1 Introduction

The K-Means algorithm for clustering has the drawback of always maintaining K clusters. This leads to ineffective handling of noisy data and outliers. Noisy data is defined as having little similarity with the closest cluster’s centroid. In K-Means a noisy data item is placed in the most similar cluster, despite this similarity is low relative to the similarity of other data items in the same cluster with the centroid. In part one this project, I have implemented an expanding version of the K-Means algorithm in an attempt to deal with noisy data more effectively. The idea is to first create a new cluster whenever a data item has a distance with the most similar cluster’s centroid beyond a threshold, and then place this data item in it. In other words, the number of clusters K is expandable to accommodate the clustering of data items which contains noisy data.

In real applications of clustering, it is often the case that a data item belongs to more than a single class. For example, an article about the profitability of small market baseball teams has its place in both business and sports sections of a newspaper. In regular K-Means, each data item is placed in the most similar cluster, and this one only. In part two of this project, I have experimented with an overlapping version of K-Means. It places each data item in the clusters which have a similarity measure greater than or equal to a threshold. This allows a data item to be placed in as many clusters as necessary to deal with the clustering of data items which have overlapping classes.

2 Proposed Algorithms

The following is regular K-Means:

1. Select K data points as the initial representatives.
2. For i = 1 to N, assign item x_i to the most similar centroid (this gives K clusters).
3. For j = 1 to K, recalculate the cluster centroid C_j.
4. Repeat steps 2 and 3 until there is (little or) no change in clusters.

As an alternative, the following is the expanding version of K-Means (part one):

1. Select K data points as the initial representatives.
2. For i = 1 to N, assign item x_i to the most similar centroid (this gives K clusters).
3. For j = 1 to K, calculate the mean and standard deviation of similarity measure between every data item in Cluster_j and cluster centroid C_j.
4. For i = 1 to N, if the similarity measure between item x_i and its cluster’s centroid has a z-score < threshold, place item x_i in cluster K+1.
5. If step 4 was applied to any data item \( x_i \), then reassign \( K = K + 1 \).
6. For \( j = 1 \) to \( K \), recalculate the cluster centroid \( C_j \).
7. Repeat steps 2 to 6 until there is no change in clusters.

The similarity measure used is Cosine similarity:

\[
\text{sim}(X,Y) = \frac{\sum_i (x_i \ast y_i)}{\sqrt{\sum_i x_i^2 \ast \sum_i y_i^2}}
\]

The standard deviation is the square root of variance:

\[
\text{sd} = \sqrt{\frac{\sum_i (x_i - \text{mean})^2}{n - 1}}
\]

The z-score is defined as:

\[
z' = \frac{z - \text{mean}}{\text{sd}}
\]

In the expanding version of K-Means after the reallocation of items in step two, the similarity measure of every item with its cluster’s centroid is calculated. Then for each cluster, the mean and standard deviation of all similarity measures are computed for the purpose of deciding which data items are too distant (noisy) from their cluster’s centroid. If noisy data is detected, a new cluster is created for them. A data item is considered to have significant distance from its cluster’s centroid if the data item’s z-score is below the z-score threshold. A data item which is more similar to its cluster’s centroid relative to other members of the cluster has a positive z-score, and a data item which is more distant from its cluster’s centroid relative to other members of the cluster has a negative z-score. Therefore, the appropriate z-score threshold should be set as less than zero to screen for noisy data. The more negative a z-score threshold is used means the more tolerant the algorithm of noisy data.

If during any iteration of the algorithm a new cluster is created due to the detection of noisy data, \( K \) is incremented by one in the next iteration.

As another alternative to K-Means, the following is the overlapping version of K-Means (part two):

1. Select \( K \) data points as the initial representatives.
2. For \( i = 1 \) to \( N \), assign item \( x_i \) to the most similar centroid (this gives \( K \) clusters).
3. For \( i = 1 \) to \( N \), assign item \( x_i \) to every cluster which has a similarity measure with the item \( \geq \) threshold.
4. For \( j = 1 \) to \( K \), recalculate the cluster centroid \( C_j \).
5. Repeat steps 2, 3, and 4 until there is no change in clusters.

In the overlapping version of K-Means, after the reallocation of items in step two, each data item is checked with all clusters for a similarity measure which meets or is above the given similarity threshold. If such clusters are found, the data item is placed in each of them. If no such clusters are found, then the data item is placed in only the cluster that has a centroid most similar to the data item (step 2). When the similarity threshold is set very low, every item is likely to be placed in all \( K \) clusters. When the threshold is set to
be maximum (1.0), the algorithm resembles regular K-Means, because step three of the algorithm is not likely to succeed in finding such clusters.

3 Evaluation

3.1 Description of Data Set

The data set for part one of the experiments is a collection of 45 web documents spanning over six different classes. These classes are of topics AI introduction, Camry review, A. Einstein biography, M.L. King biography, NBA finals, and K. Rowlands biography. The first four of these classes have ten documents each, and the last two classes have four and one documents respectively. The documents in the last two classes are designated as noisy data. The documents are initially preprocessed using a separate program to remove punctuation marks and symbols, and change all letters to lower case. The main program also does some preprocessing of data such as changing words to their canonical form by applying Porter’s stemming algorithm, and removing stop words and numbers because they are mostly semantically meaningless. The purpose of these preprocessing tasks is to normalize similar words so that their occurrences are accounted as the same word, and remove words that occur commonly in many documents because they offer no help in document discrimination.

The data set for part two of the experiments is a collection of 15 web documents which are inclusive of two classes. These classes are the topics of football and baseball. There are five documents in the class of football, five in the class of baseball, and five that overlap these two classes. The same preprocessing procedures described previously are also applied to these documents.

3.2 Document Representation

In the data structure of part one, each document is represented by a vector of binary weights. These binary weights indicate which terms a document contains. Binary weights seem sufficient given that the documents used in the experiments of part one are quite uniform in length throughout the collection, and the topics in the document collection are diverse, so there exist terms that are unique to each topic. In many instances, by merely containing a particular unique term (having a 1 in the vector attribute), the document is differentiated from others in the collection.

In part two, the document collection has slightly different characteristics. The documents have only two topics, namely football and baseball. These topics of documents are similar in that they are both about sports, so many terms normally considered as unique to a class may appear in the other class as well. What determines a document’s apparent topic is the frequency of these unique terms it contains. For example, the terms ‘San Francisco’ (for simplicity let’s call this one term) and ‘Cincinnati’ appear across the two classes of documents. If a document has a higher frequency of the term ‘San Francisco’ than ‘Cincinnati’, then this document is more likely to be from the class of football, because all of the documents in the collection are about Deion Sanders, a former player of the San Francisco 49’ers football team, even
though there exists a baseball team in the same city called the Giants. If another
document has a higher frequency of the term ‘Cincinnati’ than ‘San Francisco’, then this
document is more likely to be from the class of baseball, because Deion Sanders is a
former player of the Cincinnati Reds baseball team, even though there exists a football
team in the same city called the Bengals. If some document has about equal frequency of
the terms ‘Cincinnati’ and ‘San Francisco’, then this document is likely to be an
overlapping document about the sports career of Deion Sanders as a 49’er and a Red.
Because the frequency of unique terms is the determinant of a document’s class, in part
two a vector of term frequency weights instead of binary weights is used as the data
structure to represent a document in the collection.

3.3 Description of Experiments

The purpose of experiment one is to compare the effectiveness of regular K-Means and
the expanding version of K-Means across various z-score thresholds in clustering data
which contains noise. The initial clusters for the experiments are kept constant for all
trial runs to avoid variations in the results due to difference in initial clusters setup. The
initial clusters are chosen to be four documents (K = 4), each one representing an initial
cluster. The four documents are from AI introduction, Camry review, A. Einstein
biography, and M.L. King biography which are designated as main topics. The
remainder of the documents are not assigned to any clusters initially. The documents of
the remaining topics NBA and K. Rowlands are treated as noisy data. It is important that
each main topic is represented in an initial cluster for the algorithm to be effective in
producing good resulting clusters.

The purpose of experiment two is to compare the effectiveness of regular K-
Means and the overlapping version of K-Means across various sim-value thresholds in
clustering documents which overlap classes. The documents about football are initialized
as a cluster, and the documents about baseball are initialized as another cluster. The
documents yet to be initialized are the overlapping documents. By starting out the
algorithm with initial clusters that are good representatives of the two classes, we could
focus on finding the sim-value thresholds that are effective in clustering the overlapping
data items.

3.4 Cluster Evaluation methods

In experiment one, the quality of resulting clusters is measured in terms of entropy. The
more classes of documents in a cluster, the higher the entropy. The more similar the
documents (in terms of being from the same class), the lower the entropy, and hence
better quality of cluster. But for the expanding version of K-Means, it is not enough to
consider just the entropy within a cluster. The entropy within a class is also important.
This is because the algorithm permits the expansion of the number of clusters, so it is
possible, at an extreme, to return a set of clusters with a single document in each cluster.
In this scenario, every cluster has the lowest possible (perfect) entropy, but each class of
documents is so dispersed in different clusters that the result of clustering is useless
(assuming that each class contains more than one document). The measure of class
entropy would give us an idea about how concentrated or dispersed a class of documents
end up in the resulting clusters. At the other extreme, a class having all of its documents in a single cluster has the lowest possible (perfect) entropy. The measure of cluster quality is therefore the addition of the sum of cluster entropy and the sum of class entropy. The formula for entropy is defined as:

\[ I = - \sum p_i \cdot \log_2 p_i \]

The quality of resulting clusters is the sum of entropy of all clusters and entropy of all classes:

\[ I_{\text{total}} = I_{\text{cluster}} + I_{\text{class}} \]

In experiment two, the evaluation is different from the previous section. Here the measure of effectiveness is the percentage of correct classification of documents in the K clusters. At the beginning of the overlapping version of K-Means algorithm, two initial clusters with each one having five documents of a single class are created. These documents give the initial clusters a good representation of the two classes in the collection. With such good initial clusters, it is assumed (and shown in test runs) that the centroid of these clusters don’t change much (class of documents stays the same) upon the addition of overlapping documents to clusters, so the remainder of the algorithm’s iterations are mostly calculating which of the clusters should be added the overlapping documents. After clustering is completed, we know which cluster(s) to look for a document, because we know the class designation of each resulting cluster and the class(es) of that document. The evaluation proceeds with checking each document whether it is correctly classified as a member or non-member of a cluster, for all K clusters. The same is done for all documents and the sum of these correct classifications is used in the computing the percentage of correct classification of documents in the K clusters:

\[ \frac{\sum \text{number of times document}_i \text{ is classified correctly in the K clusters}}{\text{number of documents} \times K} \]

In this experiment the entropy of clusters is not used in evaluation. It is assumed that the cluster entropy is taken into account by the percentage of correct classification measure, because cluster entropy is equivalent to the incorrect classification of documents. The percentage of correct classification is similar to class entropy, except we have designated which cluster(s) a document should and should not be found.

3.5 Experimental Results

In experiment one, the expanding version of K-Means was tested against the regular version of K-Means for the total entropy of the resulting clusters. The regular version of K-Means total entropy for the resulting clusters was 2.74. The entropy was a result of misplacement of noisy data. The noisy data of class NBA and K. Rowlands being placed in the main clusters (clusters which have documents that are designated as main topics as the majority) contributed to cluster entropy, and the noisy data of class NBA being placed
in two different clusters contributed to class entropy. The expanding version of K-Means total entropy for the resulting clusters varies with the z-value threshold. The z-value thresholds used to experiment ranged from -2.0 to -1.0. The total entropy is highest (10.46) when z-value threshold was -1.0, and it is lowest (zero) when z-value threshold was -1.7. When z-value threshold is high (-1.0), the tolerance for deviation in a cluster is low. With low tolerance new clusters are created more frequently. In fact, seven new clusters are created when z-value threshold of -1.0 was used, even though the document collection contained a total of six classes (four main, and two noise). The expanding version of K-Means performed best using -1.7 z-value threshold (zero entropy). At this threshold, the noisy data did not get placed in the four main clusters, and instead they were placed in two new clusters with each cluster containing a single topic of noisy documents. The experiment shows the expanding version of K-Means is more effective in clustering this particular document collection using z-score threshold of -2.0 to -1.6 than the regular version of K-Means which does not increase the number of clusters to accommodate noisy data.

In experiment two, the overlapping version of K-Means was tested against the regular version of K-Means for the percentage of correct classification as seen in the resulting clusters. The regular version of K-Means percentage of correct classification for this document collection was 83.3%. The algorithm failed to correctly classify the overlapping documents, because it is only capable of placing each overlapping document in a single most similar cluster. At higher (0.7 to 1.0) sim-value thresholds, the overlapping version of K-Means resembled regular K-Means. This is because step three of the algorithm is not likely to succeed as mentioned before. At lower (0 to 0.5) sim-value thresholds, the algorithm performed worse than regular K-Means. The problem was that at low thresholds, most, if not all, documents were classified into all K clusters, even if they didn’t belong. These incorrect classifications contributed to lower percentages of correct classification. The highest percentage of correct classification 96.7% was generated by the overlapping version of K-Means at sim-value threshold = 0.65. It was able to classify all, except for one, overlapping documents into two classes they belong in. This shows that the overlapping version of K-Means is more effective in
clustering this particular document collection using sim-value of 0.6-0.65 than the regular version of K-Means which is incapable of placing overlapping documents in more than one cluster.

![Fig. 2. Correct classification percentage of documents using K-Means and K-Means at various sim-value thresholds.](image)

4 Conclusion

In this project, the K-Means algorithm extensions are examined. I described the expanding version of K-Means to cluster data containing noise, and the overlapping version of K-Means to cluster data containing overlapping classes. At certain thresholds, these alternative algorithms have shown to be more effective than regular K-Means in clustering of the data collections provided, and the thresholds for these successful runs in the experiments are reported. Other experiments may be performed using a combination of the two algorithm extensions to cluster data which contains both noise and overlapping classes.