

TP1- Entropia, Informação Mútua e Codificação de Huffman

Cronograma de execução	Semana 1: pontos 1, 2, 3 e 4 . Semana 2: pontos 5, 6 . Semana 3: pontos 7 e 8 . Semana 4: pontos 9 e 10 . Semana 5: ponto 11 e relatório.
Data de entrega	Sábado 04/11/2023 - 23:59h
Formato de Entrega	Código completo e Relatório a submeter no InforEstudante
Métodos de avaliação	- Avaliação contínua em sala de aula - Avaliação do código e relatório - Defesa individual
Objetivo	Adquirir sensibilidade para as questões fundamentais de teoria de informação, em particular entropia e informação mútua, assim como as suas aplicações práticas.

Introdução

Para a execução do presente trabalho, é disponibilizado um conjunto de dados relacionados ao consumo de combustível de diferentes modelos de carros. O rendimento do combustível (**MPG**, miles per gallon) irá depender de diferentes características como a aceleração do carro (**Acceleration**), o número de cilindros do motor (**Cylinders**), o deslocamento ou volume de ar descarregado por curso do pistão (**Displacement**), os cavalos de potência (**Horsepower**), o ano de fabrico (**Model Year**), e o peso do carro (**Weight**). Toda esta informação é disponibilizada no ficheiro **CarDataset.xlsx**, contendo dados para 407 modelos de carros.

Nota: para uma melhor execução do trabalho, os dados fornecidos no ficheiro excel foram alterados em relação aos dados originais, disponibilizados pela *University of California, Irvine* (<https://archive.ics.uci.edu/dataset/9/auto+mpg>).

1. Carregar o conjunto de dados contido no ficheiro **CarDataset.xlsx**


- a. Para isto pode utilizar a função `read_excel` do módulo Pandas. Exemplo:

```
import pandas as pd
data = pd.read_excel(path+'CarDataset.xlsx')
```

- b. Construir uma matriz com os valores da tabela.
c. Construir uma lista com os nomes das variáveis contidas na tabela.
Exemplo:

```
varNames=data.columns.values.tolist()
```

2. Representar graficamente a variável **MPG** em função de cada uma das restantes variáveis, seguindo o exemplo da Figura 1.

- a. Os pares de valores deverão ser apresentados como valores discretos.
b. Todos os gráficos devem ser inseridos numa mesma figura.
c. Os eixos X e Y deverão ter os nomes das variáveis correspondentes, e cada gráfico deverá ter um título.
d.  Comente a relação de **MPG** com as restantes variáveis.

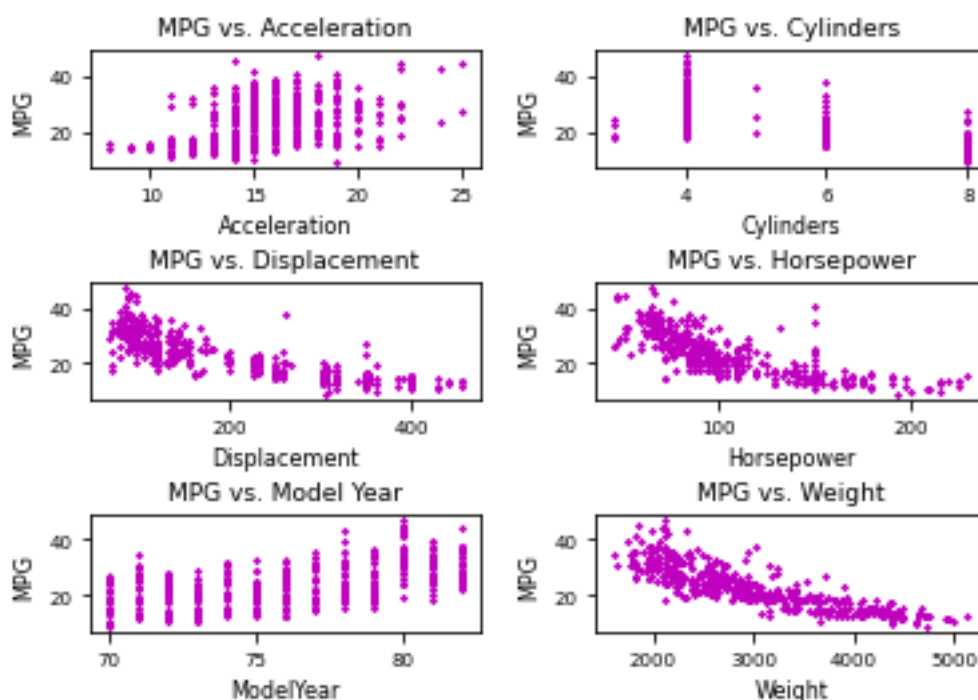


Figure 1

3. Definir um alfabeto apropriado para o conjunto de dados.
a. Converter primeiro os dados para o tipo `uint16`.
b. Definir o respetivo alfabeto.

4. Implementar uma função própria que calcule o número de ocorrências para cada símbolo do alfabeto, para cada uma das variáveis.
5. Implementar uma função que permita representar mediante um gráfico de barras o resultado obtido no ponto anterior, seguindo o exemplo da Figura 2.
 - a. O resultado para cada variável, deve ser apresentado em figuras individuais.
 - b. Representar no gráfico somente elementos do alfabeto com número de ocorrências não nulo.

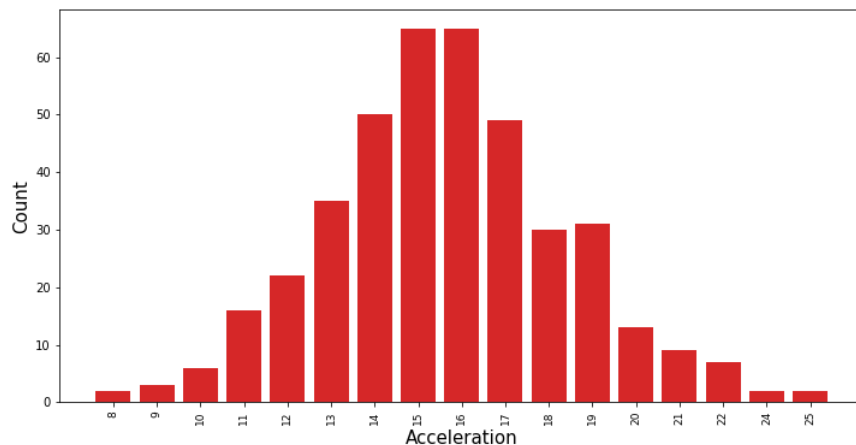


Figura 2

6. Fazer agrupamento de símbolos (*binning*) para as variáveis **Weight**, **Distance** e **Horsepower**.
 - a. Na fonte, símbolos dentro de um intervalo pré-definido, deverão assumir todos o mesmo valor. Exemplo:

$S_0 = [1, 4, 2, 8, 4, 9, 0, 3, 1, 3, 2, 7, 2, 6, 5, \dots]$

Considerando os símbolos no intervalo entre $[0, 3]$, e atribuindo a todos os elementos deste intervalo o valor 2, obtêm-se:

$S_1 = [2, 4, 2, 8, 4, 9, 2, 2, 2, 2, 2, 7, 2, 6, 5, \dots]$


Continuando com o intervalo entre $[4, 7]$, e atribuindo a todos os elementos deste intervalo o valor 4, obtêm-se:

$S_1 = [2, 4, 2, 8, 4, 9, 2, 2, 2, 2, 2, 4, 2, 4, 4, \dots]$

e o processo continua até completar o alfabeto.
 - b. A escolha do símbolo mais representativo para cada intervalo, e que irá substituir todos os elementos do intervalo, deverá ser aquele com maior número de ocorrências.
 - c. Uma vez feita a substituição, deverá repetir os pontos 4 e 5, para estas três variáveis.

- d. Para a variável **Weight**, deve considerar o agrupamento de 40 símbolos consecutivos, começando pelo primeiro elemento do alfabeto. Este parâmetro deverá ser inserido como variável de entrada da função.
- e. Para as variáveis **Distance** e **Horsepower** deverá considerar um agrupamento de 5 símbolos consecutivos, começando pelo primeiro elemento do alfabeto.



Nota: o agrupamento que deve implementar no contexto do presente trabalho, é relevante não só para efeitos de visualização, mas também para o cálculo de Informação Mútua. Muitos *bins* podem introduzir erros de estimativa para a distribuição conjunta entre as duas variáveis (perda de generalidade), enquanto poucos compartimentos podem obscurecer o relacionamento entre as duas variáveis (perda de sensibilidade). Esta abordagem não deve ser confundida com o agrupamento implementado para redução do número médio de bits por símbolo.

- 7. Implemente uma função que permita realizar o cálculo do valor médio (teórico) de bits por símbolo.
 - a. Aplicar o cálculo para cada uma das variáveis.
 - b. Aplicar o cálculo para o conjunto de dados completo.
 - c.  Comentar os resultados.
- 8. Determine o número médio de bits por símbolo utilizando codificação de Huffman, com as funções fornecidas em **huffmancodec.py**. Exemplo:


```
import huffmancodec as huffc

codec=huffc.HuffmanCodec.from_data(S)
symbols, lenghts=codec.get_code_len()
```

Onde **S** refere-se à fonte, **symbols** é o conjunto de símbolos contidos em P (que poderá não conter o alfabeto completo), e **lenghts** é o comprimento de cada um dos símbolos contidos em P.



- a. A partir dos comprimentos obtidos para cada símbolo, deverá calcular o valor médio de bits por símbolo e a variância dos comprimentos.
 - b.  Compare com os resultados obtidos no ponto 7, e comente as suas observações.
 - c.  Como se pode reduzir a variância dos comprimentos, e qual é a importância disto?
- 9. Calcular os coeficientes de correlação de Pearson entre a variável **MPG** e as restantes variáveis.
 - a. Utilize a função **corrcoef** do Numpy.

10. Implemente uma função que permita o cálculo da informação mútua (**MI**) entre a variável **MPG** e as restantes variáveis.

- a. Para as variáveis **Weight**, **Distance** e **Horsepower** considerar os dados após o agrupamento.
- b.  Comparar os resultados com aqueles obtidos no ponto anterior.

11. Os valores de **MPG** podem ser estimados em função das restantes variáveis, utilizando uma relação (simples) como a apresentada a seguir:

$$\begin{aligned} MPG_{pred} = & -5.5241 - 0.146 * Acceleration - 0.4909 * Cylinders \\ & + 0.0026 * Distance - 0.0045 * Horsepower + 0.6725 \\ & * Model - 0.0059 * Weight \end{aligned}$$

- a. Aplicar esta relação ao conjunto de dados fornecidos.
- b.  Comparar o resultado com os valores verdadeiros de **MPG**.
- c. Observe o efeito na estimativa de **MPG**, ao retirar da equação anterior o termo envolvendo a variável que apresentou o menor valor de MI.
- d. Repita o ponto c, agora retirando o termo cuja variável apresentou o maior valor de MI.
- e.  Comparar os resultados obtidos nos pontos a, c e d.