Assignment - 10.a (Apply clustering on Amazon Food Reviews)

September 11, 2018

1 OBJECTIVE :- Apply clustering on Amazon Food Reviews

In [2]:
```
# Importing libraries
import warnings
warnings.filterwarnings("ignore")

import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from nltk.stem.porter import PorterStemmer

import re

import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import pickle
```

2 Loading Data

In [3]:
```
# using the SQLite Table to read data.
con1 = sqlite3.connect('database.sqlite')
```

```
# Eliminating neutral reviews i.e. those reviews with Score = 3
filtered_data = pd.read_sql_query(" SELECT * FROM Reviews WHERE Score != 3 ", con1)

# Give reviews with Score>3 a positive rating, and reviews with a score<3 a negative r
def polarity(x):
    if x < 3:
        return 'negative'
    return 'positive'

# Applying polarity function on Score column of filtered_data
filtered_data['Score'] = filtered_data['Score'].map(polarity)

print(filtered_data.shape)
filtered_data.head()
```

```
(525814, 10)
```

Out[3]:

	Id	ProductId	UserId	ProfileName	\
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	

	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	\
0	1	1	positive	1303862400	
1	0	0	negative	1346976000	
2	1	1	positive	1219017600	
3	3	3	negative	1307923200	
4	0	0	positive	1350777600	

	Summary	Text
0	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	"Delight" says it all	This is a confection that has been around a fe...
3	Cough Medicine	If you are looking for the secret ingredient i...
4	Great taffy	Great taffy at a great price. There was a wid...

3 Data Cleaning: Deduplication

In [4]:
```
#Sorting data according to ProductId in ascending order
sorted_data=filtered_data.sort_values('ProductId', axis=0, ascending=True, inplace=Fals

#Deduplication of entries
final=sorted_data.drop_duplicates(subset={"UserId","ProfileName","Time","Text"}, keep=
print(final.shape)
```

```
        #Checking to see how much % of data still remains
        ((final.shape[0]*1.0)/(filtered_data.shape[0]*1.0)*100)
```

```
(364173, 10)
```

Out[4]: 69.25890143662969

In [5]:
```
        # Removing rows where HelpfulnessNumerator is greater than HelpfulnessDenominator
        final = final[final.HelpfulnessNumerator <= final.HelpfulnessDenominator]

        print(final.shape)
        final[30:50]
```

```
(364171, 10)
```

Out[5]:

	Id	ProductId	UserId	ProfileName
138683	150501	0006641040	AJ46FKXOVC7NR	Nicholas A Mesiano
138676	150493	0006641040	AMXOPJKV4PPNJ	E. R. Bird "Ramseelbird"
138682	150500	0006641040	A1IJKK6Q1GTEAY	A Customer
138681	150499	0006641040	A3E7R866M94LOC	L. Barker "simienwolf"
476617	515426	141278509X	AB1A5EGHHVA9M	CHelmic
22621	24751	2734888454	A1C298ITT645B6	Hugh G. Pritchard
22620	24750	2734888454	A13ISQV0U9GZIC	Sandikaye
284375	308077	2841233731	A3QD68O22M2XHQ	LABRNTH
157850	171161	7310172001	AFXMWPNS1BLU4	
157849	171160	7310172001	A74C7IARQEM1R	
157833	171144	7310172001	A1V5MY8V9AWUQB	
157832	171143	7310172001	A2SWO6OIW01VPX	
157837	171148	7310172001	A3TFTWTG2CC1GA	
157831	171142	7310172001	A2ZO1AYFVQYG44	
157830	171141	7310172001	AZ4O27OJ4JBZN	
157829	171140	7310172001	ADXXVGRCGQQUO	
157828	171139	7310172001	A13MS1JQG2ADOJ	
157827	171138	7310172001	A13LAE0YTXA11B	
157848	171159	7310172001	A16GY2RCF410DT	
157834	171145	7310172001	A1L8DNQYY69L2Z	

```
157850                                      H. Sandler
157849                                         stucker
157833                       Cheryl Sapper "champagne girl"
157832                                             Sam
157837                                      J. Umphress
157831                               Cindy Rellie "Rellie"
157830  Zhinka Chunmee "gamer from way back in the 70's"
157829                                  Richard Pearlstein
157828                                      C. Perrone
157827                            Dita Vyslouzilova "dita"
157848                                              LB
157834                                       R. Flores

        HelpfulnessNumerator  HelpfulnessDenominator     Score        Time  \
138683                     2                       2  positive   940809600
138676                    71                      72  positive  1096416000
138682                     2                       2  positive  1009324800
138681                     2                       2  positive  1065830400
476617                     1                       1  positive  1332547200
22621                      0                       0  positive  1195948800
22620                      1                       1  negative  1192060800
284375                     0                       0  positive  1345852800
157850                     0                       0  positive  1229385600
157849                     0                       0  positive  1230076800
157833                     0                       0  positive  1244764800
157832                     0                       0  positive  1252022400
157837                     0                       0  positive  1240272000
157831                     0                       0  positive  1254960000
157830                     0                       0  positive  1264291200
157829                     0                       0  positive  1264377600
157828                     0                       0  positive  1265760000
157827                     0                       0  positive  1269216000
157848                     0                       0  positive  1231718400
157834                     0                       0  positive  1243728000

                                                 Summary  \
138683  This whole series is great way to spend time w...
138676  Read it once. Read it twice. Reading Chicken S...
138682                                It Was a favorite!
138681                                 Can't explain why
476617                                The best drink mix
22621                                  Dog Lover Delites
22620                                      made in china
284375                Great recipe book for my babycook
157850                                 Excellent treats
157849                                  Sophie's Treats
157833                      THE BEST healthy dog treat!
157832                   My Alaskan Malamute Loves Them!!
```

```
157837                                   Best treat ever!
157831      my 12 year old maltese has always loved these
157830                  Dogs, Cats, Ferrets all love this
157829                                          5 snouts!
157828                                Best dog treat ever
157827                            Great for puppy training
157848                                             Great!
157834                                     Terrific Treats

                                                    Text
138683  I can remember seeing the show when it aired o...
138676  These days, when a person says, "chicken soup"...
138682  This was a favorite book of mine when I was a ...
138681  This book has been a favorite of mine since I ...
476617  This product by Archer Farms is the best drink...
22621   Our dogs just love them.  I saw them in a pet ...
22620   My dogs loves this chicken but its a product f...
284375  This book is easy to read and the ingredients ...
157850  I have been feeding my greyhounds these treats...
157849  This is one product that my welsh terrier can ...
157833  This is the ONLY dog treat that my Lhasa Apso ...
157832  These liver treas are phenomenal. When i recei...
157837  This was the only treat my dog liked during ob...
157831  No waste , even if she is having a  day when s...
157830  I wanted a treat that was accepted and well li...
157829  My Westie loves these things!  She loves anyth...
157828  This is the only dog treat that my terrier wil...
157827  New puppy loves this, only treat he will pay a...
157848  My dog loves these treats!  We started using t...
157834  This is a great treat which all three of my do...
```

OBSERVATION :- Here books with ProductId - 0006641040 and 2841233731 are also there so we have to remove all these rows with these ProductIds from the data

```
In [6]: final = final[final['ProductId'] != '2841233731']
        final = final[final['ProductId'] != '0006641040']
        final.shape

Out[6]: (364136, 10)
```

4 Text Preprocessing: Stemming, stop-word removal and Lemmatization.

```
In [7]: #set of stopwords in English
        from nltk.corpus import stopwords
        stop = set(stopwords.words('english'))
        words_to_keep = set(('not'))
```

```
stop -= words_to_keep
#initialising the snowball stemmer
sno = nltk.stem.SnowballStemmer('english')

 #function to clean the word of any html-tags
def cleanhtml(sentence):
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, ' ', sentence)
    return cleantext

#function to clean the word of any punctuation or special characters
def cleanpunc(sentence):
    cleaned = re.sub(r'[?|!|\'|"|#]',r'',sentence)
    cleaned = re.sub(r'[.|,|)|(|\|/]',r' ',cleaned)
    return  cleaned
```

In [8]:
```
#Code for removing HTML tags , punctuations . Code for removing stopwords . Code for cl
# also greater than 2 . Code for stemmimg and also to convert them to lowercase letters
i=0
str1=' '
final_string=[]
all_positive_words=[] # store words from +ve reviews here
all_negative_words=[] # store words from -ve reviews here.
s=''
for sent in final['Text'].values:
    filtered_sentence=[]
    #print(sent);
    sent=cleanhtml(sent) # remove HTML tags
    for w in sent.split():
        for cleaned_words in cleanpunc(w).split():
            if((cleaned_words.isalpha()) & (len(cleaned_words)>2)):
                if(cleaned_words.lower() not in stop):
                    s=(sno.stem(cleaned_words.lower())).encode('utf8')
                    filtered_sentence.append(s)
                    if (final['Score'].values)[i] == 'positive':
                        all_positive_words.append(s) #list of all words used to descri
                    if(final['Score'].values)[i] == 'negative':
                        all_negative_words.append(s) #list of all words used to descri
                else:
                    continue
            else:
                continue

    str1 = b" ".join(filtered_sentence) #final string of cleaned words


    final_string.append(str1)
    i+=1
```

In [9]:
```
#adding a column of CleanedText which displays the data after pre-processing of the re
final['CleanedText']=final_string
final['CleanedText']=final['CleanedText'].str.decode("utf-8")
#below the processed review can be seen in the CleanedText Column
print('Shape of final',final.shape)
final.head()
```

```
Shape of final (364136, 11)
```

Out[9]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text	CleanedText
476617	515426	141278509X	AB1A5EGHHVA9M	CHelmic	1	1	positive	1332547200	The best drink mix	This product by Archer Farms is the best drink...	product archer farm best drink mix ever mix fl...
22621	24751	2734888454	A1C298ITT645B6	Hugh G. Pritchard	0	0	positive	1195948800	Dog Lover Delites	Our dogs just love them. I saw them in a pet ...	dog love saw pet store tag attach regard made ...
22620	24750	2734888454	A13ISQV0U9GZIC	Sandikaye	1	1	negative	1192060800	made in china	My dogs loves this chicken but its a product f...	dog love chicken product china wont buy anymor...
157850	171161	7310172001	AFXMWPNS1BLU4	H. Sandler	0	0	positive	1229385600	Excellent treats	I have been feeding my greyhounds these treats...	feed greyhound treat year hound littl finicki ...
157849	171160	7310172001	A74C7IARQEM1R	stucker	0	0	positive	1230076800	Sophie's Treats	This is one product that my welsh terrier can ...	one product welsh terrier eat sophi food alerg...

RANDOMLY SAMPLING 40K POINTS OUT OF WHOLE DATASET

In [10]:
```
##Sorting data according to Time in ascending order for Time Based Splitting
time_sorted_data = final.sort_values('Time', axis=0, ascending=True, inplace=False, ki

# We will collect different 40K rows without repetition from time_sorted_data datafra
my_final = time_sorted_data.take(np.random.permutation(len(final))[:40000])

x = my_final['CleanedText'].values
```

5 Implementing K-Means++ and K-medoids

6 (1). Bag of Words (BoW)

```
In [11]: #BoW
         count_vect = CountVectorizer(min_df = 1000)
         data = count_vect.fit_transform(x)
         print("the type of count vectorizer :",type(data))
         print("the shape of out text BOW vectorizer : ",data.get_shape())
         print("the number of unique words :", data.get_shape()[1])
```

```
the type of count vectorizer : <class 'scipy.sparse.csr.csr_matrix'>
the shape of out text BOW vectorizer :  (40000, 268)
the number of unique words : 268
```

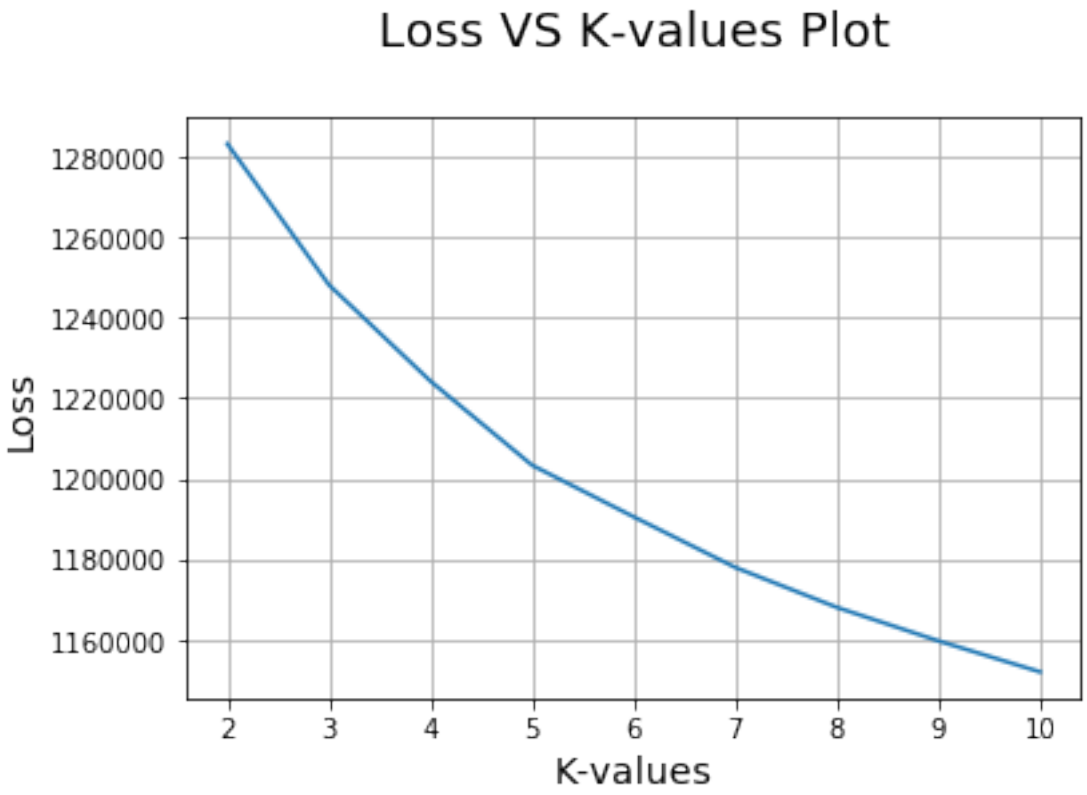
7 (1.a). K-Means++ Implementation

```
In [12]: from sklearn.cluster import KMeans

         k_values = [2,3,4,5,6,7,8,9,10]
         loss = []
         for i in k_values:
             kmeans = KMeans(n_clusters=i, n_jobs=-1).fit(data)
             loss.append(kmeans.inertia_)
```

ELBOW METHOD :

```
In [13]: # Draw Loss VS K values plot
         plt.plot(k_values, loss)
         plt.xlabel('K-values',size=14)
         plt.ylabel('Loss',size=14)
         plt.title('Loss VS K-values Plot\n',size=18)
         plt.grid()
         plt.show()
```



OBSERVATION :- From above we can see that there is inflection at K = 5 . Befor it loss was decreasing faster as compared to the loss decreasing after it . So , the best value of K is 5.

```
In [14]: optimal_k = 5
         # Variable that will be used in the conclusion
         bow_means_k = optimal_k

         # Implementing K-Means++ using optimal value of K
         kmeans = KMeans(n_clusters=optimal_k, n_jobs=-1).fit(data)
```

```
In [15]: reviews = my_final['Text'].values
         # Getting all the reviews in different clusters
         cluster1 = []
         cluster2 = []
         cluster3 = []
         cluster4 = []
         cluster5 = []

         for i in range(kmeans.labels_.shape[0]):
             if kmeans.labels_[i] == 0:
                 cluster1.append(reviews[i])
             elif kmeans.labels_[i] == 1:
                 cluster2.append(reviews[i])
```

```
            elif kmeans.labels_[i] == 2:
                cluster3.append(reviews[i])
            elif kmeans.labels_[i] == 3:
                cluster4.append(reviews[i])
            else :
                cluster5.append(reviews[i])

        # Number of reviews in different clusters
        print("No. of reviews in Cluster-1 : ",len(cluster1))
        print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
        print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
        print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
        print("\nNo. of reviews in Cluster-5 : ",len(cluster5))
```

```
No. of reviews in Cluster-1 :  31882

No. of reviews in Cluster-2 :  5108

No. of reviews in Cluster-3 :  1299

No. of reviews in Cluster-4 :  911

No. of reviews in Cluster-5 :  800
```

READING REVIEWS MANUALLY:

```
In [16]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purcl

Review-2 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more r

Review-3 :
 I will order from here again; love these beans; mom lives in texas and cannot find these where
```

```
In [17]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1
```

```
Review-1 :
 The first sip of Izze's sparkling apple juice was a bit of a shock, albeit a pleasant one.  It

Review-2 :
 This is a great product with a fresh taste.  I am a school teacher and I use this powder every

Review-3 :
 These are the absolute best things I have bought for my dog! We got them when she was a puppy
```

```
In [18]: # Three Reviews of cluster 3
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster3[i]))
             count +=1
```

```
Review-1 :
 I'm going to admit I'm a coffee addict but not a Starbucks fan.  Assuming you like Starbucks c

Review-2 :
 Let me first say that I absolutely LOVE my coffee.<br /><br />The taste is decent, but I would

Review-3 :
 Nice strong coffee - smooth but does have a bit of acidic taste at the end. My favorite is Co
```

```
In [19]: # Three Reviews of cluster 4
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster4[i]))
             count +=1
```

```
Review-1 :
 While this tea is good, its name promised more than the flavor delivered, at least the way I p

Review-2 :
 I am an avid tea drinker who turned to the drink after realizing my body can't really tolerate

Review-3 :
 I have found to really enjoy the flavor of this tea.<br />I would give it 4 1/2 stars.<br />I
```

```
In [20]: # Three Reviews of cluster 5
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster5[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 It took him a few days, but Roofus the Cat came around to having this new type of food in his

Review-3 :
 My son who eats everything, actually refused to eat this baby food.  I got it as a 'spare' for
```

8 (1.b) K-Medoids Implementation

In [22]:
```
from sklearn.metrics.pairwise import pairwise_distances
import kmedoids

data1 = data[0:4000,:]
# distance matrix
D = pairwise_distances(data1, metric='euclidean')

# split into optimal value of clusters
M, C = kmedoids.kMedoids(D, optimal_k)

# Getting the reviews in all clusters
cluster1 = []
cluster2 = []
cluster3 = []
cluster4 = []
cluster5 = []

for label in C:
    for point_idx in C[label]:
        if label == 0 :
            cluster1.append(reviews[point_idx])
        elif label == 1:
            cluster2.append(reviews[point_idx])
        elif label == 2:
            cluster3.append(reviews[point_idx])
        elif label == 3:
            cluster4.append(reviews[point_idx])
        else :
            cluster5.append(reviews[point_idx])

# Number of reviews in different clusters
print("No. of reviews in Cluster-1 : ",len(cluster1))
print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
```

```
        print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
        print("\nNo. of reviews in Cluster-5 : ",len(cluster5))
```

```
No. of reviews in Cluster-1 :  102

No. of reviews in Cluster-2 :  65

No. of reviews in Cluster-3 :  3570

No. of reviews in Cluster-4 :  8

No. of reviews in Cluster-5 :  255
```

READING REVIEWS MANUALLY:

```
In [23]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 It took him a few days, but Roofus the Cat came around to having this new type of food in his

Review-3 :
 I'm a relatively new cat owner and admittedly have not used any other litter but this.<br />I
```

```
In [24]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1
```

```
Review-1 :
 I agree with the reviewer that stated the flavor isn't strong enough, a problem with all of t

Review-2 :
 I tried Switch Orange Tangerine drink.  Excellent flavor and not so sweet as to make you feel

Review-3 :
 I'm usually one to try new coffee flavors, though vanilla is definitely one of my favorites.
```

In [25]:
```
# Three Reviews of cluster 3
count=1
for i in range(3):
    print('Review-%d : \n %s\n'%(count,cluster3[i]))
    count +=1
```

```
Review-1 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purcl

Review-2 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more r

Review-3 :
 I will order from here again; love these beans; mom lives in texas and cannot find these where
```

In [26]:
```
# Three Reviews of cluster 4
count=1
for i in range(3):
    print('Review-%d : \n %s\n'%(count,cluster4[i]))
    count +=1
```

```
Review-1 :
 The roller ball on this treat (basically gravy) does not roll around readily from the friction

Review-2 :
 I have tried lots of treats for my dogs, but these are the only ones they never seem to tire o

Review-3 :
 SO WORTH IT.  And when you do the math, it's actually a bargain.  More on that later...<br /><
```

In [27]:
```
# Three Reviews of cluster 5
count=1
for i in range(3):
    print('Review-%d : \n %s\n'%(count,cluster5[i]))
    count +=1
```

```
Review-1 :
 This is a great product with a fresh taste.  I am a school teacher and I use this powder every

Review-2 :
 I'll get this out of the way first... If you are expecting the quality of a few tenths of an o

Review-3 :
 Let me first say that I absolutely LOVE my coffee.<br /><br />The taste is decent, but I would
```

9 (2) TFIDF

```
In [28]: tf_idf_vect = TfidfVectorizer(min_df=1000)
         data = tf_idf_vect.fit_transform(x)
         print("the type of count vectorizer :",type(data))
         print("the shape of out text TFIDF vectorizer : ",data.get_shape())
         print("the number of unique words :", data.get_shape()[1])
```

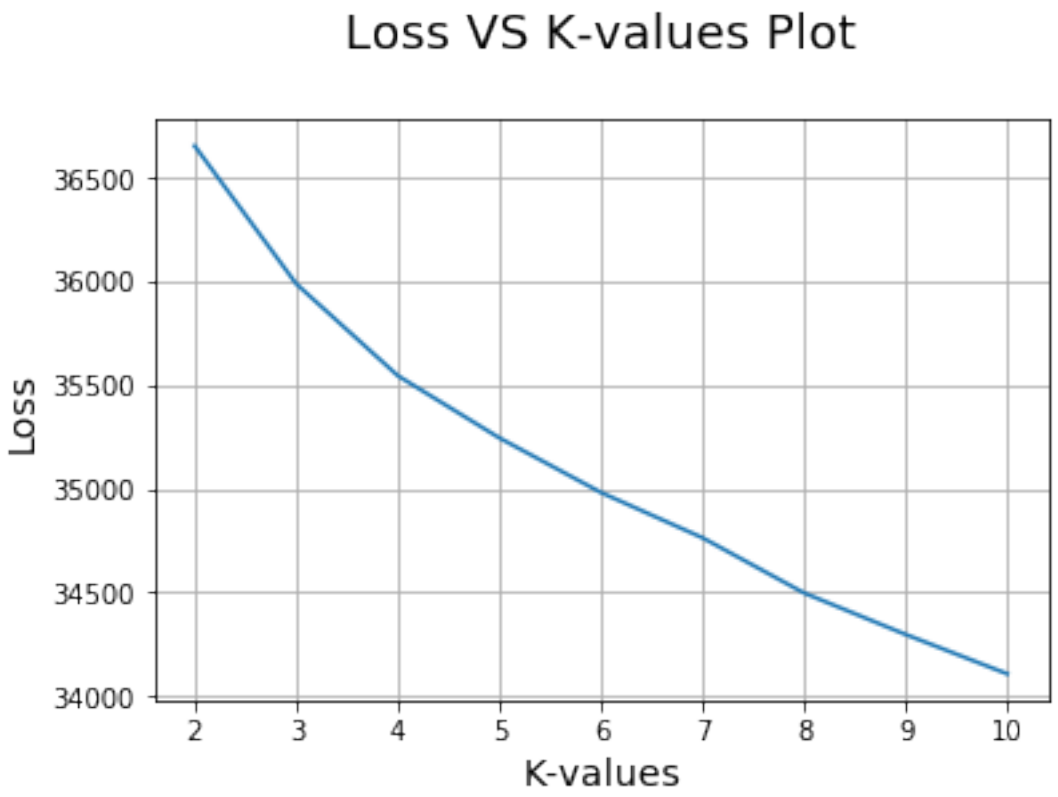
```
the type of count vectorizer : <class 'scipy.sparse.csr.csr_matrix'>
the shape of out text TFIDF vectorizer :  (40000, 268)
the number of unique words : 268
```

10 (2.a). K-Means++ Implementation

```
In [29]: k_values = [2,3,4,5,6,7,8,9,10]
         loss = []
         for i in k_values:
             kmeans = KMeans(n_clusters=i, n_jobs=-1).fit(data)
             loss.append(kmeans.inertia_)
```

ELBOW METHOD :

```
In [30]: # Draw Loss VS K values plot
         plt.plot(k_values, loss)
         plt.xlabel('K-values',size=14)
         plt.ylabel('Loss',size=14)
         plt.title('Loss VS K-values Plot\n',size=18)
         plt.grid()
         plt.show()
```



OBSERVATION :- From above we can see that there is inflection at K = 4 . Befor it loss was decreasing faster as compared to the loss decreasing after it . So , the best value of K is 4.

```
In [31]: optimal_k = 4
         # Variable that will be used in the conclusion
         tfidf_means_k = optimal_k

         # Implementing K-Means++ using optimal value of K
         kmeans = KMeans(n_clusters=optimal_k, n_jobs=-1).fit(data)

         # Getting all the reviews in different clusters
         cluster1 = []
         cluster2 = []
         cluster3 = []
         cluster4 = []

         for i in range(kmeans.labels_.shape[0]):
             if kmeans.labels_[i] == 0:
                 cluster1.append(reviews[i])
             elif kmeans.labels_[i] == 1:
                 cluster2.append(reviews[i])
             elif kmeans.labels_[i] == 2:
```

```
                cluster3.append(reviews[i])
            else :
                cluster4.append(reviews[i])

        # Number of reviews in different clusters
        print("No. of reviews in Cluster-1 : ",len(cluster1))
        print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
        print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
        print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
```

```
No. of reviews in Cluster-1 :  2824

No. of reviews in Cluster-2 :  3355

No. of reviews in Cluster-3 :  3501

No. of reviews in Cluster-4 :  30320
```

READING REVIEWS MANUALLY:

```
In [32]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 I  have  bought  many  cases of  this  before i  dcided  to  check  on the  internet  for it.

Review-2 :
 While this tea is good, its name promised more than the flavor delivered, at least the way I

Review-3 :
 I am an avid tea drinker who turned to the drink after realizing my body can't really tolerat
```

```
In [33]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 These are the absolute best things I have bought for my dog! We got them when she was a puppy
```

```
Review-3 :
 It took him a few days, but Roofus the Cat came around to having this new type of food in his
```

```
In [34]: # Three Reviews of cluster 3
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster3[i]))
             count +=1
```

```
Review-1 :
 I love fruit-flavored coffees, especially blueberry. I also like darker roasts, and unfortunat

Review-2 :
 This is really terrible tasting coffee.  I love Van Houttes vanilla, and Green Mountain Caram

Review-3 :
 I had searched for years for a decaf coffee that kept the wonderful flavor of a full-bodied c
```

```
In [35]: # Three Reviews of cluster 4
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster4[i]))
             count +=1
```

```
Review-1 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purc

Review-2 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more

Review-3 :
 I will order from here again; love these beans; mom lives in texas and cannot find these wher
```

11 (2.b) K-Medoids Implementation

```
In [36]: data1 = data[0:4000,:]
         # distance matrix
         D = pairwise_distances(data1, metric='euclidean')

         # split into optimal value of  clusters
         M, C = kmedoids.kMedoids(D, optimal_k)
```

```
        # Getting the reviews in all clusters
        cluster1 = []
        cluster2 = []
        cluster3 = []
        cluster4 = []

        for label in C:
            for point_idx in C[label]:
                if label == 0 :
                    cluster1.append(reviews[point_idx])
                elif label == 1:
                    cluster2.append(reviews[point_idx])
                elif label == 2:
                    cluster3.append(reviews[point_idx])
                else :
                    cluster4.append(reviews[point_idx])

        # Number of reviews in different clusters
        print("No. of reviews in Cluster-1 : ",len(cluster1))
        print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
        print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
        print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
```

```
No. of reviews in Cluster-1 :  1883

No. of reviews in Cluster-2 :  372

No. of reviews in Cluster-3 :  316

No. of reviews in Cluster-4 :  1429
```

READING REVIEWS MANUALLY:

```
In [37]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more r

Review-3 :
 The product was really good and arrived fairly quickly. The kicker was, the sample they sent a
```

```
In [38]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1

Review-1 :
 I will order from here again; love these beans; mom lives in texas and cannot find these where

Review-2 :
 These are the absolute best things I have bought for my dog! We got them when she was a puppy

Review-3 :
 I will definetly be ordering another shipment of this sweet and tasty (yet low calorie) popcor
```

```
In [39]: # Three Reviews of cluster 3
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster3[i]))
             count +=1

Review-1 :
 I just got these today and they had almost all leaked. Every box was soaked with juice. Instea

Review-2 :
 I'm going to admit I'm a coffee addict but not a Starbucks fan.  Assuming you like Starbucks c

Review-3 :
 excellent<a href="http://www.amazon.com/gp/product/B000CSEFQ0">Kellogg's Cereal in a Cup, Favo
```

```
In [40]: # Three Reviews of cluster 4
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster4[i]))
             count +=1

Review-1 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purch

Review-2 :
 I love fruit-flavored coffees, especially blueberry. I also like darker roasts, and unfortunat
```

```
Review-3 :
 The first sip of Izze's sparkling apple juice was a bit of a shock, albeit a pleasant one.  It
```

12 Word2Vec

```
In [41]: # List of sentence in X_train text
         sent_x = []
         for sent in x :
             sent_x.append(sent.split())


         # Train your own Word2Vec model using your own train text corpus
         # min_count = 5 considers only words that occured atleast 5 times
         w2v_model=Word2Vec(sent_x,min_count=5,size=50, workers=4)

         w2v_words = list(w2v_model.wv.vocab)
         print("number of words that occured minimum 5 times ",len(w2v_words))
```

```
number of words that occured minimum 5 times  8576
```

13 (3). Avg Word2Vec

```
In [42]: # compute average word2vec for each review for X_train .
         train_vectors = [];
         for sent in sent_x:
             sent_vec = np.zeros(50)
             cnt_words =0;
             for word in sent: #
                 if word in w2v_words:
                     vec = w2v_model.wv[word]
                     sent_vec += vec
                     cnt_words += 1
             if cnt_words != 0:
                 sent_vec /= cnt_words
             train_vectors.append(sent_vec)

         data = train_vectors
```

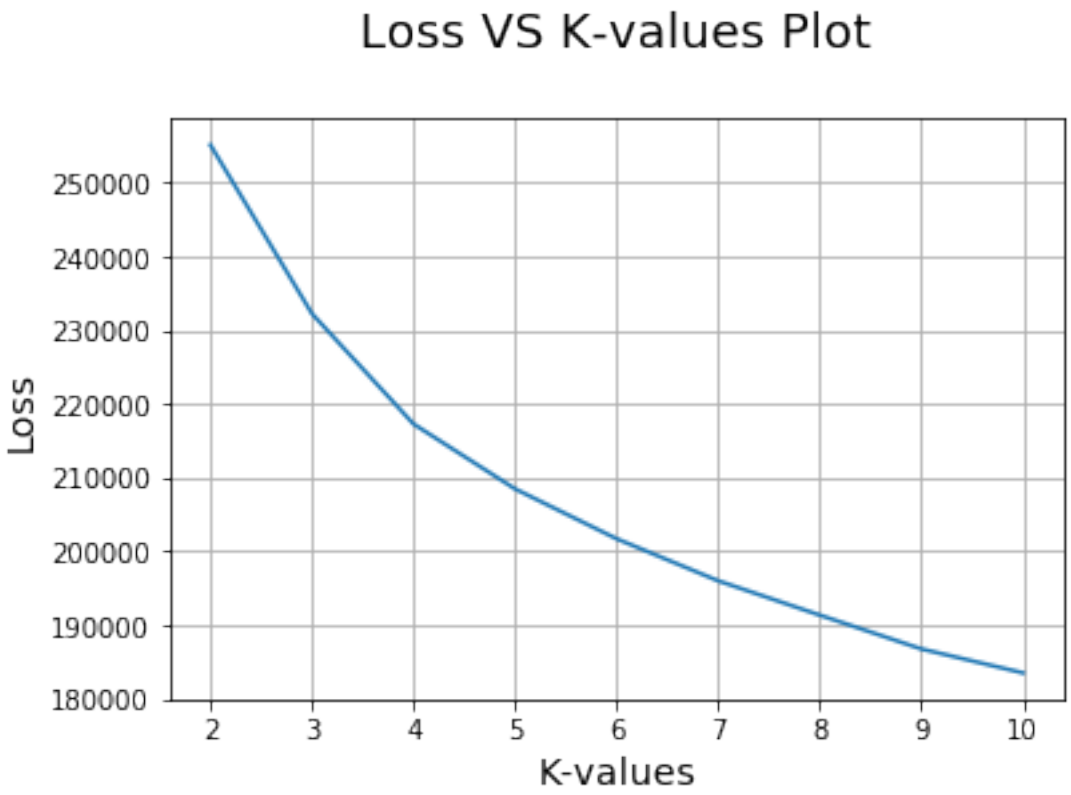
14 (3.a). K-Means++ Implementation

```
In [43]: k_values = [2,3,4,5,6,7,8,9,10]
         loss = []
         for i in k_values:
```

```
        kmeans = KMeans(n_clusters=i, n_jobs=-1).fit(data)
        loss.append(kmeans.inertia_)
```

ELBOW METHOD :

```
In [44]: # Draw Loss VS K values plot
         plt.plot(k_values, loss)
         plt.xlabel('K-values',size=14)
         plt.ylabel('Loss',size=14)
         plt.title('Loss VS K-values Plot\n',size=18)
         plt.grid()
         plt.show()
```



OBSERVATION :- From above we can see that there is inflection at K = 4 . Befor it loss was decreasing faster as compared to the loss decreasing after it . So , the best value of K is 4.

```
In [50]: optimal_k = 4
         # Variable that will be used in the conclusion
         avg_w2v_means_k = optimal_k

         # Implementing K-Means++ using optimal value of K
         kmeans = KMeans(n_clusters=optimal_k, n_jobs=-1).fit(data)
```

```
        # Getting all the reviews in different clusters
        cluster1 = []
        cluster2 = []
        cluster3 = []
        cluster4 = []

        for i in range(kmeans.labels_.shape[0]):
            if kmeans.labels_[i] == 0:
                cluster1.append(reviews[i])
            elif kmeans.labels_[i] == 1:
                cluster2.append(reviews[i])
            elif kmeans.labels_[i] == 2:
                cluster3.append(reviews[i])
            else :
                cluster4.append(reviews[i])

        # Number of reviews in different clusters
        print("No. of reviews in Cluster-1 : ",len(cluster1))
        print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
        print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
        print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
```

```
No. of reviews in Cluster-1 :  17085

No. of reviews in Cluster-2 :  6577

No. of reviews in Cluster-3 :  6033

No. of reviews in Cluster-4 :  10305
```

READING REVIEWS MANUALLY :

```
In [51]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purch

Review-2 :
 Good sack o' nuts with a nice twist of cocoa. Wish it was more 'clustery' but eh. Nice snack t

Review-3 :
 this is very tasty popcorn and is great for smaller children because it pops with almost no sh
```

```
In [52]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1
```

```
Review-1 :
 I love fruit-flavored coffees, especially blueberry. I also like darker roasts, and unfortunat

Review-2 :
 This is really terrible tasting coffee.  I love Van Houttes vanilla, and Green Mountain Caram

Review-3 :
 I had searched for years for a decaf coffee that kept the wonderful flavor of a full-bodied c
```

```
In [53]: # Three Reviews of cluster 3
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster3[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 I keep these at my desk because I get so busy sometimes I have to skip lunch and these are pe

Review-3 :
 These are the absolute best things I have bought for my dog! We got them when she was a puppy
```

```
In [54]: # Three Reviews of cluster 4
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster4[i]))
             count +=1
```

```
Review-1 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more

Review-2 :
 I will order from here again; love these beans; mom lives in texas and cannot find these where

Review-3 :
 The product was really good and arrived fairly quickly. The kicker was, the sample they sent
```

15 (3.b) K-Medoids Implementation

```
In [55]: data1 = data[0:4000]
         # distance matrix
         D = pairwise_distances(data1, metric='euclidean')

         # split into optimal value of  clusters
         M, C = kmedoids.kMedoids(D, optimal_k)

         # Getting the reviews in all clusters
         cluster1 = []
         cluster2 = []
         cluster3 = []
         cluster4 = []

         for label in C:
             for point_idx in C[label]:
                 if label == 0 :
                     cluster1.append(reviews[point_idx])
                 elif label == 1:
                     cluster2.append(reviews[point_idx])
                 elif label == 2:
                     cluster3.append(reviews[point_idx])
                 else :
                     cluster4.append(reviews[point_idx])

         # Number of reviews in different clusters
         print("No. of reviews in Cluster-1 : ",len(cluster1))
         print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
         print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
         print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
```

```
No. of reviews in Cluster-1 :  1189

No. of reviews in Cluster-2 :  1055

No. of reviews in Cluster-3 :  918

No. of reviews in Cluster-4 :  838
```

READING REVIEWS MANUALLY :

```
In [56]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purcl

Review-2 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more r

Review-3 :
 I will order from here again; love these beans; mom lives in texas and cannot find these where
```

```
In [57]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1
```

```
Review-1 :
 this is a good as it gets next to starbucks and since it is in your own home, it is even bette

Review-2 :
 I love fruit-flavored coffees, especially blueberry. I also like darker roasts, and unfortunat

Review-3 :
 The first sip of Izze's sparkling apple juice was a bit of a shock, albeit a pleasant one.  It
```

```
In [58]: # Three Reviews of cluster 3
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster3[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 This is my first 1 star review...I can't believe I'm writing it.<br /><br />I wanted to like t

Review-3 :
 I love the sweet potato fries so when i seen these figured i'd give them a try. they are so yu
```

```
In [59]: # Three Reviews of cluster 4
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster4[i]))
             count +=1
```

```
Review-1 :
 Good sack o' nuts with a nice twist of cocoa. Wish it was more 'clustery' but eh. Nice snack t

Review-2 :
 this is very tasty popcorn and is great for smaller children because it pops with almost no sh

Review-3 :
 This sunflower seed butter has a nice roasted taste & is very creamy.  The consistency is actu
```

16 (4). TFIDF-Word2Vec

```
In [60]: # TF-IDF weighted Word2Vec
         tf_idf_vect = TfidfVectorizer()

         # final_tf_idf1 is the sparse matrix with row= sentence, col=word and cell_val = tfid
         final_tf_idf1 = tf_idf_vect.fit_transform(x)

         # tfidf words/col-names
         tfidf_feat = tf_idf_vect.get_feature_names()

         # compute TFIDF Weighted Word2Vec for each review for X_test .
         tfidf_vectors = [];
         row=0;
         for sent in sent_x:
             sent_vec = np.zeros(50)
             weight_sum =0;
             for word in sent:
                 if word in w2v_words:
                     vec = w2v_model.wv[word]
                     # obtain the tf_idfidf of a word in a sentence/review
                     tf_idf = final_tf_idf1[row, tfidf_feat.index(word)]
                     sent_vec += (vec * tf_idf)
                     weight_sum += tf_idf
             if weight_sum != 0:
                 sent_vec /= weight_sum
             tfidf_vectors.append(sent_vec)
             row += 1

         data = tfidf_vectors
```

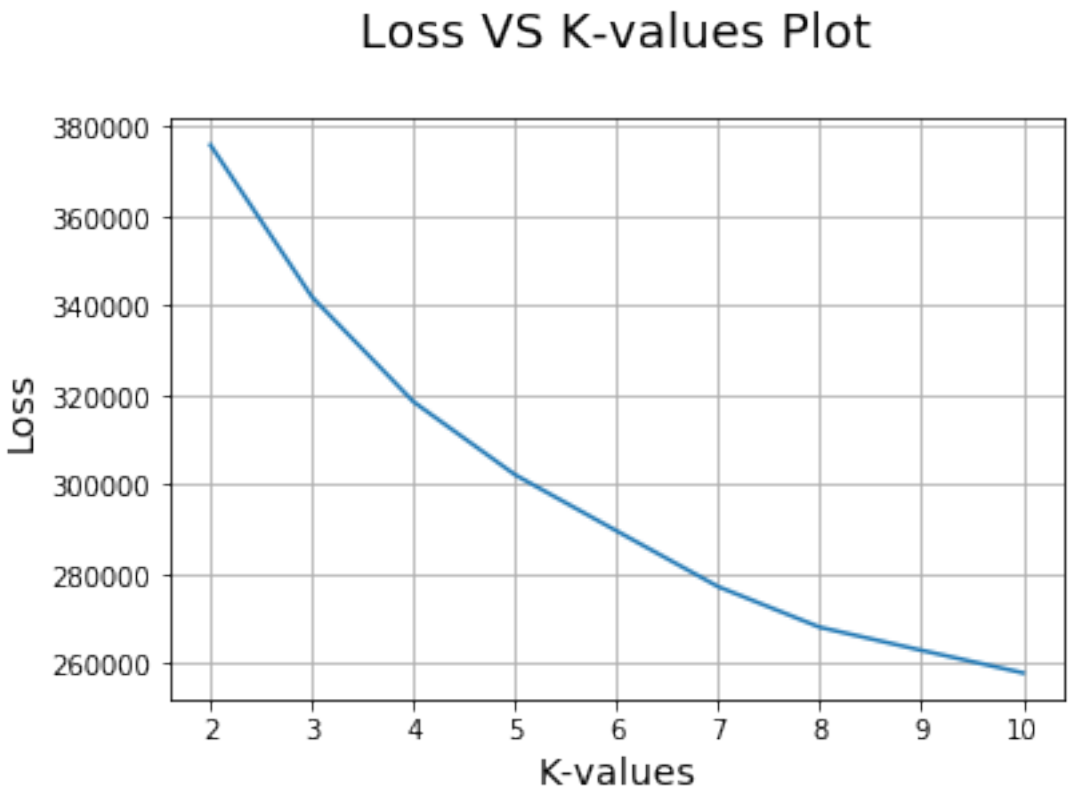
17 (4.a). K-Means++ Implementation

```
In [61]: k_values = [2,3,4,5,6,7,8,9,10]
         loss = []
         for i in k_values:
```

```
        kmeans = KMeans(n_clusters=i, n_jobs=-1).fit(data)
        loss.append(kmeans.inertia_)
```

ELBOW METHOD :

```
In [62]: # Draw Loss VS K values plot
         plt.plot(k_values, loss)
         plt.xlabel('K-values',size=14)
         plt.ylabel('Loss',size=14)
         plt.title('Loss VS K-values Plot\n',size=18)
         plt.grid()
         plt.show()
```



OBSERVATION :- From above we can see that there is inflection at K = 4 . Befor it loss was decreasing faster as compared to the loss decreasing after it . So , the best value of K is 4.

```
In [63]: optimal_k = 4
         # Variable that will be used in the conclusion
         tfidf_w2v_means_k = optimal_k

         # Implementing K-Means++ using optimal value of K
         kmeans = KMeans(n_clusters=optimal_k, n_jobs=-1).fit(data)
```

```
        # Getting all the reviews in different clusters
        cluster1 = []
        cluster2 = []
        cluster3 = []
        cluster4 = []

        for i in range(kmeans.labels_.shape[0]):
            if kmeans.labels_[i] == 0:
                cluster1.append(reviews[i])
            elif kmeans.labels_[i] == 1:
                cluster2.append(reviews[i])
            elif kmeans.labels_[i] == 2:
                cluster3.append(reviews[i])
            else :
                cluster4.append(reviews[i])

        # Number of reviews in different clusters
        print("No. of reviews in Cluster-1 : ",len(cluster1))
        print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
        print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
        print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
```

```
No. of reviews in Cluster-1 :  5383

No. of reviews in Cluster-2 :  4095

No. of reviews in Cluster-3 :  13209

No. of reviews in Cluster-4 :  17313
```

READING REVIEWS MANUALLY :

```
In [64]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 This is really terrible tasting coffee.  I love Van Houttes vanilla, and Green Mountain Caram

Review-2 :
 I had searched for years for a decaf coffee that kept the wonderful flavor of a full-bodied c

Review-3 :
 While this tea is good, its name promised more than the flavor delivered, at least the way I
```

```
In [65]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 These are the absolute best things I have bought for my dog! We got them when she was a puppy

Review-3 :
 It took him a few days, but Roofus the Cat came around to having this new type of food in his
```

```
In [66]: # Three Reviews of cluster 3
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster3[i]))
             count +=1
```

```
Review-1 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more

Review-2 :
 I will order from here again; love these beans; mom lives in texas and cannot find these where

Review-3 :
 The product was really good and arrived fairly quickly. The kicker was, the sample they sent a
```

```
In [67]: # Three Reviews of cluster 4
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster4[i]))
             count +=1
```

```
Review-1 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purch

Review-2 :
 Good sack o' nuts with a nice twist of cocoa. Wish it was more 'clustery' but eh. Nice snack t

Review-3 :
 this is very tasty popcorn and is great for smaller children because it pops with almost no sh
```

18 (4.b) K-Medoids Implementation

```
In [68]: data1 = data[0:4000]
         # distance matrix
         D = pairwise_distances(data1, metric='euclidean')

         # split into optimal value of  clusters
         M, C = kmedoids.kMedoids(D, optimal_k)

         # Getting the reviews in all clusters
         cluster1 = []
         cluster2 = []
         cluster3 = []
         cluster4 = []

         for label in C:
             for point_idx in C[label]:
                 if label == 0 :
                     cluster1.append(reviews[point_idx])
                 elif label == 1:
                     cluster2.append(reviews[point_idx])
                 elif label == 2:
                     cluster3.append(reviews[point_idx])
                 else :
                     cluster4.append(reviews[point_idx])

         # Number of reviews in different clusters
         print("No. of reviews in Cluster-1 : ",len(cluster1))
         print("\nNo. of reviews in Cluster-2 : ",len(cluster2))
         print("\nNo. of reviews in Cluster-3 : ",len(cluster3))
         print("\nNo. of reviews in Cluster-4 : ",len(cluster4))
```

```
No. of reviews in Cluster-1 :  967

No. of reviews in Cluster-2 :  354

No. of reviews in Cluster-3 :  1546

No. of reviews in Cluster-4 :  1133
```

READING REVIEWS MANUALLY :

```
In [69]: # Three Reviews of cluster 1
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster1[i]))
             count +=1
```

```
Review-1 :
 I will order from here again; love these beans; mom lives in texas and cannot find these where

Review-2 :
 The product was really good and arrived fairly quickly. The kicker was, the sample they sent a

Review-3 :
 this is a good as it gets next to starbucks and since it is in your own home, it is even bette
```

```
In [70]: # Three Reviews of cluster 2
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster2[i]))
             count +=1
```

```
Review-1 :
 I love fruit-flavored coffees, especially blueberry. I also like darker roasts, and unfortunat

Review-2 :
 I had searched for years for a decaf coffee that kept the wonderful flavor of a full-bodied co

Review-3 :
 While this tea is good, its name promised more than the flavor delivered, at least the way I p
```

```
In [71]: # Three Reviews of cluster 3
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster3[i]))
             count +=1
```

```
Review-1 :
 I have three cats, all with specific dietary needs. Evo is the only food that meets all their

Review-2 :
 Love this stuff...great flavor...you can't get more authentic Mexican. Don't hesitate to purch

Review-3 :
 I used to grow my own garden of Collard Greens, and ate them regularly.  Since I travel more r
```

```
In [72]: # Three Reviews of cluster 4
         count=1
         for i in range(3):
             print('Review-%d : \n %s\n'%(count,cluster4[i]))
             count +=1
```

```
Review-1 :
 Good sack o' nuts with a nice twist of cocoa. Wish it was more 'clustery' but eh. Nice snack t

Review-2 :
 this is very tasty popcorn and is great for smaller children because it pops with almost no sh

Review-3 :
 I love the sweet potato fries so when i seen these figured i'd give them a try. they are so yu
```

19 CONCLUSION :-

20 (a). Procedure Followed :

STEP 1 :- Text Preprocessing

STEP 2 :- Taking all text data and ignoring class variable .

STEP 3:- Training the vectorizer on text_data and later applying same vectorizer on text_data to transform it into vectors

STEP 4:- Applying Elbow Method using K-means++ in order to find optimal value of K(i.e. number of clusters)

STEP 5:- Draw loss VS K-values plot

STEP 6:- Once we find optimal value of K then again train K-Means++ and K-medoids for clustering text_data into K clusters .

STEP 7:- Reading reviews manually for each cluster

Repeat from STEP 3 to STEP 7 for each of these four vectorizers : Bag Of Words(BoW), TFIDF, Avg Word2Vec and TFIDF Word2Vec

21 (b). Table (Model with their K values) :

```
In [73]: # Creating table using PrettyTable library
         from prettytable import PrettyTable

         names = ['K-means++ for BoW','K-medoids for BoW','K-means++ for TFIDF','K-medoids for
                  'K-medoids for Avg_Word2Vec','K-means++ for tfidf_Word2Vec','K-medoids for ti

         optimal_k = [bow_means_k,bow_means_k,tfidf_means_k,tfidf_means_k,\
                      avg_w2v_means_k,avg_w2v_means_k,tfidf_w2v_means_k,tfidf_w2v_means_k]

         numbering = [1,2,3,4,5,6,7,8]

         # Initializing prettytable
         table = PrettyTable()

         # Adding columns
         table.add_column("S.NO.",numbering)
```

```
table.add_column("MODEL",names)
table.add_column("Number of Clusters ",optimal_k)

# Printing the Table
print(table)
```

```
+-------+------------------------------+----------------------+
| S.NO. |            MODEL             | Number of Clusters   |
+-------+------------------------------+----------------------+
|   1   |      K-means++ for BoW       |          5           |
|   2   |      K-medoids for BoW       |          5           |
|   3   |     K-means++ for TFIDF      |          4           |
|   4   |     K-medoids for TFIDF      |          4           |
|   5   |  K-means++ for Avg_Word2Vec  |          4           |
|   6   |  K-medoids for Avg_Word2Vec  |          4           |
|   7   | K-means++ for tfidf_Word2Vec |          4           |
|   8   | K-medoids for tfidf_Word2Vec |          4           |
+-------+------------------------------+----------------------+
```