Name: Hang Jiang
Report 4: k-means clustering

Files:
1. kmeans.py
2. glass.csv and wine_data.csv (glass1.csv is modified to fit Weka format)
3. README
4. Report4
5. wine_data.out and glass.out

In this hw4, I implemented K-means algorithm and tested the algorithm against two datasets. One dataset is the wine_data.csv. Both datasets only contain continuous numeric attributes and no missing values. The wine dataset has 3 classes and 13 other attributes. The second dataset, glass.csv, contains 10 attributes and 7 classes (in fact 6 because one class is never shown in the dataset). The last column in each dataset is the class variable. The metrics to evaluate the clustering results is purity score. We take the most popular label of each cluster as the label of the cluster and count the number of points with the same label for each cluster. The count divided by the total number of points is the purity score.

Since I used numpy and pandas libraries to help me process the data, I need the attribute values to be consistent. So I first convert the class variables which are ‘a’, ‘b’, ‘c’ … in the dataset into corresponding 1, 2, 3. At the end of the clustering, I will convert them back to ‘a’, ‘b’, ‘c’ … (Seen in the output file) Besdies, I normalized the variables using max-min normalization to make them between 0 and 1. The normalization is to avoid the distance weighing too much towards the variables with bigger values. Also, since the centroids are initialized in the beginning and there is a chance they are not well initialized, meaning some initialized centroids happen to be from the same cluster, I make the program run 5 times and choose the best result.

An interesting finding is that when I increase the number of k, the purity scores goes up for both dataset, showing that each cluster becomes more homogeneous when the number of clusters increases. But for this task, I will choose the k that Weka chose to compare against the results from Weka. From the table below, it is obvious that my result is almost the same as the Weka result. The difference in score for wine.csv is probably due to the way of centroid initialization. (Mine is slightly better since I used 5 iterations)

	My implementation	Weka
Purity score for wine.csv (k=2)	60.112%	59.887%
Purity score for glass.csv (k=2)	44.3925%	44.3925%

To explore how data preprocessing can affect the purity score. I commented out the max-min normalization in my algorithm and the purity score for both datasets lowered by around 1%. So it is okay to say data normalization is certainly helpful in improving the result.

To explore how different clustering algorithms can improve the clustering. I chose different clustering methods from Weka. From the result below, we can see that K-means and density-based clustering are both in fact better than hierarchical clustering for this dataset. I think it has something to do with the features of hierarchical clustering that its distance measurement can affect the result (like average, min and max distance measurement), which makes it unstable. By multiple iterations, K-means overcame the problem of bad initialization and achieved comparable results with density-based clustering.

	Hierarchical clustering	Density-based clustering
Purity score for wine.csv	39.548%	59.887%
Purity score for glass.csv	36.4496%	45.7944%