TEPIAN Vol. 2 No. 2 (June 2021) p-ISSN 2721-5350 e-ISSN 2721-5369
doi.org/10.51967/tepian.v2i2.404

Android App Rating Classification on Google Play Store Using Random Forest Algorithm with SQL Server Preprocessing

Raissa Maringka
Amikom University, Yogyakarta, 55281, Indonesia
raissa.1273@students.amikom.ac.id

Aulia Khoirunnita*
Amikom University, Yogyakarta, 55281, Indonesia
aulia.1270@students.amikom.ac.id
** Corresponding author*

Rodney Maringka
Amikom University, Yogyakarta, 55281, Indonesia
rodney.1272@students.amikom.ac.id

Ema Utami
Amikom University, Yogyakarta, 55281, Indonesia
ema.u@amikom.ac.id

Kusnawi
Amikom University, Yogyakarta, 55281, Indonesia
khusnawi@amikom.ac.id

Received: 2021-04-07; Revised: 2021-04-29; Accepted: 2021-05-22; Published: 2021-06-01

Abstract- The increasing number of Android applications available (App) on the Google Play Store with the benefits the developers get has attracted the attention of many Android application developers. To benefit from developing Android apps, one way is to know the characteristics of highly rated apps on the Google Play Store. This research will investigate the features of size, installs, reviews, type (free / paid), rating, category, content rating, and price on applications on the Google. To classify high-ranking applications, the authors use 8-fold cross validation using the Random Forest algorithm and get better results than the Gradient Boost, K-NN, and Decision Tree algorithms with an accuracy of 83%. The results of the Random Forest algorithm also have better performance than the algorithm from the previous research conclusions, with a 0.8% increase in accuracy. To classify high-ranking applications, the authors use 8-fold cross validation using the Random Forest algorithm and get better results than the Gradient Boost, K-NN, and Decision Tree algorithms with an accuracy of 83%. The results of the Random Forest algorithm also have better performance than the algorithm from the previous research conclusions, with a 0.8% increase in accuracy. To classify high-ranking applications, the authors use 8-fold cross validation using the Random Forest algorithm and get better results than the Gradient Boost, K-NN, and Decision Tree algorithms with an accuracy of 83%. The results of the Random Forest algorithm also have better performance than the algorithm from the previous research conclusions, with a 0.8% increase in accuracy.

Keywords- Random-Forest Classifier, Feature Important, Performance Evaluation, Root Mean Squared Error, 8-Fold Cross Validation

I. Introduction

Currently the number of Android applications with rapidly growing users affects the rapidly developing economy (App Download and Usage Statistics (2020) - Business of Apps, nd). According to the 2018 report, the economic value of global applications was USD 81.7 billion in 2017, and increased to USD 106.4 billion in 2018, and is expected to increase to USD 156.5 by 2022 (The 2017-2022 App Economy Forecast: 6 Billion Devices, $ 157 Billion in Spend & More, nd). The rapidly growing mobile phone application market is attracting the attention of many developers with more than 11 million developers already submitting their applications on the Google Play Store and App Store according to a 2016 report (There Are 12 Million Mobile Developers Worldwide, and Nearly Half Develop for Android First , nd). Based on data from the fueled article, it is reported that the average income per Android application developer on Google is USD 97,600 per year (How Much Money Can You Earn With an App in 2019? | Fueled, nd). Many Android app developers are still struggling to find ways to profit from their apps by including them on the Play Store. Knowing the characteristics of a successful application can be a way for developers to create successful applications.

Various characteristics can be used as benchmarks for making a successful application. Just like previous studies (Bavota et al., 2015), in this study the authors chose ranking ratings as a benchmark for successful applications, therefore the factors or features that influence application success, namely the characteristics of high-ranking applications, need to be known. Previous research on the prediction of success from applications on the google play store, researchers obtained random forest model results that predict the success of applications on the google play store with an accuracy of 82.92% (Mueez et al., 2018).

To classify a successful application it can be seen from the number of downloads, but many users download the application without using it. The Google Play Store displays the number of downloads based on a range, not the actual value (example: 10M + Downloads) which

makes it difficult to distinguish between apps that are in the same range. Another way to identify successful apps is to analyze the comments that users enter when rating them, but many users rate them without including comments.

In previous research on the analysis of factors that affect application ratings, it was found that applications with high ratings had lower API bug fix values compared to applications with low ratings after the data was tested with the Mann-Whitney test (p-value <0.0001) and Cliff's Delta (0.37). . This shows that there is a relationship between application rankings with other factors such as changes in Android APIs and the complexity of the user interface (Bavota et al., 2015).

Then in other research on the effect of advertisements (Ad Libraries) on Android rankings, after the data was tested with the Spearman rank correlation between ad libraries in an app and app rankings, the value was 0.016 (weak correlation). It was found that the more advertisements that developers put into their applications, it turned out that they did not really have an effect on Android rankings (Ruiz et al., 2014). Previous research by Aralikatte (2018) which tested the correlation between application rankings and the average sentiment value (+ 1 = positive correlation, 0 = no correlation, and -1 = total negative correlation) using the Pearson and Spearman correlation resulted in a value of 0.5 for each correlation, which states that there is a correlation even though it is not large (Aralikatte et al., 2018).

Previous research by Harman et al. (2012) which tested the correlation between prices, ratings, and downloads from the Blackberry App Store, using the Spearman Correlation, it was found that there was a strong correlation between ratings and downloads, namely 0.79, and a low correlation value of 0.12 between price and download (Harman et al. al., 2012). The purpose of this study is to examine other factors or features related to app rating ratings and to examine the most influential features to identify high ranking apps.

The features studied are size, installs, reviews, types (free / paid), rating, category, and price according to the dataset from kaggle (Google Play Store Apps | Kaggle, nd). In this study, the authors used a random forest to investigate the important features that affect the ranking of applications. The author chose random forest because this classification method has given good results and the system is good at avoiding overfitting (Why Random Forest Is the Greatest! | By Super Albert | The Making Of… a Data Scientist | Medium, nd). The benefit of this research is that in the future it will be able to provide useful information for application developers and users to find out the characteristics of high-ranking Android applications.

II. RESEARCH METHODS

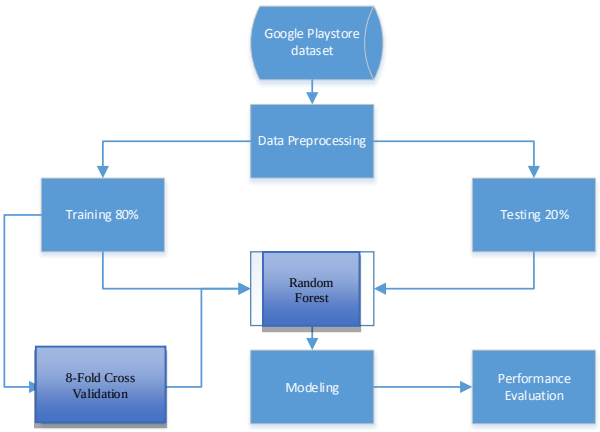
The data used in this study is the Google Play Store dataset which can be accessed at Kaggle (Google Play Store Apps | Kaggle, nd). This dataset has 10842 rows, and 13 attributes which are described in Table 1.

Table 1. Dataset Attributes

Attribute	Detail
App	Application name
Category	Type the category of the application
Rating	How big is the rating of the application
Review	How many reviews from users
Size	The size of the application
Installs	The number of users who installed the application
Type	The type of application
Price	Application price
Content Rating	Who this app is made for
Genres	A more specific type of application
Last Updates	When was the last time the application was updated
Current Ver.	The latest version of the application
Android Ver.	Android version that can use the application

A. *Research design*

Picture 1 shows the ranking process for Android apps on the Google Play Store. The first process is taking the dataset from Kaggle, followed by data preprocessing to process data using the SQL server. The dataset is divided into 80% training data consisting of 8671 data divided into 7956 high rated and 715 low rated data, and 20% testing data consisting of 2167 data divided into 1988 high rated and 179 low rated data. This study used 8-fold cross validation to divide the data into 8 parts and tested it 8 times before modeling. Then the data is processed using the random forest algorithm, which will then be modeled and evaluated, as shown in Picture 1.



Picture 1. Architecture for Android Application Rating Classification

B. *Preprocessing Data*

Preprocessing data is divided into 2 parts, namely: cleaning data and data reduction. Data cleaning is the process of cleaning incomplete data on the attributes in the dataset to make the data more consistent. Meanwhile, data reduction is the process of removing data on less dominant attributes so that data can be reduced, but still produce

accurate data. In the data cleaning process, the writer classifies and assigns the ranking data label to be high rated (> 3.5) and low rated (≤ 3.5), removes the k and m symbols in the size column, removes the + symbol in the installs column and in the data reduction process the writer deletes the data that is is in the attributes current version, android version, genre, and last updated (see Picture 2).

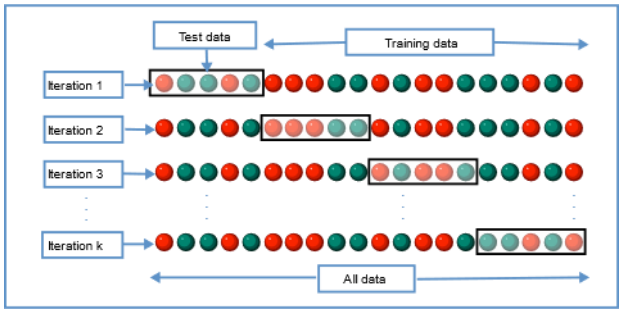
```
UPDATE apps
SET Rating_Numeric = 'high rated'
WHERE Rating_Numeric > 3.5;

UPDATE apps
SET Rating_Numeric = 'low rated'
WHERE Rating_Numeric <= 3.5;
```

Picture 2. SQL Server Data Preprocessing

C. 8-Fold Cross Validation

After the data has been divided into 80% training data and 20% testing data, 8-fold cross validation will be carried out on the training data. Cross Validation is a technique for evaluating a model by partitioning the original sample into a training set to train the model, and a test set to evaluate the model. In k-fold cross validation, the original sample is randomly partitioned into a k equal size subsample. From subsample k, one subsample will be used as testing data and the rest will be training data. The cross validation process will be repeated k times (multiples), with each subsample k being used once as data validation(A Gentle Introduction to K-Fold Cross-Validation, nd). Picture 3 shows the 8-fold cross validation process, data is divided into 8 partitions and will be tested 8 times before the model is made. Can be seen in Picture 3.



Picture 3. 8-Fold Cross Validation

D. Random Forests

Random Forests or random decision forests are an ensemble learning method for classification, regression, and other tasks that operate by building multiple decision trees at the time of training and issuing classes which are class mode (classification) or average / average (regression) prediction of each. - each tree (Ho, 1995).

The Random Forest algorithm is used to solve regression and classification problems, making it a diverse model that is widely used by engineers. The Random Forest algorithm consists of different decision trees, each with the same node, but using different data leading to different leaves. It combines the decisions from several decision trees to find an answer, which represents the average of all these decision trees. The random forests algorithm is a supervised learning model; this model uses labeled data to "learn" how to classify unlabeled data.

When performing Random Forests based on classification data, it is important to note that the Gini index should be used (Random Forest Algorithm for Machine Learning | by Madison Schott | Capital One Tech | Medium, nd), the formula used to decide how the nodes are in the decision tree branch (1).

$$Gini = 1 - \sum_{i=1}^{c}(P_i)^2 \quad (1)$$

This formula uses class and probability to determine the Gini of each branch on the node, determining which branch is more likely to occur. Here, pi represents the relative frequency of the classes observed in the dataset and c represents the number of classes.

Random Forests also can use entropy to determine how nodes branch in a decision tree (2).

$$Entropy = 1 - \sum_{i=1}^{c} -P_i * log_2(P_i) \quad (2)$$

Entropy uses certain possible outcomes to make decisions about how nodes should branch (Random Forest Algorithm for Machine Learning | by Madison Schott | Capital One Tech | Medium, nd). Unlike the Gini index, it is more mathematically intensive due to the logarithmic function used to calculate it.

E. Performance Evaluation

After making the model, the next step is to evaluate with performance evaluation. Performance evaluations are useful for testing the performance of a classifier. Recall, precision, and accuracy. Recall is a collection of positive data that is classified correctly as positive data. Precision is a data set classified as positive that is really positive. Accuracy is the accuracy of data classification(Accuracy, Precision, Recall or F1? | By Koo Ping Shung | Towards Data Science, nd).

The following is the formula for recall, precision, and accuracy in performance evaluation (3) (4) (5)

$$Recall = (TP) / (TP + FN) \quad (3)$$

$$Precision = (TP + TN) / (TP + TN + FP + FN) \quad (4)$$

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (5)$$

Information (3) (4) (5):

TP : The value is true positive
TN : Value is true negative
P : Number of positive data
FP : False positive value
N : The number of negative data
FN : False negative value

Maringka, R., Khoirunnita, A., Maringka, R., Utami, E., & Kusnawi. (2021). Android App Rating Classification on Google Play Store Using Random Forest Algorithm with SQL Server Preprocessing. TEPIAN, 2(2). https://doi.org/10.51967/tepian.v2i2.404

F. Feature Important

The important feature method plays an important role in selecting significant attributes, by eliminating irrelevant attributes, and therefore it can be used to identify influential attributes. The author uses the information gain ratio method to determine how much influence an attribute has in the dataset. Machine learning information gain can be used to rank attributes that have high information gain and must be rated higher than other attributes because they are more influential in classifying data.(Feature Selection Using Information Gain | by Muhammad Yunus | Medium, nd). The following is the formula for information gain (6) (Azhagusundari & Thanamani, 2013):

$$IG(A) = H(S) - \sum \frac{Si}{S} H(S_i) \quad (6)$$

Information (6)
H (S) : Entropy of the dataset
H (Si) : Entropy of the subset i produced by partition S
A : Attributes in the dataset

G. Root Mean Squared Error

Root Mean Squared Error (RMSE) is the standard deviation of the residuals (prediction error). The residual is a measure of how far away from the regression line point; The RMSE is a measure of how this residual is dispersed(RMSE: Root Mean Square Error - Statistics How To, nd). The following is the RMSE (7)

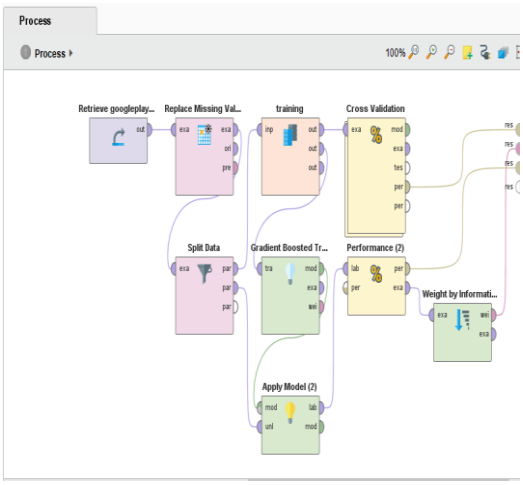
$$RMSE = \sqrt{(f-0)^2} \quad (7)$$

Information (7)
f: Estimate (expected value or unknown result)
o: Observed value (known result)

III. Results And Discussion

A. Random Forest Process

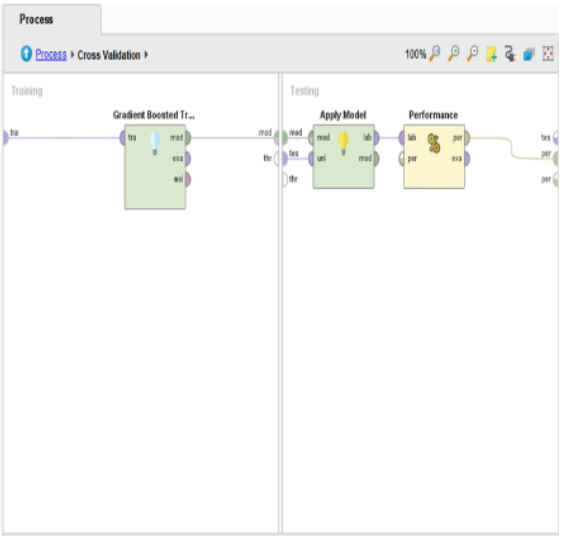
An algorithm used in the classification of large amounts of data. Random forest classification is carried out by merging trees by conducting training on the sample data they have. The tree that is built is recursively divided from data in the same class can be seen in Picture 4.



Picture 4. Random Forest Process

B. Cross Validation Process

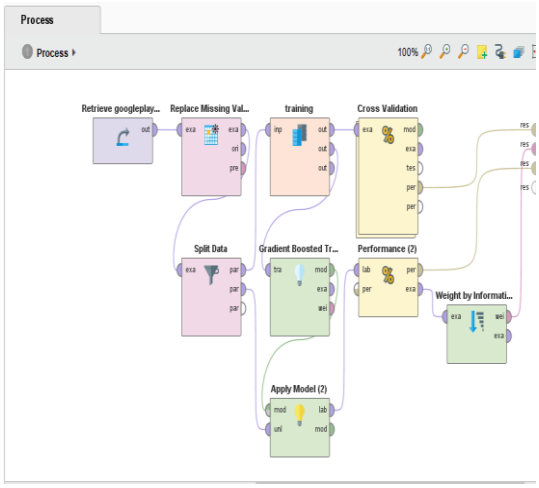
is a statistical method that can be used to evaluate the performance of a model or algorithm where the data is separated into two subsets, namely learning process data and validation / evaluation data which can be seen in Picture 5.



Picture 5. Cross Validation Process

C. Feature Important Results

Based on Picture 6, it can be concluded that the Reviews and Installs attribute is the most influential in predicting high-rated applications with a weight of 0.0374 for Reviews, and 0.0258 for Installs, for other attributes, Apps Name is 0.075, Size is 0.052, Price is 0.051, Content Rating is worth 0.010, Category has a value of 0.009, and Type is worth 0.003.



Picture 6. Feature Important Results

D. Independent Performance Evaluation Results Dataset Against Android App Ratings

Tabel 2 and Picture 7 show, without using Cross Validation, namely with an independent dataset. From the results it can be seen that the Random Forest algorithm has the highest accuracy and precision of 83.12% and 83.09% compared to other algorithms. The smallest root mean square error is generated by the Random Forest and Gradient Boost algorithm of 0.348.

Table 1. Independent Performance Evaluation Results

Independent Dataset				
Algorithm	Accuracy (%)	Recall (%)	Precision (%)	RMSE
Random Forest	83.12%	70.02%	83.09%	0.348
K-NN	78.14%	69.78%	71.20%	0.395
Gradient Boost	81.09%	76.94%	75.31%	0.348
Decision Tree	83.03%	70.25%	82.34%	0.357

PerformanceVector

```
PerformanceVector:
accuracy: 83.12%
ConfusionMatrix:
True:   high rated      low rated
high rated:     1562    317
low rated:      49      240
weighted_mean_recall: 70.02%, weights: 1, 1
ConfusionMatrix:
True:   high rated      low rated
high rated:     1562    317
low rated:      49      240
weighted_mean_precision: 83.09%, weights: 1, 1
ConfusionMatrix:
True:   high rated      low rated
high rated:     1562    317
low rated:      49      240
root_mean_squared_error: 0.348 +/- 0.000
```

Picture 7. Results of Random Forest Performance

Table 2 and Picture 8 show, performance evaluation of 8-Fold Cross Validation Dataset against Android Applications. Table 2 shows the results of the performance evaluation of the classification algorithm using 8-fold cross validation. From these results it is known that the Random Forest algorithm has the highest accuracy and precision compared to other algorithms with an accuracy value of 83%, precision of 82.97 %.

Table 2. Independent Performance Evaluation Results

8 Fold Cross Validation Dataset				
Algorithm	Accuracy (%)	Recall (%)	Precision (%)	RMSE
Random Forest	83%	69.79%	82.97%	0.347
KNN	78.15%	68.99%	71.18%	0.395
Gradient Boost	81.33%	76.74%	75.62%	0.347
Decision Tree	82.54%	69.18%	82.11%	0.357

PerformanceVector

```
PerformanceVector:
accuracy: 83.00% +/- 0.78% (micro average: 83.00%)
ConfusionMatrix:
True:   high rated      low rated
high rated:     6250    1278
low rated:      196     949
weighted_mean_recall: 69.79% +/- 1.46% (micro average: 69.79%), weights: 1, 1
ConfusionMatrix:
True:   high rated      low rated
high rated:     6250    1278
low rated:      196     949
weighted_mean_precision: 82.97% +/- 1.25% (micro average: 82.95%), weights: 1, 1
ConfusionMatrix:
True:   high rated      low rated
high rated:     6250    1278
low rated:      196     949
root_mean_squared_error: 0.347 +/- 0.007 (micro average: 0.347 +/- 0.000)
```

Picture 8. Performance Vector - Confusion Matrix Random Forest through 8 - fold cross validation

IV. CONCLUSION

From the 4 evaluated algorithms, namely Random Forest, K-Nearest Neighbor, Gradient Boost, and Decision Tree, it can be concluded that the Random Forest algorithm has the best performance among other algorithms with 83.12% accuracy, 83.09% precision, and 0.348 RMSE for results. independent while the results for 8 - fold cross validation have results of 83% accuracy, 82.97% precision, and 0.347 RMSE in classifying high rated and low rated android applications in the google play store. The results of the Random Forest algorithm also have better performance than the algorithm from the previous research conclusions, with an accurate 0.8% increase.

REFERENCES

A Gentle Introduction to k-fold Cross-Validation. (nd). Retrieved November 8, 2020, from https://machinelearningmastery.com/k-fold-cross-validation/

Accuracy, Precision, Recall or F1? | by Koo Ping Shung | Towards Data Science. (nd). Retrieved November 8, 2020, from https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

App Download and Usage Statistics (2020) - Business of Apps. (nd). Retrieved November 8, 2020, from https://www.businessofapps.com/data/app-statistics/

Aralikatte, R., Sridhara, G., Gantayat, N., & Mani, S. (2018). Fault in your stars: An analysis of android app reviews. ACM International Conference Proceeding Series. https://doi.org/10.1145/3152494.3152500

Azhagusundari, B., & Thanamani, AS (2013). Feature Selection based on Information Gain. International Journal of Innovative Technology and Exploring Engineering (IJITEE).

Bavota, G., Linares-Vásquez, M., Bernal-Cárdenas, CE, Di Penta, M., Oliveto, R., & Poshyvanyk, D. (2015). The impact of API change- and fault-proneness on the user ratings of android apps. IEEE Transactions on Software Engineering. https://doi.org/10.1109/TSE.2014.2367027

Feature Selection using Information Gain | by Muhammad Yunus | Medium. (nd). Retrieved November 8, 2020, from https://medium.com/@yunusmuhammad007/feature-selection-mengusing-information-gain-ba94ca66f658

Google Play Store Apps | Kaggle. (nd). Retrieved November 8, 2020, from https://www.kaggle.com/lava18/google-play-store-apps

Harman, M., Jia, Y., & Zhang, Y. (2012). App store mining and analysis: MSR for app stores. IEEE International Working Conference on Mining Software Repositories. https://doi.org/10.1109/MSR.2012.6224306

Ho, TK (1995). Random decision forests. Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 1, 278–282. https://doi.org/10.1109/ICDAR.1995.598994

How Much Money Can You Earn With an App in 2019? | Fueled. (nd). Retrieved November 8, 2020, from https://fueled.com/blog/much-money-can-earn-app/