National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Ruchira Talekar
Student ID: X18185703

School of Computing
National College of Ireland

Supervisor: Dr. Paul Stynes
Dr. Pramod Pathak

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Ruchira Talekar
Student ID:	X18185703
Programme:	MSc. Data Analytics **Year:** 2020
Module:	Research Project
Lecturer:	Dr. Paul Stynes, Dr. Pramod Pathak
Submission Due Date:	28th September 2020
Project Title:	Classification of Customer Satisfaction for the Development of Hospitality Businesses
Word Count:	1025 **Page Count:** 12

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Ruchira Talekar

Date: 28th September 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Ruchira Talekar
X18185703
28th September 2020

1 Introduction

This document is representing the instructions to reproduce the classification of customer reviews for business development and to predict customer satisfaction. The steps and requirements for reproducing the machine learning models are as follows.

2 System Configuration

Hardware and software setup for the research work is explained below with respective diagrams.

2.1 Hardware Configuration

For hardware configuration, ASUS laptop has been used, its specification is intel core i5-8265U with the speed of 1.8GHz and 8 GB RAM with 1TB HDD shown in Figure 1.

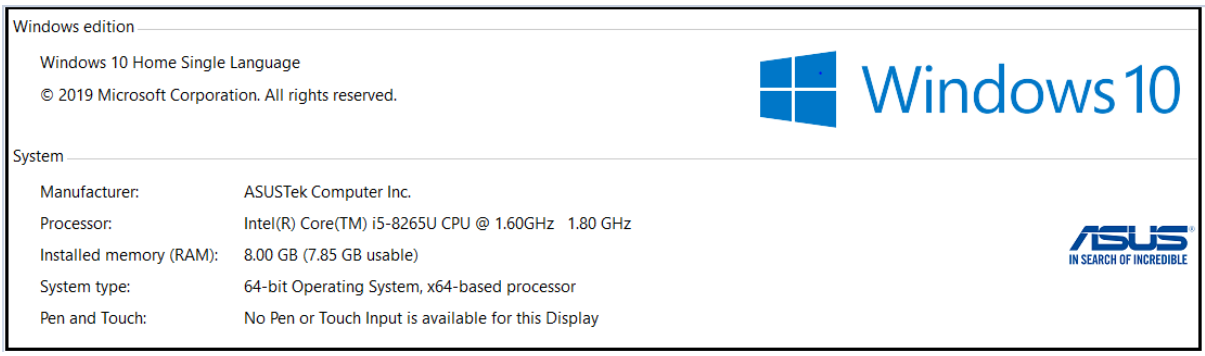

Figure 1: Hardware Configuration

2.2 Software Configuration

For software configuration, various software has been used like Jupyter notebook, MS-Excel, Power Bi and Twitter API Setup. Figure 2 is showing the version of the Jupyter Notebook that has been used with the help of Anaconda Navigator, while Figure 3 and Figure 4 is showing how to create an account and get the API keys for the twitter dataset.

1. Anaconda Navigator and Jupyter Notebook (6.0.1)

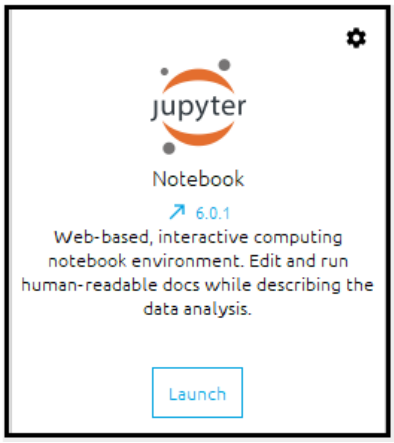

Figure 2: Software Configuration

2. Microsoft Excel- It has been used to store the datasets.

3. Power Bi- This software has been used for exploratory data analysis and visualization.

4. Twitter API Account creation and API keys

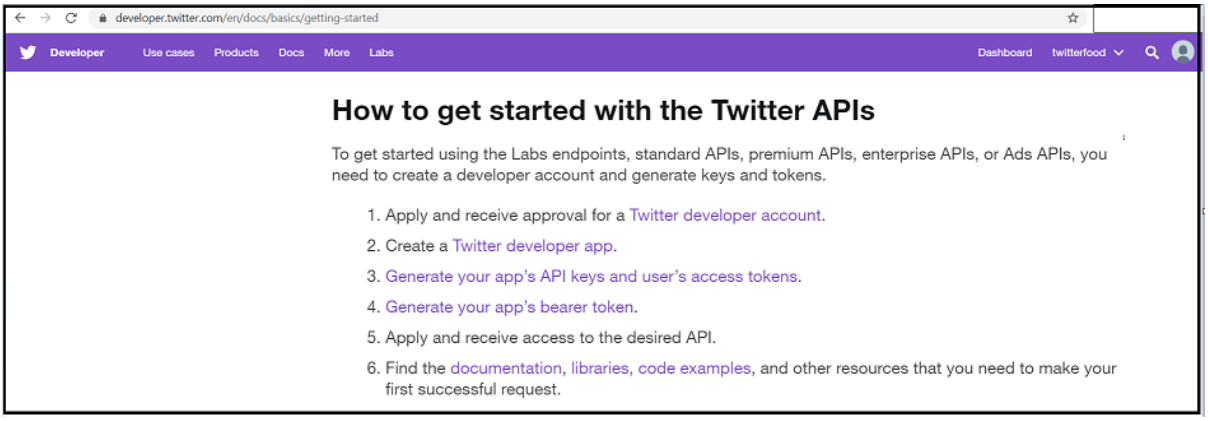

Figure 3: Steps for Twitter API Account Creation

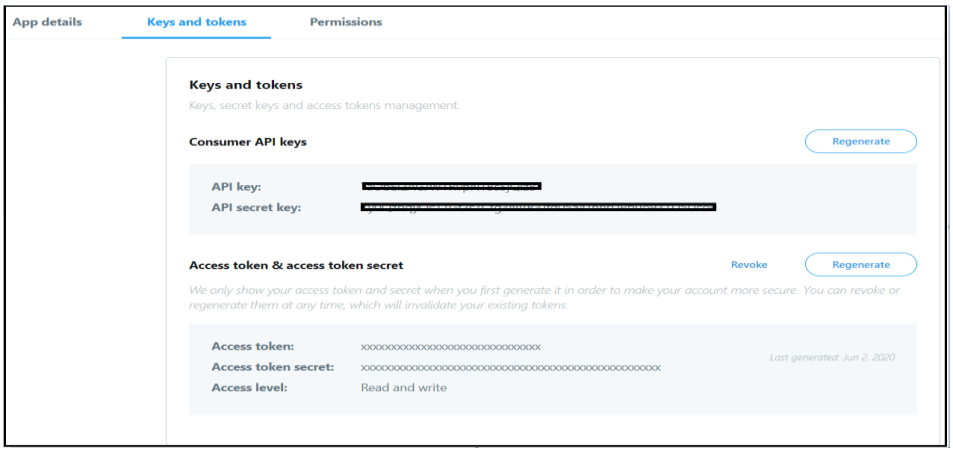

Figure 4: API keys and Access tokens provided by twitter

3 Implementation

3.1 Data Source

The list of data sources used in this project are given below:

Twitter Dataset: https://developer.twitter.com/en
Yelp Dataset: https://www.kaggle.com/yelp-dataset/yelp-dataset

3.2 Feature Engineering

1. Twitter Dataset has been extracted from twitter API using the below code shown in Figure 5 and Figure 6.

```
In [11]:

consumer_key = '************************'
consumer_secret = '*************************'
access_token= '********************'
access_token_secret = '*************************'


# attempt authentication
try:
# create OAuthHandler object
    auth = OAuthHandler(consumer_key, consumer_secret)
    # set access token and secret
    auth.set_access_token(access_token, access_token_secret)
    # create tweepy API object to fetch tweets
    api = tweepy.API(auth)
except:
    print("Error: Authentication Failed")
```

Figure 5: Code to retrieve the twitter data

```
In [ ]: ##########################Code to retrive tweet#####################################
import json
#Import the necessary methods from tweepy library
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream

#This is a basic listener that just prints received tweets to stdout.
class StdOutListener(StreamListener):

    def on_data(self, data):
        try:

            datajson = json.loads(data)

            created_at = datajson['created_at']
            # Writing to sample.json
            with open("result.txt", "a") as outfile:
                outfile.write(json.dumps(datajson))

            print("Tweet collected at " + str(created_at))

            #self.db.twitter_search.insert(datajson)
        except Exception as e:
            print(e)

    def on_error(self, status):
        print(status)

    def on_connect(self):
        # Called initially to connect to the Streaming API
        print("You are now connected to the streaming API.")

if __name__ == '__main__':

    #This handles Twitter authetification and the connection to Twitter Streaming API
    consumer_key = '************'
    consumer_secret = '************'
    access_key= '***************'
    access_secret = '***************'

    l = StdOutListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_key, access_secret)
    stream = Stream(auth, l)

    WORDS =['lamb', 'pork']
    #This line filter Twitter Streams to capture data by the keywords: lamb and pork
    stream.filter(track=WORDS)
```

Figure 6: Code to retrieve the twitter data

2. In the above code, JSON data has been stored in the text file. In the below code in Figure 7 and Figure 8, data has been converted into a structured format using python DataFrame for both the datasets.

Twitter Dataset

```
In [11]: def get_tweets3():

             tweet_path = 'result.txt'
             tweets_arr = []
             tweets_file = open(tweet_path, "r")
             for line in tweets_file:
                 try:
                     tweet = json.loads(line)
                     tweets_arr.append(tweet)
                 except:
                     continue



             tweets = []

             for i in range(0, len(tweets_arr)):
                 parsed_tweet = {}

                 # saving text of tweet
                 parsed_tweet['text'] =clean_tweet(tweets_arr[i]['text'])
                 # saving sentiment of tweet
                 parsed_tweet['sentiment'] = get_tweet_sentiment(tweets_arr[i]['text'])
                 parsed_tweet['Country']= tweets_arr[i]['user']['location']
                 list_of_hashtags=[]
                 for tag in tweets_arr[i]['text'].split():
                     if tag.startswith("#"):
                          list_of_hashtags.append( tag.strip("#"))
                 parsed_tweet['hashtag']=list_of_hashtags
                 # appending parsed tweet to tweets list
                 if tweet['retweet_count'] > 0:
                     # if tweet has retweets, ensure that it is appended only once
                     if parsed_tweet not in tweets:
                         tweets.append(parsed_tweet)
                 else:
                     tweets.append(parsed_tweet)

             # return parsed tweets
             return tweets
```

Figure7: Code for JSON data to Structured Data

Yelp Dataset

```
In [3]: path='D:/Subject Docs/513782_947211_compressed_review/xab.txt'
        json_data = pd.read_json(path, lines=True)
        json_data.head()
```

Out[3]:

	review_id	user_id	business_id	stars	useful	funny	cool	text	date
0	1kXW2On0xOoAjxnhOjDlqQ	5AyIq95TseujePx9S0NKJg	3-aEgS7X2jrbxA7sA1nARw	5	1	0	1	Really loved this place. \nI ordered two appet...	2016-11-12 23:52:59
1	sUTJxFI9oYPuwDZuayASaA	i-PZwf519Vu4ybNt9D8oEA	htVvtLIFftBLqzRISjReDw	3	1	0	0	My friends and I went to the Cheesecake Factor...	2018-01-11 06:31:09
2	_er2fysEUUsi-Ib0GPnt_A	fQVaFbT1NA7uM8sN_sC3Gg	dBX0TUPNZ1WtVzv5jfYE2w	4	0	0	0	I was in Richmond Hill visiting family when th...	2014-02-28 03:35:56
3	4d6AdWZM27vwET5vsOYBkA	qXn9y0Dq0DWF9Q6_t9t6qQ	Wxxvi3LZbHNIDwJ-ZimtnA	5	0	0	0	This is one of the top 5 hotels in Las Vegas, ...	2016-03-09 17:43:27
4	5RcZelO4u3-4oBa5rY9rVw	gYLmEjgqSz9RF3vybDtW2g	d_L-rfS1vT3JMzgCUGtiow	4	3	0	0	Great places if you can get a seat on the pati...	2018-06-10 23:47:27

Figure 8: Code for JSON data to Structured Data

3. Data has been pre-processed and cleaned that has been shown in Figure 9 and Figure 10 for both the datasets.

Twitter Dataset

```
In [11]: def processTweet(tweet):
             tweet  = "".join([char for char in tweet if char not in string.punctuation])
             tweet = re.sub('[0-9]+', '', tweet)
             tweet = tweet.lower() # convert text to lower-case
             tweet = re.sub('((www\.[^\s]+)|(https?://[^\s]+))', 'URL', tweet) # remove URLs
             tweet = re.sub('@[^\s]+', 'AT_USER', tweet) # remove usernames
             tweet = re.sub(r'#([^\s]+)', r'\1', tweet) # remove the # in #hashtag

             return tweet

In [12]: def clean_tweet(tweet):
             return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t]) |(\w+:\/\/\S+) |", " ", tweet).split())
```

Figure 9: Pre-processing of Twitter Data

Yelp Dataset

```
In [8]: def preprocess(text):
            clean_data = []
            for x in (text[:]):
                new_text = re.sub('<.*?>', '', str(x))   # remove HTML tags
                new_text = re.sub(r'[^\w\s]', '', new_text) # remove punc.
                new_text = re.sub(r'\d+','',new_text)# remove numbers
                new_text = new_text.lower() # lower case, .upper() for upper
                new_text = new_text.strip() #whitespace
                new_text = nltk.word_tokenize(new_text)
                if new_text != '':
                    clean_data.append(" ".join(new_text))

            return clean_data

In [9]: json_data['clean_txt'] = preprocess(json_data['text'])

In [12]: json_data= json_data.drop(['useful'], axis = 1)
         json_data= json_data.drop(['funny'], axis = 1)
         json_data= json_data.drop(['cool'], axis = 1)
```

Figure 10: Pre-processing of Yelp Data

4. Sentiment as a dependent column created using below code shown in Figure 11 and Figure 12

Twitter Dataset

```
In [13]:
def get_tweet_sentiment(tweet):
    # create TextBlob object of passed tweet text
    analysis = TextBlob(clean_tweet(tweet))
    # set sentiment
    if analysis.sentiment.polarity > 0:
        return 'positive'
    elif analysis.sentiment.polarity == 0:
        return 'neutral'
    else:
        return 'negative'
```

Figure 11: Dependent column created Twitter Data

Yelp dataset

```
In [5]:
for index, row in json_data.iterrows():
    if row['stars'] < 3:
        json_data.at[index,'sentiment']='negative'
    elif row['stars'] == 3:
        json_data.at[index,'sentiment']='neutral'
    else:
        json_data.at[index,'sentiment']='positive'


print(json_data.head())
```

Figure 12: Dependent column created Yelp Data

5. Stemming of words shown in below Figure 13.

```
In [16]:
from nltk.stem.snowball import SnowballStemmer
snowball = SnowballStemmer(language = 'english')
def stemming(words):
    new = []
    stem_words = [snowball.stem(x) for x in (words[:][0])]
    new.append(stem_words)
    return new
```

Figure 13: Stemming of Text

6. Term frequency-inverse document frequency (TF-IDF) Vectorization shown in below Figure 14.

```
In [65]: from sklearn.feature_extraction.text import TfidfVectorizer
         # Create feature vectors
         vectorizer = TfidfVectorizer(min_df = 5,
                                      max_df = 0.8,
                                      sublinear_tf = True,
                                      use_idf = True)
         train_vectors = vectorizer.fit_transform(X_train)
         test_vectors = vectorizer.transform(X_test)
```

Figure 14: Term frequency-inverse document frequency (TF-IDF) Vectorization of Text

7. Below Figure 15 is the Naïve Bayes Model (Experiment 1 and 2) that has been implemented for both the dataset.

```
In [61]:
         X = dataframe['text'].values
         y = dataframe['sentiment'].values

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         MNB = MultinomialNaiveBayes(
         classes=np.unique(y),
         tokenizer=Tokenizer()
         ).fit(X_train, y_train)

         y_hat = MNB.predict(X_test)
```

Figure 15: Naïve Bayes Classifier

8. Below Figure 16 is the Support Vector Machine Model (Experiment 1 and 2) that has been implemented for both the dataset.

```
In [3]: import time
        from sklearn import svm
        from sklearn.metrics import classification_report
        # Perform classification with SVM, kernel=linear
        classifier_linear = svm.SVC(kernel='linear')
        t0 = time.time()
        classifier_linear.fit(train_vectors, y_train)
        t1 = time.time()
        prediction_linear = classifier_linear.predict(test_vectors)
        t2 = time.time()
        time_linear_train = t1-t0
        time_linear_predict = t2-t1
        # results
        print("Training time: %fs; Prediction time: %fs" % (time_linear_train, time_linear_predict))
```

Figure 16: Support Vector Machine Classifier

9. Below Figure 17 is the Random Forest Model (Experiment 3) that has been implemented for the Yelp dataset.

```
In [342]: from sklearn.feature_extraction.text import TfidfVectorizer
          # Create feature vectors
          vectorizer1 = TfidfVectorizer(min_df = 5,
                                        max_df = 0.8,
                                        sublinear_tf = True, use_idf = True,
                                        norm='l2', ngram_range=(1, 2))
```

```
In [322]:
# this block is to split the dataset into training and testing set
X1 = json_data_split['clean_txt']
Y1 = json_data_split['sentiment']
X_train1, X_test1, y_train1, y_test1 = train_test_split(X1, Y1, test_size=0.25)

# instead of doing these steps one at a time, we can use a pipeline to complete then all at once
pipeline = Pipeline([('vect', vectorizer),
                     ('chi',  SelectKBest(chi2, k=1200)),
                     ('clf', RandomForestClassifier())])

# fitting our model and save it in a pickle for later use
model = pipeline.fit(X_train1, y_train1)
with open('RandomForest.pickle', 'wb') as f:
    pickle.dump(model, f)

ytest = np.array(y_test1)
```

Figure 17: Random Forest Classifier

10. K-means Clustering with term frequency-inverse document frequency (TF-IDF) (Experiment 4) has been implemented using the below code shown in Figure 18.

```
In [64]: from sklearn.cluster import KMeans
         num_clusters = 4
         km = KMeans(n_clusters=num_clusters)
         km.fit(train_vectors)
         clusters = km.labels_.tolist()

In [65]: review = {'review': X_train.tolist(), 'Cluster': clusters}
         frame = pd.DataFrame(review, index = [clusters])
         frame
```

Figure 18: K-means Clustering with TF-IDF

11. Counts of reviews per cluster have been shown in Figure 19.

```
In [66]: frame['Cluster'].value_counts()

Out[66]: 0    19577
         3    16654
         1    15837
         2    11932
         Name: Cluster, dtype: int64
```

Figure 19: Value Count Per Cluster

12. Different clusters with reviews are shown below in Figure 20.

	review	Cluster
2	i see all the poor reviews and honestly when b...	2
0	darren the manager was exceptional with our pa...	0
3	this is one of those places that i go back to ...	3
0	great street tacos pollo carnitas al pastorall...	0
0	refreshing and delicious homemade popsicles gr...	0
...	...	...
2	i got my first tattoo done by tyson and i coul...	2
2	omg the line for jcole is insane i knew someon...	2
1	we loved this place we had lunch there and the...	1
3	ive never had any problems with this store the...	3
1	first and last time i will come here took over...	1

Figure 20: Cluster with Reviews

3.3 Evaluation Methods

1. For machine learning classifiers, various performance measures have been represented using the below code shown in Figure 21. This is the result of the random forest algorithm.

```
In [344]: print(classification_report(ytest3, model.predict(X_test3)))
          print(confusion_matrix(ytest3, model.predict(X_test3)))
```

```
              precision    recall  f1-score   support

    negative       0.83      0.77      0.80      4633
    positive       0.92      0.95      0.93     13137

    accuracy                           0.90     17770
   macro avg       0.88      0.86      0.87     17770
weighted avg       0.90      0.90      0.90     17770

[[ 3569  1064]
 [  712 12425]]
```

Figure 21: Classification Report for Random Forest Model

2. For clustering, the elbow method has been implemented using the below code shown in Figure 22.

```
In [62]: from sklearn.cluster import KMeans
         wcss = []
         for k in range(1, 10):
             km = KMeans(n_clusters=k).fit(train_vectors)
             km.fit(train_vectors)
             wcss.append(km.inertia_)

In [63]: import matplotlib.pyplot as plt
         K=range(1,10)
         plt.plot(K, wcss, 'bx-')
         plt.xlabel('k')
         plt.ylabel('wcss')
         plt.title('elbow')
         plt.show()
```

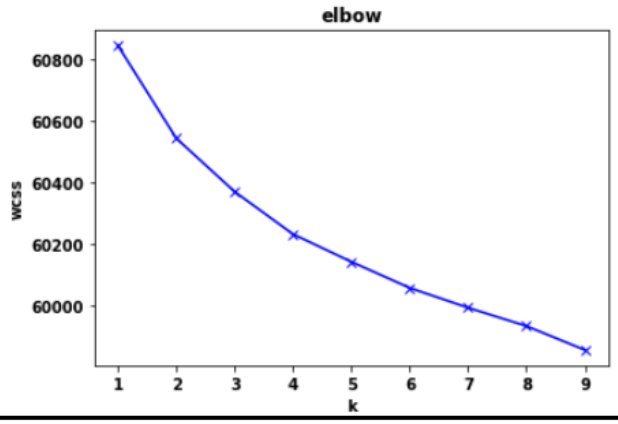

Figure 22: Elbow Method for K-means Clustering

4 Visualization and Exploratory Data Analysis

1. Yelp Businesses have been spread all over the world. Figure 23 is representing the locations of various businesses around the world.

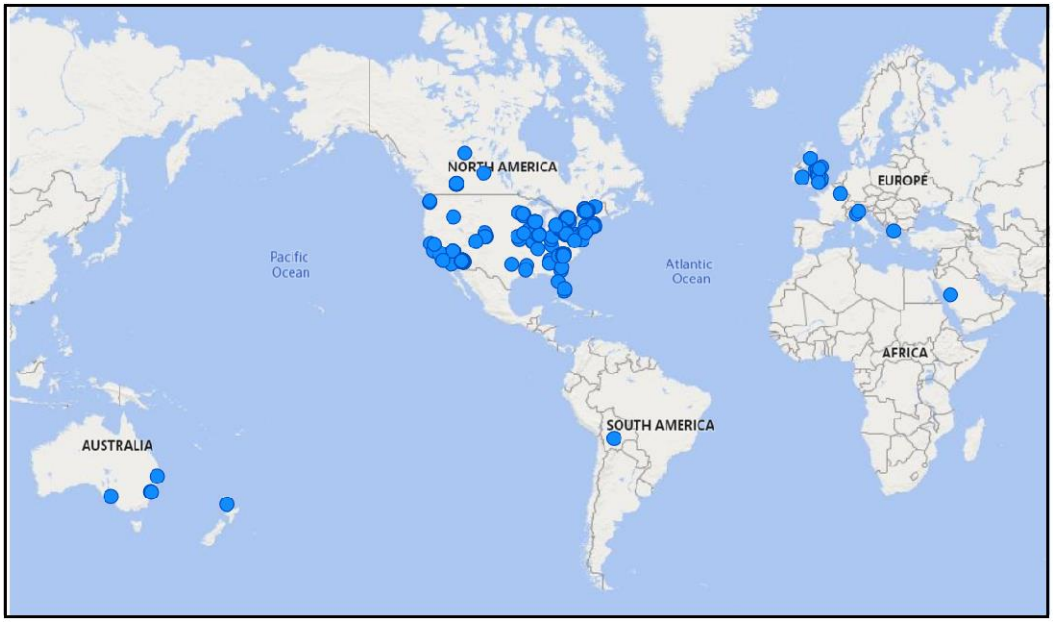

Figure 22: World Map of Yelp Businesses

2. Pie chart of positive, negative, and neutral reviews that has been shown in Figure 23 given below.

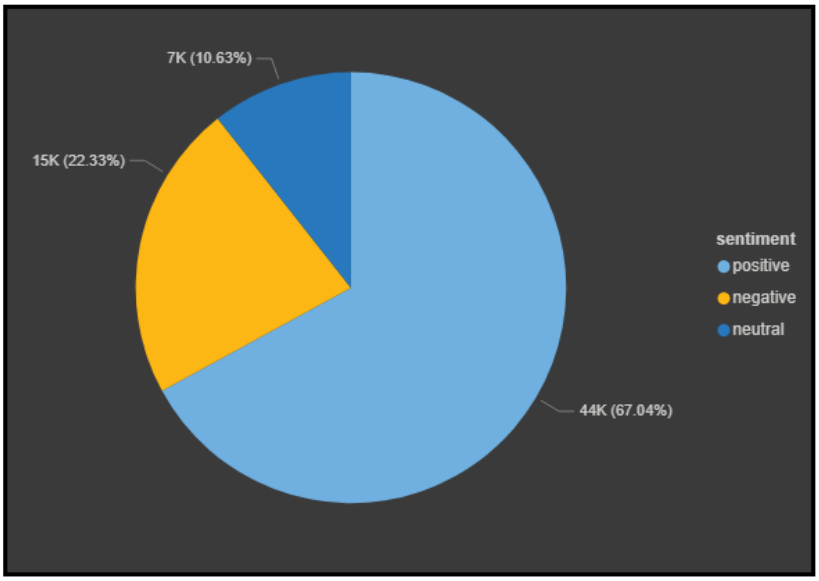

Figure 22: Pie Chart of Sentiments

3. Review Count per business has been shown in Figure 23 using the Bar Chart.

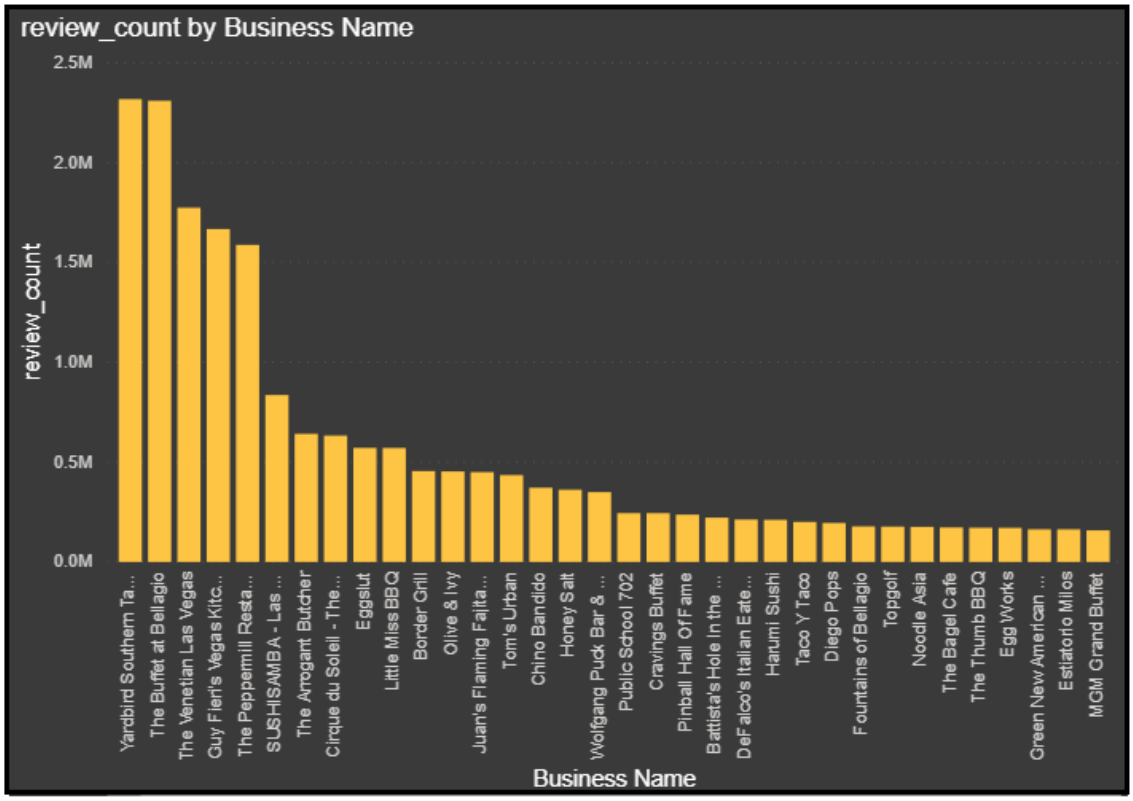

Figure 22: Bar Chart of Review Count Vs. Business

4. The popularity of various hospitality businesses has been shown below in Figure 23.

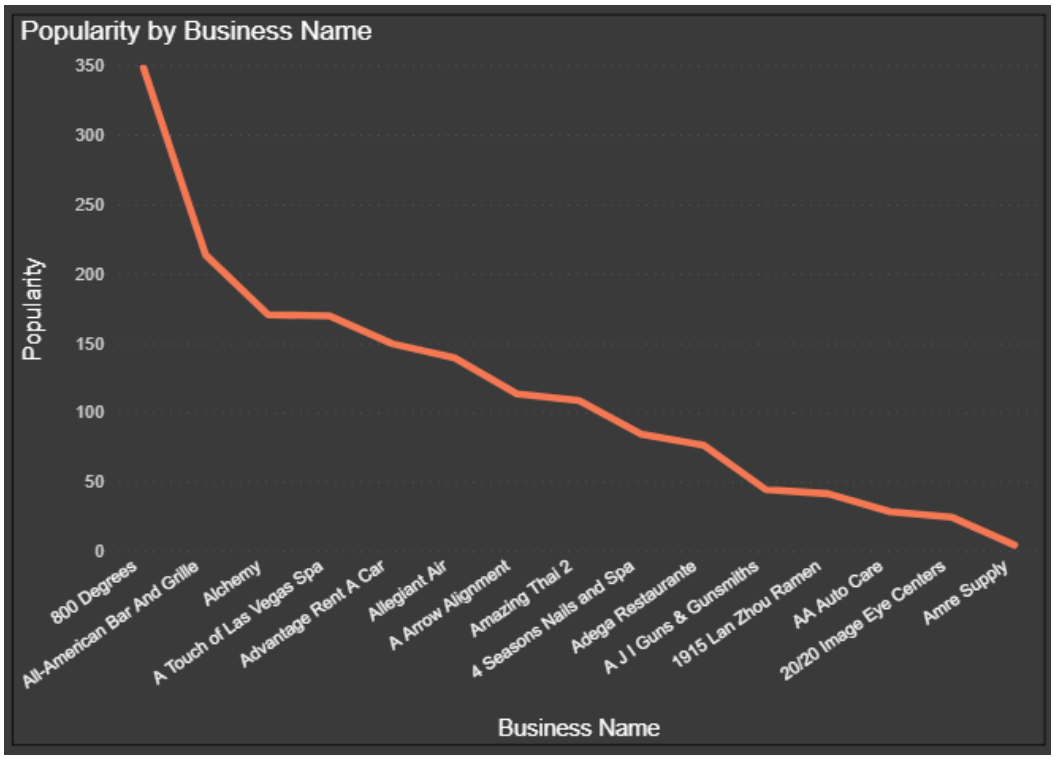

Figure 23: Popularity Vs Business