**WEKA & LIWC - PROCESS TO GENERATE SCORES**

**WEKA**

**STEP 1**

Extract data from database using following SQL command -

select Reviewid, ReviewerID, productid,DatePosted, Rating,WordCount, content from amazonproducthelpfulldis6 INTO OUTFILE 'D:/AmazonReviews.txt'

FIELDS TERMINATED BY '||'

ENCLOSED BY '"'

LINES TERMINATED BY '\n';

**STEP 2**

**Python Code -** To divide it into separate files of 200K records

**Input File -** File extracted from database

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**fil="/Users/love/Desktop/Final\_WEKA & LIWC/OutputARFF/AmazonReviews\_WekaLIWC\_3.csv"**

**csvfilename = open(fil, 'r').readlines()**

**file = 1**

**for j in range(len(csvfilename)):**

**if j % 200000 == 0:**

**open(str(fil)+ str(file) + '.csv', 'w+').writelines(csvfilename[j:j+200000])**

**file += 1**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

\*\*Dividing files to run efficiently in WEKA on local machine capacity.

**Add following header to each of the file before processing through WEKA -**

Reviewid,ReviewerID, productid,DatePosted,Rating,WordCount,content

**STEP 3**

**Run WEKA filters to get the required scores -**

* WEKA -> Explorer -> Open (file is loaded)
* Remove all the attributes except content
* Select required filter & click on Apply
* Save the result in .xlxs format
* Go to filter tab & right click to edit configuration
* Copy paste below configuration for respective filter

**Generate result for following filters -**

1- TweetToSentiStrengthFeatureVector

**weka.filters.unsupervised.attribute.TweetToSentiStrengthFeatureVector -L /Users/love/wekafiles/packages/AffectiveTweets/lexicons/SentiStrength/english -stemmer weka.core.stemmers.NullStemmer -stopwords-handler "weka.core.stopwords.Null " -I 1 -U -tokenizer "weka.core.tokenizers.WordTokenizer -delimiters \" \\r\\n\\t.,;:\\\'\\\"()?!\""**

2- TweetToLexiconFeatureVector

**weka.filters.unsupervised.attribute.TweetToLexiconFeatureVector -F -D -R -A -T -L -N -P -J -H -Q -stemmer weka.core.stemmers.NullStemmer -stopwords-handler "weka.core.stopwords.Null " -I 1 -U -tokenizer "weka.core.tokenizers.WordTokenizer -delimiters \" \\r\\n\\t.,;:\\\'\\\"()?!\""**

3- TweetToInputLexiconFeatureVector

**weka.filters.unsupervised.attribute.TweetToInputLexiconFeatureVector -lexicon\_evaluator "affective.core.ArffLexiconEvaluator -lexiconFile /Users/love/wekafiles/packages/AffectiveTweets/lexicons/arff\_lexicons/NRC-AffectIntensity-Lexicon.arff -B NRC-Affect-Intensity -A 1 -lex-stemmer weka.core.stemmers.NullStemmer" -stemmer weka.core.stemmers.NullStemmer -stopwords-handler "weka.core.stopwords.Null " -I 1 -U -tokenizer "weka.core.tokenizers.WordTokenizer -delimiters \" \\r\\n\\t.,;:\\\'\\\"()?!\""**

Settings - Word Tokenizer & Null stemmer, rest of the settings default

*\*\*If there is an error while loading the file correct it by removing required characters or line*

*Eg. following lines have been removed from main file before generating files in STEP2 -*

"R29IHUQCXLJPT7","A2CKQVXBSJEBRM","B003YJAZZ4","2012-12-31","5.00","1","790-cc790cc-4"

"RJPVQ7VIGYS01","A3DYZDY6864TEI","B00HZYFK72","2014-12-07","5.00","1","(\_)"

"RSHZUGVC213Q8","AJC43KPB753YX","B003YJAZZ4","2014-07-14","5.00","1","~~"

"RH5KD53GQMT30","A3TW266QHSQXFY","B0040ATVI8","2014-12-13","1.00","1","badWTF!!"

**Generating word count file** - Load each file in “Split\_File” folder in WEKA ,remove all attributes except WordCount & Content, then save them in “WC\_File” folder.

**Filter 1**

Generate WEKA score for filter 1 for all files in folder **“Split\_File”** & save result in folder **“Filter1\_input”.**

**Python Code -** To Combine results of all the the 10 files

**Input Files** - All files under ‘’Filter1\_Input’’ folder

**Output File -** Filter1.csv

\*\*\*\*\*\*\*

import pandas as pd

import math

dff = pd.DataFrame()

for i in range (1,11):

p1 = '/Users/love/Desktop/Final\_WEKA & LIWC/WC File/File%s\_WC.xlsx'%(i)

p2 = '/Users/love/Desktop/Final\_WEKA & LIWC/Filter1\_Input/File%s\_Filter1.xlsx'%(i)

dw = pd.read\_excel(p1,index\_col=None, header=0)

df = pd.read\_excel(p2,index\_col=None, header=0)

dw = pd.concat([dw, df], axis=1)

dff= pd.concat([dff, dw], axis=0)

dw = pd.DataFrame()

df = pd.DataFrame()

dff.insert(loc=4, column='Ratio', value='Nan')

dff['Ratio'] = abs(abs(dff['SentiStrength-posScore'])-abs(dff['SentiStrength-negScore']))/dff['WordCount']

dff.to\_csv('/Users/love/Desktop/Final\_WEKA & LIWC/Filter1.csv')

**Filter 2**

Generate WEKA score for filter 2 for all files in folder **“Split\_File”** & save result in folder **“Filter2\_input”.**

**Python Code -** To Combine results of all the the 10 files

**Input Files** - All files under ‘’Filter2\_Input’’ folder

**Output File -** Filter2.csv

\*\*\*\*\*\*\*\*\*\*\*\*\*

import pandas as pd

import math

dff = pd.DataFrame()

for i in range (1,11):

p1 = '/Users/love/Desktop/Final\_WEKA & LIWC/WC File/File%s\_WC.xlsx'%(i)

p2 = '/Users/love/Desktop/Final\_WEKA & LIWC/Filter2\_Input/File%s\_Filter2.csv'%(i)

dw = pd.read\_excel(p1,index\_col=None, header=0)

df = pd.read\_csv(p2,index\_col=None, header=0)

dw = pd.concat([dw, df], axis=1)

dff= pd.concat([dff, dw], axis=0)

dw = pd.DataFrame()

df = pd.DataFrame()

dff.insert(loc=4, column='Ratio1', value='Nan')

dff['Ratio1'] = abs(abs(dff['mpqa-posCount'])-abs(dff['mpqa-negCount']))/dff['WordCount']

dff.insert(loc=7, column='Ratio2', value='Nan')

dff['Ratio2'] = abs(abs(dff['BingLiu-posCount'])-abs(dff['BingLiu-negCount']))/dff['WordCount']

dff.insert(loc=10, column='Ratio3', value='Nan')

dff['Ratio3'] = abs(abs(dff['AFINN-posScore'])-abs(dff['AFINN-negScore']))/dff['WordCount']

dff.insert(loc=13, column='Ratio4', value='Nan')

dff['Ratio4'] = abs(abs(dff['S140-posScore'])-abs(dff['S140-negScore']))/dff['WordCount']

dff.insert(loc=16, column='Ratio5', value='Nan')

dff['Ratio5'] = abs(abs(dff['NRC-Hash-Sent-posScore'])-abs(dff['NRC-Hash-Sent-negScore']))/dff['WordCount']

dff.insert(loc=27, column='Ratio6', value='Nan')

dff['Ratio6'] = abs(abs(dff['NRC-10-negative'])-abs(dff['NRC-10-positive']))/dff['WordCount']

dff.insert(loc=38, column='Ratio7', value='Nan')

dff['Ratio7'] = abs(abs(dff['NRC-10-Expanded-negative'])-abs(dff['NRC-10-Expanded-positive']))/dff['WordCount']

dff.insert(loc=49, column='Ratio8', value='Nan')

dff['Ratio8'] = abs(abs(dff['SentiWordnet-posScore'])-abs(dff['SentiWordnet-negScore']))/dff['WordCount']

dff.to\_csv('/Users/love/Desktop/Final\_WEKA & LIWC/Filter2.csv')

**Filter 3**

Generate WEKA score for filter 3 for all files in folder **“Split\_File”** & save result in folder **“Filter3\_input”.**

**Python Code -** To Combine results of all the the 10 files

**Input Files** - All files under ‘’Filter3\_Input’’ folder

**Output File -** Filter3.csv

\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import pandas as pd

import math

dff = pd.DataFrame()

for i in range (1,11):

p1 = '/Users/love/Desktop/Final\_WEKA & LIWC/WC File/File%s\_WC.xlsx'%(i)

p2 = '/Users/love/Desktop/Final\_WEKA & LIWC/Filter3\_Input/File%s\_Filter3.xlsx'%(i)

dw = pd.read\_excel(p1,index\_col=None, header=0)

df = pd.read\_excel(p2,index\_col=None, header=0)

dw = pd.concat([dw, df], axis=1)

dff= pd.concat([dff, dw], axis=0)

dw = pd.DataFrame()

df = pd.DataFrame()

dff.to\_csv('/Users/love/Desktop/Final\_WEKA & LIWC/Filter3.csv')

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Combine All the result from Filter 1, Filter 2 & Filter 3 metrics**

Python Code

\*\*\*\*\*\*\*\*\*\*\*

import pandas as pd

import math

p1 = '/Users/love/Desktop/Final\_WEKA & LIWC/Filter1.csv'

p2 = '/Users/love/Desktop/Final\_WEKA & LIWC/Filter2.csv'

p3 = '/Users/love/Desktop/Final\_WEKA & LIWC/Filter3.csv'

df1 = pd.read\_csv(p1,index\_col=None, header=0)

df2 = pd.read\_csv(p2,index\_col=None, header=0)

df3 = pd.read\_csv(p3,index\_col=None, header=0)

df1 = df1.drop(['Unnamed: 0'], axis=1)

df2 = df2.drop(['Unnamed: 0','WordCount','content '], axis=1)

df3 = df3.drop(['Unnamed: 0','WordCount','content '], axis=1)

dw = pd.concat([df1, df2,df3], axis=1)

dw.to\_csv('/Users/love/Desktop/Final\_WEKA & LIWC/WEKA\_Output.csv')

\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LIWC**

**Python Code** - To construct input file for LIWC processing

**Input File** - WEKA Final Output file

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import pandas as pd

import math

p = '/Users/love/Desktop/Final\_WEKA & LIWC/WEKA\_Output.csv'

dw = pd.read\_csv(p,index\_col=None, header=0)

du = dw['content ']

du.to\_csv('/Users/love/Desktop/Final\_WEKA & LIWC/LIWC\_Input.csv')

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Process output file from above code through LIWC to produce result metrics

**Generate Final Output File -**

**Python Code -** Combine WEKA & LIWC results in one csv

**Input File** - WEKA Final Output File & LIWC Result File

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

p1 = '/Users/love/Desktop/Final\_WEKA & LIWC/LIWC2015 Results.csv'

dft = pd.read\_csv(p1,index\_col=None, header=0)

dft = dft.drop(['A','B','WC'], axis=1)

p = '/Users/love/Desktop/Final\_WEKA & LIWC/WEKA\_Output.csv'

dw = pd.read\_csv(p,index\_col=None, header=0)

dfinal = pd.concat([dw, dft], axis=1)

dfinal.insert(loc=91, column='Ratio9', value='Nan')

dfinal['Ratio9'] = abs(abs(dfinal['posemo'])-abs(dfinal['negemo']))/dfinal['WordCount']

dfinal = dfinal.drop(['Unnamed: 0'],axis=1)

dfinal.to\_csv('/Users/love/Desktop/Final\_WEKA & LIWC/Final\_Output.csv')

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**R- Code for Exponential Fitting Data**

library (readr)

# Loading dataset

Data <- read.csv(file="/Users/love/Desktop/Final\_WEKA & LIWC/Final\_Output.csv", header=TRUE, sep=",")

# View loaded data

View(Data)

# Removing word count less than 6

df <- Data[Data$WordCount > 6,]

View(df)

library(MASS)

fitdistr(df$Ratio, "exponential")

fitdistr(df$Ratio2, "exponential")

fitdistr(df$Ratio3, "exponential")

fitdistr(df$Ratio4, "exponential")

fitdistr(df$Ratio5, "exponential")

fitdistr(df$Ratio6, "exponential")

fitdistr(df$Ratio7, "exponential")

fitdistr(df$Ratio8, "exponential")

fitdistr(df$Ratio9, "exponential")

**Output -**

**Ratio 1**

rate

19.32111028

( 0.01504845)

**Ratio 2**

rate

12.554337520

( 0.009778079)

**Ratio 3**

rate

5.87804776

(0.00457818)

**Ratio 4**

rate

10.117594259

( 0.007880196)

**Ratio 5**

rate

8.908864458

(0.006938764)

**Ratio 6**

rate

19.25423277

( 0.01499636)

**Ratio 7**

rate

16.07244166

( 0.01251819)

**Ratio 8**

rate

14.15755452

( 0.01102676)

**Ratio 9**

rate

2.783396295

(0.002167878)

**Algorithm -**

*Select word count c1 (Try with c1=4, c2=7 and c3=10)*

*For reviews with count>c1, calculate the rate of the distribution of each of the ratios generated from R*

*Derive the value c2 = 1 - exp(-a\*rate)*

*Identify all the reviews with values equal of less than c2 for their ratio.*

*Extract those reviews and store them in separate file*

a = 0.95

**Below is code in R for first ratio column “Ratio” with c1 = 4** -

library (readr)

# Loading dataset

Data <- read.csv(file="/Users/love/Desktop/Final\_WEKA & LIWC/Final\_Output.csv", header=TRUE, sep=",")

# View loaded data

View(Data)

# Removing word count less than c1 = 4

df <- Data[Data$WordCount > 4,]

View(df)

library(MASS)

# Determining rate

y = fitdistr(df$Ratio, "exponential")

y

rate = coef(y)

rate

#Calculate c2

c2 = 1 - exp(-0.05 \* rate)

c2

# Separating dataframe

**df2 <- df[df$Ratio > c2,]**

**df3 <- df[df$Ratio <= c2,]**

View(df2)

View(df3)

write.csv(df2, "/Users/love/Desktop/filename.csv")

write.csv(df3, "/Users/love/Desktop/filename1.csv")

**Output File** - df2.csv & df3.csv

**c1 -> 4,7 & 10**

**Alpha -> 0.95**

**Below is code in R for all ratio columns -**

library (readr)

library(MASS)

# Loading dataset

Data <- read.csv(file="/Users/love/Desktop/Final\_WEKA & LIWC/Final\_Output.csv", header=TRUE, sep=",")

# View loaded data

View(Data)

df <- Data[Data$WordCount > 4,] ***#Change values as 4,7 & 10 & generate output csv files***

View(df)

alpha = 0.01**#Change values of alpha**

y = fitdistr(df$Ratio, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df1 <- df[df$Ratio <= c2,]

write.csv(df1, "/Users/love/Desktop/CutOff/Cutoff1/Ratio.csv")***#Change path according to c1***

y = fitdistr(df$Ratio1, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df2 <- df[df$Ratio1 > c2,]

write.csv(df2, "/Users/love/Desktop/CutOff/Cutoff1/Ratio1.csv")

y = fitdistr(df$Ratio2, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df3 <- df[df$Ratio2 > c2,]

write.csv(df3, "/Users/love/Desktop/CutOff/Cutoff1/Ratio2.csv")

y = fitdistr(df$Ratio3, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df4 <- df[df$Ratio3 > c2,]

write.csv(df4, "/Users/love/Desktop/CutOff/Cutoff1/Ratio3.csv")

y = fitdistr(df$Ratio4, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df5 <- df[df$Ratio4 > c2,]

write.csv(df5, "/Users/love/Desktop/CutOff/Cutoff1/Ratio4.csv")

y = fitdistr(df$Ratio5, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df6 <- df[df$Ratio5 > c2,]

write.csv(df6, "/Users/love/Desktop/CutOff/Cutoff1/Ratio5.csv")

y = fitdistr(df$Ratio6, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df7 <- df[df$Ratio6 > c2,]

write.csv(df7, "/Users/love/Desktop/CutOff/Cutoff1/Ratio6.csv")

y = fitdistr(df$Ratio7, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df8 <- df[df$Ratio7 > c2,]

write.csv(df8, "/Users/love/Desktop/CutOff/Cutoff1/Ratio7.csv")

y = fitdistr(df$Ratio8, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df9 <- df[df$Ratio8 > c2,]

write.csv(df9, "/Users/love/Desktop/CutOff/Cutoff1/Ratio8.csv")

y = fitdistr(df$Ratio9, "exponential")

rate = coef(y)

rate

c2 = 1 - exp(-alpha \* rate)

c2

df10 <- df[df$Ratio9 > c2,]

write.csv(df10, "/Users/love/Desktop/CutOff/Cutoff1/Ratio9.csv")

**Input File** - Final\_Output.csv (final WEKA output file with all the ratios)

**Output File** - 10 csv files according for each Ratio

**Note -** Run above code to generate output for different c1 values (4,7,10)

**Python Code to combine output & Produce metrics**

import pandas as pd

import math

p1 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio.csv'

p2 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio1.csv'

p3 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio2.csv'

p4 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio3.csv'

p5 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio4.csv'

p6 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio5.csv'

p7 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio6.csv'

p8 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio7.csv'

p9 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio8.csv'

p10 = '/Users/love/Desktop/CutOff/Cutoff1/Ratio9.csv'

df1 = pd.read\_csv(p1,index\_col=None, header=0)

df2 = pd.read\_csv(p2,index\_col=None, header=0)

df3 = pd.read\_csv(p3,index\_col=None, header=0)

df4 = pd.read\_csv(p4,index\_col=None, header=0)

df5 = pd.read\_csv(p5,index\_col=None, header=0)

df6 = pd.read\_csv(p6,index\_col=None, header=0)

df7 = pd.read\_csv(p7,index\_col=None, header=0)

df8 = pd.read\_csv(p8,index\_col=None, header=0)

df9 = pd.read\_csv(p9,index\_col=None, header=0)

df10 = pd.read\_csv(p10,index\_col=None, header=0)

df1 = df1.drop(['Unnamed: 0'], axis=1)

df2 = df2.drop(['Unnamed: 0'], axis=1)

df3 = df3.drop(['Unnamed: 0'], axis=1)

df4 = df4.drop(['Unnamed: 0'], axis=1)

df5 = df5.drop(['Unnamed: 0'], axis=1)

df6 = df6.drop(['Unnamed: 0'], axis=1)

df7 = df7.drop(['Unnamed: 0'], axis=1)

df8 = df8.drop(['Unnamed: 0'], axis=1)

df9 = df9.drop(['Unnamed: 0'], axis=1)

df10 = df10.drop(['Unnamed: 0'], axis=1)

dff= pd.concat([df1, df2, df3, df4, df5, df6, df7, df8, df9, df10], axis=0)

dff

dff = dff.drop\_duplicates()

dff

dff.insert(loc=3, column='CutOffRatio', value= dff.X.isin(df1.X).astype(np.int8))

dff.insert(loc=4, column='CutOffRatio1', value= dff.X.isin(df2.X).astype(np.int8))

dff.insert(loc=5, column='CutOffRatio2', value= dff.X.isin(df3.X).astype(np.int8))

dff.insert(loc=6, column='CutOffRatio3', value= dff.X.isin(df4.X).astype(np.int8))

dff.insert(loc=7, column='CutOffRatio4', value= dff.X.isin(df5.X).astype(np.int8))

dff.insert(loc=8, column='CutOffRatio5', value= dff.X.isin(df6.X).astype(np.int8))

dff.insert(loc=9, column='CutOffRatio6', value= dff.X.isin(df7.X).astype(np.int8))

dff.insert(loc=10, column='CutOffRatio7', value= dff.X.isin(df8.X).astype(np.int8))

dff.insert(loc=11, column='CutOffRatio8', value= dff.X.isin(df9.X).astype(np.int8))

dff.insert(loc=12, column='CutOffRatio9', value= dff.X.isin(df10.X).astype(np.int8))

dff

dw.to\_csv('/Users/love/Desktop/CutOff/Cutoff1\_Final/Output.csv')

**Input** - csv files generated for different Ratios

**Output** - Final csv file for each c1 value

**Note -** For each c1 value there will be separate final file with required metrics. Just update path to run above code for different c1 values.

**Requirement -**

*Select word count c1 (Try with c1=4, c2=7 and c3=10)*

*For reviews with count>c1, calculate the rate of the distribution of each of the ratios generated from R*

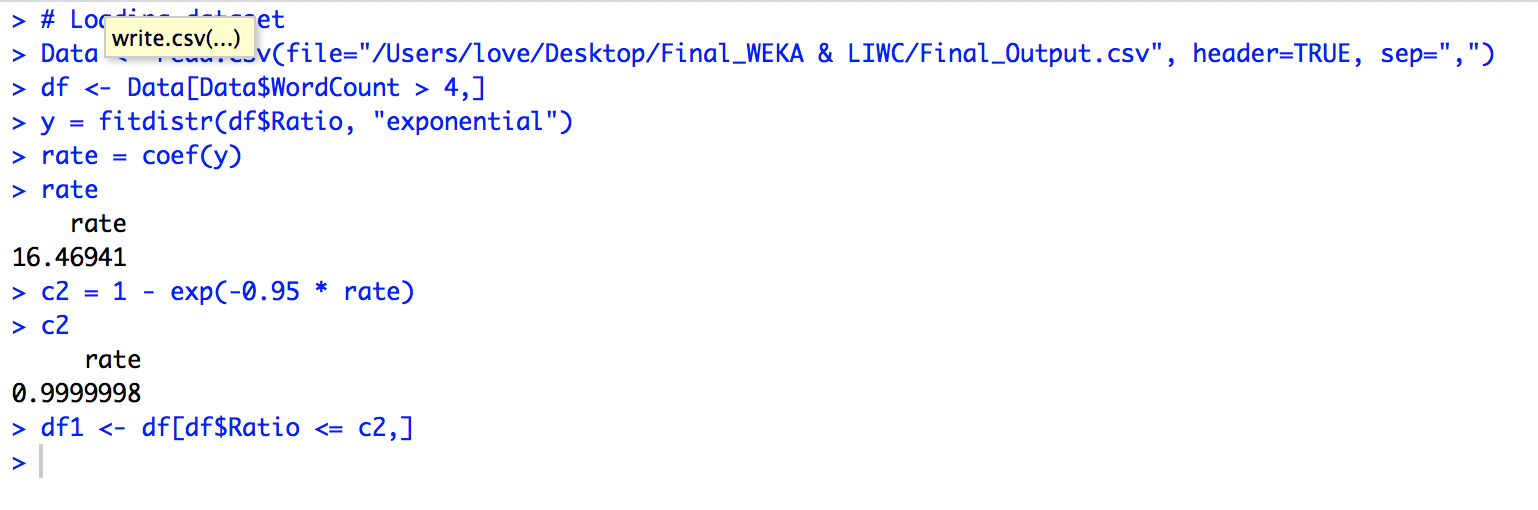
*Derive the value c2 = 1 - exp(-a\*rate)*

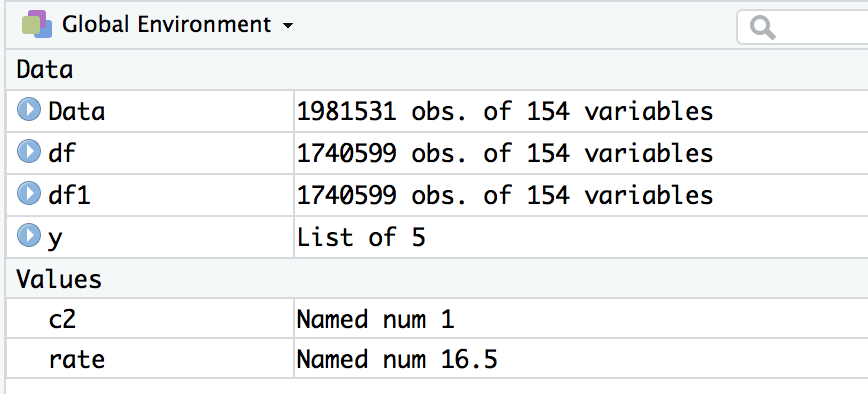
*Identify all the reviews with values equal of less than c2 for their ratio.*

*Extract those reviews and store them in separate file*

**Output**

c1 == 4 & alpha == 0.95





Total no of rows - 1981531

df - 1740599 rows left after filtering out word count > 4

df1-1740599 rows left after filtering out “Ratio <= c2”

Same is the case for all Ratio’s (more than 80% rows are filtering out in result)

**Note -** Same is case for c1 = 7 & 9 as well

**Results with different combination of alpha**

**Mapping of dataframes to refer below results -**

Data - Final result after WEKA

df- subset with wordcount greater than c1 (4,7 & 9)

df1-Ratio

df2-Ratio1

df3-Ratio2

df4-Ratio3

df5-Ratio4

df6-Ratio5

df7-Ratio6

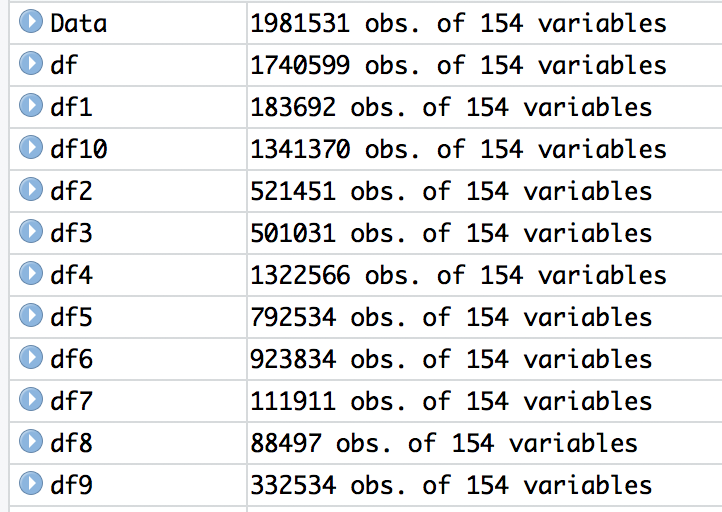
df8-Ratio7

df9-Ratio8

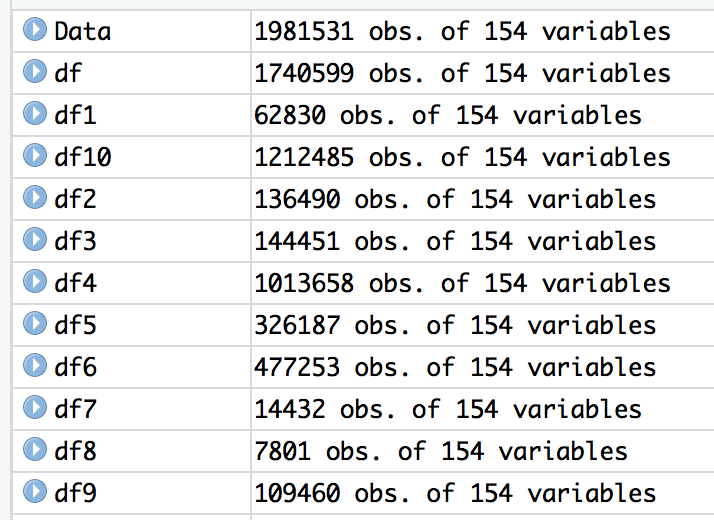
df10-Ratio9

***Word Count > 4***

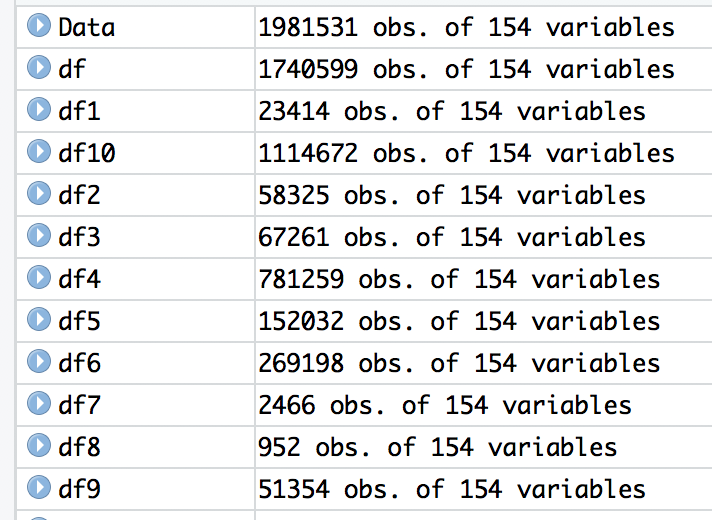
***Alpha = 0.01***

******

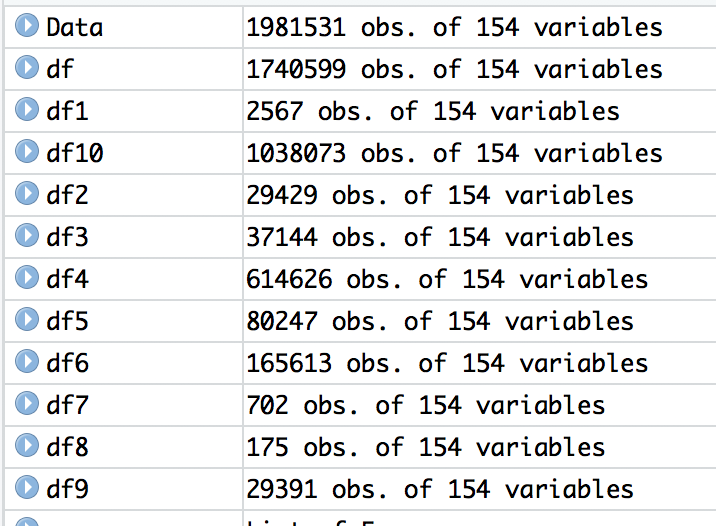
***Alpha = 0.02***



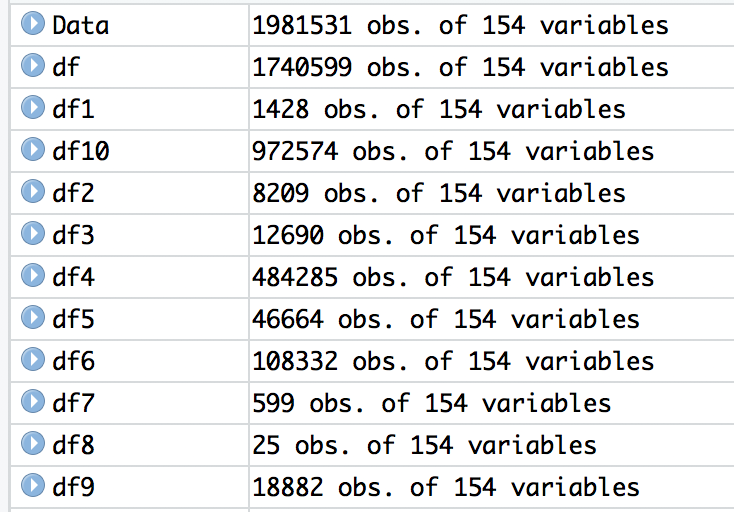
***Alpha = 0.03***

******

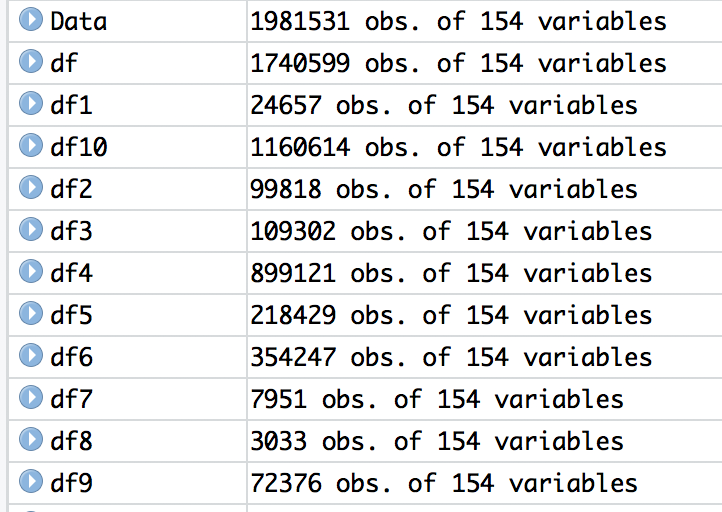
***Alpha = 0.04***

******

***Alpha = 0.05***

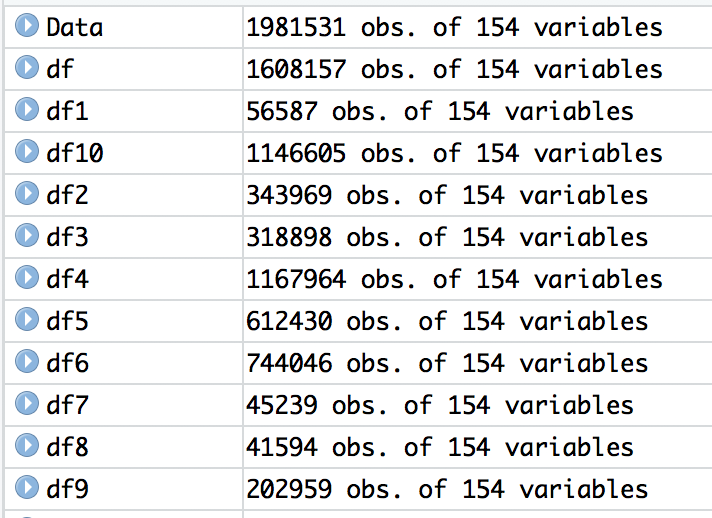
******

***Alpha = 0.025***

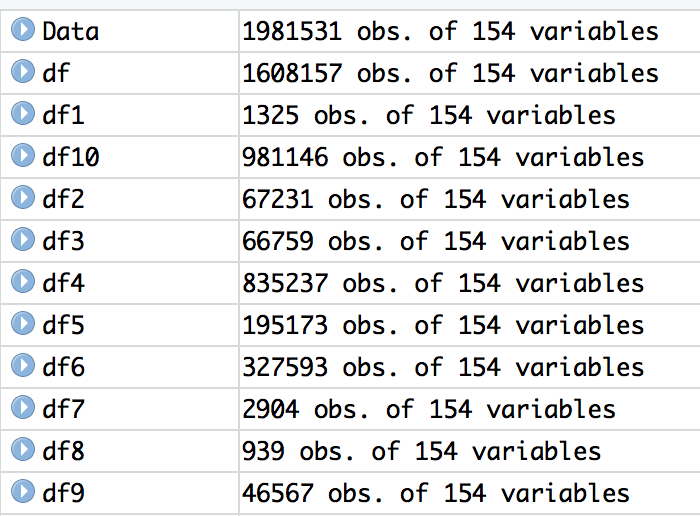
******

***Word Count > 7***

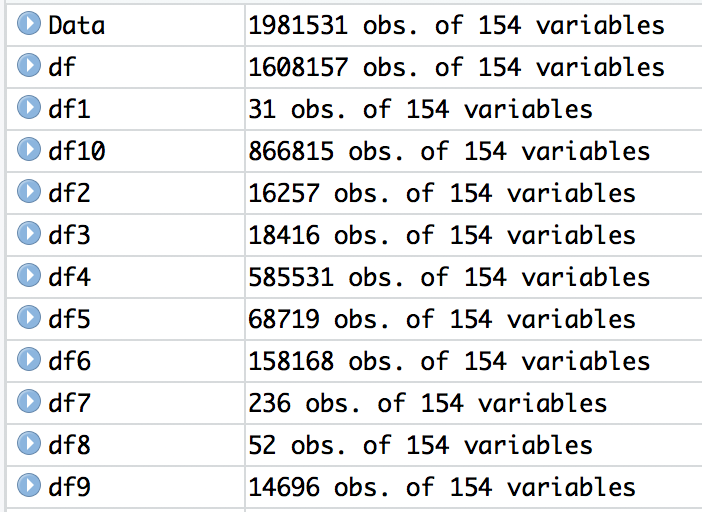
***Alpha = 0.01***

******

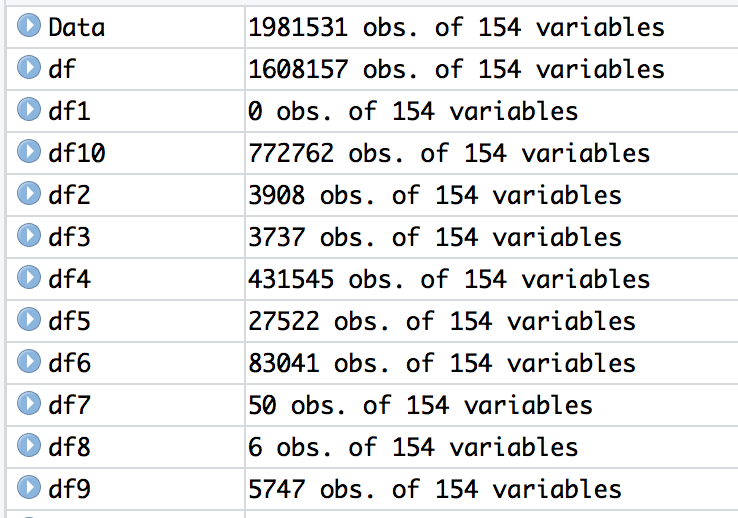
***Alpha = 0.02***

******

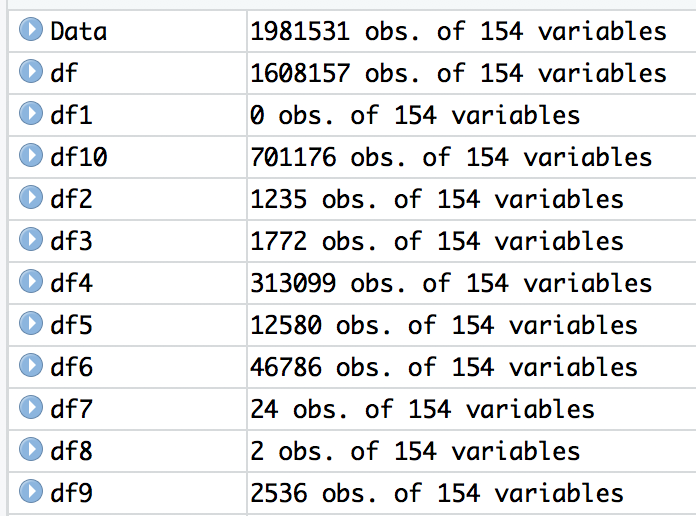
***Alpha = 0.03***

******

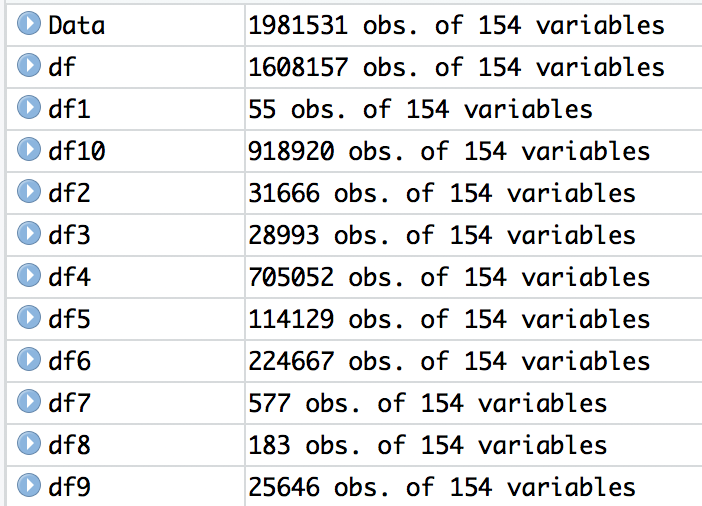
***Alpha = 0.04***

******

***Alpha = 0.05***

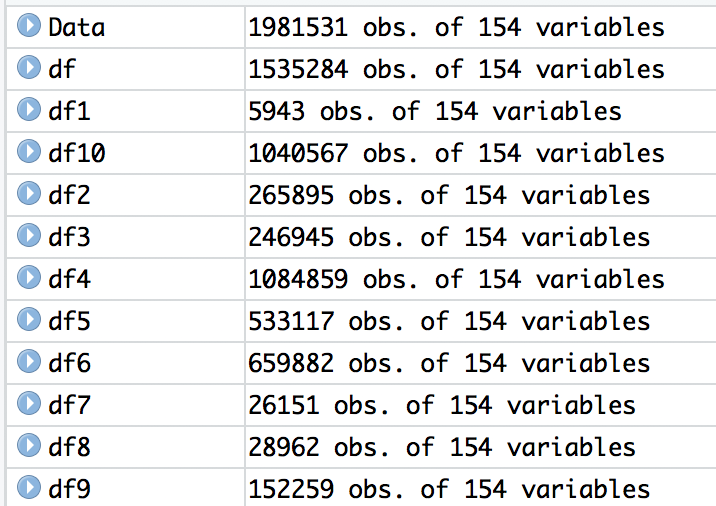
******

***Alpha = 0.025***

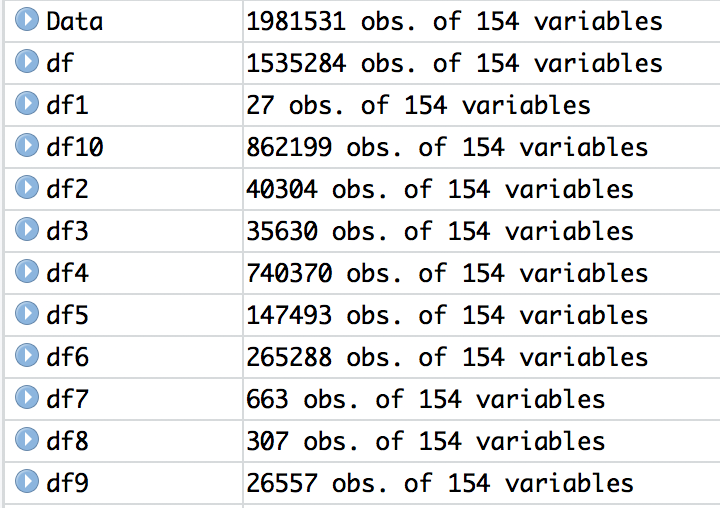
******

***Word Count > 9***

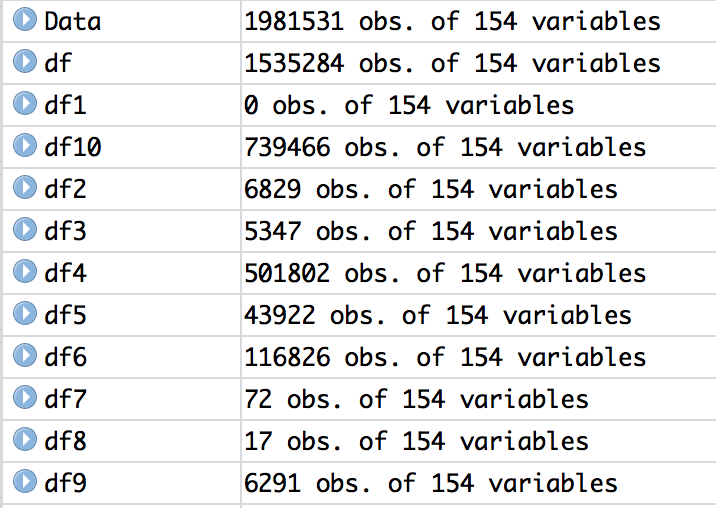
***Alpha = 0.01***

******

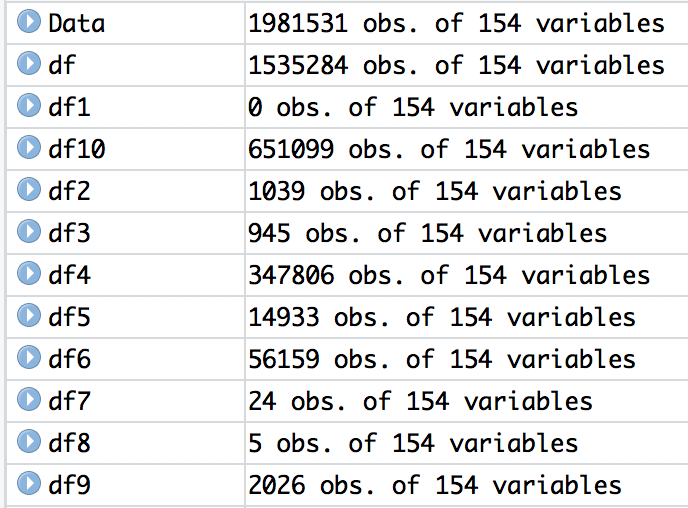
***Alpha = 0.02***

******

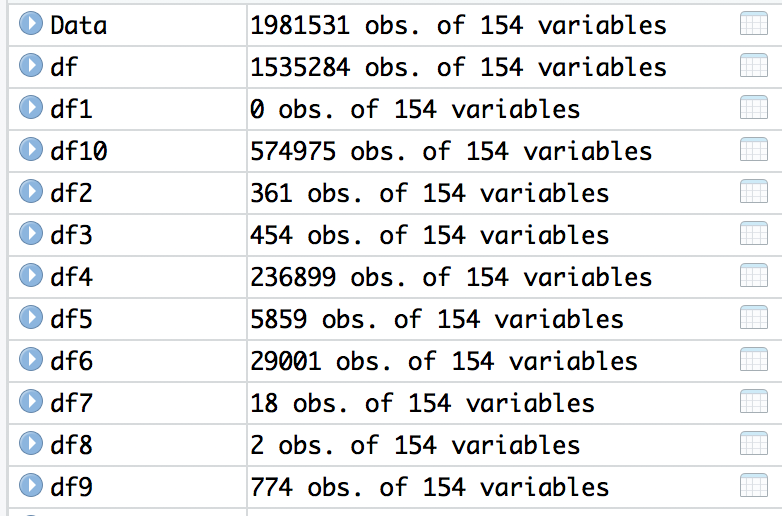
***Alpha = 0.03***

******

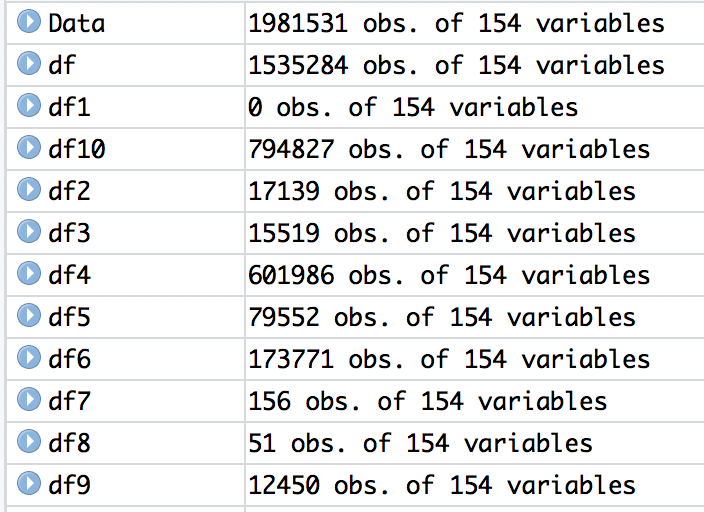
***Alpha = 0.04***

******

***Alpha = 0.05***

******

***Alpha = 0.025***

******

***Alpha equals to 0.04 & 0.025 for word count 4,7 & 9 at following path-***

***D:\WEKA & LIWC\Experiment with different Alpha\Word Count\_4\Alpha=0.04***

***D:\WEKA & LIWC\Experiment with different Alpha\Word Count\_4\Alpha=0.025***

***D:\WEKA & LIWC\Experiment with different Alpha\Word Count\_7\Alpha=0.04***

***D:\WEKA & LIWC\Experiment with different Alpha\Word Count\_7\Alpha=0.025***

***D:\WEKA & LIWC\Experiment with different Alpha\Word Count\_9\Alpha=0.04***

***D:\WEKA & LIWC\Experiment with different Alpha\Word Count\_9\Alpha=0.025***

***File Name - Output.csv***

***This file contains required metrics with values 1(Yes) or 0(No) representing whether review is within the cutoff or not.***