Journal of Numerical Optimization and Technology Management

Vol. 1, No. 2, 2023

htttps://shmpublisher.com/index.php/jnotm

p-ISSN: xxxx-xxxx
e-ISSN: xxxx-xxxx

.

Improvement accuracy of gradient boosting in app rating prediction on google playstore

Rofik[1], Dwika Ananda Agustina Pertiwi[2], Much Aziz Muslim[3]
[1, 3]Departement of Computer Science, FMIPA, Universitas Negeri Semarang
[2, 3]Faculty of Technology Management and Business, Universiti Tun Hussein Onn Malaysia, Batu Pahat, Malaysia

Article Info

Article history:

Received October 2023
Revised November 2023
Accepted November 2023

Keywords:

App rating
Gradient boosting
Google play store
Classification

ABSTRACT

Google Playstore is a platform that provides various useful applications for smartphone users, especially Android users. But in reality, users are often faced with many choices of applications with various features and functions in the Play Store itself. Rating applications on the Google Play store help users evaluate and choose applications that suit their needs. The purpose of this research is to optimize accuracy in predicting app ratings on the Google Play store using the Gradient Boosting algorithm. This research uses publicly accessible data on the Kaggle platform. The research process includes data collection, pre-processing, Data Spliting, algorithm modeling, and model evaluation. Apart from using the Gradient Boosting algorithm, this research also applies and optimizes other algorithms such as XGBoost, KNN, Logistic Regression, Decision Tree, Random Forest, LightGBM, AdaBoost, and SVM to predict app ratings on Google Playstore. By implementing and optimizing these algorithms, this study succeeded in achieving an accuracy of 92.62%, with MAE 0.311, RMSE 0.467, and R-square 0.144 using the Gradient Boosting algorithm. This research contributes to the development of better prediction methods in the mobile application industry and provides new insights regarding the factors that influence app ratings on the Google Play store.

Corresponding Author:

Rofik,
Departement of Computer Science,
Faculty of Mathematics and Natural Sciences,
Universitas Negeri Semarang, Indonesia.
Email: rofikn4291@students.unnes.ac.id
https://doi.org/10.00000/jnotm.0000.00.00.000

1. INTRODUCTION

Application or software is a form of technological work that can make everyone unable to turn their backs on it. In today's all-digital era, it is not uncommon for people to depend on their cellphone screens in their daily life. None other than because the smartphone can be a tool that can be relied upon in assisting various activities. Whether it's an application that can be accessed with the internet or not, they are always actively

working all the time. From various daily fields such as education, beauty, health, entertainment, economy, government, and others, it is often used in practice in the form of applications that can be accessed whenever and wherever humans are [1]. It cannot be denied that the existence of the internet which helps access online applications can improve the quality of human life, work that was previously done by humans can now be done by machines that can work better [2].

It was reported from the sensor tower that the number of application downloads increased in the 1st quarter of 2021 by 8.7%, of which as much of the increase was contributed by the Play Store's 24.4 billion downloads [2]. The various needs of every person in various parts of the world make the application market's needs even more increasing. In line with the ease of accessing and downloading existing applications from the Play Store users, application developers compete to make various kinds of applications [3]. This has an impact on the high level of competition in the business world, which makes every company able to survive in this world, one of which is the precise way of analyzing data as a decision-making process [4]. Likewise with users, who can easily choose and consider which applications they will download and use. However, the reviews and ratings listed on the Google Play store are very important [5]. This can be a recommendation tool between users and provide feedback for developers to report failures in applications, and it can also be used as a new feature proposal [6], [7]. Because it also happens that users who have selected millions of applications in the Play store and then downloaded them find that the application is not helpful and useful for them, because it is considered irrelevant to their needs [8], [9].

The problem of choosing and evaluating applications in the Play store is even easier in this era. Users can easily see the application rating that is displayed. Rating is an assessment given by previous users regarding the performance and quality of the application, which can be used as a benchmark for other users to download the application [4]. Ratings can make it easier for users to choose, but they can also be a challenge for developers and application companies. Because the performance of companies that produce application products can be known easily by anyone, this affects the success of the company. Therefore, developers and companies need to predict the rating of the product or software to prevent errors or deficiencies and be able to do development first. Before the application is well-known, used, and poorly rated by users, Predicting application ratings can be done by classifying them; classification is the process of identifying objects into categories, which in machine learning are known as classes, based on definitions, procedures, and provisions that have been made or selected [10].

Research on predicting application ratings using machine learning algorithms have previously been carried out by Dwika Ananda et al using the XGBoost algorithm with an accuracy of 77.5% [11]. Green Sandag also predicts application ratings using the Random Forest algorithm with an accuracy rate of 86.27% [12]. The research was also carried out by the author, by predicting the Google Play store application rating using several algorithms such as XGBoost, KNN, Logistic Regression, Decision Tree, Random Forest, LightGBM, Gradient Boosting, AdaBoost, and SVM. The research was conducted using publicly accessible data sourced from Kaggle. In this study, a comparison of the results of the accuracy produced by each algorithm was also carried out, which then obtained the algorithm that produced the greatest accuracy among the others.

2. METHOD

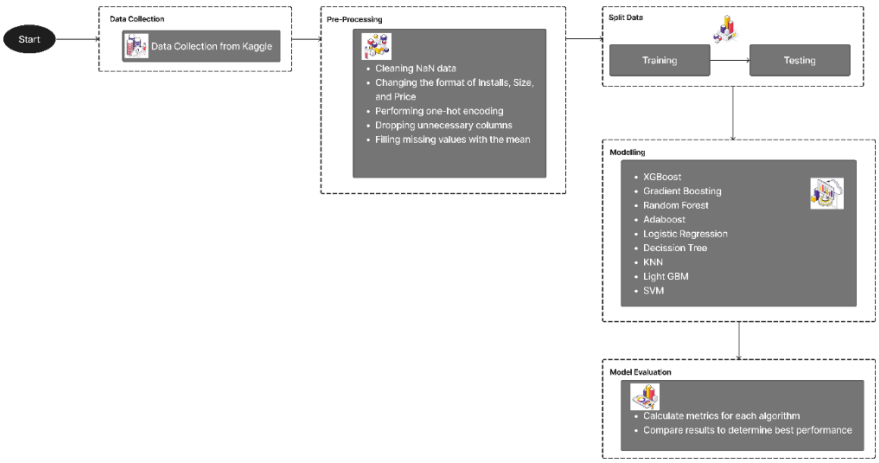

Figure 1. Research methods

The research method was carried out in stages, which began with collecting datasets, then continued with the data preprocessing stage, then modeling and evaluation. Dataset modeling is done using 9 algorithms, namely Decission Tree, Random Forest, XGBoost, KNN, Logistic Regression, Light GBM, Gradient Boosting, SVM, Adaboost. After modeling with various algorithms, a performance comparison process is carried out based on the level of accuracy obtained.

Data Collection

The data collection stage is carried out on the Kaggle platform which is publicly accessible. Data was taken from the Madhav000 account which was uploaded in 2020 and until this research, there were 4801 viewers. The data consists of 10840 rows, with a total of 13 attributes including rating, application name, application category, reviews, size, number of downloaders, payment type, price, content rating, genre, last update, latest version, and android version. The dataset used in the research can be accessed via the URL link: https://www.kaggle.com/code/madhav000/app-rating-prediction-in-playstore.

Pre-Processing Stage

At this stage, data cleaning is carried out, which prepares the dataset so that it is ready to be used in the modeling process by carrying out several steps and data transformation. Among them is deleting rows that contain missing values. After going through the data cleaning process, we managed to get the remaining 9366 data records. The missing value is when some of the values in the attributes in the data set contain empty data or have no value (NaN) [13]. In the missing value dataset, there are rating, type, and content rating features. Fixed structure errors including character conversions, such as the format in the column install, size, and price adjusted. The install column is changed by removing the '+' and commas, then the values are changed to integer data types. For the size feature, the 'M' and 'K' suffixes are removed, and if the size is in kilobytes, then the value is divided by 1024. The column in the price feature is also formatted into a numeric format by removing the dollar symbol. After cleaning the data, irrelevant columns such as app, last_ipdate, current ver, and android ver were deleted because they were deemed unnecessary in the modeling process [14]. To overcome missing values, the average value of each column is used to enter the blank values. Performing these pre-processing steps helps ensure the quality and integrity of the data used in the modeling and subsequent evaluation processes.

Data Spliting

The data spliting method is carried out after the process of cleaning and pre-processing the data in the previous stage. In this section, the 'train_test_split' function from the 'sklearn.model selection' library is used to divide the data into two sets, namely the training data set and the test data set. The distribution of this data is done randomly and carried out with a certain proportion to achieve optimal performance. It also aims to avoid bias in the evaluation process. In its application in predicting application ratings on Google Play Store 9, the algorithm is applied by dividing the data by 80% as training data and 20% as testing data.

Modeling

At this modeling stage, predictions of application ratings are made from the data that has been cleaned and divided into training data and testing data. The modeling stage is carried out using a tool in the form of

Google Colab Python. The genres, app-size, and review features are mandatory features used in modeling the algorithms used in this study because these features are the most influential and greatly impact application rankings [15]. Some explanations regarding the 9 algorithms used in predicting app ratings on the Google Play store can be seen below.

Classification of decision tree

The Decision Tree algorithm is an algorithm that builds a tree-like structure that separates data by dividing the data set sequentially into smaller subsets based on testing the data attributes [16]. This algorithm is a popular machine learning algorithm in classification and regression work. The structure of this algorithm is in the form of a binary tree starting from the "root" node which contains all the training data. The “root” node is divided into 2 child nodes using the predicted variable as the initial separator. Then the cleanest child or internal node among all child nodes is selected. The division step is repeated until a terminal node is obtained as the final result, to find the predicted class [17]. Each node in the decision tree represents a condition or rule that must be met, while the branches or leaves represent rating predictions.

Classification of random forest

Random Forest is an ensemble classification technique in machine learning that performs voting from several decision trees obtained from different bootstrap samples [18]. The ensemble technique means bagging and improving which means entering data into the basic model, then taking decision samples from each bootstrap. Then do the aggregation, namely making the majority decision on the test data [19]. Weighting is carried out, where a higher weight is given to the decision tree which results in a low error rate so that predictions are made with minimal error [18]. An easy explanation is, Random Forest is an algorithm that works by building a large number of decision trees together during its operation [20]. Decision-making or classification results are based on the most votes from the results of each tree. But it can also happen that several trees that are built may produce the same correlation which leads to errors. And each tree that is built has its strengths, where a tree that has a low error rate is a strong classification and vice versa. This algorithm can overcome the problems of non-linearity and complexity that often occur in application rating predictions.

Classification of XGBoost

XGBoost or what can be called Extreme Gradient Boosting is a machine learning algorithm that has been busy in recent years because of its high speed and accuracy in modeling, it performs the formation and addition of sequential decision trees to correct errors in previous models [21]. XGBoost improves gradient trees with several extensions, one of which is sparsity awareness which can deal with missing value problems [22]. In the context of this research, XGBoost can take into account the interaction between variables in the application dataset, thus providing accurate prediction accuracy.

Classification of KNN

The KNN algorithm is an algorithm that can be considered important and easy, this algorithm can recognize patterns. The way this algorithm works is if a predicted point is in the same position or can be said to be adjacent to the point in the training set, then it can be concluded that this point also has the same characteristics and the predicted results are also not much different from the point in the training set [23]. When there is a new application whose rating is predicted, KNN looks for K applications that are closest to it based on the distance between the application's features and existing applications. This allows us to predict the rating of new apps based on the user's experience and preferences for similar apps. By using the KNN algorithm we can get an advantage, which in modeling this algorithm does not require knowledge and maintains a given model which can have an impact on adjustments with rapid changes [24].

Classification of logistic regression

Logistic Regression (LR) is an extension of linear regression which is useful for detecting the relationship between one or more independent variables with the dependent variable which is binary or multi categorical [25]. Logistic Regression can be used when encountered in a non-linear relationship problem [26]. In this case, Logistic Regression will determine the regression coefficient for each app feature, which indicates how much influence each feature has on the rating prediction. By understanding the patterns and relationships between features and application ratings on the training data, the logistic regression model can provide

estimates of the probability of a high rating or a low rating based on a combination of existing features. Thus, it can assist in making strategic decisions regarding development, marketing, and improving the quality of applications to achieve a better rating on the platform.

Classification of lightgbm

The Light GBM algorithm is a decision tree algorithm with a new gradient enhancement that also uses GOSS and EFB [27]. GOSS is used to minimize the number of calculations and maximize the speed and storage of all data that has large gradients and data that have small gradients. Meanwhile, EFB is used to reduce feature dimensions by combining special features into one. This algorithm was chosen for modeling because it has high training speed, unquestionable efficiency, good accuracy, low memory usage, the ability to process large-scale data, and welcomes parallel and distributed learning.

Classification of gradient boosting

With the cleaned dataset, modeling is carried out using the Gradient Boosting algorithm. The Gradient Boosting Algorithm is a type of ensemble classification algorithm that is built sequentially so that it can repeatedly reduce errors in the previous model [28]. Gradient Boosting can provide high accuracy in predicting application ratings because it takes into account the interaction between the variables in the dataset and places an emphasis on inaccurate data processing.

Classification of support vector machine

Support vector machine (SVM) is the most commonly used machine learning algorithm in document classification and sentiment classification work [28]. In general, the way SVM works is by forming one or several hyperplanes in high or infinite dimensional space. In this case, SVM can find the optimal dividing line between positive and negative ratings or even model more than two rating classes. SVM performs linear classification or non-linear classification using the kernel method by placing input into a high-dimensional feature space [29]. However, in this algorithm there are difficulties if you have to classify unbalanced power, this is related to determining the optimal hyperplane which will be difficult for the unbalanced data [30]. In addition, in predicting application ratings on the Google Play store, SVM can utilize application features such as category, type, size, and price to build a model that can predict ratings with high accuracy.

Classification of decition ADABoost

Adaboost is an algorithm introduced by Freeud and Schapire in 1995, this algorithm is used to increase the accuracy of prediction rules by combining many inaccurate and weak rules [31]. Adaboost is used to obtain strong classifiers by adjusting for errors between weak classifiers. Iteratively and gradually updated the weights for each classification to obtain stronger results [32]. One of Adaboost's strengths is its ability to handle unbalanced data. In the case of this application rating prediction, there may be an imbalance in the number of applications with high and low ratings. Apart from that, Adaboost is also able to perform feature selection automatically, this is very helpful in identifying the most relevant features in predicting application ratings. By looking at the capabilities and advantages, the Adaboost algorithm participates in the modeling.

3. RESULTS AND DISCUSSIONS

The process of increasing application rating predictions on the Google Playstore begins by using data sourced from Kaggle with a total of 10,840 records with 13 features. The data is modeled using the help of a tool in the form of a colab. The 13 features in the dataset include App_Name, Category, Rating, Reviews, Size, Install, Type, Price, Content_Rating, Genres, Last_Updates, Current_Version, and Android_Version. Because the data varies greatly in terms of writing and is not clean, it is very difficult and will affect performance in modeling, pre-processing is carried out. In the pre-processing stage, rows containing missing values are deleted. To be able to model features that contain non-uniform data types, character conversions are carried out as found in the rating, type, and content rating features. The size feature that ends in 'M' and 'K' is also deleted, but it is divided by 1024 if the size is in kilobytes. The price feature has also been changed to a numeric format by removing the dollar symbol. To overcome missing values, the average value of each column is used to enter the blank value. Also deleted irrelevant columns such as app, last_update, current_ver, and android ver. After the pre-processing stage, 9366 records are ready to be used in modeling. Modeling uses a ratio of 80% and 20% with details of 7492 records as training data and 1874 as testing data.

This study uses 9 machine learning algorithms in modeling the Google Play store app rating prediction. These algorithms include Decision Tree, Random Forest, XGBoost, KNN, Logistic Regression, Light GBM, Gradient Boosting, SVM, and Adaboost. Modeling was carried out with these algorithms to find

1 algorithm that provides the best performance in predicting application ratings on the Google Play store from the accuracy obtained with the same dataset. This study found that the prediction performance of application ratings on the Google Play store using the Gradient Boosting algorithm produces the greatest accuracy. The performance of these algorithms and the accuracy that is less than Gradient Boosting is used as a basis for strengthening the research results of the algorithm which shows the greatest accuracy with the Gradient Boosting. The performance of each algorithm can be seen from the resulting accuracy, which can be seen in the Figure 2.

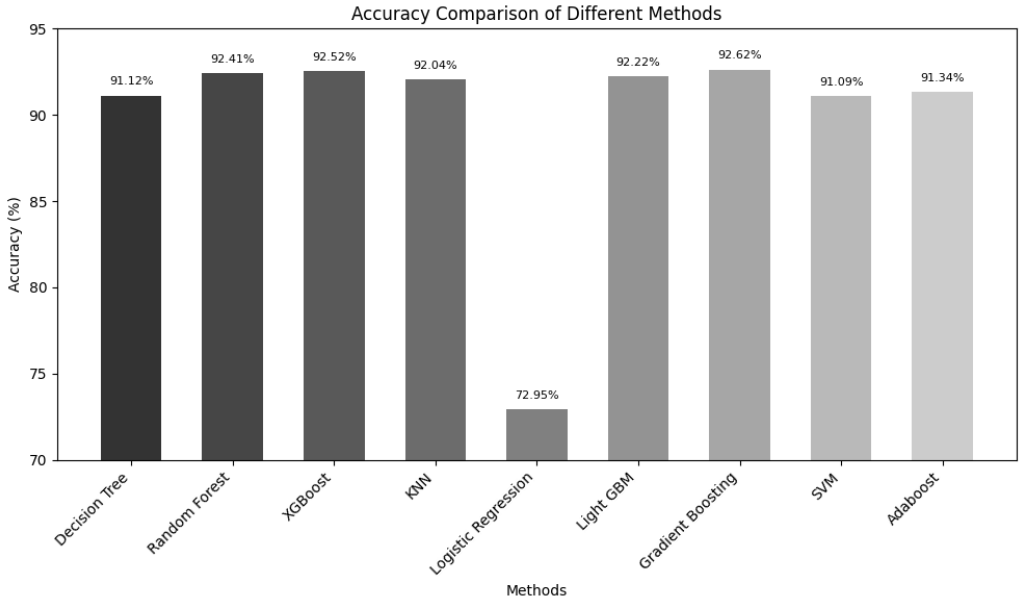

Figure 2. Accuracy comparison of 9 modeling algorithms

Figure 2 shows that performance of the Gradient Boosting algorithm outperforms the performance of other algorithms, which is 92.62% in the case of predicting app ratings on Google Playstore. And this shows that this research has also succeeded in increasing the accuracy of app rating predictions on the Google Play store with previous research using the XGBoost method with the greatest accuracy rate of 77.5% [12].

Evaluation of metrics is carried out to measure the quality and performance of the models that have been built. In this study, the evaluation metric used is MAE (Mean Absolute Error), to measure the average error between the class's actual grade and the predicted value. So the lower the MAE value, the better the model is at making predictions. RMSE (Root Mean Squared Error), to measure the average error in terms of the square root of the difference between the true value and the predicted value. This RMSE gives greater weight to large errors so that it can describe the overall level of error variability. Same with MAE, the lower the value, the more accurate the model is in predicting. And the last one is R-square. In the case of app rating predictions on the Google Play store, r-square indicates the extent to which the model can explain variations in in-app ratings based on the features present in the dataset used. Detailed information regarding the metric evaluation performed on the 9 algorithms can be seen in the Table 1.

Table 1. Evaluation of metrics of 9 model algorithms

Metode	MAE	RMSE	R-Square
Decision Tree	0.213	0.477	0.079
Random Forest	0.319	0.465	0.121
XGBoost	0.135	0.477	0.079
KNN	0.335	0.474	0.090
Logistic Regression	0.213	0.462	-0.283
Light GBM	0.327	0.493	0.013
Gradient Boosting	0.311	0.467	0.114

SVM	0.375	0.574	-0.336
Adaboost	0.364	0.496	0.002

Based on the evaluation of the metrics that have been carried out, it was found that Logistic Regression has good performance with the lowest MAE and low RMSE, this is in contrast to the accuracy it has obtained. In this case, logistic regression produces predictions that are biased towards the majority class or have a uniform level ofs imprecision across the application's rating range. However, the evaluation of metrics in the ability to explain variations in application ratings, Gradient Boosting, and Random Forest shows better results with superior R-square values. This shows that these models have a better ability to explain variations in application ratings. With a good combination of MAE, RMSE, and R-square, the Gradient Boosting algorithm is superior in predicting app ratings on Google Playstore with high curation. Because of its ability to deal with complex data, understand complex patterns, and iterative adjustments made, it is an effective choice of algorithms for increasing prediction accuracy.

4. CONCLUSION

Predicting application ratings is an important thing to do to maintain the company's image. Machine learning provides several solutions that can be used to overcome this problem. This study uses several machine learning algorithms in the form of predicting app ratings on the Google Play store, using the Kaggle data source. The machine learning model with the Gradient Boosting algorithm is applied to 9,366 data lines and 13 features, resulting in the highest accuracy rate of 92.62% with MAE: 0.311, RMSE: 0.467, and R-square: 0.114. The application of this machine learning method is effective in predicting application ratings with a high degree of accuracy, helping companies improve the quality and performance of their applications. In addition, application users can also be helped by more accurate predictions of application ratings, by helping them choose applications that suit their needs and preferences.

REFERENCES

[1] R. Adyatma Subagja, Y. Widiastiwi, and N. Chamidah, "Klasifikasi Ulasan Aplikasi Jenius pada Google Play Store Menggunakan Algoritma Naive Bayes," *Inform. J. Ilmu Komput.*, vol. 17, no. 3, p. 197, 2021, doi: 10.52958/iftk.v17i3.3652.

[2] A. Lidwina, "Jumlah Unduhan Aplikasi Global Naik 8,7% pada Kuartal I-2021," *databoks*, 2021.

[3] A. Chandra Saputra, P. Raya Jln Hendrik Timang, P. Raya, P. Studi Teknik Informatika, F. Teknik, and U. Palangka Raya Jln Hendrik Timang, "KLASIFIKASI RATING APLIKASI ANDROID DI GOOGLE PLAY STORE MENGGUNAKAN ALGORITMA GRADIENT BOOST Agus Sehatman Saragih," *Oktober*, vol. 6, no. 1, pp. 18–29, 2022.

[4] A. A. Nurdin, G. N. Salmi, K. Sentosa, A. R. Wijayanti, and A. Prasetya, "Utilization of Business Intelligence in Sales Information Systems," *J. Inf. Syst. Explor. Res.*, vol. 1, no. 1, pp. 39–48, Dec. 2022, doi: 10.52465/joiser.v1i1.101.

[5] M. Nilashi *et al.*, "The impact of multi-criteria ratings in social networking sites on the performance of online recommendation agents," *Telemat. Informatics*, vol. 76, no. 1, p. 101919, 2023, doi: 10.1016/j.tele.2022.101919.

[6] P. Priyanga and A. R. Nadira Banu Kamal, "Mobile App Usage Pattern Prediction Using Hierarchical Flexi-Ensemble Clustering (HFEC) for Mobile Service Rating," *Wirel. Pers. Commun.*, vol. 122, no. 4, pp. 3247–3268, 2022, doi: 10.1007/s11277-021-09048-0.

[7] A. A. Qureshi, M. Ahmad, S. Ullah, M. N. Yasir, F. Rustam, and I. Ashraf, "Performance evaluation of machine learning models on large dataset of android applications reviews," *Multimed. Tools Appl.*, 2023, doi: 10.1007/s11042-023-14713-6.

[8] S. Sadiq, M. Umer, S. Ullah, S. Mirjalili, V. Rupapara, and M. Nappi, "Discrepancy detection between actual user reviews and numeric ratings of Google App store using deep learning," *Expert Syst. Appl.*, vol. 181, no. April, p. 115111, 2021, doi: 10.1016/j.eswa.2021.115111.

[9] M. R. Hamednai, G. Kim, and S. je Cho, "SimAndro: an effective method to compute similarity of Android applications," *Soft Comput.*, vol. 23, no. 17, pp. 7569–7590, 2019, doi: 10.1007/s00500-019-03755-4.

[10] O. Arifin, K. Saputra, and H. Fathoni, "Implementation of Data Mining using Naïve Bayes Classifier Method in Food Crop Prediction," *Sci. J. Informatics*, vol. 8, no. 1, pp. 43–50, 2021, doi: 10.15294/sji.v8i1.28354.

[11] D. A. A. Pertiwi, T. Mustaqim, and M. A. Muslim, "Prediksi Rating Aplikasi Playstore Menggunakan Xgboost," *Proc. SNIK*, no. November, 2020.

[12] G. A. Sandag, "Prediksi Rating Aplikasi App Store Menggunakan Algoritma Random Forest," *CogITo Smart J.*, vol. 6, no. 2, pp. 167–178, 2020, doi: 10.31154/cogito.v6i2.270.167-178.

[13] Y. Yustikasari, H. Mubarok, and R. Rianto, "Comparative Analysis Performance of K-Nearest Neighbor Algorithm and Adaptive Boosting on the Prediction of Non-Cash Food Aid Recipients," *Sci. J. Informatics*, vol. 9, no. 2, pp. 205–217, 2022, doi: 10.15294/sji.v9i2.32369.

[14] A. R. Safitri and M. A. Muslim, "Improved Accuracy of Naive Bayes Classifier for Determination of Customer Churn Uses SMOTE and Genetic Algorithms," *J. Soft Comput. Explor.*, vol. 1, no. 1, Sep. 2020, doi: 10.52465/joscex.v1i1.5.

[15] A. Mahmood, "Identifying the influence of various factor of apps on google play apps ratings," *J. Data, Inf. Manag.*, vol. 2, no. 1, pp. 15–23, 2020, doi: 10.1007/s42488-019-00015-w.

[16] A. Benghanem, J. Audet, and A. Lussier-Desbiens, "Alpine skis categorization with comprehensive decision tree models," *Sport. Eng.*, vol. 26, no. 1, 2023, doi: 10.1007/s12283-023-00405-9.

[17] S. Banerjee and P. S. Bhowmik, "A machine learning approach based on decision tree algorithm for classification of transient events in microgrid," *Electr. Eng.*, 2023, doi: 10.1007/s00202-023-01796-5.

[18] N. Aslam *et al.*, "Improving the review classification of Google apps using combined feature embedding and deep convolutional neural network model," *J. Ambient Intell. Humaniz. Comput.*, vol. 14, no. 4, pp. 4257–4272, 2023, doi: 10.1007/s12652-023-04529-5.

[19] G. Chaubey, P. R. Gavhane, D. Bisen, and S. K. Arjaria, "Customer purchasing behavior prediction using machine learning classification techniques," *J. Ambient Intell. Humaniz. Comput.*, no. 0123456789, 2022, doi: 10.1007/s12652-022-03837-6.

[20] C. Mason, J. Twomey, D. Wright, and L. Whitman, "Predicting Engineering Student Attrition Risk Using a Probabilistic Neural Network and Comparing Results with a Backpropagation Neural Network and Logistic Regression," *Res. High. Educ.*, vol. 59, no. 3, pp. 382–400, 2018, doi: 10.1007/s11162-017-9473-z.

[21] P. Carmona, A. Dwekat, and Z. Mardawi, "No more black boxes! Explaining the predictions of a machine learning XGBoost classifier algorithm in business failure," *Res. Int. Bus. Financ.*, vol. 61, no. November 2020, p. 101649, 2022, doi: 10.1016/j.ribaf.2022.101649.

[22] D. A. Rusdah and H. Murfi, "XGBoost in handling missing values for life insurance risk prediction," *SN Appl. Sci.*, vol. 2, no. 8, pp. 1–10, 2020, doi: 10.1007/s42452-020-3128-y.

[23] S. Gündoğdu, "Efficient prediction of early-stage diabetes using XGBoost classifier with random forest feature selection technique," *Multimed. Tools Appl.*, 2023, doi: 10.1007/s11042-023-15165-8.

[24] D. M. Atallah, M. Badawy, A. El-Sayed, and M. A. Ghoneim, "Predicting kidney transplantation outcome based on hybrid feature selection and KNN classifier," *Multimed. Tools Appl.*, vol. 78, no. 14, pp. 20383–20407, 2019, doi: 10.1007/s11042-019-7370-5.

[25] Q. Gu, W. Sun, X. Li, S. Jiang, and J. Tian, "A new ensemble classification approach based on Rotation Forest and LightGBM," *Neural Comput. Appl.*, vol. 3, 2023, doi: 10.1007/s00521-023-08297-3.

[26] Y. Kang, M. G. Kim, and K. M. Lim, "Machine-learning based prediction models for assessing skin irritation and corrosion potential of liquid chemicals using physicochemical properties by XGBoost," *Toxicol. Res.*, vol. 39, no. 2, pp. 295–305, 2023, doi: 10.1007/s43188-022-00168-8.

[27] C. Zheng *et al.*, "Time-to-event prediction analysis of patients with chronic heart failure comorbid with atrial fibrillation: a LightGBM model," *BMC Cardiovasc. Disord.*, vol. 21, no. 1, pp. 1–12, 2021, doi: 10.1186/s12872-021-02188-y.

[28] X. Pu, G. Wu, and C. Yuan, "Exploring overall opinions for document level sentiment classification with structural SVM," *Multimed. Syst.*, vol. 25, no. 1, pp. 21–33, 2019, doi: 10.1007/s00530-017-0550-0.

[29] A. P. Gopi, R. N. S. Jyothi, V. L. Narayana, and K. S. Sandeep, "Classification of tweets data based on polarity using improved RBF kernel of SVM," *Int. J. Inf. Technol.*, vol. 15, no. 2, pp. 965–980, 2020, doi: 10.1007/s41870-019-00409-4.

[30] J. Grzyb and M. Woźniak, "SVM ensemble training for imbalanced data classification using multi-objective optimization techniques," *Appl. Intell.*, 2022, doi: 10.1007/s10489-022-04291-9.

[31] K. Shah, H. Patel, D. Sanghvi, and M. Shah, "A Comparative Analysis of Logistic Regression, Random Forest and KNN Models for the Text Classification," *Augment. Hum. Res.*, vol. 5, no. 1, p. 12, Dec. 2020, doi: 10.1007/s41133-020-00032-0.

[32] A. Lestari, "Increasing Accuracy of C4.5 Algorithm Using Information Gain Ratio and Adaboost for Classification of Chronic Kidney Disease," *J. Soft Comput. Explor.*, vol. 1, no. 1, pp. 32–38, 2020, doi: 10.52465/joscex.v1i1.6.

.