

# **LAPORAN UJIAN AKHIR SEMESTER**

## **Analisis dan Prediksi Rating Terhadap 3 Varian Parfum Pada Merk Tertentu**

*Disusun untuk Memenuhi Tugas Kuliah*

*Mata Kuliah: Pemrosesan Teks*

**Dosen Pengampu:**  
*Riskyana Dewi Intan Puspitasari, M.Kom*



Disusun Oleh:

Muhammad Fabyan Putroagung	23031554029
Muhammad Taufiqulhakim	23031554111
Aufatir Diaul Haq	23031554127
Akhmad Alviantio	23031554238

Kelas: Sains Data 2023 D

PROGRAM STUDI SAINS DATA UNIVERSITAS NEGERI SURABAYA TAHUN 2024

## 1. Ide Project

Judul Project : Analisis dan Prediksi Rating Terhadap 3 Varian Parfum Pada Merk Tertentu

a. Latar Belakang, Manfaat, dan Tujuan :

**Latar Belakang:** E-commerce telah menjadi salah satu sektor utama dalam ekonomi digital, memungkinkan konsumen untuk mengakses berbagai produk dengan mudah melalui platform online. Dalam persaingan bisnis yang semakin ketat, ulasan pelanggan memainkan peran penting dalam menentukan keberhasilan suatu produk. Ulasan dan rating tidak hanya menjadi indikator kualitas produk tetapi juga memengaruhi keputusan pembelian konsumen. Namun, ulasan yang tersebar di berbagai platform sering kali sulit diolah secara langsung. Oleh karena itu, diperlukan pendekatan untuk mengumpulkan, memproses, dan menganalisis data tersebut secara otomatis. Salah satu pendekatan yang banyak digunakan adalah teknik web scraping untuk mengambil data dari situs web, seperti nama produk, deskripsi, rating, dan ulasan konsumen.

**Manfaat:**

**Bagi Penjual atau Produsen:**

- Mendapatkan wawasan mengenai persepsi konsumen terhadap produk.
- Mengidentifikasi kelemahan produk berdasarkan ulasan konsumen untuk perbaikan.
- Meningkatkan strategi pemasaran dengan memahami preferensi konsumen.

**Bagi Konsumen:**

- Menyediakan informasi yang lebih terstruktur tentang produk sebelum melakukan pembelian.
- Mempermudah membandingkan kualitas produk berdasarkan ulasan dan rating.

**Tujuan:**

1. Mengembangkan sistem otomatis untuk mengumpulkan data produk dan ulasan dari situs e-commerce.
2. Melakukan pre-processing data untuk memastikan data siap digunakan dalam analisis lebih lanjut.
3. Mengeksplorasi informasi penting dari ulasan konsumen, seperti analisis sentimen atau pola tertentu dalam preferensi konsumen.
4. Membuat dataset yang terstruktur yang dapat digunakan untuk penelitian dan pengembangan teknologi analitik di masa depan.

c. Dataset yang digunakan:

Dataset diambil dari deskripsi dan kolom review dengan menggunakan metode *Web Scraping* pada website Tokopedia pada store tertentu seperti HMNS, Mykonos dan, Saff & Co.

d. Referensi/publikasi acuan yang digunakan (prosiding/jurnal internasional) :

Hsin-Yu Chen, Huang-Chin Yen, Tian Tian. *Female Clothing E-commerce Review and Rating*.  
<https://medium.com/analytics-vidhya/predicting-the-ratings-of-reviews-of-a-hotel-using-machine-learning-bd756e6a9b9b>

## 2. Preprocessing

- a. **Mengubah teks menjadi huruf kecil:**
  - text = text.lower()
  - Semua karakter diubah menjadi huruf kecil untuk memastikan konsistensi saat analisis.
- b. **Mengganti emoji dengan teks deskriptif:**
  - replace\_emojis mengonversi emoji menjadi representasi teks melalui fungsi emoji.demojize() dan membersihkan karakter seperti - atau : dalam nama emoji.
  - Contoh: 😊 → smiling\_face
- c. **Menghapus karakter yang berulang:**
  - remove\_repeated\_characters mengganti karakter yang diulang lebih dari dua kali menjadi satu.
  - Contoh: "haaaaappy" → "hapy".
- d. **Menghapus tanda baca:**
  - text.translate(str.maketrans("", "", string.punctuation)) menghapus semua tanda baca, seperti tanda titik, koma, tanda seru, dan sebagainya.
- e. **Menghapus angka:**
  - text = re.sub(r"\d+", "", text) menghapus semua angka dari teks.
- f. **Menghapus kata yang berakhiran "nya":**
  - text = re.sub(r"\b\w\*nya\b", "", text) menghapus kata yang berakhiran -nya untuk menormalkan kata.
  - Contoh: "rumahnya" → "rumah".
- g. **Menghapus elipsis dan karakter khusus lainnya:**
  - Menghapus pola seperti ..., , atau karakter lain yang dianggap tidak relevan dalam analisis.
- h. **Tokenisasi teks:**
  - tokens = word\_tokenize(text) memecah teks menjadi daftar kata berdasarkan spasi atau pemisah lainnya.
  - Contoh: "Saya suka makan nasi" → ["Saya", "suka", "makan", "nasi"].
- i. **Lemmatization menggunakan Sastrawi:**
  - lemmatized\_tokens = [lemmatizer.stem(token) for token in tokens] mereduksi kata ke bentuk dasarnya (kata dasar dalam bahasa Indonesia).
  - Contoh: "makanannya" → "makan", "berlari" → "lari".
- j. **Penggabungan kembali teks:**
  - ''.join(lemmatized\_tokens) menggabungkan token yang sudah dilemmatize menjadi teks bersih untuk digunakan kembali.

## Exploratory Data Analysis:

## 1. Word Cloud:

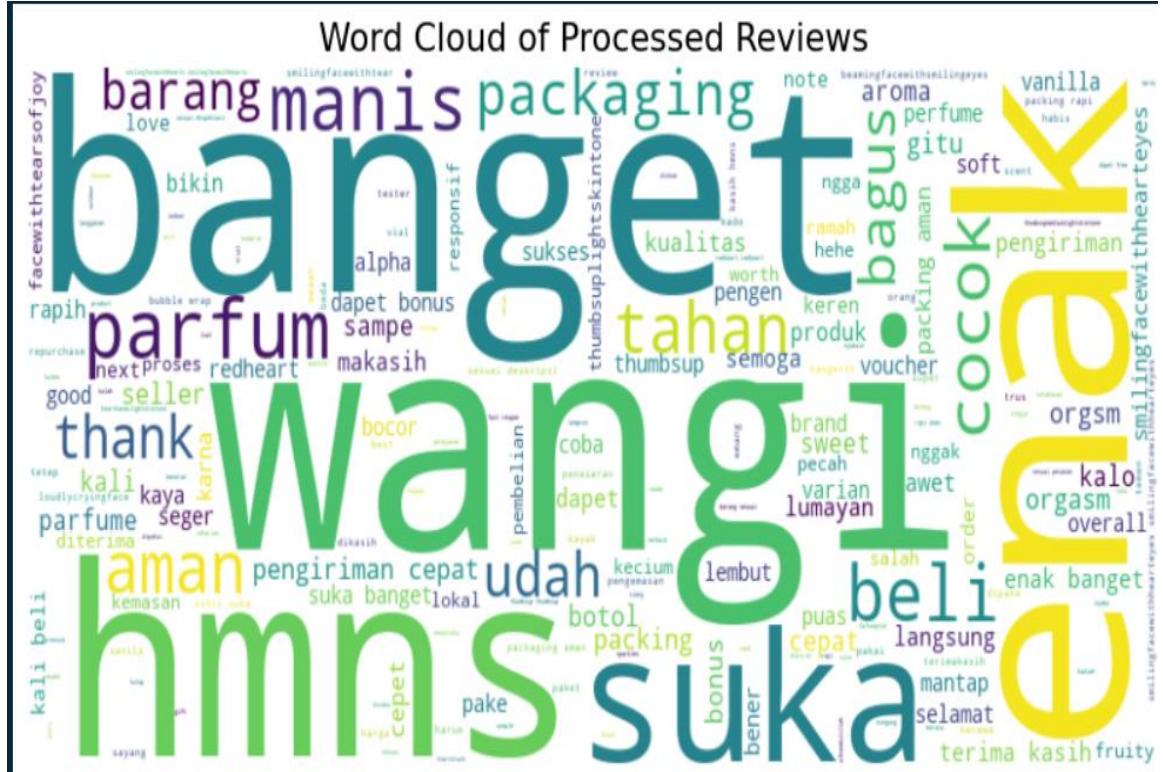
HMNS:

Essence of The Sun (4,9/5, 8200 (5-Stars))

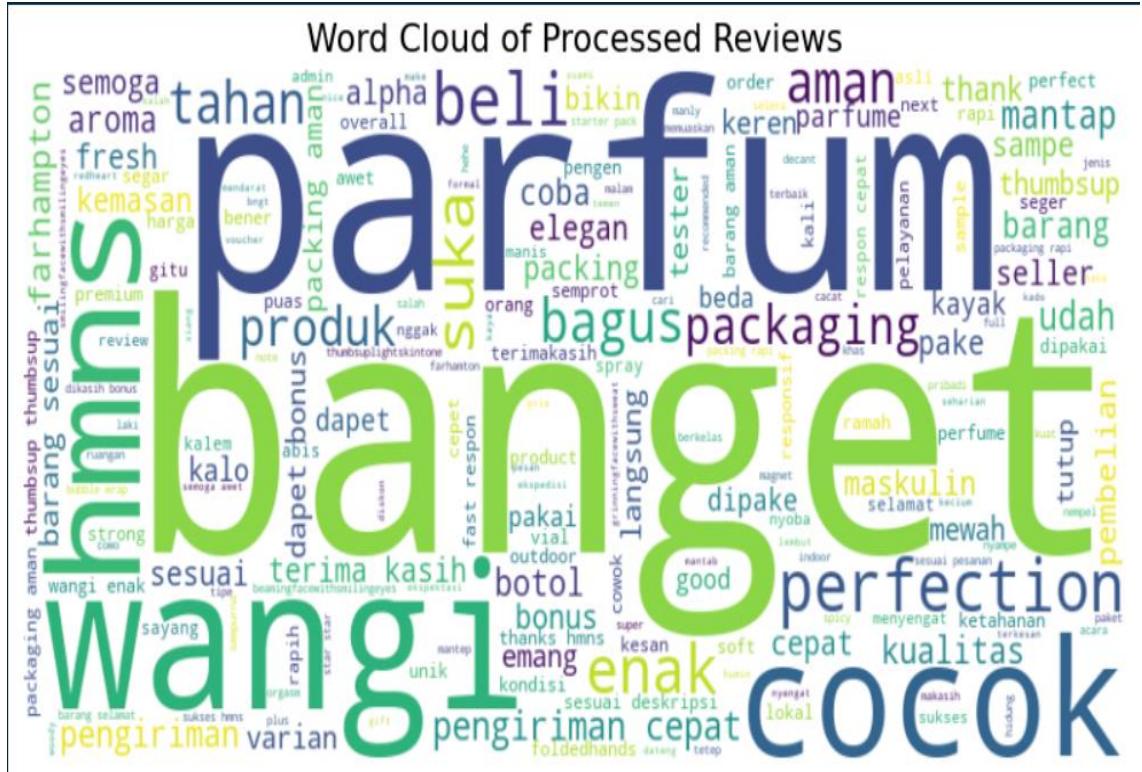


Pada Word Cloud menunjukkan bahwa Essence of The Sun memiliki aroma manis vanilla yang soft dan wangi.

Orgasm (4,9/5, 19644 (5-Stars))



Pada Word Cloud menunjukkan bahwa Orgasm memiliki aroma manis, fruity, segar dan wangi.



Pada Word Cloud menunjukkan bahwa The Perfection memiliki aroma elegan, fresh, mewah, maskulin dan wangi.

## Mykonos:

Aphrodite (4,9/5, 604 (5-Stars))



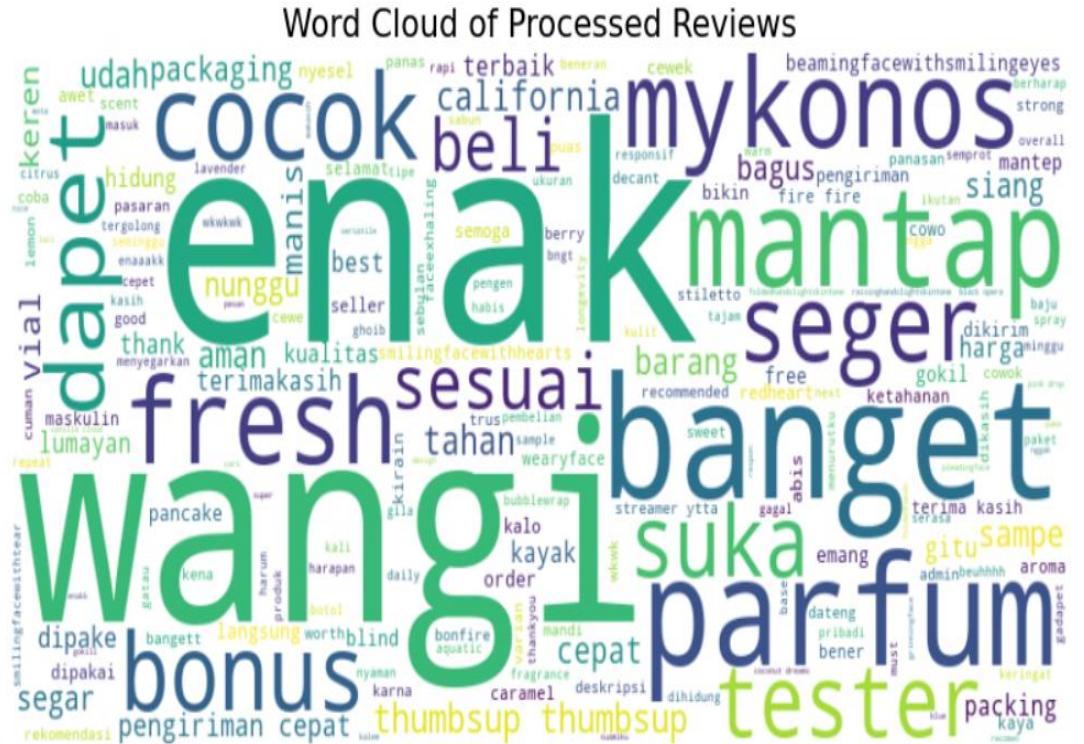
Pada Word Cloud menunjukkan bahwa Aphrodite memiliki aroma vanilla, soft, powdery, woody, manis, dan cocok untuk wanita.

Bonfire Vanilla (4,9/5, 862 (5-Stars))



Pada Word Cloud menunjukkan bahwa Bonfire Vanilla memiliki aroma vanilla, smoky, spicy, dan cocok untuk acara formal.

California (5/5, 725 (5-Stars))



Pada Word Cloud menunjukkan bahwa California memiliki aroma fresh, maskulin, dan cocok untuk waktu siang hari.

Saff & Co :

Loui (4,9/5, 4512 (5-Stars))

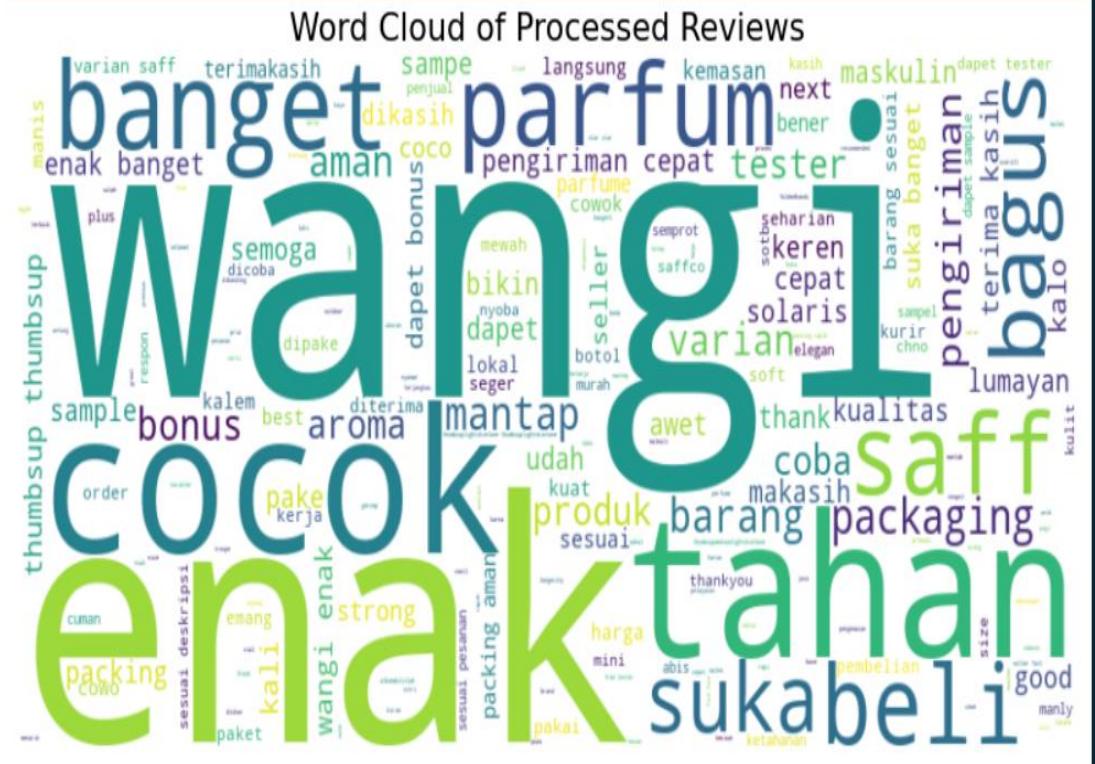


Pada Word Cloud menunjukkan bahwa loui memiliki aroma fresh, floral, dan soft.

S.O.T.B(5/5, 8833 (5-Stars))



Pada Word Cloud menunjukkan bahwa loui memiliki aroma vanilla , soft , seger , dan awet.



Pada Word Cloud menunjukkan bahwa saff memiliki aroma strong, maskulin, dan cocok untuk digunakan saat bekerja.

## Kesimpulan:

- Pada Merk HMNS varian parfum Orgasm menjadi paling favorit dengan memiliki rating bintang 5 berjumlah 19644 karna memiliki aroma manis, fruity, segar dan wangi.
  - Pada Merk Mykonos varian parfum Bonfire Vanilla menjadi paling favorit dengan memiliki rating bintang 5 berjumlah 862 karna memiliki aroma vanilla, smoky, spicy, dan cocok untuk acara formal.
  - Pada Merk Saff & Co varian parfum S.O.T.B menjadi palign favoriti dengan memiliki rating bintang 5 berjumlah 8833 karna memiliki aroma vanilla , soft , seger , dan awet.

### 3. Feature Engineering

Agar dapat menganalisis data teks dan menggunakannya dalam model, kita perlu mengubah data teks menjadi fitur numerik. Ada berbagai model untuk representasi kata (word embedding), termasuk **Bag-of-Words (BOW)**, **Term Frequency-Inverse Document Frequency (TF-IDF)**, dan **Word2Vec**, dll. Dalam laporan ini, kami menggunakan **TF-IDF** karena metode ini paling umum digunakan dan mudah diterapkan.

**TF-IDF** adalah model berbobot yang sering digunakan untuk masalah pencarian informasi. Model ini mengonversi dokumen teks menjadi model vektor berdasarkan kemunculan kata-kata dalam dokumen tanpa memperhatikan urutan kata yang tepat.

#### Komponen TF-IDF

##### TF (Term Frequency)

- Mengukur seberapa sering suatu kata muncul dalam sebuah ulasan, yang kemudian dibagi dengan total jumlah kata dalam dokumen tersebut.
- Hal ini penting karena kata dalam ulasan panjang cenderung muncul lebih sering dibandingkan ulasan pendek. Oleh karena itu, pembagian dengan panjang ulasan digunakan untuk "menormalkan" frekuensi kata.

##### IDF (Inverse Document Frequency)

- Mengukur seberapa penting suatu kata.
- Saat menghitung TF, semua kata dianggap memiliki tingkat kepentingan yang sama. Namun, beberapa kata seperti "adalah", "dari", dan "itu" mungkin muncul sangat sering tetapi memiliki kepentingan yang rendah. Oleh karena itu, IDF bertujuan untuk menurunkan bobot kata-kata yang sering muncul sekaligus meningkatkan bobot kata-kata yang jarang.

#### Implementasi dengan Scikit-learn

Scikit-learn menyediakan dua cara untuk memvectorisasi teks dan menghasilkan matriks bobot TF-IDF:

##### 1. Proses Dua Tahap

- Menggunakan **CountVectorizer** untuk menghitung berapa kali setiap kata muncul di setiap ulasan.
- Dilanjutkan dengan **TfidfTransformer** untuk menghitung bobot setiap kata dan menghasilkan matriks bobot TF-IDF.

##### 2. Proses Tunggal

- Menggunakan **TfidfVectorizer**, yang menggabungkan kedua langkah di atas menjadi satu proses.

## Explanatory Data Analysis:

Dalam laporan ini, kami memilih metode pertama (proses dua tahap). Metode ini memungkinkan kita mengaitkan setiap kata dalam ulasan dengan angka yang merepresentasikan tingkat kepentingan setiap kata dalam ulasan tersebut.

Tabel di bawah ini menunjukkan skor TF-IDF dari setiap kata dalam setiap ulasan.

	abis	ac	acara	ada	ada wangi	ada yg	admin	agak	agak kurang	aja	...	yang saya	yang suka	yg	yg ini	yg lain	yg suka	you	Keyword	Max	Sum
0	0.0	0.0	0.0	0.093991	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.076536	0.0	0.0	0.0	0.0	gw	0.308952	6.123470
1	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	terima	0.308371	5.183218
2	0.0	0.0	0.0	0.120161	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	jam	0.295589	5.341378
3	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	rasa	0.415410	4.991125
4	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	produk	0.409323	3.952108
5	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	mau beli	0.394884	3.479775
6	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	cocok di	0.400633	3.325876
7	0.0	0.0	0.0	0.183444	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.074688	0.0	0.0	0.0	0.0	di	0.393734	5.245010
8	0.0	0.0	0.0	0.125098	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	kurir	0.432410	4.682538
9	0.0	0.0	0.0	0.091206	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.074268	0.0	0.0	0.0	0.0	cuman	0.457878	5.702982

- Keyword: Kata/frasa terpenting dalam dokumen berdasarkan TF-IDF.
- Max: Bobot TF-IDF terbesar.
- Sum: Jumlah total bobot TF-IDF dari semua kata/frasa dalam dokumen.

Semakin tinggi skor TF-IDF suatu kata, semakin relevan kata tersebut dalam ulasan tertentu. Selain itu, ulasan yang memiliki kata-kata relevan yang serupa akan menghasilkan vektor yang serupa. Hal ini adalah karakteristik yang kita cari dalam algoritma machine learning, karena memungkinkan model untuk mengenali pola dan hubungan antara kata-kata dalam ulasan.

```
cvec = CountVectorizer(ngram_range=(1,2), min_df=.005, max_df=.9)
cvec.fit(data2['processed_review'])
transformer = TfidfTransformer()
transformed_weights = transformer.fit_transform(cvec_counts)
transformed_weights = transformed_weights.toarray()

vocab = cvec.get_feature_names_out()

model = pd.DataFrame(transformed_weights, columns=vocab)

model['Keyword'] = model.idxmax(axis=1)
model['Max'] = model.drop(columns=['Keyword']).max(axis=1)

model['Sum'] = model.drop(columns=['Keyword', 'Max']).sum(axis=1)
```

## 4. Metode

### a. Penentuan Metode:

TF-IDF (Term Frequency-Inverse Document Frequency)

TF-IDF adalah metode representasi teks numerik berbasis statistik yang memberikan bobot kepada setiap kata berdasarkan frekuensi kata dalam dokumen dan seberapa jarangnya kata tersebut di semua dokumen.

Logistic regression

Regresi Logistik adalah metode statistik untuk menganalisis dataset di mana hasilnya diukur dengan variabel dikotomis (hanya memiliki dua kemungkinan hasil). Salah satu keterbatasan Regresi Logistik adalah sangat bergantung pada penyajian dataset yang tepat, di mana semua variabel independen yang penting harus diidentifikasi dengan jelas. Keterbatasan lainnya adalah bahwa metode ini hanya dapat memprediksi hasil kategorikal.

WordCloud

Word Cloud digunakan untuk memvisualisasikan dan menyoroti istilah/kata yang populer atau sedang tren berdasarkan frekuensi penggunaan dan keunggulannya.

SupportVectorMachine(SVM)

SVM mengklasifikasikan data ke dalam kategori dengan menggambar hyperplane untuk memisahkan kelompok data berdasarkan pola. Algoritma ini sering diterapkan dalam kategorisasi teks, klasifikasi gambar, pengenalan tulisan tangan, dan di bidang sains. Namun, jika dataset besar dan memiliki lebih banyak noise, SVM kurang cocok untuk digunakan.

Naïve Bayes

Sebagai model pembelajaran mesin probabilistik, model Naïve Bayes membantu pengguna menemukan probabilitas terjadinya A, dengan asumsi bahwa B telah terjadi.

Tree based method

Classification tree - Pohon klasifikasi adalah alat yang digunakan untuk membatasi kelas dari pengklasifikasi. Struktur berbentuk pohon ini mudah diinterpretasikan dan dengan jelas menunjukkan kepada pengguna klasifikasi mana yang menghasilkan hasil yang benar.

Random Forest - Terdiri dari sejumlah besar pohon keputusan individu. Setiap pohon individu dalam random forest menghasilkan prediksi kelas. Kelas dengan suara terbanyak digunakan untuk analisis lebih lanjut. Namun, karena dalam Random Forest jangkauan prediksi dibatasi oleh label tertinggi dan terendah dalam data pelatihan, dapat terjadi masalah pergeseran kovariat jika input pelatihan dan prediksi memiliki jangkauan yang berbeda.

**XGBoost** - XGBoost adalah singkatan dari Extreme Gradient Boosting. XGBoost menghitung gradien orde kedua untuk mendapatkan informasi lebih lanjut tentang arah gradien serta mencapai minimum dari fungsi loss. Selain itu, metode ini meningkatkan regularisasi sehingga generalisasi model juga menjadi lebih baik.

b. Kesesuaian metode dengan penyelesaian masalah dalam project:

#### 1. Logistic Regression

- **Penjelasan:** Logistic regression digunakan untuk memprediksi variabel kategorikal (dalam hal ini, rating yang dikelompokkan sebagai kategori, misalnya positif/negatif atau bintang tertentu).
- **Kesesuaian:**
  - Cocok untuk memprediksi rating biner (seperti positif/negatif).
  - Kurang cocok jika rating memiliki banyak kategori (multikategori), kecuali diubah menjadi beberapa model biner (one-vs-rest).
  - Efektif jika data teks sudah dikonversi ke representasi numerik seperti TF-IDF atau word embeddings.
  - Membutuhkan preprocessing yang baik untuk performa optimal.

#### 2. Word Cloud

- **Penjelasan:** Word cloud digunakan untuk memvisualisasikan kata-kata yang sering muncul dalam dataset.
- **Kesesuaian:**
  - Bukan metode prediksi, tetapi sangat berguna dalam **EDA** (exploratory data analysis).
  - Membantu memahami kata-kata yang sering digunakan dalam review, yang dapat memberikan insight untuk preprocessing (misalnya, kata stopwords atau kata kunci penting).
  - Tidak menghasilkan model prediktif secara langsung.

#### 3. Support Vector Machine (SVM)

- **Penjelasan:** SVM menggunakan hyperplane untuk memisahkan data ke dalam kelas-kelas tertentu.
- **Kesesuaian:**
  - Cocok untuk masalah klasifikasi teks, termasuk prediksi rating berdasarkan teks review.
  - Sangat efektif pada dataset dengan dimensi tinggi, seperti teks yang direpresentasikan dengan TF-IDF atau word embeddings.
  - Kurang efisien pada dataset yang besar atau memiliki banyak noise.
  - Tidak mendukung prediksi kontinu (hanya klasifikasi).

#### 4. Naïve Bayes

- **Penjelasan:** Model probabilistik sederhana yang menghitung kemungkinan suatu kelas berdasarkan distribusi kata.
- **Kesesuaian:**
  - Cocok untuk klasifikasi teks yang sederhana dan cepat.
  - Sering digunakan dalam pengkategorian teks, seperti klasifikasi sentimen atau rating.

- Kurang efektif jika fitur (kata-kata) tidak bersifat independen, meskipun model ini tetap kuat terhadap pelanggaran asumsi.
- Tidak cocok untuk dataset besar dengan fitur kompleks.

## 5. Metode Berbasis Pohon

### a. Classification Tree

- **Penjelasan:** Model pohon keputusan yang memisahkan data berdasarkan atribut.

- **Kesesuaian:**

- Cocok untuk memprediksi rating dalam bentuk kategorikal.
- Mudah dipahami dan divisualisasikan, tetapi rentan terhadap overfitting pada dataset besar.

### b. Random Forest

- **Penjelasan:** Kombinasi dari banyak pohon keputusan untuk menghasilkan prediksi yang lebih stabil dan akurat.

- **Kesesuaian:**

- Cocok untuk prediksi rating kategorikal atau kontinu.
- Baik dalam menangani dataset dengan fitur teks yang kompleks setelah representasi numerik.
- Rentan terhadap masalah jika data pelatihan dan data prediksi memiliki distribusi yang berbeda (covariate shift).

### c. XGBoost

- **Penjelasan:** Versi lanjutan dari Random Forest yang menggunakan gradient boosting.

- **Kesesuaian:**

- Sangat cocok untuk masalah prediksi rating, baik kategorikal maupun kontinu.
- Dapat mengatasi dataset besar dan kompleks dengan baik.
- Memerlukan tuning parameter yang hati-hati untuk performa terbaik.

## 6.TFIDF

Metode TF-IDF sangat cocok untuk project ini karena dapat mengubah data ulasan menjadi fitur numerik dengan memberi bobot tinggi pada kata-kata unik yang relevan dan menurunkan bobot kata-kata umum yang kurang bermakna. TF-IDF membantu menyoroti kata kunci penting dalam ulasan, efisien untuk dataset besar, serta mudah diimplementasikan menggunakan library Scikit-learn. Metode ini menghasilkan vektor numerik yang ideal untuk berbagai algoritma, seperti Logistic Regression, SVM, dan Random Forest, sehingga mendukung analisis eksplorasi, visualisasi, dan prediksi. Dengan menggunakan TF-IDF, data ulasan menjadi lebih terstruktur dan relevan untuk mencapai tujuan project.

### c. Penentuan Metrik Evaluasi

Untuk menilai performa model yang dikembangkan, digunakan metrik evaluasi **accuracy**. Accuracy adalah ukuran performa yang menunjukkan seberapa besar persentase prediksi yang benar terhadap keseluruhan data. Nilai ini dihitung dengan membandingkan jumlah prediksi yang benar dengan total jumlah prediksi yang dibuat oleh model.

Dalam implementasinya, perhitungan nilai akurasi dilakukan menggunakan fungsi `accuracy_score` yang tersedia di library **scikit-learn**. Fungsi ini menerima dua parameter utama, yaitu:

1. **y\_true**: Berisi nilai label sebenarnya (ground truth) dari dataset.
2. **y\_pred**: Berisi nilai label yang diprediksi oleh model.

Rumus matematis dari accuracy adalah sebagai berikut:

$$\text{Accuracy} = \frac{\text{Jumlah Prediksi Benar}}{\text{Total Jumlah Data}}$$

Sebagai contoh, jika dari 100 data terdapat 90 prediksi yang sesuai dengan label sebenarnya, maka nilai akurasi model adalah 0.9 atau 90%.

Pemilihan accuracy sebagai metrik evaluasi sangat sesuai jika dataset memiliki distribusi kelas yang seimbang, yaitu ketika setiap kelas memiliki jumlah data yang kurang lebih sama. Namun, jika dataset memiliki ketidakseimbangan kelas (class imbalance), metrik lain seperti **precision**, **recall**, atau **F1-score** mungkin lebih relevan untuk digunakan.

Dengan menggunakan `accuracy_score`, evaluasi model dapat dilakukan secara efisien dan mudah, memberikan gambaran awal tentang tingkat keakuratan model dalam menyelesaikan tugas klasifikasi.

## 5. Proses

### 1. Membersihkan Konten Teks

Sebelum menerapkan model machine learning pada data kita, kita harus terlebih dahulu menangani data teks. Oleh karena itu, penting untuk membersihkan konten teks ulasan sebagai persiapan model melalui dua langkah berikut:

Langkah pertama adalah memecah ulasan menjadi kata-kata kecil (huruf kecil) dan menggunakan **nltk** untuk menghilangkan **noise**, seperti tanda baca, angka, tautan, dan stopwords (kata-kata umum dalam suatu bahasa – seperti "adalah", "saya", "itu", "di", "dalam", dll.).

Langkah berikutnya adalah menormalisasi kata-kata, yang merupakan langkah penting untuk pemodelan fitur berbasis teks. Proses ini mengonversi fitur berdimensi tinggi (misalnya, N fitur yang berbeda) menjadi ruang berdimensi rendah (1 fitur). Sebagai contoh: kata-kata seperti "lihat", "melihat" adalah variasi berbeda dari kata "lihat". Meskipun maknanya berbeda, secara kontekstual, mereka semua memiliki kesamaan.

Ada dua metode normalisasi leksikon: **Stemming** atau **Lemmatization**. Dalam laporan ini, kita menerapkan **Lemmatization**, karena metode ini mengembalikan bentuk dasar (root form) dari setiap kata, bukan sekadar menghapus akhiran.

Sebelum Pembersihan	Sesudah Pembersihan
Sekali lagi, packaging keren. Untuk 1x semprot wanginya lumayan enak, 30 menit setelahnya lebih enak lagi. Tapi di kulitku ga tahan lama, kurang lebih cuma 3 jam udah ilang. Entah apa penyebabnya, karena ada testimoni lain katanya tahan 7 jam"	sekali lagi packaging keren semprot lumayan enak menit lebih enak kulit tahan lama kurang lebih jam ilang testimoni tahan jam

```
factory = StemmerFactory()
lemmatizer = factory.create_stemmer()

def replace_emojis(text):
    text = emoji.demojize(text)

    text = re.sub(r"[_-]", "", text)

    text = text.replace(":", " ")

    text = re.sub(r"\s+", " ", text).strip()

    return text

def remove_repeated_characters(text):
    return re.sub(r'(.)\1{2,}', r'\1', text)

def preprocess_review(text):
    text = text.lower()

    text = replace_emojis(text)

    text = re.sub(r"\.\.\.|...", " ", text)

    text = text.replace(",", " ")
    text = text.replace("?", "")

    text = text.translate(str.maketrans("", "", string.punctuation))

    text = re.sub(r"\d+", "", text)

    text = remove_repeated_characters(text)

    text = re.sub(r"\b\w*nya\b", "", text)

    tokens = word_tokenize(text)

    lemmatized_tokens = [lemmatizer.stem(token) for token in tokens]
```

## 2. Menvectorisasi Data Teks

### Text Vectorization dengan CountVectorizer

- Menggunakan **CountVectorizer** untuk membentuk representasi teks berupa frekuensi kata/frasa.
- Parameter yang digunakan:
  - `ngram_range=(1, 2)` menghasilkan **unigram** (kata tunggal) dan **bigram** (kombinasi dua kata berurutan).
  - `min_df=0.005` dan `max_df=0.9` bertujuan untuk menyaring kata/frasa:
  - Kata/frasa harus muncul di minimal 0.5% dokumen agar dipertimbangkan.
  - Kata/frasa yang terlalu sering muncul di lebih dari 90% dokumen akan diabaikan.
- Output dari langkah ini adalah matriks frekuensi kata/frasa.

### Konversi ke TF-IDF

- Menggunakan **TfidfTransformer** untuk mengubah matriks frekuensi menjadi bobot **TF-IDF**.
- **Tujuan TF-IDF:**

- Memberikan bobot yang lebih besar pada kata/frasa yang sering muncul dalam sebuah dokumen, tetapi jarang muncul di dokumen lainnya.
- Memungkinkan analisis untuk lebih fokus pada kata-kata yang relevan.

### Membentuk DataFrame Fitur

- Matriks TF-IDF dikonversi menjadi DataFrame dengan:
  - **Baris:** Setiap dokumen (review).
  - **Kolom:** Kata/frasa (dari unigram dan bigram).
  - **Nilai:** Bobot TF-IDF untuk setiap kata/frasa di dokumen.

### Ekstraksi Fitur Tambahan

- Untuk setiap dokumen, diambil fitur tambahan:
  - **Keyword:** Kata/frasa dengan bobot TF-IDF tertinggi di dokumen.
  - **Max:** Nilai TF-IDF tertinggi untuk dokumen.
  - **Sum:** Total nilai TF-IDF semua kata/frasa di dokumen.

	abis	ac	acara	ada	ada wangi	ada yg	admin	agak	agak kurang	aja	...	yang saya	yang suka	yg	yg ini	yg lain	yg suka	you	Keyword	Max	Sum
0	0.0	0.0	0.0	0.093991	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.076536	0.0	0.0	0.0	0.0	gw	0.308952	6.123470
1	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	terima	0.308371	5.183218
2	0.0	0.0	0.0	0.120161	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	jam	0.295589	5.341378
3	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	rasa	0.415410	4.991125
4	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	produk	0.409323	3.952108
5	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	mau beli	0.394884	3.479775
6	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	cocok di	0.400633	3.325876
7	0.0	0.0	0.0	0.183444	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.074688	0.0	0.0	0.0	0.0	di	0.393734	5.245010
8	0.0	0.0	0.0	0.125098	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	kurir	0.432410	4.682538
9	0.0	0.0	0.0	0.091206	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.074268	0.0	0.0	0.0	0.0	cuman	0.457878	5.702982

```
cvec = CountVectorizer(ngram_range=(1,2), min_df=.005, max_df=.9)
cvec.fit(data2['processed_review'])
transformer = TfidfTransformer()
transformed_weights = transformer.fit_transform(cvec_counts)
transformed_weights = transformed_weights.toarray()

vocab = cvec.get_feature_names_out()

model = pd.DataFrame(transformed_weights, columns=vocab)

model['Keyword'] = model.idxmax(axis=1)
model['Max'] = model.drop(columns=['Keyword']).max(axis=1)

model['Sum'] = model.drop(columns=['Keyword', 'Max']).sum(axis=1)
```

### 3. Machine Learning

Setelah data teks diubah menjadi matriks bobot, kita dapat memasukkan matriks tersebut ke dalam model machine learning. Dalam dataset ini, terdapat lima kelas skor rating, sehingga menjadi masalah **multiklasifikasi** untuk mengelompokkan skor-skor rating tersebut. Kami menggunakan tujuh algoritma machine learning yang berbeda untuk memprediksi rating berdasarkan ulasan, yaitu:

1. **Logistic Regression**
2. **K-Nearest Neighbors (KNN)**
3. **Support Vector Machine (SVM)**
4. **Naive Bayes**
5. **Random Forest Classification**
6. **Boosted Decision Tree**

Dalam proses prediksi nilai rating berdasarkan ulasan teks, awalnya dataset memiliki **5 kelas** (rating 1 hingga 5). Namun, setelah dilakukan analisis distribusi data, ditemukan bahwa jumlah ulasan dengan rating **5 dan 4** jauh lebih tinggi dibandingkan rating lainnya (**1, 2, dan 3**). Ketidakseimbangan data ini dapat menyebabkan model machine learning bias terhadap kelas mayoritas (rating 5 dan 4), sehingga mengurangi kemampuan model untuk mengenali pola pada kelas minoritas.

Untuk mengatasi masalah ini, nilai rating dikonsolidasikan menjadi **3 kelas**:

1. **Kelas 1:** Menggabungkan rating rendah (1 dan 2).
2. **Kelas 2:** Rating menengah (3).
3. **Kelas 3:** Menggabungkan rating tinggi (4 dan 5).

Pendekatan ini memiliki beberapa keungulan:

- **Mengurangi Ketidakseimbangan Data:** Penggabungan kelas menciptakan distribusi yang lebih merata di antara kategori, sehingga model tidak terlalu bias terhadap rating tinggi.
- **Meningkatkan Akurasi Model:** Dengan data yang lebih seimbang, model dapat lebih fokus mengenali pola signifikan di setiap kelas.
- **Mempermudah Interpretasi Hasil:** Dengan hanya 3 kelas, hasil prediksi menjadi lebih sederhana dan mudah dipahami.

```
model_p['Rating_3'] = [1 if x == 3 else 0 for x in model_p['rating']]
model_p['Rating_12'] = [1 if x == 1 or x == 2 else 0 for x in model_p['rating']]
model_p['Rating_45'] = [1 if x == 4 or x == 5 else 0 for x in model_p['rating']]
```

**Logistic Regression :**

```
[21]    from sklearn.linear_model import LogisticRegression
        from sklearn.metrics import classification_report
[21]    ✓ 0.0s

[22]    log_model = LogisticRegression(class_weight='balanced', solver = "lbfgs", C = 20)
        log_model.fit(X_train, np.argmax(y_train.to_numpy(), axis = 1))
[22]    ✓ 0.2s
...
          LogisticRegression
          i ? 
          LogisticRegression(C=20, class_weight='balanced')
```

Accuracy of Training Data: 0.9606760952052432

Accuracy of Testing Data: 0.831858407079646

### K-Nearest Neighbour :

```
from sklearn.neighbors import KNeighborsClassifier
```

✓ 0.0s

```
knn_model = KNeighborsClassifier(n_neighbors=5)  
knn_model.fit(X_train, y_train)
```

✓ 0.0s

▼ KNeighborsClassifier ⓘ ⓘ

KNeighborsClassifier()

Accuracy of Training Data: 0.8989306657468092

Accuracy of Testing Data: 0.8897827835880934

### SVC :

```
from sklearn.svm import SVC
```

✓ 0.0s

```
svc_model_linear = SVC(kernel='linear', class_weight='balanced')
```

```
svc_model_linear.fit(X_train, np.argmax(y_train.to_numpy(), axis = 1))
```

✓ 1.5s

▼

SVC

ⓘ ⓘ

SVC(class\_weight='balanced', kernel='linear')

Accuracy of Training Data: 0.9041048637461193

Accuracy of Testing Data: 0.7996781979082864

### Multinomial Naïve Bayes:

```
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import MultinomialNB
✓ 0.0s

mnb_model = MultinomialNB()
mnb_model.fit(X_train, np.argmax(y_train.to_numpy(), axis = 1))
✓ 0.0s
```

▼ MultinomialNB ⓘ ?

MultinomialNB()

Accuracy of Training Data: 0.8968609865470852

Accuracy of Testing Data: 0.8962188254223652

### Random Forest Classifier :

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
✓ 0.2s

rf_model = RandomForestClassifier(n_estimators=1000, class_weight='balanced', random_state=10)
rf_model.fit(X_train, y_train)
✓ 13.2s
```

▼ RandomForestClassifier ⓘ ?

RandomForestClassifier(class\_weight='balanced', n\_estimators=1000,  
random\_state=10)

Accuracy of Testing Data: 0.9851672990686443

Accuracy of Testing Data: 0.8913917940466614

## 6. Hasil dan Analisis

### a. Hasil Eksperimen/Permodelan

Tabel Perbandingan Hasil:

Model	Training Data Accuracy	Testing Data Accuracy
Logistic Regression	0.9606760952052432	0.831858407079646
KNN	0.8989306657468092	0.8897827835880934
SVC	0.9041048637461193	0.7996781979082864
Multinomial Naïve Bayes	0.8968609865470852	0.8962188254223652
Random Forest Classifier	0.9851672990686443	0.8913917940466614
XGBoost	0.938254570541566	0.8962188254223652

Lalu akan dicoba testing secara manual dengan menginput teks lalu model machine learning akan memprediksi dan akan dianalisis

```
class TextPreprocessor:  
    def __init__(self):  
        factory = StemmerFactory()  
        self.lemmatizer = factory.create_stemmer()  
  
    def replace_emojis(self, text):  
        text = emoji.demojize(text)  
        text = re.sub(r"\-_]", "", text)  
        text = text.replace(":", " ")  
        text = re.sub(r"\s+", " ", text).strip()  
        return text  
  
    def remove_repeated_characters(self, text):  
        return re.sub(r"(.)\1{2,}", r"\1", text)  
  
    def preprocess_review(self, text):  
        text = text.lower()  
        text = self.replace_emojis(text)  
        text = re.sub(r"\.\.\.", " ", text)  
        text = text.replace(":", " ")  
        text = text.replace("[", "")  
        text = text.replace("]", "")  
        text = text.translate(str.maketrans("", "", string.punctuation))  
        text = re.sub(r"\d+", "", text)  
        text = self.remove_repeated_characters(text)  
        text = re.sub(r"\b(w'nya)b", "", text)  
        tokens = word_tokenize(text)  
        lemmatized_tokens = [self.lemmatizer.stem(token) for token in tokens]  
        return ' '.join(lemmatized_tokens)  
  
    def predict_manual_input(model, preprocessor, vectorizer):  
        review_input = input("Masukkan review untuk prediksi: ")  
  
        processed_review = preprocessor.preprocess_review(review_input)  
  
        vectorized_input = vectorizer.transform([processed_review]).toarray()  
  
        prediction = model.predict(vectorized_input)  
        print(review_input)  
        print(f"Prediksi untuk review tersebut adalah: {prediction[0]}")  
  
        preprocessor = TextPreprocessor()  
        predict_manual_input(svc_model_linear, preprocessor, cvvec)
```

0 = Buruk (Rating 1-2)

1 = Sedang (Rating 3)

2 = Suka (Rating 4-5)

Teks	LogisticRegression	KNN	SVC	Random Forest Classifier	Multinomial Naïve Bayes	XGBoost
Parfum ini sangat wangi dan tahan lama cocok disegala medan	2	2	2	2	2	1
Parfum ini enak namun kurang cocok	1	1	1	2	2	1
Parfum ini cocok	2	2	2	2	2	1
Parfum ini sangat mengecewakan	0	0	0	0	2	1
Parfum enak service bagus	2	2	2	2	2	1
Parfum enak tapi service kurang	1	2	1	2	2	1

b. Penentuan Metode Terbaik

Dalam analisis table diatas LogisticRegression dan SVC model mendapatkan hasil prediksi yang akurat namun dalam perhitungan akurasi LogisticRegression mendapatkan hasil yang lebih tinggi.

c. Analisis atau Penjelasan Hasil Eksperimen

Kami memutuskan untuk menggunakan model **Logistic Regression** sebagai pendekatan utama dalam pemecahan masalah ini. Keputusan ini didasarkan pada hasil uji coba yang menunjukkan performa model ini sangat baik, baik pada skor evaluasi maupun ketika dilakukan pengujian secara manual dengan memasukkan data teks secara langsung. Model ini mampu menghasilkan prediksi yang konsisten dan akurat dalam skenario yang telah kami simulasikan.

Meskipun terdapat beberapa model lain yang juga memberikan nilai skor evaluasi yang tinggi, kenyataannya saat diuji dengan input manual, model-model tersebut kerap kali menghasilkan prediksi yang meleset. Dalam hal ini, Logistic Regression dan **Support Vector Classifier (SVC)** menonjol karena memiliki performa prediksi manual yang cukup serupa. Namun, **Logistic Regression** berhasil unggul dengan nilai skor pengujian (test score) sebesar **0.8319**, lebih tinggi dibandingkan dengan nilai skor yang dihasilkan oleh SVC, yaitu **0.7997**.

Faktor lain yang mendukung pemilihan Logistic Regression adalah kesederhanaan dan efisiensinya dalam menangani dataset berbasis teks. Logistic Regression tidak hanya memberikan performa yang stabil, tetapi juga lebih mudah diinterpretasikan dibandingkan beberapa model lain. Dalam konteks ini, model ini menunjukkan keseimbangan yang sangat baik antara akurasi skor evaluasi dan kinerja prediksi saat diuji dengan input manual. Keunggulan-keunggulan inilah yang menjadi dasar utama keputusan kami untuk menggunakan Logistic Regression sebagai model terbaik untuk tugas ini.

## 7. Kesimpulan dan Saran

### a. Kesimpulan

Proyek ini bertujuan untuk melakukan analisis dan prediksi **rating** berdasarkan teks **review** yang diberikan oleh pengguna pada platform e-commerce **Tokopedia**. Data yang digunakan dalam proyek ini diambil dari kolom **review**, dengan fokus pada dua fitur utama: **content** (isi ulasan teks) dan **rating** (nilai ulasan dalam rentang 1 hingga 5). Tujuan utama dari proyek ini adalah memahami bagaimana informasi dari ulasan teks dapat digunakan untuk memprediksi nilai rating yang diberikan oleh pengguna.

Untuk mencapai tujuan tersebut, proyek ini melibatkan proses **pra-pemrosesan data**, eksplorasi fitur teks, serta evaluasi berbagai model machine learning untuk menemukan model yang paling sesuai dalam menyelesaikan masalah prediksi rating. Eksperimen dilakukan untuk mengidentifikasi kombinasi terbaik dari fitur dan model yang dapat memberikan hasil akurasi prediksi yang optimal. Beberapa model yang diuji mencakup berbagai algoritma populer, seperti Logistic Regression, Support Vector Classifier (SVC), dan lainnya, yang masing-masing dievaluasi berdasarkan skor evaluasi (misalnya, accuracy) dan performa saat diuji menggunakan input manual.

Selain itu, proyek ini juga bertujuan untuk menyelidiki data review terkait dengan 3 merek parfum tertentu yang populer di Tokopedia. Eksperimen ini dilakukan untuk mengidentifikasi varian parfum terbaik berdasarkan ulasan pengguna, dengan mempertimbangkan korelasi antara teks ulasan dan skor rating yang diberikan. Melalui analisis ini, hasil proyek tidak hanya membantu memahami pola dan tren dari review pengguna, tetapi juga memberikan rekomendasi berdasarkan prediksi data ulasan yang dianalisis.

Hasil akhir dari proyek ini mencakup dua hal utama:

1. Menentukan varian parfum terbaik di antara 3 merek yang dianalisis berdasarkan rating ulasan.
2. Mengidentifikasi model machine learning terbaik yang mampu memprediksi rating pengguna berdasarkan teks review dengan tingkat akurasi yang tinggi.

Dengan pendekatan ini, proyek ini diharapkan memberikan kontribusi signifikan dalam memahami dinamika ulasan pengguna di platform e-commerce serta menawarkan solusi prediktif yang andal dalam menganalisis dan memanfaatkan data teks untuk meningkatkan pengambilan keputusan bisnis.

### b. Kendala Berdasarkan Hasil Eksperimen

Salah satu kendala utama yang dihadapi dalam proyek ini adalah **ketidakseimbangan jumlah bintang rating**, terutama pada kategori dengan bintang di bawah 4. Data menunjukkan bahwa sebagian besar ulasan pengguna cenderung memberikan rating tinggi (4 dan 5), sementara jumlah ulasan dengan rating rendah (1, 2, dan 3) sangat terbatas. Ketidakseimbangan ini dapat memengaruhi kinerja model prediksi karena model cenderung lebih akurat dalam memprediksi kelas yang dominan, yaitu rating tinggi, dibandingkan kelas dengan frekuensi rendah.

Untuk mengatasi masalah ini, dilakukan langkah **pengelompokan ulang** rating menjadi tiga kategori utama yang lebih seimbang dan representatif:

1. **Buruk**: Meliputi rating 1 dan 2, mencerminkan ulasan dengan pengalaman negatif.
2. **Sedang**: Meliputi rating 3, mencerminkan ulasan yang netral atau biasa saja.
3. **Suka**: Meliputi rating 4 dan 5, mencerminkan ulasan dengan pengalaman positif.

Pendekatan pengelompokan ini tidak hanya membantu mengurangi dampak ketidakseimbangan data, tetapi juga menyederhanakan tugas klasifikasi, mengingat model kini hanya perlu memprediksi tiga kelas utama. Dengan begitu, proses pelatihan model

menjadi lebih stabil, dan hasil prediksi lebih mencerminkan pengalaman pengguna secara keseluruhan.

Pengelompokan ini juga memungkinkan analisis yang lebih praktis dan mudah diinterpretasikan, terutama dalam konteks memberikan wawasan kepada pemangku kepentingan. Selain itu, langkah ini memberikan dasar yang lebih baik untuk membangun model yang dapat memberikan hasil akurasi yang lebih tinggi di seluruh kategori, tanpa bias terhadap kelas tertentu.

b. Rencana Lanjut Hasil Eksperimen

Untuk pengembangan lebih lanjut, **Eksperimen Model Lanjutan**

Dalam eksperimen awal, **Logistic Regression** dan **Support Vector Classifier (SVC)** menunjukkan performa yang memadai. Namun, eksperimen lebih lanjut akan mencakup penggunaan model **Deep Learning models** seperti **Recurrent Neural Networks (RNN)** atau **Transformers**, yang dapat menangani data teks dengan lebih efektif. Model-model ini akan diuji untuk melihat apakah mereka dapat meningkatkan performa prediksi lebih jauh, terutama dalam hal akurasi dan generalisasi terhadap data yang belum terlihat sebelumnya. Setelah proses perbaikan dan penyempurnaan model, tahap berikutnya adalah implementasi model dalam sistem nyata. Hal ini termasuk pembuatan pipeline prediksi yang dapat memproses data review secara otomatis, memberikan rating secara real-time, serta monitoring performa model dalam kondisi sebenarnya. Melakukan pemantauan terus-menerus terhadap efektivitas model di dunia nyata adalah langkah penting untuk memastikan keberhasilan implementasi dan memperoleh wawasan tambahan untuk perbaikan model yang lebih berkelanjutan

Link Google Drive :

<https://drive.google.com/drive/u/3/folders/1MDEcIIexE7SyCRKzmKN139aC2MxvtvD2>