BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 BPJS Kesehatan

BPJS Kesehatan adalah lembaga yang menyelenggarakan program jaminan sosial kesehatan di Indonesia. Program ini bertujuan untuk memastikan akses pelayanan kesehatan bagi seluruh warga negara. Peserta membayar iuran berdasarkan prinsip gotong royong, dan BPJS Kesehatan menanggung biaya pelayanan kesehatan di fasilitas tingkat pertama dan rujukan tingkat lanjutan. Meskipun ada beberapa faktor penghambat, program ini berperan penting dalam mewujudkan hak atas pelayanan kesehatan bagi masyarakat (Astuti, 2020),

2.1.2 Analisis Sentimen

Analisis sentimen, juga dikenal sebagai *opinion mining*, merupakan bidang penelitian yang memanfaatkan teknik pemrosesan bahasa alami (NLP) untuk mengidentifikasi, mengekstrak, mengukur, dan mempelajari secara terstruktur keadaan emosional dan informasi subjektif dalam teks. Secara sederhana, tujuan dari analisis sentimen adalah untuk memahami dan menganalisis pendapat, evaluasi, sikap, penilaian, dan emosi yang terkandung dalam suatu teks (Subarna et al., 2023).

2.1.3 Twitter

Twitter adalah layanan jejaring sosial dan *microblogging* yang memungkinkan penggunanya untuk bertukar pesan singkat yang disebut *"tweet"*

(Karami et al., 2020). *Tweet* memiliki batas karakter maksimum 280 karakter, dan dapat berisi teks, gambar, video, dan tautan. Pengguna dapat mengikuti akun lain untuk melihat tweet mereka, dan dapat berinteraksi dengan tweet dengan menyukai, me-retweet, dan membalasnya.. Twitter didirikan pada tahun 2006 oleh Jack Dorsey, Noah Glass, Biz Stone, dan Evan Williams

2.1.4 Data Mining

Data mining adalah proses analisis data untuk menemukan pola dan hubungan yang signifikan dalam kumpulan data besar. Data mining memanfaatkan teknik dan algoritma canggih untuk menemukan pola tersembunyi dan hubungan yang tidak jelas dalam data yang kompleks, yang berguna dan mendapatkan wawasan dari data (Andrea, 2017).

2.1.5 Text Mining

Text mining merupakan suatu proses analisis yang digunakan untuk mengekstrak informasi yang bermanfaat dari sejumlah dokumen teks. Metode ini melibatkan penerapan algoritma *machine learning* untuk mengidentifikasi pola, tren, serta hubungan yang terdapat dalam data teks tersebut. Dengan menggunakan teknik ini, kita dapat menggali informasi yang bernilai dari dokumen-dokumen teks yang tersedia, memungkinkan kita untuk membuat keputusan yang lebih terinformasi serta mengungkap wawasan yang mendalam dari data yang sebelumnya mungkin tersembunyi (Jan et al., 2020).

2.1.6 *Scraping Data*

Scraping data adalah suatu proses yang melibatkan ekstraksi atau pengumpulan data dari berbagai situs web dengan menggunakan teknik-teknik

khusus. Tujuan utamanya adalah untuk memperoleh data yang berkualitas dari beragam sumber yang tersedia di internet (Chapagain, 2023). Dengan memanfaatkan teknik scraping, peneliti dapat secara otomatis mengumpulkan informasi yang relevan dari situs web.

Hal ini memungkinkan untuk memperoleh data yang luas dan aktual dalam waktu yang singkat, yang dapat digunakan untuk analisis lebih lanjut atau untuk mendukung temuan dalam penelitian. Meskipun *scraping data* menawarkan manfaat yang signifikan dalam hal efisiensi dan akurasi, tetapi juga perlu dicatat bahwa penggunaannya harus memperhatikan etika dan hukum yang berkaitan dengan penggunaan data secara online.

2.1.7 *Text Preprocessing*

Text preprocessing adalah langkah awal dalam *Natural Language Processing* (NLP) yang melibatkan pembersihan dan persiapan data teks untuk diproses lebih lanjut. Proses ini penting karena data teks mentah seringkali mengandung unsur-unsur yang tidak diperlukan atau mengganggu, seperti tanda baca, angka, dan karakter khusus yang dapat mempengaruhi hasil analisis (Vasiliev, 2020). Dengan melakukan *text preprocessing*, data teks menjadi lebih terstruktur dan seragam, memudahkan algoritma NLP untuk mengidentifikasi pola dan melakukan analisis semantik. Proses ini juga membantu dalam mengurangi dimensi data, yang penting untuk model pembelajaran mesin yang efisien. Berikut adalah tahapan utama dalam *Text Preprocessing* :

2.1.7.1 Text Cleaning

Text cleaning adalah proses menghilangkan karakter-karakter yang tidak diinginkan dari teks, seperti tanda baca, angka, karakter spesial, atau tautan web. Tujuan dari tahap ini adalah untuk menyederhanakan teks agar lebih mudah diolah dan dipahami oleh model.

2.1.7.2 Case Folding

Case folding adalah proses mengubah semua huruf dalam teks menjadi huruf kecil atau huruf besar, tergantung pada kebutuhan atau preferensi. Hal ini dilakukan untuk menghindari ambiguitas karena perbedaan huruf besar dan huruf kecil dalam pemrosesan teks oleh model. Contoh proses case folding adalah mengubah "Pagi" menjadi "pagi" atau "MARAH" menjadi "marah".

2.1.7.3 Stopwords Removal

Stopwords adalah kata-kata yang umumnya tidak memiliki makna yang signifikan dalam analisis teks karena kebanyakan muncul secara berulang dalam teks dan tidak memberikan kontribusi yang besar terhadap konten informasi. Contoh *stopwords* dalam bahasa indonesia adalah "yang", "di", "dan", "dari", "ke", dll. Tahap ini melibatkan penghapusan *stopwords* dari teks untuk meningkatkan kualitas representasi teks.

2.1.7.4 Lemmatization

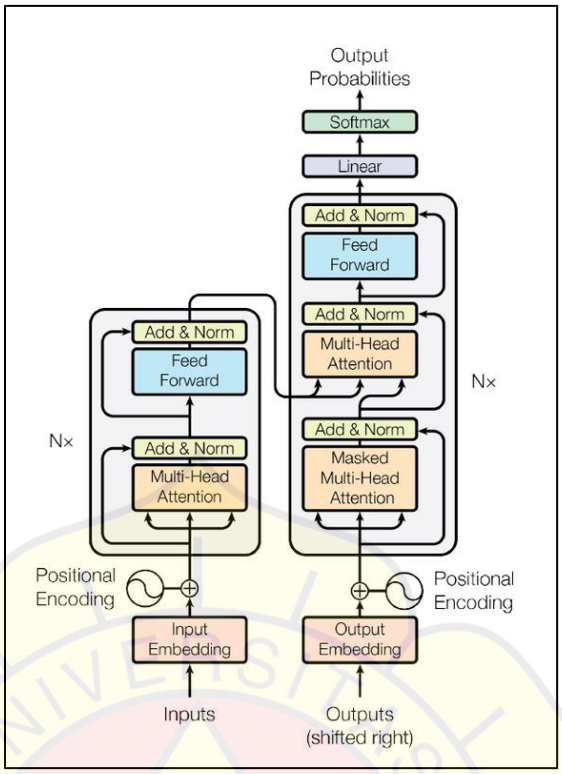
Lemmatization adalah proses mengubah kata-kata dalam teks menjadi bentuk dasarnya atau lemma. Tujuan dari *lemmatization* adalah untuk mengurangi variasi kata dalam teks sehingga kata-kata dengan makna yang sama akan diwakilkan oleh bentuk dasarnya yang benar secara morfologis. *Lemmatization* mempertimbangkan

konteks linguistik dan menggunakan analisis morfologis untuk menentukan bentuk dasar yang tepat.

Contoh *lemmatization* adalah mengubah kata-kata seperti "berlari", "berlari-lari", dan "berlari-lah" menjadi "lari", atau mengubah "lebih baik" dan "terbaik" menjadi "baik" dalam bahasa Indonesia. Berbeda dengan *stemming* yang hanya memotong akhiran, *lemmatization* dapat menghasilkan bentuk dasar yang berbeda sama sekali dari kata aslinya jika diperlukan.

2.1.8 *Transformer*

Transformer merupakan model pembelajaran mendalam yang diperkenalkan dalam jurnal "Attention Is All You Need" oleh (Vaswani et al., 2017). Model ini menggunakan mekanisme perhatian (*attention*) untuk mempelajari hubungan antar kata dalam urutan, tanpa memerlukan arsitektur RNN *(Recurrent Neural Network)* yang kompleks. Mekanisme perhatian memungkinkan *Transformer* untuk fokus pada bagian-bagian urutan yang relevan untuk tugas yang sedang dihadapi. Hal ini membuat *Transformer* lebih efisien dan akurat daripada model sebelumnya, terutama untuk urutan yang panjang. Selain itu, *Transformer* juga lebih mudah untuk diparalelkan, sehingga dapat dilatih dengan lebih cepat dan pada perangkat keras yang lebih besar.



Gambar 2. 1 Arsitektur *Transformer*

Transformer didasarkan pada arsitektur *encoder-decoder*. *Encoder* memproses input teks dan menghasilkan representasi tersembunyi dari setiap kata dalam urutan. *Decoder* kemudian menggunakan representasi ini untuk menghasilkan output teks, satu kata pada satu waktu. Arsitektur *Transformer* terdiri dari beberapa komponen utama:

2.1.8.1 Attention Mechanism

Ini adalah komponen inti dari algoritma *Transformer*. *Attention* memungkinkan model untuk memberikan bobot yang berbeda pada setiap token dalam input, tergantung pada seberapa relevan token tersebut dalam konteks pemrosesan yang sedang dilakukan. Hal ini memungkinkan model untuk "mengambil perhatian" pada bagian-bagian penting dari input. Rumus dasar dari *attention* adalah:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{Dk}}\right)V \quad (1)$$

Dimana :

- Q,K,V adalah matriks representasi dari query, key, dan value.
- Dk adalah dimensi dari key.

2.1.8.2 Multi-Head Attention

Untuk meningkatkan kapasitas dan fleksibilitas model, *attention* sering kali dijalankan beberapa kali secara paralel (*head*). Setiap head memiliki representasi berbeda dari bagaimana attention seharusnya bekerja. Dengan memungkinkan model untuk belajar dari beberapa sudut pandang *attention*, *multi-head attention* dapat meningkatkan kinerja model secara signifikan. Rumusnya adalah:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head1}, \ldots, \text{head}h)W^O \quad (2)$$

$$\text{where head}_i = \text{Attention}(QWi^Q, KWi^K, VWi^V) \quad (3)$$

Dimana :

- $\text{head}_i = \text{Attention}(QWi^Q, KWi^K, VWi^V)$ untuk i = 1, …, h, dengan QWi^Q, KWi^K, VWi^V adalah matriks bobot untuk masing-masing head.
- W^O adalah matriks bobot output untuk menggabungkan hasil dari semua head.

2.1.8.3 Positional Encoding

Karena *Transformer* tidak memiliki komponen yang secara inheren memperhitungkan urutan dalam input, *positional encoding* diperlukan untuk memberikan informasi tentang posisi relatif dari token-token dalam urutan. Ini

dilakukan dengan menambahkan vektor-posisi tertentu ke vektor representasi dari setiap token. Dengan demikian, model dapat memahami urutan dari input. Salah satu rumus yang umum digunakan untuk positional encoding adalah:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2/d}model}\right) \quad (4)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2/d}model}\right) \quad (5)$$

Dimana :

- pos adalah posisi dari token dalam urutan.
- d_{model} adalah indeks dimensi dalam vektor positional encoding.

2.1.8.4 Feed-Forward Network

Setelah melalui mekanisme attention, representasi dari setiap token diumpankan melalui *feed-forward network* (FFN). Ini adalah lapisan jaringan saraf biasa yang terdiri dari beberapa lapisan linear dan fungsi aktivasi *non-linear* seperti *ReLU*. Tujuan dari FFN adalah untuk memproses informasi lebih lanjut secara lokal di setiap token, sehingga memungkinkan model untuk mengekstrak fitur-fitur yang lebih kompleks dari input. Rumus umum untuk feed-forward network adalah:

$$FFN(\chi) = ReLU(\chi W_1 + b_1)W_2 + b_2 \quad (6)$$

Dimana :

- χ adalah input dari feed-forward network.
- W_1 dan b_1 adalah bobot dan bias untuk lapisan linear pertama.
- $W_2 + b_2$ adalah bobot dan bias untuk lapisan linear kedua.

- $ReLU$ adalah fungsi aktivasi ReLU.

2.1.9 *Naive Bayes*

Naive Bayes adalah algoritma klasifikasi berbasis probabilitas yang digunakan dalam machine learning. Metode ini didasarkan pada Teorema Bayes yang memberikan cara untuk menghitung probabilitas posterior dari suatu hipotesis berdasarkan informasi atau bukti yang ada. Algoritma ini disebut "*naive*" karena mengasumsikan bahwa setiap fitur dalam data adalah independen satu sama lain, yang jarang terjadi dalam kasus nyata (Aurélien, 2022). Namun, meskipun asumsi ini sering tidak valid, *Naive Bayes* bekerja dengan baik dalam berbagai aplikasi nyata, terutama dalam klasifikasi teks seperti analisis sentimen dan penyaringan spam. Berikut adalah rumus dasar untuk *naïve bayes* :

$$P(C|X) = \frac{P(X|C).P(C)}{P(X)} \quad (7)$$

Dimana :

- P(C|X) adalah probabilitas kelas C diberikan fitur X (probabilitas posterior).
- P(X|C) adalah probabilitas fitur X diberikan kelas C (likelihood).
- P(C) adalah probabilitas a priori dari kelas C.
- P(X) adalah probabilitas a priori dari kelas X.

Algoritma *Naive Bayes* sering digunakan dalam klasifikasi teks karena kesederhanaannya dan efisiensinya dalam menangani data berukuran besar. Dalam konteks analisis sentimen, setiap kata dalam teks dianggap sebagai fitur, dan model *Naive Bayes* digunakan untuk menghitung probabilitas suatu teks termasuk dalam kategori sentimen tertentu (positif, negatif, atau netral) berdasarkan distribusi kata-kata dalam data pelatihan.

2.1.10 Support Vector Machine (SVM*)*

Support Vector Machine (SVM) adalah algoritma pembelajaran mesin yang digunakan untuk klasifikasi dan regresi. SVM bekerja dengan mencari *hyperplane* optimal yang memisahkan kelas-kelas data dalam ruang fitur (Aurélien, 2022). Hyperplane ini dipilih sedemikian rupa sehingga margin, yaitu jarak antara *hyperplane* dan titik data terdekat dari masing-masing kelas, adalah maksimum. SVM sangat efektif dalam ruang berdimensi tinggi dan sering digunakan dalam klasifikasi teks dan pengenalan pola. Rumus dasar dari SVM adalah :

$$w \cdot x - b = 0 \quad (8)$$

Dimana :

- w adalah vektor bobot.
- x adalah vektor fitur.
- b adalah bias.

SVM juga dapat menggunakan kernel trick untuk memetakan data ke dalam ruang berdimensi lebih tinggi sehingga data yang tidak dapat dipisahkan secara linear dalam ruang asli dapat dipisahkan dalam ruang yang baru. Beberapa kernel yang umum digunakan termasuk linear, polynomial, dan *Radial Basis Function* (RBF). Dalam analisis sentimen, SVM dapat digunakan untuk memisahkan teks berdasarkan sentimen yang diungkapkan, misalnya, positif atau negatif. Algoritma ini terkenal karena kemampuannya untuk menangani data yang tidak terstruktur dan menghasilkan model klasifikasi dengan akurasi tinggi. SVM bekerja dengan baik dalam kondisi di mana jumlah fitur (kata-kata) sangat besar dibandingkan dengan jumlah sampel (teks).

2.1.11 Confusion Matrix

Confusion matrix adalah tabel yang digunakan untuk mengevaluasi kinerja model klasifikasi. Matriks ini memberikan gambaran tentang bagaimana model melakukan prediksi dibandingkan dengan nilai sebenarnya. Dalam kasus klasifikasi biner, confusion matrix terdiri dari empat elemen: True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN) (Tharwat, 2021)

- *Precision*

  Precision mengukur proporsi prediksi positif yang benar-benar positif. Ini dihitung dengan rumus:

  $$\text{Precision} = \frac{\text{True Positive}}{\textit{True Positive} + \textit{False Positive}} \quad (9)$$

- *Recall*

  Recall, juga dikenal sebagai *sensitivity* atau *true positive rate*, mengukur proporsi kasus positif yang berhasil diidentifikasi. Rumusnya adalah:

  $$\text{Precision} = \frac{\text{True Positive}}{\textit{True Positive} + \textit{False}\ \text{Negative}} \quad (10)$$

- *F1- Score*

  F1-Score adalah rata-rata harmonik dari precision dan recall, memberikan satu nilai yang menyeimbangkan kedua metrik tersebut. Rumusnya adalah:

  $$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (11)$$

2.1.12 Cross-Entropy Loss

Cross-Entropy Loss, juga dikenal sebagai *Log Loss*, adalah fungsi kerugian yang umum digunakan dalam masalah klasifikasi, terutama dalam model pembelajaran mesin dan jaringan saraf. Fungsi ini mengukur kinerja model klasifikasi yang outputnya adalah nilai probabilitas antara 0 dan 1. *Cross-Entropy*

Loss berakar dari teori informasi dan konsep entropi, yang mengukur ketidakpastian atau ketidakteraturan dalam suatu sistem (Aggarwal, 2023).

Dalam konteks pembelajaran mesin, *Cross-Entropy Loss* mengukur perbedaan antara distribusi probabilitas yang diprediksi oleh model dan distribusi probabilitas yang sebenarnya (ground truth). Semakin kecil nilai Cross-Entropy Loss, semakin dekat prediksi model dengan distribusi sebenarnya. Rumus untuk menghitung *Cross-Entropy Loss* adalah :

$$L(y, y\hat{}) = \sum_{C=1}^{C} yc \log(yc\hat{}) \quad (12)$$

Dimana :

- y adalah vektor one-hot dari label sebenarnya, dengan nilai 1 pada indeks kelas yang benar dan 0 di tempat lain.
- y^ adalah vektor probabilitas prediksi dari model untuk masing-masing kelas.
- C adalah jumlah kelas (dalam hal ini, C=3).

2.1.13 *Unified Modeling Language*

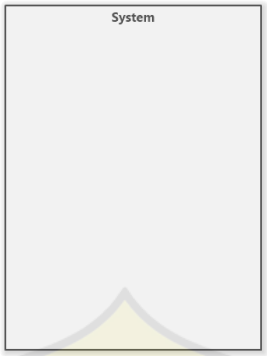
Unified Modeling Language (UML) adalah bahasa pemodelan standar yang digunakan untuk memvisualisasikan, merancang, dan mendokumentasikan sistem perangkat lunak (Sundaramoorthy, 2022). UML menyediakan seperangkat diagram yang dapat digunakan untuk memodelkan berbagai aspek sistem, seperti struktur, perilaku, dan interaksi. UML banyak digunakan dalam industri perangkat lunak untuk membantu pengembang memahami, merancang, dan mengimplementasikan sistem yang kompleks.

2.1.14 *Use case Diagram*

Use Case Diagram adalah diagram *Unified Modeling Language* (UMl) yang digunakan untuk menggambarkan interaksi antara aktor dan system (Sundaramoorthy, 2022). Aktor adalah pihak luar yang berinteraksi dengan sistem, sedangkan sistem adalah kumpulan proses, data, dan perangkat keras yang bekerja sama untuk mencapai tujuan tertentu.

Tabel 2. 1 Komponen Use case Diagram

No.	Komponen	Simbol	Fungsi
1	*Use Case*		*Use case* adalah deskripsi tentang bagaimana sistem akan digunakan oleh para pengguna untuk mencapai tujuan tertentu. Ini membantu dalam pemahaman secara keseluruhan tentang fungsionalitas sistem dari perspektif pengguna.
2	*Actor*		*Actor* adalah entitas (bisa berupa pengguna manusia, sistem eksternal, atau perangkat lain) yang berinteraksi dengan sistem. *Actor* ini memainkan peran

			dalam menjalankan *use case* tertentu.
3	*System Boundary*		*System Boundary* menandai batas antara sistem yang sedang dibahas dan dunia luar. Ini membantu untuk memperjelas apa yang termasuk dalam sistem dan apa yang dianggap sebagai input atau output dari sistem.
4.	*Association*		*Association* adalah hubungan antara dua entitas yang menunjukkan bahwa mereka terkait dalam suatu konteks tertentu. Ini bisa mengacu pada hubungan antara *actor* dan *use case* atau antara *use case* itu sendiri
5.	*Include*		*Include* adalah salah satu bentuk relasi antara *use case* di mana suatu *use*

			case membutuhkan fungsi dari *use case* lain untuk menyelesaikan tugasnya. *Use case* yang termasuk akan dijalankan setiap kali *use case* induk dijalankan.
6.	*Extend*		Extend juga merupakan bentuk relasi antara *use case*, namun berbeda dengan *include*. *Extend* menunjukkan bahwa suatu *use case* memiliki perilaku tambahan yang opsional yang dapat terjadi dalam situasi tertentu.
7.	*Generalization*		*Generalization* adalah konsep dalam pemodelan yang menunjukkan bahwa satu entitas adalah bentuk umum dari beberapa entitas lainnya. *Generalization* dapat digunakan untuk menggambarkan bahwa

			satu *use case* adalah bentuk yang lebih umum dari beberapa *use case* spesifik lainnya, dengan fungsionalitas yang serupa namun detailnya berbeda.

2.1.15 *Activity Diagram*

Activity Diagram adalah diagram UML yang digunakan untuk menggambarkan alur kerja sistem (Sundaramoorthy, 2022). Alur kerja adalah urutan langkah-langkah yang diambil untuk menyelesaikan suatu tugas.

Tabel 2. 2 Komponen Activity Diagram

No.	Komponen	Simbol	Fungsi
1.	*Initial State*		*Initial state* adalah titik awal dari aktivitas. Ini menandakan titik mulai di mana proses atau aktivitas dimulai.
2.	*Final State*		*Final state* adalah titik akhir dari aktivitas. Ini menunjukkan bahwa aktivitas telah selesai dan

			sistem telah mencapai kondisi akhir.
3.	*Swimlanes*		*Swimlanes* adalah bagian vertikal di dalam diagram aktivitas yang membagi aktivitas menjadi kelompok fungsional atau entitas yang berbeda. Ini membantu dalam memvisualisasikan bagaimana tanggung jawab dibagi di antara entitas atau peran yang terlibat dalam proses.
4.	*Action State*		*Action state* menunjukkan tindakan konkret yang dilakukan dalam aktivitas. Ini bisa berupa tugas, operasi, atau proses yang harus dilakukan dalam rangka mencapai tujuan aktivitas.
5.	*Synchronization (Join/Fork)*		*Synchronization* adalah elemen yang digunakan

			untuk menunjukkan sinkronisasi antara dua atau lebih jalur dalam diagram aktivitas. *Fork* digunakan untuk membagi aliran eksekusi menjadi beberapa jalur paralel, sementara *join* digunakan untuk menyatukan jalur-jalur tersebut kembali.
6.	*Decision*		*Decision* digunakan untuk menunjukkan cabang dalam alur aktivitas berdasarkan kondisi tertentu. Ini memungkinkan untuk membuat keputusan berdasarkan nilai dari variabel atau kondisi lainnya.
7.	*Flow Final*		*Flow final* adalah titik di mana aliran aktivitas berakhir. Ini menandakan bahwa aktivitas telah

			menyelesaikan eksekusinya dan sistem telah mencapai titik akhir.
8.	*Transition*		*Transition* adalah hubungan antara dua *state* atau *action* dalam diagram aktivitas yang menunjukkan perpindahan aliran kontrol dari satu ke yang lainnya. Ini menunjukkan bagaimana sistem bergerak dari satu *state* atau *action* ke *state* atau *action* berikutnya dalam aktivitas.

2.1.16 Software dan Tools yang digunakan

2.1.16.1. Python

Python merupakan bahasa pemrograman tingkat tinggi yang banyak digunakan dalam pengembangan perangkat lunak, analisis data, dan pembelajaran mesin. Dengan sintaks yang sederhana dan mudah dipahami (Silaparasetty, 2020), Python mendukung berbagai paradigma pemrograman, termasuk pemrograman prosedural, objek-orientasi, dan fungsional. Python juga dikenal dengan

perpustakaan standarnya yang luas serta ekosistem pustaka pihak ketiga yang kaya, yang memungkinkan pengembang untuk mengimplementasikan berbagai fungsi mulai dari pengolahan data hingga pembuatan aplikasi web.

2.1.16.2. Google Colab

Google Colab (singkatan dari *Google Colaboratory*) adalah layanan *cloud* berbasis Jupyter Notebook yang disediakan oleh Google. Layanan ini memungkinkan pengguna untuk menulis dan menjalankan kode Python melalui browser tanpa memerlukan setup apapun (Bisong, 2019). Google Colab menawarkan akses gratis ke sumber daya komputasi termasuk GPU, membuatnya sangat populer di kalangan ilmuwan data, peneliti, dan mahasiswa untuk analisis data, machine learning, dan tugas-tugas komputasi intensif lainnya. Google Colab mendukung penulisan kode, visualisasi data, dan teks naratif dalam satu dokumen interaktif, memungkinkan pengguna untuk menggabungkan eksekusi kode langsung dengan penjelasan dan hasil dalam format yang terstruktur dan mudah dibagikan.

2.1.16.3. Streamlit

Streamlit adalah *framework open-source* untuk pengembangan aplikasi web yang berfokus pada ilmu data dan machine learning. Framework ini dibangun menggunakan bahasa pemrograman Python dan memungkinkan pengguna untuk membuat aplikasi web interaktif dengan mudah dan cepat (Raghavendra, 2023). Streamlit menyediakan berbagai fitur untuk memvisualisasikan data, membangun model machine learning, dan menerapkannya ke dalam aplikasi web.

2.1.17 Cross-Industry Standard Process for Data Mining (CRISP-DM)

CRISP-DM, singkatan dari *Cross-Industry Standard Process for Data Mining*, adalah metodologi proses yang banyak digunakan untuk proyek data mining. Metodologi ini menyediakan pendekatan terstruktur untuk merencanakan dan melaksanakan proyek data mining dengan efektif (Matsui et al., 2022). CRISP-DM adalah metodologi standar non-proprietary untuk data mining yang dikembangkan di Eropa pada tahun 1996 oleh 5 perusahaan yaitu Integral Solutions Ltd (ISL), Teradata, Daimler AG, NCR Corporation, dan OHRA. Menurut (Andrea, 2017) Metodologi CRISP-DM menganggap aktivitas analitis sebagai suatu proses siklus, dengan fase-fase yang harus diulang hingga hasil yang memuaskan tercapai. Oleh karena itu, fase-fase CRISP-DM, mulai dari *business understanding* hingga *deployment*, umumnya digambarkan sebagai progresi lingkaran. Berikut adalah tahapan-tahapan dalam CRISP-DM :

1. *Business understanding*

   Memahami tujuan dan syarat-syarat proyek dari sudut pandang bisnis, lalu mentransformasikan pengetahuan ini menjadi definisi masalah dalam data mining beserta rencana awal untuk mencapai tujuan yang ditetapkan.

2. *Data understanding*

   Dimulai dengan pengumpulan data awal untuk membentuk hipotesis tentang informasi yang terkandung di dalamnya. Tahap ini mencakup eksplorasi data, evaluasi kualitas data, dan penemuan subset awal dari pola-pola menarik.

3. *Data preparation*

   Meliputi semua kegiatan yang diperlukan untuk menyusun kumpulan data akhir dari data mentah awal. Tugas-tugas seperti membersihkan data, membuat

data turunan, mengintegrasikan data, dan melakukan transformasi data dilakukan pada tahap ini.

4. *Modeling*

   Berbagai teknik pemodelan diterapkan dan parameter teknis disesuaikan untuk mencapai optimasi. Tujuannya adalah memilih model terbaik yang memenuhi tujuan bisnis yang telah ditetapkan.

5. *Evaluation*

   Model yang dihasilkan dievaluasi dan dianalisis untuk memastikan bahwa tujuan bisnis tercapai. Tahap ini sering kali melibatkan pengambilan keputusan apakah akan melanjutkan ke tahap penyebaran atau memulai kembali seluruh proses dari awal.

6. *Deployment*

   Pada fase ini, model yang telah dievaluasi diterapkan dalam aplikasi bisnis. Hal ini dapat melibatkan integrasi model dengan sistem yang ada, pengembangan laporan dan visualisasi, dan pelatihan pengguna untuk menggunakan model.

2.2 Kajian Literatur

Dibawah ini adalah beberapa hasil penelitian yang relevan yang dijadikan acuan riset ini, yang terangkum dalam tabel 2.3. berikut.

Tabel 2. 3 Penelitian Terkait

No.	Penulis	Judul	Metode	Publikasi
1	(Matheos Sarimole &	Analisis Sentimen Terhadap Satu Sehat Pada Twitter Menggunakan	*Naive Bayes*	Jurnal Sains dan Teknologi, Volume 5 No.

	Kudrat, 2024)	Algoritma Naive Bayes Dan Aplikasi Support Vector Machine	*Support, Vector Machine*	3, Februari 2024
	Hasil : Hasilnya menunjukkan bahwa SVM memiliki akurasi lebih tinggi (87.95%) dibandingkan dengan *Naive Bayes* (81.65%), dengan sebagian besar sentimen yang dianalisis cenderung negatif			
	Keterbatasan Penelitian : Penelitian menggunakan sampel data Twitter yang terbatas pada tahun 2023, yang mungkin tidak mencerminkan perubahan sentimen publik secara keseluruhan atau jangka panjang.			
2.	(Atmajaya et al., 2023)	Metode SVM dan *Naïve Bayes* untuk Analisis Sentimen ChatGPT di Twitter	*Naive Bayes, Support Vector Machine*	Indonesian Journal of Computer Science dengan ISSN cetak 2302-4364 dan ISSN online 2549-7286
	Hasil : Hasilnya menunjukkan bahwa SVM yang digabungkan dengan alat analisis sentimen Vader lebih unggul, mencapai akurasi 59% dan F1-score 55%,			

	dibandingkan dengan *Naive Bayes* yang mencapai akurasi maksimum 47% dengan Vader.			
	Keterbatasan Penelitian : Penelitian menggunakan 1000 dataset Twitter yang terkait dengan ChatGPT, yang mungkin tidak mencakup variasi sentimen yang lebih luas atau representasi yang lebih komprehensif dari opini publik. Terdapat perbedaan yang signifikan dalam hasil analisis sentimen tergantung pada alat label yang digunakan, yaitu Vader atau RoBERTa, yang menunjukkan variabilitas dalam penentuan sentimen.			
3	(Davoodi & Mezei, 2022)	A Comparative Study of Machine Learning Models for Sentiment Analysis: Customer Reviews of E Commerce Platforms	Tranformer	AIS Electronic Library (AISeL)
	Hasil : Penelitian membandingkan dua jenis model klasifikasi sentimen untuk menganalisis ulasan pelanggan yaitu model pembelajaran mesin tradisional (SVM, *Naive Bayes*) dan model transformer baru (BERT, RoBERTa). Hasilnya menunjukkan bahwa model transformer jauh lebih unggul, mencapai akurasi lebih dari 98% dalam menentukan apakah ulasan positif, negatif, atau netral. Temuan ini menunjukkan bahwa model *transformer* lebih efektif dalam memahami konteks dan makna dalam ulasan dibandingkan model tradisional.			

	Keterbatasan Penelitian : Penelitian ini memanfaatkan sampel ulasan yang mungkin tidak secara sepenuhnya mewakili populasi secara keseluruhan. Oleh karena itu, pengujian model pada dataset yang lebih luas dan beragam toko *e commerce* diperlukan. Meskipun telah dilakukan anotasi dan validasi silang oleh dua peneliti, masih ada potensi kesalahan dalam penugasan sentimen yang mungkin memengaruhi pembangunan model dan kinerjanya.