International Journal of Advancement in Electronics and Computer Engineering (IJAECE)
Volume 14, Issue 1, January. 2025, pp. 101-105 ISSN 2278 -1412
Copyright © 2012: IJAECE (www.ijaece.com)

Google Play Store App Rating Prediction using Artificial Intelligence Technique

Manisha Panthi[1], Firdous Qureshi[2], Nitya Khare[3], Swati Khanve[4]
[1] Research Scholar, manishapanthi1023@gmail.com, CSE SIRTE, Bhopal, India
[2,4]Assis. Prof., swatikhanve55.sk@gmail.com, CSE SIRTE, Bhopal, India
[3]HOD, CSE SIRTE, Bhopal, India

Abstract *– With millions of mobile applications available on the Google Play Store, users face challenges in identifying the best apps suited to their needs. App ratings significantly influence the decision-making process, with higher ratings often leading to greater visibility and increased downloads. This research focuses on predicting app ratings on the Google Play Store using artificial intelligence (AI) techniques, specifically the K- Nearest Neighbors (KNN) and Random Forest algorithms. By leveraging app metadata, user reviews, and other relevant features, the study develops a model capable of forecasting app ratings with high accuracy. The KNN algorithm, known for its simplicity and effectiveness in classification problems, is utilized alongside Random Forest, a powerful ensemble learning method that excels in handling complex datasets and predicting continuous values. The proposed approach is evaluated on a dataset containing various features of Google Play Store apps, and the results demonstrate the potential of AI in predicting app ratings. This study provides valuable insights for both developers and users by improving app discoverability and rating prediction accuracy.*

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

I. INTRODUCTION

In today's digital landscape, mobile applications have become an integral part of our daily lives, with millions of apps available on platforms like the Google Play Store. As users search for apps that meet their needs, they often rely on the ratings and reviews left by other users to make informed decisions. These ratings can serve as a reflection of an app's quality, usability, and overall performance. Consequently, understanding the factors that influence app ratings and predicting future ratings can be incredibly valuable for both developers and users.

The challenge, however, lies in the sheer volume of apps available, making it difficult for users to sift through and identify the best ones. App ratings are subjective, and their predictive accuracy depends on multiple variables, including the app’s category, description, update history, and most importantly, user feedback. The feedback provided by users in the form of reviews is often lengthy and contains various sentiments, making it a complex task to extract useful insights automatically.

This research aims to tackle the problem of app rating prediction by employing artificial intelligence (AI) techniques, specifically focusing on machine learning models and natural language processing (NLP). By analyzing a combination of app metadata and user reviews, the proposed AI system will be trained to predict the ratings of apps with higher accuracy. These predictions can help users discover quality apps and assist developers in enhancing their apps based on potential user ratings.

Recent advancements in AI, particularly in machine learning and NLP, have made it feasible to predict user ratings by processing vast amounts of data from the Play Store. By extracting meaningful patterns from user reviews and metadata, these AI models can evaluate how different factors—such as user sentiment, app updates, and download numbers—contribute to the overall rating. This predictive capability opens up new avenues for personalized app discovery and performance analysis, making the Play Store ecosystem more user-friendly.

Figure 1: Mobile App

With enormous challenge from everywhere throughout the globe, it is basic for a designer to realize that he is continuing in the right heading. To hold this income and their place in the market the application designers may need to figure out how to stick into their present position. The Google Play Store is observed to be the biggest application platform. It has been seen that in spite of the fact that it creates more than two fold the downloads than the Apple App Store yet makes just a large portion of the cash contrasted with the App Store. In this way, I scratched information from the Play Store to direct our examination on it.

With the fast development of advanced cells, portable applications (MobileApps) have turned out to be basic pieces of our lives. Be that as it may, it is troublesome for us to follow along the fact and to understand everything about the apps as new applications are entering market each day. It is accounted for that Android1market achieved a large portion of a million applications in September 2011. Starting at now, 0.675 million Android applications are accessible on Google Play App Store. Such a lot of applications are by all accounts an extraordinary open door for clients to purchase from a wide determination extend. We trust versatile application clients consider online application surveys as a noteworthy impact for paid applications. It is trying for a potential client to peruse all the literary remarks and rating to settle on a choice. Additionally, application engineers experience issues in discovering how to improve the application execution dependent on generally speaking evaluations alone and would profit by understanding a huge number of printed remarks.

II. BACKGROUND

Data R. Gomes et al., work aims to create inference engines, allowing the prediction of application ratings, using the KNN and Random Forest regression techniques. The Random Forest showed better results than the KNN [1]. C. Zhu et al., present the offline experiments on three large-scale datasets validate the superior performance of AIM. A three-week online A/B test in the recommendation service of a mainstream app market shows that AIM improves DeepFM model by 4.4% in terms of CTR [2].

G. S. Bhat et al., find a correlation between the particulate matter (PM) found indoors and the outside weather with the PEFR. The PEFR results are classified into three categories such as ‘Green’ (Safe), ‘Yellow’ (Moderate Risk) and ‘Red’ (High Risk) conditions in comparison to the best peak flow value obtained by each individual. Convolutional neural network (CNN) architecture is used to map the relationship between the indoor PM and weather data to the PEFR values [3]. Z. Wu et al., discovered three factors causing the inconsistency between descriptions and permission usages to be: 1) human interventions in writing description; 2) bad practices on permission usages; and 3) prolific developers. These findings will facilitate developers to refine app descriptions and optimize permission usages in the apps [4].

Z. Shen et al., work aims to predict a set of apps a user will open on her mobile device in the next time slot. Such information is essential for many smart phone operations, e.g., app pre-loading and content pre-caching, to improve user experience [5]. Z. Xu et al., propose a new cross-triplet deep feature embedding method, called CDFE, for cross-app JIT bug prediction task. The CDFE method incorporates a state-of-the-art cross-triplet loss function into a deep neural network to learn high-level feature representation for the cross- app data [6].

K. Zhao et al., conduct experiments on 10 Android mobile apps and experimental results show that SDF performs significantly better than comparative methods in terms of 3 performance indicators [7]. G. Aceto et al.,[8] compare the results with several ML approaches, showing performance comparable to a state-of-the-art ML predictor (Random Forest Regressor). Also, with this work we provide a viable and theoretically sound traffic-analysis toolset to help improving ML evaluation (and possibly its design), and a sensible and interpretable baseline [8].

Y. Zhang et al., develops multi-level modules based on Recurrent Neural Network with attention mechanism and generates multi-step time series predictions by fusing the outputs of modules. Experiments on a real- world dataset show that DeePOP outperforms state-of- the-art methods in prediction accuracy, effectively reducing the Root Mean Square Error (RMSE) to 0.088 [9].

Q. Zhu et al., present a fairness-aware APP recommendation method named FARM. The principal study of this method emphasizes on the fairness issue during the recommendation process. In this method, APP candidates are divided into high visibility and low visibility APPs, and implement recommendation algorithm respectively [10].

S. şahın et al., a novel approach to decision feedback reliability estimation through online prediction is proposed and applied for SISO FIR DFE with either a posteriori probability (APP) or expectation propagation (EP) based soft feedback. This novel method for filter computation is shown to improve detection performance compared to previously known alternative methods [11]. S. Rezaei et al., this work, propose a deep learning model for mobile app identification that works even with encrypted traffic. The proposed model only needs the payload of the first few packets for classification, and, hence, it is suitable even for applications that rely on early prediction, such as routing and QoS provisioning [12].

III. IMPLEMENTATION

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

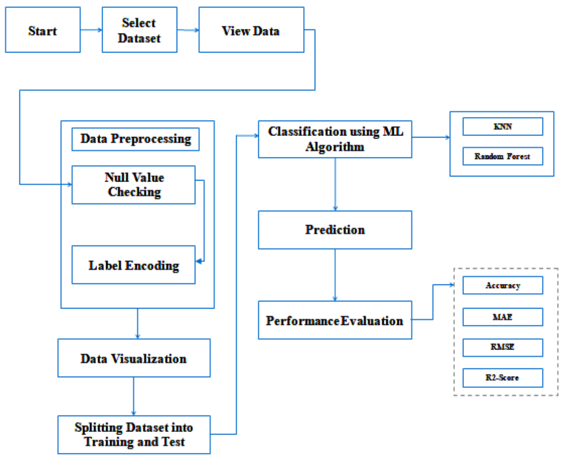


Figure 2: Flow Chart

The proposed model is introduced to overcome all the disadvantages that arise in the existing system. This system will increase the accuracy of the classification results by classifying the data based on the googleplay apps. The dataset collected from the Google Play store is semi structured or unstructured and contains significant superfluous data (defined as not contributing significantly to the prediction process). Training a supervised machine learning algorithm requires textual documents to be represented in vectorial form. For this purpose, textual data must be converted into numbers without losing information. Apply Machine Learning algorithms to predict app's rating with Regressors (KNN and Random Forest). Finally, the Performance of classifiers is evaluated on the accuracy, MAE, RMSE and R^2-Score.

This step involves preparing the dataset for machine learning by addressing data quality issues.

a. Null Value Checking

- Check for missing or null values in the dataset.
- Handle missing data using appropriate techniques such as:
- Imputation (e.g., filling missing ratings with the mean or median).
- Dropping rows or columns if the missing data percentage is too high.

b. Label Encoding

• Convert categorical variables (e.g., app category, content rating) into numerical values to make them usable for machine learning algorithms.

• Use Label Encoding or One-Hot Encoding depending on the algorithm's requirements.

c. Data Visualization

• Create visualizations to explore the dataset and understand relationships between features.

Examples:

▪ Scatter plots for reviews vs. ratings.

▪ Bar charts for app categories and their average ratings.

▪ Histograms to analyze the distribution of ratings.

d. Splitting Dataset into Training and Testing

Similar apps (nearest neighbors) based on selected features.

• Use hyperparameter tuning to optimize the value of 'k'.

Random Forest

• Train a Random Forest model that uses an ensemble of decision trees to predict app ratings.

• Adjust hyperparameters such as the number of trees and tree depth for better performance.

Prediction

• Use the trained models to predict the app ratings on the test dataset.

• Compare the predicted ratings with the actual ratings to assess the model's accuracy.

Performance Evaluation

Evaluate the models using the following metrics:

a. Accuracy- Measures the overall correctness of the predictions.

• Divide the dataset into two parts:

• Training Set: Used to train the machine learning models (e.g., 80% of the data).

• Testing Set: Used to evaluate the model's performance (e.g., 20% of the data).

Classification using ML Algorithm

The core step where machine learning models are applied.

KNN (K-Nearest Neighbors)

End

Mean Absolute Error (MAE)- Calculates the average absolute difference between predicted and actual ratings.

Root Mean Squared Error (RMSE)- Computes the square root of the average squared differences between predicted and actual ratings.

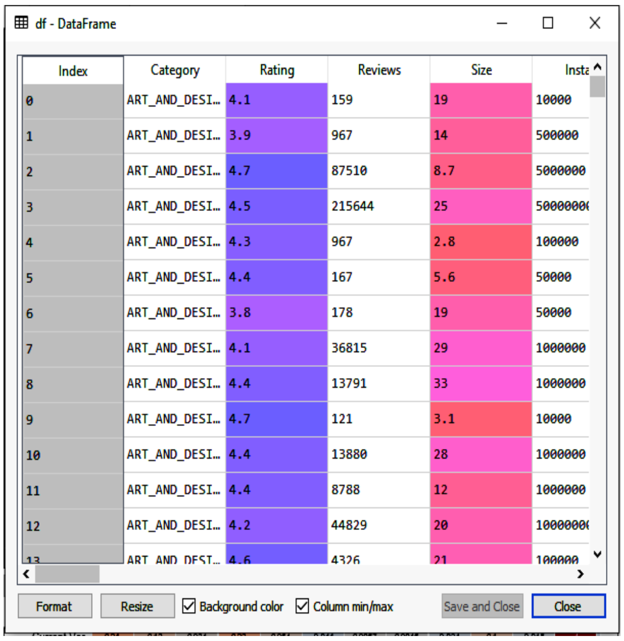
R-Squared (R^2) Score- Indicates how well the model explains the variance in the target variable (rating).

Build a KNN model to predict app ratings by identifying the average ratings of the 'k' most

Conclude the process after selecting the best-performing model (KNN or Random Forest) based on the evaluation metrics.

IV. SIMULATION RESULT

The execution of the proposed calculation is done over python spyder 3.7. The sklearn, numpy, pandas, matplotlib, pyplot, seaborn, os library assists us with utilizing the capacities accessible in spyder climate for different strategies.



Index	Category	Rating	Reviews	Size	Insta
0	ART_AND_DESI…	4.1	159	19	10000
1	ART_AND_DESI…	3.9	967	14	500000
2	ART_AND_DESI…	4.7	87510	8.7	5000000
3	ART_AND_DESI…	4.5	215644	25	50000000
4	ART_AND_DESI…	4.3	967	2.8	100000
5	ART_AND_DESI…	4.4	167	5.6	50000
6	ART_AND_DESI…	3.8	178	19	50000
7	ART_AND_DESI…	4.1	36815	29	1000000
8	ART_AND_DESI…	4.4	13791	33	1000000
9	ART_AND_DESI…	4.7	121	3.1	10000
10	ART_AND_DESI…	4.4	13880	28	1000000
11	ART_AND_DESI…	4.4	8788	12	1000000
12	ART_AND_DESI…	4.2	44829	20	10000000
13	ART_AND_DESI…	4.6	4326	21	100000

Figure 3: Dataset frame

Figure 3 is showing the dataset in the python environment. The dataset have various numbers of rows and column. The signal features name is also mentioned.

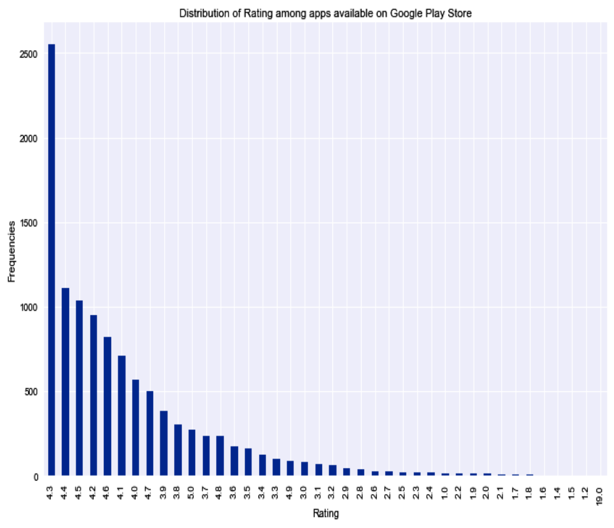


Figure 4: Distribution of rating

Figure 4 presents the distribution of the google play store rating prediction. There are maximum ratings is 4.9 and minimum rating is 1.0. The maximum and minimum rating app is less, the average rating of good app lie between 3.9 to 4.5.

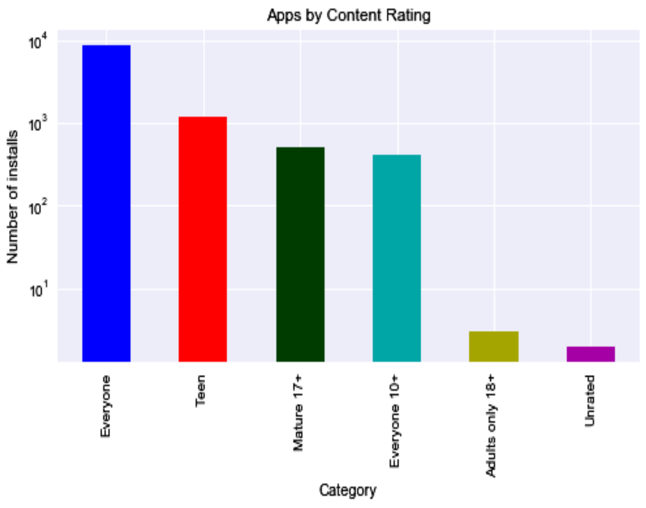


Figure 5: Apps by content rating

Figure 5 is presenting the various apps by the content rating, There are most installed apps which is used by everyone, other app downloads is less like apps for teen, mature, unrated etc.

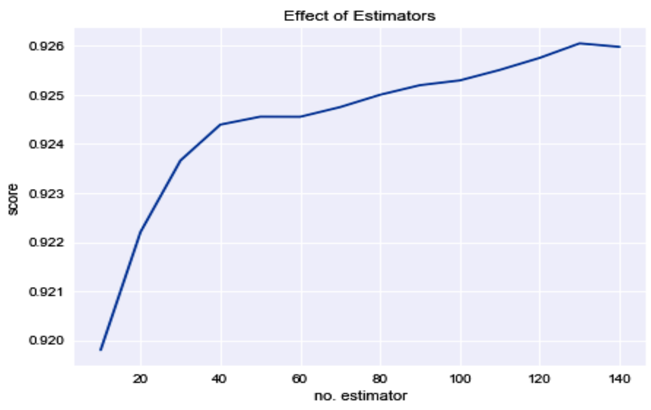


Figure 6: Effect of estimators

Figure 6 presents the effect of the estimators; the total number of the estimators is upto 140. The total score is aprox 95% for multiple check.

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%

V. CONCLUSION

This paper presents efficient machine learning technique for rating prediction of google play store apps. The increasing number of Android apps available on Google Play Store with the developers' advantages has attracted many Android apps developers' attention. To benefit from developing Android apps is to know the characteristics of high rated applications on the Google Play Store. The overall accuracy is achieved by the proposed technique is 95.41% while previous it is achieved by the 93.8%. The error rate is 4.59% in the proposed work and the 6.2% by the previous work. Therefore the proposed efficient technique archived better results than the previous..

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] Cheng Zhu, Yuanhong Zhao, Jiaqi Li, Xiaoyong Du, and Liang Zhao, "AIM: Automatic Interaction Machine for Click-Through Rate Prediction," IEEE Transactions on Knowledge and Data Engineering, doi: 10.1109/TKDE.2021.3134985.

[3] Gurudatt S. Bhat, Kalyan K. Srinivasa, Poonam S. Sharma, and Saumyajit Saha, "Machine Learning-Based Asthma Risk Prediction Using IoT and Smartphone Applications," IEEE Access, vol. 9, pp. 118708-118715, 2021, doi: 10.1109/ACCESS.2021.3103897.

[4] Zhan Wu, Xiaofeng Chen, Muhammad Usama Khan, and Sung Uk Jung Lee, "Enhancing Fidelity of Description in Android Apps With Category-Based Common Permissions," IEEE Access, vol. 9, pp. 105493-105505, 2021, doi: 10.1109/ACCESS.2021.3100118.

[5] Zheng Shen, Ke Yang, Zheng Xi, Jun Zou, and Wei Du, "DeepAPP: A Deep Reinforcement Learning Framework for Mobile Application Usage Prediction," IEEE Transactions on Mobile Computing, doi: 10.1109/TMC.2021.3093619.

[6] Zhen Xu, Xiaolong Yuan, Chao Zhang, Wenjie He, and Hongyu Zhang, "Effort-Aware Just-in-Time Bug Prediction for Mobile Apps Via Cross-Triplet Deep Feature Embedding," IEEE Transactions on Reliability, doi: 10.1109/TR.2021.3066170.

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

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

[10]Yue Zhang, Jian Liu, Bing Guo, Zhiqiang Wang, Yusheng Liang, and Zhiwen Yu, "App Popularity Prediction by Incorporating Time-Varying Hierarchical Interactions," IEEE Transactions on Mobile Computing, doi: 10.1109/TMC.2020.3029718.

[11]Qinghua Zhu, Qing Sun, Zhen Li, and Shuiguang Wang, "FARM: A Fairness-Aware Recommendation Method for High Visibility and Low Visibility Mobile APPs," IEEE Access, vol. 8, pp. 122747-122756, 2020, doi: 10.1109/ACCESS.2020.3007617.

[12]Sait Șahin, Alessandro M. Cipriano, Christophe Poulliat, and Michel Boucheret, "Iterative Decision Feedback Equalization Using Online Prediction," IEEE Access, vol. 8, pp. 23638-23649, 2020, doi: 10.1109/ACCESS.2020.2970340.

[13]Shayan Rezaei, Bernhard Kroencke, and Xiaoming Liu, "Large-Scale Mobile App Identification Using Deep Learning," IEEE Access, vol. 8, pp. 348-362, 2020, doi: 10.1109/ACCESS.2019.2962018.

[14]Shichang Zhao, Lili Liu, Fenglong Ma, Jing Gao, and Aidong Zhang, "Gender Profiling From a Single Snapshot of Apps Installed on a Smartphone: An Empirical Study," IEEE Transactions on Industrial Informatics, vol. 16, no. 2, pp. 1330-1342, Feb. 2020, doi: 10.1109/TII.2019.2938248.

[15]Chunyang Min, Anwar Haque, Xiaojing Du, David Chu, and Xia Zhou, "Scalable Power Impact Prediction of Mobile Sensing Applications at Pre-Installation Time," IEEE Transactions on Mobile Computing, vol. 19, no. 6, pp. 1448-1464, June 2020, doi: 10.1109/TMC.2019.2909897.

[16]Google Play Store Apps Dataset, Kaggle. Available: https://www.kaggle.com/datasets/lava18/google-play-store-apps.