Inteligência Artificial
Relatório 3 - Árvores de Decisão

Grupo 8

Luís Pinto, nº 201704025
Sónia Almeida, nº 201811293
Miguel Lançóis, nº 201506342

30 de Abril de 2019

Conteúdo

1 Introdução

Neste trabalho, vamos explorar a criação de árvores de decisão com base em algorítmos estudados em aula. Estas árvores são geradas através de ditos algorítmos e conjuntos de dados obtidos previamente.

NOTA: Sempre que aparecerem parênteses retos em frente a alguma palavra, é uma referência à bibliografia/webgrafia onde os números dentro dos parênteses representam os links na secção 6.

Árvores de Decisão

- **O que são?**
  Estes grafos são árvores que nos permitem determinar o resultado final de um dado conjunto de atributos. Cada nó, que não uma folha, representa um atributo ao qual estão associadas arestas que representam o valor associado a certo atributo. Se o nosso problema fosse gerar uma árvore de decisão para lançamento de moedas, o nó raiz seria o atributo "Qual face saíu?", as arestas "Cara/Coroa"e os nós filhos "True/False", dependendo do resultado final pretendido, por exemplo.

- **Para que servem?**
  Com árvores de decisão, torna-se possível organizar informação de forma mais compacta e eficaz para evitar ter em consideração todos os pequenos aspetos ao tomar decisões. Se o problema em questão for descobrir quais as melhores características de uma galinha para obter ovos mais saborosos, podemos arranjar, por exemplo, um conjunto de dados com os atributos: Côr das penas, Côr dos olhos, Tamanho, Comida. Com estes dados, podemos gerar uma árvore de decisão que nos permitirá escolher as melhores galinhas para obter ovos mais saborosos, por exemplo.

2 Algorítmos para árvores de decisão

Existem vários algoritmos que permitem gerar árvores de decisão a partir de uma série de dados. Existem dois tipos de árvores de decisão:

- As árvores de classificação (= classification trees)
- As árvores de regressão (= regression trees)

As **árvores de classificação** permitem prever a classe da variável desejada: é prevista a classe da variável. Para este tipo de árvore, queremos maximizar a homogeneidade de um subconjunto construido a partir de um conjunto maior. Existem várias formas de medir esta homogeneidade, as mais famosas sendo a entropia de Shannon e o critério de Gini. Estas duas métricas são usadas, respetivamente, nos algoritmos ID3 (Iterative Dichotomiser 3) e CART (Classification And Regression Tree).

As **árvores de regressão**, elas, permitem prever uma quantidade real como um tempo de espera por exemplo ou um preço: neste caso, é previsto um valor numérico. Neste caso, queremos maximizar a variância entre classes para que os valores sejam as mais distantes possível.

Os algoritmos que vão ser apresentados no que segue são os seguintes: o ID3 (Iterative Dichotomiser 3), o CART (Classification And Regression Tree), o MARS (Multivariate adaptive regression spline) e o CHAID (CHi-squared Automatic Interaction Detector) embora existam dezenas de algoritmos para árvores de decisão. [4 e 6]

2.1 ID3

O ID3 usa a função de entropia de Shannon para construir a sua árvore. Em termodinâmica, a entropia designa uma grandeza extensiva, ou seja, que é proporcional a uma grandeza que caracteriza um sistema, como a sua massa por exemplo, ao contrário de uma grandeza intensiva que não depende do tamanho do sistema, como a massa volúmica. A entropia caracteriza a desordem de um sistema. Quanto mais entropia, maior a desordem. Assim, um sistema ordenado tem um valor de entropia nulo. Em informática, a entropia de Shannon caracteriza a falta de informação de um sistema. Quanto menos informação, maior o valor de entropia. Este valor de entropia, para uma variável aleatória V com valores $v_k, k \in 1, \ldots, n$, com probabilidade $P(v_k)$, é calculado da seguinte maneira: [7]

$$H(V) = \sum_{k=1}^{n} P(v_k) \log_2 \frac{1}{P(v_k)} = -\sum_{k=1}^{n} P(v_k) \log_2 P(v_k) \text{ e } H(V) \in [0, 1]$$

A partir da entropia, é avaliado o ganho de informação, que é calculado da seguinte forma: determina-se a falta de informação, entre 0 e 1, e substrai-se a entropia. Se a falta de informação for total, o valor de ganho de informação é 1 - entropia. Isto acontece quando o atributo não nos traz informação nenhuma. Isto acontece com o atributo Class no restaurante, que contém 6 Yes e 6 No: não nos traz informação nenhuma. Este algoritmo é baseado sobre um conjunto de classes iniciais. Para que o modelo seja geral que chegue, devemos ter cuidado para que essas classes não influenciem de mais o modelo: queremos evitar overfitting e ter um modelo muito refinado para os dados que temos. Nesse caso, o modelo é demasiado específico e não é válido para outro tipo de dados. [7] Todavia, queremos obter um modelo genérico e não demasiado específico. Outra limitação do ID3 é que só é usado um atributo de cada vez para tomar uma decisão. Classificar dados contínuos pode ser computacionalmente caro. [5]

2.2 Cart

O CART usa o índice de Gini para classificar os seus dados. O índice Gini é muito utilizado para caracterizar as inequalidades dentro de uma população. Este índice varia entre 0 e 1. Se o índice tiver valor 0, a igualdade é perfeita, se tiver valor 1, a inequalidade é inteira (por exemplo, só uma pessoa é que possui as riquezas todas). Em informática, este valor é calculado multiplicando o valor da probabilidade que um elemento seja escolhido com a probabilidade que este elemento esteja mal classificado. [4]

2.3 Mars

Para agrupar os dados, é calculada uma soma ponderada de regressões lineares. [3]

2.4 Chaid

Este algoritmo é muito utilizado no domínio de marketing para selecionar um grupo de consumidores e prever as consequências das suas escolhas sobre outras variáveis. [2] É baseado sobre o teste de Bonferroni [1], mais especificamente sobre o ajusto do critério de rejeição controlando a probabilidade de ocorrer um certo erro.

3 Implementação

Linguagem: Para a implementação do código decidimos utilizar a linguagem *Python 3.7*.

Estrutura de Dados: Na implementação do nosso código, utilizamos dicionários e listas para manipulação da informação. Ambas estas estruturas de dados são de fácil manipulação e essa foi a razão principal para as termos escolhido ao invés de outras estruturas. Os "dataset", ficheiros em formato csv, são lidos e convertidos para um dicionário, utilizando o "ID"de cada linha como chave para o seu conteúdo. De seguida, construímos um dicionário com todos os valores para cada atributo presente no "dataset"que servirá de guia para a construção da árvore de decisão. São utilizadas listas quando se pretende analisar o conteúdo de um ou dois atributos isoladamente.

Para armazenamento da árvore de decisão é utilizada a classe "TreeNode"que é constituida pelo atributo *att*, com o valor do nó correspondente, e por um dicionário *children* onde, para cada chave correspondente ao valor do atributo, se irá referenciar o nó seguinte da árvore.

Organização do código: O código inicia-se com uma chamada da função *main()* onde se faz a leitura do ficheiro csv, *get_dictionary()*, e, de seguida, obtém-se o dicionário com os valores de cada atributo com a função *branching()*. Com os valores obtidos é, finalmente, chamado o algoritmo ID3 que irá retornar a árvore de decisão.

Para a implementação do ID3 foi necessário a implementação de várias funções auxiliares, tais como:

- *is_all_the_same_value()*: permite indicar se todos os exemplos restantes apresentam o mesmo valor da classe final;
- *most_common_value()*: retorna o valor mais comum da classe final;
- *get_next_att()*: calculando a entropia dos valores do atributo, retorna o atributo que deverá ser escolhido de seguida para a árvore.

4 Resultados

4.1 Restaurant

```
runfile('D:/FEUP/IA/TP/Trabalho_3/tp3(3).py', wdir='D:/FEUP/IA/TP/Trabalho_3')
Please insert the name of the file with the data: restaurant.csv
time_make_dic =  0.0
time_dic = 5.41792106628418
time_parent_branching = 0.0
time_ent = 0.0
time_branching = 0.0
time_ID3 = 0.0
 <Pat>
        Full:
                <Price>
                        $:
                                <Rain>
                                        No:
                                                <Bar>
                                                        No:
                                                                <Fri>
```

```
                                                                    No: No (1)
                                                                    Yes: Yes (1)
                                                        Yes: Yes (1)
                                            Yes: No (1)
                        $$: No (4)
                        $$$: No (2)
        None: No (2)
        Some: Yes (4)
time_print = 0.0
```

4.2 Weather

```
runfile('D:/FEUP/IA/TP/Trabalho_3/tp3(3).py', wdir='D:/FEUP/IA/TP/Trabalho_3')
Please insert the name of the file with the data: weather.csv
time_make_dic =  0.0
time_dic = 4.520612716674805
time_parent_branching = 0.0
time_ent = 0.0
time_branching = 0.0
time_ID3 = 0.0
 <Temp>
        (64.0, 69.0):
                <Weather>
                        overcast: yes (1)
                        rainy:
                                <Humidity>
                                        (65.0, 70.0): no (1)
                                        (70.0, 80.0): no (1)
                                        (80.0, 90.0): yes (1)
                                        (90.0, 95.0): no (1)
                                        (95.0, 97.0): no (1)
                        sunny: yes (2)
        (69.0, 72.0):
                <Weather>
                        overcast: yes (2)
                        rainy:
                                <Humidity>
                                        (65.0, 70.0): no (1)
                                        (70.0, 80.0): no (1)
                                        (80.0, 90.0): no (1)
                                        (90.0, 95.0): no (1)
                                        (95.0, 97.0): yes (1)
                        sunny: yes (1)
        (72.0, 75.0):
                <Weather>
                        overcast: yes (1)
                        rainy: no (1)
                        sunny: no (1)
        (75.0, 83.0):
                <Humidity>
                        (65.0, 70.0): yes (3)
                        (70.0, 80.0): yes (2)
                        (80.0, 90.0): yes (1)
                        (90.0, 95.0): no (1)
                        (95.0, 97.0): yes (3)
```

```
        (83.0, 86.0):
                <Weather>
                        overcast: yes (1)
                        rainy: no (1)
                        sunny: no (1)
time_print = 0.0
```

4.3 Iris

```
runfile('D:/FEUP/IA/TP/Trabalho_3/tp3(3).py', wdir='D:/FEUP/IA/TP/Trabalho_3')
Please insert the name of the file with the data: iris.csv
time_make_dic =  0.0
time_dic = 4.952691316604614
time_parent_branching = 0.0
time_ent = 0.0
time_branching = 0.0
time_ID3 = 0.006543159484863281
 <petallength>
        (1.0, 1.5): Iris-setosa (23)
        (1.5, 3.9):
                <sepallength>
                        (4.3, 5.0):
                                <sepalwidth>
                                        (2.0, 2.7): Iris-versicolor (1)
                                        (2.7, 3.0): Iris-setosa (8)
                                        (3.0, 3.1): Iris-setosa (8)
                                        (3.1, 3.4): Iris-setosa (6)
                                        (3.4, 5.4): Iris-setosa (2)
                        (5.0, 5.6):
                                <sepalwidth>
                                        (2.0, 2.7): Iris-versicolor (5)
                                        (2.7, 3.0): Iris-setosa (17)
                                        (3.0, 3.1): Iris-setosa (1)
                                        (3.1, 3.4): Iris-setosa (1)
                                        (3.4, 5.4): Iris-setosa (15)
                        (5.6, 6.1):
                                <sepalwidth>
                                        (2.0, 2.7): Iris-versicolor (1)
                                        (2.7, 3.0): Iris-versicolor (1)
                                        (3.0, 3.1): Iris-setosa (2)
                                        (3.1, 3.4): Iris-setosa (2)
                                        (3.4, 5.4): Iris-setosa (2)
                        (6.1, 6.6): Iris-setosa (27)
                        (6.6, 8.9): Iris-setosa (27)
        (3.9, 4.7):
                <sepallength>
                        (4.3, 5.0): Iris-virginica (1)
                        (5.0, 5.6): Iris-versicolor (5)
                        (5.6, 6.1): Iris-versicolor (15)
                        (6.1, 6.6): Iris-versicolor (8)
                        (6.6, 8.9): Iris-versicolor (3)
        (4.7, 5.4):
                <sepalwidth>
                        (2.0, 2.7):
                                <sepallength>
```

```
                                        (4.3, 5.0): Iris-virginica (3)
                                        (5.0, 5.6): Iris-virginica (3)
                                        (5.6, 6.1): Iris-virginica (2)
                                        (6.1, 6.6):
                                                <petalwidth>
                                                        (0.1, 0.2): Iris-versicolor (1)
                                                        (0.2, 1.2): Iris-versicolor (1)
                                                        (1.2, 1.5): Iris-versicolor (1)
                                                        (1.5, 1.9): Iris-versicolor (1)
                                                        (1.9, 3.5): Iris-virginica (1)
                                        (6.6, 8.9): Iris-virginica (3)
                                (2.7, 3.0):
                                        <sepallength>
                                        (4.3, 5.0): Iris-virginica (8)
                                        (5.0, 5.6): Iris-virginica (8)
                                        (5.6, 6.1):
                                                <petalwidth>
                                                        (0.1, 0.2): Iris-virginica (4)
                                                        (0.2, 1.2): Iris-virginica (4)
                                                        (1.2, 1.5): Iris-virginica (4)
                                                        (1.5, 1.9): Iris-versicolor (1)
                                                        (1.9, 3.5): Iris-virginica (4)
                                        (6.1, 6.6):
                                                <petalwidth>
                                                        (0.1, 0.2): Iris-virginica (4)
                                                        (0.2, 1.2): Iris-virginica (4)
                                                        (1.2, 1.5): Iris-versicolor (2)
                                                        (1.5, 1.9): Iris-virginica (3)
                                                        (1.9, 3.5): Iris-virginica (1)
                                        (6.6, 8.9): Iris-versicolor (1)
                                (3.0, 3.1):
                                        <petalwidth>
                                        (0.1, 0.2): Iris-virginica (5)
                                        (0.2, 1.2): Iris-virginica (5)
                                        (1.2, 1.5): Iris-virginica (5)
                                        (1.5, 1.9):
                                                <sepallength>
                                                        (4.3, 5.0): Iris-virginica (3)
                                                        (5.0, 5.6): Iris-virginica (3)
                                                        (5.6, 6.1): Iris-virginica (2)
                                                        (6.1, 6.6): Iris-virginica (1)
                                                        (6.6, 8.9): Iris-versicolor (1)
                                        (1.9, 3.5): Iris-virginica (2)
                                (3.1, 3.4):
                                        <petalwidth>
                                        (0.1, 0.2): Iris-versicolor (5)
                                        (0.2, 1.2): Iris-versicolor (5)
                                        (1.2, 1.5): Iris-versicolor (1)
                                        (1.5, 1.9): Iris-versicolor (4)
                                        (1.9, 3.5): Iris-virginica (3)
                                (3.4, 5.4): Iris-virginica (19)
                (5.4, 7.9): Iris-virginica (30)
time_print = 0.0
```

5 Conclusão

Durante a realização deste trabalho, vários detalhes surgiram relativamente ao algorítmo utilizado para gerar uma árvore de decisão e, também, à árvore em si. São referidos, de seguida, alguns aspetos negativos e positivos observados neste trabalho.

Negativos:

- Ao aumentar o número de casos observados, a árvore gerada começa a demorar bastante tempo o que pode não ser muito benéfico para certas situações;
- Dados com muitos atributos diferentes, podem gerar árvores com muitos ramos o que é desvantajoso para árvores de decisão;
- Por vezes podem ser observados imensos casos e parecer que um atributo está ligado a uma dada classe mas, na verdade, pode não estar e induzir em erro.

Positivos:

- Conseguimos condensar grandes quantidades de informações em árvores de tamanhos relativamente pequenos;
- Depois de obtida a árvore, é muito fácil de verificar certas poss iveis relações entre os atributos e as classes;
- Permitem-nos focar mais em dados atributos sabendo por onde começar um dado estudo mais profundo.

6 Bibliografia/webgrafia

1. `https://en.wikipedia.org/wiki/Chi-square_automatic_interaction_detection`
2. `https://fr.wikipedia.org/wiki/CHAID`
3. `https://fr.wikipedia.org/wiki/R%C3%A9gression_multivari%C3%A9e_par_spline_adaptative`
4. `https://fr.wikipedia.org/wiki/Arbre_de_d%C3%A9cision_(apprentissage)`
5. `https://fr.slideshare.net/HARDIKSINGH7/id3-algorithm-56632554]`
6. `https://medium.com/deep-math-machine-learning-ai/chapter-4-decision-trees-algorithms-b93975f7a1f1`
7. *Artificial Intelligence, a Modern Approach*, Russell and Norvig, cap 18.