# 1. INTRODUCTION

The Mobile App is a very popular and well known concept due to the rapid advancement in the mobile technology. Due to the large number of mobile Apps, ranking fraud is the key challenge in front of the mobile App market. There are millions of apps are available in market for the application of mobile users. However, all the mobile users first prefer high ranked apps when downloading it. To download application smart phone user has to visit play store such as Google Play Store, Apples store etc. When user visit play store then he is able to see the various application lists. This list is built on the basis of promotion or advertisement. User doesn’t have knowledge about the application (i.e. which applications are useful or useless). So user looks at the list and downloads the applications. But sometimes it happens that the downloaded application won’t work or not useful. That means it is fraud in mobile application list. To avoid this fraud, we are making application in which we are going to list the applications. In this paper, we provide a brief view of ranking fraud and propose a ranking fraud detection system for mobile Apps. Specifically, we first propose to accurately locate the ranking fraud by mining the active periods by using mining leading session algorithm. Furthermore, we investigate three types of evidences, i.e., ranking based evidences, rating based evidences and Permission based evidences, by studying historical records. We used an optimal aggregation method to integrate all the evidences for fraud detection. Finally, we evaluate the proposed system with real-world App data collected from the Google 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.

## 1.2 PROJECT DESCRIPTION

The quantity of mobile Apps has developed at an amazing rate in the course of recent years. For instances, the growth of apps were increased by

1.6 million at Apple's App store and Google Play.To increase the development of mobile Apps, many App stores launched daily App leaderboards, which demonstrate the chart rankings of most popular Apps. Indeed, the App leaderboard is one of the most important ways for promoting mobile Apps. A higher rank on the leaderboard usually leads to a huge number of downloads and million dollars in revenue. Therefore, App developers tend to explore various ways such as advertising campaigns to promote their Apps in order to have their Apps ranked as high as possible in such App leaderboards. However, as 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 botfarms or human water armies to inflate the App downloads, ratings and Permissions in a very short time.

There are some related works, for example, we positioning spam recognition, online survey spam identification and portable App suggestion, but the issue of distinguishing positioning misrepresentation for mobile Apps is till under-investigated. The problem of detecting ranking fraud for mobile Apps is still underexplored. Toovercome these essentials, in this paper, we build a system for positioning misrepresentation discovery framework for portable apps that is the model for detecting ranking fraud in mobile apps. For this, we have to identify several important challenges.

First, fraud is happen any time during the whole life cycle of app, so the identification of the exact time of fraud is needed. Second, due to the huge number of mobile Apps, it is difficult to manually label ranking fraud for each App, so it is important to automatically detect fraud without using any basic information. Mobile Apps are not always ranked high in the leaderboard, but only in some leading events ranking that is fraud usually happens in leading sessions.

Therefore, main target is to detect ranking fraud of mobile Apps within leading sessions. First propose an effective algorithm to identify the leading sessions of each App based on its historical ranking records. Then, with the analysis of Apps’ ranking behaviors, find out the fraudulent Apps often have different ranking patterns in each leading session compared with normal Apps. Thus, some fraud evidences are characterize from Apps’ historical ranking records. Then three functions are developed to extract such ranking based fraud evidences. Therefore, further two types of fraud evidences are proposed based on Apps’ rating and Permission history, which reflect some anomaly patterns from Apps’ historical rating and Permission records. In addition, to integrate these three types of evidences, an unsupervised evidenceaggregation method is developed which is used for evaluating the credibility of leading sessions from mobile Apps.

## 1.3 OVERVIEW OF THE PROJECT

With the increase in the number of web Apps, to detect the fraud Apps, this project proposes a simple and effective system. Fig.1 shows the

Framework of Fraud ranking discovery in mobile app

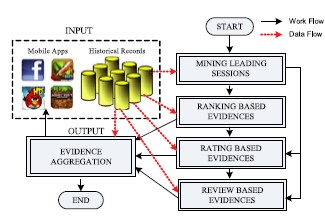


Fig.1 Framework of Ranking Fraud Detection System for Mobile Apps

Indeed, careful observation reveals that mobile Apps are not always ranked high in the leaderboard, but only in some leading events, which form different leading sessions. In other words, ranking fraud usually happens in these leading sessions. Therefore, detecting ranking fraud of mobile Apps is actually to detect ranking fraud within leading sessions of mobile Apps. Specifically, this system first proposes a simple yet effective algorithm to identify the leading sessions of each App based on its historical ranking records. Then, with the analysis of Apps’ ranking behaviors, find that the fraudulent Apps often have different ranking patterns in each leading session compared with normal Apps. Thus, it characterizes some fraud evidences from Apps’ historical ranking records, and develop three functions to extract such ranking based fraud evidences. Nonetheless, the ranking based evidences can be affected by App developers’ reputation and some legitimate marketing campaigns, such as “limited-time discount”. As a result, it is not sufficient to only use ranking based evidences. Therefore, it further proposes two types of fraud evidences based on Apps’ rating and Permission history, which reflect some anomaly patterns from Apps’ historical rating and Permission records. In addition, it develop an unsupervised evidence-aggregation method to integrate these three types of evidences for evaluating the credibility of leading sessions from mobile Apps.

It is worth noting that all the evidences are extracted by modeling Apps’ ranking, rating and Permission behaviors through statistical hypotheses tests. The proposed framework is scalable and can be extended with other domain generated evidences for ranking fraud detection. Experimental results show the effectiveness of the proposed system, the scalability of the detection algorithm as well as some regularity of ranking fraud activities.

## 1.4 OBJECTIVES

The mobile industry is developing rapidly; therefore the numbers of mobile applications are increasing day by day in the market. As there are many apps available in market users are in fuzzy state while downloading the apps for their use. Different App stores like Google play store and Apple store launched their leader board on daily basis to inspire the users to download most popular applications by observing the ranking of applications. In fact to advertise a particular mobile Apps, leader board of apps is the most important way in the market. An app which is at the top on the leader board leads to large number of downloads and it will gain maximum profit. In order to have their Apps ranked as high as possible, app developers promote their apps using various ways such as advertising, offers etc. Such applications damage to phone and also may cause data thefts. Hence such applications must be identified, so that they will be identifiable for play store users. So we are proposing an android application which will process the information*,* comments and three Permissions of the application with natural language processing to give results. So it will be easier to decide fraud application. The main objectives are,

* To rank fraud for mobile application.
* To improve the fraud detection efficiency**.**

## 1.5 MODULES

This system consists of four modules described as follows:

1. Rating Based Evidences
2. Permission Based Evidences
3. Ranking Based Evidences
4. Evidence Aggregation

### 1.5.1 RATING BASED EVIDENCES

After downloading an app users generally rate the app. The rating given by the user is one of the most important factors for the popularity of the app. An app having higher rating always attracts more number of users to download it and naturally it can also be ranked higher in the chart rankings. Thus, in ranking fraud of apps, rating based evidences is also an important feature so they are needs to be considered.

### Pre-processing of ratings

General ratings are between one to five, in this module it will consider, the rating which are less than or equal to three are considered as negative ratings and rating above three are considered as positive ratings

### Rating Score Calculation

Generally, ratings are between one to five, in this module we compute the average rating of particular app and compare it with threshold. The rating which are less than or equal to three are considered as negative ratings and rating above three are considered as positive ratings. Finally, the output is in the form of zeros and ones i.e. negative rating gives zero as an output while positive rating gives one as an output.

### 1.5.2 PERMISSION BASED EVIDENCES

Along with rating users are allowed to write their Permissions about the app. Such Permissions are showing the personalized experiences of usage for particular mobile Apps. The Permission given by the user is one of the most important factors for the popularity of the app. As the Permissions are given in natural language so pre-processing of Permissions and then sentiment analysis on preprocessed Permissions is performed. The system will find sentiment of the Permission which can be positive or negative. Positive Permission adds plus one to positive score, if negative it will add one to negative score. In this way it will find out score of each of the Permissions and determine whether app is fraud or not on the basis of Permission based evidences. This module contains two subparts given below:

### Pre-processing Permissions

This phase consists of following steps:

1. **Tokenization**: Tokenization is the process of breaking a stream of text into words, phrases, symbols or meaningful elements called as tokens. The list of tokens becomes input for further processing.
2. **Stop word removal**: Stop words are commonly used words such as: a, the, and, for, from, is, in and many more.
3. **Stemming**: Stemming algorithm is used to find base word. Porter Stemmer Algorithm is used to find base words.

**Porter Stemmer algorithm**: Porter Stemmer algorithm is a process for removing suffixes from words in English.

Example: A stemming algorithm reduces the words: stems, stemmer, stemming, stemmed as based on “stem”.

### Sentiment Analysis

After pre-processing of Permissions system find out the sentiments of the Permissions. It will classify the Permission as positive or negative. The system will find sentiment of the Permission which can be positive or negative. Positive Permission adds plus one to positive score, if negative it will add one to negative score. In this way it will find out score of each of the Permissions and determine whether app is fraud or not on the basis of Permission based evidences.

**1.5.3 RANKING BASED EVIDENCES**

### Finding app ranking behavior

In this phase, we detect Apps’ ranking behavior, by finding three phases of ranking, namely, rising phase, maintaining phase and recession phase. If the apps ranking reach to peak position in the leaderboard that phase is called as rising phase and maintaining same peak position for specific time period is called as maintaining phase. If the ranking of the app decreases rapidly in the leading event then it is called as recession phase.

### 1.5.4 EVIDENCE AGGREGATION

After three types of fraud evidences are extracted, the next work is to combine them for ranking fraud detection. Every evidence is given a Boolean weight as 0 or 1 where 0 indicate fraud nature and 1 indicate no fraud nature.

# 2. SYSTEM ANALYSIS

## 2.1 EXISTING SYSTEM

In the literature, while there are some related work, such as web ranking spam detection, online Permission spam detection and mobile App recommendation, the problem of detecting ranking fraud for mobile Apps is still under-explored. Generally speaking, the related works of this study can be grouped into three categories. The first category is about web ranking spam detection. The second category is focused on detecting online Permission spam. Finally, the third category includes the studies on mobile App recommendation

### 2.1.1 DISADVANTAGES

* Although some of the existing approaches can be used for anomaly detection from historical rating and Permission records, they are not able to extract fraud evidences for a given time period (i.e., leading session).
* Cannot able to detect ranking fraud happened in Apps’ historical leading sessions
* There is no existing benchmark to decide which leading sessions or Apps really contain ranking fraud.

## 2.2 PROPOSED SYSTEM

In today’s era, due to rapid development in the mobile technology and mobile devices, the applications i.e. mobile apps are being very interesting and popular concept. As there is large number of mobile Apps, ranking fraud is the challenging factor in front of the mobile App market. Ranking fraud is the term used for referring to fraudulent or suspicious activities having the intention of boosting up the Apps in the popularity list. In fact, App developers are using tricky means frequently for increasing their Apps sales. The main aim is to develop such system that find ranking, rating and Permission behaviours for investigating Permission based evidences, rating based evidences and ranking based evidences and then aggregation based on optimization to combine all the evidences for detection of fraud.

### 2.2.1 Advantages

* The proposed framework is scalable and can be extended with other domain generated evidences for ranking fraud detection.
* Experimental results show the effectiveness of the proposed system, the scalability of the detection algorithm as well as some regularity of ranking fraud activities.
* To the best of our knowledge, there is no existing benchmark to decide which leading sessions or Apps really contain ranking fraud. Thus, we develop four intuitive baselines and invite five human evaluators to validate.

## 2.3 FEASIBLITY ANALYSIS

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* Economical Feasibility
* Technical Feasibility
* Social Feasibility

### 2.3.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available.

Only the customized products had to be purchased.

### 2.3.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### 2.3.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. Users’ level of confidence must be raised so that the user is also able to make some constructive criticism, which is welcomed, as end user is the final user of the system.

# 3. SYSTEM SPECIFICATION

## 3.1 HARDWARE REQUIREMENTS

|  |  |
| --- | --- |
| RAM | : 1 GB |
| Hard Disk | : 40GB @ 5400 RPM |
| Processor | : Pentium 4 (1.6 GHZ or above) |

Mouse : Logical Mouse

Keyboard : Logical Multimedia Keyboard

## 3.2 SOFTWARE REQUIREMENTS

|  |  |
| --- | --- |
| Package | : Anaconda IDE, |
| Operating System | : Windows SP2, Vista, Windows 7 |
| Language | : Python |
| Documentation | : Microsoft Office |

## 3.3 ABOUT THE SOFTWARE

**SOFTWARE ENVIRONMENT**

**Python:**

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

* **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**History of Python**

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

**Python Features**

Python's features include −

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
* **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases** − Python provides interfaces to all major commercial databases.
* **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* It supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

**Getting Python**

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python [https://www.python.org](https://www.python.org/).

Windows Installation

Here are the steps to install Python on Windows machine.

* Open a Web browser and go to <https://www.python.org/downloads/>.
* Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
* To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
* Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

**First Python Program**

Let us execute programs in different modes of programming.

**Interactive Mode Programming**

Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

$ python

Python2.4.3(#1,Nov112010,13:34:43)

[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2

Type"help","copyright","credits"or"license"for more information.

>>>

Type the following text at the Python prompt and press the Enter −

>>>print"Hello, Python!"

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**. However in Python version 2.4.3, this produces the following result −

Hello, Python!

**Script Mode Programming**

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file −

print"Hello, Python!"

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

$ python test.py

This produces the following result −

Hello, Python!

**Flask Framework:**

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods −

|  |  |
| --- | --- |
| **Sr.No** | **Methods & Description** |
| 1 | **GET**  Sends data in unencrypted form to the server. Most common method. |
| 2 | **HEAD**  Same as GET, but without response body |
| 3 | **POST**  Used to send HTML form data to server. Data received by POST method is not cached by server. |
| 4 | **PUT**  Replaces all current representations of the target resource with the uploaded content. |
| 5 | **DELETE**  Removes all current representations of the target resource given by a URL |

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

<html>

<body>

<formaction="http://localhost:5000/login"method="post">

<p>Enter Name:</p>

<p><inputtype="text"name="nm"/></p>

<p><inputtype="submit"value="submit"/></p>

</form>

</body>

</html>

Now enter the following script in Python shell.

from flask importFlask, redirect,url\_for, request

app=Flask(\_\_name\_\_)

@app.route('/success/<name>')

def success(name):

return'welcome %s'% name

@app.route('/login',methods=['POST','GET'])

def login():

ifrequest.method=='POST':

user=request.form['nm']

return redirect(url\_for('success',name= user))

else:

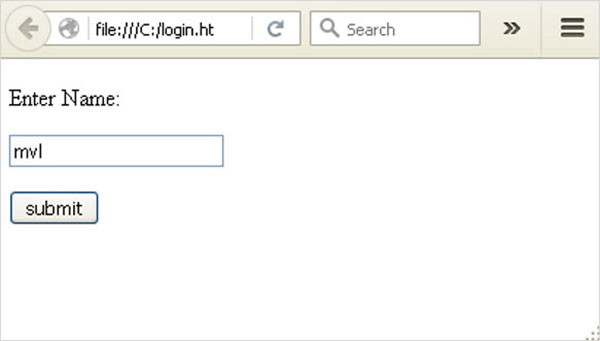
user=request.args.get('nm')

return redirect(url\_for('success',name= user))

if \_\_name\_\_ =='\_\_main\_\_':

app.run(debug =True)

After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.

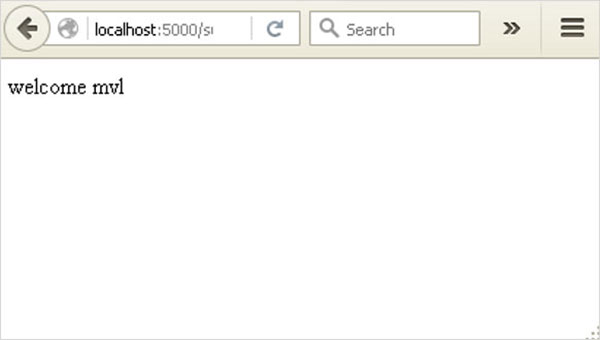


Form data is POSTed to the URL in action clause of form tag.

**http://localhost/login** is mapped to the **login()** function. Since the server has received data by **POST** method, value of ‘nm’ parameter obtained from the form data is obtained by −

user = request.form['nm']

It is passed to **‘/success’** URL as variable part. The browser displays a **welcome** message in the window.



Change the method parameter to **‘GET’** in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of ‘nm’ parameter is now obtained by −

User = request.args.get(‘nm’)

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to ‘nm’ parameter is passed on to ‘/success’ URL as before.

**What is Python?**

Python is a popular programming language. It was created in 1991 by Guido van Rossum.

It is used for:

* web development (server-side),
* software development,
* mathematics,
* system scripting.

**What can Python do?**

* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

**Why Python?**

* Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
* Python has a simple syntax similar to the English language.
* Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
* Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
* Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

* The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
* In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

* Python was designed to for readability, and has some similarities to the English language with influence from mathematics.
* Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
* Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

## Python Install

Many PCs and Macs will have python already installed.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

C:\Users\Your Name>python --version

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

python --version

If you find that you do not have python installed on your computer, then you can download it for free from the following website: <https://www.python.org/>

## Python Quickstart

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

C:\Users\Your Name>python helloworld.py

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

helloworld.py

print("Hello, World!")

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

C:\Users\Your Name>python helloworld.py

The output should read:

Hello, World!

Congratulations, you have written and executed your first Python program.

## The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

C:\Users\Your Name>python

From there you can write any python, including our hello world example from earlier in the tutorial:

C:\Users\Your Name>python  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("Hello, World!")

Which will write "Hello, World!" in the command line:

C:\Users\Your Name>python  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("Hello, World!")  
Hello, World!

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

exit()

Execute Python Syntax

As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

>>> print("Hello, World!")  
Hello, World!

Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:

C:\Users\*Your Name*>python myfile.py

Python Indentations

Where in other programming languages the indentation in code is for readability only, in Python the indentation is very important.

Python uses indentation to indicate a block of code.

Example

if 5 > 2:  
  print("Five is greater than two!")

Python will give you an error if you skip the indentation:

Example

if 5 > 2:  
print("Five is greater than two!")

Comments

Python has commenting capability for the purpose of in-code documentation.

Comments start with a #, and Python will render the rest of the line as a comment:

Example

Comments in Python:

#This is a comment.  
print("Hello, World!")

Docstrings

Python also has extended documentation capability, called docstrings.

Docstrings can be one line, or multiline.

Python uses triple quotes at the beginning and end of the docstring:

Example

Docstrings are also comments:

"""This is a   
multiline docstring."""  
print("Hello, World!")

# 4. SYSTEM DESIGN

## 4.1 INPUT DESIGN

Input design is the process of converting user inputs into computer-based format. The project requires a set of information from the user to prepare a report. In order to prepare a report, when organized input data are needed.

In the system design phase, the diagram identifies logical data flow, data stores and destination. Input data is collected and organised into groups of similar data. The goal behind designing input data is to make the data entry easy and make it free from logical errors. The input entry to all type of clients is the user name and password only. If they are valid the client is allowed to enter into the software.

### Objectives

* To produce a cost-effective method of input.
* To achieve the highest possible level of accuracy.
* To ensure that the input is acceptable and understandable.
* The input design is actually designing of screens. Some of the major screens involved in project.
* Registration form for new client.
* Form for login file.
* Form for displaying connected clients.
* Form for displaying result.

## 4.2 OUTPUT DESIGN

Outputs are the most important and direct source of information to the user and to management. Efficient and eligible output design should improve the system’s relationship with the user and help in decision making. Output Design generally deals with the results generated from stored or calculated values.

Reports are displayed either as screen pPermission or printed form. Most end users will not actually operate the information systems or enter data through workstations, but they will use the output from the system.

### Form Design

The cost of collecting raw data and cost of distributing processed information are major costs of a system. So careful forms design can affect the cost effectiveness of the system. Well-designed forms can increase efficiency; improve workflow and lower system costs.

### Code Design

When a large volume of data is being handled, it is important that items be identified, sorted or selected easily. To accomplish this, each data item must have a unique identification and must be relates to other items of data of the same type. Thus codes are used to identify item uniquely.

A Code is group of characters or numbers used to identify an item or data. While identification is the main function of a code, it may also show relationships between items of data. A good coding scheme should be expandable, precise, concise, convenient and meaningful. Based on above idea, codes are used which contain alphanumeric characters.

## 4.3 DATABASE DESIGN

A general theme begin a database is to handle information as an integrated whole. A database is a collection of inter-related data stored with minimum redundancy to server many users quickly and efficiently.

The general objective is to make information access easy, quick, expensive and flexible for the user. In database design several specific objectives are consider:

**Control Redundancy:**

Redundant data occupies space and therefore, is wasteful. If versions of the same data are in different phase of updating, a system often gives conflicting information. A unique aspect of database design is storing data only once, which controls redundancy and improves system performance.

**Data Independence:**

An important database objective is changing hardware and storage procedures for adding raw new data without having to rewrite application programs.

**Accuracy and Integrity:**

The accuracy and database ensures the data quality content remain constant. Integrity controls detects data inaccuracy where occur.

**Privacy and Security:**

For the data to remain private, security measures must be taken to an unauthorized access. Database security means that data are protected from various forms of destructions. Uses must be positively identifies and actions monitored.Managing the database require a DataBase Administrator (DBA) whose key functions are to be managing data activities, The database structure and the DBMS. In addition a managerial background the DBA needs a technical knowledge to deal with database designer.

## 4.4 DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical system model that shows all of the main requirements for an information system in one diagram: inputs and outputs, processes, and data storage. A DFD describes what data flows rather than how it is processed. Everyone working on a development project can see all aspects of the system working together at once with DFD. That is one reason for its popularity. The DFD is also easy to read because it is graphical model. The DFD is mainly used during problem analysis. End Users, management, and all information systems workers typically can read and interpret the DFD with minimal training.

### Level 0

User

User

Login

Outp

ut

User

Authenticati

on

**Level 1:**

User login

Upload

feedback

Database

Optimize

records

**Level 2**

Apps with

Historical

records

Rating based

evidence

Ranking based

evidence

Permission based

evidence

Optimization

based

aggregation

**Level 3**

Apps with

Historical

records

Rating based

evidence

Ranking based

evidence

Permission based

evidence

Optimization

based

aggregation

Compare with

previous evidence

Detect fraud

### 4.5 USE CASE DIAGRAM

Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core conceits.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development conceits such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and 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. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

user

server

User registration

Upload Permission

Apps with

historical records

Rating based evidence

Permission based

evidence

Ranking based

evidence

Optimization based

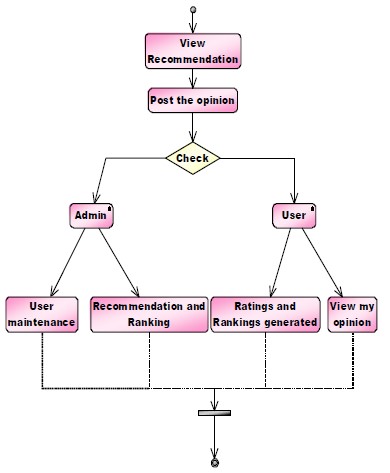
aggregation

Compare with previous

evidence to find defect

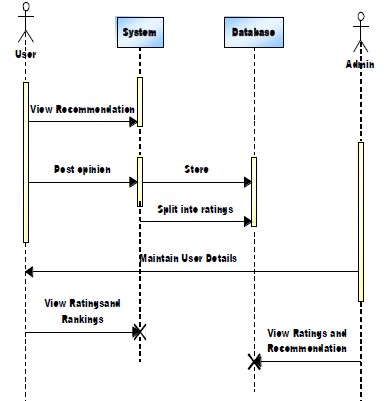
## 4.6. SYSTEM FLOW DIAGRAM

Flowcharts are graphical representations of workflows of stepwise flow and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. A flow chart shows the overall flow of control.



## 4.7. SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams



## 4.8. CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

Server

User

Optimization based

Rating based evidence

Fetect fraud range ()

Ranking based evidence

User name

Company

address

Contact details

Purchase apps()

Permission apps()

Compare with previous

Permission based evidence

Mobile apps

records

Apps name

Apps details

Apps Permissions()

Maitains

historical recods()

### 4.9 ACTIVITY DIAGRAM

Server

User

Login

New patient

register

Permission apps

Purchase apps

yes

No

Post apps

View apps details

Optimization

Compare

previous

Detect fraud range

# 5. SYSTEM TESTING

Testing is the process of running a system with the intention of finding errors. Testing enhances the integrity of a system by detecting deviations in design and errors in the system. Testing aims at detecting errorprone areas. This helps in the prevention of errors in a system. Testing also adds value to the product by conforming to the user requirements.

The main purpose of testing is to detect errors and error-prone areas in a system. Testing must be thorough and well-planned. A partially tested system is as bad as an untested system. And the price of an untested and under-tested system is high.

The implementation is the final and important phase. It involves user-training, system testing in order to ensure successful running of the proposed system. The user tests the system and changes are made according to their needs. The testing involves the testing of the developed system using various kinds of data. While testing, errors are noted and correctness is the mode.

## 5.1 OBJECTIVES OF TESTING

The objectives of testing are:

* Testing is a process of executing a program with the intent of finding errors.
* A Successful test case is one that uncovers an as- yet-undiscovered error.

System testing is a stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as per the user need, before the live operation commences. As stated before, testing is vital to the success of a system. System testing makes a logical assumption that if all parts of the as system are correct, the goal will be successfully achieved. A series of tests are performed before the system is ready for the user acceptance test.

## 5.2 TESTING METHODS

System testing is the stage of implementation. This is to check whether the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. The candidate system is subject to a variety of tests: on line response, volume, stress, recovery, security and usability tests. A series of tests are performed for the proposed system is ready for user acceptance testing.

## 5.3 THE TESTING STEPS

### 5.3.1 Unit Testing

Unit testing focuses efforts on the smallest unit of software design. This is known as module testing. The modules are tested separately. The test is carried out during programming stage itself. In this step, each module is found to be working satisfactory as regards to the expected output from the module.

### 5.3.2 Integration Testing

Data can be lost across an interface. One module can have an adverse effect on another, sub functions, when combined, may not be linked in desired manner in major functions. Integration testing is a systematic approach for constructing the program structure, while at the same time conducting test to uncover errors associated within the interface. The objective is to take unit tested modules and builds program structure. All the modules are combined and tested as a whole.

### 5.3.3 Validation Testing

At the culmination of the integration testing, Software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test begin in validation testing.

Validation testing can be defined in many ways, but a simple definition is that the validation succeeds when the software functions in a manner that is expected by the customer. After validation test has been conducted, one of the three possible conditions exists.

* The function or performance characteristics confirm to specification and are accepted.
* A deviation from specification is uncovered and a deficiency lists is created.
* Proposed system under consideration has been tested by using validation test and found to be working satisfactory.

### 5.3.4 Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in a specific format. The output format on the screen is found to be correct. The format was designed in the system design time according to the user needs. For the hard copy also; the output comes as per the specified requirements by the user. Hence output testing did not result in any correction for the system.

### 5.3.5 User Acceptance Testing

User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes whenever required.

This is done in regard to the following point:

* Input Screen Design
* Output Screen Design
* Format of reports and other outputs.

## 5.4 Error Messages

Error messages and warning messages are “bad news” delivered to the user’s iterative systems where something has gone away or wrong. Therefore in this developed software there are some messages which will be displaying while using this developed software if the user goes wrong. These messages are used to help the user for better use. When the user enters wrong password the displayed message would be Invalid Password! Please try again .While adding information for various modules no field should be empty when saving the information in the database otherwise corresponding error messages are given immediately. While updating the information if any field is left empty then the messages are displayed accordingly Like select the task title, please Enter Numeric values etc.

# 6. SYSTEM IMPLEMENTATION

System Implementation is the creation and the installation of the method to follow the engineering principles to remove part of the human element in the equation. Implementation is the process of realizing the design as a program. The data footprint results presented is might unveil relative performance of different classification techniques(given the memory system is generally deemed as the bottleneck),computation steps are the mechanism involved in dealing with the data structures are equally important and have to be taken into consideration.

To arrive at more accurate evaluation, here executed al classification and throughput performance is measured. System Implementation is a practice of creating or modifying a system to create a new business process or replace an existing business process. Implementation of software refers to the final installation of the packages in its real environment, to the satisfaction of the intended users and the operation of the systems. The people are not sure that the software is meant to make their job easier.

* The active user must be aware of the benefits of using the system
* Their confidence in the software buildup
* Proper guidance is impaired to the user so that it is comfortable in using the application

The system implementation phase consists of the following steps :

* Testing the developed software with sample data
* Correction of any errors if identified
* Creating the files of the system with actual data
* Making necessary changes to the system to find out errors
* Training of users personnel

The system has been tested with sample data, changes are made to the user requirements and run in parallel with the existing system to find out the discrepancies. When the number of threads are rises with respect to the number of cores. The user has also been appraised how to run the system during the training period.

## Implementation Plan

Implementation is the stage, which is crucial in the life cycle of the new system designed. Implementation means converting a new or revised system design into an operational one. The mechanism involved in dealing with the data structures are equally important and have to be taken into consideration .This is the stage of the project where the theoretical design is turned into a working system. In this project implementation includes all those activities that take place to convert from the old system to the new one. The important phase of implementation plan is changeover.

The implementation phase’s construction, installation and operations lie on the new system .The most crucial and very important stage in achieving a new successful system and in giving confidence on the new system for the user that it will work efficiently and effectively.

There are several activities involved while implementing project:

* Careful planning
* Investigation current system and its constraints on implementation
* Design of methods to achieve the change over
* Training of the staff in the changeover procedure and evaluation of change over method

The implementation is the final stage and it is an important phase. It involves the individual programming system testing, user training and the operational running of developed proposed system that constitutes the application subsystems .On major task of preparing for implementation is education of users, which would really have taken place much earlier in the project when were being involved in the investigation and design work. The implementation phase of software development is concerned with translating design specifications into source code. The user tests the developed system and changes are made according to their needs.

## Changeover

The implementation is to be done step by step since testing with dummy data will not always reveal the faults. The system will be subjected to the employees to work. If such error or failure is found, the system can be corrected before it is implemented in full stretch. The trail should be done as long as the system is made sure to function without any failure or errors. Precautions should be taken so that any error if occurred should not totally make the process to a halt. Such a care should be taken. The system can be fully established if it does not create any error during the testing periods.

## Education and User Training

Well-designed and technically elegant systems can succeed or fail because of the way they are operated and used. Therefore the quality of the training received by the personnel involved with the systems help or hinder, and may even prevent, the successful completion of the system. An analysis of user training focuses on user capabilities and the nature of the system being installed. Those users are verifying type and nature. Some of them may not have any knowledge about the computers and the others may be intelligent. The requirements of the system also range from simple to complex tasks. So the training has to be generated to the specific user based on his/her capabilities and system’s complexity. The user tests the develop system and changes are made according to their needs.

Implementation is the stage, which is crucial in the life cycle of the new system designed. Implementation means converting a new or revised system design into an operational one. The mechanism involved in dealing with the data structures are equally important and have to be taken into consideration .This is the stage of the project where the theoretical design is turned into a working system. In this project implementation includes all those activities that take place to convert from the old system to the new one. The important phase of implementation plan is changeover.

User training must instruct individuals in trouble shooting the system, determining whether a problem that arises is caused by hardware or software. The implementation phase of the software development is concerned with translating design specifications into source code. The user tests the developed system and changes are made according to their needs. A good or bad perfect documentation which instructs the user on how to start the system and the various functions and meanings of the various codes must be prepared and that will help the user to understand the system in a better manner.

## System Maintenance

The process of modifying a software system or component after delivery to correct faults, improves Performance or other attributes, or adapt to a changed environment. Systems must be maintained and improved to meet changing business demands or to correct processing errors. Systems maintenance phase begins when a system becomes operational and ends when it is replaced. At some point, a new system will reach the end of its useful life, and the analyst must be able to recognize the signs of system obsolescence. There are many reasons for maintaining a system that fall into the categories given above

* An error / bug is serious enough to need fixing.
* A new business process needs to be incorporated.
* A security vulnerability in the system has been found and needs patching.
* An user has identified how the system could be improved.
* The hardware or network is being improved and so the system should take advantage of that.

There are four major activities that occur to perform maintenance.

## Corrective Maintenance

Corrective maintenance can be defined as the maintenance which is required when an item has failed or worn out, to bring it back to working order. Corrective maintenance is carried out on all items where the consequences of failure or wearing out are not significant and the cost of this maintenance is not greater than preventative maintenance.

Corrective Maintenance activity may consist of repair, restoration or replacement of equipment. This activity will be the result of a regular inspection, which identifies the failure in time for corrective maintenance to be planned and scheduled, then performed during a routine maintenance shutdown.

Corrective maintenance can be subdivided into

## "Immediate Corrective Maintenance" - in which work starts

immediately after a failure

**"Deferred Corrective Maintenance"** - in which work is delayed in conformance to a given set of maintenance rules. In this project the end user can easily correct the faults.

## Preventative Maintenance

Preventative maintenance is maintenance which is carried out to prevent an item failing or wearing out by providing systematic inspection, detection and prevention of incipient failure. The preventative maintenance efforts are aimed at preserving the useful life of equipment and avoiding premature equipment failures, minimising any impact on operational requirements. Preventative maintenance is carried out only on those items where a failure would have expensive or unacceptable consequences e.g. lifts, fire alarms, electricity supply and gas supply. Many of these items are also subject to a statutory requirement for inspection and preventive maintenance.

## Perfective Maintenance

Perfective maintenance is a system of making data input screen a better one to deal with. It has a more advance system to work with, correcting error in a more unique and perfect way. This is maintenance that will improve the performance of the ICT system. Usually this will involve adding features not originally present to the software to make it produce the information from a database faster or to improve the speed of a network.

Examples of making the system 'more perfect' include

* A better data input screen or form
* A more advanced help system
* Tweaks to the code so it is more responsive
* Providing shortcuts commands that experts can use instead of the slower standard menu system

## Adaptive Maintenance

Adaptive maintenance is modifying the system to cope with changes in the software environment. This type of maintenance often occurs as a result of external influences or strategic changes within the company. The system is being *adapted* to remain up to date. This system is developed as run in all types of environments.

# CHAPTER 7

**CONCLUSION AND FUTURE ENHANCEMENTS**

## 7.1 CONCLUSIONS

A ranking fraud detection system for mobile Apps has been developed in this project. Specifically, it 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, it identified ranking based evidences, rating based evidences and Permission based evidences for detecting ranking fraud. Moreover, it proposed an optimization based aggregation method to integrate all the evidences for evaluating the credibility of leading sessions from mobile Apps. A unique perspective of this approach is that all the evidences can be modeled by statistical hypothesis tests, thus it is easy to be extended with other evidences from domain knowledge to detect ranking fraud. Finally, it validates the proposed system with extensive experiments on realworld App data collected from the Apple’s App store. Experimental results showed the effectiveness of the proposed approach.

## 7.2 FUTURE WORK

In the future, it is planned to study more effective fraud evidences and analyze the latent relationship among rating, Permission and rankings. Moreover, it will be extended to ranking fraud detection approach with other mobile App related services, such as mobile Apps recommendation, for enhancing user experience.

# APPENDIX

## SOURCE CODE

# BIBLIOGRAPHY

## BOOKS

1. Allen Jones, Rakesh Rajan (2006), Visual C# 2005 Recipes,A Problem- Solution Approach.McGraw-Hill, 2000.
2. H.M. Deitel (2005), Visual C# 2005: How to program, 2nd Edition
3. O’Reilly (2005), Visual C# 2005: A Developer’s Notebook.
4. Matthew MacDonald “Beginning ASP.NET 3.5 in C# 2008”
5. Pressman, Roger S., Software Engineering “A Practitioner’s Apporach”, Fifth Edition,
6. Shlomy gantz president, Blue Brick Inc, SQL Server 2005.
7. System Analysis and Design by Elias M Awad, 2nd edition, Galgotia Publications, 2006

## WEBSITES

1. <http://venturebeat.com/2012/07/03/> -apples-crackdown-on-appranking-manipulation/
2. [http://www.ibtimes.com/applethreatens- crackdown-biggest-app-storeranking-fra ud-406764](http://www.ibtimes.com/applethreatens-%20crackdown-biggest-app-store-ranking-fra%20ud-406764)
3. <http://en.wikipedia.org/wiki/>information retrieval and validation

## JOURNALS

1. 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.
2. A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *Proceedings of the 15th international conference on*

*World Wide Web*, WWW ’06, pages 83–92, 2006.

1. N. Spirin and J. Han. Survey on web spam detection: principles and algorithms. *SIGKDD Explor. Newsl.*, 13(2):50–64, May 2012.
2. E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting product Permission spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM ’10, pages 939–948, 2010.
3. Z.Wu, J.Wu, J. Cao, and D. Tao. Hysad: a semi- supervised 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

1. S. Xie, G. Wang, S. Lin, and P. S. Yu. Permission 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.

1. B. Yan and G. Chen. Appjoy: personalized mobile application discovery. In *Proceedings of the 9th international conference on Mobile systems, applications, and services*, MobiSys ’11, pages 113– 126, 2011.
2. K. Shi and K. Ali. Getjar mobile application recommendations with very sparse datasets. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD ’12, pages 204–212, 2012.
3. R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” in VLDB, 1994.
4. H. Zhu, E. Chen, K. Yu, H. Cao, H. Xiong, and J. Tian. Mining personal contextaware preferences for mobile users. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 1212–1217, 2012.

[11]Hengshu Zhu, Hui XiongDiscovery of Ranking Fraud for Mobile Apps. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING,2013.