TÉCNICO LISBOA

Aprendizagem 2021/22

Homework IV – Group 6

I. Pen-and-paper

1) Initialisation:

$$\Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \mu_1 = x_1 = \begin{bmatrix} 2 \\ 4 \end{bmatrix} \quad \Sigma_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \quad \mu_2 = x_2 = \begin{bmatrix} -1 \\ -4 \end{bmatrix}$$

Cluster (C)	P(C) (Prior)	$P(y_1, y_2 \mid C)$
C_1	0.7	$N(y_1, y_2 \mid \mu_1, \Sigma_1)$
C_2	0.3	$N(y_1, y_2 \mid \mu_2, \Sigma_2)$

$$N(y_1, y_2 \mid \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}}{2\pi\sqrt{|\Sigma|}}$$

Expectation (E-step):

$$P(C \mid x = (y_1, y_2)) = \frac{P(x = (y_1, y_2) \mid C) \times P(C)}{P(x = (y_1, y_2))} = \frac{P(x = (y_1, y_2) \mid C) \times P(C)}{\sum_{i=1}^{2} P(x = (y_1, y_2) \mid C_i) \times P(C_i)}$$

x_i	$P(x = (y_1, y_2) \mid C)$ (Likelihood)		$P(x = (y_1, y_2) \mid C) \times P(C)$ (Joint)		$P(C \mid x = (y_1, y_2))$ (Posterior)	
	C_1	C_2	C_1	C_2	C_1	C_2
x_1	1.5915×10^{-1}	9.4387×10^{-10}	1.1141×10^{-1}	2.8316×10^{-10}	**1.0000**	0.0000
x_2	2.2391×10^{-17}	7.9577×10^{-2}	1.5673×10^{-17}	2.3873×10^{-2}	0.0000	**1.0000**
x_3	2.3927×10^{-4}	9.8206×10^{-6}	1.6750×10^{-4}	2.9462×10^{-6}	**0.9827**	0.0173
x_4	7.2256×10^{-6}	2.8137×10^{-6}	2.8316×10^{-6}	8.4410×10^{-7}	**0.8570**	0.1430

Maximization (M-step):

New priors:

$$w_i = \sum_{j=1}^{4} P(C_i \mid x_j) \quad P(C_i)^{new} = \frac{w_i}{\sum_{j=1}^{2} w_j}$$

C_i	w_i	$P(C_i)^{new}$ (New prior)
C_1	2.8397	0.7099
C_2	1.1603	0.2901

TÉCNICO LISBOA

Aprendizagem 2021/22

Homework IV – Group 6

New values of μ_1 and μ_2:

$$\boldsymbol{\mu_i^{new}} = \frac{\sum_{j=1}^{4} \boldsymbol{P(C_i|x_j) \times x_j}}{\boldsymbol{w_i}}$$

$$\boldsymbol{\mu_1^{new}} = \frac{1.0000\begin{bmatrix}2\\4\end{bmatrix} + 0.0000\begin{bmatrix}-1\\-4\end{bmatrix} + 0.9827\begin{bmatrix}-1\\2\end{bmatrix} + 0.8570\begin{bmatrix}4\\0\end{bmatrix}}{2.8397} \approx \begin{bmatrix}\mathbf{1.5654}\\\mathbf{2.1007}\end{bmatrix}$$

$$\boldsymbol{\mu_2^{new}} = \frac{0.0000\begin{bmatrix}2\\4\end{bmatrix} + 1.0000\begin{bmatrix}-1\\-4\end{bmatrix} + 0.0173\begin{bmatrix}-1\\2\end{bmatrix} + 0.1430\begin{bmatrix}4\\0\end{bmatrix}}{1.1603} \approx \begin{bmatrix}\mathbf{-0.3837}\\\mathbf{-3.4176}\end{bmatrix}$$

New values of Σ_1 and Σ_2:

$$\boldsymbol{\Sigma_i^{new}} = \begin{bmatrix}\boldsymbol{\sigma_{i11}} & \boldsymbol{\sigma_{i12}}\\\boldsymbol{\sigma_{i21}} & \boldsymbol{\sigma_{i22}}\end{bmatrix}$$

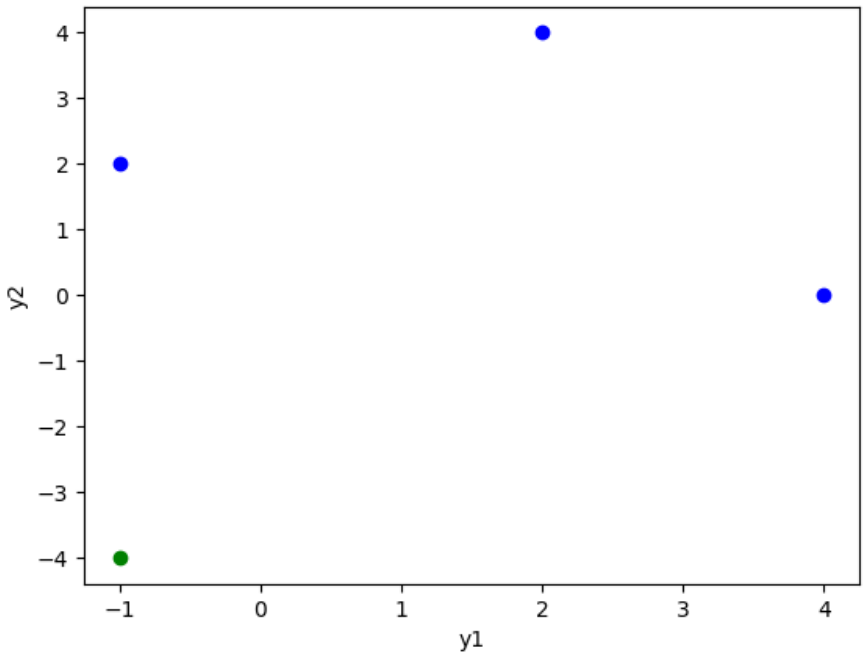
$$\sigma_{i11} = \frac{\sum_{j=1}^{4} P(C_i|x_j) \times (x_{j1} - \mu_{i1}^{new})^2}{w_i}$$

$$\sigma_{i12} = \sigma_{i21} = \frac{\sum_{j=1}^{4} P(C_i|x_j) \times (x_{j1} - \mu_{i1}^{new})(x_{j2} - \mu_{i2}^{new})}{w_i}$$

$$\sigma_{i22} = \frac{\sum_{j=1}^{4} P(C_i|x_j) \times (x_{j2} - \mu_{i2}^{new})^2}{w_i}$$

$$\boldsymbol{\Sigma_1^{new}} \approx \begin{bmatrix}\mathbf{4.1328} & \mathbf{-1.1634}\\\mathbf{-1.1634} & \mathbf{2.6056}\end{bmatrix} \quad \boldsymbol{\Sigma_2^{new}} \approx \begin{bmatrix}\mathbf{2.7017} & \mathbf{2.1062}\\\mathbf{2.1062} & \mathbf{2.1692}\end{bmatrix}$$

Given the obtained posteriors, we can divide the dataset into 2 clusters: $\boldsymbol{c_1} = \{\boldsymbol{x_1}, \boldsymbol{x_3}, \boldsymbol{x_4}\}$ and $\boldsymbol{c_2} = \{\boldsymbol{x_2}\}$. A visual representation of them can be found below (observations belonging to $\boldsymbol{c_1}$ are in **blue**, and the one belonging to $\boldsymbol{c_2}$ is in **green**):



TÉCNICO LISBOA

Aprendizagem 2021/22

Homework IV – Group 6

2) The **silhouette coefficient** for a point is given by:

$$\boldsymbol{S(x) = 1 - \frac{a(x)}{b(x)}}$$

Such that:

$$\boldsymbol{a(x) = Average\ distance\ of\ x\ to\ the\ points\ in\ its\ cluster}$$

$$\boldsymbol{b(x) = Average\ distance\ of\ x\ to\ the\ points\ in\ the\ other\ clusters}$$

$$\boldsymbol{S(x_1)} = 1 - \frac{a(x_1)}{b(x_1)} = 1 - \frac{\frac{3.6055 + 4.4732}{2}}{8.5440} = \boldsymbol{0.5273}$$

$$\boldsymbol{S(x_2)} = 1 - \frac{a(x_2)}{b(x_2)} = 1 - \frac{0}{b(x_2)} = \boldsymbol{1}$$

$$\boldsymbol{S(x_3)} = 1 - \frac{a(x_3)}{b(x_3)} = 1 - \frac{\frac{4.4721 + 5.3852}{2}}{6} = \boldsymbol{0.2508}$$

$$\boldsymbol{S(x_4)} = 1 - \frac{a(x_4)}{b(x_4)} = 1 - \frac{\frac{4.721 + 5.3852}{2}}{6.4031} = \boldsymbol{0.2303}$$

We can then calculate the silhouette coefficient for a given cluster by calculating the mean silhouette of its points:

$$\boldsymbol{S(c_1)} = \frac{\boldsymbol{S(x_1) + S(x_3) + S(x_4)}}{\boldsymbol{3}} = \frac{0.5273 + 0.2508 + 0.2303}{3} = \boldsymbol{0.3361}$$

$$\boldsymbol{S(c_2) = S(x_2) = 1}$$

Since the silhouette coefficient measures both cohesion and separations, we conclude that the cluster c_2 is better than the c_1 cluster.

Finally, to compute the silhouette coefficient of the clustering solution we compute the mean of its clusters:

$$\boldsymbol{Silhouette} = \frac{\boldsymbol{S(c_1) + S(c_2)}}{\boldsymbol{2}} = \frac{0.3361 + 1}{2} = \boldsymbol{0.6681}$$

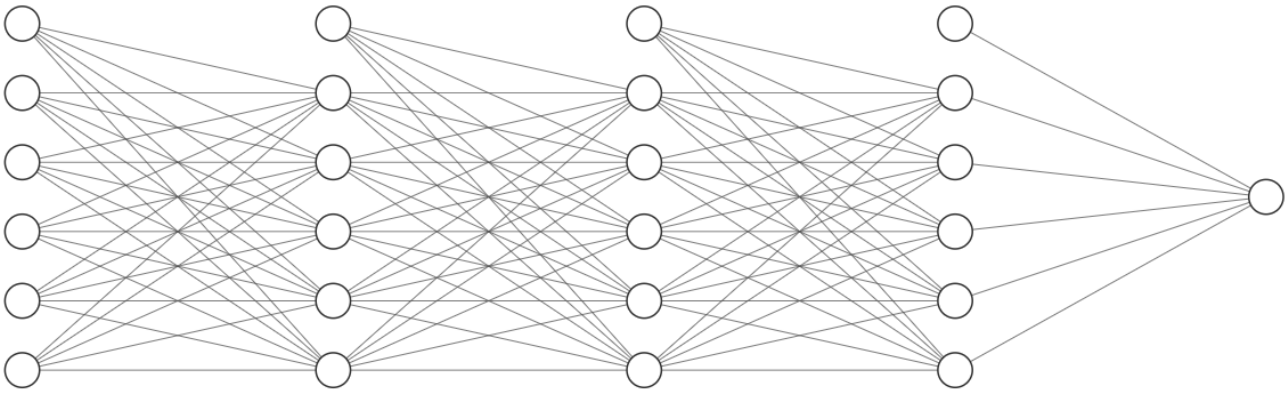
TÉCNICO LISBOA

Aprendizagem 2021/22

Homework IV – Group 6

3) (a) The VC dimension can be estimated by the number of parameters need to compute the classifier

(i) The MLP described in the question can be drawn as such:



Therefore, we need the following parameters to perform the necessary computations:

$$\boldsymbol{W^{[1]}(5\times 5), b^{[1]}(5\times 1), W^{[2]}(5\times 5), b^{[2]}(5\times 1), W^{[3]}(5\times 5), b^{[3]}(5\times 1), W^{[4]}(1\times 5), b^{[4]}(1\times 1)}$$

For an MLP whose hidden layers have as many nodes as the amount of input variables (m):

$$\boldsymbol{d_{VC}(MLP)} \approx \sum_{i} \left|\boldsymbol{W^{[i]}}\right| + \left|\boldsymbol{b^{[i]}}\right| = (m^2+m)\times 3 + (m+1) = \boldsymbol{3m^2+4m+1}$$

As such, since, in this case, $m = 5$: $\boldsymbol{d_{VC}(MLP)} \approx 5^2 \times 3 + 4 \times 5 + 1 = \boldsymbol{96}$.

(ii) For a decision tree using m input variables, where all variables are discretized using 3 bins:

$$\boldsymbol{d_{VC}(BDT) = 3^m}$$

For $m = 5$: $\boldsymbol{d_{VC}(BDT)} = 3^5 = \boldsymbol{243}$.

(iii) For a two-class Bayesian classifier, we need to calculate, for both C = 0 and C = 1:

$$\boldsymbol{P(C|x)} = \frac{\boldsymbol{P(x|C)\times P(C)}}{\boldsymbol{P(x)}}$$

Since the denominator can be calculated as the sum of both calculated numerators, and P(C = 1) = 1 – P(C = 0), the amount of needed parameters is obtained from P(C = 0), P(x|C = 0) and P(x|C = 1). As P(x|C) = $N(x|\mu,\Sigma)$, we must find the number of different parameters for μ and Σ.

For each value of C, μ has $m \times 1$ dimensions, so we need m different values. On the other hand, Σ ($m \times m$), being symmetrical, has $\frac{m(m+1)}{2}$ distinct parameters.

In conclusion:

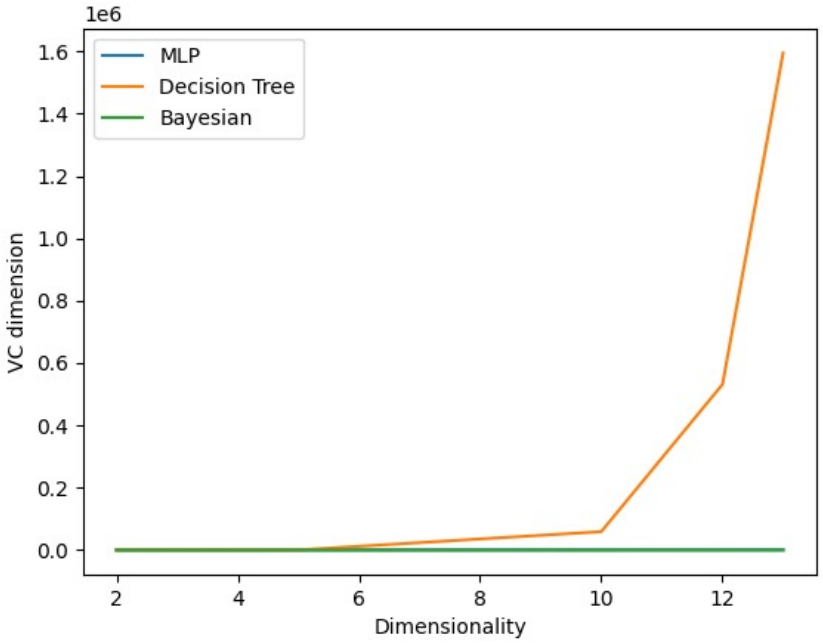
$$\boldsymbol{d_{VC}(BDT)} = 2\times\left(m + \frac{m(m+1)}{2}\right) + 1 = \boldsymbol{m^2+3m+1}$$

For $m = 5$: $\boldsymbol{d_{VC}(MLP)} = 5^2 + 3 \times 5 + 1 = \boldsymbol{41}$.

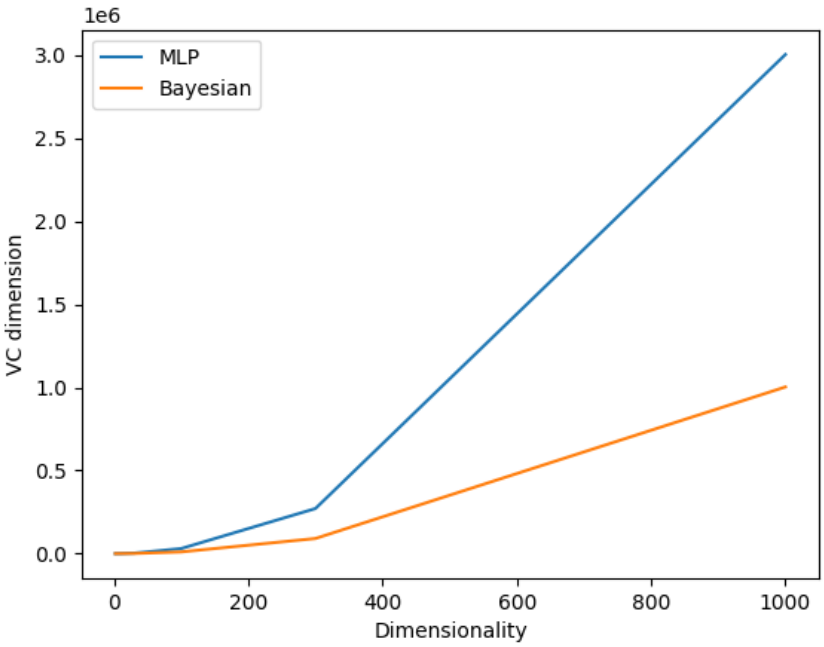
Aprendizagem 2021/22

Homework IV – Group 6

(b) After analysing the following graph, we concluded that the decision tree has a better shattering potential for data with higher dimensionality levels (over 5), at the cost of a higher complexity (which can induce overfitting errors), due to the exponential growth rate of its VC dimension.



(c) The following graph shows us that the MLP has a higher VC dimension for high-dimension samples, showing, as a result, a higher complexity and better discriminative capacity.



TÉCNICO LISBOA

Aprendizagem 2021/22

Homework IV – Group 6

II. Programming and critical analysis

4) Using the code in the Appendix, the **k-means** algorithm was applied for k ∈ {2, 3}. After the clustering was made, the following values for ECR and silhouette were computed:

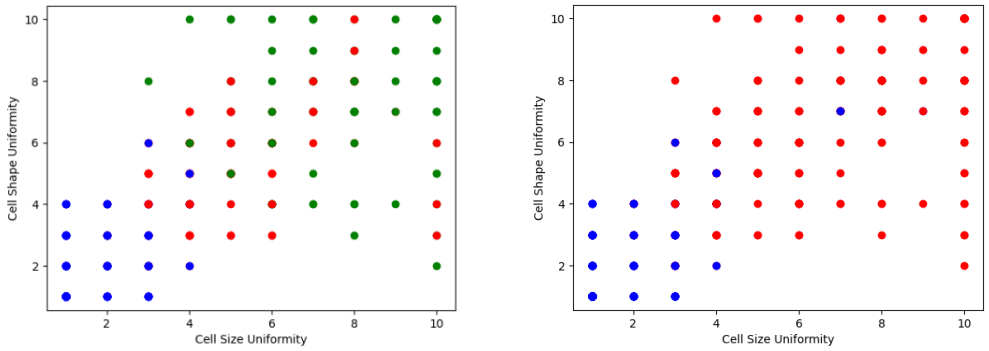
k	ECR	Silhouette
2	13.5	**0.5968**
3	**6.6667**	0.5250

More cohesive and separated clusters were created after using k-means with **k = 2**, due to a **higher silhouette score**, while a **lower ECR** was obtained for **k = 3**, meaning **a separation in 3 may create a set of clusters whose outputs inside each one don't differ as much as in a 2-cluster solution**.

It's worth noting that the clustering solutions, and, consequently, the ECR and silhouette scores, may change depending on the initial centroids (for this exercise, the function's default means were used).

5) The feature selection function, based on mutual information, returned the features **"Cell Size Uniformity"** and **"Cell Shape Uniformity"**.

The clustering solution for k = 3, obtained in **4)**, is shown on the left, while the output data (**blue = benign**, **red = malignant**) is shown on the right. It's worth informing that there are some overlapped observations since they have equal values for the 2 analysed features. These can return different clusters (since the clustering algorithm considers all 10 features) and classifications, so some of the outputs might be unobservable.



6) By analysing the produced solution with the features that were chosen in **5)**, we noticed a **lower cohesion and separation in the red and green clusters** (left hand side figure), compared to the blue cluster. This might indicate **that the observations in each of those two clusters must be closer to each other if we considered other sets of variables.**

Additionally, we can conclude **most of the observations on the blue cluster correspond to benign diagnoses, and vice-versa**, while **those on the red and green clusters are deeply tied to malign diagnoses**, confirming the low ECR value and the good discriminative power of the clustering solution, despite being fully unsupervised.

Finally, we can infer that these two features are able to separate benign and malignant observations fairly well (right hand side figure). However, the fact that lower values for both features are associated with benign diagnoses leads us to assume that higher uniformity rates are actually related to lower values.

III. APPENDIX

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.io import arff
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.feature_selection import mutual_info_classif, SelectKBest

def loadDataFrame(file_name):
    data = arff.loadarff(file_name)
    df = pd.DataFrame(data[0])
    df['Class'] = df['Class'].str.decode('utf-8')
    return df.values.tolist()

def splitFeatureLabel(df):
    df_features = [x[:-1] for x in df]
    df_labels = [x[-1] for x in df]

    return df_features, df_labels

def findClustersAndECR(k, kmeans_labels, df_labels):
    clusters = []
    counts = []
    for i in range(k):
        clusters.append([])
    for i in range(len(kmeans_labels)):
        clusters[kmeans_labels[i]].append(df_labels[i])
    for i in range(k):
        _, freq = np.unique(clusters[i], return_counts = True)
        max_freq = np.amax(freq)
        counts.append(len(clusters[i])-max_freq)
    return np.mean(counts)

def computeKMeans(df_features, k):
    kmeans = KMeans(n_clusters = k)
    kmeans_labels = kmeans.fit_predict(df_features)
    return kmeans_labels

def computeKMeansAndSilhouette(df_features, df_labels):
    k = [2,3]
    for ki in k:
        kmeans_labels = computeKMeans(df_features,ki)
        print(findClustersAndECR(ki, kmeans_labels, df_labels))
        print(silhouette_score(df_features, labels=kmeans_labels))

# The code continues in the next page!
```

```
def getHigherMutualInfoClusters(df_features, df_labels):

    def plotGraph(labels, colors, file_name):
        result = zip(new_features,labels)
        for value in result:
            plt.scatter(x=value[0][0], y=value[0][1], color=colors[int(value[1])])

        plt.xlabel("Cell Size Uniformity")
        plt.ylabel("Cell Shape Uniformity")
        plt.savefig(file_name)
        plt.show()

    kmeans_labels = computeKMeans(df_features,3)

    sel_cols = SelectKBest(mutual_info_classif, k = 2)
    sel_cols.fit(df_features,df_labels)

    indexes = [int(col[1]) for col in sel_cols.get_feature_names_out()]
    new_features = ([[x[i] for i in indexes] for x in df_features])

    plotGraph(kmeans_labels,["red","blue","green"],"clusters.png")
    plt.clf()
    plotGraph([1 if label=="benign" else 0 for label in df_labels],["red","blue"],"classified.png")

if __name__ == "__main__":
    df = loadDataFrame("breast.w.arff")
    df_features, df_labels = splitFeatureLabel(df)
    computeKMeansAndSilhouette(df_features, df_labels)
    getHigherMutualInfoClusters(df_features, df_labels)
```

END