Homework 1

Sundesh Raj, 1001633297

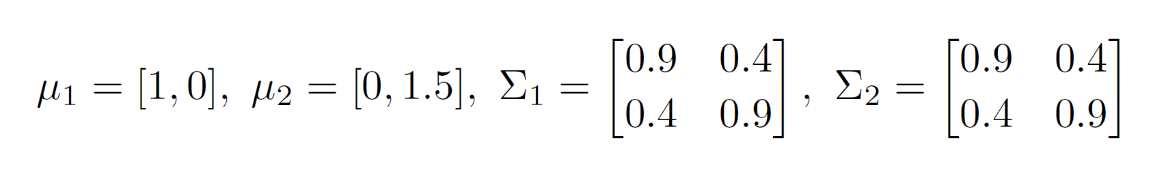
Due Date: 04th March 2020

**Inferences: Following points answered**

* Describing what you have solved
* What implementation has been done
* Interpretation of the results

## Problem 1:

Generate 2 sets of 2D Gaussian random data, each set containing 500 samples using the given parameters,

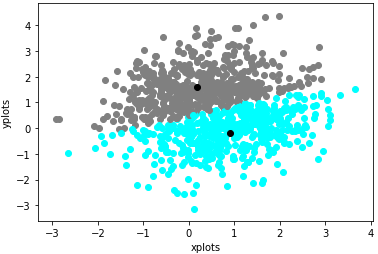


1. Write a function myKmeans (X, k, c), that clusters the data in X into k clusters. Here c are the initial cluster centres. Terminate the iteration when the l2 norm between the previous and the updated centre is < 0.001 or the number of iterations reaches 10000.

* Initially we create 2 separate random data sets with the given parameters. Once we have both the data sets, we concatenate them to form the 2D data set.
* I have created 2 data sets randData1 and randData2 respectively and concatenate both to form randDataSet. This is one of the parameters that is required to pass to the myKmeans (X, k, c) method, here randDataSet is the X.
* Next, I declare a variables k1 and k2 respectively. These are the number of clusters that we will be working with.
* K1 depends on clusterCenters1 and k2 depends on clusterCenters2 respectively.
* The myKmeans (X, k, c) takes the X (randDataSet) along with k1/k2 (for k = 2 and k = 4) and clusterCenters1/ clusterCenters2 (for cluster centres respectively)
* There is loop within the myKmeans() function which loops to converge the centres until the l2 norm of the updated centre is <=0.001 or the number of iterations reaches 10000
* Finally, with the newfound centres we plot the scatter plot along with the new centres.

2. To the data obtained in 1a we apply k=2 and with centres c1=(10,10) and c2=(-10,-10)

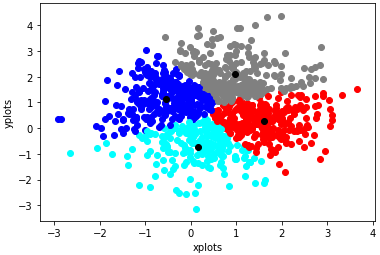
* The myKmeans(X, k, c) method here takes k=2 and the centres mentioned above, and we get the below mentioned output details.
* The number of iterations taken when k=2 is 19
* The new centres for the 2 clusters are [[0.17172398 1.61602226], [0.91170748 -0.18450387]]
* Below is the scatter plot for the above-mentioned clusters and centres,



* Here we can see the centres amidst the 2 clusters.

3. To the data obtained in 1a we apply k=2 and with centres c1=(10,10), c2=(-10,-10), c3=(10,-10) and c4=(-10,10)

* The myKmeans(X, k, c) method here takes k=4 and the centres mentioned above, and we get the below mentioned output details.
* The number of iterations taken when k=4 is 25
* The new centres for the 4 clusters are [[ 0.96642861, 2.10847756],[ 0.15191004, 0.73979856], [ 1.59885762, 0.27326394], [-0.5585929, 1.1544236 ]]
* Below is the scatter plot for the above-mentioned clusters and centres,



* Here we can see the centres amidst the 4 clusters.

## Problem 2:

Using the Amazon reviews csv dataset, compute tf-idf weight matrix, count matrix and visualize.

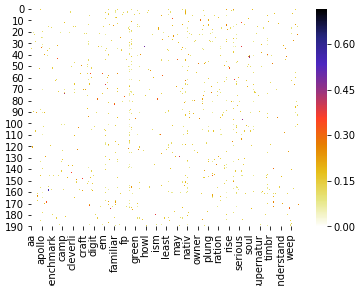
1. Let us focus on the reviews from the dataset without their labels. Build a weight matrix about

all the words in the review part. Represent each review as a real-valued vector of tf-idf. Report each

pre-processing step you applied and attach the corresponding part of your code. Visualize the matrix

with colour code (i.e., show the matrix as an 2D image where pixel intensity represents the weight)

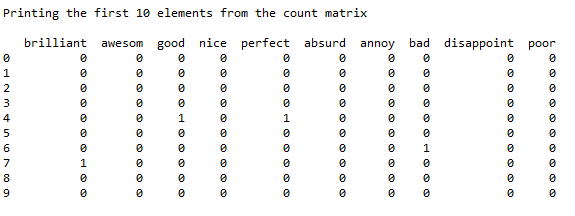
* Firstly, I have used the pandas library to import the Amazon reviews csv data file to the python file
* For pre-processing the data, I have used the stop-words from NLTK library and the Porter Stemmer library for stemming.
* Once pre-processing is complete, I obtain the complete set of revised reviews as corpus.
* I create an object of the TfidfVectorizer() to fit the corpus and form the tf-idf weight matrix.
* Once we obtain the tf-idf weight matrix, I have used the seaborn visualization package to create the 2D colour coded map of the weight matrix, as shown below,



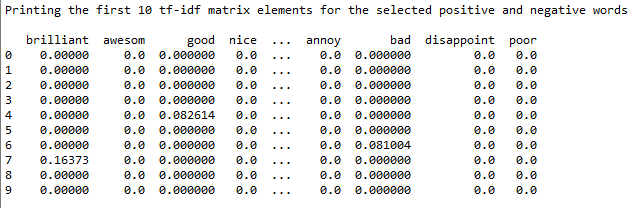
2. Pick your own 5 “positive” words and 5 “negative” words, which indicate if a product is good

or bad, respectively. List the words you selected. Represent each review in a vector space of these ten words (i.e., count matrix) as well as tf-idf weight matrix.

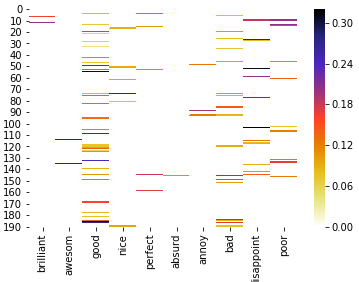
* The five positive words that I have chosen are -> **brilliant, awesom, good, nice, perfect**
* The five negative words that I have chosen are -> **absurd, annoy, bad, disappoint, poor**
* I have used the CountVectorizer() package to fit the corpus data to find the count matrix
* Once I obtain the complete count matrix, I save a new data frame which contains only the columns with the 5 positive and negative words mentioned above.
* Listing the column names from the new data frame gives us a list of all the positive and negative words that we have provided.
* For the purpose of clear visuals, I am printing only the first 10 rows of the count matrix, as seen below,



* From the previously obtained tf-idf weight matrix in Q2a, I am extracting the columns for only the positive and negative words mentioned above.
* For the purpose of clear visuals, I am printing only the first 10 rows of the tf-idf matrix shown below,



* The new 2D colour coded image for the tf-idf positive and negative words are as shown below,



3. For each review, sum up the frequency of “positive” words and “negative” words. Represent

each review as a vector of length 2. Now the reviews can be shown in 2D space, while one dimension

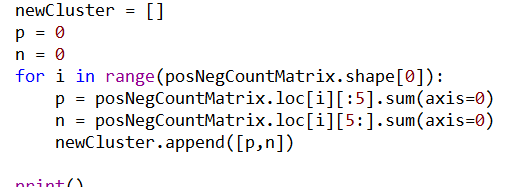
is about “positive” and the other one is “negative”. Apply your code from Problem 1 to this 2D data

with k = 2; 3; 4 with randomly initialized centers. In your report, report the centers found for each

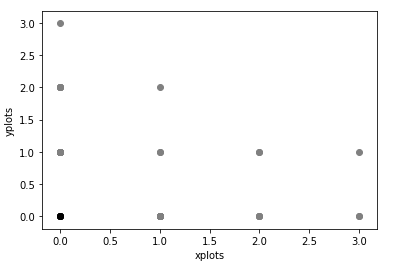
cluster. How many iterations did it take? Show a scatter plot of the data and the centers of clusters

found.

* From the count matrix obtained in the previous step, I have used a loop to go through each row of the data and count the numbers for the first 5 values as positive words and next 5 values as negative values, as shown below,



* The newCluster will contain the dataset cluster which will be passed as X to the myKmeans(X, k, c) method to find the new centers.
* I have implemented another loop to compute the kmeans for values k=2, k=3 and k=4 and computing the kmeans along with the scatter plots.
* All the scatter plots for the k values are similar since there are a lot of vectors with (0,0) values. Because of this the centres are always at (0,0), as shown below,



* We can see that the center is always at (0,0) and the number of iterations taken for all k values is 0.

## External References:

* <https://www.geeksforgeeks.org/python-nlp-analysis-of-restaurant-reviews/>
* <https://stackoverflow.com/questions/41519991/how-to-make-seaborn-heatmap-larger-normal-size>
* <https://stackoverflow.com/questions/16503560/read-specific-columns-from-a-csv-file-with-csv-module>
* <https://www.kaggle.com/muonneutrino/sentiment-analysis-with-amazon-reviews>
* <https://kavita-ganesan.com/extracting-keywords-from-text-tfidf/#.Xl_i1ahKjIX>
* <https://github.com/sagar3122/Data-Mining/blob/master/K-Means/k-means.py/>
* <https://www.bogotobogo.com/python/NLTK/tf_idf_with_scikit-learn_NLTK.php>
* <https://www.kaggle.com/sanikamal/text-classification-with-python-and-keras>
* <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html>
* <https://github.com/mwaskom/seaborn/issues/375>
* <https://www.bogotobogo.com/python/NLTK/tf_idf_with_scikit-learn_NLTK.php>
* <https://towardsdatascience.com/machine-learning-algorithms-part-9-k-means-example-in-python-f2ad05ed5203>
* <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html>
* <https://triton.ml/blog/tf-idf-from-scratch>
* <https://www.pythonprogramming.in/create-document-term-matrix-with-tf-idf.html>
* <https://www.freecodecamp.org/news/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3/>