Um algoritmo de indução de árvore de decisão baseado em agrupamento

Rodrigo C. Barros[1], **Márcio P. Basgalupp**[2], **André C.P.L.F. de Carvalho**[1], **Marcos G. Quiles**[2]

[1]Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo (USP)

[2]Instituto de Ciência e Tecnologia - Universidade Federal de São Paulo (UNIFESP)

{rcbarros,andre}@icmc.usp.br, {basgalupp,quiles}@unifesp.br

Abstract. *Decision tree induction algorithms are well known techniques for assigning objects to predefined categories in a transparent fashion. Most decision tree induction algorithms rely on a greedy top-down recursive strategy for growing the tree, and pruning techniques to avoid overfitting. Even though such a strategy has been quite successful in many problems, it falls short in several others. For instance, there are cases in which the hyper-rectangular surfaces generated by these algorithms can only map the problem description after several sub-sequential partitions, which results in a large and incomprehensible tree. Hence, we propose a new decision tree induction algorithm based on clustering which seeks to provide more accurate models and/or shorter descriptions, which are comprehensible for the end-user. We do not base our performance analysis solely on the straightforward comparison of our proposed algorithm to baseline methods. Instead, we propose a data-dependent analysis in order to look for evidences which may explain in which situations our algorithm outperforms a well-known decision tree induction algorithm.*

1. Introdução

A classificação é um tarefa de aprendizado supervisionado cujo objetivo geral é construir um modelo conciso de distribuição dos valores do atributo classe em função dos valores dos demais atributos, designados preditivos. Nas últimas décadas, pesquisadores na área de Aprendizado de Máquina (AM), estatística e reconhecimento de padrões têm investigado novos algoritmos de classificação tanto para problemas genéricos como para domínios específicos de aplicação. Dentre as diferentes formas de representação do conhecimento adquirido, destacam-se as árvores de decisão (ADs), uma técnica poderosa e amplamente utilizada em processos de tomadas de decisão inteligentes. Seu sucesso pode ser explicado por diversos fatores, tais como [Tan et al. 2005]: (i) interpretabilidade do conhecimento adquirido - uma árvore de decisão usa uma representação gráfica e pode ser facilmente convertida em regras; (ii) robustez na presença de ruídos; (iii) algoritmos de indução de árvores de decisão não são computacionalmente custosos, mesmo para grandes conjuntos de dados; e (iv) capacidade de lidar com atributos redundantes e irrelevantes, os quais, se não tratados adequadamente, podem prejudicar o desempenho do classificador. Alguns algoritmos bem conhecidos para indução de árvores de decisão são o ID3 [Quinlan 1986], o C4.5 [Quinlan 1993] e o CART (*Classification and Regression Trees*) [Breiman et al. 1984].

Embora a maioria dos algoritmos de indução de árvores de decisão empregue uma estratégia gulosa, *top down* e com particionamento recursivo, trabalhos mais recentes propõem outras estratégias [Basgalupp et al. 2009] [Barros et al. 2011]. Por exemplo, um tópico crescente na comunidade de AM e *Data Mining* (DM) é a utilização de um conjunto (*Ensemble*) de árvores de decisão. Um *Ensemble* é a composição de várias árvores de decisão construídas a partir dos exemplos de treino, e o resultado final da classificação é, geralmente, calculado por um esquema de votação [Hastie et al. 2009, Seni and Elder 2010]. Geralmente, um *Ensemble* de classificadores melhora a taxa de acerto quando comparado com um único classificador. No entanto, essa melhora se dá às custas da interpretabilidade, principal virtude das árvores de decisão. Um único

classificador pode ser interpretado por um especialista; porém, na prática, dificilmente um especialista consegue interpretar uma combinação de classificadores, mesmo que eles sejam, isoladamente, fáceis de serem compreendidos. Além de tal análise ser tediosa e demorada, os modelos de classificação que compõem um *Ensemble* estão sujeitos a inconsistências, as quais são necessárias para o aumento de sua acurácia preditiva, dificultando ainda mais a interpretabilidade. Uma vez que cada modelo pode ser considerado uma hipótese para explicar os padrões preditivos, isso significa que um *Ensemble* não representa uma única hipótese coerente sobre os dados, mas sim um conjunto grande de hipóteses inconsistentes, as quais podem confundir o especialista [Freitas et al. 2010]. Por essa razão, a utilização de *Ensembles* não é recomendada em domínios de aplicação em que a interpretabilidade é essencial.

Outra abordagem que tem sido constantemente utilizada é a indução de árvores de decisão por meio de Algoritmos Evolutivos (AEs), cuja principal vantagem está na capacidade de evitar convergência prematura para ótimos locais. Isso porque os AEs realizam uma busca robusta e global no espaço de soluções candidatas, sendo menos suscetíveis à convergência para ótimos locais. Adicionalmente, como resultado dessa busca global, os AEs tendem a lidar melhor que os métodos gulosos em relação à interação entre atributos, pois os relacionamentos complexos entre os atributos não detectados durante uma avaliação gulosa podem ser descobertos por um AE [Freitas 2008]. No entanto, os AEs também apresentam desvantagens, especialmente relacionadas ao tempo de execução. Os AEs constituem uma heurística sólida porém custosa computacionalmente. Além disso, um AE utiliza um grande número de parâmetros a serem definidos (e sintonizados) para otimizar o resultado de sua execução, o que também pode consumir um tempo considerável.

Como em qualquer algoritmo de AM desenvolvido até hoje, há prós e contras em escolher uma ou outra estratégia para indução de ADs. Além disso, algoritmos que apresentam bons resultados em determinados conjuntos de dados podem apresentar resultados ruins em outros. Isso é explicado pelo teorema NFL [Wolpert and Macready 1997]. Embora muitos estudos estejam constantemente propondo novos algoritmos de indução de classificadores, muitos deles negligenciam o fato de que o desempenho do classificador induzido é altamente dependente dos dados. Além disso, esses estudos normalmente apresentam análises superficiais que não indicam claramente os casos em que o classificador proposto apresentará ganho significativo em relação a outros classificadores.

Com o objetivo de melhorar a acurácia em relação aos algoritmos tradicionais para indução de árvores de decisão e ao mesmo tempo tentar preservar a interpretabilidade do modelo, este trabalho propõe o Clus-DTI (*Clustering for improving Decision Tree Induction*), um novo algoritmo de indução de AD baseado em agrupamento. A ideia chave deste trabalho é que um problema difícil pode ser decomposto em subproblemas mais simples. A hipótese é que é possível utilizar agrupamento para melhorar a classificação com a premissa de que, em vez de solucionar diretamente um problema difícil, é possível melhorar o desempenho da classificação ao solucionar de forma independente subproblemas mais fáceis.

A contribuição deste trabalho pode ser dividida em duas linhas: (i) é proposto um novo algoritmo de indução de AD baseado em agrupamento, e uma análise empírica é realizada com diversos conjuntos de dados públicos; (ii) são investigados cenários nos quais o algoritmo proposto é a melhor opção em relação ao C4.5, um algoritmo tradicional para indução de árvores de decisão. Essa investigação é baseada na análise de dependência de dados, isto é, da forma com que as características estruturais dos conjuntos de dados influenciam no desempenho do algoritmo proposto.

O restante deste trabalho é organizado como segue. Na Seção 2 é apresentado em detalhes o algoritmo Clus-DTI. A Seção 3 apresenta uma comparação entre o algoritmo proposto, Clus-DTI, e o tradicional algoritmo de indução de árvores de decisão, C4.5 [Quinlan 1993] (mais especificamente a sua versão em Java, denominada J48). Na Seção 4 é realizada a análise de dependência de dados, em que são apresentados os possíveis

cenários que sugerem a utilização do Clus-DTI. A Seção 5 apresenta os trabalhos relacionados. As conclusões e sugestões de trabalhos futuros são apresentados na Seção 6.

2. Algoritmo Clus-DTI

Neste trabalho, foi desenvolvido um novo algoritmo chamado Clus-DTI (*Clustering for improving decision tree induction*), cujo objetivo é utilizar agrupamento do conjunto de dados original para melhorar o desempenho das árvores de decisão geradas. Este trabalho é baseado na seguinte premissa: alguns problemas de classificação são difíceis de serem solucionados; assim, em vez de construir uma única árvore de decisão para solucionar um problema de classificação potencialmente difícil, talvez dividir o conjunto de dados em pequenos subproblemas pode tornar menos complexa a solução do problema completo. Em outras palavras, o algoritmo proposto aproxima a função alvo em regiões específicas do espaço de busca. Essas regiões não são aleatoriamente escolhidas, mas sim definidas por um algoritmo de agrupamento disseminado na área. Logo, é razoável supor que exemplos (objetos) pertencentes à mesma distribuição de classes possam ser previamente agrupados e, assim, facilitar a classificação final. No entanto, cabe destacar que o objetivo não é encontrar um mapeamento perfeito entre grupos (*clusters*) e classes. A hipótese deste trabalho é que solucionar subproblemas independentemente em vez de solucionar diretamente o problema completo pode gerar melhores resultados de forma geral. Essa ideia não é nova e tem motivado diversas estratégias na área da ciência da computação, como, por exemplo, a estratégia dividir-para-conquistar.

Assim, Clus-DTI combina dois algoritmos de Aprendizado de Máquina para gerar os modelos de classificação. Seu funcionamento é apresentado como segue.

1. Dado um conjunto de treino $\mathbf{X}$, Clus-DTI divide $\mathbf{X}$ em p partições $\mathbf{P_i}$ de $i+1$ conjuntos (sem sobreposição) de $\mathbf{X}$, tal que a primeira partição tenha dois grupos, a segunda tenha três grupos, e assim por diante. Por exemplo, partição $\mathbf{P_1} = \{C_1, C_2\}$, e na forma geral $\mathbf{P_i} = \{C_1, ..., C_{i+1}\}$, para $1 \leq i \leq p$. Adicionalmente, $\mathbf{C_j} \cap \mathbf{C_k} = \emptyset$ para $j \neq k$ e $\mathbf{C_1} \cup ... \cup \mathbf{C_{i+1}} = \mathbf{X}$. Neste passo, Clus-DTI usa o algoritmo bem conhecido *Expectation Maximization* (EM) [Dempster et al. 1977, Hastie et al. 2009]. Uma vez que o EM é um método estatístico de agrupamento em que todos exemplos pertencem a todos os grupos com uma dada probabilidade, ele atribuirá cada exemplo ao grupo mais provável. Obviamente, Clus-DTI ignora o atributo classe na etapa de agrupamento.
2. Após cada exemplo ser atribuído a um único grupo, Clus-DTI cria uma árvore de decisão para cada grupo utilizando seus respectivos dados (exemplos) como conjunto de treino. Nessa etapa, é utilizado o algoritmo C4.5 [Quinlan 1993]. Assim, a partição $\mathbf{P_i}$ terá um conjunto de $i+1$ árvores de decisão distintas, $D = \{Dt_1, ..., Dt_{i+1}\}$, treinadas de acordo com os dados pertencentes a cada grupo.
3. Para decidir qual partição será selecionada dentre as p disponíveis, Clus-DTI emprega duas opções. A primeira é avaliar cada partição utilizando um critério bem conhecido para validação de agrupamento, o *Simplified Silhouette Width Criterion* (SSWC) [Vendramin et al. 2010]. A partição selecionada é aquela que maximiza SSWC, ou seja, minimiza a dissimilaridade intra-grupo e maximiza a dissimilaridade inter-grupo. A medida de dissimilaridade utilizada foi a distância de Mahalanobis [Mahalanobis 1936], já que os grupos produzidos por essa versão rígida do EM terão um formato elipsoidal. A segunda opção é mais simples e direta: escolher o número de grupos da partição que maximiza a acurácia de treino. Uma vez que o algoritmo está sendo proposto para problemas de classificação, talvez a melhor opção para escolher o número ideal de grupos seja, de fato, um critério relacionado à própria tarefa de classificação. Por ser a mais utilizada, a acurácia foi escolhida como medida de avaliação.
4. Após a partição $\mathbf{P_i}$ ser escolhida, cada exemplo do conjunto de teste $\mathbf{Z}$ é atribuído a um dos $i+1$ grupos, e então a árvore de decisão correspondente $\mathbf{Dt_k}$, $1 \leq k \leq i+1$ é utilizada para classificar esse exemplo. Esse procedimento é repetido para todos os exemplos do conjunto de teste $\mathbf{Z}$.

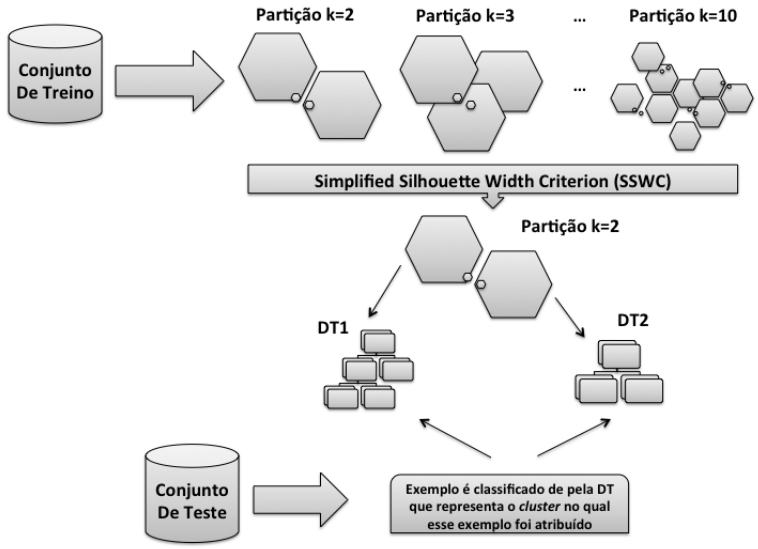

Figura 1. Exemplo de execução do Clus-DTI. Neste cenário, SSWC seleciona a partição com dois grupos como melhor opção dentre as 9 possíveis.

A Figura 1 ilustra esses passos executados pelo Clus-DTI. Nesse cenário fictício, a partição $\mathbf{P_2}$ é aquela que maximiza SSWC (considerando $p = 9$) e, portanto, as duas árvores de decisão Dt_1 e Dt_2 são construídas a partir dos exemplos de treino pertencentes a cada um dos dois grupos. A seguir, cada exemplo do conjunto de teste é atribuído ao grupo com o menor valor da distância de Mahanalobis. A árvore de decisão correspondente ao grupo em questão é então utilizada para classificar esse exemplo em particular.

Novamente, é importante destacar que não se espera um mapeamento total entre grupos e classes. Mesmo que o agrupamento dos dados possa gerar grupos contendo exemplos de uma mesma classe, o que significaria dizer que EM foi capaz de descobrir os parâmetros corretos de distribuição de probabilidades de tal classe, este dificilmente será o caso para a grande maioria dos conjuntos de dados. O objetivo do algoritmo Clus-DTI é encontrar subconjuntos de dados que tornam o problema mais fácil de ser resolvido do que o original. Em outras palavras, o objetivo é identificar situações em que a soma dos desempenhos das árvores de decisão criadas a partir dos subconjuntos apresente melhor resultado geral do que uma simples árvore considerando o conjunto de dados original. Embora esse conceito seja similar ao conceito de *Ensemble*, há uma diferença importante: não é utilizado nenhum esquema de votação para classificar os exemplos. A ausência desse esquema de votação implica em uma grande vantagem de Clus-DTI em relação aos *Ensembles*: é possível acompanhar a árvore de decisão usada para classificar um exemplo em particular em vez de apenas receber a média dos resultados fornecidos por um conjunto de árvores de decisão inconsistente.

3. Análise Comparativa

Para avaliar o desempenho do algoritmo Clus-DTI, foram utilizados 29 conjuntos de dados obtidos da *UCI Machine Learning Repository* [Frank and Asuncion 2010]. Os resultados obtidos foram comparados com o J48 (implementação em Java do algoritmo C4.5, disponível na ferramenta Weka [Witten and Frank 1999]). Os conjuntos de dados utilizados foram: *anneal, audiology, autos, balance-scale, breast-cancer, breast-w, colic, credit-a, credit-g, diabetes, glass, heart-c, heart-h, heart-statlog, hepatitis, ionosphere, iris, kr-vs-kp, labor, letter, lymph, mushroom, primary-tumor, segment, sick, sonar, soybean, vote* e *waveform-5000*. Os conjuntos de dados com k classes, sendo $k > 2$, foram transformados em k conjuntos de duas classes (uma classe contra todas as outras).

Tabela 1. Comparação entre Clus-DTI e J48.

	J48	Clus-DTI
# Wins	34	39
Greatest Difference	8.88%	10%
Average Difference	1.82%	1.84%
Smallest Difference	0.8%	0.01%
Difference $\geq 1\%$	14	19
Difference $\geq 3\%$	10	5
Difference $\geq 5\%$	6	2
Difference $\geq 7\%$	2	1
Difference $\geq 10\%$	0	1

Assim, todos os problemas de classificação utilizados neste trabalho são binários. Essa restrição de lidar apenas com problemas binários é devido às medidas de complexidade [Ho and Basu 2002, Ho et al. 2006] utilizadas na análise de dependência de dados, as quais, em sua maioria, são aplicáveis apenas a problemas de duas classes. Com essas transformações, os 29 conjuntos de dados originais se transformaram em 156 novos conjuntos, os quais foram divididos como segue: 70% para treino[1] e 30% para teste.

A Tabela 1 apresenta os resultados da comparação entre Clus-DTI e J48. *# Wins* indica o número de conjuntos de dados em que cada método superou o outro em termos de acurácia. *Greatest (Average/Smallest) Difference* é a maior (média/menor) diferença em termos de acurácia entre os métodos quando um deles supera o outro. *Difference* $\geq x\%$ é o número de conjuntos de dados em que um método obteve uma diferença maior ou igual a $x\%$ em termos de acurácia.

Em termos de número de vezes em que um algoritmo supera o outro, é possível notar que os dois métodos apresentam resultados similares. Sem considerar os 83 casos em que ocorreu empate, o Clus-DTI superou mais vezes o J48 do que o contrário. Isso é um indicativo de que é vantajoso realizar o agrupamento como etapa prévia à classificação em muitos conjuntos de dados. Entretanto, o teste estatístico não-paramétrico de *Wilcoxon Signed Rank* [Wilcoxon 1945] não apontou nenhuma diferença significativa entre os resultados obtidos pelos dois algoritmos considerando todos os conjuntos de dados. Uma vez que o Clus-DTI é mais custoso computacionalmente que o J48, não seria prudente sugerir o uso de Clus-DTI para quaisquer bases de dados.

Assim, considerou-se de extrema importância investigar os casos em que Clus-DTI supera J48 para, então, tentar encontrar uma relação desses resultados com as características (metadados e medidas de complexidade) desses conjuntos de dados. Isso possibilitaria, então, sugerir a utilização ou não do Clus-DTI de acordo com as características do conjunto de dados a ser utilizado. Essa análise é apresentada na próxima seção.

4. Análise de Dependência dos Dados

Costuma-se dizer que o desempenho de um classificador é fortemente dependente dos dados. Não obstante, trabalhos que propõem novos classificadores normalmente negligenciam essa dependência durante a análise de seus desempenhos. Normalmente, a estratégia adotada é a de fornecer alguns casos em que o classificador proposto supera algum classificador tradicional (comparativo) conforme alguma medida de avaliação dos resultados. Da mesma forma, os estudos teóricos que se propõem a analisar o comportamento dos classificadores também tendem a negligenciar essa dependência. Recentemente, alguns esforços têm sido feitos na busca pela associação das características dos dados com o desempenho de diferentes classificadores com o objetivo de construir sistemas de recomendação. Os algoritmos conhecidos como meta-aprendizado são

[1]Recomenda-se a leitura de [Sánchez et al. 2003] para os interessados em saber como obter conjuntos de treino de alta qualidade para classificação.

uma tentativa de compreender os dados *a priori* da execução de um algoritmo de aprendizado. Dados que descrevem as características dos conjuntos de dados e algoritmos de aprendizado são chamados de metadados, e um novo algoritmo de aprendizado é responsável por interpretar esses metadados e sugerir um modelo de aprendizado em particular [Smith-Miles 2009].

Meta-aprendizado para a seleção de algoritmos geralmente se baseia em medidas estatísticas ou informações teóricas obtidas dos conjunto de dados. Embora essas descrições possam ser suficientes para a recomendação de algoritmos, elas não explicam às características geométricas das distribuições dos exemplos em suas classes, ou seja, negligenciam a maneira pela qual as classes são separadas ou intercaladas, o que representa um fator crítico para determinar a precisão da classificação. Dessa forma, medidas geométricas foram propostas em [Ho and Basu 2002, Ho et al. 2006] para caracterizar a complexidade geométrica de problemas de classificação. O estudo dessas medidas foi um primeiro esforço para compreender melhor a dependência entre conjuntos de dados e classificadores. Além disso, ao estabelecer de forma quantitativa a dificuldade de um problema de classificação, diversos aspectos do processo de classificação podem ser estabelecidos, como recomendação de algoritmos, pré-processamento guiado dos dados e desenvolvimento de algoritmos de classificação para domínios específicos.

Nesta seção, são utilizados os metadados citados e também as medidas propostas em [Ho and Basu 2002, Ho et al. 2006] com o objetivo de compreender as características e a complexidade geométrica de diversos conjuntos de dados para classificação. Primeiro, um resumo dos metadados e das medidas geométricas utilizadas é apresentado. Deve-se observar que as medidas geométricas visam encontrar a complexidade geométrica aparente dos dados, considerando que a probabilidade de distribuição real de cada classe é desconhecida.

4.1. Metadados

A primeira tentativa de caracterizar um conjunto de dados para avaliação de desempenho de algoritmos de aprendizado foi realizada por [Rendell et al. 1987]. A abordagem adotada teve como objetivo prever o tempo de execução dos algoritmos de classificação utilizando exemplos bastante simples de metacaracterísticas, como número de atributos e número de exemplos. Um avanço dessa abordagem é o projeto STATLOG [Michie et al. 1994], o qual investigou o desempenho de vários algoritmos de aprendizado utilizando mais de 20 conjuntos de dados. Outros autores aprofundaram essa abordagem utilizando o mesmo conjunto de metacaracterísticas para caracterização dos dados [Brazdil et al. 1994, Gama and Brazdil 1995]. Esse conjunto pode ser dividido em três categorias: (i) simples; (ii) estatístico; e (iii) baseado na teoria da informação. Um conjunto melhorado de metacaracterísticas foi posteriormente apresentado em [Kalousis 2002], e, dessas, as seguintes são adotadas neste trabalho: número de exemplos (N), número de atributos (n), número de atributos contínuos (con), número de atributos nominais (nom), número de atributos binários (bin), número de classes (cl), percentual de valores ausentes ($\%mv$), entropia de classe ($H(Y)$), entropia média dos atributos (MAE), média do atributo Gini (MAG), informação mútua média entre classes e atributos (MMI) e coeficiente de incerteza (UC). Além dessas medidas apresentadas em [Kalousis 2002], a razão entre o número de exemplos das classes com menor e maior quantidade de exemplos também foi adotada. Essa medida indica o nível de balanceamento do conjunto de dados. Quanto menor o valor (entre zero e um), maior o desbalanceamento.

Em seguida, são apresentadas medidas que procuram explicar como os dados são estruturados geometricamente a fim de avaliar a dificuldade de um problema de classificação.

4.2. Medidas de Complexidade Geométricas

Em [Ho and Basu 2002, Ho et al. 2006], um conjunto de medidas é apresentado para caracterização do conjunto de dados conforme sua estrutura geométrica. Essas medidas são dividas em três categorias: (i) medidas de sobreposição no espaço de atributos; (ii)

medidas de separabilidade de classe; e (iii) medidas de geometria, topologia e densidade. A Tabela 2 apresenta um sumário dessas medidas.

Tabela 2. Sumarização das medidas de complexidade.

Medida	Categoria	Descrição
F1	Sobreposição no Espaço de Atributos	Computa a taxa de discriminação de Fisher. Um valor alto para F1 indica que existe um vetor de transformação que pode separar os exemplos pertencentes a diferentes classes após a projeção dos exemplos neste novo espaço de atributos.
F2	Sobreposição no Espaço de Atributos	Computa a sobreposição dos limites definidos pelos exemplos de cada classe. Um valor baixo da métrica representa que os atributos podem discriminar os exemplos de classes distintas.
F3	Sobreposição no Espaço de Atributos	Computa o poder discriminativo de atributos individuais e retorna o valor do atributo que é capaz de discriminar o maior número de exemplos de treino. Um problema é dito fácil se existe um atributo para o qual os intervalos dos valores gerados por cada classe não se sobrepõem.
F4	Sobreposição no Espaço de Atributos	Segue a mesma ideia utilizada em F3, porém considera o poder discriminativo de todos os atributos (coletivamente).
N1	Separabilidade de Classes	Fornece uma estimativa do comprimento dos limites da classe. Valores altos desta medida sugerem que a maioria dos exemplos está situada nas fronteiras de classes, indicando que o problema em questão é de solução complexa.
N2	Separabilidade de Classes	Compara a distância dos dados de uma classe com a distância aos vizinhos mais próximos pertencentes às outras classes. Valores menores de N2 sugerem que os exemplos pertencentes à mesma classe estão situados em locais próximos no espaço de atributos, e são, portanto, mais facilmente classificados.
N3	Separabilidade de Classes	Denota quão próximos estão os exemplos pertencentes a classes distintas. Retorna o erro *leave-one-out* do vizinho mais próximo (classificador kNN assumindo k=1). Valores baixos de N3 indicam que existe uma grande margem entre classes.
L1	Separabilidade de Classes	Avalia a separabilidade linear do conjunto de treino. Para tal, a soma das diferenças entre o rótulo obtido por um classificador linear e o rótulo real dos exemplos é computada. L1 retorna o valor zero quando o problema é linearmente separável.
L2	Separabilidade de Classes	Fornece informações sobre a separabilidade linear do conjunto de treino. Um classificador linear, como definido em L1, é utilizado e L2 retorna o erro de treinamento.
L3	Geometria, Topologia e Densidade	Implementa a medida de não-linearidade proposta em [Hoekstra and Duin 1996]. A partir do conjunto de treino, o método define um conjunto de testes por meio de uma interpolação linear com coeficientes aleatórios entre os pares de exemplos de uma mesma classe selecionados aleatoriamente.
N4	Geometria, Topologia e Densidade	Define um conjunto de testes conforme apresentado na métrica L3 e retorna o erro utilizando um classificador 1NN.
T1	Geometria, Topologia e Densidade	Descreve a forma das variedades das classes em relação ao subconjunto de adesão. Em suma, o subconjunto de adesão é definido por uma esfera centrada em um exemplo, e tal esfera cresce tanto quanto for possível antes de tocar um exemplo pertencente à outra classe. Assim, um subconjunto de adesão contém um conjunto de exemplos de uma mesma classe de tal forma que é impossível da mesma crescer sem que exemplos de uma outra classe sejam incluídos.
T2	Geometria, Topologia e Densidade	Retorna o razão entre o número de exemplos do conjunto de dados pelo número de atributos. Se apresenta como um indicador da dispersão dos dados.

4.3. Resultados

Foram calculadas 13 medidas de complexidade e 13 metacaracterísticas para os 156 conjuntos de dados. Após, um conjunto de treino foi criado da seguinte forma: cada exemplo correspondendo a um dos 156 conjuntos de dados e cada atributo correspondendo a uma das 26 medidas. Foram também incluídos outros dois atributos, um correspondendo ao número k de grupos (*clusters*) e outro atributo categórico indicando qual método (Clus-DTI ou J48) apresentou o melhor desempenho, em termos de acurácia, para o conjunto de dados em questão. Esse processo resultou em um conjunto de treino com 156 linhas (exemplos) e 28 colunas (atributos).

A ideia de obter esse conjunto de treino teve o objetvo de realizar uma análise descritiva dos resultados e, assim, entender quais aspectos dos dados têm maior influência em determinar o desempenho dos algoritmos. Em particular, foram investigadas evidências que poderiam ajudar o usuário a escolher *a priori* um dos algoritmos de indução. Para isso, foi criada uma árvore de decisão, a partir do conjunto de treino gerado, com o intuito de interpretar essa árvore e entender em quais circunstâncias um algoritmo supera o outro. A Figura 2 ilustra essa árvore de decisão, a qual explica o comportamento de aproximadamente 85% do conjunto de treino.

Ao analisar a árvore de decisão, é possível concluir que Clus-DTI é mais robusto em relação a valores ausentes do que o J48. Uma hipótese que pode explicar esse comportamento é o fato de que o número de valores ausentes tende a ser amortizado entre

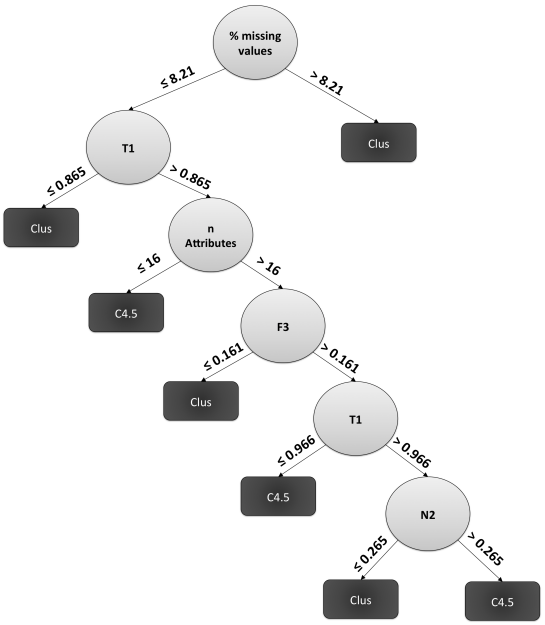

Figura 2. Árvore de decisão que descreve o relacionamento entre as medidas e o melhor algoritmo a ser utilizado.

os grupos, e então cada árvore criada tem um número menor desses valores para tratar. Outra conclusão que pode ser feita é em relação aos conjuntos de dados com poucos valores ausentes (aqueles em que a percentagem de valores ausentes é menor que 8.21%) e a medida T1. Só para relembrar, a medida T1 introduz o conceito de subconjuntos aderentes. Ela projeta hiperesferas que incluem exemplos da mesma classe e normaliza o número de hiperesferas pelo número total de exemplos. Quanto maior é o valor de T1, menor é a tendência dos exemplos serem agrupados em hiperesferas monoclasse. A árvore de decisão indica que os casos em que os exemplos têm uma tendência alta de serem agrupados em hiperesferas monoclasse ($T1 \leq 0.865$) são melhor classificados com o Clus-DTI. Essa parece ser uma conclusão coerente, pois o Clus-DTI agrupa os conjuntos de dados em subconjuntos, e quanto mais puros são esses subconjuntos, mais fáceis são de serem classificados. Um problema potencial nessa análise é que o Clus-DTI não gera grupos limitados por hiperesferas, mas sim por hiperelipses. Mesmo assim, essa descoberta parece ser válida embora toda suposição seja baseada em hiperesferas. Para trabalhos futuros, tem-se a intenção de modificar T1 para que ela possa projetar hiperelipses.

Para os casos restantes (baixa percentagem de valores ausentes e altos valores de T1), a árvore de decisão apresenta outras relações relevantes. Por exemplo, ela testa o número de dimensões e também o F3. A Figura 3 ilustra a análise bivariada de F3 e o número de atributos. O retângulo horizontal indica o primeiro teste sobre o número de atributos separando aqueles exemplos com menos de 16 dimensões. Os círculos que estão dentro desse retângulo são os que já foram filtrados pela árvore de decisão durante os testes sobre a baixa percentagem de valores ausentes e também sobre o valor de T1. Portanto, o terceiro teste sobre o número de atributos de fato separa os casos em que o J48 supera o Clus-DTI. O retângulo vertical, por sua vez, representa o resultado do quarto teste, sobre os valores de F3 (com a restrição adicional de ter mais que 16 dimensões, pois o quarto teste é uma consequência do terceiro).

Uma análise semelhante pode ser feita com os atributos restantes da árvore de decisão. O próximo passo é tentar entender o comportamento dos casos extremos, ou seja, casos em que a diferença de desempenho entre Clus-DTI e J48 é bem alta. As Tabelas 3 and 4 apresentam esses casos, bem como algumas medidas como: (i) balanceamento (*Balance*), a taxa de frequência da classe menos frequente sobre a classe mais frequente,

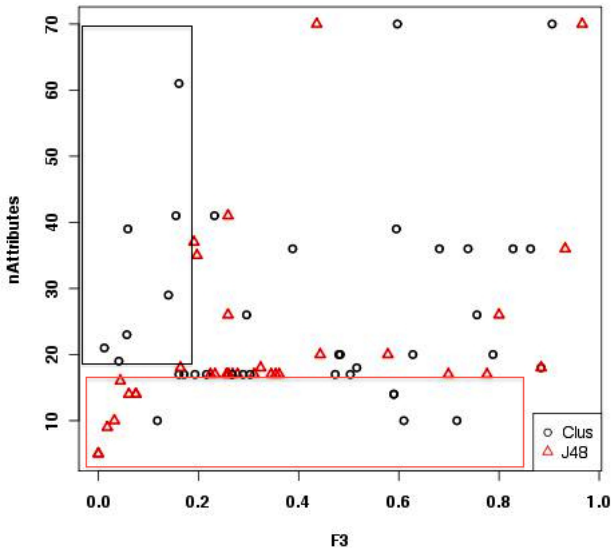

Figura 3. Análise bivariada entre F3 e o número de atributos. Os círculos indicam os conjuntos de dados em que o Clus-DTI supera o J48. Os triângulos indicam os conjuntos de dados em que o J48 supera o Clus-DTI.

Tabela 3. Conjuntos de dados em que o Clus-DTI supera o J48 com uma margem alta.

Conjunto de Dados	Balance	Original TS	Clus Average TS	K	Accuracy J48	Accuracy Clus
Audiology C_2	0.3393	11	8.14	2	96%	100%
Lymph C_2	0.68	15	7.28	3	70%	80%
Primary Tumor C_{17}	0.09	1	2.43	6	91%	94%
Soybean C_{12}	0.03	1	9.53	2	96%	100%

(ii) tamanho da árvore gerada pelo J48 considerando o conjunto de dados completo (*Original TS*), (iii) média ponderada dos tamanhos das árvores geradas pelos grupos formados por Clus-DTI (*Clus Average TS*), (iv) acurácia obtida pelo J48, e (v) acurácia obtida pelo Clus-DTI.

Uma análise cuidadosa indica que o Clus-DTI apresenta melhores resultados que o J48 para conjuntos de dados não balanceados (*primary tumor* e *soybean*). Isso porque o J48 gera apenas um nodo folha representando a classe mais frequente, enquanto o Clus-DTI é apto a explorar as classes menos frequentes nos seus grupos. Em *soybean*, por exemplo, o Clus-DTI classifica corretamente todos os exemplos de uma classe rara.

Outro fator interessante é que o Clus-DTI gera árvores pequenas para todos os conjuntos de dados (excluindo os conjuntos de dados não balanceados, em que o J48 gera apenas um nodo). De fato, isso é previsível porque o conjunto de dados está sendo reduzido e, em alguns casos, estão sendo eliminadas diversas arestas que poderiam ser construídas devido às regiões em que as classes se sobrepõem com frequência. Árvores menores são mais fáceis de interpretar, além de serem mais robustas a *overfitting*.

Tabela 4. Conjuntos de dados em que o J48 supera o Clus-DTI com uma margem alta.

Conjunto de Dados	Balance	Original TS	Clus Average TS	K	Accuracy J48	Accuracy Clus
Autos C_3	0.47	33	6.44	2	79%	73%
Heart-c C_1	0.83	28	11.94	4	86%	80%
Heart-Statlog	0.8	37	16.98	3	82%	74%
Ionosphere	0.56	15	8.13	2	93%	87%

Analisando o conjunto de dados *hepatitis*, por exemplo (não apresentados nas Tabelas 3 e 4), embora a diferença entre J48 e Clus-DTI seja de 3.85% a favor do J48, Clus-DTI gerou duas árvores distintas com 5 nodos cada, enquanto a árvore gerada pelo J48 tem 17 nodos. Dependendo do cenário em questão, é possível que o usuário prefira duas árvores pequenas, as quais são mais fáceis de analisar, às custas de 3.85% de acurácia. Esse tipo de situação deixa claro que o número de vitórias de Clus-DTI poderia ser ajustado ao levar-se em conta o custo/benefício entre acurácia e interpretabilidade (em termos de tamanho da árvore).

Por fim, a última recomendação dessa análise é para os usuários que realmente precisam de modelos simples e interpretáveis: se nenhum dos cenários apontados anteriormente sugerem que Clus-DTI é a melhor opção em relação ao J48, mesmo assim ainda pode ser útil utilizá-lo devido ao tamanho das árvores geradas ser menor. No entanto, essa vantagem desaparece em casos que o número de grupos (*clusters*) sugeridos pelo Clus-DTI multiplicado pela média dos tamanhos das árvores desses grupos excede o tamanho da árvore do J48.

5. Trabalhos Relacionados

[Gaddam et al. 2007] propuseram um algoritmo denominado *cascading* K-Médias [Lloyd 1982] e ID3 [Quinlan 1986] com objetivo de melhorar a detecção de anomalias nos dados. Nesses modelos, o conjunto de treino é dividido em k grupos disjuntos (k é empiricamente definido) pelo algoritmo de agrupamento K-Médias. Um exemplo de teste é primeiramente associado a um dos grupos gerados, e seu rótulo é definido conforme a maioria dos exemplos rotulados dentro do grupo selecionado. Uma árvore de decisão é construída com o algoritmo ID3 a partir dos exemplos do conjunto de treino associado a cada grupo (uma árvore por grupo). Essas árvores também são utilizadas para rotular os exemplos de teste associados aos seus respectivos grupos. O modelo também mantém a informação dos f grupos mais próximos ao exemplo de teste e utiliza um esquema combinado para verificar se o rótulo sugerido pelo ID3 coincide com a do K-Médias. Observando a semelhança entre o modelo de Gaddam et al. com o algoritmo proposto neste trabalho, as principais diferenças entre ambos são destacadas a seguir: (i) Clus-DTI faz uso de dois algoritmos de AM mais robutos do que os utilizados em [Gaddam et al. 2007]; (ii) Clus-DTI não possui o esquema de rotulação combinada dos exemplos de teste utilizado em [Gaddam et al. 2007]. Dessa forma, o modelo proposto permite, de maneira compreensiva, verificar quais foram as regras utilizadas no processo de validação, apresentando, assim, uma vantagem sobre o modelo proposto em [Gaddam et al. 2007] e outros que também adotam um sistema de rotulação combinada ou baseada em votos; (iii) Clus-DTI não está restrito a aplicações de detecção de anomalias; (iv) Clus-DTI define automaticamente o número "ideal' de grupos segundo critérios de validação dos grupos gerados; (v) Clus-DTI é conceitualmente mais simples e de mais fácil implementação que o K-Médias+ID3, mesmo considerando que algoritmos mais robustos são utilizados.

6. Conclusões e Trabalhos Futuros

As principais conclusões desse trabalho são apresentadas abaixo:

- Clus-DTI se apresenta como um algoritmo mais robusto a valores ausentes, pois superou o J48 em todos os conjuntos de dados cujos percentuais de valores ausentes superaram um dado limiar (8%).
- A aderência ao conceito de subconjunto é uma importante variável na seleção de algoritmos. Todos os casos em que os exemplos possuem uma maior tendência de serem agrupados em um único grupo hiperesférico (isto é, $T1 \leq 0.865$) são melhor tratados pelo Clus-DTI. Esse fato se mostra coerente, pois o agrupamento é realizado *a priori* ao processo de classificação. No entanto, é importante salientar que os grupos gerados pelo EM possuem forma geométrica de hiperelipses e não hiperesperas. Assim, a reformulação de T1 para o crescimento de hiperelipses pode gerar resultados ainda superiores.

- Outros fatores importantes na seleção de um algoritmo estão relacionados ao número de atributos, à medida F3 e à medida N2. Nesse último caso, é intuitivo pensar que valores menores de N2 (classes bem separadas de acordo com a distância euclidiana) aumentam as chances do Clus-DTI produzir melhores resultados, assumindo que este primeiramente realiza o agrupamento dos exemplos, tornando o processo de classificação mais simples.
- Clus-DTI é robusto a classes desbalanceadas. Ele fornece resultados melhores do que J48 em todos os casos em que a árvore resultante apresenta um único nodo. Clus-DTI reduz o impacto do desbalanceamento das classes quando essas são agrupadas em grupos distintos, possibilitando, inclusive, a produção de regras que descrevem classes pouco representadas.
- J48 apresentou resultados superiores em 7 dos 8 conjuntos de dados formados por classes mais balanceadas.

Este trabalho se caracteriza como um primeiro esforço no estudo dos potenciais benefícios do agrupamento de dados na classificação. Ele apresenta um novo algoritmo de classificação que pode ser visto como um arcabouço mais elegante do que simplesmente pré-processamento de dados por meio de técnicas de agrupamento. Ele abriu diversas linhas de trabalhos futuros. Pretende-se implementar vários métodos alternativos para escolher o melhor valor de k. Em particular, duas estratégias: (i) utilizar a acurácia de um conjunto de validação para escolher o melhor valor de k, e (ii) implementar a abordagem proposta por [Kothari and Pitts 1999] para determinar o valor de k. Além disso, também pretende-se testar outros algoritmos de agrupamento, bem como outros algoritmos de classificação a fim de avaliar se as mesmas conclusões podem ser generalizadas para qualquer par de algoritmos. Finalmente, tem-se a intenção de testar Clus-DTI em conjuntos de dados artificiais, com a finalidade de garantir sua eficácia em diferentes cenários, tais como conjuntos de dados linearmente separáveis, não-linearmente separáveis, desbalanceados, com alta dimensionalidade e esparsos.

Referências

Barros, R. C., Ruiz, D. D., and Basgalupp, M. P. (2011). Evolutionary model trees for handling continuous classes in machine learning. *Information Sciences*, 181:954–971.

Basgalupp, M., Barros, R. C., de Carvalho, A., Freitas, A., and Ruiz, D. (2009). Legal-tree: a lexicographic multi-objective genetic algorithm for decision tree induction. In Shin, S., Ossowski, S., Martins, P., Menezes, R., Virol, M., Hong, J., Shin, D., Palakal, M., Fritzke, U., Crosby, M., and Haddad, H., editors, *Proceedings of the 2009 ACM Symposium on Applied Computing*, pages 1085–1090. ACM Press.

Brazdil, P., Gama, J. a., and Henery, B. (1994). Characterizing the applicability of classification algorithms using meta-level learning. In *Proceedings of the European conference on machine learning on Machine Learning*, pages 83–102, Secaucus, NJ, USA. Springer-Verlag New York, Inc.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Frank, A. and Asuncion, A. (2010). UCI machine learning repository.

Freitas, A. A. (2008). *Soft Computing for Knowledge Discovery and Data Mining*, chapter A Review of evolutionary Algorithms for Data Mining, pages 79–111. Springer US.

Freitas, A. A., Wieser, D. C., and Apweiler, R. (2010). On the importance of comprehensible classification models for protein function prediction. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 7:172–182.

Gaddam, S. R., Phoha, V. V., and Balagani, K. S. (2007). K-means+id3: A novel method for supervised anomaly detection by cascading k-means clustering and id3 decision

tree learning methods. *IEEE Transactions on Knowledge and Data Engineering*, 19:345–354.

Gama, J. and Brazdil, P. (1995). Characterization of classification algorithms. In *Proceedings of the 7th Portuguese Conference on Artificial Intelligence: Progress in Artificial Intelligence*, pages 189–200, London, UK. Springer-Verlag.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer Series in Statistics. Springer, 2nd ed. 2009. corr. 3rd printing edition.

Ho, T., Basu, M., and Law, M. (2006). Measures of geometrical complexity in classification problems. In Jain, L., Wu, X., Basu, M., and Ho, T. K., editors, *Data Complexity in Pattern Recognition*, Advanced Information and Knowledge Processing, pages 1–23. Springer London.

Ho, T. K. and Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(3):289–300.

Hoekstra, A. and Duin, R. P. (1996). On the nonlinearity of pattern classifiers. In *In Proceedings of the 13th ICPR*, pages 271–275.

Kalousis, A. (2002). *Algorithm Selection via Meta-Learning*. PhD thesis, Université de Genève, Centre Universitaire d'Informatique.

Kothari, R. and Pitts, D. (1999). On finding the number of clusters. *Pattern Recognition Letters*, 20(4):405 – 416.

Lloyd, S. (1982). Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129 – 137.

Mahalanobis, P. C. (1936). On the generalised distance in statistics. In *Proceedings National Institute of Science, India*, volume 2, pages 49–55.

Michie, D., Spiegelhalter, D. J., Taylor, C. C., and Campbell, J., editors (1994). *Machine learning, neural and statistical classification*. Ellis Horwood, Upper Saddle River, NJ, USA.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Rendell, L., Sheshu, R., and Tcheng, D. (1987). Layered concept-learning and dynamically variable bias management. In *Proceedings of the 10th international joint conference on Artificial intelligence - Volume 1*, pages 308–314, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Sánchez, J. S., Barandela, R., Marqués, A. I., Alejo, R., and Badenas, J. (2003). Analysis of new techniques to obtain quality training sets. *Pattern Recogn. Lett.*, 24:1015–1022.

Seni, G. and Elder, J. (2010). *Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions*. Morgan and Claypool Publishers.

Smith-Miles, K. A. (2009). Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Comput. Surv.*, 41:6:1–6:25.

Tan, P.-N., Steinbach, M., and Kumar, V. (2005). *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Vendramin, L., Campello, R. J. G. B., and Hruschka, E. R. (2010). Relative clustering validity criteria: A comparative overview. *Stat. Anal. Data Min.*, 3:209–235.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics*, 1:80–83.

Witten, I. H. and Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.