TÉCNICO LISBOA

Aprendizagem 2022/23

Homework I – Group 009

I. Pen-and-paper

1)

		Real	
		P	**N**
Previsão	**P**	8	4
	N	3	5

2) Depois de um post-pruning da árvore de decisão com profundidade máxima de 1, a matriz de confusão passa a ser a seguinte:

		Real	
		P	**N**
Previsão	**P**	5	2
	N	6	7

Com base nesta nova matriz de confusão, conseguimos obter o Recall:

$$Recall = \frac{5}{5+6} = \frac{5}{11}$$

E a precisão:

$$Precisão = \frac{5}{5+2} = \frac{5}{7}$$

Após o cálculo do Recall e da Precisão, calculamos o seguinte training F1:

$$F_1 = \frac{1}{\frac{1}{2} x (\frac{1}{Precisão} + \frac{1}{Recall})} = \frac{1}{\frac{1}{2} x (\frac{11}{5} + \frac{7}{5})} = \frac{5}{9}$$

3) Duas razões possíveis são:

- Para evitar problema de overfitting;
- Por existirem observações com features iguais mas com resultados diferentes, o que pode levar a que decompor o caminho não aumente o ganho de informação.

4) O ganho de informação da variável y1 é dado pela seguinte fórmula:

$$IG(Output/y_1) = H(Output) - H(Output/y_1)$$

Onde H(Output) é dado por:

$$H(Output) = -\sum_{i=1}^{n} p(o_i) \log_2(p(o_i)) = -\frac{11}{20} \log_2(\frac{11}{20}) - \frac{9}{20} \log_2(\frac{9}{20}) = 0.993$$

H(Output / y1) é dado por:

$$H(Output/y_1)=p(y_1=A)H(Output/y_1=A)+p(y_1=B)H(Output/y_1=B)$$

Onde:

$$H(Output/y_1=A)=-\frac{5}{7}\log_2(\frac{5}{7})-\frac{2}{7}\log_2(\frac{2}{7})$$
$$H(Output/y_1=B)=-\frac{6}{13}\log_2(\frac{6}{13})-\frac{7}{13}\log_2(\frac{7}{13})$$

Portanto temos que:

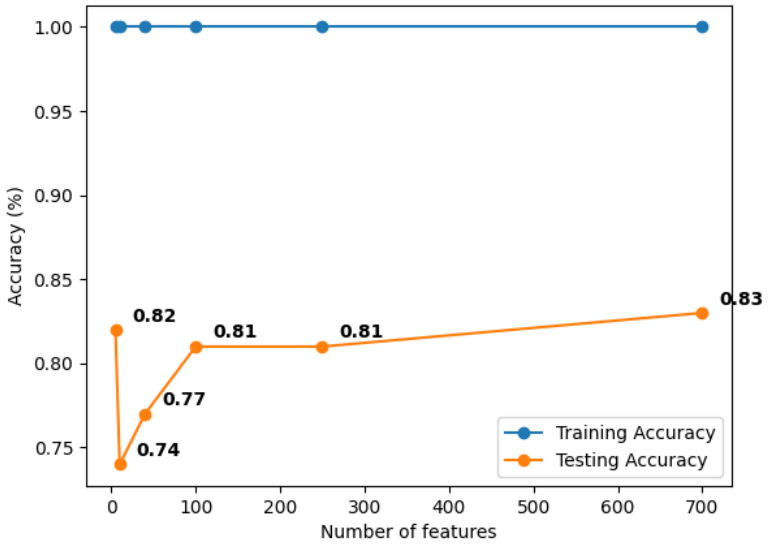
$$H(Output/y_1)=0.9493$$

Logo o ganho de informação da variável y1 é:

$$IG(Output/y_1)=0.993-0.9493=0.043$$

II. Programming and critical analysis

5)



6) Como se trata de uma árvore de decisão com critérios default, sem limites de profundidade, o crescimento da árvore só termina quando todas as instâncias do conjunto de treino estão corretamente classificadas ou quando não existem mais variáveis disponíveis. Portanto, a training accuracy será sempre de 100%, desde que não existam duas ou mais observações com as mesmas features mas com resultados diferentes. Logo como não existem observações com as features iguais mas com resultados diferentes no nosso conjunto de treino, a training accuracy é sempre de 100%.

III. APPENDIX

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

def getData(dataFrame,scores, numberFeatures):
	features = []
	for i in range(numberFeatures):
		features.append(scores[i][0])
	return dataFrame[features]

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

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

minfo = mutual_info_classif(X,y,random_state=1)
scores = {}
for i in range(len(minfo)):
	scores[X.columns.values[i]] = minfo[i]

sort_scores = sorted(scores.items(), key=lambda x: x[1], reverse=True)

features_options = [5,10,40,100,250,700]
training_accuracy = []
testing_accuracy = []
predictor = DecisionTreeClassifier(random_state=1)

for number_of_features in features_options:
	X_selected = getData(df,sort_scores, number_of_features)
	X_train, X_test, y_train, y_test = train_test_split(X_selected,y, stratify=y, train_size=0.7,
random_state=1)
	predictor.fit(X_train, y_train)
	y_train_pred = predictor.predict(X_train)
	y_test_pred = predictor.predict(X_test)
	training_accuracy.append(round(metrics.accuracy_score(y_train,y_train_pred),2))
	testing_accuracy.append(round(metrics.accuracy_score(y_test, y_test_pred),2))

plt.plot(features_options,training_accuracy, label="Training Accuracy", marker='o')
plt.plot(features_options,testing_accuracy, label="Testing Accuracy", marker='o')
plt.xlabel('Number of features')
plt.ylabel('Accuracy (%)')
plt.legend()
for a,b in zip(features_options, testing_accuracy):
   plt.text(a+20, b+0.005, str(b),fontweight='bold')

plt.show()
```

END