

**SENTIMEN ANALISIS PILPRES 2024 PADA MEDIA SOSIAL
TWITTER MENGGUNAKAN *NAÏVE BAYES CLASSIFIER***

SKRIPSI

**Karya tulis sebagai salah satu syarat
untuk memperoleh gelar Tingkat Sarjana**

Oleh

**MUCHAMMAD FAHD ISHAMUDDIN
NPM : 411550050180048**



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS LANGLANGBUANA
2023**

LEMBAR PENGESAHAN

SENTIMEN ANALISIS PILPRES 2024 PADA MEDIA SOSIAL TWITTER MENGGUNAKAN *NAÏVE BAYES CLASSIFIER*

Oleh

**MUCHAMMAD FAHD ISHAMUDDIN
NPM : 411550050180048**

**untuk memperoleh gelar Tingkat Sarjana dari
Program Studi Teknik Informatika
Universitas Langlangbuana**

Menyetujui

Pembimbing 1

Pembimbing 2

(Arief Ginanjar, S.T., M. Kom.) (Wahyu Purnama sari, S.Kom., M.T.)
NIDN: 0423107805 NIDN: 0401028001

Mengetahui

Dekan
Fakultas Teknik

Ketua Program Studi
Teknik Informatika

(Dr. Hj. Hennie Husniah, Dra., M.T.) (Kusmaya, S. S.Kom., M.T)
NIDN: 0024066801 NIDN: 0423096501

LEMBAR PERNYATAAN KEASLIAN TUGAS AKHIR

Saya yang bertanda tangan dibawah ini :

Nama : MUCHAMMAD FAHD ISHAMUDDIN

NPM : 411550050180048

Judul Tugas Akhir : SENTIMEN ANALISIS PILPRES 2024 PADA MEDIA SOSIAL TWITTER MENGGUNAKAN *NAÏVE BAYES CLASSIFIER*

Menyatakan dengan sebenarnya bahwa penulisan Tugas Akhir ini berdasarkan hasil penelitian, pemikiran dan pemaparan asli dari saya sendiri, baik untuk naskah laporan maupun kegiatan pembangunan aplikasi yang tercantum sebagai bagian dari Tugas Akhir ini. Jika terdapat karya orang lain, saya akan mencantumkan sumber yang jelas.

Demikian pernyataan ini saya buat dengan sesungguhnya dan apabila dikemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya tulis ini dan sanksi lain sesuai dengan peraturan yang berlaku di Universitas Langlangbuana. Demikian pernyataan ini saya buat dalam keadaan sadar tanpa paksaan dari pihak manapun.

Bandung, 18 Juni 2023
Yang membuat pernyataan,

MUCHAMMAD FAHD ISHAMUDDIN

NPM : 411550050180048

"Maka sesungguhnya bersama kesulitan ada kemudahan, sesungguhnya bersama kesulitan ada kemudahan"

Qs. Al-Insyirah 5-6

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat Allah Subhanahu Wa Ta'ala yang telah memberikan rahmat dan karunianya, sholawat serta salam kepada nabi akhir zaman, yakni Nabi Muhammad Sallahu Alaihi Wassalam. Laporan tugas akhir ini merupakan salah satu syarat untuk menyelesaikan program studi Teknik Informatika jenjang Strata-1 di Universitas Langlangbuana.

Penulis menyadari bahwa tidak lepas dari peran berbagai pihak. Ucapan terimakasih dan penghargaan yang sebesar-besarnya atas bantuan, nasehat, bimbingan dan dukungan serta motivasi. Dalam kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Bapak Dr. HR. AR. Harry Anwar, S.H., M.H selaku Rektor Universitas Langlangbuana
2. Ibu DR. Hennie Husniah DRA., M.T. selaku Dekan Fakultas Teknik Universitas langlangbuana yang telah memeberikan fasilitas dan prasarana pendidikan sehingga terlaksana laporan ini.
3. Bapak Kusmaya, S.Kom., M.T. selaku Ketua Program Studi Teknik Informatika Universitas Langlangbuana yang telah memberikan fasilitas dan prasarana pendidikan sehingga terlaksana laporan ini.
4. Bapak Arief Ginanjar, S.T., M.Kom. sebagai pembimbing I yang telah membimbing serta memberikan motivasi kepada penulis sehingga dapat menyelesaikan laporan Skripsi ini.
5. Ibu Wahyu Purnama Sari, S.Kom., M.T. sebagai dosen pembimbing II yang telah membimbing serta memberikan motivasi kepada penulis sehingga dapat menyelesaikan laporan Skripsi ini.
6. Bapak Hadi Prasetyo Utomo, S.Kom., M.T., Ph.D. selaku Wali Dosen.
7. Kepada Orang tua penulis, yakni Sugiyanto Prasetyolaksono, S.Si dan Rr Masayu Diah Ratna N, S.E. Yang telah mendukung secara materil maupun imateril dalam menjalani pendidikan jenjang Strata-1 sejak awal masuk hingga terciptanya Skripsi ini.

8. Kepada saudara Firman Cahyadi, Dzakiy Muhammad Urwah dan Mohammad Rofiat Darajat yang senantiasa mendukung secara moril serta mendoakan.
9. Seluruh Staff dosen Program Studi Teknik Informatika Universitas Langlangbuana yang banyak memberikan pengetahuan dan pengajaran pada penulis.
10. Semua Staff dan Karyawan Tata Usaha Fakultas Teknik Universitas Langlangbuana yang membantu administrasi perkuliahan.
11. Semua rekan kelas Informatika-A1 yang telah memberikan kenangan dalam menempuh pendidikan.
12. Kepada rekan alumni SDIT At-taqwim yang sudah menjadikan ruang cerita dan memotivasi penulis.
13. Kepada semua pihak yang penulis tidak dapat lagi sebutkan satu per-satu.

Penulis menyadari bahwa dalam penyusunan Skripsi ini banyak kekurangannya. Oleh karena itu penulis mengharapkan saran dan kritik yang dapat menyempurnakan penulisan ini sehingga dapat bermanfaat dan berguna bagi pengembangan ilmu pengetahuan. Aamiin Ya Robbal Aalamin.

Bandung, 18 Juni 2023

Penyusun

Muchammad Fahd Ishamuddin

ABSTRAK

Pemilihan Umum (PEMILU) merupakan pesta demokrasi rakyat yang diselenggarakan 5 tahun sekali, pada pesta rakyat tersebut salah satunya ada Pemilihan Presiden. PEMILU 2024 sangatlah istimewa yakni terjadi pada era modern dan banyaknya pemilih pemula yang menjadi peserta pada pesta demokrasi tersebut. Data pada tahun 2022 menyatakan bahwa 191 juta jiwa masyarakat Indonesia sudah aktif dalam ber media sosial, seperti yang kita ketahui bahwa *spotlight* seluruh media sekarang berpindah menjadi media digital. media digital pada saat ini menjadi tempat masif-nya para juru kampanye untuk mendongkrak elektabilitas calon yang diusung oleh partainya, mulai dari Prabowo Subianto, Anies Baswedan, Ridwan Kamil dan Ganjar Pranowo. Hal tersebut memantik penulis untuk menganalisa nilai sentiment pada tweet yang terkumpul sejak awal hingga akhir 2022, penulis menganalisa menggunakan model machine learning naive bayes classifier dan melakukan penulisan pada python notebook, model mendapatkan akurasi sebesar 71,32%. Setelah melakukan modeling penulis mengevaluasi dengan monte-carlo cross validation mendapatkan nilai rata rata dari 150 iterasi yakni 71,32% dan pengaplikasian web sederhana dengan library streamlit.

Kata Kunci: *Machine Learning, Naïve bayes classifier, Pilpres 2024, Prabowo, Ganjar Pranowo, Anies Baswedan, streamlit, monte-carlo cross validation, pyhton notebook.*

ABSTRACT

The General Election (PEMILU) is a people's democratic party which is held every 5 years, one of which is the Presidential Election. The 2024 ELECTION is very special, because it took place in the modern era and many first-time voters participated in this democratic party. Data for 2022 states that 191 million Indonesian people are already active on social media, as we know that the spotlight of all media has now shifted to digital media. Digital media is currently a massive place for campaigners to boost the electability of candidates promoted by their parties, starting from Prabowo Subianto, Anies Baswedan, Ridwan Kamil and Ganjar Pranowo. This sparked the author to analyze the sentiment value in the tweets collected from the beginning to the end of 2022, the author analyzed using the naive Bayes classifier machine learning model and wrote on a python notebook, the model obtained an accuracy of 71.32%. After doing the modeling, the authors evaluate it with Monte-Carlo cross validation, getting an average value of 150 iterations, namely 71.32% and a simple web application with the Streamlit library.

Keywords: *Machine Learning, Naïve bayes classifier, Pilpres 2024, Prabowo, Ganjar Pranowo, Anies Baswedan, streamlit, monte-carlo cross validation, pyhton notebook.*

DAFTAR ISI

LEMBAR PENGESAHAN	ii
LEMBAR PERNYATAAN KEASLIAN TUGAS AKHIR	iii
KATA PENGANTAR	v
ABSTRAK	vii
ABSTRACT	viii
DAFTAR ISI	ix
DAFTAR GAMBAR	xi
DAFTAR TABEL.....	xiv
DAFTAR SINGKATAN DAN ISTILAH	xv
DAFTAR LAMPIRAN.....	xvi
BAB I PENDAHULUAN	I-1
I.1 Latar Belakang.....	I-1
I.2 Rumusan Masalah.....	I-3
I.3 Batasan Masalah.....	I-3
I.4 Tujuan Penelitian.....	I-3
I.5 Keluaran Penelitian.....	I-3
I.6 Sistematika Penulisan	I-4
BAB II LANDASAN TEORI	II-1
II.1 Teori Terkait Permasalahan.....	II-1
II.1.1 <i>Sentiment Analysis</i>	II-1
II.1.2 <i>Naïve Bayes Classifier</i>	II-1
II.1.3 <i>Machine Learning</i>	II-2
II.2 Teori Pendukung.....	II-3
II.2.1 Python	II-3
II.2.2 <i>Text Mining</i>	II-3
II.2.3 <i>Jupyter Notebook</i>	II-4
II.2.4 <i>Preprocessing</i>	II-4
II.2.5 <i>Bag of Words</i>	II-8
II.2.6 <i>Monte Carlo Cross Validation</i>	II-9
II.2.7 <i>Confusion Matrix</i>	II-10

II.2.8	<i>Cross Industry Standard Process for Data Mining (CRISP-DM)</i> ...	II-11
II.3	Penelitian-penelitian Terdahulu.....	II-13
II.3.1	Implementasi Metode <i>Naïve Bayes</i> untuk Analisis Sentimen Warga Jakarta Terhadap Kehadiran <i>Mass Rapid Transit</i>	II-13
II.3.2	Algoritma <i>Naïve Bayes Classifier</i> Untuk Analisis <i>Sentiment</i> Pengguna <i>Twitter</i> Terhadap <i>Provider By.u</i>	II-13
II.3.3	<i>Sentiment Analysis</i> Menggunakan <i>Naïve Bayes Classifier</i> pada <i>Tweet</i> Tentang Zakat.....	II-14
II.4	<i>State Of Art</i>	II-14
	BAB III METODOLOGI PENELITIAN.....	III-1
III.1	Metode Penelitian	III-1
III.2	Metodologi Pengembangan Sistem	III-1
III.2.1	<i>Cross Industry Standard Process for Data Mining</i>	III-1
III.2.2	<i>Scrapping</i>	III-2
III.2.3	Pendekatan <i>Supervised Learning</i>	III-2
III.2.4	<i>Modeling</i>	III-2
III.2.5	Evaluasi	III-2
III.3	Tahapan Penelitian	III-3
	BAB IV HASIL DAN PEMBAHASAN	IV-1
IV.1	Pembahasan Hasil Menurut Metode CRISP-DM	IV-1
IV.1.1	<i>Business Understanding</i>	IV-1
IV.1.2	<i>Data Understanding</i>	IV-1
IV.1.3	<i>Data Preparation</i>	IV-5
IV.1.4	<i>Modeling</i>	IV-14
IV.1.5	<i>Evaluation</i>	IV-16
IV.1.6	<i>Deployment</i>	IV-22
IV.2	Pengujian	IV-24
	BAB V SIMPULAN DAN SARAN	V-1
V.1	Simpulan.....	V-1
V.2	Saran	V-1
	DAFTAR PUSTAKA	xv

DAFTAR GAMBAR

Gambar 1. 1 Grafik pengguna media sosial di Indonesia	I-2
Gambar 2. 1 Contoh case folding.....	II-5
Gambar 2. 2 Filtering	II-5
Gambar 2. 3 Contoh Tokenizing	II-6
Gambar 2. 4 Contoh stemming	II-7
Gambar 2. 5 Contoh stopword removal	II-8
Gambar 2. 6 Bag Of Words.....	II-8
Gambar 2. 7 Monte Carlo Cross Validation.....	II-9
Gambar 2. 8 Alur CRISP-DM.....	II-11
Gambar 3. 1 Alur pendekatan Supervised Learning	III-1
Gambar 4. 1 Query Scrapping.....	IV-2
Gambar 4. 2 Get data	IV-2
Gambar 4. 3 Load dataset.....	IV-2
Gambar 4. 4 Banyaknya data terduplicasi	IV-3
Gambar 4. 5 Banyaknya jenis tweet.....	IV-3
Gambar 4. 6 Cek data kosong	IV-3
Gambar 4. 7 Data yang memiliki tweet kosong	IV-4
Gambar 4. 8 Data yang memiliki username kosong	IV-4
Gambar 4. 9 Word Cloud kata terbanyak.....	IV-4
Gambar 4. 10 Bar plot kata terbanyak	IV-5
Gambar 4. 11 Bar plot username tweet terbanyak	IV-5
Gambar 4. 12 Fungsi case folding dan filtering	IV-6
Gambar 4. 13 Tweet semula.....	IV-6

Gambar 4. 14 Tweet setelah case folding dan filtering.....	IV-7
Gambar 4. 15 Fungsi Tokenisasi.....	IV-7
Gambar 4. 16 Output Tokenisasi	IV-7
Gambar 4. 17 Fungsi Stemming.....	IV-8
Gambar 4. 18 Output dari proses Stemming.....	IV-8
Gambar 4. 19 Fungsi Stopword Removal	IV-9
Gambar 4. 20 Output Stopword Removal.....	IV-9
Gambar 4. 21 Fungsi Labeling.....	IV-10
Gambar 4. 22 Hasil Labeling	IV-10
Gambar 4. 23 Data terduplicasi setelah preprocessing	IV-11
Gambar 4. 24 Pie chart pembagian sentiment.....	IV-11
Gambar 4. 25 Bar plot kalimat positif.....	IV-12
Gambar 4. 26 Wordcloud kalimat positif.....	IV-12
Gambar 4. 27 Bar plot kalimat netral.....	IV-12
Gambar 4. 28 Wordcloud kalimat netral.....	IV-13
Gambar 4. 29 Bar plot kalimat negatif.....	IV-13
Gambar 4. 30 Wordcloud kalimat negatif.....	IV-13
Gambar 4. 31 Modelling	IV-14
Gambar 4. 32 Hasil Training.....	IV-14
Gambar 4. 33 Monte-Carlo Cross Validation	IV-17
Gambar 4. 34 Perbandingan data uji dan data latih	IV-18
Gambar 4. 35 Confusion matrix	IV-19
Gambar 4. 36 Save model format Pickle	IV-22
Gambar 4. 37 Code streamlit	IV-22
Gambar 4. 38 Hasil positif	IV-23

Gambar 4. 39 Hasil negatif IV-23

Gambar 4. 40 Hasil netral IV-23

DAFTAR TABEL

Tabel 2. 1 Tabel Confusion Matrix	II-10
Tabel 2. 2 Tabel State Of Art	II-15
Tabel 4. 1 Hasil evaluasi model dengan MCCV.....	IV-17
Tabel 4. 2 Pengujian sentimen dari tweet	IV-24

DAFTAR SINGKATAN DAN ISTILAH

Singkatan	Kepanjangan
NBC	<i>Naive Bayes Classifier</i>
CRISP-DM	<i>Cross Industry Standard Process for Data Mining</i>
ML	<i>Machine Learning</i>
MCCV	<i>Monte Carlo Cross validation</i>
PEMILU	Pemilihan Umum
PILPRES	Pemilihan Presiden
KPU	Komisi Pemilihan Umum

DAFTAR LAMPIRAN

LAMPIRAN A: PYTHON NOTEBOOK SCRAPPING.....	xvii
LAMPIRAN B: PYTHON NOTEBOOK DATA UNDERSTANDING.....	xviii
LAMPIRAN C: PYTHON NOTEBOOK DATA PREPARATION	xxi
LAMPIRAN D:PYTHON NOTEBOOK MODELLING	xxix
LAMPIRAN E:PYTHON NOTEBOOK EVALUATION	xxxii
LAMPIRAN F:PYTHON NOTEBOOK DEPLOYMENT	xxxv

BAB I PENDAHULUAN

I.1 Latar Belakang

Indonesia merupakan negara yang memiliki bentuk pemerintahan presidensial dan demokrasi. Pemerintahan presidensial berarti kepemimpinan pada negara tersebut dipimpin oleh seorang presiden, demokrasi berarti kekuasaan tertinggi ada di tangan rakyat sehingga yang dapat memilih siapa pemimpin pada negara tersebut. Seperti yang kita ketahui jika pemilihan presiden diadakan dengan Pemilu (Pemilihan Umum) yang bertujuan untuk menentukan eksekutif dan legislatif serta diselenggerakan oleh KPU (Komisi Pemilihan Umum).

Pemilu pertama kali dilaksanakan pada 29 September 1955 untuk memilih anggota DPR dilanjutkan pada 15 Desember 1955 untuk memilih anggota Dewan Konstituante, pada saat ini Indonesia masih dipimpin oleh Ir. Soekarno. Pada tahun 1967 ada SUPERSEMAR (Surat Perjanjian Sebelas Maret) yang menyatakan penyerahan kepemimpinan dari Ir. Soekarno kepada Soeharto. Pemilu pada tahun 1971, 1977, 1997 ketiga pemilu tersebut hanya digunakan untuk memilih DPR hingga akhirnya pada tahun 1999, setelah masa kepemimpinan Ir. B. J. Habibie, FREng. Diadakan pemilihan presiden (Pilpres) tetapi melalui sidang paripurna MPR, dengan mencatatkan Abdurrahman Wahid (Gusdur) menjadi presiden ke-4 Indonesia dan didampingi Megawati Soekarnoputri sebagai wakil presiden, hingga pada tahun 2004 merupakan pertama kalinya pemilihan presiden dilakukan secara luberjurdil (langsung, umum, bebas, rahasia, jujur dan adil) dengan memenangkan Susilo Bambang Yudhoyono sebagai presiden dan Jusuf Kalla sebagai wakil presiden. Terakhir dilaksanakan pada 2019 yang dimenangkan oleh petahan yakni Ir. H. Joko Widodo.

Hingga sekarang pemilihan presiden masih dilakukan untuk menentukan pemimpin negara Indonesia, tetapi ada yang berbeda antara zaman terdahulu dengan zaman sekarang, pada zaman sekarang kita bisa mengetahui reaksi masyarakat terhadap pemilu dan juga calon presiden yang diusung, apalagi melalui media sosial yang merupakan komponen primer manusia era *modern*.



Gambar 1. 1 Grafik pengguna media sosial di Indonesia
(dataindonesia.id, 2022)

Pengguna media sosial di Indonesia per tahun 2022 dapat dilihat pada gambar 1.1 di atas dan menunjukkan sudah mencapai 191 juta jiwa dan total masyarakat Indonesia ialah 275 juta jiwa, berarti sudah mencapai 69% jiwa di Indonesia menggunakan sosial media hal ini bisa menjadi alasan media sosial adalah media yang bisa menjadi pertimbangan dalam melihat elektabilitas maupun perspektif masyarakat terhadap Pemilu, hal tersebut merupakan tugas dari juru kampanye atau tim pemenangan dari suatu partai atau calon yang ingin manyalonkan dirinya di Pemilu 2024 untuk menaikkan elektabilitas partai atau calon yang ia dukung. Media sosial yang massif dan menjadi persebaran opini dari masyarakat Indonesia ialah *twitter*, karena banyaknya pendapat yang disampaikan dari warganet Indonesia akan menghasilkan berbagai macam reaksi, maka dapat dilakukan sentiment analisis untuk mendapatkan nada emosional *tweet* warganet Indonesia, algoritma yang banyak digunakan untuk mendapatkan sentimen ialah *Naïve Bayes*, dengan menghitung probabilitas dengan dasar *bayes theorem*.

Setelah mendalami permasalahan tersebut, maka penulis tertarik untuk menganalisa data dari *twitter* tentang Pilpres 2024 dan menuangkannya pada penelitian Skripsi yang berjudul "**SENTIMENT ANALISIS PILPRES 2024 PADA MEDIA SOSIAL TWITTER MENGGUNAKAN NAÏVE BAYES CLASSIFIER**".

I.2 Rumusan Masalah

Berdasarkan dari latar belakang yang sudah ditulis, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana kita mengetahui akurasi dari pengolahan data menggunakan algoritma *Naive Bayes Classifier*, sehingga juru kampanye dapat memanfaatkan hasil penelitian ini dengan mengetahui nilai sentimen positif, negatif dan netral.
2. Bagaimana juru kampanye dapat melakukan uji sentimen pada data *twitter* dengan mudah.

I.3 Batasan Masalah

Adapun Batasan masalah yang ada pada penelitian ini adalah sebagai berikut:

1. Data yang digunakan lebih dari 100000 (seratus ribu) data
2. *Tweet* yang didapat ialah *tweet* sejak Januari 2022 hingga Desember 2022

I.4 Tujuan Penelitian

Berdasarkan rumusan masalah di atas, maka yang menjadi tujuan penelitian yang akan dilakukan antara lain:

1. Menerapkan metode *Naïve Bayes Classifier* guna melakukan sentimen analisis dari media sosial *twitter* agar menjadi pertimbangan kontestan politik.
2. Mengetahui akurasi terbaik dan melakukan sentimen analisis tentang pilpres 2024 guna mengetahui bentuk opini positif, negatif dan netral.

I.5 Keluaran Penelitian

Dalam penelitian ini luaran yang akan dihasilkan sebanyak 2(dua) luaran, yakni:

1. Laporan Penelitian, Python *Notebook* dan visualisasi data
2. Web sederhana penerapan model sentimen analisis

I.6 Sistematika Penulisan

Untuk memberikan gambaran mengenai penelitian ini, maka disusun sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN : membahas secara singkat mengenai latar belakang rumusan masalah, batasan masalah, tujuan penelitian, keluaran penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI : membahas tentang artikel-artikel jurnal dari karya para peneliti sebelumnya yang berguna dalam proses pembuatan sistem.

BAB III METODOLOGI PENELITIAN : menjelaskan bagaimana urutan langkah penyelesaian masalah berdasarkan rumusan masalah, serta penjelasan metode penelitian yang akan digunakan.

BAB IV HASIL DAN PEMBAHASAN : tentang hasil penelitian dan hasil analisis dari sistem.

BAB V SIMPULAN DAN SARAN : simpulan yang diperoleh dari analisis sistem dan saran yang bermanfaat.

BAB II LANDASAN TEORI

II.1 Teori Terkait Permasalahan

II.1.1 *Sentiment Analysis*

Sentiment Analysis adalah pengumpulan pandangan orang tentang setiap peristiwa yang terjadi dalam kehidupan nyata. Dalam situasi seperti itu di mana dunia sedang melalui, memahami emosi dari orang-orang berdiri sangat penting. Skenario kubur dimana orang tidak bisa keluar dari rumah mereka menuntut eksplorasi-ing apa orang-orang benar-benar berpikir tentang keseluruhan skenario. Oleh karena itu, penulis telah merencanakan pekerjaan ini di bawah menghadapi situasi yang menuntut terutama di media sosial (Chakraborty, dkk., 2020)

Analisis sentimen adalah proses untuk mengidentifikasi dan mengenali atau mengkategorikan emosi pengguna atau pendapat untuk layanan apa pun seperti film, masalah produk, acara, atau setiap atribut adalah positif, negatif atau netral. Sumber untuk analisis ini adalah saluran komunikasi sosial yaitu situs Web yang Meliputi *review*, forum diskusi, *blog*, *micro-blog*, *twitter* dll. Bidang penelitian ini sangat populer saat ini karena data pendapatnya di mana pengguna dapat menemukan ulasannya layanan apa pun yang berguna untuk kehidupan sehari-hari mereka. Besar jumlah data opini disimpan dalam bentuk digital. Untuk topik tertentu atau pendapat analisis sentimen yang menghubungkan penambangan data bekerja dan memberikan *output*. (Pandya & Mehta, 2020)

II.1.2 *Naïve Bayes Classifier*

Naïve Bayes Classifier adalah metode klasifikasi berdasarkan teorema Bayes. Pengklasifikasi *Naïve Bayes* dikenal lebih baik daripada beberapa metode klasifikasi lainnya. Karena pertama, ciri utama dari *Naïve Bayes* adalah asumsi independensi (naif) yang sangat kuat dari setiap kondisi atau peristiwa. Kedua, modelnya simple dan mudah dibuat. Ketiga, model dapat diimplementasikan untuk set data yang besar. Dasar salah satu teorema *Naïve Bayes* yang digunakan adalah rumus bayes sebagai berikut: (Han, dkk., 2012).

Metode *Naïve bayes classifier* berasal dari *bayes theorem* yang ditemukan oleh Thomas Bayes pada tahun 1770. Teorema bayes adalah sebuah teorema

dengan dua penafsiran berbeda. Teorema ini menyatakan seberapa jauh derajad kepercayaan subjektif harus berubah secara rasional ketika diberikan petunjuk baru. Teori ini juga berasal dari penerapan teori probabilitas.

Persamaan 1 adalah teori *Naïve Bayes*:

$$P(H|e) = \frac{P(e|H)P(H)}{P(e)} \quad \text{Persamaan (1)}$$

P(H|e) = peluang kejadian **H** apabila **e** terjadi

P(e|H) = peluang kejadian **e** apabila **H** terjadi

P(H) = probabilitas kejadian (H)

P(e) = probabilitas (**e**) atau disebut *prior probability*. Berlaku jika $(e) \neq 0$

II.1.3 *Machine Learning*

Salah satu penerapan bidang kecerdasan buatan adalah *Machine Learning* atau dikenal juga dengan pembelajaran mesin dalam bahasa Indonesia. Gagasan di balik *Machine Learning* adalah memberi komputer kemampuan untuk belajar sendiri dari kumpulan data yang disediakan sebelumnya, memanfaatkan algoritme dan model untuk membuat prediksi. Tujuan utama pembelajaran mesin adalah untuk mengidentifikasi pola yang tepat dalam sekumpulan data sehingga dapat membuat model untuk mengimplementasikan operasi input-output tanpa memerlukan kode program secara eksplisit.(Tiwari, 2017). *Machine Learning* dibagi dalam 3 bentuk, yakni *Supervised Learning*, *Unsupervised Learning* dan *Generative Learning*. Sentiment analisis dengan algoritma *Naïve Bayes* menggunakan metode *Supervised Learning*.

1. *Supervised Learning*

Supervised Learning adalah bidang pengenalan pola dan statistik dalam ilmu komputer. Ini adalah studi ilmiah tentang algoritma dan model statistik, yang digunakan untuk melakukan tugas tertentu secara efisien, tanpa menggunakan instruksi eksplisit, tetapi mengandalkan model. Algoritma pembelajaran yang diawasi membangun model matematika dari data sampel untuk membuat prediksi

tanpa memerlukan pemrograman eksplisit untuk melakukan tugas. (P. Xu, dkk, 2021)

Supervised Learning adalah suatu metode untuk menciptakan *Artificial Intelligence* (AI), untuk mengidentifikasi pola dalam kumpulan data yang tidak di klasifikasikan atau tidak di beri label. Algoritma yang bertujuan untuk memperkirakan fungsi pemetaan sehingga ketika ada variabel *input* (X) kita dapat memprediksi variabel *output* (Y). Algoritma *Supervised Learning* dapat digunakan untuk memproses berbagai jenis data, mulai data yang terstruktur hingga yang tidak terstruktur. (Altamevia & Oktafia Lingga Wijaya, 2023)

II.2 Teori Pendukung

II.2.1 Python

Python adalah bahasa pemrograman tingkat tinggi, intuitif, dan dinamis. Lekukan alih-alih kurung kurawal untuk mengamankan menggambarkan blok kode yang masuk ke dalam titik bahasa, tetapi tidak menerima cinta karena dibandingkan dengan pembelajaran mesin bahasa lain dan kecerdasan buatan. Perkembangan Python berawal dari hobi penciptanya dan menggulirkannya karena ingin membuat bahasa yang indah dan mudah dibaca semua orang. Python mendapat perhatian karena membuat pekerjaan jauh lebih produktif dan sederhana. Lalu, mengapa kita menolak untuk berupaya menjadi lebih produktif meskipun kita memiliki akses yang luar biasa ke daya komputasi (Rawat, 2020)

Python adalah bahasa pemrograman interpretative yang dianggap mudah dipelajari serta berfokus pada keterbacaan kode, dengan kata lain python diklaim sebagai Bahasa pemrograman yang memiliki kode-kode pemrograman yang sangat jelas, lengkap dan mudah untuk dipahami. (Jubilee, 2019)

II.2.2 Text Mining

Text Mining adalah salah satu bidang yang sampai saat ini masih berkembang dengan pesat, dengan tugasnya dalam mengekstraksi atau mengumpulkan informasi yang bermakna dari teks alami suatu bahasa. Ini dapat diartikan sebagai proses menganalisis suatu teks untuk kemudian diekstrak informasi-informasi yang berguna dari teks tersebut untuk tujuan terentu. Dalam

budaya modern, teks adalah salah satu media dalam pertukaran informasi, dibanding dengan *database*, teks tidak terstruktur, memiliki bermacam-macam bentuk, dan lebih sulit ditangani menggunakan algoritma tertentu.(Witten, 2004)

Pada kasus ini, yaitu *Text Mining* sumber data yang berupa teks tidak memiliki struktur yang jelas dan memiliki bermacam bentuk, sehingga disebut sebagai *unstructured data*. Maka dari itu, butuh proses untuk membuat data menjadi lebih terstruktur sehingga ekstraksi informasi dari teks akan lebih mudah, tepat, dan sangat penting dalam proses *text mining*. Sumber data yang digunakan, yaitu novel berbahasa Indonesia merupakan *unstructured data*, sehingga butuh proses untuk membuat data menjadi lebih terstruktur. Salah satunya adalah dengan diawali oleh *preprocessing*, yang mana nanti akan menghasilkan fitur yang lebih representatif dibanding sumber data *tweet* berbahasa Indonesia yang belum dipersiapkan dan masih tidak berstruktur.

II.2.3 Jupyter Notebook

Jupyter adalah organisasi non-profit untuk mengembangkan software interaktif dalam berbagai bahasa pemrograman. *Notebook* adalah satu software buatan *Jupyter*, adalah aplikasi *web open-source* yang memungkinkan Anda membuat dan berbagi dokumen interaktif yang berisi kode *live*, persamaan, visualisasi, dan teks naratif yang kaya.(Prijono, 2019)

Pada penelitian ini, *Jupyter Notebook* digunakan sebagai salah satu text editor untuk menuliskan kode-kode program Python serta memvisualisasikan data hasil olah pada sistem ini.

II.2.4 Preprocessing

Tahap awal dari *Text Mining* adalah *preprocessing*. Fitur (yaitu, variabel) diekstraksi dari teks menggunakan penambangan teks kosakata tertutup, penambangan teks kosakata terbuka, atau keduanya setelah keputusan prapemrosesan telah dibuat dan dipraktikkan. Unit analisis kemudian fitur tersebut. (Hickman, dkk., 2022).

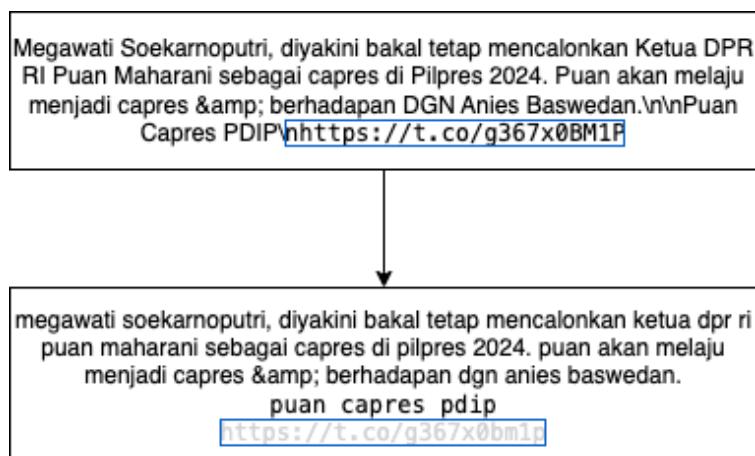
Proses ini akan melakukan penyeragaman *case tweet* yang merubah semua *tweet* menjadi *lowercase*, menghilangkan tanda *mention* serta *username* *termention*,

menghilangkan tautan pada *tweet*, kemudian membuat token dari data *input*, sehingga data lebih bersih, terstruktur dan dapat diolah lebih lanjut.

Pada penelitian ini, tahap *preprocessing* yang diterapkan adalah *Case Folding*, *Lemmatization*, *Stopword Removal* dan *Tokenizing*.

1. Case Folding

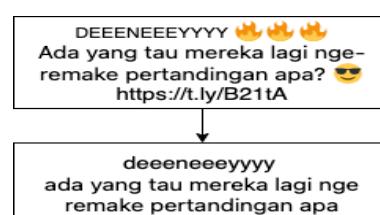
Case Folding digunakan untuk menyeragamkan seluruh teks dalam case yang seragam, menjadi huruf kecil (*lowercase*) ataupun huruf kapital (*uppercase*). *Case Folding* digunakan pada penelitian ini adalah penyeragaman menjadi *lowercase*. Contoh dari penerapan *Case Folding* bisa dilihat pada gambar dibawah ini:



Gambar 2. 1 Contoh *case folding*

2. Filtering

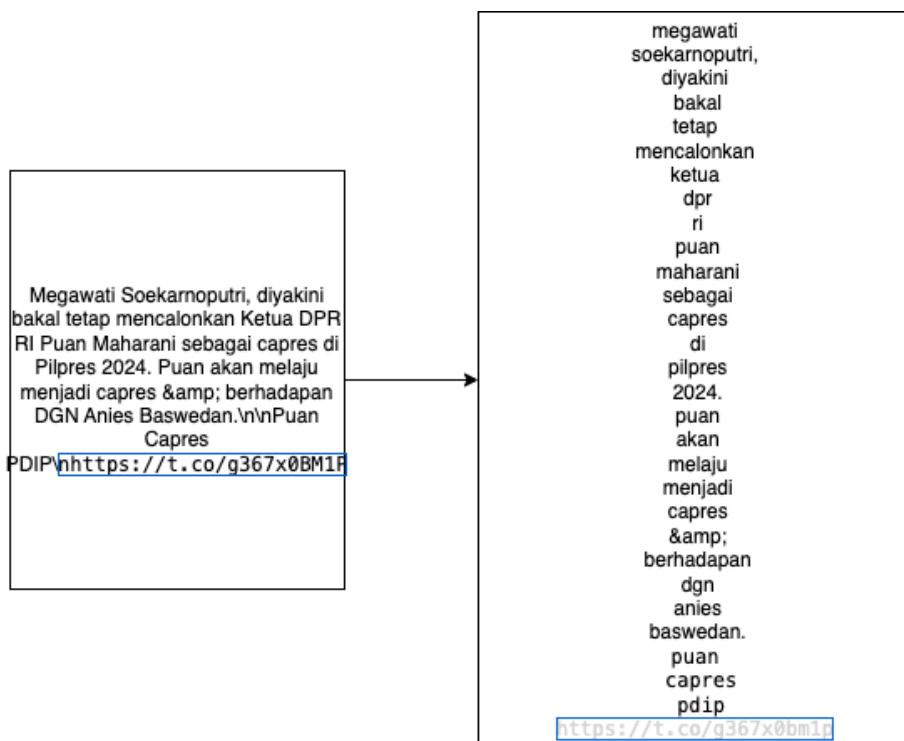
Menghapus semua angka, tanda baca, dan/atau karakter khusus dari teks. Juga umum untuk hanya menghapus angka atau tanda baca daripada semua karakter non alfabet termasuk *emoticon* dan *hyperlink*. Hal ini membuat kalimat yang akan diuji lebih fokus kepada kalimat penting saja.



Gambar 2. 2 *Filtering*

3. Tokenizing

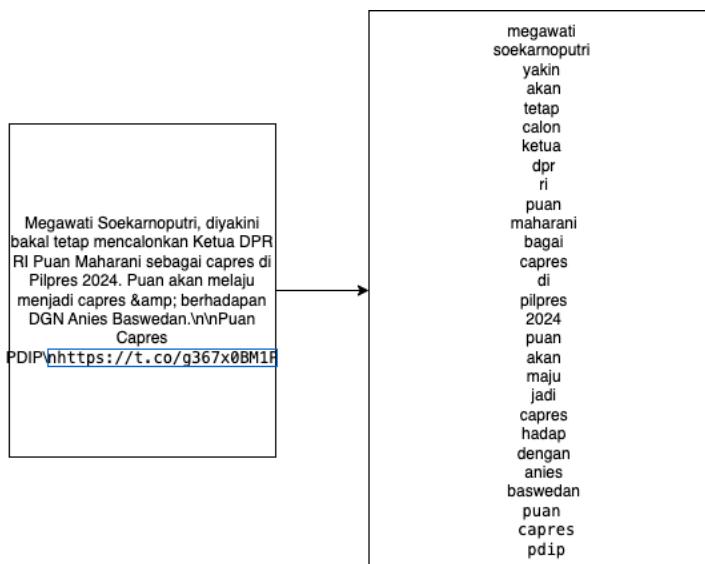
Proses *tokenizing* adalah proses dimana string dipotong menjadi beberapa bagian dengan melihat delimiternya, seperti tipe kapitalisasi, keberadaan digit, tanda baca, karakter special dan sebagainya. Pemecahan dokumen menjadi kata – kata tunggal silakukan dengan caara men-scan dokumen dan setiap kata akan teridentifikasi atau terpisahkan dengan kalimatnya oleh *delimiter*. *Tokenizing* adalah proses dimana data input dibagi menjadi beberapa token sesuai dengan jumlah kalimat menggunakan delimiter ‘.’ Pada teks input contoh *tokenizing* ada pada gambar dibawah.



Gambar 2. 3 Contoh *Tokenizing*

4. Stemming

Stemming adalah teknik pada *Natural Language Processing* yang digunakan untuk mengembalikan kata kepada kata dasarnya yang disesuaikan dengan kamus Bahasa Indonesia, proses stemming dilakukan dengan menggunakan *library* sastrawi. *Stemming* digunakan pada kebutuhan yang berhubungan dengan text mining seperti *information retrieval* yang dilakukan pada tahap preprocessing.



Gambar 2. 4 Contoh *stemming*

5. Stopword Removal

Stopword Removal adalah tahap pemilihan kata-kata yang dianggap penting. Terdapat dua metode yang dapat digunakan dalam tahap *Stopword Removal*, yakni:

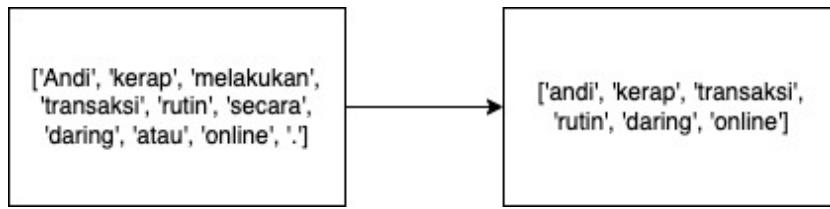
a) *Stoplist*

Pada metode ini, kita menyiapkan kumpulan kata yang tidak deskriptif/tidak penting yang disebut *stoplist*. Kata yang termasuk ke dalam *stoplist* akan dibuang dan tidak digunakan pada proses selanjutnya.

b) *Wordlist*

Wordlist merupakan kebalikan dari *stoplist*, pada metode ini kita menyiapkan kumpulan kata yang deskriptif yang disebut *wordlist*. Hanya

kata yang termasuk ke dalam *wordlist* yang akan digunakan pada proses selanjutnya, sementara kata lainnya akan dibuang.



Gambar 2. 5 Contoh *stopword removal*

II.2.5 *Bag of Words*

Bag of Words adalah salah satu cara paling efektif untuk merepresentasikan data teks, dengan model *Bag of Words*. Teks adalah format utama untuk semua informasi *online*, termasuk Wikipedia, Gmail, dan sumber lainnya. Teks ini dikategorikan dan beberapa fitur teks diekstrak menggunakan model *Bag of Words*. Tulisan yang dapat diakses secara online ini merupakan data tidak terstruktur yang mengikuti struktur yang telah ditentukan. (Kumar, 2019)

Bag of Words adalah model serbaguna yang dapat digunakan sebagai fitur algoritma seleksi, klasifikasi dokumen dan gambar-gambar. Dalam pengklasifikasian citra dapat dilakukan dengan membuat *bag* untuk setiap fitur gambar, dan dalam klasifikasi dokumen, *Bag of Words* adalah vektor jumlah kemunculan kata, yang juga disebut histogram dari dokumen. *Bag of Words* juga salah satu metode yang digunakan untuk seleksi fitur dan klasifikasi. Metode ini memiliki kemampuan yang hebat untuk memilih dan mengklasifikasikan fitur dengan membuat *bag* untuk setiap jenis instans (Qader, dkk., 2019)

Model *Bag of Words*, yang hanya menghitung berapa kali setiap kata muncul dalam kalimat (atau dokumen), adalah contoh yang mudah dipahami, "umar sedang bermain di rumah om umar" terdiri dari frasa "umar" (dua kali), "sedang", "bermain", "di", "rumah" dan "om" (sekali).

umar bermain di rumah om umar					
umar	bermain	di	rumah	om	tante
2	1	1	1	1	0

Gambar 2. 6 Bag Of Words

II.2.6 Monte Carlo Cross Validation

Monte Carlo Cross Validation (MCCV), suatu bentuk validasi model, pertama kali diperkenalkan oleh Picard dan Cook (1984). Shao (1993) membuktikan bahwa metode ini konsisten secara asimtotik dan memiliki peluang lebih besar daripada *cross validation* lain untuk memilih model terbaik dengan kemampuan prediksi yang lebih akurat. MCCV meninggalkan bagian penting dari sampel pada suatu waktu selama pembuatan model dan validasi dan mengulangi prosedur berkali-kali. Jika dibandingkan dengan metode biasa untuk memilih variabel prediktor terbaik (yaitu regresi bertahap dan menggunakan statistik seperti Mallows Cp atau hipotesis *P-value*), MCCV mungkin lebih diinginkan karena mengevaluasi model yang berbeda sesuai dengan kemampuan prediksi mereka menggunakan banyak model yang berbeda. Kombinasi set data validasi. Menariknya, MCCV belum diuji dalam analisis regresi hidrologi, di mana orang sering berurusan dengan kumpulan data terbatas yang sangat terbatas dan langka. (Haddad et al., 2013)

Validasi *Monte Carlo* menggunakan lipatan pengganti yang dipilih secara acak, sebuah proses yang dikenal sebagai *bootstrap*. Akibatnya, pendekatan *Monte Carlo* dapat menyebabkan beberapa sampel digunakan beberapa kali untuk data pelatihan dan pengujian, atau tidak digunakan sama sekali. Sebagian besar waktu, metode *Monte Carlo* menggunakan 1.000 simulasi atau lebih, yang dapat membuatnya lambat. (Ramezan, dkk, 2019)



Gambar 2. 7 Monte Carlo Cross Validation

II.2.7 *Confusion Matrix*

Alat evaluasi visual untuk pembelajaran mesin disebut *Confusion Matrix*. Dalam *Confusion Matrix*, baris sesuai dengan hasil kelas yang sebenarnya, dan kolom sesuai dengan hasil kelas yang diprediksi. Ini mencantumkan setiap skenario yang dapat muncul dari masalah klasifikasi. Dengan menggunakan masalah klasifikasi biner sebagai contoh, sebuah matriks konfusi memiliki dimensi 2 kali 2. Matriks Konfusi dapat digunakan untuk membuat sejumlah metrik kinerja algoritma, termasuk laju pemeriksaan wilayah positif dan laju penarikan kembali kelas negatif. Kriteria ini dapat digunakan dengan algoritma klasifikasi apa pun. (J. Xu, dkk, 2020)

Sebuah *Confusion Matrix*, sering disebut tabel kontingensi atau matriks kesalahan, adalah struktur tabel yang memungkinkan untuk melihat seberapa baik algoritma *Supervised Learning* bekerja. Contoh pada kelas prediksi diwakili oleh setiap kolom matriks, sedangkan kejadian pada kelas aktual diwakili oleh setiap baris. Diagonal tabel berisi semua prediksi akurat, membuatnya mudah untuk angka bukan nol di luar diagonal untuk mengidentifikasi kesalahan. (Patro & Ranjan Patra, 2014)

Tabel 2. 1 Tabel *Confusion Matrix*

	C1	C2
C1	<i>True Positive</i> (TP)	<i>False Positive</i> (FP)
C2	<i>False Negative</i> (FN)	<i>True Negative</i> (TN)

Tabel confusion matrix berfungsi untuk mencari akurasi (persamaan 2), yang mana bertujuan mewakili bagian keseluruhan dari contoh yang diklasifikasikan dengan benar (hasil positif dan negatif), *Recall* (persamaan 3) untuk mengukur kemampuan model prediksi untuk memilih *instance* kelas tertentu dari kumpulan data, *Precision* (persamaan 4) untuk menghitung akurasi asalkan kelas tertentu telah diprediksi dan *F1-score* (persamaan 5) untuk menghitung harmonisasi antara *precision* dan *recall*.

$$\text{Akurasi} = \frac{TP + TN}{TP + FP + FN + TN} \quad \text{Persamaan (2)}$$

$$Recall = \frac{TP}{TP + FN} \quad \text{Persamaan (3)}$$

$$Precision = \frac{TP}{TP + FP} \quad \text{Persamaan (4)}$$

TP = *True Positive*

TN = *True Negative*

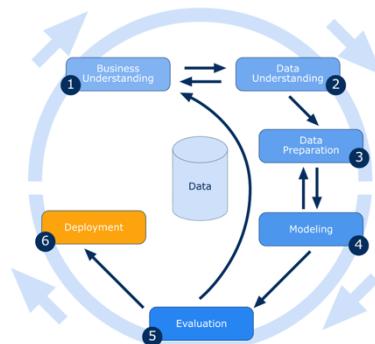
FP = *False Positive*

FN = *False Negative*

$$F1-score = \frac{2 \times recall \times precision}{recall + precision} \quad \text{Persamaan (5)}$$

II.2.8 Cross Industry Standard Process for Data Mining (CRISP-DM)

Cross Industry Standard Process for Data Mining atau yang biasa kita sebut dengan CRISP-DM merupakan model proses independen industri untuk data mining. CRISP-DM sendiri terdiri dari enam fase iteratif dari *business understanding* hingga deployment (Schröer, dkk, 2021)



Gambar 2. 8 Alur CRISP-DM

point point pada gambar 2. 8 akan dijelaskan dibawah ini:

1. Business Understanding

Bisnis harus dinilai untuk mendapatkan gambaran tentang sumber daya yang tersedia dan dibutuhkan. Penentuan tinjauan data mining merupakan salah satu aspek terpenting dalam fase ini. Jenis data mininig harus dijelaskan dan kriteria keberhasilan data mining. Rencana proyek wajib dibuat.

2. Data Understanding

Mengumpulkan data dari sumber data, mengeksplorasi dan mendeskripsikannya serta memeriksa kualitas data adalah tugas penting dalam fase ini.

3. Data Preparation

Pemilihan data harus dilakukan dengan menentukan kriteria inklusi dan eksklusi. Kualitas data yang buruk dapat ditangani dengan cleaning data.

4. Modeling

Tahap permodelan terdiri dari pemilihan teknik permodelan, membangun kasus uji dan model. Pada tahap ini dilakukan metode statistika dan Machine Learning untuk penentuan terhadap teknik data mining, alat bantu data mining, dan algoritma data mining yang akan diterapkan. Semua teknik data mining dapat digunakan.

5. Evaluation

fase evaluasi hasilnya diperiksa terhadap tujuan bisnis yang ditetapkan. Oleh karena itu, hasilnya harus ditafsirkan dan tindakan lebih lanjut harus ditentukan.

6. Deployment

fase deployment atau rencana penggunaan model adalah tahap yang paling dihargai dari proses CRISP-DM. Perencanaan untuk *Deployment* dimulai selama Business Understanding dan harus menggabungkan tidak hanya bagaimana untuk menghasilkan nilai model, juga bagaimana mengkonversi skor keputusan, dan bagaimana untuk menggabungkan keputusan dalam sistem operasional.

II.3 Penelitian-penelitian Terdahulu

II.3.1 Implementasi Metode *Naïve Bayes* untuk Analisis Sentimen Warga Jakarta Terhadap Kehadiran *Mass Rapid Transit*

Sarika A, Helena N, Noor F, Ika N.(2019), melakukan penelitian menggunakan data dari sosial media yaitu *twitter* dengan *keyword* “MRTJakarta” yang dilakukan selama masa uji coba public MRT yaitu dari tanggal 5 – 23 maret 2019. *Tweet* yang diambil sebanyak 1000 *tweet* (800 *tweet* untuk training dan 200 *tweet* untuk testing). Dalam penelitian ini naive bayes dapat memprediksi sentimen dari *tweet* yang sudah dikumpulkan terkait animo masyarakat terhadap MRTJakarta dengan akurasi sebesar 75%.

II.3.2 Algoritma *Naïve Bayes Classifier* Untuk Analisis *Sentiment* Pengguna *Twitter* Terhadap *Provider By.u*

Ike V, Bagas S.(2022), melakukan penelitian ini dapat diambil kesimpulan bahwa Algoritma *Naïve Bayes Classifier* dapat melakukan analisis sentimen dengan benar dan melakukan klasifikasi secara otomatis setelah melalui tahapan- tahapan proses, yaitu *Preprocessing* data, pembobotan kata, membuat model untuk klasifikasi otomatis dan dibuatnya data training untuk melatih klasifikasi pada data testing. Tahapan proses tersebut dapat berjalan dengan baik dan mengklasifikasikan data dengan parameter positif dan negatif. Setelah dilakukan 3 kali pengujian didapatkan hasil akurasi 80%, 80%, dan 85%. Didapatkan hasil akurasi paling tinggi pada pengujian terakhir yakni sebesar 85%. Dengan pengujian menggunakan 3 *dataset* yang memiliki jumlah data yang berbeda, dan setelah mendapatkan hasil tingkat akurasi dari proses analisis sentimen dapat disimpulkan bahwa jumlah *dataset* dalam pengujian sangat berpengaruh terhadap tingkat akurasi Algoritma *Naïve Bayes Classifier*. Hal ini ditunjukan oleh hasil tingkat akurasi pada pengujian ketiga dengan 3000 *dataset* mendapatkan nilai akurasi 85%, lebih besar daripada pengujian pertama dengan 1000 *dataset* yang hanya memiliki akurasi sebesar 80%.

II.3.3 *Sentiment Analysis Menggunakan Naïve Bayes Classifier pada Tweet Tentang Zakat*

Adhyaksa H (2020), hasil klasifikasi sentiment dari 50 *tweet* data uji menggunakan algoritma *naïve bayes* dengan seleksi fitur *Term-Frequency* serta metode *lexicon Based*, didapatkan jumlah sentiment positif yang lebih dominan dibandingkan sentimen negatif maupun netral dikarenakan pada pengujian dengan metode *lexicon based* terdapat lebih banyak *tweet* yang mengandung kata dalam kamus *lexicon* positif dibanding kata dalam kamus *lexicon* negatif. Selanjutnya, pada pengujian dengan masing-masing seleksi fitur, sentiment positif lebih dominan dikarenakan tidak keseimbangan jumlah sentiment positif, negatif dan netral dalam klasifikasi data latih menggunakan metode *lexicon based* dimana sentimen positif lebih besar sehingga sistem lebih condong dalam mengklasifikasi sentimen positif.

II.4 *State Of Art*

State of the art adalah hal yang cukup penting bagi penelitian, bermanfaat untuk mengetahui bagaimana berkembangnya ilmu pada bidang masalah *general* yang sedang diteliti sampai penulis menemukan masalah penelitian yang dapat memberikan kontribusi. *State of art* dapat diketahui lewat penelitian terdahulu.

Perbandingan penelitian terdahulu dapat dilihat dari segi persamaan penelitian atau perbedaan yang ada pada penelitian sebelumnya. Penelitian terdahulu juga ditujukan untuk membantu menemukan inspirasi bagi penelitian selanjutnya. Penelitian terdahulu yang dapat menjadi acuan penelitian ini dikemas dalam bentuk tabel dan deskripsi agar dapat mempermudah perbandingan antar satu penelitian dan penelitian lainnya. Tabel *state of art* dapat dilihat pada tabel 2.2

Tabel 2. 2 Tabel *State Of Art*

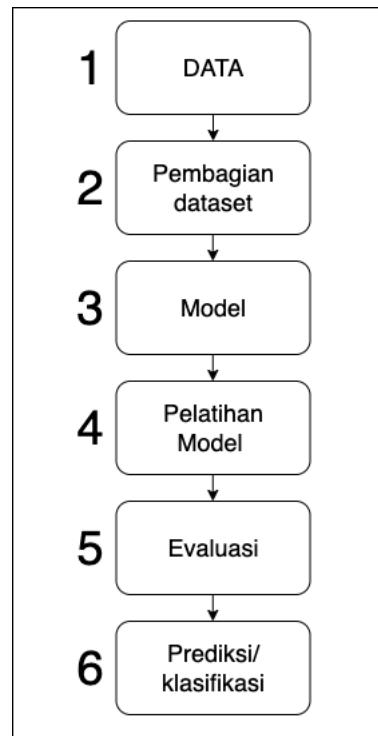
No.	Judul Penelitian	Penulis	Tahun Penelitian	Kata Kunci	Jangka Waktu Data	Banyaknya tweet	Pembagian dataset	akurasi
1.	Implementasi Metode <i>Naïve Bayes</i> untuk Analisis Sentimen Warga Jakarta Terhadap Kehadiran <i>Mass Rapid Transit</i>	Sarika A, Helena N, Noor F, Ika N.	2019	"MRTJakarta"	5 - 23 Maret 2019	1000 tweet	800 <i>Training</i> 200 <i>Test</i> (80:20)	75%
2.	Algoritma <i>Naïve Bayes Classifier</i> Untuk Analisis Sentiment	Ike V, Bagas S	2022	"By.U"	23 - 26 Mei 2021	1000,2000,3000 tweet 3x pengujian	-	85%

No.	Judul Penelitian	Penulis	Tahun Penelitian	Kata Kunci	Jangka Waktu Data	Banyaknya tweet	Pembagian dataset	akurasi
	Pengguna Twitter Terhadap Provider By.u							
3.	<i>Sentiment Analysis Menggunakan Naïve Bayes Classifier pada Tweet Tentang Zakat</i>	Adhyaksa, H.	2020	"@baznasindonesia"	4 Juni 2019	1000 tweet	950 <i>Training</i> 50 <i>Test</i> (95:5)	74%

BAB III METODOLOGI PENELITIAN

III.1 Metode Penelitian

Metode penelitian yang digunakan ialah pendekatan *supervised learning*. metode ini memiliki tahapan tahapan yaitu pengumpulan data, *preprocessing data*, pelabelan data, pemilihan model dan prediksi, bisa dilihat pada gambar 3.1.



Gambar 3. 1 Alur pendekatan *Supervised Learning*

III.2 Metodologi Pengembangan Sistem

Metode pengembangan sistem adalah pendekatan atau prosedur yang digunakan untuk merencanakan, merancang, mengembangkan, mengimplementasikan, dan memelihara sistem perangkat lunak atau sistem informasi. Metode ini menyediakan panduan dan kerangka kerja untuk mengatur langkah-langkah dalam pengembangan sistem, memastikan bahwa proyek berjalan secara terstruktur, efisien, dan efektif.

III.2.1 *Cross Industry Standard Process for Data Mining*

penelitian ini menggunakan *Cross Industry Standard Process for Data Mining* (CRISP-DM) sebagai metode pengembangan sistem, CRISP-DM memiliki

6 tahapan yakni, *business understanding*, *data understanding*, *data preparation*, *modeling*, *evaluation*, dan *deployment*. Alasan menggunakan CRISP-DM ialah karena memiliki tahapan *deploying* yang mana tahapan tersebut sesuai dengan luaran dari penelitian ini.

III.2.2 *Scrapping*

Metode pengambilan data pada penelitian ini menggunakan *scrapping*, dengan menggunakan *library* snscreape yang menggunakan bahasa pemrograman python, data yang direquest memiliki query "pilpres 2024 OR prabowo OR ganjar OR anies until:2022-12-31 since:2022-01-01" yang berarti data yang diambil adalah data yang ada sejak awal januari hingga akhir desember, data yang berhasil didapatkan ialah sebanyak 124544.

III.2.3 Pendekatan *Supervised Learning*

Penelitian ini menggunakan pendekatan *supervised learning* dengan pelaksanaan dimulai dari data, pembagian *dataset*, model, pelatihan model, evaluasi dan prediksi/klasifikasi. Pengambilan pendekatan *supervised learning* sebagai metode penelitian karena model *Naive Bayes Classifier* merupakan jenis *machine learning* dengan metode *supervised learning* dan memiliki *output* klasifikasi.

III.2.4 *Modeling*

Penelitian ini menggunakan *modeling Naive Bayes Classifier*, model ini diambil dari *bayes theorem*, model ini adalah salah satu model untuk melakukan klasifikasi, yang mana pada penelitian ini penulis ingin melatih model yang dapat melakukan klasifikasi positif, negatif dan netral.

III.2.5 *Evaluasi*

Monte Carlo Cross Validation merupakan metode yang digunakan untuk evaluasi, cara kerja MCCV ialah pengetesan secara acak pada data *test* dan *training* sehingga dapat menghasilkan representasi yang baik pada model hasil *test* dan *train*. *Confusion matrix* juga digunakan untuk membantu evaluasi pada model *Naive Bayes Classifier* ini.

III.3 Tahapan Penelitian

Berikut adalah tahapan penelitian dengan metode penelitian pendekatan *supervised learning* yang ada pada gambar 3.1.

1. Data

Aktivitas yang dilakukan dalam proses machine learning ialah pengumpulan data, pengumpulan data dilakukan dengan metode *scraping* yang sudah dijelaskan pada *point III.2.2*, dari proses ini didapatkan data kotor sebanyak 124544 data kotor, sehingga harus dibersihkan dahulu agar bisa menghilangkan nilai bias yang ada pada data dan dapat mempengaruhi *output* dari *machine learning*.

2. Pembagian *dataset*

Setelah mendapatkan data, dan sebelum melakukan pelatihan pada model *machine learning* yang akan digunakan, peneliti harus melakukan pembagian *dataset*, pada pembagian *dataset* memiliki aturan, yakni data *test* jangan sampai melebihi data *training*, karena akan menimbulkan banyak bias. Penulis mengambil perbandingan 80:20 yang berarti 80% data menjadi data *training* dan 20% menjadi data *test*.

3. Model

Model yang dipilih penulis sesuai dengan judul penelitian, yakni *Naive Bayes Classifier*, dimana model ini merupakan model jenis klasifikasi. Klasifikasi merupakan bentuk *output* yang diinginkan dimana model dapat melakukan klasifikasi pada naskah baru. Klasifikasi yang dicari ialah apakah naskah tweet ini positif, negatif, dan netral.

4. Pelatihan model

Model dilatih dengan menggunakan *dataset* pelatihan. Selama pelatihan, model mencoba menemukan pola atau relasi yang ada antara fitur-fitur data dan label yang sesuai. Tujuannya adalah untuk membuat model yang dapat mempelajari dan mewakili hubungan yang ada dalam data.

5. Evaluasi

Pada tahap ini penulis menggunakan metode *Monte Carlo Cross Validation* (MCCV) pada tahap evaluasi. MCCV melakukan evaluasi dengan cara mengacak pengambilan sample uji, serta nilai yang dicari ialah akurasi, presisi, *recall*, *F1-score*.

6. Prediksi/Klasifikasi

Setelah model dilatih dan dievaluasi, model dapat digunakan untuk melakukan prediksi atau klasifikasi pada data baru yang belum diketahui. Model ini memanfaatkan pola dan relasi yang telah dipelajari dari *dataset* pelatihan untuk menghasilkan prediksi atau klasifikasi yang sesuai. serta hasil model diexport dengan format pickle.

BAB IV HASIL DAN PEMBAHASAN

Metodologi pengembangan sistem yang digunakan pada penelitian ini ialah Cross Industry Standard Process for Data mining. Metode ini memiliki 6 langkah yaitu, *business understanding, data understanding, data preparation, modeling, evaluation dan deployment.*

IV.1 Pembahasan Hasil Menurut Metode CRISP-DM

IV.1.1 *Business Understanding*

Pemilihan umum merupakan acara demokrasi bagi warga Indonesia dan hanya diselenggarakan setiap 5 tahun sekali, dalam 5 tahun tersebut pemimpin yang terpilih wajib melakukan tanggung jawab sebagai pemimpin negara yang telah dipilih oleh masyarakat Indonesia, setelah waktu berlalu 5 tahun masyarakat dapat memilih pemimpin baru kembali. Pemilihan umum pada era digital cukup menarik, karena semua elemen masyarakat dapat berdiskusi perihal calon pemimpin negara lewat sosial media, lebih spesifik yakni pada media sosial *twitter*. Melalui satu cuitan dalam media sosial dapat mempengaruhi opini masyarakat terhadap calon tersebut, belum lagi pada tahun 2024 nanti banyak pemilih pemula dan pertama kalinya mengikuti pesta demokrasi 5 tahunan ini.

1. *Goal*

Tujuan dari analisis sentimen ini ialah:

1. Mengetahui akurasi dari *dataset* yang ada terhadap sentimen
2. Membuat web sederhana yang menggunakan model machine learning hasil analisis sentimen

IV.1.2 *Data Understanding*

Data yang digunakan dalam melakukan analisis ini merupakan data dari *twitter* yang didapat dengan cara crawling menggunakan *library python snscreape*.

Penulis menggunakan query "pilpres 2024 OR prabowo OR anies OR ganjar since:2022-01-01 until:2022-12-31" query tersebut berarti mencari *tweet* yang memiliki keyword pilpres 2024, prabowo, anies, ganjar dan data tersebut ditweet pada awal januari hingga akhir desember tahun 2022.

```

import pandas as pd
from tqdm.notebook import tqdm
import snscreape.modules.twitter as sntwitter

import datetime

scraper = sntwitter.TwitterSearchScraper('pilpres 2024 OR prabowo OR anies OR ganjar since:2023-01-01 until:2023-05-31')

```

Gambar 4. 1 Query Scrapping

```

tweets = []
n_tweet = 100000
for i, tweet in tqdm(enumerate(scraper.get_items()), total=n_tweet):
    data = [tweet.date, tweet.renderedContent, tweet.user.username]
    tweets.append(data)
    if i > n_tweet:
        break

```

Gambar 4. 2 Get data

Data disimpan menggunakan format CSV(*Comma Separated View*), agar memudahkan penulis meneliti menggunakan bahasa pemrograman python yang dibantu dengan library pandas. Data tersebut memiliki kolom yakni:

1. *Date* : tanggal dimana *tweet* tersebut dibuat
2. *Tweet*: isi dari *tweet* tersebut
3. *Username*: pengguna *twitter* yang melakukan *tweet*

Dari loading data yang berhasil di *scrapping*, penulis mendapatkan data sebanyak 124544 (seratus dua puluh ribu lima ratus empat puluh empat) data, dengan kolom *date*, *tweet* dan *username*, masing masing kolom memiliki tipe data *object*.

```

data = pd.read_csv('gab.csv')
print('data ada sebanyak', len(data), 'baris')
data = data.drop(['Unnamed: 0'], axis = 1)
print(data.columns)
print(data.dtypes)

✓ 0.6s

data ada sebanyak 124544 baris
Index(['date', 'tweet', 'username'], dtype='object')
date      object
tweet     object
username   object
dtype: object

```

Gambar 4. 3 Load dataset

Setelah *load dataset* kita harus membersihkan data dari data yang terduplicasi dan data yang *null*, penulis mencari informasi perihal berapa banyak data yang terduplicasi dan didapatkan ada 10867 (sepuluh ribu delapan ratus enam puluh tujuh), dari data tersebut digali lagi ternyata dari 10867 cuitan yang sama itu ada 1509 jenis *tweet* yang terduplicasi.

```
duplikasi = len(data.tweet)-len(data.tweet.drop_duplicates())
print(f'ada sebanyak {duplikasi} data yang terduplicasi')

✓ 0.0s                                         Python

ada sebanyak 10867 data yang terduplicasi
```

Gambar 4. 4 Banyaknya data terduplicasi

```
a = (x['size'] > 1).sum()
print(f'jadi ada tweet sebanyak {a} yang memiliki cuitan yang')

✓ 0.0s                                         Python

jadi ada tweet sebanyak 1509 yang memiliki cuitan yang sama den
```

Gambar 4. 5 Banyaknya jenis tweet

Pembersihan data duplikasi sudah dilakukan, maka tahap selanjutnya ialah melakukan pembersihan terhadap data yan memiliki data *null* atau kosong, yakni dengan cara menulis `data.isnull().sum()` dan akan menunjukkan tabel mana saja yang memiliki data kosong, dari hasil yang dilakukan peneliti mendapatkan 6 data username kosong dengan rincian 3 dari data tersebut tidak memiliki *tweet*, maka dilakukan `dropna()` agar data yang kosong tersebut hilang.

```
data.isna().sum()

✓ 0.1s

date      0
tweet     3
username  6
dtype: int64
```

Gambar 4. 6 Cek data kosong

```
tnull_rows = data[data['tweet'].isnull()]
print(tnull_rows)

✓ 0.0s
```

		date	tweet	username
108271	mbois_man75	NaN	NaN	NaN
108273	mbois_man75	NaN	NaN	NaN
108452	ismailpmk_	NaN	NaN	NaN

Gambar 4. 7 Data yang memiliki *tweet* kosong

```
unull_rows = data[data['username'].isnull()]
print(unull_rows)

✓ 0.0s

                                date \
108270 2022-01-30 02:10:01+00:00
108271          mbois_man75
108272 2022-01-30 02:09:48+00:00
108273          mbois_man75
108451 2022-01-29 03:49:00+00:00
108452          ismailpmk_

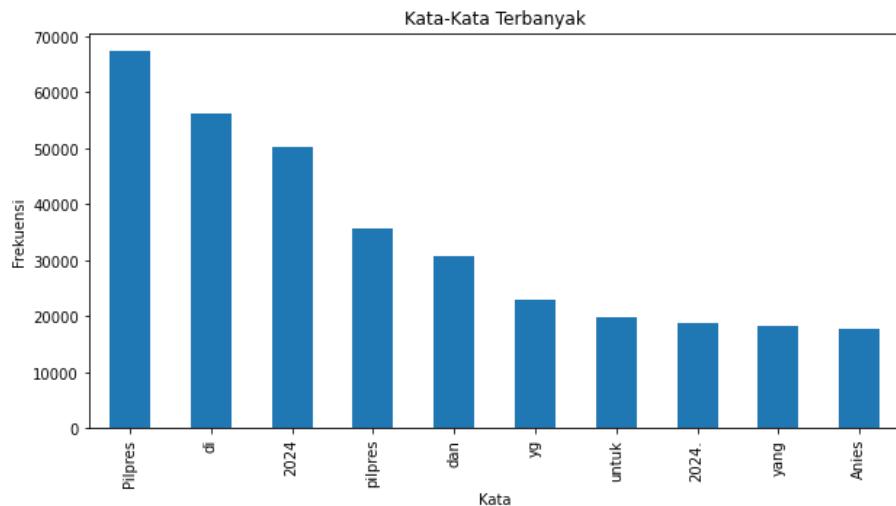

                                tweet username
108270 AHY Ajak Kader Demokrat Bersiap Hadapi Pileg d...    NaN
108271                               NaN    NaN
108272 AHY Ajak Kader Demokrat Bersiap Hadapi Pileg d...    NaN
108273                               NaN    NaN
108451 AHY Ajak Kader Demokrat Bersiap Hadapi Pileg d...    NaN
108452                               NaN    NaN
```

Gambar 4. 8 Data yang memiliki *username* kosong

Setelah melakukan pembersihan data, maka peneliti melakukan visualisasi data, terhadap *dataset* yang dibantu oleh *library* matplotlib, maka menghasilkan *wordcloud* terhadap kalimat terbanyak seperti gambar 4.9 dan versi *bar plot* kalimat terbanyak ada pada gambar 4.10.

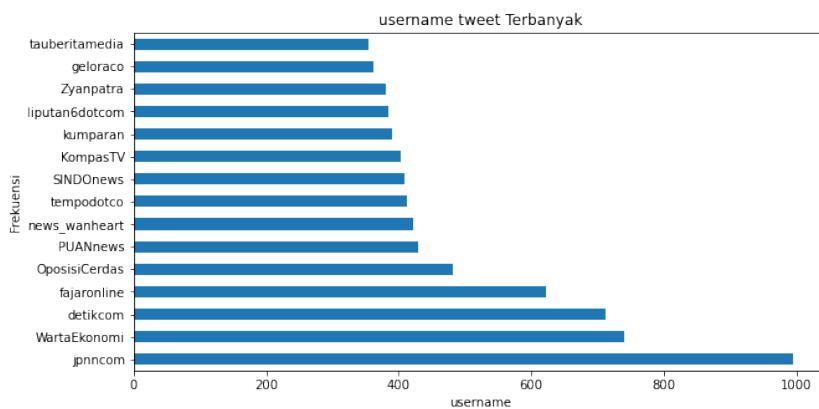


Gambar 4. 9 *Word Cloud* kata terbanyak



Gambar 4. 10 Bar plot kata terbanyak

selain menggambarkan visual dari kalimat terbanyak, peneliti juga melakukan visualisasi data terhadap username yang paling banyak melakukan tweet.



Gambar 4. 11 Bar plot username tweet terbanyak

Dari gambar 4.11 bisa dilihat ada 15 *username* yang melakukan *tweet* terbanyak dan semua tweet itu berasal dari akun media terkenal seperti Kompas TV, liputan6dotcom, kumparan, SINDOnews dan tempodotco.

IV.1.3 *Data Preparation*

Sebelum dilakukan pelatihan pada machine learning, data wajib dibersihkan dahulu, fase ini disebut dengan *preprocessing* yang mana *output* dari *preprocessing* ini ialah membuat data yang sebelumnya kotor menjadi bersih, sehingga bisa merendahkan bias yang ada pada *dataset*, hal ini sangatlah vital dalam pembuatan *machine learning* agar data yang digunakan bisa mencapai akurasi yang diinginkan dan rendah akan bias. Tahapan *preprocessing* pada penelitian ini ada 4 yaitu:

1. Case Folding dan Filtering

Case folding dan *filtering* merupakan tahap awal dari *preprocessing*, *output* dari tahap ini ialah penyamarataan huruf menjadi *lowercase*, serta menghapus *username* *twitter*, *retweet* dan *hashtag* pada data *tweet*. *Code* yang akan digunakan ada pada gambar 4. 12 pertama, kita harus menulis *import library* regex yakni 'import re' sudah ditulis pada *line* 1, lalu menulis fungsi *preprocess_tweet2* dengan parameter *tweet*. *Filtering* dilakukan dengan mendeklarasi *emoji pattern* dengan ascii dari emoji, kemudian menghapus angka, melakukan *case folding*, menghapus tautan yang ada pada tweet, menghapus *username* dan *hashtag* yang ada pada *tweet*, lalu menghapus "\n" yang ada pada setiap *tweet*, serta *remove punctuations* seperti koma, titik dan lainnya. Contoh *tweet* awal ada pada gambar 4.13

```
def preprocess_tweet2(tweet):
    EMOJI_PATTERN = re.compile(
        "["
        "\U0001F1E0-\U0001F1FF" # flags (iOS)
        "\U0001F300-\U0001F5FF" # symbols & pictographs
        "\U0001F600-\U0001F64F" # emoticons
        "\U0001F680-\U0001F6FF" # transport & map symbols
        "\U0001F700-\U0001F77F" # alchemical symbols
        "\U0001F780-\U0001F7FF" # Geometric Shapes Extended
        "\U0001F800-\U0001F8FF" # Supplemental Arrows-C
        "\U0001F900-\U0001F9FF" # Supplemental Symbols and Pictographs
        "\U0001FA00-\U0001FA6F" # Chess Symbols
        "\U0001FA70-\U0001FAFF" # Symbols and Pictographs Extended-A
        "\U00002702-\U000027B0" # Dingbats
    "])")
    tweet = re.sub(r'[0-9]+','', str(tweet))
    tweet = tweet.lower() # convert to lower case
    tweet = re.sub(r"HTTP[S]+\www[S]+\https[S]+", '', tweet, flags=re.MULTILINE) # remove URLs
    tweet = re.sub(r"@[\w+|\#|\w+]', '', tweet) # remove mentions and hashtags
    tweet = re.sub(r'\d+', '', tweet) # remove numbers
    tweet = re.sub(r'\.', '', tweet) # remove punctuation
    tweet = tweet.translate(str.maketrans("", "", string.punctuation)) # remove punctuations
    tweet = tweet.strip()
    tweet = re.sub(EMOJI_PATTERN, r'', tweet)
    tweet = re.sub(r'\n', '', tweet)
    tweet = re.sub(r'\s+', '', tweet)
    return tweet
data['tweet']=data['tweet'].apply(preprocess_tweet2)
```

Gambar 4. 12 Fungsi *case folding* dan *filtering*

"@ganjarpranowo
menggandeng\xa0ulama\xa0hingga\xa0tokoh
agama\xa0dalam memberikan suasana 'adem'
menjelang\xa0Pemilu\xa02024 di Jateng. Menurut
Ganjar, peran mereka sangat penting untuk
mengedukasi masyarakat menjelang Pilpres 2024.
<https://t.co/x0DARLOxzG>

Gambar 4. 13 Tweet semula

Setelah kita mengamati apa yang ada pada gambar 4.13, kita lakukan proses yang ada pada gambar 4.12 maka hasilnya dapat dilihat pada gambar 4.14.

menggandeng ulama hingga tokoh agama dalam memberikan suasana adem menjelang pemilu di jateng menurut ganjar peran mereka sangat penting untuk mengedukasi masyarakat menjelang pilpres

Gambar 4. 14 Tweet setelah *case folding* dan *filtering*

2. Tokenisasi

Tokenisasi merupakan tahap ketika setiap kalimat/tweet dipisah menjadi per kata agar mudah melakukan tahap selanjutnya yakni *stemming* dan *stopword removal*, fungsi untuk tokenisasi dapat dilihat pada gambar 4.8.

```
from nltk.tokenize import word_tokenize

def tokenize_column(text):
    if isinstance(text, str): # Memastikan bahwa text adalah string
        return word_tokenize(text)
    else:
        return [] # Mengembalikan list kosong jika text bukan string

# Contoh penggunaan:
data['tweet'] = data['tweet'].apply(tokenize_column)
✓ 16.9s
```

Gambar 4. 15 Fungsi Tokenisasi

output yang dihasilkan dari fungsi tokenisasi pada gambar 4.15 dapat dilihat pada gambar 4.16

['menggandeng','ulama','hingga',
 'tokoh','agama','dalam',
 'memberikan','suasana','adem','menjelang',
 'pemilu','di','jateng','menurut',
 'ganjar','peran','mereka','sangat','penting',
 'untuk','mengedukasi','masyarakat','menjelang',
 'pilpres']

Gambar 4. 16 Output Tokenisasi

bisa dilihat, data yang asalnya seperti gambar 4.14, kini berubah menjadi pengutipan tiap kata, yang menjadi data tersebut ter-tokenisasi.

3. Stemming/Lemmatization

Proses ini, adalah proses pelemasan kata, atau membuat kata menjadi kembali ke kata asal. Seperti kalimat 'menjadi', setelah di stemisasi akan menjadi kalimat 'jadi' dan kalimat yang lainnya pun menjadi kembali pada kalimat asal. Proses ini termasuk proses penting pada *preprocessing*. Fungsi untuk melakukan *stemming* dapat dilihat pada gambar 4.17.

```
stemmer = StemmerFactory().create_stemmer()
def stemming(batch):
    # Menerapkan stemming pada setiap teks dalam batch
    stemmed_batch = [stemmer.stem(text) for text in batch]

    # Melakukan penghapusan stopwords pada setiap teks dalam batch

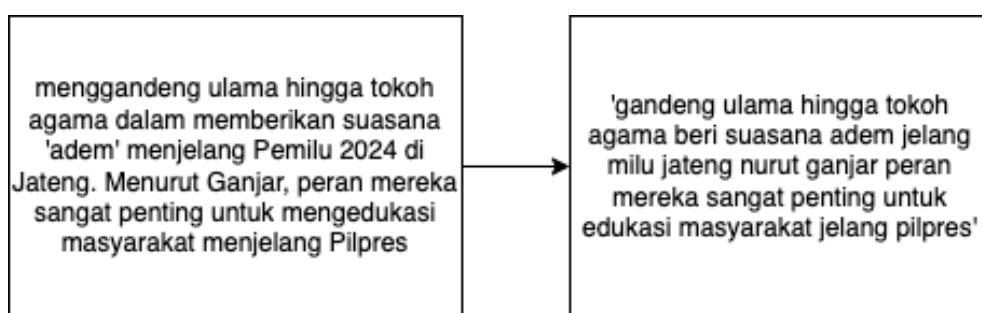
    return stemmed_batch

# Menambahkan hasil preprocessing ke dalam DataFrame
data['tweet'] = data['tweet'].apply(stemming)

# Hasil preprocessing data
print(data)
✓ 48.4s
```

Gambar 4. 17 Fungsi *Stemming*

setelah kalimat dimasukan pada fungsi *stemming*, yang mana data stemming bahasa indonesia didapat oleh library sastrawi. Kita harus mendeklarasikan '*from Sastrawi.Stemmer.Stemmerfactory import StemmerFactory*' lalu deklarasi variabel baru yakni stemmer, seperti pada gambar 4.17 dimana variabel ini memanggil fungsi dari *StemmerFactory*. untuk hasil *outputnya* dapat dilihat pada gambar 4.18.



Gambar 4. 18 *Output* dari proses *Stemming*

Dapat dilihat kalimat menggandeng menjadi gandeng, memberikan menjadi beri, inilah hasil dari proses *stemming* menggunakan *library sastrawi*.

4. Stopword Removal

Proses *stopword removal* tak kalah penting dengan proses stemming, proses ini berfungsi untuk menghilangkan kalimat kalimat yang sering muncul pada kalimat, jika kita ambil contoh bahasa inggris, kalimat yang akan hilang yakni, *i, me, you*, dll. Kalimat yang sangat sering muncul dan tidak mempengaruhi dari nilai sentimen tersebut. Fungsi *Stopword Removal* ada pada gambar 4.19.

```
stopword_remover = StopWordRemoverFactory().create_stop_word_remover()
def stopword(batch):
    # Melakukan penghapusan stopwords pada setiap teks dalam batch
    cleaned_batch = [stopword_remover.remove(text) for text in batch]
    cleaned_batch = [text for text in cleaned_batch if 'yg' not in text.lower()]

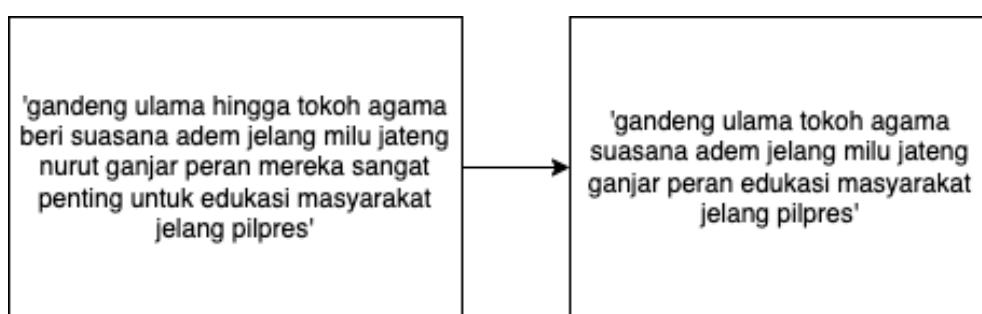
    return cleaned_batch

# Menambahkan hasil preprocessing ke dalam DataFrame
data['tweet'] = data['tweet'].apply(stopword)

# Hasil preprocessing data
print(data)
✓ 2.8s
```

Gambar 4. 19 Fungsi Stopword Removal

Sama seperti *stemming*, *stopword removal* juga menggunakan *library* dari sastrawi dengan fungsi *StopWordRemoverFactory* dan dipanggil dengan variabel '*stopword_remover*'. *Output* dari *stopword removal* ada pada gambar 4.20.



Gambar 4. 20 Output Stopword Removal

Jika dilihat secara seksama kalimat hingga, beri, nurut hilang, karena kalimat tersebut ada pada kamus *stopword removal* atau kalimat itu termasuk pada kalimat stopword, sehingga kalimat tersebut hilang.

Setelah melakukan *preprocessing*, kita melakukan labeling yakni memberi label kepada setiap *tweet* yang ada pada *dataset*, apakah ini positif, negatif atau netral. labeling sangatlah krusial karena label adalah target yang akan dilatih sebagai hasil dari latihan tersebut. Fungsi *labeling* ada pada gambar 4.21.

```
from nltk.tokenize import word_tokenize
def labelling(tweet):
    # tweet = preprocess_tweet(tweet)
    clean_tokens = word_tokenize(tweet) #tokenize
    # clean_tokens = [stemmer.stem(word) for word in tweet_tokens if word not in
    positive_words = open("positive.txt").read().splitlines()
    negative_words = open("negative.txt").read().splitlines()
    positive_count = sum([1 for word in clean_tokens if word in positive_words])
    negative_count = sum([1 for word in clean_tokens if word in negative_words])
    if positive_count > negative_count:
        return 'Positive'
    elif positive_count < negative_count:
        return 'Negative'
    else:
        return 'Neutral'
data['sentiment'] = data['tweet'].apply(labelling)
✓ 2m 27.3s
```

Gambar 4. 21 Fungsi *Labeling*

Setelah melakukan labelling maka akan menambah kolom baru lagi yakni kolom sentimen, karena pada gambar 4. 14 jelas, bahwa kolom sentimen dihasilkan dari fungsi labelling, dan menghasilkan data neutral sebanyak 49714 (empat puluh sembilan ribu tujuh ratus empat belas), data positif sebanyak 43522 (empat puluh tiga ribu lima ratus dua puluh dua), dan data negatif sebanyak 20439 (dua puluh ribu empat ratus tiga puluh sembilan), dapat dilihat pada gambar 4.15.

```
data.sentiment.value_counts()
✓ 0.0s
      Neutral    49714
      Positive   43522
      Negative   20439
      Name: sentiment, dtype: int64
```

Gambar 4. 22 Hasil *Labeling*

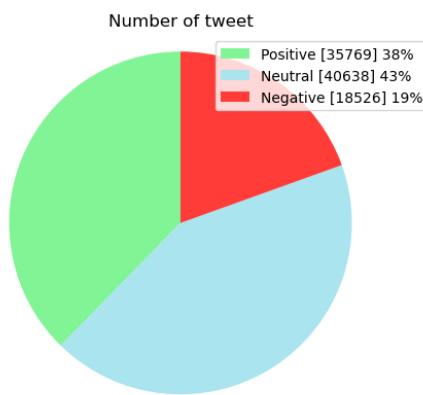
Setelah melakukan *preprocessing* kita harus melakukan *data cleaning* kembali, karena dikhawatirkan masih ada data yang terduplikasi karena adanya *tweet spam* yang tidak terdeteksi pada fase *cleaning* di *data preparation*, contohnya *tweet* "@adzam bagaimana kabarmu" dan "@manusia bagaimana kabarmu" akan terlihat

beda pada fase data preparation karena ada "@adzam" dan "@manusia" yang membedakan, jika sudah pada fase *preprocessing* maka *mention* dan *hashtag* sudah menghilang dan kedua *tweet* tersebut akan terbaca menjadi "bagaimana kabarmu" dan akan terbaca sama atau terduplicasi.

```
dat = len(data.tweet)-len(data.tweet.drop_duplicates())
print(f'ada sebanyak {dat} data yang terduplicasi')
✓ 0.0s
ada sebanyak 18742 data yang terduplicasi
```

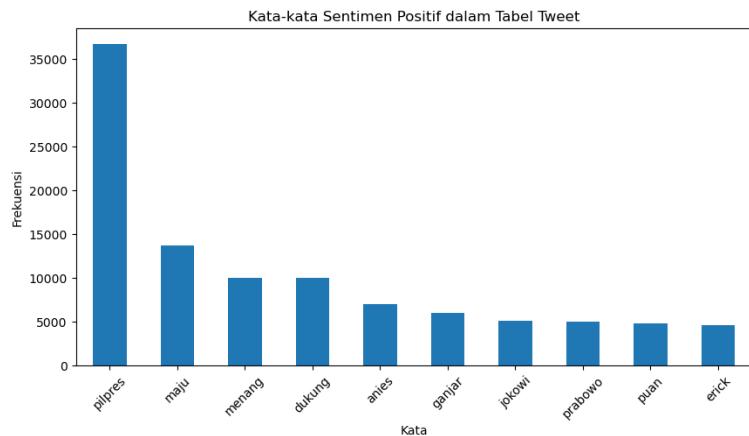
Gambar 4. 23 Data terduplicasi setelah *preprocessing*

Dapat kita lihat pada gambar 4.23 didapatkan jika ada 18742 *tweet* yang terduplicasi dan setelah menghapus data terduplicasi, data yang tersisa sebesar 94933 *tweet* dan nilai sentimen yang tersedia ialah 40638 untuk data netral, 35769 untuk data positif dan 18526 untuk negatif.

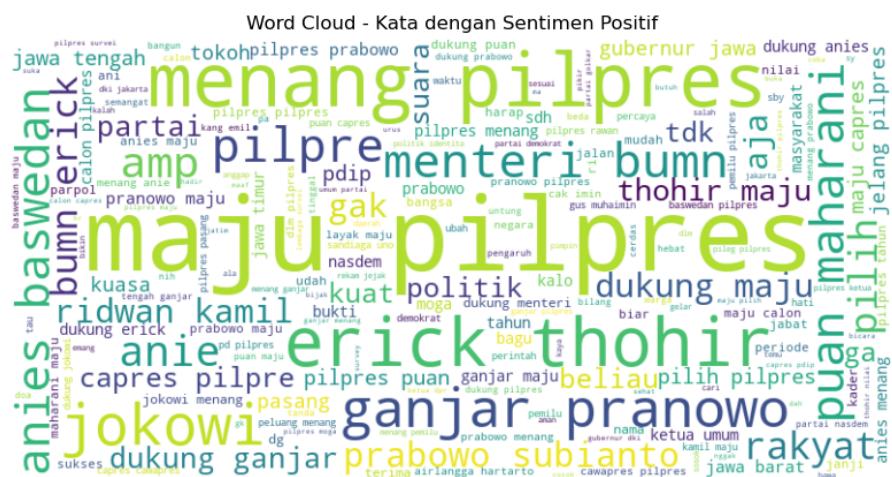


Gambar 4. 24 *Pie chart* pembagian sentimen

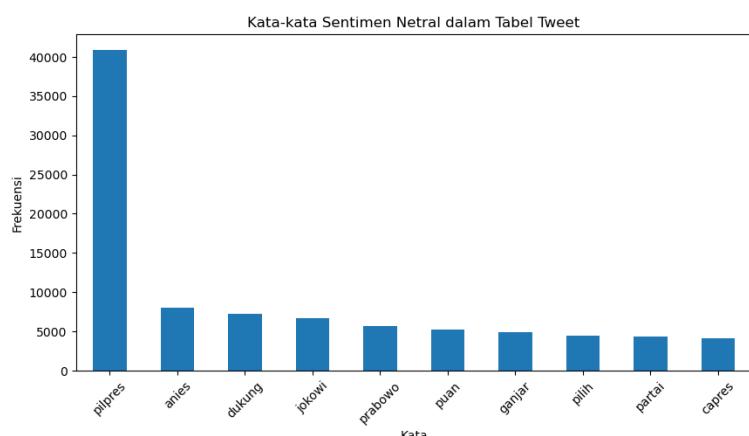
Pembagian sentimen sudah dilakukan, untuk mengetahuui persebaran kalimat yang ada pada sentimen maka peneliti melakukan visualisasi data terhadap nilai nilai label sentimen, serta jenis visualisasi data yang digunakan masing masing label sentimen ialah *barplot* dan *wordcloud*. Nilai sentimen positif ada pada gambar 4.25 dan gambar 4.26, nilai sentimen netral ada pada 4.27 dan gambar 4.28, dan nilai sentimen negatif ada pada gambar 4.29 dan 4.30.



Gambar 4. 25 Bar plot kalimat positif



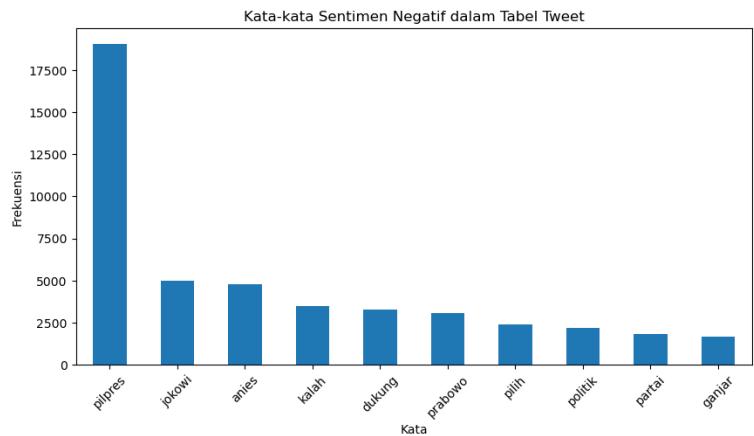
Gambar 4. 26 Wordcloud kalimat positif



Gambar 4. 27 Bar plot kalimat netral



Gambar 4. 28 Wordcloud kalimat netral



Gambar 4. 29 Bar plot kalimat negatif



Gambar 4. 30 Wordcloud kalimat negatif

IV.1.4 Modeling

Model yang akan digunakan yakni *Naive Bayes Classifier*, yang dasarnya diambil dari bayes theorem, rumusnya dapat dilihat pada persamaan(1). Model ini sangatlah terkenal pada proses klasifikasi, selain sentimen analisis, model ini biasa digunakan pada forecasting cuaca. Proses Modelling dapat dilihat pada gambar 4. 31

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, f1_score, precision_score, recall_score
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import Pipeline
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

vectorizer = CountVectorizer()

# Melakukan transformasi teks menjadi vektor fitur
X = vectorizer.fit_transform(data['tweet'].astype(str))
# Y = vectorizer.fit_transform(data['sentiment'])

epoch = 1
accuracy_values = np.zeros(epoch)
# Membagi data menjadi data latih dan data uji
for epochs in range(epoch):
    X_train, X_test, y_train, y_test = train_test_split(X, data['sentiment'], test_size=0.2, random_state=42)
    naive_bayes = MultinomialNB()
    training = naive_bayes.fit(X_train, y_train)
    prediction = naive_bayes.predict(X_test)

    accuracy = accuracy_score(y_test, prediction)
    accuracy_values[epochs] = accuracy

    print("epoch:", epochs + 1, "acc:", accuracy)
    plt.plot(range(1,epoch + 1), accuracy_values, marker='o')
    plt.xlabel('epoch')
    plt.ylabel('Accuracy')
    plt.title('ACC per Epoch')
    plt.show()

y_pred = naive_bayes.predict(X_test)

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

Gambar 4. 31 Modelling

Pada proses ini penulis membuat data latih sebanyak 80% dari banyaknya *dataset* yang ada, yang diambil secara acak menggunakan parameter *random_state*. Maka *X_train* sebanyak 80% label *X* dan 20% sisanya menjadi *X_test*, hal ini juga berlaku pada *y_train* dan *y_test*. Output dari training data uji menghasilkan akurasi sebesar 71.32%. Dapat dilihat pada gambar 4. 32 yang menunjukkan hasil pelatihan *dataset* dengan model *Naive Bayes Classifier*.

Akurasi: 71.32%

Gambar 4. 32 Hasil Training

Contoh perhitungan Naive Bayes Classifier:

Kalimat yang ingin diuji yakni "prabowo yakin mendapatkan kemenangan", Kalimat tersebut apabila setelah di *stemming* dan *stopword removal* menjadi "prabowo menang". Seperti yang sudah dijelaskan pada persamaan(1) maka rumus untuk mencari sentimen kalimat yakni:

Positif:

$$P(kalimat|positif) = \frac{P(positif|kalimat)P(positif)}{P(kalimat)}$$

Negatif:

$$P(kalimat|netral) = \frac{P(netral|kalimat)P(netral)}{P(kalimat)}$$

Netral:

$$P(kalimat|negatif) = \frac{P(negatif|kalimat)P(negatif)}{P(kalimat)}$$

Contoh perhitungan :

$$P(netral) = 32555/75946 = 0,43$$

$$P(positif) = 28566/75946 = 0,38$$

$$P(negatif) = 13825/75946 = 0,19$$

$$P(netral|prabowo) = 3930 / 9138 = 0,43$$

$$P(positif|prabowo) = 3270/9138 = 0,36$$

$$P(negatif|prabowo) = 1938/9138 = 0,21$$

$$P(netral|menang) = 1386/9240 = 0,15$$

$$P(positif|menang) = 7095/9240 = 0,77$$

$$P(negatif|menang) = 759/9240 = 0,08$$

$$\begin{aligned}
p(kalimat) &= (p(net)*p(kalimat|net)) + (p(pos)*p(kalimat|pos)) + \\
&\quad (p(neg)*p(kalimat|neg)) \\
&= (0,43*0,43*0,15) + (0,38*0,36*0,77) + (0,19*0,21*0,08) \\
&= 0,14
\end{aligned}$$

Contoh perhitungan positif:

$$\begin{aligned}
P(kalimat|positif) &= \frac{P(positif|prabowo)P(positif|menang)P(positif)}{P(kalimat)} \\
P(kalimat|positif) &= \frac{0,36 * 0,77 * 0,38}{0,14} \\
P(kalimat|positif) &= 0,75
\end{aligned}$$

Contoh perhitungan netral:

$$\begin{aligned}
P(kalimat|netral) &= \frac{P(netral|prabowo)P(netral|menang)P(netral)}{P(kalimat)} \\
P(kalimat|netral) &= \frac{0,43 * 0,15 * 0,43}{0,14} \\
P(kalimat|netral) &= 0,22
\end{aligned}$$

Contoh perhitungan negatif:

$$\begin{aligned}
P(kalimat|negatif) &= \frac{P(negatif|prabowo)P(negatif|menang)P(negatif)}{P(kalimat)} \\
P(kalimat|negatif) &= \frac{0,21 * 0,08 * 0,19}{0,14} \\
P(kalimat|negatif) &= 0,03
\end{aligned}$$

Dari perhitungan diatas bisa dilihat jika sentimen positif memiliki 0,75 atau 75% dari kalimat "prabowo menang" memiliki sentimen positif, sedangkan untuk nilai negatif memiliki 3% dan netral sebesar 22%.

IV.1.5 *Evaluation*

Monte-Carlo Cross Validation digunakan sebagai evaluasi dari *dataset*, MCCV (*Monte-Carlo Cross Validation*) merupakan pengujian setiap data uji dan data latih

yang diambil secara acak dan dicari akurasinya, untuk code MCCV dapat dilihat pada gambar 4. 33.

```
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score
shuffle_split = ShuffleSplit(n_splits=150, test_size=0.2, random_state=42)

# Melakukan cross-validation dengan ShuffleSplit
scores = cross_val_score(naive_bayes, X, data['sentiment'], cv=shuffle_split, scoring='accuracy')
x = []
# Menampilkan skor akurasi untuk setiap iterasi cross-validation
for i, score in enumerate(scores):
    print(f"Iterasi {i+1}: {score}")
    x.append(score)

# Menampilkan rata-rata skor akurasi dari cross-validation
print(len(x))
print("Rata-rata skor akurasi: {:.2f} %".format(scores.mean()*100))
print("Skor tertinggi: {:.2f} %".format(scores.max()*100))
plt.plot(range(len(x)), x, marker='o')
```

Gambar 4. 33 Monte-Carlo Cross Validation

Dapat dilihat pada variabel *shullfe_split* saya mendeklarasikan sebanyak 150 kali acak dengan pengambilan data uji sebanyak 20% dan random pengambilan sebanyak 42. Variabel score memanggil *cross_val_score* yang mana memanggil *cv=shuffle_split* yakni metode *cross validation* menggunakan *shuffle_split*. Hasil dari evaluasi ini dapat dilihat pada Tabel dibawah ini.

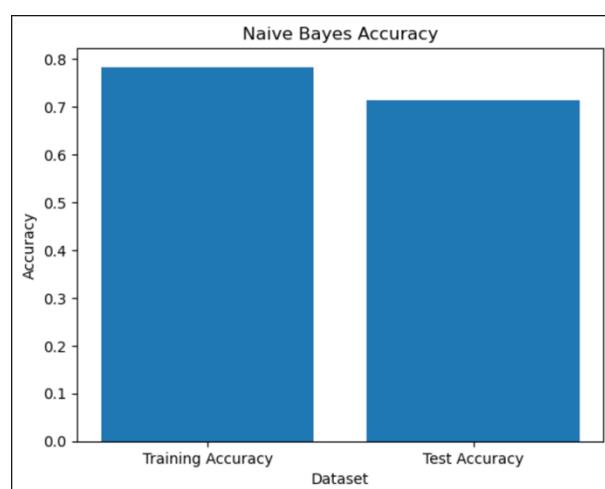
Tabel 4. 1 Hasil evaluasi model dengan MCCV

Iterasi ke-n	Akurasi
1-10	71,27 %
11-20	71,34 %
21-30	71,32 %
31-40	71,29 %
41-50	71,31 %
51-60	71,29 %
61-70	71,34 %
71-80	71,33 %
81-90	71,32 %

Iterasi ke-n	Akurasi
91-100	71,24 %
101-110	71,30 %
111-120	71,45 %
121-130	71,37 %
131-140	71,43 %
141-150	71,19 %
Mean	71.32%

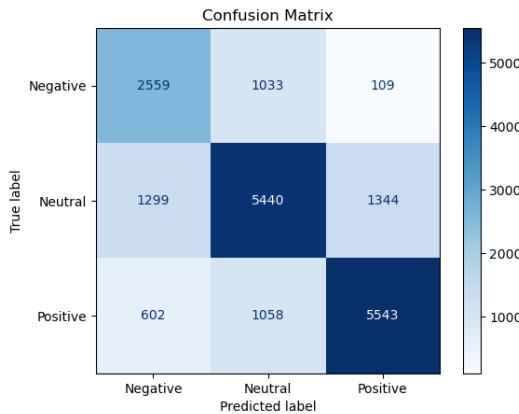
Hasil dari evaluasi ini cukup baik, karena tidak terlalu jauh dengan akurasi pada modelling, jika hasil jauh, seperti 60% maka bisa dicurigai data tersebut *underfitting*.

Selain melakukan evaluasi dengan MCCV, kita juga harus membandingkan hasil prediksi data latih dan data uji, hasil dari pengujian data latih menghasilkan akurasi sebesar 78,30% tidak jauh dengan data uji yakni 71,32%, data dinyatakan overfit jika data latih dan data uji memiliki jarak yang relatif jauh bisa hingga selisih 20%. gambar barplot perbedaan data uji dan data latih ada pada gambar 4. 34



Gambar 4. 34 Perbandingan data uji dan data latih

Selain menggunakan MCCV pada penelitian ini juga menggunakan *confusion matrix* untuk evaluasi dari model *machine learning* ini, berikut gambar *confusion matrix* dari model *naive bayes*.



Gambar 4. 35 *Confusion matrix*

Dari gambar 4.35 kita mencari akurasi menggunakan rumus persamaan (2)

Akurasi:

$$\text{Akurasi} = \frac{TP + TN}{TP + FP + FN + TN}$$

$$= \frac{2559 + 5440 + 5543}{2559 + 1033 + 109 + 1299 + 5440 + 1344 + 602 + 1058 + 5543}$$

$$= 0.71322483804$$

dari perhitungan diatas maka didapat akurasi sebesar 0.7132 atau 71,32%, selanjutnya kita mencari *recall* dengan persamaan (3), pada perhitungan *recall* kita harus mencari *recall* setiap label yakni positif, netral dan negatif.

Recall negatif :

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$= \frac{2559}{2559 + 1033 + 109}$$

$$= 0.69143474736$$

Recall positif :

$$\frac{TP}{TP + FN}$$
$$\frac{5543}{5543 + 1058 + 602}$$
$$0.76954046924$$

Recall netral :

$$\frac{TP}{TP + FN}$$
$$\frac{5440}{5440 + 1299 + 1344}$$
$$0.67301744401$$

jika kita jumlahkan seluruhnya dan dibagi tiga maka akan didapat nilai rata rata yakni 0.7113, yang berarti model ini dapat mengidentifikasi positif sentimen negatif, positif dan netral sebesar 71,13% *macro average* jika kita mencari *weighted average* maka nilainya menjadi 0,7132 atau 71.32%, selanjutnya kita mencari nilai *precision* dengan rumus persamaan (4), seperti halnya *recall*, *precision* juga harus dicari dengan cara satu persatu.

Precision negatif :

$$\frac{TP}{TP + FP}$$
$$\frac{2559}{2559 + 1299 + 602}$$
$$0.57376681614$$

Precision positif :

$$\frac{TP}{TP + FP}$$
$$\frac{5543}{5543 + 1344 + 109}$$
$$0.79230989136$$

Precision netral :

$$\frac{TP}{TP + FP}$$
$$\frac{5440}{5440 + 1058 + 1033}$$
$$0.72234762979$$

dari ketiga data diatas kita mendapat kan nilai rata rata 0.70 yang berarti model ini dapat mengklasifikasikan data sebagai nilai sentimen yang tepat sebesar 70% *macro average* jika kita mencari *weighted average* maka 0.7199 atau 71.99%, selanjutnya kita mencari nilai harmoni antara *precision* dan *recall* setiap nilai sentimen, rumus *f1-score* sudah di deskripsikan pada persamaan (5)

F1-score negatif :

$$\frac{2 \times recall \times precision}{recall + precision}$$
$$\frac{2 \times 0.69 \times 0.57}{0,69 + 0,57}$$
$$0,63$$

F1-score positif :

$$\frac{2 \times recall \times precision}{recall + precision}$$
$$\frac{2 \times 0.79 \times 0.77}{0,79 + 0,77}$$
$$0,78$$

F1-score netral :

$$\frac{2 \times recall \times precision}{recall + precision}$$
$$\frac{2 \times 0.72 \times 0.67}{0,72 + 0,67}$$
$$0,70$$

dari perhitungan diatas, kita akan mencari *macro average* dari model ini yakni 0.70 atau sebesar 70% jika mencari nilai *weighted average* maka akan mendapat nilai 0.7151 atau 71.51%. setelah kita melakukan perhitungan dengan *confusion matrix* maka didapat akurasi sebesar 71.32%, presisi sebesar 71.99%, *recall* sebesar 71.32% dan *f1-score* sebesar 71.51%. Hasil tersebut menunjukkan bahwa model memiliki kinerja yang seimbang dalam mengklasifikasikan ketiga kelas.

IV.1.6 Deployment

Deployment pada penelitian ini yakni menggunakan streamlit sebagai library untuk membuat web sederhana dengan python yang bermodalkan format pickle dari data latih. *Code* pembuatan model dapat dilihat pada gambar 4.36.

```
model_data = {
    'model': naive_bayes,
    'vectorizer': vectorizer
}

with open('trained_model1.pkl', 'wb') as file:
    pickle.dump(model_data, file)
```

Gambar 4. 36 Save model format Pickle

Variabel yang ada pada `model_data` dapat dilihat pada gambar 4.16 dimana `naive_bayes` ialah *MultinomialNB* pada fungsi *modelling* dan `vectorizer` pada model menggunakan *CountVectorizer*.

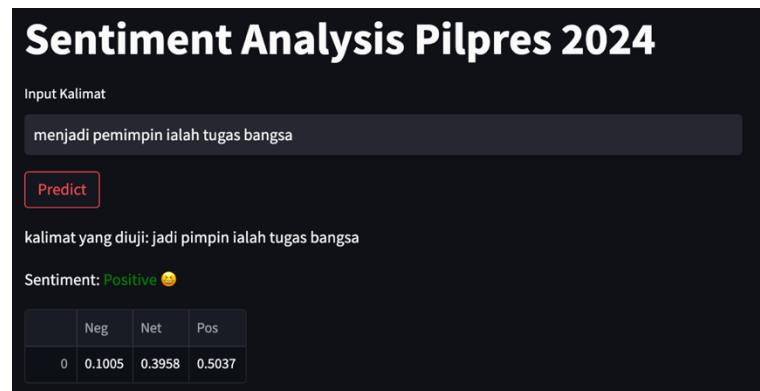
Selanjutnya ialah membuat web sederhana dengan library streamlit yang didukung oleh bahasa pemrograman python, untuk lebih jelas bisa dilihat pada gambar 4.37.

```
st.title("Sentiment Analysis Pilpres 2024")

# Input teks
text = st.text_input("Input Kalimat")
kolom = ['Neg', 'Net', 'Pos']
# Tombol untuk melakukan prediksi
if st.button("Predict"):
    if text:
        sentiment = predict_sentiment(text)
        st.write("kalimat yang diuji:", sentiment[2])
        df = pd.DataFrame(sentiment[1], columns=kolom)
        if sentiment[0] == 'Positive':
            st.write("Sentiment:", "<span style='color:green;'>Positive 😊</span>", unsafe_allow_html=True)
            st.write(df)
        elif sentiment[0] == 'Negative':
            st.write("Sentiment:", "<span style='color:red;'>Negative 😞</span>", unsafe_allow_html=True)
            st.write(df)
        else:
            st.write("Sentiment:", "<span style='color:white;'>Neutral 😐</span>", unsafe_allow_html=True)
            st.write(df)
    else:
        st.write("Please input text.")
```

Gambar 4. 37 Code streamlit

Gambar diatas menunjukkan model di *import*, lalu *load value* yang ada pada model tersebut yakni model dan *vectorizer*. Gambaran *output* dari *code* streamlit dapat dilihat pada gambar 4.38 yang menjelaskan gambar positif, 4.39 menjelaskan gambar negatif dan 4.40 menjelaskan gambar netral.



Gambar 4. 38 Hasil positif



Gambar 4. 39 Hasil negatif



Gambar 4. 40 Hasil netral

IV.2 Pengujian

Pada tahap ini penulis mencoba 20 kalimat di inputkan pada model *Naive Bayes* yang sudah dilatih, hasilnya dapat dilihat pada tabel dibawah ini:

Tabel 4. 2 Pengujian sentimen dari *tweet*

No.	Kalimat	Probabilitas Negatif	Probabilitas Netral	Probabilitas Positif	Nilai Sentiment
1.	Masyallah Capres saya emang keren, salah satu putra terbaik bangsa.. Semoga Allah meridhoi Pak Anies menang di Pilpres 2024 nanti, Aamiin Allahumma Aamiin 🙏❤️	0	0.00001	0.9999	Positif
2.	Saya berpendapat Presiden Joko Widodo sudah layak menjalani proses pemeriksaan pemakzulan (impeachment) karena sikap tidak netralnya sudah nampak di depan mata alias cawe-cawe dalam Pilpres 2024	0.0813	0.849	0.0697	Netral
3.	untuk apa menjegal Anies semua tahu anies tidak punya prestasi	0.2487	0.5318	0.2195	Netral
4.	Kalau seperti ini masih percaya akan jurdil pilpres 2024...???	0.1807	0.4478	0.3716	Netral

No.	Kalimat	Probabilitas Negatif	Probabilitas Netral	Probabilitas Positif	Nilai Sentiment
5.	Gak perlu iri lah kalian ya drun... Orang waras dan cerdas pasti akan memilih pak Ganjar Pranowo di pilpres 2024... 😊💪😉	0.0642	0.1388	0.797	Positif
6.	Menurut Ratna Sarumpaet, Hoax tersebut sengaja dibuat dan disebarluaskan oleh kubu Prabowo untuk merusak nama baik presiden Jokowi yang saat itu bersaing di pilpres	0.9948	0.0044	0.0007	Negatif
7.	Prabowo Subianto CAPRES 2024 dari partai GERINDRA peraih suara no 2 terbesar	0.0138	0.6569	0.3294	Netral
8.	Maju terus pak Anies hanya orang yang cerdas dan bijaksana yg pantas memimpin bangsa ini👉	0.0004	0.0012	0.9984	Positif
9.	Kadrun2 pendukung Anies ini panik liat elektabilitas Anies yg rendah sedangkan Pilpres tinggal sebentar lagi, makanya mereka lakukan segala cara untuk melengserkan Jokowi sebelum Pilpres	0.8391	0.1563	0.0046	Negatif

No.	Kalimat	Probabilitas Negatif	Probabilitas Netral	Probabilitas Positif	Nilai Sentiment
10.	Cegah Kecurangan Pilpres, Gerakan Nasional Anies Presiden 2024 Gelar Bimtek dan Mitigasi	0.3353	0.4162	0.2485	Netral
11.	Bacapres PDIP Ganjar udah nampak di bawah kendali para oligarki keturunan China Jatim termasuk Teguh Kinarto, bos properti yg baru diperiksa KPK terkait dugaan korupsi	0.5309	0.3647	0.1044	Negatif
12.	Masak gak kapok milih PDIP...kalo aku gak mau	0.5217	0.4464	0.0319	Negatif
13.	Pedeipe isinya maling semua Partai Koruptor	0.5188	0.355	0.1262	Negatif
14.	Gugatan judicial review agar diberlakukan sistem pemilu tertutup untuk Pemilu 2024 sebelumnya dilayangkan PDI Perjuangan pada November 2022 lalu	0.0009	0.9978	0.0013	Netral
15.	Itu yang sering disebut main 2 kaki. Ketika ada 2 Menteri dari Partai Gerindra, ada @fadlizon yang bebas menghina Pak	0.8325	0.0481	0.1194	Negatif

No.	Kalimat	Probabilitas Negatif	Probabilitas Netral	Probabilitas Positif	Nilai Sentiment
	Presiden @jokowi . Lagipula hingga kini blm pernah ada pendapat Pak Prabowo pribadi mengenai HTI & FPI? Juga nasib kedua organisasi itu jika ia berkuasa				
16.	Andai benar Demokrat merapat ke PDIP dan meninggalkan Koalisi Perubahan, namun di sisi lain Koalisi Perubahan merapat ke Gerindra dengan memasang Anies jadi cawapres, maka Prabowo-Anies akan sangat mungkin mengalahkan Ganjar	0.0233	0.8196	0.1571	Netral
17.	Maju terus Nasdem dengan ketegasan pak Surya Paloh....kuat sikap teguh pendirian meskipun di hadapkan posisi sulit ... "Sekali Layar Berkembang , Surut Kita Berpantang "	0.0013	0.1012	0.8975	Positif
18.	Nasdem telah mengambil keputusan yang Fatal 😞	0.9475	0.0484	0.0041	Negatif

No.	Kalimat	Probabilitas Negatif	Probabilitas Netral	Probabilitas Positif	Nilai Sentiment
19.	Jelang Pemilu 2024 jaga keutuhan bangsa Indonesia	0.2428	0.392	0.3652	Netral
20.	Kesuksesan Pemilu 2024 adalah tanggungjawab bersama, termasuk para pemilih muda yang menjadi pemilih pemula	0.0024	0.0193	0.9783	Positif

BAB V SIMPULAN DAN SARAN

V.1 Simpulan

Setelah melakukan penelitian ini dapat dilihat warganet pada *twitter* memiliki sentimen positif, sentimen negatif dan sentimen netral pada setiap cuitan yang dilakukan dengan data sebanyak 124544 (seratus dua puluh empat lima ratus empat puluh empat) data yang dihimpun sejak Januari 2022 hingga Desember 2022 dengan pembagian *dataset* sebesar 80% untuk data latih dan 20% menjadi data uji, dapat disimpulkan sebagai berikut:

1. Didapatkan akurasi sebesar 71,32% menggunakan algoritma *naive bayes classifier*, setelah dilakukan evaluasi menggunakan MCCV, akurasi yang didapat yakni 71,32%. Sehingga juru kampanye dapat menggunakan *machine learning* ini.
2. Juru kampanye bisa melakukan uji sentimen dengan web, yang dapat diakses pada: <https://sentimenpilpres2024.streamlit.app/>

V.2 Saran

Saran yang ingin disampaikan penulis ialah:

1. Melakukan penelitian dengan data baru (setelah pemilu dilaksanakan).
2. Memuat *dataset* lebih banyak lagi agar akurasi dapat mencapai 76% keatas.
3. Adanya *library* normalisasi untuk membersihkan kalimat *slang* berbahasa Indonesia.
4. memberikan labeling pada *dataset* yang konstektual.
5. Membuat algoritma automasi untuk scrapping data yang terlambat jauh karena *twitter* API hanya mengizinkan scrapping data seminggu yang lampau.

DAFTAR PUSTAKA

- Chakraborty, K., Bhatia, S., Bhattacharyya, S., Platos, J., Bag, R. and Hassanien, A.E., 2020. Sentiment Analysis of COVID-19 tweets by Deep Learning Classifiers—A study to show how popularity is affecting accuracy in social media. *Applied Soft Computing*, 97, p.106754.
- Mehta, P. and Pandya, S., 2020. A review on sentiment analysis methodologies, practices and applications. *International Journal of Scientific and Technology Research*, 9(2), pp.601-609.
- Han, J., Kamber, M., & Pei, J. (2012). Data Mining Concepts and Techniques (3rd ed). USA: Elsevier Inc.
- Xu, P., Song, Z., Yin, Q., Song, Y.Z. and Wang, L., 2020. Deep self-supervised representation learning for free-hand sketch. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(4), pp.1503-1513.
- Altamevia, F., Wijaya, H.O.L. and Elmayati, E., 2023. Analisis Pola Penjualan Obat di Apotek Srikandi Menggunakan Algoritma Supervised Learning. *Kesatria: Jurnal Penerapan Sistem Informasi (Komputer dan Manajemen)*, 4(1), pp.170-176.
- Enterprise, J., 2019. *Python untuk Programmer Pemula*. Elex media komputindo.
- I. H. Witten, “Text mining,” *Pract. Handb. Internet Comput.*, pp. 14-1-14–22, 2004, doi: 10.1201/9780203507223
- B. Priyono, “Pengenalan dan Panduan Jupyter Notebook untuk Pemula,” 2019. [Online]. Available: <https://indoml.com/2019/09/29/pengenalan- dan-panduan-jupyter-notebook-untuk-pemula/>. [Accessed: 23-Apr-2023].
- Haddad, K. and Rahman, A., 2020. Regional flood frequency analysis: evaluation of regions in cluster space using support vector regression. *Natural Hazards*, 102, pp.489-517.
- Schröer, C., Kruse, F. and Gómez, J.M., 2021. A systematic literature review on applying CRISP-DM process model. *Procedia Computer Science*, 181, pp.526-534.

- Rawat, A., 2020. A Review on Python Programming. International Journal of Research in Engineering, Science and Management, 3(12), pp.8-11.
- Xu, J., Zhang, Y. and Miao, D., 2020. Three-way confusion matrix for classification: A measure driven view. *Information sciences*, 507, pp.772-794.
- Patro, V.M. and Patra, M.R., 2014. Augmenting weighted average with confusion matrix to enhance classification accuracy. *Transactions on Machine Learning and Artificial Intelligence*, 2(4), pp.77-91.
- Deng, X., Liu, Q., Deng, Y. and Mahadevan, S., 2016. An improved method to construct basic probability assignment based on the confusion matrix for classification problem. *Information Sciences*, 340, pp.250-261.
- Hickman, L., Thapa, S., Tay, L., Cao, M. and Srinivasan, P., 2022. Text preprocessing for text mining in organizational research: Review and recommendations. *Organizational Research Methods*, 25(1), pp.114-146.
- VM, N. and Kumar R, D.A., 2019, May. Implementation on text classification using bag of words model. In Proceedings of the second international conference on emerging trends in science & technologies for engineering systems (ICETSE-2019).
- Qader, W.A., Ameen, M.M. and Ahmed, B.I., 2019, June. An overview of bag of words; importance, implementation, applications, and challenges. In *2019 international engineering conference (IEC)* (pp. 200-204). IEEE.

LAMPIRAN A: PYTHON NOTEBOOK SCRAPPING

```
!pip install git+https://github.com/JustAnotherArchivist/snscreape.git

import pandas as pd

from tqdm.notebook import tqdm

import snscreape.modules.twitter as sntwitter

scraper = sntwitter.TwitterSearchScraper('pilpres 2024 OR prabowo OR anies OR
ganjar since:2022-01-01 until:2022-12-31')

for tweet in scraper.get_items():

    break

    tweet

tweets = []

n_tweet = 1000000

for i, tweet in tqdm(enumerate(scraper.get_items()),total=n_tweet):

    data = [tweet.date, tweet.renderedContent, tweet.user.username]

    tweets.append(data)

    if i > n_tweet:

        break

tweet_df = pd.DataFrame(tweets, columns=['date','tweet','username'])

tweet_df.to_csv('2022pilpres.csv')
```

LAMPIRAN B: PYTHON NOTEBOOK DATA UNDERSTANDING

```
data = pd.read_csv('gab.csv')

print('data ada sebanyak',len(data),'baris')

data = data.drop(['Unnamed: 0'], axis = 1)

print(data.columns)

print(data.dtypes)

duplikasi = len(data.tweet)-len(data.tweet.drop_duplicates())

print(f'ada sebanyak {duplikasi} data yang terduplicasi')

x = data.groupby(data.tweet.tolist(), as_index=False).size()

x.sort_values(by='size',ascending=False)

a = (x['size'] > 1).sum()

print(f'jadi ada tweet sebanyak {a} yang memiliki cuitan yang sama dengan total {duplikasi} tweet')

data.isna().sum()

tnull_rows = data[data['tweet'].isnull()]

print(tnull_rows)

unull_rows = data[data['username'].isnull()]

print(unull_rows)
```

```
data = data.dropna()
```

```
data.isna().sum()
```

```
data = data.drop_duplicates(subset=['tweet'])
```

```
len(data)
```

```
import pandas as pd
```

```
from wordcloud import WordCloud
```

```
import matplotlib.pyplot as plt
```

```
all_words = ''.join(str(data['tweet']))
```

```
word_counts = data['tweet'].str.split(expand=True).stack().value_counts()
```

```
wordcloud = WordCloud(width=800, height=400,  
background_color='white').generate_from_frequencies(word_counts)
```

```
plt.figure(figsize=(10, 5))
```

```
plt.imshow(wordcloud, interpolation='bilinear')
```

```
plt.axis('off')
```

```
plt.title('Word Cloud - Kata-Kata Terbanyak')
```

```
plt.show()
```

```
word_counts = data['tweet'].str.split(expand=True).stack().value_counts()
```

```
top_words = word_counts.head(10) # Mengambil 10 kata teratas, ganti jumlah  
sesuai dengan kebutuhan Anda
```

```
plt.figure(figsize=(10, 5))
```

```
top_words.plot(kind='bar')

plt.xlabel('Kata')

plt.ylabel('Frekuensi')

plt.title('Kata-Kata Terbanyak')

plt.show()
```

```
username = data.username.value_counts()

username.head(10)
```

```
username_top = username.head(15)

plt.figure(figsize=(10, 5))

username_top.plot(kind='barh')

plt.xlabel('username')

plt.ylabel('Frekuensi')

plt.title('username tweet Terbanyak')

plt.show()
```

LAMPIRAN C: PYTHON NOTEBOOK DATA PREPARATION

```
def preprocess_tweet2(tweet):

    EMOJI_PATTERN = re.compile(
        "["
        "\U0001F1E0-\U0001F1FF" # flags (iOS)
        "\U0001F300-\U0001F5FF" # symbols & pictographs
        "\U0001F600-\U0001F64F" # emoticons
        "\U0001F680-\U0001F6FF" # transport & map symbols
        "\U0001F700-\U0001F77F" # alchemical symbols
        "\U0001F780-\U0001F7FF" # Geometric Shapes Extended
        "\U0001F800-\U0001F8FF" # Supplemental Arrows-C
        "\U0001F900-\U0001F9FF" # Supplemental Symbols and Pictographs
        "\U0001FA00-\U0001FA6F" # Chess Symbols
        "\U0001FA70-\U0001FAFF" # Symbols and Pictographs Extended-A
        "\U00002702-\U000027B0" # Dingbats
    "])")

    tweet = re.sub(r'[0-9]+,", str(tweet))

    tweet = tweet.lower() # convert to lower case

    tweet = re.sub(r"http\S+|www\S+|https\S+", "", tweet, flags=re.MULTILINE) # remove URLs

    tweet = re.sub(r'@\w+|\#\w+', "", tweet) # remove mentions and hashtags

    tweet = re.sub(r'\d+', "", tweet) # remove numbers

    tweet = re.sub(r'!+', tweet) #

    tweet = tweet.translate(str.maketrans("", "", string.punctuation)) # remove punctuations

    tweet = tweet.strip()
```

```

tweet = re.sub(EMOJI_PATTERN, r'', tweet)

tweet = re.sub(r'\n+', " ", tweet)

tweet = re.sub(r'^\s+', " ", tweet)

tweet = re.sub(r'\brt', " ", tweet)

return tweet

data['tweet']=data['tweet'].apply(preprocess_tweet2)

from nltk.tokenize import word_tokenize

def tokenize_column(text):

    if isinstance(text, str): # Memastikan bahwa text adalah string

        return word_tokenize(text)

    else:

        return [] # Mengembalikan list kosong jika text bukan string

# Contoh penggunaan:

data['tweet'] = data['tweet'].apply(tokenize_column)

stemmer = StemmerFactory().create_stemmer()

def stemming(batch):

    # Menerapkan stemming pada setiap teks dalam batch

    stemmed_batch = []

    unique_token = "aasdpepemilu" # Token unik untuk kata "pemilu"

```

```
for text in batch:
```

```
    text = text.replace("pemilu", unique_token)
```

```
    stemmed_text = stemmer.stem(text) if text.lower() != "pemilu" else text
```

```
    stemmed_text = stemmed_text.replace(unique_token, "pemilu")
```

```
    stemmed_batch.append(stemmed_text)
```

```
# Melakukan penghapusan stopwords pada setiap teks dalam batch
```

```
return stemmed_batch
```

```
# Menambahkan hasil preprocessing ke dalam DataFrame
```

```
data['tweet'] = data['tweet'].apply(stemming)
```

```
# Hasil preprocessing data
```

```
print(data)
```

```
stopword_remover = StopWordRemoverFactory().create_stop_word_remover()
```

```
def stopword(batch):
```

```
# Melakukan penghapusan stopwords pada setiap teks dalam batch
```

```
cleaned_batch = [stopword_remover.remove(text) for text in batch]
```

```
cleaned_batch = [text for text in cleaned_batch if all(word not in text.lower() for word in ['yg', 'dgn', 'kl', 'spt', 'pk', 'tp', 'krn', 'dr', 'utk', 'lg', 'gw', 'si', 'jg', 'jd', 'shg', 'sbg'])]
```

```
return cleaned_batch
```

```
# Menambahkan hasil preprocessing ke dalam DataFrame
```

```

data['tweet'] = data['tweet'].apply(stopword)

# Hasil preprocessing data
print(data)

def to_text(daftar_token):
    tokens = [token for token in daftar_token if token != '']
    kalimat = ' '.join(tokens)
    return kalimat

data['tweet'] = data['tweet'].apply(to_text)

from nltk.tokenize import word_tokenize

def labelling(tweet):
    # tweet = preprocess_tweet(tweet)

    clean_tokens = word_tokenize(tweet) #tokenize

    # clean_tokens = [stemmer.stem(word) for word in tweet_tokens if word not in
    # stopwords] #stemming & stopword removal

    positive_words = open("positive.txt").read().splitlines()

    negative_words = open("negative.txt").read().splitlines()

    positive_count = sum([1 for word in clean_tokens if word in positive_words])

    negative_count = sum([1 for word in clean_tokens if word in negative_words])

    if positive_count > negative_count:
        return 'Positive'

    elif positive_count < negative_count:

```

```
    return 'Negative'

else:
    return 'Neutral'

data['sentiment'] = data['tweet'].apply(labelling)

data.sentiment.value_counts()

dat = len(data.tweet)-len(data.tweet.drop_duplicates())
print(f'ada sebanyak {dat} data yang terduplikasi')

x = data.groupby(data.tweet.to_list(), as_index=False).size()
x.sort_values(by='size', ascending=False)

b = (x['size'] > 1).sum()
print(f'jadi ada tweet sebanyak {b} yang memiliki cuitan yang sama dengan total
{dat} tweet')

data = data.drop_duplicates(subset=['tweet'])

len(data)

data.sentiment.value_counts()

p = 40638
net = 35769
```

```

neg = 18526

labels = ['Positive [' + str(p) + '] 46%', 'Neutral [' + str(net) + '] 33%', 'Negative [' + str(neg) + '] 20%']

size = [p, net, neg]

colors = ["#81F495", "#A9E4EF", "#FF3C38"]

patches, texts = plt.pie(size, colors=colors, startangle=90)

plt.style.use('default')

plt.legend(labels)

plt.title('Number of tweet')

plt.axis('equal')

plt.show()

```

```

positif_word_counts = data[data['sentiment'] == 'Positive']['tweet'].str.split(expand=True).stack().value_counts()

top_pos_words = positif_word_counts.head(10)

plt.figure(figsize=(10, 5))

top_pos_words.plot(kind='bar')

plt.xlabel('Kata')

plt.ylabel('Frekuensi')

plt.title('Kata-kata Sentimen Positif dalam Tabel Tweet')

plt.xticks(rotation=45)

plt.show()

```

```

data_positif = data[data['sentiment'] == 'Positive']

all_words_positif = ' '.join(data_positif['tweet'])

```

```
wordcloud_positif = WordCloud(width=800, height=400,
background_color='white').generate(all_words_positif)
```

```
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud_positif, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud - Kata dengan Sentimen Positif')
plt.show()
```

```
negatif_word_counts = data[data['sentiment'] == 'Negative']['tweet'].str.split(expand=True).stack().value_counts()
top_negatif_words = negatif_word_counts.head(10)

plt.figure(figsize=(10, 5))
top_negatif_words.plot(kind='bar')
plt.xlabel('Kata')
plt.ylabel('Frekuensi')
plt.title('Kata-kata Sentimen Negatif dalam Tabel Tweet')
plt.xticks(rotation=45)
plt.show()
```

```
data_negatif = data[data['sentiment'] == 'Negative']
all_words_neg = ''.join(data_negatif['tweet'])

wordcloud_neg = WordCloud(width=800, height=400,
background_color='white').generate(all_words_positif)
```

```
plt.figure(figsize=(10, 5))

plt.imshow(wordcloud_neg, interpolation='bilinear')

plt.axis('off')

plt.title('Word Cloud - Kata dengan Sentimen Negatif')

plt.show()
```

```
netral_word_counts = data[data['sentiment'] == 'Neutral']['tweet'].str.split(expand=True).stack().value_counts()

top_net_words = netral_word_counts.head(10)

plt.figure(figsize=(10, 5))

top_net_words.plot(kind='bar')

plt.xlabel('Kata')

plt.ylabel('Frekuensi')

plt.title('Kata-kata Sentimen Netral dalam Tabel Tweet')

plt.xticks(rotation=45)

plt.show()
```

```
data_netral = data[data['sentiment'] == 'Neutral']

all_words_net = ' '.join(data_netral['tweet'])

wordcloud_net = WordCloud(width=800, height=400,
                           background_color='white').generate(all_words_positif)

plt.figure(figsize=(10, 5))

plt.imshow(wordcloud_net, interpolation='bilinear')

plt.axis('off')

plt.title('Word Cloud - Kata dengan Sentimen Netral')

plt.show()
```

LAMPIRAN D: PYTHON NOTEBOOK MODELLING

```
from sklearn.naive_bayes import MultinomialNB  
  
from sklearn.model_selection import train_test_split  
  
from sklearn.metrics import classification_report, confusion_matrix,  
accuracy_score, f1_score, precision_score, recall_score, plot_confusion_matrix  
  
from sklearn.feature_extraction.text import CountVectorizer  
  
from sklearn.feature_extraction.text import TfidfVectorizer  
  
from sklearn.pipeline import Pipeline  
  
import numpy as np  
  
import seaborn as sns  
  
import matplotlib.pyplot as plt  
  
  
vectorizer = CountVectorizer()  
  
  
# Melakukan transformasi teks menjadi vektor fitur  
X = vectorizer.fit_transform(data['tweet'].astype(str))  
# Y = vectorizer.fit_transform(data['sentiment'])  
  
  
epoch = 10  
  
accuracy_values = np.zeros(epoch)  
  
# Membagi data menjadi data latih dan data uji  
for epochs in range(epoch):  
  
    X_train, X_test, y_train, y_test = train_test_split(X, data['sentiment'],  
test_size=0.2, random_state=42)  
  
    naive_bayes = MultinomialNB()  
  
    training = naive_bayes.fit(X_train, y_train)
```

```
prediction = naive_bayes.predict(X_test)

accuracy = accuracy_score(y_test, prediction)
accuracy_values[epochs] = accuracy

print("epoch:", epochs + 1, "acc:", accuracy)

plt.plot(range(1,epoch + 1), accuracy_values, marker='o')
plt.xlabel('epoch')
plt.ylabel('Accuracy')
plt.title('ACC per Epoch')
plt.show()
```

```
y_pred = naive_bayes.predict(X_test)

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

LAMPIRAN E: PYTHON NOTEBOOK EVALUATION

```
from sklearn.model_selection import ShuffleSplit

from sklearn.model_selection import cross_val_score

shuffle_split = ShuffleSplit(n_splits=150, test_size=0.2, random_state=42)

# Melakukan cross-validation dengan ShuffleSplit

scores = cross_val_score(naive_bayes, X, data['sentiment'], cv=shuffle_split,
scoring='accuracy')

x = []

# Menampilkan skor akurasi untuk setiap iterasi cross-validation

for i, score in enumerate(scores):

    print(f"Iterasi {i+1}: {score}")

    x.append(score)

# Menampilkan rata-rata skor akurasi dari cross-validation

print(len(x))

print("Rata-rata skor akurasi: {:.2f} %".format(scores.mean()*100))

print("Skor tertinggi: {:.2f} %".format(scores.max()*100))

plt.plot(range(len(x)), x, marker='o')

acc_mnb = accuracy_score(y_test, y_pred)

print("Accuracy: {:.4f}".format(acc_mnb))

conf_matrix_mnb = confusion_matrix(y_test, y_pred)

print("Confusion matrix:\n {}".format(conf_matrix_mnb))
```

```
print(classification_report(y_test, y_pred))

class_names = sorted(data['sentiment'].unique())

disp = plot_confusion_matrix(training, X_test, y_test,
                             display_labels=class_names,
                             cmap=plt.cm.Blues, values_format='d')

disp.ax_.set_title("Confusion Matrix")

print(disp.confusion_matrix)
```

```
plt.show()
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score
```

```
# Melakukan prediksi pada data uji
```

```
y_pred = naive_bayes.predict(X_test)
```

```
# Menghitung dan menampilkan akurasi
```

```
accuracy = accuracy_score(y_test, y_pred)
```

```
print("Akurasi: {:.2f}%".format(accuracy * 100))
```

```
# Menghitung dan menampilkan presisi
```

```
precision = precision_score(y_test, y_pred, average='weighted')
```

```
print("Presisi: {:.2f}%".format(precision * 100))
```

```

# Menghitung dan menampilkan recall

recall = recall_score(y_test, y_pred, average='weighted')
print("Recall: {:.2f}%".format(recall * 100))

# Menghitung dan menampilkan F1-score

f1 = f1_score(y_test, y_pred, average='weighted')
print("F1-score: {:.2f}%".format(f1 * 100))

positive_count = (y_pred == 'Positive').sum()
negative_count = (y_pred == 'Negative').sum()
neutral_count = (y_pred == 'Neutral').sum()
total_count = len(y_pred)

positive_percentage = (positive_count / total_count) * 100
negative_percentage = (negative_count / total_count) * 100
neutral_percentage = (neutral_count / total_count) * 100

# Tampilkan hasil

print("Persentase Klasifikasi:")

print("Positif: {:.1f}%".format(positive_percentage))
print("Negatif: {:.1f}%".format(negative_percentage))
print("Netral: {:.1f}%".format(neutral_percentage))

print("total: {}".format(total_count))

import matplotlib.pyplot as plt

from sklearn.naive_bayes import MultinomialNB

from sklearn.metrics import accuracy_score

```

```
# Melatih model Naive Bayes

model = MultinomialNB()

model.fit(X_train, y_train)

# Memprediksi label pada data latih dan data uji

y_train_pred = model.predict(X_train)

y_test_pred = model.predict(X_test)

# Menghitung akurasi pada data latih dan data uji

akurasi_train = accuracy_score(y_train, y_train_pred)

akurasi_test = accuracy_score(y_test, y_test_pred)

# Membuat grafik akurasi

labels = ['Training Accuracy', 'Test Accuracy']

values = [akurasi_train, akurasi_test]

plt.bar(labels, values)

plt.xlabel('Dataset')

plt.ylabel('Accuracy')

plt.title('Naive Bayes Accuracy')

plt.show()

print(akurasi_train - akurasi_test)

print(akurasi_train)

print(akurasi_test)
```

LAMPIRAN F: PYTHON NOTEBOOK DEPLOYMENT

```
import pickle

model_data = {
    'model': naive_bayes,
    'vectorizer': vectorizer
}

with open('trained_model.pkl', 'wb') as file:
    pickle.dump(model_data, file)

import streamlit as st
import re
import string
import pickle
import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory #stopword remover
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory #stemming

# Load model dan objek CountVectorizer dari file pickle
with open('trained_model.pkl', 'rb') as file:
    model_data = pickle.load(file)
```

```

model = model_data['model']

vectorizer = model_data['vectorizer']

stemmer = StemmerFactory().create_stemmer()

stopword_remover = StopWordRemoverFactory().create_stop_word_remover()

# Fungsi untuk melakukan prediksi sentimen

def preprocess_tweet2(tweet):

    EMOJI_PATTERN = re.compile(
        "["
        "\U0001F1E0-\U0001F1FF" # flags (iOS)
        "\U0001F300-\U0001F5FF" # symbols & pictographs
        "\U0001F600-\U0001F64F" # emoticons
        "\U0001F680-\U0001F6FF" # transport & map symbols
        "\U0001F700-\U0001F77F" # alchemical symbols
        "\U0001F780-\U0001F7FF" # Geometric Shapes Extended
        "\U0001F800-\U0001F8FF" # Supplemental Arrows-C
        "\U0001F900-\U0001F9FF" # Supplemental Symbols and Pictographs
        "\U0001FA00-\U0001FA6F" # Chess Symbols
        "\U0001FA70-\U0001FAFF" # Symbols and Pictographs Extended-A
        "\U00002702-\U000027B0" # Dingbats
    "]")
    tweet = re.sub(r'[0-9]+,", str(tweet))

    tweet = tweet.lower() # convert to lower case

    tweet = re.sub(r"http\S+|www\S+|https\S+", "", tweet, flags=re.MULTILINE) # remove URLs

    tweet = re.sub(r'@\w+|\#\w+', "", tweet) # remove mentions and hashtags

```

```

tweet = re.sub(r'\d+', "", tweet) # remove numbers

tweet = re.sub(r'\.\\', tweet) #

tweet = tweet.translate(str.maketrans("", "", string.punctuation)) # remove
punctuations

tweet = tweet.strip()

tweet = re.sub(EMOJI_PATTERN, r'', tweet)

tweet = re.sub(r'\\n+', "", tweet)

tweet = re.sub(r'^\\s+', "", tweet)

tweet = re.sub(r'\\br', "", tweet)

return tweet

#stemming

def stemming(text):

    # Menerapkan stemming pada setiap teks dalam batch

    unique_token = "aasdpe\\milu" # Token unik untuk kata "pemilu"

    text = text.replace("pemilu", unique_token) # Mengganti kata "pemilu" dengan
    token unik

    stem_text = stemmer.stem(text) if text.lower() not in ["pemilu", "pilpres"] else
    text # Melakukan proses stemming

    stem_text = stem_text.replace(unique_token, "pemilu") # Mengembalikan token
    unik menjadi kata "pemilu"

return stem_text

# Stopword

def stopword(text):

    # Melakukan penghapusan stopwords pada setiap teks dalam batch

```

```

stopwords = ['yg', 'dgn', 'kl', 'spt', 'pk', 'tp', 'krn', 'dr', 'utk', 'lg', 'gw', 'si', 'jg', 'jd',
'shg', 'sbg']

stopwords = set(stopwords)

words = text.split()

cleaned_words = [word for word in words if word.lower() not in stopwords]

cleaned_text = ' '.join(cleaned_words)

return cleaned_text


def predict_sentiment(text):

    clean = preprocess_tweet2(text)

    stem_text = stemming(clean)

    clean_text = stopword(stem_text)

    text_vectorized = vectorizer.transform([clean_text])

    prediction = model.predict(text_vectorized)

    menarik = model.predict_proba(text_vectorized)

    return prediction[0], menarik, clean_text


# Tampilan aplikasi dengan Streamlit

st.title("Sentiment Analysis Pilpres 2024")

```

```

# Input teks

text = st.text_input("Input Kalimat")

kolom = ['Neg','Net','Pos']

# Tombol untuk melakukan prediksi

if st.button("Predict"):

```

```
if text:

    sentiment = predict_sentiment(text)

    st.write("kalimat yang diuji:", sentiment[2])

    df = pd.DataFrame(sentiment[1], columns=kolom)

    if sentiment[0] == 'Positive':

        st.write("Sentiment:", "<span style='color:green;'>Positive 😊</span>", unsafe_allow_html=True)

        st.write(df)

    elif sentiment[0] == 'Negative':

        st.write("Sentiment:", "<span style='color:red;'>Negative 😢</span>", unsafe_allow_html=True)

        st.write(df)

    else:

        st.write("Sentiment:", "<span style='color:white;'>Neutral 😎</span>", unsafe_allow_html=True)

        st.write(df)

    else:

        st.write("Please input text.")
```