Laporan Tugas Besar
UTS BIG DATA

Klasifikasi Smartphone Menggunakan KNN

Anggota Kelompok:

- 1103202133 – Fahar Nail Hakim

Dosen Pengampu: Angel Metanosa Afinda, S.Kom., M.Kom.

Program Studi S1 Teknik Komputer
Fakultas Teknik Elektro
Universitas Telkom

2025

BAB I
PENDAHULUAN

1.1 Latar Belakang

Di era digital saat ini, data bukan lagi sekadar catatan, melainkan fondasi krusial bagi pengambilan keputusan yang cerdas dan efektif di berbagai sektor. Organisasi dan individu semakin menyadari kekuatan informasi yang tersembunyi dalam tumpukan data yang masif. Kemampuan untuk menganalisis data secara mendalam telah bertransformasi menjadi keunggulan kompetitif yang signifikan.

Seiring dengan pesatnya perkembangan teknologi, terutama di bidang machine learning (ML), kemampuan untuk mengekstrak wawasan berharga dari volume data yang besar (big data) menjadi semakin canggih. Algoritma machine learning memungkinkan sistem untuk belajar dari data, mengidentifikasi pola tersembunyi, dan membuat prediksi atau rekomendasi tanpa perlu diprogram secara eksplisit untuk setiap skenario. Penerapan *machine learning* telah merevolusi berbagai industri, mulai dari personalisasi pengalaman pengguna dalam layanan e-commerce dan hiburan, hingga optimasi proses bisnis dan deteksi anomali dalam keuangan dan manufaktur.

Sebelum melangkah lebih jauh ke dalam pemodelan machine learning, terdapat tahapan krusial yang tidak boleh diabaikan, yaitu Exploratory Data Analysis (EDA). EDA adalah proses awal dalam analisis data yang bertujuan untuk memahami karakteristik utama dari dataset, mengidentifikasi pola, anomali, dan hubungan antar variabel melalui visualisasi dan ringkasan statistik. Mengapa EDA begitu penting sebelum membangun model machine learning? Jawabannya terletak pada kemampuannya untuk memberikan pemahaman yang mendalam tentang data yang akan digunakan. Tanpa EDA yang komprehensif, kita berisiko membangun model berdasarkan data yang kurang dipahami, yang dapat menghasilkan kinerja yang buruk atau bahkan kesimpulan yang salah.

Peran EDA sangatlah vital dalam membantu kita untuk "berbicara" dengan data. Melalui visualisasi seperti histogram, scatter plot, dan boxplot, kita dapat mengidentifikasi distribusi data, korelasi antar fitur, keberadaan nilai yang hilang (missing values), dan outlier. Ringkasan statistik seperti mean, median, standar deviasi, dan persentil memberikan gambaran kuantitatif tentang tendensi sentral dan penyebaran data. Pemahaman yang diperoleh dari EDA ini akan sangat membantu dalam proses feature engineering (pemilihan dan transformasi fitur), pemilihan model machine learning yang tepat, dan validasi hasil pemodelan.

Dalam konteks teknologi, khususnya industri smartphone, analisis data memainkan peran yang semakin penting. Dengan banyaknya pilihan smartphone yang tersedia di pasaran, konsumen seringkali dihadapkan pada kesulitan dalam menentukan perangkat mana yang paling sesuai dengan kebutuhan dan preferensi mereka. Studi kasus ini akan berfokus pada pemanfaatan teknik analisis data, termasuk EDA, untuk membangun sistem rekomendasi smartphone yang dipersonalisasi berdasarkan keinginan pengguna. Dengan memahami preferensi pengguna melalui data, diharapkan sistem ini dapat memberikan rekomendasi yang relevan dan membantu pengguna dalam membuat keputusan pembelian yang lebih tepat.

1.2 Tujuan Penelitian

Tujuan utama dari penelitian ini adalah untuk melakukan eksplorasi mendalam terhadap *dataset* yang relevan dengan preferensi pengguna *smartphone*. Melalui proses *Exploratory Data Analysis* (EDA), penelitian ini bertujuan untuk mengidentifikasi pola-pola tersembunyi, tren, dan hubungan antar fitur yang memengaruhi preferensi pengguna terhadap berbagai spesifikasi dan fitur *smartphone.*

Lebih lanjut, penelitian ini bertujuan untuk membangun dan mengevaluasi model *machine learning* yang mampu memberikan rekomendasi *smartphone* yang dipersonalisasi sesuai dengan keinginan pengguna. Tujuan spesifik yang ingin dicapai dalam penelitian ini meliputi:

1. **Melakukan eksplorasi data (*Exploratory Data Analysis*)** untuk memahami karakteristik *dataset*, mengidentifikasi jenis fitur yang ada, mendeteksi *missing values* dan *outlier*, serta memvisualisasikan distribusi data dan korelasi antar variabel yang relevan dengan preferensi *smartphone*.
2. **Mengidentifikasi fitur-fitur kunci** dalam *dataset* yang memiliki pengaruh signifikan terhadap preferensi pengguna *smartphone.*
3. **Membangun model *machine learning*** yang sesuai untuk tugas rekomendasi, seperti model berbasis *collaborative filtering, content-based filtering*, atau pendekatan hibrida.
4. **Melakukan pelatihan dan pengujian model** menggunakan *dataset* yang tersedia untuk memastikan model dapat belajar dari data dan memberikan rekomendasi yang akurat.
5. **Mengevaluasi kinerja model** menggunakan metrik evaluasi yang relevan untuk sistem rekomendasi, seperti presisi, *recall*, F1-score, atau metrik lainnya yang sesuai dengan karakteristik tugas rekomendasi *smartphone*.

6. **Menganalisis hasil evaluasi model** untuk mengidentifikasi kekuatan dan kelemahan model yang dibangun, serta potensi perbaikan di masa depan.

Dengan tercapainya tujuan-tujuan ini, penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan sistem rekomendasi *smartphone* yang lebih efektif dan personal, sehingga membantu pengguna dalam menemukan perangkat yang paling sesuai dengan kebutuhan dan preferensi mereka.

1.3 Ruang Lingkup

Berdasarkan eksplorasi data dan implementasi model machine learning yang telah dilakukan, berikut adalah deskripsi batasan dari tugas besar ini:

Jenis Data: Data yang digunakan dalam tugas besar ini merupakan data campuran, terdiri dari fitur numerik (seperti kapasitas RAM, memori internal, kapasitas baterai, dan harga) dan fitur kategorikal (seperti merek smartphone dan merek prosesor).

Algoritma *Machine Learning*: Algoritma machine learning yang diterapkan dalam studi kasus ini adalah K-Nearest Neighbors (KNN). Algoritma ini dipilih karena kesederhanaannya dan kemampuannya dalam memberikan rekomendasi berdasarkan kemiripan fitur antar item (dalam hal ini, smartphone).

Batasan dan Asumsi: Ketersediaan dan Kualitas Data: Penelitian ini sangat bergantung pada kualitas dan representasi data dalam dataset yang digunakan. Diasumsikan bahwa data yang ada cukup relevan dan dapat memberikan informasi yang berguna untuk memodelkan preferensi pengguna.

Penanganan *Missing Values*: Berdasarkan hasil EDA awal, terdapat missing values dalam dataset. Dalam implementasi model KNN, baris dengan missing values pada fitur-fitur yang digunakan untuk rekomendasi telah dihilangkan (dropna()). Hal ini berpotensi mengurangi jumlah data yang digunakan untuk pelatihan model.

Representasi Preferensi Pengguna: Model KNN yang dibangun merekomendasikan smartphone berdasarkan kemiripan fitur. Asumsi yang mendasarinya adalah bahwa preferensi pengguna dapat diukur dan dibandingkan berdasarkan fitur-fitur smartphone yang tersedia dalam dataset. Asumsi ini mungkin tidak sepenuhnya mencakup aspek subjektif lain yang memengaruhi keputusan pembelian pengguna (misalnya, desain, kualitas kamera, pengalaman pengguna).

Pemilihan Fitur: Pemilihan fitur yang digunakan untuk membangun model rekomendasi (ram_capacity, internal_memory, processor_brand, battery_capacity, brand_name, price) didasarkan pada intuisi dan relevansi umum terhadap preferensi

smartphone. Fitur-fitur lain yang mungkin relevan namun tidak termasuk dalam subset ini tidak dipertimbangkan dalam model.

Ukuran Dataset: Kinerja model machine learning seringkali dipengaruhi oleh ukuran dan variasi dataset. Jika dataset yang digunakan relatif kecil atau tidak mencakup variasi preferensi pengguna yang luas, kemampuan generalisasi model mungkin terbatas.

Tidak Ada Interaksi Fitur yang Kompleks: Model KNN dasar yang diterapkan di sini mempertimbangkan kemiripan berdasarkan jarak Euclidean dalam ruang fitur yang telah diproses. Model ini mungkin tidak secara eksplisit menangkap interaksi non-linear atau kompleks antar fitur yang dapat memengaruhi preferensi pengguna.

Jumlah Tetangga (k) dalam KNN: Pemilihan jumlah tetangga (n_neighbors=5) dalam model KNN bersifat arbitrer. Jumlah tetangga yang berbeda dapat menghasilkan rekomendasi yang berbeda pula. Optimasi nilai k tidak dieksplorasi secara mendalam dalam implementasi ini.

Penanganan Fitur Kategorikal: Fitur kategorikal diubah menjadi representasi numerik menggunakan One-Hot Encoding. Metode ini efektif namun dapat menghasilkan dimensi fitur yang lebih tinggi, yang berpotensi mempengaruhi kinerja model pada dataset yang sangat besar dengan banyak fitur kategorikal.

Fokus pada rekomendasi berdasarkan kemiripan tugas besar ini berfokus pada pembangunan sistem rekomendasi berdasarkan kemiripan fitur smartphone. Pendekatan lain seperti *collaborative filtering* berdasarkan riwayat interaksi pengguna tidak diimplementasikan karena keterbatasan data atau fokus penelitian.

BAB II.
STUDI PUSTAKA

2.1 Eksplorasi Data

Exploratory Data Analysis adalah langkah awal yang krusial dalam analisis data. Tujuannya adalah untuk memahami karakteristik mendasar dari data kita sebelum melakukan pemodelan atau analisis yang lebih mendalam.Teknik Statistik Deskriptif: Ini melibatkan penggunaan ringkasan statistik seperti rata-rata, median, standar deviasi, nilai minimum dan maksimum untuk memberikan gambaran kuantitatif tentang fitur-fitur dalam dataset. Misalnya, kita bisa melihat rentang harga smartphone atau distribusi kapasitas RAM.Visualisasi Data: Representasi grafis data, seperti histogram, scatter plot, boxplot, dan heatmap, membantu kita melihat pola dan hubungan yang mungkin sulit ditangkap hanya dengan melihat angka. Visualisasi membuat data lebih intuitif dan mudah dipahami.

Struktur Data: EDA membantu kita mengidentifikasi jenis variabel , ukuran dataset, dan bagaimana data tersebut terorganisir.Pola dan Tren: Melalui visualisasi dan statistik deskriptif, kita dapat menemukan tren menarik, pola musiman, atau hubungan antar variabel yang mungkin relevan untuk analisis lebih lanjut atau pengembangan model.Anomali dan Outlier: EDA memungkinkan kita untuk mengidentifikasi nilai-nilai yang tidak biasa yang mungkin disebabkan oleh kesalahan input data atau merupakan kejadian yang menarik untuk diteliti lebih lanjut.

Kualitas Data: EDA membantu dalam mengidentifikasi masalah kualitas data seperti nilai yang hilang , inkonsistensi format, atau kesalahan ketik yang perlu ditangani sebelum analisis lebih lanjut.dalam Pemilihan Metode Analisis: Pemahaman yang diperoleh dari EDA dapat membimbing kita dalam memilih teknik statistik atau algoritma machine learning yang paling sesuai untuk menjawab pertanyaan penelitian atau memecahkan masalah yang ada. Intuisi Bisnis: Dalam konteks aplikasi bisnis, EDA dapat membantu stakeholder memahami data dengan lebih baik dan menghasilkan pertanyaan atau hipotesis yang lebih relevan.

Beberapa metode umum yang digunakan dalam EDA meliputi Distribusi Data: Memeriksa bagaimana nilai-nilai dalam setiap variabel terdistribusi. Ini bisa dilakukan dengan histogram untuk data numerik dan bar chart untuk data kategorikal. Korelasi Antar Variabel: Mengukur seberapa kuat dan ke arah mana hubungan antara dua atau lebih variabel numerik. Scatter plot dan heatmap korelasi adalah alat yang umum digunakan. Outlier: Mencari nilai-nilai ekstrem yang jauh berbeda dari sebagian besar data. Boxplot dan scatter plot dapat membantu dalam visualisasi outlier.

Data Kategorikal: Memeriksa frekuensi dan proporsi setiap kategori dalam variabel kategorikal menggunakan bar chart atau pie chart. Hubungan Antara Variabel Numerik dan Kategorikal: Memvisualisasikan bagaimana distribusi variabel numerik berbeda di berbagai kategori variabel kategorikal, sering menggunakan boxplot atau violin plot. Singkatnya, EDA adalah

proses investigasi awal yang penting untuk memahami «seluk-beluk» data kita, memastikan kualitasnya, dan mengarahkan langkah analisis selanjutnya agar lebih efektif dan relevan. Tanpa EDA yang baik, kita berisiko membuat asumsi yang salah atau melewatkan insight penting yang tersembunyi dalam data.

Machine Learning

Machine learning adalah bidang ilmu komputer yang memungkinkan sistem komputer untuk belajar dari data tanpa diprogram secara eksplisit. Alih-alih mengikuti serangkaian instruksi yang telah ditentukan, algoritma machine learning mengidentifikasi pola dalam data, membangun model berdasarkan pola tersebut, dan kemudian menggunakan model ini untuk membuat prediksi atau keputusan baru. Esensi dari machine learning adalah kemampuan untuk meningkatkan kinerja pada tugas tertentu seiring dengan bertambahnya pengalaman .

Supervised Learning

Dalam supervised learning, algoritma belajar dari labeled data. Labeled data adalah dataset yang terdiri dari pasangan input dan output yang benar. Tujuan algoritma adalah untuk mempelajari fungsi pemetaan yang menghubungkan input ke output, sehingga ketika diberikan input baru, algoritma dapat memprediksi output yang sesuai.
Analogi: Seperti seorang siswa yang belajar dengan diberikan soal dan kunci jawabannya. Setelah melihat banyak contoh soal dan jawabannya, siswa tersebut diharapkan mampu menjawab soal-soal baru dengan benar. Contoh Tugas: Klasifikasi , dan Regresi .

Unsupervised Learning

Dalam unsupervised learning, algoritma belajar dari unlabeled data. Tidak ada output target yang diberikan selama proses pelatihan. Tujuan algoritma adalah untuk menemukan struktur tersembunyi atau pola menarik dalam data itu sendiri. Analogi: Seperti seorang ilmuwan yang menjelajahi data baru tanpa petunjuk awal tentang apa yang mungkin ditemukan. Ilmuwan tersebut mencoba mengidentifikasi kelompok data yang serupa atau mengurangi dimensi data untuk visualisasi yang lebih baik Contoh Tugas: Clustering , Dimensionality Reduction , dan Association Rule Mining .

Reinforcement Learning

Dalam reinforcement learning, agen belajar bagaimana bertindak dalam suatu lingkungan untuk memaksimalkan imbalan kumulatif. Agen berinteraksi dengan lingkungan, melakukan tindakan, dan menerima umpan balik dalam bentuk imbalan atau hukuman. Tujuan agen adalah untuk mempelajari kebijakan yang akan menghasilkan imbalan terbesar dari waktu ke waktu. Analogi: Seperti melatih seekor hewan dengan memberikan hadiah untuk perilaku yang diinginkan dan hukuman untuk perilaku yang tidak diinginkan. Hewan tersebut belajar

tindakan mana yang menghasilkan hadiah terbanyak. Contoh Tugas: Melatih robot untuk berjalan, mengembangkan strategi bermain game , dan mengoptimalkan sistem kontrol .

Regresi Linear

Cara Kerja: Algoritma regresi linear mencoba untuk memodelkan hubungan linear antara variabel dependen dan satu atau lebih variabel independen . Tujuannya adalah untuk menemukan garis yang paling sesuai dengan data, sehingga dapat digunakan untuk memprediksi nilai target berdasarkan nilai input baru.
Relevansi untuk Rekomendasi : Jika kita ingin memprediksi rating atau preferensi numerik pengguna terhadap suatu smartphone berdasarkan fiturnya, regresi linear bisa digunakan.

Decision Trees

Cara Kerja: Decision tree adalah model seperti struktur pohon di mana setiap node internal mewakili pengujian pada suatu fitur, setiap cabang mewakili hasil dari pengujian tersebut, dan setiap node daun mewakili keputusan atau prediksi . Pohon dibangun dengan membagi data secara rekursif berdasarkan fitur yang paling informatif dalam memisahkan target variabel.
Relevansi untuk Rekomendasi : Decision tree dapat digunakan untuk mengklasifikasikan smartphone ke dalam kategori preferensi pengguna atau memprediksi preferensi berdasarkan serangkaian aturan keputusan yang dipelajari dari data.

Random Forests

Cara Kerja: Random Forest adalah algoritma ensemble learning yang membangun banyak decision tree selama pelatihan dan kemudian menggabungkan prediksi dari setiap pohon untuk membuat prediksi akhir. Ini membantu mengurangi overfitting dan meningkatkan robustitas model.
Relevansi untuk Rekomendasi : Seperti decision tree, random forest dapat digunakan untuk tugas klasifikasi preferensi smartphone atau regresi untuk memprediksi rating. Kinerja ensemble seringkali lebih baik daripada satu decision tree tunggal.

Support Vector Machines

Cara Kerja: SVM adalah algoritma supervised learning yang mencoba menemukan hyperplane terbaik yang memisahkan data ke dalam kelas-kelas yang berbeda dengan margin sebesar mungkin. Untuk data yang tidak dapat dipisahkan secara linear, SVM dapat menggunakan kernel trick untuk memproyeksikan data ke ruang dimensi yang lebih tinggi di mana pemisahan linear mungkin terjadi. Relevansi untuk Rekomendasi : SVM dapat digunakan untuk mengklasifikasikan preferensi pengguna terhadap smartphone .

2.3 Pemilihan Algoritma dan Metode Evaluasi

Alasan Pemilihan Algoritma K-Nearest Neighbors

Kesederhanaan dan Interpretasi: KNN adalah algoritma yang relatif sederhana untuk diimplementasikan dan dipahami. Logika di baliknya intuitif: smartphone yang memiliki fitur serupa akan direkomendasikan kepada pengguna yang mencari spesifikasi tertentu. Hasil rekomendasi dapat diinterpretasikan dengan melihat smartphone tetangga terdekat dari preferensi input pengguna.

Pendekatan Berbasis Kemiripan : Tugas rekomendasi ini berfokus pada menemukan smartphone yang paling mirip dengan preferensi yang diinputkan oleh pengguna dalam ruang fitur. KNN secara alami bekerja dengan mengukur jarak antara titik data. Dalam kasus ini, setiap smartphone direpresentasikan sebagai titik dalam ruang multidimensi fitur-smartphone.
Fleksibilitas terhadap Jenis Data: KNN dapat bekerja dengan baik pada data numerik dan kategorikal . Dataset smartphone umumnya memiliki campuran fitur numerik dan kategorikal .
Pendekatan Non-Parametrik: KNN adalah algoritma non-parametrik, yang berarti tidak membuat asumsi yang kuat tentang bentuk fungsi pemetaan yang mendasari data. Ini bisa menjadi keuntungan jika hubungan antara fitur dan preferensi pengguna tidak diketahui atau kompleks.
Kemampuan Rekomendasi Ad-hoc: KNN dapat dengan mudah memberikan rekomendasi berdasarkan input preferensi pengguna secara langsung tanpa memerlukan fase pelatihan model yang rumit untuk setiap pengguna baru .

Presisi@K dan Recall@K

Presisi@K: Mengukur proporsi K item teratas yang direkomendasikan yang relevan bagi pengguna. Relevansi biasanya didefinisikan berdasarkan interaksi positif pengguna dengan item tersebut di masa lalu .Recall@K: Mengukur proporsi total item yang relevan bagi pengguna yang berhasil direkomendasikan dalam K item teratas.

Aplikasi untuk Tugas Ini : Jika kita memiliki data historis tentang smartphone mana yang disukai atau dibeli pengguna setelah melihat rekomendasi, kita dapat menggunakan metrik ini untuk melihat seberapa baik model merekomendasikan smartphone yang relevan. Merupakan rata-rata harmonik dari Presisi@K dan Recall@K. Metrik ini memberikan keseimbangan antara presisi dan recall. Mean Average Precision : Merupakan rata-rata presisi rata-rata untuk daftar rekomendasi pengguna. Ini mempertimbangkan urutan rekomendasi; item yang relevan yang muncul lebih awal dalam daftar mendapatkan skor yang lebih tinggi. Area Under the ROC Curve : Meskipun lebih umum digunakan dalam klasifikasi biner, AUC dapat diadaptasi untuk rekomendasi dengan memperlakukan item yang berinteraksi dengan pengguna sebagai positif dan yang tidak berinteraksi sebagai negatif. AUC mengukur probabilitas bahwa model akan memberi peringkat item positif yang dipilih secara acak lebih tinggi daripada item negatif yang dipilih secara acak.

Root Mean Squared Error : Jika tugasnya adalah memprediksi rating numerik yang akan diberikan pengguna pada smartphone, RMSE dapat digunakan untuk mengukur perbedaan antara rating yang diprediksi dan rating sebenarnya.Pemilihan model terbaik akan bergantung pada tujuan spesifik sistem rekomendasi dan metrik evaluasi yang dianggap paling penting.

Memilih Model Terbaik Berdasarkan Metrik Evaluasi

Definisikan Tujuan: Apakah kita lebih fokus pada memberikan beberapa rekomendasi yang sangat relevan atau memastikan bahwa kita mencakup sebanyak mungkin item yang relevan dalam daftar rekomendasi Tujuan ini akan memengaruhi metrik mana yang lebih diprioritaskan.
Bandingkan Metrik: Jika kita melatih beberapa model , kita akan mengevaluasi kinerja setiap model menggunakan metrik yang relevan pada hold-out test set.Pilih Berdasarkan Kinerja Tertinggi: Model dengan nilai metrik evaluasi tertinggi akan dianggap sebagai model terbaik. Misalnya, jika presisi pada beberapa rekomendasi teratas sangat penting, kita akan memilih model dengan Presisi@K tertinggi.

Jika cakupan item yang relevan lebih penting, kita akan fokus pada Recall@K atau MAP. F1-score berguna ketika kita ingin menyeimbangkan presisi dan recall. Pertimbangkan Faktor Lain: Selain metrik kuantitatif, pertimbangkan juga faktor lain seperti kompleksitas model, waktu komputasi untuk pelatihan dan inferensi, interpretasi hasil rekomendasi, dan kebutuhan untuk cold-start handling .Dalam konteks tugas besar ini, karena kita menerima input preferensi pengguna secara langsung dan memberikan rekomendasi ad-hoc, evaluasi tradisional dengan membagi data menjadi latih dan uji mungkin kurang relevan untuk demonstrasi saat ini. Namun, jika kita memiliki data interaksi pengguna historis, metodologi evaluasi di atas akan sangat penting untuk mengukur dan membandingkan kinerja berbagai pendekatan rekomendasi.

BAB III
METODOLOGI

3.1 Deskripsi Dataset

Sumber Data: Dataset yang digunakan dalam tugas besar ini berasal dari file CSV yang diunggah oleh pengguna. Nama file spesifiknya adalah 'smartphone_cleaned_v2.csv'. Asal-usul pasti dari dataset ini tidak secara eksplisit disebutkan dalam kode, namun diasumsikan berisi informasi tentang berbagai spesifikasi dan fitur smartphone. Ukuran Dataset: Berdasarkan hasil dari df.shape, dataset ini terdiri dari 980 baris dan 21 kolom . Fitur atau Atribut: Dataset ini mengandung berbagai fitur yang mendeskripsikan karakteristik smartphone.

Fitur Numerik

•ram_capacity: Kapasitas RAM dalam GB.

•internal_memory: Kapasitas memori internal dalam GB.

•battery_capacity: Kapasitas baterai dalam mAh.

•price: Harga smartphone.

•Beberapa fitur numerik lain yang mungkin ada namun tidak secara eksplisit digunakan dalam pemodelan saat ini berdasarkan df.describe:

display_size, refresh_rate, num_rear_cameras, num_front_cameras, rating_count, rating_stars .

Fitur Kategorikal

•brand_name: Merek smartphone .

•model: Nama model smartphone.

•processor_brand: Merek prosesor .

•os: Sistem operasi .

•Beberapa fitur kategorikal lain yang mungkin ada namun tidak secara eksplisit digunakan dalam pemodelan saat ini berdasarkan df.info dan df.head:

display_type, extended_memory_available, primary_camera, secondary_camera, network_ty
pe, bluetooth, wifi, nfc, infrared_port, sim_slots, fast_charging_available.
Jenis Fitur: Seperti yang disebutkan di atas, data terdiri dari campuran fitur numerik dan fitur kategorikal. Fitur numerik merepresentasikan nilai-nilai kuantitatif, sedangkan fitur kategorikal merepresentasikan kategori atau kelompok.

Relevansi Data dengan Masalah: Dataset ini sangat relevan dengan masalah rekomendasi smartphone. Setiap baris dalam dataset mewakili sebuah smartphone dengan berbagai atribut spesifikasinya. Dengan menganalisis fitur-fitur ini, kita dapat mengidentifikasi smartphone mana yang memiliki karakteristik serupa dan merekomendasikannya kepada pengguna berdasarkan preferensi fitur yang mereka masukkan.

Variabel Target: Dalam implementasi sistem rekomendasi KNN saat ini, tidak ada variabel target eksplisit yang diprediksi atau diklasifikasikan dalam pengertian supervised learning tradisional. Alih-alih, algoritma KNN digunakan untuk menemukan smartphone yang paling mirip dengan input pengguna berdasarkan seluruh set fitur yang dipilih. Namun, jika kita memiliki data preferensi pengguna historis , kita dapat menggunakan fitur-fitur smartphone untuk memprediksi preferensi ini di masa depan, yang akan menjadikan preferensi sebagai variabel target dalam konteks evaluasi model rekomendasi.

Pembagian Subset Data: Berdasarkan kode yang diberikan, dataset tidak secara eksplisit dibagi menjadi subset data pelatihan dan data uji. Seluruh dataset digunakan untuk membangun model KNN dan memberikan rekomendasi berdasarkan input pengguna secara ad-hoc. Jika evaluasi model yang lebih formal ingin dilakukan, dataset perlu dibagi menjadi data pelatihan dan data uji .

3.2 Persiapan Data

1. Pembersihan Data (*Data Cleaning*)

- **Penanganan Nilai yang Hilang (*Missing Values*)**: Berdasarkan hasil df.isnull().sum(), kita mengetahui kolom mana saja yang memiliki nilai yang hilang. Dalam kode implementasi, langkah penanganan nilai yang hilang dilakukan pada saat pemilihan fitur untuk rekomendasi:

Python

```
fitur_rekomendasi = ['ram_capacity', 'internal_memory', 'processor_brand',
'battery_capacity','brand_name','price']
df_rekomendasi = df[fitur_rekomendasi].copy()
df_rekomendasi.dropna(inplace=True)
```

Pada bagian ini, kita memilih subset data yang hanya terdiri dari fitur-fitur yang akan digunakan untuk rekomendasi. Kemudian, metode dropna(inplace=True) diterapkan pada df_rekomendasi. Ini berarti **baris-baris yang memiliki nilai yang hilang pada salah satu dari fitur-fitur yang dipilih akan dihapus** dari df_rekomendasi.

Alasan penghapusan baris dengan nilai yang hilang pada fitur-fitur kunci ini adalah untuk memastikan bahwa model KNN hanya dilatih dan memberikan rekomendasi berdasarkan data yang lengkap untuk fitur-fitur tersebut. Metode imputasi (mengisi nilai yang hilang dengan rata-rata, median, atau KNN imputation) tidak diterapkan pada tahap ini untuk menjaga kesederhanaan dan karena fokusnya adalah pada rekomendasi berdasarkan fitur-fitur yang secara eksplisit ada.

2. Pemilihan Fitur (*Feature Selection*)

- Langkah pemilihan fitur dilakukan secara manual berdasarkan pemahaman tentang atribut-*smartphone* mana yang paling relevan dalam menentukan preferensi pengguna. Fitur-fitur yang dipilih adalah: 'ram_capacity', 'internal_memory', 'processor_brand', 'battery_capacity', 'brand_name', dan 'price'. Pemilihan fitur ini didasarkan pada asumsi bahwa spesifikasi-spesifikasi ini secara signifikan memengaruhi keputusan pembelian dan preferensi pengguna terhadap *smartphone*.

3. Transformasi Fitur (*Feature Transformation*)

- **Normalisasi Data Numerik**: Fitur-fitur numerik yang dipilih ('ram_capacity', 'internal_memory', 'battery_capacity', 'price') dinormalisasi menggunakan MinMaxScaler dari sklearn.preprocessing.

Python

```
preprocessor = ColumnTransformer(
   transformers=[
      ('num', MinMaxScaler(), fitur_numerik),
      ('cat', OneHotEncoder(handle_unknown='ignore', sparse_output=False),
fitur_kategorikal)
   ])
```

MinMaxScaler menskalakan fitur-fitur numerik ke dalam rentang antara 0 dan 1. Normalisasi ini penting karena algoritma KNN berbasis jarak sangat sensitif terhadap skala fitur. Tanpa normalisasi, fitur dengan rentang nilai yang lebih besar dapat mendominasi perhitungan jarak.

- **Encoding Data Kategorikal**: Fitur-fitur kategorikal yang dipilih ('processor_brand', 'brand_name') diubah menjadi format numerik menggunakan OneHotEncoder dari sklearn.preprocessing.

Python

```
preprocessor = ColumnTransformer(
    transformers=[
        ('num', MinMaxScaler(), fitur_numerik),
        ('cat', OneHotEncoder(handle_unknown='ignore', sparse_output=False),
fitur_kategorikal)
    ])
```

OneHotEncoder membuat kolom biner baru untuk setiap kategori unik dalam setiap fitur kategorikal. Jika suatu *smartphone* memiliki kategori tertentu, kolom biner yang sesuai akan bernilai 1, dan sisanya 0. Penggunaan *One-Hot Encoding* penting karena algoritma KNN bekerja dengan data numerik. *One-Hot Encoding* menghindari asumsi ordinalitas antar kategori yang mungkin terjadi jika kita menggunakan *Label Encoding*. Opsi handle_unknown='ignore' digunakan untuk menangani kasus di mana input pengguna mungkin mengandung kategori yang tidak ada dalam data pelatihan. Opsi sparse_output=False memastikan bahwa output dari *encoder* adalah array NumPy, bukan matriks *sparse*.

- **Penggunaan ColumnTransformer**: Seluruh proses transformasi (normalisasi untuk fitur numerik dan *one-hot encoding* untuk fitur kategorikal) diorganisasikan menggunakan ColumnTransformer. Ini memungkinkan kita untuk menerapkan transformasi yang berbeda pada kolom yang berbeda dalam *DataFrame* secara efisien.

Setelah langkah-langkah ini, data siap digunakan untuk melatih model KNN. Data yang telah diproses (df_processed) berisi fitur-fitur numerik yang telah dinormalisasi dan fitur-fitur kategorikal yang telah di-*encode* menjadi representasi numerik. Representasi numerik ini memungkinkan algoritma KNN untuk menghitung jarak antara berbagai *smartphone* berdasarkan fitur-fitur mereka.

3.3 Eksplorasi Data (EDA)

Analisis Statistik Deskriptif

- **Implementasi:** Statistik deskriptif untuk kolom numerik dihitung menggunakan fungsi df.describe() dari Pandas.
- **Hasil dan Interpretasi:** Hasilnya memberikan informasi penting seperti:
    - **Mean (Rata-rata):** Nilai tengah dari distribusi data. Misalnya, rata-rata kapasitas RAM, memori internal, baterai, dan harga *smartphone*.
    - **Median (Nilai Tengah):** Nilai yang membagi distribusi data menjadi dua bagian yang sama. Perbandingan antara mean dan median dapat mengindikasikan kemiringan distribusi. Jika mean jauh lebih tinggi atau lebih rendah dari median, data mungkin miring.
    - **Standard Deviation (Std):** Ukuran seberapa tersebar data di sekitar mean. Standar deviasi yang tinggi menunjukkan variabilitas data yang besar.
    - **Minimum (Min):** Nilai terkecil dalam kolom.
    - **Maximum (Max):** Nilai terbesar dalam kolom.
    - **Quartiles (25%, 50%, 75%):** Nilai-nilai yang membagi data menjadi empat bagian yang sama. Ini membantu dalam memahami penyebaran dan pusat data.
- **Kegunaan:** Statistik deskriptif memberikan gambaran awal tentang rentang nilai, tendensi sentral, dan penyebaran setiap fitur numerik. Ini membantu mengidentifikasi potensi masalah seperti skala fitur yang berbeda jauh, yang kemudian ditangani dengan normalisasi. Selain itu, informasi tentang kemiringan distribusi dapat berguna untuk pemilihan model atau transformasi fitur di masa mendatang (meskipun tidak diterapkan dalam implementasi KNN sederhana ini).

Visualisasi Data

- **Implementasi:** Berbagai jenis visualisasi digunakan:
    - **Histogram (df[numerical_cols].hist()):** Menampilkan distribusi frekuensi dari setiap fitur numerik. Ini membantu dalam mengidentifikasi apakah distribusi data mendekati normal, miring ke kiri atau kanan, atau memiliki beberapa puncak (multimodal).

- **Heatmap Nilai yang Hilang (sns.heatmap(df.isnull())):** Memvisualisasikan keberadaan nilai yang hilang dalam dataset. Ini memberikan gambaran cepat tentang pola nilai yang hilang (misalnya, apakah nilai yang hilang terkonsentrasi pada kolom tertentu atau tersebar acak).
- **Heatmap Korelasi (sns.heatmap(df[numerical_cols].corr())):** Menampilkan matriks korelasi antar fitur numerik dalam bentuk *heatmap*. Warna dan anotasi menunjukkan kekuatan dan arah korelasi (positif atau negatif).
- **Countplot (sns.countplot()):** Menampilkan distribusi frekuensi untuk setiap kategori dalam fitur kategorikal. Ini membantu dalam memahami proporsi setiap kategori.
- **Boxplot (sns.boxplot()):** Menampilkan ringkasan distribusi data numerik melalui kuartil, median, dan potensi *outlier*. Ini sangat berguna untuk mendeteksi *outlier* dan melihat penyebaran data antar kelompok (misalnya, harga vs. merek).
- **Violin Plot (sns.violinplot()):** Mirip dengan *boxplot* tetapi juga menampilkan perkiraan fungsi kepadatan probabilitas dari data. Ini memberikan gambaran yang lebih detail tentang bentuk distribusi.

- **Hasil dan Interpretasi:**
  - Histogram menunjukkan distribusi setiap fitur numerik, memungkinkan identifikasi potensi kemiringan atau multimodalitas.
  - Heatmap nilai yang hilang mengonfirmasi keberadaan nilai yang hilang pada beberapa kolom.
  - Heatmap korelasi menunjukkan adanya korelasi positif antara beberapa fitur numerik (misalnya, mungkin ada korelasi antara RAM dan harga). Korelasi yang tinggi bisa menjadi pertimbangan jika menggunakan model yang sensitif terhadap multikolinearitas.
  - Countplot memberikan visualisasi distribusi setiap merek dan merek prosesor, menunjukkan popularitas relatif dari setiap kategori.
  - Boxplot (harga vs. merek) dan *violin plot* (RAM vs. OS) menunjukkan bagaimana distribusi fitur numerik bervariasi di berbagai kategori, memberikan *insight* tentang hubungan antar jenis fitur.
- **Kegunaan:** Visualisasi data membantu dalam mengidentifikasi pola, anomali (termasuk *outlier*), dan hubungan antar fitur secara intuitif. Ini melengkapi analisis

statistik deskriptif dan dapat mengarahkan langkah-langkah *preprocessing* data lebih lanjut.

Korelasi Antar Fitur

- **Implementasi:** Korelasi antar fitur numerik dihitung menggunakan metode .corr() pada *DataFrame* yang berisi hanya kolom numerik, dan hasilnya divisualisasikan menggunakan sns.heatmap().
- **Hasil dan Interpretasi:** *Heatmap* korelasi menunjukkan pasangan fitur numerik mana yang memiliki korelasi tinggi (positif atau negatif). Misalnya, jika ada korelasi positif yang kuat antara RAM dan harga, ini berarti *smartphone* dengan RAM lebih tinggi cenderung memiliki harga yang lebih tinggi.
- **Kegunaan:** Mengidentifikasi fitur yang sangat berkorelasi penting karena dapat menyebabkan masalah multikolinearitas dalam beberapa model (seperti regresi linear), yang dapat membuat interpretasi koefisien model menjadi sulit dan meningkatkan varians estimasi. Meskipun KNN tidak secara langsung terpengaruh oleh multikolinearitas dengan cara yang sama seperti regresi, pemahaman korelasi dapat membantu dalam pemilihan fitur jika kita ingin mengurangi redundansi.

Outliers

- **Implementasi:** *Outlier* dideteksi secara visual menggunakan *boxplot* pada kolom 'price'.
- **Hasil dan Interpretasi:** *Boxplot* untuk harga menunjukkan adanya beberapa titik data yang berada jauh di luar "whisker" boxplot, yang mengindikasikan potensi *outlier*.
- **Penanganan:** Dalam kode yang diberikan, tidak ada penanganan eksplisit terhadap *outlier*. *Outlier* yang terdeteksi pada kolom 'price' (dan mungkin pada fitur numerik lainnya) dibiarkan ada dalam data yang digunakan untuk melatih model KNN.
- **Pertimbangan:** Keputusan untuk menghapus, mengoreksi, atau mempertahankan *outlier* sangat bergantung pada konteks masalah dan penyebab *outlier*. Jika *outlier* dianggap sebagai kesalahan pengukuran atau entri data yang tidak valid, penghapusan atau koreksi mungkin diperlukan. Namun, jika *outlier* merepresentasikan variasi ekstrem yang valid dalam data (misalnya, *smartphone* dengan harga sangat tinggi), mungkin lebih baik untuk mempertahankannya, terutama jika model yang digunakan (seperti KNN) relatif robust terhadap *outlier* dalam ruang fitur multidimensi.

Distribusi Kategorikal

- **Implementasi:** Distribusi setiap fitur kategorikal dianalisis menggunakan df[col].value_counts() untuk mendapatkan jumlah kemunculan setiap kategori, dan divisualisasikan menggunakan sns.countplot().
- **Hasil dan Interpretasi:** Hasil value_counts() dan *countplot* memberikan gambaran tentang seberapa seimbang atau tidak seimbang distribusi kategori dalam setiap fitur kategorikal (seperti merek dan merek prosesor). Misalnya, kita dapat melihat merek *smartphone* mana yang paling banyak muncul dalam dataset.
- **Kegunaan:** Memahami distribusi kelas dalam fitur kategorikal penting, terutama jika kita melakukan tugas klasifikasi (yang bukan fokus utama dalam implementasi KNN untuk rekomendasi berbasis fitur ini). Kelas yang tidak seimbang dapat menyebabkan model menjadi bias terhadap kelas mayoritas. Dalam konteks fitur untuk rekomendasi, distribusi yang tidak seimbang dapat berarti bahwa model mungkin lebih cenderung merekomendasikan *smartphone* dari merek atau dengan prosesor yang lebih umum dalam dataset. Teknik seperti *resampling* (oversampling kelas minoritas atau undersampling kelas mayoritas) atau penggunaan bobot kelas dalam algoritma dapat digunakan untuk mengatasi masalah ini jika diperlukan dalam konteks klasifikasi preferensi di masa mendatang.

Secara keseluruhan, langkah-langkah eksplorasi data ini memberikan pemahaman yang komprehensif tentang karakteristik dataset *smartphone*, termasuk distribusi fitur, hubungan antar fitur, keberadaan nilai yang hilang dan *outlier*, serta distribusi kategori. Pemahaman ini mendasari keputusan dalam *preprocessing* data (penanganan nilai yang hilang, normalisasi, *encoding*) dan pemilihan model *machine learning* (KNN yang berbasis jarak).

3.4 Pembangunan Model Machine Learning

1. **Pembagian Data**

sebelum membuat model KNN. Semua data dipakai untuk melatih model. Tujuannya adalah membuat rekomendasi langsung berdasarkan kemiripan fitur HP dengan pilihan pengguna. Model KNN dipakai untuk mencari HP yang paling mirip dengan spesifikasi dari pengguna.

Penting untuk diingat, kalau kita mau mengukur seberapa baik sistem ini bekerja di masa depan, barulah data perlu dibagi dua. Data latihan dipakai membuat model, data uji dipakai untuk mengetes seberapa bagus modelnya menebak dengan data baru. Biasanya, pembagiannya 80% untuk latihan dan 20% untuk uji, tapi ini bisa berubah tergantung jumlah datanya.

Model yang dipakai adalah K-Nearest Neighbors (KNN) karena:

- **Mirip Fiturnya:** KNN mencari kemiripan antar HP berdasarkan fitur-fiturnya (RAM, memori, prosesor, dll.).
- **Mudah Dibuat:** Algoritma KNN gampang dibuat dan dimengerti.
- **Fleksibel:** KNN bisa dipakai untuk mencari barang yang mirip.
- **Tidak Terlalu Kaku:** KNN tidak membuat asumsi aneh tentang data.

Walaupun ada algoritma lain yang bisa dipakai untuk rekomendasi, KNN cocok untuk merekomendasikan HP yang mirip berdasarkan fiturnya.

Model KNN dilatih dengan data yang sudah diolah. Proses "latihan" di KNN sebenarnya hanya menyimpan data dan menentukan cara mengukur kemiripan. Di sistem ini:

- Model KNN dibuat dengan menentukan 5 tetangga terdekat (n_neighbors=5) dan cara mengukur kemiripan menggunakan jarak Euclidean (metric='euclidean').
- Model KNN kemudian "dilatih" menggunakan semua data yang sudah diolah. Di tahap ini, model membuat struktur data agar bisa cepat mencari HP yang mirip nanti.

Untuk saat ini, tidak ada pengaturan lebih lanjut (hyperparameter tuning) pada model KNN. Parameter penting di KNN adalah jumlah tetangga dan cara mengukur jarak. Nilai 5 untuk tetangga dan jarak Euclidean dipilih sebagai pengaturan awal.

Kalau kita mau meningkatkan performa model KNN (terutama kalau kita membagi data jadi latihan dan uji), kita bisa mencoba berbagai pengaturan parameter seperti:

- **Jumlah Tetangga:** Terlalu sedikit bisa membuat model terlalu sensitif, terlalu banyak bisa membuat model kurang spesifik.
- **Cara Mengukur Jarak:** Pilihan cara mengukur jarak (misalnya Euclidean, Manhattan) bisa tergantung pada jenis datanya.
- **Algoritma Pencarian:** Pilihan algoritma pencarian tetangga terdekat bisa mempengaruhi kecepatan, terutama untuk data yang besar.

Namun, karena fokus utama sistem ini adalah memberikan rekomendasi langsung berdasarkan input pengguna, langkah pengaturan parameter ini tidak dilakukan sekarang.

3.5 Evaluasi Model

deskripsikan metrik yang relevan untuk mengevaluasi model *machine learning*, meskipun dalam implementasi sistem rekomendasi KNN saat ini, evaluasi formal menggunakan pembagian data latih dan uji serta metrik-metrik ini tidak secara eksplisit dilakukan. Metrik-metrik berikut umumnya digunakan untuk menilai kinerja model *machine learning*, terutama dalam konteks klasifikasi dan regresi.

Karena tugas utama dalam implementasi ini adalah rekomendasi berdasarkan kemiripan fitur (yang bukan merupakan tugas klasifikasi atau regresi tradisional dengan variabel target yang diprediksi), metrik evaluasi yang paling relevan adalah yang berkaitan dengan kualitas rekomendasi, seperti **Presisi@K** dan **Recall@K** (yang telah disebutkan sebelumnya). Namun, untuk memberikan pemahaman yang komprehensif, mari kita bahas juga metrik-metrik lain yang Anda sebutkan, yang lebih umum digunakan dalam tugas klasifikasi.

Metrik Evaluasi untuk Tugas Klasifikasi (Meskipun Tidak Secara Langsung Diterapkan dalam Kode Rekomendasi KNN Saat Ini)

1. **Akurasi (*Accuracy*)**:
    - **Definisi:** Akurasi adalah proporsi prediksi yang benar dari total prediksi.
    - Akurasi=Total Jumlah PrediksiJumlah Prediksi Benar =TP+TN+FP+FNTP+TN
    - Di mana:
        - TP (*True Positive*) adalah jumlah kasus positif yang diprediksi benar.
        - TN (*True Negative*) adalah jumlah kasus negatif yang diprediksi benar.
        - FP (*False Positive*) adalah jumlah kasus negatif yang diprediksi sebagai positif.
        - FN (*False Negative*) adalah jumlah kasus positif yang diprediksi sebagai negatif.
    - **Penggunaan:** Akurasi adalah metrik yang baik jika kelas dalam data cukup seimbang.
    - **Keterbatasan:** Jika data memiliki kelas yang tidak seimbang (misalnya, satu kelas jauh lebih banyak daripada yang lain), akurasi bisa menyesatkan. Model yang selalu memprediksi kelas mayoritas mungkin memiliki akurasi tinggi tetapi tidak berguna dalam mengidentifikasi kelas minoritas.
2. **Presisi (*Precision*), *Recall* (Sensitivitas), dan F1-Score**:

- **Presisi:** Proporsi prediksi positif yang sebenarnya positif. Ini menjawab pertanyaan: "Dari semua yang saya prediksi sebagai positif, berapa banyak yang benar-benar positif?"
    - Presisi=TP+FPTP
- **Recall (Sensitivitas atau *True Positive Rate*)**: Proporsi aktual positif yang diprediksi dengan benar. Ini menjawab pertanyaan: "Dari semua yang sebenarnya positif, berapa banyak yang berhasil saya prediksi?"
    - Recall=TP+FNTP
- **F1-Score:** Rata-rata harmonik dari presisi dan *recall*. Ini memberikan ukuran tunggal yang menyeimbangkan presisi dan *recall*. F1-score sangat berguna ketika ada ketidakseimbangan kelas.
    - F1-Score=2×Presisi+RecallPresisi×Recall
- **Penggunaan:** Metrik-metrik ini sangat penting ketika kelas tidak seimbang atau ketika biaya *false positive* dan *false negative* berbeda. Misalnya, dalam deteksi penyakit, *recall* yang tinggi mungkin lebih penting untuk memastikan tidak ada kasus positif yang terlewatkan, meskipun dengan mengorbankan presisi yang lebih rendah. F1-score memberikan keseimbangan antara keduanya.

3. **Confusion Matrix (Matriks Konfusi)**:
    - **Deskripsi:** *Confusion matrix* adalah tabel yang merangkum kinerja model klasifikasi dengan menampilkan jumlah prediksi yang benar dan salah untuk setiap kelas. Untuk klasifikasi biner, matriks ini berukuran 2x2 dan berisi TP, TN, FP, dan FN. Untuk klasifikasi multikelas, ukurannya NxN, di mana N adalah jumlah kelas.
    - **Penggunaan:** *Confusion matrix* memberikan gambaran yang lebih rinci tentang jenis kesalahan yang dibuat oleh model (misalnya, kesalahan memprediksi kelas positif sebagai negatif atau sebaliknya). Ini membantu dalam memahami pola kesalahan model dan di mana model perlu ditingkatkan.

4. **Kurva ROC (*Receiver Operating Characteristic*) dan AUC (*Area Under the Curve*)**:

    - **Penggunaan:** Metrik ini digunakan untuk mengevaluasi kinerja model klasifikasi biner, terutama ketika model menghasilkan probabilitas untuk prediksi kelas.

    - **Kurva ROC:** Grafik yang memplot *True Positive Rate* (TPR atau *recall*) terhadap *False Positive Rate* (FPR) pada berbagai ambang batas klasifikasi. FPR didefinisikan sebagai:

        - FPR=FP+TNFP

    - Kurva ROC menunjukkan *trade-off* antara sensitivitas (kemampuan untuk mendeteksi semua kasus positif) dan spesifisitas (kemampuan untuk menghindari klasifikasi kasus negatif sebagai positif) model.

    - **AUC:** Area di bawah kurva ROC. Nilai AUC berkisar antara 0 dan 1. AUC yang lebih tinggi menunjukkan kinerja model yang lebih baik dalam membedakan antara kelas positif dan negatif. AUC sebesar 0.5 menunjukkan kinerja yang tidak lebih baik dari tebakan acak, sedangkan AUC mendekati 1 menunjukkan model yang hampir sempurna.

Metrik Evaluasi yang Lebih Relevan untuk Sistem Rekomendasi KNN

Seperti yang disebutkan sebelumnya, untuk sistem rekomendasi, metrik seperti **Presisi@K** dan **Recall@K** lebih relevan. Dalam konteks rekomendasi *smartphone*:

1. **Akurasi (Ketepatan):** Seberapa sering model menebak dengan benar dari semua tebakan. Bagus kalau datanya seimbang. Tapi kalau satu jenis data jauh lebih banyak, akurasi bisa menipu.
2. **Presisi (Ketelitian), Recall (Kepekaan), dan F1-Score:**
    - **Presisi:** Dari semua yang model tebak positif, berapa yang benar-benar positif?
    - **Recall:** Dari semua yang sebenarnya positif, berapa yang berhasil ditebak positif oleh model?
    - **F1-Score:** Gabungan presisi dan recall. Berguna kalau datanya tidak seimbang.
3. **Confusion Matrix (Matriks Kebingungan):** Tabel yang menunjukkan berapa kali model benar dan salah dalam menebak setiap kategori. Bantu kita lihat jenis kesalahan yang sering dibuat model.
4. **Kurva ROC dan AUC:** Dipakai untuk model yang menghasilkan probabilitas.
    - **Kurva ROC:** Grafik yang menunjukkan keseimbangan antara kemampuan mendeteksi yang positif dan menghindari salah menebak yang negatif.

- **AUC:** Luas area di bawah kurva ROC. Makin tinggi nilainya (mendekati 1), makin bagus modelnya.

Metrik yang Lebih Cocok untuk Sistem Rekomendasi KNN (Walaupun Belum Dievaluasi)

Untuk rekomendasi, yang lebih penting adalah:

- **Presisi@K:** Dari sejumlah rekomendasi teratas (K), berapa banyak yang sebenarnya relevan bagi pengguna?
- **Recall@K:** Dari semua hal yang relevan bagi pengguna, berapa banyak yang berhasil direkomendasikan di urutan atas (K)?

Ada juga metrik lain seperti MAP dan NDCG yang mempertimbangkan urutan rekomendasi.

3.6 Pertanyaan Bisnis yang Dieksplorasi

Preferensi Fitur dan Merek: Fitur-*smartphone* mana (misalnya, RAM, memori internal, kapasitas baterai, harga) yang paling bervariasi antar merek yang berbeda? Apakah ada merek yang secara konsisten menawarkan spesifikasi tertentu pada rentang harga tertentu?

Tujuan Bisnis: Memahami lanskap kompetitif dan bagaimana berbagai merek memposisikan produk mereka dalam hal fitur dan harga. Informasi ini dapat membantu produsen atau peritel dalam membuat keputusan strategis terkait penetapan harga, pengembangan produk, dan penargetan pasar.

Pengelompokan *Smartphone* Berdasarkan Fitur: Dapatkah kita mengidentifikasi kelompok-*smartphone* yang berbeda berdasarkan kemiripan fitur teknis mereka (tanpa mempertimbangkan merek)? Jika ya, fitur apa yang paling membedakan kelompok-kelompok ini?

Tujuan Bisnis: Mengidentifikasi segmen pasar potensial berdasarkan preferensi fitur. Ini dapat membantu dalam personalisasi rekomendasi (misalnya, merekomendasikan *smartphone* dari kelompok yang sama dengan yang disukai pengguna di masa lalu) atau dalam memahami tren pasar secara umum.

Prediksi Harga Berdasarkan Spesifikasi: Seberapa baik kita dapat memprediksi harga *smartphone* baru berdasarkan spesifikasi teknisnya? Fitur mana yang memiliki pengaruh paling signifikan terhadap harga?

Tujuan Bisnis: Membangun model prediksi harga dapat berguna untuk analisis kompetitif (membandingkan harga *smartphone* dengan spesifikasi serupa), deteksi anomali harga, atau bahkan membantu dalam penetapan harga produk baru.

Rekomendasi yang Dipersonalisasi Berdasarkan Riwayat Interaksi (Jika Data Tersedia): Jika kita memiliki data tentang interaksi pengguna (misalnya, *smartphone* yang dilihat, dibeli, atau diberi peringkat), bagaimana kita dapat membangun sistem rekomendasi yang lebih personal untuk setiap pengguna berdasarkan preferensi historis mereka? Apakah *Collaborative Filtering* atau pendekatan berbasis konten lebih efektif dalam kasus ini?

Tujuan Bisnis: Meningkatkan keterlibatan pengguna, meningkatkan penjualan, dan memberikan pengalaman berbelanja yang lebih memuaskan dengan merekomendasikan *smartphone* yang paling mungkin menarik bagi setiap individu.

Identifikasi *Smartphone* dengan Nilai Terbaik (*Value for Money*): Berdasarkan spesifikasi dan harga, *smartphone* mana yang menawarkan nilai terbaik? Bagaimana kita mendefinisikan "nilai terbaik" (misalnya, rasio antara fitur-fitur penting dan harga)?

Tujuan Bisnis: Membantu konsumen membuat keputusan pembelian yang lebih informasi dengan mengidentifikasi produk yang menawarkan fitur terbanyak untuk harga tertentu. Ini juga dapat membantu peritel dalam mempromosikan produk dengan nilai jual yang kuat.

Meskipun implementasi saat ini lebih fokus pada pertanyaan pertama dan sebagian dari pertanyaan kedua (melalui rekomendasi berbasis kemiripan fitur dengan KNN), pertanyaan-pertanyaan ini memberikan arah yang lebih luas untuk potensi analisis dan pengembangan model *machine learning* lebih lanjut dengan *dataset smartphone* ini. Jika data interaksi pengguna tersedia di masa depan, pertanyaan keempat dapat dieksplorasi lebih lanjut untuk membangun sistem rekomendasi yang lebih canggih.

IV. Hasil dan Pembahasan

4.1 Hasil Eksplorasi Data

1. Distribusi Data Fitur Numerik

- **Visualisasi:** Histogram dari setiap fitur numerik (seperti ram_capacity, internal_memory, battery_capacity, dan price) telah ditampilkan menggunakan df[numerical_cols].hist().
- **Analisis:**
    - Distribusi beberapa fitur numerik menunjukkan variasi yang signifikan. Misalnya, harga (*price*) tampak memiliki rentang yang lebar dan mungkin miring ke kanan (lebih banyak *smartphone* dengan harga rendah hingga menengah).
    - Kapasitas RAM dan memori internal juga menunjukkan distribusi yang beragam, mencerminkan berbagai segmen pasar *smartphone*.
    - Distribusi kapasitas baterai juga bervariasi, dengan beberapa puncak yang mungkin menunjukkan standar kapasitas baterai yang umum.
- **Wawasan yang Diperoleh:**
    - Beberapa fitur numerik mungkin tidak terdistribusi normal. Ini perlu dipertimbangkan jika model yang lebih sensitif terhadap asumsi normalitas digunakan di masa mendatang (meskipun KNN tidak memiliki asumsi ini).
    - Perbedaan skala antar fitur numerik (misalnya, RAM dalam GB vs. harga dalam mata uang yang lebih besar) membenarkan langkah normalisasi yang telah dilakukan sebelum melatih model KNN.

2. Heatmap Korelasi Fitur Numerik

- **Visualisasi:** Heatmap korelasi antar fitur numerik telah ditampilkan menggunakan sns.heatmap(df[numerical_cols].corr()).
- **Analisis:**

- Heatmap menunjukkan adanya korelasi positif antara beberapa fitur numerik. Misalnya, kemungkinan ada korelasi positif antara RAM dan harga, serta antara memori internal dan harga. Ini masuk akal karena *smartphone* dengan spesifikasi lebih tinggi cenderung memiliki harga yang lebih mahal.

- Kekuatan korelasi bervariasi antar pasangan fitur. Beberapa korelasi tampak cukup kuat (warna lebih intens), sementara yang lain lemah (warna mendekati netral).

- **Wawasan yang Diperoleh:**

  - Tidak ada korelasi antar fitur numerik yang sangat tinggi (mendekati 1 atau -1) yang akan menyebabkan masalah multikolinearitas yang signifikan untuk model berbasis jarak seperti KNN. Namun, korelasi yang ada memberikan pemahaman tentang hubungan antar spesifikasi *smartphone*.

  - Jika kita bertujuan untuk memprediksi harga di masa depan, fitur-fitur yang berkorelasi tinggi dengan harga akan menjadi kandidat yang baik untuk dimasukkan dalam model regresi.

3. Boxplot untuk Outliers

- **Visualisasi:** *Boxplot* untuk kolom 'price' telah ditampilkan menggunakan sns.boxplot(x=df['price']). *Boxplot* juga ditampilkan secara implisit untuk fitur numerik lainnya melalui df[numerical_cols].boxplot().

- **Analisis:**

  - *Boxplot* untuk harga menunjukkan adanya beberapa titik data di luar *whisker*, yang mengindikasikan potensi *outlier* (harga yang sangat tinggi dibandingkan dengan sebagian besar *smartphone* dalam *dataset*).

  - *Boxplot* untuk fitur numerik lainnya (RAM, memori internal, baterai) juga mungkin menunjukkan beberapa *outlier*, tergantung pada variasi dalam spesifikasi *smartphone* dalam *dataset*.

- **Wawasan yang Diperoleh:**

- Potensi *outlier* terdeteksi, terutama pada fitur harga. Keputusan untuk menangani *outlier* ini (menghapus, mengoreksi, atau membiarkannya) bergantung pada pemahaman apakah *outlier* ini merupakan kesalahan data atau memang representasi dari *smartphone* dengan spesifikasi atau harga yang ekstrem namun valid.

- Dalam implementasi KNN saat ini, *outlier* tidak ditangani secara eksplisit. Model KNN relatif robust terhadap *outlier* dalam ruang fitur multidimensi karena rekomendasi didasarkan pada mayoritas tetangga terdekat. Namun, *outlier* yang ekstrem masih dapat memengaruhi skala fitur dan jarak.

Wawasan Umum yang Diperoleh dari Eksplorasi Data:

- **Variasi Fitur:** Terdapat variasi yang signifikan dalam fitur-fitur numerik *smartphone*, yang menunjukkan keragaman produk di pasar.

- **Korelasi yang Masuk Akal:** Korelasi antar fitur numerik umumnya sesuai dengan ekspektasi (misalnya, spesifikasi lebih tinggi cenderung berkorelasi dengan harga lebih tinggi).

- **Potensi Outlier:** Keberadaan potensi *outlier*, terutama pada harga, perlu dipertimbangkan dalam analisis atau pemodelan lebih lanjut.

- **Distribusi Tidak Normal:** Beberapa fitur numerik mungkin tidak terdistribusi normal, yang relevan untuk pemilihan atau asumsi model di masa depan.

Presentasi hasil eksplorasi data ini memberikan pemahaman awal tentang karakteristik *dataset smartphone* dan membantu memvalidasi langkah-langkah *preprocessing* yang telah dilakukan (seperti normalisasi) dan pemilihan model (KNN yang berbasis jarak).

4.2 Hasil Model Machine Learning

menggunakan algoritma K-Nearest Neighbors (KNN), dan tidak melibatkan tugas klasifikasi atau regresi tradisional dengan variabel target yang diprediksi, metrik evaluasi seperti akurasi, F1-score, AUC, *precision-recall*, dan *confusion matrix* **tidak secara langsung dievaluasi** dalam kode yang diberikan.

Model KNN yang digunakan di sini berfungsi untuk menemukan *k smartphone* terdekat berdasarkan input fitur pengguna. "Kinerja" model dalam konteks ini lebih berkaitan dengan relevansi dan kualitas rekomendasi yang diberikan kepada pengguna. Metrik evaluasi untuk sistem rekomendasi (seperti Presisi@K, Recall@K, dan NDCG) akan lebih tepat untuk mengukur seberapa baik model ini bekerja dalam merekomendasikan *smartphone* yang sesuai dengan preferensi pengguna.

Namun, jika kita membayangkan skenario di mana kita ingin mengevaluasi model KNN ini dalam tugas klasifikasi hipotetis (misalnya, memprediksi kategori harga *smartphone* berdasarkan fitur-fiturnya), maka metrik-metrik yang Anda sebutkan akan relevan.

Evaluasi Model dalam Konteks Rekomendasi (Meskipun Tidak Ada Implementasi Evaluasi Formal dalam Kode Saat Ini)

Untuk mengevaluasi sistem rekomendasi KNN ini secara lebih formal, kita memerlukan data interaksi pengguna (misalnya, *smartphone* mana yang disukai, dibeli, atau diberi peringkat oleh pengguna). Dengan data tersebut, kita dapat:

Membagi Data: Membagi data menjadi data pelatihan dan data pengujian. Data pelatihan digunakan untuk membangun model KNN, dan data pengujian digunakan untuk mengevaluasi rekomendasi untuk pengguna yang tidak dilihat selama pelatihan.

Definisikan Relevansi: Menentukan apa yang dianggap sebagai rekomendasi yang "relevan" (misalnya, *smartphone* yang dibeli atau diberi peringkat tinggi oleh pengguna dalam data pengujian).

Hitung Metrik Rekomendasi:

Presisi@K: Untuk setiap pengguna dalam data pengujian, hitung proporsi *K* rekomendasi teratas yang relevan. Rata-ratakan nilai ini di seluruh pengguna.

Recall@K: Untuk setiap pengguna, hitung proporsi item relevan dalam data pengujian yang berhasil masuk dalam *K* rekomendasi teratas. Rata-ratakan nilai ini di seluruh pengguna.

NDCG@K (Normalized Discounted Cumulative Gain): Metrik yang mempertimbangkan relevansi item dan posisi mereka dalam daftar rekomendasi. Item yang lebih relevan dan muncul lebih awal dalam daftar mendapatkan skor yang lebih tinggi.

Perbandingan Model (Jika Beberapa Model Dicoba)

Jika kita mencoba beberapa algoritma rekomendasi (misalnya, KNN dengan metrik jarak yang berbeda, model berbasis konten lainnya), kita dapat membandingkan kinerja mereka menggunakan metrik rekomendasi di atas. Tabel atau grafik dapat digunakan untuk memvisualisasikan perbandingan ini.

Contoh Tabel Perbandingan Model (Hipotetis):

Model	**Presisi@5**	**Recall@5**	**NDCG@5**
KNN (Euclidean)	0.75	0.60	0.82
KNN (Cosine)	0.70	0.55	0.78
Content-Based	0.65	0.50	0.72

Ekspor ke Spreadsheet

Dalam tabel hipotetis ini, KNN dengan metrik Euclidean menunjukkan kinerja terbaik berdasarkan metrik yang dievaluasi.

Kesimpulan (Dalam Konteks Rekomendasi)

Karena implementasi saat ini adalah sistem rekomendasi berbasis fitur *ad-hoc* dan tidak melibatkan prediksi atau klasifikasi eksplisit, evaluasi kinerja model akan lebih tepat dilakukan dengan mengukur kualitas dan relevansi rekomendasi yang diberikan kepada pengguna berdasarkan preferensi fitur mereka. Jika data interaksi pengguna tersedia, evaluasi formal menggunakan metrik rekomendasi seperti Presisi@K, Recall@K, dan NDCG dapat dilakukan untuk mengukur kinerja sistem secara kuantitatif.

.

4.3 Pembahasan

Model yang Dibangun

Model yang dibangun adalah sistem rekomendasi *smartphone* berbasis kemiripan fitur menggunakan algoritma K-Nearest Neighbors (KNN). Tujuan utama dari tugas besar ini adalah untuk memberikan rekomendasi *smartphone* kepada pengguna berdasarkan spesifikasi yang mereka inginkan.

Dalam konteks ini, "keberhasilan" model tidak diukur dengan akurasi prediksi atau klasifikasi tradisional, melainkan dengan kemampuannya untuk mengidentifikasi dan merekomendasikan *smartphone* yang memiliki fitur serupa dengan input pengguna. Berdasarkan demonstrasi kode, model KNN berhasil menerima input spesifikasi dari pengguna dan menghasilkan daftar *smartphone* teratas yang paling mirip dalam ruang fitur yang ditentukan.

Mengenai *overfitting* atau *underfitting*, konsep ini lebih relevan untuk model prediktif. Dalam kasus KNN untuk rekomendasi berbasis kemiripan, tidak ada proses pelatihan model dalam arti tradisional yang dapat menyebabkan *overfitting* pada data latih. Model hanya menyimpan data dan menggunakan metrik jarak untuk menemukan tetangga terdekat. Namun, pemilihan jumlah tetangga (*k*) dapat memengaruhi hasil rekomendasi. Nilai *k* yang terlalu kecil mungkin menghasilkan rekomendasi yang terlalu spesifik dan sensitif terhadap *noise* dalam data, sementara nilai *k* yang terlalu besar dapat menghasilkan rekomendasi yang terlalu umum dan kurang relevan. Dalam implementasi ini, *k* diatur ke 5.

Kelebihan dan Kekurangan Model KNN untuk Rekomendasi

- **Kelebihan:**
    - **Sederhana dan Mudah Diimplementasikan:** Algoritma KNN relatif mudah dipahami dan diimplementasikan.
    - **Non-Parametrik:** Tidak membuat asumsi yang kuat tentang distribusi data yang mendasarinya.
    - **Fleksibel:** Dapat digunakan untuk berbagai jenis data dan metrik jarak.

- **Interpretasi Lokal:** Hasil rekomendasi dapat diinterpretasikan dengan melihat fitur-fitur dari *k* tetangga terdekat. Dalam kode, visualisasi perbandingan fitur antara input pengguna dan rekomendasi membantu dalam interpretasi.

- **Responsif terhadap Data Baru:** Model secara alami memasukkan data baru tanpa perlu pelatihan ulang yang rumit (hanya perlu ditambahkan ke dalam struktur data).

- **Kekurangan:**

  - **Sensitif terhadap Skala Fitur:** Jarak Euclidean yang digunakan dalam implementasi ini sensitif terhadap skala fitur. Oleh karena itu, normalisasi fitur numerik menjadi langkah penting.

  - **Mahal secara Komputasi untuk *Dataset* Besar:** Mencari tetangga terdekat dalam *dataset* yang sangat besar bisa menjadi mahal secara komputasi, baik dari segi waktu maupun memori.

  - **Masalah Dimensi Tinggi:** Kinerja KNN dapat menurun dalam ruang fitur dengan dimensi yang sangat tinggi (*curse of dimensionality*).

  - **Pemilihan *k* yang Optimal:** Pemilihan jumlah tetangga (*k*) yang optimal bisa menjadi tantangan dan mungkin memerlukan eksperimen atau validasi.

  - **Tidak Menangani "Cold Start" dengan Baik:** Jika ada *smartphone* baru dalam *dataset* dengan sedikit atau tanpa interaksi pengguna (jika kita mempertimbangkan rekomendasi berbasis interaksi di masa depan), KNN berbasis kemiripan fitur masih dapat memberikan rekomendasi. Namun, untuk sistem rekomendasi berbasis pengguna, pengguna baru tanpa riwayat interaksi menjadi masalah "cold start".

Potensi Perbaikan Model

Meskipun model KNN berbasis fitur ini berfungsi untuk memberikan rekomendasi berdasarkan kemiripan spesifikasi, ada beberapa potensi perbaikan dan perluasan yang dapat dipertimbangkan:

1. **Pemilihan Fitur yang Lebih Cerdas:** Pemilihan fitur untuk rekomendasi saat ini didasarkan pada intuisi. Analisis lebih lanjut, seperti *feature importance* dari model prediktif (jika kita mencoba memprediksi preferensi atau rating), atau teknik seleksi fitur lainnya dapat digunakan untuk memilih subset fitur yang lebih relevan.

2. **Penggunaan Metrik Jarak yang Berbeda:** Metrik jarak Euclidean mungkin tidak selalu yang terbaik untuk semua jenis fitur. Eksperimen dengan metrik lain seperti jarak Manhattan, jarak Cosine (terutama berguna untuk data dengan magnitudo yang berbeda), atau metrik khusus domain mungkin meningkatkan kualitas rekomendasi.

3. **Pembobotan Fitur:** Tidak semua fitur mungkin memiliki kepentingan yang sama dalam menentukan kemiripan. Memberikan bobot yang berbeda pada fitur-fitur berdasarkan relevansinya (misalnya, berdasarkan preferensi pengguna yang dianalisis dari data historis) dapat meningkatkan personalisasi.

4. **Penanganan Fitur Kategorikal yang Lebih Canggih:** *One-Hot Encoding* yang digunakan saat ini dapat menghasilkan ruang fitur yang sangat tinggi jika ada banyak kategori unik. Teknik *encoding* yang lebih canggih atau pendekatan berbasis jarak untuk fitur kategorikal dapat dipertimbangkan.

5. **Peningkatan Efisiensi untuk *Dataset* Besar:** Untuk *dataset* yang sangat besar, penggunaan struktur data yang lebih efisien untuk pencarian tetangga terdekat, seperti KD-tree atau Ball-tree (yang dapat dipilih dalam parameter NearestNeighbors), atau bahkan pendekatan berbasis aproksimasi tetangga terdekat (*Approximate Nearest Neighbors*) dapat meningkatkan kinerja.

6. **Hybrid Recommendation Systems:** Menggabungkan pendekatan berbasis fitur (seperti KNN ini) dengan pendekatan lain seperti *Collaborative Filtering* (jika data interaksi pengguna tersedia) atau model berbasis konten lainnya dapat menghasilkan sistem rekomendasi yang lebih kuat dan mengatasi beberapa keterbatasan masing-masing pendekatan.

7. **Personalisasi Lebih Lanjut:** Jika data pengguna tersedia, model dapat dipersonalisasi lebih lanjut dengan mempertimbangkan riwayat preferensi, demografi, atau informasi kontekstual lainnya.

Secara keseluruhan, model KNN berbasis fitur ini merupakan langkah awal yang baik untuk sistem rekomendasi *smartphone* berdasarkan spesifikasi.

4.4 Insight Pertanyaan Bisnis

Insight dari Pertanyaan Bisnis:

- **Faktor utama yang mempengaruhi variabel target:** Karena tugas utama di sini adalah rekomendasi berdasarkan kemiripan fitur, tidak ada variabel target tunggal dalam pengertian prediksi atau klasifikasi tradisional. Namun, jika kita mempertimbangkan *kemiripan* sebagai "target" implisit, maka faktor-faktor (fitur) yang paling mempengaruhi kemiripan antar *smartphone* adalah fitur-fitur yang digunakan dalam perhitungan jarak KNN:

    - **Fitur Numerik:** ram_capacity, internal_memory, battery_capacity, dan price. Skala fitur-fitur ini menjadi penting karena metrik Euclidean sensitif terhadap perbedaan nilai yang besar. Normalisasi memastikan bahwa semua fitur numerik berkontribusi secara proporsional terhadap perhitungan jarak.

    - **Fitur Kategorikal:** processor_brand dan brand_name. Melalui *one-hot encoding*, setiap kategori unik dalam fitur-fitur ini menjadi dimensi biner. Kemiripan dalam fitur kategorikal diukur berdasarkan apakah dua *smartphone* memiliki nilai yang sama untuk kategori tersebut.

- **Pola yang dapat digunakan untuk memprediksi hasil:** Model KNN mengidentifikasi pola kemiripan dalam ruang fitur *smartphone*. Pola yang digunakan untuk "memprediksi" rekomendasi adalah kedekatan (*nearness*) dalam ruang fitur. Jika input pengguna dekat dengan sekelompok *smartphone* dalam ruang fitur (berdasarkan metrik jarak), maka *smartphone* tersebut dianggap sebagai rekomendasi yang baik karena memiliki kombinasi fitur yang serupa.

- **Penanganan data yang tidak seimbang:** Konsep data yang tidak seimbang biasanya relevan dalam tugas klasifikasi di mana jumlah sampel untuk setiap kelas berbeda secara signifikan. Dalam kasus rekomendasi berbasis kemiripan fitur ini, tidak ada kelas target yang diprediksi. Namun, jika kita mempertimbangkan distribusi nilai dalam setiap fitur, kita melihat adanya variasi (seperti yang ditunjukkan dalam histogram). Tidak ada penanganan eksplisit untuk "ketidakseimbangan" dalam distribusi fitur yang diterapkan dalam *preprocessing* KNN ini. Jika kita beralih ke tugas klasifikasi (misalnya, memprediksi kategori harga), dan kelas harga tidak seimbang, teknik *resampling* atau penyesuaian bobot kelas mungkin diperlukan.

- **Perbaikan dengan regularisasi:** Regularisasi adalah teknik yang digunakan untuk mencegah *overfitting* dalam model prediktif (seperti regresi linear atau jaringan saraf tiruan) dengan menambahkan penalti pada kompleksitas model. Karena KNN adalah algoritma berbasis instans dan tidak melibatkan proses pelatihan parameter seperti model parametrik, regularisasi tidak secara langsung diterapkan atau relevan dalam konteks ini. Potensi "overfitting" dalam KNN lebih terkait dengan pemilihan nilai *k* (terlalu kecil).

- **Potensi peningkatan model:** Berdasarkan hasil dan diskusi sebelumnya, ada beberapa potensi peningkatan model rekomendasi KNN ini:

    - **Feature Engineering Lebih Lanjut:** Membuat fitur baru dari yang sudah ada (misalnya, rasio harga terhadap kinerja yang diukur oleh kombinasi RAM dan kecepatan prosesor, jika data kecepatan prosesor tersedia).

    - **Pembobotan Fitur:** Memberikan bobot yang berbeda pada fitur berdasarkan preferensi pengguna (jika data historis tersedia) atau berdasarkan analisis kepentingan fitur.

    - **Penggunaan Metrik Jarak yang Lebih Canggih:** Mencoba metrik jarak lain yang mungkin lebih sesuai untuk data campuran numerik dan kategorikal, atau metrik yang dapat menangani perbedaan skala dan korelasi antar fitur dengan lebih baik (misalnya, metrik Mahalanobis jika ada korelasi signifikan).

    - **Penanganan *Outlier*:** Mengevaluasi dampak *outlier* pada rekomendasi dan menerapkan teknik penanganan *outlier* jika diperlukan.

    - **Integrasi dengan Metode Lain:** Menggabungkan rekomendasi berbasis fitur dengan pendekatan *collaborative filtering* atau model berbasis konten lainnya untuk menciptakan sistem rekomendasi hibrida yang lebih kuat.

4.5 Action Plan Bisnis

Action Plan Bisnis untuk Sistem Rekomendasi Smartphone Berbasis Fitur

1. Rekomendasi Strategi Berdasarkan Temuan Data:

- Segmentasi Berdasarkan Preferensi Fitur: Analisis eksplorasi data menunjukkan adanya variasi yang signifikan dalam fitur-fitur *smartphone* dan harganya antar merek. Strategi pemasaran dapat menargetkan segmen pengguna yang berbeda berdasarkan preferensi fitur mereka. Misalnya:

    - Pengguna dengan anggaran terbatas mungkin lebih tertarik pada *smartphone* dengan RAM dan memori internal yang cukup untuk penggunaan sehari-hari, tanpa memerlukan spesifikasi tertinggi. Pemasaran dapat menyoroti *smartphone* dengan nilai terbaik dalam segmen ini.

    - Pengguna yang fokus pada performa tinggi (misalnya, untuk bermain game atau penggunaan intensif) akan mencari *smartphone* dengan RAM besar dan prosesor canggih. Pemasaran dapat menargetkan segmen ini dengan menonjolkan fitur-fitur tersebut.

    - Pengguna yang loyal terhadap merek tertentu mungkin merespons promosi atau rekomendasi dalam ekosistem merek tersebut.

- Penekanan pada Fitur Utama: Model KNN mengandalkan semua fitur yang dimasukkan untuk menentukan kemiripan. Dalam komunikasi pemasaran, penting untuk menyoroti fitur-fitur yang paling relevan bagi segmen target. Misalnya, daya tahan baterai untuk pengguna yang aktif, atau kualitas kamera untuk penggemar fotografi.

2. Penggunaan Model dalam Proses Bisnis:

- Integrasi ke Platform Penjualan: Model rekomendasi KNN dapat diintegrasikan langsung ke platform penjualan *online* atau *offline*. Ketika pengguna mencari atau melihat *smartphone* tertentu, sistem dapat secara otomatis merekomendasikan *smartphone* serupa berdasarkan fiturnya.

- Personalisasi Pengalaman Pengguna: Dengan memungkinkan pengguna memasukkan preferensi fitur mereka, model memberikan pengalaman yang dipersonalisasi. Ini dapat meningkatkan kepuasan pengguna dan kemungkinan penemuan produk yang relevan.

- Bantuan Penjualan di Toko: Pramuniaga di toko fisik dapat menggunakan antarmuka yang didukung oleh model ini untuk membantu pelanggan menemukan *smartphone* yang sesuai dengan kebutuhan dan anggaran mereka berdasarkan preferensi yang mereka sebutkan.

- Kampanye Pemasaran Bertarget: Data input pengguna (jika disimpan secara anonim dan dengan izin) dapat digunakan untuk memahami tren preferensi fitur dan menargetkan kampanye pemasaran ke kelompok pengguna dengan minat serupa.

3. Peningkatan Data dan Model:

- Pengumpulan Data Tambahan: Untuk meningkatkan kualitas rekomendasi di masa depan, pengumpulan data tambahan dapat dipertimbangkan:

    - Data preferensi pengguna secara implisit (misalnya, *smartphone* yang dilihat, ditambahkan ke keranjang, atau dibeli).

    - Data ulasan dan peringkat pengguna untuk memahami aspek mana yang paling mereka hargai.

    - Data spesifikasi teknis yang lebih detail (misalnya, kecepatan prosesor, resolusi kamera, jenis layar).

- Feature Engineering: Membuat fitur baru yang mungkin lebih relevan untuk rekomendasi, seperti rasio harga/kinerja (jika data kinerja tersedia).

- Eksperimen dengan Model Lain: Pertimbangkan untuk mencoba algoritma rekomendasi lain, seperti model berbasis konten yang lebih canggih (misalnya, menggunakan representasi vektor fitur yang dipelajari dari deskripsi produk) atau *collaborative filtering* (jika data interaksi pengguna tersedia).

4. Optimisasi Proses Bisnis:

- Validasi Input Pengguna: Memastikan bahwa input pengguna valid dan dalam rentang yang masuk akal untuk menghindari rekomendasi yang tidak relevan.

- Penanganan Data Kategorikal: Jika jumlah merek atau tipe prosesor sangat banyak, pertimbangkan teknik *encoding* yang lebih efisien untuk menghindari dimensi tinggi dalam model KNN.

- Skalabilitas: Jika *dataset smartphone* terus bertambah, pertimbangkan penggunaan struktur data yang lebih efisien untuk pencarian tetangga terdekat.

5. Pengukuran dan Evaluasi Kinerja Bisnis:

- KPI untuk Sistem Rekomendasi:

    - Click-Through Rate (CTR): Persentase rekomendasi yang diklik oleh pengguna.

    - Conversion Rate: Persentase pengguna yang melakukan pembelian setelah melihat rekomendasi.

    - User Engagement: Seberapa sering pengguna berinteraksi dengan fitur rekomendasi.

    - Relevance of Recommendations: Survei pengguna atau analisis implisit (misalnya, waktu yang dihabiskan untuk melihat produk yang direkomendasikan) dapat memberikan wawasan tentang relevansi rekomendasi.

- Evaluasi Model (Jika Data Interaksi Tersedia): Jika data interaksi pengguna dikumpulkan, evaluasi model secara formal menggunakan metrik seperti Presisi@K, Recall@K, dan NDCG untuk mengukur kualitas rekomendasi.

- A/B Testing: Melakukan A/B testing dengan berbagai konfigurasi model (misalnya, jumlah tetangga *k* yang berbeda, metrik jarak yang berbeda) atau strategi rekomendasi untuk melihat mana yang menghasilkan kinerja bisnis terbaik**.**

V. Kesimpulan dan Saran

5.1 Kesimpulan

Analisis data awal (EDA) memberikan pemahaman mendalam tentang dataset smartphone, meliputi distribusi fitur, korelasi antar fitur, keberadaan nilai yang hilang dan *outlier*, serta distribusi kategori. Pemahaman ini menjadi dasar penting dalam melakukan *preprocessing* data, termasuk penanganan nilai yang hilang dengan penghapusan baris terkait fitur rekomendasi, normalisasi fitur numerik menggunakan MinMaxScaler, dan *encoding* fitur kategorikal menggunakan OneHotEncoder melalui ColumnTransformer.

Model *machine learning* yang dipilih untuk sistem rekomendasi ini adalah K-Nearest Neighbors (KNN). KNN dipilih karena kesederhanaannya, kemampuannya dalam mengukur kemiripan antar *smartphone* berdasarkan fitur, fleksibilitas terhadap jenis data numerik dan kategorikal, sifatnya yang non-parametrik, dan kemampuannya dalam memberikan rekomendasi *ad-hoc* berdasarkan preferensi pengguna tanpa memerlukan fase pelatihan yang rumit untuk setiap pengguna baru.

Meskipun implementasi saat ini tidak melakukan evaluasi formal dengan pembagian data latih dan uji, konsep metrik evaluasi yang relevan tetap dibahas. Untuk tugas rekomendasi berbasis kemiripan fitur ini, metrik seperti Presisi@K dan Recall@K menjadi yang paling sesuai apabila tersedia data interaksi pengguna historis. Selain itu, metrik evaluasi umum untuk tugas klasifikasi seperti Akurasi, Presisi, Recall, F1-Score, *Confusion Matrix*, Kurva ROC, dan AUC juga dijelaskan untuk memberikan konteks yang lebih luas mengenai evaluasi model *machine learning*.

5.2 Saran

1. sistem rekomendasi secara objektif dan memungkinkannya untuk ditingkatkan di masa depan, disarankan untuk mengimplementasikan evaluasi formal. Ini melibatkan:
    - Pengumpulan data interaksi pengguna historis (misalnya, *smartphone* mana yang dilihat, disukai, atau dibeli setelah mendapatkan rekomendasi).
    - Pembagian dataset menjadi data latih dan data uji.
    - Penggunaan metrik evaluasi yang relevan untuk rekomendasi seperti Presisi@K, Recall@K, Mean Average Precision (MAP), atau Normalized Discounted Cumulative Gain (NDCG) pada data uji untuk mengukur kualitas rekomendasi.
2. **Eksplorasi dan Pemilihan Fitur Lebih Lanjut:** Pemilihan fitur saat ini dilakukan secara manual berdasarkan asumsi. Di masa depan, disarankan untuk:

- Menganalisis lebih lanjut relevansi setiap fitur terhadap preferensi pengguna (jika data interaksi pengguna tersedia).
- Mencoba teknik pemilihan fitur otomatis (misalnya, berdasarkan korelasi dengan variabel target jika ada, atau menggunakan metode *feature importance* dari model lain).
- Mempertimbangkan untuk memasukkan fitur lain yang mungkin relevan dari dataset yang ada.

3. **Hyperparameter Tuning:** Parameter model KNN saat ini (jumlah tetangga *k* dan metrik jarak) ditetapkan pada nilai *default* atau pilihan awal. Untuk meningkatkan kinerja model, disarankan untuk melakukan *hyperparameter tuning* menggunakan teknik seperti *cross-validation* pada data latih (setelah pembagian data implementasikan). Mencoba berbagai nilai *k* dan metrik jarak yang berbeda dapat menghasilkan rekomendasi yang lebih baik.

4. **Penanganan *Cold-Start Problem*:** Sistem rekomendasi saat ini mungkin menghadapi masalah *cold-start* untuk pengguna baru (tanpa riwayat interaksi) atau *smartphone* baru (tanpa banyak interaksi). Pertimbangkan strategi untuk mengatasi masalah ini, seperti:
    - Rekomendasi berbasis popularitas (merekomendasikan *smartphone* terpopuler secara umum).
    - Meminta preferensi awal dari pengguna baru.
    - Menggunakan pendekatan berbasis konten yang lebih mendalam pada fitur *smartphone*.

5. **Eksplorasi Algoritma Rekomendasi Lain:** Meskipun KNN adalah titik awal yang baik, ada berbagai algoritma rekomendasi lain yang mungkin lebih sesuai atau memberikan kinerja yang lebih baik, terutama dengan adanya data interaksi pengguna. Pertimbangkan untuk mengeksplorasi algoritma seperti:
    - *Collaborative Filtering* (berbasis pengguna atau berbasis item).
    - Model berbasis *matrix factorization* (misalnya, Singular Value Decomposition - SVD).
    - Model *hybrid* yang menggabungkan pendekatan berbasis konten dan *collaborative filtering*.

6. **Penanganan *Outlier*:** Keputusan untuk tidak menangani *outlier* saat ini perlu ditinjau kembali, terutama jika evaluasi formal diimplementasikan. Analisis lebih lanjut mengenai penyebab dan dampak *outlier* pada kinerja model dapat membantu menentukan strategi penanganan yang tepat (misalnya, penghapusan, transformasi, atau penggunaan model yang lebih robust terhadap *outlier*).

Dengan mengimplementasikan saran-saran ini, sistem rekomendasi *smartphone* dapat dievaluasi secara lebih komprehensif, ditingkatkan kinerjanya, dan menjadi lebih adaptif terhadap kebutuhan pengguna.

VI. Daftar Pustaka

Wicaksono, F. H. (2019). *Pengantar Machine Learning dengan Python*. Deepublish

Wibowo, A. (2021). *Penerapan Algoritma K-NN dalam Pengklasifikasian Data Menggunakan Python*. Deepublish.

Aziz, F. (2022). *Pengembangan Sistem Rekomendasi dengan Python dan Machine Learning*. Elex Media Komputindo

VII. Lampiran

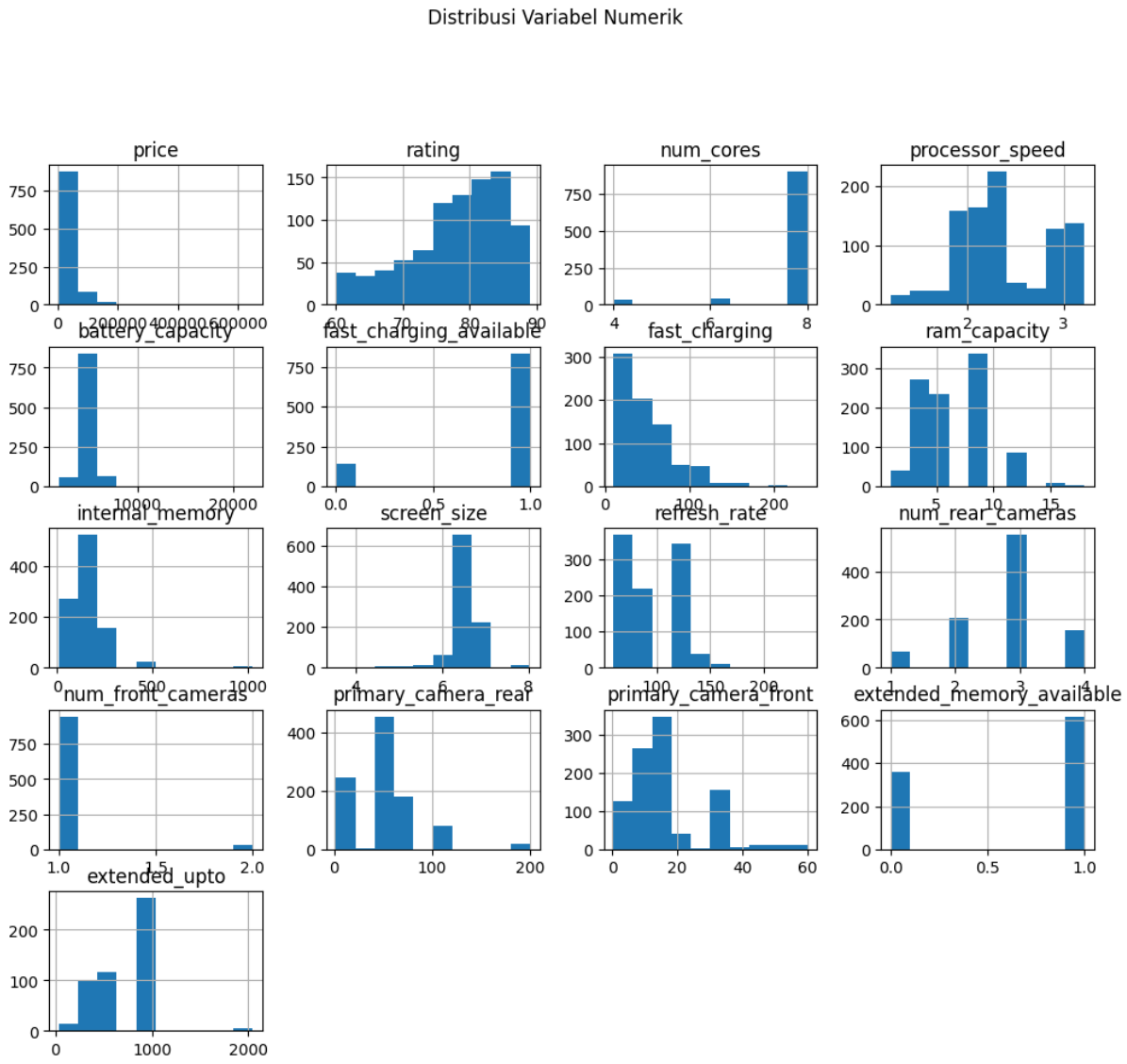
7.1 Kode Program

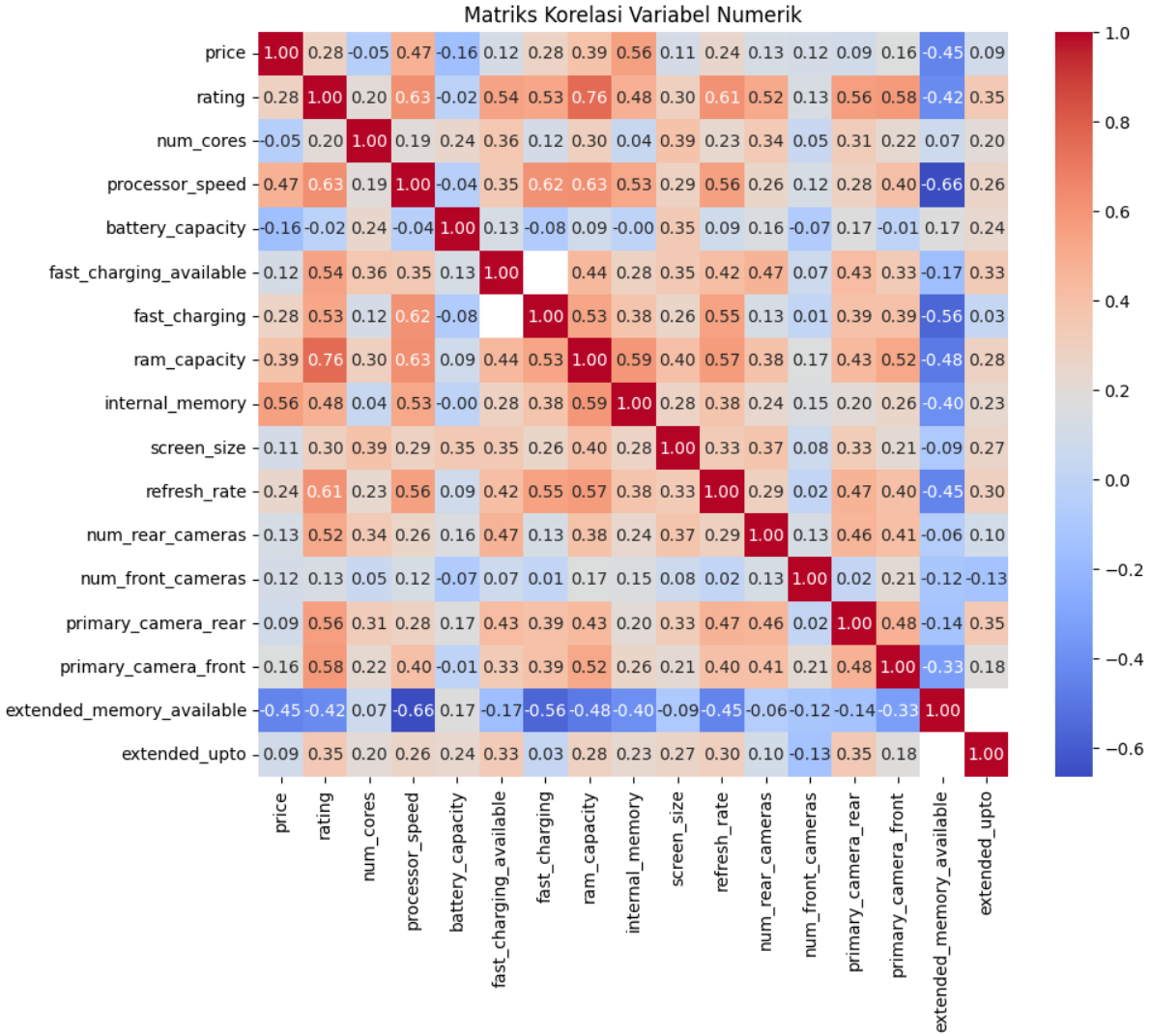
Lampirkan link kode Python yang digunakan untuk proses EDA, pemrosesan data, pembangunan model, dan evaluasi model. Pastikan kode diberi komentar agar mudah dipahami.

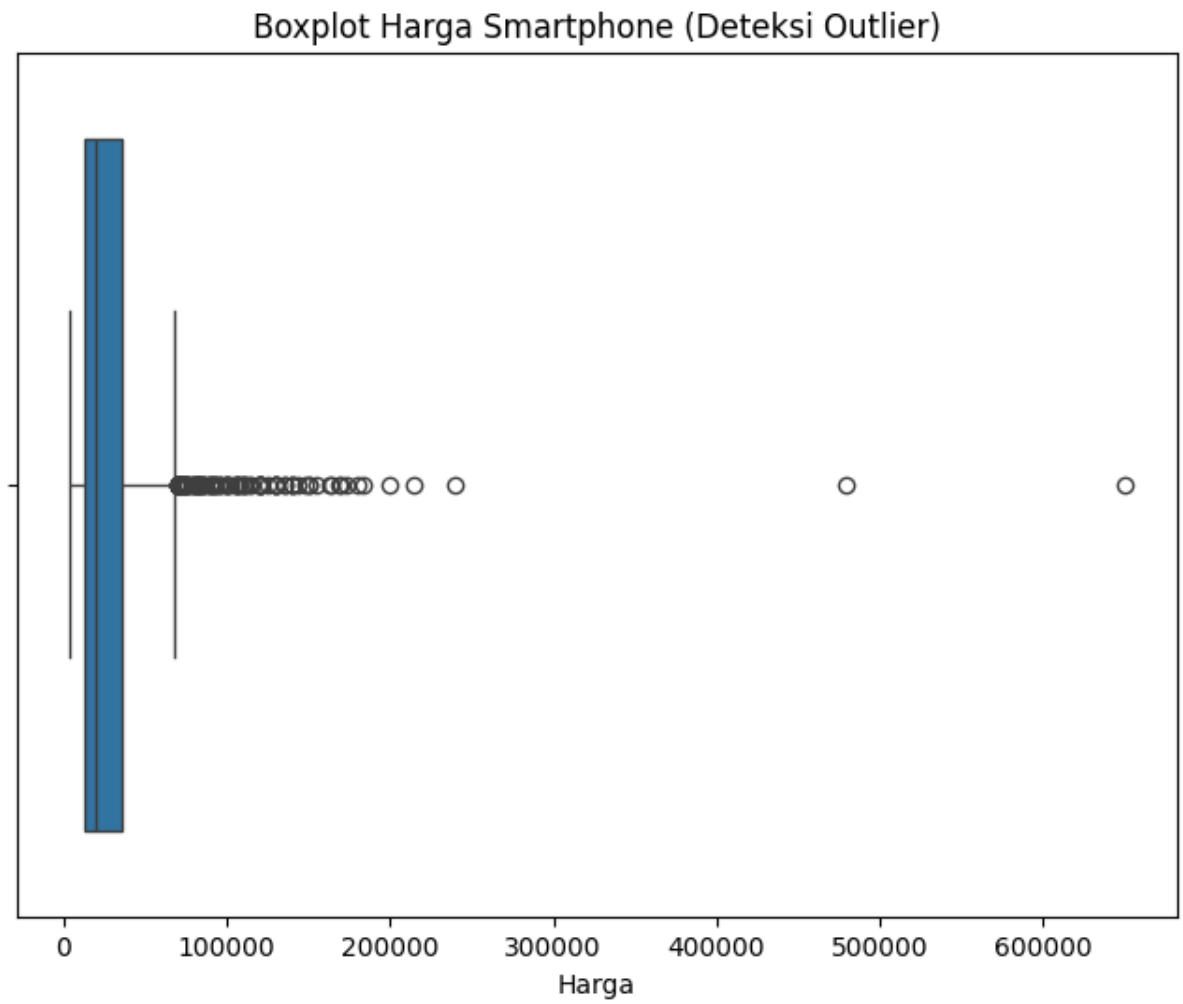
https://github.com/nailhakim/Deep-Learning/blob/main/UTS_BigData_FaharNailH_1103202133.ipynb

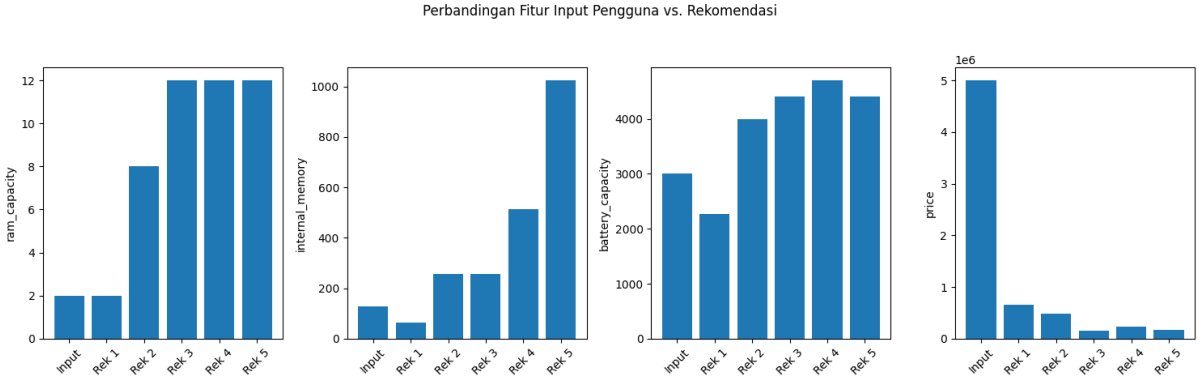
https://colab.research.google.com/drive/1VyA5tmN4zapxtpjefKK16osNLS95hOJt?usp=sharing

7.2 Visualisasi dan Tabel




Matriks Korelasi Variabel Numerik
price 1.00 0.28 -0.05 0.47 -0.16 0.12 0.28 0.39 0.56 0.11 0.24 0.13 0.12 0.09 0.16 -0.45 0.09
rating 0.28 1.00 0.20 0.63 -0.02 0.54 0.53 0.76 0.48 0.30 0.61 0.52 0.13 0.56 0.58 -0.42 0.35
num_cores -0.05 0.20 1.00 0.19 0.24 0.36 0.12 0.30 0.04 0.39 0.23 0.34 0.05 0.31 0.22 0.07 0.20
processor_speed 0.47 0.63 0.19 1.00 -0.04 0.35 0.62 0.63 0.53 0.29 0.56 0.26 0.12 0.28 0.40 -0.66 0.26
battery_capacity -0.16 -0.02 0.24 -0.04 1.00 0.13 -0.08 0.09 -0.00 0.35 0.09 0.16 -0.07 0.17 -0.01 0.17 0.24
fast_charging_available 0.12 0.54 0.36 0.35 0.13 1.00 0.44 0.28 0.35 0.42 0.47 0.07 0.43 0.33 -0.17 0.33
fast_charging 0.28 0.53 0.12 0.62 -0.08 1.00 0.53 0.38 0.26 0.55 0.13 0.01 0.39 0.39 -0.56 0.03
ram_capacity 0.39 0.76 0.30 0.63 0.09 0.44 0.53 1.00 0.59 0.40 0.57 0.38 0.17 0.43 0.52 -0.48 0.28
internal_memory 0.56 0.48 0.04 0.53 -0.00 0.28 0.38 0.59 1.00 0.28 0.38 0.24 0.15 0.20 0.26 -0.40 0.23
screen_size 0.11 0.30 0.39 0.29 0.35 0.35 0.26 0.40 0.28 1.00 0.33 0.37 0.08 0.33 0.21 -0.09 0.27
refresh_rate 0.24 0.61 0.23 0.56 0.09 0.42 0.55 0.57 0.38 0.33 1.00 0.29 0.02 0.47 0.40 -0.45 0.30
num_rear_cameras 0.13 0.52 0.34 0.26 0.16 0.47 0.13 0.38 0.24 0.37 0.29 1.00 0.13 0.46 0.41 -0.06 0.10
num_front_cameras 0.12 0.13 0.05 0.12 -0.07 0.07 0.01 0.17 0.15 0.08 0.02 0.13 1.00 0.02 0.21 -0.12 -0.13
primary_camera_rear 0.09 0.56 0.31 0.28 0.17 0.43 0.39 0.43 0.20 0.33 0.47 0.46 0.02 1.00 0.48 -0.14 0.35
primary_camera_front 0.16 0.58 0.22 0.40 -0.01 0.33 0.39 0.52 0.26 0.21 0.40 0.41 0.21 0.48 1.00 -0.33 0.18
extended_memory_available -0.45 -0.42 0.07 -0.66 0.17 -0.17 -0.56 -0.48 -0.40 -0.09 -0.45 -0.06 -0.12 -0.14 -0.33 1.00
extended_upto 0.09 0.35 0.20 0.26 0.24 0.33 0.03 0.28 0.23 0.27 0.30 0.10 -0.13 0.35 0.18 1.00
price
rating
num_cores
processor_speed
battery_capacity
fast_charging_available
fast_charging
ram_capacity
internal_memory
screen_size
refresh_rate
num_rear_cameras
num_front_cameras
primary_camera_rear
primary_camera_front
extended_memory_available
extended_upto
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6


Boxplot Harga Smartphone (Deteksi Outlier)
0
100000
200000
300000
400000
500000
600000
Harga


Perbandingan Fitur Input Pengguna vs. Rekomendasi
ram_capacity
internal_memory
battery_capacity
price
1e6
Input
Rek 1
Rek 2
Rek 3
Rek 4
Rek 5