SIX MONTH INDUSTRIAL TRAINING FINAL REPORT ON "SENTIMENT ANALYSIS AND REVIEW CLASSFICATION OF SERVICE QUALITY REVIEWS TO PROVIDE INTELLIGENT FEEDBACK"

Submitted in partial fulfilment of the requirements for award of Degree of

BACHELOR OF TECHNOLOGY
(COMPUTER SCIENCE AND ENGINEERING)

SESSION: 2015-2019

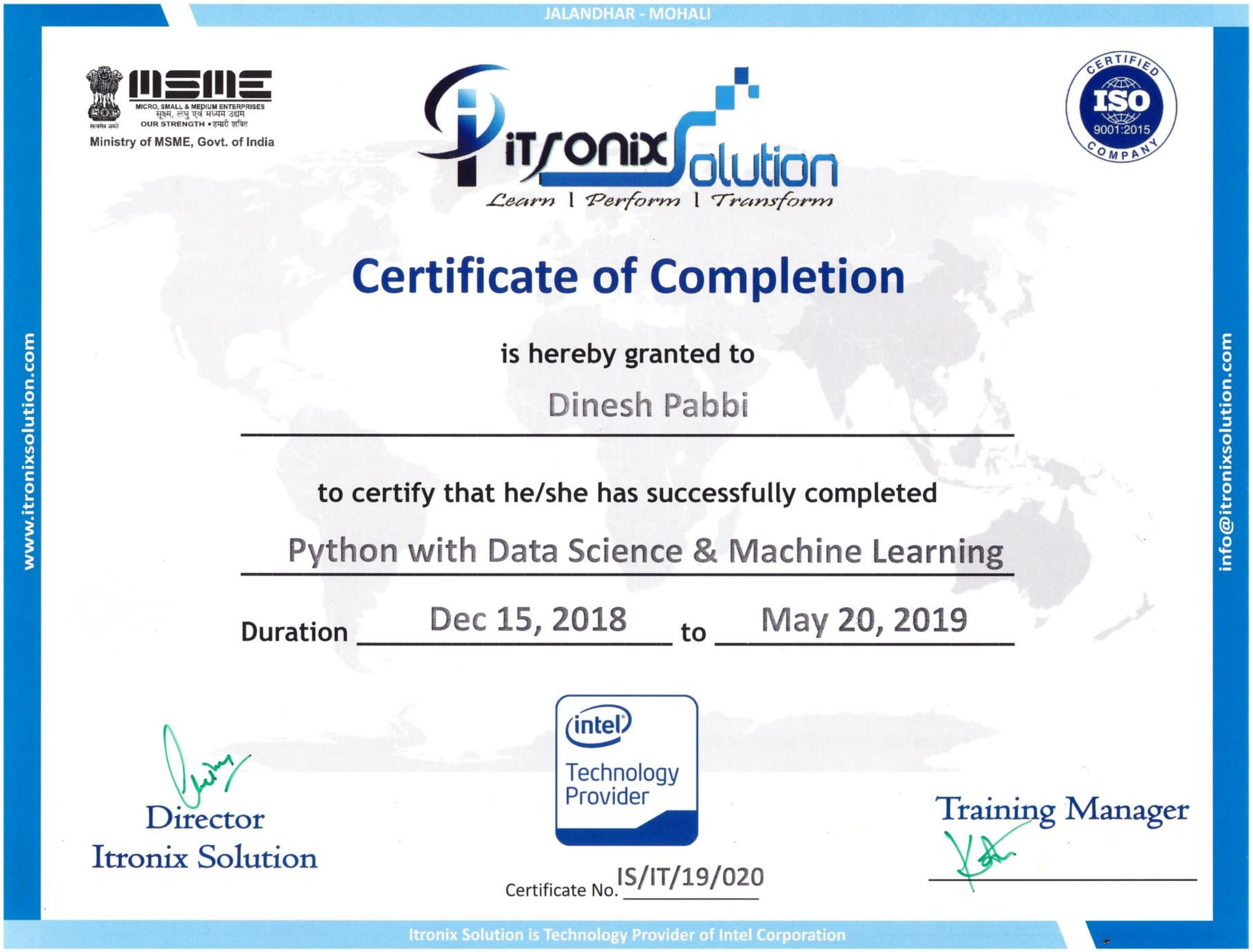
SUBMITTED TO:

Dr. PANKAJDEEP KAUR

SUBMITTED BY:

DINESH PABBI
2015CSA1453

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GURU NANAK DEV UNIVERSITY, AMRITSAR (PUNJAB) INDIA


JALANDHAR - MOHALI

MICRO, SMALL & MEDIUM ENTERPRISES
सूक्ष्म, लघु एवं मध्यम उद्यम
OUR STRENGTH • हमारी शक्ति
Ministry of MSME, Govt. of India

iTronix Solution
Learn | Perform | Transform

CERTIFIED ISO 9001:2015 COMPANY

Certificate of Completion

is hereby granted to

Dinesh Pabbi

to certify that he/she has successfully completed

Python with Data Science & Machine Learning

Duration Dec 15, 2018 to May 20, 2019

Director
Itronix Solution

intel Technology Provider

Certificate No. IS/IT/19/020

Training Manager

www.itronixsolution.com

info@itronixsolution.com

Itronix Solution is Technology Provider of Intel Corporation

DECLARATION

We the undersigned solemnly declare that the report of the project work entitled **“SENTIMENT ANALYSIS AND REVIEW CLASSFICATION OF SERVICE QUALITY REVIEWS TO PROVIDE INTELLIGENT FEEDBACK “,** is based my own work carried out during the course of my study under the supervision of Dr. PANKAJDEEP KAUR.

We assert that the statements made and conclusions drawn are an outcome of the project work. We further declare that to the best of my knowledge and belief that the project report does not contain any part of any work which has been submitted for the award of any other degree/diploma/certificate in this University or any other University.

DINESH PABBI

2015CSA1453

ACKNOWLEDGEMENT

It is our privilege to express our sincerest regards to our project coordinator, **Dr. PANKAJDEEP KAUR**, for their valuable inputs, able guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of our project.

It was purely on the basis of her experience and knowledge that we are able to gain all the required information and guidance to initiate this project.

It is of immense pleasure and profound privilege to express our gratitude and indebtedness along with sincere thanks to **Er. KARAN ARORA**, Itronix Solution for providing us the opportunity to work on project "**SENTIMENT ANALYSIS AND REVIEW CLASSFICATION OF SERVICE QUALITY REVIEWS TO PROVIDE INTELLIGENT FEEDBACK**".

We take this opportunity to thank all our lecturers who have directly or indirectly helped our project. We pay our respects and love to our parents and all other family members and friends for their love and encouragement throughout our career. Last but not the least we express our thanks to our friends for their cooperation and support.

DINESH PABBI

2015CSA1453

ABSTRACT

Sentiment analysis is the computational study of the people's opinions, sentiments, attitudes and emotions expressed in the form of feedback or comments in written language. It is one of the most engaging research areas in Natural Language Processing and text mining. Its popularity is mainly due to following reasons:

1.) It has wide range of applications, because opinions in the written form of language are the key influencers of the human behavioral and thought process. Opinions helps the person to choose optimum from number of available alternatives.
2.) It makes acquainted with many research problems and thus rapid growth of Natural Language Processing and Sentiment Analysis coincide with the social media on the Internet because, huge amount of opinions are shared over these websites in the digital form.

In this Project, We have discussed Sentiment Analysis and Review Classification as mainstream and then we describe the recent work on Sentiment Analysis.

Review Classification is the process to analyze the subjective information in the digital text and then classify the opinion into different categories such as positive opinion, negative opinion or neutral opinion. Where Sentiment Analysis is the procedure by which information extracted from the reviews, feedbacks and comments of the people on the particular service, entities and events. Sentiment Analysis is carried out three different levels i.e. phrase, sentence and documents. This pRoject aims at analyzing a solution for the review analysis on the reviews by different group of people on the Airline Service Quality and to determine how the extracted information can be used as an application of Recommendation System or Feedback System. Every day, over 100,000 flights carry passengers to and from destinations all around the world, and it's safe to say air travel brings out a fairly mixed bag of emotions in people. Through social media, customers now have a platform to say exactly what's on their mind while they are traveling, creating a real-time stream of customer opinion on social networks or directly on the website of the Airline.

For this, polarity of the sentence can be given in three categories as positive, negative and neutral.

ABOUT THE INSTITUTE

Itronix Solutions is one of the most effective Industrial Training Center in the Northern Region. They are the biggest Corporate Trainer in Mohali and Punjab. Many Professionals join Itronix Solutions for upgrading and sharpening their technical skillsets and soft skills. They provide a comprehensive portfolio of Industrial Training and help learners to achieve and sustain competitive edge in the IT Industry.

We opted Itronix Solutions for our 8th Semester Industrial Training Program because of the following reasons.

- There is a huge gap between skills required and skills available.
- Many Organizations lack quality manpower especially if it is sourced locally.
- Academic curriculum is rot in line with corporate requirements.
- Huge upfront investment required in training of resources before they start delivering.
- Itronix Solutions provides best corporate training whether it is updating of technical skills or soft skills like effective communication, etiquette, better management practices etc.

We have been pursuing our Industrial Training in the most trending field of Computer Science known as DATA SCIENCE WITH PYTHON AND MACHINE LEARNING.

TABLE OF CONTENTS

DECLARATION	1
ACKNOWLEDGEMENT	2
ABSTRACT	3
ABOUT THE INSTITUTE	4
TABLE OF CONTENTS	5
LIST OF ABBREVATIONS	8
LIST OF FIGURES	9
CHAPTERS	PAGE NUMBER
Chapter 1 : Introduction 1.1) Application Overview and Scope 1.2) Need for the Application 1.3) Proposed System	10-12
Chapter 2 : Literature Survey and Analysis 2.1) Machine Learning Algorithms 2.1.1) Logistic Regression 2.1.2) Support Vector Machine 2.1.3) Naïve Bayes Classification 2.2) Deep Learning Algorithms 2.2.1) Dense Neural Network 2.2.2) Convolutional Neural Network 2.2.3) Recurrent Neural Network 2.2.4) LSTM 2.2.5) Bidirectional LSTM	12-26

2.3) TF-IDF 2.4) Word Embedding 2.5) Methodologies used in Sentiment Analysis 2.5.1) Classifications 2.5.2) Term Frequency 2.5.3) n-grams 2.5.4) Part-of-Speech 2.5.6) Negations	
Chapter 3 : Application Environment 3.1) Hardware Requirements 3.2) Software Requirements	26-28
Chapter 4 : System Design 4.1) Design Methodologies 4.1.1) The Data Science Process 4.1.2) The NLP Process 4.1.3) The Model Training Process 4.2) Architecture of the Project 4.3) Data Flow Diagrams	28-41
Chapter 5 : Implementation & Application Design 5.1) Implementation 5.1.1) Exploring Data 5.1.2) Pre-Processing Reviews 5.1.3) Training Models 5.1.4) Business View Code 5.1.2) Presentation View Code	42-61
Chapter 6 : 6.1) Technologies Used	62-69

6.1.1) Presentation View 6.1.2) Business View 6.1.3) Database View	
Chapter 7 : Conclusion & Future Scope 7.1) Conclusion and Results 7.2) Future Scope	69-74
Chapter 8 : References	74

LIST OF ABBREVATIONS

Ml – Machine Learning

DS – Data Science

RNN - Recurrent Neural Network

CNN – Convolutional Neural Network

LSTM – Long-short term memory

SVM – Support Vector Machine

DFD – Data Flow Diagram

RAM – Random Access Memory

SQL – Structured Query Language

CSV – Coma Separated Values

DNN – Dense Neural Network

NumPy – Numerical Python

PANDAS- Python Data Analysis Library

GUI – Graphical User Interface

NBC- Naïve Bayes Classification

ETC – Etcetera

LIST OF FIGURES

1. Figure 1 : Dense Neural Network : Page Number 17
2. Figure 2 : Convolutional Neural Network: Page Number 18
3. Figure 3 : Recurrent Neural Network : Page Number 19
4. Figure 4 : LSTM: Page Number 20
5. Figure 5 : Bidirectional LSTM : Page Number 21
6. Figure 6 : Bag of Words : Page Number 22
7. Figure 7 : TF-IDF Representation : Page Number 23
8. Figure 8 : Word Embeddings : Page Number 24
9. Figure 9 : Data Science Process : Page Number 28
10. Figure 10 : Lemmatization : Page Number 34
11. Figure 11: Stop Words Removal : Page Number 34
12. Figure 12: Machine Learning Process : Page Number 34
13. Figure 13 : Application Architecture: Page Number 37
14. Figure 14 : User Database: Page Number 38
15. Figure 15 : Posts Database : Page Number 39
16. Figure 16: Comments : Page Number 39
17. Figure 17 : Relation of Databases : Page Number 39
18. Figure 18 : DFD Level 0 : Page Number 40
19. Figure 19 : DFD Level 1 : Page Number 40
20. Figure 20 : DFD Level 2 : Page Number 41
21. Figure 21 : Most Reviewed Airline : Page Number 42
22. Figure 22 : Polarity Chart: Page Number 43
23. Figure 23 : Classes of Sentiment Chart : Page Number 44
24. Figure 24 : Word Cloud : Page Number 44
25. Figure 25 : Home Page : Page Number 55
26. Figure 26 : Login Page : Page Number 57
27. Figure 27 : Comment Page : Page Number 58
28. Figure 28 : Detail View: Page Number 60
29. Figure 29 : Feedback Portal: Page Number 61
30. Figure 30 : Word Grouping : Page Number 73
31. Figure 31 : Categorization : Page Number 73
32. Figure 32 : Insight Feed : Page Number 74

CHAPTER 1

1.) Introduction

Sentiment analysis or opinion mining is defined as the computational study of people’s opin-ions, appraisals, attitudes, and emotions toward entities, individuals, issues, events, topics and their attributes. (Liu & Zhang, 2012). It is a field of machine learning that employs computational power and well-designed software such as the natural language tool kit (NLTK) to process large amounts of text data with a view of analysing sentiments or opinions expressed in these text corpuses.

Humans have always made decisions based on one or more sentiments of others. E.g. a prospective student in applying to a university would base his/her choice on the positive sentiments about the university; the choice not to buy a product might be borne out of negative reviews the product has generated, these and so many other similar situations go a long way to illustrate the importance of sentiment based decision making. (Liu & Zhang, 2012)

However, decision making that is irreversible after they have been made would not just require a few sentiments, but rather hundreds or thousands of varying opinions in order to make the best decision and this situation is what governments face daily in the governance process. Analysing thousands of sentiments is beyond human capability and is one that requires the use of computational processes to effectively analyse these thousands of sentiment data. With computational processes now easily available and relatively cheaper these days, the application of sentiment analysis is more practicable and easier to undertake now than ever before. (Pang & Lee, 2008)

Sentiment analysis is a prominent and active area of research, spurred particularly by the rap-id growth of web social media and the opportunity to access the valuable opinions of numerous participants on various business and social issues. (Ghiassi, Skinner, & Zimbra, 2013) The field of sentiment analysis forms part of the wider discipline of business intelligence

1.1) Application Overview and Scope

Sentiment Analysis is the task of grouping textual data into categories based upon their sentiment/emotions with which the text was written. Sentiment Analysis is a significant learning problem that is at the core of many information management and retrieval tasks. Sentiment Analysis performs an essential role in various applications that deals with recommendation, classifying, summarization and concisely representing a significant amount of information. Automatic sentiment analysis can be broadly classified into two categories. These are supervised and Semi-supervised. In Supervised sentiment analysis, some mechanism external to the analysis model (generally human) provides information related to the correct sentiment for the text. Thus, in case of supervised sentiment analysis, it becomes easy to test the accuracy of the model. In case of Semi-supervised sentiment analysis, parts of the documents are labelled by an external mechanism.

Taking about challenges, there are two main factors which contribute to making sentiment analysis a challenging task: (a) feature extraction; (b) topic ambiguity. First, Feature extraction deals with taking out the right set of features that accurately describes the sentiment and helps in building a good analysis model. Second, many broad topic are themselves so complicated that it becomes difficult to put it into any specific category

1.2) <u>Need of the Application</u>

Some of the applications that make use of the above techniques for document classification are listed below:

• Email routing: Routing an email to a general address, to a specific address or mailbox depending on the topic of the email.

• Language identification: Automatically determining the language of a text. It can be useful in many use cases one of them being the direction in which the language should be processed.

• Readability Assessment: Automatically determining how readable any document is for an audience of a certain age.

• Sentiment Analysis: Determining the sentiment of a speaker based on the content of the document.

1.3) <u>Proposed System</u>

We used the train models to create a feedback system to inform doctors about the negative reviews and the keywords used in the negative reviews. We use our machine learning model to detect if the new review is negative. After the detection of the polarity of review, we use the rave algorithm to extract keywords from the review and then ask the reviewer about more details about certain aspect of the hospital they might want to complain about. This form will show up only if the review is negative and will stay hidden if the patient has positive review for the hospital.

The information entered in the feedback form will be coupled with the keywords extracted from the rake machine learning model to provide high quality feedback to hospitals to focus on things that are commonly complained by the patients. This will help improve not only customer service but also the quality of healthcare and also help reduce the costs spend on improving the hospitals as the hospital will know where the money is to be invested to improve the patient satisfaction

We deploy our model as a flask web service which can be used to give predictions on textual data. The service can be accessed using an http request and the output of the request is the prediction in JSON format. We also developed a website using expressJS, mongodB database and passportJS for user authentication and simple bootstrap for giving UI. The website developed uses the machine learning model to show feedback form as well as extract the keywords from reviews to submit to the specific doctor which has been reviewed. The feedback system developed can be used by any website and the models can be trained on any dataset to be used in any domain.

CHAPTER 2

Literature Survey and Analysis

In this project work, we use two class of supervised learning algorithms the machine learning algorithms such as logistic regression, SVM(Linear), Naïve Bayes, Decision tree and then Deep Learning models such as Dense Neural Nets, Convolutional Neural Networks, GRU, LSTM, Bidirectional LSTM and Recurrent Neural Networks.

2.1) Machine Learning Algorithms

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering, and computer vision, where it is infeasible to develop an algorithm of specific instructions for performing the task. Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

2.1.1) Logistic Regression

In statistics, the logistic model (or logit model) is a widely used statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail, win/lose, alive/dead or healthy/sick; these are represented by an indicator variable, where the two values are labeled "0" and "1".

In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names.

Analogous models with a different sigmoid function instead of the logistic function can also be used, such as the probit model; the defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each dependent variable having its own parameter; for a binary independent variable this generalizes the odds ratio.The probability that the output is 1 given its input can be represented as:

$$P(y = 1 \mid x)$$

Linear regression model can generate the predicted probability as any number ranging from negative to positive infinity, whereas probability of an outcome can only lie between 0< P(x)<1. This is solved by logistic regression which is expressed as :

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X.$$

where, the left hand side is called the logit or log-odds function, and p(x)/(1-p(x)) is called odds.

2.1.2) Support Vector Machine

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

When data is unlabelled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups. The support-vector clustering algorithm, created by Hava Siegelmann and Vladimir Vapnik, applies the statistics of support vectors, developed in the support vector machines algorithm, to categorize unlabeled data, and is one of the most widely used clustering algorithms in industrial applications.For linear kernel the equation for prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

f(x) = B(0) + sum(ai * (x,xi))

2.1.3) Naive Bayes Classification

_In machine learning, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

Naive Bayes has been studied extensively since the 1960s. It was introduced (though not under that name) into the text retrieval community in the early 1960s, and remains a popular (baseline) method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. It also finds application in automatic medical diagnosis.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

In the statistics and computer science literature, naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naive Bayes is not (necessarily) a Bayesian method.

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

In plain English, using Bayesian probability terminology, the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$$

2.2) <u>Deep Learning Algorithms</u>

Deep learning is a sub-field of machine learning dealing with algorithms inspired by the structure and function of the brain called artificial neural networks. In other words, It mirrors the functioning of our brains. Deep learning algorithms are similar to how nervous system structured where each neuron connected each other and passing information.

Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on artificial neural networks. Learning can be supervised, semi-supervised or unsupervised.

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases superior to human experts.

Neural networks were inspired by information processing and distributed communication nodes in biological systems. ANNs have various differences from biological brains. Specifically, neural networks tend to be static and symbolic, while the biological brain of most living organisms is dynamic (plastic) and analog.

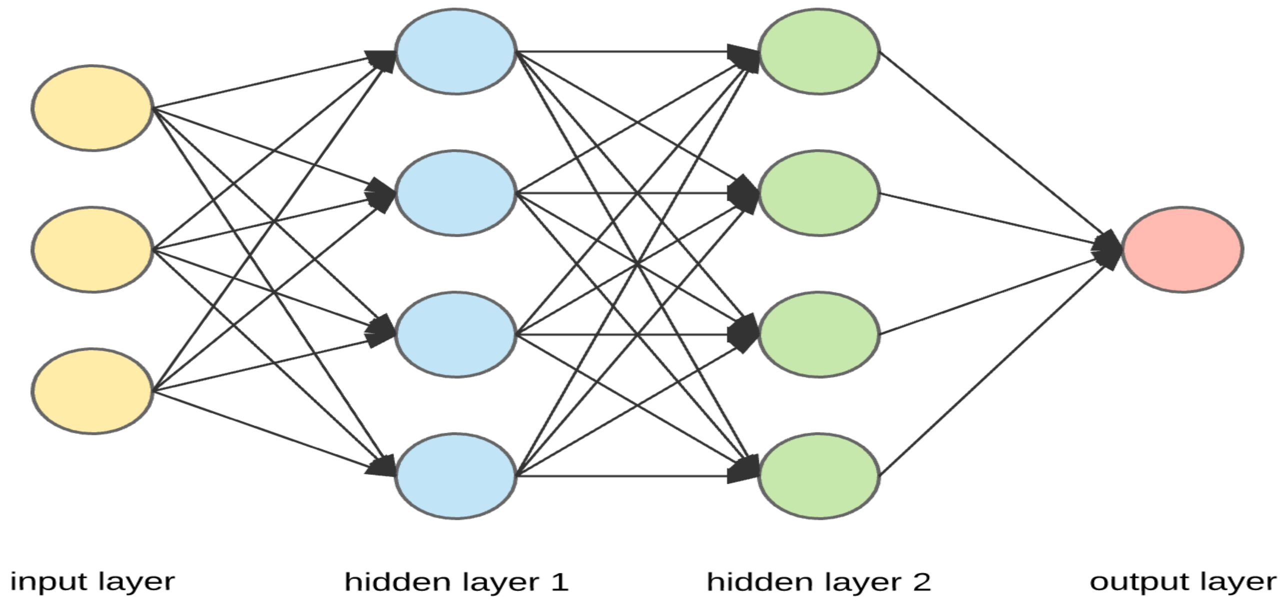
2.2.1) Dense Neural Network

Artificial neural networks (ANN) or connectionist systems are computing systems vaguely inspired by the biological neural networks and astrocytes that constitute animal brains. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge about cats, for example, that they have fur, tails, whiskers and cat-like faces. Instead, they automatically generate identifying characteristics from the learning material that they process.

An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it.

In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called 'edges'. Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

The original goal of the ANN approach was to solve problems in the same way that a human brain would. However, over time, attention moved to performing specific tasks, leading to deviations from biology. Artificial neural networks have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games and medical diagnosis.



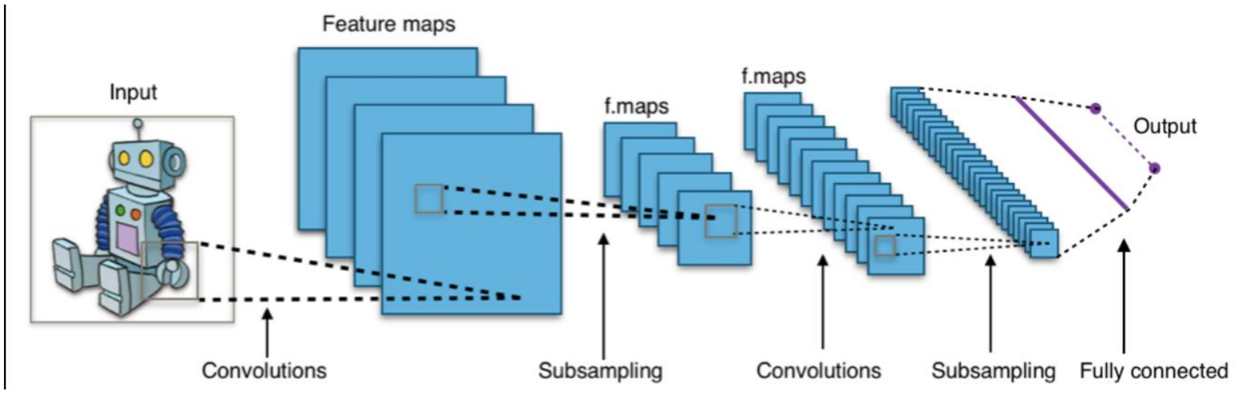
2.2.2) Convolutional Neural Network

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery.

CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually refer to fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks make them prone to overfitting data. Typical ways of regularization includes adding some form of magnitude measurement of weights to the loss function. However, CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme.

Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage.

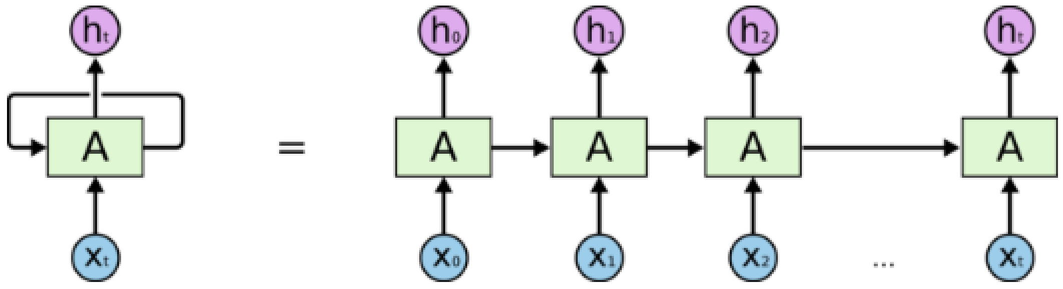


2.2.3) Recurrent Neural Network

A a recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition.

A chunk of neural network, A, looks at some input xt and outputs a value ht. A loop allows information to be passed from one step of the network to the next. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor.Consider what happens if we unroll the loop.

This chain-like nature reveals that recurrent neural networks are intimately related to sequences and lists. They're the natural architecture of neural network to use for such data. And they certainly are used! In the last few years, there have been incredible success applying RNNs to a variety of problems: speech recognition, language modeling, translation, image captioning… The list goes on.



An unrolled recurrent neural network.

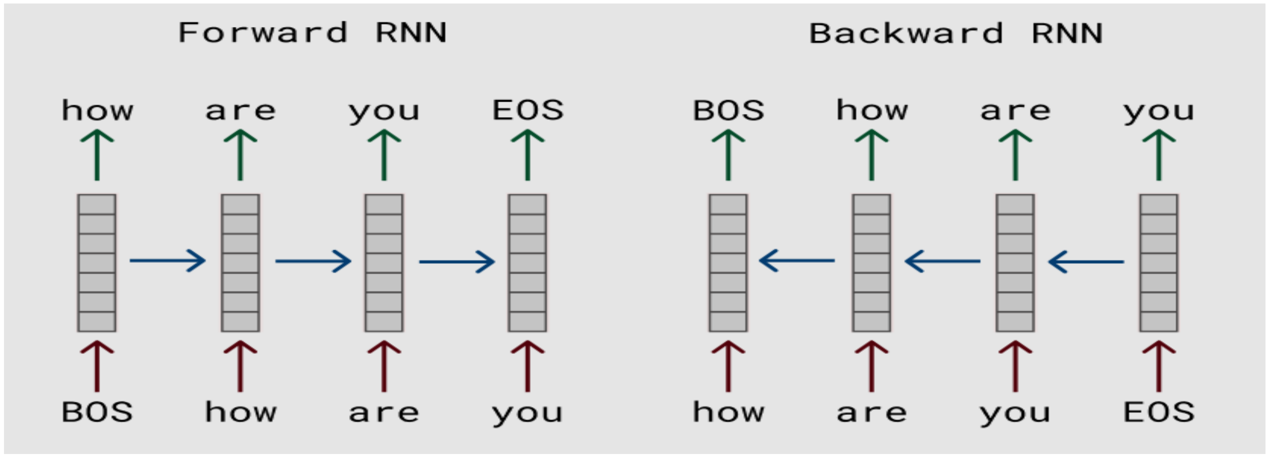
2.2.4) LSTM

LSTM is a recurrent neural network (RNN) architecture that REMEMBERS values over arbitrary intervals. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. Relative insensitivity to gap length gives an advantage to LSTM over alternative RNNs, hidden Markov models and other sequence learning methods.

The structure of RNN is very similar to hidden Markov model. However, the main difference is with how parameters are calculated and constructed. One of the advantage with LSTM is **insensitivity to gap length**. RNN and HMM rely on the hidden state before emission / sequence. If we want to predict the sequence after

1,000 intervals instead of 10, the model forgot the starting point by then. LSTM **REMEMBERS**.

The **long-term memory** is usually called the **cell state**. The looping arrows indicate recursive nature of the cell. This allows information from previous intervals to be stored with in the LSTM cell. Cell state is modified by the forget gate placed below the cell state and also adjust by the input modulation gate. From equation, the previous cell state forgets by multiply with the forget gate and adds new information through the output of the input gates.

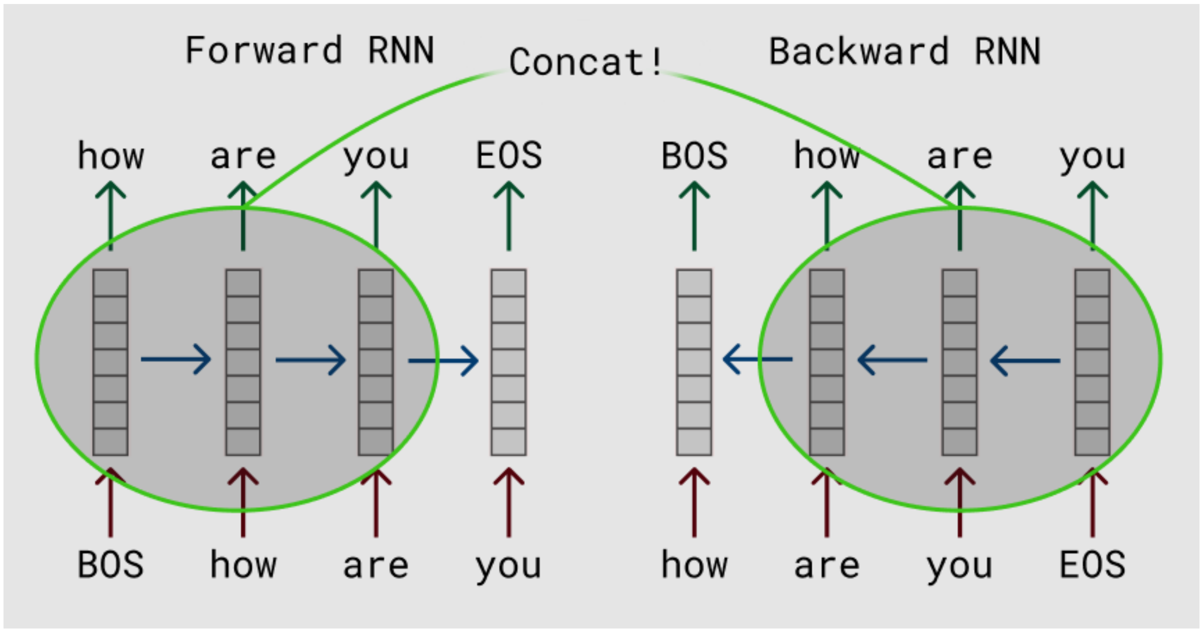


2.2.5) Bidirectional LSTM

The usual approach in building a language model is to predict a word given the previous words. We can use either use an ngram language model or a variant of a recurrent neural network (RNN). An RNN (theoretically) gives us infinite left context (words to the left of the target word). But what we would really like is to use both left and right contexts see how well the word fits in the sentence. A bidirectional language model can enable this.

Most deep learning frameworks will have support for bidirectional RNNs. They will usually return two sets of RNN hidden vectors where one is the output of the forward RNN and the other is the output of the backward RNN. These hidden vectors will be used to predict the next word in the sentence, where next word is the previous word for the backward RNN.

The simple trick is to stagger the hidden vectors so after concatenating them, they are predicting on the same token. Remember to add some padding at both ends of the sentence so we have enough context to predict the words. Here, we add a BOS (beginning of sentence) and EOS (end of sentence) padding tokens.



Review Representation

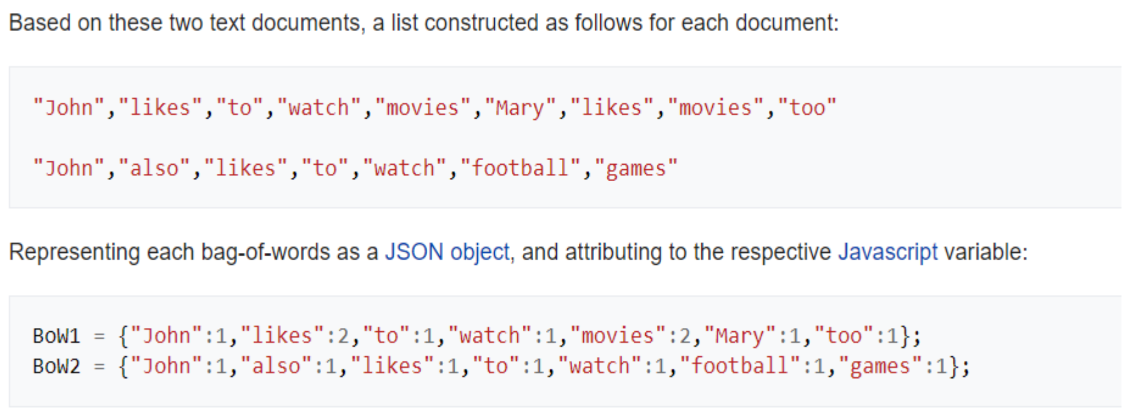
Since Different Machine Learning models take as input different word representation of the textual data. We opted for three types of representation mainly, **TF-IDF, CountVector and Word2Vec**. TF-IDF and CountVector are mainly used classic machine learning algorithms such as SVM etcetera and Word2Vec is very commonly used in deep learning to feed textual data. All these techniques are as follow:

CountVector / Bag of Words

The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. The bag-of-words model has also been used for computer vision.

The bag-of-words model is commonly used in methods of document classification where the (frequency of) occurrence of each word is used as a feature for training a classifier

Bag-of-word model is an orderless document representation—only the counts of words mattered. For instance, in the above example "John likes to watch movies. Mary likes movies too", the bag-of-words representation will not reveal that the verb "likes" always follows a person's name in this text. As an alternative, the n-gram model can store this spatial information. Applying to the same example above, a bigram model will parse the text into the following units and store the term frequency of each unit as before.



Based on these two text documents, a list constructed as follows for each document:

```
"John","likes","to","watch","movies","Mary","likes","movies","too"

"John","also","likes","to","watch","football","games"
```

Representing each bag-of-words as a JSON object, and attributing to the respective Javascript variable:

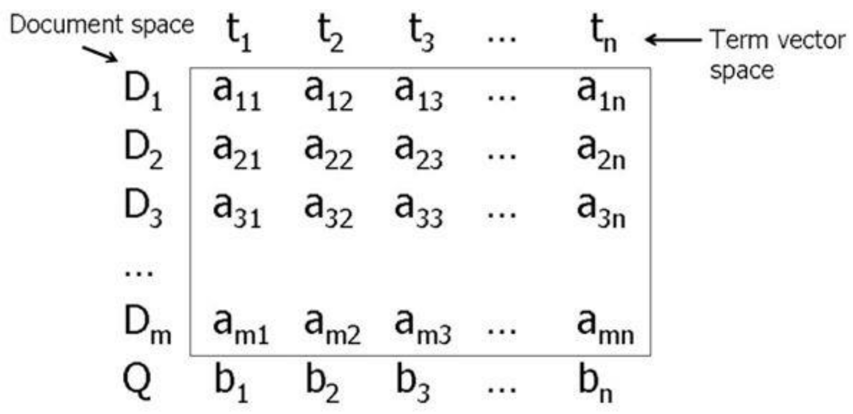
```
BoW1 = {"John":1,"likes":2,"to":1,"watch":1,"movies":2,"Mary":1,"too":1};
BoW2 = {"John":1,"also":1,"likes":1,"to":1,"watch":1,"football":1,"games":1};
```

TF-IDF

In information retrieval, tf–idf or TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. Tf–idf is one of the most popular term-weighting schemes today; 83% of text-based recommender systems in digital libraries use tf–idf.

Variations of the tf–idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query. tf–idf can be successfully used for stop-words filtering in various subject fields, including text summarization and classification.

One of the simplest ranking functions is computed by summing the tf–idf for each query term; many more sophisticated ranking functions are variants of this simple model.


Document space
Term vector space
t_1 t_2 t_3 … t_n
D_1 a_{11} a_{12} a_{13} … a_{1n}
D_2 a_{21} a_{22} a_{23} … a_{2n}
D_3 a_{31} a_{32} a_{33} … a_{3n}
…
D_m a_{m1} a_{m2} a_{m3} … a_{mn}
Q b_1 b_2 b_3 … b_n

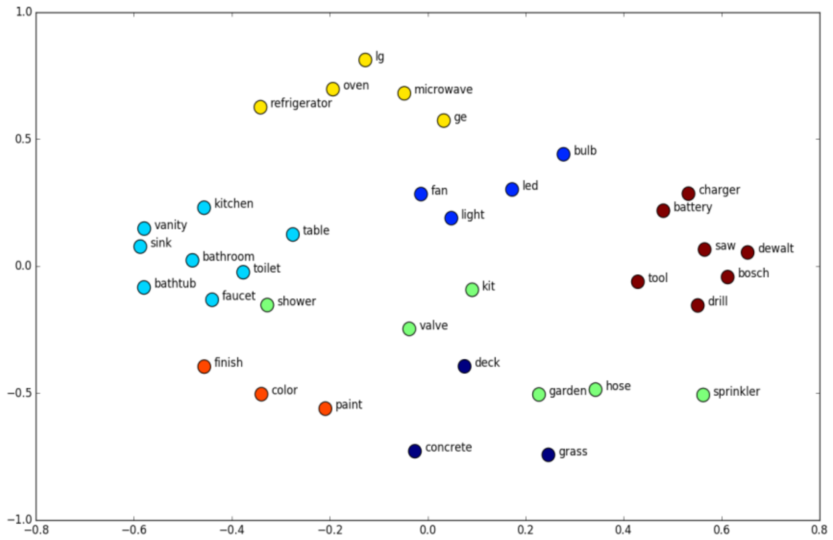
Word Embeddings

Word embedding is the collective name for a set of language modeling and feature learning techniques in natural language processing (NLP) where words or phrases from the vocabulary are mapped to vectors of real numbers. Conceptually it involves a mathematical embedding from a space with many dimensions per word to a continuous vector space with a much lower dimension.

Methods to generate this mapping include neural networks, dimensionality reduction on the word co-occurrence matrix, probabilistic models, explainable knowledge base method, and explicit representation in terms of the context in which words appear.

Word and phrase Embeddings, when used as the underlying input representation, have been shown to boost the performance in NLP tasks such as syntactic parsing and sentiment analysis.

One of the main limitations of word Embeddings (word vector space models in general) is that possible meanings of a word are conflated into a single representation (a single vector in the semantic space). Sense Embeddings are proposed as a solution to this problem: individual meanings of words are represented as distinct vectors in the space.



METHODOLOGIES USED IN SENTIMENT ANALYSIS

A wide range of tools and techniques can be employed to tackle the goals described in theprevious section. This section therefore describes some of the most common and widely usedones.

Classification

: Many of the tasks in Sentiment Analysis can be thought of as classifi-cation. (Mejova, 2009) Machine Learning offers many algorithms designed to under-take that, but this task of classifying text according to its sentiment presents many

unique challenges. These can be formulated in one question: “What kinds of features

do we use?”

Term Frequency or Presence

: Traditional Information Retrieval systems have long emphasized the importance of term frequency. The widely used TF-IDF (Term Frequency - Inverse Document Frequency) measure is well-used in modelling documents according to Jones. (Jones, 1972)TF-IDA is a measure of how concentrated into relatively few documents is the co-currencies of a given word. (Rajaraman &Ullman, 2011) The intuition is that terms that often appear in the document but seldom in the whole collection are more informative as to what the document is about as compared to the terms mentioned just once. (Mejova, 2009) TF-IDF have been shown to be quiet effective in sentiment classification (Liu & Zhang, 2012) In the field of Sentiment Analysis we find that instead of paying attention to most frequent terms, it is more beneficial to seek out the most unique ones. Pang et al improved the performance of this system using term presence instead of frequency. Hoffman states in their paper that, "apparently people are creative when they are being opinionated", implying the importance of low-frequency terms in opinionated texts.

n-grams

Term positions are also important in document representation for Sentiment Analysis. The position of terms determines, and sometimes reverses, the polari-ty of the phrase. So, position information is sometimes encoded into the feature vec-tor. (pang, Lee, & Vaithyanathan, 2002) Wiebe and Hoffman selects n-grams(n=1,2,3,4) based on precision calculated using annotated documents. (Wiebe &Hoffmann, 2005) The n-grams are a word-stem, part-of-speech pair, for instance is a 3-gram.

Part-of-Speech

Adjectives are a good indicator of sentiment in text, and in the past decade they have been commonly exploited in Sentiment Analysis (Whitelaw, Gag ,& Argamon, 2005). This is true for other fields in textual analysis, since part-of-speech tags can be considered to be a crude form of word sense disambiguation.(Wilks & Stevenson, 1998). In his work, Turney used part-of-speech patterns, to including an adjective and even went further to used adverb as well, for sentiment detection at the document level. (Turney, 2002) Syntax information has also been used in feature sets, though there is still discussion about the advantages of this information in Sentiment classification (Pang & Lee, 2008). This information however may include important text features such as negation, intensifiers, and diminishes used sub tree-based boosting algorithm with dependency tree-based features for polarity classification, and show that it outperforms the bag-of-words baseline. (Kennedy &Inkpen, 2006)

Negations

Negations have been long known to be integral in Sentiment Analysis. The usual bag-of-words representation of text disconnects all of the words, and considers sentences like “I like this car” and “I don’t like this car” very similar, since only one word distinguishes one from the other. But when talking about sentiment, a negation changes the polarity of a whole phrase. Negations are often considered in post-processing of results, while the original representation of text ignores them (Hu & Liu, 2005), one could explicitly include the negation in the document representation by appending them to the terms that are close to negations; for example term “like - NOT “would be extracted form “I don’t like this book” (Pang & Lee, 2008). Though using co-location may be too crude a technique. It would be incorrect to negate the sentiment in a sentence such as “No wonder everyone loves this car”. To handle such cases. (Na, Sui , Khoo , Chan , & Zhou, 2004) use specific part-of-speech tags patterns to identify the negations relevant to the sentiment polarity of a phrase

CHAPTER 3

Application Environment:

(i) HARDWARE REQUIREMENT :-

• **Operating System**	Window 7, Window 8,8.1 etc
• **PROCESSOR**	Dual Core
• **RAM**	Minimum 4GB

• **STORAGE**	10-20 GB recommended
• **GPU**	Preferred for better numerical calculation speeds(1GB)

(ii) **SOFTWARE REQUIREMENT**:-

- **Jupyter Notebook**

  The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning, transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

- **Visual Studio Code**

  It is an open Source Code Editor developed and managed by Microsoft. It provides the support of various languages and also implemented with internal Intelligence code completion feature of all the supported languages.

- **MongoDB**

  MongoDB is a relational database management system . As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet).

- **Virtual Environment**

If the application is to be run on the localhost then it is recommended to use virtual environment so that, the required dependencies do not mess with the other programs.

- **Google Collab**

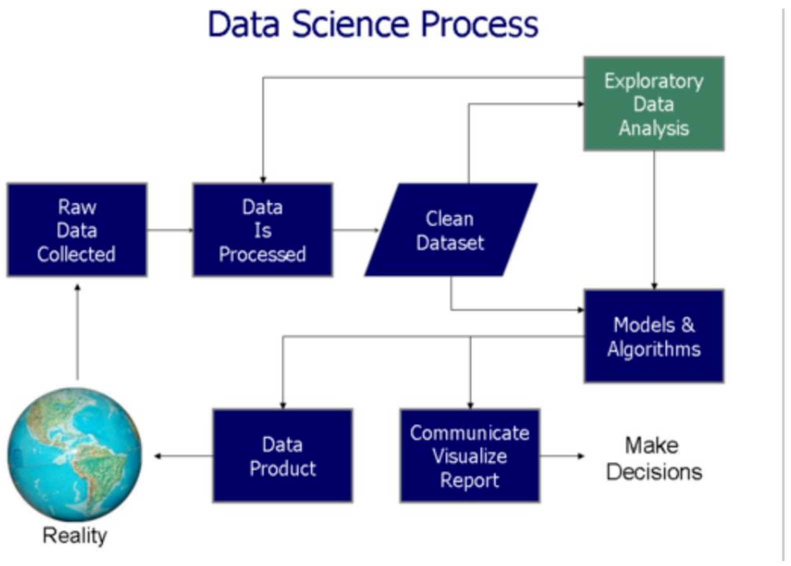
Colaboratory is a Google research project created to help disseminate machine learning education and research. It's a Jupyter notebook environment that requires no setup to use and runs entirely in the cloud. This means that as long as you have a Google account, you can freely train your models on a K80 GPU.

When you log in and connect to the Colaboratory runtime, you will get your own virtual machine with a K80 GPU (not always entirely yours, read below) and a Jupyter notebook environment. You can use it to your heart's content for up to 12 hours. Or until you close your browser window. Be warned, sometimes the runtime can disconnect randomly.

CHAPTER 4

4.1) DESIGN METHODOLIGIES

4.1.1 The Data Science Process:



Step 1: Frame the problem

The first thing you have to do before you solve a problem is to define exactly what it is. You need to be able to translate data questions into something actionable.

You'll often get ambiguous inputs from the people who have problems. You'll have to develop the intuition to turn scarce inputs into actionable outputs–and to ask the questions that nobody else is asking.

Say you're solving a problem for the VP Sales of your company. You should start by understanding their goals and the underlying why behind their data questions. Before you can start thinking of solutions, you'll want to work with them to clearly define the problem.

A great way to do this is to ask the right questions.

You should then figure out what the sales process looks like, and who the customers are. You need as much context as possible for your numbers to become insights.

You should ask questions like the following:

> Who are the customers?
>
> Why are they buying our product?
>
> How do we predict if a customer is going to buy our product?
>
> What is different from segments who are performing well and those that are performing below expectations?
>
> How much money will we lose if we don't actively sell the product to these groups?

In response to your questions, the VP Sales might reveal that they want to understand why certain segments of customers have bought less than expected. Their end goal might be to determine whether to continue to invest in these segments, or de-prioritize them. You'll want to tailor your analysis to that problem, and unearth insights that can support either conclusion.

It's important that at the end of this stage, you have all of the information and context you need to solve this problem.

Step 2: Collect the raw data needed for your problem

Once you’ve defined the problem, you’ll need data to give you the insights needed to turn the problem around with a solution. This part of the process involves thinking through what data you’ll need and finding ways to get that data, whether it’s querying internal databases, or purchasing external datasets.

You might find out that your company stores all of their sales data in a CRM or a customer relationship management software platform.You can export the CRM data in a CSV file for further analysis.

Step 3: Process the data for analysis

Now that you have all of the raw data, you’ll need to process it before you can do any analysis. Oftentimes, data can be quite messy, especially if it hasn’t been well-maintained. You’ll see errors that will corrupt your analysis: values set to null though they really are zero, duplicate values, and missing values. It’s up to you to go through and check your data to make sure you’ll get accurate insights.

You’ll want to check for the following common errors:

Missing values, perhaps customers without an initial contact date

Corrupted values, such as invalid entries

Timezone differences, perhaps your database doesn’t take into account the different timezones of your users

Date range errors, perhaps you’ll have dates that makes no sense, such as data registered from before sales started

You’ll need to look through aggregates of your file rows and columns and sample some test values to see if your values make sense. If you detect something that doesn’t make sense, you’ll need to remove that data or replace it with a default value. You’ll need to use your intuition here: if a customer doesn’t have an initial contact date, does it make sense to say that there was NO initial contact date? Or do you have to hunt down the VP Sales and ask if anybody has data on the customer’s missing initial contact dates?

Once you’re done working with those questions and cleaning your data, you’ll be ready for exploratory data analysis (EDA).

Step 4: Explore the data

When your data is clean, you’ll should start playing with it!

The difficulty here isn't coming up with ideas to test, it's coming up with ideas that are likely to turn into insights. You'll have a fixed deadline for your data science project (your VP Sales is probably waiting on your analysis eagerly!), so you'll have to prioritize your questions. '

You'll have to look at some of the most interesting patterns that can help explain why sales are reduced for this group. You might notice that they don't tend to be very active on social media, with few of them having Twitter or Facebook accounts. You might also notice that most of them are older than your general audience. From that you can begin to trace patterns you can analyze more deeply.

Step 5: Perform in-depth analysis

This step of the process is where you're going to have to apply your statistical, mathematical and technological knowledge and leverage all of the data science tools at your disposal to crunch the data and find every insight you can.

In this case, you might have to create a predictive model that compares your underperforming group with your average customer. You might find out that the age and social media activity are significant factors in predicting who will buy the product.

If you'd asked a lot of the right questions while framing your problem, you might realize that the company has been concentrating heavily on social media marketing efforts, with messaging that is aimed at younger audiences. You would know that certain demographics prefer being reached by telephone rather than by social media. You begin to see how the way the product has been has been marketed is significantly affecting sales: maybe this problem group isn't a lost cause! A change in tactics from social media marketing to more in-person interactions could change everything for the better. This is something you'll have to flag to your VP Sales.

You can now combine all of those qualitative insights with data from your quantitative analysis to craft a story that moves people to action.

Step 6: Communicate results of the analysis

It's important that the VP Sales understand why the insights you've uncovered are important. Ultimately, you've been called upon to create a solution throughout the data science process. Proper communication will mean the difference between action and inaction on your proposals.

You need to craft a compelling story here that ties your data with their knowledge. You start by explaining the reasons behind the underperformance of the older demographic. You tie that in with the answers your VP Sales gave you and the insights you've uncovered from the data. Then you move to concrete solutions that address the problem: we could shift some resources from social media to personal calls. You tie it all together into a narrative that solves the pain of your VP Sales: she now has clarity on how she can reclaim sales and hit her objectives.

4.1.2 The NLP Process:

Let's look at a piece of text from Wikipedia:

London is the capital and most populous city of England and the United Kingdom. Standing on the River Thames in the south east of the island of Great Britain, London has been a major settlement for two millennia. It was founded by the Romans, who named it Londinium.

Step 1: Sentence Segmentation

The first step in the pipeline is to break the text apart into separate sentences. That gives us this:

"London is the capital and most populous city of England and the United Kingdom."

"Standing on the River Thames in the south east of the island of Great Britain, London has been a major settlement for two millennia."

"It was founded by the Romans, who named it Londinium."

We can assume that each sentence in English is a separate thought or idea. It will be a lot easier to write a program to understand a single sentence than to understand a whole paragraph.

Coding a Sentence Segmentation model can be as simple as splitting apart sentences whenever you see a punctuation mark. But modern NLP pipelines often use more complex techniques that work even when a document isn't formatted cleanly.

Step 2: Word Tokenization

Now that we’ve split our document into sentences, we can process them one at a time. Let’s start with the first sentence from our document:

“London is the capital and most populous city of England and the United Kingdom.”

The next step in our pipeline is to break this sentence into separate words or tokens. This is called tokenization. This is the result:

“London”, “is”, “ the”, “capital”, “and”, “most”, “populous”, “city”, “of”, “England”, “and”, “the”, “United”, “Kingdom”, “.”

Tokenization is easy to do in English. We’ll just split apart words whenever there’s a space between them. And we’ll also treat punctuation marks as separate tokens since punctuation also has meaning.

Step 3: Text Lemmatization

In English (and most languages), words appear in different forms. Look at these two sentences:

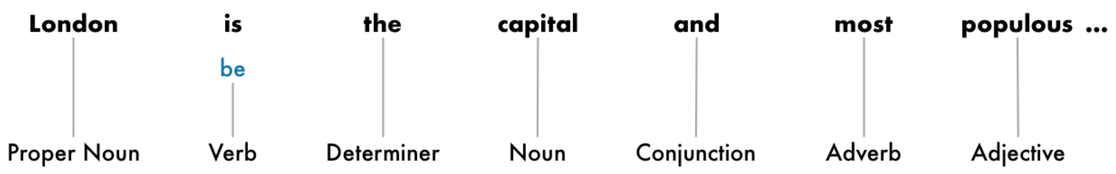
I had a pony.

I had two ponies.

Both sentences talk about the noun pony, but they are using different inflections. When working with text in a computer, it is helpful to know the base form of each word so that you know that both sentences are talking about the same concept. Otherwise the strings “pony” and “ponies” look like two totally different words to a computer.

In NLP, we call finding this process lemmatization—figuring out the most basic form or lemma of each word in the sentence.

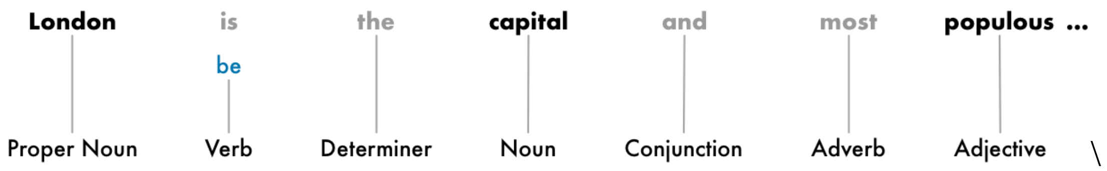
The same thing applies to verbs. We can also lemmatize verbs by finding their root, unconjugated form. So “I had two ponies” becomes “I [have] two [pony].”



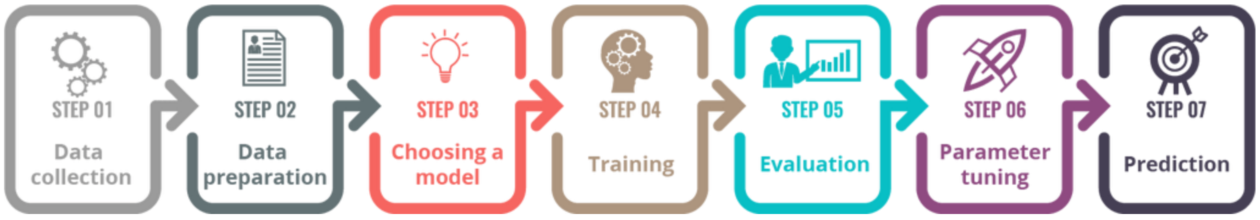
Step 4: Identifying Stop Words

Next, we want to consider the importance of a each word in the sentence. English has a lot of filler words that appear very frequently like "and", "the", and "a". When doing statistics on text, these words introduce a lot of noise since they appear way more frequently than other words. Some NLP pipelines will flag them as stop words —that is, words that you might want to filter out before doing any statistical analysis.

Here's how our sentence looks with the stop words grayed out:



4.1.3 The Model Training Process:



Machine learning is a field of computer science that gives computers the ability to learn without being programmed explicitly. The power of machine learning is that you can determine how to differentiate using models, rather than using human judgment. The basic steps that lead to machine learning and will teach you how it works are described below in a big picture:

1. Gathering Data

Once you know exactly what you want and the equipments are in hand, it takes you to the first real step of machine learning- Gathering Data. This step is very crucial as the quality and quantity of data gathered will directly determine how good the predictive model will turn out to be. The data collected is then tabulated and called as Training Data.

2. Data Preparation

After the training data is gathered, you move on to the next step of machine learning: Data preparation, where the data is loaded into a suitable place and then prepared for use in machine learning training. Here, the data is first put all together and then the order is randomized as the order of data should not affect what is learned.

This is also a good enough time to do any visualizations of the data, as that will help you see if there are any relevant relationships between the different variables, how you can take their advantage and as well as show you if there are any data imbalances present. Also, the data now has to be split into two parts. The first part that is used in training our model, will be the majority of the dataset and the second will be used for the evaluation of the trained model’s performance. The other forms of adjusting and manipulation like normalization, error correction, and more take place at this step.

3. Choosing a model

The next step that follows in the workflow is choosing a model among the many that researchers and data scientists have created over the years. Make the choice of the right one that should get the job done.

4. Training

After the before steps are completed, you then move onto what is often considered the bulk of machine learning called training where the data is used to incrementally improve the model’s ability to predict.

The training process involves initializing some random values for say A and B of our model, predict the output with those values, then compare it with the model's prediction and then adjust the values so that they match the predictions that were made previously.

This process then repeats and each cycle of updating is called one training step.

5. Evaluation

Once training is complete, you now check if it is good enough using this step. This is where that dataset you set aside earlier comes into play. Evaluation allows the testing of the model against data that has never been seen and used for training and is meant to be representative of how the model might perform when in the real world.

6. Parameter Tuning

Once the evaluation is over, any further improvement in your training can be possible by tuning the parameters. There were a few parameters that were implicitly assumed when the training was done. Another parameter included is the learning rate that defines how far the line is shifted during each step, based on the information from the previous training step. These values all play a role in the accuracy of the training model, and how long the training will take.

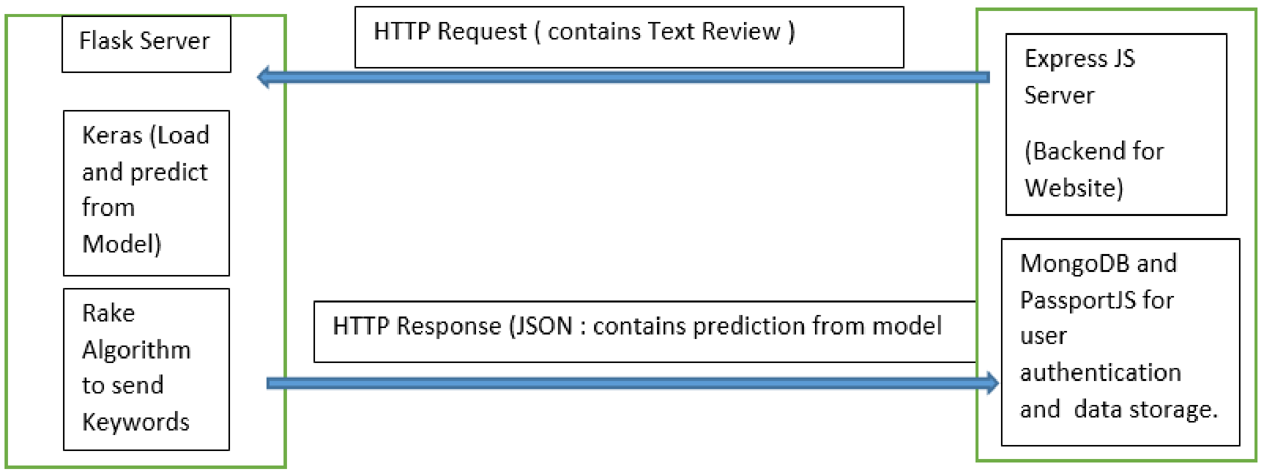
For models that are more complex, initial conditions play a significant role in the determination of the outcome of training. Differences can be seen depending on whether a model starts off training with values initialized to zeroes versus some distribution of values, which then leads to the question of which distribution is to be used. Since there are many considerations at this phase of training, it's important that you define what makes a model good. These parameters are referred to as Hyper parameters. The adjustment or tuning of these parameters depends on the dataset, model, and the training process. Once you are done with these parameters and are satisfied you can move on to the last step.

7. Prediction

Machine learning is basically using data to answer questions. So this is the final step where you get to answer few questions. This is the point where the value of machine learning is realized. Here you can finally use your model to predict the outcome of what you want.

The above-mentioned steps take you from where you create a model to where you Predict its output and thus acts as a learning path.

Architecture of the Project:


Flask Server
Keras (Load and predict from Model)
Rake Algorithm to send Keywords
HTTP Request (contains Text Review)
HTTP Response (JSON : contains prediction from model
Express JS Server
(Backend for Website)
MongoDB and PassportJS for user authentication and data storage.

We used the train models to create a feedback system to inform doctors about the negative reviews and the keywords used in the negative reviews. We use our machine learning model to detect if the new review is negative. After the detection of the polarity of review, we use the rave algorithm to extract keywords from the review and then ask the reviewer about more details about certain aspect of the hospital they might want to complain about. This form will show up only if the review is negative and will stay hidden if the patient has positive review for the hospital.

The information entered in the feedback form will be coupled with the keywords extracted from the rake machine learning model to provide high quality feedback to hospitals to focus on things that are commonly complained by the patients. This will help improve not only customer service but also the quality of healthcare and also help reduce the costs spend on improving the hospitals as the hospital will know where the money is to be invested to improve the patient satisfaction

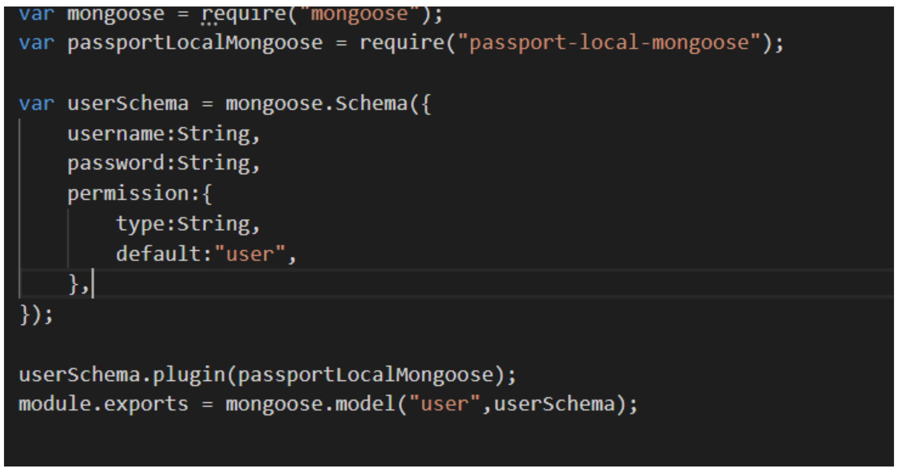
We deploy our model as a flask web service which can be used to give predictions on textual data. The service can be accessed using an http request and the output of the request is the prediction in JSON format. We also developed a website using expressJS, mongodB database and passportJS for user authentication and simple bootstrap for giving UI. The website developed uses the machine learning model to show feedback form as well as extract the keywords from reviews to submit to the specific doctor which has been reviewed. The feedback system developed can be used by any website and the models can be trained on any dataset to be used in any domain. Below is the diagram that shows how the machine learning model and our website work together.

Model Used For Feedback System For the feedback system, we chose bidirectional LSTM as the model of our choice since it gave us the best accuracy as well as validation accuracy. The precision and recall metrics score was also very good for the model.

Databases used for the website

For the Placeholder website, we used mongodB as our choice for database storage service. MongoDB is an open source database management system (DBMS) that uses a document-oriented database model which supports various forms of data. It is one of numerous nonrelational database technologies which arose in the mid-2000s under the NoSQL banner for use in big data applications and other processing jobs involving data that doesn't fit well in a rigid relational model. Instead of using tables and rows as in relational databases, the MongoDB architecture is made up of collections and documents.In our Project , we made use of three databases :

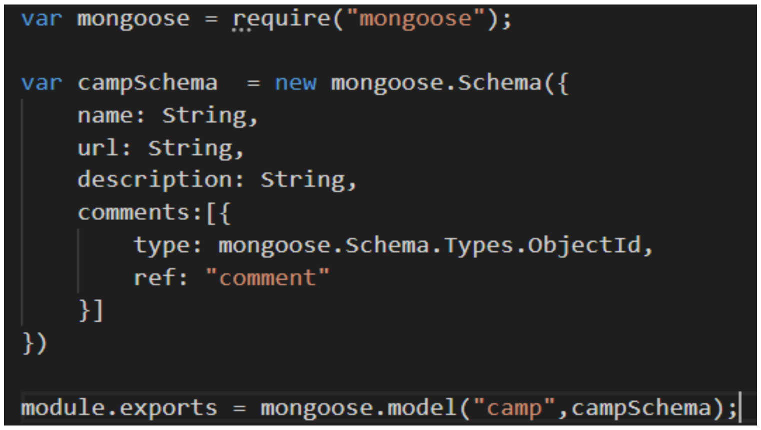
USER



```
var mongoose = require("mongoose");
var passportLocalMongoose = require("passport-local-mongoose");

var userSchema = mongoose.Schema({
    username:String,
    password:String,
    permission:{
        type:String,
        default:"user",
    },
});

userSchema.plugin(passportLocalMongoose);
module.exports = mongoose.model("user",userSchema);
```

To Register users to the website and to manage their permissions. The users were authenticated with passportJS , an authentication library.

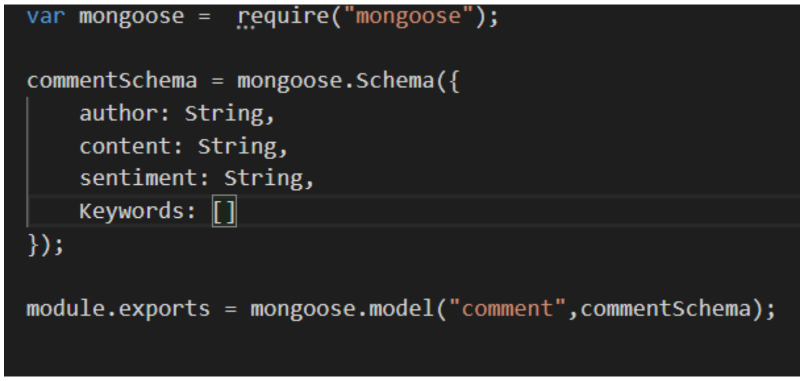
POSTS



```
var mongoose = require("mongoose");

var campSchema  = new mongoose.Schema({
    name: String,
    url: String,
    description: String,
    comments:[{
        type: mongoose.Schema.Types.ObjectId,
        ref: "comment"
    }]
})

module.exports = mongoose.model("camp",campSchema);
```

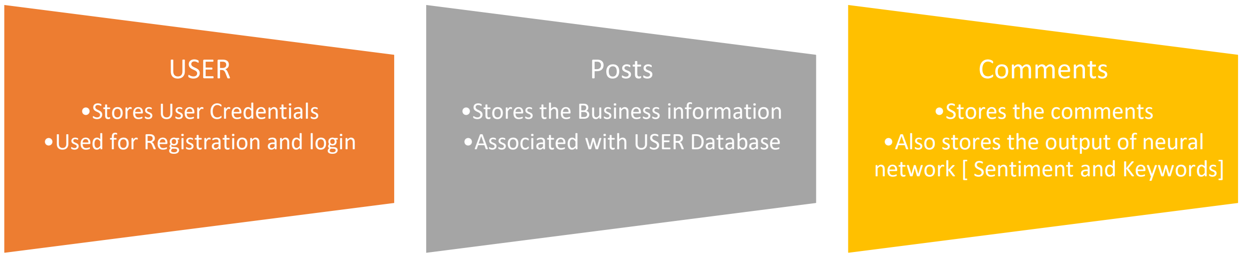
To Store the business information and display them on the web app. The database is associated with another database called “Comments” which is used to store comments on the business.

COMMENTS

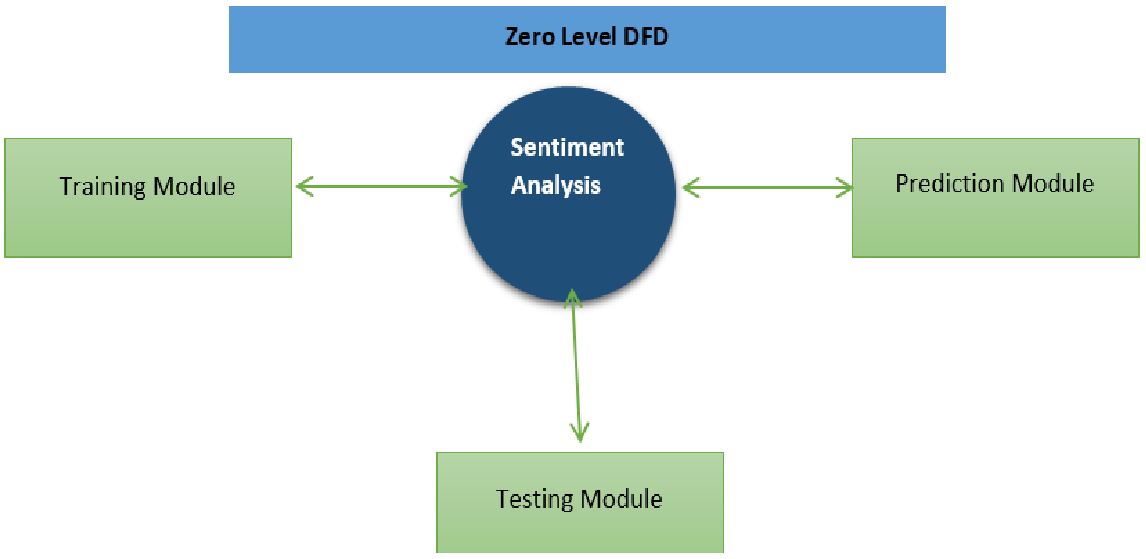


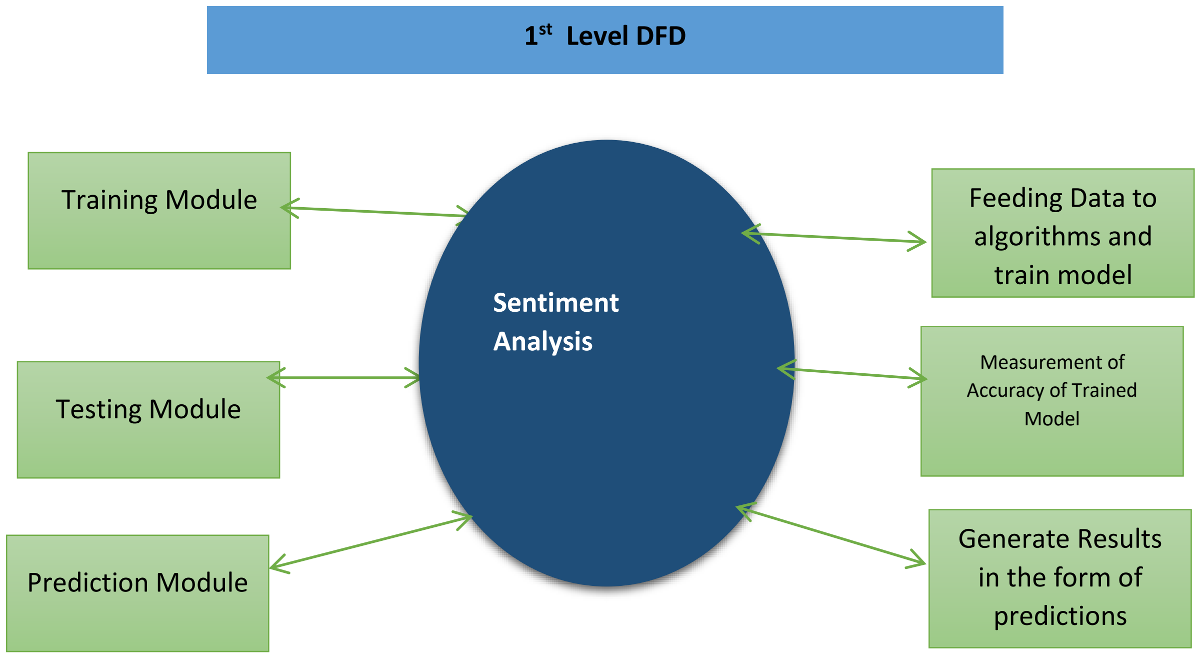
```
var mongoose =  require("mongoose");

commentSchema = mongoose.Schema({
    author: String,
    content: String,
    sentiment: String,
    Keywords: []
});

module.exports = mongoose.model("comment",commentSchema);
```

This Database is used to store comments. It has four fields, the author and the content that is the comment itself. Sentiment field is used to store the sentiment predicted by the neural network. Keywords stores an array of words that are important in a review that is their “KEYWORD”.

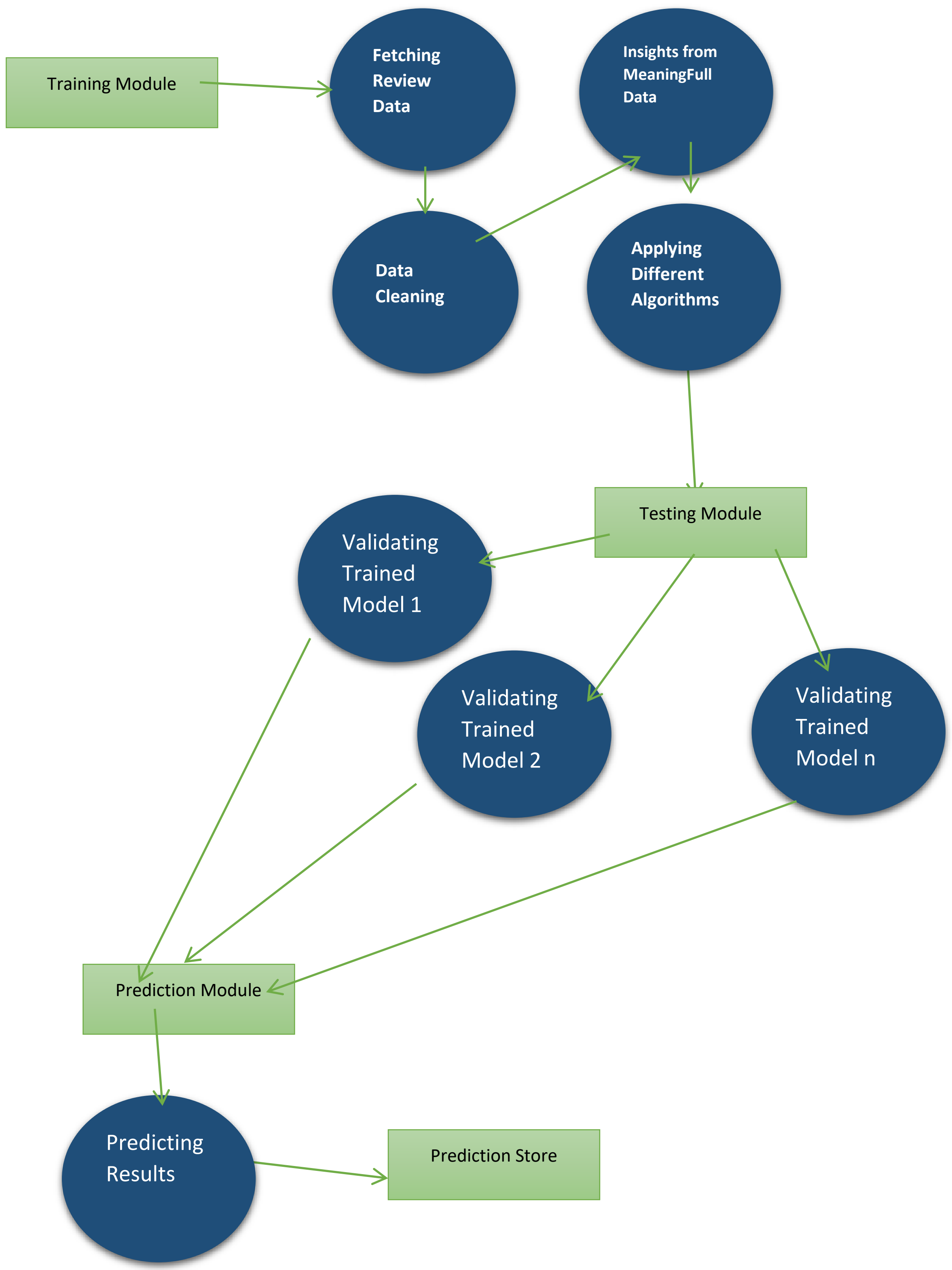

USER
•Stores User Credentials
•Used for Registration and login
Posts
•Stores the Business information
•Associated with USER Database
Comments
•Stores the comments
•Also stores the output of neural network [Sentiment and Keywords]

4.2 DFD for the Project:




1st Level DFD
Training Module
Testing Module
Prediction Module
Sentiment Analysis
Feeding Data to algorithms and train model
Measurement of Accuracy of Trained Model
Generate Results in the form of predictions

2nd Level DFD


Training Module
Fetching Review Data
Data Cleaning
Insights from MeaningFull Data
Applying Different Algorithms
Testing Module
Validating Trained Model 1
Validating Trained Model 2
Validating Trained Model n
Prediction Module
Predicting Results
Prediction Store

CHAPTER 5

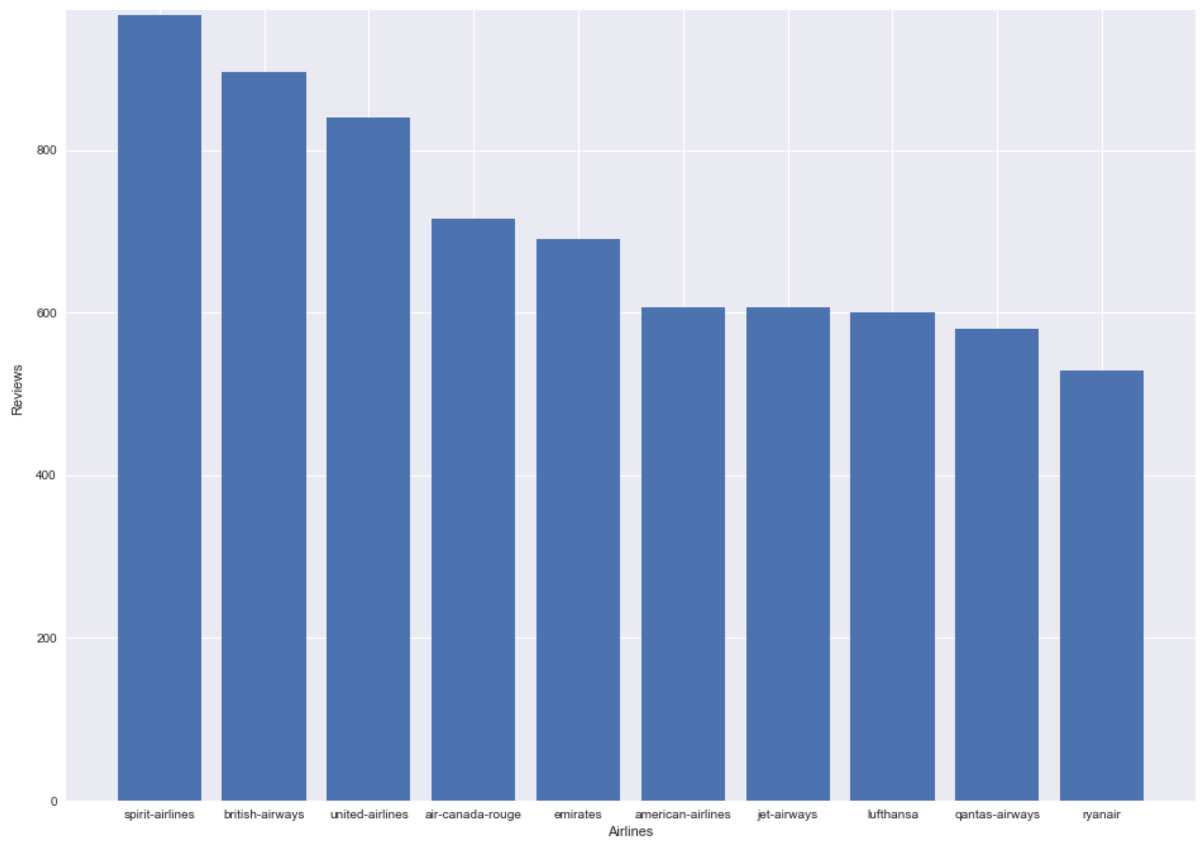
5.1 Implementation

- **The Data Science Process Code**

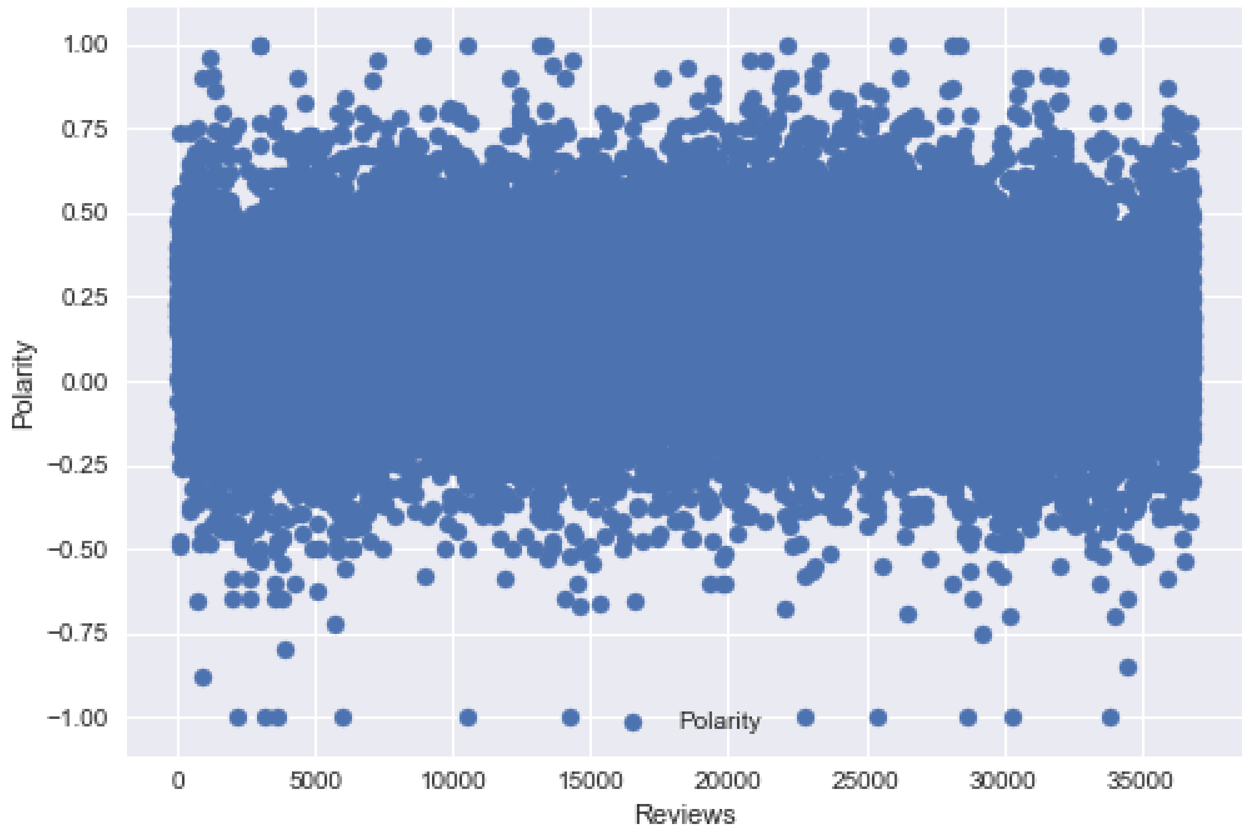
Exploratory Data Analysis

The review data is collected from the skytraxratings.com online freely available repository. This data is then processed and explored before diving into the project.

The observed top 10 most reviewed airlines according to the received data:



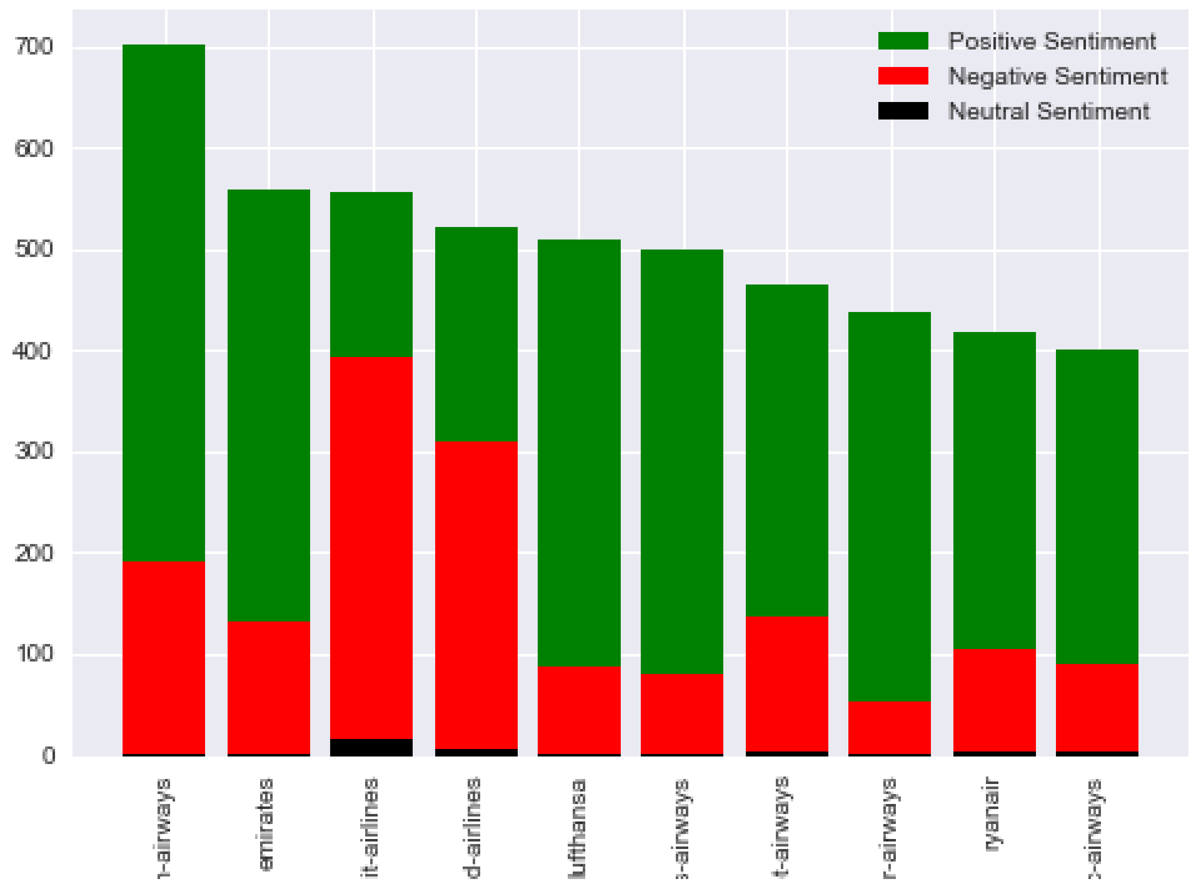
Since the data received is not labelled with polarities so by using the textblob library from python we firs determined the polarities of the each review on the scale of -1 to 1 and then assigned the review classification in three categories i.e -1 -> Negative, 0 -> Neutral and 1 -> Positive Review. On the basis of these categories the further investigation is carried out.



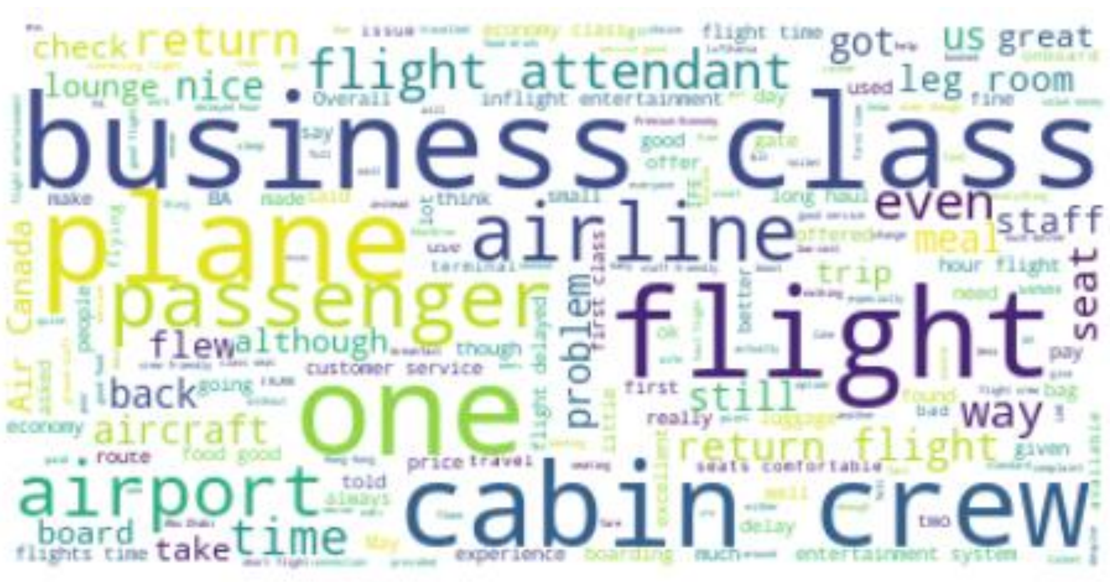
After this the average positive, average neutral and average negative reviews of each airline is calculated and visualize using the matplotlib library of python in the form of bar chart for the further data science process. This will provide the clear picture about the reputation of each airline that is being used for the data analysis.

Scatterplot of the polarities of the reviews give the picture about the reviews that they are uniformly divided on the scale of -1 to 1. The total of 36861 reviews are analysed for 357 different airlines.

I consider the polarity calculation as the best method to assign the sentiment value to the reviews instead of provided average rating because in the data repository above 90% of average rating data was missing and the provided average rating can’t be trusted to train the classical as well as deep neural networks.



The wordcloud from the reviews shows the most keywords about which the emotion is expressed by the customer whether he satisfied with the service or not.



Pre-processing

```
import pandas as pd
import numpy as np

airline_data=pd.read_csv('airline.csv')
airline_data.head(1)
reviews=airline_data.content.str.lower()

import re
import string
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.tokenize import word_tokenize

# Remove Numbers
reviews1=[]
for i in reviews:
    reviews1.append(re.sub(r'\d+', '', i))

# Remove Punctuation
reviews2=[]
for i in reviews1:
    table = str.maketrans({key: None for key in string.punctuation})
    reviews2.append(i.translate(table))

# Remove Whitespaces
reviews3=[]
for i in reviews2:
    reviews3.append(i.strip())

# Removing Stop Words
# stop_words = set(stopwords.words('english'))
tokenize_reviews_no_stop_words=[]
for i in reviews3:
    tokens=tokenize(i)
    result=[j for j in tokens if not j in stop_words]
    temp=' '.join(result)
    tokenize_reviews_no_stop_words.append(temp)
```

Providing Polarity Sentiment Values

```
from textblob import TextBlob
polarity=[]
numlist=[]
```

```
num=1
for i in tokenize_reviews_no_stop_words:
    temp=TextBlob(i).sentiment.polarity
    polarity.append(temp)
    numlist.append(num)
    num=num+1

import matplotlib.pyplot as plt
plt.style.use('seaborn')
plt.scatter(numlist,polarity,label='Polarity')
plt.xlabel('Reviews')
plt.ylabel('Polarity')
plt.legend()

sentiment=[]
for i in polarity:
    if i==0:
        sentiment.append(0)
    elif i<0:
        sentiment.append(-1)
    elif i>0:
        sentiment.append(1)

combined=[]
for i in range(len(sentiment)):
    combined.append([tokenize_reviews_no_stop_words[i],sentiment[i]])

df=pd.DataFrame(combined,columns=['Reviews','Sentiment'])
df.head()
```

- **Model Preparation**

Long short term memory artificial neural network

```
from keras.layers import
Dense,Conv1D,MaxPool1D,Embedding,Flatten,Dropout,GRU,LSTM, Bidirectional
from keras.preprocessing.text import Tokenizer
from keras.models import Sequential
from keras.preprocessing.sequence import pad_sequences
from sklearn.preprocessing import OneHotEncoder
from keras.optimizers import Adam
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
import numpy
import pandas
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import LabelEncoder
from sklearn.pipeline import Pipeline

reviews_array=np.array(df.Reviews)
sentiment_array=np.array(df.Sentiment)

# encode class values as integers
encoder = LabelEncoder()
encoder.fit(sentiment_array)
encoded_Y = encoder.transform(sentiment_array)
# convert integers to dummy variables (i.e. one hot encoded)
dummy_y = np_utils.to_categorical(encoded_Y)

token=Tokenizer()
token.fit_on_texts(reviews_array)
vocab_size=len(token.word_index) +1
print(vocab_size)
l = 0
for i in reviews_array:
  l += len(i)

avg_length = l/len(reviews_array)
review_training = [x[:int(avg_length)] for x in reviews_array]

encoded = token.texts_to_sequences(reviews_array)
l = []
for i in encoded:
  l.append(len(i))

padded_docs = pad_sequences(encoded, maxlen=38, padding='post')
n_model1 = Sequential()
n_model1.add(Embedding(48802,64,input_length=38))
```

```
n_model1.add(LSTM(64, activation='tanh', recurrent_activation='hard_sigmoid',
use_bias=True, kernel_initializer='glorot_uniform',
recurrent_initializer='orthogonal', bias_initializer='zeros',
unit_forget_bias=True, kernel_regularizer=None, recurrent_regularizer=None,
bias_regularizer=None, activity_regularizer=None, kernel_constraint=None,
recurrent_constraint=None, bias_constraint=None, dropout=0.0,
recurrent_dropout=0.0, implementation=1, return_sequences=True,
return_state=False, go_backwards=False, stateful=False, unroll=False))
n_model1.add(Dropout(0.5))
n_model1.add(LSTM(64,return_sequences=False))
n_model1.add(Dropout(0.5))
model.add(Dense(8, input_dim=1, activation='relu'))
n_model1.add(Dense(3, activation='softmax'))
n_model1.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
hist =
n_model1.fit(padded_docs,dummy_y,epochs=5,batch_size=100,validation_split=0.2)
```

Initially the web application runs from the __init__ file named as app.py in our project. This is the root file and all the business logic is defined in this application. All the routings of the application is defined in this file and this file controls the whole logic of the application.

- **Business View Code**

```
def preprocess_review(sentence):

    result=re.sub(r'\d+','',sentence)

    table = str.maketrans({key: None for key in string.punctuation})

    result_new=result.translate(table)

    result_new=result_new.strip()

    tokens=word_tokenize(result_new)

    stop_words=set(stopwords.words('english'))

    final=[j for j in tokens if not j in stop_words]

    temp=' '.join(final)

    return temp


def keywords(review):

        rake=Rake()
        rake.extract_keywords_from_text(review)
        keys=rake.get_word_degrees()
        return list(keys.keys())


def prediction_lstm(sentence):
        final=[]
        loaded_model=pickle.load(open('lstmModel0.1.pkl','rb'))
        token=pickle.load(open('lstmToken.pkl','rb'))
        sen=token.texts_to_sequences(sentence)
        padded_docs = pad_sequences(sen, maxlen=38, padding='post')
        proba=loaded_model.predict(padded_docs)
        result=np.argmax(proba)
        if result==0:
                final.append(-1)
        if result==1:
                final.append(0)
        if result==2:
                final.append(1)
        # K.clear_session()
        # loaded_model=pickle.load(open('lstmModel0.1.pkl','rb'))
        # proba=loaded_model.predict(padded_docs)
        final.append(proba[0][0]*100)
        final.append(proba[0][1]*100)
        final.append(proba[0][2]*100)
        K.clear_session()
        return final


@app.route('/lstmresult',methods=['GET','POST'])
```

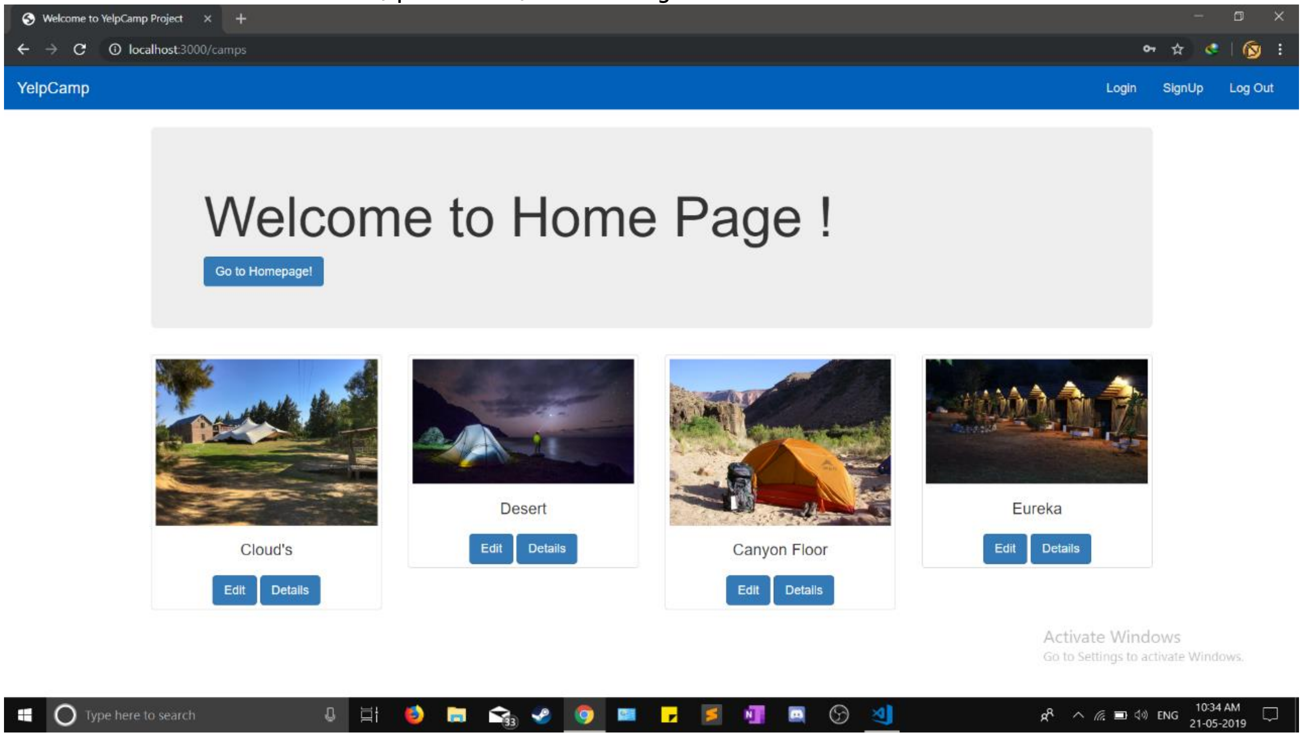
```
def lstm_result():
    if(request.method=='POST' or request.method=='GET'):
        review=request.args.get('review')
        # # sentence=request.form.to_dict()
        # sentence=list(sentence.values())
        keyyy=keywords(review)
        result=prediction_lstm([review])
        sentiment=result[0]
        sentimentresult=''
        if(sentiment==-1):
            sentimentresult='Negative'
        elif(sentiment==0):
            sentimentresult='Neutral'
        elif(sentiment==1):
            sentimentresult='Positive'

        mydict={"Sent":sentimentresult,"Keywords":keyyy}

        return flask.jsonify(mydict)

if (__name__=='__main__'):
                                port = int(os.environ.get("PORT", 5000))
                                app.run(debug=True, port=port)
```

- **Presentation View Code**
  <u>Main Project:</u>
  a) App Component:

```
mongoose.connect("mongodb://localhost/yelpcamp",{useNewUrlParser:true})
;
app.use(eSession({
    secret:"This app is made by Dinesh Pabbi",
    saveUninitialized:false,
    resave:false

}));
app.use(expressBack());
app.use(bodyParser.urlencoded({extended:true}));
app.use(methodOverride("_method"));
app.use(passport.initialize());
app.use(passport.session());
passport.use(new localStrategy(user.authenticate()));
passport.serializeUser(user.serializeUser());
passport.deserializeUser(user.deserializeUser());


app.get("/",(req,res)=>{
    console.log(req.user);
    res.render("home.ejs");
})

app.get("/camps/new",isAdmin,(req,res)=>{
    res.render("new.ejs")
})


app.post("/camps",(req,res)=>{
    camp.create(req.body.c,(err,camp)=>{
        if(err){
            console.log(err);
        }else{
            res.redirect("/camps");
        }
    })

})

app.put("/camps/:id",(req,res)=>{
    camp.findByIdAndUpdate(req.params.id,req.body.c,(err,result)=>{
        if(err){
            console.log(err);
        }else{
            res.redirect("/camps");
        }
    })
});

app.get("/login",(req,res)=>{
    res.render("login.ejs");
})

app.post("/login",passport.authenticate("local",{
    successRedirect: "/camps",
    failureRedirect:"/login"
}),(req,res)=>{});

app.get("/signup",(req,res)=>{
    res.render("signup.ejs");
});

app.post("/signup",(req,res)=>{

user.register({username:req.body.username},req.body.password,(err,resul
t)=>{
        if(err){
            console.log(err);
            res.redirect("/signup");
        }
        passport.authenticate("local")(req,res,function(){
            res.redirect("/");
        })
    });
})

app.get("/camps",(req,res)=>{
    camp.find({},(err,camps)=>{
        if(err){
            console.log(err);
        }else{
            res.render("camps.ejs",{camps:camps});
        }
    })

});

app.get("/camps/:id",(req,res)=>{

camp.findById(req.params.id).populate("comments").exec((err,result)=>{
        if(err){
            console.log(err);
        }else{
            res.render("desc.ejs",{c:result});
        }
    });
});

app.get("/camps/:id/edit",isAdmin,(req,res)=>{
    camp.findById(req.params.id,(err,result)=>{
        if(err){
            console.log(err);
        }else{
            res.render("edit.ejs",{camp:result});
        }
    });

});

app.get("/logout",(req,res)=>{
    req.logout();
    res.redirect("/");
})

app.get("/camps/comment/:id",isLoggedIn,(req,res)=>{
    res.render("comment.ejs",{id:req.params.id});
});

app.post("/camps/comment/:id",(req,res)=>{
    var com = new comment(req.body.c);
    com.save();
    camp.findById(req.params.id,(err,result)=>{
        if(err){
            console.log(err);
        }else{
            result.comments.push(com);
            result.save();
            var u = "https://dinesh-sentiment-
api.herokuapp.com/lstmresult?review="+com["content"];
            request(u,(err,response,body)=>{
                if(err){
                    console.log(err);
                }else{
                    output = JSON.parse(body);
                    keywords = output["Keywords"].join();
                    if(output["Sent"] == "Negative"){

res.redirect("/feedback/"+keywords+"/"+req.params.id);
                    }else{
                        res.redirect("/camps/"+req.params.id);
                    }
                }
            });
        }
    });
});

app.get("/feedback/:keywords/:id",(req,res)=>{
    res.render("feedback.ejs",{keywords:req.params.keywords})
});

function isLoggedIn(req,res,next){
    if(req.isAuthenticated()){
        return next();
    }else{
        res.redirect("/login");
    }
}
function isAdmin(req,res,next){
    if(req.isAuthenticated()){
        user.findOne({username:req.user.username},(err,u)=>{
            if(err){
                console.log(err);
                res.redirect("/login");
            }else{
                if(u.permission == "admin"){
                    return next()
                }else{
                    res.redirect("/");
                }
            }
        });
    }else{
        res.redirect("/login");
    }
}
app.listen(3000,()=>{console.log("Server is up and running !")});
```

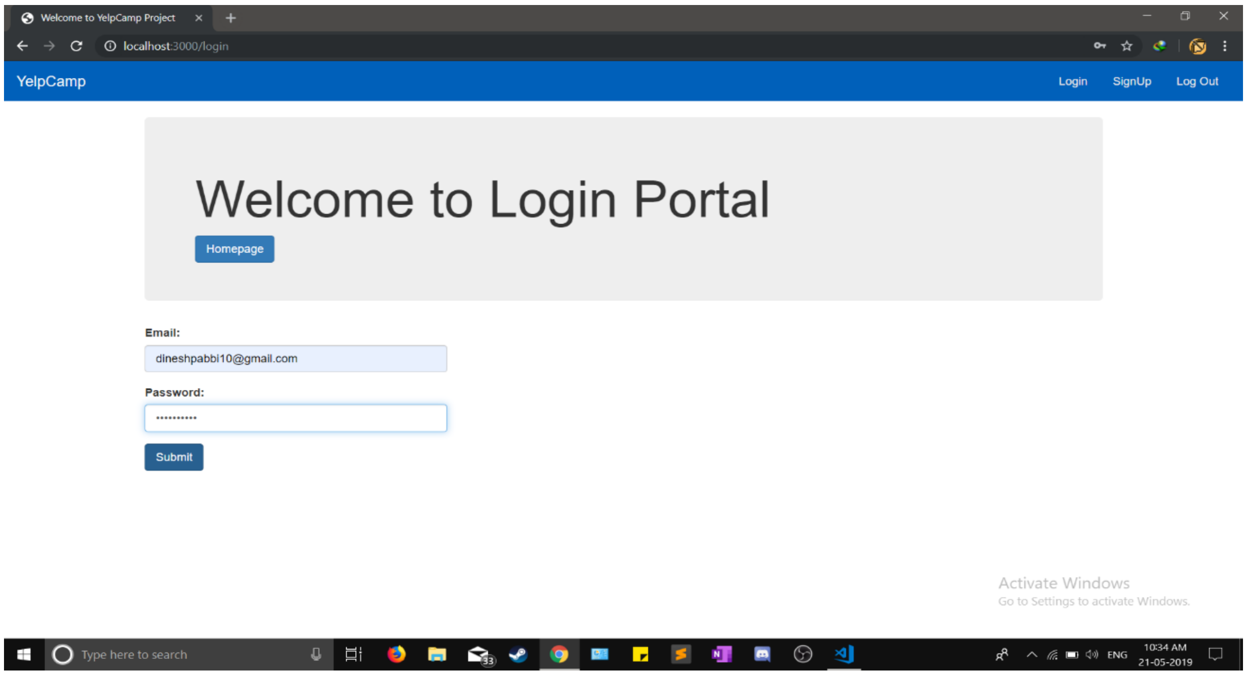
Home Page

```
<% include ./partials/header.ejs %>

<div class = "container">
    <header class="jumbotron">
        <h1>Welcome to Home Page !</h1>
        <a class = "btn btn-primary btn-large" href="/">Go to
Homepage!</a>
     </header>

    <div class="row" style = "display: flex;flex-wrap: wrap">
        <% camps.forEach((camp)=>{ %>
            <div class = "col-md-3 col-sm-6">
                <div class="thumbnail">
                    <img class = "img-responsive" src= <%= camp.url
%> >
                    <center>
                        <h4 class="caption"><%= camp.name %></h4>
                        <a class = "btn btn-primary btn-large"  href
= "camps/<%= camp._id %>/edit">Edit</a>
                        <a class = "btn btn-primary btn-large"  href
= "camps/<%= camp._id %>">Details</a>
                    </center>
                </div>
            </div>
        <% }) %>
</div>
</div>

<% include ./partials/footer.ejs %>
```



Login Portal

```
<% include ./partials/header.ejs %>

<div class="container">
        <header class="jumbotron">
            <h1> Welcome to Login Portal</h1>
            <a class="btn btn-primary btn-large"
href="/">Homepage</a>
        </header>
        <div class = "row">
            <div class ="col-md-4 col-sm-6">
                <form action = "/login" method="POST">
                    <div class="form-group">
                        <label>Email:</label>
                        <input class="form-control" type="email"
name = "username" placeholder="Email">
                    </div>

                    <div class="form-group">
                        <label>Password:</label>
                        <input class="form-control" type="password"
name = "password" placeholder="password">
                    </div>

                    <div class="form-group">
                            <input type="submit" class="btn btn-
primary btn-large">
                    </div>

                </form>
            </div>
        </div>
    </div>

<% include ./partials/footer.ejs %>
```



Welcome to Login Portal

Email:

dineshpabbi10@gmail.com

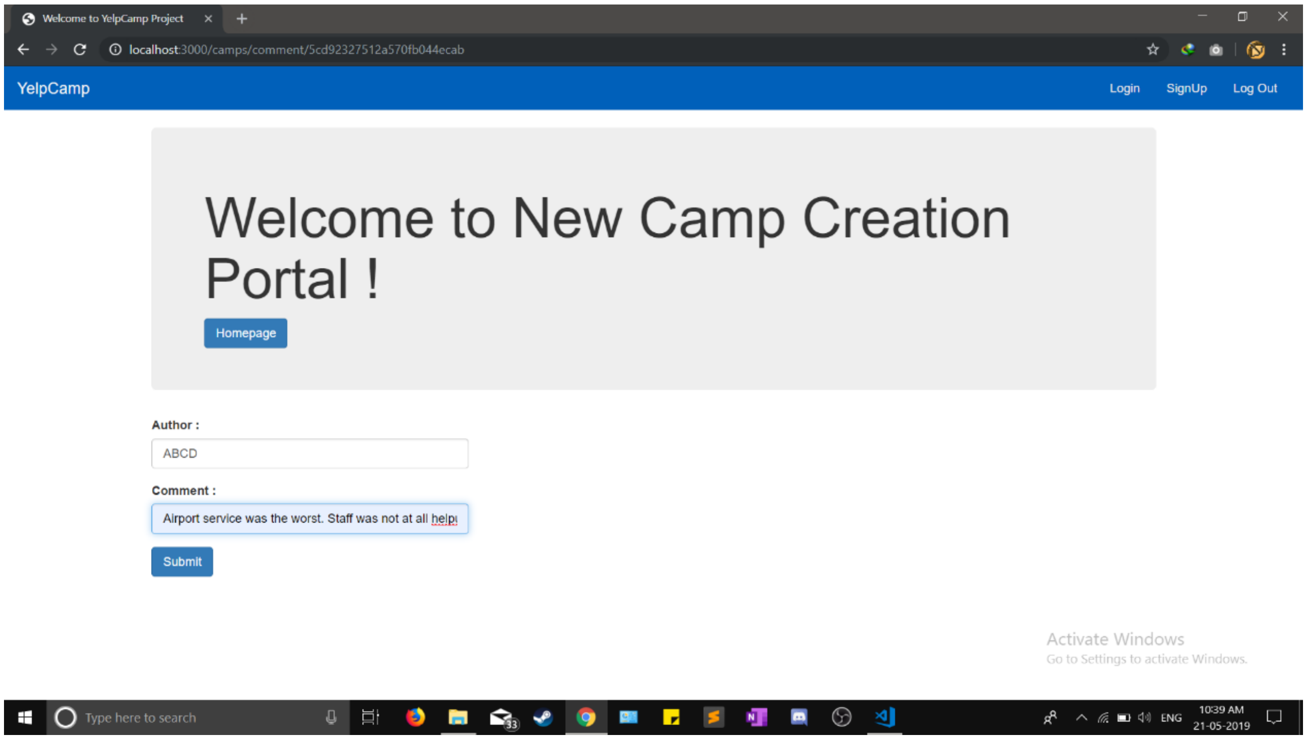
Password:

••••••••••

Comments

```
<% include ./partials/header.ejs %>

<div class="container">
        <header class="jumbotron">
            <h1> Welcome to New Camp Creation Portal !</h1>
            <a class="btn btn-primary btn-large"
href="/">Homepage</a>
        </header>
        <div class = "row">
            <div class ="col-md-4 col-sm-6">
                <form action = "/camps/comment/<%= id %>"
method="POST">
                    <div class="form-group">
                        <label>Author :</label>
                        <input class="form-control" type="text" name
= "c[author]" placeholder="Name">
                    </div>

                    <div class="form-group">
                        <label>Comment :</label>
                        <input class="form-control" type="text" name
= "c[content]" placeholder="url">
```

```
                                    </div>

                                    <div class="form-group">
                                            <input type="submit" class="btn btn-
                primary btn-large">
                                    </div>

                                </form>
                            </div>
                        </div>
                    </div>
                <% include ./partials/footer.ejs %>
```



Welcome to YelpCamp Project

localhost:3000/camps/comment/5cd92327512a570fb044ecab

YelpCamp

Login SignUp Log Out

Welcome to New Camp Creation Portal !

Homepage

Author :

ABCD

Comment :

Airport service was the worst. Staff was not at all helpi

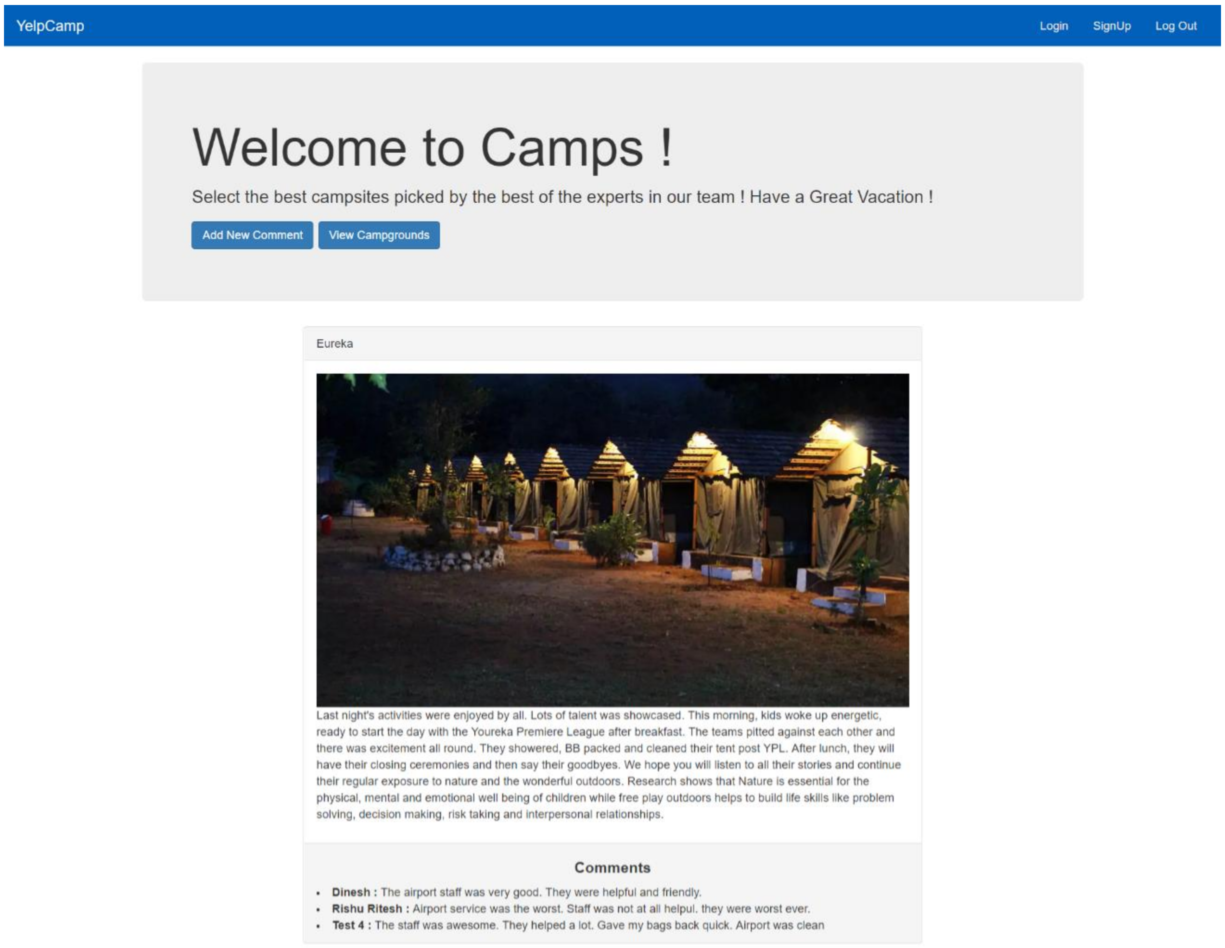
Submit

Activate Windows
Go to Settings to activate Windows.

Detail View

```
<% include ./partials/header.ejs %>
<div class="container">
        <header class="jumbotron">
            <h1> Welcome to Camps !</h1>
            <p>Select the best campsites picked by the best of the
experts in our team ! Have a Great Vacation !</p>
            <p>
```

```
                <a class="btn btn-primary btn-large"
href="/camps/comment/<%= c._id %>">Add New Comment</a>
                <a class="btn btn-primary btn-large" href =
"/camps">View Campgrounds</a>
            </p>

        </header>

<div class="row">
    <div class="col-md-2 col-sm-1 col-xs-1"></div>
    <div class="col-md-8 col-sm-10 col-xs-10">
        <div class="panel panel-default">
            <div class="panel-heading"><%= c.name %></div>
            <div class="panel-body">
            <center>
                    <img class="img-responsive" src = "<%= c.url
%>">
            </center>
            <p><%= c.description %></p>
            </div>
            <div class="panel-footer">
                <center>
                    <h4><b>Comments</b></h4>
                </center>
                <% c.comments.forEach((x)=>{ %>
                    <li>
                        <b><%= x.author  %> : </b> <%= x.content %>
                    </li>
                <% }) %>
            </div>
        </div>
    </div>
    <div class="col-md-2 col-sm-1 col-xs-1"></div>

</div>

</div>
<% include ./partials/footer.ejs %>
```



Welcome to Camps !

Select the best campsites picked by the best of the experts in our team ! Have a Great Vacation !

Eureka

Last night's activities were enjoyed by all. Lots of talent was showcased. This morning, kids woke up energetic, ready to start the day with the Youreka Premiere League after breakfast. The teams pitted against each other and there was excitement all round. They showered, BB packed and cleaned their tent post YPL. After lunch, they will have their closing ceremonies and then say their goodbyes. We hope you will listen to all their stories and continue their regular exposure to nature and the wonderful outdoors. Research shows that Nature is essential for the physical, mental and emotional well being of children while free play outdoors helps to build life skills like problem solving, decision making, risk taking and interpersonal relationships.

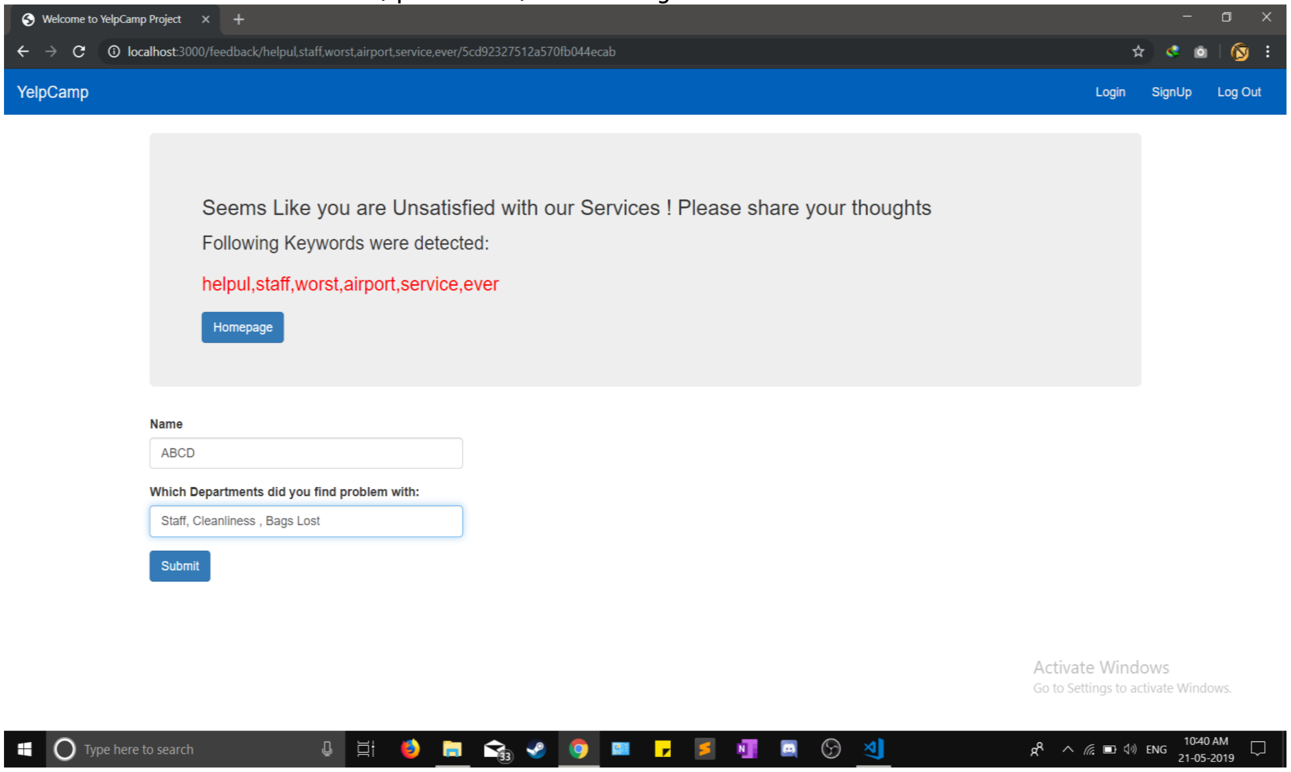
Comments

- **Dinesh :** The airport staff was very good. They were helpful and friendly.
- **Rishu Ritesh :** Airport service was the worst. Staff was not at all helpul. they were worst ever.
- **Test 4 :** The staff was awesome. They helped a lot. Gave my bags back quick. Airport was clean

Feedback Portal

```
<% include ./partials/header.ejs %>

<div class="container">
        <header class="jumbotron">
            <h3> Seems Like you are Unsatisfied with our Services !
Please share your thoughts</h3>
            <p>Following Keywords were detected:</p>
            <p style="color:red;"><%= keywords %></p>
            <a class="btn btn-primary btn-large"
href="/">Homepage</a>
        </header>
        <div class = "row">
            <div class ="col-md-4 col-sm-6">
                <form action = "/" method="GET">
                    <div class="form-group">
                        <label>Name</label>
                        <input class="form-control" type="text" name
= "name" placeholder="Name">
```

```
                    </div>

                    <div class="form-group">
                        <label>Which Departments did you find
problem with:</label>
                        <input class="form-control" type="text" name
= "complaint" placeholder="Complaints">
                    </div>

                    <div class="form-group">
                            <input type="submit" class="btn btn-
primary btn-large">
                    </div>

                </form>
            </div>
        </div>
    </div>

<% include ./partials/footer.ejs %>
```



Welcome to YelpCamp Project

localhost:3000/feedback/helpul,staff,worst,airport,service,ever/5cd92327512a570fb044ecab

YelpCamp — Login — SignUp — Log Out

Seems Like you are Unsatisfied with our Services ! Please share your thoughts

Following Keywords were detected:

helpul,staff,worst,airport,service,ever

Homepage

Name

ABCD

Which Departments did you find problem with:

Staff, Cleanliness , Bags Lost

Submit

Activate Windows
Go to Settings to activate Windows.

Type here to search

10:40 AM
21-05-2019

CHAPTER 6

6.1 TECHNOLOGIES USED

6.1.1 Presentation View/Front End Development Technologies

- **Python-Tkinter (GUI)**

Tkinter is not the only Gui Programming toolkit for Python. It is however the most commonly used one. Cameron Laird calls the yearly decision to keep TkInter "one of the minor traditions of the Python world.". Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

- **HTML5**

HTML5 is the latest and most enhanced version of HTML. Technically, HTML is not a programming language, but rather a mark up language The term represents two different concepts. It is a new version of the language HTML, with new elements, attributes, and behaviours, **and** a larger set of technologies that allows the building of more diverse and powerful Web sites and applications. This set is sometimes called HTML5 & friends and often shortened to just HTML5.

- **CSS**

CSS is used to control the style of a web document in a simple and easy way.CSS is the acronym for "Cascading Style Sheet". It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language. The separation of HTML from CSS makes it easier to maintain sites, share style sheets across pages, and tailor pages to different environments.

- **JavaScript**

JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

- **jQuery**

JQuery is a JavaScript library that allows web developers to add extra functionality to their websites. It is open source and provided for free under the MIT license. In recent years, jQuery has become the most popular JavaScript library used in web development.

- **Bootstrap**

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions.

- **Photoshop**

Photoshop is unlike other common software interfaces which emulate virtual typewriters or graphing paper. Photoshop creates an artist's virtual studio/darkroom. When you open the program you see a toolbox on the left with tools you will use to manipulate your images, and on the right, a white square which is your "canvas" or work area. The grey area surrounding the canvas is not part of your image, but only defines its edges

- **EJS (Embedded Javascript Templating)**

When creating quick on-the-fly Node applications, an easy and fast way to template our application is sometimes necessary. So far we've been building out full MEAN applications and Angular acts as our templating engine.

For applications that need quick templating, there are many options that we can use. Jade comes as the view engine for Express by default but that syntax just flat out scares me. EJS is one alternative does that job well and is very easy to set up. Let's take a look at how we can create a simple application and use EJS to include repeatable parts of our site (partials) and pass data to our views.

6.1.2 Business View/Logical View End Development Technologies

- **Python 3.6**

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.[26] Van Rossum led the language community until stepping down as leader in July 2018. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- **NumPy**

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

- **Pandas**

In computer programming, **pandas** is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals

- **Keras**

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System),and its primary author and maintainer is François Chollet, a Google engineer. Chollet also is the author of the XCeption deep neural network model[4].

In 2017, Google's TensorFlow team decided to support Keras in TensorFlow's core library. Chollet explained that Keras was conceived to be an interface rather than a standalone machine-learning framework. It offers a higher-level, more intuitive set of abstractions that make it easy to develop deep learning models regardless of the computational backend used.Microsoft added a CNTK backend to Keras as well, available as of CNTK v2.0.

- **SciKit**

Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Scikit-learn is largely written in Python, with some core algorithms written in Cython to achieve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR.

- **Matplotlib**

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter, has an active development community,and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012, and further joined by Thomas Caswell.

- **NLTK**

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language. It was developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the University of Pennsylvania. NLTK includes graphical demonstrations and sample data. It is accompanied by a book that explains the underlying concepts behind the language processing tasks supported by the toolkit, plus a cookbook.

NLTK is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, cognitive science, artificial intelligence, information retrieval, and machine learning. NLTK has been used successfully as a teaching tool, as an individual study tool, and as a platform for prototyping and building research systems. There are 32 universities in the US and 25 countries using NLTK in their courses. NLTK supports classification, tokenization, stemming, tagging, parsing, and semantic reasoning functionalities.

- **TensorFlow**

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 on November 9, 2015.

6.1.3 Database View/Back End Development Technologies

- **MongoDB**

MongoDB is a relational database management system . As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet).

- **CSV**

CSV is a simple **file** format used to store tabular data, such as a spreadsheet or database. **Files** in the **CSV** format can be imported to and exported from programs that store data in tables, such as Microsoft Excel or Open Office Calc. **CSV** stands for "comma-separated values"

- **Flask(For developing API)**

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries.[3] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program.

- **ExpressJS(For developing Application)**

Express.js, or simply Express, is a web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js.

The original author, TJ Holowaychuk, described it as a Sinatra-inspired server, meaning that it is relatively minimal with many features available as plugins. Express is the backend component of the MEAN stack, together with the MongoDB database software and AngularJS frontend framework.

- **PassportJS**

Passport is authentication middleware for Node.js. Extremely flexible and modular, Passport can be unobtrusively dropped in to any Express-based web application. A comprehensive set of strategies support authentication using a username and password, Facebook, Twitter, and more.

- **Mongoose**

Mongoose is an “elegant MongoDB object modeling for Node.js“. If you’ve used MongoDB before and tried basic database operations, you might have noticed that MongoDB is “schema-less”. When you’re looking to implement a more structured database and want to leverage the power of MongoDB, Mongoose is one of the ODM (Object Data Mapping) solutions.

MongoDB will never complain about the variation in the number of columns (key-value pairs). This is very flexible. But when you want to keep your data more organized and structured, you would need to maintain that in your server code, writing validation, making sure nothing irrelevant is stored in a collection. This is where Mongoose makes life easy.

- **Heroku**

Heroku is a cloud platform as a service (PaaS) supporting several programming languages. Heroku, one of the first cloud platforms, has been in development since June 2007, when it supported only the Ruby programming language, but now supports Java, Node.js, Scala, Clojure, Python, PHP, and Go. For this reason, Heroku is said to be a polyglot platform as it has features for a developer to build, run and scale applications in a similar manner across most languages. Heroku was acquired by Salesforce.com in 2010 for $212 million.

CHAPTER 7

7.1) Comparison Of Different Models Trained

Model	**Precision**	**Recall**	**F1-Score**	**Validation Accuracy**	**Word Representation**
Dense NN	0: .62 1: .89	0: .56 1: .91	0: .59 1: .90	83.8%	Word Embeddings
Convolution 1-D NN	0: .58 1: .89	0: .59 1: .89	0: .59 1: .89	82.7%	Word Embeddings
GRU	0: .58 1: .88	0: .52 1: .90	0: .55 1: .89	82.3%	Word Embeddings
LSTM	0: .63 1: .91	0: .66 1: .90	0: .64 1: .90	84.8%	Word Embeddings
Bi-Directional LSTM	0: .64 1: .91	0: .67 1: .90	0: .65 1: .91	85.2 %	Word Embeddings
Logistic Regression	0: .71 1: .87	0: .46 1: .95	0: .56 1: .91	84.8%	TF-IDF
Linear SVM	0: .69 1: .91	0: .66 1: .92	0: .67 1: .92	86.7%	TF-IDF
Naïve Bayes	0: 1.0 1: .8	0: .05 1: 1.0	0: .09 1: .89	80.2%	TF-IDF
Logistic Regression	0: .69 1: .90	0: .61 1: .93	0: .64 1: .91	86.1%	CountVector
Linear SVM	0: .70 1: .91	0: .64 1: .93	0: .67 1: .92	86.8%	CountVector
Naïve Bayes	0: .75 1: .88	0: .48 1: .96	0: .59 1: .92	85.9%	CountVector

Conclusion On Comparison of Models:

In our models, we used three types of word/text representation techniques to feed data as input. Word Embeddings are the best methods for word representation when the models being used are neural networks. They help us capture the context and help visualize the words in the multidimensional space. For the classical machine learning models such as logistic regression and naïve bayes, we make use to two commonly used word representations called “TF-IDF Vector” and “Count Vector”. In order to catch the context of up to two words, we make use of N-Gram approach and hence we take into consideration 1-2 gram.

Increasing the words to be considered increased the vector representation length to exceedingly high which was costly on memory and that is the reason both these methods are only suitable for small datasets because as the vocabulary size increases the footprint of the vector used to represent data increases a lot and it becomes impractical to train classical models on such huge data.

From the results, it can be observed that Linear SVM still manages to outperform all the models in terms of validation accuracy. The precision and recall scores are also among the best for LinearSVM making it the most accurate model among the models being considered. Surprisingly, it manages to outperform even Bidirectional LSTM which is well known for its efficiency in handling textual data. This can be attributed to the fact that neural networks typically tend to perform well if the dataset is huge and the dataset under consideration has about 7000 reviews which is very small. On the other hand, SVM is considered the best performer on small datasets and hence clearly shows its prowess in the results.

If we consider neural networks, clearly, bidirectional LSTM and LSTM perform the best in the classification task both having a validation accuracy of about 85% which is very good considering the dataset in very small. Both LSTM and Bidirectional LSTM show very comparable precision and recall scores to the LinearSVM model which further shows the strength of LSTM on textual data. Other neural networks such as Convolution 1-D, GRU and Dense Neural Nets lag behind LSTM in terms of both validation accuracy score and the metrics (precision and recall) score.

It is conspicuous that CountVector representation of textual data consistently gave better results on all the classical machine learning models. Naïve Bayes particularly had a big jump on validation accuracy increasing from 80.2% to 85.9%. Logistic Regression had a jump of about 2% in validation accuracy when using CountVector model for word representation. Consequently, All the classical models displayed better precision and recall scores when TF-IDF was used to feed data.

Overall, the neural networks lag behind classical models in terms in terms of validation accuracy, precision and recall score. This clearly shows that neural networks though being powerful need huge amount of data to show exemplary accuracies. The classical model, hence perform better one small datasets and they usually do not generalize well to huge datasets. The reason for classical models not generalizing well to huge datasets and also performing well on small datasets lie in the fact that the representation of textual data used in these classical model allows them to perform well but as the data increases, it becomes difficult to use the word representation techniques used in these models as the memory footprint would be too large to be handled by such models.

On the other hand, neural networks can easily handle large data by increasing the depth of network and hence they perform better on large datasets. In our case, we are targeting the domain of doctor reviews and it makes sense that classical models perform well on a small dataset targeted at this particular domain.

However , we still choose Bi-Directional LSTM as our model of choice because it was able to generalize well as we gather more data. This will help the website become more efficient as we gather data from users.

7.3) **Conclusion On The Feedback System:**

Machine learning can be used to make predictions based on historical feedback data. This gives marketers good insight into what their visitors may do in the future, whether that’s how likely they are to churn, what they will do next, or even ways to keep the visitor as a loyal customer

Natural Language Processing is relevant for insights such as the sentiment behind feedback results and grouping strings of text together to quickly uncover patterns and trends within the feedback.

Businesses can benefit from an intelligent feedback system as the system allows for extracting the gist of the reviews that is the opinions. In case there are tonnes of negative reviews being left on a business portal and the business is unable to read whole reviews it becomes difficult for the service/business to know where the things are going wrong. This can lead to unprofitable investment on departments which might not even be the cause of negative reviews.

Hence , to save time and to provide targeted feedback to businesses and in the process make the customer feel valuable , a neural network based feedback system that continuously detects whether new reviews being entered are positive or not can help fire a special feedback form for customers. Also the network can also extract the keywords as it is trained on this task. Such keywords can help business understand what the user are talking about in the reviews. Hence they can understand which department is lacking and which department needs investment.

The system will be able to identify trends and developments and come up with suggestions and insights automatically. These trends are based on all variables of a feedback form, meaning text, scores, metadata, date/time, variables from a data layer (i.e. customer segments) and more.

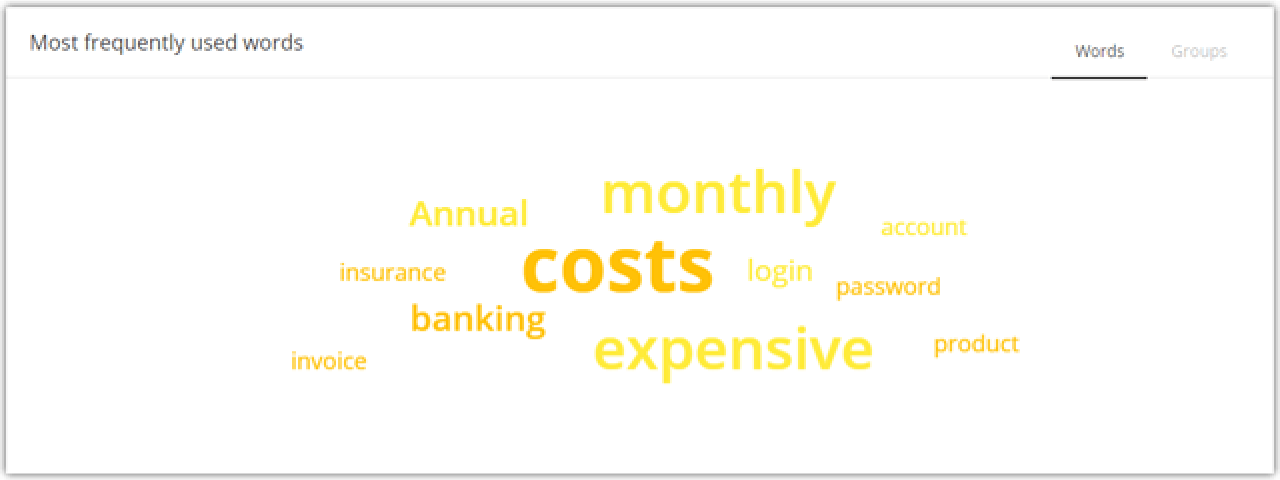
Hence, Text analytics is the gateway for getting in touch with your customers' thoughts and feelings in a more intimate manner. With this type of qualitative analysis, you're not only assessing small-scale issues here and there, but also the entire online customer journey as a whole.

7.4 **Future Scope:**

Transfer Learning is something that might help neural networks further improve accuracy. Pre trained word Embeddings such as glove , Word2Vec by google are trained on huge datasets and hence can help improve the representation of the words. More features can also be added to the feedback system which are as follow:

Word grouping (or word pairing)

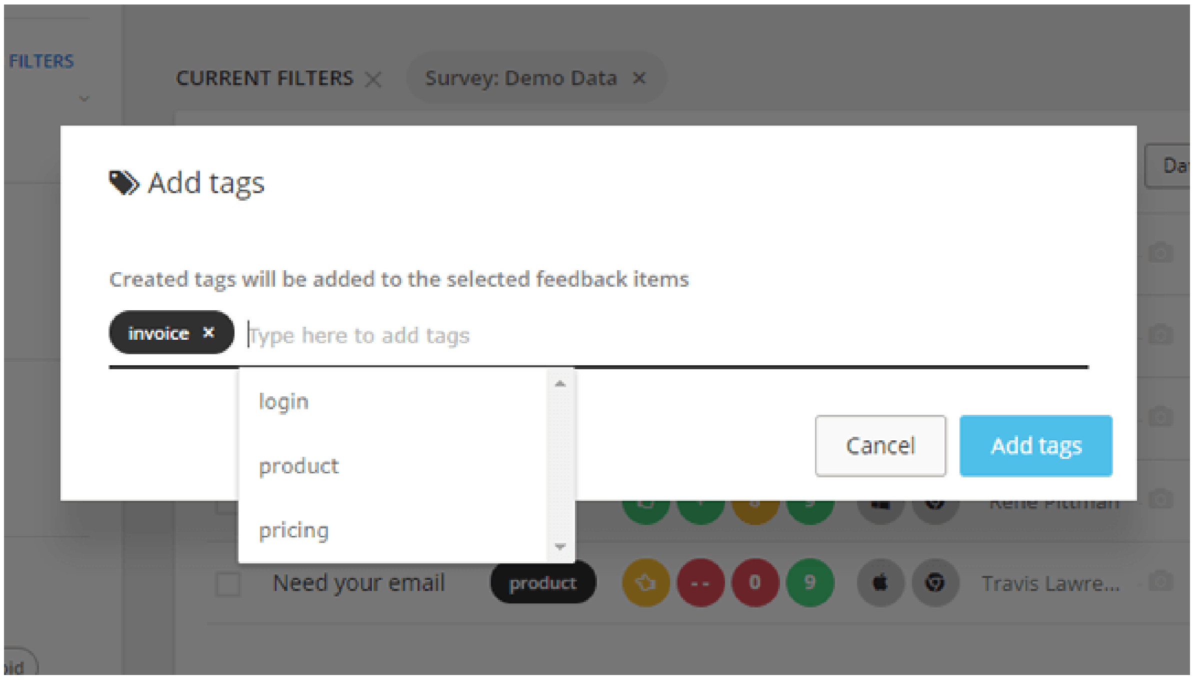
Often times a group of words can provide you with more insight than just one word alone. For example, the words "costs", "expensive" and "monthly" are grouped together. With this information, it's pretty safe to conclude that there are many customers that think the monthly costs for one of your products or services are too expensive. But to take a closer look you can always open the individual comments.



Automatic categorisation using Machine Learning

For example, the user see the following feedback item, 'I can't complete the checkout process'. You could potentially label this with the tags, 'Product' and 'Error'. This is an ideal way of making an increasing pile of customer feedback just a little bit more manageable.

With machine learning, however, we are now able to take this feature one step further. With this technology in place, users who only occasionally label their feedback, for example, can let the system take over, in which case it will categorise items automatically.

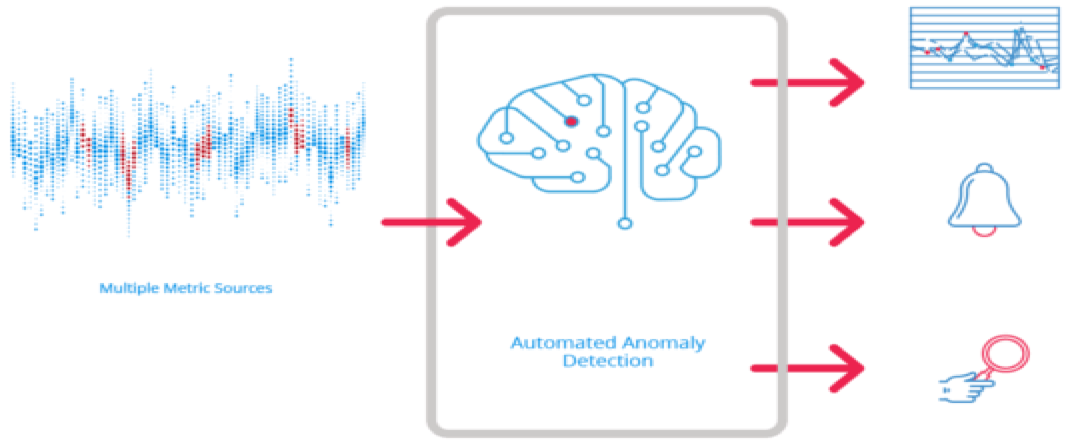


Created tags will be added to the selected feedback items

Insight Feed (Automatic Anomaly Detection)

This is a particularly interesting development we have coming up. Using our anomaly detection algorithm, the system will be able to identify trends and developments and come up with suggestions and insights automatically. These trends are based on all variables of a feedback form, meaning text, scores, metadata, date/time, variables from a data layer (i.e. customer segments) and more.

For example, 'Customers using Chrome are suddenly more negative about the website than customers using Safari', or 'Last week there was considerably more negative feedback from visitors in process x.'



CHAPTER 8

References:

1. https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714 [LSTM]
2. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148 [CNN]
3. https://spacy.io/usage/processing-pipelines [NLP Pipeline]
4. https://towardsdatascience.com/the-7-steps-of-machine-learning-2877d7e5548e [Steps of Machine Learning]
5. https://medium.com/@jaeger.rob/introduction-to-nodes-express-js-db5617047150 [ExpressJS]
6. https://medium.com/@purposenigeria/build-a-restful-api-with-node-js-and-express-js-d7e59c7a3dfb [API development]