k-means Clustering

Vinay Bhaip*

January 2019

1 Introduction

We have primarily covered a lot of supervised algorithms, where we map some input to some output. These types of algorithms are great when we have labeled data. However, when we have unlabeled data, and need to find groupings within the data, k-means Clustering is a popular algorithm to use. Clustering algorithms are used to find data points with similar features and group them within the data.

2 Algorithm

1. Randomly pick k points as centroids
2. Assign each data point to a group, based on whichever centroid is the closest to it
3. Shift each of the k points to the new centroid of the points in their new group
4. Repeat steps 2 and 3 until none of the grouping assignments change or a certain number of iterations is reached

k represents the number of clusters that we want the data to be separated into. To calculate the distance between the centroid and data points, we use the squared Euclidean distance:

$$d(x, y)^2 = \sum_{i=0}^{n} (x_i - y_i)^2 = ||\mathbf{x} - \mathbf{y}||^2 \tag{1}$$

where n denotes the number of dimensions.

*Based off Sylesh Suresh's lecture

3 Hyperparameter Optimization

The obvious hyperparameter within this algorithm is the value of k. For certain problems, this can be obvious. For example, if we're trying to cluster based off animal species, we might already have a fixed number of groups we know. In other cases, it might not be as clear. If we're just given data and told to cluster it, we need to find how to determine k.

The naive approach to this would be to just look at the data and see where the general clusters are. Clearly, this isn't a good approach, not only because this introduces another degree of arbitrariness, but also because it becomes harder to do this the more dimensions there are.

3.1 Elbow Method

A common method for optimizing the value of k is known as the Elbow Method.

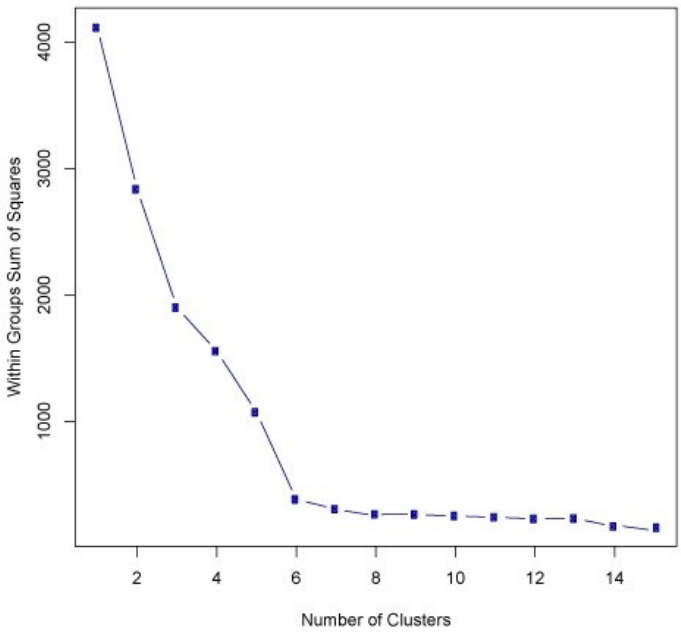

Figure 1: Elbow Method

The graph shows the number of clusters, k, compared to the Within-Groups Sum of Squares (WGSS), the sum of the distances for each of the points within a cluster. The goal is to have each cluster contain points that are the closest to it and to minimize the distance. Generally, as the number of clusters increases,

the WGSS decreases. The graph shows when $k = 6$, the decrease in WGSS is minimal. This "elbow point" is a good indicator on a sufficient k to choose.

4 *k*-means++

The k-means algorithm relies a lot on the initial starting k values, which can be a problem if, for example, the k points are all near each other within the same apparent cluster. To solve this, the k-means++ algorithm picks the starting values of k based off a distribution.

To do this, we select the first initial point randomly from the data. From there, we calculate the squared Euclidean distance from the closest point from centroids we've chosen to each of the other data points, which we will denote as $d(x_i, k_j)$. The probability that the next initial point will be x_i given that the closest point towards it is k_j goes as follows:

$$P(x_i|k_j) = \frac{x_i}{\sum_{i=0}^{n} d(x_i, k_j)))} \tag{2}$$

From this we can see that points that are further apart are favored for the initial k points. After the initial points are chosen, the algorithm follows the original k-means algorithm.