Inteligência Artificial – Trabalho 4

Relatório Sobre Àrvores de Decisão

Miguel Gomes(201505151), Pedro Moreira(201507254) e Rafael Novais(201508010)

1 - Introdução

Uma árvore de decisão é uma representação de uma função que através de uma prévia avaliação de um determinado conjunto de dados, consegue retornar uma decisão para uma nova situação. A utilização de uma árvore de decisão é o método de representação de dados que nos permite ser mais simples e obter uma maior taxa de sucesso quando utilizados em conjunção com algoritmos de indução em árvores de decisão.

Uma árvore de decisão toma como entrada um objeto ou situação descrita por um conjunto de propriedades e gera uma "decisão" de sim / não. As árvores de decisão representam uma função booleana, funções com uma faixa maior de saídas também podem ser representadas, mas por simplicidade nós geralmente ficamos com o caso booleano. Cada nó interno na árvore corresponde a um teste do valor de uma das propriedades e os ramos do nó são rotulados com os valores possíveis do teste. Cada nó da folha na árvore especifica o valor booleano a ser retornado se essa folha for atingida.

2 - Algoritmos de indução de árvores de decisão

As árvores de decisão são apenas uma estrutura de dados, ou seja sem algoritmos capazes de construir estes modelos de previsão de forma eficiente, elas não nos servem de nada.

O processo de indução de árvores de decisão exige um grande poder computacional, mas uma vez construída, o seu uso é imediato e muito rápido, tornando-se uma das vantagens da sua utilização. Dado um conjunto de dados de exemplo consegue representar de forma genérica as relações existentes entre estes e um determinado acontecimento que que estamos a considerar como resultado e partindo destas relações constroem de forma automática a árvore de decisão. Estes algoritmos normalmente utilizam princípios greedy, utilizando métricas para efectuar a escolha de nós ao longo da construção da árvore.

2.1 - Métricas

2.1.1 Ganho de informação: é baseado no conceito de entropia. A entropia é a medida de incerteza que há num sistema. Usando o conceito de entropia podemos calcular a probabilidade de ocorrência de uma determinado evento em cada conjunto de resultados. Dado um conjunto S, com instâncias pertencentes à classe i, com probabilidade , temos:

$$Entropia = \sum_{i} \varphi log \varphi$$

2.1.2 Índice de Gini: considera divisões binárias para cada atributo. Deve-se examinar todos os possíveis subconjuntos que podem ser formados usando os valores conhecidos de um atributo, para determinar a melhor divisão para aquele atributo. Dado um conjunto S, com instâncias pertencentes à classe j, com probabilidade , temos:

$$\acute{I}ndiceDeGini = 1 - \sum_{j} \varphi^{2}$$

2.2 - ID3

O algoritmo ID3 utiliza princípios greedy, escolhendo assim em cada nó da árvore aquele que acha que lhe permitirá alcançar uma árvore mais compacta. Para esta escolha do melhor atributo utiliza como métrica o ganho de informação. O ID3 segue os seguintes passos:

- Comecar com todos os exemplos de treino;
- Escolher o teste que melhor divide os exemplos, ou seja agrupar exemplos da mesma classe ou exemplos semelhantes;
- Para o atributo escolhido, criar um nó filho para cada valor possível do atributo;
- Transportar os exemplos para cada filho tendo em conta o valor do filho;
- Repetir o procedimento para cada filho não “puro”. Um filho é puro quando cada atributo X tem o mesmo valor em todos os exemplos.

2.3 - CART

O algoritmo CART é baseado em Classification and Regression Trees por Breiman.

Uma árvore CART é uma árvore de decisão binária que é construída dividindo um nó em dois nós filhos repetidamente, começando com o nó raiz.

Como métrica para ganho utiliza o índice de Gini. O algoritmo segue as seguintes regras para a construção de árvores de decisão:

- Os valores das variáveis são escolhidos de forma a que a divisão seja a melhor possível.
- Logo que a divisão se aplique a este processo é aplicado aos seus filhos.

- Esta divisão acaba quando o CART deteta que já não existe nenhum ganho ao fazer a divisão ou no caso da existência de outra regra que já foi previamente identificada.

2.4 C4.5

C4.5 constrói árvores de decisão a partir de um conjunto de dados de treinamento da mesma forma que o algoritmo_ID3, sendo uma melhoria do mesmo, utilizando o conceito de_Entropia. Outros dos melhoramentos é a possibilidade de existência de atributos não avaliados num determinado grupo de dados de exemplo. Quando num determinado exemplo existe um atributo sem valor este não entra para o cálculo da métrica. Para além disto o melhoramento mais interessante deste algoritmo é quando este acaba a construção de uma árvore de decisão tenta podar ramos da árvore que não sejam necessários e substitui-los por nós folha.

Este algoritmo possui alguns casos básicos:

- todas as amostras do conjunto pertencem a uma mesma categoria; quando este caso ocorre, o algoritmo simplesmente cria um nó folha para a árvore de decisão e escolhe a categoria em questão;
- nenhuma das características fornece ganho de informação. Neste caso, o algoritmo C4.5 cria um nó de decisão árvore acima usando o valor esperado;
- instâncias previamente não vistas. Novamente, o algoritmo C4.5 cria um nó de decisão árvore acima usando o valor esperado.

3 - Implementação

Neste trabalho utilizamos Java, principalmente porque é uma linguagem simples em que todos nós estamos à vontade para usar e programar mas também devido à facilidade de manipular objetos, listas e strings.

Como estruturas de dados foi utilizada uma estrutura de árvore com a informação guardada nos nodes. Cada node guardava informações como por exemplo a matrix correspondente numa Lista de Listas de Strings, profundidade, entropia dos dados, Lista de Nós filhos, Lista de atributos usados, etc;

A árvore é gerada com o nó root e a partir dele geram-se os filhos que são adicionados a queue para serem processados. São processados baseados na profundidade, ou seja, geram-se os nós de cada nível primeiro. Quando são processados, é calculado o melhor atributo que melhor divide os dados e geram-se os filhos separados pelos diferentes valores do melhor atributo e adicionados a queue. A partir daqui, o algoritmo vai processar até a queue ficar vazia sendo que verifica sempre se os nós são folha e se isto verificado, não os adiciona a queue. Foram usadas algumas HashMaps para guardar valores de contadores.

4 - Resultados

Os resultados foram inconclusivos devido ao facto de termos um problema na geração de nós descendentes o que está a causar com que o programa tenha alguns problemas relativamente à ordem dos nós.

```
--------2: Atribute: null    Label: null   Entropy:0.47
------------4: Atribute: Weather    Label: sunny   Entropy:0.47
------------4: Atribute: Weather    Label: overcast   Entropy:0.47
----------3: Atribute: Weather    Label: rainy   Entropy:0.47
---1: Atribute: Temp    Label: 85   Entropy:-0.88
----0: Atribute: Temp    Label: 80   Entropy:-0.88
------1: Atribute: Temp    Label: 72   Entropy:-0.88
--------2: Atribute: Temp    Label: 69   Entropy:-0.88
----------3: Atribute: Temp    Label: 75   Entropy:-0.88
----0: Atribute: Humidity    Label: 86   Entropy:0.00
------1: Atribute: Humidity    Label: 65   Entropy:0.00
--------2: Atribute: Humidity    Label: 90   Entropy:0.00
----------3: Atribute: Humidity    Label: 75   Entropy:0.00
------1: Atribute: Wind    Label: FALSE   Entropy:0.39
--------2: Atribute: Wind    Label: TRUE   Entropy:0.39
```

```
------------------------------------------------------------------------33: Atr
---1: Atribute: sepalwidth    Label: 3.5   Entropy:0.79
----0: Atribute: sepalwidth    Label: 3.8   Entropy:0.79
------1: Atribute: sepalwidth    Label: 3.7   Entropy:0.79
--------2: Atribute: sepalwidth    Label: 3.3   Entropy:0.79
----------3: Atribute: sepalwidth    Label: 3.4   Entropy:0.79
------------4: Atribute: sepalwidth    Label: 2.7   Entropy:0.79
--------------5: Atribute: sepalwidth    Label: 2.5   Entropy:0.79
----------------6: Atribute: sepalwidth    Label: 3.2   Entropy:0.79
------------------7: Atribute: sepalwidth    Label: 2.8   Entropy:0.79
--------------------8: Atribute: sepalwidth    Label: 3.1   Entropy:0.79
----------------------9: Atribute: sepalwidth    Label: 3.0   Entropy:0.79
----0: Atribute: petallength    Label: 1.4   Entropy:0.98
------1: Atribute: petallength    Label: 1.5   Entropy:0.98
--------2: Atribute: petallength    Label: 4.9   Entropy:0.98
----------3: Atribute: petallength    Label: 3.3   Entropy:0.98
------------4: Atribute: petallength    Label: 4.5   Entropy:0.98
```

Podemos ver nas figuras output's dos ficheiros weather.csv e iris.csv e com eles conseguimos concluir que os nós estão a ser mal gerados no que toca a ordem e que provavelmente temos um problema de manipulação de data das tabelas que nos está a causar problemas no cálculo do algoritmo.

5 - Conclusões

Apesar de não termos conseguido terminar o trabalho, conseguimos ter uma ideia de como funciona todo o processo de criação de árvores de decisão, a implementação dos algoritmos e sobretudo alguma experiência na linguagem. Concluímos que foram adquiridas capacidades e conhecimento no que toca na implementação e uso de árvores de decisão.

6 - Referências

https://courses.cs.washington.edu/courses/cse415/98wi/id3/id3.html

http://www.dcc.fc.up.pt/~ines/aulas/1718/IA/learning.pdf

https://www.linkedin.com/pulse/how-does-id3-algorithm-works-decision-trees-sagarnil-das

https://www.slideshare.net/HARDIKSINGH7/id3-algorithm-56632554

http://web.arch.usyd.edu.au/~wpeng/DecisionTree2.pdf

https://en.wikipedia.org/wiki/Decision_tree#Overview