Árvores de decisão

Relatório

João Vivas - número 202108177

Gelson Stalino Varela – número 202109347

O que é uma árvore de decisão?

O tema deste trabalho é árvores de decisão. Elas representam uma função que liga um conjunto de atributos a um resultado (uma "decisão"). Para conseguirem fazer uma decisão elas fazem um conjunto de testes que começa na raiz e que vai avançando na árvore até que chegue a uma folha. Cada nó corresponde a um atributo do input e os ramos correspondem aos diferentes resultados do atributo e as folhas correspondem ao resultado da função.

As árvores de decisão são uma forma de prever possíveis consequências, resultados ou custos sobre cenários específicos.

Algoritmo usado

O algoritmo usado no nosso trabalho foi seguido o algoritmo usado no livro "Artificial Intelligence a Modern Approach", onde usa uma estratégia greedy dividir para conquistar, onde primeiro escolhe o atributo com mais importância e resolve-o e depois vai recursivamente, escolher os outros atributos por ordem decrescente de importância. O atributo mais importante é aquele que permite chegar às repostas corretas com o menor número de testes. Para escolhermos o atributo com mais importância nós usamos a entropia, que mede a incerteza de uma variável.

Implementação

Este projeto foi feito em Python e está dividido em quatro ficheiros ***main.py***, ***util.py***, ***tree.py***, ***decisiontree.py***. O ficheiro executável é o ficheiro ***main.py***. Este ficheiro começa por chamar ler o ficheiro de input com a função *readInput* do ficheiro *util.py*.

Esta função começa por executar uma função auxiliar *read_input* que vai ser a que vai abrir o ficheiro e ler o conteúdo para ser processado na árvore. Primeiro pede que o utilizador insira o nome do ficheiro de input. O programa tenta então abrir o programa com o nome que o utilizador inseriu e lê a primeira linha que vai ter os nomes dos atributos e coloca-os numa lista chamada *attribs*. Como o primeiro atributo normalmente é um ID que não é necessário então descarta-se esse atributo e como o último é a classificação do resultado do exemplo colocamos o nome do último atributo numa variável *classification*. Para cada atributo em *attribs* é colocado como chave num dicionário *attributes* com um conjunto de elementos que por agora vai estar vazio. A função lê então as restantes linhas uma a uma, sempre ignorando a primeira coluna que vai ser para o ID que não é necessário para a árvore. Para cada linha é criado um dicionário chamado *example* que vai guardar os dados do exemplo cuja chave é o atributo. Cada valor na linha é colocado como elemento no dicionário *example* e no dicionário *attributes* e a chave nestes dois dicionários é o atributo correspondente, quando for o último elemento da linha, ou seja, a classificação do exemplo também se coloca a frequência do resultado no dicionário *output_values_frequency* sendo a chave a classificação e o valor da chava a frequência. Quando já se leu o ficheiro todo é retornado quatro variáveis *examples* que vai ser uma lista de dicionários com os dados de cada linha, *attributes* que vai ser um dicionário com todos os resultados presentes no ficheiro de teste para cada atributo, *total_examples_in_dataset* o número total de exemplos no ficheiro e *probability_list* que é uma lista das probabilidades das possíveis classificações que um exemplo tem.

Com este dados então para diminuir o tamanho da árvore de decisão na função *readInput* vai discretizar os valores numéricos. Vai primeiro chamar *define_intervals* que vai criar o número de intervalos necessários usando a **fórmula de Sturge** e definir quais os valores de cada intervalo. Depois chama *discretize_variable*. Que apenas insere a variável no intervalo correspondente. E depois os atributos numéricos em vez de ter vários valores diferentes vão ter apenas intervalos numéricos. E assim já temos todos os dados para criar a árvore de decisão de forma adequada.

Então agora o programa cria a árvore de decisão com os dados do input usando a estrutura DecisionTree definida em *decisiontree.py*. A estrutura de dados da árvore de decisão **(continua...)**