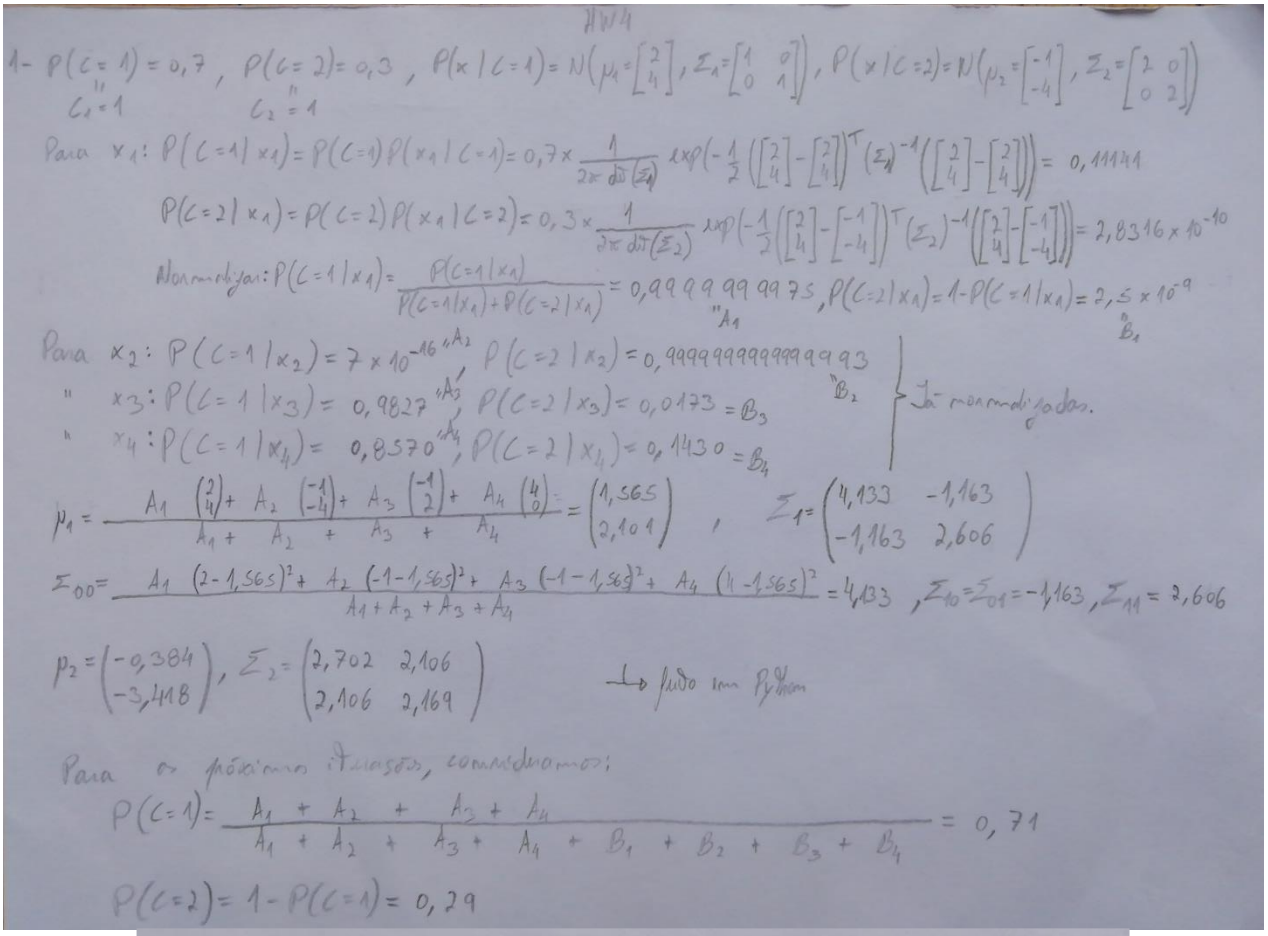
TÉCNICO LISBOA

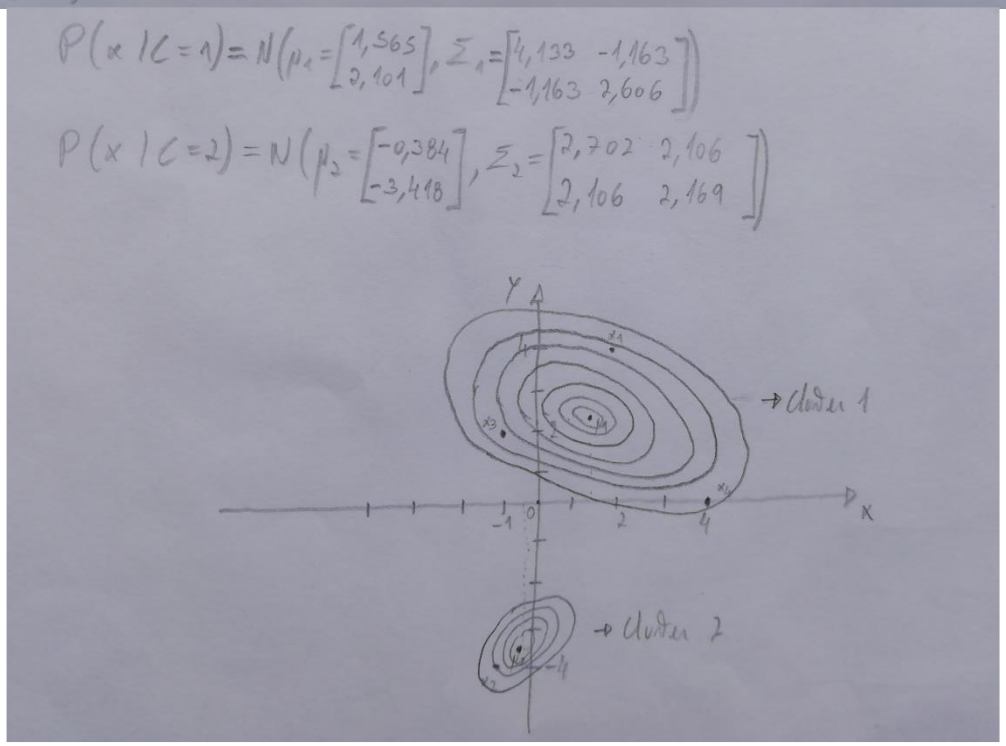
Aprendizagem 2021/22

Homework IV – Group 96

I. Pen-and-paper

1)



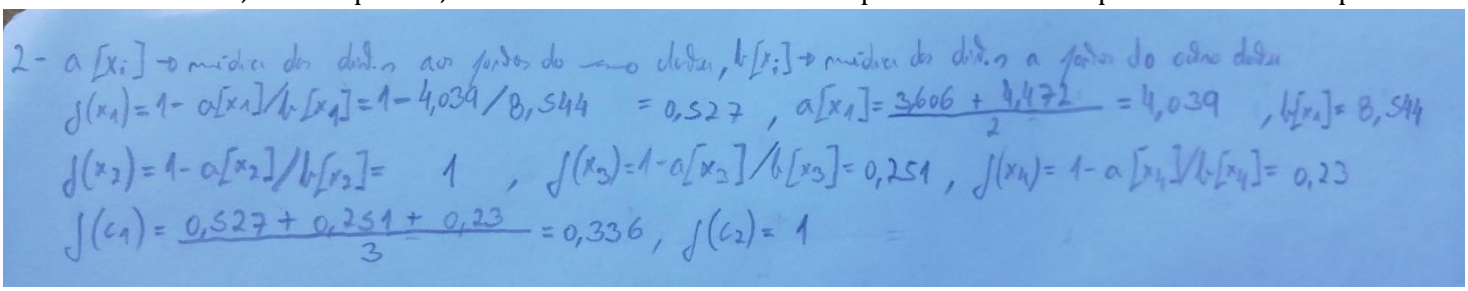


TÉCNICO LISBOA

Aprendizagem 2021/22

Homework IV – Group 96

2) A silhueta associada ao *cluster*/centroide 2 é maior, logo, é melhor, já que possui apenas um único ponto. O *cluster* 1, tendo 3 pontos, tem uma silhueta menor e acaba por ser menos compacto e ter mais dispersão.



3)

a. $MLP = n \times n \times 3 + n \times 1 \times 3 + 2 \times n + 2 \times 1$

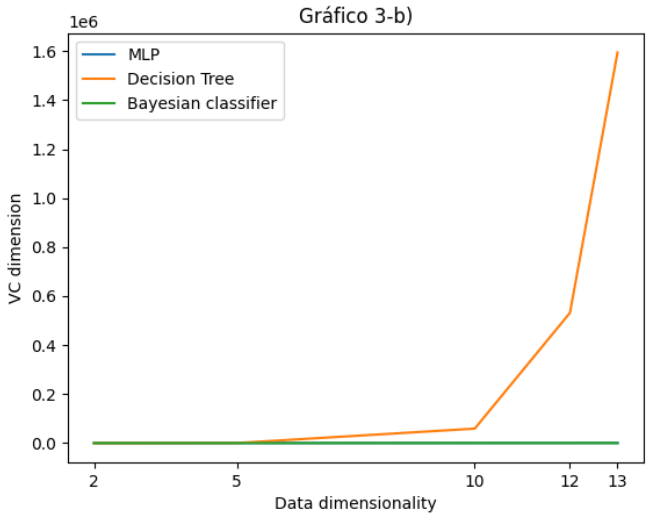
$Decision\ Tree = 3^n$

$Bayesian\ Classifier = 1 + 2\ \times (n + \frac{n \times n - n}{2})$

Assim, para n = 5, temos d_{vc}= 102 para o *MLP*, d_{vc}= 243 para a *Decision Tree* e d_{vc}= 31 para o *Bayesian Classifier*.

b. Usando as equações acima, obtemos:

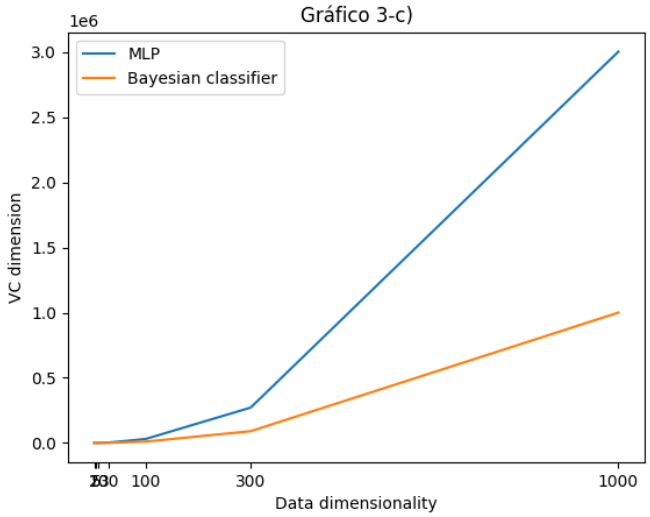
	2	**5**	**10**	**12**	**13**
i.	24	102	352	494	574
ii.	9	243	59049	531441	1594323
iii.	7	31	111	157	183



Concluímos que a *VC dimension* da *Decision Tree* aumenta de modo exponencial, ou seja, de modo muito superior aos outros dois classificadores (polinomiais).

c. Além dos dados da alínea anterior para m = 2, 5 e 10 para i. e iii., usando as equações acima, obtemos:

	30	**100**	**300**	**1000**
i.	2852	30502	271502	3005002
iii.	931	10101	90301	1001001

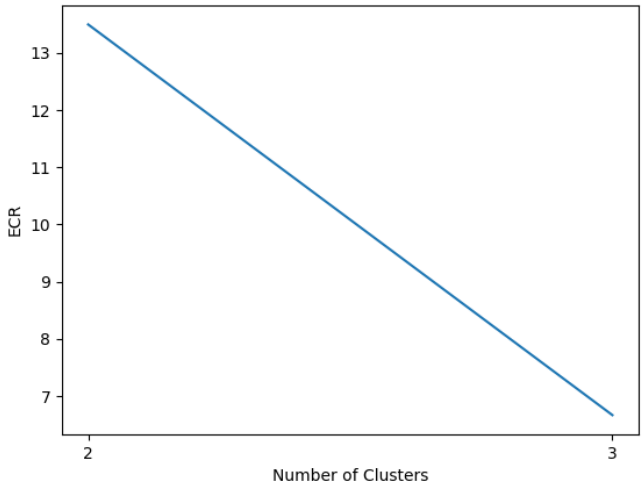


Concluímos que a *VC dimension* do *MLP* e do *Bayesian Classifier* aumenta de forma polinomial e é mais acentuada no *MLP* devido ao termo $3n^2$ em comparação com o termo n^2 do *Bayesian classifier*.

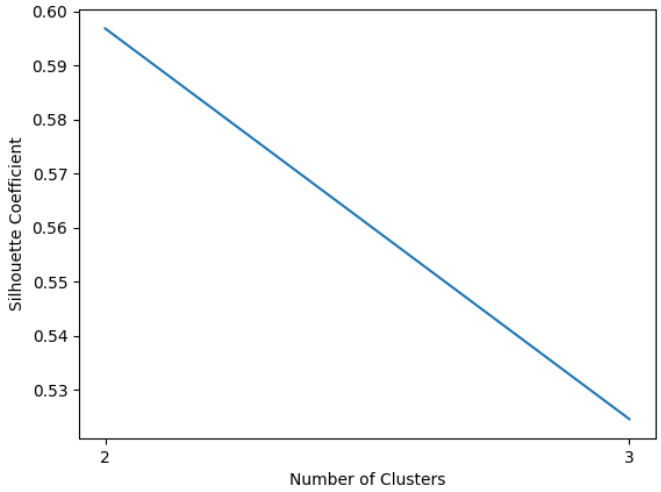
II. Programming and critical analysis

4)

a. Com 3 *clusters*, o *ECR* é menor, o que significa que há menos classificações erradas.



b. Com 3 *clusters*, a *silhouette* é menor, o que significa que os *clusters* estão mais compactos, minimizando a distância entre pontos do mesmo *cluster* e maximizando a distância entre pontos de *clusters* diferentes.



5)

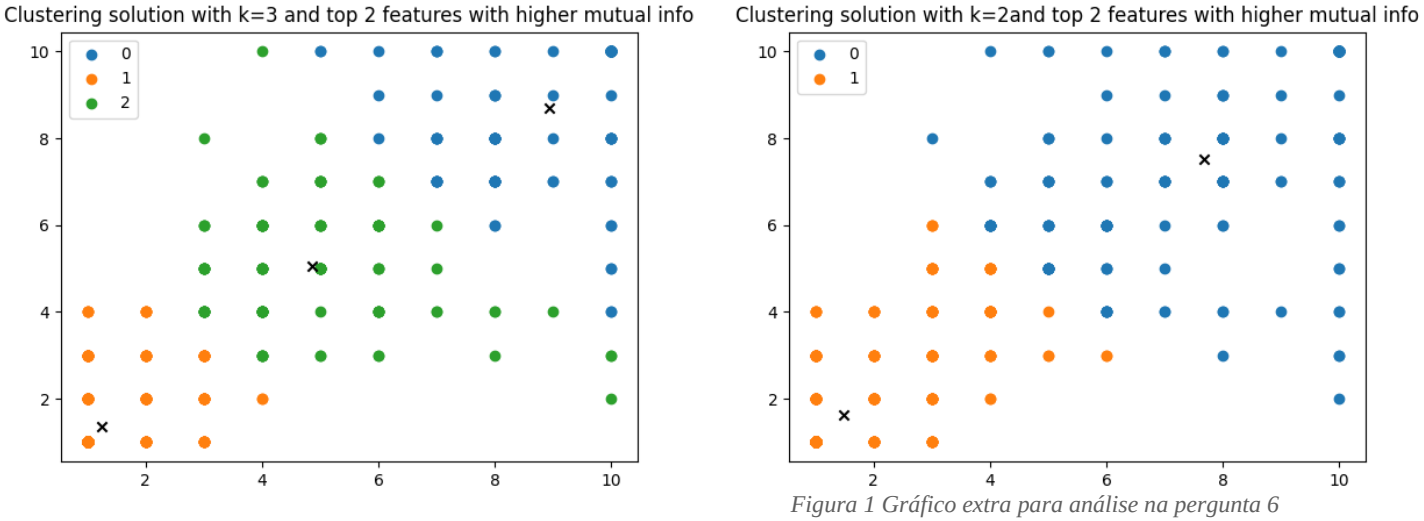


Figura 1 Gráfico extra para análise na pergunta 6

6) Com base no gráfico da pergunta anterior, concluímos que 3 *clusters* é uma boa solução, já que diferencia bem as classes reais, *malign* e *benign*. O cluster do "meio", a verde, corresponde à classe *malign* e, apesar de conter algumas amostras da classe *benign*, o erro não é assim muito grande (11,6). Se compararmos com o uso de apenas 2 *clusters* na mesma situação (gráfico extra na pergunta 5), concluímos que há muito mais classificações erradas, com um *ECR* de 32,0, levando a uma solução pior do que com 3 *clusters*.

III. APPENDIX

```
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt
```

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework IV – Group 96

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import mutual_info_classif

df = pd.read_csv("test.csv", dtype={"Clump_Thickness": int,
                                    "Cell_Size_Uniformity": int,
                                    "Cell_Shape_Uniformity": int,
                                    "Marginal_Adhesion": int,
                                    "Single_Epi_Cell_Size": int,
                                    "Bare_Nuclei": float,
                                    "Bland_Chromatin": int,
                                    "Normal_Nucleoli": int,
                                    "Mitoses": int,
                                    "Class": str}, na_values=["?"])

df = df.dropna()

columns = ["Clump_Thickness", "Cell_Size_Uniformity", "Cell_Shape_Uniformity",
"Marginal_Adhesion", "Single_Epi_Cell_Size", "Bare_Nuclei", "Bland_Chromatin",
"Normal_Nucleoli", "Mitoses", "Class"]
features = columns[:-1]
classes = ["benign", "malign"]

Y = df["Class"]
X = df.drop(columns=["Class"])

Y = [0 if x == "benign" else 1 for x in Y]
Y = np.ravel(pd.DataFrame(Y))

# Pergunta 4


def ecr(true_labels, clf_labels, n_clusters):
    cluster0, cluster1, cluster2 = [], [], []  # indices dos pontos pertences aos
cluster K

    for i in range(0, len(true_labels)):
        if clf_labels[i] == 0:
            cluster0 += [i]
        elif clf_labels[i] == 1:
            cluster1 += [i]
        else:
            cluster2 += [i]

    # cK indica a classe real maioritaria no cluster K
    l0 = list(true_labels[cluster0])
    c0 = max(l0, key=l0.count) if l0 != [] else 0
    l1 = list(true_labels[cluster1])
    c1 = max(l1, key=l1.count) if l1 != [] else 0
    l2 = list(true_labels[cluster2])
    c2 = max(l2, key=l2.count) if l2 != [] else 0

    print(c0, c1, c2)

    errors = 0
    l = [[cluster0, c0], [cluster1, c1], [cluster2, c2]]
    for c in l:
        for i in c[0]:
            if true_labels[i] != c[1]:  # comparar classe real com a classe desse
cluster e quantas classificacoes erradas
```

```
                errors += 1

    return (1/n_clusters) * errors


silhouette_coefficients = []
ecrs = []
n_clusters = (2, 3)
for k in n_clusters:
    clf = KMeans(n_clusters=k, random_state=0)
    clf.fit(X)
    ecrs.append(ecr(Y, clf.labels_, k))
    score = silhouette_score(X, clf.labels_)
    silhouette_coefficients.append(score)

plt.plot(n_clusters, ecrs)
plt.xticks(n_clusters)
plt.xlabel("Number of Clusters")
plt.ylabel("ECR")
plt.show()

plt.plot(n_clusters, silhouette_coefficients)
plt.xticks(n_clusters)
plt.xlabel("Number of Clusters")
plt.ylabel("Silhouette Coefficient")
plt.show()

# Pergunta 5


ecrs2 = []
silhouette_coefficients2 = []

select = SelectKBest(score_func=mutual_info_classif, k=2)
X_new = select.fit_transform(X, Y)

for k in n_clusters:
    clf = KMeans(n_clusters=k, random_state=0)
    clf.fit(X_new)

    label = clf.labels_
    u_labels = np.unique(label)
    centroids = clf.cluster_centers_

    for i in u_labels:
        plt.scatter(X_new[label == i, 0], X_new[label == i, 1], label=i)
    plt.scatter(centroids[:, 0], centroids[:, 1], marker="x", color='k')
    plt.legend()
    plt.title("Clustering solution with k=" + str(k) + "and top 2 features with
higher mutual info")
    plt.show()

    print(ecr(Y, clf.labels_, k))
```

END