Project 5: k-means

CS3753/CS5163: Data Science Summer 2023

Instructor: Dr. Mohammad Imran Chowdhury

Total Points: 65

Due: 08/08/2023 11:59 PM

In this project, I invite you to do the following:

1. Import and prepare the `iris.csv` dataset.
2. Conduct a k-means cluster analysis.
3. Visualize the clusters.

Task 1: Import and prepare the `iris.csv` dataset **(15 points)**

This has two steps. In the first step you read the iris data from **'data/iris.csv'** in the data folder then, save it in dataFrame **``df"**, and display the first 5 rows of **``df"**. The output should be as follows:

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In the second step, separates the class variable in ``y", removes the ``y" column from dataFrame ``df", standardizes ``df", and displays the first 5 rows. Note that to standardize call the StandardScaler().fit_transform() method. The output of should be as follows:

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977

Task 2: Conduct a k-means cluster analysis. **(25 points)**

You've to set up a `KMeans` object with the following parameters:

- `n_clusters`: Total number of clusters to make; set to three (03).
- `random_state`: Set to one (01) to reproduce these results.
- `init`: How to initialize the k-means centers; use `k-means++`.
- `n_init`: Number of times k-means would be run; set to ten (10).

Fits the model to the data and displays the parameters of the fitted model. The output should be as follows:

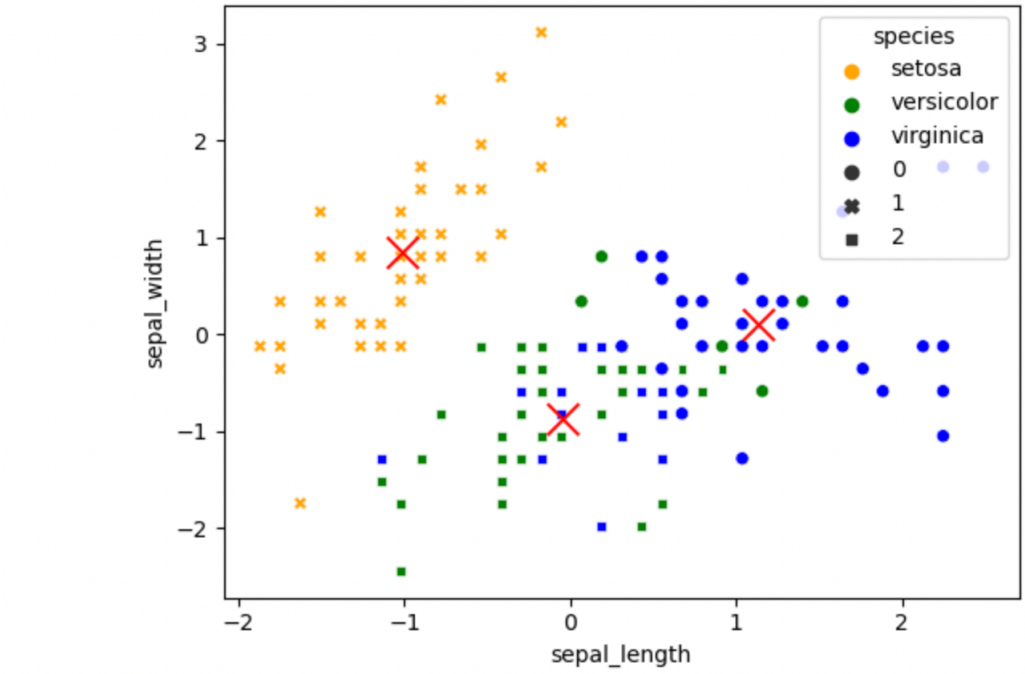
```
Out[4]: {'algorithm': 'auto',
         'copy_x': True,
         'init': 'k-means++',
         'max_iter': 300,
         'n_clusters': 3,
         'n_init': 10,
         'random_state': 1,
         'tol': 0.0001,
         'verbose': 0}
```

Task 3: Visualize the clusters. **(25 points)**

Your code should create a scatterplot of the first two features. Each point is colored according to its actual label. For comparison, each instance should be drawn with a marker according to the label found by the clustering algorithm.

The output should be as follows:

```
Out[5]: <matplotlib.collections.PathCollection at 0x7fedd02a4e80>
```



The submission grading rubric is as follows (points out of 65 total):

Project element	**Points**
Task 1	15
Task 2	25
Task 3	25

Submission Instructions: Create a compressed file (.zip or .tar.gz files are accepted) with your all source files such as .ipynb files and data files. Generally speaking to complete Task1 through Task3, you just need one .ipynb file. But it's better to submit everything as a compressed zip file. Submit the compressed zip file to the "Project Submissions" area on the Blackboard course website.

Late submission policy: As described in the syllabus, any late submission will the penalized with 10% off after each 24 hours late. For example, an assignment worth 100 points turned in 2 days late will receive a 20 point penalty. Assignments turned in 5 or more days after the due date will receive a grade of 0.