SISTEM IDENTIFIKASI SENTIMEN MENGGUNAKAN METODE NAIVE BAYES
FINAL PROJECT PENGANTAR PEMROSESAN DATA MULTIMEDIA

OLEH :

KELOMPOK 5 (KELAS B)

I Wayan Wikananda Adikara	**2108561027**
Monika Hermiani Yolanda Simamora	**2108561051**
I Made Ryan Prana Dhita	**2108561107**

PROGRAM STUDI INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS UDAYANA

BALI

2023

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada era modern ini, sentimen atau opini masyarakat semakin bertambah luas dan bebas diungkapkan di berbagai media. Sentimen analisis adalah suatu metode yang digunakan untuk mengidentifikasi, memahami dan mengklasifikasikan sentimen atau pendapat di dalam teks, seperti komentar media sosial, ulasan pelanggan, dan lain-lain. Tujuan dari sentimen analisis adalah untuk menentukan apakah sentimen yang terkandung dalam teks tersebut positif atau negatif. Sentimen dapat menjadi umpan balik bagi perusahaan yang ingin mengetahui respon dari masyarakat terhadap produk dagangan mereka. Oleh karena itu, sangat penting untuk mengembangkan sistem yang dapat secara otomatis mengidentifikasi sentimen dari data teks tersebut.

Tujuan dari laporan ini adalah membangun sistem aplikasi untuk mengidentifikasi sentimen atau emosi dari beberapa ulasan teks. Terdapat dua sentimen yaitu sentimen positif dan sentimen negatif. Adapun tahapan yang akan dilakukan pada sistem ini yaitu tahap preprocessing data untuk menghilangkan noise atau gangguan pada dataset sehingga data lebih siap diolah untuk proses selanjutnya. Tahap selanjutnya yaitu ekstraksi fitur dengan Term-Frequency (TF) yang mengubah data berupa teks menjadi representasi numerik agar lebih sudah diproses pada mesin. Tahap selanjutnya yaitu seleksi fitur dengan Chi Square yang berfungsi mengurangi penggunaan fitur yang tidak relevan atau tidak memberikan informasi penting dalam proses analisis sentimen. Tahap berikutnya yaitu membangun model klasifikasi menggunakan metode Multinomial Naive Bayes yang akan dibahas lebih lanjut pada bab berikutnya.

1.2 Tujuan

Adapun tujuan dari laporan ini sebagai berikut :

1. Untuk membangun sistem aplikasi yang dapat mengidentifikasi sentimen dari beberapa ulasan teks.
2. Untuk mengetahui tahapan *text preprocessing*, tahapan ekstraksi fitur, dan tahapan seleksi fitur pada data teks.
3. Untuk mengetahui tahapan *training* pada model klasifikasi sentimen data teks.
4. Untuk mengevaluasi kinerja model yang dibangun menggunakan ukuran evaluasi akurasi, precision, recall, dan F1-Score.

1.3 Manfaat

Adapun manfaat dari laporan ini sebagai berikut :

1. Meningkatkan pemahaman dalam pengimplementasian sistem pengidentifikasian sentimen atau emosi dari beberapa ulasan.
2. Meningkatkan pemahaman tentang algoritma klasifikasi Multinomial Naive Bayes dalam identifikasi sentimen data teks.

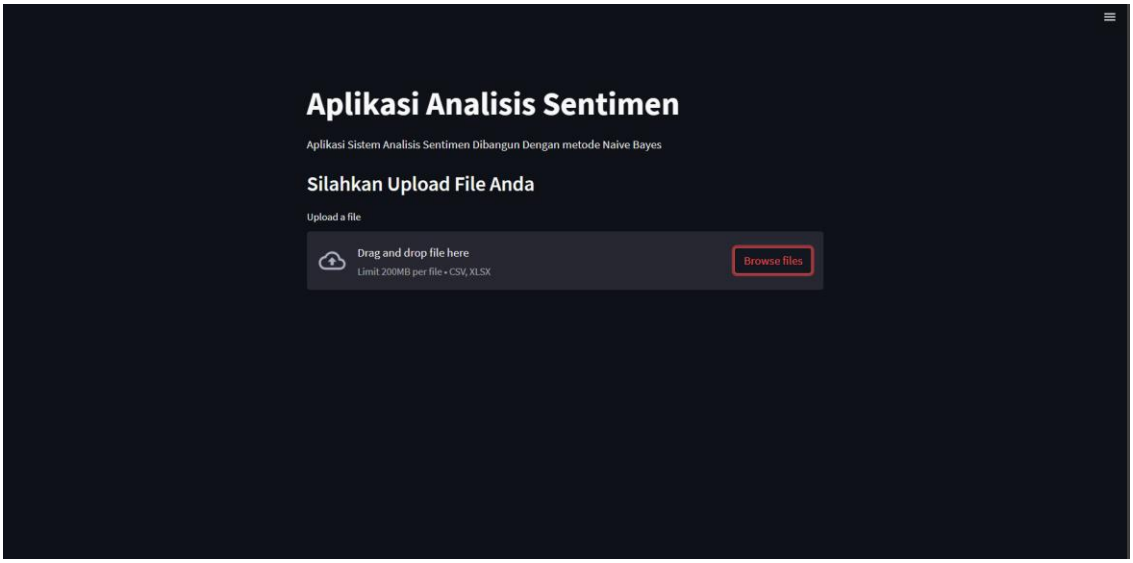
BAB II

ISI

2.1 Manual Aplikasi

Antar Muka Sistem dan Fitur Sistem

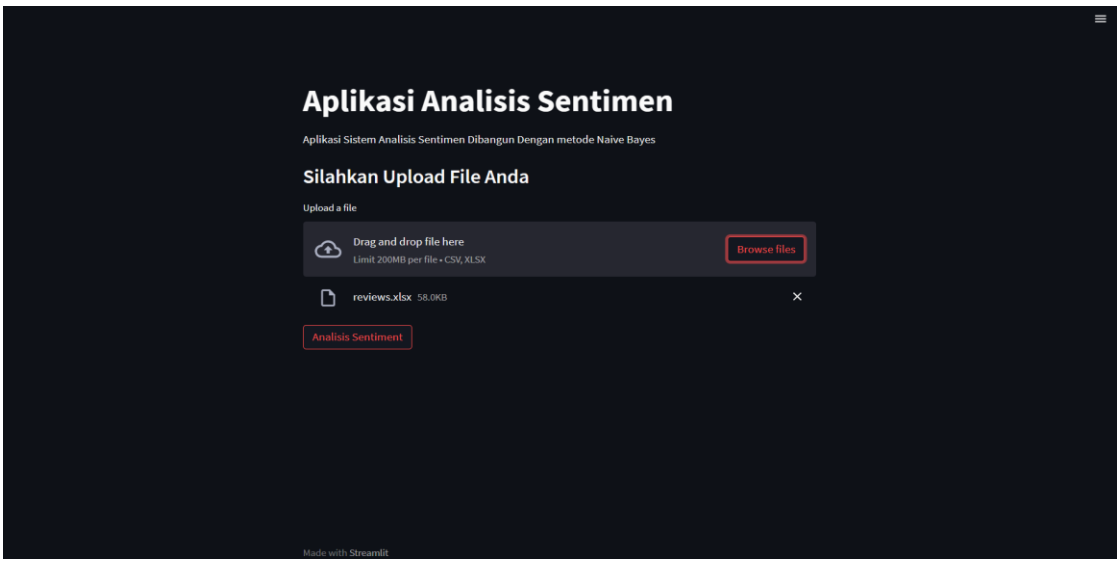
- **Step 1 :** Setelah aplikasi dibuka akan terlihat tampilan seperti gambar di bawah ini.



Tampilan diatas adalah tampilan awal ketika membuka aplikasi. Fitur yang ada dalam aplikasi ini ada fitur menginput data. Data yang dapat diinput dalam sistem adalah data dengan format csv dan xlsx.

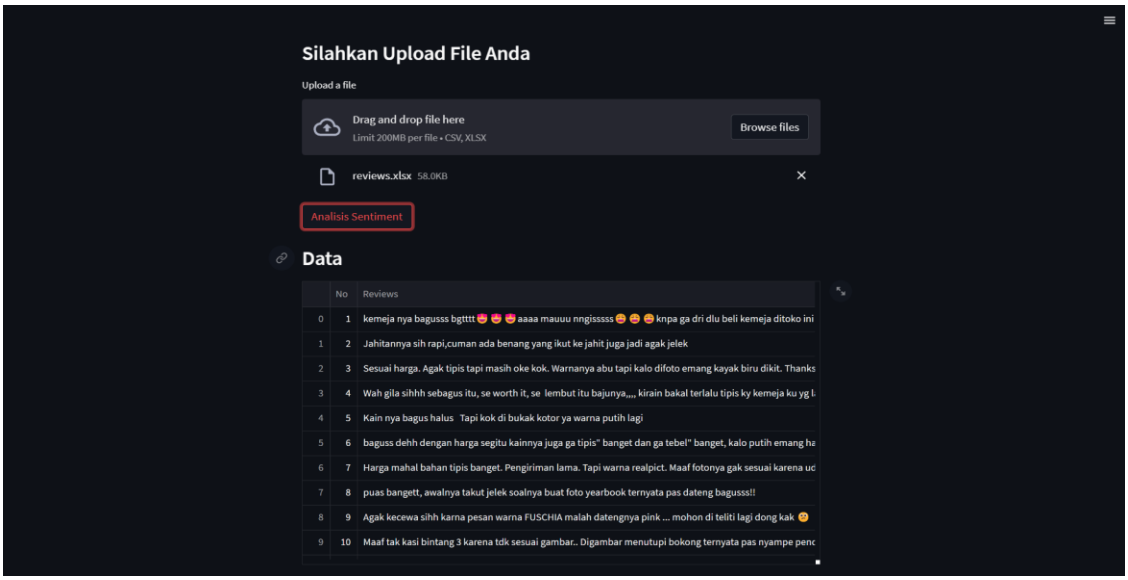
Silahkan pilih file sentimen yang ingin dianalisis dengan menekan tombol browse file lalu pilih file dengan format csv, xlsx.

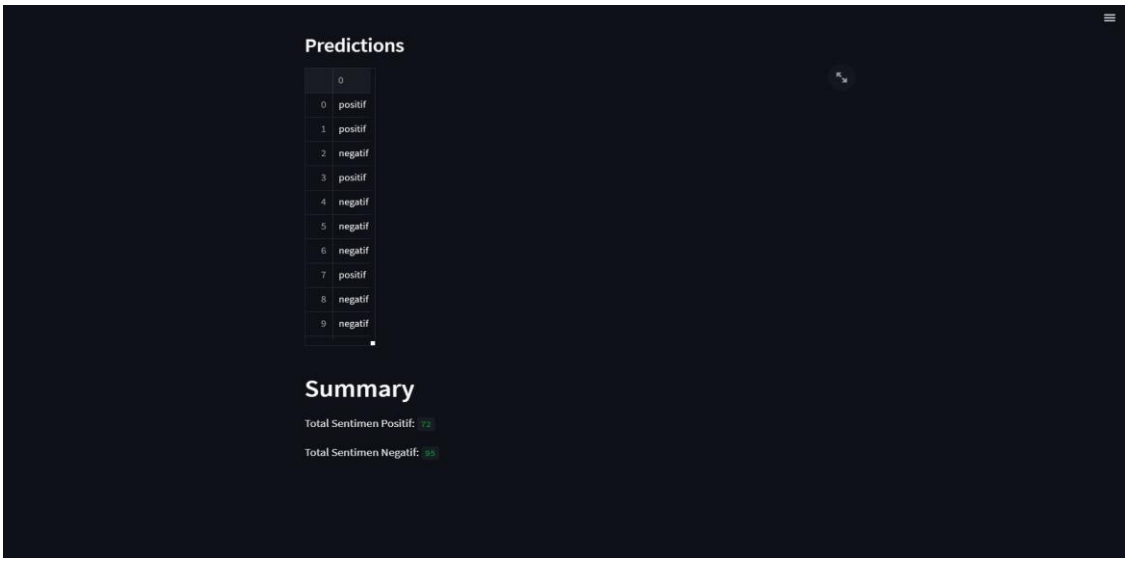
- **Step 2 :** Setelah file terpilih maka akan muncul tampilan seperti ini.



Jika ingin mengganti file yang akan diolah maka silahkan tekan [x] pada sebelah kanan input file. Jika tidak silahkan tekan tombol analisis sentimen, silahkan tunggu sampai proses analisis selesai.

- **Step 3 :** Ketika proses analisis selesai maka akan muncul tampilan data seperti gambar di bawah ini.





dapat dilihat pada gambar ketika data sudah berhasil diolah maka akan muncul output berupa total data sentimen yang telah dimasukan serta hasil dari prediksi sentimen tersebut apakah sentimen tersebut berupa positif atau sentimen negatif.

2.2 Source Code Modul

Tahap Preprocessing

`# Function to remove unnecessary characters and numbers from sentences` `def remove(sentence):` `  sentence = re.sub(r'[0-9]', ' ', sentence)` `  sentence = re.sub(r'[^\w\s]', ' ', sentence)` `  sentence = re.sub(r'[^A-Za-z\s]', '', sentence)` `  sentence = sentence.lower()` `  sentence = sentence.strip()` `  sentence = re.sub(r'\s+', ' ', sentence)` `  sentence = sentence.replace('\n', '')` `  sentence = sentence.replace('_', '')` `  return sentence`	Pada fungsi remove ini, dilakukan pembersihan data dengan menghapus angka, menghapus karakter khusus kecuali spasi, menghapus karakter non-alfabet, menghapus spasi berlebih, menghapus karakter newline dan underscore. Selain itu, dilakukan pengubahan huruf kapital ke bentuk huruf kecil.
`# Function to tokenize sentences` `def tokenize(sentence):` `  return word_tokenize(sentence)`	Pada fungsi tokenize ini, dilakukan proses memecah input menjadi satuan token atau kata
`# Function to remove stopwords from tokens` `def remove_stopwords(tokens):` `  factory = StopWordRemoverFactory()` `  stopwords_remover = factory.create_stop_word_remover()` `  stopwords_dictionary = [` `  'gak', 'masa', 'bisa', 'lagi', 'banget', 'sama', 'nya', 'saya', 'semua', 'kalo', 'saat', 'sambil', 'ya',` `  'untuk', 'segitu', 'lain', 'sih', 'sangat', 'tidak', 'yang', 'tapi', 'itu', 'aduh', 'lah', 'buat', 'mah',`	Pada fungsi remove stopword, dilakukan proses menghapus token atau kata umum yang tidak memberikan pengaruh dalam proses klasifikasi. Pada fungsi ini stopword list yang digunakan adalah kombinasi stopword yang ada pada library nltk dan stopword yang telah ditambahkan ke stopwords_dictionary.

'tahu', 'apa', 'mau', 'banyak', 'di', 'karena', 'bakal', 'padahal', 'ni', 'orang', 'terus', 'lain', 'sini', 'hanya', 'dengan', 'aja', 'dan', 'ada', 'sekali', 'udh', 'kali', 'walaupun', 'pdhl', 'dari', 'cuma', 'juga', 'sesuai', 'ini', 'jadi', 'tt'] combined_stopwords = set(stopwords.words('indonesian')) | set(stopwords_dictionary) filtered_tokens = [token for token in tokens if token.lower() not in combined_stopwords] return filtered_tokens	
# Function to perform stemming on tokens def stemming(tokens): factory = StemmerFactory() stemmer = factory.create_stemmer() stemmed_tokens = [stemmer.stem(token) for token in tokens] return stemmed_tokens	Pada fungsi stemming, dilakukan proses mengubah token dalam bentuk kata berimbuhan ke dalam bentuk kata dasar.

Tahap Ekstraksi Fitur

# Preprocess the data preprocessed_data = preprocess_data(data) # Convert the preprocessed data to text representation text_data = preprocessed_data['stemming'].apply(lambda tokens: ' '.join(tokens))	Pada proses ini, data yang telah melalui tahap preprocessing disimpan dalam bentuk representasi teks pada text_data. Kemudian dilakukan perhitungan term frequency dengan menggunakan library CountVectorizer pada modul

`` # Create the CountVectorizer object vectorizer = CountVectorizer() # Transform the data into term frequency matrix tf_matrix = vectorizer.fit_transform(text_data) # Get the list of features (unique words) features = vectorizer.get_feature_names_out() # Create a DataFrame from the term frequency matrix df_tf = pd.DataFrame(tf_matrix.toarray(), columns=features) ```	sklearn. Kemudian hasil perhitungan term frequency disimpan dalam data frame.

Tahap Seleksi Fitur

`` # Calculate chi-square scores labels = preprocessed_data['Label'] chi2_scores = chi2(df_tf, labels)[0] # Select the top features based on chi-square scores total_features = df_tf.shape[1] k = int(0.5 * total_features) selector = SelectKBest(chi2, k=k) selected_matrix = selector.fit_transform(df_tf, labels) selected_features_indices = selector.get_support(indices=True) selected_features = [features[i] for i in selected_features_indices] df_selected_features = pd.DataFrame(selected_matrix, ```	Pada proses ini, dilakukan pemilihan fitur dengan seleksi fitur Chi Square. Fitur-fitur yang tidak relevan dalam proses klasifikasi akan dihapus. Proses seleksi fitur dilakukan dengan menggunakan library SelectKBest dan chi2 pada modul sklearn. Nilai k pada program menentukan berapa banyak fitur yang digunakan pada seleksi fitur.

```
columns=selected_features)
    df_selected_features['Label'] = labels
```

Adapun beberapa percobaan yang dilakukan untuk mengetahui penggunaan seleksi fitur yang menghasilkan akurasi tertinggi pada proses klasifikasi yang ditampilkan dalam bentuk sebagai berikut.

Penggunaan Fitur	**Hasil Akurasi**
Menggunakan 10% dari total fitur	Akurasi: 0.9820359281437125 Presisi: 0.9827741776720531 Recall: 0.9820359281437125 F1-Score: 0.9820832833655356
Menggunakan 20% dari total fitur	Akurasi: 0.9880239520958084 Presisi: 0.988356620093147 Recall: 0.9880239520958084 F1-Score: 0.9880459152118297
Menggunakan 30%-70% dari total fitur (menghasilkan nilai akurasi yang sama)	Akurasi: 0.9880239520958084 Presisi: 0.988356620093147 Recall: 0.9880239520958084 F1-Score: 0.9880459152118297
Menggunakan 80% dari total fitur	Akurasi: 0.9820359281437125 Presisi: 0.9827741776720531 Recall: 0.9820359281437125 F1-Score: 0.9820832833655356
Menggunakan 90% dari total fitur	Akurasi: 0.9760479041916168 Presisi: 0.9773426120731509 Recall: 0.9760479041916168 F1-Score: 0.9761284448646566

Setelah dilakukan beberapa percobaan, dapat diketahui bahwa hasil seleksi fitur dengan 20% hingga 70% dari total fitur yang sama menghasilkan nilai akurasi yang sama.

Tahap Klasifikasi

`# Split the data into train and test`	Pada proses ini dilakukan

sets X = df_selected_features y = preprocessed_data['Label'].apply(lambda x: 'negatif' if x == 0 else 'positif') X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=40) # Train the Multinomial Naive Bayes model model = MultinomialNB() model.fit(X_train, y_train) # Make predictions on the test set y_pred = model.predict(X_test) return y_pred	proses klasifikasi menggunakan Multinomial Naive Bayes. Sebelumnya data dipisah menjadi fitur X yaitu kata atau token-token hasil dari seleksi fitur dan fitur y yang merupakan label kategori atau class dari data. Label kategori dalam fitur y masih berupa angka sehingga dilakukan proses untuk mengubah angka tersebut menjadi nilai sentimen negatif dan sentimen positif. Kemudian data dibagi menjadi data latih dan data uji menggunakan library train_test_split. Setelah itu dilakukan proses training data pada model Multinomial Naive Bayes.

Evaluasi Model

```
from sklearn import metrics

# Hitung metrik evaluasi
accuracy = metrics.accuracy_score(y_test, y_pred)
precision = metrics.precision_score(y_test, y_pred, average='weighted')
recall = metrics.recall_score(y_test, y_pred, average='weighted')
f1_score = metrics.f1_score(y_test, y_pred, average='weighted')

# Tampilkan hasil
print("Akurasi:", accuracy)
print("Presisi:", precision)
print("Recall:", recall)
print("F1-Score:", f1_score)
```

```
Akurasi: 0.9880239520958084
Presisi: 0.988356620093147
Recall: 0.9880239520958084
F1-Score: 0.9880459152118297
```

Untuk mengukur evaluasi kinerja model yang telah dilatih, digunakan library metrics dari modul sklearn. Dengan library metrics, dapat dihitung nilai akurasi, presisi, recall, dan f1-

score. Berdasarkan beberapa percobaan pada penggunaan seleksi fitur, maka digunakan penggunaan fitur sebanyak 50% dari total fitur. Dari tahapan klasifikasi ini didapatkan hasil akurasi tertinggi yaitu nilai akurasi sebesar 98,8%, nilai presisi sebesar 98,83%, nilai recall sebesar 98,8%, dan nilai f1-score sebesar 98,8%

Tahap Deployment Website

```
def main():
    st.title("Aplikasi Analisis Sentimen")
    st.write("Aplikasi Sistem Analisis Sentimen Dibangun Dengan metode Naive Bayes")

    # File upload widget
    st.subheader("Silahkan Upload File Anda")
    uploaded_file = st.file_uploader("Upload a file", type=["csv", "xlsx"])

    if uploaded_file is not None:
        # Read the uploaded file
        if uploaded_file.name.endswith('.csv'):
            data = pd.read_csv(uploaded_file)
        elif uploaded_file.name.endswith('.xlsx'):
            data = pd.read_excel(uploaded_file)
        else:
            st.error("Invalid file format. Please upload a CSV or Excel file.")
            return

        # Define stopwords dictionary
        stopwords_dictionary = [
            'gak', 'masa', 'bisa', 'lagi', 'banget', 'sama', 'nya', 'saya', 'semua', 'kalo', 'saat', 'sambil', 'ya',
            'untuk', 'segitu', 'lain', 'sih', 'sangat', 'tidak', 'yang', 'tapi', 'itu', 'aduh', 'lah', 'buat', 'mah',
            'tahu', 'apa', 'mau', 'banyak', 'di', 'karena', 'bakal', 'padahal', 'ni', 'orang', 'terus', 'lain', 'sini',
            'hanya', 'dengan', 'aja', 'dan', 'ada', 'sekali', 'udh', 'kali', 'walaupun', 'pdhl', 'dari', 'cuma', 'juga',
            'sesuai', 'ini', 'jadi', 'tt'
        ]
```

```
    # Train the model and make predictions
    if st.button("Analisis Sentiment"):
        predictions = train_model(data)

        # Display the data and predictions
        st.subheader("Data")
        st.write(data)

        st.subheader("Predictions")
        st.write(predictions)

        # Menghitung total sentimen
        total_negatif = (predictions == 'negatif').sum()
        total_positif = (predictions == 'positif').sum()

        st.header('Summary')
        st.write("Total Sentimen Positif: ", total_positif)
        st.write('Total Sentimen Negatif: ', total_negatif)
```

Pada tahapan deployment kami menggunakan streamlit yaitu framework berbasis python dan bersifat open-source yang dibuat untuk memudahkan dalam membangun aplikasi web. Pada website ini hanya dapat menerima inputan beberapa data teks berupa file, kemudian outputnya berupa hasil sentimen atau identifikasi ulasan dari file teks yang diinputkan.

BAB III
PENUTUP

3.1 Kesimpulan

Sentimen analisis adalah suatu metode yang digunakan untuk mengidentifikasi dan mengklasifikasikan sentimen atau ulasan ke dalam sentimen positif maupun negatif. Untuk melakukan sentimen analisis diperlukan beberapa tahap pemrosesan agar proses sentimen analisis bekerja dengan efisien serta memberikan hasil yang akurat. Proses pertama yaitu text preprocessing yang bertujuan untuk mempersiapkan data teks agar menjadi data yang lebih terstruktur. Tahapan dalam text preprocessing yaitu cleaning data, tokenization, stopword removal, dan stemming. Proses kedua yaitu ekstraksi fitur Term Frequency (TF) yang bertujuan mengubah data teks menjadi representasi numerik agar lebih mudah diolah oleh model pembelajaran mesin. Proses ketiga yaitu seleksi fitur Chi Square yang bertujuan untuk mengurangi penggunaan fitur yang tidak relevan dalam proses analisis. Proses keempat yaitu tahap klasifikasi menggunakan Multinomial Naive Bayes.

Berdasarkan percobaan klasifikasi sentimen yang telah dilakukan, diketahui bahwa model klasifikasi yang dibangun dapat melakukan sentimen analisis dengan baik. Dalam proses evaluasi didapatkan nilai akurasi tertinggi sebesar 98,8%, nilai presisi sebesar 98,83%, nilai recall sebesar 98,8%, dan nilai f1-score sebesar 98,8%.

DAFTAR PUSTAKA

[1] Ahmad Zuli Amrullah, A. S. (2020). Analisis Sentimen Movie Review Menggunakan Naive Bayes Classifier Dengan Seleksi Fitur Chi Square. *Jurnal BITe Vol.2, No. 1*, 40-44.

[2] Atmadja, B. R. (2022). Analisis Sentimen Bahasa Indonesia Pada Tempat Wisata di Kabupaten Sukabumi Dengan Naive Bayes. *Jurnal Ilmiah Elektronika dan Komputer, Vol.15, No.2*, 371-382.

[3] Fika Hastarita Rachman, I. (2022). Pendekatan Data Science untuk Mengukur Empati Masyarakat terhadap Pandemi Menggunakan Analisis Sentimen dan Seleksi Fitur. *JEPIN (Jurnal Edukasi dan Penelitian Informatika, Vol. 8, No. 3*, 492-499.

[4] Sartika Mandasari, B. H. (2022). Analisis Sentimen Pengguna Transportasi Online Terhadap Layanan Grab Indonesia Menggunakan Multinomial Naive Bayes Classifier. *Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD Vol.5, No. 2*, 118-126.