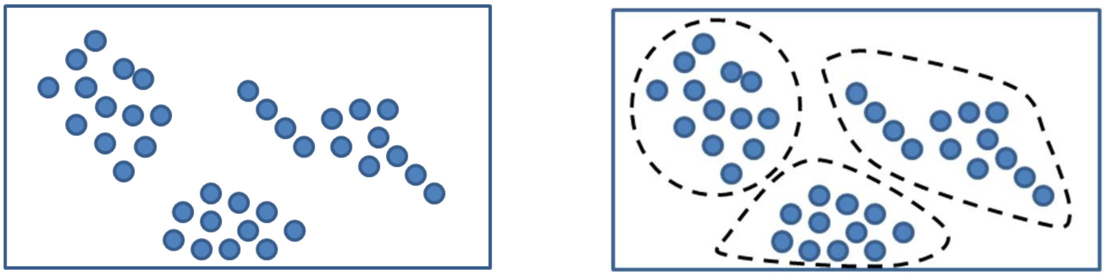
Lecture 16

K-Means Clustering

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable *K*. The algorithm works iteratively to assign each data point to one of *K* groups based on the features that are provided. Data points are clustered based on feature similarity.



(a) (b)

Fig: Unsupervised learning (clustering): (a) shows the same feature set as above while missing the label set. After performing the clustering algorithm, three underlined groups are discovered from the data in (b). Also, users can perform other kinds of unsupervised learning algorithm to learn different kinds of knowledge (ex. Probability distribution) from the unlabeled dataset.

Algorithm

The way k-means algorithm works is as follows:

1. Specify number of clusters *K*.
2. Initialize centroids by first shuffling the dataset and then randomly selecting *K* data points for the centroids without replacement.
3. Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn't changing.

- Compute the sum of the squared distance between data points and all centroids.
- Assign each data point to the closest cluster (centroid).

- Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

The results of the *K*-means clustering algorithm are:

1. The centroids of the *K* clusters, which can be used to label new data
2. Labels for the training data (each data point is assigned to a single cluster)

Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.
The *K*-means clustering algorithm uses iterative refinement to produce a final result. The algorithm inputs are the number of clusters *K* and the data set. The data set is a collection of features for each data point. The algorithms starts with initial estimates for the *K* centroids, which can either be randomly generated or randomly selected from the data set. The algorithm then iterates between two steps:

1. Data assignment step:

Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if c_i is the collection of centroids in set C, then each data point x is assigned to a cluster based on

$$\underset{c_i \in C}{\arg\min}\ dist(c_i, x)^2$$

where $dist(\cdot)$ is the standard Euclidean distance.

2. Centroid update step:

In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster.

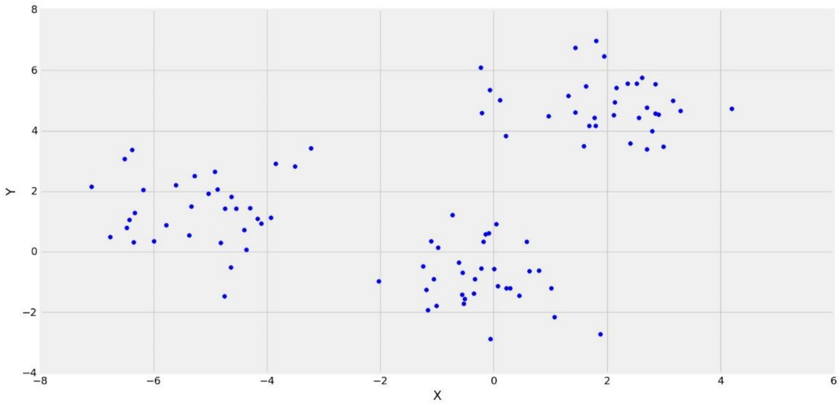
The algorithm iterates between steps one and two until a stopping criteria is met (i.e., no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached).

This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e. not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomized starting centroids may give a better outcome.

Choosing K

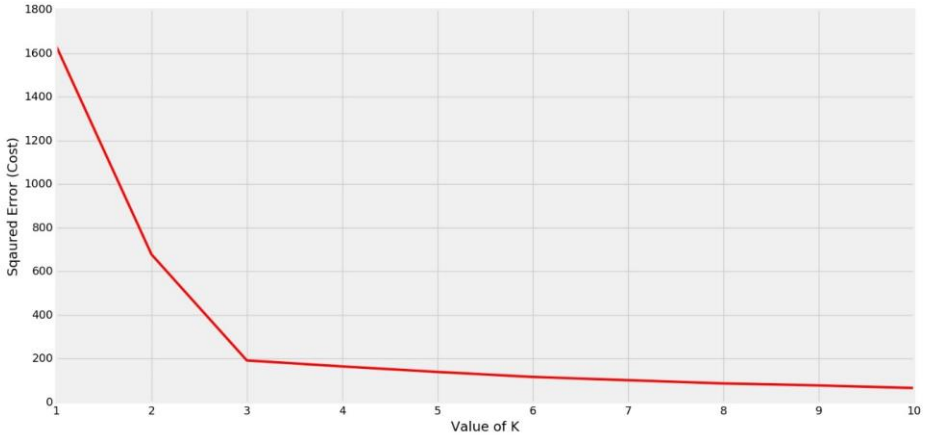
A fundamental step for any unsupervised algorithm is to determine the optimal number of clusters into which the data may be clustered. The **Elbow Method** is one of the most popular methods to determine this optimal value of k.

The idea of the elbow method is to run k-means clustering on the dataset for a range of values of *k* (say, *k* from 1 to 10 in the examples above), and for each value of *k* calculate the sum of squared errors (SSE).



3 clusters are forming

In the above figure, its clearly observed that the distribution of points are forming 3 clusters. Now, let's see the plot for the squared error (Cost) for different values of K.



Then, plot a line chart of the SSE for each value of *k*. If the line chart looks like an arm, then the "elbow" on the arm is the value of *k* that is the best. The idea is that we want a small SSE, but that the SSE tends to decrease toward 0 as we increase *k* (the SSE is 0 when *k* is equal to the number of data points in the dataset, because then each data point is its own cluster, and there is no error between it and the center of its cluster). So our goal is to choose a small value of *k* that still has a low SSE, and the elbow usually represents where we start to have diminishing returns by increasing *k*.

At the top we see a number line plotting each point in the dataset, and below we see an elbow chart showing the SSE after running k-means clustering for *k* going from 1 to 10. We see a pretty clear elbow at *k* = 3, indicating that 3 is the best number of clusters.

Business Uses

The *K*-means clustering algorithm is used to find groups which have not been explicitly labeled in the data. This can be used to confirm business assumptions about what types of groups exist or to identify unknown groups in complex data sets. Once the algorithm has been run and the groups are defined, any new data can be easily assigned to the correct group.

This is a versatile algorithm that can be used for any type of grouping. Some examples of use cases are:

- Behavioral segmentation:
  - Segment by purchase history
  - Segment by activities on application, website, or platform
  - Define personas based on interests
  - Create profiles based on activity monitoring

- Inventory categorization:
  - Group inventory by sales activity
  - Group inventory by manufacturing metrics

- Sorting sensor measurements:
  - Detect activity types in motion sensors
  - Group images
  - Separate audio
  - Identify groups in health monitoring

- Detecting bots or anomalies:
    - Separate valid activity groups from bots
    - Group valid activity to clean up outlier detection

In addition, monitoring if a tracked data point switches between groups over time can be used to detect meaningful changes in the data.

KNN vs. K-mean

Many people get confused between these two statistical techniques- K-mean and K-nearest neighbor. See some of the difference below -

1. K-mean is an unsupervised learning technique (no dependent variable) whereas KNN is a supervised learning algorithm (dependent variable exists)

2. K-mean is a clustering technique which tries to split data points into K-clusters such that the points in each cluster tend to be near each other whereas K-nearest neighbor tries to determine the classification of a point, combines the classification of the K nearest points.