CS 6375
ASSIGNMENT 3: Part 2

Names of students in your group:

Derek Carpenter / dxc190017
Randy Suarez Rodes / rxs179030

Number of free late days used: 0

Note: You are allowed a **total** of 4 free late days for the **entire semester**. You can use at most 2 for each assignment. After that, there will be a penalty of 10% for each late day.

Please list clearly all the sources/references that you have used in this assignment.

Abstract

This report lays out the programming methods and solution for computing the clusters of a dataset of tweets. We will use python to implement a K-means algorithm using Jaccard Distance as a metric to determine clusters. To evaluate the results of our program we will use sum of squared error (SSE). Detailed charts and tables of the results will also be shown in this report.

Contents

Problem Description

Tweets contain 140-characters of words that have some meaning. The goal of this program is to apply a k-means unsupervised learning algorithm to a given dataset of tweets. This will allow us to group together similar tweets that can be used for various things such as trend analysis. To implement this k-means algorithm we will use Jaccard Distance to determine clusters.
In summary, the primary objectives of this assignment are the following:

- Compute the similarity between tweets using the Jaccard Distance metric.
- Cluster tweets using the K-means clustering algorithm.

Data Import and Preprocessing

We are given a dataset of thousands of tweets in a text file with each new line containing one tweet formatted in the following way:

220146274970763264|Tue Jul 03 13:25:36 +0000 2012|Heavy Coffee Intake May Affect Fertility Treatments: Study: http://on-msn.com/P5E3ge

To use this information we must convert this tweet into a usable object in python. To do this we will store each tweet as a list of words, for example:

Tweet1 = ['heavy', 'coffee', 'intake', 'may', 'affect', 'fertility', 'treatments', 'study']
Tweets = [Tweet1, Tweet2, …. Tweetn]

To filter out all the unnecessary information (words that start with @, hashtags#, date, id, URLs). Then we use a simple Regular Expression that extracts all words in the tweet. This can be seen in the python code.

K-Means

Now that we have our preprocessed data, we will create a class that performs K-Means clustering on the Tweets dataset from above. K-Means works by determining k centroids and calculating distances between data points and the centroids. The point is assigned to the closest of the k centroids.

Jaccard Distance

To determine distances, we will use the Jaccard Distance which is defined below.

$$Dist(A,B) = 1 - \frac{|A \bigcap B|}{|A \bigcup B|} = \frac{|A \bigcup B| - |A \bigcap B|}{|A \bigcup B|}$$

Converting the tweet word list to a set and computing the intersection and union of the two tweets, we can calculate the Jaccard distance. If the Jaccard Distance(A,B) is large, the tweets are not similar, conversely if it is small then the tweets are similar.

Calculation Centroid

Initially we randomly select k centroids and compute distances and assign datapoints to centroids. Then we must recalculate the true centroid of the cluster. To do this we find the element of the cluster with minimum sum of distances.

$$New\ Centroid_i = \frac{ArgMin}{point_j \epsilon C_i} \sum_{k=1}^{n_{clust}} Dist(point_j, point_k)$$

Once we have our new centroid calculated, we will iterate the process again until there are no new changes in points to centroids.

Results

To evaluate the results, we will use sum of squared error as defined below:

$$SSE = \sum_{i=1}^{K} \sum_{x \epsilon C_i} dist^2(m,x)$$

To best visually see the SSE we will generate an elbow plot seen in Figure 1 that computes the SSE for k = 1 to n clusters. We will see the error go down quickly reaching an ‘elbow’ then have a smaller rate of change of error after a certain k.

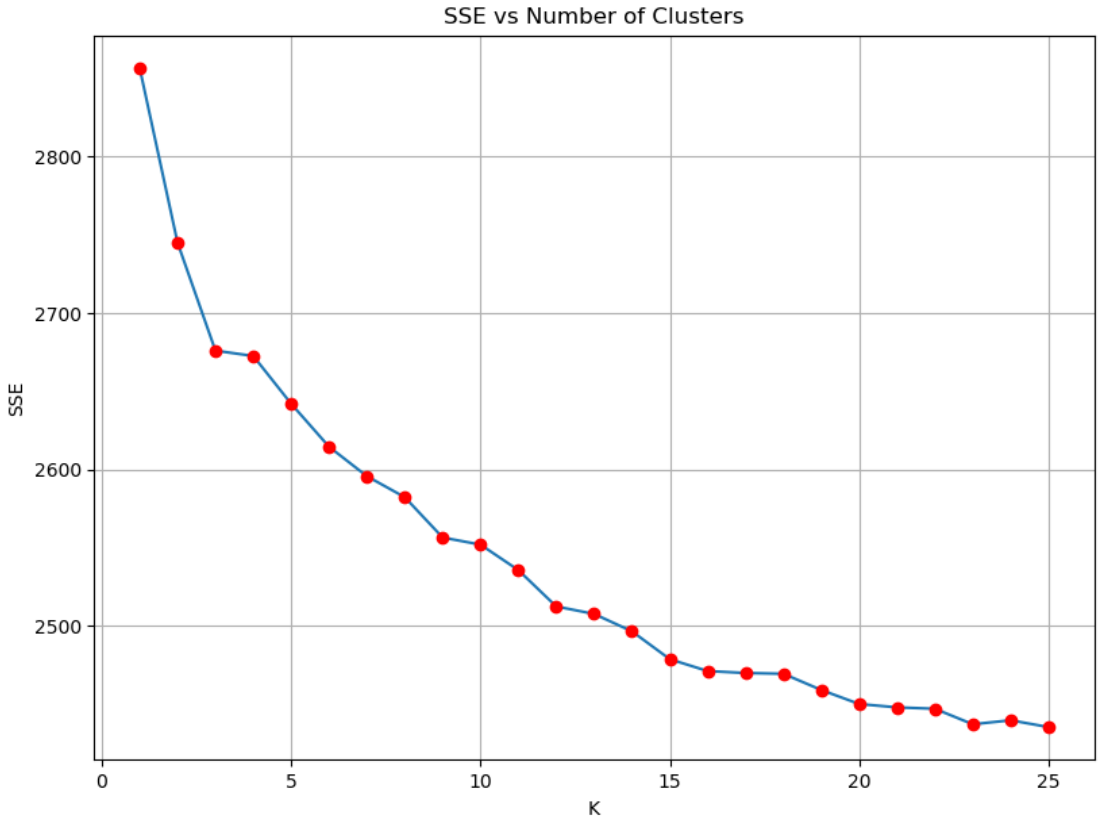

Figure 1: Elbow Plot of K-Means on Tweet Dataset

To verify that our tweets are being clustered correctly we have extracted 5 tweets from an example cluster. We can see below that there are similar words between the tweets.

Centroid Tweet:
['amid', 'outbreak', 'foster', 'farms', 'steps', 'up', 'food', 'safety']

Tweets in Cluster:
['terminally', 'ill', 'basketball', 'player', 'signs', 'up', 'for', 'hospice', 'care']
['ryan', 'reynolds', 'michael', 'j', 'fox', 'team', 'up', 'on', "parkinson's"]
['snack', 'tax', 'navajo', 'lawmakers', 'ok', 'price', 'hike', 'on', 'junk', 'food']
['ebola', 'aware', 'cdc', 'steps', 'up', 'traveler', 'monitoring']
['ohio', 'mumps', 'outbreak', 'tops', 'whole', 'u.s', 'last', 'year']

To further verify out results, we have generated a table with the k value, SSE for each k, and the number of data points in each cluster.

Value of K	SSE	Number of Tweets per Cluster (Cluster: Tweets)
10	2552.06	['1: 181', '2: 584', '3: 763', '4: 557', '5: 358', '6: 27', '7: 275', '8: 103', '9: 169', '10: 180']
11	2535.755	['1: 449', '2: 1028', '3: 304', '4: 341', '5: 26', '6: 136', '7: 95', '8: 235', '9: 109', '10: 177', '11: 297']
12	2512.555	['1: 399', '2: 868', '3: 333', '4: 248', '5: 331', '6: 22', '7: 186', '8: 223', '9: 101', '10: 228', '11: 88', '12: 170']
13	2507.759	['1: 397', '2: 843', '3: 215', '4: 313', '5: 278', '6: 23', '7: 185', '8: 225', '9: 102', '10: 135', '11: 227', '12: 90', '13: 164']
14	2496.635	['1: 538', '2: 230', '3: 782', '4: 215', '5: 183', '6: 277', '7: 22', '8: 225', '9: 85', '10: 106', '11: 150', '12: 143', '13: 88', '14: 153']
15	2478.83	['1: 499', '2: 232', '3: 721', '4: 204', '5: 211', '6: 187', '7: 285', '8: 22', '9: 71', '10: 102', '11: 122', '12: 136', '13: 85', '14: 153', '15: 167']

References