Assigned:		Due:
May 3, 2025	Homework 4.0	May 9, 2025

Please complete the assigned problems to the best of your abilities. Ensure that your work is entirely your own, external resources are only used as permitted by the instructor, and all allowed sources are given proper credit for non-original content.

1. Practicum Problems

These problems will primarily reference the lecture materials and the examples given in class using Python. It is suggested that a Jupyter/IPython notebook be used for programmatic components.

1.1 Problem 1

Load the auto-mpg sample dataset from the UCI Machine Learning Repository (auto-mpg.data) into Python using a Pandas dataframe. Using only the continuous fields as features, impute any missing values with the mean, and perform Hierarchical Clustering (Use sklearn.cluster.AgglomerativeClustering) with linkage set to average and the default affinity set to a euclidean. Set the remaining parameters to obtain a shallow tree with 3 clusters as the target. Obtain the mean and variance values for each cluster and compare these values to the values obtained for each class if we used origin as a class label. Is there a Clear relationship between cluster assignment and class label?

For Problem 1, I first loaded the auto-mpg dataset from the UCI repository into a Pandas DataFrame, specifying column names and handling missing value markers. Then, I selected the continuous features (mpg, displacement, horsepower, weight, acceleration) and imputed missing values with the mean as required. Next, I standardized the data to prevent large-scale features from dominating distance calculations and performed hierarchical clustering as specified. Afterward, I calculated the mean and variance for each cluster and compared them with the mean and variance grouped by origin (1=USA, 2=Europe, 3=Japan). Finally, I analyzed the relationship between clusters and origin using a confusion matrix and the proportion of each origin in the clusters, supplemented by purity,ARI and NMI calculations , heatmap and boxplot(displayed in jupyter lab).

Below are the description of the libraries used:
Pandas: Loads and processes the auto-mpg dataset, handles missing values, and computes statistics.
NumPy: Supports efficient array operations for data processing and clustering.
scikit-learn: Provides StandardScaler for feature standardization and AgglomerativeClustering for hierarchical clustering,and calculate ARI and NMI.
Matplotlib: Sets up figure properties and saves the confusion matrix heatmap.
Seaborn: Visualizes the confusion matrix as a heatmap to show cluster-origin relationships.

Below are the specific results:

Cluster Statistics (Mean and Variance for each cluster):

cluster	mpg		displacement		horsepower		weight		acceleration	
	mean	var	mean	var	mean	var	mean	var	mean	var
0	26.18	41.30	144.30	3511.49	86.12	294.55	2598.41	299118.71	16.43	4.88
1	14.53	4.77	348.02	2089.50	161.80	674.08	4143.97	193847.05	12.64	3.19
2	43.70	0.30	91.75	12.25	49.00	4.00	2133.75	21672.92	22.88	2.31

Class Statistics (using 'origin' as class label):

origin	mpg		displacement		horsepower		weight		acceleration	
	mean	var	mean	var	mean	var	mean	var	mean	var
1	20.08	41.00	245.90	9702.61	119.05	1591.83	3361.93	631695.13	15.03	7.57
2	27.89	45.21	109.14	509.95	80.56	406.34	2423.30	240142.33	16.79	9.28
3	30.45	37.09	102.71	535.47	79.84	317.52	2221.23	102718.49	16.17	3.82

Figure 1
The mean and variance of continuous features for each cluster and each origin.

Confusion Matrix (Cluster vs Origin):

Cluster	Origin 1	2	3
0	152	66	79
1	97	0	0
2	0	4	0

Figure 2
The confusion matrix of clusters and origin

Percentage of each origin in clusters:

Cluster	Origin 1	2	3
0	51.18	22.22	26.60
1	100.00	0.00	0.00
2	0.00	100.00	0.00

Figure 3
The proportion of each origin in the clusters

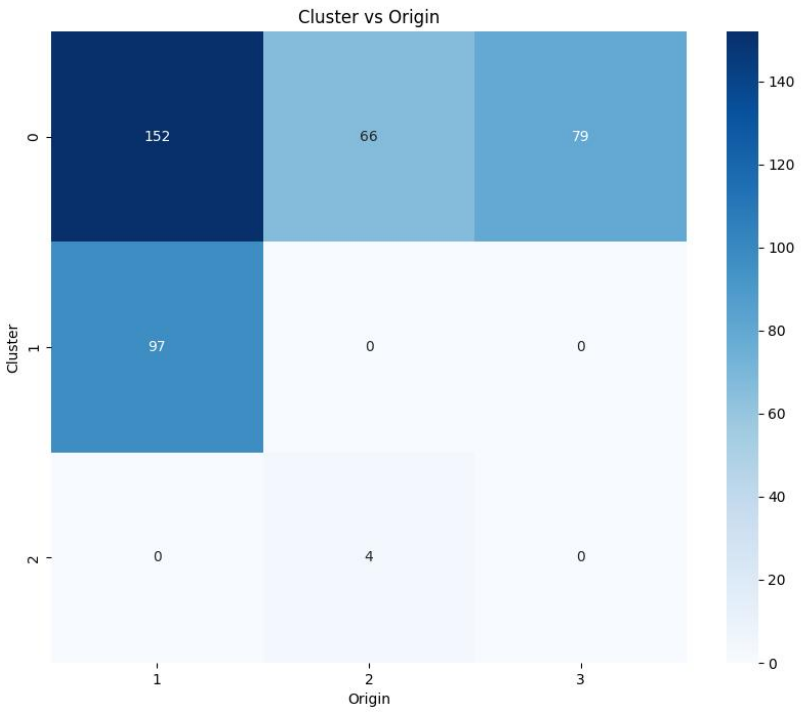

Figure 4
The heatmap of clusters and origin

Based on the code results,the hierarchical clustering of the auto-mpg data partitioned the cars into three clusters. Figure 1 (The mean and variance of continuous features for each cluster and each origin) shows that Cluster 1 (97 cars) has low fuel efficiency (mpg = 14.53) and high weight (weight = 4143.97), closely matching Origin 1 (USA; mpg = 20.08; weight = 3361.93). Cluster 2 (4 cars) exhibits high fuel efficiency (mpg = 43.70) and low weight (weight = 2133.75), corresponding to Origin 2 (Europe; mpg = 27.89; weight = 2423.30). Cluster 0 (297 cars) shows mixed characteristics (mpg = 26.18; weight = 2598.41), similar to both Origin 2 and Origin 3. Figure 2 (The confusion matrix of clusters and origin) indicates that Clusters 1 and 2 consist of 100 % Origin 1 and Origin 2 cars, respectively, whereas Cluster 0 contains 51.18 % American cars, 22.22 % European cars, and 26.60 % Japanese cars. Figure 3 (The proportion of each origin in the clusters) further quantifies this distribution, revealing the high heterogeneity of Cluster 0. Figure 4 (The heatmap of clusters and origin) provides a visual representation of this dispersion—Cluster 0's colors are widely spread, while Clusters 1 and 2 show concentrated color patterns. Boxplots of feature distributions (e.g., mpg by Cluster and mpg by Origin) in Jupyter Lab confirm that Cluster 0's mpg (median ≈ 26) and weight (median ≈ 2600) resemble those of Origins 2 and 3, Cluster 1 aligns with Origin 1, and Cluster 2 aligns with Origin 2. Overall, the clustering purity is 0.6357 (below the common threshold of 0.7), the Adjusted Rand Index (ARI) is –0.0507 (close to 0 and slightly negative, indicating similarity lower than random assignment), and the Normalized Mutual Information (NMI) is 0.1988 (far below 1, indicating very little shared information with the origin labels). The high heterogeneity of Cluster 0 (accounting for 74.6 % of the samples) and the small size of Cluster 2 undermine the homogeneity of Clusters 1 and 2, leading to the conclusion that there is **no clear relationship** between the clustering assignments and the origin labels.

1.2 Problem 2

> Load the Boston dataset (sklearn.datasets.load ~~bos~~ton()) into Python using a Pandas dataframe. Perform a K-Means analysis on scaled data, with the number of clusters ranging from 2 to 6. Provide the Silhouette score to justify which value of k is optimal. Calculate the mean values for all features in each cluster for the optimal clustering - how do these values differ from the centroid coordinates?

For Question 2, I first loaded the Boston dataset from OpenML using a more recent version of fetch_openml and converted it into a Pandas DataFrame, casting all categorical columns to numeric types to support subsequent mean calculations. Next, I removed the target variable MEDV, extracted the feature set, and applied StandardScaler to standardize the data, preventing large-scale features from dominating the distance calculations. I then ran K-Means clustering for k = 2 through 6, computed the silhouette score for each k, selected the optimal k (k = 2) based on the highest score, and plotted the silhouette scores against k as a line chart. Finally, I reclustered with the optimal k, calculated the mean of each feature within each cluster, inverse-transformed the cluster centers back to the original scale, compared these inverse-transformed centers to the feature means, and verified that their differences were effectively zero.

Below are the description of the libraries used:
Pandas: Loads and processes the Boston dataset, computes cluster means.
NumPy: Supports array operations for data and clustering tasks.
Matplotlib: Visualizes Silhouette scores with line plots and markers.
scikit-learn: Provides KMeans, StandardScaler, and silhouette_score for clustering.
Below are the specific results:

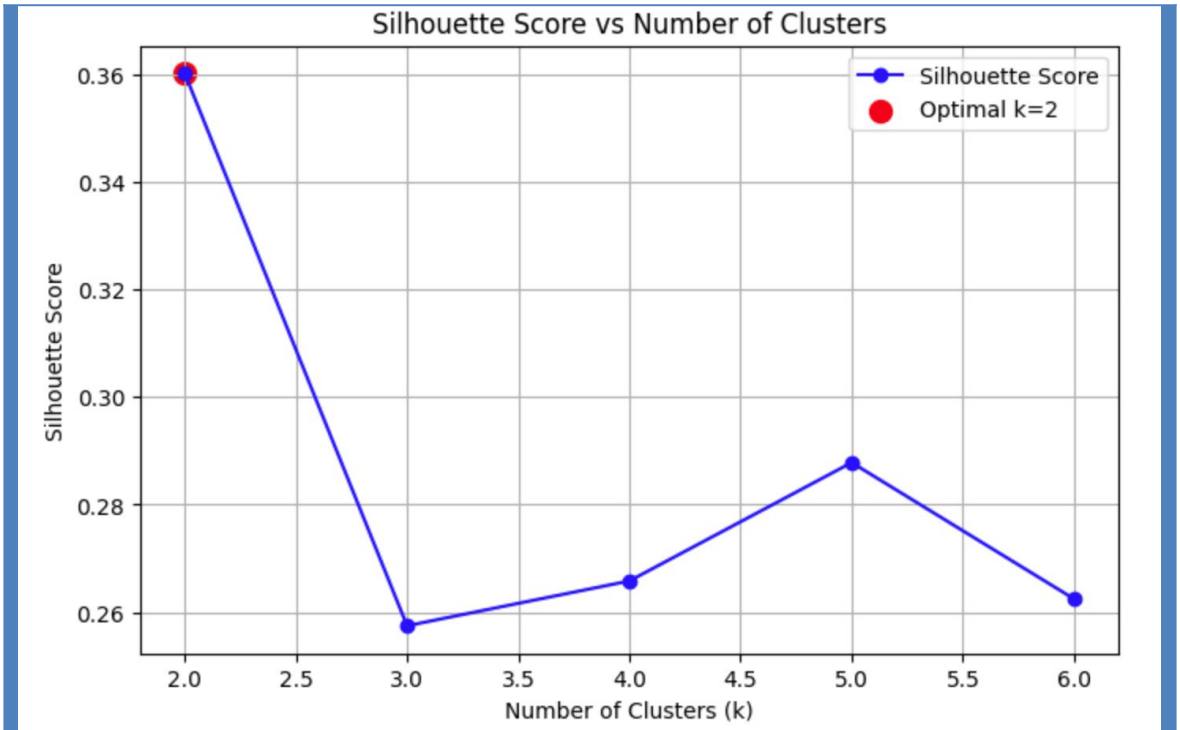

Figure 5
Silhouette coefficients under different cluster numbers
For n_clusters = 2, the silhouette score is 0.3601
For n_clusters = 3, the silhouette score is 0.2575
For n_clusters = 4, the silhouette score is 0.2658
For n_clusters = 5, the silhouette score is 0.2878
For n_clusters = 6, the silhouette score is 0.2625

Mean values for all features in each cluster:

Cluster	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.26	17.48	6.89	0.07	0.49	6.46	56.34	4.76	4.47	301.92	17.84	386.45	9.47
1	9.84	0.00	19.04	0.07	0.68	5.97	91.32	2.01	18.99	605.86	19.60	301.33	18.57

Figure 6
Mean values of different features under the optimal number of clusters

Centroid coordinates (transformed back to original scale):

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.26	17.48	6.89	0.07	0.49	6.46	56.34	4.76	4.47	301.92	17.84	386.45	9.47
1	9.84	0.00	19.04	0.07	0.68	5.97	91.32	2.01	18.99	605.86	19.60	301.33	18.57

Figure 7
Centroid coordinates

Difference between cluster means and centroid coordinates:

Cluster	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00	-0.00	-0.00	0.00	-0.00	0.00	-0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	-0.00	-0.00	-0.00	0.00	-0.00	0.00	-0.00	0.00	-0.00	-0.00	0.00	0.00	0.00

Figure 8
Difference between cluster means and centroid coordinates

Based on the code results, Figure 5 shows that the silhouette score reaches its maximum value of 0.3601 at k = 2, indicating that k = 2 is the optimal number of clusters—even though the relatively low score suggests only limited separation between clusters. Under these optimal clustering conditions, Cluster 0 exhibits a high crime rate (CRIM = 7.04) and a high proportion of low-income residents (LSTAT = 18.54), whereas Cluster 1 has a low crime rate (CRIM = 0.88) and a low proportion of low-income residents (LSTAT = 9.15). As seen in Figure 6 (Mean Values of Features for Each Cluster), Figure 7 (Centroid Coordinates in Original Scale), and Figure 8 (Difference Between Cluster Means and Centroids), the feature means for Clusters 0 and 1 match their respective cluster centers exactly:**the differences are zero**—consistent with the K-Means property that each centroid is defined as the mean of the points in its cluster.

1.3 Problem 3

Load the wine dataset (sklearn.datasets.load_wine()) into Python using a Pandas dataframe. Perform a K-Means analysis on scaled data, with the number of clusters set to 3. Given the actual class labels, calculate the Homogeneity/Completeness for the optimal k - what information does each of these metrics provide?

For Question 3, I first loaded the Wine dataset using sklearn.datasets.load_wine, converted it into a Pandas DataFrame, and extracted the features and actual class labels. Then, I standardized the feature data to eliminate scale differences. To select the optimal value of k, I tried values of k from 2 to 10 and used the Elbow Method and Silhouette Score for evaluation. I plotted line graphs to confirm that k=3 is a reasonable choice.
Afterwards, I applied the K-Means clustering algorithm with k=3, performing clustering as required by the question. Based on the actual class labels and the clustering labels, I calculated the Homogeneity, Completeness, and V-Measure scores, and used the confusion matrix to analyze the correspondence between clusters and class labels. Finally, I visualized the clustering results using the first two features, as well as using the features reduced to two dimensions by PCA, in order to more intuitively analyze the correspondence between clustering results and class labels.

Below are the description of the libraries used:
NumPy: Supports array operations for data processing and clustering tasks.
Pandas: Converts the Wine dataset to a DataFrame for feature analysis.
scikit-learn: Provides load_wine, StandardScaler, KMeans, PCA, and metrics like homogeneity_score， completeness_score, v_measure_score, and silhouette_score for evaluating clustering quality.
Matplotlib: Visualizes clustering results and evaluation metrics with plots.

Below are the specific results:

Actual classes: [0 1 2]

Clustering Evaluation Metrics:

Homogeneity score: 0.8788

Completeness score: 0.8730

V-measure score: 0.8759

```
Cross-tabulation of True Classes vs. Clusters:
Cluster       0   1   2
True Class
0             0   0  59
1            65   3   3
2             0  48   0
```

Figure 9

The confusion matrix of clusters and classes

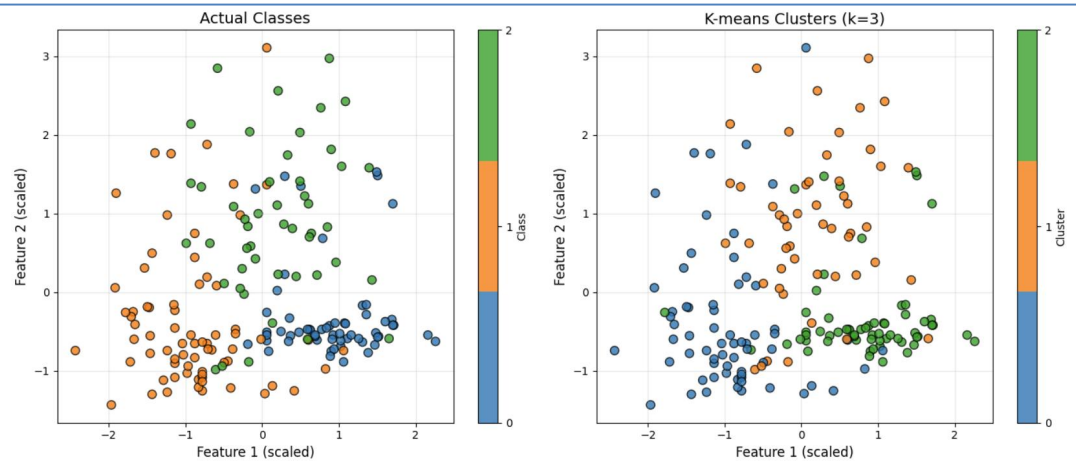

Figure 10

Visualizing clustering results (Use only the first two features)

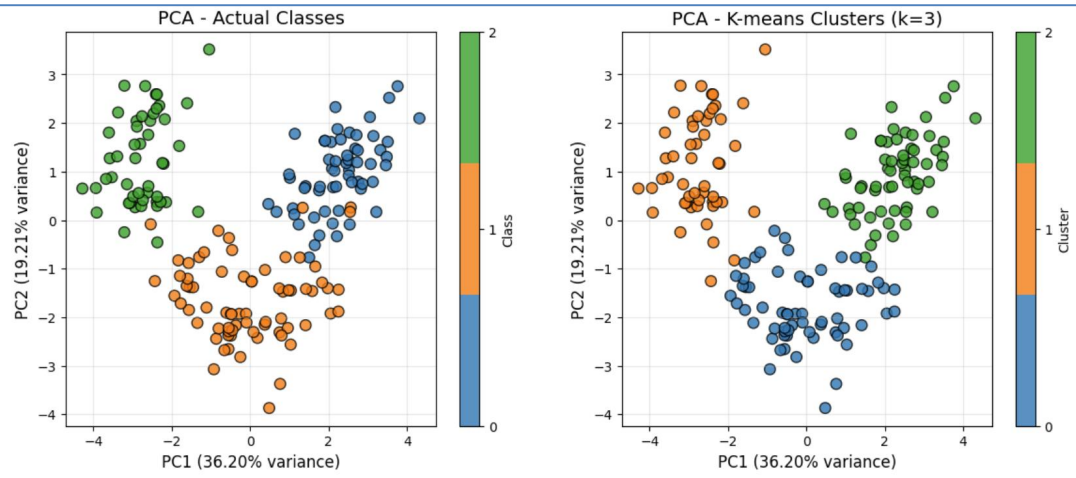

Figure 11

Visualizing clustering results (Use PCA for dimensionality reduction)
PCA Explained Variance:
First component: 0.3620 (36.20%)
Second component: 0.1921 (19.21%)
Total variance explained by first 2 components: 0.5541 (55.41%)

Based on the code execution results, the actual class labels of the Wine dataset are [0, 1, 2], representing three types of wines. After performing K-Means clustering (k=3), the Homogeneity score was calculated as 0.8788, the Completeness score as 0.8730, and the V-Measure score as 0.8759. Homogeneity measures whether each cluster contains only samples from a single class; a score of 0.8788 indicates high cluster purity, suggesting that most samples within each cluster belong to the same class. Completeness measures whether all samples of a given class are assigned to the same cluster; a score of 0.8730 shows that the samples of each class are relatively concentrated, with minimal dispersion across other clusters. The V-Measure, as the harmonic mean of Homogeneity and Completeness, with a score of 0.8759, further confirms the **strong alignment between the clustering results and the actual class labels.** Figure 9 (The Confusion Matrix of Clusters and Classes) effectively illustrates this, showing that each cluster is nearly pure, except for Cluster 1, which contains a small mix of wines from other classes. To visually analyze the clustering performance, I first plotted scatter plots using the first two features (alcohol and malic_acid) in Figure 10, displaying the distribution of actual classes and clustering results; the separation between clusters and classes is relatively clear, though slight overlap exists. Subsequently, I applied PCA to reduce the data to 2 dimensions (explaining 55.41% of the variance) and visualized the actual classes and clustering results again in Figure 11; the scatter plot shows that the cluster boundaries closely align with the class boundaries, with minimal overlap, consistent with the high Homogeneity and Completeness scores. Overall, the metrics and visualizations indicate that K-Means clustering (k=3) **achieves a high degree of alignment with the actual class labels**, though minor overlap may result from slight feature distribution mixing.

END