**CHAPTER 1**

**INTRODUCTION**

* 1. **INTRODUCTION & BACKGROUND OF PROJECT**

With the increase in the number of mobile applications in the day to day life, it is important to keep track as to which ones are safe and which one aren’t. One can’t judge how safe and true each application is based only on the reviews that are mentioned for each application. Hence it is a need to keep track and develop a system to make sure the apps present are genuine or not. The objective is to develop a system in detecting fraud apps before the user downloads by using sentimental analysis and data mining.

Sentimental analysis is to help in determining the emotional tones behind words which are expressed in online. This method is useful in monitoring social media and helps to get a brief idea of the public’s opinion on certain issues. The user cannot always get correct or true reviews about the application on the internet. We can check for user’s sentimental comments on multiple application. The reviews may be fake or genuine. Analyzing the rating and reviews together involving both user and admin’s comments, we can determine whether the app is genuine or not. Using sentimental analysis and data mining, the machine is able to learn and analyze the sentiments, emotions about reviews and other texts. The manipulation of review is one of the key aspects of App ranking fraud, by using sentimental analysis and data mining, analyzing reviews and comments can help to determine the correct application for both Android and iOS platforms.

Certain times, just for the upliftment of the developers, they tend to hire teams of workers who commit to fraud collectively and provide false comments and ratings over an application. This is known to be termed as crowd turfing. Hence it is always important to ensure that before installing an app, the users are provided with proper and genuine comments in order to avoid certain mishaps. For this, an automated solution is required to overcome and systematically analyses the various comments and ratings that are provided for each application. With mobile phones being a quite popular need, it is essential that suspicious applications must be marked as a fraud in order to be identified by the store users. It will be difficult for the user to determine the comments that they scroll past or the ratings they see is a scam or a genuine one for their benefit. Thereby, we are proposing a system which will identify such fraudulent applications on Play or App store by providing a holistic view of ranking fraud detection system By considering data mining and sentiment analysis, we can get a higher probability of getting real reviews and hence we propose a system that intakes reviews from registered users for a single product or multiple and evaluate them as a positive or negative rating. This can also be useful to determine the fraud application and ensure mobile security well. This influences in detecting local anomaly than the global anomaly of the app ranking. Furthermore, we inspect through two types of evidence namely rating based, and review based by modelling the consolidation of the two through statistical hypotheses tests.

* 1. **MOTIVATION FOR RESEARCH**

[According to a recent Avast survey in 2018](about:blank) mobile apps are Android or iOS applications that mimic the look and/or functionality of legitimate applications to trick unsuspecting users to install them. Once downloaded and installed, the applications perform a variety of malicious actions. Some fake applications are built to aggressively display advertisements to rake in ad revenue, other apps are designed to harvest credentials, intercept sensitive data, divert revenue or infect devices. More than half of users cannot distinguish between real and fake apps

For years, increasingly crafty developers have been sneaking “fake apps” into the App Store and the Google Play store. These apps piggyback on either the names, designs or functions of more popular apps in an attempt to trick users into downloading them.

Some of these apps exist just to make a quick buck, but others contain malware or attempt to access permissions to steal personal information.

To protect the data and for users privacy, it is necessary that the user finds about the application before downloading it. hence it is necessary to develop a system which will provide you details about the application depending upon the past user experiences.

* 1. **PROBLEM STATEMENT AND OBJECTIVES**

Fraud applications are published to damage or harm the user device and to acquire the user data without his/her knowledge. As the fake applications have increased it is difficult to recognise the genuine applications hence our objective is to check if the application is genuinely based on the historical records by the users.

Challenges in sentiment analysis

* The comments given by the user for a product is considered positive in one situation and negative in another situation.
* Some people don’t express opinions in the same way. Most reviews will have both positive and negative comments, which somewhat manageable by analyzing sentences one at a time.
* Sometimes people may give fake comments about the product which may give the wrong result.

**1.4 REPORT ORGANIZATION**

The introduction section provides an overall picture of what has been completed to date and a brief introduction about each chapter.

**Chapter 1: INTRODUCTION**

This section focuses on the introduction, motivation, problem statement and objective of our project ‘approaches of fake application detection’. In this section, we briefly explain why fake applications are spread and how they can cause major problems to the user.

**Chapter 2: LITERATURE REVIEW**

This section focuses on the already existing solutions that are used to predict or detect the fake application. This section consists of a summary of the research papers which we have studied and come up with our own solution for detecting fake application. We are also able to identify the advantages and disadvantages of the technologies used in the existing solution.

**Chapter 3: PROPOSED METHODOLOGY**

This section consists of a detailed description of the way by which data is collected in the

project. The detailed explanation of all the preprocessing techniques that are performed on data is also described in this section. The Methodology that is used in the project is also summarized in the project.

**Chapter 4: CONCLUSION**

This section deals with the completion of the module and the future scope of the project.

**CHAPTER 2**

**LITERATURE REVIEW**

**2.1 INTRODUCTION**

The sentiment is an emotion or attitude prompted by the feelings of the customer. Sentiment analysis is also referred to as opinion mining, as opinions are collected from customer is mined to reveal the rating of the app. The process of Sentiment analysis comes under machine learning. [1]. Information is gathered and is analyzed to determine the sentiment about the information such as negative or positive sentiment. Before purchasing the app people always enquire about the opinion of the app by the other users [2]. The process of Sentiment analysis uses natural language processing (NLP) to collect and examine the opinion or sentiment of the sentence. It is popular as many people prefer to take some advice from the users. As the amount of opinions in the form of reviews, blogs, etc. are increasing continuously, it is beyond the control of manual techniques to analyze a huge amount of reviews and to aggregate them to an efficient decision. Sentiment analysis performs these tasks into automated processes with less user support [3]. It is not always possible to have one technique to fit in all solution because different types of sentences express sentiments/opinions in different ways. Sentiment words (also called as opinion words) (e.g., great, beautiful, bad, etc) cannot distinguish an opinion sentence from a non-opinion one. A conditional sentence may contain many sentiment words or sentences but express no opinion. The type of sentences, i.e., conditional sentences, it has some unique characteristics which make it hard to determine the orientation of sentiments on topics/features in such of the sentences. By sentiment orientation, we mean positive, negative or neutral opinions. Conditional sentences are sentences which describe implications or hypothetical situations and their consequences. In the English language, a variety of conditional connectives can be used to form these sentences. A conditional sentence contains two clauses: the condition clause and the consequent clause, that are dependent on each other. Their relationship has significant implications on whether the sentence describes an opinion [4]. As there are more than millions of apps on the App store, there exists competition between apps to be on top of the leader board on the basis of popularity. As leader board is the most important way of promoting apps. The higher rank on the leader board leads to a huge number of downloads & million dollars of profit. Apps give advertisement to promote their apps on the leader board. Many apps use fraudulent means to boost their ranking on the leader board of the App store. There are various means to increase downloads and ranking of the app which is done by "bot farms" or "human water armies", human water armies are a group of internet ghostwriters who are paid to post fake reviews. The app is said to fraud on the basis of 3 parameters: Ranking, Rating and Review of the app. In ranking based we check the historical ranking of the app, there are 3 different ranking phases, rising phase, maintaining phases and recession phase. The apps ranking rising to peak position on leader board (ie. rising phase), to keep at the peak position on the leader board (ie. maintaining phase), and finally decreasing till the end of the event (ie. recession phase). The reviews are taken from the dataset and are converted into tokens on which sentiment analysis is performed.

**2.2 REVIEW OF EXISTING RESEARCH/WORK**

## A website is created and data is collected from it i.e. Star rating and Textual format for the process of Data pre-processing techniques like part of speech (POS) tagging, negation phase identification algorithm, Feature extraction, K-means cluster algorithm is used and performance can be evaluated. The semantic orientation of a phrase is calculated as the mutual information between the given phrase and word excellent minus the mutual information between the given phrase and the word poor, then polarity is assigned to each the feature using support vector machines (SVM) and sequential minimal optimization. Sentiment analysis using k means and naïve bayes algorithm that saves running time and reduces computational complexity. [1]

Some movie reviews are taken and data is divided into training and testing data. In training data, feature selection is performed using MI, IG, TF-idf, chi-square and both the training and testing data is classified using a vector machine. they have also focused on unigram and bigram technique to extract sentiments as a result unigram is the best method to extract sentiment. We found that unigram is the best method to extract sentiment from the review. Specifically, it is clear that unigram with stemming with stop word and unigram with stemming without stop word gives the accuracy of 82.9% and 83% in positive class. In negative class unigram with stemming and without stop word gives better accuracy of 83.1%. Both classes gives better result with information gain. As future work, we can suggest that ensemble feature selection technique, it would be useful to perform additional experiment on this work. [2]

There are various natural language processing challenges such as at document level, sentence level, feature level and lexicon level. they have also compared different techniques and approaches to solve the natural language processing challenges such as naïve bayes, k-nearest neighbour, centroid, support vector machine, lexicon based, statistical based. Opinion mining has its boundaries extended from computer science to management sciences. Sentiment analysis, though recently introduced as in research focus for commercial and social content. A detailed analysis of the problem through ML based techniques has made it clear that SA and NLP has many open issues that are beyond the control of the methods in practice. Having close relevance to NLP, sentiment analysis faces NLP issues like co-reference resolution, negation handling, and word sense disambiguation etc, which add more difficulties due to their variation. However, it is also useful to realize that sentiment analysis is a highly restricted NLP problem because the system does not need to fully understand the semantics of each and every word. Complex network analysis has been popularly used for various problems and can produce useful patterns in subjective text. Knowledge-bases systems incorporate domain specific guidance from a knowledge source to improve results in specialized domains. [3]

Conditional sentences have some unique characteristics that make it hard to determine the orientation of sentiments in such sentences. Consequent Whole-sentence-based classification: classifier and Condition classifier is used to perform sentiment analysis accurately, we argue that a divide-and-conquer approach is needed, i.e., focused study on each type of sentences. It is unlikely that there is a one-size-fit-all solution. This paper studied one type, i.e., conditional sentences, which have some unique characteristics that need special handling. Our study was carried out from both the linguistic and computational perspectives. In the linguistic study, we focused on canonical tense patterns, which have been showed useful in classification. In the computational study, we built SVM models to automatically predict whether opinions on topics are positive, negative or neutral. Experimental results have shown the effectiveness of the models. [4]

Data can be collected from user forums: Cell phone, Automobile, LCD TV, Audio systems and Medicine. Clustering based approach is a new way of detecting sentence polarity it helps to categorized similar words together without human interference. With the help of clustering, it combines similar type of data. Accuracy is highest in supervised learning approach, acceptable in case of clustering approach and low in symbolic approach. Efficiency from the time point of view is very fast in symbolic approach, and for supervised learning approach, it is very slow on the training data and fast on the test data while the cluster approach gives fast on the data. Symbolic approach and clustering approach are not required human participation at all but a supervised learning approach does., we propose a methodology to evaluate the security of Android mobile apps based on the cloud computing platform. We also implement a prototype system, i.e., MobSafe, for automation forensic analysis of mobile apps’ static code and dynamical behaviour. Based on the real trace from AppChina, a mobile app market, we can estimate that the number of active Android apps and the average number apps installed in one Android device, and the increasing ratio of mobile apps. We adopt ASEF and SAAF, the two representative dynamic analysis method and static analysis method, to evaluate the Android apps and estimate the total time needed to evaluate all the apps stored in a mobile app market. As the mobile app market serves as the main line of defence against mobile malware, it is practical to use cloud computing platform to defence malware in mobile app markets. [5]

Apk file of mobile application can be uploaded on the web application. APK parser is used to extract information about the application such as reviews, ratings and historical record. Natural Language Processing is used to perform sentiment analysis on the reviews. By applying the rule for detection of fraud application, it generates the graph results. If the rating count is greater than 3 then it is considered as a positive result. And if the rating count is less than 3 then it is considered a negative result. Methodologies used are cloud stack, data mining and NLP. [6]

Application reviews can be extracted and converted into tokens. Tokenization is process of converting a stream of text into words, phrases, symbols known as tokens. These tokens are the input for preprocessing. After pre-processing of reviews system determine the user emotions. Positive reviews add 1 to positive score and negative review adds 1 to negative score. With this, it we will determine score of every review and confirm whether the application is real or fake [7]. Lexicon based approach and machine learning approach can be compared. Lexicon based approach deals with searching the sentiment words form the sentence and comparing with existing list of words, it has two branches dictionary and corpus-based approach. Lexicon–based approach does not require training set whereas Naïve Bayes requires training set. Lexicon–based method is accurate than Naïve Bayes classifier when sentence is processed completely with training set data It is a task of identifying the orientation of opinion or sentiment words in a text. Sentiment analysis can be of three-level document level (such as blog), sentence-level (such as comments) and word level. In this paper, we compare the two methods of sentiment analysis lexicon-based approach and machine learning approach [8]

User reviews can be collected using open source scrapping tools and stored in MySQL database. Titles and comments are extracted from stored dataset. Collocation finding algorithm provided by NLTK toolkit is used for extraction of features from user reviews. User sentiments are extracted about the identified features and given them a general score across all reviews. Finally, topic modelling techniques are used to group fine-grained features into more meaningful high-level features The approach produces two summaries with different granularity levels. These summaries can help app analysts and developers to analyze and quantify users’ opinions about the single app features and to use this information e.g., for identifying new requirements or planning future releases. We generate the summaries by combining a collocation finding algorithm, lexical sentiment analysis, and topic modelling. We obtained a precision up to 91% (59% average) and a recall up to 73% (51% average). The results show that the generated summaries are coherent and contain the most mentioned features in the reviews. [9]

The Tweets Sentiment Analysis Model analyses tweets data. It can identify positive, negative or neutral sentiments and measure intensity of positive/negative opinions in regard to any category. The framework of the TSAM consists of three modules: Feature selection module that extracts the relevant words from each sentence Sentiment identification module that associates expressed opinions with each relevant entity in each sentence level. Sentiment aggregation and scoring module calculates the sentiment scores for each entity. [10]

Google API calculation approach is used to calculate the rank of the applications using calculation algorithm where they take application ratings from play store and calculate the ranks using the calculations. [11]

Feature extraction in sentiment analysis is an emergent research field so in this paper, we have concentrated on related work performed to identify directions for future work. There are many feature selection techniques, NLP based, Machine learning or clustering-based, Statistical, Hybrid, are discussed. Features are categorized as syntactic, semantic, lexico-structural, implicit, explicit and frequent, making it easy for the future researchers to work on. Different pre-processing modules like POS tagging, stop word removal, stemming and lemmatization are discussed. Finally, we conclude that feature space reduction, redundancy removal and evaluating performance of hybrid methods of feature selection can be the future direction of research work for all researchers in the field of feature extraction in sentiment analysis. [14]

Developers have developed a ranking fraud detection system for mobile Apps. Specifically, we show that ranking fraud happened in the leading sessions and provided a method for mining leading sessions for each App from its historical ranking records. Then, we identify ranking based evidences and rating based evidences for detecting ranking fraud. Moreover, we proposed an optimization-based aggregation method to integrate all the evidences for evaluating the credibility of leading sessions from mobile Apps. A unique perspective of this approach is that all the evidences can be modelled by statistical hypothesis tests, thus it is easy to be extended with other evidences from domain knowledge to detect ranking fraud. Finally, we validate the proposed system with extensive experiments on the real-world App data collected from the App store. Experimental results showed the effectiveness of the proposed approach. [15]

The main objective is fraud application detection using fuzzy logic to differentiate the actual fraud applications. The proposed system performs classification of applications and detects their group whether they belong to good, bad, neutral, very good, very bad. Different class value and threshold value gives different results of accuracy of time required for execution. [16]

Sentiment Analysis is major task of NLP (natural language processing) [13]. Data used as input are online app reviews. The objective content from the sentences are removed and subjective content is extracted. The subjective content consists of sentiment sentences. In NLP, part-of-speech (POS) taggers are developed to classify words based on POS. Adjective and verbs convey opposite sentiment with the help of negative prefixes. Sentiment score is computed for all sentiment tokens. [17]

Table 2.2.1 Comparison table for different methodology

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RefNo | Category | Methodology | Data Set | Advantages | Disadvantage | Efficiency |
| 1 | Online product review | Machine learning,  NLP,  vector quantization | Online shopping website | Helps in finding accurate review of product | Most reviews have both positive and negative situation | 89% |
| 2 | Features on sentiment analysis | Support vector machine | Movie review | Unigram is the best method to extract sentiment | Bigram with streaming of stop words gives less accuracy | 82% |
| 3 | Sentiment analysis on complex natural language | NLP,  Sentiment analysis,  Machine learning | Web 2.0 | Naïve Bayes: assumes feature independence. Centroid evaluation is sensitive to noise. K\_nearest neighbour: sensitive to irrelevant features. Support vector machine: require more resources. Lexicon based: struggle with domain context | naïve Bayes: simple and fast. Centroid evaluation classify on vector distance K\_nearest neighbour: handle co-related features. Support vector machine: classify on hyperplane. Lexicon based: can identify new lexicon | - |
| 4 | Conditional Sentences in sentiment analysis | Part of speech tagging | User forums: Cell phone, Automobile, LCD TV, Audio systems and Medicine. | Canonical tense patterns- have been showed useful in classification. | Special conditional sentences that do not use easily recognizable conditional connectives are few but difficult to recognize them | - |
| 5 | Clustering-based sentiment analysis of reviews | Support vector machine | Google Play Store | Well performed, efficient and non-human participating approach on solving sentiment analysis problem. Produce accurate cluster results in short time. | Clustering results are unstable due to random selection of centroids in k-means. The size of the document set might influence the outcome. | - |
| 6 | Cloud Stack And Data Mining | Natural Language Processing,  Data Mining,  Cloud Stack,  Sentiment Analysis, Part Of Speech tagging | Apple Store  Google Play Store | Able to compare two applications | Most reviews have both positive and negative situation | - |
| 7 | Natural language processing based app reviews | Natural Language Processing, Data Mining, Sentiment Classification | Blogs  Movie Review | Detect Fraud using rating, review and ranking evidences | APK file is needed | - |
| 8 | Lexicon based and Naïve Bayes Classifier in Sentiment Analysis | Supervised machine learning  Approach,  Corpus-Based  Approach | Apple Store  Google Play Store | Lexicon Based is more accurate | Ineffective to find fraud in leading sessions | Lexicon bases = 92%  Naive Bayes=  86% |
| 9 | Sentiment Analysis Of App Review | Natural Language Processing,  Sentiment Analysis, Lexical | Twitter | Precision results were higher for short reviews | Naive Bayes is less accurate compared to lexicon-based | Precision=91%  Recall=73% |
| 10 | Sentiment Analysis On Tweets For Social Events | Sentiment Analysis,  NLP, Feature Selection and Extraction | Google Play Store | It is feasible and accurate | Non frequently mentioned features are not detected  Lexical sentiment analysis has limited handling of negation and no handling of past tense and sarcasm | - |
| 11 | Google API rating system | Google API calculation approach | Google Play Store | Finds accurate ranking if sufficient and updated data provided | POS Processing not used | - |
| 12 | Comparing sentences and relation data mining | Data mining, Part of speech tagging | Google Play Store | Helps to calculate polarity of comparative sentence | Due to cumulative nature of the current store rating, it is difficult to climb up from an initial poor rating after huge number of raters have rated apps | - |
| 13 | Ranking | Data Mining, POS tagging, machine learning | Google Play Store | Evidences can be modelled by statistical Hypothesis test, Easy to extend with other evidences from domain knowledge | After extorting three times of evidences, there is no step to merge them | - |
| 14 | Feature Extraction in Sentiment Analysis | Data mining, Sentiment Analysis, Feature Extraction,  POS tagging | Google Play Store & Apple App Store | Many feature selection techniques are discussed | Many issues are faced in feature extraction | - |
| 15 | Ranking | Data Mining, Natural Language Processing, Sentiment Analysis | Google Play Store | Detect Fraud Ranking, Anomaly Detection | Ineffective to find fraud in leading sessions | - |
| 16 | Fuzzy Logic on reviews | Fuzzy Logic | App store | Classification of Apps & detect their group(good, bad, neutral) | Ranking parameter not used | 93.75% |
| 17 | Product review | Sentiment analysis or opinion mining, POS tagging, | User reviews from app store | Tackles problems on Sentiment analysis, sentiment polarity categorization | Not work well for reviews that purely contains implicit sentiment | - |
| 21 | Information Extraction using User Review. | Filtering, Content Classification, Sentiment Analysis, support vector machine | Google Play Store | SVM work better than other classification algorithm comparing with Naïve Bayes and Logistic Regression, SVM achieved best F-measure using unigram feature | Only used existing methods for filtering, classification, sentiment analysis and topic modelling | 83.5% |

We have compared methods based on four different classification methods

1.Multinomial naïve bayes

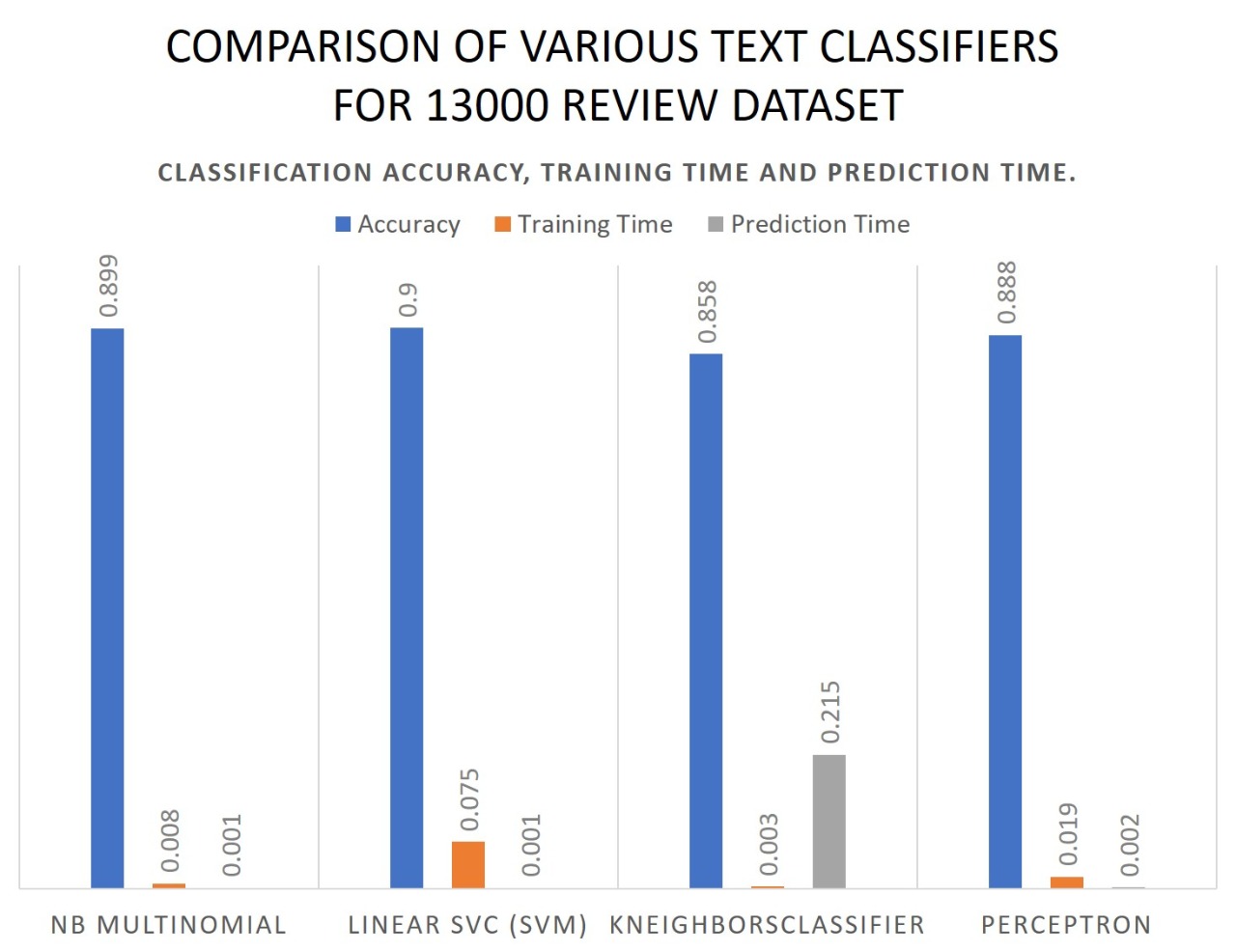
2.linear SVM

3.K-neighbor

4.Perception

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr no | Method | Accuracy | Training time | Prediction time |
| 1 | Multinomial naïve bayes | 89.9% | 0.008 sec | 0.001 sec |
| 2 | linear SVM | 90% | 0.075 sec | 0.001 sec |
| 3 | K-neighbor | 85.8% | 0.003 sec | 0.003 sec |
| 4 | Perception | 88.8% | 0.019 sec | 0.002 sec |

**Table 2.2.2 : Comparison table**



**Fig2.2.1: Comparison of text classifiers**

As seen in above table multinomial naïve bayes has a accuracy of 89.9% and less computing time hence it is more preferred over other methods. Our fraud application detection method uses it as classification method.

**2.3 DOMAIN SPECIFIC LITERATURE**

**2.3.1 Machine Learning**

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

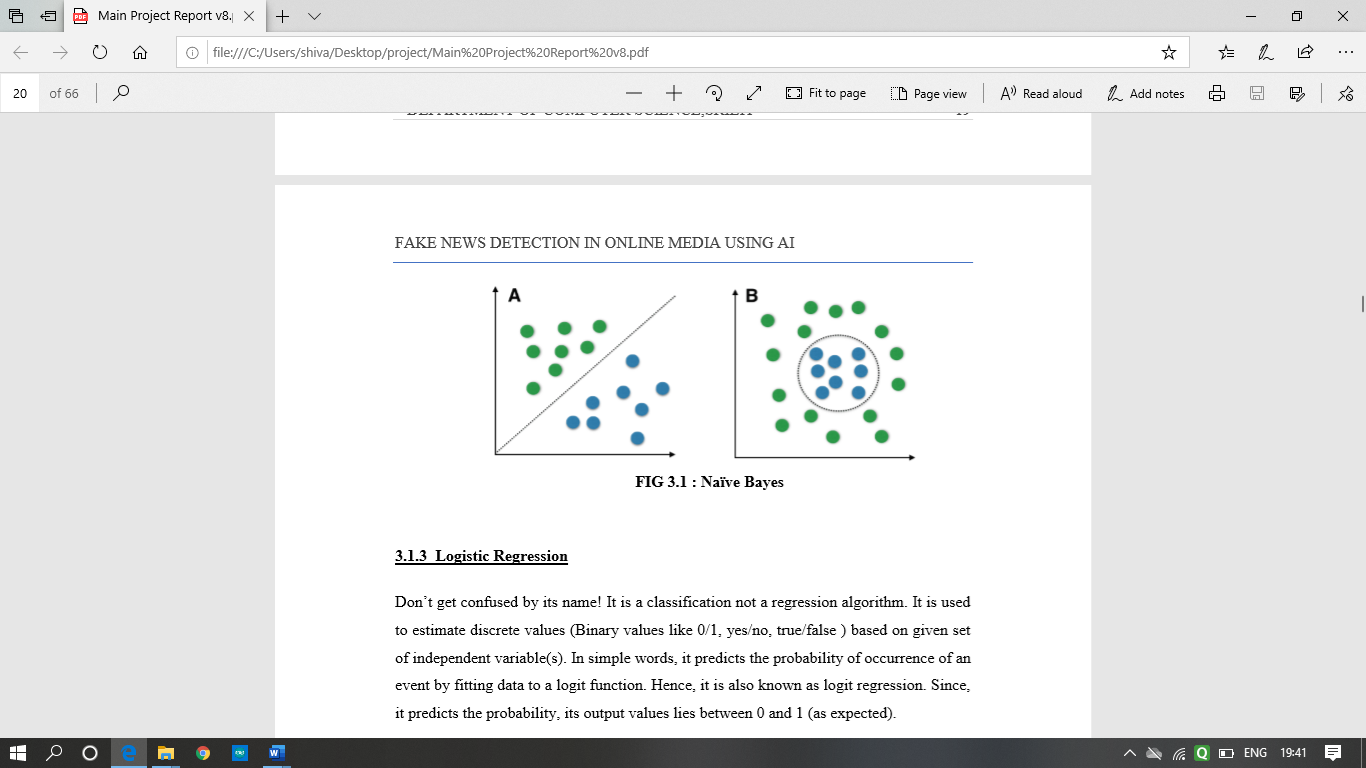
Machine learning algorithms are often categorized as supervised or unsupervised.

* Supervised machine learning algorithms can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.
* In contrast, unsupervised machine learning algorithms are used when the information used to train is neither classified nor labelled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.

**2.3.2. Naive Bayes Classifiers**

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes’ Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the colour, roundness, and diameter features.

For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods. Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficiency of naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests. An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.



**Fig2.3.2.1. Naïve Bayes Classifier**

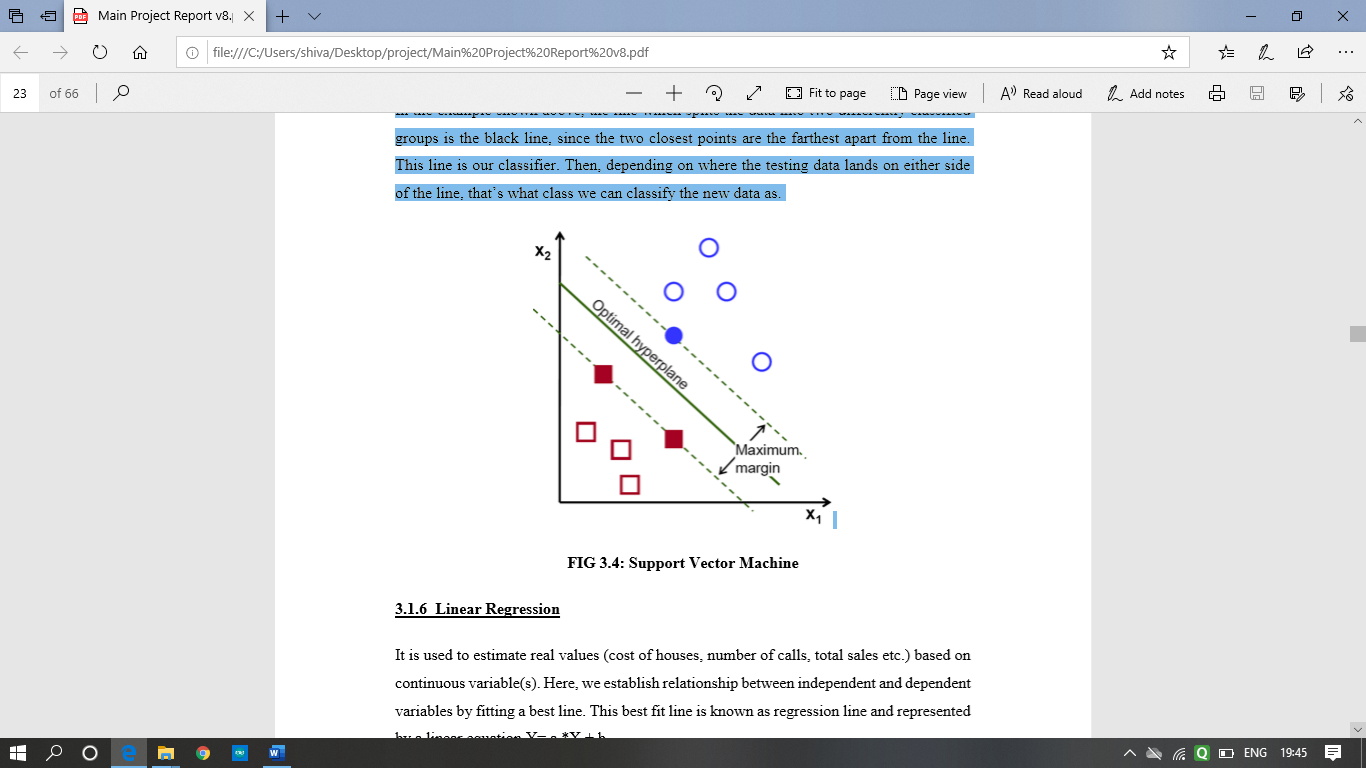
**2.3.3. SVM (Support Vector Machine)**

It is a classification method. In this algorithm, we plot each data item as a point in ndimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.

For example, if we only had two features like Height and Hair length of an individual, we’d first plot these two variables in two dimensional space where each point has two coordinates (these co-ordinates are known as Support Vectors)

Now, we will find some line that splits the data between the two differently classified groups of data. This will be the line such that the distances from the closest point in each of the two groups will be farthest away.

In the example shown above, the line which splits the data into two differently classified groups is the black line, since the two closest points are the farthest apart from the line. This line is our classifier. Then, depending on where the testing data lands on either side of the line, that’s what class we can classify the new data as.



**Fig 2.3.3.1 Support Vector Machine**

**2.3.4. POS Tagging**

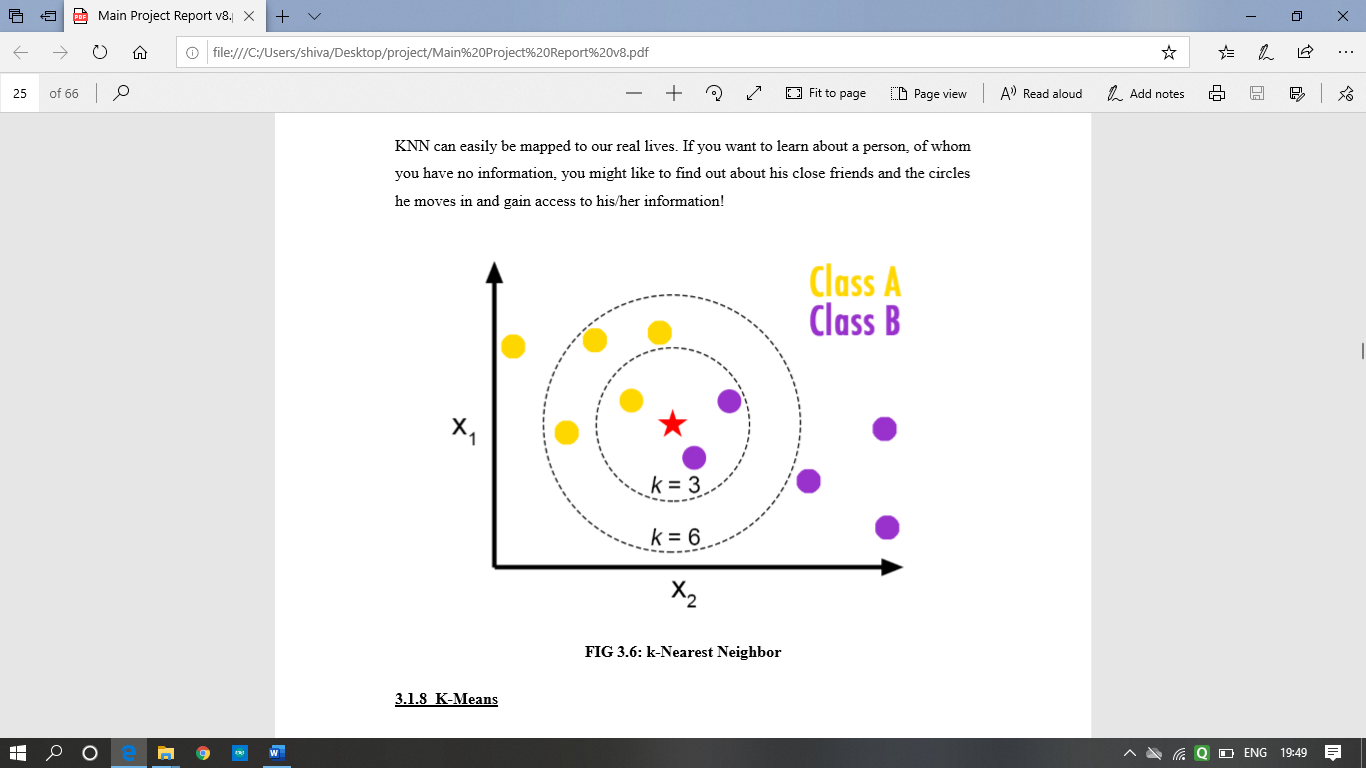
**POS tagging** is the process of marking up a word in a corpus to a corresponding part of a speech **tag**, based on its context and definition. This task is not straightforward, as a particular word may have a different **part of speech** based on the context in which the word is used. POS tagging is now done in the context of computational linguistics, using algorithms which associate discrete terms, as well as hidden parts of speech, in accordance with a set of descriptive tags. POS-tagging algorithms fall into two distinctive groups: rule-based and stochastic. E. Brill's tagger, one of the first and most widely used English POS-taggers, employs rule-based algorithms.

**2.3.5. *k*-NEAREST NEIGHBOURS ALGORITHM (*k*-NN)**

In pattern recognition, the ***k*-nearest neighbors algorithm** (***k*-NN**) is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.

In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors. *k*-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor. The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.



**Fig 2.3.5.1. k-Nearest Neighbours Algorithm**

# **2.3.6. NLP (Natural Language Processing)**

Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data. Challenges in natural language processing frequently involve speech recognition, natural language understanding, and natural language generation. Many different classes of machine-learning algorithms have been applied to natural-language-processing tasks. These algorithms take as input a large set of "features" that are generated from the input data. Some of the earliest-used algorithms, such as decision trees, produced systems of hard if-then rules similar to the systems of handwritten rules that were then common. Increasingly, however, research has focused on statistical models, which make soft, probabilistic decisions based on attaching real-valued weights to each input feature. Such models have the advantage that they can express the relative certainty of many different possible answers rather than only one, producing more reliable results when such a model is included as a component of a larger system**.**

Systems based on machine-learning algorithms have many advantages over hand-produced rules:

* The learning procedures used during machine learning automatically focus on the most common cases, whereas when writing rules by hand it is often not at all obvious where the effort should be directed.
* Automatic learning procedures can make use of statistical inference algorithms to produce models that are robust to unfamiliar input (e.g. containing words or structures that have not been seen before) and to erroneous input (e.g. with misspelt words or words accidentally omitted). Generally, handling such input gracefully with handwritten rules, or, more generally, creating systems of handwritten rules that make soft decisions, is extremely difficult, error-prone and time-consuming.
* Systems based on automatically learning the rules can be made more accurate simply by supplying more input data. However, systems based on handwritten rules can only be made more accurate by increasing the complexity of the rules, which is a much more difficult task. In particular, there is a limit to the complexity of systems based on handcrafted rules, beyond which the systems become more and more unmanageable. However, creating more data to input to machine-learning systems simply requires a corresponding increase in the number of man-hours worked, generally without significant increases in the complexity of the annotation process.

**2.3.7. TOKENIZATION**

**Tokenization**is the process of tokenizing or splitting a string, text into a list of tokens. One can think of token as parts like a word is a token in a sentence, and a sentence is a token in a paragraph. Tokenization is generally considered as easy relative to other tasks in natural language, and one of the more uninteresting tasks (for English and other segmented languages). However, errors made in this phase will propagate into later phases and cause problems. To address this problem, a number of advanced methods which deal with specific challenges in tokenization have been developed to complement standard tokenizers.

**2.3.8.LEXICON BASED APPROACH**

Lexicon based approaches use as knowledge base lexical resources named opinion lexicon, that associates words to their sentiment orientation represented for positive and negative ”scores”. Their use in sentiment analysis research starts from the assumption that single words can be considered as a unit of opinion information, and therefore it can provide indications to detect document sentiment and subjectivity. The annotation can be done either manually or by automatic, semi-supervised, processes that, using linguistic resources like a corpus, a thesaurus, or a more sophisticated one like Wordnet.

**2.3.9.PYTHON**

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released 2000, introduced features like list comprehensions and a garbage collection system capable of collecting reference cycles. Python 3.0, released 2008, was a major revision of the language that is not completely backwards-compatible, and much Python 2 code does not run unmodified on Python 3. Due to concern about the amount of code written for Python 2, support for Python 2.7 (the last release in the 2.x series) was extended to 2020. Language developer Guido van Rossum shouldered sole responsibility for the project until July 2018 but now shares his leadership as a member of a five-person steering council.

Python interpreters are available for many operating systems. A global community of programmers develops and maintains CPython, an open source[32] reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

* **Pandas**

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

* **Numpy**

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with Python.

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.

Array in Numpy is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In Numpy, number of dimensions of the array is called rank of the array. A tuple of integers giving the size of the array along each dimension is known as shape of the array. An array class in Numpy is called as ndarray. Elements in Numpy arrays are accessed by using square brackets and can be initialized by using nested Python Lists.

* **SciPy**

SciPy is a free and open-source Python library used for scientific computing and technical computing.SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools likeMatplotlib, pandas and SymPy, and an expanding set of scientific computing libraries. This NumPystack has similar users to other applications such as Matlab, GNU Octave, and Scilab. The NumPystack is also sometimes referred to as the SciPy stack. SciPy is also a family of conferences for users and developers of these tools: SciPy (in the United States), EuroSciPy (in Europe) and SciPy.in (in India). Enthought originated the SciPy conference in the United States and continues to sponsor many of the international conferences as well as host the SciPy website.

The SciPy library is currently distributed under the BSD license, and its development is sponsored and supported by an open community of developers. It is also supported by NumFOCUS, a community foundation for supporting reproducible and accessible science.

**HTML5**

**HTML5** is a markup language used for structuring and presenting content on the World Wide Web.

HTML5 was the fifth and last major version of HTML that is a World Wide Web Consortium (W3C) recommendation. The current specification is known as the HTML Living Standard and is maintained by a consortium of the major browser vendors (Apple, Google, Mozilla, and Microsoft), the Web Hypertext Application Technology Working Group (WHATWG).

HTML5 was first released in public-facing form on 22 January 2008, with a major update and "W3C Recommendation" status in October 2014. Its goals were to improve the language with support for the latest multimedia and other new features; to keep the language both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc., without XHTML's rigidity; and to remain backward-compatible with older software. HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-platform mobile applications, because it includes features designed with low-powered devices in mind.

**Electron js**

**Electron** (formerly known as **Atom Shell**) is an open-source framework developed and maintained by GitHub. Electron allows for the development of desktop GUI applications using web technologies: It combines the Chromium rendering engine and the Node.js runtime. Electron is the main GUI framework behind several notable open-source projects including Atom, GitHub Desktop, Light Table, Visual Studio Code, and WordPress Desktop.

**Scikit-Learn**

Scikit-learn is a Python module integrating a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems. This package focuses on bringing machine learning to non-specialists using a general-purpose high-level language. Emphasis is put on ease of use, performance, documentation, and API consistency. It has minimal dependencies and is distributed under the simplified BSD license, encouraging its use in both academic and commercial settings

# **2.4 SUMMARY AND CONCLUSION**

There are several approaches used to detect fraud applications using different approaches but techniques using machine learning algorithms have proved to be more reliable. Fuzzy Logic produces good results for detecting fraud but is then left behind by Naïve Bayes and lexicon-based approach which achieves better accuracy in detecting fraud applications. From various datasets used to train the model, it has come forward that the use of versatile and moderate size dataset with more words improves the accuracy and gives a better result. The reviews obtained using Google API may result in false prediction due to less data availability. We will try to solve this problem by combining lexicon-based and Naïve Bayes to improve the accuracy and the use of dataset with proper reviews and ratings.

**CHAPTER 3**

**PROPOSED METHODOLOGY**

**3.1 DATA COLLECTION DETAILS**

Data collection is an important part of Machine Learning. Data collection is the process of gathering and measuring information for different available sources. Machine Learning requires a huge set of data having multiple attributes, to be able to classify some input parameters more accurately. Data collection is the important aspect that makes the algorithm training possible. It has been observed that greater number of attributes yields a better result

Training data for fraud app detection is obtained from

Training data

Training dataset was used for training the algorithm so that algorithm learns and produce results. Training dataset consist of 13000 entries (reviews,sentiment value). Training dataset consists 50% of positive and 50% of negative reviews.

Testing dataset: Testing dataset was used for evaluating the model/algorithm with trained dataset. Testing dataset is real time dataset which is extracted from google playstore.

**3.2 DATA PREPROCESSING**

The process of converting data to something a computer can understand is called Preprocessing. The dataset which is obtained in data collection is not in the form which can be used by the classifier. Various Data preprocessing and feature extraction techniques must be performed on the dataset to make it suitable for generation of classification model. The python library Pandas is used to perform the preprocessing techniques on the dataset.

Preprocessing steps are:

Tokenization

Tokenization basically refers to splitting up a large body of text into smaller lines, words or even creating words for a non-English language. Various tokenization functions are inbuilt into nltk module itself.

Stopwords Removal

Stop Word Removal is a process of filtering out useless data. In NLP, useless words are referred to as stopwords

Lowercase conversion

In this all the upper case letters are converted to lower case

Tfidf Vectorizer

The Tfidf Vectorizer will tokenize documents, learn the vocabulary and inverse document frequency weightings, and allow you to encode new documents

**3.3 PROPOSED RESEARCH DESIGN**

The proposed approach for the system can be carried out by using corpus based and Naïve Bayes based approach to detect fraud application. First the dataset is prepared so that it can be used for the classifier. The dataset is first stored in a data structure dataframe which can be made by the pandas library. By using the tfidfVectorizer function, various features are extracted based on which the classifier is prepared which is then used for detection of fraud applications. For testing data we are using real time user reviews extracted from the Google play store using Google play scraper. Along with the reviews, ratings of the application are also extracted. On this reviews preprocessing is done by using tfidfVectorizer. This input is then given to the naïve bayes classifier which predicts the polarity for each review. The input to the model is the name of the application. The model extracts the reviews and gives it to the classifier for prediction. If the review is positive then score given to it is 1 and if the review is negative than the score given to it is 0. Finally on the basis of sentiments and ratings of the application pie chart and ratings are displayed and recommendation is given to the user.

Algorithm: Basic steps describing the proposed algorithm are as follows:

1. Data Preprocessing

* Tokenization
* Stopwords Removal
* Lowercase conversion
* Feature Extraction

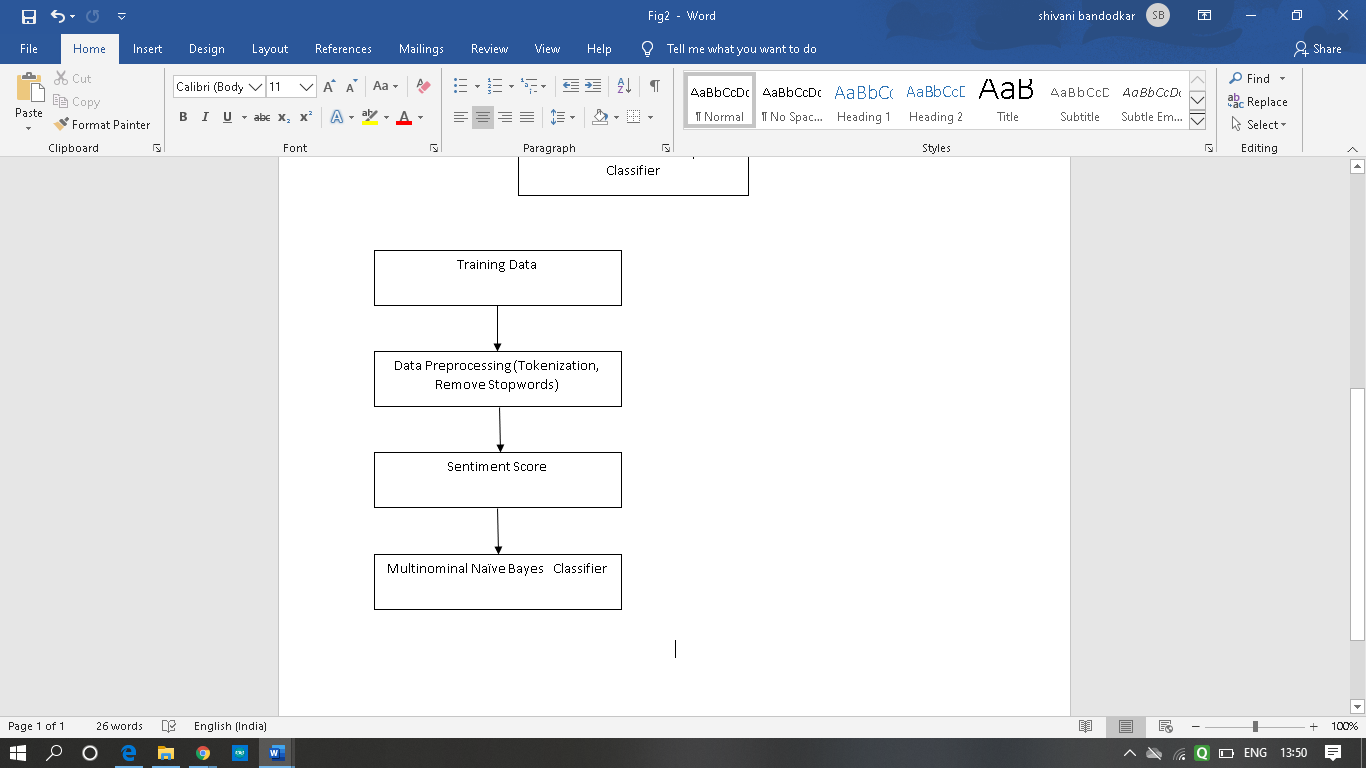
1. Naïve Bayes Classification

* Sentiment Score Generation

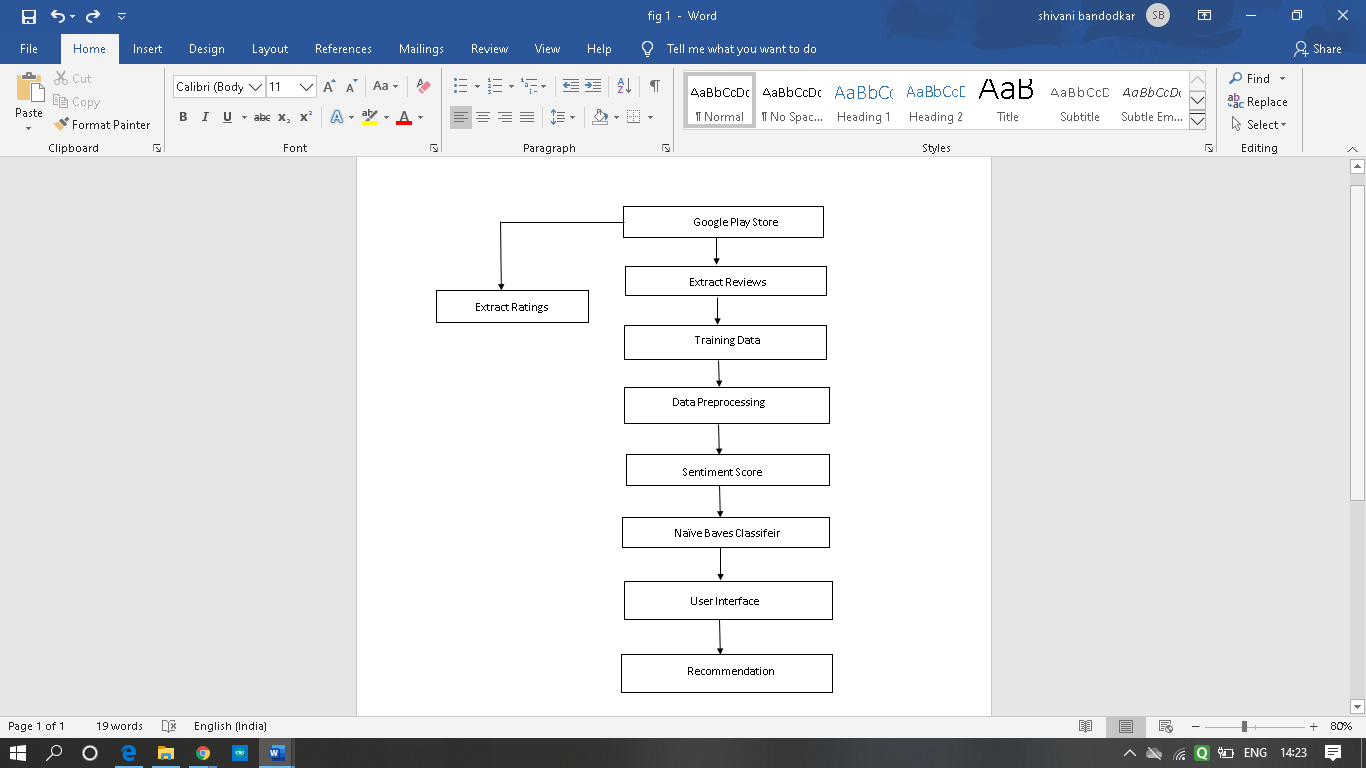
1. Online Review Extraction

* Data preprocessing
* Naïve Bayes Classifier
* Prediction

1. Online Rating Extraction
2. Final Predicted Recommendation



**Fig 3.3.1. Training data**



**Fig 3.3.2. Testing data**

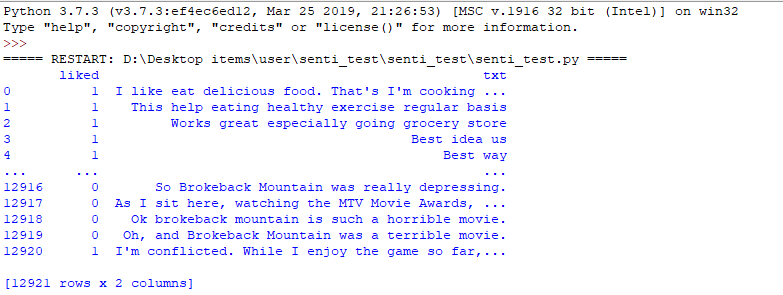
**CHAPTER 4**

**IMPLEMENTATION & RESULT ANALYSIS**

**Data Preparation**

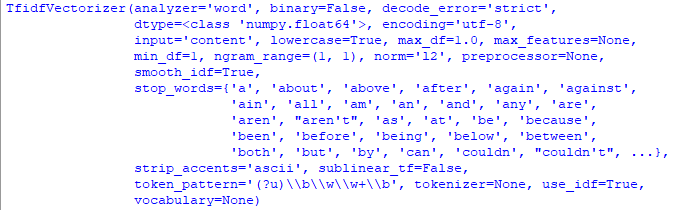
The Libraries used for data preparation are **pandas, CSV, NumPy and NLTK**. **Pandas** is the most popular Python library that is used for data analysis. The **CSV** module implements classes to read and write tabular data in CSV format without knowing the precise details of the CSV format. **Numpy** is a general-purpose array-processing package. It provides a high-performance multidimensional array object and tools for working with these arrays. **NLTK** is used for building Python programs to work with human language data along with text processing libraries for **classification, tokenization, stemming, tagging, parsing and semantic reasoning**. The training and testing dataset are in CSV file format. The values of the dataset are then put into a data frame. A data frame is a two-dimensional (2-D) data structure defined in pandas which consists of rows and columns.

**Training data**



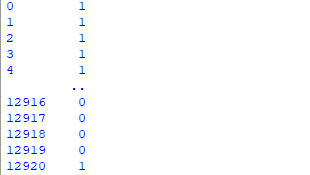
**Img 4.1.** **Training dataset**

We are using a feature from nltk.corpus import stopwords to import the corpus library for stop words. The Tfidf Vectorizer will tokenize documents, learn the vocabulary and inverse document frequency weightings, and allow you to encode new documents.

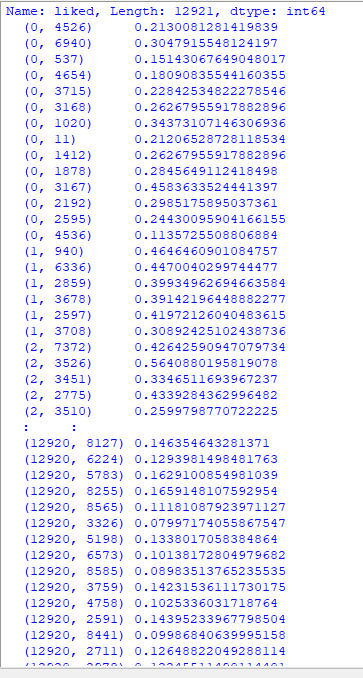


**Img 4.2. Tfid Vectorizer**

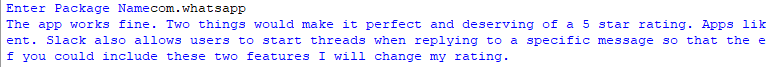
After removing the stop words the reviews are categorized as positive or negative ie. 1 indicates positive and 0 indicates negative.



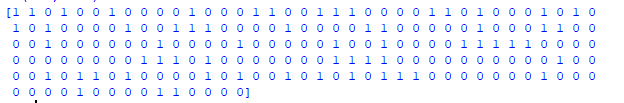
**Img 4.3. Positive or negative**



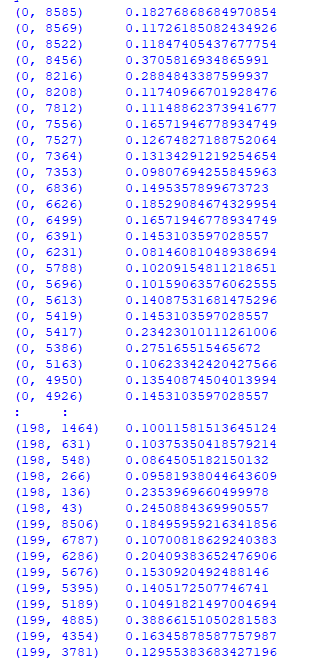
**Img 4.4. Polarity of training data**



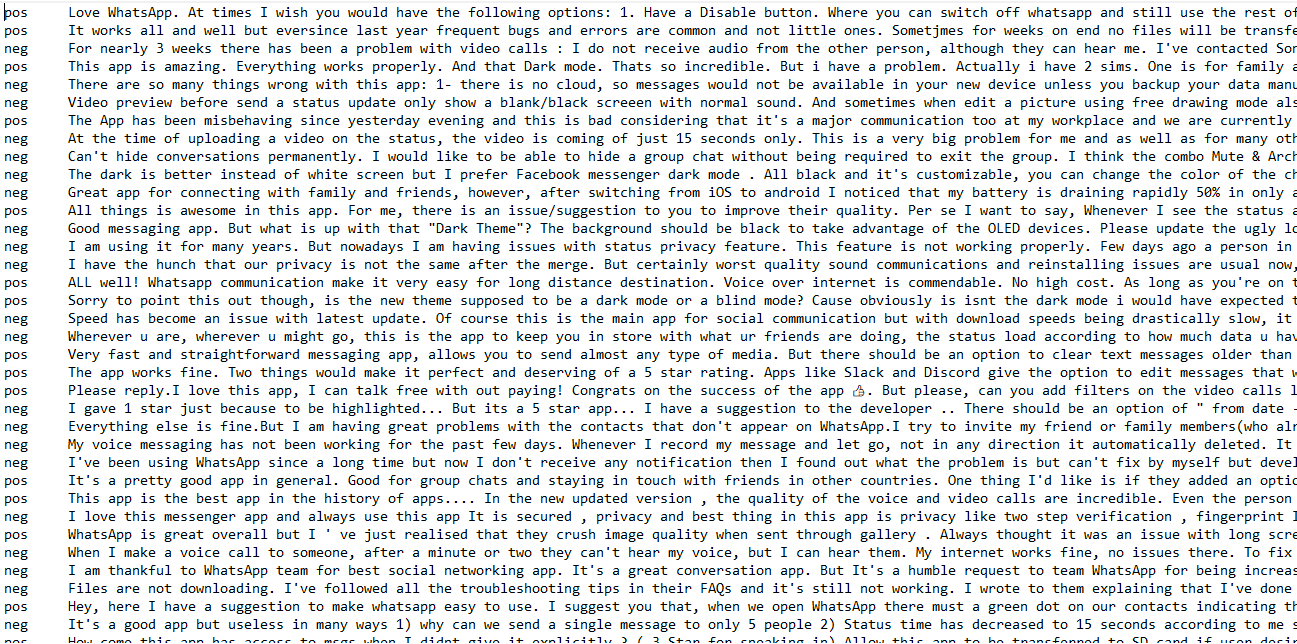
**Img 4.5. Extracting data from google play**



**Img 4.6. Positive or negative of testing data**

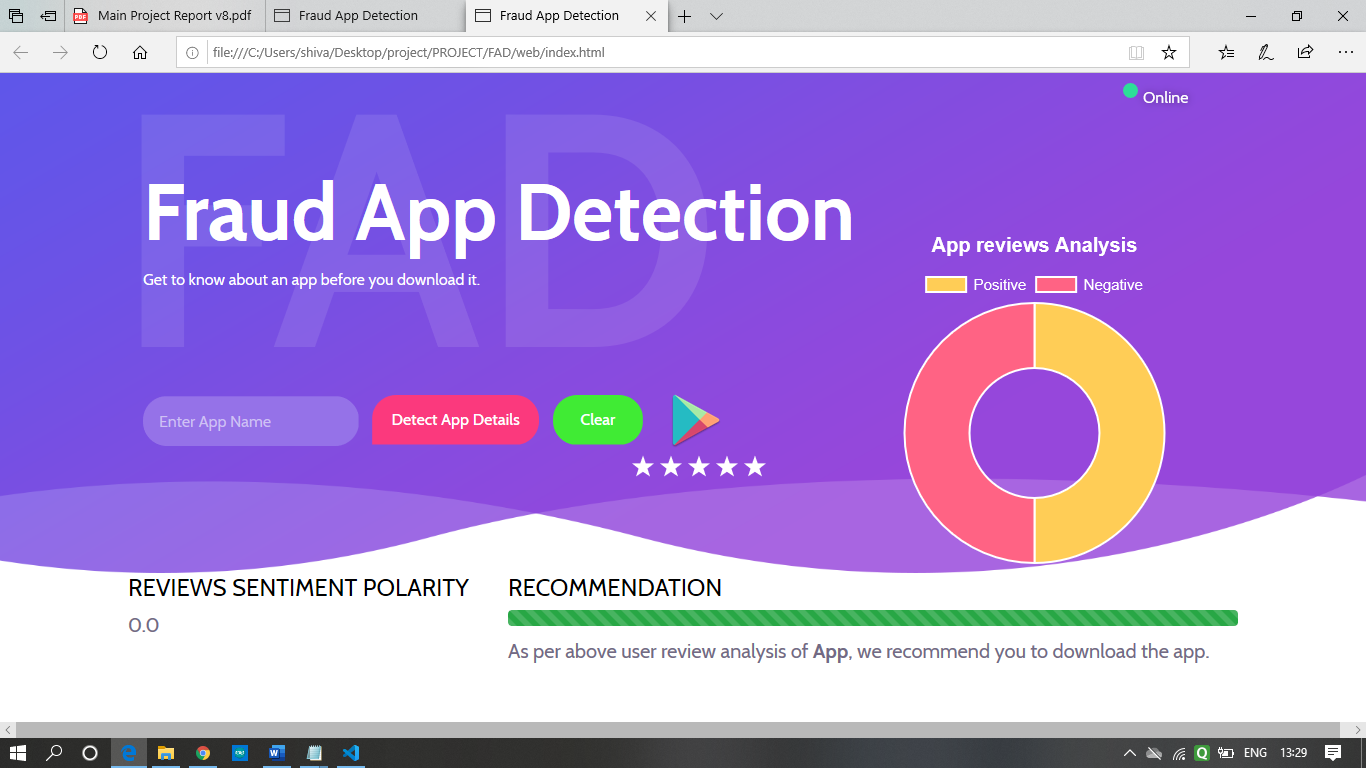


**Img 4.7.Polarity of testing data**

**Img 4.8. Text stored as positive or negative in the data file**

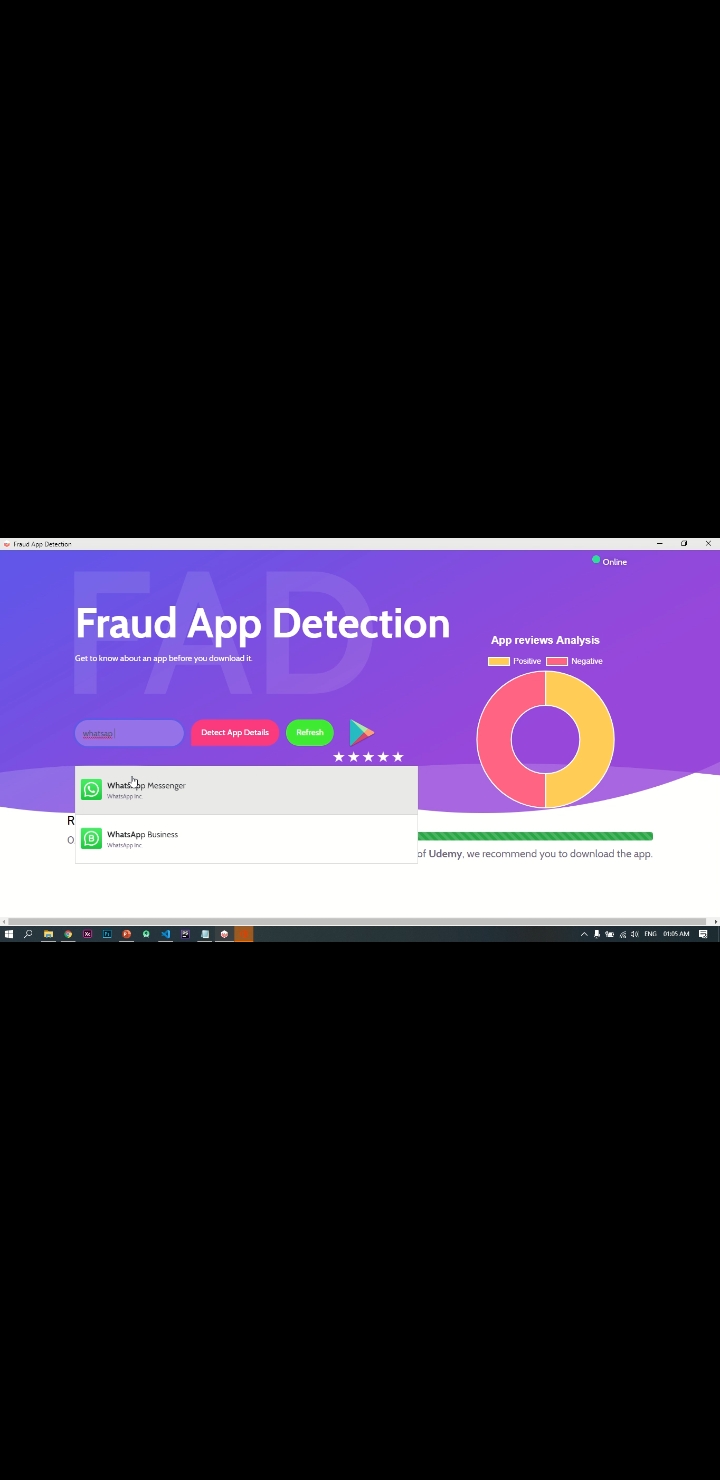
**Creation of HTML Page**

The HTML page ‘**index.html**’ is the UI of the project and is run when the browser extension is clicked. HTML is used to define the content of the page. Cascading Style Sheet(CSS) is used to specify the styling and layout of the webpage.

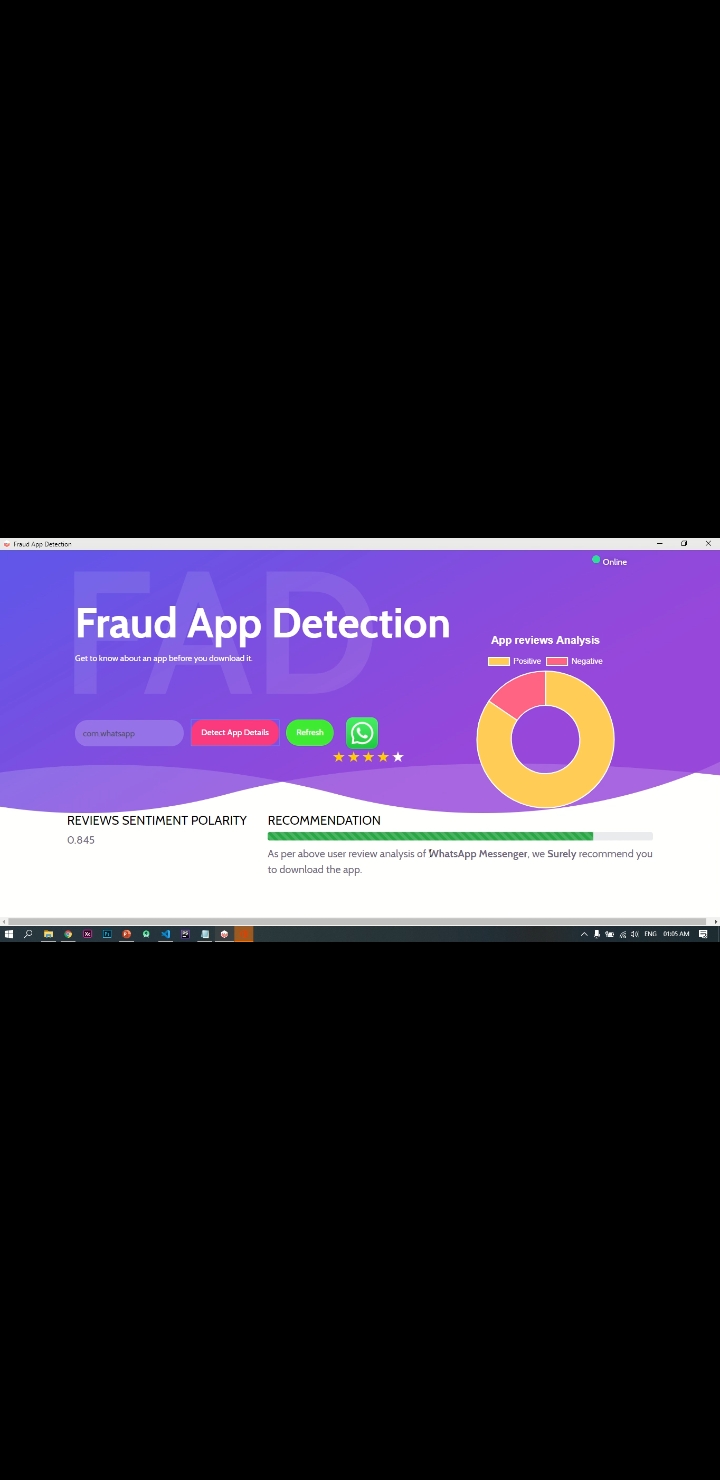


**Img 4.9. The display page**

When the user enters the app name and clicks on the detect app details button on the index.html webpage, It takes the text stored in the textbox and performs prediction function on it. From the playstore, the real-time reviews are collected as explained in the previous part.

****

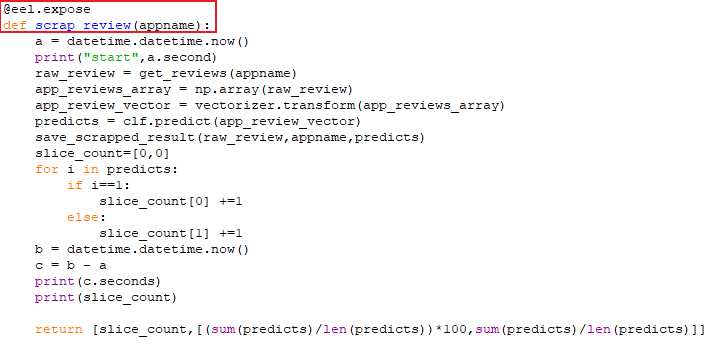
**Img 4.10. Enter the app name**



**Img 4.11. Result**

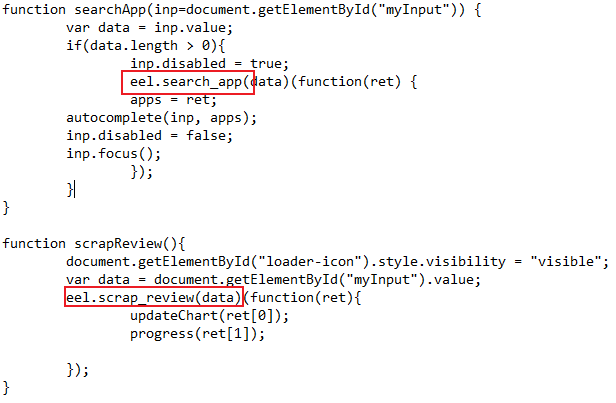
**The connection between the Front end ( HTML, Js ) and Back end ( Python )**

The python library ‘**eel**’ is used to establish the connection between JavaScript and Python. In the python script, ‘**eel.init('web')’** is used to initialise the Front end directory toeel which generates the **‘eel.js’** in the front end directory ‘web’, **@eel.expose** command is used to expose the functions of python to the javascript file ‘**eel.js**’. as shown below.

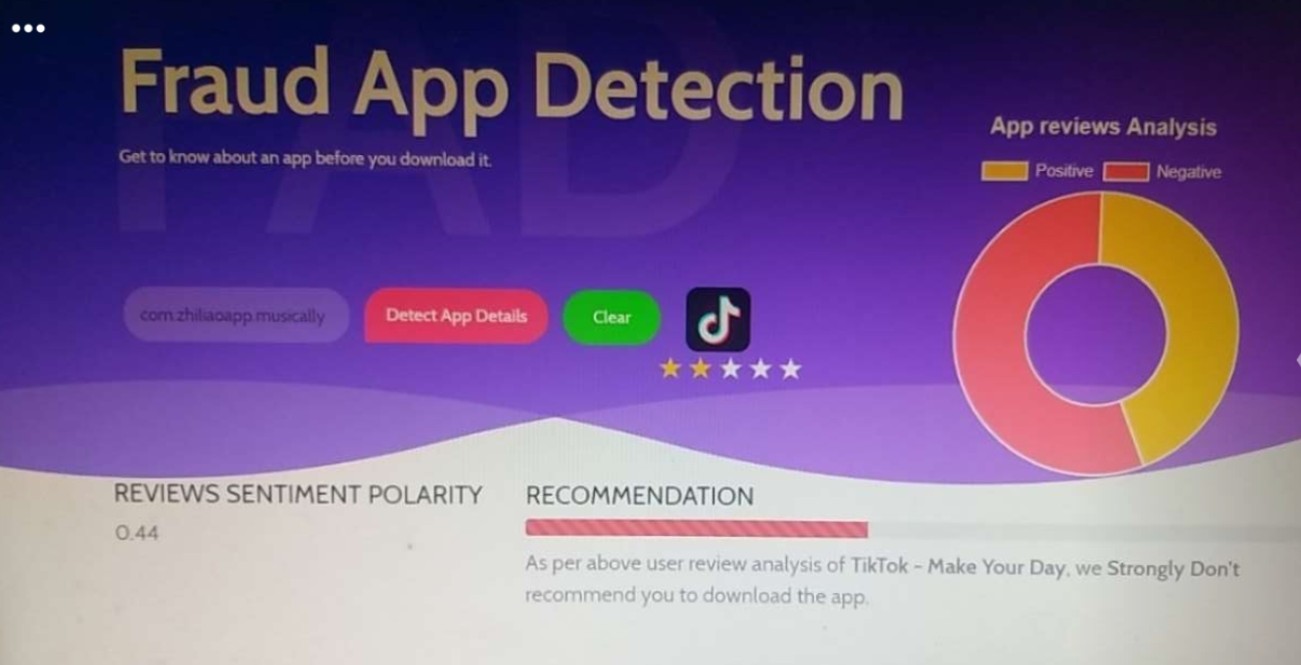


**Img 4.12. Use of @eel.expose**

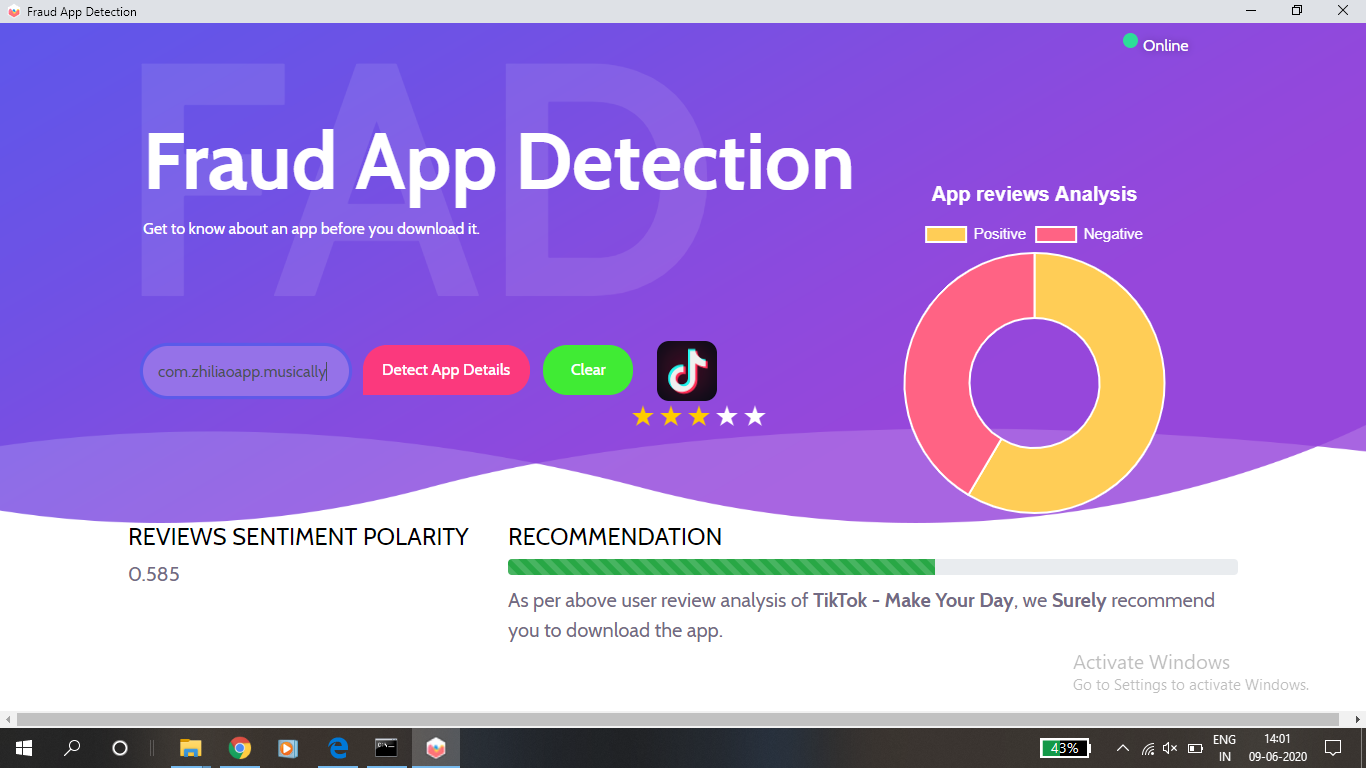
In the ‘**main.js’** file in the front end directory ‘web’ the ‘**searchApp**’uses the python function ‘**search\_app**’ and ‘**scrapReview**’ uses the python function ‘**scrap\_review**’ by using it from eel.js as shown below.



**Img 4.13.** **Using functions of python from eel.js**



**Img 4.14:musically/tiktok on 19th may 2020**



**Img 4.15:musically/tiktok on 9th June 2020**

**CHAPTER 5**

**CONCLUSION**

**5.1 Conclusion**

Machine Learning uses a statistical technique to give the computer the ability to learn with data hence widely used for detection of fraud app. The proposed model introduces the efficient method of detecting fraud apps. In our model, we have used a technique called Sentiment analysis which comes under machine learning. Sentiment analysis is also referred to as opinion mining, as opinions are collected from customer is mined to reveal the nature of the app. The dataset is first pre-processed using preprocessing techniques such as stop word removal, tokenization and stemming. The preprocessing is done using NLTK(Natural Language Toolkit). The TF-IDF technique was used to extract features. This model performs fraud app detection by using Google play API as an input which not only validates app and other related parameters like App rating and reviews. A Django Web framework of Python is used to create a web application that provides an efficient classification of Fraud app. Based on this research, we have observed that machine learning algorithms are much more efficient in detecting fake news than by using comparison with a dataset of fraud app.

**5.2 Future scope**

For the future development, a multiclass of sentiment classification such as positive, negative or neutral and so on can be taken into consideration, which may lead to a better understanding of natural language opinions and will more efficiently bridge the gap between multimodal information and machine-processable data. As presently we are dealing with two classes i.e positive and negative. In future precise classes will be provided for training which may help the classification to predict the sentiments in a better way. We can further improve our model by using other parameters which may help to increase the accuracy of our model.

**REFERENCES**

1. Safrin, Raheesa, K. R. Sharmila, TS Shri Subangi, and E. A. Vimal. "Sentiment analysis on online product review." *Int. Res. J. Eng. Technol* 4, no. 04 (2017).
2. Shahana, P. H., and Bini Omman. "Evaluation of features on sentimental analysis." *Procedia Computer Science* 46 (2015): 1585-1592.
3. “[Muhammad Taimoor Khan](about:blank#auth-1), Mehr Durrani, Armughan Ali, Irum Inayat, Shehzad Khalid & [Kamran Habib Khan](about:blank#auth-6)  “Sentiment analysis and complex natural language” (2016) a springeropen journal
4. Narayanan, Ramanathan, Bing Liu, and Alok Choudhary. "Sentiment analysis of conditional sentences." In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pp. 180-189. Association for Computational Linguistics, 2009.
5. Li, Gang, and Fei Liu. "A clustering-based approach on sentiment analysis." In *2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering*, pp. 331-337. IEEE, 2010.

Publisher: IEEE

1. Xu, Jianlin, Yifan Yu, Zhen Chen, Bin Cao, Wenyu Dong, Yu Guo, and Junwei Cao. "MobSafe: cloud computing-based forensic analysis for massive mobile applications using data mining." *Tsinghua Science and Technology* 18, no. 4 (2013): 418-427.
2. Tichkule, Ashwini, Nidhi Nikhar, Dewanand Kapgate, and Omkar Dudhbure. "Revelation of Fraud Apps Using Sentiment Analysis App Reviews,
3. Rohini, V., and M. Thomas. "Comparison of Lexicon based and Naïve Bayes Classifier in Sentiment Analysis." *International Journal for Scientific Research & Development* 3, no. 4 (2015).
4. Guzman, Emitza, and Walid Maalej. "How do users like this feature? a fine-grained sentiment analysis of app reviews." In *2014 IEEE 22nd international requirements engineering conference (RE)*, pp. 153-162. IEEE, 2014.
5. Zhou, Xujuan, Xiaohui Tao, Jianming Yong, and Zhenyu Yang. "Sentiment analysis on tweets for social events." In *Proceedings of the 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 557-562. IEEE, 2013.
6. Israel J. Mojica Ruiz, Meiyappan Nagappan, Bram Adams, Thorsten Berger, Steffen Dienst, [Ahmed E. Hassan](about:blank) “An examination of the current rating system used in mobile app store” IEEE
7. Jindal, Nitin, and Bing Liu. "Mining comparative sentences and relations." In *AAAI*, vol. 22, no. 13311336, p. 9. 2006.
8. Zhu, Hengshu, Hui Xiong, Yong Ge, and Enhong Chen. "Discovery of ranking fraud for mobile apps." *IEEE Transactions on knowledge and data engineering* 27, no. 1 (2014): 74-87.
9. Asghar, Muhammad Zubair, Aurangzeb Khan, Shakeel Ahmad, and Fazal Masud Kundi. "A review of feature extraction in sentiment analysis." *Journal of Basic and Applied Scientific Research* 4, no. 3 (2014): 181-186.
10. Zhu, Hengshu, Hui Xiong, Yong Ge, and Enhong Chen. "Ranking fraud detection for mobile apps: A holistic view." In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pp. 619-628. ACM, 2013.
11. Neha Puram, Neha M., and K. R. Singh. "An Implementation to Detect Fraud App Using Fuzzy Logic."
12. Fang, Xing, and Justin Zhan. "Sentiment analysis using product review data." *Journal of Big Data* 2, no. 1 (2015):
13. Mudinas, Andrius, Dell Zhang, and Mark Levene. "Combining lexicon and learning-based approaches for concept-level sentiment analysis." In *Proceedings of the first international workshop on issues of sentiment discovery and opinion mining*, p. 5. ACM, 2012.
14. Cambria, Erik, and Bebo White. "Jumping NLP curves: A review of natural language processing research." *IEEE Computational intelligence magazine* 9, no. 2 (2014): 48-57.
15. Qiu, Guang, Bing Liu, Jiajun Bu, and Chun Chen. "Opinion word expansion and target extraction through double propagation." *Computational linguistics* 37, no. 1 (2011): 9-27.
16. Suprayogi, Erry, Indra Budi, and Rahmad Mahendra. "Information Extraction for Mobile Application User Review." In *2018 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pp. 343-348. IEEE, 2018.
17. Rao, Shivani, and Misha Kakkar. "A rating approach based on sentiment analysis." In *2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence*, pp. 557-562. IEEE, 2017.