Aprendizagem 2021/22
Homework IV – Group 35

I. Pen-and-paper

1) Temos,

$$\pi_1 = P(c_1 = 1) = 0.7 \qquad P(y_1, y_2 \mid c_1 = 1) = N(\mu_1, \Sigma_1) = N\left(\begin{pmatrix}2\\4\end{pmatrix}, \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$$

$$\pi_2 = P(c_2 = 1) = 0.3 \qquad P(y_1, y_2 \mid c_2 = 1) = N(\mu_2, \Sigma_2) = N\left(\begin{pmatrix}-1\\-4\end{pmatrix}, \begin{pmatrix}2 & 0\\0 & 2\end{pmatrix}\right)$$

Nota: todos os cálculos foram realizados em *Python* sem recorrer a arredondamentos intermédios, porém, para os apresentar no relatório, foram realizados arredondamentos a 4 casas decimais.

E-Step:

Likelihoods:

$$P(x_n \mid c_k = 1) = N(x_n \mid \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k{}^{-1}(x_n - \mu_k)\right)$$

$$P(x_1 \mid c_1 = 1) = N\left(\begin{pmatrix}2\\4\end{pmatrix} \mid \begin{pmatrix}2\\4\end{pmatrix}, \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right) = \frac{1}{(2\pi)^{\frac{1}{2}}|\Sigma_k|^{\frac{1}{2}}} = 0.1592$$

$$P(x_1 \mid c_2 = 1) = N\left(\begin{pmatrix}2\\4\end{pmatrix} \mid \begin{pmatrix}-1\\-4\end{pmatrix}, \begin{pmatrix}2 & 0\\0 & 2\end{pmatrix}\right) = 9.4388e^{-10}$$

$P(x_2 \mid c_1 = 1) = 2.2391e^{-17}$ $\quad P(x_3 \mid c_1 = 1) = 0.0002$ $\quad P(x_4 \mid c_1 = 1) = 7.2256e^{-6}$
$P(x_2 \mid c_2 = 1) = 0.0769$ $\quad P(x_3 \mid c_2 = 1) = 9.8206e^{-6}$ $\quad P(x_4 \mid c_2 = 1) = 2.8137e^{-6}$

Joints:

$$P(c_k = 1, x_n) = P(c_k = 1) \cdot P(x_n \mid c_k = 1) = \pi_k \cdot N(x_n \mid \mu_k, \Sigma_k)$$

$P(c_1 = 1, x_1) = P(c_1 = 1) \cdot P(x_1 \mid c_1 = 1) = \pi_1 \cdot N(x_1 \mid \mu_1, \Sigma_1) = 0.7 \times 0.1592 = 0.1114$

$P(c_2 = 1, x_1) = P(c_2 = 1) \cdot P(x_1 \mid c_2 = 1) = 0.3 \times 9.4388e^{-10} = 2.8316e^{-10}$

$P(c_1 = 1, x_2) = 1.5674e^{-17}$ $\quad P(c_1 = 1, x_3) = 0.0002$ $\quad P(c_1 = 1, x_4) = 5.0579e^{-6}$

$P(c_2 = 1, x_2) = 0.0239$ $\quad P(c_2 = 1, x_3) = 2.9462e^{-6}$ $\quad P(c_2 = 1, x_4) = 8.4410e^{-7}$

De seguida, temos

$$p(x_n) = \sum_{k=1}^{K} p(c_k = 1, x_n) = \sum_{k=1}^{K} \pi_k \cdot N(x_n \mid \mu_k, \Sigma_k)$$

$p(x_1) = \sum_{k=1}^{2} p(c_k = 1, x_1) = 0.1114$

$p(x_2) = 0.0239$ $\quad p(x_3) = 0.0002$ $\quad p(x_4) = 5.9020e^{-6}$

Agora usando o Teorema de Bayes (normalização)

$$\gamma(c_{nk}) = p(c_k = 1 \mid x_n) = \frac{p(c_k = 1, x_n)}{p(x_n)}$$

$\gamma(c_{11}) = p(c_1 = 1 \mid x_1) \approx 1,$ $\quad \gamma(c_{21}) = 6.5653e^{-16},$ $\quad \gamma(c_{31}) = 0.9827,$ $\quad \gamma(c_{41}) = 0.8570$
$\gamma(c_{12}) = p(c_2 = 1 \mid x_1) = 2.5417e^{-9},$ $\quad \gamma(c_{22}) \approx 1,$ $\quad \gamma(c_{32}) = 0.0173,$ $\quad \gamma(c_{42}) = 0.1430$

M-Step:

$$P_1 = \begin{pmatrix} 1 \\ 6.5653e^{-16} \\ 0.9827 \\ 0.8570 \end{pmatrix} \qquad P_2 = \begin{pmatrix} 2.5417e^{-9} \\ 1 \\ 0.0173 \\ 0.1430 \end{pmatrix}$$

$$w_1 = 2.8397 \qquad\qquad w_2 = 1.1603$$

Calcular as novas medias:

$$\mu_k = \frac{1}{w_k} \cdot \sum_{k=1}^{N} \gamma(c_{nk}) \cdot x_n$$

$$\mu_1 = \frac{1}{1.1603}\left(1 \cdot \begin{pmatrix} 2 \\ 4 \end{pmatrix} + 6.5653e^{-16} \cdot \begin{pmatrix} -1 \\ -4 \end{pmatrix} + 0.9827 \cdot \begin{pmatrix} -1 \\ 2 \end{pmatrix} + 0.8570 \cdot \begin{pmatrix} 4 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} 1.5654 \\ 2.1007 \end{pmatrix}$$

$$\mu_2 = \frac{1}{2.0971}\left(2.5417e^{-9} \cdot \begin{pmatrix} 2 \\ 4 \end{pmatrix} + 1 \cdot \begin{pmatrix} -1 \\ -4 \end{pmatrix} + 0.0173 \cdot \begin{pmatrix} -1 \\ 2 \end{pmatrix} + 0.1430 \cdot \begin{pmatrix} 4 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} -0.3837 \\ -3.4176 \end{pmatrix}$$

e as novas matrizes de covariância:

$$\Sigma_k = \frac{1}{w_k} \cdot \sum_{n=1}^{N} \gamma(c_{nk}) \cdot (x_n - \mu_k) \cdot (x_n - \mu_k)^T = \frac{1}{w_k} \cdot \sum_{n=1}^{N} \gamma(c_{nk}) \cdot X_n$$

Para $k = 1$:

$$X_1 = \begin{pmatrix} 2 - 1.5654 \\ 4 - 2.1007 \end{pmatrix} \cdot \begin{pmatrix} 2 - 1.5654 & 4 - 2.1007 \end{pmatrix} = \begin{pmatrix} 0.1889 & 0.8255 \\ 0.8254 & 3.6072 \end{pmatrix}$$

$$X_2 = \begin{pmatrix} -1 - 1.5654 \\ -4 - 2.1007 \end{pmatrix} \cdot \begin{pmatrix} -1 - 1.5654 & -4 - 2.1007 \end{pmatrix} = \begin{pmatrix} 6.5812 & 15.6507 \\ 15.6507 & 37.2189 \end{pmatrix}$$

$$X_3 = \begin{pmatrix} 6.5812 & 0.2584 \\ 0.2584 & 0.0101 \end{pmatrix} \;,\; X_4 = \begin{pmatrix} 5.9274 & -5.1145 \\ -5.1145 & 4.4131 \end{pmatrix}$$

$$\Sigma_1 = \frac{1}{1.1603}(1 \cdot X_1 + 6.5653e^{-16} \cdot X_2 + 0.9827 \cdot X_3 + 0.8570 \cdot X_4) = \begin{pmatrix} 4.1328 & -1.1634 \\ -1.1634 & 2.6056 \end{pmatrix}$$

Para $k = 2$:

$$X_1 = \begin{pmatrix} 2 - (-0.3837) \\ 4 - (-3.4176) \end{pmatrix} \cdot \begin{pmatrix} 2 - (-0.3837) & 4 - (-3.4176) \end{pmatrix} = \begin{pmatrix} 5.6820 & 17.6813 \\ 17.6813 & 55.0205 \end{pmatrix}$$

$$X_2 = \begin{pmatrix} 0.3798 & 0.3589 \\ 0.3589 & 0.3392 \end{pmatrix} \;,\; X_3 = \begin{pmatrix} 0.3798 & -3.3388 \\ -3.3388 & 29.3501 \end{pmatrix} \;,\; X_4 = \begin{pmatrix} 19.2169 & 14.9817 \\ 14.9817 & 11.6798 \end{pmatrix}$$

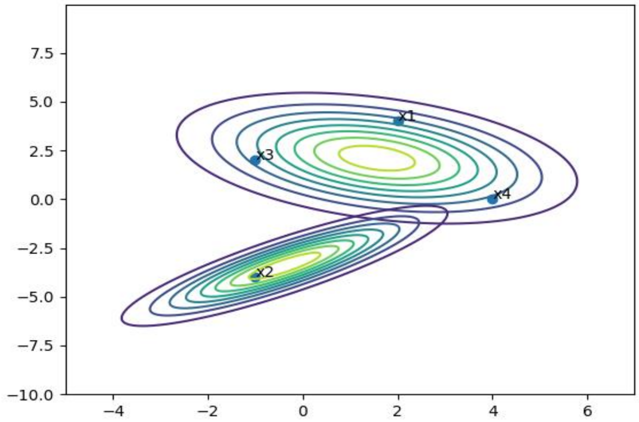
$$\Sigma_2 = \frac{1}{2.0971}(2.5417e^{-9} \cdot X_1 + 1 \cdot X_2 + 0.0173 \cdot X_3 + 0.1430 \cdot X_4) = \begin{pmatrix} 2.7017 & 2.1062 \\ 2.1062 & 2.1692 \end{pmatrix}$$

Novos *Priors*:

$$\pi_k = p(c_k = 1) = \frac{w_k}{W}$$

$$\pi_1 = p(c_1 = 1) = \frac{w_1}{w_1 + w_2} = 0.7099 \qquad , \qquad \pi_2 = p(c_2 = 1) = \frac{w_2}{w_1 + w_2} = 0.2901$$

<u>Sketch da solução:</u>



2) Ao comparar os valores das normais com os novos parâmetros ($N(x_n \mid \mu_1, \Sigma_1)$ *e* $N(x_n \mid \mu_2, \Sigma_2)$), concluímos que $\{x_1, x_3, x_4\} \in cluster_1$ e $\{x_2\} \in cluster_2$ então,

$$s(x_1) = 1 - \frac{a(x_1)}{b(x_1)} = 1 - \frac{\frac{1}{2}(\|x_1 - x_3\|_2 + \|x_1 - x_4\|_2)}{\|x_1 - x_2\|_2} = 0.5273$$

$$s(x_2) = 1 - \frac{a(x_2)}{b(x_2)} = 1 - \frac{0}{\frac{1}{3}(\|x_2 - x_1\|_2 + \|x_2 - x_3\|_2 + \|x_2 - x_4\|_2)} = 1$$

$$s(x_3) = 1 - \frac{a(x_3)}{b(x_3)} = 1 - \frac{\frac{1}{2}(\|x_3 - x_1\|_2 + \|x_3 - x_4\|_2)}{\|x_3 - x_2\|_2} = 0.2508$$

$$s(x_4) = 1 - \frac{a(x_4)}{b(x_4)} = 1 - \frac{\frac{1}{2}(\|x_4 - x_1\|_2 + \|x_4 - x_3\|_2)}{\|x_4 - x_2\|_2} = 0.2303$$

$$s(c_1) = \frac{s(x_1) + s(x_3) + s(x_4)}{3} = 0.3361$$

$$s(c_2) = s(x_2) = 1$$

A silhueta da solução é a média das silhuetas dos clusters:

$$silhouette(C) = \frac{s(c_1) + s(c_2)}{2} = 0.6681$$

3) A VC-Dimension é a medida de graus de liberdade de um classificador. Para aproximar o número de graus de liberdade vamos estimar o número de parâmetros para cada classificador pedido. Começamos por demonstrar para o caso de N dimensões e depois explicitamos para dimensão igual a 5.

a)

i. <u>MLP com 3 hidden layers com tantos nós como o número de variáveis de input (N):</u>

N – N – N – N – 1

$W^{[1]} = N \times N$ $W^{[2]} = N \times N$ $W^{[3]} = N \times N$ $W^{[4]} = 1 \times N$

$b^{[1]} = N \times 1$ $b^{[2]} = N \times 1$ $b^{[3]} = N \times 1$ $b^{[4]} = 1 \times 1$

No MLP a input layer não tem parâmetros pois são inputs.

Entre a input layer e a primeira hidden layer temos a matriz de de pesos $W^{[1]}$ de dimensões $N \times N$ e o vetor bias $b^{[1]}$ de dimensões $N \times 1$ e, portanto, temos $\mathrm{N}^2 + \mathrm{N}$ parâmetros.

Entre a primeira e segunda hidden layers temos a matriz de pesos $W^{[2]}$ de dimensões $N \times N$ e o vetor bias $b^{[2]}$ de dimensões $N \times 1$ e, portanto, temos também $\mathrm{N}^2 + \mathrm{N}$ parâmetros.

Entre a segunda e terceira hidden layers temos a matriz de pesos $W^{[3]}$ e o vetor bias $b^{[3]}$ com as mesmas dimensões que entre a primeira e segunda hidden layers e, portanto, temos novamente $\mathrm{N}^2 + \mathrm{N}$ parâmetros.

Entre a terceira hidden layer e a output layer temos a matriz de pesos $W^{[4]}$ de dimensões $1 \times N$ e o vetor bias $b^{[2]}$ de dimensões 1×1 e portanto temos $\mathrm{N} + 1$ parâmetros.

Não havendo mais parâmetros, temos então um total de $3N^2 + 4N + 1$ parâmetros, ou seja, para N=5 temos $d_{VC} = 3 \cdot 5^2 + 4 \cdot 5 + 1 = 96$.

ii. <u>Árvore de decisão em que as variáveis são discretizadas em 3 bins</u>

Para uma árvore de decisão com d features em que as variáveis são discretizadas em 3 valores podemos notar que o número máximo de pontos diferentes é 3^d. Por consequência podemos admitir desde já que $d_{VC} \leq 3^d$.

Desta forma, dividindo em três todas as features, conseguimos criar uma árvore com altura d+1 com uma folha para cada ponto possível. Assim, temos uma árvore que consegue fazer shatter a todo o dataset, uma vez que, independentemente do ponto e da label escolhida, vai conseguir realizar a classificação. Provamos assim que $d_{VC} \geq 3^d$.

Por fim, combinando as duas desigualdades podemos concluir que $d_{VC} = 3^d$, ou seja, para $\mathrm{N} = 5$ temos $d_{VC} = 3^5 = 243$.

iii. <u>Classificador Bayesiano com likelihood Gaussiana Multivariada</u>

Para o Classificador Bayesiano precisamos de estimar os priors e as likelihoods.

No caso dos priors precisamos apenas de um parâmetro, uma vez que, sendo o classificar binário, calculado o primeiro prior, P(C), conseguimos determinar o segundo fazendo 1-P(C).

No caso das likelihoods, sendo Gaussiana Multivariada, será necessário um vetor de médias e uma matriz de covariâncias. Para N dimensões:

- O vetor de médias terá dimensões $N \times 1$ e, portanto, contamos com N parâmetros.

- A matriz de covariâncias terá dimensões $N \times N$, no entanto, a matriz é simétrica pelo que apenas contamos com os parâmetros da diagonal e da parte da matriz superior á diagonal, o que nos dá $N + \frac{N^2-N}{2}$ parâmetros.

Como se trata de um classificador binário temos 2 likelihoods com o mesmo número de parâmetros: $P(y_1 \ldots y_{N)}|C = 0)$ e $P(y_1 \ldots y_{N)}|C = 1)$. Sendo N o número de dimensões, então temos um total de $1 + 2\left(N + N + \frac{N^2-N}{2}\right) = 1 + 3N + N^2$ parâmetros. Desta forma, para $\mathrm{N} = 5$ temos $d_{VC} = 1 + 3 \cdot 5 + 5^2 = 41$.

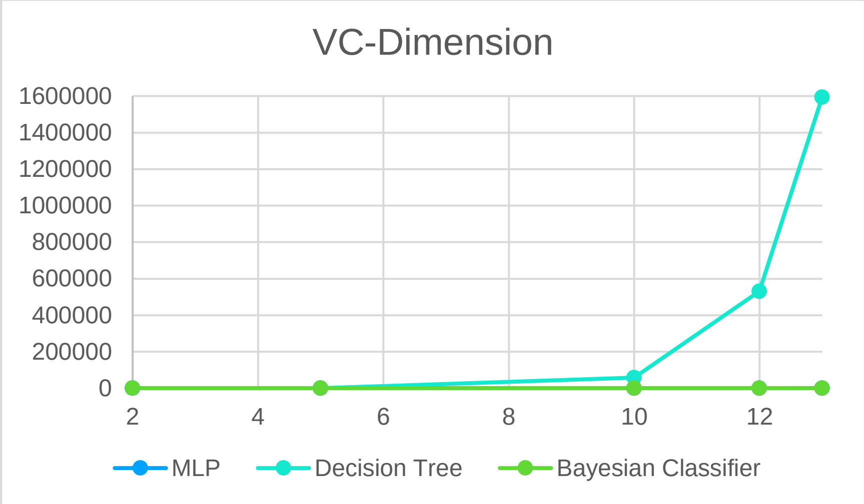
b)

Para calcularmos as VC-Dimensions para cada caso utilizamos as seguintes fórmulas obtidas na alínea anterior:

i) $d_{VC} = 1 + 4N + 3N^2$

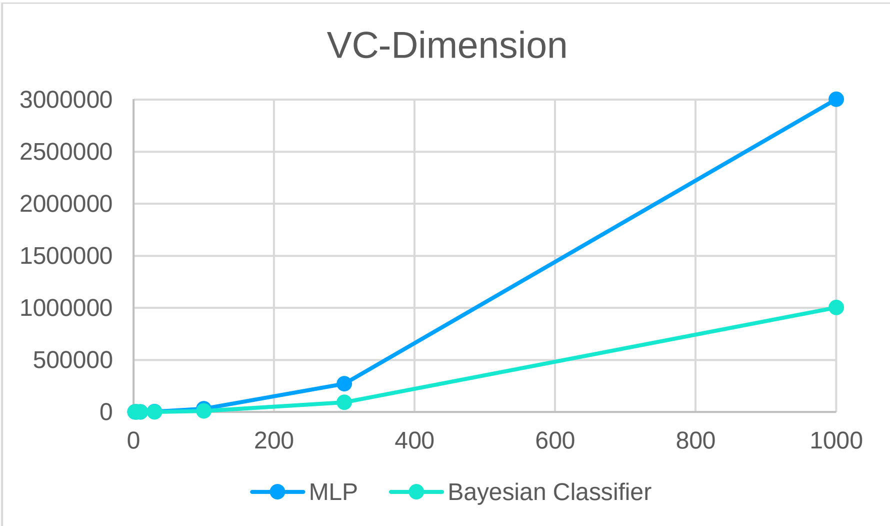
ii) $d_{VC} = 3^N$

iii) $d_{VC} = 1 + 3N + N^2$



Observando o seguinte gráfico conseguimos concluir que a Árvore de Decisão é o classificador que, com o aumento de features, tem maior aumento de VC-Dimension (exponencial) o que indica que o modelo é mais complexo e tem uma maior suscetibilidade a *overfitting*, enquanto que os outros classificadores possuem d_{VC} com crescimento quadrático.

c)



Observando o gráfico acima podemos concluir que a VC-Dimension do MLP aumenta aproximadamente três vezes mais com o aumento das features (gerando uma diferença considerável para valores extremamente grandes) e, por isso, é mais complexo e suscetível a *overfitting*, enquanto que o Classificador Bayesiano é mais suscetível a *underfitting*.

II. Programming and critical analysis

4) Ao aplicar *k-means clustering* não supervisionado aos dados originais obtivemos os seguintes valores:

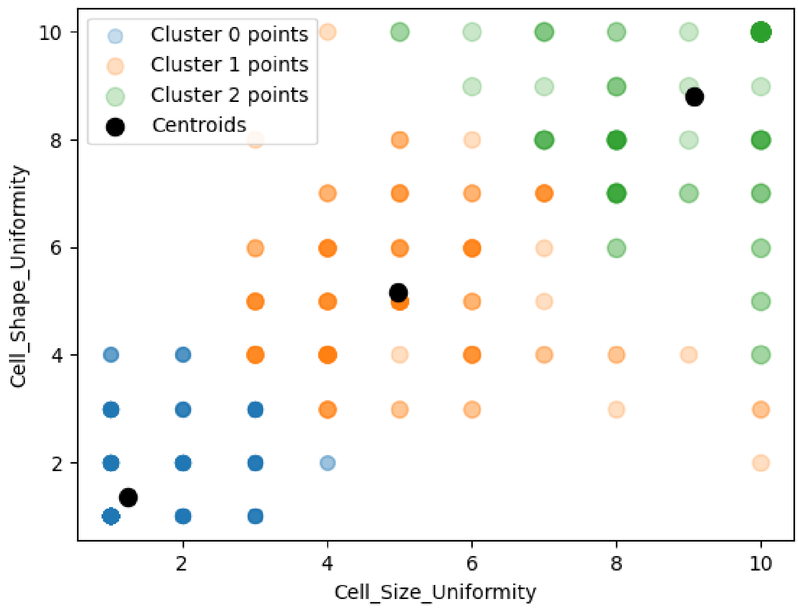
Para <u>$k = 2$</u>: ECR = 13.5 ; Silhouette score = 0.5968. Para <u>$k = 3$</u>: ECR = 6.6667 ; Silhouette score = 0.5245.

a. Tendo por base estes valores, conseguimos observar que para k = 2, a error-classification rate (ECR) é o dobro pelo que existem mais pontos mal classificados em cada cluster comparando com as labels verdadeiras dadas. Deste, modo a solução com 3 clusters aparenta ajustar-se melhor aos *targets*.

b. Por outro lado, o *Silhouette Coefficient* é bastante semelhante, sendo até superior no caso com 2 clusters. Um maior *Silhouette Coefficient* indica uma melhor técnica de *clustering*, pelo que quanto mais o valor estiver perto de 1, mais os clusters estão bem separados e distintos. Ambos os valores obtidos (0.5968 e 0.5245) mostram existem evidências aceitáveis para afirmar que os clusters estão bem separados e coesos. Comparando os valores, observamos então que a k = 2 apresenta uma maior separação e coesão dos clusters.

Tendo ambas as informações em consideração para cada k, temos que para k = 2, os resultados produzidos apresentam um maior erro na classificação, no entanto, os clusters encontram-se mais bem separados e distintos. Para k = 3, os clusters encontram-se ligeiramente menos separados, no entanto, a classificação é mais precisa.

5) Como apresentado no código no Appendix, após utilizar o *SelectKBest* para excluir todas as features exceto as duas com maior *mutual information* e fixando o k = 3, obtivemos a seguinte solução:



6) Tendo em consideração os resultados obtidos no exercício 5, observamos que ter em conta apenas as melhores 2 features não é suficiente para obter resultados melhores. Os clusters, após o *KMeans* considerando apenas as duas melhores features, permanecem bem afastados, no entanto, existe uma quantidade elevada de pontos mal classificados se tivermos em conta a sua classificação com todas as features. A representação em 2D torna difícil tirar grandes conclusões dado que muitos pontos ficam sobrepostos no gráfico, tornando difícil a sua representação e leitura a 2 dimensões. Decidimos então avaliar os dados de acordo com o ECR (critério externo) e com o *Silhouette Coefficient* (critério interno).

Tendo apenas em consideração as top-2 features para com maior informação mútua e as labels resultantes do treino com todas as features, observámos um ECR de 11.(6) o que confirma que existe um aumento de pontos mal classificados em cada cluster. Por outro lado, obtivemos um *Silhouette Coefficient* de 0.7074 o que indica uma maior separação/coesão dos clusters. Tendo em conta duas features, seria de esperar que os clusters fossem melhor separados, no entanto, apenas duas features não são suficientes para melhorar a precisão da classificação.

III. APPENDIX

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

# Variables
k = [2,3]

data = arff.loadarff('breast.w.arff')
df = pd.DataFrame(data[0])
df = df.dropna()
data = df.drop(columns=["Class"]).values
results = df[df.keys()[-1]].astype('string').values

def ECR(results, pred_labels):
    clusters = np.unique(pred_labels)
    ecr=0
    for i in clusters:
        points = results[pred_labels == i]
        _, counts = np.unique(points, return_counts=True)
        ecr += 1/len(clusters)*(np.sum(counts) - np.max(counts))
    return ecr

# 4
for i in k:
    kmeans = KMeans(n_clusters=i).fit(data)
    pred_labels = kmeans.labels_

    print(f'## {i}-Means: ##')
    print(f'\tECR: {ECR(results, pred_labels)}')

    silh_score = silhouette_score(data, pred_labels, metric='euclidean')
    print(f'\tSilhouette score: {silh_score}')

# 5
kmeansX = KMeans(n_clusters=3)
kmeans4 = kmeans.fit(data)
selector = SelectKBest(score_func=mutual_info_classif, k=2)
data_best2 = selector.fit_transform(data, results)
kmeans5 = kmeans.fit(data_best2)
centroids = kmeans5.cluster_centers_
labels4 = kmeans4.labels_

for i in np.unique(labels4):
    plt.scatter(data_best2[labels4 == i,0], data_best2[labels4 == i,1], label=f'Cluster {i} points',
                    alpha=0.25, s=50+15*i)
```

```
plt.scatter(centroids[:,0], centroids[:,1], s=75, c='black', label='Centroids')
cols = selector.get_support(indices=True)
features = df.iloc[:,cols].columns
plt.xlabel(features[0]) ; plt.ylabel(features[1])
plt.legend()
plt.show()

#6
print(f'\nWith top-2 features:\n\tECR: {ECR(results, labels4)}')
silh_score5 = silhouette_score(data_best2, labels4, metric='euclidean')
print(f'\tSilhouette score: {silh_score5}')
```

END