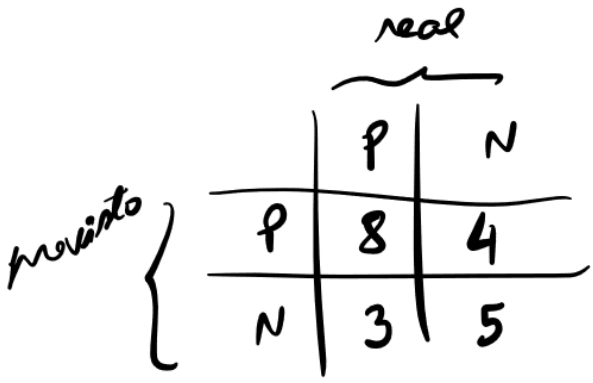
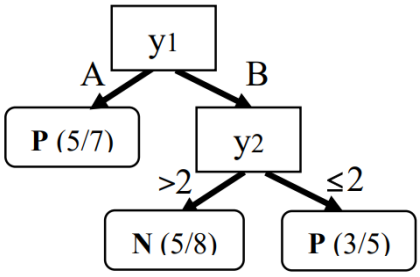
Aprendizagem 2021/22

Homework I – Group 104

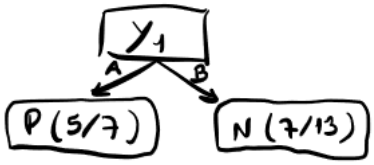
I. Pen-and-paper

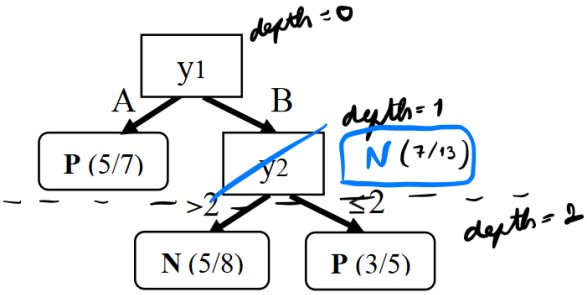
1) Training confusion matrix:



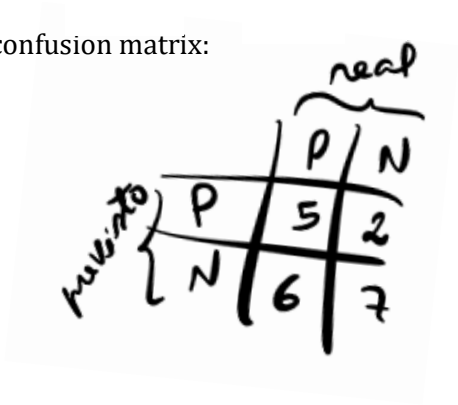


2) Pruning of the given tree (maximum depth of 1):





New confusion matrix:



F-measure calculation:

$$F_1 = \frac{1}{0.5\frac{1}{p} + 0.5\frac{1}{r}} = \frac{1}{0.5 \times \frac{7}{5} + 0.5 \times \frac{11}{5}} \simeq 0.56$$

$p \rightarrow$ precision
$r \rightarrow$ recall

$$\text{recall} = \frac{5}{11}$$

$$\text{precision} = \frac{5}{7}$$

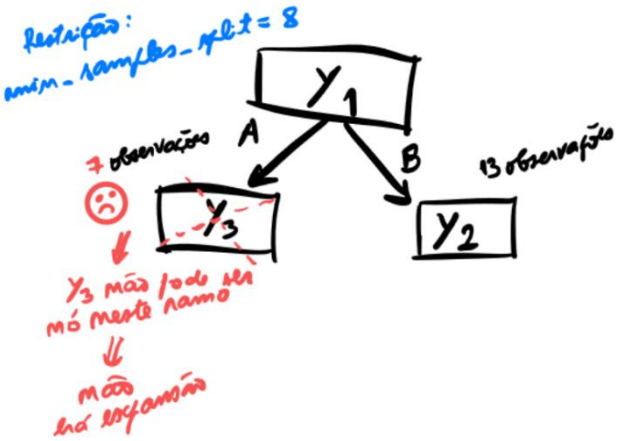
Aprendizagem 2021/22

Homework I – Group 104

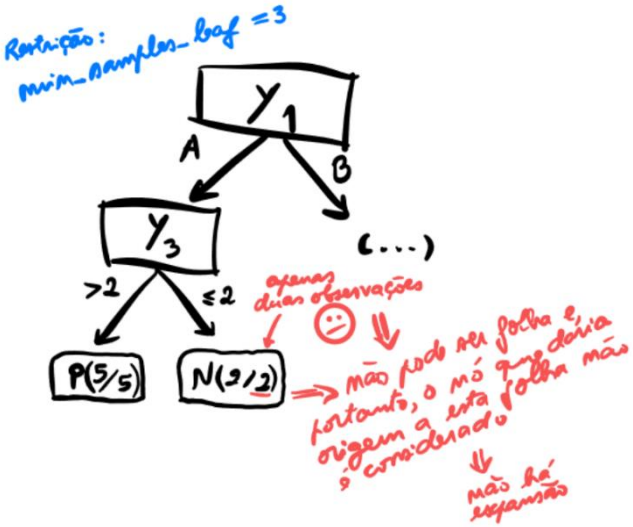
3) Reason 1: the left branch didn't expand because there was the risk of overfitting – the model would overfit the results to the specific training dataset that was given, meaning it could turn out to be too specific for it and, ultimately, the model would, then, make decisions based on a very low number of observations, compromising the results (in fact, the number of observations of this dataset is low – 20 – with only 7 on the left branch).

Reason 2: to limit the expansion of the branch, in this specific case, we could:

1) define a minimum number of observations for a certain feature (input variable) to be considered valid to be included in the Decision Tree (during training) – min_samples_split*



2) define a minimum number of observations for the tree leafs, after the node split, so that such node would only be on the tree if both the left and right branches had more observations than the stipulated – min_samples_leaf*



TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 104

* Parameters of the function DecisionTreeClassifier from the sklearn library

4) Computation of y1's information gain:

$$H(z) = -\frac{12}{20}\log\frac{12}{20} - \frac{8}{20}\log\frac{8}{20} \simeq 0.971$$

$$H(z|y_1) = \frac{7}{20}H(z|y_1=A) + \frac{13}{20}H(z|y_2=B)$$

$$= \frac{7}{20}\left(-\frac{5}{7}\log\frac{5}{7} - \frac{2}{7}\log\frac{2}{7}\right) + \frac{13}{20}\left(-\frac{6}{13}\log\frac{6}{13} - \frac{7}{13}\log\frac{7}{13}\right)$$

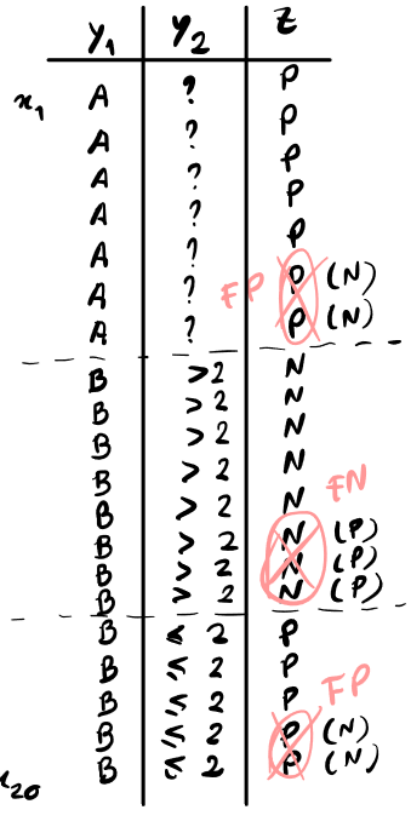
$$\simeq 0.949$$

$$IG(z|y_1) = H(z) - H(z|y_1)$$

$$= 0.971 - 0.949$$

$$\simeq 0.022$$

Auxiliar:



TÉCNICO LISBOA

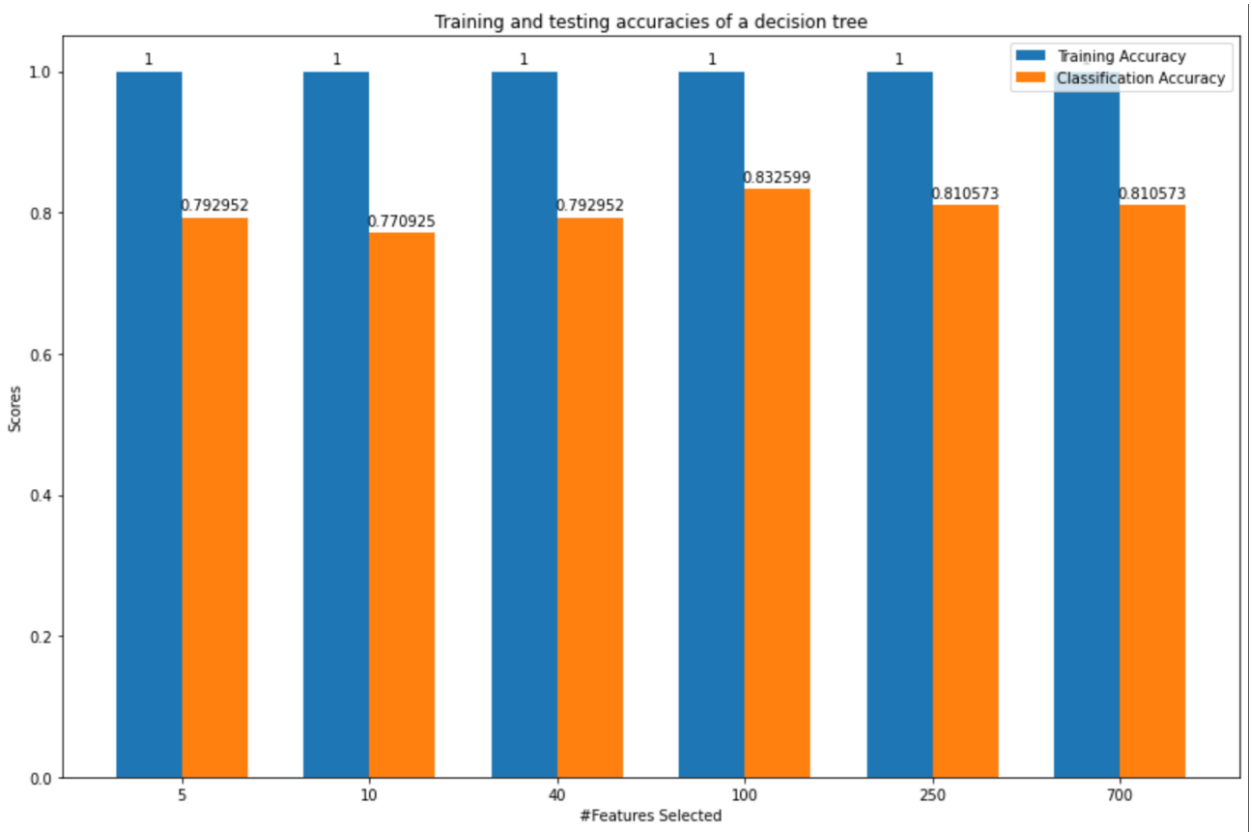
Aprendizagem 2021/22

Homework I – Group 104

II. Programming and critical analysis

1) Please see Appendix for code.

2)



The training accuracy is persistently 1 because the Decision Tree learnt without any restrictions, such as limiting its depth or defining a minimum number of observations for a feature to be allowed to be a node on the tree. Therefore, the model learnt all the decision rules inherent to the training data, explaining its accuracy.

The highest accuracy – 83% - was obtained using the 100 best features selected by the mutual information between variables criteria, that measures the dependency between them which, in this case, is the dependency between dataset features and the target - "class".

For models trained with less features – 5, 10 and 40 – the accuracy is lower. Since the number of features that are being chosen from the dataset is really low comparing with the high dimensionality of this dataset - 752 features - it is possible that there are relevant features (with a high degree of dependency with the target) that are being discarded. This leads to a loss of information, and consequently, to a decline of the accuracy.

For models trained with more features – 250 and 700 – the accuracy seems to decline a little, possibly because in the training are being included features that are not that relevant (low/medium dependency with the target) and, therefore, it is producing noise for the outcome of the learning process.

III. APPENDIX

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

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

X = df.iloc[:, :-1]
y = df['class']

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1, test_size=0.3,
stratify=y)

features_numbs = [5, 10, 40, 100, 250, 700]
training_acc, classif_acc = [], []

for num in features_numbs:

  selector = SelectKBest(mutual_info_classif, k=num)
  X_reduced = selector.fit_transform(X_train, y_train)
  cols = selector.get_support(indices=True)
  X_test_reduced = X_test.iloc[:,cols]

  clf = tree.DecisionTreeClassifier(random_state=1)
  clf = clf.fit(X_reduced, y_train)

  predictions = clf.predict(X_test_reduced)
  t_acc = clf.score(X_reduced, y_train)
  c_acc = accuracy_score(y_test, predictions)
  training_acc.append(t_acc)
  classif_acc.append(c_acc)


x = np.arange(len(features_numbs))
width = 0.35
fig, ax = plt.subplots(figsize=(12,8))
```

```
rects1 = ax.bar(x - width/2, training_acc, width, label='Training Accuracy')
rects2 = ax.bar(x + width/2, classif_acc, width, label='Classification Accuracy')

ax.set_ylabel('Accuracy score')
ax.set_xlabel('#Features Selected')
ax.set_title('Training and testing accuracies of the model')
ax.set_xticks(x, features_numbs)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()
plt.show()
```

END