Clustering

June 6, 2018

0.0.1 Clustering for Amazon food reviews:

```
In [3]: #importing required Modules
        %matplotlib inline
        import sqlite3
        import pandas as pd
        import numpy as np
        import nltk
        import string
        import pickle
        import matplotlib.pyplot as plt
        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
        from sklearn.preprocessing import StandardScaler
        from sklearn.model_selection import TimeSeriesSplit
        from sklearn.model_selection import GridSearchCV
        from sklearn.model_selection import RandomizedSearchCV
        from sklearn.cluster import KMeans
        import warnings
        warnings.filterwarnings('ignore')
```

```
In [4]: #getting cleaned data from db
        conn = sqlite3.connect('final_clean_LR.sqlite')
        final_review = pd.read_sql_query("""
        SELECT *
        FROM Reviews_final
        """, conn)
```

```
In [5]: #SORT by time for TBS
        final_review = final_review.sort_values(by='Time')
```

```
In [4]: final_review.drop('Score',axis=1,inplace=True)
```

```
In [6]: #info of data
        final_review.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 364171 entries, 23 to 345187
Data columns (total 15 columns):
level_0                   364171 non-null int64
index                     364171 non-null int64
Id                        364171 non-null int64
ProductId                 364171 non-null object
UserId                    364171 non-null object
ProfileName               364171 non-null object
HelpfulnessNumerator      364171 non-null int64
HelpfulnessDenominator    364171 non-null int64
Score                     364171 non-null object
Time                      364171 non-null int64
Summary                   364171 non-null object
Text                      364171 non-null object
CleanedTextBow            364171 non-null object
final_text                364171 non-null object
final_stem_text           364171 non-null object
dtypes: int64(6), object(9)
memory usage: 44.5+ MB
```

```
In [7]: #Converting to int8
        final_review.HelpfulnessNumerator = final_review.\
                              HelpfulnessNumerator.astype(np.int8)
        final_review.HelpfulnessDenominator = final_review.\
                              HelpfulnessDenominator.astype(np.int8)
```

```
In [8]: train_df = final_review.iloc[:round(final_review.shape[0]*0.70),:]
        test_df = final_review.iloc[round(final_review.shape[0]*0.70):,:]
```

K-Means

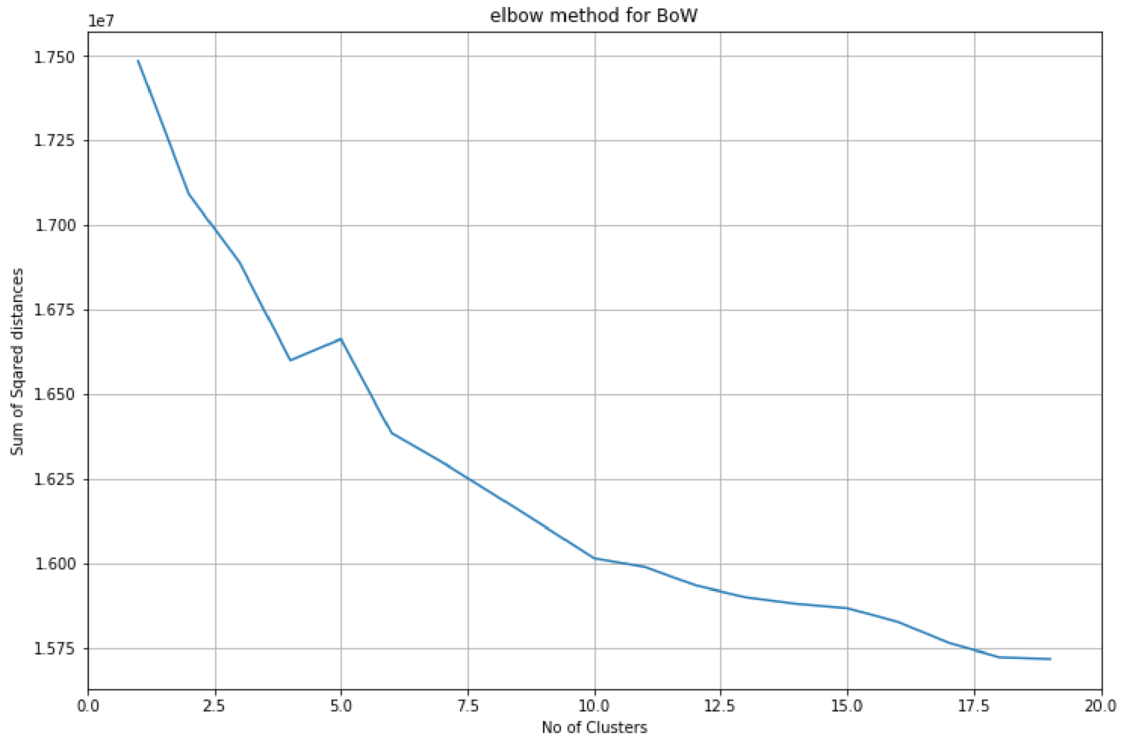
Bag of Words

```
In [8]: #BoW with cleaned data and without stopwords
        #simple cv for train data
        scores_train = []
        from nltk.corpus import stopwords
        stop = set(stopwords.words('english'))
        stop.remove('not')
        stop.remove('very')
        #CountVectorizer for BoW
        count_vect = CountVectorizer(stop_words=list(stop),dtype=np.int8)
        #X_test_cv = train_df.iloc[round(train_df.shape[0]*0.70):,:]
        final_counts_train = count_vect.fit_transform(
                train_df['final_text'].values)
        final_counts_test = count_vect.transform(
                test_df['final_text'].values)
```

```
In [11]: ssd = {}
         centers = {}
         for k in range(1,31):
             model = KMeans(n_clusters=k,n_init=10,max_iter=800,random_state=25,n_jobs=-1)
             model.fit(final_counts_train)
             ssd[k] = model.inertia_
             centers[k] = model.cluster_centers_
             print('No of clusters',k,'Sum of Squared dist',model.inertia_)
```

```
No of clusters 2 Sum of Squared dist 17090972.935039584
No of clusters 3 Sum of Squared dist 16889129.80389441
No of clusters 4 Sum of Squared dist 16599901.618375564
No of clusters 5 Sum of Squared dist 16662864.771436755
No of clusters 6 Sum of Squared dist 16384658.648390105
No of clusters 7 Sum of Squared dist 16298392.490236422
No of clusters 8 Sum of Squared dist 16205244.435071379
No of clusters 9 Sum of Squared dist 16110676.756141247
No of clusters 10 Sum of Squared dist 16014505.905228948
No of clusters 11 Sum of Squared dist 15988800.120337114
No of clusters 12 Sum of Squared dist 15934268.485165115
No of clusters 13 Sum of Squared dist 15899301.555551339
No of clusters 14 Sum of Squared dist 15880090.2596671
No of clusters 15 Sum of Squared dist 15867199.892136786
No of clusters 16 Sum of Squared dist 15826427.815612264
No of clusters 17 Sum of Squared dist 15764300.773115376
No of clusters 18 Sum of Squared dist 15722229.345665809
No of clusters 19 Sum of Squared dist 15717216.974098468
No of clusters 20 Sum of Squared dist 15722633.094975237
No of clusters 21 Sum of Squared dist 15681793.791078253
No of clusters 22 Sum of Squared dist 15655262.02774996
No of clusters 23 Sum of Squared dist 15622175.54597837
No of clusters 24 Sum of Squared dist 15573855.17671994
No of clusters 25 Sum of Squared dist 15564228.805704951
No of clusters 26 Sum of Squared dist 15552663.195803063
No of clusters 27 Sum of Squared dist 15544752.867035514
No of clusters 28 Sum of Squared dist 15471454.512619289
No of clusters 29 Sum of Squared dist 15464379.974905053
No of clusters 30 Sum of Squared dist 15455837.630540872
```

```
In [229]: plt.figure(figsize=(12,8))
          plt.plot(list(ssd.keys()),list(ssd.values()))
          plt.xlim(0,20)
          plt.xlabel('No of Clusters')
          plt.ylabel('Sum of Sqared distances')
          plt.title('elbow method for BoW')
          plt.grid()
```



from above we can observe that 10 cluster are good

```
In [9]: model_bow = KMeans(n_clusters=10,n_init=10,random_state=25)
        model_bow.fit(final_counts_train)

In [233]: #labes for all the data in train_df
          model_bow.labels_

Out[233]: array([0, 0, 0, ..., 0, 0, 2], dtype=int32)

In [234]: train_df['bow_pred_label'] = model_bow.labels_

In [235]: #no of points in each group
          train_df.groupby('bow_pred_label')['final_text','ProductId'].count()

Out[235]:                 final_text  ProductId
          bow_pred_label
          0                   142764     142764
          1                     9029       9029
          2                    14666      14666
          3                     2027       2027
          4                    22661      22661
          5                     5917       5917
          6                     3289       3289
          7                    10456      10456
          8                     1222       1222
          9                    42889      42889
```

```
In [236]: grp = train_df.groupby('bow_pred_label')[['final_text','ProductId']]
```

```
In [131]: print('group 0')
          for i in grp.get_group(0).sample(5).values:
              print(i[0])
              print(i[1])
              print()
```

```
group 0
great product bad price but why would pay amazon twice what they charge brick and mortar store
B000NN5GU8

this awesome this better than peanut butter hand down it only got one ingredient and thats orga
B002OK2FCA

little guy these are perfect for little guy they are the perfect size for him and love that the
B003GBHX5K

awesome popcorn have been ordering this popcorn from bob red mill for several year and great al
B004VLV6ES

too good there huge problem with these cracker cant limit myself cant describe the taste except
B002ZJMSKO
```

```
In [132]: print('group 1')
          for i in grp.get_group(1).sample(5).values:
              print(i[0])
              print(i[1])
              print()
```

```
group 1
great dog food great price and very good food for our dog her coat shiny bad breath and she lov
B0009X63VS

great price great ingredient searched the dog food rating page and found this one the best for
B0018CE6DQ

great treat dog loved this treat wouldnt recommend for large dog mine lb and she finished about
B000RI40UQ

aggressive chewer semi satisfied saying month lab puppy aggressive chewer understatement she tr
B0061PPLYI

garlic juice spray recently ordered this product because wa told would help control flea dog re
B002G8EJSI
```

```
In [134]: print('group 2')
          for i in grp.get_group(2).sample(5).values:
              print(i[0])
              print(i[1])
              print()
```

```
group 2
alkaline tea for all who have acid reflux Gerd great tasting alkaline decaf tea which should us
B000CMIZ0I

garbage purchased the december and returned back bus wholesale club the have very little coffee
B003WAX3H2

awesome flavor love this coffee the morning noon night doe not have too much coconut flavor but
B003G52BN0

disappointing you may wondering what disappointing about four star coffee this better than aver
B006N3HYYS

coffee Folgers instant coffee come jar which very convenient because coffee not loos it freshne
B001EQ4F14
```

```
In [138]: print('group 3')
          for i in grp.get_group(3).sample(5).values:
              print(i[0])
              print(i[1])
              print()
```

```
group 3
great coffee decided try this coffee whim since the town college town that ha very limited coff
B003OP558A

Dumm bought this local grocery store use new coffee espresso machine big coffee nut even roast
B001E5DYTE

kick Starbucks butt ive never been huge fan the overpriced coffee Starbucks even the minority i
B0024KGQJI

timothy world coffee pack first all favorite cup the timothy colombian decaffeinated you are lo
B002AQ0OW6

great coffee and price have been purchasing this brand coffee for approximately year now first
B001E95KLK
```

```
In [144]: print('group 4')
          for i in grp.get_group(4).sample(5).values:
            print(i[0])
            print(i[1])
            print()

group 4
work expected after month chasing two dog out the garden and ruining few plant ordered the fron
B000HHO9EE

soft and yummy wa hesitant give these try because they are expensive but after reading the ingr
B004GYFK9M

magic indeed the aroma upon opening the lentil wa intense and appealing wa expecting one spice
B0043OU7LG

rare treat from the black sesame porridge first discovered black sesame porridge when wa taiwar
B000LQPLOI

big hit senior doxie wa extremely picky about treat begin with then she wa diagnosed with renal
B002R8J7YS
```

```
In [148]: print('group 5')
          for i in grp.get_group(5).sample(5).values:
              print(i[0])
              print(i[1])
              print()

group 5
were these review written dale corporate short and girlfriend sent bottle for christmas she lik
B001EO5YR0

nice treat but substitute moderate drinker both soda and juice wa intrigued the promise the swi
B001LG940E

wonderful fresh herb received aerogarden two and half week ago and thrilled with the only reaso
B000FI4O90

from blog picked package accelerate hydro today accelerate the direct competitor favorite drink
B003CN5NPE

flavor not for hesitate give bad review especially when it some type food and there the whole t
B005A1LINC
```

```
In [150]: print('group 6')
          for i in grp.get_group(6).sample(5).values:
              print(i[0])
```

```
            print(i[1])
            print()
```

```
group 6
 not normally one for writing review since normally other people cover own concern praise thei
B001BOVE54

the worst part cat day now the best have struggled for over year and half giving cat daily pil
B000JOE224

impressive aroma taste very impressive aroma fresh bit wild impressive taste impressive ingred
B001EGZKH2

BPA lid bought this line baby food because wa organic and came glass jar prefer make own baby
B001BM8SS2

made kitty into addict have three cat the two older one are huge dish male litter mate who hav
B001BCVY4W
```

```
In [151]: print('group 7')
          for i in grp.get_group(7).sample(5).values:
              print(i[0])
              print(i[1])
              print()
```

```
group 7
deliciously cool this second purchase this tea this year and love ha such crisp clean cool tas
B001ELLB20

great tea this wonderful smooth and flavorful tea love this stuff and when serve guest they ra
B000SATIKK

real cup tea this the only decaf tea the dozen have tried that make real cup tea strong with t
B002M3QZKM

good price for loose leaf tea used get mine from english tea store that ha recloseable ziplock
B000SAPXF4

great tea great price fast shipping love this tea cost much le than most other brand ive seen
B001F10XUU
```

```
In [153]: print('group 8')
          for i in grp.get_group(8).sample(5).values:
              print(i[0])
```

```
            print(i[1])
            print()

group 8
fine camomile tea lazos calm herbal infusion nice chamomile tea taste the chamomile and some fl
B004EDFLPS

rather splendid keen blend this blend tea from few different regional plantation overall the ta
B000F4DK9Y

this tea yummy ive been drinking the republic tea strawberry chocolate tea for almost whole yea
B00478L5T6

really nice blend smooth tasty satisfying recommended our local supermarket phasing out their (
B0000DBN1Q

make great iced tea ive been using this tea for for about month and drink iced tea daily micro
B001EHDMY4
```

```
In [241]: print('group 9')
          for i in grp.get_group(9).sample(5).values:
              print(i[0])
              print(i[1])
              print()

group 9
great buy husband diabetic and this ha carbs which are sugar not too bad for snack have many pr
B001M08XS8

not recommended the manufacturer but work anyway although sugar floss not recommended for use t
B000UYICMO

quality product thats surpassed expectation chose this product because it shipped quickly wonde
B00029F5E0

didnt find all that jazzy have admit that when first tried this one wasnt crazy about seemed la
B001D0GVAO

sriracha discovered this tasty condiment the asian food section the grocery store maybe wa the
B001E05ZHO
```

0 - Some Miscellaneous food products and reviews size are small
1 - Dog food related produtc
2 - coffee related products with small reviws
3- Coffe related produts with high reviews

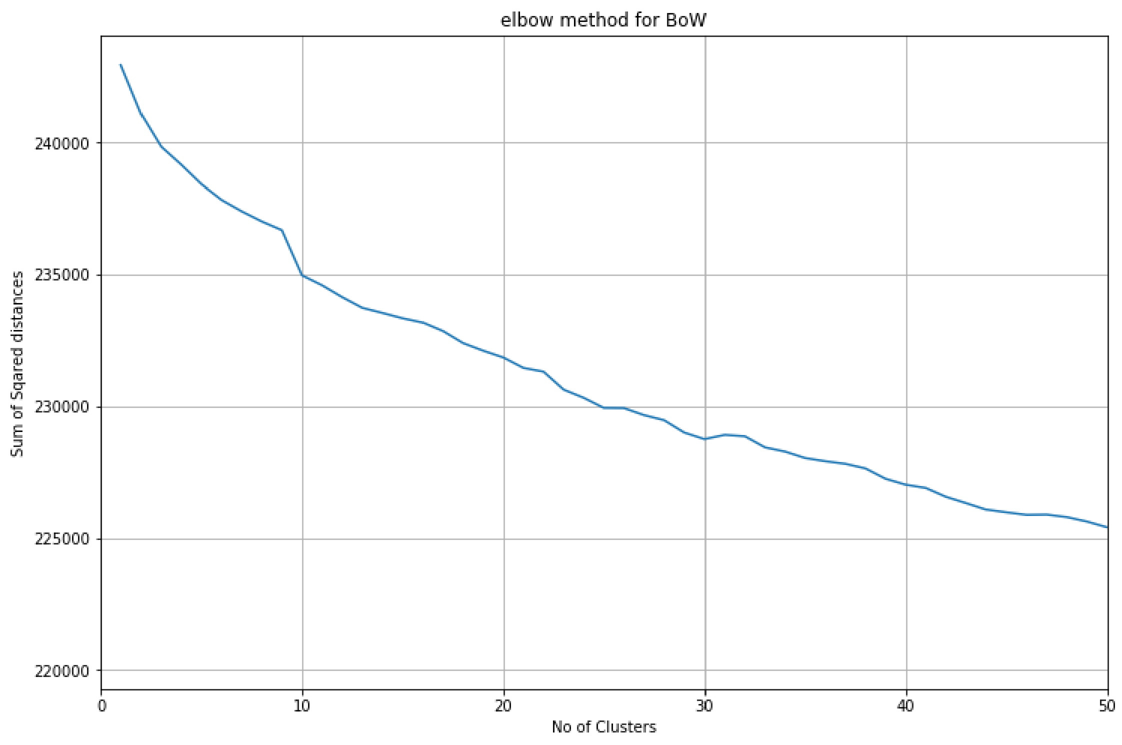
4 - Some coffe related products with differnt flovers
5 - related to drinks and soups
6 - some animal foods
7 - about tea
8 - about tea with large reviews
9 - Miscellaneous food products with some good reviews

0.0.2 Tf-Idf

```
In [12]: #TFIDF with (1,2) gram with cleaned data
         #tfidf vec
         tf_idf_vect = TfidfVectorizer(ngram_range=(1,1))
         final_counts_train = tf_idf_vect.fit_transform(
                 train_df['final_text'].values)
```

```
In [13]: for i in range(1,100):
             model = KMeans(n_clusters=i,n_init=50,max_iter=300,random_state=25,n_jobs=-1)
             model.fit(final_counts_train)
             ssd[i] = model.inertia_
             centers[i] = model.cluster_centers_
             labels = model.labels_
             print('No of clusters',i,'Sum of Squared dist',model.inertia_)
```

```
In [16]: plt.figure(figsize=(12,8))
         plt.plot(list(ssd.keys()),list(ssd.values()))
         plt.xlim(0,50)
         plt.xlabel('No of Clusters')
         plt.ylabel('Sum of Sqared distances')
         plt.title('elbow method for BoW')
         plt.grid()
```



after 10 clusters squared distances is decreasing by very less rate than before 10. so 10 may be better choice of clusters.

```
In [27]: model = KMeans(n_clusters=10,n_init=10,max_iter=300,random_state=25,n_jobs=-1)
         model.fit(final_counts_train)

In [28]: model.labels_

Out[28]: array([2, 2, 2, ..., 9, 9, 0], dtype=int32)

In [30]: train_df['tfidf_pred_label'] = model.labels_

In [32]: #no of points in each group
         train_df.groupby('tfidf_pred_label')['final_text','ProductId'].count()

Out[32]:                   final_text  ProductId
         tfidf_pred_label
         0                      18315      18315
         1                      16372      16372
         2                      85268      85268
         3                       3611       3611
         4                       7254       7254
         5                      18734      18734
         6                       4713       4713
         7                      59518      59518
         8                       2300       2300
         9                      38835      38835
```

```
In [34]: grp = train_df.groupby('tfidf_pred_label')[['final_text','ProductId']]

In [65]: for i in range(10):
             print('group ',i)
             for i in grp.get_group(i).sample(5).values:
                 print(i[0])
                 print(i[1])
                 print()
```

```
group  0
great cup coffee this coffee really good have been drinking for our morning for the last year l
B0001ES9FI

perfect for aeropress this coffee the best ive found far and love that it organic wa using maj
B000KSTY86

yum crave this coffee didnt expect never order chocolate glazed donut but for tastebud this hi
B0033HPPIO

bam this great coffee this coffee ha spoiled for all the other flavor need good strong coffee a
B001D0IZBM

definite favorite delicious like chocolate covered cherry especially like added coffee will gi
B000KFVAF4

group  1
must for chai lover picked this tea with another chai white tea and mix the too together the t
B0026GBTQA

better than tip but the bag are poorly made variety must with tea best tea ive had always look
B000GINU0S

great intro loose tea reasonable price and good quality got this sampler because came after se
B0009TMZIM

best tea vendor service wa exceptionally quick and the tea ha been the best tea for after dinn
B004EDFLPS

great quality great price really great quality for little price the tea bit strong first but a
B001U052TY

group  2
yummy this chocolate yummy delicious and will make you fat enjoy and then get the treadmill
B000MM4SC2

very good seed bought these seed not spice for food but for hair ive read that fenugreek seed a
B000JMAVZS
```

```
great source when turned vegan worried about finding reliable source ground Salba ultra conveni
B0017I9P3W

soy oil undermines taste and health benefit decent large sardine for cheap moroccan production
B002T5TLYA

house vermont curry curry fan like the savory aroma delivered the finished product unless your
B001ATZQ68

group  3
doe not taste like pumpkin sadly this pumpkin soup tasted nothing like pumpkin the pumpkin used
B005DHXVVK

lousy after getting this make believe product wrote the distributor bookbinder soup and bookbin
B000H27MQG

one option when you cant find locally cannot always find this locally happy see here even the p
B00286BJ90

great for gluten free cooking since being diagnosed with celiac disease nearly two year ago hav
B001BM3POY

delicious soup kind expensive the taste this soup really great taste just good those japanese r
B001IZ9NDQ

group  4
best eaten right away youre going baking this your bread machine then sharing with friend famil
B001EPQ9JG

absolutely delicious this mix delicious not much cook but easy make amazing healthy waffle with
B000V1O3ZQ

great product love this mix taste great and easy use family ha celiac disease and cant have glu
B001D0676C

terrific bran muffin mix recently purchased this mix after twice ordering the vitalicious deep
B002MAX026

great product for baking already tried the poppy seed filling and got this too for more special
B000GZSB6E

group  5
very convenient thrilled can get friskiest subscribe and save disabled and have two old cranky
B002CJAOR6

perfect for aggressive chewer have american Staffordshire terrier aka pitbull and she just over
B0041LHND6
```

```
mint bone dog love them and they give her minty pleasing breath for short period time
B000G6T2UW

frustrating for dog got the this morning and have used twice dispense dog breakfast and dinner
B000KV61FC

cat love this stuff cat were never very enthusiastic about dry food premium otherwise until th
B001PMCFL4

group  6
best cereal ever ive been looking for cereal like this for many year taste great stay crunchy
B001E5E0AG

worst tasting cereal ever have complains about the nutritional benefit kashi lean cereal it go
B001E5E05G

great for breakfast bought these bar because never eat breakfast hate egg dont like most cerea
B000P09RJA

great taste absolutely love this cereal the raisin are plump and there are plenty them and lot
B000QSR2X4

sweet and crunchy life it name taste sweeter than calorie and gram fat zero saturated fat the
B0044CPA28

group  7
Olivier olive oil from france very hard find store usa used sold occitan store all over that f
B001GR3VUM

lavazza daily standard really like lavazza coffee general recently tried the super cream since
B000SDKDM4

steak strip good brand they must pay their sale people lot tho saw these store for least more
B000GW46D4

perfection this very good quality candy with elongated shape that come assortment fruit flavor
B004TLVVUE

excellent popcorn okay didnt buy this product from amazon actually but will consider ordering
B000SU1LUU

group  8
wonderful leave the italian make great pasta sure have colander with tiny hole before cooking
B000R9Q40E

great pasta amazon price lousy live this pasta gluten free house however amazon bulk price mor
B000FK7PQW
```

```
great angel hair pasta debones gluten free angel hair pasta great for people with gluten intole
B002DZRZ80

best brown rice elbow ive tried many different brand and shape rice pasta over the year even yo
B000FK7PQW

love give you the taste and texture fresh pasta also like the fact cook quick and doesnt keep i
B001IZBT0Q

group  9
great meat substitute have been eating these child usually bread and fry these but they can als
B000DLB2E4

you want white plastic piece all over your floor get after week having them really dont care fo
B000084E6V

baking bargain bonanza love bake and probably bake time per week hate running out ingredient bu
B000IMSSF4

love these little bunny absolutely love sannies snack mix ha great flavor and love the unique s
B002QU2ILQ

absolutely wonderful first saw these cousin house and wa just amazed they are the best for roas
B003WKO150
```

0 - coffe
1- tea
2 - some seed like produts but overlap is there
3- soups 4- bread cakes
5 -dog related
6 - some cereal produts
7 - misc produts
8 - pasta and noodle based
9 -candys and some junk food

Word2Vec

```
In [9]: #importing
        from gensim.models import Word2Vec
        from gensim.models import KeyedVectors
        import pickle
        import gensim
```

```
In [10]: import gensim
         list_of_sent=[]
         for sent in final_review.final_text.values:
             list_of_sent.append(sent.split())
```

```
In [11]: #word2vec model with 50 dim vector
         w2v_model_50=gensim.models.Word2Vec(list_of_sent,min_count=5,size=50, workers=8)
         #word2vec model with 100 dim vector
         w2v_model_100=gensim.models.Word2Vec(list_of_sent,min_count=5,size=100, workers=8)
         #word2vec model with 300 dim vector
         w2v_model_300=gensim.models.Word2Vec(list_of_sent,min_count=5,size=300, workers=8)
```

```
In [69]: #loading from disk
         w2v_model_100 = pickle.load(open('w2v_model_dt_100.p','rb'))
         w2v_model_50 = pickle.load(open('w2v_model_dt_50.p','rb'))
         w2v_model_300 = pickle.load(open('w2v_model_dt_300.p','rb'))
```

Avg Word2Vec

```
In [13]: # the avg-w2v for each sentence/review is stored in this list
         def avg_w2v(list_of_sent,model,d):
             '''
             Returns average of word vectors for
             each sentance with dimension of model given
             '''
             sent_vectors = []
             for sent in list_of_sent: # for each review/sentence
                 doc = [word for word in sent if word in model.wv.vocab]
                 if doc:
                     sent_vec = np.mean(model.wv[doc],axis=0)
                 else:
                     sent_vec = np.zeros(d)
                 sent_vectors.append(sent_vec)
             return sent_vectors
```

```
In [14]: list_of_sent_train=[]
         for sent in train_df.final_text.values:
             list_of_sent_train.append(sent.split())
```

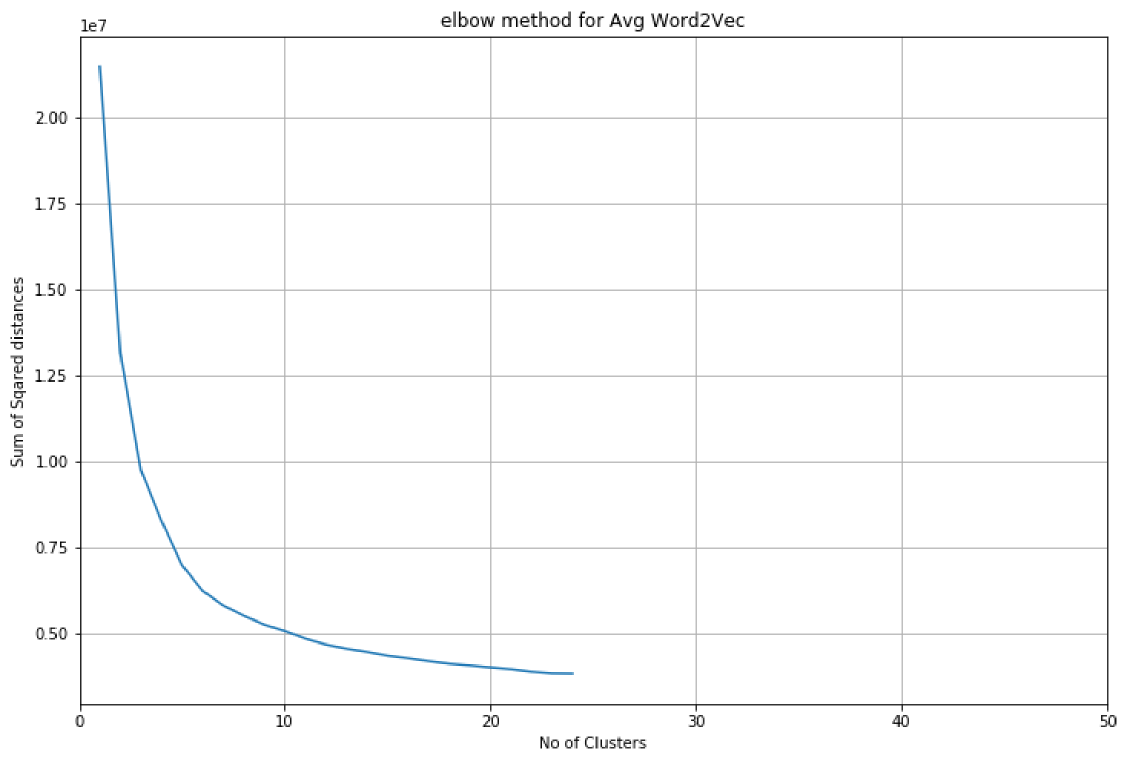
```
In [15]: #avg word2vec for
         sent_vector_avgw2v_300 = avg_w2v(list_of_sent_train,w2v_model_300,300)
         #stacking columns
         train_avgw2v_300 = np.hstack((sent_vector_avgw2v_300,
                     train_df[['HelpfulnessNumerator','HelpfulnessDenominator']]))
         column = list(range(0,300))
         column.extend(['HelpfulnessNumerator','HelpfulnessDenominator'])
         train_df_avgw2v_300 = pd.DataFrame(train_avgw2v_300,columns=column)
```

```
In [125]: train_df_avgw2v_300.to_csv('train_df_avgw2v_300_km.csv')
```

```
In [18]: ssd = {}
         centers = {}
         s = []
         for i in range(1,25):
            model = KMeans(n_clusters=i,n_init=10,max_iter=300)
            model.fit(train_df_avgw2v_300)
            ssd[i] = model.inertia_
            centers[i] = model.cluster_centers_
            print('No of clusters',i,'Sum of Squared dist',model.inertia_)
```

```
No of clusters 1 Sum of Squared dist 21478458.5677
No of clusters 2 Sum of Squared dist 13151425.9848
No of clusters 3 Sum of Squared dist 9763301.69767
No of clusters 4 Sum of Squared dist 8287436.51602
No of clusters 5 Sum of Squared dist 6996673.5982
No of clusters 6 Sum of Squared dist 6249683.64389
No of clusters 7 Sum of Squared dist 5827920.66685
No of clusters 8 Sum of Squared dist 5528591.41371
No of clusters 9 Sum of Squared dist 5267355.5692
No of clusters 10 Sum of Squared dist 5083635.81328
No of clusters 11 Sum of Squared dist 4860846.33753
No of clusters 12 Sum of Squared dist 4685988.30479
No of clusters 13 Sum of Squared dist 4566925.31898
No of clusters 14 Sum of Squared dist 4473099.36642
No of clusters 15 Sum of Squared dist 4363387.89202
No of clusters 16 Sum of Squared dist 4288240.38983
No of clusters 17 Sum of Squared dist 4203399.91147
No of clusters 18 Sum of Squared dist 4130410.6869
No of clusters 19 Sum of Squared dist 4079661.38805
No of clusters 20 Sum of Squared dist 4024256.8153
No of clusters 21 Sum of Squared dist 3973378.92582
No of clusters 22 Sum of Squared dist 3900402.97946
No of clusters 23 Sum of Squared dist 3853867.75265
No of clusters 24 Sum of Squared dist 3846995.83497
```

```
In [19]: plt.figure(figsize=(12,8))
         plt.plot(list(ssd.keys()),list(ssd.values()))
         plt.xlim(0,50)
         plt.xlabel('No of Clusters')
         plt.ylabel('Sum of Sqared distances')
         plt.title('elbow method for Avg Word2Vec')
         plt.grid()
```



From elbow metod 8 cllusters is good choice

```
In [103]: model = KMeans(n_clusters=8,n_init=10,max_iter=300)
          model.fit(train_df_avgw2v_300)

Out[103]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
              n_clusters=8, n_init=10, n_jobs=1, precompute_distances='auto',
              random_state=None, tol=0.0001, verbose=0)

In [109]: model.labels_

Out[109]: array([6, 0, 6, ..., 6, 6, 6], dtype=int32)

In [110]: train_df['avgw2v_label'] = model.labels_

In [111]: #no of points in each group
          train_df.groupby('avgw2v_label')['final_text','ProductId'].count()

Out[111]:               final_text  ProductId
          avgw2v_label
          0                  66829      66829
          1                   2043       2043
          2                  25081      25081
          3                     75         75
          4                    227        227
```

```
        5                    6721       6721
        6                  153289     153289
        7                     655        655
```

```
In [112]: grp = train_df.groupby('avgw2v_label')[['final_text','ProductId']]

In [132]: for i in range(8):
              print('group ',i)
              for i in grp.get_group(i).sample(5).values:
                  print(i[0])
                  print(i[1])
                  print()
```

```
group  0
this too expensive this cost the super market why going pay almost for one can this product
B00510NMKO

atomic fire ball not sure why but order came and most the fireball are soft and few are harder
B003U4Q0E8

super ordered the pop tart and they came within week really nice have this option available tha
B000M2UNIO

good for the money must admit after realized this wa foreign honey did feel bit guilty for orde
B001652KD8

comparable Kellogg nutrigrain bar very similar Kellogg nutrigrain cereal bar but little more ta
B000PIX38S

group  1
keurig cause keurig coffee maker problem are our third keurig coffee maker the original model I
B002AQ00S0

minty sweet herbal sedation between third and fourth year medical school did four week electiv
B00020HHEA

great for diabetic wow wa surprised the negative review because think miracle noodle are great
B004WZ4HC6

musty wa very excited find this product bulk healthy and organic what wasnt happy about wa the
B001IZM92S

best coffee ive ever had bought the coffee originally because liked the idea organic coffee tha
B002ESSASK

group  2
organic and great price since not being able get Kirkland organic ground coffee Costco searche
B002ESSASK
```

```
great honey that packaged not great bottle ambrosia pure honey delicious and unlike most honey
B001EO616S

way too salty see this product got lot excellent review but disagree with the other reviewer wi
B000F2RIQC

contaminated bought this product from Costco this past weekend student teacher and gave jar mer
B002SHYCQQ

very fine quality have tried lot different coffee brand this year and this one far the best alt
B00015ECGC

group  3
fettuccini shape inedible texture and taste cant believe this wa made and packaged for human co
B000AQFQC6

wine get the venturi skip the stand first all you are wine drinker get this work else ive been
B002OL2MWM

gastrointestinal armageddon when got these couldnt contain excitement and ate about quarter bag
B000EVQWKC

yum wa the fence about buying these for month read the review and were impressed them but the p
B001RVFDOO

good price for popular gift bought this for parent christmas gift wa big hit and immediately bi
B005NYXE8S

group  4
very pleased far studied the keurig machine and others for over month before deciding this one
B004GWSWCQ

not what appears sent this item close relative for christmas having read the review this produc
B0010OF8UW

full medium roast espresso bean espresso bean can roasted different way lavazza and illy repres
B0002E2GQU

great tasting coffee many other review have mentioned these are not the normal but they work mu
B005ZBZLT4

betty crocker frosting fancy package they charged for shipping and handling figured okay temper
B0008IT4OM

group  5
great inexpensive little item have been using these cap constantly for the past week and havent
B00764BRS2
```

```
overpriced amazon like all the larabars that have tried they taste good and are able satisfy ap
B000ENUC3S

wonderfully smooth and great flavor better than almond alternative have been drinking this sinc
B002BG38I2

classy and romantic this wa gift wife for valentine day and she really liked wa worried the lav
B0001WAMN2

fry chicken this oil great high smoking point great for deep frying doesnt actually add any fla
B001EO5RBS

group  6
great comfort food the noodle are big and have firm texture dont follow the cheese sauce recipe
B000CQ01NS

great flour add waffle make waffle with about the mix being this flour and they taste great are
B001EO682A

the best these are favorite tomato the world put them everything everything and sometimes just
B000LKVH8S

some the best for pasta sauce the thing that make these great for pasta sauce aside from the ta
B000N17T4G

favorite tea have found this flavor Gazo tea hard find and wa very happy have found amazon this
B0017WO2H2

group  7
great mainstream option people please dont review product before youve tried give the impressio
B002AQP5FW

easter bunny make great alternative chocolate egg five year old daughter loved this easter pres
B00012182G

really work well ill this silly little thing actually work how can you get the effect decanted
B002OL2MWM

triclopyr the good stuff kill poison ivy dead this contains triclopyr which more aggressive tha
B000A0OG5K

way too expensive much cheaper through espresso these pod cost way more than ordering through e
B0099HD3YA
```

There is so much overlap in many groups and not able to divide by product types. because of

Word similarity avg vectors some reviews are overlapping. some of them i find
0 - Misc products but maximum contains good ratings
1- Candies chocolates
2 - oil/seed related
3- Misc products so much overlap but maximum contains bad ratings
4- Coffe/tea related
5 - max dog related

Tf-Idf Weighted Word2Vec

```
In [14]: from sklearn.base import BaseEstimator, TransformerMixin

         class TfidfWeightedWord2Vec(BaseEstimator, TransformerMixin):
             '''
             Class for Tfidf Weighted Word2Vec Calculations
             '''
             def __init__(self, word2vec):
                 self.word2vec = word2vec
                 self.word2weight = None
                 self.dim = word2vec.vector_size
                 self.tfidf = None

             def fit(self, X, y=None):
                 tfidf = TfidfVectorizer()
                 tfidf.fit(X[:,0])
                 self.tfidf = tfidf
                 #print(self.word2vec.wv.vocab.keys())
                 return self

             def tf_idf_W2V(self,feature_names,tf_idf_trans_arr,list_of_sent):
                 '''
                 tfidf weighted word2vec calculation
                 '''
                 import operator
                 dict_tfidf = {k: v for v, k in enumerate(feature_names)}
                 sent_vectors = []
                 i = 0
                 for sent in list_of_sent: # for each review/sentence
                     doc = [word for word in sent if word in self.word2vec.wv.vocab.keys()]
                     if doc:
                         #itemgetter
                         f = operator.itemgetter(*doc)
                         try:
                             #itemgetter from dict
                             final = f(dict_tfidf)
                             final = tf_idf_trans_arr[i,final]
                             #converting to dense
                             final = final.toarray()
```

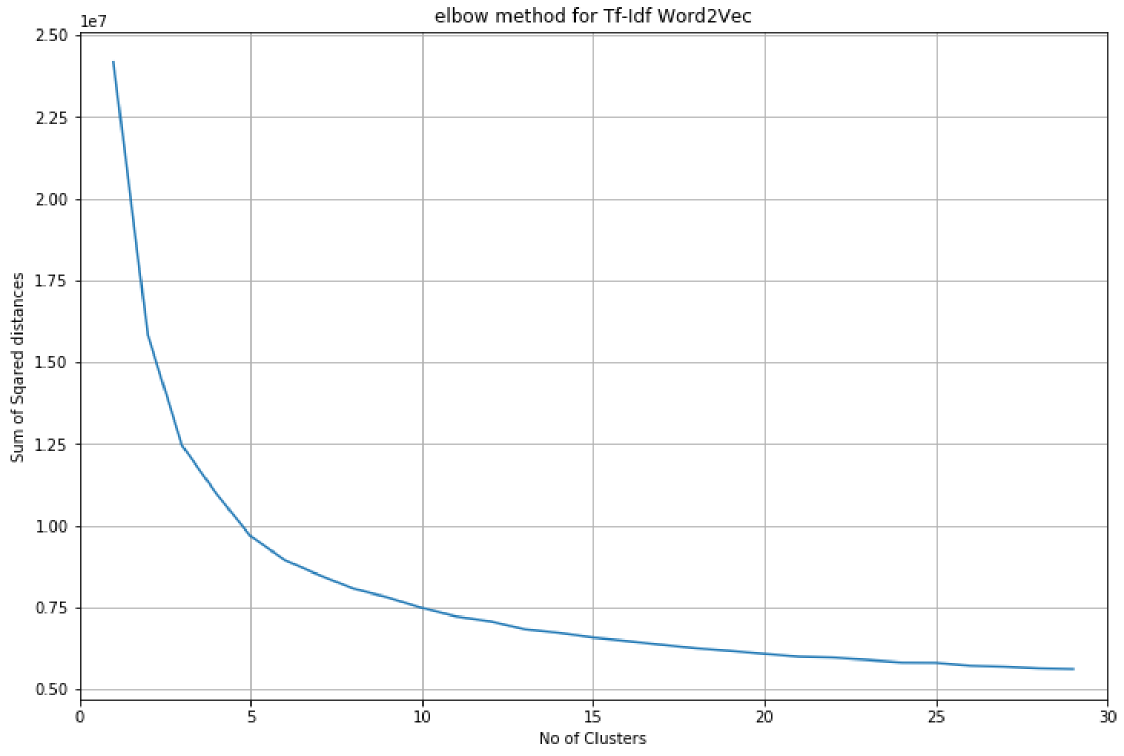
```
                                #converting to diagnol matrix for multiplication
                                final= np.diag(final[0])
                                sent_vec = np.dot(final,np.array(self.word2vec.wv[doc]))
                                #tfidf weighted word to vec
                                sent_vec = np.sum(sent_vec,axis=0) / np.sum(final)
                            except:
                                sent_vec = np.zeros(self.dim)
                        else:
                            sent_vec = np.zeros(self.dim)
                        sent_vectors.append(sent_vec)
                        i = i+1
                    return sent_vectors

                def transform(self, X):
                    #transform data
                    tf_idf_trans_arr = self.tfidf.transform(X[:,0])
                    feature_names = self.tfidf.get_feature_names()
                    list_of_sent = []
                    for sent in X[:,0]:
                        list_of_sent.append(sent.split())
                    temp_vec = self.tf_idf_W2V(feature_names,tf_idf_trans_arr,list_of_sent)
                    temp_vec= np.hstack((temp_vec,X[:,[1,2]]))
                    return temp_vec
```

```
In [15]: # For simple cv
         tfidfvect_w2v = TfidfWeightedWord2Vec(w2v_model_300)
         tfidfvect_w2v.fit(train_df[['final_text','HelpfulnessNumerator',
                                     'HelpfulnessDenominator']].values)
         X_train = tfidfvect_w2v.transform(train_df[['final_text',
                            'HelpfulnessNumerator','HelpfulnessDenominator']].values)
```

```
In [140]: ssd = {}
          centers = {}
          s = []
          for i in range(1,30):
              model = KMeans(n_clusters=i,n_init=10,max_iter=300)
              model.fit(X_train)
              ssd[i] = model.inertia_
              centers[i] = model.cluster_centers_
              print('No of clusters',i,'Sum of Squared dist',model.inertia_)
```

```
No of clusters 1 Sum of Squared dist 24171833.5224
No of clusters 2 Sum of Squared dist 15844570.7006
No of clusters 3 Sum of Squared dist 12456011.6976
No of clusters 4 Sum of Squared dist 10979501.604
No of clusters 5 Sum of Squared dist 9679840.63928
No of clusters 6 Sum of Squared dist 8944502.93965
No of clusters 7 Sum of Squared dist 8487686.19372
No of clusters 8 Sum of Squared dist 8071787.08606
No of clusters 9 Sum of Squared dist 7804478.04089
No of clusters 10 Sum of Squared dist 7482311.49733
No of clusters 11 Sum of Squared dist 7219493.00424
No of clusters 12 Sum of Squared dist 7071082.9754
No of clusters 13 Sum of Squared dist 6827409.11229
No of clusters 14 Sum of Squared dist 6715620.45991
No of clusters 15 Sum of Squared dist 6573033.69893
No of clusters 16 Sum of Squared dist 6466839.46489
No of clusters 17 Sum of Squared dist 6356654.12795
No of clusters 18 Sum of Squared dist 6247787.95109
No of clusters 19 Sum of Squared dist 6166076.20957
No of clusters 20 Sum of Squared dist 6073954.80593
No of clusters 21 Sum of Squared dist 5987239.57045
No of clusters 22 Sum of Squared dist 5960550.71549
No of clusters 23 Sum of Squared dist 5891144.76002
No of clusters 24 Sum of Squared dist 5810811.42773
No of clusters 25 Sum of Squared dist 5805908.30299
No of clusters 26 Sum of Squared dist 5714962.28228
No of clusters 27 Sum of Squared dist 5686507.08913
No of clusters 28 Sum of Squared dist 5634394.73677
No of clusters 29 Sum of Squared dist 5613232.60975
```

```
In [142]: plt.figure(figsize=(12,8))
          plt.plot(list(ssd.keys()),list(ssd.values()))
          plt.xlim(0,30)
          plt.xlabel('No of Clusters')
          plt.ylabel('Sum of Sqared distances')
          plt.title('elbow method for Tf-Idf Word2Vec')
          plt.grid()
```



from elbow metod we can find after 10 decrease is less so optimal may be 10.

```
In [109]: model = KMeans(n_clusters=10,n_init=10,max_iter=300)
          model.fit(X_train)

Out[109]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
              n_clusters=10, n_init=10, n_jobs=1, precompute_distances='auto',
              random_state=None, tol=0.0001, verbose=0)

In [110]: train_df['tfidfw2v_pred_label'] = model.labels_

In [111]:  #no of points in each group
          train_df.groupby('tfidfw2v_pred_label')['final_text','ProductId'].count()

Out[111]:                      final_text  ProductId
          tfidfw2v_pred_label
          0                         24524      24524
          1                         74134      74134
          2                          2142       2142
          3                          6315       6315
          4                           742        742
          5                         42445      42445
          6                         17689      17689
          7                            76         76
          8                         86578      86578
          9                           275        275
```

```
In [112]: grp = train_df.groupby('tfidfw2v_pred_label')[['final_text','ProductId']]

In [113]: for i in range(10):
              print('group ',i)
              for i in grp.get_group(i).sample(5).values:
                  print(i[0])
                  print(i[1])
                  print()

group  0
delightful tea think this the best jasmine tea ever full jasmine flower and fragrance and not l
B000NIHZMU

instant coffee hope you are reading these review did not this instant coffee inside container t
B006N3I69A

kahlua mocha coffee review wonderful tasting and wonderful smell the kahlua mocha coffee also r
B002TIR3BU

tea good addictive dont single day without drinking least one cup revolution sweet ginger peach
B0000WE0HY

very good husband just love this green tea buy allot get the diet with berry and the diet with
B003NL4TSM

group  1
wow this stuff awesome this time better than cheerio love this cereal eat for breakfast and eve
B001E5E060

great pistachio these pistachio arrived very quickly and tasted great other pistachio have trie
B001KW8UIG

chaka mm sauce marinade this hand down the best marinade weve ever tasted used year grilling it
B000E4AMUA

versatile received the seasoning gift and loved especially with olive oil for dipping warm crus
B001E5DR5K

sugar free cooky enjoyed the cooky they are great snack love the chocolate chip the best the br
B0008JF9FY

group  2
awesome these tablet are the best brand have tried far and they work very well the tablet itsel
B002INDU22

longer good used love them most recent purchase through amazon wa not good wa wild planet susta
B002EY5TTW
```

```
label changed but hurrah found favorite quirky ingredient what pomegranate molasses some probal
B001TZMCD8

vanilla bean the product excellent and fragrant have already used half what purchased making ic
B000ET4SM8

great nut almost dud love pistachio nut tried these since the price wa good and always looking
B001IZIEGS

group  3
read and read well one read this product ingredient they will find that the first ingredient co
B0026RHUP8

restricted diet beware for those restricted diet concerned about what you are ingesting aware t
B000QSS2C4

acquired taste but good grew big frank admit they are acquired taste they dont taste like regul
B000BEZVW2

get what you pay for the price wa cheap wa the product these rawhide bone literally disintegrat
B001CMMJFY

never received goat milk wa given track number and told wa delivered POBox never happened recei
B001E5DZTS

group  4
not the plant ordered dont buy this plant ordered this plant and cant even believe youre allowe
B0000DGF5S

puff pastry frozen dough ordered this product and arrived thawed and soured very disappointing
B000NY4SYW

not happy with this product have great amount experience with water have been producing home fo
B003JO71D8

cross contamination possible safe for diet just got order not here comment how the product beha
B0006ZN538

not too weak not too strong it just right found these first local supermarket ive been looking
B001EYUE5M

group  5
had better decided try and find horchata that tasted like the kind drink when growing san diego
B002GPNT32

exquisite balsamic vinegar this exquisite thick sweet aged balsamic vinegar when traveled rome
B000306AUG
```

```
both dog love these treat have australian shepherd and border collie both love these treat due
B0029NIF44

hot but not since amazon doesnt carry the hot and spicy sardine love thought would try these fo
B001225KXM

good for insulin spike why earth did start taking cinnamon powder you ask diet le than perfect
B001EQ5ABI

group  6
great but not this price Nornis are great they are all natural and even fit into weight loss pr
B003FY82YE

grreeaatt got first package the chili mix from did not know what wa for but gave try made accor
B000H2405M

not soy milk tasty like coffee mate which actually like but afraid it far le nutritious than so
B001E5E1PA

egg protein review found this good alternative whey based protein powder have only used this pr
B00166D8TW

really good ive been eating low carb for while now and every once while make some type low carb
B004LCCGZK

group  7
energy this green powder good just had get the word out others ive tried whole food brand orlea
B00112ILZM

neat and cheap alternative espresso capsule although wa bit skeptical first got say these are p
B005UP1M4I

having just eaten this for dinner have advice and raf first all dont put off the price youve ke
B000KEPBBY

prepare them yummy way following the south beach diet and saw these who wouldnt want have pasta
B004CLCEDE

our favorite rich jet fuel our favorite the coffee people fuel wake call and black tiger it dai
B0029XLH4Y

group  8
worst ever worst cinnamon tooth pick ive ever had dont know the were old what they are terrible
B0007OPW5C

wheres the bulk discount why would pay for can buy one pack for love udon noodle and difficult
B000LKX6RS
```

```
old ate the entire box didnt like this product all that well wa really tangy and hurt stomach w
B000Q5X876

shell purchased two package the stand stuff shell out twenty shell only five were intact very p
B000EFBMCQ

smart buy fit need perfectly more sticky finger lick off use only what you need and plenty ther
B0025UCA3I

group  9
inexpensive nylon grocery bag bag are very fashionable and useful they are also least bigger th
B00384ABOA

new out box and blender doe not work solved with DIY fix many other one star reviewer have post
B004Q3LBTG

little rainforest kitchen well have five these garden now obviously this product ive been very
B000FI4O90

great little sushi making kit this kit great gift idea and really good way start making yer ver
B000H241DS

bait switch fraud where come from bait switch called fraud thought that surely now amazon com w
B001HTJ2BQ
```

Not find any tyoe of groups by product type and upto 3 clusters the sum of sqared distance drop is very high so tried with 3 and got some results as below

```
In [56]: model = KMeans(n_clusters=3,n_init=10,max_iter=300)
         model.fit(X_train)

Out[56]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
             n_clusters=3, n_init=10, n_jobs=1, precompute_distances='auto',
             random_state=None, tol=0.0001, verbose=0)

In [57]: model.labels_

Out[57]: array([1, 1, 1, ..., 1, 1, 1], dtype=int32)

In [58]: train_df['tfidfw2v_pred_label'] = model.labels_

In [59]: train_df.tfidfw2v_pred_label.value_counts()

Out[59]: 1    231946
         0     21462
         2      1512
         Name: tfidfw2v_pred_label, dtype: int64
```

```
In [50]: train_df['Score'] = train_df['Score'].apply(lambda x : 0 if x == 'positive' else 1)

In [60]: train_df.Score.value_counts()

Out[60]: 0    216889
         1     38031
         Name: Score, dtype: int64

In [61]: #no of points in each group
         train_df.groupby('tfidfw2v_pred_label')['final_text','ProductId'].count()

Out[61]:                      final_text  ProductId
         tfidfw2v_pred_label
         0                         21462      21462
         1                        231946     231946
         2                          1512       1512

In [88]: acc = 0
         pos = 0
         for idx,row in train_df.iterrows():
             c = row['tfidfw2v_pred_label']
             if c == 1:
                 c = 0
             elif c == 0:
                 c = 1
             if row['Score'] == c:
                 acc = acc + 1
                 if c == 0:
                     pos = pos + 1

In [89]: accuracy = acc/len(train_df)
         accuracy

Out[89]: 0.8151851561274125

In [90]: tpr = pos/sum(train_df.Score==0)
         tpr

Out[90]: 0.9261004476944427
```

Bag of Word representation:

there is overlap but i can find maximum of them as below categories. 0 - Some Miscellaneous food products and reviews size are small
1 - Dog food related produtc 2 - coffee related products with small reviws
3- Coffe related produts with high reviews
4 - Some coffe related products with differnt flovers
5 - related to drinks and soups
6 - some animal foods
7 - about tea

8 - about tea with large reviews
9 - Miscellaneous food products with some good reviews
###### Tf-Idf representation: 0 - coffe related
1- tea related
2 - some seed like produts but overlap is there
3- soups 4- bread cakes
5 -dog related
6 - some cereal produts
7 - misc produts
8 - pasta and noodle based
9 -candys and some junk food
###### avg Word2Vec: There is so much overlap in many groups and not able to divide by product types. because of Word similarity avg vectors some reviews are overlapping. some of them i find
0 - Misc products but maximum contains good ratings
1- Candies chocolates
2 - oil/seed related
3- Misc products so much overlap but maximum contains bad ratings
4- Coffe/tea related
5 - max dog related
###### Tf-Idf Weighted Word2vec: i find there is so much overlap in the clusters and tried with 3 clusters, and checked for positive and negative samples in those clusters and got accuracy of 82% and true positive rate of 92%.

0.1 Hierarchical clustering

```
In [7]: from sklearn.cluster import AgglomerativeClustering
        from scipy.cluster.hierarchy import dendrogram, linkage
```

```
In [30]: ### Code from sklearn docs
         ##  https://github.com/scikit-learn/scikit-learn/
         ###blob/70cf4a676caa2d2dad2e3f6e4478d64bcb0506f7/examples/cluster/plot_hierarchical_c
         def plot_dendrogram(model, **kwargs):

             # Children of hierarchical clustering
             children = model.children_

             # Distances between each pair of children
             # Since we don't have this information, we can use a uniform one for plotting
             distance = np.arange(children.shape[0])

             # The number of observations contained in each cluster level
             no_of_observations = np.arange(2, children.shape[0]+2)

             # Create linkage matrix and then plot the dendrogram
             linkage_matrix = np.column_stack([children, distance, no_of_observations]).astype

             # Plot the corresponding dendrogram
             dendrogram(linkage_matrix, **kwargs)
```

```
In [9]: s = final_review.sample(5000)
```

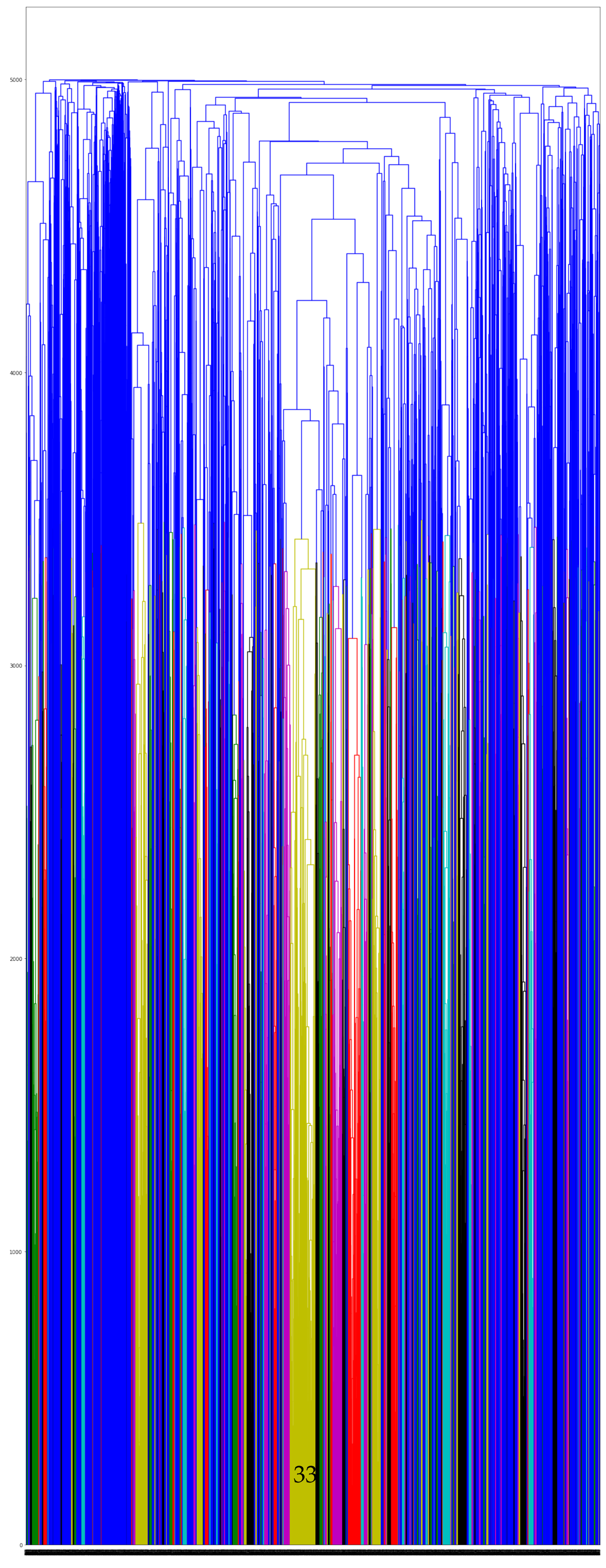
Bag of Words

```
In [10]: #BoW with cleaned data and without stopwords
         #simple cv for train data
         scores_train = []
         from nltk.corpus import stopwords
         stop = set(stopwords.words('english'))
         stop.remove('not')
         stop.remove('very')
         #CountVectorizer for BoW
         count_vect = CountVectorizer(stop_words=list(stop),dtype=np.int8)
         #X_test_cv = train_df.iloc[round(train_df.shape[0]*0.70):,:]
         final_counts_train = count_vect.fit_transform(
                 s['final_text'].values)
```

```
In [13]: model = AgglomerativeClustering(n_clusters=2,affinity='euclidean',linkage='ward')
         model.fit(final_counts_train.toarray())
```

```
Out[13]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                     connectivity=None, linkage='ward', memory=None, n_clusters=2,
                     pooling_func=<function mean at 0x1514781ff268>)
```

```
In [32]: plt.figure(figsize=(20,55))
         plot_dendrogram(model)
```


5000
4000
3000
2000
1000
0

```
In [39]: model6 = AgglomerativeClustering(n_clusters=5,affinity='euclidean',linkage='ward')
         model6.fit(final_counts_train.toarray())

Out[39]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                     connectivity=None, linkage='ward', memory=None, n_clusters=5,
                     pooling_func=<function mean at 0x1514781ff268>)

In [40]: s['bow_label'] = model6.labels_

In [42]: grp = s.groupby('bow_label')['final_text','ProductId']

In [43]: grp.count()

Out[43]:            final_text  ProductId
         bow_label
         0                3759       3759
         1                 283        283
         2                 437        437
         3                 354        354
         4                 167        167

In [47]: for i in range(5):
             print('group ',i)
             for i in grp.get_group(i).sample(5).values:
                 print(i[0])
                 print(i[1])
                 print()
```

```
group  0
it lot pepper it basic black pepper however for the price and the amount one heck deal for cou
B003LKDV4I

school snack this item wa exactly pictured would cant wait eat them front friend school and ma
B004RREF5S

day christmas review the day christmas going review Morton salt substitute pack taste great it
B00473QUGO

love this stuff crunch sweetness little salt this satisfies all the craving and ha crisp textu
B001E52Z2G

horrible ordered the sugar added version usually like sugar added low sugar product ice cream
B000KFVAF4

group  1
gunpowder green tea this anything but the sickly taste have grown used green tea robust tasty
```

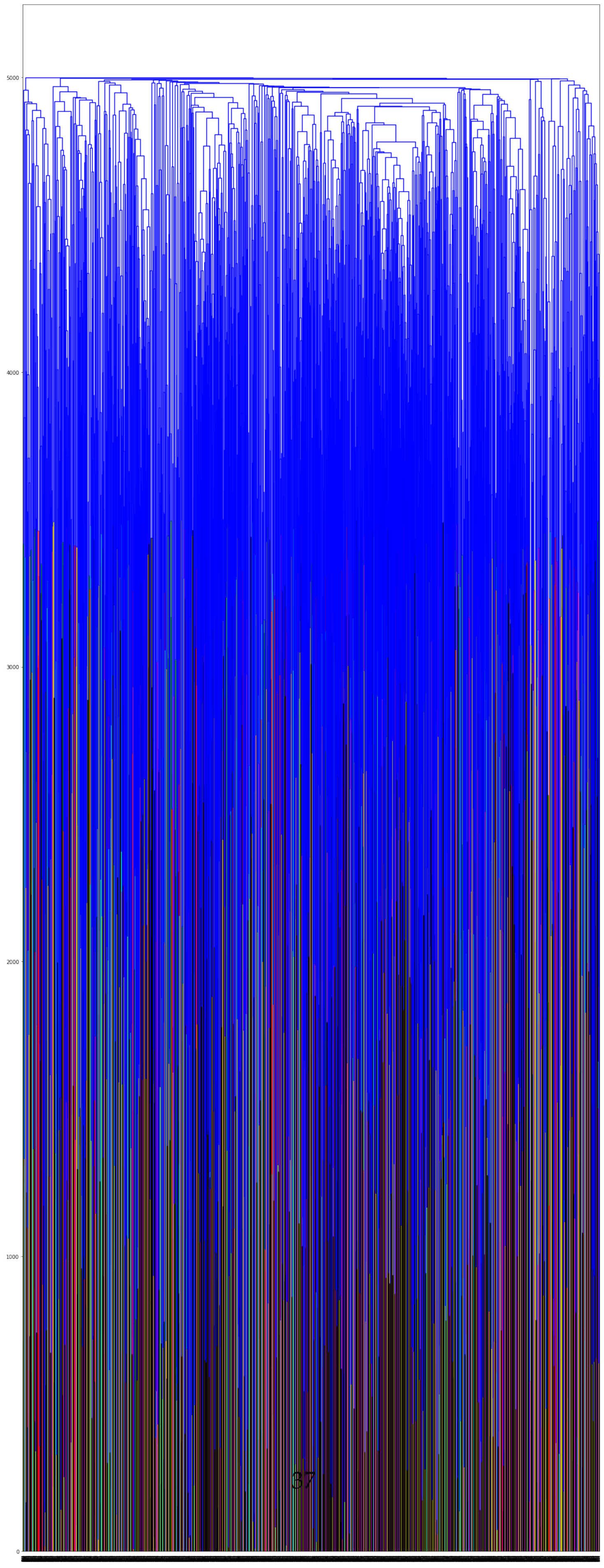
```
B000SATIFA

reliably good black tea will say for this tea since leaning toward but everyone else who drink
B000XEV9YE

good pretty tea label tea bag this cute little teapot and the tea delicious one complaint that
B000FFIL92

excellent tea but inconsistent quality this tea far the best tea available however some box hav
B001E5E34E

hard find decaf green tea with roasted brown rice keep ordering this product because hard find
B000LU146S

group  2
disappointing had the same experience with pasta not cooking all the way through even microwave
B0040HASFQ

just what wa looking for for weapon collector this grenade set just what wa looking for give fi
B004F1J98Y

great product for great price thanks all star health bought this product from the amazon vendor
B003B3OOPA

hard find whorehouse club never thought getting spice and such from amazon com however none our
B001EQ57PC

beware this product ha carrageenan gave this product one star because contains carrageenan whic
B001HTRDVC

group  3
another great bean another great whole bean from coffee bean direct drink lot coffee and like b
B002GWHAXK

very good deep flavor really full bodied coffee brew this aeropress with superb result personal
B001EO6FO6

good coffee for pod coffee system this one the best very flavorful
B0001ES9FI

when you want the best cafe broad just hit three home run book quality from the opening the bag
B004NV44AE

keurig hot chocolate pod watery and weak had brew the smallest cup and then put another pod get
B00389Q4XW

group  4
cat love this value cat love this cat food come when his bowl empty and meow which what call yo
```

```
B0029NICD8

cat are addicted this stuff this vendor entirely pet always prompt with delivery cant with out
B000ALGNHI

dog say not tasty recently ran low castor pollux organic adult le active canine formula dry do
B0007G9FD0

cat enthusiastic have cat and the day switched them this food they were eating like crazy have
B0051GBKZC

there soy the bible very unhappy with texture also very sad see the ezekiel brand jump the soy
B000LKV4LS
```

0 - So much overlap and found maxiumu candies/chololates
1 - Tea
2 - max milk/oil related and so much overlap
3 - coffe related
4 - animal related cat/dog

Tf-Idf:

```
In [48]: #TFIDF with (1,2) gram with cleaned data
         #tfidf vec
         tf_idf_vect = TfidfVectorizer(ngram_range=(1,2))
         final_counts_train = tf_idf_vect.fit_transform(
                 s['final_text'].values)

In [51]: model = AgglomerativeClustering(n_clusters=2,affinity='euclidean',linkage='ward')
         model.fit(final_counts_train.toarray())

Out[51]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                     connectivity=None, linkage='ward', memory=None, n_clusters=2,
                     pooling_func=<function mean at 0x1514781ff268>)

In [52]: plt.figure(figsize=(20,55))
         plot_dendrogram(model)
```


5000
4000
3000
2000
1000
0

From dendogram 5 is the better choice

```
In [57]: model5 = AgglomerativeClustering(n_clusters=5,affinity='euclidean',linkage='ward')
         model5.fit(final_counts_train.toarray())

Out[57]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                     connectivity=None, linkage='ward', memory=None, n_clusters=5,
                     pooling_func=<function mean at 0x1514781ff268>)

In [62]: s['tfidf_label'] = model5.labels_

In [63]: grp = s.groupby('tfidf_label')['final_text','ProductId']

In [64]: grp.count()

Out[64]:              final_text  ProductId
         tfidf_label
         0                  3907       3907
         1                   367        367
         2                   386        386
         3                   254        254
         4                    86         86

In [68]: for i in range(5):
             print('group ',i)
             for i in grp.get_group(i).sample(5).values:
                 print(i[0])
                 print(i[1])
                 print()

group  0
adam Scotts dream sauce carolina style barbecue compared barbecue the rest the united state ba
B0000DG5A8

great mix very tasty love these thing ive got all family hooked falafel the trick use boiling
B000S671XU

love this sauce visited belize earlier this year and brought back small sampler bottle this sa
B002VLY9HQ

great product great price wonderful flour right choice for those who are seeking organic bakin
B0049YMAII

ranging favorite little juice friend nothing comforting and fun the ranging bottle the photo d
B000121BY6

group  1
```

```
wonderful daughter just love this cereal make feel good feeding her not only something she love
B000YSQ9GC

worked great for cat this great food and cat love the only thing that seems bit strange the di
B003SE52K8

organic dog food chicken all organic dog food the food delivered monthly schedule via the subs
B000VK33C6

this great dog food few year back prompted the death one our dog due inoperable brain tumor sw
B001CWVZUY

great dog obsessed with literally come running when call out you want greenie also comfort know
B000KAKZ2I

group  2
love amazon but this not deal love this coffee and drink all the time but the same box availabl
B00196QVPM

great cup one the better cup variety that are available taste like coffee and thats real plus
B004M62COU

decaf never tasted star wonderful smooth satisfying decaf coffee with fabulously rich flavor ch
B002TMV3E4

good good cup coffee not good Newmann for liking but better that the kona blend and priced righ
B001EO5Y8Y

favorite too love this coffee drink frequently always use cream creamer and sugar the taste rem
B001CHJ01A

group  3
love the tea smell great brewing love them the only thing the fine tea leave still get thru the
B001ET5Y16

wonderful tea bengal spice favorite tea all time ha perfect blend cinnamon ginger nutmeg and hi
B000E63LDS

the box say this the finest quality everyday black tea bought box discount store the box bag th
B000MPRP4C

picker upper love this tea just dont drink too close bedtime because can keep you awake really
B001ELLCT2

enjoy very good tea enjoy the decaffeinated option dont typically drink caffeine
B000GG5J0Y

group  4
```

```
may have made our kitty sick try feed our cat wholesome healthy food thought these treat would
B00027CL5S

surprised how many people like this cereal it dog food look and smell like and taste bland fed
B001E5E060

cat candy you buy one tin buy ten because your cat will think these are wildly delicious strang
B00068K7UE

avoderm cat favorite food avoderm natural select cut chicken chunk canned cat food cat ha been
B001VIY6ZU

pet hate Newmann product ive bought Newmann own product couple time and both dog and cat spit
B000VKADSS
```

0 - So much overlap but maximum are well rated
1 - dog related
2 - coffe
3 - tea related
4 - cat related food

Avg Word2vec

```
In [70]: #importing
         from gensim.models import Word2Vec
         from gensim.models import KeyedVectors
         import pickle
         import gensim

In [71]: #loading from disk
         w2v_model_100 = pickle.load(open('w2v_model_dt_100.p','rb'))
         w2v_model_50 = pickle.load(open('w2v_model_dt_50.p','rb'))
         w2v_model_300 = pickle.load(open('w2v_model_dt_300.p','rb'))

In [72]: # the avg-w2v for each sentence/review is stored in this list
         def avg_w2v(list_of_sent,model,d):
             '''
             Returns average of word vectors for
             each sentance with dimension of model given
             '''
             sent_vectors = []
             for sent in list_of_sent: # for each review/sentence
                 doc = [word for word in sent if word in model.wv.vocab]
                 if doc:
                     sent_vec = np.mean(model.wv[doc],axis=0)
                 else:
```

```
                sent_vec = np.zeros(d)
            sent_vectors.append(sent_vec)
        return sent_vectors
```

```
In [76]: list_of_sent_train=[]
         for sent in s.final_text.values:
             list_of_sent_train.append(sent.split())
```

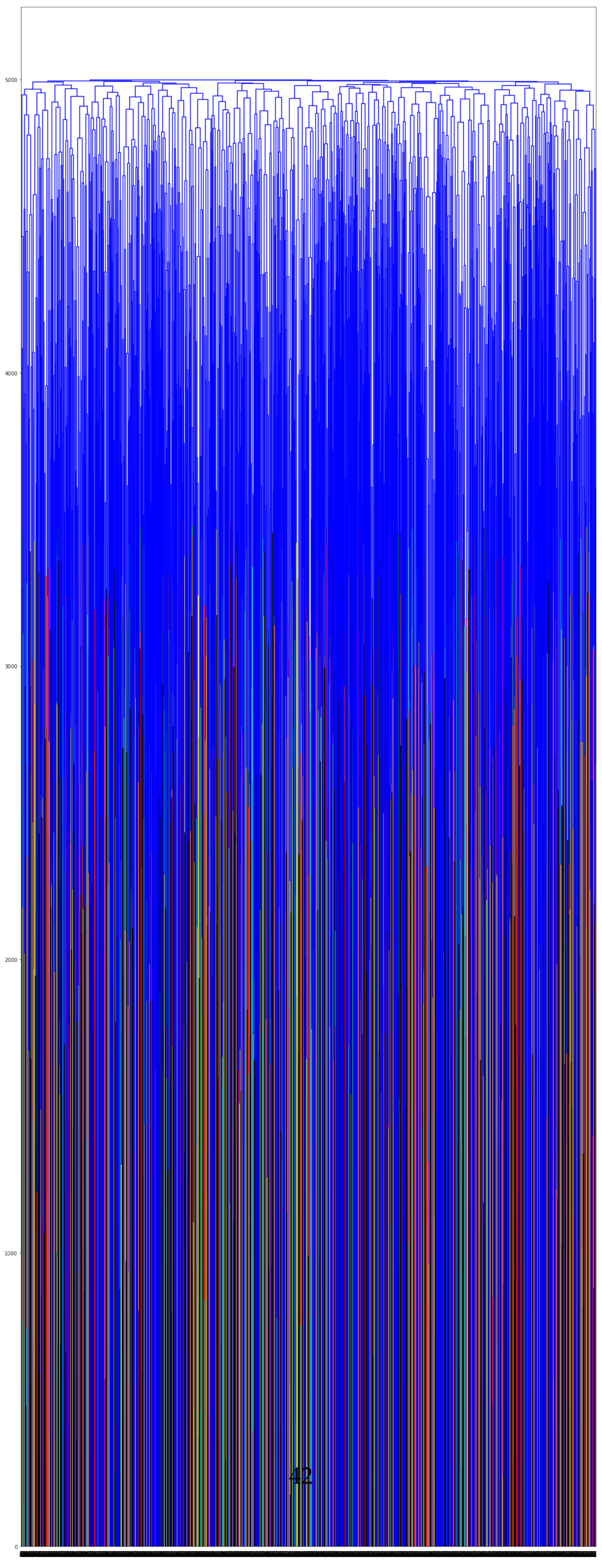
```
In [78]: #avg word2vec for
         sent_vector_avgw2v_300 = avg_w2v(list_of_sent_train,w2v_model_50,50)
```

```
In [79]: scale = StandardScaler()
         sent_vector_avgw2v_300_sc = scale.fit_transform(sent_vector_avgw2v_300)
```

```
In [82]: model = AgglomerativeClustering(n_clusters=2,affinity='euclidean',linkage='ward')
         model.fit(sent_vector_avgw2v_300_sc)
```

```
Out[82]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                     connectivity=None, linkage='ward', memory=None, n_clusters=2,
                     pooling_func=<function mean at 0x1514781ff268>)
```

```
In [83]: plt.figure(figsize=(20,55))
         plot_dendrogram(model)
```


5000
4000
3000
2000
1000
0

```
In [85]: model9 = AgglomerativeClustering(n_clusters=9,affinity='euclidean',linkage='ward')
         model9.fit(sent_vector_avgw2v_300_sc)

Out[85]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                     connectivity=None, linkage='ward', memory=None, n_clusters=9,
                     pooling_func=<function mean at 0x1514781ff268>)

In [87]: s['avgw2v_label'] = model9.labels_

In [90]: grp = s.groupby('avgw2v_label')['final_text','ProductId']

In [91]: grp.count()

Out[91]:               final_text  ProductId
         avgw2v_label
         0                   1341       1341
         1                    779        779
         2                    635        635
         3                    344        344
         4                    458        458
         5                    517        517
         6                    278        278
         7                    257        257
         8                    391        391

In [92]: for i in range(9):
             print('group ',i)
             for i in grp.get_group(i).sample(5).values:
                 print(i[0])
                 print(i[1])
                 print()

group  0
best especially for waffle have tried lot different mix and when found this new mix from Bisqu
B004391DK0

SWS grew with grandmother making dessert using holland rusk wa and favorite dessert and now mal
B000FEDHHE

addictive first had this local mexican restaurant and went crazy for even though thought might
B0000GHNSQ

all that edible from bee hive more it here think this acquired taste quite intense and flavorfu
B00014FT06

org stellar gluten free pasta eating bowl this amazing linguine with butter parmesan cheese and
```

```
B000LKUTIM

group  1
perfect every way these arrived perfectly breakage all came fast and the package were all fres
B000F9Z1XW

great service good product service wa great and the tea wa attractively packaged small tin tha
B0009TMZIM

new holiday tradition father life mile from and used order pecan pie from swiss colony and hav
B00310X7T2

uninspiring taste and calorie per cookie the chip ahoy chewy gooey chocofudge along with their
B004U43Z00

easy for the babysitter babysit for good friend kid give them date night every now and again t
B001DIQBM4

group  2
fairly strange taste dont know what the odd taste smell lightly vanilla and ive never brewed S
B003GTR8I0

very bad taste this wa the worst coffee lavazza gold selection whole bean coffee bag have trie
B000SDI03E

favorite chai ever ive tried several different brand chai tea the various coffee shop ive been
B0000E2RIA

new favorite this new favorite coffee very full flavored yet smooth win out over Starbucks Dun
B001EQ5P7C

grove square cappuccino french vanilla despite lot negative review still bought this product b
B005K4Q1YA

group  3
best rated cat food about com natural balance ultra premium wa rated canned food for cat and s
B000QSQTRY

delicious due some childhood incident have developed taste for dog chew the dingo big chew one
B000FPM5FS

summed this baby food wa summed right little Owen liked much wanted suck right out the contain
B0051C0PH6

crack for cat cat love this more than catnip these are shaved and dried piece bonito tuna some
B000IW71BQ

brought dog back life when first adopted dog from the local seattle animal rescue had hair his
```

```
B0019X02HI

group  4
popcorn bought this item because wa the perfect individual serving for everyone just right reco
B000STZRTW

imposter they are different from the version they dont taste feel quite right still tasty thoug
B003SG8GYK

yummy these cranberry are very yummy soft and chewy they are little sweeter than would have lil
B004563H2I

tasty treat really love these cooky there great balance mild cookie texture and flavorful choc
B004DC9BCO

lot yummy harico gumma bear big bag they send you nice large bag gumma bear they are fresh and
B000EVOSE4

group  5
the amount squeeze confusing first the kraft Mio web site doesnt say how much how long squeeze
B004E4EBMG

phenomenal agree with the last reviewer this indeed the best chai tea have ever tasted truthful
B001E5E1XW

love big health freak and this far best water flavor enhancer drink lot work and stay hydrated
B004XG2H94

these will get you moving highly recommended celsius calorie burner drink are every bit more e
B003OP8O1K

like oatmeal like like men thick and chunky well like oatmeal that way anyway take bit longer
B004VLVO9A

group  6
expensive but worth this coffee cake good wa worth pay the extra money it not available superma
B0002MS8HI

best pretzel love these pretzel and the amazon price make them even better they are local store
B000EVG8FQ

not worth this major disappointment found much better one elsewhere online better yet the store
B005NYXE24

tortilla scarce honest politician the fat free bandarita flour tortilla are great when can fin
B001SAVOJW

difficult find love this product and cant find here new mexico pleasantly surprised find here
```

```
B001E4S88W

group  7
clear scalp hair oil have permed hair found that this product didnt much for frizz but doe make
B007RTR9G0

good but inconsistent below previous review the next bag ordered had few strange piece red plas
B000R0R8QS

doe not work coffee brewer updated have the brewer WW amazon com and this the third box ive pu
B002AQ00L2

great concept but still need some work wa excited when this finally arrived the mail have cat a
B0002ASC04

scalp and hair conditioner when first tried this balm did apply too much and when washed out th
B007RTR9DS

group  8
really really good peanut husband peanut gourmet happy with the quality these peanut and happy
B002S5HJ1I

good gluten free cereal really liked this cereal and havent had regular cereal year the cornfla
B001E5E3E0

great love these little plant there fun watch check them every day for moisture they dry out pr
B0051S7P54

uncle eddy trail mix terrible maybe you have have taste for these cooky bought them and hated t
B000162MSG

good but not really dark chocolate kind nut spice dark chocolate nut sea salt bar are tasty sna
B007PE7ANY
```

0 - So much overlap
1 - max cookies
2 - coffe related
3 - pet related dog/cat
4 - bread/cakes
5 - Drinks related
6 - some candy/chips related
7 - max some oil related
8 - max cereal/penut butter related

Tf-Idf Weighted Word2Vec

```
In [97]: from sklearn.base import BaseEstimator, TransformerMixin

         class TfidfWeightedWord2Vec(BaseEstimator, TransformerMixin):
             '''
             Class for Tfidf Weighted Word2Vec Calculations
             '''
             def __init__(self, word2vec):
                 self.word2vec = word2vec
                 self.word2weight = None
                 self.dim = word2vec.vector_size
                 self.tfidf = None

             def fit(self, X, y=None):
                 tfidf = TfidfVectorizer()
                 tfidf.fit(X[:,0])
                 self.tfidf = tfidf
                 #print(self.word2vec.wv.vocab.keys())
                 return self

             def tf_idf_W2V(self,feature_names,tf_idf_trans_arr,list_of_sent):
                 '''
                 tfidf weighted word2vec calculation
                 '''
                 import operator
                 dict_tfidf = {k: v for v, k in enumerate(feature_names)}
                 sent_vectors = []
                 i = 0
                 for sent in list_of_sent: # for each review/sentence
                     doc = [word for word in sent if word in self.word2vec.wv.vocab.keys()]
                     if doc:
                         #itemgetter
                         f = operator.itemgetter(*doc)
                         try:
                             #itemgetter from dict
                             final = f(dict_tfidf)
                             final = tf_idf_trans_arr[i,final]
                             #converting to dense
                             final = final.toarray()
                             #converting to diagnol matrix for multiplication
                             final= np.diag(final[0])
                             sent_vec = np.dot(final,np.array(self.word2vec.wv[doc]))
                             #tfidf weighted word to vec
                             sent_vec = np.sum(sent_vec,axis=0) / np.sum(final)
                         except:
                             sent_vec = np.zeros(self.dim)
                     else:
                         sent_vec = np.zeros(self.dim)
                     sent_vectors.append(sent_vec)
```

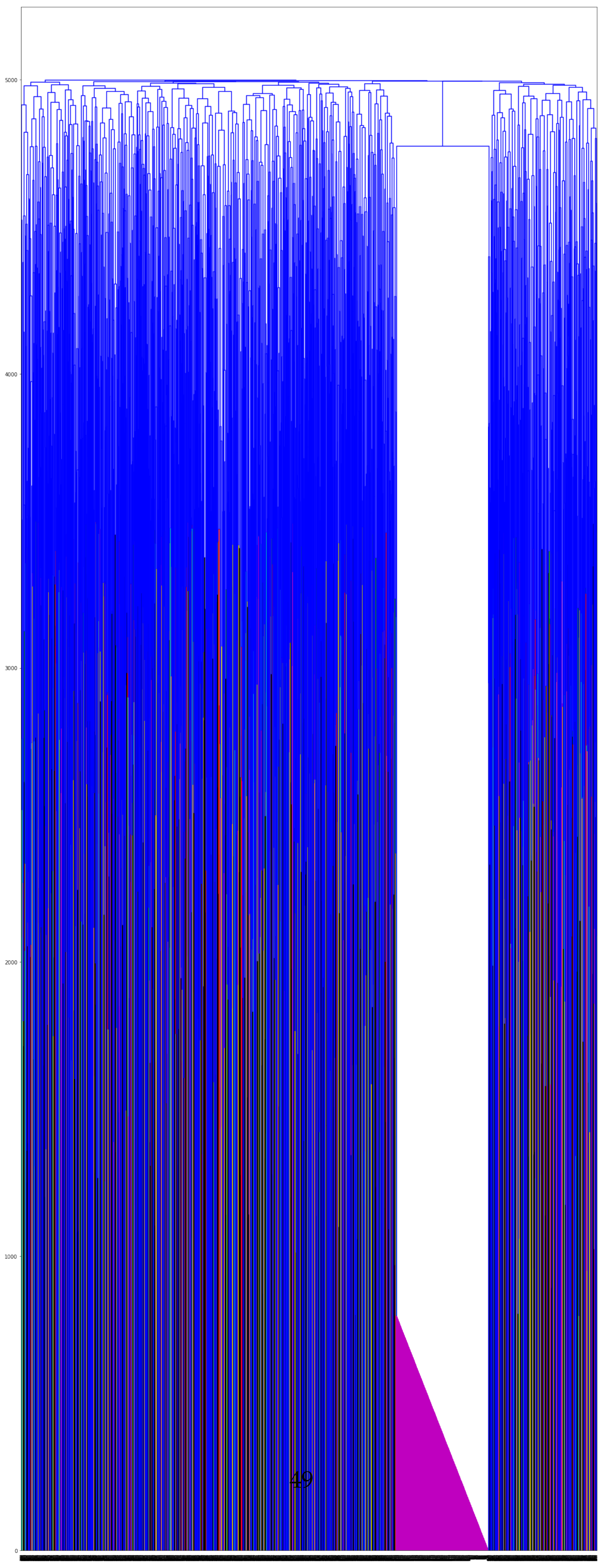
```
                i = i+1
            return sent_vectors

        def transform(self, X):
            #transform data
            tf_idf_trans_arr = self.tfidf.transform(X[:,0])
            feature_names = self.tfidf.get_feature_names()
            list_of_sent = []
            for sent in X[:,0]:
                list_of_sent.append(sent.split())
            temp_vec = self.tf_idf_W2V(feature_names,tf_idf_trans_arr,list_of_sent)
            #temp_vec= np.hstack((temp_vec,X[:,[1,2]]))
            return temp_vec
```

```
In [101]: tfidfvect_w2v = TfidfWeightedWord2Vec(w2v_model_50)
          tfidfvect_w2v.fit(s[['final_text','HelpfulnessNumerator',
                                     'HelpfulnessDenominator']].values)
          X_train = tfidfvect_w2v.transform(s[['final_text',
                          'HelpfulnessNumerator','HelpfulnessDenominator']].values)
```

```
In [103]: model = AgglomerativeClustering(n_clusters=2,affinity='euclidean',linkage='ward')
          model.fit(X_train)
```

```
Out[103]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                      connectivity=None, linkage='ward', memory=None, n_clusters=2,
                      pooling_func=<function mean at 0x1514781ff268>)
```

```
In [104]: plt.figure(figsize=(20,55))
          plot_dendrogram(model)
```


5000
4000
3000
2000
1000
0

```
In [116]: model7 = AgglomerativeClustering(n_clusters=7,affinity='euclidean',linkage='ward')
          model7.fit(sent_vector_avgw2v_300_sc)

Out[116]: AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                      connectivity=None, linkage='ward', memory=None, n_clusters=7,
                      pooling_func=<function mean at 0x1514781ff268>)

In [117]: s['tfidfw2v_label'] = model7.labels_

In [118]: grp = s.groupby('tfidfw2v_label')['final_text','ProductId']

In [119]: grp.count()

Out[119]:                 final_text  ProductId
          tfidfw2v_label
          0                      601        601
          1                      908        908
          2                     1341       1341
          3                      779        779
          4                      458        458
          5                      635        635
          6                      278        278

In [124]: for i in range(7):
              print('group ',i)
              for i in grp.get_group(i).sample(5).values:
                  print(i[0])
                  print(i[1])
                  print()
```

```
group  0
best out the bunch son would not breastfeed had supplement and eventually supplant with formula
B000RH4FG6

happy dog dog get variety rawhide bone from dingo the generic rawhide german shepherd and boxe
B002ZF06JI

saved cat and the glucotest really work wanted alternative sticking diabetic cat ear test his
B0000VJ55U

very satisfied the product doe provide faster muscle time readiness for that next set and incr
B000ELVUKY

best tool for puppy training have new puppy and are attempting train her using system this wor
B005CUU23S
```

```
group  1
taste horrible thank god amazon gave money back and ordered some other earl grey tea which rece
B001EO5WZ4

flavor review best flavor yet and instant favorite cant stop eating choc chip cookie dough all
B000ENUC3S

very tasty HMC use this sweeten our raw oat breakfast which add chopped apple dried cranberry r
B000FBQ5HQ

top notch mustard the taste this mustard cannot matched just the right balance ingredient full
B001EQ4L18

revision coca under law act absolute white ball blah blah even smell like coca chance getting e
B002N5IHSW

group  2
good you want orange zest dont get this product doe not taste anything like orange zest you wan
B001VNKVPE

black peppercorn not expert and have complaint with this product the shipment this product will
B000F0EX2Q

muesli doesnt get any better than this for once life addicted something that good for day doesn
B000EDDS6Q

tasty purchased single pack from supermarket made them using the provided method but added egg
B000H23ZE4

kitchen india dinner the product arrived promptly had not expected receive quickly definite thu
B002GQ6OEM

group  3
waffle mix the taste wa excellent there wa any negative wa very large amount did miss something
B0035LYCPU

love this candy found this chinatown next the white rabbit candy and thought give try wow love
B0009ZC482

disappointing these look very tasty and being candy freak had try them say the least wa extreme
B0000DISV2

little tough bought these for dried fruit bread take lot chopping get them the size that the fo
B001E5305C

complete but broken box arrived schedule the only compliant would the number broken cracker spi
B0016510LG
```

```
group  4
good health glory kettle sweet potato chip pack awesome sweet potato chip that arent real salt
B004L35LEC

remarkably good excellent quality dried mango with the right balance moisture and portability
B00433JTPC

very good quality these are very good organic walnut they are not dried out and have sort swee
B005BYP7RG

love these bar this one the best tasting snack bar have ever tasted trying lose weight and lik
B001FSK3IA

delicious but crumbled these are great delicious and healthy chip only complaint that when the
B00286DN5S

group  5
reliably good black tea will say for this tea since leaning toward but everyone else who drink
B000XEV9YE

excellent tasting tea cup this wa great tea not weak use the size and still good glad am doing
B001D0GV72

pod based previous review tried the san francisco bay coffee and they are excellent love the p
B007TGDXMU

delicious fresh green tea blended with macchia powder grew with green tea from japan wa always
B000WB1YSE

kahlua flavored coffee having lot fun with keurig coffee brewer there are som many flavor out
B002QGK2V8

group  6
good product this product came exactly describe high quality tomato paste tube you can use onl
B001FA1KLW

just like remember kid grew with valley kid outside tacoma the plant ha moved the midwest it b
B0057FT870

small price big reward vet initially recommended the Ceta hygiene for darling doggy boomer Bar
B001P05K6I

you gotta kidding paid buck for one package cracker not buy these you get individually wrapped
B0036Q07Z0

favorite stevia have used the sweetleaf stevia for year and ive tried others but the sweetleaf
B003W00I6C
```

0 - Max pet related
1 - Maximum good rated produts but much overlap to classify
2 - some misc items
3 - candies and breds
4 - Chips related
5 - coffe/tea related

0.1.1 DBSCAN:

Word2Vec

```
In [9]: #importing
        from gensim.models import Word2Vec
        from gensim.models import KeyedVectors
        import pickle
        import gensim
```

```
In [10]: #loading from disk
         w2v_model_100 = pickle.load(open('w2v_model_dt_100.p','rb'))
         w2v_model_50 = pickle.load(open('w2v_model_dt_50.p','rb'))
         w2v_model_300 = pickle.load(open('w2v_model_dt_300.p','rb'))
```

```
In [11]: # the avg-w2v for each sentence/review is stored in this list
         def avg_w2v(list_of_sent,model,d):
             '''
             Returns average of word vectors for
             each sentance with dimension of model given
             '''
             sent_vectors = []
             for sent in list_of_sent: # for each review/sentence
                 doc = [word for word in sent if word in model.wv.vocab]
                 if doc:
                     sent_vec = np.mean(model.wv[doc],axis=0)
                 else:
                     sent_vec = np.zeros(d)
                 sent_vectors.append(sent_vec)
             return sent_vectors
```

```
In [15]: s = final_review.sample(n=75000)
```

```
In [13]: final_review.iloc[s.index].groupby('Score')['final_text'].count()
```

```
Out[13]: Score
         negative     8847
         positive    66153
         Name: final_text, dtype: int64
```

```
In [16]: list_of_sent_train=[]
         for sent in s.final_text.values:
             list_of_sent_train.append(sent.split())
```

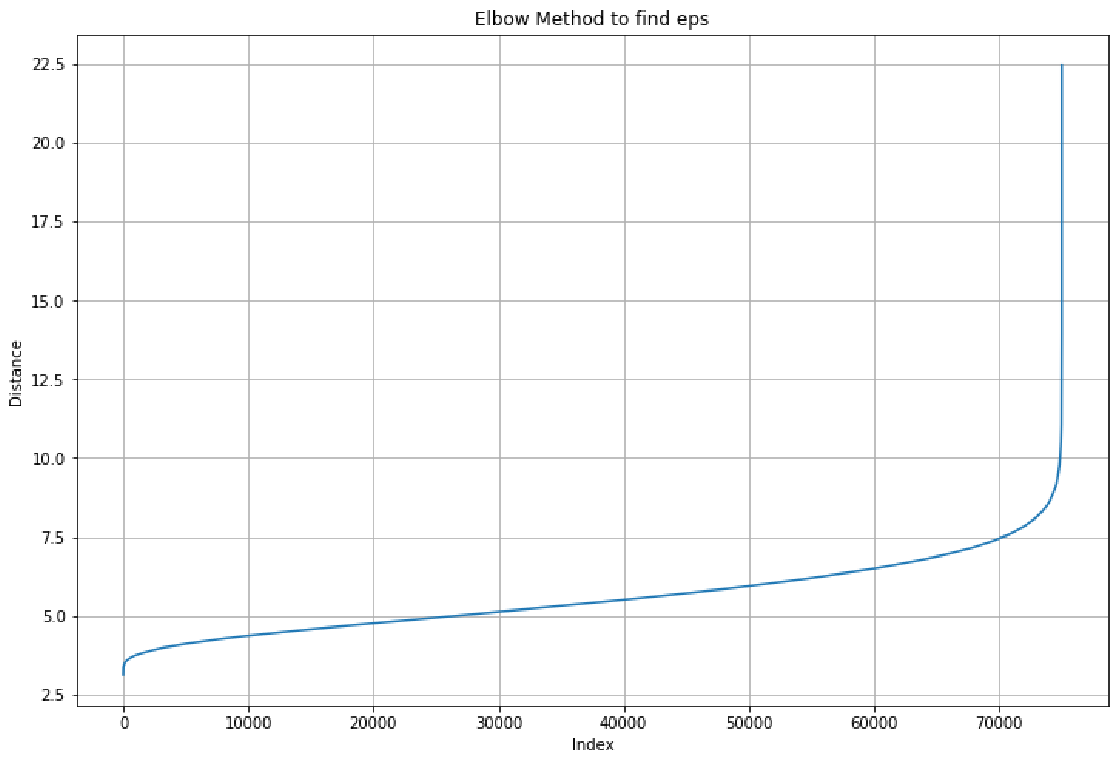
```
In [21]: #avg word2vec for
         sent_vector_avgw2v_300 = avg_w2v(list_of_sent_train,w2v_model_50,50)
```

```
In [22]: scale = StandardScaler()
         sent_vector_avgw2v_300_sc = scale.fit_transform(sent_vector_avgw2v_300)
```

```
In [14]: from sklearn.neighbors import NearestNeighbors
         nbrs = NearestNeighbors(n_neighbors=110, algorithm='ball_tree',n_jobs=-1).fit(X)
         distances, indices = nbrs.kneighbors(sent_vector_avgw2v_300_sc)
```

```
In [48]: dist = list(distances[:,-1])
         dist.sort()
```

```
In [54]: %matplotlib inline
         plt.figure(figsize=(12,8))
         plt.plot(list(range(len(dist))),dist)
         plt.xlabel('Index')
         plt.ylabel('Distance')
         plt.title('Elbow Method to find eps')
         plt.grid()
```

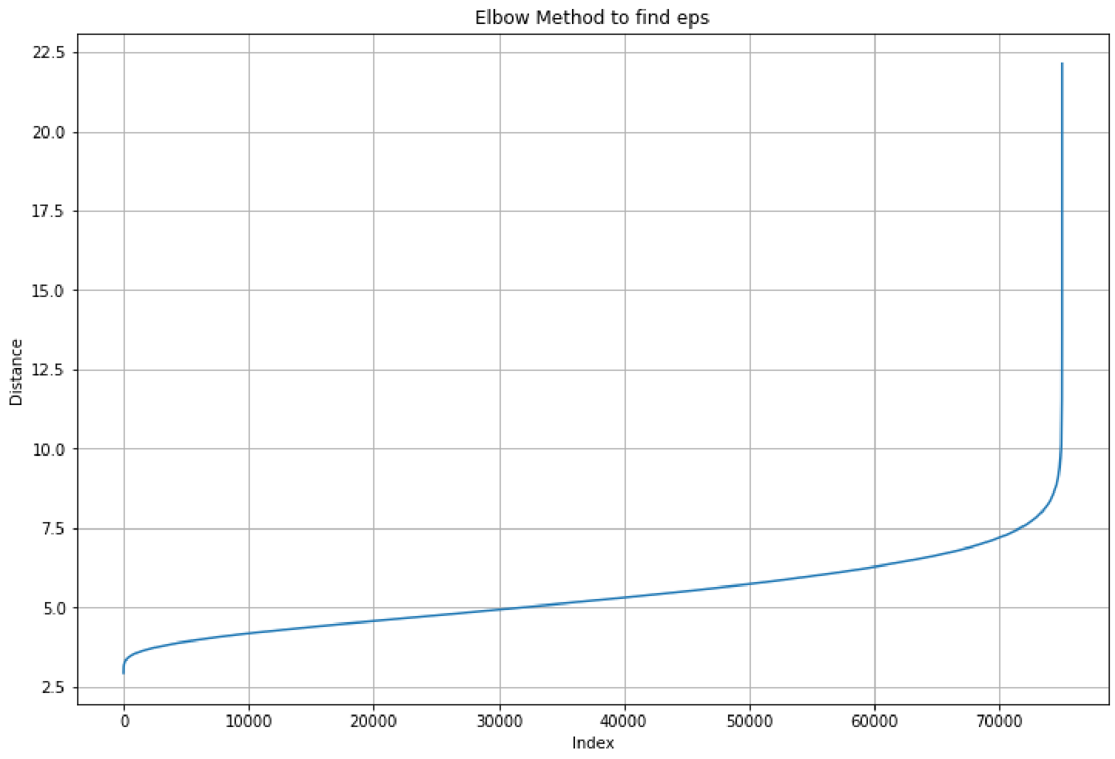


from above elbow metod eps of 8 may be better choice for the data.

```
In [82]: model_dbscan_avg = DBSCAN(eps=8.1,min_samples=105,n_jobs=-1)
         model_dbscan_avg.fit(sent_vector_avgw2v_300_sc_75)
         pickle.dump(model,open('model_dbscan_avg.p','wb'))

In [95]: s['db_label'] = model_dbscan_avg.labels_

In [96]: grp = s.groupby('db_label')['final_text','ProductId']

In [97]: grp.count()

Out[97]:           final_text  ProductId
         db_label
         -1               105        105
          0             74895      74895

In [125]: model_dbscan_avg1 = DBSCAN(eps=7.9,min_samples=105,n_jobs=-1)
          model_dbscan_avg1.fit(sent_vector_avgw2v_300_sc_75)
          pickle.dump(model_dbscan_avg1,open('model_dbscan_avg7.9.p','wb'))

In [126]: np.unique(model_dbscan_avg1.labels_)

Out[126]: array([-1,  0])
```

There is noise and all are one cluster for the above hyperparameters

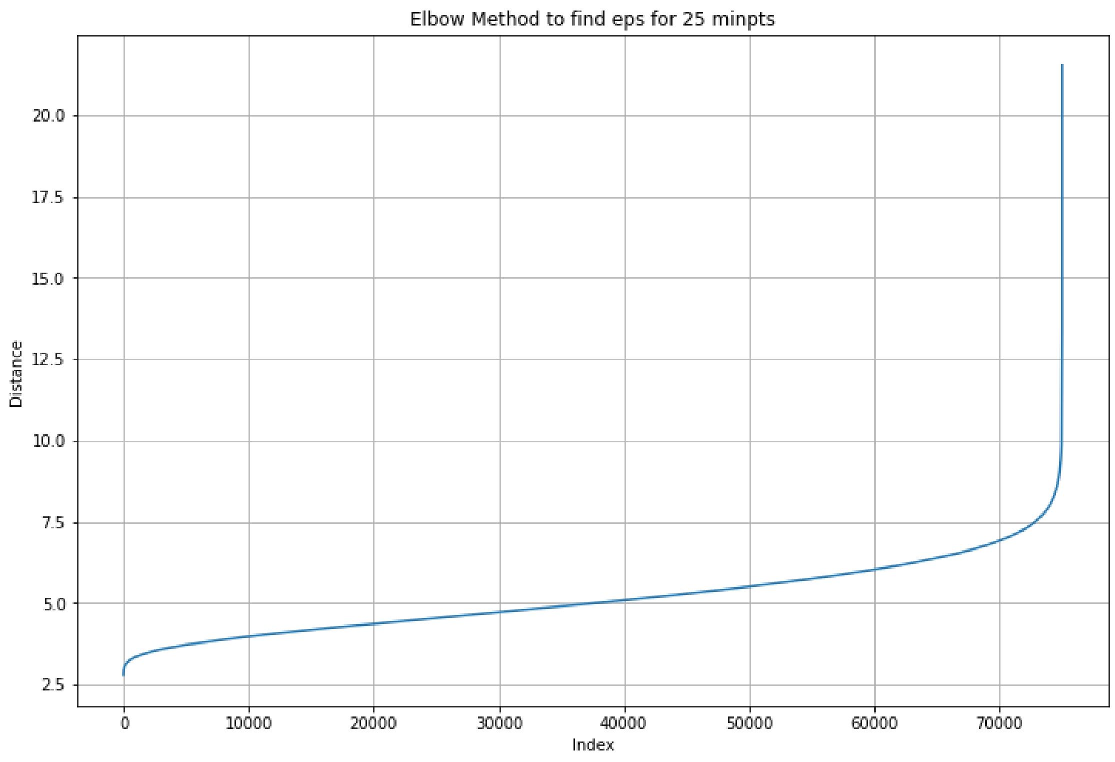
```
In [24]: nbrs = NearestNeighbors(n_neighbors=55, algorithm='ball_tree',n_jobs=-1).fit(sent_vec
         distances, indices = nbrs.kneighbors(sent_vector_avgw2v_300_sc)

In [25]: dist = list(distances[:,-1])
         dist.sort()

In [26]: %matplotlib inline
         plt.figure(figsize=(12,8))
         plt.plot(list(range(len(dist))),dist)
         plt.xlabel('Index')
         plt.ylabel('Distance')
         plt.title('Elbow Method to find eps')
         plt.grid()
```



```
In [42]: model_dbscan_avg55 = DBSCAN(eps=8.7,min_samples=55,n_jobs=-1)
         model_dbscan_avg55.fit(sent_vector_avgw2v_300_sc)
         pickle.dump(model,open('model_dbscan_avg55.p','wb'))

In [45]: set(model_dbscan_avg55.labels_)

Out[45]: {-1, 0}

In [2]: from scipy.stats import uniform
        z = uniform.rvs(7.4,1.6,10)
        for i in z:
            model = DBSCAN(eps=i,min_samples=55,n_jobs=-1)
            model.fit(sent_vector_avgw2v_300_sc)
            print('eps',i,set(model.labels_))

eps 7.699881438303933 {0, -1}
eps 7.775610106244951 {0, -1}
eps 8.771890878125285 {0, -1}
eps 7.508171138481954 {0, -1}
eps 8.706704451109745 {0, -1}
eps 8.191955625745837 {0, -1}
eps 8.95390860225897 {0, -1}
eps 8.81200211597915 {0, -1}
eps 8.223456200306703 {0, -1}
eps 8.135946474956082 {0, -1}
```

tried with different eps in the range of 7.4 to 9 but only one cluster is coming wit the minpoints = 55 also.

```
In [15]: distances, indices = pickle.load(open('list_dist_tfidf25.p','rb'))

In [16]: dist = list(distances[:,-1])
         dist.sort()

In [17]: %matplotlib inline
         plt.figure(figsize=(12,8))
         plt.plot(list(range(len(dist))),dist)
         plt.xlabel('Index')
         plt.ylabel('Distance')
         plt.title('Elbow Method to find eps for 25 minpts')
         plt.grid()
```



```
In [14]: z = [7.3,7.6,7.8,8,8.2,8.4,8.5]
         for i in z:
             model = DBSCAN(eps=i,min_samples=25,n_jobs=-1)
             model.fit(sent_vector_avgw2v_300_sc)
             print('eps',i,set(model.labels_))
```

```
eps 7.3 {0, -1}
eps 7.6 {0, -1}
eps 7.8 {0, -1}
eps 8 {0, -1}
eps 8.2 {0, -1}
eps 8.4 {0, -1}
eps 8.5 {0, -1}
```

```
In [15]: for i in z:
             model = DBSCAN(eps=i,min_samples=170,n_jobs=-1)
             model.fit(sent_vector_avgw2v_300_sc)
             print('eps',i,set(model.labels_))
```

```
eps 7.3 {0, -1}
eps 7.6 {0, -1}
eps 7.8 {0, -1}
eps 8 {0, -1}
eps 8.2 {0, -1}
eps 8.4 {0, -1}
eps 8.5 {0, -1}
```

Tried differnt eps and minpts but getting one cluster and another one is only noise

Tf-Idf Weighted Word2Vec

```
In [16]: from sklearn.base import BaseEstimator, TransformerMixin

         class TfidfWeightedWord2Vec(BaseEstimator, TransformerMixin):
             '''
             Class for Tfidf Weighted Word2Vec Calculations
             '''
             def __init__(self, word2vec):
                 self.word2vec = word2vec
                 self.word2weight = None
                 self.dim = word2vec.vector_size
                 self.tfidf = None

             def fit(self, X, y=None):
                 tfidf = TfidfVectorizer()
                 tfidf.fit(X[:,0])
                 self.tfidf = tfidf
                 #print(self.word2vec.wv.vocab.keys())
                 return self

             def tf_idf_W2V(self,feature_names,tf_idf_trans_arr,list_of_sent):
                 '''
```

```
                tfidf weighted word2vec calculation
                '''
                import operator
                dict_tfidf = {k: v for v, k in enumerate(feature_names)}
                sent_vectors = []
                i = 0
                for sent in list_of_sent: # for each review/sentence
                    doc = [word for word in sent if word in self.word2vec.wv.vocab.keys()]
                    if doc:
                        #itemgetter
                        f = operator.itemgetter(*doc)
                        try:
                            #itemgetter from dict
                            final = f(dict_tfidf)
                            final = tf_idf_trans_arr[i,final]
                            #converting to dense
                            final = final.toarray()
                            #converting to diagnol matrix for multiplication
                            final= np.diag(final[0])
                            sent_vec = np.dot(final,np.array(self.word2vec.wv[doc]))
                            #tfidf weighted word to vec
                            sent_vec = np.sum(sent_vec,axis=0) / np.sum(final)
                        except:
                            sent_vec = np.zeros(self.dim)
                    else:
                        sent_vec = np.zeros(self.dim)
                    sent_vectors.append(sent_vec)
                    i = i+1
                return sent_vectors

            def transform(self, X):
                #transform data
                tf_idf_trans_arr = self.tfidf.transform(X[:,0])
                feature_names = self.tfidf.get_feature_names()
                list_of_sent = []
                for sent in X[:,0]:
                    list_of_sent.append(sent.split())
                temp_vec = self.tf_idf_W2V(feature_names,tf_idf_trans_arr,list_of_sent)
                #temp_vec= np.hstack((temp_vec,X[:,[1,2]]))
                return temp_vec
```

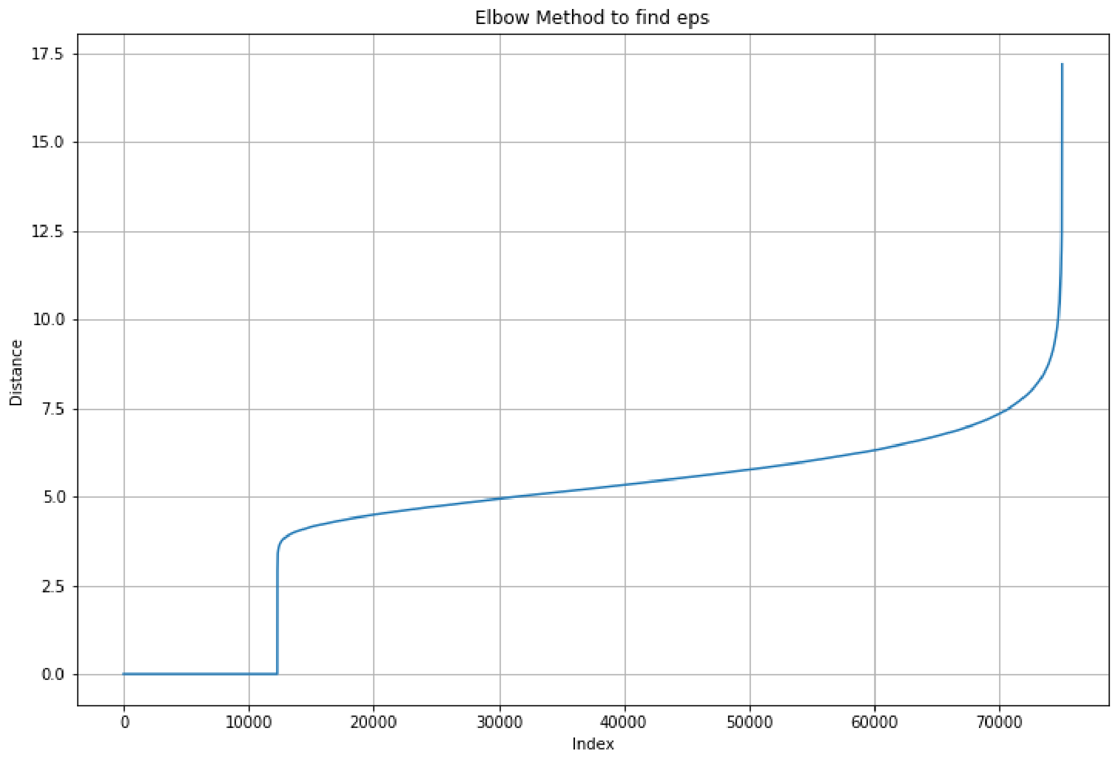
```
In [17]: tfidfvect_w2v = TfidfWeightedWord2Vec(w2v_model_50)
         tfidfvect_w2v.fit(s[['final_text','HelpfulnessNumerator',
                                           'HelpfulnessDenominator']].values)
         X_train = tfidfvect_w2v.transform(s[['final_text',
                             'HelpfulnessNumerator','HelpfulnessDenominator']].values)
```

```
In [18]: scale = StandardScaler()
         X_train_sc = scale.fit_transform(X_train)
```

```
In [137]: nbrs = NearestNeighbors(n_neighbors=110, algorithm='ball_tree',n_jobs=-1).fit(X_trai
          pickle.dump(nbrs,open('nbrs_tfidf.p','wb'))
          distances, indices = nbrs.kneighbors(X_train_sc)
```

```
In [138]: dist = list(distances[:,-1])
          dist.sort()
```

```
In [139]: %matplotlib inline
          plt.figure(figsize=(12,8))
          plt.plot(list(range(len(dist))),dist)
          plt.xlabel('Index')
          plt.ylabel('Distance')
          plt.title('Elbow Method to find eps')
          plt.grid()
```



from above elbow metod eps of 7.7 may be better choice for the data.

```
In [160]: from sklearn.cluster import DBSCAN
          model = DBSCAN(eps=7.7,min_samples=110,n_jobs=-1)
          model.fit(X_train_sc)
          pickle.dump(model,open('model_dbscan_tfidf7.p','wb'))
```

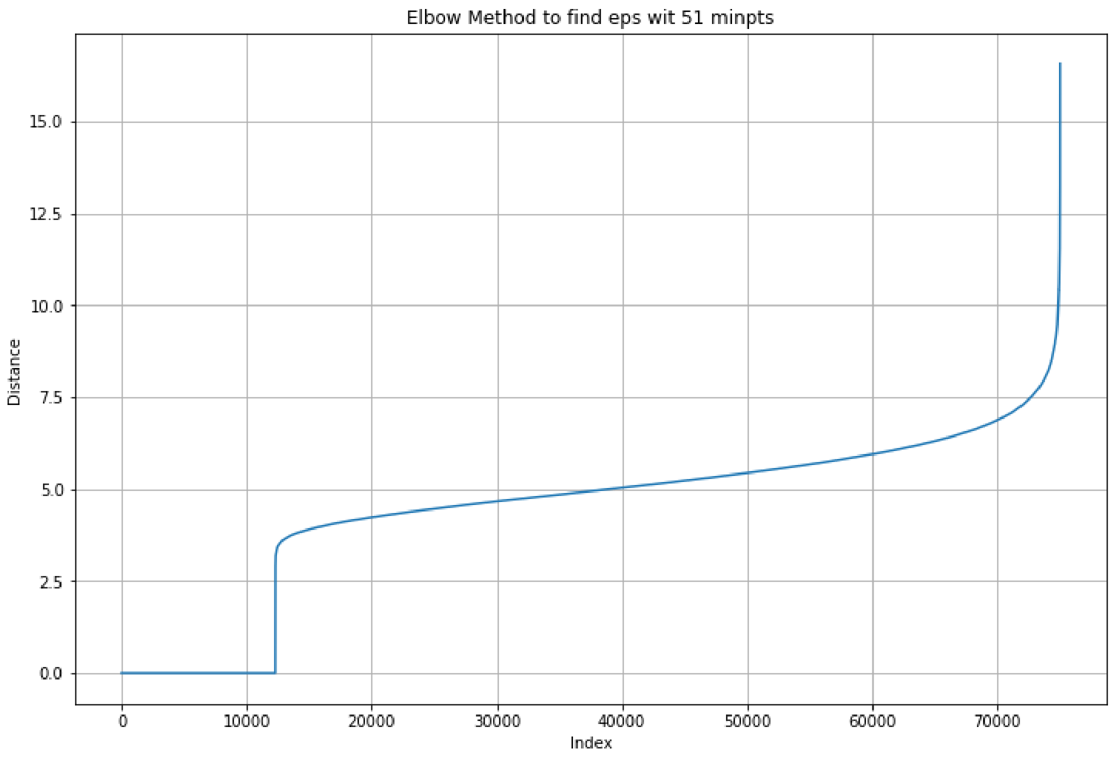
```
In [162]: np.unique(model_dbscan_tfidf7.labels_)
```

```
Out[162]: array([-1,  0])
```

```
In [34]: nbrs = NearestNeighbors(n_neighbors=51, algorithm='ball_tree',n_jobs=-1).fit(X_train_s
         distances, indices = nbrs.kneighbors(X_train_sc)
```

```
In [35]: dist = list(distances[:,-1])
         dist.sort()
```

```
In [36]: %matplotlib inline
         plt.figure(figsize=(12,8))
         plt.plot(list(range(len(dist))),dist)
         plt.xlabel('Index')
         plt.ylabel('Distance')
         plt.title('Elbow Method to find eps wit 51 minpts')
         plt.grid()
```



```
In [17]: z = [7.1,7.2,7.3,7.4,7.5,7.6,7.7,7.8,7.9,8]
         for i in z:
             model = DBSCAN(eps=i,min_samples=51,n_jobs=-1)
             model.fit(X_train_sc)
             print('eps',i,set(model.labels_))
```

```
eps 7.1 {0, -1}
eps 7.2 {0, -1}
eps 7.3 {0, -1}
eps 7.4 {0, -1}
eps 7.5 {0, -1}
eps 7.6 {0, -1}
eps 7.7 {0, -1}
eps 7.8 {0, -1}
eps 7.9 {0, -1}
eps 8 {0, -1}
```

Tried with different min samples and eps, got only one cluster and one noise cluster.