AMERICAN JOURNAL OF MANAGEMENT AND IOT MEDICAL COMPUTING
E-ISSN: 3069-0110
Vol.4, No. 4(2025)
www.ajmimc.com

FRAUD APP DETECTION OF GOOGLE PLAYSTORE APPS USING MACHINE LEARNING

Mrs.R.Sreelakshmi[1], Nalla Lavanya[2], T Raghavendra Goud[3], Konda Mahalakshmi[4], Narkatpalli Rahul[5]

[1]*Assistant Professor, Department of CSE(DS), Samskruti College Of Engineering And Technology, Kondapur (V), Ghatkesar (M), Medchal (D), Hyderabad, India.*

[2,3,4,5]*Student, Department of CSE(DS), Samskruti College Of Engineering And Technology, Kondapur (V), Ghatkesar (M), Medchal (D), Hyderabad, India.*

Corresponding email ID: sree.balu12@gmail.com

ABSTRACT:

Along the rise in the various mobile applications which are used in daily life, it's more necessary than ever to stay on top of things to decide which are safe and which don't. It is impossibleto pass judgment. Our system is based on four parameters that include ratings, reviews, in app purchases and Contains ad to predict. Our system compares three models Decision Tree classifier, Logistic Regression and Naïve Bayes. These models were further analyzed on four parameters of F1 score, Recall, Precision and Accuracy. A good F1 score should be greater than 0.7 and are call score greater than 0.5 is considered to be good with higher precision and accuracy. On analysis we found Decision tree model as a good model with accuracy of 85%, F1score of 0.815, Recall value of 0.85 and precision of 0.87

Keywords: Decision Tree classifier, Logistic Regression and Naïve Bayes

1 INTRODUCTION

The Google Play Store stands as a vibrant marketplace hosting millions of mobile applications, catering to a diverse range of user needs. However, within this expansive ecosystem lies a persistent threat: fraudulent applications. These deceptive apps not only jeopardize users' financial security but also pose significant risks of data breaches and privacy violations. To address this growing concern, the application of machine learning (ML) techniques emerges as a crucial strategy.Traditional approaches to fraud detection often struggle to keep pace with the evolving tactics of malicious actors. Rule-based systems, while useful, may fail to detect subtle patterns indicative of fraudulent behavior. In contrast, machine learning offers a dynamic and adaptive approach, capable of discerning intricate patterns from vast datasets.

This paper delves into the realm of fraud detection within the Google Play Store, leveraging the power of machine learning algorithms. Our objective is to develop a robust framework capable of identifying and flagging potentially fraudulent applications with high precision.

By analyzing prevalent fraud patterns and tactics employed by malicious actors, we seek to gain insights into the evolving landscape of app-based fraud. Through empirical evaluation, we aim to assess the efficacy of various machine learning algorithms in detecting fraudulent activities, considering factors such as feature engineering, model selection, and performance evaluation metrics. Our ultimate goal is to contribute towards building a scalable and adaptable fraud detection system that can effectively counter emerging threats within the dynamic app ecosystem. By enhancing the security and integrity of the Google Play Store platform, we endeavor to create a safer and more trustworthy environment for both users and developers.

2 LITERATURE SURVEY

Fraudulent activity in mobile app market means intrudersor app developers uses had means to increase their app rating to bring their app in top 20 list to inflate their app sale. Knowledge engineering domain usually uses the methodologies to extract the useful knowledge from the given Large data. On going rapid growths of online data have created the need of KDD. Also ongoing rapid growth of online rating and review system to the app, make fraud app has

Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |

been launched in the mobile market and let them bed own loaded and used by many users. The fraud mobile app is not worth to use and wasting device memory. Sometimes such app is created with malicious software which is harmful to the device. To avoid this situation the fraud app should be find out. In existing work, fraud rankings are detected by applying the mining algorithm in app review. Local anomaly was detected instead of global anomaly. The analysis had be end one reported. Human evaluators evaluated and produce the result.

Now days, mobile App is a very popular and well known concept due to the rapid advancement in the mobile technology and mobile devices. Due to the large number of mobile Apps, ranking fraud is the key challenge infront of the mobile App market. Ranking fraud refers to fraudulent or vulnerable activities which have a purpose of bumping up the Apps in the popularity list. In fact, it becomes more and more frequent for App developers to use tricky means, like increasing their Apps' sales or posting fake App ratings, to commit ranking fraud. While the importance and necessity of preventing ranking fraud has been widely recognized. After understanding the details of ranking fraud and the need of ranking fraud detection, the paper proposes a ranking fraud detection system for mobile Apps. The proposed system mines the active periods such as leading sessions of mobile apps to accurately locate the ranking fraud.

In Today's world, smart phone are very important in our daily life. In today's Scenario every one is using smart phone. Nowadays, the rear enumerous applications out there on web due to that user cannot continuously get correct or true reviews concerning the merchandise on web. There are so many fraud applications on the internet. The growth of apps was increased by 1.6 million at App Store and Go ogle Play. There are many apps from which any app can be fraud, so the identification of true app is needed.

Rank misrepresentation in the portable Application advertises all u destoextortion/misleading exercises whose lone object is to have a reason for hitting up the Applications in the prominence list. It turns out to be more incessant for Application designers to utilize terrible means, for example, expanding their Application deals or posting false App evaluations, to confer positioning extortion. It is vital to avoid positioning fraud as there is restricted comprehension and research in this field. Up till now, in this paper, we have given a comprehensive perspective of positioning misrepresentation and recommended positioning extortion identification framework.

Sentiment analysis is an application of natural language processing. It is also known as emotion extraction or opinion mining. This is a very popular field of research in text mining. The basic idea is to find the polarity of the text and classify it into positive, negative or neutral. It helps in human decision making. To perform sentiment analysis, one has to perform various tasks like subjectivity detection, sentiment classification, aspect term extraction, feature extraction etc. This paper presents the survey of main approaches used for sentiment classification.

Web spam is an illegal and immoral way to increase the ranking of web pages by deceiving search engine algorithms. Therefore, different methods have been proposed to detect and improve the quality of results. Since a web page can be viewed from two aspects of the content and the link, the number of extracting features is high. Thus, selection of features with high separating ability can be considered as a pre processing step in order to decrease computational time and cost. In this study, a new backward elimination approach is proposed for feature selection. The main idea of this method is measuring the impact of eliminating a set of features on the performance of a classifier instead of a single feature which is similar to the sequential backward selection. This method seeks for the largest feature subset that their omission from whole set features not only reduces the efficiency of the classifier but also improves it.

The IEEE International Conference on Data Mining (ICDM) has established itself as the world's premier research conference in data mining. It provides an international forum for presentation of original research results, as well as exchange and dissemination of innovative and

Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |

practical development experiences. The conference covers all aspects of data mining, including algorithms, software, systems, and applications. ICDM draws researchers, application developers, and practitioners from a wide range of data mining related areas such as big data, deeplearning, pattern recognition, statistical and machine learning, databases, data warehousing, data visualization, knowledge-based systems, and high-performance computing. By promoting novel, high-quality research findings, and innovative solutions to challenging data mining problems, the conference seeks to advance the state- of-the-art in data mining.

3 SYSTEMANALYSIS

3.1 EXISITINGSYSTEM

In the context of the increasing prevalence of mobile applications in daily life, the need to discern their safety has become more crucial than ever. Passing judgment on each application individually is impractical, prompting the development of a systematic approach. The proposed system relies on four key parameters—ratings, reviews, in-app purchases, and the presence of ads—to make predictions about the safety of mobile applications. A comparative analysis was conducted using three machine learning models: Decision Tree classifier, Logistic Regression, and Naïve Bayes. The evaluation focused on four performance metrics—F1score, Recall, Precision, and Accuracy.Typically, a good F1 score is considered to be greater than 0.7, while a recall score exceeding 0.5 is deemed satisfactory, along with higher precision and accuracy. The analysis revealed that the Decision Tree model emerged as a robust performer, boasting an accuracy rate of85%, an F1 score of 0.815, a Recall value of 0.85, and a precision of 0.87. These results position the Decision Tree model as a favorable choice for predicting the safety of mobile applications in this system.

The reliance on customer ratings and reviews introduces subjectivity into the assessment process. Different users may interpret and rate an app differently, making it challenging to ascertain genuine feedback from fraudulent ones. The existing system's focus on a narrow set of parameters overlooks other potential indicators of fraudulent behavior. Malicious apps can employ tactics that bypass the current detection criteria. Relying on manual assessment of each app's authenticity is impractical due to the vast number of applications available. This in efficiency hinders the ability to effectively identify and mitigate fraudulent apps.

3.2 PROPOSEDSYSTEM

We offer a solution that can detect these phony apps in the app store or on Google Play. In order to find out how likely it is that an app is trying to swindle its users, we provide a system that takes into account four features—in-app purchases, contains ads, ratings, and reviews. A thorough framework for detecting quality fraud has been suggested by Nevon projects; this framework can be enhanced with extra evidence produced by the domain. In terms of applying information algorithms to identify fraudulent applications, it is among the most cutting-edge efforts now underway. When it comes to identifying fraudulent applications, this tool only has an accuracy of 75-80%.

The main objective of the proposed system is to examine the methods used for detecting fraudulent applications in the Google Play Store and to distinguish between specific fraudulent applications, also known as spam applications, using the four parameter approach. The proposed method for detecting fraud or false applications is based on experimental study of several methodologies. Ad ratings, in-app purchases, evidence-based reviews, and other forms of proof will all be used to detect fraud in our system. Along with it, all four components to identify fraud are part of the development-based integration strategy.

By analysis, we found that our given proposed method provides 85% accuracy compared to other algorithms. While independent thinking still exists, the decision tree section performs better compared to other models such as the recession and the naïve bayes. It is an intuitive algorithm for separation problems. It is a reliable real-time guess, a setback problem. Decision trees can manage non- linear data sets effectively. It plays a role in decision-making in various fields of life, including engineering, social planning,

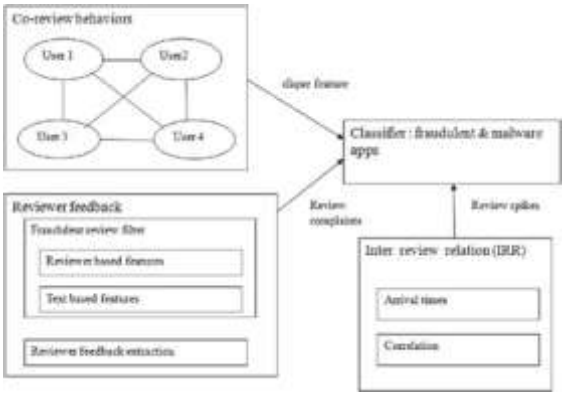
Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |

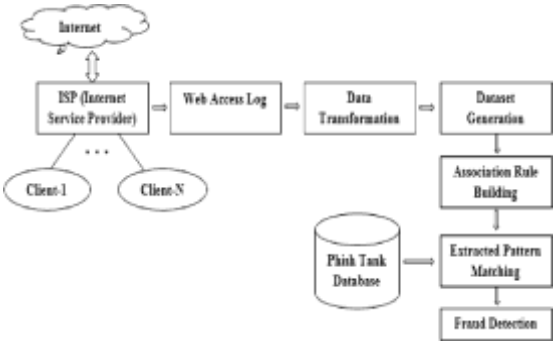
business, and even law. In this pursuit, we implemented various machine learning models, each yielding different accuracy results. Our thorough analysis revealed that our proposed method achieves an accuracy of 85%, outperforming alternative algorithms. Among these models, the decision tree algorithm stood out, demonstrating superior performance compared to other approaches like regression and naïve Bayes. The decision tree algorithm excels in solving complex separation problems, providing reliable real-time predictions, and efficiently handling non-linear datasets. Its applicability extends across diverse domains, including engineering, social planning, business, and legal contexts. Our system, with its comprehensive approach and utilization of advanced machine learning techniques, represents a significant step forward in combating fraudulent applications. Its robust accuracy and adaptable framework contribute to the ongoing efforts to ensure the integrity and security of app marketplaces.

The proposed system offers a range of advantages, including robust fraud detection, automated analysis, real-time decision-making user protection and a commitment to transparency. By addressing the challenges of fraudulent applications head-on, the system contributes to the overall integrity and security of app marketplaces.

4 SYSTEM DESIGN

4.1 SYSTEMARCHITECTURE





4.2 UMLDIAGRAMS

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 anotation. In the future, some form of method or process may also be added to or associated with, UML. The 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 nonsoftware systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important partof developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

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 concepts.
3. Be independent of particular programming languages and development process.

Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |

4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

Activity diagrams are graphical representations of workflows of stepwise activities 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.An activity diagram shows the overall flow of control.

5 IMPLEMENTATION

MODULES

1.DataCollection

2.DataPre-processingandfiltration

3.Modelcreationand training

```
#
==============================
==================
# PHISHING & FRAUD DETECTION USING ASSOCIATION RULES
#
==============================
==================

# Import necessary libraries
import pandas as pd
import re
from mlxtend.frequent_patterns import apriori, association_rules
from urllib.parse import urlparse

# ------------------------------------------------
# STEP 1: LOAD AND PARSE WEB ACCESS LOG
# ------------------------------------------------
def load_web_logs(file_path):
    """
    Loads raw web access logs and extracts useful fields.
    Example log line format:
    192.168.1.1 - - [12/Jan/2025:10:15:32] "GET http://example.com/login HTTP/1.1" 200 1234
    """
    logs = []
    with open(file_path, 'r') as f:
        for line in f:
            match = re.search(r'GET\s+(http[s]?://[^\s]+)', line)
            if match:
                url = match.group(1)
                logs.append(url)
    return logs

# ------------------------------------------------
# STEP 2: DATA TRANSFORMATION
# ------------------------------------------------
def transform_data(logs):
    """
    Extract domain and path from URLs for feature generation.
    """
    data = []
    for url in logs:
        domain = urlparse(url).netloc
        path = urlparse(url).path
        data.append({'domain': domain, 'path': path})
    df = pd.DataFrame(data)
    return df

# ------------------------------------------------
# STEP 3: DATASET GENERATION
# ------------------------------------------------
def generate_dataset(df):
    """
    Create one-hot encoded dataset for rule mining.
    """
    # Create binary indicators for unique paths
    df['feature'] = df['domain'] + df['path']
    basket = (df.groupby(['domain', 'feature'])['feature']
```

Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |

```
.count().unstack().reset_index().fillna(0)
            .set_index('domain'))
    basket = basket.applymap(lambda x: 1 if x > 0 else 0)
    return basket

# ------------------------------------------------
# STEP 4: ASSOCIATION RULE BUILDING
# ------------------------------------------------
def build_association_rules(basket):
    """
    Use Apriori algorithm to find frequent patterns in domains.
    """
    frequent_items = apriori(basket, min_support=0.2, use_colnames=True)
    rules = association_rules(frequent_items, metric="lift", min_threshold=1)
    return rules

# ------------------------------------------------
# STEP 5: PHISHTANK DATABASE MATCHING
# ------------------------------------------------
def load_phishtank_database(phishtank_path):
    """
    Load known phishing URLs (CSV or text file).
    """
    phishtank = pd.read_csv(phishtank_path)
    return set(phishtank['url'])

def match_patterns(df, phishtank_db):
    """
    Compare extracted domains with known phishing entries.
    """
    df['is_phishing'] = df['domain'].apply(lambda x: any(p in x for p in phishtank_db))
    return df

# ------------------------------------------------
# STEP 6: FRAUD DETECTION
# ------------------------------------------------
def detect_fraud(df):
    """
    Flag suspicious domains based on rule patterns or PhishTank hits.
    """
    frauds = df[df['is_phishing'] == True]
    print(f"□ Detected {len(frauds)} fraudulent entries.")
    return frauds

# ------------------------------------------------
# MAIN PIPELINE
# ------------------------------------------------
def main():
    # Paths (example)
    web_log_path = "web_access.log"
    phishtank_path = "phishtank.csv"

    # Step 1: Load and parse logs
    logs = load_web_logs(web_log_path)

    # Step 2: Transform data
    df = transform_data(logs)

    # Step 3: Generate dataset
    basket = generate_dataset(df)

    # Step 4: Build association rules
    rules = build_association_rules(basket)
    print("□ Association Rules Generated:")
    print(rules.head())

    # Step 5: Match with PhishTank
    phishtank_db = load_phishtank_database(phishtank_path)
    df = match_patterns(df, phishtank_db)

    # Step 6: Fraud detection
    frauds = detect_fraud(df)
    print(frauds[['domain', 'is_phishing']])

# ------------------------------------------------
```

```
# RUN SCRIPT
# ----------------------------------------------
if __name__ == "__main__":
    main()
```

6 RESULTS/DISCUSSIONS

SYSTEMTESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercisingsoftwarewiththeintentofensuringthatthe Softwaresystemmeetsitsrequirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPESOF TESTS

Unittesting

Unit testing involves the design of test cases that validate that the internal program logic is functioningproperly,andthatprograminputsproduc evalidoutputs.Alldecisionbranchesand internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structuraltesting,thatreliesonknowledgeofitsconstr uctionandisinvasive.Unittestsperform basic tests at component level and test a specific business process, application, and/or system configuration.Unittestsensurethateachuniquepatho fabusinessprocessperformsaccurately to the documented specifications and contains clearly defined inputs and expected results.

Integrationtesting

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

Functionaltest

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and usermanuals. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

SystemTest

The complete integrated software system must be tested to guarantee it meets all requirements. In order to provide known and predictable results, it tests a setup. System integration tests that are configuration-oriented are one kind of system testing. The foundation of system testing lies in the documentation and flow of processes, with an emphasis on the integration points and pre-driven process connections.

WhiteBox Testing

WhiteBoxTestingisatestinginwhichinwhichthesoft waretesterhasknowledgeoftheinner workings,structureandlanguageofthesoftware,oratl eastitspurpose.Itispurpose.Itisused to test areas that cannot be reached from a black box level.

BlackBox Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structureor languageofthemodulebeing tested. Black box tests, as most otherkinds oftests, must be written from a definitive source document, such as specification or requirements document,suchasspecificationorrequirementsdocu ment.Itisatestinginwhichthesoftware undertestistreated,asablackbox.youcannot“see”int oit.Thetestprovidesinputsand responds to outputs without considering how the software works.

UnitTesting

Unittestingisusuallyconductedaspartofacombinedc odeandunittestphaseofthesoftware lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |

TESTINGSTRATEGIES

Fieldtestingwill beperformedmanuallyandfunctionaltestswillbewrittenindetail.

Testobjectives

- Allfieldentriesmustwork properly.
- Pagesmust beactivatedfromtheidentifiedlink.
- Theentryscreen, messagesandresponsesmust notbedelayed.

Featurestobe tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

IntegrationTesting

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. componentsinasoftwaresystemor–onestepup–softwareapplicationsatthecompanylevel – interact without error.

TestResults: Allthetestcasesmentionedabovepassedsuccessfully.Nodefects encountered.

AcceptanceTesting

UserAcceptanceTestingisacriticalphaseofanyprojectandrequiressignificantparticipation by the end user. It also ensures that the system meets the functional requirements.

TestResults: Allthetestcasesmentionedabovepassedsuccessfully.Nodefects encountered.

SCREENSHOTS

FigHomePage

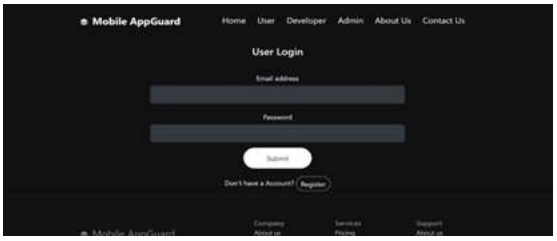


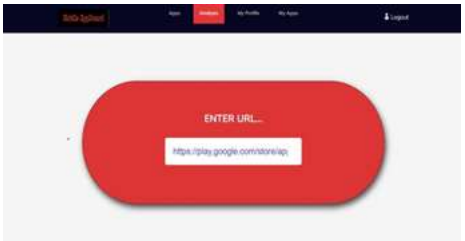
Fig: User Login Page

Fig AdminLogin Page

FigAdminDashboard

UPLOAD:Onthispage, theuserofthesystemcanupload.csvfile.Theuserhastoselectthefileby clicking on the Choose file button

Fig AdminUpload Dataset



FigLinkUpload Page

Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |

Fig: Negative Result

FigPositiveResult

CONCLUSION

Concerns about safety have grown in importance in today's technologically advanced world. While the widespread use of digital platforms has undoubtedly improved people's lives, it has also introduced new risks to people's safety. A problem that has recently arisen is the abundance of fake apps in Google's app stores. In addition to endangering users' privacy and sensitive data, these harmful apps undermine the security of the entire digital ecosystem. Regardless of this serious worry, our research has been focused on creating a strong solution to identify fraudulent software. App ratings, review scores, in-app purchases, and new content additions are the four main metrics we've been tracking in our all-encompassing strategy. Through careful examination of these vital aspects, our goal is to differentiate between valid programs and those that could harm users.

Our strategy is built around using complex algorithms to determine the validity of apps. For this purpose, we compared three well-known algorithms: logistic regression, naive bayes, and decision trees. We were astounded to find that the Decision Tree algorithm achieved an astounding 85% accuracy, the highest rate of any algorithm we tested.

With its built-in scalability and measureability, our proposed system is a versatile tool for fraud detection in many different fields. Its ability to detect fraudulent activity is further improved by its modular design, which lets you add more domain-specific evidence. The system's adaptability to future issues and developing fraud strategies is highlighted by its extensibility. We have proven the efficacy of the four-proposed approach by conducting extensive experiments and analyzing the results empirically. The algorithmic detection mechanism's evaluated performance and accuracy demonstrate its potential to be an essential instrument in the fight against fraudulent operations. Also, by helping to standardize fraud detection procedures, our solution makes it easier to tell good apps from bad ones.

At the end of the day, our fresh method has great potential for making app stores more secure. The overall integrity of online markets is enhanced by our system's ability to detect and rank fake applications. Confidence among users, privacy, and sensitive data are all enhanced as a result. Staying ahead of security solutions and protecting consumers in an increasingly linked digital world is our goal, and our research is a testament to that. Technology is always evolving.

FUTURE SCOPE

Beyond Decision trees, advanced models like Random Forest and deep learning could bolster accuracy. Deeper feature engineering, incorporating NLP for app descriptions and analyzing user interactions could yield richer insights. Real-time analysis, transparent AI decisions, and continuous learning are vital for adaptive systems. Integrating user feedback, geolocation data, and cross-platform detection can expand scope. Regulatory and collaboration with experts ensure ethical and effective development. By exploring blockchain and embracing smart contracts, you can fortify app verification. As technology advances, our project has agility and innovation will be essential to combating emerging fraud tactics.

REFERENCES

1. EstherNowroji,Vanitha,“DetectionOfFraudRankingForMobileAppUsingIPAddress Recognition Technique”, vol. 4.
2. JavvajiVenkataramaiah,BommavarapuSu

shen,Mano.R,Dr.GladispushpaRathi,"Anenhanced mining leading session algorithm for fraud app detection in mobile applications"

3. S.R.Srividhya,S.Sangeetha– "AMethodologyto DetectFraud AppsUsingSentimentAnalysis"
4. Keerthana.B,Sivashankari.KandShaisthaTabasum.S,"DetectingMalwaresandSearch RankFraud in Google Search Using Rabin Karp Algorithm", IJARSE,7(02), 2018, pp.504-527.
5. ShashankBajaj,NikhilNigam,PriyaVandana,SrishtiSingh,"Detectionoffraudappsusing sentiment analysis", International Journal of Innovative Science and Research Technology.
6. HarpreetKaur,VeenuMangatandNidhi,— "ASurveyofSentimentAnalysis techniques"
7. InternationalconferenceonI-SMAC(IoTinSocial,Mobile,Analyticsand Cloud)(I-SMAC),2017, pp.921
8. Jing Wan, Mufan Liu, Junkai Yi and Xuechao Zhang, "Detecting Spam Webpages through Topic andSemanticsAnalysis",IEEEGlobalSummitonComputerandInformationTechnology(GSCIT), 2015, pp. 83-92.
9. Navdeep Singh, Prashant Kr.Pandey and Mr.Srinivasan,— "ImprovedDiscoveryofRatingFake for Cellular Apps", IEEE International Conference on Science Technology Engineering and Management (ICONSTEM), 2016, pp. 135-140.
10. Weiman Wang, Restricted Boltzmann Machine. GitHub. Aug 2017. [Online] Available: https://github.com/aaxwaz/Fraud-detection-usingdeep-learning/blob/master/rbm/rbm.py.
11. DubeyVeena,G.D.(2016).SentimentAnalysisBasedonOpinionClassificationTechniques:A Survey .International Journal of Advanced Research in Computer Science and Software Engineering, 5358.
12. RankingfraudMiningpersonalcontext-awarepreferencesformobileusers.H.Zhu,E.Chen,K. Yu, H. Cao, H. Xiong, and J. Tian. In Data Mining (ICDM), 2012 IEEE 12th International Conference on, pages1212–1217, 2012.
13. NandimathJyoti,K.B.(2017).EfficientlyDetectingandAnalyzingSpamReviewsUsing LiveData Feed. International Research Journal of Engineering and Technology (IRJET) , 1421-1424.
14. Detectingproductreviewspammersusingratingbehaviors.E.-P.Lim,V.-A.Nguyen,N.Jindal,B. Liu,andH.W.LauwInProceedingsofthe19thACMinternationalconferenceonInformationand knowledge management, CIKM "10 pages 939–948, 2013.
15. Detection for mobile apps H. Zhu, H. Xiong, Y. Ge , and E. Chen. A holistic view. In Proceedings of the 22nd AC MinternationalconferenceonInformationandknowledgemanagement,CIKM"13.

Received: 20-09-2025 | Accepted: 24-10-2025 | Published: 31-10-2025 |