DR. T. THIMMAIAH INSTITUTE OF TECHNOLOGY

OORGAUM, K.G.F. – 563 120 (KARNATAKA)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled

"*ProGuard:* Detecting Malicious Accounts in Social-Network-Based Online Promotions"

Is bonafied work carried out by

AVINASH S	**1GV15CS401**
JANCY SWETHA M	**1GV15CS404**
MATHEW SAJJAN S	**1GV14CS027**
MOHAMMED ASHRAF V A	**1GV14CS029**

In partial fulfillment for the award of degree of **BACHELOR OF ENGINEERING** in Computer Science and Engineering of **VISVESWARAYA TECHNOLOGICAL UNIVERSITY**, Belgaum during the year 2017-2018.It is certified that all corrections suggestions indicated for internal assessment has been incorporated in the report kept in the department library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Signature of Guide	Signature of H.O.D	Signature of Principal
(Mrs. Sophia S)	(Mrs. Vinutha B.A)	(Dr. Syed Ariff)
	External Viva	PRINCIPAL Dr. T. Thimmaiah Institute of Technology Oorgaum, K.G.F. - 563 120.
Name of the Examiners		Signature with Date
1. Manjunath Singh. H		13/6/18
2. VASUDEVA R		13/06/18

ACKNOWLEDGMENT

It gives us a great pleasure to acknowledge with thanks, to the assistance and contribution of many individuals, who have been actively, involved at various stages of this project a success. We also wish to express our sincere feelings of gratitude to **VISVESVARAYA TECHNOLOGICAL UNIVERSITY** for providing as an opportunity of accomplishing a long cherished future if software engineering.

We are grateful to our institution **DR.T.THIMMAIAH INSTITUTE OF TECHNOLOGY** for providing as a great opportunity to pursue of degree course. We wish to express a whole hearted thanks to our principal **DR. SYED ARIFF** for creating an excellent and technical sound academic environment in our institute.

We thank **Mrs. VINUTHA B.A H.O.D, Dept. of CSE** for giving as valuable guidelines and advise to complete this project successfully.

We are also thankful to our project coordinator, **Mr. MANJUNATH SINGH,** Asst. Prof., Dept. of CSE for giving us valuable guidance and advice to complete this project successfully.

We thank our guide **Mrs. SOPHIA S,** Asst. Prof., Dept. of CSE who guided us with useful comments and timely suggestion during the course of project.

We would also thank all the teaching and non-teaching staff of CSE department whose suggestion enabled us to surpass many of the seemingly impossible hurdles and last but not least we would like to thank my parents for what we are today and finally our friends who helped me in successful completion of our project.

AVINASH S

JANCY SWETHA M

MATHEW SAJJAN S

MOHAMMED ASHRAF VA

ABSTRACT

Online social networks gradually integrate financial capabilities by enabling the usage of real and virtual currency. They serve as new platforms to host a variety of business activities such as online promotion events, where users can possibly get virtual currency as rewards by participating such events. Both OSNs and business partners are significant concern when attackers instrument a set of accounts to collect virtual currency from these events, which make these events ineffective and result in significant financial loss. It becomes of great importance to proactively detecting these malicious accounts before the online promotion activities and subsequently decreases their priority to be rewarded.

Since online social networks play an increasing important role in both cyber and business world, detecting malicious users in OSNs becomes of great importance. Many detection methods have been consequently proposed .Considering the popularity of spammers in OSNs; these methods almost exclusively focus on detecting accounts that send malicious content. A spamming attack can be considered as an information flow initiated from an attacker, through a series of malicious accounts, and finally to a victim account.

In order to effectively detect malicious accounts in online promotion activities by overcoming the aforementioned challenges, we have designed a novel system, namely ProGuard. ProGuard employs a collection of behavioral features to profile an account that participates in an online promotion event. These features aim to characterize an account from three aspects including i) its general usage profile, ii) how an account collects virtual currency, and iii) how the virtual currency is spent. ProGuard further integrates these features using a statistical classifier so that they can be collectively used to discriminate between those accounts controlled by attackers and benign ones.

We have performed extensive experiments based on data collected from Ten cent QQ, a global leading OSN with built-in financial management activities. Experimental results have demonstrated that our system can accomplish a high detection rate of 96.67% at a very low false positive rate of 0.3%.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Online social networks (OSNs) that integrate virtual currency serve as an appealing platform for various business activities, where online, interactive promotion is among the most active ones. Specifically, a user, who is commonly represented by her OSN account, can possibly get reward in the form of virtual currency by participating online promotion activities organized by business entities. She can then use such reward in various ways such as online shopping, transferring it to others, and even exchanging it for real currency. Such virtual-currency-enabled online promotion model enables enormous outreach, offers direct financial stimuli to end users, and meanwhile minimizes the interactions between business entities and financial institutions. As a result, this model has shown great promise and gained huge prevalence rapidly.

However, it faces a significant threat: attackers can control a large number of accounts, either by registering new accounts or compromising existing accounts, to participate in the online promotion events for virtual currency. Such malicious activities will fundamentally undermine the effectiveness of the promotion activities, immediately voiding the effectiveness of the promotion investment from business entities and meanwhile damaging ONSs' reputation. Moreover, a large volume of virtual currency, when controlled by attackers, could also become a potential challenge against virtual currency regulation.

It therefore becomes of essential importance to detect accounts controlled by attackers in online promotion activities. In the following discussions, we refer to such accounts as malicious accounts. The effective detection of malicious accounts enables both OSNs and business entities to take mitigation actions such as banning these accounts or decreasing the possibility to reward these accounts. However, designing an effective detection method is faced with a few significant challenges. First, attackers do not need to generate malicious content (e.g., phishing URLs and malicious executables) to launch successful attacks.

Comparatively, attackers can effectively perform attacks by simply clicking links offered by business entities or sharing the benign content that is originally distributed by business partners. These actions themselves do not perceivably differentiate from benign accounts.' Second, successful attacks do not need to depend on social structures (e.g., "following" or

"friend" relationship in popular social networks). To be more specific, maintaining active social structures does not benefit to attackers, which is fundamentally different from popular attacks such as spammers in online social networks. These two challenges make the detection of such malicious OSN accounts fundamentally different from the detection of traditional attacks such as spamming and phishing. As a consequence, it is extremely hard to adopt existing methods to detect spamming and phishing accounts.

ProGuard employs a collection of behavioral features to profile an account that participates in an online promotion event. These features aim to characterize an account from three aspects including i) its general usage profile ii) how an account collects virtual currency, and iii) how the virtual currency is spent. ProGuard further integrates these features using a statistical classifier so that they can be collectively used to discriminate between those accounts controlled by attackers and benign ones.

1.1Objective

- Design a online shopping site to collect features
- Design a naïve Bayer's classifier and train it to classify the account to malicious or normal.

1.2 Scope

Considering the active trend of integrating OSNs with financial capabilities, detecting malicious accounts that engage in suspicious financial activities becomes of central importance. Although the design and evaluation of ProGuard are based on real-world data collected from Ten cent QQ, a leading OSN with 899 million active accounts, the features and the detection framework can be easily applied to other OSNs that integrate financial activities. Specifically, all the proposed features are based on essential financial functions such as recharging and gifting.

In addition, all current features rely on coarse-grained information that minimizes privacy concerns, which may foster the deployment of the proposed system in a detection-as-service model. Despite the fact that ProGuard can effectively detect malicious accounts used for collecting virtual currency from online promotion activities, it is not designed for detecting malicious accounts used for transferring and laundering virtual currency. Extending ProGuard to include such detection capabilities falls into our future work.

CHAPTER 02

LITERATURE SURVEY

Online social network is a platform that benefits peoples to create there profiles, finding and making friends. Online social networks are more popular now days. There are some OSN popular networks such as Face book, Instagram and Twitter. Our key contribution is for Face book. Now a day's third party apps encourage the enhancement in online social network (OSN). Such enhancements include interesting or entertaining ways of communicating among online friends and diverse activities such as playing games or listening to songs. That is Face book provides an API [4] that facilitates integration of user experience. 20M apps installed everyday [3] and interesting to know there are 500K apps are available on social networks [5].

Recently, hackers have started taking advantage of the popularity of this third-party apps platform and deploying malicious applications [6]–[8]. Face book is more popular online social network site among all over the world that causes increase in black market services [23] that encourage growth in fake likes, comments and tags. Fake accounts are categorized into two types called as duplicate accounts and false accounts. Duplicate Account: A duplicate account refers to an account maintained by a user in addition to his/her principal account. False Accounts: False accounts are further broken down into two categories user misclassified accounts and undesirable accounts.

User-misclassified accounts: It represents the personal profiles created by users for a business, organization, or non-human entity such as a pet (Face book's terms of service permits such entities as a Page rather than a personal profile). Undesirable accounts: These are the user profiles that are intended to be used for purposes that violate Face book's terms of service, such as spamming. Fake accounts are mainly used to unfairly increase ones power and influence within a target community [24].

In this work we focused on detecting malicious applications and fake user accounts. Detection of fake user is done on the basis of the user activities and their interaction with other users on Face book through analysis of user feed data. There are many ways that hackers can benefit from a malicious app: 1) The app can reach large numbers of users and their friends to spread spam; 2) The app can obtain users' personal information such as email

address, hometown and gender; and 3) The app can —reproducel by making other malicious apps popular. The malicious apps are simplified by using ready-to use toolkit [9]. Because of this many malicious activities are spent more time on Face book [10]. Most research on Face book works in spam and malware [11]-[13].

In this paper, author develop FRAppE, a suite of efficient classification techniques for identifying whether an app is malicious or not and also detection of fake user. For building FRAppE we use My Page Keeper a Face book's security app [17].

A technique for computer detection and correction of spelling errors. [18] Authors F. J. Damerau. Description: A technique for computer detection and spelling errors. This paper describes that which word cannot be match in a dictionary, missing or extra letter or a single transpositions. The unique word which is get entered is compared to the dictionary again, testing each time to see whether the words match- assuming one of these errors occurred. The words which might be wrong or missing are get detected and correct to it.

Beyond Blacklist: learning to detect malicious web sites from suspicious URL's. [20] Authors: - J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Description: In this paper we describe an approach to this problem based on automated URL classification, using statistical methods. The resulting classifier obtain 95-99% accuracy, detecting large number of malicious web sites from their URL's, with only modest false positive.

Detecting suspicious URL's in Twitter stream. [22] Authors: - S. Lee, J. Kim. Description: Twitter can suffer from malicious tweets containing suspicious URLs for spam, phishing, and malware distribution. Attackers have limited resources and thus have to reuse them; a portion of their redirect chains will be shared. We focus on these shared resources to detect suspicious URLs. We have collected a large number of tweets from the Twitter public timeline and trained a statistical classifier with features derived from correlated URLs and tweet context information. Our classifier has high accuracy and low false- positive and false negative rates.

Design and evaluation of real time URL spam filtering service. [21] Authors: - K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song. Description: In particular, we find that spam targeting email qualitatively differs in significant ways from spam campaigns targeting Twitter. We explore the distinctions between email and Twitter spam, including the abuse of public web hosting and redirector services. Finally, we demonstrate Monarch's scalability,

showing our system could protect a service such as Twitter--which needs to process 15 million URLs/day for a bit under $800/day.

Only customer must hold the barcode side of the product wrapper in front of barcode scanner. Then corresponding data regarding product will be displayed on display. By using this trolley, customer can buy large number of product in very less time with less effort. At the billing counter, computer can be easily interfaced for verification and bill print out.

2.1 Problem Definition

Online social networks gradually integrate financial capabilities by enabling the usage of real and virtual currency. They serve as new platforms to host a variety of business activities such as online promotion events, where users can possibly get virtual currency as rewards by participating such events. Both OSNs and business partners are significantly concerned when attackers instrument a set of accounts to collect virtual currency from these events, which make these events ineffective and result in significant financial loss. It becomes of great importance to proactively detecting these malicious accounts before the online promotion activities and subsequently decreases their priority to be rewarded. In this paper, we propose a novel system, namely ProGuard, to accomplish this objective by systematically integrating features that characterize accounts from three perspectives including their general behaviors, their recharging patterns, and the usage of their currency.

2.2 Objective

Our objective is to design a detection system capable of identifying malicious accounts that participate in online promotion events for virtual currency collection (at the collection phase) before rewards are committed.

Detecting malicious accounts at this specific time point (i.e., before the commitment of rewards and at the collection phase) results in unique advantages. First, as a simple heuristic to prevent freshly registered accounts that are likely to be bots, business entities usually require the participating accounts to be registered for a certain amount of time (e.g., a few weeks).

Therefore, the detected and mitigated malicious accounts cannot be immediately replaced by the newly registered accounts, thereby drastically limiting attacker's capabilities.

In contrast, no constraint is applied for accounts used for virtual currency transferring and laundering. This implies such accounts can be easily replaced by attackers if detected, resulting negligible impact to attackers' capabilities. Second, our detection system will label whether an account is malicious when it participates in an online promotion event; this enables business entities to make actionable decisions such as de-prioritize this account from being rewarded in this event. Therefore, it can proactively mitigate the financial loss faced by business entities.

CHAPTER 03

PROPOESD SYSTEM

We propose a novel system, namely ProGuard, to accomplish this objective by systematically integrating features that characterize accounts from three perspectives including their general behaviors, their recharging patterns, and the usage of their currency.

3.1 Proposed Architecture

The System is first trained using features. Once the classifier is ready, account features are extracted from database of online shopping portal and given to classifier to classify the accounts to Malicious or normal.

The fundamental point of this part is to see if the framework is sufficiently achievable or not. Hence various types of examination, for example, execution investigation, specialized investigation, practical examination and so forth is performed.

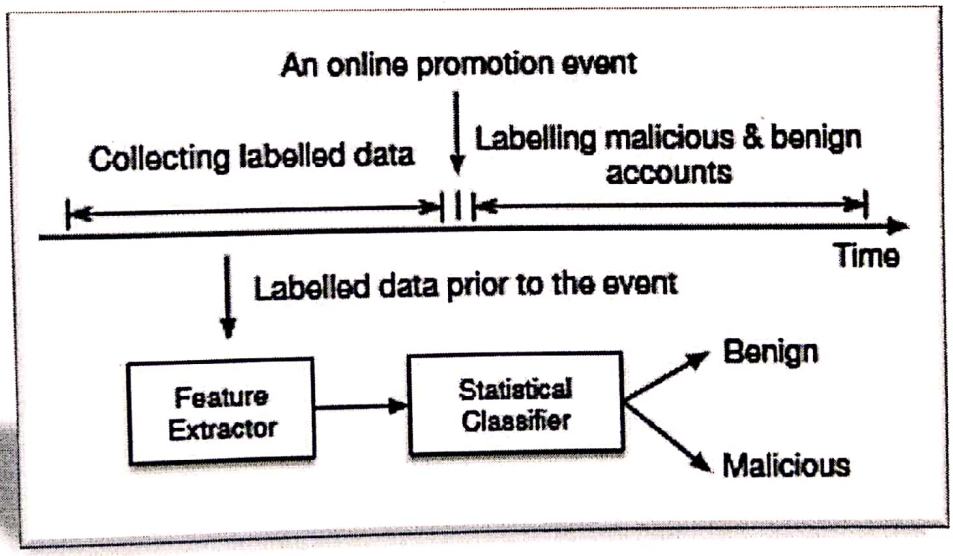

Figure 3.1: The Architectural Overview of the system

ProGuard is composed of two phases, namely the training phase and the detection phase. In the training phase, a statistical classifier is learnt from a set of pre-labeled malicious and benign accounts. In the detection phase, an unknown account will first be converted to a feature vector and then analyzed by the statistical classifier to assess its maliciousness.

The bottom of Figure 3 presents the architectural overview of ProGuard. As a variety of statistical classifiers have been developed and widely used, designing features capable of discriminating between malicious accounts and benign accounts becomes of central focus.

In this section, we will introduce various features and demonstrate their effectiveness on differentiating malicious accounts from benign ones. We propose three general guidelines to steer the feature design.

General Behaviors: Benign accounts are usually used by regular users for variety of activities such as chatting, photo sharing, and financial activities. In contrast, malicious Accounts are more likely to be driven by online promotion events. Therefore, the benign accounts tend to be more socially active compared to malicious accounts.

Currency Collection: The malicious accounts under investigation focus on using online promotion activities to collect virtual currency. In contrast, benign users are likely to obtain virtual currency from multiple resources.

Currency Usage: Attackers' ultimate objective is to monetize the virtual currency. In contrast, benign users use their virtual currency in much more diversified ways.

CHAPTER 04

SYSTEM REQUIREMENT SPECIFICATION

Software requirement specification captures all the functionalities to be implemented in this project.

4.1 Functional Requirement

The functionalities to be implemented are:

- Train a Naïve Bayer's classifier to categorize account to Malicious or Normal.
- Extract features from online shopping portal
- Classify the features using Naïve Bayer's classifier to classify the account to Malicious or Normal
- Block the Malicious account from participating in promotions.

4.2 Non-functional Requirement

The non functional comes under following category,

- Product Requirements
- Organizational Requirements
- User Requirements
- Basic Operational Requirements

4.2.1 Product Requirements

Portability: The project can work on any platform.

Accuracy: The classification accuracy must be higher.

Convenience: It should easy for the online shopping portal to use this solution.

Scalability: The system can work for any number of users.

4.2.2 Organizational Requirements

Process Standards: IEEE models are utilized to build up the application which is the standard utilized by the greater part of the standard programming designers everywhere throughout the world.

Design Methods: Modular Design approach is used to design the project.

4.2.3 User Requirements

- The user must be able to view the training and classification results.
- The GUI must be easy to use and informative.

4.2.4 Basic Operational Requirements

Mission profile or situation: The mission of the project is design a system to detect malicious users in online shopping and block them.

4.3 Resource Requirement

Netbeans IDE 7.0.1: Netbeans is a multi-dialect programming improvement environment involving an incorporated advancement environment (IDE) and an extensible module framework. It is composed principally in Java and can be utilized to create applications in Java and, by method for the different modules, in different dialects also, including C, C++, COBOL, Python, Perl, PHP, and others.

Netbeans utilizes modules with a specific end goal to give the greater part of its usefulness on top of (and including) the runtime framework, as opposed to some different applications where usefulness is regularly hard coded.

The Netbeans SDK incorporates the Netbeans java improvement devices (JDT), offering an IDE with an inherent incremental Java compiler and a full model of the Java source documents. This takes into consideration progressed refactoring strategies and code investigation. The IDE likewise makes utilization of a workspace, for this situation an arrangement of metadata over a level document space permitting outer record changes the length of the relating workspace "asset" is invigorated a short time later.

Swing: The Java Foundation Classes (JFC) comprises of five noteworthy parts: AWT, Swing, and Accessibility, Java 2D, and Drag and Drop. Java 2D has turned into a fundamental piece of AWT, Swing is based on top of AWT, and Accessibility backing is incorporated with Swing. The five sections of JFC are surely not fundamentally unrelated, and Swing is relied upon to union all the more profoundly with AWT in future variants of Java.

Swing is an arrangement of classes that gives more effective and adaptable segments than are conceivable with the AWT. Notwithstanding the commonplace parts, Swing supplies tabbed sheets, scroll sheets, trees, and tables.

4.4 Hardware Requirements

Processors	:	Intel I3 2.2 GHZ
RAM	:	4 GB.
Storage	:	100 GB.
Monitor	:	15"

4.5 Software (Tools &Technologies) Requirements

Coding	: Java ,JSP
Platform	: JDK 1.7
Database	: MySQL
Tool	: Net bean ide 7.2
OS	: Windows OS
Front end	: Swings , HTML

This chapter gives subtle elements of the practical necessities, non-useful prerequisites, asset necessities, equipment necessities, programming prerequisites and so forth. Again the non-practical necessities thus contain item prerequisites, hierarchical prerequisites, client prerequisites, fundamental operational prerequisites and so forth.

CHAPTER 5

SYSTEM DESIGN

5.1 System development methodology

System development method is a process through which a product will get completed or a product gets rid from any problem. Software development process is described as a number of phases, procedures and steps that gives the complete software. It follows series of steps which is used for product progress. The development method followed in this project is waterfall model.

5.1.1 Model phases

The waterfall model is a successive programming improvement process, in which advance is seen as streaming relentlessly downwards (like a waterfall) through the periods of Requirement start, Analysis, Design, Implementation, Testing and upkeep.

Prerequisite Analysis: This stage is worried about gathering of necessity of the framework. This procedure includes producing record and necessity survey.

Framework Design: Keeping the prerequisites at the top of the priority list the framework details are made an interpretation of into a product representation. In this stage the fashioner underlines on:- calculation, information structure, programming design and so on.

Coding: In this stage developer begins his coding with a specific end goal to give a full portray of item. At the end of the day framework particulars are just changed over into machine coherent register code.

Usage: The execution stage includes the genuine coding or programming of the product. The yield of this stage is regularly the library, executables, client manuals and extra programming documentation

Testing: In this stage all projects (models) are coordinated and tried to guarantee that the complete framework meets the product prerequisites. The testing is worried with check and approval.

Support: The upkeep stage is the longest stage in which the product is upgraded to satisfy the changing client need, adjust to suit change in the outside environment, right mistakes and oversights beforehand undetected in the testing stage, improve the proficiency of the product.

5.1.2Reason for choosing waterfall model as development method

- Clear venture destinations.
- Stable undertaking necessities.
- Progress of framework is quantifiable.
- Strict close down necessities.
- Helps you to be great.
- Logic of programming improvement is plainly caught on.
- Production of a formal detail
- Better asset designation.

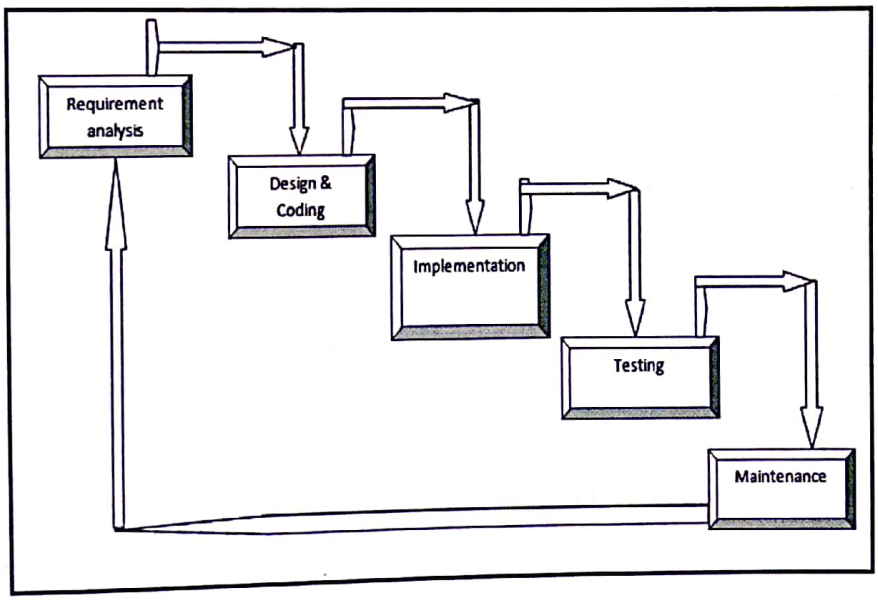

Fig 5.1 The Waterfall model

5.2 Software Architecture

The System architecture is shown below.

The Figure presents the architectural overview of ProGuard. As a variety of statistical classifiers have been developed and widely used, designing features capable of discriminating Between malicious accounts and benign accounts becomes of central focus.

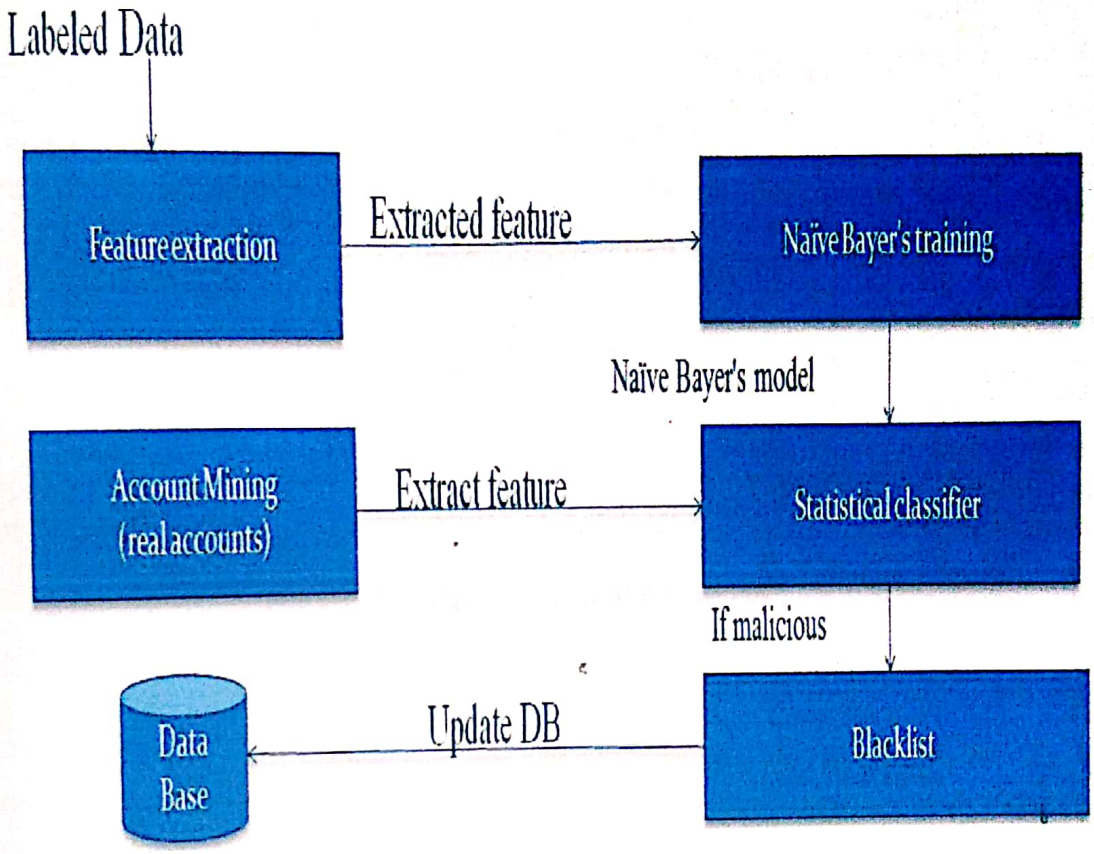

Fig 5.2: The software architecture of the proposed system.

5.3 Classes Designed for the system

A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes.

The class diagram is shown below.

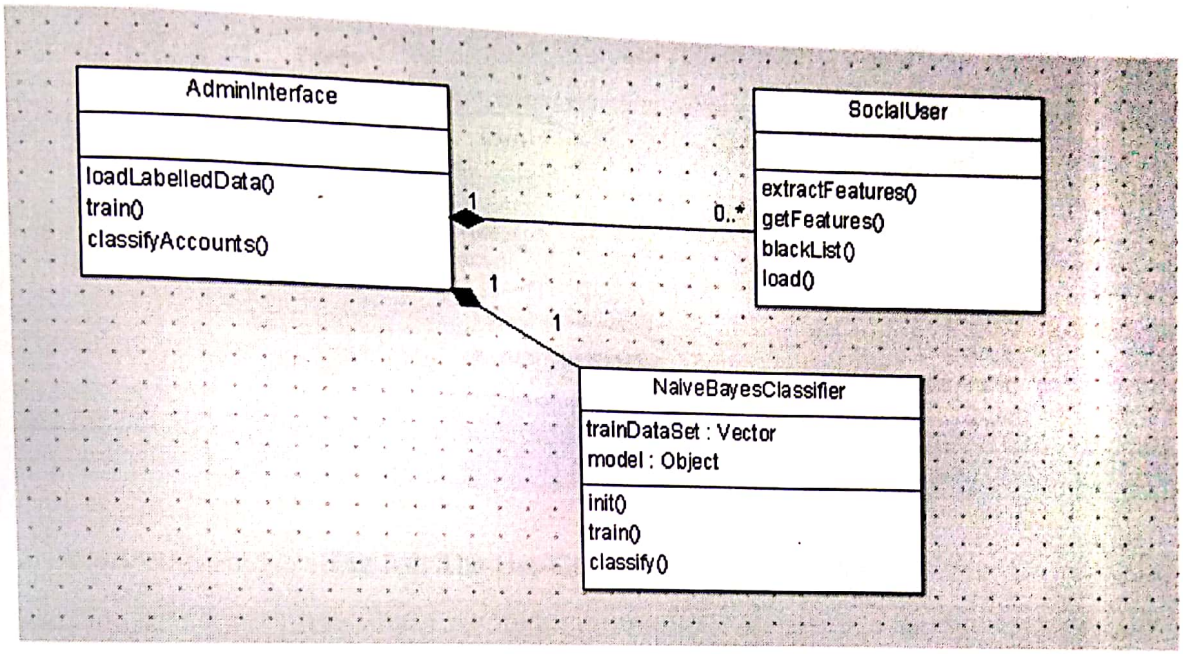

Fig 5.3: The class diagram of the system.

5.4 Use case Diagram of the system

A use case diagram is a type of behavioral diagram created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

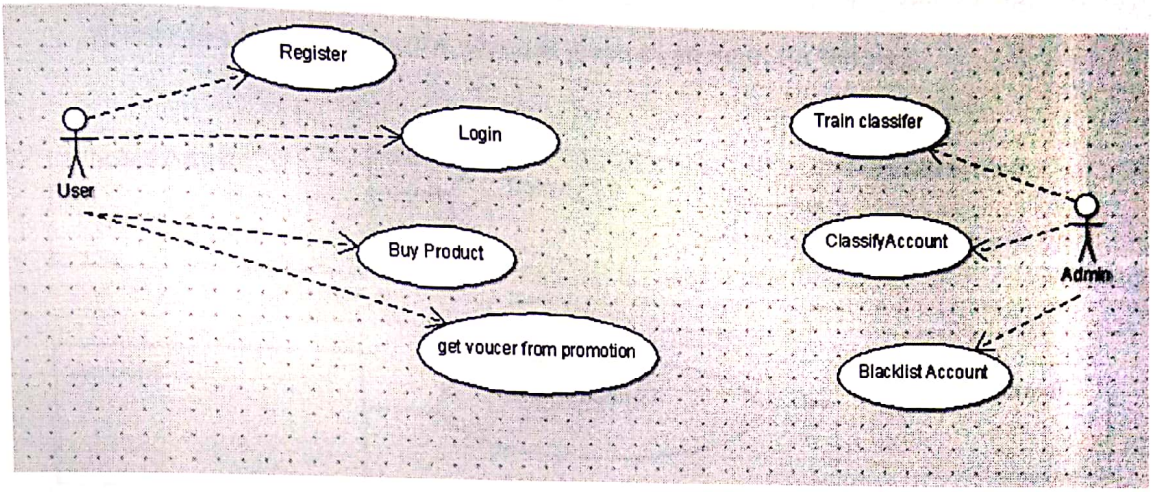

Fig 5.4: The Use-Case diagram of the system.

5.5 Data Flow Diagram of the system

The input, output and the process flow in the system is given in this section.

- We can either download or can create our own data set corresponding to the real time accounts of online social networks.
- The labeled data-set comprises of data values represented in terms of integers that specify the account behavior which is given as an input for training.

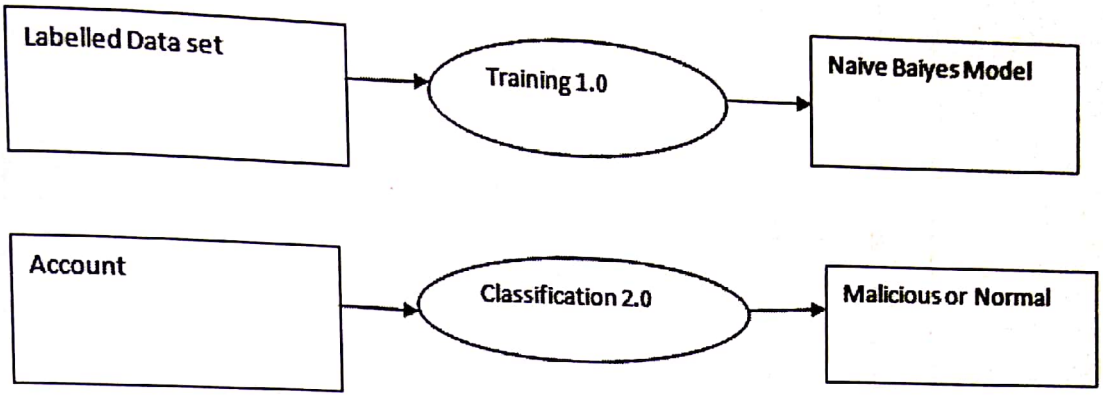

Fig 5.5: The level 0 data-flow diagram.

- Using the naïve Bayer's algorithm, we train for a statistical model which can probabilistically discriminate malicious and benign accounts.
- Later, the derived statistical model is used in conjunction with the real accounts behavioral features to classify those accounts are either malicious or benign probabilistically.

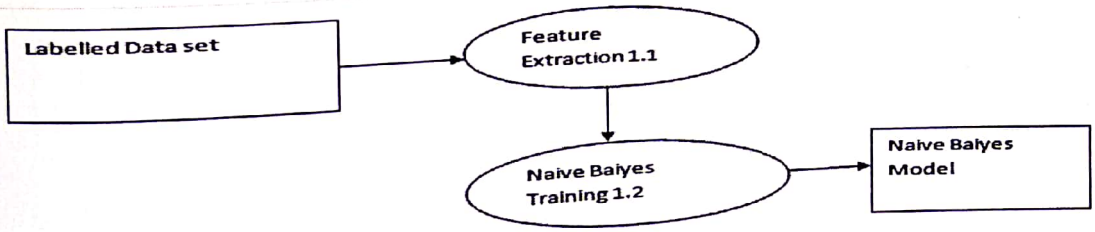

Fig 5.6: The level 01 data-flow diagram.

5.6 Sequence Diagram of the system

Sequence diagram for the two phases in system

Training phase

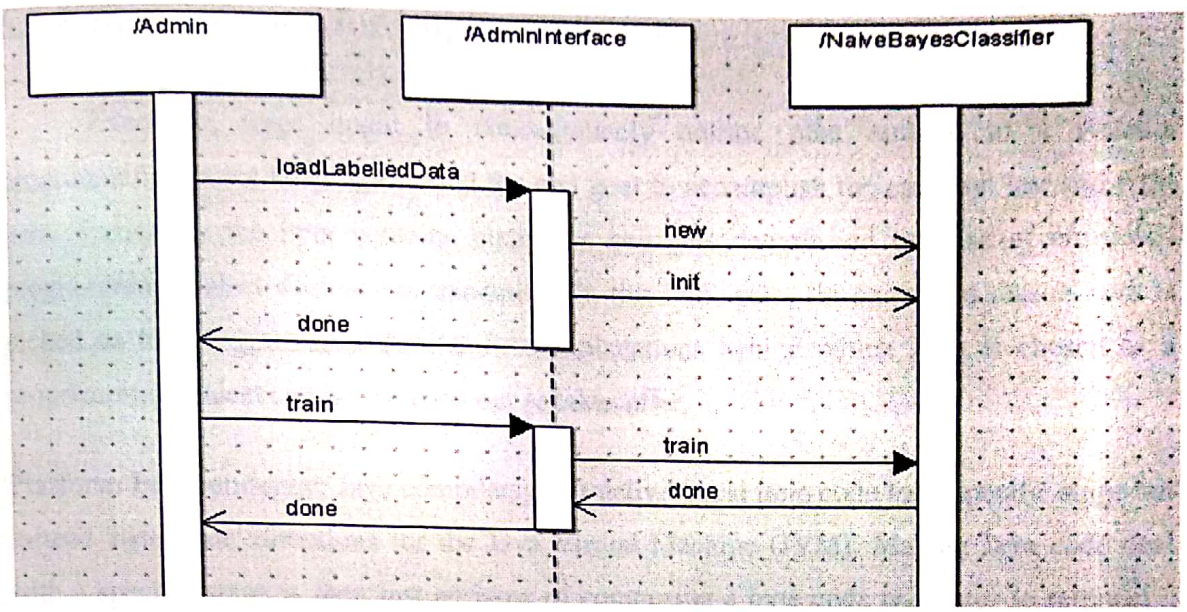

Fig 5.7: Sequence diagram for Training-phase.

Classification phase

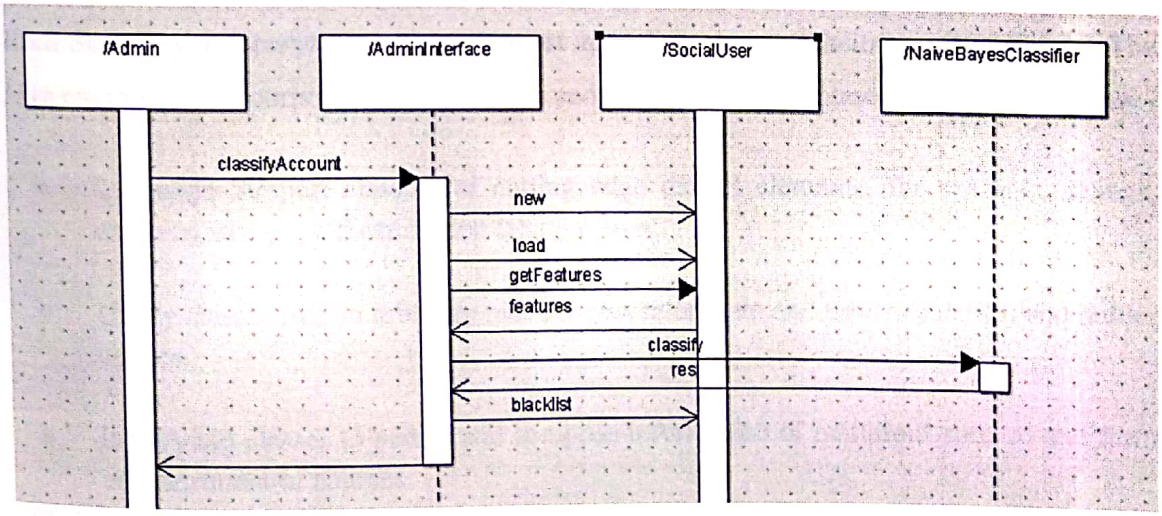

Fig 5.8: Sequence diagram for classification-phase.

CHAPTER 6

IMPLEMENTATION

6.1 Language used for implementation

Execution stage ought to consummately outline plan archive in a suitable programming dialect keeping in mind the end goal to accomplish the essential last and right item. Frequently the item contains blemishes and gets demolished because of erroneous programming dialect decided for execution. In this anticipate, for execution reason Java is picked as the programming dialect. Few explanations behind which Java is chosen as a programming dialect can be sketched out as takes after.

Platform Independence: Java compilers don't deliver local item code for a specific stage but instead 'byte code' directions for the Java Virtual Machine (JVM). Making Java code deal with a specific stage is then just an issue of composing a byte code translator to re-enact a JVM. What this all methods is that the same assembled byte code will run unmodified on any stage that backings Java.

Objects Orientation: Java is an unadulterated item situated dialect. This implies everything in a Java system is an article and everything is slipped from a root object class.

Rich Standard Library: One of Java's most appealing elements is its standard library. The Java environment incorporates many classes and strategies in six noteworthy practical zones:-

- Language Support classes for cutting edge dialect elements, for example, strings, exhibits, strings, and exemption taking care of.
- Utility classes like an arbitrary number generator, date and time capacities, and holder classes.
- Input/yield classes to peruse and compose information of numerous sorts to and from an assortment of sources.
- Networking classes to permit between PC correspondences over a neighborhood system or the Internet.
- Abstract Window Toolkit for making stage autonomous GUI applications.

- Applet is a class that gives you a chance to make Java programs that can be downloaded and keep running on a customer program.

Applet Interface: notwithstanding having the capacity to make remain solitary applications, Java engineers can make programs that can download from a site page and keep running on a customer program.

Java Collection: Java does not oblige software engineers to expressly free powerfully distributed memory. This makes Java programs less demanding to compose and less inclined to memory blunders.

Swing: Swing was produced to give a more refined arrangement of GUI segments than the prior Abstract Window Toolkit.

6.2 UPDATING FEATURES TO DATA-BASE

- Whenever user login , F1 feature is incremented

```
query = "update userprofile set F1=F1+1 where userid='" + userName + "'";
db.executeUpdate(query);.
```

Whenever user adds Friend , F2 feature is incremented.

```
String q = "update userprofile set F2=F2+1 where userid='" + uname + "'";
db.executeUpdate(q);
```

- Whenever user participates in online promotion virtual currency is added to user account.

```
String q = "update userprofile set virtualcurrency=virtualcurrency+50 where userid='" +
uname + "'"
db.executeUpdate(q);
q = "update budget set remaincount=remaincount-1 where eventid=1";
db.executeUpdate(q);
q = "update userprofile set F5=F5+50 where userid='" + uname + "'";
db.executeUpdate(q);
```

- Whenever user purchases product following happens.

```
if (vcavail>ivcvalue) {

int bal = vcavail-ivcvalue;

query = "update userprofile set virtualcurrency="+bal + " where userid='" + uname +"'";

db.executeUpdate(query);

int frombank=productprice-ivcvalue;

query = "update userprofile set F7=F7+"+frombank + " where userid='" + uname +"'";

db.executeUpdate(query);

query = "update userprofile set F8=F8+"+ivcvalue + " where userid='" + uname +"'";

db.executeUpdate(query);
}
```

```
else
{
 query = "update userprofile set F7=F7+"+productprice + " where userid='" + uname +"'";
db.executeUpdate(query);
}
query = "update userprofile set F3=F3+1 where userid='" + uname + "'";
db.executeUpdate(query);
query = "update userprofile set F4=F4+"+ rechfree+ " where userid='" + uname + "'";
db.executeUpdate(query);
 query = "update userprofile set virtualcurrency=virtualcurrency+"+ rechfree + " where
userid='" + uname + "'";
db.executeUpdate(query);
query = "update userprofile set F6=F6+"+ productprice + " where userid='" + uname + "'";
db.executeUpdate(query);
}
```

CHAPTER 7

TESTING

Software Testing is the process of trying to discover every conceivable fault or weakness in a work product. The purpose of testing is to discover errors. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are several types of testing. Each test type addresses a specific testing requirement.

7.1 Types of testing

Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results. Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Integration Testing

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components. Software integration testing is the incremental integration testing of two or more integrated software components on a single

platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications.

e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Functional Testing

Functional testing provides systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is cantered on the following items:

Valid Input	: Identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
System/procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

7.2 UNIT TESTING

Table 7.1: Unit testing for the modules

Classes integrated	Tests done	Remarks
Class: **AdminInterface**	Class tested for functionalities on extracting features from database and calling on NaiveBayesClassifer	Success
Class: **NaiveBayesClassifer**	Class tested to check training and classification works on NaiveBayesClassifier	Success
Class: **SocialUser**	Class tested to check if social user does any operation the corresponding features are updated	Success

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application.

Test Results: All test cases mentioned above passed successfully, No defects encountered.

7.3 Validation Testing

Table 7.2: Validation testing for the modules

Functionality to be tested	Input	Tests done	Remarks
Working of Front-End	User interaction with help of a mouse and keyboard	Appropriate forms open when buttons are clicked	Success
Working of training	User has to upload training dataset and train	Training is done and naive Bayes model is created	Success
Working of classificaiton	The account features available in Database	Based on features account is classified and malicious account is blocked	Success

At the culmination of integration testing, software is completed and assembled as a package. Interfacing errors are uncovered and corrected. Validation testing can be defined in many ways. Here the testing validates the software function in a manner that is reasonably expected by the customer.

Test Results: All test cases mentioned above passed successfully, No defects encountered.

CHAPTER 8

EXPERIMENTAL ANALYSIS

The extensive evaluation of ProGuard which focuses on the overall detection accuracy, the importance of each feature, and the correlation among these features. For this evaluation, we used totally 56,000 accounts whose entire dataset is divided into 28,000 malicious accounts and 28,000 benign accounts. Such data serve as a well-balanced data set for training a statistical classifier.

Detection Accuracy

We have used the normalized Random Forest (RF) as the statistical classifier for ProGuard and evaluated its detection accuracy. RF classifier [20] is an ensemble of unproved classification trees, which is trained over bootstrapped samples of the original data and the prediction is made by aggregating majority vote of the ensemble. In order to avoid the bias caused by the selection of specific training set, we also performed 10-fold cross-validation. Specifically, the entire dataset is partitioned to 10 equal-size sets (i.e., 10-folds); then iteratively 9-folds are used for training and the remaining 1-fold is adopted for testing.

The RF classifier was trained with 3000 trees and randomly sampled 4 features for each of tree splitting [21]. The receiver operating characteristic (ROC) that characterizes the overall detection performance of ProGuard is presented in Fig 3. The experimental results have shown that ProGuard can achieve high detection accuracy.

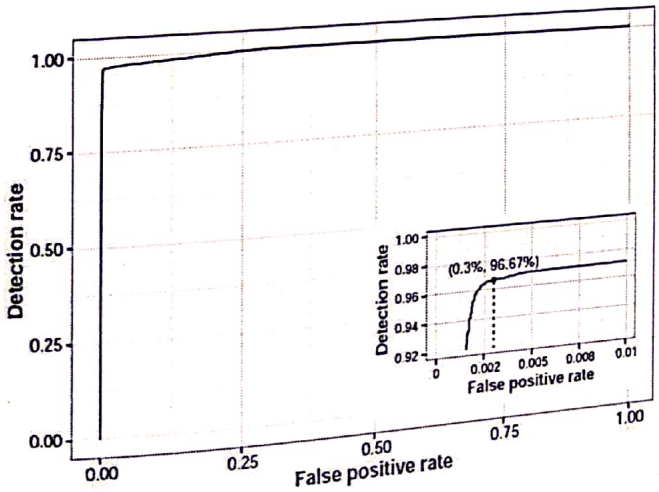

Fig 8.1: ROC curve on 8 features

Given the false positive rate of 0.3%, ProGuard can accomplish a high detection rate of 96.67%.In practice; alternative statistical classifiers might be adopted to render new performance benefits such as scalability.

Therefore, we also evaluate how ProGuard performs when alternative classifiers are used. As a means towards this end, we used Support Vector Machine (SVM) [22] and Gradient-Boosted Tree [23] to repeat our experiments.

Specifically, we used 10- fold cross validation for each of classifiers and calculated the area under the ROC curve (AUC) [24], a widely used measure of quality of supervised classification models, which is equal to the probability that a randomly chosen sample of malicious accounts will have a higher estimated probability of belonging to malicious accounts than a randomly chosen sample of benign accounts. Since AUC is cutoff-independent and values of AUC range from 0.5 (no predictive ability) to 1.0 (perfect predictive ability), a higher AUC of a classifier indicates the better prediction performance, irrespective of the cutoff selection.

Feature Importance and Correlation

We investigated the relative importance of the proposed features in the context of Random Forest classifier, which has accomplished the best detection accuracy according to our experiments. We employed the variable importance of each feature to the Random Forest classification model using permutation test [21]. The variable importance for each feature is computed by mean decrease in accuracy, which is defined as a prediction error rate after permuting an each feature [21].

The rank of features based on the variable importance is shown in Table below. Specifically, the ratio of active days (Feature 1), the average recharge amount of virtual currency (Feature 4), and the percentage of expenditure from banks (Feature 7) represent the most significantly for detection. It is worth noting that these top three features cover three complementary aspects including the general behaviors, currency collection, and currency usage that guide the feature design.

Table 8.1: Feature importance rank of ProGuard

Rank	Variable importance
Feature 1	465.4
Feature 4	349.9
Feature 7	246.6
Feature 2	61.31
Feature 5	56.91
Feature 8	52.17
Feature 6	46.44
Feature 3	35.63

We also performed the correlation among various features, where the correlation implies the extent to which a feature might be redundant given other features. Two widely-adopted methods have been used in our experiments. First, the upper triangular of correlation matrix is carried out for discovering if a pair of strongly correlated features appear within the features, where each column in the upper triangular matrix represents the Pearson's r correlation coefficient [25] of a pair of two distinct features.

Most of features are not strongly correlated one to each other (i.e., Pearson's correlation coefficient jrj _ 0:9).For example, a pair of two features, Feature 1 (The Ratio of Active Days) and Feature 8 (The Percentage of Expenditure as Gifts) represents that the highest negative correlation score is 0.07 and the highest positive correlation between Feature 4 (The Average Recharge Amount of Virtual Currency) and Feature 6 (The Total Amount of Expenditure) is 0.82.

Next, we analyzed Principal Component Analysis (PCA), which can be used to evaluate variable correlation in regard to the variance of the data [26]. Figure 14 shows the experimental result on PCA variables factor map [27]. In the variable factor map, each of features is expressed as an arrow and the angle between the two arrows of features implies the correlation among the respective features on the third and fourth principal components (PC).

CHAPTER 9

INTERPRETATION OF RESULT

The following snapshots define the results or outputs that we will get after step by step execution of all the modules of the system.

Register an account in online shopping site.

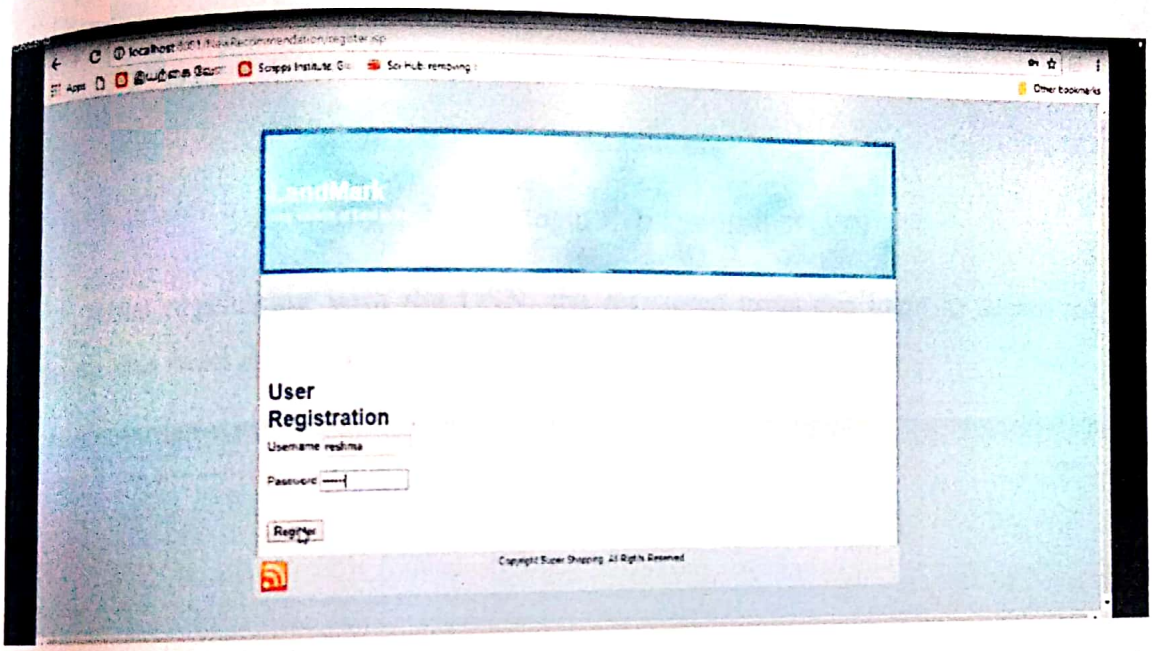

Fig 9.1: User registration web-template of OSN.

This web page interface allows the aspirants to register to the online shopping site by specifying the user credentials-username, password.

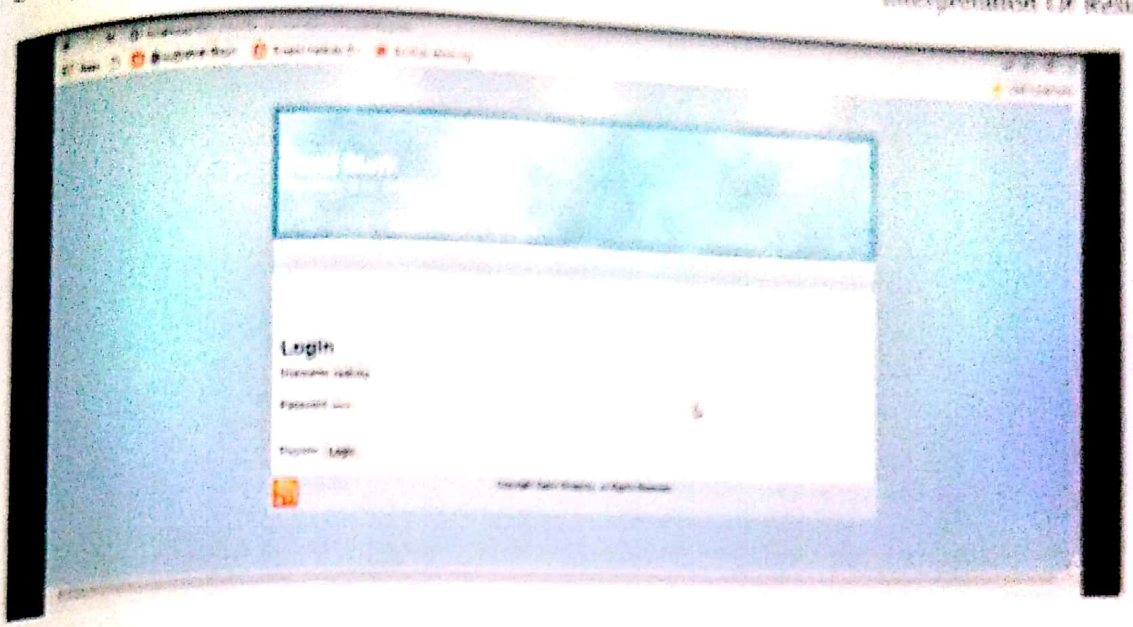

Fig 9.2: User login web-template of OSN.

After registering with the OSN, the registered users can login to access the online shopping site from the web-page interface shown.

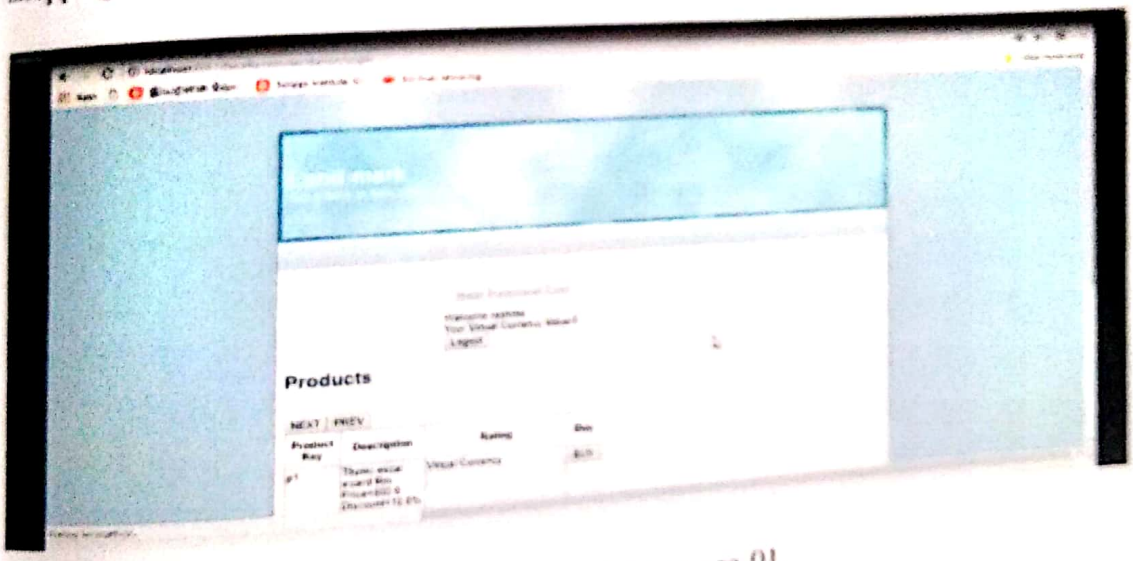

Fig 9.3: User home page-01.

After the successful login, users can see the home page of online shopping site listed and displayed with many products and their detailed descriptions.

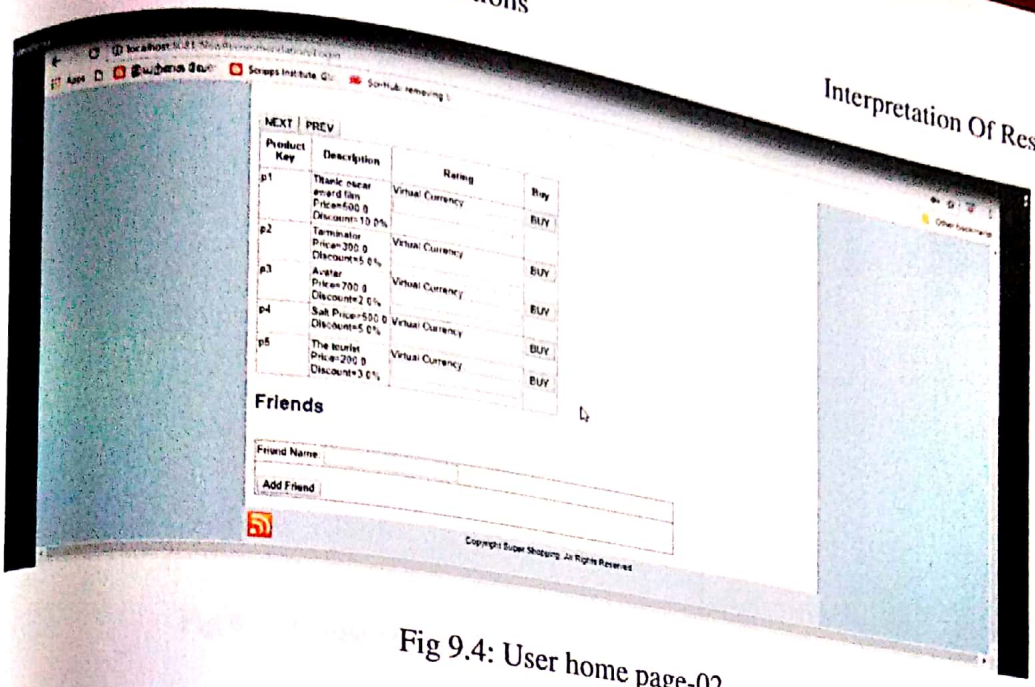

Fig 9.4: User home page-02.

Users can browse the products by pressing prev and next button and can make online purchases. This interface also provides a flexibility to connect with friends.

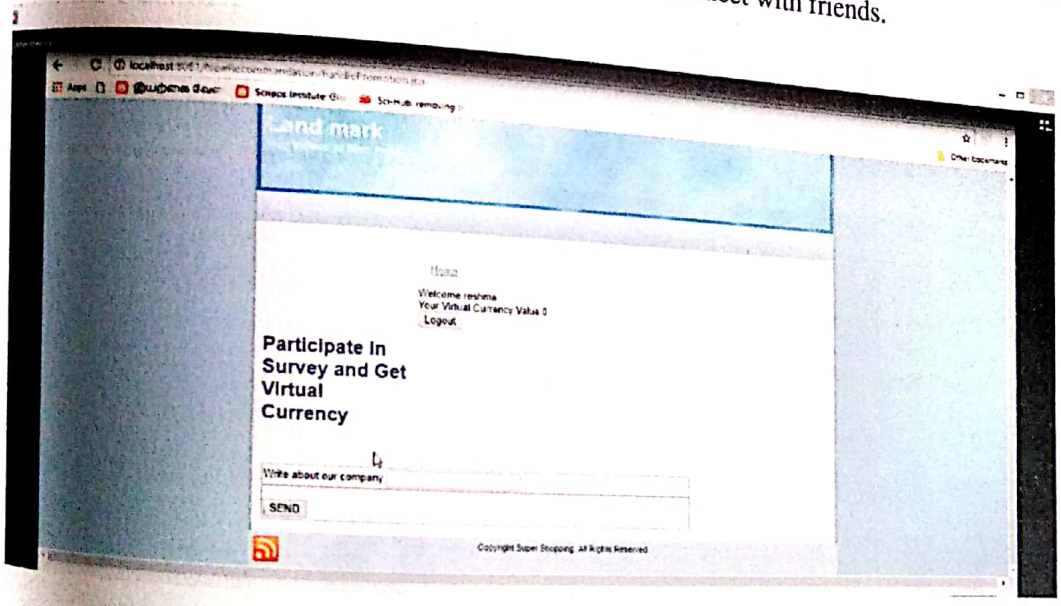

Fig 9.5: Home promotional event interface.

Users can participate in online promotional events and can get virtual currency through this we interface.

CHAPTER 10

CONCLUSION AND FUTURE SCOPE

This project presents a novel system, ProGuard, to automatically detect malicious OSN accounts that participate in online promotion events. ProGuard leverages three categories of features including general behaviour, virtual-currency collection, and virtual-currency usage. Proof of concept system was implemented and the malicious accounts were blocked from participating in online social promotions.

10.1 Limitations of the project

All current features rely on coarse-grained information that minimizes privacy concerns, which may foster the deployment of the proposed system in a detection-as-service model.

10.2 Future enhancement

Despite the fact that ProGuard can effectively detect malicious accounts used for collecting virtual currency from online promotion activities, it is not designed for detecting malicious accounts used for transferring and laundering virtual currency. Extending ProGuard to include such detection capabilities falls into our future work.

BIBLIOGRAPHY

[1] Y. Wang and S. D. Mainwaring, "Human-currency interaction: learning from virtual currency use in china," in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2008, pp. 25–28.

[2] J. S. Gans and H. Halaburda, "Some economics of private digital currency," Rotman School of Management Working Paper, no. 2297296, 2013.

[3] X. Hu, J. Tang, and H. Liu, "Online social spammer detection," in Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. AAAI, 2014, pp. 59–65.

[4] ——, "Leveraging knowledge across media for spammer detection in microblogging," in Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. ACM, 2014, pp. 547–556.

[5] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Detecting automation of twitter accounts: Are you a human, bot, or cyborg?" IEEE Transactions on Dependable and Secure Computing, vol. 9, no. 6, pp. 811–824, 2012.

[6] Z. Chu, S. Gianvecchio, A. Koehl, H. Wang, and S. Jajodia, "Blog or block: Detecting blog bots through behavioral biometrics," Computer Networks, vol. 57, no. 3, pp. 634–646, 2013.

[7] S. Fakhraei, J. Foulds, M. Shashanka, and L. Getoor, "Collective spammer detection in evolving multi-relational social networks," in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2015, pp. 1769–1778.

[8] Y.-R. Chen and H.-H. Chen, "Opinion spammer detection in web forum," in Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2015, pp. 759–762.

[9] F. Wu, J. Shu, Y. Huang, and Z. Yuan, "Social spammer and spam message co-detection in microblogging with social context regularization," in Proceedings of the 24th ACM International on Conference on InformationandKnowledgeManagement. ACM,2015,pp.1601–1610.

[10] Z. Miller, B. Dickinson, W. Deitrick, W. Hu, and A. H. Wang, "Twitter spammer detection using data stream clustering," Information Sciences, vol. 260, pp. 64–73, 2014.

[11] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao, "Detecting and characterizing social spam campaigns," in Proceedings of the 10th ACM SIGCOMM conference on Internet measurement. ACM, 2010, pp. 35–47.

[12] S. Lee and J. Kim, "Warningbird: Detecting suspicious urls in twitter stream." in NDSS, vol. 12, 2012, pp. 1–13.

[13] C. Yang, R. C. Harkreader, and G. Gu, "Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers," in International Workshop on Recent Advances in Intrusion Detection. Springer, 2011, pp. 318–337.

[14] A. Abdallah, M. A. Maarof, and A. Zainal, "Fraud detection system: A survey," Journal of Network and Computer Applications, vol. 68, pp. 90 – 113, 2016.

[15] J. West and M. Bhattacharya, "Intelligent financial fraud detection: A comprehensive review," Computers & Security, vol. 57, pp. 47 – 66, 2016.

[16] D. Olszewski, "Fraud detection using self-organizing map visualizing the user profiles," Knowledge-Based Systems, vol. 70, pp. 324 – 334, 2014.

[17] C.-C. Lin, A.-A. Chiu, S. Y. Huang, and D. C. Yen, "Detecting the financial statement fraud: The analysis of the differences between data mining techniques and experts' judgments," Knowledge-Based Systems, vol. 89, pp. 459 – 470, 2015.

[18] C. S. Throckmorton, W. J. Mayew, M. Venkatachalam, and L. M. Collins, "Financial fraud detection using vocal, linguistic and financial cues," Decision Support Systems, vol. 74, pp. 78 – 87, 2015.

[19] Z. Afzal, M. J. Schuemie, J. C. van Blijderveen, E. F. Sen, M. C. Sturkenboom, and J. A. Kors, "Improving sensitivity of machine learning methods for automated case identification from free-text electronic medical records," BMC medical informatics and decision making, vol. 13, no. 1, p. 1, 2013.

[20] L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5–32, 2001

VISVESVARAYA TECHNOLOGY UNIVERSITY

"JNANA SANGAMA", BELGAUM-590014

A

PROJECT REPORT

On

"*ProGuard:* Detecting Malicious Accounts in Social-Network- Based Online Promotions"

Submitted in the partial fulfillment of the requirement for the award of Degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

By

AVINASH S	1GV15CS401
JANCY SWETHA M	1GV15CS404
MATHEW SAJJAN S	1GV14CS027
MOHAMMED ASHRAF V A	1GV14CS029

Under the guidance of

Mrs. SOPHIA S

Asst. Prof.,Dept. of CSE.

2017-2018

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DR.T.THIMMAIAH INSTITUTE OF TECHNOLOGY,

Oorgaum, Kolar Gold Fields-563 122