Text Classification of Reviews using Machine Learning Models, Review Summarization and Building Content-Based Hotel Recommender System

Himanshu Gupta

A Dissertation

Presented to the University of Dublin, Trinity College

in partial fulfilment of the requirements for the degree of

Master of Science in Computer Science (Data Science)

Supervisor: Dr. Bahman Honari

September 2020

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Himanshu Gupta

September 7, 2020

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Himanshu Gupta

September 7, 2020

Acknowledgments

I would attribute the successful completion of this thesis to various people who have guided and assisted me all through the project right from its inception.

First and foremost, I would like to take this opportunity to express my sincere gratitude to my supervisor, Prof. Bahman Honari for his continuous guidance, and motivation. He assisted me with insightful discussions and invaluable feedback time and again throughout my thesis.

I am also extremely thankful to my second reader Prof. Susan Connolly who gave me invaluable suggestions for improving and orchestrating my thesis during my presentation. I owe utmost gratitude to Trinity College Dublin, the School of Computer Science and Statistics for immense learning and technical exposure to enhance my skills.

Finally, my heartfelt gratitude to my family and friends for their indispensable support and motivation during this course of study. They suggested different ideas in making this project unique.

Himanshu Gupta

University of Dublin, Trinity College
September 2020

Text Classification of Reviews using Machine Learning Models, Review Summarization and Building Content-Based Hotel Recommender System

Himanshu Gupta, Master of Science in Computer Science

University of Dublin, Trinity College, 2020

Supervisor: Dr. Bahman Honari

Reviews are preliminary elements that travellers look at and analyse before booking any hotel. These reviews are of utmost importance for new customers as they can get insights about the hotels based on others' experiences. Moreover, it can benefit the hotel management staff so that they can upgrade the services and products based on the submitted feedback after thorough analysis. As per the Google trends, Europe was the most visited destination in 2018, approximately 710 million international tourists' arrivals occurred. Hence the number of travellers submitting the reviews was huge.

A widely faced challenge due to this huge chunk of reviews was that when a new traveller comes in, he would be reluctant to go through all the reviews which can lead to selecting an undesired hotel even though the ratings are considerably good. This outstanding challenge has motivated me to work on the European hotel dataset which would present the travellers with the most relevant reviews, which in turn would help them in choosing the best hotels in comparatively less time.

To achieve this, we resort to Review Classification method, which classifies the reviews into different classes (good and bad reviews) and further compare different machine learning models. In case the customer wants to analyse each hotel's review, it would be preferable to select the most relevant sentences from lengthy reviews. We will employ the Review Summarization technique to address this. Finally, customers may have some preferences like 'large room', 'couple-friendly' etc. based on which they would want to filter the hotels. This will be further accomplished by building a simple content-based recommender system. The results obtained in this study include – accurately classified reviews, precisely summarised huge reviews and recommended hotel list in order of relevance. Also, it was found that deep learning methods could classify reviews more accurately as compared to the other machine learning approaches.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

With the advent of enhanced online booking systems, travellers are becoming more vigilant about their choice of accommodation which can facilitate their decision-making process. There are various applications in the market now, which can assist users in their travel-related queries such as preferred location, the shortest route from famous places, etc. To address these requirements various recommendation systems such as demographic-based, social filtering based systems are created by researchers to help the users to make judicious decisions[1]. The filtered suggestions which show up at the end result are based on the user's experiences, time of travel, activities, pictures, etc. One of the prominent software - 'TripMatcher' is based on artificial intelligence that suggests travellers a best-suited destination based on the user profile. There has been a good amount of work done to assist customers already but enhancing the travel system to make it more vigorous continues to be a popular topic among data scientists.

1.2 Motivation

Travelling has become a common recreational activity for different reasons, be it for learning new cultures, boosting inner self-confidence, exploring new destinations, or exploring adventure. It has become a stress reliever activity and the advancement in booking systems of flight, hotels, etc. has made common people more accessible to tourist places without much dependency and hassle. Also, before booking hotels (or availing any service), customers are more interested in knowing about these hotels beforehand. Most of the customers want to inquire about the overall service including the accessibility of any hotel, i.e. whether the service is ‘Good’ or ‘Bad’ without deep-diving into specific reviews. Moreover, there might be multiple reviews for a hotel, hence it becomes a tedious task to go through every review, which could be very time-consuming at times. These are some of the prominent reasons when customers prefer overall feedback for hotels whether it is ‘Good’ or ‘Bad’. Additionally, summarising these good and bad reviews would be a great add-on and would ease the process of filtering the desired hotels.

For instance, some customers want to read the reviews to understand if a particular hotel aligns with their expectations (based on services like ‘breakfast’, ‘room size’, ‘WiFi’ or other amenities). As the reviews for any hotels will be in huge number, hence there should be some way to read the important aspects about the hotel instead of scanning through all the sentences present. So, it is imperative to have a review summarization implemented which can provide the most relevant information about all the hotels at a go.

Furthermore, customers usually compare different hotel ratings with others but all the customers have different points of view about the hotel while submitting the review score. For example, if a customer found the finds facility of ‘Wifi’ to be very good, he may give the ratings based on the ‘Wifi’ but there are other factors involved too which decide whether the hotel is good or bad which eventually depends on the customer’s requirement. For an old couple, the comfortability of bed and the presence of elevators are more important than ‘Wifi’, hence it is very important to have a system in place which can recommend a list of hotels based on the customer preferences rather than going deep into the details of every review. This can be addressed by developing a content-based recommender system.

1.3 Research Question

How accurately can we classify the hotel reviews, precisely summarise them and further recommend the most relevant hotels based on the user preferences?

1.4 Research Objective

This dissertation aims at studying and finding various approaches that can make the customer's search efficient in selecting the desired hotel when a huge number of reviews are present.

In this course of study, firstly, the reviews are classified using traditional machine learning and deep learning approaches and hence compare these techniques to identify the better model based on the evaluation metrics. The techniques TF-IDF vectorization and Word Embedding (via Global Vector) would be employed to represent text in vector during model training.

Secondly, the submitted reviews would be summarized into top 'n' ranked sentences based on the cosine similarity which will help customers in selecting the hotel faster rather than scanning all the reviews for a particular hotel.

Finally, we will build a simple content-based recommender system which will display the top 'n' list of hotels taking customer's preference as input. The customer preference can be a quality of services he expects, which can be provided in textual form ('good location', 'friendly staff' etc.) along with other features (like travel reason = Business/Leisure, Group Type = Solo/Couple/Family, Night Stay = 4 /8).

1.5 Research Challenge

- Due to COVID-19, the accessibility to high computational machines for training the model was less. As the dataset was huge, it took more time during the model training.

- There were many features (almost 17) present in the dataset, hence finding the best-related features and visualizing their relationship with the target variable was a challenge.
- Finding a suitable number of hidden layers is an uphill task when training a deep learning model. Increasing the number of layers would introduce more complexity in the model therefore we need to choose layers that can provide the trade-off between complexity and ease of training.
- Understanding various recommendation system techniques and finally selecting Content-based filtering as the most plausible option for this dataset.

1.6 Thesis Overview

The thesis will present the classification of the reviews and consequently compare various models created using TF-IDF vectorization via ML algorithms and Word Embedding via Deep Learning library Keras. The classification of reviews is presented as ‘Good’ or ‘Bad’ based on existing review scores provided by the customers. Important features are then extracted from the reviews for which the relationship with the target variable (‘Good’ or ‘Bad’) is estimated and is further graphically visualized using various python libraries presented in the chapter Data Visualization and Analysis.

Generally, Customers prefer reading only the relevant positive and negative aspects of the hotels in a precise manner based on other customer experiences rather than scanning through all the lengthy reviews. Hence implementation of the aforementioned approach is well presented in the dissertation where positive and negative reviews are summarized based on cosine similarity with a technique known as Text Summarization.

Further, a content-based recommender system is developed where preferences are provided by customers as input. This will list all the preferred hotels in order of relevance, when customers provide input as Keywords, Travel Reason, Stayed Nights, Solo/Couple/Group, and Country Name, which saves a considerable amount of time for customers looking for the intended hotels. We have also provided the flexibility to modulate the number of preferred hotel lists which decides how many matching hotels will be displayed with respective scores. The score values close to '1' indicate that the hotel is highly recommended while tending to '0' signify not recommended.

1.7 Thesis Structure

Below is the flow of the presented thesis:

Chapter 1: Deals with a brief introduction to the research. It discusses about the background, research objective, challenges involved, and overview of the thesis.

Chapter 2: Depicts literature review which deals with text classification, sentiment analysis, Imbalanced in data, summarization of text, and Recommender Systems.

Chapter 3: Describes the collection and description of the dataset used in our study, followed by feature engineering and handling of missing and imbalanced hotel data.

Chapter 4: Imparts understanding about the data via Data Visualization.

Chapter 5: Explains various methodologies used to achieve the research objectives.

Chapter 6: Describes various evaluation metrics used in our study.

Chapter 7: Presents the results achieved during the course of our study.

Chapter 8: Provides an overall conclusion and future work that needs to be done.

1.8 Keywords

Tokenization, Lemmatization, TF-IDF, Word Embeddings, GloVe, LSTM, Cosine Similarity, Similarity Matrix, Imbalanced dataset, Correlation, Sampling methods, Text Summarization, Recommender System, Confusion Matrix, Precision, Recall.

Chapter 2

Literature Review

In this chapter, we will discuss the insights about work done in the NLP field on various techniques. This understanding has helped me in achieving my research objective. Let's discuss them individually.

2.1 Classification of Text Reviews

The online product trading is exponentially growing day by day and this increase in e-commerce services has lead people to first review the product on various sites before buying them. For any product or hotel, there are numerous reviews present online, these reviews have positive as well as negative feedback. So to find that any product or hotel is good or bad, these reviews have to be classified with some machine learning algorithms. Much researches have been done on this topic to precisely identify the sentiments (good/bad) about any product based on written text. In the research paper [2], the classification is performed via Naïve Bayes and SVM techniques on movie data, it was found that Naïve Bayes had a good accuracy compared to SVM. This paper suggests that selecting the features based on the data analysis plays a key role in enhancing the model's accuracy.

This paper [3] presents the domain-based approach by extracting important words from the corpus. The resulting dictionary was used in the classification of reviews based on sentiment analysis. The customer reviews were classified into two classes good and bad, this approach is called lexicon-based classification. The reviews were taken from

the TripAdvisor written in the English language. It is worth noting that in paper [4], the author presented that a lexicon specific approaches are better than the statistically qualified classifiers. The author stated that lexicon specific approach increases the system efficiency when texts come from various areas.
The paper [5] describes the rule-based derivation of features from Amazon product review using the POS tagging, words, and sentence polarity detection. Steps used in Feature extraction are: POS tagging, using grammar features are derived, derivation of words, word/sentence polarity detection and eventually merging results to generate a summary.

2.2 Sentiment Analysis

The sentiment analysis is a field of study which deals with the analysis of people's reaction, emotions, viewpoint, and attitude from the written text. This technique is widely used in the analysis of data present in social media, blogging sites, biomedical domain, customer's feedback about any product etc. It basically classifies the text as positive, neutral, and negative sentiments at sentence or document level termed as polarity. The results which are achieved after the sentiment analysis render any organization to build strategies to improve customer satisfaction by reforming the quality of the product. Some of the researchers focused on extracting key features from the text, assigning different polarities for the given text, build different machine learning models to predict the sentiments hence compare the effectiveness of these models and the importance of using the multiple classifiers to enhance the accuracy of analysis [6].

The paper [7] includes the sentiment analysis approach to determine how the sentiments are considered in any text, and can be classified as favorable or unfavorable opinions. In this paper researchers basically used the fragment of text rather than the whole document to find out the relation between the expressions and subject term. So by applying the different algorithms on the text fragment, the model yielded good accuracy. In order to identify the structure of text fragment, POS tagging was used which describes verbs, adjectives, adverbs, and nouns in any sentence.The paper[8]describes about restricting the textual data in two ways that are (i) considering only Adjectives as a feature and (ii) creating a domain with common words, and hence created the

classification models based on these restrictions.
There is an abundance of data present on social media and one of the platforms is Twitter which has more than 330 million users and 200 million tweets per day. In the paper [9], the researchers have extracted the entities from the tweets and hence added these entities as a new features to improve the accuracy of polarity detection when applied to multiple twitter datasets. It was important to note that on combining POS tagging with n-gram models enhances the score of polarity detection. In my paper, I have presented the importance of extracting this information from the reviews which can be used as predictor variables for sentiment classification.

2.3 Sampling for Imbalanced Dataset

The skewness in the data is a challenge that needs to be addressed before feeding this data for building any machine learning model. While building a classification model, this imbalance in data introduces a learning problem which basically decreases the accuracy of classifiers. If the dataset has more number of records for one class (majority) while less number of records for other class (minority), the chances of the model getting biased towards the majority class is expected. There are various researches which are done to balance this skewed data. The paper [10], presented various methods to deal with this problem namely: oversampling (increasing the number of minority samples randomly), undersampling (decreasing the number of majority samples randomly).

There were some researchers [11] who develop a naïve method to balance the data using SMOTE where randomly new samples are created from the minority class based on the 'k' nearest neighbors. The new models (trained via classifiers as NB, Decision Tree, etc.) were compared and it was found that the SMOTE approach was very effective on the imbalanced dataset.
The paper [12] presented key insights about resolving the problem of data imbalance, below are some important inferences from the paper:

1. Selecting the right features for training can yield good results as compared to sampling techniques.

2. While performing feature selection it is recommended to use less complicated

searching approach as the results can be inaccurate otherwise.

3. As per the results obtained via performing sampling in a larger dataset, it was found that undersampling is more effective as compared to other techniques.

The researchers in the paper [13] presented an enhanced approach to solve the imbalance problem using EMOTE (Enhanced Minority Oversampling Technique). When trained using various classifiers (NB, Random Forest, etc.) then evaluating the results based on F-measures AUC etc., it was found that there was minimal loss of information using the EMOTE method.

2.4 Text Summarization

To opt for any product or service, it is always suggested to read and analyse the previously added reviews by the users. This provides the level of trust and satisfaction before buying the product. As the customers usually give reviews in large numbers so it is not possible for others to read all the comments for that particular product. Hence researchers in the paper [14], identified the positive and negative polarities in the reviews and eventually summarized the review comments based on features selected so that other users can get an idea about the product. As per this paper, the feature-based text summarization is done using a set of methods such as opinion word extraction, POS tagging, Identification of Orientation, feature pruning, etc.
The text summarization is used to convert the long text documents into a short version without losing meaningful information. There two ways to perform summarization, extractive, and abstractive approaches. In the paper [15] the researchers have used the extractive technique which chooses the important words, sentence, and paragraphs from the document. While in the case of the Abstractive technique described in paper [16], first the main idea is studied and formulated from the document employing the various linguistic methods to well describe the text and thereby generating the new text which precisely conveys the intended information.

This technique is very primitive which can be observed in a paper published in 1969 [17] the researchers create a summarization system that uses the sentence weights to classify the importance. They have used three methods for identifying the sentence weights: *Cue Technique* - Weights assignment was based on the existence of cue words in a defined dictionary. *Title Technique* - The Importance of weights was based on the tokens present in the title of documents. *Location Technique* - This approach was based on the sentences present in the initial place in the paragraph which is assumed to have high importance as compared to others.

2.5 Topic Modelling

It is an unsupervised machine learning method that employs examining documents, sentences, phrases, and words. Once these patterns have been identified, clusters will be formed based on the similarity of the words. The paper [18] describes two frameworks namely Latent Dirichlet Allocation and Word2Vec used for modeling a political based dataset. The data present in the corpus was pre-processed before feeding for the training LDA model. For training, the Genism library was opted using variation Bayes. Once LDA yielded the results, Word2Vec was employed on pre-processed data where words having lower frequency were ignored. Hence depending upon the proposal given by Mikolov and Baroni [19], the model was created via the CBOW method, which created a dense vector representation and this vector was restricted to 200 dimensions. Once both the trainings were performed, clustering was done to form the topics, and evaluation was done on purity criteria.

2.6 Recommender System

Due to the advent of the internet, now most of hotel bookings, product purchases, flight booking, etc. can be done easily. But this has introduced some problem for the customers, a customer finds cumbersome to search the desired products as a lot of information is present on online sites. Hence it is likely to have a system that can recommend the list of products, flights, etc. based on location or some other filter feature, this is termed as a recommender system. As per the various researches, the recommender systems can be derived from content, collaborative filtering, demographic and statistical techniques.

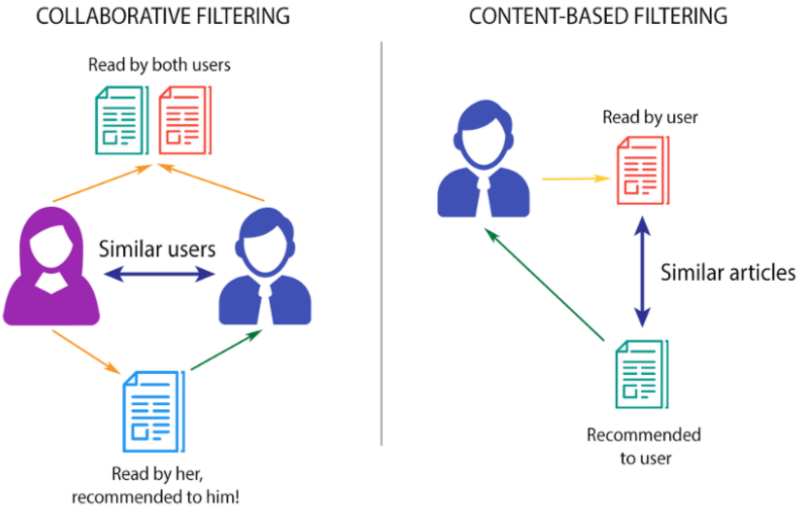


Figure 2.1: Comparison among the Content and Collaborative Filtering RS

In the case of demographic-based system [20], the customer's private data consisting of features like locations,etc. are used to recommend any product. The content-based filtering usually predicts the ratings of the unrated products of any customer. The features used in this filtering can be ratings, textual reviews [21], user sessions, etc.

The collaborative methods [22] use a concept of product similarity that is if some product is reviewed by a multiple users then the same product will be recommended to the in-flight users sharing the similarity in product ratings. It is important to specify that majority of these systems are not based on textual reviews [23] due to the complexity present in the understanding of written languages by machine. So basically the techniques to recommend the product depend upon TF-IDF, topic signature, and several occurrences of words. The comparison between the collaborative filtering and content-based approaches can be seen from the figure, which is discussed in the paper[24].

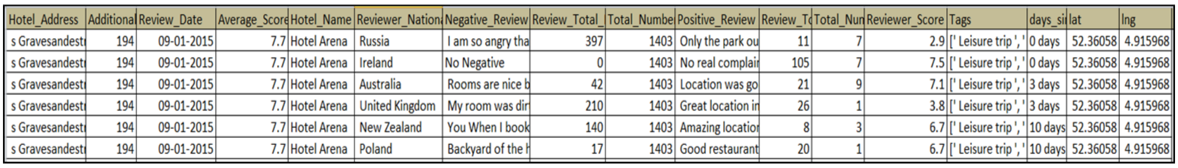
Chapter 3

Data Overview and Pre-processing

3.1 Data Overview and Feature Engineering

3.1.1 Data Collection

The data is related to reviews of European hotels and was scraped from Booking.com site. This data is available on Kaggle website in csv format consisting of 515k data records.

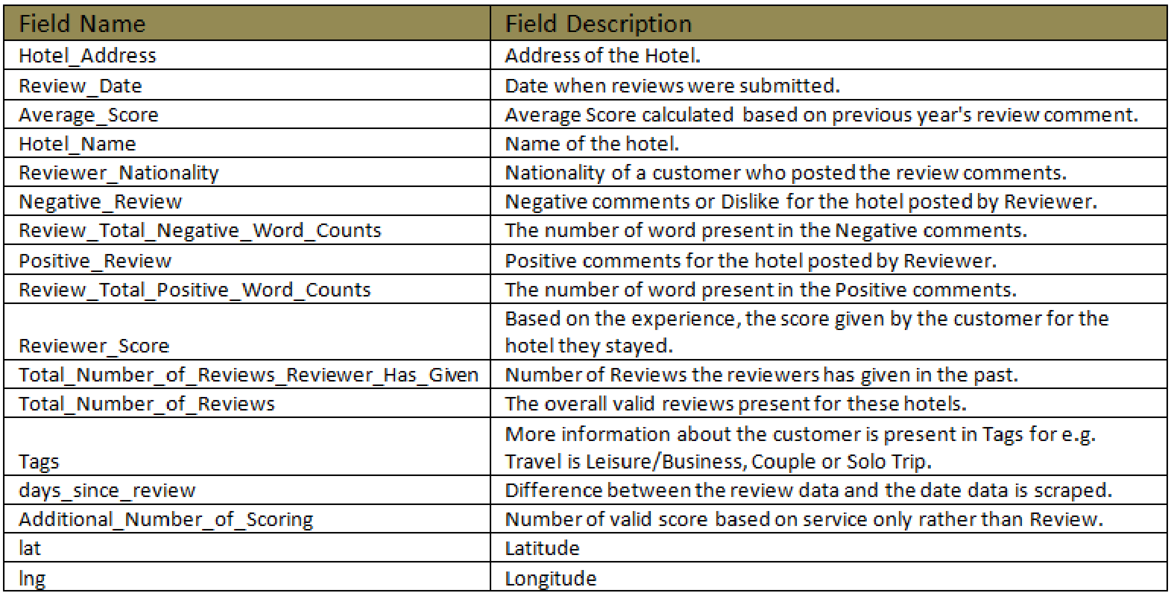


Hotel_Address	Additional	Review_Date	Average_Score	Hotel_Name	Reviewer_Nation	Negative_Review	Review_Total_	Total_Numbe	Positive_Review	Review_Tc	Total_Nun	Reviewer_Score	Tags	days_si	lat	lng
s Gravesandest	194	09-01-2015	7.7	Hotel Arena	Russia	I am so angry tha	397	1403	Only the park ou	11	7	2.9	[' Leisure trip ', '	0 days	52.36058	4.915968
s Gravesandest	194	09-01-2015	7.7	Hotel Arena	Ireland	No Negative	0	1403	No real complair	105	7	7.5	[' Leisure trip ', '	0 days	52.36058	4.915968
s Gravesandest	194	09-01-2015	7.7	Hotel Arena	Australia	Rooms are nice b	42	1403	Location was go	21	9	7.1	[' Leisure trip ', '	3 days	52.36058	4.915968
s Gravesandest	194	09-01-2015	7.7	Hotel Arena	United Kingdom	My room was dir	210	1403	Great location in	26	1	3.8	[' Leisure trip ', '	3 days	52.36058	4.915968
s Gravesandest	194	09-01-2015	7.7	Hotel Arena	New Zealand	You When I book	140	1403	Amazing locatior	8	3	6.7	[' Leisure trip ', '	10 days	52.36058	4.915968
s Gravesandest	194	09-01-2015	7.7	Hotel Arena	Poland	Backyard of the h	17	1403	Good restaurant	20	1	6.7	[' Leisure trip ', '	10 days	52.36058	4.915968

Figure 3.1: Few Samples of Hotel Review Data

3.1.2 Dataset Description

The dataset consists of reviews about 1493 hotels located across Europe. It has 515000 customer review records with 17 columns (fields). Below is the description of these fields:



Field Name	Field Description
Hotel_Address	Address of the Hotel.
Review_Date	Date when reviews were submitted.
Average_Score	Average Score calculated based on previous year's review comment.
Hotel_Name	Name of the hotel.
Reviewer_Nationality	Nationality of a customer who posted the review comments.
Negative_Review	Negative comments or Dislike for the hotel posted by Reviewer.
Review_Total_Negative_Word_Counts	The number of word present in the Negative comments.
Positive_Review	Positive comments for the hotel posted by Reviewer.
Review_Total_Positive_Word_Counts	The number of word present in the Positive comments.
Reviewer_Score	Based on the experience, the score given by the customer for the hotel they stayed.
Total_Number_of_Reviews_Reviewer_Has_Given	Number of Reviews the reviewers has given in the past.
Total_Number_of_Reviews	The overall valid reviews present for these hotels.
Tags	More information about the customer is present in Tags for e.g. Travel is Leisure/Business, Couple or Solo Trip.
days_since_review	Difference between the review data and the date data is scraped.
Additional_Number_of_Scoring	Number of valid score based on service only rather than Review.
lat	Latitude
lng	Longitude

Figure 3.2: Features with their description

3.1.3 Data Imputation

The number of missing values was checked and it was found that latitude and longitude columns have 3268 null values each, which sum up to 6535 total null values in the given dataset. There is no need to fill or remove these 'NA' values as these columns will not be used as predictor variables which will be discussed later in the coming chapters. Few positive and negative reviews consist of text as 'No Positive' and 'No Negative' respectively, which were changed to blank before concatenating them to make an overall review. The other changes in the data will be explained in the upcoming section.

3.1.4 Feature Engineering

There is new information extracted from the existing columns like ‘Hotel Address’, ‘Positive Review’, ‘Negative Review’, and ‘Tags’.

- The country names from the hotel address are extracted and the new column ‘Country Name’ is made.
- As almost all records have positive as well as negative reviews so polarity cannot be detected merely checking these columns. Also, there are some reviews

which have values like ‘No Negative’ and ‘No Positive’, which don’t give much information about good and bad views corresponding any hotel, so these values are replaced by spaces. To find the overall sentiment’s polarity and classify the reviews, Positive and Negative reviews are concatenated to find the overall review ‘Overall Review’ for each record in the dataset.

- There are some customer preferences mentioned in the column ‘Tags’ like ‘Trip is for Leisure or Business, Couple or Solo or Group, Number of Stayed Nights, etc. These details are extracted and new columns are made for each of these tags. Later, while performing the data visualization, it will be discussed that these tags carry important information that impact the hotel’s overall ratings.

- The frequency of bi-gram words is found from the reviews after removing the stop words, tokenizing, and lemmatizing the sentences. The new columns are created for the most frequent words which can impact the hotel’s rating. The few most frequent bi-gram words used in positive and negative reviews are ‘room small’, ‘hotel great’, ‘perfect location’ etc.

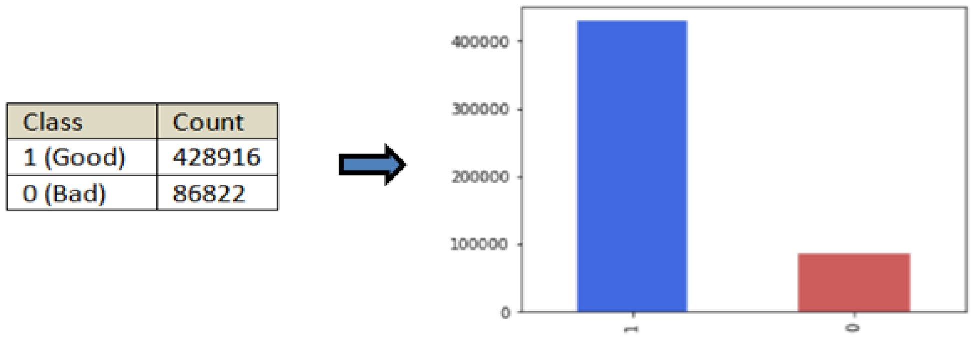
- When selecting any dependent variable it is vital to understand the effect of features (reviews, country, etc.) on review score. In the given dataset, it is found that the range of review scores varies from 1 to 10 (include real number) having 37 distinct values. As these 37 values are not uniformly divided, there will be biases associated with these classes.
  Hence the new variable ‘Review Status’ has been created with two classes Good and Bad based on the review score. If Review Score $>$ 6.7, it is considered as ’Good’ otherwise ’Bad’ reviews. The threshold value ‘6.7’ has been chosen after analyzing all the positive and negative frequent words from the dataset for every review. It was found that scores less than ‘6.7’ have more negative keywords as compared to positive.

- After cleaning the text and merging positive and negative reviews, we found a new feature having a number of words count in the overall cleaned reviews.

3.1.5 Imbalance in Data

In order to a train model, the dataset should be a balanced one which means that class proportion should not be skewed. The skewed dataset w.r.t classes introduce the classification problem[25] which should be addressed first to avoid biases towards any class. The classes which are more in number are referred to as Majority class while those are less in numbers are referred to as minority class. Hence to get rid of the model's learning problem, these classes are preferred to have less distribution gap (or comparable). As discussed in the previous section 'Review Status' was created consisting of good and bad classes based on 'Review Score'. It can be observed that these classes are not divided uniformly.
Hence different sampling techniques can be adopted to solve this issue as discussed in the literature review. Here, hybrid sampling (Oversampling and Undersampling) has been used to create a balanced dataset.

Classes before Sampling:
Total number of data records present: 515738

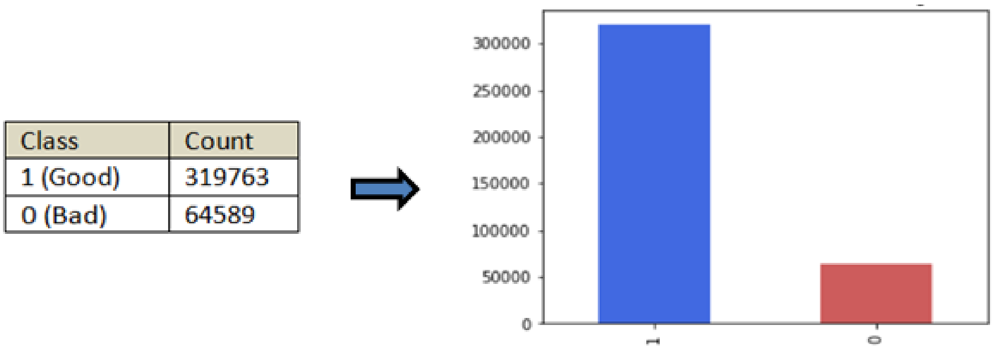


Class	Count
1 (Good)	428916
0 (Bad)	86822

Figure 3.3: Class Distribution in Imbalanced Overall Dataset before Sampling

As seen in the figure 3.4 (Imbalanced data points in the Training Set), the percentage distribution of '1' and '0' is 83% and 17% respectively which may introduce a bias towards majority classes when fed to an algorithm for training.

Total number of Training records after split: 384352



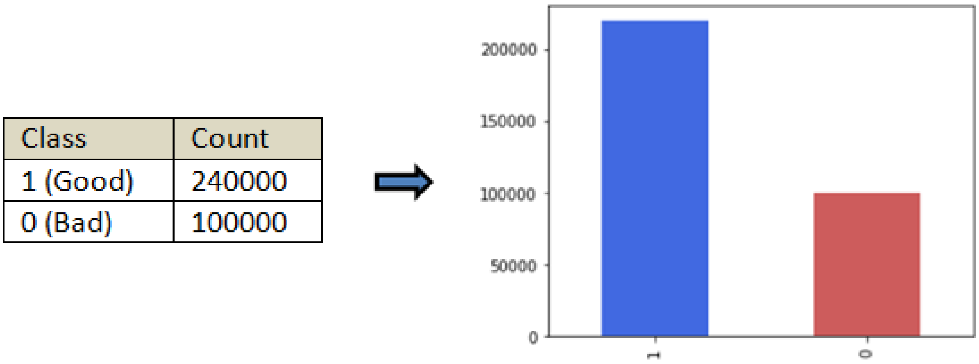
Class	Count
1 (Good)	319763
0 (Bad)	64589

Figure 3.4: Class Distribution in Imbalanced Training Set before Sampling

Classes After Sampling:

First the data points are segmented into train and test, and then the re-sampling process is applied only on training records to balance the classes. Any re-sampling was not applied on test records as these samples should remain unseen by the model before feeding for evaluation.

Total number of balanced training records: 340000



Class	Count
1 (Good)	240000
0 (Bad)	100000

Figure 3.5: Class Distribution in Balanced Training Set

The distribution gap of ‘1’ and ‘0’ classes got reduced after applying re-sampling technique. We should not make equal distribution when there is a large difference in the majority and minority classes, if done so then it may introduce multiple duplicates in the training dataset.

3.1.6 Data Splitting

During the training of any model, it is very imperative to define the training and testing set from the actual dataset. The training set is used to create a classification model, include an optimization techniques to reduce the loss and tune the parameter to increase the accuracy of the model. On the other hand, testing set is used to check the efficiency of the model on the unseen data. In this dissertation, the data ratio between the training and testing sets is taken as 0.75.

3.2 Data Pre-Processing

The pre-processing is an indispensable process before converting the text into vectors and feeding to ML algorithms for training models. There are various steps involved in this process based on the problem statement to achieve good accuracy. If any of these steps are missed out then it may hamper our efficiency in text classification, topic modeling, summarization of text, etc. Below are the various steps which are involved to yield the final text in the given hotel review dataset:

3.2.1 Stop words Removal

The stop words are frequently occurring words that do not convey any meaning and just add noise in document vectors. These words may be pronouns, articles, prepositions, etc. which should be removed from the text before processing. These words can be easily validated with a predefined set of words in a file which can help us increase the computational ability. So, the NLTK library was used which is a commonly used Python library for NLP where we can find the set of words in the corpus module. To remove stop words, first the given text will be split into words and then less important words will be removed based on the list available in the NLTK library. The 'stopwords' and 'word tokenize' functions will be imported from nltk.corpus and nltk.tokenize modules respectively and eventually, iterate all the tokenize words to compare it with words present in the stop-words module. This will remove all the stop words from the text and display the remaining word set.

3.2.2 Lowercase

The programming language used for modeling text is case sensitive towards the textual data so the words ‘And’ is considered different from the word ‘and’ due to character coding. Hence it is imperative to convert the first textual data into the lower case during pre-processing. So the textual reviews in the hotel dataset are converted into the lower case before transforming the text into a vector.

3.2.3 Punctuation

These are redundant symbols that just add up to noise during text modeling. Therefore, these symbols can be removed with the help of python built-in regular expression function termed as ‘re’. But there are cases when the absence of these punctuation changes the actual meaning like ‘U.S’ is United State where ‘re’ will evaluate ‘U.S’ as ‘us’ so care must be taken when removing the punctuation.

3.2.4 Tokenization

It is the process of segmenting the text chunk into smaller pieces known as tokens and removing certain characters at the same instant known as punctuation. So paragraphs can be segmented into sentences, various sentences into respective words, and eventually words into characters. For the hotel review dataset, the NLTK python library was used to tokenize the text based on the word. The module tokenize() and method word tokenize() were used to segment the sentences into tokens or words. Once the word tokenization is performed, the output can be fed for stemming or lemmatizing which is the next pre-processing step. There are some drawbacks encountered during the tokenization which is dealing with out-of-vocabulary words which are new words absent in the vocabulary. This can be resolved somewhat by mapping these words with UNK tokens (unknown) but still, the information is lost about the words.

3.2.5 Stemming

In the hotel reviews dataset, there are many words like ‘amuse’, ‘amusements’ etc. which basically state a similar meaning but are considered different when texts are represented in vector format. This method will help us to reduce the words present in reviews into common form ('amus') by removing the prefixes and suffixes from the actual words[26]. Hence the vector dimensions become dense which is an advantage to ML algorithms.

But there is also some downside, let's take some examples from the hotel review dataset under the Positive Reviews column which are 'beautiful location' and 'beautifully decorated hotel'. Both examples 'beautiful' and 'beautifully' signify the same context but when passed to stemmer (e.g. Porter Stemmer) yield results as stem 'beauti' which does not have any meaning.

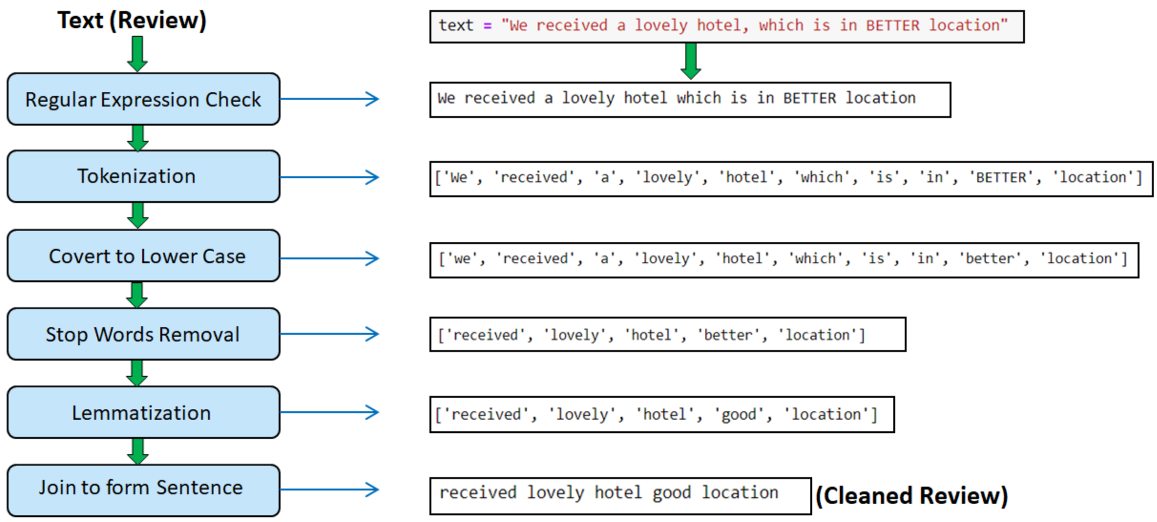
3.2.6 Lemmatization

This is also a method to reduce the dimension of a vector into dense but by taking into consideration the vocabulary of words and morphological analysis of inflected words[27]. These words are converted into dictionary root words known as a lemma. In order to implement lemmatization, we downloaded WordNet corpus, created an instance of WordNetLemmatizer(), and used lemmatize() function on these words to generate the root dictionary words. For example, the Positive Reviews has bi-grams like 'better hotel', 'best location', 'good accommodation' etc. when these words (better, best and good) are lemmatized we get the result as 'good' which has an actual dictionary meaning. In the thesis, lemmatization is used instead of stemming for the conversion of words into base form,

3.2.7 Why Lemmatization is preferred to Stemming for Hotel Reviews Analysis

As seen in the above examples, the stemming algorithm operates by removing prefix or suffix from the words while lemmatization requires linguistic knowledge where lemma created from intelligent operation conveys proper meaning. These roots (lemmas) generated are better features for training machine learning models in NLP[27] as compared to stems (generated from stemming).

3.2.8 Pre-processing Steps to Clean Textual Reviews



```
text = "We received a lovely hotel, which is in BETTER location"
```

```
We received a lovely hotel which is in BETTER location
```

```
['We', 'received', 'a', 'lovely', 'hotel', 'which', 'is', 'in', 'BETTER', 'location']
```

```
['we', 'received', 'a', 'lovely', 'hotel', 'which', 'is', 'in', 'better', 'location']
```

```
['received', 'lovely', 'hotel', 'better', 'location']
```

```
['received', 'lovely', 'hotel', 'good', 'location']
```

```
received lovely hotel good location
```

Figure 3.6: Pre-processing Steps

One of the examples was taken from the hotel review. It could be observed that all the special characters, conversion to lower letter words, unwanted removal of words and change of words to base form were performed in the flow diagram. Eventually, the tokens (words) were concatenated to form a clean sentence.

Chapter 4

Data Visualization and Analysis

In order to learn about intricate relationship, visual graphical representation plays a key role which can give insights about various features within the data. Let's visualize this hotel review dataset to understand the insights about the data.

- Countries with different numbers of reviews and their distribution on pie chart.

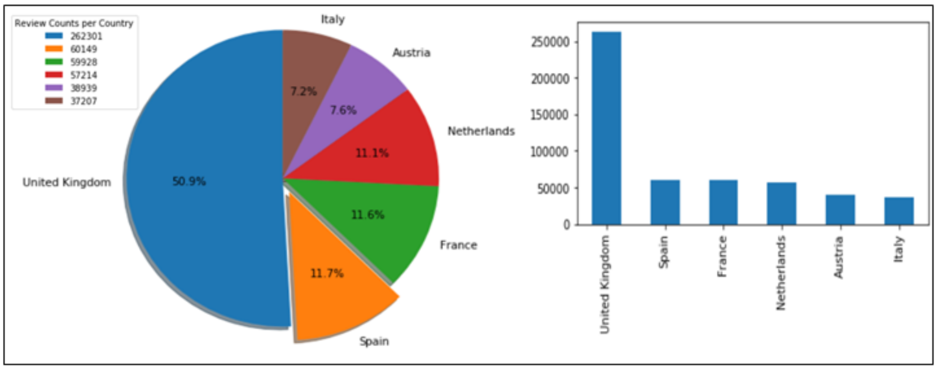


Figure 4.1: Distribution of reviews in various countries

As shown in the above charts, the reviewed hotels are located in 6 different countries that are the United Kingdom, Spain, France, Netherlands, Austria, and Italy. These countries have been extracted from the data present in hotel address and almost 51% of total reviews i.e. 262301 reviews belong to the United Kingdom. It would be further observed if it is rational to include the country as one of the features.

- Reviews Distribution in top 10 Reviewer's Nationality.

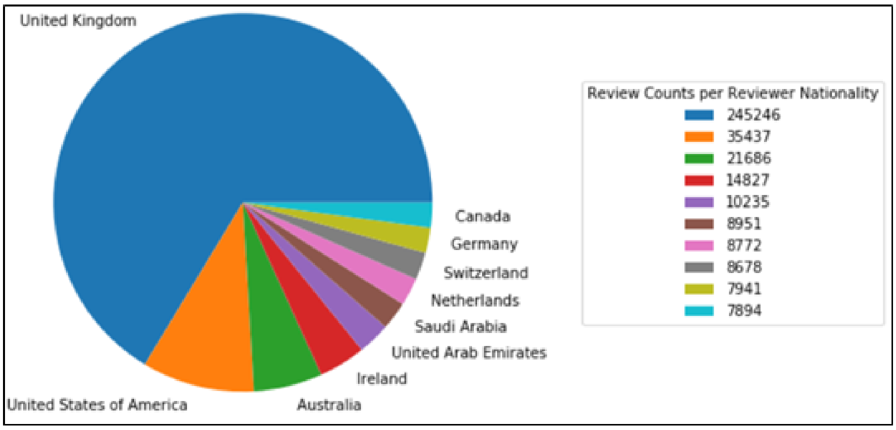


Figure 4.2: Distribution of reviews based on Nationality

It can be observed from the above chart that the majority of the tourists who reviewed these hotels were the residents of the United Kingdom followed by the USA. It may be further seen that the reviewer nationality plays any role in deciding the hotel's rating.

- Frequency of reviews posted on various dates shown in descending order.

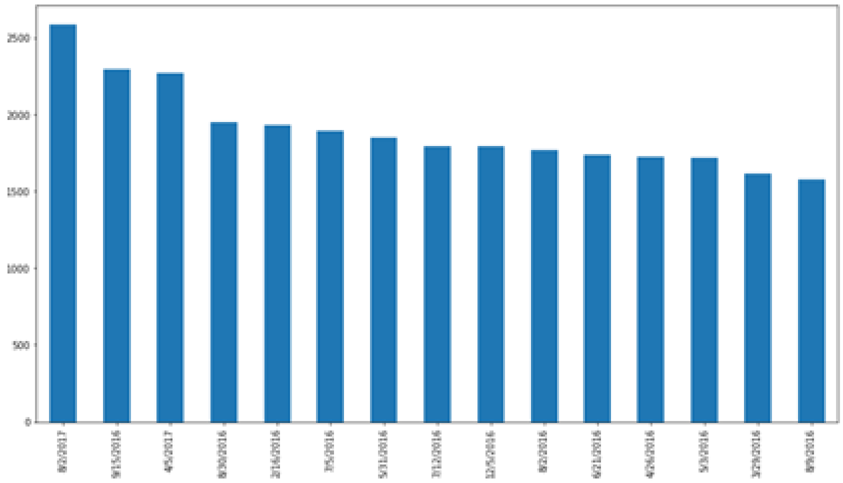


Figure 4.3: Reviews Count submitted on various dates

We can see that reviews were submitted maximum on 8/2/2017 - count is 2585.

- Below are trends of most rated hotels, ‘Hotel Casa Camper’ shows a maximum variation with some glitches on 3/16/2016 and 1/11/2017. And review score remains almost constant for ‘Ritz Paris’. There are other factors which might impact these variation like climate, disaster, etc.

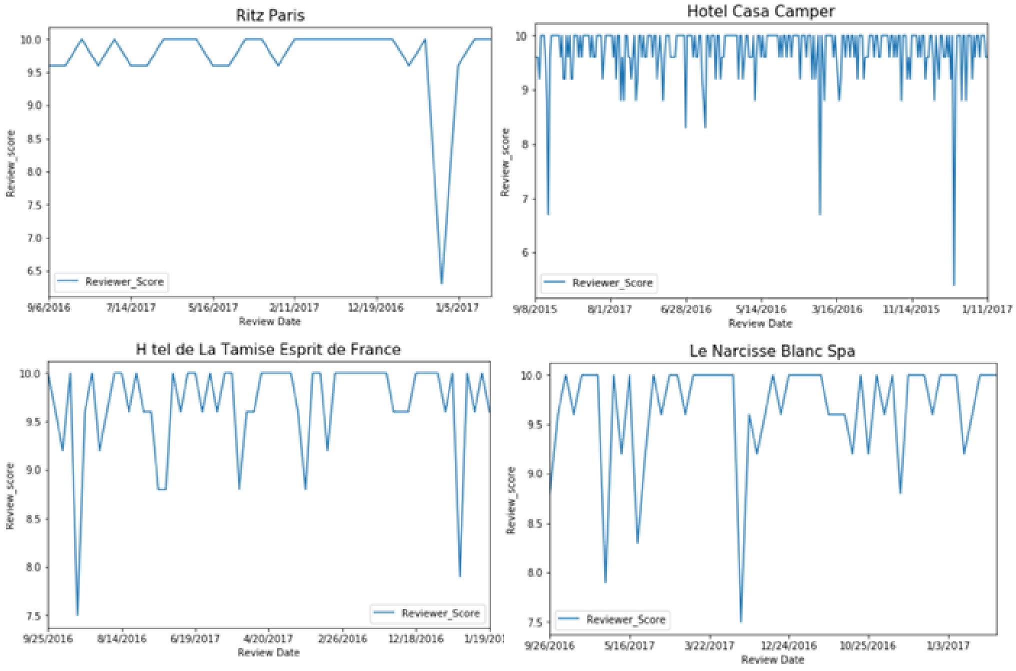


Figure 4.4: Trend in top 4 most rated Hotels in various years

- It can be noted from the below chart that the hotels ‘Park Plaza Westminster Bridge London’ and ‘Strand Palace Hotel’ have maximum good reviews and less bad reviews.

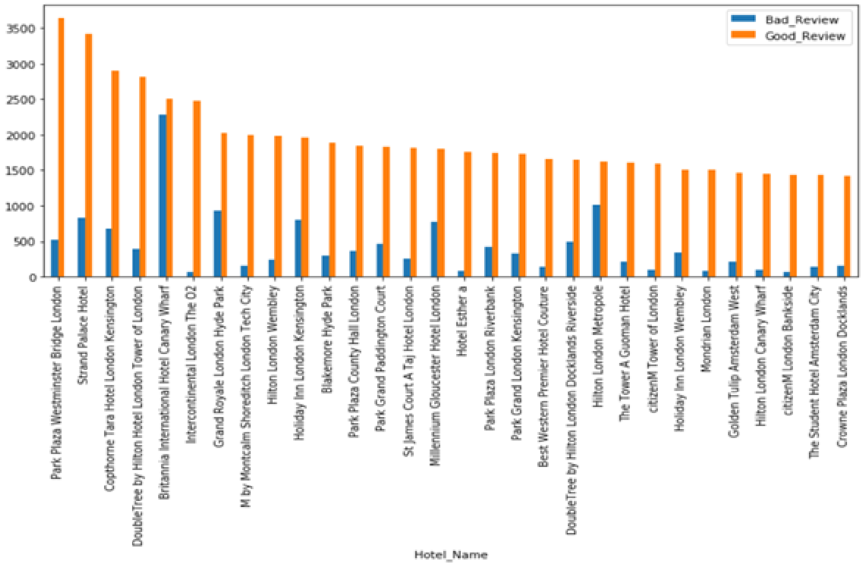


Figure 4.5: Distribution of Good and Bad Reviews in Top 30 Hotels.

In the given chart, the orange bars and blue bars represent good and bad review counts respectively.

- In order to classify reviews as good and bad, it is very important to analyse the impact of features on the target variable (here ‘Review Score). There may be various relationships between these variables like linear increasing or decreasing, exponential etc. Some features shown in the below plots signify that there are no relations with the target variable hence they will not contribute in classification and can be ignored.

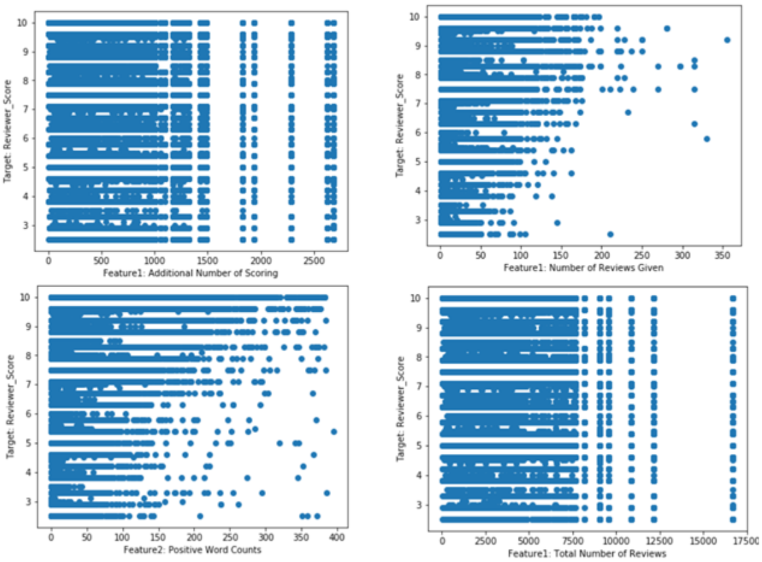


Figure 4.6: Relationship between various features and Review Score

- Relationship between the different features in the hotel review dataset.

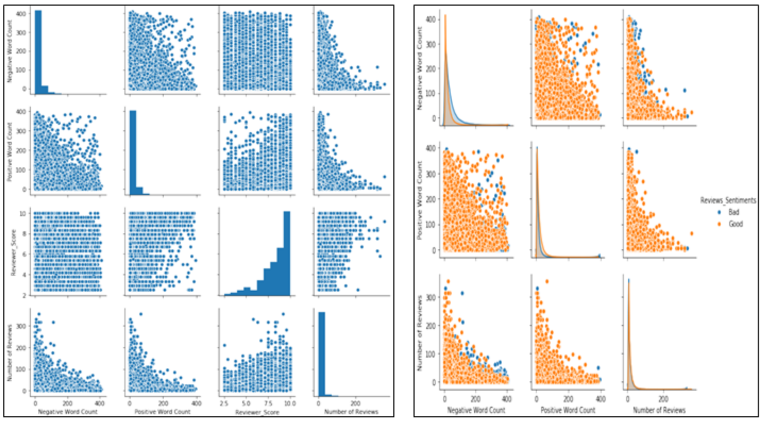
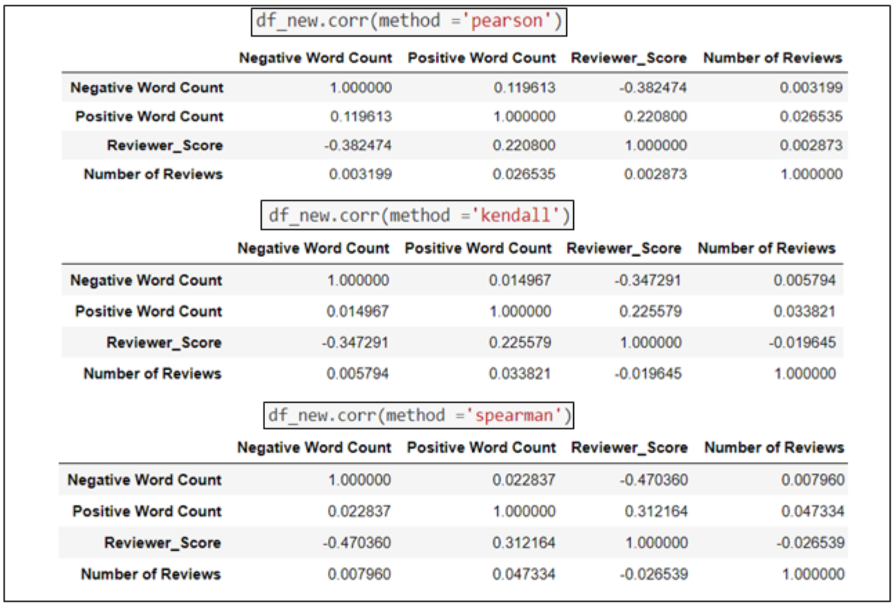


Figure 4.7: Pairwise relations between different features

Below depicts the pairwise relationship between the various features. It can be seen that there is not much correlation between the MetaData present in the hotel dataset. The relationship between features for good and bad reviews can be seen separately (in the orange chart).

- Below are the various correlation methods to find the relationship between the features.



```
df_new.corr(method ='pearson')
```

	Negative Word Count	Positive Word Count	Reviewer_Score	Number of Reviews
Negative Word Count	1.000000	0.119613	-0.382474	0.003199
Positive Word Count	0.119613	1.000000	0.220800	0.026535
Reviewer_Score	-0.382474	0.220800	1.000000	0.002873
Number of Reviews	0.003199	0.026535	0.002873	1.000000

```
df_new.corr(method ='kendall')
```

	Negative Word Count	Positive Word Count	Reviewer_Score	Number of Reviews
Negative Word Count	1.000000	0.014967	-0.347291	0.005794
Positive Word Count	0.014967	1.000000	0.225579	0.033821
Reviewer_Score	-0.347291	0.225579	1.000000	-0.019645
Number of Reviews	0.005794	0.033821	-0.019645	1.000000

```
df_new.corr(method ='spearman')
```

	Negative Word Count	Positive Word Count	Reviewer_Score	Number of Reviews
Negative Word Count	1.000000	0.022837	-0.470360	0.007960
Positive Word Count	0.022837	1.000000	0.312164	0.047334
Reviewer_Score	-0.470360	0.312164	1.000000	-0.026539
Number of Reviews	0.007960	0.047334	-0.026539	1.000000

Figure 4.8: Distribution of reviews based on Nationality

As already seen in various graphs that features (like 'Positive Word Count, the total number of reviews) will not be good predictors as there seems to exist very little relationship. Above are three correlation methods that mathematically clearly support this observation.

- The feature 'Travel reason', 'Group Type', 'Night Stayed', 'Mobile Device' has been extracted from the column 'Tags' of the hotel dataset. The graph states that more than 400000 hotel bookings were made for Leisure travel. Similarly, couples traveled together more as compared to other groups.

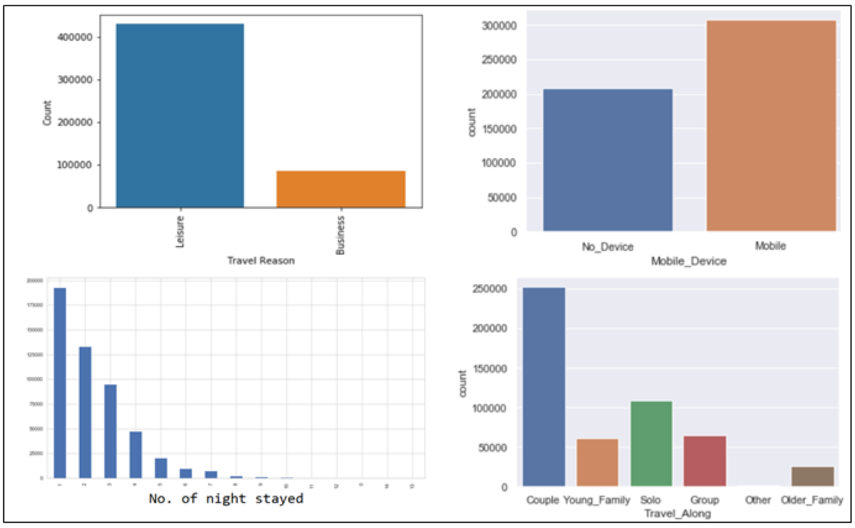
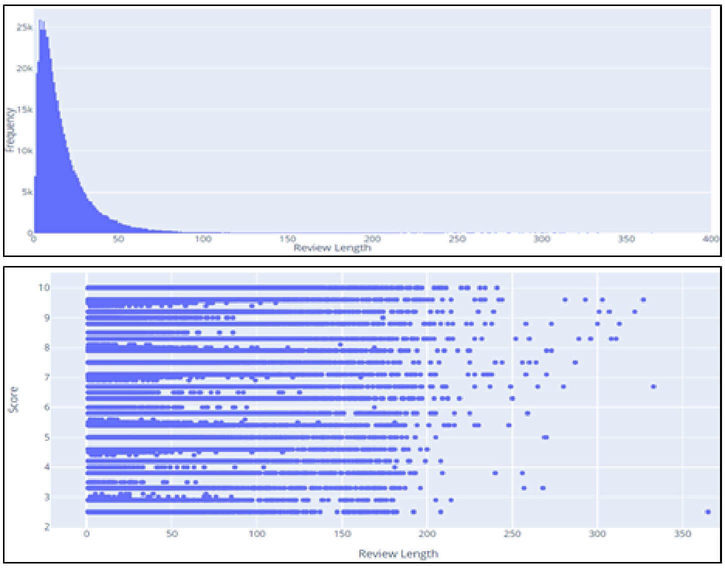


Figure 4.9: Distribution of reviews based on Nationality

- Understanding the impact of the review's length by plotting the distribution and correlating with the score. It can be observed from the histogram that the majority of reviews are short less than 50 word counts having approximate distribution is bi-modal.



```
Review Length to Rating Correlation: -0.15220115541687831
```

Figure 4.10: Review Length Distribution and correlation with Score

- Word cloud of Positive and Negative Bi-gram words in hotel reviews dataset. The words like 'friendly staff', 'good location' falls under positive word cloud while 'room small', 'breakfast problem' falls under negative word cloud as shown:

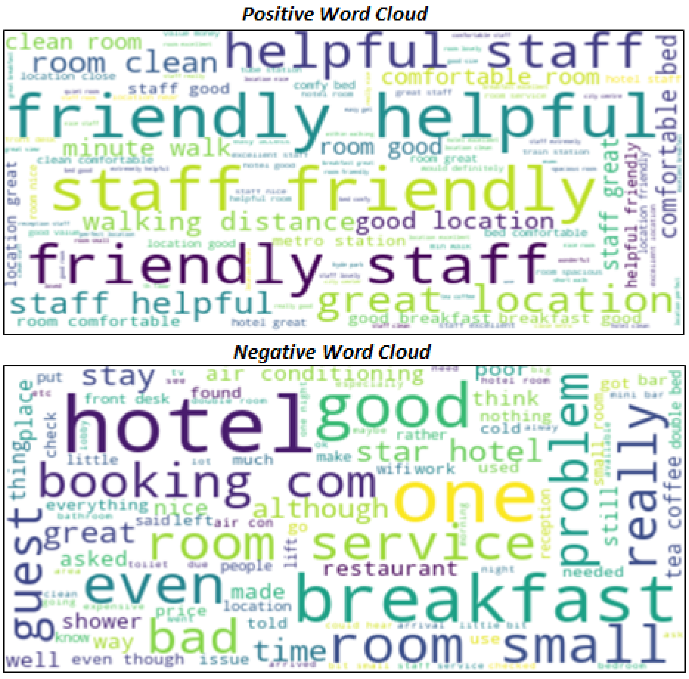


Figure 4.11: Frequent Positive and Negative Bi-gram words

- The picture below consist of different charts like density plot of sentiments, room small, hotel state and location view plotted w.r.t good and bad reviews. It seems the 'Good' review curve is shifted more towards sentiments having values greater than 0.5:

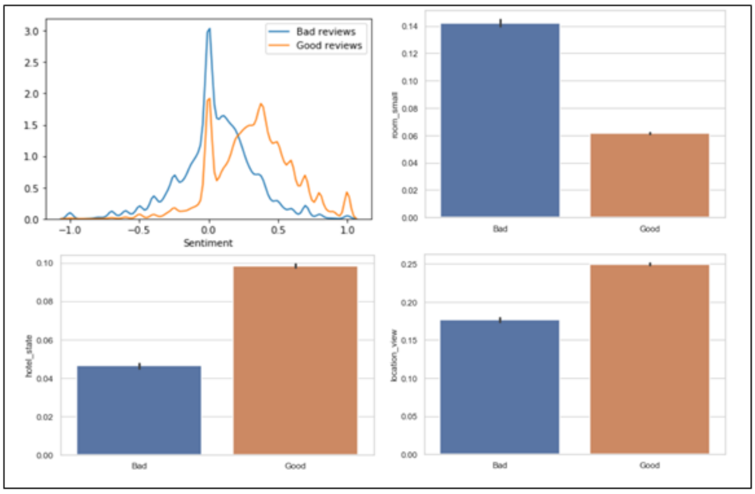


Figure 4.12: Extracted new features plots based on Good and Bad reviews

Chapter 5

Methodology

5.1 Experiment1: Text Classification with TF-IDF using traditional ML methods

In order to perform the text classification, text summarization, and content-based recommender system, there are various techniques like Bag of Words, TFIDF, Word Embedding, etc. involved, which represent the text into a vector as ML algorithms cannot be directly applied to the text. The ML algorithms (Random Forest, Decision Tree classifiers, etc.) are used to train the models to achieve text classification. We will discuss various NLP techniques and training used in this experiment.

5.1.1 Bag of Words

In simple terms, it is a technique to extract the features from the text and represent these features in vector form. It measures the occurrence of each word in any document and interprets each word as features[28]. We have considered the bi-gram modeling approach in the thesis where the sequence of two consecutive words in any document is selected as a feature. In the experiments presented in paper[29], it has been proved that bi-gram can provide accurate features set, evaluated using information gain metrics and it is successful in raising the F1 measures. In the experiment, texts are classified in good or bad reviews hence bi-gram words like 'good hotel', 'bad location', 'small room', 'no breakfast' etc. play a very imperative role as compared to other n-gram

words. Once the occurrence of each word is captured, the sparse vector is generated which is equal to the number of words present in any document or text. This vector is known as sparse because most of its elements are zero, in the case of a large dataset the model may become worse after training.
There are few disadvantages like the order of the words and grammar is not validated, the absence of linguistic meaning, and the presence of trivial words (Stop words) create noise during analysis. These problems can be easily handled with the advent of the TF-IDF technique which will be used in our Experiment 1 instead of BOW.

5.1.2 TF-IDF Vectorization

This technique measures the importance of words in any document by computing weights of each word. It re-scales the frequency of words based on their occurrence in all the texts[30]. So the words like 'the', 'an' etc. does not convey much information but they do appear quite often, hence these words are penalised by this approach.

Term Frequency: It is a row count of words present in any document represented by tf(t,d).

$$\boxed{TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ term\ in\ the\ document}} \tag{5.1}$$

Inverse Term frequency: It states that how frequent or scarce the words across the document, it is estimated by taking logarithms of division of total number of documents present by number of documents consisting of that word [31].

$$\boxed{idf(t, D) = \log \frac{N}{|\{d\epsilon D : t\epsilon d\}|}} \tag{5.2}$$

The TF-IDF is calculated by multiplying the term frequency (TF) and Inverse term frequency (IDF) Mathematically TF-IDF can be expressed as:

$$\boxed{tfidf(t, d, D) = tf(t, d) \,.\, idf(t, D)} \tag{5.3}$$

Terminologies in TF-IDF:
- t: term (word)
- d: document (set of words)
- N: count of total documents in corpus N = $\|D\|$

In our hotel review dataset, when overall reviews are created after cleaning the text (removal of stop words, tokenization, convert to lower case, lemmatization, etc.) and converting the text into vectors, it becomes really intricate to manage the huge sparse vectors as the computational task may utilize the maximum amount of system's resources (may crash depends upon system's memory). Representing all the extracted features in TF-IDF to train the ML models, can introduce overfitting which may impact the effectiveness of text classification. Hence the parameter with 'max features' = 500 is used which manages the size of the vector before feeding it to algorithms. This will select the highest 500 features based on TF-IDF value during model training.

5.1.3 Training the traditional Machine Learning models using TF-IDF

Once the reviews are cleaned using the various pre-processing techniques, they are converted into the sparse vector using the TF-IDF vectorizer function. In this experiment, we have taken the maximum number of features as '500' which means, each cleaned review will be represented into 500 different numerical features. The categorical variables are encoded using a label encoder and hybrid sampling is performed to balance the classes in the data. The new extracted features like 'Travel Reason', 'Sentiments', etc. are also added along with '500' features to train a model. There are almost 13 different classifiers used for this experiment namely Logistic Regression, Decision Tree, etc. The various evaluation metrics are used which will select the best suitable classifier for the review classification. In the next chapters, the evaluations and results are covered for this experiment.

5.2 Experiment2: Text Classification with Word Embedding using Deep Learning Networks

In this experiment, the text classification is implemented using deep learning techniques on the hotel review dataset which involve the use of densely connected neural networks. Deep Learning can employ optimum utilization of resources with less wastage of space due to text representation in dense vector instead of the sparse vector. The model is trained using Keras python library used in deep learning approaches.

5.2.1 Word Embedding

It is an efficient way to represent the text into dense vectors where similar words have the same vector representation. These word embedding has trainable parameters and can have dimensions up to 1024 based on the dataset's size (large or small dimensions). Initially, the base model for word embedding with semi-supervised learning was proposed [32] as one hidden layered model which understands the probability function for sequence of words.

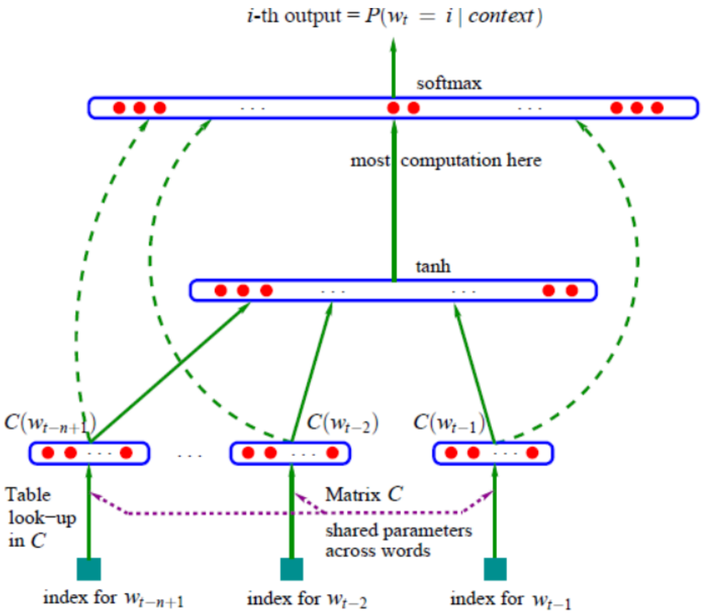


Figure 5.1: One-hidden layered architecture for word representation

There are two types of word embedding techniques Word2Vec and GloVe for generating dense vectors. The Word2Vec model uses two learning models that are CBOW model and Skip-Gram model[33]. In this experiment, GloVe (Global Vector) is used for finding vector representation of words, it is a log bi-linear regression model[34] and is defined as ratio between probabilities of tokens existing close to each other than probabilities examined separately.

The model is based on the notion that the proportion of word-to-word co-occurrence can extract important information that can further be ciphered as divergence in vector. This experiment uses the pre-trained GloVe model named ‘glove.6B.200d’ which is trained on 6 billion word tokens and consists of 200 vector dimensions. The paper [34] proposed GloVe model where **co-occurrence probability** can be defined as:

$$\boxed{p_{co}(w_k, w_i) = \frac{C(w_i, w_k)}{C(w_i)}} \tag{5.4}$$

The cost function for the GloVe model can be defined as:

$$\boxed{L_\theta = \sum_{i=1,j=1}^{V} f(C(w_i, w_j))(w_i^T \tilde{w}_j + b_i + b_j - \log C(w_i, w_j))^2} \tag{5.5}$$

$$where\, C(w_i, w_k)\; is\;\; co - occurrence\; among\; tokens\; w_i\; and\; w_k$$

5.2.2 Preparation of Embedding Layer

The Python library Keras is used which makes implementation of word embedding simple. Once the text reviews are pre-processed, these reviews are split in train and test (75%-25%) using ‘sklearn’ python library function. To find the embedded matrix using a pre-trained GloVe model, the function Tokenizer() is used from keras.preprocessing.text with the parameter ’num of words’ = 5000 (based on frequency of words most frequent 5000 words will be filtered) to generate word to index dictionary. The Xtrain consists of 250000 records and each of these records have an integer value.

The size of this list is set to a maximum ‘200’ length to capture maximum information about each record. The pad sequence has been performed to make sure that if the list has a length less than 200 then it will be padded with zeros while list having length greater will be truncated to 200. The size of the vocabulary is found to further calculate the embedded matrix, vocabulary size of pre-processed reviews was found to be '41277' which states that there are '41277' unique words in the hotel review corpus. Furthermore, the GloVe pre-trained model 'glove.6B.200d' is used to generate the dictionary which consists of words as keys while embedding list as respective values. The embedding matrix formed has dimensions as (41277, 200) having a list of all the words with their respective embedding.

5.2.3 Training Deep Learning Model using Recurrent Network (LSTM)

In order to train a deep learning model, the functional API is used from the Keras python library. It is important to note that Keras model can be created using both Sequential and Functional API. But with Functional API a model for multiple inputs (text reviews and other Metadata) can be created which we would be seen later in this chapter.

The three different models have been developed based on the features given in the hotel review dataset. The comparison between the metrics (losses, accuracy, precision and recall) will be discussed in detail for these models under the Evaluation section. The reviews will be classified using deep learning methods into two classes (that is Good and Bad Reviews). The variable inputs have been used to train three different deep learning models as discussed below:

1. **Text Classification using only pre-processed review texts.**
   The input layer known as the embedding layer is created with vocabulary size '41277' consisting of an embedded matrix, followed by single LSTM layers with 128 units and eventually output layer. This output layer is also called a dense layer with two neurons where 'soft-max' is used as an activation function. The textual reviews are to be classified into two classes (Good or Bad), hence 2 neurons in the final layer. The parameters passed during model compilation are loss function as 'categorical cross-entropy' and optimization function as 'Adam' to minimize the cost function. The model developed is fit on the training set with batch size = 128 and epochs = 22. The binary cross-entropy can be used

instead of categorical cross-entropy as a loss function due to the presence of only two classes.

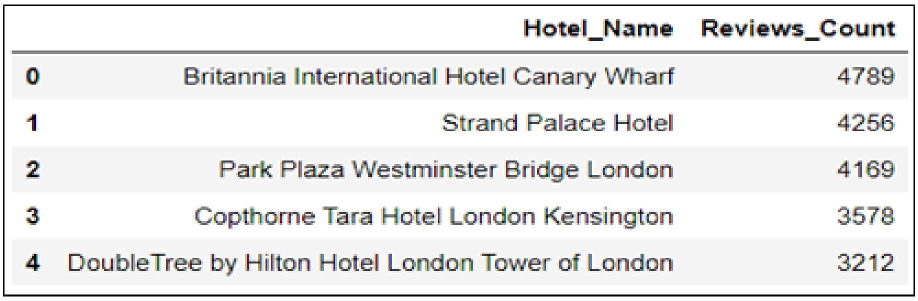
2. **Text Classification using only Metadata (sentiments, Travel Reason, etc.).**
   The single input layer with 9 neurons (as 9 features to be passed to model) is created, followed by two dense layers having 10 neurons each with activation function as 'relu'. The final layer is an output layer with soft-max function and 2 neurons based on two classes (Good and Bad). The loss and optimization functions used are 'categorical cross-entropy' and 'Adam' respectively. In order to fit the model, the parameters used are batch size = 128 and epochs = 22. Hence using evaluate function, the loss and accuracy of the model are computed. The 9 features which have been used as an input are : 'Country Name', 'Travel Reason', 'Travel Along', 'Stay Days', 'room small', 'hotel state', 'location view', 'Sentiment' and 'num of words'.

3. **Text Classification using pre-processed reviews and Metadata together.**
   It is considered that using more features as an input to the model may enhance accuracy. So the overall review is combined with 9 other features to learn the changes in loss and enhancement in the accuracy. The functional API from Keras library has been used to create a model consisting of a single input layer with '9' number of neurons (i.e.Meta data), one embedded layer with size '200' (text reviews), followed by an LSTM layer with 128 neurons, two dense layers having 10 neurons each with 'relu' as an activation function. The two layers 'text reviews' and 'MetaData' are concatenated to form the dense layer with 10 neurons and eventually the output layers with 'softmax' as an activation function with 2 neurons. The loss and optimization functions used are the same that is 'categorical cross-entropy' and 'Adam' respectively. The total features used are 10 namely: 'Cleaned Review', 'Country Name', 'Travel Reason', 'Travel Along', 'Stay Days', 'room small', 'hotel state', 'location view', 'Sentiment' and 'num of words'.

The discussion regarding the comparison in results will be discussed in the further chapter under the Evaluation section.

5.3 Extractive Review Summarization

In the hotel dataset, there are around 515000 reviews given by the customers where each record has both positive and negative reviews. There are around 1493 unique hotels in the given dataset consisting of multiple reviews for every hotel, below are some of the reviews count for 5 hotels. As can be seen from the table that reviews submitted for any hotel is large in number, so it is very difficult for an interested customer to read all the reviews about any hotels which may consist of a different point of view based on the user's experience. Hence it becomes really important for any customer to understand most of the information in a few lines as compared to many. This objective can be achieved with the use of a concept known as Text Summarization where short and coherent reviews can be extracted from the larger textual reviews based on the rank of importance. The customers have been given the flexibility to choose as many lines of reviews as they want to read for any particular hotel. Hence, at a glance, the customers can decide about the good and bad aspects of any hotel based on the summary extracted from the positive and negative reviews.



	Hotel_Name	Reviews_Count
0	Britannia International Hotel Canary Wharf	4789
1	Strand Palace Hotel	4256
2	Park Plaza Westminster Bridge London	4169
3	Copthorne Tara Hotel London Kensington	3578
4	DoubleTree by Hilton Hotel London Tower of London	3212

Figure 5.2: Top 5 hotels having maximum reviews

Basically in two ways we can summarize our reviews as discussed in paper [35]:

1. **Extractive Approach**: The reviews are summarized based on the ranking of the sentences where important and relevant sentences are given priority during the extraction of reviews.

2. **Abstractive Approach**: The important features or information from the reviews are extracted and presented in a new way rather than the selection of

sentences from reviews.

Extractive Review Summarization using Text Rank Algorithm

So in the thesis, the Extractive approach has been employed to filter out the top 'n' important sentences from the hotel reviews. The value of 'n' depends upon the customer so that this value can be changed based on the requirement.

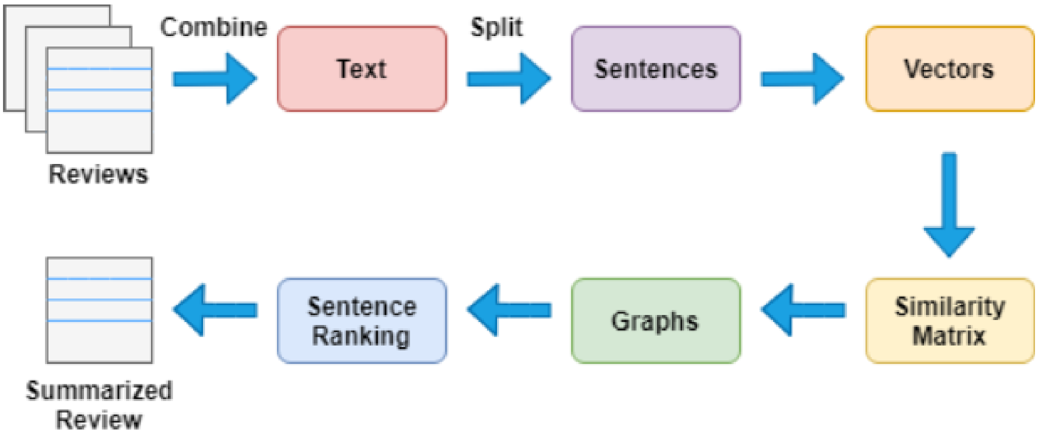


Figure 5.3: Steps involved in Review Summarization

Below are steps to produce review summarization:

1. The reviews from datasets are cleaned using NLTK library (stop words removal, lower case, etc.) and read the text, now vectors for two sentences to be compared are formed.

2. The Cosine similarity values for each pair are calculated using cosine distance function under the class 'nltk.cluster.util' and Similarity matrix is generated.

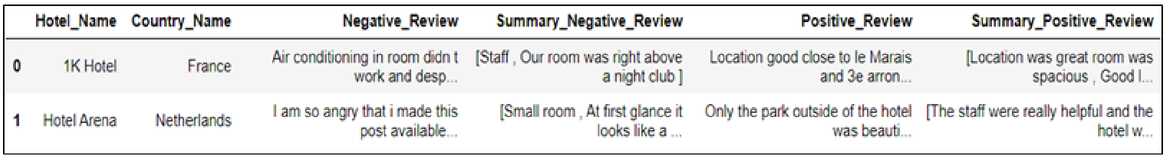
3. In order to calculate the rank, the matrix is transformed into a graph where sentences and scores are considered as vertices and edges respectively.

4. Once the graph is generated across the sentences, the ranking of sentences is performed using the function 'pagerank' under the library 'networkx'.

5. Now, the value of 'n' is defined which will represent the top 'n' sentences in reviews for any hotel. Using function sorted, the sentences are arranged in descending order based on the similarity scores.

6. Eventually loop is used to select the top ‘n’ sentences from the list of already sorted sentences and the output is displayed as shown below. It must be noted that the user can provide the list of hotels for which they want to have a review summary.

List of Hotels to be summarized: **[’Hotel Arena’,’1K Hotel’]**



	Hotel_Name	Country_Name	Negative_Review	Summary_Negative_Review	Positive_Review	Summary_Positive_Review
0	1K Hotel	France	Air conditioning in room didn t work and desp...	[Staff , Our room was right above a night club]	Location good close to le Marais and 3e arron...	[Location was great room was spacious , Good l...
1	Hotel Arena	Netherlands	I am so angry that i made this post available...	[Small room , At first glance it looks like a ...	Only the park outside of the hotel was beauti...	[The staff were really helpful and the hotel w...

Figure 5.4: Summarized Positive and Negative reviews for given Hotels

From the above figure, it can be seen that **Summary Negative Review and Summary Positive Review** are the summarization of the negative and positive reviews posted by the users for the two hotels ’Hotel Arena’ and ’1K Hotel’. Below mentioned are the library and techniques which are used in review summarization.

Library used: **NLTK**, it is written in python used to perform various NLP tasks.

Methods used:
Sentence Tokenization: In the previous chapter, tokenization process was already discussed, which is used to divide the hotel reviews (Positive and Negative) into smaller parts known as tokens. The sentence tokenization is responsible to segment the text reviews into sentences. Stop Words Removal:The stop words like ‘the’, ‘is’ etc. have been removed from the reviews which do not convey any information but just add some space and processing time instead.

Cosine Similarity: It is used to estimate the similarity between two n-dimensional vectors in space derived from the text reviews[36]. It can be interpreted as the cosine angle between the two vectors, so if the two vectors are similar then similarity value will be ‘1’ (angle will be zero). The range of cosine similarity will vary from ‘0’ to ‘1’

(-1 ruled out will be as term frequency value >0).

$$\boxed{sim(x, y) = \frac{x.y}{||x||||y||}} \tag{5.6}$$

$\|x\|$ and $\|y\|$ are the Euclidean norm of vectors x and y.

The similarity function for two sentences Si, Sj are given below. The result is a dense graph representing the various reviews as discussed in the Text Rank algorithm [37].

$$\boxed{Sim(S_i, S_j) = \frac{(w_k/w_k \, \epsilon \, S_i \; \& \; w_k \epsilon S_j)}{\log |S_i| + \log |S_j|}} \tag{5.7}$$

Cosine Distance: It can be described as below:

$$\boxed{CosineDistance = 1 - CosineSimilarity} \tag{5.8}$$

Similarity Matrix: It is a matrix of scores which represent resemblance among two sentence vectors.

5.4 Recommender System

As we know that in today's world of the internet there are huge numbers of customer responses stored on the big data about various services (hotels, flights, etc.) or products. This makes customers curious to first review and then avail any services which are quite logical. So the customer will have some requirements based on which he might intend to know all the information present online about that service or product as discussed in reviews summarization. But this process may consume too much time to figure it out the intended service as the amount of data is huge, also chances of selecting the wrong hotels are also probable when reviews are manually scanned.

This can be solved with the help of a recommendation system where it will pilot the particular user in a well-directed way to intended services with possible options. This will provide the user with choices based on his requirements and hence a good amount of time will be saved with customer satisfaction. There are various types of recommender systems as discussed in the literature reviews namely: Content-based recommender system, Collaborative Filter recommender system, and Demographic based recommender system.

The similarity in the reviews submitted by the customers is compared, hence the Content-based recommender systems are designed which will be helpful in providing the users about listing the hotel name with sorted scores.

5.4.1 Content-Based Recommender Systems

This method uses the attributes present in the dataset to recommend any hotel based on user preference. The preferences of the users are extracted from the raw data presented in the hotel dataset.

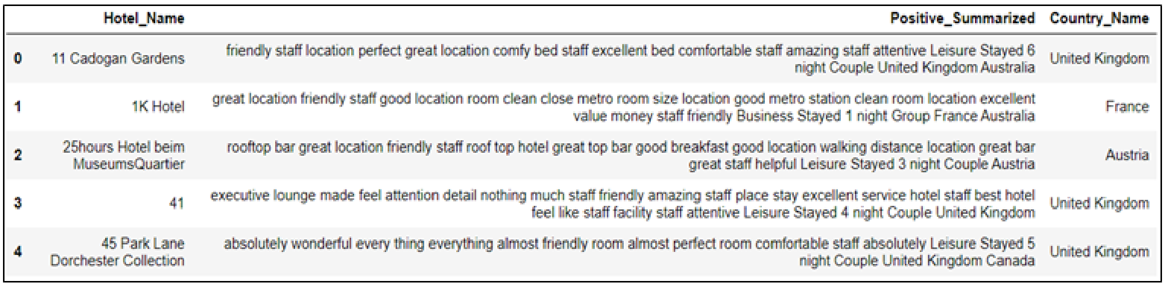
Python libraries used:

Numpy, pandas, and NLTK

Implementation

The features positive reviews, Tags, country name, and hotel name have been extracted from the hotel review dataset. These reviews are further cleaned where stop words removal, lemmatization, lower case, removal of other symbols are done. Once the reviews are cleaned, the approach of bi-gram words extraction is performed where high frequency words are chosen from the reviews. All reviews are concatenated based on the hotel name and data records of 1493 (unique hotels) got created.

Now the information from the column 'Tags' is extracted and appended with cleaned texts consisting of positive review's keywords (bi-gram words). The information present in the 'Tags' is about the customer's travel reason, with whom they traveled and the number of nights stayed. The country name and the nationality of the reviewer are also appended with positive keywords and Tags. The Cosine Similarity and other Python methods are used to compare different vectors with input string (preferred by the customer) while building the system. The system will eventually recommend the top five matching hotels with their respective scores. Below are few records of final data formed after pre-processing, consisting of 1493 unique hotels.



	Hotel_Name	Positive_Summarized	Country_Name
0	11 Cadogan Gardens	friendly staff location perfect great location comfy bed staff excellent bed comfortable staff amazing staff attentive Leisure Stayed 6 night Couple United Kingdom Australia	United Kingdom
1	1K Hotel	great location friendly staff good location room clean close metro room size location good metro station clean room location excellent value money staff friendly Business Stayed 1 night Group France Australia	France
2	25hours Hotel beim MuseumsQuartier	rooftop bar great location friendly staff roof top hotel great top bar good breakfast good location walking distance location great bar great staff helpful Leisure Stayed 3 night Couple Austria	Austria
3	41	executive lounge made feel attention detail nothing much staff friendly amazing staff place stay excellent service hotel staff best hotel feel like staff facility staff attentive Leisure Stayed 4 night Couple United Kingdom	United Kingdom
4	45 Park Lane Dorchester Collection	absolutely wonderful every thing everything almost friendly room almost perfect room comfortable staff absolutely Leisure Stayed 5 night Couple United Kingdom Canada	United Kingdom

Figure 5.5: Pre-processed data for Recommender System

Input string to be compared is in format: *'Keywords'* + *'Tags'* + *'Country Name'* Where Tags = *'Leisure/Business'*, *'Couple/Solo/Young Family/Old Family'*, *'Number of to be Night Stayed'*

Once the data is prepared for the systems, the recommender system will estimate the cosine similarity between the reviews in RS and eventually return the top 5 hotels with maximum similarity scores .We have already discussed the Cosine Similarity in the previous section in detail. Below diagram will represent the similarity between the two vectors in the 3-dimensional space.

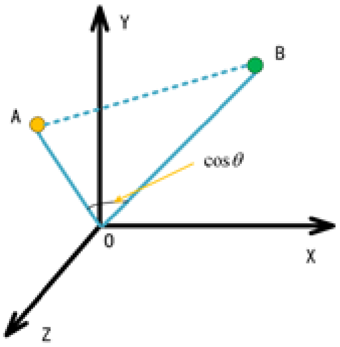


Figure 5.6: Cosine Similarity between A and B vectors in 3D space

The function *cosine similarity* will find the similarity matrix of the two texts i.e. positive bi-gram words with Tags *(keywords + Tags + Country name)* and input string (provided by incoming customer). The two texts will be represented in the form of vectors in word space, the words will be compared in both the reviews so if any of them does not match with other then the value for that index is calculated as zero value in the similarity matrix.

Steps included in working of function 'cosine similarity':

- It will parse two texts [text1 = (keywords + Tags + Country name) and text 2 = input string]
- Find the words with the help of regular expression
- Estimate the frequency of each word using the function Counter where keys = word and value = frequency of that word
- Estimate the similar words in both the vectors.
- Compute the cosine similarity for these vectors using below formula:

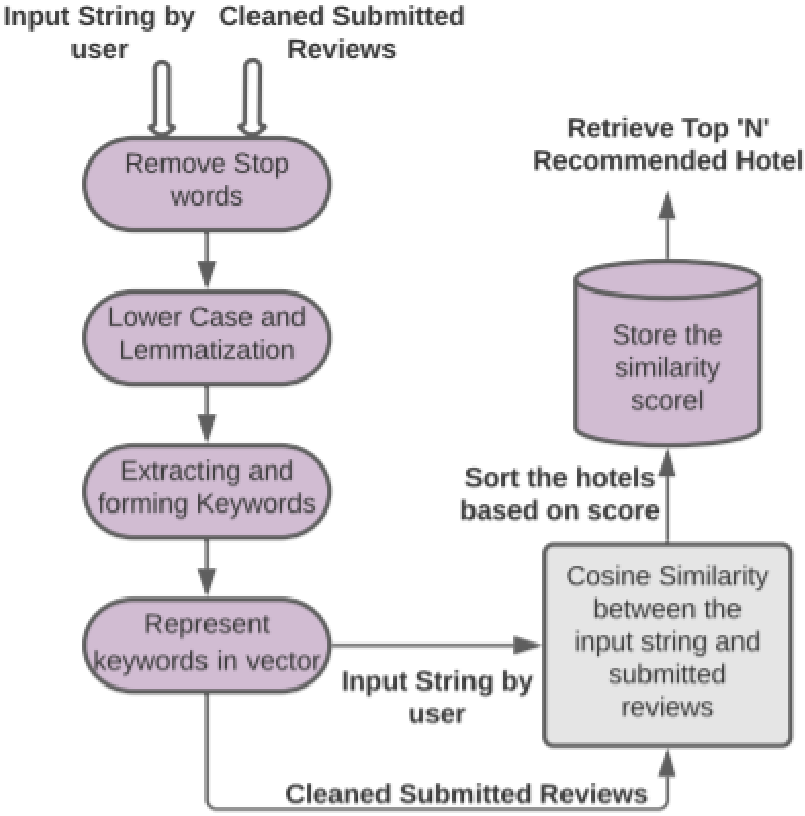

Input String by user
Cleaned Submitted Reviews
Remove Stop words
Lower Case and Lemmatization
Extracting and forming Keywords
Represent keywords in vector
Input String by user
Cleaned Submitted Reviews
Cosine Similarity between the input string and submitted reviews
Sort the hotels based on score
Store the similarity scorel
Retrieve Top 'N' Recommended Hotel

Figure 5.7: Flow Diagram of Content-Based Recommender System

While creating the recommender engine, the keywords and the text reviews (positive/negative) would be compared. **The recommendation function will perform below steps:**

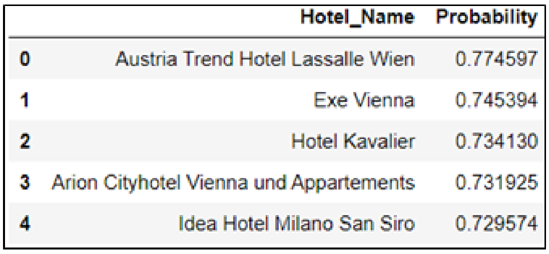
- Parse the textual reviews and find the cosine similarity value.
- Estimate the cosine similarity of each and every text with respective keywords and form a dictionary for storing.
- Once the score dictionary is created, the hotel are sorted by score and indexed using the function 'sorted'.
- Use the value of n = 5 to find and display sorted top 5 scored hotels.

What Customer Needs to Provide as a preference

The customer will provide input string consisting of positive or negative bi-gram keywords along with Tag information (Travel Reason, No.of Night stayed, Solo/Couple/Group, Country Name) as per their preference, where the recommender system will compute and display the list of hotel 'n' matching with the provided input string. The system is flexible as the customer can change the value of 'n' based on his requirement.

Experiment output when performed for input string =

"friendly staff room clean spacious room good Wifi good breakfast Business Stayed 2-night Couple Austria"



	Hotel_Name	Probability
0	Austria Trend Hotel Lassalle Wien	0.774597
1	Exe Vienna	0.745394
2	Hotel Kavalier	0.734130
3	Arion Cityhotel Vienna und Appartements	0.731925
4	Idea Hotel Milano San Siro	0.729574

Figure 5.8: Experiment output for given input string

Chapter 6

Method of Evaluation

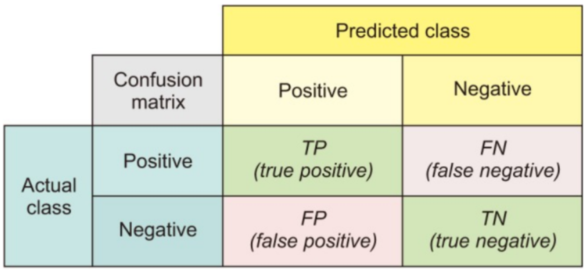
In this chapter, we will discuss in detail about various evaluation metrics and their respective implementation.

6.1 Evaluation Metrics Discussion

The evaluation of classification models is performed based on metrics like Precision, Recall, F1-Score, AUC, Confusion matrix, and Accuracy achieved on test data.

6.1.1 Confusion Matrix

It helps us to evaluate the performance of classification models created. It compares the predicted values obtained from the model with the actual values and presents the results in the N x N matrix[38].



		Predicted class	
	Confusion matrix	Positive	Negative
Actual class	Positive	*TP* *(true positive)*	*FN* *(false negative)*
	Negative	*FP* *(false positive)*	*TN* *(true negative)*

Figure 6.1: Confusion Matrix

In the case of hotel dataset, the classification is binary i.e. good and bad reviews (1 Or 0). Above shows 2 x 2 Confusion Matrices with parameters TP, FP, FN, and TN:

True Positive - TP: The actual value is Positive and the model has also predicted Positive.
True Negative - TN: The actual value is Negative and the model has also predicted Negative.
False Positive - FP: The model predicted value as positive but the actual value is negative.
False Negative - FN: The model predicted value as negative but the actual value is positive.

FP and FN are also known as Type 1 and Type 2 errors respectively. In this chapter, we have shown the results of the confusion matrix of all the machine learning algorithms.

6.1.2 Precision, Recall, AUC and Accuracy

Precision: It measures how many correctly predicted values are actually positive.

$$\boxed{Precision = TP/(TP + FP)} \tag{6.1}$$

Recall:It measures that how many of actual positive values are precisely predicted by the model.

$$\boxed{Recall = TP/(TP + FN)} \tag{6.2}$$

F1 Score:It is a weighted average of both Recall and Precision.

$$\boxed{F1Score = 2 * (Recall * Precision)/(Recall + Precision)} \tag{6.3}$$

Accuracy:It is the ratio of correctly predicted cases to the total number of cases.

$$\boxed{Accuracy = TP + TN/(TP + FP + FN + TN)} \tag{6.4}$$

ROC curve:This graph represents efficiency of the model at all classification thresholds. The plot is between the True Positive and False Positive Rate.

6.2 Implementation of Evaluation Metrics

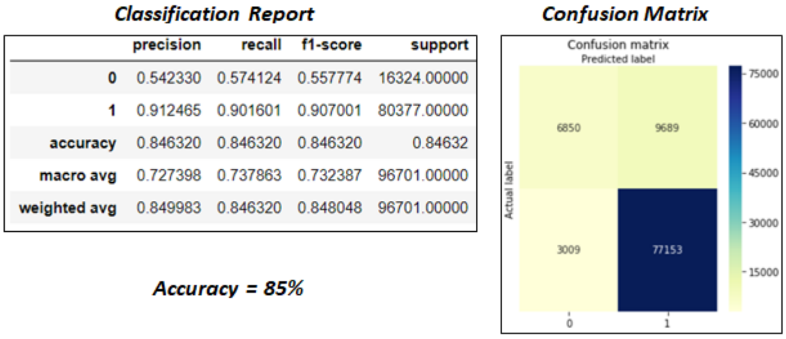
We will discuss on the evaluation performed during review classification, review summarization, and content-based filtering technique which uses a cosine similarity approach. Various machine learning and deep learning models are used for understanding which model is best suited for the hotel review dataset.

6.2.1 Traditional Machine Learning Classifiers

There are various classifiers used in the Experiment 1, the evaluation of some of the classifiers are shown below:

1. **Logistics Regression Classifier**

   The review texts are distributed into ‘Good’ (1) and ‘Bad’ (0) reviews, hence this simple classifier as a statistical technique can be used to classify the incoming reviews. The Logistic regression model is used when the target variable has discrete classes that model probability of a class, and the predictions are performed via logistic function also known as a sigmoid function[39].



Classification Report

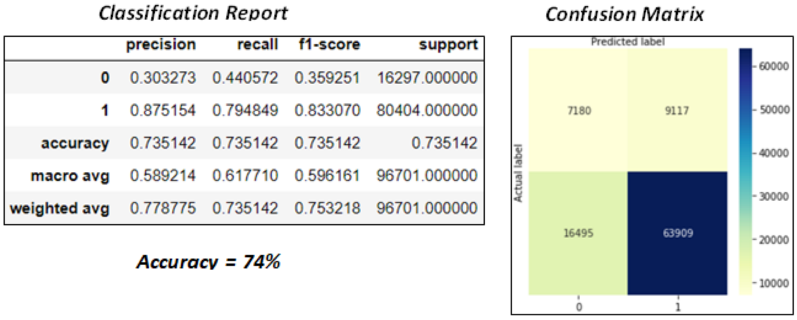
	precision	recall	f1-score	support
0	0.542330	0.574124	0.557774	16324.00000
1	0.912465	0.901601	0.907001	80377.00000
accuracy	0.846320	0.846320	0.846320	0.84632
macro avg	0.727398	0.737863	0.732387	96701.00000
weighted avg	0.849983	0.846320	0.848048	96701.00000

Accuracy = 85%

Figure 6.2: Evaluation Metrics for Logistic Classifier

2. **’K’ Nearest Neighbours Classifier**

   This algorithm is based on how close the data points are there in space. It basically calculates the distance between the data points and then classifys and groups the reviews as per the similarity.



Classification Report

	precision	recall	f1-score	support
0	0.303273	0.440572	0.359251	16297.000000
1	0.875154	0.794849	0.833070	80404.000000
accuracy	0.735142	0.735142	0.735142	0.735142
macro avg	0.589214	0.617710	0.596161	96701.000000
weighted avg	0.778775	0.735142	0.753218	96701.000000

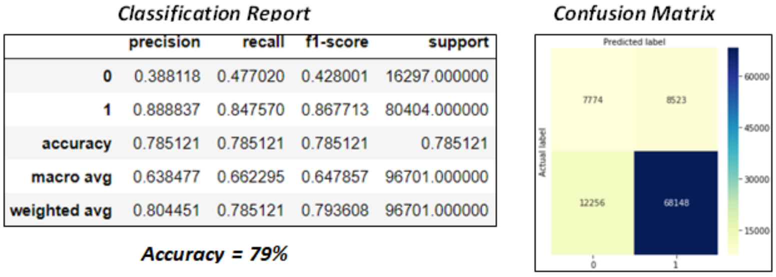
Accuracy = 74%

Confusion Matrix
Predicted label
Actual label
7180
9117
16495
63909
0
1
60000
50000
40000
30000
20000
10000

Figure 6.3: Evaluation Metrics for KNN Classifier

3. **Decision Tree Classifier**

   The data is broken down into smaller subsets (trees) consisting of decision and leaf nodes. The splitting criterion of the tree is based on 'Information Gain' or 'Gini Index'. The classification accuracy can be improved by employing a decision tree classifier as the noises and outliers in the dataset are traced and corresponding tree pruning is performed.



Classification Report

	precision	recall	f1-score	support
0	0.388118	0.477020	0.428001	16297.000000
1	0.888837	0.847570	0.867713	80404.000000
accuracy	0.785121	0.785121	0.785121	0.785121
macro avg	0.638477	0.662295	0.647857	96701.000000
weighted avg	0.804451	0.785121	0.793608	96701.000000

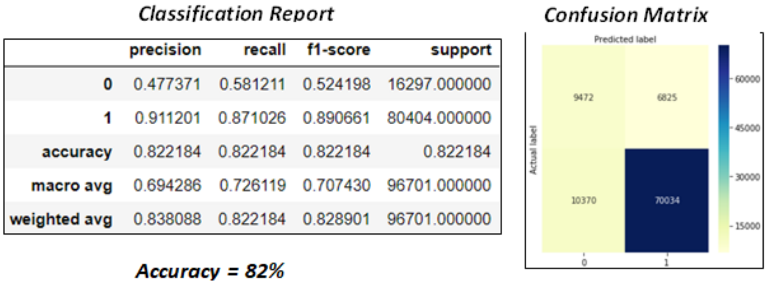
Accuracy = 79%

Confusion Matrix
Predicted label
Actual label
7774
8523
12256
68148
0
1
60000
45000
30000
15000

Figure 6.4: Evaluation Metrics for DT Classifier

4. **Random Forest Classifier**

   The random forest is based on ensemble learning, where multiple algorithms of similar type merged together like 5 Decision Trees formed on different samples merged together. As our hotel review dataset has more class '1' than '0', there may be the possibility of biasness which will be reduced by employing random forest classifier. The model has been trained using both numerical and categorical variables hence this algorithm can work well.



Classification Report

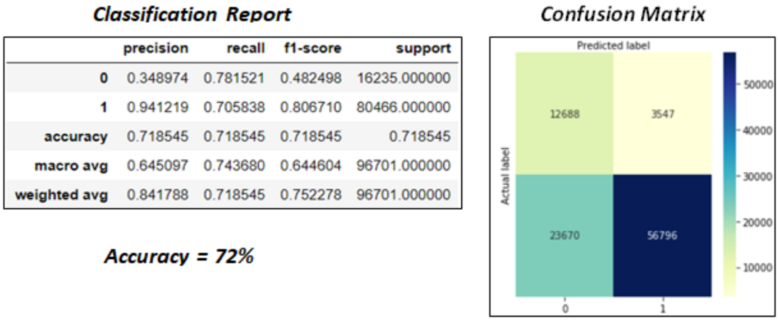
	precision	recall	f1-score	support
0	0.477371	0.581211	0.524198	16297.000000
1	0.911201	0.871026	0.890661	80404.000000
accuracy	0.822184	0.822184	0.822184	0.822184
macro avg	0.694286	0.726119	0.707430	96701.000000
weighted avg	0.838088	0.822184	0.828901	96701.000000

Accuracy = 82%

Figure 6.5: Evaluation Metrics for RF Classifier

5. **Gaussian Naive Bayes Classifier**

   Supervised machine learning model based on Naïve Bayes and Gaussian distribution, it supports features having continuous data. It basically calculates the z-score distance between the data record and class mean.



Classification Report

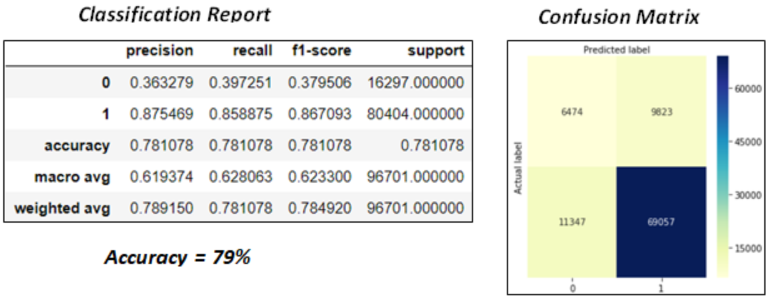
	precision	recall	f1-score	support
0	0.348974	0.781521	0.482498	16235.000000
1	0.941219	0.705838	0.806710	80466.000000
accuracy	0.718545	0.718545	0.718545	0.718545
macro avg	0.645097	0.743680	0.644604	96701.000000
weighted avg	0.841788	0.718545	0.752278	96701.000000

Accuracy = 72%

Figure 6.6: Evaluation Metrics for GB Classifier

6. **Multinomial Classifier**

   It is a classification technique that is used when there is a presence of categorical dependent features with multiple discrete levels. It is identical to the binomial logistic regression which has nominal independent variables.



Classification Report

	precision	recall	f1-score	support
0	0.363279	0.397251	0.379506	16297.000000
1	0.875469	0.858875	0.867093	80404.000000
accuracy	0.781078	0.781078	0.781078	0.781078
macro avg	0.619374	0.628063	0.623300	96701.000000
weighted avg	0.789150	0.781078	0.784920	96701.000000

Accuracy = 79%

Figure 6.7: Evaluation Metrics for Multinomial Classifier

6.2.2 Deep Learning Models using LSTM

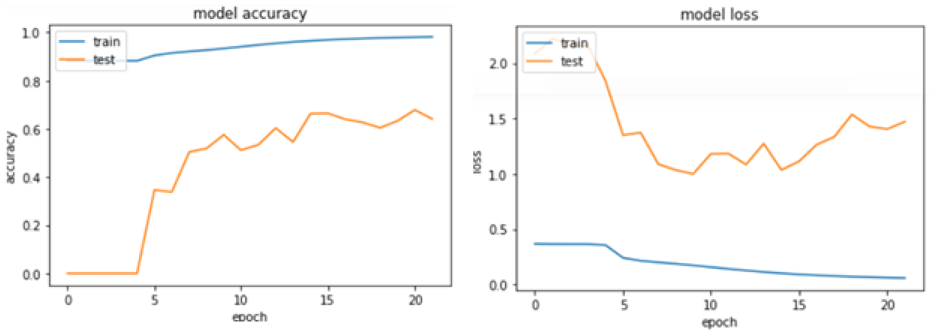
The Keras functional API has been used to create a graph of Layers, where ‘keras.layers.LSTM’ is used to create a model without making intricate changes in configurations.

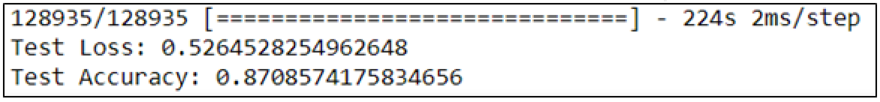
In order to reduce the vanishing gradient problem[40] as compared to other recurrent networks, LSTM is a specially designed neural network which is good at performing text sequence prediction. Hence this quality of LSTM can produce good accuracy in the text classification of hotel reviews.

Model Accuracy:
It is used to measure the performance of algorithm on training and validation sets denoted by ‘acc’ and ‘val acc’ respectively. It means how accurate the predictions are when compared to true labels.

Model Loss:
In order to train the model via deep learning method, an optimizer and loss functions are required to estimate the model error. Here, we have used optimizer=‘adam’ and loss function=‘categorical cross-entropy’.

It is observed that at every epoch there was an increase in accuracy and decrease in model loss.

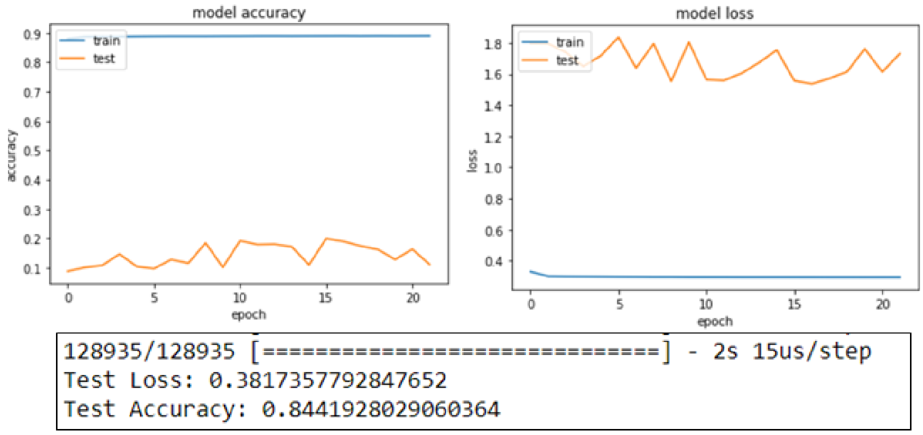




```
128935/128935 [==============================] - 224s 2ms/step
Test Loss: 0.5264528254962648
Test Accuracy: 0.8708574175834656
```

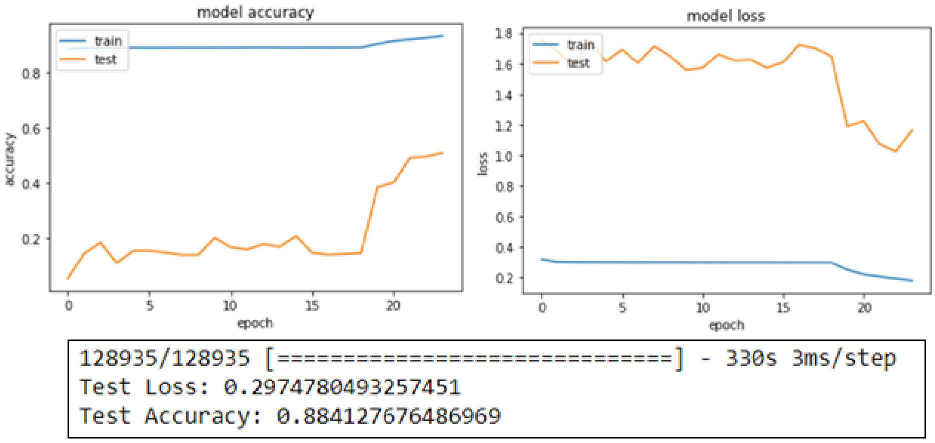
Figure 6.8: Text Classification using Cleaned reviews only trained at epoch = 22

When model is trained using the input features as Meta Data, the validation accuracy increases when epoch = 5 and then it starts decreasing as can be seen in figure 6.9.



```
128935/128935 [==============================] - 2s 15us/step
Test Loss: 0.3817357792847652
Test Accuracy: 0.8441928029060364
```

Figure 6.9: Text Classification using Meta data (Sentiments, Travel Reason etc.) only at epoch = 22



```
128935/128935 [==============================] - 330s 3ms/step
Test Loss: 0.2974780493257451
Test Accuracy: 0.884127676486969
```

Figure 6.10: Text Classification using Cleaned reviews and Meta data together at epoch = 22

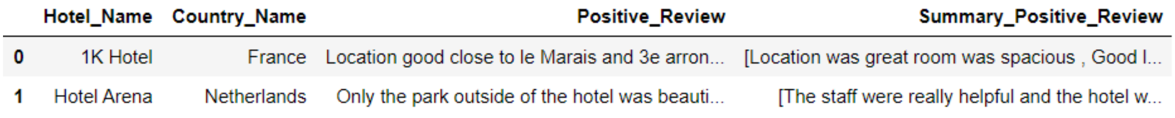
In figures 6.8 and 6.10, it can be observed that when input features were combined together (textual reviews and MetaData), the accuracy improved a little as compared to textual reviews only.

6.2.3 Evaluation of Review Summarization

The extractive summarization approach has been implemented to find the summary of positive and negative reviews in topmost 'n' sentences from the hotel dataset. The value of 'n' can be defined by the user based on how many sentences he wants to read before availing any service.

It is very important to evaluate the summarized reviews, a simple approach is applied to validate the results i.e. tallying the number of most frequent N-gram words (here bi-gram) present in reviews (positive or negative) with the keywords used in summarized sentences. This approach is similar to ROUGE-2 where bi-gram units are taken into consideration for evaluation.

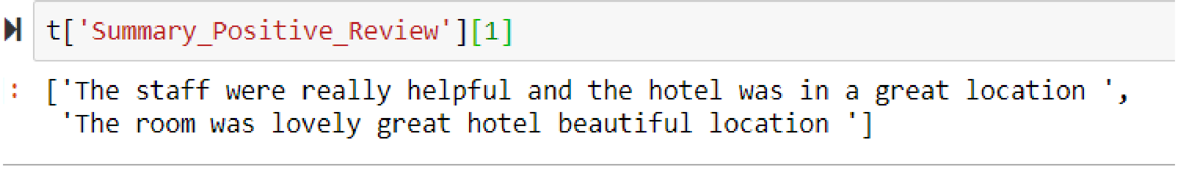
Results Obtained for positive reviews after summarization:



	Hotel_Name	Country_Name	Positive_Review	Summary_Positive_Review
0	1K Hotel	France	Location good close to le Marais and 3e arron...	[Location was great room was spacious , Good l...
1	Hotel Arena	Netherlands	Only the park outside of the hotel was beauti...	[The staff were really helpful and the hotel w...

Figure 6.11: Selected n = 2 (two important sentences)

Full summarized review of 'Hotel Arena' when n = 2:

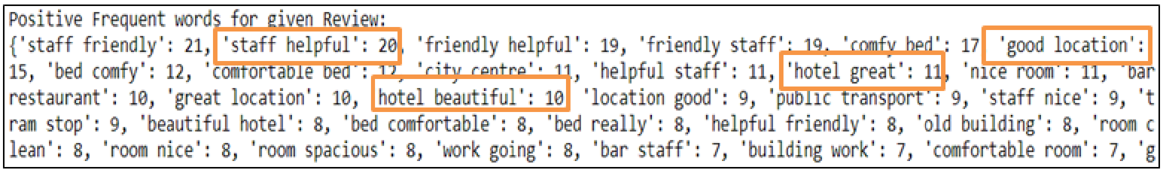


```
t['Summary_Positive_Review'][1]

: ['The staff were really helpful and the hotel was in a great location ',
   'The room was lovely great hotel beautiful location ']
```

Figure 6.12: Summarized Positive Review)

Frequency of bi-gram words applied on positive reviews for 'Hotel Arena':



```
Positive Frequent words for given Review:
{'staff friendly': 21, 'staff helpful': 20, 'friendly helpful': 19, 'friendly staff': 19, 'comfy bed': 17, 'good location':
15, 'bed comfy': 12, 'comfortable bed': 12, 'city centre': 11, 'helpful staff': 11, 'hotel great': 11, 'nice room': 11, 'bar
restaurant': 10, 'great location': 10, 'hotel beautiful': 10, 'location good': 9, 'public transport': 9, 'staff nice': 9, 't
ram stop': 9, 'beautiful hotel': 8, 'bed comfortable': 8, 'bed really': 8, 'helpful friendly': 8, 'old building': 8, 'room c
lean': 8, 'room nice': 8, 'room spacious': 8, 'work going': 8, 'bar staff': 7, 'building work': 7, 'comfortable room': 7, 'g
```

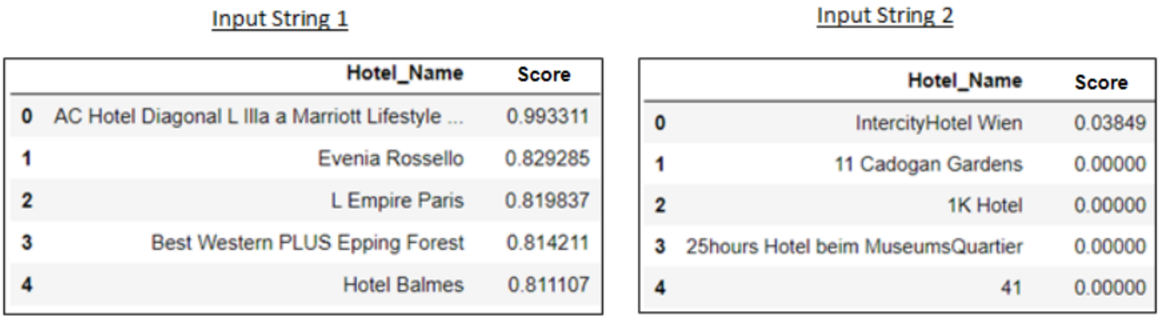
Figure 6.13: Frequent Bi-gram Positive words

It can be observed that the number of frequent bi-gram words used in reviews for 'Hotel Arena' lies in the summarized reviews as shown in the above two figures.

There are various other approaches that can be used for evaluation of summarization results described in paper [41]. As per the researchers, QARLA [42] can be used, which estimates the quality of generated text using similarity metrics precision, recall, and grammatical distribution. One of the famous metrics is ROUGE [43]which provides the scores based on the comparison between machine-generated text and human written model. There are few more approaches like DEPEVAL, GEMS, and Factoid Score, etc. which can also be employed for this evaluation.

6.2.4 Evaluation of Recommender System

In order to evaluate the accuracy of the developed content-based recommender system, we extracted the exact text from the cleaned data and passed it as an input string, it was found that the hotel for which the text was taken has the almost similarity scores as '1'. Now the same process has been repeated for the input string which was entirely different from the existing texts for various hotels, the score value near to '0' was found when the list of hotels was displayed. This way we validated the efficiency of the recommender system. Below the two results validated for the described scenarios:



Input String 1

	Hotel_Name	Score
0	AC Hotel Diagonal L Illa a Marriott Lifestyle ...	0.993311
1	Evenia Rossello	0.829285
2	L Empire Paris	0.819837
3	Best Western PLUS Epping Forest	0.814211
4	Hotel Balmes	0.811107

Input String 2

	Hotel_Name	Score
0	IntercityHotel Wien	0.03849
1	11 Cadogan Gardens	0.00000
2	1K Hotel	0.00000
3	25hours Hotel beim MuseumsQuartier	0.00000
4	41	0.00000

Figure 6.14: List of Hotels relevant to Input Strings

Input string 1 = *"good breakfast helpful staff location excellent room spacious clean nicely business center comfortable nice coffee machine staff nice bed comfortable friendly always helpful room Business Stayed 1 night Solo Spain Andorra"* (quality about hotel service).

Input string 2 = *"ticket cheap flight on time cabin crew supportive pilot trained Romania"* (quality about flight service).

Chapter 7

Result and Discussion

In this chapter, we will discuss the results obtained by applying various NLP techniques on the hotel dataset. It includes discussion on data analysis, review classification for which the two experiments (Experiment1 and Experiment2) were conducted.Eventually, we walk-through the developed Extractive Review Summarization and Content-based Recommender System.

7.1 Data Analysis Results

- The country names were extracted from the hotel address, and later they were used as input in the recommender system to display top hotels. The countries were namely: the United Kingdom, Spain, France, Netherlands, Austria, and Italy. The hotel 'Park Plaza Westminster Bridge London' had the highest number of good reviews.

- To understand the features responsible for good hotel ratings, correlations were calculated. It was found that most of the features do not show any relation with the target variable. This was evaluated by plotting scatter plots and employing correlation methods like 'Pearson', 'Kendall', and 'Spearman'.

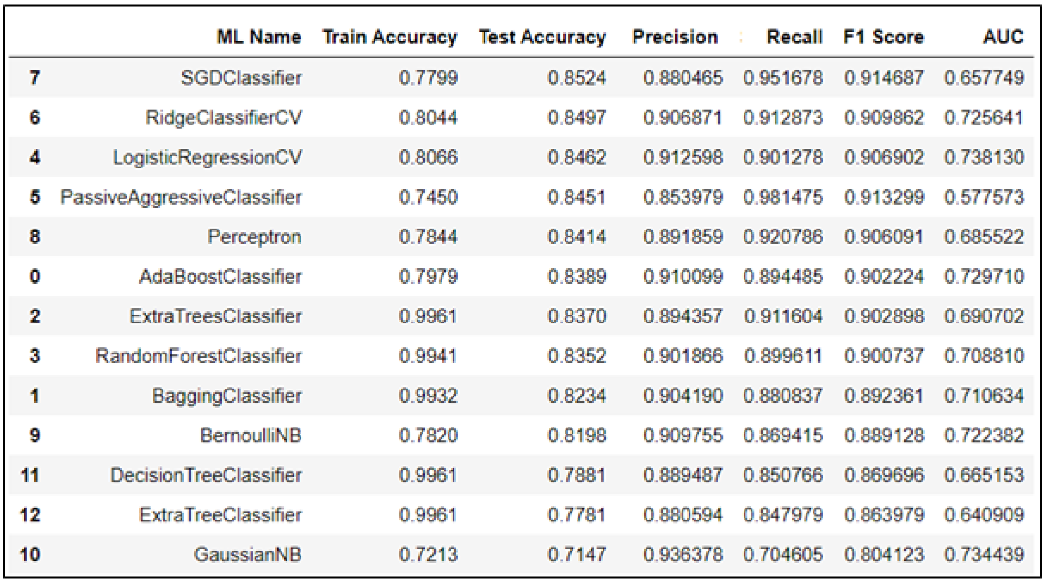
- The features like 'Trip Reason', 'Group Travel', etc. were extracted from the column 'Tags'. They were helpful in classifying reviews and developing the content based Recommender system.

- The frequent bi-gram words extracted from the reviews were also used as features in classifying reviews. Some of the bi-gram tokens used were ‘room small’ (bad reviews), ‘poor Breakfast’ (bad reviews), ‘best location’ (good reviews) , ‘staff friendly’ (good reviews) and so on.

7.2 Results from Experiment1: Text Classification of Hotel Reviews with TF-IDF Vectorization

In Experiment1, the review classification was performed using 13 different machine learning Classifiers like Logistic, Decision Tree, etc. The efficiency of these models was compared on the basis of evaluation metrics like confusion matrix, accuracy, precision, recall and others. TF-IDF vectorization was used to convert the cleaned textual reviews to a sparse vector.

It can be observed that models SGDClassifier, RidgeClassifierCV, and LogisticRegressionCV showed more desirable results but the Logistic Regression Classier was preferred because the overall values of evaluation metrics like Accuracy, AUC, and Precision were fairly better.



	ML Name	Train Accuracy	Test Accuracy	Precision	Recall	F1 Score	AUC
7	SGDClassifier	0.7799	0.8524	0.880465	0.951678	0.914687	0.657749
6	RidgeClassifierCV	0.8044	0.8497	0.906871	0.912873	0.909862	0.725641
4	LogisticRegressionCV	0.8066	0.8462	0.912598	0.901278	0.906902	0.738130
5	PassiveAggressiveClassifier	0.7450	0.8451	0.853979	0.981475	0.913299	0.577573
8	Perceptron	0.7844	0.8414	0.891859	0.920786	0.906091	0.685522
0	AdaBoostClassifier	0.7979	0.8389	0.910099	0.894485	0.902224	0.729710
2	ExtraTreesClassifier	0.9961	0.8370	0.894357	0.911604	0.902898	0.690702
3	RandomForestClassifier	0.9941	0.8352	0.901866	0.899611	0.900737	0.708810
1	BaggingClassifier	0.9932	0.8234	0.904190	0.880837	0.892361	0.710634
9	BernoulliNB	0.7820	0.8198	0.909755	0.869415	0.889128	0.722382
11	DecisionTreeClassifier	0.9961	0.7881	0.889487	0.850766	0.869696	0.665153
12	ExtraTreeClassifier	0.9961	0.7781	0.880594	0.847979	0.863979	0.640909
10	GaussianNB	0.7213	0.7147	0.936378	0.704605	0.804123	0.734439

Figure 7.1: Comparison between 13 Classifiers based on Evaluation Metrics

Graph comparison between Precision and Recall for these 13 classifiers:

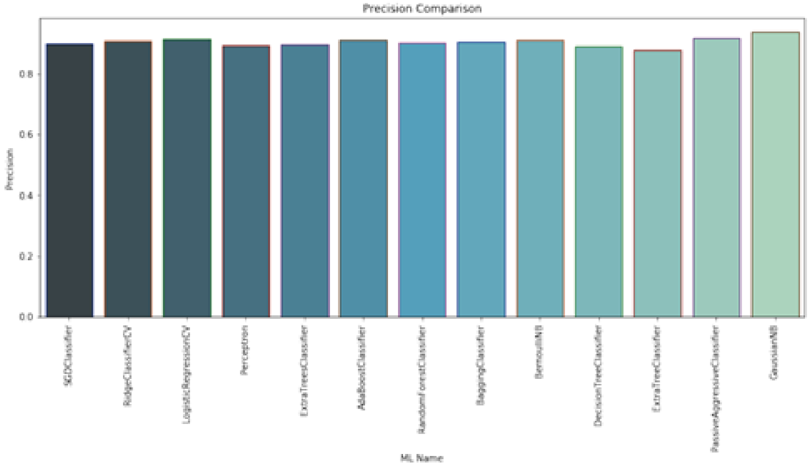


Figure 7.2: Precision Comparison

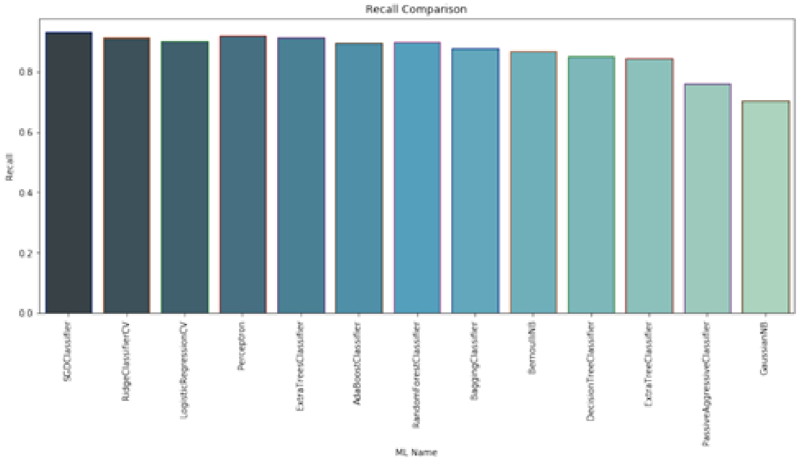


Figure 7.3: Recall Comparison

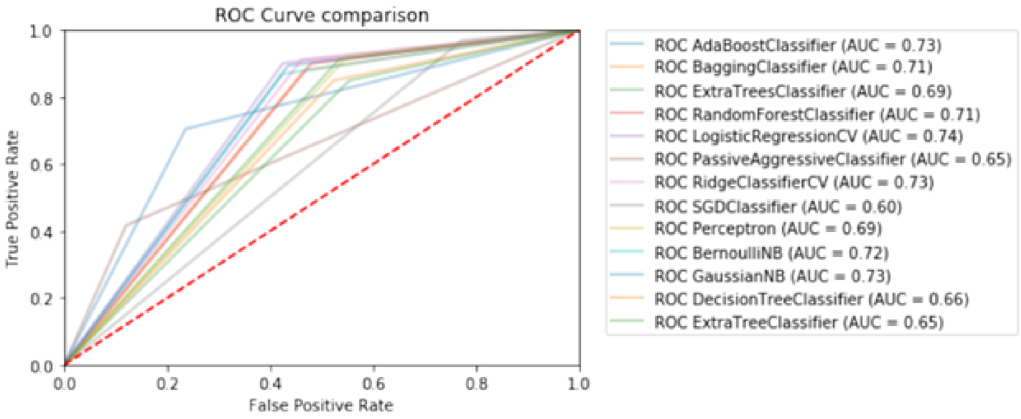


Figure 7.4: ROC Comparison

It can also be observed that the area under ROC curve is more for Logistic Regression, hence models performs better among others.

Classifier Name	Features Used	Accuracy	Precision	AUC
Logistic Regression Classifier	Cleaned Reviews, Country Name, Travel Reason, Stay Days, room small, hotel state, location view, Sentiment, num of words and Group Type	85%	'Good' - 91% 'Bad' - 55%	0.74

Table 7.1: Details of Best traditional ML model for Review Classification

7.3 Results from Experiment2: Text Classification with Word Embedding via Deep Learning

As already discussed, the intricate non-linear relationship was seen within the data. Therefore, the model trained via Deep Learning can yield a good result in review categorization as compared to the previous traditional ML approaches.

In Experiment2, the model was trained in three ways based on the input features, i.e., only text, only Meta data and combination of both text & Meta data. The GloVe embedding was used as a pre-trained model (glove.6B.200d) to generate the dense vector unlike the sparse vector in Experiment1.

From the below Table 7.2, it can be observed that the deep learning models with input features as 'Text' only and combination of ('Text' + MetaData) yielded almost similar results. The accuracy of the later is 1% more than the former.

The additional features (MetaData) can be considered depending on the resource availability, when included can consume more computational power and time than the models trained with 'Text' only feature. It can be concluded that these two DL models gave better testing results as compared to the traditional ML approaches discussed in the previous section.

Models Details	Features Used	Accuracy	Precision	Loss
Implemented with Text & MetaData	Cleaned Reviews, Country Name, Travel Reason, Stay Days, Room Small, Hotel State, Location View, Sentiment, Num of Words and Group Type	88%	'Good' - 90% 'Bad' - 75%	0.3
Implemented with Only Text	Cleaned Reviews	87%	'Good' - 90% 'Bad' - 66%	0.5
Implemented with Only MetaData	Country Name, Travel Reason, Stay Days, Room Small, Hotel State, Location View, Sentiment, Num of Words and Group Type	84%	'Good' - 85% 'Bad' - 75%	0.4

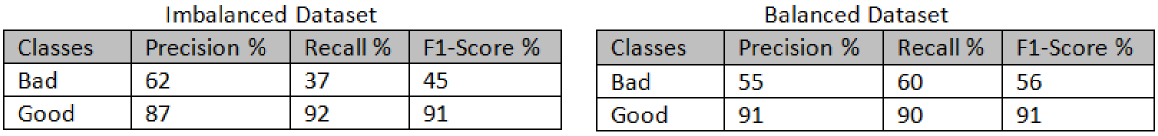
Table 7.2: Deep Learning model result with different Input Features

7.4 Importance of Sampling in Imbalanced Dataset

Imbalanced data in training dataset leads to classification problem which would have introduced errors in text categorization. The minority class would have been impacted more as compared to majority class. If trained on an imbalanced dataset, the statistical classifiers (like Logistic Regression) developed an inclination towards majority classes.

Hybrid sampling method was used to address the classification problem. The evaluation metric showed good results when Upsampling is applied to minority class and Downsampling is applied to majority class.

The Recall, F1-Score and accuracy were increased as shown below (LR classifier). *It should be noted that we did not perform any up or down sampling on the test samples as it would introduce bias.*



Imbalanced Dataset

Classes	Precision %	Recall %	F1-Score %
Bad	62	37	45
Good	87	92	91

Balanced Dataset

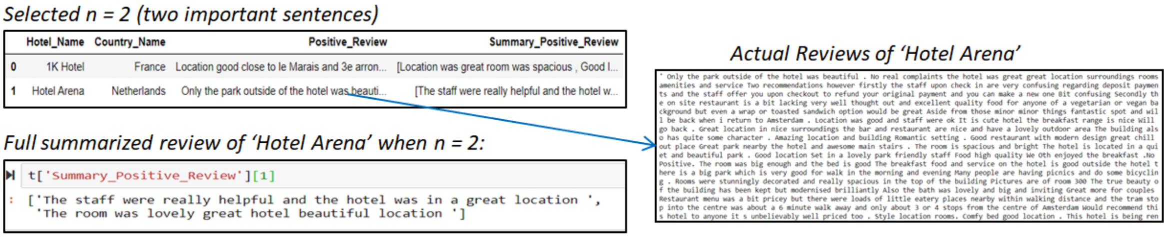
Classes	Precision %	Recall %	F1-Score %
Bad	55	60	56
Good	91	90	91

Figure 7.5: Comparison of evaluation metrics between Imbalance and Balance Datasets

7.5 Review Summarization Results

The reviews were summarized based on the cosine similarity technique which generates a similarity matrix between the different vectors. The output produced a range of values between -1 and +1, i.e., +1, if the vectors lie along the same direction and -1, if vectors lie in the exact opposite direction.

Below figures describe the 'Actual Reviews' summarized into the two most relevant sentences followed by 'Similarity Matrix' and sentences with their respective scores for 'Hotel Arena'.



Selected n = 2 (two important sentences)

	Hotel_Name	Country_Name	Positive_Review	Summary_Positive_Review
0	1K Hotel	France	Location good close to le Marais and 3e arron...	[Location was great room was spacious , Good l...
1	Hotel Arena	Netherlands	Only the park outside of the hotel was beauti...	[The staff were really helpful and the hotel w...

Full summarized review of 'Hotel Arena' when n = 2:

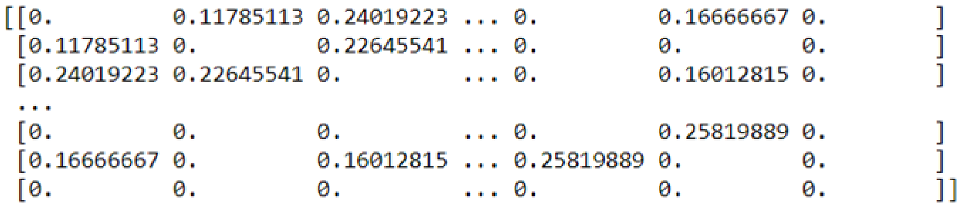
```
t['Summary_Positive_Review'][1]
: ['The staff were really helpful and the hotel was in a great location ',
   'The room was lovely great hotel beautiful location ']
```

Actual Reviews of 'Hotel Arena'

```
' Only the park outside of the hotel was beautiful . No real complaints the hotel was great great location surroundings rooms
amenities and service Two recommendations however firstly the staff upon check in are very confusing regarding deposit paymen
ts and the staff offer you upon checkout to refund your original payment and you can make a new one Bit confusing Secondly th
e on site restaurant is a bit lacking very well thought out and excellent quality food for anyone of a vegetarian or vegan ba
ckground but even a wrap or toasted sandwich option would be great Aside from those minor minor things fantastic spot and wil
l be back when i return to Amsterdam . Location was good and staff were ok It is cute hotel the breakfast range is nice Will
go back . Great location in nice surroundings the bar and restaurant are nice and have a lovely outdoor area The building als
o has quite some character . Amazing location and building Romantic setting . Good restaurant with modern design great chill
out place Great park nearby the hotel and awesome main stairs . The room is spacious and bright The hotel is located in a qui
et and beautiful park . Good location Set in a lovely park friendly staff Food high quality We Oth enjoyed the breakfast .No
Positive. The room was big enough and the bed is good The breakfast food and service on the hotel is good outside the hotel t
here is a big park which is very good for walk in the morning and evening Many people are having picnics and do some bicyclin
g . Rooms were stunningly decorated and really spacious in the top of the building Pictures are of room 300 The true beauty o
f the building has been kept but modernised brilliantly Also the bath was lovely and big and inviting Great more for couples
Restaurant menu was a bit pricey but there were loads of little eatery places nearby within walking distance and the tram sto
p into the centre was about a 6 minute walk away and only about 3 or 4 stops from the centre of Amsterdam Would recommend thi
s hotel to anyone it s unbelievably well priced too . Style location rooms. Comfy bed good location . This hotel is being ren
```

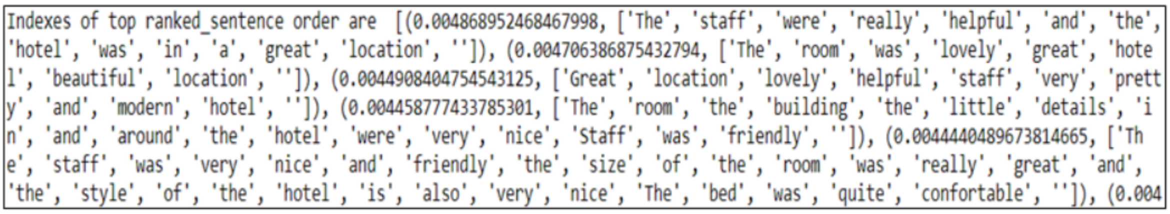
Figure 7.6: Actual Reviews and Summarized Reviews for 'Hotel Arena'

After estimating similarity, scores were calculated using 'pagerank' function. The sentences were then filtered and ranked based on their respective sentence's score as shown in the below figures.



```
[[0.         0.11785113 0.24019223 ... 0.         0.16666667 0.        ]
 [0.11785113 0.         0.22645541 ... 0.         0.         0.        ]
 [0.24019223 0.22645541 0.         ... 0.         0.16012815 0.        ]
 ...
 [0.         0.         0.         ... 0.         0.25819889 0.        ]
 [0.16666667 0.         0.16012815 ... 0.25819889 0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]]
```

Figure 7.7: Similarity Matrix of positive reviews submitted for 'Hotel Arena'



```
Indexes of top ranked_sentence order are  [(0.004868952468467998, ['The', 'staff', 'were', 'really', 'helpful', 'and', 'the',
'hotel', 'was', 'in', 'a', 'great', 'location', '']), (0.004706386875432794, ['The', 'room', 'was', 'lovely', 'great', 'hote
l', 'beautiful', 'location', '']), (0.0044908404754543125, ['Great', 'location', 'lovely', 'helpful', 'staff', 'very', 'prett
y', 'and', 'modern', 'hotel', '']), (0.004458777433785301, ['The', 'room', 'the', 'building', 'the', 'little', 'details', 'i
n', 'and', 'around', 'the', 'hotel', 'were', 'very', 'nice', 'Staff', 'was', 'friendly', '']), (0.004440489673814665, ['Th
e', 'staff', 'was', 'very', 'nice', 'and', 'friendly', 'the', 'size', 'of', 'the', 'room', 'was', 'really', 'great', 'and',
'the', 'style', 'of', 'the', 'hotel', 'is', 'also', 'very', 'nice', 'The', 'bed', 'was', 'quite', 'confortable', '']), (0.004
```

Figure 7.8: Top ranked reviews are selected based on scores for 'Hotel Arena'

On passing ‘n’ value as a parameter into the function, only top ‘n’ sentences would be selected based on the estimated sentence score.

7.6 Content Based Recommender System Results

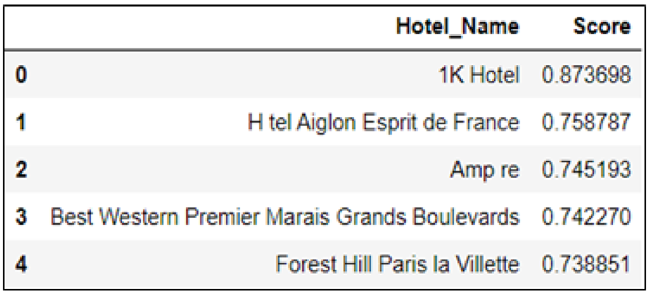
The Content-Based Recommender System was created which displayed the list of hotels in order of relevance based on user preferences (‘Keywords’+‘Tags’ + ‘Country Name’) as input. The ’Tag’ can take value as (Couple/Old/Young + Business/Leisure + No. of Nights to be Stayed).

The keywords of the input were compared with pre-processed reviews using cosine similarity. The ’Keywords’ in the input string refers to the positive bi-gram words which are supplied by the users. The comparison of input string was performed with the positive reviews present in the dataset. Moreover, the is system is flexible enough to make use of negative reviews in case the user wants to check negative aspects about a hotel.

The customer’s have a provision to select ‘n’ number of hotels based on their requirements. To observe the behavior of the recommender system, three different input strings were used, for which it yielded three different results in order of relevance.

Below are three different inputs and respective top ‘n’ hotels (n = 5):

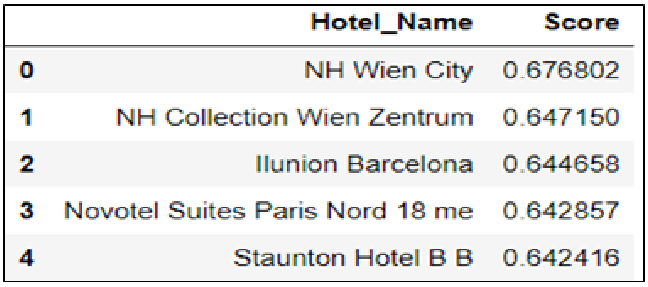
- Top 5 hotel list having **Input string 1**=*”great location helpful staff close metro room size metro station clean room friendly helpful location excellent value money Business Stayed 1-night Group France Australia”*



	Hotel_Name	Score
0	1K Hotel	0.873698
1	H tel Aiglon Esprit de France	0.758787
2	Amp re	0.745193
3	Best Western Premier Marais Grands Boulevards	0.742270
4	Forest Hill Paris la Villette	0.738851

Figure 7.9: The highly recommended hotels based on score

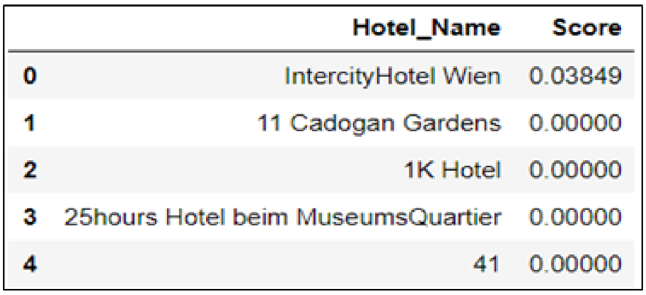
- Top 5 hotel list having fewer similarities (keywords + Tags). **Input string 2=** *"good staff clean room great location metro near gym beach view, big room breakfast United Kingdom"*



	Hotel_Name	Score
0	NH Wien City	0.676802
1	NH Collection Wien Zentrum	0.647150
2	Ilunion Barcelona	0.644658
3	Novotel Suites Paris Nord 18 me	0.642857
4	Staunton Hotel B B	0.642416

Figure 7.10: Recommended hotels with less similarity score

- When text given as input correspond to flight service instead of hotels: **Input string 3=** *"ticket cheap fight on time cabin crew supportive pilot trained"*



	Hotel_Name	Score
0	IntercityHotel Wien	0.03849
1	11 Cadogan Gardens	0.00000
2	1K Hotel	0.00000
3	25hours Hotel beim MuseumsQuartier	0.00000
4	41	0.00000

Figure 7.11: No hotels are recommended for such input

It was observed in the first two inputs that the keywords were positive and valid Group Type (Couple), Travel Reason (Business), and Country were used. Based on the review submitted the similarity between the vectors was found and hotel scores were displayed close to '1' in descending order.

The last input corresponds to flight service, which was invalid for the hotel. Hence very low similarity scores were displayed (close to '0') and were not recommended to customers.

Chapter 8

Conclusion, Future Work and Limitation

8.1 Conclusion

The dataset consists of various features which could have helped improve the accuracy of review classification into ’Good’ and ’Bad’. After plotting various graphs and studying the correlation, we found very little relationship between the target variable and some features. Therefore, these features were exempted during the classification of reviews.

In Experiment 1, Logistic Regression Classifier gave the best classification compared to other traditional machine learning classifiers. While, in Experiment 2, the deep learning models created using text and additional features together gave the best results. The evaluation metric results were compared between Experiment 1 and Experiment 2. It was found that the model trained using deep learning (LSTM) technique could enhance the performance of reviews classification.

Also, the results obtained from three deep learning models trained using LSTM suggests that the model having features as (‘Reviews’) or (Reviews + MetaData) showed better accuracy which is more than 87%. It can be deduced that adding MetaData (other features) to textual reviews played a little role in improving the accuracy as compared to model with textual reviews alone.

When re-sampling technique was performed on the imbalanced dataset, some skewness was removed, which in turn noticeably improved accuracy, recall and F1 score of the model.

The review summarization results were satisfactory as it successfully shortened the long reviews keeping the most relevant sentences intact. The users were given flexibility to summarize the reviews of any hotel in a preferred number of sentences (‘n’). Based on the value of the cosine similarity score, the sentences were displayed keeping the information intact, and thus adding ease to the user experience.

Furthermore, the content-based recommender system with input string = ‘Keywords’+ ‘Tags’ + ‘Country Name’) yielded a list of hotel names ranked in order of relevance. The similarity score values were listed along with hotel names - the hotels having score values close to ‘1’ are highly recommended while scores with a value close to ‘0’ are least/not recommended.

It is important to note that the implemented ideas and training methods can be re-used for reviews submitted in different languages apart from ‘English’. Accordingly, it can be easily implemented in any language without much revamp.

8.2 Future Work

In order to balance the data, we have used both up-sampling for minority classes and down-sampling for majority classes. So, changing the sampling method to SMOTE may produce a good review classification accuracy. This technique arbitrarily chooses samples from less frequent class and determines its ’k’ nearest neighbors.

In an attempt to improve the text classification accuracy using traditional machine learning techniques, multiple models can be stacked together to generate a robust model. The various classifiers, as explained in Experiment 1, can be taken as reference models for stacking.

We performed an extractive review summarization technique where important information was extracted in the form of most relevant sentences based on similarity score. Instead, we may employ an abstractive approach that generates new sentences which might not exist in the actual reviews while preserving the overall meaning.

We may also acquire important topics from the reviews which would help us evaluate trends based on customer's preferences by employing the Topic Modelling Technique.

8.3 Limitation

The positive and negative reviews were analysed, and were divided into only two classes - 'Good' and 'Bad', based on the review score of the hotel.

Employing and training the deep learning model takes a substantial amount of time due to the presence of a parameter 'epoch'. The execution time increases with an increase in epoch value.

The number of fake reviews are negligible as compared to the correct reviews in the dataset. The presented dataset consists of more than 500000 data records, hence it is assumed that the number of fake reviews is very less.

The models developed are specific to 'English' language, although it can be easily adopted without much changes when textual data is in other languages.

Bibliography

[1] N. Vaidya and A. Khachane, “Recommender systems-the need of the ecommerce era,” pp. 100–104, 07 2017.

[2] H. Shaziya, “Text categorization of movie reviews for sentiment analysis,” vol. 4, pp. 11255–11262, 11 2018.

[3] D. Gräbner, M. Zanker, G. Fliedl, and M. Fuchs, “Classification of customer reviews based on sentiment analysis,” *19th Conference on Information and Communication Technologies in Tourism (ENTER)*, 05 2012.

[4] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, “Lexicon-based methods for sentiment analysis,” *Computational linguistics*, vol. 37, no. 2, pp. 267–307, 2011.

[5] A. Bhatt, A. Patel, H. Chheda, and K. Gawande, “Amazon review classification and sentiment analysis,” *International Journal of Computer Science and Information Technologies*, vol. 6, no. 6, pp. 5107–5110, 2015.

[6] R. Prabowo and M. Thelwall, “Sentiment analysis: A combined approach,” *Journal of Informetrics*, vol. 3, no. 2, pp. 143–157, 2009.

[7] T. Nasukawa and J. Yi, “Sentiment analysis: Capturing favorability using natural language processing,” in *Proceedings of the 2nd international conference on Knowledge capture*, pp. 70–77, 2003.

[8] D. Campos, R. Silva, and J. Bernardino, “Text mining in hotel reviews: Impact of words restriction in text classification,” pp. 442–449, 01 2019.

[9] H. Saif, Y. He, and H. Alani, "Semantic sentiment analysis of twitter," in *International semantic web conference*, pp. 508–524, Springer, 2012.

[10] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Handling imbalanced datasets: A review," *GESTS International Transactions on Computer Science and Engineering*, vol. 30, pp. 25–36, 11 2005.

[11] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Intell. Res. (JAIR)*, vol. 16, pp. 321–357, 01 2002.

[12] H. Yin and K. Gai, "An empirical study on preprocessing high-dimensional class-imbalanced data for classification," in *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*, pp. 1314–1319, IEEE, 2015.

[13] B. Santhalingam and N. Ananthanarayanan, "Emote: Enhanced minority oversampling technique," *Journal of Intelligent Fuzzy Systems*, vol. 33, pp. 67–78, 06 2017.

[14] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 168–177, 2004.

[15] V. Gupta and G. S. Lehal, "A survey of text summarization extractive techniques," *Journal of emerging technologies in web intelligence*, vol. 2, no. 3, pp. 258–268, 2010.

[16] G. Erkan and D. R. Radev, "Lexrank: Graph-based lexical centrality as salience in text summarization," *Journal of artificial intelligence research*, vol. 22, pp. 457–479, 2004.

[17] H. P. Edmundson, "New methods in automatic extracting," *J. ACM*, vol. 16, p. 264–285, Apr. 1969.

[18] F. Esposito, A. Corazza, and F. Cutugno, "Topic modelling with word embeddings," 12 2016.

[19] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting similarities among languages for machine translation," *arXiv preprint arXiv:1309.4168*, 2013.

[20] B. Krulwich, "Lifestyle finder: Intelligent user profiling using large-scale demographic data," *AI Magazine*, vol. 18, pp. 37–45, 01 1997.

[21] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*, pp. 325–341, Springer, 2007.

[22] Z. Huang, D. Zeng, and H. Chen, "A comparison of collaborative-filtering recommendation algorithms for e-commerce," *IEEE Intelligent Systems*, vol. 22, no. 5, pp. 68–78, 2007.

[23] M. Balabanovic and Y. Shoham, "Fab: Content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, pp. 66–72, 03 1997.

[24] M. Mohamed, M. Khafagy, and M. Ibrahim, "Recommender systems challenges and solutions survey," 02 2019.

[25] R. Longadge and S. Dongre, "Class imbalance problem in data mining review," *arXiv preprint arXiv:1305.1707*, 2013.

[26] J. B. Lovins, "Development of a stemming algorithm," *Mech. Transl. Comput. Linguistics*, vol. 11, pp. 22–31, 1968.

[27] V. Balakrishnan and E. Lloyd-Yemoh, "Stemming and lemmatization: a comparison of retrieval performances," 2014.

[28] Y. Goldberg, "Neural network methods for natural language processing," *Synthesis Lectures on Human Language Technologies*, vol. 10, pp. 1–309, 04 2017.

[29] C.-M. Tan, Y. Wang, and C.-D. Lee, "The use of bigrams to enhance text categorization," *Information Processing Management*, vol. 38, pp. 529–546, 07 2002.

[30] L.-P. Jing, H.-K. Huang, and H.-B. Shi, "Improved feature selection approach tfidf in text mining," pp. 944 – 946 vol.2, 02 2002.

[31] H. C. Wu, R. W. P. Luk, K. F. Wong, and K. L. Kwok, "Interpreting tf-idf term weights as making relevance decisions," *ACM Transactions on Information Systems (TOIS)*, vol. 26, 2008.

[32] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 384–394, 2010.

[33] S. Ghannay, B. Favre, Y. Esteve, and N. Camelin, "Word embedding evaluation and combination," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 300–305, 2016.

[34] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[35] F. Kiani and O. Tas, "A survey automatic text summarization," vol. 5, pp. 205–213, 06 2017.

[36] J. Han, M. Kamber, and J. Pei, "Data mining: Concepts and techniques," The Morgan Kaufmann Series in Data Management Systems, 2012.

[37] F. Barrios, F. López, L. Argerich, and R. Wachenchauzer, "Variations of the similarity function of textrank for automated summarization," *arXiv preprint arXiv:1602.03606*, 2016.

[38] J. Langeni, "Demystifying the confusion matrix," 2019.

[39] simplilearn, "Classification," 2019.

[40] J. Brownlee, "A gentle introduction to long short-term memory networks," 2017.

[41] S. Saziyabegum and P. S., "Literature review on extractive text summarization approaches," *International Journal of Computer Applications*, 12 2016.

[42] E. Amigó, J. Gonzalo, A. Peñas, and M. Verdejo, "Qarla: A framework for the evaluation of text summarization systems.," 01 2005.

[43] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," 01 2004.

Appendix

Abbreviations

ML - Machine Learning

DL - Deep Learning

RS - Recommender System

NLP - Natural Language Processing

TFIDF - Term frequency–Inverse Document Frequency

LSTM - Long Short Term Memory networks

NB - Naive Bayes

KNN - K Nearest Neighbor

DT - Decision Tree