Homework 4 - Group 076

Aprendizagem 2021/2022

1 Pen and Paper

1) Let $\mathbf{c}$ be a 2-dimensional random vector for which $c_k \in \{0,1\}$ is an indicator of cluster k ($k \in \{1,2\}$ and $\sum c_k = 1$). Let $\pi_k = p(c_k = 1)$ and let $\gamma(c_{nk}) = p(c_k = 1|\mathbf{x}^{(n)})$.

- **E-step**

  In this step, we assign each instance $\mathbf{x}^{(n)}$ to the cluster c_k that yields the largest posterior $\gamma(c_{nk})$. Given that $\mathbf{x}^{(n)}|c_k = 1 \sim \mathcal{N}(\mu_k, \Sigma_k)$, by Bayes rule, we have:

$$\gamma(c_{nk}) = p(c_k = 1|\mathbf{x}^{(n)}) = \frac{p(c_k = 1)p(\mathbf{x}^{(n)}|c_k = 1)}{\sum_{l=1}^{2} p(c_l = 1)p(\mathbf{x}^{(n)}|c_l = 1)} = \frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k)}{\sum_{l=1}^{2} \pi_l \mathcal{N}(\mathbf{x}^{(n)}|\mu_l, \Sigma_l)}$$

  Setting $\mu_k = \mathbf{x}^{(k)}$ ($k \in \{1,2\}$), we now compute the posteriors:

  - For $\mathbf{x}^{(1)}$:

$$\gamma(c_{11}) = \frac{\pi_1 \mathcal{N}(\mathbf{x}^{(1)}|\mu_1, \Sigma_1)}{\sum_{l=1}^{2} \pi_l \mathcal{N}(\mathbf{x}^{(1)}|\mu_l, \Sigma_l)} = \frac{0.7 \cdot 0.1591549}{0.7 \cdot 0.1591549 + 0.3 \cdot 0.0} = \frac{0.11141}{0.11141} = 1.0$$
$$\gamma(c_{12}) = 1 - \gamma(c_{11}) = 0.0$$

  - For $\mathbf{x}^{(2)}$:

$$\gamma(c_{21}) = \frac{0.7 \cdot 0.0}{0.7 \cdot 0.0 + 0.3 \cdot 0.0795775} = 0.0 \quad \gamma(c_{22}) = 1 - \gamma(c_{21}) = 1.0$$

  - For $\mathbf{x}^{(3)}$:

$$\gamma(c_{31}) = \frac{0.7 \cdot 0.0002393}{0.7 \cdot 0.0002393 + 0.3 \cdot 0.0000098} = 0.98271 \quad \gamma(c_{32}) = 1 - \gamma(c_{31}) = 0.01729$$

  - For $\mathbf{x}^{(4)}$:

$$\gamma(c_{41}) = \frac{0.7 \cdot 0.0000072}{0.7 \cdot 0.0000072 + 0.3 \cdot 0.0000028} = 0.85698 \quad \gamma(c_{42}) = 1 - \gamma(c_{41}) = 0.14302$$

- **M-step**

  We re-compute μ_k, Σ_k and π_k ($k \in \{1,2\}$) so as to increase the likelihood of the data. The new cluster centroids are given by:

$$\mu_k = \frac{\sum_{n=1}^{4} \gamma(c_{nk})\mathbf{x}^{(n)}}{\sum_{n=1}^{4} \gamma(c_{nk})}$$

  And so we have:

$$\mu_1 = \frac{1.0 \begin{bmatrix} 2 \\ 4 \end{bmatrix} + 0.0 \begin{bmatrix} -1 \\ -4 \end{bmatrix} + 0.98271 \begin{bmatrix} -1 \\ 2 \end{bmatrix} + 0.85698 \begin{bmatrix} 4 \\ 0 \end{bmatrix}}{1.0 + 0.0 + 0.98271 + 0.85698} = \begin{bmatrix} 1.56538 \\ 2.10073 \end{bmatrix}$$

$$\mu_2 = \frac{0.0\begin{bmatrix}2\\4\end{bmatrix} + 1.0\begin{bmatrix}-1\\-4\end{bmatrix} + 0.01729\begin{bmatrix}-1\\2\end{bmatrix} + 0.14302\begin{bmatrix}4\\0\end{bmatrix}}{0.0 + 1.0 + 0.01729 + 0.14302} = \begin{bmatrix}-0.3837\\-3.41758\end{bmatrix}$$

The new covariance matrices are now given by:

$$\Sigma_k = \frac{\sum_{n=1}^4 \gamma(c_{nk})(\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T}{\sum_{n=1}^4 \gamma(c_{nk})}$$

And so:

$$\Sigma_1 = \frac{1.0\begin{bmatrix}4.0 & 8.0\\8.0 & 16.0\end{bmatrix} + 0.0\begin{bmatrix}1.0 & 4.0\\4.0 & 16.0\end{bmatrix} + 0.98271\begin{bmatrix}1.0 & -2.0\\-2.0 & 4.0\end{bmatrix} + 0.85698\begin{bmatrix}4.0 & 0.0\\0.0 & 0.0\end{bmatrix}}{1.0 + 0.0 + 0.98271 + 0.85698} = \begin{bmatrix}4.13282 & -1.16337\\-1.16337 & 2.6056\end{bmatrix}$$

$$\Sigma_2 = \frac{0.0\begin{bmatrix}4.0 & 8.0\\8.0 & 16.0\end{bmatrix} + 1.0\begin{bmatrix}1.0 & 4.0\\4.0 & 16.0\end{bmatrix} + 0.01729\begin{bmatrix}1.0 & -2.0\\-2.0 & 4.0\end{bmatrix} + 0.14302\begin{bmatrix}4.0 & 0.0\\0.0 & 0.0\end{bmatrix}}{0.0 + 1.0 + 0.01729 + 0.14302} = \begin{bmatrix}2.70166 & 2.10624\\2.10624 & 2.16924\end{bmatrix}$$

Finally, the new prior probabilities can now be written as:

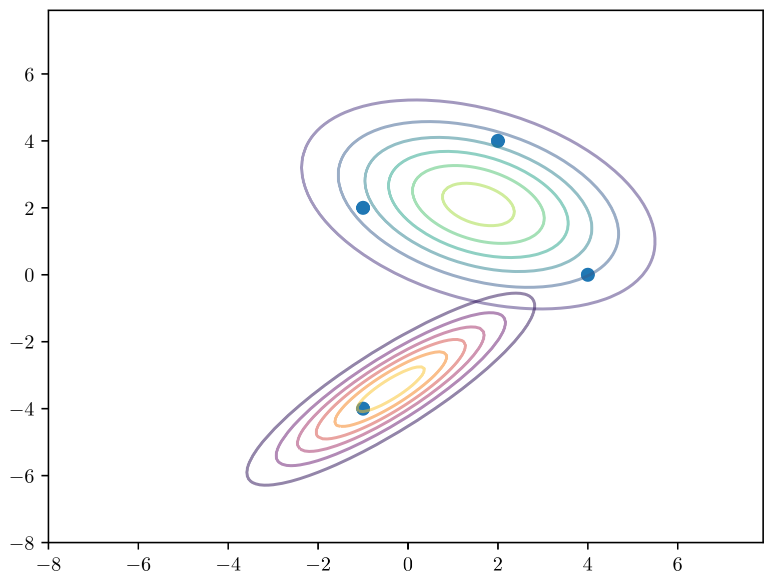
$$\pi_k = \frac{\sum_{n=1}^4 \gamma(c_{nk})}{4}$$

and thus:

$$\pi_1 = \frac{1.0 + 0.0 + 0.98271 + 0.85698}{4} = 0.70992$$

$$\pi_2 = \frac{0.0 + 1.0 + 0.01729 + 0.14302}{4} = 0.29008$$

Hence, this iteration of the EM algorithm assigned points $\mathbf{x}^{(1)}$, $\mathbf{x}^{(3)}$ and $\mathbf{x}^{(4)}$ to cluster c_1, while $\mathbf{x}^{(2)}$ was assigned to cluster c_2. The following figure depicts this clustering:



2) First, we compute the silhouette of cluster c_1:

$$s(\mathbf{x}^{(1)}) = 1 - \frac{a(\mathbf{x}^{(1)})}{b(\mathbf{x}^{(1)})} = 1 - \frac{\frac{\|\mathbf{x}^{(1)}-\mathbf{x}^{(3)}\|_2+\|\mathbf{x}^{(1)}-\mathbf{x}^{(4)}\|_2}{2}}{\|\mathbf{x}^{(1)}-\mathbf{x}^{(2)}\|_2} = 1 - \frac{\frac{3.60555+4.47214}{2}}{8.544} = 0.52729$$
$$s(\mathbf{x}^{(3)}) = 1 - \frac{a(\mathbf{x}^{(3)})}{b(\mathbf{x}^{(3)})} = 1 - \frac{\frac{\|\mathbf{x}^{(3)}-\mathbf{x}^{(1)}\|_2+\|\mathbf{x}^{(3)}-\mathbf{x}^{(4)}\|_2}{2}}{\|\mathbf{x}^{(3)}-\mathbf{x}^{(2)}\|_2} = 1 - \frac{\frac{3.60555+5.38516}{2}}{6.0} = 0.25077$$
$$s(\mathbf{x}^{(4)}) = 1 - \frac{a(\mathbf{x}^{(4)})}{b(\mathbf{x}^{(4)})} = 1 - \frac{\frac{\|\mathbf{x}^{(4)}-\mathbf{x}^{(1)}\|_2+\|\mathbf{x}^{(4)}-\mathbf{x}^{(3)}\|_2}{2}}{\|\mathbf{x}^{(4)}-\mathbf{x}^{(2)}\|_2} = 1 - \frac{\frac{4.47214+5.38516}{2}}{6.40312} = 0.23027$$

$$s(c_1) = \frac{s(\mathbf{x}^{(1)}) + s(\mathbf{x}^{(3)}) + s(\mathbf{x}^{(4)})}{3} = \frac{0.52729 + 0.25077 + 0.23027}{3} = 0.33611$$

Since $|c_2| = 1$, we set $s(c_2) = 1$. The silhouette of the solution is defined as the arithmetic mean of the silhouettes of the clusters:

$$s(C) = \frac{s(c_1) + s(c_2)}{2} = \frac{0.33611 + 1.0}{2} = 0.66806$$

3) (a) i) The specified MLP is described by the following parameters: weight matrices $W^{[1]}$, $W^{[2]}$, $W^{[3]}$ and $W^{[4]}$, along with bias vectors $b^{[1]}$, $b^{[2]}$, $b^{[3]}$ and $b^{[4]}$. For input data of dimention d, we have:

- $W^{[1]}, W^{[2]}, W^{[3]} \in \mathbb{R}^{d\times d}$
- $W^{[4]} \in \mathbb{R}^{2\times d}$
- $b^{[1]}, b^{[2]}, b^{[3]} \in \mathbb{R}^{d}$
- $b^{[4]} \in \mathbb{R}^{2}$

Consequently, the total number of parameters is $3d^2 + 2d + 3d + 2 = 3d^2 + 5d + 2$. Given that the number of parameters is a reasonable estimate of the VC dimension of the model, for $d = 5$, we have an approximation of the VC dimension of the MLP of $3 \cdot 5^2 + 5 \cdot 5 + 2 = 102$.

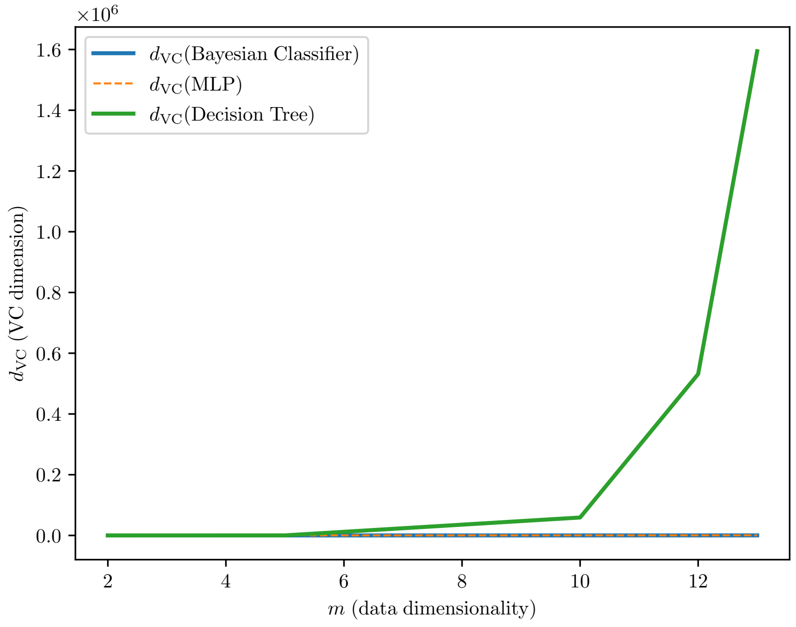
ii) For input data of dimensionality d, we know that $d_{\text{VC}} \leq 3^d$, as 3^d is precisely the size of the input space (there are d features with 3 possible values). Hence, it is impossible for the decision tree to shatter a number of points greater than 3^d. On another hand, we note that any set of size 3^d can be shattered by a decision tree; for that purpose, given any dichotomy, it suffices to consider a tree with $d+1$ levels with a single non-leaf node which is associated with a point in the set. In that node, we test equality w.r.t. that point, and we classify the example with the label of the corresponding point or continue down the tree accordingly. Thus, $d_{\text{VC}} \geq 3^d$, and so $d_{\text{VC}} = 3^d = 3^5 = 243$.

iii) We again estimate VC dimension based on the number of parameters of the model. Given a point $\mathbf{x} \in \mathbb{R}^d$ and a binary output variable z, the Bayesian Classifier estimates the following probability:

$$p(z = 1|\mathbf{x}) = \frac{p(\mathbf{x}|z = 1)p(z = 1)}{p(z = 0)p(\mathbf{x}|z = 0) + p(z = 1)p(\mathbf{x}|z = 1)}$$

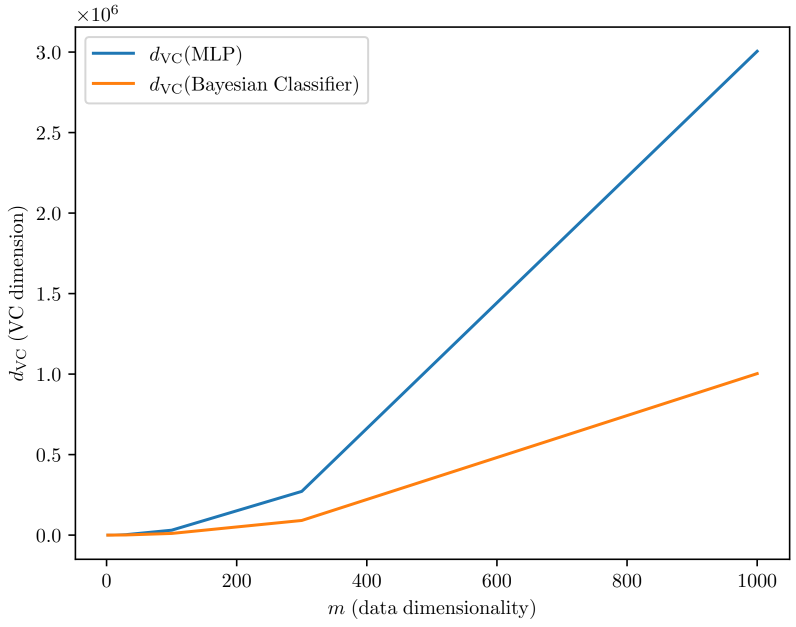
Thus, we need to estimate the prior $p(z)$ and the likelihood $p(\mathbf{x}|z)$. We know that $z \sim \text{Bernoulli}(p)$, so there is only one parameter associated with the prior; we also have that $\mathbf{x}|z = k \sim \mathcal{N}(\mu_k, \Sigma_k)$ $(k \in \{0, 1\})$, where $\mu_k \in \mathbb{R}^d$ and $\Sigma_k \in \mathbb{R}^{d\times d}$. Since Σ_k is symmetric, we can discard the lower diagonal part of the matrix. Thus, there are $1 + 2(d + d + \frac{d^2-d}{2}) = 1 + 4d + d^2 - d = d^2 + 3d + 1$ parameters, and we have an approximate VC dimension of $5^2 + 3 \cdot 5 + 1 = 41$.

(b)



In comparison to the other models, the VC dimension of the decision tree starts to explode exponentially for data dimensionalities as low as 12 (thus requiring exponentially larger datasets to mitigate overfitting risks).

(c)



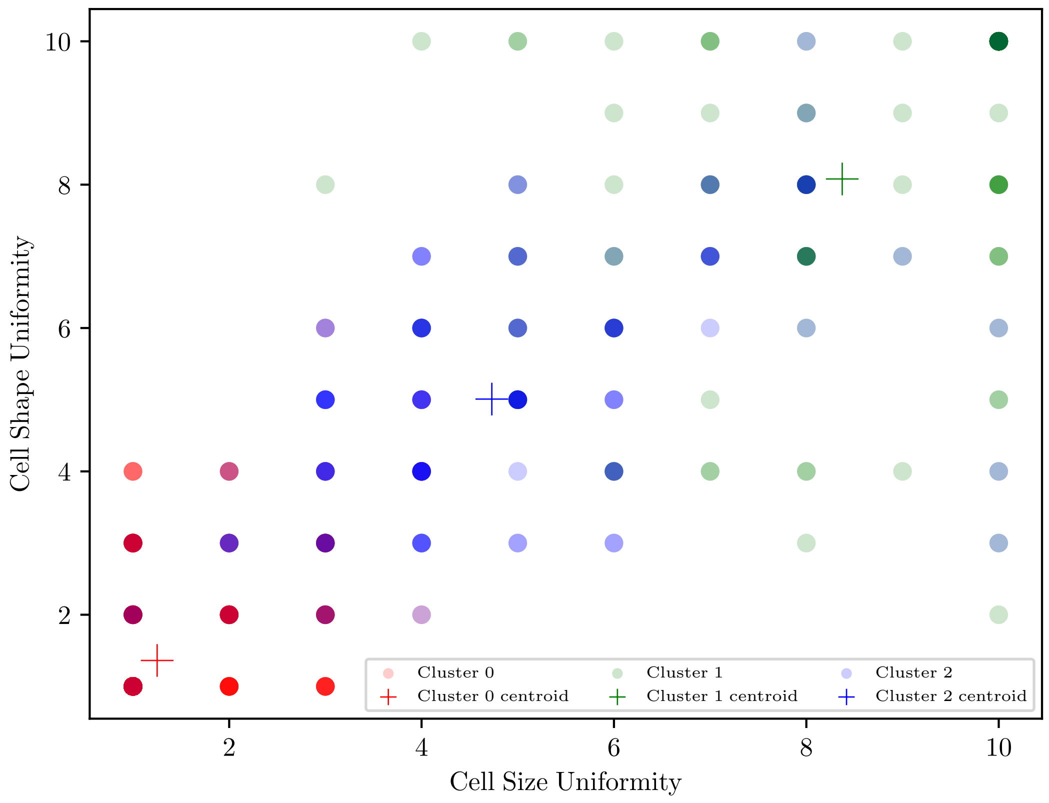
We estimate that $d_{\mathrm{VC}}(\mathrm{MLP}) > d_{\mathrm{VC}}(\text{Bayesian Classifier})$, with these values only diverging for $m \geq 300$ (albeit by not more than a multiplying constant, given their similar asymptotic behaviour).

2 Programming and critical analysis

4) (a) The produced solution yielded an ECR value of 13.5 for 2-means and 6.67 for 3-means. Hence, 3-means performed better according to this metric.

(b) The produced solution yielded a Silhouette value of 0.5968 for 2-means and 0.52454 for 3-means. Hence, 2-means performed better according to this metric.

5)



6) Assuming that the selected features are able to correctly capture the structure of the dataset, the plot above shows that the clustering solution yields poor separation, given the existence of instances with contiguous feature values that are assigned to different clusters. Furthermore, cohesion is also unsatisfactory: note the variance in cluster 1, which contains instances with feature values (4, 10) and (10, 2). This suboptimal performance can be explained by the fact that the dataset lacks circle-like concentrations of points that make the application of k-means suitable. Alternatively, the fact that k-means only considers Euclidean distances (and thus neglects feature importance) may make discerning cluster structures in this reduced input space not possible.

3 Appendix

```python
import numpy as np
import pandas as pd
from scipy.io import arff
from sklearn import cluster, metrics, feature_selection
import matplotlib.pyplot as plt
plt.rcParams["text.usetex"] = True

def load_data(filename):
    dataset = arff.loadarff(filename)
    dataset = pd.DataFrame(dataset[0])
    str_columns = [col for col in dataset.columns if dataset[col].dtype == "object"]
    dataset[str_columns] = dataset[str_columns].apply(lambda x: x.str.decode('utf8'))
    dataset = dataset.dropna()
    return dataset

# ECR is the mean over all cluster of cluster_size - label_mode_in_each_cluster
def compute_ecr(labels, outputs):

    cluster_indexes = np.unique(labels)
    ecr_measure = 0

    for index in cluster_indexes:
        cluster = outputs[labels == index]  # get instances in each cluster
        c = cluster.shape[0]
        phi = np.amax(np.unique(cluster, return_counts = True)[1]) # get count of majority label in cluster
        ecr_measure += c - phi

    return ecr_measure / cluster_indexes.shape[0]

data = load_data("../data/breast.w.arff")
inputs = data.iloc[:, :-1].to_numpy()
outputs = data.iloc[:, [-1]].to_numpy().T.flatten()
clusters_2means = cluster.KMeans(n_clusters = 2, random_state = 76).fit(inputs)
clusters_3means = cluster.KMeans(n_clusters = 3, random_state = 76).fit(inputs)

print(f"ECR k = 2: {compute_ecr(clusters_2means.labels_, outputs)}")
print(f"ECR k = 3: {compute_ecr(clusters_3means.labels_, outputs)}")
print(f"Silhouette score k = 2: {round(metrics.silhouette_score(inputs, clusters_2means.labels_), 5)}")
print(f"Silhouette score k = 3: {round(metrics.silhouette_score(inputs, clusters_3means.labels_), 5)}")

mutual_info_2best = feature_selection.SelectKBest(feature_selection.mutual_info_classif, k = 2)
inputs_top_features = mutual_info_2best.fit_transform(inputs, outputs)
extracted_features = data.iloc[:, :-1].columns[mutual_info_2best.get_support()].values
cluster_centers = clusters_3means.cluster_centers_[:, mutual_info_2best.get_support()]

fig, ax = plt.subplots()
for index, c in zip(np.unique(clusters_3means.labels_), ("r", "g", "b")):
    ax.scatter(inputs_top_features[:, 0][clusters_3means.labels_ == index], \
               inputs_top_features[:, 1][clusters_3means.labels_ == index], \
               color = c, alpha = 0.2, label = "$\\mathrm{{Cluster\;{}}}$".format(index))
    ax.scatter(cluster_centers[index, 0], cluster_centers[index, 1], \
                color = c, marker = "+", label = "$\\mathrm{{Cluster\;{}\;centroid}}$".format(index), \
                linewidths = 0.5, s = 150)

plt.legend(loc = "lower right", prop = {'size': 6}, markerscale = 0.50, ncol = 3)
ax.set_xlabel("$\mathrm{{ {} }}$".format(extracted_features[0].replace("_", "\;")))
ax.set_ylabel("$\mathrm{{ {} }}$".format(extracted_features[1].replace("_", "\;")))
plt.savefig("output/3means.pdf")
```