ESTUDO COMPARATIVO ENTRE ALGORITMOS DE APRENDIZADO INCREMENTAL DE ÁRVORES DE DECISÃO

FELIPE R. P. CARNEIRO, ANDRÉ P. LEMOS

Programa de Pós-Graduação em Engenharia Elétrica - Universidade Federal de Minas Gerais - Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG,

E-mails: feliperoncalli@ufmg.br, andrepl@cpdee.ufmg.br

Abstract— This study aims to evaluate four decision trees algorithms with online learning. The evaluation will use the same datasets (databases). These bases have heterogeneous characteristics, thus allowing evaluated algorithms can be evaluated in different situations. The evaluated algorithms are: ID5R, ITI, VFDT and CVFDT. It's expected to demonstrate their robustness and show where each of them provides the best results, using accuracy, precision, sensitivity and specificity as metrics.

Keywords--Incremental learning, online learning, ID5R, ITI, VFDT, CVFDT, Hoeffding Tree.

Resumo— Este estudo tem por objetivo avaliar quatro algoritmos de árvores de decisão com aprendizado online. A avaliação utilizará os mesmos conjuntos de dados (bases). Essas bases apresentam características heterogêneas, permitindo assim que os algoritmos avaliados possam ser avaliados em situações distintas. Os algoritmos avaliados são: ID5R, ITI, VFDT e CVFDT. Espera-se demonstrar sua robustez e onde cada um deles apresenta os melhores resultados, utilizando acurácia, precisão, sensibilidade e especificidade como métricas.

Palavras-chave— Aprendizado Incremental, Aprendizado online, ID5R, ITI, VFDT, CVFDT, árvore de Hoeffding.

1 Introdução

O propósito de algoritmos de aprendizagem online (incremental) é aprender um conceito de uma entrada de streaming de exemplos não assumindo nenhuma distribuição subjacente, esperando que sem a dependência de ordem consiga classificar os exemplos da maneira mais assertiva possível.

Em muitas aplicações, novas informações podem surgir ao longo do tempo. Por exemplo, um sistema de reconhecimento facial certamente não irá reconhecer todas as variações faciais com antecedência, pois rostos são facilmente alterados. Assim, existe a necessidade de implementações de processo dinâmicos baseados em aprendizagem incremental. As técnicas de aprendizagem incremental envolvem a adaptação gradual das estruturas de aprendizagem (Utgoff, 1989).

A principal diferença entre algoritmos de árvore de decisão incremental quando comparado aos algoritmos de árvore de decisão com aprendizado em lote, como ID3 e C4.5 (têm acesso a todos os exemplos na memória), é que algoritmos on-line conseguem aprender de forma incremental, acessando o exemplo apenas uma vez. Árvores de decisão online geram um classificador através da transmissão de exemplos rotulados, com objetivo de igualar o desempenho de um algoritmo de árvore de decisão em lote, porém sem a necessidade de armazenar informações sobre todos os exemplos visitados (Rosset, 2015).

Os algoritmos que serão tratados por este estudo, têm sua origem em abordagens distintas, uns derivam do ID3 (ID5R e ITI) enquanto outros derivam da árvore de Hoeffding (VFDT e CVFDT). Assim surge a necessidade de avaliar as diferentes abordagens diante de cenários distintos. Estes cenários seriam compostos de bases de dados pequenas com poucos atributos, pequenas com muitos atributos, médias com poucos e muitos atributos e bases com um grande número de instâncias com poucos e muitos atributos.

Em (Boidol, 2015) é apresentado um estudo onde se realiza a comparação de algoritmos árvores de decisão incremental baseado no algoritmo VFDT, comparando três implementações de árvores distintas sobre um conjunto de 5 bases.

Assim o principal objetivo é comparar os seguintes algoritmos de árvores de decisão com aprendizado incremental: ID3, ITI, VFDT e CVFDT. As métricas utilizadas são a acurácia, precisão, sensibilidade e especificidade. Essas métricas serão avaliadas quando executadas sobre diferentes conjuntos de dados com características distintas. Assim espera-se conseguir identificar sobre quais condições cada algoritmo apresenta seus melhores resultados.

O seguinte estudo está organizado da seguinte maneira: A segunda seção trata de uma revisão bibliográfica composta pelos algoritmos de árvore de decisão online utilizados neste estudo; na terceira seção é apresentada a descrição do procedimento realizado para avaliação dos algoritmos, bem como as bases de dados utilizadas no experimento; a quarta seção traz resultados obtidos com o detalhamento das informações; por fim na quinta seção tem-se a conclusão deste estudo comparativo realizado neste trabalho.

2 Revisão Bibliográfica

2.1 Árvores de Decisão

As árvores de decisão nada mais são do que meios de representar resultados na forma de árvore, e elas constituem uma técnica bastante popular na realização da tarefa de classificação devido as seguintes características (Kothari, 2000):

- Velocidade de geração;
- Facilidade de aplicação;
- Facilidade de compreensão das decisões finais.

As gerações de árvores de decisão comumente utilizam a técnica de divisão-e-conquista (Quinlan, 1986). Para essa geração, um atributo é selecionado inicialmente, de modo que maximize alguma medida de predição, normalmente um critério de ramificação, que geralmente é indicado utilizando de fórmulas como entropia e ganho de informação. O ganho de informação é baseado no cálculo da entropia para cada atributo. Os cálculos da entropia e do ganho de informação são mostrados nas equações 1 e 2, respectivamente (Utgoff, 1989):

$$Ent(S) = \sum_{i=0}^{c} p_i log_2 p_i \qquad (1)$$

$$Ganho(S, A) = Ent(S) - \sum_{v \in Valores(A)} \frac{|S_v|}{|S|} Ent(S_v) \qquad (2)$$

Onde Ent indica a entropia, S indica o conjunto de exemplos, p indica cada instância do conjunto de exemplo, c o número de classes, A o atributo corrente e v o valor para cada possibilidade do atributo A.

Após a seleção do atributo, é definida em quantas partes será dividida a base de dados e então, são identificados em quais valores do domínio do atributo ocorrerá a divisão. Essa divisão é fundamentada por testes lógicos, sendo chamado de nó de decisão. O primeiro nó de decisão é um nó especial chamado de nó raiz da árvore. Para cada subconjunto de dados gerado pela raiz, o processo é repetido, recursivamente, até que algum critério de parada determine a interrupção do processo, formando os nós-folhas. Nas árvores de classificação, esses nós folhas possuem a informação das classes de um problema. Assim como o critério de ramificação, o critério de parada também pode ser definido segundo a vontade do criador da árvore. O mais comum é que se interrompa a indução quando não houver mais progresso (dentro de um limite aceitável) ao se criar uma nova ramificação ou quando um subconjunto a ser criado for formado por apenas um registro.

Um exemplo de árvore de decisão é demonstrado na figura 1, onde X1 e X2, são valores para atributos da base de dados que devem ser comparados com os valores 20 e 80, respectivamente, para descobrir para que galho (lado) da árvore a inferência deve seguir; C1, C2 e C3 são as classes que devem ser atribuídas as instâncias que alcançarem aquele nó-folha da árvore.

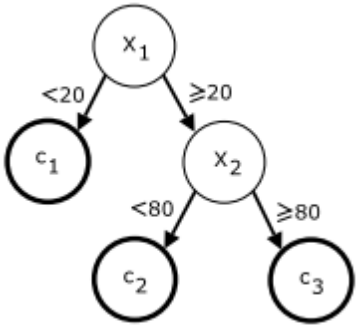

Figura 1. Exemplo de árvore de decisão

2.2 ID3 (Iterative Dichotomiser 3)

O algoritmo ID3 é o algoritmo mais conhecido e simples para aprendizado utilizando árvores de decisão (Quinlan, 1986). O ID3 utiliza a abordagem de busca gulosa top-down, tentando assim descobrir qual o melhor atributo, se baseando na pergunta: "Qual atributo deve ser testado no nó corrente da árvore?". Assim, o algoritmo funciona procurando o melhor atributo para cada posição na árvore, começando pela sua raiz. Basicamente, o ID3 busca o atributo que possui o melhor ganho de informação, para o conjunto de entrada, cria para cada valor desse atributo um nó filho e o processo se repete, mas agora olhando apenas para os exemplos que possuem valores de atributos correspondentes aos valores acima na árvore. A escolha do melhor atributo está diretamente ligada ao ganho de informação, conforme mencionado acima e o ganho de informação depende diretamente do cálculo da entropia. (Quinlan, 1986).

A seguir são apresentados dois algoritmos que derivam do ID3, porém através de modificações utilizam aprendizado incremental para a montagem da estrutura da árvore.

2.3 ID5R (An Incremental ID3)

O Algoritmo ID5R é a primeira implementação de aprendizado incremental baseada no algoritmo ID3 que apresenta um resultado realmente promissor (Kothari, 2000). A evolução proposta pelo ID5R em relação ao ID3 é que a nova implementação trabalha reestruturando a árvore de decisão, fazendo assim que o melhor atributo momentâneo esteja no nó raiz da árvore ou sub-árvore. (Utgoff, 1989).

Esta etapa de reestruturação é denominada *pull-up* e consiste em manipular a árvore de maneira que a mesma não perca a consistência já adquirida com as instâncias de formação, porém o *pull-up* exerce a função de trazer o atributo mais interessante para o nó raiz da árvore ou sub-árvore. Abaixo segue um esquema do funcionamento da etapa de reestruturação (Utgoff, 1989).

-Se o atributo a_{new} a ser promovido já se encontra na raiz da árvore ou sub-árvore em questão, então o processo pode ser finalizado;

-Caso contrário,

1: Deslocar de forma recursiva o atributo a_{new} para a raiz da sub-árvore logo acima. Converter qualquer sub-árvore não expandida, expandindo-a e definindo qual o melhor atributo através do ganho de informação para ser eleito para o nó decisão;

2: Transpor a árvore, resultando em uma nova árvore, onde o atributo a_{new} esteja na raiz e o antigo atributo que ocupava o nó raiz transforma-o em uma sub-árvore ligada diretamente ao nó raiz.

O grande diferencial do processo de reestruturação da árvore é que ele proporciona que os atributos que estão nos nós de decisão sejam analisados e se necessário atualizados sem a necessidade de se reexaminar as instâncias de formação. Durante a implementação recursiva verifica as sub-árvores, remontando-as quando for necessário, para que cada atributo de teste em um determinado nó apresente o maior ganho de informação (Kothari, 2000).

2.4 ITI (Incremental Decision Tree Induction)

O algoritmo ITI pode ser considerado como uma segunda versão do ID5R publicado oito anos após, com um diferencial em relação a implementação anterior. Essa diferença se justifica quando se insere um novo exemplo (Utgoff, 1997).

O comportamento ao se adicionar um novo exemplo segue as seguintes características: Se o exemplo for da mesma classe que o nó folha, o mesmo é simplesmente adicionado ao conjunto de exemplos classificados pelo nó folha. Se tiver um rótulo de classe diferente do rótulo do nó folha, o algoritmo atua transformando o nó folha em um nó de decisão, escolhendo o atributo que apresenta o melhor resultado para ser promovido ao teste do nó decisão. Após esse passo os exemplos que já estavam representados pelo nó que deixou de ser folha e se transformou em nó decisão são incorporados de forma recursiva sendo submetidos a um novo teste para que sejam classificados novamente de forma correta na nova estrutura (Utgoff, 1997).

Outra etapa efetuada pelo ITI é chamada de transposição recursiva, que consiste em promover ou rebaixar um determinado nó decisão. Durante a transposição recursiva, o algoritmo trabalha buscando o atributo que apresente o maior ganho de informação para que ele esteja no nó raiz da árvore ou sub-árvore. Assim a reestruturação provoca uma atualização em todos os nós decisão envolvidos na transposição, sendo necessário reavaliar os testes que estão sendo executados em cada um dos nós decisão abaixo e se necessário atualizá-los para que os testes sejam executados por atributos que apresentem um maior ganho de informação (Utgoff, 1997).

Existe ainda um marcador que é mantido em cada nó decisão da árvore que tem a função de indicar o quanto um teste em um nó decisão está defasado. Caso um teste seja considerado defasado, o ITI atua através da transposição recursiva identificando o teste que apresenta o maior ganho de informação e o promove recursivamente para o nó decisão, refletindo recursivamente a alteração para os ramos abaixo do nó alterado em questão (Utgoff, 1997).

O custo da atualização independe do número de instâncias de formação sobre a qual a árvore é baseada e a árvore resultante independe da ordem em que as instâncias são inseridas (Utgoff, 1997).

2.5 Árvore de Hoeffding

Árvores de Hoefdding projetam um modelo de árvore de decisão que trate de grandes conjuntos de dados. Este modelo deve exigir que cada exemplo seja lido no máximo uma vez, e apenas uma pequena constante de tempo para processá-lo. Isto torna possível construir árvores potencialmente complexas com custo computacional aceitável. Assim, dada uma corrente de exemplos, os primeiros serão utilizados para escolher o teste do nó raiz; Uma vez que o atributo da raiz é escolhido, os exemplos seguintes, vão ser passados para baixo, para as folhas correspondentes e usados para escolher dentre os atributos existentes o mais apropriado, e assim por diante de forma recursiva (Hulten, 2001).

Uma questão complexa é decidir exatamente quantos exemplos são necessários em cada nó. Para essa definição utiliza-se um cálculo estatístico conhecido como o limite de Hoeffding (Hoeffding, 1963).

O limite de Hoeffding é definido da seguinte forma: seja uma variável aleatória *r* em um domínio *R*. Após a realização de *n* observações independentes de *r* e calculamos sua média $\bar{r}$. Hoeffding nos diz que a probabilidade: $P(\bar{r} - \epsilon \leq \mu_r \leq \bar{r} + \epsilon) = 1 - \delta$ onde μ_r é a verdadeira média de *r*, $\epsilon = \sqrt{\frac{R^2 ln\left(\frac{2}{\delta}\right)}{2n}}$ e δ é um valor muito baixo.

A propriedade chave do algoritmo de árvore Hoeffding é que é possível garantir segundo previsões realistas que as árvores produzidas são assintoticamente próximas as oriundas por um modelo supervisionado (Hulten, 2001). Em outras palavras, a natureza incremental do algoritmo de árvore Hoeffding não afeta de forma significativa a qualidade das árvores que ela produz.

A seguir são apresentados dois algoritmos que se baseiam na Árvore de Hoeffding.

2.6 VFDT (Very Fast Decision Tree)

O algoritmo VFDT utiliza aprendizado incremental para a montagem de sua árvore a partir de dados de streaming utilizando tempo e memória constantes por exemplos de formação. A construção é baseada em uma árvore de que utiliza como um dos preceitos o tempo constante por exemplo avaliado (o pior caso é proporcional ao número de exemplos utilizados), enquanto apresenta quase a mesma estrutura de um aprendizado em lote (não incremental), se fornecidos exemplos suficientes (Domingos, 2000).

Apenas informações necessárias à manutenção da árvore são armazenadas em memória, assim os

exemplos podem ser processados mais rapidamente. Os primeiros exemplos que chegam ao fluxo de dados são necessários para escolher o atributo de teste no nó raiz; os subsequentes são utilizados através da parte induzida da árvore até que cheguem a um nó folha, sendo utilizados na escolha de uma possível divisão e criação de um nó de teste (decisão) e assim ocorre sucessivamente. Para determinar a quantidade de exemplos necessários para cada nó decisão o algoritmo VFDT utiliza um resultado estatístico conhecido como limites de Hoeffding (Domingos, 2000).

Os dados empíricos demonstram que os algoritmos de aprendizado não incrementais apresentam um melhor desempenho dentro do seu domínio: a sua precisão apresenta valores mais altos quando o número de exemplos *n* é pequeno, porém depois de alguns milhares de dezenas de exemplos, o algoritmo VFDT ultrapassa com uma boa margem os algoritmos de aprendizado não incrementais. Eles também provam que se *d* é o número máximo de atributos, *v* é o número máximo de valores para esses atributos, e *c* é o número de classes, o VFDT necessita de *O(dvc)* de memória por nó (Domingos, 2000).

Abaixo segue o esquema do algoritmo VFDT (Domingos, 2000):

1: Raiz: Exs = N primeiros exemplos de treino, onde N e obtido pelo limitante de Hoeffding.

2: Baseado no conjunto de exemplos Exs, selecione o melhor atributo A como teste do no.

3: Crie um ramo para cada valor v_i do atributo discreto A, correspondendo ao teste A = vi.

4: Descarte os exemplos que estavam antes no nó. Separe os próximos exemplos entre os filhos do nó corrente, de acordo com o valor de A, criando um conjunto de exemplos Exs para cada filho. O número de exemplos de cada filho será também determinado pelo limitante de Hoeffding.

5: Repita o passo 2 para cada filho.

2.7 CVFDT (Concept-adapting Very Fast Decision Tree)

Uma das primeiras e mais relevantes falhas do algoritmo VFDT era a suposição do conceito de estagnação ao longo do tempo, ou seja, acreditar que o comportamento dos exemplos irá seguir alguns padrões (conceitos), sem apresentar mudanças com o passar do tempo.

Diante dessa questão o CVFDT considera que se gera uma série de conceitos, a partir dos dados informados, ou por uma função de conceito com parâmetros variáveis no tempo, portanto, implementa uma janela deslizante de exemplos de base para a construção e modificação da árvore com o objetivo de manter sempre atualizada a sua estrutura. Essa janela móvel deve ser suficientemente menor que a taxa de derivação do conceito (que indica o quanto um conceito está defasado) mais suficientemente grande para aprender totalmente o conceito em questão (Hulten, 2001).

O processo de atualização para um exemplo é bem semelhante ao usado pelo VFDT: Incrementar a contagem correspondente ao novo exemplo, diminuindo a contagem do exemplo mais antigo dentro da janela móvel (exemplo agora que deverá ser ignorado). Assim ocorre a mudança no conceito, algumas divisões que antes passavam no teste de Hoeffding agora não passarão mais, devido a um atributo alternativo apresentar um maior ganho. Neste caso o CVFDT começa a desenvolver uma nova possibilidade para substituir o teste considerado obsoleto. O algoritmo CVFDT realiza uma verificação periódica nos nós internos buscando aqueles onde o atributo de teste escolhido pode estar obsoleto. Assim os próximos exemplos são utilizados para comparar os nós que estão com os testes defasados com os nós candidatos a substituí-los. Assim as sub-árvores que não foram demonstradas melhora, são podadas. A escolha de um novo atributo de teste é semelhante à escolha de um atributo para o teste no nó raiz, porém com critérios mais rigorosos para evitar o crescimento da árvore de forma excessiva (Hulten, 2001).

O parâmetro que define o tamanho da janela de tempo *w* não se adequa a todos os conceitos, por esse fator ele deve ser dinâmico. Por exemplo, pode fazer sentido escolher um determinado valor para *w* quando muitos nós na árvore de Hoeffding tornam-se questionáveis de uma só vez, ou em decorrência de uma drástica mudança na característica dos exemplos, uma vez que estes eventos podem indicar uma mudança circunstancial de conceito. De maneira semelhante algumas aplicações podem ser beneficiadas por um aumento no tamanho da janela *w* quando se tem poucos nós questionáveis, pois isso indica que o conceito está estável (Hulten, 2001).

3 Experimento

O experimento executado neste estudo consiste em avaliar um conjunto composto por nove bases de dados a serem classificadas pelos algoritmos (ID5R, ITI, VFDT e CVFDT) apresentados acima.

As bases de dados utilizadas neste experimento são difundidas na literatura e estão disponíveis através do repositório UCI (Lichman, 2013). As bases utilizadas estão caracterizadas abaixo.

Tabela 1. Informações sobre as bases utilizadas.

Base	Número de Instâncias	Número de Atributos	Número de Classes
Car Evaluation	1728	6	4
Connect-4	67557	42	3
MONK's Problem	432	7	2
Mushroom	8124	22	2
Nursery	12960	8	5
Molecular Biology	106	58	2
Tic-Tac-Toe Endgame	958	9	2

Congressional Voting Records	435	16	2
Zoo	101	17	7

Os dados utilizados englobam bases pequenas e com poucos atributos, bases pequenas com muitos atributos, bases grandes com poucos atributos e bases grandes com muitos atributos. Dentre o conjunto apresentado, destaca-se a connect-4 com um número considerável de atributos (42) e um número grande de instancias (67557).

Foram avaliados a acurácia, precisão, sensibilidade e especificidade. A acurácia é o percentual de classificações corretas sobre todo o conjunto. Precisão trata sobre a porcentagem de exemplos classificados como uma determinada classe e que realmente pertencem a esta classe. Sensibilidade diz respeito aos exemplos de uma determinada classe e que foram classificados corretamente. Por fim a especificidade trata dos exemplos que não pertencem a classe em questão e que não foram classificados como da classe em questão (Powers, 2011).

No experimento foi utilizado o método de validação cruzada (Kohavi, 1995) com o número de subconjuntos definidos em cinco. Em resumo a técnica de validação cruzada consiste no particionamento do conjunto de dados em subconjuntos mutualmente exclusivos, utiliza-se alguns desses subconjuntos para a estimação do modelo e o restante para a validação do mesmo (Kohavi, 1995).

4 Resultados

As avaliações foram separadas de acordo com a métrica a ser avaliada. Abaixo são apresentadas quatro tabelas com os resultados obtidos de acordo com as métricas: acurácia, precisão, sensibilidade e especificidade.

Tabela 2 – Resultado Acurácia

Base	Acurácia			
	ID5R	ITI	VFDT	CVFDT
Car	0,9557	0,9852	0,9823	0,9823
Connect-4	0,7397	0,7705	0,9155	0,9083
Monks	0,9018	0,9191	0,9883	0,9883
Mushroom	0,9958	1,0000	1,0000	1,0000
Nursery	0,9776	0,9975	0,9925	0,9781
Molecular	0,6738	0,6938	0,7500	0,7500
Tictactoe	0,9472	0,9748	0,9947	0,9947
Voting	0,9850	1,0000	1,0000	1,0000
Zoo	0,9577	0,9720	0,9720	0,9720

Tabela 3 – Resultado Precisão

Base	Precisão			
	ID5R	ITI	VFDT	CVFDT
Car	0,9508	0,9802	0,9788	0,9788
Connect-4	0,4920	0,5125	0,7809	0,7328
Monks	0,8929	0,9194	0,9886	0,9886
Mushroom	0,9929	1,0000	1,0000	1,0000
Nursery	0,7690	0,7807	0,7769	0,7547
Molecular	0,6928	0,7125	0,7500	0,7500
Tictactoe	0,9531	0,9761	0,9960	0,9960
Voting	0,9850	1,0000	1,0000	1,0000
Zoo	0,9333	0,9857	0,5476	0,5476

Tabela 4 – Resultado Sensibilidade

Base	Sensibilidade			
	ID5R	ITI	VFDT	CVFDT
Car	0,8579	0,8844	0,8818	0,8818
Connect-4	0,8630	0,8806	0,9061	0,8869
Monks	0,8969	0,9290	0,9884	0,9884
Mushroom	0,9990	1,0000	1,0000	1,0000
Nursery	0,9538	0,9657	0,7679	0,7111
Molecular	0,6894	0,7035	0,7525	0,7525
Tictactoe	0,9549	0,9745	0,9923	0,9923
Voting	0,9800	1,0000	1,0000	1,0000
Zoo	0,9427	0,9857	0,5714	0,5714

Tabela 5 – Resultado Especificidade

Base	Especificidade			
	ID5R	ITI	VFDT	CVFDT
Car	0,9514	0,9808	0,9794	0,9794
Connect-4	0,6899	0,6969	0,8529	0,8627
Monks	0,9393	0,9488	0,9884	0,9884
Mushroom	0,9926	1,0000	1,0000	1,0000
Nursery	0,9789	0,9977	0,9946	0,9830
Molecular	0,6960	0,7111	0,7525	0,7525
Tictactoe	0,9513	0,9725	0,9923	0,9923
Voting	0,9825	1,0000	1,0000	1,0000
Zoo	0,9757	0,9857	0,9857	0,9857

Ao observar as métricas, fica destacado que o algoritmo VFDT apresenta em geral os melhores resultados e quando isso não ocorre a diferença para o melhor resultado no geral é pequena.

Outro ponto de destaque ocorre quando se observa os resultados atrelados a base Connect-4. Devido a o número elevado de instâncias ela apresenta uma discrepância maior quando analisamos as métricas, tal discrepância só não ocorre quando a métrica analisada é a sensibilidade. Assim apenas os algoritmos VFDT e CVFDT apresentaram resultados considerados promissores para esta base nas demais métricas analisadas.

5 Conclusão

Para a maioria dos problemas apresentados neste trabalho, os algoritmos ITI, VFDT e CVFDT apresentaram resultados próximos, ocorrendo uma alternância sobre qual apresentava o melhor resultado.

Realizando uma análise mais detalhada, ao se tratar de problemas com um número pequeno de instâncias (máximo 1000) os algoritmos ITI, VFDT e CVFDT apresentaram os melhores resultados. Ao se analisar problemas com número de instâncias até quinze mil instâncias o algoritmo ITI apresenta o melhor desempenho dentre os concorrentes. Finalmente ao se analisar um problema com mais de sessenta mil instâncias (Connect-4), no geral o algoritmo VFDT apresentou o melhor resultado, seguido de perto pelos resultados obtidos pelo CVFDT. Quando comparado aos outros ocorre uma grande discrepância nos resultados, tal discrepância é justificada pelo fato dos algoritmos baseados na árvore de Hoeffding estarem preparados para trabalhar de forma mais eficiente com um grande volume de dados.

Ao se analisar a diferença entre o VFDT e CVFDT observa-se que para o problema abordado, a janela de tempo conseguiu identificar de maneira significativa os conceitos importantes, otimizando assim o desempenho do CVFDT. Devido ao fato do VFDT trabalhar mantendo informações referentes a todas as instâncias analisadas, atingiu os melhores resultados para este problema.

Assim pode se concluir que em linhas gerais o algoritmo VFDT apresentou os melhores resultados. Ele também mostrou sua robustez ao conseguir se destacar na base Connect-4. O CVFDT e o ITI também apresentaram resultados interessantes, com destaque para o ITI trabalhando com um número maior de classes, como ocorre na base ZOO. Por fim o ID5R apresentou alguns bons resultados, mais em nenhuma base ele conseguiu sobrepor o ITI.

Agradecimentos

O presente trabalho foi realizado com o apoio financeiro da CAPES – Brasil.

Referências Bibliográficas

Boidol, J; Hapfelmeier, A. and Tresp, V. Probabilistic Hoeffding Trees. In: Industrial Conference on Data Mining. Springer International Publishing, 2015. p. 94-108.

Domingos, P. and Hulten, G. Mining high-speed data streams. Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2000.

Hamroun, M. and Gouider, M. S. (2015). Possibilistic Very Fast Decision Tree for Uncertain Data Streams. In Intelligent Decision Technologies (pp. 195-207). Springer International Publishing.

Hoeffding, W (1963). Probability inequalities for sums of bounded random variables. Journal of the American statistical association, 58(301), 13-30.

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

Kohavi, R (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In Ijcai (Vol. 14, No. 2, pp. 1137-1145).

Kothari, R. and Dong, M (2000). Decision trees for classification: A review and some new results. Pattern Recognition: From Classical to Modern Approaches, edited by SK Pal and A. Pal (World Scientific, Singapore, 2001), 169-184.

Lichman, M (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Powers, D. M (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.

Quinlan, J. R (1986). Induction of decision trees. Machine learning, 1(1), 81-106.

Quinlan, J. R (2014). C4. 5: programs for machine learning. Elsevier.

Rosset, C (2015). A Review of Online Decision Tree Learning Algorithms.

Utgoff, P. E (1989). Incremental induction of decision trees. Machine learning, 4(2), 161-186.

Utgoff, P. E; Berkman, N. C. and Clouse, J. A. (1997). Decision tree induction based on efficient tree restructuring. Machine Learning, 29(1), 5-44.