1 Solution to Problem1

1.1 Dataset Description

The Auto MPG dataset is a classic multivariate dataset from the UCI Machine Learning Repository. Each record corresponds to a specific car model, described by a set of attributes related to its engine and performance characteristics.It contains 398 instances with 9 attributes. The attributes in the dataset are as follows:

Table 1: Auto MPG Dataset Attribute Description

Attribute	**Type**	**Description**
`mpg`	Continuous	Miles per gallon
`cylinders`	Discrete	Number of cylinders
`displacement`	Continuous	Engine displacement (cubic inches)
`horsepower`	Continuous	Engine horsepower
`weight`	Continuous	Vehicle weight in pounds
`acceleration`	Continuous	Time to accelerate from 0 to 60 mph (in seconds)
`model year`	Discrete	Year of model release (e.g., 70 = 1970)
`origin`	Categorical	Country of origin (1 = USA, 2 = Europe, 3 = Japan)
`car name`	String	Car name (e.g., "chevrolet impala")

1.2 Data Preprocessing

We first load `auto-mpg.data` from the UCI Machine Learning Repository into Python using a Pandas dataframe. Then, we retain only the continuous numerical features and impute any missing values using the mean of each column. The variable `origin` is used as the class label. A pairplot of the data with `origin` as the hue is shown in Figure 1a.

1.3 Hierarchical Clustering

We standardize the numerical features and apply hierarchical clustering using `AgglomerativeClustering` from `sklearn.cluster`, with the number of clusters set to 3, linkage set to `average`, and the distance metric as `euclidean`. The resulting cluster labels are visualized using a pairplot in Figure 1b.

Comparing Figures 1a and 1b, we observe that the cluster distributions differ significantly from the origin class, suggesting no clear relationship between the cluster assignment and the class label.

(a) Pairplot of Auto MPG Dataset by Origin Class (b) Pairplot of Auto MPG Dataset by Cluster

Figure 1: Pairplot of Auto MPG Dataset

We used the `linkage` and `dendrogram` functions from the `scipy.cluster.hierarchy` module to generate a dendrogram. The distance threshold for forming three clusters was computed using `linked[-3, 2]`, and we set the `color_threshold` parameter to visually distinguish the clusters. The resulting dendrogram with the threshold line dividing the data into three clusters is shown in Figure 2.

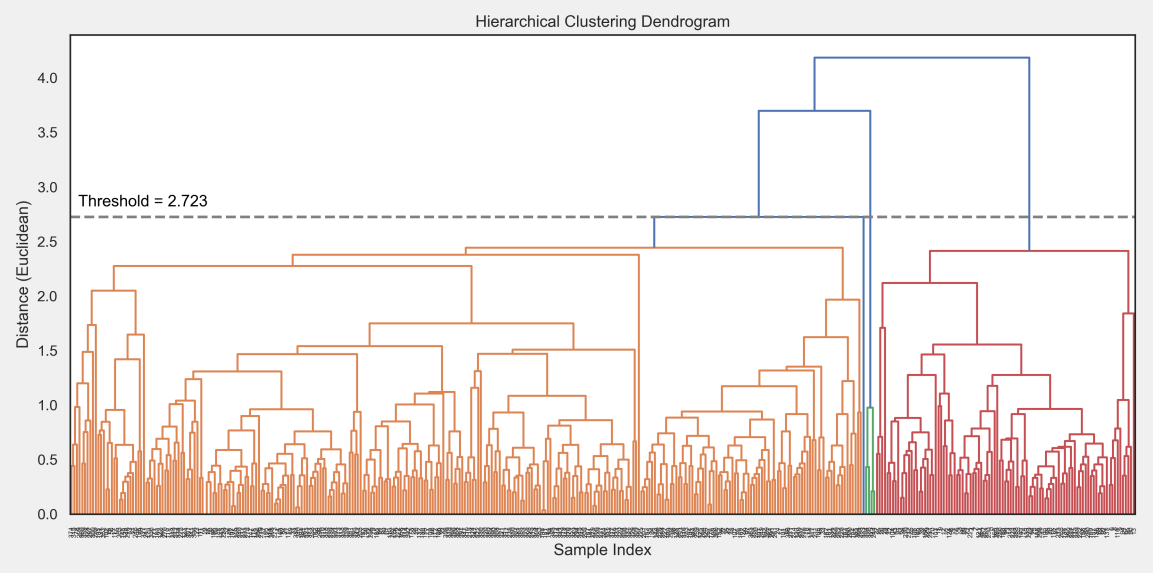

Figure 2: Hierarchical Clustering Dendrogram

From Figure 2, we observed that displaying all samples leads to a highly dense diagram with overlapping lines. To address this, we applied `truncate_mode='lastp'` to prune the leaf nodes and showed only the last 100 merged clusters. The pruned dendrogram is presented in Figure 3.

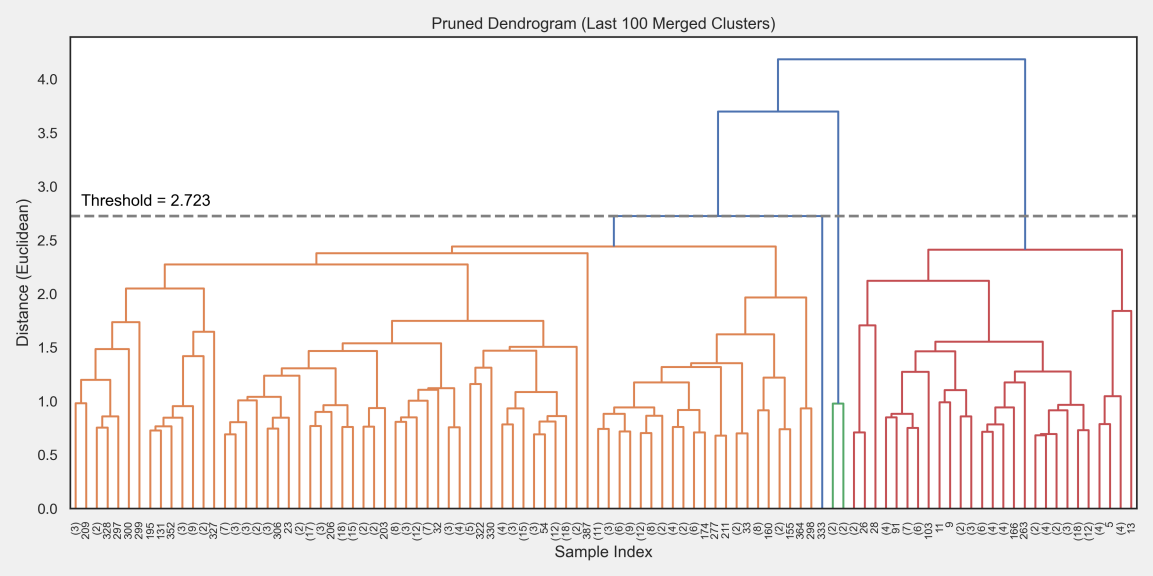

Figure 3: Pruned Dendrogram (Last 100 Merged Clusters)

As shown in Figure 3, the colors orange, green, and red represent the three clusters. The distance threshold used to divide the data into these three clusters is 2.723.

1.4 Result and Analysis

1.4.1 Class Statistics

We calculate the mean and variance of each cluster, as shown in Table 2.

Table 2: Cluster Statistics

Feature	**Mpg**		**Displacement**		**Horsepower**		**Weight**		**Acceleration**	
Cluster	mean	var	mean	var	mean	var	mean	var	mean	var
0	26.18	41.3	144.3	3511.49	86.12	294.55	2598.41	299118.71	16.43	4.88
1	14.53	4.77	348.02	2089.5	161.8	674.08	4143.97	193847.05	12.64	3.19
2	43.7	0.3	91.75	12.25	49	4	2133.75	21672.92	22.88	2.31

Similarly, we compute the mean and variance of the original `origin` classes, shown in Table 3.

Table 3: Origin Class Statistics

Feature	**Mpg**		**Displacement**		**Horsepower**		**Weight**		**Acceleration**	
Cluster	mean	var	mean	var	mean	var	mean	var	mean	var
0	20.09	41	245.9	9702.61	119.05	1591.83	3361.93	631695.13	15.03	7.57
1	27.89	45.21	109.14	509.95	80.56	406.34	2423.3	240142.33	16.79	9.28
2	30.45	37.09	102.71	535.47	79.84	317.52	2221.23	102718.49	16.17	3.82

We also plot the boxplots for both the clusters and the origin class in Figures 4 and 5.

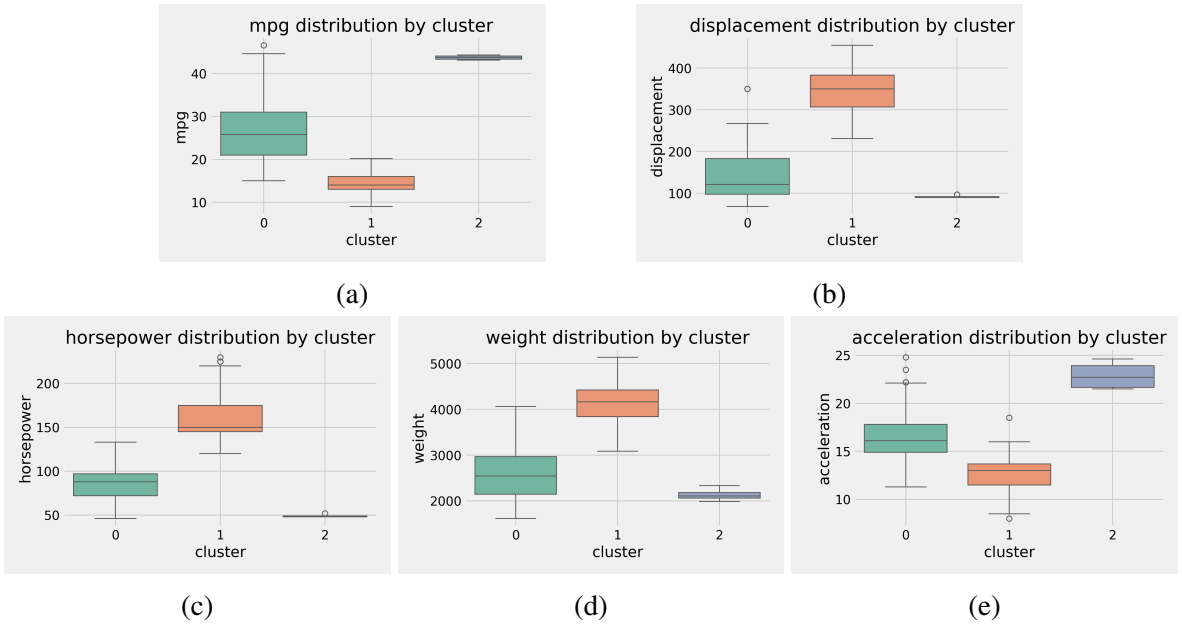

Figure 4: Boxplot by Cluster

By comparing the distributions, we find that the data distributions differ between clusters and the origin class, suggesting no clear relationship.

1.4.2 External Clustering Metrics

1. **Adjusted Rand Index (ARI)**

   The Adjusted Rand Index is a measure of the similarity between two data clusterings. It ranges from -1 to 1, where 1 indicates perfect agreement, 0 represents random labeling, and negative values indicate worse-than-random results.

$$\text{ARI} = \frac{\text{RI} - \mathbb{E}[\text{RI}]}{\max(\text{RI}) - \mathbb{E}[\text{RI}]}$$

   where RI is the original Rand Index and $\mathbb{E}[\text{RI}]$ is expectation under random conditions.

$$\text{RI} = \frac{a+b}{a+b+c+d}$$

   where a denotes the two samples are in the same cluster and in the same true class, b denotes the two samples are in different clusters and in different true classes, c denotes the two samples are in the same cluster, but not in the same true class, d denotes the two samples are in different clusters, but in the same true class.

   Using `adjusted_rand_score()`, we compute ARI = -0.051, indicating that the clustering structure is not aligned with the actual class labels.

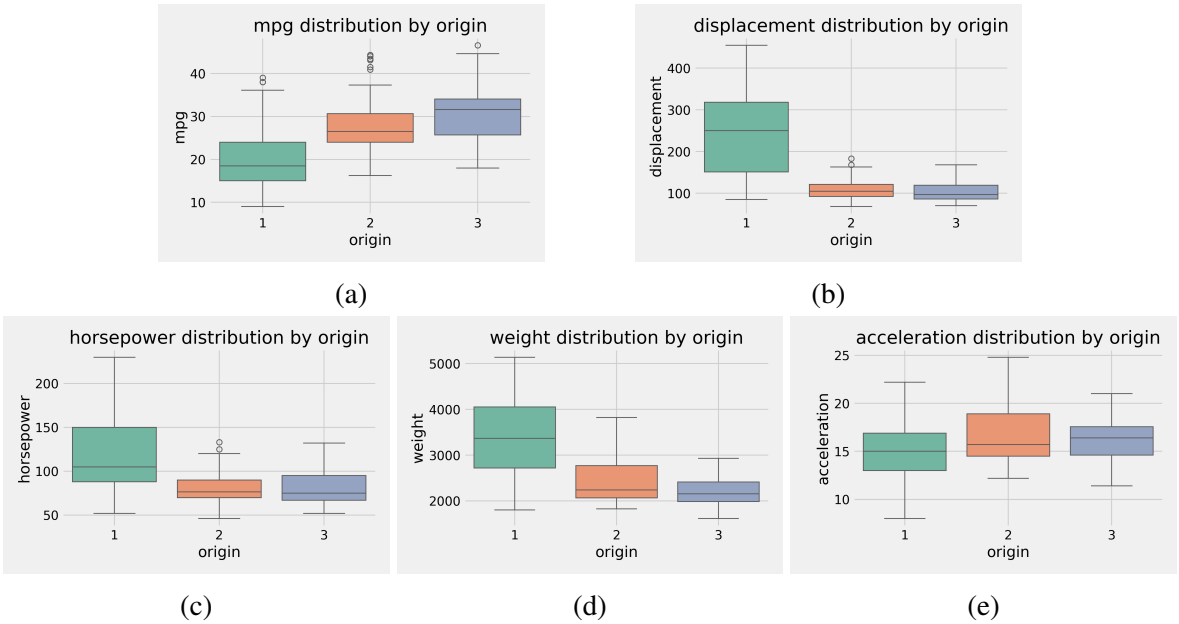

Figure 5: Boxplot by Origin

2. **Normalized Mutual Information (NMI)**

   Normalized Mutual Information (NMI) evaluates the mutual dependence between clustering labels and ground truth labels. It ranges from 0 to 1, with 1 indicating perfect correlation.

$$\mathrm{NMI}(U,V) = \frac{2 \cdot I(U;V)}{H(U)+H(V)}$$

   where $I(U;V)$ is the mutual information between cluster U and ground truth V, and $H(\cdot)$ denotes entropy.

   Using `normalized_mutual_info_score()`, we obtain NMI = 0.199, indicating weak information shared between clustering results and actual class labels.

3. **V-measure**

   V-measure is the harmonic mean of Homogeneity and Completeness.

   Homogeneity: Measures whether each cluster contains only members of a single class.

$$\mathrm{Homogeneity} = 1 - \frac{H(C|K)}{H(C)}$$

   where C is clustering results, K is real label and $H(\cdot)$ denotes entropy.

   Completeness: Measures whether all members of a given class are assigned to the same cluster.

$$\mathrm{Completeness} = 1 - \frac{H(K|C)}{H(K)}$$

   V-measure: Harmonic mean of Homogeneity and Completeness.

$$V = 2 \cdot \frac{\text{Homogeneity} \cdot \text{Completeness}}{\text{Homogeneity} + \text{Completeness}}$$

Using `homogeneity_score()`, `completeness_score()`, and `v_measure_score()`, we compute:

- Homogeneity = 0.165
- Completeness = 0.250
- V-measure = 0.199

These results confirm that clusters contain mixed classes and that samples from the same class are scattered across different clusters. Hence, there is no clear relationship between the cluster assignment and the class label.

1.4.3 Dimensionality Reduction via PCA

To further explore the relationship between the cluster assignment and the origin class label, we create a crosstab (Table 4) showing the contingency between clusters and origin.

Table 4: Cross-tab between Clusters and Origin

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

We also apply Principal Component Analysis (PCA) to reduce the data to 2D for visualization. Figure 6 shows a scatter plot with points colored by cluster and shaped by origin class.

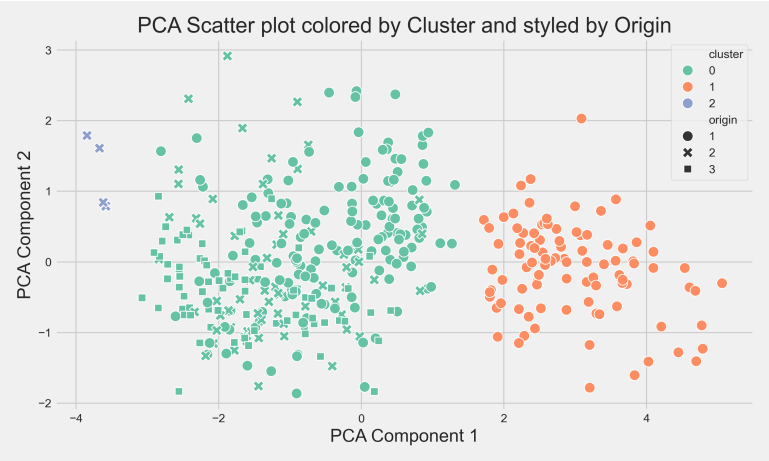

Figure 6: PCA Scatter plot colored by Cluster and styled by Origin

The overlap and dispersion of shapes and colors further demonstrate that the cluster assignment does not clearly correspond to the class label.

1.5 Conclusion

Combining the insights from class statistics, external clustering metrics (ARI, NMI, Homogeneity, Completeness, V-measure), and PCA visualization, we conclude that there is no clear relationship between the cluster assignment and the actual origin class label.

2 Solution to Problem2

2.1 Dataset Description

3 Dataset Description

The Boston Housing dataset is a classic regression dataset collected from the 1970 U.S. Census, widely used for predicting housing prices in the Boston suburbs. It contains 506 instances with 13 attributes. The attributes are described in Table 5.

Table 5: Boston Housing Dataset Attribute Description

Attribute	Type	Description
CRIM	Numerical	Per capita crime rate by town
ZN	Numerical	Proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS	Numerical	Proportion of non-retail business acres per town
CHAS	Categorical	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
NOX	Numerical	Nitric oxides concentration (parts per 10 million)
RM	Numerical	Average number of rooms per dwelling
AGE	Numerical	Proportion of owner-occupied units built prior to 1940
DIS	Numerical	Weighted distances to five Boston employment centers
RAD	Ordinal	Index of accessibility to radial highways
TAX	Numerical	Full-value property tax rate per $10,000
PTRATIO	Numerical	Pupilteacher ratio by town
B	Numerical	$1000(B_k - 0.63)^2$, where B_k is the proportion of Black residents
LSTAT	Numerical	Percentage of lower status of the population

3.1 Data Preprocessing

We first load the Boston dataset using `sklearn.datasets.load_boston()` into Python with a Pandas DataFrame. After inspection, we found that there are no missing values in the dataset.

3.2 K-Means Clustering

We begin by standardizing the dataset and then apply the `KMeans()` algorithm for clustering. The `silhouette_score()` function is used to compute the Silhouette score for the number of clusters k

ranging from 2 to 6. The results are shown in Table 6, and the relationship between the silhouette score and the number of clusters k is illustrated in Figure 7.

Table 6: Silhouette Score for Different Numbers of Clusters

Number of Custers (k)	**Silhouette Score**
2	0.3601
3	0.2448
4	0.2275
5	0.2389
6	0.2291

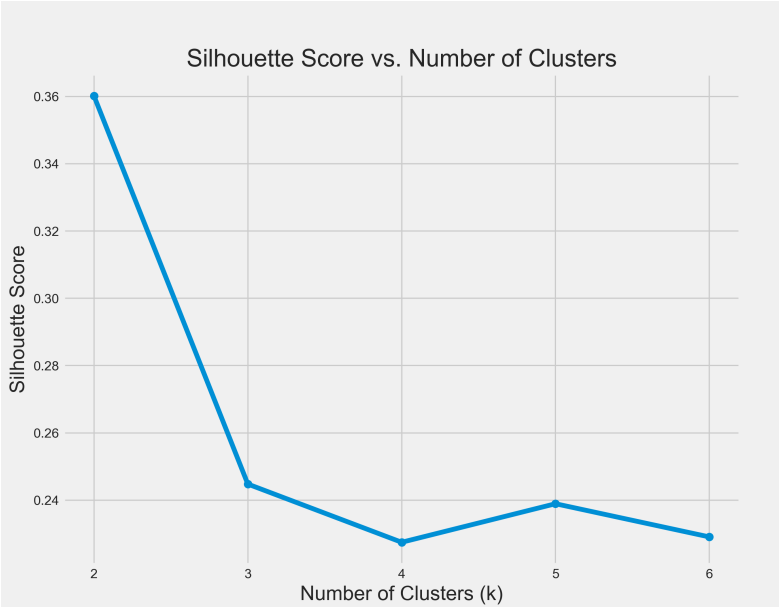

Figure 7: Silhouette Score vs. Number of Clusters

We observe that the silhouette score reaches a maximum value of 0.3601 when $k = 2$, indicating that 2 is the optimal number of clusters. A K-Means analysis is then performed with $k = 2$, and a pairplot of selected features is shown in Figure 8.

Furthermore, we apply Principal Component Analysis (PCA) to reduce the feature dimensions to two. The PCA-transformed data is added to the DataFrame. The cluster centroids from the original feature space are projected into the PCA space. The resulting K-Means Clustering visualization is shown in Figure 9.

As shown in Figure 9, the data is effectively clustered into two groups, and the clustering appears reasonably well-separated in the PCA-transformed space.

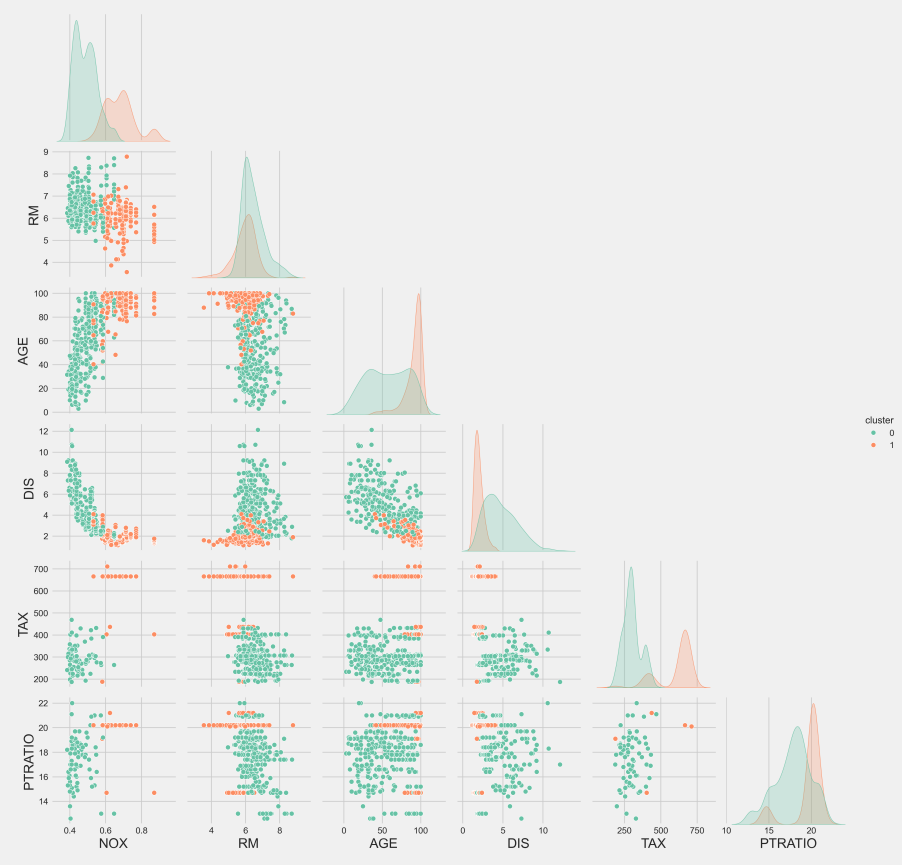

Figure 8: Pairplot of Selected Features of Boston Dataset by Cluster

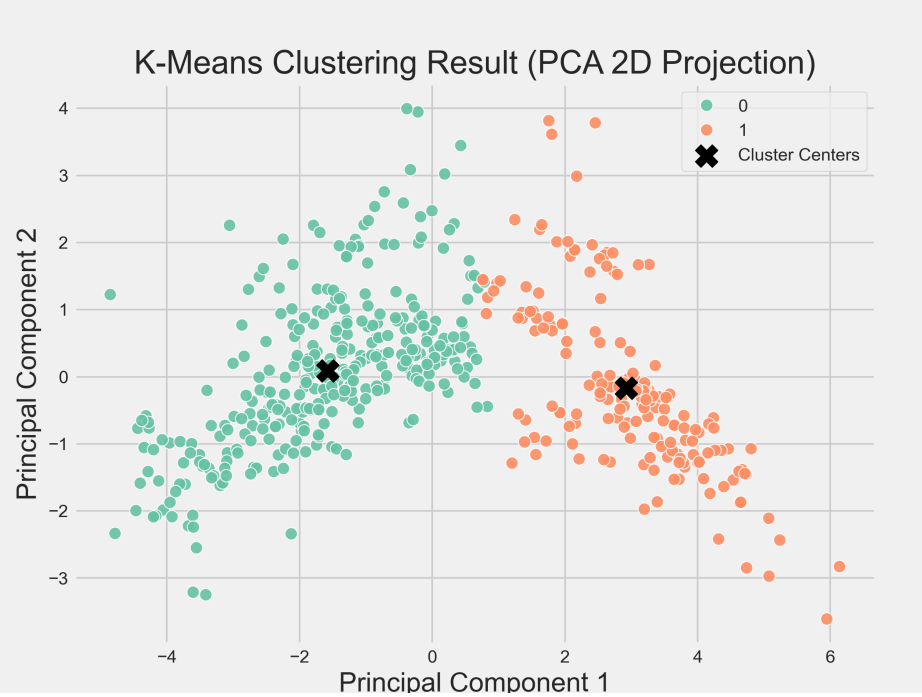

Figure 9: K-Means Clustering Result with PCA

3.3 Result and Analysis

3.3.1 Calculation of Cluster Means and Centroid Coordinates

We first compute the mean value of each feature for each cluster in the original data scale. Then, we use the `.cluster_centers_` attribute to obtain the centroids in the standardized space. The centroids are then inverse-transformed using `.inverse_transform()` to map them back to the original data scale. The Cluster Means and Centroid Coordinates are presented in Table 7 and Table 8, respectively.

Table 7: Cluster Means

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	19.04	0.07	0.68	5.97	91.32	2.01	18.99	605.86	19.6	301.33	18.57

Table 8: Centroid Coordinates

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	19.04	0.07	0.68	5.97	91.32	2.01	18.99	605.86	19.6	301.33	18.57

From the comparison of Table 7 and Table 8, it is evident that the Cluster Means and Centroid Coordinates are exactly the same.

3.3.2 Reason Analysis

The core principle of K-Means clustering is to minimize the within-cluster sum of squares (WCSS), which is the squared distance between data points and their respective cluster centroids. During each iteration of the algorithm, the centroid of a cluster is updated as the mean of the data points assigned to that cluster. Therefore,

$$\text{Centroid}_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i = \text{Mean of Cluster } k$$

where $|C_k|$ is the number of samples in cluster C_k, x_i is the i-th sample, and Centroid_k is the centroid of cluster C_k.

Hence, in theory and practice, the cluster means and the centroids from K-Means are exactly identical.

4 Solution to Problem3

4.1 Dataset Description

The Wine dataset is often used for the testing and demonstration of classification, clustering and dimensionality reduction algorithms. It contains 178 instances with 14 attributes.

Table 9: Wine Dataset Attributes Description

Feature Name	Type	Description
Alcohol	Continuous	Alcohol content
Malic acid	Continuous	Malic acid concentration
Ash	Continuous	Ash content
Alcalinity of ash	Continuous	Alkalinity of ash (alkaline property of certain minerals in ash)
Magnesium	Continuous	Magnesium content
Total phenols	Continuous	Total phenolic compounds
Flavanoids	Continuous	Flavonoids (a type of antioxidant polyphenol)
Nonflavanoid phenols	Continuous	Non-flavonoid phenolic compounds
Proanthocyanins	Continuous	Proanthocyanins (natural antioxidants)
Color intensity	Continuous	Color intensity of the wine
Hue	Continuous	Hue (shade or tint of the wine)
OD280/OD315 of diluted wines	Continuous	Optical density ratio at 280nm and 315nm (related to polyphenol concentration)
Proline	Continuous	Proline content (an amino acid influencing wine aroma)
target	Categorical	Class of wine(0 = Cultivar 1, 1 = Cultivar 2, 2 = Cultivar 3)

4.2 Data Preprocessing

First, we load the wine dataset (`sklearn.datasets.load_wine()`) into Python using a Pandas dataframe. Upon inspection, we found no missing values. Taking the `target` as the class label, we plotted a pairplot of selected features, as shown in Figure 10.

Figure 10: Pairplot of Selected Features of Wine Dataset by Target

4.3 K-Means Clustering

We first standardized the data, then performed clustering using `KMeans()`. We computed the Silhouette score for each number of clusters k from 2 to 6 using `silhouette_score()`, as shown in Table 10. The corresponding plot of silhouette scores versus k is shown in Figure 11.

Table 10: Silhouette Score for Different Numbers of Clusters

Number of Custers k	Silhouette Score
2	0.2855
3	0.3715
4	0.3008
5	0.2485
6	0.1627

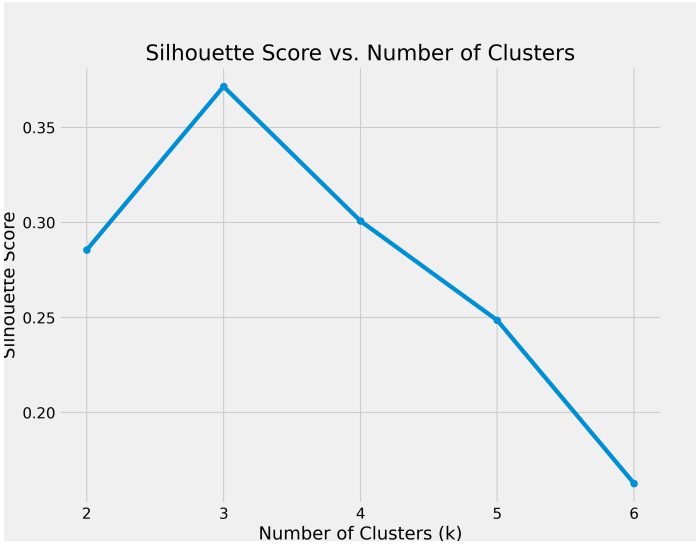

Figure 11: Silhouette Score vs. Number of Clusters

We found that the maximum silhouette score of 0.3715 occurs when $k = 3$, which is also the number of clusters specified by the task. Therefore, we performed K-Means clustering with $k = 3$, and generated a pairplot of selected features using the cluster assignment as the label, as shown in Figure 12.

To more intuitively illustrate the relationship between the clusters and the original class labels, we applied `PCA()` to reduce the data to two dimensions. We then plotted a scatter plot with cluster assignment represented by color and the `target` class by shape, as shown in Figure 13.

From Figure 13, we can observe that `cluster 0` corresponds well with `target 1`, `cluster 1` with `target 2`, and `cluster 2` with `target 0`. This indicates that the cluster assignments have a clear correspondence to the class labels, suggesting a strong relationship between the clustering structure and the true labels.

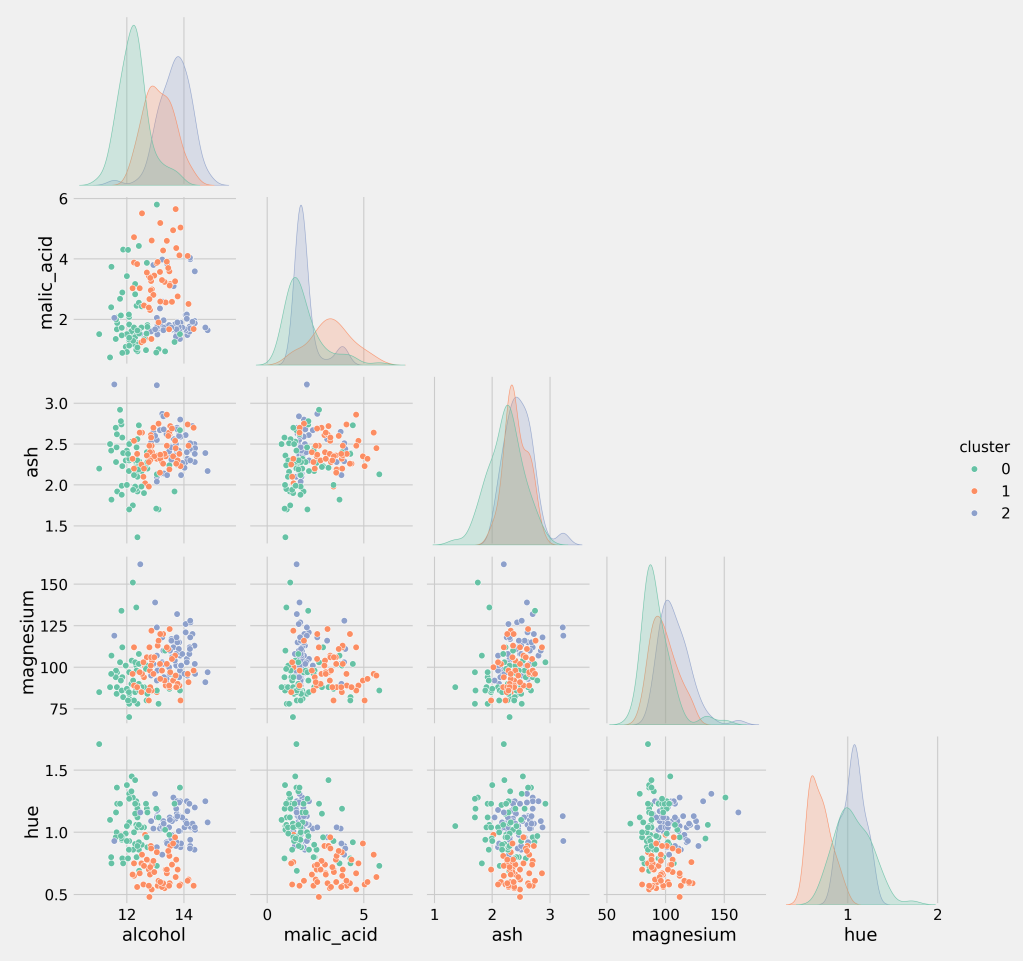

Figure 12: Pairplot of Selected Features of Wine Dataset by Cluster

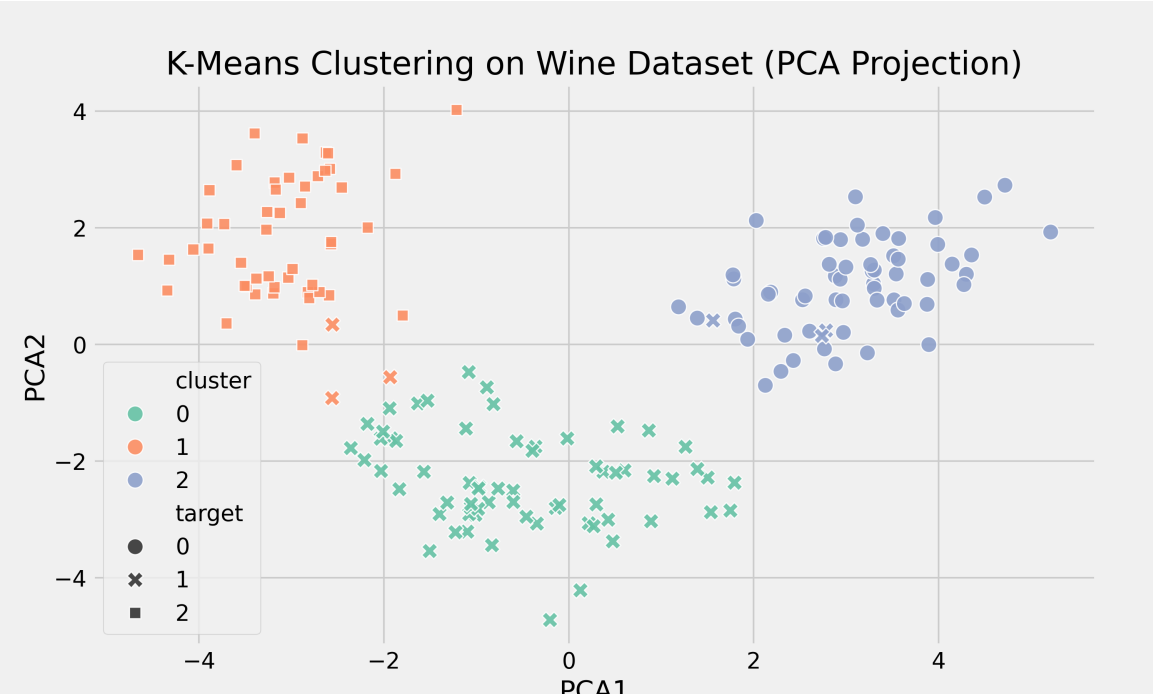

Figure 13: K-Means Clustering Result with PCA

4.4 Result and Analysis

4.4.1 Homogeneity

Homogeneity measures whether each cluster contains only members of a single class. Its value ranges from 0 to 1, where 1 indicates that all samples in a cluster belong to the same class (maximum purity), and 0 indicates a complete mix of multiple classes within clusters.

$$\text{Homogeneity} = 1 - \frac{H(C|K)}{H(C)}$$

where C is clustering results, K is real label and $H(\cdot)$ denotes entropy.

Using `homogeneity_score()`, we obtained a homogeneity score of 0.8788, indicating high internal consistency within clusters. This suggests that K-Means clustering rarely mixes different class samples into the same cluster.

4.4.2 Completeness

Completeness measures whether all members of a given class are assigned to the same cluster. Its value also ranges from 0 to 1, where 1 means all samples of a class are contained in one cluster (no fragmentation), and 0 indicates that the class is split across many clusters.

$$\text{Completeness} = 1 - \frac{H(K|C)}{H(K)}$$

Using `completeness_score()`, we obtained a completeness score of 0.8730, suggesting that samples from the same class were not scattered across multiple clusters.

4.4.3 V-measure

V-measure is the harmonic mean of homogeneity and completeness. It ranges from 0 to 1, with values closer to 1 indicating that the clustering closely reflects the true label structure, while values near 0 indicate little or no relation.

$$V = 2 \cdot \frac{\text{Homogeneity} \cdot \text{Completeness}}{\text{Homogeneity} + \text{Completeness}}$$

Using `v_measure_score()`, we obtained a V-measure of 0.8759, indicating that the overall clustering quality is excellent and that there is a strong correspondence between the cluster structure and the actual labels.

4.4.4 Conclusion

These results demonstrate that the K-Means clustering algorithm performs very well on the Wine dataset. The clustering structure is highly consistent with the true classes. Both the homogeneity and completeness scores are close to 0.88, indicating that the clustering effectively recovers the underlying class structure.