3
Árvore de Decisão

3.1
Introdução

Árvores de decisão são modelos estatísticos que utilizam um treinamento supervisionado para a classificação e previsão de dados. Em outras palavras, em sua construção é utilizado um conjunto de treinamento formado por entradas e saídas. Estas últimas são as classes.

Estes modelos utilizam a estratégia de dividir para conquistar: um problema complexo é decomposto em sub-problemas mais simples e recursivamente esta técnica é aplicada a cada sub-problema (Gama, 2004).

As árvores de decisão estão entre os mais populares algoritmos de inferência e tem sido aplicado em várias áreas como, por exemplo, diagnóstico médico e risco de crédito (Mitchell, 1997), e deles pode-se extrair regras do tipo "se-então" que são facilmente compreendidas.

A capacidade de discriminação de uma árvore vem da divisão do espaço definido pelos atributos em sub-espaços e a cada sub-espaço é associada uma classe.

3.2
Representação de uma árvore de decisão

A figura abaixo representa uma árvore de decisão onde cada nó de decisão contém um teste para algum atributo, cada ramo descendente corresponde a um possível valor deste atributo, o conjunto de ramos são distintos, cada folha está associada a uma classe e, cada percurso da árvore, da raiz à folha corresponde uma regra de classificação. No espaço definido pelos atributos, cada folha corresponde a um hiper-retângulo onde a interseção destes é vazia e a união é todo o espaço (Gama, 2004).

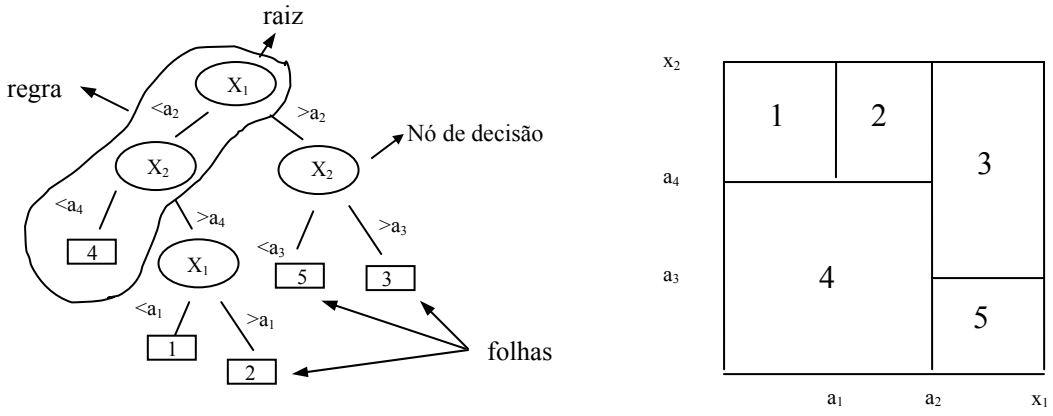

Figura 12 - Representação de uma árvore de decisão e sua respectiva representação no espaço
Fonte: Gama, 2004

O critério utilizado para realizar as partições é o da utilidade do atributo para a classificação. Aplica-se, por este critério, um determinado ganho de informação a cada atributo. O atributo escolhido como atributo teste para o corrente nó é aquele que possui o maior ganho de informação. A partir desta aplicação, inicia-se um novo processo de partição.

Nos casos em que a árvore é usada para classificação, os critérios de partição mais conhecidos são baseados na entropia e índice Gini. (Onoda, 2001)

3.3 Entropia

Entropia é o cálculo do ganho de informação baseado em uma medida utilizada na teoria da informação. A entropia caracteriza a (im)pureza dos dados: em um conjunto de dados, é uma medida da falta de homogeneidade dos dados de entrada em relação a sua classificação. Por exemplo, a entropia é máxima (igual a 1) quando o conjunto de dados é heterogêneo (Mitchell (1997) - Coimbra (2004)).

Dado um conjunto de entrada (S) que pode ter *c* classes distintas, a entropia de S será dada por

$$\text{Entropia}(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Onde: p_i é a proporção de dados em S que pertencem à classe i.

Abaixo têm-se algumas propriedades da entropia.

- Entropia máxima ($\log_2 c$) se $p_i = p_j \ \forall i \neq j$.
- Entropia(S) = 0 se $\exists$ i tal que $p_i = 1$.
- Por hipótese, $o \cdot \log_2 0 = 0$

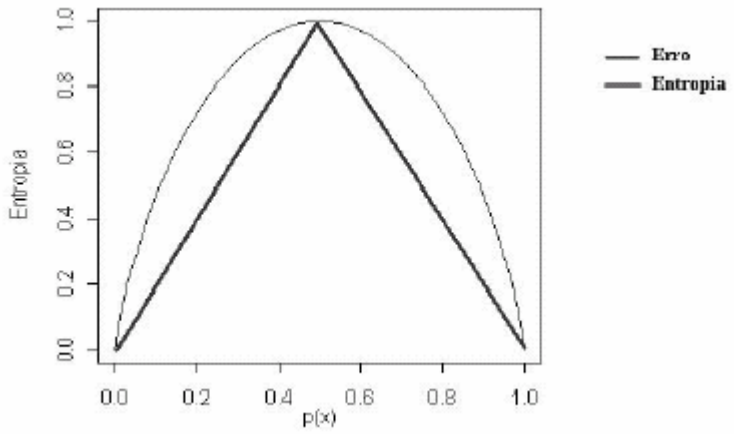

Figura 13 - Representação gráfica p(x) *versus* entropia
Fonte: Gama 2004

O ganho de informação para um atributo A de um conjunto de dados S nos dá a medida da diminuição da entropia esperada quando utilizamos o atributo A para fazer a partição do conjunto de dados.

Seja P(A) o conjunto dos valores que A pode assumir; seja x um elemento deste conjunto e seja S_x o subconjunto de S formado pelos dados em que A = x; a entropia que se obtém ao particionar S em função do atributo A é dada por

$$E(A) = \sum_{x \in P(A)} \frac{|S_x|}{|S|} \text{Entropia}(S_x)$$

O ganho de informação será dado por

$$\text{ganho}(S, A) = \text{Entropia}(S) - E(A)$$

Onde:

Entropia(S) é uma medida de (não)homogeneidade do conjunto S

P(A) é uma medida de (não)homogeneidade estimada para o conjunto S caso utilize o atributo A para fazer a próxima partição

A construção de uma árvore de decisão tem três objetivos, quais sejam: diminuir a entropia (a aleatoriedade da variável objetivo), ser consistente com o conjunto de dados e possuir o menor número de nós.

3.4 Índice Gini

O índice Gini, desenvolvido por Conrado Gini em 1912, mede o grau de heterogeneidade dos dados. Logo, pode ser utilizado para medir a impureza de um nó (Onoda, 2001).

Este índice num determinado nó é dado por:

$$\text{Índice Gini} = 1 - \sum_{i=1}^{c} p_i^2$$

Onde:

p_i é a freqüência relativa de cada classe em cada nó

c é o número de classes

Quando este índice é igual a zero, o nó é puro. Por outro lado, quando ele se aproxima do valor um, o nó é impuro (aumenta o número de classes uniformemente distribuídas neste nó).

Quando, nas árvores de classificação com partições binárias, se utiliza o critério de Gini tende-se a isolar num ramo os registros que representam a classe mais freqüente. Quando se utiliza a entropia, balanceia-se o número de registros em cada ramo.

3.5 O problema do *overfitting*

Overfitting é o ajuste demasiado dos dados de treinamento.

No algoritmo de partição recursiva, a árvore estende a sua profundidade até o ponto de classificar perfeitamente os elementos do conjunto de treinamento. Quando o conjunto de treinamento não possui ruído, o número de erros no

treinamento pode ser zero. Quando este conjunto, entretanto, possui ruído, ou quando o conjunto de treinamento não é representativo, este algoritmo pode produzir árvores em que há *overfitting*.

A definição de *overfitting* (Gama (2004)), é: "uma árvore de decisão *d* faz sobre-ajustamento aos dados se existir uma árvore *d*' tal que: *d* tem menor erro que *d'* no conjunto de treino mas *d*' tem menor erro na população".

Para saber qual é o número ótimo de nós, representa-se graficamente o percentual de erro no conjunto de treinamento e teste *versus* o número de nós da árvore. Quando o erro no conjunto de teste começar a crescer, verifica-se o número de nós nesse ponto.

Na figura abaixo é exemplificado o processo para evitar o *overfitting*.

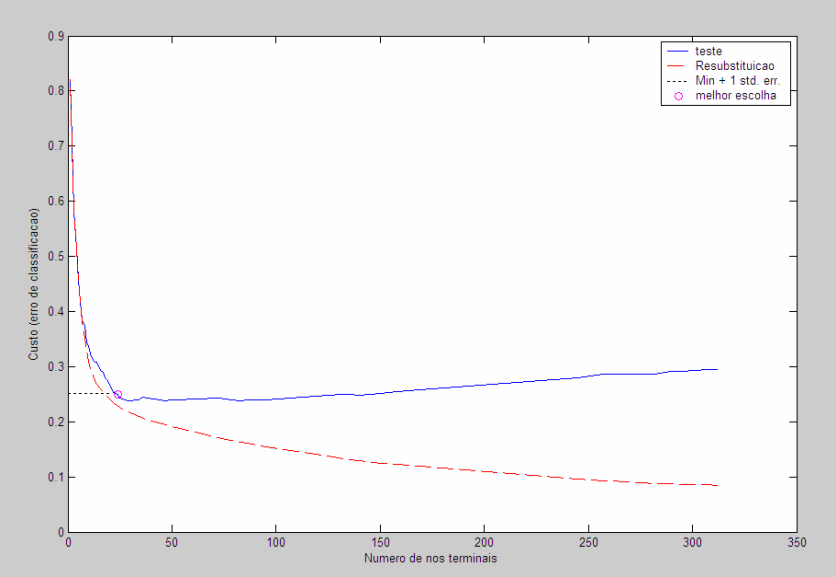

Figura 14 - Representação gráfica o percentual de erro no conjunto de treinamento e teste *versus* o número de nós da árvore

Os problemas encontrados em um algoritmo recursivo são: obter estimativas confiáveis do erro usado para selecionar as partições, otimizar o erro no conjunto de teste e minimizar o erro no treinamento e a dimensão da árvore.

No algoritmo CART, erro no treinamento mais a dimensão da árvore é conhecido como *Cost Complexity Pruning*.

Segundo Breiman (1998), para prevenir o problema do *overfitting* e melhorar a classificação deve-se realizar a poda da árvore.

3.6 Podagem

A podagem pode ser dada de duas circunstâncias. Pode ser usada para parar o crescimento da árvore mais cedo, chamada de pré-podagem ou poda descendente ou pode acontecer com a árvore já completa, chamada de pós-podagem ou poda ascendente.

O processo de podagem pode ser feito da seguinte maneira:

1. Percorre a árvore em profundidade.
2. Para cada nó de decisão calcula:
   - erro no nó
   - soma dos erros nos nós descendentes
3. Se o erro do nó é menor ou igual à soma dos erros dos nós descendentes então o nó é transformado em folha.

Na figura abaixo, tem-se um exemplo de poda. Tome-se por exemplo o nó B, o seu erro é 2 e a soma dos erros nos nós descendentes é 2 + 0 = 2. Daí, o nó B transforma-se em folha.

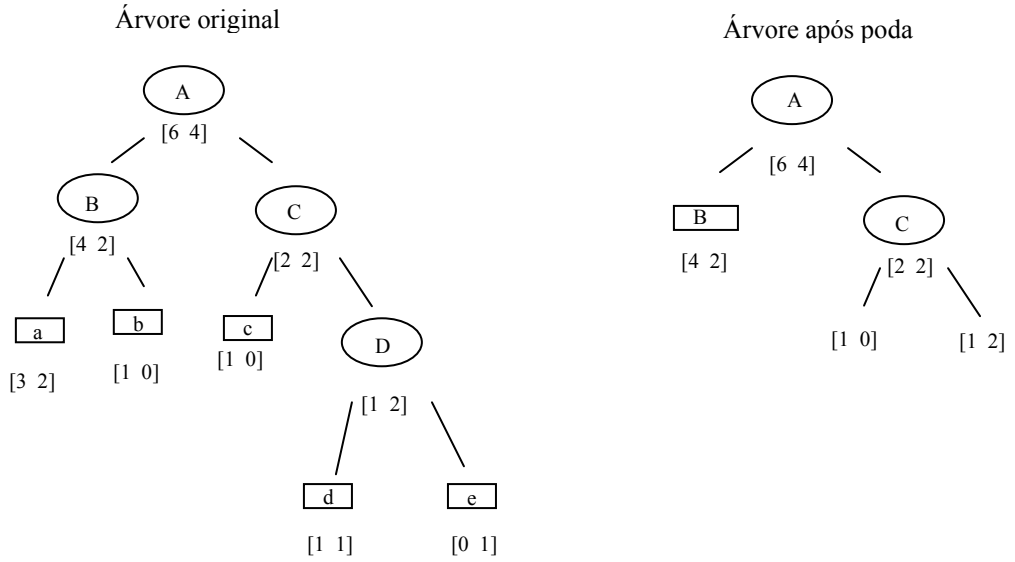

Figura 15 - Exemplo de podagem
Fonte: Adaptado Gama 2004

A medida da diferença dada por uma função baseada nas proporções das classes entre o nó corrente e seus descendentes valoriza a pureza das partições.

Na pós-podagem a árvore é gerada no tamanho máximo e então a árvore é podada aplicando métodos de evolução confiáveis. A sub-árvore com o melhor desempenho será a escolhida.

Este processo pode ser computacionalmente ineficiente pelo fato de gerar uma árvore muito grande e depois esta mesma árvore é reduzida a uma árvore mínima.

Para interromper o crescimento da árvore, verifica-se se a divisão é confiável ou não. Caso seja confiável, para o crescimento da árvore. Este processo é conhecido como pré-podagem da árvore.

A pré-podagem é mais rápida porém menos eficiente que a pós-podagem pelo fato do risco de interromper o crescimento da árvore ao selecionar uma árvore sub-ótima. (Breiman, 1988)

3.7 Algoritmos

Existem vários algoritmos de classificação que utilizam a árvore de decisão. Não se pode determinar qual é a melhor. Dependendo do problema, um algoritmo pode ser mais eficiente que outro.

Dentre os algoritmos tem-se:

- ID3 (Quinlan 1979)
- CART – Classification and Regression Trees (Breiman et al. 1984)
- Assistant (Cestnik et al. 1987)
- C4.5 (Quinlan 1993)
- C5 (See5, Quinlan) disponível no WEKA, Clementine
- CHAID (Chi Square Automatic Interaction Detection)

Neste trabalho será utilizado o algoritmo CART, que utiliza a partição recursiva binária. Segundo Sobral, “suas principais características são: definir o conjunto de regras para dividir cada nó da árvore; decidir quando a árvore está completa; associar cada nó terminal a uma classe ou a um valor preditivo no caso de regressão.” (2003)

Os procedimentos do CART são:

Definir o conjunto de regras para realizar a divisão de um nó em dois nós filhos. As perguntas do algoritmo têm como resposta apenas “sim” ou “não”.

Encontrar a melhor divisão através do critério Gini.

Repetir o processo de divisão até que esta seja impossível ou interrompida.

Aplicar o processo de pós-podagem para encontrar a árvore com o menor custo, ou seja, aquela que possui menor taxa de erro e menor complexidade.