Klasifikasi Peringkat Aplikasi Android di Google Play Store Menggunakan

Algoritma Gradient Boost

Apps Rating Classification on Play Store Using Gradient Boost Algorithm

**1Rodney Maringka**

1Program Studi Sistem Informasi, Universitas Klabat, Airmadidi

e-mail: 1s21610035@student.unklab.ac.id

***Abstrak***

*Semakin banyaknya aplikasi Android yang tersedia di Google Play Store dengan keuntungan yang didapatkan pengembangnya telah menarik perhatian banyak pengembang aplikasi Android. Untuk mendapatkan keuntungan dari mengembangkan aplikasi Android, salah satu caranya adalah dengan mengetahui karakteristik aplikasi berperingkat tinggi di Google Play Store. Penelitian ini akan menyelidiki fitur size, installs, reviews, type(gratis/bayar), rating, category, content rating, dan price pada aplikasi di Google Play Store untuk mengetahui karakteristik aplikasi berperingkat tinggi. Penelitian ini menggunakan algoritma Gradient Boost untuk mengidentifikasi fitur yang paling berpengaruh pada aplikasi dengan peringkat tinggi di Google Play Store. Pada tahap preprocessing, penelitian ini menggunakan metode data cleaning dan data reduction. Penelitian ini menggunakan feature important untuk mengetahui atribut yang paling berpengaruh pada peringkat tinggi aplikasi Android di Google Play Store. Untuk mengklasifikasi aplikasi berperingkat tinggi penulis menggunakan algoritma Gradient Boost dengan hasil yang lebih baik dibandingkan dengan algoritma Random Forest, K-NN, dan Decision Tree yaitu 99.93% accuracy, 99.91% recall, 99.91% precision, dan 0.062 Root Mean Squared Error.*

***Kata kunci* : Aplikasi Android, Google Play Store, Karateristik, Peringkat tinggi, Random-Forest Classifier**

***Abstract***

*The increasing number of Android apps available on Google Play Store with the advantages that the developers got has attracted the attention of many Android apps developer. To get the benefit from developing an Android apps, one way is to know the characteristics of high rated applications on Google Play Store. This research will explore the features of size, installs, reviews, types (free/paid), rating, category, content rating, and price on apps on Google Play Store to find out the characteristics of high rated apps. This research uses random-forest classifier to identify the most significant features in high rated apps on Google Play Store. This research use Gradient Boost Algorithm to identificate the most influential attributes in high rating apps on Google Play Store. To classify the high rated apps writers use Gradient Boost algorithm that perfoms better than Random Forest, K-NN, and Decision Tree algorithm with a result of 99.93% accuracy, 99.91% recall, 99.91% precision, and 0.062 Root Mean Squared Error.*

***Keywords*** : **Android Apps, Google Play Store, Characteristics, High Rated, Random-Forest Classifier**

1. PENDAHULUAN

Saat ini jumlah aplikasi *Android* dengan penggunanya berkembang dengan cepat mempengaruhi perekonomian berkembang pesat [1]. Menurut laporan tahun 2018, nilai ekonomi aplikasi global adalah *USD* 81.7 milyar di tahun 2017, dan naik menjadi *USD* 106.4 milyar pada tahun 2018, dan diperkirakan akan naik menjadi *USD* 156.5pada tahun 2022 [2]. Pasar aplikasi *mobile phone* yang berkembang dengan cepat menarik perhatian banyak pengembang dengan lebih dari 11 juta pengembang yang sudah memasukkan aplikasi mereka di *Google Play Store* dan *App Store* berdasarkan laporan tahun 2016 [3]. Berdasarkan data dari artikel *fueled*, dilaporkan rata-rata pendapatan per pengembang aplikasi *Android* di *Google* adalah sebesar *USD* 97,600 per tahun [4]. Banyak pengembang aplikasi *Android* masih berjuang mencari cara untuk mendapat keuntungan dari aplikasi yang mereka buat dengan memasukkannya di *Play* *Store*. Dengan mengetahui karakteristik dari aplikasi yang sudah berhasil dapat menjadi salah satu cara bagi pengembang untuk membuat aplikasi yang berhasil.

Berbagai karakteristik dapat dijadikan sebagai patokan untuk dapat membuat aplikasi yang berhasil. Sama seperti penelitian sebelumnya [5], dalam penelitian ini penulis memilih penilaian peringkat sebagai patokan untuk aplikasi yang berhasil, oleh karena itu faktor atau fitur yang mempengaruhi keberhasilan aplikasi yaitu karakteristik aplikasi berperingkat tinggi perlu diketahui.

Untuk mengklasifikasi aplikasi yang berhasil dapat dilihat dari jumlah unduhan, tetapi banyak pengguna mengunduh aplikasi tanpa menggunakannya. *Google* *Play* *Store* menampilkan jumlah unduhan berdasarkan rentang, bukan nilai yang sebenarnya (contohnya: 10M+ *Downloads*) yang akan mempersulit untuk membedakan aplikasi yang berada pada rentang yang sama. Cara lain untuk mengidentifikasi aplikasi yang berhasil adalah dengan menganalisa komentar yang dimasukkan pengguna saat memberikan peringkat, tetapi banyak pengguna memberikan peringkat tanpa memasukkan komentar.

Dalam penelitian sebelumnya tentang analisa faktor yang mempengaruhi peringkat aplikasi, didapatkan aplikasi dengan peringkat tinggi mempunyai nilai perbaikan *bug API* lebih rendah dibandingkan dengan aplikasi dengan peringkat rendah setelah data diuji dengan *Mann-Whitney test* (*p-value* < 0.0001) dan *cliff’s Delta* (0.37). Menunjukkan bahwa ada hubungan antara peringkat aplikasi dengan faktor yang lain seperti perubahan *Android* *APIs* dan kompleksitas dari *user interface* [5].

Kemudian pada penelitian yang lain tentang pengaruh iklan (*Ad Libraries*) terhadap peringkat *Android*, setelah data diuji dengan *spearman rank correlation* antara *ad libraries* dalam suatu *app*  dengan peringkat *app* menghasilkan nilai 0.016 (*weak correlation*). didapatkan bahwa semakin banyak iklan yang dimasukkan pengembang terhadap aplikasinya ternyata tidak terlalu berpengaruh terhadap peringkat *Android*  [6]. Penelitian sebelumnya oleh Aralikatte (2018) yaitu menguji korelasi antara peringkat aplikasi dengan rata-rata nilai sentiment (+1=korelasi positif, 0=tidak ada korelasi, dan -1=total korelasi negatif) menggunakan *Pearson* dan *Spearman correlation* menghasilkan nilai sebesar 0.5 untuk setiap korelasi, yang menyatakan bahwa terdapat korelasi meskipun tidak besar [7].

Penelitian sebelumnya oleh Harman et al. (2012) yaitu menguji korelasi antara harga, peringkat, dan unduhan dari *Blackberry App Store*, dengan menggunakan *Spearman Correlation* didapatkan bahwa ada korelasi yang kuat antara peringkat dan unduhan yaitu sebesar 0.79, dan nilai korelasi yang rendah yaitu 0.12 antara harga dan unduhan [8]. Tujuan dari penelitian ini adalah untuk memeriksa faktor atau fitur lain yang berhubungan dengan penilaian peringkat aplikasi dan meneliti fitur yang paling berpengaruh untuk mengidentifikasi aplikasi dengan peringkat tinggi.

Fitur yang diteliti adalah *size*, *installs*, *reviews*, *types (free/paid*), *rating*, *category*,dan *price* sesuai dengan *dataset* dari *kaggle* [9]. Dalam penelitian ini, penulis menggunakan *gradient boost* untuk menyelidiki fitur penting yang berpengaruh terhadap penilaian peringkat aplikasi. Penulis memilih *gradient boost* karena metode klasifikasi ini mampu mengklasifikasi data yang tidak seimbang dan digunakan untuk menangani data yang banyak. Manfaat dari penelitian ini adalah kedepannya mampu memberikan informasi yang berguna bagi pengembang aplikasi dan pengguna untuk mengetahui karakteristik aplikasi *Android* berperingkat tinggi.

2. METODE PENELITIAN

*2.1 High Rated and Low Rated Apps on Google Play Store*

Data yang digunakan pada penelitian ini adalah *Google Play Store dataset* yang dapat di akses di *Kaggle* [9]. *Dataset* ini memiliki 10842 *row*, dan 13 *attributes* yang dijelaskan di Tabel 1**.**

Tabel 1. Atribut Dataset

|  |  |
| --- | --- |
| **Atribut** | **Detail** |
| App | Nama aplikasi |
| Category | Jenis category dari aplikasi |
| Rating | Seberapa besar rating dari aplikasi |
| Review | Seberapa banyak review dari user |
| Size | Ukuran dari aplikasi |
| Installs | Jumlah user yang menginstall aplikasi |
| Type | Tipe dari aplikasi tersebut |
| Price | Harga aplikasi |
| Content Rating | Untuk siapa aplikasi ini dibuat |
| Genres | Jenis yang lebih spesifik dari aplikasi |
| Last Updates | Kapan terakhir aplikasi di update |
| Current Ver | Versi terbaru dari aplikasi |
| Android Ver | Versi Android yang bisa menggunakan aplikasi |

*2.2 Desain Penelitian*

Gambar 1 memperlihatkan proses klasifikasi peringkat aplikasi *Android* di *Google Play Store*. Proses pertama adalah mengambil *Google Play Store dataset* yang diambil dari *Kaggle*, dilanjutkan *data preprocessing* untuk mengolah data. Dataset dibagi menjadi 80% *training data* yang terdiri atas 8671 data yang terbagi atas 7956 *high rated* dan 715 *low rated data*, dan 20% *testing data* yang terdiri atas 2167 data yang terbagi atas 1988 *high rated* dan 179 *low rated data*. Penelitian ini menggunakan *10*-*fold cross validation* untuk membagi data menjadi 10 bagian dan diuji sebanyak 10 kali sebelum melakukan *modelling*. Selanjutnya data di proses menggunakan algoritma *random forest*, yang kemudian akan dibuat modelnya dan dievaluasi.



Gambar 1. Arsitektur Untuk Klasifikasi Peringkat Aplikasi Android

*2.2.1 Data Preprocessing*

Data *preprocessing* dibagi menjadi 2 bagian, yaitu: *data cleaning* dan *data reduction*. *Data cleaning* adalah proses pembersihan data *incomplete* pada *attribute* di *dataset* untuk membuat data menjadi lebih konsisten. Sedangkan, *data reduction* adalah proses untuk menghapus data pada *attribute* yang kurang dominan sehingga data bisa dikurangi, namun tetap menghasilkan data yang akurat. Dalam proses *data cleaning* penulis mengklasifikasi dan menetapkan label data peringkat menjadi high rated (> 3.5) dan low rated (≤ 3.5), menghilangkan symbol k dan m pada kolom *size,* menghilangkan symbol + pada kolom *installs* dan dalam proses data reduction penulis menghapus data yang ada pada atribut *current version*, *android version*, *genre*, dan *last updated*.

*2.2.2. 10-Fold Cross Validation*

Setelah data telah dibagi menjadi 80% *data training* dan 20% *data testing,* maka akan dilakukan 10-*fold cross validation* pada *data training*. *Cross Validation* adalah teknik untuk mengevaluasi model dengan cara mempartisi sampel asli ke dalam *training set* untuk melatih model, dan *test set* untuk mengevaluasi model. Dalam *k-fold cross validation*, sampel asli secara acak dipartisi dalam *k equal size subsample*. Dari *subsample* k, satu *subsample* akan digunakan sebagai *testing data* dan sisanya akan menjadi *training data*. Proses *cross validation* akan diulang sebanyak k kali (kelipatan), dengan masing – masing dari *subsample* k digunakan sekali sebagai *validation data* [10]. Pada Gambar 3 menunjukkan proses 10-*fold cross validation*, data dibagi menjadi 10 partisi dan akan diuji sebanyak 10 kali sebelum dibuat modelnya.



Gambar 2. 10 Fold Cross Validation

*2.2.3. Gradient Boost*

*Gradient* *bossting*, seperti halnya keluarga algoritma *Boosted* lainnya memiliki kemampuan untuk meningkatkan akurasi prediktif model. Beberapa algoritma *boosting* lainnya seperti: *XGBoost*, *AdaBoost* dan *GentleBoost* memiliki formula matematika tersendiri dan bervariasi. Konsep *Gradient* *Boosting* terletak pada pengengembangannya yang mana memiliki ekspansi tambahan terhadap *fitting* *criterion* [11]. Berawal dari metode *Bagging* yaitu mengambil sampel data secara acak, bangun algoritma dan hitung rata-rata segala kemungkinan yang terjadi termasuk error dan akurasi. Daripada mengambil secara acak, pemilihan sample dapat dilakukan secara lebih cerdas dengan menggunakan fungsi *Boost*. Misalkan diketahui sebuah model sebagai M dengana akurasi sebesar 82%. Meningkatkan akurasi dapat dilakukan dengan cara membangun ulang model secara keseluruhan menggunakan input variabel yang baru dan bangun ulang model. *Gradient* *Boost* melakukan ekspansi

terhadap model matematika berikut [12]:

………………………....(1)

*2.2.4. Performance Evaluation*

Setelah pembuatan model maka langkah selanjutnya adalah melakukan evaluasi dengan *performance evaluation*. *Performance evaluation* berguna untuk menguji performa dari *classifier*. *Recall*, *precision*, dan *accuracy*. *Recall* adalah kumpulan data positif yang diklasifikasikan dengan benar sebagai data positif. *Precision* adalah kumpulan data yang diklasifikasikan sebagai positif yang benar – benar positif. *Accuracy* adalah ketepatan klasifikasi data [13].

Berikut ini adalah rumus *recall*, *precision*, dan *accuracy* dalam *performance evaluation*:

Rumus *Recall*: *Recal* = (TP ) / (TP+FN) , ………………………………….(2)

Rumus *Precision*: *Precision* = (TP +TN) / (TP+TN+FP+FN) , ……………….(3)

Rumus *Accuracy*: *Accuracy* = (TP +TN) / (TP+TN+FP+FN) , ……………….(4)

Keterangan:

TP : Nilai *true positive* TN : Nilai *true negative*

P : Jumlah data *positive* FP : Nilai *false* *positive*

N : Jumlah data *negative* FN : Nilai *false* negative

2.3 *Feature Important*

Metode *feature* *important* memegang peran penting dalam memilih attribute yang siginifikan, melalui penghapusan attribute yang tidak relevan, dan oleh karena itu dapat digunakan untuk identifikasi attribute yang berpengaruh [14]. Penulis menggunakan metode *information gain ratio* untuk menentukan berapa besar pengaruh suatu atribut dalam dataset. *Machine learning information gain* dapat digunakan untuk membuat peringkat dari atribut-atribut yang memiliki *information gain* yang tinggi harus diberi peringkat lebih tinggi daripada attributes yang lain karena lebih berpengaruh dalam mengklasifikasikan data [14]. Berikut ini adalah rumus dalam *information gain*:

, …………………………….….(5)

Keterangan:

H(S) : Entropi dari dataset

H(Si) : Entropi dari i subset yang dihasilkan oleh partisi S

A : Atribut dalam dataset

2.4. *Root Mean Squared Error*

Root Mean Squared Error (RMSE) adalah standar deviasi dari residual (kesalahan prediksi). Residual adalah ukuran seberapa jauh jarak dari titik garis regresi; RMSE adalah ukuran bagaimana residual ini tersebar [15]. Berikut ini adalah rumus RMSE:

, .....………………………………………..(6)

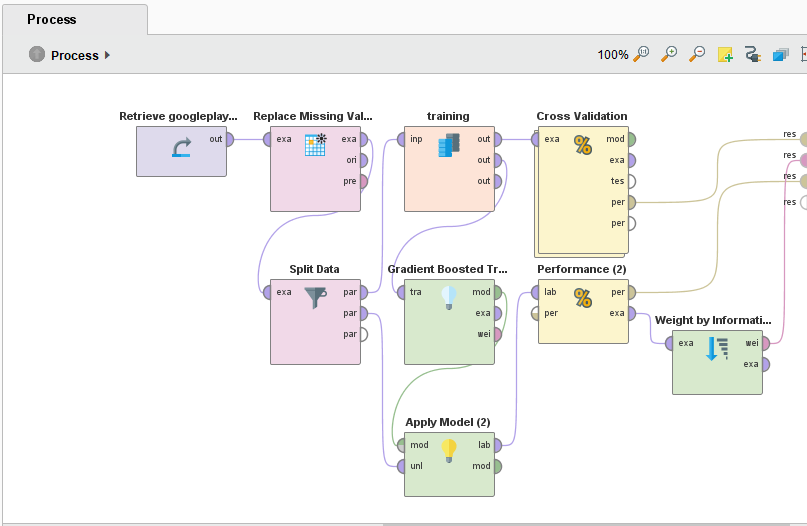
Keterangan:

f : Perkiraan (nilai yang diharapkan atau hasil yang tidak diketahui)

o : Nilai yang diamati (hasil yang diketahui­­­)

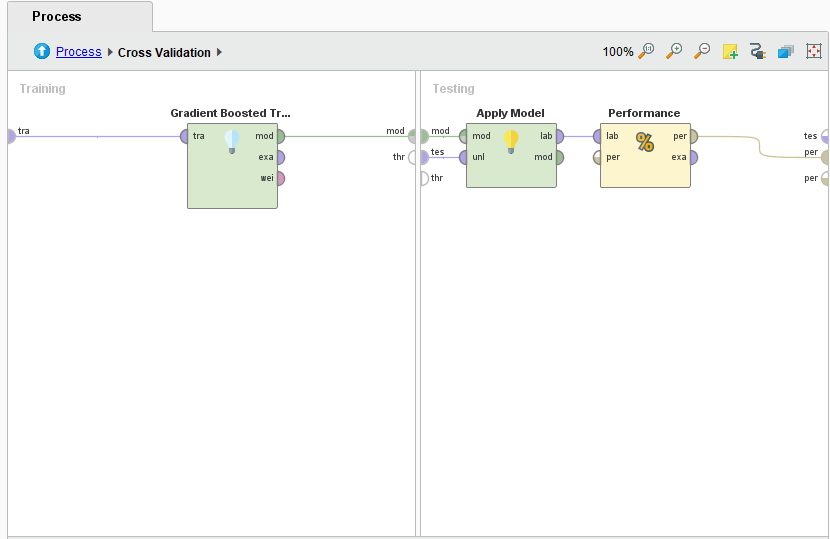
3. HASIL DAN PEMBAHASAN

Pada bagian ini penulis melakukan analisa terhadap *Google Play Store dataset* menggunakan algoritma *random forest*

*3.1 Process Gradient Boost*

Gambar 3.1 Process Gradient Boost

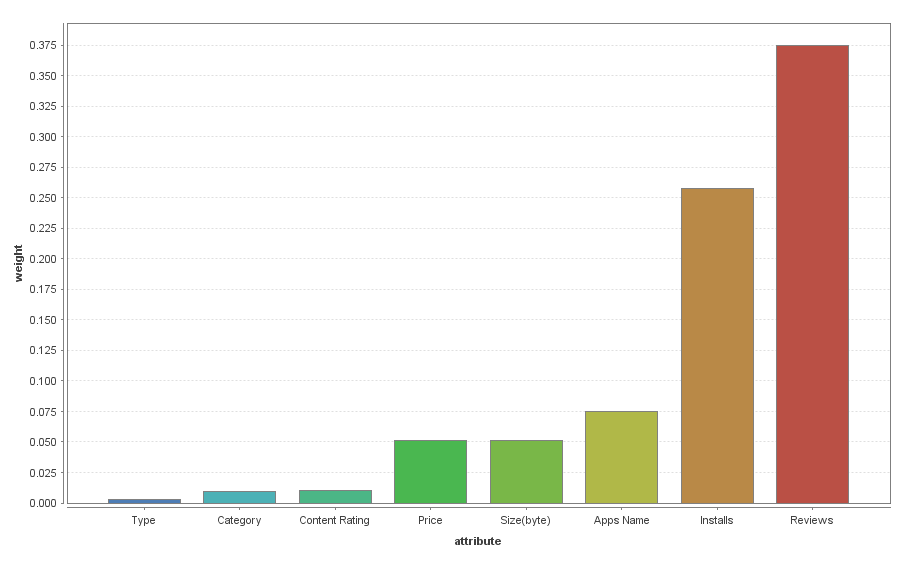
*3.1 Process Cross Validation*



Gambar 3.2 Process Cross Validation

*3.3 Hasil Feature Important*

Berdasarkan Gambar 5 dapat disimpulkan bahwa attribute *Reviews*, dan *Installs* paling berpengaruh dalam memprediksi aplikasi berperingkat tinggi dengan weight sebesar 0.0374, dan 0.0258 untuk *Installs*, untuk atribut lain, *Apps Name* bernilai 0.075, *Size* bernilai 0.052, *Price* bernilai 0.051, *Content Rating* bernilai 0.010, *Category* bernilai 0.009, dan *Type* bernilai sebesar 0.003.

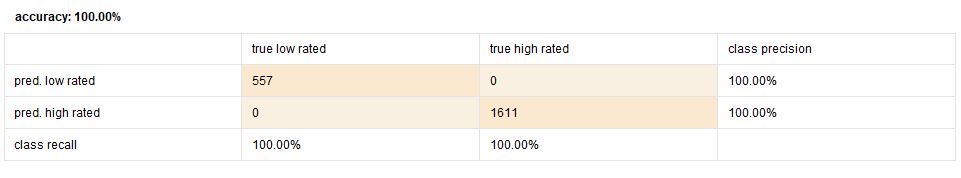


Gambar 5. Hasil Feature Important

*3.4 Performance Evaluation Independent Dataset Terhadap Peringkat Aplikasi Android*

Tabel 1. Hasil Performance Evaluation Independet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Independent Dataset** | | | | |
| **Algorithm** | **Accuracy (%)** | **Recall (%)** | **Precision (%)** | **RMSE** |
| Gradient Boost | 100% | 100% | 100 % | 0.059 |
| K-NN | 79.52% | 70.48% | 73.21% | 0.384 |
| Random Forest | 83.39% | 70.09% | 84.21% | 0.342 |
| Decision Tree | 83.39 % | 69.56% | 85.27% | 0.354 |



Gambar 6. Performance Vector - Confusion Matrix Gradient Boost Independent

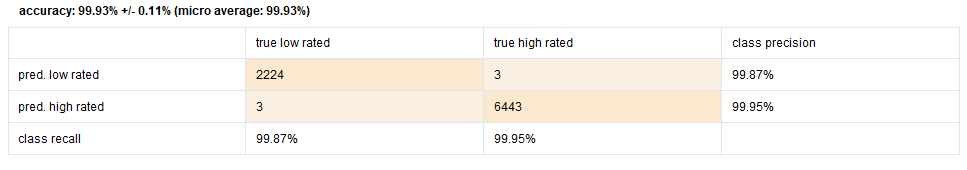
Pada tabel 1 menunjukkan hasil *Performance Evaluation* dari algoritma klasifikasi tanpa menggunakan *Cross Validation* yaitu dengan *independent dataset*. Dari hasil dapat dilihat bahwa algoritma *Gradient Boost* memiliki *accuracy, precision* dan *recall* tertinggi sebesar 100 %, 100% dan 100% dibandingkan dengan algoritma yang lain. Untuk nilai *root mean square error* terkecil dihasilkan oleh algoritma *Gradient Boost* sebesar 0.059.

*3.5 Performance Evaluation of 10-Fold Cross Validation Dataset Terhadap Aplikasi Android*

Tabel 2 menunjukkan hasil *performance evaluation* dari algoritma klasifikasi yang menggunakan 10-*fold cross validation*. Dari hasil tersebut diketahui bahwa algoritma *Gradient Boost* memiliki *accuracy*, dan *recall* tertinggi dibandingkan dengan algoritma lain dengan nilai *accuracy* sebesar 99.93%, *recall* sebesar 99.91%, *precision* sebesar 99.91%, dan nilai *RMSE* 0.062.

Tabel 2. Hasil Performance Evaluation Cross Validation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **10 Fold Cross Validation Dataset** | | | | |
| **Algorithm** | **Accuracy (%)** | **Recall (%)** | **Precision (%)** | **RMSE** |
| Gradient Boost | 99.93% | 99.91% | 99.91% | 0.062 |
| KNN | 76.92% | 67.62% | 69.40% | 0.400 |
| Random Forest | 82.95% | 69.26% | 83.69% | 0.348 |
| Decision Tree | 82.35% | 68.92% | 81.97% | 0.359 |



Gambar 7. Performance Vector - Confusion Matrix Gradient Boost melalui 10 fold cross validation

4. KESIMPULAN

Dari 4 algoritma yang telah dievaluasi yaitu *Random Forest, K-Nearest Neigbor, Gradient Boost, dan Decision Tree* dapat disimpulkan bahwa algoritma *Gradient Boost* memiliki performa yang paling baik diantara algoritma lainnya dengan hasil 100% *accuracy,* 100% *Recall,* 100 % *Precision*, dan 0.059 RMSE untuk hasil *independent* sedangkan untuk hasil 10 *fold cross validation* memiliki hasil 99.93% *accuracy*, 99.91% *recall*, 99.91% *precision*, dan 0.062 *RMSE* dalam mengklasifikasi *high rated* dan *low rated* aplikasi *android* di *google play store*.

5. SARAN

Untuk kedepannya diharapkan model ini dapat berguna untuk pembuatan aplikasi klasifikasi peringkat *android apps*, dan untuk penelitian selanjutnya diharapkan peneliti dapat menggunakan metode dan algoritma lain agar dapat memaksimalkan *performance*, dan mengurangi nilai *error* dalam proses modeling.

DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | A. Dogtiev [Online]. Available: http://www.businessofapps.com/data/app-statistics [Accessed 19 Nov 2019] |
| [2] | S. Cheney [Online]. https://www.appannie.com/en/insights/market-data/app-annie-2017-2022-forecast [Accessed 19 Nov 2018]. |
| [3] | A. Boxall [Online]. Available: http://www.businessofapps.com/12-million-mobile-developers-worldwide-nearly-half-develop-android-first [Accessed 19 Nov 2018]. |
| [4] | P. Charuza [Online]. Available: https://fueled.com/blog/much-money-can-earn-app [Accessed 19 Nov 2018]. |
| [5] | G. Bavota, M. L. V´asquez, C. E. Bernal-C´ardenas, M. D. Penta, R. Oliveto, and D. Poshyvanyk. “The impact of API change- and faultproneness on the user ratings of android apps,” IEEE Trans. Software Eng., 41(4):384–407, 2015. |
| [6] | I. J. M. Ruiz, M. Nagappan, B. Adams. “On the Relationship between the Number of Ad Libraries in an Android App and its Rating,” 2014. |
| [7] | R. Aralikatte, N. Gantayat, G. Sridhara, S. Mani. “Fault in your stars: An Analysis of Android App Reviews,” 2018 |
| [8] | M. Harman, Y. Jia, Y. Zhang. “App Store Mining and Analysis: MSR for App Stores,” 2012 |
| [9] | “kaggle” [Online]. Available: https://www.kaggle.com/lava18/google-play-store-apps [Accessed 16 Nov 2018]. |
| [10] | "OpenML," [Online]. Available: https://www.openml.org/a/estimation-procedures/1. [Accessed 26 Nov 2018]. |
| [11] | J. H. Friedman, ―Greedy Function Approximation: A Gradient Boosting Machine,‖ Ann. Stat., vol. 29, p. 5, 2001. |
| [12] | Eka, A. L. (2017, February 4). Retrieved from https://ojs.amikom.ac.id/index.php/semnasteknomedia/article/viewFile/1728/1456 |
| [13] | Shung, K. P. (2018, March 15). Retrieved from towardsdatascience.com: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9 |
| [14] | Retrieved from www.wikipedia.org: https://en.wikipedia.org/wiki/Feature\_selection |
| [15] | Barnston, A., (1992). “Correspondence among the Correlation [root mean square error] and Heidke Verification Measures; Refinement of the Heidke Score.” Notes and Correspondence, Climate Analysis Center. |