A

Major Project

On

FRAUD DETECTION OF MOBILE APPS

(Submitted in partial fulfilment of the requirements for the award of Degree)

BATCHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

P. Shyamala	(187R1A0546)
P. Vidya	(187R1A0543)
R.Kiran Franklin	(187R1A0547)

Under the Guidance of

J.SRI VIDYA

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGCAUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGC Act.1956, Kandlakoya

(V), Medchal Road, Hyderabad-501401.

2018-22

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**Fraud detection of Mobile Apps**" being submitted by **P.SHYAMALA (187R1A0546) ,P.VIDYA (187R1A0543) &R.KIRAN FRANKLIN (187R1A0547)** in partial fulfilment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

J.Srividya
Assistant Professor
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
Head Of Department

EXTERNAL EXAMINER

Submitted for viva voice Examination held on_____

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **J.Srividya,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement through out the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) **Mr. A. Uday Kiran, Mr. J. Narasimha Rao, Dr. T.S. Mastan Rao, Mrs. G. Latha, Mr. A. Kiran Kumar,** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

P.Shyamala(187R1A0546)
P.Vidya (187R1A0543)
R.KiranFranklin(187R1A0547)

ABSTRACT

Ranking fraud in the mobile App market refers to fraudulent or deceptive activities which have a purpose of bumping up the Apps in the popularity list. Indeed, it becomes more and more frequent for App developers to use shady means, such as inflating their Apps sales or posting phony App ratings, to commit ranking fraud. While the importance of preventing ranking fraud has been widely recognized, there is limited understanding and research in this area.

To this end, in this paper, we provide a holistic view of ranking fraud and pro-pose a ranking fraud detection system for mobile Apps. Specifically, we first propose to accurately locate the ranking fraud by mining the active periods, namely leading sessions, of mobile Apps. Such leading sessions can be leveraged for detecting the local anomaly instead of global anomaly of App rankings.

Furthermore, we investigate three types of evidences, i.e., ranking based evidences, rating based evidences and review based evidences, by modelling Apps ranking, rating and review behaviour's through statistical hypotheses tests. In addition, we propose an optimization based aggregation method to integrate all the evidences for fraud detection. Finally, we evaluate the proposed system with real-world App data collected from the iOS App Store for a long time period. In the experiments, we validate the effectiveness of the proposed system, and show the scalability of the detection algorithm as well as some regularity of ranking fraud activities.

LIST OF FIGURES

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1.INTRODUCTION:

Over the last few years the number of mobile Apps has been growing on a very large scale. At the end of April 2016 there is number of more than 3.4 million Applications at Apples App store and Google Play. Different App stores launched their leader board on daily basis to inspire the development of mobile Apps which displays the chart rankings of most popular Apps. In fact for promoting mobile Apps, leader board of apps is the most important ways in the market. An app ranking at the top on the leader board ultimately leads to a large number of downloads and million dollars in revenue. This results in exploring of different ways by the App developers like organizing promotional drives to advertise their Apps in order to get top position in App leader boards. The very recent trend followed in market by the corrupt App developers for bumping up of an App is to use deceptive means to intentionally boost their apps. Lastly, the chart rankings on a App store are also manipulated. This is usually implemented by using so-called internet bots or human water armies to raise the App downloads, ratings and reviews in a very little time. Venture Beat is an article that reported, using ranking manipulation when an App was promoted, in Apples top free leader board it could be push forward from number 1,800 to the upmost 25 and new users more than 50,000-100,000 could be acquired within a couple of days.

In reality, such ranking fraud leads to great concerns to the industry of mobile App. For example, App developers who commit ranking fraud in the App store, Apple has warned of cracking down on them. As per the observation the mobile apps does not always ranked high in the leader boards, in fact in some leading events only. Collection of leading events of mobile Apps ultimately leads to different leading sessions. Thus, detecting ranking fraud of mob Apps happens in leading sessions and perhaps the process of detecting ranking fraud is done within the leading session of the mobile Apps. Especially, on the basis of historical ranking records of the mobile apps this paper proposes a simple and effective algorithm for the recognition of the leading sessions of each mobile App. This is one of the evidence collected from historical ranking records of apps against fraud. Moreover, there are two more types of fraud evidences proposed on the basis of Apps rating and review history, which provides few anomaly patterns from Apps historical rating and review records. Additionally, system propose an unsupervised evidence-aggregation

method to combine these three types of evidences collected for the assessment of credibility of leading sessions from mobile Apps.

1.2 Aim:

The mobile industry is growing rapidly, subsequently the number of mobile apps coming in the market is also increasing. As there are many apps available in market users are confused while downloading the apps for their use.

They check the daily app leader boards foe selecting app. But few fraudulent app developers are using shady means for bumping up their apps on the leader board in order to get revenue.So detect such fraud apps we develop a system based on evidences i.e.Ranking fraud detection using opinion mining for mobile apps.

2. LITERATURE SURVEY

2. LITERATURE SURVEY

A Flexible Generative Model For Preference Aggregation:

Many areas of study, such as information retrieval, collaborative filtering, and social choice face the preference aggregation problem, in which multiple preferences over objects must be combined into a consensus ranking. Preferences over items can be expressed in a variety of forms, which makes the aggregation problem difficult. In this work we formulate a flexible probabilistic model over pairwise comparisons that can accommodate all these forms. Inference in the model is very fast, making it applicable to problems with hundreds of thousands of preferences. Experiments on benchmark datasets demonstrate superior performance to existing methods.

•Getjar Mobile Application Recommendations With Very Sparse Datasets:

The Netflix competition of 2006 [2] has spurred significant activity in the commendations field, particularly in approaches using latent factor models [3,5,8,12]However, the near ubiquity of the Netflix and the similar Movie Lens datasets1may be narrowing the generality of lessons learned in this field. At GetJar, our goal is to make appealing recommendations of mobile applications (apps).For app usage, we observe a distribution that has higher kurto sis (heavier head and longer tail) than that for the aforementioned movie datasets. This happens primarily because of the large disparity in resources available to app developers and the low cost of app publication relative to movies.

In this paper we compare a latent factor (PureSVD) and a memory based model with our novel PCA-based model, which we call Eigen app. We use both accuracy and variety as evaluation metrics. PureSVD did not perform well due to its reliance on explicit feedback such as ratings, which we do not have. Memory-based approaches that perform vector operations in the original high dimensional space over-predict popular apps because they fail to capture the neighbourhood of less popular apps. They have high accuracy due to the concentration of mass in the head, but did poorly in terms of variety of apps exposed. Eigenapp, which exploits neighbourhood information in low dimensional spaces, did well both on precision and variety, underscoring the importance of dimensionality reduction to form quality neighbourhoods in high kurtosis distributions.

• Detecting Spam Web Pages Through Content Analysis:

In this paper, we continue our investigations of ”web spam”: the injection of artificially created pages into the web in order to influence the results from search engines, to drive traffic to certain pages for fun or profit. This paper considers some previously undescribed techniques for automatically detecting spam pages, examines the effectiveness of these techniques in isolation and when aggregated using classification algorithms.

Spotting Opinion Spammers Using Behavioural Boot prints:

Opinionated social media such as product reviews are now widely used by individuals and organizations for their decision making. However, due to the reason of profit or fame, people try to game the system by opinion spamming(e.g., writing fake reviews) to promote or to demote some target products. In recent years, fake review detection has attracted significant attention from both the business and research communities. However, due to the difficulty of human labelling needed for supervised learning and evaluation, the problem remains to be highly challenging. This work proposes a novel angle to the problem by modelling spamicity as latent. An unsupervised model, called Author3 Spamicity Model (ASM), is proposed. It works in the Bayesian setting, which facilitates modelling spamicity of authors as latent and allows us to exploit various observed behavioural footprints of reviewers. The intuition is that opinion spammers have different behavioural distributions than non-spammers. This creates a distributional divergence between the latent population distributions of two clusters: spammers and non-spammers. Model inference results in learning the population distributions of the two clusters. Several extensions of ASM are also considered leveraging from different priors. Experiments on a real life Amazon review dataset demonstrate the effectiveness of the proposed models which significantly outperform the state-of-the-art competitors.

•Unsupervised Rank Aggregation With Domain- Specific Expertise:

Consider the setting where a panel of judges is repeatedly asked to (partially) rank sets of objects according to given criteria, and assume that the judges' expertise depends on the objects' domain. Learning to aggregate the irrankings with the goal of producing a better joint ranking is a fundamental problem in many areas of Information Retrieval and Natural Language Processing, amongst others. However, supervised ranking data is generally

difficult to obtain, especially if coming from multiple domains. Therefore, we propose a framework for learning to aggregate votes of constituent rankers with domain specific expertise without supervision. We apply the learning framework to the settings of aggregating full rankings and aggregating top-k lists, demonstrating significant improvements over a domain-agnostic baseline in both cases.

3. SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

Most of the data which is gathered from web sources is unstructured in nature. The huge amount of data which is an significant source to analyze and reuse for further research applications can be processed by humans up to some extent even with difficulty. But it is difficult for us write a learning algorithm to process and analyze the data that we have gathered. There are several feature extraction methods available for the procedure of bringing a list of words together to enhance the analysis of the information gathered. This emphasizes the working nature of recommendation engine to provide the better recommendation of items for the customers.

.

3.1.EXISTING SYSTEM

In existing learning methodologies studies of a recent trend, instead of relying on traditional marketing solutions, shady App developers resort to some fraudulent means to deliberately boost their Apps and eventually manipulate the chart rankings on an App store.

This is usually implemented by using so-called “bot farms” or “human water armies” to inflate the App downloads, , ratings and reviews in a very short time.

3.1.2.DISADVANTAGES:

- When an app was promoted with help of ranking manipulation it could be top in leader board and more new users could be purchased that product effect other App reputation.

- The encouraging results show that the Existing algorithms are less effective for feature selection tasks of online applications.CMRTC 9

3.2 PROPOSED SYSTEM:

We are developing a ranking fraud detection system for mobile Apps. Specifically, we first showed that ranking fraud happened in leading sessions and provided method for mining leading sessions for each App from its historical ranking records.

Then, we identified ranking based evidences, rating based evidences and review based evidences for detecting ranking fraud by using Mining Leading Sessions.

3.2.1.ADVANTAGES:

- Detecting ranking fraud of mobile Apps is actually to detect ranking Fraud within leading sessions of mobile Apps.
- Contribute the methods to enhance Fraud detection to improve the discovery of fraud in mobile apps.

3.3 SYSTEM SPECIFICATION:

3.3.1Software requirements

Operating System : Windows

Front End : PYTHON

Back End : MYSQL

IDE : Eclipse

3.3.2Hardware requirements

Processor : Pentium4

Hard disc : 500GB

RAM : 4GB

System with all standard accessories like monitor, keyboard, mouse, etc.

3.4FEASIBILITY STUDY

3.4.1Technical feasibility:

Surveying the particular probability is the trickiest bit of a believability consider. This is in light of the fact that, starting at the present moment, not a lot of point by point layout of the system, making it difficult to get to issues like execution, costs on (by excellence of the kind of development to be passed on) et cetera. Different issues must be

considered while doing a particular examination. Grasp the differing progressions required in the proposed system.

3.4.2Operational feasibility:

Proposed wanders are profitable just if they can be changed into information systems that will meet the affiliations working necessities. Simply communicated, this trial of probability asks with reference to whether the structure will work when it is made and presented. Is there sufficient help for the wander from organization from customers? In case the present structure is particularly cherished and used to the extent that individuals won't have the ability to see purposes behind change, there may be resistance. Are the present business methodologies qualified to the customer? If they are not, Users may welcome a change that will accomplish a more operational and supportive systems.

3.4.3Economical feasibility:

The electronic structure manages the present existing system's data stream and technique absolutely and should make each one of the reports of the manual structure other than a substantial gathering of the other organization reports. It should be filled in as an electronic application with specific web server and database server. Advance a segment of the associated trades happen in different ranges.

4. SYSTEM DESIGN

4.SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

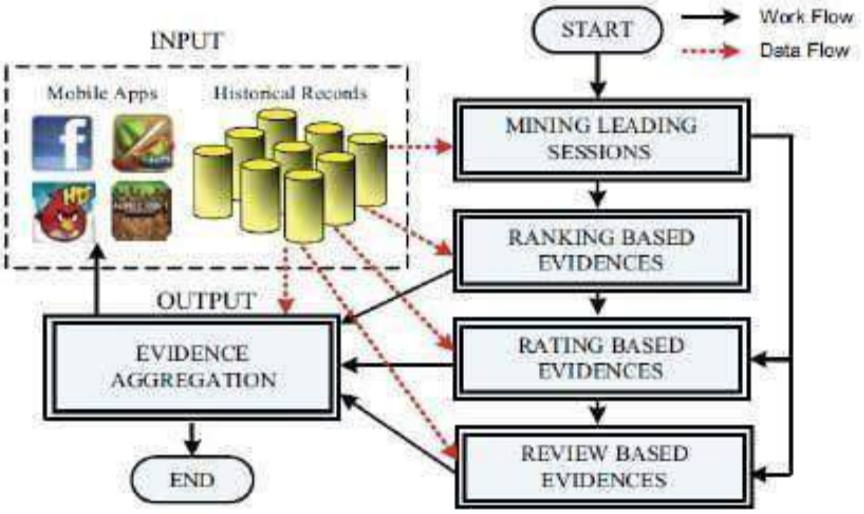

Fig 4.1.System Architecture

4.2 UML Diagrams

UML (Unified Modeling Language) is a standard vernacular for choosing, envisioning, making, and specifying the collectibles of programming structures. UML is a pictorial vernacular used to make programming blue prints. It is in like way used to exhibit non programming structures similarly like process stream in a gathering unit and so forth.UML is not a programming vernacular yet rather instruments can be utilized to make code in different tongues utilizing UML graphs.

4.3 CLASS DIAGRAM

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. The class diagrams are widely used in the modeling of objectoriented systems because they are the

only UML diagrams, which can be mapped directly with object-oriented languages.

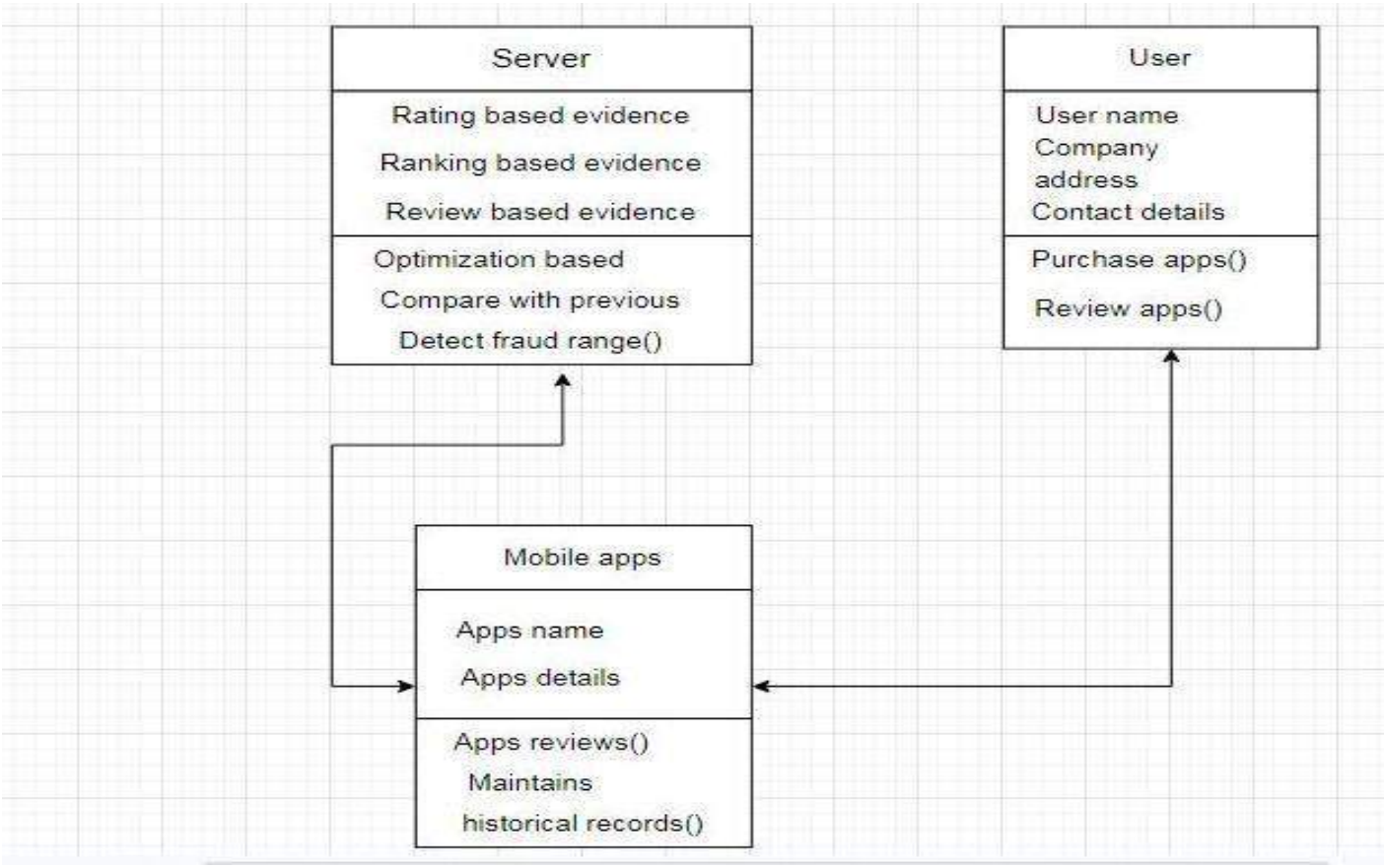

Fig 4.4.Class Diagram

4.4 USECASE DIAGRAM

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as a mail-order product sales and service Web site.

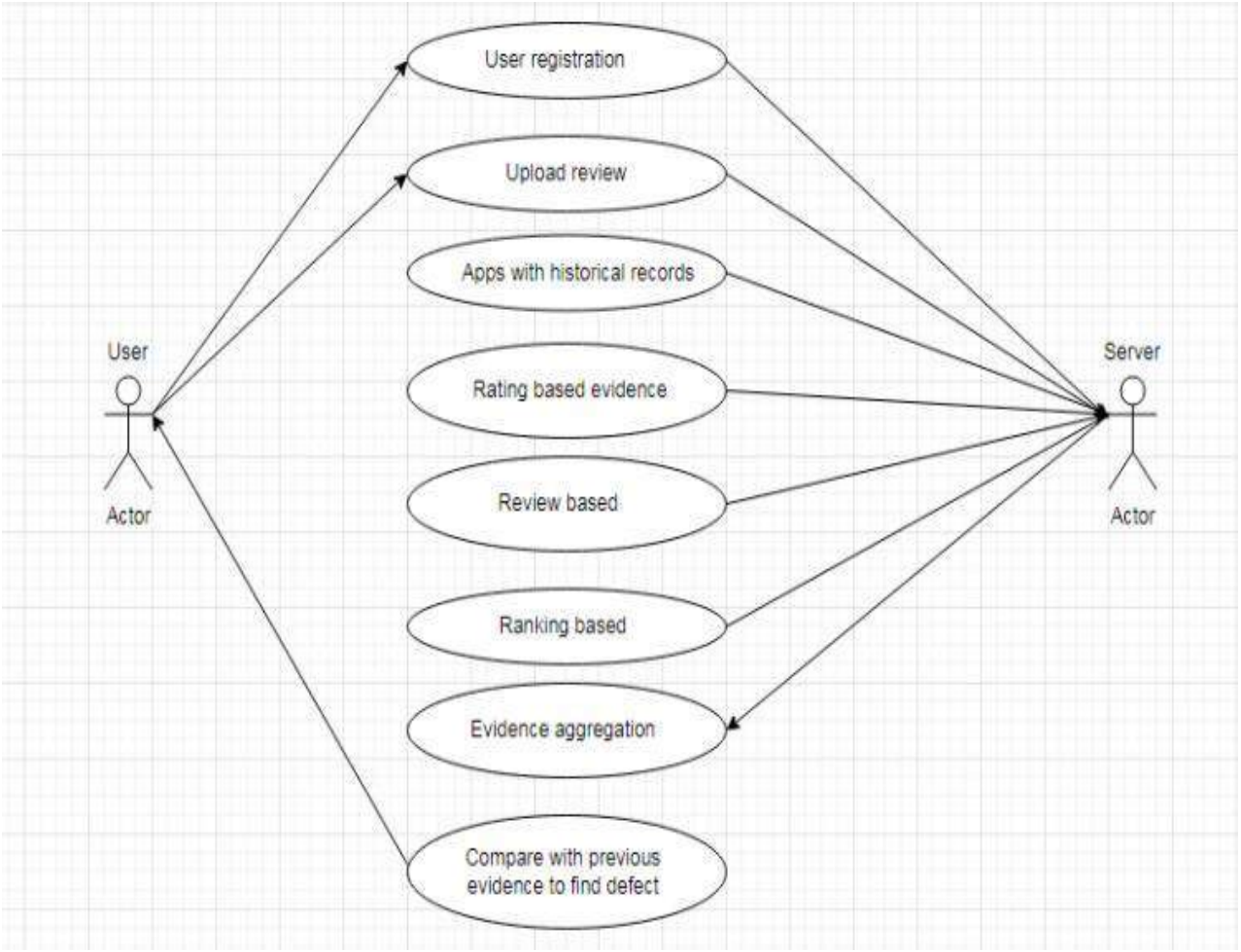

Fig 4.5.UseCase diagram

4.5 SEQUENCE DIAGRAM

A sequence diagram is an interaction diagram. From the name, it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another. Interaction among the components of a system is very important from implementation and execution perspective. Sequence diagram is used to visualize the sequence of calls in a system to perform a specific functionality.

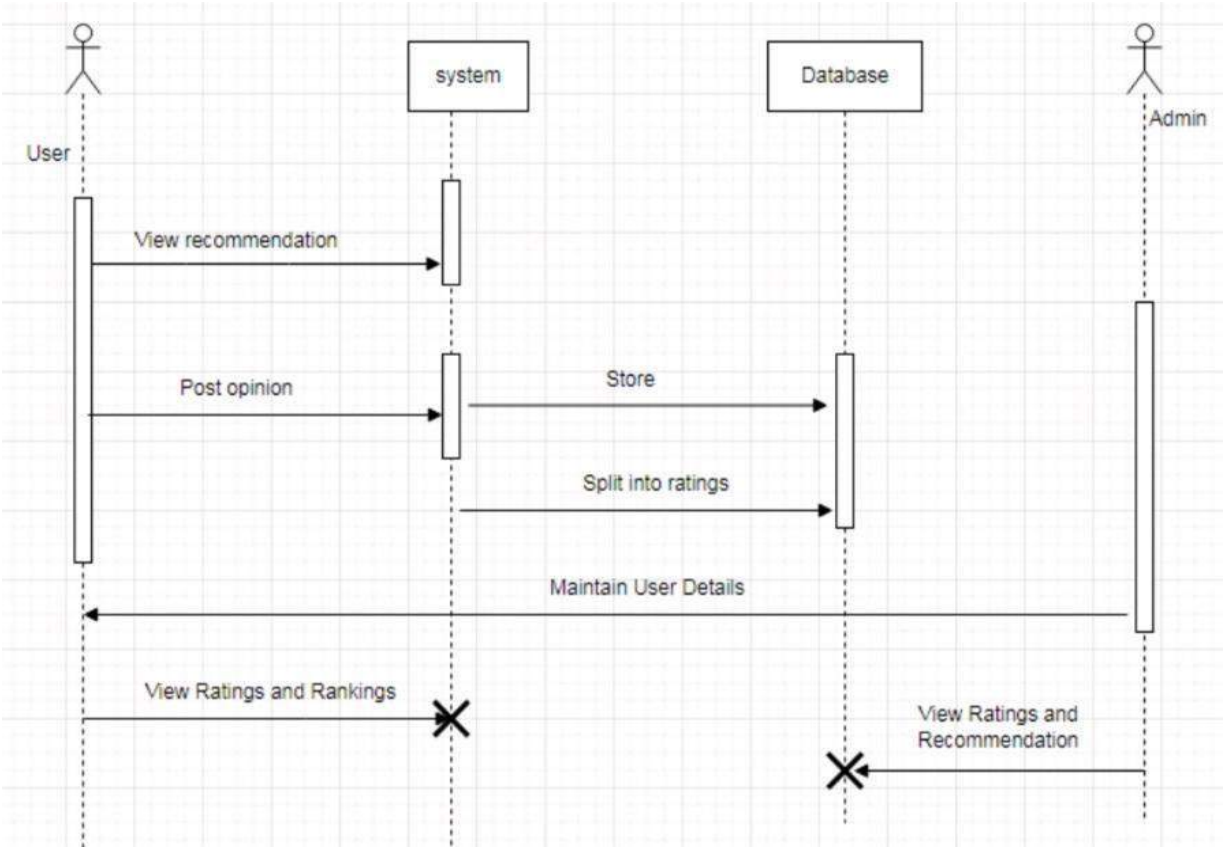

Fig 4.6.Sequence diagram

4.6.ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of work flows of stepwise activities and actions with support for choice, iteration and concurrency.

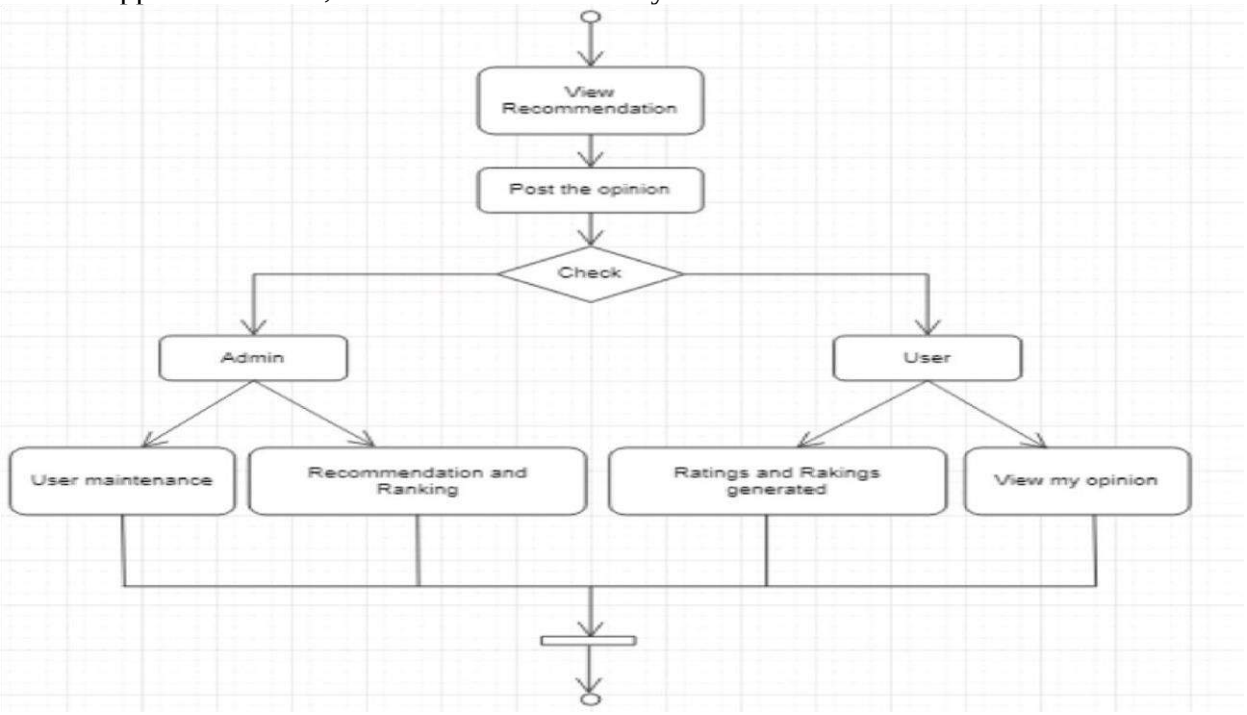

Fig.4.7.Activity Diagram

5.IMPLEMENTATION

5.IMPLEMENTATION

5.1.SAMPLE CODE:

```
from tkinter import * import tkinter from
tkinter import filedialog from tkinter.filedialog
import askopenfilename
import numpy as np
import os
from tkinter import messagebox
import pandas as pd from
datetime import datetime
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
import matplotlib.pyplot as plt
sid = SentimentIntensityAnalyzer()
main = tkinter.Tk()
main.title("FRAUD DETECTION OF MOBILE APPS")
main.geometry("1200x1200") global filename,
dataset,total_apps, fraud_count
def uploadDataset():
global filename text.delete('1.0', END)     filename =
filedialog.askopenfilename(initialdir="Dataset")
text.insert(END,str(filename)+" Dataset Loaded\n\n")
pathlabel.config(text=str(filename)+" Dataset Loaded\n\n")
def processDataset():
global filename, dataset, total_apps     text.delete('1.0', END)
dataset = pd.read_csv(filename,
usecols=["content","score","thumbsUpCount","at","appId"])
   text.insert(END,str(dataset.head()))     unique, count =
np.unique(dataset["appId"],return_counts=True)     total_apps =
len(unique)     height = count     bars = unique     y_pos =
np.arange(len(bars))     plt.bar(y_pos, height)     plt.xticks(y_pos,
bars)
   plt.title("Totals Apps with Reviews, Rating & Rank Count")
plt.xticks(rotation=90)     plt.show()
def getReviewSentiment(review): #calculate review sentiment
   sentiment_dict = sid.polarity_scores(review)
   compound = sentiment_dict['compound']
result = "     if compound >= 0.05 :
result = 'Positive'



   elif   compound   <   0.05   :
result = 'Negative'
   return result

def getTimePeriod(start, end): #calculate time gap between start and next review     fmt
= '%Y-%m-%d %H:%M:%S'
   tstamp1 = datetime.strptime(start, fmt)
tstamp2 = datetime.strptime(end, fmt)
if tstamp1 > tstamp2:
```

```
td = tstamp1 - tstamp2     else:
       td = tstamp2 - tstamp1
   td_mins   =   int(round(td.total_seconds()   /   60))
return td_mins


def fraudDetection(): #function to detect fraud from dataset
global dataset, fraud_count     fraud_count = 0
text.delete('1.0', END)
   app_id = np.unique(dataset["appId"]) #finding all uniques app from review history
dataset = dataset.values     for j in range(len(app_id)):
       old_session = None
old_rating = 0
old_rank = 0        count
= 0        fraud_event = 0
old_review = None
       text.insert(END,"Analysing App: "+app_id[j]+"\n")
       text.update_idletasks()
       for i in range(len(dataset)): #extracting review, rating, ranking from past history dataset
          review = dataset[i,0]
ranking = dataset[i,1]
 ratings = dataset[i,2]
session_time = dataset[i,3]
app_name = dataset[i,4]
          #getting review sentiment whether user is positive or negative and if continuous
positive received from negative then fraud detected
 current_review = getReviewSentiment(review)
 if count == 0 and app_id[j] == app_name:
             old_session = session_time
#getting first session time
old_rating = ratings
old_rank = ranking
old_review = current_review
             count = 1
 elif app_id[j] == app_name and count > 0:
 lead_session = getTimePeriod(old_session, session_time)
#calculating leading session time
             #if mobile app review has less than 3000 second time period gap between old new
review
             #if continuous ranking is increasing from low rank
             #if continous rating increase from low rating and suddenly start receiving
positive review then fraud detected                if lead_session < 3000 and ratings >
old_rating and ranking > old_rank and current_review == 'Positive':
                old_session = session_time
old_rating = ratings
 old_rank = ranking
old_review = current_review
fraud_event = fraud_event + 1
 if fraud_event > 1:
  fraud_count = fraud_count + 1
          text.insert(END,app_id[j]+"Detected                as                Fraud\n\n")
text.update_idletasks()
          print("detected fraud "+str(fraud_event)+" "+app_id[j])
```

```
def graph():
 height = [total_apps,fraud_count]
bars = ["Totals Apps","Fraud App Detected"]
 y_pos = np.arange(len(bars))
plt.bar(y_pos, height)
plt.xticks(y_pos, bars)
   plt.title("Fraud Apps Detection Graph")
   plt.xticks(rotation=90)
   plt.show()

def close():
   main.destroy()

font = ('times', 14, 'bold')
title = Label(main, text='FRAUD DETECTION OF MOBILE APPS')
title.config(bg='DarkGoldenrod1', fg='black')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=5,y=5)
font1 = ('times', 13, 'bold')
uploadButton = Button(main, text="Upload Mobile Reviews Dataset",
command=uploadDataset)
uploadButton.place(x=50,y=100)
uploadButton.config(font=font1)
pathlabel = Label(main)
pathlabel.config(bg='brown', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=560,y=100)

matchButton = Button(main, text="Read & Process Dataset",
command=processDataset) matchButton.place(x=50,y=150)
matchButton.config(font=font1)

fraudButton = Button(main, text="Run Rating, Ranking & Review Based Fraud
Detection", command=fraudDetection)
fraudButton.place(x=50,y=200)
fraudButton.config(font=font1)

graphButton = Button(main, text="Detection Graph", command=graph)
graphButton.place(x=50,y=250)
graphButton.config(font=font1)


exitButton = Button(main, text="Exit", command=close)
exitButton.place(x=50,y=300)
exitButton.config(font=font1)
```

```
font1 = ('times', 12, 'bold')
text=Text(main,height=25,width=100)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set) text.place(x=520,y=150)
text.config(font=font1)


main.config(bg='LightSteelBlue1')
main.mainloop()
```

5.2 MODULES:

5.2.1 Mining Leading Sessions:

In the first module, we develop our system environment with the details of App like an app store. Intuitively, the leading sessions of a mobile App represent its periods of popularity, so the ranking manipulation will only take place in these leading sessions. Therefore, the problem of detecting ranking fraud is to detect fraudulent leading sessions. Along this line, the first task is how to mine the leading sessions of a mobile App from its historical ranking records. There are two main steps for mining leading sessions. First, we need to discover leading events from the Apps historical ranking records. Second, we need to merge adjacent leading events for constructing leading sessions.

5.2.2 Ranking Based Evidences:

In this module, we develop Ranking based Evidences system. By analyzing the Apps historical ranking records, web serve that Apps ranking behaviors in a leading event always satisfy a specific ranking pattern, which consists of three different ranking phases, namely, rising phase, maintaining phase and recession phase. Specifically,in each leading event, an Apps ranking first increases to a peak position in the leader-board (i.e., rising phase), then keeps such peak position for a period (i.e., maintaining phase), and finally decreases till the end of the event (i.e., recession phase).

5.2.3 Rating Based Evidences:

In the third module, we enhance the system with Rating based evidences module.The ranking based evidences are useful for ranking fraud detection. However, sometimes, it is not sufficient to only use ranking based evidences. For example, some Apps created by the famous developers, such as Gameloft, may have some leading events with large values of u1 due to the developers credibility and the word-of-mouth advertising effect. Moreover, some of the legal marketing services, such as limited-time discount, may also result in significant ranking based evidences. To solve this issue,we also study how to extract fraud evidences from Apps historical rating records.

5.2.4 Review Based Evidences:

In this module we add the Review based Evidences module in our system. Besides ratings, most of the App stores also allow users to write some textual comments as App reviews. Such reviews can reflect the personal perceptions and usage experiences of existing users for particular mobile Apps. Indeed, review manipulation is one of the16

most important perspective of App ranking fraud. Specifically, before downloading or purchasing a new mobile App, users often first read its historical reviews to ease their decision making, and a mobile App contains more positive reviews may attract more users to download. Therefore, imposters often post fake reviews in the leading sessions of a specific App in order to inflate the App downloads, and thus propel the Apps ranking position in the leader board.

5.2.5 Evidence Aggregation:

In this module we develop the Evidence Aggregation module to our system. After extracting three types of fraud evidences, the next challenge is how to combine them for ranking fraud detection. Indeed, there are many ranking and evidence aggregation methods in the literature, such as permutation based models score based models and Dempster-Shafer rules . However, some of these methods focus on learning a global ranking for all candidates. This is not proper for detecting ranking fraud for new Apps. Other methods are based on supervised learning techniques, which depend on the labeled training data and are hard to be exploited. Instead, we propose an unsupervised approach based on fraud similarity to combine these evidences.

6.SCREENSHOTS

RESULT:

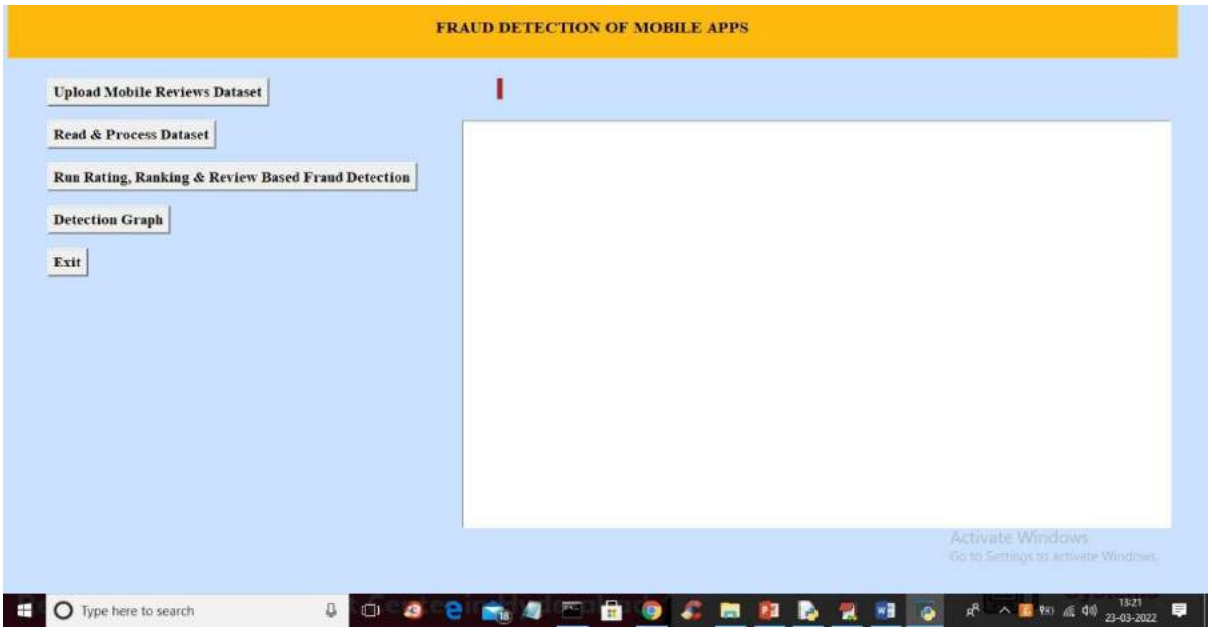

Fig.6.1

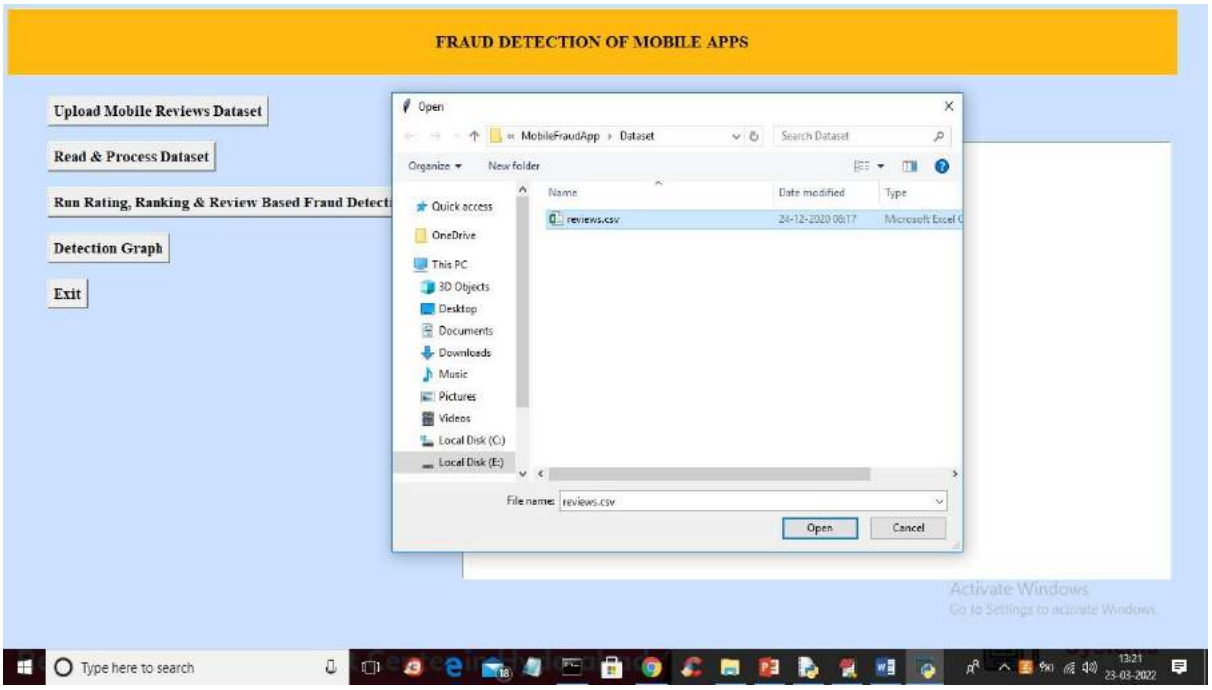

Fig.6.2

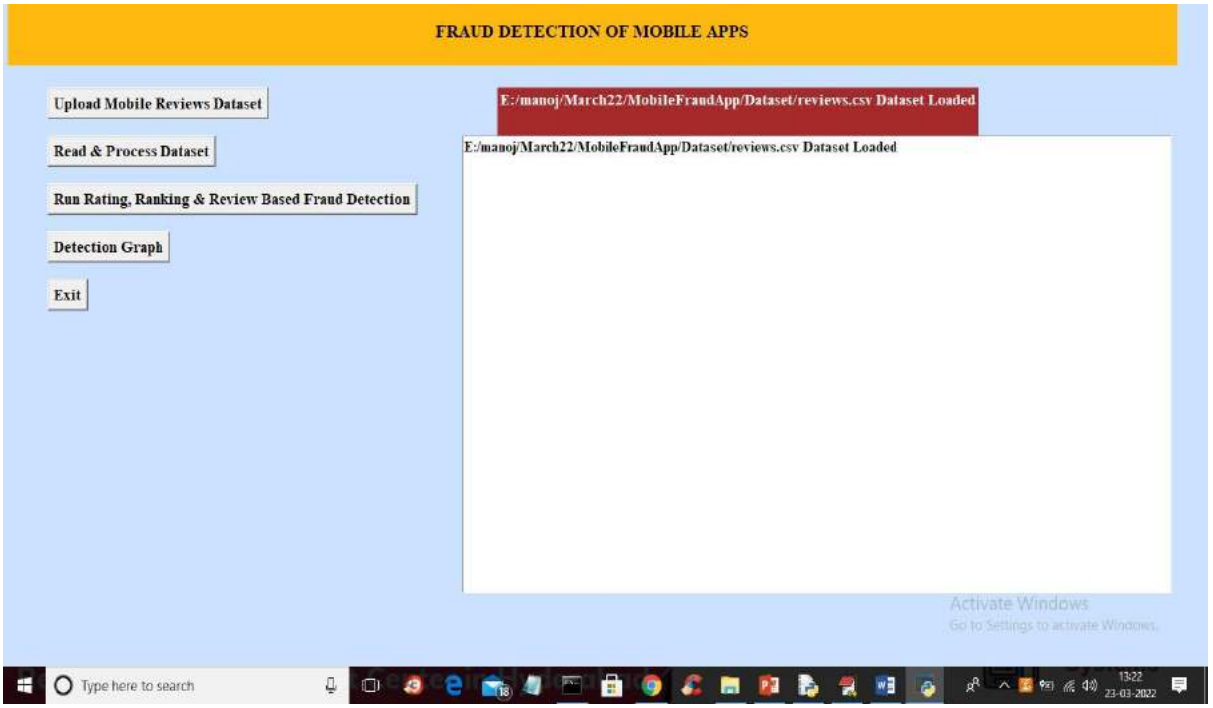
FRAUD DETECTION OF MOBILE APPS
Upload Mobile Reviews Dataset
Read & Process Dataset
Run Rating, Ranking & Review Based Fraud Detection
Detection Graph
Exit
E:/manoj/March22/MobileFraudApp/Dataset/reviews.csv Dataset Loaded
E:/manoj/March22/MobileFraudApp/Dataset/reviews.csv Dataset Loaded
Activate Windows
Type here to search

Fig.6.3

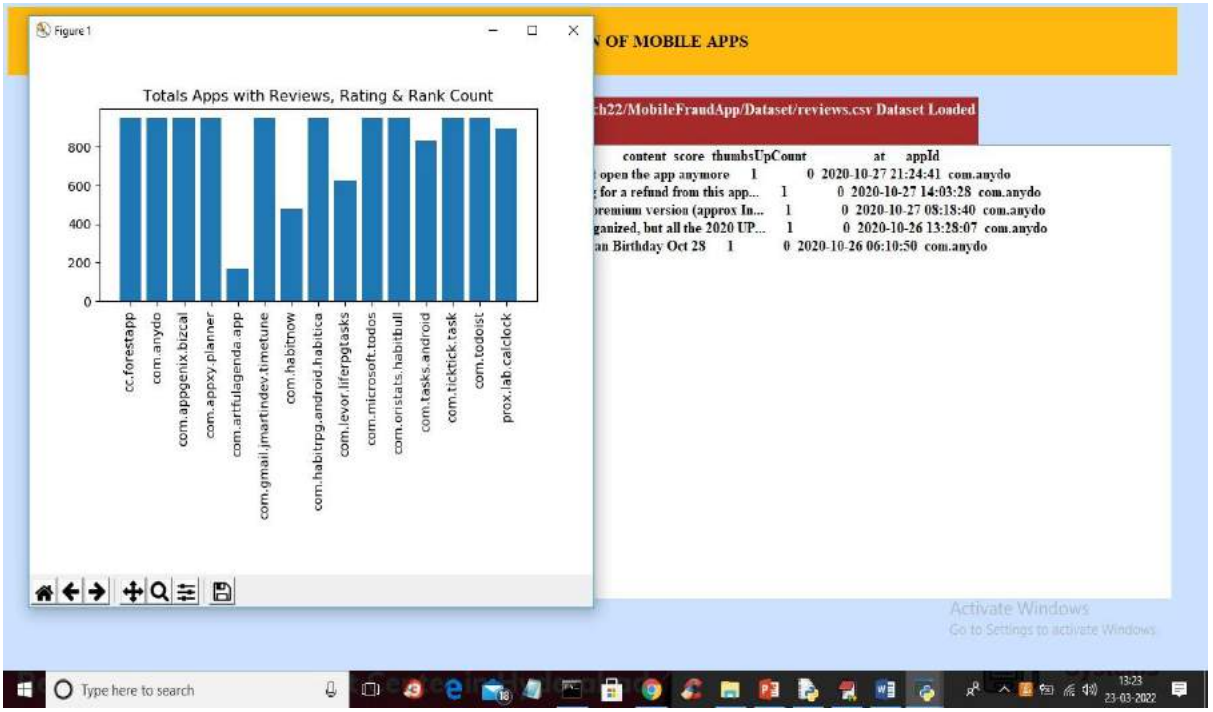
Figure 1
Totals Apps with Reviews, Rating & Rank Count
800
600
400
200
0
cc.forestapp
com.anydo
com.appgenix.bizcal
com.appxy.planner
com.artfulagenda.app
com.gmail.jmartindev.timetune
com.habitnow
com.habitrpg.android.habitica
com.levor.liferpgtasks
com.microsoft.todos
com.oristats.habitbull
com.tasks.android
com.ticktick.task
com.todoist
prox.lab.calclock
N OF MOBILE APPS
h22/MobileFraudApp/Dataset/reviews.csv Dataset Loaded
content score thumbsUpCount at appId
open the app anymore 1 0 2020-10-27 21:24:41 com.anydo
for a refund from this app... 1 0 2020-10-27 14:03:28 com.anydo
premium version (approx In... 1 0 2020-10-27 08:18:40 com.anydo
ganized, but all the 2020 UP... 1 0 2020-10-26 13:28:07 com.anydo
an Birthday Oct 28 1 0 2020-10-26 06:10:50 com.anydo
Activate Windows
Type here to search

Fig.6.4

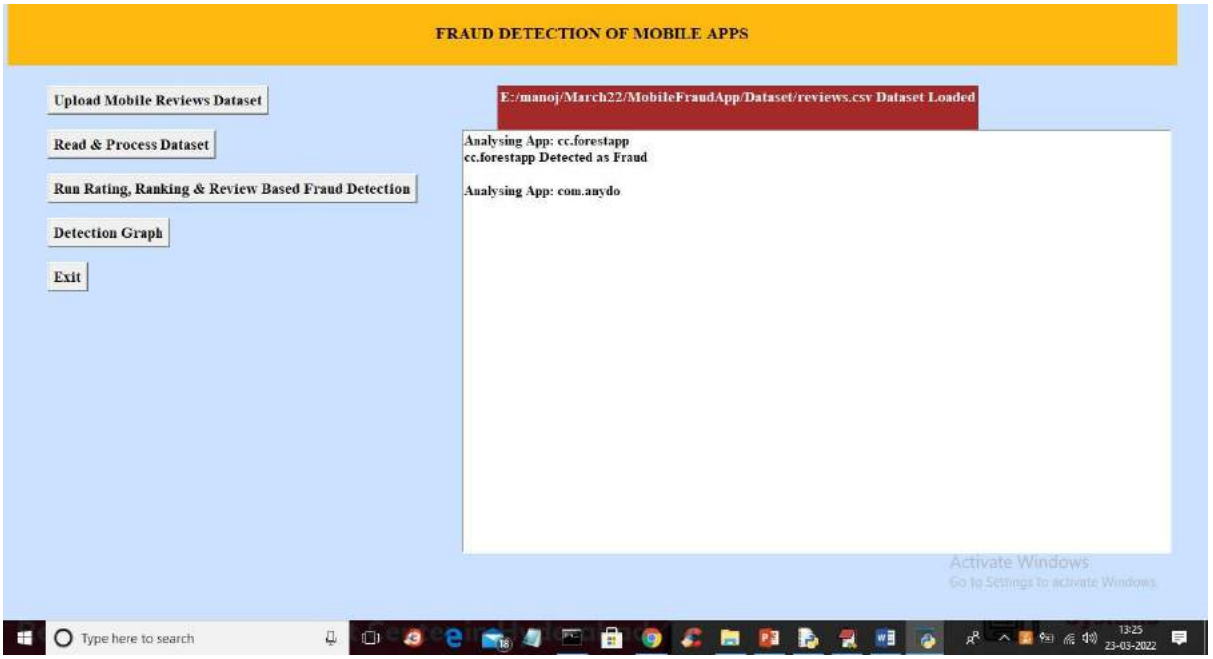
FRAUD DETECTION OF MOBILE APPS
Upload Mobile Reviews Dataset
Read & Process Dataset
Run Rating, Ranking & Review Based Fraud Detection
Detection Graph
Exit
E:/manoj/March22/MobileFraudApp/Dataset/reviews.csv Dataset Loaded
Analysing App: cc.forestapp
cc.forestapp Detected as Fraud
Analysing App: com.anydo
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:25
23-03-2022

Fig.6.5

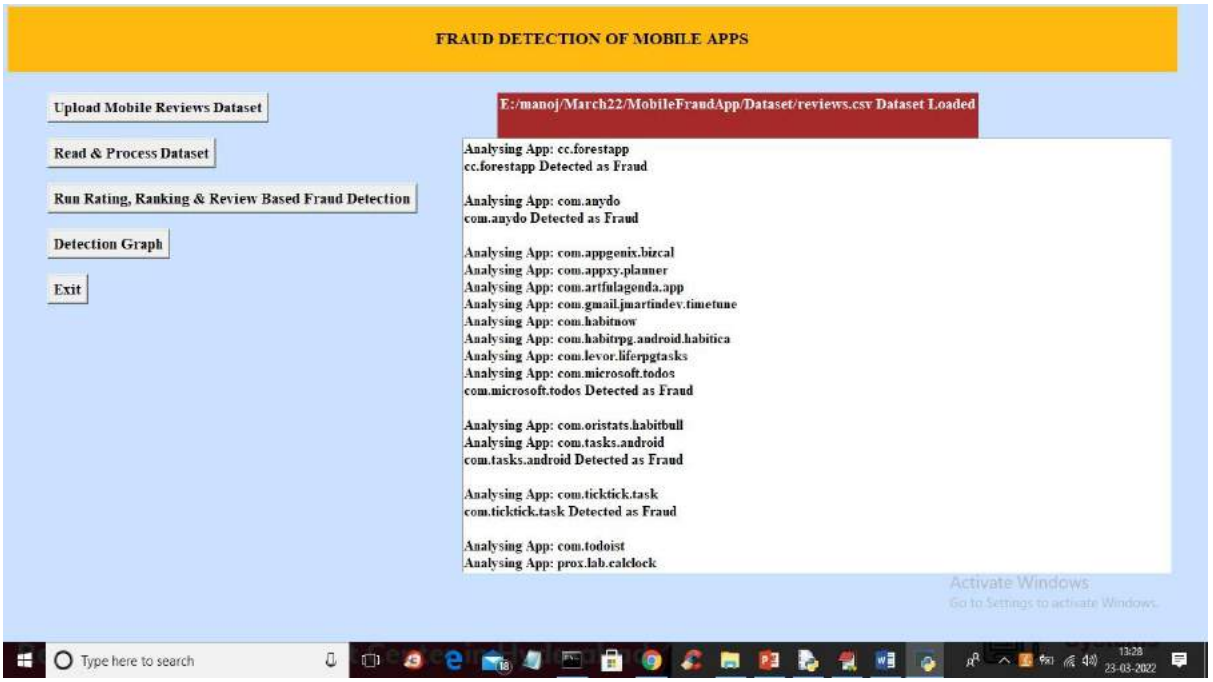
FRAUD DETECTION OF MOBILE APPS
Upload Mobile Reviews Dataset
Read & Process Dataset
Run Rating, Ranking & Review Based Fraud Detection
Detection Graph
Exit
E:/manoj/March22/MobileFraudApp/Dataset/reviews.csv Dataset Loaded
Analysing App: cc.forestapp
cc.forestapp Detected as Fraud
Analysing App: com.anydo
com.anydo Detected as Fraud
Analysing App: com.appgenix.bizcal
Analysing App: com.appxy.planner
Analysing App: com.artfulagenda.app
Analysing App: com.gmail.jmartindev.timetune
Analysing App: com.habitnow
Analysing App: com.habitrpg.android.habitica
Analysing App: com.levor.liferpgtasks
Analysing App: com.microsoft.todos
com.microsoft.todos Detected as Fraud
Analysing App: com.oristats.habitbull
Analysing App: com.tasks.android
com.tasks.android Detected as Fraud
Analysing App: com.ticktick.task
com.ticktick.task Detected as Fraud
Analysing App: com.todoist
Analysing App: prox.lab.calclock
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:28
23-03-2022

Fig.6.6

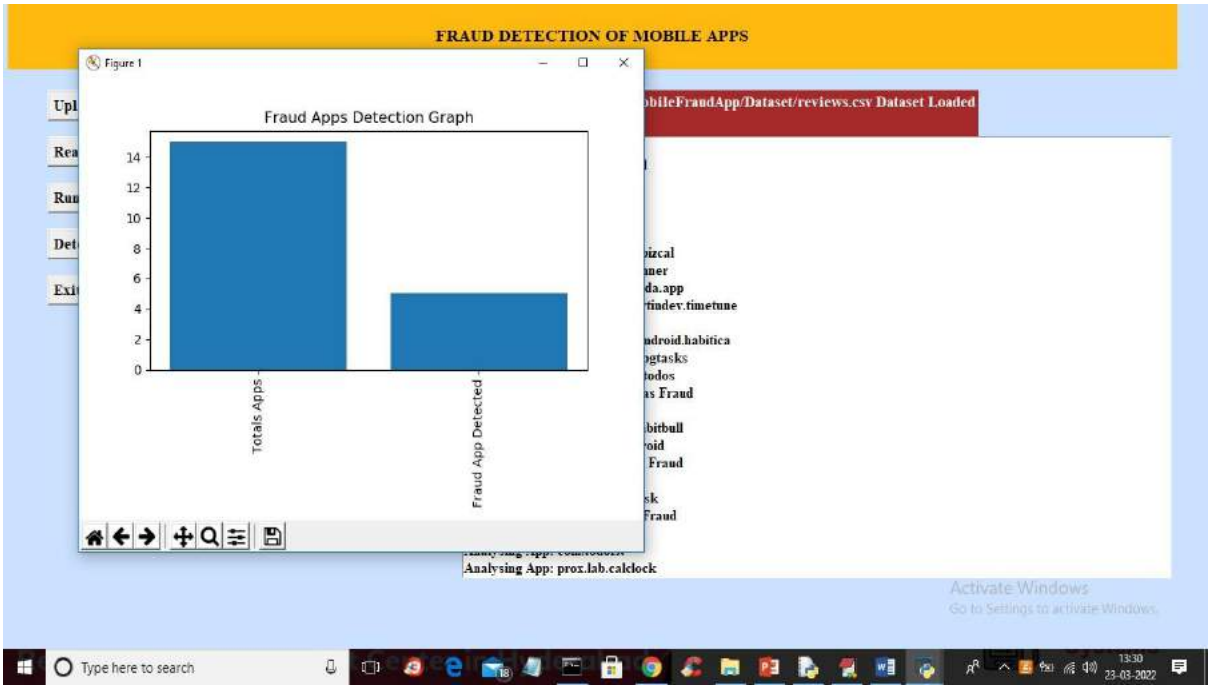
FRAUD DETECTION OF MOBILE APPS
Figure 1
Fraud Apps Detection Graph
14
12
10
8
6
4
2
0
Totals Apps
Fraud App Detected
obileFraudApp/Dataset/reviews.csv Dataset Loaded
Analysing App: prox.lab.calclock
Activate Windows
Type here to search

Fig.6.7

7.TESTING

7.TESTING

7.1.Unit Testing

Unit testing, a testing technique using which individual modules are tested to determine if there are issues by the developer himself.. it is concerned with functional correctness of the standalone modules. The main aim is to isolate each unit of the system to identify, analyze and fix the defects.

Unit Testing Techniques:

Black Box Testing - Using which the user interface, input and output are tested.

White Box Testing –Used to test each one of those functions behavior is tested.

7.2.Data Flow Testing

Data flow testing is a family of testing strategies based on selecting paths through the program's control flow in order to explore sequence of events related to the status of Variables or data object. Dataflow Testing focuses on the points at which variables receive and the points at which these values are used.

7.3.Integration Testing

Integration Testing done upon completion of unit testing, the units or modules are to be integrated which gives raise too integration testing. The purpose of integration testing is to verify the functional, performance, and reliability between the modules that are integrated.

7.4.Alpha Testing

It is the most common type of testing used in the Software industry. The objective of this testing is to identify all possible issues or defects before releasing it into the market or to the user. Alpha Testing is carried out at the end of the software development phase but before the Beta Testing. Still, minor design changes may be made as a result of such testing. Alpha Testing is conducted at the developer's site. In-house virtual user environment can be created for this type of testing.

7.5.Beta Testing

Beta Testing is a formal type of Software Testing which is carried out by the customer.It is performed in the Real Environment before releasing the product to the market for the actual end-users. Beta Testing is carried out to ensure that there are no major failures in the software or product and it satisfies the business requirements from an end-user perspective. Beta Testing is successful when the customer accepts the software. Usually, this testing is typically done by end-users or others. It is the final testing done before releasing an application for commercial purpose.

8.CONCLUSION & FUTURE ENHANCEMENT

8.CONCLUSION&FUTURE ENHANCEMENT:

We developed a ranking fraud detection system for mobile Apps. Specifically, we first showed that ranking fraud happened in leading sessions and provided a method for mining leading sessions for each App from its historical ranking records. Then, we identified ranking based evidences, rating based evidences and review based evidences for detecting ranking fraud. Moreover, we proposed an optimization based on admin verification method for evaluating the credibility of leading sessions from mobile Apps. An unique perspective of this approach is that all the evidences can be model by statistical hypothesis tests, thus it is easy to be extended with other evidences from domain knowledge to detect ranking fraud. The admin can detect the ranking fraud for mobile application. The Review or Rating or Ranking given by users is correctly calculated. Hence, a new user who wants to download an app for some purpose can get clear view about the available applications. Finally; we validate the proposed system with extensive experiments on real-world App data collected from the App store. Experimental results showed the effectiveness of the proposed approach. In the future, we plan to study more effective fraud evidences and analyze the latent relationship among rating, review and rankings. Moreover, we will extend our ranking fraud detection approach with other mobile App related services, such as mobile Apps recommendation, for enhancing user experience.

9.REFERENCES

9.REFERENCES:

[1] (2012). [Online]. Available: http://venturebeat.com/2012/07/03/ applescrackdown-on-appranking-manipulation/

[2] N. Spirin and J. Han. Survey on web spam detection: principles and algorithms. SIGKDD Explor. Newsl., 13(2):50–64, May 2012.

[3]https://developer.apple.com/news/index.php?id=0206 2012a.

[4] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting product review spammers using rating behaviors. In Proceedings of the 19th ACM international conference on Information and knowledge management, CIKM ’10, pages 939–948, 2010.

[5] M. N. Volkovs and R. S. Zemel. A flexible generative model for preference aggregation. In Proceedings of the 21st international conference on World Wide Web, WWW ’12, pages 479–488, 2012.

[6] B. Zhou, J. Pei, and Z. Tang. A spamicity approach to web spam detection. In Proceedings of the 2008 SIAM International Conference on Data Mining, SDM’08, pages 277–288, 2008.

[7] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting product review spammers using rating behaviors. In Proceedings of the 19th ACM international conference on Information and knowledge management, CIKM ’10, pages 939–948, 2010.

[8] Z. Wu, J. Wu, J. Cao, and D. Tao. Hysad: a semisupervised hybrid shilling attack detector for trustworthy product recommendation. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD ’12, pages 985–993, 2012.

[9] S. Xie, G. Wang, S. Lin, and P. S. Yu. Review spam detection via temporal pattern discovery. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD ’12, pages 823–831, 2012.

[10] B. Yan and G. Chen, “AppJoy: Personalized mobile application discovery,” in Proc. 9th Int. Conf. Mobile Syst., Appl., Serv., 2011, pp. 113– 126.

[11] H. Zhu, H. Cao, E. Chen, H. Xiong, and J. Tian, “Exploiting enriched contextual information for mobile app classification,” in Proc. 21stACMInt. Conf. Inform. Knowl. Manage., 2012, pp. 1617– 1621.

APPARELRECOMMENDATIONSYSTEM