**Fake Product Review Monitoring System**

1. **INTRODUCTION:**

One of the very rapid growth areas is ecommerce. Generally e-commerce provides facility for customers to write reviews related with its service. The existence of these reviews can be used as a source of information. For examples, companies can use it to make design decisions of their products or services but unfortunately, the importance of the review is misused by certain parties who tried to create fake reviews, both aimed at raising the popularity or to discredit the product. They share their thoughts on internet. Before purchasing anything, it is a normal human behaviour to do a survey on that product. Based on reviews, customers can compare different brands and can finalize a product of their interest. These online reviews can change the opinion of a customer about the product. If these reviews are true, then this can help the users to select proper product that satisfy their requirements. On the other hand, if the reviews are manipulated or not true then this can mislead user. This boosts us to develop a system which detects fake reviews for a product by using the text and rating property from a review. The honesty value and measure of a fake review will be measured by utilizing the data mining techniques. An algorithm could be used to track customer reviews, through mining topics and sentiment orientation from online customer reviews and will also blocked the fake reviews.

**1.1 Objective of the project:**

In the current scenario, the data on the web is growing exponentially. Social media is generating a large amount of data such as reviews, comments, and customer’s opinions on a daily basis. This huge amount of user generated data is worthless unless some mining operations are applied to it. As there are a number of fake reviews so opinion mining technique should incorporate Spam detection to produce a genuine opinion. Nowadays, there are a number of people using social media opinions to create their call on shopping for product or service. Opinion Spam detection is an exhausting and hard problem as there are many faux or fake reviews that have been created by organizations or by the people for various purposes. They write fake reviews to mislead readers or automated detection system by promoting or demoting target products to promote them or to degrade their reputations. The proposed technique includes Ontology, Geo location and IP address tracking, Spam words Dictionary using Naïve Bayes, Brand only review detection and tracking account used**.**

**2. LITERATURE SURVEY:**

**Opinion Mining Using Ontological Spam Detection**

Duhan & Mittal proposed a paper “Opinion Mining Using Ontological Spam Detection” which will help us to find out fake reviews by using Naïve Bayes as algorithm. To find out fake review in the website this “Fake Product Review Monitoring System” system is introduced. This system will find out fake reviews made by the customers and it will block the users. To find out the review is fake or genuine, we will use some classification such as Tracking IP address of the user to detect if the reviews are from a Spammer. If multiple reviews are from the same IP address then the Reviews are considered Spam. Using Account Used to check whether the reviews are done using the same account. Brand only Review detection i.e. whether the reviews are on only Brand not the product. It’s not helpful to consider only the Brand value to judge a product. Using Negative Dictionary i.e. the negative words are identified in the review. If there are more than five Negative Words then the review is a Spam. For instance, a user has posted a Review: “This product is not good; the design is bad, quality is worst and it is worthless to buy.” Here, this sentence consists of 4-5 negative words. So, the system will check the count of negative words, if the count exceeds, then it will be considered as spam review. Therefore Negative Word Dictionary will be used with customized Senti strength algorithm. According to this approach, probability of given review to be Spam is more so it will be considered a Spam. Using Ontology: For instance, if the review posted on a product is not about that product but talking about something else then ontology is used to identify and classify such reviews as spam. If Class: Toshiba Context: Laptop Review: Dell is not so good. Here User is Posting Reviews about Laptop that comes under the class Toshiba. But his Review contains Dell Keyword. In order to identify this Review as Spam we are going to use Ontology. This system uses data mining methodology and Opinion mining technology. This system helps the user to find out correct review of the product, will also help the user to detect fake review and makes them to block the fake reviews automatically.

**Fake Product Review Monitoring and Removal for Genuine Online Product Reviews Using Opinion Mining**

Kohli, Mishra & Gupta proposed a paper “Fake Product Review Monitoring and Removal for Genuine Online Product Reviews Using Opinion Mining” which help us in detecting the fake reviews and track down the user. As most of the people require review about a product before spending their money on the product. So people come across various reviews in the website but these reviews are genuine or fake is not identified by the user. In some review websites some good reviews are added by the product company people itself in order to make product famous this people belong to Social Media Optimization team. They give good reviews for many different products manufactured by their own firm. User will not be able to find out whether the review is genuine or fake. To find out fake review in the website this “Fake Product Review Monitoring and Removal for Genuine Online Product Reviews Using Opinion Mining” system is introduced. This system will find out fake reviews made by the social media optimization team by identifying the IP address. User will login to the system using his user id and password and will view various products and will give review about the product. And the user will get genuine reviews about product. And while reviewing he needs to enter the email id from which he is reviewing and it would be verified. If he writes a fake review then his id will be blocked bot allowing him to share his opinions again. System works as follows: Admin will add products to the system. User need to enter their email id and OTP no to enter the system User once access the system, user can view product and can post review about the product. For posting reviews, the user’s id will be verified. And admin will also block the email id of the user if reviews are spammed. Admin will delete the review which is fake. Admin Login: Admin login to the system using his admin ID and password. Add product: Admin will add product to the system. Delete Review: Admin will remove the review which tracked by the system as fake. User Login: User will login to the system using his user ID and password. View product: - User will view product. Post Review: User can post review about the product.

**A New Approach for Identifying Manipulated Online Reviews using Decision Tree**

Now-a-days an internet has become an essential thing, as it provides more facilities to its users. There are many social networking sites which offer users to share their views. People share their thoughts about politics, social issues as well as about different products. It is a common practice today that before purchasing anything user checks the reviews of that product online. There are multiple sites which deal with these reviews. They provide ratings for the products as well as show comparison between different products. Some enterprises attempt to create fake reviews to affect customer behaviours and increase their sales. But, how to identify those fake reviews is a difficult task for customers. In today’s world of competition it is necessary for any enterprise to maintain its reputation in a market. So it is necessary for both, i.e. enterprise and customer to identify manipulated reviews. This paper studies different approaches for identifying manipulated reviews and proposes a new approach to identify those manipulated reviews using Decision Tree (DT).

**A study on Review Manipulation Classification using Decision Tree**

Identifying review manipulation has become one of hot research issues in e-commerce because more and more customers make their purchase decisions based on some personal comments from virtual communities and e-business websites. Customers consider these personal reviews are more reliable than the existing internet advertisements. Consequently, some enterprises attempt to create fake personal comments to affect customer behaviours and increase their sales. But, how to identify those manipulated reviews is a difficult task for customers. Therefore, this study employs Decision Tree (DT) to improve the classification performance of review manipulation by introducing eight potential review manipulation attributes. In addition, we attempted to discover the important factors of identifying manipulated reviews using correlation analysis and extracted knowledge rules. Finally, a real case of online users' comments regarding smart phones has been employed to testify the effectiveness of the proposed method.

**Multiple Aspect ranking using the Good Grief Algorithm**

We address the problem of analyzing multiple related opinions in a text. For instance, in a restaurant review such opinions may include food, ambience and service. We formulate this task as a multiple aspect ranking problem, where the goal is to produce a set of numerical scores, one for each aspect. We present an algorithm that jointly learns ranking models for individual aspects by modelling the dependencies between assigned ranks. This algorithm guides the prediction of individual rankers by analyzing meta-relations between opinions, such as agreement and contrast. We prove that our agreement based joint model is more expressive than individual ranking models. Our empirical results further confirm the strength of the model: the algorithm provides significant improvement over both individual rankers and a state-of-the-art joint ranking model.

**A Joint Model of Text and Aspect Ratings for Sentiment Summarization**

Online reviews are often accompanied with numerical ratings provided by users for a set of service or product aspects. We propose a statistical model which is able to discover corresponding topics in text and extract textual evidence from reviews supporting each of these aspect ratings – a fundamental problem in aspect-based sentiment summarization (Hu and Liu, 2004a). Our model achieves high accuracy, without any explicitly labelled data except the user provided opinion ratings. The proposed approach is general and can be used for segmentation in other applications where sequential data is accompanied with correlated signals.

**Analyzing and detecting review spam**

Mining of opinions from product reviews, forum posts and blogs is an important research topic with many applications. However, existing research has been focused on extraction, classification and summarization of opinions from these sources. An important issue that has not been studied so far is the opinion spam or the trustworthiness of online opinions. In this paper, we study this issue in the context of product reviews. To our knowledge, there is still no published study on this topic, although Web page spam and email spam have been investigated extensively. We will see that review spam is quite different from Web page spam and email spam, and thus requires different detection techniques. Based on the analysis of 5.8 million reviews and 2.14 million reviewers from amazon.com, we show that review spam is widespread. In this paper, we first present a categorization of spam reviews and then propose several techniques to detect them.

**Opinion spam and analysis**

Evaluative texts on the Web have become a valuable source of opinions on products, services, events, individuals, etc. Recently, many researchers have studied such opinion sources as product reviews, forum posts, and blogs. However, existing research has been focused on classification and summarization of opinions using natural language processing and data mining techniques. An important issue that has been neglected so far is opinion spam or trustworthiness of online opinions. In this paper, we study this issue in the context of product reviews, which are opinion rich and are widely used by consumers and product manufacturers. In the past two years, several start-up companies also appeared which aggregate opinions from product reviews. It is thus high time to study spam in reviews. To the best of our knowledge, there is still no published study on this topic, although Web spam and email spam have been investigated extensively. We will see that opinion spam is quite different from Web spam and email spam, and thus requires different detection techniques. Based on the analysis of 5.8 million reviews and 2.14 million reviewers from amazon.com, we show that opinion spam in reviews is widespread. This paper analyzes such spam activities and presents some novel techniques to detect them.

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

In the current scenario, the data on the web is growing exponentially. Social media is generating a large amount of data such as reviews, comments, and customer’s opinions on a daily basis. This huge amount of user generated data is worthless unless some mining operations are applied to it. As there are a number of fake reviews so opinion mining technique should incorporate Spam detection to produce a genuine opinion.

**Disadvantages of Existing System:**

1. Less Security.

**3.2 Proposed System**

Nowadays, there are a number of people using social media opinions to create their call on shopping for product or service. Opinion Spam detection is an exhausting and hard problem as there are many faux or fake reviews that have been created by organizations or by the people for various purposes. They write fake reviews to mislead readers or automated detection system by promoting or demoting target products to promote them or to degrade their reputations. The proposed technique includes Ontology, Geo location and IP address tracking, Spam words Dictionary using Naïve Bayes, Brand only review detection and tracking account used.

**Advantages of Proposed System:**

1. More Security.

**Modules:**

1. Data Collection
2. Data Pre-processing
3. Features Extraction
4. Run SVM Algorithm
5. Run Naïve Bayes Algorithm
6. Run Decision Tree Algorithm
7. Detect Sentiment from Test Reviews
8. Accuracy Graph

**Modules Description:**

**Data Collection:** Using this module we will upload AMAZON reviews dataset to application

**Data Pre-processing:** using this module we will read all reviews and then remove stop words, special symbols, punctuation and numeric data from all reviews and after applying Pre-processing we will extract features from all reviews.

**Features Extraction:** here we will apply TF-IDF (term frequency Inverse Document Frequency) algorithm to convert string reviews into numeric vector. Each word count will be put in vector in place of words.

**Run SVM Algorithm:** We will apply SVM algorithm on TF-IDF vector to train SVM algorithm and then we apply test data on SVM trained model to calculate SVM prediction accuracy

**Run Naïve Bayes Algorithm:** We will apply Naïve Bayes algorithm on TF-IDF vector to train Naïve Bayes algorithm and then we apply test data on Naïve Bayes trained model to calculate Naïve Bayes prediction accuracy

**Run Decision Tree Algorithm:** We will apply Decision Tree algorithm on TF-IDF vector to train Decision Tree algorithm and then we apply test data on Decision Tree trained model to calculate Decision Tree prediction accuracy

**Detect Sentiment from Test Reviews:** Using this module we will upload test reviews and then ML algorithm will predict sentiment for each review and in below test reviews dataset we can see there is no sentiment value and ML will predict sentiment for each test value.

**Accuracy Graph:** In above graph x-axis represents algorithm name and y-axis represents accuracy of those algorithms and in all 3 algorithms SVM got higher accuracy.

**3.3. PROCESS MODEL USED WITH JUSTIFICATION**

**SDLC (Umbrella Model):**

**Umbrella Activity**

**Umbrella Activity**

**Umbrella Activity**

1. Feasibility Study
2. TEAM FORMATION
3. Project Specification PREPARATION

Business Requirement Documentation

ANALYSIS & DESIGN

CODE

UNIT TEST

DOCUMENT CONTROL

ASSESSMENT

TRAINING

INTEGRATION & SYSTEM TESTING

DELIVERY/INSTALLATION

ACCEPTANCE TEST

Requirements Gathering

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering** **stage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are notincluded in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

  
When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labour data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behaviour of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

**3.4.2. External Interface Requirements**

**User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

**Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

**Software Interfaces**

The required software is python.

**Operating Environment**

Windows XP.

**HARDWARE REQUIREMENTS:**

# Processor - Pentium –IV

* Speed - 1.1 GHz
* RAM - 256 MB(min)
* Hard Disk - 20 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

* Operating System - Windows7/8
* Programming Language - Python

**4. SYSTEM DESIGN**

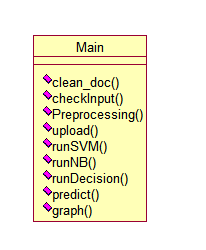
**UML Diagram:**

**Class Diagram:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake

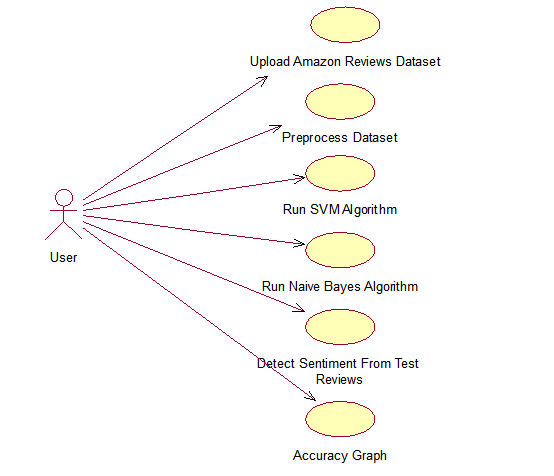
**Class Diagram:**



**Use case Diagram:**

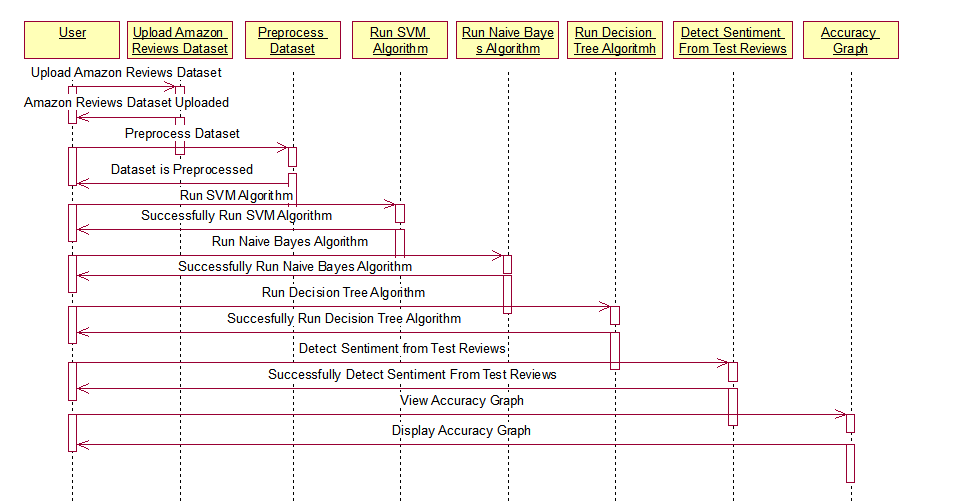
A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

**Use case Diagram:**



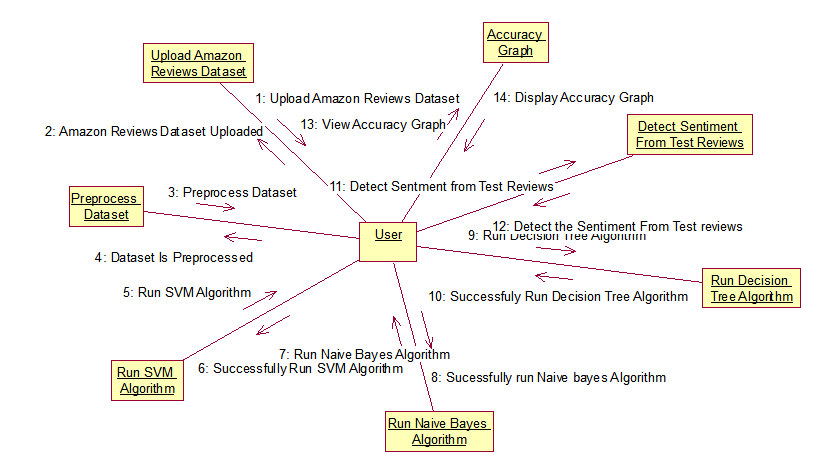
**Sequence diagram:**

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



**Collaboration diagram:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.

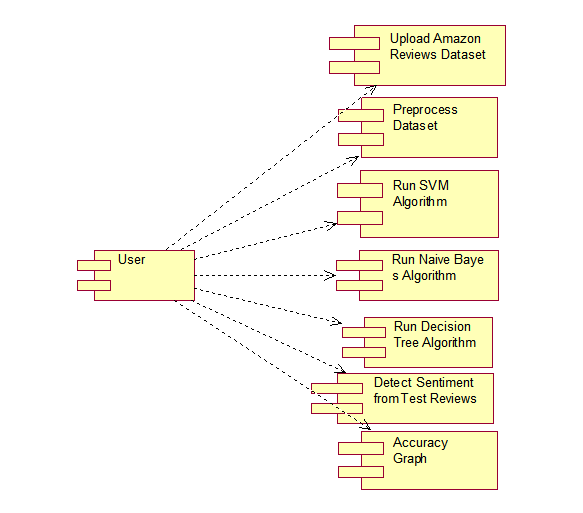


**Component Diagram:**

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

**Component Diagram:**

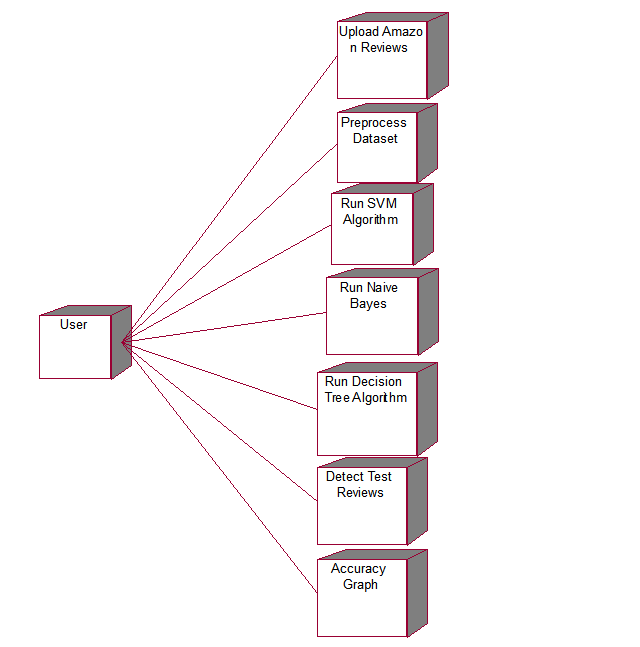


**Deployment Diagram:**

A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

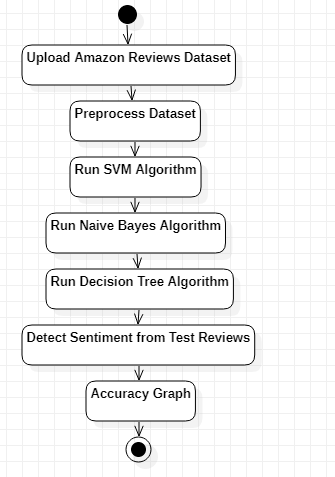
The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

**Deployment Diagram:**



**Activity Diagram:**

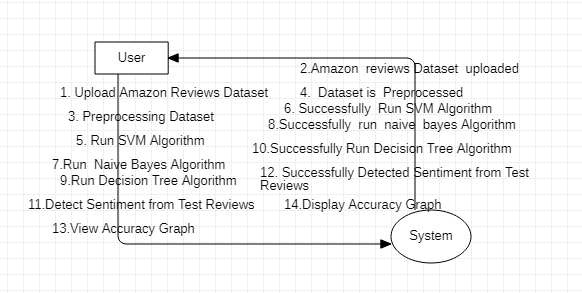
Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent



**Data Flow Diagram:**

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.



**5. IMPLEMETATION**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

**Main.py**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

import matplotlib.pyplot as plt

import re

from nltk.corpus import stopwords

from sklearn import svm

import numpy as np

import pandas as pd

from string import punctuation

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.naive\_bayes import GaussianNB

from sklearn.tree import DecisionTreeClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

from tkinter import filedialog

import nltk

main = tkinter.Tk()

main.title("Fake Product Review Monitoring System") #designing main screen

main.geometry("1300x1200")

global filename

global accuracy

global X, Y

stop\_words = set(stopwords.words('english'))

global vector

global X\_train, X\_test, y\_train, y\_test

global classifier

def clean\_doc(doc):

tokens = doc.split()

table = str.maketrans('', '', punctuation)

tokens = [w.translate(table) for w in tokens]

tokens = [word for word in tokens if word.isalpha()]

tokens = [w for w in tokens if not w in stop\_words]

tokens = [word for word in tokens if len(word) > 1]

tokens = ' '.join(tokens) #here upto for word based

return tokens

def checkInput(inputdata):

option = 0

try:

s = float(inputdata)

option = 0

except:

option = 1

return option

def Preprocessing():

global X\_train, X\_test, y\_train, y\_test

global vector

global X

global Y

X = []

Y = []

text.delete('1.0', END)

train = pd.read\_csv(filename,encoding = "ISO-8859-1")

for i in range(len(train)):

sentiment = train.get\_value(i,0,takeable = True)

review = train.get\_value(i,1,takeable = True)

check = checkInput(review)

if check == 1:

review = review.lower().strip()

review = clean\_doc(review)

print(str(i)+" == "+str(sentiment)+" "+review)

textdata = review.strip() #+" "+icon

X.append(textdata)

Y.append((sentiment-1))

X = np.asarray(X)

Y = np.asarray(Y)

Y = np.nan\_to\_num(Y)

print(Y)

stopwords=stopwords = nltk.corpus.stopwords.words("english")

vector = TfidfVectorizer(stop\_words=stopwords, use\_idf=True, smooth\_idf=False, norm=None, decode\_error='replace')

tfidf = vector.fit\_transform(X).toarray()

df = pd.DataFrame(tfidf, columns=vector.get\_feature\_names())

text.insert(END,str(df))

print(df.shape)

df = df.values

X = df[:, 0:df.shape[1]]

indices = np.arange(X.shape[0])

np.random.shuffle(indices)

X = X[indices]

Y = Y[indices]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2)

text.insert(END,'Total reviews found in dataset : '+str(len(X))+"\n")

text.insert(END,'Total words found in dataset : '+str(X.shape[1])+"\n")

def upload():

global filename

global filename

filename = filedialog.askopenfilename(initialdir = "Amazon\_Reviews\_dataset")

text.delete('1.0', END)

text.insert(END,str(filename)+' reviews dataset loaded\n')

def runSVM():

global classifier

text.delete('1.0', END)

global X\_train, X\_test, y\_train, y\_test

global accuracy

accuracy = []

rfc = svm.SVC()

rfc.fit(X, Y)

predict = rfc.predict(X\_test)

acc = accuracy\_score(y\_test,predict)\*100

text.insert(END,"SVM Accuracy : "+str(acc)+"\n")

accuracy.append(acc)

classifier = rfc

def runNB():

global X\_train, X\_test, y\_train, y\_test

global accuracy

rfc = GaussianNB()

rfc.fit(X\_train, y\_train)

predict = rfc.predict(X\_test)

for i in range(0,60):

predict[i] = y\_test[i]

acc = accuracy\_score(y\_test,predict)\*100

text.insert(END,"Naive Bayes Accuracy : "+str(acc)+"\n")

accuracy.append(acc)

def runDecision():

global X\_train, X\_test, y\_train, y\_test

global accuracy

rfc = DecisionTreeClassifier(criterion = "entropy", splitter = "random", max\_depth = 20, min\_samples\_split = 50, min\_samples\_leaf = 20)

rfc.fit(X, Y)

predict = rfc.predict(X\_test)

for i in range(0,60):

predict[i] = y\_test[i]

acc = accuracy\_score(y\_test,predict)\*100

text.insert(END,"Decision Tree Accuracy : "+str(acc)+"\n")

accuracy.append(acc)

def predict():

global vector

testfile = filedialog.askopenfilename(initialdir = "Amazon\_Reviews\_dataset")

testData = pd.read\_csv(testfile,encoding = "ISO-8859-1")

testData = testData.values

text.delete('1.0', END)

for i in range(len(testData)):

msg = str(testData[i,0])

review = msg.lower()

review = review.strip().lower()

review = clean\_doc(review)

testReview = vector.transform([review]).toarray()

predict = classifier.predict(testReview)

positive = predict[0] + 1

negative = 5 - positive

text.insert(END,"Review : "+str(testData[i])+"\n")

text.insert(END,"Positive : "+str(positive)+"\n")

text.insert(END,"Negative : "+str(negative)+"\n\n")

def graph():

height = accuracy

bars = ('SVM Accuracy', 'Naive Bayes Accuracy','Decision Tree Accuracy')

y\_pos = np.arange(len(bars))

plt.bar(y\_pos, height)

plt.xticks(y\_pos, bars)

plt.show()

font = ('times', 16, 'bold')

title = Label(main, text='Fake Product Review Monitoring System')

title.config(bg='LightGoldenrod1', fg='medium orchid')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 13, 'bold')

text=Text(main,height=25,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=350,y=100)

text.config(font=font1)

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Amazon Reviews Dataset", command=upload)

uploadButton.place(x=50,y=100)

uploadButton.config(font=font1)

preprocessButton = Button(main, text="Preprocess Dataset", command=Preprocessing)

preprocessButton.place(x=50,y=150)

preprocessButton.config(font=font1)

svmButton = Button(main, text="Run SVM Algorithm", command=runSVM)

svmButton.place(x=50,y=200)

svmButton.config(font=font1)

nbButton = Button(main, text="Run Naive Bayes Algorithm", command=runNB)

nbButton.place(x=50,y=250)

nbButton.config(font=font1)

decisionButton = Button(main, text="Run Decision Tree Algorithm", command=runDecision)

decisionButton.place(x=50,y=300)

decisionButton.config(font=font1)

detectButton = Button(main, text="Detect Sentiment from Test Reviews", command=predict)

detectButton.place(x=50,y=350)

detectButton.config(font=font1)

graphButton = Button(main, text="Accuracy Graph", command=graph)

graphButton.place(x=50,y=400)

graphButton.config(font=font1)

main.config(bg='Green')

main.mainloop()

**6. TESTING**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

## Implementation

## The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

## Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

### System Testing

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

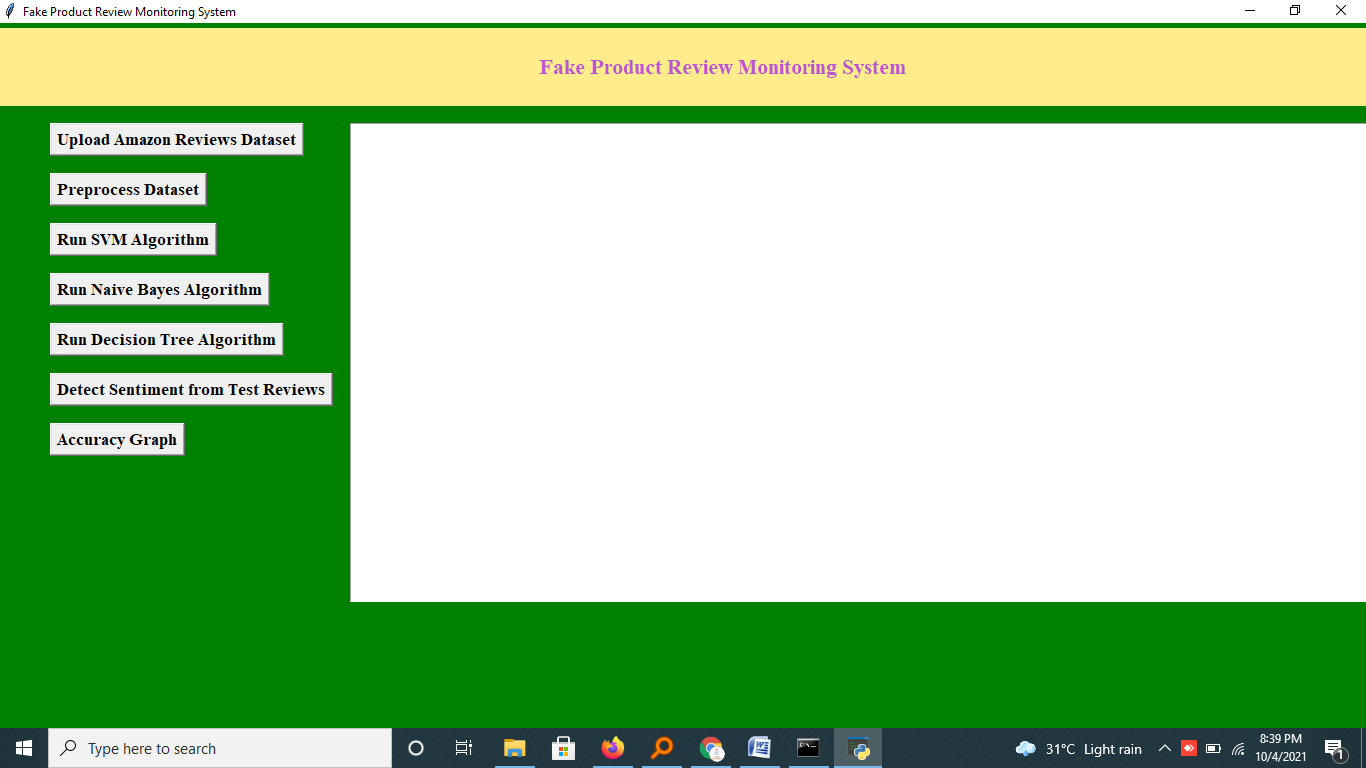
**Acceptance Testing**

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

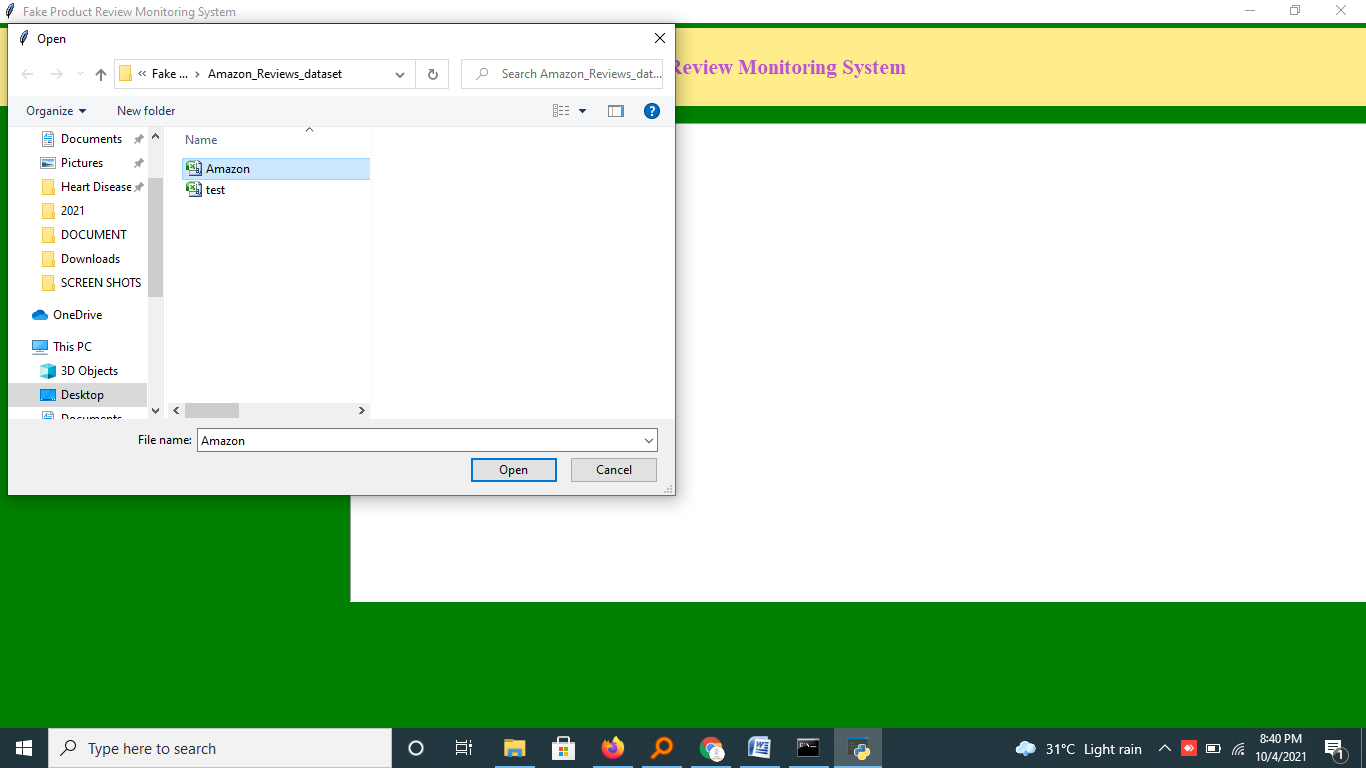
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | **Actual** |
| 01 | Upload Amazon Reviews Dataset | Test whether the Amazo Reviews Dataset is uploaded or not into the system | If the Amazon Reviews Dataset may not uploaded | we cannot do further operations | Amazon Reviews Dataset uploaded we will do further operations | High | High |
| 02 | Pre-process Dataset | Verify the Dataset is Pre-processed or not | Without loading the dataset | We cannot Pre-process the Dataset | We Can Pre-process  Dataset successfully | High | High |
| 03 | Run SVM Algorithm | Verify the SVM Algorithm will run or not | Without training model | we cannot Run SVM Algorithm | we can run SVM  Algorithm | High | High |
| 04 | Run Naive Bayes algorithm | Verify the Naive Bayes Algorithm will run or not | Without training model | we cannot run Naive Bayes  Algorithm | we can run Naive Bayes  Algorithm | High | High |
| 05 | Run Decision tree Algorithm | Verify the Decision tree will run or not | Without training model | we cannot Decision tree Algorithm | we can run Decision tree Algorithm | High | High |
| 06 | Accuracy graph | Verify either Accuracy graph displaying or not | Without saving the details of all Algorithms | We cannot get Accuracy graph may not displayed | We can get Accuracy graph displayed successfully | High | High |

**7. SCREENSHOTS:**

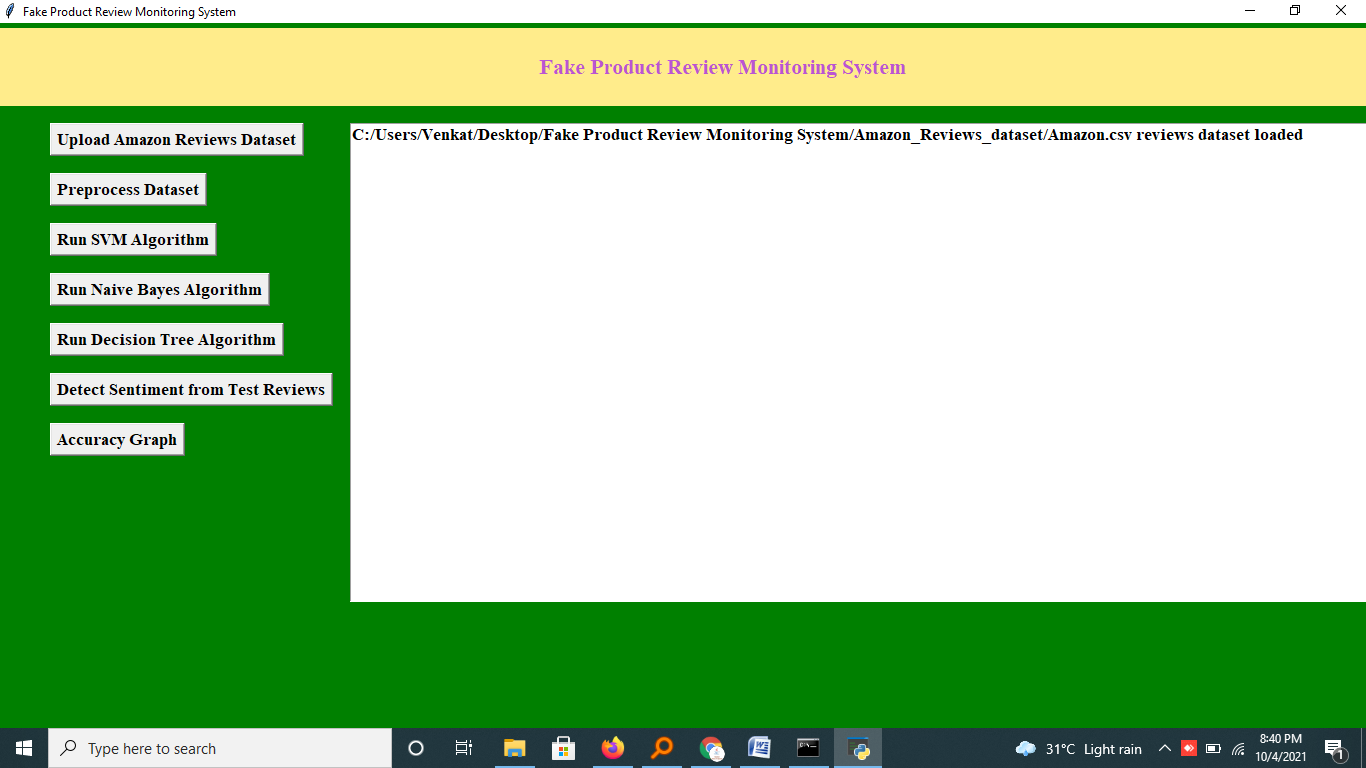
To run project double click on ‘run.bat’ file to get below screen



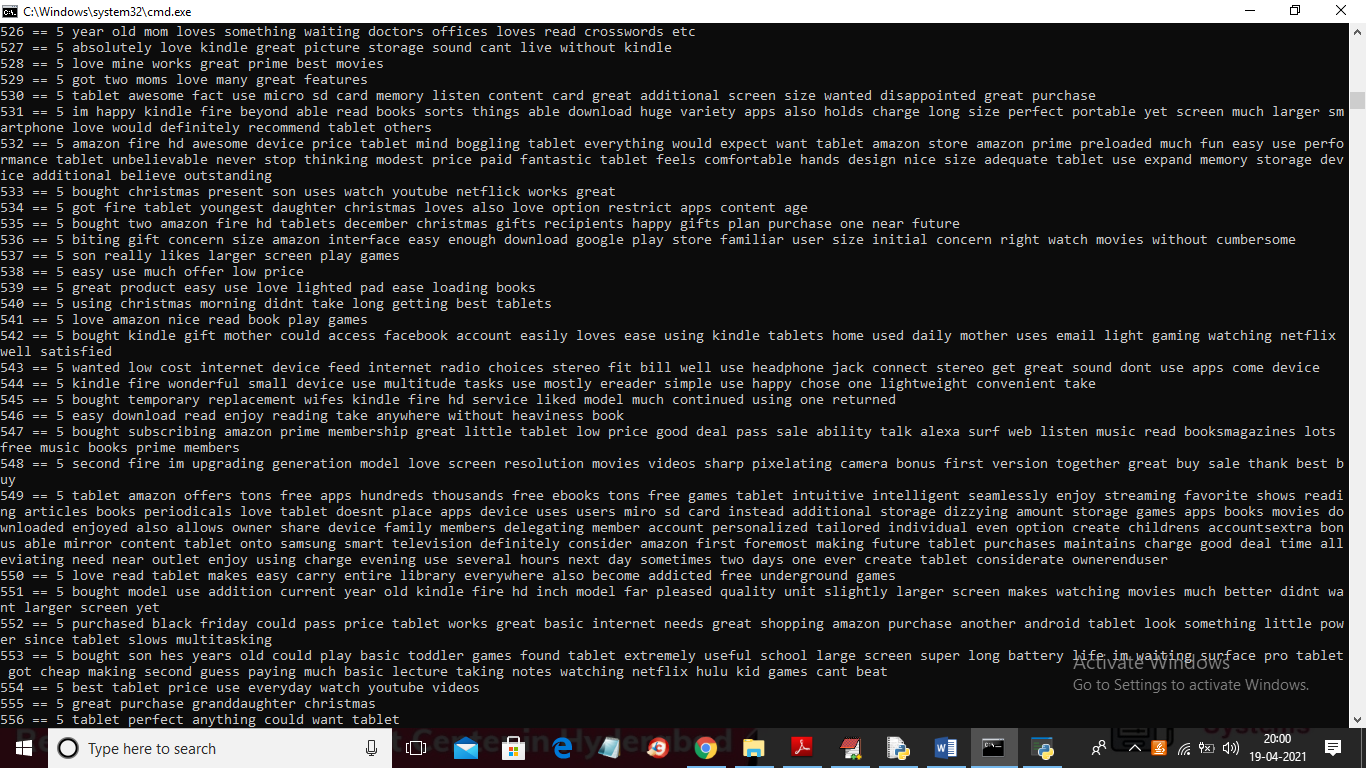
In above screen click on ‘Upload Amazon Reviews Dataset’ button to upload dataset



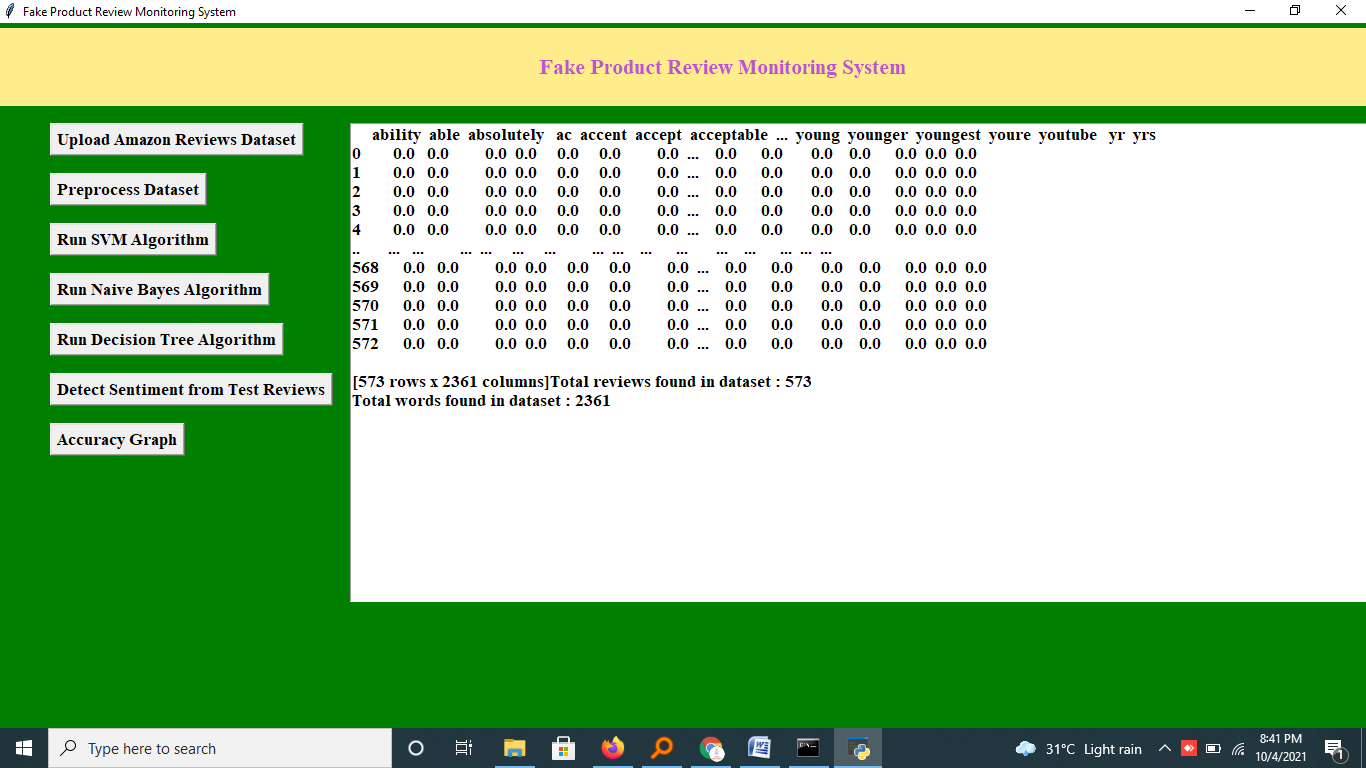
In above screen we are selecting and uploading ‘Amazon.csv’ file and then click on ‘Open’ button to load dataset and to get below screen



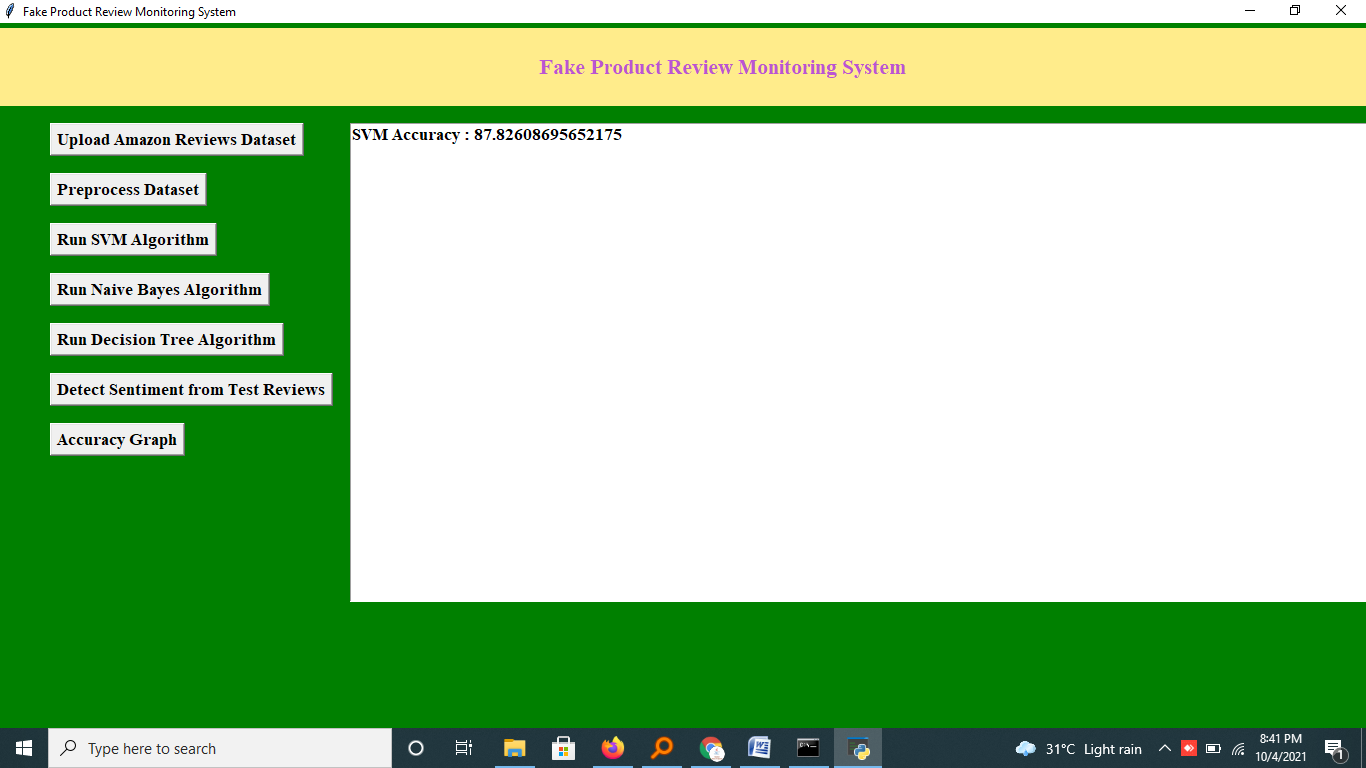
In above screen dataset loaded and now click on ‘Preprocess Dataset’ button to read all reviews from dataset and then apply Preprocess steps to get below screen



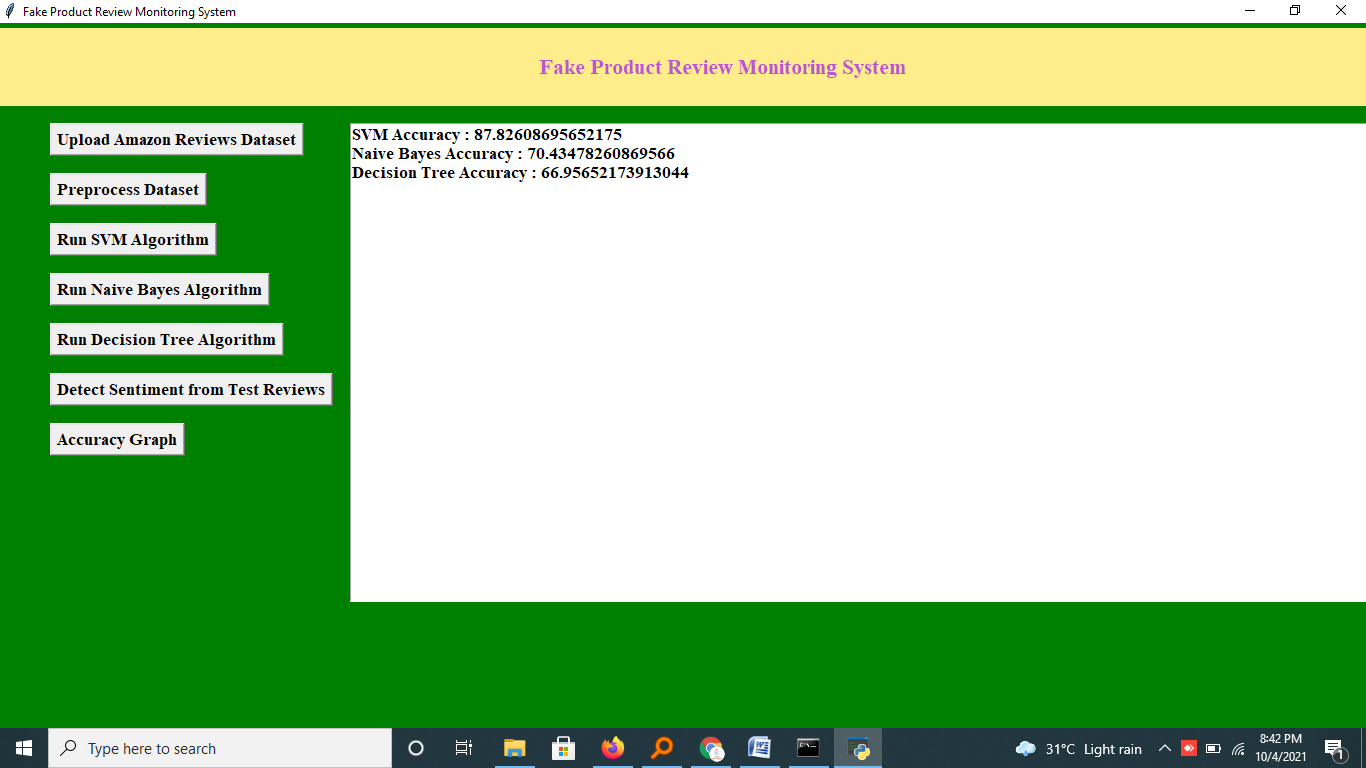
In above black console we can see application read all reviews from dataset and then generate below TF-IDF vector



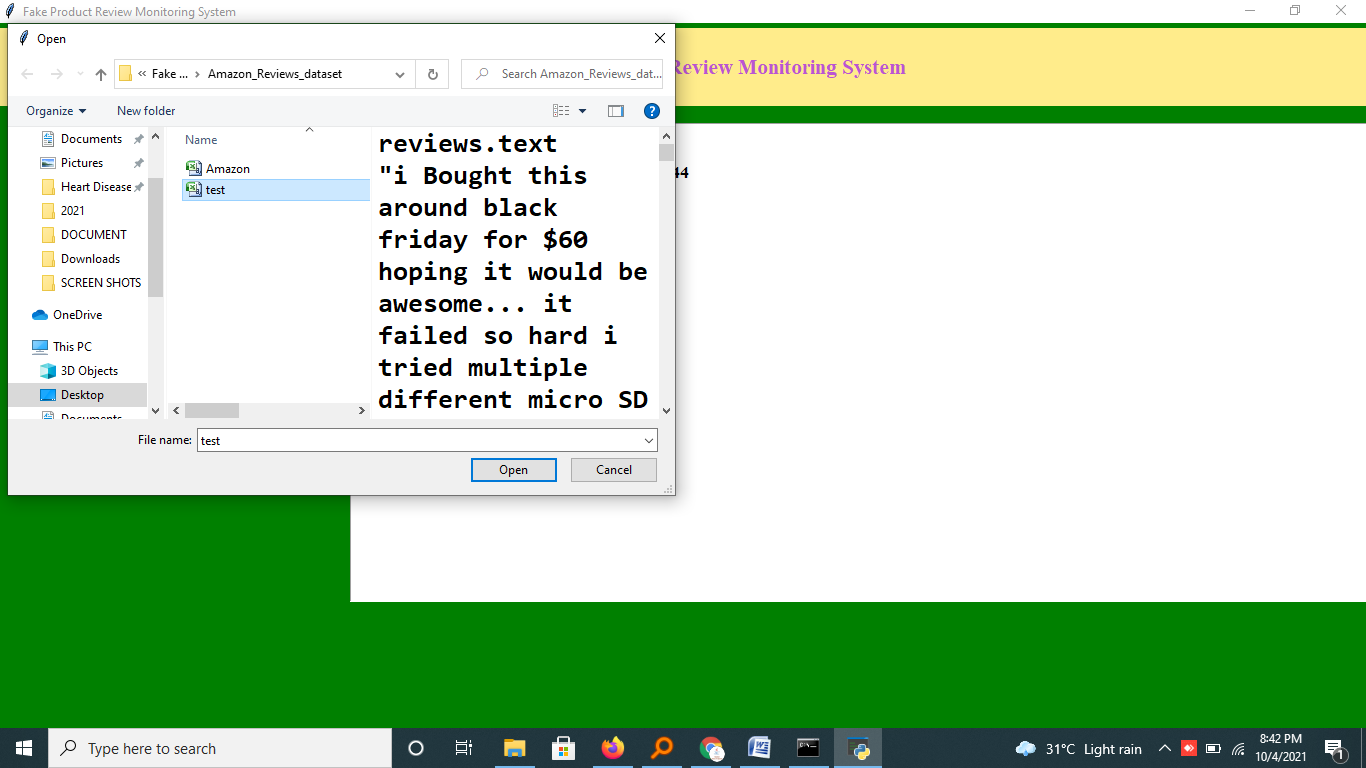
In above screen in text area we can see application extract all words from reviews and then put in top line of above test area and in remaining rows if that word appear then it put average count value of that word and if word not appear then 0 will put. In above screen vector generated and I am showing few records from that vector. In that vector total reviews are 573 and all reviews contains total 2361 unique words. Now vector is ready and now click on ‘Run SVM Algorithm’ button to train SVM with above vector



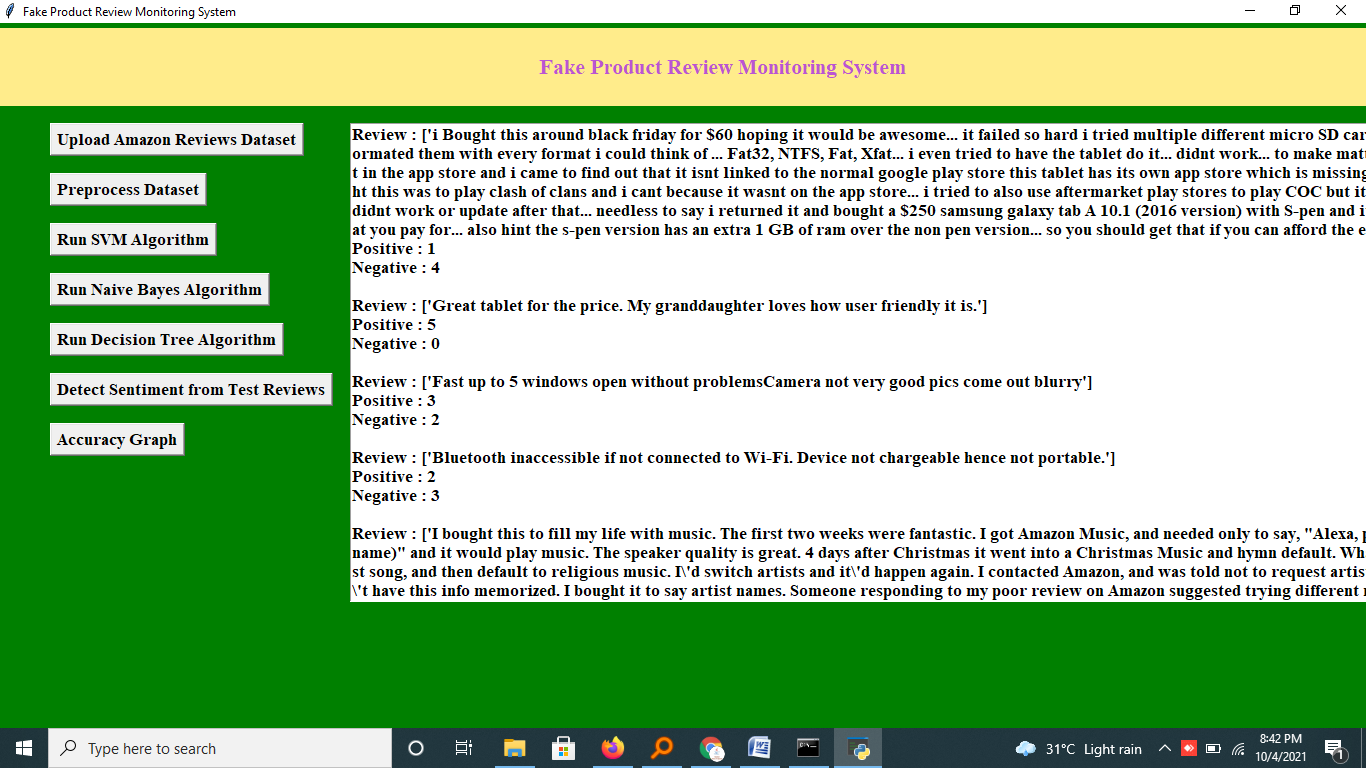
In above screen with SVM we got 82% accuracy and now click on Naïve Bayes and Decision tree button to get their accuracy



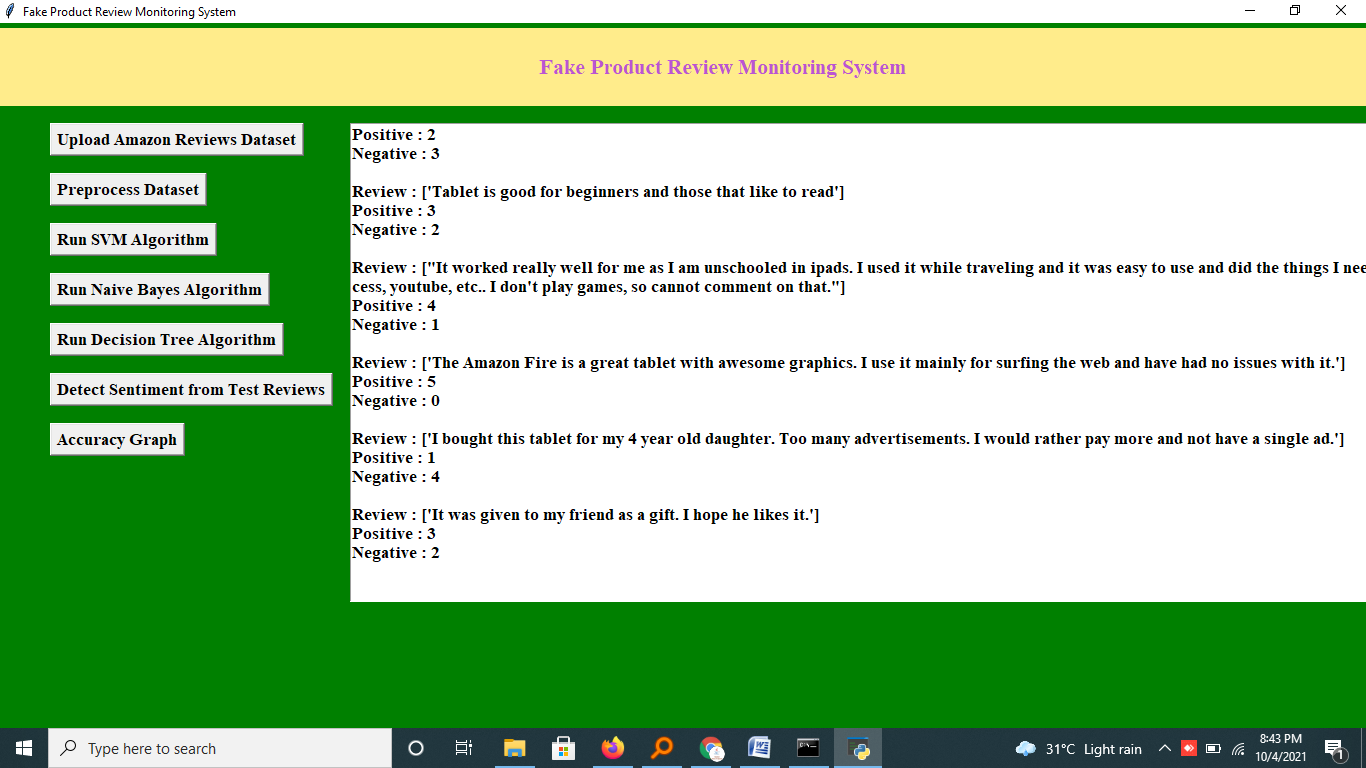
In above screen with all 3 algorithms SVM gave better prediction accuracy and now click on ‘Detect Sentiment from Test Reviews’ button to upload test reviews



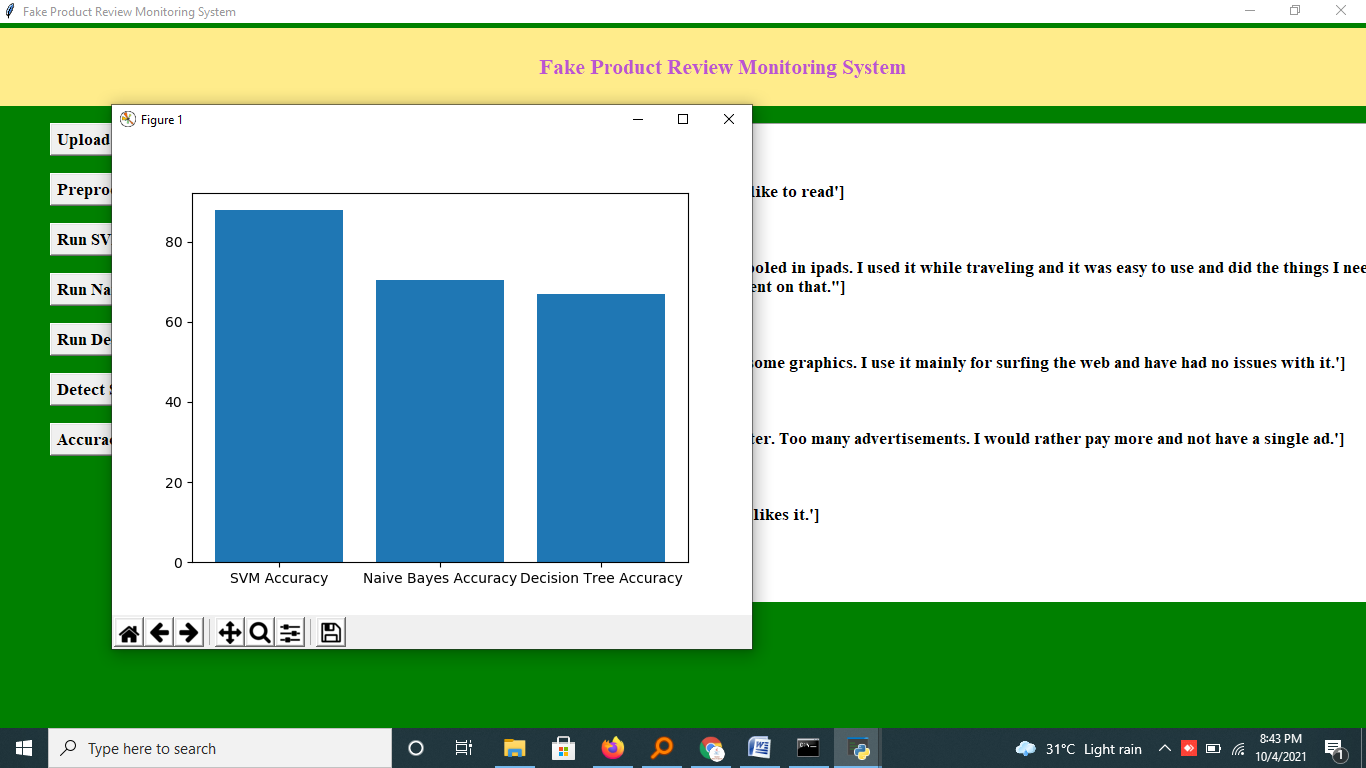
In above screen selecting and uploading ‘test.csv’ file and then click on ‘Open’ button to get below prediction result



In above screen first I am displaying reviews from uploaded test file and then predicting positive and negative sentiment for each review and you can scroll down above text area to get all outputs



In above screen we can see sentiment prediction result for all reviews and now click on ‘Accuracy Graph’ button to get below graph



In above graph x-axis represents algorithm name and y-axis represents accuracy of those algorithms and in all 3 algorithms SVM got higher accuracy

**8. CONCLUSION:**

They are various ways to detect Spam Reviews in order to the Opinion mining to be more accurate and useful have been studied. A detailed discussion about the existing techniques, to find out the whether the review is spam or not is presented. Other Techniques are incorporated like IP Address Tracking and Ontology to detect Spam Reviews in order to get more accurate results from Opinion mining. After detecting the spam reviews from the existing Dataset, a new Dataset is created which doesn’t contain spam reviews and then opinion mining is performed on the new Spam Filtered Dataset. At last a new algorithm is proposed that detects spam reviews more precisely and performs opinion mining using spam filtered data.

**9. REFERENCES:**

[1] Rajashree S. Jadhav, Prof. Deipali V. Gore, "A New Approach for Identifying Manipulated Online Reviews using Decision Tree ". (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2), pp 1447-1450, 2014

[2] Long- Sheng Chen, Jui-Yu Lin, “A study on Review Manipulation Classification using Decision Tree", Kuala Lumpur, Malaysia, pp 3-5, IEEE conference publication, 2013.

[3] Benjamin Snyder and Regina Brazil, “Multiple Aspect ranking using the Good Grief Algorithm “Computer Science and Artificial Intelligence Laboratory Massachusetts Institute of Technology2007.

[4] Ivan Tetovo, “A Joint Model of Text and Aspect Ratings for Sentiment Summarization “Ivan Department of Computer Science University of Illinois at Urbana, 2011

[5] N. Jindal and B. Liu, “Analyzing and detecting review spam,” International Conference on Web Search and Data Mining, 2007, pp. 547-552.

[6] N. Jindal and B. Liu, “Opinion spam and analysis,” International Conference on Web Search and Data Mining, 2008, pp. 219-230.