Nama : Sarwahita Dwi Prasanti
NIM : A11.2022.13987
Kelp : A11.4701

Sentiment Analysis

- Permasalahan
  Dalam era digital, ulasan pelanggan terhadap produk atau layanan di platform online meningkat pesat. Ulasan ini memberikan wawasan penting mengenai pengalaman pengguna. Namun, karena jumlahnya yang sangat besar, manual reading menjadi tidak efisien, sehingga diperlukan pendekatan otomatis untuk:
  - Analisis Sentimen: Mengidentifikasi apakah ulasan tersebut bernada positif atau negatif.
  - Analisis Topik: Mengetahui tema utama yang sering dibahas dalam ulasan.

- Tujuan
  - Model Sentimen: Membuat model berbasis Naive Bayes untuk klasifikasi ulasan sebagai positif atau negatif.
  - Pemodelan Topik: Menggunakan Latent Dirichlet Allocation (LDA) untuk menemukan topik-topik dominan yang sering muncul.

- Alur Kerja
  

  Dimulai dengan memuat tiga dataset: train.csv, test.csv, dan sampled_train.csv. Dataset diberi nama kolom untuk mempermudah manipulasi data, kemudian kolom title dan text digabung menjadi satu kolom reviewText untuk analisis. Setelah itu, dilakukan pembersihan data seperti menghapus duplikat, memilih kolom yang relevan (polarity dan reviewText), mapping label sentimen (1 menjadi 0 untuk negatif dan 2 menjadi 1 untuk positif), serta menghapus baris kosong. Pada tahap preprocessing teks, dilakukan penghapusan tanda baca dan angka, konversi ke huruf kecil, penghapusan kata-kata umum (stopwords), serta tokenisasi untuk memisahkan teks menjadi daftar kata. Selanjutnya, teks diproses menjadi representasi numerik menggunakan TF-IDF Vectorization, yang diikuti dengan pelatihan model Naive Bayes menggunakan data sampled_train.csv, lalu diperbarui dengan data train.csv. Model kemudian dievaluasi menggunakan metrik seperti akurasi, confusion matrix, dan classification report pada data uji. Akhirnya, Latent Dirichlet Allocation (LDA) digunakan untuk analisis topik, yang menghasilkan topik dominan dari teks yang dianalisis. Hasil model dan vektorizer disimpan untuk digunakan lebih lanjut.

- Dataset
  Dataset ini berasal dari kaggle
  https://www.kaggle.com/datasets/kritanjalijain/amazon-reviews
  berisi ulasan produk dari Amazon, yang digunakan untuk analisis sentimen dan pemodelan topik. Dataset yang terdiri dari tiga bagian utama, yaitu train dataset, test dataset, dan sampled train dataset. Train dataset digunakan untuk melatih model klasifikasi, test dataset untuk evaluasi model, dan sampled train dataset untuk pelatihan awal model sebelum diperbarui dengan dataset utama. Setiap dataset memiliki kolom 'polarity', 'title', dan 'text', yang kemudian digabungkan menjadi satu kolom bernama 'reviewText' untuk analisis teks lebih lanjut.

[2]:
```python
# Step 1: Load Data
train_df = pd.read_csv("train.csv", header=None)
test_df = pd.read_csv("test.csv", header=None)
sample_df = pd.read_csv("sampled_train.csv", header=None)
```

[3]:
```python
# Add headers
train_df.columns = ['polarity', 'title', 'text']
test_df.columns = ['polarity', 'title', 'text']
sample_df.columns = ['polarity', 'title', 'text']
```

[4]:
```python
# Step 2: Combine columns for analysis
train_df['reviewText'] = train_df['title'] + " " + train_df['text']
test_df['reviewText'] = test_df['title'] + " " + test_df['text']
sample_df['reviewText'] = sample_df['title'] + " " + sample_df['text']
```

- EDA

  Langkah awal dalam eksplorasi data adalah membersihkan dataset. Proses ini meliputi penghapusan duplikasi pada kolom 'reviewText', konversi nilai 'polarity' dari 1 menjadi 0 dan 2 menjadi 1 untuk label biner (0 untuk negatif, 1 untuk positif), serta penghapusan baris yang memiliki nilai kosong pada 'reviewText'. Distribusi data juga dianalisis untuk memastikan keseimbangan antara label positif dan negatif.

```python
# Step 3: Data Cleaning
train_df = train_df.drop_duplicates(subset=['reviewText'])
test_df = test_df.drop_duplicates(subset=['reviewText'])
sample_df = sample_df.drop_duplicates(subset=['reviewText'])

train_df = train_df[['polarity', 'reviewText']]
test_df = test_df[['polarity', 'reviewText']]
sample_df = sample_df[['polarity', 'reviewText']]
```

```python
# Map polarity labels
label_map = {1: 0, 2: 1}
train_df['polarity'] = train_df['polarity'].map(label_map)
test_df['polarity'] = test_df['polarity'].map(label_map)
sample_df['polarity'] = sample_df['polarity'].map(label_map)

train_df = train_df.dropna(subset=['reviewText'])
test_df = test_df.dropna(subset=['reviewText'])
sample_df = sample_df.dropna(subset=['reviewText'])
```

Berikutnya adalah preprocessing teks. Setiap teks dalam kolom 'reviewText' dibersihkan dari angka dan tanda baca, diubah menjadi huruf kecil, lalu di-tokenisasi untuk memisahkan kata-kata. Stopwords yang tidak relevan dihapus menggunakan pustaka NLTK. Setelah itu, teks diubah menjadi representasi numerik menggunakan TF-IDF Vectorizer, yang memilih maksimal 1000 fitur teratas berdasarkan skor TF-IDF.

[7]:
```
# Step 4: Text Preprocessing
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    text = re.sub(r"[^a-zA-Z\s]", "", text)  # Remove numbers and punctuation
    text = text.lower()  # Lowercase
    text = text.split()  # Tokenization
    text = [word for word in text if word not in stop_words]  # Remove stopwords
    return " ".join(text)

train_df['reviewText'] = train_df['reviewText'].apply(preprocess_text)
test_df['reviewText'] = test_df['reviewText'].apply(preprocess_text)
sample_df['reviewText'] = sample_df['reviewText'].apply(preprocess_text)
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\sarwa\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

[8]:
```
# Step 5: TF-IDF Vectorization
tfidf = TfidfVectorizer(max_features=1000, stop_words='english')
X_sample_train = tfidf.fit_transform(sample_df['reviewText'])
X_train = tfidf.transform(train_df['reviewText'])
X_test = tfidf.transform(test_df['reviewText'])

# Extract target labels
y_sample_train = sample_df['polarity'].values
y_train = train_df['polarity'].values
y_test = test_df['polarity'].values
```

- Proses Learning/Modeling & Performa Model
  Model klasifikasi yang digunakan adalah Naive Bayes Multinomial, yang dilatih menggunakan sampled train dataset dan diperbarui dengan train dataset utama menggunakan metode partial_fit. Proses evaluasi dilakukan dengan menghitung akurasi, membuat confusion matrix, dan menghasilkan classification report. Model menunjukkan performa yang cukup baik dengan akurasi sebesar 82.66%. Confusion matrix menunjukkan bahwa model berhasil memprediksi 163,081 ulasan negatif dengan benar sebagai negatif (True Negatives) dan 167,577 ulasan positif dengan benar sebagai positif (True Positives). Namun, terdapat kesalahan prediksi sebanyak 36,903 ulasan negatif yang salah diprediksi sebagai positif (False Positives) dan 32,415 ulasan positif yang salah diprediksi sebagai negatif (False Negatives). Hasil ini divisualisasikan dalam bentuk heatmap untuk memperjelas distribusi prediksi.

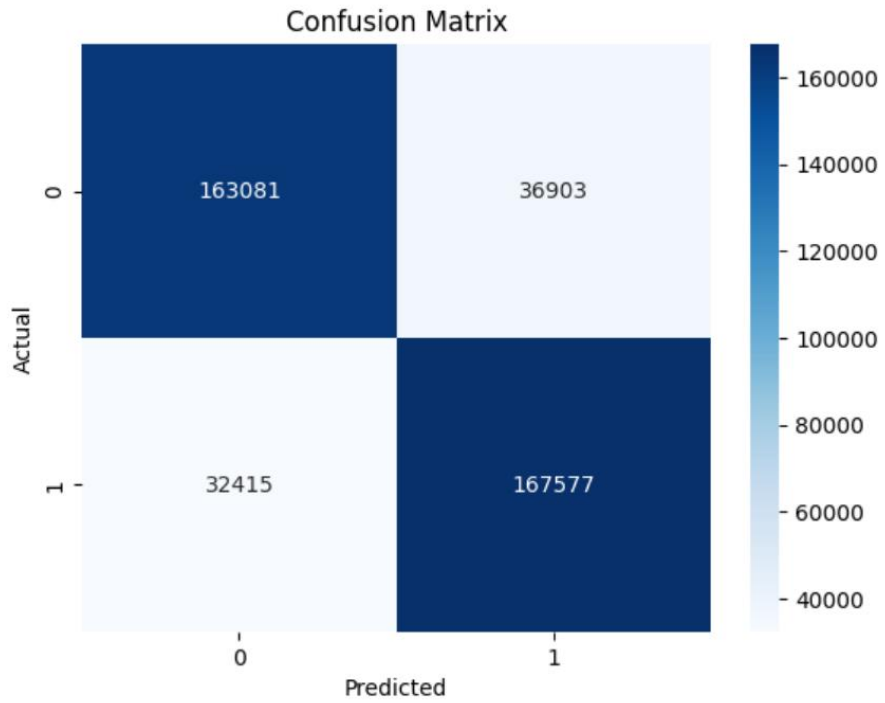
```
[9]: # Step 6: Train and Update Model
model = MultinomialNB()
model.fit(X_sample_train, y_sample_train)
model.partial_fit(X_train, y_train, classes=np.unique(y_train))
```

```
[9]: MultinomialNB
MultinomialNB()
```

```
[10]: # Step 7: Evaluate Model
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.8266946016761005
Confusion Matrix:
 [[163081  36903]
 [ 32415 167577]]
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.82      0.82    199984
           1       0.82      0.84      0.83    199992

    accuracy                           0.83    399976
   macro avg       0.83      0.83      0.83    399976
weighted avg       0.83      0.83      0.83    399976
```

```
[11]: # Step 8: Visualize Confusion Matrix
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt="d", cmap="Blues")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

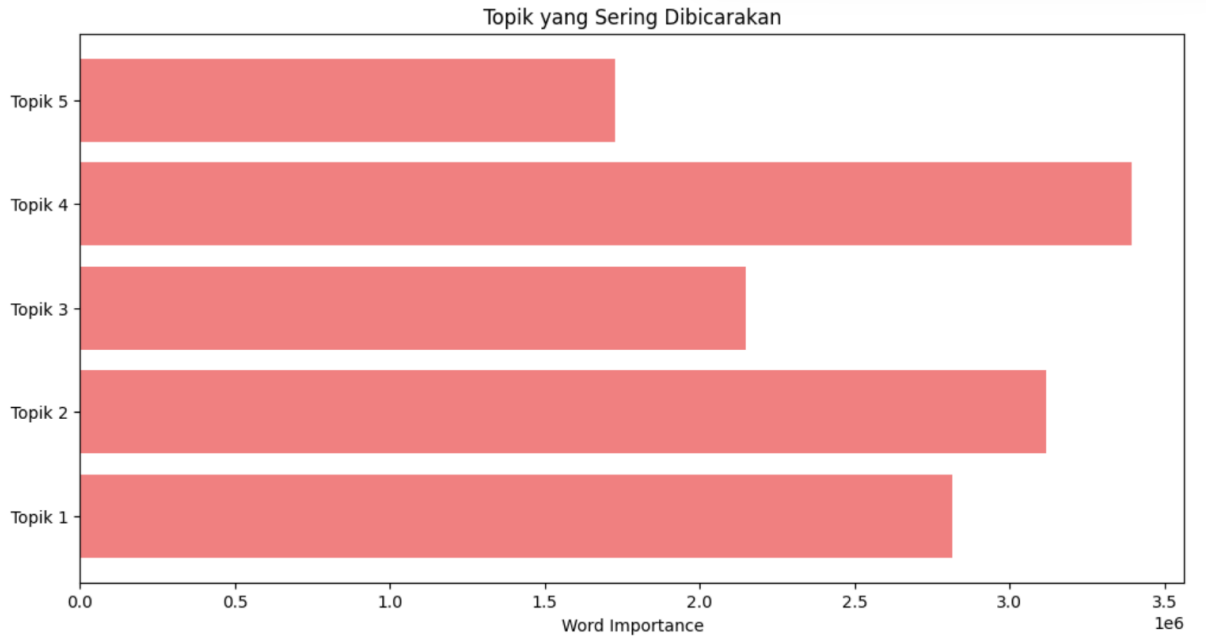


proyek ini juga mencakup analisis tema menggunakan Latent Dirichlet Allocation (LDA) untuk menemukan pola atau tema utama dalam data teks. Model LDA dilatih pada representasi TF-IDF, dan hasilnya digunakan untuk menampilkan kata-kata kunci pada setiap topik. Setiap topik diidentifikasi dengan kata-kata yang paling signifikan berdasarkan bobot kontribusinya.

[12]:
```
# Step 9: Topic Modeling with LDA
lda = LatentDirichletAllocation(n_components=5, random_state=42)
lda.fit(X_train)

# Display topics
def display_topics(model, feature_names, no_top_words):
    topics = []
    for topic_idx, topic in enumerate(model.components_):
        topic_words = " | ".join([feature_names[i] for i in topic.argsort()[:-no_top_words - 1:-1]])
        print(f"Topik {topic_idx + 1}: {topic_words}")
        topics.append(f"Topik {topic_idx + 1}: {topic_words}")
    return topics

topics = display_topics(lda, tfidf.get_feature_names_out(), 10)
```

```
Topik 1: book | read | books | good | great | author | information | reading | written | life
Topik 2: movie | book | game | story | like | good | read | film | great | characters
Topik 3: cd | album | music | songs | like | song | great | best | good | sound
Topik 4: product | use | great | work | good | works | dont | bought | buy | money
Topik 5: product | great | amazon | item | price | received | ordered | bought | gift | good
```

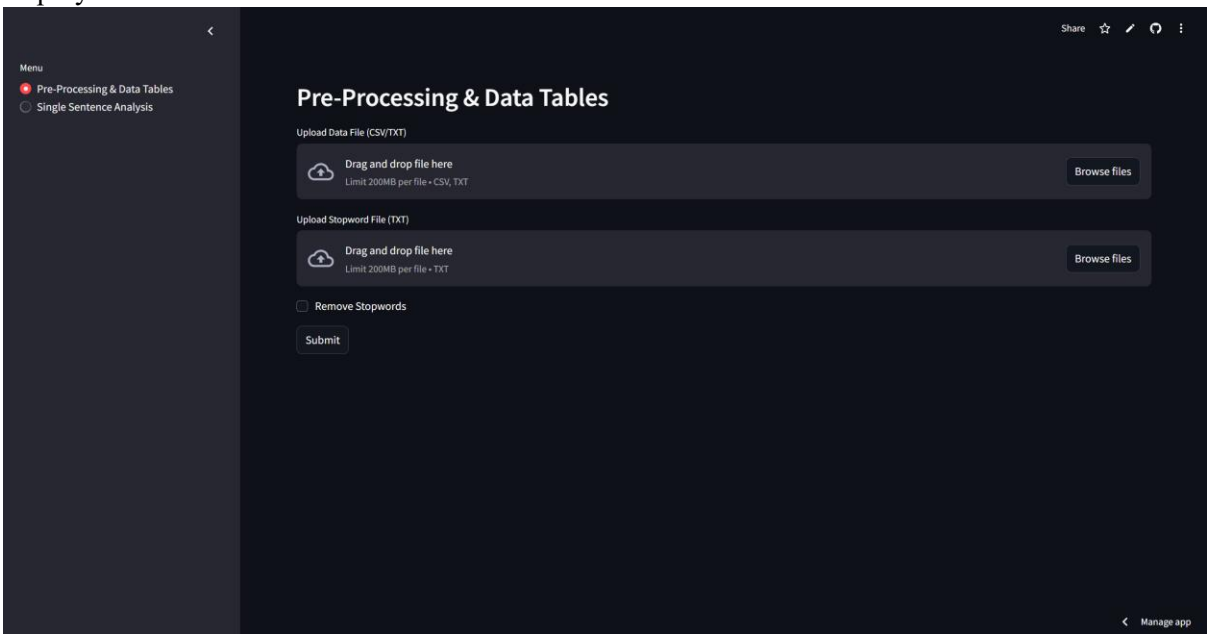
[13]:
```
# Visualize topics
def visualize_topics(model, feature_names, no_top_words):
    topic_words = []
    for topic in model.components_:
        words = [feature_names[i] for i in topic.argsort()[:-no_top_words - 1:-1]]
        topic_words.append(" | ".join(words))

    plt.figure(figsize=(12, 6))
    plt.barh(range(len(topic_words)), model.components_.sum(axis=1), color='lightcoral')
    plt.yticks(range(len(topic_words)), [f"Topik {i+1}" for i in range(len(topic_words))])
    plt.xlabel("Word Importance")
    plt.title("Topik yang Sering Dibicarakan")
    plt.show()

visualize_topics(lda, tfidf.get_feature_names_out(), 10)
```

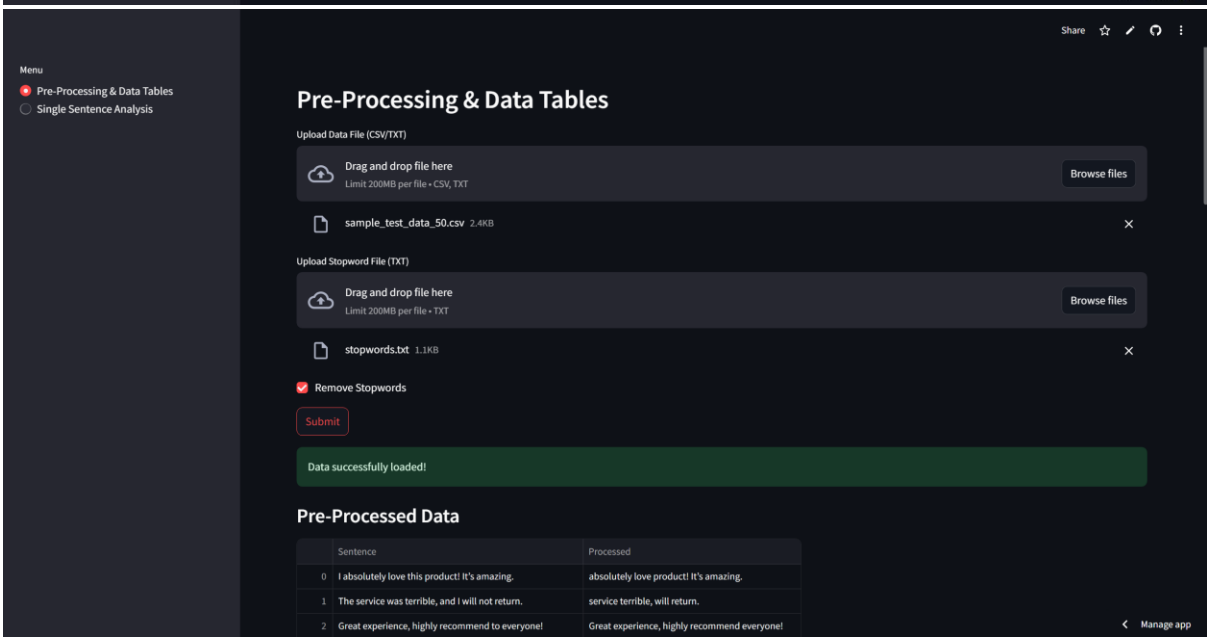


Model Naive Bayes dan TF-IDF Vectorizer disimpan ke dalam file model.pkl dan vectorizer.pkl

- Diskusi Hasil dan kesimpulan
  Berdasarkan hasil evaluasi, model menunjukkan performa yang cukup baik dengan tingkat akurasi mencapai 82,66%. Model berhasil memprediksi sebagian besar ulasan dengan benar, meskipun masih terdapat beberapa kesalahan prediksi yang tercermin pada nilai False Positives dan False Negatives. Kesalahan ini dapat diminimalkan dengan melakukan optimasi lebih lanjut, khususnya pada tahap preprocessing data dan pemilihan fitur yang lebih relevan. Selain itu, analisis topik menggunakan metode LDA berhasil mengidentifikasi lima tema utama dalam ulasan, yaitu buku, musik, produk, dan belanja, yang dapat memberikan wawasan mendalam untuk analisis lebih lanjut.

- Deploy





Menu

- Pre-Processing & Data Tables
- Single Sentence Analysis

Share

Pre-Processed Data

	Sentence	Processed
0	I absolutely love this product! It's amazing.	absolutely love product! It's amazing.
1	The service was terrible, and I will not return.	service terrible, will return.
2	Great experience, highly recommend to everyone!	Great experience, highly recommend everyone!
3	Not worth the money, very disappointing.	worth money, disappointing.
4	The quality exceeded my expectations, fantastic purchase.	quality exceeded expectations, fantastic purchase.
5	Worst experience ever, I regret buying this.	Worst experience ever, regret buying this.
6	Decent product, but the price is too high.	Decent product, price high.
7	The customer service was outstanding and very helpful.	customer service outstanding helpful.
8	I wouldn't recommend this to my friends.	wouldn't recommend friends.
9	Excellent value for money, very satisfied!	Excellent value money, satisfied!

Data Tables With Full Features

	Sentence	Processed	Positive	Negative	Analysis
0	I absolutely love this product! It's amazing.	absolutely love product! It's amazing.	0.8202	0.1798	Positive
1	The service was terrible, and I will not return.	service terrible, will return.	0.0549	0.9451	Negative
2	Great experience, highly recommend to everyone!	Great experience, highly recommend everyone!	0.8606	0.1394	Positive
3	Not worth the money, very disappointing.	worth money, disappointing.	0.0531	0.9469	Negative
4	The quality exceeded my expectations, fantastic purchase.	quality exceeded expectations, fantastic purchase.	0.6796	0.3204	Positive
5	Worst experience ever, I regret buying this.	Worst experience ever, regret buying this.	0.135	0.865	Negative
6	Decent product, but the price is too high.	Decent product, price high.	0.4664	0.5336	Negative
7	The customer service was outstanding and very helpful.	customer service outstanding helpful.	0.3186	0.6814	Negative

Manage app

Menu

- Pre-Processing & Data Tables
- Single Sentence Analysis

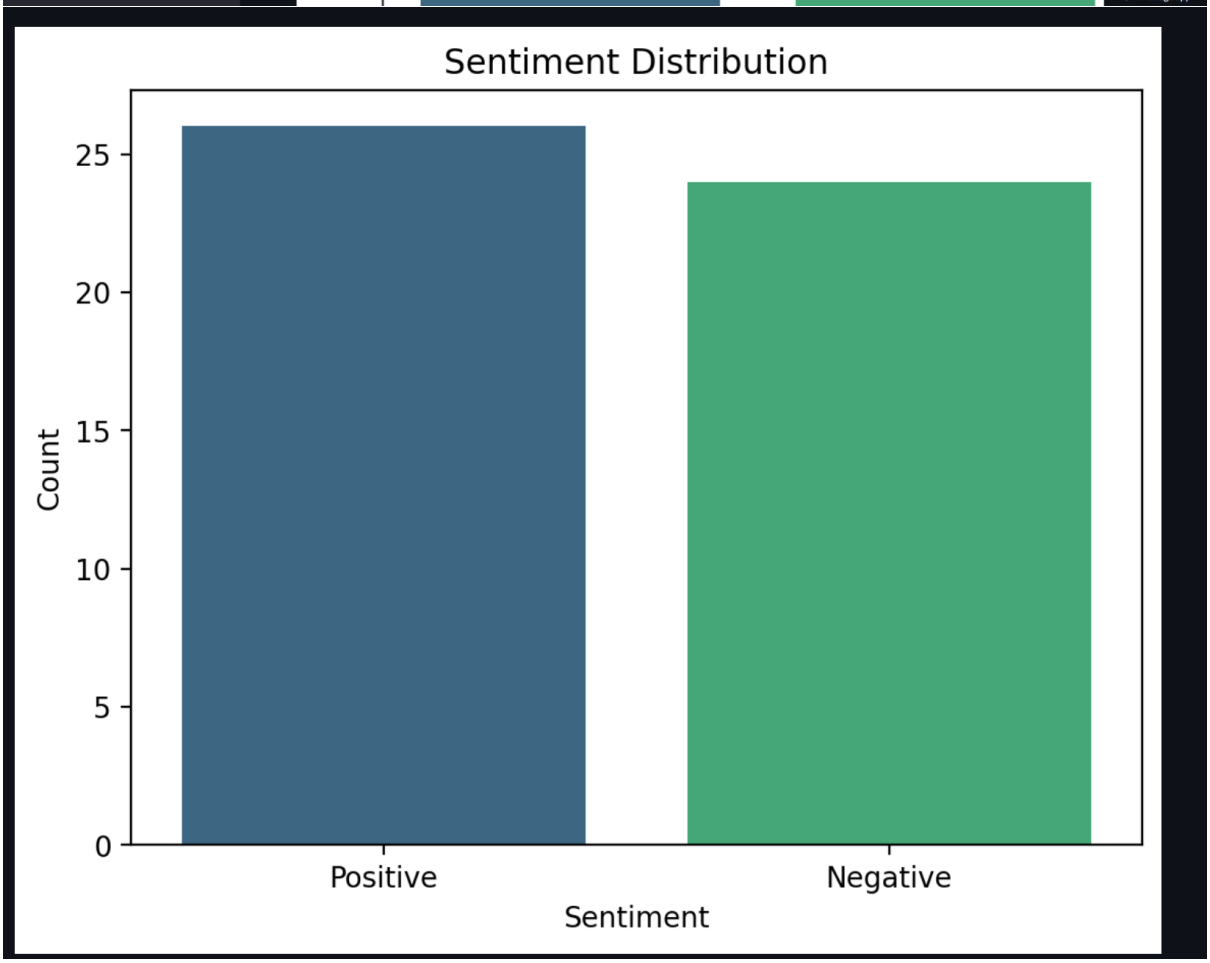
Share

Data Tables With Full Features

	Sentence	Processed	Positive	Negative	Analysis
0	I absolutely love this product! It's amazing.	absolutely love product! It's amazing.	0.8202	0.1798	Positive
1	The service was terrible, and I will not return.	service terrible, will return.	0.0549	0.9451	Negative
2	Great experience, highly recommend to everyone!	Great experience, highly recommend everyone!	0.8606	0.1394	Positive
3	Not worth the money, very disappointing.	worth money, disappointing.	0.0531	0.9469	Negative
4	The quality exceeded my expectations, fantastic purchase.	quality exceeded expectations, fantastic purchase.	0.6796	0.3204	Positive
5	Worst experience ever, I regret buying this.	Worst experience ever, regret buying this.	0.135	0.865	Negative
6	Decent product, but the price is too high.	Decent product, price high.	0.4664	0.5336	Negative
7	The customer service was outstanding and very helpful.	customer service outstanding helpful.	0.3186	0.6814	Negative
8	I wouldn't recommend this to my friends.	wouldn't recommend friends.	0.737	0.263	Positive
9	Excellent value for money, very satisfied!	Excellent value money, satisfied!	0.7802	0.2198	Positive

Sentiment Analysis Visualization

Manage app



Berikut ada link deploy : https://analysis-statment.streamlit.app/
Link repository Github : https://github.com/Tataaa2/analisis-statment