COMP8270
Programming for Artificial Intelligence

Class 9

Clustering with Python

1. Load the dataset `ClusterDs1.csv` from the Moodle page into a `DataFrame` and make a scatter plot using the `plt.scatter()` function.

2. From the scatter plot of the previous exercise, you got a good idea of how many cluster the data represents. Now create a K-Means model to find the clusters, and fit it to the data points from the previous exercise.

3. Using the model trained in the previous step, use the `predict()` method to get a list of cluster labels. Assign the result to a variable named `labels`.

4. Using the labels from the previous exercise, update the plot to show points in different colour depending on what cluster they belong.

5. Load the dataset `ClusterDs2.csv` from the Moodle page into a separate `DataFrame`. Using the model trained from the previous step, use the `predict()` method to get a list of cluster labels for the data. Assign the result to a variable named `predicted_labels`.

6. For each data point in the DataFrame of the previous step, calculate the Euclidean distance to each of the centroids of the cluster model. You can obtain the coordinates of the centroids using the `.cluster_centers_` attribute of K-Means model. The Euclidean distance is given by:

$$d(p, c) = \sqrt{\sum_{i=1}^{n} (p_i - c_i)^2}$$

where $p_i$is the value of the i-th dimension for the point p, $c_i$is the value of the i-th dimension for the centroid c, and n is the number of dimensions.

7. Using the distance values calculated in the previous step, add a new column to the DataFrame from Step 5 to store the a value representing the index of the centroid that

the point is closer to – e.g., `0` if the data point is closer to centroid 0, `1` if the data point is closer to centroid 1, and so forth.

8. Compare the predicted cluster labels (Step 5) with the cluster labels calculated in Step 7 – they should be the same.