http://www.jctjournals.com ISSN(ONLINE):2278 – 3814

Volume-13, Issue-08, August 2024
JOURNAL OF COMPUTING TECHNOLOGIES (JCT)
International Journal
Page Number: 01-09

A Review of Efficient Machine Learning Methods for Google Play Store App Rating Prediction

Deepika [1], Prof. Rani Kushwaha[2] Prof. Jayshree Boaddh [3],
M.Tech Scholar [1], Assistant Professor [2] HOD CSE [3]
[1, 2, 3] Department of Computer Science & Engineering
[1, 2] Mittal Institute of Technology, Bhopal (M.P.)
[3]Vaishnavi Institutes of Technology and Science, Bhopal (M.P.)
[1]deepikapal6267@gmail.com [2]ranikush996@gmail.[3]jayshree.boaddh@gmail.com

***Abstract*— with the increasing number of mobile applications available on the Google Play Store, accurate and reliable app rating prediction is essential for guiding consumer choices and helping developers enhance app quality. This review paper examines various machine learning techniques applied to the task of rating prediction for Google Play Store apps. Focusing on methods that leverage user reviews, app-specific information (such as category, size, and version), and developer updates, we analyze the performance of approaches including Random Forest, Gradient Boosting, and Neural Networks.**

***Keywords*— Machine Learning (ML), App Rating Prediction, Google Play Store, Sentiment Analysis, App Metadata, Random Forest (RF).**

I. INTRODUCTION

Machine learning approaches are fundamental as far as we're concerned to deal with various issues. Machine learning has various applications in various viewpoints and has unbelievable progression potential. It is unsurprising that machine learning could set up ideal theories to explain its displays. Meanwhile, its abilities of solo learning will be improved since there is a lot of data in the world anyway adding names to all of them isn't important. It is also guessed that brain framework designs will end up being progressively eccentric with the objective that they can isolate every one of the more semantically significant features. Also, significant learning will solidify with help adjusting better and we can use these focal points to accomplish more tasks.

Figure 1: Mobile App

In the present situation we can see that mobile apps playing a significant job in any singular's life. It has been seen that the improvement of the mobile application publicize amazingly affects progressed development. Having said that, with the reliably creating adaptable application grandstand there is furthermore a prominent climb of compact application originators definitely achieving high as can be pay by the overall versatile application industry.

With tremendous test from wherever all through the globe, it is fundamental for a creator to understand that he is going on in the right heading. To hold this pay and their spot in the market the application creators might have to sort out some way to stick into their current position. The Google Play Store is seen to be the greatest application stage. It has been seen that notwithstanding the way that it makes in excess of two overlap the downloads than the Apple App Store yet makes simply a huge piece of the money stood out from the App Store. Along these lines, I scratched data from the Play Store to coordinate our assessment on it.

With the quick improvement of cutting edge cells, versatile applications (MobileApps) have ended up being essential bits of our lives. In any case, it is problematic as far as we're concerned to track with the reality and to comprehend everything about the apps as new applications are entering market every day. It is represented that Android1market accomplished an enormous part of 1,000,000 applications in September 2011. Beginning at

now, 0.675 million Android applications are open on Google Play App Store. Such a ton of applications are apparently an exceptional entryway for clients to buy from a wide assurance expand. We trust flexible application clients consider online application overviews as an important effect for paid applications. It is pursuing for a likely client to examine every one of the scholarly comments and rating to make a decision. Furthermore, application engineers experience issues in finding how to further develop the application execution reliant upon as a rule alone and would benefit by figuring out the an enormous number of printed comments.

GOOGLE PLAY STORE AND MOBILE APP

Google Play, also known as the Google Play Store and Android Market, is a computerized distribution system created by Google that serves as the primary app store for Android devices and its subsystems, as well as Chrome OS. It offers access to programs, music, books, movies, and TV programs, which can be accessed through an internet browser or through Android and iOS applications.

Applications can be downloaded for free or at a cost, and can be downloaded directly from the Google Play Store mobile app or by delivering the app to a device from the Google Play site. However, the store has faced security issues, with harmful software being accepted and transmitted to the store and downloaded by customers.

Google Play was launched on June 6, 2012, and has since been rebranded several times, including Google News, YouTube Music, and Google TV. In 2022, Play Games is set to remove its mobile app for an Android emulator for Windows, while Play Books will be replaced by an independent mobile app.

With over 3.5 million Android apps in 2017, Google Play has grown to over 3 million apps. Developers can distribute applications in over 150 countries, with delivery and labor costs taking 15% of the application cost, and designers receiving 85%. Customers can pre-order apps, movies, music, books, and games, and request discounts within 48 hours after purchase.

MACHINE LEARNING TECHNIQUES

Machine learning (ML) is the scientific study of algorithms and statistical models that PC systems employ to do a job without explicit instructions, utilizing patterns and deduction. It is part of computerized reasoning. Machine learning algorithms may anticipate or decide without being explicitly programmed by constructing a numerical model from sample information, called "preparing information". Machine learning algorithms are employed in many applications, such as email screening and PC vision, when conventional calculations are difficult or impossible.

Decision Tree

Decision trees are built using recursive partitioning to classify the data, i.e., by splitting the training set into distinct nodes, where one node contains all of or most of one category of the data. A decision tree can be constructed by considering the attributes one by one:

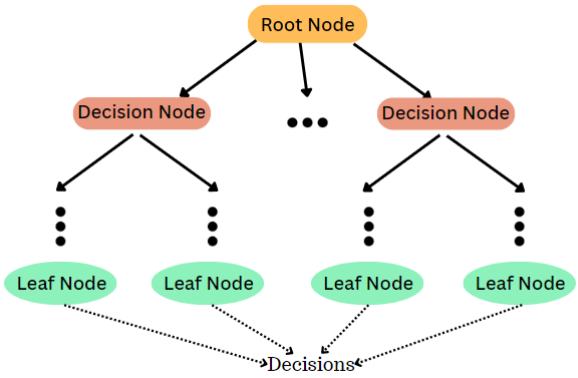

Figure 2: Decision tree

- First, choose an attribute from our dataset.
- Calculate the significance of the attribute in the splitting of the data.
- Next, split the data based on the value of the best attribute,
- Then go to each branch and repeat it for the rest of the attributes.
- After building this tree, you can use it to predict the class of unknown cases Decision trees are about testing an attribute and branching the cases based on the result of the test:

1. Each internal node corresponds to a test
2. Each branch corresponds to a result of the test
3. Each leaf node assigns a patient to a class

Naïve Bayes

In Naïve Bayes classifier is a supervised calculation which classifies the dataset on the basis of Bayes hypothesis. The Bayes hypothesis is a standard or the numerical idea that is used to get the likelihood is called Bayes hypothesis. Bayes hypothesis requires some free assumption and it requires autonomous variables which is the basic assumption of Bayes theorem.

Logistic Regression

Logistic regression is a classification calculation for unmitigated variables. Logistic regression is analogous to linear regression, however tries to foresee a clear cut or discrete objective field, such as 0 or 1, yes or no, and so forth., instead of a numeric one. Subordinate variables should be continuous. In the event that clear cut, they should be sham or marker coded. This means we need to transform them to some continuous worth. Logistic regression can be used for both twofold classification and multi-class classification. Sigmoid functions are a principle part of logistic regression.

K-Nearest Neighbors Algorithm (KNN)

The K-Nearest Neighbors is a calculation for supervised learning and is a classification calculation that takes a lot of named points and uses them to figure out how to name different points. This calculation classifies cases based on their similarity to different cases. In K- Nearest Neighbors, information points that are close to one another are said to

be neighbors. K-Nearest Neighbors is based on this worldview. Thus, the distance between two cases is a measure of their dissimilarity. There are various ways to figure the similarity or conversely, the distance or dissimilarity of two information points. For instance, this should be possible using Euclidean distance.

Random Forest

Random Forest is an adaptable, easy to use machine learning calculation that produces, even without hyper-parameter tuning, an extraordinary result most of the time. It is also one of the most used algorithms, because its simplicity and the way that it tends to be used for both classification and regression tasks. Right now, will realize, how the random forest calculation works and several other significant things about it.

One major bit of leeway of random forest is, that it tends to be used for both classification and regression problems, which structure most of current machine learning systems.

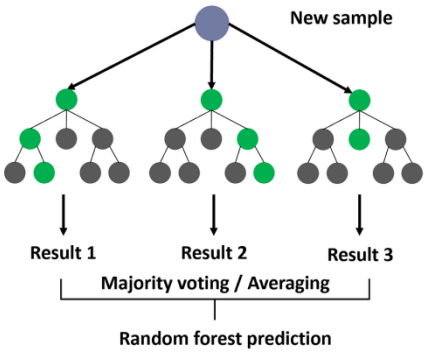

Figure 3: Random forest

II. LITERATURE REVIEW

R. Gomes et al., [1] People need uses for daily exercise. As progress occurs, interest in its causes grows. We satisfy Google Play Store accomplishment requirements using classifiers. KNN and Irregular Backwoods were used to quantify application relapses using speculation, connection, and relapse metrics. This effort will create induction motors to predict usage appraisals using KNN and Irregular Backwoods relapse. The Arbitrary Woodland outperformed the KNN.

C. Zhu et al., [2] Deep models for active visitor clicking percentage expectation need element insertion learning and element communication showing. Most deep CTR models suffer from these three difficulties. Include cooperations are either physically planned or listed. Second, all component associations communicate similarly. In most models, highlights with the same implanting size cause memory failure. For these three issues, we propose Programmed Connection Machine (Point) with three main parts, Element Communication Search, Collaboration Capability Search (Uncertainties), and Implanting Aspect Search, to naturally choose large component cooperations, fitting communication capabilities, and essential installing aspects in a bound together system. FIS recognizes various orders of fundamental element cooperations and prunes futile ones; Uncertainties chooses fitting connection capabilities for each component communication in a learnable manner; and EDS naturally assesses component implanting size. Disconnected examines three massive datasets to support Point's unmatched presentation. Point improves DeepFM model by 4.4% in CTR in a three-week online A/B test in a regular application market recommendation administration.

G. S. Bhat et al.,[3] Author introduce an ML-based asthma risk expectation device in this study. The whole tool is run on a phone as an m-healthcare app using Web of Things. External tools like top stream meters evaluate top expiratory stream rates, which are asthama risk markers. We use the PEFR to determine a link between indoor PM and outside climate. Compared to the highest stream value achieved by each participant, PEFR findings are divided into three classes: 'Green' (Safe), 'Yellow' (Moderate Gamble), and 'Red' (High Gamble). Convolutional brain organization engineering links indoor PM and climate data to PEFR values. The root mean square and mean outright error exactness measurements of the suggested technique are compared to cutting-edge deep brain organization (DNN) methods. Over other writing styles, these presentation metrics are favored for the suggested strategy. The whole process is an app on a phone. Data is collected using an IoT framework and Raspberry Pi. For asthma attack prevention, this device may be cost-effective.

Z. Wu et al., [4] Application portrayal inspection has several uses in computer programming. Besides the unpredictability of regular language, insufficient consent semantics make it difficult to predict capabilities and authorized usages from program portrayals. More specifically, developers purposefully abridge application class functions due to the specified amount of characters, and consents are often over-guaranteed. These are the main reasons application portrayals provide false advantages in predicting authorizations. In earlier studies that didn't assist engineers improve application descriptions and avoid security risks, such unmentioned authorizations should be considered suspicious. We propose the FideDroid, a framework to identify class-based normal consents to balance core functionality while analyzing application depiction consistency. Our structure expands the named application depictions dataset for consent forecasting. FideDroid compares collected and used authorizations to identify suspicious and unnecessary ones based on anticipation. It helps developers refine application portrayals and track consent use. We found in our experiments on large real-world applications that class-based normal consents may cover additional unmentioned functionality without considering all conceivable authorizations during application depiction inspection. Three factors caused the discrepancy between portrayals and approved uses: 1) hard-copy human mediations; 2) bad consent processes; and 3) productive engineers. These findings will help designers enhance app depictions and consent use.

Z. Shen et al., [5] This study predicts which apps a customer will access on her phone in the next timeframe. This data is essential for most mobile phone actions, such

as application pre-stacking and content pre-reserving, to improve customer experience. It's hard to create a model that accurately captures the complex environment and anticipates several uses. DeepAPP, a deep support learning structure, learns a realistic model prophetic brain network from verified application usage data. Web-based refreshing is meant to adapt the foresight organization to time-varying application usage. To turn DeepAPP into a viable deep support learning framework, a setting portrayal strategy for complex relevant climate, an overall specialist to overcome information sparsity, and a lightweight customized specialist to reduce expectation time are addressed. Wide testing on an anonymous application use dataset show that DeepAPP has high accuracy (70.6% and review of 62.4%) and reduces cutting edge expectation season by 6.58 times. A 29-person field study shows DeepAPP may reduce application startup time.

Z. Xu et al.,[6] In the nick of time (JIT) bug expectation is a strong quality confirmation movement that detects if a code change will introduce problems into the portable program and provides concise feedback to pros for survey. Some flexible apps can't acquire enough identified bug information, thus cross-application models may be used. This paper proposes CDFE, a cross-trio deep element implanting approach for cross-application JIT bug expectation. The CDFE technique integrates a cutting-edge cross-trio misfortune capacity into a deep brain structure to learn cross-application data component representation. This unpleasant capacity works with cross-application learning and means to get acquainted with another element space to shorten the distance of commit events with identical names and expand the distance of commit cases with different markings. To reduce cross-application bug information inconsistency, this misfortune feature assigns larger loads to cross-application example matches than intra-application event matches. Our CDFE approach is tested on a benchmark bug dataset comprising 19 portable apps with two exertion aware pointers. Our CDFE method outperforms 14 conventional techniques in 342 cross-application matches.

III. PROBLEM IDENTIFICATION

T The mobile app developing grows rapidly. The customer gives the reviews after using the app. The mobile app developers can understand the target audience easily.

- Meeting user requirements.
- Choosing the operating system.
- Choosing the development platform.
- Security.

After developing and using the app, customer rating is very useful to success the app. Therefore during the literature survey some of the observation is carried out, which is as followings-

- Low accuracy rate of true data prediction from given dataset.
- Using traditional System Analysis alone not sufficient for proper feature extraction.
- More classification error.
- No adaptive approach to prediction of true rating of apps.
- Sensitivity, Specificity, Precision, Recall and F measure values is not good optimized.

IV. EXPECTED SOLUTION

Expected Solution:

- **Existing Models**: Summarize current machine learning and deep learning models used in rating prediction, including supervised learning algorithms (e.g., Random Forest, Gradient Boosting), ensemble methods, and neural networks.
- **Feature Engineering**: Highlight how features like user review sentiment, app metadata (category, size, update frequency), and historical ratings are used for predictions. Discuss innovative feature extraction techniques, such as Natural Language Processing (NLP) for sentiment analysis and app description mining.
- **Evaluation Metrics**: Discuss common evaluation metrics used in the studies, such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), and highlight why these metrics are suitable for rating prediction.

Proposed Workflow for Efficient Rating Prediction:

- **Data Collection and Preprocessing**: Suggest a pipeline that handles missing values, feature normalization, and outlier removal, which are essential steps in cleaning the app data. Explain preprocessing methods for text data from user reviews, such as tokenization and sentiment scoring, and justify why these steps contribute to model efficiency.
- **Feature Selection**: Identify key features that correlate strongly with app ratings. Methods like recursive feature elimination, principal component analysis, or LASSO regression could be highlighted for reducing dimensionality and improving computational efficiency.
- **Sentiment Analysis Integration**: Emphasize the role of sentiment analysis in extracting valuable information from user reviews, enhancing prediction accuracy when combined with app metadata.

Efficient Machine Learning Algorithms:

- **Model Selection and Comparison**: Compare the performance of different algorithms, including Random Forest, Gradient Boosting, and Neural Networks. Explain why some models may be more suitable for this task due to their handling of high-dimensional data, interpretability, or computation time.
- **Performance Optimization**: Discuss techniques to improve model efficiency, such as hyperparameter tuning (using Grid Search or Random Search) and ensemble methods that leverage the strengths of multiple algorithms to improve prediction stability and accuracy.

Expected Results and Evaluation:

- Present anticipated outcomes, showing that models combining app metadata and user sentiment achieve lower MAE and RMSE, outperforming traditional models that use metadata alone.
- Emphasize the importance of a balanced approach between prediction accuracy and computational cost, which can make the solution more practical for real-world applications.

Implications for Developers and Consumers:

- **For Developers**: Explain how rating prediction models can guide app updates and improvements, focusing on aspects that matter most to users.
- **For Consumers**: Highlight the benefits of accurate rating predictions in helping users make informed choices.

V. Conclusion

In conclusion, the rapid growth of mobile applications on the Google Play Store makes reliable app rating prediction an invaluable tool for both consumers and developers. This review highlights the effectiveness of machine learning techniques—such as Random Forest, Gradient Boosting, and Neural Networks—in enhancing rating prediction accuracy. Key factors, including user reviews, app metadata (category, size, version), and developer updates, emerge as critical inputs for these models, with sentiment analysis from user reviews playing a significant role in refining predictions.

REFERENCES

[1.] R. Gomes da Silva, J. de Oliveira Liberato Magalhães, I. R. Rodrigues Silva, R. Fagundes, E. Lima and A. Maciel, "Rating Prediction of Google Play Store apps with application of data mining techniques," in IEEE Latin America Transactions, vol. 19, no. 01, pp. 26-32, January 2021, doi: 10.1109/TLA.2021.9423823.

[2.] C. Zhu et al., "AIM: Automatic Interaction Machine for Click-Through Rate Prediction," in IEEE Transactions on Knowledge and Data Engineering, doi: 10.1109/TKDE.2021.3134985.

[3.] G. S. Bhat et al., "Machine Learning-Based Asthma Risk Prediction Using IoT and Smartphone Applications," in IEEE Access, vol. 9, pp. 118708-118715, 2021, doi: 10.1109/ACCESS.2021.3103897.

[4.] Z. Wu, X. Chen, M. U. Khan and S. U. -J. Lee, "Enhancing Fidelity of Description in Android Apps With Category-Based Common Permissions," in IEEE Access, vol. 9, pp. 105493-105505, 2021, doi: 10.1109/ACCESS.2021.3100118.

[5.] Z. Shen, K. Yang, Z. Xi, J. Zou and W. Du, "DeepAPP: A Deep Reinforcement Learning Framework for Mobile Application Usage Prediction," in IEEE Transactions on Mobile Computing, doi: 10.1109/TMC.2021.3093619.

[6.] Z. Xu et al., "Effort-Aware Just-in-Time Bug Prediction for Mobile Apps Via Cross- Triplet Deep Feature Embedding," in IEEE Transactions on Reliability, doi: 10.1109/TR.2021.3066170.

[7.] K. Zhao, Z. Xu, T. Zhang, Y. Tang and M. Yan, "Simplified Deep Forest Model Based Just-in-Time Defect Prediction for Android Mobile Apps," in IEEE Transactions on Reliability, vol. 70, no. 2, pp. 848-859, June 2021, doi: 10.1109/TR.2021.3060937.

[8.] G. Aceto, G. Bovenzi, D. Ciuonzo, A. Montieri, V. Persico and A. Pescapé, "Characterization and Prediction of Mobile-App Traffic Using Markov Modeling," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 907-925, March 2021, doi: 10.1109/TNSM.2021.3051381.

[9.] Y. Zhang, J. Liu, B. Guo, Z. Wang, Y. Liang and Z. Yu, "App Popularity Prediction by Incorporating Time-Varying Hierarchical Interactions," in IEEE Transactions on Mobile Computing, doi: 10.1109/TMC.2020.3029718.

[10.] Q. Zhu, Q. Sun, Z. Li and S. Wang, "FARM: A Fairness-Aware Recommendation Method for High Visibility and Low Visibility Mobile APPs," in IEEE Access, vol. 8, pp. 122747-122756, 2020, doi: 10.1109/ACCESS.2020.3007617.

[11.] S. şahın, A. M. Cipriano, C. Poulliat and M. Boucheret, "Iterative Decision Feedback Equalization Using Online Prediction," in IEEE Access, vol. 8, pp. 23638-23649, 2020, doi: 10.1109/ACCESS.2020.2970340.

[12.] S. Rezaei, B. Kroencke and X. Liu, "Large-Scale Mobile App Identification Using Deep Learning," in IEEE Access, vol. 8, pp. 348-362, 2020, doi: 10.1109/ACCESS.2019.2962018.

[13.] S. Zhao et al., "Gender Profiling From a Single Snapshot of Apps Installed on a Smartphone: An Empirical Study," in IEEE Transactions on Industrial Informatics, vol. 16, no. 2, pp. 1330-1342, Feb. 2020, doi: 10.1109/TII.2019.2938248.

[14.] C. Min et al., "Scalable Power Impact Prediction of Mobile Sensing Applications at Pre-Installation Time," in IEEE Transactions on Mobile Computing, vol. 19, no. 6, pp. 1448-1464, 1 June 2020, doi: 10.1109/TMC.2019.2909897.

[15.] B. Guo, Y. Ouyang, T. Guo, L. Cao and Z. Yu, "Enhancing Mobile App User Understanding and Marketing With Heterogeneous Crowdsourced Data: A Review," in IEEE Access, vol. 7, pp.

68557-68571, 2019, doi: 10.1109/ACCESS.2019.2918325.

[16.] G. Meng, M. Patrick, Y. Xue, Y. Liu and J. Zhang, "Securing Android App Markets via Modeling and Predicting Malware Spread Between Markets," in IEEE Transactions on Information Forensics and Security, vol. 14, no. 7, pp. 1944-1959, July 2019, doi: 10.1109/TIFS.2018.2889924.

[17.] Y. Ouyang, B. Guo, X. Lu, Q. Han, T. Guo and Z. Yu, "CompetitiveBike: Competitive Analysis and Popularity Prediction of Bike-Sharing Apps Using Multi- Source Data," in IEEE Transactions on Mobile Computing, vol. 18, no. 8, pp. 1760- 1773, 1 Aug. 2019, doi: 10.1109/TMC.2018.2868933.

[18.] C. Fang, Y. Wang, D. Mu and Z. Wu, "Next-App Prediction by Fusing Semantic Information With Sequential Behavior," in IEEE Access, vol. 6, pp. 73489-73498, 2018, doi: 10.1109/ACCESS.2018.2883377.

[19.] E. Liotou, K. Samdanis, E. Pateromichelakis, N. Passas and L. Merakos, "QoE-SDN APP: A Rate-guided QoE-aware SDN-APP for HTTP Adaptive Video Streaming," in IEEE Journal on Selected Areas in Communications, vol. 36, no. 3, pp. 598-615, March 2018, doi: 10.1109/JSAC.2018.2815421.

[20.] Y. Kwon et al., "Mantis: Efficient Predictions of Execution Time, Energy Usage, Memory Usage and Network Usage on Smart Mobile Devices," in IEEE Transactions on Mobile Computing, vol. 14, no. 10, pp. 2059-2072, 1 Oct. 2015, doi: 10.1109/TMC.2014.2374153.

