Algoritmo de *Ensemble* para Classificação em Fluxo de Dados com Classes Desbalanceadas e Mudanças de Conceito

Douglas Amorim de Oliveira[1],
Karina Valdivia Delgado[1], **Marcelo de Souza Lauretto**[1]

[1]School of Arts, Sciences and Humanities – University of São Paulo
Ermelino Matarazzo – SP – Brazil.

{douglas.amorim.oliveira, kvd, marcelolauretto}@usp.br

Abstract. *With the exponencial growth in data generation observed in the last decades, performing classification tasks on such data poses several challenges. These datasets are sometimes imbalanced in terms of their classes and changes in the formation of classes may occur over time, called concept drift. Among the algorithms aimed to address these problems, the Kappa Updated Ensemble (KUE) has presented good performance in data stream with concept drift. As its original formulation is not designed for imbalanced classes, this paper proposes modifications to KUE in order to make it more robust and adherent to the scenario of imbalanced datasets.In numerical experiments on eight datasets with different rates of imbalance, the modified KUE outperformed the original version in five datasets and yielded statistically equivalent performance in the remaining three. These results are promising and motivate further developments for this approach.*

Resumo. *Com o crescimento exponencial na geração de dados observado nas últimas décadas, a realização de tarefas de classificação sobre esses dados apresenta diversos desafios. Estes conjuntos de dados, por vezes, não são balanceadas quanto às suas classes e podem ocorrer alterações da formação das classes ao longo do tempo, chamadas de mudança de conceito. Dentre os algoritmos que visam solucionar esses problemas, o Kappa Updated Ensemble (KUE) tem apresentado bom desempenho em fluxo de dados com mudança de conceito. Como sua formulação original não é projetada para classes desbalanceadas, neste trabalho foram realizadas modificações no KUE afim de torná-lo mais robusto e aderente ao cenário de desbalanceamento nas bases de dados. Em experimentos realizados sobre oito conjuntos de dados com diferentes taxas de desbalanceamentos, o KUE modificado superou a versão original em cinco conjuntos de dados e produziu desempenho estatisticamente equivalente nos três restantes. Estes resultados são promissores e motivam novos desenvolvimentos para esta abordagem.*

1. Introdução

Com o crescimento exponencial na geração de dados observado nas últimas décadas, a realização de tarefas de classificação sobre esses dados apresenta diversos desafios. Com isso, um campo de estudos que vem crescendo é o de tratamento destes dados, gerados continuamente, para seu uso em modelos de aprendizado de máquina.

Neste artigo abordamos o problema de classificação supervisionada, definida como a tarefa de aprender uma função alvo que mapeie cada vetor de atributos x para um

dos rótulos de classes y pré-determinados [Tan et al. 2009]. Em aplicações reais de classificação, é comum a ocorrência de um alto desequilíbrio nas proporções das classes do conjunto de treinamento, levando diversos pesquisadores a buscarem abordagens para lidar com esse problema [Han et al. 2005, Kubat et al. 1997, Weiss 2004]. Além do desbalanceamento quanto às suas classes podem ocorrer alterações, chamadas de mudança de conceito (em inglês *concept drift*), da formação das classes ao longo do tempo.

Outro cenário desafiador é o de fluxo de dados, que podem ser entendidos como instâncias de dados geradas em alta velocidade e que chegam de forma contínua, trazendo desafios aos sistemas computacionais para realizar seu armazenamento e processamento [Gaber 2012]. Quando se trabalha com fluxo de dados, pela própria natureza contínua da chegada de dados, pode-se ter exposição aos cenários de mudança de conceito e também ao desbalanceamento dos dados, seja pelas características intrínsecas da base ou por mera casualidade de uma janela temporal específica.

Para atuar com esse cenário, de aprendizado em fluxo de dados, diversos autores propuseram várias soluções, tendo grande destaque na área as abordagens de *ensemble* [Krawczyk et al. 2017, Cano and Krawczyk 2020, Gomes et al. 2017]. Por dispor de múltiplos especialistas dentro do conjunto, além da possibilidade de realizar aprendizado incremental nos seus modelos internos e modificações na formação da composição do *ensemble* ao longo do tempo, esta abordagem tem se mostrado promissora.

Uma implementação que demonstrou resultados robustos para o cenário de aprendizado em fluxo de dados com mudanças de conceito é o Kappa Updated Ensemble (KUE) [Cano and Krawczyk 2020]. Implementando o algoritmo Very Fast Decision Tree (VFDT) como base para seus especialistas e criando políticas de atualização dos modelos, abstenção de votos e análise com a estatística Kappa, o KUE trouxe resultados superiores a outros algoritmos estado-da-arte [Cano and Krawczyk 2020]. O KUE, porém, não foi arquitetado para otimizar seu funcionamento com fluxos de dados que apresentem classes desbalanceadas.

Assim, neste artigo é proposta uma evolução no Kappa Updated Ensemble, adicionando uma etapa de reamostragem no pré processamento, abordagem a nível de dados para lidar com desproporção em bases de dados, a fim de obter um novo *ensemble* mais robusto e com maior desempenho no cenário de classificação em fluxo de dados com classes desbalanceadas e mudanças de conceito.

Este artigo tem a seguinte organização: a Seção 2 apresenta os fundamentos teóricos essenciais para o desenvolvimento deste trabalho, sendo eles: fluxo de dados, mudança de conceito, *ensemble* e bases de dados com classes desbalanceadas. Na Seção 3 é apresentada a modificação realizada no Kappa Updated Ensemble. Na Seção 4 são apresentados os resultados experimentais. Na Seção 5 têm-se a conclusão com considerações finais.

2. Fluxo de Dados

Segundo [Gaber 2012], fluxos de dados (em inglês *Data Streams*) podem ser entendidos como instâncias de dados geradas em alta velocidade e que chegam de forma contínua, trazendo desafios aos sistemas computacionais para realizar seu armazenamento e processamento. Se analisados de maneira eficiente, porém, são uma fonte importante de informação para o auxílio na tomada de decisão em tempo real.

Fluxo de dados pode ser definido como uma sequência $S = <S_1, S_2, ..., S_n, ...>$, em que cada S_j é um conjunto de instâncias de tamanho $N \geq 1$, sendo o cenário em que $N = 1$ chamado de aprendizado *online* [Gaber 2012]. Via de regra, as instâncias são independentes e geradas aleatoriamente por uma distribuição estacionária D_j.

Podemos definir cada elemento do fluxo de dados como na Equação 1. Um elemento $p_j(\mathbf{x}, y)$ é uma distribuição conjunta da $j - ésima$ instância, definida pelo espaço $d - dimensional$ de espaço de características e pertencendo à classe y. Cada instância do fluxo de dados é independente e aleatoriamente gerado por uma distribuição estacionária [Cano and Krawczyk 2020].

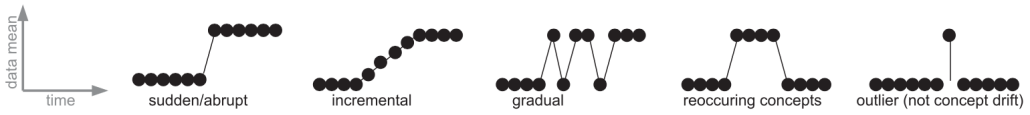
$$S_j \sim p_j(x^1, x^2, ..., x^d, y) = p_j(\mathbf{x}.y) \tag{1}$$

2.1. Mudança de Conceito em Fluxo de Dados

Em cenários da vida real temos que a distribuição D_j é sujeita a mudanças ao longo do tempo, em que as características e definições do fluxo de dados evoluem. Este fenômeno é chamado de mudança de conceito (em inglês *concept drift*) [Brzezinski and Stefanowski 2013a, Gama et al. 2014, Webb et al. 2016]. Ou seja, em cenários em que tem-se uma Base de Dados $S = <S_1, S_2, ..., S_n, ...>$, gerada por uma distribuição D, se $S_j \rightarrow S_{j+1}$ com $D_j = D_{j+1}$, tem-se o que chamamos de um base de dados estacionária. Quando este comportamento sofre alteração ao longo do tempo é o que chamamos de mudança de conceito.

Segundo [Gama et al. 2014], as mudanças de conceito no padrão de formação da distribuição das classes podem ser segmentadas em 4 categorias de ocorrências: repentina, incremental, gradual ou recorrente, como visto na Figura 1.

Figura 1. Tipos de Mudanças de Conceito



[Gama et al. 2014]

O tratamento de mudança de conceito em fluxos de dados pode ser feito de forma implícita ou explícita. Na abordagem implícita, espera-se que o *ensemble*, ou seus especialistas internos, adaptem-se naturalmente à mudança de conceito. Desta forma, algoritmos de classificação que dão pesos maiores às instâncias mais recentes tendem a, com tempo suficiente, tornar irrelevantes as instâncias que ocorreram antes da mudança de conceito, adaptado-se assim ao novo cenário. Outras formas de lidar de maneira implícita com a mudança de conceito seriam: (i) no nível de cada especialista, utilizar de janelas deslizantes para que o algoritmo treine sempre com um conjunto mais recente de dados [Zhang et al. 2016], eventualmente removendo da janela as instâncias que não representam a nova distribuição de formação dos dados; e (ii) no nível de *ensemble*, realizar inicialização paralela de novos especialistas de forma contínua, treinando com as novas instâncias de dados que chegam ao *ensemble*, e definir políticas de substituição de especialistas, desta forma removendo eventualmente os especialistas que apresentem pior desempenho por estar com o viés dos dados pré-mudança de conceito.

Na abordagem explícita de tratamento de mudança de conceito, são criados mecanismos dedicados, chamados detectores, externos ao especialista ou ao *ensemble*, com a responsabilidade de detectar a ocorrência de uma mudança de conceito [Barros and Santos 2018]. Sua intenção é agir de forma tempestiva no *ensemble* quando os dados vierem a apresentar uma mudança de conceito. Uma implementação desta abordagem se baseia em treinar um algoritmo de classificação, externo ao *ensemble*, com a tarefa de classificar se o fluxo de dados corrente apresenta ou não uma mudança de conceito e, se for identificada a mudança de conceito, sinalizar ao *ensemble* para que este possa implementar suas políticas de controle, como a criação de novos especialistas para substituir os anteriormente contidos no *ensemble*.

Na abordagem explícita temos como vantagem a existência de mecanismos dedicados à detecção de mudança de conceito, que pode conseguir identificar de forma mais eficiente a ocorrência desse fenômeno. Com isso, é esperado que neste cenário o *ensemble* se adapte mais rapidamente ao novo cenário pós mudança de conceito, por existir uma sinalização que o leva a ativar políticas de controle. Já na abordagem implícita, temos como vantagem a simplificação tanto de implementação quanto de custo computacional de manutenção do *ensemble*. Apesar de esta abordagem só garantir que a adaptação do ensemble ocorra após um número suficientemente grande de novas instâncias, apresenta o benefício de não gerar sobrecarga de computação dedicada a executar um mecanismo externo ao *ensemble* focado somente nesta tarefa.

2.2. Classificação com *Ensemble* em Fluxo de Dados

As abordagens baseadas em *Ensemble* tem ganhado protagonismo em problemas de classificação com fluxo de dados. Este protagonismo é explicado por sua capacidade de inicialização flexível do *ensemble*, alterar a relevância e participação de seus especialistas internos de maneira dinâmica e por dispor de mecanismos implícitos para incorporar novas informações com a exposição a novas observações de dados [Krawczyk et al. 2017, Cano and Krawczyk 2020, Gomes et al. 2017].

Uma condição para que um *ensemble* de classificadores gere resultados superiores aos dos especialistas, de maneira isolada, é a de que seus componentes sejam diversos [Hansen and Salamon 1990]. Esta característica é favorecida com a exposição às novas instâncias, oriundas do fluxo de dados.

Para lidar com os *ensembles* em fluxo de dados pode-se adotar três abordagens, de maneira isolada ou combinadas [Cano and Krawczyk 2020]:

- **Modificações de Relevância:** essa abordagem se concentra em modificar os pesos atribuídos aos classificadores, na sua influência no resultado final da classificação pelo *ensemble*, afim de refletir suas competências atuais no fluxo de dados [Kolter and Maloof 2007, Ren et al. 2018, Cano and Krawczyk 2020];
- **Substituições Dinâmicas:** essa abordagem se concentra na substituição dinâmica dos especialistas que compõem o *ensemble*. Ao processar novos lotes de instâncias, 1 ou k classificadores novos são treinados externamente ao *ensemble* para que possam ser adicionados ao *ensemble* [Brzezinski and Stefanowski 2013b]. Se o número de especialistas configurado para o *ensemble* for superado, então um mecanismo de poda é aplicado para remover especialistas que estejam apresentando o desempenho inferior [Cano and Krawczyk 2020].

- **Atualização do *Ensemble* em Tempo Real:** essa abordagem se concentra na atualização dos classificadores em tempo real, sendo atualizados à medida que novas instâncias são apresentadas ao *ensemble* [Pietruczuk et al. 2017, Zhai et al. 2017, Pesaranghader et al. 2018, Cano and Krawczyk 2020].

Estas abordagens, de forma combinada, fazem com que o *ensemble* disponha de adaptação natural para lidar com mudanças de conceito e incremente sua diversidade ao longo do fluxo de dados. Ao ocorrer uma mudança de conceito, por exemplo, a propriedade 3 (atualização em tempo real) fará com que os classificadores busquem incorporar esta nova característica em suas tomadas de decisão. Ainda que não seja uma incorporação eficiente, pela propriedade 1 têm-se novos modelos sendo treinados sob a nova distribuição de formação dos dados e, pela propriedade 2, podendo substituir os especialistas que não estejam desempenhando de forma eficiente nos novos dados.

2.3. Fluxo de Dados com Classes Desbalanceadas

Na classificação em fluxos de dados, por apresentar a característica de ser exposto a um conjunto reduzido de dados, tem-se com frequência a ocorrência de dados desbalanceados. Este desbalanceamento pode ocorrer tanto por uma natureza intrínseca ao escopo de dados a ser tratado, como em uma base de dados médica que dispõe de ocorrências pontuais de uma rara doença, como também podem ocorrer pela mera casualidade de o lote de dados sendo processado não dispor de diversidade representativa de instâncias de cada classe.

Para lidar com o cenário de classes desbalanceadas no aprendizado de máquina em fluxo de dados podemos trabalhar com 3 abordagens [Krawczyk 2016]:

- **Métodos no nível dos dados:** consiste em modificar o conjunto de exemplos para fazer um balanceamento das classes;
- **Métodos no nível dos algoritmos:** consiste em modificar os algoritmos de aprendizado existentes para mitigar o viés para as classes majoritárias;
- **Métodos híbridos:** combina as estratégias dos dois métodos acima citados.

3. Kappa Updated *Ensemble* com Pré Processamento

Kappa Updated Ensemble (KUE) [Cano and Krawczyk 2020] é um *ensemble* desenhado para atuar nas tarefas de classificação com aprendizado de máquina em fluxos de dados, com mudança de conceito. Foi idealizado para trabalhar com o escopo de bases de dados que apresentam uma proporção equilibrada de instâncias de cada classe, não tendo como objetivo especializar a abordagem para problemas em cenários com classes desbalanceadas.

O algoritmo KUE faz uso das 3 abordagens citadas na Seção 2.2 para *ensembles*. O item 1 é contemplado pela política de abstenção de votos, que despreza os especialistas que apresentam $Kappa < 0$. O item 2 é implementado com o treinamento de especialistas externos ao *ensemble* e, posteriormente, comparando-os com os internos para possíveis substituições, quando os externos obtiverem $Kappa$ superior ao de especialistas internos. O item 3, por fim, é contemplado com a atualização dos especialistas internos a cada novo lote de dados S_i que chega ao *ensemble*.

O KUE recebe como parâmetros de entrada um fluxo de dados S, o número f de características que as instâncias contemplam, o número k de componentes do *ensemble* e o

número q de novos componentes a serem treinados por fora do *ensemble*, que servirão para substituir os componentes que estiverem com desempenho inferior. Este desempenho é auferido pela métrica obtida com a estatística Kappa [Cohen 1960]. Por padrão, o número de componentes do *ensemble* é o de $k = 10$ especialistas e, para cada lote de dados processados pelo *ensemble*, é treinado um novo especialista externo ($q = 1$), que pode vir a substituir um interno.

Por lidar com fluxo de dados, um requisito que o *ensemble* tem que tratar é a capacidade de fazer o processamento dos dados que chegam de maneira contínua, e possivelmente volumosa, de maneira eficiente. Para isso, todos os especialistas que compõem o *ensemble* executam o algoritmo Very Fast Decision Tree [Hulten et al. 2001].

Para garantir a diversidade no *ensemble*, necessária para uma bom desempenho da abordagem [Hansen and Salamon 1990], o algoritmo treina seus especialistas com subespaços distintos. Cada modelo, na sua inicialização, tem sorteado de maneira aleatória um número r, com r menor ou igual ao número de características das instâncias, que será a quantidade de características que ele usará como insumo no seu treinamento e classificação. Além de ter um número sorteado para cada modelo, as próprias r características em si também são aleatoriamente escolhidas.

Como saída do KUE temos o *ensemble* com os especialistas treinados, o conjunto de subspaços de características para cada um dos especialistas e a estatística Kappa atribuída aos componentes.

Além do comportamento de inicialização, atualização, treinamento e substituição de componentes, o KUE realiza a seleção da classificação do *ensemble* via método de voto majoritário. Buscando melhorar o desempenho e evitar influência de classificação de modelos com baixo desempenho, o algoritmo proposto por [Cano and Krawczyk 2020] tem uma política de abstenção de voto. Neste caso, especialistas que disponham da estatística Kappa abaixo de 0, em um intervalo que vai de -100 a 100, não tem seu voto considerado para a saída do *ensemble*.

Com este conjunto de especificações, o KUE se mostrou um algoritmo robusto para lidar com aprendizado de máquina em fluxos de dados, comparado com um conjunto de 15 métodos [Cano and Krawczyk 2020] de *ensemble* considerados estado-da-arte. Apesar da robustez, porém, o Kappa Updated Ensemble não faz uso de técnicas específicas para lidar com bases de dados desbalanceadas.

Para lidar com esse problema de classificação em fluxos de dados com classes desbalanceadas, este trabalho se propôs a realizar uma expansão no KUE, adicionando uma etapa de pré processamento, um método a nível de dados para lidar com a desproporção das classes, com a intenção de criar um *ensemble* mais aderente para esta natureza de desproporção de classes nas bases de dados.

No algoritmo disposto no pseudocódigo 1 estão destacadas em cor azul as modificações que foram realizadas no KUE adaptado com o uso de técnicas de rebalanceamento no pré processamento.

Ao receber um lote de dados S_i para ser processado, o algoritmo percorre o mesmo contabilizando a quantidade de ocorrências por cada classe e, com a proporção encontrada, realiza uma sobreamostragem da classe minoritária substituindo ocorrências

da majoritária por cópias da minoritária buscando uma proximidade do equilíbrio, obtendo o novo conjunto S_i'. Após essa reamostragem, o restante do funcionamento do *ensemble* se mantém como o original. Com esta modificação buscou-se reduzir o desbalanceamento das classes nos dados que servem de insumos para o treinamento dos especialistas do *ensemble*.

No Algoritmo 1 o processo de reamostragem dos dados, descrito anteriormente, é disposto na linha 2, resultando no lote de dados S_i'. O lote S_i' é então utilizado nas etapas de: inicialização do ensemble (nas linhas 6, 7 e 8 do algoritmo); atualização do *ensemble* (nas linhas 12, 13 e 14); e no treinamento de novos especialistas externos ao modelo (nas linhas 18, 19 e 20).

Foram realizados experimentos também variando o parâmetro q, que representa o número de especialistas a ser treinados a cada lote de dados para, possivelmente, substituir modelos internos ao do *Ensemble*. O autor utiliza q fixado em 1, enquanto neste artigo são apresentados os resultados para o algoritmo modificado com $q = 1$ e $q = 2$.

As bases de dados rebalanceadas são utilizadas apenas no momento de treinamento do modelo. Para a etapa de testes são utilizados os conjuntos originais de dados.

4. Resultados Experimentais

Nesta seção são mostradas as análises e resultados obtidos com o código desenvolvido para execução no ambiente Massive Online Analysis (MOA). Os experimentos foram executados em Java 17.0.2 com IntelliJ IDEA 2021.3.2 (Community Edition) no sistema operacional macOS Monterey.cp, processador Intel Core i7 6-Core 2.6GHZ, 16GB de memória RAM 2667 MHz DDR4, placa gráfica AMD Radeon Pro 5300M 4 GB e Intel UHD Graphics 630 1536 MB.

São comparados os desempenhos, mensurados pelas métricas Acurácia, F1-Score e Kappa, dos seguintes *ensembles* submetidos às bases de dados desbalanceadas:

1. Kappa Update Ensemble com $q = 1$;
2. Kappa Update Ensemble modificado com o uso de técnicas de rebalanceamento no pré processamento (KUE_R) e com $q = 1$;
3. Kappa Update Ensemble modificado com o uso de técnicas de rebalanceamento no pré processamento (KUE_R) e com $q = 2$.

Lembrando que o parâmetro q representa a quantidade de especialistas treinados externamente ao *ensemble*, para substituição de especialistas internos utilizando a estatística Kappa como critério de substituições.

Foram utilizadas 8 bases de dados, sendo 5 delas geradas no MOA de maneira sintética e 3 reais. As bases de dados artificiais foram geradas utilizando os parâmetros da Tabela 1. As bases reais utilizadas estão contidas na Tabela 2.

4.1. Análise na Base de Dados RandomTreeGenerator-IR

Nesta seção são apresentados os resultados das métricas Acurácia, Estatística Kappa e F1-Score obtidos ao executar os 3 algoritmos na base de dados sintética RandomTreeGenerator-IR com IR 1:19, divididos em lotes com 5.000 instâncias e instanceRandomSeed 42.

Algorithm 1 KUE com rebalanceamento (KUE$_R$).

Input: S: data stream, f : number of features, k: number of ensemble components, q: number of new components to train
Output: ε : ensemble of k γ classifiers,
φ: subspace of features for each of the k components,
κ: Kappa for each of the k components

for $S_i \in \{S_1, \ldots, S_n\}$ **do**
  $S'_i \leftarrow S_i$ with oversampling of minority class and undersampling of majority class to 50% of chunk size
  if S_1 **then** ▷ Ensemble initialization
    for $j \in \{1, \ldots, k\}$ **do**
      $r \leftarrow$ random integer with uniform probability $[1, f]$
      $\varphi_j \leftarrow$ r-dimensional random subspace in S'_1 where instances are weighted according to $Poisson(1)$
      $\gamma_j \leftarrow$ new classifier on $\varphi_j(\ S'_1\ )$
      $k_j \leftarrow$ compute Kappa of γ_j on $\varphi_j(\ S'_1\ )$
    end for
  else ▷ Ensemble model update
    for $j \in \{1, \ldots, k\}$ **do**
      $\varphi_j \leftarrow$ instances in S'_i are weighted according to $Poisson(1)$ keeping the r-dimensional subspace
      $\gamma_j \leftarrow$ incremental train of γ_j on $\varphi_j(\ S'_i\ )$
      $k_j \leftarrow$ compute Kappa of γ_j on $\varphi_j(\ S'_i\ )$
    end for
    for $\{1, \ldots, q\}$ **do** ▷ Train new components
      $r \leftarrow$ random integer with uniform probability $[1, f]$
      $\varphi' \leftarrow$ r-dimensional random subspace in S'_i where instances are weighted according to $Poisson(1)$
      $\gamma' \leftarrow$ new classifier on $\varphi'(\ S'_i\ )$
      $k' \leftarrow$ compute Kappa of γ' on $\varphi'(\ S'_i\ )$
      if $k' > k_{min(k)}$ **then** ▷ Replace weakest $\gamma \in \varepsilon$
        $\varphi_{min(k)} \leftarrow \varphi'$
        $\gamma_{min(k)} \leftarrow \gamma'$
        $k_{min(k)} \leftarrow k'$
      end if
    end for
  end if
end for

Tabela 1. Parâmetros para a geração das bases de dados artificiais

Base de Dados	Instâncias	Atributos	Classes	Imb.Ratio
RandomTreeGenerator-IR	2.000.000	10	2	1:19
RandomTreeGenerator-IR	2.000.000	10	2	1:9
RandomRBFGenerator-IR	2.000.000	10	2	1:9
AssetNegGenerator-IR	1.000.000	5	2	1:9
SEAGenerator-IR	1.000.000	3	2	1:9

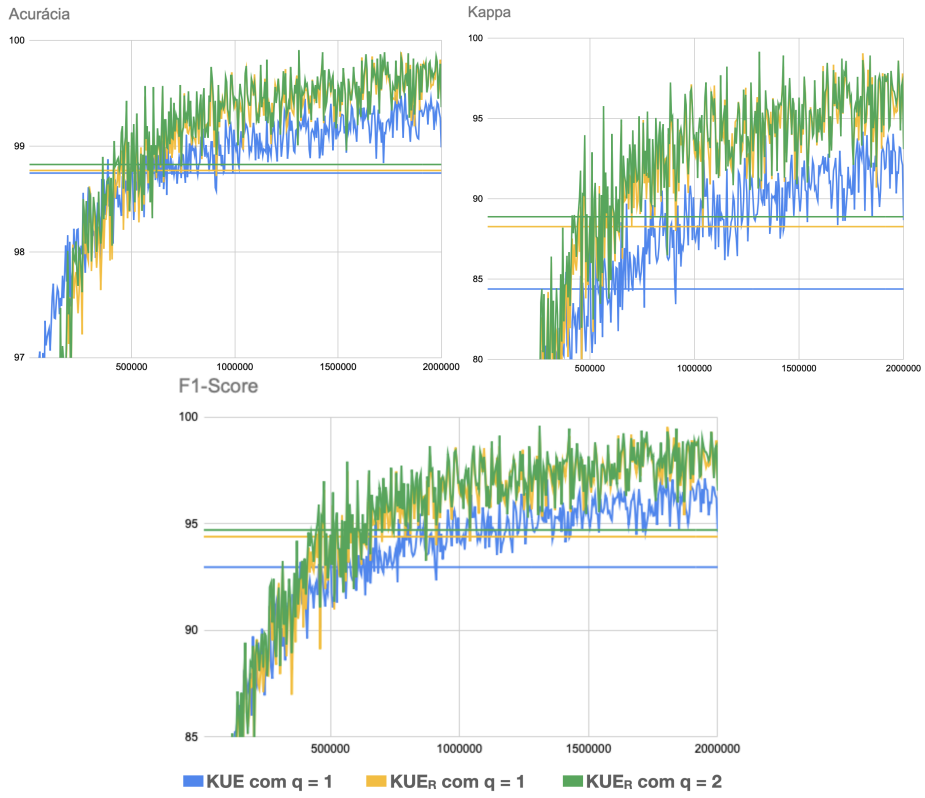
Tabela 2. Características das bases de dados reais

Base de Dados	Instâncias	Atributos	Classes	Imb.Ratio
poker-lsn-1-2vsAll-IR	1.000.000	11	2	1:13
BNG_bridges-1vsAll	1.000.000	13	2	1:7
covtypeNorm-1-2vsAll	1.000.000	5	2	1:7

As imagens apresentadas nesta seção, que ilustram os desempenhos dos algoritmos sob as métricas supracitadas, contém tanto o gráfico da evolução do valor absoluto da métrica, ao longo do experimento, quanto uma linha de média para cada algoritmo. As linhas que apresentam as médias, paralelas ao eixo X, são obtidas pela média aritmética simples dos valores de acurácia ao longo de todo o experimento, com o intuito de apresentar visualmente um comparativo da diferença de desempenho dos *ensembles* no experimento completo.

Na Figura 2 são exibidas a evolução da Acurácia, da estatística Kappa e F-1 Score. É realizado um recorte no eixo Y para melhor visualização das diferenças de desempenho dos algoritmos.

Figura 2. Evolução da Acurácia, Kappa e F1-Score na Base de Dados RandomTreeGenerator-IR.



Pode-se notar na Figura 2 que, de início, a acurácia do KUE original se encontra superior ao KUE com amostragem. Isso pode se dar por, em um momento com poucos dados vistos, o descarte de instâncias da classe majoritária na nossa expansão representar perdas significativas de informação que impactam o modelo. Nota-se que em determinado momento as acurácias se encontram e andam próximas, assim como a Kappa e F1-Score.

Em um momento onde o volume de dados apresentados ao modelo passa a ser mais volumoso, por volta das 500000 instâncias, o gráfico do KUE com amostragem, para as 3 métricas, começa a dominar a do KUE original, se mantendo consistentemente com desempenho superior.

Uma técnica que pode-se empregar para lidar com esse desbalanceamento a nível de dados sem perder informações novas da classe majoritária seria a utilização de janelas deslizantes por cada classe, com igual tamanho, possivelmente resolvendo esse cenário inicial com baixo volume de dados.

4.2. Resultados por F1-Score para as 8 Bases de Dados

Na Tabela 3 são apresentados os resultados do F1-Score. O F1-Score em uma dada base de dados específica foi calculado utilizando a média de 10 execuções de cada algoritmo sobre o lote final de instâncias da base. Além da média foi calculado o desvio-padrão e realizado um teste de T Student de cada algoritmo contra o algoritmo com maior desempenho absoluto, com a hipótese nula de que ambos os resultados são equivalentes, com $\alpha = 0.05$.

Na Tabela 3 as células dos algoritmos que apresentam o maior desempenho absoluto da média de F1-Score, para cada base de dados, é destacado com a cor cinza escura. Os algoritmos que não apresentaram diferença estatística com relação ao melhor resultado, pelo teste de T Student, são destacadas com a cor cinza clara.

Tabela 3. Média e desvio-padrão do F1-Score por base de dados.

		KUE $q = 1$	KUE$_R$ $q = 1$	KUE$_R$ $q = 2$
#	Base de Dados	Média (d.p.)	Média (d.p.)	Média (d.p.)
1	RandomTreeGenerator-IR 1:19	97.63 (0.26)	98.55 (0.38)	98.61 (0.64)
2	RandomTreeGenerator-IR 1:9	94.61 (0.30)	96.72 (0.24)	96.55 (0.19)
3	RandomRBFGenerator-IR 1:9	94.32 (0.27)	94.14 (0.56)	94.37 (0.53)
4	AssetNegGenerator-IR 1:9	86.80 (0.46)	87.46(0.00)	87.46 (0.00)
5	SEAGenerator-IR 1:9	77.91 (0.60)	80.39 (0.11)	80.42 (0.08)
6	poker-lsn-1-2vsAll 1:13	83.54 (2.53)	82.50 (2.93)	82.45 (2.86)
7	BNG_bridges-1vsAl 1:7	90.10 (0.98	89.89 (0.37)	89.79 (0.70)
8	covtypeNorm-1-2vsAll 1:7	72.83 (4.63)	77.64 (6.05)	80.13 (2.74)

Analisando os resultados apresentados na Tabela 3 visualiza-se que as mudanças propostas no algoritmo trouxeram ganho de desempenho, mensurado pelo F1-Score. Das 8 bases, em 2 delas (25%) o KUE$_R$ com $q = 1$ obteve maior desempenho absoluto. Em 5 das 8 bases (62,5%), o KUE$_R$ com $q = 2$ obteve maior desempenho absoluto. Além de os *ensembles* com reamostragem obterem desempenho absoluto superior ao Kappa Updated Ensemble em 75% das bases, nas 25% demais os algoritmos não tiveram diferença estatística relevante. Ainda que o KUE$_R$ com $q = 2$ tenha obtido maior valor nominal na maioria das bases, nota-se que o KUE$_R$ com $q = 1$ não foi estatisticamente inferior em nenhum dos conjuntos de dados, o que indica que o responsável pelo melhor desempenho de F1-Score no KUE$_R$ frente ao KUE original se deve de fato à reamostragem, e não à inclusão de mais um classificador. Outra observação é a de que, além de os algoritmos modificados apresentarem resultados superiores com diferenças estatisticamente significantes, seus

resultados numéricos e absolutos apresentaram ganhos expressivos, como pode-se notar nos conjuntos de dados 2, 5 e 8.

5. Conclusão

Este trabalho apresentou uma proposta de modificação no Kappa Updated Ensemble (KUE), para lidar com o problema de classificação em fluxo de dados, com classes desbalanceadas. Com os experimentos, realizados em 8 bases de dados, tanto sintéticas quanto reais, e seus resultados, notamos que a proposta se mostra promissora e apresentam um desempenho superior ao KUE.

Como trabalhos futuros, serão realizadas evoluções no *ensemble* buscando torná-lo mais robusto e aderente ao escopo de classificação em fluxo de dados, com mudança de conceito e com classes desbalanceadas. Entre as mudanças a serem realizadas estão: (i) adicionar janelas deslizantes, por classe, para lidar com as classes desbalanceadas; (ii) substituir o algoritmo VFDT, dos especialistas do *ensemble*, pelo Extremely Fast Decision Tree [Manapragada et al. 2018], buscando maior qualidade preditiva do *ensemble*.

Agradecimentos

Este trabalho teve o apoio do CEPID-CeMEAI - Centro de Ciências Matemáticas Aplicadas à Indústria, Processo 2013/07375-0, Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP). Teve também o apoio do Centro de Inteligência Artificial C4AI-USP financiado pela FAPESP (Processo 2019/07665-4) e pela IBM.

Referências

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

Brzezinski, D. and Stefanowski, J. (2013a). Classifiers for concept-drifting data streams: evaluating things that really matter. In *ECML PKDD 2013 Workshop on Real-World Challenges for Data Stream Mining, September 27th, Prague, Czech Republic*, pages 10–14. Citeseer.

Brzezinski, D. and Stefanowski, J. (2013b). Reacting to different types of concept drift: The accuracy updated ensemble algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):81–94.

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

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Gaber, M. M. (2012). Advances in data stream mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):79–85.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37.

Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G., and Abdessalem, T. (2017). Adaptive random forests for evolving data stream classification. *Machine Learning*, 106(9):1469–1495.

Han, H., Wang, W.-Y., and Mao, B.-H. (2005). Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer.

Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001.

Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time-changing data streams. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 97–106.

Kolter, J. Z. and Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *The Journal of Machine Learning Research*, 8:2755–2790.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232.

Krawczyk, B., Minku, L. L., Gama, J., Stefanowski, J., and Woźniak, M. (2017). Ensemble learning for data stream analysis: A survey. *Information Fusion*, 37:132–156.

Kubat, M., Matwin, S., et al. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pages 179–186. Citeseer.

Manapragada, C., Webb, G. I., and Salehi, M. (2018). Extremely fast decision tree. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1953–1962.

Pesaranghader, A., Viktor, H., and Paquet, E. (2018). Reservoir of diverse adaptive learners and stacking fast hoeffding drift detection methods for evolving data streams. *Machine Learning*, 107(11):1711–1743.

Pietruczuk, L., Rutkowski, L., Jaworski, M., and Duda, P. (2017). How to adjust an ensemble size in stream data mining? *Information Sciences*, 381:46–54.

Ren, S., Liao, B., Zhu, W., and Li, K. (2018). Knowledge-maximized ensemble algorithm for different types of concept drift. *Information Sciences*, 430:261–281.

Tan, P.-N., Steinbach, M., and Kumar, V. (2009). *Introdução ao datamining: mineração de dados*. Ciência Moderna.

Webb, G. I., Hyde, R., Cao, H., Nguyen, H. L., and Petitjean, F. (2016). Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994.

Weiss, G. M. (2004). Mining with rarity: a unifying framework. *ACM Sigkdd Explorations Newsletter*, 6(1):7–19.

Zhai, T., Gao, Y., Wang, H., and Cao, L. (2017). Classification of high-dimensional evolving data streams via a resource-efficient online ensemble. *Data Mining and Knowledge Discovery*, 31(5):1242–1265.

Zhang, L., Lin, J., and Karim, R. (2016). Sliding window-based fault detection from high-dimensional data streams. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(2):289–303.