

Laporan Projek Tugas Akhir Mata Kuliah
Informatika Pariwisata Kelas B

Analisis Sentimen Terhadap Objek Wisata Sunan Drajat di Lamongan Menggunakan Metode Algoritma K-Nearest Neighbors (KNN)

(200411100086) Mochammad Rizki Aji Santoso
(200411100205) Ahmad Dani Kurniawan

Abstrak

Tempat wisata merupakan salah satu tempat yang dimana sering dikunjungi orang untuk mencari kesenangan diluar aktivitas sehari-hari. Lamongan sebagai salah satu kabupaten yang ada di Jawa Timur memiliki banyak destinasi wisata, salah satunya tempat wisata religi yaitu wisata religi Sunan Drajat. Sunan Drajat merupakan salah satu dari sembilan wali di Pulau Jawa yang berdakwah menyebarkan agama Islam. Analisis sentimen diperlukan untuk mengambil keputusan terbaik. Tujuan dari penelitian untuk menganalisis ulasan pengunjung dari tempat wisata religi Sunan Drajat dari Google Maps. Metode K-Nearest Neighbor digunakan untuk melakukan klasifikasi data sentimen pada wisata religi Sunan Drajat. Klasifikasi data dibagi ke dalam sentimen positif, negatif dan netral. Penelitian ini menghasilkan sebuah analisis sentimen yang dapat mengidentifikasi dan mengklasifikasi ulasan pengunjung wisata religi Sunan Drajat dengan menggunakan algoritma K-Nearest Neighbor (KNN). Penelitian ini bermanfaat untuk evaluasi dan memperbaiki tempat wisata religi Sunan Drajat oleh petugas yang terkait agar mendapatkan citra yang lebih baik kedepannya oleh wisatawan.

Kata Kunci : Tempat Wisata, Analisis sentimen, Sunan Drajat, K-NN.

Pendahuluan

Analisis sentimen merupakan suatu bidang ilmu untuk menganalisis suatu sentimen, opini, pendapat, emosi, sikap, evaluasi, penilaian seseorang terhadap suatu produk, individu, organisasi, masalah, peristiwa ataupun topik tertentu. Selanjutnya sentimen akan diklasifikasikan menjadi tiga yaitu positif, negatif, dan netral. Untuk mengklasifikasikannya tersebut bisa menggunakan metode (KNN) yang merupakan sebuah metode machine learning yang dapat melakukan klasifikasi terhadap objek berdasarkan dataset atau data latih berdasarkan jarak paling dekat dengan objek tersebut.

Google maps adalah aplikasi panduan berbasis internet gratis dari Google. Google Maps dapat diakses melalui browser internet atau melalui ponsel. Anda dapat menggunakan Google maps untuk mengetahui suatu area yang tidak diketahui sebelumnya. Penelitian ini bertujuan untuk menganalisis ulasan wisatawan yang sudah berkunjung ke Makam Sunan Drajat di Lamongan untuk berbagi pengalaman dengan cara memberikan ulasan di Google Maps. Penelitian ini menggunakan data yang bersumber dari ulasan wisatawan terhadap objek wisata religi Sunan Drajat.

Dari sekian banyak pengunjung yang datang, tidak sedikit dari mereka meninggalkan kesan-kesan berupa review atau ulasan terhadap tempat yang telah dikunjunginya. Ulasan ini bisa menjadi informasi yang sangat penting bagi pengelola wisata maupun bagi wisatawan lainnya. Bagi pengelola wisata informasi ini sebagai bahan evaluasi dan perbaikan terhadap kekurangan yang ada guna meningkatkan kualitas tempat wisata. Sedangkan bagi wisatawan lain informasi ini bisa bermanfaat bagi mereka yang dijadikan sebagai bahan pertimbangan sebelum memutuskan untuk berkunjung ke tempat wisata tersebut.

Namun banyaknya jumlah ulasan dan beragamnya ulasan yang ada membuat sulit untuk menyimpulkan isi dari ulasan-ulasan tersebut karena harus membaca satu per satu yang tentunya akan membutuhkan waktu yang cukup lama.

Metode Usulan

K-Nearest Neighbors (KNN) adalah metode yang digunakan dalam machine learning untuk klasifikasi dan regresi. Metode KNN berbasis pada konsep bahwa data yang serupa cenderung berada dalam tetangga terdekat di dalam ruang fitur. Algoritma KNN bekerja dengan cara mengukur jarak antara data baru yang akan diklasifikasikan dengan data latih yang sudah ada. Metrik jarak yang umum digunakan adalah Euclidean Distance atau Manhattan Distance. Data baru kemudian dikelompokkan ke dalam kategori yang paling dominan di antara tetangga terdekatnya.

KNN memiliki beberapa kelebihan, antara lain sederhana dalam implementasi, tidak memerlukan asumsi tertentu tentang data, serta mampu menangani data non-linear dan masalah multikelas. Namun, KNN juga memiliki kelemahan seperti sensitif terhadap data pencilan (outlier) dan perlu memilih nilai K yang optimal. KNN dapat diterapkan dalam berbagai masalah seperti klasifikasi teks, analisis sentimen, pengenalan pola, dan lain sebagainya.

Dalam konteks sentimen analisis, K-Nearest Neighbors (KNN) adalah metode yang dapat digunakan untuk mengklasifikasikan sentimen atau pendapat pada teks berdasarkan klasifikasi dari tetangga terdekat.

Dalam analisis sentimen, data input biasanya berupa teks atau ulasan yang ingin diklasifikasikan menjadi kategori sentimen positif, negatif, atau netral. Metode KNN dapat digunakan untuk memprediksi sentimen dari data baru berdasarkan klasifikasi sentimen data latih yang sudah ada. Berikut adalah langkah-langkah umum dalam menggunakan KNN untuk sentimen analisis:

1. Persiapan Data:
 - Membaca data ulasan atau teks yang akan digunakan untuk sentimen analisis.
 - Memisahkan data menjadi fitur (teks ulasan) dan target (label sentimen).
2. Pra-pemrosesan Teks:
 - Melakukan pra-pemrosesan pada teks ulasan, seperti membersihkan data dari karakter khusus, tanda baca, atau tautan yang tidak relevan.
 - Melakukan stemming atau lemmatisasi untuk mengubah kata-kata menjadi bentuk dasarnya.
 - Menghapus kata-kata yang tidak relevan atau stopwords.
3. Pembangunan Fitur:
 - Membangun representasi numerik untuk teks ulasan menggunakan metode ekstraksi fitur seperti TF-IDF atau penghitungan kata.
 - Memisahkan data menjadi data latih dan data uji untuk evaluasi model.
4. Pelatihan Model KNN:
 - Menginisialisasi dan melatih model KNN dengan menggunakan data latih yang telah diproses.
 - Menentukan jumlah tetangga terdekat (K) yang optimal untuk model KNN.
5. Klasifikasi Sentimen:
 - Menggunakan model KNN yang telah dilatih untuk melakukan klasifikasi sentimen pada data uji atau data baru.
 - Memperoleh label sentimen dengan mengambil mayoritas label dari tetangga terdekat.
6. Evaluasi Model:
 - Menggunakan data uji yang tidak digunakan selama pelatihan untuk mengevaluasi performa model.
 - Menghitung metrik evaluasi seperti akurasi, presisi, recall, atau F1-score untuk mengukur kinerja model dalam klasifikasi sentimen.

Dengan menggunakan metode KNN dalam sentimen analisis, kita dapat mengklasifikasikan teks atau ulasan ke dalam kategori sentimen yang relevan, sehingga dapat membantu dalam pemahaman dan analisis lebih lanjut terhadap data sentimen.

Beberapa Penelitian Terkait yaitu dengan judul "Sentiment Analysis of Hotel Reviews using K-Nearest Neighbor Algorithm" (2020) oleh D. Dharavath dan A. Kumar: Penelitian ini melakukan analisis sentimen pada ulasan hotel menggunakan metode KNN. Data diambil dari dataset Tripadvisor yang terdiri dari 20.000 ulasan hotel. Hasil penelitian menunjukkan bahwa KNN memberikan hasil yang baik dalam klasifikasi sentimen pada ulasan hotel.

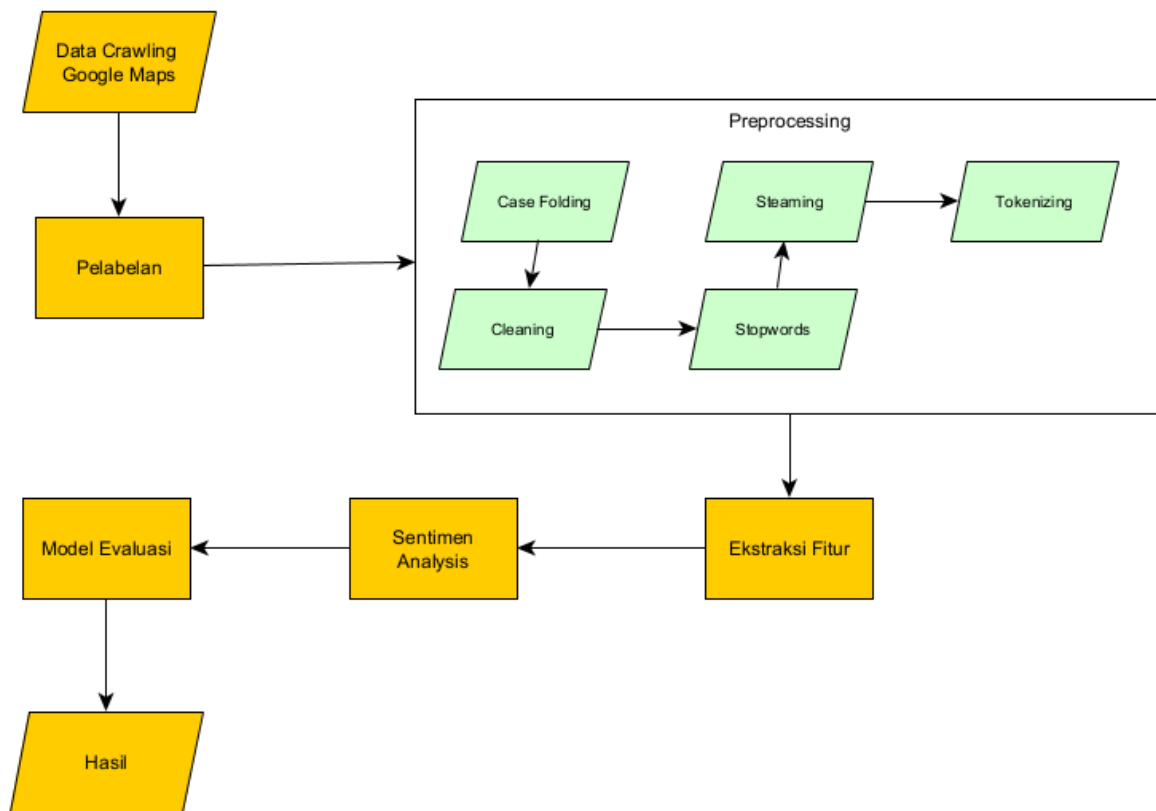
Dataset

Data yang akan digunakan yaitu ulasan dari pengunjung di Makam Sunan Drajat Lamongan, sumber data diambil dari google maps pada kolom komentar / ulasan, jumlah data yang akan diambil sebanyak 100 ulasan, waktu periode data yang akan diambil dari yang terbaru.

Contoh 5 data:

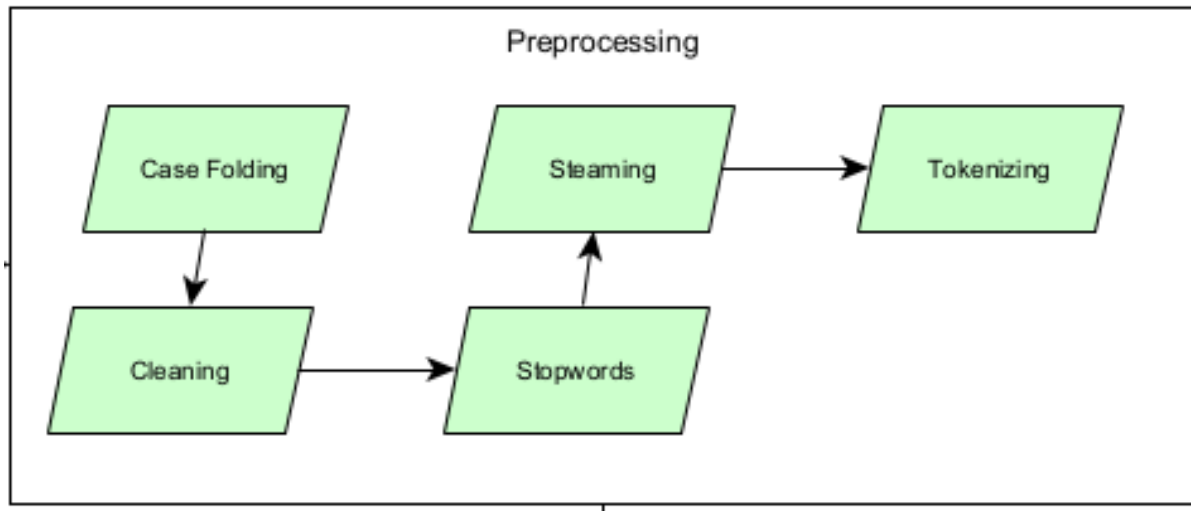
Ulasan	Rating
Akan lebih nyaman jika kebersihan di sekitar lorong jalan masuk lebih terjaga sebagaimana di lingkungan sekitar makam	4
Oke sip !!!	5
Tidak ada	5
Pengemisnya suka memaki-maki orang yang memberi uang.	1
Airnya sejuk & segar	5

Arsitektur Sistem



- Data Ulasan: Pertama-tama, data ulasan Google Maps dicrawling dengan melalui web out.scraper.com. Lalu disimpan didalam bentuk CSV dan di upload melalui repository github dan dipanggil melalui link raw pada github..
- Text Preprocessing: Selanjutnya, data teks dari ulasan Google Maps diproses menggunakan teknik preprocessing seperti menghapus case folding, cleaning, stopword, stemming, untuk menghasilkan data teks yang lebih bersih dan terstruktur.
- Feature Extraction: Setelah data teks telah diproses, fitur-fitur dari data teks diekstraksi menggunakan teknik seperti Term Frequency-Inverse Document Frequency (TF-IDF) atau Word2Vec.
- Sentiment Analysis: KNN kemudian diterapkan pada fitur-fitur dari data teks untuk mengklasifikasikan sentimen dari setiap ulasan Google Maps. Proses ini melibatkan perhitungan jarak antara setiap fitur dan kelas yang diberikan oleh tetangga terdekat dari data pelatihan. Data uji kemudian diklasifikasikan ke kelas yang paling sering muncul di antara K tetangga terdekat.
- Model Evaluation: Untuk mengevaluasi kinerja model, data diklasifikasikan dibagi menjadi data pelatihan dan data pengujian. Kemudian model dievaluasi dengan menghitung akurasi, presisi, recall, dan F1-score.
- Hasil: Setelah model dinyatakan berhasil, model dapat diterapkan pada data teks baru untuk melakukan analisis sentimen pada ulasan Google Maps. Data baru kemudian diproses melalui proses yang sama seperti pada data pelatihan dan data uji.

Preprocessing / Proses Bisnis



Tahapan Preprocessing:

a) Case folding

merupakan salah satu tahapan dalam preprocessing pada analisis sentimen yang melibatkan pengubahan semua huruf dalam teks menjadi huruf kecil atau huruf besar. Tujuan dari case folding adalah untuk mengurangi variasi dalam teks dan mengabaikan perbedaan kasus yang tidak relevan dalam analisis sentimen.

Dalam banyak kasus, case folding dilakukan dengan mengubah semua huruf dalam teks menjadi huruf kecil. Hal ini dilakukan untuk memastikan bahwa kata-kata yang sebenarnya memiliki makna yang sama, tetapi ditulis dengan huruf besar atau kecil yang berbeda, dianggap identik. Misalnya, "Suka" dan "suka" akan diubah menjadi "suka" setelah proses case folding. Dalam proses ini menggunakan library pandas.

b) Cleaning

Cleaning merupakan proses pembersihan data dari karakter yang tidak relevan seperti tanda baca, mention, character hashtag atau URL. Tujuan dari tahap ini adalah untuk membuat data lebih mudah diolah dan menghilangkan noise pada data. Cleaning text menggunakan library re dan pandas.

c) Deteksi data tidak baku (Stopwords)

Stop Words merupakan proses mengidentifikasi kata-kata dalam teks yang bukan berasal dari bahasa standar atau formal, sehingga perlu diubah atau dikembalikan ke bentuk yang lebih baku atau standar. Hal ini bertujuan untuk memudahkan pemrosesan data dan meningkatkan akurasi analisis pada tahap selanjutnya. Stopwords Menggunakan library NLTK dan pandas

d) Stemming atau Lemmatization:

Mereduksi kata-kata dalam teks menjadi bentuk dasar atau kata dasar. Stemming menghilangkan afiks (imbuhan) pada kata-kata, sedangkan lemmatization mengubah kata-kata menjadi bentuk dasar dengan menggunakan kamus kata. Hal ini dilakukan

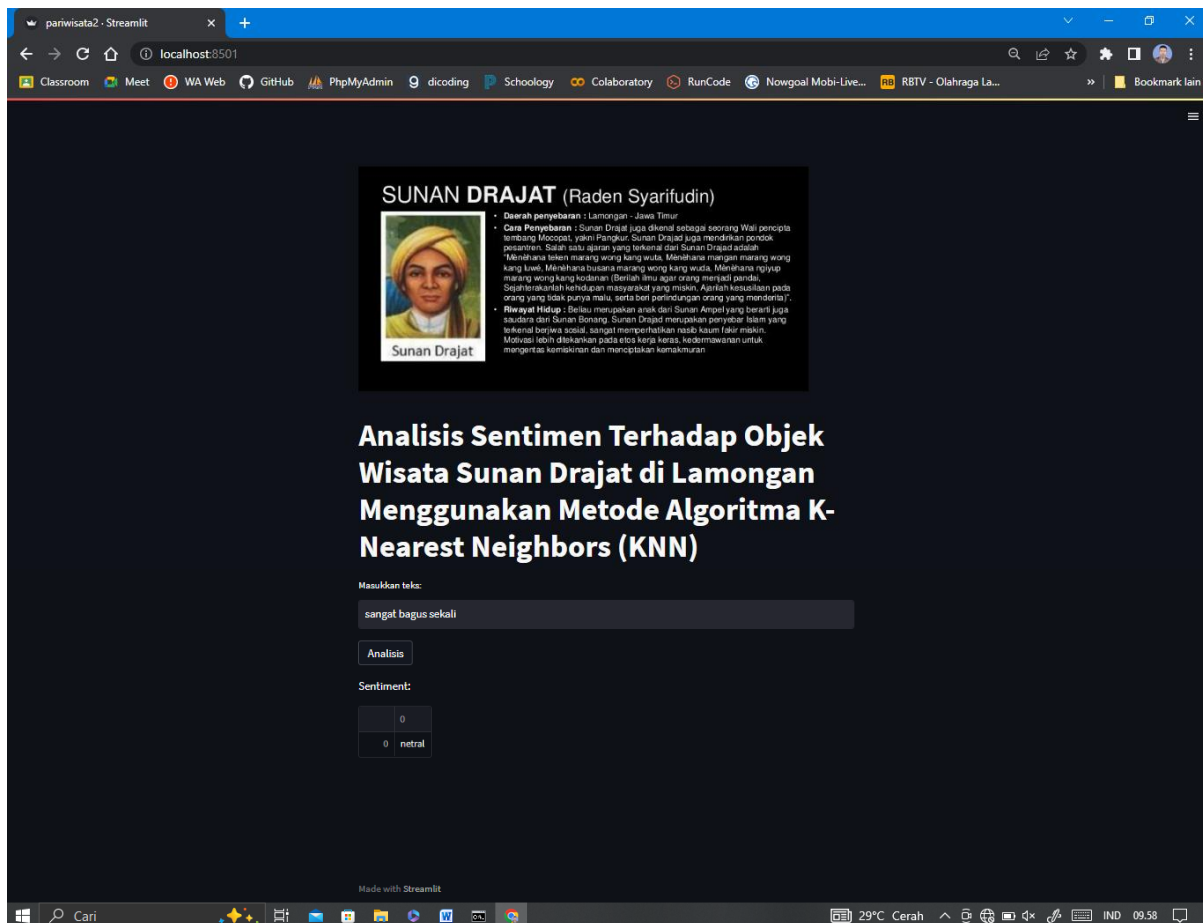
untuk mengurangi variasi kata yang memiliki akar atau makna yang sama. Library yang digunakan adalah sastrawi dan pandas.

e) Tokenizing

Memisahkan teks menjadi unit-unit yang lebih kecil yang disebut "token". Token ini bisa berupa kata, frasa, atau bahkan karakter tergantung pada pendekatan yang digunakan. Tujuan dari tokenisasi adalah untuk memecah teks menjadi unit-unit terpisah sehingga dapat diolah lebih lanjut. Library yang digunakan adalah sastrawi dan pandas.

Hasil dan pembahasan

Screenshot UI:



Keterangan:

penjelasan dari baris code, itu sudah diberi saat pertama kali membuat program supaya memudahkan ketika menjelaskan

Implementasi dari metode KNN:

1. Import library

Pertama kali yaitu import library pada baris ke 1 import library “pandas” untuk membuat tabel data frame, pada baris ke 2 import library “re” untuk cleaning pembersihan data dari karakter yang tidak relevan seperti tanda baca, mention, character hashtag atau URL. Baris ke 3 import library pickle untuk menyimpan hasil modelling.

```
1) import pandas as pd
2) import re
3) import pickle
```

2. Import file CSV yang sudah di crawling dari server Github

Pada baris ke 1 mengimport dataset yang sudah di crawling sebelumnya tersimpan di github.

```
1) df = pd.read_csv
    ('https://raw.githubusercontent.com/rizkyluxszerr/IFpariwisata/main/ulasan.csv')
```

3. Lalu data di beri label dengan menggunakan library textblob agar bisa diketahui ulasan itu bersifat positif, negatif atau netral.

```
1) from textblob import TextBlob
2) for index, row in df.iterrows():
3)     review = row['review_text']
4)         #Melakukan analisis sentimen menggunakan TextBlob
5)         analysis = TextBlob(review)
6)         sentiment = analysis.sentiment.polarity
7)         if sentiment > 0:
8)             label = 'positif'
9)         elif sentiment < 0:
10)            label = 'negatif'
11)        else:
12)            label = 'netral'
13)        df.at[index, 'Label'] = label
```

Penjelasan :

- Baris ke 1 berfungsi untuk memanggil atau menggunakan library textblob.
- Dari baris ke 2 hingga baris ke 13, merupakan berfungsi untuk melakukan pelabelan pada ulasan yang bersifat positif, negatif atau netral.

4. Setelah diberi label data kemudian menampilkan beberapa kolom saja dengan fungsi df library pandas

```
1) df[['author_title',      'review_text',      'review_rating',
      'Label']]
    # Memilih kolom yang ingin ditampilkan
```


	author_title	review_text	review_rating	Label
0	Radiya "Ribogel" Proferly	Bagus dan menyenangkan	5	netral
1	UM1 4N153	lama tidak kesini semenjak corona. sekarang b...	5	netral
2	AHMAD AL HASAN AL HASAN (AHMAD)	Alhamdulillah tempat nya luas dan sedikit peng...	5	netral
3	Javanica	Sangat tenang dan adem	5	netral
4	Mommy Ika	Salah satu wisata religi di kota lamongan yang ...	5	netral
...
95	Ahmad Samsul	Saya suka sekali jarak para wali,moga2 besok ...	5	netral
96	Ragil MVC	Alhamdulillah semakin kesini semakin indah	5	netral
97	MLAJ Aweabin	Tidak bosan-bosannya, untuk mencintai beliau 🥰	5	netral
98	Fahreza Padaninya	Ngalap barokah para wali	5	netral
99	Moh Hainul	Mantap	5	netral

100 rows x 4 columns

5. Masuk ke tahap preprocessing

- Case folding: untuk mengubah semua huruf dalam teks menjadi huruf kecil, dengan fungsi `str.lower()` lalu menampilkan hasil case folding dengan fungsi `pd.DataFrame` dan `data_lower_case` untuk memanggil dataframe.

```
#Insialisai variable yang akan dilakukan case folding dengan
fungsi lower()
1) lower_case_komentar = df['review_text'].str.lower()

#Menampilkan data yang telah dilakukan case folding
2) data_lower_case = pd.DataFrame(lower_case_komentar)
3) data_lower_case
```

Penjelasan :

- Baris ke-1, untuk melakukan Insialisasi variable yang akan dilakukan case folding.
- Baris ke-2 dan Baris ke-3, berfungsi untuk menampilkan data yang telah dilakukan pada case folding.

	review_text
0	bagus dan menyenangkan
1	lama tidak kesini semenjak corona. sekarang b...
2	alhamdulillah tempat nya luas dan sedikit peng...
3	sangat tenang dan adem
4	salah satu wisata religi di kota lamongan yang ...
...	...
95	saya suka sekali jarak para wali,moga2 besok ...
96	alhamdulillah semakin kesini semakin indah
97	tidak bosan-bosannya, untuk mencintai beliau 🥰
98	ngalap barokah para wali
99	mantap

100 rows x 1 columns

- Cleaning: pembersihan data dari karakter yang tidak relevan seperti tanda baca, mention, character hashtag atau URL. Dengan menginsialisasi `dataset_clean = []` lalu melakukan perulangan dari data case folding dengan menggunakan library 're' untuk menghapus mention, hastag, url link, dan character. Setelah itu memasukkan hasil clean kedalam array kosong dan menampilkan ke dalam bentuk dataframe.

```
#Insialisasi dataset clean
1) clean =[]

#Melakukan perulangan sepanjang data case folding
2) for i in range (len(lower_case_komentar)):

#clenasing mention
3) clean_tag = re.sub("@[A-Za-z0-9_]+","",
```

```

        lower_case_komentar[i])

#clenasing hashtag
4) clean_hashtag = re.sub("#[A-Za-z0-9_]+","", clean_tag)

#cleansing url link
5) clean_https = re.sub(r'http\S+', '', clean_hashtag)

#cleansing character
6) clean_symbols = re.sub("[^a-zA-Zi ]+","", clean_https)

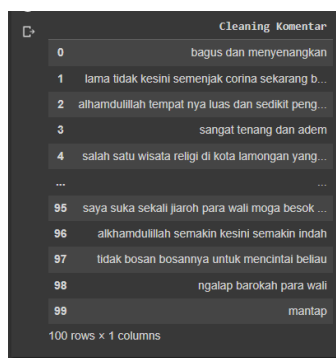
#Memasukkan hasil clean kedalam array kosong yang di
inisialisasi sebelumnya
7) clean.append(clean_symbols)

#Menampilkan ke dalam bentuk dataframe
8) clean_result =
    pd.DataFrame(clean,columns=['Cleaning_Komentar'])
9) clean_result

```

Penjelasan :

- Baris ke-1, berfungsi untuk melakukan insialisasi pada dataset yang sudah di clean atau dibersihkan.
- Baris ke-2, berfungsi untuk melakukan perulangan sepanjang data yang sudah di case folding.
- Baris ke-3, berfungsi untuk melakukan pembersihan ulasan dari mention atau tag akun.
- Baris ke-4, berfungsi untuk melakukan pembersihan ulasan dari hastag.
- Baris ke-5, berfungsi untuk melakukan pembersihan ulasan dari url link.
- Baris ke-6, berfungsi untuk melakukan pembersihan ulasan dari karakter yang tidak jelas.
- Baris ke-7, berfungsi untuk melakukan atau memasukkan hasil clean kedalam sebuah array kosong yang sudah di inisialisasi sebelumnya.
- Baris ke-8 dan Baris ke-9, berfungsi untuk menampilkan hasil ke dalam dataframe atau table.



	Cleaning Komentar
0	bagus dan menyenangkan
1	lama tidak kesini semenjak corina sekarang b...
2	alhamdulillah tempat nya luas dan sedikit peng...
3	sangat tenang dan adem
4	salah satu wisata religi di kota lamongan yang ...
...	...
95	saya suka sekali jiaroh para wali moga besok ...
96	alhamdulillah semakin kesini semakin indah
97	tidak bosan bosannya untuk mencintai beliau
98	ngalap barokah para wali
99	mantap

100 rows x 1 columns

- Deteksi kata tidak baku (Stopwords): mengidentifikasi kata-kata dalam teks yang bukan berasal dari bahasa standar atau formal, untuk diubah atau dikembalikan ke bentuk yang lebih baku atau standar.

Mengimport NLTK untuk memproses dan menganalisis teks dalam bahasa alami.

```
1) import nltk
2) nltk.download('stopwords')
3) nltk.download('punkt')
4) from nltk.tokenize import sent_tokenize, word_tokenize
5) from nltk.corpus import stopwords
```

Penjelasan :

- Baris ke-1, berfungsi untuk mengimport NLTK untuk memproses dan menganalisis teks dalam bahasa alami.
- Baris ke-2 dan ke-3, berfungsi untuk mengunduh fungsi dari NTLK yaitu stopwords dan punkt.
- Baris ke-4 dan ke-5, berfungsi untuk mengimpor modul dan fungsi yang diperlukan dari library NLTK.

Membuat kamus slang words dan kata Indonesia yang benar (contoh)

```
1) slang_dict = {"abis": "habis", "ad": "ada", "adlh": "adalah", "ahaha": "haha", "aj": "saja"}
```

Penjelasan :

- Baris ke-1 untuk membuat kamus slang words dan kata Indonesia yang benar menurut KBBI.

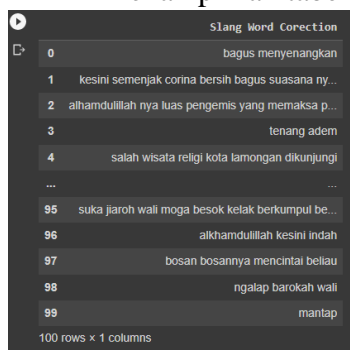
```
# Membuat fungsi untuk mengubah slang words menjadi kata
Indonesia yang benar,
1) def replace_slang_words(text):
2)     words = nltk.word_tokenize(text.lower())
3)     words_filtered = [word for word in words if word not in
        stopwords.words('indonesian')]
4)     for i in range(len(words_filtered)):
5)         if words_filtered[i] in slang_dict:
6)             words_filtered[i] = slang_dict[words_filtered[i]]
7)     return ' '.join(words_filtered)

# Memasukan Kata yang telah di clean ke dalam fungsi deteksi
slang words
8) slang_words=[]
9) for i in range(len(clean)):
10)     slang = replace_slang_words(clean[i])
11)     slang_words.append(slang)
12) data_slang = pd.DataFrame(slang_words, columns=["Slang
        Word Corection"])
13) data_slang
```

Penjelasan :

- Baris ke-1, berfungsi untuk menggantikan kata-kata slang dalam teks dengan bentuk standar atau bentuk yang umum.
- Baris ke-2, berfungsi untuk menampung daftar kata-kata dalam teks yang telah di tokenisasi dan dikonversi menjadi huruf kecil.

- Baris ke-3, berfungsi untuk memfilter kata-kata yang tidak termasuk dalam daftar stopwords dalam kamus besar bahasa Indonesia (KBBI).
- Baris ke-4, berfungsi untuk menginisialisasi perulangan pada setiap kata.
- Baris ke-5 dan ke-6, berfungsi memeriksa jika kata yang sedang diperiksa terdapat dalam kamus slang yang sudah dibuat, maka akan menggantikan kata slang itu dengan nilai yang sesuai dalam kamus.
- Baris ke-7, berfungsi untuk menggabungkan kata-kata yang sudah selesai menjadi satu string yang dipisahkan oleh spasi.
- Baris ke-8, berfungsi untuk menyimpan hasil penggantian kata-kata slang.
- Baris ke-9, berfungsi untuk melakukan iterasi pada setiap elemen yang diasumsikan sebagai daftar teks yang akan diproses.
- Baris ke-10 dan ke-11, berfungsi untuk menggantikan kata slang dan menambahkan hasil penggantian kata slang ke dalam list menggunakan metode append.
- Baris ke-12 dan ke-13, berfungsi untuk membuat tabel dan menampilkan tabel dari hasil yang didapatkan.



	Slang Word Corection
0	bagus menyenangkan
1	kesini semenjak corina bersih bagus suasana ny...
2	alhamdulillah nya luas pengemis yang memaksa p...
3	tenang adem
4	salah wisata religi kota lamongan dikunjungi
...	...
95	suka jarang wali moga besok kelak berkumpul be...
96	alkhamdulillah kesini indah
97	bosan bosannya mencintai beliau
98	ngalap barokah wali
99	mantap
100 rows x 1 columns	

- Stemming: Mereduksi kata-kata dalam teks menjadi bentuk dasar atau kata dasar.

Mengimport library sastrawi untuk pemrosesan teks dalam bahasa Indonesia, termasuk tokenisasi, stemming

```
1) pip install Sastrawi
```

```
#Import Library Sastrawi
1) from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

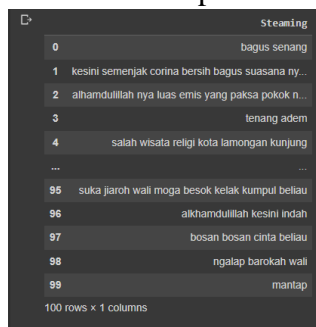
#Membuat fungsi steaming dengan library
2) factory = StemmerFactory()
3) steaming = factory.create_stemmer()

#Melakukan perulangan untuk memasukan kata kedalam fungsi
steaming
4) hasil_steaming = []
5) for i in range (len(slang_words)):
6)     stem = steaming.stem(slang_words[i])
7)     hasil_steaming.append(stem)
```

```
#Menampilkan data hasil steaming kedalam dataframe
8) data_steaming = pd.DataFrame(hasil_steaming,
    columns=["Steaming"])
9) data_steaming
```

Penjelasan :

- Baris ke-1, berfungsi untuk mengimport salah kelas yang ada library sastrawi.
- Baris ke-2 dan ke-3, berfungsi untuk membuat objek factory dan membuat objek steaming dari objek factory yang akan menghasilkan objek stemmer yang digunakan pada stemming.
- Baris ke-4, berfungsi untuk menyimpan hasil stemming kata-kata.
- Baris ke-5 hingga ke-7, berfungsi melakukan iterasi, melakukan stemming pada slang words dan hasil yang didapat di simpan dengan metode append.
- Baris ke-8 dan ke-9, berfungsi membuat, menambahkan dan menampilkan tabel yang berisikan data hasil stemming.



	Steaming
0	bagus senang
1	kesini semenjak corina bersih bagus suasana ny...
2	alhamdulillah nya luas emis yang paksa pokok n...
3	tenang adem
4	salah wisata religi kota lamongan kunjung
...	...
95	suka jiaroh wali moga besok kelak kumpul beliau
96	alhamdulillah kesini indah
97	bosan bosan cinta beliau
98	ngalap barokah wali
99	mantap

100 rows x 1 columns

- Tokenizing: memisahkan teks menjadi unit-unit yang lebih kecil bisa berupa kata, frasa, atau bahkan karakter.

```
#Import Library sastrawi stop words
1) from Sastrawi.StopWordRemover.StopWordRemoverFactory
    import StopWordRemoverFactory

#Inisialisasi fungsi stop words
2) stop_factory = StopWordRemoverFactory()
3) words = []

#Membuat perulangan untuk memasukkan dataset ke dalam
tekonisasi dan list stopwords
4) for i in range (len(hasil_steaming)):
    #Inisialisai fungsi tokenisasi dan stopword
    5) tokens = word_tokenize(hasil_steaming[i])
    6) more_stopword = ['dengan',
        'ia', 'bahwa', 'oleh', 'aalysis', 'aam', 'kunci']
    7) data = stop_factory.get_stop_words()+more_stopword
    8) stopword = stop_factory.create_stop_word_remover()
```

```
#Melakukan removed kata
9) removed = []
10) for t in tokens:
11)     if t not in data:
12)         removed.append(t)

#Memasukkan hasil removed kedalam variable words
13) words.append(removed)
14) print(removed)
```

Penjelasan :

- Baris ke-1, berfungsi untuk memanggil kelas stopwords remover yang berfungsi untuk menghapus kata-kata yang tidak penting sesuai dengan KBBI.
- Baris ke-2 dan ke-3, berfungsi membuat objek dan membuat list untuk menyimpan hasil tokenisasi dan penghapusan stop words.
- Baris ke-4 hingga ke-8, berfungsi untuk melakukan iterasi, melakukan tokenisasi pada hasil stemming dan disimpan pada list, membuat list untuk menambahkan kata yang ingin ditambahkan ke daftar stopwords, untuk menggabungkan daftar stop words asli dengan stopwords terbaru untuk mendapatkan daftar stop words lengkap yang akan digunakan dalam penghapusan dan membuat objek stopwords yang akan digunakan untuk melakukan penghapusan stopwords.
- Baris ke-9 dan ke-12, berfungsi untuk menyimpan hasil dari stop words, untuk memeriksa kondisi jika token ada suatu kata tidak ada dalam daftar stop words maka akan di tambahkan ke dalam list menggunakan metode append.
- Baris ke-13 dan ke-14, berfungsi untuk menambahkan hasil ke dalam list dan mencetak kata-kata setelah penghapusan stop words.

```
[ 'bagus', 'sungguh',
  'kesini', 'sejarah', 'corina', 'beraih', 'bagus', 'suasana', 'nyaman', 'datang', 'subuh',
  'alhamdulillah', 'nya', 'luas', 'emis', 'paka', 'pokok', 'nyaman', 'tempat', 'emis', 'paka', 'maka', 'beda', 'makan', 'sunan', 'gunung', 'jati', 'cirebon',
  'tenang', 'adem',
  'salah', 'wisata', 'religi', 'kota', 'lamongan', 'kunjung',
  'wasib', 'kunjung', 'wisata', 'religi',
  'nyaman', 'beraih', 'lorong', 'jalan', 'masuk', 'jaga', 'lingkung', 'makan',
  'oke', 'sip',
  'emis', 'tuka', 'maki', 'maki', 'orang', 'tungg',
  'terjemah', 'google', 'air', 'dingin', 'segar', 'mali', 'air', 'sejak', 'segar',
  'suasana', 'tenang', 'wisata', 'religi', 'hening', 'adem', 'pagi',
  'alhamdulillah', 'makan', 'sunan', 'drajat', 'lamongan', 'rombong', 'ziarah', 'gp', 'amor', 'kalisari', 'cilongok', 'banyuwangi', 'maga', 'berkah', 'hikmah',
  'terjemah', 'google', 'oke',
  'sejarah', 'religi',
  'tenang',
  'wisata', 'religi',
  'menelusur', 'jejak', 'sejarah', 'adab', 'sebar', 'agama', 'jaga', 'arif', 'lokal',
  'alhamdulillah', 'ziarah', 'makan', 'salah', 'auliya', 'sunan', 'drajat', 'putra', 'dr', 'sunan', 'ampel', 'saudara', 'sunan', 'bonang', 'kunjung', 'musium',
  'alhamdulillah', 'kesini', 'pendak',
  'religi', 'online', 'muslim',
  'alam', 'rombong', 'samarinda', 'komplek', 'makan', 'sunan', 'drajat', 'magrib', 'subhanallah', 'pandang', 'warung', 'milit', 'mushola', 'outdoor', 'jalan',
  'sunan', 'drajat', 'salah', 'sunan', 'sebelah', 'sunan', 'mali', 'songo', 'mana', 'kecil', 'raden', 'qualis', 'gala', 'raden', 'syarifudin', 'sunan', 'drajat',
  'ziarah', 'makan', 'mali', 'songo', 'latih', 'umroh', 'mana', 'desak', 'banyaknya', 'orang', 'sungguh', 'uras', 'tenaga',
  'makan', 'sunan', 'drajat', 'dan', 'karam', 'sedia', 'nya', 'tarik', 'biaya', 'bak', 'bab', 'mandi', 'tarik', 'biaya', 'mabon', 'mas', 'dilan', 'nya', 'kenan',
  'mapi', 'murtat', 'lokasi', 'muda', 'dijungka', 'beraih', 'nyaman', 'masuk', 'biaya', 'parkir', 'tak', 'mobil', 'pribadi', 're', 'sepi', 'ziarah', 'yag', 'dan',
  'ramai', 'dg', 'datang', 'ziarah',
  'alhamdulillah', 'mali', 'kesini',
  'lokasi', 'parkir', 'luas', 'lokasi', 'makan',
  'sepat', 'longgar' ]
```

```
1) gabung=[]
   #Membuat perulangan untuk menggabungkan kata
2) for i in range(len(words)):
3)     joinkata = ' '.join(words[i])
4)     gabung.append(joinkata)

5) result = pd.DataFrame(gabung, columns=['JoinKata'])
6) result
```

Penjelasan :

- Baris ke-1, berfungsi untuk menyimpan kata-kata yang telah digabungkan.
- Baris ke-2 hingga ke-4, berfungsi untuk melakukan iterasi, menggabungkan kata-kata menjadi satu string yang akan dipisahkan oleh spasi dan hasilnya akan ditambahkan penggabungan kata kedalam list.
- Baris ke-5 dan ke-6, berfungsi untuk membuat table dan menampilkan tabel yang berisikan hasilnya.

0	bagus senang
1	kesini semenjak corina bersih bagus suasana ny...
2	alhamdulillah nya luas emis paksa pokok nyaman...
3	tenang adem
4	salah wisata religi kota lamongan kunjung
...	...
95	suka jiaroh wali moga besok kelak kumpul beliau
96	alhamdulillah kesini indah
97	bosan bosan cinta beliau
98	ngalap barokah wali
99	mantap

6. Setelah data di preprocessing, masuk ke Feature Extraction

```
# Import fungsi tf-idf
1) from sklearn.feature_extraction.text import TfidfVectorizer,
    CountVectorizer
```

Penjelasan :

- Baris ke-1, berfungsi mengimport fungsi yang akan digunakan untuk menghasilkan representasi vector dari teks.

```
# Inisialisasi objek TfidfVectorizer
1) tfidf_vectorizer = TfidfVectorizer()

# Proses ekstraksi fitur TF-IDF
# Metode ini akan menghitung bobot TF-IDF untuk setiap kata dalam
teks dan menghasilkan matriks TF-IDF yang mewakili teks tersebut.
# pd.DataFrame.toarray() digunakan untuk mengubah matriks sparse
menjadi matriks denser, dan columns diatur menggunakan metode
get_feature_names_out()
# dari objek TfidfVectorizer. Kolom-kolom dataframe akan menjadi
fitur-fitur TF-IDF yang diekstraksi, dan nilai-nilai dalam matriks
TF-IDF akan menjadi
# nilai-nilai dalam dataframe.
2) tfidf_matrix = tfidf_vectorizer.fit_transform(df['review_text'])

# Membuat dataframe dari matriks TF-IDF
3) tfidf_df = pd.DataFrame(tfidf_matrix.toarray(),
    columns=tfidf_vectorizer.get_feature_names_out())
4) tfidf_df
```

Penjelasan :

- Baris ke-1, berfungsi untuk menghitung bobot TF-IDF dalam setiap kata di teks.
- Baris ke-2, berfungsi untuk melakukan ekstraksi fitur TF-IDF dan menghasilkan matriks TF-IDF yang mewakili teks.
- Baris ke-3 dan ke-4, berfungsi untuk membuat tabel yang akan di simpan hasil dari Langkah sebelumnya dan akan ditampilkan berbentuk tabel.

	01	10rb	1470	2000	22	23	25	aamiin	ada	adalah	...	wudlu	ya	yag	yakni	yang	yg	yra	yyy	ziarah	ziarah
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	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	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	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	0.126959	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.236527	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	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	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	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.23986	0.000000	0.0	0.0	0.0	0.0
...
95	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0
96	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0
97	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0
98	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0
99	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0

100 rows x 550 columns

7. Modelling dengan KNN

```

1. from sklearn.feature_extraction.text import TfidfVectorizer
2. from sklearn.neighbors import KNeighborsClassifier
3. from sklearn.model_selection import train_test_split
4. from sklearn.metrics import accuracy_score

# Memisahkan fitur (tweet) dan label (sentimen) menjadi variabel X
dan y.
# Kemudian, kita membagi dataset menjadi data pelatihan dan data
pengujian menggunakan train_test_split dari scikit-learn.
5. X = tfidf_df
6. y = df['Label']

# Membagi dataset menjadi data pelatihan dan data pengujian
# test_size: Ukuran yang ingin Anda alokasikan untuk data testing,
misalnya 0.2 untuk 20% data testing dan 80% data training.
# random_state: Nilai yang digunakan untuk mengontrol pemilihan acak
saat membagi data.
# Dalam machine learning, penting untuk memastikan bahwa pemilihan
acak yang sama diaplikasikan pada setiap percobaan.
7. X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=42)

# Membuat objek k-NN classifier dengan k=5
# untuk mengambil 3 tetangga terdekat dalam k-NN dan menggunakan
metrik jarak Euclidean.
8. knn = KNeighborsClassifier(n_neighbors=3, metric='euclidean')

# Melatih model k-NN

```



```
# melatih model menggunakan data pelatihan dengan fit().
    knn.fit(X_train, y_train)

# Melakukan prediksi pada data pengujian
9. y_pred = knn.predict(X_test)

# Menghitung akurasi
10. accuracy = accuracy_score(y_test, y_pred)
11. print("Akurasi: {:.2f}%".format(accuracy * 100))
```

Penjelasan :

- Baris ke-1 hingga ke-4, berfungsi untuk mengimport kelas TF-IDF dan KNN, mengimport fungsi train test split dan accuracy score yang digunakan untuk membagi dataset menjadi data latih dan data test kemudian juga menghitung akurasi prediksi.
- Baris ke-5 dan ke-6, berfungsi untuk menetapkan variable X dengan nilai matriks TF-IDF sebelumnya dan menetapkan variable Y dengan kolom label yang berisikan sentiment yang sesuai dengan teks.
- Baris ke-7, berfungsi untuk memisahkan dataset menjadi data latih dan data test, parameter random state berfungsi untuk mengontrol atau memilih secara acak yang sama saat membagi data.
- Baris ke-8, berfungsi untuk memprediksi menggunakan KNN dengan mengambil jarak 3 tetangga terdekat dalam dataset.
- Baris ke-9, berfungsi untuk melatih model KNN menggunakan data latih.
- Baris ke-10 dan ke-11, berfungsi untuk melakukan prediksi pada data uji dengan model KNN dan menghitung akurasi prediksi dengan membandingkan label yang sebenarnya dan menampilkan prediksi.

□ Akurasi: 95.00%

Kesimpulan

pada penelitian ini, masalah yang diangkat adalah ulasan objek tempat wisata Sunan Drajat di google maps. Dalam penelitian ini, kami menggunakan metode algoritma KNN untuk mendapatkan hasil analisis, prediksi dan akurasi. Hasilnya berupa analisis dari setiap ulasan apakah ulasan tersebut berartikan positif, negatif atau netral. Setiap hasil analisis tersebut di prediksi berapa banyak ulasan yang berartikan positif, negatif atau netral, kemudian setelah diketahui baru di akurasi hasil prediksi tersebut. Hasil akurasi mendapatkan presentase sebesar 95% sama seperti hasil dari perbandingan oleh dataset pelabelan manual.

Referensi

Sentimen, A., Wisata, O., Di, B., Maps, G., Siti Utami, D., Utami, D. S., & Erfina, A. (2022). Analisis Sentimen Objek Wisata Bali Di Google Maps Menggunakan Algoritma Naive Bayes. In *Jurnal Sains Komputer & Informatika (J-SAKTI)* (Vol. 6, Issue 1).

Sari, R. (2020). Analisis Sentimen Pada Review Objek Wisata Dunia Fantasi menggunakan Algoritma K-Nearest Neighbor (K-NN). *Jurnal Sains Dan Manajemen*, 8(1).
www.tripadvisor.com.

Rifa, A., Sujaini, H., Prawira, D., & Hadari Nawawi, J. H. (n.d.). *JEPIN (Jurnal Edukasi dan Penelitian Informatika) Sentiment Analysis Objek Wisata Kalimantan Barat Pada Google Maps Menggunakan Metode Naive Bayes*.

Ginantra, N. L. W. S. R., Yanti, C. P., Prasetya, G. D., Sarasvananda, I. B. G., & Wiguna, I. K. A. G. (2022). Analisis Sentimen Ulasan Villa di Ubud Menggunakan Metode Naive Bayes, Decision Tree, dan K-NN. *Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI)*, 11(3), 205–215. <https://doi.org/10.23887/janapati.v11i3.49450>

Sibyan, H., & Hasanah, N. (n.d.). ANALISIS SENTIMEN PADA WISATA DIENG DENGAN ALGORITMA K-NEAREST NEIGHBOR (K-NN). *Jurnal Penelitian Dan Pengabdian Kepada Masyarakat UNSIQ*, 9(1), 38–47.

Karelo Hesay, I., & Adinugroho, S. (2021). *Analisis Sentimen Ulasan Pengunjung Simpang Lima Gumul Kediri menggunakan Metode BM25 dan Neighbor-Weighted K-Nearest Neighbor* (Vol. 5, Issue 7). <http://j-ptiik.ub.ac.id>

Khofifah, W., Rahayu, D. N., & Yusuf, A. M. (2022). Analisis Sentimen Menggunakan Naive Bayes Untuk Melihat Review Masyarakat Terhadap Tempat Wisata Pantai Di Kabupaten Karawang Pada Ulasan Google Maps. *Jurnal Interkom: Jurnal Publikasi Ilmiah Bidang Teknologi Informasi Dan Komunikasi*, 16(4), 28–38.
<https://doi.org/10.35969/interkom.v16i4.192>

Herlawati, H., Handayanto, R. T., Atika, P. D., Khasanah, F. N., Yusuf, A. Y. P., & Septia, D. Y. (2021). Analisis Sentimen Pada Situs Google Review dengan Naïve Bayes dan Support Vector Machine. *Jurnal Komtika (Komputasi Dan Informatika)*, 5(2), 153–163.
<https://doi.org/10.31603/komtika.v5i2.6280>

Firdaus, M. F. el, Nurfaizah, N., & Sarmini, S. (2022). Analisis Sentimen Tokopedia Pada Ulasan di Google Playstore Menggunakan Algoritma Naïve Bayes Classifier dan K-Nearest Neighbor. *JURIKOM (Jurnal Riset Komputer)*, 9(5), 1329.
<https://doi.org/10.30865/jurikom.v9i5.4774>

Iwandini, I., Triayudi, A., & Soepriyono, G. (2023). Analisa Sentimen Pengguna Transportasi Jakarta Terhadap Transjakarta Menggunakan Metode Naives Bayes dan K-Nearest Neighbor. *Journal of Information System Research (JOSH)*, 4(2), 543–550.
<https://doi.org/10.47065/josh.v4i2.2937>

Fikri, M., Syahbani, N., & Ramadhan, G. (2023). *JURNAL MEDIA INFORMATIKA BUDIDARMA Klasifikasi Gerakan Yoga dengan Model Convolutional Neural Network Menggunakan Framework Streamlit*. <https://doi.org/10.30865/mib.v7i1.5520>

Putri, T. A. Q., Triayudi, A., & Aldisa, R. T. (2023). Implementasi Algoritma Decision Tree dan Naïve Bayes Untuk Klasifikasi Sentimen Terhadap Kepuasan Pelanggan Starbucks. *Journal of Information System Research (JOSH)*, 4(2), 641–649. <https://doi.org/10.47065/josh.v4i2.2949>

Sonia Shabrilianti, S., Triayudi, A., & Avrilia Lantana, D. (2023). Analisis Klasifikasi Performace KPI Salesman Menggunakan Metode Decision Tree Dan Naïve Bayes. *Jurnal Riset Komputer*, 10(1), 2407–389. <https://doi.org/10.30865/jurikom.v10i1.5628>

Karelo Hesay, I., & Adinugroho, S. (2021). *Analisis Sentimen Ulasan Pengunjung Simbang Lima Gumul Kediri menggunakan Metode BM25 dan Neighbor-Weighted K-Nearest Neighbor* (Vol. 5, Issue 7). <http://j-ptiik.ub.ac.id>

Setiawan, R., & Triayudi, A. (2022). Klasifikasi Status Gizi Balita Menggunakan Naïve Bayes dan K-Nearest Neighbor Berbasis Web. *JURNAL MEDIA INFORMATIKA BUDIDARMA*, 6(2), 777. <https://doi.org/10.30865/mib.v6i2.3566>