Aprendizagem - Homework 1

Pedro Curvo (ist1102716) | Salvador Torpes (ist1102474)

1º Semestre - 23/24

Dataset

Consideramos o seguinte dataset D:

D	y_1	y_2	y_3	y_4	y_{out}
x_1	0.24	1	1	0	A
x_2	0.06	2	0	0	B
x_3	0.04	0	0	0	B
x_4	0.36	0	2	1	C
x_5	0.32	0	0	2	C
x_6	0.68	2	2	1	A
x_7	0.90	0	1	2	A
x_8	0.76	2	2	0	A
x_9	0.46	1	1	1	B
x_{10}	0.62	0	0	1	B
x_{11}	0.44	1	2	2	C
x_{12}	0.52	0	2	0	C

Tabela 1: Dataset D

Exercício 1.

De modo a corretamente completar a árvore de decisão, é necessário calcular o Information gain (IG) da variável de output y_{out} condicionada a cada uma das variáveis y_2, y_3 e y_4.

Escolha do 2º nó

Como queremos completar o ramo $y_1 > 0.4$, vamos apenas considerar as ocorrencias em que $y_1 > 0.4$ para calcular o IG.

Information Gain de y_{out} condicionada a y_2

$$IG(y_{out}|y_2) = H(y_{out}) - H(y_{out}|y_2)$$

$$H(y_{out}) = \left(-\sum_{i=1}^{3} p_{out_i}(\log_2 p_{out_i})\right) = -\left(\frac{3}{7}\log_2\left(\frac{3}{7}\right) + \frac{2}{7}\log_2\left(\frac{2}{7}\right) + \frac{2}{7}\log_2\left(\frac{2}{7}\right)\right) = 1.5567$$

$$H(y_{out}|y_2) = \sum_{i=0}^{2} p_{y_2=i} H(y_{out}|y_2 = i)$$

Tabela dividida em 3 sub-tabelas, cada uma com os dados que verificam $y_2 = 0$, $y_2 = 1$ e $y_2 = 2$, respetivamente:

D	y_2	y_{out}
x_7	0	A
x_{10}	0	B
x_{12}	0	C

Tabela 2: Dataset D com $y_2 = 0$

D	y_2	y_{out}
x_9	1	B
x_{11}	1	C

Tabela 3: Dataset D com $y_2 = 1$

D	y_2	y_{out}
x_6	2	A
x_8	2	A

Tabela 4: Dataset D com $y_2 = 2$

$$H(y_{out}|y_2 = 0) = -\left(\frac{1}{3}\log_2\left(\frac{1}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right) = 1.58496$$

$$H(y_{out}|y_2 = 1) = -\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right) = 1$$

$$H(y_{out}|y_2 = 2) = -(\log(1)) = 0$$

Assim, podemos calcular a entropia de y_{out} condicionada a y_2:

$$H(y_{out}|y_2) = \frac{3}{7}H(y_{out}|y_2 = 0) + \frac{2}{7}H(y_{out}|y_2 = 1) + \frac{2}{7}H(y_{out}|y_2 = 2) =$$
$$= \frac{3}{7} \times 1.58496 + \frac{2}{7} \times 1 + \frac{2}{7} \times 0 = 0.96498$$

Por fim, podemos calcular o Information Gain:

$$IG(y_{out}|y_2) = H(y_{out}) - H(y_{out}|y_2) = 1.5567 - 0.96498 = 0.59172$$

Information Gain de y_{out} condicionada a y_3

$$IG(y_{out}|y_3) = H(y_{out}) - H(y_{out}|y_3)$$

$$H(y_{out}|y_3) = \sum_{i=0}^{2} p_{y_3=i} H(y_{out}|y_3 = i)$$

Tabela dividida em 3 sub-tabelas, cada uma com os dados que verificam $y_3 = 0$, $y_3 = 1$ e $y_3 = 2$, respetivamente:

D	y_3	y_{out}
x_{10}	0	B

Tabela 5: Dataset D com $y_3 = 0$

D	y_3	y_{out}
x_7	1	A
x_9	1	B

Tabela 6: Dataset D com $y_3 = 1$

D	y_3	y_{out}
x_6	2	A
x_8	2	A
x_{11}	2	C
x_{12}	2	C

Tabela 7: Dataset D com $y_3 = 2$

$$H(y_{out}|y_3 = 0) = -(\log(1)) = 0$$

$$H(y_{out}|y_3 = 1) = -\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right) = 1$$

$$H(y_{out}|y_3 = 2) = -\left(\frac{2}{4}\log_2\left(\frac{2}{4}\right) + \frac{2}{4}\log_2\left(\frac{2}{4}\right)\right) = 1$$

Assim, podemos calcular a entropia de y_{out} condicionada a y_3:

$$H(y_{out}|y_3) = \frac{1}{7}H(y_{out}|y_3 = 0) + \frac{2}{7}H(y_{out}|y_3 = 1) + \frac{4}{7}H(y_{out}|y_3 = 2) =$$

$$= \frac{1}{7} \times 0 + \frac{2}{7} \times 1 + \frac{4}{7} \times 1 = 0.85714$$

Por fim, podemos calcular o Information Gain:

$$IG(y_{out}|y_3) = H(y_{out}) - H(y_{out}|y_3) = 1.5567 - 0.85714 = 0.69956$$

Information Gain de y_{out} condicionada a y_4

$$IG(y_{out}|y_4) = H(y_{out}) - H(y_{out}|y_4)$$

$$H(y_{out}|y_4) = \sum_{i=0}^{2} p_{y_4=i} H(y_{out}|y_4 = i)$$

Tabela dividida em 3 sub-tabelas, cada uma com os dados que verificam $y_4 = 0$, $y_4 = 1$ e $y_4 = 2$, respetivamente:

D	y_4	y_{out}
x_8	0	A
x_{12}	0	C

Tabela 8: Dataset D com $y_4 = 0$

D	y_4	y_{out}
x_6	1	A
x_9	1	B
x_{10}	1	B

Tabela 9: Dataset D com $y_4 = 1$

D	y_4	y_{out}
x_7	2	A
x_{11}	2	C

Tabela 10: Dataset D com $y_4 = 2$

$$H(y_{out}|y_4 = 0) = -\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right) = 1$$

$$H(y_{out}|y_4 = 1) = -\left(\frac{2}{3}\log_2\left(\frac{2}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right) = 0.918295$$

$$H(y_{out}|y_4 = 2) = -\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right) = 1$$

Assim, podemos calcular a entropia de y_{out} condicionada a y_4:

$$H(y_{out}|y_4) = \frac{2}{7}H(y_{out}|y_4 = 0) + \frac{3}{7}H(y_{out}|y_4 = 1) + \frac{2}{7}H(y_{out}|y_4 = 2) =$$

$$= \frac{2}{7} \times 1 + \frac{3}{7} \times 0.918295 + \frac{2}{7} \times 1 = 0.96498$$

Por fim, podemos calcular o Information Gain:

$$IG(y_{out}|y_4) = H(y_{out}) - H(y_{out}|y_4) = 1.5567 - 0.96498 = 0.59172$$

Comparação dos IG Podemos confirmar, pelos cálculos acima, que:

$$IG(y_{out}|y_2) = IG(y_{out}|y_4) < IG(y_{out}|y_3)$$

Assim, a variável y_3 é a que tem maior IG, pelo que é a variável que escolhemos para o 2º nó da árvore de decisão no ramo $y_1 > 0.4$. Este nó vai ter três ramos, um para cada valor possível de y_3: a ocorrência $y_3 = 0$ tem apenas uma ocorrência e a ocorrência $y_3 = 1$ tem apenas duas ocorrências, pelo que estes dois nós não são expandidos. Por outro lado, $y_3 = 2$ tem 4 ocorrências, pelo que é o único nó que é expandido. Falta averiguar qual a variável que vai ser usada para expandir este nó.

Escolha do 3º nó

Queremos agora completar o ramo que verifica $y_1 > 0.4$ e $y_3 = 2$. Para isso, vamos calcular o IG de y_{out} para y_2 e y_4 considerando apenas as ocorrências que verificam $y_1 > 0.4$ e $y_3 = 2$:

Information Gain de y_{out} condicionada a y_2

$$IG(y_{out}|y_2) = H(y_{out}) - H(y_{out}|y_2)$$

$$H(y_{out}) = \left(-\sum_{i=1}^{3} p_{out_i}(\log_2 p_{out_i})\right) = -\left(\frac{2}{4}\log_2\left(\frac{2}{4}\right) + \frac{2}{4}\log_2\left(\frac{2}{4}\right)\right) = 1$$

$$H(y_{out}|y_2) = \sum_{i=0}^{2} p_{y_2=i} H(y_{out}|y_2 = i)$$

As entropias condicionadas de y_{out} para cada valor de y_2 são:

$$H(y_{out}|y_2 = 0) = -\left(\log_2(1)\right) = 0$$

$$H(y_{out}|y_2 = 1) = -\left(\log_2(1)\right) = 0$$

$$H(y_{out}|y_2 = 2) = -\left(\log_2(1)\right) = 0$$

Assim, podemos calcular a entropia de y_{out} condicionada a y_2:

$$H(y_{out}|y_2) = \frac{1}{4}H(y_{out}|y_2 = 0) + \frac{1}{4}H(y_{out}|y_2 = 1) + \frac{2}{4}H(y_{out}|y_2 = 2) =$$

$$= \frac{1}{4} \times 0 + \frac{1}{4} \times 0 + \frac{2}{4} \times 0 = 0$$

Por fim, podemos calcular o Information Gain:

$$IG(y_{out}|y_2) = H(y_{out}) - H(y_{out}|y_2) = 1 - 0 = 1$$

Information Gain de y_{out} condicionada a y_4

$$IG(y_{out}|y_4) = H(y_{out}) - H(y_{out}|y_4)$$

$$H(y_{out}|y_4) = \sum_{i=0}^{2} p_{y_4=i} H(y_{out}|y_4 = i)$$

As entropias condicionadas de y_{out} para cada valor de y_4 são:

$$H(y_{out}|y_4 = 0) = -\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right) = 1$$

$$H(y_{out}|y_4 = 1) = -\left(\log_2(1)\right) = 0$$

$$H(y_{out}|y_4 = 2) = -\left(\log_2(1)\right) = 0$$

Assim, podemos calcular a entropia de y_{out} condicionada a y_4:

$$H(y_{out}|y_4) = \frac{2}{4}H(y_{out}|y_4 = 0) + \frac{1}{4}H(y_{out}|y_4 = 1) + \frac{1}{4}H(y_{out}|y_4 = 2) =$$

$$= \frac{2}{4} \times 1 + \frac{1}{4} \times 0 + \frac{1}{4} \times 0 = 0.5$$

Por fim, podemos calcular o Information Gain:

$$IG(y_{out}|y_4) = H(y_{out}) - H(y_{out}|y_4) = 1 - 0.5 = 0.5$$

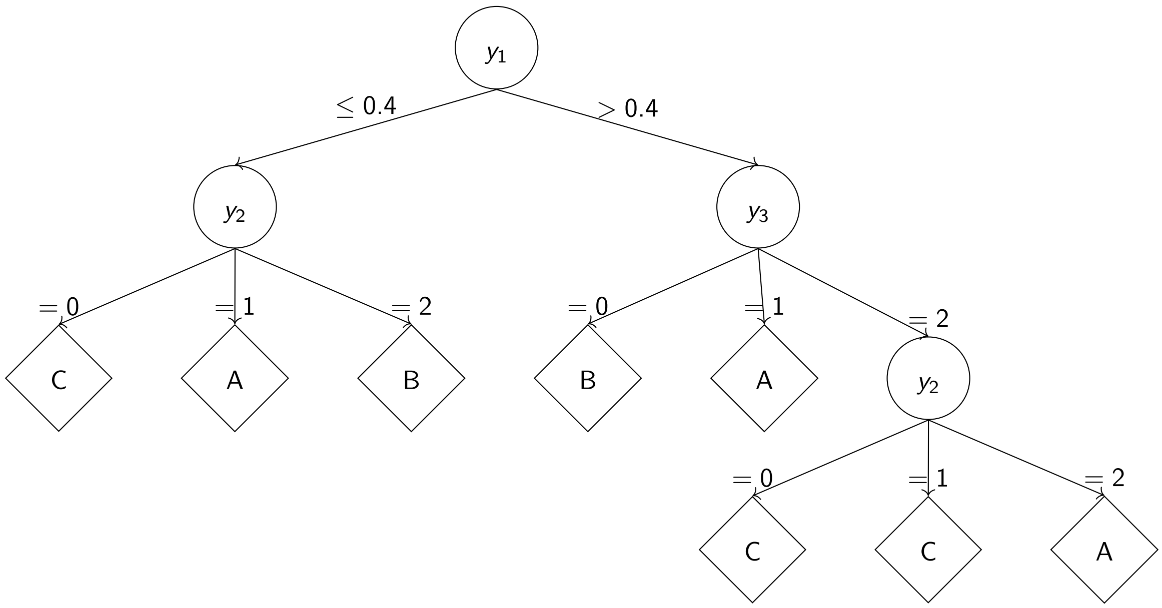
Comparação dos IG Podemos confirmar, pelos cálculos acima, que:

$$IG(y_{out}|y_2) > IG(y_{out}|y_4)$$

Assim, a variável y_2 é a que tem maior IG, pelo que é a variável que escolhemos para o 3º nó da árvore de decisão no ramo $y_1 > 0.4$ e $y_3 = 2$. Todos os nós desta árvore têm menos que 4 ocorrências, pelo que nehum deles é expandido e termina a árvore de decisão.

Construção da árvore de decisão

Para completar a árvore, resta preencher os nós terminais com os valores de y_{out} que são mais prováveis em cada ramo. Em caso de empate, escolhemos por ordem alfabética. A árvore de decisão final é:



Exercício 2.

Com o objetivo de desenhar a matriz de confusão da árvore de decisão construída acima, começamos por calcular os valores previstos para o output, $\hat{y}_{out}$, para cada uma das ocorrências do dataset D:

D	y_1	y_2	y_3	y_4	$\hat{y}_{out}$	y_{out}
x_1	0.24	1	1	0	A	A
x_2	0.06	2	0	0	B	B
x_3	0.04	0	0	0	C	B
x_4	0.36	0	2	1	C	C
x_5	0.32	0	0	2	C	C
x_6	0.68	2	2	1	A	A
x_7	0.90	0	1	2	A	A
x_8	0.76	2	2	0	A	A
x_9	0.46	1	1	1	A	B
x_{10}	0.62	0	0	1	B	B
x_{11}	0.44	1	2	2	C	C
x_{12}	0.52	0	2	0	C	C

Tabela 11: Dataset D com $\hat{y}_{out}$

Assim, desenhamos a **matriz de confusão**:

		Valores reais		
		A	B	C
Valores Previstos	A	4	1	0
	B	0	2	0
	C	0	1	4

Tabela 12: Matriz de confusão

Exercício 3.

Para calcular o F_1-score para cada uma das classes de y_{out}, começamos por calcular a precisão e o recall para cada uma delas:

A precisão é dada por:

$$P = \frac{TP}{TP + FP}$$

O recall é dado por:

$$R = \frac{TP}{TP + FN}$$

Assim, obtemos que:

$$P_A = \frac{4}{4+1+0} = \frac{4}{5} \qquad P_B = \frac{2}{2+0+0} = 1 \qquad P_C = \frac{4}{4+1+0} = \frac{4}{5}$$

$$R_A = \frac{4}{4+0+0} = 1 \qquad R_B = \frac{2}{2+1+1} = \frac{1}{2} \qquad R_C = \frac{4}{4+0+0} = 1$$

Por fim, o F_1-score é dado por:

$$F_1 = \frac{1}{0.5 \cdot \frac{1}{P} + 0.5 \cdot \frac{1}{R}}$$

Assim, podemos calcular o F_1-score para cada uma das classes:

$$F_1(A) = \frac{1}{0.5 \cdot \frac{5}{4} + 0.5 \cdot 1} = \frac{1}{\frac{5}{8} + \frac{1}{2}} = \frac{1}{\frac{5}{8} + \frac{4}{8}} = \frac{1}{\frac{9}{8}} = \frac{8}{9}$$

$$F_1(B) = \frac{1}{0.5 \cdot 1 + 0.5 \cdot 2} = \frac{1}{0.5 + 1} = \frac{1}{1.5} = \frac{2}{3}$$

$$F_1(C) = \frac{1}{0.5 \cdot \frac{5}{4} + 0.5 \cdot 1} = \frac{1}{\frac{5}{8} + \frac{1}{2}} = \frac{1}{\frac{5}{8} + \frac{4}{8}} = \frac{1}{\frac{9}{8}} = \frac{8}{9}$$

Resposta: Assim, podemos concluir que a classe com menor F_1-score é a classe B, com um F_1-score de $\frac{2}{3}$.

Exercício 4.

Para calcular o coeficiente de Spearman entre as variáveis y_1 e y_2, começamos por calcular o rank de cada uma das variáveis:

D	y_1	y_1 rank	y_2	y_2 rank
x_1	0.24	3	1	8
x_2	0.06	2	2	11
x_3	0.04	1	0	3.5
x_4	0.36	5	0	3.5
x_5	0.32	4	0	3.5
x_6	0.68	10	2	11
x_7	0.90	12	0	3.5
x_8	0.76	11	2	11
x_9	0.46	7	1	8
x_{10}	0.62	9	0	3.5
x_{11}	0.44	6	1	8
x_{12}	0.52	8	0	3.5

Tabela 13: Dataset D com ranks

A fórmula para o coeficiente de Spearman é:

$$r_S = \frac{\text{cov}(rank(y_1), rank(y_2))}{\sqrt{\text{var}(rank(y_1)) \cdot \text{var}(rank(y_2))}} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\left(\frac{1}{n}\sum_{i=1}^{n}(rank(x_i) - ran\bar{k}(x))^2\right) \cdot \left(\frac{1}{n}\sum_{i=1}^{n}(rank(y_i) - ran\bar{k}(y))^2\right)}} = 0.079659$$

Resposta: Como o coeficiente de Spearman entre as duas variáveis é $<< 1$, podemos concluir que as duas variáveis não estão correlacionadas.

Exercício 5.

Queremos desenhar os histogramas da variável y_1 condicionados aos diferentes outcomes da variável y_{out}. Assim, é necessário calcular os valores dos bins para 3 histogramas diferentes. Em primeiro lugar, utilizamos os dados:

y_{out}	y_1
A	0.24
A	0.68
A	0.90
A	0.76
B	0.06
B	0.04
B	0.46
B	0.62
C	0.36
C	0.32
C	0.44
C	0.52

Tabela 14: Dataset D com y_1 e y_{out}

Queremos que cada histograma tenha 5 bins sendo a range total $[0, 1]$, logo, os bins possíveis são $[0, 0.2]$, $[0.2, 0.4]$, $[0.4, 0.6]$, $[0.6, 0.8]$ e $[0.8, 1]$.

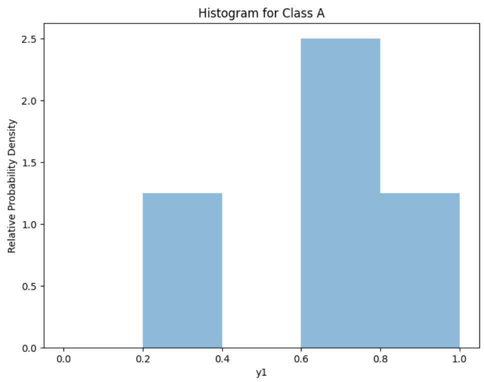
Bins	Contagens com $y_{out} = A$	Altura do bin para $y_{out} = A$	Contagens com $y_{out} = B$	Altura do bin para $y_{out} = B$	Contagens com $y_{out} = C$	Altura do bin para $y_{out} = C$	Classe pre-dominante no bin
$[0, 0.2]$	0	0	2	$\frac{2}{4 \cdot 0.2} = 2.5$	0	0	B
$[0.2, 0.4]$	1	$\frac{1}{4 \cdot 0.2} = 1.25$	0	0	2	$\frac{2}{4 \cdot 0.2} = 2.5$	C
$[0.4, 0.6]$	0	0	1	$\frac{1}{4 \cdot 0.2} = 1.25$	2	$\frac{2}{4 \cdot 0.2} = 2.5$	C
$[0.6, 0.8]$	2	$\frac{2}{4 \cdot 0.2} = 2.5$	1	$\frac{1}{4 \cdot 0.2} = 1.25$	0	0	A
$[0.8, 1]$	1	$\frac{1}{4 \cdot 0.2} = 1.25$	0	0	0	0	A

Na tabela acima encontram-se os cálculos para a altura de cada bin em cada um dos três histogramas. A altura é dada pela fórmula:

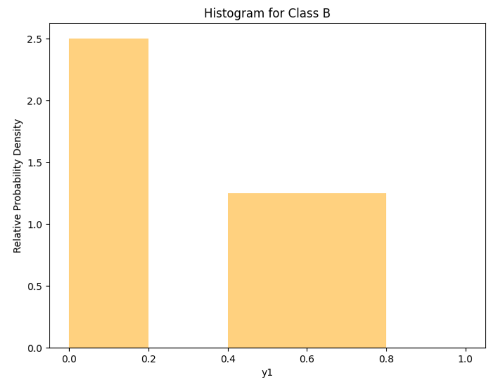
$$h = \frac{C}{n \cdot l}$$

Onde C é o número de ocorrências no bin, n é o número total de ocorrências e l é a largura do bin.

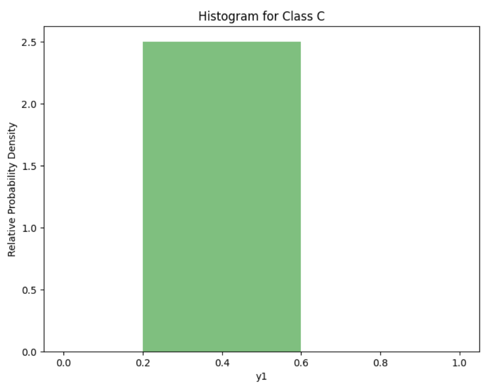
Obtivemos os seguintes histogramas da variável y_1 condicionados aos diferentes outcomes da variável y_{out}:



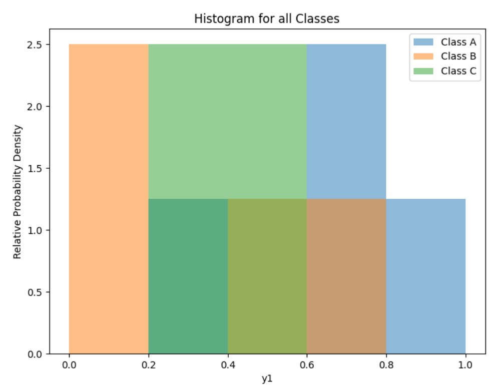
(a) Histograma de y_1 para $y_{out} = A$



(b) Histograma de y_1 para $y_{out} = B$



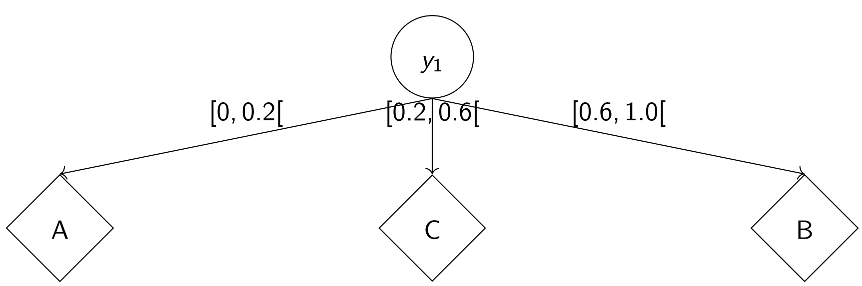
(c) Histograma de y_1 para $y_{out} = C$



(d) Histogramas de y_1 para $y_{out} = A$, $y_{out} = B$ e $y_{out} = C$

Figura 1: Histogramas de y_1 condicionados a y_{out}

Usando o critério da probabilidade máxima, podemos concluir que a classe predominante em cada bin é a classe que tem maior altura no bin, referido na tabela acima. Deste modo e usando este critério como root split, podemos desenhar a árvore de decisão:



Anteriormente, usámos a o ganho de informação com entropia para escolher a variável que iria ser usada para o root split. Contudo, outros métodos podem ser utilizados. Neste caso, podemos verificar que usando probabilidades condicionadas também conseguimos construir uma árvore de decisão. No exemplo acima, apenas usámos a variável y_1 para construir a árvore de decisão, mas poderíamos usar mais variáveis.

Componente de Programação

Imports

```
import sklearn as sk
from sklearn.feature_selection import f_classif
from sklearn.model_selection import train_test_split
from sklearn import tree
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from pathlib import Path
from scipy.io.arff import loadarff
```

Listing 1:

Dataset

```
IMAGES_DIR = Path('images')
DATA_DIR = Path('data')
DATA_FILE = 'column_diagnosis.arff'
DATA_PATH = DATA_DIR / DATA_FILE
data = loadarff(DATA_PATH)
df = pd.DataFrame(data[0])
df['class'] = df['class'].str.decode('utf-8')
# Show the first 5 rows
df.head()
```

Listing 2:

Question 1

F-Score

```
X = df.drop('class', axis=1)
y = df['class']
f_statistic, p_values = f_classif(X, y)

f_statistic_df = pd.DataFrame({'feature': X.columns,
                               'F': f_statistic,
                               'p-value': p_values})
f_statistic_df.sort_values('p-value')

highest_discriminative_feature = f_statistic_df['feature'][f_statistic_df['F'].idxmax()]
lowest_discriminative_feature = f_statistic_df['feature'][f_statistic_df['F'].idxmin()]

print(f_statistic_df.sort_values('p-value'))
print("\n")
print(f"Variable with the highest F-statistic and lowest p_value: {
  highest_discriminative_feature}\n")
print(f"Variable with the lowest F-statistic and highest p_value: {
  lowest_discriminative_feature}\n")
```

Listing 3:

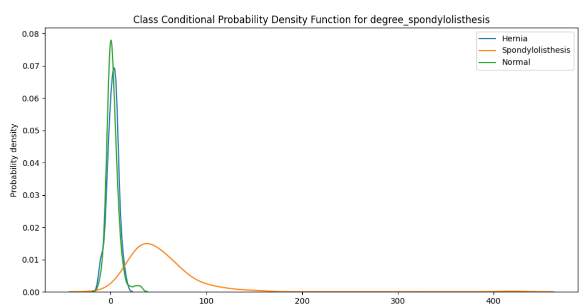
```
Variable with the highest F-statistic and lowest p_value: degree_spondylolisthesis

Variable with the lowest F-statistic and highest p_value: pelvic_radius
```

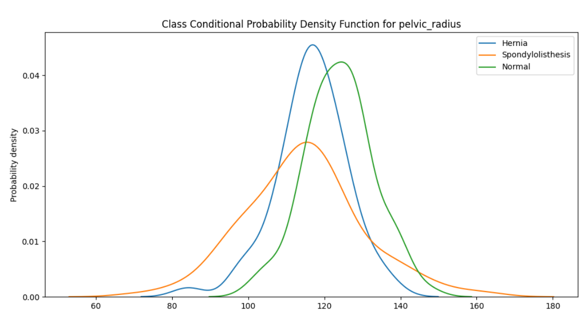
Listing 4:

Class Conditional probability density functions

```
# Plot the class-conditional probability density functions for the two selected features
plt.figure(figsize=(12, 6))

for class_name in df['class'].unique():
    sns.kdeplot(df[df['class'] == class_name][highest_discriminative_feature], label=
  class_name)

plt.title(f"Class Conditional Probability Density Function for {
  highest_discriminative_feature}")
plt.xlabel(highest_discriminative_feature)
plt.ylabel('Probability density')
plt.legend()
plt.savefig(IMAGES_DIR / 'prob_dens_highest_discriminative_feature.png')
plt.show()

plt.figure(figsize=(12, 6))
for class_name in df['class'].unique():
    sns.kdeplot(df[df['class'] == class_name][lowest_discriminative_feature], label=
  class_name)

plt.title(f"Class Conditional Probability Density Function for {
  lowest_discriminative_feature}")
plt.xlabel(lowest_discriminative_feature)
plt.ylabel('Probability density')
plt.legend()
plt.savefig(IMAGES_DIR /'prob_dens_lowest_discriminative_feature.png')
plt.show()
```

Listing 5:

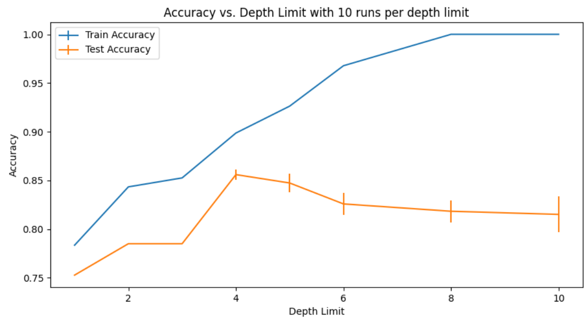


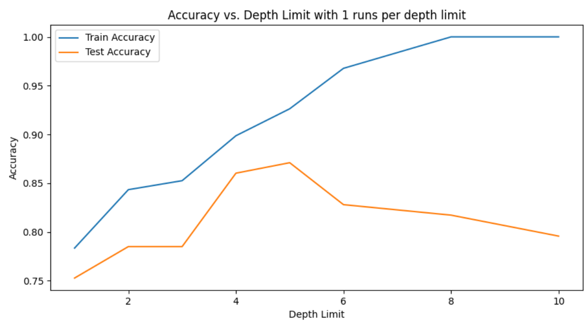


Question 2

```python
# Load and partition data
X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    train_size=0.7,
                                                    random_state=0,
                                                    stratify=y)

# Define the range of depth limits and number of runs to average over
depths = [1, 2, 3, 4, 5, 6, 8, 10]
runs = 10

# Arrays to keep Acc Values
train_acc = np.zeros((runs, len(depths))) # This is going to be a 10x8 array, because we want to keep all the accuracies for each run
test_acc = np.zeros((runs, len(depths)))

# Loop over the runs
for j, depth in enumerate(depths): # Need enumerate since we need to index the depths to keep in arrays
    for i in range(runs):
        # Learn Classifier
        predictor = tree.DecisionTreeClassifier(max_depth=depth)
        predictor.fit(X_train, y_train)

        # Predict on train and test set
        y_predicted_train = predictor.predict(X_train)
        y_predicted_test = predictor.predict(X_test)

        # Compute accuracy
        train_acc[i, j] = sk.metrics.accuracy_score(y_train, y_predicted_train)
        test_acc[i, j] = sk.metrics.accuracy_score(y_test, y_predicted_test)

# Compute mean and standard deviation for train and test accuracies
train_acc_mean = np.mean(train_acc, axis=0)
train_acc_std = np.std(train_acc, axis=0)
test_acc_mean = np.mean(test_acc, axis=0)
test_acc_std = np.std(test_acc, axis=0)

# Plot Optional
plt.figure(figsize=(10, 5))
plt.title('Accuracy vs. Depth Limit with 10 runs per depth limit')
plt.errorbar(depths, train_acc_mean, yerr=train_acc_std, label='Train Accuracy')
plt.errorbar(depths, test_acc_mean, yerr=test_acc_std, label='Test Accuracy')
plt.xlabel('Depth Limit')
plt.ylabel('Accuracy')
plt.legend()
plt.savefig(IMAGES_DIR /'accuracy_vs_depth_limit.png')
plt.show()


# Plot the Exercise without Optional Part
plt.figure(figsize=(10, 5))
plt.title('Accuracy vs. Depth Limit with 1 runs per depth limit')
plt.plot(depths, train_acc[0], label='Train Accuracy') # Selecting the first run for each depth limit
plt.plot(depths, test_acc[0], label='Test Accuracy')
plt.xlabel('Depth Limit')
plt.ylabel('Accuracy')
plt.legend()
plt.savefig(IMAGES_DIR /'accuracy_vs_depth_limit_without_optional.png')
plt.show()
```

Listing 6:





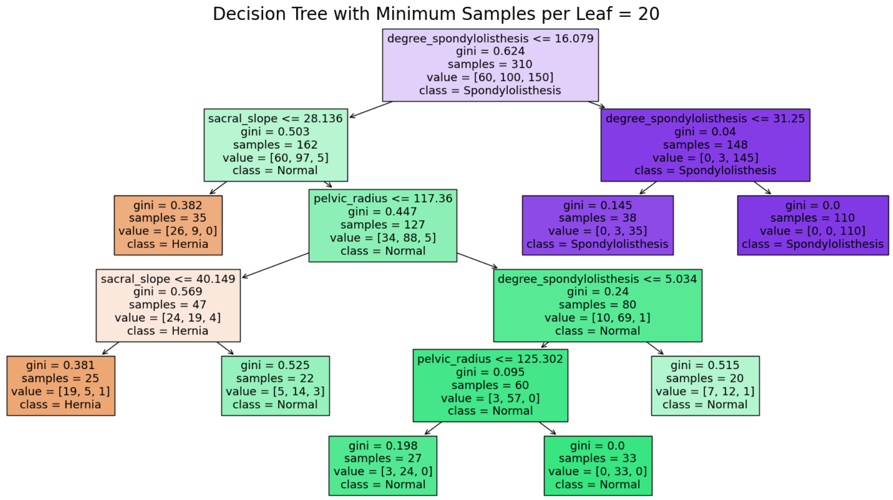
Question 3

Como se vê em ambos os gráficos, a precisão de treino aumenta à medida que o limite de profundidade da árvore de decisão aumenta. No entanto, a precisão de teste aumenta até um certo limite de profundidade (neste caso, cerca de 4 ou 5) e depois começa a diminuir. Isto acontece porque a árvore de decisão está a dar overfit aos dados de treino e não está a aprender de modo a corretamente avaliar os dados de teste. No que toca à parte opcional, também podemos ver que o desvio padrão geralmente é pequeno no caso do treino, o que indica que os resultados são consistentes em diferentes inicializações aleatórias dos dados. No entanto, o desvio padrão é maior para os dados de teste, o que indica que os dados de teste são mais sensíveis à inicialização aleatória dos dados.

Question 4

```
    # Learn a decision tree with a minimum number of samples per leaf = 20
    predictor = tree.DecisionTreeClassifier(min_samples_leaf=20,
                                            random_state=0)

    # Fit all the data
    predictor.fit(X, y)

    # Plot the tree
    plt.figure(figsize=(20, 10))
    tree.plot_tree(predictor,
                   feature_names=X.columns,
                   filled=True,
                   class_names=predictor.classes_)
    plt.title('Decision Tree with Minimum Samples per Leaf = 20', fontsize=20)
    plt.savefig(IMAGES_DIR / 'decision_tree_min_samples_leaf_20.png')
    plt.show()
```

Listing 7:



De acordo com a árvore de decisão construída na alínea anterior, existem dois casos nos quais o indivíduo em questão tem um outcome de diagnóstico com Hérnia. Em primeiro lugar temos degree spondylolisthesis ≤ 16.079 e sacral slope ≤ 28.136. Em segundo lugar, temos degree spondy—olisthesis ≤ 16.079, sacral slope > 28.136, pelvic radius ≤ 117.36 e sacral slope ≤ 40.149.