An Enhanced Random Forest (ERF)-based Machine Learning Framework for Resampling, Prediction, and Classification of Mobile Applications using Textual Features

Shahbaz Hussain
Department of Computer Science, Faculty of Computer Science & IT, Superior University, Lahore, Pakistan
su92-mscsw-f22-011@superior.edu.pk

Nadeem Sarwar
Department of Computer Science, Bahria University, Lahore Campus, Lahore, Pakistan
nadeem_srwr@yahoo.com

Arshad Ali
Faculty of Computer and Information Systems, Islamic University of Madinah, Al Madinah Al Munawarah, Saudi Arabia
a.ali@iu.edu.sa

Hamayun Khan
Department of Computer Science, Faculty of Computer Science & IT, Superior University, Lahore, Pakistan
hamayun.khan@superior.edu.pk

Irfanud Din
Department of Computer Science, New Uzbekistan University, Tashkent, Uzbekistan
irfan@newuu.uz

Abdullah M. Alqahtani
College of Engineering & Computer Science, Department of Electrical & Electronic Engineering, Jazan University, Saudi Arabia
amqahtani@jazanu.edu.sa

Mohamed Shabir
Network Security Forensic Group, School of Technology, Asia Pacifc University, Malaysia
mohamed.shabbir@apu.edu.my

Aitizaz Ali
School of Technology, Network Security Forensic Group, Asia Pacific University, Malaysia
aitizaz.ali@apu.edu.my (corresponding author)

Received: 1 October 2024 | Revised: 1 November 2024 and 14 November 2024 | Accepted: 5 December 2024

Licensed under a CC-BY 4.0 license | Copyright (c) by the authors | DOI: https://doi.org/10.48084/etasr.9148

ABSTRACT

The amount of mobile applications is increasing rapidly, and it is difficult for software developers to identify the numerous key factors that affect their rating and performance. This study presents a machine-learning framework to improve decisions in adding new features to mobile applications and enhancing overall performance. A dataset of app attributes from the Apple AppStore was used, exploiting NLP techniques to preprocess the textual information and develop an Enhanced Random Forest (ERF) framework to assess and forecast ratings for multifunctional apps and investigate the connections between features and user ratings. The ERF model was compared with other renowned ML methods including Decision Trees (DT), Naive Bayes (NB), CNN, and ANN. The experimental results showed that the proposed model predicts app ratings more effectively compared to other complex models. The proposed model achieved precision, recall, and F1-score of 92.76%, 99.33%, and 95.93%, respectively.

Keywords-machine learning; reliability; mobile applications; sustainable learning; predicting mobile app ratings; user ratings; XGBoost; random forest; NLP high-dimensional datasets; convolutional neural network (CNN); NSL-KDD; UNSW-NB15; mean square error

I. INTRODUCTION

The mobile application industry has been developed radically in recent years. Applications have changed communication, support, and engagement. Customer reviews demonstrate the quality and support of an application to enhance its performance and high ratings attract users [1]. This trend is followed by a growing number of mobile software companies that provide a massive number of mobile applications. Specifically, there are two giant platforms on the market, provided by Google Play Store (GPS) and Apple Store. Developers who wish to increase client engagement and app popularity must understand what drives those assessments. User ratings, determined by app estimate, category, and user input, are significant for app success [2-4]. The application rating represents all reviews received from users. However, since not all applications have excellent ratings, machine learning techniques can be used to evaluate app ratings. This study examines Machine Learning (ML)-based models, such as XGBoost, Random Forest (RF), and Logistic Regression (LR), to assess ratings and performance.

In [5], a GPS dataset was used to examine the general popularity of an application and use the number of installations as a measure of how diverse app features affect user evaluations. App estimate, pricing, category, and user reviews commonly appear to be significant determinants of overall assessment. Moreover, estimation procedures are vital, as clients appreciate applications more when they give them great value for money or utilize successful models [6]. Furthermore, an application's rating may be heavily influenced by its category [7]. Applications in specific categories, such as gaming and social networking, receive higher ratings than apps in other categories, such as business or utility. User expectations vary, and participation from different app categories primarily generates this variation.

XGBoost and RF are ensemble approaches that demonstrate remarkable prediction performance. In RF, several Decision Trees (DTs) are combined to increase prediction accuracy and resilience while reducing the likelihood of overfitting in XGBoost [8]. In [9], a predictive model was presented, using weak learners and gradient-boosting methods, achieving outstanding results. Other studies compared the accuracy of machine learning algorithms in evaluating app ratings, where XGBoost outperformed other approaches and conventional regression models in predicting app success measures, such as user ratings [10, 11]. In [12], collaborative content-based sifting approaches were used with machine learning algorithms to develop a combined technique to predict app ratings. The results showed that hybrid models can better forecast results utilizing explicit app features and verifiable user preferences. Recent advances in Natural Language Processing (NLP) have substantially improved the predictive capability of ML models. The study in [13] focused on the importance of opinion analysis in understanding user evaluations, which may be a critical pointer of app evaluation. Researchers can improve forecast accuracy and better understand client perspectives by utilizing everyday NLP strategies to examine textual data from user reviews. In addition, deep learning models are becoming progressively prevalent in this field. In [14], a personalized real-time suggestion framework for versatile apps, based on deep learning, yielded great results. These algorithms, distinguishing complex designs from expansive datasets, provide a reasonable avenue for advanced research on in-app rating prediction [15].

The interpretability of complicated models is an additional substantial hurdle. Although deep learning models and ensemble approaches provide great prediction accuracy, their decision-making procedures are frequently opaque. For applications to be practical, efforts must be made to develop more interpretable models or improve the transparency of existing ones. The methods and conclusions of app rating prediction research are widely used in various industries. For example, analyzing user reviews and forecasting ratings can also be utilized for other online assessments, including product reviews on e-commerce sites. Furthermore, product development and marketing tactics in other sectors could benefit from the knowledge gathered from studying customer preferences in the app industry.

Although conventional regression models are helpful, more sophisticated ensemble techniques, such as RF and XGBoost, frequently outperform them. Predictive accuracy may be increased in novel ways thanks to the combination of helpful deep learning models with NLP techniques. However, issues with the interpretability of models, data quality, and the dynamic nature of app markets continue to make this task challenging. Future studies should focus on resolving these issues and investigating the more extensive uses of these prediction techniques.

II. METHODOLOGY

A. Data Collection & Preprocessing

The study analyzes a dataset of 7,197 mobile applications from the Apple App Store [16] focusing on various features hypothesized to influence user ratings. The dataset incorporates the app name, size in bytes, cost, average client rating, content rating, class, number of supported devices, and a screenshot. These features are used to investigate the variables that influence how users rate and perceive mobile apps. Data preprocessing is fundamental to ensure the quality and utility of the dataset for ML models. This procedure involves deleting records with significant missing data or replacing missing values with appropriate replacements, such as the average for numerical characteristics. For ML algorithms to significantly utilize categorical variables such as app category and substance rating, one-hot encoding is used to change them into numerical values. Furthermore, numerical parameters, such as app estimate, estimating, and user rating counts are scaled using normalization techniques to ensure that each feature contributes similarly to the model's execution. The dataset was divided in an 80/20 ratio into training and test sets, allowing the model evaluation on unseen data.

B. Feature Selection

Feature engineering involves selecting and altering the most fundamental features. The app's size in bytes (size bytes), cost, total rating count (rating_count_tot), average user rating (user rating), app sort (prime genre), and number of supported devices (sup_devices.num) were chosen as the essential predictors for this study. These features were selected for their expected impact on client evaluations and their availability within the dataset.

C. Model Selection

This study used LR, RF, and XGBoost to predict the class of ratings for mobile apps. LR is a linear model used for classification. It employs the logistic function to calculate the probability of a categorical variable by anticipating which of two potential classes a perception may belong to. This model is simple to handle and gets each attribute's impact. Selecting these preprocessing methods and models aimed at creating an effective prediction system for app evaluations.

Figure 1 represents a typical ML pipeline for a classification task. It begins with the dataset, which contains the raw data to be utilized for modeling. The data are preprocessed, where it is cleaned and altered to ensure quality and consistency. Then, each of the three ML models is trained and evaluated exclusively, where measurements such as accuracy, precision, review, and F1 scores are calculated. These metrics are used to compare the models.

D. Model Evaluation

Each model was evaluated using various assessment measures in conjunction with cross-validation. The rate of all precise predictions the model makes is known as accuracy, and precision denotes the rate of accurate predictions by the total projected positives. The model's recall measures how well it can recognize each true positive. The F1 score is the harmonic mean of recall and precision, which is important when working with unbalanced datasets. The confusion matrix records the amounts of true positives, true negatives, false positives, and false negatives. This framework allows one to determine model advantages and impediments, providing a more advanced picture of its predictive capacity.

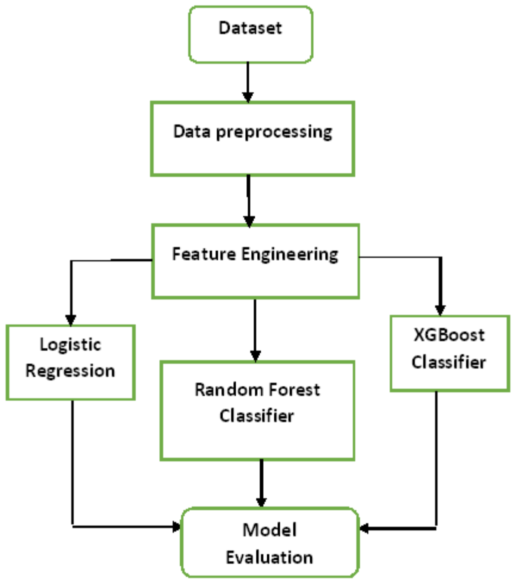

Fig. 1. Model selection process.

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \quad (1)$$

Accuracy is the proportion of correctly predicted perceptions (true positives and negatives) to all observations. It measures how often the classifier is correct.

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \quad (2)$$

Precision is the ratio of correctly predicted positive observations to the total predicted positives. It indicates how many of the predicted positives are truly positive.

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \quad (3)$$

Recall, also known as sensitivity or true positive rate, is the ratio of correctly predicted positive observations to all observations in the actual positive class and measures the classifier's ability to find all the positive samples.

$$\text{F1}-\text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

The ROC-AUC score (Area under the Receiver Operating Characteristic Curve) is a metric that summarizes the execution of a parallel classifier. The ROC curve plots the genuine positive rate against the false positive rate, and the AUC measures the complete two-dimensional range underneath the bend. Matthew's Correlation Coefficient (MCC) is used to evaluate the performance of RF and LR by considering various parameters used for precision, recall, and accuracy.

$$\text{ROC}-\text{AUC} = \int_0^1 \text{TPR}(\text{FPR})d(\text{FPR}) \quad (5)$$

$$\text{MCC} = \frac{(\text{TP}\times\text{TN})-(\text{FP}\times\text{FN})}{\sqrt{(\text{TP+FP})(\text{TP+FN})(\text{TN+FP})(\text{TN+FN})}} \quad (6)$$

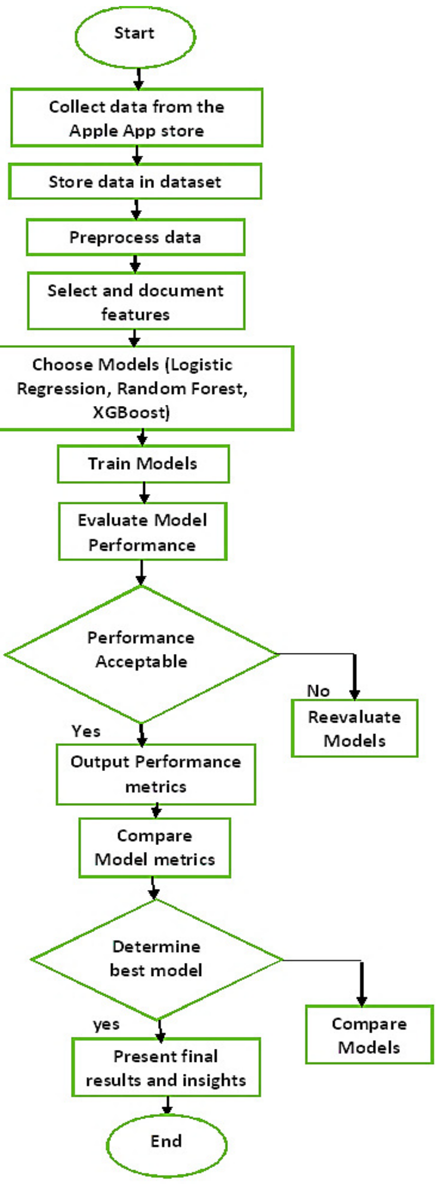
Start
Collect data from the Apple App store
Store data in dataset
Preprocess data
Select and document features
Choose Models (Logistic Regression, Random Forest, XGBoost)
Train Models
Evaluate Model Performance
Performance Acceptable
No
Reevaluate Models
Yes
Output Performance metrics
Compare Model metrics
Determine best model
yes
Present final results and insights
Compare Models
End

Fig. 2. Flowchart of the method.

Figure 2 outlines the process for predicting mobile app ratings using machine learning models. The process begins with collecting data from an Apple App Store dataset. The data is then preprocessed, followed by selecting and documenting relevant features. Machine learning models are chosen and trained, including LR, RF, and XGBoost. Their performance is evaluated, and if it is acceptable, the results are documented, and the models' metrics are compared to determine the best-performing model. The final results and insights are then presented. If the performance is unacceptable, the models are re-evaluated, and the process is repeated. The flowchart concludes with the end of the process after presenting the final results or re-comparing models if necessary.

```
Algorithm 1: Proposed XGBoost Classifier
for Predicting Mobile App Ratings
Input: Preprocessed data (app features)
Output: Predicted mobile app ratings
1:  // Data Collection
    app_data = read_csv('app_data.CSV)
2:  // Data Preprocessing
    Handle missing values: app_data =
    fill_missing_values(app_data)
    Encode categorical variables: app_data
    = one_hot_encode(app_data)
    Normalize numerical features: app_data
    = normalize_features(app_data)
3:  // Data Splitting
    train_data, test_data =
    train_test_split(app_data,
    test_size=0.2)
4:  // Model Initialization
    Initialize XGBoost Classifier with
    parameters:
    model = XGBClassifier(
    learning_rate=0.1, max_depth=6,
    n_estimators=100)
5:  // Model Training
    model.fit(train_data.features,
    train_data.labels)
6:  // Model Evaluation
    Predict on test data: predictions =
    model.predict(test_d.features)
    Evaluate performance:
    Calculate accuracy: accuracy =
    calculate_accuracy(t_d.labels,
    predictions)
    Generate confusion matrix:
    conf_matrix =
    confusion_matrix(test_data.labels,
    predictions)
    Compute F1 score: f1 =
    calculate_f1_score(test_data.labels,
    predictions)
7:  // Feature Importance Analysis
    feature_importance =
    model.feature_importances
8:  // Prediction
    predicted_ratings =
    model.predict(new_app_data.features)
```

III. RESULTS AND DISCUSSION

The LR model's performance metrics in testing indicate a moderate accuracy (0.72), indicating that 72% of the model's predictions were correct. With a precision of 0.68, the model was accurate 68% of the time when it predicted a positive app rating. The model correctly identified 70% of the actual positive ratings, as evidenced by the recall rate of 0.70. Its F1 score was 0.69, indicating a balance between the two metrics.

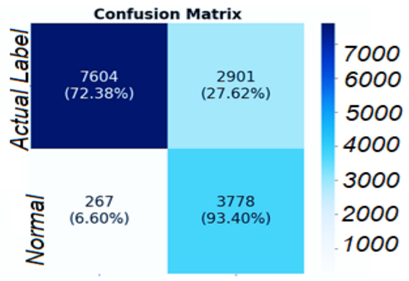

Fig. 3. Confusion matrix for LR.

Figure 3 shows the model's performance in predicting whether ratings are below or equal to/over 4. The color intensity indicates the number of observations in each quadrant, with darker shades representing higher counts. LR appears to have a high number of true positives and true negatives, suggesting good predictive performance.

The performance metrics of the RF classifier were significantly improved over simpler models. During training, its accuracy was 0.85, indicating that 85% of the predictions were correct. The training precision of 0.82 indicates that when the model predicts a positive rating, it was accurate 82% of the time. The recall of 0.84 shows that the model successfully identified 84% of the positive ratings. The F1 score of 0.83 reflects a good balance between precision and recall. These metrics demonstrate the effectiveness of the RF classifier in predicting app ratings, highlighting its robustness and higher accuracy compared to LR. The RF model had a high number of true positives and true negatives, suggesting that it is performing well in distinguishing between the two classes..

TABLE I. PERFORMANCE OF DIFFERENT MODELS

Metric	LR	RF	XGBoost
Accuracy	0.72	0.85	0.82
Precision	0.68	0.85	0.81
Recall	0.70	0.87	0.81
F1 Score	0.69	0.83	0.80

Figure 4 shows how the RF model's testing accuracy improves over time, starting from 0.6 and gradually increasing to 0.88 by the 10th epoch.

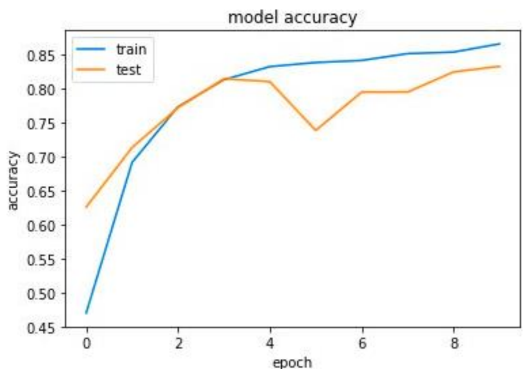

Fig. 4. RF accuracy over epochs.

Figures 5 and 6 indicate that the RF model is better at correctly identifying positive instances out of all its optimistic predictions compared to NODE, TABNET, and Gradient Boosting (GB) using SMOTE_ENN, ADASYN, and Tomek to address class imbalance. NODE, GB, and TABNET are renowned deep learning architectures that are used to compare the performance of machine learning models. Using both SFS and fused feature selection, the RF model achieved better F1 scores.

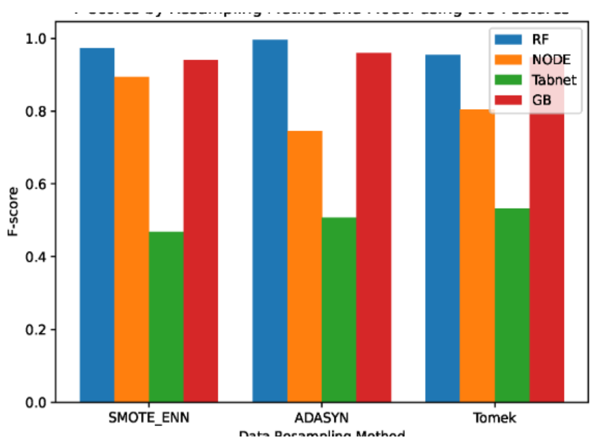

Fig. 5. F1-score comparison with feature selection.

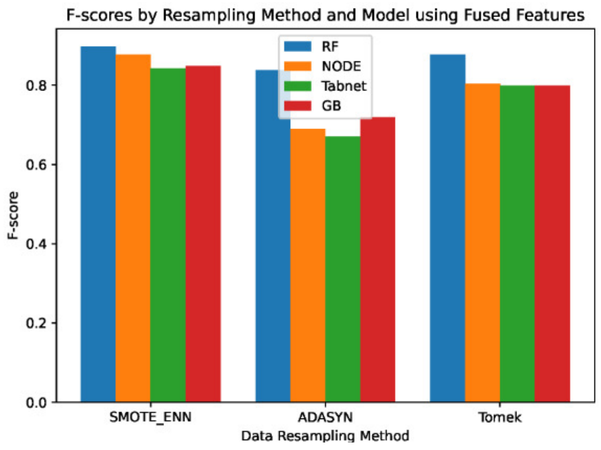

Fig. 6. F1-score comparison using fused feature selection.

IV. CONCLUSIONS AND FUTURE RECOMMENDATIONS

This study examined ML-based predictive analysis techniques to evaluate the authenticity, availability, performance, and accuracy of user reviews available on the Apple App Store. For this purpose, a dataset was used to evaluate mobile app ratings by applying and comparing LR, RF, and XGBoost. The results showed that ensemble methods, especially RF and XGBoost, provide essentially higher prediction accuracy compared to simpler models such as LR. The proposed RF technique emerged as the top performer, showing its robustness in accurately predicting app ratings based on client feedback and app attributes. In the future, a scenario where an application can use ML models to proactively identify issues and take steps to address them before negatively affecting user satisfaction can be employed by integrating NLP procedures to better capture user sentiment from surveys or comments, a key determinant of user evaluations. Extending this model to other app stages might further improve its adaptability and reliability. This study focused on the requirement for modern ML approaches in predictive analytics within the app industry, helping developers improve app quality and user satisfaction.

REFERENCES

[1] P. M. Dhulavvagol and S. G. Totad, "Performance Enhancement of Distributed Processing Systems Using Novel Hybrid Shard Selection Algorithm," *Engineering, Technology & Applied Science Research*, vol. 14, no. 2, pp. 13720–13725, Apr. 2024, https://doi.org/10.48084/etasr.7128.

[2] J. Song, J. Kim, D. R. Jones, J. Baker, and W. W. Chin, "Application discoverability and user satisfaction in mobile application stores: An environmental psychology perspective," *Decision Support Systems*, vol. 59, pp. 37–51, Mar. 2014, https://doi.org/10.1016/j.dss.2013.10.004.

[3] C. Z. Liu, Y. A. Au, and H. S. Choi, "Effects of Freemium Strategy in the Mobile App Market: An Empirical Study of Google Play," *Journal of Management Information Systems*, vol. 31, no. 3, pp. 326–354, Jul. 2014, https://doi.org/10.1080/07421222.2014.995564.

[4] A. Ghose and S. P. Han, "An Empirical Analysis of User Content Generation and Usage Behavior on the Mobile Internet," *Management Science*, vol. 57, no. 9, pp. 1671–1691, Sep. 2011, https://doi.org/10.1287/mnsc.1110.1350.

[5] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. John Wiley & Sons, 2013.

[6] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, https://doi.org/10.1023/A:1010933404324.

[7] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, Aug. 2016, pp. 785–794, https://doi.org/10.1145/2939672.2939785.

[8] V. Balakrishnan, Z. Shi, C. L. Law, R. Lim, L. L. Teh, and Y. Fan, "A deep learning approach in predicting products' sentiment ratings: a comparative analysis," *The Journal of Supercomputing*, vol. 78, no. 5, pp. 7206–7226, Apr. 2022, https://doi.org/10.1007/s11227-021-04169-6.

[9] M. R. Dehkordi, H. Seifzadeh, G. Beydoun, and M. H. Nadimi-Shahraki, "Success prediction of android applications in a novel repository using neural networks," *Complex & Intelligent Systems*, vol. 6, no. 3, pp. 573–590, Oct. 2020, https://doi.org/10.1007/s40747-020-00154-3.

[10] X. Wu and Y. Zhu, "A Hybrid Approach Based on Collaborative Filtering to Recommending Mobile Apps," in *2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)*, Wuhan, China, Dec. 2016, pp. 8–15, https://doi.org/10.1109/ICPADS.2016.0011.

[11] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1–2, pp. 1–135, Jul. 2008, https://doi.org/10.1561/1500000011.

[12] M. C. Chiu, J. H. Huang, S. Gupta, and G. Akman, "Developing a personalized recommendation system in a smart product service system based on unsupervised learning model," *Computers in Industry*, vol. 128, Jun. 2021, Art. no. 103421, https://doi.org/10.1016/j.compind.2021.103421.

[13] N. Jindal and B. Liu, "Opinion spam and analysis," in *Proceedings of the international conference on Web search and web data mining - WSDM '08*, Palo Alto, CA, USA, 2008, pp. 219-230, https://doi.org/10.1145/1341531.1341560.

[14] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, Aug. 2016, pp. 1135–1144, https://doi.org/10.1145/2939672.2939778.

[15] S. M. Mudambi and D. Schuff, "Research Note: What Makes a Helpful Online Review? A Study of Customer Reviews on Amazon.com," *MIS Quarterly*, vol. 34, no. 1, pp. 185–200, 2010, https://doi.org/10.2307/20721420.

[16] G. Prakash, "Apple AppStore Apps." [Online]. Available: https://www.kaggle.com/datasets/gauthamp10/apple-appstore-apps.