Analyzing Users' Sentiment towards Video Games Based on Reviews from Microblog

Thesis submitted in partial fulfilment of the requirement for the degree of

Bachelor of Computer Science and Engineering

Under the Supervision of

Dr. Jia Uddin

By

Abhijeet Roy (13301130)

Mobtasim Hasan Khan (13201039)

Shandro Chakraborty (14101107)

School of Engineering and Computer Science

25th March 2018

BRAC University, Dhaka, Bangladesh

Declaration

We declare that this thesis is based on the results we obtained from our work. Due acknowledgement has been made in the text to all other material used. This thesis, neither in whole nor in part, has been previously submitted by anyone to any other university or institute for the award of any degree.

Signature of Supervisor

Dr. Jia Uddin

Signature of the Authors

Abhijeet Roy (13301130)

Mobtasim Hasan Khan (13201039)

Shandro Chakraborty (14101107)

Acknowledgement

First and foremost, we would like to thank Almighty for enabling us to initiate the research, to put our best efforts and successfully conclude it.

Secondly, we would like to thank our supervisor Dr. Jia Uddin for his support, guidance and contribution in conducting the research and preparation of the report. He motivated us to do this research thoroughly and always was there to help us with any aid he could offer. His great supervision, inspiration and effective guidance allowed us to complete this paper. We are truly grateful to him.

We revere the patronage and moral support extended with love, by our parents as well as our friends. They helped us with their direct or indirect suggestions which aided in achieving our goal. We would also like to acknowledge the assistance we received from numerous resources over the internet especially from fellow researchers' work.

Last but not the least, we thank BRAC University for providing us the opportunity of conducting this research and for giving us the chance to complete our Bachelor degree.

Contents

List of Figure

List of Tables

List of Abbreviations

API – Application programming interface

CSV – Comma separated value

IDE – Integrated development environment

JSON – Java script object notation

MatPlotLib – Mathematical plotting library

NLP – Natural Language Processing

NLTK - Natural Language Tool Kit

Abstract

This project proposes a new model of sentiment analysis for video game's reviews. In these days people tend to check reviews and ratings of video games before spending money and time for a game. In the proposed model, ratings for video game will be generated by doing sentiment analysis on public opinion. As Twitter is one of the most popular micro-blogging sites, for public opinion we collected data from Twitter. Before fitting the algorithms we pre-processed the gathered data to a supervised form. In the model Naïve Bayes, Support Vector Machine, Logistic Regression and Stochastic Gradient Descent algorithm were used for performance comparison. They were trained on a training set and to validate the performance the algorithms were tested several times on a test set to get better accuracy. After that a new classifier was used which acted as a voting classifier for the algorithms. This classifier was used for sentiment analysis on the data to get polarity. To validate the model, we generated rating from calculating polarity for each attribute which contains gameplay, graphics, sound, multiplayer and plotted in a graph where results are shown.

CHAPTER 1

Introduction

Nowadays people like to check reviews before spending money and time on a video game. Games are expensive and time consuming. Majority of the demographic of video game users use Metacritic or similar score aggregator sites for buying decision. Problem with those websites is that they tend to use ratings from sites where professional game reviewers review games. Which means those ratings never represent the opinion of general public. Therefore in many cases public's opinion of a game differs from the professional reviews. Thus the decision making process becomes more challenging. In this research we'll build a model that generates video game ratings based on public opinion on different attributes of a game, which will give users of the model accurate reflection of the public's opinion about any particular video game.

1.1 Motivation

There are numbers of dedicated websites and YouTube channels that review video games. Professional reviewers are appointed for reviewing games for the websites and video channels. Normally nearly all of them reviews games when a particular game is released initially. This practice of reviewing games has some major drawbacks. Firstly, a video game nowadays after releasing, most often gets patches and updates to make the experience better or in some cases worse. But reviewing a game once after releasing never reflects the changes that developers made to the game. Therefore the rating on a reviewing site never represents the current state of a game. Secondly, websites that reviews games have dedicated professional reviewers for each category of games for example those who specializes on reviewing racing games tend to review racing game only. Which is a drawback as people would not get the best idea about a particular game because of this. Then most importantly a professional review never reflects any controversy or major changes to any game. One other reason ratings from a professional reviewing site is not a reflector of the state of a game because most often reviewers rush to finish reviewing as they normally have to review a certain number of games before a deadline which makes it hard to take their review as a comprehensive guide for buying decision as they often tend to look over something that general people find while playing the games later. Many games that had bad reviews later turned into a

solid game after number of patches, on the other hand sometimes a well-reviewed game often turns unplayable because of studio business decisions, for example: implementing micro-transition. Therefore public opinion is a better reflector of a video game than a professional reviewing website's rating. This is why taking buying advice for video game from people's opinion is a better option than relying on professional review site.

1.2 Contribution Summary

For public opinion mining, numbers of research have been done by various researchers on sentiment analysis. Very few research on sentiment analysis is done on the field of video game reviews. Among those few none of the research focused on generating rating based on public opinion of video game which helps the decision making process of a potential buyer of a certain game. The main purpose of this research is that this model will allow users to analyze video games and each attributes of a certain video game based on public opinion. The rating of each video game and its attributes will be generated from public opinion which is a better reflector of current state of a game. In this research, we propose a model that will provide ratings based on public opinion and therefore users of this model can rely on the generated ratings for buying decision.

1.3 Thesis Outline

The rest of the thesis is organized as follows:

- Chapter 2 includes the necessary background information regarding the proposed approach and the algorithms of sentiment analysis
- Chapter 3 presents the proposed model of the research which includes discussion about dataset, Tweet data, data pre-processing, algorithm implementation, accuracy, sentiment analysis on gathered data.
- Chapter 4 demonstrates the experimental results with graphical representation.
- Chapter 5 concludes the thesis and states the future research directions.

CHAPTER 2

Background Analysis

2.1 Literature Review

Sentiment analysis is growing area of research in the field of natural language processing (NLP). In the past few years many research on sentiment analysis have been done. A fair amount of research focused on how sentiments are expressed in various categories such as online reviews, news article etc. along with how sentiments are expressed given the informal language of social media and micro-blogs [1]. From this research, we took the idea about extracting user opinion from review and classifying them, along with that we also observed their results which helps us decide which algorithm we can use in our model. Research on Twitter data has found that Twitter data has an impressive predictive power that ranges from stock market to movie performance [2]. In this research, sentiment analysis was performed on the data extracted from Twitter, from analyzing this research it became clear that to get user opinion from microblog Twitter will be a ideal platform, thus Twitter was used as the platform for user opinion extraction in our model. Minqing Hu and Bing Liu in 2004 proposed a model they aimed to mine and summarize online opinions in reviews, blogs and forums [3]. Their work in sentiment analysis on online reviews was one of the first research that was done in the respective field and this research set the motion for future research on sentiment analysis on online opinion summarizing. For opinion summarization the focused on quantitative aspect of the opinions. E. Junque de Fortuny, T De Smedt, D Martens in 2010 proposed a model to scrape relevant text from websites and perform sentiment analysis on the scraped data. Their subject of sentiment analysis was Belgian elections. The corpus used for processing were gathered from online versions of all Flemish newspaper. A web crawler was used, each adjective was manually given a polarity score for sentiment analysis [4]. In research [4] they used a scrapper for extracting online data and then assigned weight to each words, by analyzing this research we implemented the idea of using scrapper for extracting Tweets and assigning weights to each words for sentiment analysis. Efthymios Kouloumpis, Theresa Wilson , Johanna Moore in 2011 proposed a model that used the features that capture information about the informal

language used in micro blogging to analyze the features for detecting the sentiment of Twitter messages. This research also evaluated the usefulness of existing lexical resources [5]. In research [5] they performed sentiment analysis on informal language that user use in Twitter and analyzed the featuers of informal language used in Twitter, also in the research they used existing lexical resouces for sentiment analysis which turned out to be less useful for sentiment analysis on Tweets, therefore for our research based on the idea from analyzing their research, we opted to go for creating our own feature set. Ike Pertiwi Windasari, Fajar Nurul Uzzi, Kodrat Iman Satoto in 2017 did a research on sentiment analysis on Twitter posts where they performed data pre-processing, feature extraction but only used one algorithm for sentiment analysis, the stated that they want to implement multiple algorithms in the future and compare accuracy among them to find out which one performs best [6]. By analyzing their research, it was found that there are room for improvement when it comes to using more than one algorithm, therefore in our research we decided to use multiple algortihms and compare between them for accuracy. K. Kaviya, C. Roshini, V. Vaidhehi and J. Dhalia Sweetlin in 2017 proposed a model for assigning rating to restaurants from user review [7]. In their proposed model they rated the restaurants based on the user reviews on a scale of 1 to 10, by analyzing their model we found room for improvement. Instead of just assigning a rating on the whole restaurant they can improve their model by assigning rating to each attributes of the restaurant. In our proposed model, we will generate rating for video game based on the user opinion where each attribute of a game, for example, gameplay, audio, graphics, multiplayer, will be rated based on the user opinion. From above research, that we discussed, we found a number of opportunities to propose a improved model for sentiment analysis.

2.2 Supervised Learning

Supervised machine learning is where the training data is labeled with desired output and using an algorithm it can learn the mapping function from the input to output. The main goal of the supervised machine learning is to approximate the mapping function well enough to predict output variable given a relatively trained input data. A supervised learning algorithm analyzes the training data and based on the training it gives a output that are then tested against a test set to see the accuracy of the algorithm on that particular training data. In supervised learning the algorithm

iteratively makes prediction on the training data, the learning is usually stopped when acceptable level of accuracy is achieved.

To solve a problem using supervised machine learning some steps need to be performed. They are as follows:

1. Defining the type of training set. The type of training set must be selected before starting anything else because the algorithm will be trained based on this training set.
2. Gathering the training set. A well represented training set needs to be gathered that has input object as well as corresponding output. This will be used for mapping the input with the output using a supervised training algorithm.
3. Define supervised machine learning algorithm. What kind of learning algorithm will be used will determine the accuracy as some algorithm works better on a given data set than others. As per No Free Lunch Theorem there cannot be one algorithm that can work well on all the supervised problems. Therefore choosing the learning algorithm is crucial part the process.
4. Train the data set using learning algorithm. Learning algorithm may need to be run multiple times on the dataset. Each time to get close to acceptable level, user may need to optimize the training set to get better result each time.
5. Evaluation of the accuracy. After optimizing the data set and training the algorithm, the algorithm will be run on a test set to test the accuracy.

2.3 Sentiment Analysis

Sentiment analysis at its basic is an opinion mining process that determines whether a piece of writing is positive, negative or neutral. The reason researchers use sentiment analysis is to determine how people feel about a particular topic [8]. For example in our research we determine how people on Twitter feel about a certain number of video games. Now sentiment analysis on the given twitter data can answer what is the opinion of the people of twitter. Sentiment analysis is useful to monitor social media and micro-blog as it gives us an overview of the wider public opinion for a certain topic. In this research, we extract insights from micro-blog using sentiment analysis and determine the public opinion for a number of video games.

There are a few websites that compiles reviews and ratings for video games in the internet. Those reviews are mostly written by professional journalists and reviewers of the dedicated site like New York Times, Huffington Post. Even though there are websites that compiles the rating from the public, the problem with that rating is that average rating on a certain type of video game can misrepresent the quality of the video game often also two very different kind of video games can have similar rating but in context their quality may vary drastically. There's also the problem of newly released video games. On average any person can read around 300 words per minute. The time to completely read a video game will vary with the length of the video game itself. Therefore a longer video game will take more time, which means the public rating and review of that particular video game to appear online will take longer time. This is exactly where sentiment analysis can help tremendously. While reading a video game anyone can share their opinion on the video game on the social media or a micro-blog. From there it can be determined their opinion on the video game. Therefore a prediction can be done using sentiment analysis to find out the rating of that particular video game. On the other hand the video games that are already published for a while already has multiple reviews and ratings. The reason we prioritize using sentiment analysis is because written review gives us actual opinion of a user on a particular video game than numbered rating. Therefore using sentiment analysis we can generate ratings that accurately represents the opinion the review.

2.4 Algorithms

There are a number of algorithms that are used in supervised machine learning for sentiment analysis. In this research paper Naïve Bayes classifiers, Support Vector Machine classifiers, Logistic Regression classifier, Stochastic Gradient Descent classifier have been used for sentiment analysis.

2.4.1 Naïve Bayes

Naïve Bayes is a classification technique that is based on the Bayes' theorem. Initially a Naïve Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Even if the features of a class depend on each other, all of the properties independently contribute to the determining of the probability. Naïve Bayes model is easy to implement and it is particularly useful for large data sets as Naïve Bayes is known for its

scalability. Even though Naïve Bayes is relatively simple algorithm, it is known to outperform some of the sophisticated classifiers [9]. As mentioned before it is based on the Bayes' theorem. Bayes' theorem calculates posterior probability P(c|x) from P(c),P(x), and P(x|c). Below is the mathematical representation of the equation.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad \text{(1)}$$

Here,

P(c|x) is the probability of the class(c) given predictor(x)

P(c) is the prior probability of class

P(x|c) is the probability of predictor given class

P(x) is the prior probability of predictor.

The Naïve Bayes algorithm uses Bayes' theorem with strong independent assumptions. It is a conditional probability. Conditional probability means something may happen given that something related has already happened. It can predict the probability for each class meaning given a data it can provide probability whether a data point belongs to a particular class or not. Given possible m classes A={a1,a2,…….a_m} for reviews R={t1,t2,……tm} then using the Bayes' rule probability of the review r to be in a class can be predicted.

$$P(a|r) = \frac{P(a)P(r|a)}{P(r)} \quad \text{(2)}$$

To simplify in plain English it can be written as

$$posterior = \frac{prior * likelihood}{evidence} \quad \text{(3)}$$

Now in Naïve Bayes it assumes that each feature is conditionally independent of every other feature. Therefore the equation becomes

$$P(a|t) \propto P(a) \prod_{k=1}^{nd} [P(wk|a)]^{tk} \quad \text{(4)}$$

In this research paper Scikit learn(python library) has been used. In Scikit learn there are three types of Naïve Bayes model under the library. Among them we have used Multinomial Naïve Bayes and Bernoulli Naïve Bayes.

2.4.2 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for text classification [10]. In the algorithm each data item is considered as a point in n-dimensional space with the value of each feature being the value of a particular coordinate. Here n is the number of features. Then the classification is performed by finding the hyper-plane that differentiate the classes. Therefore given labeled training data, the algorithm outputs an optimal hyper-plane which categorizes new data.

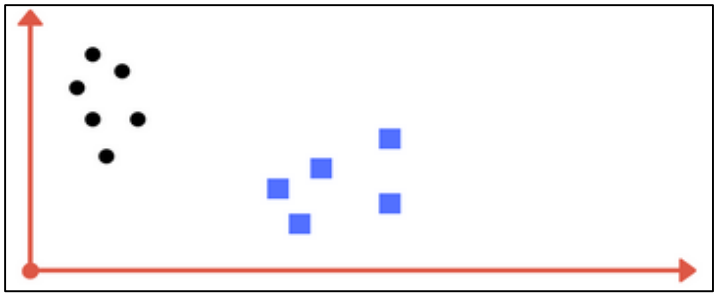

Figure 1: SVM Classification of Two Classes

Here is a plot of two label classes on graph as shown in Figure 1. Now the support vector machine algorithm will classify these two classes. It will find out a hyper-plane that separates the two classes by a clear gap that is as wide as possible as shown in Figure 2.

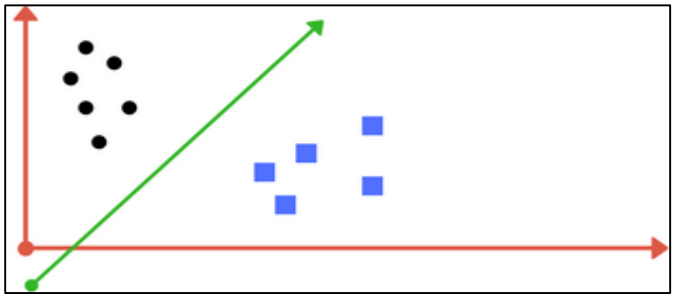

Figure 2: SVM Classification using Hyper-Plane

For a classification task with only two features as shown in Figure 2, a hyper-plane is a line separates and classifies the data set. The further the data are from the hyper-plane lie, the more

confident we are that they have been correctly classified. Therefore to classify data using support vector machine the data points need to be as far as possible while being on the correct side of the hyper-plane. Therefore when new test data is run on the algorithm, whatever side of the hyper-plane it lands will decide the class that is assigned to it. Support vector machine algorithm choose a hyper-plane with the greatest possible margin between the hyper-plane, this is done to correctly classify the data set. In real world data set it is rare that the data points are clearly separable. The dataset will be full with jumbled data points as shown in Figure 3.

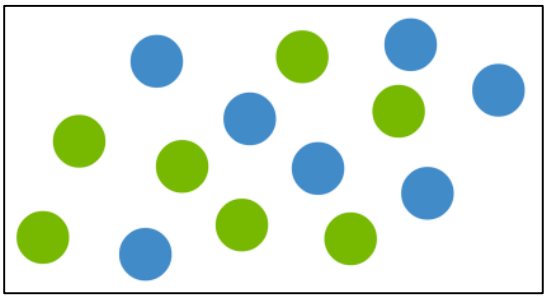

Figure 3: SVM Classifications of Jumbled Dataset

Therefore to classify the data set the SVM algorithm moves away from a 2D view of the data to a 3D view. Then the data points are mapped in the higher dimension, it is known as kernelling. Then the data is mapped into higher and higher dimensions until a hyper-plane can be drawn to separate the classes as shown in Figure 4.

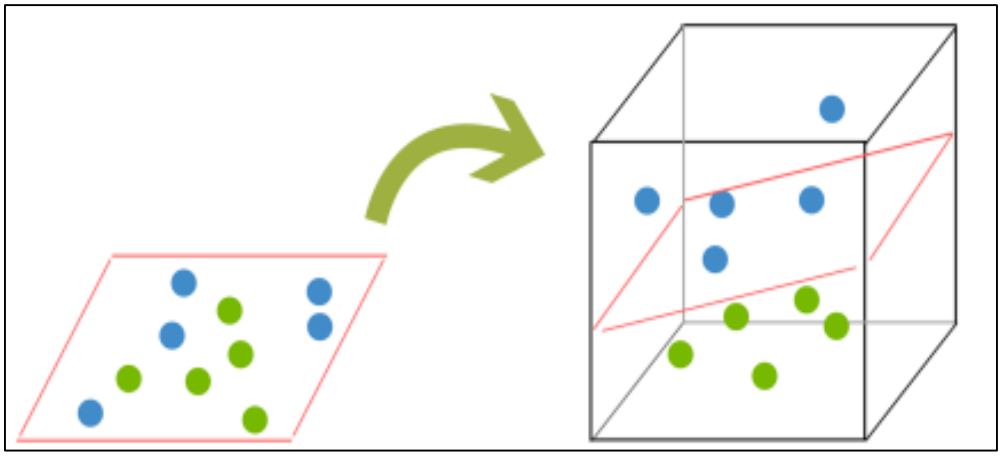

Figure 4: SVM Classifications in Higher Dimension

In this research Scikit learn library was used to implement the support vector machine algorithm. In Scikit lean library there is different implementation of the support vector machine

algorithm such as NuSVC, Linear SVC. In the research all three of them have been used for sentiment analysis. Because the implementations are different from each other, the results also vary from each other.

2.4.3 Stochastic Gradient Descent (SGD)

Stochastic gradient descent algorithm is based on gradient descent optimizing algorithm. Gradient descent is the process of minimizing a function by following the gradients of the cost function. This involves knowing the form of the cost along with the derivative so that from given point that is already a known gradient can move in the direction. Gradient Descent can be used to learn a set of classifier coefficients for parameterized learning. The implementation of gradient descent can be slow to run on the large datasets. This is where Stochastic Gradient Descent is being used. SGD is a simple modification to the standard gradient descent algorithm. SGD is a machine learning algorithm that is capable of making thee classifier learn even if it's based on non-differentiable loss function [11].In SGD it evaluates and updates the coefficients every iteration to minimize the error of a model on the training data. In this optimization algorithm each instance of the training data shown to the model one at a time. Then algorithm predicts based on that training instance. After predicting the error is calculated and the model is updated to reduce the error for the next iteration. This procedure is used to find the set of coefficients in a model that result in the smallest error on the training data. In each iteration the coefficients are updated using equation No.5.

$$b = b - learning\_rate * error * x \quad (5)$$

Here,

b is the coefficient being optimized.

learning_rate is the learning rate that is conFigure d.

error is the prediction error for the model on the training data attributed to the coefficients

x is the input value.

Therefore SGD can convergence faster on the data set with minimum error without any loss on accuracy. In the Figure 5 and No.6 pseudo code of SGD and gradient descent has been shown.

```
Vanilla gradient descent                                                    Python
1 while True:
2     Wgradient = evaluate_gradient(loss, data, W)
3     W += -alpha * Wgradient
```

Figure 5: Gradient Descent Algorithm without Batch

```
Stochastic Gradient Descent (SGD)                                           Python
1 while True:
2     batch = next_training_batch(data, 256)
3     Wgradient = evaluate_gradient(loss, batch, W)
4     W += -alpha * Wgradient
```

Figure 6: SGD with Training Batch

2.4.4 Logistic Regression

Logistic Regression is classification algorithm that is used to predict the probability of a categorical dependent variable. It is a popular statistical technique to model a binomial outcome with one or more explanatory variables. Logistic Regression is an algorithm that is preferred in many studies because of its competitiveness in terms of CPU and memory consumption [12]. Logistic Regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function. The probability function used in Logistic Regression is the "Sigmoid Function" which is shown in equation No.6.

$$Transformed = 1 / (1 + e^{(-z)}) \ldots\ldots\ldots\ldots\ldots\ldots \quad (6)$$

Logistic regression predicts the probabilities of an event occurring. Data points are modeled using the standard logistic function by Logistic Regression, which is an S- shaped curve. Logistic Regression algorithm solves the equation No.7.

$$Y = \text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 * x1 + \cdots + \beta_k * xk = Bt.X \ldots\ldots\ldots\ldots\ldots\ldots \quad (7)$$

Where,

- p = probability that y=1 given the values of the input features, x.
- x1,x2,..,xk = set of input features, x.
- B0,B1,..,Bk = parameter values to be estimated via the maximum likelihood method. B0,B1,..Bk are estimated as the 'log-odds' of a unit change in the input feature it is associated with.
- Bt = vector of coefficients
- X = vector of input features

Probability, odds and log odds are needed to estimate the values of B0,B1,…Bk.

Here,

Probability of an event = (no. of instances of that event) / (total no. of instances present) which ranges from 0 to 1.

Odds of an event = (probability of occurrence of the event) / (probability of not occurring the event) which ranges from 0 to ∞

Log odds = Log(Odds) which ranges from -∞ to +∞

Logistic regression model takes real-valued inputs which then transformed into the range [0,1] and makes a prediction whether the probability of the input belongs to the default class which is class 0. If the probability is greater than .5 then the output is considered to be in the default class otherwise the predictions implies that the output is on the other class which is class 1.

CHAPTER 3

Proposed Model

3.1 Overview

Following are the steps taken in our proposed model:

- The first step is the collection of the dataset which we used to train and test algorithm.
- The second step is the collection of twitter data of which we will generate the rating of.
- The third step is pre-processing of gathered data to a supervised form using multiple data pre-processing methods.
- The fourth step is to train the algorithms to the training set.
- The fifth step is to test the algorithm on the testing set.
- The sixth step is to use a classifier that will act as a voting classifier for the used algorithms.
- The seventh step is to test the accuracy of the classifiers and show the graphical representation of the accuracy.
- The eighth step is to do sentiment analysis on the gathered Twitter data to get polarity.
- The ninth step is to analyze results and generate rating and show graphical representation of the generated rating using the classifiers.

Figure 7 demonstrates the block diagram that represents the implementation procedure of the proposed model.

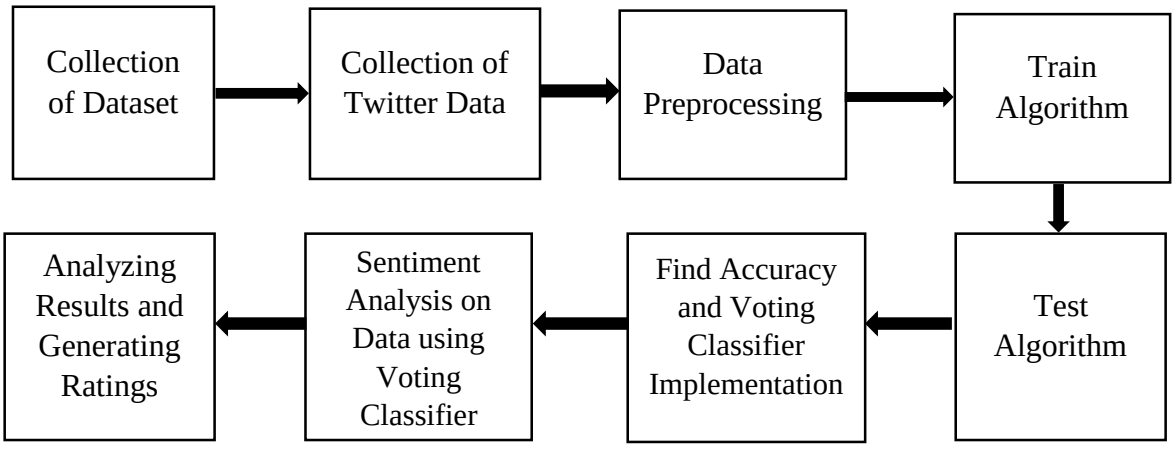

Figure 7: Block Diagram of Workflow of the Proposed Model

3.2 Dataset Collection

In this research we used Amazon product data's video game reviews [18]. This dataset consisted of 53780 user reviews of video games. The columns in the dataset are consisted of reviewerID, asin, reviewerName, helpful, reviewText, overall, summary, unixReviewTime, reviewTime. Where reviewerID is the unique identification number of the reviewer, asin is the identification number of the product in this case the video game, reviewerName is the username of the person who wrote the review, helpful is the helpfulness of the given review, overall represents the rating of the video game given by the user on a scale of 1 to 5, summary is the column consisting review summary, reviewText represents the review written by the user, unixReviewTime is the unix time of the review and reviewTime is the raw time of the review.

3.3 Collection of Twitter Data

Twitter has provided their REST API which can be used by the developers to access and read Twitter data, along with that to access data in real time they have also provided the Streaming API. Most programs that are used to access Twitter data are limited the limitation of the Twitter provided API's. With their search API only 180 requests can be sent every 15 minutes. Where the maximum number of tweets that can be retrieved is limited to only 100 per request. The biggest disadvantages of the Search API is that users can only get access to Tweets that are written in the past 7 days. This is a serious problem as we needed to collect older data from Twitter. Therefore to collect Twitter data we used a script to scrape tweets of the video games from Twitter using the python package requests to retrieve the content and Beautifulsoup4 to parse the retrieved content. Beautiful Soup is a Python Library which is used for pulling data out from HTML and XML files. In Figure 8, extraction of Twitter data is shown.

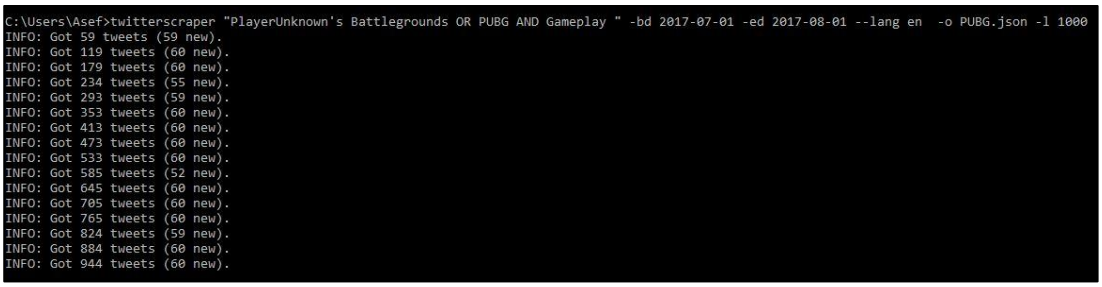

Figure 8: Scrapping Tweet Data

The following information can be retrieved using the script:

- Username and full name
- Tweet id
- Tweet url
- Tweet text
- Tweet html
- Tweet timestamp
- No. of likes
- No. of replies
- No. of retweets

With this we can extract a given number of tweet with specific time period. Therefore using script we scrap the tweets of the video games that we will generate the ratings of without any limitation.

3.4 Data Pre-Processing

Data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data. Proper preparation of the data is a necessary step, not just for a valid experiment but also to enable mining of a dataset using the means of machine learning in the first place [13]. A set of preprocessing steps required for the machine learning framework and algorithms to be able to read and analyze the data, as well as for reducing the dataset to contain the data points and attributes relevant for the study. Likewise it might also be relevant to create or calculate additional attributes from the data, if such derived attributes might be able to aid the analysis and thereby enable better predictions. As we used data from social media, it is necessary to clean the data sets wisely. Basically the data from social media like Twitter cannot be extracted in particular way. So we have used our own techniques for cleaning those tweets for analyzing sentiments properly.

3.4.1 Lemmatization

Lemmatization is the algorithmic process of determining the lemma of a word based on its intended meaning. Unlike stemming, lemmatization depends on correctly identifying the intended

parts of speech and meaning of a word in a sentence, as well as within the larger context surrounding that sentence, such as neighbouring sentences or even an entire document. As a result, developing efficient lemmatization algorithms is an open area of research [14].
A lemma (plural lemmas or lemmata) is the canonical form, dictionary form, or citation form of a set of words. For example run, ran,runs, running are forms of the same lexeme, with run as the lemma. In short lemmatization handles matching (in English) “car” to “cars” along with matching “car” to “automobile”.

We have used the resource WordNet from the platform NLTK (Natural Language Toolkit) for doing lemmatization. As for NLTK, the WordNet Lemmatizer does use the part of speech,that was provided by us. Passing it "dove" and "v" yields "dive" while "dove" and "n" yields “dove".

So first of all we have tokenized the whole tweet using word_tokenize from NLTK. Than we have pos_tag to have the tag values of those tokens. After that for classifying the tokens according to their parts of speech we have designed our get_pos_tag method for retrieving the classification. In Figure 9 the process of parts of speech (POS) tagging is shown.

```
def get_pos_tag(tag):
    if tag.startswith('V'): #verb
        return 'v'
    elif tag.startswith('N'): #noun
        return 'n'
    elif tag.startswith('J'): #adjective
        return 'a'
    elif tag.startswith('R'):#adverb
        return 'r'
    else:
        return 'n'
```

Figure 9: POS Tagging

Next we have processed the tokens using the WordNet Lemmatizer. Because of the limitation of the WordNet we have manually removed the ‘ly’ part from the adverb to acquire the desired lexeme. For that we have replaced the ‘ly’ with blank token ‘ ’.

In Figure 10 we show a Tweet from Twitter before lemmatization process and in Figure 11 we show the result after lemmatization process on the Tweet.

GTAV makes the city of Los Santos feel like a living world where anything can happen. First time playing makes me feel like omg, multiplayer's so much fun #firmhandshakes http://bit.ly/2mwVH2C Follow me @twitter facebook: www.facebook.com/dennis.johnson.37

Figure 10: Tweet before Lemmatization

```
=> Lemmatization Done:

GTAV make the city of Los Santos feel like a live world where anything can happen. First time
play make me feel like omg, multiplayer's so much fun #firmhandshakes http://bit.ly/2mwVH2C.
Follow me @twitter facebook: www.facebook.com/dennis.johnson.37
```

Figure 11: Result of Lemmatization on the Tweet

3.4.2 General Cleaning

Next we have dropped all the blank rows from the data table. Because blank rows bear no sentiment. After that we have taken care of all the hyperlinks from the data. We have used regular expression with the pandas from python. For that we have identified the words starting with 'http', 'https' or 'www'. Whenever a word starting with those strings, we trace out the whole link string. After that we simply remove it with blank string. In Figure 12 we show the result after removing the hyperlinks.

```
=> Hyperlinks Removed:

GTAV make the city of Los Santos feel like a live world where anything can happen. First time play
make me feel like omg, multiplayer's so much fun #firmhandshakes. Follow me @twitter facebook:
```

Figure 12: Result of Removing Hyperlinks from Tweet

Next we removed the twitter tags. Twitter tags are used to tag someone or something, for example: for tagging a person or a place etc. These tags bears no sentiments. That's why we have eliminated these tags using string replacement. Figure 13 demonstrates the result of removing Twitter tags.

```
=> Tweeter Tag Removing Done:

GTAV make the city of Los Santos feel like a live world where anything can happen. First
 time play make me feel like omg, multiplayer's so much fun #firmhandshakes . Follow me
facebook:
```

Figure 13: Result of Removing Twitter Tags

Next we have removed all the non ASCII characters. These non ASCII characters has no influence on sentiment analysis. Rather these characters increase unnecessary crowd on data. Here we have used the built-in python string replace method str.replace() to get rid of the non ASCII characters. Using regular expression we have got rid of the HTML Tags, Numbers, the Hash Tags and the unnecessary commas. In the Figure 14, the result of removing HTML tags, numbers, Hash tags and unnecessary commas are shown.

```
=> Hash-Tag Removing Done:

GTAV make the city of Los Santos feel like a live world where anything can happen. First
 time play make me feel like omg, multiplayer's so much fun . Follow me facebook:
```

Figure 14: Result of Removing non-ASCII characters, Hash tags, numbers

Finally we have removed all the apostrophes and replace them to their original form, for example: “Neil’s going to San Francisco” is changed to “Neil is going to San Francisco”. Figure 15 demonstrates the result of removing the apostrophes.

```
=> Apostrophe Removing Done:

GTAV make the city of Los Santos feel like a live world where anything can happen First
time play make me feel like omg multiplayer is so much fun  Follow me facebook
```

Figure 15: Result of Removing Apostrophes

Next, we have removed all the unnessessary punctuations from the tweets. For doing so, we have used regular expression and string replace method. After removing punctuations the Tweet looks like as shown in Figure 16.

```
=> Punctuations Removing Done:

GTAV make the city of Los Santos feel like a live world where anything can happen First
time play make me feel like omg multiplayer's so much fun  Follow me facebook
```

Figure 16: Result of Removing Unnecessary Punctuations

Finally we have replaced the short formed words and informal words with their actual forms. For example : ‘omg’ becomes ‘oh my god’ and ‘h8’ becomes ‘hates’ etc. Therefore, the previous Tweet after going through the process becomes as shown in Figure 17.

```
=> Slang Words Removing Done:

GTAV make city Los Santos feel like live world anything happen First time play make feel
 like oh my god multiplayer much fun  Follow
```

Figure 17: Result of Removing Slang Words

3.4.3 Stop Words

Stop words are words which are filtered out before or after processing of natural language data (text). Some examples of stop words are: "a", "and", "but", "how", "or", and “what.”. These words increase rush on the data, having no sentiment values. So we removed these

stop words. We have used the stopwords (english) resource from nltk corpus. In Figure 18 the result of removing stop words is shown.

```
=> Stop Words Removing Done:

GTAV make city Los Santos feel like live world anything happen First time play make feel
 like omg multiplayer much fun  Follow
```

Figure 18: Result of Removing Stop Words

3.4.4 Handling Negative Adjectives

"This video game was not so good" this type of syntax will result in positive outcome though the meaning denoted negative emotion. To overcome this issue, we have checked for this type of syntaxes (not good/ not so good, not bad / not so bad) in our data and replaced them with their respective positive/ negative words.

3.4.5 Featured List

We have created a dictionary consisting of 45,000 words, generated from our feature set. This featured list set is created for advanced cleansing of data. We match every word in the dictionary with the tweeter data. If no word is matched with our featured list, then we drop the whole row from the scraped twitter data file. In figure 19, we show how we can get rid from unnecessary tweets.

```
for i in range(len(data2)):
    for k in range(len(featuredList)):
        if featuredList[k] not in data.text[i]:
            data2 = data2[~data2['text'].isin(featuredList)]
            data2 = data2[data2.text.str.contains('|'.join(featuredList))]
```

Figure 19: Process of Deleting Unnecessary Tweets.

3.5 Training and Testing Data

One of the important parts of Machine Learning model is to split the dataset into two parts. They are:

1. Training Set and 2.Testing Set

The reason we split our dataset into these two parts is to test the accuracy of the algorithms. The idea is that we will train the algorithms on the test set where we already know the answers. Here the algorithms will be trained using the labeled data. In this research, we have chosen Twitter data for generating the ratings of the video games. The problem with directly testing on that data is that future instance, in this case Twitter data, have unknown target values therefor we cannot check the accuracy of the algorithms on that data. That is where testing set helps. By splitting the Amazon dataset into training set and test set we can test the algorithm on the test set where the reviews were manually labeled. We trained 1 and 2 rated reviews as bad reviews and 4 and 5 rated reviews as good reviews. This helps us to see the accuracy of the algorithms that are used in the model. This accuracy will give us a better idea about how correct the generated rating is for each algorithm.

Generally the most usual way to split a dataset into training set and test subset is usually with a ratio of 70-85% percent for training set and 15-30% for the test set [15]. For training set we took 80% of the data and the rest 20% was used for testing set. That means the algorithms are trained on 80% of the dataset then tested against rest of the 20% of dataset.

3.6 Feature Set

In this research, we created a feature set consisting of 45000 words. From dataset we extracted the most frequent 45000 words. Then we created the feature list from the extracted words by finding these top 45000 words in our negative and positive reviews, making their presence as either positive or negative. This feature set allows us find which word appears more in a good review and in a bad review. In this research, we trained the algorithms using the dictionary for sentiment analysis. In the Figure 20 we show the feature set.

Figure 20: Feature Set

3.7 Accuracy and Voting Classifier

As per No Free Lunch Theorem, there cannot be one algorithm that can fit into any data model well enough [16]. Therefore to see which algorithm performs better in our model, two or more algorithms needs to run on the model and needs to be compared against each other to find out which one fits our model better. In this step, we manually increase the number of feature set and each time we test all the classifiers to see accuracy. This will help us to move forward with the research in two very important way. The first one is that in this step we will get to know at what number of feature set the algorithms perform better and the second one is that we will get to know at that number of feature set which algorithm performs the best and which algorithm performs the worst.

In this research, we are going to use Naïve Bayes, Support Vector Machine, Logistic Regression and Stochastic Gradient Descent algorithm models. As mentioned earlier we are using Scikit learn library and NLTK library in Python to implement the algorithms to fit into the model. The reason we are using those library is because they have built in implementation of the classifiers [17]. Therefore to be specific we are going to use seven classifiers of the mentioned algorithms in our model. They are:

- Naïve Bayes(NB), MultinomialNB(MNB), BernoulliNB(BNB)
- LinearSVC, NuSVC

- Logistic Regression(LR)
- Stochastic Gradient Descent Classifier(SGDClassifier)

Multinomial Naive Bayes determines the probability of the classes from the training data, which is denoted as Priors. Priors can be derived from P(c)=N_c/N, where c stands for class. Next the conditional probabilities for each token (w) are derived for the predetermined classes(c). These conditional probabilities are generated using $P(w \mid c)=(count(w,c)+1)/(count(c)+|V|)$. Finally the class of the testing set (P(class|test data)) is determined for each training class by multiplying the Prior and Conditional Probabilities for each occurrence.

Naive Bayes classifier classifies data just like the multinomial naive bayes. The only difference between these two is Naive Bayes classifier refers to conditional independence of each of the features in the model, where Multinomial Naive Bayes classifier is a specific instance of a Naive Bayes classifier which uses a multinomial distribution for each of the features.

Bernoulli Naive Bayes models both the presence and absence of a feature. For a test case it checks whether the features are present and absent in that text. For a given text , BNB calculates both the probability of good and bad class and checks which classification gets more probability value.

Before performing SVM, LR and SGD to the data, we need to assign weight to each word for doing sentiment analysis using those algorithms. For assigning weight to each words TfidfVectorizer class is used in Scikit Learn. This TfidfVectorizer class takes the features, creates a gigantic matrix, which is called bag of words. This consists the words count of every single word. In this class Tfidf part is called term frequency inverse document frequency. Tfidf gives every word a certain weight. How it does is that every word that appears in many of the documents in our case reviews, will get a low rate and words that do not occur often will get a higher rate. TfidfVectorizer class actually does two very important things in one class. There are two separate classes for bag of words and tfidf which are called CountVectorizer and TfidfTransformer. In the combined class TfidfVectorizer both of the steps are done. Ngram is used for telling the algorithms how many words to consider as features when training. In this proposed model, we used the default parameters of the class where ngram_range is (1,1) which means it will tell the algorithms to treat every single word as separate feature. Now after assigning weights to the words we run

LinearSVC, NuSVC, Logistic Regression and Stochastic Gradient Descent Classifier to the dataset to get the accuracy.

Now to find out at which number of feature set the algorithms perform better, we start by setting the feature set to 1000. Then we manually increase the number of feature set in each iteration then test each classifiers with the test set using that same feature set. We continue to do that until we find the best number of feature set where the most classifiers have the best accuracy. In our research we find that when the number of feature set is set to 45000 most classifiers then have the most accuracy.

In Table 1 the accuracy of the classifiers at increasing number feature set is shown.

Table 1: Accuracy of Classifiers at Different Number of Feature Set

Number of feature-sets	NB	MNB	BNB	LR	SGD	LinearSVC	NuSVC
1000	79.67	78.04	79.84	80.76	80.29	80.03	80.63
5000	83.49	81.72	83.86	84.86	84.16	84.12	83.50
10000	86.49	85.71	86.36	87.52	87.29	87.31	86.28
15000	89.25	88.85	89.06	89.94	89.70	91.33	87.82
20000	89.54	89.43	89.23	90.11	89.68	91.74	87.76
25000	89.51	88.59	89.28	90.32	89.64	91.46	88.36
30000	90.11	89.32	89.88	90.30	90.22	91.57	88.42
35000	88.76	87.82	88.63	89.77	89.38	91.05	87.78
40000	89.92	89.13	89.70	90.30	89.53	91.44	88.44
45000	90.22	89.32	90.01	90.18	90.02	92.02	88.16
50000	89.49	89.01	89.32	90.02	89.60	91.48	88.23

Naïve Bayes

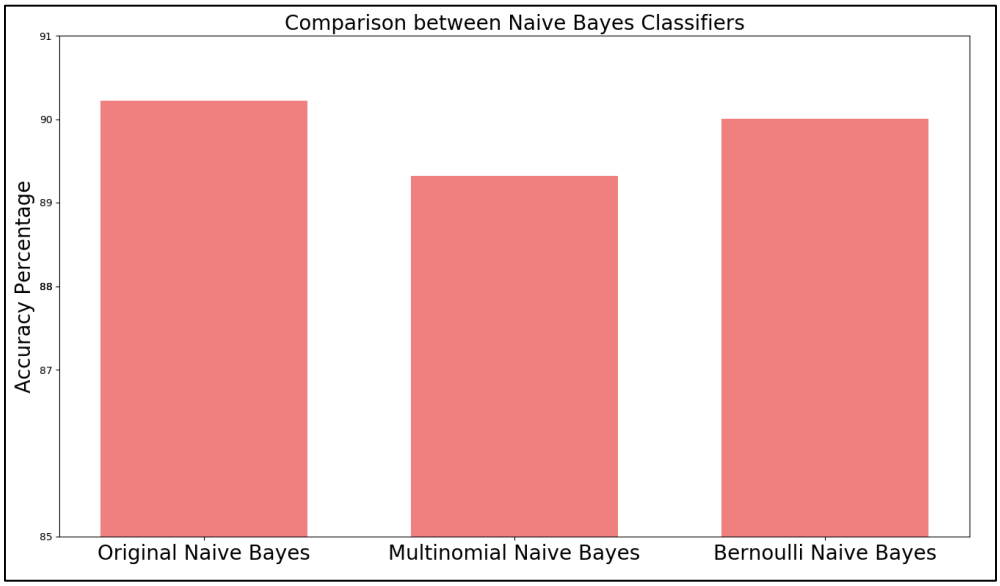

Figure 21: Comparison between Naive Bayes Classifiers

Figure 21 shows the comparison between Original Naïve Bayes, Multinomial Naïve Bayes and Bernouli Naïve Bayes classifiers at 45000 feature set.

Support Vector Machines

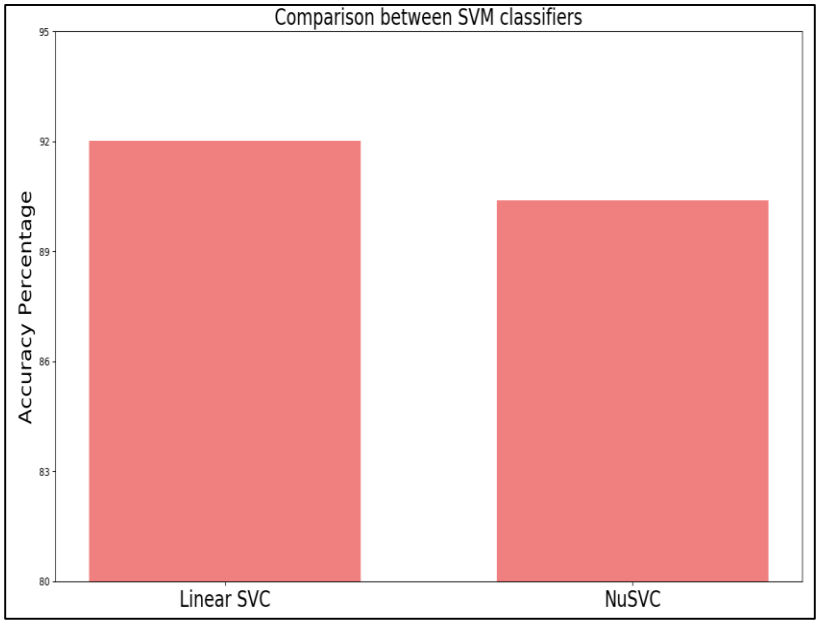

Figure 22: Comparison Between SVM Classifiers

Figure 22 shows the comparison between LinearSVC and NuSVC classifiers at 45000 feature set.

Logistic Regression and Stochastic Gradient Descent

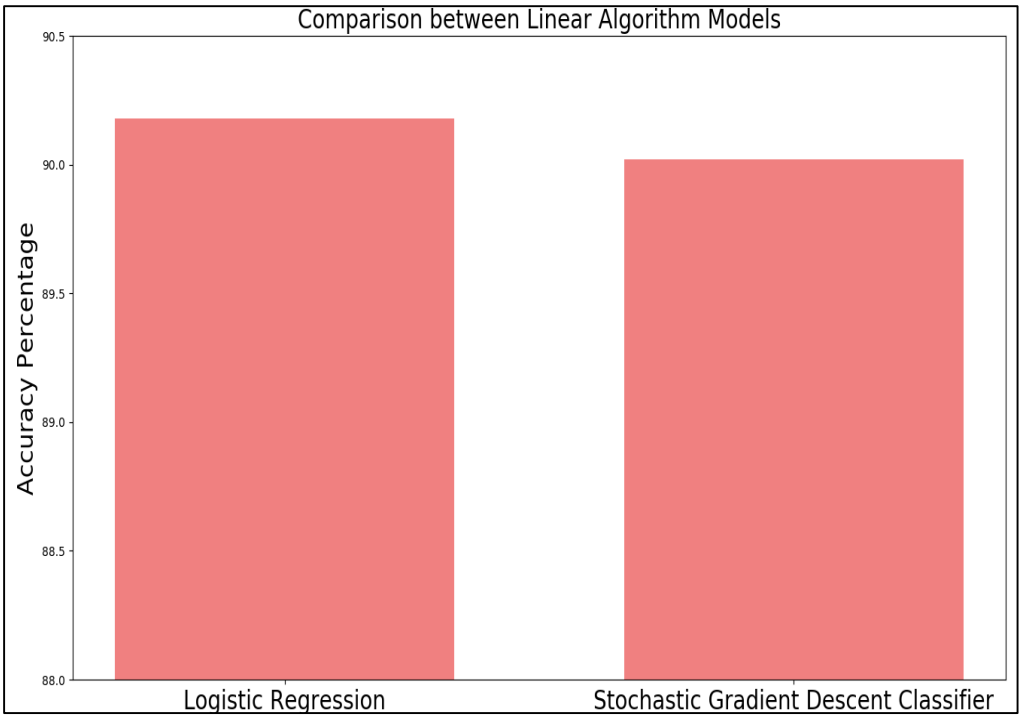

Figure 23: Comparison between Linear Algorithm Models

Figure 23 shows the comparison between Logistic Regression and Stochastic Gradient Descent classifiers at 45000 feature set.

Now if we look at the comparison among the classifiers as shown in Figure 24 we see that LinearSVC has the highest accuracy with 92.019 % and NuSVC has the lowest accuracy among the classifiers with 88.16%.

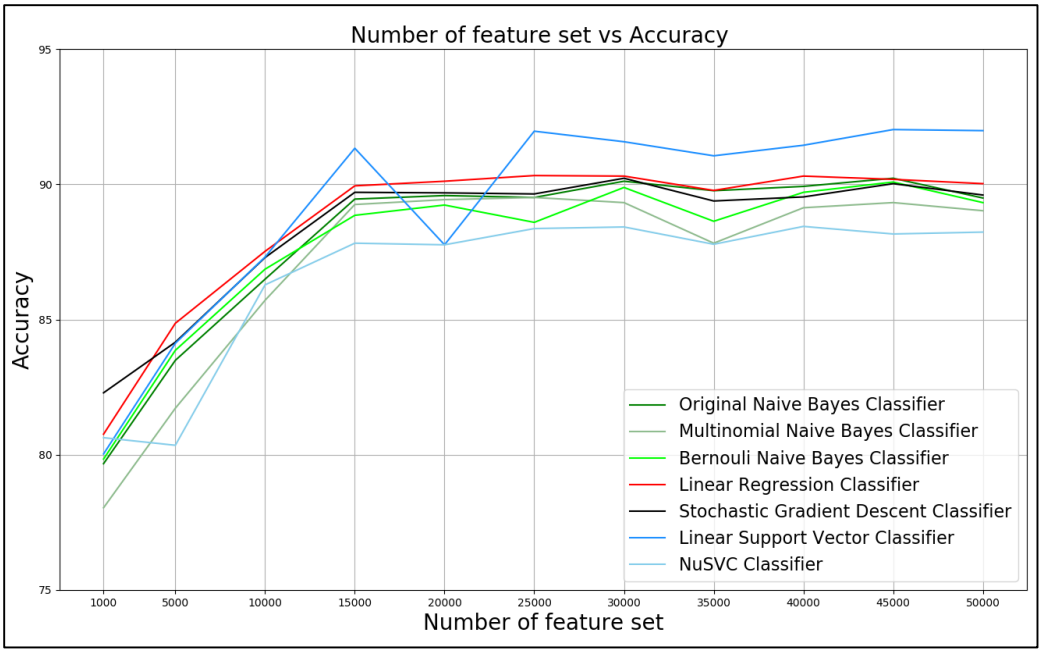

Figure 24: Comparison between classifiers

Now that we have identified the accuracy of the classifiers naturally a question may rise that which algorithm to select for generating rating from user opinion. For this in this step of the research we implemented a new classifier that will work as a voting classifier for all the above classifiers. As mentioned above Naïve Bayes, MultinomialNB, Bernoulli NB, Logistic Regression, SGD, Linear SVC, NuSVC classifiers were used. All of these algorithms uses different ways to classify natural language. As a result, different classifications can be generated for same texts. We felt the necessity of uniting the algorithms to generate ratings. In this classifiers an elective method is implemented that gives an output based on all the classifiers used. This classifier keeps track of each algorithms and how it is classifying a text. The classifier then generates classification of a text based on the most vote given to a classification of that particular text. Then the produced result is being tested against the test set for testing accuracy. In our test we found that the voted classifier has an accuracy of 90.86%. This is in general how this classifier will work. Since this classifier takes inputs from all the classifiers mentioned above and produces results that are based on the most voted classification, this classifier will be used for the sentiment analysis and rating generation.

In the Figure 25 we the accuracy of the classifiers including the accuracy of the voting classifier.

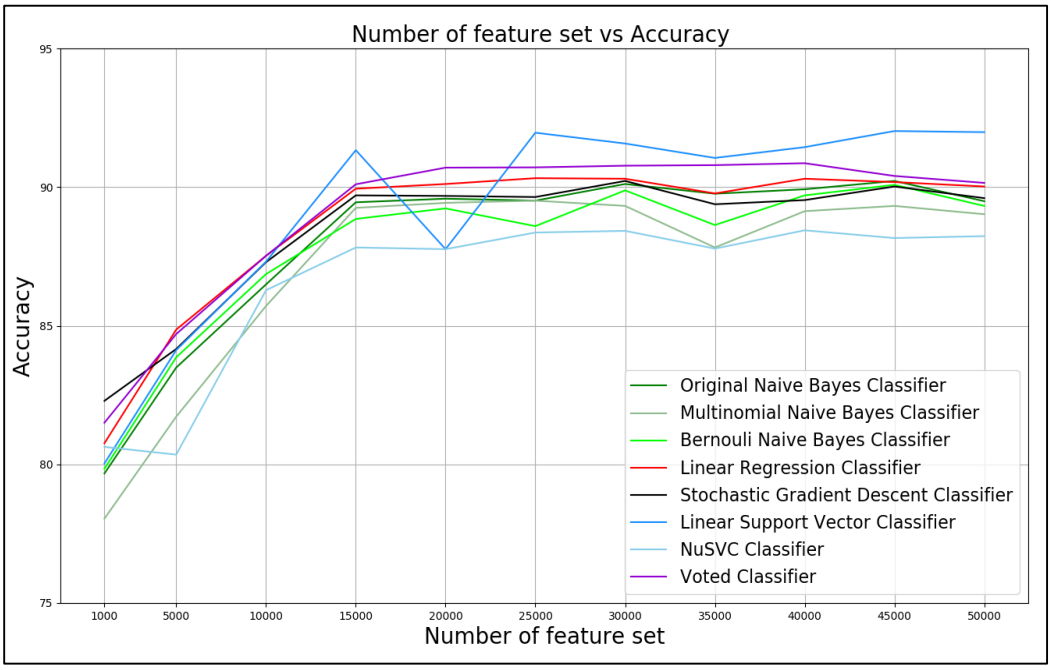

Figure 25: Accuracy of Voting Classifier Compared to Other Classifiers

3.8 Sentiment Analysis on Twitter

Sentiment analysis is the process of determining emotional tone behind some series of words. In this research so far we have trained the classifiers, cleaned the gathered data, tested the classifiers to see the accuracy. We have used Scikit learn library and NLTK platform in Python for sentiment analysis. For sentiment analysis on Twitter we used a voting classifier that uses votes of classification from Naïve Bayes, MultinomialNB, BernoulliNB, LinearSVC, NuSVC, Logistic Regression, Stochastic Gradient Descent Classifier(SGDClassifier). This classifiers was implemented using NLTK platform and Scikit learn library in Python. Pre-processed data was used for training the classifiers and then applied on the Tweet data to generate sentiment polarity.

In Figure 26, we see that the Idle IDE console is giving outputs on the given Twitter data.

Figure 26: Sentiment analysis on tweet data

3.9 Graphical Representation

We used python as our programming language. The library called MatPlotLib is library of python that is widely used for creating graphical representation. With MatPlotLib bar charts, pie charts, box diagram, histogram etc. can be created [23]. In this research we used different kinds of plotting methods to show different kinds of results.

In the Figure 27 and Figure 28 we have shown some sample graphical representation of the sentiment analysis.

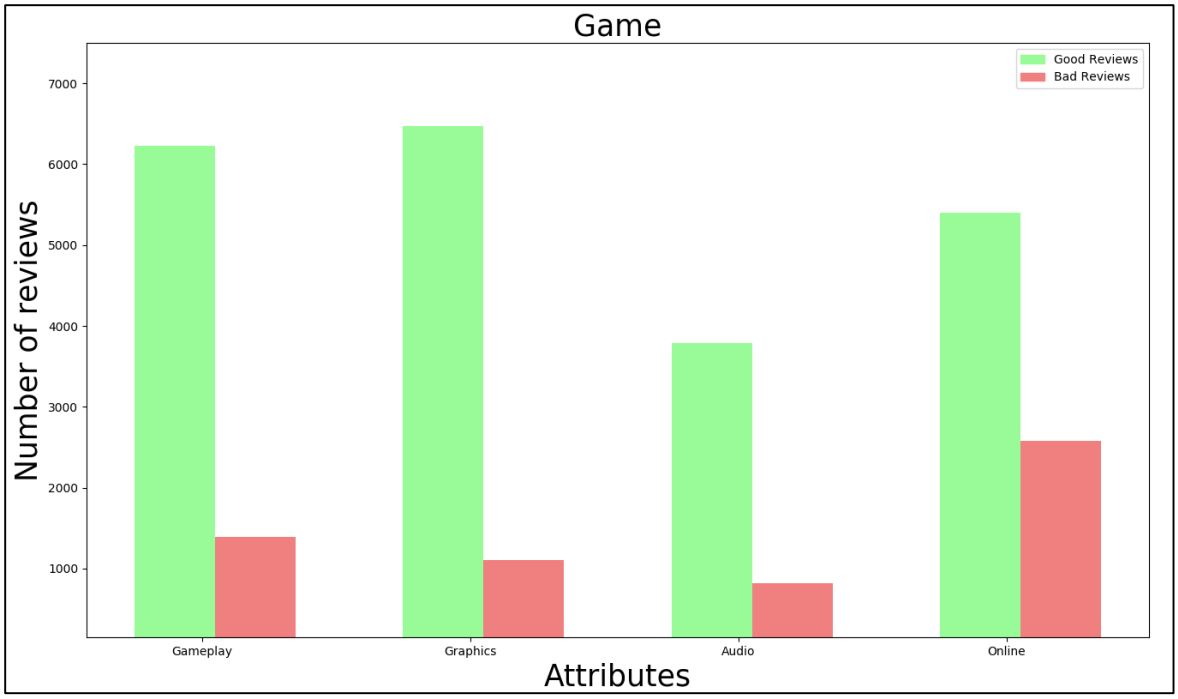

Figure 27: Sample Chart Good vs Bad Reviews

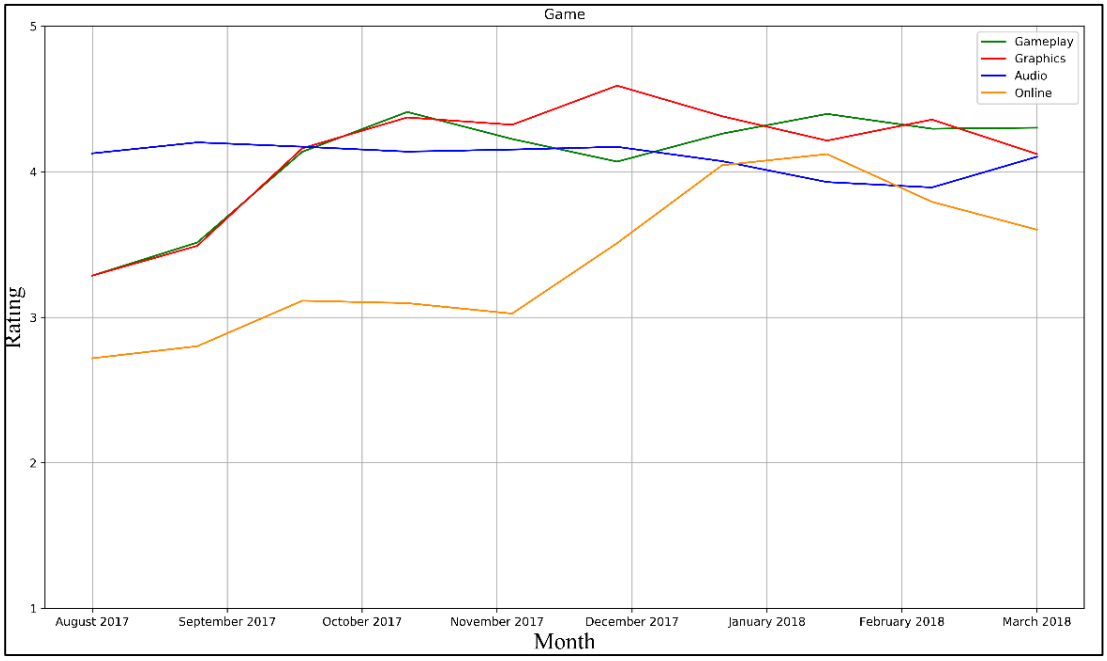

Figure 28: Sample Diagram of Generated Ratings

3.10 Rating Generation

For generating rating we took the polarity values given by voting classifier. Then for each attribute we calculated the rating individually by taking the polarity values and doing arithmetic mean and scaling the values on scale of 1 to 5 to show the rating.

For each attribute,

$$rating\ of\ attribute, x = \frac{good\ reviews\ of\ attribute\ x}{total\ number\ of\ reviews\ of\ attribute\ x} * 5 \ldots\ldots\ldots\ldots\ldots (8)$$

For overall rating each game,

$$overall\ rating = \frac{number\ of\ good\ reviews\ of\ all\ attributes}{total\ number\ of\ review\ of\ all\ attributes} * 5 \ldots\ldots\ldots\ldots\ldots\ldots (9)$$

CHAPTER 04

Experimental Setup & Result Analysis

The experiment was performed in this research in the following setup:

- Intel Core-i5 4590
- 16 GB RAM
- Intel 535 series SSD

For the programming language we used Python (v3.6.2) and used Idle IDE. To implement the algorithm classifiers we used Scikit Learn library. Scikit Learn is used for research in machine learning data analysis as Scikit Learn has machine learning algorithms built into its library [24]. We trained each algorithm on a dataset of amazon video game reviews and used a script to collect Twitter data.

After applying each algorithm model, we get the sentiment of those review and from the sentiment polarity we calculated ratings.

Table 2: Good and Bad Reviews of "Grand Theft Auto V" for last 8 months

Attributes	Good Reviews	Bad Reviews
Gameplay	6225	1386
Graphics	6470	1109
Audio	3784	818
Multiplayer	5405	2579

Table 2 shows the number of good reviews and bad reviews for each attribute of the game Grand Theft Auto V.

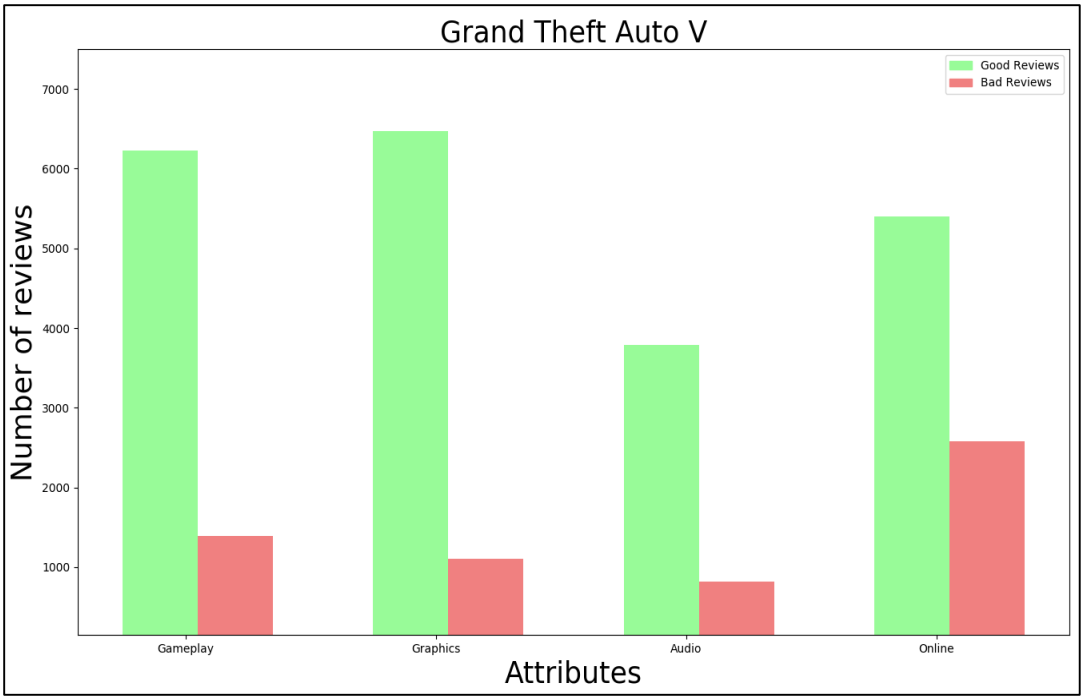

Figure 29: "Grand Theft Auto V" polarity

In the Figure 29 we show number of good and bad reviews based on our sentiment analysis of each attributes for "Grand Theft Auto V".

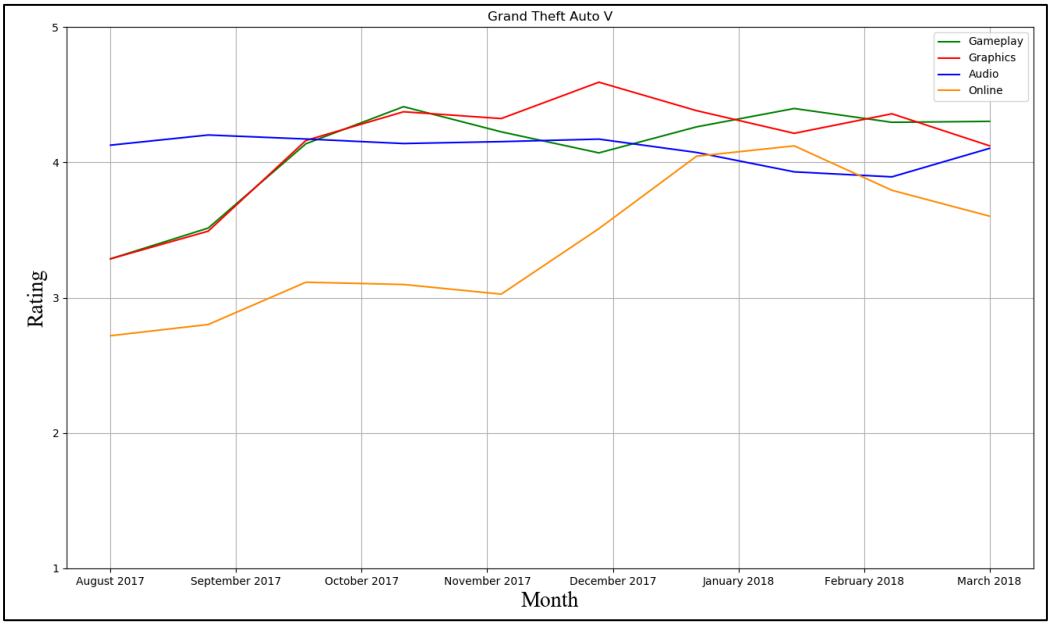

Figure 30 "Grand Theft Auto V" rating

Figure 30 is a time vs. rating graph for the video game “Grand Theft Auto V”. In the graph x-axis is the time and y-axis is the generated rating. We averaged the polarities of the data from one month and input the result with respect to the month. This graph shows the users’ opinion shifting from month to month as the game gets new patches and updates.

Table 3: Good and Bad Reviews of “Ghost Recon Wildlands” for last 8 months

Attributes	Good Reviews	Bad Reviews
Gameplay	5101	2580
Graphics	4842	1054
Audio	2902	723
Multiplayer	4466	1466

Table 3 shows the number of good reviews and bad reviews for each attribute of the game Ghost Recon Wildlands.

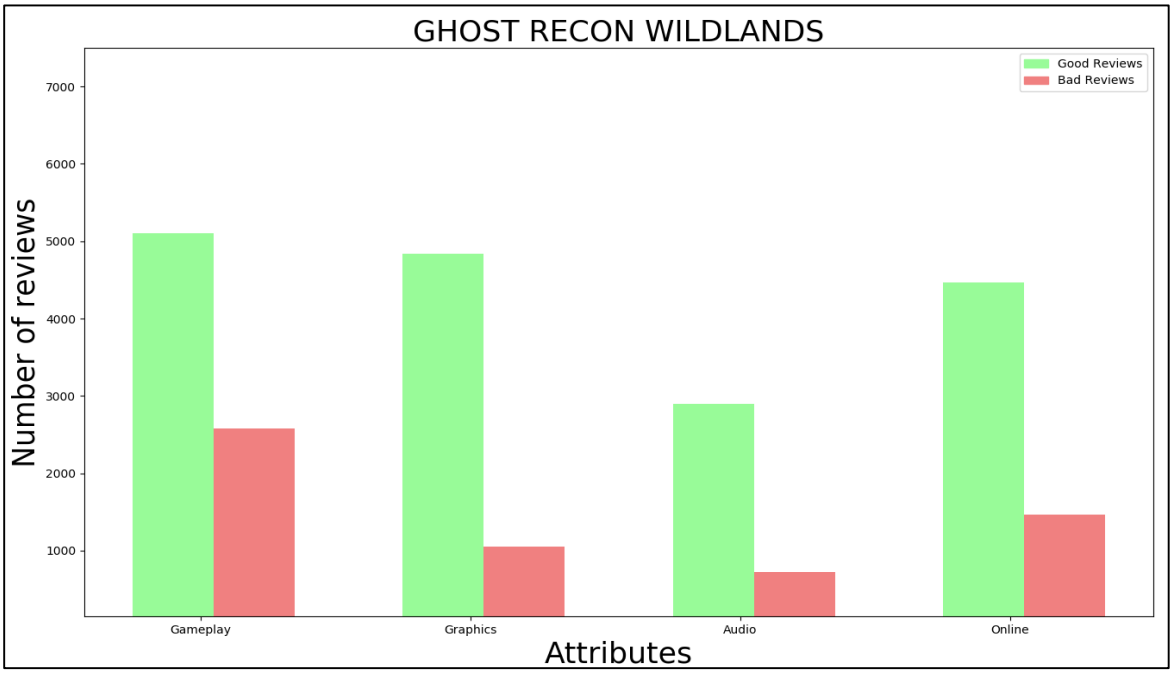

Figure 31: “Ghost Recon Wildlands” polarity

In Figure 31 we show number of good and bad reviews based on our sentiment analysis of each attributes for “Ghost Recon Wildlands”.

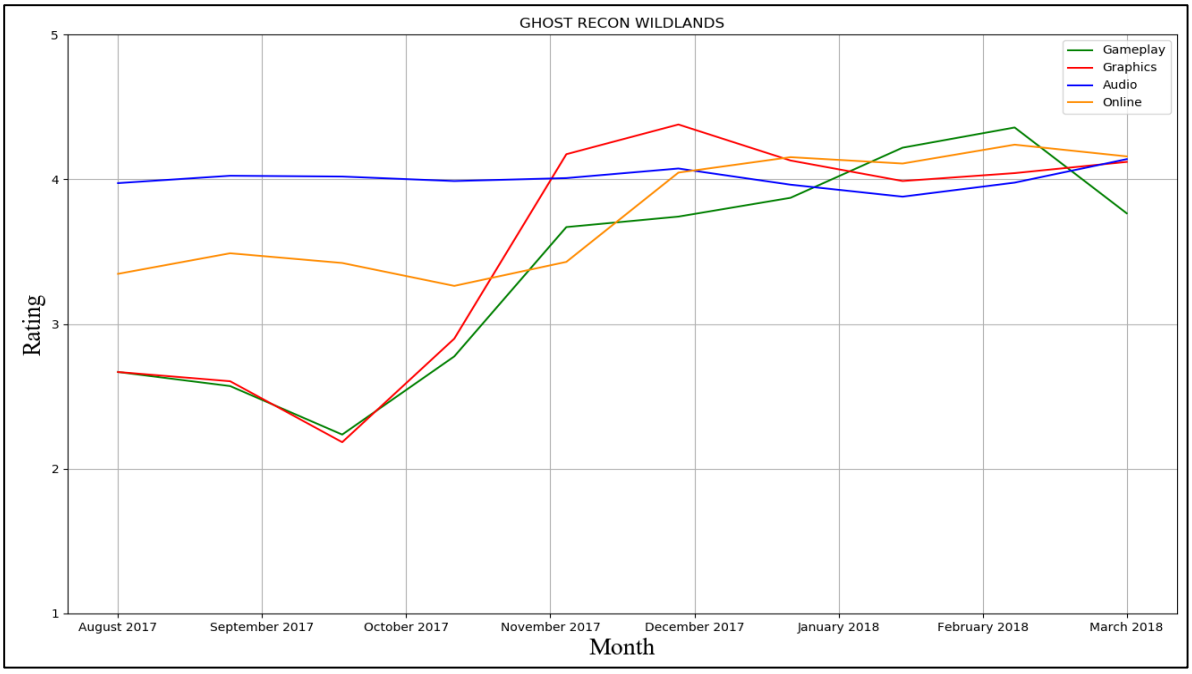

Figure 32: Generated Rating of "Ghost Recon Wildlands"

Figure 32 is a time vs. rating graph for the video game "Ghost Recon Wildlands". In the graph x-axis is the time and y-axis is the generated rating. We averaged the polarities of the data from one month and input the result with respect to the month. This graph shows the users opinion shifting from month to month.

Table 4: Good and Bad Reviews of "PlayerUnknown's Battleground" for last 8 months

Attributes	Good Reviews	Bad Reviews
Gameplay	5909	1866
Graphics	5548	2358
Audio	4929	1085
Multiplayer	6218	1482

Table 4 shows the number of good reviews and bad reviews for each attribute of the game PlayerUnknown's Battleground.

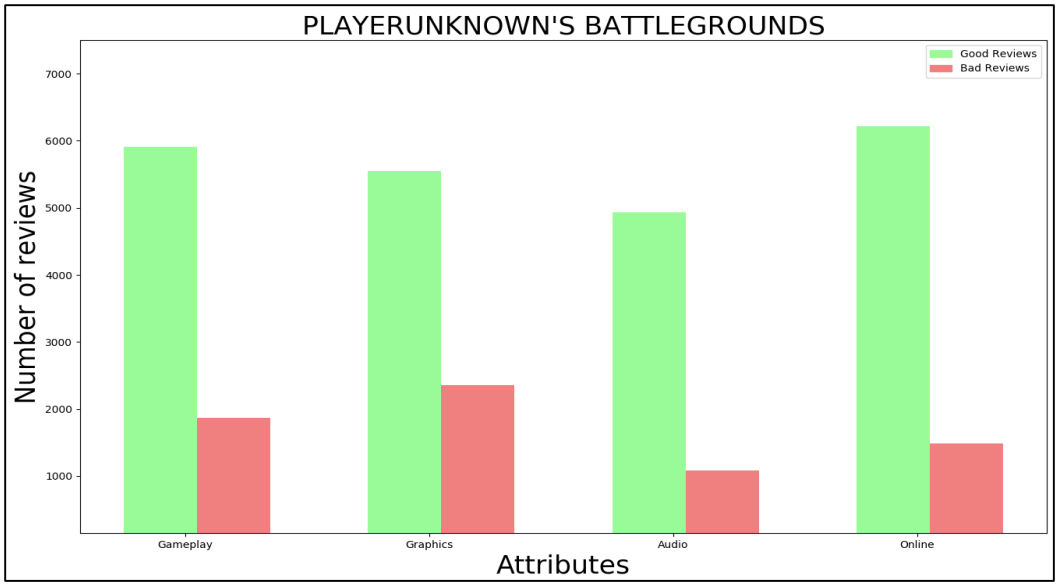

Figure 33: "PlayerUnknown's Battleground" polarity

In Figure 33 we show number of good and bad reviews based on our sentiment analysis of each attributes for "PlayerUnknown's Battleground".

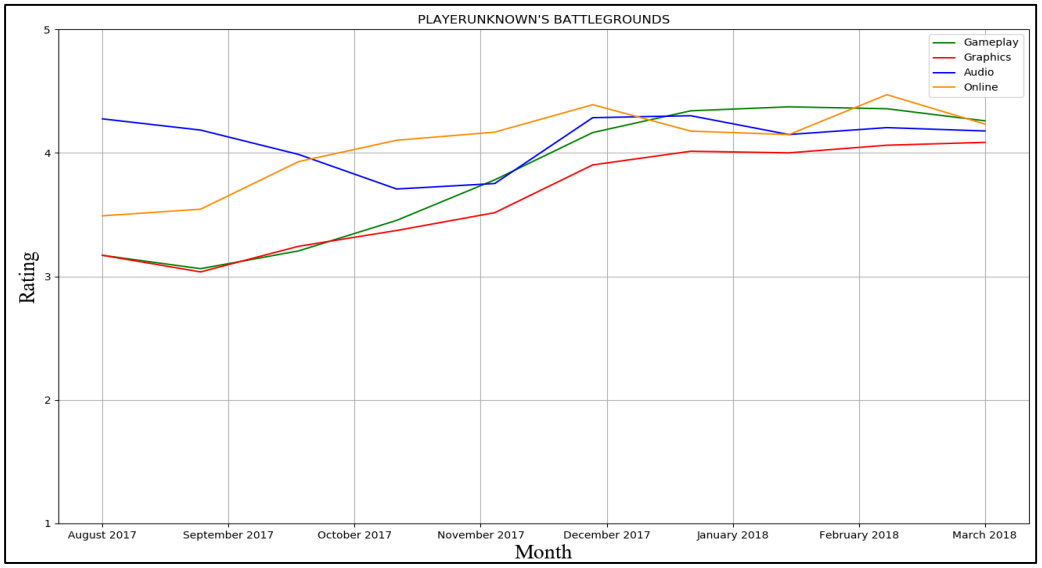

Figure 34: Generated Rating of "PlayerUnknown's Battleground"

Figure 34 is a time vs. rating graph for the video game "Playerunknown's Battleground". In the graph x-axis is the time and y-axis is the generated rating. We averaged the polarities of the data from one month and input the result with respect to the month. This graph shows the users opinion shifting from month to month.

Table 5: Total Number of Good and Bad Reviews of All Three Games for last 8 months

Game	Good Reviews	Bad Reviews
GTA V	21884	5892
GRWL	17311	5823
PUBG	22604	6791

Table 5 shows the total number of good reviews and bad reviews of all three games for last 8 months.

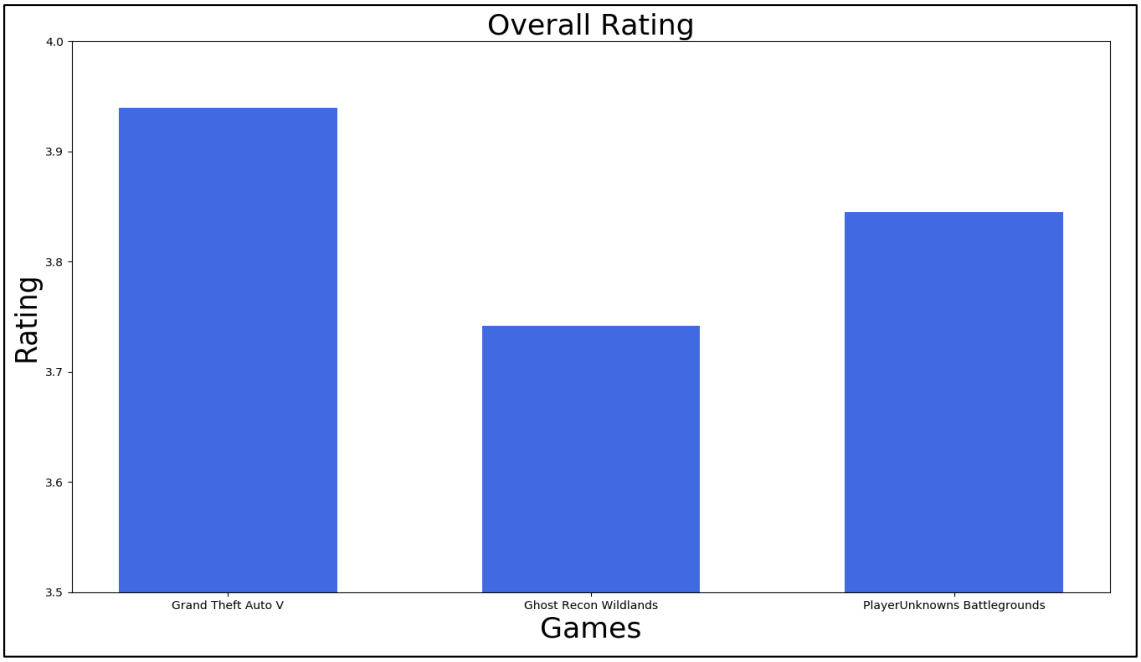

Figure 35: Rating of Games by Averaging the Attributes

In Figure 35 we show the rating of each games. The rating were generated by doing arithmetic mean of the attributes of each game. Here the generated rating is the representation of the current state of the respective video games. From Figure 33 we see that Grand Theft Auto V has an average rating of 3.94, Ghost Recon Wildlands has an average rating of 3.74 and PlayerUnknown's Battlegrounds has a rating of 3.84. All of these rating represents how the games are being received by the public.

CHAPTER 05

Conclusion and Future Work

5.1 Conclusion

Game rating on professional websites do not represent the current state of a video game therefore we wanted to build a model where user can get accurate representation of the current state of a video game. In this research, we have proposed a model to generate rating of video games based on public opinion. In this model we have used four algorithms to check the performance of the model. In this model we built a feature set consisting of 45000 words and implemented a voting classifier that unifies the outputs of the algorithms and gives a result that is based on the outputs of each algorithm which gives more consistent accuracy compared to the individual algorithms. Then we did sentiment analysis on Twitter data using voting classifier which unites all the other classifiers to get consistently accurate results. From our analysis we saw that user opinion shifts from time to time for a given game.

Public opinion on a given game accurately represents the current state of the said video game. Therefore rating generated by our proposed model reflects public's opinion more accurately than a general rating on a game reviewing website. This model will allow users to analyze numbers of video games along with their attributes based on their rating and choose the best one depending on user's preferred genre and time. Therefore our proposed model will allows users to analyze and find best rated video games.

5.2 Future Work

In this research., we have built a model to rate video game based on public opinion. Our plan is to build on this research and improve this model. For future work we want to analyze games on multiplatform, meaning for a game that is available on more than one platform we want compare how user opinion varies for a game on different platforms and also show rating for each version. Along with that we also want to find a correlation between user opinion and sales number of a game to find out if there is a relation between user opinion of a game that is shifting from time to time and sales number of the game in that given time. In the future, we want to improve this model and bring to a stage where it not only works flawlessly with video game reviews but works

with other art form reviews for example movies, TV shows, books. Along with that we also want to provide a way to make our model more user friendly. Also, we want to implement live sentiment analysis from twitter on our model.

We are looking forward to make decision making process easier when it comes to selecting video games for the users. Our belief is that we can make this model user friendly and accurate enough for users to use this on regular basis for judging video games and save their time from looking to different sites to find good reviews.

Reference

1. K. L. S. Kumar, J. Desai, and J. Majumdar, “Opinion mining and sentiment analysis on online customer review,” *2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 2016.
2. S. K. Khatri and A. Srivastava, “Using sentimental analysis in prediction of stock market investment,” *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2016.
3. M. Hu and B. Liu, “Mining and summarizing customer reviews,” *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 04*, 2004.
4. E. J. D. Fortuny, T. D. Smedt, D. Martens, and W. Daelemans, “Media coverage in times of political crisis: A text mining approach,” *Expert Systems with Applications*, vol. 39, no. 14, pp, 2012.
5. A. Dalmia, M. Gupta, and V. Varma, “IIIT-H at SemEval 2015: Twitter Sentiment Analysis – The Good, the Bad and the Neutral!,” *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015.
6. I. P. Windasari, F. N. Uzzi, and K. I. Satoto, “Sentiment analysis on Twitter posts: An analysis of positive or negative opinion on GoJek,” *2017 4th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, 2017.
7. K. Kaviya, C. Roshini, V. Vaidhehi, and J. D. Sweetlin, “Sentiment analysis for restaurant rating,” *2017 IEEE International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*, 2017.
8. B. Liu, “Sentiment Analysis and Opinion Mining,” *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, 2012.
9. O. Abdelwahab, M. Bahgat, C. J. Lowrance, and A. Elmaghraby, “Effect of training set size on SVM and Naïve Bayes for Twitter sentiment analysis,” *2015 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2015.
10. N. Zainuddin and A. Selamat, “Sentiment analysis using Support Vector Machine,” *2014 International Conference on Computer, Communications, and Control Technology (I4CT)*, 2014.

11. A. Bifet and E. Frank, “Sentiment Knowledge Discovery in Twitter Streaming Data,” *Discovery Science Lecture Notes in Computer Science*, pp. 1–15, 2010.
12. W. P. Ramadhan, S. T. M. T. Astri Novianty, and S. T. M. T. Casi Setianingsih, “Sentiment analysis using multinomial logistic regression,” *2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC)*, 2017.
13. A. Krouska, C. Troussas, and M. Virvou, “The effect of preprocessing techniques on Twitter sentiment analysis,” *2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA)*, 2016.
14. M. Mhatre, D. Phondekar, P. Kadam, A. Chawathe, and K. Ghag, “Dimensionality reduction for sentiment analysis using pre-processing techniques,” *2017 International Conference on Computing Methodologies and Communication (ICCMC)*, 2017.
15. J. Prusa, T. M. Khoshgoftaar, and N. Seliya, “The Effect of Dataset Size on Training Tweet Sentiment Classifiers,” *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, 2015.
16. M. Koppen, D. Wolpert, and W. Macready, “Remarks on a recent paper on the ‘no free lunch’ theorems,” *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 3, pp. 295–296, 2001.
17. B. Samal, A. K. Behera, and M. Panda, “Performance analysis of supervised machine learning techniques for sentiment analysis,” *2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS)*, 2017.
18. McAuley, J. (n.d.). Amazon product data. Retrieved March 24, 2018, from http://jmcauley.ucsd.edu/data/amazon/