Decision Trees

Trabalho Realizado Por:

Miguel Proença Fernandes Ramalho Amaro, up202106985

Miguel Filipe Miranda dos Santos, up202105289

Nuno dos Santos Moreira, up202104873

Índice

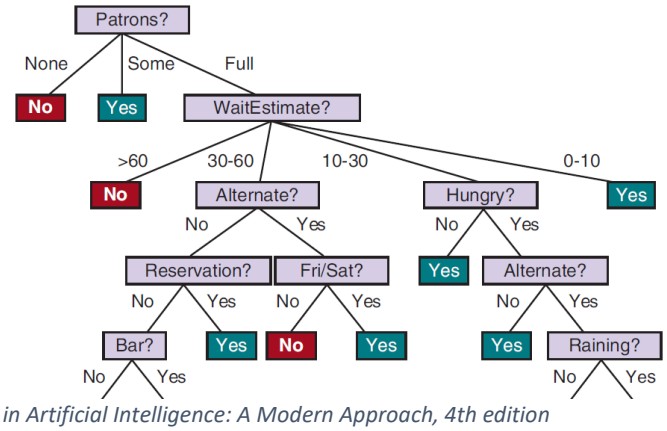
Introdução

Neste trabalho, visamos implementar um algoritmo para indução de *decision trees*. O conceito de *decision tree*, embora não muito complexo, necessita uma contextualização de modo a ser totalmente compreendido. Passaremos, então, a essa contextualização, e definição do conceito e propósito de *decision trees*.

Diz-se *machine learning* (em português, aprendizagem computacional) quando um computador observa dados, constrói um modelo baseado nesses dados, e utiliza esse modelo tanto como uma hipótese relativa ao mundo, como um software capaz de resolver problemas. Esta aprendizagem é possível através de várias formas, sendo uma delas *supervised learning* (em português, aprendizagem supervisionada), na qual o computador "aprende" através do fornecimento dos dados na sua totalidade, tendo acesso ao input e ao output.

Tendo estas definições esclarecidas, podemos afirmar que d*ecision trees* nada mais é do que um algoritmo que se insere na categoria *supervised learning* de aprendizagem computacional. Este algoritmo funciona através do mapeamento dos atributos do input de modo a obter um único valor output (uma decisão, obtida através da realização de testes com as informações fornecidas). É comumente utilizado.

Na imagem abaixo temos um exemplo de uma *decision tree* utilizada para decidir se alguém deve esperar, ou não, por uma mesa num restaurante.



in Artificial Intelligence: A Modern Approach, 4th edition

Algoritmos para *Decision Tree Induction*

A forma como a árvore é construída depende do algoritmo escolhido para esta função. Estes algoritmos variam um dos outros de acordo com o tipo de dados fornecidos, as métricas que utilizam para dividir os sets e o peso que lhes atribuem. Temos, como exemplo, as seguintes métricas:

- *Estimate of Positive Correctness* – calculado através do *confusion matrix*, resultando da subtração dos falsos positivos aos verdadeiros positivos;
- *Gini's Impurity Index* – mede quantas vezes um elemento aleatoriamente selecionado de um set é incorretamente classificado, se fosse classificado aleatoriamente (tem valor zero quando todos os elementos se encontram todos na messa classe);
- *Information Gain* – baseia-se na quantidade de informação relativa a uma variável através da observação de outra variável, estando relacionado com o conceito de entropia (menor entropia traduz-se num maior *information gain*);

Estas diferentes métricas dão, portanto, origem aos diferentes algoritmos de *Decision Tree Induction*. Entre os vários algoritmos existentes, iremos abordar o *Classification and Regression Tree*, *ID3* e *ID4.5*

O *Classification and Regression* Tree baseia-se na métrica *Gini's Impurity Index*. É regularmente usado devido às suas amplas vantagens, tais como o efeito insignificante de inputs outliers, a possibilidade de conjugação com outros algoritmos para escolher o input set das variáveis, entre outras. Infelizmente, e como todos os outros algoritmos, o CART também tem algumas limitações, como overfitting e alta variância.

O *ID3* utiliza o cálculo da entropia (ou de *information gain*) para separar os sets em subsets, isto é, seleciona o atributo com menor entropia e parte o set de acordo com o atributo escolhido. É muito utilizado em áreas como aprendizagem computacional e processamento de linguagem natural, uma vez que conta com vantagens como a criação de árvores curtas em pouco tempo, uso de regras criadas a partir dos dados de treino, e outras. Todavia, este algoritmo sofre de problemas de overfitting, apenas testa um elemento de cada vez para fazer uma decisão, pode ser computacionalmente dispendioso em problemas de dados contínuos.

O *ID4.5* é o sucessor do ID3. O funcionamento permanece o mesmo, mas houve melhorias tais como a possibilidade de utilizar elementos com atributos em falta, a criação de um limite que permite separar valores contínuos, a remoção de ramos posterior à construção da árvore.

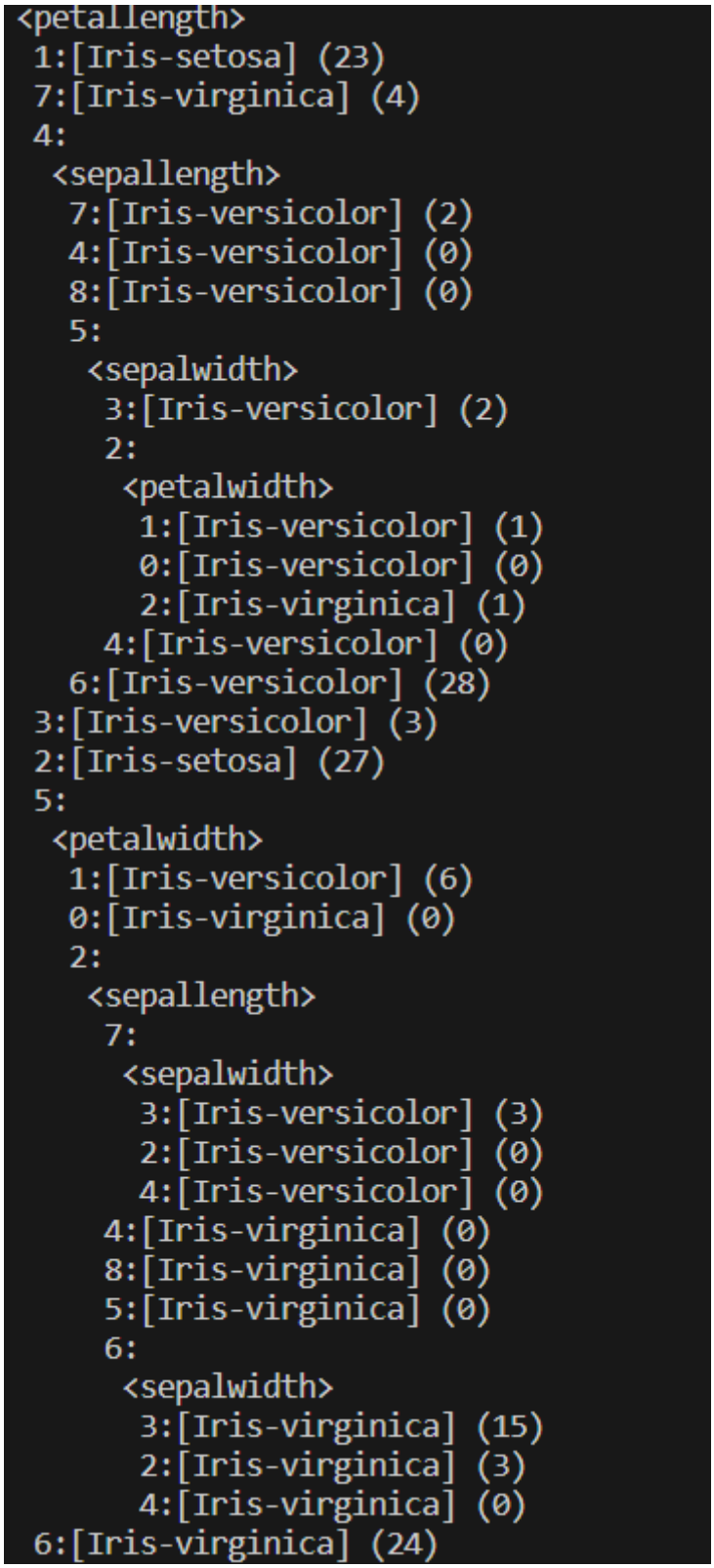
Implementação

Para fazer este trabalho foi utilizado a linguagem python, dado á nossa familiaridade com a língua.

As estruturas que utilizamos foram listas, para guardar os valores dos datasets e sets,para guardas os valores únicos de cada atributo.

O nosso código tem duas classes, Node e Tree, que servem para representar os nós da arvore e a arvore de decisão respetivamente.

Resultados



Iris data set

```
<Temp>
 71:[no] (1)
 70:[yes] (1)
 64:[yes] (1)
 72:
  <Weather>
   sunny:[no] (1)
   overcast:[yes] (1)
   rainy:[yes] (0)
 68:[yes] (1)
 65:[no] (1)
 75:[yes] (2)
 83:[yes] (1)
 81:[yes] (1)
 80:[no] (1)
 85:[no] (1)
 69:[yes] (1)
```

Weather dataset

```
<Pat>
 None:[No] (2)
 Full:
  <Hun>
   No:[No] (2)
   Yes:
    <Type>
     French:[No] (0)
     Italian:[No] (1)
     Thai:
      <Fri>
       No:[No] (1)
       Yes:[Yes] (1)
     Burger:[Yes] (1)
 Some:[Yes] (4)
```

Restaurant dataset

Comentários e Conclusões

Podemos observar que a arvore de decisão ID3 tende a escolher atributos que taem o maior número de valores diferentes, fazendo com que só tenham um exemplo por folha.

Podemos também ver que ao utilizar a arvore de decisão para ajudar a escolher o próximo movimento fez com o algoritmo Monte Carlo jogasse pior comparativamente a maneira utilizada antes.

Bibliografia

https://moodle.up.pt/pluginfile.php/194243/mod_resource/content/1/ebin.pub_artificial-intelligence-a-modern-approach-global-edition-4nbsped-9780134610993-1292401133-9781292401133-9781292401171.pdf

https://en.wikipedia.org/wiki/Decision_tree_learning

https://en.wikipedia.org/wiki/ID3_algorithm

https://en.wikipedia.org/wiki/C4.5_algorithm

https://en.wikipedia.org/wiki/Predictive_analytics#Classification_and_regression_trees_(CART)

https://athena.ecs.csus.edu/~mei/177/ID3_Algorithm.pdf