Aprendizagem 2022/23

Homework I – Group 001

(ist1102082, ist1103252)

I. Pen-and-paper

1)

1)

- $H(y) = -\sum_{r \in U_y} p(r) \cdot \log p(r)$
- $IG(z|y) = H(z) - \underbrace{H(z|y)}_{\sum_{r \in U_y} P(y=r) \cdot H(z|}$

- $H(y_{out} | y_1 > 0.4) = -\left(\frac{3}{7}\log_2\left(\frac{3}{7}\right) + \frac{2}{7}\log_2\left(\frac{2}{7}\right) + \frac{2}{7}\log_2\left(\frac{2}{7}\right)\right) \simeq 1.557$

- $H(y_{out} | y_1 > 0.4, y_2) = \frac{3}{7}\left(-\left(\frac{1}{3}\log_2\left(\frac{1}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right) + \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right)\right) + \frac{2}{7}\left(-\left(0 + \frac{1}{2}\log_2\left(\frac{1}{2}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right)\right) + \frac{2}{7}\left(-\left(1\log_2(1) + 0 + 0\right)\right) =$

  $= -\frac{3}{7}\log_2\left(\frac{1}{3}\right) - \frac{2}{7}\log_2\left(\frac{1}{2}\right) \simeq 0.964984$

- $H(y_{out} | y_1 > 0.4, y_3) = \frac{1}{7}\left(-\left(0 + 1\log_2(1) + 0\right)\right) + \frac{4}{7}\left(-\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + 0 + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right)\right) + \frac{2}{7}\left(-\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + \frac{1}{2}\log_2\left(\frac{1}{2}\right) + 0\right)\right) = -\frac{4}{7}\log_2\left(\frac{1}{2}\right) - \frac{2}{7}\log\left(\frac{1}{2}\right) \simeq 0.857143$

- $H(y_{out} | y_1 > 0.4, y_4) = \frac{2}{7}\left(-\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + 0 + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right)\right) + \frac{3}{7}\left(-\left(\frac{1}{3}\log_2\left(\frac{1}{3}\right) + \frac{2}{3}\log_2\left(\frac{2}{3}\right) + 0\right)\right) + \frac{2}{7}\left(-\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + 0 + \frac{1}{2}\log_2\left(\frac{1}{2}\right)\right)\right) =$

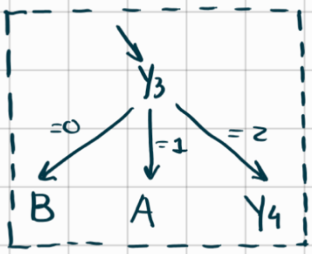
  $= -\frac{4}{7}\log_2\left(\frac{1}{2}\right) - \frac{1}{7}\log_2\left(\frac{1}{3}\right) - \frac{2}{7}\log_2\left(\frac{2}{3}\right) \simeq 0.964984$

- $IG(y_{out} | y_1 > 0.4, y_2) = H(y_{out} | y_1 > 0.4) - H(y_{out} | y_1 > 0.4, y_2) = 1.557 - 0.96484 = 0.592$

- $IG(y_{out} | y_1 > 0.4, y_3) = H(y_{out} | y_1 > 0.4) - H(y_{out} | y_1 > 0.4, y_3) = 1.557 - 0.857143 = 0.700$

- $IG(y_{out} | y_1 > 0.4, y_4) = H(y_{out} | y_1 > 0.4) - H(y_{out} | y_1 > 0.4, y_4) = 1.557 - 0.964984 = 0.592$

We choose the feauture that maximizes the information gain for each level. In this case, we will choose y_3.



- For $y_3 = 0$, we have only one observation ({B}), therefore the node stops there.
- For $y_3 = 1$, we only have two observations ({A, B}), therefore the node stops there and we pick {A} - ascending alphabetic order.

- For $y_3 = 2$, we will need to calculate information gain again, in order to know if we pick y_2 or y_3.

Aprendizagem 2022/23

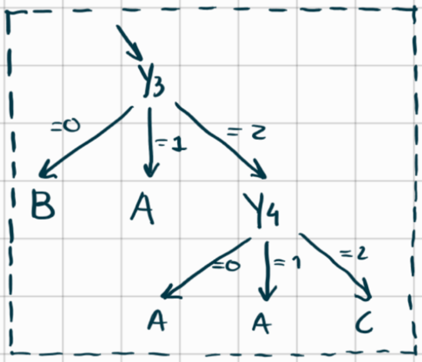
Homework I – Group 001

(ist1102082, ist1103252)

• $IG(y_{out} | y_1 > 0.4, y_3 = 2, y_2) = \frac{1}{4}\left(-\left(0+0+1\log_2(1)\right)\right) + \frac{1}{4}\left(-\left(0+1\log_2(1)+0\right)\right) + \frac{1}{2}\left(-\left(1\log_2(1)+0+0\right)\right) = 0$

• $IG(y_{out} | y_1 > 0.4, y_3 = 2, y_4) = \frac{1}{2}\left(-\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right)+0+\frac{1}{2}\log_2\left(\frac{1}{2}\right)\right)\right) + \frac{1}{4}\left(-\left(1\log_2(1)+0+0\right)\right) + \frac{1}{4}\left(-\left(0+0+1\log_2(1)\right)\right) = \frac{1}{2}$

• Since the information gain using y_4 is higher, that's our pick for $y_3 = 2$.



• For $y_4 = 0$, we have two observations ({A, C}), therefore the node stops there and we pick {A} - descending alphabetic order

• For $y_4 = 1$, we have one observation ({A}), therefore the node stops there.

• For $y_4 = 2$, we only have one observation ({C}), therefore the node stops there.

2)

2)

		REAL		
		A	B	C
PREDICTED	A	4	1	1
	B	0	2	0
	C	0	1	3

3)

3) $F1\text{-}score = 2 \times \frac{precision \times recall}{precision + recall}$, $recall = \frac{TP}{TP+FN}$, $precision = \frac{TP}{TP+FP}$

$Recall_A = \frac{4}{4+0+0} = 1$, $Precision_A = \frac{4}{4+1+1} = \frac{2}{3}$, $F_1\text{-}score_A = 2 \times \frac{1 \times \frac{2}{3}}{1 + \frac{2}{3}} = \frac{4}{5}$

$Recall_B = \frac{2}{2+1+1} = \frac{1}{2}$, $Precision_B = \frac{2}{2+0+0} = 1$, $F_1\text{-}score_B = 2 \times \frac{\frac{1}{2} \times 1}{\frac{1}{2} + 1} = \frac{2}{3}$

$Recall_C = \frac{3}{3+1+0} = \frac{3}{4}$, $Precision_C = \frac{3}{3+0+1} = \frac{3}{4}$, $F_1\text{-}score_C = 2 \times \frac{\frac{3}{4} \times \frac{3}{4}}{\frac{3}{4} + \frac{3}{4}} = \frac{3}{4}$

The class with the lowest F1-score is B.

4) Answer 4

4) Spearman

Ordered $y_1 = [0.04, 0.06, 0.24, 0.32, 0.36, 0.44, 0.46, 0.52, 0.62, 0.68, 0.76, 0.9]$

ranks $\longrightarrow$ 1 2 3 4 5 6 7 8 9 10 11 12

$y_1' = [3, 2, 1, 5, 4, 10, 12, 11, 7, 9, 6, 8]$

Ordered $y_2 = [0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 2, 2]$

ranks $\longrightarrow$ 3.5 3.5 3.5 3.5 3.5 3.5 8 8 8 11 11 11]

$y_2' = [8, 11, 3.5, 3.5, 3.5, 11, 3.5, 11, 8, 3.5, 8, 3.5]$

	y_1'	y_2'	$y_1'^2$	$y_2'^2$	$y_1' \times y_2'$
	3	8	9	64	24
	2	11	4	121	22
	1	3.5	1	12.25	3.5
	5	3.5	25	12.25	17.5
	4	3.5	16	12.25	14
	10	11	100	121	110
	12	3.5	144	12.25	42
	11	11	121	121	121
	7	8	49	64	56
	9	3.5	81	12.25	31.5
	6	8	36	64	48
	8	3.5	64	12.25	28
Σ	78	78	650	628.5	517.5

$$r = \frac{\operatorname{cov}(y_1, y_2)}{\sigma_{y_1} \times \sigma_{y_2}} = \frac{\Sigma y_1 y_2 - \frac{\Sigma y_1 \Sigma y_2}{n}}{\sqrt{\left(\Sigma y_1^2 - \frac{(\Sigma y_1)^2}{n}\right)\cdot\left(\Sigma y_2^2 - \frac{(\Sigma y_2)^2}{n}\right)}} = \frac{517.5 - \frac{78 \times 78}{12}}{\sqrt{\left(650 - \frac{78^2}{12}\right)\cdot\left(628.5 - \frac{78^2}{12}\right)}}$$

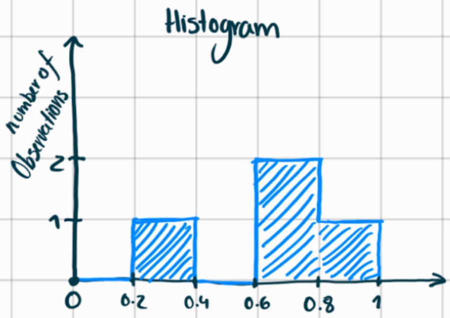
$$\simeq 0.07966$$

We conclude that y_1 and y_2, due to the low spearman value, have a low correlation.

5)

5)

Class A:



$Bin_{A\,]0,0.2]} = \emptyset$

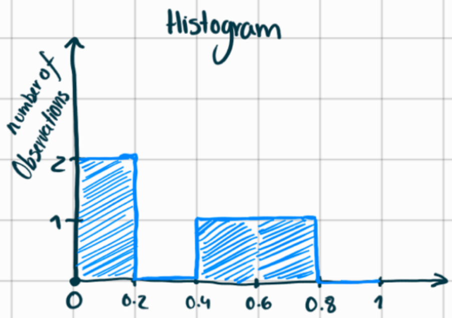
$Bin_{A\,]0.2,0.4]} = \{x_1\}$

$Bin_{A\,]0.4,0.6]} = \emptyset$

$Bin_{A\,]0.6,0.8]} = \{x_6, x_8\}$

$Bin_{A\,]0.8,1]} = \{x_7\}$

Class B:



$Bin_{B\,]0,0.2]} = \{x_2, x_3\}$

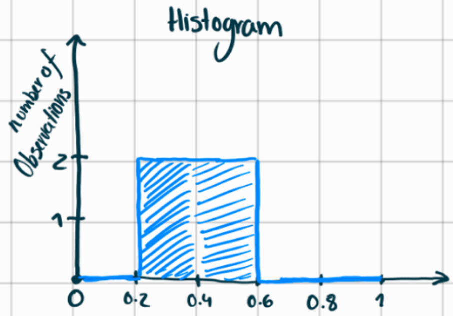
$Bin_{B\,]0.2,0.4]} = \emptyset$

$Bin_{B\,]0.4,0.6]} = \{x_9\}$

$Bin_{B\,]0.6,0.8]} = \{x_{10}\}$

$Bin_{B\,]0.8,1]} = \emptyset$

Class C:



$Bin_{C\,]0,0.2]} = \emptyset$

$Bin_{C\,]0.2,0.4]} = \{x_4, x_5\}$

$Bin_{C\,]0.4,0.6]} = \{x_{11}, x_{12}\}$

$Bin_{C\,]0.6,0.8]} = \emptyset$

$Bin_{C\,]0.8,1]} = \emptyset$

Challenge: Using the discriminant rules from these empirical distributions

In the range $]0, 0.2]$ the most discriminant is Class B $\Leftrightarrow$ $Bin_{]0,0.2]} = B$

In the range $]0.2, 0.4]$ the most discriminant is Class C $\Leftrightarrow$ $Bin_{]0.2,0.4]} = C$

In the range $]0.4, 0.6]$ the most discriminant is Class C $\Leftrightarrow$ $Bin_{]0.4,0.6]} = C$

In the range $]0.6, 0.8]$ the most discriminant is Class A $\Leftrightarrow$ $Bin_{]0.6,0.8]} = A$

In the range $]0.8, 1]$ the most discriminant is Class A $\Leftrightarrow$ $Bin_{]0.8,1]} = A$

II. Programming and critical analysis

Aprendizagem 2022/23

Homework I – Group 001

(ist1102082, ist1103252)

1)

Loading the data from the arff file and converting it into a dataframe

```
from scipy.io.arff import loadarff
import pandas as pd

# Load the data
data = loadarff('./column_diagnosis.arff')
df = pd.DataFrame(data[0])
df['class'] = df['class'].str.decode('utf-8')
df.head()
```

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis	class
0	63.027817	22.552586	39.609117	40.475232	98.672917	-0.254400	Hernia
1	39.056951	10.060991	25.015378	28.995960	114.405425	4.564259	Hernia
2	68.832021	22.218482	50.092194	46.613539	105.985135	-3.530317	Hernia
3	69.297008	24.652878	44.311238	44.644130	101.868495	11.211523	Hernia
4	49.712859	9.652075	28.317406	40.060784	108.168725	7.918501	Hernia

Separating the input data and the output data, required for some sklean functions

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

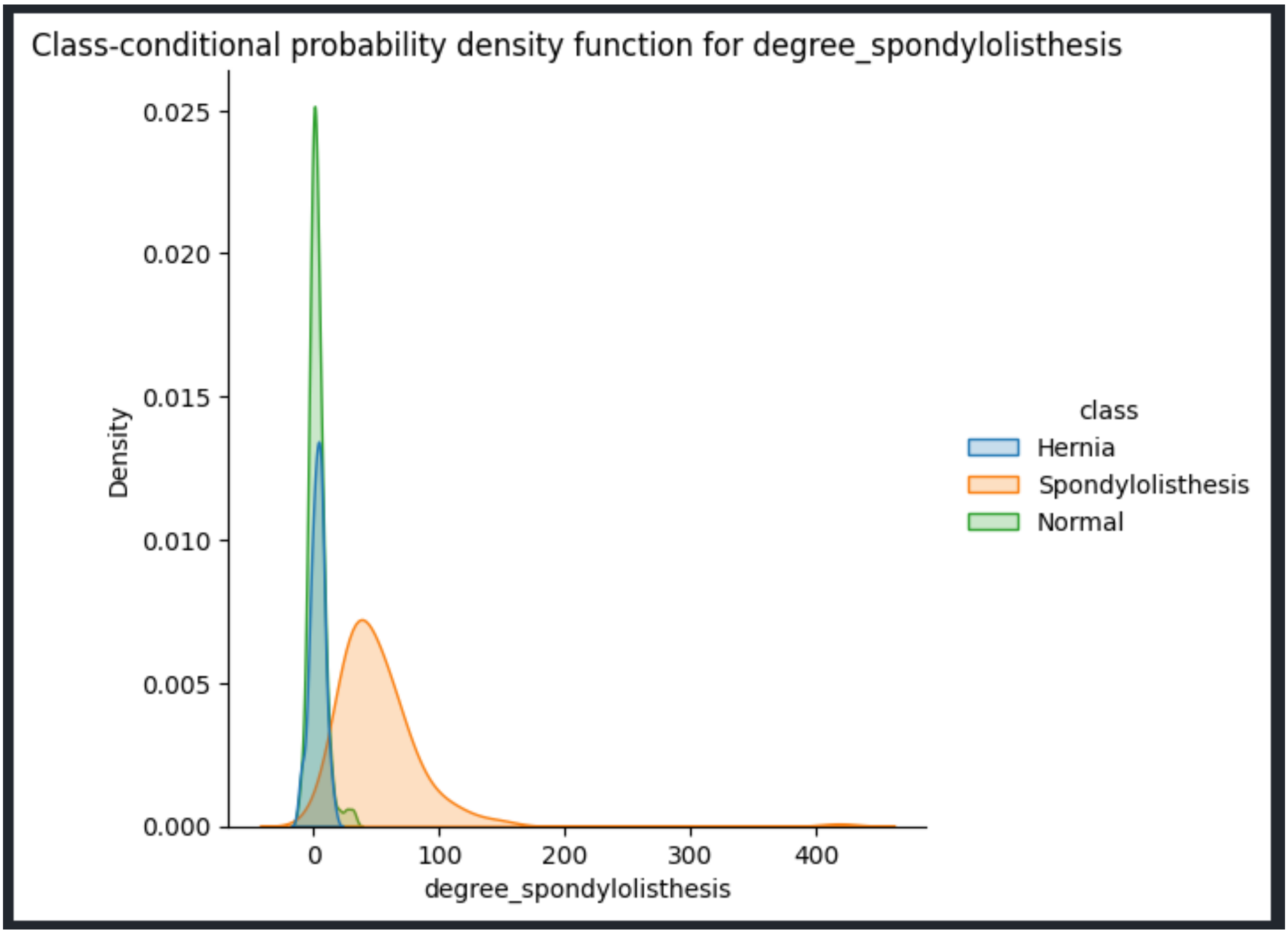
Checking the discriminative power of each feature in accordance with f_classif criterion. The higher the value, the more discriminative the feature is.

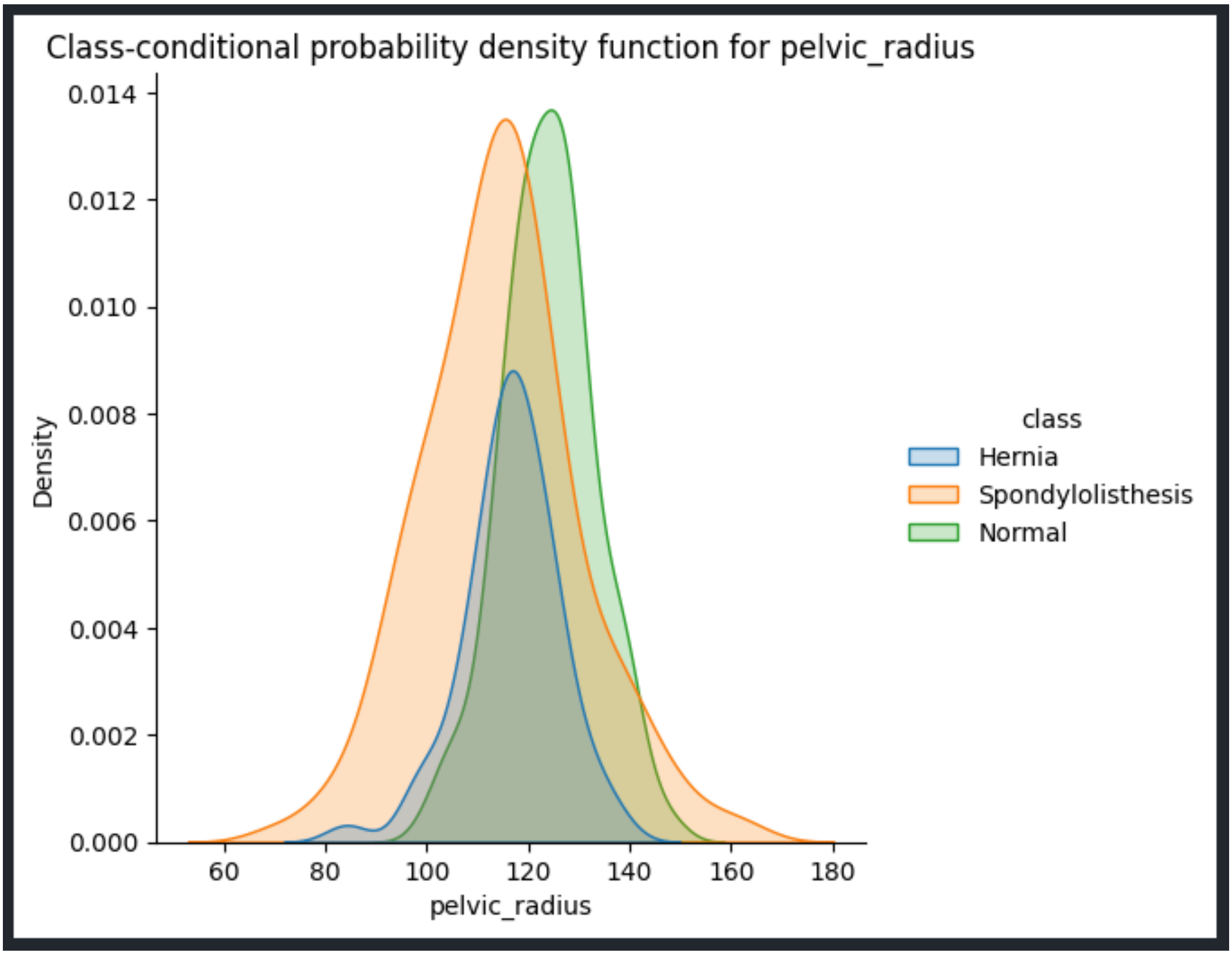
```
from sklearn.feature_selection import f_classif

fimportance = f_classif(x, y)

scores = fimportance[0]

highest_discriminative_power = x.columns.values[scores.argmax()]
lowest_discriminative_power = x.columns.values[scores.argmin()]
print('Input variable with highest discriminative power: ', highest_discriminative_power)
print('Input variable with lowest discriminative power: ', lowest_discriminative_power)
```

```
Input variable with highest discriminative power:  degree_spondylolisthesis
Input variable with lowest discriminative power:  pelvic_radius
```

Plotting the class-conditional probability density functions of these two input variables.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Plot the class-conditional probability density functions of these two input variables.

sns.displot(data=df, x=highest_discriminative_power, kind='kde', hue='class', fill=True)
plt.title('Class-conditional probability density function for ' + highest_discriminative_power)
plt.show()

sns.displot(data=df, x=lowest_discriminative_power, kind='kde', hue='class', fill=True)
plt.title('Class-conditional probability density function for ' + lowest_discriminative_power)
plt.show()
```





2)

First, we will split the data into training and testing sets. We will use 70% of the data for training and 30% for testing.

```
from sklearn import metrics, tree
from sklearn.model_selection import train_test_split
import numpy as np

# Split the data into training and test sets
avg_train_accs, avg_test_accs = [], []
# Define the depth limits
depth_limits = [1,2,3,4,5,6,8,10]
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(x, y, train_size=0.7, random_state=0, stratify=y)
```

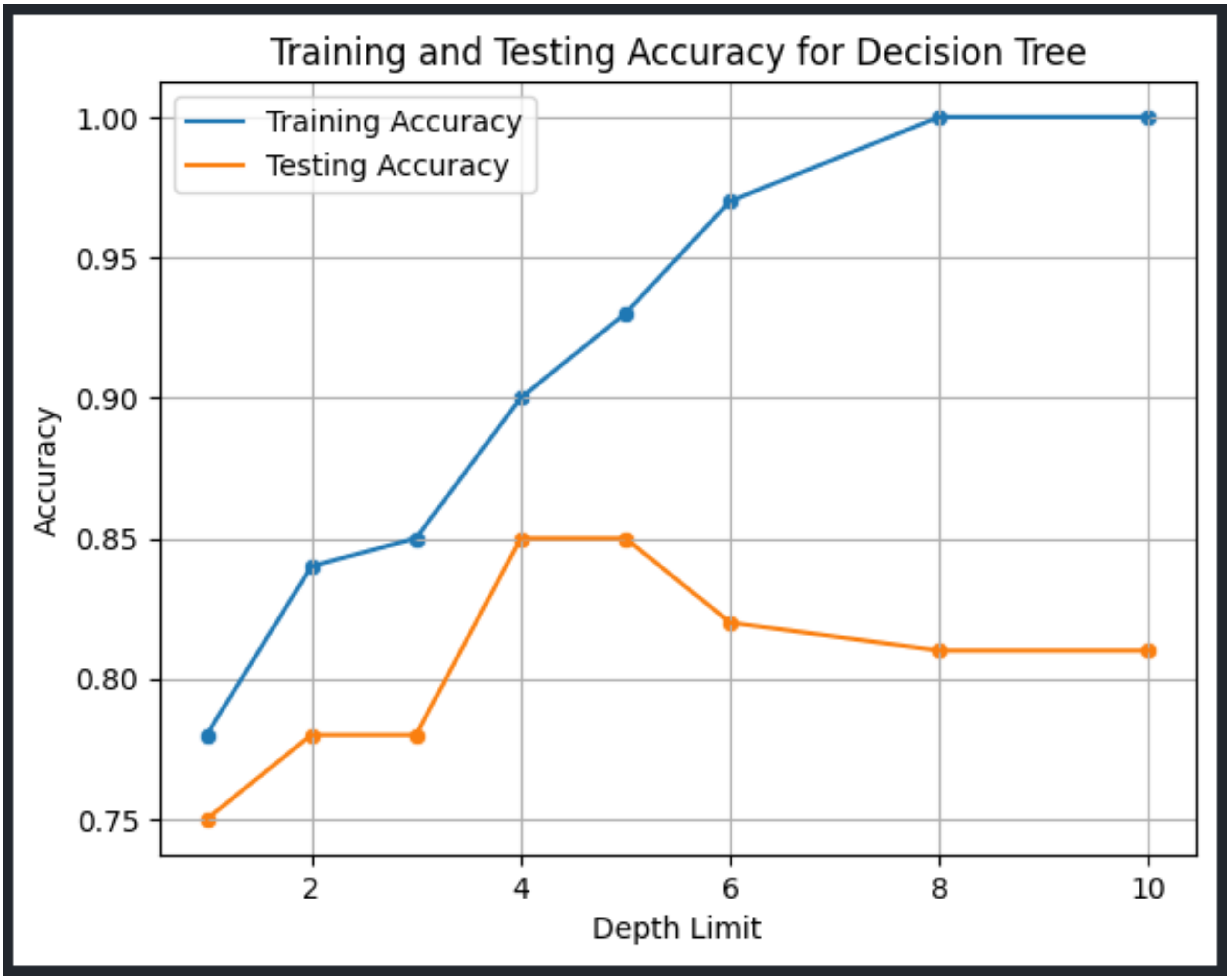
Now, we will use the training data to train the model. We will use the DecisionTreeClassifier from sklearn.tree to train the model. We will be doing 10 runs of the model and will be averaging the accuracy of the model.

```python
n_runs = 10
for i in depth_limits:
    train_accs, test_accs = [], []
    for j in range(n_runs):
        # train classifier
        predictor = tree.DecisionTreeClassifier(max_depth=i, random_state=0)
        # fit classifier
        predictor.fit(X_train, y_train)
        # test classifier
        y_pred_test = predictor.predict(X_test)
        y_pred_train = predictor.predict(X_train)

        # calculate accuracy
        train_acc = round(metrics.accuracy_score(y_train, y_pred_train),2)
        test_acc = round(metrics.accuracy_score(y_test, y_pred_test),2)

        train_accs.append(train_acc)
        test_accs.append(test_acc)

    avg_train_accs.append(np.mean(train_accs))
    avg_test_accs.append(np.mean(test_accs))
```

Python

Assess in a single plot both the training and testing accuracies of a decision tree with depth limits

```python
sns.lineplot(x=depth_limits, y=avg_train_accs, label='Training Accuracy')
sns.lineplot(x=depth_limits, y=avg_test_accs, label='Testing Accuracy')
sns.scatterplot(x=depth_limits, y=avg_train_accs)
sns.scatterplot(x=depth_limits, y=avg_test_accs)
plt.grid()
plt.xlabel('Depth Limit')
plt.ylabel('Accuracy')
plt.title('Training and Testing Accuracy for Decision Tree')
plt.show()
```



3)

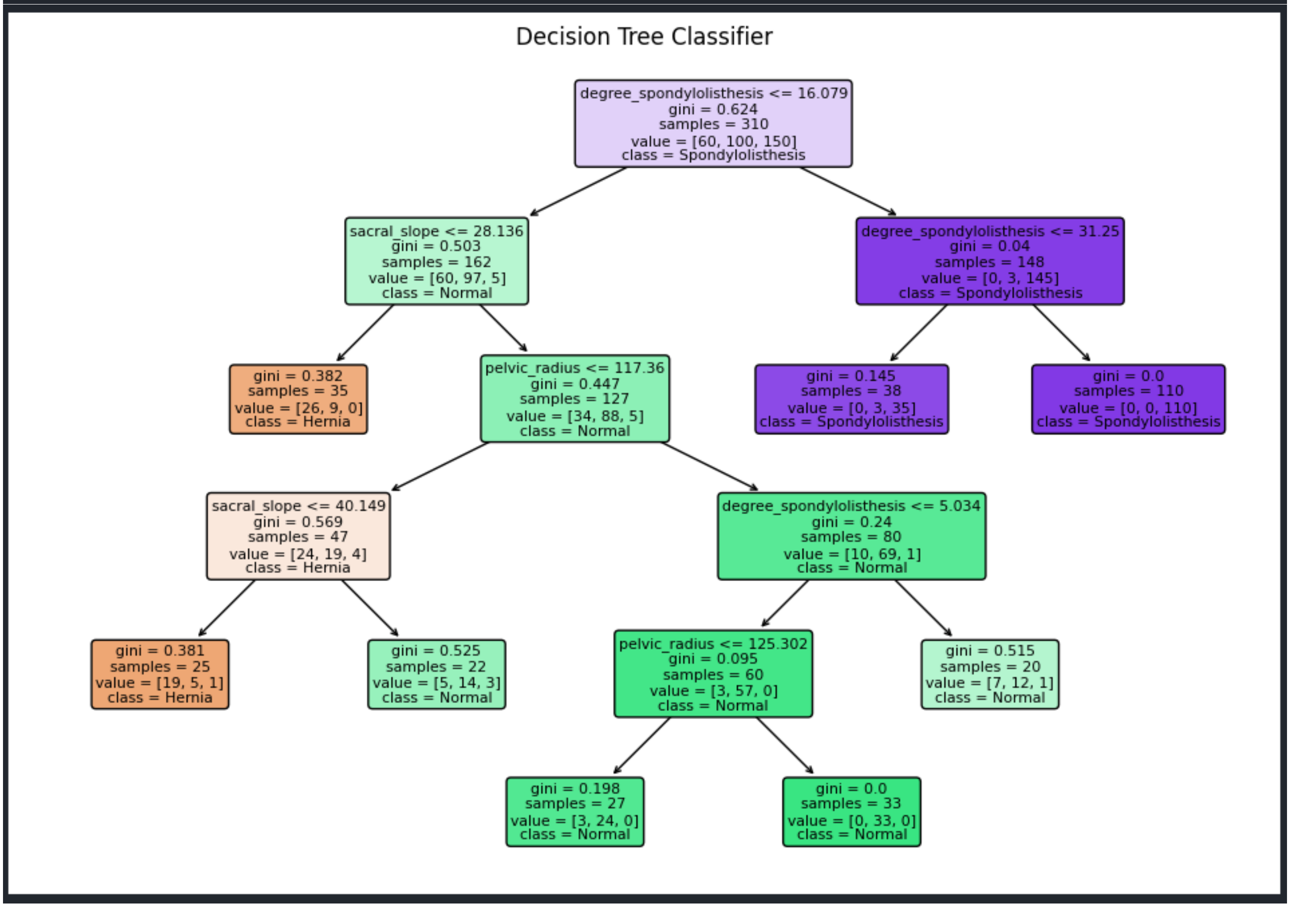
In the results from Question 2, we can observe the relationship between the max depth of the decision tree and its training and testing accuracies. We can see some key points from the plot:

- As the max depth of the decision tree increases, the training accuracy generally improves. This is expected because a deeper tree can fit the training data better, and hence, the training accuracy will improve.
- When analyzing the testing accuracy, it shows a different trend. Initially, as the tree uses a smaller depth limits, the testing accuracy improves. But, after a certain point (around depth of 5), the testing accuracy starts to decline. This is a clear sign of overfitting. The model is overfitting the training data and hence, the testing accuracy is declining.
- Considering the prior analysis, the optimal max depth for the decision tree is 5. Beyond that point, the model tens to overfit the training data and hence, the testing accuracy starts to decline.

4)

i)

```python
# Create a Decision Tree Classifier with the specified parameters
predictor = tree.DecisionTreeClassifier(random_state=0, min_samples_leaf=20)

# Fit the classifier on all available data
predictor.fit(x, y)

# Plot the decision tree
plt.figure(figsize=(12, 8))
tree.plot_tree(predictor, filled=True, feature_names=x.columns.values, class_names=predictor.classes_ , rounded=True)
plt.title("Decision Tree Classifier")
plt.show()
```



ii)

The conditions to indentify an hernia are, according to the decision tree are:

- a degree_spondylolisthesis value less than 16.079 and a sacral_slope value less than 28.136.
- a degree_spondylolisthesis value less than 16.079 and a pelvic_radius value less than 117.36 and a sacral_slope value less than 40.149.

END