Chapter 4

Unsupervised Learning

Unsupervised learning

- With unsupervised learning, the data come without any labels.
- The machine learning model learns to recognize patterns and structure in the data without the input data being labeled with the correct output.
- In customer segmentation, for instance, the model learns to group customers according to their behavior using the input data.
- When training this model, the dataset does not include the segments of each customer.
- Clustering, principal component analysis (PCA), and association rule mining are a few common unsupervised learning algorithms.

Unsupervised learning

The steps that make up unsupervised learning are as follows:

- **Collection of data**: Gather unlabeled data consisting solely of the input.
- **Preprocessing of data**: Preprocess the data and clean it up.
- **Choosing a model**: Select a problem-appropriate unsupervised learning model.
- **Model training**: Use the unlabeled data to teach the unsupervised learning model.
- **Evaluation of a model**: Make use of your domain expertise to evaluate the effectiveness of the unsupervised learning model.
- **Model deployment**: Utilize the model to discover structure and patterns in brand-new data.

Clustering

- Clustering is like having an intuitive assistant that groups similar tasks in our list, making it easier to prioritize and accomplish them efficiently.
- Let’s consider an example of a cabinet filled with clothes.
- We want to organize the cabinet so that it becomes easier to find what we need.
- This is where clustering can be helpful.
- We can think of the clothes as data points and associate features for each data point to define it,

Clustering

- We can start by category; all shirts would go in one group,.
- We can group clothes with similar colors so that they match well.
- We can further organize clothes based on the season — for instance, warm clothes for winter and lighter clothes for summer.
- In this example, clustering can help organize our cabinet by grouping similar items.
- This helps us find what we need faster, ensure that our outfits are coordinated, and simplify maintaining an organized cabinet.

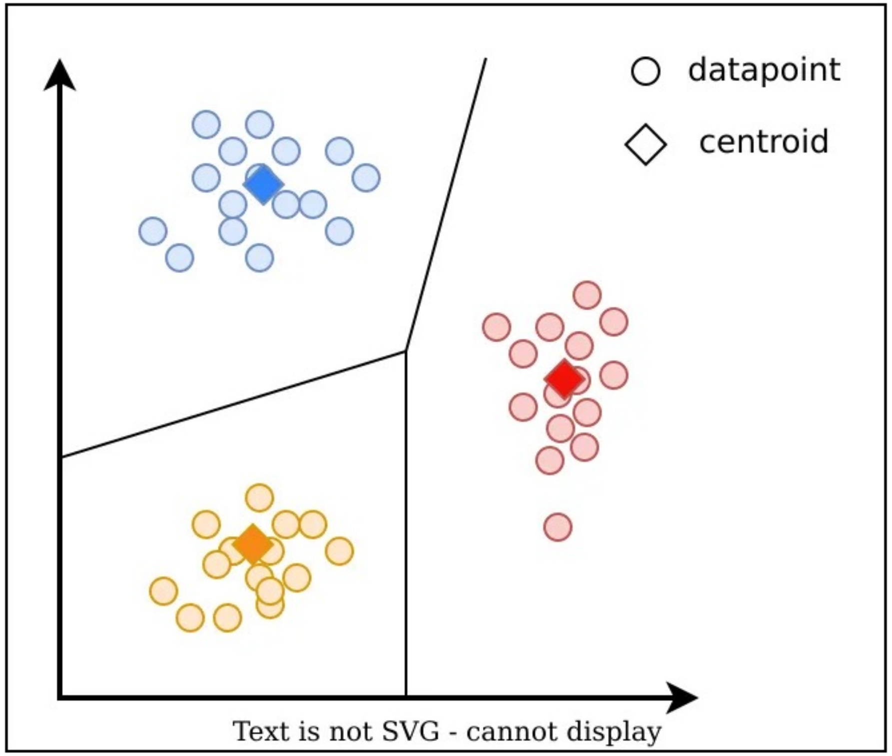
Types of clustering

Centroid-based clustering

- Centroid-based clustering partitions the data into nonoverlapping clusters around centroids.
- A centroid represents the average of all the data points in a cluster.
- During clustering, a data point is assigned to the nearest centroid.
- Centroid-based clustering is widely used for its simplicity and efficiency, particularly with large datasets.

Common use cases of clustering.

- **Marketing**: We can segment the customers using centroid-based clustering algorithms into clusters based on features like purchasing behavior, demographics, age, and gender.
- **Anomaly detection**: We can detect a behavior that deviates from the norm.
- Any data points that are far from any centroid can be considered anomalies.
- **Document clustering**: We can identify document similarities using centroid-based algorithms where each centroid defines the genre.



Centroid-based clustering with three partitions

K-means clustering

- is a popular centroid-based clustering algorithm.
- K-means clustering iteratively associates each data point with a centroid by comparing the sum of distances between the data point and the centroids.
- This measure is commonly referred to as a distance metric.
- The algorithm assigns each data point to a cluster so that the distance metric is minimized.

K-means clustering…

Definition :

- “k-means clustering aims to partition ’n’ observations into ‘k’ clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.”.
- “The key assumptions behind the k-means algorithm:
  1) The center of each cluster is the mean of all the data points that belong to it (hence the name “k-means”).
  2) Each data point belongs to the cluster with the nearest center point.
- These two assumptions are actually sufficient to describe the entire algorithm.
- All the k-means algorithm does is iterate the steps, each trying to satisfy one of these conditions!”

Similarity/Dissimilarity Measures

- Each clustering problem is based on some kind of “distance” or “nearness measurement” between data points.
- Distances are normally used to measure the similarity or dissimilarity between two data objects

Similarity/Dissimilarity Measures

Method	Description
`'chessboard'`	In 2-D, the chessboard distance between (x_1, y_1) and (x_2, y_2) is $max(\lvert x_1 - x_2 \rvert, \lvert y_1 - y_2 \rvert)$
`'cityblock'`	In 2-D, the cityblock distance between (x_1, y_1) and (x_2, y_2) is $\lvert x_1 - x_2 \rvert + \lvert y_1 - y_2 \rvert$
`'euclidean'`	In 2-D, the Euclidean distance between (x_1, y_1) and (x_2, y_2) is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ This is the default method.
`'quasi-euclidean'`	In 2-D, the quasi-Euclidean distance between (x_1, y_1) and (x_2, y_2) is $\lvert x_1 - x_2 \rvert + (\sqrt{2} - 1)\lvert y_1 - y_2 \rvert, \lvert x_1 - x_2 \rvert > \lvert y_1 - y_2 \rvert$ $(\sqrt{2} - 1)\lvert x_1 - x_2 \rvert + \lvert y_1 - y_2 \rvert, otherwise$

Cluster Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters;

Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion

- *k-means*: Each cluster is represented by the center of the cluster
  - **K** is the number of clusters to partition the dataset
  - **Means** refers to the average location of members of a particular cluster

The *K-Means* Clustering Method

- Algorithm:
  - Select K cluster points as initial centroids (the initial centroids are selected randomly)
  - Given *k*, the *k-means* algorithm is implemented as follows:
    - Repeat
      - Partition objects into *k* nonempty subsets
      - Recompute the centroids of each K clusters of the current partition (the centroid is the center, i.e., *mean point*, of the cluster)
      - Assign each object to the cluster with the nearest seed point
    - Until the centroid don’t change

Example Problem

- Cluster the following eight points (with (x, y) representing locations) into three clusters : A1(2, 10) A2(2, 5) A3(8, 4) A4(5, 8) A5(7, 5) A6(6, 4) A7(1, 2) A8(4, 9).
  - Assume that initial cluster centers are: A1(2, 10), A4(5, 8) and A7(1, 2).
- The distance function between two points $a=(x1, y1)$ and $b=(x2, y2)$ is defined as:

$$\text{dis}(a, b) = |x2 - x1| + |y2 - y1| \ .$$

- Use k-means algorithm to find optimal centroids to group the given data into three clusters.

Iteration 1

First we list all points in the first column of the table below. The initial cluster centers – centroids, are (2, 10), (5, 8) and (1, 2) - chosen randomly.

		(2,10)	(5, 8)	(1, 2)	
	Point	**Mean 1**	**Mean 2**	**Mean 3**	**Cluster**
A1	(2, 10)	0	5	9	1
A2	(2, 5)	5	6	4	3
A3	(8, 4)	12	7	9	2
A4	(5, 8)	5	0	10	2
A5	(7, 5)	10	5	9	2
A6	(6, 4)	10	5	7	2
A7	(1, 2)	9	10	0	3
A8	(4, 9)	3	2	10	2

Next, we will calculate the distance from each points to each of the three centroids, by using the distance function:

$$dis(point\ i, mean\ j) = |x2 - x1| + |y2 - y1|$$

Iteration 1

- Starting from point A1 calculate the distance to each of the three means, by using the distance function:
  dis (A1, mean1) = |2 – 2| + |10 – 10| = 0 + 0 = 0
  dis(A1, mean2) = |5 – 2| + |8 – 10| = 3 + 2 = 5
  dis(A1, mean3) = |1 – 2| + |2 – 10| = 1 + 8 = 9
  - Fill these values in the table & decide which cluster should the point (2, 10) be placed in? The one, where the point has the shortest distance to the mean – i.e. mean 1 (cluster 1), since the distance is 0.

- Next go to the second point A2 and calculate the distance:
  dis(A2, mean1) = |2 – 2| + |10 – 5| = 0 + 5 = 5
  dis(A2, mean2) = |5 – 2| + |8 – 5| = 3 + 3 = 6
  dis(A2, mean2) = |1 – 2| + |2 – 5| = 1 + 3 = 4
  - So, we fill in these values in the table and assign the point (2, 5) to cluster 3 since mean 3 is the shortest distance from A2.

- Analogically, we fill in the rest of the table, and place each point in one of the clusters

Iteration 1

- Next, we need to re-compute the new cluster centers (means). We do so, by taking the mean of all points in each cluster.
- For Cluster 1, we only have one point A1(2, 10), which was the old mean, so the cluster center remains the same.
- For Cluster 2, we have five points and needs to take average of them as new centroid, i,e.

  ((8+5+7+6+4)/5, (4+8+5+4+9)/5) = (6, 6)
- For Cluster 3, we have two points. The new centroid is:

  ((2+1)/2, (5+2)/2) = (1.5, 3.5)

- That was Iteration1 (epoch1). Next, we go to Iteration2 (epoch2), Iteration3, and so on until the centroids do not change anymore.
  - In Iteration2, we basically repeat the process from Iteration1 this time using the new means we computed.

Second epoch

Using the new centroid we have to compute cluster members.

		(2,10)	(6, 6)	(1.5, 3.5)	
	Point	**Mean 1**	**Mean 2**	**Mean 3**	**Cluster**
A1	(2, 10)	0	8	7	1
A2	(2, 5)	5	5	2	3
A3	(8, 4)	12			2
A4	(5, 8)	5			2
A5	(7, 5)	10			2
A6	(6, 4)	10			2
A7	(1, 2)	9			3
A8	(4, 9)	3	5	8	1

- After the 2nd epoch the results would be:
  cluster 1: {A1,A8} with new centroid=(3,9.5);
  cluster 2: {A3,A4,A5,A6} with new centroid=(6.5,5.25);
  cluster 3: {A2,A7} with new centroid=(1.5,3.5)

Third epoch

- Using the new centroid we have to compute cluster members.

		(3,9.5)	(6.5, 5.25)	(1.5, 3.5)	
	Point	**Mean 1**	**Mean 2**	**Mean 3**	**Cluster**
A1	(2, 10)	1.5	9.25	7	1
A2	(2, 5)			2	3
A3	(8, 4)				2
A4	(5, 8)				1
A5	(7, 5)				2
A6	(6, 4)				2
A7	(1, 2)				3
A8	(4, 9)			8	1

- After the 3rd epoch the results would be:
  - cluster 1: {A1,A4,A8} with new centroid=(3.66,9);
  - cluster 2: {A3,A5,A6} with new centroid=(7,4.33);
  - cluster 3: {A2,A7} with new centroid=(1.5,3.5)

Fourth epoch

- Using the new centroid we have to compute cluster members.

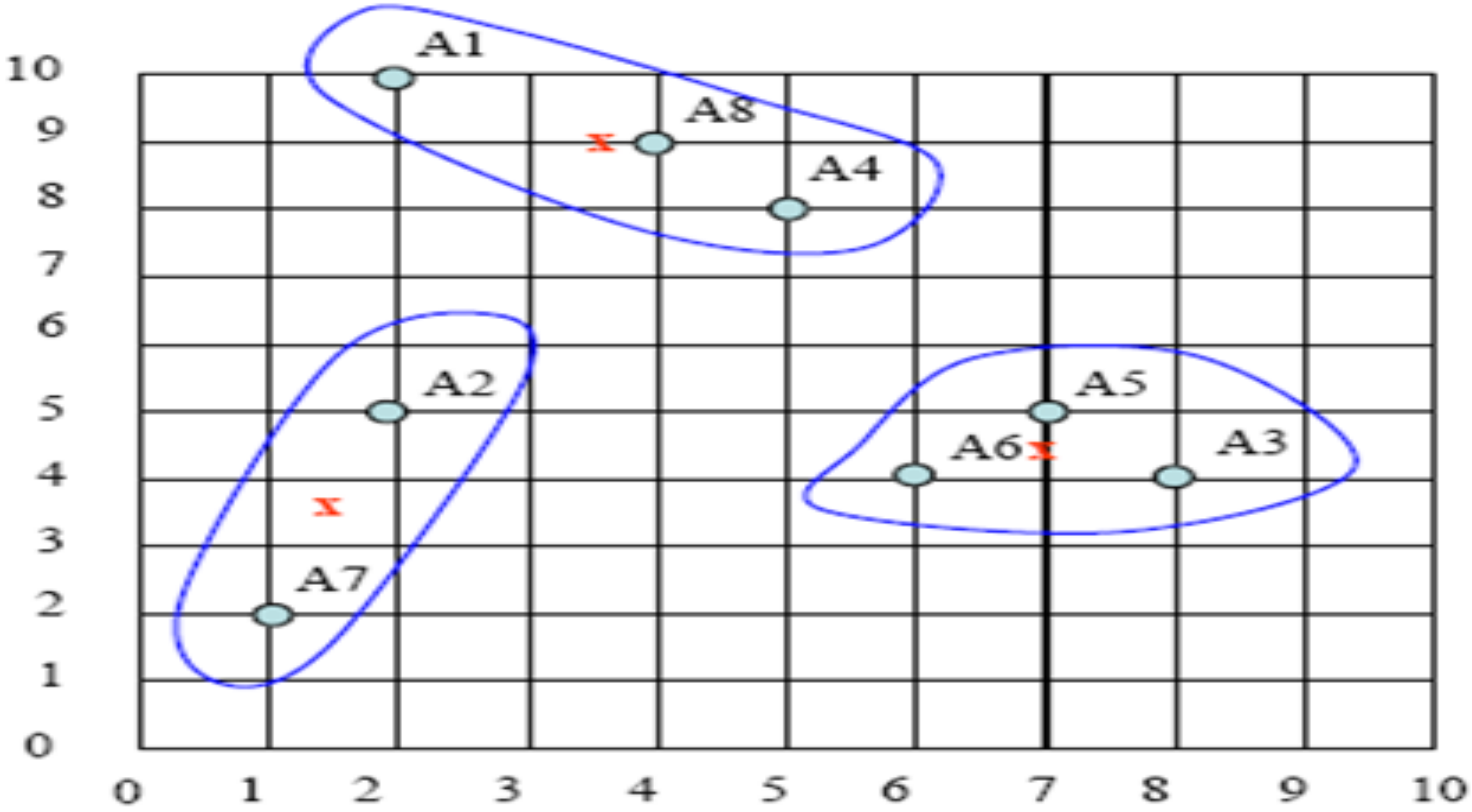
		(3.66,9)	(7, 4.33)	(1.5, 3.5)	
	Point	**Mean 1**	**Mean 2**	**Mean 3**	**Cluster**
A1	(2, 10)	2.67	10.67	7	1
A2	(2, 5)				3
A3	(8, 4)				2
A4	(5, 8)				1
A5	(7, 5)				2
A6	(6, 4)				2
A7	(1, 2)				3
A8	(4, 9)				1

- After the 4^{th} epoch the results would be:

  cluster 1: {A1,A4,A8} with new centroid=(3.66,9);
  cluster 2: {A3,A5,A6} with new centroid=(7,4.33);
  cluster 3: {A2,A7} with new centroid=(1.5,3.5)

Final results

- Finally in the 4^{th} epoch there is no change of members of clusters and centroids. So the algorithm stops.
- The result of clustering is shown in the following figure



Cluster Evaluation

- We use some labeled data (for classification)
  - Assumption: Each class is a cluster.
- After clustering, a confusion matrix is constructed.
- From the matrix, we compute various measurements, entropy, purity, precision, recall and F-score.
  - Let the classes in the data D be C = (c1, c2, …, ck). The clustering method produces k clusters, which divides D into k disjoint subsets, D1, D2, …, Dk.

Confusion Matrix for Performance Evaluation

		PREDICTED CLASS	
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric is measuring Accuracy of the system : The overall accuracy:

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

- Other metric for performance evaluation are Precision, Recall & F-Measure

Cluster Evaluation……Confusion matrix…

Precision:

- is the number of true positives(or true negative) divided by the total number of elements labeled as belonging to that class.
- A high precision means less false positive, while a lower precision means more false positives.

P (precision) = TP/ (TP+FP).

Cluster Evaluation……Confusion matrix…

Recall:

- is the number of true positives(or true negative) divided by the total number of items that actually belong to that class.
- A high recall means that the majority of the 'positive' items were labeled as belonging to the class 'positive'.

R (recall)= TP / (TP+FN)

Cluster Evaluation……Confusion matrix…

F-measure :

- is a measure that combines Recall and Precision into a single measure of performance, this is just the product of Precision and Recall divided by their average.
- Which is defined by the formula

F-measure=2×Precision×Recall/(Precision+Recall).

Hierarchical clustering

- Hierarchical builds a hierarchy of clusters.
- It successively merges or splits existing clusters to create a tree-like structure where the data points are grouped at different levels of granularity.
- There are two main types of hierarchical clustering:
- **Agglomerative hierarchical clustering:**
  - Here, we start by having the same number of clusters as the number of data points. The algorithm then iteratively merges the closest data points together until only one cluster remains.
- **Divisive hierarchical clustering:**
  - Alternatively, we can start with the assumption that all data points belong to a single cluster and then recursively divide them into smaller clusters until each data point is its own cluster.

Hierarchical clustering.....

Some application areas of hierarchical clustering:

- Biology:
  - This type of classification helps classify species based on genetic or morphological traits to construct phylogenetic trees.
- Social network analysis:
  - We can group users for social network analysis based on their similarities in interests.

Hierarchical clustering.....