Customer Feedback Analysis Model "CritiQali"

Customer review analysis to return company's critical issues

Interns

Abhigna L Shastry	PES1201801288
Karan Kumar G	PES1201801883
Sumukh Aithal K	PES1201801461

Mentor

Sumanth V Rao	01FB16ECS402

Microsoft Innovation Lab
PES University
Summer Internship
June - July 2019

Abstract

A Customer Feedback Model to analyze the customer review data of a company, and help identify critical issues based on their severity and impact on customers as well as management. Customer review data for a company is collected and classified into categories (buckets). Aspect-based sentiment/emotion analysis is performed to filter out the negative-sentiment issues; Non-granular reviews (reviews that don't give specific/detailed issues) are removed using dependency parsing, and issues are ranked based on a severity score that is generated, to finally return the top-ranked issues.

Contents

Chapter 1

Introduction

For any company, its entire business is dependant on its customer base; hence, customer satisfaction is of at-most importance. One way of understanding customer needs is by analyzing customer reviews, and thus identifying critical issues faced by the customers.

Manually reading a large number of reviews for a company,and then trying to gain insights has its own limitations, such as inconsistency, time constraints, etc. Thus, the aim of this project is to build a robust Customer Feedback Analysis Model to automate the process of analyzing reviews, to return the critical issues. This can help in improving the customer-company relationship and overall customer satisfaction, thus boosting the company's business.

1.1 Problem Statement

To build a robust Customer Feedback Model to analyze the customer review data of a company, and help identify critical issues based on their severity and impact on customers as well as management.

1.2 Domain

This project majorly deals with Natural Language Processing. Natural Language Processing (NLP) is the branch of Artificial Intelligence that involves

understanding and decoding the subtle nuances of Human Languages, and hence use them to enable machines understanding of the same. NLP is one of the hot topics in the field of AI, given its wide applications. Automated text-generation, text-summarizing, sentiment analysis, are a few of the many sub-domains in NLP which are constantly tried to be improved on. ChatBots (chatting robots) which have great potential in multiple industries, revolve entirely around NLP.

Sentiment Analysis is a *Natural Language Processing and Information Extraction* task that aims to obtain writers feelings expressed in positive or negative comments, questions and requests, by analyzing a large numbers of documents. Generally speaking, sentiment analysis aims to determine the attitude of a speaker or a writer with respect to some topic or the overall tonality of a document. In recent years, the exponential increase in the Internet usage and exchange of public opinion is the driving force behind Sentiment Analysis today. The Web is a huge repository of structured and unstructured data. The analysis of this data to extract latent public opinion and sentiment is a challenging task. The same concept is used later in this paper.

NLP thus happens to be an ever-green topic open for further exploration and research, given the fact there is always more to a language. Sarcasm detection, poetry understanding, metaphors, etc still happen to be a question. These intricacies of a language are often hard to comprehend, even to human beings, and there is a long way to go before one can computerize of the same.

1.3 Target Users

The end product developed targets big companies with a large customer base. Multiple industries, such as the Airlines, Hotels, Restaurants etc could be catered to. Once the reviews are automatically collected from online review platforms, or if provided with an already available dataset, the company is returned with the issues.

In the following pages, the project is first restricted to the Hotel industry, and further expanded to Airlines.

Chapter 2

Literature Survey/Related Work

Multilingual Customer Feedback Analysis [4] was one of the problem statements in International Joint Conference on Natural Language Processing 2017. The shared task provided a corpora annotated using a five-plus-one-classes categorization (comment, request, bug, complaint, meaningless and undetermined). Participants had to train classifiers for the detection of meanings in customer feedback in English, French, Spanish and Japanese.

Aspect category detection is one of the important and challenging subtasks of aspect-based sentiment analysis. Given a set of predened categories, this task aims to detect categories which are indicated implicitly or explicitly in a given review sentence.

In [5], an unsupervised method to address aspect category detection task without the need for any feature engineering is proposed.The proposed method involves manual selection of seed words for each category followed by sentence similarity and cluster similarity using soft cosine similarity technique.The final score is weighted average of sentence and the cluster score with alpha being the parameter.

[9] proposed an unsupervised method called spreading activation that performs association rule mining, using a set of seed words and a co-occurrence matrix between words to form a co-occurrence digraph to detect aspect categories.

Chapter 3

Project Workflow

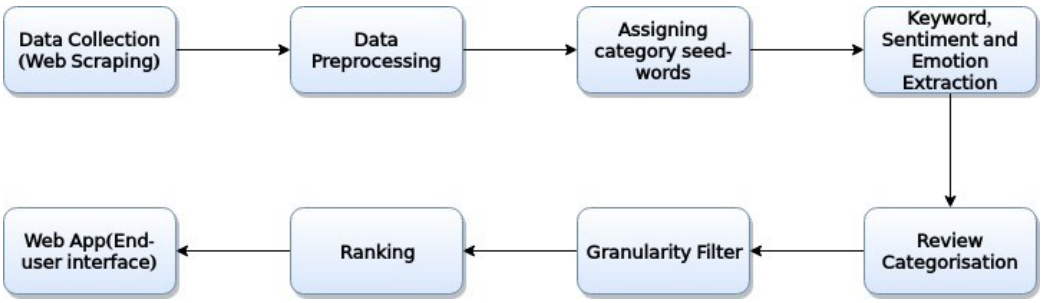

Figure 3.1: Workflow

3.1 Technology Stack

The following software/ APIs have been used.

Python for back-end
Selenium for Web-Scraping
NLTK for text-preprocessing
Spacy for review categorization
IBM NLU for keyword-extraction, sentiment score and emotion score [10]
Stanford CoreNLP for Dependency Parsing [6]
Django for UI back-end
[7] HTML, JavaScript, CSS and Bootstrap for UI Front-end

Some other Python libraries used:
Pandas, Numpy

3.2 Data Collection: Web Scraping

In order to obtain the customer reviews for the company, we scrape websites such as TripAdvisor and goIbibo.

If provided with the Hotel Name and the link to the review data, the scraping tool Selenium is used to extract the review, the date when the review was written, the review title and the contribution of the user who wrote the review. (In the case of the Hotel industry)

Attempts were made to scrape using other tools such as Scrapy and BeautifulSoup as well. However, Selenium was more suitable for the project's requirements, as BeautifulSoup and Scrapy cannot be used for dynamic webpages. The same is stored in the format of a csv file.

3.3 Data Preprocessing

The raw reviews that are scraped, are passed through a text pre-processing and filtering function. The following operations are performed:
First, sentence segmentation is performed, where large reviews comprised of multiple sentences are reduced to single sentences or "review segments".
Next, words are 'lemmatized' such that each word is reduced to its 'lemma', or the simplest form of the word.
Eg: Cleaner ———>Clean

The last step would be Emoticon-removal, where non-ascii characters are removed from the dataset.

A few other text pre-processing operations would be:
Word-stemming, where only the 'root word' form of a particular word is retained. It is often found that the root word does not exist in the dictionary, hence, this operation is not suggested in this project.

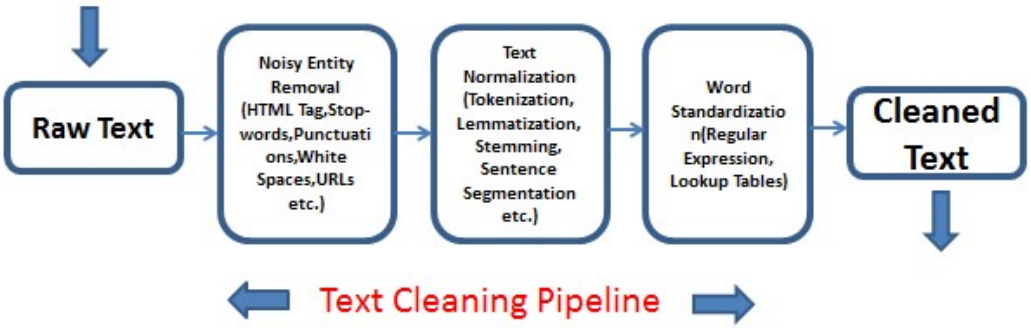

Figure 3.2: Text preprocessing

Stop-word removal is another operation, where the articles (a,an,the), prepositions (on,in,etc), conjunctions (and, but) and other words that are merely of grammatical importance are removed. However this is not being performed, since negation words such as "not", which are of significant importance in case of sentiment analysis, are also removed.

After the necessary pre-processing operations are performed, a cleaned dataset, ready to be worked on is obtained.

3.4 Keyword, Sentiment and Emotion extraction

The dataset now contains clean review segments or sentences. However, it contains both positive and negative sentences. For this project's application, only the negative sentences, which can potentially point out issues are necessary. In order to filter out the positive ones, "Sentiment Analysis" is performed. As mentioned earlier, Sentiment Analysis returns the sentiment (whether the sentence is positive or negative) of the sentence.
Aspect extraction is another sub-domain in NLP, where the "Aspects" of the sentence, or the target components (most likely the subjects) are extracted.

The pre-processed reviews are passed to IBM's NLU (Natural Language Understanding), a tool that offers different NLP operations.

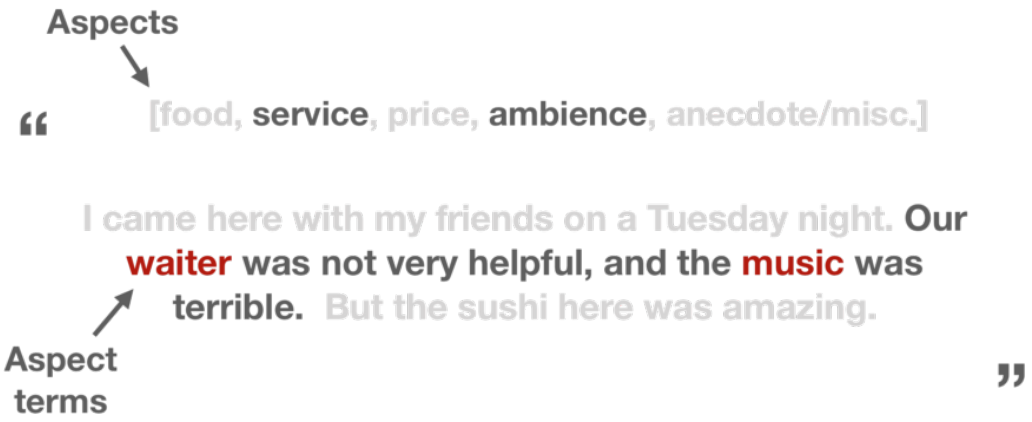

Figure 3.3: Aspect extraction

The NLU returns the "keywords" or the "subjects" in the review segment, along with a sentiment score for the whole sentence.

The sentiment score lies between -1 and 1, -1 denoting a highly negative sentiment and +1 denoting a highly positive sentiment. All the reviews with a Positive or Neutral sentiment, i.e, those with a sentiment score greater than or equal to zero are removed from the dataset, such that only those with a negative sentiment are retained. However, the sentiment score is for the entire sentence, and not for the individual keywords. Hence, a sentence such as "The food was great, but the service was horrible", which has both positive and negative sentiment associated with it, will have a sentiment score that is the average of the positive and negative chunks.

In order to overcome this issue, a concept called "Aspect based Sentiment Analysis" (ABSA) must be implemented. The problem of aspect-based sentiment analysis deals with classifying sentiments (negative, neutral, positive) for a given aspect in a sentence. In other words, instead of classifying the overall sentiment of a text into positive or negative, aspect-based analysis allows us to associate specific sentiments with different aspects of a product or service. The results are more detailed, interesting and accurate because aspect-based analysis looks more closely at the information behind a text.

Customer review about a restaurant	Basic Sentiment Analysis	ABSA
The waiter was really attentive. However, the meat was completely tasteless. Too expensive anyway.	66% negative 33% positive	Service: positive Food: negative Price: negative

Figure 3.4: SA vs ABSA

In order to use ABSA with the available resources, the keywords, along with the review segment, are passed to the "Emotion by target" function that NLU offers. This gives the emotion associated with each keyword in the review. An emotion score for each of Joy, Anger, Fear, Sadness and Disgust is returned by the NLU. These values lie between 0 and 1. A greater score would mean a greater affinity to that emotion. A high Joy score is mapped to a positive sentiment, while a high score in one of the other emotions is mapped to a negative sentiment.

Sentiment Analysis with "Textblob" was attempted initially. However the results were quite unpredictable, ranging from excellent to hopeless.

Earlier attempts at doing ABSA included, involved using Intel's NLP Architect, a tool that was capable of identifying the subject, and giving the associated opinion score and polarity with it. However, the subject-identification was merely checking if a particular word in the sentence belonged to a corpus defined by Intel. Thus, subjects not identified didn't have an associated opinion score with them. Due to these drawbacks, its implementation was not feasible.

The review segments are now tagged with a sentiment score, emotions scores and keywords.

3.5 Review Categorization

The segmented reviews are then classified into categories or buckets, which specify which domain the review segment belongs to. Buckets are manually defined, and may differ from one industry to another.

The buckets for the Hotel would be Ambience, Facilities, Staff, Service, Food, Price, Cleanliness, and specific keywords (seed-words) are explicitly defined under each of the buckets. Reviews would be classified under these buckets. This is done by measuring the similarity between each keyword in a review chunk and a bucket. A string for each bucket, containing the keywords separated with a " ", is used as a representation for the bucket.
Eg: "Food" : menu food dining taste lunch snack"
If the similarity score is greater than a threshold defined, and the keyword doesn't have a higher similarity score with any other bucket, the review is classified under that bucket.

In order to generate a similarity score, Spacy first performs an operation knows as "Word-vectorizaton" or word-embedding, where each word is converted to a corresponding vector in N-dimensions. Words that are similar are closer i.e, have a smaller angle between their word-vectors, and as the similarity decreases, the angle between the vectors increases until it reaches 180 degrees. In order to compute the similarity between words, the "Cosine similarity" or the cosine of the angle between the word-vectors is measured. Extremely similar words will have a cosine similarity score close to 1, while highly dissimilar words have a score close to -1.
A threshold is defined, and if the similarity score is greater than the threshold, the review segment is classified under that particular bucket. One sentence may be classified in one or more buckets, depending on the number of topics it covers.

Experimentation regarding unsupervised approaches for automated seed-word generation under each bucket involved usage of LDA (Latent Dirichlet Allocation) which used a Bag-Of-Words model. A second approach used a Neural Attention Model. Both yielded poor results.

Another attempt at classifying reviews under buckets was using Soft Cosine Similarity [5]. Soft cosine measure is a similarity measure that assesses the

similarity between two sentences, even when they have no words in common. A set of unlabeled review sentences are clustered into k cluster. Clustering is performed based on the Euclidean distance between the average of their word embeddings. The intuition is that sentences in the same cluster share similar information about categories they belong to. The similarity between a given sentence and a pre-dened category is dened as the soft cosine similarity between the sentence and a set of manually selected seed words corresponding to that category. The similarity values give information more about categories that sentence belongs to. The similarity between a cluster and a category is defined by averaging the similarity scores of the sentences in the cluster. There results were not very convincing.

The review segments are now classified under categories.

Figure 3.5: Categories

3.6 Granularity Analysis

Review segments such "Food was Horrible", "Service was bad" etc, which hold strong sentiment and emotion values but do not specify the exact problem, or in other words, are "vague", are termed to be "non-granular". These

non-granular reviews are not particularly of any importance, since no specific "issue" is mentioned, and cannot be acted upon.

In order to remove them, Stanford CoreNLP, a tool that facilitates "Dependency Parsing" is used. "Dependency Parsing" is a technique used to find the grammatical relationship between the words in a sentence.

After analyzing reviews, it is seen that most reviews with just "food", "service", "staff", "ambience" etc as the subject turn out to be non-granular. A set with all these words is created. A majority of the reviews that talk about just one or more of the words in the set are non-granular, and can be removed. This is done as follows.

First, a "POS taggging" operation (Parts-Of-Speech tagging) is performed

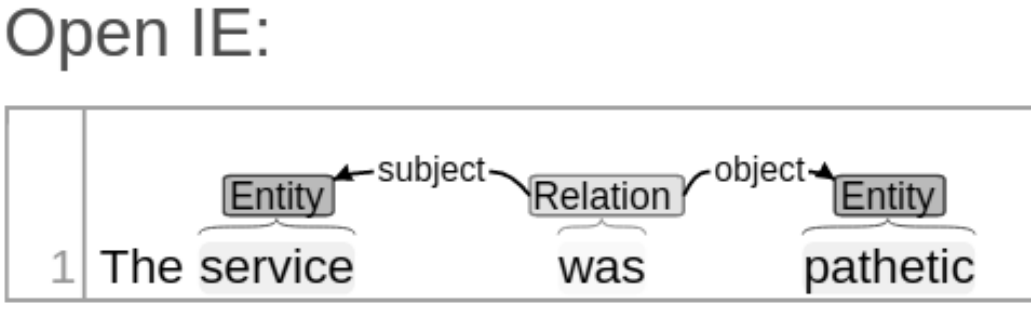

Figure 3.6: Dependency parsing

on the review segment. This tags each word in the sentence, with the corresponding part of speech.

Condition 1

All the nouns in the sentence are collected.
Each of these nouns is present in the set of words defined above.

Condition 2

The "OpenIE" annotator by Stanford CoreNLP is used to identify the "subjects" in the sentence.
Each of the subject words is present in the Set of words defined above.

If any one of the two conditions is satisfied, the sentence is termed non-granular, and is removed from the dataset.

3.7 Ranking

The last and most important step would be the Ranking. Reviews are to be ranked in the order or severity, such that the most severe issue is ranked first, and the least severe is ranked last.
Two approaches may be used for the ranking.

3.7.1 MRR

[2] The algorithm MRR (Most Relevant Review) works on the principle that the relevance of a review can be regarded as the problem of finding reviews that comment on aspects often highlighted about that product/service, such that their rating scores do not differ much from a consensus on such aspects. This approach relies on the concept of graph centrality to rank reviews according to estimated relevance.

The relationship between reviews are represented as a graph, in which the vertices are the reviews, and the edges are defined in terms of the similarity between pairs of reviews. A similarity function that combines the similarity of topics discussed in the texts of the reviews, and the similarity of the respective rating scores is defined. The hypothesis is that a relevant review has a high centrality index since it is similar to many other reviews. The centrality index produces a ranking of vertices importance, which in this approach indicates the ranking of the most relevant reviews. PageRank is used to calculate the centrality scores for each vertex.

PageRank
PageRank is an algorithm used by the Google search engine to measure the authority of a web-page. PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

The concept underlying PageRank is that the importance of a node is measured in terms of both the number and the importance of vertices it is related to. The intuition of using PageRank in MRR is that the more a review is connected to reviews that are highly similar to other reviews, the more representative it is of the opinions issued about the product.

3.7.2 RevRank

[1] The RevRank algorithm is based on a collaborative principle. Given multiple reviews of a product, RevRank identifies the most important concepts. It attempts to find those concepts that are important but infrequent. The main idea employed by RevRank is to use the given collection of reviews along with an external balanced corpus in order to define a reference "Virtual core" (VC) review. In this case, the external corpus is nothing but the data-set containing all the reviews. The VC review is not the best possible review on this product, but is, in some sense, the best review that can be extracted or generated from the given collection.

All reviews, including the VC one, are represented as feature vectors. The feature set is the lexicon of "dominant terms" contained in the reviews, so that vector coordinates correspond to the overall set of dominant terms. Dominant terms could either be frequent terms, or infrequent yet informative terms. This approach is inspired by classic information retrieval, where a document is represented as a bag of words, and each word in each document is assigned a score using the TF-IDF (Term frequency- Inverse document frequency) score that reflects the words importance in this document. However, the document frequency is replaced with the frequency of the word in the external corpus. Reviews are then ranked according to a similarity metric between their vectors and the VC vector, and given relative scores.

For the final ranking, the RevRank/MRR score is one of the parameters taken into consideration. The others would be the User's Contribution, the five emotion scores and how recent the review is. Thus there are 8 parameters being considered for the final ranking.

The recency values are calculated by finding the difference in days, between the date when the issue was written, and the present date. This gives old

reviews a higher recency value, since the difference is larger.

Eg: Consider 2 reviews, one written on 7th July 2012 and another written on 12th July 2012. If the present date is 16th July, the recency score for the first review would be 9, and that for the second would be 4. However, the second review is more recent, when compared to the first.
To tackle this, the obtained recency values are subtracted fro the maximum of the recency, such that the oldest review has a recency score of 0. In the eg. case, the review written on 7th July gets a score of 9-9 which is 0, and that on 12th July gets a score of 9-4 which is 5.

The recency values, User contribution and the RevRank/MRR scores are normalized. A final score is calculated by specifying weights to each of the parameters. Positive weights are defined to all the criterion except the "Joy" score, since a higher joy score would indicate a positive sentiment, and that review must be ranked lower. The weights are assigned manually using the trial-and-error approach to get the most optimal set of weights, to obtain a final score.

Reviews are finally sorted Hotel-wise and bucket wise in the descending order of their final scores, and saved in a csv file. The top 5 reviews are termed as the "critical" issues faced but the customers.

FOOD

Issue No.	Review	Date	Source
1	The staff were very unapproachable, the taste of food in restaurant was pathetic while the prices were of5 star rates.	Nov-18	
2	The food menu for the room service is too expensive (Better to go to the restaurant and have food to save some bucks)	Feb-18	
3	A dead fly was stuck on the bread with many other flying around (0 for hygene) the infrastructure looks old and slightly run down.	Jul-17	
4	Food is above average, room service is extremely delayed, many fountains have stopped functioning.	Apr-18	
5	The food was bad and priced at Rs 1250 per head..too expensive for the kind of food served..should have been priced at no more than Rs 500/ head.	May-17	

✕ Close Window

Figure 3.7: Ranked reviews

Chapter 4

Results and Discussion

The IBM NLU sentiment analyzer is the best in market, yet the results are not perfect. In case of sarcasm, the NLU does a mediocre job. Less-likely used words such as "impeccable" are wrongly assigned sentiments. Thus, the dataset retains a few positive reviews. This is being handled by assigning a negative weight to the joy score, so that the positive ones are ranked lower.

Secondly, although the absolutely non-granular issues are removed using dependency parsing, a few grammatically rich, yet unimportant reviews ("The food was absolutely disappointing, it was just inedible") are retained in the dataset.

While both the ranking algorithms are built robustly, it is observed that the RevRank algorithm, which takes into consideration the specific corpus, yields better results with respect to the granularity of issues. The RevRank algorithm, which checks for the "dominance" of the subject, is capable of recognizing the sentence as less-dominant, and gives it a lower score. To conclude, the RevRank algorithm works better than the MRR, for this project's purposes.

Chapter 5

Conclusions and Future Work

The project works quite well for companies with large datasets. However, there is always scope for improvement.
1.The issue of 'granularity' may be dealt with, using a more technically strong and sound approach, such that reviews that are granular, yet not frequent, also show up as 'critical'.
2.The usage of the IBM NLU could potentially be replaced with an ABSA model and a Sentiment analyzer more specific to this project's requirements. The same hasn't been already implemented, owing to time constraints, and the lack of a labelled dataset.
3.Attempts to automate the generation of weights for parameters, for each industry could be made.

This project is highly scalable, as it can be potentially expanded to multiple industries. All that must be changed would be the "categories" or "buckets", as they are industry-specific. The end goal would be to include Restaurants, Hotels, Airlines, E-commerce giants, etc.

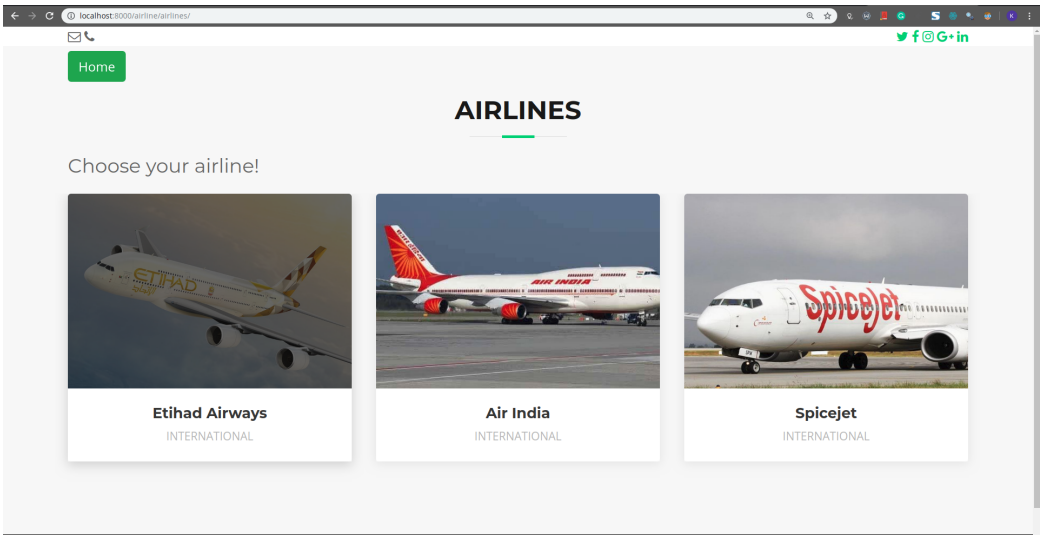

Figure 5.1: Airlines

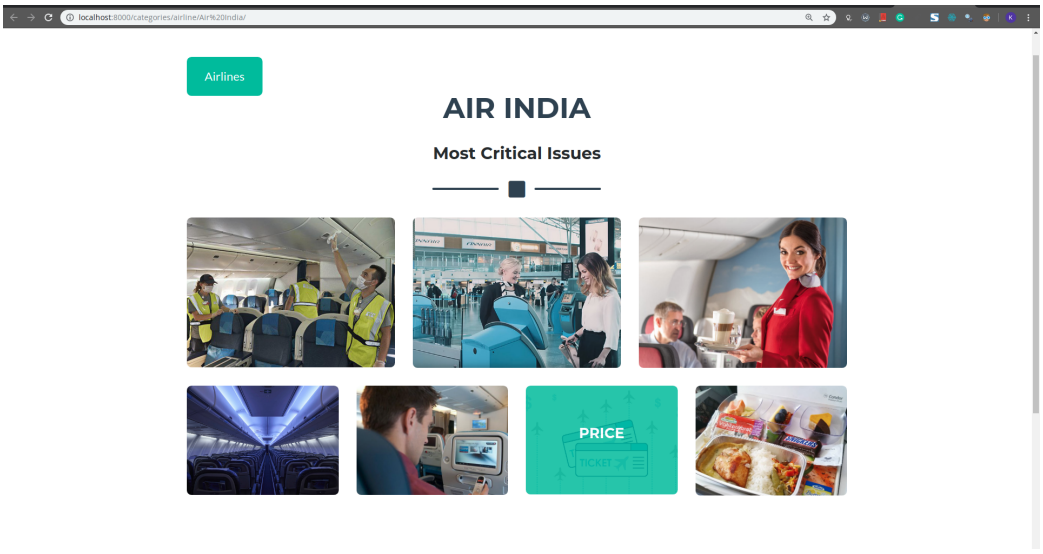

Figure 5.2: Airline Categories

Bibliography

[1] RevRank: A Fully Unsupervised Algorithm for Selecting the Most Helpful Book Reviews
"https://pdfs.semanticscholar.org/2673/34b9f31fdbe92b2f07cea8242b821cc1f47e.pdf"

[2] MRR: an Unsupervised Algorithm to Rank Reviews by Relevance
"https://www.researchgate.net/publication/319046519_MRR_an_unsupervised_algorithm_to_ra

[3] ABSA: An Unsupervised Neural Attention Model for Aspect Extraction
"https://www.comp.nus.edu.sg/ leews/publications/acl17.pdf"

[4] IJCNLP-2017 Task 4: Customer Feedback Analysis
"https://www.aclweb.org/anthology/I17-4004"

[5] An Unsupervised Approach for Aspect Category Detection Using Soft Cosine Similarity Measure
"https://www.semanticscholar.org/paper/An-Unsupervised-Approach-for-Aspect-Category-Using-Ghadery-Movahedi/6cedde502f728eaa6759cff29a956cc47417bc9a"

[6] The Stanford CoreNLP Natural Language Processing Toolkit
"https://www.aclweb.org/anthology/P14-5010"

[7] Django Documentation
"https://docs.djangoproject.com/en/2.2/"

[8] Automatic Text Categorization by Unsupervised Learning
"https://www.aclweb.org/anthology/C00-1066"

[9] Supervised and unsupervised aspect category detection for sentiment analysis with co-occurrence data
"https://sci-hub.tw/https://ieeexplore.ieee.org/document/7900330"

[10] IBM NLU
"https://cloud.ibm.com/apidocs/natural-language-understanding"

[11] Automatic analysis of textual hotel reviews
"https://link.springer.com/article/10.1007/s40558-015-0047-7"

Appendix A

A.1 User Interface

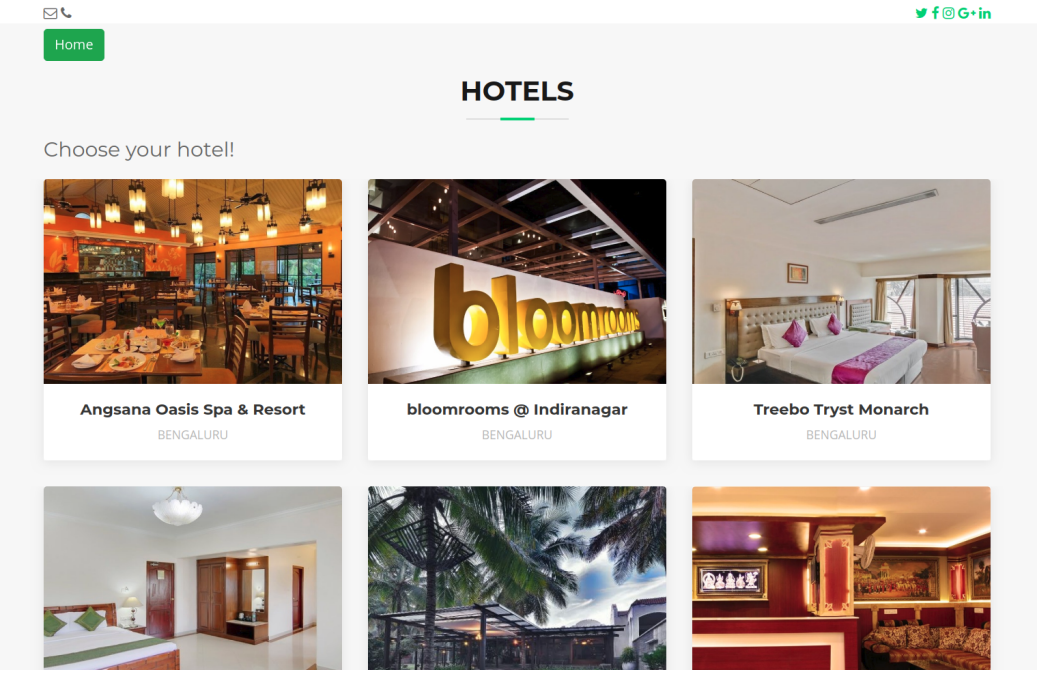

Figure A.1: Hotels

The user interface was constructed using Django [7] for the back-end; HTML, CSS, JavaScript and Bootstrap for the front-end. Pre-designed bootstrap templates have been used to build the front end. The backend has access to the csv file containing the ranked data. It picks the top 5 reviews

in each bucket for a particular hotel, and passes the same to the template.

The frontend essentially has 3 templates:
1. A landing page, where the Industry (Hotel or Airlines) must be chosen
2. A second page with all the Hotels/Airlines, from which a hotel may be selected
3. A third page where the top 5 critical issues under each bucket for that particular hotel may be accessed.

A.2 Containerization for deployment

In order to make the product deployment-worthy, containerization is done to deliver the software is packages. This is being done using "Docker".
Docker is an open source tool for running isolated containers on Linux, making the deployment of apps inside containers faster. It creates portable, self-sufficient containers from any application.

1. Docker allows multiple individual applications to run on the same number of servers
2. It facilitates easy development of encapsulated, ready-to-run applications. Delivering is in containers that hold all libraries and dependencies for an application
3. Managing and deploying applications to live servers is made easy and safe.