Project2_final

February 24, 2021

0.1 Text Mining of Hotel Reviews

0.1.1 Abstract

The objective of this project is to utilize various Text Mining techniques to attain and discover patterns, trends, and insights using the Hotel Reviews Data collected by Jiashen Liu. The outcome of this analysis will investigate the customers' reviews word for word that is up for cleaning and analysis. The reviews, positive or negative, will provide businesses valuable information on how they can enhance the quality and comfort of the customers' time of stay.

0.1.2 Introduction

Whenever a user or viewer visualizes a tweet, transcript, or a report, they do not realize how much of the text is being used by data professionals for research and authentication purposes. The concept of text mining is not only to "read" documents in a body of information, but to cover ground in terms of knowledge bases and open sources within the search. The items discovered at hand invoke emotions in their form of speech and reveal connections one cannot perceive from a linguistic standpoint. This creates a need for systems that can read and understand information in a manner that is scalable and dynamic.

0.1.3 Ethical ML Framework

The acknowledgements mentioned on the website indicate that all the data is readily accessible to the public as it is scraped and published through a travel agency website "Booking.com." However, since the Hotel Reviews dataset has an open platform and collected with demographic information, many of the ethical ML framework principles do not apply. Some of those that do apply hold major implications in terms of results from the models in our notebooks. For instance, when it comes to the area of Social Impact, human impact is highlighted when it comes to businesses achieving change and sustainability in their operations. Plus having consistent flow of communication amongst staff and employees is vital in establishing stakeholder dynamics. On the other hand, when considering Accuracy and Trust, transparency is vital when insights are clearly stated and decision-making flows back-and-forth amongst stakeholders. This can be seen in the case of market research when relying on customers to provide their feedback on their experience at the hotels they have stayed (seeking their consent in completing surveys). Hence, depending on the scenario and impact of the application, there would be more stricter measures in appropriating the techniques utilized for this project.

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```python
import re
import seaborn as sns
from geopy.geocoders import Nominatim

import nltk
from nltk.corpus import wordnet
from nltk import pos_tag
from nltk.corpus import stopwords
from nltk.tokenize import WhitespaceTokenizer
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from bs4 import BeautifulSoup
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk.corpus import names
import pycountry
from geograpy import places
import geopy.geocoders
from geopy.geocoders import Nominatim
from wordcloud import WordCloud

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import TruncatedSVD
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from sklearn.linear_model import LogisticRegression
import sklearn.model_selection
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score, StratifiedKFold
from sklearn.metrics import␣
↪log_loss,confusion_matrix,classification_report,roc_curve,auc



import string

import operator
import multiprocessing
```

```python
from time import time
from gensim.models import Word2Vec
from gensim.models.phrases import Phrases, Phraser

import collections


english_stemmer=nltk.stem.SnowballStemmer('english')
# With the following operation we set seaborn library as plotting library.
sns.set()

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
warnings.simplefilter(action='ignore', category=UserWarning)
%matplotlib inline
```

[2]:
```python
# Read the csv as DataFrame.
data = pd.read_csv('Hotel_Reviews.csv')
data.head()
```

[2]:
```
                                       Hotel_Address  \
0   s Gravesandestraat 55 Oost 1092 AA Amsterdam …
1   s Gravesandestraat 55 Oost 1092 AA Amsterdam …
2   s Gravesandestraat 55 Oost 1092 AA Amsterdam …
3   s Gravesandestraat 55 Oost 1092 AA Amsterdam …
4   s Gravesandestraat 55 Oost 1092 AA Amsterdam …

   Additional_Number_of_Scoring Review_Date  Average_Score   Hotel_Name  \
0                           194    8/3/2017            7.7  Hotel Arena
1                           194    8/3/2017            7.7  Hotel Arena
2                           194   7/31/2017            7.7  Hotel Arena
3                           194   7/31/2017            7.7  Hotel Arena
4                           194   7/24/2017            7.7  Hotel Arena

  Reviewer_Nationality                                    Negative_Review  \
0             Russia    I am so angry that i made this post available…
1            Ireland                                        No Negative
2          Australia    Rooms are nice but for elderly a bit difficul…
3     United Kingdom    My room was dirty and I was afraid to walk ba…
4        New Zealand    You When I booked with your company on line y…

   Review_Total_Negative_Word_Counts  Total_Number_of_Reviews  \
0                                397                     1403
1                                  0                     1403
2                                 42                     1403
3                                210                     1403
4                                140                     1403
```

```
                                      Positive_Review  \
0   Only the park outside of the hotel was beauti…
1   No real complaints the hotel was great great …
2   Location was good and staff were ok It is cut…
3   Great location in nice surroundings the bar a…
4    Amazing location and building Romantic setting

   Review_Total_Positive_Word_Counts  \
0                                 11
1                                105
2                                 21
3                                 26
4                                  8

   Total_Number_of_Reviews_Reviewer_Has_Given  Reviewer_Score  \
0                                           7             2.9
1                                           7             7.5
2                                           9             7.1
3                                           1             3.8
4                                           3             6.7

                                                Tags days_since_review  \
0  [' Leisure trip ', ' Couple ', ' Duplex Double…           0 days
1  [' Leisure trip ', ' Couple ', ' Duplex Double…           0 days
2  [' Leisure trip ', ' Family with young childre…           3 days
3  [' Leisure trip ', ' Solo traveler ', ' Duplex…           3 days
4  [' Leisure trip ', ' Couple ', ' Suite ', ' St…          10 days

         lat       lng
0  52.360576  4.915968
1  52.360576  4.915968
2  52.360576  4.915968
3  52.360576  4.915968
4  52.360576  4.915968
```

0.1.4 Metadata

- Hotel_Address: Address of hotel.

- Review_Date: Date when reviewer posted the corresponding review.

- Average_Score: Average Score of the hotel, calculated based on the latest comment in the last year.

- Hotel_Name: Name of Hotel

- Reviewer_Nationality: Nationality of Reviewer

- Negative_Review: Negative Review the reviewer gave to the hotel. If the reviewer does not give the negative review, then it should be: 'No Negative'

- ReviewTotalNegativeWordCounts: Total number of words in the negative review.

- Positive_Review: Positive Review the reviewer gave to the hotel. If the reviewer does not give the negative review, then it should be: 'No Positive'

- ReviewTotalPositiveWordCounts: Total number of words in the positive review.

- Reviewer_Score: Score the reviewer has given to the hotel, based on his/her experience

- TotalNumberofReviewsReviewerHasGiven: Number of Reviews the reviewers has given in the past.

- TotalNumberof_Reviews: Total number of valid reviews the hotel has.

- Tags: Tags reviewer gave the hotel.

- dayssincereview: Duration between the review date and scrape date.

- AdditionalNumberof_Scoring: There are also some guests who just made a scoring on the service rather than a review. This number indicates how many valid scores without review in there.

- lat: Latitude of the hotel

- lng: longtitude of the hotel

0.1.5 Data Unterstanding and Cleaning

[3]:
```
data.describe()
```

[3]:
```
       Additional_Number_of_Scoring  Average_Score  \
count                 515738.000000  515738.000000
mean                     498.081836       8.397487
std                      500.538467       0.548048
min                        1.000000       5.200000
25%                      169.000000       8.100000
50%                      341.000000       8.400000
75%                      660.000000       8.800000
max                     2682.000000       9.800000

       Review_Total_Negative_Word_Counts  Total_Number_of_Reviews  \
```

```
count                  515738.000000             515738.000000
mean                       18.539450               2743.743944
std                        29.690831               2317.464868
min                         0.000000                 43.000000
25%                         2.000000               1161.000000
50%                         9.000000               2134.000000
75%                        23.000000               3613.000000
max                       408.000000              16670.000000

       Review_Total_Positive_Word_Counts  \
count                      515738.000000
mean                           17.776458
std                            21.804185
min                             0.000000
25%                             5.000000
50%                            11.000000
75%                            22.000000
max                           395.000000

       Total_Number_of_Reviews_Reviewer_Has_Given  Reviewer_Score  \
count                               515738.000000   515738.000000
mean                                     7.166001        8.395077
std                                     11.040228        1.637856
min                                      1.000000        2.500000
25%                                      1.000000        7.500000
50%                                      3.000000        8.800000
75%                                      8.000000        9.600000
max                                    355.000000       10.000000

                 lat            lng
count  512470.000000  512470.000000
mean       49.442439       2.823803
std         3.466325       4.579425
min        41.328376      -0.369758
25%        48.214662      -0.143372
50%        51.499981       0.010607
75%        51.516288       4.834443
max        52.400181      16.429233
```

[4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 515738 entries, 0 to 515737
Data columns (total 17 columns):
 #   Column                                      Non-Null Count   Dtype
---  ------                                      --------------   -----
 0   Hotel_Address                               515738 non-null  object
 1   Additional_Number_of_Scoring                515738 non-null  int64
```

```
 2   Review_Date                                 515738 non-null  object
 3   Average_Score                               515738 non-null  float64
 4   Hotel_Name                                  515738 non-null  object
 5   Reviewer_Nationality                        515738 non-null  object
 6   Negative_Review                             515738 non-null  object
 7   Review_Total_Negative_Word_Counts           515738 non-null  int64
 8   Total_Number_of_Reviews                     515738 non-null  int64
 9   Positive_Review                             515738 non-null  object
 10  Review_Total_Positive_Word_Counts           515738 non-null  int64
 11  Total_Number_of_Reviews_Reviewer_Has_Given  515738 non-null  int64
 12  Reviewer_Score                              515738 non-null  float64
 13  Tags                                        515738 non-null  object
 14  days_since_review                           515738 non-null  object
 15  lat                                         512470 non-null  float64
 16  lng                                         512470 non-null  float64
dtypes: float64(4), int64(5), object(8)
memory usage: 66.9+ MB
```

0.1.6 Overview

In the Hotel Reviews Data, the dataset contains 515,000 customer reviews and scoring of 1493 luxury hotels across Europe compiled in the span of January 2015 to November 2017. As part of preprocessing, the owner Jiashen Liu removed Unicode and punctuation in the text data and transform text into lower case. Based on the different approaches to implement a text mining application, we utilize natural language processing and sentiment analysis to transform the text of customer reviews into data that can be used for cleaning and analysis. The reason we are performing this analysis is to improve our understanding of the corpus that has been inserted on the user-end. The results incurred from manipulating vast information can help identify entities and extract new relationships between them that would otherwise go undiscovered. The model to be developed can be used as a means of enhancing customers' interests that subject to trial development and quality assurance based on their feedback within the years mentioned. This application could be implemented in market research surveys (physical or online) at the end of the customer's stay at hotels to fine tune their hospitality experience for next time.

[5]:
```
len(data) #total number of reviews
```

[5]:
```
515738
```

[6]:
```
data.isnull().sum()
```

[6]:
```
Hotel_Address                                 0
Additional_Number_of_Scoring                  0
Review_Date                                   0
Average_Score                                 0
Hotel_Name                                    0
Reviewer_Nationality                          0
Negative_Review                               0
Review_Total_Negative_Word_Counts             0
```

```
Total_Number_of_Reviews                          0
Positive_Review                                  0
Review_Total_Positive_Word_Counts                0
Total_Number_of_Reviews_Reviewer_Has_Given       0
Reviewer_Score                                   0
Tags                                             0
days_since_review                                0
lat                                           3268
lng                                           3268
dtype: int64
```

[7]:
```
len(data.Hotel_Name.unique()) #total number of hotels being reviewed in this␣
↪dataset
```

[7]:
```
1492
```

[8]:
```
# First We will identify the hotel location City and Country by

print(data.Hotel_Address)
```

```
0           s Gravesandestraat 55 Oost 1092 AA Amsterdam …
1           s Gravesandestraat 55 Oost 1092 AA Amsterdam …
2           s Gravesandestraat 55 Oost 1092 AA Amsterdam …
3           s Gravesandestraat 55 Oost 1092 AA Amsterdam …
4           s Gravesandestraat 55 Oost 1092 AA Amsterdam …
                               …
515733    Wurzbachgasse 21 15 Rudolfsheim F nfhaus 1150 …
515734    Wurzbachgasse 21 15 Rudolfsheim F nfhaus 1150 …
515735    Wurzbachgasse 21 15 Rudolfsheim F nfhaus 1150 …
515736    Wurzbachgasse 21 15 Rudolfsheim F nfhaus 1150 …
515737    Wurzbachgasse 21 15 Rudolfsheim F nfhaus 1150 …
Name: Hotel_Address, Length: 515738, dtype: object
```

[9]:
```
data['Country'] = data.Hotel_Address.apply(lambda x: x.split(' ')[-1])
data['City'] = data.Hotel_Address.apply(lambda x: x.split(' ')[-2])
```

[10]:
```
print(data.Country.unique())
print(data.City.unique())
```

```
['Netherlands' 'Kingdom' 'France' 'Spain' 'Italy' 'Austria']
['Amsterdam' 'United' 'Paris' 'Barcelona' 'Milan' 'Vienna']
```

[11]:
```
data.Country = data.Country.str.replace('Kingdom','UK')
data.City = data.City.str.replace('United','London')
```

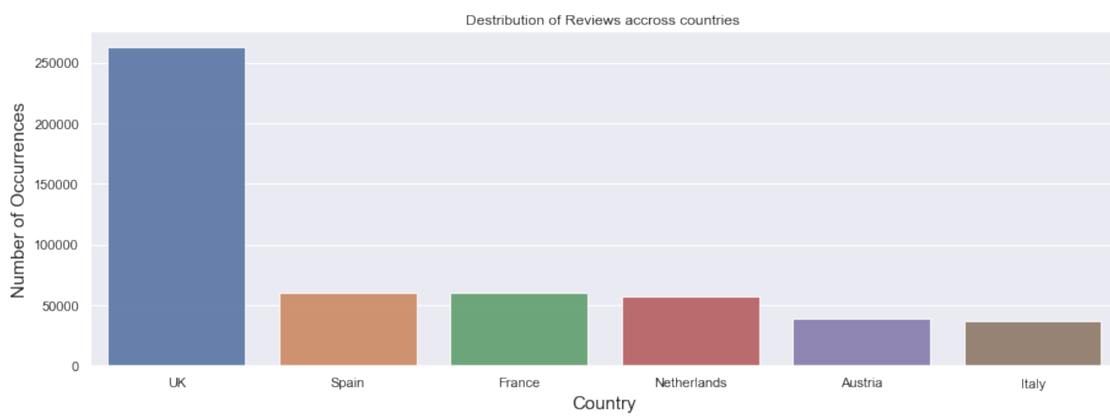
[12]:
```
print(data.Country.unique())
print(data.City.unique())
```

```
['Netherlands' 'UK' 'France' 'Spain' 'Italy' 'Austria']
['Amsterdam' 'London' 'Paris' 'Barcelona' 'Milan' 'Vienna']
```

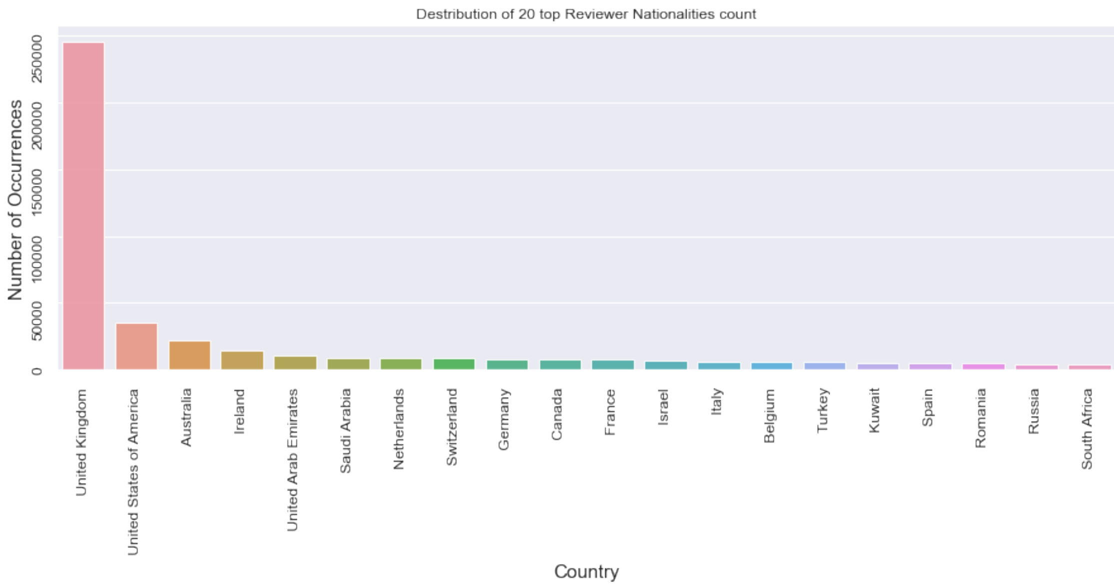
Next, we dive into Feature Engineering. Through feature engineering, we established various distributions of the reviews across the countries in Europe, the nationality of the reviewers who partook in the dataset, and the hotels taken in by the reviewers. All these distributions extracted features that catered to the average scores representing as key features in the predictive models ahead. By using feature engineering, we transform the given feature "space" and provide it with mathematical functions that help reduce the modelling error (repeated columns and numbers) for our given target model.

0.1.7 Feature engineering

[13]:
```
Country_count  = data["Country"].value_counts()
Country_count = Country_count[:10,]
plt.figure(figsize=(15,5))
sns.barplot(Country_count.index, Country_count.values, alpha=0.9)
plt.title('Destribution of Reviews accross countries ')
plt.ylabel('Number of Occurrences', fontsize=15)
plt.xlabel('Country', fontsize=15)
plt.show()
```

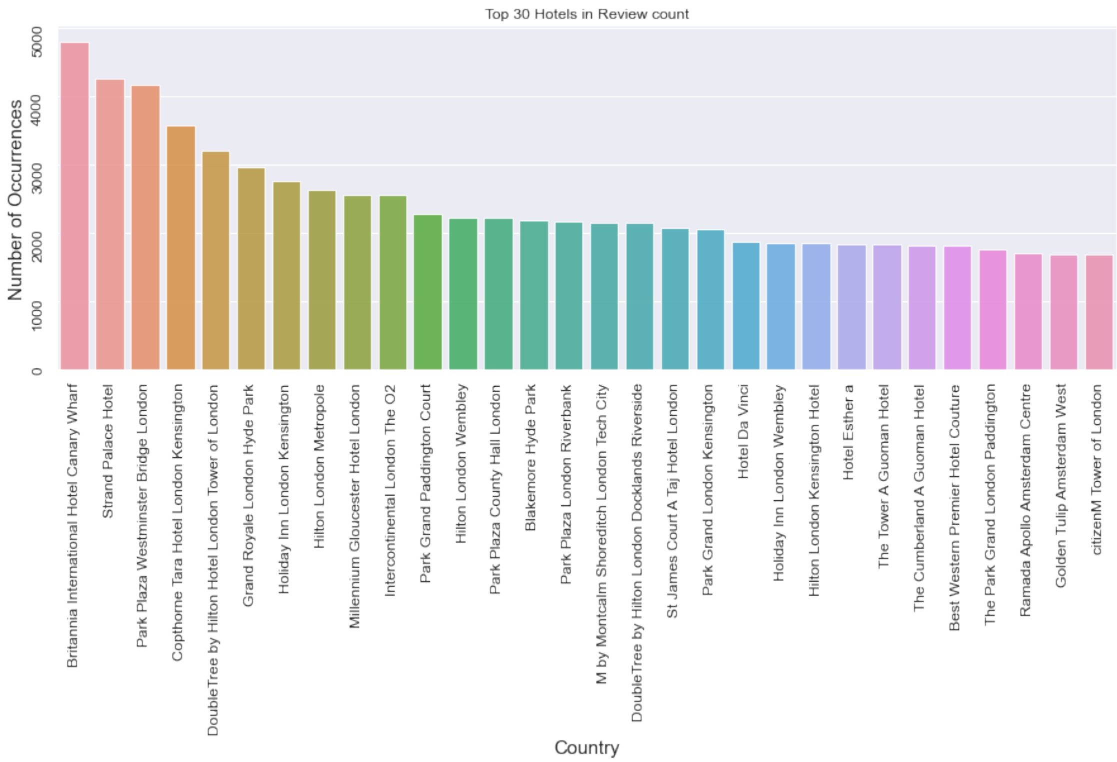


[14]:
```
Reviewer_Nationality  = data["Reviewer_Nationality"].value_counts()
Reviewer_Nationality = Reviewer_Nationality[:20,]
plt.figure(figsize=(15,5))
sns.barplot(Reviewer_Nationality.index, Reviewer_Nationality.values, alpha=0.9)
plt.title('Destribution of 20 top Reviewer Nationalities count')
plt.ylabel('Number of Occurrences', fontsize=15)
plt.xlabel('Country', fontsize=15)
plt.tick_params(labelsize=12, rotation=90)
plt.show()
```



From the above 2 graphs it seems that most of the reviews are for hotels is UK and most of the reviewers are from UK

[15]:
```
# We will look ar the top hotels and te scores fro them

Hotel_Name  = data["Hotel_Name"].value_counts()
Hotel_Name = Hotel_Name[:30,]
plt.figure(figsize=(15,5))
sns.barplot(Hotel_Name.index, Hotel_Name.values, alpha=0.9)
plt.title('Top 30 Hotels in Review count')
plt.ylabel('Number of Occurrences', fontsize=15)
plt.xlabel('Country', fontsize=15)
plt.tick_params(labelsize=12, rotation=90)
plt.show()
```



[16]:
```
# Looking at the average scors of the hotels across the dataset
data['Average_Score'].describe()
```

[16]:
```
count    515738.000000
mean          8.397487
std           0.548048
min           5.200000
25%           8.100000
50%           8.400000
75%           8.800000
max           9.800000
Name: Average_Score, dtype: float64
```

[17]:
```
dataplot = pd.DataFrame(data[['Hotel_Name',␣
 ↪'Reviewer_Nationality','Country','Average_Score','Reviewer_Score']])


Mean_Average_Score = data.Average_Score.groupby(data.Country).mean()
Mean_Reviewer_Score = data.Reviewer_Score.groupby(data.Country).mean()

print(Mean_Average_Score)
print(Mean_Reviewer_Score)
```

```
Country
Austria        8.558034
France         8.409053
Italy          8.426729
Netherlands    8.387085
Spain          8.522812
UK             8.340393
Name: Average_Score, dtype: float64
Country
Austria        8.545047
France         8.420081
Italy          8.346722
Netherlands    8.456311
Spain          8.554092
UK             8.324138
Name: Reviewer_Score, dtype: float64
```

[18]:

```
# Working to segregate from the Tags to understand the reviewers groups either␣
↪Bussines trip or Leasure trip
#First lets plot the word cloud fro the tags
tag = pd.Series(re.findall(r'[\']\s([\w\s]+)\s[\']',''.join(data.Tags))).
↪value_counts()
tag.head()
```

[18]:

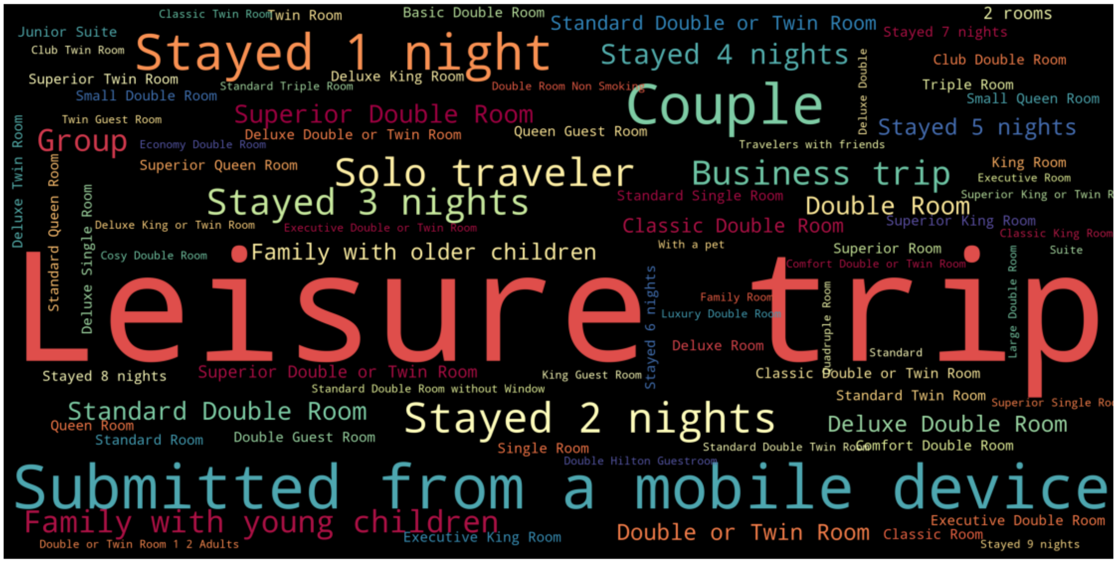
```
Leisure trip                       417778
Submitted from a mobile device     307640
Couple                             252294
Stayed 1 night                     193645
Stayed 2 nights                    133937
dtype: int64
```

Furthermore, the process that provided average scores help us create Wordclouds of the reviews that highlighted the key terms associated with the purpose of the visit of the hotels in positive and negative reviews. There were however issues encountered as some negative reviews contained outliers. Hence due to this analysis we will consider the score and number of words in the review to try to eliminate these outliers as much as possible not include any text that has words less than three words for negative reviews with score more than nine and positive reviews with cord count less than three.

[19]:

```
wordcloud = WordCloud(background_color='black', scale=10, max_font_size=70,␣
↪max_words=2000, colormap="Spectral").generate_from_frequencies(tag)
wordcloud.recolor(random_state=1)
plt.figure(1, (20,16))
plt.imshow(wordcloud)
plt.title("Groups of reviewers ")
plt.axis("off")
plt.show()
```

Groups of reviewers

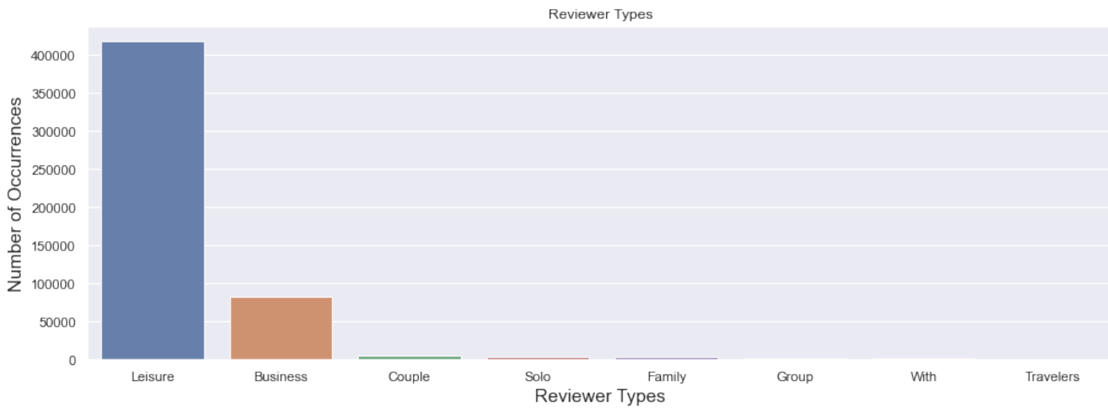


Based on the outcome of the wordcloud above, "Leisure Trip" incurred the most frequencies in terms of preference/purpose of trip, followed by "Submitted from a mobile device" and "Couple"

[20]:
```python
data['Trip_type'] = data.Tags.apply(lambda x: x.split(' ')[1])
print(data.Trip_type.unique())
Trip_type_count = data['Trip_type'].value_counts()
print(Trip_type_count)
```

```
['Leisure' 'Business' 'Couple' 'With' 'Solo' 'Group' 'Family' 'Travelers']
Leisure      416672
Business      82748
Couple         5808
Solo           4291
Family         3158
Group          1600
With           1405
Travelers        56
Name: Trip_type, dtype: int64
```

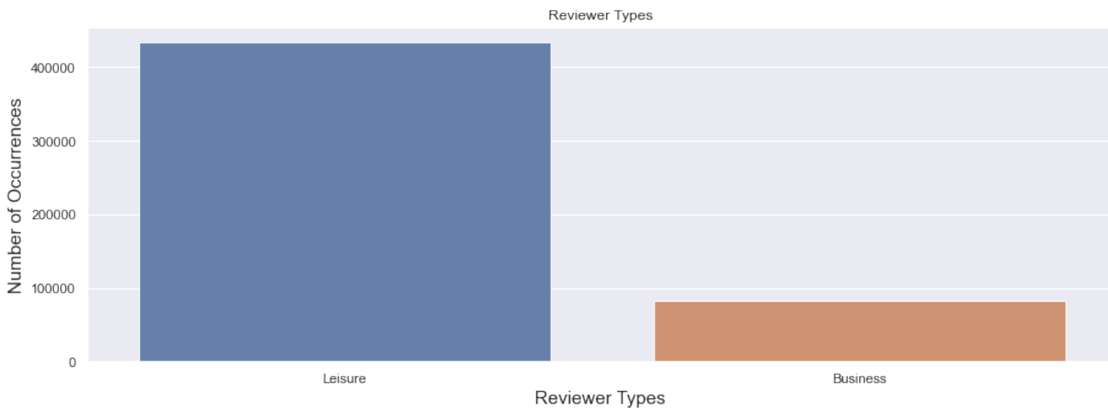
[21]:
```python
Trip_type  = data["Trip_type"].value_counts()
plt.figure(figsize=(15,5))
sns.barplot(Trip_type.index, Trip_type.values, alpha=0.9)
plt.title('Reviewer Types ')
plt.ylabel('Number of Occurrences', fontsize=15)
plt.xlabel('Reviewer Types ', fontsize=15)
plt.show()
```



We will consider any other group differen from Business is Leisure

[22]:
```
data.Trip_type = data.Trip_type.str.replace('Couple','Leisure')
data.Trip_type = data.Trip_type.str.replace('Solo','Leisure')
data.Trip_type = data.Trip_type.str.replace('Family','Leisure')
data.Trip_type = data.Trip_type.str.replace('Group','Leisure')
data.Trip_type = data.Trip_type.str.replace('With','Leisure')
data.Trip_type = data.Trip_type.str.replace('Travelers','Leisure')
```

[23]:
```
Trip_type  = data["Trip_type"].value_counts()
plt.figure(figsize=(15,5))
sns.barplot(Trip_type.index, Trip_type.values, alpha=0.9)
plt.title('Reviewer Types ')
plt.ylabel('Number of Occurrences', fontsize=15)
plt.xlabel('Reviewer Types ', fontsize=15)
plt.show()
```

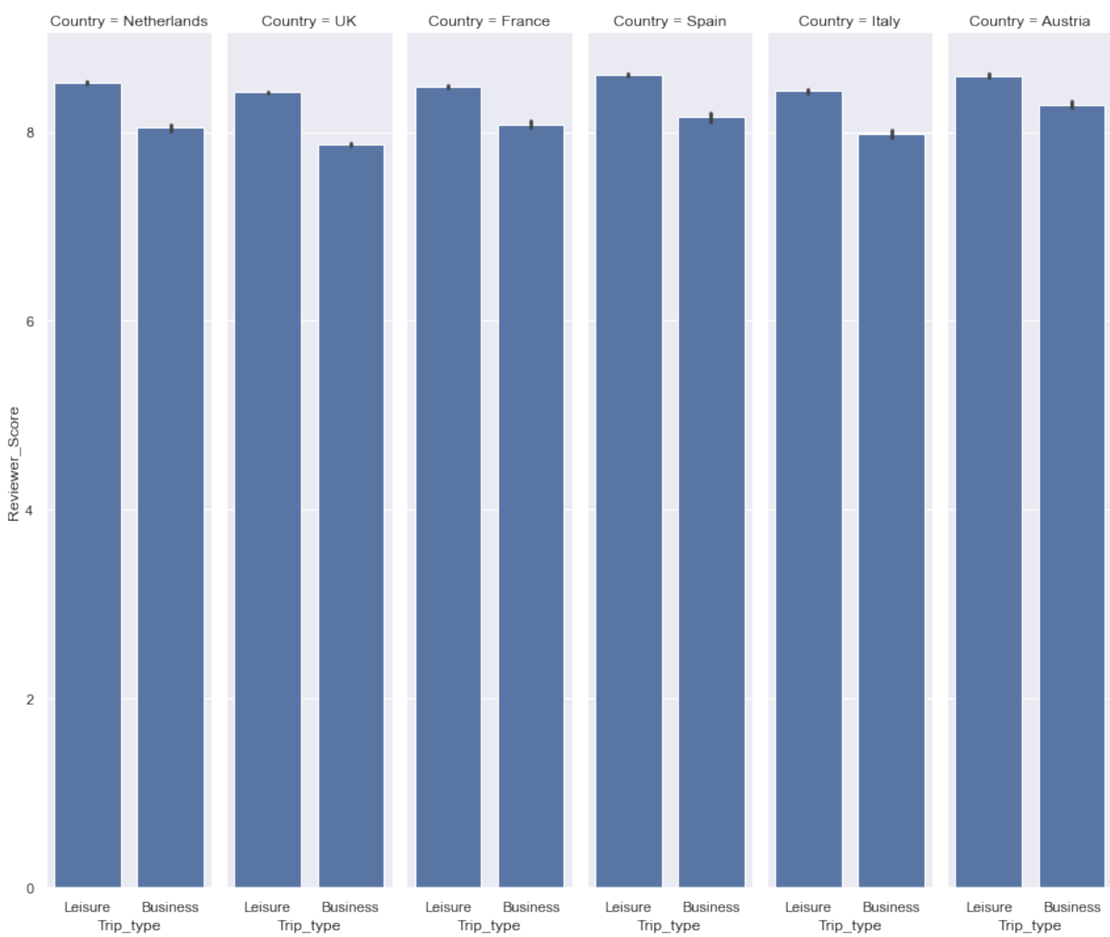


[24]:
```
Mean_Revie_typ_Score = data.Reviewer_Score.groupby(data.Trip_type).mean()
Mean_Avg_typ_Score = data.Average_Score.groupby(data.Trip_type).mean()
print(Mean_Revie_typ_Score)
print(Mean_Avg_typ_Score)
```

```
Trip_type
Business    7.972605
Leisure     8.475814
Name: Reviewer_Score, dtype: float64
Trip_type
Business    8.307659
Leisure     8.414654
Name: Average_Score, dtype: float64
```

[25]:
```
g = sns.FacetGrid(data, col="Country", height=10, aspect=.2)
g.map(sns.barplot, "Trip_type", "Reviewer_Score", order=["Leisure", "Business"])
```

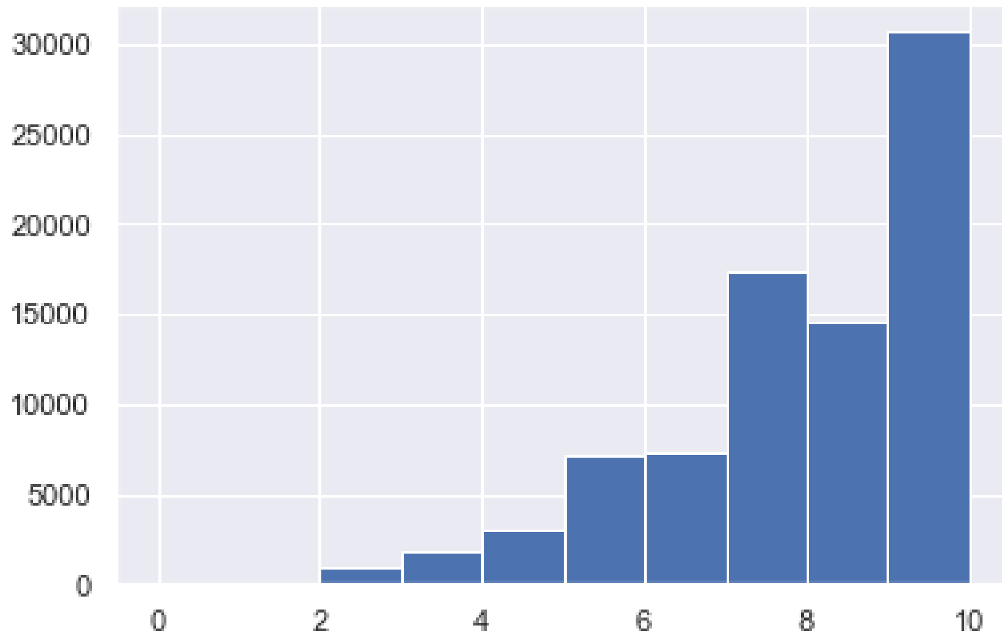
[25]: <seaborn.axisgrid.FacetGrid at 0x233763934c0>



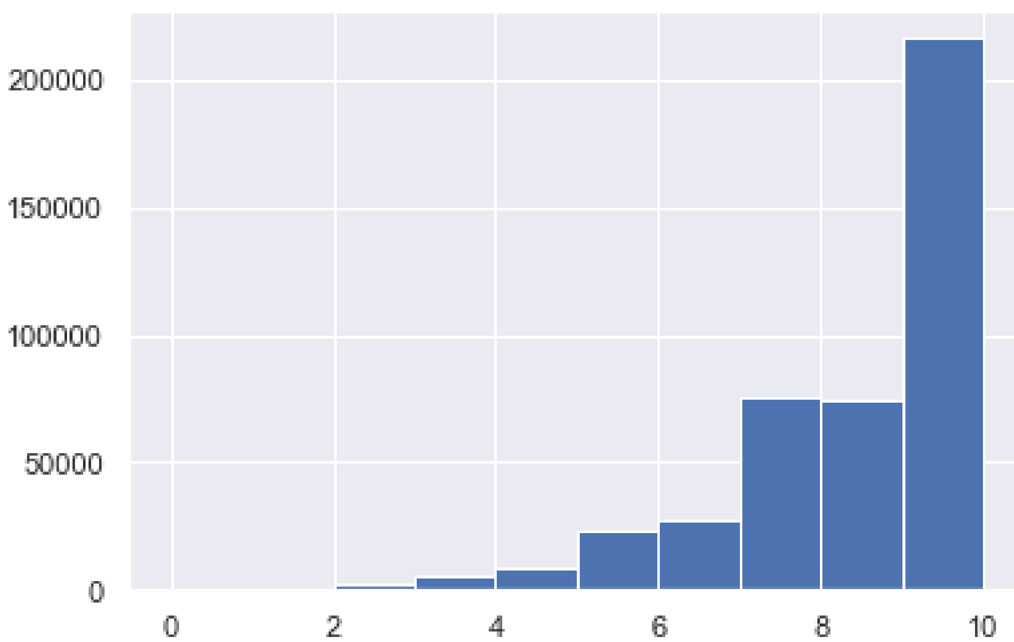
[26]:
```
sns.distplot(data['Reviewer_Score'], hist=True, kde=True,
             bins=int(10), color = 'red',
             hist_kws={'edgecolor':'black'},
             kde_kws={'linewidth': 3})
```

[26]: <AxesSubplot:xlabel='Reviewer_Score', ylabel='Density'>

[27]:
```
hist1 = plt.hist(data[data.Trip_type=='Business'].Reviewer_Score, bins = 10,␣
 ↪range=(0,10))
```




```
[28]: hist2 = plt.hist(data[data.Trip_type=='Leisure'].Reviewer_Score, bins=10,␣
      ↪range=(0,10))
```



According to the graph above, as the occurences of leisure trips increase, so do the reviewer score.

With the skewness the same as those of business trips, scores are likely to rise as more and more occurences incur

[29]:
```
Highest_Hotel_score = (data.Reviewer_Score.groupby(data.Hotel_Name).mean()).
    ↪sort_values(ascending = False)[:30]
print(Highest_Hotel_score)
```

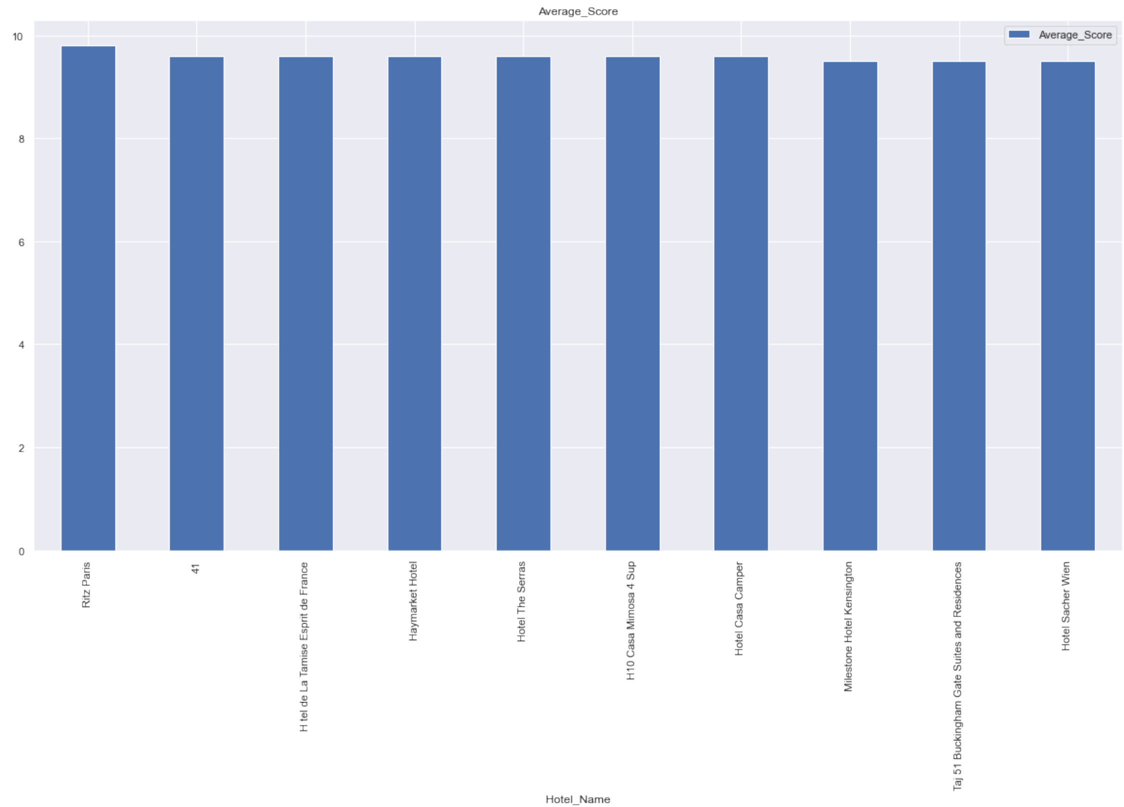
```
Hotel_Name
Ritz Paris                                   9.725000
Hotel Casa Camper                            9.718937
41                                           9.711650
H tel de La Tamise Esprit de France          9.688525
Le Narcisse Blanc Spa                        9.671930
H10 Casa Mimosa 4 Sup                        9.660345
Hotel Eiffel Blomet                          9.646667
Hotel The Serras                             9.623474
45 Park Lane Dorchester Collection           9.603571
The Soho Hotel                               9.597452
Haymarket Hotel                              9.590909
Hotel Sacher Wien                            9.589231
Covent Garden Hotel                          9.587838
Milestone Hotel Kensington                   9.572093
Hotel Plaza Athenee Paris                    9.566667
Catalonia Magdalenes                         9.561415
H tel Fabric                                 9.559223
Taj 51 Buckingham Gate Suites and Residences    9.554887
Hollmann Beletage Design Boutique            9.553922
H tel D Aubusson                             9.552041
Waldorf Astoria Amsterdam                    9.535211
Bulgari Hotel London                         9.529730
Egerton House                                9.519737
Hotel Am Stephansplatz                       9.518519
Lansbury Heritage Hotel                      9.517500
Palais Coburg Residenz                       9.512500
Hotel Monge                                  9.507826
Le 123 S bastopol Astotel                    9.506923
Hotel Sans Souci Wien                        9.506383
Boutiquehotel Das Tyrol                      9.497842
Name: Reviewer_Score, dtype: float64
```

[30]:
```
Lowest_Hotel_score = (data.Reviewer_Score.groupby(data['Hotel_Name']).mean()).
    ↪sort_values(ascending = True)[:30]
print(Lowest_Hotel_score)
```

```
Hotel_Name
Hotel Liberty                                5.121538
Kube Hotel Ice Bar                           5.852632
Villa Eugenie                                5.864516
```

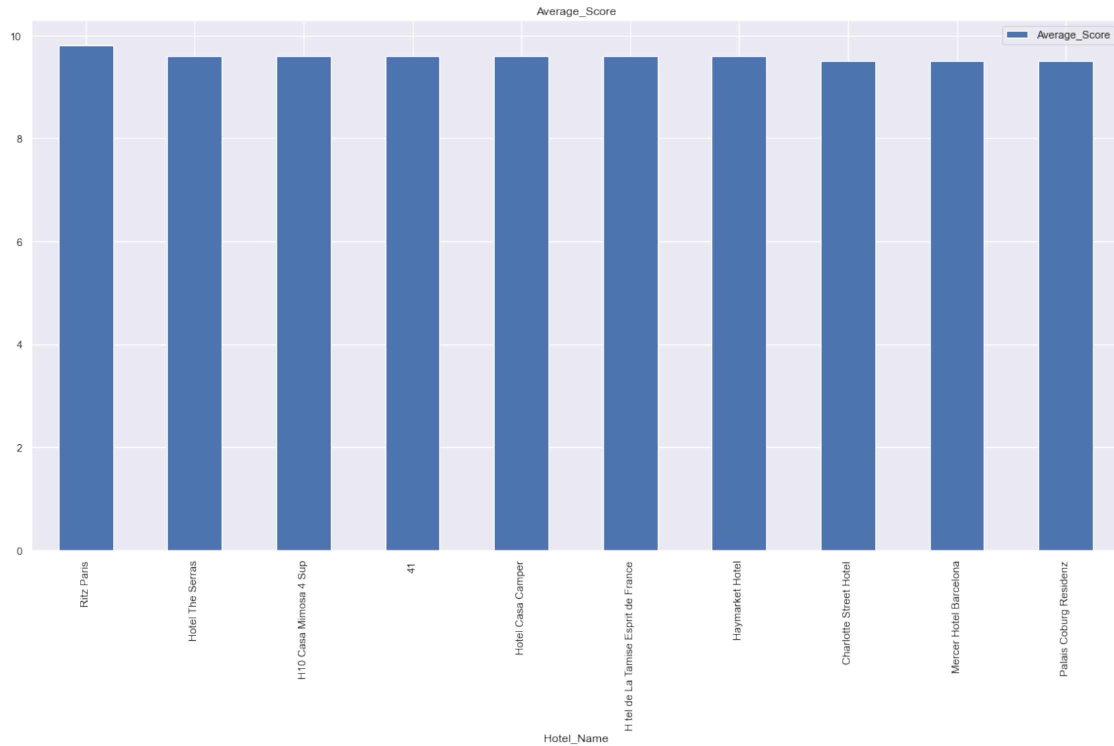
```
Savoy Hotel Amsterdam                              6.009465
Holiday Inn Paris Montparnasse Pasteur             6.329730
Best Western Maitrise Hotel Edgware Road           6.375000
Ibis Styles Milano Palmanova                       6.383333
Villa Lut ce Port Royal                            6.385106
Hotel Cavendish                                    6.442065
The Tophams Hotel                                  6.480000
Gran Hotel Barcino                                 6.520732
Commodore Hotel                                    6.554355
Hallmark Hotel London Chigwell Prince Regent       6.566955
Idea Hotel Milano San Siro                         6.580086
Gainsborough Hotel                                 6.676332
IH Hotels Milano Lorenteggio                       6.712270
Bloomsbury Palace Hotel                            6.736685
Henry VIII                                         6.785095
Eurohotel Diagonal Port                            6.809615
Hyatt Regency Paris Etoile                         6.824485
Britannia International Hotel Canary Wharf         6.826644
Mercure Paris Op ra Faubourg Montmartre            6.827273
BEST WESTERN Maitrise Hotel Maida Vale             6.884191
Park Lane Mews Hotel                               6.897256
Mercure Paris 19 Philharmonie La Villette          6.912963
Amarante Beau Manoir                               6.918868
London Elizabeth Hotel                             6.968644
Mokinba Hotels King                                6.979605
Ilunion Almirante                                  7.009091
St George Hotel                                    7.011475
Name: Reviewer_Score, dtype: float64
```

[31]:
```
Best_Les_Hotels= data[(data.Trip_type ==␣
↪'Leisure')][['Hotel_Name','Average_Score']].drop_duplicates()

Bes_htl_les = Best_Les_Hotels.groupby(Best_Les_Hotels['Hotel_Name']).mean().
↪sort_values(by = 'Average_Score' ,ascending = False)[:10]

Bes_htl_les = Bes_htl_les.plot.bar(rot=90, subplots=True, figsize=(20, 10))
```



Based on the chart above, it appears that these hotels are ranked higher on the score scale, highlighting their major worth and a luxurious visage amongst leisure trips

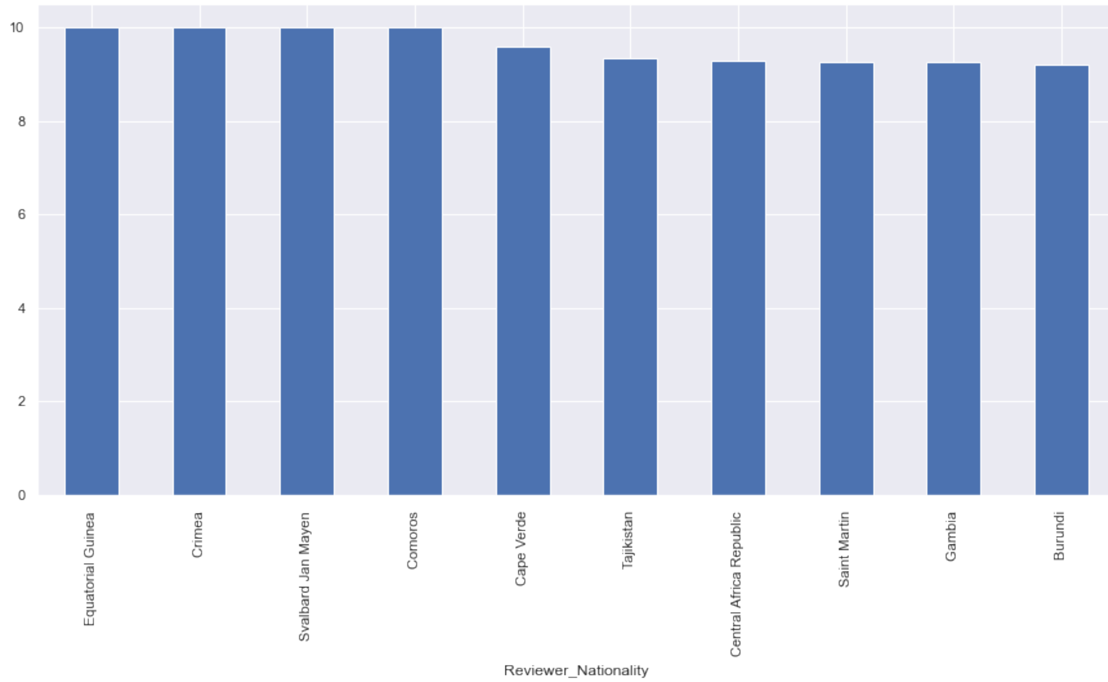
[32]:
```
Best_Bes_Hotels= data[(data.Trip_type ==␣
↪'Business')][['Hotel_Name','Average_Score']].drop_duplicates()

Bes_htl_bes = Best_Bes_Hotels.groupby(Best_Bes_Hotels['Hotel_Name']).mean().
↪sort_values(by = 'Average_Score' ,ascending = False)[:10]

Bes_htl_bes = Bes_htl_bes.plot.bar(rot=90, subplots=True, figsize=(20, 10))
```



Based on the chart above, it appears that these hotels are ranked higher on the score scale, highlighting their major worth and a luxurious visage amongst business trips

[33]:
```
fig, ax = plt.subplots(figsize=(15,7))
data.groupby(['Reviewer_Nationality']).mean()['Reviewer_Score'].
↪sort_values(ascending = False)[:10].plot.bar(ax=ax)
```

[33]: <AxesSubplot:xlabel='Reviewer_Nationality'>



Based on the chart above, it appears that top four nationalities of reviewers have been visting from countries in East Africa and Northern and Central Europe respectively

Segregation of the negtive and postive reviews

[34]:
```
#let's look back at the reviewer scores
data.Reviewer_Score.describe()
```

[34]:
```
count    515738.000000
mean          8.395077
std           1.637856
min           2.500000
25%           7.500000
50%           8.800000
75%           9.600000
max          10.000000
Name: Reviewer_Score, dtype: float64
```

[35]:
```
data.Review_Total_Positive_Word_Counts.describe()
```

[35]:
```
count    515738.000000
mean         17.776458
std          21.804185
min           0.000000
25%           5.000000
50%          11.000000
```

```
75%           22.000000
max          395.000000
Name: Review_Total_Positive_Word_Counts, dtype: float64
```

[36]:
```python
data.Review_Total_Negative_Word_Counts.describe()
```

[36]:
```
count    515738.000000
mean         18.539450
std          29.690831
min           0.000000
25%           2.000000
50%           9.000000
75%          23.000000
max         408.000000
Name: Review_Total_Negative_Word_Counts, dtype: float64
```

[37]:
```python
# lets Explore the reviews very few words
data.query('Review_Total_Negative_Word_Counts < 7').
↪head(50)[['Negative_Review', 'Review_Total_Negative_Word_Counts'␣
↪,'Positive_Review','Reviewer_Score' ]]
```

[37]:
```
                   Negative_Review  Review_Total_Negative_Word_Counts  \
1                      No Negative                                  0
10               Nothing all great                                  5
13                     No Negative                                  0
15                     No Negative                                  0
18                     No Negative                                  0
24                        Nothing                                   3
33                 Please see above                                 4
48                     No Negative                                  0
52              I loved everything                                  5
53                     No Negative                                  0
55                     No Negative                                  0
59                     No Negative                                  0
75                     No Negative                                  0
78                     No Negative                                  0
79                     No Negative                                  0
96                     No Negative                                  0
112                    No Negative                                  0
113                    No Negative                                  0
114                    No Negative                                  0
140                    No Negative                                  0
154                    No Negative                                  0
168                    No Negative                                  0
174                    No Negative                                  0
178                    No Negative                                  0
198                    No Negative                                  0
```

200	No Negative	0
204	No Negative	0
207	No Negative	0
211	No Negative	0
215	No Negative	0
220	No Negative	0
222	No Negative	0
223	No Negative	0
225	No Negative	0
228	n a	3
229	No Negative	0
231	No bad experiences	4
236	No Negative	0
242	No Negative	0
245	No Negative	0
250	Minimal space around the bed	6
251	No Negative	0
253	No Negative	0
255	No Negative	0
256	Very long check in process	6
258	Unusual room layout	4
263	Loud aircondition	3
265	No Negative	0
268	Nothing	3
269	No Negative	0

	Positive_Review	Reviewer_Score
1	No real complaints the hotel was great great …	7.5
10	Rooms were stunningly decorated and really sp…	10.0
13	This hotel is being renovated with great care…	9.2
15	This hotel is awesome I took it sincirely bec…	10.0
18	Public areas are lovely and the room was nice…	7.1
24	Lovely hotel with extremely comfortable huge …	9.6
33	The hotel is going through renovations and un…	6.7
48	The quality of the hotel was brilliant and ev…	10.0
52	The location in a quiet park with a great ter…	10.0
53	Beautiful setting in a lovely park room very …	10.0
55	The hotel is lovely and the staff were amazin…	10.0
59	Basically everything The style of the hotel i…	9.6
75	The whole hotel was very clean the staff were…	10.0
78	Hotel was really nice staff were very friendl…	9.2
79	We have stayed here a few times and always en…	9.2
96	We upgraded to a larger room Had the bath inf…	10.0
112	Architecturally the building is terrific you …	10.0
113	Breakfast was very nice and the staff were so…	9.6
114	This hotel was great value for money Only a f…	8.8
140	Beautiful gothic hotel Very good location Som…	10.0

```
154   Bar and restraint staff were great reception …           6.7
168   The staff were exceptionally friendly and att…          10.0
174   The hotel is amazing Beautiful interior We we…          10.0
178   Stayed in this hotel for 4 nights with my boy…          10.0
198   Staff were really friendly 24hour front desk …           9.2
200   Staff were very friendly on arrival we were u…           9.2
204   Good location next to the tram but is under g…          10.0
207   Hotel and rooms were beautiful My friend and …           9.2
211   This was our first time in Amsterdam and the …          10.0
215   It was a well kept hotel in a lovely area The…          10.0
220   I loved the hotel design and decoration Staff…           9.6
222   Got in late and hadn t eaten 24 hour pizza de…           9.6
223   We stayed at the hotel for 5 nights Hotel is …           9.6
225   The matress and pillows were exceptionally go…           9.6
228   This is a fantastic place to stay and felt li…           9.2
229   We loved our stay at hotel arena The hotel wa…          10.0
231   Amazing rooms with beautiful view of river Fo…           9.6
236   This is an older building with a wonderful re…          10.0
242   I loved the room There was so much space span…           9.6
245   Great restaurant bar very green area you dine…           9.6
250   Breakfast was very good but it wasn t include…           5.8
251   Super stylish hotel next to a lovely park Clo…           9.6
253                   Decor service fresh herb garden           10.0
255   Standard room is amazing location is adorable…          10.0
256                                  Impressive venue            7.9
258                    Beautiful location next to park           6.3
263   Room was spacious enough Bed was very comfort…           7.9
265                                  Room was awesome           10.0
268                                       everything           10.0
269   Amazing view to the park not so busy area Nic…          10.0
```

[38]:
```python
data.query('Review_Total_Positive_Word_Counts < 7').
 ↪head(50)[['Negative_Review', 'Review_Total_Positive_Word_Counts'␣
 ↪,'Positive_Review','Reviewer_Score' ]]
```

[38]:
```
                                        Negative_Review  \
8     Even though the pictures show very clean room…
11    6 30 AM started big noise workers loading woo…
12    The floor in my room was filfy dirty Very bas…
27    Careful they are still renovating the buildin…
32    Our bathroom had an urine order Shower was ve…
46    The hotel is under construction which was nev…
49    Service horrible Pillows super stiff and big …
51    When arriving I was told I had to pay 19 city…
60    The place is completely mismanaged The proper…
65    Not being told a hedkandi night was across fr…
87    Maintenance work on facade of hotel no advanc…
```

```
90     Even allowing for the hotel being under major…
92     In a terrible state with builders everywhere …
98     Got charged 50 for a birthday package when it…
100    Building work starting at 7am waking us up no…
107    No Limited A C in common areas Dangerous meta…
116    I wouldnt be able to recommend my grandparent…
121    The first room had steep steps to a loft bed …
124    The shower was useless and when it worked it …
134    Foyer was a mess Only place to relax was the …
146    We booked a 3 night stay in a suite On arriva…
159    the hotel was in a bad condition Not really c…
169    Nothing One Of The Receptionist she did a rac…
172    Hotel under sonstruction which we weren t awa…
175    The bathroom door was hanging off the light f…
187    The hotel is being renovated and an extensive…
189    the restaurant food was terrible my first roo…
194    No plug with washbasin Staff did not think th…
196    The toilet has glass walls no privacy The roo…
202    Renovation around the hotel sometimes can sta…
203    The room wasn t cleaned two days in a row I d…
209    Not given the room type we had booked and pre…
212    Building work going on at the hotel didn t kn…
232    floor tiles in bath room shower were slippery…
233    The bathroom needed to be deep cleaned Tiles …
244    service of the breakfast team plates were not…
246       Beds sucked Air conditioner too loud for use
247          The beds were a little small and very low
256                         Very long check in process
258                                Unusual room layout
259    Room design resulted in limited storage space…
265                                        No Negative
266            they robbed me blind of nearly 500 Euros
267    Room was very small and had a small staircase…
268                                           Nothing 
272                                  check in time 3pm
278                                        No Negative
280    I booked a twin room so thought there would b…
285                                        No Negative
287              The heating was a bit too low but eeeh

    Review_Total_Positive_Word_Counts                      Positive_Review  \
8                                   0                          No Positive
11                                  4                 Style location rooms
12                                  6                Comfy bed good location 
27                                  6     Great hotel original concept style
32                                  0                          No Positive
46                                  3                           Massive bed
```

49	4	clean and new
51	6	The location and views
60	5	The property is beautiful
65	6	Great trip staff very friendly
87	6	Pretty building interesting rooms
90	4	Large bed
92	5	Good location Cheap
98	0	No Positive
100	2	Location
107	6	Good water pressure in shower
116	5	food recommendations were brilliant
121	0	No Positive
124	6	Liked the staff The location
134	0	No Positive
146	0	No Positive
159	5	Bed was nice
169	0	No Positive
172	0	No Positive
175	6	Room service good quality staff
187	4	Polite staff
189	2	nothing
194	3	Breakfast Staff
196	3	The hall
202	0	No Positive
203	5	Good Wi fi
209	0	No Positive
212	5	Great location lovely room
232	6	staff good location suited us
233	4	Great historic buildig
244	4	location outside terras
246	6	Nice property and building
247	0	No Positive
256	3	Impressive venue
258	6	Beautiful location next to park
259	3	Excellent Breakfasts
265	4	Room was awesome
266	2	nothing
267	6	Bed was really comfortable
268	3	everything
272	4	location rooms staff
278	6	Comfy bed tasty hot chocolates
280	0	No Positive
285	4	Bed extra comfy
287	4	Everything Classy

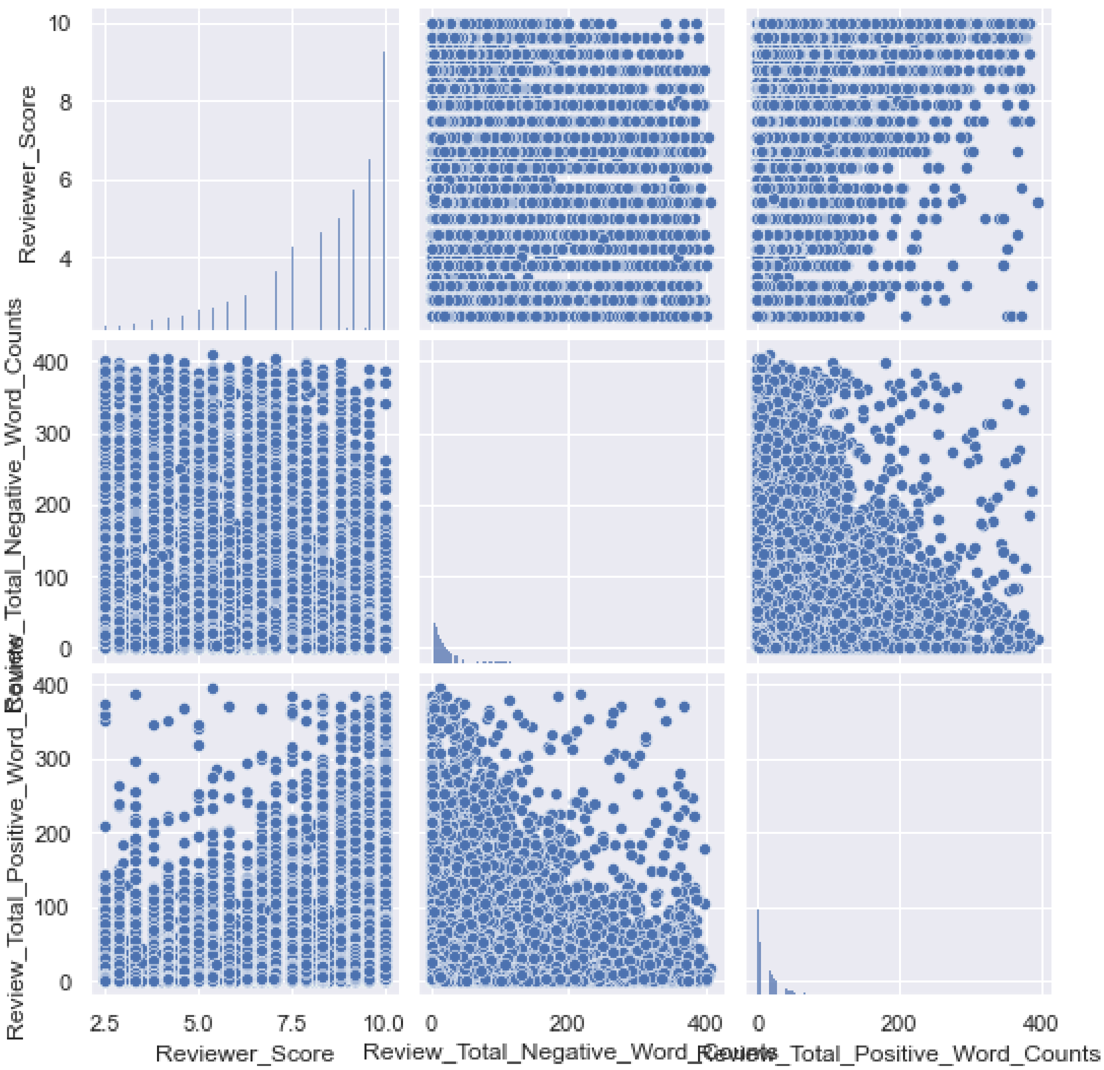
	Reviewer_Score
8	6.5

11	5.8
12	4.6
27	8.3
32	4.2
46	4.2
49	5.4
51	7.1
60	4.6
65	8.8
87	5.8
90	4.2
92	5.0
98	5.0
100	5.0
107	3.8
116	9.2
121	8.3
124	4.6
134	4.6
146	2.5
159	4.2
169	2.9
172	3.8
175	6.7
187	4.2
189	3.1
194	7.1
196	4.6
202	5.8
203	6.3
209	4.6
212	5.8
232	7.1
233	5.8
244	6.7
246	7.5
247	7.5
256	7.9
258	6.3
259	7.9
265	10.0
266	2.5
267	9.2
268	10.0
272	8.8
278	10.0
280	8.3

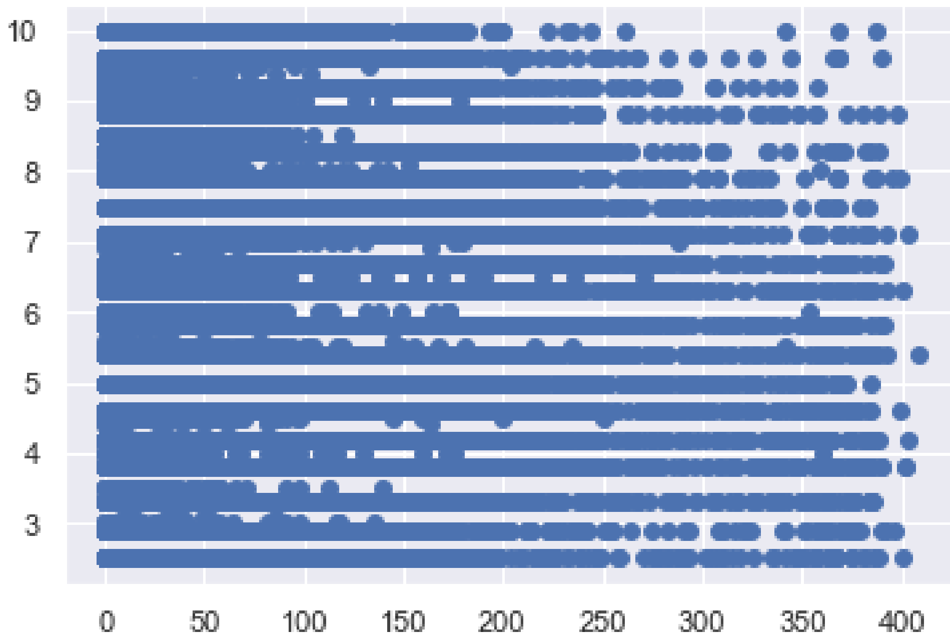
```
285            9.2
287            9.6
```

[39]:
```
#let's plot number of words against reveiw score
Wrds_cnt = pd.DataFrame(data, columns␣
 ↪=['Reviewer_Score','Review_Total_Negative_Word_Counts','Review_Total_Positive_Word_Counts'])
sns.pairplot(Wrds_cnt)
```

[39]: <seaborn.axisgrid.PairGrid at 0x23364dafdf0>



[40]:
```
plt.scatter(x = Wrds_cnt.Review_Total_Negative_Word_Counts, y = Wrds_cnt.
 ↪Reviewer_Score)
plt.show()
```



[41]:
```
plt.scatter(x = Wrds_cnt.Review_Total_Positive_Word_Counts, y = Wrds_cnt.
↪Reviewer_Score)
plt.show()
```

From the above graphs it seems that reviewers tend to write more for the negtive reviews even with the higer rating scores also from looking at the reviewes it seems that the terms 'No Negative' and 'No Postive' are stsem genrated wich means the user did not add any comment so we will filter these 2 terms and replace them with emty cell for the modeling

[42]:
```
data.Negative_Review = data.Negative_Review.str.replace('No Negative',' ')
data.Positive_Review = data.Positive_Review.str.replace('No Positive',' ')
```

[43]:
```
pos_out = data.query('Review_Total_Positive_Word_Counts <= 3 and␣
↪Reviewer_Score< 3')[['Negative_Review', 'Review_Total_Positive_Word_Counts'␣
↪,'Positive_Review','Reviewer_Score' ]]
wordcloud = WordCloud(background_color='white', scale=3, max_font_size=40,␣
↪max_words=25).generate_from_text(' '.join(list(pos_out['Positive_Review'])))
wordcloud.recolor(random_state=1)
plt.figure(1, (20,16))
plt.imshow(wordcloud)
plt.title("Wordcloud for positive review")
plt.axis("off")
plt.show()
```

The above Wordcloud show the reviews with less than 3 postive words with very low score and and the most used term is "Nothing" that is why we will try to filter such comments from the analysis so it will not have a an effect on the models

[44]:
```
neg_out = data.query('Review_Total_Negative_Word_Counts <= 3  and ␣
↪Reviewer_Score > 9')[['Negative_Review', 'Review_Total_Negative_Word_Counts'␣
↪,'Positive_Review','Reviewer_Score' ]]
```

```
wordcloud = WordCloud(background_color='white', scale=3, max_font_size=40,␣
↪max_words=25).generate_from_text(' '.join(list(neg_out['Negative_Review'])))
wordcloud.recolor(random_state=1)
plt.figure(1, (20,16))
plt.imshow(wordcloud)
plt.title("Wordcloud for positive review")
plt.axis("off")
plt.show()
```

The above Wordcloud show the negative commnets we number of words less than 3 with very high score and and the most used term is "Nothing" that is why we will try to filter such comments from the analysis so it will not have a an effect on the models

[45]:
```
print(pos_out.info())
print(pos_out. Positive_Review .unique())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2194 entries, 146 to 515735
Data columns (total 4 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   Negative_Review                     2194 non-null   object
 1   Review_Total_Positive_Word_Counts   2194 non-null   int64
 2   Positive_Review                     2194 non-null   object
 3   Reviewer_Score                      2194 non-null   float64
dtypes: float64(1), int64(1), object(2)
memory usage: 85.7+ KB
None
[' ' ' nothing' ' Nothing' ' London' ' Nothing ' ' NOTHING' ' None'
```

```
 ' None ' ' EVERYHTING ' ' Not much' ' free wifi' ' Rubbish'
 ' horrible staff' ' Location' ' nothing ' ' the shower' ' Nothing really'
 ' Not ng' ' Wedding ceromony' ' No thing' ' The lobby'
 ' Absolutely nothing' ' the location' ' The location' ' ZERO'
 ' good breakfast' ' Nice location' ' Good breakfast' ' Fa' ' not things'
 ' Nothing Th' ' Nothink' ' Lousy hotel' ' Location only'
 ' Not recommended' ' Was cleanish' ' Location ' ' Horrible'
 ' smoking rooms' ' awful staff' ' Very bad' ' Bed comfy' ' Nothing good'
 ' Nada ' ' Nothings ' ' All excellent' ' The lifts' ' X' ' Place'
 ' The locality' ' Breakfast' ' Leaving' ' NO HEAT' ' location' ' The AC'
 ' breakfast' ' Not all' ' Not thing' ' lobby' ' Food' ' Nice shower'
 ' Rubbish hotel' ' Almost nothing' ' N t' ' Diddly Squats'
 ' quiet neighbourhood' ' The breakfast' ' Very little' ' Notjing'
 ' Dirty rooms' ' The room' ' yes' ' G' ' Good' ' location ok'
 ' good location' ' Poor' ' no thing' ' Poor service' ' Comfy bed'
 ' Good location' ' not much' ' The bed' ' Good neighborhood'
 ' Bed comfortable' ' EVERYTHING' ' WiFi' ' View' ' Breakfast only'
 ' Nathing' ' The gym' ' No' ' everything ok' ' Bed' ' NOTHING '
 ' the concierge' ' Non ' ' Actually leaving' ' Location wifi' ' Leaving '
 ' The food' ' Shower' ' Nil' ' Excellent breakfast' ' Sights'
 ' Central location' ' Location good' ' Actually nothing'
 ' Excellent room' ' Everything Bad' ' Restaurant' ' Nice carpet'
 ' The ending' ' NA' ' Nithing' ' The restaurant' ' Everything'
 ' Rooftop deck' ' Only breakfast' ' Very noisy' ' The parking'
 ' Friendly staff' ' Nice views' ' Nowhere' ' The Location'
 ' The bathroom' ' Modern design' ' Not Applicable' ' Comfortable beds'
 ' Broke bed' ' Didnt like' ' Great breakfast' ' Robbed' ' Nothing like'
 ' Absolutely appalling' ' Just location' ' Bad experience' ' Nil '
 ' Nothings' ' nothing intersting' ' Nulla' ' Nithibg' ' Nothing Rubbish'
 ' Overpriced' ' Nice bed' ' Beautiful city' ' nothing actually'
 ' great hotel' ' Needs updating' ' Hot water' ' Was quiet' ' Bright room'
 ' no' ' NONE' ' t' ' All' ' Anything' ' Excellent' ' Location crearness'
 ' Only location' ' Vary bad' ' not good' ' Really Nithing'
 ' Location amddv' ' Not' ' Excellent hotel' ' Spa facilities'
 ' The worst' ' Design' ' Nada horrible' ' Absolutely nil' ' NOthing'
 ' Nohing' ' Horrible hotel' ' TERRIBLE' ' Nothng' ' Friendly place'
 ' Nice lobby' ' non' ' friendly service' ' none ' ' Design '
 ' Unfortunately nothing' ' Terribile' ' Free Parking' ' apperetivo'
 ' New hotel' ' Free wifi' ' Position' ' position' ' Very poor' ' none'
 ' Horrible service' ' Shower bathroom' ' EVERYTHING ' ' System sucks'
 ' nothing really' ' Nothing honestly' ' Gym' ' The reception'
 ' Reception ']
```

[46]:
```
neg_out.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 126375 entries, 13 to 515732
Data columns (total 4 columns):
```

```
 #   Column                            Non-Null Count   Dtype
---  ------                            --------------   -----
 0   Negative_Review                   126375 non-null  object
 1   Review_Total_Negative_Word_Counts  126375 non-null  int64
 2   Positive_Review                   126375 non-null  object
 3   Reviewer_Score                    126375 non-null  float64
dtypes: float64(1), int64(1), object(2)
memory usage: 4.8+ MB
```

by exploring the reviewes it seems we will face some issues with the negtive and postive reviews as some Negative reviews are considered outlier as 'Nothing' 'everything was just righ' or 'I loved everything' as negtive reviews... Due to this analysis we will consider the score and number of words in the review to try to eleminate these outliers as much as possible not include any text that has words less than 3 words for negative reviews with score more than 9 and postive reviews with cord count less than 3

[47]:
```
# We will creat a a new dataframe for the text analysis
Rev_text = data.query('Review_Total_Positive_Word_Counts >= 3 and␣
 ↪Reviewer_Score>= 2.8')[['Positive_Review' , 'Reviewer_Score']]
Rev_text['Postive_text']=1
Rev_text.columns= ['Review_Text','Reviewer_Score', 'Postive_text']
Rev_text.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 457814 entries, 0 to 515737
Data columns (total 3 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   Review_Text     457814 non-null  object
 1   Reviewer_Score  457814 non-null  float64
 2   Postive_text    457814 non-null  int64
dtypes: float64(1), int64(1), object(1)
memory usage: 14.0+ MB
```

[48]:
```
Rev_text.head(50)
```

[48]:
```
                                          Review_Text  Reviewer_Score  \
0    Only the park outside of the hotel was beauti…             2.9
1    No real complaints the hotel was great great …             7.5
2    Location was good and staff were ok It is cut…             7.1
3    Great location in nice surroundings the bar a…             3.8
4     Amazing location and building Romantic setting              6.7
5    Good restaurant with modern design great chil…             6.7
6    The room is spacious and bright The hotel is …             4.6
7    Good location Set in a lovely park friendly s…            10.0
9    The room was big enough and the bed is good T…             7.9
10   Rooms were stunningly decorated and really sp…            10.0
11                             Style location rooms              5.8
```

12	Comfy bed good location	4.6
13	This hotel is being renovated with great care…	9.2
14	It was very good very historic building that …	8.8
15	This hotel is awesome I took it sincirely bec…	10.0
16	Great onsite cafe Amazing building Park locat…	6.3
17	We loved the location of this hotel The fact …	7.5
18	Public areas are lovely and the room was nice…	7.1
19	I liked the hotels history And for such an en…	7.5
20	Friendly staff OostPark a few yards away Good…	6.3
21	The breakfast was the only positive element o…	3.8
22	The location is good You need 15min to 20min …	5.4
23	Bed was extremely comfy and the staff where w…	9.6
24	Lovely hotel with extremely comfortable huge …	9.6
25	Great location in the park near museums and r…	8.3
26	The Hotel itself is in a lovely location a 5m…	9.6
27	Great hotel original concept style	8.3
28	The hotel itself is beautiful restaurant is v…	8.3
29	The hotel is located in a beautiful old monas…	9.2
30	The staff were so friendly and helpful plus t…	9.2
31	Friendly staff The bar restaurant area is lov…	7.1
33	The hotel is going through renovations and un…	6.7
34	The room was very big spacious The bath tub w…	7.9
35	Very nice hotel manager he upgraded us becaus…	8.3
36	the building meeting rooms modern style of my…	7.1
37	Very nice hotel located in a park Ca 30 minut…	8.8
38	The location was amazing the room was fantast…	8.8
39	Location on the park with easy access to tram…	6.3
40	The hotel is nicely localted directly within …	7.5
41	Nice restaurant although felt breakfast was r…	6.7
42	Love the design of the renovated product The …	2.9
43	Staff were amazing very very friendly and pro…	9.6
44	The brunch to purchase in the morning was good	3.3
45	The location of the hotel is super opening ou…	7.9
46	Massive bed	4.2
47	The hotel looks really impressive Set in a be…	8.3
48	The quality of the hotel was brilliant and ev…	10.0
49	clean and new	5.4
50	My room was upgraded because they are doing r…	9.6
51	The location and views	7.1

	Postive_text
0	1
1	1
2	1
3	1
4	1
5	1

```
6                    1
7                    1
9                    1
10                   1
11                   1
12                   1
13                   1
14                   1
15                   1
16                   1
17                   1
18                   1
19                   1
20                   1
21                   1
22                   1
23                   1
24                   1
25                   1
26                   1
27                   1
28                   1
29                   1
30                   1
31                   1
33                   1
34                   1
35                   1
36                   1
37                   1
38                   1
39                   1
40                   1
41                   1
42                   1
43                   1
44                   1
45                   1
46                   1
47                   1
48                   1
49                   1
50                   1
51                   1
```

[49]:
```
Rev_neg_text = data.query('Review_Total_Negative_Word_Counts >= 4 and␣
↪Reviewer_Score <= 9.8')[['Negative_Review','Reviewer_Score' ]]
```

```
Rev_neg_text['Postive_text']=0
Rev_neg_text.columns= ['Review_Text','Reviewer_Score', 'Postive_text']
Rev_neg_text.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 302216 entries, 0 to 515737
Data columns (total 3 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   Review_Text     302216 non-null  object 
 1   Reviewer_Score  302216 non-null  float64
 2   Postive_text    302216 non-null  int64  
dtypes: float64(1), int64(1), object(1)
memory usage: 9.2+ MB
```

[50]:
```
Rev_neg_text.head(50)
```

[50]:
```
                                          Review_Text  Reviewer_Score  \
0   I am so angry that i made this post available…             2.9   
2   Rooms are nice but for elderly a bit difficul…             7.1   
3   My room was dirty and I was afraid to walk ba…             3.8   
4   You When I booked with your company on line y…             6.7   
5   Backyard of the hotel is total mess shouldn t…             6.7   
6   Cleaner did not change our sheet and duvet ev…             4.6   
8   Even though the pictures show very clean room…             6.5   
9   The aircondition makes so much noise and its …             7.9   
11  6 30 AM started big noise workers loading woo…             5.8   
12  The floor in my room was filfy dirty Very bas…             4.6   
14  The staff in the restaurant could of been mor…             8.8   
16  Very steep steps in room up to the bed not sa…             6.3   
17  We did not like the fact that breakfast was n…             7.5   
19  We had issues with our electronic key everyda…             7.5   
20  Bed was on upper level with a narrow twist st…             6.3   
21  Our room was an overrated disaster room 231 d…             3.8   
22  Sadly I cannot say that the rooms are clean e…             5.4   
23  Transportation was a bit of a pain but on rou…             9.6   
25  The bathroom in our room was a black glass bo…             8.3   
26  Nothing at all to do with the Hotel of course…             9.6   
27  Careful they are still renovating the buildin…             8.3   
28  We had 2 different rooms here and both were d…             8.3   
29  There is an ongoing construction enlarging th…             9.2   
30                      Little bit on the pricey side             9.2   
31  Extensive restorations works going on We had …             7.1   
32  Our bathroom had an urine order Shower was ve…             4.2   
33                                   Please see above             6.7   
34  The rooms were cold Although nice the room de…             7.9   
35  Construction on site but not mentioned on boo…             8.3   
36  not cleaned well lady pushing to pay during m…             7.1   
```

37	The glass wall separating the bathroom and th…	8.8
38	the only thing that would of been better is i…	8.8
39	Staff a few were friendly and willing enough …	6.3
40	Several parts of the building outside are und…	7.5
41	Hotel undergoing the building of a new wing n…	6.7
42	Hotel is going through a major construction r…	2.9
43	Water pressure in my shower was no existent F…	9.6
44	The service was awful They refused to take ow…	3.3
45	Bathroom lighting could have been brighter Th…	7.9
46	The hotel is under construction which was nev…	4.2
47	The hotel is a little out of the main town bu…	8.3
49	Service horrible Pillows super stiff and big …	5.4
50	The bar was shut when I got back at midnight …	9.6
51	When arriving I was told I had to pay 19 city…	7.1
54	when you take a shower the whole floor is in …	7.9
56	there was construction work going on in the h…	7.1
57	Some building work was being carried out duri…	7.9
58	The hotel is under renovation so even though …	7.9
60	The place is completely mismanaged The proper…	4.6
61	The bathroom if someone took shower because i…	7.1

	Postive_text
0	0
2	0
3	0
4	0
5	0
6	0
8	0
9	0
11	0
12	0
14	0
16	0
17	0
19	0
20	0
21	0
22	0
23	0
25	0
26	0
27	0
28	0
29	0
30	0
31	0

```
32              0
33              0
34              0
35              0
36              0
37              0
38              0
39              0
40              0
41              0
42              0
43              0
44              0
45              0
46              0
47              0
49              0
50              0
51              0
54              0
56              0
57              0
58              0
60              0
61              0
```

[51]:
```
df_rev = Rev_text.append(Rev_neg_text)
```

[52]:
```
df_rev.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 760030 entries, 0 to 515737
Data columns (total 3 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   Review_Text      760030 non-null  object
 1   Reviewer_Score   760030 non-null  float64
 2   Postive_text     760030 non-null  int64
dtypes: float64(1), int64(1), object(1)
memory usage: 23.2+ MB
```

0.1.8 Text Cleaning

Natural Language Processing - Tokenize the reviews and build a bag-of-words model

[53]:
```
def get_wordnet_pos(pos_tag):
    if pos_tag.startswith('J'):
        return wordnet.ADJ
    elif pos_tag.startswith('V'):
```

```
        return wordnet.VERB
    elif pos_tag.startswith('N'):
        return wordnet.NOUN
    elif pos_tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN
stop = set(stopwords.words('english'))

def clean_text(text):
    # lower text.
    text = text.lower()

    # tokenize text and remove puncutation.
    text = [word.strip(string.punctuation) for word in text.split(" ")]

    # remove words that contain numbers.
    text = [word for word in text if not any(c.isdigit() for c in word)]

    # remove stop words.
    text = [x for x in text if x not in stop]

    # remove empty tokens.
    text = [t for t in text if len(t) > 0]

    # pos tag text.
    pos_tags = pos_tag(text)

    # lemmatize text.
    text = [WordNetLemmatizer().lemmatize(t[0], get_wordnet_pos(t[1])) for t in␣
 ↪pos_tags]

    # remove words with only one letter.
    text = [t for t in text if len(t) > 1]

    # join all.
    text = " ".join(text)
    return(text)

df_rev['Clean_Text'] = df_rev['Review_Text'].apply(clean_text)
df_rev.to_csv('Cleaned_Text.csv', index=False)
```

[54]:
```
df_rev.head(20)
```

[54]:
```
                                         Review_Text  Reviewer_Score  \
0   Only the park outside of the hotel was beauti…             2.9
1   No real complaints the hotel was great great …             7.5
```

```
2    Location was good and staff were ok It is cut…            7.1
3    Great location in nice surroundings the bar a…            3.8
4     Amazing location and building Romantic setting             6.7
5    Good restaurant with modern design great chil…            6.7
6    The room is spacious and bright The hotel is …            4.6
7    Good location Set in a lovely park friendly s…           10.0
9    The room was big enough and the bed is good T…            7.9
10   Rooms were stunningly decorated and really sp…           10.0
11                              Style location rooms             5.8
12                          Comfy bed good location             4.6
13   This hotel is being renovated with great care…            9.2
14   It was very good very historic building that …            8.8
15   This hotel is awesome I took it sincerely bec…           10.0
16   Great onsite cafe Amazing building Park locat…            6.3
17   We loved the location of this hotel The fact …            7.5
18   Public areas are lovely and the room was nice…            7.1
19   I liked the hotels history And for such an en…            7.5
20   Friendly staff OostPark a few yards away Good…            6.3

    Postive_text                                         Clean_Text
0              1                     park outside hotel beautiful
1              1  real complaint hotel great great location surr…
2              1  location good staff ok cute hotel breakfast ra…
3              1  great location nice surroundings bar restauran…
4              1           amaze location building romantic setting
5              1  good restaurant modern design great chill plac…
6              1  room spacious bright hotel locate quiet beauti…
7              1  good location set lovely park friendly staff f…
9              1  room big enough bed good breakfast food servic…
10             1  room stunningly decorate really spacious top b…
11             1                              style location room
12             1                          comfy bed good location
13             1  hotel renovate great care appreciation unique …
14             1                     good historic building choose
15             1  hotel awesome take sincerely bit cheap structu…
16             1  great onsite cafe amaze building park location…
17             1  love location hotel fact set park away busy ce…
18             1  public area lovely room nice window broken dra…
19             1  liked hotel history enormous hotel imagine man…
20             1  friendly staff oostpark yard away good contine…
```

[55]:
```
df_rev.isna().any()
```

[55]:
```
Review_Text        False
Reviewer_Score     False
Postive_text       False
Clean_Text         False
```

```
dtype: bool
```

0.1.9 Feature engineering and Sentiment analysis

[56]:
```python
sid_df =  df_rev.sample(frac =.03)

sid_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22801 entries, 483748 to 295564
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Review_Text     22801 non-null  object
 1   Reviewer_Score  22801 non-null  float64
 2   Postive_text    22801 non-null  int64
 3   Clean_Text      22801 non-null  object
dtypes: float64(1), int64(1), object(2)
memory usage: 890.7+ KB
```

[57]:
```python
sid_df.head(20)
```

[57]:
```
                                              Review_Text  Reviewer_Score  \
483748  Fabulous find Hotel emailed and asked what pi…            10.0
368508                                            The bed             5.0
284096                      Very comfortable bed amazing room            10.0
48845   both my wife and I got lost so we went back t…             7.9
504131  Comfortable room didn t use any of the other …             7.9
305050  Staff were incredibly helpful and informative…             8.8
242152                                           Location             7.1
327969  If you would like to find perfect location in…             8.3
355523  The room size and location of the hotel was g…             7.9
507481  This hotel has a good Location and outstandin…             9.2
471727    very nice hotel best location very clean rooms            10.0
68541   The room was run down with dirty and loose wa…             4.6
230930  Nice room but had no bedside lighting Hairdry…             7.5
477300                               Everything was exelent            10.0
345815  Thought the staff were great very helpful and…             7.5
333038                                  Location was great             6.7
377963                              Everything was perfect            10.0
275729  Had to pay for a kettle tea coffee Although o…             9.2
459528  Nice view Clean room Friendly staff Good food…             4.2
382128                   Staff were fantastic on front desk             8.8

        Postive_text                                        Clean_Text
483748             1  fabulous find hotel email ask pillow drink etc…
368508             1                                               bed
284096             1                        comfortable bed amazing room
```

```
48845               1  wife get lose go back hotel later stipulate ti…
504131              1  comfortable room use hotel facility overnight …
305050              1  staff incredibly helpful informative food rest…
242152              1                                          location
327969              1  would like find perfect location barcelona hot…
355523              1  room size location hotel good staff also reall…
507481              1  hotel good location outstanding view westminst…
471727              1                nice hotel best location clean room
68541               0  room run dirty loose wallpaper one tea bag one…
230930              0  nice room bedside light hairdryer drawer oppos…
477300              1                                everything exelent
345815              1  thought staff great helpful attentive need whe…
333038              1                                    location great
377963              1                                everything perfect
275729              0  pay kettle tea coffee although cost big inconv…
459528              1  nice view clean room friendly staff good food …
382128              1                         staff fantastic front desk
```

0.1.10 VADER Sentiment Analysis.

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media, and works well on texts from other domains.

Why is sentiment analysis so important? Businesses today are heavily dependent on data. Majority of this data however, is unstructured text coming from sources like emails, chats, social media, surveys, articles, and documents. The micro-blogging content coming from Twitter and Facebook poses serious challenges, not only because of the amount of data involved, but also because of the kind of language used in them to express sentiments, i.e., short forms, memes and emoticons. Sifting through huge volumes of this text data is difficult as well as time-consuming. Also, it requires a great deal of expertise and resources to analyze all of that. Not an easy task, in short. Sentiment Analysis is also useful for practitioners and researchers, especially in fields like sociology, marketing, advertising, psychology, economics, and political science, which rely a lot on human-computer interaction data. Sentiment Analysis enables companies to make sense out of data by being able to automate this entire process! Thus they are able to elicit vital insights from a vast unstructured dataset without having to manually indulge with it.

[58]:
```
sid = SentimentIntensityAnalyzer()
sid_df['Neg'] = 0.0
sid_df['Neu'] = 0.0
sid_df['Pos'] = 0.0
sid_df['Comp'] = 0.0

for index, row in sid_df.iterrows():
    result = sid.polarity_scores(row['Clean_Text'])
    sid_df.at[index,'Neg'] = result['neg']
    sid_df.at[index,'Neu'] = result['neu']
    sid_df.at[index,'Pos'] = result['pos']
```

```
    sid_df.at[index,'Comp'] = result['compound']
```

[59]:

```
sid_df.head(20)
```

[59]:

```
                                               Review_Text  Reviewer_Score  \
483748   Fabulous find Hotel emailed and asked what pi…           10.0
368508                                              The bed             5.0
284096                   Very comfortable bed amazing room           10.0
48845    both my wife and I got lost so we went back t…           7.9
504131   Comfortable room didn t use any of the other …           7.9
305050   Staff were incredibly helpful and informative…           8.8
242152                                             Location            7.1
327969   If you would like to find perfect location in…           8.3
355523   The room size and location of the hotel was g…           7.9
507481   This hotel has a good Location and outstandin…           9.2
471727     very nice hotel best location very clean rooms          10.0
68541    The room was run down with dirty and loose wa…           4.6
230930   Nice room but had no bedside lighting Hairdry…           7.5
477300                              Everything was exelent          10.0
345815   Thought the staff were great very helpful and…           7.5
333038                                   Location was great            6.7
377963                              Everything was perfect          10.0
275729   Had to pay for a kettle tea coffee Although o…           9.2
459528   Nice view Clean room Friendly staff Good food…           4.2
382128                     Staff were fantastic on front desk            8.8

        Postive_text                                         Clean_Text  \
483748             1   fabulous find hotel email ask pillow drink etc…
368508             1                                                bed
284096             1                           comfortable bed amazing room
48845              1   wife get lose go back hotel later stipulate ti…
504131             1   comfortable room use hotel facility overnight …
305050             1   staff incredibly helpful informative food rest…
242152             1                                           location
327969             1   would like find perfect location barcelona hot…
355523             1   room size location hotel good staff also reall…
507481             1   hotel good location outstanding view westminst…
471727             1                  nice hotel best location clean room
68541              0   room run dirty loose wallpaper one tea bag one…
230930             0   nice room bedside light hairdryer drawer oppos…
477300             1                                   everything exelent
345815             1   thought staff great helpful attentive need whe…
333038             1                                       location great
377963             1                                   everything perfect
275729             0   pay kettle tea coffee although cost big inconv…
459528             1   nice view clean room friendly staff good food …
382128             1                             staff fantastic front desk
```

```
          Neg    Neu    Pos    Comp
483748  0.000  0.625  0.375  0.9499
368508  0.000  1.000  0.000  0.0000
284096  0.000  0.220  0.780  0.7964
48845   0.139  0.619  0.242  0.6361
504131  0.000  0.708  0.292  0.5106
305050  0.000  0.424  0.576  0.7778
242152  0.000  1.000  0.000  0.0000
327969  0.000  0.623  0.377  0.9022
355523  0.000  0.539  0.461  0.7178
507481  0.000  0.420  0.580  0.7845
471727  0.000  0.236  0.764  0.8658
68541   0.353  0.647  0.000 -0.7269
230930  0.000  0.763  0.237  0.4215
477300  0.000  1.000  0.000  0.0000
345815  0.000  0.504  0.496  0.7845
333038  0.000  0.196  0.804  0.6249
377963  0.000  0.213  0.787  0.5719
275729  0.394  0.606  0.000 -0.4404
459528  0.000  0.357  0.643  0.9246
382128  0.000  0.455  0.545  0.5574
```

0.1.11 TF-IDF

short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word. The tf–idf is the product of two statistics, term frequency and inverse document frequenc

[60]:
```
vectorizer_tfidf = TfidfVectorizer(analyzer='word', ngram_range=(1, 1),␣
 ↪max_df=1.0,
                                    min_df=10, max_features=None)
X_train_counts_tfidf = vectorizer_tfidf.fit_transform(df_rev['Clean_Text'])
```

[61]:
```
print(X_train_counts_tfidf.shape)
```

```
(760030, 11489)
```

0.1.12 Word2Vec

[62]:
```
sent = [row.split() for row in df_rev['Clean_Text']]
```

[63]:
```
phrases = Phrases(sent, min_count=30, progress_per=10000)
```

[64]:
```
len(phrases.vocab)
```

[64]: 1601177

[65]:
```
bigram = Phraser(phrases)
```

[66]:
```
sentences = bigram[sent]
```

[67]:
```
# Count the number of cores in the computer.
cores = multiprocessing.cpu_count()
cores
```

[67]: 8

[68]:
```
w2v_model = Word2Vec(min_count=20,
                     window=2,
                     size=300,
                     sample=6e-5,
                     alpha=0.03,
                     min_alpha=0.0007,
                     negative=20,
                     workers=cores-1)
```

[69]:
```
t = time()

w2v_model.build_vocab(sentences, progress_per=10000)

print('Time to build vocab: {} mins'.format(round((time() - t) / 60, 2)))
```

```
Time to build vocab: 0.73 mins
```

[70]:
```
t = time()

w2v_model.train(sentences, total_examples=w2v_model.corpus_count, epochs=30,␣
↪report_delay=1)

print('Time to train the model: {} mins'.format(round((time() - t) / 60, 2)))
```

```
Time to train the model: 17.68 mins
```

[71]:
```
w2v_model.save('w2v_model')
```

[72]:
```
w2v_model = Word2Vec.load('w2v_model')
```

[73]:
```
X_wv = w2v_model[w2v_model.wv.vocab]
```

```
<ipython-input-73-0e467e2f040b>:1: DeprecationWarning: Call to deprecated
`__getitem__` (Method will be removed in 4.0.0, use self.wv.__getitem__()
instead).
  X_wv = w2v_model[w2v_model.wv.vocab]
```

[74]:
```
X_wv.shape
```

[74]:
```
(9893, 300)
```

[75]:
```
w2v_model.wv.vocab

print(w2v_model.wv.most_similar(positive=["bedroom"]))
```

```
[('bathroom', 0.7053378224372864), ('room', 0.6507285833358765), ('en_suite',
0.5351669192314148), ('shower', 0.5151000022888184), ('smelt_musty',
0.5012235045433044), ('toilet', 0.49664899706840515), ('shower_cubicle',
0.495363712310791), ('cubicle', 0.4854569733142853), ('ensuite',
0.4834802746772766), ('small', 0.4825647175312042)]
```

[76]:
```
print(w2v_model.wv.most_similar(positive=["smell"]))
```

```
[('smelt', 0.7130299210548401), ('smelling', 0.7024638056755066), ('odor',
0.6983328461647034), ('strong_smell', 0.6854673027992249), ('stench',
0.6745424270629883), ('odour', 0.6662341952323914), ('smelled',
0.6581301093101501), ('stank', 0.6419482231140137), ('smell_stale',
0.6371287107467651), ('stunk', 0.6257504820823669)]
```

[77]:
```
w2v_model.wv.similarity('excellent','board')
```

[77]:
```
-0.08814627
```

[78]:
```
w2v_model.wv.similarity('expensive','bad')
```

[78]:
```
0.15592173
```

[79]:
```
w2v_model.wv.similarity('burger','steak')
```

[79]:
```
0.5474285
```

[80]:
```
w2v_model.wv.doesnt_match(['nice', 'good', 'expensive'])
```

[80]:
```
'expensive'
```

[81]:
```
w2v_model.wv.most_similar(positive=["good", "location"],␣
↪negative=["resturant"], topn=3)
```

[81]:
```
[('great', 0.6724668145179749),
 ('perfect', 0.6318007707595825),
 ('excellent', 0.6296056509017944)]
```

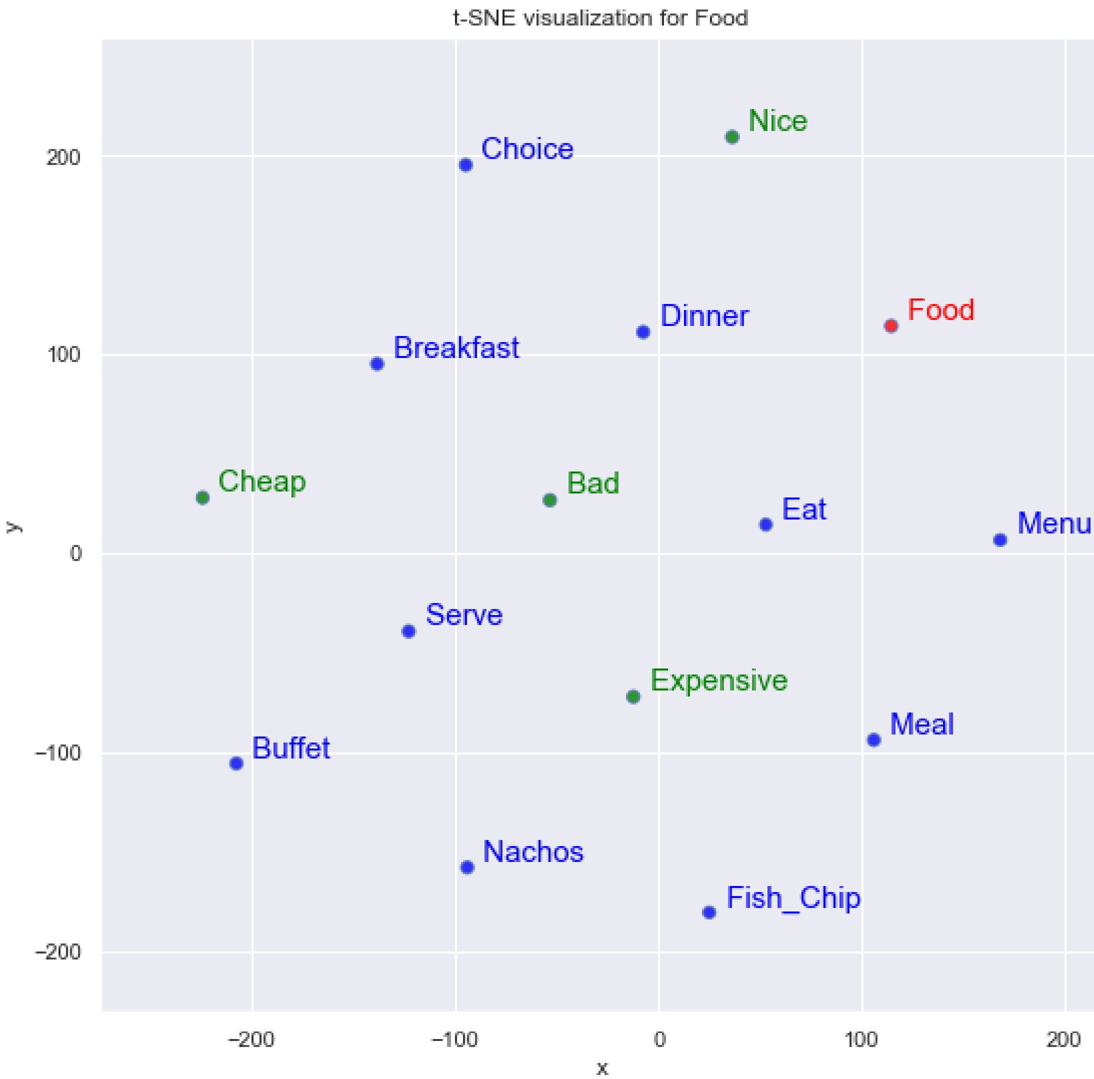
[82]:
```
def tsnescatterplot(model, word, list_names):
    """ Plot in seaborn the results from the t-SNE dimensionality reduction␣
↪algorithm of the vectors of a query word,
    its list of most similar words, and a list of words.
```

```
    """
    arrays = np.empty((0, 300), dtype='f')
    word_labels = [word]
    color_list  = ['red']

    # adds the vector of the query word
    arrays = np.append(arrays, model.wv.__getitem__([word]), axis=0)

    # gets list of most similar words
    close_words = model.wv.most_similar([word])

    # adds the vector for each of the closest words to the array
    for wrd_score in close_words:
        wrd_vector = model.wv.__getitem__([wrd_score[0]])
        word_labels.append(wrd_score[0])
        color_list.append('blue')
        arrays = np.append(arrays, wrd_vector, axis=0)

    # adds the vector for each of the words from list_names to the array
    for wrd in list_names:
        wrd_vector = model.wv.__getitem__([wrd])
        word_labels.append(wrd)
        color_list.append('green')
        arrays = np.append(arrays, wrd_vector, axis=0)

    # Reduces the dimensionality from 300 to 15 dimensions with PCA
    reduc = PCA(n_components=15).fit_transform(arrays)

    # Finds t-SNE coordinates for 2 dimensions
    np.set_printoptions(suppress=True)

    Y = TSNE(n_components=2, random_state=0, perplexity=15).fit_transform(reduc)

    # Sets everything up to plot
    df = pd.DataFrame({'x': [x for x in Y[:, 0]],
                       'y': [y for y in Y[:, 1]],
                       'words': word_labels,
                       'color': color_list})

    fig, _ = plt.subplots()
    fig.set_size_inches(9, 9)

    # Basic plot
    p1 = sns.regplot(data=df,
                     x="x",
                     y="y",
                     fit_reg=False,
```

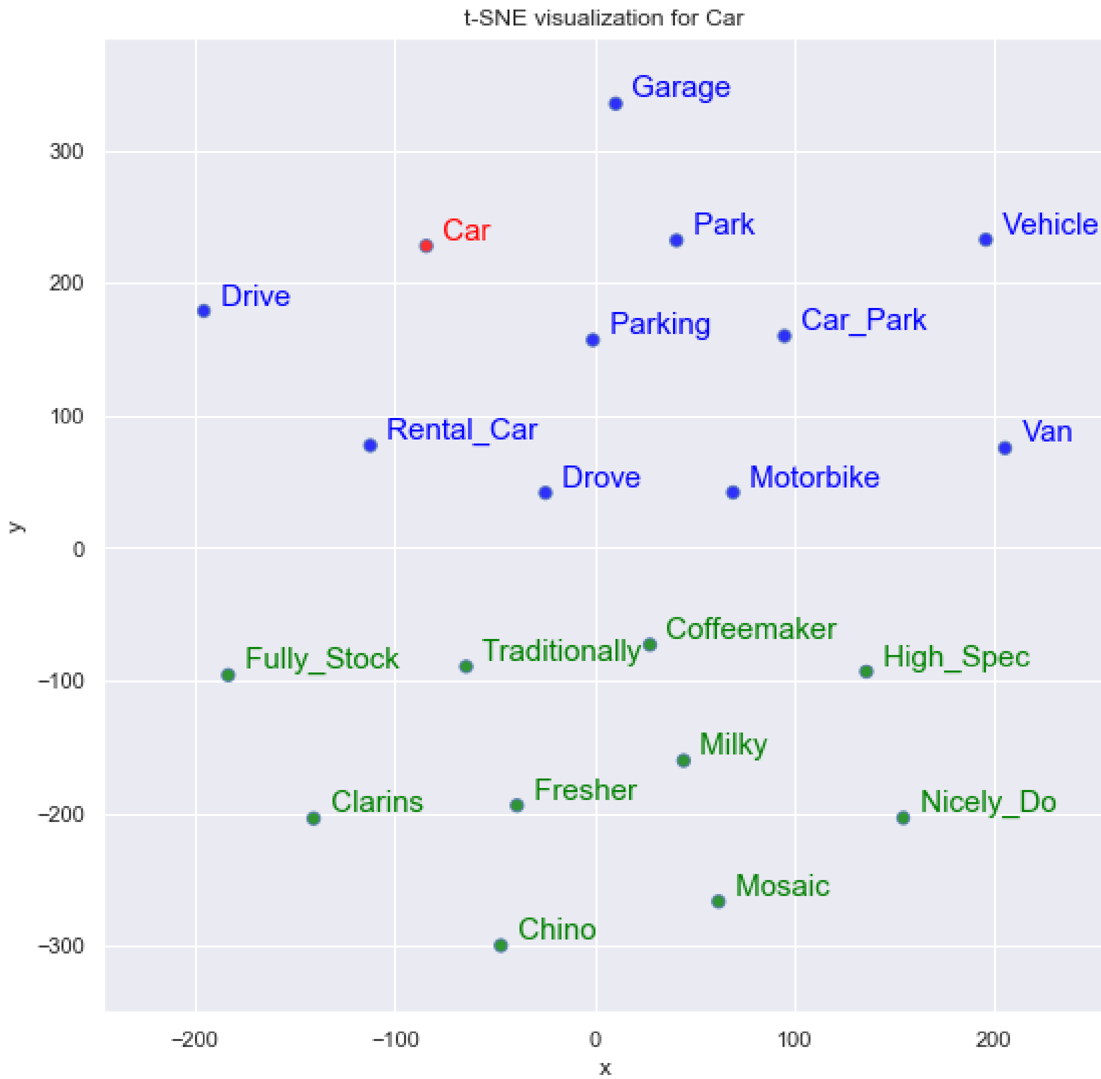
```
                     marker="o",
                     scatter_kws={'s': 40,
                                  'facecolors': df['color']
                                 }
                    )

    # Adds annotations one by one with a loop
    for line in range(0, df.shape[0]):
         p1.text(df["x"][line],
                 df['y'][line],
                 '  ' + df["words"][line].title(),
                 horizontalalignment='left',
                 verticalalignment='bottom', size='medium',
                 color=df['color'][line],
                 weight='normal'
                ).set_size(15)


    plt.xlim(Y[:, 0].min()-50, Y[:, 0].max()+50)
    plt.ylim(Y[:, 1].min()-50, Y[:, 1].max()+50)

    plt.title('t-SNE visualization for {}'.format(word.title()))
```

By utlizing a neural network model, the word2vec technique will help establish word assocations from these corpus of texts through numerical representations. The numerical representations will provide value that will be scalable to the models attained

[83]:
```
tsnescatterplot(w2v_model, 'food', ['bad', 'expensive', 'cheap', 'nice'])
```



[89]:
```
tsnescatterplot(w2v_model, 'car', [i[0] for i in w2v_model.wv.
↪most_similar(negative=["car"])])
```



Based on the scatterplot above, if one were to add vectors for the word such as "car", it appears to be just below zero into the negative meaning that it has a weaker value as compare others such as "Rental_Car". Words highlighted in green represent correlations of positivity in value towards the actual word "Car" at hand

[88]:
```
word_freq = collections.defaultdict(int)
for sent in sentences:
    for i in sent:
        word_freq[i] += 1
len(word_freq)
```

[88]: 70762

[90]:
```python
sorted_with_freq = sorted(word_freq.items(),key=operator.
 ↪itemgetter(1),reverse=True)
sorted_with_freq[:15]
```

[90]:
```
[('room', 379752),
 ('staff', 233951),
 ('hotel', 204586),
 ('location', 192493),
 ('breakfast', 138271),
 ('good', 134868),
 ('great', 113888),
 ('bed', 101076),
 ('friendly', 89975),
 ('helpful', 80251),
 ('clean', 79883),
 ('nice', 78815),
 ('stay', 65774),
 ('comfortable', 65116),
 ('excellent', 63847)]
```

[91]:
```python
v = sorted_with_freq[:15]
for word,freq in v:
    print(word)
```

```
room
staff
hotel
location
breakfast
good
great
bed
friendly
helpful
clean
nice
stay
comfortable
excellent
```

0.1.13 Modeling

We will try to do a suppervised learing model for the predection if the Raw text is postive or negtive text then we will do another unsupervised learining modle to cluster the text based on the analysis of the WordeVec and TF-IDF

0.1.14 Supervisioned Learning

Supervised Learning based on TF-IDF

[92]:
```
#We will use the matrix extracted erlier from the TF-IDF
X_train_counts_tfidf.shape
```

[92]: (760030, 11489)

0.1.15 Naive Bayes with TF-IDF

[93]:
```
params = {
    'alpha':[x for x in range(0, 50, 10)]
}

# Initialize Bayes Classifier.
nb = MultinomialNB()

# We need to tune the alpha parameter, in order to do this we use GridSearchCV.
clf = GridSearchCV(nb, param_grid=params,cv=2)

# We X_train_counts_bow because svd has negative values.
x_train, x_test, y_train, y_test = train_test_split(X_train_counts_tfidf,␣
↪df_rev['Postive_text'], test_size=0.3, random_state=0)

clf.fit(x_train, y_train)
print("Best param for alpha is {}".format(clf.best_params_))
```

```
Best param for alpha is {'alpha': 10}
```

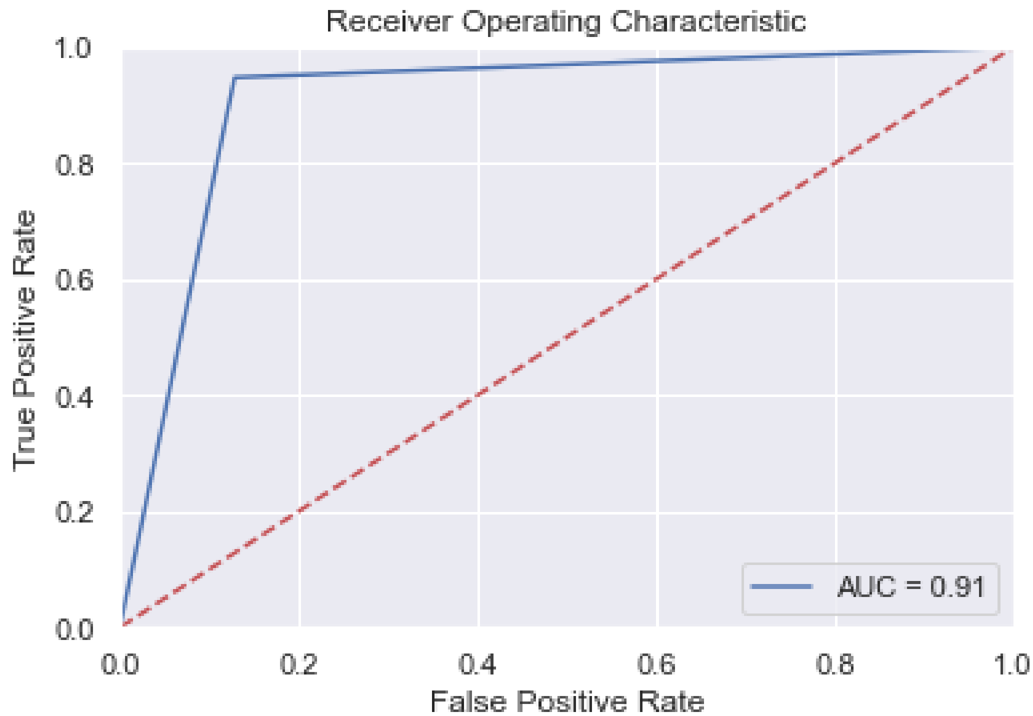
[94]:
```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
y_pred = clf.predict(x_test)
print(confusion_matrix(y_test, y_pred))
print("Accuracy score: {:.5f}".format(accuracy_score(y_test, y_pred)))
```

```
[[ 78965  11630]
 [  7203 130211]]
Accuracy score: 0.91740
```

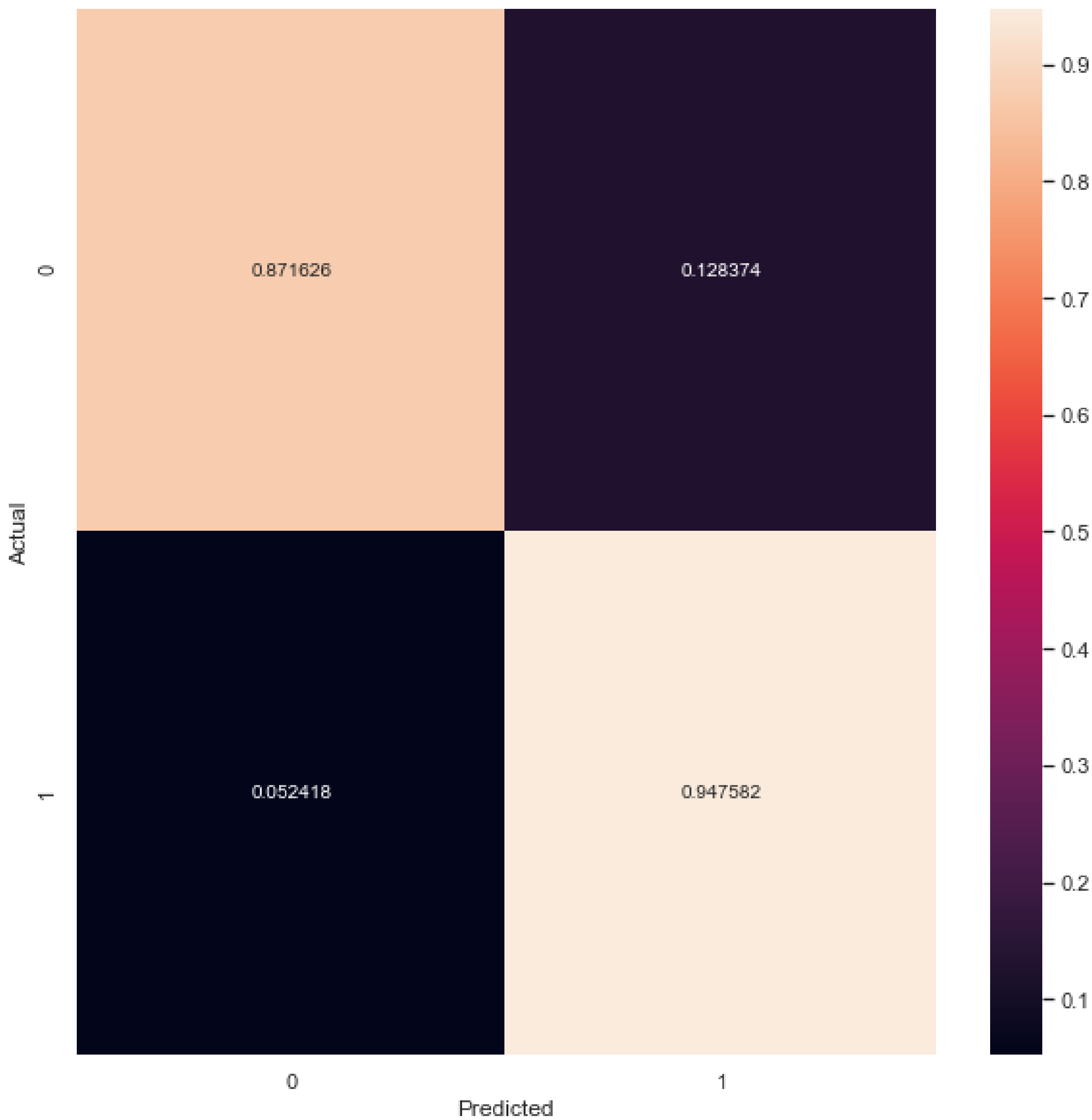
[95]:
```
probs = clf.predict_proba(x_test)
preds = probs[:,1]
fpr, tpr, threshold = roc_curve(y_test, y_pred)
roc_auc = auc(fpr, tpr)
```

[96]:
```
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
```

```
plt.xlabel('False Positive Rate')
plt.show()
```



[97]:
```
conf_mat = confusion_matrix(y_test, y_pred)
conf_mat = conf_mat / conf_mat.astype(np.float).sum(axis=1)[:, np.newaxis]
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(conf_mat, annot=True, fmt='f')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```



0.1.16 Logistic Regression using TF-IDF

[98]:
```
X_train_counts_tfidf

x_train, x_test, y_train, y_test = train_test_split(X_train_counts_tfidf,␣
↪df_rev['Postive_text'], test_size=0.3, random_state=55000)
```

[99]:
```
print(x_train.shape)
print(y_train)
print(x_test.shape)
print(y_test.shape)
```

```
(532021, 11489)
276162    1
146525    1
324711    0
364216    0
452105    1
         ..
264295    1
298295    0
176942    1
4350      1
171774    1
Name: Postive_text, Length: 532021, dtype: int64
(228009, 11489)
(228009,)
```

[100]:
```
X_train_counts_tfidf
```

[100]:
```
<760030x11489 sparse matrix of type '<class 'numpy.float64'>'
        with 8193553 stored elements in Compressed Sparse Row format>
```

[101]:
```
LR_model = LogisticRegression(solver='lbfgs')
LR_model.fit(x_train, y_train)
y_predict_lr = LR_model.predict(x_test)
```

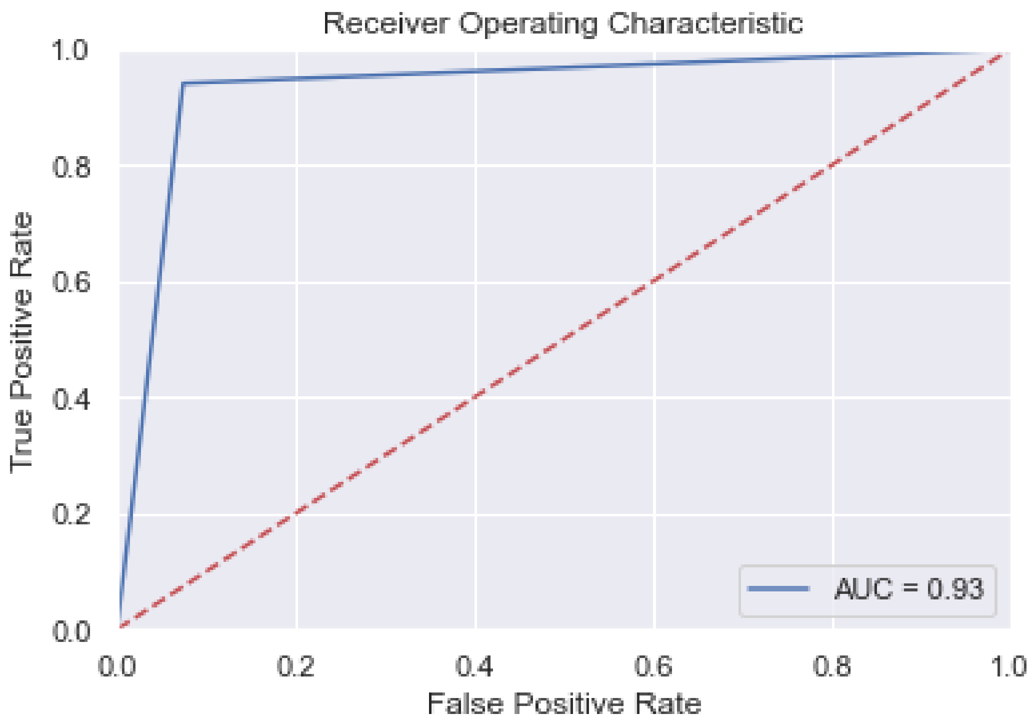
[102]:
```
print('\nConfusion matrix\n',confusion_matrix(y_test, y_predict_lr))
print(classification_report(y_test, y_predict_lr))
```

```
Confusion matrix
 [[ 84044   6717]
 [  8106 129142]]
              precision    recall  f1-score   support

           0       0.91      0.93      0.92     90761
           1       0.95      0.94      0.95    137248

    accuracy                           0.93    228009
   macro avg       0.93      0.93      0.93    228009
weighted avg       0.94      0.93      0.94    228009
```

[103]:
```
probs = LR_model.predict_proba(x_test)
preds = probs[:,1]
fpr, tpr, threshold = roc_curve(y_test, y_predict_lr)
roc_auc = auc(fpr, tpr)
```

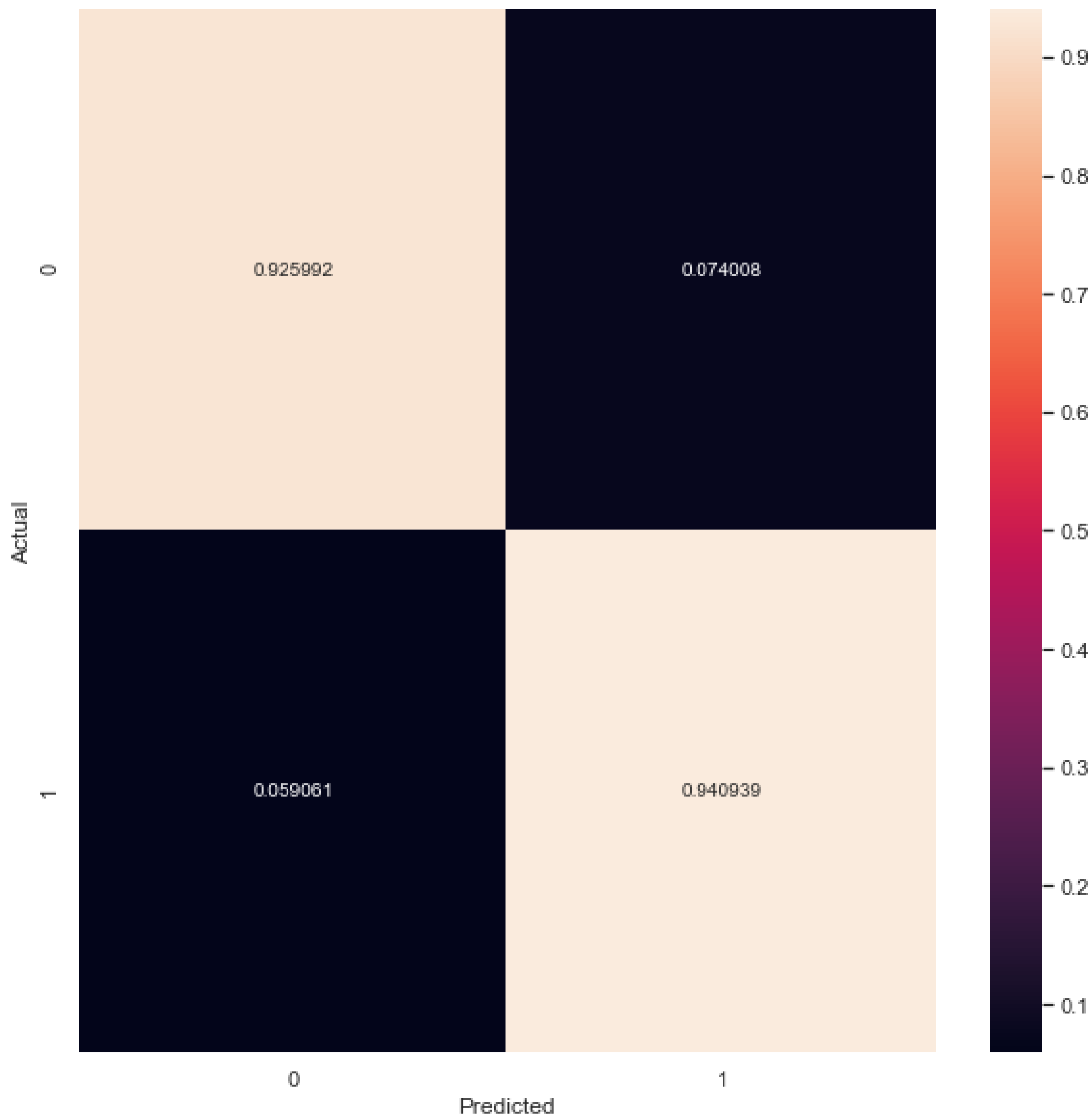
[104]:

```
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



[105]:

```
conf_mat = confusion_matrix(y_test, y_predict_lr)
conf_mat = conf_mat / conf_mat.astype(np.float).sum(axis=1)[:, np.newaxis]
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(conf_mat, annot=True, fmt='f')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```



Supervised Learning based on Word2Vec

[106]:
```
len(sentences)
```

[106]: 760030

[107]:
```
def sentence_vectors(model, sentences):
    #Collecting words that are known to the model
    model_voc = set(model.wv.vocab.keys())

    X = []
```

```
    for sentence in sentences:

        # Empty array of zeros.
        sent_vector = np.zeros(model.vector_size, dtype="float32")

        # Use a counter variable for number of words in a text
        nwords = 0

        # Sum up all words vectors that are know to the model
        for word in sentence:
            if word in model_voc:
                sent_vector += model[word]
                nwords += 1.

        # Now get the average
        if nwords > 0:
            sent_vector /= nwords
        X.append(sent_vector)
    return X

X_vw = sentence_vectors(w2v_model, sentences)
```

```
<ipython-input-107-c9bcfe8d30c1>:18: DeprecationWarning: Call to deprecated
`__getitem__` (Method will be removed in 4.0.0, use self.wv.__getitem__()
instead).
  sent_vector += model[word]
```

the Array to be used in the modles is X__vw

0.1.17 Logistic Regression with Word2 Vector

[108]:
```
x_train_wv, x_test_wv, y_train_wv, y_test_wv = train_test_split(X_vw,␣
 ↪df_rev['Postive_text'], test_size=0.3, random_state=55000)
```

[109]:
```
LR_model_wv = LogisticRegression(solver='lbfgs')
LR_model_wv.fit(x_train_wv, y_train_wv)
y_predict_lr_wv = LR_model_wv.predict(x_test_wv)
```

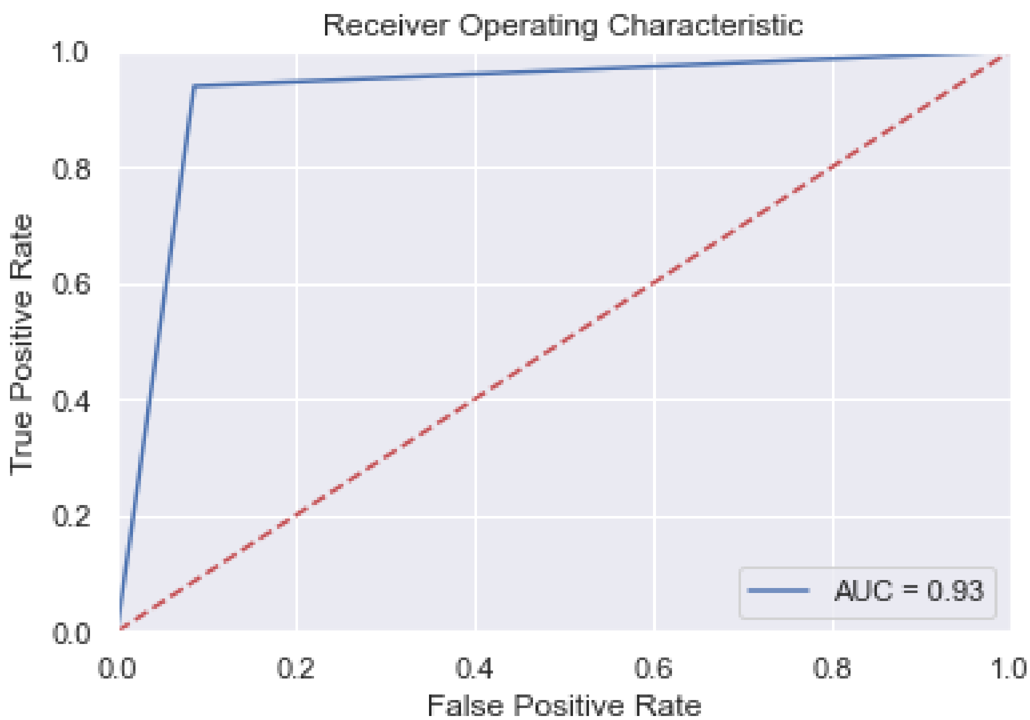
[110]:
```
print('\nConfusion matrix\n',confusion_matrix(y_test_wv, y_predict_lr_wv))
print(classification_report(y_test_wv, y_predict_lr_wv))
```

```

Confusion matrix
 [[ 82920   7841]
 [  8281 128967]]
              precision    recall  f1-score   support

           0       0.91      0.91      0.91     90761
           1       0.94      0.94      0.94    137248
```

```
   accuracy                           0.93    228009
  macro avg       0.93      0.93      0.93    228009
weighted avg      0.93      0.93      0.93    228009
```

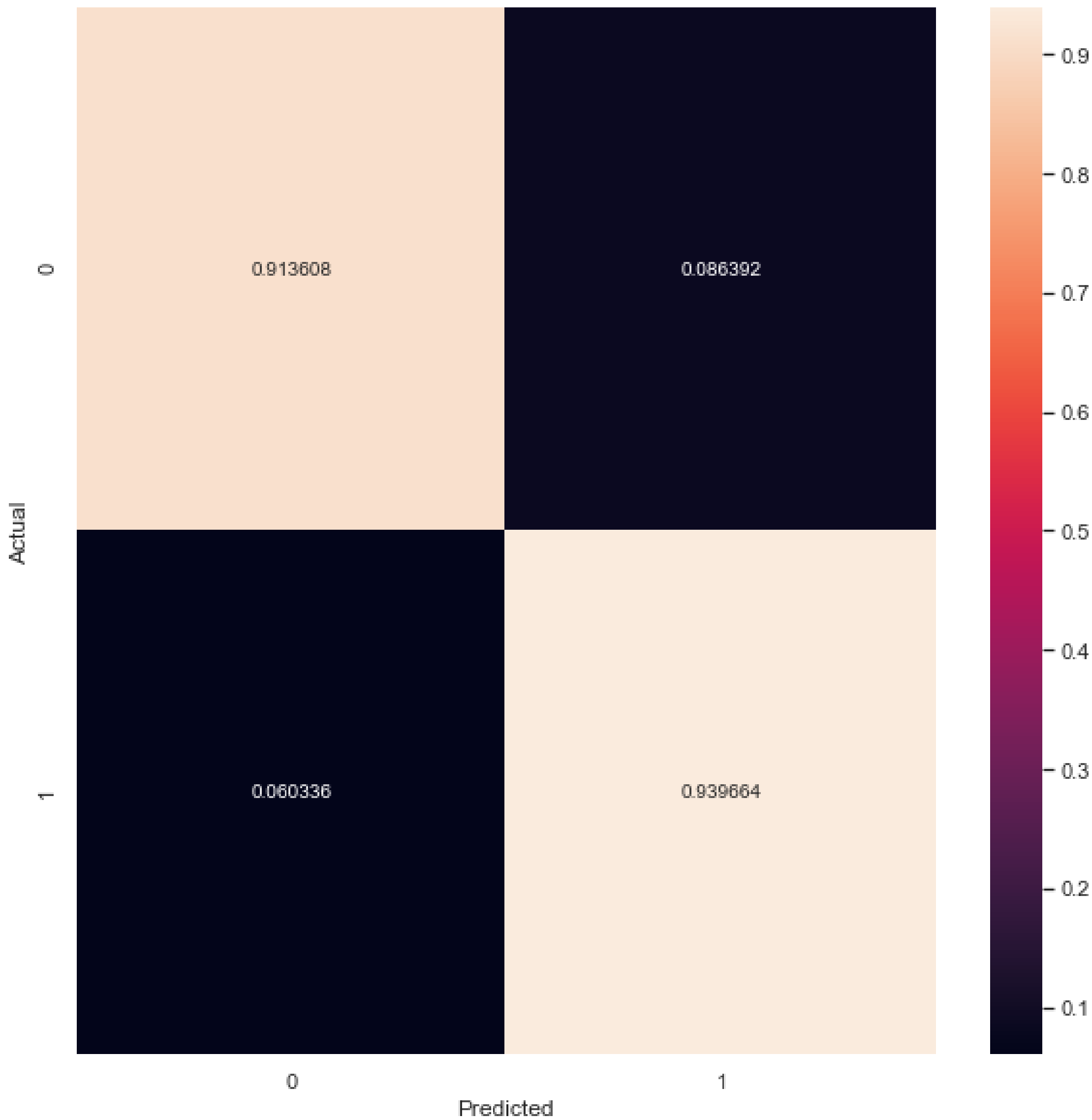
[111]:
```
probs_wv = LR_model_wv.predict_proba(x_test_wv)
preds_wv = probs_wv[:,1]
fpr_wv, tpr_wv, threshold_wv = roc_curve(y_test_wv, y_predict_lr_wv)
roc_auc = auc(fpr_wv, tpr_wv)
```

[112]:
```
plt.title('Receiver Operating Characteristic')
plt.plot(fpr_wv, tpr_wv, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



[113]:
```
conf_mat = confusion_matrix(y_test_wv, y_predict_lr_wv)
conf_mat = conf_mat / conf_mat.astype(np.float).sum(axis=1)[:, np.newaxis]
```

```
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(conf_mat, annot=True, fmt='f')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```



[114]:
```
import joblib
from joblib import dump
```

[115]:
```
filename = 'finalized_model.sav'
joblib.dump(LR_model_wv, filename)
```

[115]: ['finalized_model.sav']

We attained three models that comprise of the actual and predicted scores of the word-turned-vectors showing different correlations for each review. The first one is a Naive Bayes with TF-IDF where it produced a positive value of 0.9475, a negative value of 0.8716, and an average score of 0.91. The second one is a Logistic Regression with TF IDF. This one produced a positive value of 0.9409, a negative value of 0.9259, and an average score of 0.93. The third and last model is a logistic regression model using Word2Vec. This one produced a positive value of 0.9399, a negative value of 0.9128, and an average score of 0.93.

Based on the output of the three models built, it appears that the Logistic Regression with TF IDF model has a better accuracy and balance of positive to negative text. While the accuracy score of this one is the same as the third model, the second model edges out in terms of positive accuracy.

0.1.18 Deployment

The model developed provides a plain demonstration for the interested third parties. This model has the potential to be used to aid hotels and sites that provide hospitality services in taking reviews from customers on the experience they have had in their hotel stays. However, the model does not hold much of its use of interactivity considering the lack of accuracy and latency.

Since the Hotel Reviews dataset was gathered in the span of two years. It is significant to update it like a consensus every four to five years in order to attain standards of ethical research. This involves running as many backend instances as possible in order to finetune scalability. With the utilization of Python, the Dash app developed would be the appropriate deployment method in gathering new data. Our app is also created to cater to the marketing experts and analysts who would like to use it for their analysis and insight of topics as such as of the dataset.

The model developed in this project was utilized to create the Dash App. The coding for this application can be found on Github

Dash App will using the Vader sentiment analyser to do life analysis for the complains in addtion to using the insights from the Word2Vector to help the marketing team to extract the common issues the Hotels are facing