HTILDE-RT: Um algoritmo para aprender Árvores de Regressão Relacionais em grandes conjuntos de dados

Glauber M. C. Menezes [1], **Gerson Zaverucha** [1]

[1] Programa de Engenharia de Sistemas de Computação (PESC)
Universidade Federal do Rio de Janeiro (COPPE/UFRJ)
Caixa Postal 68.511 – 21.945-970 – Rio de Janeiro – RJ – Brasil

{glauber,gerson}@cos.ufrj.br

Abstract. *Currently, modern organizations store their data under the form of relational databases which grow faster than hardware capacities. However, extracting information from such databases has become crucial. In this work we propose HTILDE-RT, an algorithm to learn relational regression trees efficiently from huge databases. It is based on the ILP system TILDE and the propositional system VFDT learner. The algorithm uses Hoeffding bound to scale up the learning process. We compared HTILDE-RT with TILDE-RT in two large datasets, each with two million examples, yielding more than three times faster learning times with no statistically significant difference in Pearson correlation coefficient.*

Resumo. *Atualmente, organizações modernas armazenam seus dados sob a forma de bancos de dados relacionais que crescem numa velocidade superior à capacidade de hardware. Entretanto, extrair informações destas bases tornou-se crucial. Neste trabalho propomos o HTILDE-RT, um algoritmo capaz de aprender árvores de regressão relacionais em grandes massas de dados. O algoritmo é baseado no sistema ILP TILDE e no sistema proposicional VFDT. O algoritmo usa o limitante de Hoeffding para acelerar o aprendizado. Comparamos o HTILDE-RT com o TILDE-RT em dois conjuntos de dados grandes, com dois milhões de exemplos cada, obtendo tempos de aprendizado mais de três vezes mais rápidos sem diferenças significativas no coeficiente Pearson.*

1. Introdução

Mineração em fluxos de dados tornou-se objeto de pesquisa em variados campos da Ciência da Computação tais como Aprendizado de Máquina e Mineração de Dados. Como exemplos de onde técnicas para fluxos são úteis podemos citar: aplicações de suporte a transações de cartões de crédito, tráfego de Internet, monitoramento em telecomunicações e pesquisa científica em dados biológicos.

Os bancos dados de organizações modernas cresceram numa escala superior aos recursos de hardware e memória, entretanto, extrair informações ocultas em tais bancos tornou-se tarefa crítica para competitividade e sobrevivência das corporações.

Atualmente, as bases de dados são estruturadas sob o paradigma relacional. Porém, algoritmos de aprendizagem tradicionais não são perfeitamente aplicáveis em tal

tipo de dados, uma vez que utilizam o paradigma proposicional, não sendo capazes de descrever relações entre objetos nos dados. Uma outra importante questão é o fato de que os algoritmos tradicionais não são capazes de manipular massas de dados tão grandes, pois necessitam que todos os exemplos estejam na memória principal ao mesmo tempo, o que é impraticável neste cenário.

Em tal ambiente dados novos chegam em taxas elevadas e um algoritmo eficiente deve processá-los prontamente. Porém, há a restrição de que armazená-los como um todo na memória é irrealizável. Dado isto, as áreas de Aprendizado de Máquina e Mineração de Dados direcionaram esforços para tratar tal tipo de demanda de forma eficiente.

Domingos e Hulten [Domingos and Hulten 2000] desenvolveram uma metodologia para tratar fluxos de dados, que culminou no algoritmo VFDT (Very Fast Decision Trees). O VFDT é um algoritmo proposicional, que usa uma técnica de amostragem baseada no limitante de Hoeffding [Hoeffding 1963] para escolher qual é o melhor teste a ser introduzido no nó de uma árvore de decisão, sem a necessidade de analisar todos os exemplos ao mesmo tempo. O VFDT é o estado da arte para este tipo de problema.

O TILDE (Top-down Induction of Logical Decision Trees) [Blockeel and Raedt 1998] é a versão relacional do algoritmo de árvores de decisão, o qual é capaz de modelar a grande maioria dos esquemas relacionais nos bancos de dados modernos, sua eficiência é garantida através do emprego das técnicas de pacotes de cláusulas (Query Packs) [Blockeel et al. 2002] e Aprendizado por Interpretações (Learning from Interpretations) [Blockeel et al. 1999]. Tal algoritmo é implementado como parte do sistema de Programação em Lógica Indutiva (ILP) ACE [H. Blockeel and Fierens 2005].

Para obter a versão relacional do VFDT, Lopes e Zaverucha [Lopes and Zaverucha 2009] modificaram o TILDE, desenvolvendo o algoritmo HTILDE (Hoeffding TILDE), baseado no TILDE e no VFDT. Então, o HTILDE é a versão relacional do VFDT para problemas de classificação.

Neste trabalho apresentamos o HTILDE-RT, que é uma modificação do HTILDE para árvores de regressão, que combina a expressividade do paradigma relacional e a eficiência da técnica de amostragem apoiada no limitante de Hoeffding. Uma árvore de regressão é um algoritmo capaz de aprender funções numéricas mantendo a interpretabilidade e eficiência da versão original.

2. Conhecimentos Preliminares

Nesta seção explicaremos rapidamente o VFDT, o TILDE e o HTILDE, sendo todos base do HTILDE-RT, bem como outros trabalhos relacionados ao tema de regressão.

2.1. Limitante de Hoeffding

Seja r uma variável aleatória real cujo domínio tem tamanho R. Considere n observações independentes desta variável e seja $\overline{r}$ sua média computada. O limitante de Hoeffding [Hoeffding 1963] atesta que $P(\mu_r \geq \overline{r} - \epsilon) = 1 - \delta$, onde μ_r é a verdadeira média da variável r, ϵ é dado pela equação 1 e δ é um número muito pequeno.

$$\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}} \tag{1}$$

2.2. Modelagem para Árvores de Decisão

A seguinte modelagem foi proposta por Domingos e Hulten [Domingos and Hulten 2000]. Seja G a heurística a ser maximizada, sejam X_a e X_b os atributos com os maiores valores para G. Seja $\Delta G_r = G(X_a) - G(X_b)$ a diferença nos valores das heurísticas dos atributos quando o algoritmo utiliza todo o conjunto de dados, e seja $\Delta\overline{G} = \overline{G}(X_a) - \overline{G}(X_b)$ a diferença observada quando apenas n exemplos chegaram a um nó.

Então, dado um valor para δ através do limitante de Hoeffding, sabemos que com probabilidade $1-\delta$ que $\Delta G_r \geq \Delta\overline{G} - \epsilon$, onde ϵ é dado pela equação 1. Se $\Delta\overline{G} > \epsilon$, então podemos dizer que com mesma probabilidade $\Delta G_r > 0$, logo $G(X_a) > G(X_b)$, i.e., com probabilidade $1-\delta$, X_a é o melhor atributo.

Em termos práticos, para podermos afirmar se X_a é a melhor escolha, precisamos verificar se $\Delta\overline{G} \geq \epsilon$. Então, o que precisamos fazer é processar exemplos até acontecer que $\Delta\overline{G} \geq \epsilon$, para o valor escolhido de δ e o número corrente de exemplos.

Destacamos que o limitante de Hoeffding independe da distribuição da variável considerada. Esta independência vem com o custo de ser necessário processar mais exemplos do que em métodos que usam limitantes específicos. Porém, dado o cenário das grandes massas de dados, isto não configura um problema. Além disso, para o ganho de informação temos $0 \leq G \leq lg(|C|)$, onde C é o conjunto das possíveis classes.

2.3. VFDT

Utilizando tal modelagem, Domingos e Hulten desenvolveram o algoritmo VFDT (Very Fast Decision Tree) [Domingos and Hulten 2000], um algoritmo de aprendizado proposicional e escalável em grandes bases de dados. Os autores provaram que o VFDT gera árvores assintoticamente próximas às geradas pelo algoritmo tradicional.

O VFDT tem alguns elementos para melhorar ainda mais sua performance. Iremos mencionar dois deles, sendo estes um parâmetro para acúmulo de exemplos nas folhas $emin$ e um parâmetro para desempate entre atributos τ. Mais informações sobre o VFDT podem ser encontradas em [Domingos and Hulten 2000].

O cômputo da heurística é um processo custoso. O parâmetro $emin$ reduz o tempo de indução, permitindo que o usuário determine que cada novo cômputo da heurística seja feito somente quando $emin$ novos exemplos chegaram desde a folha vazia ou o último cômputo. Isto melhora a performance pois o algoritmo não precisa computar a heurística para todos os possíveis refinamentos todas as vezes que um único exemplo chega à folha. Portanto, no contexto de conjuntos de dados gigantes, isto tem um grande impacto.

Quando dois ou mais atributos possuem valores de heurísticas muito próximos, o algoritmo pode necessitar de um número grande de exemplos até escolher o melhor atributo, o que é um desperdício, uma vez que não fará grande diferença qual atributo seja escolhido já que estão muito próximos. Esta situação é chamada de empate e para resolvê-la o VFDT usa um parâmetro fornecido pelo usuário que impõe um limiar de desempate. Caso $\Delta\overline{G} < \epsilon < \tau$, o melhor atributo é escolhido como o teste para o nó.

Terminemos esta subseção mostrando as diferenças entre o algoritmo tradicional de de árvores de decisão [Mitchell 1997] e o VFDT para comparação.

Algoritmo 1 Algoritmo de Árvore de Decisão [Mitchell 1997]

1: Raiz: $Exs =$ todos os exemplos de treino.
2: Baseado no conjunto de exemplos Exs, selecione o melhor atributo A como teste do nó.
3: Crie um ramo para cada valor v_i do atributo discreto A, correspondendo ao teste $A = v_i$.
4: Separe os exemplos entre os nós filhos do nó corrente de acordo com o valor de A, criando um novo conjunto de exemplos Exs para cada filho.
5: Repita o passo 2 para cada filho.

Algoritmo 2 VFDT [Domingos and Hulten 2000]

1: *Raiz: $Exs = N$ primeiros exemplos de treino, onde N é obtido pelo limitante de Hoeffding ($\Delta\overline{G} > \epsilon$ ou $\Delta\overline{G} < \epsilon < \tau$).*
2: Baseado no conjunto de exemplos Exs, selecione o melhor atributo A como teste do nó.
3: Crie um ramo para cada valor v_i do atributo discreto A, correspondendo ao teste $A = v_i$.
4: *Descarte os exemplos que estavam antes no nó. Separe os próximos exemplos entre os filhos do no 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.

2.4. TILDE

TILDE (Top-down Induction of Logical Decision Trees) é a versão relacional para aprendizado de árvores de decisão, isto implica que os nós internos são representados por conjunções de literais de Lógica de Primeira Ordem em vez de atributos. Por conseqüência, os modelos gerados têm a estrutura de uma árvore binária, tal que cada nó interno possui dois ramos, sendo o esquerdo um caminho para os exemplos que sucedem no teste do nó e o direito o caminho para os exemplos que falham no teste lógico do nó. O TILDE é capaz de rodar nos modos de classificação e regressão (TILDE-RT).

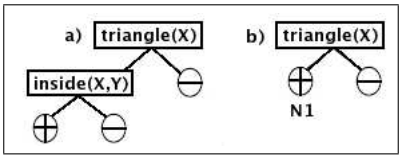

Figura 1. Exemplos de árvore gerada pelo TILDE.

Os candidatos a testes são chamados de refinamentos e são especificados pelo usuário através do viés de linguagem do problema. Quando os valores da heurística são computados o TILDE considera o caminho associado à consulta entre a raiz e a folha em avaliação.

Por exemplo, considere a árvore b mostrada na figura 1. Suponha que desejamos saber quais testes podem ser colocados no nó $N1$. Uma vez que este nó é o filho

esquerdo da raiz, sabemos que triangle(X) é verdade em $N1$ e portanto todos os refinamentos conterão este literal. Aplicando os demais refinamentos a este nó, de acordo com o viés de linguagem previamente especificado, podemos obter quatro refinamentos para $N1$: $triangle(X), inside(X, Y)$; $triangle(X), inside(Y, X)$; $triangle(X), square(Y)$ e $triangle(X), circle(Y)$. Se o melhor refinamento for o primeiro então o teste colocado no nó $N1$ será $inside(X, Y)$.

É importante notar que tais refinamentos são processados sob a condição de θ-subsumption [Blockeel et al. 1999]. Também são consideradas restrições sobre o espaço de busca escolhidas pelo usuário, que podem ser estabelecidas, principalmente, pelo uso dos tipos e modos das variáveis. Os tipos definem o domínio das variáveis restringindo a unificação. Os modos indicam se uma variável é de entrada, saída ou os dois. Uma referência mais completa sobre as configurações dos refinamentos e viés de linguagem pode ser encontrada em [H. Blockeel and Fierens 2005].

Para alcançar eficiência, o TILDE usa pacotes de cláusulas e aprendizado por interpretações para reduzir o custo computacional da prova dos exemplos. Estes mecanismos são importantes em função de o TILDE aplicar os testes aos exemplos através de provas lógicas, evitando-se provas de literais de forma repetitiva e acelerando o mecanismo de prova.

O TILDE assume que cada exemplo é uma interpretação [Blockeel et al. 1999]. Cada exemplo possui um rótulo de sua classe e um conjunto de fatos, os quais codificam propriedades e relações que podem ocorrer, de tal maneira que cada exemplo pode ser processado de forma independente dos demais. O conhecimento sobre o domínio (background knowledge) é também fornecido, através do qual fatos adicionais podem ser deduzidos, sendo representado sob a forma de um programa Prolog. O Aprendizado por Interpretações tem menor poder de expressividade que o Aprendizado por Implicação Lógica [Muggleton 1991], o qual é tradicional em ILP, porém é mais eficiente. Apesar disso, para a maioria das aplicações o aprendizado por interpretações é suficiente, sendo a total expressividade tradicional da ILP nem sempre necessária [Blockeel et al. 1999].

Um pacote de cláusulas (Query Pack) [Blockeel et al. 2002] é um método de representar cláusulas sob a forma de disjunções a fim de evitarem-se provas idênticas de literais. Cláusulas de mesma cabeça têm suas caudas reunidas sob a forma de uma disjunção, de tal modo que literais comuns a clausulas diferentes são devidamente fatorados e evidenciados. Isto é importante no momento em que o TILDE computa a heurística para cada refinamento, onde é comum que dois ou mais refinamentos tenham literais em comum.

O algoritmo 3 destaca as diferenças entre o TILDE e o algoritmo proposicional tradicional.

2.5. HTILDE

O HTILDE (Hoeffding TILDE) [Lopes and Zaverucha 2009] é um algoritmo baseado no TILDE e no VFDT, aliando expressividade e eficiência através da combinação do paradigma relacional e a metodologia proposta por Domingos e Hulten no VFDT. É um algoritmo ILP escalável que se propõe a tratar grandes bases de dados relacionais.

HTILDE processa os refinamentos da mesma forma que o TILDE e possui três parâmetros herdados do VFDT: δ, $emin$, e τ. Desta forma, o HTILDE é um algoritmo

Algoritmo 3 TILDE [Blockeel and Raedt 1998]

1: Raiz: $Exs =$ todos os exemplos de treino.
2: *Baseado no conjunto de exemplos Exs, selecione o melhor refinamento R como teste para o nó.*
3: *Crie dois ramos: um correspondente ao teste $R = True$ e outro para $R = False$.*
4: *Separe os exemplos entre os nós filhos do nó corrente de acordo com o valor lógico obtido no teste de R, criando um novo conjunto de exemplos Exs para cada filho.*
5: Repita o passo 2 para cada filho.

incremental que aprende árvores de decisão relacionais, usando o limitante de Hoeffding para tratar grandes massas de dados.

O algoritmo 4 destaca as diferenças entre o HTILDE e o VFDT. Destacamos que o HTILDE usa os mesmos parâmetros δ, $emin$, e τ que o VFDT, bem como processa os refinamentos da mesma forma que o TILDE.

Algoritmo 4 HTILDE [Lopes and Zaverucha 2009]

1: Raiz: $Exs = N$ primeiros exemplos de treino, onde N é obtido pelo limitante de Hoeffding ($\Delta\overline{G} > \epsilon \; ou \; \Delta\overline{G} < \epsilon < \tau$).
2: *Baseado no conjunto de exemplos Exs, selecione o melhor refinamento R como teste para o nó.*
3: *Crie dois ramos: um correspondente ao teste $R = True$ e outro para $R = False$.*
4: *Descarte os exemplos que estavam antes no nó. Separe os próximos exemplos entre os dois filhos do no corrente, de acordo com o valor lógico obtido no teste de R, 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.

2.6. Árvores de Regressão

Uma árvore de regressão é uma versão do algoritmo 1 que trata do aprendizado de funções numéricas. Suas principais diferenças são a função de heurística e a predição. Árvores de classificação usam heurísticas aplicáveis a domínios discretos como o ganho de informação e o índice de Gini, porém no contexto da regressão a heurística mais comumente usada é redução na variância do atributo alvo. Além disso, árvores de classificação predizem a classe majoritária da folha em que um exemplo a ser classificado alcançar e as árvores de regressão tradicionalmente predizem o valor médio observado do atributo alvo dentre os exemplos de uma folha.

A heurística padrão usada é a redução da variância [Breiman et al. 1984], dada por:

$$\phi(T, l) = S^2(L) - \sum_{T_j} \frac{|T_j|}{|L|} S^2(T_j) \tag{2}$$

onde L é o conjunto dos exemplos da folha l, T é um atributo de teste em l e T_j é o conjunto dos exemplos de l tais que o atributo T tem valor j. Se T for contínuo, tradici-

onalmente usa-se um passo de discretização para determinar os intervalos em que j deve iterar. A variância S^2 para um conjunto de exemplos A é dada por:

$$S^2(A) = \frac{1}{|A|} \sum_{i=1}^{|A|} (a_i - \overline{a})^2 \tag{3}$$

onde a_i é o valor do atributo alvo no exemplo i e $\overline{a}$ é o valor médio do predicato alvo observado nos exemplos de A.

O algoritmo FIMT-DD (Fast Incremental Model Trees with Drift Detection), proposto em [Ikonomovska et al. 2011], é uma versão proposicional para o problema de regressão baseada no VFDT e CVFDT. O algoritmo trata atributos numéricos através de árvores binarias de busca estendidas (Extended Binary Search Tree - E-BST) em vez de discretização. Sua heurística é o desvio padrão e o modelo de regressão armazenado nas folhas são perceptrons e o algoritmo é também capaz de tratar mudanças de distribuição [Hulten et al. 2001]. Mais informações sobre o sistema proposicional FIMT-DD podem ser encontradas em [Ikonomovska et al. 2011].

O algoritmo TILDE-RT [Blockeel and Raedt 1998] é a versão do TILDE para tratar problemas de regressão. O TILDE-RT usa como heurística a soma dos erros quadráticos em vez da variância, tal heurística é também listada em [Breiman et al. 1984]. Conforme o algoritmo tradicional de árvores de regressão, o modelo preditivo armazenado nas folhas é a média dos valores observados para o atributo alvo nos exemplos. A heurística do TILDE-RT será definida na seção 3.

Devido à limitação de espaço deste trabalho e ao vasto campo da regressão, recomenda-se a inspeção dos trabalhos [Breiman et al. 1984],[Vens et al. 2006] e [Ikonomovska et al. 2011] para maiores esclarecimentos sobre o tema, bem como as referências relacionadas ao tema que são citadas pelos dois últimos trabalhos.

3. HTILDE-RT

HTILDE-RT é uma modificação do HTILDE para tratar problemas de regressão e por isso processa exemplos da mesma forma. O TILDE-RT, usa como heurística a redução dos erros quadráticos [H. Blockeel and Fierens 2005], conforme equação 4, portanto nossa escolha para heurística é muito semelhante, com a alteração de ocorrer normalização para que seus valores fiquem delimitados.

Logo, desejamos o refinamento que cause a maior redução possível no erro quadrático. Além disso, o limitante de Hoeffding necessita que a variável observada seja limitada e, para tal, basta uma normalização com o fator $SS(L)$. Formalmente, desejamos o refinamento ρ que maximiza H na equação 5:

$$H(\rho, l) = SS(L) - (SS(\boldsymbol{N}(\rho)) + SS(\boldsymbol{P}(\rho))) \tag{4}$$

$$H(\rho, l) = \frac{SS(L) - (SS(\boldsymbol{N}(\rho)) + SS(\boldsymbol{P}(\rho))}{SS(L)} \tag{5}$$

onde $SS(L)$ é a soma dos erros quadráticos do predicado alvo para o conjunto de exemplos L da folha l, ρ um candidato a refinamento de l, $P(\rho)$ o subconjunto de exemplos de L que passam no teste de ρ e $N(\rho)$ o subconjunto de exemplos de L que falham em ρ. A soma dos erros quadráticos do predicado alvo num conjunto de exemplos A qualquer é dada por:

$$SS(A) = \sum_{i=1}^{|A|}(a_i - \overline{a})^2 \tag{6}$$

Note que, H na equação 5 fica limitado inferiormente por 0 e superiormente por 1, logo está devidamente delimitada, então podemos usar $R = 1$ na equação do limitante de Hoeffding 1 resultando na equação 7.

$$\epsilon = \sqrt{\frac{ln(1/\delta)}{2n}} \tag{7}$$

Sejam ρ_1 e ρ_2 os dois melhores refinamentos com heurísticas devidamente normalizadas, então a variável aleatória para o limitante de Hoeffding é $\Delta\overline{H} = \overline{H}(\rho_1) - \overline{H}(\rho_2)$, onde $\Delta\overline{H}$ é a diferença observada entre os exemplos correntes na folha, de forma análoga a $\Delta\overline{G}$. No algoritmo 5 destacamos a diferença no algoritmo do HTILDE-RT.

Algoritmo 5 HTILDE-RT

1: Raiz: $Exs = N$ primeiros exemplos de treino, onde N é obtido pelo limitante de Hoeffding ($\Delta\overline{H} > \epsilon$ *ou* $\Delta\overline{H} < \epsilon < \tau$).
2: *Baseado no conjunto de exemplos Exs, selecione o refinamento R, entre os demais, cujo valor* ***normalizado*** *da heurística seja máximo, como teste para o nó.*
3: Crie dois ramos: um correspondente ao teste $R = True$ e outro para $R = False$.
4: Descarte os exemplos que estavam antes no nó. Separe os próximos exemplos entre os dois filhos do no corrente, de acordo com o valor lógico obtido no teste de R, 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.

Algumas diferenças entre os algoritmos 4 e 5 e devem ser reforçadas. O HTILDE utiliza o ganho de informação como heurística, a qual está sempre entre 0 e $lg(|C|)$, em nosso caso necessitamos do fator de normalização $SS(T(\rho))$ para que as fronteiras dos valores da heurística estejam determinadas. Além disso, o modelo preditivo é a média dos exemplos em relação ao predicado alvo em vez da classe majoritária.

4. Experimentos

Conforme mencionado em [Vens et al. 2006], há uma escassez de conjuntos de dados relacionais para regressão que sejam públicos, portanto conduzimos experimentos com dados sintéticos descritos em [Vens et al. 2006]. Usamos geradores para obter os nossos conjuntos de dados, Artificial 1 e Artificial 2, ambos com dois milhões de exemplos e tamanhos médios de 340Mb e 590Mb respectivamente.

Em ambos experimentos usamos os parâmetros do HTILDE-RT com $emin = 300$, $\tau = 0,1$ e $\delta = 10^{-6}$. Estes valores são os mesmos que os default do VFDT com exceção de τ, escolhidos por apresentaram melhores resultados em testes preliminares, porém ainda nos falta fazer o ajuste destes parâmetros. Mantivemos os parâmetros do TILDE-RT com valores default.

Avaliamos os algoritmos adicionando ruído gaussiano nos dados, conforme feito em [Vens et al. 2006]. Cada exemplo teve um valor de ruído adicionado a seu predicato alvo. Os valores de ruído foram gerados aleatoriamente com distribuição normal.

Usamos o TILDE-RT como comparativo, uma vez que ambos algoritmos rodam sob o mesmo sistema e ambos são relacionais. Marcamos os resultados referentes ao ruído com um asterisco: HTILDE-RT* e TILDE-RT*.

As medidas de interesse avaliadas são: Tempo de aprendizado (Tempo), tamanho das teorias ou número de nós na árvore (Nós), número total de literais (Literais), coeficiente de correlação de Pearson (r na equação 8) entre os valores reais e os preditos do predicado alvo, erro quadrático médio (MSE) e erro absoluto médio (MAE), equações 9 e 10, conforme [H. Blockeel and Fierens 2005].

$$r = \frac{\sum_{i=1}^{n}(p_i - \overline{p})(a_i - \overline{a})}{\sqrt{\sum_{i=1}^{n}(p_i - \overline{p})^2 \sum_{i=1}^{n}(a_i - \overline{a})^2}} \tag{8}$$

$$MSE = \sqrt{\frac{\sum_{i=1}^{n}(p_i - a_i)^2}{n}} \tag{9}$$

$$MAE = \frac{\sum_{i=1}^{n}|p_i - a_i|}{n} \tag{10}$$

Onde, p_i é o valor predito para o exemplo i e a_i seu valor real. $\overline{p}$ e $\overline{a}$ são as médias e n o número de exemplos.

4.1. Experimentos com Artificial 1

O primeiro conjunto de dados (Artificial 1) contém dois atributos preditivos: $x(X)$ (determinado) e $y(Y)$ (não determinado). Cada exemplo contém 8 literais sobre y, para os quais os valores podem ser agregados. Todos os valores numéricos são aleatórios e uniformemente distribuídos entre 0 e 10. A função objetivo para este conjunto de dados está mostrada na figura 2.

Executamos validação cruzada 5x2 [Dietterich 1998]. Os resultados indicaram que o HTILDE-RT obteve medidas de qualidade muito próximas às do TILDE-RT com tempos de aprendizado menores e árvores mais simples. Realizamos o teste t pareado e corrigido [Nadeau and Bengio 2003] [Witten and Frank 2005] com confiança de 0,001 e o HTILDE-RT obteve significância estatística em termos de ter melhor eficiência e gerar árvores menores.

Embora o HTILDE-RT tenha obtido medidas de erro ligeiramente maiores, não houve diferença no coeficiente de correlação, que é a avaliação mais importante. Em

$x(X), X < 5?$
$+yes : maxY|y(Y) < 9?$
$| + yes : 4x(X) + 6avgY|y(Y)$
$| + no : 3maxY|y(Y) + 1$
$+no : avgY|y(Y) < 4?$
$| + yes : 3x(X) + 4$
$| + no : x(X) - 2maxY|y(Y) + 3avgY|y(Y)$

Figura 2. Função Objetivo do Artificial 1

	Tempo (s)	Nós	Literais	Coef. Pearson	MSE	MAE
TILDE-RT	1046,8	290,0	580,0	0,92	28,44	3,28
HTILDE-RT	334,6	162,0	1213,0	0,92	28,55	3,29
TILDE-RT*	1054,0	280,5	561,0	0,92	29,45	3,44
HTILDE-RT*	335,1	157,8	1173,3	0,92	29,56	3,45

Tabela 1. HTILDE-RT x TILDE-RT - Artificial 1

contrapartida, o TILDE-RT foi mais de três vezes mais lento e obteve teorias quase duas vezes mais extensas, i.e., árvores com mais nós. O ruído não afetou de forma impactante nenhum dos dois algoritmos.

O número maior de literais indica que, refinamentos com conjunções maiores separam melhor os conjuntos quando menos exemplos são observados, i.e., rodando em fluxos usando amostragem. No caso do TILDE-RT, por observar todos os exemplos, bastam poucos literais para conseguir a separação.

4.2. Experimentos com Artificial 2

O segundo conjunto de dados (Artificial 2) inclui três atributos preditivos: $x(X)$, $y(C, Y)$ e $z(Z)$, dos quais y é não determinado e possui 15 literais para cada exemplo. Igualmente, os valores numéricos para x, y e z são uniformemente distribuídos entre 0 e 10. A variável C no literal de y possui domínio booleano. Seguem, a função objetivo na figura 3 e os resultados na tabela 2.

$x(X), X < 6?$
$+yes : minY|y(\_, Y) < 1?$
$| + yes : 2x(X) + 3maxY|y(true, Y) + 3$
$| + no : 2x(X) + 5minY|y(\_, Y)$
$+no : -2x(X) + z(Z)?$

Figura 3. Função objetivo do Artificial 2

No Artificial 2 o HTILDE-RT foi estatisticamente mais rápido, com tempos mais de três vezes menores, com confiança de 0,001. Neste conjunto de dados o HTILDE-RT parece ter conseguido uma performance melhor, por ser um conjunto de dados mais

	Tempo (s)	Nós	Literais	Coef. Pearson	MSE	MAE
TILDE-RT	2632,7	132,0	264,0	0,99	1,48	0,95
HTILDE-RT	734,8	197,0	1756,0	0,99	1,49	0,95
TILDE-RT*	2636,6	123,0	246,0	0,98	2,48	1,25
HTILDE-RT*	723,1	198,0	1692,5	0,98	2,49	1,25

Tabela 2. HTILDE-RT x TILDE-RT - Artificial 2

complexo. Não houve diferença estatística significativa para o coeficiente de correlação nem para o erro absoluto quadrático. Desta vez o ruído afetou ambos algoritmos, porém de forma semelhante e discreta. Novamente, refinamentos com mais literais foram mais utilizados pelo HTILDE-RT.

Conforme argumentado em [Ikonomovska et al. 2011], à medida que acrescenta-se ruído aos dados no fluxo o algoritmo adia suas decisões para ajustar-se melhor à perturbação, reduzindo o tamanho dos modelos gerados, conforme observado com o HTILDE-RT. Entretanto, contemplou-se efeito parecido no TILDE-RT. Isto pode ser explicado pela estratégia de dividir-para-conquistar do algoritmo tradicional, onde a decisão de dividir uma folha necessitará de tão mais exemplos quanto maior for a magnitude da perturbação nos dados, logo é razoável ter-se árvores menores.

Destacamos que melhorias são possíveis com ajustes dos parâmetros $emin$, τ e δ e que outros resultados mostraram que ajustes do $emin$ melhoram o tamanho das teorias.

5. Conclusão

Neste trabalho apresentamos um algoritmo para aprender Árvores Relacionais de Regressão, HTILDE-RT, capaz de tornar o aprendizado mais rápido em bases de dados grandes. Tal algoritmo é uma modificação do HTILDE, o qual é baseado no TILDE e VFDT, para o problema de árvores relacionais de regressão.

Os experimentos indicaram que o HTILDE-RT foi capaz de acelerar o processo de aprendizado sem perdas aos modelos gerados, chegando a obter desempenho mais de três vezes mais rápido, sem diferenças significativas no coeficiente Pearson. Além disso, no dataset Artificial 1 conseguiu teorias quase duas vezes menores, apesar de não rodar usando todos os exemplos de uma só vez. A adição de ruído afetou de forma semelhante e discreta ambos algoritmos.

Como trabalhos futuros, desejamos variar os parâmetros δ, $emin$ e τ para melhores resultados e testar o HTILDE-RT em outros conjuntos de dados relacionais gigantes no problema da regressão. Planejamos estender o uso do limitante de Hoeffding no algoritmo ReMauve (Relational Model Trees) [Vens et al. 2006] bem como desenvolver o suporte a mudanças de distribuição [Hulten et al. 2001].

6. Agradecimentos

Gostaríamos de agradecer a Hendrik Blockeel e Jan Ramon por fornecerem o código fonte do TILDE, a Celine Vens, Daan Fierens, Jan Struyf pela ajuda com o código e em particular a Celine Vens pela ajuda com os dados artificiais. Agradecemos também a Carina Lopes pela ajuda com o HTILDE. Gostaríamos de agradecer à CAPES, ao CNPq, à FAPERJ e à FACEPE pelo suporte financeiro.

Referências

Blockeel, H., Dehaspe, L., Demoen, B., Janssens, G., mon, J., and Vandecasteele, H. (2002). Improving the efficiency of inductive logic programming through the use of query packs. *Journal of Artificial Intelligence Research*, 16:135–166.

Blockeel, H. and Raedt, L. D. (1998). Top-down induction of first-order logical decision trees. *Artificial Intelligence*, 101(1-2):285–297.

Blockeel, H., Raedt, L. D., Jacobs, N., and Demoen, B. (1999). Scaling up inductive logic programming by learning from interpretations. *Data Mining and Knowledge Discovery*, 3(1):59–93.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Chapman & Hall, New York, NY.

Dietterich, T. G. (1998). Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923.

Domingos, P. and Hulten, G. (2000). Mining high-speed data streams. In *Knowledge Discovery and Data Mining*, pages 71–80.

H. Blockeel, L. D. J. R. J. S. A. V. A. C. V. and Fierens, D. (2005). *The ACE Data Mining System - User's Manual*.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. 58:13–30.

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*, pages 97–106, San Francisco, CA. ACM Press.

Ikonomovska, E., Gama, J., and Džeroski, S. (2011). Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery*, 23:128–168. 10.1007/s10618-010-0201-y.

Lopes, C. and Zaverucha, G. (2009). Htilde: scaling up relational decision trees for very large databases. In *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC), Honolulu, Hawaii, USA*, pages 1475–1479.

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill, New York.

Muggleton, S. (1991). Inductive logic programming. *New Generation Computing*, 8(4):295–318.

Nadeau, C. and Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, 52(3):239–281.

Vens, C., Ramon, J., and Blockeel, H. (2006). Remauve: A relational model tree learner. In Muggleton, S., Otero, R. P., and Tamaddoni-Nezhad, A., editors, *ILP*, volume 4455 of *Lecture Notes in Computer Science*, pages 424–438. Springer.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Series in Data Management Sys. Morgan Kaufmann, second edition.