UNIVERSITY OF EXETER

COLLEGE OF ENGINEERING, MATHEMATICS, AND PHYSICAL SCIENCES

ECM3420/ECMM445

Learning from Data

Continuous Assessment

Date set: 14th November 2019
Hand-in date: 4th December 2019

This CA comprises 20% of the overall module assessment for the level 3 students and 10 % for the M-Level students .

This is an **individual** exercise, and your attention is drawn to the guidelines on collaboration and plagiarism in the College handbook.

See ELE for detailed submission instructions. This assignment requires you to make an electronic submission using the Electronic Coursework Submission System (`empslocal.ex.ac.uk/cgi-bin/submit/prepare`). The electronic submission should consist of a single Jupyter Notebook (.ipynb) file containing the program code you are asked to produce and their outputs;

The questions are overleaf.

What you should submit

- A Jupyter Notebook file (.ipynb) containing the code and outputs for all questions.

See next page

1 Preamble

In this coursework you will have to implement a basic clustering analysis framework from scratch (i.e., **without using the Scikit-learn** implementations) including both the algorithm and the validation functions.

1. For this coursework **you should not use** the `sklearn` or `mlxtend` packages;
2. You can use other libraries such as `pandas`, `numpy`, `scipy`, and `matplotlib` ;
3. This coursework comprises 6 (six) work pieces WP.

2 Work specification

You will have to implement the three algorithms below:

(WP_1) **[25 marks]** The k-means algorithm with the Euclidean distance

(WP_2) **[15 marks]** The Davies-Bouldin index

(WP_3) **[15 marks]** Silhouette score

Additionally, using your implementations, you will have to perform the analyses described below on the provided data files: `iris.txt`, `wine.txt` and `cluster_validation_data.txt`:

(WP_4) **[15 marks]** Perform model selection for selecting the partition order k generating a plot like the one shown in Figure 1 using your implementation of the Davies-Bouldin index and analyze the results commenting on: (1) the best number of partitions k for all the three datasets and (2) For the wine and iris data, how well did your clustering algorithm performed compared to the ground-truth known classes?

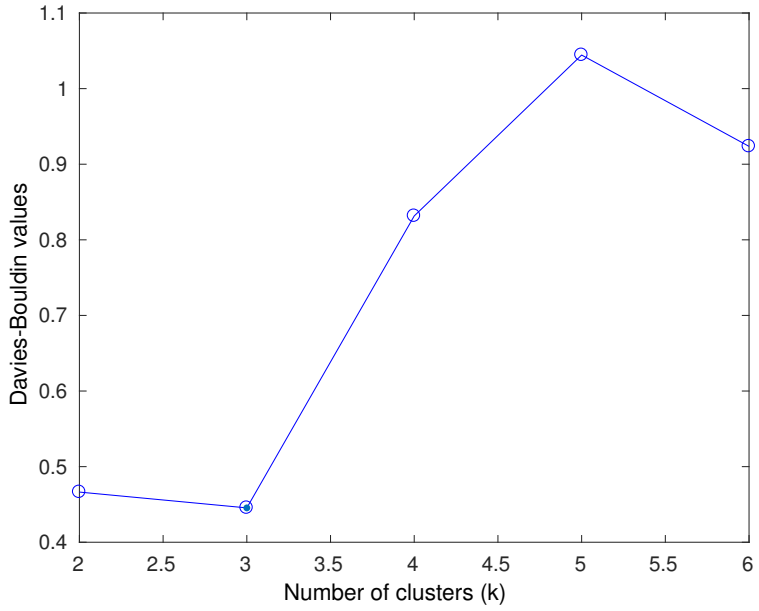

Figure 1: Davies-Bouldin index

(WP_5) **[10 marks]** Modify the function `plot_silhouette` provided in the `lfd_utils.py` file to use your `kmeans` and `silhouette_scores` functions and generate a plot similar to Figure 2 for each value of k (from 2 to `max_k`)

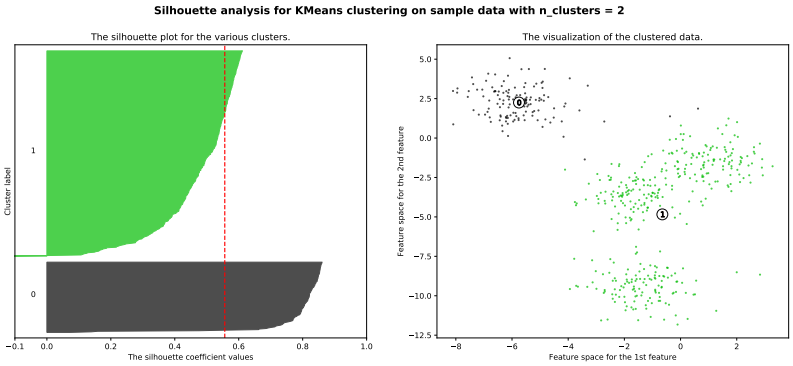

Figure 2: Example of a Silhouette plot for $k = 2$

(WP_6) **[20 marks]** Perform model selection for selecting the partition order k using the silhouette plots for k from 2 to 6 commenting on: **(1) how to interpret the silhouette scores** for each value of k taking into account the **width** of the partitions and **how they compare with the average silhouette** score, (2) **what would be the best number of partitions k** for all the three datasets and (3) For the wine and iris data, for the best k you found, **how well did your clustering algorithm performed compared to the ground-truth** known classes?

3 Functions definitions

Your implementations for the k-means and silhouette score functions should have the following signatures

- `kmeans(x,k,max_itr=100)`
  - Parameters:
    - `x`: the data do be clustered
    - `k`: the number of clusters
    - `max_itr`: the maximum number of iterations
  - Returns:
    - `cluster_labels`: the cluster membership labels for each element in the data x
- `silhouette_scores(x, cluster_labels)`
  - Parameters:
    - `x`: the data
    - `cluster_labels`: the cluster membership vector produced by the k-means algorithm.
  - Returns:
    - `scores`: a vector containing the silhouette score for each data sample in x.