Classificação de Dados: uma visão geral

Autoria: Paulo Sérgio de Souza Coelho, Gerson Lachtermacher, Nelson Ebecken

Resumo

As técnicas de classificação de dados têm se tornado uma importante ferramenta para as empresas, com aplicações relativas a aprovação de créditos, diagnósticos médicos, previsão de performance, marketing seletivo, dentre outras. O volume de dados armazenados atualmente contém muitas informações que não podem ser acessadas através de simples consultas (*queries*) aos bancos de dados. O papel da classificação é determinar rótulos, ou classes, para diferenciar grupos de objetos dos quais não se têm maiores informações (objetos novos). Em geral, os rótulos possíveis já são conhecidos de antemão, através de alguns objetos que já estão classificados (objetos conhecidos). No caso de dados armazenados nas empresas os objetos são registros, ou seja, estão descritos através de um conjunto de atributos; o objetivo principal é estabelecer modelos (matemáticos e/ou estatísticos) que possam determinar a classe dos registros de acordo com os valores dos seus atributos. Existem diversas técnicas diferentes para a tarefa de classificação de dados. Este artigo faz uma discussão dos aspectos relevantes à atividade de classificação, considerando algumas das técnicas de classificação mais usuais, no intuito de mostrar os detalhes das técnicas. Duas bases de dados reais foram utilizadas para comparações.

1 Introdução

Já não é mais nenhuma novidade a crescente utilização do nível de informatização das empresas em todo o mundo. Considerações extremas como [5] chegam a afirmar que recentemente este crescimento se deu de maneira exponencial. De qualquer maneira, diante do imenso crescimento na utilização dos sistemas de informação, e sabendo da sua natureza relativa ao armazenamento dos dados que surgem através das operações cotidianas dos sistemas. No caso das organizações, estas operações estão relacionadas com os cadastros de produtos/serviços, clientes e fornecedores, bem como as transações que são realizadas com estes, chegando ao nível dos registros financeiros e contábeis.

O armazenamento de dados sempre esteve diretamente associado com a recuperação de informações. Este processo é conhecido como **consulta** (*query*) ao banco de dados. As consultas ao banco de dados estão relacionadas com registros específicos (por exemplo, o que possui maior ou menor valor de um determinado atributo) ou com estatísticas de um conjunto de registros (por exemplo, média dos valores de um determinado atributo). Para realizar consultas em bancos de dados foram desenvolvidas linguagens específicas, das quais a mais utilizada é chamada de **SQL – *Structured Query Language*** ([4]). Entretanto, torna-se necessário para o usuário das informações contidas o banco de dados é uma análise mais profunda das informações ali contidas. O resultado destas análises é o que chamamos de conhecimento contido no banco de dados, e está relacionado a medidas de similaridade, co-ocorrência (correlação), relações de dependência, de ordem, etc. A aquisição do conhecimento requer modelos matemáticos e estatísticos elaborados; não pode ser obtido a partir de simples consultas ao banco de dados.

Neste sentido, o desenvolvimento dos sistemas de banco de dados diferencia os sistemas de **banco de dados operacionais** (dinâmico) e os ***Data Warehouses***, ou armazéns de dados, (estáticos). Os *Data Warehouses* tornam-se fundamentais para a otimização dos recursos computacionais. A quantidade de informação que está armazenada e é consultada,

alterada ou inserida nos bancos de dados operacionais e a freqüência que estas operações são realizadas não permite operações de análise. Os *Data Warehouses* possuem uma estrutura de armazenamento de dados que privilegia a análise, ao contrário dos bancos operacionais (relacionais) que privilegiam a velocidade e evitam a duplicidade das informações. Os *Data Warehouses* oferecem as mesmas ferramentas de consulta que o sistema de banco de dados operacional, mas os dados ficam estruturados em forma de um cubo, e esta estrutura é uma importante ferramenta de análise dos dados, chamada de **OLAP – *On Line Analytical Process***, ou **cubo OLAP**.

As ferramentas OLAP são úteis para fazer análises de segmentos, considerando as dimensões do cubo (como centros de custo, localidades, departamentos, produtos, etc.), mas representam apenas um passo no caminho em busca da extração do conhecimento contido nos *Data Warehouses*. Algumas ferramentas computacionais que pudessem modelar as informações das massas de dados em conhecimento foram desenvolvidas. Este conjunto de ferramentas é o que chamamos de ***Data Mining***. Estas de ferramentas servem para captar e evidenciar características dos dados que estão sendo analisados. Estas características podem, dentro de alguma medida estatística, ser utilizadas para projeções, de registros não observados. Tipicamente, o conhecimento de que 90% dos clientes que possuem um determinado perfil apresentaram problemas de inadimplência sinaliza para a empresa que é necessária cautela com clientes com este perfil.

O conceito de ***Knowledge Discovery in Databases (KDD)***, ou Descoberta de Conhecimento em Banco de Dados, é aqui diferenciado do conceito de *Data Mining*. Consideramos que *Data Mining* é apenas uma das fases do KDD. Consideramos ainda que a atividade de KDD compreende cinco segmentos de trabalho: definição do problema (seleção dos dados); limpeza e preparação dos dados; transformação de atributos; *Data Mining* e interpretação e análise de resultados. A Figura 1 ilustra o processo KDD:

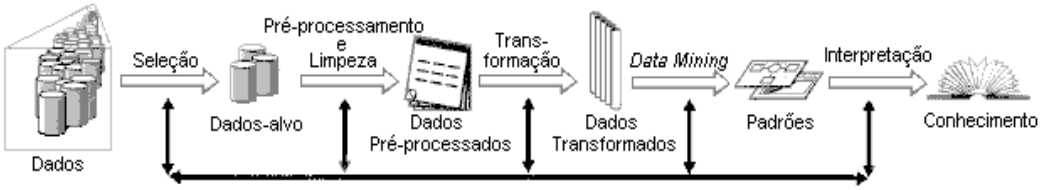

Figura 1: Etapas do KDD. Fonte: [13]

A primeira etapa é a atividade de seleção das informações contidas no *Data Warehouse*. Geralmente nem todas as informações contidas no *Data Warehouse* são utilizadas para a análise em questão. A segunda etapa é possivelmente a etapa mais importante do processo. O pré-processamento é fundamental para a etapa de modelagem do conhecimento (*Data Mining*). Um pré-processamento ineficiente pode reduzir ou até eliminar as chances de uma modelagem eficiente ([15]). Nesta etapa os dados são preparados para a modelagem, resolvendo problemas de redundância, inconsistência e ausência de valores, sendo por isso chamada de etapa de limpeza.

Para a técnica de classificação, as atividades de pré-processamento que se fazem necessárias são a limpeza (eliminação ou redução de ruídos e valores faltando), análise de relevância (eliminação de atributos irrelevantes ou redundantes) e transformação dos dados (generalização, discretização, substituição de valores alfanuméricos ou até normalização, dependendo da técnica que será empregada).

Geralmente, diversas ferramentas podem fazer o processo de modelagem, e cada ferramenta requer um diferente formato de entrada dos dados. A etapa de transformação adapta os dados já limpos para a ferramenta de modelagem que será utilizada. A etapa de *Data Mining* pode ser considerada como a etapa relacionada efetivamente com a descoberta do conhecimento, quando os modelos são construídos, e as avaliações deste modelo são realizadas (estimadas). Os tipos de modelos de *Data Mining* podem ser divididos em três grupos diferentes Regras de Associação, Classificação ou Segmentação (*Clustering*). Historicamente as atividades de agrupamento são as mais antigas [10], e as Regras de Associação foram descobertas no início dos anos 90 [1]. Finalmente, a etapa de Interpretação é responsável pela adequação da saída da ferramenta de modelagem às necessidades do usuário. Algumas ferramentas de visualização e de navegação em dados podem ser utilizadas.

Este trabalho concentra-se particularmente na etapa de *Data Mining*, e mais especificamente nas técnicas de classificação. Existem diversas técnicas de classificação, que se diferem fundamentalmente pela metodologia de tratamento do problema. As conseqüências podem ser percebidas em fatores como precisão, tempo, habilidade de trabalhar com grandes massas de dados e complexidade do modelo. Diante de todas opções, é importante fazer a escolha da técnica a ser utilizada de maneira a buscar aquela que melhor se adapta ao problema em questão. Esta pode ser uma tarefa árdua, exigindo o experimento de várias possibilidades antes de definir qual tipo de modelo deverá ser usado.

O presente estudo é um guia para esta busca. Este artigo faz uma discussão sobre as questões que devem nortear o processo de escolha da técnica de classificação, notadamente a precisão, aplicada às principais técnicas de classificação. Os conceitos teóricos fundamentais estão descritos na próxima seção. A seção de Descrição Técnica mostra os fundamentos matemáticos e estatísticos envolvidos nas técnicas estudadas. Para fazer a comparação, alguns softwares foram utilizados, o que pode ser visto na seção 4, sobre Experimentos. Nesta seção existe também uma descrição detalhada das bases de dados utilizadas, suas características físicas e um histórico de tratamentos realizados. Finalmente, na seção de Conclusões, um resumo final e considerações sobre os resultados encontrados.

2 Classificação

Existem duas formas bem distintas de construir um modelo de classificação. Ele pode ser obtido entrevistando especialistas – muitos sistemas baseados em conhecimento são construídos desta maneira, apesar das dificuldades envolvidas neste enfoque. Alternativamente, classificações já realizadas (se armazenadas) podem ser examinadas automaticamente e um modelo pode ser construído indutivamente, generalizando estes exemplos específicos [16]. Esta segunda forma de construir modelos é a atividade que em *Data Mining* é chamada de classificação.

Existem muitas técnicas diferentes para a tarefa de classificação: árvores de decisão, classificação bayesiana e redes bayesianas (*belief networks*), indução de regras, redes neurais, classificadores baseados em proximidade (*KNN – k-nearest neighbor*), algoritmos genéticos, *rough sets*, lógica *fuzzy*, entre outros. Novas técnicas têm sido desenvolvidas a partir da fusão de algumas técnicas clássicas, principalmente usando a lógica *fuzzy* (árvores de decisão *fuzzy*, redes neuro-*fuzzy*, indução de regras *fuzzy*).

A classificação é um processo realizado em duas fases. A primeira fase é chamada de fase de treinamento, e a segunda é chamada de fase de teste. Na fase de treinamento, o modelo é construído descrevendo um conjunto predeterminado de classes contidas em dados conhecidos. Este processo é feito através da análise dos registros de uma base de dados. Cada registro deve pertencer a uma classe predefinida, que é o valor de um dos seus atributos,

chamado atributo de rótulo da classe. No presente contexto, registros são referidos como amostras, exemplos ou objetos. Os registros analisados para a construção do modelo formam, coletivamente, o conjunto de dados de treinamento. O conjunto de treinamento pode ser uma parte do conjunto de registros classificados ou todos os registros, dependendo da técnica de estimação do erro utilizada (três diferentes técnicas serão descritas ainda nesta seção). Cada elemento deste conjunto é chamado de amostra de treinamento, ou registro de treinamento. Como o rótulo da classe de cada registro de treinamento é conhecido, este processo é comumente conhecido como aprendizado supervisionado (em oposição às atividades de segmentação que são conhecidas como aprendizado não supervisionado).

O propósito da fase seguinte é estimar a precisão do modelo construído na primeira fase. Uma maneira de medir a precisão de um classificador é testá-lo em casos subseqüentes (desconhecidos) cuja classificação correta seja conhecida e comparar a classificação feita pelo modelo com a classificação correta. Conceitualmente, depois que o classificador foi construído usando um conjunto de casos, toma-se um outro conjunto de casos suficientemente grande (virtualmente infinito), retirado da mesma amostra que o primeiro conjunto. Observa-se a classe correta de cada um destes casos e então se compara com o resultado oferecido pelo classificador. Pode-se tomar uma taxa percentual de erros (ou acertos).

Para fins formais, um modelo probabilístico é estabelecido em [2], que definiu que a taxa de erro é a probabilidade que a classificação seja feita de maneira errada em um novo caso que tenha sido retirado da mesma distribuição que os casos que serviram para estabelecer o classificador. Em problemas reais, geralmente a distribuição é desconhecida, e, portanto a probabilidade precisa ser estimada. A partir do conjunto de casos já classificados, que será usado tanto para construir o classificador quanto para estimar sua precisão, é feita a estimativa. Três diferentes maneiras de estimar o erro do modelo são normalmente utilizadas:

a) **Estimativa por resubstituição**: é a menos precisa. O mesmo conjunto de casos que foi utilizado para construir o classificador é utilizado para estimar a taxa de erro do classificador. Os casos são submetidos ao classificador, e o resultado é comparado com a classe a que efetivamente pertencem. O problema com esta técnica para estimativa é que usa a mesma amostra da que foi utilizada no treinamento (construção do classificador) para estimar a precisão. Isto torna a estimativa absolutamente tendenciosa.

b) **Estimativa por amostra de teste**: o conjunto de casos já classificados é dividido em dois subconjuntos. Um destes será utilizado como conjunto de treinamento na construção do classificador. O outro conjunto, chamado de conjunto de teste, será utilizado para estimar a precisão do classificador que foi construído com o primeiro conjunto. É comum fazer com que a divisão seja feita na proporção de 2 para o conjunto de treinamento e 1 para o conjunto de teste ([2]). Uma desvantagem deste procedimento é reduzir o tamanho do conjunto de treinamento. Se o conjunto de casos já classificados for suficientemente grande isto não é um problema. Entretanto, se o conjunto de casos for pequeno, esta técnica de estimativa de erro pode ser indesejada.

c) **Validação cruzada** (***cross-validation***): este método divide o conjunto de casos cuja classificação é conhecida em uma determinada quantidade, digamos n, de subconjuntos. O procedimento estimará a taxa de erro através da média de n taxas de erro estimadas por amostra de teste considerando n diferentes classificadores. Cada classificador é criado considerando um dos n subconjuntos como conjunto de teste e considerando os outros $n - 1$ subconjuntos reunidos como conjunto de treinamento. Esta técnica é altamente recomendada quando o conjunto de casos classificados é pequeno, pois cada um dos casos é utilizado para construir os classificadores, e é usado exatamente uma vez como caso de teste. Além disso, esta técnica de estimativa da taxa de erro é muito estável, e a estimativa é bem mais próxima

do valor probabilístico esperado. É comum proceder a Validação Cruzada utilizando $n = 10$ ([2]).

3 Descrição Técnica

As técnicas de classificação que utilizamos podem ser agrupadas em três principais tipos: árvores de decisão, classificadores bayesianos (*naïve* e *belief network*) e baseados em vizinhança (estatísticos). A compreensão das técnicas é fundamental para o entendimento dos modelos obtidos, que podem requerem parâmetros, e dos resultados, como são obtidos.

3.1 Árvore de Decisão

Árvores de decisão são ferramentas poderosas e populares para as atividades de classificação. Possuem uma forma de representação simples, de maneira que o modelo criado é fácil de ser compreendido pelo usuário, pois as árvores de decisão representam regras, que são estruturas facilmente compreendidas. A estrutura de árvore é como um diagrama de fluxo, aonde cada nó interno representa um teste em um atributo, cada galho representa um resultado do teste, e os nós folha representam classes ou, possivelmente, uma distribuição de probabilidades das classes.

Um exemplo de árvore de decisão pode ser visto na Figura 2, que classifica possíveis compradores de uma determinada loja fictícia, aonde nós internos estão representados por retângulos e os nós folhas estão representados por círculos. Para classificar um registro desconhecido, os valores dos seus atributos são testados na árvore. Um caminho é definido desde a raiz até uma folha, que contém a previsão de classe para este registro. Observe que muitas folhas podem fazer a mesma classificação, mas como cada folha tem um caminho exclusivo para a raiz (uma árvore é um grafo acíclico) a classificação possui um conjunto de condições específico.

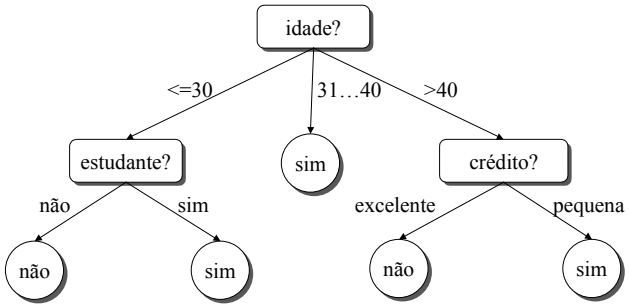

Figura 2: Árvore de Decisão. Adaptado de [7].

Pode-se considerar também que uma árvore de decisão representa uma série de perguntas. A resposta de cada pergunta determina um caminho na árvore e, portanto, qual será a pergunta seguinte. Para o processo de construção da árvore, se as perguntas forem bem escolhidas, apenas uma pequena quantidade de perguntas será suficiente para classificar os registros.

Algumas ferramentas computacionais que induzem Árvores de Decisão necessitam que todos os atributos devem ser **categóricos**, ou seja, possuem apenas um conjunto suficientemente pequeno de valores previamente conhecidos. Registros com valores numéricos devem ser discretizados, de maneira a criar categorias. Na Figura 2 podemos ver que o atributo idade foi discretizado em (pelo menos) três categorias: até 30 anos, de 31 a 40 anos e acima de 40 anos. Isto representa uma grande limitação do método. O procedimento de

discretização pode trazer grandes instabilidades ao modelo, em função da quebra da continuidade. Por exemplo, haverá mesmo diferença entre dois clientes, um com 30 e outro com 31 anos? Para suavizar este problema, técnicas baseadas em aritmética *fuzzy* têm sido desenvolvidas (maiores detalhes técnicos em [6]e [11]).

O problema de possuir o atributo rótulo de classe com valores contínuos é um problema a parte. Na literatura, este problema assume um nome específico: problemas de previsão. As técnicas mais utilizadas para problemas de previsão são as equações de regressão ([8], [19] e [21]), e as árvores de regressão ([2]).

Existem diversas estratégias para a construção de uma Árvore de Decisão, mas a mais conhecida é devida a [16], e é a seguinte: a árvore começa como um nó simples, que representa todas as amostras do treinamento. Se todas as amostras forem da mesma classe, o nó se transforma em uma folha e esta é rotulada como aquela classe. Caso exista mais de uma classe no conjunto de amostra o algoritmo usa uma medida, baseada na entropia, chamada de **ganho de informação** como uma heurística para selecionar o atributo que melhor distinguirá as amostras em classes individuais. Este atributo torna-se o "teste" do nó intermediário. Um galho é criado para cada valor conhecido do teste dos atributos e as amostras são divididas entre estes galhos; o algoritmo utiliza o mesmo processo recursivamente para construir a partir das amostras obtidas em cada divisão uma nova (sub) árvore. Uma vez que um atributo já é usado como teste, este não pode ser considerado em nenhum dos próximos nós. O processo de repartição da árvore só acaba quando uma das seguintes condições torna-se verdadeira: todas as amostras em um determinado nó pertencem à mesma classe ou não existam mais atributos pelos quais a amostra pode ser dividida. Neste caso, a classificação pode ser imposta pela maioria.

A medida de ganho de informação é usada para selecionar o atributo teste em cada um dos nós da árvore. O atributo com o maior ganho de informação, ou com a melhor redução de entropia, é escolhido como o atributo teste para o corrente nó. Este atributo minimiza a quantidade de informação necessária para classificar a amostra no resultado das partições.

Um termo conhecido como *overfitting* é utilizado para descrever o modelo que se ajusta completamente aos dados. Este ajustamento completo é indesejado, pois durante a construção da árvore, muitos galhos irão refletir anomalias dos dados em função de ruídos e *outliers* (valores que são inesperados por serem muito maiores ou muito menores do que o restante). Usa-se uma estratégia conhecida como poda da árvore para resolver este problema de *overfitting* dos dados. Esta estratégia geralmente utiliza medidas estatísticas para remover os galhos mais incertos, geralmente resultando em uma classificação mais rápida e com maiores probabilidades de se classificar corretamente um novo dado.

A eficiência de algoritmos de árvore de decisão, tais como CART [2], ID3 [16]e C4.5 [16], já são considerados bem estabelecidos [7]. Sobretudo quando o trabalho acontece com bancos de dados do mundo real, ou seja, bancos de dados com muitas informações, onde aspectos como eficiência e escalabilidade tornam-se preocupantes. Este problema é relevante porque a maioria dos algoritmos, inclusive os de indução de Árvores de Decisão, têm a restrição de que as amostras de treinamento residam na memória principal do comutador. Entretanto, as árvores de decisão figuram como técnicas rápidas quando comparadas com outras técnicas, como redes neurais ou bayesianas.

3.2 Classificadores Bayesianos

Classificadores Bayesianos são classificadores estatísticos, que podem predizer a probabilidade de ocorrência de cada classe assim como a probabilidade de uma dada amostra

pertencer a uma determinada classe. A classificação Bayesiana é baseada no teorema de Bayes. Estudos comparando algoritmos de classificação apontaram que o classificador Bayesiano "Ingênuo" (*Naïve Bayesian Classifier*) pode ser comparado em nível de performance com as árvores de decisão e com os classificadores de Redes Neurais [7]. Uma das características marcantes dos classificadores Bayesianos é a exatidão e velocidade quando aplicados a grandes bases de dados.

O classificador Bayesiano Ingênuo assume que o efeito do valor de um atributo em uma determinada classe é independente do valor dos outros atributos desta mesma classe. Esta suposição é conhecida como independência condicional das classes e é assumida para reduzir o esforço computacional. Esta hipótese simplificadora não é verdade na maioria dos dados reais, o que justifica o adjetivo ao algoritmo (ingênuo).

A metodologia é a seguinte: cada amostra é representada por um vetor *n*-dimensional de características, $X = (x_1, x_2, ..., x_n)$, retratando *n* medidas feitas na amostra. Estas medidas equivalem ao valor observado dos atributos, $A_1, A_2, \Downarrow, A_n$. Suponha que existem *m* classes, $C_1, C_2, \Downarrow, C_m$. Dada uma amostra desconhecida, *X*, o classificador determinará que *X* pertence a classe que possuir a maior probabilidade *a posteriori*, condicionada por *X*. Isto quer dizer, o classificador Bayesiano Ingênuo designa uma amostra desconhecida *X* a uma classe C_i se, e somente se, $P(C_i \mid X) > P(C_j \mid X)$, para $1 \leq j \leq m, j \neq i$.

Assim, maximiza-se $P(C_i \mid X)$. A classe C_i pela qual $P(X \mid C_i)$ é maximizada é chamada de hipótese *a posteriori* máxima. Pelo teorema de Bayes, $P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)}$. Uma vez que $P(X)$ é constante para todas as classes, somente $P(X \mid C_i)P(C_i)$ precisa ser maximizado. Se as probabilidades das classes *a priori* não são conhecidas, é comum assumir que as classes são iguais, isto é, $P(C_1) = P(C_2) = P(C_m)$ e então é bastante maximizar $P(X \mid C_i)$. As probabilidades das classes *a priori* podem ser estimadas por $P(C_i) = \frac{s_i}{s}$, onde s_i é o número de amostras que foram usadas no treinamento e que são classificadas como C_i, e *s* é o tamanho total do conjunto de treinamento.

Quando os conjuntos de dados possuem muitos atributos torna-se muito alto o esforço computacional para determinar $P(X \mid C_i)$. De forma a reduzir este esforço computacional, entra em ação a hipótese de independência condicional das classes. Assim,

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i). \quad (1)$$

As probabilidades $P(x_1 \mid C_i)$, $P(x_2 \mid C_i)$, ..., $P(x_n \mid C_i)$ podem ser estimadas da amostra de treinamento, onde:

Se A_k, é categórico, ou seja, discreto, então $P(x_k \mid C_i) = \frac{s_{ij}}{s_i}$, onde s_{ik} é o número de amostras de treinamento da classe C_i tendo o valor x_k para o atributo A_k, e s_i é o número de amostras da base de treinamento pertencendo à classe C_i.

Se A_k é numérico, então pode-se assumir que o atributo possui uma distribuição Gaussiana,

$$P(x_k \mid C_i) = (gx_k, \mu_{Ci}, \sigma_{Ci}) = \frac{1}{\sqrt{2\pi} \cdot \sigma_{Ci}} e^{-\frac{(x_k - \mu_{Ci})^2}{2\sigma_{Ci}^2}}, \quad (2)$$

onde é $g(x_k, \mu_{Ci}, c_{Ci})$ a função Gaussiana para o atributo A_k, enquanto que $\mu_{Ci} e c_{Ci}$, são respectivamente a média e o desvio padrão, dando os valores ao atributo A_k para as amostras de treinamento da classe C_i.

De forma a classificar uma amostra desconhecida *X*, $P(X \mid C_i)\, P(C_i)$ é estimada para cada classe C_i. A amostra *X* é designada a classe C_i, se e somente se,

$$P(X \mid C_i) \cdot P(C_i) > P(X \mid C_j) \cdot P(C_j), \text{ for } 1 \le j \le m, j \ne 1. \quad (3)$$

Classificadores Bayesianos possuem, em geral, a menor taxa de erro em comparação com os outros classificadores.

Enquanto que o classificador Bayesiano Ingênuo assume a independência entre as condições das classes, as Redes de Confiança Bayesiana especificam distribuições de probabilidades condicionais. Elas permitem definir as independências condicionais das classes através de subconjuntos de variáveis. Estas redes são também conhecidas como Redes de Confiança, Redes Bayesianas ou Redes Probabilísticas.

3.3 K-Vizinhos Mais Próximos

Estes classificadores são baseados no aprendizado por analogia. As amostras de treinamento são descritas por vetores numéricos *n*-dimensionais. Cada amostra representa um ponto em um espaço *n*-dimensional, chamado de espaço de padrões (*pattern space*). Quando uma amostra não classificada é fornecida, o classificador busca no espaço de padrões as *k* amostras que sejam mais próximas da amostra desconhecida. Estas *k* amostras são as ***k*-vizinhas mais próximas**. A proximidade é definida em termos de alguma métrica (distância). Ou seja, dados dois pontos em um espaço *n*-dimensional $X = (x_1, x_2, \ldots, x_n)$ e $Y = (y_1, y_2, \ldots, y_n)$, a distância entre eles pode ser dada por:

$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$ Métrica Euclidiana

$d(X,Y) = \sum_{i=1}^{n} |x_i - y_i|$ Métrica de Manhatan

$d(X,Y) = \sqrt[q]{\sum_{i=1}^{n} (x_i - y_i)^q}$ Métrica de Minkowski

sendo que a mais comum é a Métrica Euclidiana.

A amostra desconhecida é associada à classe mais comum entre seus *k*-vizinhos mais próximos. Quando *k* = 1, a amostra desconhecida é associada à classe da amostra de treinamento que lhe é mais próxima no espaço de padrões.

Nearest neighbor classifiers são considerados modelos de aprendizado "preguiçosos" (*lazy learners*), no sentido de que nenhum modelo é construído. As amostras de treinamento são armazenadas e consultadas quando uma amostra nova e desconhecida necessite ser classificada. Classificadores assim são muito rápidos na etapa de treinamento (que representa apenas o armazenamento dos dados), mas consomem bastante tempo para comparar uma amostra não rotulada. Normalmente, são requeridas técnicas de indexação eficientes para agilizar esta consulta. Além disso, se a memória principal do computador não for suficiente para o armazenamento dos dados de treinamento, o procedimento pode ficar muito mais demorado.

4 Experimentos

As técnicas anteriormente descritas foram experimentadas em um total de seis softwares, comerciais e acadêmicos. Para os experimentos escolhemos duas bases de dados, de porte pequeno a médio. A menor é uma base de dados meteorológicos (chamada de **meteo**), e a maior é uma base de dados de perfis de clientes de uma agência de seguro (chamada de **insur**). Vamos fazer uma breve descrição histórica das bases, desde sua coleta até as fases de preparação dos dados.

A base meteo foi criada na década de 50. Durante dez anos (de 1951 a 1960), o Instituto de Pesquisas Meteorológicas da UNESP – Universidade Estadual Paulista, coletou, de hora em hora, 39 informações meteorológicas, gerando um conjunto de dados com aproximadamente 88.000 registros e 39 atributos. Inicialmente armazenados em cartões perfurados e com uma série de limitações, a base de dados foi totalmente organizada e preparada por [9] e posteriormente por [3], resultando em uma base com 19 atributos (incluindo o rótulo da classe) e 26.482 registros. Os atributos incluem informações de hora, dia, mês e ano da observação além das variáveis meteorológicas. Os rótulos de classe podem assumir 9 diferentes valores, e indicam o tempo presente. Dos 9 níveis apenas 7 estão presentes nos dados.

A base insur foi apresentada ao mundo acadêmico em 1998, quando houve uma competição proposta em um congresso ([11]). Os dados foram extraídos de um *Data Warehouse* de uma companhia privada de seguro de vida alemã. Os arquivos disponibilizados pela companhia descrevem o relacionamento entre clientes, contratos de seguro e componentes de tarifa. As tabelas foram unificadas por [3], gerando uma base com 80 atributos e 147.478 registros. Os registros descrevem os clientes, e os atributos caracterizam o cliente, incluindo algumas informações pessoais e dados sobre a apólice. Pela natureza confidencial dos dados, não se têm informações precisas sobre muitos dos atributos. Depois da preparação dos dados feita por [3], a base fica com a dimensão final de 64 colunas e 130.143 linhas. A classe possui apenas dois valores para rótulos, classificando os clientes como fraudadores ou não.

Ao todo, 4 softwares baseados em árvores foram utilizados, todos comerciais: Mineset (SGI), Intelligent Miner (IBM), CART e XpertRule Miner. O único representante dos *Naïve Bayesian Classifiers* é o Roc [18]. Na verdade o software oferece recursos especiais para tratamento de atributos ausentes e para definir distribuições dos atributos, mas estes recursos não foram utilizados. Finalmente, representando os classificadores estatísticos, utilizamos o software comerciais PRW.

De uma maneira geral, os softwares apresentaram uma boa interface, apesar de em alguns casos, bastante peculiar. O XPertRule Miner possui uma esquema para entrada dos dados robusta, permitindo conexão diretamente com um DBMS, mas não é muito intuitivo. Até por que arquivos de texto com tabelas isoladas não podem ser utilizados. O Roc apresenta uma interface baseada em *wizard*, ou seja, passo a passo. O PRW também apresenta uma interface parcialmente baseada em *wizard*. Atenção especial para as interfaces das suítes Mineset e Intelligent Miner. A interface do Mineset é bem compacta diante de toda a sua versatilidade, com uma excelente ferramenta de visualização dos resultados. A interface do Intelligent Miner é bastante carregada, exigindo experiência em navegação no sistema para conseguir os resultados desejados.

Do ponto de vista de tempo, o XPertRule apresentou-se o mais lento comparativamente aos outros softwares. Por outro lado, o CART mostrou-se o mais rápido. E, os dois são baseados em árvores! Naturalmente a diferença está na estratégia de indução da

árvore e no tratamento da escalabilidade. O Roc, que esperávamos se mostrar bem rápido, não mostrou tanta vantagem diante destas bases, tendo um comportamento mediano. O seu problema deve estar no fato de ser acadêmico, e, portanto não ter sido desenvolvido para bases de dados reais, que são realmente grandes.

Sobre a escalabilidade, ponto negativo para o Roc e o Mineset. O Roc só aceita entrada com até aproximadamente 65.000 registros (pareceu-nos o mesmo limite de tamanho de tabela do Excel). Curiosamente, quando usamos bases de dados com mais registros que este limite, o software apresenta um erro inesperado, sobre o tipo de dado interno, ao invés da informação de limitação do software. Como o subconjunto de treinamento da base insur é maior do que este limite, não foi possível testar esta desta base com esta divisão de treinamento e teste (veja descrição da execução abaixo). O Mineset também apresentou um problema com escalabilidade. Utilizamos o software em dois ambientes: MS-Windows e IRIX, este último ambiente nativo. Em nenhum dos dois ambientes o software conseguiu fazer *Cross-Validation* com a base insur. O erro apresentado pelo software também não foi muito explicativo, pois não apresentou informações se a limitação era do software ou do hardware.

Para estimar as taxas precisão dos softwares, optamos pela técnica de amostragem de teste, que estava disponível em todos os sofwares. Exclusivo o Mineset permitiu que fizéssemos a estimativa por *Cross-Validation* ou por amostra de teste, e por isso aparecem dois valores na Tabela 1. Assim, dividimos cada uma das bases em dois conjuntos, o de treinamento e o de teste. A divisão foi feita de maneira que o conjunto de treinamento fosse 70% da base total. A divisão é a mesma para todos os softwares, exceto o MineSet, que fez sua própria amostragem. A base insur para o Roc também teve amostragem separada, pois foi necessário reduzir o tamanho da base de treinamento diante da limitação de tamanho apresentada pelo software. Para o RoC foi fornecido um conjunto de treinamento com 50% da base inicial total.

A Tabela 1 mostra o resumo das performances dos softwares testados. Alguns softwares foram testados com diferentes parâmetros e aparecem em mais de uma linha. No caso do Mineset, as duas linhas representam as duas técnicas para estimar a taxa de erro. As taxas de precisão estão acompanhadas dos desvios padrão. A opção sem *Cross-Validation* indica que a técnica foi usando amostra de teste. O CART foi testado com várias opções, mas o resultado foi sempre precisamente o mesmo, e por isso aparece em apenas uma linha. O XPertRule não ofereceram parâmetros de ajustamento.

Software e opções	**Meteo**	**Insur**
Mineset s/ CrossValidation	84,17% ±0,39%	94,77% ±0,11%
Mineset c/ Cross Validation	84,78% ±0,22%	-
Intelligent Data Miner	83,37%	88,27%
CART	74,16%	93,14%
XPertRule Miner	79,36%	74,75%
Roc normalization 10x4	68,67% / 68,69%	70,82% / 70,90%
Roc normalization 30x8	67,60% / 67,58%	
PRW	75,28% (k = 150)	76,54% (k = 300)
PRW	76,05% (k = 75)	76,54% (k = 300)
PRW	76,41% (k = 50)	

Tabela 1: Precisão dos experimentos realizados

Os parâmetros do Roc foram ignorados, pois gostaríamos de utilizar um classificador *naïve*. Entretanto duas etapas no procedimento permitiam diferentes opções. Como o software

só trabalha com valores normalizados, durante a entrada dos dados é necessário discretizar os atributos numéricos. É necessário então definir a quantidade de valores a partir da qual o atributo será discretizado, e o número de categorias que devem ser criadas neste caso. O valor sugerido pelo software é 10 valores para a variável ser considerada numérica, e o número de categorias a serem criadas é 4 (10x4). Utilizamos também a opção 30x8. No momento da classificação (teste), é possível optar entre dominância estocástica forte ou fraca, que são os dois valores que aparecem nas linhas do RoC.

Estas bases também foram utilizadas em alguns trabalhos anteriores realizados por alguns alunos da COPPE – UFRJ. Nestes trabalhos, a precisão não era o objetivo final. A Tabela 2 mostra os melhores resultados adquiridos. Maiores detalhes sobre as técnicas envolvidas podem ser visto nas referências citadas.

Técnica e Referencia	Meteo	Insur
Neural Network [3]	84,02%	58,64%
Decision Tree [14]	73,25%	61,59%

Tabela 2: Precisão dos experimentos anteriores

5 Conclusões

A análise dos dados através das técnicas de *Data Mining* oferece a possibilidade de obter um conhecimento muito valioso que está contido nos dados. Principalmente no mercado brasileiro, ainda são poucas as empresas que utilizam tais técnicas. É um conceito ainda pouco difundido, e por isso esperamos que nos próximos anos haja um crescimento deste segmento tecnológico.

A precisão aponta para os softwares baseados em Árvores de Decisão. Este resultado, é reforçado pelo ponto de vista da interpretabilidade, pois dentre todos os softwares examinados, as árvores são os únicos que permitem uma interpretação (leitura) do conhecimento contido no modelo. Mais reforço ainda se considerarmos o ponto de vista da velocidade, pois o CART foi o software mais rápido de todos, e o Mineset e o Intelligent Miner ficaram em posições medianas.

Os resultados encontrados são muito específicos para as bases de dados estudadas. Convém observar que o ranking de precisão estabelecido para uma base é diferente para a outra base. Para fazer a escolha da técnica e da ferramenta a ser utilizada é necessário um exame cuidadoso, considerando questões como velocidade, escalabilidade e precisão. Os softwares aqui estudados mostraram que não podem ser deixados de lado no processo de seleção, mas detalhes intrínsecos da base de dados podem alterar completamente o cenário que encontramos.

Finalmente, os resultados mostraram que, com exceção das duas suítes da IBM e da Silicon Graphics, os softwares ainda não estão preparados para bases realmente grandes. Acreditamos que as bases utilizadas têm porte médio, pois em algumas empresas bases bem maiores estão disponíveis. Isto é uma inconsistência, pois um dos princípios do *Data Mining* é a escalabilidade. Os dados do mundo real não estão limitados a uma planilha do Excel. O avanço da capacidade computacional (hardware) permite que os softwares sejam bem mais eficientes nestas situações extremas. Mesmo as duas suítes comerciais tiveram alguns problemas, como mostramos anteriormente.

Bibliografia

[1] Agrawal, R., Imielinksy, T. & Swami, A. Mining Association Rules between Sets of Items in Large Databases. *In: Proceedings of the 1993 ACM SIGMOD Conference*. Washington DC, EUA, 1993.

[2] Breiman, L. et al. *Classification and Regression Trees*. Boca Raton, Florida, EUA: Chapman & Hall/CRC, 1984. 358 p.

[3] Costa, M.C.A. *Data Mining em computadores de alto desempenho utilizando-se Redes Neurais*. Tese (Doutorado em Computação de Alto Desempenho) – COPPE, UFRJ: Rio de Janeiro, 1999.

[4] Date, C. J. *An Introduction to Database Systems*. 7. ed. Addison Wesley, 1999. 960 p.

[5] Dilly, Ruth. *Data Mining: an introduction*. Disponível em: <http://www.pcc.qub.ac.uk/tec/courses/datamining/ohp/dm-OHP-final_1.html>. Acesso em: 12 ago. 1996.

[6] FID : Fuzzy Decision Tree. Disponível em: < http://www.cs.umsl.edu/~janikow/fid/>. Acessado em 10 mar. 2001.

[7] Han, J. & Kamber, M. *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers: San Francisco, 2001.

[8] Hillier, Frederick S. & Lieberman, Gerald J. *Introduction to Operations Research*. 7. ed. McGraw-Hill, 2002.

[9] Hruschka, E. R. *Um estudo sobre a extração de regras de redes neurais em aplicações de Data Mining*. MSc Thesis, COPPE – UFRJ: Rio de Janeiro, 1998.

[10] Kaufman L., & Rousseeuw, P.J. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons: New York, 1990.

[11] Janikow, Cezary Z.. *Fuzzy Decision Trees: Issues and Methods*. IEEE Transactions on Systems, Man, and Cybernetics, Vol. 28, 1-14, 1998.

[12] KDD-Sisyphus. *KDD Sisyphus Workshop – Data Preparation, Preprocessing and Reasoning for Real-World Data Mining Applications*. Disponível em: <http://research.swisslife.ch/kdd-sisyphus, 1998>. Acessado em: 15 mai. 2002.

[13] Machado, C. *O abc da mineração de dados*. Info Exame. São Paulo, V.154, 1999.

[14] Onoda, M. *Estudo sobre um algoritmo de árvore de decisão acoplado a um sistema de banco de dados relacional*. Tese (Mestrado em Computação de Alto Desempenho) – COPPE, UFRJ: Rio de Janeiro, 2001.

[15] Pyle, D. *Data Preparation for Data Mining*. Morgan Kaufmann Publishers: San Francisco, 1999.

[16] Quinlan, J. R. *Induction of Decision Trees*. Machine Learning, 1:81-106, 1986.

[17] Quinlan, J.R. *C4.5*: *Programs for Machine Learning*. Morgan Kaufmann Publishers: San Mateo, 1993.

[18] Ramoni, M. & Sebastiani, P. *Robust Bayesian Classification*. Technical Report TR-68. Knowledge Midia Institute, The Open Univesity. 1999.

[19] Taha, Hamdy A. *Operations Research: An Introduction*. 7. ed. Pearson Education, 2002.

[20] Unica Technologies, Inc. *Pattern Recognition Workbench 2.1 User´s Guide*. Unica Technologies Press: Lincoln, 1996.

[21] Winston, Wayne L. *Operations Research*. 3. ed. Wadsworth Publisher, 1994.