4. Sentiment Analysis of Twitter Dataset

```
import re import pandas as pd  import numpy as np  import matplotlib.pyplot
as plt  import seaborn as sns import string import nltk import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
%matplotlib inline train =
pd.read_csv('https://raw.githubusercontent.com/dD2405/Twitter_Sentiment_An
alysis/master/train.csv') train_original=train.copy() test =
pd.read_csv('https://raw.githubusercontent.com/dD2405/Twitter_Sentiment_An
alysis/master/test.csv') test_original=test.copy()
```

STEP — 1 :
Combine the train.csv and test.csv files.

```
combine = train.append(test,ignore_index=True,sort=True)
```

STEP — 2
Removing Twitter Handles(@User)

```
def remove_pattern(text,pattern):

   # re.findall() finds the pattern i.e @user and puts it in a list for further task     r =
re.findall(pattern,text)


   # re.sub() removes @user from the sentences in the dataset     for i in r:         text =
re.sub(i,"",text)     return text combine['Tidy_Tweets'] =
np.vectorize(remove_pattern)(combine['tweet'],
"@[\w]*") combine.head() combine['Tidy_Tweets'] =
combine['Tidy_Tweets'].str.replace("[^a-zA-Z#]", "
")
combine.head(10)
```

STEP — 4
Removing Short Words

```
combine['Tidy_Tweets'] = combine['Tidy_Tweets'].apply(lambda x: ' '.join([w for w in
x.split() if len(w)>3])) combine.head(10)
```

STEP — 5 Tokenization tokenized_tweet =

```
combine['Tidy_Tweets'].apply(lambda x: x.split()) tokenized_tweet.head()
```

STEP — 6
Stemming

```
from nltk import PorterStemmer

ps = PorterStemmer()

tokenized_tweet = tokenized_tweet.apply(lambda x: [ps.stem(i) for i in x])

tokenized_tweet.head() for i in
range(len(tokenized_tweet)):
   tokenized_tweet[i] = ' '.join(tokenized_tweet[i])

combine['Tidy_Tweets'] = tokenized_tweet combine.head()
```

Importing packages necessary for generating a WordCloud

```
from wordcloud import WordCloud,ImageColorGenerator from PIL
import Image import urllib import requests
```

Generating WordCloud for tweets with label ‘0’.
Store all the words from the dataset which are non-racist/sexist.

```
all_words_positive = ' '.join(text for text in combine['Tidy_Tweets'][combine['label']==0]) #
combining the image with the dataset
Mask = np.array(Image.open(requests.get('http://clipart-
library.com/image_gallery2/Twitter-PNG-Image.png', stream=True).raw))

# We use the ImageColorGenerator library from Wordcloud
# Here we take the color of the image and impose it over our wordcloud image_colors =
ImageColorGenerator(Mask)

# Now we use the WordCloud function from the wordcloud library  wc =
WordCloud(background_color='black', height=1500,
width=4000,mask=Mask).generate(all_words_positive)
# Size of the image generated  plt.figure(figsize=(10,20))

# Here we recolor the words from the dataset to the image's color
# recolor just recolors the default colors to the image's blue color
# interpolation is used to smooth the image generated
plt.imshow(wc.recolor(color_func=image_colors),interpolation="hamming")
plt.axis('off') plt.show()

# combining the image with the dataset
Mask = np.array(Image.open(requests.get('http://clipart-
library.com/image_gallery2/Twitter-PNG-Image.png', stream=True).raw))

# We use the ImageColorGenerator library from Wordcloud
# Here we take the color of the image and impose it over our wordcloud image_colors =
ImageColorGenerator(Mask)

# Now we use the WordCloud function from the wordcloud library  wc =
WordCloud(background_color='black', height=1500,
width=4000,mask=Mask).generate(all_words_negative)
# Size of the image generated  plt.figure(figsize=(10,20))

# Here we recolor the words from the dataset to the image's color
# recolor just recolors the default colors to the image's blue color
# interpolation is used to smooth the image generated
plt.imshow(wc.recolor(color_func=image_colors),interpolation="gaussian")

plt.axis('off') plt.show()

Function to extract hashtags from tweets def
Hashtags_Extract(x):
   hashtags=[]
   # Loop over the words in the tweet     for i in
x:
      ht = re.findall(r'#(\w+)',i)        hashtags.append(ht)        return hashtags ht_positive =
Hashtags_Extract(combine['Tidy_Tweets'][combine['label']==0]) ht_positive
ht_positive_unnest = sum(ht_positive,[]) ht_negative =
Hashtags_Extract(combine['Tidy_Tweets'][combine['label']==1]) ht_negative
word_freq_positive = nltk.FreqDist(ht_positive_unnest) word_freq_positive df_positive =
pd.DataFrame({'Hashtags':list(word_freq_positive.keys()),'Count':list(word_fre
q_positive.values())}) df_positive.head(10) df_positive_plot =
df_positive.nlargest(20,columns='Count')
sns.barplot(data=df_positive_plot,y='Hashtags',x='Count') sns.despine() word_freq_negative
= nltk.FreqDist(ht_negative_unnest) word_freq_negative df_negative =
pd.DataFrame({'Hashtags':list(word_freq_negative.keys()),'Count':list(word_fre
q_negative.values())}) df_negative.head(10) df_negative_plot =
df_negative.nlargest(20,columns='Count')
sns.barplot(data=df_negative_plot,y='Hashtags',x='Count') sns.despine()
from sklearn.feature_extraction.text import CountVectorizer
bow_vectorizer = CountVectorizer(max_df=0.90, min_df=2, max_features=1000,
stop_words='english')
# bag-of-words feature matrix
bow = bow_vectorizer.fit_transform(combine['Tidy_Tweets'])
df_bow = pd.DataFrame(bow.todense())
df_bow
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf=TfidfVectorizer(max_df=0.90, min_df=2,max_features=1000,stop_words='english')

tfidf_matrix=tfidf.fit_transform(combine['Tidy_Tweets'])

df_tfidf = pd.DataFrame(tfidf_matrix.todense())

df_tfidf
train_bow = bow[:31962]

train_bow.todense()
train_tfidf_matrix = tfidf_matrix[:31962]

train_tfidf_matrix.todense()
from sklearn.model_selection import train_test_split
Bag-of-Words Features
x_train_bow, x_valid_bow, y_train_bow, y_valid_bow =
train_test_split(train_bow,train['label'],test_size=0.3,random_state=2)
TF-IDF features
x_train_tfidf, x_valid_tfidf, y_train_tfidf, y_valid_tfidf =
train_test_split(train_tfidf_matrix,train['label'],test_size=0.3,random_state=17)
```

```
from sklearn.metrics import f1_score
```

Logistic Regression

The first model we are going to use is Logistic Regression.

```
from sklearn.linear_model import LogisticRegression
Log_Reg = LogisticRegression(random_state=0,solver='lbfgs')
```

Bag-of-Words Features

Fitting the Logistic Regression Model.

```
Log_Reg.fit(x_train_bow,y_train_bow)
```

Predicting the probabilities.

```
prediction_bow = Log_Reg.predict_proba(x_valid_bow)

prediction_bow
# if prediction is greater than or equal to 0.3 than 1 else 0
# Where 0 is for positive sentiment tweets and 1 for negative sentiment tweets
prediction_int = prediction_bow[:,1]>=0.3

# converting the results to integer type
prediction_int = prediction_int.astype(np.int)
prediction_int

# calculating f1 score
log_bow = f1_score(y_valid_bow, prediction_int)

log_bow
```

Fitting the Logistic Regression Model.

```
Log_Reg.fit(x_train_tfidf,y_train_tfidf)
```

Predicting the probabilities.

```
prediction_tfidf = Log_Reg.predict_proba(x_valid_tfidf)

prediction_tfidf
# if prediction is greater than or equal to 0.3 than 1 else 0
# Where 0 is for positive sentiment tweets and 1 for negative sentiment tweets
prediction_int = prediction_tfidf[:,1]>=0.3
prediction_int = prediction_int.astype(np.int)
prediction_int

# calculating f1 score
log_tfidf = f1_score(y_valid_tfidf, prediction_int)

log_tfidf
```
The next model we use is XGBoost.
```
from xgboost import XGBClassifier
```
Bag-of-Words Features
```
model_bow = XGBClassifier(random_state=22,learning_rate=0.9)
```
Fitting the XGBoost Model
```
model_bow.fit(x_train_bow, y_train_bow)
```
Predicting the probabilities.
```
xgb = model_bow.predict_proba(x_valid_bow)

xgb
# if prediction is greater than or equal to 0.3 than 1 else 0
# Where 0 is for positive sentiment tweets and 1 for negative sentiment tweets
xgb=xgb[:,1]>=0.3

# converting the results to integer type
xgb_int=xgb.astype(np.int)

# calculating f1 score
xgb_bow=f1_score(y_valid_bow,xgb_int)

xgb_bow
```
TF-IDF Features
```
model_tfidf = XGBClassifier(random_state=29,learning_rate=0.7)
```
Fitting the XGBoost model
```
model_tfidf.fit(x_train_tfidf, y_train_tfidf)
```
Predicting the probabilities.
```
xgb_tfidf=model_tfidf.predict_proba(x_valid_tfidf)
```

```
xgb_tfidf
```

The last model we use is Decision Trees.

```
from sklearn.tree import DecisionTreeClassifier
dct = DecisionTreeClassifier(criterion='entropy', random_state=1)
```

Bag-of-Words Features

Fitting the Decision Tree model.

```
dct.fit(x_train_bow,y_train_bow)
```

Predicting the probabilities.

```
dct_bow = dct.predict_proba(x_valid_bow)

dct_bow
dct.fit(x_train_tfidf,y_train_tfidf)
```

Predicting the probabilities.

```
dct_tfidf = dct.predict_proba(x_valid_tfidf)

dct_tfidf
# if prediction is greater than or equal to 0.3 than 1 else 0
# Where 0 is for positive sentiment tweets and 1 for negative sentiment tweets
dct_tfidf=dct_tfidf[:,1]>=0.3

# converting the results to integer type
dct_int_tfidf=dct_tfidf.astype(np.int)

# calculating f1 score
dct_score_tfidf=f1_score(y_valid_tfidf,dct_int_tfidf)

dct_score_tfidf
```

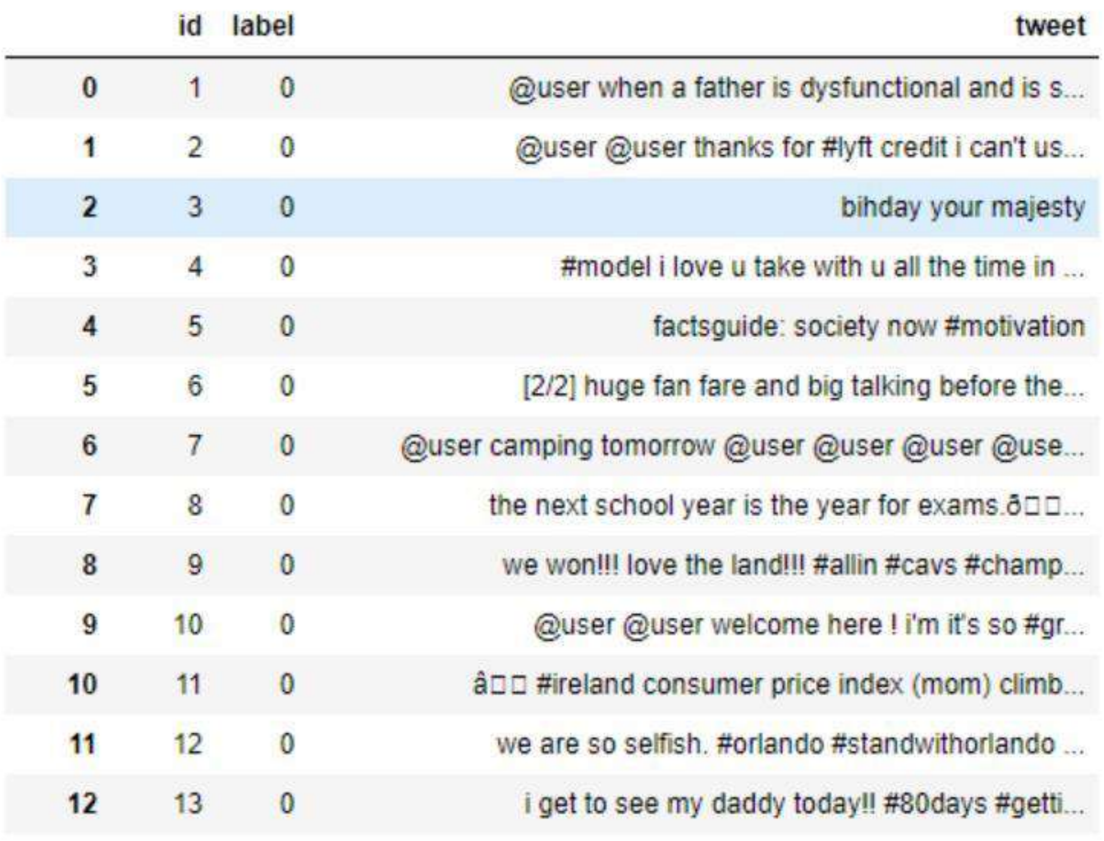
Model Comparison

Now, let us compare the different models we have applied on our dataset with different word embedding techniques.

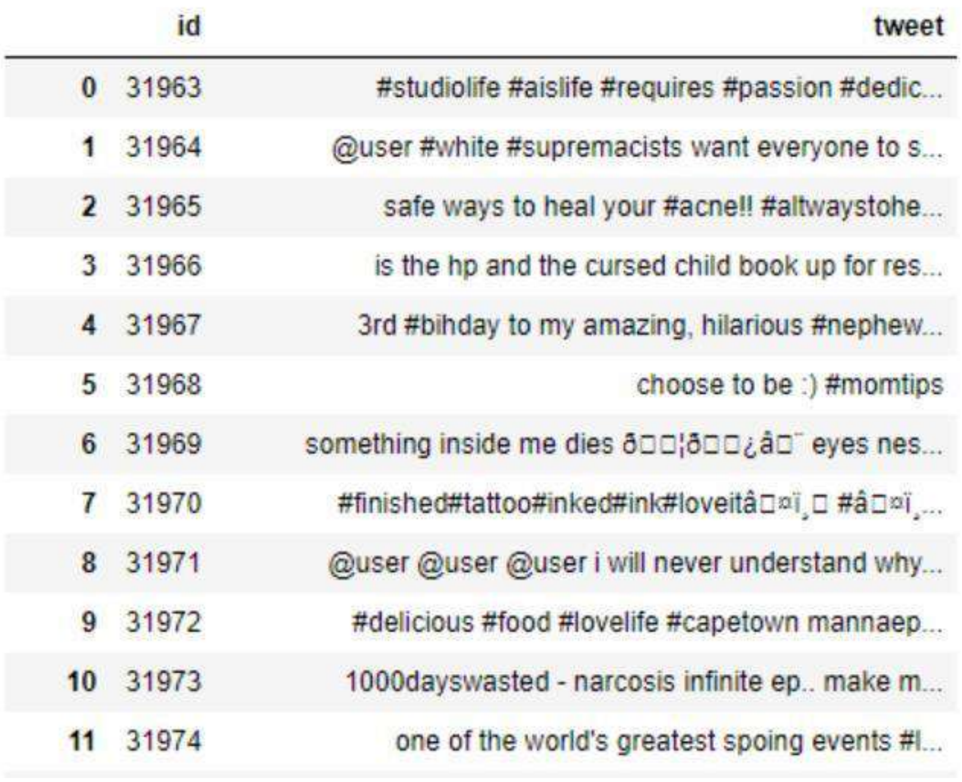
Bag-of-Words

```
Algo_1 = ['LogisticRegression(Bag-of-Words)','XGBoost(Bag-of-
Words)','DecisionTree(Bag-of-Words)']

score_1 = [log_bow,xgb_bow,dct_score_bow]

compare_1 = pd.DataFrame({'Model':Algo_1,'F1_Score':score_1},index=[i for i in
range(1,4)])

compare_1.T
plt.figure(figsize=(18,5))

sns.pointplot(x='Model',y='F1_Score',data=compare_1)

plt.title('Bag-of-Words')
plt.xlabel('MODEL')
plt.ylabel('SCORE')

plt.show()
Algo_2 = ['LogisticRegression(TF-IDF)','XGBoost(TF-IDF)','DecisionTree(TF-IDF)']

score_2 = [log_tfidf,score,dct_score_tfidf]

compare_2 = pd.DataFrame({'Model':Algo_2,'F1_Score':score_2},index=[i for i in
range(1,4)])

compare_2.T
plt.figure(figsize=(18,5))

sns.pointplot(x='Model',y='F1_Score',data=compare_2)

plt.title('TF-IDF')
plt.xlabel('MODEL')
plt.ylabel('SCORE')
plt.show()

Algo_best = ['LogisticRegression(Bag-of-Words)','LogisticRegression(TF-IDF)']

score_best = [log_bow,log_tfidf]

compare_best = pd.DataFrame({'Model':Algo_best,'F1_Score':score_best},index=[i for i in
range(1,3)])

compare_best.T
plt.figure(figsize=(18,5))

sns.pointplot(x='Model',y='F1_Score',data=compare_best)

plt.title('Logistic Regression(Bag-of-Words & TF-IDF)')
plt.xlabel('MODEL')
plt.ylabel('SCORE')

plt.show()
test_tfidf = tfidf_matrix[31962:]
test_pred = Log_Reg.predict_proba(test_tfidf)

test_pred_int = test_pred[:,1] >= 0.3
test_pred_int = test_pred_int.astype(np.int)

test['label'] = test_pred_int

submission = test[['id','label']]
submission.to_csv('result.csv', index=False)
res = pd.read_csv('result.csv')
res
```

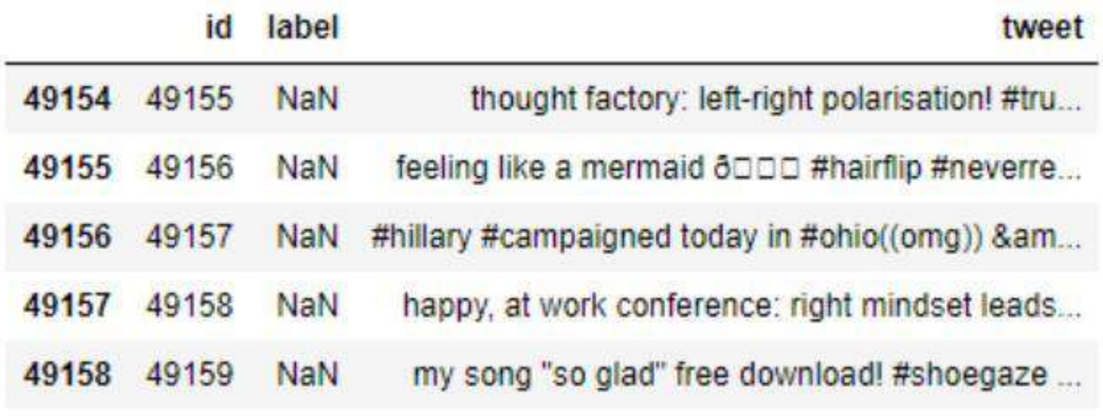
Output-



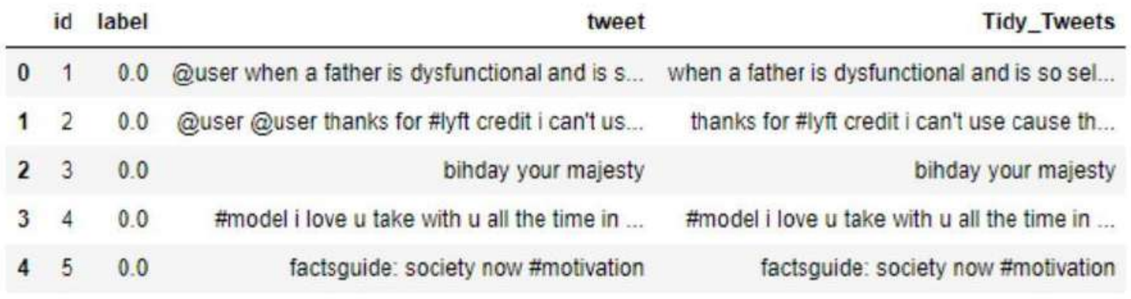
	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation
5	6	0	[2/2] huge fan fare and big talking before the...
6	7	0	@user camping tomorrow @user @user @user @use...
7	8	0	the next school year is the year for exams.ð□□...
8	9	0	we won!!! love the land!!! #allin #cavs #champ...
9	10	0	@user @user welcome here ! i'm it's so #gr...
10	11	0	â□□ #ireland consumer price index (mom) climb...
11	12	0	we are so selfish. #orlando #standwithorlando ...
12	13	0	i get to see my daddy today!! #80days #getti...



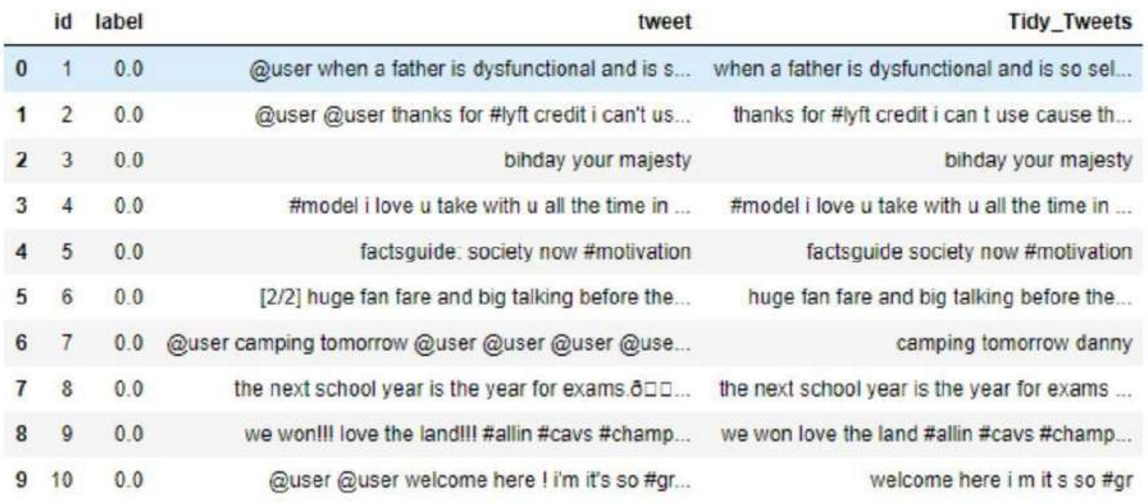
	id	tweet
0	31963	#studiolife #aislife #requires #passion #dedic...
1	31964	@user #white #supremacists want everyone to s...
2	31965	safe ways to heal your #acne!! #altwaystohe...
3	31966	is the hp and the cursed child book up for res...
4	31967	3rd #bihday to my amazing, hilarious #nephew...
5	31968	choose to be :) #momtips
6	31969	something inside me dies ð□□¦ð□□¿â□¨ eyes nes...
7	31970	#finished#tattoo#inked#ink#loveitâ□¤ï¸□ #â□¤ï¸...
8	31971	@user @user @user i will never understand why...
9	31972	#delicious #food #lovelife #capetown mannaep...
10	31973	1000dayswasted - narcosis infinite ep.. make m...
11	31974	one of the world's greatest spoing events #l...



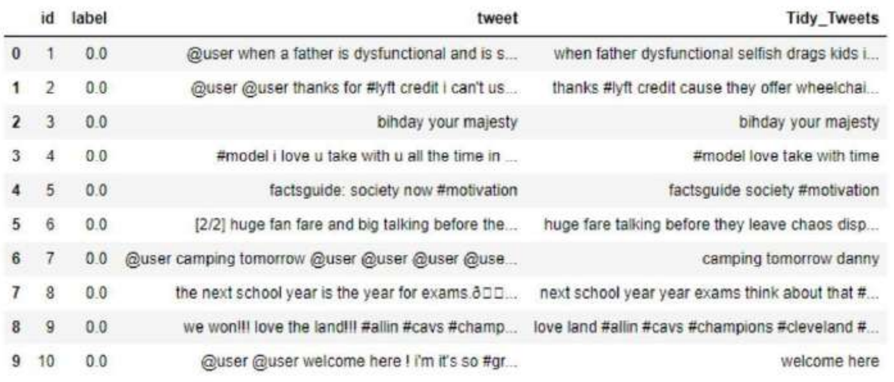
	id	label	tweet
49154	49155	NaN	thought factory: left-right polarisation! #tru...
49155	49156	NaN	feeling like a mermaid ð□□□ #hairflip #neverre...
49156	49157	NaN	#hillary #campaigned today in #ohio((omg)) &am...
49157	49158	NaN	happy, at work conference: right mindset leads...
49158	49159	NaN	my song "so glad" free download! #shoegaze ...



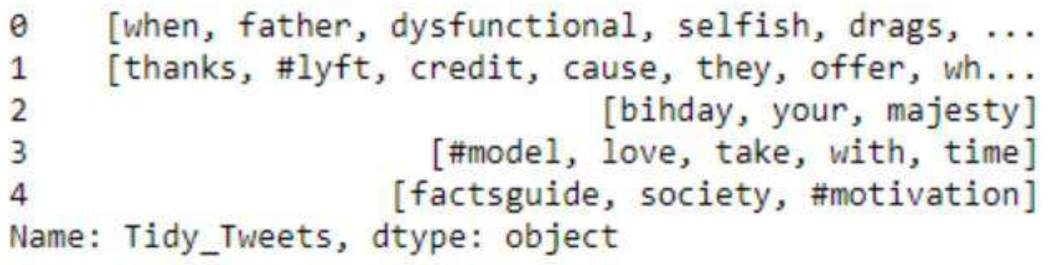
	id	label	tweet	Tidy_Tweets
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i can't use cause th...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model i love u take with u all the time in ...
4	5	0.0	factsguide: society now #motivation	factsguide: society now #motivation



	id	label	tweet	Tidy_Tweets
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i can t use cause th...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model i love u take with u all the time in ...
4	5	0.0	factsguide: society now #motivation	factsguide society now #motivation
5	6	0.0	[2/2] huge fan fare and big talking before the...	huge fan fare and big talking before the...
6	7	0.0	@user camping tomorrow @user @user @user @use...	camping tomorrow danny
7	8	0.0	the next school year is the year for exams.ð□□...	the next school year is the year for exams ...
8	9	0.0	we won!!! love the land!!! #allin #cavs #champ...	we won love the land #allin #cavs #champ...
9	10	0.0	@user @user welcome here ! i'm it's so #gr...	welcome here i m it s so #gr

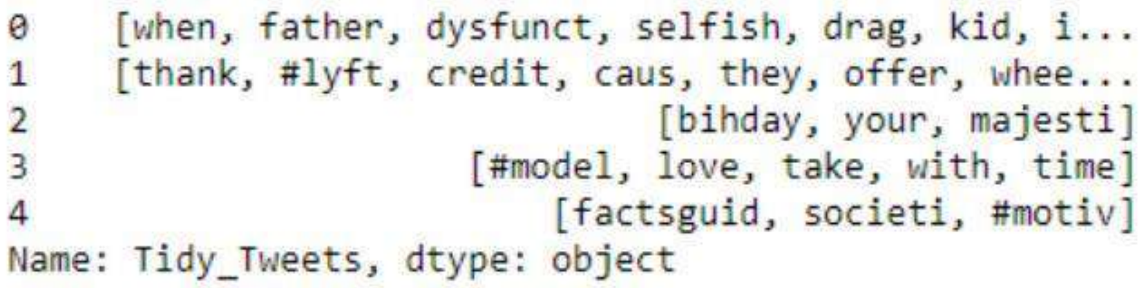


	id	label	tweet	Tidy_Tweets
0	1	0.0	@user when a father is dysfunctional and is s...	when father dysfunctional selfish drags kids i...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks #lyft credit cause they offer wheelchai...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model love take with time
4	5	0.0	factsguide: society now #motivation	factsguide society #motivation
5	6	0.0	[2/2] huge fan fare and big talking before the...	huge fare talking before they leave chaos disp...
6	7	0.0	@user camping tomorrow @user @user @user @use...	camping tomorrow danny
7	8	0.0	the next school year is the year for exams.ð□□...	next school year year exams think about that #...
8	9	0.0	we won!!! love the land!!! #allin #cavs #champ...	love land #allin #cavs #champions #cleveland #...
9	10	0.0	@user @user welcome here ! i'm it's so #gr...	welcome here

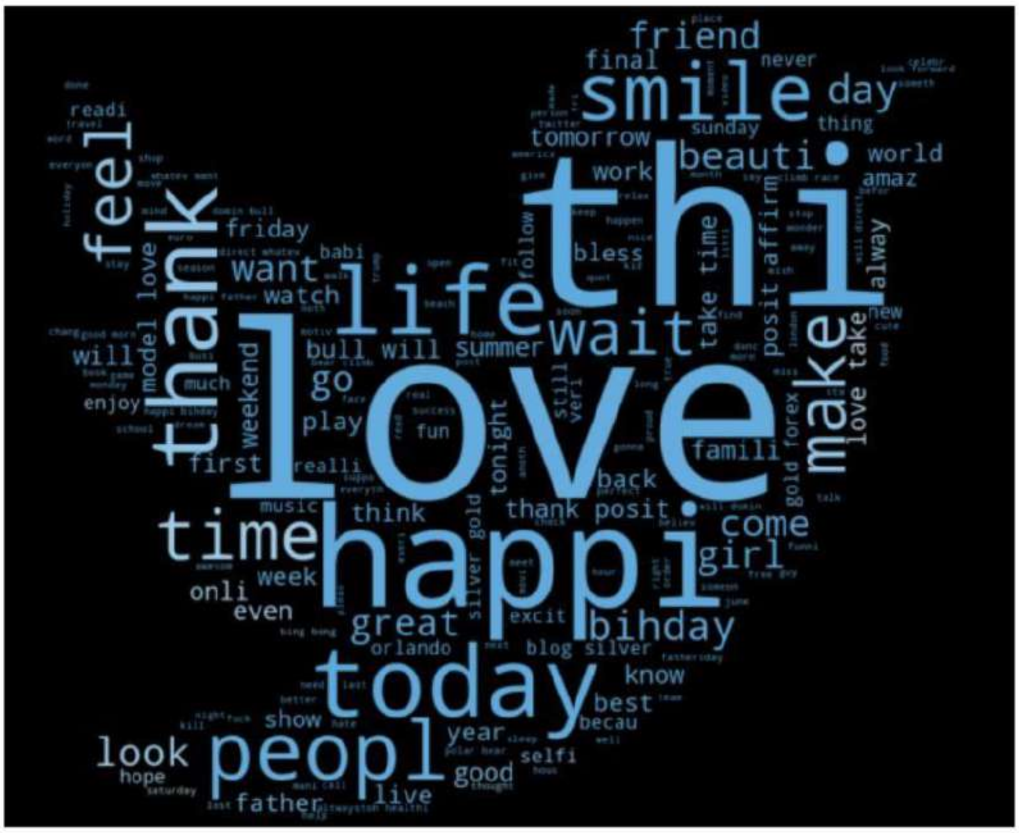


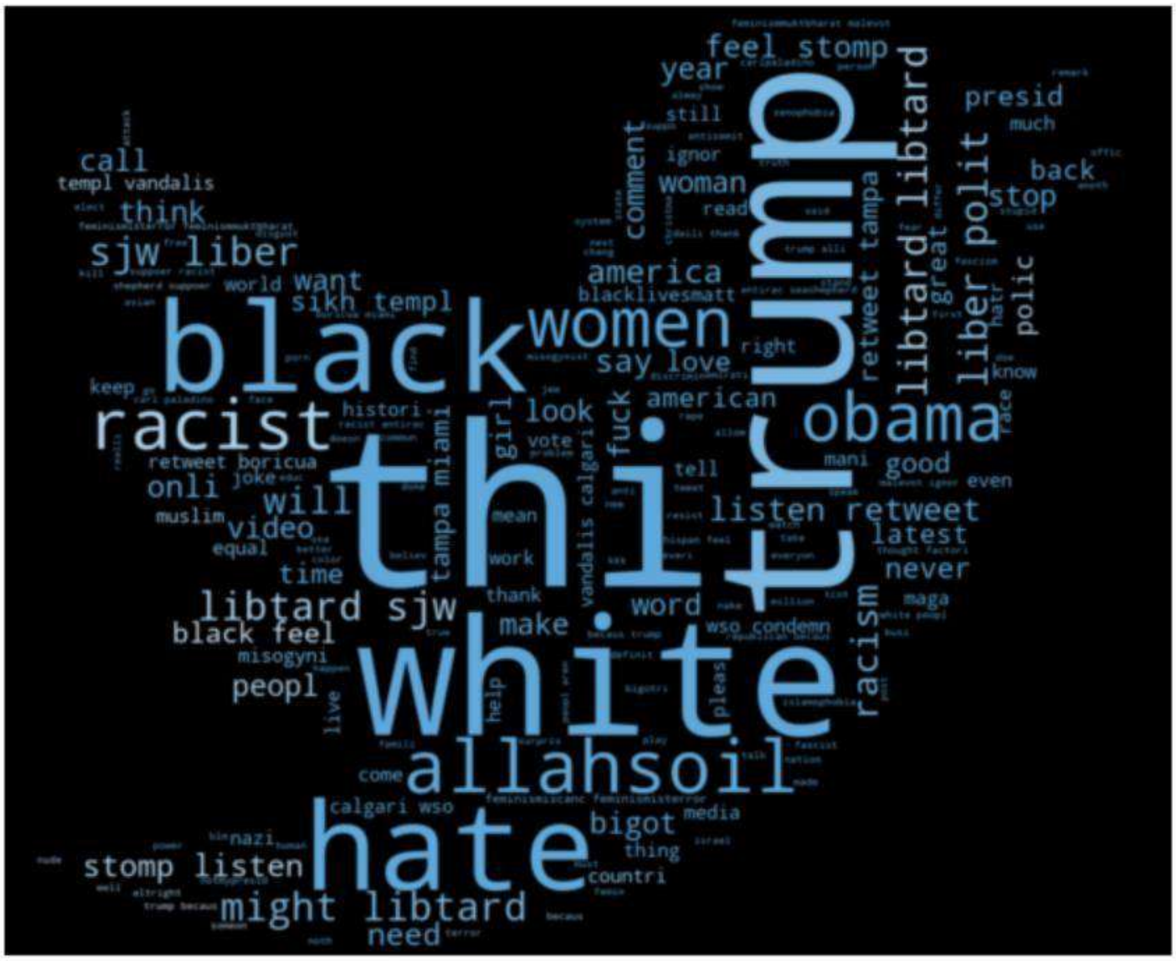
```
0    [when, father, dysfunctional, selfish, drags, ...
1    [thanks, #lyft, credit, cause, they, offer, wh...
2                              [bihday, your, majesty]
3                       [#model, love, take, with, time]
4                     [factsguide, society, #motivation]
Name: Tidy_Tweets, dtype: object
```

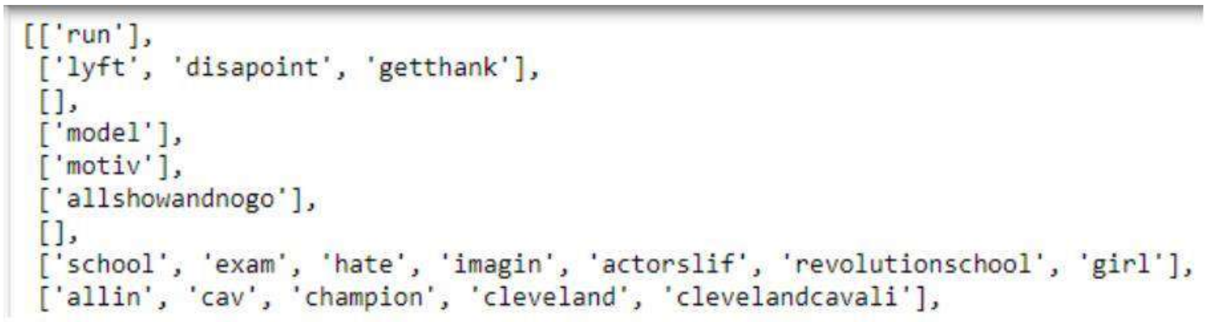
Results after tokenization



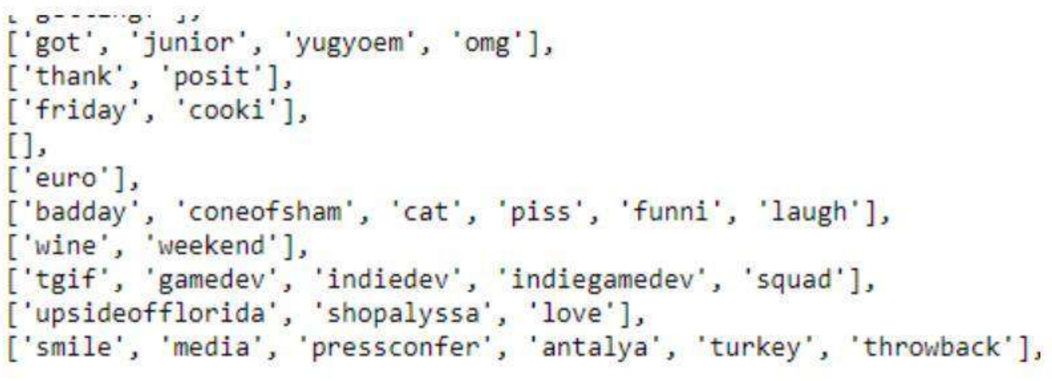
```
0    [when, father, dysfunct, selfish, drag, kid, i...
1    [thank, #lyft, credit, caus, they, offer, whee...
2                             [bihday, your, majesti]
3                      [#model, love, take, with, time]
4                           [factsguid, societi, #motiv]
Name: Tidy_Tweets, dtype: object
```


love
thi
happi
today
peopl
thank
life
smile
time
wait
make
feel
friend
beauti
day
bihday
great
look
father

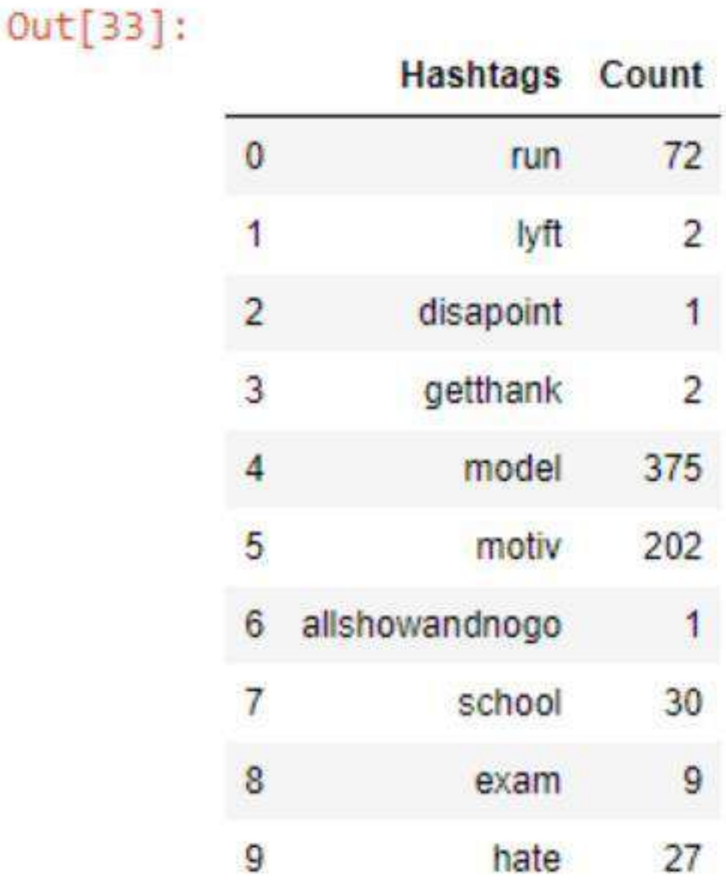




```
[['run'],
 ['lyft', 'disapoint', 'getthank'],
 [],
 ['model'],
 ['motiv'],
 ['allshowandnogo'],
 [],
 ['school', 'exam', 'hate', 'imagin', 'actorslif', 'revolutionschool', 'girl'],
 ['allin', 'cav', 'champion', 'cleveland', 'clevelandcavali'],
```

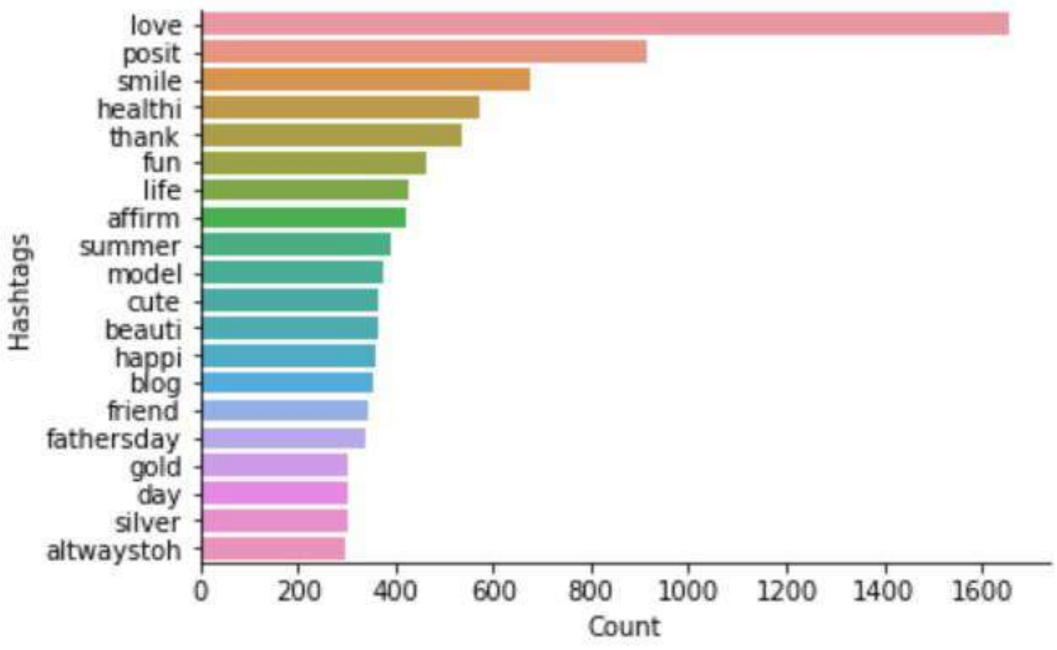


```
['got', 'junior', 'yugyoem', 'omg'],
['thank', 'posit'],
['friday', 'cooki'],
[],
['euro'],
['badday', 'coneofsham', 'cat', 'piss', 'funni', 'laugh'],
['wine', 'weekend'],
['tgif', 'gamedev', 'indiedev', 'indiegamedev', 'squad'],
['upsideofflorida', 'shopalyssa', 'love'],
['smile', 'media', 'pressconfer', 'antalya', 'turkey', 'throwback'],
```

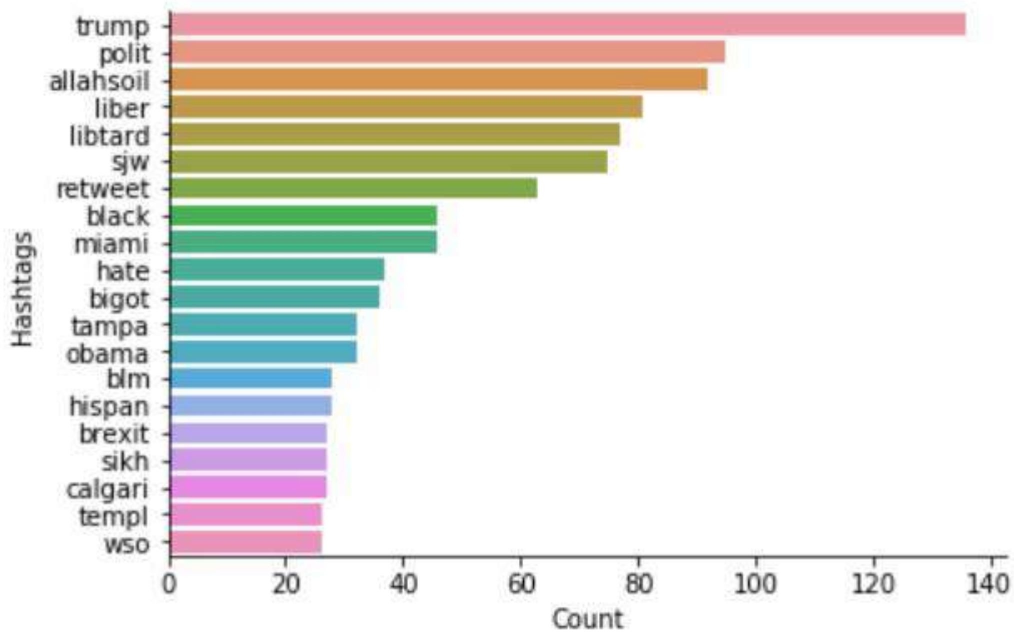


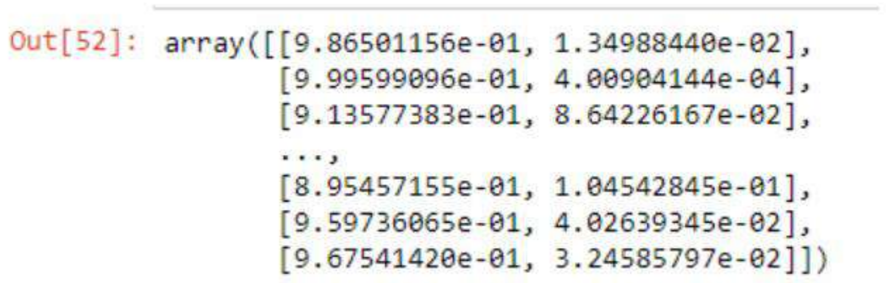
Out[33]:

	Hashtags	Count
0	run	72
1	lyft	2
2	disapoint	1
3	getthank	2
4	model	375
5	motiv	202
6	allshowandnogo	1
7	school	30
8	exam	9
9	hate	27



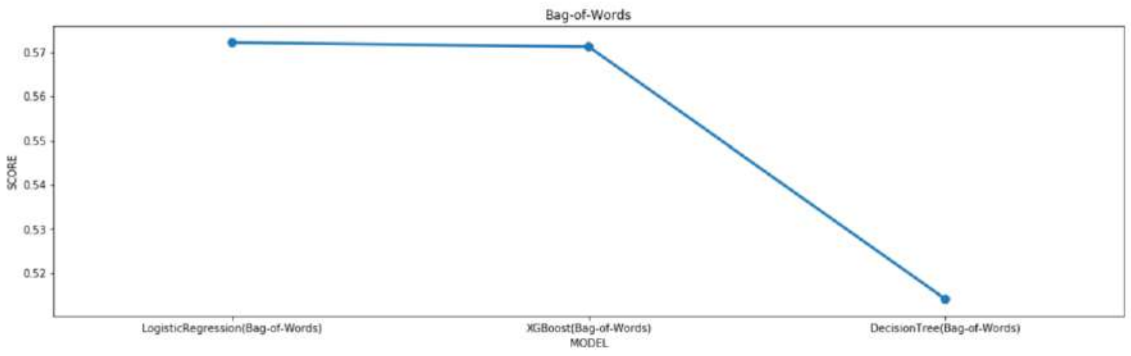
Count BarPlot





```
Out[52]: array([[9.86501156e-01, 1.34988440e-02],
                [9.99599096e-01, 4.00904144e-04],
                [9.13577383e-01, 8.64226167e-02],
                ...,
                [8.95457155e-01, 1.04542845e-01],
                [9.59736065e-01, 4.02639345e-02],
                [9.67541420e-01, 3.24585797e-02]])
```

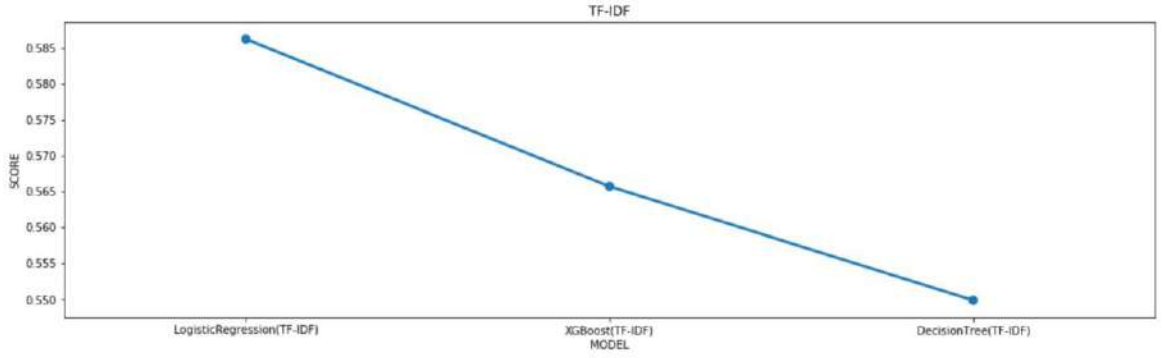
Predicting the probabilities for a tweet falling into either Positive or Negative class.



TF-IDF

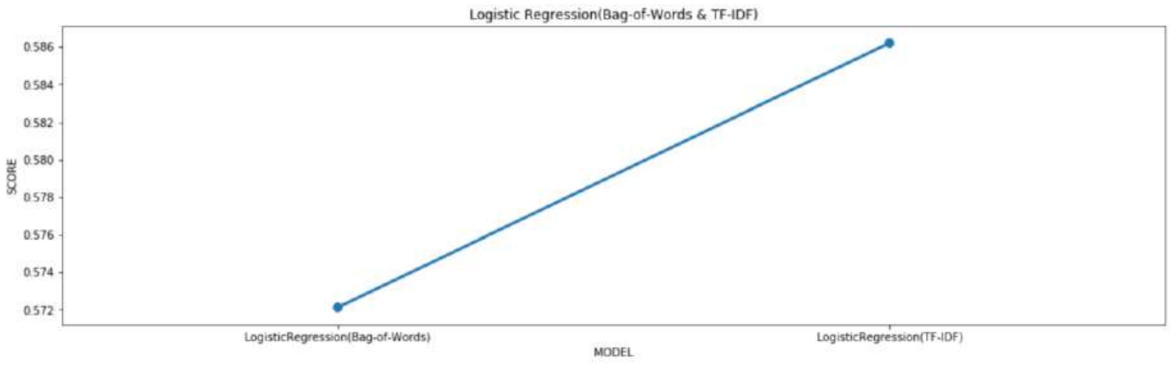
Out[79]:

	1	2	3
Model	LogisticRegression(TF-IDF)	XGBoost(TF-IDF)	DecisionTree(TF-IDF)
F1_Score	0.586207	0.565705	0.549882



Out[81]:

	1	2
Model	LogisticRegression(Bag-of-Words)	LogisticRegression(TF-IDF)
F1_Score	0.572135	0.586207



```
Out[82]:
              id  label
       0   31963      0
       1   31964      0
       2   31965      0
       3   31966      0
       4   31967      0
       5   31968      0
       6   31969      0
       7   31970      0
       8   31971      0
       9   31972      0
      10   31973      0
      11   31974      0
      12   31975      0
      13   31976      0
```