BAB I
PENDAHULUAN

1.1 Latar Belakang

1.2 Tujuan

BAB II
LANDASAN TEORI

2.1 Studi Pustaka

BAB III
METODE PENELITIAN

3.1 Kerangka Pemikiran

Kerangka berpikir adalah gambaran yang menggambarkan bagaimana penelitian akan berlangsung secara logis dan menyeluruh. Tahapan-tahapan dalam kerangka berpikir penelitian meliputi antara lain:

1. Pengumpulan Data

   Penelitian ini menggunakan data dari platform *e-commerce* Tokopedia. Data yang diambil adalah riwayat ulasan pelanggan yang pernah berbelanja di toko Hanafashion_shop, yang diperoleh melalui teknik web scraping.
2. Pelabelan

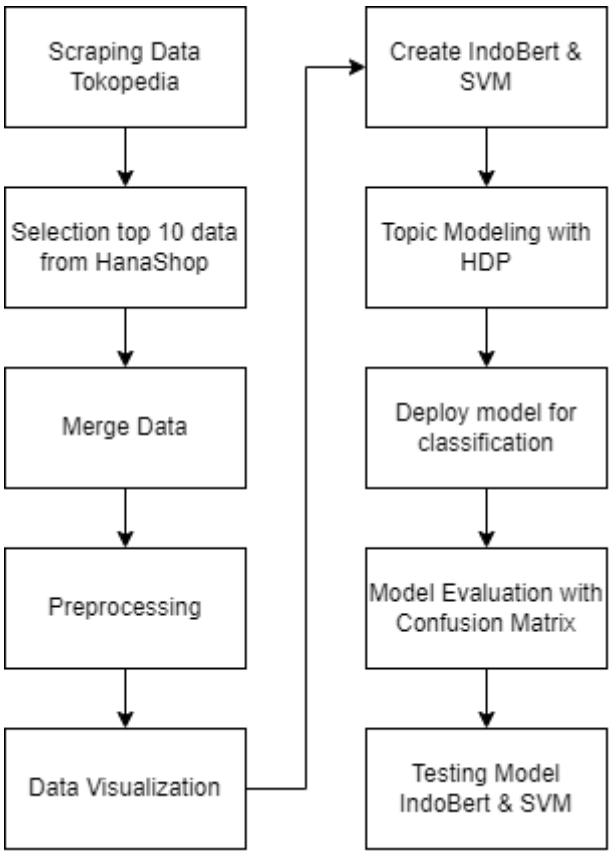
   Pelabelan adalah tahap di mana setiap ulasan diberi label yang akan digunakan dalam proses pelatihan pada tahap klasifikasi. Atribut ulasan berisi tentang pengalaman pelanggan terkait kepuasan mereka saat berbelanja di toko Hanafashion_shop. Pelabelan dilakukan berdasarkan nilai rating produk, di mana rating 1-3 dikategorikan sebagai negatif, sementara rating 4-5 dikategorikan sebagai positif.
3. Preprocessing

   Adapun tahapan yang dilakukan pada preprocessing yaitu sebagai berikut :
   a. *Cleaning*, yaitu untuk menghapus data yang memiliki nilai yang sama *(duplicate)* dan data yang kosong (*nan)*.
   b. *Normalize*, yaitu untuk mengoreksi kata-kata yang salah ketik atau singkatan agar kembali ke bentuk aslinya.
   c. *Stopword*, yaitu untuk menghapus kata-kata umum yang tidak memiliki informasi penting untuk analisis dari teks.
   d. *Tokenizing*, yaitu untuk melakukan pemenggalan pada tiap suku kata.
   e. *Stemming*, yaitu untuk mengubah kata-kata menjadi bentuk dasarnya dengan menghapus akhiran atau imbuhan.
4. Klasifikasi

   Proses klasifikasi dibedakan menjadi dua proses, diantaranya :
   a. *Training* adalah proses melatih algoritma klasifikasi, yakni *Support Vector Machine* dan *IndoBert*, agar dapat berfungsi sesuai harapan. Pertama, data atribut ulasan diberi bobot menggunakan perhitungan TF-IDF, namun hanya memperhitungkan

frekuensi istilah *(term frequency)*. Proses ini menghasilkan model klasifikasi yang kemudian digunakan dalam tahapan *testing*.

b. *Testing* adalah tahap di mana dataset diklasifikasikan dengan menggunakan model klasifikasi yang dihasilkan selama proses *training* data. Pada tahap ini, ulasan dikategorikan ke dalam sentimen positif dan negatif. Langkah-langkah proses ini dapat dilihat pada gambar di bawah.

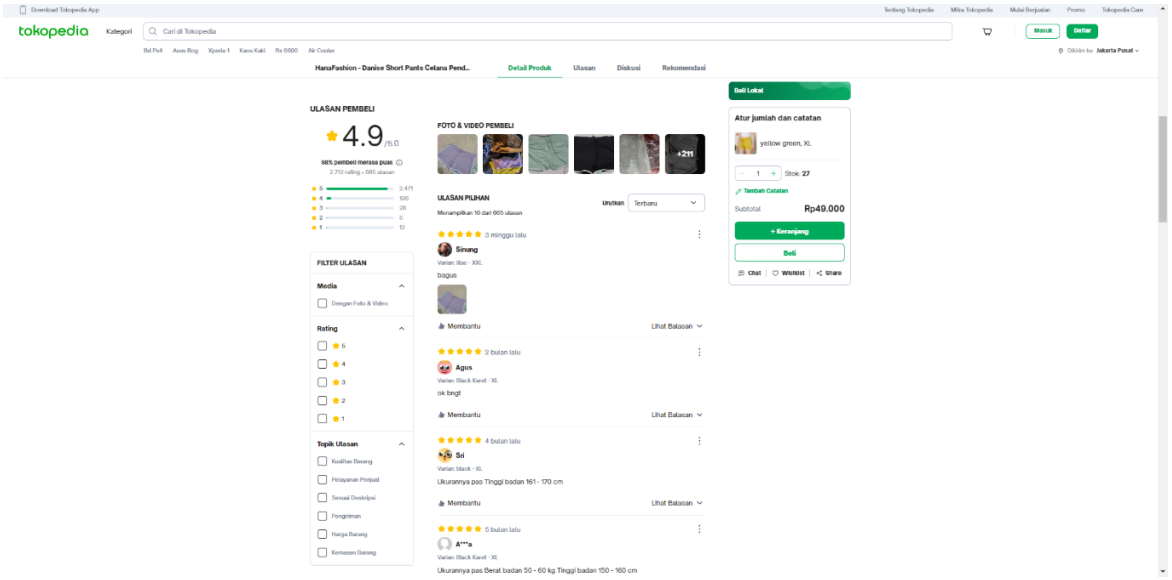


Gambar 3. 1 Kerangka Berpikir

3.2 Bahan/Data

3.2.1 Prosedur Pengumpulan Data

Penelitian ini menggunakan metode scraping menggunakan Selenium untuk mengumpulkan data ulasan toko Hanafashion_shop di situs web Tokopedia. Metode ini memungkinkan pengendalian browser untuk mengakses halaman toko Hanafashion_shop, mengambil ulasan pelanggan, dan mengumpulkan data secara otomatis. Dengan demikian, penelitian ini dapat menganalisis ulasan pelanggan dengan efisien dan mendalam untuk memperoleh wawasan yang relevan terkait dengan pengalaman pelanggan di toko Hanafashion_shop di Tokopedia. Tampilan flowchart pengumpulan data dan scraping data Tokopedia dapat dilihat pada gambar di bawah.

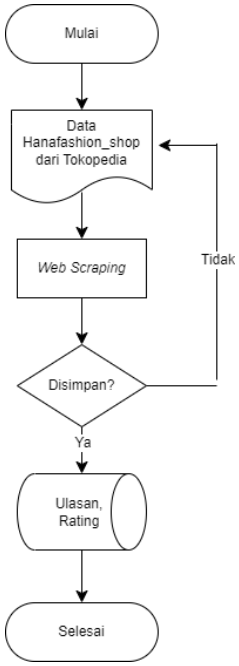


Gambar 3. 2 Produk Hanafashion_shop

Selanjutnya, Selenium akan berinteraksi dengan halaman toko Hanafashion_shop di situs web Tokopedia untuk mencari ulasan yang relevan. Proses ini mungkin melibatkan mengklik pada halaman produk yang tepat, menavigasi melalui halaman ulasan pelanggan, dan mengumpulkan data ulasan yang ada. Selenium akan mengekstrak teks ulasan dari elemen HTML yang sesuai dan menyimpannya dalam format csv untuk preprocessing lebih lanjut. Langkah-langkah tambahan seperti pembersihan data, penghapusan karakter khusus, atau penggabungan ulasan yang terpisah juga dapat dilakukan untuk memastikan data siap untuk analisis sentimen.

3.2.2 Data yang diperoleh

Penelitian ini menggunakan data ulasan toko Hanafashion_shop yang diambil melalui platform Tokopedia sebagai sumber data primer. Data ulasan pelanggan yang dikumpulkan dari toko Hanafashion_shop di Tokopedia akan digunakan sebagai dataset utama dalam penelitian ini. Dengan menggunakan data ulasan yang diperoleh secara langsung dari toko Hanafashion_shop, penelitian ini dapat memberikan wawasan yang lebih spesifik dan relevan terkait dengan pengalaman pelanggan dalam berinteraksi dengan produk-produk yang ditawarkan oleh toko tersebut. Contoh flowchart dan hasil pengumpulan data dapat dilihat pada gambar di bawah.



Gambar 3. 3 Flowchart Pengumpulan Data

	Nama Pelanggan	Produk	Ulasan	Rating
0	tomi	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	Bahan celananya bagus, beliin buat si kaka pas ukurannya	5
1	P***u	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	kualitas bagus.	5
2	Isna	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	Ukurannya pas	5
3	F***n	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	bumil approved 👍	5
5	E***t	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	Ukurannya terlalu besar	5
6	Andre	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	Ukurannya pas Tinggi badan 150 - 160 cm Berat badan 50 - 60 kg	5
7	Putri	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	kain lumayan tipis	4
8	Andrea	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	Ukurannya pas. L. tinggi 152 berat 52. udah order kedua kalinya k	5
9	Martin	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	Tidak ada ulasan	3
10	Andrea	HanaFashion - Danise Short Pants Celana Pendek Wanita - SP061 - yellow green, XL	Ukurannya pas Berat badan 50 - 60 kg	5

Gambar 3. 4 Data Ulasan Tokopedia

Data yang diambil setiap kali melakukan preprocessing berjumlah maksimal 200 baris data.

```
# Alur program di Streamlit
if selected == "Crawling":
    col1, col2 = st.columns([1,8])
    with col1:
        st.image('img/tokopedia.png', width=80)
    with col2:
        st.title("Crawling Data")

    url = st.text_input("Masukkan URL produk:")
    nama_file = st.text_input("Masukkan nama file hasil scraping data:")
    jumlah_data = st.number_input("Masukkan jumlah data yang ingin diambil:", min_value=1, step=1,
value=10)
    rating_min = st.number_input("Masukkan rating minimum yang ingin diambil (1-5):", min_value=1,
max_value=5, step=1, value=1)
    rating_max = st.number_input("Masukkan rating maksimum yang ingin diambil (1-5):", min_value=1,
max_value=5, step=1, value=5)
    folder_path = "data/dataScrapingHanaShop/"
    file_path = folder_path + nama_file

    if st.button("Mulai Scraping"):
        scrapingFunction.scrape_tokopedia_reviews(url, jumlah_data, file_path, rating_min, rating_max)
        st.success(f"Data telah disimpan ke: {file_path}.csv")
```

Gambar 3. 5 Code Scraping Data

Pengguna perlu memasukkan URL sumber produk Hanafashion_shop dari platform Tokopedia, serta jumlah data yang ingin di-scrape. Selain itu, pengguna perlu menentukan kisaran rating untuk ulasan produk yang akan di-scrape. Rentang rating ini akan digunakan untuk menentukan label sentimen ulasan, rating 1-3 dianggap negatif, sedangkan rating 4-5 dianggap positif. Setelah semua input dimasukkan, pengguna dapat mengklik tombol untuk memulai proses scraping data.

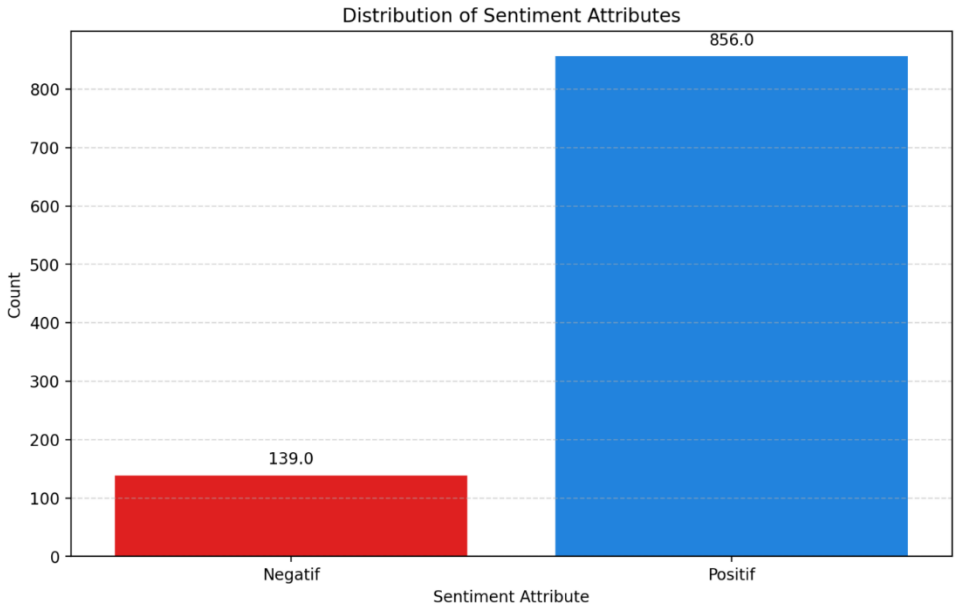
3.2.3 Labeling

Pelabelan adalah tahap di mana ulasan produk diberi label yang nantinya akan digunakan untuk melatih model dalam proses klasifikasi. Ulasan tersedia untuk 10 produk dengan ulasan terbanyak di toko Hanafashion_shop. Labeling didasarkan pada rating yang diberikan oleh pelanggan: rating 1-3 dianggap ulasan negatif, sedangkan rating 4-5 dianggap ulasan positif. Kelemahan pelabelan berdasarkan rating adalah adanya subjektivitas dalam hasilnya. Pada tahap pelabelan, ulasan diberi label sentimen positif dan negatif. Kode program untuk pelabelan dapat dilihat pada gambar di bawah ini.

```python
def labeling(rating):
    if rating == '4' or rating == '5':
        return 'Positif'
    else :
        return 'Negatif'

df['Sentimen'] = df['Rating'].apply(preprocessingFunction.labeling)
```

Gambar 3. 6 Code Labeling Data

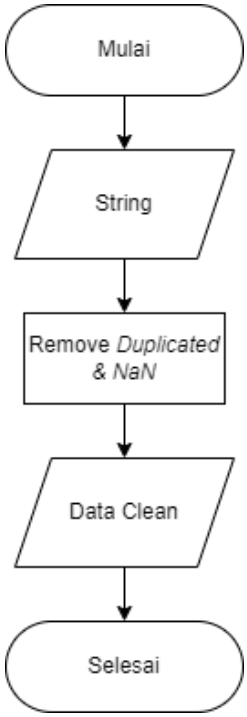
Dari 995 data yang telah diambil dari web scraping sebelumnya, dilakukan pelabelan berdasarkan rating dan menghasilkan sentiment positif dan negatif sebagai berikut :



Gambar 3. 7 Visualisasi Sentimen

3.2.4 Cleaning

Dalam proses *cleaning* data, tahap awal adalah mengidentifikasi dan menghapus nilai yang terduplikasi. Selain itu, langkah penting lainnya adalah menangani nilai yang hilang (NaN) agar data menjadi lebih bersih dan siap digunakan dalam pelatihan model SVM dan indoBert, sehingga kinerja model dapat dioptimalkan. Adapun *flowchart* dan kode program dari cleaning data dapat dilihat pada gambar di bawah.



Gambar 3. 8 Flowchart Cleaning Data

```
def clean(text):
  text = text.strip()
  text = text.lower()
  text = re.sub(r'[^a-zA-Z]+', ' ', text)
  return text
  return go(f, seed, [])
}

df['Ulasan'] = df['Ulasan'].fillna('')
df['Ulasan'] = df['Ulasan'].apply(preprocessingFunction.clean)
st.write(df['Ulasan'])
```

Gambar 3. 9 Code Cleaning Data

3.2.5 Normalize

Pada tahapan *normalize* data, fokus utamanya adalah memperbaiki kata-kata yang mungkin disingkat atau tertulis dengan typo seperti "seller", "dll", "bgos", dan sejenisnya. Proses ini bertujuan untuk memastikan konsistensi dalam penggunaan kata-kata sehingga data menjadi lebih mudah dipahami dan diolah. Adapun contoh dari tahap *normalize* dapat dilihat pada table di bawah.

Tabel 3. 1 Contoh *Normalize*

Text Input	Text Output
seller responsif brg ny ok bangettt	penjual responsif, barang nya oke banget

Normalisasi data adalah tahap penting dalam pra-pemrosesan data yang bertujuan untuk menghasilkan konsistensi dalam representasi data. Dengan melakukan normalisasi, kita dapat menghindari ambiguitas dan meningkatkan akurasi analisis data. Selain itu, normalisasi membantu dalam menyediakan input yang lebih baik untuk model pembelajaran mesin, seperti SVM dan indoBert, karena mengurangi variasi dalam data yang mungkin membingungkan model. Dengan demikian, normalisasi data membantu meningkatkan kualitas dan konsistensi data, yang pada gilirannya dapat meningkatkan kinerja model yang dibangun berdasarkan data tersebut.

```
norm= {" dgn " : " dengan ", "quality":"kualitas", 'baguss':'bagus'}

def normalisasi(text):
  for i in norm:
    text = text.replace(i, norm[i])
  return text
```

Gambar 3. 10 Code *Normalize*

3.2.6 Stopword

Pada tahap ini, dilakukan penghapusan kata-kata yang kurang relevan atau sering muncul dalam teks, yang dikenal sebagai stopword. Stopword ini mencakup kata-kata penghubung dan kata-kata keterangan umum yang tidak memberikan kontribusi signifikan terhadap makna atau inti teks, seperti "sebuah", "oleh", "pada", dan lainnya. Proses ini bertujuan untuk menyederhanakan representasi teks, memfokuskan pada kata-kata kunci, dan mengurangi dimensi yang tidak perlu dalam analisis data, sehingga memungkinkan model untuk lebih fokus pada informasi yang penting. Adapun contoh dari tahap *stopword* dapat dilihat pada table di bawah.

Tabel 3. 2 Contoh *Stopword*

Hasil *Normalize*	Hasil *Stopword*
penjual responsif, barang nya oke banget	penjual responsif, barang oke banget

```
def stopword(text):
    stop_words = set(stopwords.words('indonesian'))
    words = text.split()
    filtered_words = [word for word in words if word.casefold() not in stop_words]
    cleaned_text = ' '.join(filtered_words)
    return cleaned_text
```

Gambar 3. 11 Code *Stopword*

Tabel 3. 3 List *Stopword*

Kata	Kata	Kata	Kata
adalah	banyak	bermula	di
adanya	bapak	bersama	dia
adapun	baru	bersama-sama	diakhiri
agak	bawah	bersiap	diakhirinya
agaknya	beberapa	bersiap-siap	dialah
agar	begini	bertanya	diantara
akan	beginian	bertanya-tanya	diantaranya
akankah	beginikah	berturut	diberi
akhir	beginilah	berturut-turut	diberikan
akhiri	begitu	bertutur	diberikannya
akhirnya	begitukah	berujar	dibuat
aku	begitulah	berupa	dibuatnya
akulah	begitupun	besar	didapat
amat	bekerja	betul	didatangkan
amatlah	belakang	betulkah	digunakan
anda	belakangan	biasa	diibaratkan
andalah	belum	biasanya	diibaratkannya
antar	belumlah	bila	diingat

antara	benar	bilakah	diingatkan
antaranya	benarkah	bisa	diinginkan
apa	benarlah	bisakah	dijawab
apaan	berada	boleh	dijelaskan
apabila	berakhir	bolehkah	dijelaskannya
apakah	berakhirlah	bolehlah	dikarenakan
apalagi	berakhirnya	buat	dikatakan
apatah	berapa	bukan	dikatakannya
artinya	berapakah	bukankah	dikerjakan
asal	berapalah	bukanlah	diketahui
asalkan	berapapun	bukannya	diketahuinya
atas	berarti	bulan	dikira
atau	berawal	bung	dilakukan
ataukah	berbagai	cara	dilalui
ataupun	berdatangan	caranya	dilihat
awal	beri	cukup	dimaksud
awalnya	berikan	cukupkah	dimaksudkan
bagai	berikut	cukuplah	dimaksudkannya
bagaikan	berikutnya	cuma	dimaksudnya
bagaimana	berjumlah	dahulu	diminta
bagaimanakah	berkali-kali	dalam	dimintai
bagaimanapun	berkata	dan	dimisalkan
bagi	berkehendak	dapat	dimulai
bagian	berkeinginan	dari	dimulailah
bahkan	berkenaan	daripada	dimulainya
bahwa	berlainan	datang	dimungkinkan
bahwasanya	berlalu	dekat	dini
baik	berlangsung	demi	dipastikan
bakal	berlebihan	demikian	diperbuat
bakalan	bermacam	demikianlah	diperbuatnya
balik	bermacam-macam	dengan	dipergunakan

3.2.7 Tokenizing

Pada tahap *tokenizing*, dokumen diubah menjadi serangkaian term dengan menghapus semua karakter tanda baca yang ada pada token. Tujuan dari proses ini adalah untuk menghasilkan kumpulan kata-kata yang merupakan representasi dari teks atau dokumen tersebut. Dengan demikian, output yang dihasilkan adalah sekumpulan kata-kata yang membentuk inti dari teks, tanpa kehadiran tanda baca yang mungkin tidak relevan dalam analisis atau pemrosesan berikutnya. Adapun contoh dari tahap *tokenizing* dapat dilihat pada table di bawah.

Tabel 3. 4 Contoh *Tokenizing*

Hasil *Stopword*	**Hasil *Tokenizing***
penjual responsif, barang oke banget	penjual | responsif | barang | oke | banget

Tokenizing adalah proses memecah teks menjadi unit kata. Ini dilakukan dengan menggunakan karakter whitespace seperti enter, tabulasi, dan spasi sebagai pemisah kata. Namun, karakter tunggal seperti tanda kutip tunggal ('), titik (.), semikolon (;), titik dua (:), dan lainnya juga dapat berperan sebagai pemisah kata, tergantung pada konteksnya. Proses tokenisasi ini penting untuk mempersiapkan teks agar dapat diolah lebih lanjut, dengan menghasilkan kumpulan kata-kata yang mewakili teks tersebut dengan tepat.

Gambar 3. 12 Code *Tokenizing*

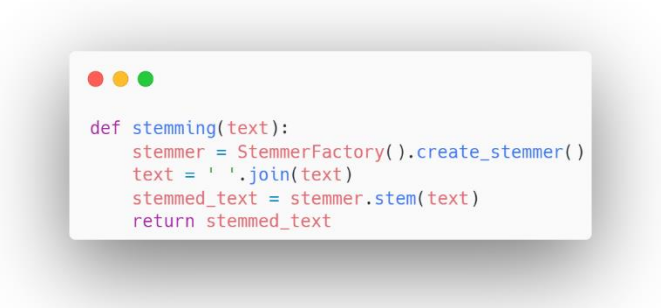
3.2.8 Stemming

Stemming merupakan tahap dalam pra-pemrosesan teks yang bertujuan untuk menghapus imbuhan, awalan, dan akhiran dari kata-kata guna mengubahnya menjadi bentuk dasarnya. Proses ini membantu dalam menghasilkan representasi yang lebih konsisten dari kata-kata dalam teks, memungkinkan model untuk lebih mudah mengenali dan memahami

makna kata-kata yang sebenarnya. Adapun contoh dari tahap *stemming* dapat dilihat pada table di bawah.

Tabel 3. 5 Contoh *Stemming*

Hasil *Tokenizing*	Hasil *Stemming*
penjual | responsif | barang | oke | banget	jual | responsif | barang | oke | banget



```
def stemming(text):
    stemmer = StemmerFactory().create_stemmer()
    text = ' '.join(text)
    stemmed_text = stemmer.stem(text)
    return stemmed_text
```

Gambar 3. 13 Code *Stemming*

3.2.9 Pembobotan Kata

Dalam klasifikasi sentiment ulasan tokopedia, pembobotan kata digunakan untuk mendapatkan suatu kategori. Salah satu metode pembobotan adalah TF-IDF *(Term Frequency –Inverse Document Frequency)*.

Term Weighting TI-IDF adalah salah satu pembobotan yang sering digunakan dan merupakan gabungan dari *Term Frequency dan Inverse Document Frequency*. TF-IDF terdiri dari frekuensi term dan inverse dokumen yang didapatkan dari membagi seluruh jumlah dokumen terhadap jumlah dokumen yang memiliki term tersebut. Dalam TI-IDF bobot akan ditemukan dalam persamaan berikut:

$$Wi,j = tfi,j\ X \log\left(\frac{N}{dfi}\right) \qquad (3.\ 1)$$

Keterangan:

{tf}_{i,j} = Bobot dari istilah i dalam dokumen j

{df}_i = Frekuensi munculnya istilah i dalam dokumen j

N = Total istilah pada dokumen

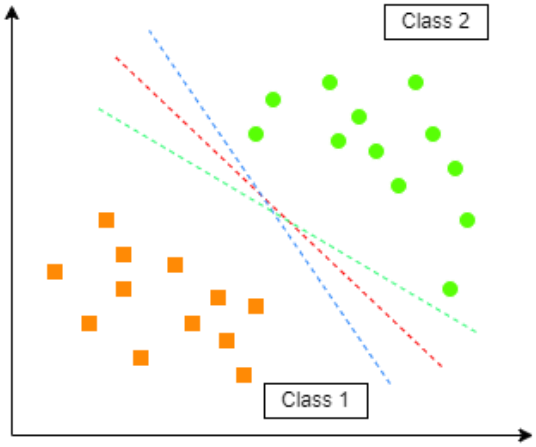
3.2.10 Hierarchical Dirichlet Process (HDP)

isi kalimat…

3.2.11 Support Vector Machine

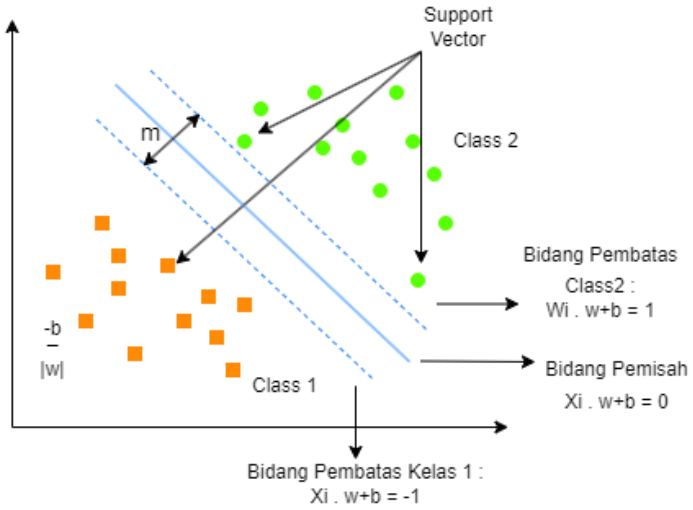
Support Vector Machine (SVM) adalah teknik yang digunakan untuk melakukan prediksi, baik pada kasus klasifikasi maupun regresi. Prinsip dasar SVM adalah pemisah linier, yaitu kemampuan untuk mengklasifikasikan data yang secara linier dapat dipisahkan. Namun, SVM telah dikembangkan untuk bekerja pada masalah non-linier dengan mengadopsi konsep kernel, memungkinkan model bekerja dalam ruang berdimensi tinggi [2].

Pada Gambar 3.11, dijelaskan konsep dasar algoritma *Support Vector Machine* (SVM), yang berfokus pada mencari hyperplane terbaik sebagai pemisah antara dua kelas dalam sebuah data.



Gambar 3. 14 Klasifikasi Linear SVM

Pada input space terdapat dua kelas yang berbeda, +1 dan -1, beserta masing-masing pattern yang digambarkan dengan simbol kotak warna orange untuk pattern -1 dan simbol lingkaran hijau untuk pattern +1.



Gambar 3. 15 Detail Klasifikasi Linear SVM

Pada gambar 3.7 dijelaskan dalam mengklasifikasi untuk mendapat hasil yang baik hyperplane digunakan untuk memisahkan menjadi dua kelas dengan mengukur margin *hyperplane* tersebut dan mencari titik maksimalnya. Margin adalah jarak antara hyperplane terdekat dengan pattern terdekat dari masing-masing kelas dan pattern yang paling dekat dengan *hyperplane* disebut *support vector*. Seperti gambar dibawah garis tidak putus-putus yangterletak tepat di tengah-tengah kedua keas. Sedangkan *support vector* tampak sebagai *pattern* yang berpotongan dengan garis putus-putus. Dari Gambar 3.6 bidang pemisah dapat dirumuskan :

m = jarak antara dua bidang

w = bidang normal

b = posisi relative terhadap origin

jarak garis dirumuskan wx+b=c ke origin adalah (c-b)/|w|

$$m = \frac{1 - b - (-1 - b)}{|w|} = \frac{2}{|w|} \qquad (3.\ 2)$$

Margin m dimaksimalkan dengan memenuhi konstrain 2 bidang pembatas yang sejajar dan data yang ada pada bidang pembatas disebut *support vector.* Bidang pembatas kelas pertama membatasi kelas pertama sedangkan bidang pembatas kelas kedua membatasi kelas kedua. Sehingga diperoleh :

$$x\ i.w + b \geq +1\ for\ yi = +1 \qquad (3.\ 3)$$

$$x\ i.w + b \geq +1\ for\ yi = -1 \quad (3.4)$$

Nilai maksimal margin harus memenuhi rumus di atas dan nilai b dan w dikalikan dengan sebuah konstanta yang akan menghasilkan nilai margin yang dikalikan dengan konstanta yang sama. Konstrain merupakan scaling constraint dengan dipenuhi rescaling b dan w. Karena maksimalkan dan minimalkan w dirumuskan dengan pertidaksamaan rumus di atas.

$$yi\,(x\ i.w + b) - 1 \geq 0 \quad (3.5)$$

Dengan mengalikan b dan sebuah konstanta, maka menghasilkan nilai m kemudian dikalikan dengan konstanta yang sama. Konstrain merupakan scaling constraint yang dipenuhi dengan rescaling b dan w. Maksimalkan $\frac{1}{|w|}$ minimumkan $|w|^2$.

Untuk mencari nilai margin terbesar untuk bidak pemisah terbaik dapat dirumuskan menjadi masalah optimasi konstrain, yaitu :

$$min\frac{1}{2}\ |w|^2 \quad (3.6)$$

$$s.t\ yi(xi.w + b) - 1 \geq 0 \quad (3.7)$$

Dengan lebih mudah untuk menyelesaikan permasalah optimasi konstrain dalam formulasinya dirubah kedalam formula lagrangian yang menggunakan lagrange multiplier yang diubah menjadi :

$$min\ (w.b.a) = \frac{1}{2}\ |w| - \sum_{i=1}^{n} a_i y_i (x_i w + b) \sum_{i=1}^{n} a_i \quad (3.8)$$

Formula pencarian bidang pemisah terbaik ini adalah permasalahan *quadratic programming,* sehingga nilai maksimum global dari a_i akan selalu dapat ditemukan setelah solusi permasalahan *quadratic programming* ditemukan (nilai a_i), maka kelas dari data pengujian x dapat ditentukan berdasarkan nilai dari fungsi keputusan :

$$f(Xd) = \sum_{i=1}^{ns} ai\ yi\ xi\ xd + b \quad (3.9)$$

xi = *support vector,*

Ns = jumlah *support vector*

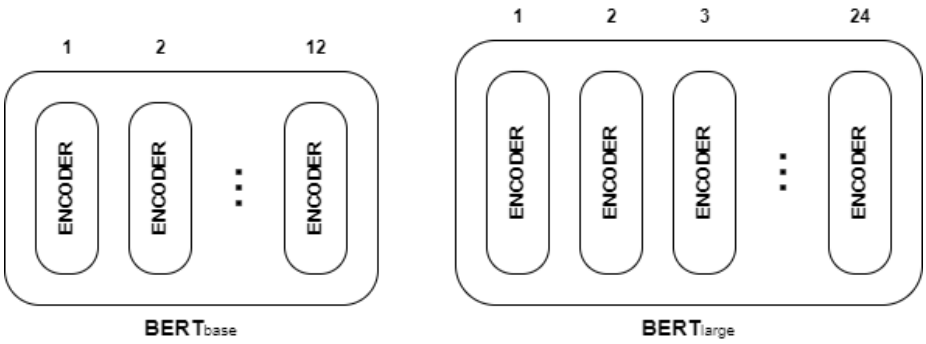
xd = data yang akan diklasifikasikan

3.2.12 IndoBert

Model word embedding yang telah dilatih sebelumnya, dikenal sebagai model pre-trained word embedding, dirancang untuk meningkatkan pemahaman makna dan sintaksis dari teks. Model-model ini dilatih menggunakan dataset besar yang beragam agar dapat mengenali pola bahasa secara luas. Pada tahun 2018, model canggih bernama Bidirectional Encoder Representations from Transformers (BERT) diperkenalkan dan berhasil mencapai hasil unggul dalam berbagai penelitian di bidang pemrosesan bahasa alami (NLP). BERT memanfaatkan arsitektur Transformer dengan mekanisme self-attention untuk memahami konteks hubungan antara kata-kata dalam teks. Di Indonesia, perkembangan signifikan terjadi pada tahun 2020 dengan hadirnya model pre-trained BERT yang dikenal sebagai IndoBERT. Model ini khusus disesuaikan untuk bahasa Indonesia, memungkinkan pemahaman yang lebih baik terhadap bahasa lokal dan meningkatkan performa model dalam tugas-tugas NLP berbahasa Indonesia [1].

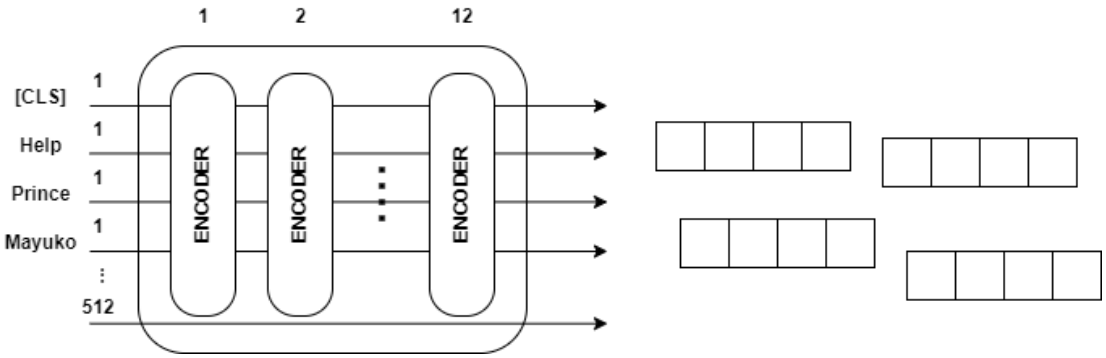
Model ini bertujuan untuk meminimalkan gabungan fungsi kerugian dari Masked LM dan Next Sentence Prediction, sehingga menghasilkan model bahasa yang kuat dengan kemampuan yang ditingkatkan dalam memahami konteks dalam kalimat dan hubungan antar kalimat. Ada beberapa hal pada arsitektur BERT, diantaranya :

- BERT BASE memiliki 1 2 lapisan di tumpukan Encoder sedangkan BERT LARGE memiliki 24 lapisan di tumpukan Encoder . Ini lebih dari arsitektur Transformer yang dijelaskan dalam makalah asli (6 lapisan encoder).
- Arsitektur BERT (BASE dan LARGE) juga memiliki jaringan feedforward yang lebih besar (masing-masing 768 dan 1024 unit tersembunyi), dan lebih banyak perhatian (masing-masing 12 dan 16) daripada arsitektur Transformer yang disarankan dalam makalah asli. Ini berisi 512 unit tersembunyi dan 8 kepala perhatian.
- BERT BASE berisi 110 juta parameter sedangkan BERT LARGE memiliki 340 juta parameter.



Gambar 3. 16 Arsitektur Bert Base & Bert Large

Model ini memproses masukan dimulai dengan token CLS, yang merupakan token klasifikasi, diikuti oleh serangkaian kata sebagai masukan. Token CLS ini berfungsi sebagai penanda awal untuk pemrosesan. Masukan kemudian diteruskan ke lapisan-lapisan di atasnya. Setiap lapisan menggunakan mekanisme perhatian mandiri dan hasilnya kemudian diteruskan ke jaringan feedforward sebelum akhirnya diserahkan ke pembangun enkode berikutnya. Model ini menghasilkan vektor dengan ukuran tersembunyi (768 untuk BERT BASE). Jika kita ingin menggunakan model ini untuk klasifikasi, kita dapat menggunakan keluaran yang terkait dengan token CLS.

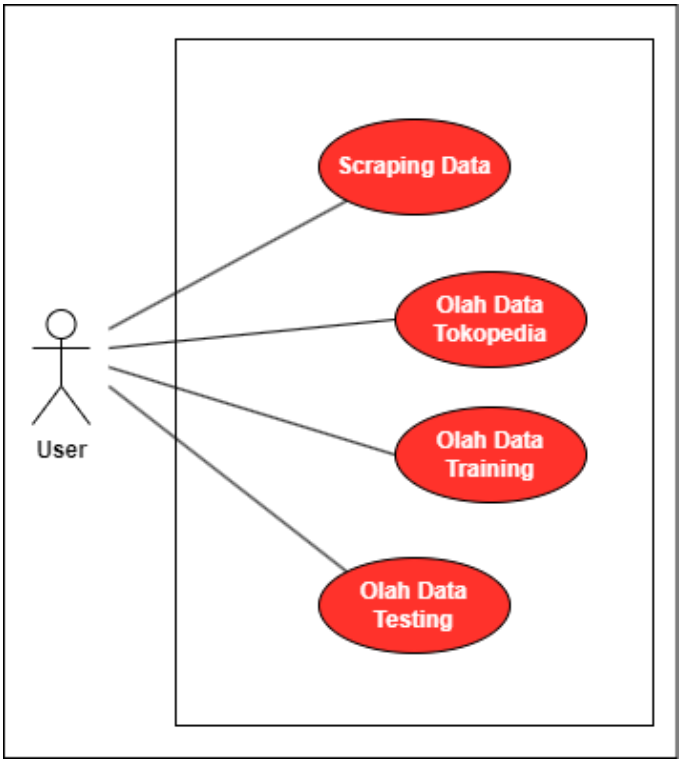


Gambar 3. 17 Bert Embeddings

3.3 Perancangan Sistem

3.3.1 Use Case Diagram

Berikut ini adalah use case diagram pada sistem analisis sentimen terhadap data ulasan Hanafashion_shop di Tokopedia :



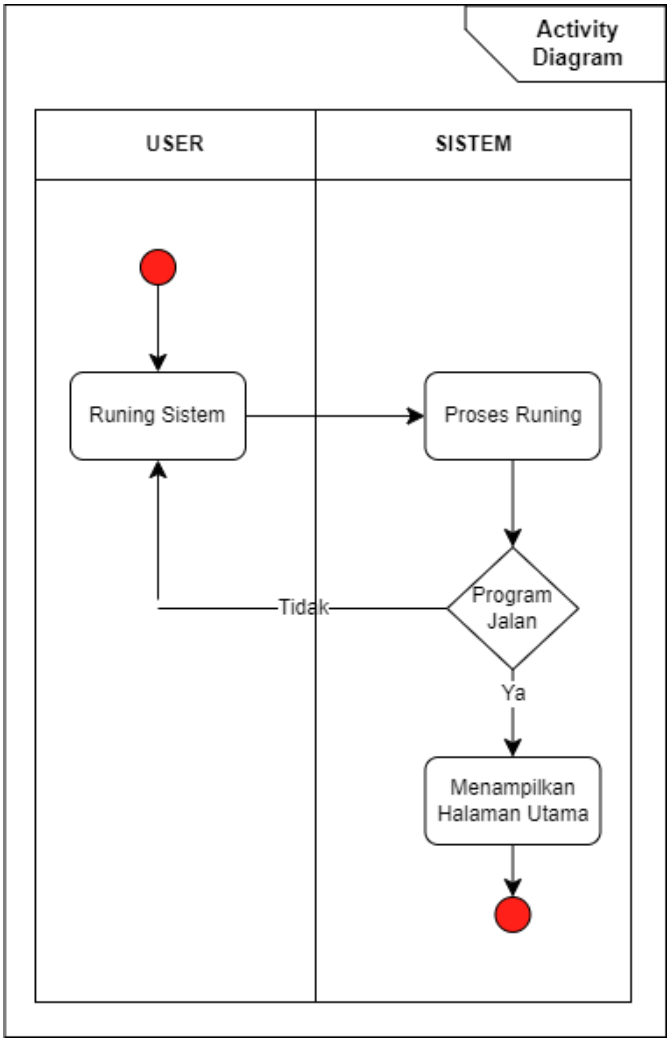
Gambar 3. 18 *Use Case* Diagram

3.3.2 Activity Diagram

Activity diagram merupakan alur aktivitas pengguna terhadap system. Dengan adanya *activity diagram* dapat mengetahui alur interaksi yang terjadi pada *use case diagram* :

1. *Activity Diagram Running System*

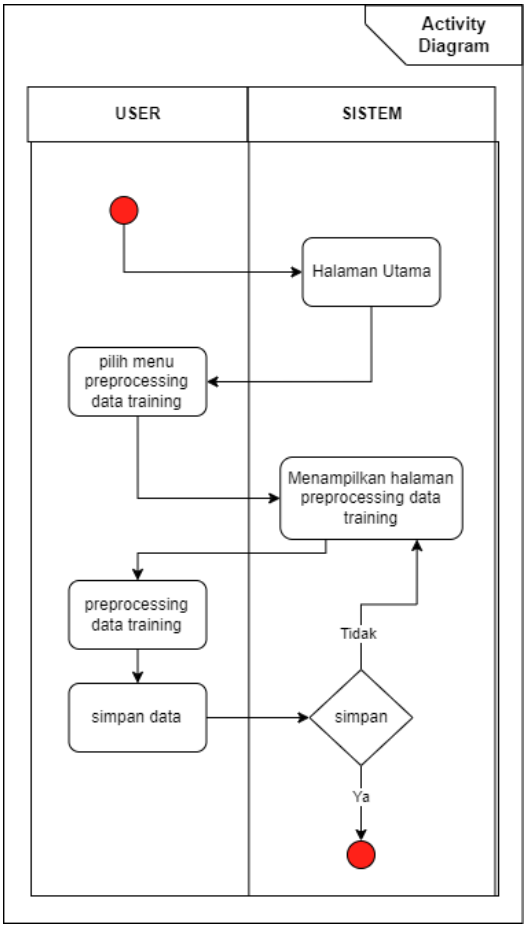
   Diagram aktifitas ini menggambarkan aktifitas *system interface* menggunakan streamlit dijalankan. Adapun alur aktivitas proses running system dapat dilihat pada gambar di bawah.



Gambar 3. 19 *Activity Diagram Running System*

2. *Activity Diagram Preprocessing*

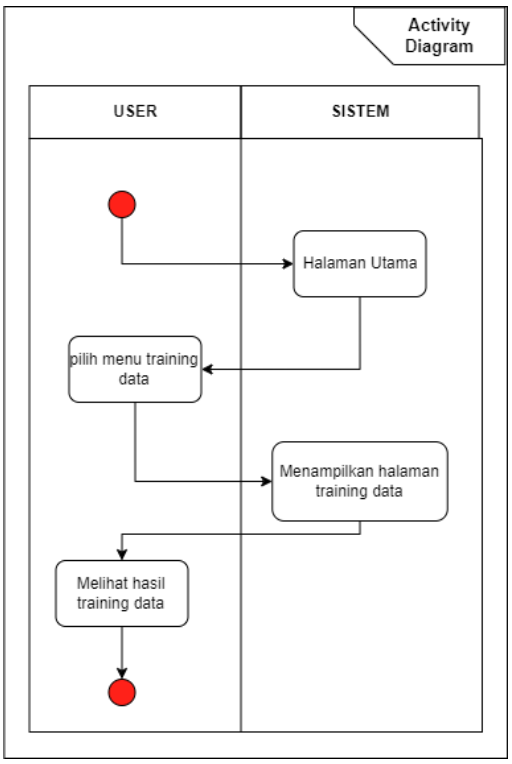
   Diagram aktivitas ini menggambarkan *preprocessing* data dari ulasan Hanafashion_shop yang ada di Tokopedia sebelum dilakukan proses training data menggunakan algoritma *Support Vector Machine* dan *indoBert*. Adapun alur aktivitas *preprocessing* dapat dilihat pada gambar di bawah.



Gambar 3. 20 *Activity Diagram Preprocessing*

3. *Activity Diagram Training Data*

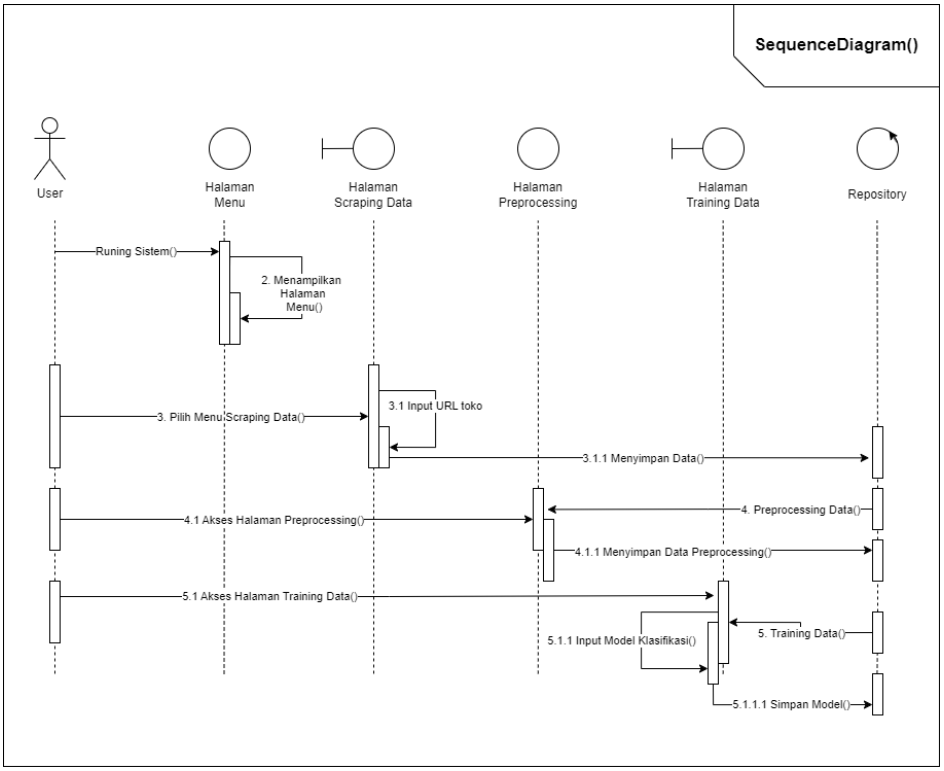
   Diagram aktivitas ini menggambarkan tahapan training data menggunakan algoritma *Support Vector Machine* dan *indoBert*, user bisa memasukkan input parameter sebelum melakukan training data. Adapun alur aktivitas *training data* dapat dilihat pada gambar di bawah.



Gambar 3. 21 *Activity Diagram Training Data*

3.3.3 Sequence Diagram

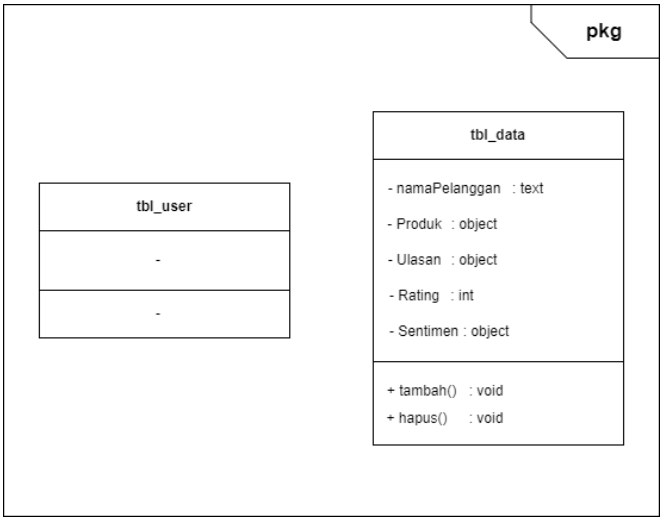
Diagram sequence menggambarkan detail alur proses berdasarkan urutan waktu. Dalam proses analisis sentimen, pengguna memulai dengan mengakses menu aplikasi yang tersedia. Sistem kemudian melakukan *scraping* data dari sumbernya, seperti media sosial, lalu melakukan *preprocessing* untuk membersihkan dan mempersiapkan data teks. Setelah itu, sistem melatih model analisis sentimen menggunakan data yang telah diproses dan menyimpannya ke dalam repository untuk penggunaan dan referensi di masa depan. Adapun tampilan proses *sequence diagram* dapat dilihat pada gambar di bawah.



Gambar 3. 22 *Sequence Diagram*

3.3.4 Class Diagram

Berikut ini adalah *class diagram* dari analisis sentiment pada toko Hanafashion_shop dari platform Tokopedia menggunakan algoritma Support Vector Machine dan IndoBert.



Gambar 3. 23 *Class Diagram*

BAB IV
HASIL DAN PEMBAHASAN

4.1 Environment Testing

Hasil dari langkah ini adalah persiapan perangkat keras dan perangkat lunak yang akan digunakan untuk mendesain sistem, mengembangkan sistem, dan mengujinya. Perangkat pengembangan yang dipakai termasuk:

1. Kebutuhan minimum perangkat keras

   Perangkat minimum yang dibutuhkan untuk dapat menjalankan system perangkat computer atau laptop dengan detail spesifikasi sebagai berikut :

   a. Intel Core i5-10351G
   b. Memory RAM 8 GB
   c.

2. Kebutuhan perangkat lunak
   a. Sistem operasi *Microsoft Windows* 11
   b. *Google Chrome*
   c. *Visual Studio Code*
   d. *Python & Library*

4.2 Deskripsi Data

Penjelasan…

4.2.1 Data Hasil Scraping

Penjelasan…

4.2.2 Data Training Model

Penjelasan…

4.3 Hasil Perbandingan Model

Untuk mengevaluasi kinerja dalam studi ini, peneliti menggunakan metode *confusion matrix* untuk menghitung recall, presisi, dan akurasi dari setiap kategori. Dari total 996 ulasan yang diambil, peneliti melakukan pembagian data menjadi dua bagian: 80% untuk data pelatihan dan 20% untuk data pengujian. Berikut adalah hasil evaluasi kinerja dari data pengujian yang kami peroleh menggunakan *confusion matrix*.

4.3.1 Hasil Model *Support Vector Machine*

Dalam eksperimen menggunakan algoritma *Support Vector Machine* dengan splitting data 80/20, confusion matrix yang dihasilkan menunjukkan bahwa dari total 190 sampel,

terdapat 5 prediksi positif yang benar (True Positives), 2 prediksi positif yang salah (False Positives), 165 prediksi negatif yang benar (True Negatives), dan 18 prediksi negatif yang salah (False Negatives). Evaluasi kinerja model dilakukan dengan menghitung metrik seperti akurasi, presisi, recall, dan F1-score, yang menyediakan pemahaman yang lebih holistik tentang kecocokan model terhadap data uji. Adapun tabel dan perhitungan evaluasi dapat dilihat sebagai berikut :

Tabel 4. 1 *Test Result Support Vector Machine*

Actual Class	Positif	Negatif
Positif	5	2
Negatif	18	165

Positif :

$$Recall = \frac{5}{5+2} = 0.71$$

$$Precision = \frac{5}{5+18} = 0.22$$

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision} + \frac{2 * 0.71 * 0.22}{0.71 + 0.22} = 0.33$$

Negatif :

$$Recall = \frac{165}{165+18} = 0.90$$

$$Precision = \frac{165}{165+2} = 0.98$$

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision} + \frac{2 * 0.90 * 0.98}{0.90 + 0.98} = 0.93$$

Accuracy :

$$Accuracy = \frac{5+165}{5+2+18+165} * 100\% = 0.89\%$$

4.3.2 Hasil Model *IndoBert*

Dalam eksperimen menggunakan model *IndoBERT* dengan splitting data 80/20, confusion matrix yang dihasilkan menunjukkan bahwa dari total 190 sampel, terdapat 154

prediksi positif yang benar (True Positives), 13 prediksi positif yang salah (False Positives), 11 prediksi negatif yang benar (True Negatives), dan 12 prediksi negatif yang salah (False Negatives). Evaluasi kinerja model dilakukan dengan menghitung metrik seperti akurasi, presisi, recall, dan F1-score, yang memberikan gambaran menyeluruh tentang kemampuan model IndoBERT dalam mengklasifikasikan data uji. Adapun tabel dan perhitungan evaluasi dapat dilihat sebagai berikut :

Tabel 4. 2 *Test Result IndoBert*

Actual Class	Positif	Negatif
Positif	154	13
Negatif	12	11

Positif :

$$Recall = \frac{154}{154 + 13} = 0.92$$

$$Precision = \frac{154}{154 + 12} = 0.93$$

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision} + \frac{2 * 0.92 * 0.93}{0.92 + 0.93} = 0.92$$

Negatif :

$$Recall = \frac{11}{11 + 12} = 0.48$$

$$Precision = \frac{11}{11 + 13} = 0.46$$

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision} + \frac{2 * 0.48 * 0.46}{0.48 + 0.46} = 0.23$$

Accuracy :

$$Accuracy = \frac{154 + 11}{154 + 13 + 12 + 11} * 100\% = 0.87\%$$

4.4 Analisa Model Terbaik

Penjelasan...

Tabel 4. 3 Perbandingan Performa Model

Performa Model	***Accuracy***	***Precision***	***Recall***	***F-Measure***
Support Vector Machine	**89%**	**98%**	**90%**	**93%**
IndoBert	87%	93%	92%	92%

4.5 Prosedur Penggunaan Aplikasi

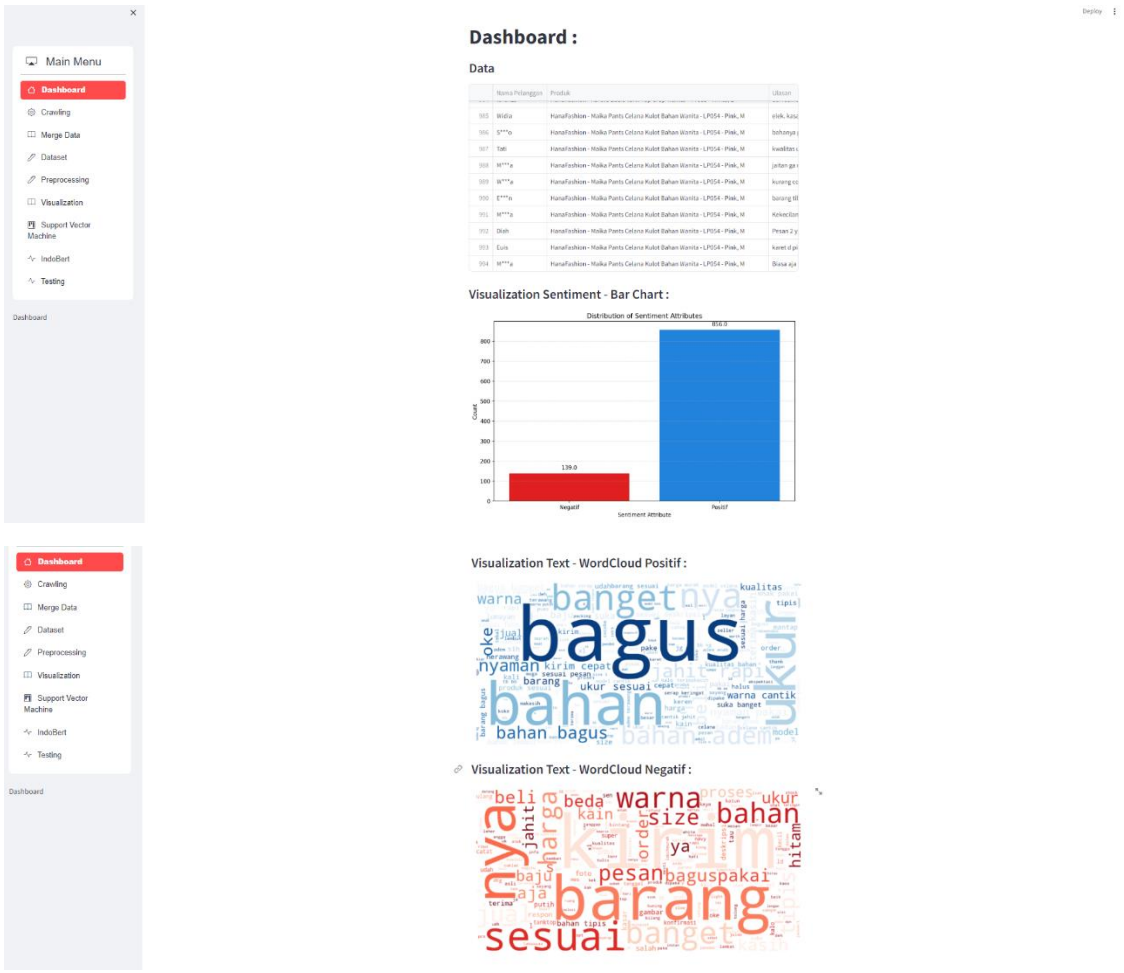
Tambahkan penjelasan...

4.5.1 Halaman Dashboard

Pada halaman dashboard, dilakukan proses untuk menampilkan data dan visualisasi ulasan dari Hanafashion_shop yang kita dapatkan dari Tokopedia dengan menggunakan Python dan Selenium. Data yang kita tampilkan adalah untuk 10 produk dengan jumlah ulasan terbanyak dari toko Hanafashion_shop. Sentimen dari ulasan ditampilkan berdasarkan rating pada masing-masing produk, sementara wordcloud digunakan untuk memvisualisasikan teks dari ulasan berdasarkan atribut tertentu pada data Hanafashion_shop..Adapun script dan tampilan dashboard dapat dilihat pada gambar di bawah.

```
if selected == 'Dashboard':
    st.title("Dashboard :")
    st.subheader("Data")
    df_dashboard = pd.read_csv("data/dataHasilPenggabungan/dataSentimenProduk1-10.csv")
    df_dashboard = df_dashboard[['Nama Pelanggan', 'Produk', 'Ulasan', 'Rating']]
    dfVisualization = pd.read_csv("data/dataHasilPreprocessing/hasilPreprocessing1.csv")
    if 'Ulasan' not in df_dashboard.columns:
        st.warning("Data yang dimasukkan tidak sesuai.")
    else:
        st.dataframe(df_dashboard)

    with st.spinner('Performing Visualization...'):
        if 'Sentimen' not in dfVisualization.columns:
            st.warning("Data yang dimasukkan tidak sesuai.")
        else:
            st.subheader("Visualization Sentiment - Bar Chart :")
            custom_palette = {'Negatif': 'red', 'Positif': '#0384fc'}
            plt.figure(figsize=(10, 6))

            ax = sns.countplot(x='Sentimen', data=dfVisualization, order=['Negatif', 'Positif'],
palette=custom_palette)
            ax.grid(axis='y', linestyle='--', alpha=0.5)
            plt.title('Distribution of Sentiment Attributes')
            plt.xlabel('Sentiment Attribute')
            plt.ylabel('Count')
            for p in ax.patches:
                ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                            ha='center', va='center', xytext=(0, 10), textcoords='offset points')
            st.pyplot(plt)
```

Gambar 4. 1 Script Tampilan Dashboard



Gambar 4. 2 Tampilan Dashboard

Gambar di atas menunjukkan visualisasi untuk sentiment dari data dan wordcloud ulasan. Visualisasi sentiment menunjukkan ketidakseimbangan antara sentiment positif dan negatif, dikarenakan mayoritas rating produk Hanafashion_shop cenderung tinggi. Dalam wordcloud untuk visualisasi teks positif, kata yang paling dominan adalah 'bagus', sedangkan untuk visualisasi teks negatif, kata yang muncul dominan adalah 'barang'.

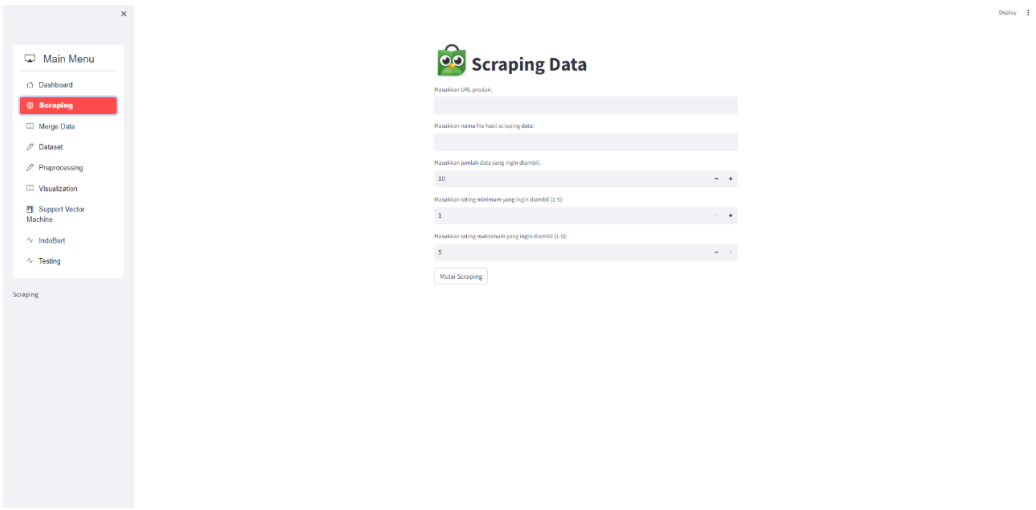
4.5.2 Halaman Scraping

Pada tahap implementasi, user bisa melakukan proses scraping data ulasan toko Hanafashion_shop di Tokopedia menggunakan bahasa pemrograman Python dan library Selenium. Selenium digunakan untuk mengendalikan browser dan melakukan tugas-tugas seperti mengklik, memasukkan teks, dan menelusuri halaman web. Dengan menggunakan Selenium, sistem dapat mengakses halaman ulasan toko Pengrajin.com, menemukan elemen HTML yang berisi ulasan, dan mengekstrak teks ulasan tersebut. Selanjutnya, data ulasan yang

berhasil diambil akan disimpan dalam format yang sesuai untuk proses analisis sentimen selanjutnya. Adapun script dan tampilan scraping dapat dilihat pada gambar di bawah.

```
elif selected == "Scraping":
    col1, col2 = st.columns([1,8])
    with col1:
        st.image('img/tokopedia.png', width=80)
    with col2:
        st.title("Scraping Data")

    url = st.text_input("Masukkan URL produk:")
    nama_file = st.text_input("Masukkan nama file hasil scraping data:")
    jumlah_data = st.number_input("Masukkan jumlah data yang ingin diambil:", min_value=1, step=1,
value=10)
    rating_min = st.number_input("Masukkan rating minimum yang ingin diambil (1-5):", min_value=1,
max_value=5, step=1, value=1)
    rating_max = st.number_input("Masukkan rating maksimum yang ingin diambil (1-5):", min_value=1,
max_value=5, step=1, value=5)
    folder_path = "data/dataScrapingHanaShop/"
    file_path = folder_path + nama_file

    if st.button("Mulai Scraping"):
        scrapingFunction.scrape_tokopedia_reviews(url, jumlah_data, file_path, rating_min, rating_max)
        st.success(f"Data telah disimpan ke: {file_path}.csv")
```

Gambar 4. 3 Script Tampilan Scraping



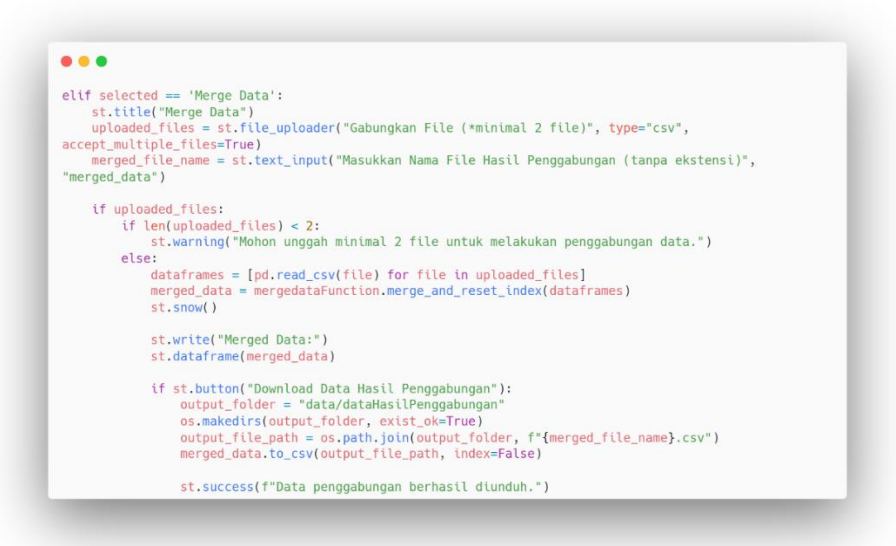
Gambar 4. 4 Tampilan Scraping

Implementasi halaman scraping di atas memanfaatkan fitur input URL, nama file, jumlah baris data, dan rentang rating produk yang ingin diambil. Pengguna diminta untuk memasukkan URL toko Hanafashion_shop untuk menentukan produk yang akan diambil. Selanjutnya, mereka diminta untuk menentukan nama file tempat hasil scraping disimpan, jumlah baris data yang ingin diambil, dan rentang rating dari 1 hingga 5. Proses scraping bertujuan untuk mengumpulkan informasi penting seperti Nama Pelanggan, Nama Produk,

Ulasan, dan Rating. Halaman scraping menampilkan kolom input URL dan gambaran visual dari elemen-elemen yang akan diambil dari toko Hanafashion_shop.

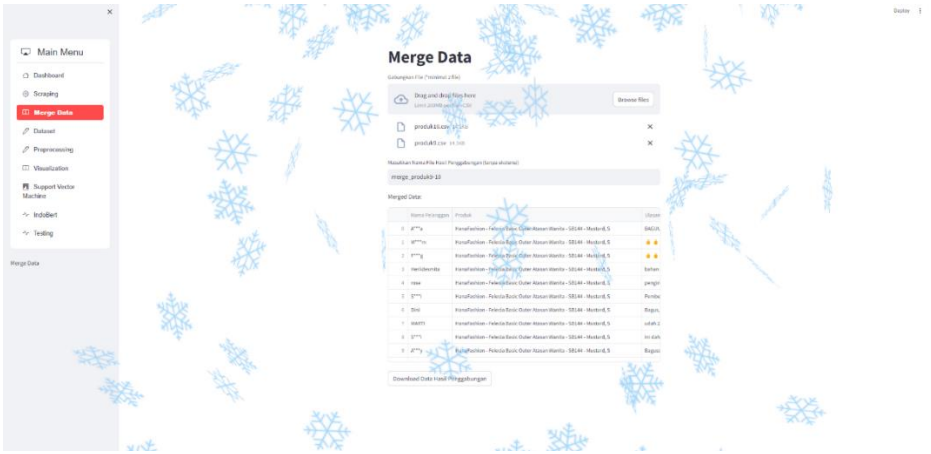
4.5.3 Halaman Merge Data

Pada tahap implementasi, user bisa menggabungkan data dari berbagai sumber menjadi satu set data yang lengkap dan terpadu. Dengan cara memetakan kolom-kolom dari masing-masing sumber dan menggunakan kriteria penggabungan yang ditentukan, seperti kunci primer atau kolom tertentu, halaman ini memungkinkan untuk menggabungkan baris-baris data yang sesuai. Tujuannya adalah untuk menciptakan satu set data yang komprehensif yang dapat digunakan untuk analisis lebih lanjut atau pelaporan, memungkinkan pengguna untuk mendapatkan pemahaman yang lebih baik tentang situasi atau tren yang ada dan membuat keputusan yang lebih terinformasi. Adapun script dan tampilan merge data dapat dilihat pada gambar di bawah.



```
elif selected == 'Merge Data':
    st.title("Merge Data")
    uploaded_files = st.file_uploader("Gabungkan File (*minimal 2 file)", type="csv",
accept_multiple_files=True)
    merged_file_name = st.text_input("Masukkan Nama File Hasil Penggabungan (tanpa ekstensi)",
"merged_data")

    if uploaded_files:
        if len(uploaded_files) < 2:
            st.warning("Mohon unggah minimal 2 file untuk melakukan penggabungan data.")
        else:
            dataframes = [pd.read_csv(file) for file in uploaded_files]
            merged_data = mergedataFunction.merge_and_reset_index(dataframes)
            st.snow()

            st.write("Merged Data:")
            st.dataframe(merged_data)

            if st.button("Download Data Hasil Penggabungan"):
                output_folder = "data/dataHasilPenggabungan"
                os.makedirs(output_folder, exist_ok=True)
                output_file_path = os.path.join(output_folder, f"{merged_file_name}.csv")
                merged_data.to_csv(output_file_path, index=False)

                st.success(f"Data penggabungan berhasil diunduh.")
```

Gambar 4. 5 Script Merge Data



Gambar 4. 6 Tampilan Merge Data

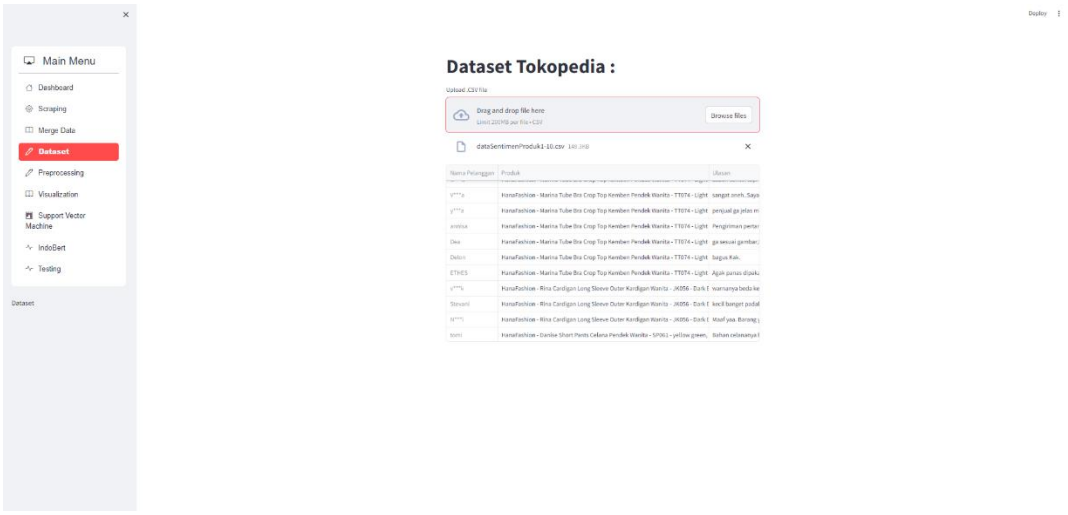
Implementasi halaman merge data di atas melibatkan penggabungan data untuk produk 9 dan produk 10 dari toko Hanafashion_shop. Hasil penggabungan data ini dapat diunduh oleh user dan disimpan di komputer lokal mereka. Data yang dihasilkan dari penggabungan ini akan digunakan untuk proses lebih lanjut di halaman preprocessing, memungkinkan pengguna untuk melakukan persiapan data yang diperlukan sebelum melakukan analisis lebih lanjut.

4.5.4 Halaman Dataset

Pada tampilan dataset, user diberi akses untuk menampilkan dataset dari berbagai sumber dan proses dengan format .csv. Tujuannya adalah agar pengguna dapat memiliki gambaran menyeluruh tentang data yang akan mereka proses di tahap berikutnya. Dengan demikian, mereka dapat mengidentifikasi keberadaan data kosong atau duplikat, mempersiapkan langkah-langkah pembersihan yang diperlukan di tahap *preprocessing* data. Adapun script dan tampilan dataset dapat dilihat pada gambar di bawah.

```
elif selected == "Dataset":
    st.title("Dataset Tokopedia :")
    uploaded_file = st.file_uploader("Upload .CSV file", type=["csv"])
    if uploaded_file is not None:
        try :
            df = pd.read_csv(uploaded_file, dtype={"Rating":"object"}, index_col=0)
            st.dataframe(df)
        except pd.errors.EmptyDataError:
            st.write("File is empty, please check your input.")
        except pd.errors.ParserError:
            st.write("Invalid data format, please check your input.")
```

Gambar 4. 7 Script Dataset



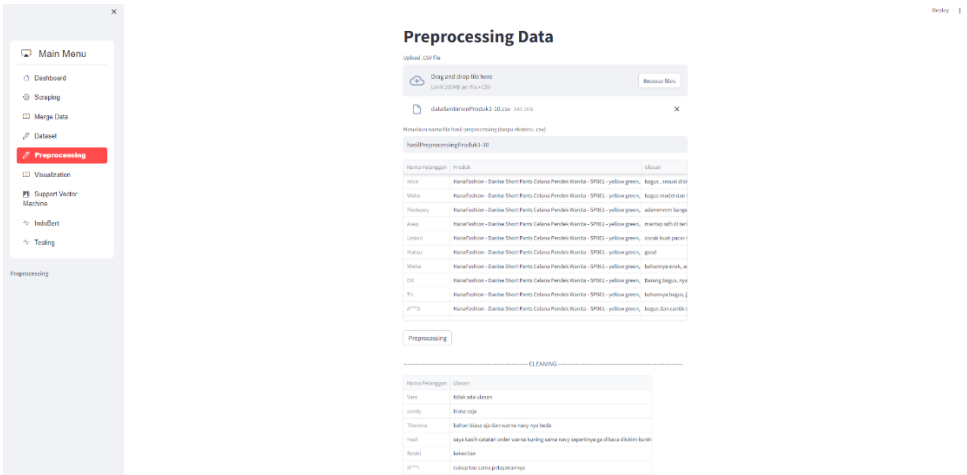
Gambar 4. 8 Tampilan Dataset

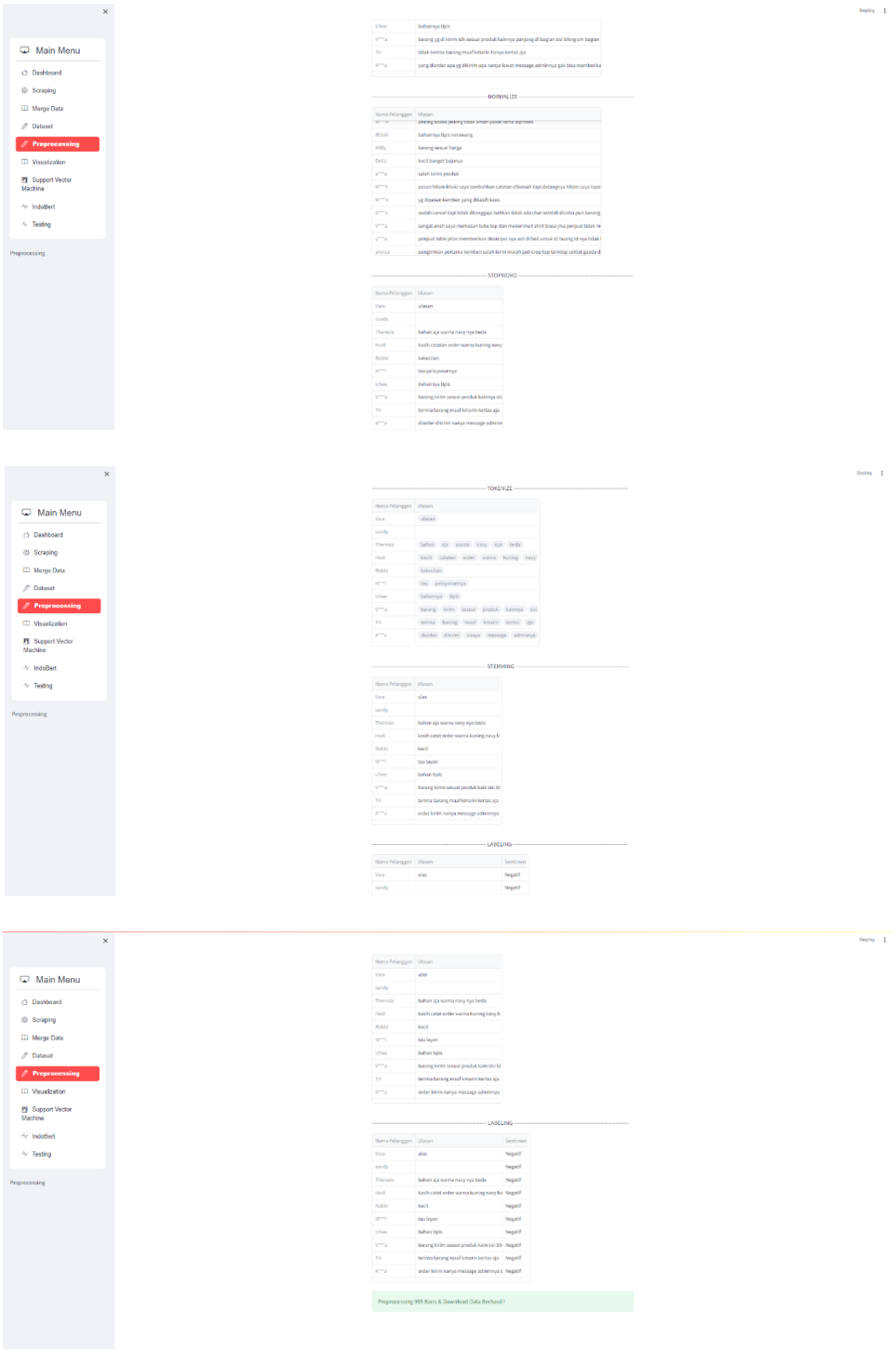
4.5.5 Halaman Preprocessing

Pada tahap implementasi preprocessing, data ulasan toko Hanafashion_shop yang telah berhasil diambil akan melalui serangkaian langkah untuk mempersiapkannya sebelum dilakukan analisis sentimen. Proses preprocessing meliputi langkah-langkah seperti menghapus karakter khusus, mengubah teks menjadi huruf kecil, menghilangkan stopwords (kata-kata umum yang tidak memberikan makna signifikan), serta melakukan tokenisasi untuk memisahkan kata-kata dalam ulasan. Selain itu, langkah-langkah tambahan seperti stemming atau lemmatisasi dapat dilakukan untuk mengubah kata-kata menjadi bentuk dasar mereka. Semua langkah ini bertujuan untuk membersihkan dan mempersiapkan data ulasan agar siap digunakan dalam proses analisis sentimen selanjutnya dengan metode *Support Vector Machine* dan *IndoBert*. Adapun tampilan dan script *preprocessing* dapat dilihat pada gambar di bawah.

```python
elif selected == "Preprocessing":
    st.title("Preprocessing Data")
    uploaded_file = st.file_uploader("Upload .CSV file", type=["csv"])
    file_name_input = st.text_input("Masukkan nama file hasil preprocessing (tanpa ekstensi .csv):")
    if uploaded_file is not None:
        try :
            df = pd.read_csv(uploaded_file, dtype={"Rating":"object"}, index_col=0)
            st.dataframe(df)
        except pd.errors.EmptyDataError:
            st.write("File is empty, please check your input.")
        except pd.errors.ParserError:
            st.write("Invalid data format, please check your input.")

    preprocessing = st.button("Preprocessing")
    if preprocessing:
        with st.spinner('Sedang melakukan preprocessing...'):
            time.sleep(2)
            # st.success("Preprocessing Berhasil & Data Disimpan!")
            df['Ulasan'] = df['Ulasan'].fillna('')
            df['Ulasan'] = df['Ulasan'].apply(preprocessingFunction.clean)
            st.write('')
            st.write(f'---------------------------------------------------------------  CLEANING  --------
-------------------------------------------------')
            st.write(df['Ulasan'])

            df['Ulasan'] = df['Ulasan'].apply(preprocessingFunction.normalisasi)
            st.write('')
            st.write(f'---------------------------------------------------------------  NORMALIZE  -------
-------------------------------------------------')
            st.write(df['Ulasan'])

            df['Ulasan'] = df['Ulasan'].apply(preprocessingFunction.stopword)
            st.write('')
            st.write('---------------------------------------------------------------  STOPWORD  ---------
-------------------------------------------------')
            st.write(df['Ulasan'])

            df['Ulasan'] = df['Ulasan'].apply(preprocessingFunction.tokenisasi)
            st.write('')
            st.write(f'---------------------------------------------------------------  TOKENIZE  --------
-------------------------------------------------')
            st.write(df['Ulasan'])

            # Melakukan stemming pada kolom "Ulasan"
            df['Ulasan'] = df['Ulasan'].apply(preprocessingFunction.stemming)
            st.write('')
            st.write('---------------------------------------------------------------  STEMMING ----------
-------------------------------------------------')
            st.write(df['Ulasan'])

            df['Sentimen'] = df['Rating'].apply(preprocessingFunction.labeling)
            st.write('')
            st.write(f'---------------------------------------------------------------  LABELING  --------
-------------------------------------------------')
            st.write(df[['Ulasan', 'Sentimen']])
```

Gambar 4. 9 Script Preprocessing





Gambar 4. 10 Tampilan Preprocessing

Pada tampilan diatas sistem melakukan preprocessing terhadap 1990 baris data. Tujuan dari preprocessing data adalah untuk mempersiapkan data agar memenuhi syarat sebelum dilakukan training ke model *Support Vector Machine* dan *IndoBert*. Proses dari tahapan preprocessing diantaranya yaitu cleaning, normalize, stopword, tokenize, stemming, dan labelling. Data tersebut beberapa atribut sebelum dilakukan preprocessing, lalu hasil akhir dari

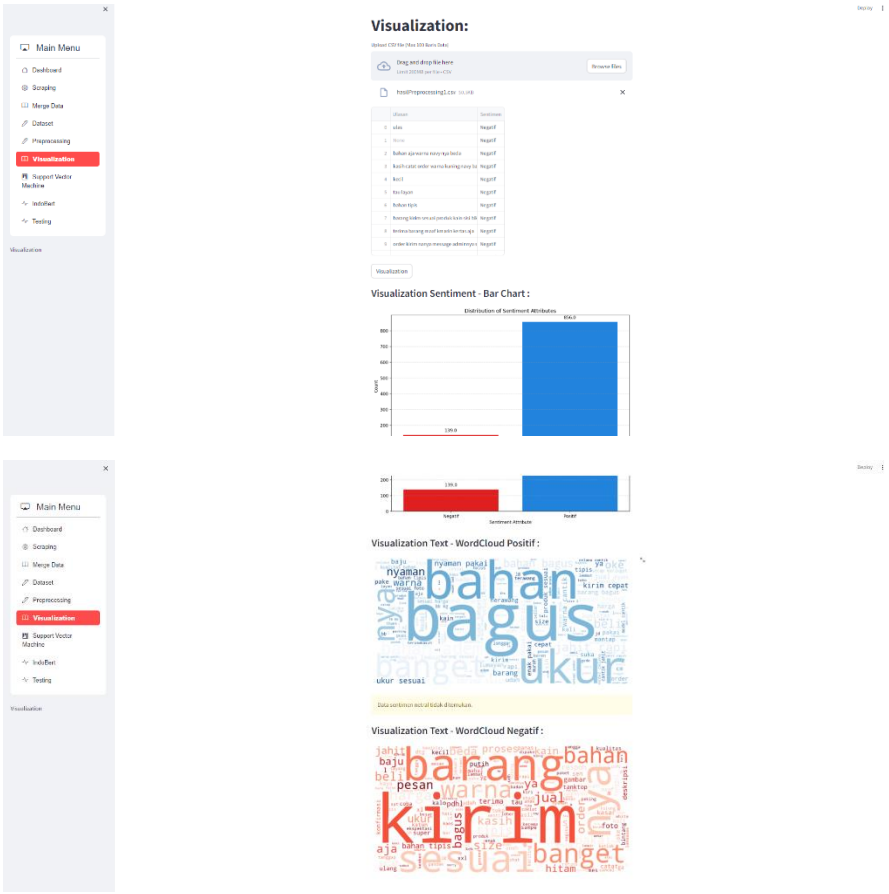
preprocessing hanya mengambil berapa dari kondisi atribut awal yaitu mengambil 996 dari 1113 Ulasan dan Rating, karena peneliti ingin melakukan pengolahan teks pada atribut ulasan sebelum dilakukan pengujian sentiment.

4.5.6 Halaman Visualization

Pada tampilan visualization, fokusnya adalah pada visualisasi sentimen negatif dan positif serta visualisasi teks dari sentimen tersebut menggunakan wordcloud. Visualisasi sentimen negatif dan positif memungkinkan pengguna untuk dengan cepat memahami pola umum dari data sentimen, memungkinkan identifikasi tren dan pola yang mungkin tersembunyi. Sementara visualisasi teks menggunakan wordcloud memberikan representasi visual dari frekuensi kata-kata dalam teks, dengan ukuran kata yang lebih besar menunjukkan frekuensi yang lebih tinggi, memberikan wawasan langsung tentang tema atau topik yang dominan dalam sentimen negatif dan positif. Adapun script dan tampilan visualization dapat dilihat pada gambar di bawah.

```
elif selected == 'Visualization':
    st.title("Visualization:")
    uploaded_file = st.file_uploader("Upload CSV file (Max 100 Baris Data)", type=["csv"])

    if uploaded_file is not None:
        try:
            dfVisualization = pd.read_csv(uploaded_file)

            if 'Ulasan' not in dfVisualization.columns:
                st.warning("Data yang dimasukkan tidak sesuai.")
            else:
                st.dataframe(dfVisualization)

                Visualization = st.button("Visualization")

                if Visualization:
                    with st.spinner('Performing Visualization...'):
                        if 'Sentimen' not in dfVisualization.columns:
                            st.warning("Data yang dimasukkan tidak sesuai.")
                        else:
                            st.subheader("Visualization Sentiment - Bar Chart :")
                            custom_palette = {'Negatif': 'red', 'Positif': '#0384fc'}
                            plt.figure(figsize=(10, 6))

                            ax = sns.countplot(x='Sentimen', data=dfVisualization, order=['Negatif',
'Positif'], palette=custom_palette)
                            ax.grid(axis='y', linestyle='--', alpha=0.5)
                            plt.title('Distribution of Sentiment Attributes')
                            plt.xlabel('Sentiment Attribute')
                            plt.ylabel('Count')
                            for p in ax.patches:
                                ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2.,
p.get_height()),
                                            ha='center', va='center', xytext=(0, 10), textcoords='offset
points')
                            st.pyplot(plt)
```

Gambar 4. 11 Script Visualization



Gambar 4. 12 Tampilan Visualization

Pada hasil visualisasi diatas, user dapat dengan cepat mengidentifikasi sejumlah insight penting. Pertama, pola umum dari sentimen negatif dan positif dapat terlihat secara jelas, memberikan pemahaman mendalam tentang perasaan umum terhadap suatu topik atau produk yang ada di toko Hanafashion_shop. Kedua, melalui wordcloud, dapat dilihat kata-kata kunci yang paling sering muncul dalam konteks sentimen negatif dan positif, memungkinkan pengguna untuk fokus pada aspek-aspek yang paling memengaruhi persepsi secara keseluruhan. Dengan demikian, halaman visualisasi memberikan pemahaman yang lebih dalam dan insight yang berguna bagi pengguna untuk mengambil tindakan yang sesuai.

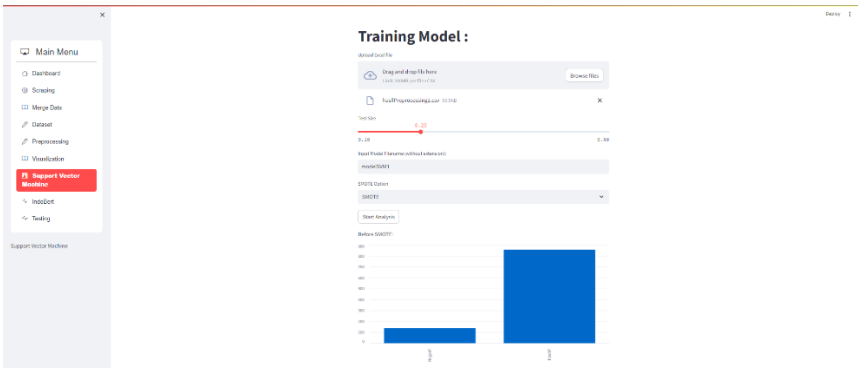
4.5.7 Halaman Support Vector Machine

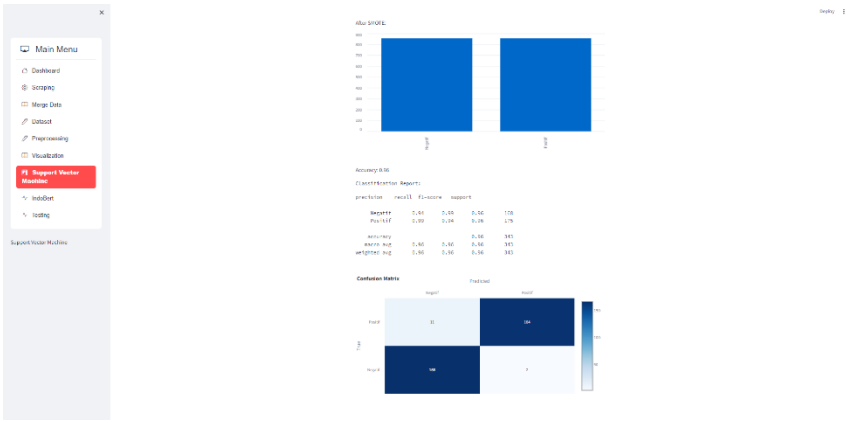
Pada tampilan *support vector machine*, terdapat beberapa komponen penting yang disajikan. Pertama, input data hasil preprocessing menjadi tahap awal yang mempersiapkan data untuk proses pelatihan model. Selanjutnya, terdapat input nama model dan vectorizer hasil dari proses pelatihan data, yang memungkinkan pengguna untuk melakukan proses testing dengan input ulasan dari user. Kemudian, proses splitting data train dan testing memberikan gambaran tentang bagaimana data dibagi menjadi dua subset untuk melatih model dan menguji

performanya. Selain itu, terdapat opsi untuk menggunakan SMOTE (Synthetic Minority Over-sampling Technique) atau tanpa menggunakan SMOTE, yang memungkinkan pengguna untuk memilih apakah ingin menerapkan teknik oversampling pada data atau tidak, tergantung pada kebutuhan spesifik dan karakteristik dari dataset yang digunakan. Dengan demikian, halaman training data menyajikan informasi yang lengkap dan relevan untuk memahami proses pelatihan model SVM beserta opsi yang tersedia untuk meningkatkan kualitasnya.

```
elif selected == 'Support Vector Machine':
    st.title("Training Model :")
    uploaded_file = st.file_uploader("Upload Excel file", type=["csv"])

    if uploaded_file is not None:
        try:
            data = pd.read_csv(uploaded_file)
            model_name = SVC()
            test_size = st.slider("Test Size", min_value=0.1, max_value=0.5, step=0.1, value=0.2)
            model_filename = st.text_input("Input Model Filename (without extension):")
            smote_option = st.selectbox("SMOTE Option", ["SMOTE", "TANPA SMOTE"])

            if model_filename and st.button("Start Analysis"):
                if smote_option == "SMOTE":
                    # Proses SMOTE
                    data_resampled = data.copy()
                    X = data_resampled.drop(columns=['Sentimen'])
                    y = data_resampled['Sentimen']
                    oversample = RandomOverSampler(sampling_strategy='auto')
                    X_resampled, y_resampled = oversample.fit_resample(X, y)
                    data_resampled = pd.concat([X_resampled, y_resampled], axis=1)

                    # Visualisasi jumlah sentimen sebelum SMOTE
                    sentimen_before = data['Sentimen'].value_counts()
                    st.write("Before SMOTE:")
                    st.bar_chart(sentimen_before)

                    # Visualisasi jumlah sentimen sesudah SMOTE
                    sentimen_after = data_resampled['Sentimen'].value_counts()
                    st.write("After SMOTE:")
                    st.bar_chart(sentimen_after)

                accuracy, report, model_filename_with_extension, vectorizer_filename, fig =
svmFunction.analyze_sentiment(data_resampled if smote_option == "SMOTE" else data, model_name, test_size,
model_filename)

                if accuracy is not None and report is not None:
                    st.write(f"Accuracy: {accuracy:.2f}")
                    st.text("Classification Report:")
                    st.text(report)
                    st.plotly_chart(fig)
                    st.success(f"Model saved to {model_filename_with_extension}")
                    st.success(f"Vectorizer saved to {vectorizer_filename}")

        except Exception as e:
            st.warning("Data tidak sesuai. Pastikan file yang diunggah memiliki format yang benar dan
kolom yang diperlukan.")
```

Gambar 4. 13 Script Support Vector Machine



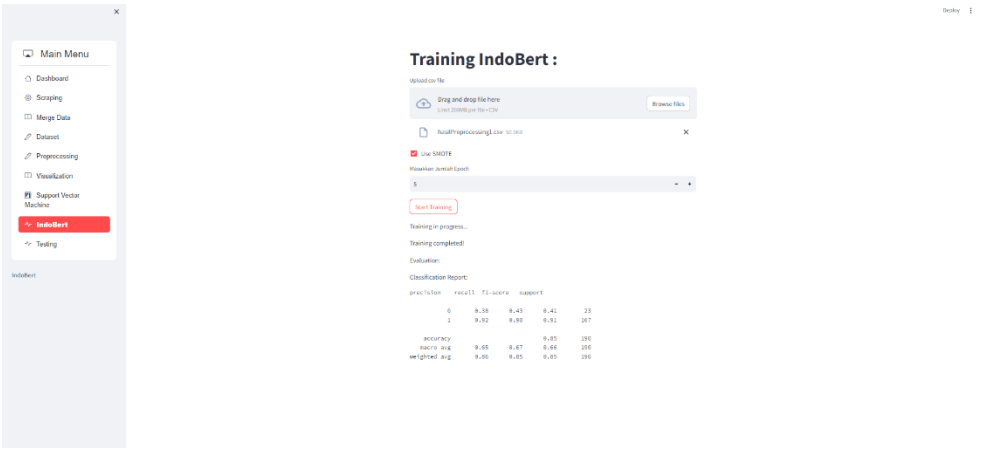


Gambar 4. 14 Tampilan Support Vector Machine

4.5.8 Halaman IndoBert

```
elif selected == 'IndoBert':
    st.title("Training IndoBert :")
    uploaded_file = st.file_uploader("Upload csv file", type=["csv"])

    if uploaded_file is not None:
        df = pd.read_csv(uploaded_file)
        use_smote = st.checkbox("Use SMOTE")
        df = indoBertFunction.preprocess_data(df, use_smote)

        model_name = 'indobenchmark/indobert-base-p1'
        tokenizer = indoBertFunction.BertTokenizer.from_pretrained(model_name)
        model = indoBertFunction.TFBertForSequenceClassification.from_pretrained(model_name)

        reviews = df['Ulasan'].tolist()
        labels = df['Sentimen'].tolist()

        max_length = 128
        input_ids, attention_masks, labels = indoBertFunction.tokenize_data(reviews, labels, tokenizer,
max_length)

        train_indices, test_indices = train_test_split(range(len(input_ids)), test_size=0.2,
random_state=42)
        train_data = (tf.gather(input_ids, train_indices), tf.gather(attention_masks, train_indices),
tf.gather(labels, train_indices))
        test_data = (tf.gather(input_ids, test_indices), tf.gather(attention_masks, test_indices),
tf.gather(labels, test_indices))

        optimizer = tf.keras.optimizers.Adam(learning_rate=2e-5)
        loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)
        metric = tf.keras.metrics.SparseCategoricalAccuracy('accuracy')

        epochs = st.number_input("Masukkan Jumlah Epoch", min_value=1, max_value=20, value=10, step=1)
        batch_size = 16

        if st.button("Start Training"):
            st.write("Training in progress...")
            history = indoBertFunction.train_model(model, train_data, test_data, optimizer, loss, metric,
epochs, batch_size)
            st.write("Training completed!")

            st.write("Evaluation:")
            model.evaluate([test_data[0], test_data[1]], test_data[2])

            test_predictions = model.predict([test_data[0], test_data[1]])
            predicted_labels = tf.argmax(test_predictions.logits, axis=1)

            st.write("Classification Report:")
            st.text(classification_report(test_data[2], predicted_labels))
```

Gambar 4. 15 Script IndoBert

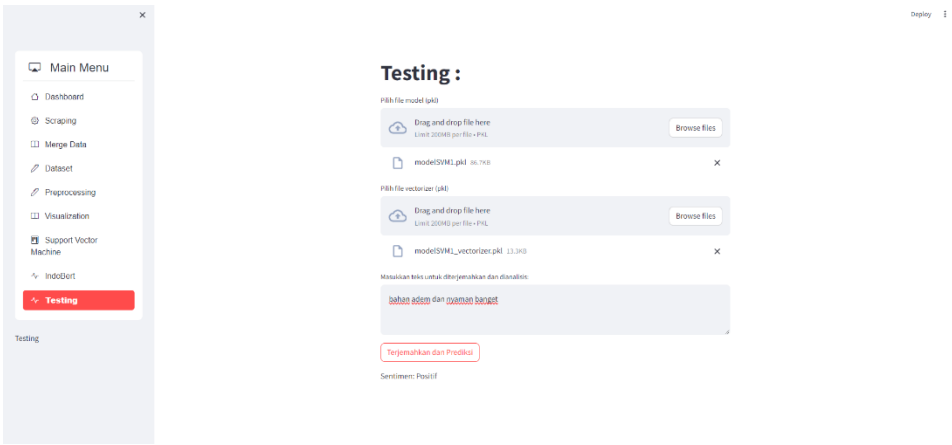


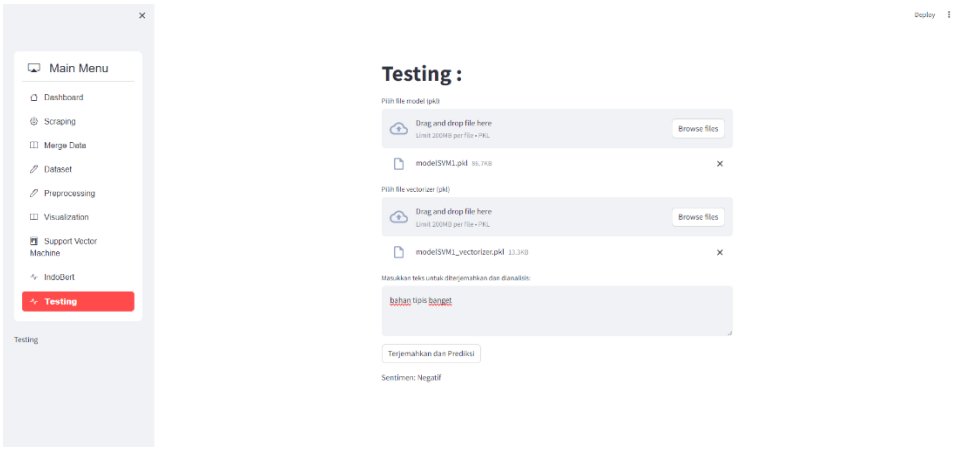
Gambar 4. 16 Tampilan IndoBert

4.5.9 Halaman Testing

```python
elif selected == 'Testing':
    st.title("Testing :")
    model_file = st.file_uploader('Pilih file model (pkl)', type=['pkl'])
    vectorizer_file = st.file_uploader('Pilih file vectorizer (pkl)', type=['pkl'])

    if model_file and vectorizer_file:
        # Load model dan vectorizer dari file yang diunggah
        model = joblib.load(model_file)
        vectorizer = joblib.load(vectorizer_file)

        # Input teks dari pengguna
        user_input = st.text_area('Masukkan teks untuk diterjemahkan dan dianalisis:')

        # Jika tombol ditekan untuk menganalisis
        if st.button('Terjemahkan dan Prediksi'):
            if user_input:
                # Panggil fungsi predict_sentiment dengan model, vectorizer, dan teks sebagai argumen
                sentiment = svmFunction.predict_sentiment(model, vectorizer, user_input)

                # Tampilkan hasil prediksi sentimen
                st.write('Sentimen:', sentiment)
            else:
                st.warning('Masukkan teks untuk menganalisis.')
    else:
        st.warning('Pilih file model dan vectorizer sebelum melakukan testing.')
```

Gambar 4. 17 Script Testing





Gambar 4. 18 Tampilan Testing