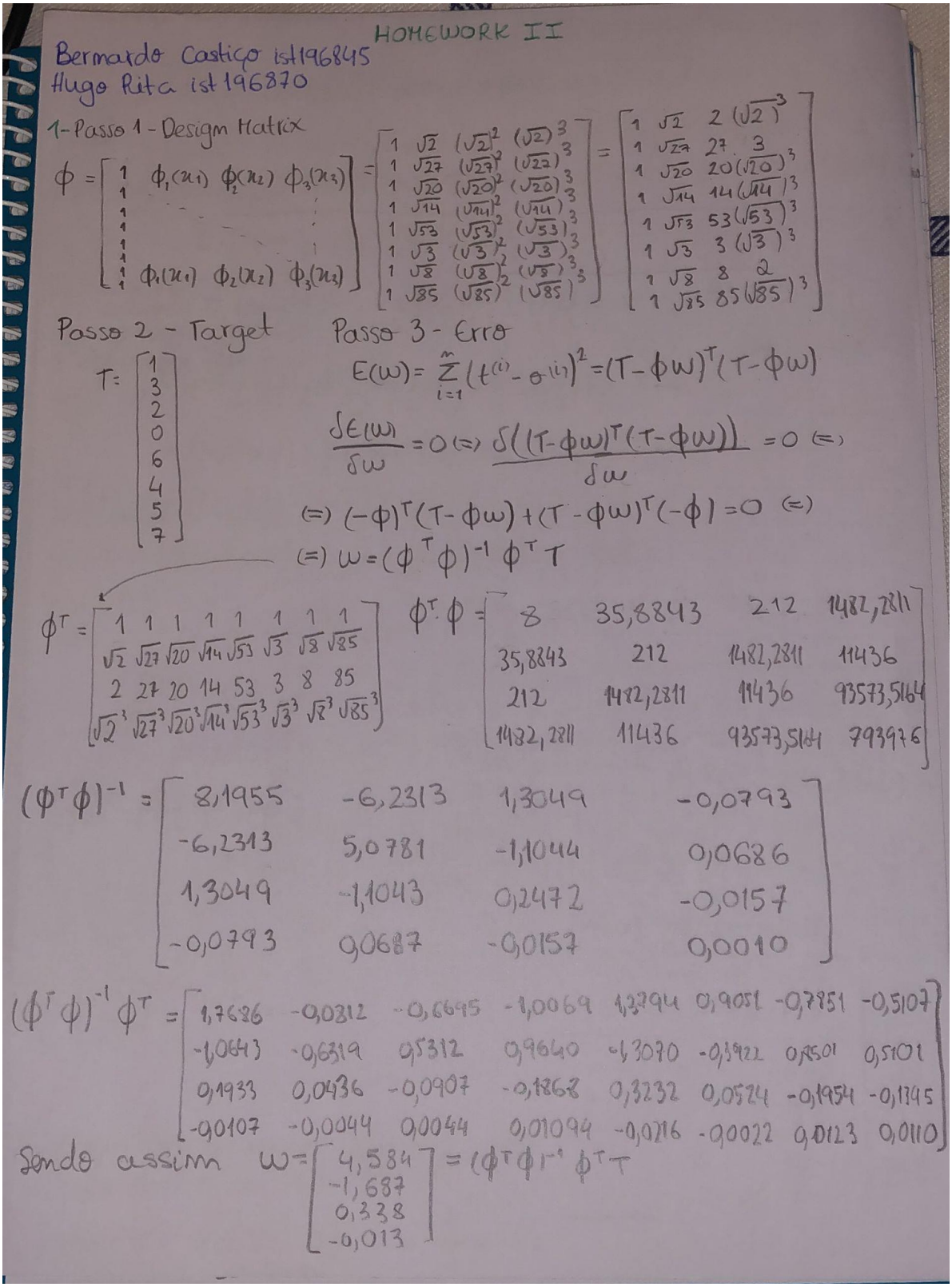
TÉCNICO LISBOA

Aprendizagem 2021/22

Homework II – Group 117

I. Pen-and-paper

Answer 1

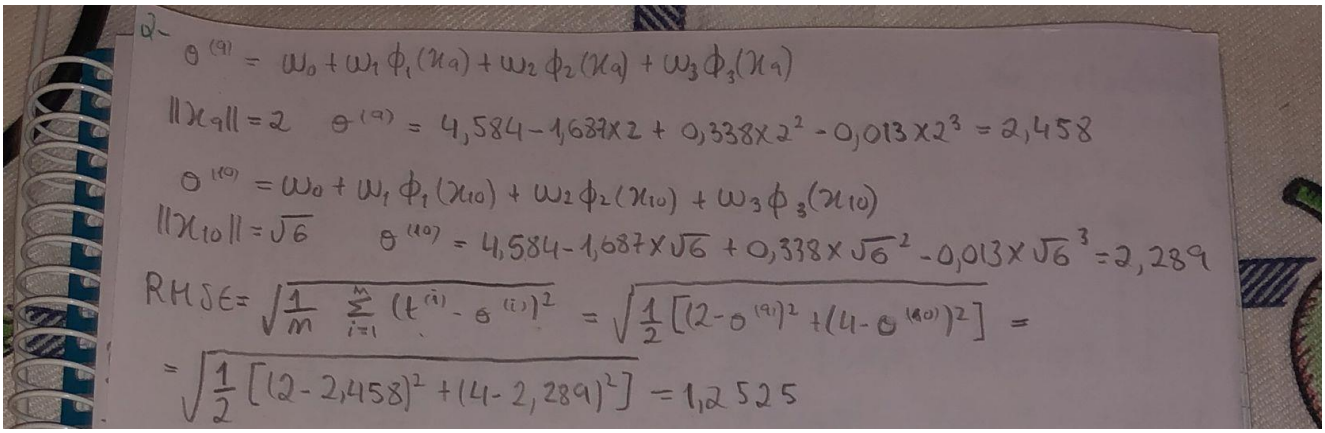


TÉCNICO LISBOA

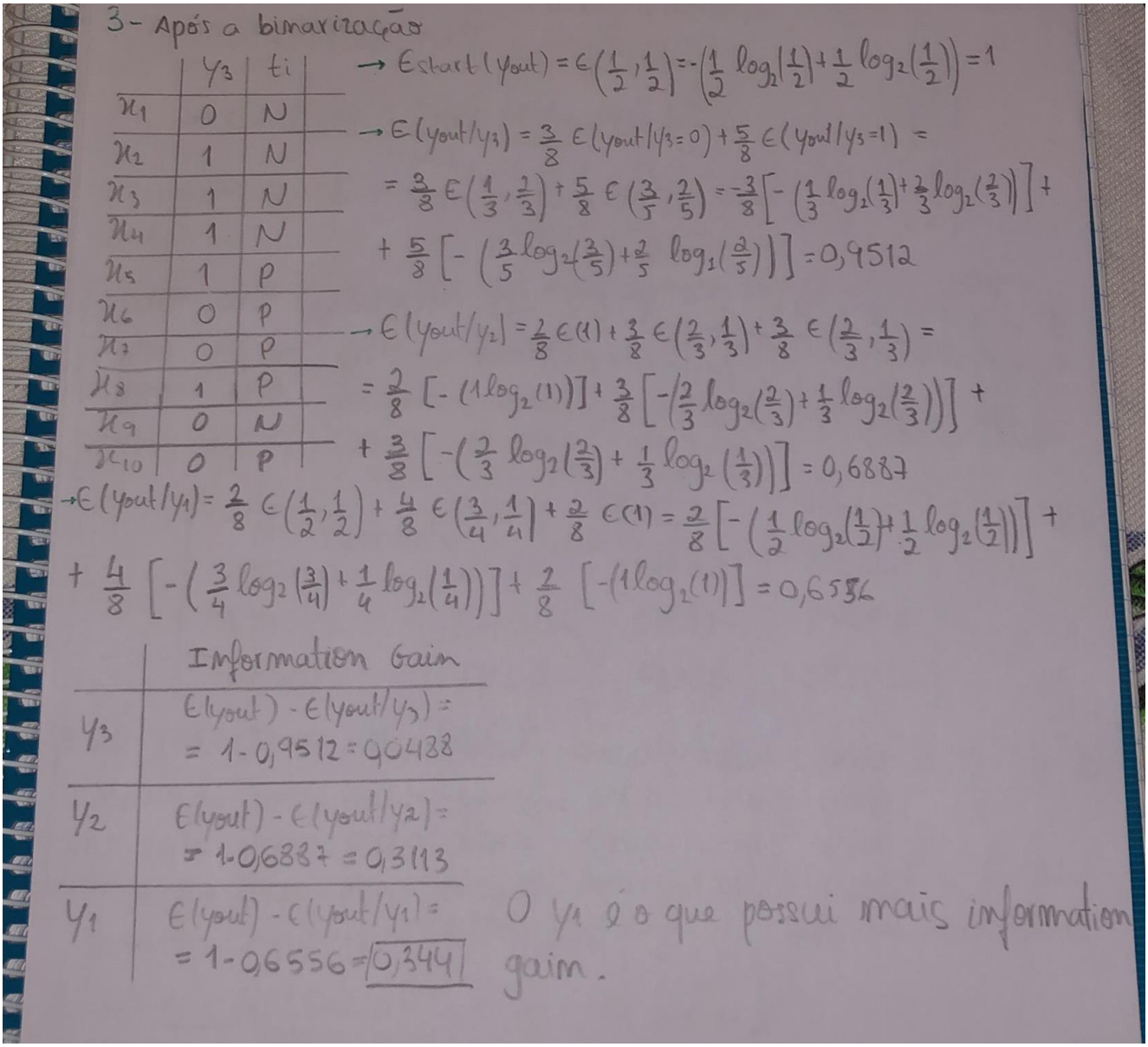
Aprendizagem 2021/22

Homework II – Group 117

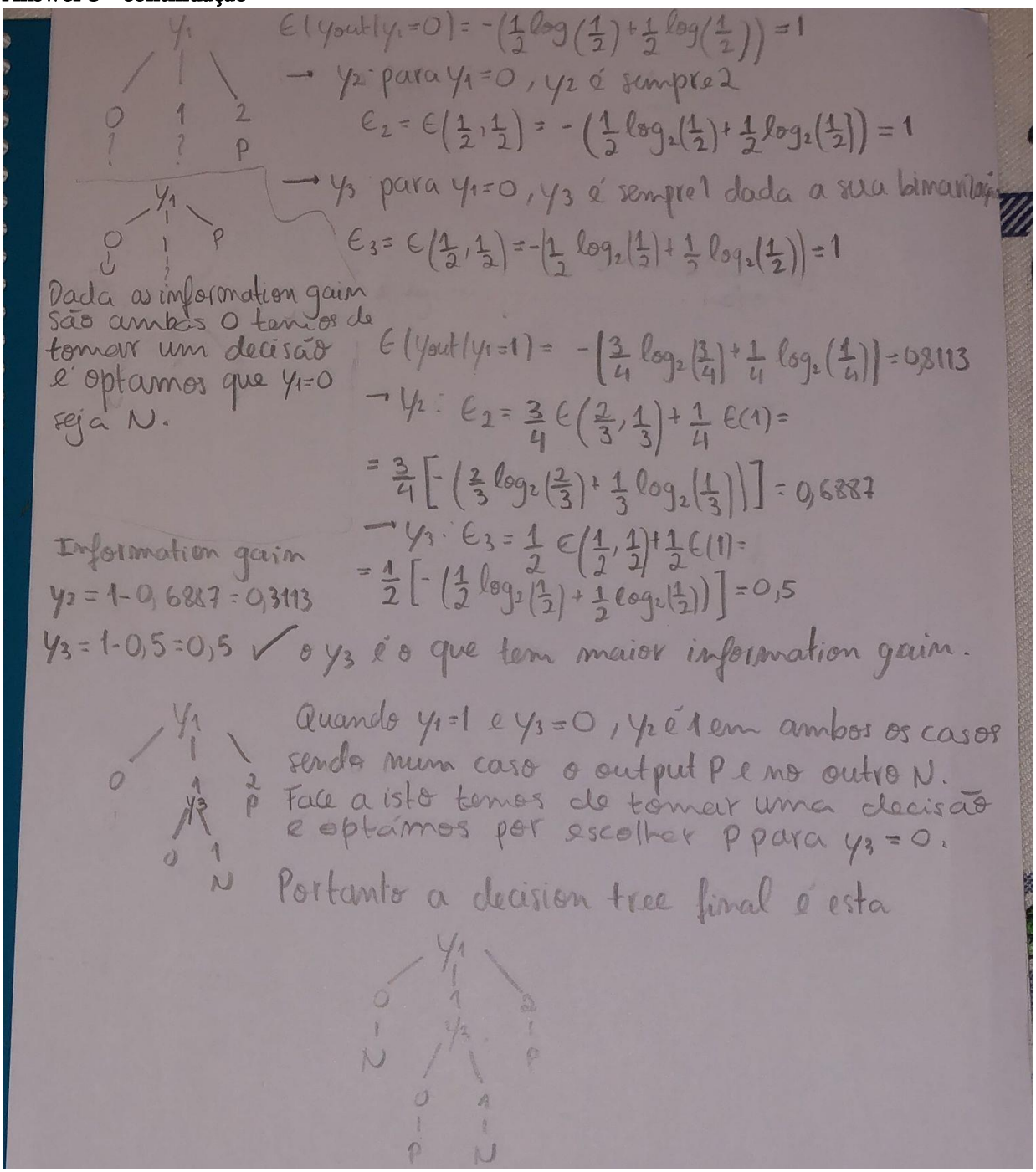
Answer 2



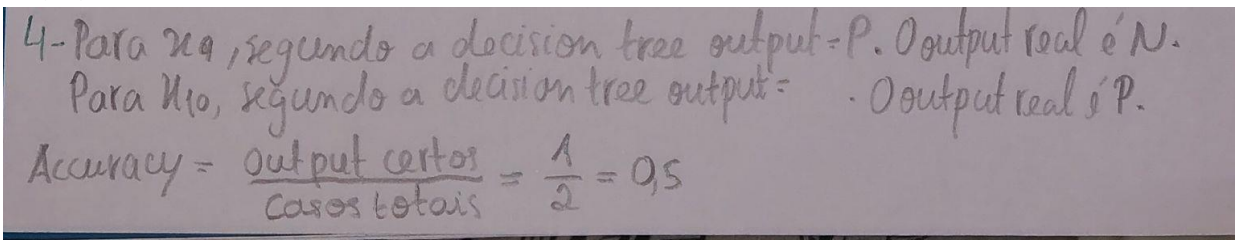
Answer 3



Answer 3 - continuação



Answer 4

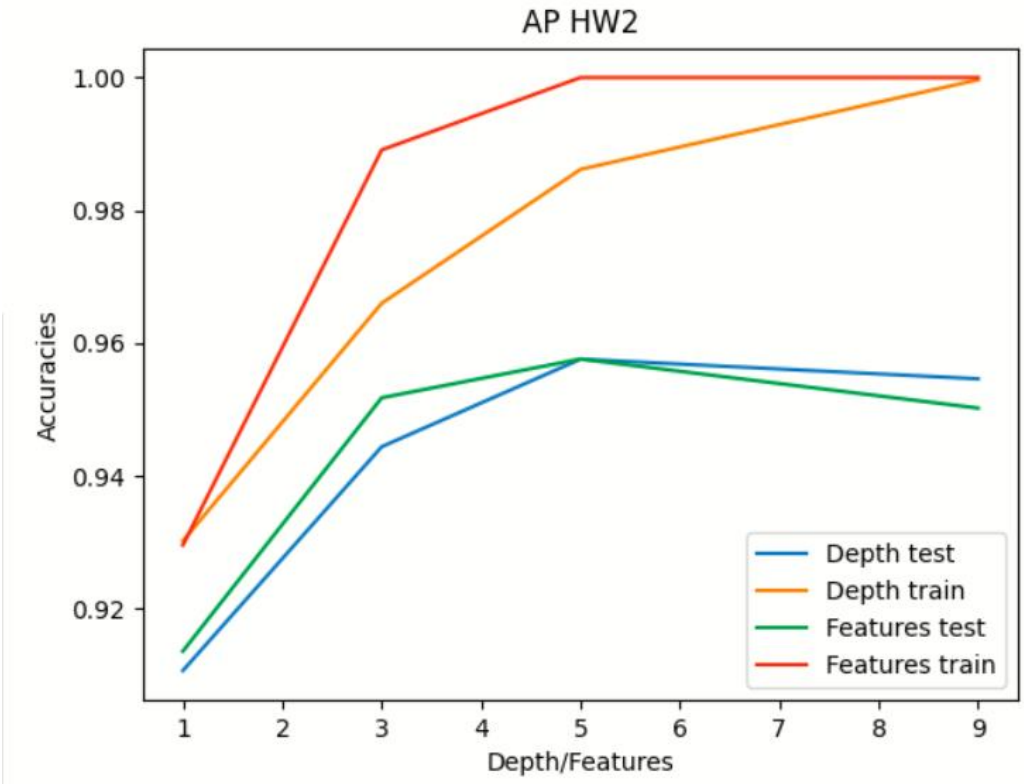


Aprendizagem 2021/22

Homework II – Group 117

II. Programming and critical analysis

Answer 5



Para a obtenção deste gráfico começámos por separar a data do ficheiro fornecido usando um 10-fold cross validation. Posto isto, selecionámos as melhores features usando a função do python SelectKBest sendo o resultado desta função usado para a alínea i. Assim, para calcular as decision trees usamos a função DecisionTreeClassifier com os parâmetros default, excepto o max depth na alínea ii.

Por fim, fizemos fit da data e predict usando as decision trees para poder comparar resultados.

Answer 6

Optámos por calcular a correlação de Pearsson com o objetivo de obter um valor para melhor esclarecer a existência de tal correlação, sendo o valor obtido = 0.98768982.

Uma das razões para a correlação observada deve-se ao facto de apesar de no caso i. se selecionar as max features e no caso ii. se selecionar a max depth. Pode-se verificar que ao selecionar max features = i $\in$ [1,3,5,9] estamos a selecionar tambem uma max depth correspondente ao valor de i que selecionamos.

A outra razão passa pelo facto de ao limitarmos tanto a max depth como a max features com valores baixos, teremos pouca informação para testar a data, o que se reflete em accuracies mais baixas. O mesmo efeito acontece para valores altos como k = 9, sendo que nestes casos o facto de existir uma data bastante vasta também leva a uma ligeira diminuição na accuracy.

Answer 7

A depth que selecionamos é k = 5, uma vez que para uma tree com max depth igual a 5 ao testarmos a nossa test data é aí que se atinge um valor máximo, ocorrendo para valores superiores a 5 overfit.

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework II – Group 117

III. APPENDIX

Nota: Falamos com o professor das teóricas do tagus (Andreas Miroslaus Wichert) e ele disse-nos que não havia qualquer problema em exceder o numero de páginas relativamente às referenciadas no enunciado.

Paste your programming code here using Consolas 9pt or 10pt.

Use **highlighting** or colored text to facilitate the analysis by your faculty hosts.

```
# Grupo 117 Aprendizagem HomeWork 2
# Bernardo Castico ist196845
# Hugo Rita ist196870
from sklearn import tree
from sklearn.model_selection import KFold
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import mutual_info_classif
import matplotlib.pyplot as plt

#Res = the 10-fold cross validation with our group number (117)
Res = KFold(n_splits=10, random_state=117, shuffle=True)

def getDataToMatrix(lines):
    realLines = []
    data = []
    toDelete = []
    for i in range(len(lines)):
        if i > 11:
            realLines += [lines[i]]
    for i in range(len(realLines)):
        for j in range(len(realLines[i])):
            if realLines[i][j] == "benign\n":
                realLines[i][j] = 1
            elif realLines[i][j] == "malignant\n":
                realLines[i][j] = 0
            elif realLines[i][j] == '?':
                toDelete += [i]
            else:
                realLines[i][j] = int(realLines[i][j])
    for i in range(len(realLines)):
        if i not in toDelete:
            data += [realLines[i]]
    return data

def splitData(list):
    a = []
    b = []
    for i in list:
        a.append(i[:-1])
        b.append(i[-1])
    return [a,b]
```

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework II – Group 117

```
def main():
    depthTestX, finalAccuraciesAllFeatures, finalAccuraciesDepth, res , AllFeaturesTrainY,
AllFeaturesTrainX = [],[],[],[],[],[]
    depthTestY , AllFeaturesTestY, AllFeaturesTestX, depthTrainY, depthTrainX = [],[],[],[],[]

    with open("HW2.txt") as f:
        lines = f.readlines()
    for line in lines:
        tmp = line.split(',')
        res.append(tmp)
    data = getDataToMatrix(res)

    for i in [1,3,5,9]:
        counter11, counter12, counter21, counter22 = 0,0,0,0
        accuraciesDepth, accuraciesAllFeatures = [],[]

        for train, test in Res.split(data):
            testData, trainData = [],[]
            accuracyAuxDepthTest, accuracyAuxDepthTrain = 0,0
            accuracyAuxAllFeaturesTest, accuracyAuxAllFeaturesTrain = 0,0

            for j in test:
                testData += [data[j]]
            for j in train:
                trainData += [data[j]]
            trainDataSplit = splitData(trainData)
            testDataSplit = splitData(testData)

            decision = SelectKBest(mutual_info_classif, k=i).fit(trainDataSplit[0],
trainDataSplit[1])
            decisionTrainData = decision.transform(trainDataSplit[0])
            decisionTestData = decision.transform(testDataSplit[0])

            resultDepth = tree.DecisionTreeClassifier(max_depth=i, criterion="gini",
max_features=None)
            resultAllFeatures = tree.DecisionTreeClassifier(max_depth=None, criterion="gini",
max_features=None)

            resultDepth.fit(trainDataSplit[0], trainDataSplit[1])
            resultAllFeatures.fit(decisionTrainData, trainDataSplit[1])

            predictionsTest = resultDepth.predict(testDataSplit[0])
            predictionsTrain = resultDepth.predict(trainDataSplit[0])

            predictionsTestFeatures = resultAllFeatures.predict(decisionTestData)
            predictionsTrainFeatures = resultAllFeatures.predict(decisionTrainData)
```

```
            for j in range(len(predictionsTestFeatures)):
                if predictionsTestFeatures[j] == testDataSplit[1][j]:
                    accuracyAuxAllFeaturesTest += 1
                if predictionsTest[j] == testDataSplit[1][j]:
                    accuracyAuxDepthTest += 1
            for j in range(len(predictionsTrainFeatures)):
                if predictionsTrainFeatures[j] == trainDataSplit[1][j]:
                    accuracyAuxAllFeaturesTrain += 1
                if predictionsTrain[j] == trainDataSplit[1][j]:
                    accuracyAuxDepthTrain += 1
            accuraciesAllFeatures += [[accuracyAuxAllFeaturesTest/len(predictionsTestFeatures),
accuracyAuxAllFeaturesTrain/len(predictionsTrainFeatures)]]
            accuraciesDepth += [[accuracyAuxDepthTest / len(predictionsTest), accuracyAuxDepthTrain
/ len(predictionsTrain)]]

        for k in range(len(accuraciesDepth)):
            counter11 += accuraciesDepth[k][0]
            counter12 += accuraciesDepth[k][1]
            counter21 += accuraciesAllFeatures[k][0]
            counter22 += accuraciesAllFeatures[k][1]

        finalAccuraciesDepth += [[counter11 / 10, counter12 / 10]]
        finalAccuraciesAllFeatures += [[counter21 / 10, counter22 / 10]]
    #Plot
    for i in range(4):
        depthTestX = [1,3,5,9]
        depthTestY += [finalAccuraciesDepth[i][0]]
        depthTrainX = [1, 3, 5, 9]
        depthTrainY += [finalAccuraciesDepth[i][1]]
        AllFeaturesTestX = [1,3,5,9]
        AllFeaturesTestY += [finalAccuraciesAllFeatures[i][0]]
        AllFeaturesTrainX = [1, 3, 5, 9]
        AllFeaturesTrainY += [finalAccuraciesAllFeatures[i][1]]

    plt.xlabel('Depth/AllFeatures')
    plt.ylabel('Accuracies')
    plt.title('AP HW2')
    plt.plot(depthTestX, depthTestY, label = "depth test")
    plt.plot(depthTrainX, depthTrainY, label = "depth train")
    plt.plot(AllFeaturesTestX, AllFeaturesTestY, label = "All Features test")
    plt.plot(AllFeaturesTrainX, AllFeaturesTrainY, label = "All Features train")
    plt.legend()
    plt.show()
main()
```

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework II – Group 117

END