Machine Learning #19

▼ 1. A set of one-dimensional data points is given to you: 5, 10, 15, 20, 25, 30, 35. Assume that k = 2 and that the first set of random centroid is 15, 32, and that the second set is 12, 30.

a) Using the k-means method, create two clusters for each set of centroid described above.

b) For each set of centroid values, calculate the SSE.

▼ 2. Describe how the Market Basket Research makes use of association analysis concepts.

Market Basket Analysis (MBA) is a technique used in retail to identify the relationships between products that customers purchase. It utilizes association analysis concepts, which is a technique used to discover co-occurring items in large datasets. ***In MBA, the goal is to uncover associations between products that customers tend to purchase together and use these associations to increase sales.***

MBA begins by collecting transactional data from point of sale (POS) systems, which records the purchases made by each customer. The data is then processed to identify the frequency with which products are purchased together. This frequency is expressed in terms of two measures, Support and Confidence.

Support is the percentage of transactions in which the two items occur together. It measures the frequency with which the items are purchased together relative to all transactions.

Confidence is the percentage of transactions in which the second item is purchased given that the first item has been purchased. It measures the strength of the relationship between the items.

These measures are used to generate rules that define the associations between items. For example, a rule might be "If a customer purchases bread, they are likely to purchase butter as well, with a support of 30% and a confidence of 70%."

MBA can be used to generate insights into customer behavior that can be used to increase sales. Retailers can use these insights to make decisions about product placement, promotions, and pricing strategies. For example, if the analysis reveals that customers who purchase coffee tend to also purchase milk,

the retailer may place these items near each other in the store to encourage additional sales.

▼ 3. Give an example of the Apriori algorithm for learning association rules.

Suppose you have a dataset of transactions from a grocery store, where each transaction consists of a set of items purchased by a customer. Here's a sample of the transactions:

```
Transaction 1: {milk, bread, eggs}
Transaction 2: {milk, bread}
Transaction 3: {milk, eggs}
Transaction 4: {bread, eggs}
Transaction 5: {milk}
```

We want to use the Apriori algorithm to learn association rules between items in the transactions. Specifically, we want to find all rules of the form X -> Y, where X and Y are sets of items and X union Y is the set of all items in the dataset.

Here's how the Apriori algorithm works:

1. **Generate frequent itemsets**:
   The algorithm first generates all itemsets that have a support (i.e., the fraction of transactions containing the itemset) greater than or equal to a given threshold. For example, if the support threshold is set to 50%, the algorithm would generate the following frequent itemsets:

   ```
   {milk}, {bread}, {eggs}, {milk, bread}, {milk, eggs}, {bread, eggs}
   ```

2. **Generate candidate rules**:
   For each frequent itemset, the algorithm generates all possible non-empty subsets of the itemset. For example, for the frequent itemset {milk, bread}, the algorithm would generate the following candidate rules:

   ```
   {milk} -> {bread}, {bread} -> {milk}
   ```

3. **Calculate rule confidence**:
   For each candidate rule X -> Y, the algorithm calculates the confidence of the rule as the support of the itemset X union Y divided by the support of the

itemset X. For example, if we consider the rule {milk} -> {bread}, the confidence would be calculated as:

```
support({milk, bread}) / support({milk}) = 3/5 = 0.6
```

4. **Filter rules by confidence**:
   The algorithm retains only those rules whose confidence is greater than or equal to a given confidence threshold. For example, if the confidence threshold is set to 70%, the rule {milk} -> {bread} would be retained, but the rule {bread} -> {milk} would be filtered out.

   In our example, suppose we set the support threshold to 40% and the confidence threshold to 60%. Then, the Apriori algorithm would generate the following frequent itemsets:

   ```
   {milk}, {bread}, {eggs}, {milk, bread}, {milk, eggs}
   ```

   And the following rules would be retained:

   ```
   {milk} -> {bread}, {milk} -> {eggs}, {bread} -> {milk}
   ```

These rules indicate that customers who buy milk are likely to buy bread or eggs, and customers who buy bread are likely to buy milk.

▼ ***Reference List***

Apriori Algorithm - GeeksforGeeks

A Computer Science portal for geeks. It contains well written, well thought and well explained computer science and programming articles, quizzes and

https://www.geeksforgeeks.org/apriori-algorithm/

▼ 4. In hierarchical clustering, how is the distance between clusters measured? Explain how this metric is used to decide when to end the iteration.

In hierarchical clustering, the distance between two clusters is measured using different metrics, depending on the type of linkage used. The most common

metrics are:

1. ***Single Linkage:*** The distance between two clusters is defined as the shortest distance between any two points in the two clusters.
2. ***Complete Linkage:*** The distance between two clusters is defined as the longest distance between any two points in the two clusters.
3. ***Average Linkage:*** The distance between two clusters is defined as the average distance between all pairs of points in the two clusters.

Once the distance between clusters is calculated, the clustering algorithm iteratively merges the two closest clusters until all points are grouped into a single cluster. The decision of when to stop the iteration and produce the final clustering is determined by the choice of a stopping criterion. One common criterion is to stop the iteration when the distance between the remaining clusters exceeds a certain threshold. Another criterion is to stop when a specific number of clusters have been formed.

▼ 5. In the k-means algorithm, how do you recompute the cluster centroids?

In the k-means algorithm, the cluster centroids are recomputed as the mean of the data points assigned to each cluster. The steps to recompute the cluster centroids are as follows:

1. For each cluster, calculate the mean of all the data points that are assigned to it.
2. Update the centroid of each cluster with the mean calculated in step 1.
3. Repeat the above steps until convergence, i.e., until the centroids stop changing or a maximum number of iterations is reached.

Mathematically, the centroid of each cluster is computed as follows:

$$centroid = \frac{1}{N} \sum_{i=1}^{N} x_i$$

where N is the number of data points assigned to the cluster, and xi represents the coordinates of the data points.

▼ 6. At the start of the clustering exercise, discuss one method for determining the required number of clusters.

One method for determining ***the required number of clusters at the start of clustering exercise is the elbow method.*** It involves plotting the within-cluster sum of squares (WSS) against the number of clusters, and observing the point where the decrease in WSS starts to level off. This point, where the graph looks like an "elbow," is often a good indication of the appropriate number of clusters for the data set.

To implement this method, one can perform k-means clustering for a range of k values (e.g., from 1 to 10) and calculate the WSS for each value of k. Then, plot the WSS against the number of clusters and identify the elbow point. The number of clusters at the elbow point can be chosen as the appropriate number of clusters for the data set. However, it's important to note that this method is not foolproof, and other methods such as silhouette analysis or expert knowledge may also be used to determine the appropriate number of clusters.

▼ 7. Discuss the k-means algorithm's advantages and disadvantages.

The k-means algorithm has several advantages and disadvantages:

Advantages:

1. ***Easy to implement:*** The algorithm is relatively simple and easy to understand, making it easy to implement and apply in practice.
2. ***Fast:*** The k-means algorithm is computationally efficient and can handle large datasets with ease.
3. ***Scalable***: It can be easily scaled to handle datasets with a large number of features and data points.
4. ***Clusters are well-separated:*** The clusters produced by the k-means algorithm tend to be well-separated and compact, making them easier to interpret and visualize.

Disadvantages:

1. ***Sensitive to initial centroids:*** The choice of initial centroids can greatly affect the final clustering outcome. It is possible to obtain different clustering results with different initial centroid selections.
2. ***May converge to local optima:*** The algorithm may converge to a local optimum that is not the best solution for the problem, particularly when the dataset has complex structures.

3. ***Requires a predetermined number of clusters:*** The algorithm requires a predefined number of clusters to be specified beforehand, which can be difficult to determine in practice.
4. ***Cannot handle non-linear boundaries:*** The k-means algorithm assumes that clusters are spherical and equally sized, making it unsuitable for datasets with non-linear boundaries.

Overall, the k-means algorithm is a powerful and widely used clustering algorithm that can be very effective for many clustering problems. However, it is important to be aware of its limitations and potential drawbacks when applying it in practice.

▼ 8. Draw a diagram to demonstrate the principle of clustering.

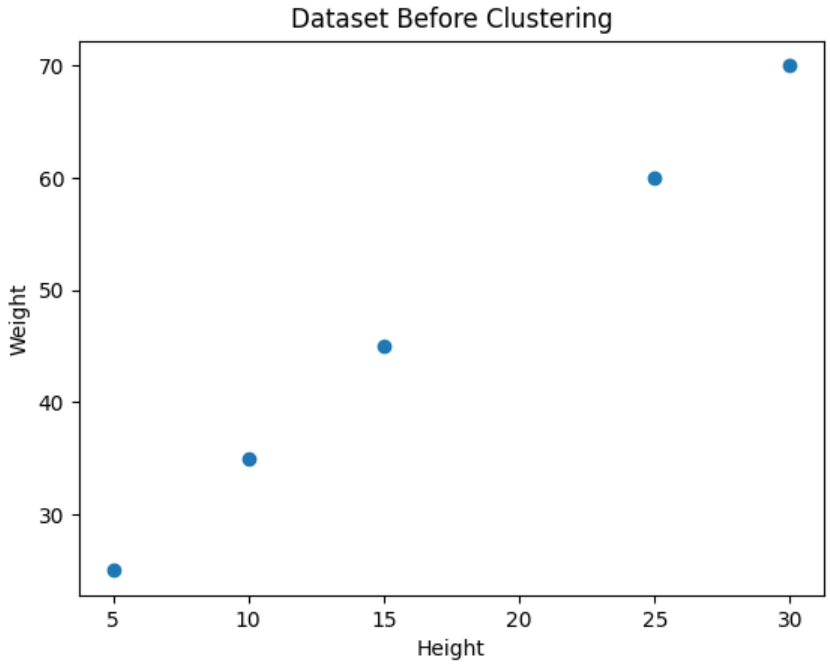
Clustering is a type of unsupervised learning that groups similar data points into clusters based on their attributes. For example, if we have a dataset of different types of fruits and we want to group them into clusters based on their shape and size, clustering algorithms can help us do that.

Let's consider a simple dataset with two features: height and weight. We have five data points in the dataset: (5, 25), (10, 35), (15, 45), (25, 60), and (30, 70). Before clustering, the dataset looks like this:

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.cluster import KMeans

# create the dataset
X = np.array([[5, 25], [10, 35], [15, 45], [25, 60], [30, 70]])

# plot the dataset before clustering
plt.scatter(X[:, 0], X[:, 1])
plt.title('Dataset Before Clustering')
plt.xlabel('Height')
plt.ylabel('Weight')
plt.show()
```



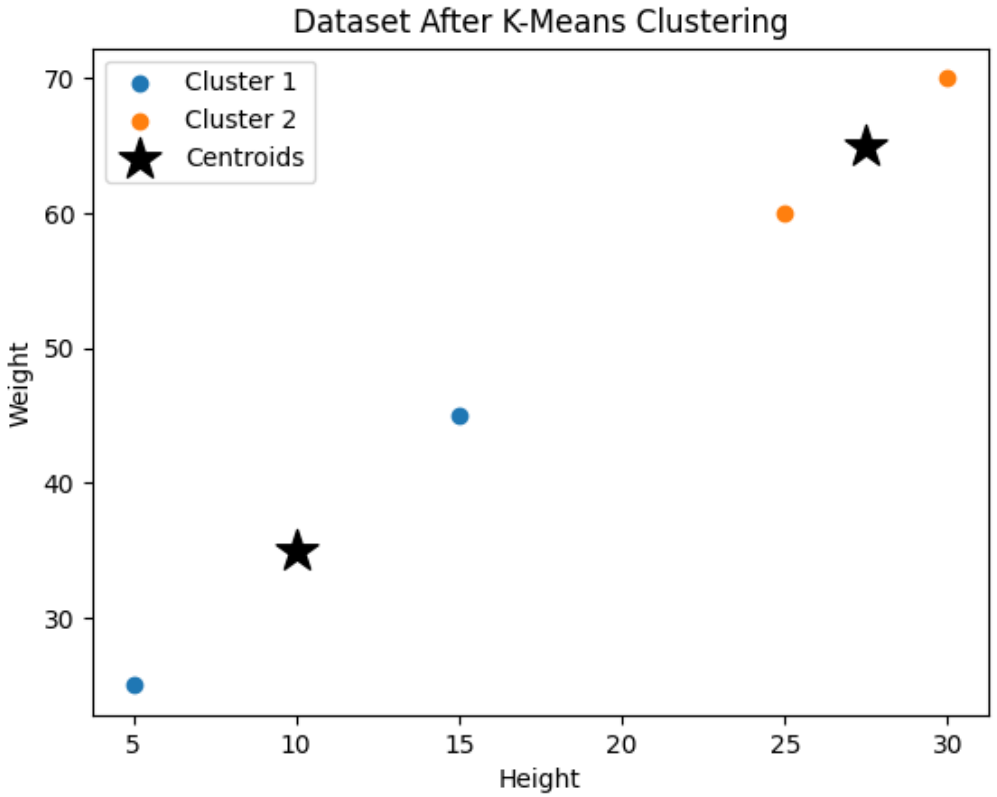
We can use the k-means algorithm to cluster this dataset. First, we randomly select k initial centroids, where k is the number of clusters we want to create. Let's choose k=2 and the initial centroids to be (10, 35) and (30, 70).

The k-means algorithm then assigns each data point to the closest centroid, based on the Euclidean distance between the point and the centroid. After the assignment, the algorithm recalculates the centroids by taking the mean of the points in each cluster. The process of assignment and centroid recalculation continues until the clusters converge or until a specified number of iterations is reached.

After applying the k-means algorithm to our dataset, the clusters look like this:

```
# apply the k-means algorithm with k=2
kmeans = KMeans(n_clusters=2, init=np.array([[10, 35], [30, 70]]), max_iter=100)
kmeans.fit(X)

# plot the clusters after k-means clustering
plt.scatter(X[kmeans.labels_ == 0, 0], X[kmeans.labels_ == 0, 1], label='Cluster
 1')
plt.scatter(X[kmeans.labels_ == 1, 0], X[kmeans.labels_ == 1, 1], label='Cluster
 2')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], marker
='*', s=300, color='black', label='Centroids')
plt.title('Dataset After K-Means Clustering')
plt.xlabel('Height')
plt.ylabel('Weight')
plt.legend()
plt.show()
```



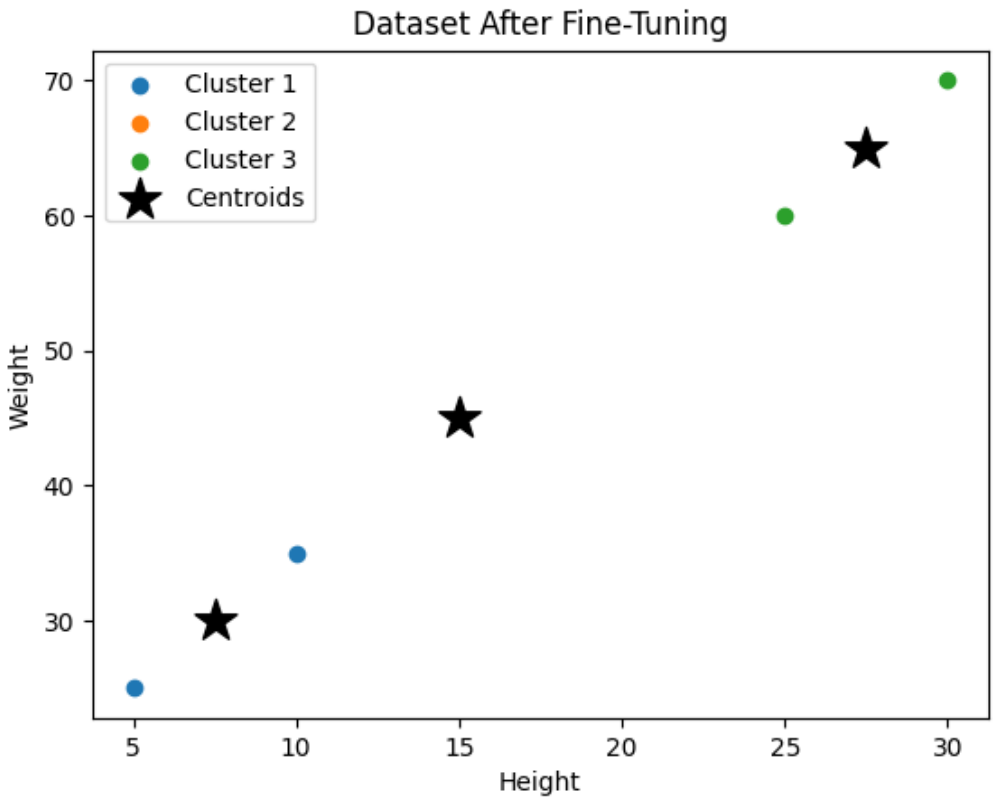
We can see that the algorithm has grouped the data points into two clusters based on their height and weight. However, the clusters are not very well-defined and some of the data points near the boundary are misclassified.

To fine-tune the clustering, we can try different values of k and different initial centroids. For example, if we set k=3 and choose initial centroids at (5, 25), (15, 45), and (30, 70), the clusters look like this:

```
# apply k-means with k=3 and different initial centroids
kmeans = KMeans(n_clusters=3, init=np.array([[5, 25], [15, 45], [30, 70]]), max_it
er=100)
kmeans.fit(X)

# plot the clusters after fine-tuning
plt.scatter(X[kmeans.labels_ == 0, 0], X[kmeans.labels_ == 0, 1], label='Cluster
 1')
plt.scatter(X[kmeans.labels_ == 1, 0], X[kmeans.labels_ == 1, 1], label='Cluster
 2')
plt.scatter(X[kmeans.labels_ == 2, 0], X[kmeans.labels_ == 2, 1], label='Cluster
 3')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], marker
='*', s=300, color='black', label='Centroids')
plt.title('Dataset After Fine-Tuning')
```

```
plt.xlabel('Height')
plt.ylabel('Weight')
plt.legend()
plt.show()
```



We can see that with more clusters, the algorithm can better capture the variations in the data and separate the points into more distinct clusters. However, there is a trade-off between the number of clusters and the interpretability of the results, so we need to choose the number of clusters carefully based on our domain knowledge and the purpose of the analysis.

▼ 9. During your study, you discovered seven findings, which are listed in the data points below. Using the K-means algorithm, you want to build three clusters from these observations. The clusters C1, C2, and C3 have the following findings after the first iteration:

C1: (2,2), (4,4), (6,6); C2: (2,2), (4,4), (6,6); C3: (2,2), (4,4),

C2: (0,4), (4,0), (0,4), (0,4), (0,4), (0,4), (0,4), (0,4), (0,

C3: (5,5) and (9,9)

What would the cluster centroids be if you were to run a second iteration? What would this clustering's SSE be?

▼ 10. In a software project, the team is attempting to determine if software flaws discovered during testing are identical. Based on the text analytics of the defect details, they decided to build 5 clusters of related defects. Any new defect formed after the 5 clusters of defects have been identified must be listed as one of the forms identified by clustering. A simple diagram can be used to explain this process. Assume you have 20 defect data points that are clustered into 5 clusters and you used the k-means algorithm.