International Journal of Advanced Technology & Engineering Research (IJATER)
www.ijater.com

An Efficient Machine Learning Technique for Rating Prediction Of Google Play Store Apps

Veena Vadinee Raikwar
M.Tech Scholar
CSE Department
NIIST Bhopal

Prof. Mahendra Sahare
Associate Professor
CSE Department
NIIST Bhopal

Prof. Anurag Shrivastava
Associate Professor & HOD
CSE Department
NIIST Bhopal

***Abstract*—The vast landscape of mobile applications, user ratings play a crucial role in determining an app's success and popularity. Predicting these ratings accurately not only aids users in making informed decisions but also assists developers in improving their apps. This paper presents an efficient machine learning technique tailored for rating prediction of Google Play Store apps. Our approach leverages a combination of advanced machine learning algorithms, feature analysis, and data pre-processing methods to achieve robust performance. Simulation results shows that the effectiveness of proposed technique through extensive experimentation on datasets, showcasing its ability to accurately forecast app ratings.**

Keywords— Machine Learning, Google,Play Store, ,Online, Rating .

I. INTRODUCTION

In recent years, the proliferation of mobile applications has revolutionized various aspects of everyday life, offering users a myriad of functionalities and services at their fingertips. With millions of apps available across different platforms, such as the Google Play Store, Apple App Store, and others, users are often confronted with the challenge of selecting the most suitable ones to full-fill their needs. In this context, user ratings serve as a primary source of information, providing insights into an app's quality, usability, and overall satisfaction.

For developers and app publishers, user ratings hold immense significance, as they directly influence an app's visibility, downloads, and revenue generation potential. Positive ratings not only attract new users but also contribute to higher rankings within app stores, leading to increased exposure and organic growth. Conversely, negative ratings can deter potential users and undermine the reputation and credibility of an app.

Given the critical role of user ratings in shaping the success of mobile applications, accurate prediction of these ratings emerges as a pertinent research area. By forecasting the likely rating that an app would receive, developers can preemptively identify areas for improvement, optimize features, and enhance user satisfaction. Moreover, users can benefit from rating prediction models by making informed decisions based on projected ratings rather than solely relying on existing ratings.

In this paper, we propose an efficient machine learning technique specifically tailored for rating prediction of Google Play Store apps. Our approach integrates a diverse set of machine learning algorithms, including but not limited to regression, ensemble methods, and deep learning, to capture intricate patterns and relationships inherent in app data. Furthermore, we employ advanced feature engineering techniques to extract relevant information from app metadata, user reviews, and other contextual factors.

Through rigorous experimentation on real-world datasets sourced from the Google Play Store, we evaluate the performance of our proposed technique and compare it against existing approaches. Our results demonstrate the efficacy and reliability of the proposed method in accurately predicting app ratings across diverse categories and user demographics. We believe that our approach holds significant promise in enhancing the app development lifecycle, empowering

developers to create more engaging and user-centric experiences while assisting users in making informed choices amidst the vast array of available options.

II. METHODOLOGY

The methodology or the flow of the work is as followings-

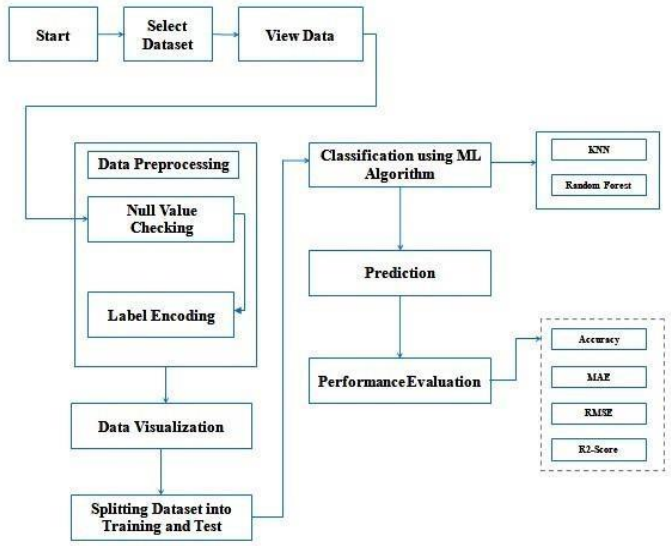

Figure 1: Flow Chart

1. Data Acquisition and Preprocessing:

We begin by collecting the necessary data from the Google Play Store, including app metadata such as app name, category, size, number of downloads, ratings, and user reviews. The dataset may also incorporate additional features such as release date, developer information, and user demographics. We ensure data quality by handling missing values, removing duplicates, and addressing outliers.

2. Feature Extraction:

Feature extraction is crucial for extracting relevant information from the dataset to enhance predictive performance.

App Metadata Features: These include categorical variables such as app category, and numerical variables such as size, number of downloads, and release date. We may derive additional features, such as app age (difference between release date and current date), to capture temporal dynamics.

User Review Features: We extract sentiment analysis features from user reviews using techniques such as natural language processing (NLP). This involves tokenization, stemming, and sentiment polarity calculation to quantify the sentiment expressed in user reviews.

Additional Contextual Features: We incorporate additional contextual features such as developer reputation, user demographics (if available), and external factors that may influence app ratings.

3. Model Selection:

We choose K-Nearest Neighbors (KNN) and Random Forest Regression as our primary models for rating prediction due to their suitability for regression tasks and their ability to capture complex relationships in the data.

K-Nearest Neighbors (KNN): KNN is a non-parametric algorithm that makes predictions based on the similarity of instances in the feature space. Given a new data point, KNN identifies the K nearest neighbors in the training data and predicts the target variable (rating) based on the average or majority vote of these neighbors.

Random Forest Regression: Random Forest is an ensemble learning technique that constructs multiple decision trees and combines their predictions to produce a robust regression model. Random Forest Regression builds a forest of decision trees and averages their predictions to estimate the target variable (rating).

4. Model Training and Evaluation:

We split the preprocessed dataset into training and testing sets (e.g., 80% training, 20% testing) to train and evaluate the performance of the KNN and Random Forest Regression models. During the training phase, the models learn the underlying

patterns in the training data. We evaluate the models' performance on the testing data using appropriate regression evaluation metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2).

III. SIMULATION RESULTS

Python spyder version 3.7 is used to carry out the computation that has been suggested using Python. We are able to make use of the capabilities that are available in spyder climate for a variety of techniques with the assistance of the sklearn, numpy, pandas, matplotlib, pyplot, seaborn, and os library.

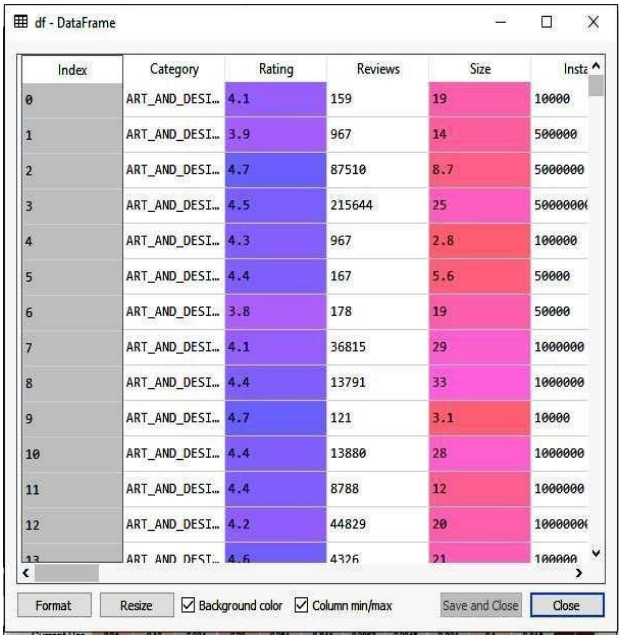

Figure 2: Dataset frame

A representation of the dataset in the Python environment may be seen in Figure 2. Several different numbers of rows and columns are included in the dataset. Additionally, the name of the signal characteristics is stated.

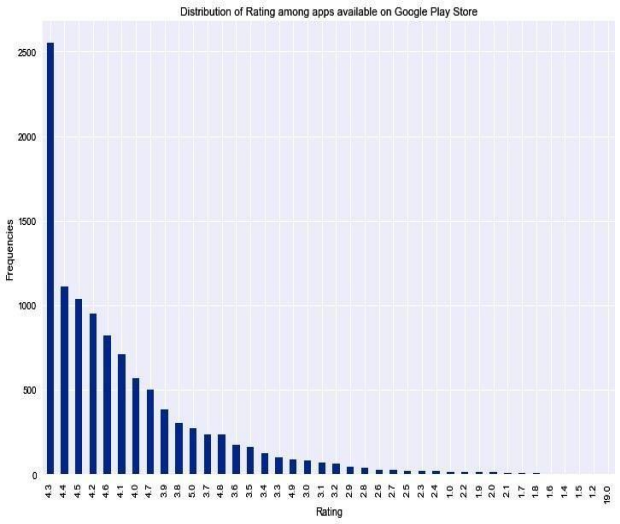

Figure 3: Distribution of rating

An example of the distribution of the Google Play Store rating forecast is shown in the figure. The lowest possible rating comes in at 1.0, while the highest possible rating is 4.9. Although the highest and lowest possible ratings for an app are lower, the average rating for a decent app falls somewhere between 3.9 and 4.5.

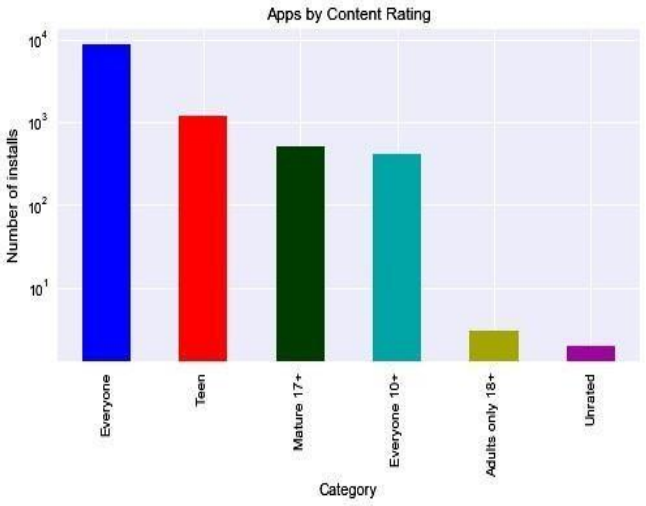

Figure 4: Apps by content rating

The different applications are shown in Figure 4 according to their content ratings. The most popular apps are those that are used by everyone, while most of the other apps that are downloaded are less popular, such as apps for teenagers, older users, unrated apps, and so on.

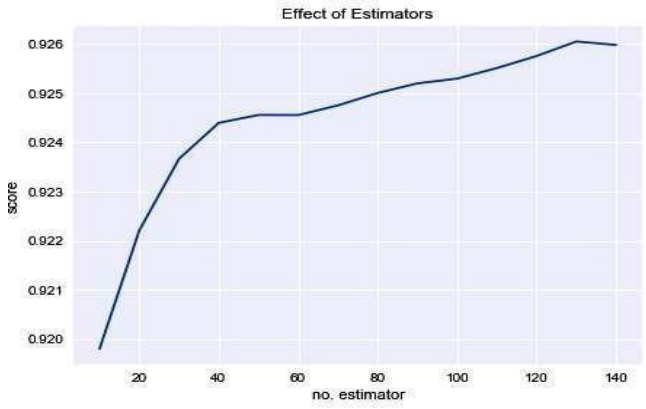

Figure 5: Effect of estimators

The influence of the estimators is shown in Figure 5, and the total number of estimators may reach up to 140. In terms of numerous checks, the overall score is about 95%.

Table 1: Result Comparison

Sr. No.	Parameters	Previous Work [1]	Proposed Work
1	Accuracy	93.8%	95.41%
2	Error rate	6.2 %	4.59%

IV. CONCLUSION

This study proposes an effective machine learning strategy for predicting ratings for applications that are available on the Google Play Store. The growing number of Android applications that are accessible on the Google Play Store, together with the benefits granted to developers, has captured the interest of a great number of Android application developers. To reap the benefits of designing Android applications, it is necessary to be familiar with the qualities that distinguish highly rated applications on the Google Play Store. 95.41% is the total accuracy that is reached by the approach that has been presented, while the prior technique achieved 93.8% of the achievable accuracy. In the work that is being presented, the error rate is 4.59%, whereas in the work that was done before, it was 6.2%. As a consequence, the efficient approach that was presented archived better outcomes than the one that was previously used.

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.