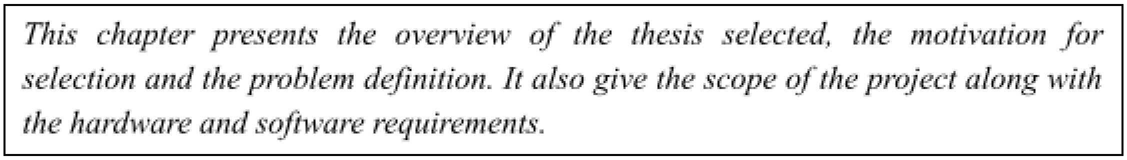
Chapter 1

Introduction



This chapter presents the overview of the thesis selected, the motivation for selection and the problem definition. It also give the scope of the project along with the hardware and software requirements.

1.1 Problem definition

The burgeoning presence of social media and open opportunities for posting personal views have flooded the internet with enormous volumes of opinions catering to all sorts of topics. However, as far as movie reviews are concerned, there are serious bottlenecks when the cost-conscious youth comes to making sense of these opinions. At the same time, the urgency to gain real time updates has necessitated a condensed representation of this information. This project aims at collecting feedback of the people from their comments on social media and then apply sentimental analysis to categorize the movie as good or bad using machine learning and obtain the final rating for a particular movie .

1.2 Scope

- The project involves extracting the comments or reviews from the database of a social media website (eg. TMDB, IMDB , Rotten Tomatoes , Metacritic).
- Each review in the database obtained would have to be analysed.
- The next step would be to identify whether the sentence is subjective or objective.
- Determine whether the sentence expresses positive or negative opinions.

- Based on this it is possible to determine their polarity, i.e., classify them as ‘good’, ‘bad’ and ‘okay’.
- In order to achieve this, feature selection and classification using ML techniques - Naive Bayes & Support Vector Machine - will be applied.
- The final rating is done out of 5, showing ratings below 2.5 are considered negative (bad), 2.5 is considered neutral (okay) and above it is classified positive (good).

1.3 Hardware and Software Requirements

Hardware Requirements for Development :-

- Computer/Laptop

Software Requirements for Development :-

- Python 3.5+
- Python NLP Kit
- 10/8/7/Vista/2003/XP (64-bit)
- 2 GB RAM minimum.
- Links for extracting datasets from TMDB, IMDB, etc.
- Scrape Storm (Desktop Application)

Hardware Requirements for Deployment :-

- Computer/Laptop

Software Requirements for Deployment :-

- Internet Browser
- Web Host (Firebase)

Thus, in Chapter 1, a brief introduction has been given about the project by means of problem definition, which has helped identify the need for a solution, and scope of the project, in which the outline of the intended plan of implementation has been provided. Moreover, the reasons behind selecting the project have been cited and the chapter concludes by listing the software and hardware requirements that will be needed for the development and implementation of the project.

In the following chapter, the different research papers that have been referred for this project as part of the literature survey have been mentioned along with a brief analysis of each paper.

Chapter 2

Literature Survey

> *This chapter presents the literature survey conducted to gain an understanding of the project domain and available algorithms. The research papers that have been analysed have been mentioned along with their proposed methodologies.*

A number of research papers pertaining to the selected domain have been analysed to gain insight on how opinion mining and sentiment extraction works, what kind of algorithms can be used, what is the efficiency of the applied algorithms, what the system requires as input and what it delivers as output. The research papers provided different methodologies and algorithms that can be applied to the system in order to obtain the required output.

In the papers [1] and [2] we explored the different approaches that are possible for implementation of the movie review system. Paper [3] presents a comparison of the different algorithms that can be used for classification of user sentiments, and their efficiencies. Further paper deals with sentiment analysis on comments extracted from twitter.

[1] **A survey on sentiment classification of movie reviews**, by Jyotika Yadav we found that the sentiment analysis has been investigated mainly at three levels. The task at this document level is to classify whether the whole document is positive or negative. The second one is the sentence level which determines whether each sentence is positive or negative. The third level which is the Entity or Aspect level performs finer-grained analysis based on the phrases, opinion , etc.

A feature-based heuristic approach introduces an aspect-oriented scheme which consists of two approaches. In the first approach they assigned a sentiword label on each comment and then estimated the overall results. After this, they assigned a SentiWordNet based scheme with two different linguistic feature selections comprising of adjectives, adverbs and verbs and n-gram feature extraction. They computed results of four SentiWordNet based approaches for two movie reviews and two blog post datasets. Each synset in SWN comprises of sentiment scores that are positive and negative score along with an objectivity score. The summation of these three scores gives the relative strength of positivity, negativity and objectivity of each synset. These values have been obtained by using many semi-supervised ternary classifiers, with the capability of determining whether a word was positive, negative, or objective. One of the algorithm which can be implemented is SVM.

[2] **Sentiment analysis of Movie reviews : A study of feature selection and & classification algorithm** by Tirath Prasad Sahu and Sanjeev Ahuja explains in the following way .

1. Extracting Sentiment words :- All the review statement contains sentiment words which have a major contribution in determining the polarity of the review.
2. Sarcasm:- This is really a challenge, to find out the tone of the comment.
3. Parsing :- In this we find out what role does verb, adverb , adjective play in your comment.
4. Scaling;- When we have a large dataset with many comments it is really important for us to find out the overall result.

Methodology proposed in this paper :-

1. **Preprocessing** :- Preprocessing of document is preparation of dataset before applying any algorithm on it. This can be done by Porter stemming where we remove the commoner morphological endings. In the stopping technique we remove the most common words according to the stop word list, which usually contains prepositions, etc. In part of speech tagging the words are marked based on the relationship with the adjacent words in the sentence.
2. **Feature Extraction** :- In the next step after preprocessing we find a common observable pattern that may affect the polarity of the document. The classification of the the words is done in the following way positive sentimental words, negative sentimental words , positive sentimental Bi-grams , negative sentimental Bi-grams, positive sentimental Tri-grams , negative sentimental Tri-grams , positive sentiment coupled with adjective, negative sentiment coupled with adjective, positive sentiment words with repeated letters , negative sentiment words with repeated letters.
3. **Feature impact analysis and reduction** :- We find the impact of this by using information gain features and feature ranking algorithm.
4. **Algorithm** :- Calculations are done and then classification is done. We found that they used parameters such as precision , recall , F-measure, Accuracy and area under the curve. After evaluation we found that best accuracy was found in NB, KNN, Bagging , with an accuracy of 88.95% .

The process described above has been represented using the following diagram :-

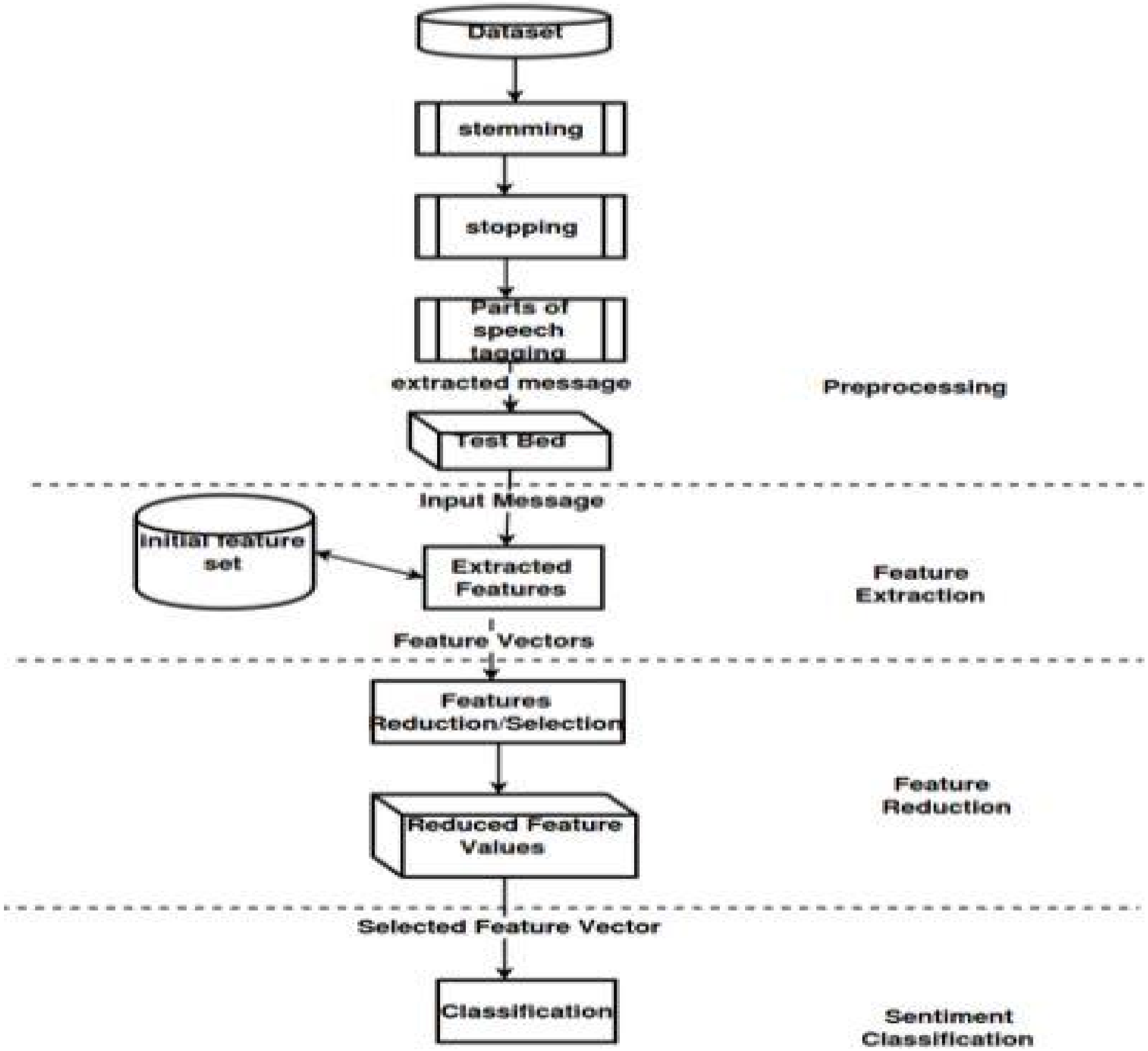


Figure 2.1 – Proposed Methodology for movie recommendation

[3] **Sentiment Analysis on movie reviews: A comparative study of machine learning algorithms and open source technology** by Mr. B Narendar we found two main methods for implementing sentimental analysis which is as follows :

1. **Bag of words Model (BoW)** :- In this method we have a dictionary which consists of words that add weight which is referred as sentiment in this context. The textual records consists of tokens which have a specific “weight” when mentioned in the context. The sentiment valuation is simply the result of the addition of the weights derived from all the textual records.
2. **Natural language processing** :- It gives us an insight in understanding the context , string of words and sentence structure. The BoW model requires massive amounts of machine learning concepts to be built in. Algorithms such as Support Vector Machine(SVM), Navïe Bayes Classifier, Maximum Entropy(MaxEnt) recognise patterns and add the weights. The BoW representation is obtained by using Navïe Bayes Classifier and the processing is done in Natural Language ToolKit(NLTK).

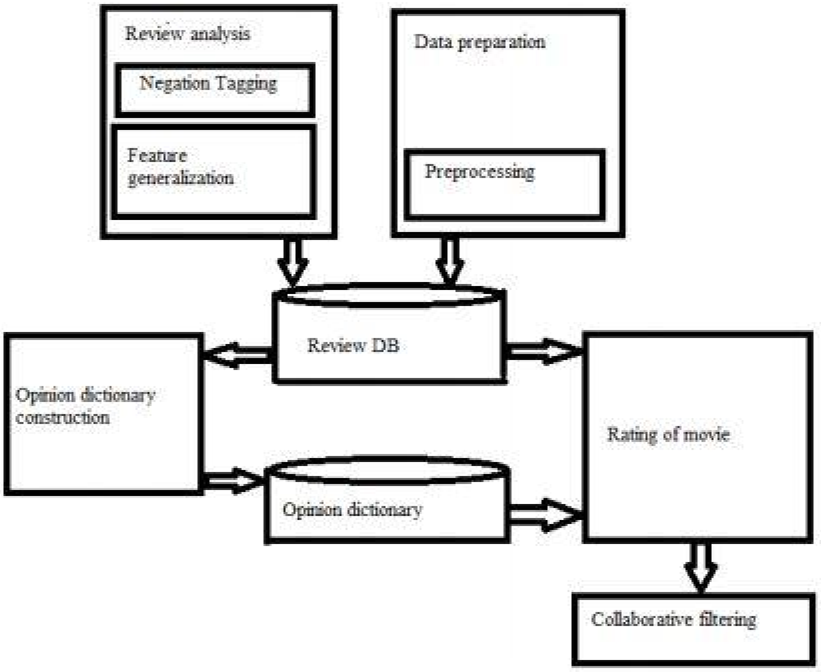


Figure 2.2 – Architecture for movie recommendation

In [6] **Movie Recommendation System Using Sentiment Analysis** by Amrutha S Nair , Sreelakshmi K, the system architecture is as shown in the above diagram. It mainly consists of three modules :

- Data Preparation
- Review Analysis
- Movie Recommendation

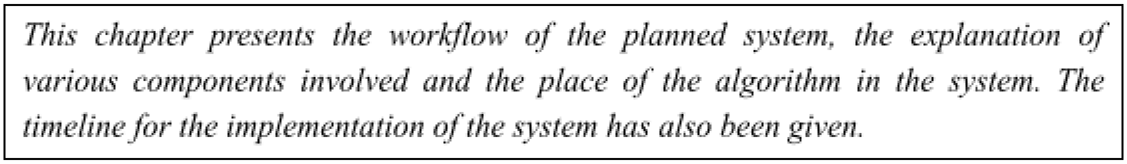
The figure gives the relationship of the components with each other and the entire flow of the process. This system architecture is very similar to the intended Movie Review System and has been very helpful in designing the proposed system model.

Thus, in Chapter 2, the various research papers which provide basic understanding of the topic at hand have been examined. An overview of the information about the processes that can be undertaken and the choice of methodologies available to perform sentiment analysis successfully that have been extracted from each paper have been described in brief.

In the following chapter, the proposed system architecture and process have been elaborated with diagrams and the project plan has been mentioned.

Chapter 3

Project Design



This chapter presents the workflow of the planned system, the explanation of various components involved and the place of the algorithm in the system. The timeline for the implementation of the system has also been given.

In order to assist with the implementation phase, the process of implementation has been charted out stage-by-stage and a proper design of the proposed system has been modelled. Components of the system have been described in the system architecture and working of the system has been shown using a use case diagram.

3.1 Proposed System Model

The systems follows the steps which are explained below:-

3.1.1 Data Collection & Preparation

The first step is to store the reviews that have been collected from authentic sources together form the review database that will be utilized in building the model. Now data needs to be prepared for feature selection. Pre-processing is applied to each review in the database. This involves removing tab spaces, newlines, dividing the sentences, removing numbers or digits, punctuation, removing stopwords (stopwords are the words that commonly used like *‘a’, ’an’, ‘the’*, that the machine learning algorithm can ignore), and converting all the letters to lowercase. Once this step is complete the output is stored in individual text files. The entire process is done using a program developed in Python.

3.1.2 Bag-of-Words Model

The ‘Bag-of-Words’ model is a simple representation of the vocabulary that is being used to build the model. This is built form the review database. This bag of words is also used by the classifier to identify the features of input given and then classify it into the respective category. The bag of words is based on natural language processing (NLP) and information retrieval (IR). It can be thought of like a dictionary that stores distinct words along with a mapping of the words to their count; count is the frequency of occurrence of that particular word in the dataset that is used to build the model. It is formed by using inbuilt collection - Counter(). This works on pre-processed words (tokens) and stores them along with their frequencies. The result is stored for later reference in a simple text file.

3.1.3 Classification

The vocabulary and review database together are used to train and test the classifier model. The partitioning of the all datasets for training and testing is done in the ratio of 60:40. Based on the words matched by the BoW model, the classifier acts as a binary classifier and will categorize the review/comment as positive or negative and assign a rating to each individual review. The average of the individual comments of a single movie is then calculated to arrive at a final rating for that particular movie.

1) **Naive Bayes Algorithm**

   Naive Bayes is a simple but effective classification algorithm. The Naive Bayes algorithm is a widely used algorithm for document classification. The basic idea is to estimate the probabilities of categories given a test document by using the joint probabilities of words and categories. The naive part of such a model is the assumption of word independence. The simplicity of this assumption makes the computation of Naive Bayes classifier far more efficient. For implementation, the linear arrays 'ytrain' and 'ytest' are formed to represent the actual classes as positive (1) and negative (0) in binary representation. The Naive Bayes classifier is trained and test using the inbuilt functions available in 'Scikit'. Actual and predicted outcomes are compared and the accuracy is determined using the ROC curve inbuilt metric.

2) Support Vector Machine (SVM)

Support vector machines (SVM), a discriminative classifier is considered the best text classification method. The support vector machine is a statistical classification method. Based on the structural risk minimization principle from the computational learning theory, SVM seeks a decision surface to separate the training data points into two classes and makes decisions based on the support vectors that are selected as the only effective elements in the training set. This is used for sentiment classification.

3) Collecting Reviews for a Movie

The movie for which a rating has to be obtained is decided. The reviews for this movie are searched on reliable online platforms like IMDB, Metacritic, Rotten Tomatoes, Tmdb. The reviews are scraped from these websites using an application called "Scrape Storm". These are extracted to a local database in the form of CSV files and then fed into the classifier to obtain a cumulative rating for the movie.

3.1.4 Final Rating of a Movie

The end result of any algorithm applied to the online reviews will be a rating. This rating is further used to arrive at a decision about whether the collective review for the movie has been positive, negative or neutral. This categorisation is done on the following basis:-

Movie Rating (r)	**Category**
0 <= r < 2.5	Bad Movie
r = 2.5	Okay Movie
2.5 < r <=5	Good Movie

Table 3.1 - Categorization of Movies

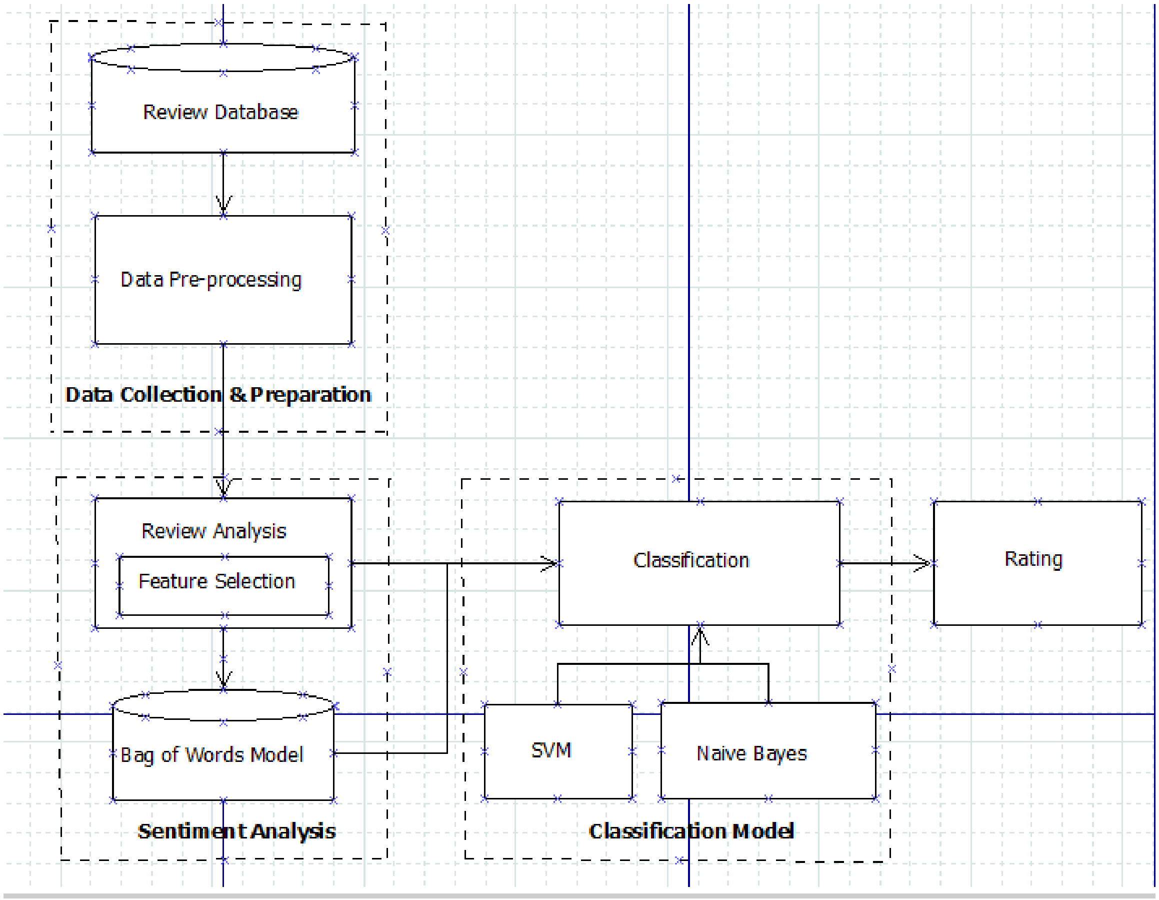
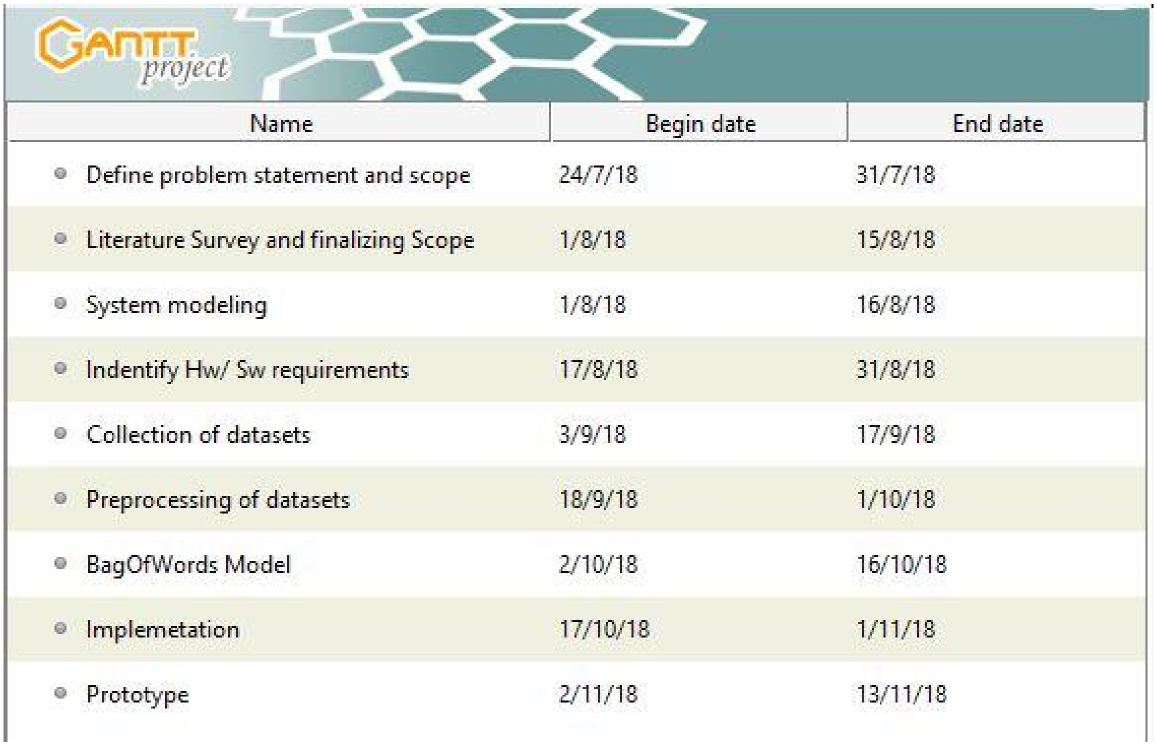


Figure 3.1 – Architecture for movie recommendation

3.2 Software Project Management Plan

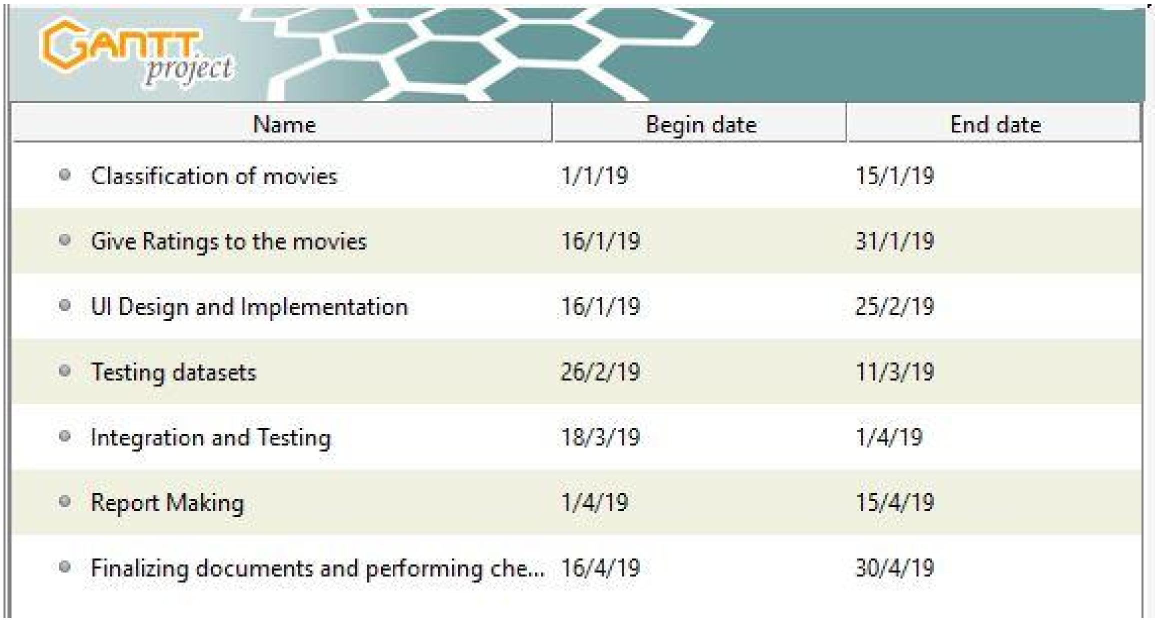
For the purpose of implementation, the entire project has been divided into a set of distinct activities. These have been scheduled for implementation through the 7th and 8th semester. The timeline is as follows :-

Plan for Semester VII and Semester VIII :



Name	Begin date	End date
Define problem statement and scope	24/7/18	31/7/18
Literature Survey and finalizing Scope	1/8/18	15/8/18
System modeling	1/8/18	16/8/18
Indentify Hw/ Sw requirements	17/8/18	31/8/18
Collection of datasets	3/9/18	17/9/18
Preprocessing of datasets	18/9/18	1/10/18
BagOfWords Model	2/10/18	16/10/18
Implemetation	17/10/18	1/11/18
Prototype	2/11/18	13/11/18

Figure 3.2 – Activity schedule for semester VII



Name	Begin date	End date
Classification of movies	1/1/19	15/1/19
Give Ratings to the movies	16/1/19	31/1/19
UI Design and Implementation	16/1/19	25/2/19
Testing datasets	26/2/19	11/3/19
Integration and Testing	18/3/19	1/4/19
Report Making	1/4/19	15/4/19
Finalizing documents and performing che...	16/4/19	30/4/19

Figure 3.3 – Activity schedule for semester VIII

Gantt chart for Semester VII :

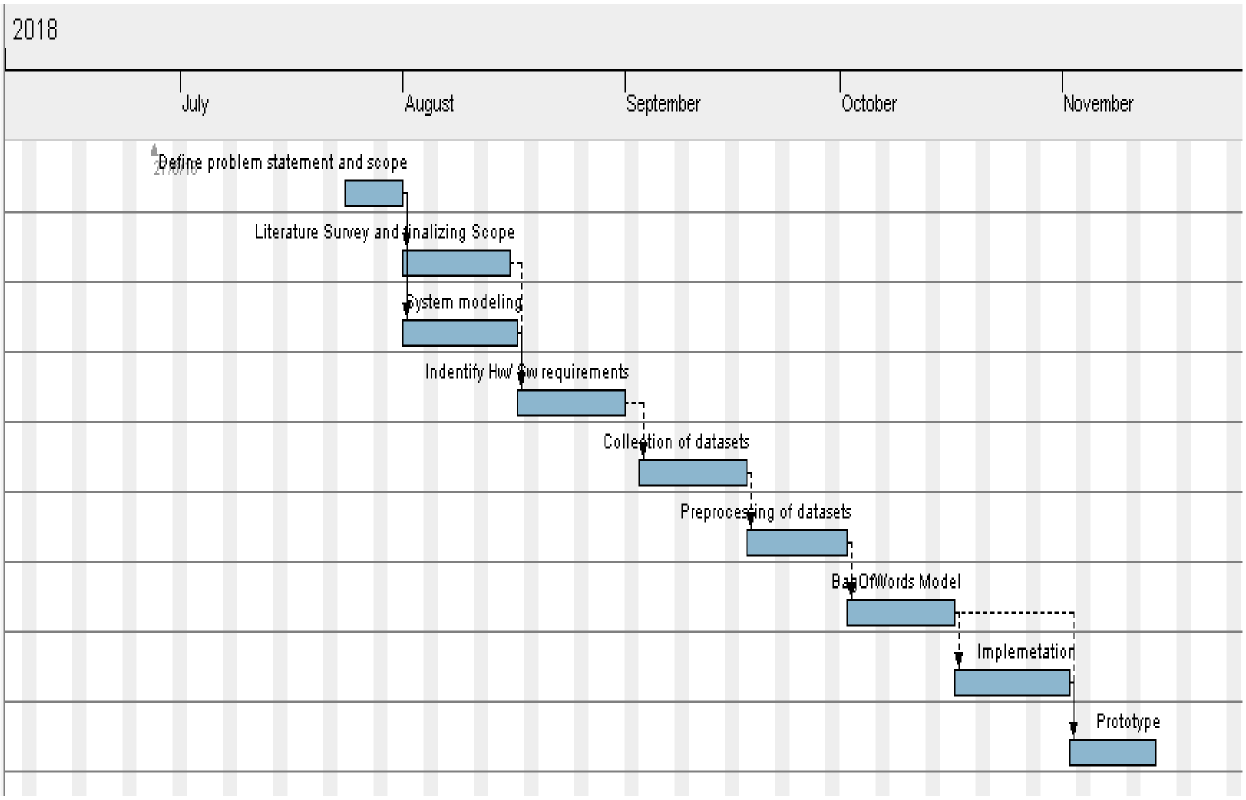


Figure 3.4 – Gantt chart for semester VII

Gantt chart for Semester VIII :

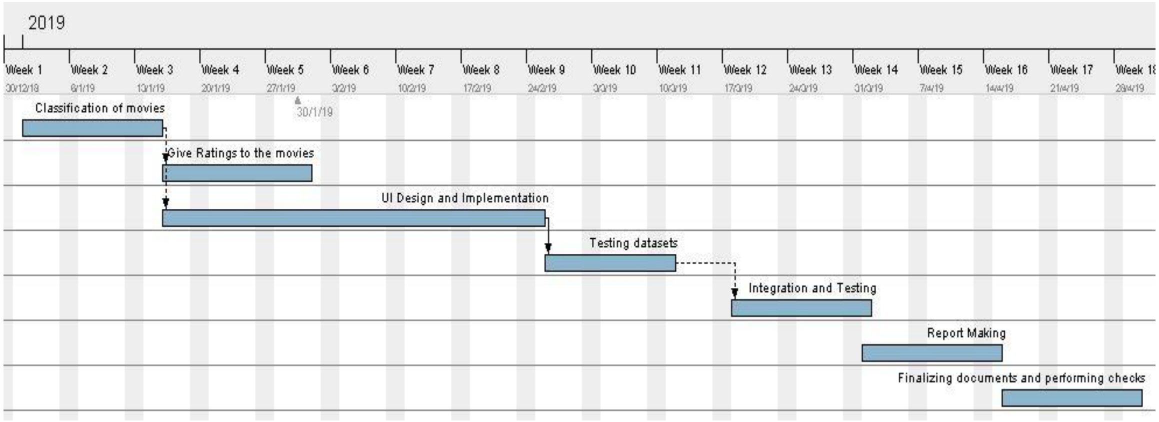


Figure 3.5 – Gantt chart for semester VIII

3.3 Software Requirement Specification Document

3.3.1 User Interface Requirements

The user interface will be a website. The website must satisfy basic design principles like being simple yet aesthetically pleasing, easy to use and easy to understand. A smooth and seamless navigation that requires minimum effort should be provided to user.

3.3.2 Hardware Interfaces

Hardware involved in implementation and deployment are the laptop/PC used for implementation or deployment respectively. The only requirement here is that the screen of the laptop/PC should be able to display the UI features properly.

3.3.3 Software Interfaces

The system can be used offline to test the input text. Implementation will require IDE for Python (Pycharm). For training and developer testing web browser will be required. API calls will be made through http communication. It needs a tool called scrape storm for collecting the comments for a particular movie.

3.3.4 Communication Interfaces

For training, developer testing and implementation, web browser will be required.

3.3.5 System Features

- **Scalability :**

  The model that has been built must be scalable, that is, applicable to datasets of large sizes as well. This is required because the reviews being taken from different sources will form datasets of variable size and the model should be able to handle them.
- **Performance Efficiency :**

  The system built should provide enough efficiency in processing. The efficiency will mainly be dependent on the classifier used for classification of the sentiment, time taken for training, testing and giving the output of the prediction.
- **Accuracy :**

  The implemented system should be able to identify the features of input text correctly and then classify them accurately.

3.4 Software Design Document

3.4.1 Use Case Diagram

The use case diagram shown below gives the simple form of interaction performed by 2 actors - the user and the admin.

- **The user** can access the website, search for the desired movie and check if its information is available on the website. If it is available then the view its ratings.

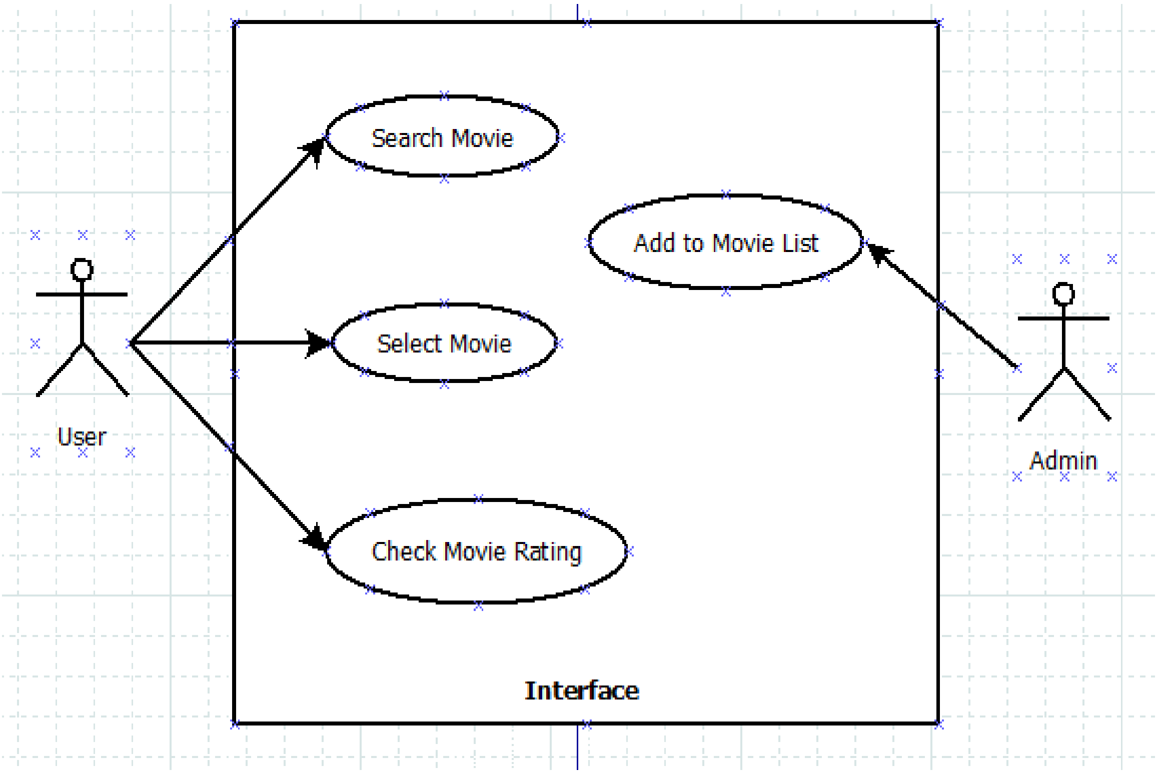


Figure 3.6 – Use Case Diagram

3.4.2 Data Flow Diagram

The Data Flow Diagram shown below gives the flow of the data through the proposed Movie Review System along with the databases that have been used or created during the process. The different stages are as follows:-

- External databases are used by the admin to obtain datasets of reviews. These datasets are already classified as positive and negative. They are cleaned and pre-processed and stores again.
- Then, the reviews are tokenized and used to build the Bag-of-Words model. This will contain the entire vocabulary and the corresponding frequencies.
- The BoW is then used to vectorize the reviews in the training and testing datasets. The vectorised training dataset is used to build the classifier model and the testing dataset is used for obtaining prediction accuracy.
- Finally, the unseen data which is obtained from external sources can be used for classification. The result is stored in a database along with the rating. This is displayed to the user.

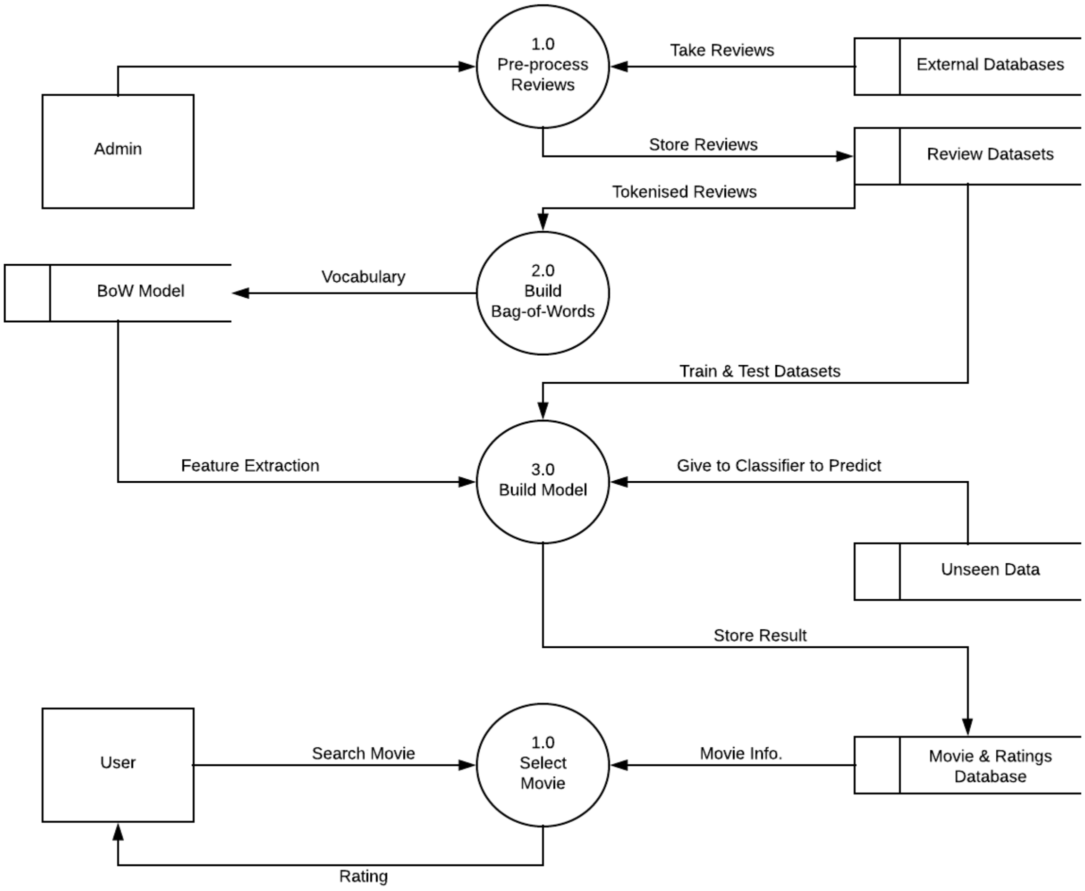


Figure 3.7 – Data Flow Diagram

Thus, in Chapter 3, the design of the actual system to be implemented has been put forth along with the plan of implementation has been mentioned with the scheduled activities. This has been followed by the documentation for the interface requirements of the system and the UML diagrams depicting the operation of the system.

In the following chapter, the details of the actual implementation of the project model according to the architecture have been described.

Chapter 4

Implementation and Experimentation

This chapter presents the implementation roadmap and the tasks performed during the execution of the planned system. It highlights the milestones that were achieved in the progress of the project and also gives the results and their analysis.

4.1 Proposed System Model Implementation

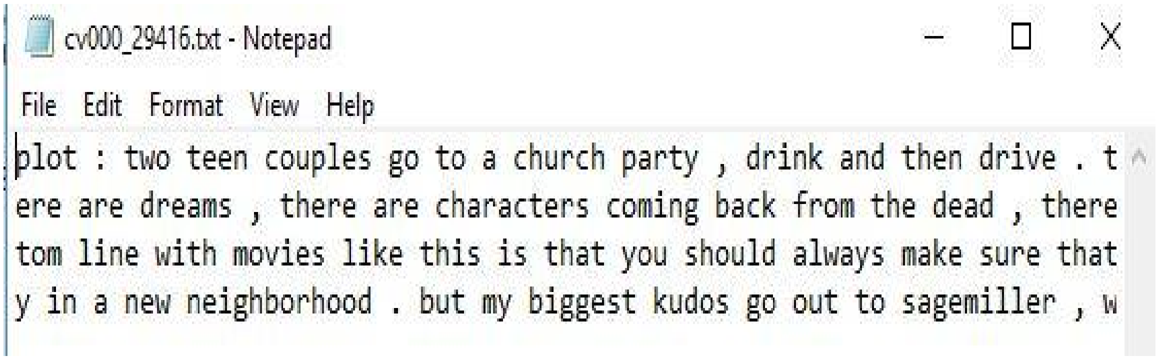
The softwares used for the implementation of the system are:

- Python 3.5+ (For building, testing the application)
- Python NLP Kit
- Windows 10 (64-bit)

4.1.1 Implementation of Algorithm

Training and testing of the algorithm requires pre-processing of the selected dataset. The dataset contains the comments in the form of text files.

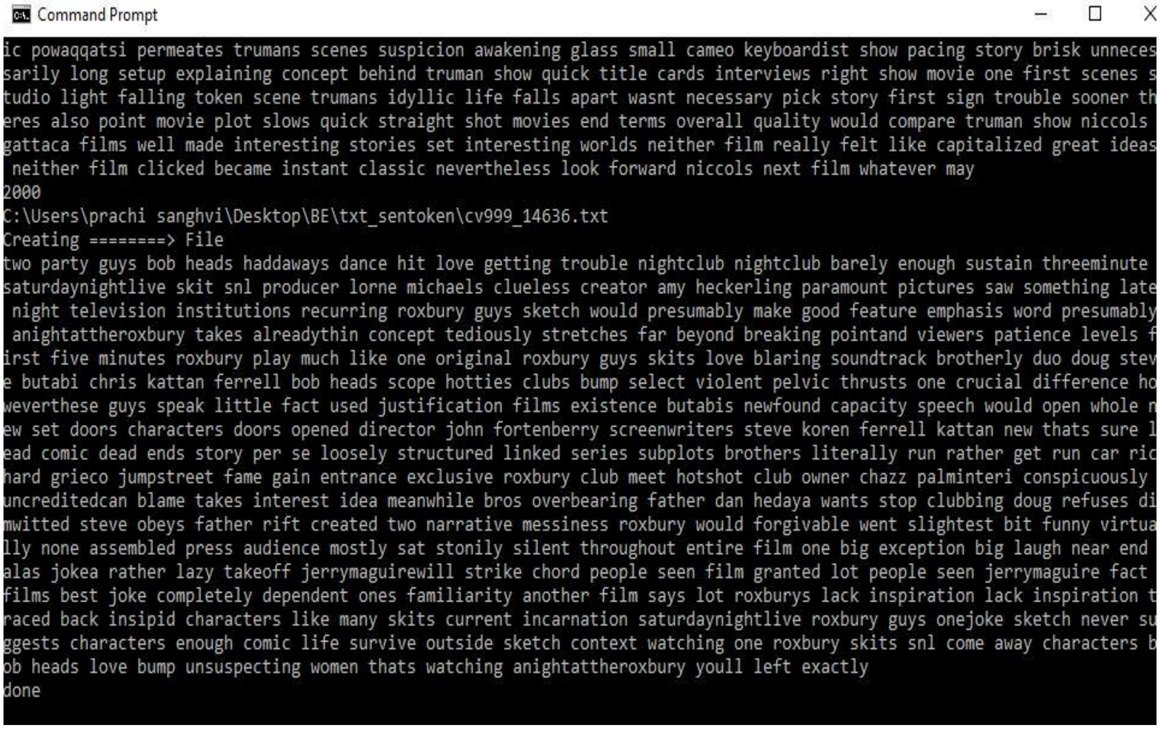
Pre-processing implementation :**-** Image of the comments before preprocessing



```
plot : two teen couples go to a church party , drink and then drive . t
ere are dreams , there are characters coming back from the dead , there
tom line with movies like this is that you should always make sure that
y in a new neighborhood . but my biggest kudos go out to sagemiller , w
```

Figure 4.1 – Input File

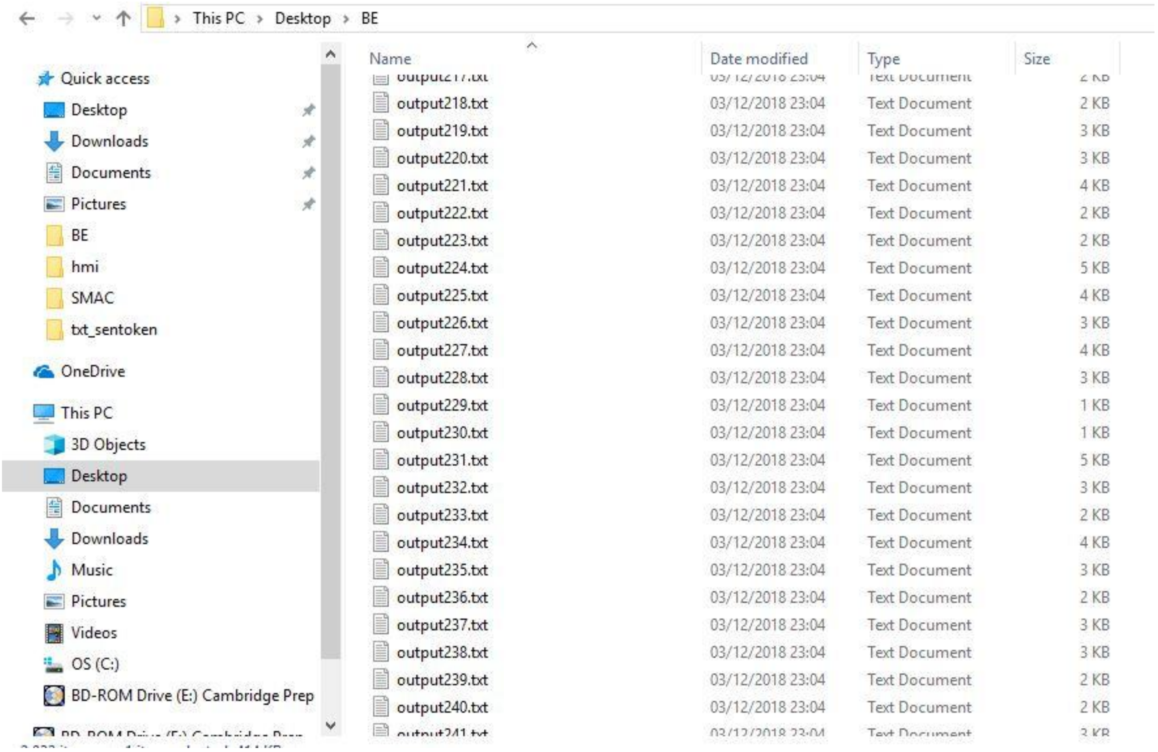
Implementation of Preprocessing code :-



```
ic powaqqatsi permeates trumans scenes suspicion awakening glass small cameo keyboardist show pacing story brisk unneces
sarily long setup explaining concept behind truman show quick title cards interviews right show movie one first scenes s
tudio light falling token scene trumans idyllic life falls apart wasnt necessary pick story first sign trouble sooner th
eres also point movie plot slows quick straight shot movies end terms overall quality would compare truman show niccols
gattaca films well made interesting stories set interesting worlds neither film really felt like capitalized great ideas
 neither film clicked became instant classic nevertheless look forward niccols next film whatever may
2000
C:\Users\prachi sanghvi\Desktop\BE\txt_sentoken\cv999_14636.txt
Creating ========> File
two party guys bob heads haddaways dance hit love getting trouble nightclub nightclub barely enough sustain threeminute
saturdaynightlive skit snl producer lorne michaels clueless creator amy heckerling paramount pictures saw something late
 night television institutions recurring roxbury guys sketch would presumably make good feature emphasis word presumably
 anightattheroxbury takes alreadythin concept tediously stretches far beyond breaking pointand viewers patience levels f
irst five minutes roxbury play much like one original roxbury guys skits love blaring soundtrack brotherly duo doug stev
e butabi chris kattan ferrell bob heads scope hotties clubs bump select violent pelvic thrusts one crucial difference ho
weverthese guys speak little fact used justification films existence butabis newfound capacity speech would open whole n
ew set doors characters doors opened director john fortenberry screenwriters steve koren ferrell kattan new thats sure l
ead comic dead ends story per se loosely structured linked series subplots brothers literally run rather get run car ric
hard grieco jumpstreet fame gain entrance exclusive roxbury club meet hotshot club owner chazz palminteri conspicuously
uncreditedcan blame takes interest idea meanwhile bros overbearing father dan hedaya wants stop clubbing doug refuses di
mwitted steve obeys father rift created two narrative messiness roxbury would forgivable went slightest bit funny virtua
lly none assembled press audience mostly sat stonily silent throughout entire film one big exception big laugh near end
alas jokea rather lazy takeoff jerrymaguirewill strike chord people seen film granted lot people seen jerrymaguire fact
films best joke completely dependent ones familiarity another film says lot roxburys lack inspiration lack inspiration t
raced back insipid characters like many skits current incarnation saturdaynightlive roxbury guys onejoke sketch never su
ggests characters enough comic life survive outside sketch context watching one roxbury skits snl come away characters b
ob heads love bump unsuspecting women thats watching anightattheroxbury youll left exactly
done
```

Figure 4.2 – Implementation on cmd

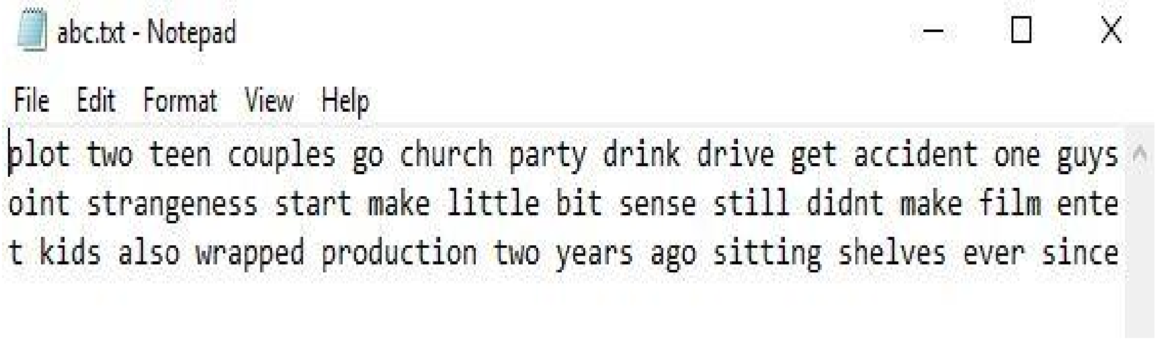
After applying the code, the output is as follows :-



Name	Date modified	Type	Size
output217.txt	03/12/2018 23:04	Text Document	2 KB
output218.txt	03/12/2018 23:04	Text Document	2 KB
output219.txt	03/12/2018 23:04	Text Document	3 KB
output220.txt	03/12/2018 23:04	Text Document	3 KB
output221.txt	03/12/2018 23:04	Text Document	4 KB
output222.txt	03/12/2018 23:04	Text Document	2 KB
output223.txt	03/12/2018 23:04	Text Document	2 KB
output224.txt	03/12/2018 23:04	Text Document	5 KB
output225.txt	03/12/2018 23:04	Text Document	4 KB
output226.txt	03/12/2018 23:04	Text Document	3 KB
output227.txt	03/12/2018 23:04	Text Document	4 KB
output228.txt	03/12/2018 23:04	Text Document	3 KB
output229.txt	03/12/2018 23:04	Text Document	1 KB
output230.txt	03/12/2018 23:04	Text Document	1 KB
output231.txt	03/12/2018 23:04	Text Document	5 KB
output232.txt	03/12/2018 23:04	Text Document	3 KB
output233.txt	03/12/2018 23:04	Text Document	2 KB
output234.txt	03/12/2018 23:04	Text Document	4 KB
output235.txt	03/12/2018 23:04	Text Document	3 KB
output236.txt	03/12/2018 23:04	Text Document	2 KB
output237.txt	03/12/2018 23:04	Text Document	3 KB
output238.txt	03/12/2018 23:04	Text Document	3 KB
output239.txt	03/12/2018 23:04	Text Document	2 KB
output240.txt	03/12/2018 23:04	Text Document	2 KB
output241.txt	03/12/2018 23:04	Text Document	3 KB

Figure 4.3 – Output Files generated

The files shown above have been created as a result of the pre-processing applied to the text files in the dataset.

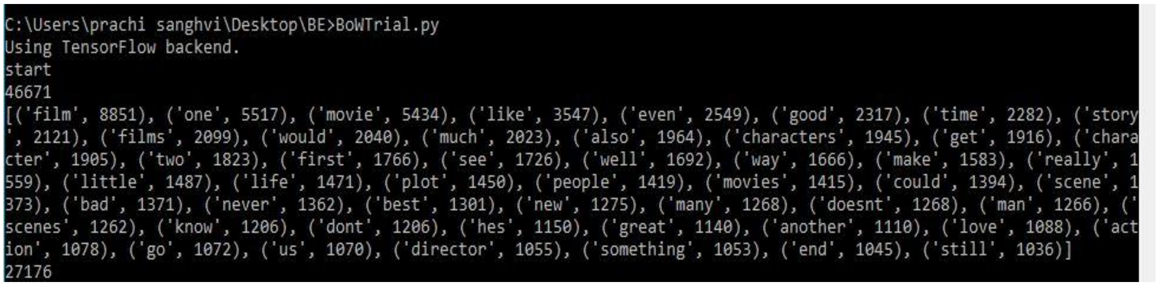


```
plot two teen couples go church party drink drive get accident one guys
oint strangeness start make little bit sense still didnt make film ente
t kids also wrapped production two years ago sitting shelves ever since
```

Figure 4.4 –Preprocessed Output Files

As it can be seen in the snapshot of the text file shown above, the tabs, the punctuations, numerals, stopwords, etc. have been removed and only the plaintext in lower case separated by whitespaces remains.

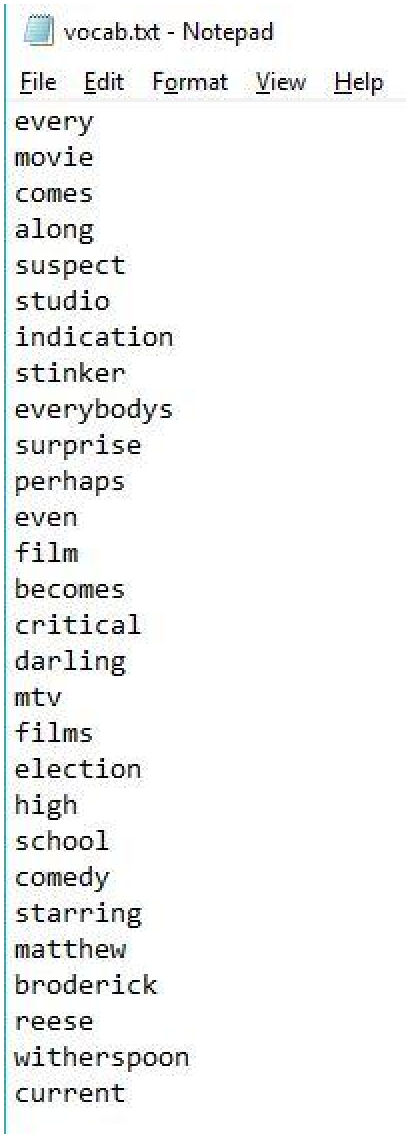
Implementation Bag Of Words Model :-



```
C:\Users\prachi sanghvi\Desktop\BE>BoWTrial.py
Using TensorFlow backend.
start
46671
[('film', 8851), ('one', 5517), ('movie', 5434), ('like', 3547), ('even', 2549), ('good', 2317), ('time', 2282), ('story
', 2121), ('films', 2099), ('would', 2040), ('much', 2023), ('also', 1964), ('characters', 1945), ('get', 1916), ('chara
cter', 1905), ('two', 1823), ('first', 1766), ('see', 1726), ('well', 1692), ('way', 1666), ('make', 1583), ('really', 1
559), ('little', 1487), ('life', 1471), ('plot', 1450), ('people', 1419), ('movies', 1415), ('could', 1394), ('scene', 1
373), ('bad', 1371), ('never', 1362), ('best', 1301), ('new', 1275), ('many', 1268), ('doesnt', 1268), ('man', 1266), ('
scenes', 1262), ('know', 1206), ('dont', 1206), ('hes', 1150), ('great', 1140), ('another', 1110), ('love', 1088), ('act
ion', 1078), ('go', 1072), ('us', 1070), ('director', 1055), ('something', 1053), ('end', 1045), ('still', 1036)]
27176
```

Figure 4.5– Bag Of Words model

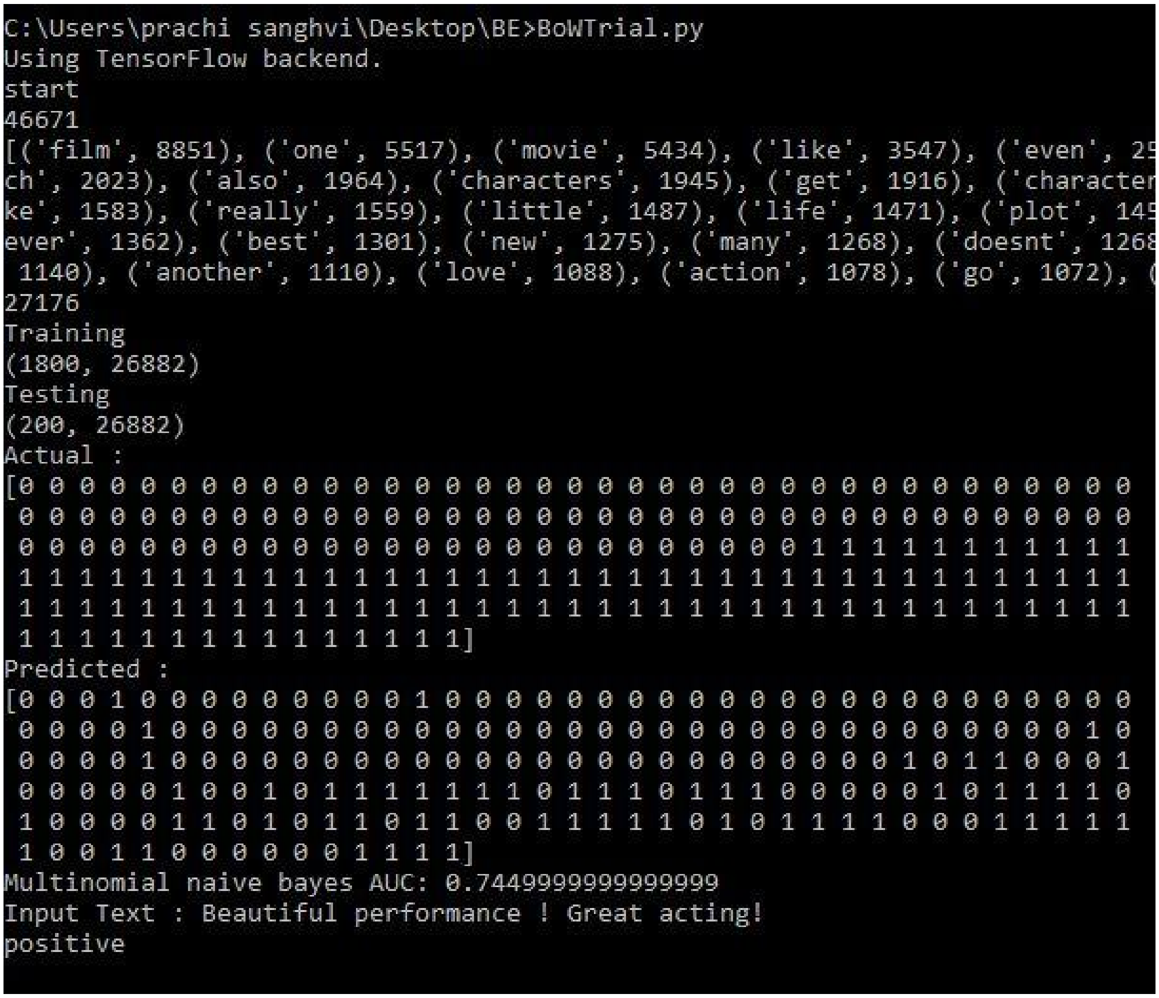
vocab.txt file is created which consists of all the words :-



```
every
movie
comes
along
suspect
studio
indication
stinker
everybodys
surprise
perhaps
even
film
becomes
critical
darling
mtv
films
election
high
school
comedy
starring
matthew
broderick
reese
witherspoon
current
```

Figure 4.6 – Words stored in file

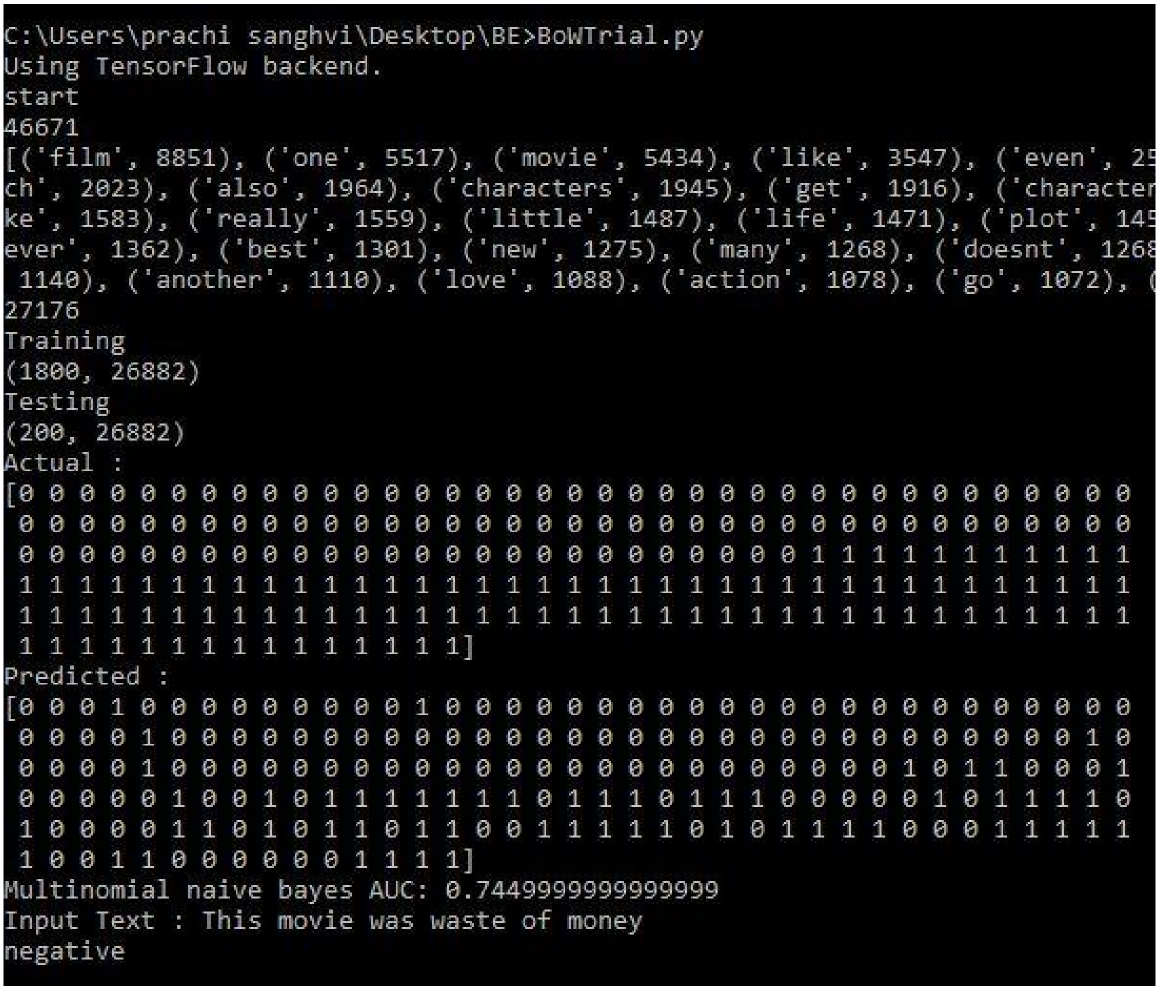
Naive Bayes Implementation :-



```
C:\Users\prachi sanghvi\Desktop\BE>BoWTrial.py
Using TensorFlow backend.
start
46671
[('film', 8851), ('one', 5517), ('movie', 5434), ('like', 3547), ('even', 25
ch', 2023), ('also', 1964), ('characters', 1945), ('get', 1916), ('character
ke', 1583), ('really', 1559), ('little', 1487), ('life', 1471), ('plot', 145
ever', 1362), ('best', 1301), ('new', 1275), ('many', 1268), ('doesnt', 1268
 1140), ('another', 1110), ('love', 1088), ('action', 1078), ('go', 1072), (
27176
Training
(1800, 26882)
Testing
(200, 26882)
Actual :
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
Predicted :
[0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 1
 0 0 0 0 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 0
 1 0 0 0 0 1 1 0 1 0 1 1 0 1 1 0 0 1 1 1 1 1 0 1 0 1 1 1 1 0 0 0 1 1 1 1 1
 1 0 0 1 1 0 0 0 0 0 0 1 1 1 1]
Multinomial naive bayes AUC: 0.7449999999999999
Input Text : Beautiful performance ! Great acting!
positive
```

Figure 4.7 – Output for positive feedback

A **positive feedback** was received for the above comment - "Beautiful performance! Great acting!" - which means the Naive Bayes algorithm identified the sentiment correctly.



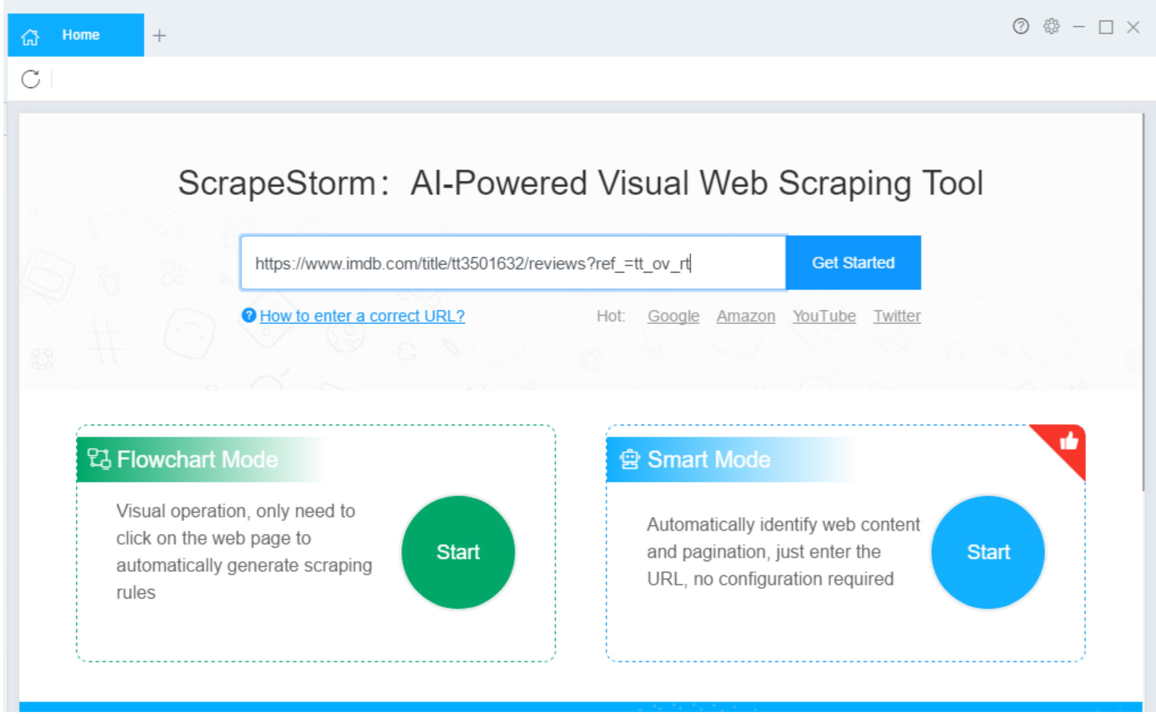
```
C:\Users\prachi sanghvi\Desktop\BE>BoWTrial.py
Using TensorFlow backend.
start
46671
[('film', 8851), ('one', 5517), ('movie', 5434), ('like', 3547), ('even', 25
ch', 2023), ('also', 1964), ('characters', 1945), ('get', 1916), ('character
ke', 1583), ('really', 1559), ('little', 1487), ('life', 1471), ('plot', 145
ever', 1362), ('best', 1301), ('new', 1275), ('many', 1268), ('doesnt', 1268
 1140), ('another', 1110), ('love', 1088), ('action', 1078), ('go', 1072), (
27176
Training
(1800, 26882)
Testing
(200, 26882)
Actual :
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
Predicted :
[0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 1
 0 0 0 0 0 1 0 0 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 0 0 0 0 0 1 0 1 1 1 1 0
 1 0 0 0 0 1 1 0 1 0 1 1 0 1 1 0 0 1 1 1 1 1 0 1 0 1 1 1 1 0 0 0 1 1 1 1 1
 1 0 0 1 1 0 0 0 0 0 0 1 1 1 1]
Multinomial naive bayes AUC: 0.7449999999999999
Input Text : This movie was waste of money
negative
```

Figure 4.8 – Output for negative feedback

A **negative feedback** was received for the sample text “This movie was a waste of money”. The algorithm again identified the sentiment correctly.

4.1.2 Implementation for a Single Movie

Collection of comments:- For obtaining the comments, we have used a scraping tool called 'Scrape Storm'. The first step is to search for a movie title on IMDB, TMDB, Rotten Tomatoes, Metacritic and go to the reviews section. The scraping tool requires the link of the page from which the comments are tone scraped. Copy the link of the user reviews and paste it in the text box provided. Click on the 'Get Started' button to start a new task from the movie that you have selected.



ScrapeStorm: AI-Powered Visual Web Scraping Tool

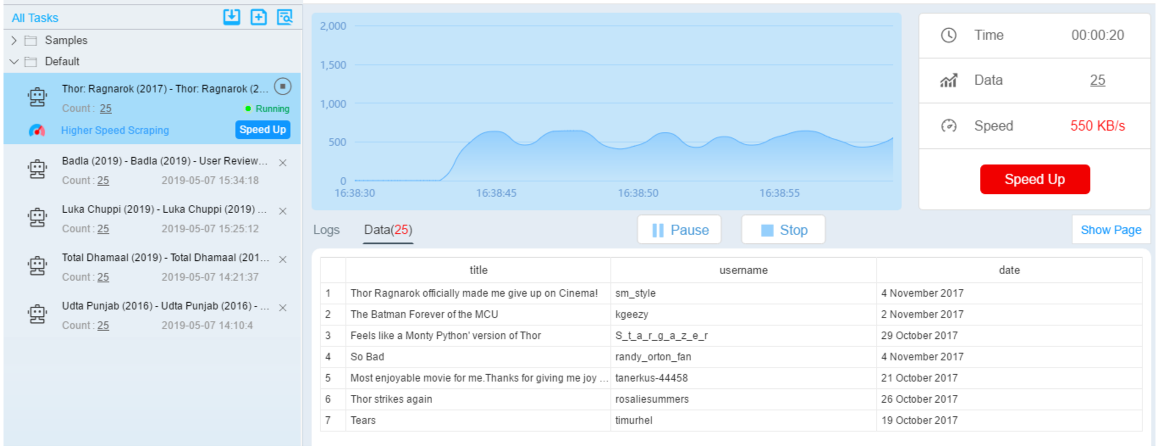
https://www.imdb.com/title/tt3501632/reviews?ref_=tt_ov_rt

Visual operation, only need to click on the web page to automatically generate scraping rules

Automatically identify web content and pagination, just enter the URL, no configuration required

Figure 4.9 – First page of scrape storm

Then the process begins :-



Thor: Ragnarok (2017) - Thor: Ragnarok (2... Count : 25 Running

Badla (2019) - Badla (2019) - User Review... Count : 25 2019-05-07 15:34:18

Luka Chuppi (2019) - Luka Chuppi (2019) ... Count : 25 2019-05-07 15:25:12

Total Dhamaal (2019) - Total Dhamaal (201... Count : 25 2019-05-07 14:21:37

Udta Punjab (2016) - Udta Punjab (2016) - ... Count : 25 2019-05-07 14:10:4

Time 00:00:20

Data 25

Speed 550 KB/s

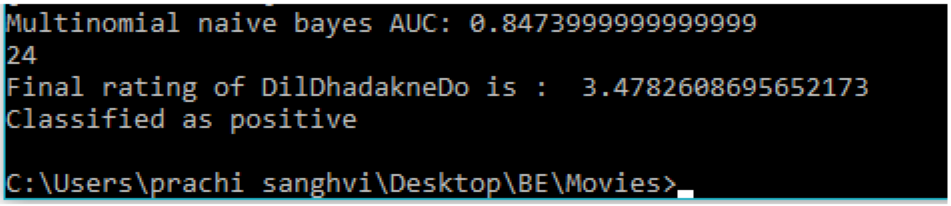
	title	username	date
1	Thor Ragnarok officially made me give up on Cinema!	sm_style	4 November 2017
2	The Batman Forever of the MCU	kgeezy	2 November 2017
3	Feels like a Monty Python' version of Thor	S_t_a_r_g_a_z_e_r	29 October 2017
4	So Bad	randy_orton_fan	4 November 2017
5	Most enjoyable movie for me.Thanks for giving me joy ...	tanerkus-44458	21 October 2017
6	Thor strikes again	rosaliesummers	26 October 2017
7	Tears	timurhel	19 October 2017

Figure 4.10 – Extracting comments for a movie

Comments can be exported from this application in the form of CSV files. After this, the program for a single movie with multiple comments from different websites is executed on the exported CSV file. The image below shows the final output in the form of a rating and classification for a given movie.

Using Naive Bayes algorithm:-

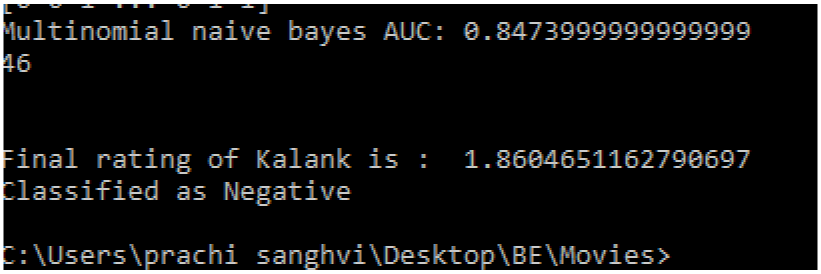
Output for a movie classified positive - Good



```
Multinomial naive bayes AUC: 0.8473999999999999
24
Final rating of DilDhadakneDo is :  3.4782608695652173
Classified as positive

C:\Users\prachi sanghvi\Desktop\BE\Movies>
```

Figure 4.11 – Output for positive feedback for a movie

Output for a movie classified negative - Bad



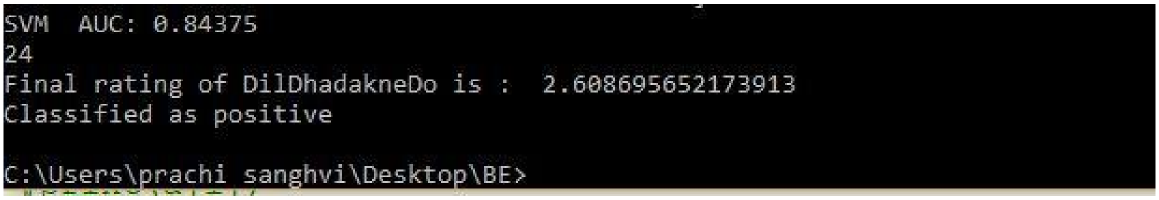
```
Multinomial naive bayes AUC: 0.8473999999999999
46


Final rating of Kalank is :  1.8604651162790697
Classified as Negative

C:\Users\prachi sanghvi\Desktop\BE\Movies>
```

Figure 4.12– Output for negative feedback for a movie

Support Vector Machine Implementation :-

The implementation is done on the dataset of 2000 reviews for training testing purpose which is divided in the ratio of 60-40. The process of scraping comments is the same as that of Naive Bayes algorithm. The image below shows the final output in SVM in the form of a rating and classification for a given movie.

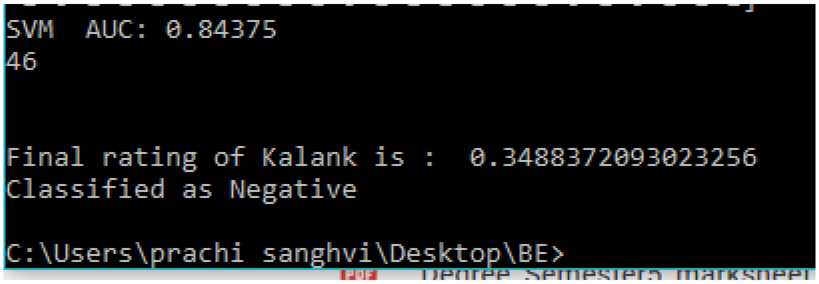
Output for a movie classified positive



```
SVM  AUC: 0.84375
24
Final rating of DilDhadakneDo is :  2.608695652173913
Classified as positive

C:\Users\prachi sanghvi\Desktop\BE>
```

Figure 4.13 – Output for positive feedback for a movie

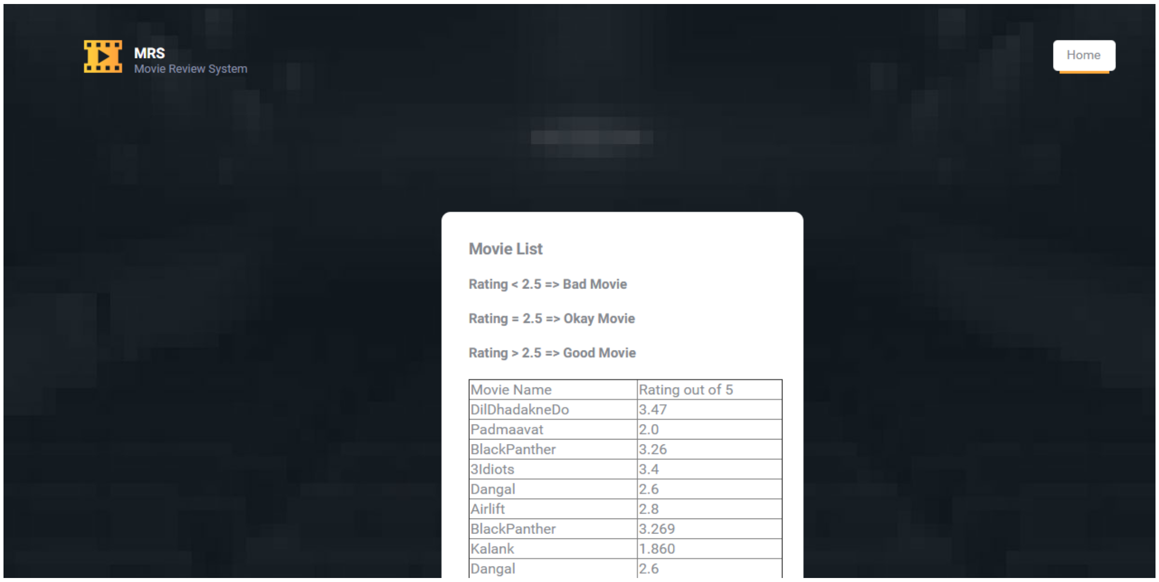
Output for a movie classified negative



```
SVM  AUC: 0.84375
46


Final rating of Kalank is :  0.3488372093023256
Classified as Negative

C:\Users\prachi sanghvi\Desktop\BE>
```

Figure 4.14 – Output for negative feedback for a movie

Result is displayed on the website



Movie List

Rating < 2.5 => Bad Movie

Rating = 2.5 => Okay Movie

Rating > 2.5 => Good Movie

Movie Name	Rating out of 5
DilDhadakneDo	3.47
Padmaavat	2.0
BlackPanther	3.26
3Idiots	3.4
Dangal	2.6
Airlift	2.8
BlackPanther	3.269
Kalank	1.860
Dangal	2.6

Figure 4.15 – Tabular display on the website

4.2 Software Testing

4.2.1 Black-Box Testing

Functional Specification:

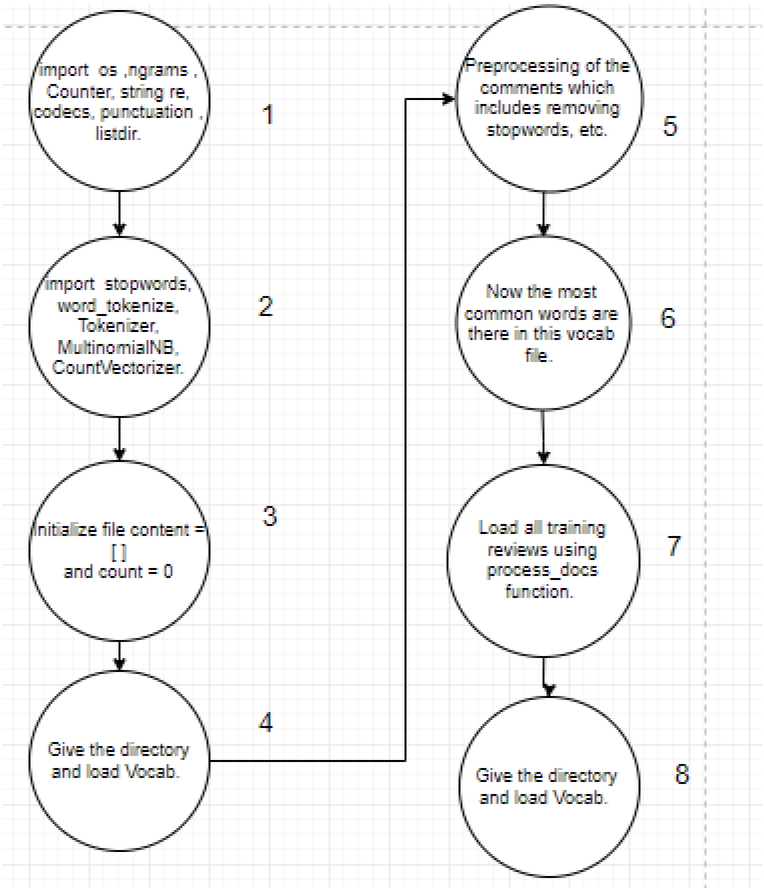
The selected modules are as follows :-

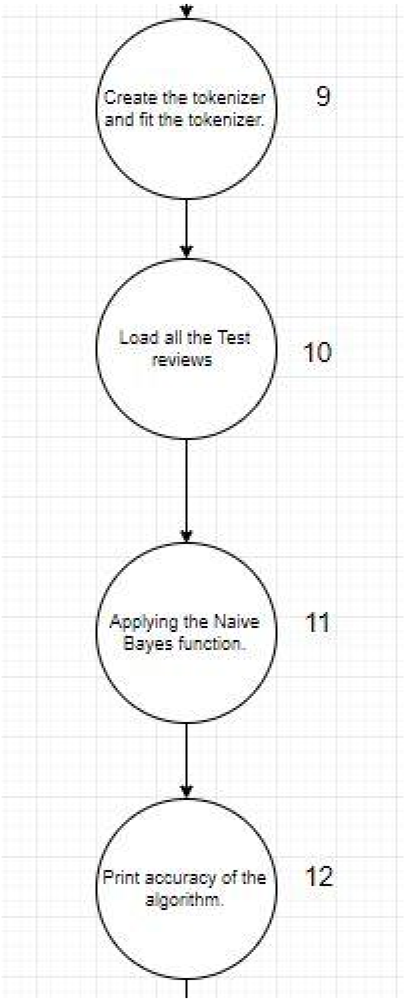
- Search - Search box is provided on the MRS homepage to enable user to search directly for movies which are then displayed in a drop down list

Module 1 : Search				
Condition	Valid movie in Search box	F	T	T
	Relevant results	-	F	T
Action	Expected Result	Error : Enter text	Error : Movie not found	Proper result
	Show Page	No results	Display related movies	Display list of movies

Table 4.1 Decision Table

4.2.2 White-Box Testing


import os ,ngrams , Counter, string re, codecs, punctuation , listdir.
1
import stopwords, word_tokenize, Tokenizer, MultinomialNB, CountVectorizer.
2
Initialize file content = [ ] and count = 0
3
Give the directory and load Vocab.
4
Preprocessing of the comments which includes removing stopwords, etc.
5
Now the most common words are there in this vocab file.
6
Load all training reviews using process_docs function.
7
Give the directory and load Vocab.
8


Create the tokenizer and fit the tokenizer.
9
Load all the Test reviews
10
Applying the Naive Bayes function.
11
Print accuracy of the algorithm.
12

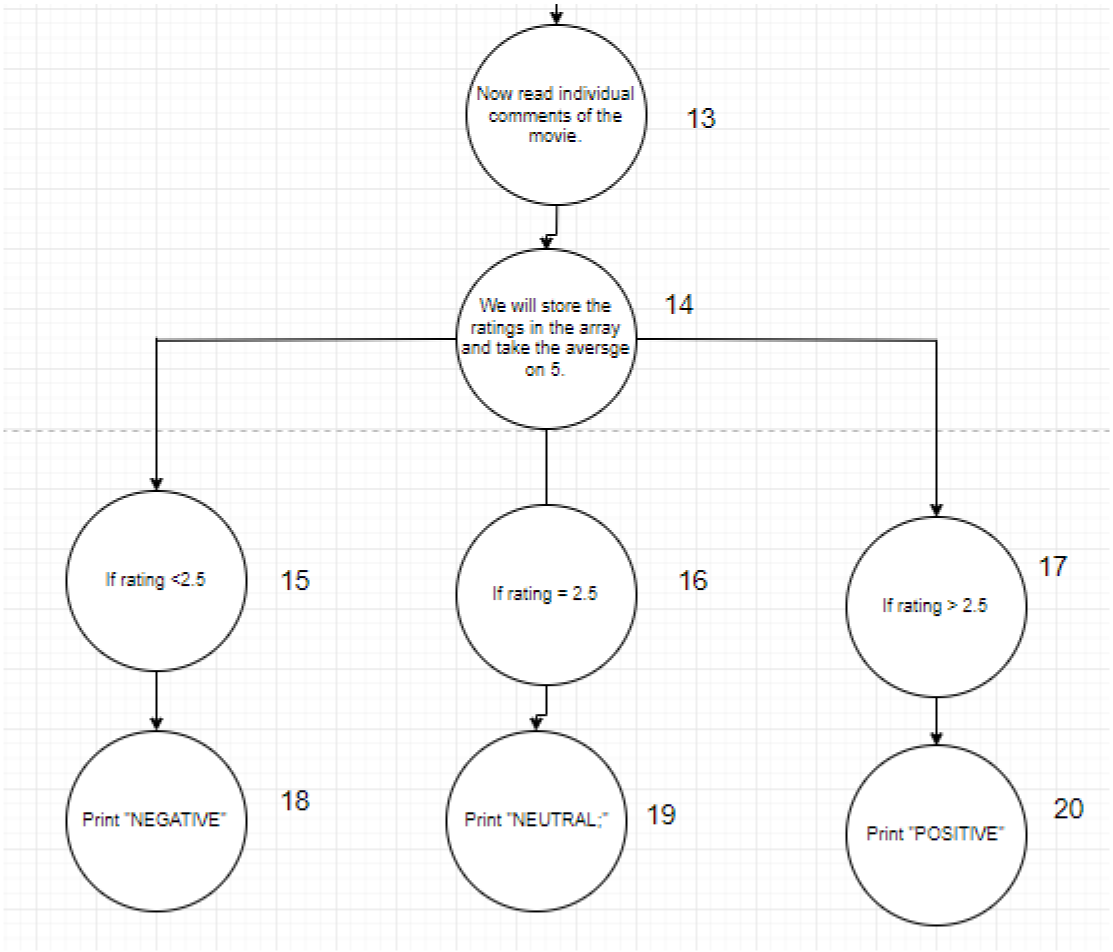


Figure 4.16– Control flow graph

Statement coverage :-

In this, an example is taken where every node of the path is covered. Say for example, the rating for the movie "**Captain America Civil War**" is to be found-

1. Import os , ngram , CountVectorizer, SVC, svm , stopwords, tokenizer, etc.
2. Initialize file content[], and count.

3. Give the Directory and load the vocab.
4. Preprocessing of the comments which includes removing stopwords, etc.
5. Now the most common words are in the vocab.
6. Load the training reviews
7. Create and fit tokenizer.
8. Load test reviews.
9. Apply Naive Bayes Function.
10. Read individual comments for the movie.
11. Store its rating in an array and find the average out of 5 and we get 2.33.
12. Depending on the rating it is classified on the following basis
    a. If rating<2.5 , then Negative.
    b. If rating=2.5 ,then Neutral.
    c. If rating>2.5, then Positive.

Path covered in statement coverage for **Captain America Civil War** is:-

1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-18.

80% of the statements are covered in any of the three cases.

Branch coverage:-

Identification of different branch :

In this, an example is taken where every node of the path is covered. Say for example the rating for the movie "**Captain America Civil War**" is to be found-

I. For Negative Movie.

1. Import os , ngram , CountVectorizer, MultinomialNB , stopwords, tokenizer, etc.
2. Initialize file content[], and count.

3. Give the Directory and load the vocab.
4. Preprocessing of the comments which includes removing stopwords, etc.
5. Now the most common words are in the vocab.
6. Load the training reviews
7. Create and fit tokenizer.
8. Load test reviews.
9. Apply Naive Bayes Function.
10. Read individual comments for the movie.
11. Store its rating in an array and find the average out of 5. The result is 2.33.
12. Depending on the rating which is <2.5 , it is Negative.

Path covered for **Captain America Civil War** is:-

1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-18.

II. For Neutral Movie.

1. Import os , ngram , CountVectorizer, MultinomialNB , stopwords, tokenizer, etc.
2. Initialize file content[], and count.
3. Give the Directory and load the vocab.
4. Preprocessing of the comments which includes removing stopwords, etc.
5. Now the most common words are in the vocab.
6. Load the training reviews
7. Create and fit tokenizer.
8. Load test reviews.
9. Apply Naive Bayes Function.
10. Read individual comments for the movie.

11. Store its rating in an array and find the average out of 5. The result is 2.5.

12. Depending on the rating which is =2.5 , it is Neutral.

Path covered for **Stanley and Iris** is:- 1-2-3-4-5-6-7-8-9-10-11-12-13-14-16-19.

III. For Positive Movie.

1. Import os , ngram , CountVectorizer, MultinomialNB , stopwords, tokenizer, etc.
2. Initialize file content[], and count.
3. Give the Directory and load the vocab.
4. Preprocessing of the comments which includes removing stopwords, etc.
5. Now the most common words are in the vocab.
6. Load the training reviews
7. Create and fit tokenizer.
8. Load test reviews.
9. Apply Naive Bayes Function.
10. Read individual comments for the movie.
11. Store its rating in an array and find the average out of 5. The result is 3.4.
12. Depending on the rating which is > 2.5 , it is Positive.

Path covered for **DilDhadakneDo** is:- 1-2-3-4-5-6-7-8-9-10-11-12-13-14-17-20.

Path selected after applying the Path selection criteria:

Path selected is Blue line for positive movie.

Path selected id green for neutral movie.

Path selected is red for negative movie.

4.3 Experimental Results & Analysis

Initially, the dataset of size 2000 was divided for training and testing in the ratio 80:20 in Naive Bayes classifier and obtained the accuracy of 74.99%. Later, after dividing training and testing dataset in the ratio of 60:40 accuracy was 78.59%. Hence, a standard ratio of 60:40 in training-testing is used for both the algorithms.

Another dataset containing 5,000 reviews was used for building both algorithms. The accuracies are shown below. Further, Naive Bayes algorithm was also implemented along with a review dataset of size 25,000 which gave an accuracy of 84.74%

The accuracy of both the algorithm on both the dataset is as follows:-

Dataset Size	Naive Bayes Accuracy (%)	SVM Accuracy (%)
2,000	78.59	84.37
5,000	79.19	80.85

Table 4.2 Accuracies of algorithm with different dataset

Movie Name	Naive Bayes Rating	SVM Rating	Actual Rating
Chakde! India	4.2	2.8	4.1
Kalank	1.8	0.48	1.85
Dil Dhadakne Do	3.4	2.6	3.4

Table 4.3 Predicted Ratings

Thus, in Chapter 4, a detailed description of the project implementation, which activities were performed and how they were carried out has been provided. It also gives the snapshots that show how the system model has been implemented, the accuracies of the different algorithms and ratings for movies based on their comments have been obtained. In the following chapter, the thesis has been concluded and future scope has been given.

Chapter 5

Conclusions and scope for further work

This chapter presents the conclusions drawn after researching and developing the prototype of the project. It also mentions the scope for further work.

5.1 Conclusion

Thus, this documentation gives the problem statement, the motivation and the scope of the project. It is followed by the literature survey that has been conducted by studying various research papers. These papers have discussed information related to different techniques and methods, like sentiment classification by Naive Bayes algorithm, SVM that can be undertaken to successfully achieve the desired system. Combining the knowledge gathered from these papers, the design of the proposed system was made consisting of 4 modules : data collection and preparation, sentiment analysis, classification and output (rating). This system uses the ‘Bag of Words’ model combined with the Naive Bayes classifier and SVM for sentiment analysis.

The implementation of algorithms done with training and testing datasets of varying size gave the results with different accuracies for the algorithms. These algorithms were then used to calculate ratings for movies and categorise them. Based on the accuracies of these ratings, the Naive Bayes model has been integrated along with the Firebase database and website. The final output and list of ratings for several movies is displayed on the website.

5.2 Scope for further work

The classifiers that have been used in this project are binary classifiers. The future scope in sentiment analysis would be to train algorithms that can classify into more that 2 sentiments. For example, each comment can be labelled as one among positive, negative or neutral. Extending this concept further, just like each movie is being given a rating out of 5, each comment can also be given rating 1, 2, 3, 4, 5. From this, the final movie rating can be in categorised into 5 groups : 0-1 as very bad, 1-2 as bad, 2-3 as okay, 3-4 as good and 4-5 as very good. This would increase the granularity of the system.

Bibliography

1. Tirath Prasad Sahu, Sanjeev Ahuja, “Sentiment analysis of movie reviews: A study on feature selection & classification algorithms.” IEEE, 2016

2. Mr. B. Narendra, Mr. K. Uday Sai, etc. “Sentiment Analysis on Movie Reviews: A Comparative Study of Machine Learning Algorithms and Open Source Technologies.” I.J. Intelligent Systems and Applications, 2016

3. Jyotika Yadav, “A Survey on Sentiment Classification of Movie Reviews.” IJEDR, 2014

4. G. Hemantha Kumar, “ Movie Recommendation based on Users' Tweets ” Volume 141- No.14 , May 2016.

5. G. Hemantha Kumar, “Movie Recommendation based on Users' Tweets ” International Journal of Computer Applications, 2016

6. Amrutha S Nair , Sreelakshmi K , “ Movie Recommendation System Using Sentiment Analysis ” International Journal for Trends in Engineering & Technology, 2017

7. Palak Baid, Apoorva Gupta, Neelam Chaplot, “Sentiment Analysis of Movie Reviews Using Machine Learning Techniques” International Journal of Computer Applications, 2017

8. G. vinodhini, RM Chandrasekaran, “Sentiment Analysis and Opinion Mining : A Survey” International Journal of Advanced Research in Computer Science and Software Engineering, 2012

9. S. Sharma, D. Singh, “Study of Sentiment Classification Techniques” International Journal of Computer Sciences and Engineering, 2018

10. Vibhor Singh, Priyansh Saxena, Siddharth Singh, S. Rajendran, “Opinion Mining and Analysis of Movie Reviews” Indian Journal of Science and Technology, 2017

11. J Sai Teja, G Kiran Sai, M Druva Kumar, R.Manikandan, “Sentiment Analysis of Movie Reviews Using Machine Learning Algorithms - A Survey” International Journal of Pure and Applied Mathematics, 2018