Aprendizagem 2022/23

Homework I – Group 105

I. Pen-and-paper

1)

		true P	true N
prediction	P	TP: 8	FP: 4
	N	FN: 3	TN: 5

2)

$$precision = \frac{TP}{TP+FP} = \frac{8}{12} = \frac{2}{3}$$

$$recall = \frac{TP}{TP+FN} = \frac{8}{11}$$

$$F_1 = 2\,\frac{precision \cdot recall}{precision + recall} = \frac{2TP}{2TP + FP + FN} = \frac{16}{23}$$

3) Reason 1: Since every entry from the dataset in which *y1* = A, results in *class* = P, the tree also classifies *class* = P as the only possible result given *y1* = A.

Reason 2: The tree was no longer expanded as to maintain it as simples as possible.

4)

$$IG(class|y_1) = E(class) - E(class|y_1)$$

$$E(class) = -\sum_{i=\{P,N\}} [p(class=i).\log_2[p(class=i)]] = -\frac{11}{20}\log_2\left(\frac{11}{20}\right) - \frac{9}{20}\log_2\left(\frac{9}{20}\right) =$$

$$= 0.9927744539878$$

$$E(class|y_1) = \sum_{i=\{A,B\}} [p(y_1=i)] \sum_{j=\{P,N\}} [-p(class=j|y_1=i).\log_2[p(class=j|y_1=i)]]$$

$$= \frac{7}{20}.\left[-\frac{5}{7}\log_2\left(\frac{5}{7}\right) - \frac{2}{7}\log_2\left(\frac{2}{7}\right)\right] + \frac{13}{20}.\left[-\frac{6}{13}\log_2\left(\frac{6}{13}\right) - \frac{7}{13}\log_2\left(\frac{7}{13}\right)\right] = 0.9493150428535$$

$$IG(class|y_1) = 0.0434594111343$$

y_1	y_2	class
A	?	P
A	?	P
A	?	P
A	?	P
A	?	P
A	?	N
A	?	N
B	≥ 2	N
B	>2	N
B	>2	N
B	>2	N
B	>2	N
B	>2	P
B	>2	P
B	>2	P
B	≤ 2	P
B	≤ 2	P
B	≤ 2	P
B	≤ 2	N
B	≤ 2	N

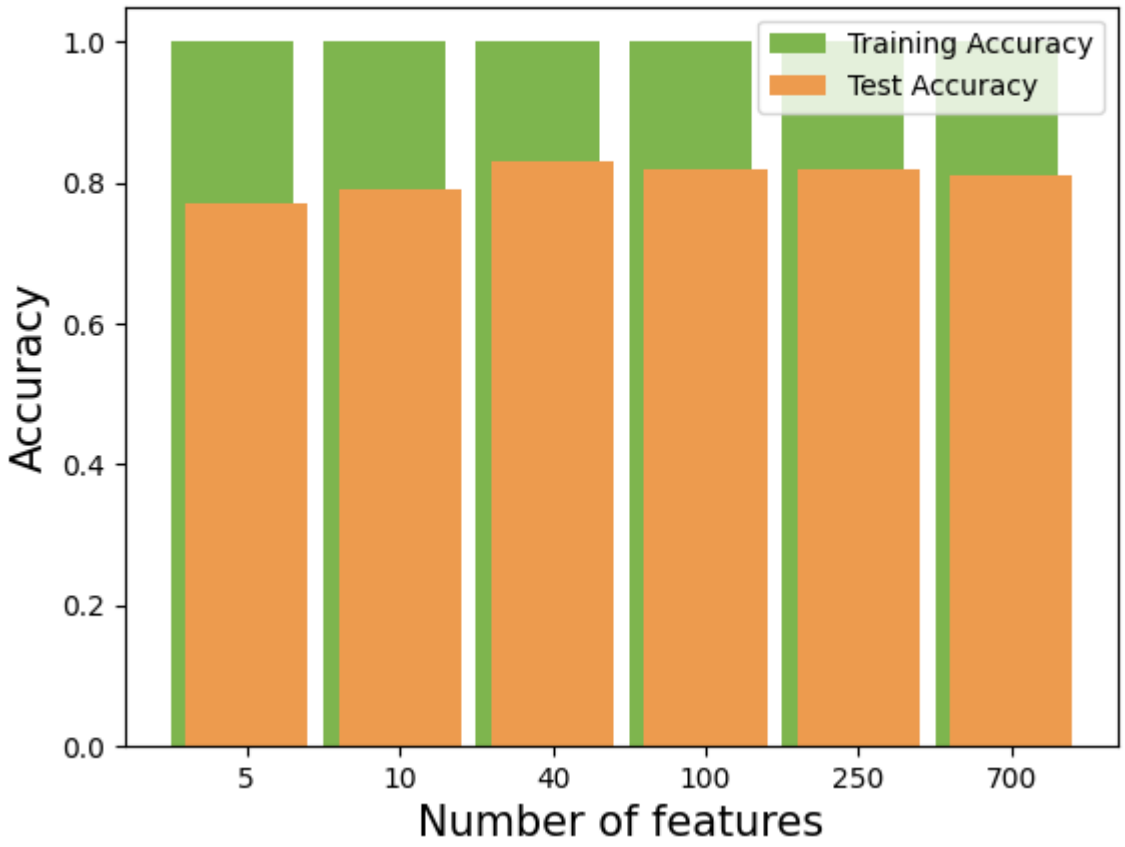
TÉCNICO LISBOA

Aprendizagem 2022/23

Homework I – Group 105

II. Programming and critical analysis

5)



6) In this case, we can see a massive improvement in the training accuracy (exactly 1) over the test accuracy (approx. 0.8) and while it is normal to have a better accuracy on the training set, such a big improvement, especially to a fixed 1 accuracy, draws us to the conclusion that our model is overfitted to the training dataset.

III. APPENDIX

```
from scipy.io.arff import loadarff
from sklearn import metrics, datasets, tree
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import mutual_info_classif
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

TÉCNICO LISBOA

Aprendizagem 2022/23

Homework I – Group 105

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

predictor = tree.DecisionTreeClassifier()
test_acc = []
train_acc = []

# 1. load and partition data
X, y = df[list(df.columns[:-1])], df[["class"]]

mutual_info = mutual_info_classif(X, y)
array_mutual_info = np.array(mutual_info)

for i in (5, 10, 40, 100, 250, 700):
    # Get the top i values
    indexes = np.argpartition(array_mutual_info, -i)[-i:]
    topi_names = []

    # Select the top i columns
    for j in range(i):
        topi_names += [df.columns[indexes[j]]]
    X = df[topi_names]

    # Train with the selected features
    X_train, X_test, y_train, y_test = train_test_split(X, y, stratify = y, train_size =
0.7, random_state = 1)

    # Predict after training
    predictor.fit(X_train, y_train)
    y_pred = predictor.predict(X_test)
    y_train_pred = predictor.predict(X_train)

    # Metrics calculation
    test_acc += [round(metrics.accuracy_score(y_test, y_pred), 2)]
    train_acc += [round(metrics.accuracy_score(y_train, y_train_pred), 2)]
```

```
bar1 = np.arange(6)
bar2 = [n + 0.1 for n in bar1]

plt.bar(bar1, train_acc, color = "#7eb54e", label = "Training Accuracy")
plt.bar(bar2, test_acc, color = "#ed9b4e", label = "Test Accuracy")

plt.xticks([x + 0.1 for x in range(6)], [5, 10, 40, 100, 250, 700])
plt.xlabel("Number of features", fontsize = 15)
plt.ylabel("Accuracy", fontsize = 15)
plt.legend()
plt.show()
```

END