LAPORAN *FINAL PROJECT*

PENGANTAR PEMROSESAN DATA MULTIMEDIA

Disusun Oleh:

Kelompok 5

Kelas A

I Putu Hanggara Diatha Putra	1908561083
I Kadek Dwi Adnyana	2108561037
Getzbie Alfredo Tpoy	2108561091
Sandrina Ferani Aisyah Putri	2108561060

PROGRAM STUDI INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS UDAYANA

JIMBARAN

2022/2023

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi menuju era digital yang semakin pesat telah memberikan kemudahan yang besar bagi pengguna dalam berbagai aspek kehidupan. Salah satu aspek yang semakin berkembang adalah penggunaan multimedia. Data multimedia terdiri dari teks, gambar, suara, animasi, video [1] yang mengandung informasi beragam dan kompleks, sehingga memerlukan pengolahan yang tepat untuk menghasilkan nilai tambah dari data tersebut dan pengalaman yang lebih baik bagi pengguna. Pentingnya pemrosesan data multimedia semakin terasa karena semakin banyaknya data multimedia yang tersedia yang semakin rumit dan besar.

Salah satu bentuk data yang semakin banyak adalah data teks, seperti pesan singkat, email, postingan di media sosial, atau laporan bisnis. Dalam situasi seperti ini, *text mining* menjadi sangat penting karena merupakan penerapan konsep dan teknik data mining untuk mencari pola dalam teks, yaitu proses penganalisisan teks guna menyarikan informasi yang bermanfaat untuk tujuan tertentu [2]. Terdapat beberapa area penerapan dalam *text mining* antara lain pengenalan pola (*pattern recognition*), klasifikasi, klastering, analisis sentiment, peringkasan, dan pendeteksi plagiarism [3].

Klasifikasi sentimen merupakan penerapan konsep dan teknik data mining yang bertujuan untuk memahami dan menginterpretasikan sentimen atau pendapat yang terkandung dalam teks sehingga memberikan wawasan tentang bagaimana suatu produk, layanan, atau topik tertentu dipersepsikan oleh pengguna atau konsumen. Klasifikasi sentimen dapat mengidentifikasi sentimen positif dan negatif. Dengan menganalisis sentimen yang terkandung dalam teks seperti ulasan produk, komentar media sosial, atau feedback pelanggan, perusahaan dapat memperoleh informasi berharga untuk meningkatkan produk, layanan, dan kepuasan pelanggan. Pada final project ini akan membahas mengenai analisis sentimen menggunakan *multinomial naive bayes* dan *chi square*.

1.2 Rumusan Masalah

1. Bagaimana tahapan membangun sistem aplikasi untuk mengidentifikasi sentimen sebuah atau beberapa ulasan?
2. Bagaimana perbandingan tingkat performa (*accuracy, precision, recall,* dan *F-1 score*) metode *Multinomial Naive Bayes* dengan pengaruh variasi jumlah fitur dari seleksi fitur *Chi Square*?

1.3 Tujuan

1. Untuk mengetahui tahapan membangun sistem aplikasi untuk mengidentifikasi sentimen sebuah atau beberapa ulasan?
2. Untuk mengetahui perbandingan tingkat performa (*accuracy, precision, recall,* dan *F-1 score*) metode *Multinomial Naive Bayes* dengan pengaruh variasi jumlah fitur dari seleksi fitur *Chi Square*?

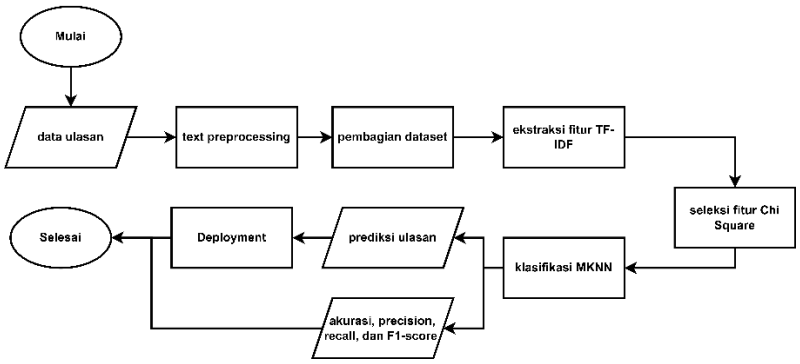
1.4 Manfaat

1. Dapat menambah wawasan mengenai proses klasifikasi *text* berupa ulasan dan melakukan pengecekan pada suatu ulasan apakah teridentifikasi positif atau negatif.
2. Dapat mengetahui perbandingan tingkat performa (*accuracy, precision, recall,* dan *F-1 score*) metode *Multinomial Naive Bayes* dengan pengaruh variasi jumlah fitur dari seleksi fitur *Chi Square*

BAB II
MANUAL APLIKASI

2.1 Fitur Sistem

Fitur sistem program dapat dilihat pada tahapan di bawah ini.



Gambar 1. Tahapan proses klasifikasi

Tahapan awal yang dilakukan dalam mengklasifikasikan sentimen ini adalah preprocessing yang meliputi tokenization, case folding, normalization, stop word removal, stemming. Sebelum dilanjutkan ke pembobotan TF dan seleksi fitur, dataset dibagi terlebih dahulu menjadi data training dan data testing. Total jumlah fitur yang digunakan adalah dari setiap data text (ulasan) tergantung dari panjang atau jumlah vocabulary setelah preprocessing (jumlah term indeks). Feature selection yang digunakan adalah formula Chi-Square dengan mempertahankan beberapa variasi jumlah fitur dari 10%, 20%, 30%, hingga 100%. Tahapan selanjutnya adalah tahapan klasifikasi menggunakan metode Multinomial Naive Bayes. Dari data testing digunakan ukuran evaluasi dari confusion matrix yang menghasilkan akurasi, precision, recall, dan F1-Score. Terakhir adalah tahapan deployment ke sistem aplikasi berbasis web dengan fitur utama adalah user menginput satu data ulasan dan outputnya adalah hasil prediksi sentiment.

2.1.1 Tampilan Program

Pada program menampilkan hasil dari fitur sistem sebelum mengidentifikasi kelas ulasan, mulai dari prerprocessing, pembagian data, pembobotan TF, seleksi fitur, klasifikasi, evaluasi menggunakan tabel confusion matrix.

	No	Reviews	Label
0	1	kemeja nya bagusss bgtttt😍😍😍aaaa mauuu nngisssss😫😫😫knpa ga dri dlu beli kemeja ditoko ini😍, ini kemejanya asli emg bagus, bahannya jga adem ga gerah,and ga nerawang jga... itu krna camera nya jelek jdi ga trlalu jelas kemejanya.. asliny baguss bgttt ga bhong sumpah	1
1	2	Jahitannya sih rapi,cuman ada benang yang ikut ke jahit juga jadi agak jelek	0
2	3	Sesuai harga. Agak tipis tapi masih oke kok. Warnanya abu tapi kalo difoto emang kayak biru dikit. Thanks sellerr	0
3	4	Wah gila sihhh sebagus itu, se worth it, se lembut itu bajunya,,,, kirain bakal terlalu tipis ky kemeja ku yg lama ternyata sedikit ky kaos bajunya baklgus bettt dah dgn harga segitu sih worth it thankyou yh,, next bakal order lgi	1
4	5	Kain nya bagus halus \nTapi kok di bukak kotor ya warna putih lagi	0
...	...	...	...
826	827	Terima kasih barang sudah sampai sesuai ukuran dan seesuai gambar bagus terima kasih ya	1
827	828	Mantapp realpicttt bangttt tapi pengemasan nya cuman plastik aja ku kira pake kardus tapi nda papa bagus polll	1
828	829	Suka bgt sama tasnya, ga kayak tas local. Keren parah pokoknyaaaaa. Suka bgt sama tasnya, ga kayak tas local. Keren parah pokoknyaaaaa. Suka bgt sama tasnya, ga kayak tas local. Keren parah pokoknyaaaaa. \nSuka bgt sama tasnya, ga kayak tas local. Keren parah pokoknyaaaaa. Suka bgt sama tasnya, ga ka	1
829	830	kualitas produk sangat baik. produk original. harga produk sangat baik.	1
830	831	Barang udah sampai dg selamat, mantul banget dah bajunya sesuai gambar lembut tebal halus pokoknya enak d pakai makasih tokonya yg sll amanah, semoga tambah laris, dan juga kurir nya yg ramah.	1

831 rows × 3 columns

Gambar 2. Tampilan Data Ulasan

	No	Reviews	Label	Tokenization	Normalization	Stopword Removal	Stemming
0	1	kemeja nya bagusss bgtttaaaa mauuu nngisssssknpa ga dri dlu beli kemeja ditoko ini ini kemejanya asli emg bagus bahannya jga adem ga gerahand ga nerawang jga itu krna camera nya jelek jdi ga trlalu jelas kemejanya asliny baguss bgttt ga bhong sumpah	1	[kemeja, nya, bagusss, bgtttaaaa, mauuu, nngisssssknpa, ga, dri, dlu, beli, kemeja, ditoko, ini, ini, kemejanya, asli, emg, bagus, bahannya, jga, adem, ga, gerahand, ga, nerawang, jga, itu, krna, camera, nya, jelek, jdi, ga, trlalu, jelas, kemejanya, asliny, baguss, bgttt, ga, bhong, sumpah]	[kemeja, nya, bagus, banget, mau, nangis kenapa, tidak, dari, dulu, beli, kemeja, ditoko, ini, ini, kemejanya, asli, memang, bagus, bahannya, juga, adem, tidak, gerahan, tidak, nerawang, juga, itu, karena, camera, nya, jelek, jadi, tidak, terlalu, jelas, kemejanya, aslinya, bagus, banget, tidak, bohong, sumpah]	[kemeja, bagus, banget, nangis kenapa, beli, kemeja, ditoko, kemejanya, asli, bagus, bahannya, adem, gerahan, nerawang, camera, jelek, kemejanya, aslinya, bagus, banget, bohong, sumpah]	[kemeja, bagus, banget, nang kenapa, beli, kemeja, toko, kemeja, asli, bagus, bahan, adem, gerah, nerawang, camera, jelek, kemeja, asli, bagus, banget, bohong, sumpah]
1	2	jahitannya sih rapicuman ada benang yang ikut ke jahit juga jadi agak jelek	0	[jahitannya, sih, rapicuman, ada, benang, yang, ikut, ke, jahit, juga, jadi, agak, jelek]	[jahitannya, sih, rapi cuman, ada, benang, yang, ikut, ke, jahit, juga, jadi, agak, jelek]	[jahitannya, rapi cuman, benang, jahit, jelek]	[jahit, rapi cuman, benang, jahit, jelek]
2	3	sesuai harga agak tipis tapi masih oke kok warnanya abu tapi kalo difoto emang kayak biru dikit thanks sellerr	0	[sesuai, harga, agak, tipis, tapi, masih, oke, kok, warnanya, abu, tapi, kalo, difoto, emang, kayak, biru, dikit, thanks, sellerr]	[sesuai, harga, agak, tipis, tapi, masih, oke, kok, warnanya, abu, tapi, kalau, di foto, memang, seperti, biru, dikit, thanks, sellerr]	[sesuai, harga, tipis, oke, warnanya, abu, di foto, biru, dikit, thanks, sellerr]	[sesuai, harga, tipis, oke, warna, abu, di foto, biru, dikit, thanks, sellerr]
3	4	wah gila sihhh sebagus itu se worth it se lembut itu bajunya kirain bakal terlalu tipis ky kemeja ku yg lama ternyata sedikit ky kaos bajunya baklgus bettt dah dgn harga segitu sih worth it thankyou yh next bakal order lgi	1	[wah, gila, sihhh, sebagus, itu, se, worth, it, se, lembut, itu, bajunya, kirain, bakal, terlalu, tipis, ky, kemeja, ku, yg, lama, ternyata, sedikit, ky, kaos, bajunya, baklgus, bettt, dah, dgn, harga, segitu, sih, worth, it, thankyou, yh, next, bakal, order, lgi]	[wah, gila, sih, sebagus, itu, se, worth, it, se, lembut, itu, bajunya, kirain, bakal, terlalu, tipis, seperti, kemeja, saya, yg, lama, ternyata, sedikit, seperti, kaos, bajunya, bagus, banget, dah, dengan, harga, segitu, sih, worth, it, thankyou, ya, next, bakal, order, lagi]	[gila, sebagus, worth, it, lembut, bajunya, kirain, tipis, kemeja, kaos, bajunya, bagus, banget, dah, harga, segitu, worth, it, thankyou, ya, next, order]	[gila, bagus, worth, it, lembut, baju, kirain, tipis, kemeja, kaos, baju, bagus, banget, dah, harga, segitu, worth, it, thankyou, ya, next, order]

Gambar 3. Tampilan Hasil Preprocessing

```
Jumlah data latih positif : 302
Jumlah data latih negatif : 362
Jumlah data uji positif : 83
Jumlah data uji negatif : 84
Jumlah data             : 831
```

Gambar 4. Tampilan Hasil Pembagian Data

TF vector train:

	aaa	aaaa	aaaaaa	aamiin	abal	abang	abis	abu	acara	ad	...	yeaaayyyyy	yen	yha	ynag	yng	you	youtube	youu	yung
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
659	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
660	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
661	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
662	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
663	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0

664 rows × 1926 columns

TF vector Test:

	aaa	aaaa	aaaaaa	aamiin	abal	abang	abis	abu	acara	ad	...	yeaaayyyyy	yen	yha	ynag	yng	you	youtube	youu	yung
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
162	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
163	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
164	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
165	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
166	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0

167 rows × 1926 columns

Gambar 5 dan 6. Tampilan Hasil Pembobotan Kata Data Tes dan Latih

	aaa	aaaa	aaaaaa	aamiin	abal	abang	abis	abu	acara	ad	...	yeaaayyyyy	yen	yha	ynag	yng	you	youtube	youu	yung
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
659	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
660	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
661	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
662	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
663	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0

664 rows × 1926 columns

Selected Features in Test Set:

	aaa	aaaa	aaaaaa	aamiin	abal	abang	abis	abu	acara	ad	...	yeaaayyyyy	yen	yha	ynag	yng	you	youtube	youu	yung
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
162	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
163	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
164	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
165	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
166	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0

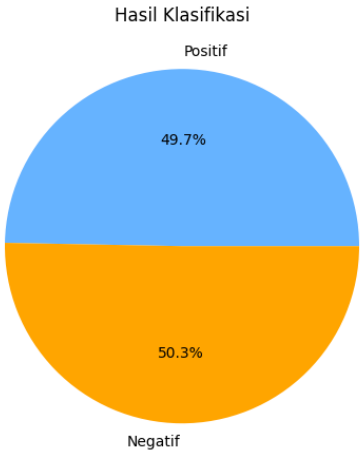
167 rows × 1926 columns

Gambar 7 dan 8. Tampilan Hasil Penyeleksian Kata Data Tes dan Latih

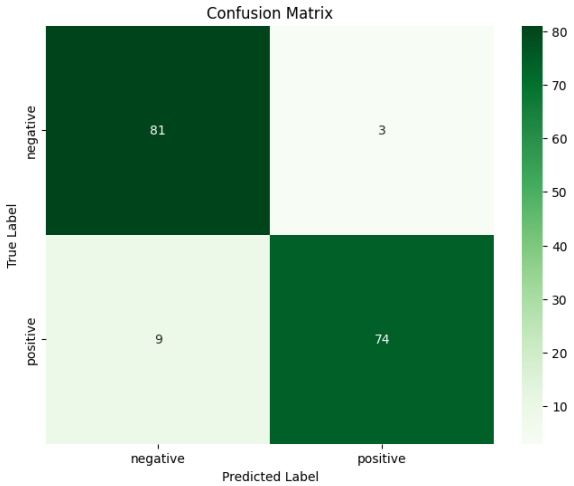
	Data Testing	Label Asli	Label Prediksi
610	0	0	0
818	1	1	1
290	2	0	0
559	3	0	0
168	4	1	1
...	...	...	...
192	162	1	1
650	163	0	0
456	164	0	0
773	165	1	1
531	166	0	0

167 rows × 3 columns

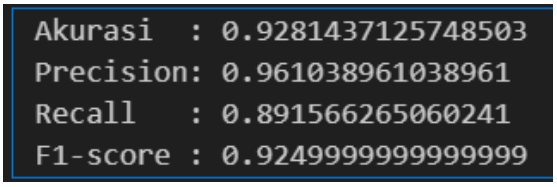
Gambar 9. Tampilan Hasil Perbandingan Label Asli dan Prediksi



Gambar 10. Tampilan Hasil Persentase Klasifikasi



Gambar 11. Tampilan Tabel Confusion Matrix



Gambar 11. Tampilan Hasil Perhitungan Tabel Confusion Matrix

2.1.2 Hasil Program dan Penjelasan

Hasil program pada file Jupyter Notebook dibuatkan sebanyak 10 file untuk mengetahui perbandingan tingkat performa (akurasi, presisi, recall, f1 score) dari pengaruh seleksi fitur *Chi-Square* dengan mempertahankan beberapa variasi jumlah fitur dari 10%, 20%, 30%, hingga 100%. Dapat dilihat pada tabel di bawah bahwa yang menghasilkan akurasi tertinggi adalah dengan mempertahankan 100% jumlah fitur sehingga model ini yang digunakan untuk tahapan klasifikasi sentimen ulasan pada sistem aplikasi yang dibangun.

Tabel 4. Performa MNB modelling use 100%

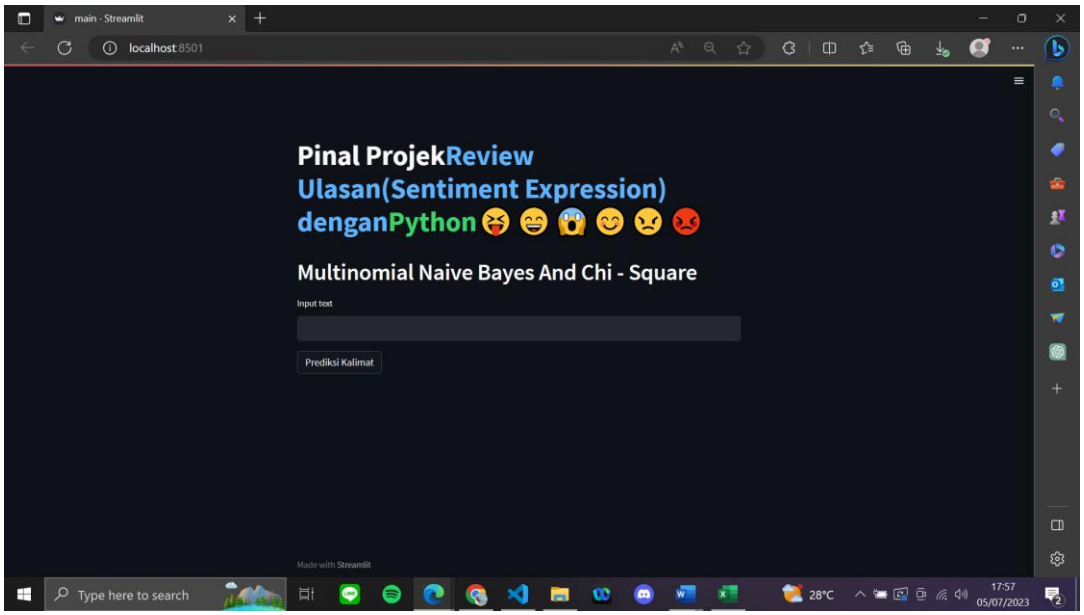
Chi Square		Akurasi	Presisi	Recall	F1-Score
Persentase	Jumlah Fitur				
10%	193	88%	92%	83,1%	87,3%
20%	385	91,1%	92,5%	89,1%	90,7%
30%	557	91,6%	93,6%	89,1%	91,3%
40%	770	88,6%	93,2%	83,1%	87,8%
50%	963	87,4%	93%	80,7%	86,4%
60%	1155	88,6%	93,2%	83,1%	87,8%
70%	1348	89,2%	94,5%	83,1%	88,4%
80%	1540	90%	94,6%	85,5%	89,8%
90%	1733	91,6%	94,8%	87,9%	91,2%
100%	1926	92,8%	96,1%	89,1%	92,4%

2.2 Antar Muka Aplikasi

2.2.1 Halaman Utama

Halaman utama merupakan interface pertama yang akan dilihat oleh user apabila memasuki website. Fitur utama dalam aplikasi yang merupakan tempat untuk pengguna melakukan identifikasi sentimen ulasan dengan cara memasukkan

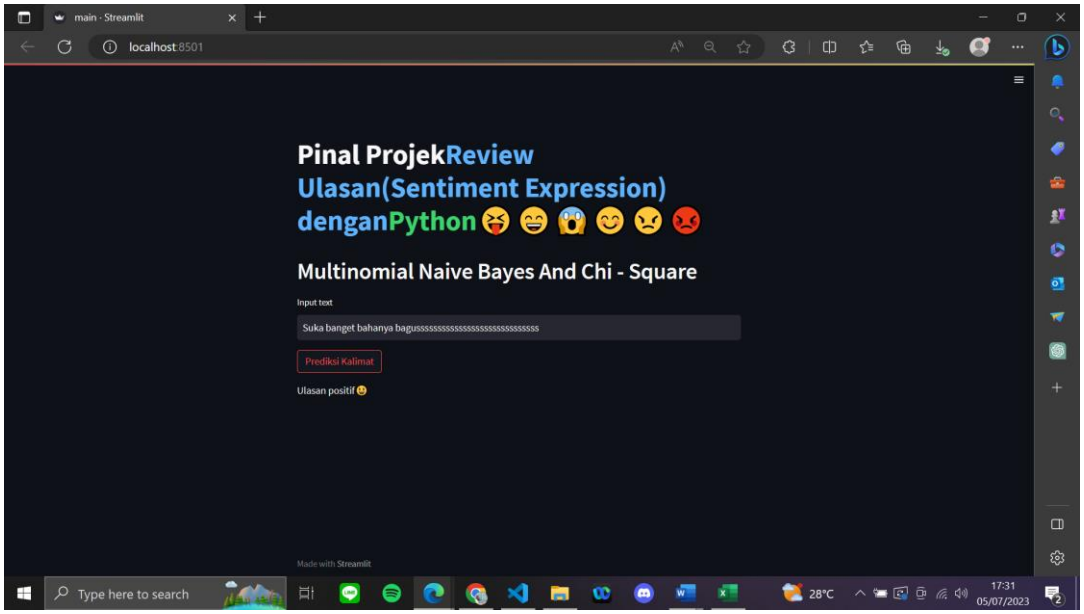
terlebih dahulu ulasan dalam text field, lalu menekan tombol “Prediksi Kalimat” untuk selanjutnya diidentifikasi dan ditampilkan status ulasan tersebut, apakah ulasan positif atau negatif.



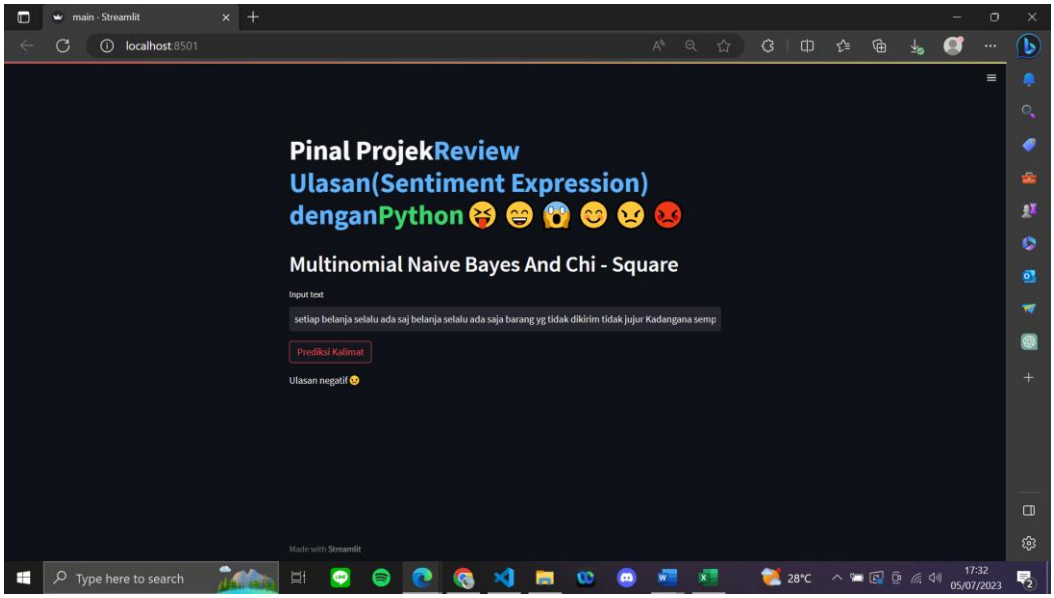
Gambar 12. Halaman Utama

2.2.2 Status Sentimen Positif atau Negatif

Menampilkan status sentiment dari ulasan yang telah dimasukkan oleh pengguna. Ada 2 jenis status ulasan, yakni pertama “Ulasan positif” dengan emoji senyum dan “Ulasan Negatif” dengan emoji marah.



Gambar 13. Status Sentimen Positif



Gambar 14. Status Sentimen Negatif

2.3 Implementasi (*Coding*) dan Penjelasan

Alur pembuatan program dilakukan dengan membuat *preprocessing*, pemodelan, dan *deploy* model ke sebuah website. Pada tahapan *preprocessing* dan pemodelan dikerjakan pada Jupyter Notebook untuk mencari model terbaik. Tahap deploy dikerjakan pada file .py dan interface menggunakan library streamlit.

2.3.1 File Preprocessing.ipynb

Tabel 2. File Preprocessing.ipynb

Kode Program	Penjelasan
1. import pandas as pd 2. reviews_data = pd.read_excel("reviews.xlsx") 3. pd.set_option('display.max_colwidth', None) 4. reviews_data	Mengimport library pandas yang digunakan untuk memetakan data ke dalam dataframe. Kemudian membuka file yang berisi teks yang akan di preprocessing
1. reviews_data["Reviews"] = reviews_data["Reviews"].str.lower() 2. print("Case Folding :\n") 3. reviews_data	Mengubah semua teks yang disimpan pada kolom Reviews menjadi huruf kecil
1. import string 2. import re #regex library 3. # import word_tokenize & FreqDist from NLTK 4. from nltk.tokenize import word_tokenize 5. from nltk.probability import FreqDist	Mengimport library string, regex, dan nltk yang digunakan untuk membersihkan beberapa kata dan mengubahnya menjadi token
6. def remove_Reviews_special(text): 7. # remove tab, new line, ans back slice 8. text = text.replace('\\t'," ").replace('\\n'," ").replace('\\u'," ").replace('\\',"")	Fungsi remove_Reviews_special digunakan untuk menghilangkan beberapa karakter pada teks seperti tabs,

9. # remove non ASCII (emoticon, chinese word, .etc) 10. **text = text.encode('ascii', 'replace').decode('ascii')** 11. # remove mention, link, hashtag 12. text = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ", text).split()) 13. # remove incomplete URL 14. return text.replace("http://", " ").replace("https://", " ")	Unicode, menghapus karakter non ASCII, menggunakan regex untuk menghilangkan seperti mention, dan menghilangkan link.
15. **reviews_data['Reviews'] = reviews_data['Reviews'].apply(remove_Reviews_special)**	Memanggil fungsi remove_Reviews_special dan menerapkannya pada kolom Reviews
16. #remove number 17. def remove_number(text): 18. return re.sub(r"\d+", "", text)	Fungsi remove_number untuk menghilangkan angka dan menganntinya dengan string kosong
19. reviews_data['Reviews'] = reviews_data['Reviews'].apply(remove_number)	Memanggil fungsi remove_number dan menerapkannya pada kolom Reviews
20. **#remove punctuation** 21. def remove_punctuation(text): 22. return text.translate(str.maketrans("","",string.punctuation))	Fungsi remove_punctuation digunakan untuk menghapus tanda baca, dengan menggunakan fungsi maketrans, yang akn membuat tabel translansi tanpa tanda baca.

23. reviews_data['Reviews'] = reviews_data['Reviews'].apply(remove_punctuation)	Memanggil fungsi remove_punctuation dan menerapkannya pada kolom Reviews
24. #remove whitespace leading & trailing 25. def remove_whitespace_LT(text): 26. return text.strip()	Funsgi remove_whitespace_LT yang digunakan untuk menghilangkan spasi, newline, tab dengan mengggunakan fungsi strip()
27. reviews_data['Reviews'] = reviews_data['Reviews'].apply(remove_whitespace_LT)	Memanggil fungsi remove_whitespace_LT dan menerapkannya pada kolom Reviews
28. #remove multiple whitespace into single whitespace 29. def remove_whitespace_multiple(text): 30. return re.sub('\s+',' ',text)	Fungsi remove_whitespace_multiple untuk menghilangkan spasi, newline, dan tab dengan menggunakan regex.
31. reviews_data['Reviews'] = reviews_data['Reviews'].apply(remove_whitespace_multiple)	Memanggil fungsi remove_whitespace_multiple dan menerapkannya pada kolom Reviews
32. # remove single char 33. def remove_singl_char(text): 34. return re.sub(r"\b[a-zA-Z]\b", "", text)	Fungsi remove_singl_char untuk menghilangkan karakter tunggal dengan menggunakan regex
35. reviews_data['Reviews'] = reviews_data['Reviews'].apply(remove_singl_char)	Memanggil fungsi remove_singl_char dan menerapkannya pada kolom Reviews

36. # NLTK word rokenize 37. def word_tokenize_wrapper(text): 38. return word_tokenize(text)	Fungsi word_tokenize_wrapper digunakan untuk memecah kata menjadi token dengan menggunakan fungsi word_tokenize
39. reviews_data['Tokenization'] = reviews_data['Reviews'].apply(word_tokenize_wrapper) 40. print('Hasil Tokenizing dan Cleaning: \n') 41. reviews_data	Membuat sebuah kolom baru pada dataframe dengan nama Tokenization dan menyimpan hasil tokenisasi data pada kolom Reviews kedalam kolom tersebut dengan memanggil fungsi word_tokenize_wrapper
1. normalizad_word = pd.read_excel("normalisasi.xlsx") 2. normalizad_word_dict = {}	Membaca file normalisasi yang akan digunakan untuk normalisasi kata yang rusak, dan membuat sebuah dictionary untuk proses selanjutnya.
3. for index, row in normalizad_word.iterrows(): 4. if row[0] not in normalizad_word_dict: 5. normalizad_word_dict[row[0]] = row[1]	Perulangan untuk melakukan pengecekan kata pada row, dengan mengecek kunci pada library. Tujuannya untuk membangun dictionary normalizad_word_dict
6. def normalized_term(document): 7. return [normalizad_word_dict[term] if term in normalizad_word_dict else term for term in document]	Fungsi untuk melakukan normalisasi kata pada document berdasrkan dictionary normalizad_word_dict

8. reviews_data['Normalization'] = reviews_data['Tokenization'].apply (normalized_term) 9. reviews_data	Membuat kolom Normalization untuk menyimpan hasil normalisasi, fungsi normalisasi digunakan pada kolom data Tokenization
1. from nltk.corpus import stopwords	Meengiport fungsi stopwords removal pada library nltk
2. list_stopwords = stopwords.words('indonesian')	Mengambil data stopwords pada fungsi word dengan bahasa Indonesia
3. list_stopwords.extend(["yg", "nya", "sih"])	Menambahkan stopwords secara manual dengan menggunakan extend
4. txt_stopword = pd.read_csv("stopwords.txt", names= ["stopwords"], header = None)	Menambahkan stopwords dengan membaca file stopwords.txt
5. list_stopwords.extend(txt_stopword ["stopwords"][0].split(' '))	Konversi data stopwords ke dalam list
6. list_stopwords = set(list_stopwords)	Konversi ke dictionary
7. def stopwords_removal(words): 8. return [word for word in words if word not in list_stopwords] 9. reviews_data['Stopword Removal'] = reviews_data['Normalization'].apply(stopwords_removal) **10. reviews_data**	Membuat fungsi stopwords removal untuk melakukan pengubahan stopwords. Kemudian membuat kolom baru dengan nama Stopword Removal untuk menyimpan hasil storpword removal. Fungsi stoword removal

	digunakan pada kolom Normalization
1. from Sastrawi.Stemmer.StemmerFactory import StemmerFactory 2. import swifter	Import fungsi stemmerfactory pada library sastrawi yang digunakan untuk melakukan stemming
3. factory = StemmerFactory() 4. stemmer = factory.create_stemmer()	Membuat stemmer
5. def stemmed_wrapper(term): 6. return stemmer.stem(term) 7. term_dict = {} 8. for document in reviews_data['Stopword Removal']: 9. for term in document: 10. if term not in term_dict: 11. term_dict[term] = ' ' 12.print(len(term_dict)) 13.print("------------------------") 14.for term in term_dict: 15. term_dict[term] = stemmed_wrapper(term) 16. print(term,":",term_dict[term]) 17.print(term_dict) 18.print("------------------------")	Fungsi untuk melakukan stemming dengan nama stemmed_wrapper. Fungsi akan membaca dokumen yang diberikan dan melakukan proses stemming dengan bantuan fungsi stemmer pada library sastrawi.
19.def get_stemmed_term(document): 20. return [term_dict[term] for term in document] 21.reviews_data['Stemming'] = reviews_data['Stopword Removal'].swifter.apply(get_stemmed_term) 22.reviews_data	Fungsi get_stemmed_term untuk mengubah data hasil stemming ke dalam bentuk dataframe Kemudian membuat kolom Stemming untuk menyimpan hasil stemming yang dilakukan pada kolom

	Stopword Removal dan menggunakan swifter untuk mempercepat prosesnya.
1. reviews_data.to_excel("reviews_Preprocessing.xlsx")	Menyimpan hasil preprocessing ke dalam file reviews_Preprocessing.xlsx

2.3.2 File MNB Modelling.ipynb

Pada tahapan pemodelan terdapat beberapa file yang ada, dikarenakan ada beberapa seleksi fitur yang digunakan. Pada laporan akan dicontohkan penjelasan program dengan menggunakan seleksi fitur 100 persen yang dipertahankan, pada file **MNB modelling use 100 percent.ipynb**

Tabel 3. File MNB modelling.ipynb

Kode program	Penjelasan
1. import pandas as pd 2. import numpy as np 3. reviews_data = pd.read_excel("reviews_Preprocessing.xlsx", usecols=["Label", "Stemming"]) 4. pd.set_option('display.max_colwidth', None) 5. **reviews_data.columns = ["label", "reviews"]** 6. reviews_data	Mengimport lib pandas dan numpy yang akan digunakan untuk mengolah dataframe. Kemudian membuka file hasil preprocessing, kolom yang digunakan adalah kolom Label dan stemming pada reviews_Preprocessing.xlsx
1. import ast 2. def join_text_list(texts): 3. texts = ast.literal_eval(texts) 4. return ' '.join([text for text in texts]) 5. **reviews_data["reviews"] = reviews_data["reviews"].apply(join_text_list)** 6. reviews_data	Mengimport library ast yang digunakan untuk mengubah string yang mewakili list menjadi list sesungguhnya Kemudian membuat fungsi join_text_list untuk

	mengubah string ke bentuk list sesungguhnya. kolom reviews digunakan untuk menyimpan hasil penggunaan fungsi join_text_list pada kolom reviews.
1. label = reviews_data["label"] 2. text = reviews_data["reviews"]	Memisahkan label dan teks yang digunakan
1. from sklearn.model_selection import train_test_split 2. train_data, test_data, train_labels, test_labels = train_test_split(text, label, test_size=0.2, random_state=42) 3. positive_count_train = (train_labels == 1).sum() 4. negative_count_train = (train_labels == 0).sum() 5. total_count_train = len(train_labels) 6. # Mencetak jumlah data positif dan negatif pada data latih 7. print("Jumlah data latih positif :", positive_count_train) 8. print("Jumlah data latih negatif :", negative_count_train) 9. positive_count_test = (test_labels == 1).sum() 10. negative_count_test = (test_labels == 0).sum() 11. total_count_test = len(test_labels) 12. # Mencetak jumlah data positif dan negatif pada data uji 13. print("Jumlah data uji positif :", positive_count_test) 14. print("Jumlah data uji negatif :", negative_count_test) 15. total_count = positive_count_train + negative_count_train +	Mengimport library yang digunakan untuk melakukan splitting data, Melakukan splitting data train dan test dengan rasio 80:20. Kemudian untuk mencetak jumlah data negative dan positive pada data latih dan uji dengan menggunakan .sum()

positive_count_test + negative_count_test 16. print("Jumlah data :", total_count)	
1. from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer 2. # Create CountVectorizer 3. cvect = CountVectorizer() 4. TF_vector_train = cvect.fit_transform(train_data) 5. # Convert TF vector to dataframe 6. TF_df_train = pd.DataFrame(TF_vector_train.toarray(), columns=cvect.get_feature_names_out()) 7. # Display the dataframes 8. print("TF vector train:") 9. TF_df_train 10. # Perhitungan TF vector pada test set menggunakan CountVectorizer yang sudah dilatih pada train set 11. TF_vector_test = cvect.transform(test_data) 12. # Convert TF vector to dataframe 13. TF_df_test = pd.DataFrame(TF_vector_test.toarray(), columns=cvect.get_feature_names_out()) 14. print("TF vector Test:") 15. TF_df_test	Mengimport library dan fungsi yang digunakan untuk menghitung TF, vector, Menghitung TF pada train set dan test set dengan menggunakn fit_transform
1. from sklearn.feature_selection import SelectPercentile, chi2 2. # Percentage of features to select (100%) 3. percent = 100 4. # Calculate the number of desired features based on the percentage 5. k = int(percent / 100 * TF_vector_train.shape[1])	Mengimport fungsi yang chi2 yang digunakan untuk melakukan seleksi fitur dengan metode chi-square. Presentase yang digunakan

6. # Apply feature selection with chi-square on the train set 7. selector = SelectPercentile(chi2, percentile=percent) 8. tf_mat_train_selected = selector.fit_transform(TF_vector_train, train_labels) 9. # Apply the same feature selection on the test set **10. tf_mat_test_selected = selector.transform(TF_vector_test)** 11. # Get selected feature names 12. selected_feature_names = [feature_name **for** feature_name, selected **in** zip(cvect.get_feature_names_out(), selector.get_support()) **if** selected] 13. # Create dataframes for selected features 14. tf_df_train_selected = pd.DataFrame(tf_mat_train_selected.toarray(), columns=selected_feature_names) **15. tf_df_test_selected = pd.DataFrame(tf_mat_test_selected.toarray(), columns=selected_feature_names)** 16. # Display the dataframes 17. **print**("Selected Features in Train Set:") 18. tf_df_train_selected 19. print("\nSelected Features in Test Set:") 20. tf_df_test_selected	adalah 100 persen yang diterapkan pada set train. Selector digunakan untuk memilih fitur berdasarkan skor chi-square. Hasil seleksi fitur disimpan pada tf_mat_train/test_selected
21. **from** sklearn.naive_bayes **import** MultinomialNB 22. # Create the Multinomial Naive Bayes model with appropriate class_prior 23. model = MultinomialNB() 24. # Train the model using the selected training data **25. model.fit(tf_mat_train_selected, train_labels)**	Mengimport fungsi mesin MultinomialNB, kemudian membuat objek model dengan prioritas kelas negative dan positif

26. # Perform predictions on the selected test data 27. predictions = model.predict(tf_mat_test_selected) 28. # Create a dataframe with testing data, true labels, and predicted labels 29. results_df = pd.DataFrame({'Data Testing': tf_df_test_selected.index, 'Label Asli': test_labels, 'Label Prediksi': predictions}) 30. # Display the dataframe 31. results_df	Melatih model dengan data latih yang sudah disekesi Melakukan prediksi pada data uji yang sudah diseleksi. Mencetak hasil perbandingan label testing dengan label prediksi
32. import matplotlib.pyplot as plt 33. # Data untuk visualisasi 34. labels = ['Positif', 'Negatif'] 35. sizes = [positive_count_test, negative_count_test] 36. colors = ['#66B3FF', '#FFA500'] # Warna biru dan oranye 37. # Membuat pie chart 38. plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%') 39. plt.title('Hasil Klasifikasi\n') 40. # Menampilkan plot 41. plt.axis('equal') 42. plt.show()	Mengimport library matplotlib untuk menampilkan data hasil klasifikasi secara visual
1. from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report, confusion_matrix 2. import seaborn as sn 3. # Menghitung akurasi 4. accuracy = accuracy_score(test_labels, predictions) 5. print("Akurasi :", accuracy) 6. # Menghitung precision 7. precision = precision_score(test_labels, predictions)	Mengimport library sklearn metrics untuk melakukan evaluasi yang menghasilkan akurasi, presisi, recall dan f1 – score serta menampilkan langsung tabel confusion matriksnya

8. print("Precision:", precision) 9. # Menghitung recall 10. recall = recall_score(test_labels, predictions) 11. print("Recall :", recall) 12. # Menghitung F1-score 13. f1 = f1_score(test_labels, predictions) 14. print("F1-score :", f1) 15. # Generate confusion matrix 16. columns = ['negative', 'positive'] 17. confm = confusion_matrix(test_labels, predictions) 18. df_cm = pd.DataFrame(confm, index=columns, columns=columns) 19. # Create heatmap of confusion matrix 20. plt.figure(figsize=(8, 6)) 21. sn.heatmap(df_cm, cmap='Greens', annot=True, fmt='d') 22. plt.title('Confusion Matrix') 23. plt.xlabel('Predicted Label') 24. plt.ylabel('True Label') 25. plt.show()	
1. import joblib 2. joblib.dump(model, 'multinomial_nb_10 percent_model.pkl')	Model yang diperoleh disimpan dengan bantuan library joblib, dengan menggunakan fungsi dump.

2.3.3 File .py

Tahapan selanjutnya adalah melakukan deploy ke dalam sebuah bentuk website dengan bantuan library streamlit. Pada tahapan deploy program ditulis ulang yang berisikan tahapan preprocessing, vektorisasi, seleksi fitur, dan klasifikasi menggunakan model yang sudah dibuat. Pada tahapan deploy program dipecah menjadi 4 modul, yaitu **main.py, preprocessing.py, lib.py, dan main.py**.

main.py

Kode Program	Penjelasan

1. from lib import* 2. from model import* 3. from preprocessing import* 4. def main(): 5. st.title("Pinal Projek:blue[Review Ulasan(Sentiment Expression) dengan]:green[Python]:stuck_out_tongue_closed_eyes::smile::scream::blush::angry::rage:") 6. st.header('Multinomial Naive Bayes And Chi - Square') 7. teks = st.text_input('Input text') 8. hasil = preprocess_text(teks) 9. kalimat_normalisasi = normalized_term(hasil) 10. def convert_to_sentence(word_list): 11. sentence = ' '.join(word_list) 12. return sentence 13. kalimat = convert_to_sentence(kalimat_normalisasi) 14. if st.button('Prediksi Kalimat'): 15. prediction = predict_text(kalimat) 16.if __name__ == '__main__': 17. main()	Kode ini akan memanggil modul lainnya Pada modul main merupakan halaman utama program. Pengguna akan memasukan inputan kata yang kemudian akan di klasifikasikan oleh mesin dan memberikan output positif ataupun negative.

preprocessing.py

Kode program	Penjelasan
1. from lib import* 2. def preprocess_text(text): 3. text= text.lower() 4. # remove tab, new line, and backslash 5. text = text.replace('\\t', ' ').replace('\\n', ' ').replace('\\', '') 6. # remove non ASCII (emoticon, Chinese word, etc) 7. text = text.encode('ascii', 'replace').decode('ascii') 8. # remove mention, link, hashtag 9. text = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)", " ", text).split()) 10. # remove incomplete URL	Langkah preprocessing sama persis dengan sama dengan file ipynb sebelumnya

```
    text = text.replace("http://", "
").replace("https://", " ")
    # remove numbers
    text = re.sub(r"\d+", "", text)
    # remove punctuation
    text =
text.translate(str.maketrans("", "",
string.punctuation))
    # remove leading and trailing
whitespace
    text = text.strip()
    # remove multiple whitespace into
single whitespace
    text = re.sub('\s+', ' ', text)
    # remove single character
    text = re.sub(r"\b[a-zA-Z]\b", "",
text)
    # tokenize words
    tokens = word_tokenize(text.lower())
    # remove stopwords
    stopword_list =
set(stopwords.words('indonesian'))
    tokens = [word for word in tokens if
word not in stopword_list]
    return tokens
normalizad_word =
pd.read_excel("normalisasi.xlsx")
normalizad_word_dict = {}
for index, row in
normalizad_word.iterrows():
    if row[0] not in
normalizad_word_dict:
        normalizad_word_dict[row[0]] =
row[1]
def normalized_term(document):
    return [normalizad_word_dict[term] if
term in normalizad_word_dict else term
for term in document]
```

lib.py

Libraray ini berisikan semoa library yang digunakan pada program ini.

Kode program	Penjelasan

1. import streamlit as st 2. import pandas as pd 3. import string 4. import re 5. from nltk.tokenize import word_tokenize 6. from nltk.probability import FreqDist 7. from nltk.corpus import stopwords 8. from Sastrawi.Stemmer.StemmerFactory import StemmerFactory 9. import swifter 10. import ast 11. from sklearn.feature_extraction.text import CountVectorizer 12. from sklearn.preprocessing import normalize 13. from sklearn.model_selection import train_test_split 14. from sklearn.feature_selection import SelectPercentile, chi2 15. from sklearn.naive_bayes import MultinomialNB 16. import joblib	Penggunaan library pada program diperuntukan untuk dataframe, preprocessing, pemodelan klasifikasi, seleksi fitur, pembobotan, dan interface.

model.py

pada file ini berisikan perhitungan vector dan seleksi fitur, kemudian klasifikasi dengan model yang sudah dibuat. Beberapa tahapan pada kode sama persis dengan file pemodelan yang sudah dijelaskan diatas, sehingga tidak dijelaskan ulang. Ada beberapa tambahan saja yang akan dijelaskan.

Kode program	Penjelasan
1. from lib import* 2. model = joblib.load('multinomial_nb_10 percent_model.pkl')	Melakukan import modul lib, kemudian meload model yang sudah disimpan dengan menggunakan joblob.

```
1. def predict_text(input_text):
2.     reviews_data =
   pd.read_excel("reviews_Preproc
   essing.xlsx",
   usecols=["Label",
   "reviews_tokens_stemmed"])
3.     reviews_data.columns =
   ["label", "reviews"]
4.     def join_text_list(texts):
5.         texts =
   ast.literal_eval(texts)
6.         return ' '.join([text
   for text in texts])
7.
   reviews_data["reviews_join"] =
   reviews_data["reviews"].apply(
   join_text_list)
8.     label =
   reviews_data["label"]
9.     text =
   reviews_data["reviews_join"]
10.    train_data, test_data,
   train_labels, test_labels =
   train_test_split(text, label,
   test_size=0.2,
   random_state=42)
11.    positive_count =
   (train_labels == 1).sum()
12.    negative_count =
   (train_labels == 0).sum()
13.    total_count =
   len(train_labels)
14.    positive_ratio =
   positive_count / total_count
15.    negative_ratio =
   negative_count / total_count
16.    cvect = CountVectorizer()
17.    TF_vector_train =
   cvect.fit_transform(train_data
   )
18.    normalized_TF_vector_train
   = normalize(TF_vector_train,
   norm='l1', axis=1)
19.    TF_vector_test =
   cvect.transform(test_data)
```

Pada kodingan ini hamper sama pada file pemodelan ipynb yang sudah dijelaskan. Perbedaannya pengklasifikasian langsung menggunakan model yang sudah disimpan.

20. **normalized_TF_vector_test = normalize(TF_vector_test, norm='l1', axis=1)** 21. percent = 10 22. k = int(percent / 100 * normalized_TF_vector_train.shape[1]) 23. selector = SelectPercentile(chi2, percentile=percent) 24. tf_mat_train_selected = selector.fit_transform(normalized_TF_vector_train, train_labels) 25. **tf_mat_test_selected = selector.transform(normalized_TF_vector_test)** 26. input_vector = cvect.transform([input_text]) 27. normalized_input_vector = normalize(input_vector, norm='l1', axis=1) 28. input_vector_selected = selector.transform(normalized_input_vector) 29. prediction = model.predict(input_vector_selected)	
1. **if prediction == 0:** 2. st.write('Ulasan negatif:angry:') 3. st.image('ANGRY.png') 4. **else:** 5. st.write('Ulasan positif:smiley:') 6. **st.image('HAPPY.png')**	Hasil pengklasifikasian akan dicetak sesudah proses dilakukan

BAB III
PENUTUP

3.1 Kesimpulan

Analisis Sentimen adalah studi komputasi untuk mengklasifikasikan bertujuan untuk memahami sentimen yang terkandung dalam teks sehingga memberikan wawasan tentang bagaimana suatu produk, layanan, atau topik tertentu dipersepsikan oleh pengguna. Analisis sentimen dapat berupa ulasan negatif atau positif. Sebelum masuk ke tahapan klasifikasi perlu dilakukan tahapan preprocessing yang terdiri dari, case folding, tokenization dan cleaning, normalization, stopwords removal, dan stemming. Selanjutnya dilakukan ekstrasksi fitur TF dan seleksi fitur *Chi Square*. Hasil dari klasifikasi diukur menggunakan confusion matrix yang menghasilkan *Accuracy, Precision, Recall*, dan *F1-Score.* Program ini dibuat dalam bahasa pemrograman Python. Fitur utama yang ada pada program yang dibuat adalah halaman utama dengan tampilan identifikasi sentimen ulasan. Hasil model terbaik dengan mempertahankan jumlah fitur 100%, yaitu 1926 kata pada seleksi fitur Chi-Square yang menghasilkan nilai akurasi sebesar 92,8%, presisi sebesar 96,1%, recall sebesar 89,1%, dan F1-Score sebesar 92,4%.

3.2 Saran

Beberapa saran yang dapat diterapkan pada pembangunan program sejenis di masa mendatang adalah sebagai berikut.

1. Menambahkan tampilan tabel *confusion matrix*, hasil performa, visualisasi klasifikasi dari model pada aplikasi website.
2. Mengindentifikasi kelas dari beberapa ulasan sekaligus.

DAFTAR PUSTAKA

[1] Fidelson. Tanzil, “Elemen-Elemen Multimedia. ,” 2018. https://socs.binus.ac.id/2018/12/26/elemen-elemen-multimedia/ (accessed Jul. 05, 2023).

[2] H. Nindito, “Teori Text Mining dan Web Mining.,” 2016. https://sis.binus.ac.id/2016/12/15/teori-text-mining-dan-web-mining/ (accessed Jul. 05, 2023).

[3] Z. I. Akbar, “Apa itu Text Mining?,” 2021. https://sis.binus.ac.id/2021/04/23/apa-itu-text-mining/ (accessed Jul. 05, 2023).