

# IST 652 FINAL PROJECT REPORT

-ROMIL GODHA

- SOUMYA SAMAL

# 1. OVERVIEW

**Zomato** is a restaurant search and discovery service founded in 2008. It currently operates in 23 countries, including Australia and United States. It provides information and reviews on restaurants, including images of menus where the restaurant does not have its own website. The primary purpose of the project was to perform statistical analysis and twitter sentimental analysis to visualize its presence over the world, types of services it provides, performance in the market as a web application. We also applied Machine learning algorithms to accurately predict the Dependent variable from Independent variables i.e. how for example rating (dependent variable) is provided based on price, services etc. (independent variable).

The source of the project was Kaggle and the dataset was downloaded from Kaggle. The contents of the dataset were csv files and xlsx files. Also for twitter sentimental analysis, Twitter API was leveraged to get insights for our analysis.

## 2. DATA

The data was pulled from <https://www.kaggle.com/shrutimehta/zomato-restaurants-data/data>. This data was collected from Zomato API and the source was provided as a CSV file. The data contained 9551 rows of information about different hotels around the world.

### 2.1. METADATA

Each hotel was described by 21 attributes:

- **Restaurant Id:** Unique id of every restaurant across various cities of the world
- **Restaurant Name:** Name of the restaurant
- **Country Code:** Country in which restaurant is located
- **City:** City in which restaurant is located
- **Address:** Address of the restaurant
- **Locality:** Location in the city
- **Locality Verbose:** Detailed description of the locality
- **Longitude:** Longitude coordinate of the restaurant's location
- **Latitude:** Latitude coordinate of the restaurant's location
- **Cuisines:** Cuisines offered by the restaurant
- **Average Cost for two:** Cost for two people in different currencies
- **Currency:** Currency of the country
- **Has Table booking:** yes/no

- **Has Online delivery:** yes/ no
- **Is delivering:** yes/ no
- **Switch to order menu:** yes/no
- **Price range:** range of price of food
- **Aggregate Rating:** Average rating out of 5
- **Rating color:** depending upon the average rating color
- **Rating text:** text on the basis of rating of rating
- **Votes:** Number of ratings casted by people

### 3. DATA PREPROCESSING

- Importing the dataset

```
# importing the data
proj= pd.read_csv("C:/Users/Romil Godha/Desktop/zomato/zomato.csv", encoding = "ISO-8859-1")
```

- Looking at the initial data

```
# printing the rows and columns number
print('Number of rows and columns:', proj.shape )
```

```
Number of rows and columns: (9551, 21)
```

```
# printing the name of columns
print('Column Labels', proj.columns.values.tolist())
```

```
Column Labels ['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address', 'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines', 'Average Cost for two', 'Currency', 'Has Table booking', 'Has Online delivery', 'Is delivering now', 'Switch to order menu', 'Price range', 'Aggregate rating', 'Rating color', 'Rating text', 'Votes']
```

```
# printing the data types of columns
print('Type of values', proj.dtypes)
```

Type of values	Restaurant ID	int64
Restaurant Name	object	
Country Code	int64	
City	object	
Address	object	
Locality	object	
Locality Verbose	object	
Longitude	float64	
Latitude	float64	
Cuisines	object	
Average Cost for two	int64	
Currency	object	
Has Table booking	object	
Has Online delivery	object	
Is delivering now	object	
Switch to order menu	object	
Price range	int64	
Aggregate rating	float64	
Rating color	object	
Rating text	object	
Votes	int64	
dtype: object		

- From the above information we chose only 14 essential attributes for our analysis.

```
# Taking only the necessary columns for analysis
proj_sub = pd.DataFrame(proj, columns=['Restaurant Name', 'City', 'Cuisines', 'Average Cost for two', 'Currency',
'Has Online delivery', 'Is delivering now', 'Has Table booking', 'Price range', 'Aggregate rating', 'Rating text',
'Votes', 'Longitude', 'Latitude'])
```

- Renaming columns for ease of use by removing spaces and replacing by “\_”

```
# Renaming columns
proj_sub.columns= ['Restaurant_Name', 'City', 'Cuisines', 'Average_Cost_for_two', 'Currency', 'Has_Online_delivery',
'Is_delivering_now', 'Has_Table_booking', 'Price_range', 'Aggregate_rating', 'Rating_text', 'Votes', 'Longitude', 'Latitude']
```

- Removing the aggregate values which have value 0

```
# Removing aggregate rating with value equal to 0
proj_sub =proj_sub.loc[proj_sub['Aggregate_rating']>0]
```

```
# printing the rows and columns number
print('Number of rows and columns:', proj_sub.shape )
```

```
Number of rows and columns: (7403, 14)
```

Merging two datasets out of which one is xlsx and the second is csv. This allowed us to add country column by merging the dataset using country code column.

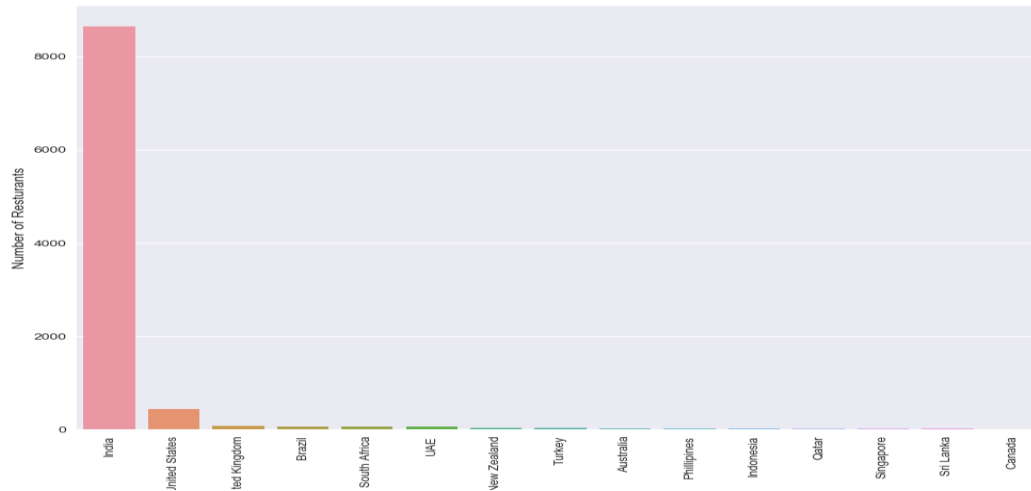
```
# Merging a dataset with country code file
Country = pd.read_excel('C:/Users/Romil Godha/Desktop/zomato/Country-Code.xlsx')
merger = pd.merge(proj, Country, on='Country Code')
```

```
In : merger.shape
(9551, 22)
```

Now checking with countries with the most number of restaurants. This was done by grouping the country code present in the country-code file.

```
#Grouping country code with data to find the countries with maximum restaurants
#India has the most number of restaurants so our analysis was based on India
country_grp= (merger.groupby(['Country'], as_index=False)['Restaurant ID'].count())
country_grp.columns = ['Country', 'Number of Restaurants']
country_grp= country_grp.sort_values(['Number of Restaurants'], ascending=False)
```

```
# Graphing the number of restaurants by country
sns.set(rc={'figure.figsize':(11.7,8.27)})
sns.barplot(country_grp['Country'], country_grp['Number of Restaurants'])
plt.xticks(rotation = 90)
plt.show()
```



90% of the data comes from India which means India has the most number of restaurants so our analysis was based on India

## 4. BUSINESS QUESTIONS

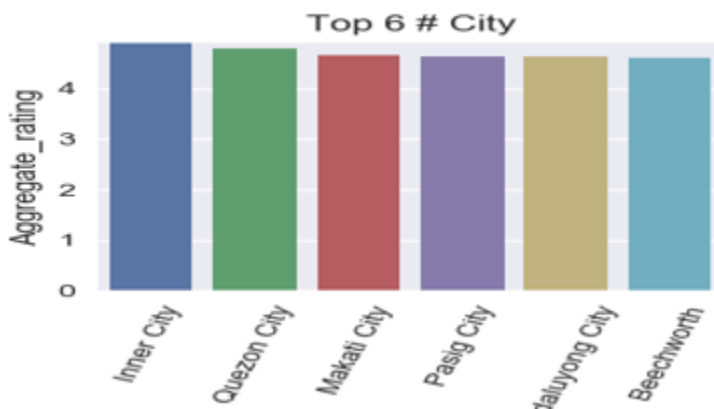
- **WORLD ANALYSIS**
  - Which countries have the maximum number of restaurants?
  - What are the top 6 cities with good aggregate rating around the world?
  - What are the top 6 cities with maximum restaurants
  - Which cuisines are preferred by customers the most?
- **INDIA ANALYSIS**
  - Which cities have the costliest food in India?
  - Which cities have the highest aggregate rating?
  - What are the price range distribution of restaurants within India?
  - What factors make the restaurant rating good?

Identifying the top 6 cities in the world that has the best aggregate ratings

```
# Identifying the best cities with good aggregate ratings
```

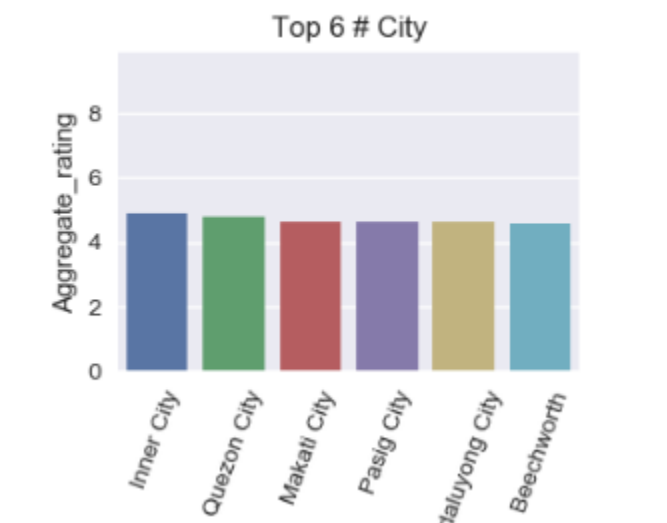
```
wealthgroupmean = proj_sub.groupby(['City']).Aggregate_rating.mean()
result = wealthgroupmean.sort_values('Aggregate_rating',ascending=False)
```

```
# Plot of top 6 cities
ax1 = plt.subplot2grid((3,3),(2,0))
new = result['Aggregate_rating'].reset_index()
sns.barplot(x = 'City', y = 'Aggregate_rating', data = new.head(6), ax = ax1)
ax1.tick_params(axis='x', rotation=70)
ax1.set_title('Top 6 # City', size = 12)
ax1.set_ylim([0, new['Aggregate_rating'].head(1).values+5])
for i, val in enumerate(new['Aggregate_rating'].head(6)):
    ax.text(i, val+50, val, color = 'grey', ha = 'center')
plt.show()
```



From the result we could see that Inner City has the best aggregate ratings for the restaurants

```
#Identifying count of restaurants in top 6 cities around the world
# plot for top 6 cities
ax = plt.subplot2grid((3,3),(2,0))
cnt = proj_sub['City'].value_counts().reset_index()
cnt.rename(columns = {'index':'City', 'City':'cnt'}, inplace = True)
sns.barplot(x = 'City', y = 'cnt', data = cnt.head(6), ax = ax)
ax.tick_params(axis='x', rotation=70)
ax.set_title('Top 6 # City', size = 12)
ax.set_ylim([0, cnt['cnt'].head(1).values+500])
for i, val in enumerate(cnt['cnt'].head(6)):
    ax.text(i, val+50, val, color = 'grey', ha = 'center')
plt.show()
```



Since India has the most number of data, so we created a dataset for India so that we can derive insight from the data.

```
#India analysis by taking data for India only
ind= pd.read_csv("C:/Users/Romil Godha/Desktop/zomato/India.csv", encoding = "ISO-8859-1")

# Taking specific columns
India = pd.DataFrame(ind, columns=['Restaurant Name','City','Cuisines','Average Cost for two', 'Currency', 'Has Online delivery', 'Is delivering now',
'Has Table booking', 'Price range', 'Aggregate rating', 'Rating text', 'Votes', 'Longitude', 'Latitude'])

# Renaming those columns
India.columns= ['Restaurant_Name','City','Cuisines','Average_Cost_for_two', 'Currency', 'Has_Online_delivery', 'Is_delivering_now',
'Has_Table_booking', 'Price_range', 'Aggregate_rating', 'Rating_text', 'Votes', 'Longitude', 'Latitude']
```

- Shape of India dataset

```
In : India.shape
(6513, 14)
```

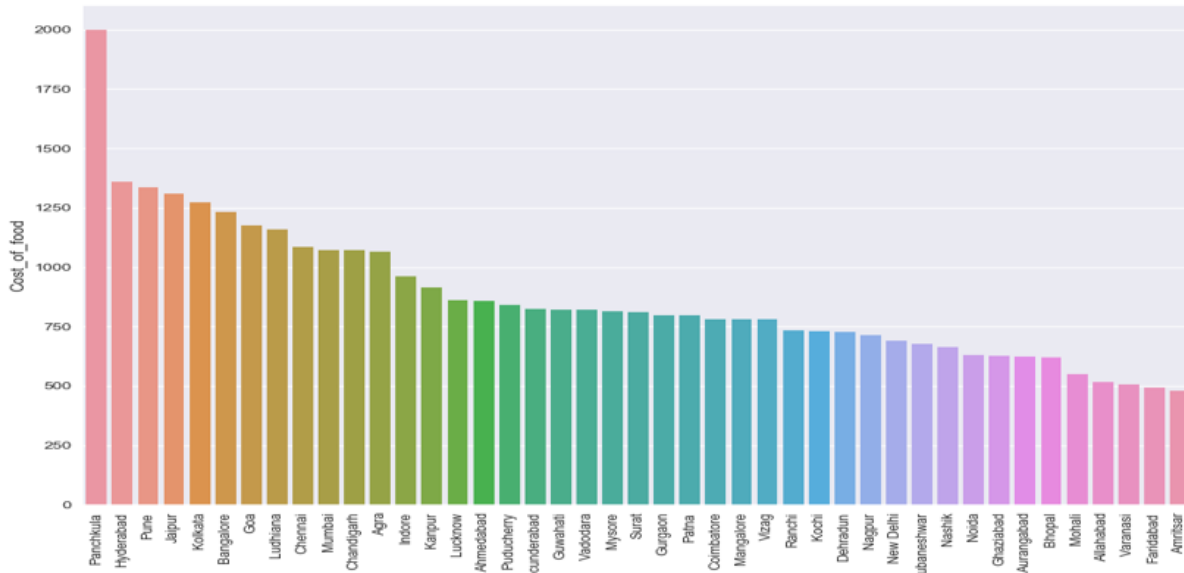
The dataset for India after preprocessing and cleaning has a total 6513 rows with 14 columns that are relevant for our study.

Identifying the cities in India which has the costliest food

```
# Which cities have the costliest food in India?
India_food=(India.groupby(['City'], as_index=False)['Average_Cost_for_two'].mean())
India_food.columns = ['City', 'Cost_of_food']
India_food= India_food.sort_values(['Cost_of_food'],ascending=False)
```



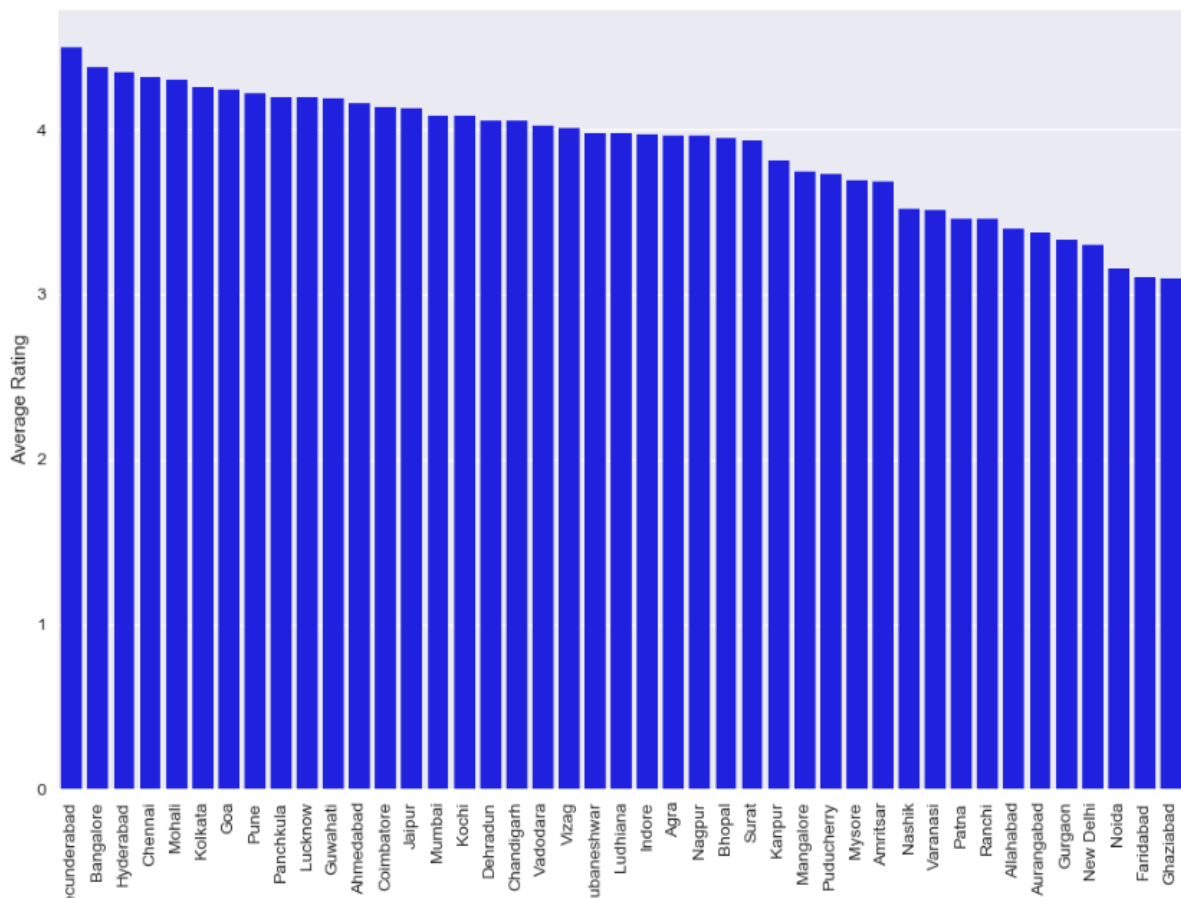
```
#Bar plot for costliest cities
sns.set(rc={'figure.figsize':(11.7,8.27)})
sns.barplot(India_food['City'], India_food['Cost_of_food'])
plt.xticks(rotation = 90)
plt.show()
```



The output shows that the city Panchkula has the most costliest food in India and Amritsar a city in the northern part of India has the least costly food.

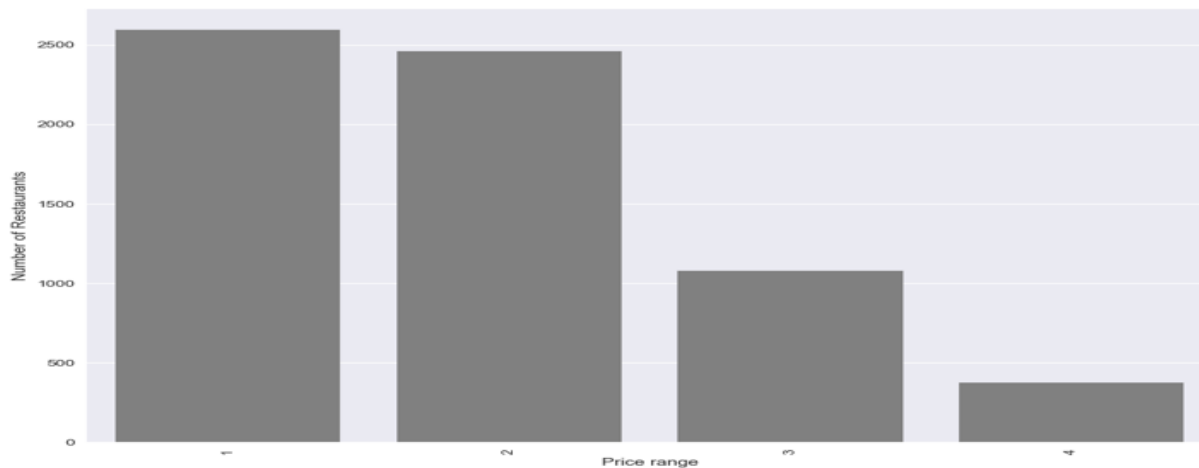
Identifying the highest aggregate rating for the Indian Cities

```
# Which cities have the highest aggregate rating in India?
India_rating=(India.groupby(['City'], as_index=False)['Aggregate_rating'].mean())
India_rating.columns = ['City', 'Average Rating']
India_rating= India_rating.sort_values(['Average Rating'],ascending=False)
```



Identifying the price range distribution distributed through the restaurants. Rating 1- Cheap cost, Rating 2- Average cost, Rating 3- high cost, Rating 4- very expensive

```
#Bar plot for price range distribution
sns.set(rc={'figure.figsize':(11.7,8.27)})
sns.barplot(India_range['Price range'], India_range['Number of Restaurants'], color= 'grey')
plt.xticks(rotation = 90)
plt.show()
```



## 5. Machine Learning

Machine learning algorithms to accurately predict the Dependent variable from Independent variables i.e. how for example rating (dependent variable) is provided based on price, services etc. (independent variable). Now by using the Accuracy is we can evaluate the classification models. Informally, **accuracy** is the fraction of predictions our model got right

```
# What factors make a restaurant successful
# Machine learning
df = pd.read_csv('India.csv')
df.head()
df1 = df.drop(['Restaurant Name', 'Country Code', 'Address', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines', 'Currency'], axis=1)
df1.head()

sns.pairplot(df1, hue='Rating text')

df1["Has Table booking"] = df1["Has Table booking"].astype('category')
df1["Has Table booking"] = df1["Has Table booking"].cat.codes
df1.head(5)

df1["Has Online delivery"] = df1["Has Online delivery"].astype('category')
df1["Has Online delivery"] = df1["Has Online delivery"].cat.codes

df1["Is delivering now"] = df1["Is delivering now"].astype('category')
df1["Is delivering now"] = df1["Is delivering now"].cat.codes

df1["Switch to order menu"] = df1["Switch to order menu"].astype('category')
df1["Switch to order menu"] = df1["Switch to order menu"].cat.codes

df1["Rating color"] = df1["Rating color"].astype('category')
df1["Rating color"] = df1["Rating color"].cat.codes

sns.pairplot(df1, hue='Rating text')
```

## 5.1 SUPPORT VECTOR MACHINES

```
from sklearn.svm import SVC
model=SVC()
model.fit(X_train,y_train)
pred = model.predict(X_test)
print(confusion_matrix(y_test,pred))
print('\n')
print(classification_report(y_test,pred))
```

**SVM**

avg / total      0.58

Using SVM or support vector machine algorithm we find that the accuracy is about 58%.

## 5.2 K-NEAREST NEIGHBOR

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=8)
knn.fit(x_train,y_train)
pred5 = knn.predict(X_test)
print(confusion_matrix(y_test,pred5))
print('\n')
print(classification_report(y_test,pred5))
```

**KNN**

avg / total      0.55

Using KNN or K- nearest neighbor algorithm we find that the accuracy is about 55%

## 6. Zomato Sentimental Analysis

### Description and Guideline

The primary aim was to extract twitter data from Zomato's twitter handle using Twitter API. After that the data was processed by putting it into a data frame. From the data various statistical computations were done to derive insights for the business questions. Finally, we applied the concept of sentiment analysis of extracted tweets from Zomato's twitter account to determine how the Zomato application is performing over the world market.

```
import tweepy          # Accessing Twitter's API
import pandas as pd    # To manipulate dataset
import numpy as np     # Used for statistical computation
```

Connecting to the Twitter API using personal developer authentication details.

```
consumer_key = '3s2IdCz9jX2Z6pej0dtebKrgl'
consumer_secret = '5604qFrlBpIcZblQpo3iKGRfGfYC9ZltdhSOJedS1gc0pwzLH7'
access_secret = 'lHiG1xT6yJsuevCF9BVYnrDgNM4f1zsTHhfKItMoCsKk'
access_token = '723396400206114816-YIt4W7Rj2zKbT60tyWT3Ex8byWnlSLR'
```

Utility function to setup the Twitter's API with the access keys provided above.

```
def twitter_setup():

    # Authentication and access using keys:
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_secret)

    # Return API with authentication:
    api = tweepy.API(auth)
    return api

extractor = twitter_setup()
```

To extract tweets as a list

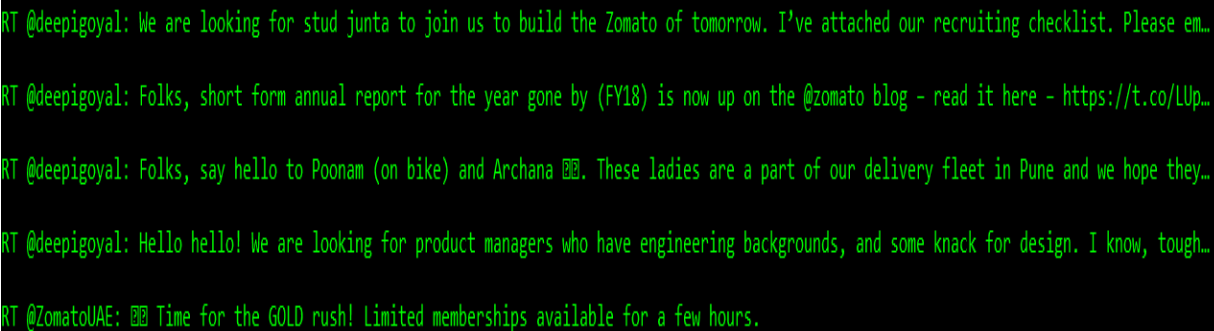
```
# We create a tweet list as follows:
tweets = extractor.user_timeline(screen_name="zomato", count=198)
print("Total tweets extracted: {}".format(len(tweets)))
```



```
Total tweets extracted: 198.
```

Getting the 5 most recent tweets

```
print("5 recent tweets:\n")
for tweet in tweets[:5]:
    print(tweet.text)
    print()
```



```
RT @deepigoyal: We are looking for stud junta to join us to build the Zomato of tomorrow. I've attached our recruiting checklist. Please em...
RT @deepigoyal: Folks, short form annual report for the year gone by (FY18) is now up on the @zomato blog - read it here - https://t.co/LUp...
RT @deepigoyal: Folks, say hello to Poonam (on bike) and Archana 🙋. These ladies are a part of our delivery fleet in Pune and we hope they...
RT @deepigoyal: Hello hello! We are looking for product managers who have engineering backgrounds, and some knack for design. I know, tough...
RT @ZomatoUAE: 🙋 Time for the GOLD rush! Limited memberships available for a few hours.
```

Extracting the Most Liked Tweet, Total characters used in the tweet and the count of likes for the tweet. Similarly, we found the most retweeted tweet, Total characters used in the tweet (quote tweet) and the count of retweets for the tweet, respectively after putting the data into a data frame.

```
data = pd.DataFrame(data=[tweet.text for tweet in tweets], columns=['Tweets'])
display(data.head(10))

data['len'] = np.array([len(tweet.text) for tweet in tweets])
data['ID'] = np.array([tweet.id for tweet in tweets])
data['Date'] = np.array([tweet.created_at for tweet in tweets])
data['Source'] = np.array([tweet.source for tweet in tweets])
data['Likes'] = np.array([tweet.favorite_count for tweet in tweets])
data['RTs'] = np.array([tweet.retweet_count for tweet in tweets])

fav_max = np.max(data['Likes'])
rt_max = np.max(data['RTs'])
```

```
fav = data[data.Likes == fav_max].index[0]
```

```
rt = data[data.RTs == rt_max].index[0]
```

```
Most Liked Tweet:
One more thing... https://t.co/RiiCeE0Xnv
Total Number of Tweets: 920
41 characters.

Most re-tweeted Tweet:
RT @alyssaherrera33: My sister has been in this world for 18 yrs and today is the first time she has been offered a menu at a restaurant ht...
Total Number of re-tweets: 99479
140 characters.
```

## 7. REGEX

A regular expression, regex or regexp is, in theoretical computer science and formal language theory, a sequence of characters that define a search pattern. Usually this pattern is then used by string searching algorithms for "find" or "find and replace" operations on strings, or for input validation

Using Regex to clean tweet by removing links and special characters.

```
return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\V\S+)", "", tweet).split())
```

## 8. SENTIMENT ANALYSIS

Sentiment Analysis is the process of determining whether a piece of writing is positive, negative or neutral. It's also known as opinion mining, deriving the opinion or attitude of text or speech. Here with the help of Textblob library, we have extracted the number of positive, negative and neutral tweets to analyze how good or bad, the brand Zomato and the application is doing on the social media and the world market.

Now writing a function to classify the polarity of a tweet using textblob

```
def analyze_sentiment(tweet):
    sentiment_analysis = TextBlob(clean_tweet(tweet))
    if sentiment_analysis.sentiment.polarity > 0:
        return 1
    elif sentiment_analysis.sentiment.polarity == 0:
        return 0
    else:
        return -1
```

Now creating a column with the result of the analysis:

```
data['SA'] = np.array([ analyze_sentiment(tweet) for tweet in data['Tweets'] ])
```

Displaying the updated dataframe with the new column:

```
positive_tweets = [ tweet for index, tweet in enumerate(data['Tweets']) if data['SA'][index] > 0]
neutral_tweets = [ tweet for index, tweet in enumerate(data['Tweets']) if data['SA'][index] == 0]
negative_tweets = [ tweet for index, tweet in enumerate(data['Tweets']) if data['SA'][index] < 0]

print("Percentage of positive tweets: {}".format(len(positive_tweets)*100/len(data['Tweets'])))
print("Percentage of neutral tweets: {}".format(len(neutral_tweets)*100/len(data['Tweets'])))
print("Percentage of negative tweets: {}".format(len(negative_tweets)*100/len(data['Tweets'])))
```

```
Percentage of positive tweets: 51.5151515151516%
Percentage of neutral tweets: 43.93939393939394%
Percentage of negative tweets: 4.545454545454546%
```

## 9. CONCLUSION:

From this project we were able to successfully address the relevant business questions using various statistical methods and machine learning methods and packages. We derived the relation between our independent and dependent variables and this might let us know how the factors like good food, food cost etc. drives the rating. Also, we performed sentimental analysis on Twitter and were able to derive that with 51.5 % positive tweets, Zomato on twitter is doing pretty well. There is a 44 % of neutral tweets and 4.5% of negative tweets which do not put Zomato on the bad light much.



## 10. CONTRIBUTIONS

CONTRIBUTIONS	
ROMIL GODHA	SOUMYA SAMAL
1. Data extraction and preprocessing	1. Twitter connection
2. Statistical operations on data	2. Tweets extraction
3. Answering the Business Questions	3. Data manipulation /Insert data in Dataframe
4. Machine Learning Algorithms	4. Creating visualizations and statistics
5. Creating Visualizations	5. Sentiment analysis using Twitter data
FINAL PROJECT PRESENTATION (TEAM)	