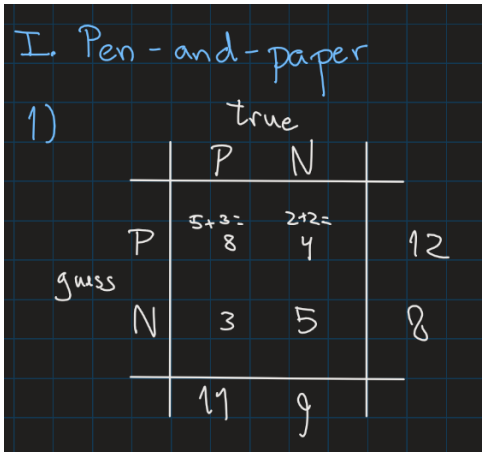
Aprendizagem 2021/22

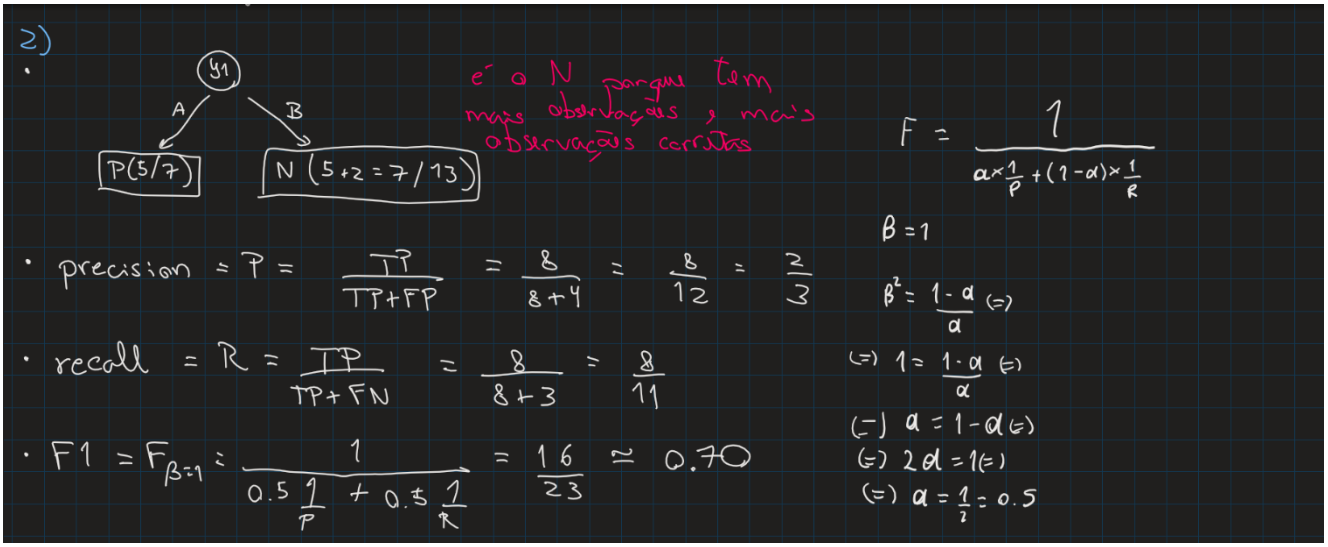
Homework I – Group 41

I. Pen-and-paper

1)



2)



3)

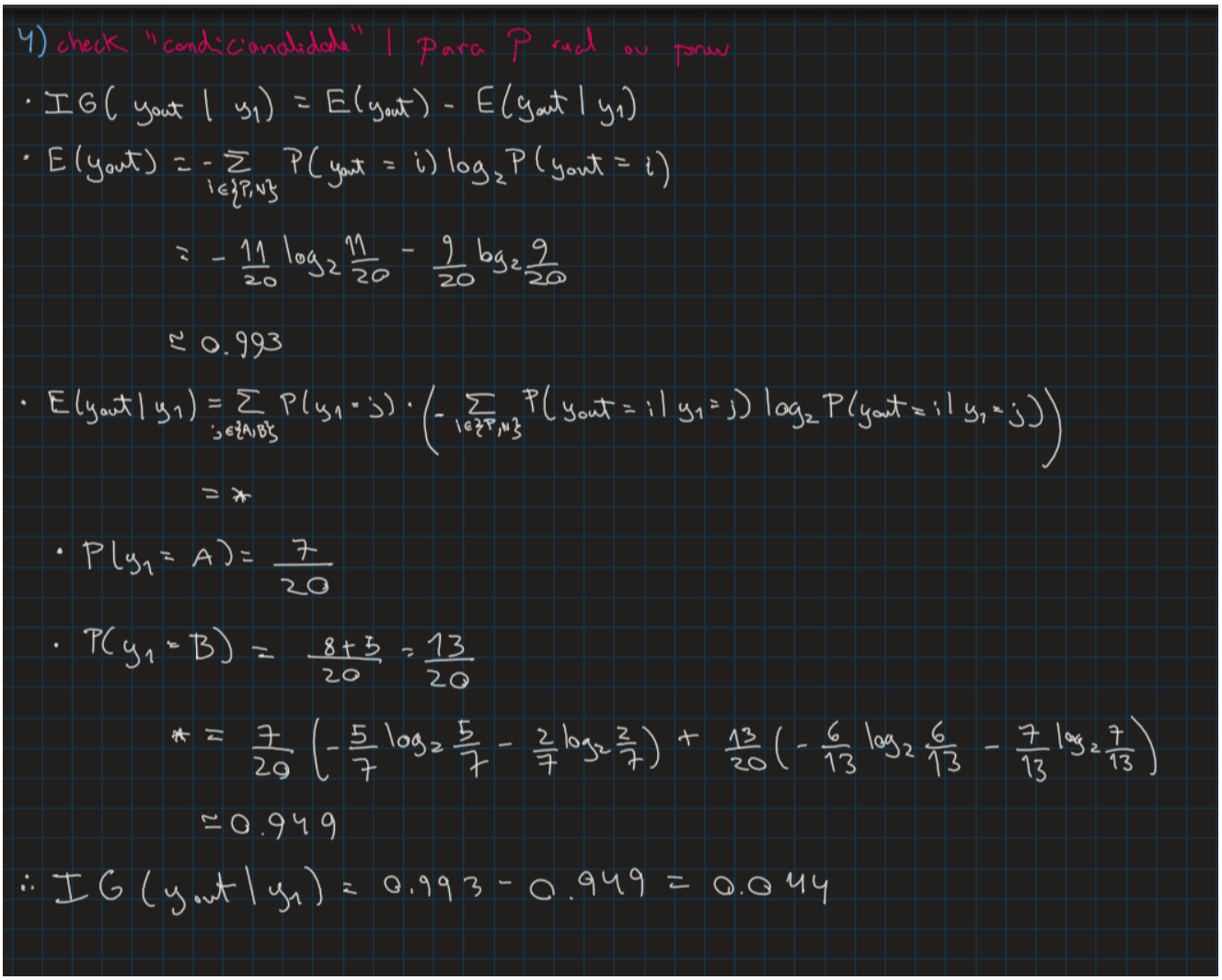
The left tree path was not further decomposed since the dataset under which the model was trained, always when the variable y1 toke the value A the output variable was always guessed as P. Also, if the expansion of the left would occur it could definitely happen overfitting of the results, resulting of the fact that the accuracy of the train dataset would be better than of the tested dataset, which was not previously used on the training of the model.

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework I – Group 41

3)



II. Programming and critical analysis

1)

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

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

#* Visualize data
#print(df.head())

X, y = df.drop("class", axis=1), np.ravel(df['class'])
```

```
k_list = [5, 10, 40, 100, 250, 700]
acc_test_list = []
acc_train_list = []

for k in k_list:
    #* select features
    transformer = GenericUnivariateSelect(mutual_info_classif, mode='k_best',
param=k)
    X_new = transformer.fit_transform(X, y)

    #* split
    X_train, X_test, y_train, y_test = train_test_split(X_new, y, train_size =
0.70, random_state=1)

    #* learn
    predictor = tree.DecisionTreeClassifier()
    predictor.fit(X_train, y_train)

    #* accurary
    y_pred_train = predictor.predict(X_train)
    y_pred_test  = predictor.predict(X_test)
    acc_train_list.append(metrics.accuracy_score(y_train, y_pred_train))
    acc_test_list.append(metrics.accuracy_score(y_test, y_pred_test))

#* final dataframe
data_acc  = []
for i in range(6):
    data_acc.append([k_list[i], acc_train_list[i], acc_test_list[i]])

df_final = pd.DataFrame(data_acc, columns = ['Features', 'Training Accuracy',
'Testing Accuracy'])

#* plot
plt.plot(k_list, acc_train_list, label = "Training Accuracy")
plt.plot(k_list, acc_test_list, label = "Testing Accuracy")
plt.legend()
plt.xlabel('x- Features')
plt.ylabel('y- Percentages')
plt.show()
```

2)

It's one because, we're predicting on the data that the model was trained, hence, will still predict everything within that subset of the dataset correctly.

Data analysis:

From the data perceived from the plot created, we can conclude that the training accuracy maintains the value 1 which means 100%, while the test accuracy has a different value in each one of the features in study that in general tends to decrease from feature to feature, averaging a percentage between 76% to 85%. This percentage indicates that there is a great difference between the data that was used to train the model and the data provided for testing, which can mean the model has been specialized to the training values, in other words there is overfitting.

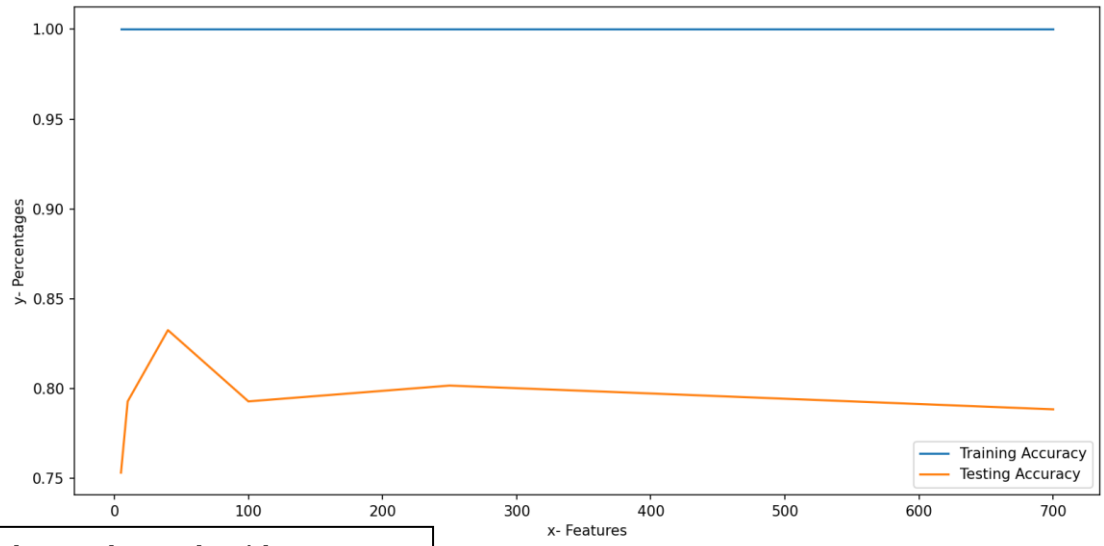


Fig.1 Plot resultante do código