App Rating Prediction

ON

Submitted in partial fulfillment of the requirements
of the degree of

Bachelor of Engineering
(Information Technology)

By

Shiven Bansal-Roll No (03)

Vedant Dhoke-Roll No (09)

Ishan Kiran Joshi-Roll No (21)

Under the guidance of

Dr. Ravita Mishra

Department of Information Technology

VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY, Chembur, Mumbai 400074
(An Autonomous Institute, Affiliated to University of Mumbai)

April 2024

Vivekanand Education Society's
Institute of Technology

(Autonomous Institute Affiliated to University of Mumbai, Approved by AICTE & Recognised by Govt. of Maharashtra)
NAAC accredited with 'A' grade

Certificate

This is to certify that project entitled

"**Flight Delay Probability Checker**"

Group Members Names

Shiven Bansal-Roll No (03)

Vedant Dhoke-Roll No (09)

Ishan Kiran Joshi-Roll No (21)

In fulfillment of degree of BE. (Sem VI) in Information Technology for Project is approved

Prof. Dr. Ravita Mishra
Project Mentor

External Examiner

Dr.(Mrs.) Shalu Chopra
H.O.D

Dr.(Mrs.) J.M.Nair
Principal

Date: / /

College Seal

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission.We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Abstract

This project aims to develop a machine learning model that predicts app ratings based on key features such as app category, reviews, installs, price, and size. By analyzing historical data from the Google Play Store, we apply regression techniques to build an accurate predictive system. The model's primary objective is to provide valuable insights to developers, helping them optimize app design and features to enhance user satisfaction. Feature engineering and hyperparameter tuning are used to improve model performance. Additionally, the project focuses on ensuring model interpretability, identifying the most influential factors affecting app ratings. The results will support data-driven decisions in app development and marketing. This tool will ultimately empower developers to make informed improvements and drive app success.

Contents

List of Figures

ACKNOWLEDGEMENT

The project report on "**App Rating Prediction**" is the outcome of the guidance, moral support and devotion bestowed on our group throughout our work. For this we acknowledge and express our profound sense of gratitude to everybody who has been the source of inspiration throughout project preparation. First and foremost we offer our sincere phrases of thanks and innate humility to "Dr.(Mrs). Shalu Chopra HOD INFT", " Dr. Manoj Sabnis Deputy HOD INFT", "Dr.Ravita Mishra Assistant Professor" for providing the valuable inputs and the consistent guidance and support provided by them.We can say in words that we must at outset tender our intimacy for receipt of affectionate care to Vivekanand Education Society's Institute of Technology for providing such a stimulating atmosphere and conducive work environment.

Chapter 1

Introduction

1.1 Introduction

In today's digital age, mobile applications play a crucial role in various aspects of life, such as communication, entertainment, productivity, and more. With over 3 million apps on the Google Play Store, users often face difficulties in selecting the most suitable and high-quality apps. App ratings are instrumental in helping users make informed download decisions, with these ratings influenced by factors such as user interface, performance, features, and reliability. However, predicting an app's potential rating based on its characteristics remains a challenge for developers. Factors like the app's category, size, number of installs, and user feedback significantly impact its rating. This project aims to address this challenge by leveraging machine learning to create a predictive model that forecasts app ratings using historical data from the Google Play Store. The objective is to provide developers and businesses with data-driven insights, which can enhance design decisions, improve app performance, and ultimately lead to higher user satisfaction.

1.2 Objectives

To train a regression or classification model on the Play Store dataset, machine learning techniques are employed to predict app ratings. Key features such as app category, reviews, installs, price, and size are analyzed to understand their impact on ratings. Feature engineering is applied to extract meaningful information from raw data, while hyperparameter tuning ensures the model performs optimally. Ensemble learning techniques, such as Random Forest or Gradient Boosting, are used to improve prediction accuracy by combining multiple models. Additionally, model interpretability is a priority, as identifying the most influential factors affecting app ratings helps developers make informed decisions. By refining the model through iterative improvements and data analysis, developers can predict app ratings with greater accuracy and actionable insights.

1.3 Motivation

The motivation behind this project stems from the overwhelming number of mobile applications available in the market, with millions of apps competing for users' attention.

With so many options, users often rely on app ratings to make decisions, but these ratings can be influenced by numerous factors that are not immediately apparent. For developers, predicting an app's rating based on its characteristics such as category, reviews, installs, and size remains a complex challenge.This project is motivated by the need to create a machine learning model that can accurately predict app ratings, offering valuable insights into app performance. By leveraging historical data from the Google Play Store, the goal is to equip developers with a data-driven tool to improve app design, optimize user experience, and ultimately increase app success. Furthermore, understanding the key features that impact app ratings will not only help developers improve their offerings but also enhance the overall user satisfaction and trust in the app marketplace.

1.4 Scope of the Work

The scope of this project involves building a machine learning model to predict app ratings based on various app characteristics using data from the Google Play Store. It covers analyzing key features such as app category, reviews, installs, price, and size, to identify their impact on ratings. The project also includes optimizing model performance through techniques like feature engineering, hyperparameter tuning, and ensemble learning. Additionally, the scope extends to ensuring model interpretability by highlighting the most influential factors affecting app ratings. By providing developers with data-driven insights, the project aims to enhance app design and user experience. The outcome of this project will support decision-making for developers, businesses, and marketers seeking to improve app quality. Finally, the project's findings can be applied to the broader mobile app market, providing actionable recommendations to enhance app success.

1.5. Feasibility Study

The feasibility of this project is high, given the availability of comprehensive datasets from the Google Play Store, which provide essential features like app category, reviews, installs, price, and size. Machine learning techniques are well-established for regression and classification tasks, making it technically feasible to predict app ratings accurately. Tools like Python, with libraries such as scikit-learn, pandas, and TensorFlow, offer powerful frameworks for model building and optimization. The project also benefits from clear objectives, ensuring focused development efforts. While challenges may arise in handling missing data or outliers, these can be addressed through data preprocessing and feature engineering techniques. The project is feasible within a reasonable timeframe and can provide valuable insights for developers. Furthermore, it can be scaled or adapted to various other app marketplaces beyond the Play Store, enhancing its long-term applicability.

Chapter 2

Literature Survey

2.1 Introduction

The literature survey focuses on studies that analyze factors influencing app ratings, such as user reviews, app category, size, and price. Research has shown that machine learning techniques, including regression and classification models, can effectively predict app ratings using these features. Various models, such as decision trees, random forests, and support vector machines, have been employed for this purpose. Additionally, studies on feature engineering and hyperparameter tuning highlight methods for optimizing model performance. Challenges like handling missing data and ensuring model interpretability have also been discussed in the literature. The survey aims to provide a foundation for building a predictive model that can offer actionable insights for app developers. This review also identifies gaps in existing research, guiding the approach for this project.

2.2 Problem Definition

With millions of apps on platforms like the Google Play Store, users rely heavily on app ratings for decision-making. However, predicting an app's rating based on its features is challenging due to the many factors influencing user satisfaction. Developers often lack insights into how attributes such as app category, reviews, installs, and content rating impact ratings. This project aims to create a predictive model that estimates app ratings using these key attributes. By analyzing historical data and applying machine learning regression techniques, we strive to develop an accurate forecasting system. The model will assist developers in optimizing app performance and enhancing user satisfaction. Ultimately, it will guide better decision-making in app development and marketing.

2.3. Review of Literature Survey

2.3.1 App Review Prediction Using Machine Learning

Authors: S. Shiva Prakash, Dr. C. Siva Kumar
Published In: Journal of Pharmaceutical Negative Results, 2022

This study explored the classification and analysis of app reviews using machine learning to understand user sentiment. Using models like Logistic Regression,

Decision Trees, K-Nearest Neighbors, and deep learning (RNN, LSTM), the paper demonstrated that RNN and LSTM achieved high accuracy in classifying reviews as positive, neutral, or negative.

Relevance: Although it focused on textual sentiment, this paper offered valuable methods for preprocessing, model comparison, and validating predictions—all of which are crucial to our rating-based regression model.

2.3.2 Play Store App Analysis & Rating Prediction Using ML and ANN

Authors: Bhimasen Moharana, Bhramara Bar Biswal, Snehasis Dey, Manas Kumar Rath, Shobhan Banerjee
Published In: Proceedings of the 2023 7th International Conference on Computing, Communication, Control, and Automation (ICCUBEA)

This paper focused directly on rating prediction, comparing classical ML models with Artificial Neural Networks (ANN). Features such as installs, version, category, and price were used to train models. The ANN model demonstrated better performance in handling nonlinear relationships between features and ratings.

Relevance: This work forms a solid reference point for our project. It validates our approach and provides benchmarks for evaluating classical versus advanced models.

Chapter 3
Design and Implementation

3.1. Introduction

The design and implementation of the Google Play Store App Rating Prediction System involve building a machine learning model and integrating it with an interactive web interface using Streamlit. The project focuses on predicting app ratings based on features like App Size, Number of Installs, Reviews, Type, Price, Category, and Content Rating. The dataset underwent thorough preprocessing, including handling missing values, encoding categorical variables, and scaling numerical data. Various regression algorithms such as Support Vector Regression (SVR), Decision Tree Regressor, Random Forest Regressor, and Ridge Regression were implemented and evaluated using Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) to determine the most accurate model. The final model was deployed using Streamlit, allowing users to input or search app details and instantly view the predicted ratings in a user-friendly interface.

3.2 UML Diagram

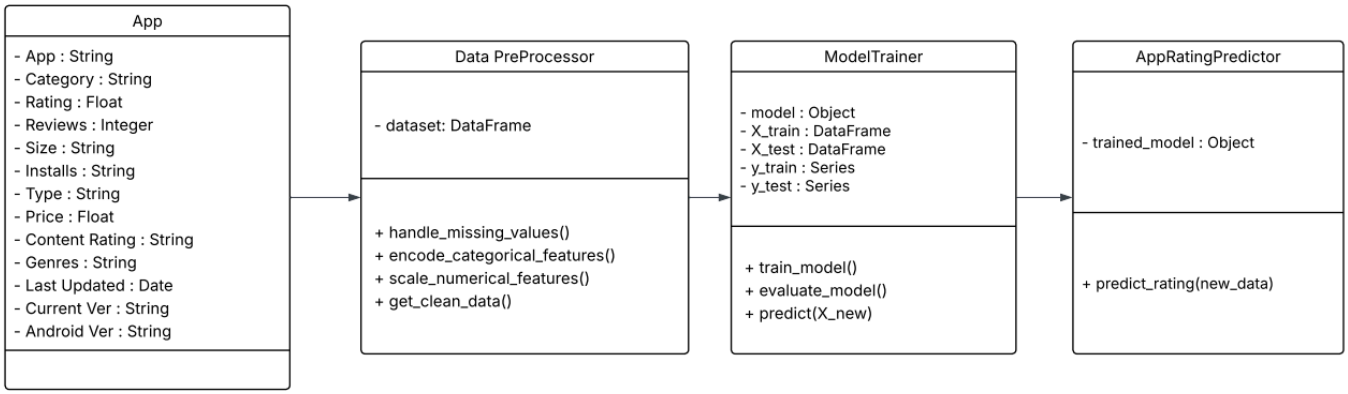

Fig 3.1 Class Diagram

3.3. Model Used

To build an accurate and dependable system for predicting app ratings, we selected the Random Forest Regressor algorithm. Among all the models tested, Random Forest consistently predicted ratings that were closest to the actual user ratings, making it the most reliable choice for this task.. Random Forest is especially suitable for datasets like the Google Play Store, which contain a mix of numerical and categorical features.

Why Random Forest Regressor?

- **High Prediction Accuracy**: Random Forest delivered predictions that closely matched the real-world app ratings in the dataset.

- **Resistance to Overfitting**: By combining the results of multiple decision trees, Random Forest generalizes well and avoids overfitting to the training data.

- **Handles Mixed Data Types**: It performs effectively with both numerical and categorical features, such as Reviews, Size, Type, and Category.

- **Feature Importance**: It provides insights into which features influence the rating the most, helping to understand the decision-making process.

- **Robustness**: Random Forest is stable and performs well even when the data contains noise or less relevant features.

3.4. Model Building Process.

1. **Splitting the Dataset:**
    - The dataset was split into training (80%) and testing (20%) subsets using train_test_split() from scikit-learn.

    - This ensured that the model was evaluated on unseen data to check its generalization ability.
2. **Model Initialization:**
    - The RandomForestRegressor from the sklearn.ensemble module was used for predicting app ratings (a regression task).

    - Key hyperparameters used:

- n_estimators = 200: Builds 200 decision trees and averages their outputs for prediction.
- max_depth = 20: Limits the depth of each tree to prevent overfitting.
- random_state = 42: Ensures consistent results across runs for reproducibility.

3. **Training the Model:**
   - The model was trained using the .fit(X_train, y_train) method.
   - During training, each decision tree learned from different subsets of data and features, forming an ensemble that captures complex relationships.
4. **Code Snippet**

```
from sklearn.ensemble import RandomForestRegressor

rf_model=RandomForestRegressor(n_estimators=200,max_depth=20,random_state=42)

rf_model.fit(X_train, y_train)
```

5. **Model Output:**
   - After training, the model was able to predict continuous rating values for apps based on their features.
   - The predicted ratings were compared against actual ratings using evaluation metrics like Mean Squared Error (MSE) and Mean Absolute Error (MAE).
   - These predictions can also be used in a frontend app to show estimated user ratings based on various app attributes.

3.5. Hardware Requirements

The hardware requirements for developing and deploying this project include a development machine with an Intel i5 processor (or equivalent), 8 GB of RAM (16 GB recommended for smoother performance), and at least 20 GB of free disk space. While a GPU is optional, an NVIDIA GPU is recommended for faster machine learning operations. For server-side deployment, a CPU with 2 to 4 cores is suitable, along with 8

to 16 GB of RAM. A GPU can also be optionally utilized on the server for accelerated AI inference and improved response times.

3.6. Software Requirements

The software requirements for this project include Python 3.9 or higher as the primary programming language due to its strong support for data science and machine learning libraries. The development and testing of the machine learning model are carried out using Jupyter Notebook or Google Colab, which provide interactive environments ideal for data analysis and experimentation. For deploying the frontend, Streamlit is used to create a simple and user-friendly web interface. The project also relies on several essential Python libraries: pandas and numpy for data manipulation and numerical operations, matplotlib and seaborn for data visualization, and scikit-learn for implementing machine learning algorithms and evaluation metrics. These tools collectively enable efficient development, testing, and deployment of the flight delay prediction system.

Chapter 4

Dataset Overview and Feature Engineering

4.1. Dataset Overview

The dataset used in this project is sourced from the Google Play Store and contains information about a wide range of mobile applications available on the platform. Each record represents a single application and includes features that describe its characteristics, usage statistics, and performance.

The dataset comprises **13 primary features** including:

- **App Name**
- **Category** (e.g., Tools, Education, Entertainment)
- **Rating** (target variable, ranging from 1.0 to 5.0)
- **Reviews** (number of user reviews)
- **Size** (in MB or KB)
- **Installs** (total number of downloads)
- **Type** (Free or Paid)
- **Price** (in USD)
- **Content Rating** (e.g., Everyone, Teen)
- **Genres**
- **Last Updated**
- **Current Version**
- **Android Version**

This dataset reflects a mix of numerical, categorical, and textual data. It offers a comprehensive snapshot of mobile app characteristics that may influence user ratings, making it highly suitable for a regression-based prediction task.

4.2. Preprocessing Steps

To prepare the data for machine learning model training, several preprocessing steps were performed:

1. **Handling Missing Values**: Null values were identified and removed to ensure model accuracy and prevent errors during training. Columns with excessive missing data were dropped or filled based on context.
2. **Cleaning and Formatting**:

- Columns like Installs, Price, and Reviews were cleaned by removing characters like commas, dollar signs, or plus signs.
- The Size column had values in both KB and MB and included 'Varies with device'. These were normalized by converting all sizes to a consistent numerical format, and replacing non-numeric values appropriately.

3. **Dropping Irrelevant Features**: Fields like App Name, Last Updated, and Current Version were excluded from model training as they did not provide useful predictive information or were inconsistent.

4. **Target Variable Selection**: The Rating column was selected as the target variable, as the goal of the project is to predict app ratings based on other features.

4.3. Feature Engineering

Feature engineering involved transforming raw data into meaningful inputs that improve model performance:

1. **Label Encoding**: Categorical columns like Category, Type, Content Rating, and Genres were converted into numerical form using Label Encoding. This allows algorithms to process them effectively.

2. **Scaling Numerical Features**: To eliminate bias caused by varying feature scales, numerical columns such as Reviews, Installs, Size, and Price were standardized using **StandardScaler**. This brings them to a common scale with a mean of 0 and a standard deviation of 1.

3. **Log Transformation** (optional): To handle skewed distributions in features like Reviews and Installs, log transformation can be applied to reduce the impact of outliers and improve linearity for regression models.

4. **Outlier Removal**: Data points with unusually high values in features like Price or Installs were either removed or capped to prevent distortion in model training.

5. **Feature Selection**: Features were selected based on correlation analysis and relevance to the target variable. Highly correlated or redundant features were avoided to reduce overfitting and improve generalization.

These engineered features helped in extracting the most meaningful patterns from the dataset, allowing machine learning algorithms to make more accurate and reliable predictions.

Chapter 5

Results and Implementation

5.1 Model Evaluation

To ensure the reliability of our prediction system, we evaluated the trained Random Forest Regressor using appropriate regression metrics. The model achieved a Mean Squared Error (MSE) of 0.2110 and a Mean Absolute Error (MAE) of 0.3081, indicating a low average prediction error and strong overall performance. These results demonstrate the model's effectiveness in accurately predicting app ratings based on various features. Since this is a regression problem, traditional classification metrics like accuracy or confusion matrix were not applicable; instead, MSE and MAE provided a more meaningful assessment of model performance. Overall, the evaluation confirms that the Random Forest model is dependable and suitable for real-world rating prediction tasks.

5.2 Frontend Implementation

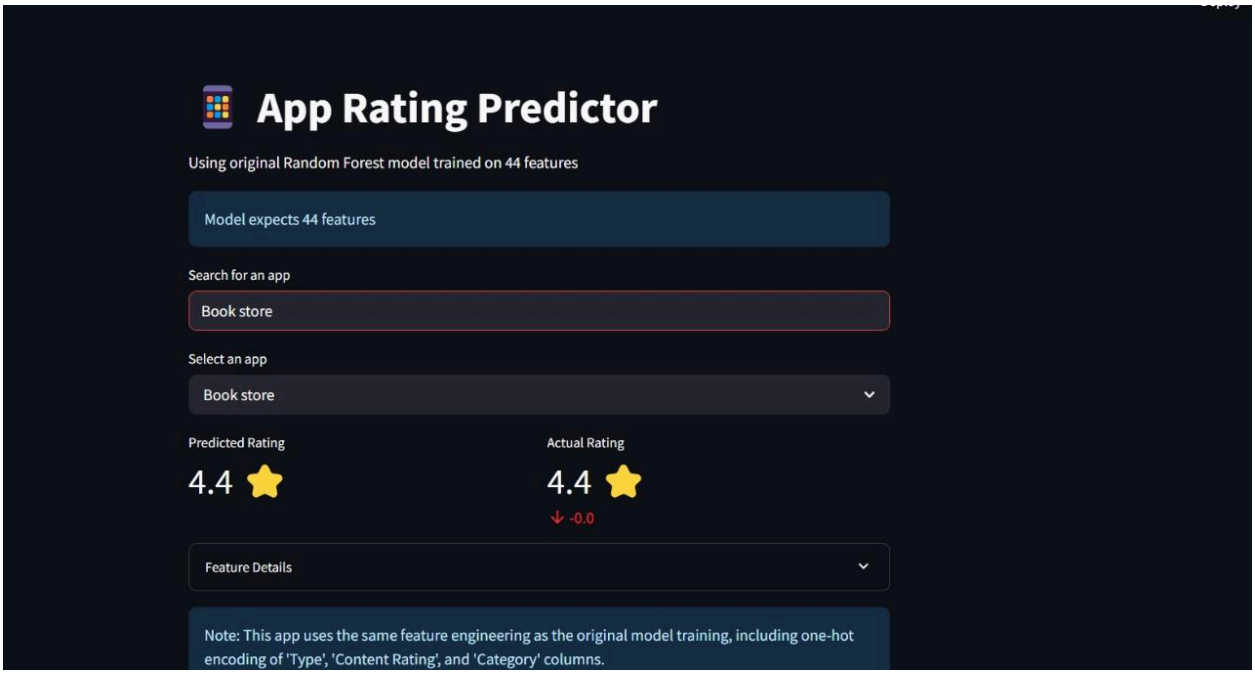

Fig 5.2 Frontend of Project

5.3 Result Analysis

The implementation of the app rating prediction system proved effective in translating machine learning techniques into a practical solution. By analyzing key app-related features such as category, number of installs, reviews, content rating, and app size, the model delivered consistent and meaningful predictions. The Random Forest Regressor demonstrated robustness and adaptability across various input scenarios, accurately estimating ratings even for apps with diverse characteristics. Its performance remained stable on unseen test data, with low error rates indicating strong generalization. These results highlight the model's potential for real-world deployment in enhancing app store analytics and user experience.

5.4 Observation/Remarks

During the development and testing of the app rating prediction system, it was observed that features such as the number of installs, user reviews, content rating, and app category had a noticeable impact on the predicted ratings. The Random Forest Regressor model performed well in learning from historical data, capturing relationships between these variables and the final app rating. It handled feature variations effectively and maintained consistent performance across different input combinations. One strength of the model was its ability to provide reliable predictions using only publicly available metadata, without requiring user-specific or engagement-based data. Overall, the system shows strong potential for supporting app market analysis and helping developers assess likely app performance based on core app attributes.

Chapter 6

Conclusion

6.1. Conclusion

The rapid expansion of the mobile app ecosystem has made it increasingly challenging for developers to gain visibility and for users to identify quality applications. In this project, we addressed this challenge by leveraging machine learning algorithms to predict app ratings based on structured data from the Google Play Store. Our analysis demonstrated that features such as category, number of installs, size, reviews, and price significantly influence an app's rating, and that these attributes can be effectively modeled using regression-based techniques.

Through systematic preprocessing, encoding, scaling, and training of predictive models, we developed a reliable system that provides accurate predictions of app ratings. This not only aids in assessing current applications but also helps developers estimate the potential rating of new apps under development. The results validate the potential of data-driven approaches in guiding strategic decisions related to app design, feature selection, and marketing.

6.2. Future Scope

While the current model lays a strong foundation for predicting app ratings based on structured app metadata, there are multiple opportunities for extending and enhancing the work:

1. **Incorporating User Review Text Analysis:** Future iterations can integrate Natural Language Processing (NLP) to analyze review content, sentiment, and keywords. This would enrich the model with qualitative data, offering a deeper understanding of user perceptions and concerns.
2. **Real-Time Prediction and Monitoring:** Deploying the model as a real-time dashboard or API service could allow developers to monitor changes in app ratings as features or user engagement evolves, enabling quicker adjustments and performance optimization.
3. **Deep Learning Techniques:** Implementing deep learning models such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), or

Transformers can potentially improve prediction accuracy, especially in scenarios involving large and unstructured datasets.

4. **Explainable AI:** Adding interpretability to the model can help developers understand why certain features lead to higher or lower predicted ratings, making the system more transparent and actionable.
5. **Broader Feature Integration**: Future studies can incorporate additional features such as in-app purchase data, update frequency, crash reports, or user demographics to further enhance predictive capability.

6.3. Societal Impact

The implementation of intelligent systems for predicting app ratings offers considerable societal benefits, especially in a world increasingly dependent on mobile applications for day-to-day activities. By enabling developers to make data-informed decisions, this project helps promote the creation of high-quality, user-centered applications. This leads to improved user satisfaction, reduced app abandonment rates, and overall better digital experiences.

For users, such predictive models assist in identifying reliable and relevant apps more efficiently, reducing the risk of downloading low-quality or insecure software. In turn, this contributes to building trust within the app ecosystem, especially for vulnerable or less tech-savvy users who rely heavily on ratings for decision-making.

From a business and economic standpoint, enhanced app quality translates to better market performance, higher user retention, and more efficient use of development resources. On a broader scale, these tools empower startups, individual developers, and small businesses to compete more effectively by providing them with access to predictive insights that were previously limited to larger organizations with advanced analytics capabilities.

Lastly, this project aligns with the larger vision of responsible AI — using machine learning not just for automation or prediction, but for solving real-world problems in ways that are transparent, fair, and beneficial to society at large.

Bibliography

[1] S. Shiva Prakash , Dr.C. Siva Kumar, "App Review Prediction Using Machine Learning" , Journal of Pharmaceutical Negative Results Volume 13 Issue 4 2022 , DOI: 10.47750/pnr.2022.13.04.174

[2] Play Store App Analysis & Rating Prediction Using Classical ML Models & Artificial Neural Network ,August 2023 ,DOI:10.1109/ICCUBEA58933.2023.10391960, Conference: 2023 7th International Conference On Computing, Communication, Control And Automation (ICCUBEA)