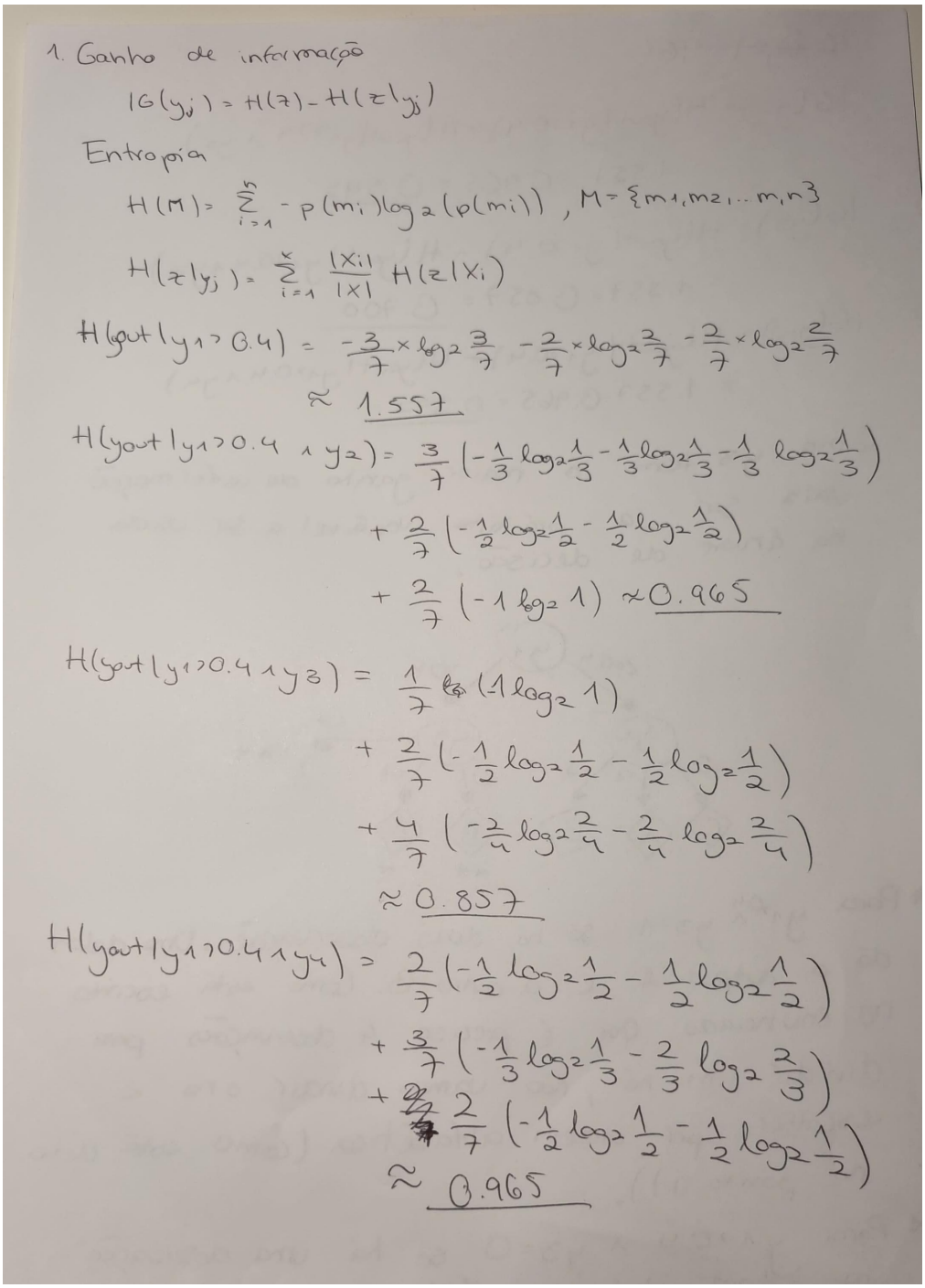
Aprendizagem 2023
Homework I – Group 036
(ist1103248, ist1103327)

Part I: Pen and paper

1.



1. Ganho de informação

$$IG(y_j) = H(Z) - H(Z|y_j)$$

Entropia

$$H(M) = \sum_{i=1}^{n} -p(m_i)\log_2(p(m_i)), \quad M = \{m_1, m_2, \ldots m_n\}$$

$$H(Z|y_j) = \sum_{i=1}^{x} \frac{|X_i|}{|X|} H(Z|X_i)$$

$$H(y_{out}|y_1 > 0.4) = -\frac{3}{7}\times\log_2\frac{3}{7} - \frac{2}{7}\times\log_2\frac{2}{7} - \frac{2}{7}\times\log_2\frac{2}{7} \approx 1.557$$

$$H(y_{out}|y_1 > 0.4 \wedge y_2) = \frac{3}{7}\left(-\frac{1}{3}\log_2\frac{1}{3} - \frac{1}{3}\log_2\frac{1}{3} - \frac{1}{3}\log_2\frac{1}{3}\right) + \frac{2}{7}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) + \frac{2}{7}\left(-1\log_2 1\right) \approx 0.965$$

$$H(y_{out}|y_1 > 0.4 \wedge y_3) = \frac{1}{7}(-1\log_2 1) + \frac{2}{7}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) + \frac{4}{7}\left(-\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4}\right) \approx 0.857$$

$$H(y_{out}|y_1 > 0.4 \wedge y_4) = \frac{2}{7}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) + \frac{3}{7}\left(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}\right) + \frac{2}{7}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) \approx 0.965$$

$IG(y_2) = H(y_{out} | y_1 > 0.4) - H(y_{out} | y_1 > 0.4 \wedge y_2)$

$= 1.557 - 0.965 = \underline{0.592}$

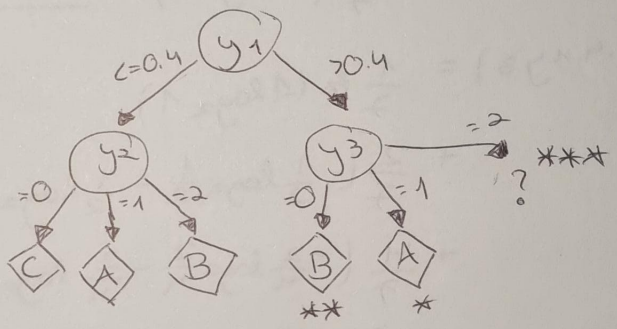
$IG(y_3) = H(y_{out} | y_1 > 0.4) - H(y_{out} | y_1 > 0.4 \wedge y_3)$

$= 1.557 - 0.857 = \underline{0.700}$

$IG(y_4) = H(y_{out} | y_1 > 0.4) - H(y_{out} | y_1 > 0.4 \wedge y_4)$

$= 1.557 - 0.965 = \underline{0.592}$

Como y_3 tem o maior ganho de informação vai ser a próxima variável a ser usada na árvore de decisão.



* Para $y_1 > 0.4 \wedge y_3 = 1$ só há duas observações. Uma delas dá o output A e a outra B. Como está escrito no enunciado que é preciso 4 observações para dividir um nó, não vamos dividir o nó e escolher por ordem alfabética (como está dito no ponto ii)).

** Para $y_1 > 0.4 \wedge y_3 = 0$ só há uma observação com output B, logo a folha vai ser B.

*** Como há 4 observações para $y_1 > 0.4$ e $y_3 = 2$ e 2 yout diferentes, vamos ter de analisar outra variável para decidir.

$$H(y_{out} \mid y_1 > 0.4 \wedge y_3 = 2) = \frac{-2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4} = \underline{1}$$

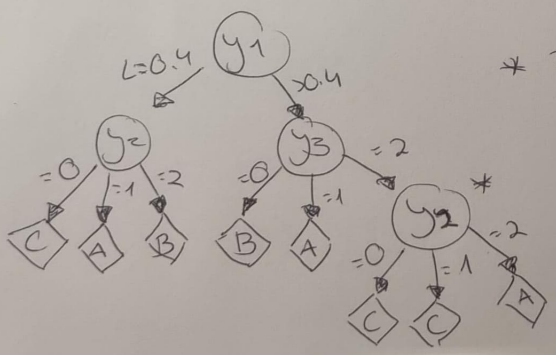
$$H(y_{out} \mid y_1 > 0.4 \wedge y_3 = 2 \wedge y_2) = \frac{1}{4}(-1\log_2 1) + \frac{1}{4}(-1\log_2 1) + \frac{2}{4}(-1\log_2 1) = \underline{0}$$

$$H(y_{out} \mid y_1 > 0.4 \wedge y_3 = 2 \wedge y_4) = \frac{2}{4}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) + \frac{1}{4}(-1\log_2 1) + \frac{1}{4}(-1\log_2 1) = \underline{0.5}$$

$$IG(y_2) = H(y_{out} \mid y_1 > 0.4 \wedge y_3 = 2) - H(y_{out} \mid y_1 > 0.4 \wedge y_3 = 2 \wedge y_2) = 1 - 0 = \underline{1}$$

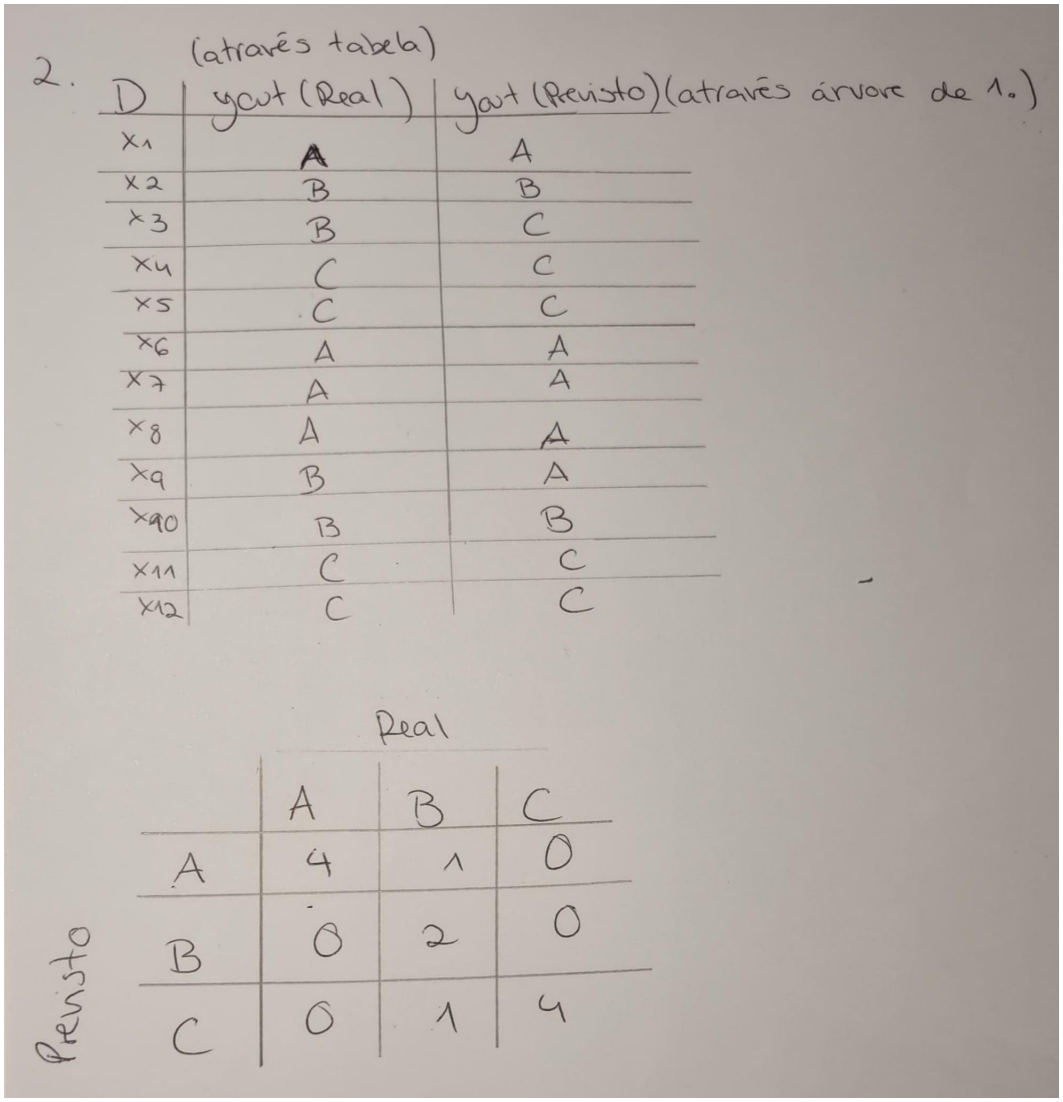
$$IG(y_4) = H(y_{out} \mid y_1 > 0.4 \wedge y_3 = 2) - H(y_{out} \mid y_1 > 0.4 \wedge y_3 = 2 \wedge y_4) = 1 - 0.5 = 0.5$$

Como y_2 tem maior ganho de informação, vai ser a próxima variável a ser usada.



* Para $y_2 = 0$ e $y_2 = 1$ só vamos ter 1 observação, logo vai ser claro o valor da folha. Para $y_2 = 2$ temos 2 observações com o mesmo yout, logo também vai ser claro.

2.



2. (através tabela)

D	yout (Real)	yout (Previsto) (através árvore de 1.)
x_1	A	A
x_2	B	B
x_3	B	C
x_4	C	C
x_5	C	C
x_6	A	A
x_7	A	A
x_8	A	A
x_9	B	A
x_{90}	B	B
x_{11}	C	C
x_{12}	C	C

Previsto \ Real	A	B	C
A	4	1	0
B	0	2	0
C	0	1	4

3.

3. $F1\text{-}score(\beta=1) = \frac{1}{F} = \frac{1}{2}\left(\frac{1}{P} + \frac{1}{R}\right)$

↳ P = Precision
R = Recall

$P = \frac{TP}{TP+FP}$ $\quad R = \frac{TP}{TP+FN}$

TP: true positive
FP: false positive
T-N: false negative

	A	B	C
Recall	$\frac{4}{4+0} = 1$	$\frac{2}{2+2} = \frac{1}{2}$	$\frac{4}{4+0} = 1$
Precision	$\frac{4}{4+1} = \frac{4}{5}$	$\frac{2}{2+0} = 1$	$\frac{4}{4+1} = \frac{4}{5}$

$\frac{1}{F_A} = \frac{1}{2}\left(\frac{1}{4/5} + \frac{1}{1}\right) \Leftrightarrow F_A = \frac{8}{9}$

$\frac{1}{F_B} = \frac{1}{2}\left(\frac{1}{1} + \frac{1}{1/2}\right) \Leftrightarrow F_B = \frac{2}{3}$

$\frac{1}{F_C} = \frac{1}{2}\left(\frac{1}{4/5} + \frac{1}{1}\right) \Leftrightarrow F_C = \frac{8}{9}$

Logo a classe com menor F1-Score é a classe B.

4.

4. Pearson Correlation Coefficient (PCC)

$$PCC(y_1, y_2) = \frac{cov(y_1, y_2)}{\sqrt{var(y_1)}\sqrt{var(y_2)}} = \frac{\Sigma y_1 y_2 - \frac{\Sigma y_1 \Sigma y_2}{n}}{\sqrt{\left(\Sigma y_1^2 - \frac{(\Sigma y_1)^2}{n}\right) \times \left(\Sigma y_2^2 - \frac{(\Sigma y_2)^2}{n}\right)}}$$

	y_1	rank y_1	y_2	rank y_2	$(r_{y_1})^2$	$(r_{y_2})^2$	$r_{y_1} \times r_{y_2}$
x_1	0.24	3	1	8	9	64	24
x_2	0.06	2	2	11	4	121	22
x_3	0.04	1	0	3.5	1	12.25	3.5
x_4	0.36	5	0	3.5	25	12.25	17.5
x_5	0.32	4	0	3.5	16	12.25	14
x_6	0.68	10	2	11	100	121	110
x_7	0.9	12	0	3.5	144	12.25	42
x_8	0.76	11	2	11	121	121	121
x_9	0.46	7	1	8	49	64	56
x_{10}	0.62	9	0	3.5	81	12.25	31.5
x_{11}	0.44	6	1	8	36	64	48
x_{12}	0.52	8	0	3.5	64	12.25	28
		$\Sigma = 78$		$\Sigma = 78$	$\Sigma = 650$	$\Sigma = 628.5$	$\Sigma = 517.5$

rank = ordenar por ordem crescente

rank $y_2 = 0$ = $\frac{1+2+3+4+5+6}{6} = 3.5$

rank $y_2 = 1$ = $\frac{7+8+9}{3} = 8$

rank $y_2 = 2$ = $\frac{10+11+12}{3} = 11$

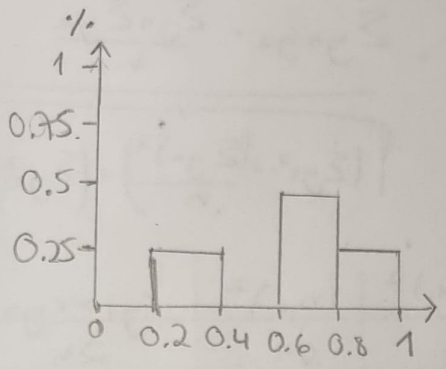
Spearman (y_1, y_2) = PCC ([3,2,1,5,4,10,12,11,7,9,6,8], [6,11,3.5,3.5,3.5,11,3.5,11,8,3.5,8,3.5])

$$= \left(517.5 - \frac{78 \times 78}{12}\right) / \sqrt{\left(650 - \frac{78^2}{12}\right) \times \left(628.5 - \frac{78^2}{12}\right)}$$

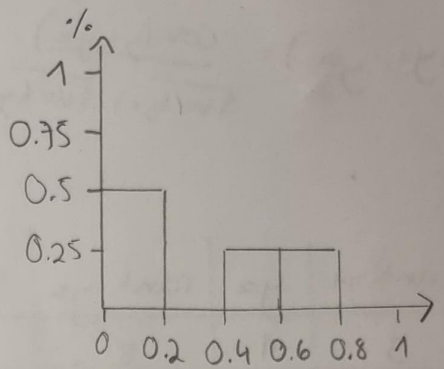
≈ 0.07966, logo y_1 e y_2 estão "loosely related"

5.

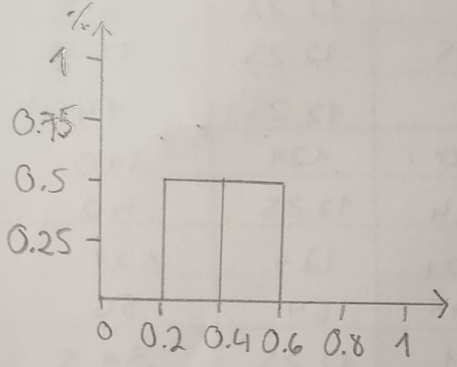
5. classe A



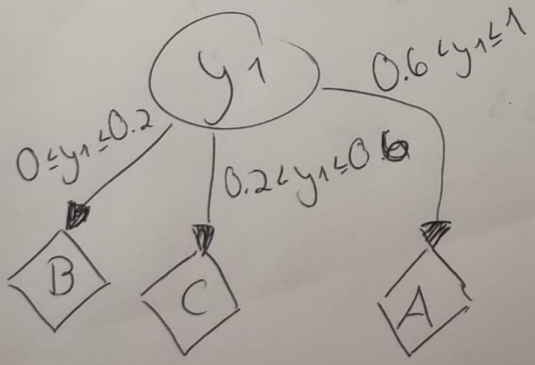
classe B



classe C



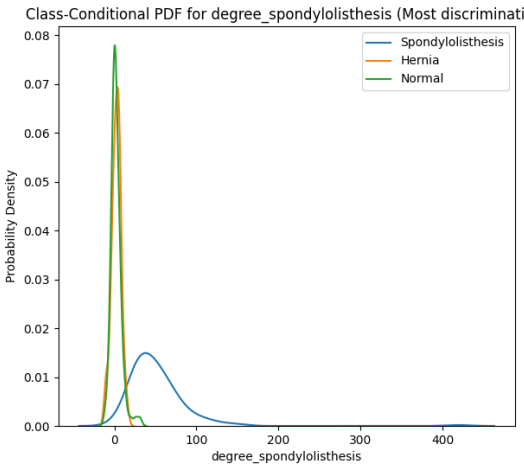
Olhando para os histogramas, conseguimos ver que a classe que tem mais observações de y_1 entre 0 e 0.2 é a B, entre 0.2 e 0.4 e entre 0.4 e 0.6 é a C e entre 0.6 e 0.8 e entre 0.8 e 1 é A. Logo podemos dividir y_1 do seguinte maneira

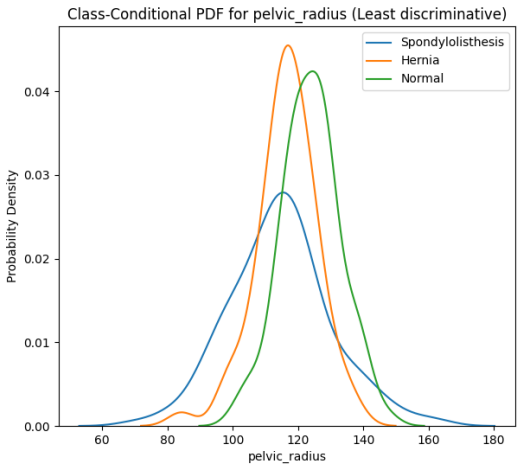


Part II: Programming

1.

```python
from sklearn.feature_selection import f_classif
from scipy.io.arff import loadarff
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

data = loadarff('column_diagnosis.arff')

df = pd.DataFrame(data[0])

x = df.drop('class', axis=1)
y = df['class']

f_values, p_values = f_classif(x, y)

highest_discriminative_idx = np.argmax(f_values)
lowest_discriminative_idx = np.argmin(f_values)

variable_names = ["pelvic_incidence", "pelvic_tilt", "lumbar_lordosis_angle", "
    sacral_slope", "pelvic_radius", "degree_spondylolisthesis"]

most_discriminative_variable = variable_names[highest_discriminative_idx]
least_discriminative_variable = variable_names[lowest_discriminative_idx]

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 6))

sns.distplot(df[df["class"] == b'Spondylolisthesis'][most_discriminative_variable
    ], hist=False, label="Spondylolisthesis", ax=ax1)
sns.distplot(df[df["class"] == b'Hernia'][most_discriminative_variable], hist=
    False, label="Hernia", ax=ax1)
sns.distplot(df[df["class"] == b'Normal'][most_discriminative_variable], hist=
    False, label="Normal", ax=ax1)
ax1.set_xlabel(most_discriminative_variable)
ax1.set_ylabel("Probability Density")
ax1.set_title(f"Class-Conditional PDF for {most_discriminative_variable} (Most
    discriminative)")

sns.distplot(df[df["class"] == b'Spondylolisthesis'][least_discriminative_variable
    ], hist=False, label="Spondylolisthesis", ax=ax2)
sns.distplot(df[df["class"] == b'Hernia'][least_discriminative_variable], hist=
    False, label="Hernia", ax=ax2)
sns.distplot(df[df["class"] == b'Normal'][least_discriminative_variable], hist=
    False, label="Normal", ax=ax2)
ax2.set_xlabel(least_discriminative_variable)
ax2.set_ylabel("Probability Density")
ax2.set_title(f"Class-Conditional PDF for {least_discriminative_variable} (Least
    discriminative)")

ax1.legend()
ax2.legend()

plt.savefig("Exercicio1.png")
plt.tight_layout()
plt.show()
```

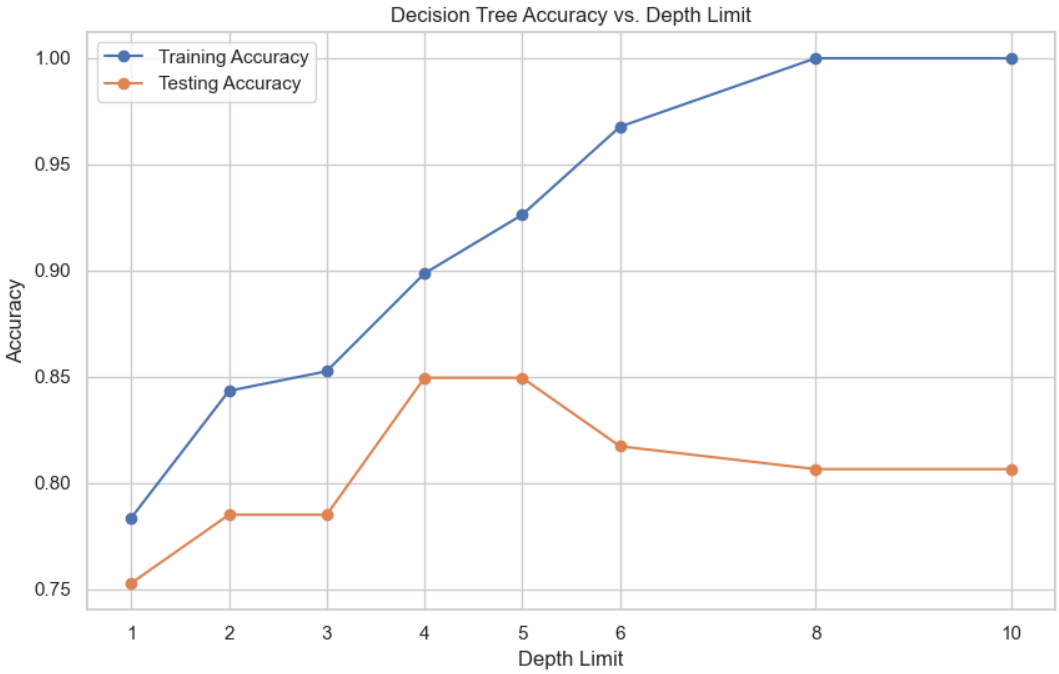




2.

```python
from scipy.io.arff import loadarff
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder

data = loadarff('column_diagnosis.arff')

df = pd.DataFrame(data[0])

le = LabelEncoder()
df['class'] = le.fit_transform(df['class'])

x = df.drop('class', axis=1)
y = df['class']

x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7,
    random_state=0, stratify=y)

depth_limits = [1, 2, 3, 4, 5, 6, 8, 10]

train_accuracies = []
test_accuracies = []

for depth in depth_limits:
    train_acc = 0
    test_acc = 0
    for _ in range(10):
        clf = DecisionTreeClassifier(max_depth=depth, random_state=0)
        clf.fit(x_train, y_train)

        train_acc += accuracy_score(y_train, clf.predict(x_train))
        test_acc += accuracy_score(y_test, clf.predict(x_test))

    train_acc /= 10
    test_acc /= 10
```

```
39
40      train_accuracies.append(train_acc)
41      test_accuracies.append(test_acc)
42
43  sns.set(style="whitegrid")
44  plt.figure(figsize=(10, 6))
45  plt.plot(depth_limits, train_accuracies, marker='o', label='Training Accuracy')
46  plt.plot(depth_limits, test_accuracies, marker='o', label='Testing Accuracy')
47  plt.title('Decision Tree Accuracy vs. Depth Limit')
48  plt.xlabel('Depth Limit')
49  plt.ylabel('Accuracy')
50  plt.legend()
51  plt.xticks(depth_limits)
52
53  plt.savefig("Exercicio2.png")
54
55  plt.show()
```



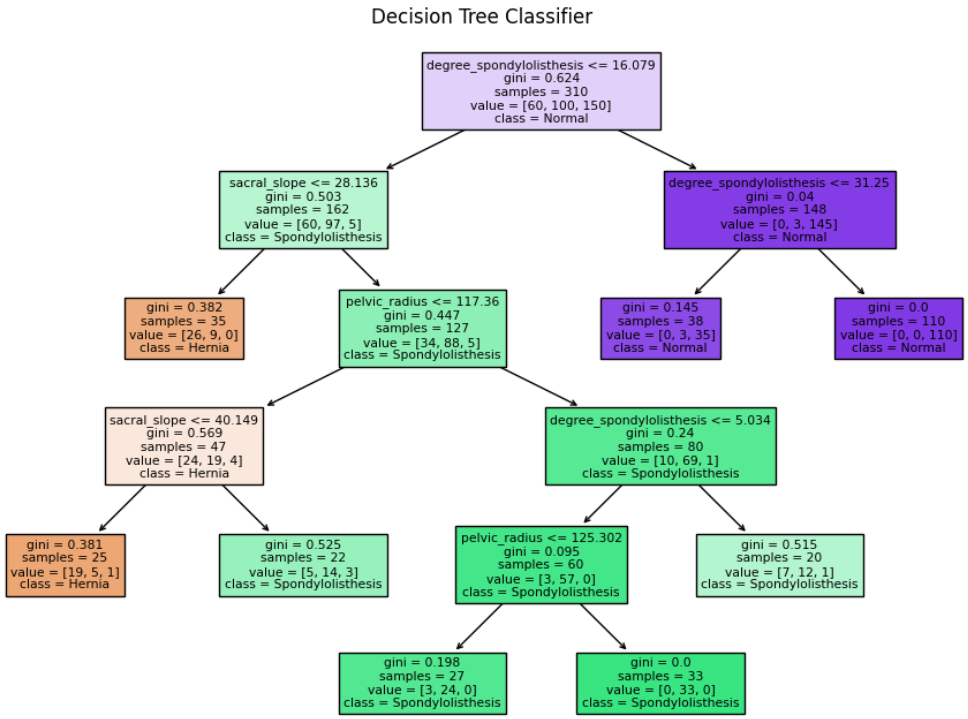
3. Ao observar o gráfico resultante do exercício anterior, reparamos que a accuracy do treino melhora com o aumento da profundidade da árvore de decisão. Este resultado faz sentido e pode ser explicado pelo facto de árvores de decisão mais profundas tedem a capturar padrões mais complexos nos dados para treino.

   Por outro lado, a accuracy de teste começa por aumentar até à profundidade de 4 ou 5, começando logo em seguida a descer. Este comportamento pode ser explicado pelo overfitting que é um fenómeno onde árvores mais profundas se tornam demasiado específicas para os valores usados no treino e por consequência não conseguem generalizar os resultados para dados novos, os dados de teste.

   O resultado mais alto da accuracy de teste é em árvores de profundidade 4 ou 5, isto quer dizer que árvores de profundidade média são as que oferecem melhores resultados evitando o problema de overfitting. Concluindo, a escolha de uma profundidade de 4 ou 5 parece ser a mais adequada para este caso.

4. i.

```python
from scipy.io.arff import loadarff
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
from sklearn.preprocessing import LabelEncoder

data = loadarff('column_diagnosis.arff')

df = pd.DataFrame(data[0])

le = LabelEncoder()
df['class'] = le.fit_transform(df['class'])

x = df.drop('class', axis=1)
y = df['class']

clf = DecisionTreeClassifier(min_samples_leaf=20, random_state=0)
clf.fit(x, y)

plt.figure(figsize=(12, 8))
plot_tree(clf, filled=True, feature_names=x.columns, class_names=['Hernia', '
    Spondylolisthesis', 'Normal'])
plt.title("Decision Tree Classifier")
plt.savefig("Exercicio4.png")
plt.show()
```



ii. Para detetar a presença de Disk Hernia através da nossa árvore de decisão é possível seguir dois ramos da árvore. O primeiro e mais simples, permite-nos chegar à conclusão que um paciente tem esta doença apenas sabendo que tem um degree_spondylolisthesis <= 16,079 e um sacral_slope <= 28,136. Para se chegar à mesma conclusão pelo segundo ramo, é preciso não só degree_spondylolisthesis <= 16,079, mas também uma pelvic_radius <= 117,36 e uma sacral_slope <= 40,149. Com isto concluímos, que se pode considerar que um paciente tem uma Disk Hernia quando tem um degree_spondylolisthesis <= 16,079 e uma sacral_slope <= 28,136, no entanto se a sacral_slope > 28,136, mas ainda for <= 40,149 e se a pelvic_radius <= 117,36 também é possível concluir que o paciente tem Disk Hernia.