# Relatório do projeto 2 de Inteligência Artificial

# Árvores de Decisão

Ricardo Nunes 71015

07 de Dezembro de 2020

## 1. Motivação

Neste segundo projeto introduzimos o conceito de aprendizagem supervisionada e uma das formas mais simples e eficazes de aprendizagem, árvores de decisão.

Uma arvore de decisão é uma função que para um dado input de atributos , respetivos valores dos atributos e classificação binária desse atributos, realiza um conjunto de testes sobre os atributos de maneira a que o resultado dessa função seja uma representação de arvore concisa e tão pequena quanto possível.

Para além de ser uma ferramenta útil e rápida para o processo de aprendizagem a representação final da árvore de decisão é , na maior parte dos casos, *user friendly* a sua leitura representação é de fácil leitura.

## 2. Algoritmo desenvolvido

Tal como foi dito no ponto anterior a  o algoritmo para criação de árvores de decisão é uma função que tem como argumentos os atributos, os respetivos valores dos atributos e uma classificação para o conjunto de valores de todos os atributos, a partir de agora iremos chamar de exemplo ao par (X, Y) onde **X** é o conjunto de valores de todos os atributos e **Y** é a classificação para esse conjunto.

O principal objetivo do algoritmo é retornar uma árvore tao pequena quanto possível apenas tendo em conta os argumentos de entrada da função mencionada em cima, para tal o algoritmo desenvolvido divide o problema original em vários sub problemas mais simples até chegar a um de três tipos de caso de paragem:

- Se a classificação para todos os exemplos for a mesma podemos assumir uma folha da arvore com essa classificação.
- Se não há mais exemplos a analisar então retornamos a maioria da classificação para os exemplos da iteração anterior, em caso de empate é escolhido aleatoriamente um dos valor.
- Se não há mais atributos a testar retornamos a maioria dos exemplos existentes nesta iteração , em caso de empate é escolhido aleatoriamente um dos valor.

Caso não se verifique nenhum dos casos mencionados acima temos de recorrer a divisão do problema segundo o atributo que garanta uma maioria de classificação classificação ou tão próximo quanto possivel.

Na escolha  atributo deve ser escolhido segundo uma função de importância a que chamamos ganho. Considerando **entropia** a quantidade de incerteza  Esta função calcula para cada um dos atributos a diferença entre  entropia do conjunto de treino atual e o conjunto de treino após a separação por esse atributo. O atributo que apresentar maior diferença é o atributo que irá dar mais **ganho** e será o escolhido para dividir o problema original em sub problemas.

A função de ganho é bastante relevante na criação de árvores de decisão simples, tomando o meu exemplo, nas primeiras versões do meu algoritmo por lapso estava a escolher o atributo por ordem numérica em vez de escolher o com maior ganho o resultado, tal como é possível ver na figura 1 e figura 2 em anexo, é uma diferença enorme no tamanho da árvore gerada para os mesmos conjuntos de treino.

3. *Overfitting* e *noise*

Apesar do algoritmo tentar criar uma árvore simples em certas ocasiões pode acontecer que para um dado conjunto de teste a árvore de decisão não consiga chegar imediatamente a um dado padrão, a este problema chamamos overfitting.

Quando tal acontece é necessário efetuar uma poda na árvore de decisão, ou seja ,é necessário remover nós que não são relevantes para a tomada da decisão.

Existem dois tipos de poda de árvores: *pre-pruning* e *post-pruning*  tal como o nome indica estes dois tipos de poda acontecem em alturas diferentes da construção da árvore.

O *pre-pruning* ou paragem antecipada é efetuado durante a construção da árvore onde evitamos desenvolver nós que tenham poucos atributos associados. Apesar de ser uma hipótese o *pre-pruning*  pode introduzir problemas ao não efetuar a expansão total da árvore de decisão e assim não conseguimos detetar maus atributos para a decisão.

O *post-pruning* oferece uma solução para o problema mencionado anteriormente, nesta metodologia a árvore é gerada na sua totalidade e assim sim vamos verificar se existe nós que podem ser podados, alguns dos métodos de *post-pruning* são o teste estatístico *chi squared* que para um dado conjunto de testes consegue validar se o atributo é relevante comparando com a tabela estatística *chi squared* o método que escolhi implementar foi o metodo *REP* (*reduced error pruning)* embora não tenha concluído o desenvolvimento a principal característica deste algoritmo é uma poda *bottom-up*, em que se testa os valores desde as folhas da árvore até a raiz e em cada teste faz a poda de uma sub arvore e verifica se a nova árvore tem um erro menor ou igual à árvore original em caso positivo então a poda é efetuada e só acaba quando o erro da árvore podada começa a ser superior ao da árvore original.

Não foram feitos desenvolvimentos sobre o ruido(noise) mas a ideia principal seria em caso de haver ruido deve ser escolhido o valor maioritário da classificação do conjunto de exemplos ou uma estimativa probabilística para a classificação

Anexos:

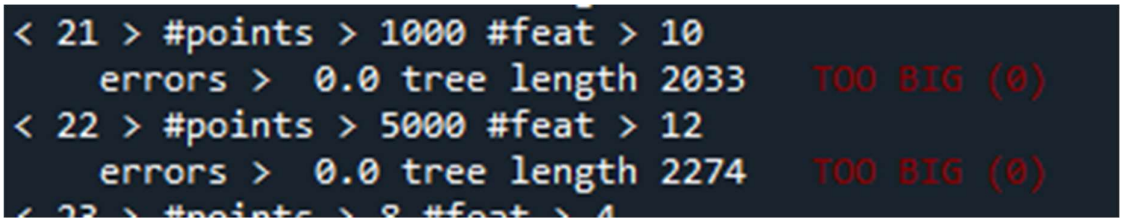


Figura 1 – Tamanho da árvore quando com um bug no algoritmo inicial.

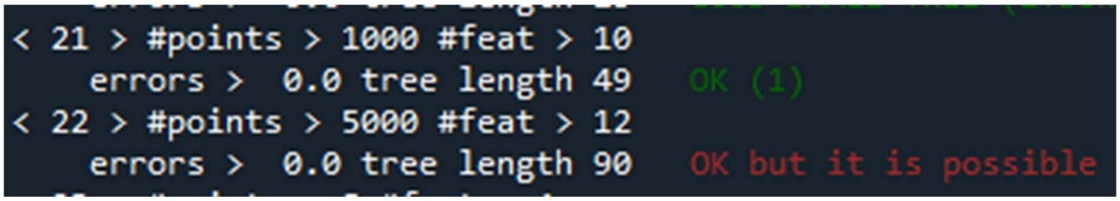


Figura 2 – Tamanho da árvore após correção do bug na escolha do atributo.