

Twitter Sentiment Analysis



July 1, 2019

Made By- Parth Goel

Path Infotech Limited



Acknowledgement

It gives me immense pleasure to express my sincere and deep feelings of gratitude to my mentor Mr. Praphul Kumar, who has given me his invaluable moral support from time to time and helped & co-operated with me in preparation of this Project.

I am also grateful to My Parents and My Siblings for their constant help and cooperation.

Contents

[II. Introduction 2](#_Toc14993271)

[III. Purpose 3](#_Toc14993272)

[IV. Scope 3](#_Toc14993273)

[V. Tools required 3](#_Toc14993274)

[VI. Applications of Sentiment Analysis 3](#_Toc14993275)

[VII. Working 4](#_Toc14993276)

[A. Libraries Used 4](#_Toc14993277)

[B. Code Explanation 5](#_Toc14993278)

[VIII. Report of Twitter Sentiment and Data Analysis of Reliance Jio and Airtel 17](#_Toc14993279)

[A. Tweet Source and Tweet Sentiment Analysis 18](#_Toc14993280)

[B. Analysis based on Likes, Retweets and Sentiment of Tweets 19](#_Toc14993281)

[C. Sentiment Distribution Analysis 21](#_Toc14993282)

[D. Analysis based on Sentiment by Hour of Day 22](#_Toc14993283)

[E. Some more informative graphs 23](#_Toc14993284)

[F. Conclusion 24](#_Toc14993285)

[IX. Bibliography 25](#_Toc14993286)

# Introduction

**What is sentiment analysis?**Sentiment Analysis is the process of ‘computationally’ determining whether a piece of writing is positive, negative or neutral. It’s also known as **opinion mining**, deriving the opinion or attitude of a speaker.

**Why sentiment analysis?**

* **Business:** In marketing field companies use it to develop their strategies, to understand customers’ feelings towards products or brand, how people respond to their campaigns or product launches and why consumers don’t buy some  
  products.
* **Politics:** In political field, it is used to keep track of political view, to detect consistency and inconsistency between statements and actions at the government level. It can be used to predict election results as well!
* **Public Actions:** Sentiment analysis also is used to monitor and analyze social phenomena, for the spotting of potentially dangerous situations and determining the general mood of the blogosphere.

# Purpose

Purpose of this document is to explain the working of Twitter Sentiment Analysis Project.

This project performs the sentiment analysis of any two topics by parsing the tweets fetched from Twitter using Python and comparing is done on the retrieved data through graphs, databases and data frames.

# Scope

Sentiment analysis can be applied at different levels of scope:

* **Document level** sentiment analysis obtains the sentiment of a complete document or paragraph.
* **Sentence level** sentiment analysis obtains the sentiment of a single sentence.
* **Sub-sentence level** sentiment analysis obtains the sentiment of sub-expressions within a sentence.

\* We’ll be doing sentence level sentiment analysis in our project.

# Tools required

* A python IDE (here PyCharm is used).
* A database managing tool (here MS Excel is used).
* Operating system used here is Windows 10.
* Twitter developer account and twitter credentials.
* Active internet connection.

# Applications of Sentiment Analysis

* [Social Media Monitoring](https://monkeylearn.com/sentiment-analysis/#sentiment-analysis-in-social-media-monitoring)
* [Brand Monitoring](https://monkeylearn.com/sentiment-analysis/#sentiment-analysis-in-brand-monitoring)
* [Customer Feedback](https://monkeylearn.com/sentiment-analysis/#sentiment-analysis-in-customer-feedback)
* [Customer Support](https://monkeylearn.com/sentiment-analysis/#sentiment-analysis-in-customer-support)
* [Workforce Analytics & Voice of the Employee](https://monkeylearn.com/sentiment-analysis/#sentiment-analysis-in-workforce-analytics-voice-of-the-employee)
* [Product Analytics](https://monkeylearn.com/sentiment-analysis/#sentiment-analysis-in-product-analytics)
* [Market Research and Analysis](https://monkeylearn.com/sentiment-analysis/#sentiment-analysis-in-market-research)

# Working

## Libraries Used

from tweepy import API  
from tweepy import OAuthHandler  
from textblob import TextBlob  
import twitter\_credentials  
import re  
import os  
import itertools  
import seaborn as sns  
import numpy as np  
import pandas as pd  
import matplotlib

* **Tweepy -** Tweepy is an open-sourced library, hosted on [GitHub](https://github.com/tweepy/tweepy) and enables Python to communicate with Twitter platform and use its API.
* **OAuthHandler -** Tweepy supports accessing Twitter via Basic Authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication so OAuth is now the only way to use the Twitter API.
* **TextBlob -** TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.
* **Re -** This module provides regular expression matching operations similar to those found in Perl. Both patterns and strings to be searched can be Unicode strings as well as 8-bit strings.
* **Os -** The main purpose of the OS module is to interact with your operating system. The primary use I find for it is to create folders, remove folders, move folders, and sometimes change the working directory. You can also access the names of files within a file path.
* **Itertools -** The Python itertools module is a collection of tools for handling iterators. Simply put, iterators are data types that can be used in a for loop.
* **Seaborn -** Seaborn is a Python data visualization library based on [matplotlib](https://matplotlib.org/). It provides a high-level interface for drawing attractive and informative statistical graphics.
* **Numpy - Numpy** is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.
* **Pandas -** Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.
* **Matplotlib -** Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

## Code Explanation

**1)**

matplotlib.rcParams.update(  
 {  
 'text.usetex': False,  
 'font.family': 'stixgeneral',  
 'mathtext.fontset': 'stix',  
 }  
)

This piece of code is used to set font in matplotlib graphs.

**2)**

pd.set\_option('display.max\_rows', 500)  
pd.set\_option('display.max\_columns', 500)  
pd.set\_option('display.width', 1000)

This code is used to set maximum number rows and columns of pandas dataframes and to set width of the dataframe.

**3)**

class TwitterClient:  
  
 def \_\_init\_\_(self, twitter\_user=None):  
 self.auth = TwitterAuthenticator().authenticate\_twitter\_app()  
 self.twitter\_client = API(self.auth)  
 self.twitter\_user = twitter\_user  
  
 def get\_twitter\_client\_api(self):  
 return self.twitter\_client  
  
  
class TwitterAuthenticator:  
  
 def authenticate\_twitter\_app(self):  
 auth = OAuthHandler(twitter\_credentials.CONSUMER\_KEY, twitter\_credentials.CONSUMER\_SECRET)  
 auth.set\_access\_token(twitter\_credentials.ACCESS\_TOKEN, twitter\_credentials.ACCESS\_TOKEN\_SECRET)  
 return auth

This whole code is used to get twitter authenticationto fetch tweets. We provide twitter credentials in this code that was provided us by twitter upon making a developer account. The credentials are saved in a different python file called twitter\_credentials.py.

**4)**

class TweetAnalyzer:  
  
 def clean\_tweet(self, tweet):  
 return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)", " ", tweet).split())  
  
 def analyze\_sentiment(self, tweet):  
 analysis = TextBlob(self.clean\_tweet(tweet))  
 if analysis.sentiment.polarity > 0:  
 return 1  
 elif analysis.sentiment.polarity == 0:  
 return 0  
 else:  
 return -1

This code has a class TweetAnalyzer which has two functions, first is clean\_tweet which is using re library of python and is used to remove all special characters like #, @ etc. and converts whole text to lower case. Second function is analyze\_sentiment, and it is used to judge the sentiment of the cleaned tweet by judging the polarity of the text. This is achieved through textblob library.

**5)**

def tweets\_to\_data\_frame(self, tweets):  
 rdf = pd.DataFrame(data=[tweet.text for tweet in tweets], columns=['Tweets'])  
 rdf['ID'] = np.array([tweet.id for tweet in tweets])  
 rdf['Length'] = np.array([len(tweet.text) for tweet in tweets])  
 rdf['Date'] = np.array([tweet.created\_at for tweet in tweets])  
 rdf['Source'] = np.array([tweet.source for tweet in tweets])  
 rdf['Likes'] = np.array([tweet.favorite\_count for tweet in tweets])  
 rdf['Retweets'] = np.array([tweet.retweet\_count for tweet in tweets])  
 rdf['User'] = np.array([tweet.user.screen\_name for tweet in tweets])  
 return rdf  
  
def tweets\_to\_data\_frame2(self, tweets2):  
 adf = pd.DataFrame(data=[tweet.text for tweet in tweets2], columns=['Tweets'])  
 adf['ID'] = np.array([tweet.id for tweet in tweets2])  
 adf['Length'] = np.array([len(tweet.text) for tweet in tweets2])  
 adf['Date'] = np.array([tweet.created\_at for tweet in tweets2])  
 adf['Source'] = np.array([tweet.source for tweet in tweets2])  
 adf['Likes'] = np.array([tweet.favorite\_count for tweet in tweets2])  
 adf['Retweets'] = np.array([tweet.retweet\_count for tweet in tweets2])  
 adf['User'] = np.array([tweet.user.screen\_name for tweet in tweets2])  
 return adf

Functions in this code are also under class TweetAnalyzer. Both functions are used to fetch certain values of tweets and put them in a dataframe. The values are tweet ID, length of tweet, date, source, likes, retweets and user. First function is creating Reliance Jio’s dataframe and second function is creating Airtel’s dataframe.

**6)**

if \_\_name\_\_ == '\_\_main\_\_':  
 twitter\_client = TwitterClient()  
 tweet\_analyzer = TweetAnalyzer()  
 api = twitter\_client.get\_twitter\_client\_api()  
  
 tweets = api.search(q="@Reliancejio", count=100, lang="en")  
 tweets2 = api.search(q="@airtelindia", count=100, lang="en")  
  
 rdf = tweet\_analyzer.tweets\_to\_data\_frame(tweets)  
 adf = tweet\_analyzer.tweets\_to\_data\_frame2(tweets2)  
  
 rdf['Sentiment'] = np.array([tweet\_analyzer.analyze\_sentiment(tweet) for tweet in rdf['Tweets']])  
 adf['Sentiment'] = np.array([tweet\_analyzer.analyze\_sentiment(tweet) for tweet in adf['Tweets']])

This code is to use twitter API called search to fetch tweets on provided query. This code is also creating dataframes and judging sentiment and creating and sentiment column in dataframes by using previously defined functions.

**7)**

date\_format = "%Y-%m-%dT%H:%M:%S"  
rdf["Date"] = pd.to\_datetime(rdf["Date"], format=date\_format)  
adf["Date"] = pd.to\_datetime(adf["Date"], format=date\_format)  
rdf["Hour"] = pd.DatetimeIndex(rdf["Date"]).hour  
adf["Hour"] = pd.DatetimeIndex(adf["Date"]).hour  
adf["Month"] = pd.DatetimeIndex(adf["Date"]).month  
rdf["Month"] = pd.DatetimeIndex(rdf["Date"]).month  
rdf["Day"] = pd.DatetimeIndex(rdf["Date"]).day  
adf["Day"] = pd.DatetimeIndex(adf["Date"]).day  
rdf["Month\_f"] = rdf["Month"].map({1: "JAN", 2: "FEB", 3: "MAR", 4: "APR", 5: "MAY", 6: "JUN", 7: "JUL",  
 8: "AUG", 9: "SEP"})  
adf["Month\_f"] = adf["Month"].map({1: "JAN", 2: "FEB", 3: "MAR", 4: "APR", 5: "MAY", 6: "JUN", 7: "JUL",  
 8: "AUG", 9: "SEP"})

This code is extracting hour, month, day from the date of tweets and creating a column in both the dataframes.

**8)**

print("\nRELIANCE JIO TWEETS\n")  
print(rdf)  
print("\nAIRTEL INDIA TWEETS\n")  
print(adf)

This code is used to print the fetched tweets’ dataframes.

**9)**

exists = os.path.isfile(r'C:\Users\parth\Desktop\RJ\_DB.csv')  
exists2 = os.path.isfile(r'C:\Users\parth\Desktop\AI\_DB.csv')  
  
if exists:  
 rdf.to\_csv(r'C:\Users\parth\Desktop\RJ\_DB.csv', mode='a', header=False)  
else:  
 rdf.to\_csv(r'C:\Users\parth\Desktop\RJ\_DB.csv', mode='a')  
if exists2:  
 adf.to\_csv(r'C:\Users\parth\Desktop\AI\_DB.csv', mode='a', header=False)  
else:  
 adf.to\_csv(r'C:\Users\parth\Desktop\AI\_DB.csv', mode='a')

This code is first checking that the csv excel file is already present or not. If its present then append the tweets, and if not present then create the file and write the tweets to csv file.

**10)**

rdf2 = pd.read\_csv(r"C:\Users\parth\Desktop\RJ\_DB.csv")  
adf2 = pd.read\_csv(r"C:\Users\parth\Desktop\AI\_DB.csv")  
  
rdf2 = rdf2.drop\_duplicates(subset='Tweets')  
adf2 = adf2.drop\_duplicates(subset='Tweets')

rdf2.to\_csv(r'C:\Users\parth\Desktop\RJ\_DB.csv', index=False, mode='w')  
adf2.to\_csv(r'C:\Users\parth\Desktop\AI\_DB.csv', index=False, mode='w')

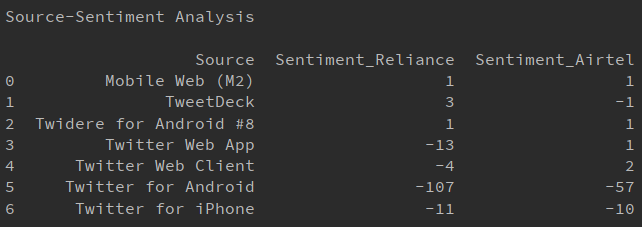
This code is reading tweets from csv file to dataframe and then dropping duplicate values to have unique tweets. Then it overwrites the csv file.

**11)**

rdf3 = rdf2.groupby('Source')['Sentiment'].sum().reset\_index()  
adf3 = adf2.groupby('Source')['Sentiment'].sum().reset\_index()  
ardf3 = pd.merge(rdf3, adf3, on="Source", suffixes=('\_Reliance', '\_Airtel'))

print("\nSource-Sentiment Analysis\n")  
print(ardf3)

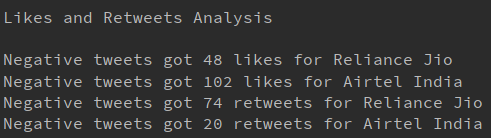
This code is grouping source column of both dataframes and giving us cumulative sentiment of both dataframes by tweet source. It also merges both dataframes to create a more readable data. For example -

****

**12)**

rdf4 = rdf2.loc[(rdf2['Likes'] >= 1) & (rdf2['Sentiment'] < 0)]  
adf4 = adf2.loc[(adf2['Likes'] >= 1) & (adf2['Sentiment'] < 0)]  
  
rdf5 = rdf2.loc[(rdf2['Retweets'] >= 1) & (rdf2['Sentiment'] < 0)]  
adf5 = adf2.loc[(rdf2['Retweets'] >= 1) & (adf2['Sentiment'] < 0)]  
  
print('\nLikes and Retweets Analysis')  
print('\nNegative tweets got ' + str(rdf4['Likes'].sum()) + ' likes for Reliance Jio')  
print('Negative tweets got ' + str(adf4['Likes'].sum()) + ' likes for Airtel India')  
  
print('Negative tweets got ' + str(rdf5['Retweets'].sum()) + ' retweets for Reliance Jio')  
print('Negative tweets got ' + str(adf5['Retweets'].sum()) + ' retweets for Airtel India')

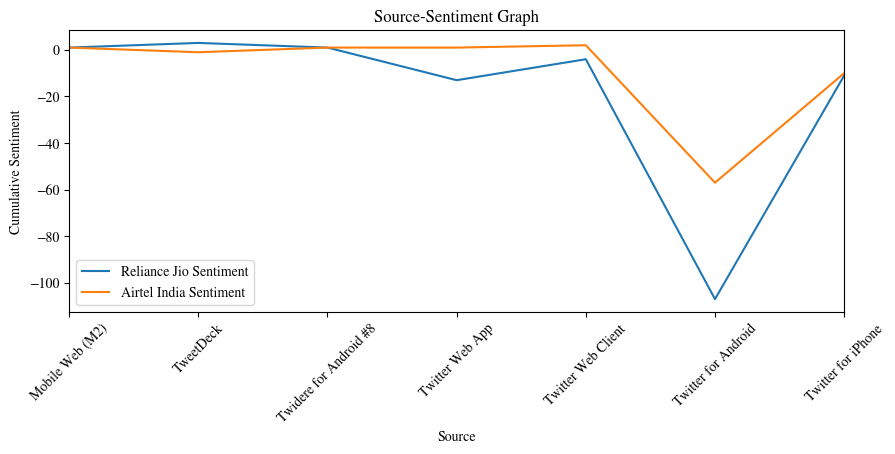
This code is used to do some data analysis on the tweets by judging how much likes and retweets did the negative tweets got for both dataframes. For example –



**13)**

# Source-Sentiment Graph  
 ax = ardf3.plot(x='Source', y='Sentiment\_Reliance')  
 ardf3.plot(ax=ax, x='Source', y='Sentiment\_Airtel', figsize=(10, 4))  
 ax.legend(["Reliance Jio Sentiment", "Airtel India Sentiment"])  
 ax.relim()  
 ax.autoscale\_view()  
 tick\_labels = tuple(ardf3['Source'])  
 x\_max = int(max(plt.xticks()[0]))  
 plt.xticks(range(0, x\_max + 1), tick\_labels, rotation=45)  
 plt.ylabel("Cumulative Sentiment")  
 plt.title("Source-Sentiment Graph")  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\1.png', bbox\_inches='tight')

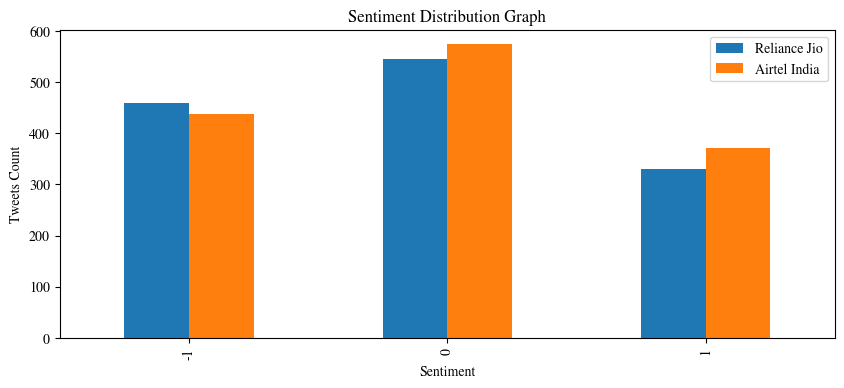
This code is used to generate Source-Sentiment graph using matplotlib and then saving the graph image to desired location. Graph generated –

****

**14)**

# Sentiment Distribution Graph  
 a1 = rdf2[rdf2['Sentiment'] == -1].shape[0]  
 a2 = adf2[adf2['Sentiment'] == -1].shape[0]  
 b1 = rdf2[rdf2['Sentiment'] == 0].shape[0]  
 b2 = adf2[adf2['Sentiment'] == 0].shape[0]  
 c1 = rdf2[rdf2['Sentiment'] == 1].shape[0]  
 c2 = adf2[adf2['Sentiment'] == 1].shape[0]  
 data = {'Reliance Jio': {'-1': a1, '0': b1, '1': c1}, 'Airtel India': {'-1': a2, '0': b2, '1': c2}}  
 df = pd.DataFrame(data)  
 df.plot(kind='bar', figsize=(10, 4))  
 plt.xlabel("Sentiment")  
 plt.ylabel("Tweets Count")  
 plt.title("Sentiment Distribution Graph")  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\2.png', bbox\_inches='tight')

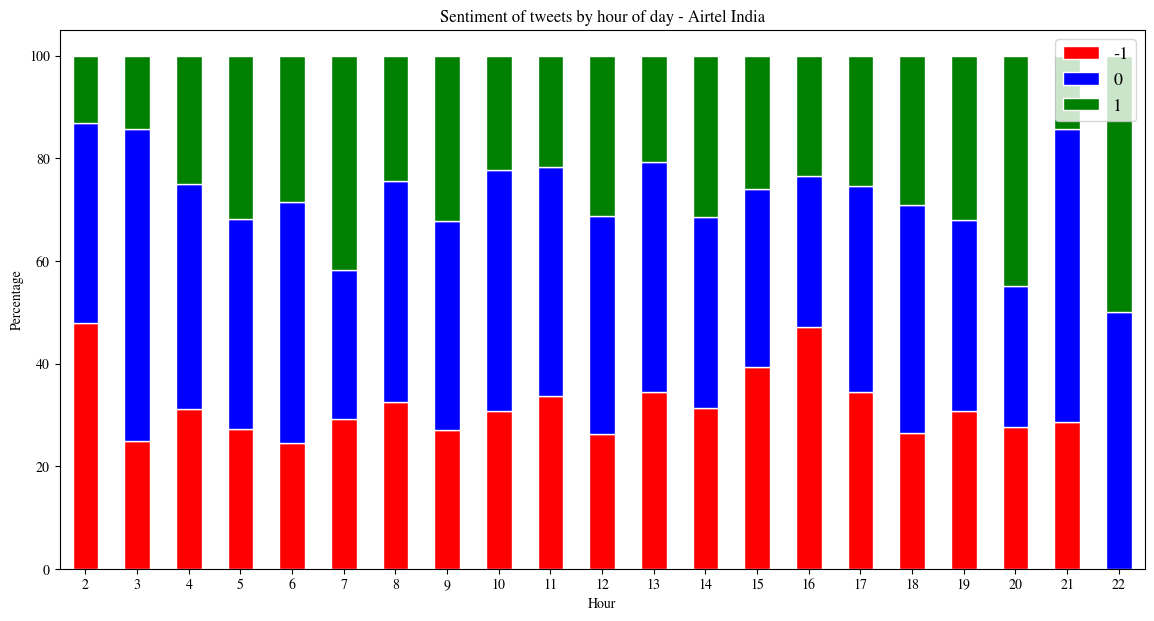
This code is used to generate Sentiment Distribution graph using matplotlib and then saving the graph image to desired location. Graph generated –

****

**15)**

# Sentiment of tweets by hour of day  
 st\_hr\_r = pd.crosstab(rdf2["Hour"], rdf2["Sentiment"])  
 st\_hr\_r = st\_hr\_r.apply(lambda r: r/r.sum()\*100, axis=1)  
  
 st\_hr\_a = pd.crosstab(adf2["Hour"], adf2["Sentiment"])  
 st\_hr\_a = st\_hr\_a.apply(lambda r: r/r.sum()\*100, axis=1)  
  
 st\_hr\_r.plot(kind="bar", figsize=(14, 7), color=["r", "b", "g"], linewidth=1, edgecolor="w", stacked=True)  
 plt.legend(loc="best", prop={"size": 13})  
 plt.title("Sentiment of tweets by hour of day - Reliance Jio")  
 plt.xticks(rotation=0)  
 plt.ylabel("Percentage")  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\3a.png', bbox\_inches='tight')  
  
 st\_hr\_a.plot(kind="bar", figsize=(14, 7), color=["r", "b", "g"], linewidth=1, edgecolor="w", stacked=True)  
 plt.legend(loc="best", prop={"size": 13})  
 plt.title("Sentiment of tweets by hour of day - Airtel India")  
 plt.xticks(rotation=0)  
 plt.ylabel("Percentage")  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\3b.png', bbox\_inches='tight')

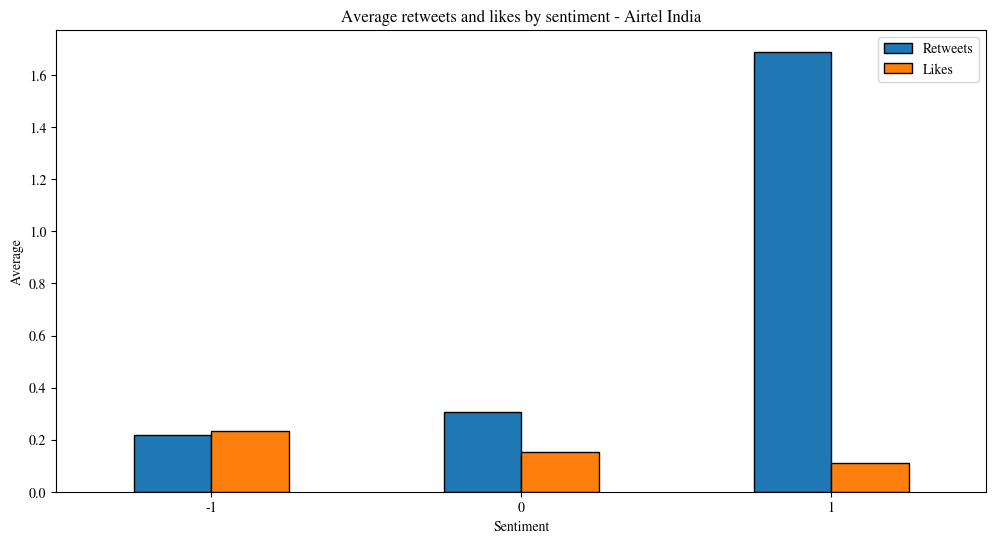
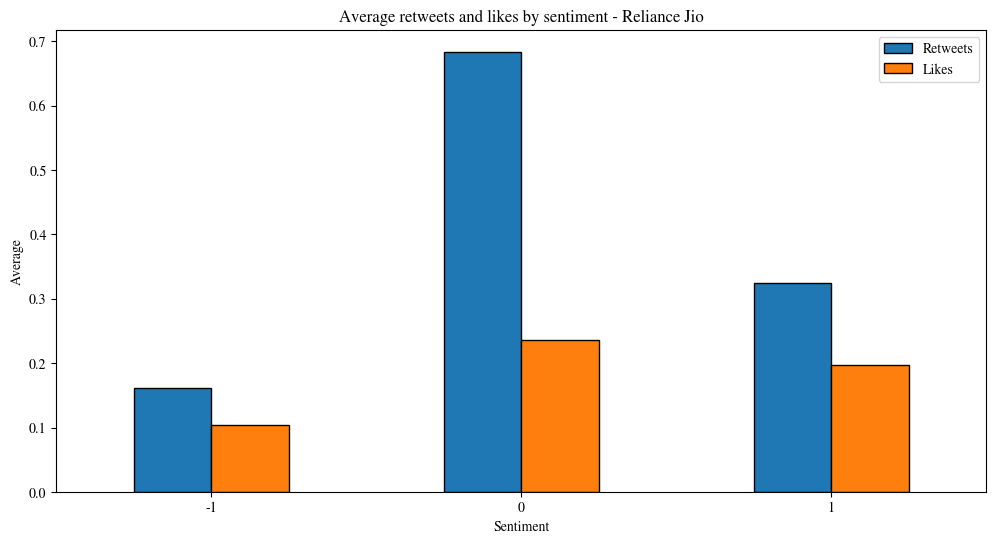
This code is used to generate Sentiment of tweets by hour of day graphs for both dataframes separately using matplotlib and then saving the graph images to desired location. Graphs generated – ****

****

**16)**

# Average retweets and likes by sentiment  
 avg\_lk\_rts\_r = rdf2.groupby("Sentiment")[["Retweets", "Likes"]].mean()  
 avg\_lk\_rts\_a = adf2.groupby("Sentiment")[["Retweets", "Likes"]].mean()  
  
 avg\_lk\_rts\_r.plot(kind="bar", figsize=(12, 6), linewidth=1, edgecolor="k")  
 plt.xticks(rotation=0)  
 plt.ylabel("Average")  
 plt.title("Average retweets and likes by sentiment - Reliance Jio")  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\4a.png', bbox\_inches='tight')  
  
 avg\_lk\_rts\_a.plot(kind="bar", figsize=(12, 6), linewidth=1, edgecolor="k")  
 plt.xticks(rotation=0)  
 plt.ylabel("Average")  
 plt.title("Average retweets and likes by sentiment - Airtel India")  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\4b.png', bbox\_inches='tight')

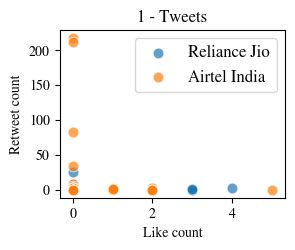
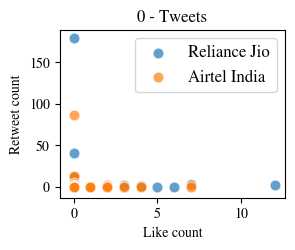
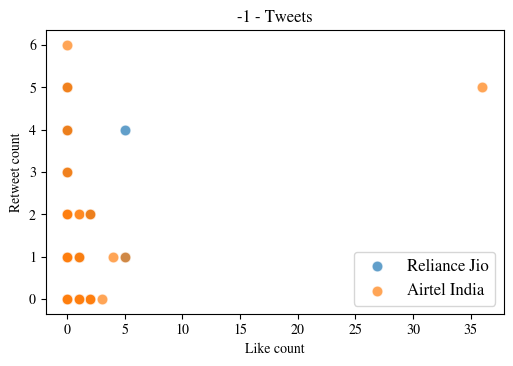
This code is used to generate Average retweet and likes by sentiment graphs for both dataframes separately using matplotlib and then saving the graph images to desired location. Graphs generated –



**17)**

# Likes and retweets by sentiment  
  
 lst = [-1, 0, 1]  
 cs = ["r", "g", "b"]  
  
 plt.figure(figsize=(13, 13))  
  
 for i, j, k in itertools.zip\_longest(lst, range(len(lst)), cs):  
 plt.subplot(2, 2, j+1)  
 plt.scatter(x=rdf2[rdf2["Sentiment"] == i]["Likes"], y=rdf2[rdf2["Sentiment"] == i]["Retweets"],  
 label="Reliance Jio", linewidth=.7, edgecolor="w", s=60, alpha=.7)  
 plt.scatter(x=adf2[adf2["Sentiment"] == i]["Likes"], y=adf2[adf2["Sentiment"] == i]["Retweets"],  
 label="Airtel India", linewidth=.7, edgecolor="w", s=60, alpha=.7)  
 plt.title(str(i) + " - Tweets")  
 plt.legend(loc="best", prop={"size": 12})  
 plt.xlabel("Like count")  
 plt.ylabel("Retweet count")  
  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\5\_{0}.png'.format(i), bbox\_inches='tight')

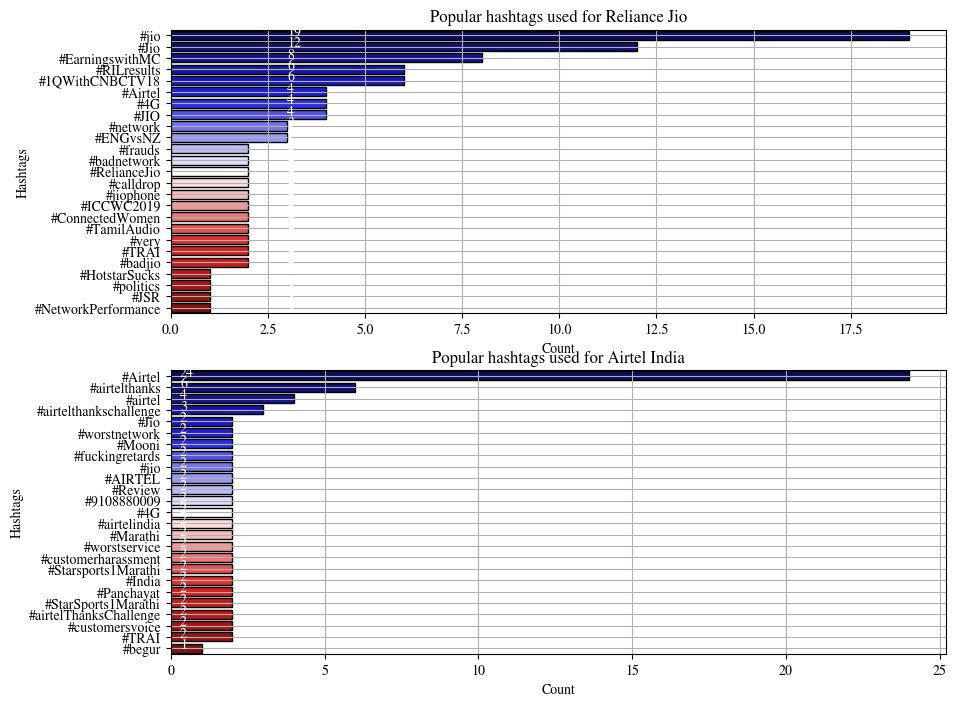
This code is used to generate likes and retweets by sentiment graphs for both dataframes separately using matplotlib and itertools and then saving the graph images to desired location. Graphs generated –

****

**18)**

# Popular hashtags  
 hashs\_r = rdf2["Tweets"].str.extractall(r'(\#\w+)')[0].value\_counts().reset\_index()  
 hashs\_r.columns = ["Hashtags", "Count"]  
 hashs\_a = adf2["Tweets"].str.extractall(r'(\#\w+)')[0].value\_counts().reset\_index()  
 hashs\_a.columns = ["Hashtags", "Count"]  
 plt.figure(figsize=(10, 20))  
 plt.subplot(211)  
 ax = sns.barplot(x="Count", y="Hashtags", data=hashs\_r[:25], palette="seismic", linewidth=1, edgecolor="k" \* 25)  
 plt.grid(True)  
 for i, j in enumerate(hashs\_r["Count"][:25].values):  
 ax.text(3, i, j, fontsize=10, color="white")  
 plt.title("Popular hashtags used for Reliance Jio")  
 plt.subplot(212)  
 ax1 = sns.barplot(x="Count", y="Hashtags", data=hashs\_a[:25], palette="seismic", linewidth=1, edgecolor="k" \* 25)  
 plt.grid(True)  
 for i, j in enumerate(hashs\_a["Count"][:25].values):  
 ax1.text(.3, i, j, fontsize=10, color="white")  
 plt.title("Popular hashtags used for Airtel India")  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\6.png', bbox\_inches='tight')

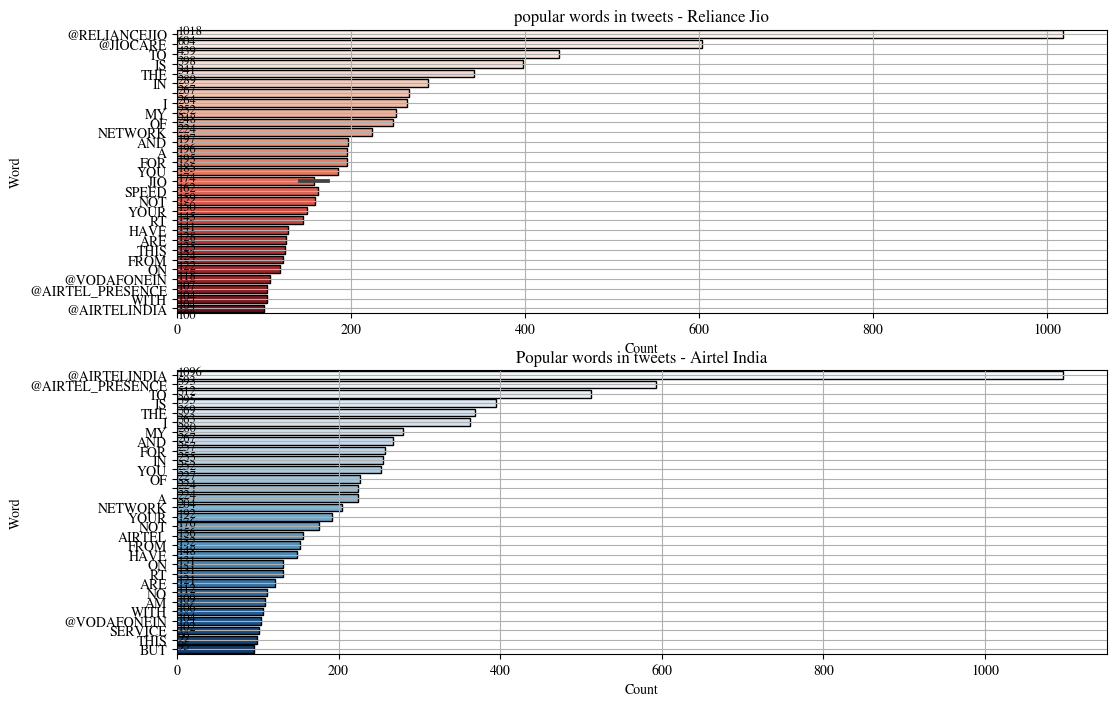
This code is used to generate popular hashtags graphs for both dataframes separately using matplotlib and seaborn and then saving the graph images to desired location. Graphs generated –

****

**19)**

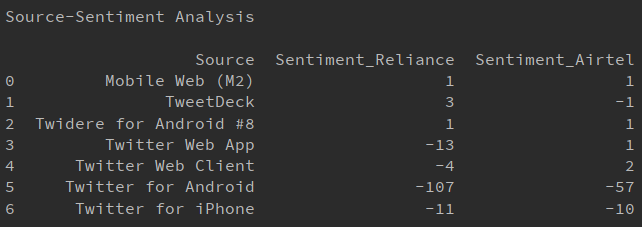
# Popular words in tweets  
 pop\_words\_r = (rdf2["Tweets"].apply(lambda x: pd.value\_counts(x.split(" "))).sum(axis=0).reset\_index().sort\_values(by=[0], ascending=False))  
 pop\_words\_r.columns = ["Word", "Count"]  
 pop\_words\_r["Word"] = pop\_words\_r["Word"].str.upper()  
 pop\_words\_a = (adf2["Tweets"].apply(lambda x: pd.value\_counts(x.split(" "))).sum(axis=0).reset\_index().sort\_values(by=[0], ascending=False))  
 pop\_words\_a.columns = ["Word", "Count"]  
 pop\_words\_a["Word"] = pop\_words\_a["Word"].str.upper()  
 plt.figure(figsize=(12, 25))  
 plt.subplot(211)  
 ax = sns.barplot(x="Count", y="Word", data=pop\_words\_r[:30], linewidth=1, edgecolor="k" \* 30, palette="Reds")  
 plt.title("popular words in tweets - Reliance Jio")  
 plt.grid(True)  
 for i, j in enumerate(pop\_words\_r["Count"][:30].astype(int)):  
 ax.text(.8, i, j, fontsize=9)  
 plt.subplot(212)  
 ax1 = sns.barplot(x="Count", y="Word", data=pop\_words\_a[:30], linewidth=1, edgecolor="k" \* 30, palette="Blues")  
 plt.title("Popular words in tweets - Airtel India")  
 plt.grid(True)  
 for i, j in enumerate(pop\_words\_a["Count"][:30].astype(int)):  
 ax1.text(.8, i, j, fontsize=9)  
 fig = plt.gcf()  
 plt.show()  
 fig.savefig(r'C:\Users\parth\Desktop\TSA2\7.png', bbox\_inches='tight')

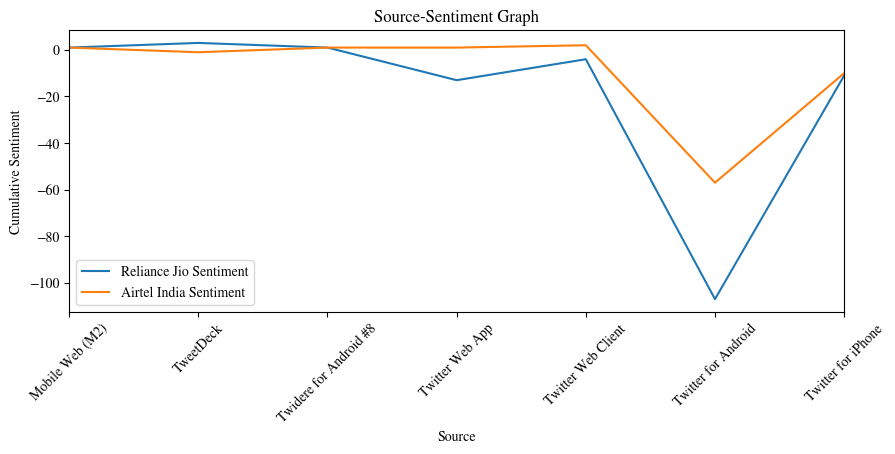
This code is used to generate popular words in tweets graphs for both dataframes separately using matplotlib and seaborn and then saving the graph images to desired location. Graphs generated –

****

# Report of Twitter Sentiment and Data Analysis of Reliance Jio and Airtel

## Tweet Source and Tweet Sentiment Analysis

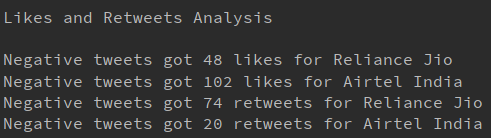
****

****

* In above sentiment analysis, the major sources are Twitter Web App, Twitter Web Client, Twitter for Android and Twitter for iPhone.
* Reliance Jio has more negative sentiment in all the sources mentioned above.
* We can see this in the graph also, the blue line of Reliance Jio is above orange line of Airtel for only one source i.e. TweetDeck. Rest of the time it’s below Airtel.
* Hence, we can clearly state that **Airtel wins this segment.**

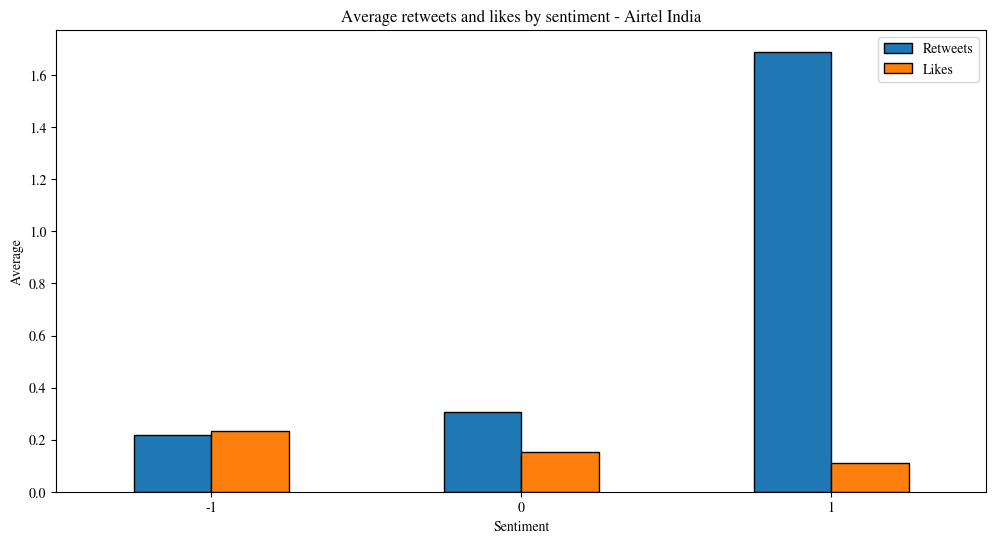
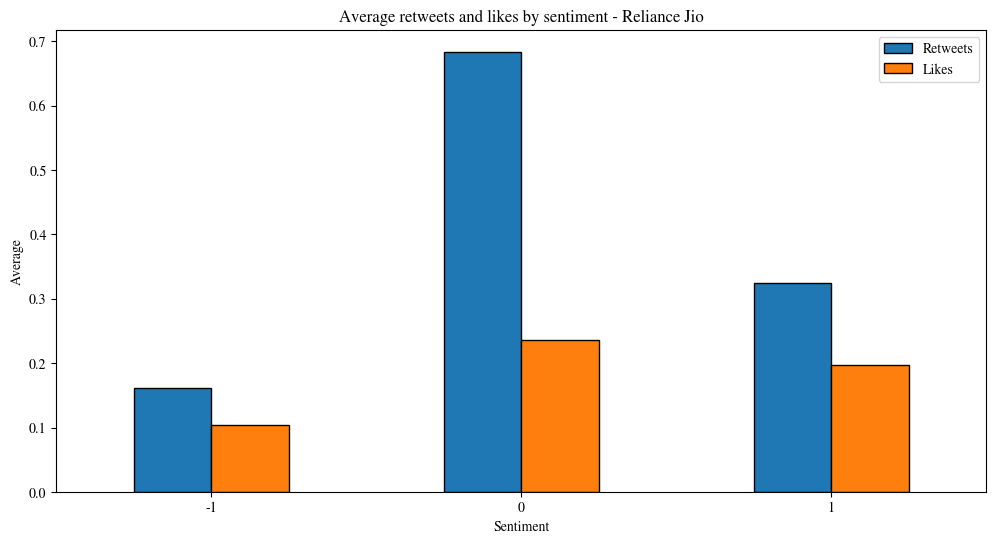
## Analysis based on Likes, Retweets and Sentiment of Tweets

**(1)**



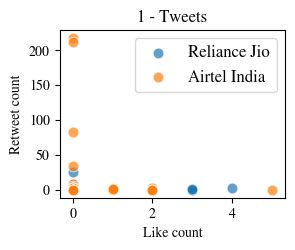
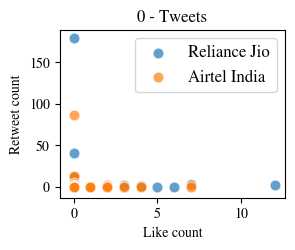
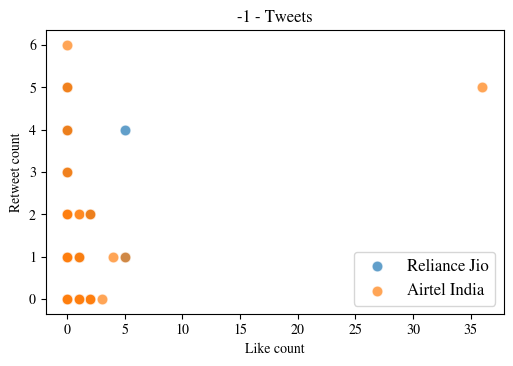
* Here we can see that how many likes and retweets did the negative tweets got for both companies.
* The lower likes and retweets the negative tweets get the better it is.
* Reliance wins in likes count because it’s lower than Airtel’s.
* But Airtel takes the lead in retweets count as they are lower than that of Reliance’s.
* As we can’t judge who won in this segment we must explore this segment further.

**(2)**



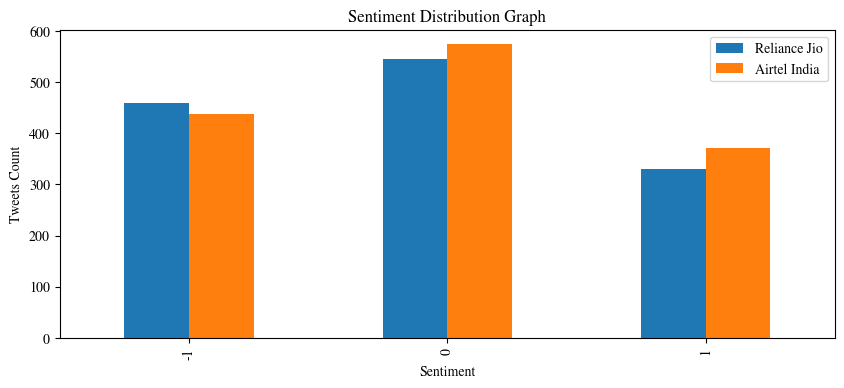
* The above graphs plot average likes and retweets count against sentiment of tweets for both companies in separate graphs.
* Negative Tweets (Denoted by ‘-1’ on x-axis) got slightly more like and retweet count average for Airtel (approx. 0.2) than that of Reliance’s (both below 0.2). So, we can say that **Reliance Jio wins here.**
* Neutral Tweets (Denoted by ‘0’ on x-axis) have higher average of likes and retweets for Reliance Jio. So, we can say that people are more likely to retweet and like tweets of Reliance Jio than Airtel. **So, Reliance Jio wins here too.**
* Positive Tweets (Denoted by ‘1’ on x-axis) have a really high average of retweet count and slightly low average of like count for Airtel. **Airtel takes the win here.**

\* We can show the correlation between both companies based on the like count, retweet count and sentiment through scatter graphs.

**(3)**

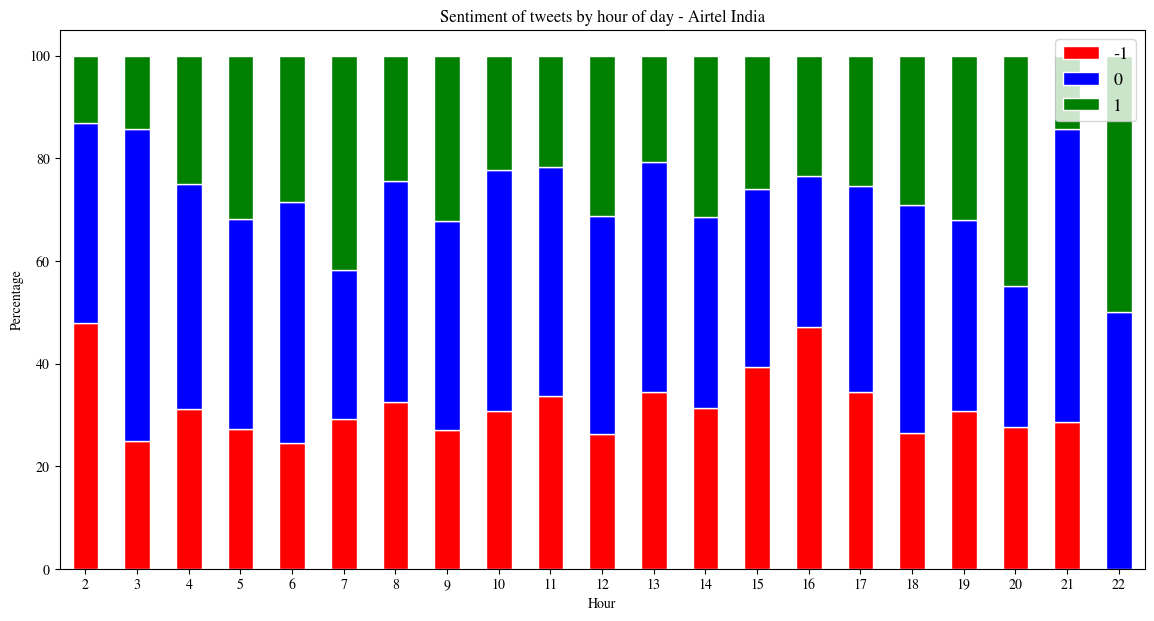
By the previous discussion in (b) part of this segment and these scatter graphs we can say that **Reliance Jio performed better in this segment.**

## Sentiment Distribution Analysis

****

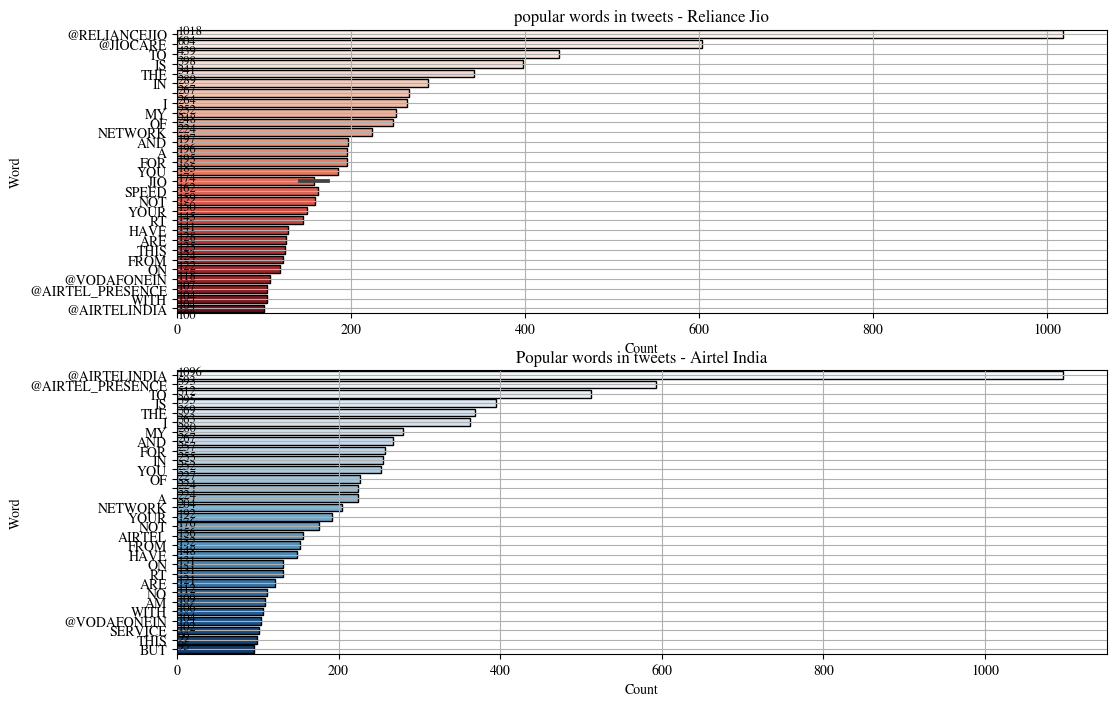
* This graph compares both companies neck-to-neck based on sentiment and tweet count.
* Reliance Jio has a slightly higher number of negative tweets than Airtel so, **Airtel wins here.**
* Airtel has a higher number of neutral tweets, hence we can say that people are more likely to tweet for Airtel than Reliance Jio. **Airtel wins here too.**
* **Airtel wins** in number of positive tweets also.
* We can clearly state that **Airtel wins this segment.**

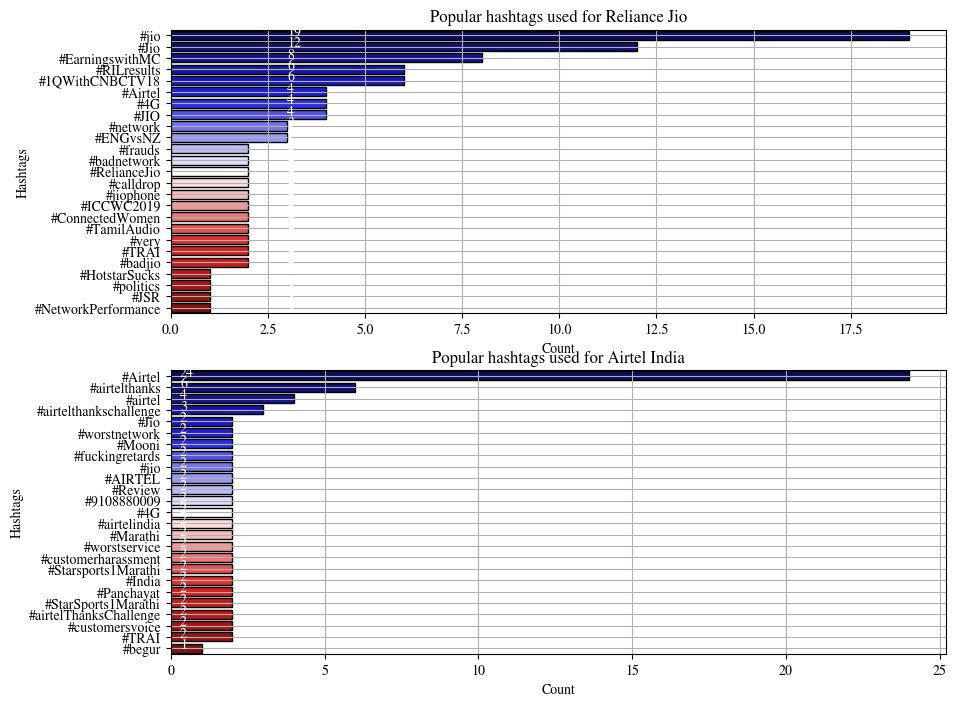
## Analysis based on Sentiment by Hour of Day



* These graphs show change in percentage of sentiment throughout the day.
* We can’t really judge based on these graphs but Airtel’s graph looks more stable than Reliance’s.

## Some more informative graphs

****

****

## Conclusion

Airtel performed better by getting less cumulative negative tweet sentiment throughout all twitter sources like android, iPhone etc. Reliance Jio failed here as all the sources had more cumulative negative sentiment. Jio performed better at only one place, likes and retweets segment where Jio barely took the lead over Airtel. Jio had less likes and retweets for negative tweets, more likes and retweets for neutral tweets but lost in number of likes and retweets for positive tweets where Airtel took the lead. Lastly, Airtel outperformed Jio in sentiment distribution segment. Airtel had a smaller number of negative tweets, a greater number of neutral and positive tweets. **Hence, it’s clear that Airtel won this competition by edging out Jio in more segments.**

# Bibliography

* geeksforgeeks.org
* stackoverflow.com
* google.com
* github.com
* medium.com
* matplotlib.org
* learnpython.org
* pythonprogramming.net
* pythoncentral.io
* seaborn.pydata.org
* kaggle.com