**App Rating Prediction Python Project**

**Project Objective**

The primary goal of this project is to develop a predictive model for app ratings based on various app-related features. The project stems from the need of the Google Play Store team to identify and boost promising apps by giving them higher visibility. The focus is on predicting which apps will receive high ratings, as user ratings serve as a crucial indicator of app quality.

**Problem Statement**

The Google Play Store team wants to make some apps more visible because they believe these apps have great potential. They want to predict which apps will get high ratings and should be promoted. To do this, they are examining a dataset to find out what factors influence app ratings.

**Dataset Information**

The dataset, named "googleplaystore.csv," contains information about various apps, including app name, category, rating, reviews, size, installs, type, price, content rating, genres, last updated, current version, and Android version.

**Project Steps**

**Data Loading and Cleaning**

* **Load Data:** Use the pandas library to load the dataset.
* **Check Null Values**: Identify and count null values in each column.
* **Drop Null Records**: Remove records with null values.

**Data Type Correction and Formatting**

Fix Data Types:

* Correct incorrect data types and inconsistent formatting.
* Convert sizes in Kb and Mb to numeric values.
* Convert reviews from string to numeric.
* Convert installs from string to numeric, removing commas and the plus sign.
* Remove the dollar sign from the price column and convert to numeric.

**Sanity Checks**

Perform sanity checks on the data.

* Drop rows with ratings outside the range of 1 to 5.
* Drop rows where reviews exceed installs.
* For free apps, ensure the price is not greater than 0.

**Univariate Analysis**

Conduct univariate analysis.

* Boxplot for Price: Identify outliers.
* Boxplot for Reviews: Check for apps with very high review numbers.
* Histogram for Rating: Observe the distribution of ratings.
* Histogram for Size: Examine the distribution of app sizes.

**Outlier Treatment**

Address outliers in the data:

* Drop records with very high prices.
* Drop records with more than 2 million reviews.
* Identify and drop records with very high install numbers based on percentiles.

**Bivariate Analysis**

Explore relationships between variables and ratings:

* Scatter Plot for Rating vs. Price: Examine the relationship between rating and app price.
* Scatter Plot for Rating vs. Size: Explore the correlation between rating and app size.
* Scatter Plot for Rating vs. Reviews: Assess the relationship between rating and the number of reviews.
* Boxplot for Rating vs. Content Rating: Investigate if content rating affects app ratings.
* Boxplot for Ratings vs. Category: Identify genres with the best ratings.

**Data Pre-processing**

Perform data pre-processing steps:

* Apply log transformation to reduce skewness in Reviews and Installs.
* Drop irrelevant columns such as App, Last Updated, Current Ver, and Android Ver.
* Convert categorical columns (Category, Genres, Content Rating) into dummy variables.

**Train-Test Split**

Split the dataset into training and testing sets using a 70-30 split.

**Model Building and Evaluation**

* Build a linear regression model to predict app ratings.
* Report the R2 score on both the training and test sets.