Volume 6, No. 1, Oktober 2022: 18 – 29

Jurnal Teknika
Jurnal Teoritis dan Terapan Bidang Keteknikan

KLASIFIKASI RATING APLIKASI ANDROID DI GOOGLE PLAY STORE MENGGUNAKAN ALGORITMA GRADIENT BOOST

Ade Chandra Saputra
Jurusan/Program Studi Teknik Informatika, Fakultas Teknik, Universitas Palangka Raya
Jln. Hendrik Timang, Palangka Raya
e-mail: adechandra@it.upr.ac.id

Agus Sehatman Saragih
Jurusan/Program Studi Teknik Informatika, Fakultas Teknik, Universitas Palangka Raya
Jln. Hendrik Timang, Palangka Raya
e-mail: assaragih@it.upr.ac.id

Abstrct: *The increasing number of Android applications available on the Google Play Store with the various advantages that developers get has attracted the attention of many Android application developers. To benefit from Android application development, one way is to know the characteristics of applications that have high ratings on the Google Play Store. This research will investigate the size, installs, reviews, type (free/paid), rating, category, content rating, and pricing features of apps in the Google Play Store to determine the characteristics of highly rated apps. This study uses the Gradient Boost algorithm to identify the features that have the most influence on applications with high ratings on the Google Play Store. At the preprocessing stage, this study used data cleaning and data reduction methods. This study uses important features to find out the attributes that most influence the high rating of Android applications on the Google Play Store. To classify high-rated apps the author uses the Gradient Boost algorithm.*

Keywords: *Android App, Google Play Store, Characteristics, High rating, Gradient Boost*

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 berrating 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 berrating tinggi. Penelitian ini menggunakan algoritma *Gradient Boost* untuk mengidentifikasi fitur yang paling berpengaruh pada aplikasi dengan rating 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 rating tinggi aplikasi *Android* di *Google Play Store*. Untuk mengklasifikasi aplikasi berrating tinggi penulis menggunakan algoritma *Gradient Boost*.

Kata kunci: : Aplikasi *Android, Google Play Store*, Karateristik, Rating tinggi, *Gradient Boost*

PENDAHULUAN

Penggunaan aplikasi mobile sekarang semakin meningkat besar pada abad ini. Banyak aplikasi *mobile* yang tersedia di *Google Play Store*. Karena aksesibilitas google play store yang dapat diakses di seluruh dunia membuat pengembang aplikasi berlomba – lomba membuat beragam aplikasi yang dapat di unduh pengguna. Dalam pengembangan aplikasi, pengembang perlu memprediksi aplikasi di pasar secara akurat, hasil prediksi yang akurat sangat penting dalam menunjukkan penilaian pengguna yang mempengaruhi keberhasilan suatu aplikasi. Rating diberikan oleh pengguna untuk menilai apakah aplikasi tersebut bagus atau tidak. Semakin tinggi rating yang diberikan oleh pengguna, berarti pengguna tersebut menyukai aplikasi tersebut dan dapat menjadi tolak ukur bagi pengguna lain untuk mendownload aplikasi tersebut. Tidak dapat disangkal bahwa begitu banyak aplikasi yang tersedia di google play store, tidak mungkin bagi pengguna untuk memilih satu per satu aplikasi di *google play store*. Oleh karena itu,

diperlukan sistem prediksi rating untuk menentukan aplikasi yang tepat berdasarkan rating yang diberikan oleh pengguna terhadap suatu aplikasi. Penelitian ini bertujuan untuk memanfaatkan machine learning dan konsep analitik visual untuk mendapatkan wawasan tentang bagaimana aplikasi menjadi sukses dan mencapai rating pengguna yang tinggi.

Kumpulan data yang dipilih untuk penelitian ini berasal dari situs data populer Kaggle. Ini berisi lebih dari 10 ribu data aplikasi, menangkap berbagai detail seperti kategori, ulasan, penginstalan, ukuran, dll. Tujuan dari penelitian ini adalah untuk memvisualisasikan secara umum distribusi kumpulan data di seluruh kategori, mengidentifikasi korelasi di antara parameter dan kemudian menemukan model yang akurat yang dapat secara akurat memprediksi rating pengguna di aplikasi apa pun saat data serupa tersedia. Pustaka Seaborn & Matplotlib dari python digunakan untuk melakukan visualisasi pada python. Selanjutnya, tiga model machine learning yang berbeda digunakan dan dilatih pada data ini.

Visualisasi menunjukkan bahwa aplikasi didistribusikan secara luas di 33 kategori berbeda dan kategori keluarga adalah yang paling populer dalam kumpulan data ini. Itu juga menunjukkan bahwa rating pengguna dalam kumpulan data adalah 0 atau sebagian besar antara 3,0 hingga 5,0. Dalam interval rating terakhir, distribusi secara kasar mengikuti distribusi normal dengan puncak pada rating perkiraan 4,5. Korelasi antara beberapa parameter utama juga divisualisasikan untuk data. Setelah visualisasi awal dan pemrosesan data, tujuannya adalah membuat model machine learning untuk memprediksi rating pengguna. Tiga model yang berbeda yaitu Regresi Linier Berganda, Regresi Pohon Keputusan, dan Model Pohon Peningkatan Gradien Ringan telah dibuat, dan dilatih berdasarkan data yang tersedia. Model LightGBM memprediksi rating pengguna dengan tingkat kesalahan paling sedikit dan jauh lebih baik jika dibandingkan dengan model machine learning lainnya. Akhirnya, parameter penting yang bertanggung jawab untuk memprediksi diidentifikasi. Sangat mencerahkan untuk melihat bahwa ukuran aplikasi memiliki suara tertinggi dalam rating pengguna diikuti oleh kehadiran banyak ulasan pengguna yang lebih jelas.

Penelitian ini diharapkan dapat membantu menjawab berbagai pertanyaan tentang data sehubungan dengan distribusi data, model mana yang akan digunakan untuk prediksi rating, dan akhirnya parameter mana yang memengaruhi rating. Pendekatan yang diadopsi dalam penelitian ini dapat dengan mudah diskalakan untuk kumpulan data besar yang serupa dan bila diterapkan dengan benar dapat memberikan keuntungan yang mendalam atas persaingan di pasar.

Rencana Target Luaran

Tabel 1.1 Rencana Target Luaran

No	Jenis Luaran			
	Kategori	**Sub Kategori**	**Wajib**	**Tambahan**
1	Artikel Imiah dimuat di jurnal	Internasional Bereputasi	Tidak ada	
		Nasional Terakreditasi	***Submitted***	
2	Artikel Ilmiah dimuat di prosiding	Internasional Terindeks	Tidak ada	
		Nasional	Tidak ada	
3	Invited speaker dalam temu ilmiah	Internasional	Tidak ada	
		Nasional	Tidak ada	
4	Visiting Lecturer	Internasional	Tidak ada	
5	Hak Kekayaan Intelektual (HKI)	Paten	Tidak ada	
		Paten Sederhana	Tidak ada	
		Hak Cipta		***Terdaftar***
		Merek Dagang	Tidak ada	
		Rahasia Dagang	Tidak ada	
		Desain Produk Industri	Tidak ada	
		Indikasi Geografis	Tidak ada	
		Perlindungan Varietas Tanaman	Tidak ada	
		Perlindungan Topografi Sirkuit Terpadu	Tidak ada	
6	Teknologi Tepat Guna		Tidak ada	
7	Model/Purwarupa/Desain/Karya Seni/Rekayasa Sosial			**Produk**
8	Bahan Ajar		Tidak ada	
9	Tingkat Kesiapterapan Teknologi		**4**	

Luaran Wajib

Jenis Luaran	Status target capaian (accepted, published, terdaftar atau granted, atau status lainnya)	Keterangan (url dan nama jurnal, penerbit, url paten, keterangan sejenis lainnya)

Luaran Tambahan

Jenis Luaran	Status target capaian (accepted, published, terdaftar atau granted, atau status lainnya)	Keterangan (url dan nama jurnal, penerbit, url paten, keterangan sejenis lainnya)

TINJAUAN PUSTAKA

Dalam penelitian sebelumnya tentang analisa faktor yang mempengaruhi rating aplikasi, didapatkan aplikasi dengan rating tinggi mempunyai nilai perbaikan bug API lebih rendah dibandingkan dengan aplikasi dengan rating rendah setelah data diuji dengan Mann-Whitney test (p-value < 0.0001) dan cliff's Delta (0.37). Menunjukkan bahwa ada hubungan antara rating 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 rating Android, setelah data diuji dengan spearman rank correlation antara ad libraries dalam suatu app dengan rating app menghasilkan nilai 0.016 (weak correlation). didapatkan bahwa semakin banyak iklan yang dimasukkan pengembang terhadap aplikasinya ternyata tidak terlalu berpengaruh terhadap rating Android [6].

Penelitian sebelumnya oleh Aralikatte (2018) yaitu menguji korelasi antara rating 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, rating, dan unduhan dari *Blackberry App Store*, dengan menggunakan *Spearman Correlation* didapatkan bahwa ada korelasi yang kuat antara rating 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 rating aplikasi dan meneliti fitur yang paling berpengaruh untuk mengidentifikasi aplikasi dengan rating tinggi. Berdasarkan tinjauan yang ada pada penelitian sebelumnya maka penulis menggunakan gradient boost untuk menyelidiki fitur penting yang berpengaruh terhadap penilaian rating 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 berrating tinggi.

METODE PENELITIAN

Metode pada penelitian ini akan dilaksanakan dalam beberapa tahapan, Tahapan-tahapan dalam metode penelitian sesuai dengan langkah berikut :

1. Penggunaan Data

2.

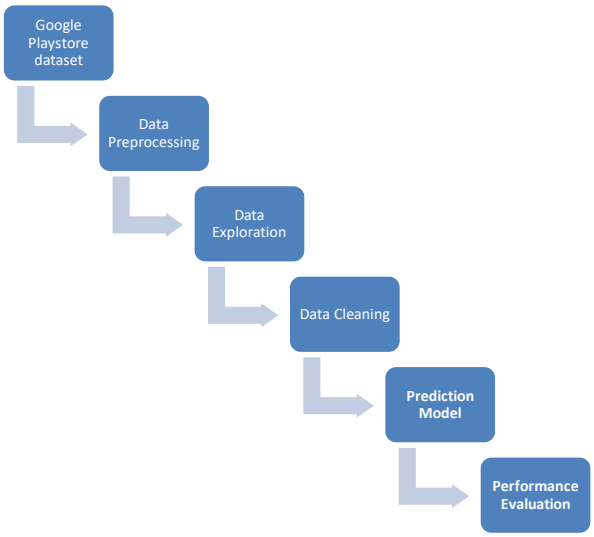
Data yang digunakan pada penelitian ini adalah Google Play Store dataset yang dapat di akses di Kaggle [9]. Dataset ini memiliki 10841 row, dan 13 attributes yang dijelaskan di Tabel 1. 12 atribut lainnya digunakan untuk memprediksi "Rating" yang merupakan rating pengguna, dengan 5.0 adalah maksimum dan 0 sesuai dengan rating pengguna minimum untuk aplikasi.

Tabel 3.1 Atribut Data set

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

3. Desain Penelitian

Gambar 1 memperlihatkan proses klasifikasi rating 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 70% training data yang terdiri atas 7558 data, dan 30% testing data yang terdiri atas 3252 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.



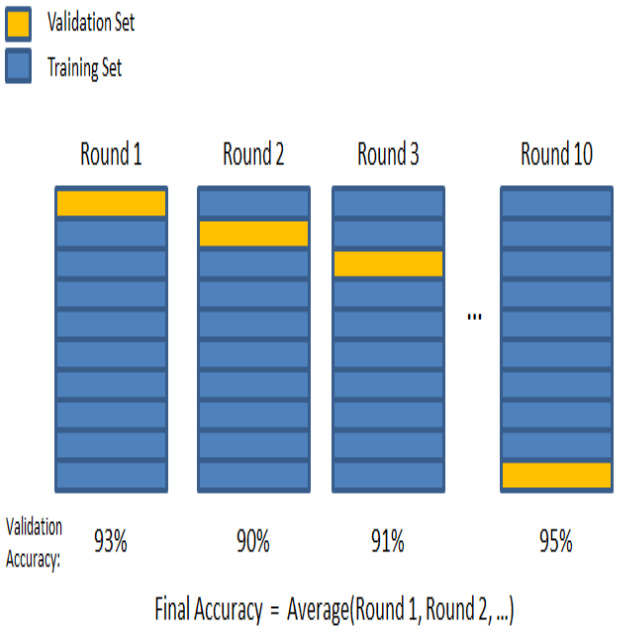
Gambar 3.1 Arsitektur Untuk Klasifikasi Rating Aplikasi di Goggleplay Store

- *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 rating 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.

- *10-Fold Cross Validation*

  Setelah data telah dibagi menjadi 70% data training dan 30% 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 3.2 Fold Cross Validation

- *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]:

  $$f(x;\ \beta_m, a_m)_1^M \quad \sum_{m=1}^{M} \beta_m h(x; a_m) \quad (1)$$

- *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:

Recal = (TP) / (TP+FN) , (2)

Precision = (TP +TN) / (TP+TN+FP+FN) (3)

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.

4. 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 rating dari atribut-atribut yang memiliki information gain yang tinggi harus diberi rating lebih tinggi daripada attributes yang lain karena lebih berpengaruh dalam mengklasifikasikan data [14]. Berikut ini adalah rumus dalam information gain:

$$\mathrm{IG(A)} = \mathrm{H(S)} - \sum \frac{\mathrm{Si}}{\mathrm{S}} \mathrm{H(Si)}, \quad (5)$$

Keterangan:
H(S) : Entropi dari dataset
H(Si) : Entropi dari i subset yang dihasilkan oleh partisi S
A : Atribut dalam dataset

5. 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:

$$\mathrm{RMSE} = \sqrt{(\mathrm{f}-0)^2} \quad (6)$$

Keterangan:

f : Perkiraan (nilai yang diharapkan atau hasil yang tidak diketahui)
o : Nilai yang diamati (hasil yang diketahui).

Tahapan-tahapan dalam metode penelitian ini dibagi menjadi 3 tahapan, yaitu:

Tahapan pertama adalah tahapan pengumpulan Analisa.Pengumpulan data dilakukan untuk memperoleh informasi mengenai objek penelitian dan penelitian-penelitian pendukung. Pengumpulan data dilakukan dengan cara studi literatur dan web browsing. Menganalisa sistem yang sedang berjalan / sistem lama serta menganalisa sistem baru atau sistem yang diusulkan Pembagian tugas bagi ketua dan anggota pada tahapan pertama adalah:

a. Ketua : melakukan studi literatur dan web browsing terkait informasi dan penelitian-penelitian pendukung.

b. Anggota : melakukan studi literatur dan web browsing terkait informasi dan penelitian-penelitian pendukung.

Tahapan kedua adalah tahap perancangan aplikasi. Metode pengembangan aplikasi yang dipilih adalah metode pengembangan sekuensial linier. Yang terdiri atas tahapan: Analisa, Desain, Code dan Pengujian[12].

Pembagian tugas bagi ketua dan anggota pada tahapan kedua adalah:

a. Ketua : melakukan tahapan Concept Analisis, Design dan Pengkodean

b. Anggota : melakukan tahapan Concept Analisis, Design dan Pengkodean

Tahapan ketiga adalah tahapan terakhir dari penelitian ini. Pada tahapan ketiga dilakukan pengujian, penyusunan laporan dan Capaian Luaran. Capaian luaran yang ditargetkan pada penelitian ini adalah menghasilkan produk, publikasi pada jurnal nasional terakreditasi dan terdaftarnya hak cipta produk.

Pembagian tugas bagi ketua dan anggota pada tahapan ketiga adalah:

a. Ketua : Melakukan pengujian.

b. Anggota : menyusun laporan dan menulis artikel.

HASIL DAN PEMBAHASAN

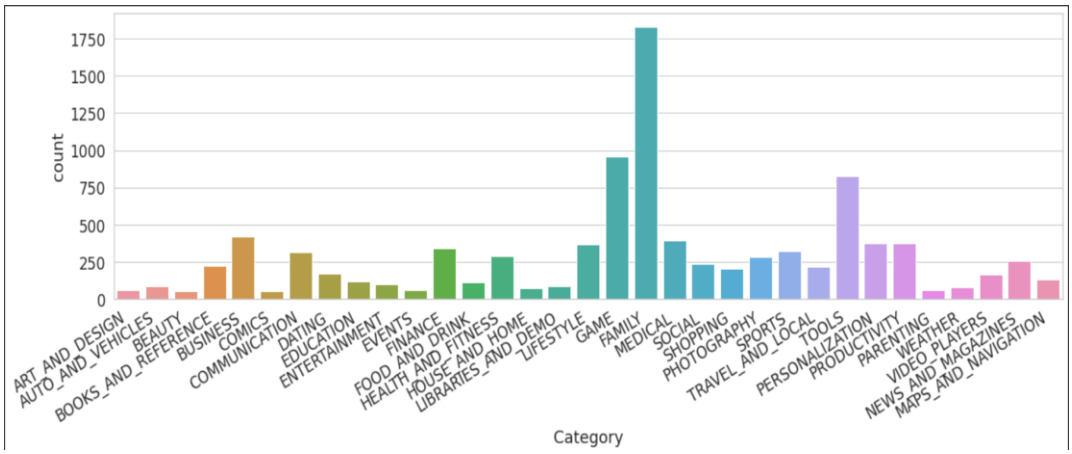
Dengan lebih dari 3,5 Juta aplikasi di dalamnya, Google Play Store menjadi tempat yang paling banyak digunakan untuk mengunduh aplikasi. Bisnis pembuatan Aplikasi sangat kompetitif dan beberapa faktor yang mendorong keberhasilan Aplikasi sangatlah penting. Sampel dataset yang digunakan adalah data dari Kaggle yang merupakan subset dari google play-store . Data berisi informasi tentang 10.000 aplikasi di 33 kategori seperti game, produktivitas, keluarga, dll. Data tersebut memberikan informasi tentang masing-masing ukuran, unduhan, ulasan, kategori, dll. dari aplikasi tersebut.

Setelah analisis awal, bahwa ada beberapa faktor yang mempengaruhi rating dan kinerja aplikasi. Rating pengguna menjadi target fokus penelitian ini. Alasan utama yang mendasari pemilihan ini adalah rata-rata rating pengguna dapat dianggap sebagai cerminan dari sentimen umum terhadap aplikasi tersebut. Selain itu, rating App store sangat penting untuk penemuan, unduhan, dan pembelian dalam aplikasi. Penelitian yang dilakukan pada dataset ini menggunakan 3 model prediksi yang berbeda untuk memprediksi rating pengguna. Ini adalah Decesin Tree, Gradient boost, dan model regresi Linear. Sebelum menjalankan model prediksi, tugas penting pembersihan data & pendeteksian outlier dilakukan. Ini diikuti dengan visualisasi data untuk lebih memahami data.

Total ada 13 atribut dan 10841 Record dalam dataset. Setiap baris sesuai dengan satu aplikasi dan kinerjanya di 13 parameter. 12 atribut lainnya digunakan untuk memprediksi “Rating” yang merupakan rating pengguna, dengan angak 5 adalah maksimum dan 0 sesuai dengan rating pengguna minimum untuk aplikasi.

1. Persiapan & Eksplorasi Data Analisis

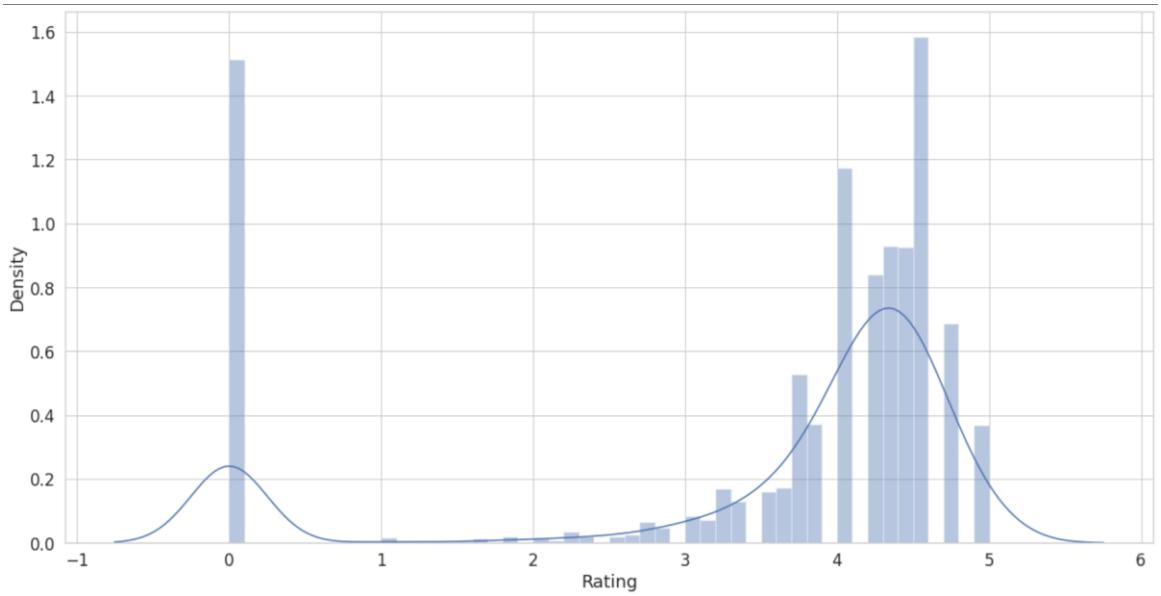
Langkah pertama dalam proses ini adalah mendeteksi baris duplikat dan rekaman ekstrem dengan nilai yang hilang. Ini menciptakan masalah selama prediksi dan karenanya telah dihapus. Setelah menghapus duplikat dan outlier menggunakan fungsi 'drop_duplicates' dan menghapus satu record yang tidak lengkap, jumlah total record bersih dalam kumpulan data adalah 9659. Ini menunjukkan bahwa sekitar 10% dari kumpulan data mentah telah digandakan dan harus dihapus. Untuk mendapatkan gambaran keseluruhan dari kumpulan data dan bagaimana berbagai aplikasi hadir di berbagai kategori, plot penghitungan kategori dibuat menggunakan paket python seaborn. Gambarnya seperti di bawah ini:



Gambar 4.1 penghitungan kategori dibuat menggunakan paket python seaborn

Dari analisis di atas terlihat bahwa 10 kategori teratas yang dimiliki aplikasi adalah sebagai berikut:

	Category	Count
11	FAMILY	1832
14	GAME	959
29	TOOLS	827
4	BUSINESS	420
20	MEDICAL	395
23	PERSONALIZATION	376
25	PRODUCTIVITY	374
18	LIFESTYLE	369
12	FINANCE	345
28	SPORTS	325



Gambar 4.2 10 kategori teratas

Tiga kategori teratas Family, Game, dan tools hampir menyumbang lebih dari 30% data. Visualisasi selanjutnya adalah untuk mendapatkan gambaran tentang bagaimana variabel prediksi 'Rating' didistribusikan ke seluruh data. Ini ditemukan dengan membuat plot distribusi menggunakan paket seaborn.

Analisis di atas menghasilkan gambaran bahwa selain dari aplikasi berperingkat 0, distribusinya adalah jenis distribusi normal dari 3 hingga 5 dengan puncak rata-rata mendekati 4,5. Matriks korelasi kemudian dibuat untuk menemukan hubungan antara variabel numerik yaitu Peringkat, Ulasan, Ukuran, Pemasangan, dan Harga. Plot korelasi ini dapat dilihat di bawah ini:

Gambar 4.3 Plot korelasi

Poin menarik yang dapat disimpulkan dari analisis di atas adalah bahwa variabel Install dan Reviews cukup berkorelasi yang masuk akal karena penggunaan aplikasi meningkat, jumlah ulasan juga meningkat.

Persiapan melibatkan pemilihan yang tepat dan konversi variabel prediktor menjadi nilai numerik yang memfasilitasi prediksi yang jauh lebih akurat. Dengan menggunakan fungsi Head() kumpulan data akan memberikan indikasi yang lebih baik.

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content_Rating	Genres	Last_Updated	Current_Ver	Android_Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19	10000	Free	0.0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14	500000	Free	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7	5000000	Free	0.0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25	50000000	Free	0.0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8	100000	Free	0.0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

Gambar 4.4 fungsi Head() pada kumpulan data

Kategori awal pertama kali dilihat dari sudut pandang logis untuk menilai apakah mereka adalah prediktor yang berguna. Prosedur ini menghasilkan penghapusan nama aplikasi kategori, genre, pembaruan terakhir, dan versi saat ini.

Jadi, prediktor utama terdiri dari dua jenis:

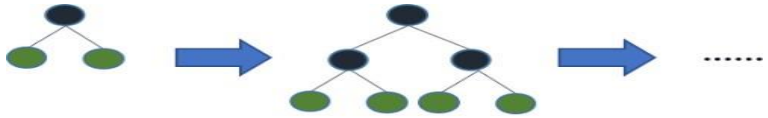
- Prediktor Kategoris: Kategori, Jenis, Rating Konten, Android Ver
- Prediktor Numerik: Ulasan, Ukuran, Pemasangan, Harga

Membuat variabel dummy adalah langkah selanjutnya untuk membuat kategori biner dari 4 prediktor kategori. Ini meningkatkan total variabel prediktor menjadi 76. Perubahan kecil dilakukan untuk mengubah prediktor numerik menjadi kolom numerik murni. Ini melibatkan penghapusan 'M' dari kolom ukuran, Tanda '+' dari kategori Pemasangan dan konversi lengkap dari kategori ini ke tipe data float. Demikian pula, variabel prediksi 'Rating' juga dikonversi ke format float.

2. Predicting Model

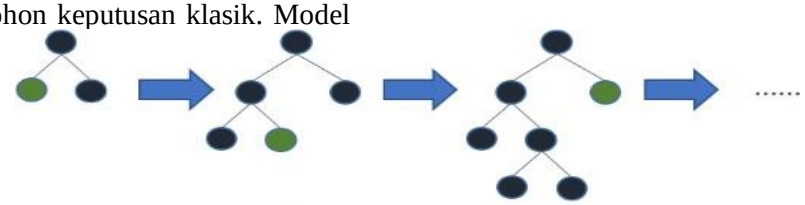
Empat model machine learning yang kuat dan banyak digunakan diterapkan untuk membuat prediksi 'Rating' untuk berbagai aplikasi. Mereka adalah sebagai berikut:

- Regresi Linier: Regresi linier mencoba memodelkan hubungan antara variabel prediktor dan variabel respon 'Rating' dengan menyesuaikan persamaan linier dengan data yang diamati.
- Regresi Pohon Keputusan: Ini menguraikan kumpulan data menjadi serangkaian himpunan bagian yang lebih kecil sambil secara bersamaan mengembangkan pohon keputusan terkait secara bertahap. Hasilnya adalah sebuah pohon dengan simpul keputusan dan simpul daun yang tumbuh.



Gambar 4.5 Level-wise tree growth

- Light Gradient Boosted Model (LightGBM): Light GBM adalah kerangka kerja peningkatan gradien berperforma tinggi berdasarkan pohon keputusan klasik. Model membagi daun pohon bukan membagi kedalaman pohon atau tingkat seperti yang diterapkan dalam berbagai algoritma lainnya.



Gambar 4.6 Level-wise tree growth

3. Pembuatan Model dan Implementasi Regresi Model

Dalam membuat model, langkah paling dasar untuk membagi data menjadi 4 kumpulan yang berbeda dan acak sangatlah penting. Data tersebut dibagi menjadi sebagai berikut:

- X_train - Variabel Prediktor untuk pelatihan
- y_train - Variabel target untuk pelatihan
- X_test - Variabel prediktor yang diketahui untuk pengujian
- y_test - Variabel target untuk pengujian

Untuk meringkas dataset asli dipecah menjadi dataset pelatihan dan dataset pengujian menggunakan pemisahan 70% - 30%. Fungsionalitas train_test_split dari perpustakaan sklearn digunakan untuk melakukan tindakan ini. Cuplikan kode kecil untuk melakukan pemisahan ini dapat dilihat di bawah

```
from sklearn.model_selection import train_test_split as ts
X_train, X_test, y_train, y_test = ts(preproc_data, target, test_size=0.3, random_state=101)
```

Gambar 4.7 Cuplikan kode kecil untuk melakukan pemisahan

Dari perpustakaan sklearn, dua regressor Linear Regression dan Decision Tree Regressor diimpor dan datanya dimasukkan ke dalam regressor ini. Dua cuplikan kode yang menunjukkan proses ini dapat dilihat di bawah:

- Banyak Linier Regresi:

```
Linear Regression

[75] lr = LinearRegression()
     lr.fit(X_train,y_train)

     LinearRegression()

[76] pred3 = lr.predict(X_test)

     mae_lr = metrics.mean_absolute_error(y_test,pred3)
     mse_lr = metrics.mean_squared_error(y_test, pred3)
     rmse_lr = np.sqrt(metrics.mean_squared_error(y_test, pred3))

     print( mae_lr, mse_lr, rmse_lr)

     1.1475782304384l3 2.4257400168745336 1.5574787372142624
```

Gambar 4.8 Banyak Linier Regresi

- Keputusan Pohon:

```
Decison Tree Model

[73] from sklearn.tree import DecisionTreeRegressor
     dt = DecisionTreeRegressor()
     #Fitting
     dt.fit(X_train,y_train)

     DecisionTreeRegressor()

[74] #Menyimpan Pengukuran model
     from sklearn import metrics
     pred1 = dt.predict(X_test)
     mae_dt = metrics.mean_absolute_error(y_test,pred1)
     mse_dt = metrics.mean_squared_error(y_test, pred1)
     rmse_dt = np.sqrt(metrics.mean_squared_error(y_test, pred1))

     print( mae_dt, mse_dt, rmse_dt)

     0.7819185645272602 2.259925810904072 1.5033049627085224
```

Gambar 4.9 Keputusan Pohon

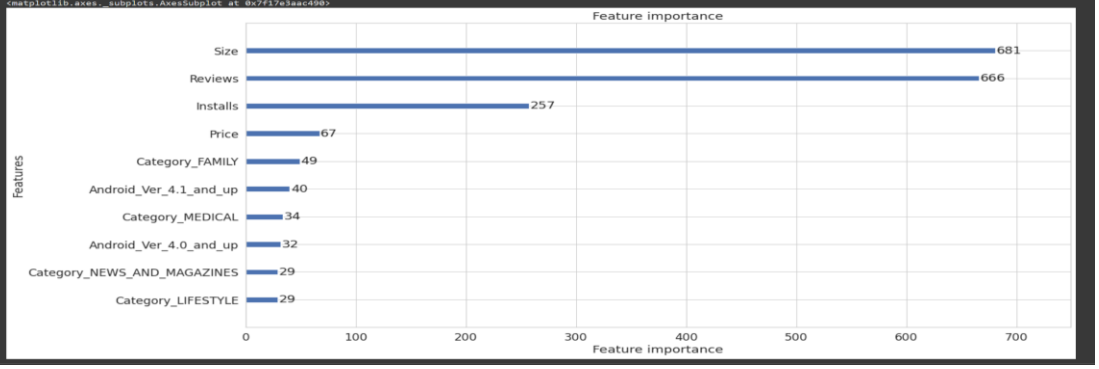
4. Model Gradien Boost

Model yang diprediksi terbaik adalah model Light Gradient Boosted Model (LightGBM). Mempertimbangkan efisiensi dan kecepatan model, Light GBM diimplementasikan pada dataset. Ini membutuhkan dan sebagian besar bergantung pada penyetelan parameter yang benar dari kecepatan pembelajaran dan jumlah daun. Model ini disusun seperti di bawah ini:

```
light gbm : Gradient boosted method

var = np.arange(1,50,5)
#Pencarian Parameter

from sklearn.model_selection import GridSearchCV
gridParams = {
'learning_rate' : [0.1, 0.12, 0.14,0.16,0.18, 0.2],
    'num_leaves' : [ 20,21,22,23,24,25,26,27]
}

#model untuk gridsearch

mdl = lgb.LGBMRegressor(metric = 'mae',
                        objective = 'regression',
                        n_estimators= 20000,
                        bagging_fraction = 0.7,
                        num_threads = 4,
                        colsample_bytree = 0.7,
                        num_boost_round = 100)

grid = GridSearchCV(mdl, gridParams, verbose =4, n_jobs = -1, scoring = 'neg_mean_absolute_error')
```

```
[79] grid.fit(X_train, y_train)

     Fitting 5 folds for each of 48 candidates, totalling 240 fits
     GridSearchCV(estimator=LGBMRegressor(bagging_fraction=0.7, colsample_bytree=0.7,
                                          metric='mae', n_estimators=20000,
                                          num_boost_round=100, num_threads=4,
                                          objective='regression'),
                  n_jobs=-1,
                  param_grid={'learning_rate': [0.1, 0.12, 0.14, 0.16, 0.18, 0.2],
                              'num_leaves': [20, 21, 22, 23, 24, 25, 26, 27]},
                  scoring='neg_mean_absolute_error', verbose=4)

[65] print(grid.best_params_)
     print(grid.best_score_)

     {'learning_rate': 0.1, 'num_leaves': 24}
     -0.6298737505237659

[66] import lightgbm as lgb

     param = {'learning_rate': 0.1, 'boosting_type': 'gbdt', 'num_leaves':25,
              'nthread':4, 'num_trees':100, 'objective': 'regression',
              'metric':'mse'}

[67] train_data = lgb.Dataset(X_train, y_train, silent =False)
     test_data = lgb.Dataset(X_test, y_test, silent = False)

     lgbt = lgb.train(param, train_set = train_data, num_boost_round =100, verbose_eval =4)

[80] pred2 = lgbt.predict(X_test, num_iteration = lgbt.best_iteration)

     mae_lgb = metrics.mean_absolute_error(y_test,pred2)
     mse_lgb = metrics.mean_squared_error(y_test, pred2)
     rmse_lgb = np.sqrt(metrics.mean_squared_error(y_test, pred2))

     print( mae_lgb, mse_lgb, rmse_lgb)

     0.6231696962356382 1.0653789968613863 1.0321719802733391
```

```
[81] #visualizing important predictors
     import matplotlib.pyplot as plt

     lgb.plot_importance(lgbt, max_num_features = 10)
```



Gambar 4.10 model Light Gradient Boosted Model (LightGBM)

Metode Gridsearchcv() digunakan untuk menemukan parameter optimal untuk learning rate dan jumlah daun. LGBM memberikan parameter error paling sedikit dengan learning rate 0,1 dan jumlah daun 25.

5. Model Perbandingan

Keempat model di atas dibandingkan berdasarkan tiga parameter MAE (Mean Absolute Error), MSE (Mean Squared Error), RMSE (Root mean Squared Error). Tabel perbandingannya bisa dilihat di bawah ini:

```
a = np.array([mae_lr,mse_lr,rmse_lr])
b = np.array([mae_dt,mse_dt,rmse_dt])
c = np.array([mae_lgb,mse_lgb,rmse_lgb])

x = pd.DataFrame(data =[a,b,c], columns = ['MAE','MSE','RMSE'], index = ['Multiple Linear Regression', 'Decision Tree Regression', 'Light Gradient Boosted Model'])
x
```

1 to 3 of 3 entries Filter

index	MAE	MSE	RMSE
Multiple Linear Regression	1.147578230438413	2.4257400168745336	1.5574787372142624
Decision Tree Regression	0.7819185645272602	2.259925810904072	1.5033049627085224
Light Gradient Boosted Model	0.6231696962356382	1.0653789968613863	1.0321719802733391

Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

Gambar 4.11 Tabel perbandingan

Model Light Gradient Boosted memberikan hasil terbaik dengan nilai MAE, MSE, dan RMSE masing-masing sebesar 0,62, 1,06, dan 1,03.

KESIMPULAN

Model Light GBM dapat menjadi titik awal yang sangat baik untuk memprediksi peringkat aplikasi tambahan berdasarkan kategori yang digunakan dalam model. Untuk menganalisis lebih lanjut wawasan yang diberikan model, fungsi fitur penting dari model diimplementasikan. Ini memberikan wawasan yang menarik tentang parameter mana yang membantu model memprediksi peringkat dengan kesalahan rendah. Analisis di atas menunjukkan bahwa dari kumpulan data saat ini dapat disimpulkan bahwa variabel Ukuran, dan Tinjauan paling memengaruhi prediksi. Dengan demikian, penelitian batu penjuru ini memberikan gambaran luas tentang bagaimana kumpulan data didistribusikan di 33 kategori aplikasi. Ini juga menunjukkan bagaimana model Light GBM memprediksi variabel 'Rating' dengan kesalahan yang sangat rendah. Prediksi untuk peringkat dapat ditingkatkan lebih lanjut dengan mengumpulkan lebih banyak data dan mendapatkan lebih banyak variabel daripada yang tersedia saat ini. Selain itu, menormalkan seluruh dataset variabel prediktor juga dapat membantu mencapai prediksi yang lebih akurat.

Pendekatan yang diadopsi untuk penelitian ini sangat fleksibel dan dapat diskalakan untuk mengakomodasi kumpulan data yang sangat besar dan lebih banyak variabel prediktor. Memanfaatkan metode dalam penelitian dapat memprediksi peringkat pengguna sebelumnya, dan sangat membantu dalam mencapai keunggulan kompetitif yang baik di ruang pasar Google Play Store yang sulit

DAFTAR PUSTAKA

A. Dogtiev [Online]. Available: http://www.businessofapps.com/data/app-statistics [Accessed 19 Nov 2019]

S. Cheney [Online]. https://www.appannie.com/en/insights/market-data/app-annie-2017-2022-forecast [Accessed 19 Nov 2018].

A. Boxall [Online]. Available: http://www.businessofapps.com/12-million-mobile-developers-worldwide-nearly-half-develop-android-first [Accessed 19 Nov 2018].

P. Charuza [Online]. Available: https://fueled.com/blog/much-money-can-earn-app [Accessed 19 Nov 2018].

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.

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.

R. Aralikatte, N. Gantayat, G. Sridhara, S. Mani. "Fault in your stars: An Analysis of Android App Reviews," 2018

M. Harman, Y. Jia, Y. Zhang. "App Store Mining and Analysis: MSR for App Stores," 2012

"kaggle" [Online]. Available: https://www.kaggle.com/lava18/google-play-store-apps [Accessed 16 Nov 2018].

"OpenML," [Online]. Available: https://www.openml.org/a/estimation-procedures/1. [Accessed 26 Nov 2018].

J. H. Friedman, ―Greedy Function Approximation: A Gradient Boosting Machine,‖ Ann. Stat., vol. 29, p. 5, 2001.

Eka, A. L. (2017, February 4). Retrieved from https://ojs.amikom.ac.id/index.php/semnasteknomedia/article/viewFile/1728/1456

Shung, K. P. (2018, March 15). Retrieved from towardsdatascience.com: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

Retrieved from www.wikipedia.org: https://en.wikipedia.org/wiki/Feature_selection

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.