A Google Play Store Fraud Detection Method Based on Decision Trees

N.Neelima Priyanka [1], S.Moshe Dayan [2],
Professor [1],Assistant Professor [2],
Department of IT, SRK INSTITUTE OF TECHNOLOGY ENIKEPADU VIJAYAWADA
Mail Id : priyanka.nutulapati@gmail.com, Mail id : dayan.sripangi@gmail.com

ABSTRACT:

Tablets, smartwatches, and lightweight laptops are all becoming more commonplace. When it comes to mobile app stores, Android is king. Because Android is freely available, hackers often attack mobile applications. The fact that most people have never downloaded or used an app before does not help things. The execution of the mobile app in issue and the user's cooperation were prerequisites for all previous methods of identifying dangerous or fraudulent applications. A system for scanning Google Play for malicious software and unearthing cheating developers that manipulate their search rankings. Combining data from rankings, peer reviews, and ratings may help expose a malicious program. Finally, we can attain a high degree of accuracy in distinguishing harmful, fraudulent, and genuine apps within typical datasets by merging the behaviors of all foreground programs. We employ incremental learning to explain more and more data sets. It worked effectively in tandem with other anti-fraud data. For ranking fraud to be detected, it is important to mine the most popular sessions of mobile applications. I.

1. INTRODUCTION

Recently, dishonest app developers have been using fraudulent techniques to artificially raise their apps and ultimately influence the chart ranks on an app store, rather than depending on standard marketing solutions. In order to rapidly increase the number of downloads, ratings, and reviews of an app, it is common practice to use the services of so-called "boot farms" or "human water armies." Indeed, our in-depth analysis shows that mobile Apps only feature prominently in the top positions during a subset of the top events that together make up the various top sessions. Please keep in mind that we will provide a more in-depth introduction to both major events and major sessions shortly. when is to say, it is at these peak times when ranking fraud is most common? As a result, uncovering ranking fraud of mobile apps requires looking closely at the top sessions of such apps. Avoid utilizing benchmark data altogether by having ranking fraud detected automatically. Lastly, we uncover certain implicit fraud tendencies of mobile Apps as evidences since it is not possible to detect and authenticate the evidences associated to ranking fraud owing to the ever-changing nature of chart ranks. Indeed, our in-depth analysis shows that mobile Apps only feature prominently in the top positions during a subset of the top events that together make up the various top sessions. Please take note that we will provide a more in-depth introduction to leading evens and sessions at a later

time. when is to say, it is at these peak times when ranking fraud is most common? As a result, uncovering ranking fraud of mobile apps requires looking closely at the top sessions of such apps. To begin, we provide a simple approach for determining which App sessions have historically performed well. Then, we analyze the ranking behaviors of Apps and discover that counterfeit Apps often exhibit distinctive ranking patterns throughout leading sessions. Therefore, we define three functions to extract ranking-based fraud evidence from Apps' historical ranking data. However, the credibility of App developers and certain genuine advertising strategies, such "limited-time discount," might impact the ranking based on evidences. Therefore, using simply ranking-based evidences is insufficient. Over the last several years, the availability of mobile applications has skyrocketed. At the end of April 2013, there were over 1.6 million apps available from both the Apple App Store and Google Play. Daily App leader boards, which display the chart ranks of most popular Apps, have been introduced by various App shops to encourage the creation of mobile Apps. Among the many effective methods of advertising mobile Apps, the App leader board stands out. More downloads and more money in the bank generally follow a high position on the leaderboard. For this reason, app makers often use promotional strategies including ad campaigns in an attempt to raise their apps' positions in app stores' popularity rankings. However, there has been a new trend of dishonest App creators using fraudulent techniques to purposefully increase their Apps and ultimately influence the chart ranks on an App store, rather than depending on conventional marketing solutions.

2. LITERATUR REVIEW

The author of study [3] suggests using a static technique to identify malicious software in mobile

applications. The technology in question uses the notion of reverse engineering to examine the APK files for their source code. The classes' framework is then crafted using structured mapping. Finally, multiple patterns for the various types of threats have been developed using the data flow paradigm, and these may be put to use to identify malicious code in software. Its efficiency is computed based on the total number of threading patterns. The author of article [1] offered a novel approach to detecting malware in mobile apps by analyzing the application's runtime behavior in a mobile environment. The text suggests that unanticipated mobile app behavior may change from app to app. It also differs depending on the specific application environment being used by the various devices. The Exposed framework allows users to make adjustments to their user and system applications without having to recompile their APK.The user may customize the criteria used to detect malicious software in mobile apps. Some cutting-edge machine learning techniques for malware detection are proposed in article [2]. This is achieved by applying these techniques to Google Play information. This provides a direct mechanism to recognize the apps, while all previous approaches to algorithm detection have relied on identifying unique features inside each mobile app. They used 25k data obtained from Google Play to set out the studies. In contrast to legitimate apps, which are updated regularly by their developers, bogus ones cannot be updated after they have been uploaded to Google Play. These studies have exclusively dealt with linear models. Non-liners models may be the subject of future research. The purpose of paper [5] is to shield legitimate reviews from review spanners and spam. The spammer may narrow their focus to that one particular firewall. After that, they provided the mobile app in question with false reviews by using a separate account to post such ratings. To find all reviews of a certain product, the author suggests a novel based grading technique.

3. SYSTEM DESIGN

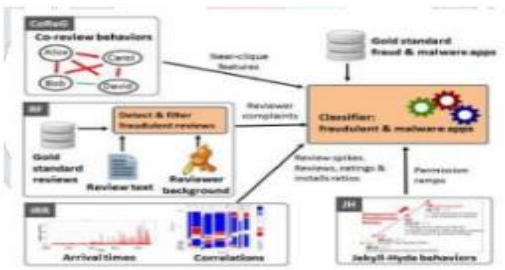
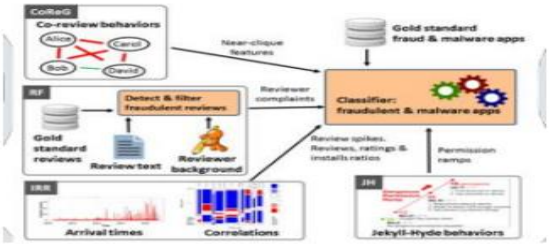


Fig1: Architecture Diagram

The Co-Review Graph (CoReG) component finds app review clusters where users have a high degree of review overlap. Genuine reviewer feedback is used by the Review Feedback (RF) module, while inter-review relations (IRR) are capitalized upon by the Inter-Review Relation (IRR) module. In order to detect programs that transform from safe to malicious, the Jekyll-Hyde (JH) module keeps an eye on permissions, paying special attention to potentially harmful ones. There are a total of 28 features generated by the program (one each from the eight modules). These include the average rating, the number of reviews, ratings, and installations, and more. Co-ReG is a module that detects suspicious co-review activity based on timing. The RF module employs linguistic methods to identify fake reviews that highlight suspicious activity. The IRR component can identify potentially harmful programs based on their behavior. The JH module locates potential permissions ramps for transitioning between Jekyll and Hyde in an application. Two hundred applications produced by credible media sites each have more than ten reviews. The 32,022 ratings for these applications have also been compiled.

4. FAIRPLAY: PROPOSED SOLUTION

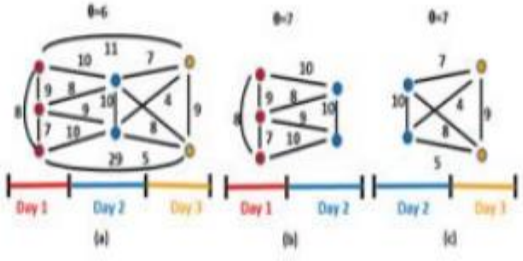
We're excited to unveil FairPlay, an automated mechanism for identifying fraudulent and harmful applications.



Schematic of the FairPlay system. Suspicious co-review activity over time may be spotted with the help of the CoReG module. The RF module employs linguistic methods to identify fake reviews that highlight suspicious activity. The IRR component can identify potentially harmful programs based on their behavior. In order to locate potential transitions between Jekyll and Hyde, the JH module uses permission ramps.

TABLE 1
FairPlay's Most Important Features, Organized by Their Extracting Module

Notation	Definition
CoReG Module	
$nCliques$	number of pseudo-cliques with $\rho \geq \theta$
$\rho_{max}, \rho_{med}, \rho_{SD}$	clique density: max, median, SD
$CS_{max}, CS_{med}, CS_{SD}$	pseudo-cliques size: max, median, SD
$inCliqueCount$	% of nodes involved in pseudo-cliques
RF Module	
$malW$	% of reviews with malware indicators
$fraudW, goodW$	% of reviews with fraud/benign words
FRI	fraud review impact on app rating
IRR Module	
$spikeCount, spike_{amp}$	days with spikes & spike amplitude
$I_1/Rt_1, I_2/Rt_2$	install to rating ratios
$I_1/Rv_1, I_2/Rv_2$	install to review ratios
JH Module	
$permCt, dangerCount$	# of total and dangerous permissions
$rampCt$	# of dangerous permission ramps
$dangerRamp$	# of dangerous permissions added



Typical PCF output and pseudo-cliques are shown in Fig. 6. Users (the nodes) are connected by edges (the weights) that represent the amount of rated applications they have in common. The time stamps for reviews are accurate to the nearest day. When u=6, PCF identifies the whole co-review graph as a pseudo-clique. When u is 7, PCF finds the two-day (b) and four-day (c) sub graphs. Only the group created by the reviews from the first day is picked up by PCF (the red nodes) when u=8.

PYTHON

WHAT IS A SCRIPT?

So far, I've mostly spoken about how Python can be used for interactive programming. It's fantastic to be able to write code and see it execute in an interactive setting like this.

SCRIPTS ARE REUSABLE:

A Python script is a text file containing the program's statements. Once the script has been developed, it may be executed repeatedly with little or no user intervention.

SCRIPTS ARE EDITABLE:

One of the most important features of this script is that it can be easily modified by just changing the statements in one file and putting them into another file using a text editor. At that time, you may start using all of the various options. Because of this, developing new software is a breeze.

YOU WILL NEED A TEXT EDITOR.

It's possible to use any text editor to write Python script files. You're free to use whichever word processor you choose, whether it Notepad, WordPad, Word, etc., from Microsoft. What Sets Scripts and Computer Code Apart

SCRIPT:

Scripts are independent from the core code of the program, which is often written in a different language, and are typically created or modified by the user. Scripts are often translated from source code or byte code, while the programs they control are commonly compiled to native machine code.

PROGRAM:

Since the program is in an executable format, it may be launched immediately. This is the same program in a form that people can understand, known as "source code" (as opposed to a built version).

DJANGO FRAMEWORK

ango is a framework for developing dynamic web applications. With so many "batteries included" in the massive Django web-framework, it's a wonder how everything works together. The idea behind including so many "batteries" is to prevent the need for external libraries by providing support for frequently-used web features directly inside the framework. The large Django community is a key factor in the framework's success. The community is so large that it warranted its own website, where programmers from all over the world contributed third-party packages including login and authorisation systems, complete content management frameworks based by Django, e-commerce extensions, and much more. There is a good chance that what you are attempting to create has already been created elsewhere, and all you have to do is incorporate it into your project.

WHY SHOULD YOU USE DJANGO?

Django is built in a manner that facilitates fast, streamlined, and functional website development. Django differentiates apart from other frameworks because to its focus on efficiency. Django is a great framework to think about using if you want to create a highly adaptable project, such a social networking site. Django's strength is in the way its users can collaborate and exchange content of all kinds. Django's ability to use community support to provide highly customized third-party plugins for usage in applications is a major plus.

PYTHON

Due to its straightforward syntax, logical structure, and general ease of usage, Python is often regarded as one of the simplest programming languages to pick up. It's widely used as a scripting language, as well as for running webpages, desktop programs, and mobile apps in a variety of embedded devices.

BATTERIES INCLUDED

URL routing, authentication, an ORM, a templating system, and db.-schema migrations are just some of the standard features that may be built using Django's included packages.

BUILT-IN ADMIN

Django has an in-built administration interface which lets you handle your models, user/ group permissions and to manage users. With model interface in place, there is no need for a separate database administration program for all but advanced database functions.

DOESN'T GET IN YOUR WAY.

When developing a Django program, you won't have to deal with any extraneous or repetitive tasks. You won't find any XML configuration files, external libraries, or required imports here.

SCALABLE

Django is built on the model-view-controller pattern. This implies that everything, including the database, the backend, and the frontend, are all separate entities. With Django, we can easily divide your site's code and dynamic elements like images, files, CSS, and JavaScript. When it comes to web servers, caching, performance management, clustering, and balancing, Django has you covered with a comprehensive collection of third-party libraries. Django's compatibility with popular email and messaging platforms and services, such as Rest and OAuth, is a key benefit.

BATTLE TESTED.

In 2005, Django was released as open source software. Django has expanded over the last 12 years to power not just news publishing websites but also Pinterest, Instagram, Disqus, Bitbucket, Event rite, and Zapier. It's a solid and dependable web framework because of this.

WHAT IS MACHINE LEARNING?

Machine learning is the study of how computers may acquire new skills without being provided with any new instructions. Machine learning is a cutting-edge technology on par with any other. As the name implies, it imparts onto the computer a skill that brings it closer to human standards: the capacity to learn. There are numerous real-world applications of machine learning today, possibly more than you would think. Machine learning is the brains behind everything from language translation software to driverless cars. It provides a means through which difficult questions may be resolved. Training an algorithm or model is the process of teaching a computer program to draw accurate conclusions from given data. The many types of machine learning challenges and the associated jargon are covered in this page.

TYPES OF MACHINE LEARNING PROBLEMS:

Machine learning challenges may be categorized in a number of different ways. We'll talk about the most basic ones here.

TOP MACHINE LEARNING ALGORITHMS TO KNOW.

Algorithms for machine learning are the backbone of every machine learning model. In order to get started in the field of machine learning, here are seven essential algorithms ranging from classification to regression:

LINEAR REGRESSION

A supervised learning method, linear regression is used to make predictions and forecasts across a continuous range of data, such as sales figures or prices. Linear regression is a statistical technique for making predictions by relating an input (X) value to an output (Y) value with a constant slope. Predictions may be made using labelled data using linear

regression by approximating a line of best fit (the "regression line") from a scatter plot. This is why predictive modeling with linear regression is preferred over classification.

2. LOGISTIC REGRESSION

Logistic regression, often known as "logit regression," is a supervised learning technique used for binary classification, including determining if an image belongs to a certain category or not. Logistic regression, a technique with its roots in statistics, is used to make predictions about how likely it is that a given input belongs to a single, overarching class. However, this may be put into practice to classify results into one of two buckets ("the primary class" or "not the primary class"). To do this, a binary classification range is established, in which values between 0 and.49 are sorted into one category and values between.50 and 1.00 are sorted into another. Consequently, in machine learning, logistic regression is often used for binary classification as opposed to predictive modeling.

3. NAIVE BAYES

Naive Bayes is a family of supervised learning algorithms that may be used to build classification and prediction models. Naive Bayes is a classification method that uses conditional probabilities, which are derived from Bayes' theorem and are mutually exclusive but together show the probability of a classification based on their combined elements. A naive Bayes method might be used by a plant identification computer to classify photos based on features such apparent size, color, and form. While each of these characteristics is distinct, when taken together they increase the possibility that a given thing is a plant.

WHAT IS AI (ARTIFICIAL INTELLIGENCE)?

INTRODUCTION:

The human race has always, and will always, be hungry for new ways to improve their quality of life. What a human mind is capable of doing has always amazed me. Artificial intelligence (AI) is one such ground-breaking innovation. What if robots have the capacity for thought? To put it simply, AI is that. Human beings are inherently smart. A machine's intelligence would be synthetic if it were to think. Therefore, AI refers to any and all intelligent devices.

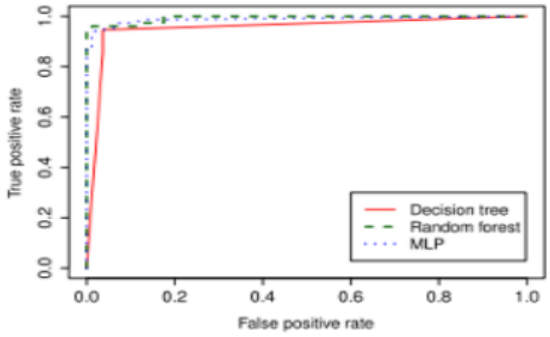
Here are some applications of AI that are already in use today. The first thing that enters my head is a robot. They are mechanical duplicates of real people. They are capable of independent thought and can make crucial choices without human intervention. However, AI devices don't always have to be humanoid in design. Self-driving vehicles, Amazon Alexa, and Siri are a few more examples. Speech recognition is another useful tool. Do you remember how you used to be able to ask Google a question instead of putting it into the search bar? That's an example of one of the uses. There are many other uses, but I need to move on. Artificial intelligence (AI) is the study and development of computer systems capable of performing activities normally requiring human intellect. The purpose of artificial intelligence is to develop computer programs and other systems that can acquire knowledge and improve themselves via experience. Narrow AI, sometimes known as "weak AI," and broad AI, also known as "strong AI," are the two types of AI systems that exist. Narrow AI is programmed to carry out one or a small number of specified tasks, such as recognizing images or voices or learning to play a game. General artificial intelligence, on the other hand, is capable of everything a person is intellectually capable of, including learning and problem solving.

5. EVALUATION

Experiment Setup

To categorize reviews and applications, we used the R tool, while Python was used to extract data from parsed pages and calculate features. In order to identify even the smallest pseudo cliques, we have set the threshold density value u to 3. The studies were run using the standard configuration of the Weka data mining package [34]. Several supervised learning methods were tried out in our experiments.

Using 10-fold cross validation [37], we give findings for the top three methods—Multilayer Perception (MLP) [35], Decision Trees (DT) (C4.5), and Random Forest (RF) [36]. We used a 0.3 learning rate and a 0.2 momentum rate for the MLP classifier's backpropagation technique. on long-term data and feature storage, we opted on MySQL. FPR stands for false positive rate, and "positive" refers to a bogus review, fraudulent app, or malware app. In the same way, "negative" indicates an actual review or harmless app, and FNR refers to the pace at which this occurs. The ROC curve is used to graphically present the compromise between the FPR and the FNR. The TPR measures the actual success rate. When the rate of positive and negative mistakes is the same, we say that the rate is the Equal Error Rate (EER). A lower EER indicates a more precise answer.



Three different classifiers—a Decision Tree, a Random Forest, and a Multilayer Perception (MLP)—are shown in Fig. 7's ROC curve. As a categorization review. At 96.26 percent, RF and MLP are tied for first place in accuracy. MLP has a very low EER of 0.036.

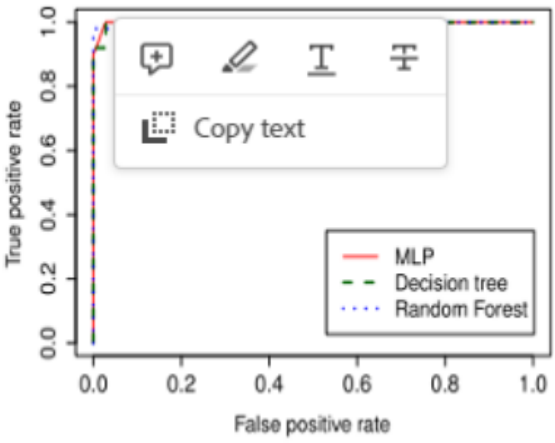


Figure 8: ROC plot of three classifiers (Decision Tree, MLP, and Bagging) used to determine if an app is trustworthy or not. With a success rate of 99.1 percent, Decision Tree is the most accurate method.

As little as 0.01% is the EER for MLP.Random Forest has the lowest FPR (1.51%) and best accuracy (96.11%) for FairPlay. It also has an AUC of 0.986 and an EER of 4%. Surprisingly, most FairPlay features effectively identify malware while being designed to detect search rank fraud. Does Fraud Involve Malware? We hypothesized that the aforementioned outcome was influenced by search rank fraud via infected programs.

CONCLUSION

We created a method to identify malicious Apps as part of this project. First, we demonstrated that fraudulent activity was concentrated in high-volume, high-ranking sessions, and then we developed a means of extracting these sessions for use in further analysis. We found that ranking, rating, and review based evidence are taken into account during the identification of the rank. To further evaluate the reliability of top sessions inside mobile apps, we also suggested an optimization-based aggregation strategy to include all available information. All the information may be described by statistical hypothesis testing, making this method particularly amenable to being supplemented with additional evidence from domain expertise in the pursuit of detecting ranking fraud. Finally, we test the proposed method extensively using data from the Apple App Store to ensure its efficacy. The suggested method was shown successful in experiments. In the future, we want to examine the hidden connection between ratings, reviews, and rankings, as well as examine other convincing fraud evidence. In addition, we want to improve the user experience by combining our ranking fraud detection method with other services relating to mobile apps, such as app recommendations.

REFERENCES

[1]Alaa Salman Imad H. Elhajj Ali Chehab Ayman Kayss, IEEE Mobile Malware Exposed.International Conference on Knowledge discovery and data mining, KDD'14 pages 978- 983.

[2]Alfonso Munoz, Ignacio Mart ˜ ´ın, Antonio Guzman, Jos ´ e Alberto Hern ´ andez, IEEE Android malware detection from Google Play meta-data: Selection of important features.2015, pages,245-251.

[3]Chia-Mei Chen, Je-Ming Lin, Gu-Hsin Lai,IEEE Detecting Mobile Application Malicious Behaviors Based on Data Flow of Source Code.2014 International Conference on Trustworthy Systems and their Applications pp 95-109.

[4]D. F. Gleich and L.-h. Lim. Rank aggregation via nuclear norm minimization. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '11, pages 60–68, 2011.Y.T. Yu, M.F. Lau, "A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions", Journal of Systems and Software, 2005, in press.

[5]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.

[6] Andy Greenberg. Malware Apps Spoof Android Market to Infect Phones. Forbes Security, 2014. IEEE Transactions on Knowledge and Data Engineering, Volume: 29, Issue: 6, Issue Date: June.1.2017 14

[7] Freelancer. *http://www.freelancer.com.*

[8] Fiverr. https://www.fiverr.com/. [9] BestAppPromotion. *www.bestreviewapp.com/.*

[10] Gang Wang, Christo Wilson, Xiaohan Zhao, Yibo Zhu, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. Serf and Turf: Crowdturfing for Fun and Profit. In Proceedings of ACM WWW. ACM, 2012.

[11] Jon Oberheide and Charlie Miller. Dissecting the Android Bouncer. SummerCon2012, New York, 2012.

[12]Alaa Salman Imad H. Elhajj Ali Chehab Ayman Kayss, IEEE Mobile Malware Exposed.International Conference on Knowledge discovery and data mining, KDD'14 pages 978- 983. VirusTotal - Free Online Virus, Malware and URL Scanner. https: //www.virustotal.com/, Last accessed on May 2015.

[13] Iker Burguera, Urko Zurutuza, and Simin Nadjm-Tehrani. Crowdroid: Behavior-Based Malware Detection System for Android. In Proceedings of ACM SPSM, pages 15–26. ACM, 2011.

[14] Asaf Shabtai, Uri Kanonov, Yuval Elovici, Chanan Glezer, and Yael Weiss. Andromaly: a Behavioral Malware Detection Framework for Android Devices. Intelligent Information Systems, 38(1):161–190, 2012.

[15] Michael Grace, Yajin Zhou, Qiang Zhang, Shihong Zou, and Xuxian Jiang. Riskranker: Scalable and Accurate Zero-day Android Malware Detection. In Proceedings of ACM MobiSys, 2012.

[16] Bhaskar Pratim Sarma, Ninghui Li, Chris Gates, Rahul Potharaju, Cristina Nita-Rotaru, and Ian Molloy. Android Permissions: A Perspective Combining Risks and Benefits. In Proceedings of ACM SACMAT, 2012.