Part I: Pen and Paper

1) $H(y_{out}|y_1>0.4) = \overbrace{-\frac{3}{7}\log_2\frac{3}{7}}^{A} \overbrace{-\frac{2}{7}\log_2\frac{2}{7}}^{B} \overbrace{-\frac{2}{7}\log_2\frac{2}{7}}^{C} =$

$\simeq 1.557$

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

$= 1.557 - \Big(\underbrace{-\frac{3}{7}\times\left(\frac{1}{3}\log_2\frac{1}{3}\times 3\right)}_{y_2=0} \underbrace{-\frac{2}{7}\times\left(\frac{1}{2}\log_2\frac{1}{2}\times 2\right)}_{y_2=1} -$

$\underbrace{-\frac{2}{7}\left(1\log_2 1\right)}_{y_2=2}\Big) =$

$= 1.557 - \left(-\frac{3}{7}\times(-1.585) - \frac{2}{7}\times(-1) - \frac{2}{7}\times 0\right) =$

$= 1.557 - 0.964 \simeq 0.592$

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

$- H(y_{out}|y_3, y_1>0.4) =$

$= 1.557 - \Big(\overbrace{-\frac{1}{7}\times(1\log_2 1)}^{y_3=0} \overbrace{-\frac{2}{7}\times\left(2\cdot\frac{1}{2}\times\log_2\frac{1}{2}\right)}^{y_3=1} -$

$-\frac{4}{7}\times\left(2\cdot\frac{1}{2}\log_2\frac{1}{2}\right)\Big) =$

$= 1.557 - \left(-\frac{2}{7}\times(-1) - \frac{4}{7}\times(-1)\right) =$

$= 1.557 - 0.857 \simeq 0.700$

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

$$- H(y_{out} \mid y_4, y_1 > 0.4) =$$

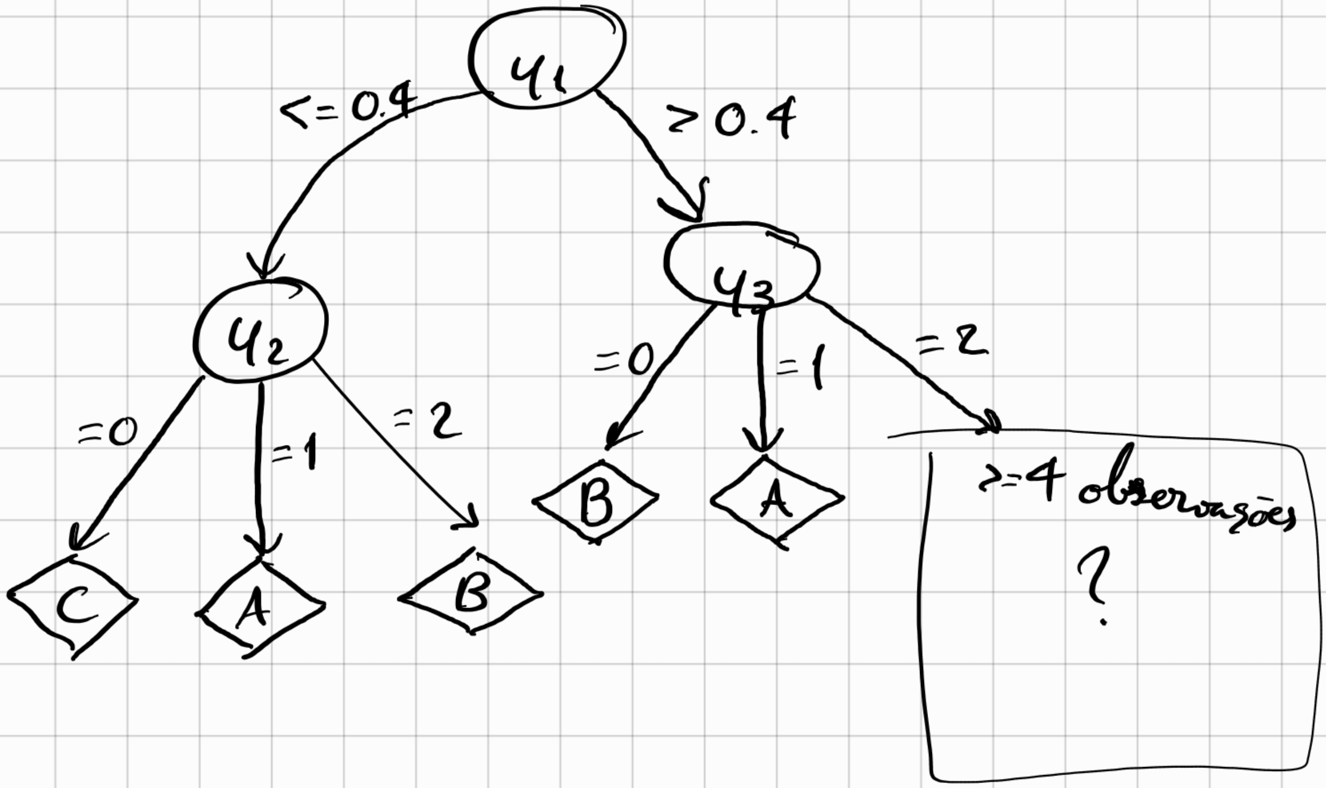
$$= 1.557 - \Big(\overbrace{-\frac{2}{7} \times \left(2 \cdot \frac{1}{2} \times \log_2 \frac{1}{2}\right)}^{y_4 = 0} - \frac{3}{7} \times \left(\frac{2}{3} \times \log_2 \frac{2}{3} + \right.$$

$$\left. + \frac{1}{3} \log_2 \frac{1}{3} \right) - \frac{2}{7} \times \left(2 \cdot \frac{1}{2} \log_2 \frac{1}{2} \right) \Big) =$$

$$= 1.557 - \left(-\frac{2}{7} \times (-1) - (-0.394) - \frac{2}{7} \times (-1) \right) =$$

$$= 1.557 - 0.965 = 0.592$$

Como $IG(y_{out} \mid y_3, y_1 > 0.4) > IG(y_{out} \mid y_2, y_1 > 0.4)$
$= IG(y_{out} \mid y_4, y_1 > 0.4)$,
então a árvore de decisão é:



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

$$IG(y_{out} \mid y_2, y_1 > 0.4, y_3 = 2) = H(y_{out} \mid y_1 > 0.4, y_3 = 2) -$$

$$- H(y_{out} \mid y_2, y_1 > 0.4, y_3 = 2) =$$

$$= 1 - \Big(\overbrace{-\frac{1}{4}(1 \log_2 1)}^{y_2 = 0} \overbrace{-\frac{1}{4}(1 \log_2 1)}^{y_2 = 1} - \frac{1}{2}(1 \log_2 1) \Big) =$$

$$= 1 - 0 = 1$$

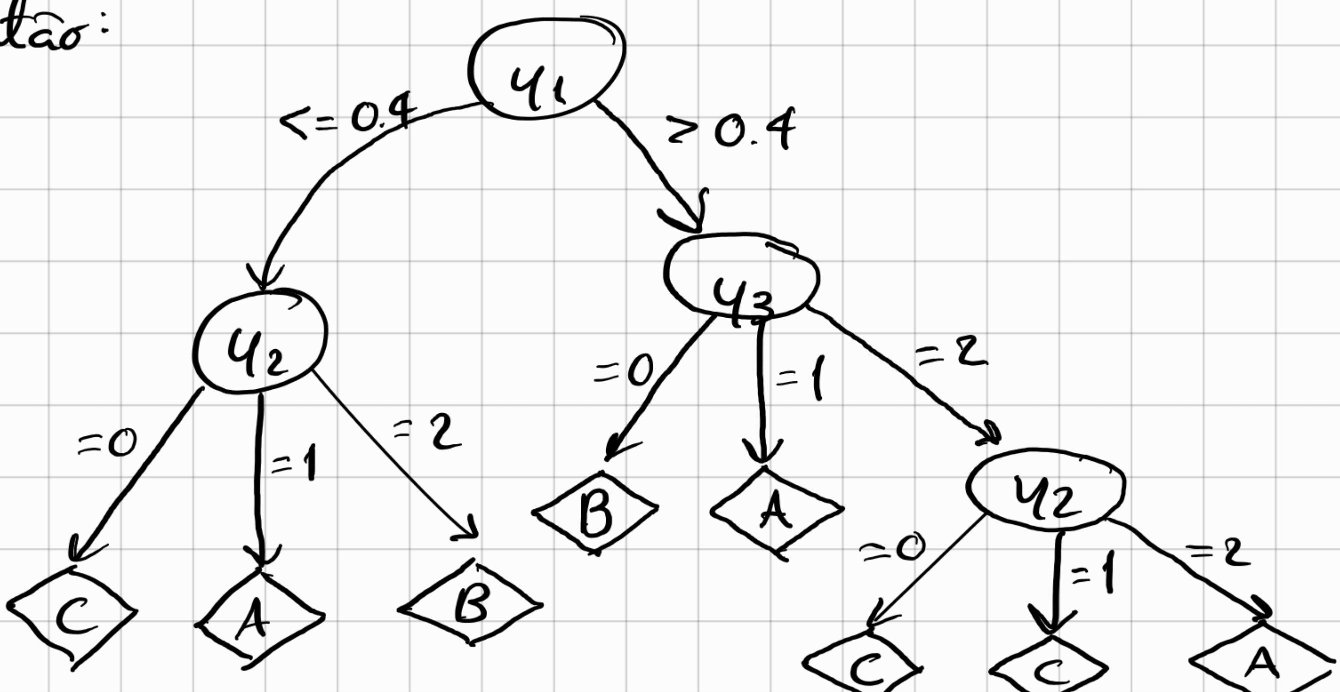
$$IG(y_{out} \mid y_4, y_1 > 0.4, y_3 = 2) = H(y_{out} \mid y_1 > 0.4, y_3 = 2) -$$

$$- H(y_{out} \mid y_4, y_1 > 0.4, y_3 = 2) =$$

$$= 1 - \Big(\overbrace{-\frac{1}{2} \times \left(2 \cdot \frac{1}{2} \times \log_2 \frac{1}{2}\right)}^{y_4 = 0} \overbrace{-\frac{1}{4} \times (1 \log_2 1)}^{y_4 = 1} \overbrace{-\frac{1}{4} \times (1 \log_2 1)}^{y_4 = 2} \Big)$$

$$= 1 - \left(-\frac{1}{2} \times (-1)\right) = \frac{1}{2}$$

Como $IG(y_{out} \mid y_2, y_1 > 0.4, y_3 = 2) > IG(y_{out} \mid y_4, y_1 > 0.4, y_3 = 2)$, então:



2) Ao usarmos os dados do dataset D para avaliação:

D	y_1	y_2	y_3	y_4	y_{out}	$\hat{y}_{out}$
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	B	C
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	0.9	0	1	2	A	A
x_8	0.76	2	2	0	A	A
x_9	0.46	1	1	1	B	A
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

Assim, tendo em conta a diferença entre os resultados previstos e os observados, chegamos à seguinte matriz de confusão:

previstos \ observados	A	B	C	
A	4	1	0	5
B	0	2	0	2
C	0	1	4	5
	4	4	4	12

3) <u>Classe A:</u>

$$\text{precisão}_A = \frac{\#TP_A}{\#TP_A + \#FP_A} = \frac{4}{4+1+0} = \frac{4}{5} = 80\%$$

$$\text{sensibilidade}_A = \frac{\#TP_A}{\#TP_A + \#FN_A} = \frac{4}{4+0+0} = 100\%$$

$$\therefore F_1\text{-measure}_A = 2 \times \frac{\text{precisão}_A \times \text{sensibilidade}_A}{\text{precisão}_A + \text{sensibilidade}_A} = 2 \times \frac{0.80 \times 1}{0.80 + 1} =$$

$$= 0.8\dot{8} \simeq 88,8\%$$

<u>Classe B:</u>

$$\text{precisão}_B = \frac{\#TP_B}{\#TP_B + \#FP_B} = \frac{2}{0+2+0} = 100\%$$

$$\text{sensibilidade}_B = \frac{\#TP_B}{\#TP_B + \#FN_B} = \frac{2}{1+2+1} = \frac{2}{4} = 50\%$$

$$\therefore F_1\text{-measure}_B = 2 \times \frac{\text{precisão}_B \times \text{sensibilidade}_B}{\text{precisão}_B + \text{sensibilidade}_B} = 2 \times \frac{1 \times 0.5}{1 + 0.5}$$

$$= 0.6\dot{6} \simeq 66,6\%$$

<u>Classe C</u>

$$\text{precisão}_C = \frac{\#TP_C}{\#TP_C + \#FP_C} = \frac{4}{4+1+0} = \frac{4}{5} = 80\%$$

$$\text{sensibilidade}_C = \frac{\#TP_C}{\#TP_C + \#FN_C} = \frac{4}{4+0+0} = 100\%$$

$$\therefore F_1\text{-measure}_c = 2\times\frac{\text{precisão}_c \times \text{sensibilidade}_c}{\text{precisão}_c + \text{sensibilidade}_c} = 2\times\frac{0.80\times 1}{0.80+1} =$$

$$= 0.8\dot{8} \simeq 88,8\,\%$$

R: Como tal, a classe B tem o menor valor da estatística F_1-measure.

4) Passo 1 Atribuir ranks aos valores das variáveis a estudar.

	y_1	y_2	Rank y_1	Rank y_2
x_1	0.24	1	3	8
x_2	0.06	2	2	11
x_3	0.04	0	1	3.5
x_4	0.36	0	5	3.5
x_5	0.32	0	4	3.5
x_6	0.68	2	10	11
x_7	0.9	0	12	3.5
x_8	0.76	2	11	11
x_9	0.46	1	7	8
x_{10}	0.62	0	9	3.5
x_{11}	0.44	1	6	8
x_{12}	0.52	0	8	3.5

C.A.

$$\frac{1+2+3+4+5+6}{6} = \frac{7}{2} = 3.5$$

$$\frac{7+8+9}{3} = 8$$

$$\frac{10+11+12}{3} = 11$$

Passo 2 Calcular o coeficiente de Pearson usando os ranks das variáveis

Obs.

$$\overline{\text{Rank}\,y_1} = \frac{1+2+\dots+12}{12} = 6.5\ ;\ \overline{\text{Rank}\,y_2} = \frac{3.5\times 6 + 8\times 3 + 11\times 3}{12} = $$

$$= 6.5$$

$$\text{Pearson}(\text{Rank } y_1, \text{Rank } y_2) =$$

$$= \frac{\sum_i (\text{Rank } y_{1i} - \overline{\text{Rank } y_1})(\text{Rank } y_{2i} - \overline{\text{Rank } y_2})}{\sqrt{\sum_i (\text{Rank } y_{1i} - \overline{\text{Rank } y_1})^2 \cdot \sum_i (\text{Rank } y_{2i} - \overline{\text{Rank } y_2})^2}} =$$

$$= \frac{(3-6.5)(8-6.5)+(2-6.5)(11-6.5)+\cdots+(8-6.5)(3.5-6.5)}{\sqrt{((3-6.5)^2+(2-6.5)^2+\cdots+(8-6.5)^2)((8-6.5)^2+(11-6.5)^2+\cdots+(3.5-6.5)^2}} =$$

$$= 0.0796$$

Visto que o valor do coeficiente de Spearman se aproxima de 0 (zero), então conclui-se que as relações y_1 e y_2 não estão correlacionadas.

5)

5 bins:	Hist. A	Hist. B	Hist. C
$[0, 0.2[$		x_2, x_3	
$[0.2, 0.4[$	x_1		x_4, x_5
$[0.4, 0.6[$		x_9	x_{11}, x_{12}
$[0.6, 0.8[$	x_6, x_8	x_{10}	
$[0.8, 1]$	x_7		

$P(0 \leq y_1 \leq 0.2 \mid y_{out} = A) = 0$

$P(0.2 \leq y_1 \leq 0.4 \mid y_{out} = A) = \frac{1}{4}$

$P(0.4 \leq y_1 \leq 0.6 \mid y_{out} = A) = 0$

$P(0.6 \leq y_1 \leq 0.8 \mid y_{out} = A) = \frac{1}{2}$

$P(0.8 \leq y_1 \leq 1 \mid y_{out} = A) = \frac{1}{4}$

$P(0 \leq y_1 \leq 0.2 \mid y_{out} = B) = \frac{1}{2}$

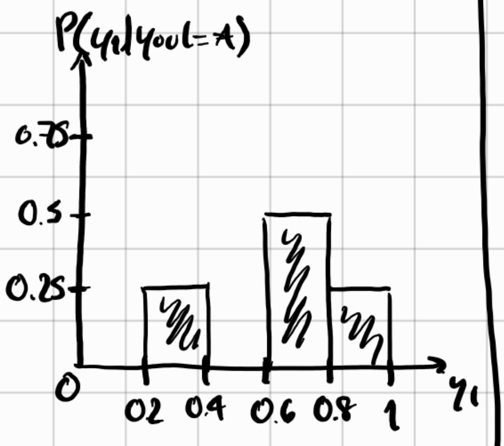
$P(0.2 \leq y_1 \leq 0.4 \mid y_{out} = B) = 0$

$P(0.4 \leq y_1 \leq 0.6 \mid y_{out} = B) = \frac{1}{4}$

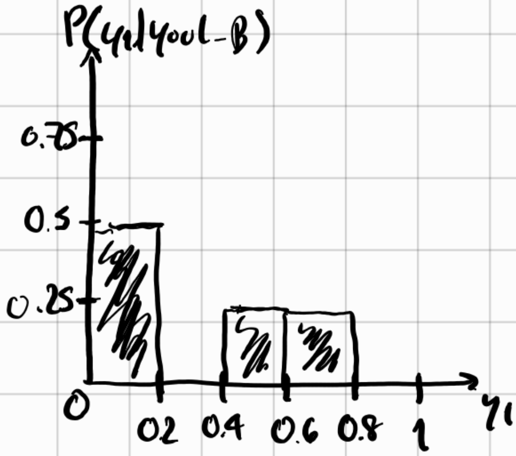
$$P(0.6 \leq y_1 \leq 0.8 \mid y_{out} = B) = \tfrac{1}{4}$$
$$P(0.8 \leq y_1 \leq 1 \mid y_{out} = B) = 0$$

$$P(0 \leq y_1 \leq 0.2 \mid y_{out} = C) = 0$$
$$P(0.2 \leq y_1 \leq 0.4 \mid y_{out} = C) = \tfrac{1}{2}$$
$$P(0.4 \leq y_1 \leq 0.6 \mid y_{out} = C) = \tfrac{1}{2}$$
$$P(0.6 \leq y_1 \leq 0.8 \mid y_{out} = C) = 0$$
$$P(0.8 \leq y_1 \leq 1 \mid y_{out} = C) = 0$$

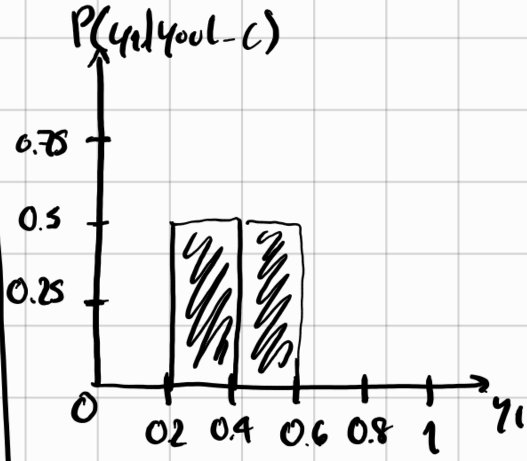
Classe A



Classe B

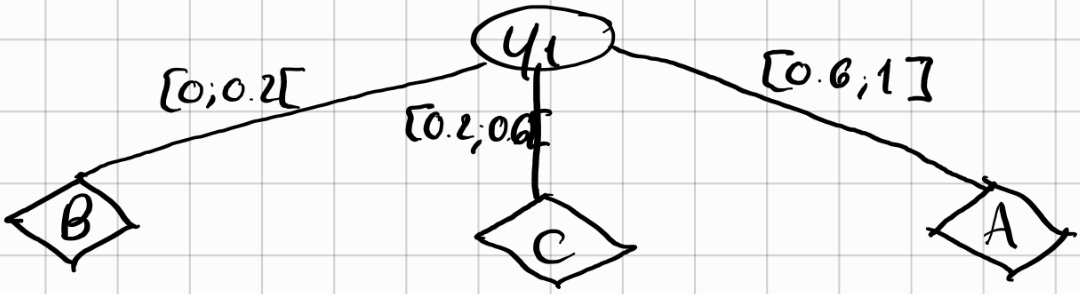


Classe C



De acordo com os gráficos gerados acima, obtivemos o seguinte root-split:

- Entre 0 e 0.2 obtém-se a classe B;
- Entre 0.2 e 0.4 obtém-se a classe C;
- Entre 0.4 e 0.6 obtém-se a classe C;
- Entre 0.6 e 0.8 obtém-se a classe A;
- Entre 0.8 e 1 obtém-se a classe A.



Homework 1 (Part II)

Aprendizagem 2023/2024 - LEIC @ IST

Group #24

- Daniel Nunes (Nº 103095)
- Gonçalo Alves (Nº 103540)

Data importing

In [ ]:
```
import pandas as pd, numpy as np
from scipy.io.arff import loadarff

data = loadarff('column_diagnosis.arff')
df = pd.DataFrame(data[0])
df['class'] = df['class'].str.decode('utf-8')

df.head()
```

Out[ ]:

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degre
0	63.027817	22.552586	39.609117	40.475232	98.672917	
1	39.056951	10.060991	25.015378	28.995960	114.405425	
2	68.832021	22.218482	50.092194	46.613539	105.985135	
3	69.297008	24.652878	44.311238	44.644130	101.868495	
4	49.712859	9.652075	28.317406	40.060784	108.168725	

Exercise 1

To first use the `f_classif` function, we first have to state explicitly which variables are considered inputs and which one is the output variable.

After this, we can use the `f_classif` function to get the F1-score between each input variable and the output variable.

In [ ]:
```
from sklearn.feature_selection import f_classif

df_inputs = df.drop('class', axis=1)
df_outputs = df['class']
```

In [ ]:
```
f_values, p_values = f_classif(df_inputs, df_outputs)

# Display output in a nicely arranged table
fimportance_data = {
    'Variable': df.columns.values[:-1],
    'F1-score': f_values,
```

```
    'P-value': p_values
}
fimportance_df = pd.DataFrame(fimportance_data)
display(fimportance_df)
```

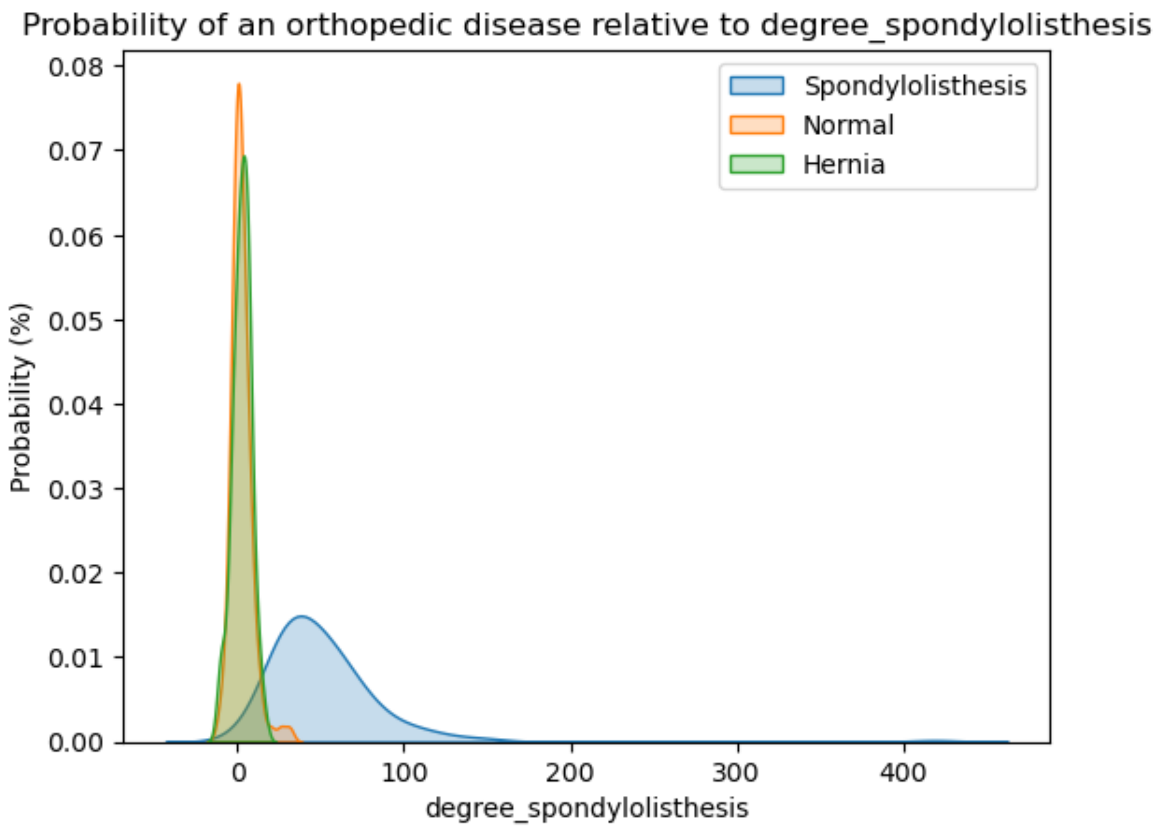
	Variable	F1-score	P-value
0	pelvic_incidence	98.539709	8.752849e-34
1	pelvic_tilt	21.299194	2.176879e-09
2	lumbar_lordosis_angle	114.982840	5.357329e-38
3	sacral_slope	89.643953	2.175670e-31
4	pelvic_radius	16.866935	1.121996e-07
5	degree_spondylolisthesis	119.122881	5.114732e-39

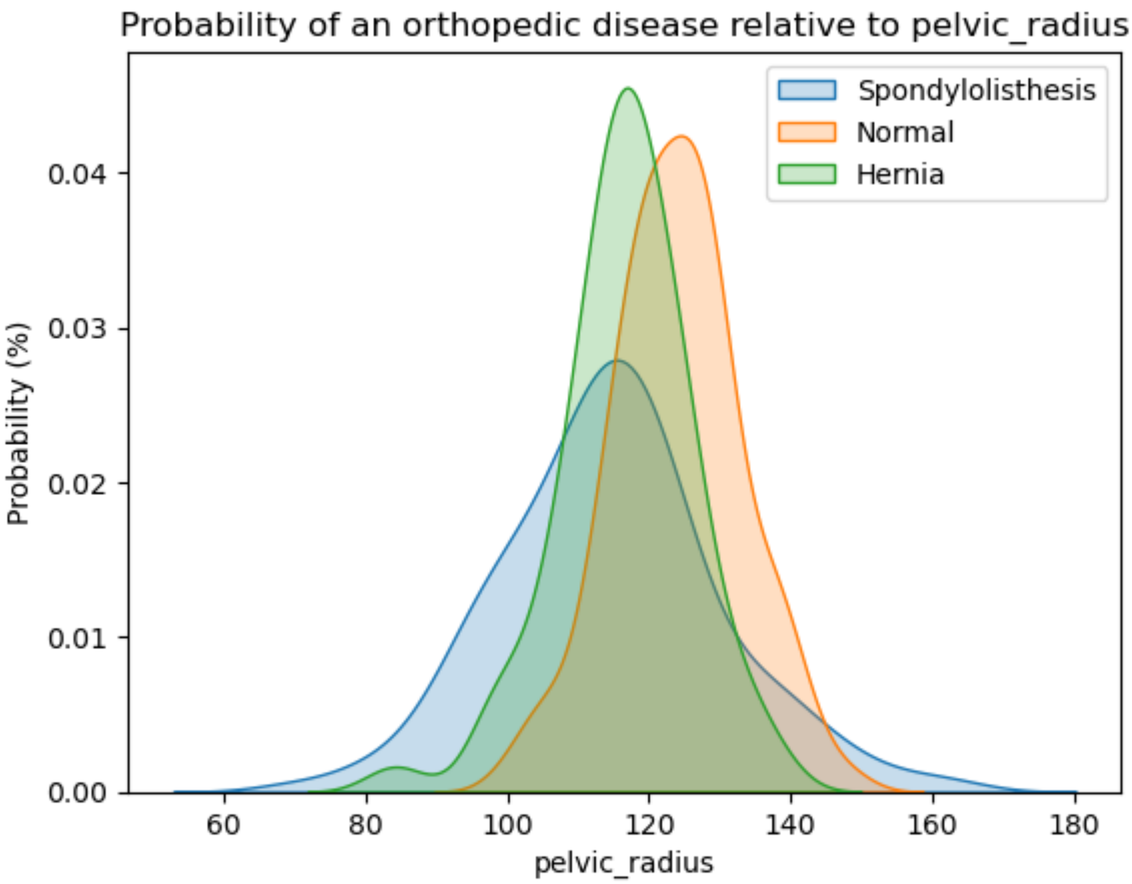
Since the F1-score is directly related to the discriminative power of a variable, then we can conclude that the `sacral_slope` has the lowest discriminative power, while `degree_spondylolisthesis` has the highest one.

Next, let's plot the class-conditional probability functions for these two variables.

In [ ]:
```
# Get the name of the most and the least discriminative variables
most_discriminative = df.columns[np.argmax(f_values)]
least_discriminative = df.columns[np.argmin(f_values)]

# Fetch the respective data
most_discriminative_data = df_inputs[most_discriminative]
least_discriminative_data = df_inputs[least_discriminative]
```

In [ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns

for type in set(df['class']):
    sns.kdeplot(most_discriminative_data[df['class'] == type], \
                label=type, fill=True)
plt.xlabel(most_discriminative)
plt.ylabel('Probability (%)')
plt.legend()
plt.title("Probability of an orthopedic disease relative to " + \
          most_discriminative)
plt.show()

for type in set(df['class']):
    sns.kdeplot(least_discriminative_data[df['class'] == type], \
                label=type, fill=True)
plt.xlabel(least_discriminative)
plt.ylabel('Probability (%)')
plt.legend()
plt.title("Probability of an orthopedic disease relative to " + \
          least_discriminative)
plt.show()
```





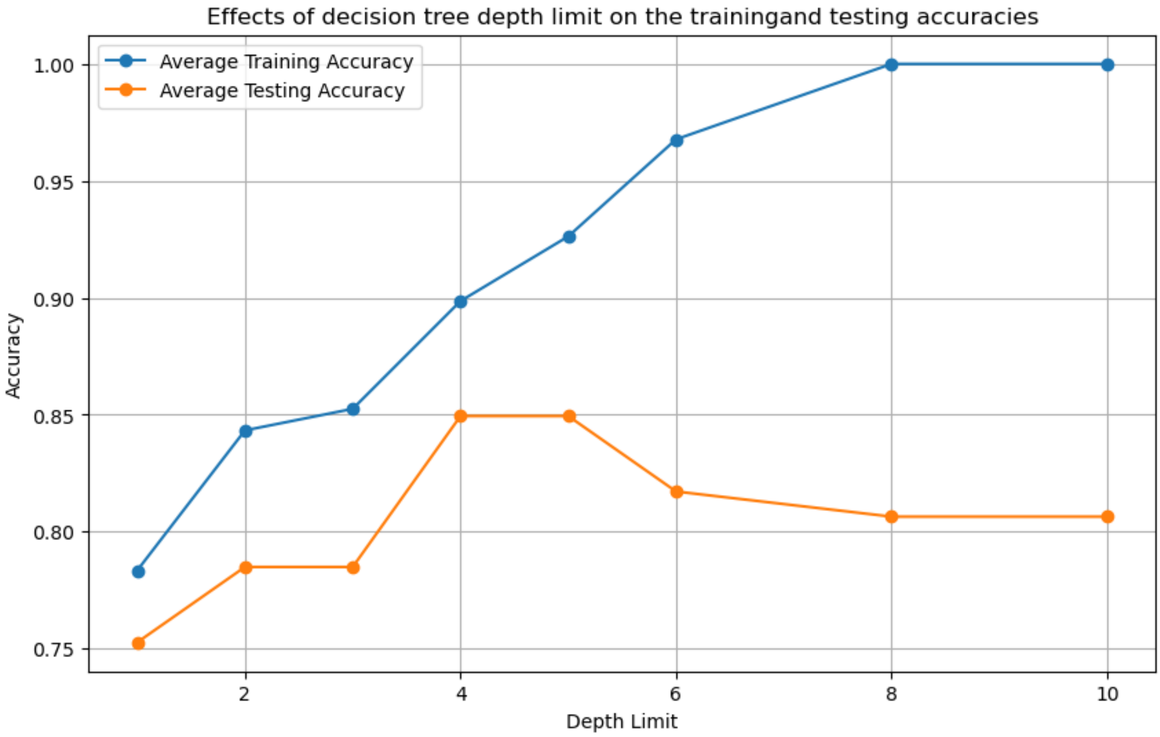
Exercise 2

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

```python
from sklearn.tree import DecisionTreeClassifier

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

In [ ]:
```python
avg_train_accuracies = []
avg_test_accuracies = []
for depth in depth_limits:
    total_train_acc = []
    total_test_acc = []

    # Use 10 runs per parametrization
    for i in range(num_runs):
        # Split data into training and testing sets with a stratified
        # 70-30 split and a fixed seed
        input_train, input_test, output_train, output_test = \
            train_test_split(df_inputs, df_outputs, train_size=0.7, \
                random_state=0, stratify=df_outputs)

        # Train a decision tree classifier with the specified depth limit
        clf = DecisionTreeClassifier(max_depth=depth, random_state=0)
        clf.fit(input_train, output_train)

        # Calculate training accuracy
        train_pred = clf.score(input_train, output_train)
        total_train_acc.append(train_pred)

        # Calculate testing accuracy
        test_pred = clf.score(input_test, output_test)
        total_test_acc.append(test_pred)

    # Calculate the average training and testing accuracies for this
    # depth limit
    avg_train_accuracy = np.mean(total_train_acc)
    avg_test_accuracy = np.mean(total_test_acc)

    avg_train_accuracies.append(avg_train_accuracy)
    avg_test_accuracies.append(avg_test_accuracy)
```

In [ ]:
```python
# Create a plot to show for average training and testing accuracies
plt.figure(figsize=(10, 6))
plt.plot(depth_limits, avg_train_accuracies, \
         label='Average Training Accuracy', marker='o')
plt.plot(depth_limits, avg_test_accuracies, \
         label='Average Testing Accuracy', marker='o')
plt.title('Effects of decision tree depth limit on the training' + \
          'and testing accuracies')
plt.xlabel('Depth Limit')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)
plt.show()
```



Exercise 3

As the gap between the training and testing accuracies widens, we approach signs of overfitting. This occurs when a model becomes overly complex due to analysing noise from the training data set.

Initially, when we have lower depth limits in the decision tree, there is a relatively smaller difference between the training and testing accuracies. However, both accuracies are relatively low, suggesting that the model is not capturing enough information from the training data, thus resulting in lower performance on both datasets.

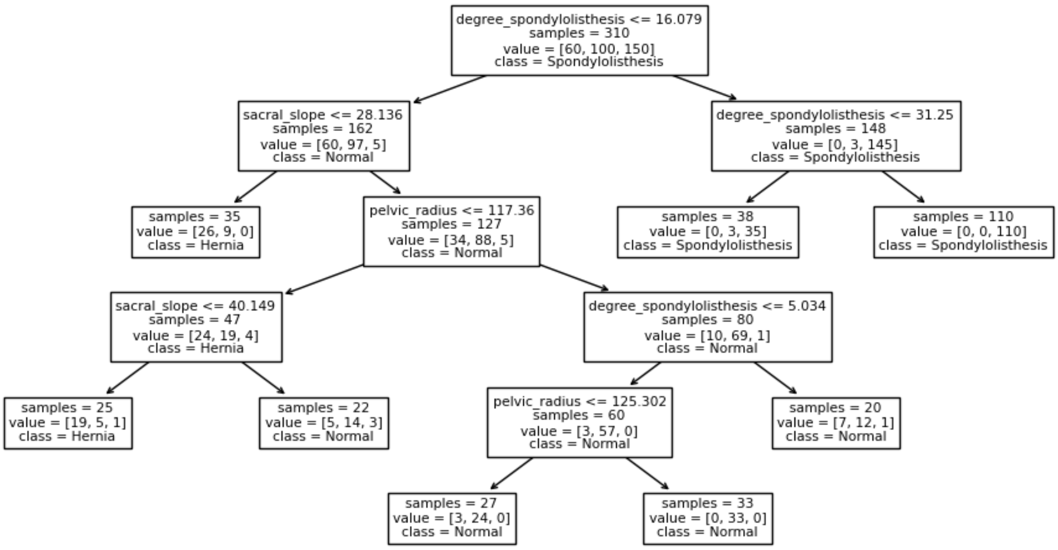
However, as we progressively increase the depth limits, we notice the difference between training and testing accuracy starts to grow. This widening gap indicates that the model is becoming more complex and has the capacity to fit the training data more closely. However, when it comes to testing the model, it starts picking up too much noise from the training data set, and starts to overcomplicate some predictions.

A depth limit of 4 or 5 appears to be the most apropriate for achieving the best accuracy while not being to far away from the accuracy of the training data.

Exercise 4

In [ ]:
```
from sklearn import tree

# Create a decision tree, ensuring that each leaf has at least
# 20 data records, and a random seed of 0
predictor = tree.DecisionTreeClassifier(min_samples_leaf=20, \
                                        random_state=0)
predictor.fit(df_inputs, df_outputs)
```

```
figure = plt.figure(figsize=(12, 6))
tree.plot_tree(predictor, feature_names=list(df.columns), \
               class_names=list(predictor.classes_), impurity=False)
plt.show()
```



From the decision tree above, we can caracterize a hernia condition when the following conditions are met:

- degree_spondylolisthesis: lower than or equal to 16.079
- sacral_slope: lower than or equal to 28.136

OR

- degree_spondylolisthesis: lower than or equal to 16.079
- sacral_slope: between 28.136 and 40.149
- pelvic_radius: lower than or equal to 117.36