Analysing Google Play Store Apps

[#] *Team 7: Sajan Kumar Kar, Guruksha Gurnani,*
Shreya Sahay and Gourab Mukherjee

Abstract—**This study delves into the prediction of app success on Google Play Store by integrating statistical methods and machine learning. As app landscape undergoes continuous evolution, deciphering the factors that impact user ratings becomes essential. Developers navigating the dynamic ecosystem gain valuable insights for anticipating app trajectories through the analysis of historical data, user behaviour and app attributes.**

I. Introduction

In the dynamic sphere of mobile applications, the Google Play App Store serves as a bustling marketplace where developers vie for visibility and user engagement. As the app ecosystem continues to expand, the imperative to comprehend and predict the factors influencing app success, specifically in terms of the number of ratings, becomes increasingly crucial. This paper delves into the complexities of this predictive challenge, where a convergence of traditional statistical methods and machine learning techniques seeks to unravel the intricacies behind app performance.

The landscape of app development and user interaction has undergone a profound transformation. The app store is no longer a mere marketplace; it has evolved into a vibrant ecosystem where users actively contribute to shaping the destiny of applications. As developers navigate this dynamic landscape, anticipating the trajectory of app ratings emerges as a pivotal asset in strategic decision-making.
Marketing strategies, user experience, and the myriad functionalities of apps all play pivotal roles in determining an app's success. However, the challenge lies in deciphering the nuanced interplay of these factors and distilling predictive patterns. From the traditional days of app marketing to the contemporary era of data-driven decision-making, the journey has been marked by a seismic shift. This paper aims to navigate through this transition, providing insights into how historical data, user behavior, and app attributes can be harnessed to forecast the number of ratings an app is likely to accumulate.
In an age where technology not only facilitates communication but also serves as a conduit for data collection and analysis, we find ourselves at the crossroads of innovation and prediction. The Google Play App Store, with its extensive repository of applications and user interactions, provides fertile ground for exploration. As we endeavor to predict the rating of an app, we traverse the ever-evolving landscape of app development, harnessing the power of data analytics and machine learning to illuminate the path forward.

II. Data Dictionary

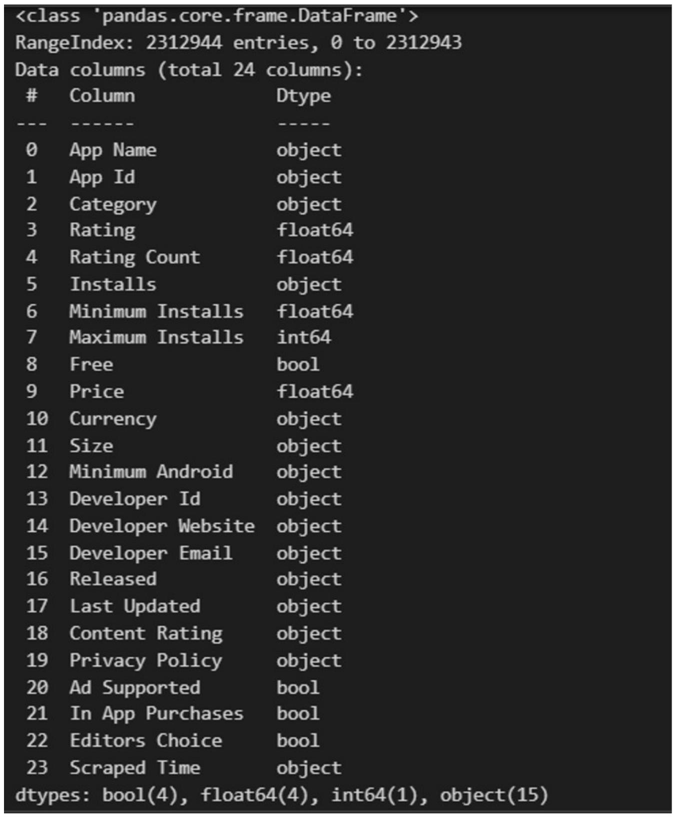
The dataset under examination sourced from Kaggle encompasses a comprehensive array of attributes associated with the Google Play Store Apps. The dataset comprises of over 2.3 million entries spanning across 24 variables. Here is a comprehensive overview of the dataset's attributes:

1. *App Name*: The name of the application
2. *App Id*: A unique identifier assigned to each app on the Google Play Store
3. *Category*: The genre or category defining the app's primary purpose or functionality.
4. *Rating:* The average user rating given to the app by its users.
5. *Rating Count*: The total number of user ratings received by the app.
6. *Installs*: The overall count of app downloads and installations on devices.
7. *Minimum Installs*: The minimum installations required for the app to display on the Google Play Store.
8. *Maximum Installs*: The highest recorded number of installations for the application.
9. *Free*: Indicates whether the app is available for free or involves a cost.
10. *Price*: The cost of the app if not available for free.
11. *Currency*: The currency in which the app's price is listed.
12. *Size*: The storage space occupied by the app on a device, measured in megabytes or gigabytes.
13. *Minimum Android*: The minimum Android operating system version required to run the app.
14. *Developer ID*: A unique identifier for the app's developer or development team.
15. *Developer Website***:** The official website of the app's developer.
16. *Developer Email:* The contact email address of the app's developer.
17. *Released***:** The date when the app was first launched on the Google Play Store.
18. *Last Updated***:** The date of the most recent update or modification to the application.
19. *Content Rating***:** The age group or target audience for which the app's content is deemed suitable.
20. *Privacy Policy***:** A link to the app's privacy policy detailing how user data is handled or protected.

21. *Ad Supported*: Indicates whether the app contains advertisements.
22. *In-App Purchases*: Indicates whether the app offers in-app purchases.
23. *Editor's Choice*: A designation by the Google Play Store Editorial Team highlighting exceptional apps.
24. *Scraped Time*: The timestamp indicating when the data was retrieved or scraped from the Google Play Store.

Metadata:

Our initial unclean dataset encompasses 2,312,944 records and 24 features with 15 categorized as objects, 5 as numeric and 4 identified as Boolean variables. Here’s a visualization of the same below:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2312944 entries, 0 to 2312943
Data columns (total 24 columns):
 #   Column             Dtype
---  ------             -----
 0   App Name           object
 1   App Id             object
 2   Category           object
 3   Rating             float64
 4   Rating Count       float64
 5   Installs           object
 6   Minimum Installs   float64
 7   Maximum Installs   int64
 8   Free               bool
 9   Price              float64
 10  Currency           object
 11  Size               object
 12  Minimum Android    object
 13  Developer Id       object
 14  Developer Website  object
 15  Developer Email    object
 16  Released           object
 17  Last Updated       object
 18  Content Rating     object
 19  Privacy Policy     object
 20  Ad Supported       bool
 21  In App Purchases   bool
 22  Editors Choice     bool
 23  Scraped Time       object
dtypes: bool(4), float64(4), int64(1), object(15)
```

Fig II.1. Dataset Overview

III. Exploratory Data Analysis (EDA)

A. Data Cleaning and Exploration

For the extensive and unclean dataset, various measures were implemented, including addressing missing values, conducting feature engineering, managing outliers, and cleaning the values of specific columns to enhance interpretability. All the points are entailed for the specific columns.

App_ID:

Upon examination, no missing values or duplicates were found in app IDs. Although multiple apps shared the same name with different IDs, this didn't pose an issue. App ID was chosen as the primary key, with five records containing missing app names removed.

Rating & Rating Count

There were 22,883 apps that did not have a rating or a rating count. Since our ultimate target is to predict the rating of an app and impute missing values, no matter how good it won't be considered original, we decided to drop these records. With over 2 million data points, we could afford to drop 22k records, so we proceeded with that approach.

Installs & Minimum Installs

The values in the 'minimum installs' column were of object type with a '+' appended at the end, and they also contained commas. Additionally, it appeared that the values in the two columns were identical. Consequently, we removed the extra characters, treated 'Installs' as a numerical column, and then compared the values to verify if they were indeed the same. The 'Install’ column was dropped.

Currency

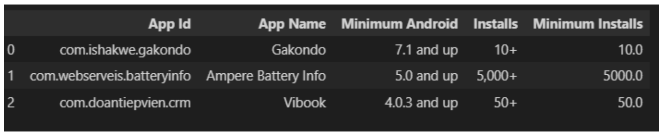
The dataset included various currencies, such as USD, CAD, EUR, INR, VND, GBP, BRL, KRW, TRY, SGD, AUD, and ZAR. Additionally, 'XXX' was used as a special code to indicate no specific currency. The majority of entries were in USD, with some in 'XXX' and a limited number in other currencies. The missing values were removed.

Size

The metrics for size varied, including GB, MB, and KB, with inconsistent capitalization. We aligned the cases and converted all sizes to KB using custom-built functions.

Minimum Android

It had 6526 NaN values, which were dropped considering our huge data size. It had values of the manner - “4.0.3 and up.” It was properly converted to float with format as an android’s version should be (4.0- considering it is the minimum version). There was a ‘varies with device’ category which was removed.



	App Id	App Name	Minimum Android	Installs	Minimum Installs
0	com.ishakwe.gakondo	Gakondo	7.1 and up	10+	10.0
1	com.webserveis.batteryinfo	Ampere Battery Info	5.0 and up	5,000+	5000.0
2	com.doantiepvien.crm	Vibook	4.0.3 and up	50+	50.0

Fig III.1. Dataset Overview (few features)

Developer ID and Developer Email:

With only 32 missing values in the 'Developer Id' column and 31 in the 'Developer Email,' column we opted to exclude rows containing simultaneous missing values in both columns. This decision was influenced by the relatively low percentage of missing data, ensuring a minimal impact on the dataset's overall integrity.

Developer Website
This has a website name which is not required for model building. We used this to create a binary categorical feature indicating whether an app will have a developer's website or not.

Released
We identify the missing values in the "Released" column and impute the missing values with the median date calculated based on the available data. This approach enhances dataset completeness for a more robust analysis. We further went ahead to calculate "the age of the app" by extracting the released date from the current date represented by the "App Age" column of the data frame.

Privacy Policy:
We first identify and handle missing values in the "Privacy Policy" column, replacing them with "NA" for consistency. To facilitate analysis, we introduce a binary feature, "Has_privacy_policy," distinguishing whether an app has a privacy policy (1) or not (0).

Content Rating:
Upon examining the "Content Rating" column through the value count's function, we identified six categories under which the apps are categorized. Given the confusing nature of category names like 'Everyone' and 'Everyone 10+', we improved clarity by redefining them as '17+' and '10+', respectively. Additionally, we replaced 'Adults only 18+' with '18+'. To visualize the distribution across content rating categories, we generated a horizontal bar plot. The plot highlights that a significant majority of apps carry an 'Everyone' label, whereas '18+' rated apps are comparatively scarce.

Last Updated:
In the "Last Updated" column of the Data Frame (df_clean), we extract the year and create a new column called 'Year Last Updated.' This process involves creating a custom function, 'splice string,' to extract the relevant information. Subsequently, the 'Year Last Updated' column is converted to an integer type. By examining the range of this column, we gain insights into the temporal distribution of app updates, revealing that the data spans from 2009 to 2021.

Category:
In our analysis of the "Category" column, we identified 48 distinct categories, with 'Education' having the highest count. The distribution exhibits significant imbalance, skewed to the right, indicating that many categories have relatively fewer values. Upon closer inspection, we recognized that some categories with minor spelling variations are essentially the same (e.g., 'Education' and 'Educational'). To enhance clarity, we consolidated such categories. The top 10 categories, including 'Education,' 'Music,' and 'Business,' were visualized, revealing that 'Education' apps constitute around 20% of the total, emphasizing their prominence in the dataset.

Maximum Installs
We do not really get much information from the maximum Install column.

Price
Using this a 'Price Status' column was featuring engineering indicating binary categorical values of whether the app was paid or free.

Ad supported
This Boolean feature had no issues and was left as it is.

In App- purchases
This Boolean feature had no issues and was left as it is.

Scraped Time
The column was deemed unnecessary and thus dropped.

As we move forward, the refined dataset sets the stage for comprehensive data visualization, enabling us to uncover patterns, trends, and relationships within the Google Play Store app data. This phase will provide a visual narrative to complement the cleaned dataset, enriching our understanding of the factors influencing app success.

B. Data Visualization and Observations

To enhance the clarity and accessibility of our findings, we employ a range of data visualizations in this section. These visual representations serve as powerful tools for elucidating the intricate relationships and patterns present in our dataset. The selection of visualization methods has been guided by the need to effectively communicate complex relationships. Bar charts, scatter plots, and heatmaps have been chosen to offer a diverse perspective, allowing us to capture the nuances and trends within the data.

Here is the list of visualization we are emphasizing to showcase relationships between variables:

1. Top10 categories
2. Distribution of Ratings
3. Distribution of App Releases over time (2010-2021)
4. Scatter Plot between Rating and Content Rating
5. Overlooking App Updates over time
6. Correlation Heat Map

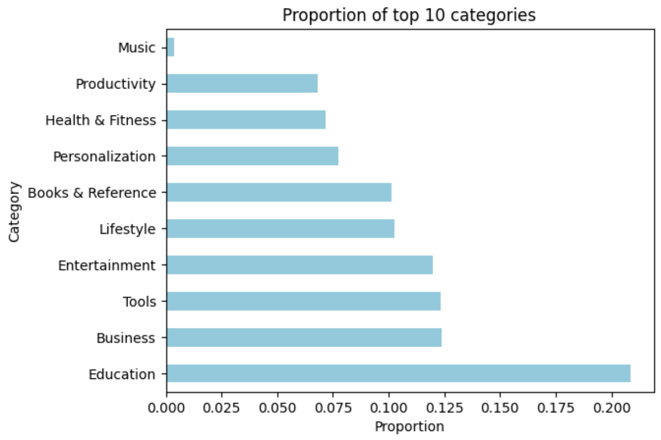


Fig III.1. Top 10 categories

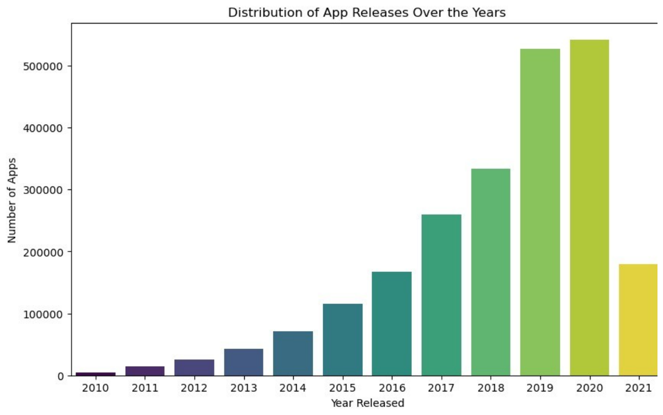


Fig III.3. Distribution of App Releases

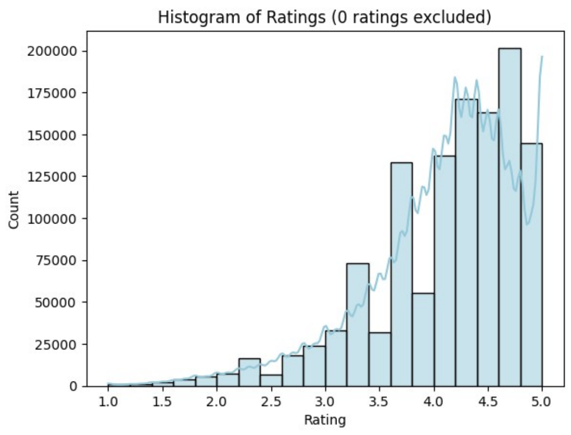


Fig III.2. Distribution of Ratings

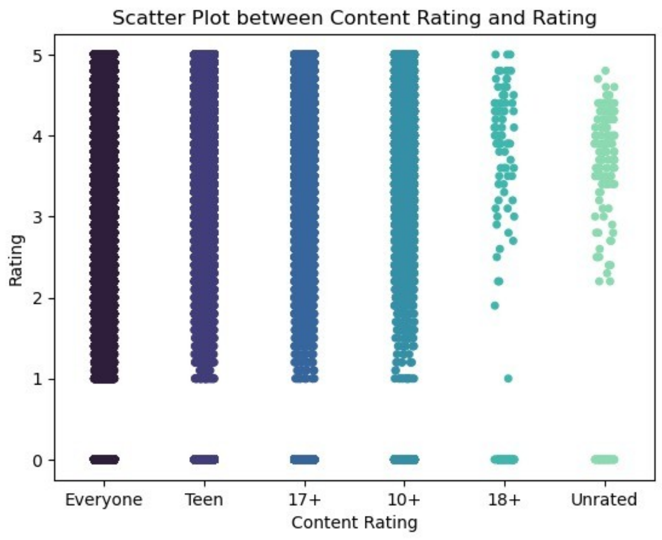


Fig III.4. Scatter Plot between content ratings and ratings

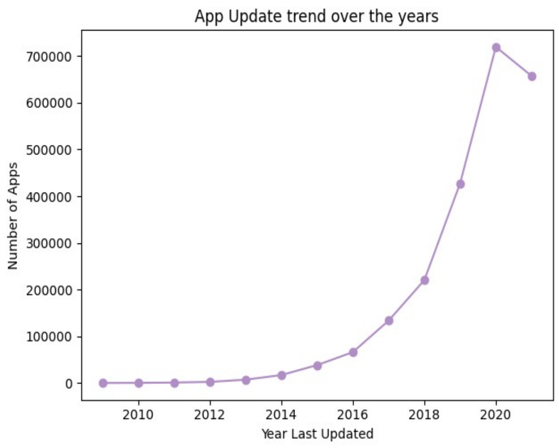


Fig III.5. App Update Trend

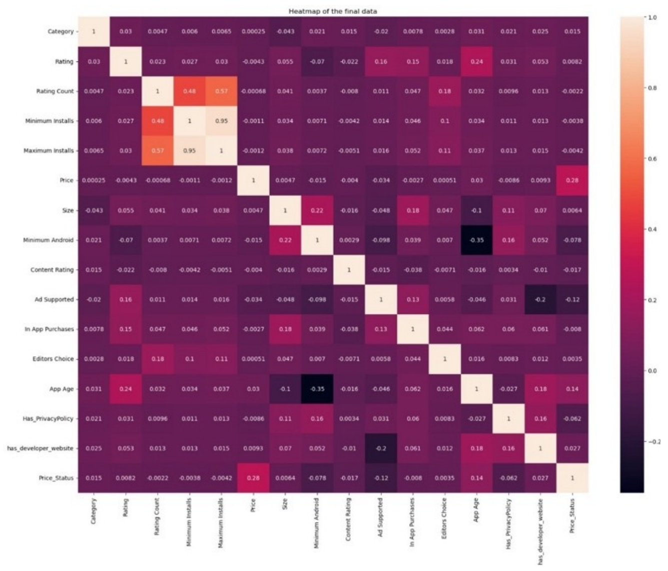


Fig III.6. Correlation Heat Map

C. *Notable Observations/ Trends*

1. In Fig1, we observe that 'Education' category app takes up about 20% of all the applications. We can see the 'Business and 'Tools' category following up and occupying almost identical portions of the customer base. Interestingly, we notice that music apps come at the end of the top categories.

2. In Fig2, we observe on average, the majority of applications are rated between 3.5 to 5. It suggests that individuals tend to either refrain from providing ratings altogether or, when they do, they exhibit a tendency to be relatively moderate in their evaluations. Harsh criticisms are less prevalent, resulting in an overall distribution leaning towards average ratings. Additionally, the observation implies that users may be more inclined to rate an app when they have a positive perception, contributing to the prevalence of higher ratings in the dataset.

3. In Fig3, we observe Application released has increased over time, peaked in the year 2020 followed by a drop in 2021.

4. In Fig4, we observe Applications deemed suitable for "Everyone" have the highest count and have the maximum number of ratings. It's usually the case that apps labeled as unrated get a smaller number of ratings.

5. In Fig5, App trends have increased over time and saw a peak in 2020, followed by a drop around 2021. This is a noteworthy finding because it indicates that the production of Android apps surged notably around 2016, marking a period of substantial growth in the years that followed.

6. In Fig6, we observe the following:

   1. There is a moderate positive correlation of 0.48 between the number of installations and the rating count, indicating that as the number of installs increases, the rating count also tends to increase.

   2. There exists a positive correlation of 0.57 between a maximum number of installs and the rating count suggesting that as the maximum installs increase, the rating count tends to rise as well.

   3. There is a negative correlation of 0.35 between the Age of the app and the mum Android version, indicating that as the app's Age increases the minimum required Android version tends to decrease.

IV. Research / Smart Questions

Our paper aims to answer the following questions:

1. Is there a correlation between "Minimum Android" version and the "Rating" of the app?
2. Does presence of a website impact the rating of the app?
3. Do "Price" and "Category" significantly impact the popularity of an app in terms of number of installs?
4. Are there any significant differences in "Rating" and "Installs" between Editor's Choice Apps and Regular Apps?
5. Does App Size app affect the number of installs?

V. Probing Smart Questions: Hypothesis Testing Approach

Question 1: Exploring the Relationship Between "Minimum Android" Version and App Ratings

Null Hypothesis (H_0): The mean ratings across all categories of "Minimum Android" are not significantly different.

Alternative Hypothesis (H_1): At least one pair of categories has a significantly different mean rating.

Observation:
Upon detailed examination, Android versions 2 and 3 exhibit higher average ratings, while higher Android versions show relatively lower average ratings. Given the eight different versions of "Minimum Android," an ANOVA test is conducted.

Conclusion:
The resulting p-value is below 0.05 (alpha), leading to the rejection of the null hypothesis. This suggests compelling evidence that at least one pair of categories has a significantly different mean rating.

Question 2: Investigating the Impact of Developer Websites on App Ratings

Null Hypothesis (H_0): There is no significant difference in the mean rating for the availability of a developer website.

Alternative Hypothesis (H_1): There is a significant difference in the mean rating for the availability of a developer website.

Observation:
Apps with a developer website tend to have higher mean ratings. To validate this observation, a two-sample independent t-test is conducted.

Conclusion:
The obtained p-value is remarkably small, leading to the rejection of the null hypothesis and the acceptance of the alternative hypothesis (H_1). Consequently, the presence of a developer website indeed impacts an app's rating.

Question 3: Assessing the Impact of "Price" and "Category" on App Popularity

Null Hypothesis (H_0): There is no significant difference in the mean number of installs across different categories.

Alternative Hypothesis (H_1): At least one pair of categories has a significant difference in the mean number of installs.
Two-Part Test:

Price Test:
Null Hypothesis (H_0): There is no significant difference in the mean installs for the price status of apps.
Alternative Hypothesis (H_1): There is a significant difference in the mean installs for the price status of apps.
The resulting extremely small p-value leads to the rejection of the null hypothesis, indicating a significant difference in mean installs based on the price status of apps.

Category Test:
We obtain p-values less than alpha for all categories, leading to the rejection of H_0. Consequently, we conclude that different categories significantly impact the number of installations.

The rejection of the null hypothesis in both the price test and category test suggests significant differences in mean installs based on both the price status of apps and their respective categories.
These findings underscore the importance of considering pricing strategies and app categories in the quest to enhance app popularity.

Question 4: Exploring Differences in "Rating" and "Installs" for Editor's Choice Apps

Rating for Editor's Choice:
Null Hypothesis (H_0): There is no significant difference in the mean rating for Editor's Choice.
Alternative Hypothesis (H_1): There is a significant difference in the mean rating for Editor's Choice.
The extremely low p-value leads to the rejection of the null hypothesis, indicating a difference in the mean rating for apps designated as Editor's Choice compared to those that are not.

Installs for Editor's Choice:
Null Hypothesis (H_0): There is no significant difference in the mean installs for Editor's Choice.
Alternative Hypothesis (H_1): There is a significant difference in the mean installs for Editor's Choice.
Another smaller p-value results in the rejection of H_0, indicating a significant difference in the mean installs for apps designated as Editor's Choice.

In conclusion, our analysis suggests that the Editor's Choice designation not only influences the mean rating but also significantly impacts the number of installs. These insights provide valuable considerations for developers and users alike, emphasizing the reliability and popularity of apps distinguished by the Editor's Choice label on the Google Play Store

Question 5: Investigating the Relationship Between App Size and Number of Installs

Null Hypothesis (H_0): The app size does not affect the number of installs.

Alternative Hypothesis (H_1): The app size affects the number of installs.

Observation: Patterns emerge where the highest minimum installs are associated with smaller-sized apps, while larger apps tend to have considerably fewer minimum installs. This suggests a potential relationship between app size and the number of installs.
To explore this relationship, a linear regression analysis using Ordinary Least Squares (OLS) method was performed for

both "Minimum Installs" and "Maximum Installs" against "Size." Despite statistically significant 'Size' coefficients, the low R-squared values indicate that 'Size' explains only a small fraction of the variability in 'Minimum Installs' and 'Maximum Installs.' Additionally, high AIC and BIC values, along with diagnostic tests on residuals, suggest potential issues with model assumptions and multicollinearity.

Conclusion:
In conclusion, while the models show statistical significance, their limited explanatory power and potential issues with residuals and multicollinearity indicate the need for further refinement and exploration, considering additional variables for a more robust analysis.
Additionally, a simple correlation analysis using the average number of installs supports our findings, affirming that app size does indeed affect the number of installs.

In the exploration of Google Play Store apps through a series of SMART questions, we have unearthed valuable insights that shed light on various facets of app performance, user engagement, and factors influencing popularity. The key findings are as follows:

1. *"Minimum Android" Version and Ratings*:
   Significant differences in mean ratings were discovered across different versions of the Android operating system. This underscores the importance of considering compatibility in app development to enhance user satisfaction and ratings.

2. *Impact of Having a Developer Website:*
   The presence of a developer website significantly influences app ratings and installs. Apps with a dedicated website tend to garner higher user engagement and ratings, emphasizing the relevance of establishing an online presence.

3. *"Price" and "Category" Influence on Popularity*:
   Both price status and app category play pivotal roles in determining app popularity, as indicated by significant differences in mean installs.

4. *Differences in "Rating" and "Installs" for Editor's Choice Apps:*
   Apps designated as Editor's Choice exhibit significantly higher mean ratings and installs compared to regular apps. This designation serves as a reliable indicator of app quality and popularity, guiding users and developers in their app selection.

5. *App Size and Number of Installs:*
   App size does influence installs, but additional variables need consideration for a more comprehensive analysis.

VI. Data Preprocessing

During the data pre-processing phase, we streamlined the dataset by eliminating unnecessary features, including:

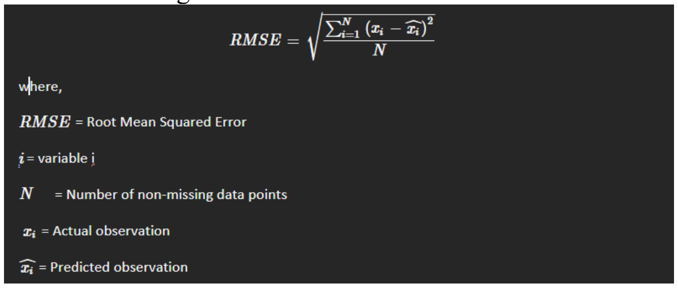
1. Free
2. App Id
3. App Name
4. Developer Website
5. Developer Email
6. Developer Id
7. Privacy Policy
8. Average Installs
9. Released
10. Month Released
11. Year Released
12. Year Last Updated
13. Last Updated
14. Currency

To facilitate the analysis, we converted 'Ad-Supported' and 'In-App Purchases' columns to numeric format. Additionally, we applied label encoding to the 'Price Status', 'Ad-Supported', 'In-App Purchases', and 'Editor's Choice' columns, treating them as binary nominal categorical features.

To ensure uniformity, the 'Currency' column was initially converted to INR using a Currency Converter, and the corresponding 'Price' values were adjusted accordingly. Finally, the dataset was partitioned into training and test sets, setting the stage for subsequent model building and analysis.

VII. Model Building

The task at hand involves a regression problem where our objective is to predict an app's rating based on various features. As we delve into evaluation metrics, we note that the R2 score increases with the growing number of features. Due to the already rich feature set in our dataset, we opt for the Root Mean Square Error (RMSE) as our preferred evaluation metric for the regression model.



$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \widehat{x_i})^2}{N}}$$

where,

$RMSE$ = Root Mean Squared Error

i = variable i

N = Number of non-missing data points

x_i = Actual observation

$\widehat{x_i}$ = Predicted observation

Fig VII.I. RSME

To ensure consistency, we establish a seed before embarking on our model selection journey, starting with a fundamental Linear Regression model as our baseline. It's worth noting that the app rating ranges from 0 to 5.

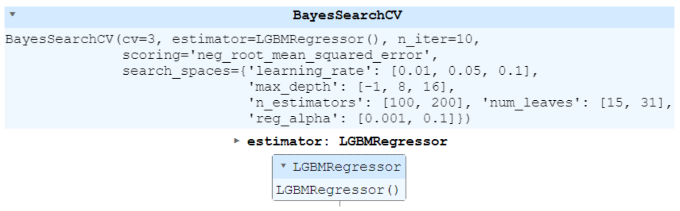
Linear Regression Model: Our baseline Linear Regression model yields an RMSE of approximately 3.96 on the training

set and a slightly higher value of 3.98 on the test set, which, in our context, is suboptimal. Fortunately, the model shows minimal overfitting, providing a silver lining.
SGDRegressor: Transitioning to the SGDRegressor, the RMSE takes a substantial leap, reaching an astonishingly high value of 5.57e+45 on the training set and 5.26e+45 on the test set.
These results underscore the limitations of these basic models, prompting us to explore more robust techniques.

LGBM Regressor: The introduction of the LGBM Regressor, a gradient boosting algorithm, proves highly effective, achieving an impressive RMSE of 0.19. This remarkable performance prompts a comparison with a Decision Tree Regressor.
Decision Tree Regressor: While the Decision Tree Regressor produces a commendable RMSE of 0.2, the LGBM model still outshines it, demonstrating an even lower RMSE. Hence, we designate the LGBM Regressor as our top-performing model, showcasing its superiority in predictive accuracy within our dataset.

*A. **Hyperparameter Tuning***

In our effort to enhance the model's performance, we fine-tuned it for better accuracy, resulting in our final model. Unlike traditional methods, we chose a Bayesian search for its efficiency with our large dataset. Our aim was a more computationally efficient tuning approach.



```
BayesSearchCV
BayesSearchCV(cv=3, estimator=LGBMRegressor(), n_iter=10,
              scoring='neg_root_mean_squared_error',
              search_spaces={'learning_rate': [0.01, 0.05, 0.1],
                             'max_depth': [-1, 8, 16],
                             'n_estimators': [100, 200], 'num_leaves': [15, 31],
                             'reg_alpha': [0.001, 0.1]})
estimator: LGBMRegressor
LGBMRegressor
LGBMRegressor()
```

Fig VII. II. Bayesian Search(params)

After exploring various hyperparameter combinations, we identified the most effective ones:

1. Learning rate: 0.1
2. Max depth: 16
3. N_estimators: 153
4. Num_leaves: 31
5. Reg_alpha: 0.06383671801269114

While the optimization impact may not be revolutionary, our meticulous fine-tuning led to a noteworthy reduction in the RMSE value, progressing from 0.1934 to 0.1919.

Final RMSE Values:

- Train RMSE: 0.1919
- Test RMSE: 0.1924

Conclusion:
In our pursuit of model excellence, the incorporation of hyperparameter tuning has ushered in a refined version of our model. The Bayesian search method, chosen for its computational efficiency, navigated the expansive search space adeptly. The resultant set of optimized hyperparameters has perceptibly improved predictive accuracy, contributing to a more robust and precise model for predicting app ratings based on our feature-rich dataset.

Conclusion

To summarize, our analysis of the relationship between app size and the number of installs reveals intriguing insights and a nuanced modelling journey. While simple correlation analysis and linear regression initially indicated a statistically significant association, the limited explanatory power and concerns about model assumptions prompted a more thorough exploration. The adoption of advanced models, including the LGBM Regressor, showcased superior predictive performance, outperforming traditional linear models, and highlighting the importance of leveraging sophisticated algorithms. The Bayesian search for fine-tuning the LGBM model yielded a slight improvement in predictive accuracy, reducing the RMSE from 0.1934 to 0.1919. Despite the incremental gain, this enhancement demonstrates the potential for optimization in predictive modelling. It is crucial to acknowledge that app installs are influenced by various factors beyond app size, and our model, while capturing a portion of this variability, may benefit from further refinement and consideration of additional variables for a more comprehensive understanding. Overall, the LGBM Regressor, with its robust performance and optimized tuning, stands as our top-performing model in predicting app installs based on size within the given dataset.

References

[1] Hodson, T. O. (2022, July 19). *Root-mean-square error (RMSE) or mean absolute error (mae): When to use them or not*. Geoscientific Model Development https://gmd.copernicus.org/articles/15/5481/2022/

[2] *How google play works*. How Google Play Works. (n.d.). https://play.google/howplayworks/

[3] Pandian, S. (2022, October 20). *A comprehensive guide on hyperparameter tuning and its techniques*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/02/a-comprehensive-guide-on-hyperparameter-tuning-and-its-techniques/

[4] Shi, A. and K. (2023, September 24). *Sgdregressor with scikit-learn: Untaught lessons you need to know*. Medium. https://towardsdatascience.com/sgdregressor-with-scikit-learn-untaught-lessons-you-need-to-know-cf2430439689