Aprendizagem 2023/24

Homework I – Group 004

(ist102695, ist102703)

I. Pen-and-paper

1)

$IG(z|y_3)$, $IG(z|y_4)$ $\quad z = y_{out} | y_1 > 0.4$

$$H(z) = H(y_{out}|y_1 > 0.4) = -\left(\tfrac{3}{7} \times \log \tfrac{3}{7} + \tfrac{2}{7} \times \log \tfrac{2}{7} + \tfrac{2}{7} \times \log \tfrac{2}{7}\right) = 1.556656707$$

$$H(z|y_3) = \tfrac{1}{7}\left(-(0 + 1 \times \log 1 + 0)\right) + \tfrac{2}{7}\left(-(\tfrac{1}{2}\log\tfrac{1}{2} + \tfrac{1}{2}\log\tfrac{1}{2} + 0)\right) + \tfrac{4}{7}\left(-(\tfrac{1}{2}\log\tfrac{1}{2} + \tfrac{1}{2}\log\tfrac{1}{2})\right) = \tfrac{6}{7} = 0.8571428571$$

$$H(z|y_4) = \tfrac{2}{7}\left(-(\tfrac{1}{2}\log\tfrac{1}{2} + \tfrac{1}{2}\log\tfrac{1}{2})\right) + \tfrac{3}{7}\left(-(\tfrac{1}{3}\log\tfrac{1}{3} + \tfrac{2}{3}\log\tfrac{2}{3} + 0)\right) + \tfrac{2}{7}\left(-(\tfrac{1}{2}\log\tfrac{1}{2} + \tfrac{1}{2}\log\tfrac{1}{2})\right) = 0.9649839289$$

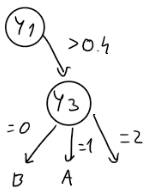
$$H(z|y_2) = \tfrac{3}{7}\left(-(3 \times \tfrac{1}{3}\log\tfrac{1}{3})\right) + \tfrac{2}{7}\left(-(\tfrac{1}{2}\log\tfrac{1}{2} + \tfrac{1}{2}\log\tfrac{1}{2})\right) + \tfrac{2}{7}\left(-(1 \log 1)\right) = 0.9649839289$$

$IG(z|y_3) = 1.556656707 - 0.8571428571 = 0.69951385$

$IG(z|y_4) = 1.556656707 - 0.9649839289 = 0.5916727781$

$IG(z|y_2) = 1.556656707 - 0.9649839289 = 0.5916727781$

$IG(z|y_3)$ é o maior, logo a 1ª variável a aparecer será y_3



$y_3 = 0 \rightarrow B$

$y_3 = 1 \rightarrow A, B$

$y_3 = 2 \rightarrow A, C$

$P(A) = \tfrac{1}{2}$ $P(B) = \tfrac{1}{2}$ → em caso de empate usar ordem alfabética segundo ii)

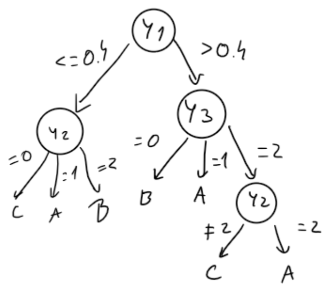
$P(A) = \tfrac{2}{4}$ $P(B) = \tfrac{2}{4}$ → Com um mínimo de 4 observações podemos dividir um nó segundo i)

$z = y_1 > 0.4 \wedge y_3 = 2$

$$H(z|y_2) = \tfrac{1}{4}(-\log 1) + \tfrac{1}{4}(-\log 1) + \tfrac{2}{4}(-\log 1) = 0$$

logo $IG(z|y_2)$ é maior e é essa variável que devemos selecionar a seguir

$$H(z|y_4) = \tfrac{2}{4}\left(-(\tfrac{1}{2}\log\tfrac{1}{2} + \tfrac{1}{2}\log\tfrac{1}{2})\right) + \tfrac{1}{4}(-\log 1) + \tfrac{1}{4}(-\log 1) = 0.5$$



2)

		Real		
		A	B	C
previsão	A	4	1	0
	B	0	2	0
	C	0	1	4

3)

3) $F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$ $\quad$ $\text{precision} = \frac{TP}{TP + FP}$

$\text{recall} = \frac{TP}{TP + FN}$

para A: $TP = 4$
$FP = 1$
$FN = 0$
$\text{prec.} = \frac{4}{5}$ $\quad$ $\text{rec.} = \frac{4}{4} = 1$

$F1 = 2 \times \frac{4/5 \times 1}{4/5 + 1} = 8/9 \simeq 0.8889$

para B: $TP = 2$
$FP = 0$
$FN = 2$
$\text{prec.} = \frac{2}{2} = 1$
$\text{rec} = \frac{2}{4} = 0.5$

$F1 = 2 \times \frac{1 \times 0.5}{1 + 0.5} = 6/9 \simeq 0.6667$

para C: $TP = 4$
$FP = 1$
$FN = 0$
$\text{prec} = \frac{4}{5}$
$\text{rec.} = 4/4 = 1$

$F1 = 2 \times \frac{4/5 \times 1}{4/5 + 1} = \frac{8}{9} \simeq 0.8889$

R: A classe B tem o F1 score mais baixo

4)

4. spearman:

y_1	0.24	0.06	0.04	0.36	0.32	0.68	0.9	0.76	0.46	0.62	0.44	0.52
R_{y_1}	3	2	1	5	4	10	12	11	7	9	6	8
y_2	1	2	0	0	0	2	0	2	1	0	1	0
R_{y_2}	8	11	3.5	3.5	3.5	11	3.5	11	8	3.5	8	3.5
$(y_1 \times \bar{y}_1) \times (y_2 \times \bar{y}_2)$	-5.25	-20.25	16.5	4.5	7.5	15.75	-16.5	20.25	0.75	-7.5	-0.75	-4.5

correlação de Pearson

$\bar{y}_1 : 6.5$

$y_2 : 6.5$

$$var(y_1) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$= \frac{\sum_{i=1}^{12} x_i^2 - n\bar{x}^2}{11}$$

$$= \frac{650 - 12(6.5)^2}{11}$$

$$= 13$$

$$var(y_2) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$= \frac{\sum^{n} (x_i)^2 - n\bar{x}^2}{n-1}$$

$$= \frac{628.5 - 12(6.5)^2}{11}$$

$$= 11.045$$

$$cov(y_1, y_2) = \frac{\sum_{i=1}^{n} (y_{1i} - \bar{y}_1) \times (y_{2i} - \bar{y}_2)}{n-1}$$

$$= \frac{10.5}{11}$$

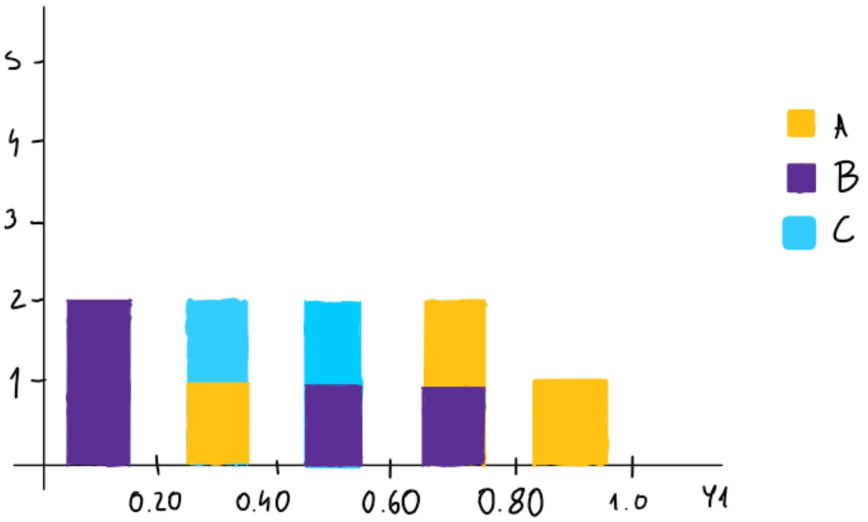
$$= 0.9545$$

$$r = \frac{Cov(y_1, y_2)}{\sigma(y_1) \times \sigma(y_2)}.$$

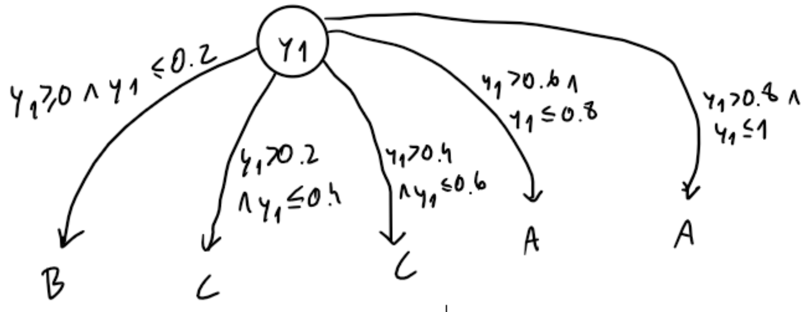
$$r = \frac{0.9545}{\sqrt{13} \times \sqrt{11.04545}}$$

$$= 0.0797 \Rightarrow \text{baixa correlação}$$

5)



Challenge



Aprendizagem 2023/24

Homework I – Group 004

(ist102695, ist102703)

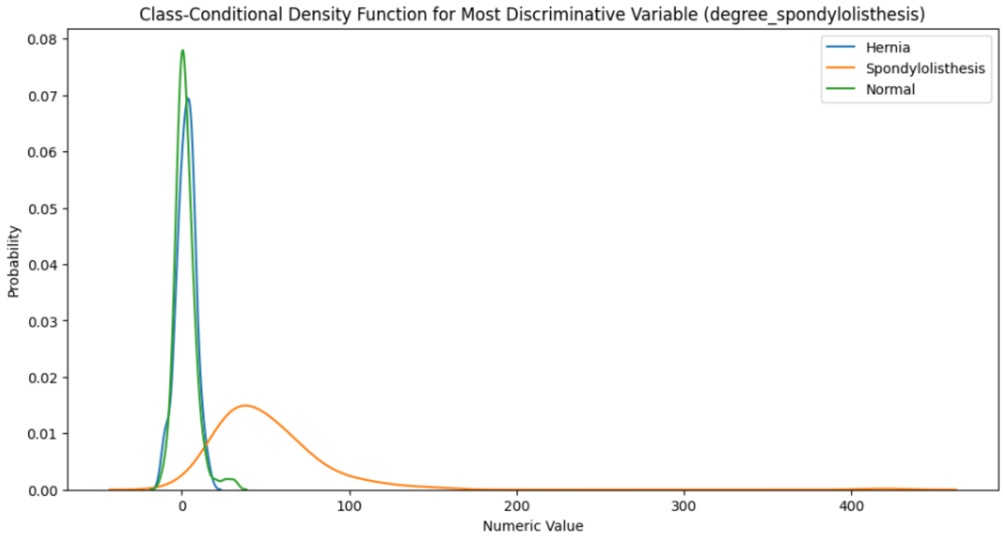
II. Programming and critical analysis

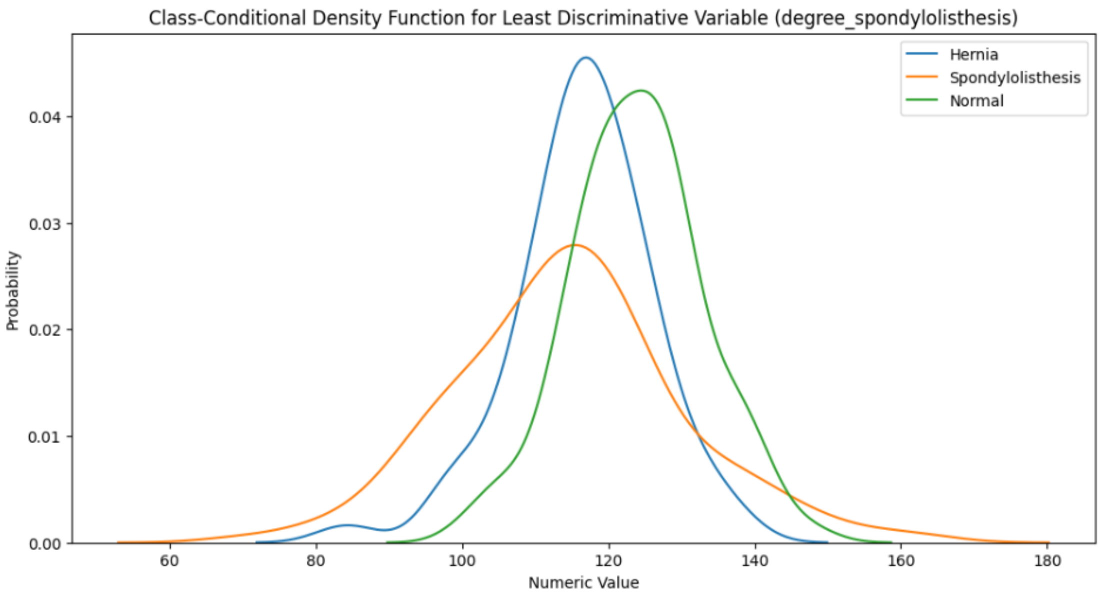
1)

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

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

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.feature_selection import f_classif

# Separate features from the outcome (class)
X = df.drop('class', axis=1)
y = df['class']

fimportance = f_classif(X, y)
results_df = pd.DataFrame({'Feature': X.columns, 'F-Score': fimportance[0]})

most_discriminative = results_df.sort_values(by='F-Score',ascending=False)

print("Most Discriminative Feature:")
print(most_discriminative['Feature'].iloc[0])
print("with F-Score:")
print(most_discriminative['F-Score'].iloc[0])

print("\nLeast Discriminative Feature:")
print(most_discriminative['Feature'].iloc[5])
print(most_discriminative['F-Score'].iloc[5])

# Plot for the most discriminative feature
plt.figure(figsize=(12, 6))
most_discriminative_feature = most_discriminative['Feature'].iloc[0]
for c in df['class'].unique():
    c_data = df[df['class'] == c]
    sns.kdeplot(data=c_data, x=most_discriminative_feature, label=c)

plt.xlabel('Numeric Value')
plt.ylabel('Probability')
plt.title(f'Class-Conditional Density Function for Most Discriminative Variable ({most_discriminative_feature})')
plt.legend()

# Create a new figure for the least discriminative feature
plt.figure(figsize=(12, 6))
least_discriminative_feature = most_discriminative['Feature'].iloc[5]
for c in df['class'].unique():
    c_data = df[df['class'] == c]
    sns.kdeplot(data=c_data, x=least_discriminative_feature, label=c)

plt.xlabel('Numeric Value')
plt.ylabel('Probability')
plt.title(f'Class-Conditional Density Function for Least Discriminative Variable ({most_discriminative_feature})')
plt.legend()
plt.show()
```

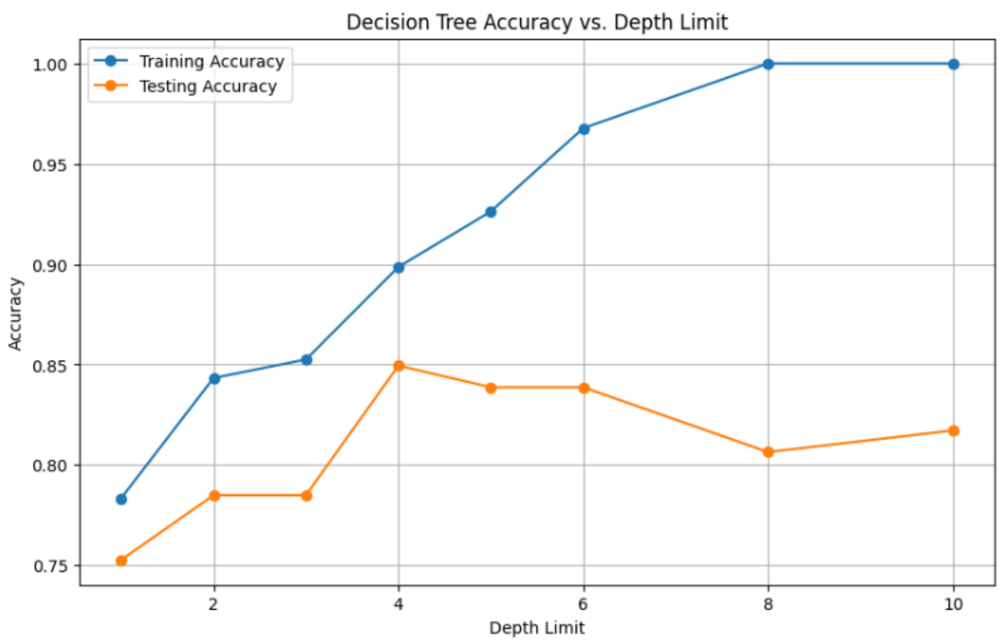
```
Most Discriminative Feature:
degree_spondylolisthesis
with F-Score:
119.12288060759764

Least Discriminative Feature:
pelvic_radius
16.86693475538006
```





2)

```
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn import metrics, tree
from sklearn.metrics import accuracy_score

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

# Initialize lists to store training and testing accuracies
train_accuracies = []
test_accuracies = []

for limit in depth_limits:
    X_train, X_test, y_train, y_test=train_test_split(X, y, train_size=0.7, stratify = y, random_state=0)
    predictor = tree.DecisionTreeClassifier(max_depth = limit)
    predictor.fit(X_train, y_train)
    # Calculate training accuracy
    train_accuracy = accuracy_score(y_train, predictor.predict(X_train))
    train_accuracies.append(train_accuracy)

    # Calculate testing accuracy
    test_accuracy = accuracy_score(y_test, predictor.predict(X_test))
    test_accuracies.append(test_accuracy)

# Plot the training and testing accuracies
plt.figure(figsize=(10, 6))
plt.plot(depth_limits, train_accuracies, label='Training Accuracy', marker='o')
plt.plot(depth_limits, test_accuracies, label='Testing Accuracy', marker='o')
plt.title('Decision Tree Accuracy vs. Depth Limit')
plt.xlabel('Depth Limit')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)
plt.show()
```



Aprendizagem 2023/24

Homework I – Group 004

(ist102695, ist102703)

3)

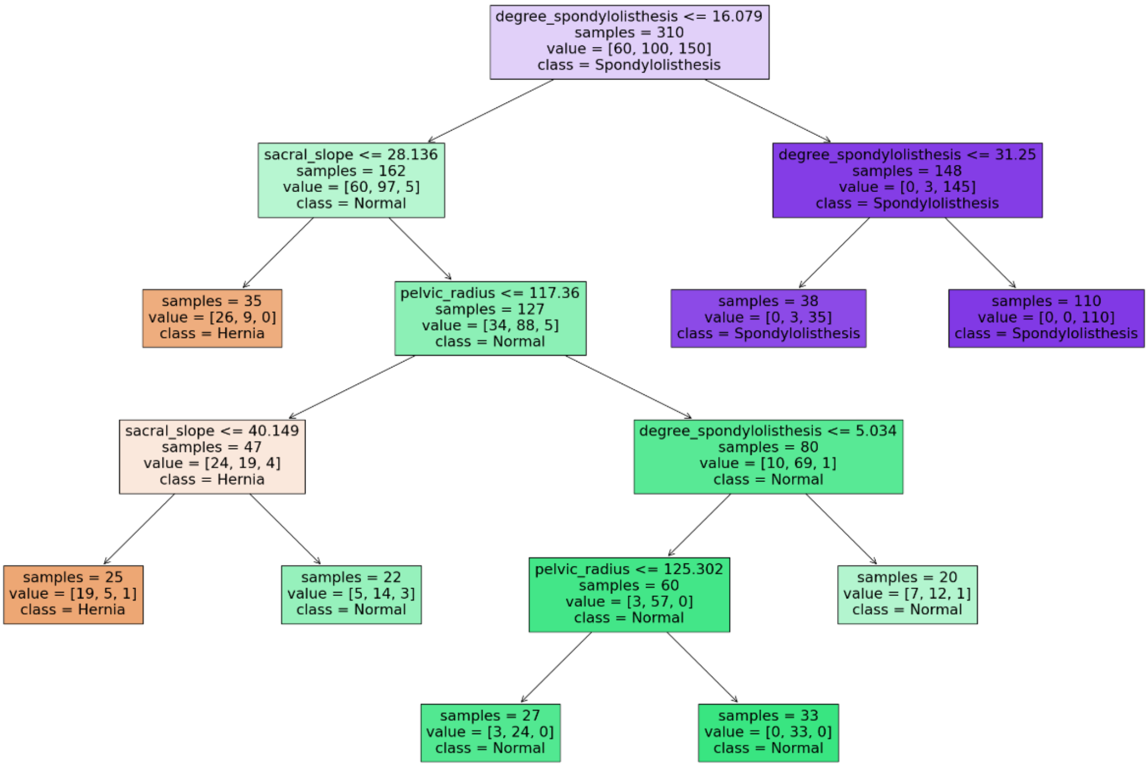
Com base no gráfico apresentado em 2) concluímos que a profundidade que melhor se adequa ao conjunto é 4 por ser o valor em que a accuracy de treino e de teste se encontram simultaneamente mais elevadas.

Observamos que a accuracy do conjunto de teste começa a baixar a partir do limite de profundidade de 4 pois, o excesso de profundidade faz com que os dados se ajustem demasiado ao conjunto de treino não conseguindo rotular com sucesso objetos ainda não observados, ou seja, estamos perante overfitting.

4)

a)

```python
import matplotlib.pyplot as plt
import numpy as np
from sklearn import tree
from scipy.io import arff
import pandas as pd

# 1. Load and preprocess data
data = arff.loadarff('column_diagnosis.arff')
df = pd.DataFrame(data[0])
df['class'] = df['class'].str.decode('utf-8')
in_vars = df.drop(columns=['class'])
out_vars = df['class']

# 2. Learn classifier
predictor = tree.DecisionTreeClassifier(min_samples_leaf=20)
predictor.fit(in_vars, out_vars)

# 3. Plot classifier
fig = plt.subplots(figsize=(30, 20))

tree.plot_tree(predictor, feature_names=in_vars.columns.tolist(), class_names=np.unique(out_vars).tolist(), filled=True, impurity=False)
plt.show()
```



b) Para alguém ter uma hernia, precisa de ter um degree_spondylolisthesis <= 16.079 e sacral slope <= 28.136. Caso tenha pelvic_radius <= 117.36, terá de ter sacral slope <= 40.149 e assim 25 pessoas da nossa amostra têm uma hernia. Caso contrário, temos 35 pessoas da nossa amostra que têm uma hernia.

END