CS 410 Technology Review

Zetian Li, zli123

This technology review will be on the topic of comparing between clustering techniques of k-means, k-medians and k-medoids. These are unsupervised learning approaches that can be used to group like documents together, in which documents within the same group are similar through quantitative measures. An example use case would be provided a group of text documents, we can first extract word vectors using the inverted index, which was covered in this course, or n-Grams, and have those be our quantitative measures, and then employ one of these techniques on those measures to group documents of similar topics together. We can then extract keywords per group that can be used as query terms for future prediction or reach a conclusion on which topics are mainly covered within the group of text documents provided.

The general idea of the three approaches is very similar, which is to group the collection of documents into k clusters. To do so, we will need to find k "center points", where all quantitative measures within each group are closer to their group's center point than those of other groups. If we are to solve this problem in the strong form, as in find a unique absolutely optimal solution, then the problem is an NP-Hard problem that takes exponential time where we need to continuously recompute the absolute solution for every additional document/data point added into our set, which is impractical. The standard way to do so is to solve this problem in its weak form, through an iterative approach, where we start off a random estimation for the k center points, group the data points together, and then find new center points based off the groups formed, and check the difference, and keep on repeating this process until the difference between two iterations becomes small enough (below threshold), which we can consider to be a very close estimation to the actual strong form solution.

With the general idea in mind, the means to compute the center points become the major difference between the three approaches. For k-means, the center of points is represented by the centroid of each cluster, which is the location of that is the closest to all points within the cluster (also known as the average coordinates of all points), through squared-Euclidean distance. For k-medians, the center is represented by the median per axis (get all x coordinates and find median, then find all y coordinates and find median), which is also the closest position to all points but measured through Manhattan distance, where the distance between points is the sum of absolute value of differences in coordinates/positions. For both previous techniques, center point does not necessarily represent a data point (it could, but unlikely); therefore, we have the k-medoid, where the center point is a data point of which the distance from it to all other data points is minimized.

As for k, which is another important measure: we need to know how many clusters to divide our collection into optimally. To compute for k, we used the Elbow's method, where we start off with k being a small value, usually 2, and compute the RMS, or "root-mean-squares", which is the sum of squares of all distances between each individual point from its respective cluster's center point. We then increment k and compute its RMS, and compare that with the k – 1's RMS, and then keep on repeating the process until we find the RMS's become very similar

such that incrementing k provides little to no decrease in the total RMS (usually RMS's become smaller with increasing k, exceptions may occur when k is too small to represent the cluster). A threshold ratio is usually used to represent the stopping point, where when (RMS of k / RMS of k – 1) exceeds this threshold ratio, we have found our optimal k.

So how does each different approach, or the different means of computing the center point, affect the results we obtain? When it comes to accuracy, k-means is the most accurate when our set of data points are mostly evenly distributed, where the distance between the points within the cluster to its center point measured through Euclidean distance are all relatively similar. In the text retrieval and mining scenario, this would mean where the documents are spread across relatively easily recognizable topics and the distribution of data points around these topics are relatively even. If this is not the case, where there are a lot of noises in the documents that do not belong to any cluster, then k-medians become a better approach. This is because k-means can be more easily affected by the noises which can cause the clusters to form incorrectly or sub-optimally. In k-medians, the clusters will form with multiple points that are close to the median and a few noisy points that are very far away. Those outlier points can then be manually removed or re-evaluated, which makes k-medians more reliable when many outliers are present. However, if the analysis needs to be conducted considering all documents to be somehow relevant to a particular topic/cluster regardless of noises, then k-means captures this most accurately. A very notable issue present with both of these approaches is that they perform poorly when the number of documents/data points is small, which can lead to the clusters being formed by different runs of the algorithm to be different, or even not converge at all. If these are concerns with the data set provided, k-medoids can be a solution to that.

k-medoids is significantly different from the previous two approaches. Due to the fact that our center point has to be a data point, we will need to try out all possible points within a cluster and capture the point with the minimum distance away from all other points. This is significantly more computationally expensive, but also the most accurate representation of a cluster. Note that both k-means and k-median may fail to converge even after many, many iterations, while k-medoids will guarantee to converge since there is a finite number of points to select from. k-medoids also carry the advantage of k-medians, where the effect of outliers in the clustering process is minimized. In addition, the property of center point always being a data point is significant when we need to interpret the center point in context. This is significant for text-based mining, since a mishmash of words from a group of documents representing a particular topic may not make sense at all. Having the center point interpretable is significant when it comes to forming query terms and extracting the actual meaning of the cluster formed. The major drawback of k-medoids is that it is computationally more expensive, as we need to compute all pairwise distances for each iteration across all points within a cluster when determining the center point. Compared to k-means and k-medians, they need O(n*i*k) time to reach a result, whereas k-medoids need O(n^2*i*k), where n is the number of points, is the number of iterations it takes to converge, and k is the number of clusters.

Based on the algorithm analysis of the three different approaches, we can see that each method has their own best area of use. For text-based mining, k-medoids offer the most robust and

flexible means of grouping documents together, but also being the most computationally expensive. In the real world, k-means and k-medians can be more useful when a time and space constraint is in place. Note though if we are looking at the bag of words model and implementing BM25 as our quantitative measure, k-means and k-medians are usually sufficient in reaching useful conclusions. If an in-depth analysis needs to be conducted to differentiate between subtle aspects of different topics, then k-medoids can be a better option, especially when the number of documents available is limited.

Reference used:

https://stackoverflow.com/questions/21619794/what-makes-the-distance-measure-in-k-medoid-better-than-k-means

https://stats.stackexchange.com/questions/109547/k-means-vs-k-median

https://towardsdatascience.com/understanding-k-means-k-means-and-k-medoids-clustering-algorithms-ad9c9fbf47ca

https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning

https://towardsdatascience.com/use-this-clustering-method-if-you-have-many-outliers-5c99b4cd380d

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5982597/