos base selecionados pelo método *ensemble best model* para as predições *ensemble* durante o processamento do *dataset* Elec2. Conforme os experimentos com este *dataset* (vide tabela 10) para a métrica *ROC Curve* acumulativa, o método *ensemble best model* foi superior a qualquer modelo base através do ranqueamento da métrica avaliada. Observa-se que nove modelos base foram selecionados. Desses noves modelos selecionados pelo método *ensemble best model* apenas dois modelos foram os melhores conforme o ranqueamento da métrica *ROC Curve*, foram eles: *ensemble StackingClassifier* (participação de 74,89%) e *Perceptron* (participação de 3,91%).

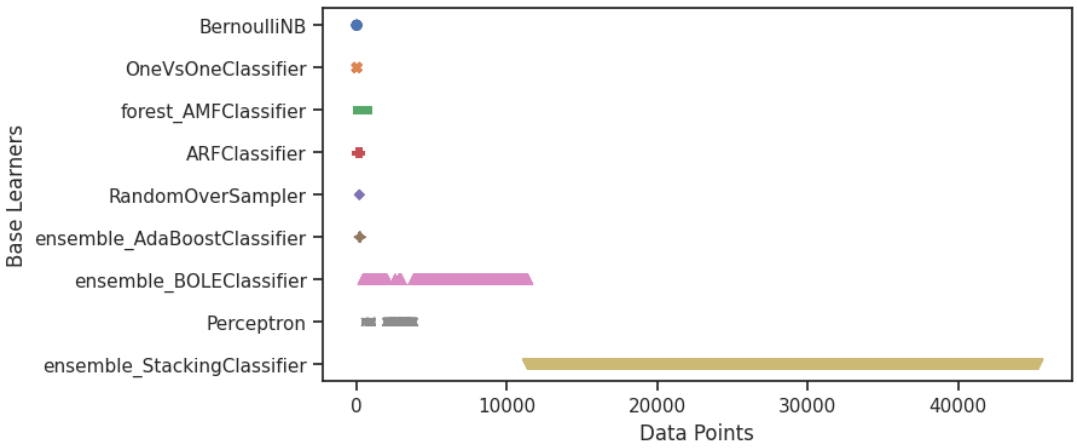

Figura 14: Modelos base selecionados pelo método *ensemble best model* durante as predições *ensemble* para o *dataset* Elec2.

A figura 15 e figura 16 a média do cálculo da métrica *ROC Curve* ao longo do processamento do *dataset* Elec2. São apresentados os três melhores modelos base considerando o ranqueamento da métrica, os modelos base selecionados pelo método *ensemble best model* para predição, e o próprio método *ensemble*.

Conforme ilustra a figura 15 e figura 16, no início das observações até aproximadamente 10000 observações o método proposto *ensemble best model* lidera com o melhor desempenho. Após as 10000 observações praticamente ficaram "empatados"o *ensemble best model*, os modelos base *ensemble BOLE Classifier*, *Perceptron* e *ensemble Stacking Classifier*. Após 15.000 observações, o desempenho do *Perceptron* começa a declinar nesta métrica avaliada, estabilizando-se em torno de 0.9 até o final das observações. Os modelos base *ensemble BOLE Classifier* e *ensemble Stacking Classifier* continuam praticamente "empatados"no desempenho dessa métrica avaliada até o final das observações com o método *ensemble best model*. O *ensemble best model* mostra-se superior por pouca diferença em comparação aos modelos base "empatados".

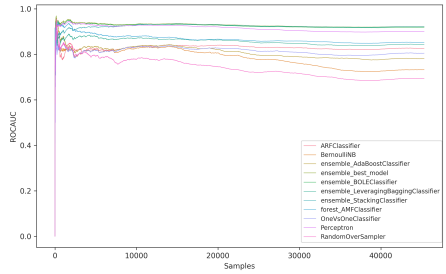

Figura 15: Desempenho dos modelos base e método *ensemble* em termos da métrica *ROC Curve* ao longo do processamento do *dataset* Elec2.

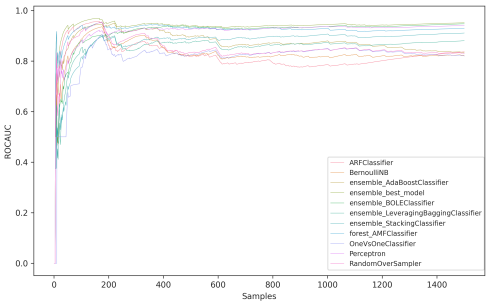

Figura 16: Desempenho dos modelos base e método *ensemble* em termos da métrica *ROC Curve* no início do processamento do *dataset* Elec2.

4.4.4 Desvio de Conceito

Na avaliação do desempenho dos métodos *ensemble* proposto nos fluxos de dados que contém desvio de conceito será utilizado a métrica *rolling ROC Curve* para o *dataset* ConceptDrift.

Na figura 17 ilustra o desempenho atráves da métrica *rolling ROC Curve* do método de *ensemble best model* ao longo do processamento do *dataset* ConceptDrift, observa-se que o método *ensemble best model* manteve acima de 0.95 na maior parte das observações. Contudo, ao examinar o momento exato do desvio de conceito, conforme ilustrado na figura 18, nota-se a ocorrência de uma deterioração do desempenho, com picos chegando até 0.8. Não obstante desses picos, o desempenho do método *ensemble* manteve-se elevada, com valores consistentemente superiores a 0.9 ao longo da maior parte das observações no momento do desvio de conceito.

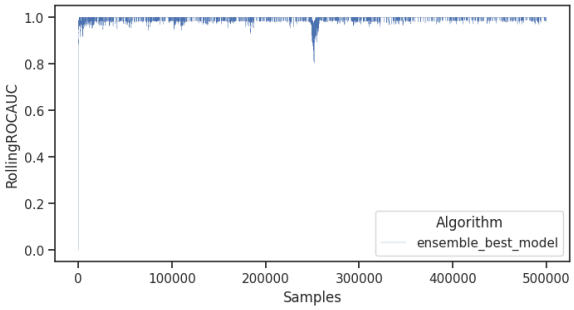

Figura 17: Desempenho do *ensemble best model* no processamento ao longo do *dataset* ConceptDrift.

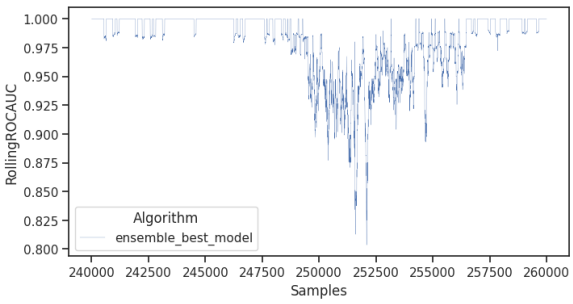

Figura 18: Desempenho do *ensemble best model* no processamento ao longo do momento de desvio de conceito no *dataset* ConceptDrift.

Nas métricas avaliadas anteriormente na seção 4.3, apenas o método *ensemble best model* se destacou, classificando-se como o melhor método no ranqueamento, enquanto

os outros métodos *ensemble* propostos mantiveram-se mais modestos no ranqueamento. No caso da métrica *rolling ROC Curve*, tornou-se impraticável avaliar todos os métodos *ensemble* propostos por meio de ranqueamanto, exigindo, portanto, uma análise gráfica para entender o comportamento do desempenho dos outros métodos *ensemble* propostos, conforme ilustrado na Figura 19. Observa-se que os demais métodos *ensemble* propostos apresentaram desempenho praticamente equivalente. Em algumas observações no *dataset* ConceptDrift, os métodos *ensemble best model threshold* e *ensemble voting* mostraram desempenho inferior em relação aos demais métodos *ensemble*.

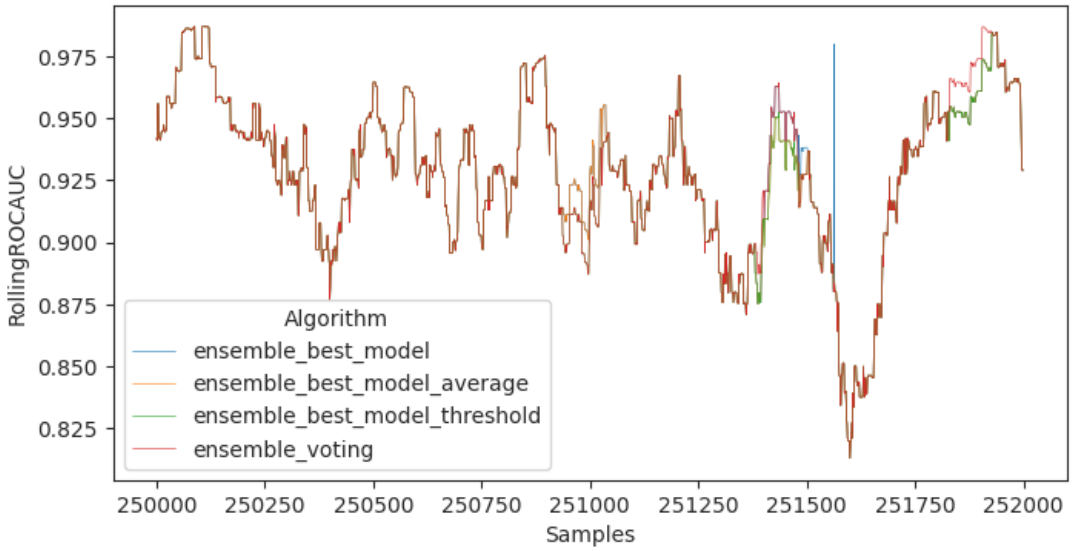

Figura 19: Comportamento dos métodos *ensemble* propostos no processamento do *dataset* ConceptDrift no momento do desvio de conceito.

Na figura 20 ilustra os modelos base selecionados pelo método *ensemble best model* para as predições durante o processamento do *dataset* ConceptDrift, lembrando que a seleção do melhor modelo base foi com base na métrica *accuracy* acumulativa. Observa-se que seis modelos base foram selecionados, com uma participação majoritária do modelo base *One Vs One Classifier* (85,78%), no segundo lugar em termos de participação ficou *ensemble Leveraging Bagging Classifier* (14.20%).

A figura 21 ilustra o valor da métrica *rolling ROC Curve* para os modelos base selecionados pelo método *ensemble best modelo* e o próprio *ensemble* ao longo do processamento do *dataset* ConceptDrift. Os modelos base *HardSamplingClassifier* e *RandomSampler* na maior parte das observações nesse *dataset* obtiveram desempenhos abaixo de 0.9. Os demais modelos base exibem um comportamento semelhante para essa métrica em questão, comparável ao do método *ensemble best model*.

Sendo necessário fazer uma análise dos modelos base que obtiveram um comportamento semelhante ao *ensemble best model* no momento do desvio de conceito, assim tendo uma avaliação com maior detalhes. A figura 22 ilustra o desempenho dos modelos base

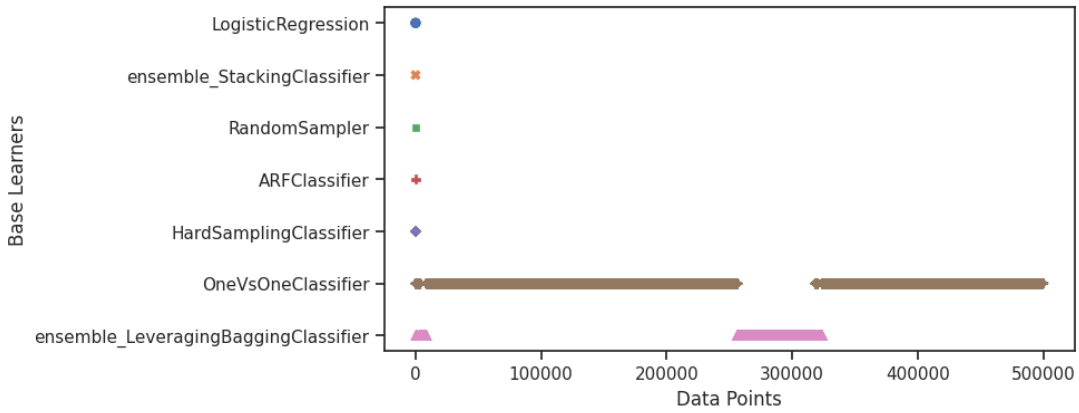

Figura 20: Modelos base selecionados pelo *ensemble best model* no *dataset* ConceptDrift

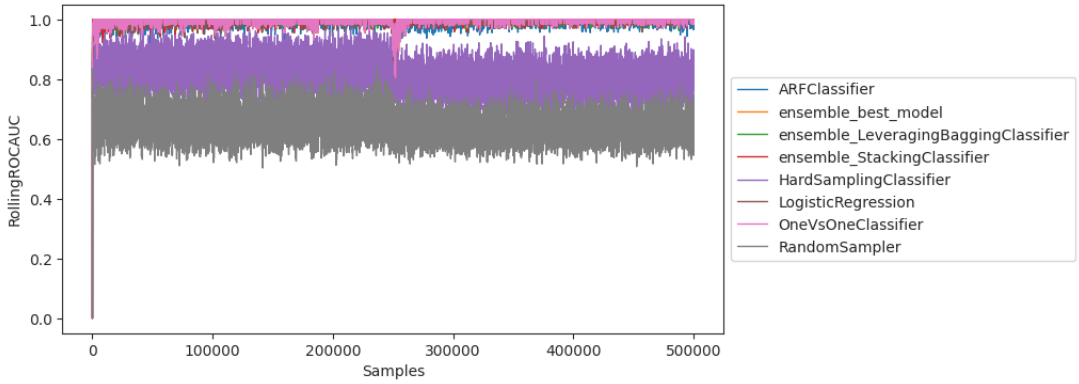

Figura 21: Desempenho dos modelos base selecionados pelo método *ensemble best model* e do próprio *ensemble* durante o processamento do *dataset* ConceptDrift.

e o prórprio *ensemble best model* no processamento das observações do fluxo de dados no desvio de conceito.

Observa-se que na figura 22, que aproximadamente nas observações 250.300 o método *ensemble best model* esteve entre os melhores modelos, todos empatados praticamente, a partir dessas observações o *ensemble best model* ficou entre os três melhores na maior parte das observações no momento do desvio de conceito. Em apenas aproximadamente 500 observações no momento do desvio de conceito o *ensemble best model* apresentou um desempenho inferior aos outros modelos base avaliados. Os modelos base *ensemble StackingClassifier*, *ensemble LeveragingBaggingClassifier* e *LogisticRegression* demonstraram bom desempenho para a maioria das observações durante o desvio de conceito.

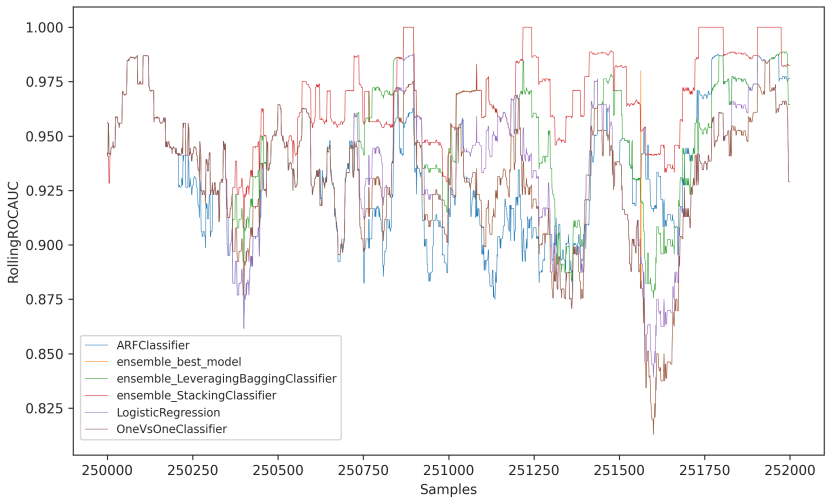

Figura 22: Recorte dos melhores modelo base selecionados pelo método *ensemble best model* para o *dataset* ConceptDrift.

4.4.5 Tempo

A explicação para o desempenho inferior dos métodos *ensemble* propostos reside na estratégia de combinação e no número de modelos base envolvidos (um total de 29 modelos base). Isso se deve ao fato de que cada modelo base requer treinamento individual e contribui para as predições do *ensemble*, resultando em um aumento significativo no tempo de execução e na complexidade computacional.

A figura 23 ilustra o tempo médio demandado pelos três melhores e piores modelos base, além dos quatro métodos *ensemble* propostos em todos os *datasets*. Comparando os métodos propostos com o pior modelo base *ensemble stacking Classifier* em relação ao tempo, os métodos *ensemble* propostos *best model average* e *best model threshold* demoraram aproximadamente três vezes mais, enquanto os métodos *ensemble best model* e *voting* demoraram em média aproximadamente duas vezes mais.

Por exemplo, para o *dataset* SMSSpam, onde os métodos propostos foram mais lentos, constatou-se um valor médio de 1.62 segundos para o processo de aprendizado e predição de cada observação.

4.5 RESPONDENDO ÀS QUESTÕES DE PESQUISA

Para responder à questão **Q1** do GQM (*Os métodos ensembles propostos superam os modelos individuais em termos de erro de predição?*), consideramos os resultados dos experimentos, especificamente os valores computadores para as métricas *accuracy*, *precision*, *ROC Curve* e *Rolling ROC Curve*. Considerando os valores obtidos para as métricas

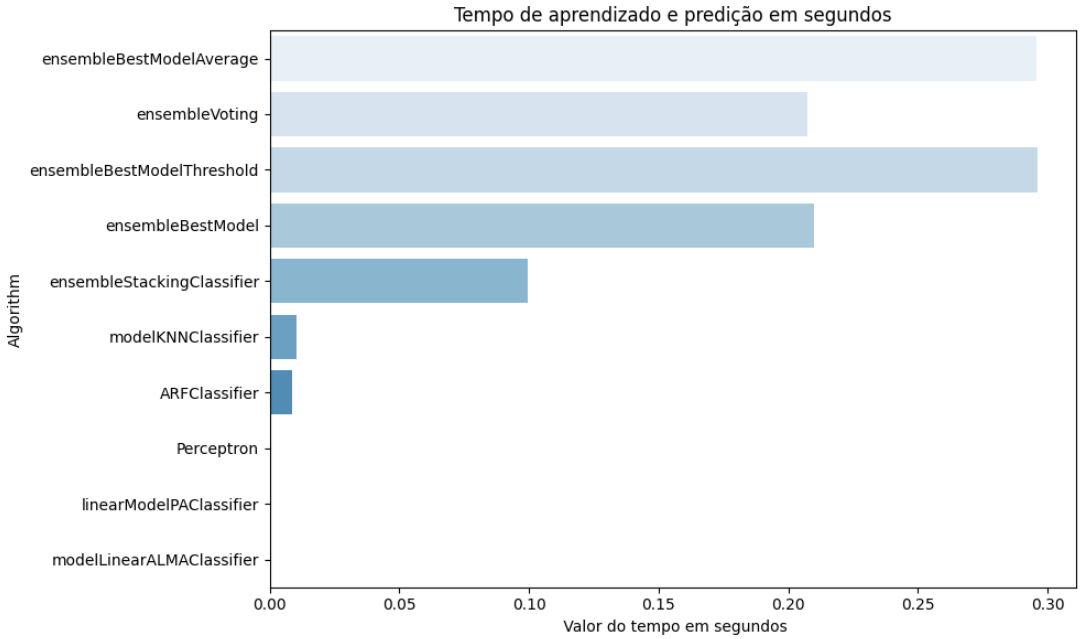

Figura 23: Desempenho médio do tempo de aprendizado e predição dos melhores modelos base e métodos ensemble.

accuracy, *precision* e *ROC Curve*, o modelo *ensemble best model* ficou em primeiro no *rank*, com a melhor precisão de acertos, enquanto o *ensemble best model average* ficou em terceiro. Em relação à métrica *precision*, considerando a sua média ("Avg"), o *ensemble best model average* liderou e o *ensemble best model* ficou em terceiro. Na métrica *Rolling ROC Curve* todos os métodos *ensemble* propostos obtiveram praticamente o mesmo desempenho.

O desempenho obtidos pelos métodos *ensemble* mensurados pelas métricas *Recall*,*precision* e *ROC Curve*, foram selecionadas para responder à questão **Q2**, que indaga sobre *a capacidade da sensibilidade de predizer as classes verdadeiro positivo corretas dos modelos ensemble propostos em cada conjunto de dados.* Observou-se que o modelo que apresentou a melhor sensibilidade na distinção das previsões corretas das classes verdeiro positivo foi o método proposto *ensemble best model.* No ranqueamento das métricas *Recall*,*precision* e *ROC Curve*, tal modelo ficou em primeiro lugar. Por outro lado, fazendo a análise da média da métrica calculada ("Avg"), apenas para a métrica *precision* o modelo proposto *ensemble best model* ficou em terceiro lugar, na primeira posição dela ficou *ensemble best model average.* Quanto a métrica *rolling ROC curve*, o desempenho dos métodos *ensemble* propostos foram significativos, ficando entre os melhores, mas alguns modelos base foram superiores em determinadas observações no *dataset* ConceptDrift.

Consideramos as métricas MCC, *accuracy*, *Balanced accuracy* e Cohen kappa, para responder à questão **Q3**, que indaga sobre o *comportamento dos modelos propostos e individuais na concordância das classes verdadeiras nas predições.* O modelo que apresentou

o melhor desempenho na avaliação da predição e acertos das classes verdadeiras, tanto as positivas (VP) quanto as negativas (VN), foi o modelo proposto *ensemble best model*, ficando sempre na melhor posição em relação ao *rank*. Ao considerar a média calculadas das métricas, tal modelo também foi o melhor. No que diz respeito ao *rank*, o modelo base *ensemble Stacking Classifier* destacou-se como o melhor entre os classificadores base, ficando em segundo lugar em termo gerais do *rank*.

Para responder à questão **Q4**, que investiga se *os métodos ensembles se comportam melhor do que os modelos individuais em algum conjunto de dados com desvio de conceito*, consideramos a métrica *rolling ROC Curve* presente na seção 4.4.4. Os métodos *ensemble* propostos obtiveram na maior parte das observações desempenho acima dos 0.9. Os métodos *ensemble* propostos no momento do desvio de conceito comportaram-se de maneira semelhante. Alguns modelos base superaram os métodos ensemble no momento do desvio de conceito, devido à métrica utilizada para a combinação (uma métrica acumulativa da *accuracy*) não foi adequada para uma adaptação melhor dos *ensemble* propostos no momento do desvio de conceito.

A questão **Q5**, cujo propósito é *investigar a eficiência dos processos de treinamento e predição dos métodos propostos em relação ao tempo*, pode ser respondida considerando os resultados presentes na tabela 11. Os resultados demonstram que todos os métodos propostos tiveram um desempenho inferior em relação ao tempo se comparados aos modelos base. Isso ocorreu devido à forma arquitetada no processamento deles, que foi de forma sequencial. Entretanto, deve-se ressaltar que os métodos propostos demandaram no máximo 1.6 segundos para predição e treinamento, o que não inviabiliza seu uso em cenários que apresentam restrições de tempos não rigorosas (*soft time requirements*).

5 CONCLUSÃO E TRABALHOS FUTURO

Este trabalho propõe quatro métodos *ensemble* para classificação binária de fluxo de dados online. Esses métodos combinam vários modelos base heterogêneos de aprendizado de máquina online. Entretanto, os resultados obtidos através de experimentos em oito *datasets* mostra a viabilidade da aplicação desses métodos nos fluxos de dados online. Na maioria das métricas selecionadas para responder o GQM, os métodos propostos *ensemble best model* e *ensemble best model average* estiveram entre os três melhores modelos. Por outro lado, os modelos *ensemble best model threshold* e *ensemble voting* demonstraram desempenho mais modesto.

Para trabalhos futuros, sugere-se explorar a substituição da métrica atualmente utilizada para a combinação dos modelos base para a predição *ensemble*. Investigar qual a métrica mais pertinente para combinação para cada um dos *dataset*, visando aumentar o acerto dos métodos *ensemble* em sua predição para a classe que tem maior relevância. Recomenda-se dar prioridade às métricas adaptadas da janela deslizante (*rolling*) e variar o tamanho da janela deslizante, especialmente em cenários que envolvem dados com desvio de conceito. A adoção dessas métricas adaptativas visa aprimorar o desempenho dos métodos *ensemble best model*, *ensemble best model threshold* e *ensemble best model average* em fluxos de dados dinâmicos, garantindo uma melhor adaptação em cada observação ao longo do tempo que contenha desvio de conceito.

Essa abordagem busca otimizar a capacidade dos modelos de responder de forma dinâmica às mudanças nos padrões dos dados, especialmente em ambientes onde o desvio de conceito é uma preocupação relevante. Ao priorizar métricas adaptativas do *rolling* e variar o tamanho da janela deslizante, espera-se melhorar a sensibilidade dos métodos propostos para acertar as classes verdadeiras ao enfrentar as variações nos fluxos de dados, resultando em um desempenho mais robusto e preciso em situações de mudança constante. Essa linha de pesquisa promete contribuir significativamente para aprimorar a capacidade dos métodos *ensemble* em lidar com cenários dinâmicos e complexos encontrados em aplicações do mundo real.

Adicionalmente, uma área promissora para pesquisas futuras é a investigação de técnicas de ajuste de hiperparâmetros em tempo real, conhecidas como *online hyperparameter tunning*. Essa abordagem visa otimizar os hiperparâmetros dos modelos de forma contínua e adaptativa à medida que novos dados chegam, permitindo uma adaptação dinâmica e automatizada dos modelos aos padrões emergentes nos fluxos de dados. A integração dessas técnicas com os métodos propostos pode potencializar ainda mais o desempenho e a capacidade de generalização dos modelos em ambientes de fluxos de dados em constante mudança.

Para aprimorar os métodos propostos em termos de eficiência do tempo, uma abor-

dagem promissora é explorar a execução em paralelo ou distribuída durante o treinamento e a predição dos métodos *ensemble* propostos. Essa estratégia visa reduzir significativamente o tempo necessário para realizar tais tarefas, aproveitando recursos computacionais distribuídos de maneira mais eficaz.

Além disso, uma análise criteriosa dos modelos base que não contribuíram para aprimorar os *ensemble* pode ser conduzida. Identificar e remover esses modelos da composição da base dos *ensemble* pode resultar em uma redução adicional do tempo de treinamento e predição, ao mesmo tempo que simplifica a arquitetura dos modelos, tornando-os mais eficientes e fáceis de interpretar.

Essas estratégias visam não apenas melhorar o desempenho dos modelos propostos em relação ao tempo de execução, mas também otimizar seus recursos computacionais, tornando-os mais escaláveis e viáveis para implementação em ambientes do mundo real.

Feito as alterações proposta, como parte dos trabalhos futuros, pretende-se aplicar os métodos propostos em dispositivos IoT, nos quais os recursos computacionais são limitados. Este estudo visa avaliar o desempenho e comportamento dos métodos propostos em ambientes com restrições de hardware, considerandos aspectos como eficiência energética, uso de memória e processamento computacional. A viabilidade desses métodos em dispositivos IoT pode abrir novas oportunidades para implementações práticas em aplicações que envolve classificação binária no campo de pesquisa sobre Internet das Coisas.

Além disso, almeja-se a aplicação dos métodos propostos em cenários de fluxos de dados reais. Essas aplicações práticas permitirão avaliar a eficácia e robustez dos métodos *ensemble* propostos em situações do mundo real, considerando as complexidades e dinâmicas dos fluxos de dados.

REFERÊNCIAS

1 SARKA, D.; BALI, R.; SHARMA, T. ***Practical Machine Learning with Python.*** [S.l.]: ed. Apress, 2018. 3, 35-44 p.

2 KRAWCZYK, B. et al. **Ensemble learning for data stream analysis: A survey.** p. 2–3, 133, 2017.

3 BROWNLEE, J. *Tour of Evaluation Metrics for Imbalanced Classification.* 2021. Disponível em: ⟨https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/⟩. Acesso em: 19 Fev. 2024.

4 SILVA, T. P. et al. **An Online Ensemble Method for Auto-Scaling NFV-based Applications in the Edge.** p. 2–13, 2023.

5 HOSSIN, M.; SULAIMAN, M. **A Review On Evaluation Metrics For Data Classification Evaluations.** p. 3–5, 2015.

6 JEEVA, M. ***Data Streams and Online Machine Learning in Python.*** [s.n.]. Disponível em: ⟨https://medium.com/analytics-vidhya/data-streams-and-online-machine-learning-in-python-a382e9e8d06a⟩. Acesso em: 30 Jan. 2024.

7 HILPISCH, Y. ***Artificial Intelligence in Finance.*** [S.l.]: ed. O'Reilly, 2021. 28 p.

8 PEREIRA, L. F.; ROSSI, R. G. **Estudo de técnicas de Ensemble para classificação binária.** *Universidade Federal de Mato Grosso do Sul*, p. 1–2, 17.

9 TATSAT, H.; PURI, S.; LOOKABAUGH, B. **Machine Learning & Data Science Blueprints for Finance: From Building Trading Strategies to Robo-Advisors Using Python.** *ed. O'Reilly*, p. 151, 6, 8, 9, 2021.

10 GOMES, H. M. et al. **A Survey on Ensemble Learning for Data Stream Classification.** p. 1–10, 2017.

11 FAN, X.; LUNG, C.-H.; AJILA, S. A. **An Adaptive Diversity-Based Ensemble Method for Binary Classification.** *IEEE*, p. 1, 823, 2017.

12 BIFET, A.; GAVALDA, R. Learning from time-changing data with adaptive windowing. *Society for Industrial and Applied Mathematics*, 2007.

13 FERRI, C.; HERNÁNDEZ-ORALLO, J.; MODROIU, R. **An experimental comparison of performance measures for classification.** p. 1–4, 2009.

14 MOSLEY, L. S. D. **A balanced approach to the multi-class imbalance problem**. p. 25, 2013.

15 COHEN, J. **A coefficient of agreement for nominal scales.** 1960.

16 NELLORE, S. B. **Various performance measures in Binary Classification - An Overview of ROC study**. p. 6, 2015.

17 MONTIEL, J. et al. River: machine learning for streaming data in python. *Journal of Machine Learning Research*, v. 22, n. 110, p. 1–8, 2021. Disponível em: ⟨http://jmlr.org/papers/v22/20-1380.html⟩.

18 OZA, N.; RUSSEL, S. **Online Bagging and Boosting**. p. 105–112, 2001.

19 BASILI, V. R.; CALDIERA, G.; RAMBACH, H. D. **The goal question metric approach. *encyclopedia of software engineering.*** p. 528–532., 1998.

20 CANBECK, G. et al. **Binary classification performance measures/metrics: A comprehensive visualized roadmap to gain new insights.** *IEEE*, p. 1, 2017.

21 WILLIAMS, G. et al. A comparative study of rnn for outlier detection in data mining. *IEEE International Conference on Data Mining*, p. 709, 2002.

22 POZZOLO, A. D. et al. Calibrating probability with undersampling for unbalanced classification. *IEEE*, 2015.

23 HARRIS, M. Splice-2 comparative evaluation: Electricity pricing. 1999.

24 ALMEIDA, T.; HIDALGO, J.; YAMAKAMI, A. Contributions to the study of sms spam filtering: new collection and results. *Proceedings of the 11th ACM symposium on Document engineering*, p. 259–262, 2011.

25 ABDELHAMID, N.; AYESH, A.; THABTAH, F. Phishing detection based associative classification data mining. *Expert systems with applications*, 2014.

A – APÊNDICE

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModel	0.99739	0.99998	0.99925	0.99987	0.9241	0.97937	0.88981	0.91593	0.96321	1.12500
ensembleStackingClassifier	0.99726	0.99998	0.99916	0.99982	0.92377	0.95981	0.8817	0.9056	0.95839	5.62500
ensembleBestModelAverage	0.99721	0.99997	0.99926	0.99979	0.88663	0.89254	0.86566	0.9032	0.94303	7.5
ensembleBestModelThreshold	0.99711	0.99997	0.99922	0.99979	0.85267	0.89343	0.87019	0.8976	0.93875	8.5
ensembleVoting	0.99700	0.99997	0.99921	0.99978	0.84042	0.89666	0.62472	0.90400	0.90772	9.75
forestAMFClassifier	0.9861	0.99998	0.99925	0.99985	0.85849	0.86399	0.88866	0.88231	0.93483	11.62500
ensembleLeveragingBaggingClassifier	0.9972	0.99996	0.99912	0.9998	0.85258	0.91855	0.53132	0.89600	0.89932	12.87500
linearModelPAClassifier	0.99634	0.99997	0.99917	0.99967	0.83543	0.90007	0.55057	0.8952	0.89705	13.37500
OneVsRestClassifier	0.99738	0.99997	0.99918	0.99975	0.81649	0.8656	0.53746	0.89191	0.88847	14.12500
ensembleVotingClassifier	0.98613	0.99998	0.99917	0.99978	0.7869	0.90635	0.62585	0.8832	0.89842	14.25
modelKNNClassifier	0.97	0.99987	0.99868	0.99983	0.82686	0.88014	0.88488	0.89672	0.93212	14.5
OutputCodeClassifier	0.99738	0.99997	0.99917	0.99979	0.81649	0.86596	0.53708	0.89271	0.88857	14.75
ARFClassifier	0.99489	0.99961	0.99777	0.99968	0.83289	0.88283	0.88337	0.90713	0.93727	14.75
ensembleBaggingClassifier	0.9972	0.99996	0.99914	0.99973	0.80888	0.87729	0.53491	0.8904	0.88844	16.62500
ensembleADWINBaggingClassifier	0.99251	0.99997	0.99913	0.99972	0.8052	0.87729	0.53472	0.8904	0.88737	17.12500
OneVsOneClassifier	0.99738	0.99991	0.99918	0.99943	0.81647	0.86519	0.53682	0.89398	0.88855	17.62500
ensembleADWINBoostingClassifier	0.99154	0.99997	0.99912	0.99972	0.80312	0.87567	0.53943	0.8864	0.88687	18.0
ensembleSRPClassifier	0.87729	0.99963	0.99777	0.99968	0.76509	0.88283	0.7307	0.90873	0.89522	18.25
ensembleAdaBoostClassifier	0.99308	0.9997	0.99826	0.99983	0.78614	0.87188	0.64314	0.88311	0.89689	18.25
BernoulliNB	0.75097	0.9961	0.99900	0.99968	0.74713	0.97865	0.55275	0.90472	0.86612	19.0
Perceptron	0.99249	0.9999	0.99859	0.99943	0.90296	0.93864	0.51717	0.8584	0.90095	19.37500
HardSamplingClassifier	0.86294	0.9961	0.99691	0.99968	0.76231	0.97757	0.56275	0.90633	0.88307	19.87500
treeExtremelyFastDecisionTreeClassifier	0.98149	0.9961	0.99777	0.99968	0.82009	0.87924	0.62597	0.88471	0.89813	20.12500
LogisticRegression	0.99613	0.9999	0.99887	0.99954	0.85117	0.86903	0.51943	0.8896	0.89046	20.37500
ensembleBOLEClassifier	0.98645	0.99996	0.99673	0.9996	0.92033	0.20348	0.64314	0.85829	0.82600	20.62500
modelGaussianNB	0.9168	0.99995	0.98419	0.99348	0.7319	0.9695	0.61521	0.88471	0.88697	22.25
RandomSampler	0.74486	0.99978	0.99777	0.99955	0.68891	0.96661	0.49207	0.89832	0.84848	23.12500
RandomUnderSampler	0.25539	0.9961	0.99905	0.99968	0.68303	0.91314	0.47037	0.90465	0.77768	23.12500
RandomOverSampler	0.74461	0.99966	0.99755	0.99967	0.69279	0.97936	0.46726	0.89672	0.8472	23.37500
treeHoeffdingTreeClassifier	0.97	0.9961	0.99777	0.99968	0.75176	0.87834	0.5652	0.80705	0.87074	23.75
linearModelSoftmaxRegression	0.86081	0.78846	0.99918	0.73207	0.80912	0.63216	0.53557	0.89111	0.78106	24.37500
treeHoeffdingAdaptiveTreeClassifier	0.97944	0.9961	0.99777	0.99968	0.78504	0.86183	0.54029	0.82146	0.8727	25.0
modelLinearALMAClassifier	0.83945	0.8516	0.99292	0.76499	0.87511	0.56961	0.50642	0.8256	0.77821	28.0

Tabela 12: Dados referentes ao *rank* da métrica *accuracy*.

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModel	0.99700	0.99864	0.92529	875.0	0.91408	0.97024	0.88923	0.88636	0.93198	4.25
ensembleBestModelAverage	0.99656	0.99955	0.9162	1.0	0.87814	1.0	0.88698	0.87002	0.94343	5.37500
ensembleStackingClassifier	0.99768	0.99955	0.88827	1.0	0.91515	0.95797	0.8878	0.88809	0.94181	5.75
ensembleVoting	0.99614	0.99955	0.91379	1.0	0.85483	1.0	0.66782	0.87413	0.91328	6.75
ensembleBestModelThreshold	0.99636	0.99955	0.91429	1.0	0.85985	0.93714	0.83599	0.85959	0.92535	9.37500
modelKNNClassifier	0.97425	0.99675	0.9596	0.79167	0.80972	0.9759	0.87767	0.88	0.90819	11.87500
forestAMFClassifier	0.98147	0.99774	0.93023	0.9	0.84757	0.13333	0.89064	0.86496	0.81824	12.37500
ensembleVotingClassifier	0.98037	0.99955	0.89326	1.0	0.81947	0.97071	0.67467	0.85263	0.89883	12.87500
linearModelPAClassifier	0.99648	0.99773	0.86082	0.44444	0.82592	0.84926	0.4875	0.87568	0.79223	13.37500
ensembleLeveragingBaggingClassifier	0.99739	0.99728	0.82927	0.78947	0.85286	0.79959	0.46137	0.87321	0.82506	13.37500
OneVsRestClassifier	0.99684	0.99818	0.89385	1.0	0.81972	0.0	0.45174	0.86396	0.75304	15.25
OutputCodeClassifier	0.99685	0.99818	0.89326	1.0	0.81965	0.0	0.45019	0.8642	0.75279	15.37500
ARFClassifier	0.99435	0.97333	0.0	0.0	0.8163	1.0	0.88688	0.89397	0.6956	16.12500
ensembleAdaBoostClassifier	0.99396	0.99132	0.59176	0.9375	0.74723	1.0	0.62143	0.84749	0.84134	17.12500
ensembleADWINBaggingClassifier	0.99259	0.99773	0.86559	0.58824	0.82062	0.69325	0.44431	0.86631	0.78358	17.37500
OneVsOneClassifier	0.99684	0.99009	0.89326	0.13043	0.81976	0.14286	0.45051	0.86918	0.66162	18.0
ensembleBaggingClassifier	0.99684	0.99548	0.85864	0.62500	0.81883	0.69325	0.44632	0.86116	0.78694	18.12500
LogisticRegression	0.99774	0.98083	0.74554	0.34091	0.83836	0.94737	0.44289	0.84864	0.76778	18.37500
ensembleADWINBoostingClassifier	0.99192	0.99638	0.85714	0.61538	0.81013	0.67089	0.46319	0.85121	0.78203	19.37500
Perceptron	0.99415	0.97996	0.66400	0.27778	0.88569	0.77626	0.46147	0.83789	0.73465	19.5
model$_{G}aussianNB$	0.88499	0.99145	0.10887	0.02572	0.81951	0.97844	0.65277	0.85166	0.66418	19.5
ensembleBOLEClassifier	0.98824	0.99548	0.24	0.36667	0.90949	0.13889	0.6152	0.83036	0.63554	19.62500
treeExtremelyFastDecisionTreeClassifier	0.98379	0.0	0.0	0.0	0.82281	0.98684	0.64812	0.87246	0.53925	20.0
linearModelSoftmaxRegression	0.99999	0.01804	0.87701	0.00059	0.81038	0.24952	0.48414	0.84193	0.5352	20.12500
ensembleSRPClassifier	0.86024	0.97987	0.0	0.0	0.74613	1.0	0.81654	0.87009	0.65911	20.37500
HardSamplingClassifier	0.83447	0.0	0.36478	0.0	0.78356	0.96837	0.51824	0.88256	0.54400	21.12500
BernoulliNB	0.72273	0.0	0.7971	0.0	0.73383	0.97432	0.50162	0.88078	0.5763	21.62500
RandomSampler	0.71600	0.97451	0.5	0.3617	0.61793	0.8423	0.46378	0.86207	0.66729	23.37500
treeHoeffdingTreeClassifier	0.9639	0.0	0.0	0.0	0.73658	1.0	0.71176	0.84152	0.53172	23.37500
RandomOverSampler	0.71579	0.96884	0.46875	0.48148	0.62262	0.97024	0.45466	0.84718	0.69119	24.37500
modelLinearALMAClassifier	0.97876	0.02551	0.1887	0.00089	0.85492	0.19744	0.45522	0.77318	0.43433	24.37500
RandomUnderSampler	0.02598	0.0	0.81068	0.0	0.60781	0.6123	0.45572	0.86371	0.42203	26.37500
treeHoeffdingAdaptiveTreeClassifier	0.98049	0.0	0.0	0.0	0.75516	0.44103	0.48157	0.8253	0.43544	26.75

Tabela 13: Dados referentes ao *rank* da métrica *precision*.

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModel	0.99435	0.99773	0.81698	0.78256	0.84452	0.90884	0.7769	0.83077	0.86908	1.12500
ensembleStackingClassifier	0.99407	0.99704	0.79543	0.65822	0.8438	0.8171	0.76051	0.80856	0.83434	5.75
ensembleBestModelAverage	0.99395	0.99659	0.8205	0.57729	0.76731	0.41983	0.72874	0.80543	0.76371	6.87500
ensembleBestModelThreshold	0.99374	0.99659	0.80957	0.57729	0.69708	0.42444	0.73996	0.79487	0.75419	8.37500
ensembleVoting	0.9935	0.99659	0.80681	0.54766	0.67197	0.45223	0.23275	0.80658	0.68851	9.5
forestAMFClassifier	0.96993	0.9975	0.81662	0.73478	0.70927	-0.00011	0.77458	0.76098	0.72044	12.0
ensembleLeveragingBaggingClassifier	0.99394	0.99545	0.79466	0.62819	0.6968	0.60586	0.01573	0.78945	0.69001	12.25
linearModelPAClassifier	0.99207	0.99614	0.8025	0.24331	0.66142	0.47554	0.01717	0.78752	0.62196	14.0
ensembleVotingClassifier	0.97004	0.99704	0.79767	0.54766	0.56193	0.51984	0.23628	0.76432	0.67435	14.62500
modelKNNClassifier	0.93488	0.98329	0.63891	0.70801	0.64407	0.30383	0.76702	0.79034	0.7213	14.75
OutputCodeClassifier	0.99432	0.99636	0.79767	0.57729	0.62187	0.0	0.0015	0.78336	0.59655	15.87500
ARFClassifier	0.98893	0.94839	0.0	0.0	0.65655	0.33293	0.76387	0.81135	0.56275	16.37500
OneVsRestClassifier	0.99432	0.99613	0.80044	0.44716	0.62187	-0.00745	0.00279	0.78165	0.57961	16.75
Perceptron	0.98372	0.98741	0.70235	0.37242	0.8014	0.73359	0.02387	0.71249	0.66466	17.37500
OneVsOneClassifier	0.99433	0.99251	0.79946	0.11588	0.62183	0.00093	0.00173	0.78535	0.53900	18.0
ensembleBaggingClassifier	0.99393	0.99523	0.79423	0.45632	0.60618	0.28492	-0.00172	0.7788	0.61349	18.0
ensembleAdaBoostClassifier	0.98501	0.95997	0.64666	0.68459	0.56253	0.19617	0.27059	0.76489	0.6338	18.12500
ensembleBOLEClassifier	0.97062	0.99546	0.22565	0.36647	0.83679	0.0474	0.2723	0.71305	0.55347	18.62500
ensembleADWINBaggingClassifier	0.98375	0.99614	0.79011	0.44268	0.59875	0.28492	-0.00341	0.77817	0.60889	18.87500
linearModelSoftmaxRegression	0.75135	0.11894	0.80271	0.00931	0.60648	0.31709	0.07269	0.78417	0.43284	19.5
RandomOverSampler	0.45214	0.95521	0.58781	0.45661	0.38381	0.90884	0.04601	0.79563	0.57326	19.5
RandomSampler	0.45276	0.97131	0.60536	0.45252	0.37664	0.86294	0.06607	0.79595	0.57294	19.5
LogisticRegression	0.99162	0.98785	0.74664	0.41264	0.6942	0.13963	-0.00683	0.7792	0.59312	19.62500
ensembleADWINBoostingClassifier	0.98166	0.99614	0.78867	0.40498	0.59406	0.26917	0.01339	0.77158	0.60246	19.75
ensembleSRPClassifier	0.73254	0.95158	0.0	0.0	0.51503	0.33293	0.46494	0.81755	0.47682	20.12500
BernoulliNB	0.46221	0.0	0.76748	0.0	0.47654	0.90553	0.07833	0.80729	0.43717	20.25
modelGaussianNB	0.82509	0.99366	0.30049	0.11609	0.45496	0.86331	0.2107	0.76771	0.5665	20.37500
HardSamplingClassifier	0.70562	-0.00012	0.43412	0.0	0.50915	0.90066	0.09309	0.81053	0.43163	21.25
RandomUnderSampler	-0.45213	0.0	0.77869	0.0	0.36992	0.72401	0.04834	0.80965	0.28481	22.25
treeExtremelyFastDecisionTreeClassifier	0.95985	0.0	0.0	0.0	0.62935	0.29433	0.23248	0.76556	0.3602	23.87500
modelLinearALMAClassifier	0.70697	0.14701	0.35028	0.01807	0.74422	0.1824	0.01408	0.6524	0.35193	24.12500
treeHoeffdingAdaptiveTreeClassifier	0.95536	0.0	0.0	0.0	0.55839	0.17158	0.04444	0.63577	0.29569	26.75
treeHoeffdingTreeClassifier	0.93497	0.0	0.0	0.0	0.48651	0.28459	0.09641	0.60833	0.30135	26.87500

Tabela 14: Dados referentes ao *rank* da métrica MCC.

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModel	0.9968	0.99841	0.86092	0.84998	0.92178	0.93434	0.88716	0.91714	0.92082	3.12500
ensembleStackingClassifier	0.99697	0.99729	0.8564	0.71667	0.92122	0.86365	0.87803	0.90475	0.89187	8.5
ensembleLeveragingBaggingClassifier	0.99681	0.99683	0.88099	0.74998	0.84422	0.75156	0.50697	0.8956	0.82787	10.87500
ensembleBestModelAverage	0.99649	0.99683	0.86764	0.66667	0.88197	0.59906	0.85975	0.90461	0.84663	11.0
ensembleBestModelThreshold	0.99634	0.99683	0.85867	0.66667	0.84315	0.60863	0.87147	0.89962	0.84267	12.62500
Perceptron	0.99187	0.99747	0.87178	0.7498	0.9007	0.86387	0.51193	0.85632	0.84297	12.75
forestAMFClassifier	0.98188	0.99864	0.85868	0.79999	0.8525	0.49999	0.88569	0.88057	0.84474	13.5
modelGaussianNB	0.88494	0.99795	0.91601	0.76348	0.69789	0.89357	0.58600	0.88553	0.82817	13.87500
ensembleVoting	0.99616	0.99683	0.85643	0.65	0.8288	0.61446	0.59654	0.90492	0.80552	14.12500
linearModelPAClassifier	0.99578	0.99728	0.8743	0.56664	0.82762	0.65037	0.50359	0.89429	0.78874	14.87500
LogisticRegression	0.99611	0.99747	0.87415	0.74985	0.84505	0.51194	0.49691	0.8919	0.79542	15.37500
modelKNNClassifier	0.96657	0.98507	0.71298	0.81664	0.81994	0.55401	0.8829	0.8954	0.82919	15.75
RandomSampler	0.6485	0.98412	0.86689	0.78318	0.69037	0.94843	0.52545	0.90006	0.79337	16.25
ensembleBaggingClassifier	0.99659	0.9975	0.86758	0.66664	0.7954	0.57046	0.49937	0.89081	0.78554	16.37500
ensembleBOLEClassifier	0.98487	0.99773	0.60686	0.68323	0.91794	0.51916	0.63396	0.85738	0.77514	16.62500
RandomOverSampler	0.64815	0.97099	0.86902	0.71659	0.69395	0.93434	0.51179	0.90065	0.78069	17.5
ensembleVotingClassifier	0.98146	0.99729	0.85641	0.65	0.76748	0.65456	0.59714	0.8836	0.79849	18.12500
ensembleADWINBoostingClassifier	0.99027	0.99796	0.86309	0.63331	0.78965	0.56556	0.50502	0.88765	0.77906	18.12500
linearModelSoftmaxRegression	0.89106	0.89246	0.8676	0.61607	0.79723	0.73217	0.53654	0.89486	0.7785	18.25
ARFClassifier	0.99382	0.96218	0.5	0.5	0.82627	0.56292	0.88007	0.90567	0.76637	18.5
OutputCodeClassifier	0.99674	0.99728	0.85641	0.66667	0.80487	0.5	0.50053	0.89305	0.77694	18.62500
OneVsOneClassifier	0.99673	0.99748	0.85801	0.5516	0.80482	0.50005	0.50062	0.8937	0.76288	18.75
BernoulliNB	0.65908	0.5	0.86974	0.5	0.73235	0.92997	0.53753	0.90475	0.70418	18.75
ensembleAdaBoostClassifier	0.99226	0.96493	0.85371	0.74999	0.7814	0.52209	0.63183	0.88431	0.79757	19.0
ensembleADWINBaggingClassifier	0.99128	0.99728	0.86086	0.66663	0.79042	0.57046	0.49877	0.89001	0.78321	19.5
OneVsRestClassifier	0.99673	0.99706	0.85865	0.6	0.80486	0.49979	0.50099	0.89214	0.76878	19.87500
RandomUnderSampler	0.35183	0.5	0.87424	0.5	0.68711	0.93287	0.51344	0.90731	0.65835	20.0
modelLinearALMAClassifier	0.86796	0.92393	0.82643	0.71584	0.87183	0.63382	0.50708	0.82852	0.77193	20.62500
HardSamplingClassifier	0.81801	0.5	0.75908	0.5	0.74208	0.92821	0.54309	0.90637	0.71211	20.87500
ensembleSRPClassifier	0.84371	0.9622	0.5	0.5	0.75354	0.56292	0.71054	0.91113	0.71800	20.87500
treeExtremelyFastDecisionTreeClassifier	0.97926	0.5	0.5	0.5	0.80884	0.5501	0.60134	0.8823	0.66523	24.0
treeHoeffdingAdaptiveTreeClassifier	0.97635	0.5	0.5	0.5	0.77789	0.54627	0.52061	0.81372	0.64186	26.25
treeHoeffdingTreeClassifier	0.96226	0.5	0.5	0.5	0.73795	0.54618	0.51708	0.79404	0.63219	27.75

Tabela 15: Dados referentes ao *rank* da métrica *balanced accuracy*.

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModel	0.99892	0.99683	0.72197	0.7	0.90643	0.87282	0.86153	0.92687	0.87317	5.25
ensembleBestModelAverage	0.99908	0.99367	0.73543	0.33333	0.85107	0.19813	0.80261	0.91606	0.72867	11.12500
RandomOverSampler	0.99548	0.94211	0.73991	0.43333	0.7016	0.87282	0.94318	0.93236	0.8201	11.75
linearModelSoftmaxRegression	0.78213	0.99729	0.73543	0.5	0.7185	0.86881	0.54588	0.92505	0.75913	11.75
ensembleBestModelThreshold	0.99912	0.99367	0.71749	0.33333	0.78011	0.21954	0.88384	0.91606	0.7304	11.87500
ensembleStackingClassifier	0.99803	0.99457	0.71300	0.43333	0.9043	0.73226	0.84259	0.89781	0.81449	12.12500
RandomSampler	0.99549	0.96834	0.73543	0.56667	0.70004	0.92359	0.84926	0.91408	0.83161	12.37500
modelGaussianNB	0.99968	0.99593	0.84753	0.53333	0.47255	0.78983	0.30303	0.89214	0.72925	12.37500
Perceptron	0.99409	0.99502	0.74439	0.5	0.88574	0.76171	0.46128	0.83942	0.77271	13.5
forestAMFClassifier	0.99706	0.99729	0.71749	0.6	0.81285	0.00268	0.8569	0.86654	0.73135	13.5
ensembleLeveragingBaggingClassifier	0.99824	0.99367	0.76233	0.5	0.78884	0.52343	0.27146	0.89234	0.71629	13.62500
LogisticRegression	0.9962	0.99502	0.74888	0.5	0.80454	0.0241	0.27904	0.91058	0.6573	13.75
ensembleBOLEClassifier	0.99058	0.99548	0.21525	0.36667	0.90211	0.95047	0.54503	0.85009	0.72696	14.87500
modelLinearALMAClassifier	0.7653	0.99683	0.65919	0.66667	0.85008	0.72155	0.51347	0.85219	0.75316	15.5
ensembleVoting	0.99918	0.99367	0.71300	0.3	0.75178	0.22892	0.32407	0.91241	0.65288	15.5
RandomUnderSampler	0.00454	0.0	0.74888	0.0	0.71411	0.95984	0.9314	0.92857	0.53592	15.87500
ensembleBaggingClassifier	0.99878	0.99502	0.73543	0.33333	0.70604	0.15127	0.15572	0.89416	0.62122	16.62500
linearModelPAClassifier	0.9978	0.99457	0.74888	0.13333	0.7759	0.30924	0.04924	0.88686	0.61198	17.25
ensembleADWINBoostingClassifier	0.99486	0.99593	0.72646	0.26667	0.70042	0.1419	0.17214	0.89781	0.61202	18.5
modelKNNClassifier	0.97892	0.97015	0.42601	0.63333	0.77407	0.10843	0.86364	0.88483	0.70492	18.62500
ensembleAdaBoostClassifier	0.99522	0.9299	0.70852	0.5	0.74995	0.04418	0.52231	0.89397	0.66800	18.87500
ensembleVotingClassifier	0.99827	0.99457	0.71300	0.3	0.63877	0.31058	0.31944	0.88686	0.64519	19.5
BernoulliNB	0.98998	0.0	0.73991	0.0	0.63449	0.86345	0.39015	0.90494	0.56536	19.75
ensembleADWINBaggingClassifier	0.9957	0.99457	0.72197	0.33333	0.69252	0.15127	0.15109	0.88686	0.61592	19.87500
OutputCodeClassifier	0.99905	0.99457	0.71300	0.33333	0.72791	0.0	0.14646	0.8958	0.60127	19.87500
ARFClassifier	0.99768	0.92447	0.0	0.0	0.78244	0.12584	0.84806	0.89397	0.57156	19.87500
OneVsOneClassifier	0.99907	0.99502	0.71622	0.10345	0.72766	0.00134	0.14947	0.89154	0.57297	20.0
OneVsRestClassifier	0.99906	0.99412	0.71749	0.2	0.7278	0.0	0.14773	0.89397	0.58502	20.12500
ensembleSRPClassifier	0.96466	0.92447	0.0	0.0	0.67701	0.12584	0.51515	0.93053	0.51721	22.37500
HardSamplingClassifier	0.97981	0.0	0.52018	0.0	0.60808	0.86078	0.35269	0.90676	0.52854	23.0
treeExtremelyFastDecisionTreeClassifier	0.9873	0.0	0.0	0.0	0.73435	0.1004	0.36279	0.86289	0.38097	25.37500
treeHoeffdingAdaptiveTreeClassifier	0.98746	0.0	0.0	0.0	0.73051	0.11513	0.32997	0.75137	0.3643	26.37500
treeHoeffdingTreeClassifier	0.99011	0.0	0.0	0.0	0.64644	0.09237	0.05093	0.68921	0.30863	30.25

Tabela 16: Tabela com Rank da Recall.

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModel	0.99434	0.99773	0.81071	0.77772	0.8445	0.90717	0.77659	0.83009	0.86736	1.12500
ensembleStackingClassifier	0.99407	0.99704	0.79063	0.60458	0.84375	0.80772	0.75969	0.80852	0.82575	6.0
ensembleBestModelAverage	0.99394	0.99658	0.81555	0.49992	0.76704	0.29969	0.72597	0.80455	0.73791	7.0
ensembleBestModelThreshold	0.99373	0.99658	0.80364	0.49992	0.69475	0.32122	0.73897	0.79353	0.73029	8.62500
ensembleVoting	0.9935	0.99658	0.80062	0.46146	0.66815	0.33958	0.20287	0.80597	0.65859	9.62500
ensembleLeveragingBaggingClassifier	0.99394	0.99545	0.79395	0.61215	0.69529	0.58912	0.0145	0.7893	0.68546	11.5
forestAMFClassifier	0.96969	0.9975	0.80976	0.71993	0.70882	-0.00003	0.77412	0.76098	0.7176	12.0
linearModelPAClassifier	0.99207	0.99613	0.80055	0.20501	0.66049	0.41126	0.00784	0.78747	0.6076	14.37500
ensembleVotingClassifier	0.96973	0.99704	0.79261	0.46146	0.55088	0.4338	0.20436	0.76384	0.64671	14.87500
modelKNNClassifier	0.93486	0.9832	0.5895	0.70362	0.6436	0.17301	0.76694	0.79033	0.69813	15.12500
OutputCodeClassifier	0.99432	0.99636	0.79261	0.49992	0.61883	0.0	0.00114	0.78295	0.58577	15.5
ARFClassifier	0.98892	0.94807	0.0	0.0	0.65611	0.19956	0.76327	0.81135	0.54591	16.87500
OneVsRestClassifier	0.99432	0.99613	0.79561	0.33326	0.61882	-0.00072	0.00212	0.78128	0.5651	17.12500
Perceptron	0.98372	0.98738	0.7012	0.35688	0.8014	0.73355	0.02387	0.71249	0.66256	17.25
ensembleBaggingClassifier	0.99393	0.99523	0.79184	0.43466	0.60166	0.21045	-0.00133	0.77835	0.6006	17.37500
OneVsOneClassifier	0.99432	0.99251	0.79459	0.1151	0.61877	0.00017	0.00132	0.78514	0.53774	17.87500
ensembleBOLEClassifier	0.97062	0.99546	0.22532	0.36647	0.83677	0.01105	0.27081	0.71289	0.54867	18.25
ensembleAdaBoostClassifier	0.98500	0.95947	0.64403	0.6521	0.56253	0.07411	0.26766	0.76399	0.61361	18.37500
ensembleADWINBaggingClassifier	0.98374	0.99613	0.78685	0.4254	0.59298	0.21045	-0.00262	0.77800	0.59637	18.5
ensembleADWINBoostingClassifier	0.98165	0.99614	0.78597	0.37197	0.58978	0.19666	0.01066	0.77068	0.58794	18.87500
RandomSampler	0.34945	0.9713	0.5942	0.44134	0.37421	0.8617	0.04724	0.79482	0.55428	19.0
linearModelSoftmaxRegression	0.72167	0.02799	0.79959	0.00055	0.60344	0.22663	0.07221	0.7812	0.40416	19.5
LogisticRegression	0.99162	0.98783	0.74664	0.40518	0.69377	0.04062	-0.00641	0.77758	0.5796	19.75
RandomOverSampler	0.34869	0.95511	0.57275	0.45598	0.38154	0.90717	0.02142	0.79251	0.5544	19.87500
BernoulliNB	0.37088	0.0	0.76694	0.0	0.47311	0.90337	0.07666	0.80705	0.42475	20.12500
ensembleSRPClassifier	0.72106	0.95118	0.0	0.0	0.51332	0.19956	0.43613	0.81601	0.45466	20.25
modelGaussianNB	0.81025	0.99366	0.18975	0.04851	0.41904	0.85694	0.18108	0.76703	0.53328	20.75
HardSamplingClassifier	0.68165	-0.00001	0.42733	0.0	0.49882	0.89862	0.08884	0.81029	0.42569	21.0
RandomUnderSampler	-0.22496	0.0	0.77808	0.0	0.36582	0.69826	0.02449	0.80787	0.3062	22.5
treeExtremelyFastDecisionTreeClassifier	0.95984	0.0	0.0	0.0	0.62652	0.1615	0.21139	0.76552	0.3406	22.75
modelLinearALMAClassifier	0.67898	0.04246	0.29095	0.00116	0.74421	0.12614	0.01398	0.64975	0.31845	24.12500
treeHoeffdingAdaptiveTreeClassifier	0.95531	0.0	0.0	0.0	0.55817	0.13456	0.04247	0.63371	0.29053	27.37500
treeHoeffdingTreeClassifier	0.93429	0.0	0.0	0.0	0.48366	0.14985	0.03741	0.60006	0.27566	27.75000

Tabela 17: Dados referentes ao *rank* da métrica Cohen Kappa.

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModel	0.9968	0.99841	0.86092	0.84998	0.92178	0.93434	0.88716	0.91714	0.92082	3.12500
ensembleStackingClassifier	0.99697	0.99729	0.8564	0.71667	0.92122	0.86365	0.87803	0.90475	0.89187	8.5
ensembleLeveragingBaggingClassifier	0.99681	0.99683	0.88099	0.74998	0.84422	0.75156	0.50697	0.8956	0.82787	10.87500
ensembleBestModelAverage	0.99649	0.99683	0.86764	0.66667	0.88197	0.59906	0.85975	0.90461	0.84663	11.0
ensembleBestModelThreshold	0.99634	0.99683	0.85867	0.66667	0.84315	0.60863	0.87147	0.89962	0.84267	12.62500
Perceptron	0.99187	0.99747	0.87178	0.7498	0.9007	0.86387	0.51193	0.85632	0.84297	12.75
forestAMFClassifier	0.98188	0.99864	0.85868	0.79999	0.8525	0.49999	0.88569	0.88057	0.84474	13.5
modelGaussianNB	0.88494	0.99795	0.91601	0.76348	0.69789	0.89357	0.58600	0.88553	0.82817	13.87500
ensembleVoting	0.99616	0.99683	0.85643	0.65	0.8288	0.61446	0.59654	0.90492	0.80552	14.12500
linearModelPAClassifier	0.99578	0.99728	0.8743	0.56664	0.82762	0.65037	0.50359	0.89429	0.78874	14.87500
LogisticRegression	0.99611	0.99747	0.87415	0.74985	0.84505	0.51194	0.49691	0.8919	0.79542	15.37500
modelKNNClassifier	0.96657	0.98507	0.71298	0.81664	0.81994	0.55401	0.8829	0.8954	0.82919	15.75
RandomSampler	0.6485	0.98412	0.86689	0.78318	0.69037	0.94843	0.52545	0.90006	0.79337	16.25
ensembleBaggingClassifier	0.99659	0.9975	0.86758	0.66664	0.7954	0.57046	0.49937	0.89081	0.78554	16.37500
ensembleBOLEClassifier	0.98487	0.99773	0.60686	0.68323	0.91794	0.51916	0.63396	0.85738	0.77514	16.62500
RandomOverSampler	0.64815	0.97099	0.86902	0.71659	0.69395	0.93434	0.51179	0.90065	0.78069	17.5
ensembleVotingClassifier	0.98146	0.99729	0.85641	0.65	0.76748	0.65456	0.59714	0.8836	0.79849	18.12500
ensembleADWINBoostingClassifier	0.99027	0.99796	0.86309	0.63331	0.78965	0.56556	0.50502	0.88765	0.77906	18.12500
linearModelSoftmaxRegression	0.89106	0.89246	0.8676	0.61607	0.79723	0.73217	0.53654	0.89486	0.7785	18.25
ARFClassifier	0.99382	0.96218	0.5	0.5	0.82627	0.56292	0.88007	0.90567	0.76637	18.5
OutputCodeClassifier	0.99674	0.99728	0.85641	0.66667	0.80487	0.5	0.50053	0.89305	0.77694	18.62500
OneVsOneClassifier	0.99673	0.99748	0.85801	0.5516	0.80482	0.50005	0.50062	0.8937	0.76288	18.75
BernoulliNB	0.65908	0.5	0.86974	0.5	0.73235	0.92997	0.53753	0.90475	0.70418	18.75
ensembleAdaBoostClassifier	0.99226	0.96493	0.85371	0.74999	0.7814	0.52209	0.63183	0.88431	0.79757	19.0
ensembleADWINBaggingClassifier	0.99128	0.99728	0.86086	0.66663	0.79042	0.57046	0.49877	0.89001	0.78321	19.5
OneVsRestClassifier	0.99673	0.99706	0.85865	0.6	0.80486	0.49979	0.50099	0.89214	0.76878	19.87500
RandomUnderSampler	0.35183	0.5	0.87424	0.5	0.68711	0.93287	0.51344	0.90731	0.65835	20.0
modelLinearALMAClassifier	0.86796	0.92393	0.82643	0.71584	0.87183	0.63382	0.50708	0.82852	0.77193	20.62500
HardSamplingClassifier	0.81801	0.5	0.75908	0.5	0.74208	0.92821	0.54309	0.90637	0.71211	20.87500
ensembleSRPClassifier	0.84371	0.9622	0.5	0.5	0.75354	0.56292	0.71054	0.91113	0.71800	20.87500
treeExtremelyFastDecisionTreeClassifier	0.97926	0.5	0.5	0.5	0.80884	0.5501	0.60134	0.8823	0.66523	24.0
treeHoeffdingAdaptiveTreeClassifier	0.97635	0.5	0.5	0.5	0.77789	0.54627	0.52061	0.81372	0.64186	26.25
treeHoeffdingTreeClassifier	0.96226	0.5	0.5	0.5	0.73795	0.54618	0.51708	0.79404	0.63219	27.75

Tabela 18: Dados referentes ao *rank* da métrica *ROC Curve*.

Algorithm	Conceptdrift	HTTP	CreditCard	SMTP	Elec2	SMSSpam	Bananas	Phishing	Avg	Rank
ensembleBestModelAverage	0.08983	0.06496	0.20097	0.07901	0.13627	1.62524	0.07201	0.09842	0.29584	2.0
ensembleVoting	0.08919	0.06429	0.2179	0.08325	0.10363	0.9216	0.08431	0.09624	0.20755	2.25
ensembleBestModelThreshold	0.08727	0.05869	0.21019	0.80996	0.13725	1.62222	0.0769	0.09607	0.2962	2.37500
ensembleBestModel	0.08627	0.06228	0.1964	0.07823	0.09724	0.98328	0.08793	0.08916	0.2101	3.37500
ensembleStackingClassifier	0.02042	0.0111	0.0812	0.01574	0.02462	0.60111	0.01857	0.02329	0.09951	5.0
modelKNNClassifier	0.00291	0.00278	0.01208	0.00364	0.00367	0.04988	0.00318	0.00492	0.01038	6.5
ARFClassifier	0.00237	0.00122	0.00200	0.00166	0.00206	0.05559	0.00217	0.00174	0.0086	8.25
forestAMFClassifier	0.00341	0.0008	0.03903	0.00161	0.00483	0.00408	0.00256	0.00204	0.00729	8.75
ensembleBOLEClassifier	0.00217	0.00075	0.00354	0.00100	0.00268	0.02504	0.00178	0.00312	0.00501	8.87500
ensembleLeveragingBaggingClassifier	0.00102	0.00076	0.00209	0.0012	0.00105	0.01757	0.00093	0.0011	0.00321	10.75
ensembleSRPClassifier	0.00121	0.0006	0.00188	0.00069	0.00124	0.0204	0.00084	0.00085	0.00346	11.25
ensembleAdaBoostClassifier	0.00104	0.00056	0.0022	0.00078	0.00119	0.00606	0.00069	0.00116	0.00171	11.87500
ensembleVotingClassifier	0.00061	0.00047	0.00113	0.00043	0.00064	0.00437	0.00054	0.00059	0.0011	14.37500
ensembleADWINBoostingClassifier	0.00054	0.00042	0.0009	0.0005	0.00053	0.00374	0.00047	0.0005	0.00095	15.25
HardSamplingClassifier	0.00042	0.00036	0.00064	0.00035	0.00044	0.02739	0.00042	0.00042	0.0038	16.12500
ensembleADWINBaggingClassifier	0.00052	0.0004	0.00076	0.00044	0.00049	0.00372	0.00044	0.00046	0.0009	16.25
OutputCodeClassifier	0.00033	0.00028	0.00064	0.00038	0.00038	0.00812	0.00034	0.00036	0.00135	18.0
treeExtremelyFastDecisionTreeClassifier	0.00057	0.00016	0.00022	0.00014	0.00074	0.03759	0.00029	0.00179	0.00519	18.37500
treeHoeffdingAdaptiveTreeClassifier	0.00051	0.00025	0.0006	0.00026	0.00049	0.00593	0.00035	0.00036	0.00109	18.375
ensembleBaggingClassifier	0.00037	0.00029	0.00068	0.00034	0.00037	0.00327	0.00033	0.00037	0.00075	19.0
BernoulliNB	0.0003	0.00027	0.00046	0.00025	0.00029	0.01432	0.00027	0.00028	0.00206	19.75
RandomOverSampler	0.00026	0.00023	0.00051	0.00022	0.00026	0.01427	0.00023	0.00025	0.00203	20.5
RandomSampler	0.00022	0.00018	0.00033	0.00018	0.00022	0.00854	0.00021	0.00022	0.00126	22.375
modelGaussianNB	0.00021	0.00018	0.00049	0.00018	0.00026	0.00069	0.00023	0.00022	0.00031	23.87500
RandomUnderSampler	0.00022	0.00016	0.00025	0.00016	0.00021	0.00629	0.0002	0.00021	0.00096	24.5
OneVsRestClassifier	0.00018	0.00014	0.0003	0.00015	0.00019	0.00235	0.00016	0.00017	0.00046	26.25
treeHoeffdingTreeClassifier	0.00018	0.00012	0.00038	0.00012	0.0002	0.00354	0.00014	0.00016	0.00061	27.0
LogisticRegression	0.00018	0.00016	0.00028	0.00014	0.00017	0.00179	0.00016	0.00017	0.00038	27.12500
OneVsOneClassifier	0.00016	0.00011	0.00025	0.00012	0.00016	0.00145	0.00015	0.00015	0.00032	29.37500
linearModelSoftmaxRegression	0.00016	0.00012	0.00031	0.00012	0.00015	0.00044	0.00013	0.00015	0.0002	29.75
Perceptron	0.00014	0.00012	0.00024	0.00012	0.00014	0.00144	0.00013	0.00014	0.00031	30.62500
linearModelPAClassifier	0.00014	0.00012	0.00028	0.00012	0.00014	0.0004	0.00013	0.00014	0.00018	31.25
modelLinearALMAClassifier	0.00012	0.0001	0.00028	0.0001	0.00013	0.00075	0.00013	0.00013	0.00022	31.62500

Tabela 19: Dados referentes ao *rank* do tempo de aprendizado e predição.