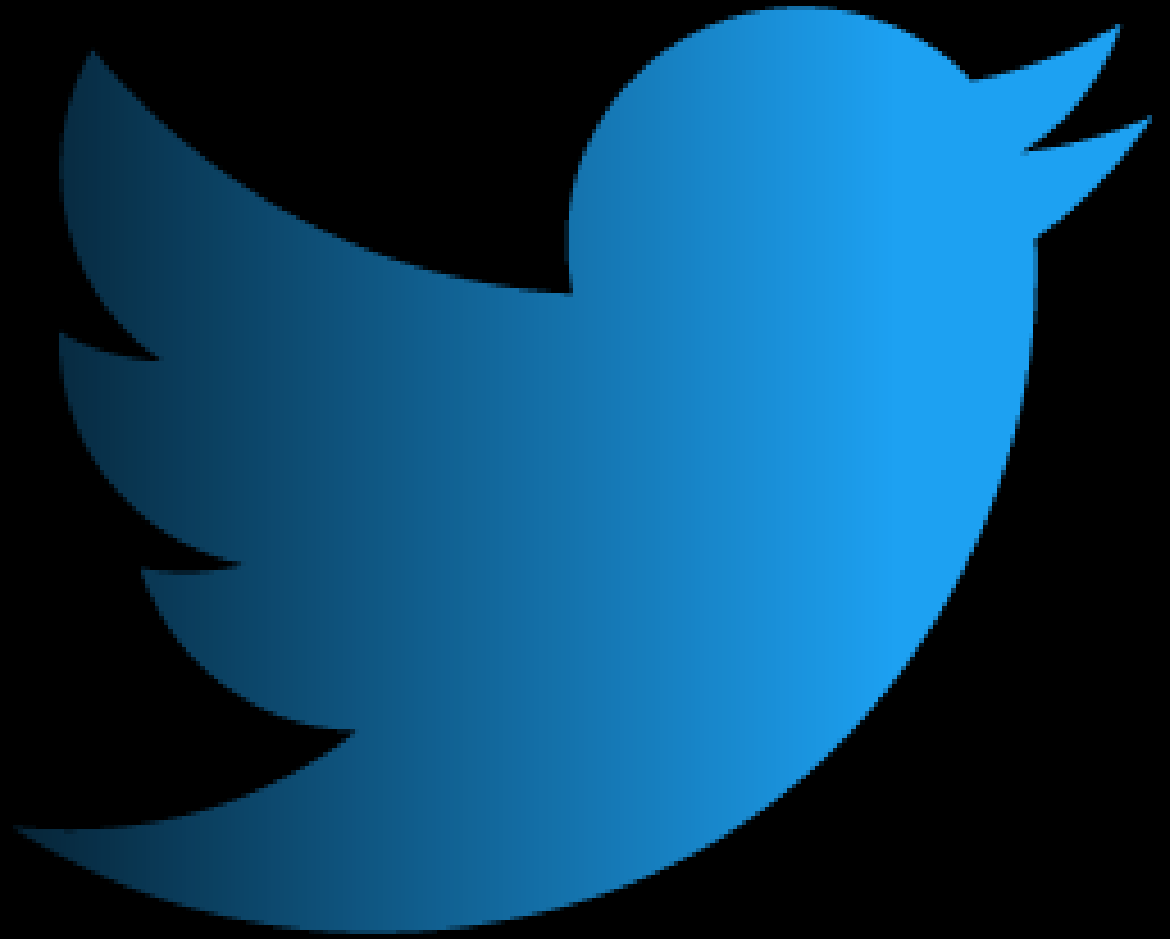
Twitter Sentiment Analysis for Product Review – iPhone 11



Made by:

Kishor Murade: 112003091

Pratik Sarode: 112003124

Rushikesh Pingle : 112003113

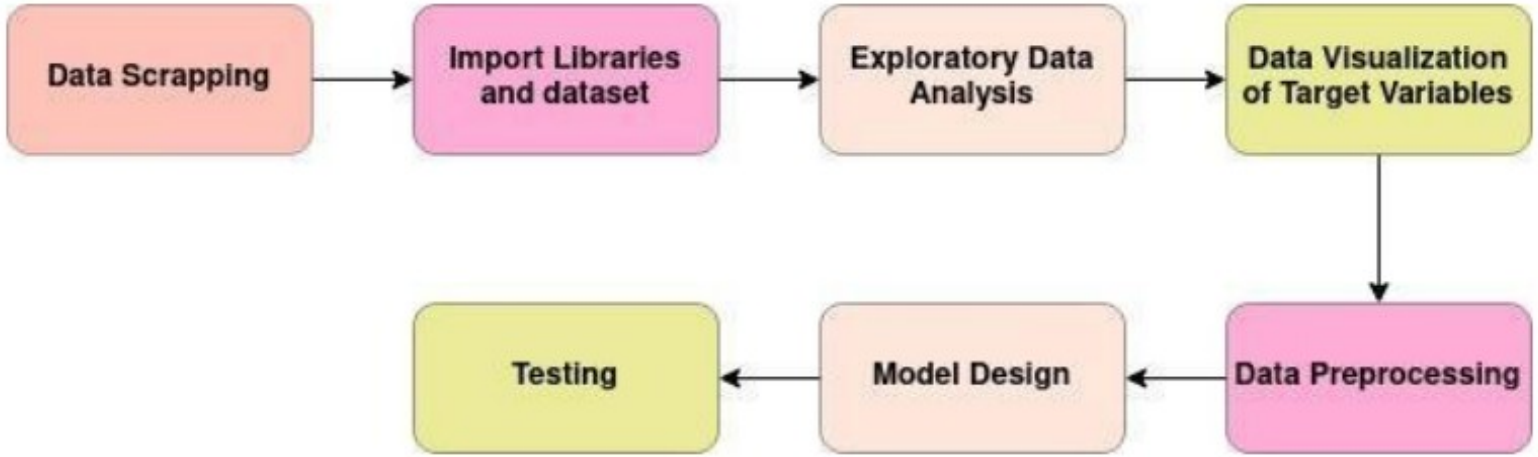
Introduction

- Sentiment analysis refers to identifying as well as classifying the sentiments that are expressed in the text source.
- Tweets are often useful in generating a vast amount of sentiment data upon analysis.
- These data are useful in understanding the opinion of the people about a variety of topics.
- Therefore, we need to develop an Automated Machine Learning Sentiment Analysis Model in-order to compute the customer perception.

Project Pipeline

Contents:

- Web Scraping
- Data preprocessing
- Data visualization
- Model Building
- Testing
- Conclusion



Web Scrapping

```python
header = ['id', 'username', 'content', 'date', 'like', 'reply', 'retweet']
fp = open('iPhone.csv', 'w', encoding='UTF8', newline='')
writer = csv.DictWriter(fp, fieldnames=header)
writer.writeheader()

limit = 15000

for i, tweet in enumerate(snscrape.modules.twitter.TwitterSearchScraper('#iphone11 since:2020-12-30 lang:en').get_items()):
    if i > limit:
        break
    writer.writerows([{'id': tweet.id,
                       'username': tweet.user.username,
                       'content': tweet.content,
                       'date': tweet.date,
                       'like': tweet.likeCount,
                       'reply': tweet.replyCount,
                       'retweet': tweet.retweetCount}])
```

Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19993 entries, 0 to 19992
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        19993 non-null  int64
 1   username  19993 non-null  object
 2   content   19993 non-null  object
 3   date      19993 non-null  object
 4   like      19993 non-null  int64
 5   reply     19993 non-null  int64
 6   retweet   19993 non-null  int64
dtypes: int64(4), object(3)
memory usage: 1.1+ MB
```

Dataset

Data Scrapping:

- Site referred: twitter.com
- Language used: Python
- Library used : twitter Search Scraper

Data Set:

- Number of Attributes: 7
- Number of Instances: 14994

	id	username	content	date	like	reply	retweet
0	1649023985798004736	YaD3v	Download #UTG pro to get all #video_games info...	2023-04-20 12:15:30+00:00	0	0	0
1	1648944166662209536	WuKiana1	Silicone mobile phone case 3D magnetic suction...	2023-04-20 06:58:19+00:00	0	0	0
2	1648918260946202624	yahakunbaru	wts want to sell jual murahhhh iphone 11 tosca...	2023-04-20 05:15:23+00:00	0	1	0
3	1648668628580016128	GeniusPhone_R	$10 Off an iPhone 11 (All Models) Lifetime Bat...	2023-04-19 12:43:26+00:00	0	0	0
4	1648661583072047105	YaD3v	Download #UTG pro to get all #video_games info...	2023-04-19 12:15:26+00:00	0	0	0
5	1648623627124854786	FonezworldAK	⭐ IPhone 11 64GB New only €479 ⭐\n💡NO FIX NO F...	2023-04-19 09:44:37+00:00	0	0	0
6	1648600538433351680	jakeyeology	UP UP UP #wts #ph #lfb #iPhone #iPhone11	2023-04-19 08:12:52+00:00	0	0	0
7	1648583621169512448	mattr	Birdman. \n\n#dublin #streetphotography #stree...	2023-04-19 07:05:39+00:00	5	0	0
8	1648579818085298177	WuKiana1	Electroplated mobile phone case with magnetic ...	2023-04-19 06:50:32+00:00	0	0	0
9	1648578989072719872	WuKiana1	Electroplated colorful mobile phone case.\n#iP...	2023-04-19 06:47:14+00:00	0	0	0
10	1648541291251154944	38jie	Apple iPhone 11 by Studio7\n฿16,600\nพิกัด👇👇\n...	2023-04-19 04:17:26+00:00	0	1	0
11	1648452145925218304	onspot_repair	Mobile Repair Van Solution 📞📱\n🕸https://t.co...	2023-04-18 22:23:13+00:00	1	0	0

Data Preprocessing

Functions :

- Remove NON-ASCII characters
- Mark emoticons as happy or sad.
  ex:- :) -> Happy , :(->Sad.
- Replace emojis with their meaning
- Remove retweet 'RT'
- Remove the user mentions
- Keeping only the word after the #

```
def data_cleaning(text):

    #remove NON-ASCII characters
    text = re.sub('\\\\u[0-9A-Fa-f]{4}','', text

    #mark emoticons as happy or sad
    text = emos_replace(text)

    #replace emojis with their meaning
    text = replace_emoji(text)

    # remove retweet 'RT'
    text = re.sub('RT[\s]+', '', text)

    #remove the user mentions
    text = re.sub('@[A-Za-z0-9]+', '', text)

    #remove numbers
    text = re.sub("[0-9]", "", text)

    #Keeping only the word after the #
    text = re.sub('#', '', text)
```

Data Preprocessing

Functions :

- Remove links (URLs/ links)
- Remove punctuations
- Removing HTML garbage
- Replace repeated letters with only two occurrences

Ex :- heeeelllloooo => heelloo

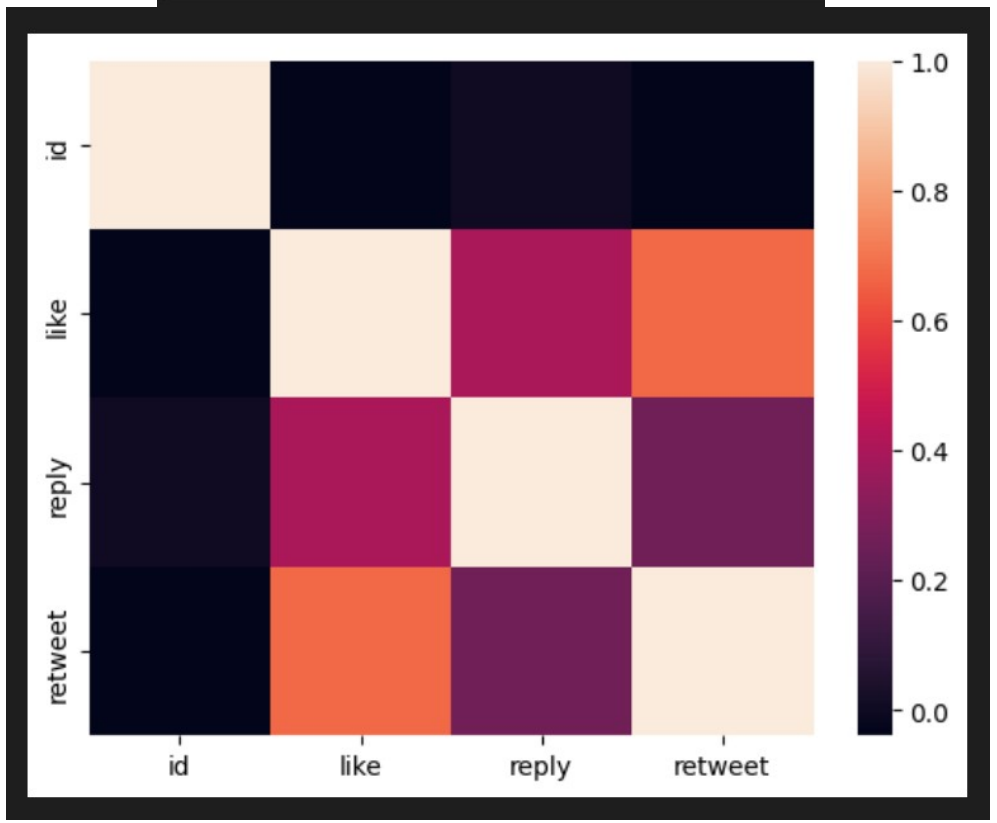
- Remove single letters
- Convert text to lower case text

```
# remove usernames
text = re.sub('@[^\s]+', '', text)
text= re.sub(r"[-\.\n]", "",text)

# remove links (URLs/ links)
text = re.sub('((www\.[^\s]+)|(https?://[^\s]+))', '', text)

# remove punctuations
text = re.sub('[!"$%&\'()*+,-./:@;<=>?[\\]^_`{|}~]', '', text)

# Removing HTML garbage
text = re.sub(r"&\w+;", "",text)

# replace repeated letters with only two occurences
# heeeelllloooo => heelloo
text = re.sub(r"(.)\1+", r"\1\1",text)

#remove single letters
text = del_singles(text)

text = text.lower()

return text
```

Data Visualization : Heat Map

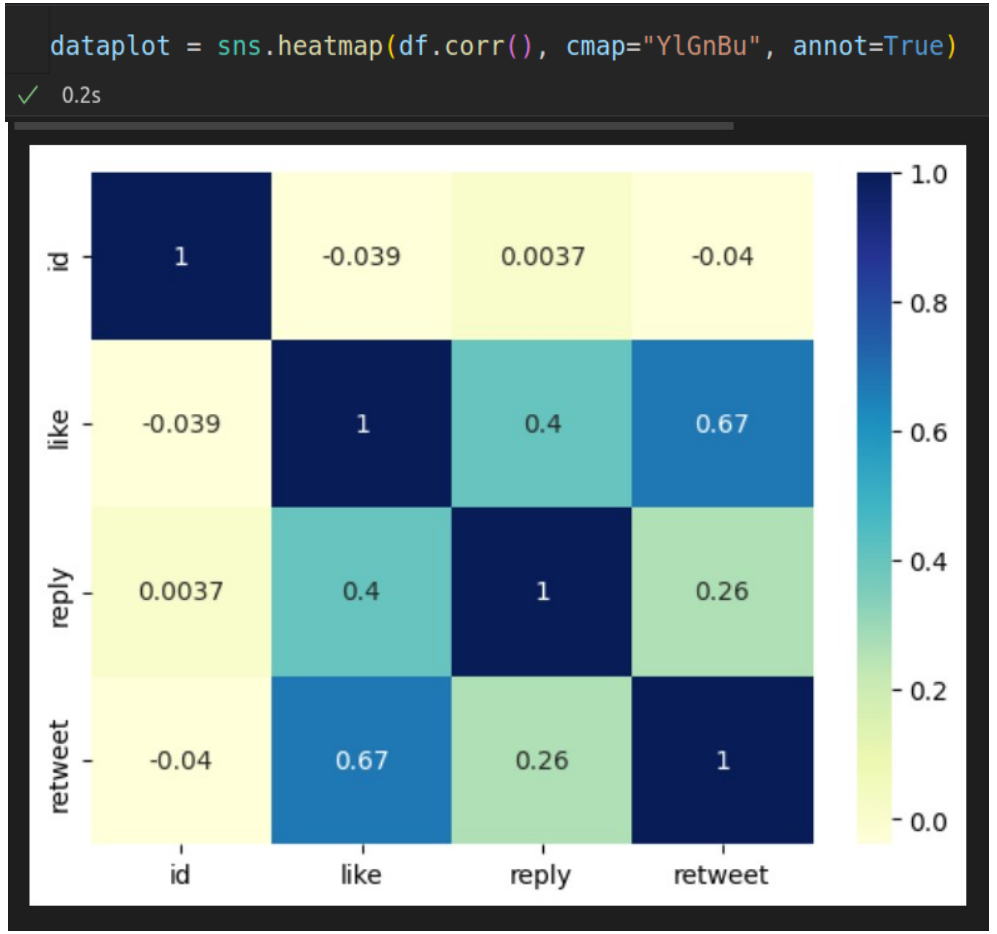
```
sns.heatmap(df.corr())
```

✓ 0.2s

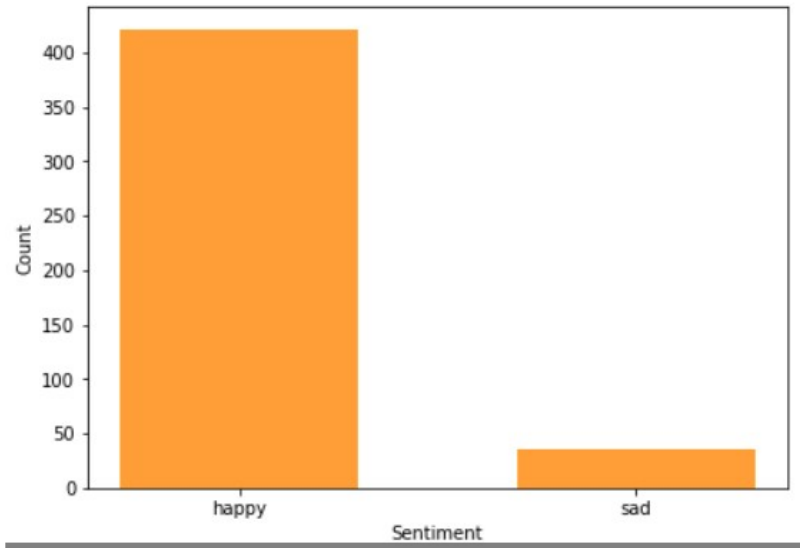



```
dataplot = sns.heatmap(df.corr(), cmap="YlGnBu", annot=True)
```

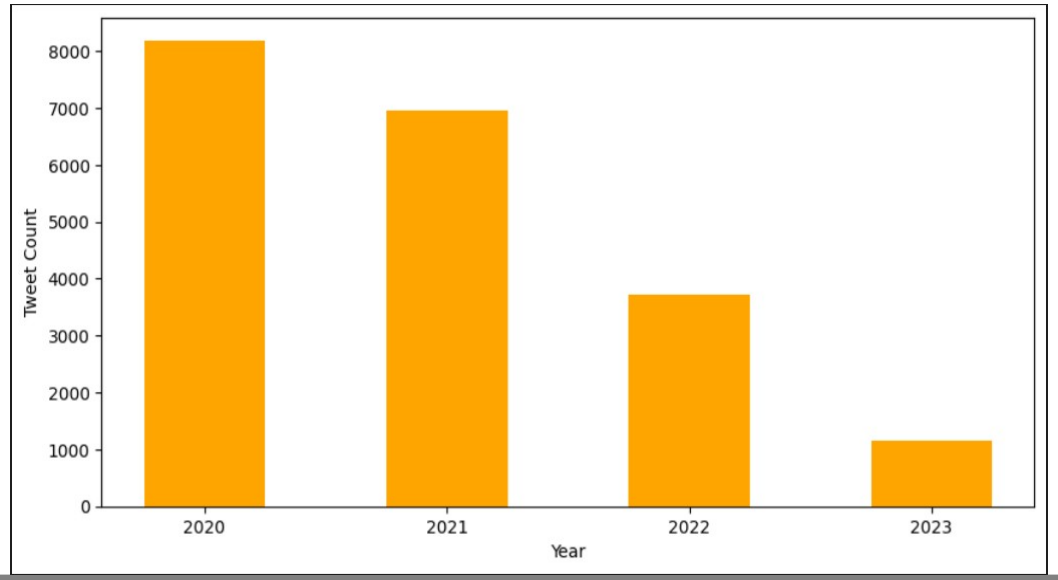
✓ 0.2s



Data Visualization



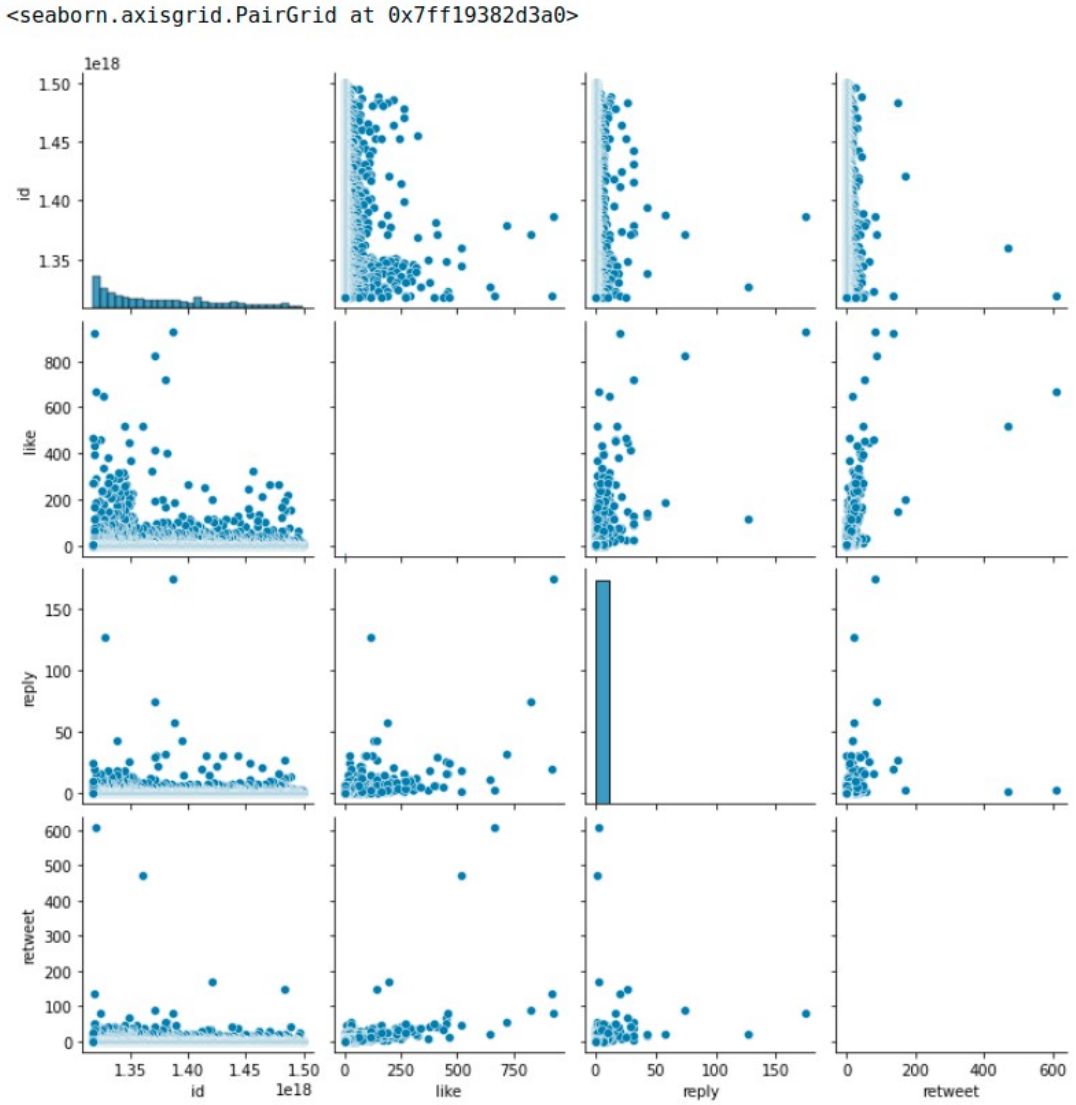
Happy v/s Sad



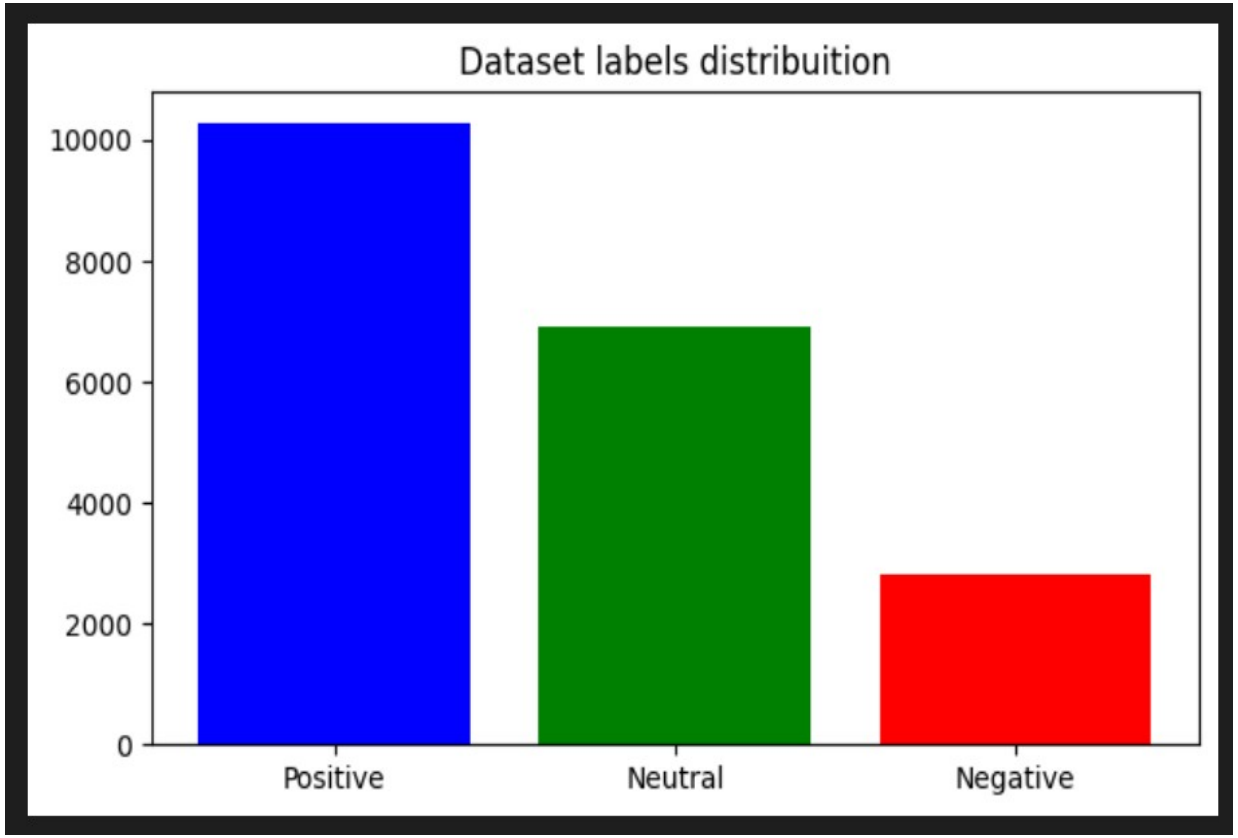
Number of tweets per year

Data Visualization

- Pair Plot between each Attribute of the data set
- Showing linear relationship Along with histogram of frequencies



Data Visualization




```
#plotting the bar plot
sentiment_cnt = Counter(df.sentiment)
plt.figure(figsize=(7,4))
plt.bar(sentiment_cnt.keys(),
        sentiment_cnt.values(),
        color=('b','g','r'))
plt.title("Dataset labels distribuition")
```

Model Building

```
x = tweet
y = df['sentiment'].values

# spliting the data for training and validation
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 1)

# print the length of train and test(validation) data
print("Train data: ", len(x_train))
print("Test data: ", len(x_test))
```

✓ 0.0s

```
Train data:  13995
Test data:  5998
```

SVM (Support Vector Machine Model)

Training

```
# Building a Support Vector Machine Model using TF-IDF approach
model_SVM = svm.SVC(C=0.98,kernel='linear')
model_SVM.fit(train_vectors, y_train)
```

✓ 24.8s

Testing

```
y_pred = model_SVM.predict(test_vectors)
cm = confusion_matrix(y_test,y_pred)
report = classification_report(y_test,y_pred)

print("\nEVALUTION MATRIX\n\n",cm)
print("\n\n\nCLASSIFICATION MATRIX\n\n ",report)
```

✓ 6.7s

```
EVALUTION MATRIX

 [[ 551  148  126]
 [  11 1933  144]
 [  76  153 2856]]



CLASSIFICATION MATRIX

              precision    recall  f1-score   support

    Negative       0.86      0.67      0.75       825
     Neutral       0.87      0.93      0.89      2088
    Positive       0.91      0.93      0.92      3085

    accuracy                           0.89      5998
   macro avg       0.88      0.84      0.86      5998
weighted avg       0.89      0.89      0.89      5998
```

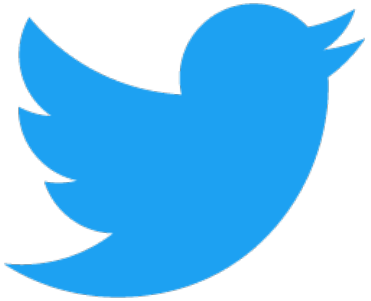
Accuracy

```
score_SVM = accuracy_score(y_test,y_pred)
print("Accuracy: ", score_SVM*100, " %")
```

✓ 0.0s

```
Accuracy:  89.02967655885296  %
```

Naive Bayes



Training

```
model_NB = MultinomialNB()
model_NB.fit(train_vectors, y_train)
```

✓ 0.0s

Accuracy

```
score_NB = accuracy_score(y_test,y_pred)
print("Accuracy: ", score_NB*100, " %")
```

✓ 0.0s

```
Accuracy:  66.50550183394465  %
```

Testing

```
y_pred = model_NB.predict(test_vectors)
cm = confusion_matrix(y_test,y_pred)
report = classification_report(y_test,y_pred)

print("\nEVALUTION MATRIX\n\n",cm)
print("\n\n\nCLASSIFICATION MATRIX\n\n ",report)
```

✓ 0.1s

```
EVALUTION MATRIX

 [[ 244   29  552]
 [   3  715 1370]
 [   2   53 3030]]



CLASSIFICATION MATRIX

              precision    recall  f1-score   support

    Negative       0.98      0.30      0.45       825
     Neutral       0.90      0.34      0.50      2088
    Positive       0.61      0.98      0.75      3085

    accuracy                           0.67      5998
   macro avg       0.83      0.54      0.57      5998
weighted avg       0.76      0.67      0.62      5998
```

Logistic Regression

Training

```
model_LR = sklearn.linear_model.LogisticRegression(penalty = "l1", C=0.1, solver="liblinear")
model_LR.fit(train_vectors, y_train)
```

✓ 0.1s

Accuracy

```
score_LR = accuracy_score(y_test,y_pred)
print("Accuracy: ", score_LR*100, " %")
```

✓ 0.0s

```
Accuracy:  74.5248416138713  %
```

Testing

```
y_pred = model_LR.predict(test_vectors)
cm = confusion_matrix(y_test,y_pred)
report = classification_report(y_test,y_pred)

print("\nEVALUTION MATRIX\n\n",cm)
print("\n\n\nCLASSIFICATION MATRIX\n\n ",report)
```

✓ 0.1s

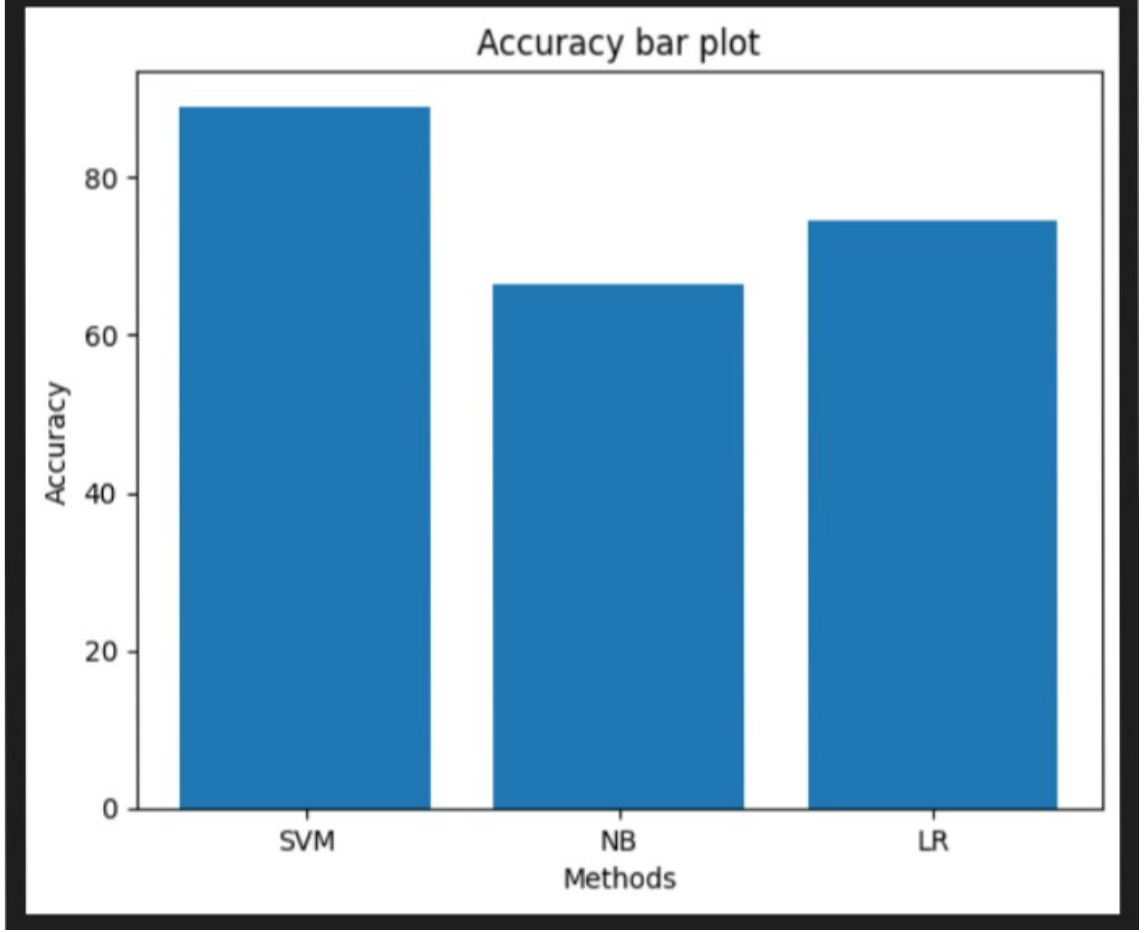
```
EVALUTION MATRIX

 [[ 326  263  236]
 [   5 1697  386]
 [  36  602 2447]]



CLASSIFICATION MATRIX

               precision    recall  f1-score   support

    Negative       0.89      0.40      0.55       825
     Neutral       0.66      0.81      0.73      2088
    Positive       0.80      0.79      0.80      3085

    accuracy                           0.75      5998
   macro avg       0.78      0.67      0.69      5998
weighted avg       0.76      0.75      0.74      5998
```

Conclusion

- From the Table, it is clear that Support Vector Machine (SVM) model accuracy is 89.02% which is greater than accuracy of Naive Bayes and Logistic Regression model .
- We therefore conclude that, SVM is the best model for identifying the sentiments of tweets .

```
x = ['SVM', 'NB', 'LR']
y = [score_SVM*100, score_NB*100, score_LR*100]

plt.bar(x,y)
plt.xlabel('Methods')
plt.ylabel('Accuracy')
plt.title('Accuracy bar plot')
plt.show()
```

✓ 0.1s



Thank You!