UNIVERSIDADE PRESBITERIANA MACKENZIE
PROGRAMA DE PÓS-GRADUAÇÃO EM
ENGENHARIA ELÉTRICA E COMPUTAÇÃO

Márcio Rubbo

Seleção de Protótipos com Mapas-Auto-Organizáveis e Entropia para Sobreposição de Classes e Desbalanceamento de Dados

São Paulo
2019

UNIVERSIDADE PRESBITERIANA MACKENZIE
PROGRAMA DE PÓS-GRADUAÇÃO EM
ENGENHARIA ELÉTRICA E COMPUTAÇÃO

Márcio Rubbo

Seleção de Protótipos com Mapas-Auto-Organizáveis e Entropia para Sobreposição de Classes e Desbalanceamento de Dados

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em ao Programa e Computação da Universidade Presbiteriana Mackenzie como parte dos requisitos para o título de Mestre em Engenharia Elétrica e Computação.

Orientador: Prof. Dr. Leandro Augusto da Silva

São Paulo
2019

R894s

Rubbo, Márcio

Seleção de protótipos com mapas-auto-organizáveis e entropia para sobreposição de classes e desbalanceamento de dados / Márcio Rubbo – São Paulo, 2019.

85 f. : il., 30 cm.

Dissertação (Mestrado em Engenharia Elétrica e Computação) - Universidade Presbiteriana Mackenzie - São Paulo, 2019.

Orientador: Prof. Dr. Leandro Augusto da Silva

Bibliografia: f. 82-85

1. Seleção de protótipos 2.SOM 3. kNN 4.Sobreposição de dados 5.Redução de Dados 6.Desbalanceamento 7.Complexidade de Dados. I. Silva, Leandro Augusto da*, orientador*. II.Título.

CDD 005.3

Bibliotecária Responsável: Maria Gabriela Brandi Teixeira – CRB 8/ 6339

MARCIO RUBBO

SELEÇÃO DE PROTÓTIPOS COM MAPAS-AUTO-ORGANIZÁVEIS E ENTROPIA PARA SOBREPOSIÇÃO DE CLASSES E DESBALANCEAMENTO DE DADOS

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e Computação da Universidade Presbiteriana Mackenzie, como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica e Computação.

Orientador: Prof. Dr. Leandro Augusto da Silva

Aprovado em 09 de agosto de 2019.

BANCA EXAMINADORA

Prof. Dr. Leandro Augusto da Silva
Universidade Presbiteriana Mackenzie

Prof. Dr. Mario Olímpio de Menezes
Universidade Presbiteriana Mackenzie

Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho
Universidade de São Paulo

.

À minha esposa Lindsei, pela compreensão e apoio durante este trabalho, que ocorreu em conjunto do nascimento da nossa querida Ana.

AGRADECIMENTOS

Agradeço a Deus pela inspiração e força durante todos os momentos.

Ao professor Dr. Leandro Augusto da Silva, que me deu a oportunidade deste trabalho e me orientou durante todo seu processo. Sem os seus conselhos e apoio, este projeto não teria sido concluído. Também agradeço a todos por todos os conselhos e conversas, que me permitiram crescer como profissional e os quais levarei comigo a diante.

Aos professores Dr. Mario Olímpio de Menezes e Dr. André Carlos Ponce de Leon Ferreira de Carvalho, que dedicaram tempo a analisar este trabalho, contribuindo com valiosas sugestões que aumentaram significativamente sua qualidade final.

Aos meus colegas Pablo e Victor, que participaram durante essa etapa da minha vida e que contribuíram com discussões e informações preciosas.

À minha amiga Carine, que separou parte do seu tempo para me auxiliar com a revisão e contribuir com a qualidade desta dissertação.

Aos meus pais, Antônio Carlos e Fátima, e ao meu irmão, Daniel, cujos apoio e compreensão durante a busca desse objetivo foram essenciais.

Especialmente a minha amada esposa Lindsei, à qual não tenho palavras para agradecer pela paciência, pelo apoio e pela compreensão que teve para me amparar durante o trabalho, justamente em uma época em que tivemos um grande novo desafio.

E, finalmente, a minha querida filha Ana, cuja alegria e sorrisos fazem tudo valer a pena.

RESUMO

O k vizinhos mais próximos (kNN) é um classificador supervisionado tradicional usado em tarefas de mineração de dados. No entanto, quando usado em aplicações reais, principalmente em uma base de dados com desbalanceamento ou sobreposição de classes, o kNN sofre com problemas na tarefa de classificação dos dados. Neste trabalho, são propostos três métodos de seleção de protótipos usando mapas-auto-organizáveis (SOM) e entropia da informação para aumentar a efetividade do classificador kNN em base de dados nessas condições. Bases de dados artificiais, simulando diferentes condições de sobreposição de dados e desbalanceamento, foram criadas e utilizadas em conjunto com bases de dados públicas para teste dos métodos. Medidas de dados complexos foram usadas para identificar sobreposição de dados e separação das classes nas bases usadas nos experimentos e uma comparação foi realizada com os resultados obtidos. Os métodos, nomeados SOMEntropyHighFilter, SOMEntropyLowFilter e SOMEntropyHighLowFilter, foram capazes de aumentar a eficiência do classificador kNN nas bases de dados artificiais e reais usadas para testes, aumentando a performance em bases de dados desbalanceadas ou com problemas de sobreposição.

Palavras-chave: Seleção de protótipos. SOM. kNN. Sobreposição de classes. Redução de dados. Desbalanceamento. Complexidade de dados.

ABSTRACT

The k nearest neighbor (kNN) is a traditional supervised classifier used in data mining tasks. However, when used in real applications, mainly in a dataset with class imbalance or class overlap, kNN suffers with problems in the task of data classification. In this work, we propose three prototype selection methods using self-organizing maps (SOM) and information entropy to increase the effectiveness of the kNN classifier in datasets with these conditions. Artificial datasets that simulate different conditions of data overlap and data imbalance were created and used together with public datasets to test the methods. Data complexity measures were used to identify data overlap and spatial distribution in the bases used in the experiment and a comparison was made with the results of the methods. The methods, named SOMEntropyHighFilter, SOMEntropyLowFilter and SOMEntropyHighLowFIlter, were able to increase the effectiveness of the kNN classifier in the artificial and real datasets used in the experiment, increasing the accuracy performance from datasets with imbalance or overlap problems.

Keywords: Prototype selection. SOM. kNN. Data class overlap. Data reduction. Data imbalance. Data complexity.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ANOVA	Análise de variância
BMU	*Best Match Unit*
kNN	K vizinhos mais próximos
PG	Geração de protótipos
PS	Seleção de protótipos
SOM	Mapas Auto-Organizáveis de Kohonen
UCI	Universidade da Califórnia Irvine

Sumário

Capítulo 1

Introdução

A classificação de dados pode ser definida como uma tarefa da mineração de dados pela qual se mapeia um exemplar representado por um conjunto de atributos descritivos a uma classe pré-estabelecida (FACELI et al., 2011; CASTRO; FERRARI, 2016; SILVA; SARAJANE; BOSCARIOLI, 2017).

A maioria dos algoritmos de classificação da literatura espera que o conjunto de dados tenha distribuições similares nos exemplares quando segregados por classe (HE; GARCIA, 2009); o problema de desbalanceamento passa a ocorrer quando uma classe tem uma representação superior a outra (HE; GARCIA, 2009). Nesse caso, quando o número de exemplares por classe em um conjunto está desbalanceado, os classificadores, em geral, podem ter problemas de desempenho (HE; GARCIA, 2009; FACELI et al., 2011; LÓPEZ et al., 2013; BRANCO; TORGO; RIBEIRO, 2016).

Outra situação que pode causar problemas no desempenho dos algoritmos de classificação é a sobreposição de classes. A sobreposição ocorre quando uma região do espaço dimensional é compartilhada em uma proporção similar por diferentes classes. Esta situação tem um impacto no desempenho do classificador, que tem uma dificuldade maior para separar as classes. Estudos sobre o assunto foram feitos e indicaram que a sobreposição de classes pode ter um impacto maior na perda do desempenho que o desbalanceamento de classes (PRATI; BATISTA; MONARD, 2004; GARCIA; SANCHEZ; MOLLINEDA, 2007; DENIL; TRAPPENBERG, 2010).

Um classificador que é especialmente vulnerável para os problemas de desbalanceamento e sobreposição de dados é o k vizinhos mais próximos (kNN) (GARCIA; SANCHEZ; MOLLINEDA, 2007). O kNN é um classificador supervisionado, que trabalha

usando a abordagem dos vizinhos mais próximos para classificar um exemplar (COVER; HART, 1967).

Diversos métodos foram propostos para resolver os problemas que afetam o kNN, sendo a redução de dados por meio da geração de uma base processada com protótipos um dos métodos sugeridos. Nesse método, a base processada representa a base original com uma nova distribuição de dados mais adequada ao classificador (TRIGUERO et al., 2012; GARCÍA et al., 2012).

Para organizar diferentes trabalhos que usam a abordagem de protótipos Garcia, Derrac, Cano, Herrera e Triguero publicaram dois artigos definindo uma taxonomia de métodos de geração de protótipos (TRIGUERO et al., 2012; GARCÍA et al., 2012). Nesses trabalhos, os autores compararam diferentes métodos de redução de dados que usam protótipos e que buscam melhorar a classificação e performance do kNN. Os autores dividem os métodos em dois grandes grupos: geração de protótipos e seleção de protótipos. Na geração de protótipos (PG, do inglês *Prototype Generation*), um conjunto de protótipos é usado para gerar um novo conjunto de exemplares artificiais, que é usado numa tentativa de representar eficientemente a distribuição das classes sem afetar a acurácia do classificador (TRIGUERO et al., 2012). O outro grupo é a seleção de protótipos (PS do inglês *Prototype Selection*), que envolve a seleção de um subconjunto de protótipos da base original sem a criação de novos dados artificiais (GARCÍA et al., 2012).

De forma a complementar os métodos de geração de protótipos, técnicas de pré-processamento de dados podem ter diversas estratégias para alterar a distribuição de dados. Dentre os métodos disponíveis na literatura destacam-se métodos randômicos, medidas de distância, técnicas de limpeza, algoritmos evolucionais de alteração e técnicas de agrupamento (BRANCO; TORGO; RIBEIRO, 2016). Analisando-se o uso de técnicas de agrupamento para pré-processamento, considerou-se o uso dos mapas de Kohonen (KOHONEN, 2013). O mapa auto-organizável de Kohonen (SOM, do inglês *Self-Organizing Map*) é um método não supervisionado que agrupa os dados de acordo com sua similaridade (KOHONEN, 2013). A versão padrão do algoritmo organiza os dados, de acordo com sua distância Euclidiana, dentro de nós (neurônios) na forma de uma rede. Dessa maneira, criando um mapa de nós, o algoritmo é processado de forma que os exemplares mais similares se encontrem dentro do mesmo nó ou de nós próximos. O uso do SOM foi escolhido como técnica de agrupamento por permitir a possibilidade de

analisar um mapa de agrupamentos (nós) que demonstram regiões com diferentes graus de similaridade da base de dados.

O objetivo deste trabalho é propor o uso do SOM para agrupar os dados e, por meio do uso da entropia de informação, pré-processar os dados agrupados nos nós do SOM por meio de uma técnica de seleção de protótipos. A hipótese assumida neste trabalho é a de que a remoção dos exemplares deixará as fronteiras de decisão de bases de dados complexas mais suaves, permitindo uma melhora no desempenho do kNN. Alguns trabalhos foram feitos usando SOM para pré-processar os dados em bases desbalanceadas, no entanto, a maior parte desses trabalhos está focada na geração de protótipos (ARABMAKKI; KANTARDZIC, 2017; DOUZAS; BACAO, 2017; MOREIRA; SILVA, 2017).

De maneira a complementar o estudo proposto, para determinar se uma base sofre com problemas que dificultem a classificação, foram pesquisadas medidas que permitam identificar características de uma base, como a sobreposição. Foram encontradas medidas na literatura nomeadas como *medidas de complexidade* (BASU; HO, 2006). Estas medidas permitem analisar uma base de dados antes de se usar um classificador e identificar condições como a sobreposição de dados e a distribuição espacial da base. As *medidas de complexidade* serão utilizadas neste trabalho para avaliar as bases e verificar se a proposta deste trabalho, usando SOM e entropia, é efetiva quanto aos problemas abordados.

A maioria dos métodos de pré-processamento e medidas de avaliação para bases desbalanceadas estão focadas no tratamento de classes binárias (BRANCO; TORGO; RIBEIRO, 2016). Assim, decidiu-se inicialmente por focar os estudos deste trabalho em problemas binários.

O restante do trabalho está dividido da seguinte maneira: no Capítulo 2, abordaremos as bases de mapa-auto organizáveis, seleção de protótipos, entropia e dados complexos que usamos no trabalho; no Capítulo 3 serão demonstrados os métodos propostos; no Capítulo 4 abordaremos a metodologia utilizada, com os resultados sendo discutidos no Capítulo 5. Finalizaremos com as conclusões e trabalhos futuros no Capítulo 6.

Capítulo 2

Fundamentos Teóricos

Neste capítulo, abordam-se os principais fundamentos usados para criação dos algoritmos que serão apresentados no trabalho.

2.1 Entropia

O conceito de entropia da informação proposto por Shannon (1948) permite identificar quanto de incerteza se tem em uma determinada distribuição de probabilidades de dados.

Segundo Shannon, é possível ter uma medida, denominada entropia, que informe quanto de incerteza existe para um processo que tem n diferentes eventos com diferentes probabilidades de ocorrência. Portanto, essa medida permite a comparação entre diferentes processos para determinar quais desses tem a maior incerteza do resultado.

Para o caso de um processo em que todos os eventos são independentes, a entropia pode ser definida como:

$$H = -\sum_{i=1}^{n} p_i * log_2(p_i) \tag{2.1}$$

Onde p_i é a probabilidade de um evento i ocorrer dentro de um processo.

A Equação 2.1 possui algumas características: o caso de $H = 0$ ocorre na situação em que se tem certeza sobre qual evento irá acontecer, ou seja, apenas uma das probabilidades p_i possui valor, com as demais sendo zero. Também pode ser verificado que H tem seu valor máximo igual a $log_2(n)$, que ocorre quando todas as probabilidades dos n eventos são iguais, com probabilidade igual a $1/n$, sendo esse o cenário de maior incerteza do processo (SHANNON, 1948).

Para o caso em que se tem somente dois eventos independentes a equação pode ser simplificada como:

$$H = -p * log_2(p) - q * log_2(q) \tag{2.2}$$

Onde p e q são probabilidades dos eventos dentro de um mesmo conjunto, onde $q = 1 - p$. A variação da entropia para esse caso de dois eventos pode ser verificada na Figura 2.1, em que está desenhada a curva da entropia pela variação da probabilidade p. Nota-se nesta figura que se tem dois valores mínimos com valor 0 nos pontos em que p é 0 ou 1, ou seja, em que se tem total certeza de qual evento irá ocorrer. O ponto máximo, com valor 1, ocorre onde p é 0,5, que representa 50% de probabilidade para cada evento, sendo o ponto de maior incerteza.

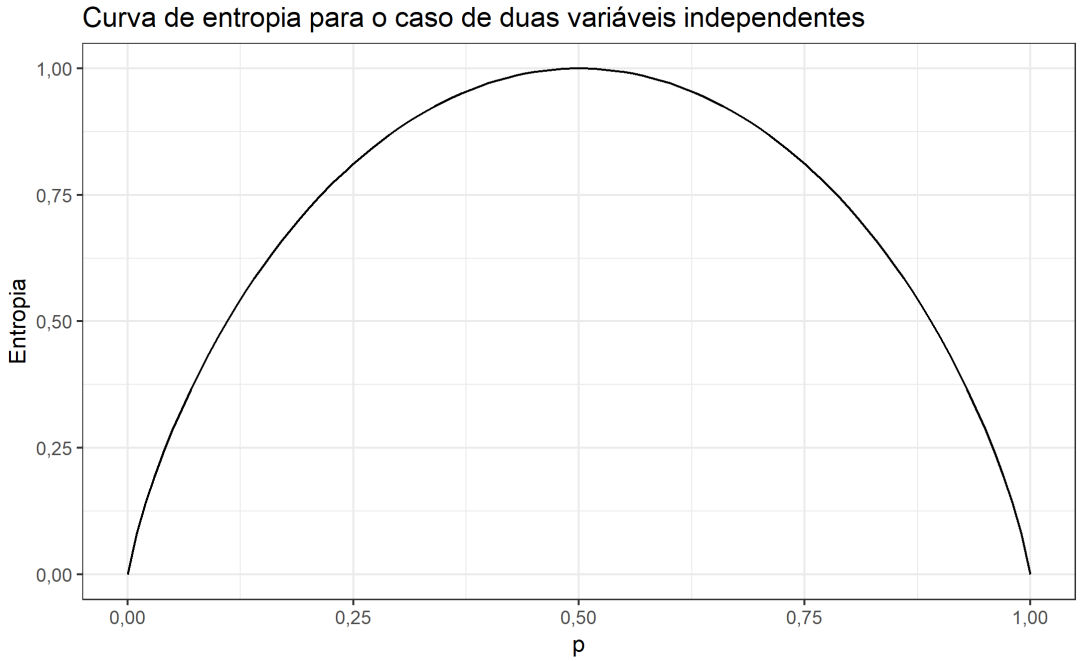


Figura 2.1: Curva de entropia para um caso da probabilidade de dois eventos independentes em um processo. Fonte: adaptado de Shannon (1948).

O conceito de entropia aplicado em uma análise de conjunto de dados pode ser visto como uma medida da impureza ou desordem entre os dados dentro de uma partição em relação a sua classe (SILVA; SARAJANE; BOSCARIOLI, 2017), em que as probabilidades são relacionadas a probabilidade de encontrar uma determinada classe dentro de um subconjunto.

Assim, considerando um subconjunto de dados formado por uma participação da base,

uma entropia alta identifica um caso em que existe uma distribuição em proporções semelhantes entre duas ou mais classes dentro de uma mesma região de dados. De forma análoga, uma região com baixa entropia indica um subconjunto mais homogêneo, em que uma classe aparece com uma distribuição maior perante as demais.

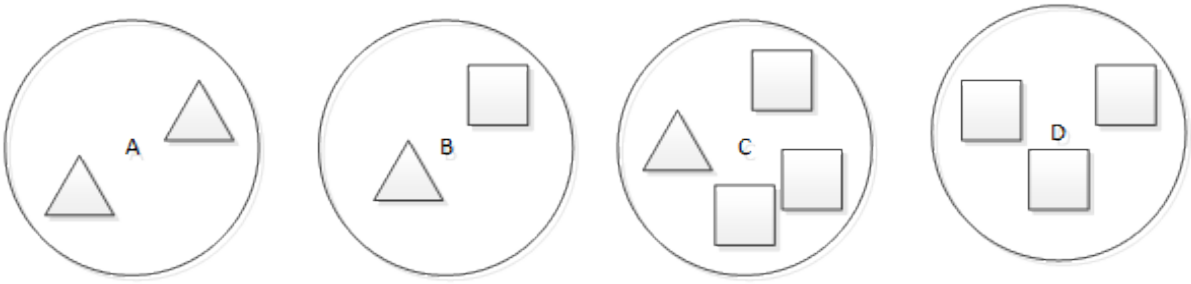


Figura 2.2: Um exemplo de como a entropia é considerada na análise de uma partição da base de dados. Nessa figura temos 4 partições com diferentes distribuições e valores de entropia. Fonte: Elaborada pelo autor.

Um exemplo do cálculo da entropia considerando um conjunto de dados pode ser observado na Figura 2.2. Calculando a entropia para cada uma das partições tem-se:

- $H_A = -\frac{2}{2} * log_2(\frac{2}{2}) = 0$
- $H_B = -\frac{1}{2} * log_2(\frac{1}{2}) - \frac{1}{2} * log_2(\frac{1}{2}) = 1$
- $H_C = -\frac{1}{4} * log_2(\frac{1}{4}) - \frac{3}{4} * log_2(\frac{3}{4}) = 0,8113$
- $H_D = -\frac{3}{3} * log_2(\frac{3}{3}) = 0$

Assim, é possível verificar pelo valor de entropia que os subconjuntos A e D são homogêneos, que a partição C representa uma distribuição intermediária e que a B representa uma partição com uma classe em que existe uma divisão igual entre as classes.

A partir do conceito de entropia pode-se deduzir que, nos casos de um conjunto de dados possuir alta entropia, um classificador possivelmente terá maior dificuldade para separação das classes do que em casos com uma entropia menor, devido a distribuição das classes.

2.2 Métricas de Distância

As métricas de distância são medidas utilizadas por diversos algoritmos de aprendizagem de máquina e que são propostas para determinar similaridade ou dissimilaridade

entre exemplares (FACELI et al., 2011). Os atributos de um exemplar são considerados como dimensões em um determinado espaço e a distância entre eles indica o quão dissimilares são, ou seja, quanto menor a medida, maior a semelhança (SILVA; SARAJANE; BOSCARIOLI, 2017).

Sendo assim, as medidas de distâncias são usadas por diversos algoritmos para identificar similaridade entre exemplares. Há, na literatura, diversas medidas e entre as mais utilizadas destacam-se: Euclidiana, Manhattan e Cosseno (CASTRO; FERRARI, 2016).

A distância Euclidiana é a métrica mais comum em algoritmos como o kNN e SOM, que serão usados neste trabalho e explicados em detalhes nos próximos capítulos. Nela, os atributos devem ser numéricos e são tratados como pontos em um espaço geométrico. Assim, a distância entre dois exemplares $\mathbf{x}_a$ e $\mathbf{x}_b$ com m atributos é calculada pela sua distância geométrica:

$$dist_{\mathbf{x}_a,\mathbf{x}_b} = \sqrt{\sum_{i=1}^{m}(x_{ai} - x_{bi})^2} \tag{2.3}$$

Nos algoritmos em geral ocorre a geração de uma matriz de distância entre pares de exemplares, de forma que é possível determinar a distância de um exemplar para qualquer outro (CASTRO; FERRARI, 2016). A matriz para um caso com n exemplares pode ser verificada em:

$$\boldsymbol{D} = \begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \ldots & & & & \\ d(n,1) & d(n,2) & \ldots & d(n,n-1) & 0 \end{bmatrix}_{nxn} \tag{2.4}$$

É importante, ao usar uma medida de distância como métrica de similaridade, que se normalize todos os atributos. O uso da normalização evita que medidas em escala maior tenham um peso superior a medidas com intervalos menores (LAROSE; LAROSE, 2015).

Existem diferentes técnicas para realizar a normalização. Uma das técnicas mais utilizadas é a normalização por reescala ou *min-max* (FACELI et al., 2011). Nessa técnica, inicialmente se definem os novos valores mínimo e máximo do atributo e, após isso, a seguinte operação é realizada para cada valor do atributo (FACELI et al., 2011):

$$x_{Novo} = norm_{Min} + (\frac{x_{Atual} - x_{Min}}{x_{Max} - x_{Min}}) * norm_{Max} \tag{2.5}$$

Onde x_{Min} e x_{Max} se referem aos valores mínimos e máximos atuais do atributo e $norm_{Min}$ e $norm_{Max}$ se referem aos novos valores desejados. É comum a escolha de 0 a 1 para os valores limites (FACELI et al., 2011; LAROSE; LAROSE, 2015).

2.3 Algoritmo de Classificação de Dados *K* Vizinhos Mais Próximos

O *k*NN (*k Nearest Neighbours* ou k Vizinhos mais Próximos) é um algoritmo de classificação de dados proposto por Cover e Hart (1967). É um classificador de aprendizagem supervisionada que identifica exemplares a partir da comparação com exemplares conhecidos (rotulados) próximos a ele.

A classificação é feita a partir de uma votação realizada entre os seus vizinhos mais próximos, consistindo em identificar a classe (rótulo) desses exemplares e verificar qual é a majoritária, sendo essa a classe a ser considerada ao exemplar desconhecido. É interessante notar que o fato de usar uma medida de número de vizinhos, em vez de uma medida de diâmetro, permite que esse classificador seja usado tanto em regiões com alta quanto com baixa densidade de dados (KONONENKO; KUKAR, 2007).

O número de vizinhos considerados é parametrizado por k, vindo daí o nome do algoritmo. No caso de empate na votação, alguma regra deve ser determinada para se escolher a classe. Em vista disso, em um cenário de duas classes, é comum escolher k como um número ímpar para evitar o empate (CASTRO; FERRARI, 2016). No caso do k já ser especificado, o mesmo pode ser identificado na nomenclatura do classificador. Por exemplo, no caso de k igual a 1, 3 e 5, podemos denominar o classificador como 1NN, 3NN e 5NN, respectivamente.

Em geral, a distância utilizada é Euclidiana, mas outras medidas podem ser usadas, de acordo com o problema abordado (CASTRO; FERRARI, 2016). Um exemplo didático de k variando pode ser verificado na Figura 2.3, que demonstra os impactos de diferentes valores de k para um problema de duas classes. No exemplo, as classes são representadas por triângulos e quadrados, o número dentro do exemplar identifica a posição da proxi-

midade com relação ao exemplar desconhecido, representado por um círculo. O exemplar será classificado de acordo com o valor de k escolhido: Para k igual a 1 ou 3 será decidido que a classe será um triângulo; para k igual a 5 será votado como um quadrado. Já para casos de k de 2 e 4 a votação termina em empate, pois teremos o mesmo número de triângulos e quadrados e algum critério deve ser escolhido para a decisão.

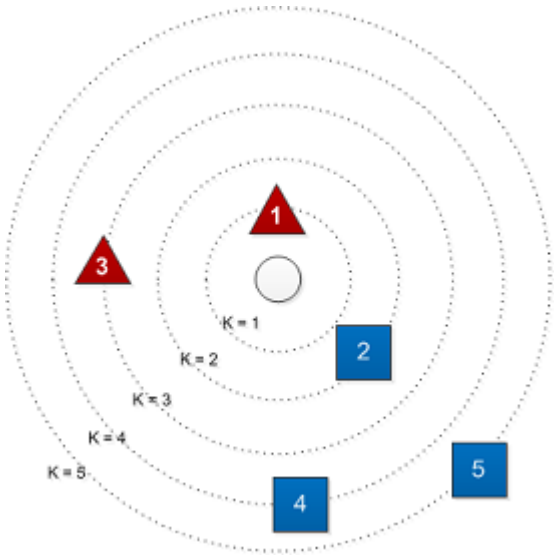


Figura 2.3: Exemplo do kNN para classificação de duas classes, representadas por triângulos e quadrados. Fonte: Elaborada pelo autor.

A escolha do k tem uma grande influência no resultado do algoritmo. Valores baixos de k podem levar a perda de generalização e correr risco de sobre-ajuste (LAROSE; LAROSE, 2015), isso é importante, principalmente, em bases com alto índice de ruídos (KONONENKO; KUKAR, 2007). Uma alternativa para essa situação é compensar essas deficiências com o aumento dos valores de k, diminuindo a probabilidade de erro na classificação. No entanto, com o aumento de k, os exemplares mais distantes passam a contribuir para decisão da informação, diminuindo o impacto da informação local e podendo acarretar no aumento do erro (LAROSE; LAROSE, 2015). Devido a essa condição antagônica, não existe um valor ideal de k que possa ser determinado para qualquer base, dessa maneira, o valor desse parâmetro deve ser escolhido de acordo com cada problema para ponderar essa situação.

Devido as suas características, o kNN é classificado como algoritmo preguiçoso (*lazy learning*) (CASTRO; FERRARI, 2016). Isso se deve ao fato de não existir fase de aprendizagem para criação do modelo. Como o processamento é todo realizado na fase da

classificação, o tempo de execução do algoritmo pode ser lento, aumentando a complexidade computacional nessa fase do algoritmo (KONONENKO; KUKAR, 2007).

Também é importante notar que é necessário ter todos os exemplares no momento de classificação para o cálculo da distância e para determinar quais são os vizinhos de um exemplar desconhecido. Esse requisito cria uma necessidade de espaço de armazenamento maior se comparado a outros classificadores, o que se deve à ausência de um modelo.

Perda de desempenho com ruídos, complexidade computacional e necessidade de espaço de armazenamento são características que fazem com que o kNN tenha dificuldades para ser usado em problemas reais (KONONENKO; KUKAR, 2007; GARCÍA et al., 2012; TRIGUERO et al., 2012).

2.4 Seleção de Protótipos

Uma técnica que permite enfrentar os problemas mencionados de espaço de armazenamento, ruído e complexidade computacional do kNN é a redução de dados (GARCÍA et al., 2012).

As técnicas de redução de dados buscam encontrar um conjunto de treinamento que tenha uma menor quantidade de exemplares e que mantenha, ou até mesmo melhore, o desempenho do classificador. Duas técnicas de redução de dados são a geração de protótipos (*PG - Prototype Generation*) e a seleção de protótipos (*PS - Prototype Selection*).

Ambas utilizam protótipos para representar uma base de dados reduzida. A diferença entre as técnicas se situa em como são escolhidos os protótipos. Na geração de protótipos, os métodos podem, além de selecionar exemplares a partir dos dados originais, gerar novos exemplares sem ligação direta com a base de dados original. Já no caso da seleção de protótipos, apenas os protótipos que representam os dados originais são utilizados. Dessa maneira, a diferença essencial é que na seleção de protótipos se presume que os dados reais são suficientes e mais adequados para classificação.

Sendo assim, a maior vantagem da seleção de protótipos é a possibilidade de criar um novo subconjunto de exemplares relevantes sem a criação de dados artificiais.

Para um entendimento formal, suponha uma base de dados de treinamento com m atributos e n exemplares, isto é $\boldsymbol{X}_{Train} = [\boldsymbol{x}_{11}, \boldsymbol{x}_{12}, ..., \boldsymbol{x}_{nm}]$, e cada exemplar possui um rótulo onde $\boldsymbol{Y}_{Train} = [y_1, y_2, ..., y_n]$. Na seleção de protótipos, um subconjunto de $\boldsymbol{X}_{Train}$

é selecionado, $\boldsymbol{X}_{PS}$, onde $\boldsymbol{X}_{PS} \subseteq \boldsymbol{X}_{Train}$. Normalmente, em um processo de classificação, $\boldsymbol{X}_{Train}$ é usado para criação do modelo para classificar os exemplares desconhecidos, no caso da seleção de protótipos, em vez do conjunto original, $\boldsymbol{X}_{PS}$ é usado para ser a base de treinamento para o classificador.

A taxonomia proposta por García et al. (2012) divide os métodos de seleção de protótipos em diferentes critérios que permitem separar os objetivos e abordagens dos métodos desenvolvidos.

2.4.1 Taxonomia de métodos de seleção de protótipos

Os métodos podem ser divididos pela forma que realizam a pesquisa no conjunto de treinamento $\boldsymbol{X}_{Train}$ para tomar a decisão de manter ou remover os exemplares e criar $\boldsymbol{X}_{PS}$.

A taxonomia de García et al. (2012) foi assim definida:

- **Incremental** - Nesse caso, $\boldsymbol{X}_{PS}$ é inicializado vazio e são adicionados os exemplares de $\boldsymbol{X}_{Train}$, conforme algum critério de escolha. Tal método tem vantagem de permitir que futuros exemplares possam ser adicionados de forma ágil, utilizando o mesmo critério. Também pode ser rápido quando comparado a demais abordagens. A grande desvantagem do método é que, no início, poucos exemplares estão disponíveis e, portanto, pouca informação, podendo estar mais suscetível a erros.

- **Decremental e Lote** - Nessas abordagens, $\boldsymbol{X}_{PS}$ é inicializado com todos exemplares de $\boldsymbol{X}_{Train}$ e algum critério é utilizado para remover os exemplares de $\boldsymbol{X}_{PS}$. A vantagem é que todos os dados estão disponíveis para análise, permitindo que o algoritmo tenha o máximo de informação para trabalhar. A desvantagem do método é que não existe redução de dados nessa etapa, o que exige maior recurso de armazenamento e tempo de treinamento. Em geral, esses métodos são mais efetivos quando se considera que o treinamento ocorra uma única vez e que futuras execuções de teste possam ocorrer com menor custo após a diminuição dos dados. Caso a remoção ocorra exemplar a exemplar, o método é nomeado como decremental. No caso de a remoção ocorrer para todos os exemplares em conjunto, o método é nomeado como lote.

- **Mixed e Fixo** - É iniciada com $\boldsymbol{X}_{PS}$ já definido por um critério pré-definido ou

aleatório e, depois, um processo pode adicionar ou remover exemplares do subconjunto, permitindo correções. No caso do *mixed* não existe nenhuma restrição para remoção ou adição, enquanto no caso do fixo, necessariamente o subconjunto deve finalizar com o mesmo número de exemplares.

2.4.2 Tipo de seleção

Outro fator que tem alta influência no comportamento do algoritmo de seleção é a região que é procurada para remover os exemplares. Com relação a isso os métodos podem ser divididos em:

- **Condensação** - Os métodos nesse grupo procuram manter os exemplares que estão próximos a classes divergentes e remover exemplares em áreas com maior homogeneidade. O nível de redução em geral é alto, já que normalmente existem mais exemplares em áreas homogêneas. No entanto, pode ocorrer um impacto negativo no desempenho.

- **Edição** - Esses métodos removem exemplares que são ruídos, ou seja, que não concordam com seus vizinhos. A remoção desses exemplares gera uma fronteira de decisão mais suave para o classificador. Devido a remoção não atacar os exemplares internos, esse tipo de método tem uma menor taxa de redução e as melhorias são focadas no aumento do desempenho.

- **Híbrido** - Métodos híbridos procuram criar o menor conjunto de treinamento enquanto mantêm ou aumentam o desempenho. Sendo assim, esses métodos trabalham tanto na fronteira quanto em exemplares internos para obter os resultados.

Diversos métodos foram criados com princípios de seleção de protótipos. Uma lista deles pode ser encontrada em García et al. (2012).

2.5 Mapas Auto-Organizáveis

Mapa auto-organizável de Kohonen (*Self-Organizing Map* ou SOM) é um tipo de rede neural que consiste em neurônios localizados em uma matriz de baixo nível dimensional. O processo, proposto por Kohonen em 1982, baseia-se em um algoritmo não supervisionado que permite o agrupamento de dados de forma que os exemplares similares sejam

associados aos mesmos neurônios ou a neurônios adjacentes, enquanto exemplares menos similares vão se situando gradualmente mais distantes no mapa (KOHONEN, 2013). O aprendizado do SOM que permite esse agrupamento é um algoritmo interativo que busca representar a distribuição de exemplares em uma matriz ou mapa de neurônios, procurando a inspiração na distribuição dos neurônios biológicos, que possuem questões topológicas, em que funções neurológicas semelhantes se distribuem em uma mesma região de uma forma ordenada.

Os mapas podem ter diferentes dimensões mas, em geral, são utilizados mapas de duas dimensões (2-D) (KOHONEN, 2013) comumente em formatos hexagonal ou retangular (SILVA; SARAJANE; BOSCARIOLI, 2017). Não existe um tamanho padrão de mapa ideal, sendo necessário testar diferentes tamanhos de mapa por tentativa e erro (KOHONEN, 2013).

O processo de aprendizagem do mapas auto-organizáveis de Kohonen pode ser dividido em 3 etapas: competição, cooperação e adaptação (LAROSE; LAROSE, 2015).

Para exemplificar o aprendizado que ocorre no SOM, suponha um conjunto de treinamento $\boldsymbol{X}_{Train}$, normalizado previamente, que será usado para treinamento da rede. O tamanho do mapa foi determinado previamente e possui um número total de neurônios l. Cada neurônio j dessa rede tem um vetor de peso tal que $\mathbf{w}_j = [w_{j1}, w_{j2}, ..., w_{jn}]^T$, onde $j = 1, 2, ..., l$; que foi inicializado de forma aleatória.

Durante o processo de competição, um exemplar é aleatoriamente selecionado do conjunto de treinamento e é comparado com os neurônios do mapa. A comparação entre $\mathbf{x}_n$ e $\mathbf{w}_j$ é feita por meio de alguma métrica de similaridade, no caso mais comum, utiliza-se a distância Euclidiana, conforme definida na Equação 2.3.

O neurônio com a menor distância é declarado vencedor e, em geral, é chamado de *Best Match Unit* (*BMU*) (SILVA; SARAJANE; BOSCARIOLI, 2017).

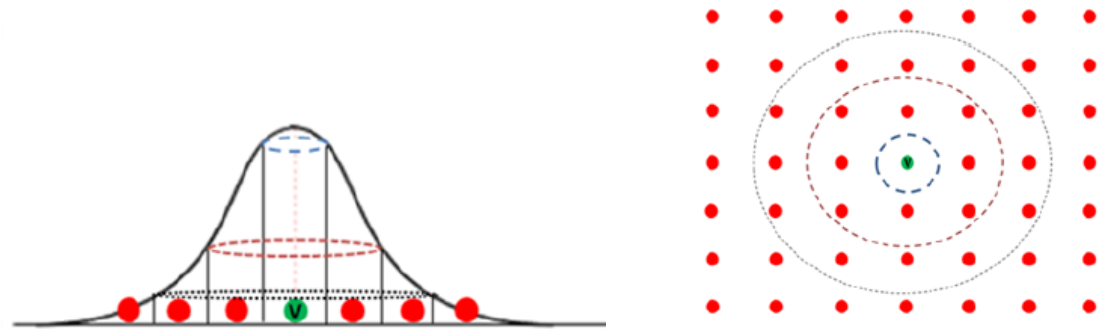
Na fase de cooperação, os neurônios na vizinhança do *BMU* também são atualizados de forma a se criar um mapa em que neurônios mais próximos indiquem características semelhantes. O tamanho da vizinhança afetada é indicado por um parâmetro de raio de vizinhança r.

No passo de adaptação, os pesos do neurônio vencedor e de seus vizinhos dentro do raio r são atualizados, como estratégia de aproximá-los do exemplar $\mathbf{x}_n$. A atualização é feita da seguinte maneira, considerando t como o indicador da época:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + h_{BMUi(t)} * \eta * [\mathbf{x}(t) - \mathbf{w}_i(t)] \tag{2.6}$$

Onde h_{BMUi} é nomeado a função de vizinhança topológica, em que o subscrito BMU indica o neurônio vencedor. Essa função é feita de forma que quanto mais distante esteja o neurônio de $\mathbf{w}_{BMU}$, menor seja a intensidade da atualização. O valor de η na equação se refere à taxa de aprendizado do algoritmo.

Essa atualização pode ser verificada na Figura 2.4, em que se apresenta um exemplo de uma função de vizinhança, no caso a Gaussiana. A visão lateral demonstrada na Figura 2.4a mostra a diferença na intensidade da atualização do $\mathbf{w}_{BMU}$, sinalizado como v na figura, e a diminuição da intensidade conforme os nós vizinhos ficam mais distantes. A visão superior demonstra um mapa retangular e o número de neurônios que é impactado considerando r igual a 1 ou r igual a 2.



(a) Visão lateral (b) Visão superior

Figura 2.4: Exemplo de uma função de vizinhança no SOM, onde o neurônio vencedor é demonstrado pelo valor v. Na figura (a) se tem a visão lateral demonstrando a diminuição da intensidade, conforme se distancia do neurônio vencedor. À direita, na figura (b), temos uma visão superior para demonstrar a diferença do número de neurônios impactados considerando r igual a 1 e 2. Fonte: Silva, Sarajane e Boscarioli (2017).

Como boa prática é de interesse que η e r tenham valores altos no início do aprendizado e diminuam conforme o processo se estabilize (KOHONEN, 2013). Veja Kohonen (KOHONEN, 2013) para uma explicação completa da regra de treinamento do mapa de SOM.

2.6 Medidas de Complexidade de Dados

Quando se tem um conjunto de dados rotulados e deseja-se realizar uma modelagem preditiva, as informações disponíveis sobre os dados não permitem identificar qual será a complexidade da tarefa de classificação de dados.

Medidas que possam indicar características de complexidade de dados são interessantes para analisar um conjunto de dados à priori e determinar níveis de sobreposição e expectativas de resultados esperados em uma modelagem preditiva. Medidas de complexidade foram estudo de diversos trabalhos na literatura (HO; BASU, 2002; BASU; HO, 2006; SÁNCHEZ; MOLLINEDA; SOTOCA, 2007; CANO, 2013; GARCIA; CARVALHO; LORENA, 2015; MORÁN-FERNÁNDEZ; BOLÓN-CANEDO; ALONSO-BETANZOS, 2017; LORENA et al., 2018).

A aplicação destas medidas permite compreender qual é a distribuição dos dados no espaço de atributos e como as classes estão segregadas. Esse conhecimento prévio permite, entre outras coisas, determinar qual seria a complexidade para resolver uma fronteira de decisão, que pode ser radicalmente diferente para diferentes bases de dados. A diferença na distribuição dos dados e sua fronteira apresentam dificuldades diferentes que impactam no uso de um classificador e na solução do problema.

Basu e Ho (2006), por exemplo, demonstram a diferença na complexidade sobre a perspectiva de fronteira de separação em quatro categorias, como ilustradas na Figura 2.5: a) problema linearmente separável com larga margem entre as classes; b) problema linearmente separável com margem estreita entre classes; c) problema com fronteira não linear; d) classes com alto nível de sobreposição.

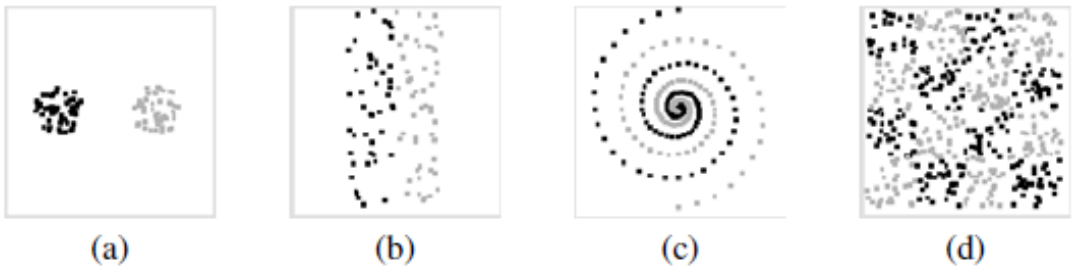


Figura 2.5: Diferentes tipos de fronteira para um problema de duas classes. As diferentes distribuições apresentam diferentes desafios para a tarefa de classificação. Fonte: Basu e Ho (2006).

Neste projeto, as medidas de complexidade são uma ferramenta para entender melhor onde o pré-processamento proposto está sendo mais efetivo e, a partir disso, validar quais bases sofrem de problemas intrínsecos como a sobreposição.

Abaixo são apresentadas medidas da literatura utilizadas nesse trabalho, considerando unicamente o problema de duas classes. Para um número maior de classes é possível extrapolar essas medidas (MORÁN-FERNÁNDEZ; BOLÓN-CANEDO; ALONSO-BETANZOS, 2017).

2.6.1 *Fischer's discrimination ratio* (F1)

A medida de Fischer (HO; BASU, 2002) permite calcular a separação entre duas classes de um respectivo atributo. A medida de Fisher para um atributo é dada pela Equação 2.7:

$$f = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{2.7}$$

Onde μ_1 e μ_2 são as médias do atributo nas respectivas classes e σ_1 e σ_2 são os respectivos desvios-padrões.

Para uma base de dados com m atributos a medida é calculada para cada um dos atributos e o valor de F1 é considerado como o maior valor de f encontrado, ou seja, o atributo que tem a maior separação.

Os valores de F1 partem de zero e crescem com afastamentos da média das classes. Quanto menor o valor de F1, maior é a proximidade dos dados, o que indica uma maior sobreposição.

2.6.2 Eficiência máxima de atributo (individual) (F3)

Essa medida procura determinar a eficiência de cada atributo em separar as classes (BASU; HO, 2006). A eficiência de um atributo é definida como a fração de exemplares que podem ser separados unicamente por ele.

A medida é feita considerando os valores máximos e mínimos do atributo dentro de cada classe. A eficiência de um atributo é considerada como a fração dos exemplares que se encontra na faixa de valores que pertence a apenas uma das classes, estando fora das faixas em que ocorre a sobreposição entre as classes. Esse processo pode ser observado em um

exemplo na Figura 2.6. Neste exemplo, tem-se o caso de um atributo em um problema com duas classes, representadas por quadrados e triângulos. No caso representado, os exemplares preenchidos são os exemplares fora da região de sobreposição. Dessa maneira, segundo a medida, tem-se 5 exemplares de 10 fora da sobreposição, obtendo-se um valor de F3 para o atributo de 0,5.

O valor de F3 é definido como o maior valor de eficiência entre todos os atributos da base de dados. O valor da medida varia de 0 a 1, com valores menores indicando maior sobreposição.

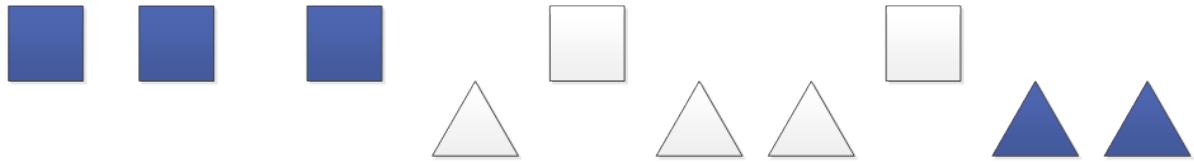


Figura 2.6: Exemplo do cálculo de F3. Fonte: Elaborada pelo autor.

2.6.3 Proporção média intra/inter classe para o vizinho mais próximo (N2)

A medida N2 calcula a razão entre a média da distância intraclasse pela média da distância interclasse dos exemplares, trazendo um indicativo da dispersão das classes internas e qual a sua distância para a classe oposta (BASU; HO, 2006). A distância intraclasse é definida como a distância do exemplar até o exemplar mais próximo da mesma classe, enquanto a distância interclasse é definida como a distância do exemplar até o exemplar mais próximo da outra classe.

Dessa forma, essa medida é dada pela Equação 2.8 :

$$N2 = \frac{\sum_{i=1}^{n} DistanciaIntra(x_i)}{\sum_{i=1}^{n} DistanciaInter(x_i)} \tag{2.8}$$

Onde x_i representa os diferentes exemplares e n o número total de exemplares. O valor de N2 varia de 0 a infinito, em que valores próximos a zero indicam exemplares mais fáceis de distinguir e, portanto, uma sobreposição menor.

Dessa maneira, N2 consegue informar qual o grau de separação das classes, sendo auxiliar às medidas F1 e F3, que permitem calcular a sobreposição. Segundo estudos, N2 é capaz de trazer resultados para verificar o desempenho do kNN em uma base de da-

dos (CANO, 2013; MORÁN-FERNÁNDEZ; BOLÓN-CANEDO; ALONSO-BETANZOS, 2017).

2.6.4 Densidade da classe na região de sobreposição (D3)

Proposto por Sánchez, Mollineda e Sotoca (2007), essa medida mede a quantidade relativa de dados que estejam em uma área de sobreposição.

Para isso, a medida usa um kNN para verificar os exemplares da própria base. Caso um exemplar esteja em discordância com seus vizinhos, ele é marcado e considerado como estando em uma área de sobreposição. A medida se refere à proporção de exemplares da base que foram marcados como sobrepostos. O valor de D3 também pode ser segregado, informando a proporção de exemplares sobrepostos por classe.

Deve ser definido o valor de k para o kNN usado para determinar a área de sobreposição. Neste trabalho foi utilizado $k = 5$, igualmente foi usado por Sánchez, Mollineda e Sotoca (2007). É interessante notar que essa medida exige um processamento superior às demais medidas por exigir que seja utilizado um classificador kNN em toda base.

Existem diversas outras medidas de complexidade de dados na literatura. Neste trabalho utilizaram-se os estudos apresentados por Cano (2013), Morán-Fernández, Bolón-Canedo e Alonso-Betanzos (2017). Os autores determinaram que, para a análise de classificadores, F1 e F3 servem como bons indicadores de sobreposição para uso geral. Os mesmos estudos também informaram que os valores de N2 são um bom indicativo para determinar o potencial de classificação do kNN. Sendo assim, decidiu-se por utilizar tais medidas neste trabalho. Como nos trabalhos supracitados não se considerou a medida D3, referente à densidade, essa será incorporada nos estudos deste trabalho.

É interessante notar que a eficácia das medidas depende do tipo de base analisada. Por exemplo, F1 representa melhor a separação entre as classes de determinados tipos de base de dados, por exemplo, no caso de duas classes separadas com os exemplares ao redor do centro com uma distribuição normal, como na Figura 2.5.a. No entanto, pode ter valores não representativos em outros tipos de base, por exemplo, em uma base com duas classes e dois atributos em que os exemplares estejam distribuídos em espirais não sobrepostas, como na Figura 2.5.c (BASU; HO, 2006).

Capítulo 3

SOMEntropyFilter

Os métodos propostos neste trabalho utilizam a seleção de protótipos para procurar melhorar o processo de um classificador 1NN em bases desbalanceadas ou com sobreposição de classes. O 1NN foi o classificador base utilizado para comparações de métodos de PS por García et al. (2012). Ele obtém benefícios dos métodos de pré-processamento devido a ser suscetível a ruídos, sobreposição e desbalanceamento de dados (TRIGUERO et al., 2012).

A proposta é que o processo de seleção de protótipo irá melhorar a performance do classificador 1NN, permitindo o uso de uma forma mais efetiva em bases de dados que sofram perda de desempenho devido à complexidade de dados. Neste trabalho, decidiu-se pelo tratamento de problemas de duas classes, considerando uma classe a positiva (classe alvo). A maior parte dos trabalhos que analisam classes desbalanceadas e métodos de avaliação são focados em problemas binários (BRANCO; TORGO; RIBEIRO, 2016).

O primeiro passo do método proposto é utilizar os mapas auto-organizáveis de Kohonen (SOM) no conjunto de treinamento ($\boldsymbol{X}_{Train}$) para agrupar os dados em diferentes nós do mapa SOM. Devido ao comportamento do SOM, espera-se que as classes sobrepostas sejam mapeadas dentro dos mesmos nós, ou seja, exemplares que dividam atributos similares coexistem dentro do mesmo nó. Assim, o SOM foi escolhido como técnica de agrupamento por permitir a possibilidade de analisar diferentes regiões com diferentes graus de similaridade da base de dados. Para esse trabalho, decidiu-se manter a distância Euclidiana como medida de similaridade para o SOM.

Como segundo passo, calcula-se a entropia dentro de cada um dos nós do SOM, usando o conceito da entropia de informação de Shannon (1948), como definido na Equação 2.2.

Para isso, considera-se a fração de exemplares da classe dentro do nó como sendo a sua probabilidade. A equação da entropia para um nó do SOM fica, portanto, assim definida:

$$p_{Pos} = \frac{N_{Pos}}{N_{Total}} \tag{3.1}$$

$$p_{Neg} = \frac{N_{Neg}}{N_{Total}} \tag{3.2}$$

$$H_{No} = -p_{Pos} * log_2(p_{Pos}) - p_{Neg} * log_2(p_{Neg}) \tag{3.3}$$

Em que N_{Pos}, N_{Neg} e N_{Total} são, respectivamente, o total de exemplares da classe positiva, negativa e a soma de ambas dentro do nó. É importante notar que a entropia é calculada de acordo com a distribuição dentro de cada nó do SOM, e que a distribuição das classes como um todo não tem nenhum impacto no resultado.

Depois de calculada a entropia em cada nó, criam-se três métodos de pré-processamento para utilizar a entropia como filtro:

- **SOMEntropyHighFilter** - Se a entropia do nó é maior ou igual a um limiar, definido como *ThresholdHigh*, os exemplares de ambas as classes dentro do nó são removidos. A ideia por trás desse método é remover exemplares sobrepostos que não adicionem informação para criação da fronteira do classificador.

- **SOMEntropyLowFilter** - Se a entropia calculada em um nó for menor ou igual a um valor de limiar, definido como *ThresholdLow*, os exemplares da classe com a menor probabilidade dentro do neurônio são marcados para remoção. O objetivo é remover exemplares dentro do nó que não concordem com a maioria, tornando a fronteira mais distinguível ao classificador. Esse método também funciona para filtrar exemplares ruidosos, removendo dados que não concordem com seus vizinhos.

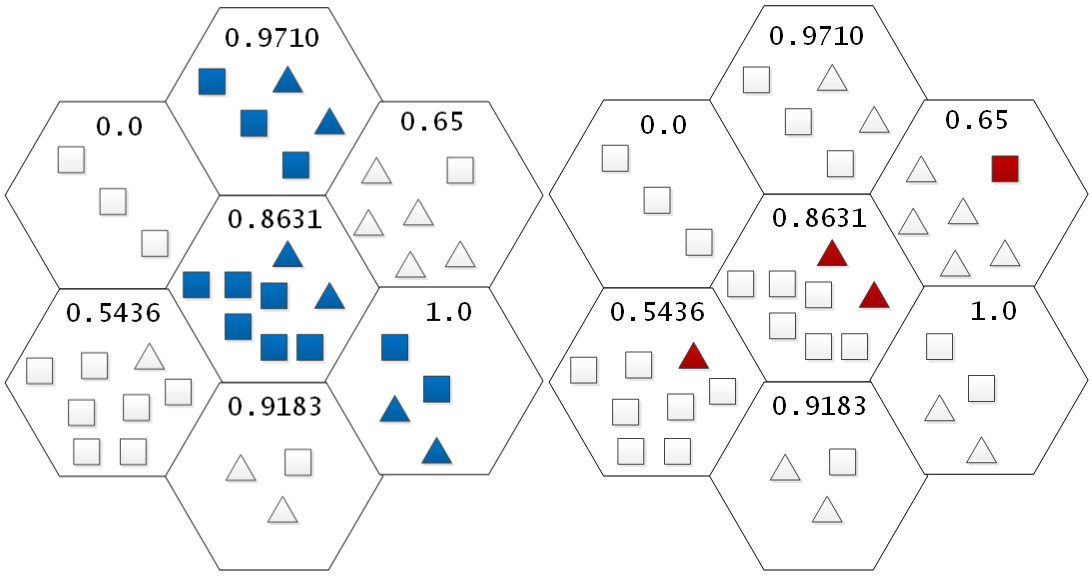
- **SOMEntropyHighLowFilter** - Nesse método, o SOMEntropyHighFilter e o SOMEntropyLowFilter são combinados para remover os exemplares das bordas, com *ThresholdHigh* maior que *ThresholdLow*. Isso servirá para testar se ambos métodos propostos, em conjunto, aumentaram o desempenho.

Após remover os exemplares, de acordo com as regras descritas anteriormente, tem-se um novo conjunto de treinamento processado, $\boldsymbol{X}_{PS}$. Esse novo conjunto é então apresentado ao kNN para a classificação. Para esse primeiro momento, decidiu-se utilizar o 1NN,

que foi escolhido como base para comparação no trabalho de García et al. (2012).

Os métodos de pré-processamento definidos neste trabalho estão ilustrados na Figura 3.1. Considerando uma grade de SOM hipotética para um conjunto de dados com duas classes (A e B), sendo a classe A representada por quadrados e a B por triângulos, demonstra-se como o pré-processamento ocorre para os casos do SOMEntropyHighFilter e SOMEntropyLowFilter.

No lado esquerdo da Figura 3.1, o SOMEntropyHighFilter foi usado com *ThresholdHigh* igual a 0,8631. Observa-se que os exemplares destacados são os cuja entropia do nó é igual ou maior ao limiar escolhido e, portanto, exemplares de ambas as classes foram removidos. Do lado direito está um exemplo do SOMEntropyLowFilter com o mesmo valor para o *ThresholdLow* igual a 0,8631. Nesse método, foram selecionados exemplares de nós com a entropia calculada menor ou igual ao parâmetro de limiar e apenas da classe que está menos presente dentro do nó.



(a) SOMEntropyHighFilter (b) SOMEntropyLowFilter

Figura 3.1: Exemplo do processo de marcação de exemplares para remoção dentro de um mapa de SOM utilizando o SOMEntropyHighFilter (à esquerda) e SOMEntropyLowFilter (à direita) em uma hipotética rede SOM com *ThresholdHigh* igual a 0,8631 e *ThresholdLow* igual a 0,8631. Essa remoção é demonstrada pelos exemplares destacados. Fonte: Elaborada pelo autor.

No caso de ser utilizado o SOMEntropyHighLowFilter, com os mesmos valores de limiar, ambas as regras seriam aplicadas. Dessa maneira, na Figura 3.1, seriam selecionados para remoção os exemplares destacados de ambos os lados.

A proposta está resumida em pseudocódigo no Algoritmo 1.

Algoritmo 1: Algoritmo do SOMEntropyFilter.

Entrada: Base de Dados ($\boldsymbol{X}$); TipoSomEntropyFilter {HighFilter,LowFilter,HighLowFilter}; ThresholdHigh; ThresholdLow; TamanhoMapaSOM; FormatoMapaSOM
Saída : Base de Dados Processada ($\boldsymbol{X}_{PS}$)

```
MapaSOM ← Treina um mapa de Kohonen usando a base X utilizando os
 parâmetros: TamanhoMapaSOM e FormatoMapaSOM;
para cada nó do MapaSOM faça
    N_Pos ← Quantidade de exemplares da classe positiva dentro do nó.
    N_Neg ← Quantidade de exemplares da classe negativa dentro do nó.
    N_Total ← N_Pos + N_Neg
    p_Pos ← N_Pos/N_Total;
    p_Neg ← N_Neg/N_Total;
    H_No ← −p_Pos ∗ log2(p_Pos) − p_Neg ∗ log2(p_Neg);
    se TipoSOMEntropyFilter é igual a HighFilter ou HighLowFilter então
        se ThresholdHigh >= H_No então
            Marca todos os exemplares do nó
    se TipoSOMEntropyFilter é igual a LowFilter ou HighLowFilter então
        se ThresholdLow <= H_No então
            se p_Pos < p_Neg então
                Marca exemplares da classe positiva;
            senão
                Marca exemplares da classe negativa;
X_PS ← Base de dados X com os exemplares marcados anteriormente removidos;
Retorna X_PS;
```

Capítulo 4

Metodologia

Neste capítulo, descrevem-se os passos para desenvolvimento e validação do algoritmo SOMEntropyFilter. Para o algoritmo e outros processos relacionados ao estudo, decidiu-se pelo uso da linguagem R (R Core Team, 2018), devido às facilidades para uso das ferramentas de aprendizagem de máquina disponíveis em bibliotecas.

4.1 Base de dados

Para validar o funcionamento dos métodos desenvolvidos foram utilizadas bases de dados artificiais e reais. Buscou-se bases com sobreposição e desbalanceamento em diferentes níveis para validar o algoritmo nessas situações. Nas seções abaixo as bases artificiais e reais são apresentadas em detalhes.

4.1.1 Bases artificiais

Para compreender melhor o algoritmo é de interesse avaliá-lo de uma forma controlada em bases com diferentes características, possibilitando compreender os efeitos de sobreposição e desbalanceamento. Para isso, foram criadas diferentes bases artificiais que simulam essas situações.

Inicialmente, foram criadas bases com 1.000 exemplares de duas classes, sendo 500 exemplares por classe. As bases são de dois atributos para facilitar a visualização em 2-D. Os exemplares foram gerados usando uma distribuição gaussiana com desvio-padrão de 1,0, utilizando-se para geração o método *rnorm* do pacote base do R (R Core Team, 2018). Para geração de uma situação de sobreposição, fixou-se a média de uma das classes

e se variou a média da segunda, baseada em uma diferença entre 0 e 5 da primeira classe, com passos de 0,5. Dessa maneira, criaram-se 11 bases de dados com diferentes níveis de sobreposição. Uma dessas classes é então considerada como a classe alvo, ou positiva da base.

Para o caso de serem geradas bases com desbalanceamento, utilizou-se a abordagem de remover exemplares da classe alvo do algoritmo. O número de exemplares final da classe positiva foi escolhido para gerar diferentes níveis de desbalanceamento, iniciando em cenários com baixo nível de desbalanceamento até os níveis mais severos. Os números de exemplares escolhidos podem ser verificados na Tabela 4.1, em que se colocaram, para identificar o desbalanceamento, as informações da proporção da classe positiva ao total de exemplares e a quantidade de exemplares da classe negativa para um exemplar da classe positiva.

Tabela 4.1: Diferentes níveis de desbalanceamento para geração das bases artificiais

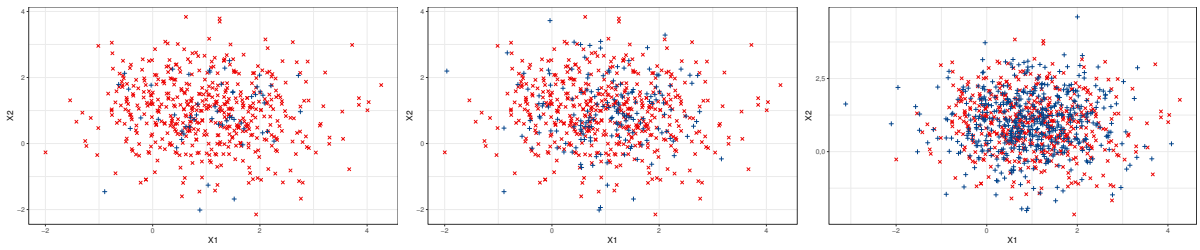
# Exemplares da classe positiva	**Proporção da classe positiva (%)**	**Taxa de desbalanceamento**
500	50%	1,00:1
409	45%	1,22:1
333	40%	1,50:1
269	35%	1,86:1
214	30%	2,34:1
167	25%	2,99:1
125	20%	4,00:1
88	15%	5,68:1
56	10%	8,93:1
26	5%	19,23:1
5	1%	100,00:1

Fonte: Elaborada pelo autor.

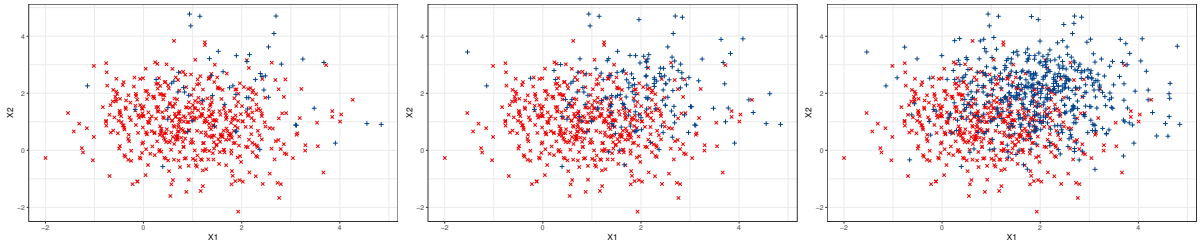
Para os testes, decidiu-se utilizar uma combinação dos 11 cenários de sobreposição com os 11 cenários de desbalanceamento, gerando um total de 121 bases que permitem simular os diferentes casos, de forma independente ou combinada. Na Figura 4.1, pode-se ver a distribuição de algumas dessas bases em duas dimensões.

4.1.2 Bases reais

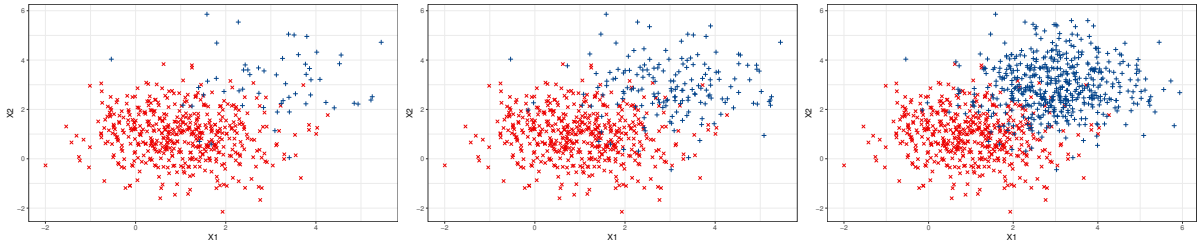
Para verificar o comportamento do algoritmo em bases reais, decidiu-se utilizar bases de domínio público do repositório de base de dados para aprendizagem de máquina da Universidade da Califórnia Irvine (UCI) (DHEERU; TANISKIDOU, 2017).



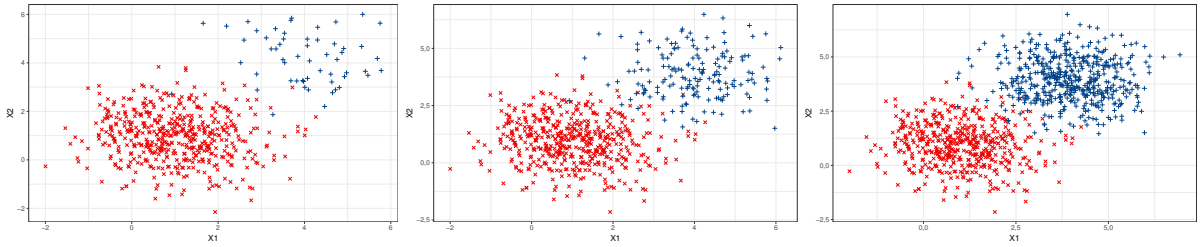
(a) Dif. 0,0; Classe Pos. 10% (b) Dif. 0,0; Classe Pos.. 25% (c) Dif. 0,0; Classe Pos. 50%



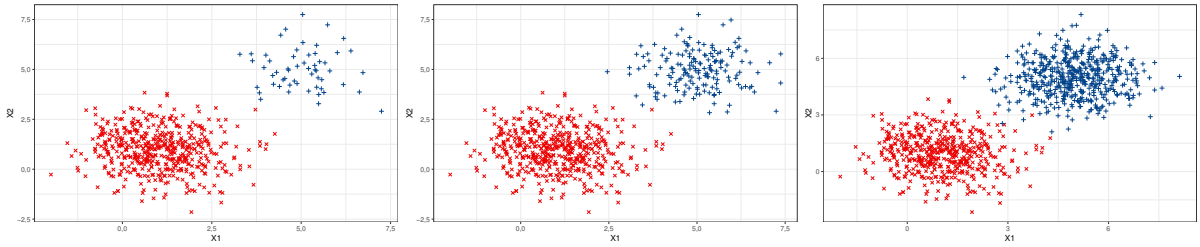
(d) Dif. 1,0; Classe Pos. 10% (e) Dif. 1,0; Classe Pos.. 25% (f) Dif. 1,0; Classe Pos. 50%



(g) Dif. 2,0; Classe Pos. 10% (h) Dif. 2,0; Classe Pos. 25% (i) Dif. 0,0; Classe Pos. 50%



(j) Dif. 3,0; Classe Pos. 10% (k) Dif. 3,0; Classe Pos.. 25% (l) Dif. 3,0; Classe Pos. 50%



(m) Dif. 4,0; Classe Pos. 10% (n) Dif. 4,0; Classe Pos. 25% (o) Dif. 4,0; Classe Pos. 50%

Figura 4.1: Distribuições das bases artificiais para algumas das diferentes configurações de diferença de média e desbalanceamento. O x azul representa a classe negativa e o + vermelho a classe positiva. Fonte: Elaborada pelo autor.

Parte das bases selecionadas são problemas multi-classe. Para adequar este tipo de base aos problemas em que se pretende avaliar o algoritmo proposto, selecionou-se arbitrariamente uma das classes para ser a positiva, procurando-se criar cenários que gerassem diferentes níveis de desbalanceamento ou sobreposição. Um resumo das bases escolhidas se encontra na Tabela 4.2 representada pelo número de classes, quantidade de atributos, classe alvo escolhida e proporção da classe alvo para o total.

Tabela 4.2: Bases utilizadas da UCI

	Base de dados	**# Exemplares**	**Classes**	**Classe positiva**	**% Classe positiva**	**# Atributos**
1	Ecoli	336	cp,im,pp, imU,om,omL, imL,imS	pp	15%	**7**
2	Glass	214	Ri,Na,Mg, Al,Si,K, Ca,Ba,Fe	Si	4%	9
3	Haberman	306	1,2	2	26%	3
4	Heart	303	0,1,2,3,4	1,2,3,4	4%	13
5	Hepatitis	155	Die,Live	Die	21%	19
6	Iris	150	Setosa, versicolour, Virginica	Versicolour	33%	4
7	Libra	360	1,2,3,4,5,6, 7,8,9,10,11, 12,13,14,15	1,2,3	20%	90
8	Mamographic	961	benign, malignant	malignant	46%	5
9	Pima	768	0,1	1	35%	8
10	SPECTF-Heart	268	0,1	0	21%	44
11	Wine	178	1,2,3	2	40%	13
12	Wiscosin	699	benign, malignant	malignant	34%	9

Fonte: Repositório da UCI (DHEERU; TANISKIDOU, 2017).

No caso de um atributo da base escolhida possuir um dado faltante, escolheu-se usar o método para preenchimento dos valores a média ou moda do atributo (CASTRO; FERRARI, 2016).

4.2 Experimento para Teste da Eficiência do Algoritmo

Para simular a eficiência do algoritmo proposto, criaram-se diferentes passos para realizar testes de performance e eficiência nas bases selecionadas.

Considerando-se a escolha de uma base ($\boldsymbol{X}$) para o processo, primeiro se deve realizar a normalização para garantir o correto funcionamento dos classificadores kNN e SOM. Para normalização, adotou-se a normalização min-max dos atributos, referenciada na Seção 2.2, utilizando para isso o método *preprocess* do pacote R *caret* (WING et al., 2018), configurando-se o método para *range* e o intervalo para 0 e 1.

Em seguida, definiu-se os conjuntos de treinamento ($\boldsymbol{X}_{Train}$) e teste ($\boldsymbol{X}_{Test}$) a partir da base escolhida. Para isso, decidiu-se utilizar a técnica de validação cruzada em k-pastas (*k-fold cross-validation*), dividindo os exemplares da base em k pastas. Após a divisão, uma pasta é selecionada como conjunto de teste e as restantes para o treinamento. Esse processo é repetido em k interações, alternando-se em cada uma dessas a pasta de teste, até que todas as pastas sejam usadas para esse fim. Para esse trabalho, utilizou=se $k = 10$, uma medida que se tornou padrão para testes de desempenho (CASTRO; FERRARI, 2016). Para esse processo, usou-se o pacote *cvTools* do R (ALFONS, 2012) com 10 pastas e distribuição aleatória.

Como primeiro passo dos algoritmos é criado um mapa SOM, com o pacote R *Kohonen* (WEHRENS; BUYDENS, 2007), usando o conjunto de treinamento $\boldsymbol{X}_{Train}$. Um fator essencial na criação do mapa é a escolha do seu formato e tamanho. Para o formato, escolheu-se um mapa hexagonal de lados iguais. Para a dimensão, como não existe um método para determinar o tamanho ideal (KOHONEN, 2013), decidiu-se utilizar diferentes tamanhos de mapa para verificar empiricamente qual o melhor parâmetro.

Dessa maneira, foi definida uma fórmula para o tamanho do mapa como descrito nas equações 4.1 e 4.2. Nela é determinada uma constante C_{Mapa}, em que se variou valores de $\{-2, -1, 0, 1, 2, 3, 4, 5\}$, com o objetivo de gerar diferentes tamanhos de mapa:

$$l_{SOM} = \frac{\sqrt{n}}{2} + C_{Mapa} \tag{4.1}$$

$$TamanhoMapaSOM = (l_{SOM})^2 \tag{4.2}$$

Em que n se refere ao número de exemplares da base de treinamento da qual estamos gerando o mapa.

Após a criação do mapa, calcula-se a entropia em cada um dos nós, considerando as classes positiva e negativa, conforme a Equação 3.3, obtendo-se o valor de entropia, denominado H_{No}, para cada um dos nós.

Após o cálculo da entropia, aplicam-se os filtros nos exemplares do nó de acordo com os parâmetros de limiar. Existem dois métodos de filtro para seleção:

- **HighFilter** - Compara-se o valor de entropia H_{No} com o valor de *ThresholdHigh*. Se a entropia calculada for maior ou igual ao *threshold*, se marcam-se os exemplares que estão dentro desse nó para exclusão. Esse filtro é utilizado nos métodos SOMEntropyHighFilter e SOMEntropyHighLowFilter.

- **LowFilter** - A entropia H_{No} calculada é comparada com o valor de *ThresholdLow*. Caso a entropia seja menor ou igual ao *threshold*, os exemplares da classe que têem menor presença dentro do respectivo nó são marcados para remoção. Nesse caso, o filtro é utilizado nos métodos SOMEntropyLowFilter e SOMEntropyHighLowFilter.

Após esse processo, o conjunto de treinamento $\boldsymbol{X}_{Train}$ é alterado para remover todos os exemplares marcados nos filtros anteriores. Desse modo, após a seleção de protótipos, tem-se um novo conjunto de treinamento que será chamado de $\boldsymbol{X}_{PS}$.

Em sequência, $\boldsymbol{X}_{PS}$ é utilizado para ser a base de comparação do 1NN para identificar qual a classe alvo do conjunto de teste $\boldsymbol{X}_{Test}$. Para isso, utilizou-se o método *KNN* do pacote R *class* (VENABLES; RIPLEY, 2002), passando os conjuntos de treinamento e teste com $k = 1$.

O método retorna classificados os dados da base de teste. Esses são então comparados com os valores originais. Por meio da comparação com os dados da variável alvo do grupo de treinamento original, cria-se uma matriz de confusão, conforme a Tabela 4.3, para validação dos resultados. Os valores são consolidados como a somatória dos resultados das 10 pastas de teste.

Tabela 4.3: Matriz de confusão

		Classe predita	
		positivo	negativo
Classe Esperada	positivo	VP (Verdadeiros positivos)	FN (Falsos negativos)
	negativo	FP (Falsos positivos)	VN (Verdadeiros negativos)

Fonte: Silva, Sarajane e Boscarioli (2017).

Esse processo é repetido por 5 vezes, com os valores finais sendo a média dos cinco resultados consolidados.

Para medir o desempenho do algoritmo, utilizam-se três medidas: a acurácia, o *F-Score* e o *G-Mean*. O *F-Score* e o *G-Mean* são comumente utilizados para validar dados

desbalanceados (HE; GARCIA, 2009), o *F-Score* mede a eficiência do classificador focado no desempenho da classe positiva, enquanto o *G-Mean* dá uma importância igual ao desempenho das classes positiva e negativa (HE; GARCIA, 2009). As suas equações encontram-se abaixo (4.3 - 4.7):

$$Acurácia = \frac{VP + VN}{VP + FP + VN + FN} \tag{4.3}$$

$$Precisão = \frac{VP}{VP + FP} \tag{4.4}$$

$$Revocação = \frac{VP}{VP + FN} \tag{4.5}$$

$$F - Score = \frac{(1 + \beta)^2 * Precisão * Revocação}{\beta^2 * Precisão + Revocação} \tag{4.6}$$

$$G - Mean = \sqrt{\frac{VP}{VP + FN} * \frac{VN}{VN + FP}} \tag{4.7}$$

O valor de β em *F-Score* é utilizado para ajustar a importância entre precisão e revocação. Neste trabalho, decidiu-se utilizar o valor de β igual a 1, que faz com que o peso seja igual entre as duas medidas (BRANCO; TORGO; RIBEIRO, 2016).

Outro ponto importante é a proporção de exemplares que são removidos pela seleção de protótipos. Apesar de a redução não ser o objetivo, bases menores têm vantagens na eficiência e tempo de execução do classificador e espaço de armazenagem (GARCÍA et al., 2012). A taxa de redução pode ser dada pela proporção de exemplares de $\boldsymbol{X}_{PS}$ em relação à base original $\boldsymbol{X}_{Train}$ (KIM; OOMMEN, 2003):

$$\text{Red.} = 1 - \frac{|\boldsymbol{X}_{PS}|}{|\boldsymbol{X}_{Train}|} \tag{4.8}$$

Os valores de taxa de redução variam de 0 a 1, valores mais próximos a 1 indicam uma redução maior com um menor número de exemplares na base pré-processada. Uma base de dados com menos exemplares traz benefícios como menor espaço de armazenagem e menor tempo de processamento.

Para investigar o impacto dos parâmetros, *ThresholdHigh* e *ThresholdLow*, utilizou-se uma série de valores de entropia baseada na proporção entre as classes. Os valores escolhidos estão resumidos na Tabela 4.4. Na tabela, entende-se que a definição de classe mais presente e menos presente se refere à proporção relativa das classes dentro do nó.

Tabela 4.4: Níveis de entropia utilizados nos experimentos

Proporção entre as classes dentro do nó (classe mais presente - classe menos presente)	**Entropia calculada**
50% - 50%	1,0000
55% - 45%	0,9928
60% - 40%	0,9710
65% - 35%	0,9341
70% - 30%	0,8813
75% - 25%	0,8113
80% - 20%	0,7219
85% - 15%	0,6098
90% - 10%	0,4690
95% - 5%	0,2864
100% - 0%	0,0000

Fonte: Elaborada pelo autor.

Esses valores de entropia foram usados como parâmetros para uso dos experimentos do SOMEntropyHighFilter e SOMEntropyLowFilter. Nos casos do SOMEntropyHigh-LowFilter, fez-se uma combinação dos parâmetros seguindo a regra em que o valor do *ThresholdHigh* deve ser superior ao do *ThresholdLow*.

Na Tabela 4.5, pode-se verificar um resumo dos parâmetros utilizados nos experimentos e, na Tabela 4.6, o resumo com o total das variações de parâmetro realizadas. Para cada uma dessas variações, realiza-se o processo descrito anteriormente.

Tabela 4.5: Resumo dos parâmetros escolhidos para uso nos experimentos

C_{Mapa}	*ThresholdHigh*	*ThresholdLow*
-2	0,9928	1,0000
-1	0,9710	0,9928
0	0,9341	0,9710
1	0,8813	0,9341
2	0,8113	0,8813
3	0,7219	0,8113
4	0,6098	0,7219
5	0,4690	0,6098
	0,2864	0,4690
	0,0000	0,2864

Fonte: Elaborada pelo autor.

Em complemento às medidas de desempenho, se calcularam-se as medidas de complexidade nas bases utilizadas no experimento. O objetivo é a comparação de desempenho com as medidas encontradas para verificar se existe algum padrão de comportamento e áreas de interesse para uso do pré-processamento.

Não foi encontrado nenhum pacote R disponível no repositório do *R-CRAN* que rea-

Tabela 4.6: Resumo com a variação dos parâmetros utilizadas nos experimentos

Método	**Variação da entropia**	**Variação do mapa de SOM**	**Total de variações**
SOMEntropyHighFilter	10	8	**80**
SOMEntropyLowFilter	10	8	**80**
SOMEntropyHighLowFilter	45	8	**360**

Fonte: Elaborada pelo autor.

lizasse o cálculo de todas as medidas. Devido a isso, desenvolveu-se os cálculos utilizando como ferramenta o pacote R *dplyr* (WICKHAM et al., 2018).

Para cada uma das bases utilizadas, artificiais e reais, decidiu-se calcular os valores de F1, F3, N2 e D3, referenciados na seção de dados complexos no capítulo de referencial teórico.

Os resultados serão analisados para verificar o ganho e comportamento nas diferentes situações, para verificar em quais condições o algoritmo tem boa performance em relação ao 1NN e análise dos parâmetros envolvidos.

Capítulo 5

Resultados

Como primeiro passo, decidiu-se analisar o desempenho do algoritmo verificando os ganhos das diferentes medidas de desempenho, acurácia, *F-Score*, e *G-Mean* nas bases artificiais e reais.

Para estudar os desempenhos dos métodos nas bases artificiais, escolheram-se as bases com proporção de classe positiva de 50%, 25% e 10% e com valores de sobreposição de 0,0 a 4,0, com passos de 0,5. Os resultados estão resumidos na Tabela 5.1, os valores se referem à média do maior valor da medida de desempenho, encontrada utilizando-se diferentes parâmetros. As bases de dados estão identificadas pela sua diferença de média (nível de sobreposição) e pela proporção da classe positiva (nível de desbalanceamento). Em negrito, encontram-se os melhores valores para cada uma das bases.

Nos resultados a seguir, para melhor visualização, identificaram-se os métodos como: Original, para o 1NN sem pré-processamento; HighFilter, para o SOMEntropyHighFilter; LowFilter, para o SOMEntropyLowFilter e HighLowFilter, para o SOMEntropyHighLow-Filter.

Incluiu-se também na tabela a taxa de redução com relação à base de dados de treinamento original, que ocorreu em função do uso do pré-processamento, ou seja, a redução de exemplares $\boldsymbol{X}_{Train}$ em relação a $\boldsymbol{X}_{PS}$ (Equação 4.8).

Tabela 5.1: Melhor resultado dos diferentes métodos para as bases artificiais

Base de dados	**Método**	**Acurácia**	**F-Score**	**G-Mean**	**Taxa de redução**
0.0 / 50%	Original	0.5340 ± 0.0103	0.5296 ± 0.0124	0.5339 ± 0.0104	0.0000 ± 0.0000
0.0 / 50%	SOMEntropyHighFilter	0.5300 ± 0.0123	0.5297 ± 0.0062	0.5300 ± 0.0123	**0.9413 ± 0.0016**
0.0 / 50%	SOMEntropyLowFilter	**0.5344 ± 0.0105**	**0.5302 ± 0.0129**	**0.5343 ± 0.0106**	0.2564 ± 0.0034

Tabela 5.1 continuada da página anterior

Base de Dados	Método	Acurácia	F-Score	G-Mean	Taxa de Redução
0.0 / 50%	SOMEntropyHighLowFilter	0.5300 ± 0.0123	0.5297 ± 0.0062	0.5300 ± 0.0123	0.9410 ± 0.0013
0.5 / 50%	Original	0.5132 ± 0.0058	0.5104 ± 0.0064	0.5131 ± 0.0057	0.0000 ± 0.0000
0.5 / 50%	SOMEntropyHighFilter	**0.6026 ± 0.0171**	**0.6015 ± 0.0179**	**0.6016 ± 0.0178**	**0.9092 ± 0.0037**
0.5 / 50%	SOMEntropyLowFilter	0.5484 ± 0.0101	0.5494 ± 0.0117	0.5483 ± 0.0100	0.2406 ± 0.0034
0.5 / 50%	SOMEntropyHighLowFilter	**0.6026 ± 0.0171**	**0.6015 ± 0.0179**	**0.6016 ± 0.0178**	**0.9092 ± 0.0037**
1.0 / 50%	Original	0.6694 ± 0.0049	0.6665 ± 0.0044	0.6693 ± 0.0049	0.0000 ± 0.0000
1.0 / 50%	SOMEntropyHighFilter	0.7434 ± 0.0076	0.7391 ± 0.0143	0.7430 ± 0.0076	**0.7037 ± 0.0071**
1.0 / 50%	SOMEntropyLowFilter	0.7290 ± 0.0090	0.7277 ± 0.0097	0.7290 ± 0.0090	0.1685 ± 0.0017
1.0 / 50%	SOMEntropyHighLowFilter	**0.7472 ± 0.0083**	**0.7435 ± 0.0065**	**0.7468 ± 0.0082**	0.7002 ± 0.0083
1.5 / 50%	Original	0.7962 ± 0.0034	0.7965 ± 0.0048	0.7962 ± 0.0034	0.0000 ± 0.0000
1.5 / 50%	SOMEntropyHighFilter	**0.8420 ± 0.0056**	**0.8418 ± 0.0055**	**0.8420 ± 0.0056**	**0.4773 ± 0.0046**
1.5 / 50%	SOMEntropyLowFilter	0.8312 ± 0.0056	0.8329 ± 0.0061	0.8311 ± 0.0056	0.1097 ± 0.0020
1.5 / 50%	SOMEntropyHighLowFilter	**0.8420 ± 0.0056**	**0.8418 ± 0.0055**	**0.8420 ± 0.0056**	0.4727 ± 0.0060
2.0 / 50%	Original	0.8636 ± 0.0017	0.8637 ± 0.0017	0.8636 ± 0.0017	0.0000 ± 0.0000
2.0 / 50%	SOMEntropyHighFilter	0.9098 ± 0.0034	0.9091 ± 0.0038	0.9097 ± 0.0035	**0.3106 ± 0.0031**
2.0 / 50%	SOMEntropyLowFilter	0.8956 ± 0.0017	0.8943 ± 0.0026	0.8955 ± 0.0017	0.0658 ± 0.0023
2.0 / 50%	SOMEntropyHighLowFilter	**0.9104 ± 0.0029**	**0.9092 ± 0.0028**	**0.9103 ± 0.0029**	0.3001 ± 0.0036
2.5 / 50%	Original	0.9476 ± 0.0011	0.9476 ± 0.0012	0.9476 ± 0.0011	0.0000 ± 0.0000
2.5 / 50%	SOMEntropyHighFilter	0.9636 ± 0.0009	0.9636 ± 0.0009	0.9636 ± 0.0009	**0.1153 ± 0.0010**
2.5 / 50%	SOMEntropyLowFilter	0.9606 ± 0.0011	0.9607 ± 0.0012	0.9606 ± 0.0011	0.0230 ± 0.0008
2.5 / 50%	SOMEntropyHighLowFilter	**0.9638 ± 0.0022**	**0.9639 ± 0.0022**	**0.9638 ± 0.0022**	0.1098 ± 0.0026
3.0 / 50%	Original	0.9700 ± 0.0014	0.9700 ± 0.0014	0.9700 ± 0.0014	0.0000 ± 0.0000
3.0 / 50%	SOMEntropyHighFilter	0.9818 ± 0.0016	0.9818 ± 0.0017	0.9818 ± 0.0016	**0.0579 ± 0.0019**
3.0 / 50%	SOMEntropyLowFilter	0.9766 ± 0.0005	0.9766 ± 0.0006	0.9766 ± 0.0005	0.0108 ± 0.0008
3.0 / 50%	SOMEntropyHighLowFilter	**0.9820 ± 0.0016**	**0.9820 ± 0.0016**	**0.9820 ± 0.0016**	0.0571 ± 0.0018
3.5 / 50%	Original	0.9876 ± 0.0005	0.9876 ± 0.0005	0.9876 ± 0.0005	0.0000 ± 0.0000
3.5 / 50%	SOMEntropyHighFilter	0.9898 ± 0.0008	0.9898 ± 0.0008	0.9898 ± 0.0008	**0.0238 ± 0.0019**
3.5 / 50%	SOMEntropyLowFilter	0.9884 ± 0.0005	0.9884 ± 0.0006	0.9884 ± 0.0005	0.0051 ± 0.0007
3.5 / 50%	SOMEntropyHighLowFilter	**0.9900 ± 0.0014**	**0.9900 ± 0.0017**	**0.9900 ± 0.0017**	**0.0238 ± 0.0019**
4.0 / 50%	Original	0.9922 ± 0.0013	0.9922 ± 0.0013	0.9922 ± 0.0013	0.0000 ± 0.0000
4.0 / 50%	SOMEntropyHighFilter	**0.9954 ± 0.0017**	**0.9954 ± 0.0017**	**0.9954 ± 0.0017**	**0.0089 ± 0.0009**
4.0 / 50%	SOMEntropyLowFilter	0.9932 ± 0.0011	0.9932 ± 0.0011	0.9932 ± 0.0011	0.0019 ± 0.0003
4.0 / 50%	SOMEntropyHighLowFilter	**0.9954 ± 0.0017**	**0.9954 ± 0.0017**	**0.9954 ± 0.0017**	**0.0089 ± 0.0009**
4.5 / 50%	Original	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 50%	SOMEntropyHighFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 50%	SOMEntropyLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 50%	SOMEntropyHighLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 50%	Original	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 50%	SOMEntropyHighFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 50%	SOMEntropyLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 50%	SOMEntropyHighLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
0.0 / 25%	Original	0.6144 ± 0.0061	0.2252 ± 0.0102	0.4082 ± 0.0106	0.0000 ± 0.0000
0.0 / 25%	SOMEntropyHighFilter	0.7433 ± 0.0082	0.2131 ± 0.0215	0.3921 ± 0.0215	**0.8266 ± 0.0038**
0.0 / 25%	SOMEntropyLowFilter	0.6918 ± 0.0215	**0.2255 ± 0.0108**	**0.4085 ± 0.0111**	0.2041 ± 0.0025
0.0 / 25%	SOMEntropyHighLowFilter	**0.7439 ± 0.0016**	0.2131 ± 0.0215	0.3921 ± 0.0215	0.8189 ± 0.0050
0.5 / 25%	Original	0.6345 ± 0.0053	0.2764 ± 0.0159	0.4582 ± 0.0157	0.0000 ± 0.0000

Tabela 5.1 continuada da página anterior

Base de Dados	Método	Acurácia	F-Score	G-Mean	Taxa de Redução
0.5 / 25%	SOMEntropyHighFilter	**0.7514 ± 0.0065**	0.2838 ± 0.0093	0.4618 ± 0.0096	**0.7659 ± 0.0031**
0.5 / 25%	SOMEntropyLowFilter	0.7166 ± 0.0045	**0.2968 ± 0.0193**	0.4631 ± 0.0171	0.1865 ± 0.0035
0.5 / 25%	SOMEntropyHighLowFilter	0.7499 ± 0.0070	0.2928 ± 0.0324	**0.4631 ± 0.0101**	0.7628 ± 0.0037
1.0 / 25%	Original	0.7379 ± 0.0086	0.4859 ± 0.0115	0.6365 ± 0.0088	0.0000 ± 0.0000
1.0 / 25%	SOMEntropyHighFilter	0.8033 ± 0.0143	0.5524 ± 0.0212	0.6647 ± 0.0212	**0.6005 ± 0.0067**
1.0 / 25%	SOMEntropyLowFilter	0.7934 ± 0.0058	0.5435 ± 0.0209	0.6630 ± 0.0173	0.1386 ± 0.0044
1.0 / 25%	SOMEntropyHighLowFilter	**0.8075 ± 0.0090**	**0.5591 ± 0.0165**	**0.6707 ± 0.0167**	0.5927 ± 0.0083
1.5 / 25%	Original	0.8156 ± 0.0075	0.6192 ± 0.0128	0.7292 ± 0.0091	0.0000 ± 0.0000
1.5 / 25%	SOMEntropyHighFilter	**0.8645 ± 0.0057**	**0.7097 ± 0.0101**	0.7866 ± 0.0113	**0.4490 ± 0.0067**
1.5 / 25%	SOMEntropyLowFilter	0.8522 ± 0.0027	0.6862 ± 0.0069	0.7711 ± 0.0058	0.0985 ± 0.0013
1.5 / 25%	SOMEntropyHighLowFilter	0.8642 ± 0.0041	**0.7097 ± 0.0101**	**0.7933 ± 0.0077**	0.4370 ± 0.0065
2.0 / 25%	Original	0.8786 ± 0.0070	0.7565 ± 0.0121	0.8327 ± 0.0070	0.0000 ± 0.0000
2.0 / 25%	SOMEntropyHighFilter	**0.9241 ± 0.0035**	**0.8456 ± 0.0132**	**0.8905 ± 0.0097**	**0.2971 ± 0.0045**
2.0 / 25%	SOMEntropyLowFilter	0.9163 ± 0.0022	0.8266 ± 0.0042	0.8727 ± 0.0034	0.0573 ± 0.0007
2.0 / 25%	SOMEntropyHighLowFilter	**0.9241 ± 0.0035**	**0.8456 ± 0.0132**	**0.8905 ± 0.0097**	0.2768 ± 0.0058
2.5 / 25%	Original	0.9490 ± 0.0053	0.8984 ± 0.0106	0.9323 ± 0.0075	0.0000 ± 0.0000
2.5 / 25%	SOMEntropyHighFilter	0.9601 ± 0.0023	0.9204 ± 0.0051	0.9467 ± 0.0059	**0.1307 ± 0.0046**
2.5 / 25%	SOMEntropyLowFilter	0.9589 ± 0.0017	0.9175 ± 0.0040	0.9430 ± 0.0028	0.0239 ± 0.0020
2.5 / 25%	SOMEntropyHighLowFilter	**0.9613 ± 0.0027**	**0.9223 ± 0.0059**	**0.9467 ± 0.0026**	0.1235 ± 0.0070
3.0 / 25%	Original	0.9742 ± 0.0007	0.9486 ± 0.0012	0.9663 ± 0.0009	0.0000 ± 0.0000
3.0 / 25%	SOMEntropyHighFilter	0.9802 ± 0.0020	0.9603 ± 0.0040	0.9725 ± 0.0041	**0.0697 ± 0.0027**
3.0 / 25%	SOMEntropyLowFilter	0.9808 ± 0.0027	0.9617 ± 0.0054	**0.9744 ± 0.0039**	0.0128 ± 0.0006
3.0 / 25%	SOMEntropyHighLowFilter	**0.9811 ± 0.0023**	**0.9622 ± 0.0046**	0.9741 ± 0.0044	0.0661 ± 0.0028
3.5 / 25%	Original	0.9928 ± 0.0013	0.9856 ± 0.0025	0.9908 ± 0.0016	0.0000 ± 0.0000
3.5 / 25%	SOMEntropyHighFilter	0.9940 ± 0.0011	0.9880 ± 0.0021	0.9924 ± 0.0015	**0.0174 ± 0.0018**
3.5 / 25%	SOMEntropyLowFilter	0.9934 ± 0.0017	0.9868 ± 0.0034	0.9916 ± 0.0025	0.0038 ± 0.0010
3.5 / 25%	SOMEntropyHighLowFilter	**0.9943 ± 0.0007**	**0.9886 ± 0.0013**	**0.9930 ± 0.0014**	**0.0174 ± 0.0018**
4.0 / 25%	Original	0.9970 ± 0.0000	0.9940 ± 0.0000	0.9960 ± 0.0000	0.0000 ± 0.0000
4.0 / 25%	SOMEntropyHighFilter	**0.9979 ± 0.0008**	**0.9958 ± 0.0016**	**0.9978 ± 0.0016**	**0.0027 ± 0.0004**
4.0 / 25%	SOMEntropyLowFilter	0.9973 ± 0.0007	0.9946 ± 0.0013	0.9966 ± 0.0013	0.0005 ± 0.0002
4.0 / 25%	SOMEntropyHighLowFilter	**0.9979 ± 0.0008**	**0.9958 ± 0.0016**	**0.9978 ± 0.0016**	**0.0027 ± 0.0004**
4.5 / 25%	Original	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 25%	SOMEntropyHighFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 25%	SOMEntropyLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 25%	SOMEntropyHighLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 25%	Original	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 25%	SOMEntropyHighFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 25%	SOMEntropyLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 25%	SOMEntropyHighLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
0.0 / 10%	Original	0.8122 ± 0.0037	0.1029 ± 0.0203	0.3076 ± 0.0333	0.0000 ± 0.0000
0.0 / 10%	SOMEntropyHighFilter	**0.8982 ± 0.0016**	0.1234 ± 0.0295	0.3190 ± 0.0531	**0.4844 ± 0.0053**
0.0 / 10%	SOMEntropyLowFilter	0.8842 ± 0.0043	0.1070 ± 0.0209	0.3091 ± 0.0334	0.0936 ± 0.0016
0.0 / 10%	SOMEntropyHighLowFilter	0.8971 ± 0.0030	**0.1249 ± 0.0287**	**0.3190 ± 0.0530**	0.4623 ± 0.0075
0.5 / 10%	Original	0.8004 ± 0.0071	0.0947 ± 0.0247	0.2997 ± 0.0390	0.0000 ± 0.0000
0.5 / 10%	SOMEntropyHighFilter	**0.8978 ± 0.0023**	0.0944 ± 0.0168	0.2927 ± 0.0260	**0.4848 ± 0.0034**
0.5 / 10%	SOMEntropyLowFilter	0.8813 ± 0.0086	**0.1140 ± 0.0351**	**0.3042 ± 0.0397**	0.0898 ± 0.0017

Tabela 5.1 continuada da página anterior

Base de Dados	Método	Acurácia	F-Score	G-Mean	Taxa de Redução
0.5 / 10%	SOMEntropyHighLowFilter	0.8957 ± 0.0018	0.1048 ± 0.0285	0.2937 ± 0.0262	0.4613 ± 0.0039
1.0 / 10%	Original	0.8741 ± 0.0075	0.3967 ± 0.0304	0.6164 ± 0.0247	0.0000 ± 0.0000
1.0 / 10%	SOMEntropyHighFilter	0.9040 ± 0.0037	0.3960 ± 0.0398	0.6046 ± 0.0183	**0.3643 ± 0.0064**
1.0 / 10%	SOMEntropyLowFilter	0.8957 ± 0.0042	0.4211 ± 0.0271	**0.6166 ± 0.0247**	0.0728 ± 0.0013
1.0 / 10%	SOMEntropyHighLowFilter	**0.9061 ± 0.0039**	**0.4218 ± 0.0344**	0.6048 ± 0.0185	0.3442 ± 0.0055
1.5 / 10%	Original	0.8759 ± 0.0042	0.3937 ± 0.0193	0.6095 ± 0.0160	0.0000 ± 0.0000
1.5 / 10%	SOMEntropyHighFilter	0.9014 ± 0.0059	0.4290 ± 0.0562	0.6099 ± 0.0245	**0.3024 ± 0.0062**
1.5 / 10%	SOMEntropyLowFilter	0.9000 ± 0.0033	0.4326 ± 0.0176	**0.6187 ± 0.0231**	0.0645 ± 0.0022
1.5 / 10%	SOMEntropyHighLowFilter	**0.9065 ± 0.0087**	**0.4399 ± 0.0262**	0.6099 ± 0.0245	0.2901 ± 0.0035
2.0 / 10%	Original	0.9381 ± 0.0010	0.6982 ± 0.0063	0.8275 ± 0.0081	0.0000 ± 0.0000
2.0 / 10%	SOMEntropyHighFilter	0.9561 ± 0.0049	0.7557 ± 0.0301	0.8356 ± 0.0130	**0.1737 ± 0.0066**
2.0 / 10%	SOMEntropyLowFilter	0.9554 ± 0.0035	0.7633 ± 0.0190	0.8429 ± 0.0166	0.0322 ± 0.0011
2.0 / 10%	SOMEntropyHighLowFilter	**0.9583 ± 0.0063**	**0.7759 ± 0.0354**	**0.8483 ± 0.0151**	0.1621 ± 0.0056
2.5 / 10%	Original	0.9737 ± 0.0021	0.8727 ± 0.0087	0.9367 ± 0.0011	0.0000 ± 0.0000
2.5 / 10%	SOMEntropyHighFilter	0.9802 ± 0.0013	0.9006 ± 0.0058	0.9385 ± 0.0039	**0.0965 ± 0.0020**
2.5 / 10%	SOMEntropyLowFilter	**0.9817 ± 0.0015**	**0.9071 ± 0.0074**	0.9413 ± 0.0047	0.0159 ± 0.0005
2.5 / 10%	SOMEntropyHighLowFilter	0.9813 ± 0.0016	0.9055 ± 0.0077	**0.9415 ± 0.0040**	0.0835 ± 0.0041
3.0 / 10%	Original	0.9799 ± 0.0008	0.9007 ± 0.0044	0.9467 ± 0.0042	0.0000 ± 0.0000
3.0 / 10%	SOMEntropyHighFilter	**0.9881 ± 0.0010**	**0.9405 ± 0.0049**	**0.9628 ± 0.0040**	**0.0411 ± 0.0032**
3.0 / 10%	SOMEntropyLowFilter	0.9853 ± 0.0008	0.9254 ± 0.0070	0.9507 ± 0.0014	0.0073 ± 0.0005
3.0 / 10%	SOMEntropyHighLowFilter	**0.9881 ± 0.0010**	**0.9405 ± 0.0049**	**0.9628 ± 0.0040**	0.0392 ± 0.0025
3.5 / 10%	Original	0.9910 ± 0.0013	0.9549 ± 0.0067	0.9709 ± 0.0065	0.0000 ± 0.0000
3.5 / 10%	SOMEntropyHighFilter	**0.9942 ± 0.0023**	0.9714 ± 0.0120	0.9840 ± 0.0119	**0.0201 ± 0.0026**
3.5 / 10%	SOMEntropyLowFilter	0.9928 ± 0.0013	0.9643 ± 0.0065	0.9800 ± 0.0064	0.0042 ± 0.0007
3.5 / 10%	SOMEntropyHighLowFilter	**0.9942 ± 0.0015**	**0.9714 ± 0.0075**	**0.9840 ± 0.0051**	**0.0201 ± 0.0026**
4.0 / 10%	Original	0.9978 ± 0.0008	0.9893 ± 0.0040	0.9972 ± 0.0040	0.0000 ± 0.0000
4.0 / 10%	SOMEntropyHighFilter	**0.9982 ± 0.0000**	**0.9912 ± 0.0000**	**0.9990 ± 0.0000**	**0.0026 ± 0.0005**
4.0 / 10%	SOMEntropyLowFilter	0.9978 ± 0.0008	0.9893 ± 0.0040	0.9972 ± 0.0040	0.0006 ± 0.0001
4.0 / 10%	SOMEntropyHighLowFilter	**0.9982 ± 0.0000**	**0.9912 ± 0.0000**	**0.9990 ± 0.0000**	**0.0026 ± 0.0005**
4.5 / 10%	Original	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 10%	SOMEntropyHighFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 10%	SOMEntropyLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
4.5 / 10%	SOMEntropyHighLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 10%	Original	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 10%	SOMEntropyHighFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 10%	SOMEntropyLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**
5.0 / 10%	SOMEntropyHighLowFilter	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**0.0000 ± 0.0000**

Fonte: Elaborada pelo autor

Verifica-se que os melhores desempenhos foram apresentados pelos métodos desenvolvidos neste trabalho, para diferentes condições de sobreposição e desbalanceamento de dados.

Para continuar a análise das bases artificiais, geraram-se diferentes gráficos com os

dados dos melhores valores de acurácia, *F-Score* e *G-Mean* encontrados nos testes, após a variação dos parâmetros de *ThresholdHigh* e *ThresholdLow* e C_{Mapa}. Esses valores foram comparados com a curva do classificador 1NN sem pré-processamento. Para se demonstrar a sobreposição, colocou-se no eixo x o valor da diferença entre as médias das classes na base artificial, sendo valores próximos a zero com alta sobreposição e valores mais distantes de zero as bases com menor sobreposição. Os gráficos também foram divididos de acordo com seu desbalanceamento, escolhendo-se alguns valores fundamentais. Desta forma, pode-se observar os impactos da sobreposição e desbalanceamento ao mesmo tempo e realizar as comparações entre os três métodos gerados e o 1NN sem alteração.

O SOMEntropyHighLowFilter teve resultados semelhantes ao SOMEntropyHighFilter, conforme se verifica na Tabela 5.1. Esse fato faz com que as curvas de ganho dos dois métodos fiquem sobrepostas. Devido a isso, decidiu-se por ocultar a curva do SOMEntropyLowFilter nesses gráficos, para melhor visualização.

Para o caso da acurácia, que pode ser observado na Figura 5.1, os métodos artificiais tiveram ganhos significativamente maiores nas bases com alta sobreposição, especialmente quando temos maior desbalanceamento. Uma possível explicação para o bom desempenho nos casos de alto desbalanceamento é, estando os parâmetros otimizados para acurácia, ter havido a remoção de uma parcela considerável de exemplares da classe positiva pelo método, o que favorece a classe negativa que, por ser majoritária, aumenta o valor da acurácia. Esse efeito diminui conforme a base se torna mais balanceada, com os ganhos sendo menores, mas existentes, inclusive para as bases balanceadas.

Com relação à sobreposição, as margens de ganhos são maiores nas bases que tem maior sobreposição e diminuem conforme o nível de sobreposição diminui, até o ponto em que não temos mais ganhos com os métodos. Isso ocorre porque nessas bases as classes já estarem suficientemente separadas, criando condições para que o classificador 1NN sem processamento tenha alto desempenho.

Para os casos do *F-Score* e *G-Mean*, nas figuras 5.2 e 5.3, os métodos SOMEntropyHighFilter e SOMEntropyHighLowFilter apresentaram o maior ganho em uma faixa intermediária de nível de sobreposição, entre 1,0 e 3,0 de diferença de média, aproximadamente. Analisando o desbalanceamento, pode-se notar que os métodos têm maiores benefícios para uma faixa intermediária de desbalanceamento entre 15% e 30%. No entanto, em situações com desbalanceamento mais severo, como 5% e 10%, os métodos

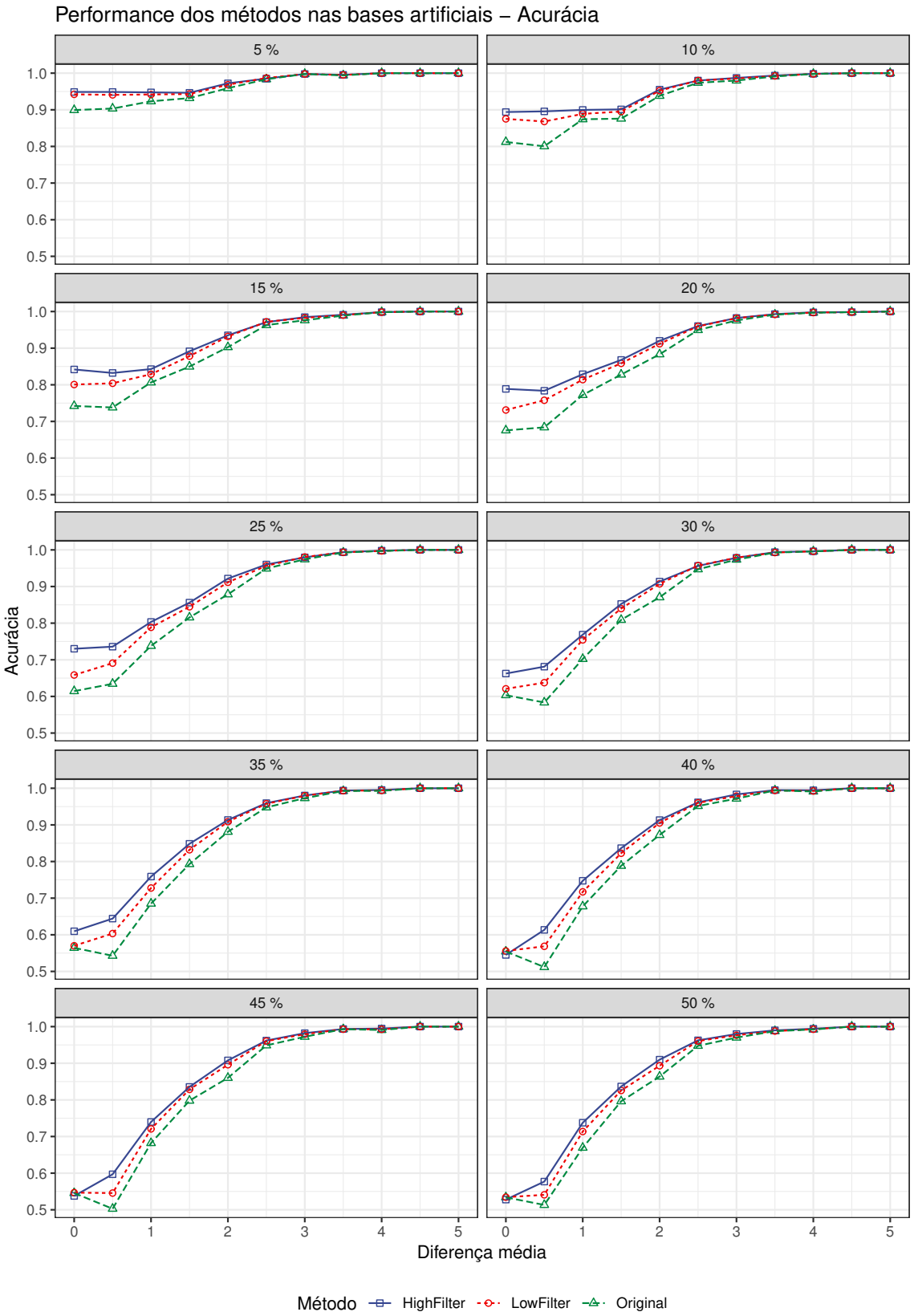


Figura 5.1: Comparação dos valores de acurácia nas bases artificiais para os métodos desenvolvidos nesse trabalho e o 1NN sem pré-processamento (original), o agrupamento foi feito por proporção da classe positiva. Fonte: Elaborada pelo autor.

tiveram ganhos menores em relação ao 1NN sem processamento.

De uma forma geral, em bases com sobreposição, o SOMEntropyHighFilter e logo também o SOMEntropyHighLowFilter, tiveram os maiores ganhos; com o ganho do SOMEntropyLowFilter estando abaixo, mas ainda superior ao método 1NN, nos casos de sobreposição. Esse comportamento suporta a hipótese de que os três métodos são efetivos para tratar casos de sobreposição e desbalanceamento, desde que não estejam em situações mais severas.

Analisando-se o resultado da taxa de redução, que se encontra na Figura 5.4, verifica-se que o SOMEntropyHighFilter e o SOMEntropyHighLowFilter apresentaram, em alguns casos, altas taxas de redução, principalmente nos casos com baixo desbalanceamento e alta sobreposição. Isso pode demonstrar um fato a ser estudado, de que os métodos podem realizar uma redução da base significativa, enquanto mantêm ou melhoram a performance do classificador.

Realizaram-se também as mesmas comparações para os casos reais, verificando-se qual foi o melhor valor para as medidas de desempenho encontradas na base para cada um dos métodos de pré-processamento após a variação dos parâmetros. A Tabela 5.2 traz os resultados com as bases reais em termos de média dos 5 processamentos realizados e desvio-padrão. Para comparação, colocaram-se os resultados do classificador 1NN sem processamento.

Como pode ser observado na Tabela 5.2, para as bases reais, os métodos demonstraram um aumento na acurácia, *F-Score* e no *G-Mean* quando comparados ao 1NN original.

Nas bases reais o SOMEntropyLowFilter foi superior ao SOMEntropyHighFilter em algumas bases de dados. Tal fato parece indicar que existem outros fatores como, por exemplo, a distribuição espacial da base, que não foram analisados nas bases artificiais geradas e que permitem um melhor desempenho do SOMEntropyLowFilter.

As melhoras para os métodos tiveram diferentes dimensões para cada base, com bases com ganhos entre 4,64%, para a base *Hepatitis*, e 0,56%, para a base *Wine*. O motivo para essas diferenças de comportamento será analisado mais adiante no trabalho.

Analisou-se também a taxa de redução das bases reais, conforme a Tabela 5.3. Calculou-se a melhor taxa de redução, sem otimização de desempenho, assim como a taxa de redução nos casos em que os parâmetros estavam configurados para o melhor desempenho em acurácia, *F-Score* e *G-Mean*.

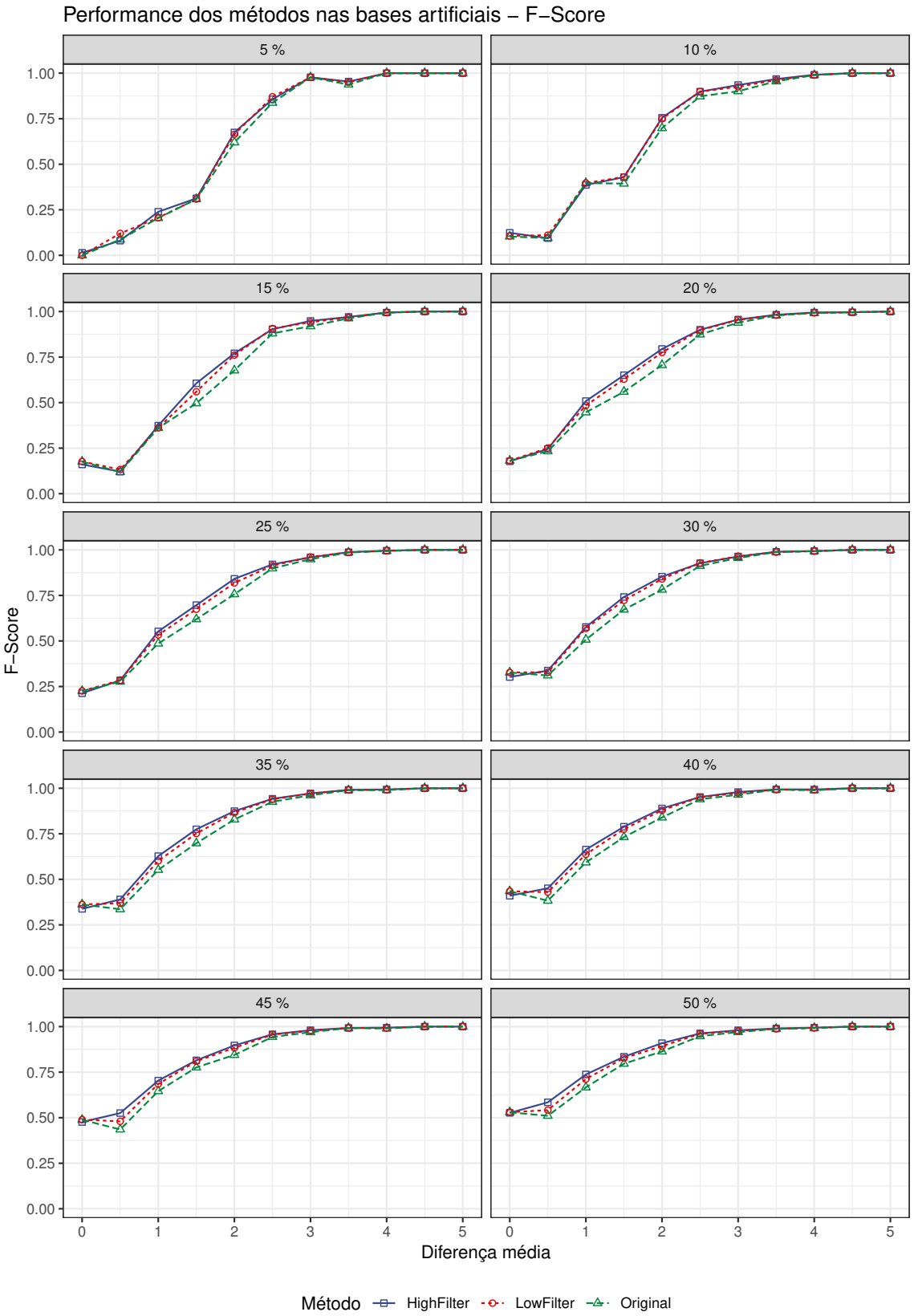


Figura 5.2: Comparação dos valores de *F-Score* nas bases artificiais para os métodos desenvolvidos nesse trabalho e o 1NN sem pré-processamento (original), o agrupamento foi feito por proporção da classe positiva. Fonte: Elaborada pelo autor.

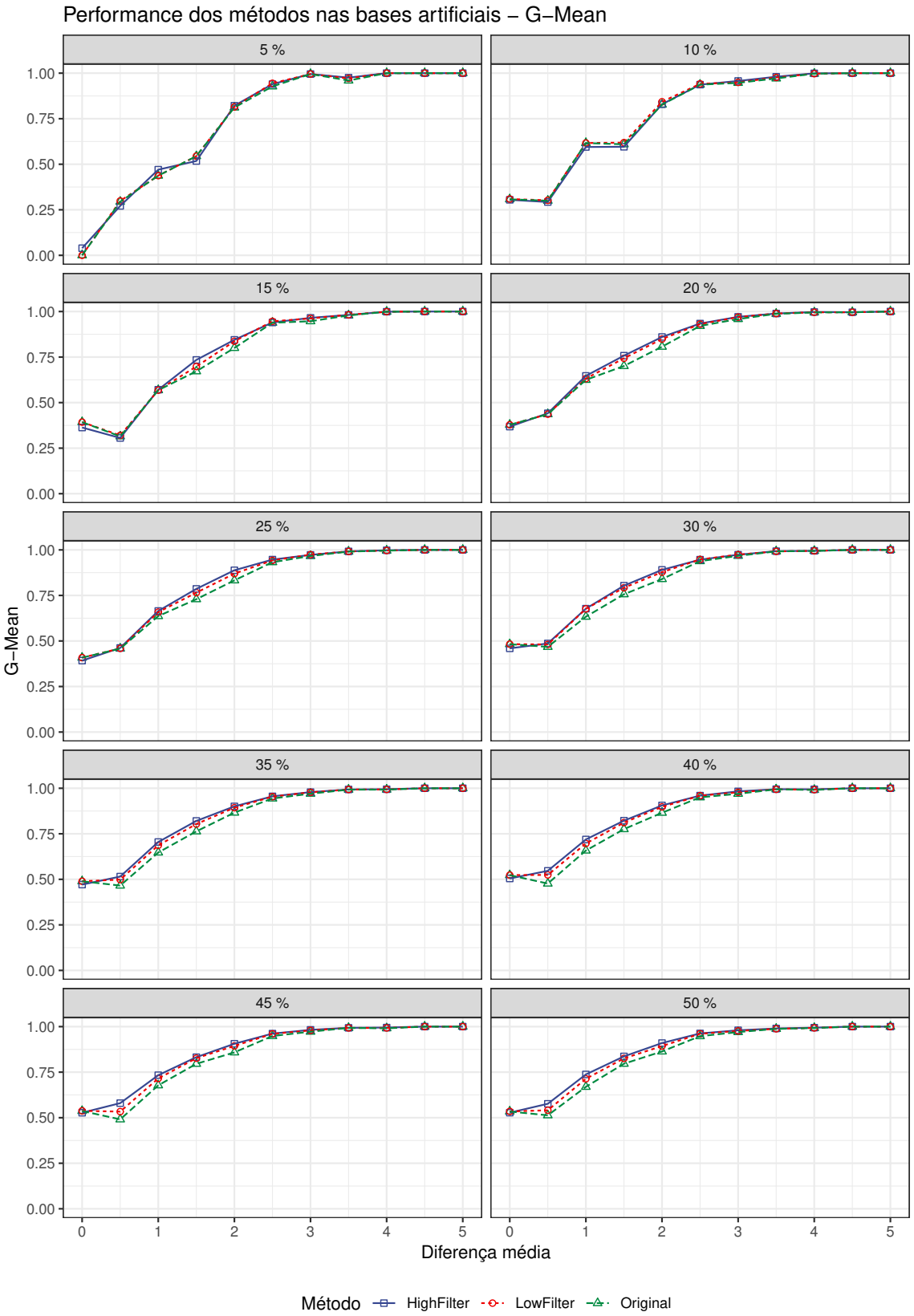


Figura 5.3: Comparação dos valores de *G-Mean* nas bases artificiais para os métodos desenvolvidos nesse trabalho e o 1NN sem pré-processamento (original), o agrupamento foi feito por proporção da classe positiva. Fonte: Elaborada pelo autor.

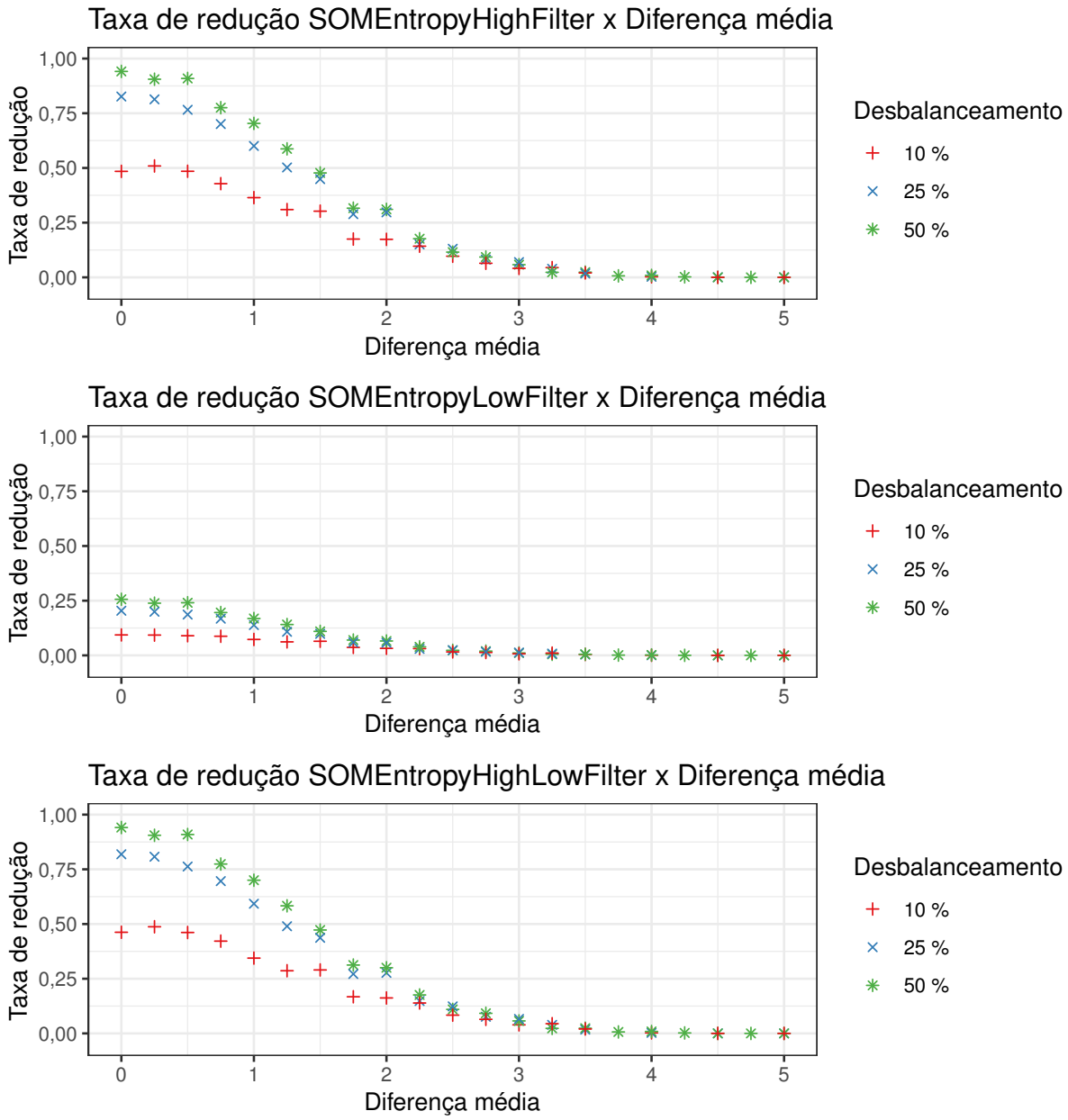


Figura 5.4: Comparação dos valores da taxa de redução nas bases artificiais para os métodos desenvolvidos nesse trabalho. Fonte: Elaborada pelo autor.

Tabela 5.2: Melhor resultado dos diferentes métodos para as bases reais utilizadas nos experimentos

Base de dados	**Método**	**Acurácia**	**F-Score**	**G-Mean**
Ecoli	Original	0.9393 ± 0.0045	0.8082 ± 0.0152	0.8908 ± 0.0127
	HighFilter	0.9613 ± 0.0030	0.8739 ± 0.0085	0.9204 ± 0.0017
	LowFilter	0.9643 ± 0.0042	0.8837 ± 0.0134	0.9271 ± 0.0092
	HighLowFilter	**0.9649 ± 0.0053**	**0.8855 ± 0.0168**	**0.9275 ± 0.0099**
Glass	Original	0.9720 ± 0.0000	0.6667 ± 0.0000	0.8105 ± 0.0000
	HighFilter	0.9748 ± 0.0026	0.6824 ± 0.0215	0.7979 ± 0.0304
	LowFilter	**0.9804 ± 0.0051**	**0.7480 ± 0.0555**	**0.8268 ± 0.0297**
	HighLowFilter	0.9757 ± 0.0039	0.6824 ± 0.0215	0.7979 ± 0.0304
Haberman	Original	0.6634 ± 0.0139	0.3324 ± 0.0320	0.4996 ± 0.0291
	HighFilter	0.7255 ± 0.0276	**0.3477 ± 0.0304**	**0.5078 ± 0.0264**
	LowFilter	0.7196 ± 0.0152	0.3359 ± 0.0315	0.5018 ± 0.0269
	HighLowFilter	**0.7366 ± 0.0132**	0.3458 ± 0.0287	0.5060 ± 0.0257
Heart	Original	0.7617 ± 0.0108	0.7386 ± 0.0120	0.7591 ± 0.0109
	HighFilter	**0.8185 ± 0.0109**	**0.8018 ± 0.0088**	**0.8167 ± 0.0094**
	LowFilter	0.7947 ± 0.0180	0.7699 ± 0.0182	0.7900 ± 0.0172
	HighLowFilter	**0.8185 ± 0.0109**	**0.8018 ± 0.0088**	**0.8167 ± 0.0094**
Hepatitis	Original	0.8013 ± 0.0161	0.5067 ± 0.0296	0.6595 ± 0.0205
	HighFilter	0.8413 ± 0.0149	0.5821 ± 0.0438	0.7022 ± 0.0384
	LowFilter	**0.8477 ± 0.0126**	**0.6096 ± 0.0369**	**0.7291 ± 0.0279**
	HighLowFilter	0.8465 ± 0.0029	0.5965 ± 0.0060	0.7173 ± 0.0263
Iris	Original	0.9547 ± 0.0030	0.9659 ± 0.0023	0.9509 ± 0.0022
	HighFilter	**0.9587 ± 0.0056**	0.9688 ± 0.0042	**0.9570 ± 0.0067**
	LowFilter	0.9573 ± 0.0037	0.9678 ± 0.0027	0.9550 ± 0.0050
	HighLowFilter	**0.9587 ± 0.0119**	**0.9689 ± 0.0089**	**0.9570 ± 0.0067**
Libra	Original	0.9911 ± 0.0012	0.9773 ± 0.0033	0.9775 ± 0.0032
	HighFilter	**0.9917 ± 0.0020**	**0.9787 ± 0.0051**	**0.9789 ± 0.0050**
	LowFilter	0.9911 ± 0.0012	0.9773 ± 0.0033	0.9775 ± 0.0032
	HighLowFilter	**0.9917 ± 0.0020**	**0.9787 ± 0.0051**	**0.9789 ± 0.0050**
Mamographic	Original	0.7536 ± 0.0073	0.7307 ± 0.0069	0.7508 ± 0.0069
	HighFilter	0.7879 ± 0.0092	0.7766 ± 0.0105	0.7879 ± 0.0099
	LowFilter	0.7869 ± 0.0052	0.7762 ± 0.0050	0.7876 ± 0.0051
	HighLowFilter	**0.7983 ± 0.0060**	**0.7866 ± 0.0064**	**0.7986 ± 0.0060**
Pima	Original	0.7109 ± 0.0080	0.5694 ± 0.0107	0.6613 ± 0.0085
	HighFilter	0.7362 ± 0.0117	0.5900 ± 0.0124	0.6761 ± 0.0102
	LowFilter	0.7411 ± 0.0053	0.6036 ± 0.0086	0.6870 ± 0.0069
	HighLowFilter	**0.7500 ± 0.0038**	**0.6129 ± 0.0080**	**0.6935 ± 0.0066**
SPECTF-Heart	Original	0.6933 ± 0.0061	0.3507 ± 0.0098	0.5530 ± 0.0096
	HighFilter	**0.7925 ± 0.0057**	0.3767 ± 0.0430	0.5703 ± 0.0398
	LowFilter	0.7343 ± 0.0166	0.3753 ± 0.0315	0.5644 ± 0.0364
	HighLowFilter	**0.7925 ± 0.0057**	**0.3850 ± 0.0493**	**0.5777 ± 0.0459**
Wine	Original	0.9506 ± 0.0025	0.9339 ± 0.0036	0.9360 ± 0.0034
	HighFilter	0.9551 ± 0.0000	0.9406 ± 0.0092	0.9437 ± 0.0121
	LowFilter	0.9528 ± 0.0031	0.9371 ± 0.0044	0.9390 ± 0.0041
	HighLowFilter	**0.9562 ± 0.0025**	**0.9420 ± 0.0034**	**0.9441 ± 0.0033**
Wiscosin	Original	0.9577 ± 0.0033	0.9381 ± 0.0050	0.9509 ± 0.0045
	HighFilter	0.9694 ± 0.0030	0.9559 ± 0.0043	0.9678 ± 0.0038
	LowFilter	0.9677 ± 0.0048	0.9534 ± 0.0070	0.9657 ± 0.0056
	HighLowFilter	**0.9714 ± 0.0027**	**0.9589 ± 0.0039**	**0.9709 ± 0.0035**

Fonte: Elaborada pelo autor.

Tabela 5.3: Taxa de redução dos diferentes métodos para as bases reais

Base de dados	**Método**	**Red. Otimizado redução**	**Red. Otimizado acurácia**	**Red. Otimizado F-Score**	**Red. Otimizado G-Mean**
Ecoli	HighFilter	**0.1980 ± 0.0068**	**0.1325 ± 0.0050**	**0.1325 ± 0.0050**	**0.1325 ± 0.0050**
	LowFilter	0.0360 ± 0.0016	0.0362 ± 0.0017	0.0362 ± 0.0017	0.0349 ± 0.0021
	HighLowFilter	0.1877 ± 0.0060	0.0426 ± 0.0020	0.0412 ± 0.0025	0.0412 ± 0.0025
Glass	HighFilter	**0.0914 ± 0.0056**	0.0181 ± 0.0041	0.0181 ± 0.0041	0.0116 ± 0.0034
	LowFilter	0.0206 ± 0.0010	0.0105 ± 0.0017	0.0068 ± 0.0010	0.0068 ± 0.0010
	HighLowFilter	0.0778 ± 0.0059	**0.0210 ± 0.0038**	**0.0181 ± 0.0040**	**0.0117 ± 0.0034**
Haberman	HighFilter	**0.8707 ± 0.0100**	**0.7548 ± 0.0088**	**0.1763 ± 0.0095**	0.0728 ± 0.0106
	LowFilter	0.2007 ± 0.0042	0.2006 ± 0.0044	0.0013 ± 0.0007	0.0013 ± 0.0007
	HighLowFilter	0.8015 ± 0.0110	0.5167 ± 0.0131	**0.1763 ± 0.0095**	**0.1763 ± 0.0095**
Heart	HighFilter	**0.7479 ± 0.0124**	**0.4717 ± 0.0100**	0.4966 ± 0.0162	0.4966 ± 0.0162
	LowFilter	0.1637 ± 0.0053	0.1377 ± 0.0046	0.1377 ± 0.0046	0.1377 ± 0.0046
	HighLowFilter	0.6937 ± 0.0184	**0.4717 ± 0.0100**	**0.4967 ± 0.0163**	**0.4967 ± 0.0163**
Hepatitis	HighFilter	**0.5868 ± 0.0235**	**0.2179 ± 0.0086**	**0.2179 ± 0.0086**	**0.2179 ± 0.0086**
	LowFilter	0.1187 ± 0.0055	0.1029 ± 0.0040	0.1029 ± 0.0040	0.1029 ± 0.0040
	HighLowFilter	0.5313 ± 0.0179	0.1757 ± 0.0134	0.1400 ± 0.0062	0.1400 ± 0.0062
Iris	HighFilter	**0.2084 ± 0.0102**	**0.0868 ± 0.0139**	**0.0868 ± 0.0139**	**0.1458 ± 0.0140**
	LowFilter	0.0353 ± 0.0019	0.0271 ± 0.0028	0.0271 ± 0.0028	0.0176 ± 0.0021
	HighLowFilter	0.1674 ± 0.0106	**0.0868 ± 0.0139**	**0.0868 ± 0.0139**	0.1430 ± 0.0141
Libra	HighFilter	**0.1025 ± 0.0104**	0.0127 ± 0.0031	0.0127 ± 0.0031	0.0127 ± 0.0031
	LowFilter	0.0214 ± 0.0017	0.0056 ± 0.0008	0.0056 ± 0.0008	0.0056 ± 0.0008
	HighLowFilter	0.0932 ± 0.0102	**0.0159 ± 0.0030**	**0.0159 ± 0.0030**	**0.0159 ± 0.0030**
Mamographic	HighFilter	**0.7508 ± 0.0052**	**0.6761 ± 0.0070**	**0.6761 ± 0.0070**	**0.6761 ± 0.0070**
	LowFilter	0.1564 ± 0.0011	0.1575 ± 0.0018	0.1575 ± 0.0018	0.1575 ± 0.0018
	HighLowFilter	0.6542 ± 0.0099	0.2227 ± 0.0046	0.2227 ± 0.0046	0.2227 ± 0.0046
Pima	HighFilter	**0.7084 ± 0.0028**	**0.4820 ± 0.0084**	**0.3764 ± 0.0059**	**0.3764 ± 0.0059**
	LowFilter	0.1685 ± 0.0028	0.1424 ± 0.0038	0.1156 ± 0.0021	0.1156 ± 0.0021
	HighLowFilter	0.7027 ± 0.0063	0.2902 ± 0.0076	0.2667 ± 0.0062	0.2667 ± 0.0062
SPECTF-Heart	HighFilter	**0.5738 ± 0.0092**	**0.5674 ± 0.0088**	0.1790 ± 0.0150	**0.1394 ± 0.0133**
	LowFilter	0.1538 ± 0.0056	0.1496 ± 0.0066	0.0743 ± 0.0049	0.0743 ± 0.0049
	HighLowFilter	0.5610 ± 0.0056	**0.5674 ± 0.0088**	**0.1793 ± 0.0149**	**0.1394 ± 0.0133**
Wine	HighFilter	**0.2320 ± 0.0115**	**0.2336 ± 0.0176**	**0.2336 ± 0.0176**	**0.2336 ± 0.0176**
	LowFilter	0.0382 ± 0.0030	0.0250 ± 0.0019	0.0250 ± 0.0019	0.0250 ± 0.0019
	HighLowFilter	0.1725 ± 0.0119	0.1546 ± 0.0129	0.1546 ± 0.0129	0.1546 ± 0.0129
Wiscosin	HighFilter	**0.0803 ± 0.0014**	**0.0689 ± 0.0025**	**0.0689 ± 0.0025**	**0.0689 ± 0.0025**
	LowFilter	0.0164 ± 0.0012	0.0144 ± 0.0007	0.0144 ± 0.0007	0.0144 ± 0.0007
	HighLowFilter	**0.0803 ± 0.0014**	0.0271 ± 0.0016	0.0281 ± 0.0015	0.0281 ± 0.0015

Fonte: Elaborada pelo autor.

Analisando a questão da redução, conforme a Tabela 5.2, verifica-se que as taxas de redução têm valores variados. Pode-se ressaltar que, em algumas bases, o SOMEntropyHighFilter e SOMEntropyHighLowFilter apresentaram taxas de redução significativas, como em *Haberman* (87,07%), *Heart* (74,79%), *Mamographic* (75,08%) e *Pima* (70,84%). Considerando-se apenas as reduções que ocorreram com os parâmetros otimizados para o melhor desempenho, obtiveram-se taxas de redução superiores a 45%, como nos casos da base *Mamographic* (67,61% para acurácia, *F-Score* e *G-Mean*) e *Heart* (47,17% para acurácia e 49,67% para *F-Score* e *G-Mean*).

Considerando que os métodos propostos são focados na melhora do desempenho, as taxas de redução obtidas foram um resultado interessante, que seria um tema importante a ser aprofundado em trabalhos futuros.

Para validar que as análises realizadas até agora estão corretas de um ponto de vista estatístico, decidiu-se utilizar um modelo que permita concluir que os métodos desenvolvidos apresentem diferença relevante com relação ao 1NN.

Na estatística, o método usualmente utilizado para comparar resultados médios é a tabela de análise de variância (ANOVA) (MORETTIN; BUSSAB, 2010). Porém, para uso do ANOVA, é premissa que o conjunto de dados possua duas características: distribuição normal e homogeneidade de variância.

Sendo assim, para verificar se os resultados experimentais deste trabalho, resumidos nas tabelas 5.1 e 5.3, atendem à premissa para aplicação do ANOVA, realizou-se o teste de Shapiro-Wilk, que valida a distribuição normal, e o teste de Levene, em que se valida a homogeneidade de variância (ZAR, 2014). Para os testes, escolheram-se como base os desempenhos das bases reais para a medida *G-Mean*. Para a realização dos testes, utilizaram-se os métodos *shapiro.test* dos pacotes R *fbasic* (WUERTZ; SETZ; CHALABI, 2017) e *leveneTest* do pacote *car* (FOX; WEISBERG, 2011).

O teste de Levene tem como hipótese nula que as bases não possuem homogeneidade de variância (ZAR, 2014). O valor calculado de p nesse teste foi de 0,7813 que, considerando um α de 0,05, permite nos concluir que a base possui homogeneidade de variância.

No caso do teste de Shapiro-Wilk, a hipótese é que de que os resultados tem uma distribuição normal (ZAR, 2014). No caso do teste realizado, essa hipótese foi rejeitada com um valor de p de $2,2 \times 10^{-16}$ que, considerando um α de 0,05, indica que a base de desempenho não possui distribuição normal.

A distribuição não-normal é uma característica comum na validação de desempenho entre algoritmos de classificação (DEMŠAR, 2006). Por isso, devem-se utilizar métodos não paramétricos para a comparação (DEMŠAR, 2006). Um desses métodos, que é equivalente ao ANOVA, é o teste de Friedman, que realiza a comparação por meio do ranqueamento dos resultados (FACELI et al., 2011).

Realizou-se o teste de Friedman nos resultados de bases artificiais e reais, descritos anteriormente nas tabelas 5.1 e 5.2. Para o cálculo dos valores, decidiu-se utilizar o método *friedman.test* do pacote R *PMCMR* (POHLERT, 2014).

Os valores de p encontrados foram consolidados na Tabela 5.4. Observa-se que os valores são todos menores que $0,001$, portanto, pode-se rejeitar a hipótese nula de que os métodos são iguais para um α de até $0,001$.

Tabela 5.4: Valores de p calculados utilizando-se o teste de Friedman

Base de dados	p **Acurácia**	p **F-Score**	p **G-Mean**
Artificiais	$2,20 \times 10^{-16}$	$2,20 \times 10^{-16}$	$2,20 \times 10^{-16}$
Reais	$5,74 \times 10^{-6}$	$8,14 \times 10^{-6}$	$1,11 \times 10^{-4}$

Fonte: Elaborada pelo autor.

No entanto, o teste de Friedman apenas permite identificar que existem diferenças entre os desempenhos dos métodos, não informando entre quais métodos essa diferença ocorre. Sendo assim, deve-se proceder com um pós-teste para realizar a comparação dos métodos com o método de controle (FACELI et al., 2011) .

Como pós-teste, decidiu-se utilizar o teste *Wilcoxon signed-ranks*. Esse é um teste não-paramétrico que permite a comparação entre algoritmos, por meio do ranqueamento das diferenças das medidas de desempenho. A sua hipótese nula indica que os algoritmos têm comportamentos semelhantes (FACELI et al., 2011). Para realização do teste, utilizou-se o método *wilcox.test* do pacote R *MASS* (VENABLES; RIPLEY, 2002).

Os testes foram realizados, comparando-se os 3 métodos desenvolvidos contra o 1NN sem processamento para as bases artificiais e reais. Os resultados se encontram na Tabela 5.5. Como os valores de p foram inferiores a $0,01$, podemos rejeitar a hipótese nula que os métodos sejam iguais ao método de controle, com uma confiança de até 99%, e podemos considerar que os métodos trazem desempenhos diferentes do 1NN.

Para encontrar os resultados de desempenho para as diferentes bases de dados de analisadas anteriormente, foi necessário testar diferentes níveis de valores dos parâmetros *ThresholdHigh* e *ThresholdLow*, além do tamanho de mapa, por meio da variação da cons-

Tabela 5.5: Valores de p calculado usando o teste par a par com o teste *Wilcoxon signed-ranks*

Base de dados	**Comparação Método**	p **Acurácia**	p **F-Score**	p **G-Mean**
Artificiais	HighFilter - Original	1.03×10^{-15}	1.05×10^{-11}	1.67×10^{-7}
	LowFilter - Original	2.60×10^{-16}	5.46×10^{-15}	8.00×10^{-15}
	HighLowFilter - Original	6.85×10^{-16}	3.04×10^{-12}	7.28×10^{-8}
Reais	HighFilter - Original	4.88×10^{-4}	4.88×10^{-3}	4.88×10^{-3}
	LowFilter - Original	3.86×10^{-3}	3.86×10^{-3}	3.86×10^{-3}
	HighLowFilter - Original	4.88×10^{-4}	4.88×10^{-3}	4.88×10^{-3}

Fonte: Elaborada pelo autor.

tante C_{Mapa}. A necessidade de identificar o melhor valor para esses parâmetros apresenta um aumento na complexidade do classificador. Para minimizar esse impacto, investigou-se a existência de valores de parâmetros ideais, procurando-se estudar o comportamento da variação dos parâmetros contra o valor do *F-Score*.

Para avaliar o parâmetro C_{Mapa}, que define o tamanho do mapa de SOM, calculou-se o ganho máximo em *F-Score* obtido para cada valor da constante para as bases reais. Em sequência, ranquearam-se os dados de acordo com melhor desempenho, considerando o agrupamento por base e por método. Um exemplo desse processo está descrito na Tabela 5.6, em que está disponibilizado o resumo dos valores de *F-Score* obtidos para a base *Ecoli* em conjunto com a posição relativa na comparação de valores de C_{Mapa}.

Com os dados ranqueados, calculou-se a média por método para cada valor de C_{Mapa}, os valores encontrados foram disponibilizados na Tabela 5.7, destacando-se em negrito os valores de C_{Mapa} que tiveram a melhor posição.

Tabela 5.6: Exemplo de ranqueamento do valor de *F-Score* por C_{Mapa} para a base *Ecoli*

	Valor máximo *F-Score*		
$\boldsymbol{C_{Mapa}}$	**HighFilter**	**LowFilter**	**HighLowFilter**
-2	0.8414 (8º)	0.8770 (1º)	0.8802 (1º)
-1	0.8618 (3º)	0.8690 (2º)	0.8740 (3º)
0	0.8707 (1º)	0.8636 (3º)	0.8786 (2º)
1	0.8659 (2º)	0.8589 (4º)	0.8721 (4º)
2	0.8562 (5º)	0.8360 (5º)	0.8622 (5º)
3	0.8584 (4º)	0.8328 (6º)	0.8597 (6º)
4	0.8536 (7º)	0.8201 (8º)	0.8590 (7º)
5	0.8551 (6º)	0.8230 (7º)	0.8563 (8º)

Fonte: Elaborada pelo autor.

Um baixo valor de média do ranqueamento indica que a respectiva constante de mapa traz, considerando a média entre as bases, o melhor resultado. Portanto, os valores

Tabela 5.7: Média do Ranqueamento do valor de *F-Score* por C_{Mapa}

	Média do ranqueamento de *F-score*		
$\boldsymbol{C_{Mapa}}$	**HighFilter**	**LowFilter**	**HighLowFilter**
-2	4.33	3.17	**3.00**
-1	4.17	**3.00**	3.50
0	**3.58**	**3.00**	3.25
1	4.83	3.17	4.33
2	4.83	4.08	4.33
3	4.25	4.83	5.08
4	4.17	5.92	5.42
5	5.75	6.08	7.00

Fonte: Elaborada pelo autor.

indicam que os melhores tamanhos de mapa se concentram nas faixas de valores menores da constante de mapa C_{Mapa}, indicando que mapas menores podem ser os mais efetivos para os métodos.

Para os parâmetros de *threshold*, decidiu-se verificar a variação desses parâmetros de forma independente para os métodos SOMEntropyHighFilter e SOMEntropyLowFilter. Inicialmente, verificou-se o resultado para as bases artificias em 3 níveis diferentes de balanceamento, 50%, 25% e 10%. O resultado dessa variação se encontra nas figuras 5.5 a 5.10.

O resultado, para os casos de diferença de média entre 4,5 e 5, indica que, a partir de um determinado valor de diferença entre as médias, os métodos não alcançaram ganhos significativos independentemente do valor de parâmetro de *threshold*. Isso ocorre porque não existe sobreposição para os métodos agirem no pré-processamento.

Verifica-se, também, que o comportamento se divide em dois cenários a partir de um certo valor de sobreposição. Com valores menores, abaixo dessa sobreposição, o método não tem um comportamento padrão, apresentando diferentes valores de threshold para o ponto ótimo. Já com valores maiores que esse valor de sobreposição, a base tem um comportamento padrão em que os melhores valores de *threshold* se encontram no ponto mais agressivo. Para o caso da base balanceada, esse ponto de divisão ocorre no caso da base com maior sobreposição com as médias iguais. Conforme se aumenta o desbalanceamento esse ponto é deslocado, incluindo, nos casos de comportamento não padrão, bases com menor sobreposição.

O mesmo foi feito para as bases reais. Os resultados da variação desses parâmetros se encontra nas figuras 5.11 e 5.12.

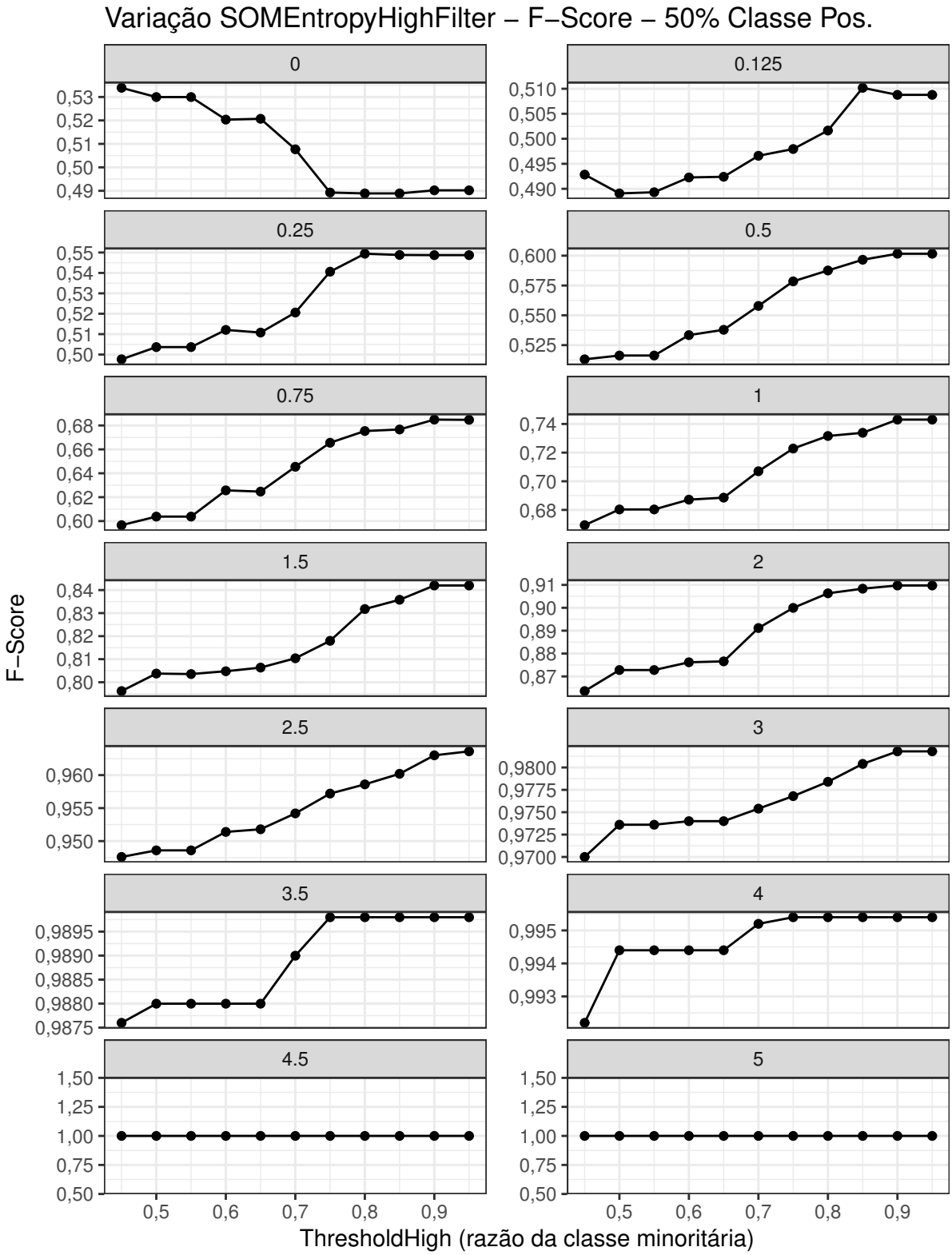


Figura 5.5: Efeito do SOMEntropyHighFilter no *F-Score* nas bases de dados artificiais com 50% da classe positiva, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito pela diferença de média da base. O ponto de 0,45 representa o ponto inicial sem pré-processamento. Fonte: Elaborada pelo autor.

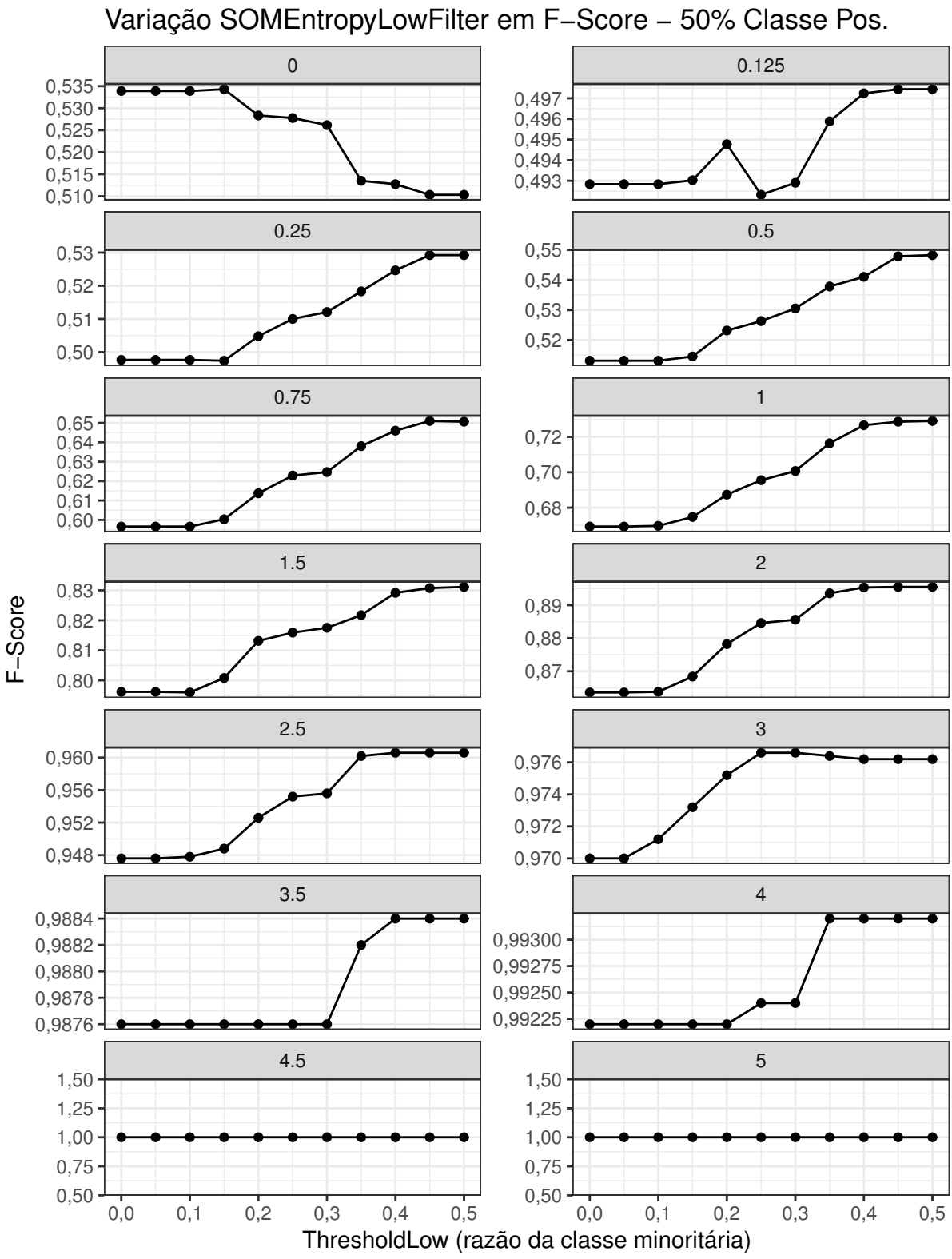


Figura 5.6: Efeito do SOMEntropyLowFilter no *F-Score* nas bases artificiais com 50% da classe positiva, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito pela diferença de média da base. O ponto de 0,0 representa o ponto inicial sem pré-processamento e o ponto de 0,5 representa entropia menor, mas não igual, a 1. Fonte: Elaborada pelo autor.

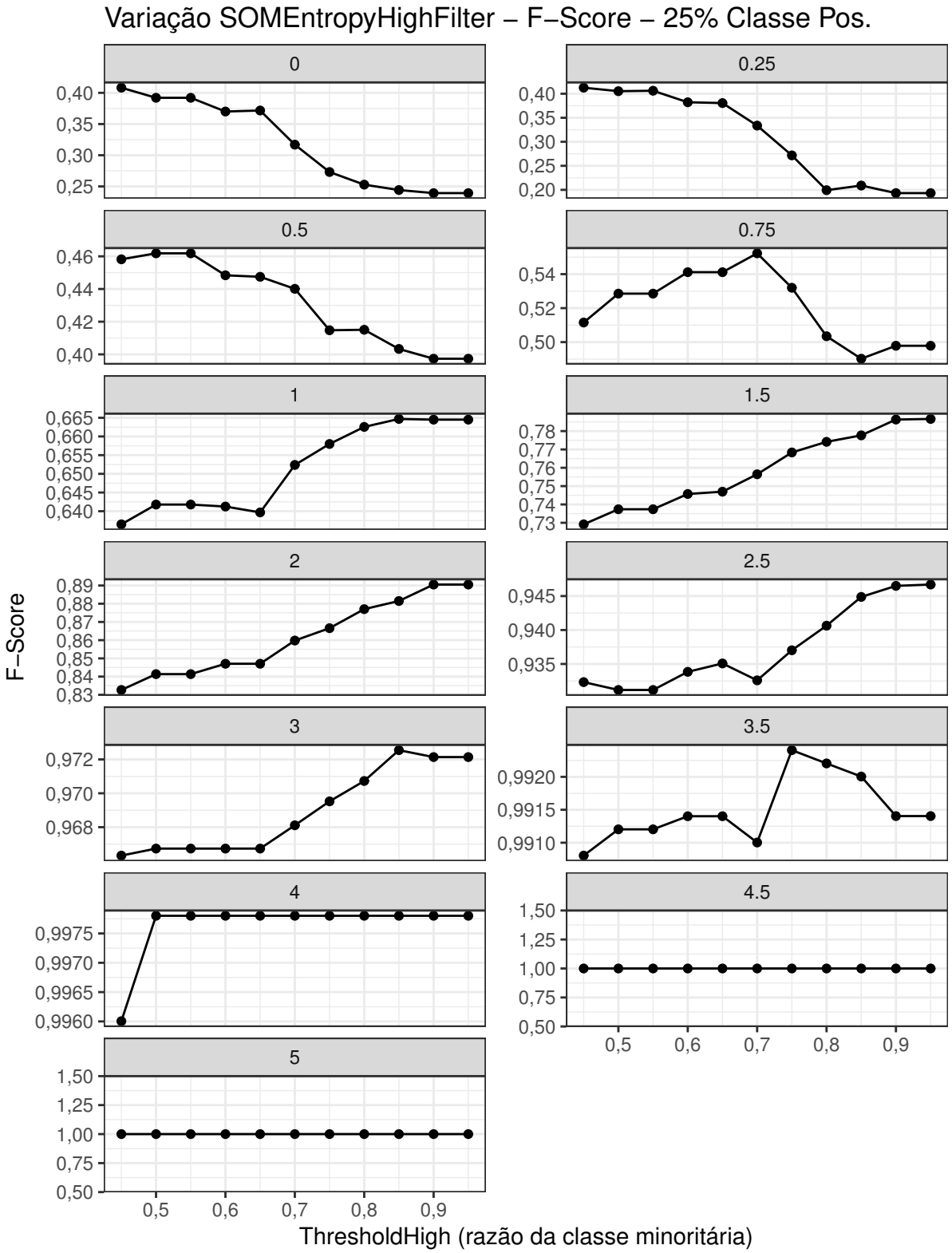


Figura 5.7: Efeito do SOMEntropyHighFilter no *F-Score* nas bases de dados artificiais com 25% da classe positiva, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito pela diferença de média da base. O ponto de 0,45 representa o ponto inicial sem pré-processamento. Fonte: Elaborada pelo autor.

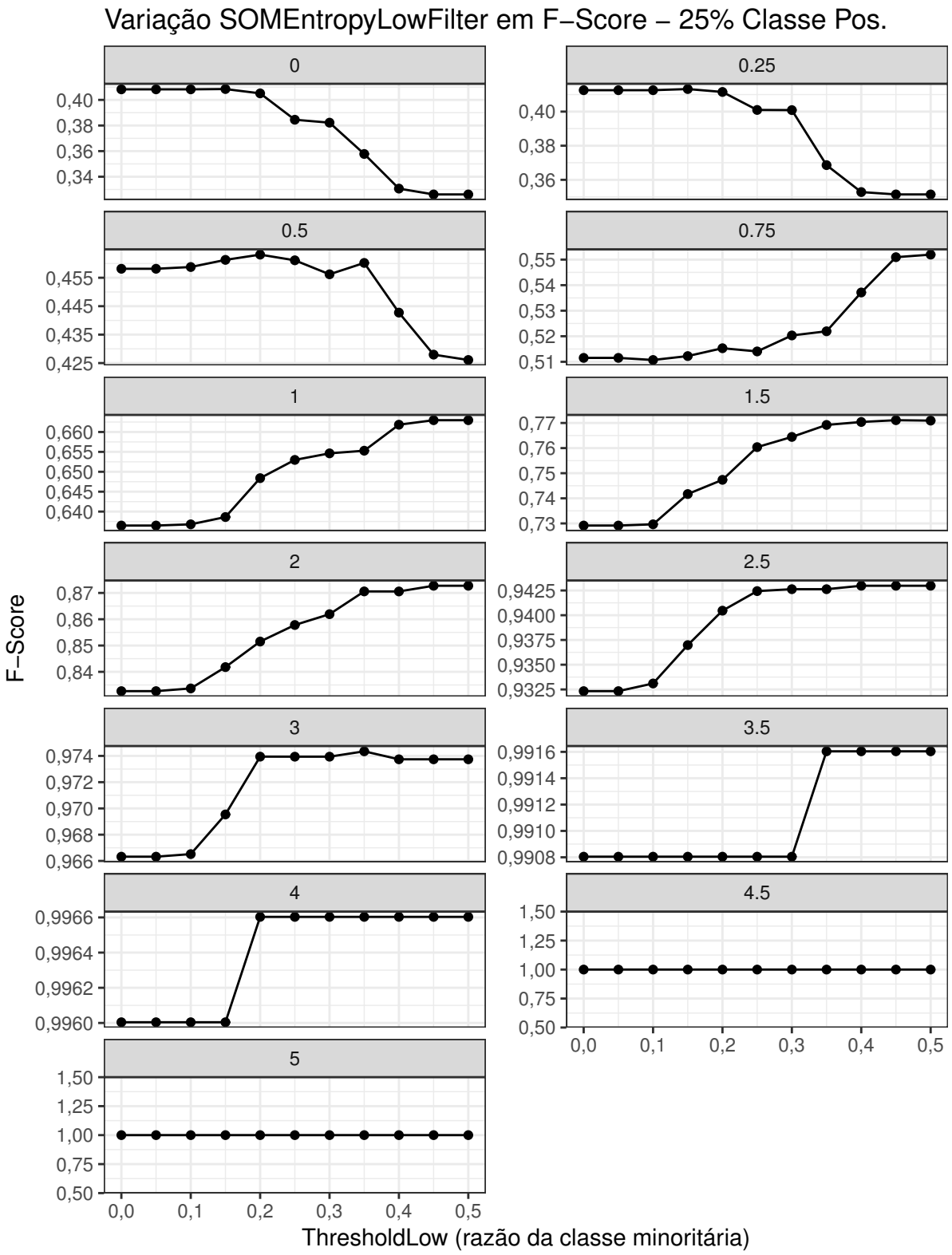


Figura 5.8: Efeito do SOMEntropyLowFilter no *F-Score* nas bases artificiais com 25% da classe positiva, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito pela diferença de média da base. O ponto de 0,0 representa o ponto inicial sem pré-processamento e o ponto de 0,5 representa entropia menor, mas não igual, a 1. Fonte: Elaborada pelo autor.

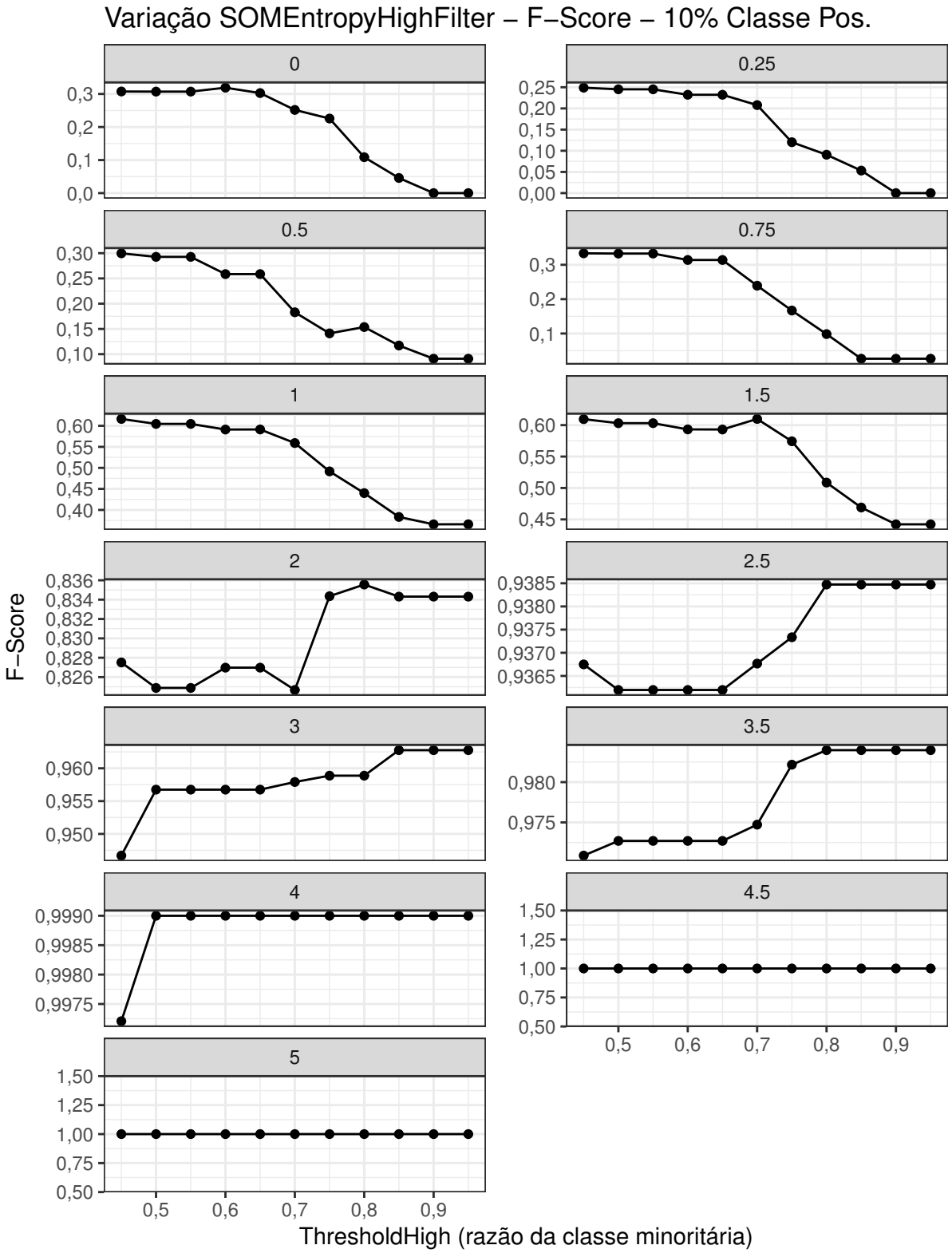


Figura 5.9: Efeito do SOMEntropyHighFilter no *F-Score* nas bases de dados artificiais com 10% da classe positiva, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito pela diferença de média da base. O ponto de 0,45 representa o ponto inicial sem pré-processamento. Fonte: Elaborada pelo autor.

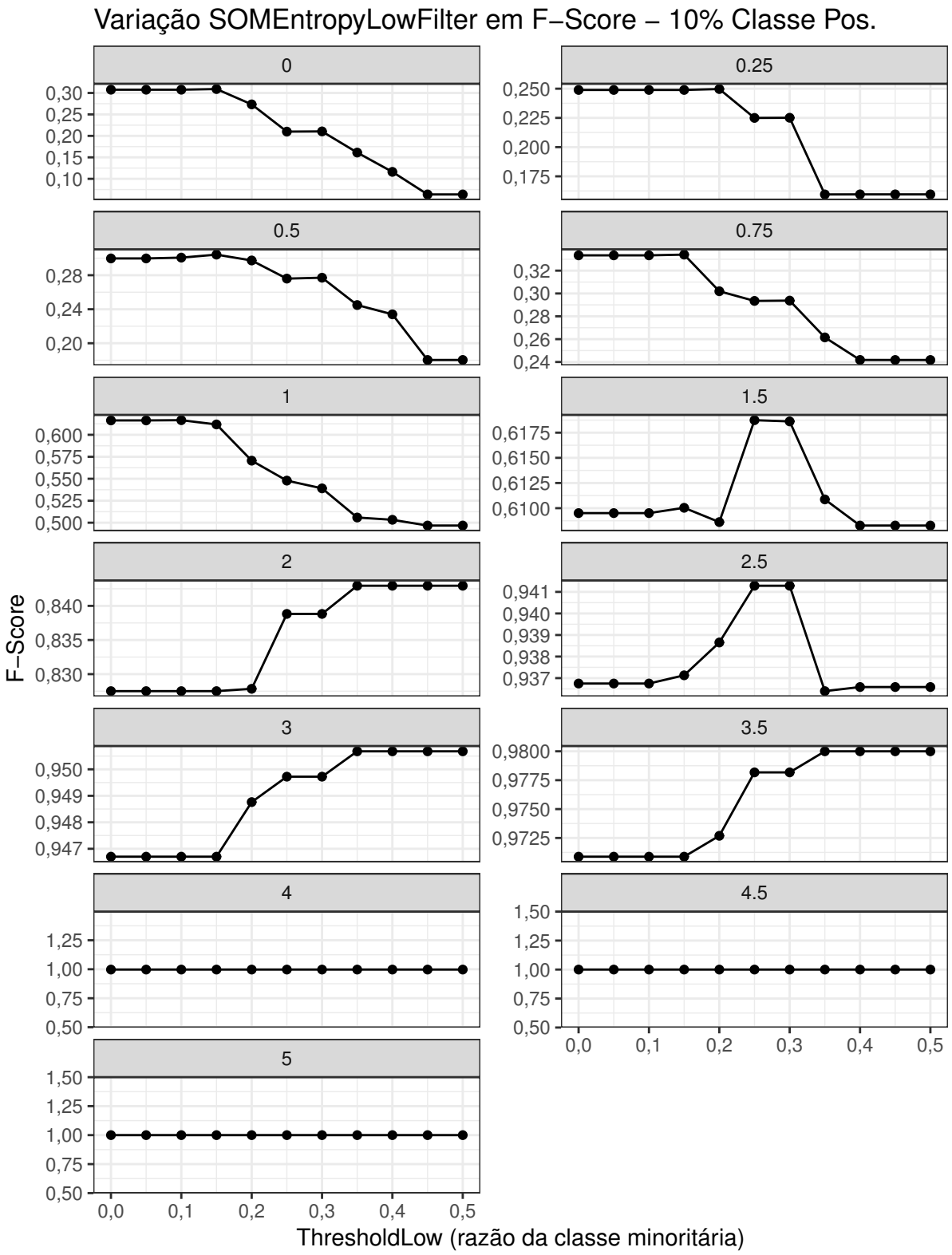


Figura 5.10: Efeito do SOMEntropyLowFilter no *F-Score* nas bases artificiais com 10% da classe positiva, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito pela diferença de média da base. O ponto de 0,0 representa o ponto inicial sem pré-processamento e o ponto de 0,5 representa entropia menor, mas não igual, a 1. Fonte: Elaborada pelo autor.

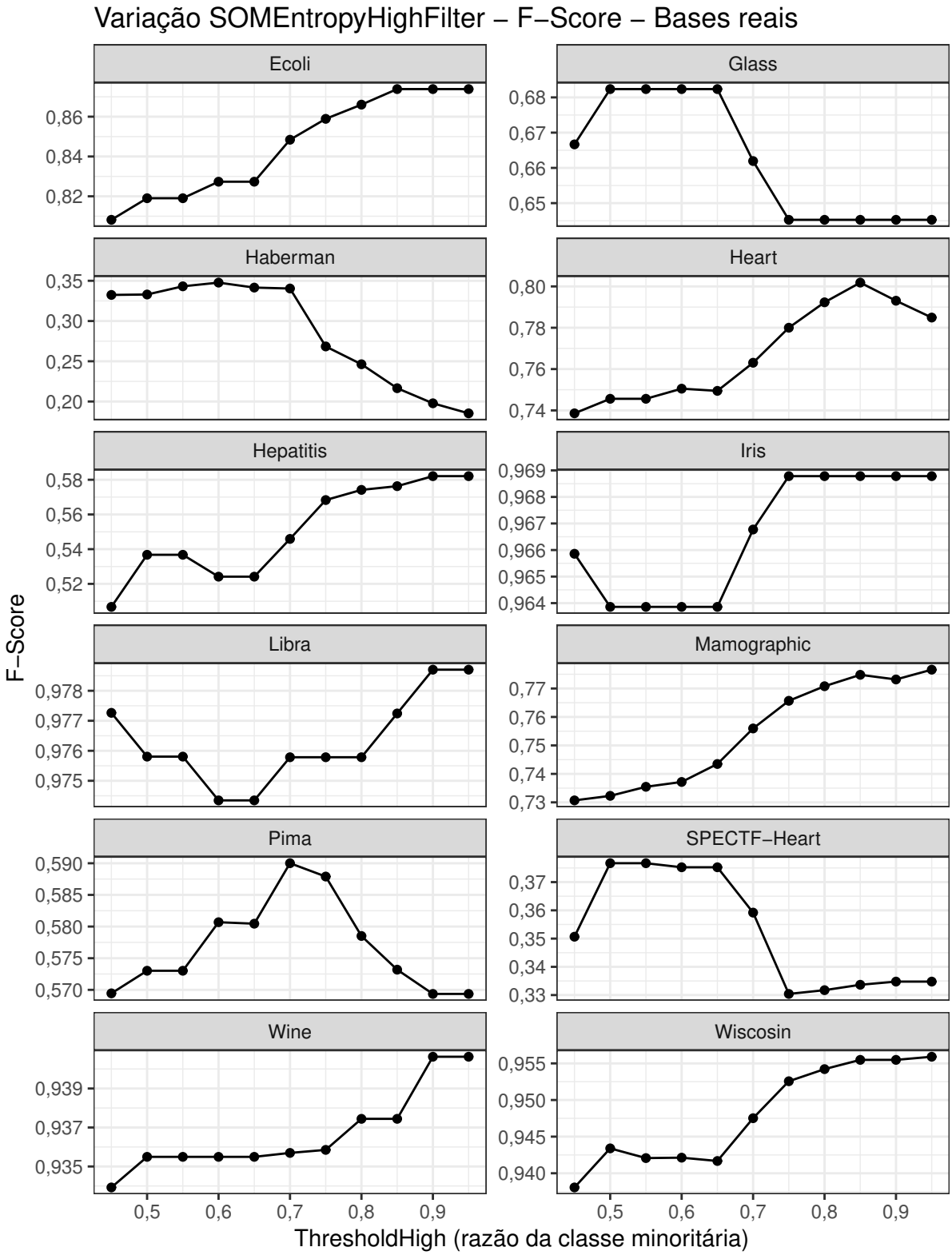


Figura 5.11: Efeito do SOMEntropyHighFilter no *F-Score* nas bases de dados reais, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito por base. O ponto de 0,45 representa o ponto inicial sem pré-processamento. Fonte: Elaborada pelo autor.

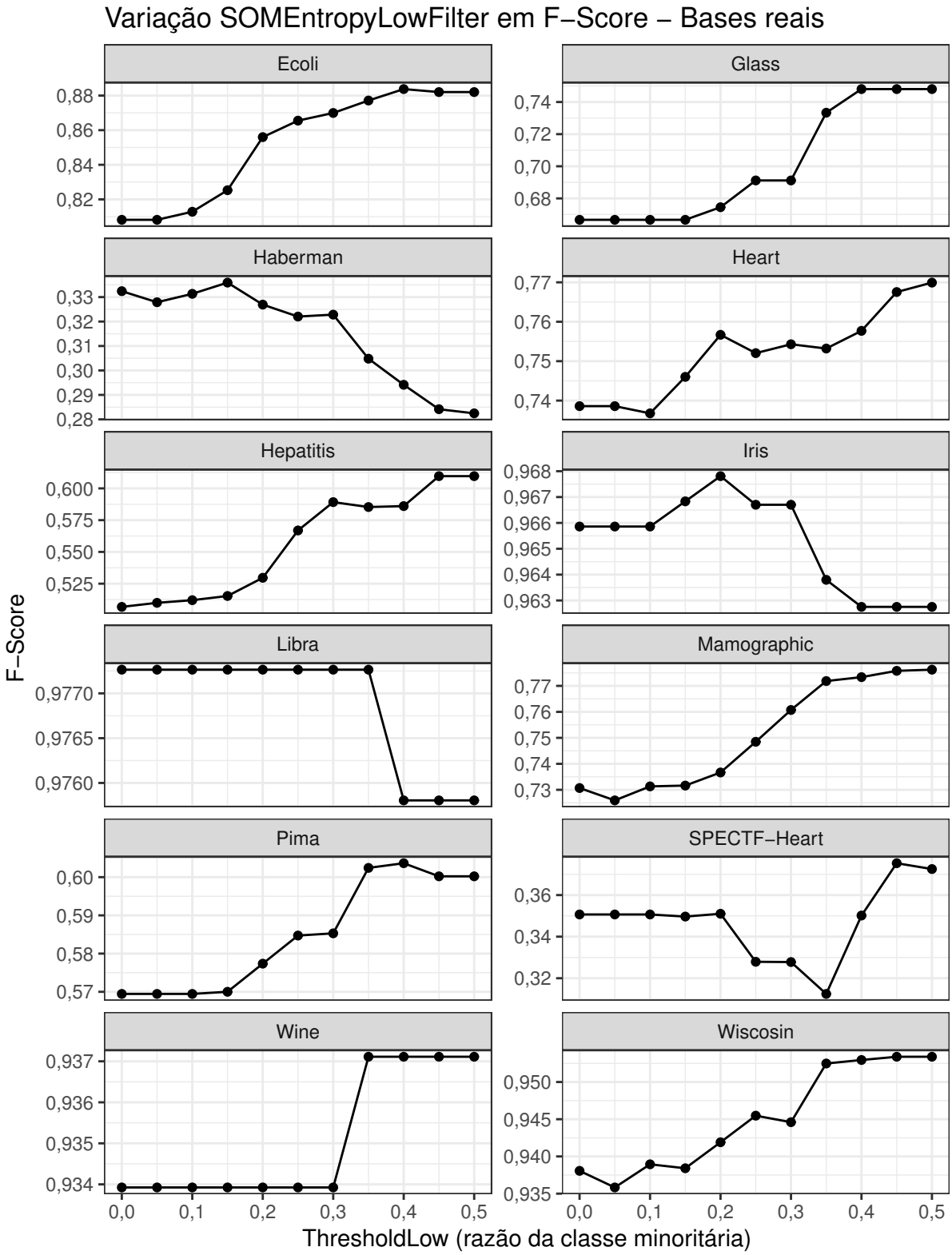


Figura 5.12: Efeito do SOMEntropyLowFilter no *F-Score* nas bases reais, conforme o processo é ajustado para ser mais agressivo na remoção de exemplares, o agrupamento foi feito por base. O ponto de 0,0 representa o ponto inicial sem pré-processamento e o ponto de 0,5 representa entropia menor, mas não igual, a 1. Fonte: Elaborada pelo autor.

O SOMEntropyHighFilter, disponível na Figura 5.11, parece ter um comportamento sem um padrão claro. Apesar de demonstrar uma melhoria em diversas bases, ele não parece ter uma tendência de comportamento que pareça apontar para um valor padrão de *ThresholdHigh*. Como o SOMEntropyHighLowFilter também usa esse filtro, o mesmo se aplica a ele.

Já com o SOMEntropyLowFilter, ilustrado na Figura 5.12, observa-se um padrão de aumento de eficiência. Conforme ocorre o aumento do valor do *ThresholdLow*, causa-se um maior ganho pelo pré-processamento. No entanto, existem algumas exceções para o caso de bases que já tinham baixo valor de *F-Score*, como os casos das bases *Haberman* e *SPECTF-Heart*, ou as que já estavam com alto valor de *F-Score*, como a base *Libra*.

Para identificar se existe alguma explicação para tal comportamento e para a diferença de desempenho dos métodos entre as bases, decidiu-se analisar as bases usando as medidas de complexidade de dados apresentadas na Seção 2.6.

Os dados de complexidades calculados para as bases artificiais estão resumidos na Tabela 5.8, assim como na Figura 5.13. Decidiu-se, novamente, focar a análise nas bases artificiais com proporção da classe positiva de 10%, 25%, e 50% para alguns valores-chave de diferença da média.

Os valores calculados indicam que F1, para o caso da base artificial gerada com distribuição gaussiana, é uma boa representação para a sobreposição de dados. Como F1 (Equação 2.7) é uma medida direta de diferença entre as médias, o mesmo apresenta um relacionamento claro entre as bases artificiais e a medida. É possível verificar também que F1 não é impactado pelo desbalanceamento, já que mede apenas as médias, a exceção ocorrendo em casos em que existem poucos exemplares da classe positiva, o que pode fazer com que o centro da média da base artificial seja deslocado.

Os valores de F3, N2 e D3 demonstram também o efeito da sobreposição. No entanto, ocorre também o impacto do desbalanceamento nas medidas quando as classes têm alguma sobreposição. É interessante notar que, com o aumento do desbalanceamento, os valores indicam que o classificador teria uma dificuldade menor para sua separação. Esse efeito é inesperado pois, a princípio, seria esperado que as medidas indicassem uma maior dificuldade do processo de classificação com o aumento do desbalanceamento e não o contrário.

No entanto, em uma análise mais detalhada, nota-se que em nenhuma dessas medidas

Tabela 5.8: Medidas de complexidade para as bases artificiais para 3 níveis de proporção da classe positiva (50%, 25%, 10%)

Diferença da média (sobreposição de dados)	**% Classe positiva (desbalanceamento de dados)**	**F1**	**N2**	**F3**	**D3**
0,0	50%	0,0016	0,9696	0,0030	0,4830
0,5	50%	0,1225	0,9604	0,0040	0,4600
1,0	50%	0,5231	0,5835	0,0120	0,2750
1,5	50%	1,0130	0,3121	0,0240	0,1550
2,0	50%	1,8996	0,1886	0,0840	0,1010
2,5	50%	3,4796	0,1155	0,2890	0,0370
3,0	50%	4,5360	0,0712	0,4340	0,0230
3,5	50%	6,0856	0,0521	0,6190	0,0110
4,0	50%	8,1638	0,0413	0,7490	0,0040
4,5	50%	10,3897	0,0303	0,8830	0,0010
5,0	50%	12,4878	0,0261	0,9700	0,0000
0,0	25%	0,0001	0,7195	0,0210	0,3208
0,5	25%	0,1864	0,7273	0,0090	0,2804
1,0	25%	0,5928	0,4429	0,0090	0,1979
1,5	25%	0,9760	0,2846	0,0180	0,1544
2,0	25%	1,7201	0,2099	0,0795	0,0705
2,5	25%	3,0953	0,1248	0,3193	0,0405
3,0	25%	4,2838	0,0760	0,4558	0,0210
3,5	25%	6,6338	0,0511	0,6702	0,0060
4,0	25%	8,5327	0,0387	0,8936	0,0015
4,5	25%	10,0142	0,0326	0,8726	0,0000
5,0	25%	14,0886	0,0270	0,9715	0,0000
0,0	10%	0,0049	0,3968	0,0665	0,1133
0,5	10%	0,1239	0,4246	0,0252	0,1115
1,0	10%	0,6125	0,3171	0,0162	0,0953
1,5	10%	0,9676	0,1589	0,3183	0,1007
2,0	10%	1,8321	0,1474	0,0701	0,0486
2,5	10%	3,1505	0,1167	0,6403	0,0216
3,0	10%	5,0228	0,0716	0,4730	0,0072
3,5	10%	6,1985	0,0454	0,7518	0,0090
4,0	10%	8,9166	0,0334	0,9604	0,0000
4,5	10%	11,2502	0,0285	0,9676	0,0000
5,0	10%	14,7532	0,0257	0,9658	0,0000

Fonte: Elaborada pelo autor.

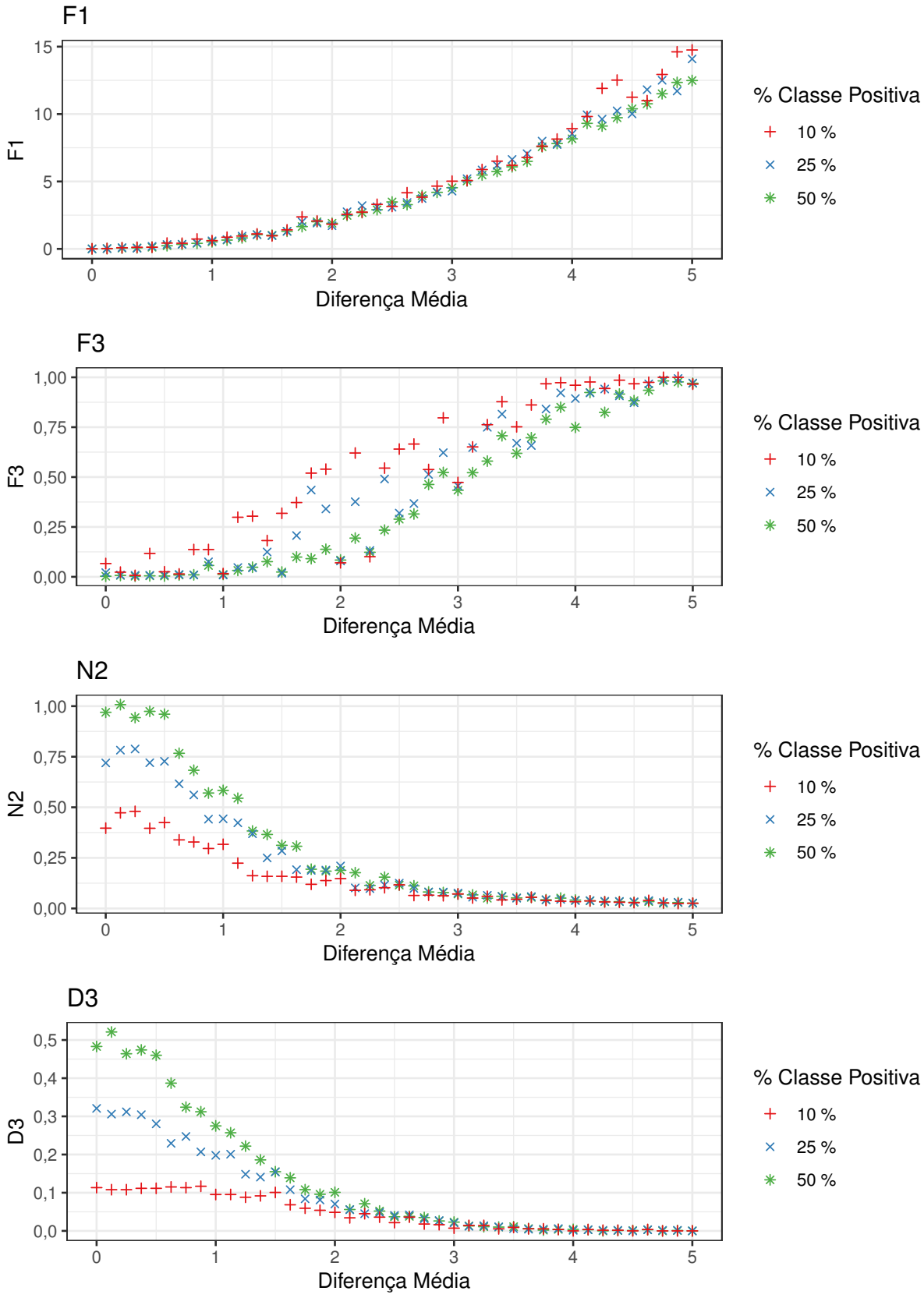


Figura 5.13: Resultado dos cálculos das medidas de complexidade para as bases artificiais. O desbalanceamento aparece distinguido por cor e forma. Fonte: Elaborada pelo autor.

existe um peso para considerar o desbalanceamento e colocar um maior peso na classe positiva. Assim, um maior desbalanceamento entre classes faz com que de fato seja mais fácil para um classificador, como o kNN, classificar a classe negativa, que está mais presente. Dessa maneira, o classificador aumenta a acurácia global em detrimento da perda de acurácia da classe positiva. Logo, essas medidas demonstram apenas o ponto de vista da melhora para o processo de classificação da base como um todo.

Por causa dessa característica, verificou-se a possibilidade de uma medida que possa demonstrar um maior enfoque na classe alvo que está desbalanceada na base. O trabalho inicial de Sánchez, Mollineda e Sotoca (2007) indica que D3 pode ser calculado segregado por classe, sendo assim, verificou-se a possibilidade de usar D3 considerando apenas a classe positiva. Essa medida, denotada como $D3_{Pos}$, é a proporção de exemplares da classe positiva que está em uma área de sobreposição.

O resultado dessa medida para as bases artificias se encontra na Tabela 5.9, em que também se incluiu D3 para comparação, e na Figura 5.14. Nota-se que, nesse caso, os valores demonstraram que, conforme esperado, o aumento do desbalanceamento fez com que a classe positiva, em menor número, tivesse uma perda na capacidade de acurácia em bases com alta sobreposição, indicadas pelo valor superior de $D3_{Pos}$. Assim, será adicionada essa medida para as futuras análises a serem realizadas nesse trabalho.

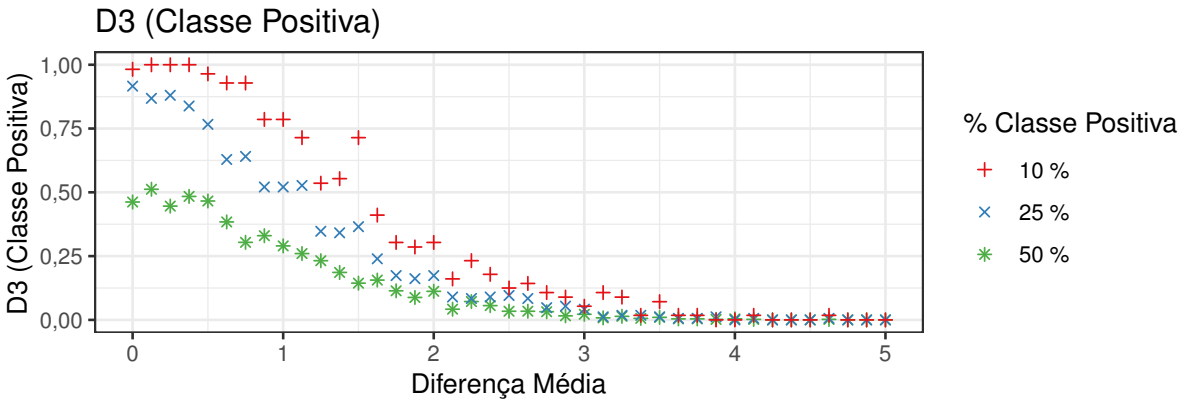


Figura 5.14: Resultado do cálculo de D3 considerando apenas a classe positiva para as bases artificiais. Essa medida, que será referenciada como $D3_{Pos}$, no trabalho traz a fração de exemplares da classe positiva na área de desbalanceamento. Fonte: Elaborada pelo autor.

Calcularam-se também as medidas complexas para as bases reais utilizadas nos experimentos. Os resultados obtidos podem ser verificados na Tabela 5.10, em conjunto com

Tabela 5.9: Medidas $D3_{Pos}$ para as bases artificiais para 3 níveis de proporção da classe positiva (50%, 25%, 10%)

Diferença da média (sobreposição de dados)	**% Classe positiva (desbalanceamento de dados)**	**D3**	$\boldsymbol{D3_{Pos}}$
0,0	50%	0,4830	0,4620
0,5	50%	0,4600	0,4660
1,0	50%	0,2750	0,2900
1,5	50%	0,1550	0,1440
2,0	50%	0,1010	0,1120
2,5	50%	0,0370	0,0340
3,0	50%	0,0230	0,0220
3,5	50%	0,0110	0,0100
4,0	50%	0,0040	0,0040
4,5	50%	0,0010	0,0000
5,0	50%	0,0000	0,0000
0,0	25%	0,3208	0,9162
0,5	25%	0,2804	0,7665
1,0	25%	0,1979	0,5210
1,5	25%	0,1544	0,3653
2,0	25%	0,0705	0,1737
2,5	25%	0,0405	0,0958
3,0	25%	0,0210	0,0419
3,5	25%	0,0060	0,0120
4,0	25%	0,0015	0,0000
4,5	25%	0,0000	0,0000
5,0	25%	0,0000	0,0000
0,0	10%	0,1133	0,9821
0,5	10%	0,1115	0,9643
1,0	10%	0,0953	0,7857
1,5	10%	0,1007	0,7143
2,0	10%	0,0486	0,3036
2,5	10%	0,0216	0,1250
3,0	10%	0,0072	0,0536
3,5	10%	0,0090	0,0714
4,0	10%	0,0000	0,0000
4,5	10%	0,0000	0,0000
5,0	10%	0,0000	0,0000

Fonte: Elaborada pelo autor.

a proporção da classe positiva para representar o desbalanceamento.

Tabela 5.10: Medidas de complexidade para as bases reais

	Base de dados	**F1**	**F3**	**N2**	**D3**	$D3_{Pos}$	**% Classe Positiva**
1	Ecoli	1,8042	0,2143	0,4247	0,0417	0,1154	15%
2	Glass	0,9531	0,2196	0,2392	0,0374	0,5556	4%
3	Haberman	0,1832	0,0294	0,7948	0,2876	0,7284	26%
4	Heart	0,7422	0,0132	0,9080	0,3399	0,4029	4%
5	Hepatitis	0,7075	0,1742	0,8789	0,2387	0,9375	21%
6	Iris	0,6802	0,5600	0,2010	0,0333	0,0600	33%
7	Libra	0,1102	0,0583	0,3102	0,0139	0,0694	20%
8	Mamographic	0,9175	0,0010	0,4624	0,2029	0,2135	46%
9	Pima	0,5743	0,0052	0,8277	0,2852	0,4776	35%
10	SPECTF-Heart	0,5443	0,0746	0,7994	0,2687	0,5893	21%
11	Wine	2,3331	0,3258	0,1692	0,2303	0,3380	40%
12	Wiscosin	3,4635	0,1187	0,3334	0,0315	0,0373	34%

Fonte: Elaborada pelo autor.

Para o caso do SOMEntropyLowFilter, verificando os dados para base de *Libra* e *Haberman*, na Tabela 5.10, nota-se que ambas as bases possuem os menores valores de F1 (menores que 0,2), o que indica que ambas as bases têm um nível de sobreposição alto quando comparado ao das demais bases analisadas. Quando comparado às bases artificiais, tal valor foi encontrado apenas quando a diferença da média entre as classes foi menor que 0,5, conforme pode ser verificado na Tabela 5.10. Esses casos são justamente os que, no caso do SOMEntropyLowFilter, tiveram padrão de perda de *F-Score* com o aumento do *ThresholdLow*. Tal comportamento parece indicar que, em caso severo de sobreposição de dados, esse método pode ter seu processo alterado, refletindo o mesmo comportamento verificado inicialmente nas bases artificiais, quando se analisou a variação do parâmetro.

Em princípio, pelas medidas encontradas, não foi possível identificar em quais características o SOMEntropyLowFilter tem um desempenho maior do que o SOMEntropyHighFilter. Nas bases em que isso ocorreu (*Glass*, *Hepatitis* e *Pima*), verificaram-se valores altos de sobreposição, com $D3_{Pos}$ maior que 0,44. No entanto, outras bases com alta sobreposição da classe positiva, como *Haberman*, *SPECTF-Heart* e as bases artificiais com alta sobreposição, não demonstraram esse comportamento, o que parece confirmar que, além da sobreposição e do desbalanceamento, existem outras características que impactam o processo dos métodos.

Para continuar com a análise, deve-se avaliar o ganho que foi obtido no *F-Score*, de

acordo com as medidas de complexidade calculadas.

Em primeiro momento, a análise será feita em cima do F1, que tem uma boa representação para a sobreposição (CANO, 2013; MORÁN-FERNÁNDEZ; BOLÓN-CANEDO; ALONSO-BETANZOS, 2017).

Pelos dados de ganho, disponibilizados na Figura 5.15, é possível notar que os ganhos dos métodos não podem ser explicados utilizando-se unicamente o valor de F1. No entanto, a análise traz algumas conclusões interessantes. O método funcionou para bases reais e artificiais, com maiores ganhos na faixa de baixo valor de F1; com ganhos aumentando até chegarem a um certo valor de F1 e caindo a partir desse valor. O valor em que ocorre a alteração é diferente para cada nível de desbalanceamento.

No caso das bases artificiais níveis maiores de desbalanceamento fazem com que os ganhos sejam maiores e exista uma faixa de F1 maior em que os métodos conseguem trazer benefícios.

Pode-se notar que, nas bases reais, os ganhos seguem um padrão semelhante de ganhos maiores em faixas intermediárias de sobreposição e desbalanceamento. Especialmente no caso do SOMEntropyLowFilter, os ganhos maiores são de bases com desbalanceamento mais severos, demonstrando que, em princípio, o método funciona bem para o problema de desbalanceamento.

No entanto, isso não é verdadeiro para todas as bases com alta sobreposição e desbalanceadas. Por exemplo, no caso das bases *Libra* e *Iris*, que são casos em que o kNN já tem bom desempenho, e no caso da base *Haberman*, que tem a maior sobreposição. Denotando, novamente, que outros fatores impactam a performance do algoritmo.

No caso dos ganhos em F3, que podem ser vistos na Figura 5.16, temos uma situação semelhante a F1, em que temos uma faixa na qual ocorrem os maiores ganhos nas áreas de maior sobreposição e na qual os ganhos passam a diminuir conforme vamos para áreas de menor sobreposição. Diferente de F1, no entanto, temos uma distribuição mais uniforme dos ganhos com a variação do valor de F3, e apenas valores mais extremos apresentando a situação em que não temos nenhum ganho dos métodos.

No caso das medidas N2 e D3, observadas nas Figuras 5.17 e 5.18, respectivamente, os ganhos não mostraram, em princípio, uma tendência clara, como as medidas F1 e F3. As bases demonstram ganhos em diferentes valores das medidas, mesmo em pontos de baixa ou alta dificuldade para o classificador. No entanto, conforme verificamos em análise

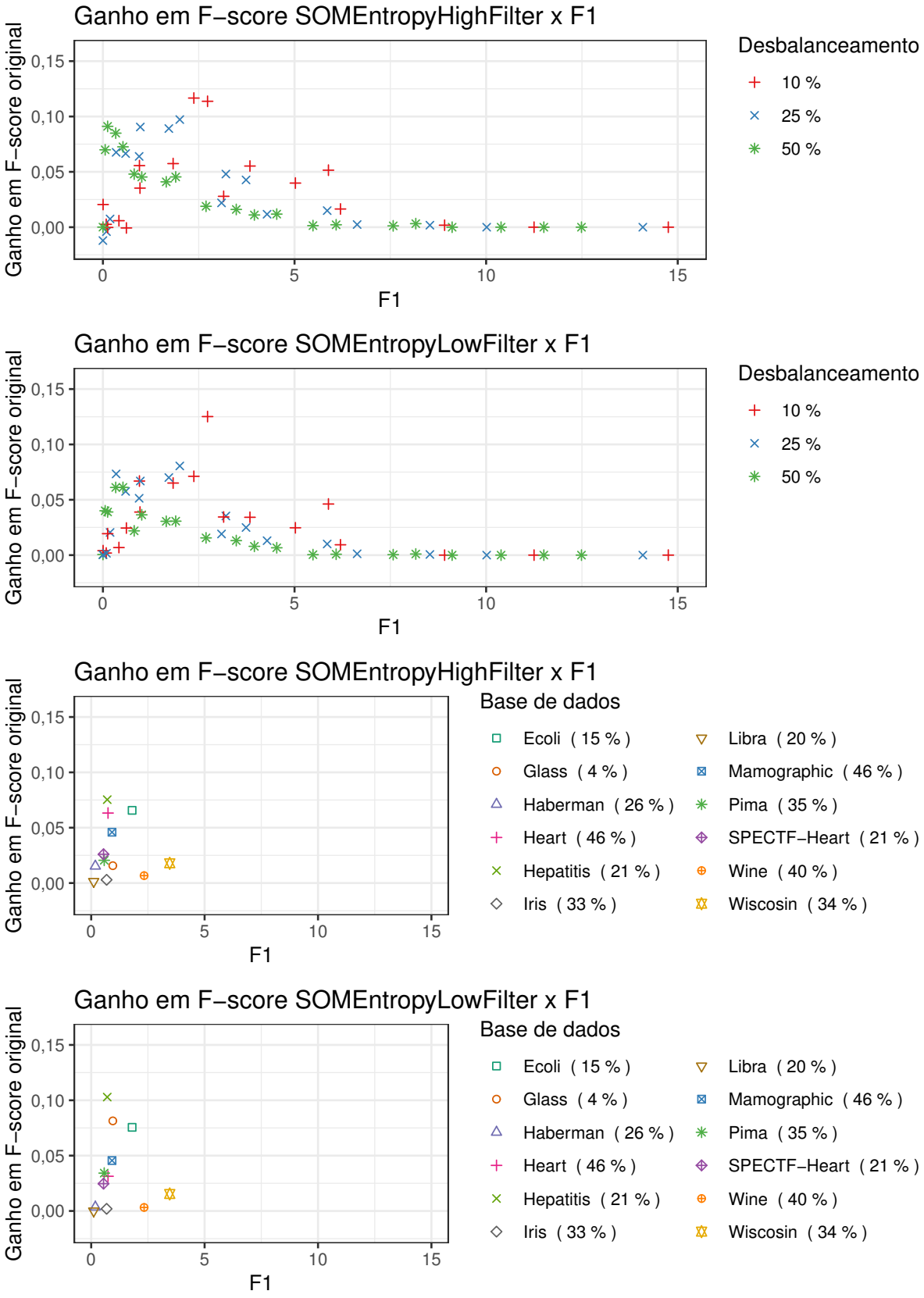


Figura 5.15: Ganho de *F-Score* por F1, nas bases está relacionado à taxa de desbalanceamento. Fonte: Elaborada pelo autor.

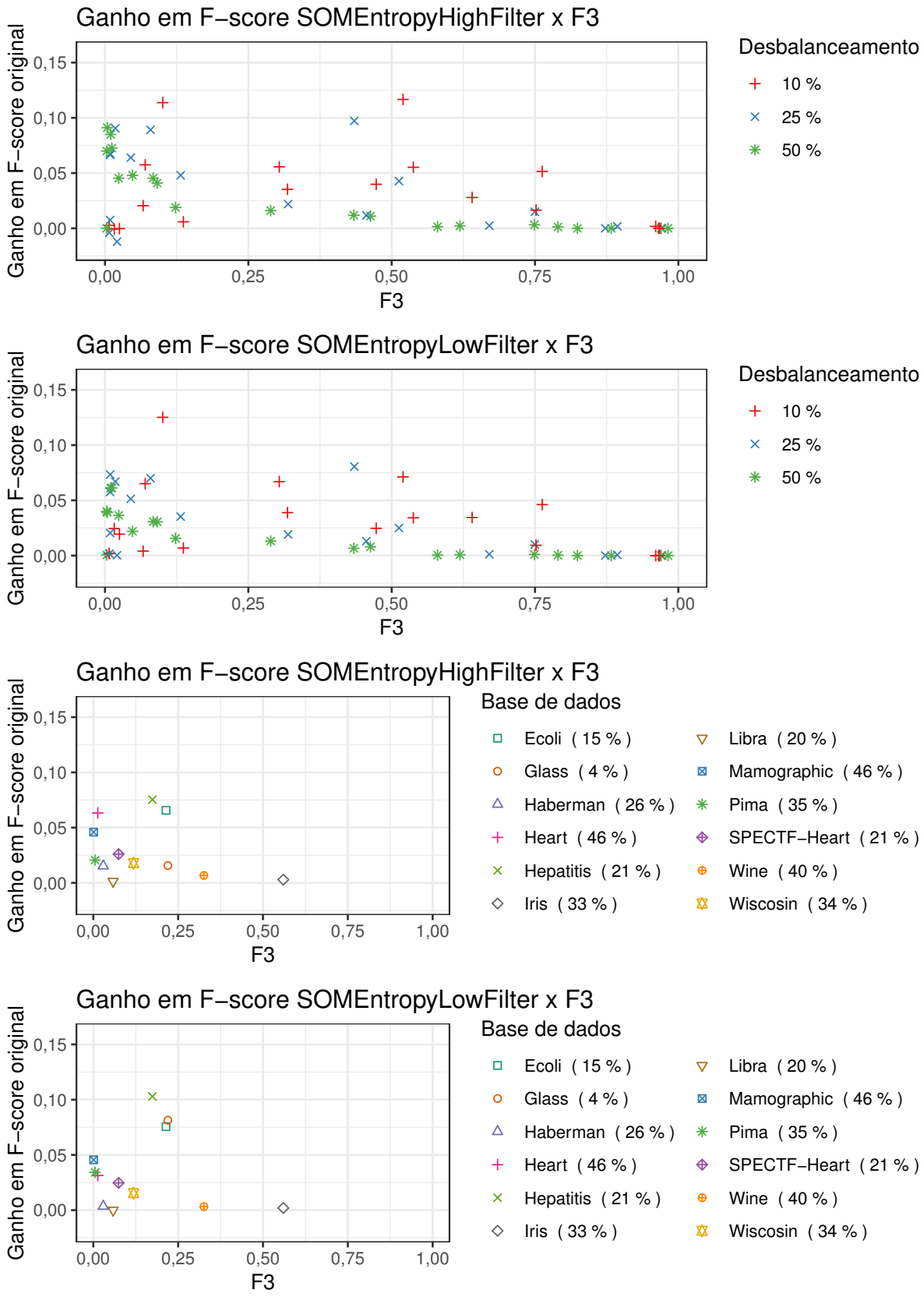


Figura 5.16: Ganho de *F-Score* por F3, nas bases está relacionado à taxa de desbalanceamento. Fonte: Elaborada pelo autor.

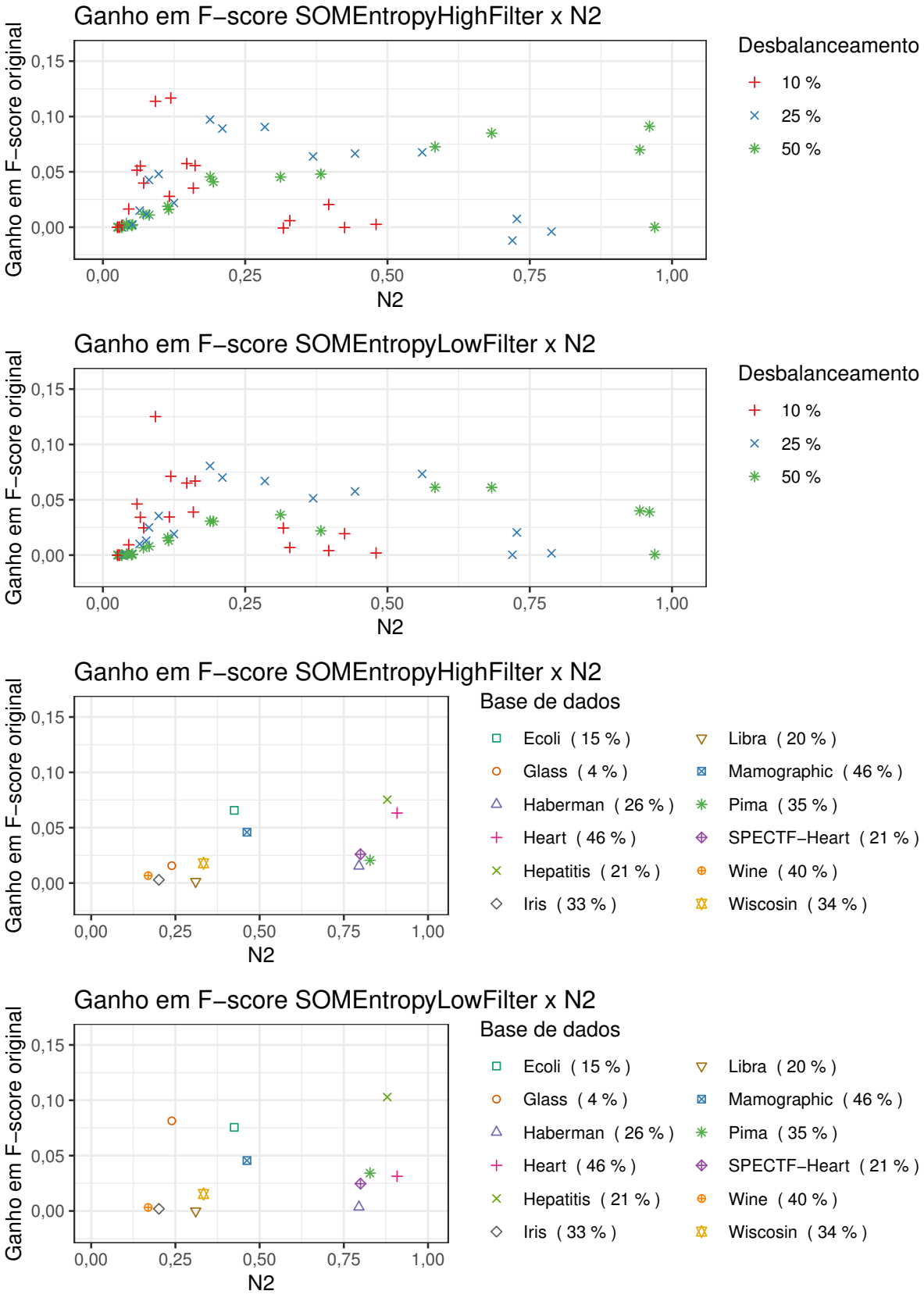


Figura 5.17: Ganho de *F-Score* por N2, nas bases está relacionado à taxa de desbalanceamento. Fonte: Elaborada pelo autor.

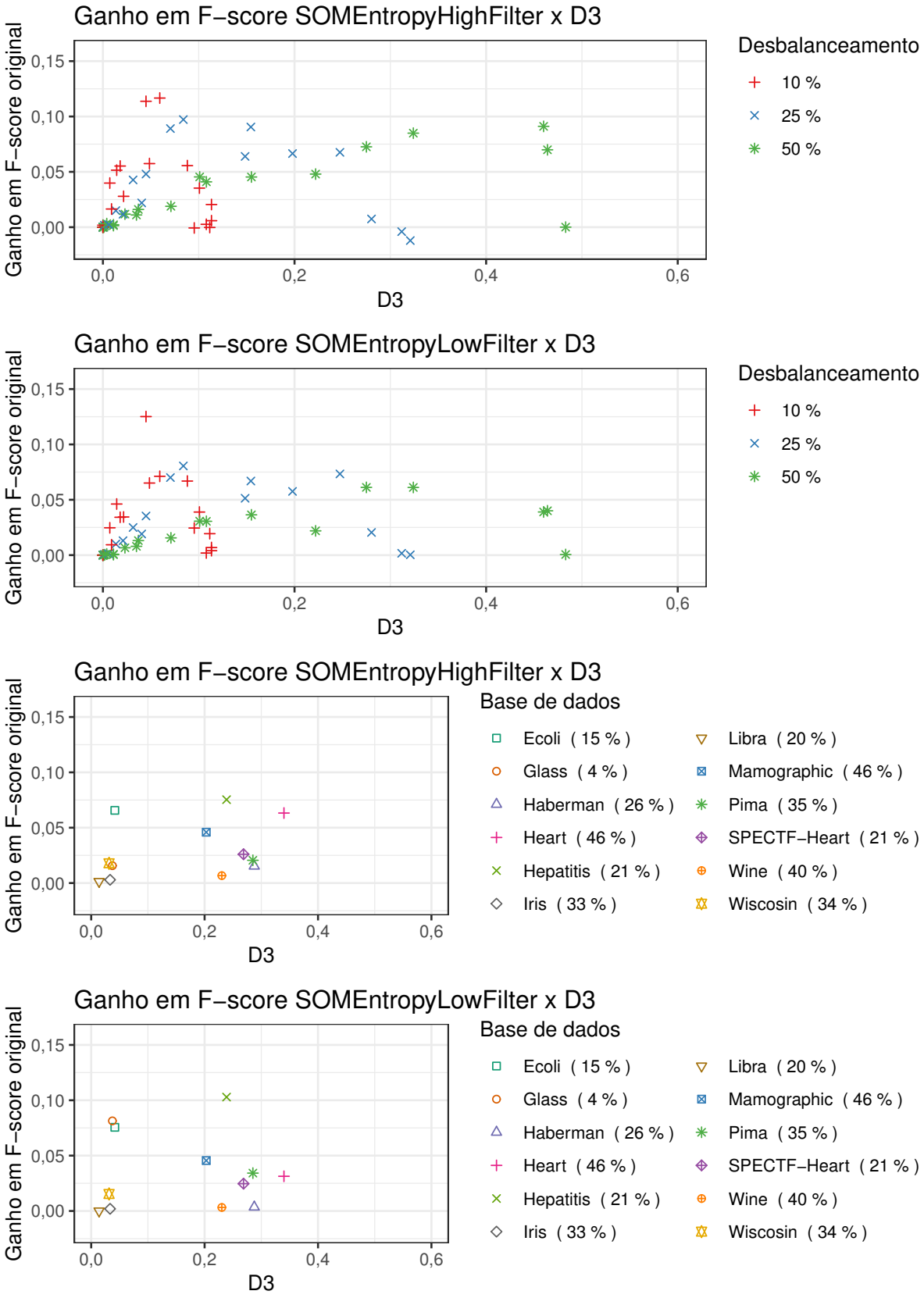


Figura 5.18: Ganho de *F-Score* por D3, nas bases está relacionado à taxa de desbalanceamento. Fonte: Elaborada pelo autor.

anterior, essas medidas apresentam a dificuldade do classificador como um todo e não consideram o desbalanceamento da classe positiva, apresentando melhores valores para classes desbalanceadas e, assim não demonstrando os pontos em que estamos trabalhando nos métodos.

No caso de $D3_{Pos}$, o ganho por *F-Score* demonstrado na Figura 5.19 teve comportamentos diferentes para base real e artificial. No caso das bases artificiais, os ganhos estão presentes em um formato de parábola, com os maiores ganhos nos pontos intermediários de sobreposição da classe positiva. Dessa maneira, $D3_{Pos}$ traduz os efeitos de ganho em faixas intermediárias de sobreposição e desbalanceamento verificadas em testes anteriores.

Nas bases reais, no entanto, esse efeito não está claro. Casos que tiveram ganhos distintos, como as bases *Hepatitis* e *Wine*, tiveram o mesmo valor de $D3_{Pos}$. Logo, essa medida não pode, individualmente, explicar os ganhos.

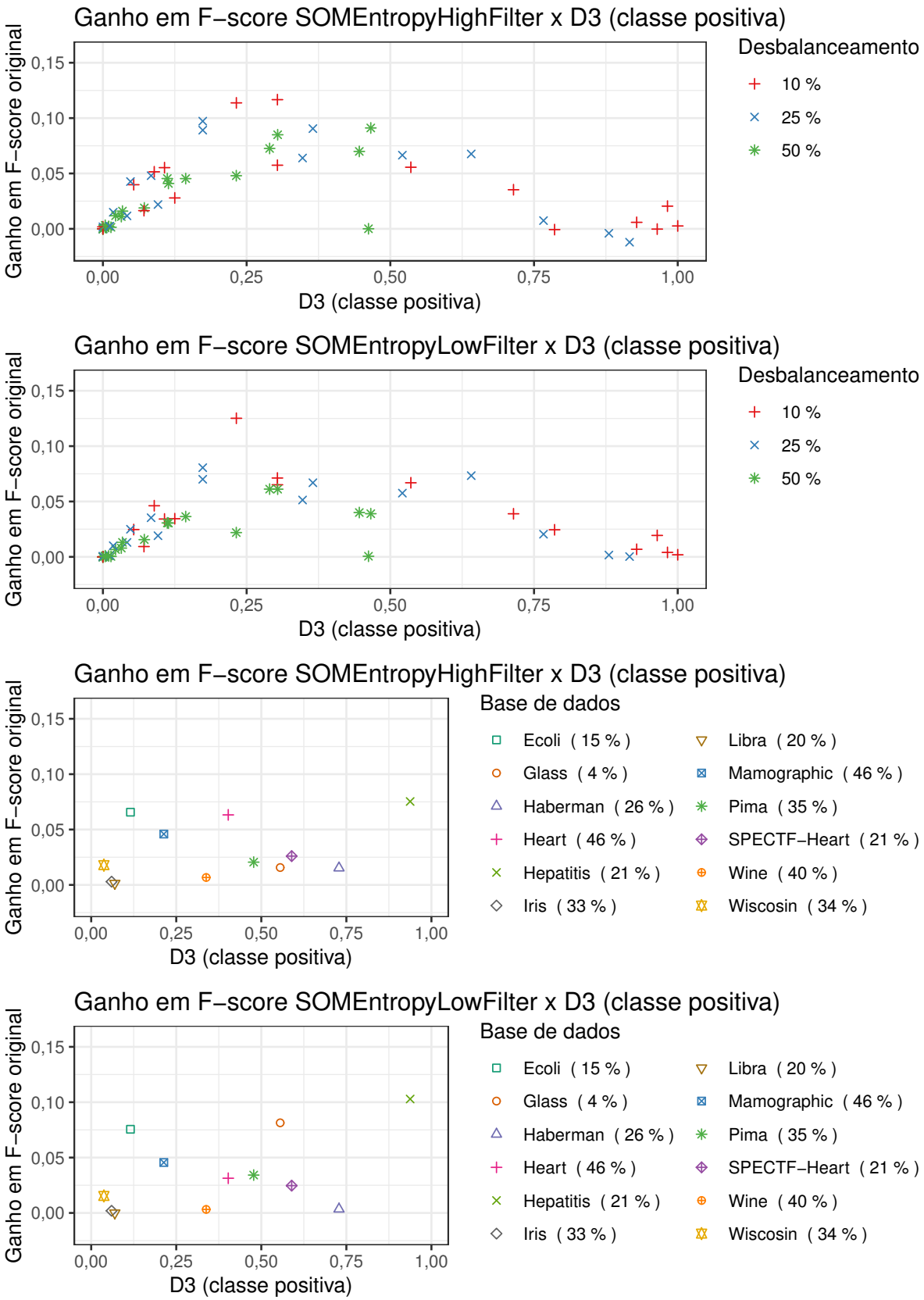


Figura 5.19: Ganho de *F-Score* por $D3_{Pos}$, nas bases está relacionado à taxa de desbalanceamento. Fonte: Elaborada pelo autor.

Capítulo 6

Conclusões e Trabalhos Futuros

Bases de dados têm diferentes características que podem influenciar o desempenho de uma tarefa de classificação de dados. Entre elas, duas têm efeitos negativos no desempenho de algoritmos de classificação: a sobreposição de dados e o desbalanceamento.

Diversas técnicas para aumentar o desempenho dos classificadores nessas condições são encontradas na literatura. Uma dessas técnicas é o pré-processamento de dados, que realiza a transformação da base de dados com o objetivo de obter um desempenho superior do classificador. As técnicas de pré-processamento podem envolver a inclusão ou remoção de exemplares. Para a remoção de exemplares, uma das possibilidades é a seleção de protótipos, que atua para selecionar, entre os exemplares da base, os exemplares mais adequados para uma tarefa de classificação.

Esse trabalho introduziu um algoritmo SOMEntropyFilter com três métodos de seleção de protótipos, nomeados SOMEntropyHighFilter, SOMEntropyLowFilter e SOMEntropyHighLowFilter, que utilizam mapas-auto-organizáveis e entropia da informação como o filtro para a seleção dos exemplares. Os métodos têm o objetivo de aumentar o desempenho de um classificador em bases de dados que sofram de sobreposição ou desbalanceamento de dados.

Resultados em bases artificiais e reais demonstraram que os métodos introduzidos melhoraram os valores de acurácia, *F-Score* e *G-Mean* de um classificador 1NN em bases com problemas de desbalanceamento ou sobreposição.

Analisou-se a taxa de redução obtida com o classificador como objetivo do processo e como complemento em casos de otimização pelo melhor desempenho. Para os métodos SOMEntropyHighFilter e SOMEntropyHighLowFilter, obtiveram-se taxas de redução

com valores superiores a 50%. Tal comportamento pode indicar que outro trabalho possa ser realizado para verificar, de forma detalhada, o comportamento das taxas de redução e validar o impacto na performance do kNN em questão de tempo de classificação e espaço de armazenamento.

Os métodos necessitam da adição dos parâmetros *ThresholdHigh*, *ThresholdLow* e a configuração do tamanho do mapa SOM. A adição desses novos parâmetros traz a desvantagem de aumentar o número de parâmetros para configuração do classificador. Para buscar uma simplificação, analisou-se o comportamento do SOMEntropyHighFilter e SOMEntropyLowFilter com relação à variação dos parâmetros de limiar, assim como uma análise do melhor mapa. Medidas de complexidade também foram utilizadas para analisar as bases utilizadas e buscar um padrão de comportamento.

Para a definição do tamanho do mapa SOM, analisaram-se diferentes tamanhos de um mapa hexagonal. Alterando o valor da constante C_{Mapa}, identificou-se que valores menores da constante foram os mais eficientes na média, demonstrando que valores menores do mapa de SOM podem ser os mais indicados. Trabalhos futuros podem se concentrar nessa região para verificar a existência de um tamanho de mapa ideal.

No caso da variação do limiar, conforme pode ser verificado nas figuras 5.11 e 5.12, identificou-se que o comportamento do SOMEntropyLowFilter teve um padrão de aumento de eficiência conforme se aumenta o valor do *ThresholdLow*, com algumas exceções, o que permite determinar que existe uma faixa de valor ideal para esse parâmetro. O uso de medidas de complexidade permitiu analisar esse comportamento e demonstrou que o comportamento diferente para o SOMEntropyLowFilter pode ser relacionado a casos em que temos alta sobreposição, indicado pelo baixo valor de F1. Já no caso do SOMEntropyHighFilter, no entanto, não se demonstrou um padrão claro com a alteração do *ThresholdHigh*.

As medidas de complexidade permitiram, pela análise de valores de F1, F3 e $D3_{Pos}$, identificar que os métodos possuem boa performance em bases com sobreposição de dados. Também foi constatado que existem outros fatores que não foram possíveis de serem observados nesse momento para diferença de ganho entre as bases de dados.

Para continuação desse trabalho, espera-se uma maior análise dos dados gerados para identificar pontos de melhoria e uma maior compreensão de como se comportam os métodos propostos e os seus benefícios. Para essa análise, é interessante incluir a validação em

novos cenários, como bases com ruído e com diferentes distribuições e dimensões. Também, um estudo com outros métodos de edição pode ser feito para comparar a performance do método com outros processos da literatura.

O método desenvolvido é baseado, principalmente, na solução do tratamento da sobreposição de classes e não é especializado para o tratamento de desbalanceamento. Uma opção para melhorar o processo seria se investigar a combinação de um método de pré-processamento para balancear a proporção das classes a priori e, em sequência, executar o SOMEntropyFilter. Dessa maneira, com a base balanceada, o método poderia ter uma melhor performance para bases desbalanceadas e com sobreposição do que utilizando-se apenas um dos métodos de pré-processamento.

Esse trabalhou considerou o problema de duas classes, no entanto, diversos problemas reais são multi-classe. Como os mapas-auto-organizáveis permitem a classificação multi-classe, em um trabalho futuro seria possível estudar a expansão dos métodos para o tratamento do problema multi-classe. Para isso, será necessário investigar a adaptação do filtro pela entropia para esse tipo de problema.

Referências

ALFONS, Andreas. *cvTools: Cross-validation tools for regression models*. Rotterdam, Holanda, 2012. R package version 0.3.2. Disponível em: <https://CRAN.R-project.org/package=cvTools>.

ARABMAKKI, Elaheh; KANTARDZIC, Mehmed. SOM-based partial labeling of imbalanced data stream. *Neurocomputing*, v. 262, p. 120–133, 11 2017. ISSN 09252312. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0925231217309839>.

BASU, Mitra; HO, Tin Kam. *Data Complexity in Pattern Recognition*. Londres, Reino Unido: Springer London, 2006. 297 p. ISBN 978-1-84628-171-6. Disponível em: <http://link.springer.com/10.1007/978-1-84628-172-3>.

BRANCO, Paula; TORGO, Luís; RIBEIRO, Rita P. A Survey of Predictive Modeling on Imbalanced Domains. *ACM Computing Surveys*, v. 49, n. 2, p. 1–50, 8 2016. ISSN 03600300. Disponível em: <http://dl.acm.org/citation.cfm?doid=2966278.2907070>.

CANO, Jose Ramon. Analysis of data complexity measures for classification. *Expert Systems with Applications*, v. 40, n. 12, p. 4820–4831, 2013. ISSN 09574174.

CASTRO, Leandro Nunes de; FERRARI, Daniel Gomes. *Introdução à mineração de dados*. 1ª. ed. São Paulo: Editora Saraiva, 2016. 352 p. ISBN 978-85-472-0098-5.

COVER, T. M.; HART, P. E. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, v. 13, n. 1, p. 21–27, 1 1967. ISSN 15579654. Disponível em: <http://ieeexplore.ieee.org/document/1053964/>.

DEMŠAR, Janez. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, JMLR.org, v. 7, p. 1–30, dez. 2006. ISSN 1532-4435. Disponível em: <http://dl.acm.org/citation.cfm?id=1248547.1248548>.

DENIL, Misha; TRAPPENBERG, Thomas. Overlap versus imbalance. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer, v. 6085 LNAI, p. 220–231, 2010. ISSN 03029743.

DHEERU, Dua; TANISKIDOU, Efi Karra. *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2017. Disponível em: <http://archive.ics.uci.edu/ml>.

DOUZAS, Georgios; BACAO, Fernando. Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning. *Expert Systems with Applications*, v. 82, p. 40–52, 2017. ISSN 09574174.

FACELI, Katti et al. *Inteligência artificial: uma abordagem de aprendizado de máquina*. 1ª. ed. Rio de Janeiro: LTC, 2011. 378 p. ISBN 9788521618805.

FOX, John; WEISBERG, Sanford. *An R Companion to Applied Regression*. Second. Thousand Oaks, Estados Unidos: Sage, 2011. Disponível em: <http://socserv.socsci.mcmaster.ca/jfox/Books/Companion>.

GARCIA, Luís P.F.; CARVALHO, André C.P.L.F. de; LORENA, Ana C. Effect of label noise in the complexity of classification problems. *Neurocomputing*, v. 160, p. 108–119, 2015. ISSN 18728286. Disponível em: <http://www.scopus.com/inward/record.url?eid=2-s2.0-84927970970&partnerID=40&md5=bc38fd0172c3091ed60ee512638f56cc>.

GARCÍA, Salvador et al. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 34, n. 3, p. 417–435, 3 2012. ISSN 01628828. Disponível em: <http://ieeexplore.ieee.org/document/6136515/>.

GARCIA, Vicente; SANCHEZ, Jose; MOLLINEDA, Ramon. An empirical study of the behavior of classifiers on imbalanced and overlapped data sets. *Progress in Pattern Recognition, Image Analysis and Applications, Proceedings*, v. 4756, p. 397–406, 2007. ISSN 0302-9743.

HE, Haibo; GARCIA, Edwardo A. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, v. 21, n. 9, p. 1263–1284, 9 2009. ISSN 1041-4347. Disponível em: <http://ieeexplore.ieee.org/document/5128907/>.

HO, Tin Kam; BASU, Mitra. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 3, p. 289–300, 2002. ISSN 01628828.

KIM, Sang-Woon; OOMMEN, B John. A brief taxonomy and ranking of creative prototype reduction schemes. *Pattern Analysis & Applications*, v. 6, n. 3, p. 232–244, 12 2003. ISSN 1433-7541. Disponível em: <http://link.springer.com/10.1007/s10044-003-0191-0>.

KOHONEN, Teuvo. Essentials of the self-organizing map. *Neural Networks*, Elsevier Ltd, v. 37, p. 52–65, 2013. ISSN 08936080. Disponível em: <http://dx.doi.org/10.1016/j.neunet.2012.09.018>.

KONONENKO, Igor; KUKAR, Matjaz. *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. 1ª. ed. Chichester, Reino Unido: Horwood Publishing Limited, 2007. 445 p. ISBN 1904275214.

LAROSE, Daniel T.; LAROSE, Chantal D. *Data Mining and Predictive Analytics*. 2ª. ed. Hoboken, Estados Unidos: John Wiley & Sons, Inc., 2015. 780 p. ISBN 9780128002292.

LÓPEZ, Victoria et al. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, Elsevier Inc., v. 250, p. 113–141, 2013. ISSN 00200255. Disponível em: <http://dx.doi.org/10.1016/j.ins.2013.07.007>.

LORENA, Ana C. et al. How Complex is your classification problem? A survey on measuring classification complexity. *CoRR*, v. 1808.03591, n. August, 8 2018. Disponível em: <http://arxiv.org/abs/1808.03591>.

MORÁN-FERNÁNDEZ, L.; BOLÓN-CANEDO, V.; ALONSO-BETANZOS, A. Can classification performance be predicted by complexity measures? A study using microarray data. *Knowledge and Information Systems*, v. 51, n. 3, p. 1067–1090, 2017. ISSN 02193116.

MOREIRA, Leandro Juvêncio; SILVA, Leandro A. Prototype Generation Using Self-Organizing Maps for Informativeness-Based Classifier. *Computational Intelligence and Neuroscience*, v. 2017, p. 1–15, 2017. ISSN 1687-5265. Disponível em: <https://www.hindawi.com/journals/cin/2017/4263064/>.

MORETTIN, Pedro A.; BUSSAB, Wilton de O. *Estatística Básica*. 6ª. ed. São Paulo: Saraiva, 2010. 557 p. ISBN 9788502081772.

POHLERT, Thorsten. *The Pairwise Multiple Comparison of Mean Ranks Package (PMCMR)*. Koblenz, Alemanha, 2014. R package. Disponível em: <https://CRAN.R-project.org/package=PMCMR>.

PRATI, Ronaldo C.; BATISTA, Gustavo E. A. P. A.; MONARD, Maria Carolina. Class Imbalances versus Class Overlapping: An Analysis of a Learning System Behavior. *MICAI 2004: Advances in Artificial Intelligence*, Cidade do México, México, v. 2972, p. 312–321, 2004. ISSN 03029743. Disponível em: <http://link.springer.com/10.1007/978-3-540-24694-7_32>.

R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2018. Disponível em: <https://www.R-project.org/>.

SÁNCHEZ, J. S.; MOLLINEDA, R. A.; SOTOCA, J. M. An analysis of how training data complexity affects the nearest neighbor classifiers. *Pattern Analysis and Applications*, v. 10, n. 3, p. 189–201, 2007. ISSN 14337541.

SHANNON, C E. The mathematical theory of communication. 1963. *M.D. computing : computers in medical practice*, v. 14, n. 4, p. 306–17, 1948. ISSN 0724-6811. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/9230594>.

SILVA, Leandro Augusto da; SARAJANE, Marques Peres; BOSCARIOLI, Clodis. *Introdução à Mineração de Dados: Com Aplicações em R*. 1ª. ed. Rio de Janeiro: Elsevier, 2017. 296 p. ISBN 978-8535284461.

TRIGUERO, Isaac et al. A taxonomy and experimental study on prototype generation for nearest neighbor classification. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, v. 42, n. 1, p. 86–100, 2012. ISSN 10946977.

VENABLES, W. N.; RIPLEY, B. D. *Modern Applied Statistics with S*. Fourth. Nova York, Estados Unidos: Springer, 2002. ISBN 0-387-95457-0. Disponível em: <http://www.stats.ox.ac.uk/pub/MASS4>.

WEHRENS, R.; BUYDENS, L.M.C. Self- and super-organising maps in r: the kohonen package. *J. Stat. Softw.*, v. 21, n. 5, 2007. Disponível em: <http://www.jstatsoft.org/v21/i05>.

WICKHAM, Hadley et al. *dplyr: A Grammar of Data Manipulation*. Stanford, Estados Unidos, 2018. R package version 0.7.6. Disponível em: <https://CRAN.R-project.org/package=dplyr>.

WING, Max Kuhn. Contributions from Jed et al. *caret: Classification and Regression Training*. New London, Estados Unidos, 2018. R package version 6.0-80. Disponível em: <https://CRAN.R-project.org/package=caret>.

WUERTZ, Diethelm; SETZ, Tobias; CHALABI, Yohan. *fBasics: Rmetrics - Markets and Basic Statistics*. Zurique, Suiça, 2017. R package version 3042.89. Disponível em: <https://CRAN.R-project.org/package=fBasics>.

ZAR, Jerrold H. *Biostatistical Analysis*. 5ª. ed. Harlow, Reino Unido: Pearson Education Limited, 2014. 756 p. ISBN 978-1-292-02404-2.