PERBANDINGAN *ALGORITMA* K-NEAREST NEIGHBOR DAN NAIVE BAYES UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE*

SKRIPSI

Oleh:

Steven Winata

20211000007

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS BUDDHI DHARMA

TANGERANG

2025

PERBANDINGAN *ALGORITMA* K-NEAREST NEIGHBOR DAN NAIVE BAYES UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana pada

Program Studi Teknik Informatika Fakultas Sains dan Teknologi

Universitas Buddhi Dharma

Jenjang Pendidikan Strata 1

Oleh :

Steven Winata

20211000007

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS BUDDHI DHARMA

TANGERANG

2025

LEMBAR PERSEMBAHAN

"Teruskan segala sesuatu yang telah kita mulai, meskipun hasilnya tak selalu sesuai harapan, tetaplah berusaha dan bertahan."

Dengan penuh rasa syukur kepada Tuhan Yang Maha Esa, karya skripsi ini saya dedikasikan kepada:

1. Bapak dan Ibu, yang selalu membesarkan, membimbing, mendukung, memotivasi, serta memberikan yang terbaik untuk saya dan tidak pernah lelah mendoakan untuk kesuksesan saya.
2. Dosen (Yo Ceng Giap, M.Kom., CPS), selaku dosen pembimbing saya yang telah meluangkan waktu, tenaga, dan pikiran untuk membimbing saya selama skripsi hingga sidang.
3. Seluruh keluarga dan teman-teman, yang selalu memberikan semangat, dukungan, dan kesetiaan.
4. Semua pihak yang telah membantu terselesaikan skripsi yang tidak bisa saya sebutkan satu per satu, saya mengucapkan terima kasih sebesar besarnya.

LEMBAR PERNYATAAN KEASLIAN SKRIPSI

Yang bertanda tangan dibawah ini.

NIM	: 20211000007
Nama	: Steven Winata
Jenjang Studi	: Strata 1
Program Studi	: Teknik Informatika
Peminatan	: Data Science

Dengan ini saya menyatakan bahwa:

1. Skripsi ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik (Sarjana) atau kelengkapan studi, baik di Universitas Buddhi Dharma maupun di Perguruan Tinggi lainnya.
2. Skripsi ini saya buat sendiri tanpa bantuan pihak lain, kecuali arahan dosen pembimbing.
3. Dalam skripsi ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasi orang lain, kecuali secara tertulis dengan jelas dan dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan daftar pustaka.
4. Dalam skripsi ini tidak terdapat pemalsuan (kebohongan), seperti: buku, artikel, jurnal, data sekunder, pengelolahan data, dan pemalsuan tanda tangan dosen atau Ketua Program Studi di Universitas Buddhi Dharma yang dibuktikan dengan keasliannya.
5. Lembar penyataan ini saya buat dengan sesungguhnya, tanpa paksaan dan apabila dikemudian hari atau pada waktu lainnya terdapat penyimpangan dan ketidakbenaran dalam penyataan ini, saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah saya peroleh karena skripsi ini, serta sanksi lainnya sesuai dengan peraturan dan norma yang berlaku

Tangerang, 06-08-2025

Yang membuat pernyataan,

(Steven Winata)

20211000007

LEMBAR PERSETUJUAN PUBLIKASI KARYA ILMIAH

NIM	: 20211000007
Nama	: Steven Winata
Jenjang Studi	: Strata 1
Program Studi	: Teknik Informatika
Peminatan	: Data Science

Dengan ini menyetujui untuk memberikan ijin kepada pihak Universitas Buddhi Dharma, Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul: "PERBANDINGAN *ALGORITMA K-NEAREST NEIGHBOR* DAN *NAIVE BAYES* UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE*", beserta perangkat yang diperlukan (apabila ada). Dengan Hak Bebas Royalti Non-Eksklusif ini pihak Universitas Buddhi Dharma berhak menyimpan, mengalih-media atau format-kan, mengelolaannya dalam pengkalan data (*database*), mendistribusikannya, dan menampilkan atau mempublikasikannya di *internet* atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama mencantumkan nama saya atau pencipta karya ilmiah tersebut. Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Universitas Buddhi Dharma, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Tangerang, 06-08-2025

Yang membuat pernyataan,

(Steven Winata)

20211000007

LEMBAR PENGESEHAN PEMBIMBING

PERBANDINGAN *ALGORITMA* K-NEAREST NEIGHBOR DAN NAIVE BAYES UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE*

Dibuat oleh:

NIM : 20211000007

Nama : Steven Winata

Telah disetujui untuk dipertahankan dihadapan Tim Penguji Ujian Komprehensif

Program Studi Teknik Informatika

Peminatan Data Science

Tahun Akademik 2024/2025

Disahkan oleh,

Tangerang, 06 Agustus 2025

Pembimbing,

Yo Ceng Giap, M.Kom., CPS

NUPTK. 7044758659131143

LEMBAR PENGESAHAN SKRIPSI

PERBANDINGAN *ALGORITMA* K-NEAREST NEIGHBOR DAN NAIVE BAYES UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE*

Dibuat Oleh:

NIM : 20211000007

Nama : Steven Winata

Telah disetujui untuk dipertahankan di hadapan Tim Penguji Ujian

Komprehensif

Program Studi Teknik Informatika

Peminatan Data Science

Tahun Akademik 2024/2025

Disahkan oleh,

Tangerang, 06 Agustus 2025

Dekan,	**Ketua Program Studi,**
Dr. Yakub, M.Kom., M.M.	**Hartana Wijaya, M.Kom.**
NUPTK. 1836747648130172	**NUPTK.3844759660131192**

LEMBAR PENGESAHAN TIM PENGUJI

Nama	:	Steven Winata
Nim	:	20211000007
Fakultas	:	Sains dan Teknologi
Judul Skripsi	:	Perbandingan *Algoritma* K-Nearest Neighbor dan Naive Bayes untuk Klasifikasi dan Deteksi Pola Penyakit *Stroke*

Dinyatakan LULUS setelah mempertahankan di depan Tim Penguji pada hari Rabu, 06 Agustus 2025.

	Nama penguji :	Tanda Tangan
Ketua Sidang	: Yusuf Kurnia, M.Kom. NUPTK. 4551765666130223	
Penguji I	: Dram Renaldi, M.Kom. NUPTK. 0443768669130203	
Penguji II	: Yo Ceng Giap, M.Kom., CPS. NUPTK. 7044758659131143	

Mengetahui,

Dekan Fakultas Sains Dan Teknologi

Dr. Yakub, M.Kom., M.M.

NUPTK. 1836747648130172

KATA PENGANTAR

Dengan memanjatkan puji syukur ke hadirat Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, penulis dapat Menyusun serta menyelesaikan skripsi berjudul "**PERBANDINGAN *ALGORITMA K-NEAREST NEIGHBOR* DAN *NAIVE BAYES* UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE***". Penulisan skripsi ini dilakukan sebagai salah satu syarat untuk menyelesaikan Pendidikan Strata 1 pada Program Studi Teknik Informatika, Universitas Buddhi Dharma.

Dalam proses penyusunan skripsi ini, penulis mendapatkan banyak bantuan, dukungan, baik secara moril maupun materiil dari berbagai pihak. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan rasa terima kasih yang tulis kepada:

1. Ibu Dr. Limajatini, S.E., M.M., B.K.P., selaku Rektor Universitas Buddhi Dharma.
2. Bapak Dr. Yakub, M.Kom., M.M. selaku Dekan Fakultas Sains dan Teknologi.
3. Bapak Hartana Wijaya, M.Kom., selaku Ketua Program Studi Teknik Informatika.
4. Bapak Yo Ceng Giap, M.Kom., CPS., sebagai dosen pembimbing yang telah memberikan bimbingan, dan dukungan, dalam penulisan skripsi ini.
5. Kedua orang tua dan keluarga terc*int*a atas segala dukungan baik secara moril maupun materiil.

Penulis juga mengucapkan terima kasih kepada semua pihak yang tidak dapat disebutkan satu per satu atas segala bantuan dan kontribusinya hingga skripsi ini dapat terselesaikan. Penulis menyadari bahwa skripsi ini masih jauh dari sempurna, untuk itu penulis sangat mengharapkan kritik dan saran yang membangun demi penyempurnaan penulisan dimasa mendatang.

Akhir kata, semoga skripsi ini dapat memberikan manfaat, baik bagi penulis sendiri maupun para pembaca yang berminat.

Tangerang, 06-08-2025

Penulis

PERBANDINGAN *ALGORITMA K-NEAREST NEIGHBOR* DAN *NAIVE BAYES* UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE*

107 halaman + XVI + 3 lampiran

ABSTRAK

Stroke merupakan salah satu penyakit kronis yang berbahaya karena dapat menyebabkan kerusakan permanen pada jaringan otak dan mengganggu fungsi tubuh, seperti kemampuan berbicara, bergerak, maupun berpikir. Penyakit ini terjadi ketika aliran darah menuju otak terhambat atau pecahnya pembuluh darah di otak. Menurut *World Health Organization* (WHO), *stroke* menjadi salah satu penyebab utama kematian dan kecacatan jangka panjang di dunia. Deteksi dini berperan penting untuk mengurangi tingkat keparahan serta meminimalkan resiko komplikasi yang lebih serius. Perkembangan teknologi, khususnya di bidang *data mining*, memungkinkan pengolahan data medis untuk mengidentifikasi pola kesehatan dan memprediksi resiko penyakit secara lebih efektif. Penelitian ini membandingkan dua metode klasifikasi, yaitu *K-Nearest Neighbor* (KNN) dan *Naïve Bayes*, dalam mengelompokkan data pasien berdasarkan potensi resiko *stroke*. *Dataset* yang digunakan berasal dari sumber daring dan telah melalui tahap *preprocessing* seperti pembersihan data, penyeimbangan kelas, serta transformasi variabel. Implementasi dilakukan menggunakan bahasa pemrograman Python untuk menghasilkan model klasifikasi yang dapat membantu proses diagnosis awal. Hasil penelitian menunjukkan bahwa kedua metode memiliki kemampuan dalam mendeteksi potensi *stroke* dengan tingkat keakuratan yang memadai, meskipun memiliki karakteristik dan pendekatan yang berbeda. Temuan ini diharapkan dapat menjadi acuan dalam pemilihan metode analisis data medis serta mendukung pengambilan keputusan di bidang kesehatan, sehingga upaya pencegahan dan penanganan dapat dilakukan lebih cepat dan tepat sasaran.

Kata Kunci: Data Mining, Klasifikasi, *K-Nearest Neighbor*, *Naïve* Bayes, *Stroke*.

COMPARISON OF K-NEAREST NEIGHBOR AND NAÏVE BAYES ALGORITHMS FOR CLASSIFICATION AND PATTERN DETECTION OF STROKE

115 pages + XVI + 3 appendix

ABSTRACT

Stroke is a dangerous chronic disease that can cause permanent damage to brain tissue and disrupt body functions, including speech, movement, and cognitive abilities. It occurs when blood flow to the brain is blocked or when a blood vessel in the brain bursts. According to the World Health Organization (WHO), stroke is one of the leading causes of death and long-term disability worldwide. Early detection plays a crucial role in reducing severity and minimizing the risk of serious complications. Advances in technology, particularly in the field of data mining, make it possible to process medical data to identify health patterns and predict disease risks more effectively. This study compares two classification methods, namely K-Nearest Neighbor (KNN) and Naïve Bayes, in categorizing patient data based on potential stroke risk. The dataset used was obtained from an online source and underwent preprocessing steps such as data cleaning, class balancing, and variable transformation. Implementation was carried out using the Python programming language to develop classification models that can assist in early diagnosis. The findings indicate that both methods are capable of detecting potential stroke cases with adequate accuracy, although each has different characteristics and approaches. These results are expected to serve as a reference for selecting suitable methods in medical data analysis and to support decision-making in healthcare, enabling preventive and treatment measures to be carried out more quickly and effectively.

Keywords: Classification, Data Mining, K-Nearest Neighbor, Naïve Bayes, Stroke.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang Penelitian

Dengan meningkatnya tekanan hidup dan pola hidup tidak sehat di masyarakat modern, *stroke* kini menjadi salah satu penyebab utama kematian dan disabilitas di dunia. *Stroke* terjadi akibat penyumbatan pada pembuluh darah yang menuju otak, sehingga menyebabkan kerusakan signifikan pada sistem saraf pusat. Sayangnya, banyak orang mengabaikan tanda-tanda ringan *stroke*, sehingga kesempatan untuk melakukan diagnosis dini kerap terlewat. Di sisi lain, faktor-faktor resiko seperti hipertensi, diabetes, obesitas, kebiasaan mengonsumsi alkohol, merokok, dan gaya hidup kurang aktif semakin memperbesar kemungkinan seseorang terkena *stroke*.

Data dari WHO mengungkapkan bahwa kasus *stroke* mencapai angka kematian sekitar 12 juta kasus (WHO, 2020), menandakan perlunya metode diagnosis dini yang efektif untuk mengurangi dampak penyakit ini. Metode Naïve Bayes hadir sebagai solusi dengan keunggulannya dalam klasifikasi yang akurat dan efisien meskipun menggunakan data pelatihan minimal. Metode ini sangat cocok diterapkan dalam sistem yang membantu masyarakat mengenali dan mengelola resiko *stroke* sejak dini, sehingga potensi kematian dapat ditekan dan kualitas hidup pasien meningkat. (Rahma et al., 2023)

Menurut laporan WHO, *stroke* merupakan penyebab kematian kedua terbanyak di dunia dan berkontribusi besar terhadap kecacatan global. Di Indonesia, penyakit ini menjadi salah satu masalah kesehatan utama, dimana data mencatat sekitar 43,1% kasus *stroke* terjadi pada orang yang berusia di atas 75 tahun. *Stroke* muncul ketika aliran darah ke otak terganggu oleh penyumbatan atau pecahnya pembuluh darah, yang menyebabkan kerusakan pada sistem saraf pusat. Faktor resiko seperti tekanan darah tinggi, penyakit jantung, diabetes,

obesitas, kebiasaan merokok, dan gaya hidup kurang sehat semakin memperbesar peluang terkena *stroke*.

Berbagai metode analisis telah dikembangkan untuk mendukung upaya prediksi dan diagnosis *stroke* melalui pemahaman pola kesehatan pada pasien. Beberapa teknik analisis seperti *Logistic Regression*, *Random Forest*, *SVM*, dan *K-Nearest Neighbors* telah menunjukkan tingkat akurasi yang tinggi dalam mengidentifikasi resiko *stroke*, terutama pada data yang tidak seimbang, dengan tingkat akurasi hingga 98,63%. Diharapkan bahwa penggunaan metode ini dapat membantu mendeteksi resiko *stroke* sejak dini, memungkinkan penerapan langkah-langkah pencegahan yang lebih optimal, sehingga dapat mengurangi angka kematian dan meningkatkan kualitas hidup pasien. (Azhar et al., 2022)

Di era modern, teknologi kesehatan berkembang dengan pesat, terutama dengan semakin luasnya penerapan metode analitik yang mendukung proses diagnosis penyakit, termasuk *stroke*. *Stroke* adalah kondisi darurat yang dapat menyebabkan kematian mendadak, karena sel-sel otak dapat rusak hanya dalam beberapa menit saat aliran darah ke otak terhenti. Di Indonesia, angka kasus *stroke* terus meningkat, dan sering kali gejala awal seperti kelemahan di wajah atau lengan tidak disadari hingga penyakit ini mencapai tahap yang lebih parah. Mengingat berbagai komplikasi yang dapat timbul akibat *stroke*, diagnosis dini sangat diperlukan untuk mengurangi tingkat keparahan serta dampak lanjutan yang mungkin terjadi.

Untuk memperoleh prediksi yang lebih akurat, analisis dan klasifikasi berbasis teknologi menjadi solusi yang efektif dalam mengenali gejala serta faktor resiko *stroke*. *Algoritma* seperti *K-Nearest Neighbor* dan *Gaussian Naive Bayes* telah terbukti mampu membantu dalam mendeteksi serta mengukur resiko *stroke* dengan tingkat akurasi yang tinggi. Dengan menerapkan *algoritma-algoritma* ini, diharapkan resiko *stroke* dapat dikenali lebih awal,

sehingga tindakan pencegahan dan penanganan yang tepat dapat dilakukan untuk menurunkan angka kematian serta meningkatkan kualitas hidup pasien.

Algoritma KNN bekerja dengan mengklasifikasikan data baru berdasarkan kedekatannya dengan titik data yang telah dikenali sebelumnya, menggunakan jarak Euclidean untuk menentukan tingkat kesamaan antara data pasien baru dengan data pelatihan. Dalam penelitian perbandingan, KNN menunjukkan akurasi sebesar 68,30%, presisi 67,20%, dan *recall* 73,34%, yang mengindikasikan bahwa *algoritma* ini dapat menghasilkan klasifikasi yang cukup akurat untuk mengidentifikasi potensi *stroke*. Dengan keunggulannya ini, KNN berpotensi diterapkan lebih luas dalam mendukung deteksi dini *stroke*, yang diharapkan dapat meningkatkan efektivitas penanganan dan menekan angka kematian akibat penyakit ini. (Rachmad et al., 2022).

Metode *K-Nearest Neighbor* dan *Naive Bayes* berfungsi untuk memprediksi hasil di masa mendatang dengan memanfaatkan pola dari data sebelumnya. Pada prinsipnya, *Naive Bayes* memperkirakan kemungkinan terjadinya suatu kejadian di masa depan berdasarkan kondisi atau variabel saat ini, sedangkan *K-Nearest Neighbor* mengklasifikasikan data baru dengan melihat jaraknya terhadap data yang sudah dikenal. Kedua *algoritma* ini cukup mudah dipahami dan diterapkan, sehingga banyak digunakan dalam berbagai aplikasi, seperti klasifikasi teks, deteksi penyakit, pengenalan pola, dan analisis perilaku.

Alasan penulis menggunakan metode *K-Nearest Neighbor* dan *Naive Bayes* adalah karena kedua metode ini sangat cocok untuk klasifikasi penyakit *stroke*. *Naive Bayes* memiliki kemampuan untuk memperhitungkan beberapa atribut secara independen dan bekerja dengan baik pada *dataset* dengan berbagai ukuran, mulai dari yang kecil hingga besar. Keunggulan lain dari *Naive Bayes* adalah efisiensi perhitungannya serta kemampuannya dalam beradaptasi dengan berbagai tipe data, sehingga cocok untuk data

medis yang memiliki variasi skala. Sementara itu, *K-Nearest Neighbor* menawarkan pendekatan yang *int*uitif dalam mengklasifikasikan data baru berdasarkan kedekatan atau kemiripannya dengan data yang sudah ada, menjadikannya efektif untuk mengidentifikasi pola-pola gejala *stroke*. *Algoritma* KNN ini sederhana dan mudah diterapkan, sehingga sering digunakan dalam aplikasi analisis data. Dengan karakteristik-karakteristik ini, baik KNN maupun *Naive Bayes* menjadi pilihan yang tepat untuk mendeteksi pola dan mendukung pengambilan keputusan terkait diagnosis penyakit *stroke*. Maka dari itu, penelitian ini berjudul "**PERBANDINGAN *ALGORITMA K-NEAREST NEIGHBOR* DAN *NAIVE BAYES* UNTUK KLASIFIKASI DAN DETEKSI POLA PENYAKIT *STROKE***".

1.2 Identifikasi masalah

Berdasarkan latar belakang yang telah dijelaskan, masalah-masalah penelitian yang dapat diidentifikasi antara lain:

1. *Stroke* merupakan salah satu penyebab utama kematian dan kecacatan di seluruh dunia, terutama seiring dengan meningkatnya tekanan hidup dan gaya hidup yang kurang sehat di masyarakat modern.
2. Banyak orang sering kali mengabaikan tanda-tanda awal *stroke*, yang menyebabkan terlewatnya diagnosis dini dan meningkatkan resiko dampak serius dari penyakit ini.
3. Faktor-faktor resiko seperti hipertensi, diabetes, obesitas, konsumsi alkohol, kebiasaan merokok, dan gaya hidup kurang aktif semakin memperbesar kemungkinan terjadinya *stroke*.

4. Berdasarkan data WHO, tingginya angka kejadian *stroke* dan angka kematian yang signifikan menunjukkan kebutuhan mendesak akan metode diagnosis dini yang efektif.

1.3 Ruang Lingkup Penelitian

Mengingat luasnya topik yang akan dibahas, maka penulis akan menentukan ruang lingkup pembahasan :

1. *Algoritma* yang digunakan dalam penelitian ini adalah *K-Nearest Neighbor* dan *Naive Bayes* sebagai metode utama untuk klasifikasi resiko *stroke*.
2. Data yang digunakan merupakan data sekunder yang didapat dari www.kaggle.com.
3. Subjek penelitian mencakup pasien dewasa dengan faktor resiko *stroke*, yang teridentifikasi melalui variabel-variabel utama. Batasan usia pasien yang diteliti berkisar antara 21 hingga 80 tahun.
4. Atribut yang digunakan dalam penelitian ini meliputi jenis kelamin, usia, hipertensi, riwayat penyakit jantung, status pernikahan, jenis pekerjaan, tipe tempat tinggal, kadar glukosa darah rata-rata, BMI, dan status merokok. Atribut-atribut ini dipilih berdasarkan keterkaitannya dengan resiko *stroke*.
5. Aplikasi ini hanya bisa memperkirakan apakah seseorang berpotensi terkena *stroke* atau tidak.

1.4 Tujuan Dan Manfaat Penelitian

1.4.1 Tujuan Penelitian

Ada beberapa tujuan utama dari penelitian ini yang diharapkan oleh peneliti, diantaranya:

1. Memperoleh informasi mengenai pola penyakit *stroke* pada pasien dewasa.
2. Membangun model klasifikasi yang mampu mendeteksi resiko *stroke* menggunakan *algoritma K-Nearest Neighbor* dan *Naive Bayes*
3. Mengevaluasi hasil prediksi dari kedua *algoritma* dengan mengukur akurasi, presisi, dan *recall* untuk menentukan metode yang paling efektif dalam proses diagnosis *stroke*.

1.4.2 Manfaat Penelitian

Ada beberapa manfaat utama dari penelitian ini yang diharapkan oleh peneliti, diantaranya:

1. Mengetahui faktor-faktor resiko utama yang berperan besar dalam kejadian *stroke*.
2. Mengimplementasikan *algoritma K-Nearest Neighbor* dan *Naive Bayes* untuk meningkatkan akurasi dalam klasifikasi resiko *stroke* pada data medis.
3. Menilai efektivitas metode KNN dan *Naive Bayes* dalam mendukung diagnosis dan prediksi resiko *stroke* serta memberikan rekomendasi penggunaan *algoritma* ini di bidang kesehatan.

1.5 Sistematika Penulisan

Sistematika bertujuan untuk memberikan gambaran kepada pembaca mengenai bab yang akan ditulis dalam penelitan ini, Berikut sistematika penulisan :

1. BAB I PENDAHULUAN

   Bab ini merupakan pengantar yang memberikan gambaran awal tentang topik atau permasalahan yang dibahas dalam penelitian. Pada bab ini terdapat

bagian-bagian seperti latar belakang, identifikasi masalah, ruang lingkup penelitian, tujuan dan manfaat penelitian, serta sistematika penulisan.

2. BAB II LANDASAN TEORI

   Bab ini menguraikan teori-teori dan pendekatan yang digunakan sebagai dasar untuk menganalisis masalah penelitian. Landasan teori meliputi pembahasan *algoritma K-Nearest Neighbor* dan *Naive Bayes*, klasifikasi penyakit *stroke*, serta teori-teori tambahan yang relevan.

3. BAB III METODOLOGI

   Bab ini menjelaskan metode dan langkah-langkah yang diterapkan dalam penelitian, termasuk teknik pengumpulan data, tahap preprocessing, serta implementasi *algoritma K-Nearest Neighbor* dan *Naive Bayes* untuk proses klasifikasi. Bab ini juga membahas metode evaluasi, seperti pengukuran akurasi, presisi, dan *recall*.

4. BAB IV HASIL PEMBAHASAN

   Bab ini menyajikan hasil penelitian yang diperoleh dari penerapan *algoritma K-Nearest Neighbor* dan *Naive Bayes* dalam mengklasifikasikan risiko *stroke*. Pembahasan mencakup perbandingan performa kedua *algoritma* dan analisis faktor-faktor yang berpengaruh dalam klasifikasi.

5. BAB V SIMPULAN DAN SARAN

   Bab ini merupakan bagian akhir yang menyajikan kesimpulan dari hasil penelitian dan saran-saran untuk penelitian berikutnya. Bab ini juga mencakup rekomendasi mengenai penggunaan *algoritma* dalam pengembangan lebih lanjut untuk klasifikasi penyakit *stroke*.

BAB II

LANDASAN TEORI

2.1 Teori Penelitian

2.1.1 Teori Umum

Dalam bab ini, penulis akan membahas teori-teori umum yang berhubungan dengan topik yang diangkat.

a. Data

Data adalah kumpulan fakta atau informasi yang diperoleh melalui pengamatan, pengukuran, atau pencatatan yang digunakan untuk analisis lebih lanjut. Dalam penelitian, data berfungsi sebagai bahan dasar untuk menghasilkan kesimpulan atau prediksi yang valid (Rachmad et al., 2022).

Data yang belum diproses atau dianalisis disebut data mentah. Untuk memastikan data tersebut dapat diandalkan dan bermanfaat, diperlukan proses analisis. Analisis data bertujuan mengubah data mentah menjadi informasi yang relevan dan bermakna, sehingga dapat digunakan untuk berbagai kebutuhan.

Dalam proses penelitian, data memiliki peranan yang sangat penting. Data memungkinkan peneliti untuk memahami, mengevaluasi, dan menggambarkan suatu fenomena dengan cara yang objektif. Dengan demikian, data menjadi dasar yang esensial dalam pengambilan keputusan dan membantu menyelesaikan berbagai permasalahan secara sistematis.

b. Informasi

Informasi adalah hasil pengolahan data mentah yang telah dianalisis dan di*int*erpretasikan sehingga memiliki nilai atau makna tertentu untuk pengambilan

keputusan atau pemahaman lebih lanjut. Informasi memungkinkan pengguna untuk mengidentifikasi pola, membuat prediksi, atau mengambil tindakan berdasarkan data yang telah diolah (Bagus et al., 2024).

Setiap individu membutuhkan informasi untuk mendukung aktivitasnya dan memperluas pengetahuan. Oleh sebab itu, informasi sering dijadikan landasan dalam proses pengambilan keputusan yang tepat. Agar dikategorikan sebagai informasi yang berkualitas, terdapat beberapa kriteria yang harus dipenuhi. Berikut adalah ciri-ciri dari informasi yang berkualitas:

1. Tepat: Informasi harus berdasarkan fakta yang benar dan bebas dari kesalahan.
2. Dapat Dipercaya: Sumber informasi harus memiliki kredibilitas dan dapat dipercaya.
3. Berguna: Informasi harus relevan dengan kebutuhan dan memberikan manfaat nyata bagi penerimanya.

c. Prediksi

Prediksi adalah proses memperkirakan hasil atau kondisi masa depan berdasarkan pola atau informasi yang diperoleh dari data historis menggunakan metode tertentu, seperti *algoritma* pembelajaran mesin atau statistik (Islam et al., 2021).

Walaupun prediksi didasarkan pada analisis data, terdapat sejumlah faktor yang dapat memengaruhi hasilnya, sehingga akurasinya tidak dapat dijamin sepenuhnya. Namun demikian, prediksi dirancang untuk memberikan estimasi yang paling mendekati kenyataan. Meskipun begitu, prediksi tetap berperan sebagai alat yang efektif untuk membantu pengambilan keputusan.

d. Aplikasi

Aplikasi adalah perangkat lunak yang dirancang untuk melaksanakan tugas-tugas tertentu sesuai dengan per*int*ah pengguna, menghasilkan output yang relevan dengan tujuan utama dari aplikasi tersebut (Harmayani et al., 2021).

Aplikasi digunakan untuk mempermudah pengguna dalam menjalankan tugas tertentu atau menyelesaikan masalah yang dihadapi. Aplikasi ini dapat membantu melalui berbagai platform seperti desktop, web, atau perangkat mobile.

2.1.2 Teori Khusus

Di bab ini, penulis akan membahas teori-teori khusus yang memiliki relevansi langsung dengan topik yang diangkat.

a. Data Mining

Data mining merupakan suatu proses yang melibatkan penggunaan teknik-teknik seperti matematika, statistika, pembelajaran mesin, dan *kecerdasan buatan* untuk secara semi-otomatis mengidentifikasi serta mengekstraksi pengetahuan dari kumpulan data yang besar. Tujuan utama proses ini adalah untuk menemukan pola, tren, dan informasi baru yang dapat digunakan untuk mendukung pengambilan keputusan secara lebih efektif (Sholikhul Fiqri & Dwi Bhakti, 2024).

Data mining bertujuan mengolah data mentah menjadi informasi yang dapat digunakan untuk mendukung pengambilan keputusan yang lebih efektif. Proses ini memungkinkan penemuan wawasan baru dari data yang ada sekaligus mengungkap hubungan tersembunyi antar data.

Dalam bidang kesehatan, *data mining* dimanfaatkan untuk memperkirakan risiko penyakit dengan menganalisis data medis yang terorganisir. Teknologi ini

mendukung dokter dalam mengambil keputusan yang lebih cepat dan akurat, sehingga berperan penting dalam meningkatkan mutu pelayanan kesehatan.

Data mining sering disebut sebagai bagian dari proses *Knowledge Discovery in Database* (KDD). KDD adalah serangkaian tahapan yang dirancang untuk menemukan pola dan informasi bermanfaat dari data. Dalam proses ini, *data mining* berperan sebagai langkah utama dengan menggunakan *algoritma* tertentu untuk mengidentifikasi pola atau hubungan dalam data yang telah diproses sebelumnya (Azhar et al., 2022). Tahapan yang dilakukan dalam proses *Knowledge Discovery in Database* (KDD) adalah sebagai berikut:

1. *Data Selection*

   Data Selection adalah tahap awal dalam proses data mining yang berfokus pada memilih data yang relevan dan sesuai dengan tujuan analisis. Data yang dipilih harus memiliki kualitas dan karakteristik yang mendukung proses mining berikutnya.

2. *Preprocessing*

   Preprocessing adalah tahap penting dalam data mining yang bertujuan untuk membersihkan, mengorganisasi, dan mempersiapkan data agar siap digunakan dalam proses analisis. Tahap ini mencakup pembersihan data dari duplikasi, inkonsistensi, atau missing value, serta mengubah data menjadi format yang sesuai untuk proses data mining.

3. *Data Transformation*

   Data Transformation adalah tahap di mana data yang telah dipilih diubah ke dalam format yang sesuai dengan kebutuhan *algoritma* data mining. Transformasi ini dapat mencakup normalisasi, pengkodean, atau agregasi data untuk memudahkan analisis.

4. *Data Mining*

   Data Mining merupakan *int*i dari seluruh proses, di mana *algoritma* atau metode tertentu diterapkan untuk menemukan pola, tren, atau hubungan dalam data. Metode yang digunakan dapat berupa klasifikasi, *clustering*, prediksi, atau asosiasi, tergantung pada informasi yang ingin diperoleh.

5. *Evaluation*

   Evaluation adalah tahap akhir dalam proses *data mining*, di mana hasil yang diperoleh dianalisis dan di*int*erpretasikan. Tahapan ini bertujuan untuk memastikan bahwa model yang dihasilkan dapat dipahami dengan jelas dan relevan untuk mendukung pengambilan keputusan atau penerapan tertentu.

Berikut adalah beberapa metode yang sering digunakan dalam proses *data mining*:

1. Prediksi

   Prediksi adalah metode *data mining* yang bertujuan untuk memperkirakan nilai atau hasil di masa depan berdasarkan data historis dan tren yang sedang berlangsung.

2. Estimasi

   Estimasi adalah metode *data mining* yang digunakan untuk memprediksi suatu nilai atau informasi yang belum diketahui dengan menggunakan data yang ada.

3. Klasifikasi

   Klasifikasi adalah metode *data mining* yang bertujuan untuk mengelompokkan suatu objek ke dalam kelas tertentu berdasarkan kesamaan atau karakteristik yang dimiliki.

4. *Clustering*

   Clustering adalah metode *data mining* yang berfokus pada pengelompokan data yang memiliki kemiripan tertentu ke dalam kelompok yang sama, sehingga pola dalam data dapat diidentifikasi.

5. Asosiasi

   Asosiasi adalah metode *data mining* yang bertujuan untuk menemukan hubungan atau pola di antara elemen dalam *dataset*, seperti kombinasi item yang sering muncul bersama.

b. Klasifikasi

Klasifikasi adalah sebuah proses yang bertujuan untuk menemukan model atau fungsi yang dapat membedakan atau menjelaskan kelompok atau kategori data tertentu (Sholikhul Fiqri & Dwi Bhakti, 2024). Klasifikasi bertujuan untuk menentukan dan memprediksi kategori atau label dari data yang belum diketahui. Proses ini mengandalkan pola atau hubungan yang ditemukan pada data yang telah diketahui sebelumnya untuk menghasilkan prediksi yang akurat. Dengan demikian, klasifikasi menjadi salah satu metode penting dalam data mining untuk mengolah dan memahami data secara mendalam.

Dalam implementasinya, data yang telah memiliki label digunakan sebagai data pelatihan (training data) untuk melatih *algoritma* dalam mengenali pola-pola tertentu. Setelah model terbentuk, data tanpa label digunakan sebagai data pengujian (testing data) untuk mengevaluasi kinerja model tersebut dalam memberikan hasil prediksi. Proses ini membantu memastikan bahwa model yang dihasilkan mampu bekerja dengan baik pada data baru yang belum pernah dianalisis sebelumnya.

c. Naïve Bayes

Naive Bayes merupakan metode klasifikasi probabilistik sederhana yang menghitung sekumpulan probabilitas dengan menjumlahkan frekuensi dan kombinasi nilai dari *dataset*. Teorema Bayes mengasumsikan bahwa semua atribut bersifat independen dan tidak saling bergantung terhadap nilai yang diberikan kepada kelas variabel(Fauziah & Dana, 2023).

Naive Bayes bekerja dengan memanfaatkan konsep teorema Bayes untuk menentukan kemungkinan suatu data masuk ke dalam kelas tertentu berdasarkan fitur-fitur yang tersedia. Metode ini menganggap bahwa setiap fitur bersifat independen satu sama lain, sehingga memungkinkan proses klasifikasi menjadi lebih sederhana dan efisien, terutama pada *dataset* dengan berbagai jenis fitur (Sholekhah et al., 2024). Metode ini juga memiliki keunggulan karena mampu melakukan prediksi kelas atau label data dengan jumlah data training yang relatif kecil menggunakan perhitungan probabilitas. Proses dalam *Algoritma* Naïve Bayes meliputi langkah-langkah berikut:

1. Menghitung jumlah kelas atau label.
2. Menghitung jumlah kasus setiap kelas.
3. Kalikan semua nilai sesuai dengan yang dicari kelasnya
4. Bandingkan hasil setiap kelas

Rumus yang digunakan dalam *algoritma* Naïve Bayes adalah sebagai berikut:

$$P(H \mid X) = \frac{P(X \mid H) \cdot P(H)}{P(X)}$$

Keterangan:

1. X: Data yang kelasnya belum diketahui (label yang akan diprediksi).

2. H: Atribut atau hipotesis data yang digunakan untuk memprediksi kelas.
3. P(H|X): Probabilitas bahwa hipotesis H benar berdasarkan data X.
4. P(X|H): Probabilitas data X terjadi berdasarkan kondisi hipotesis H.
5. P(H): Probabilitas awal dari hipotesis H.
6. P(X): Probabilitas awal dari data X.

d. *Stroke*

Stroke merupakan gangguan yang disebabkan oleh masalah pada sirkulasi darah di otak, baik karena penyumbatan (*stroke* iskemik) atau pecahnya pembuluh darah (*stroke* hemoragik). Gangguan ini mengakibatkan aliran darah ke area tertentu di otak terganggu, sehingga otak tidak menerima cukup oksigen dan nutrisi yang diperlukan (Rahma et al., 2023). Ketika aliran darah ke otak terhambat, baik karena penyumbatan maupun pecahnya pembuluh darah, hal ini dapat menyebabkan kerusakan pada jaringan otak yang berdampak pada fungsi tubuh, seperti kemampuan berbicara, bergerak, atau berpikir.

Stroke tergolong penyakit yang dapat memberikan efek jangka panjang jika tidak segera ditangani. Namun, dengan penanganan yang tepat, termasuk terapi medis, rehabilitasi, dan perubahan pola hidup seperti menjaga tekanan darah serta pola makan yang sehat, komplikasi dapat dicegah, dan kualitas hidup pasien dapat ditingkatkan. Deteksi dini dan langkah pengobatan yang sesuai sangat penting untuk mengurangi dampak serius.

Setiap tahunnya, jumlah kasus *stroke* terus bertambah dengan 13,7 juta kasus baru, di mana 5,5 juta di antaranya berakhir dengan kematian. Secara global, sekitar 70% kematian akibat *stroke* tercatat di negara-negara dengan pendapatan rendah dan menengah. Meskipun sebagian besar kasus terjadi pada kelompok usia lanjut, *stroke*

juga dapat dialami oleh orang yang lebih muda, terutama jika dipengaruhi oleh faktor risiko seperti tekanan darah tinggi, diabetes, dan pola hidup yang kurang sehat (Rahma et al., 2023).

2.1.3 Teori Rancangan

Pada bab ini, penulis akan memaparkan pembahasan terkait teori dan rancangan yang mendasari topik penelitian ini. Penjelasan ini bertujuan untuk memberikan landasan konseptual yang relevan dengan isu yang diangkat.

a. *K-Nearest Neighbor* (KNN)

K-Nearest Neighbor (KNN) adalah *algoritma* yang banyak digunakan dalam klasifikasi data dan pengenalan pola. *Algoritma* ini menyediakan pendekatan sederhana namun efektif untuk memprediksi kelas suatu data berdasarkan kemiripannya dengan data lain. KNN adalah salah satu metode yang sering diimplementasikan dalam berbagai bidang untuk menghasilkan model prediksi yang akurat dengan memanfaatkan hubungan antara data pelatihan dan data uji (Sutomo et al., 2023).

K-Nearest Neighbor (KNN) adalah *algoritma* yang mudah digunakan namun sangat fleksibel dan banyak diaplikasikan dalam klasifikasi serta prediksi data. Pendekatan ini bekerja dengan menganalisis kemiripan atau kedekatan antar data untuk menentukan kelas atau kategori yang sesuai. Tujuan utama *KNN* adalah menghasilkan klasifikasi yang akurat melalui prinsip sederhana berbasis tetangga terdekat. *Algoritma* ini menunjukkan bagaimana hubungan antar data dapat dimanfaatkan untuk mengenali pola dan membuat prediksi yang dapat dipercaya. Langkah-langkah umum dalam menggunakan KNN:

1. Pengumpulan Data

   Tahap awal dalam penerapan KNN adalah mengumpulkan *dataset* yang akan digunakan sebagai data pelatihan dan pengujian. Data pelatihan membantu model mempelajari pola-pola yang ada, sementara data pengujian digunakan untuk mengevaluasi akurasi model. *Dataset* yang dikumpulkan harus relevan dan mencakup atribut serta label untuk mendukung proses klasifikasi secara efektif.

2. *Preprocessing Data*

   Preprocessing adalah proses persiapan data sebelum digunakan oleh *algoritma* KNN. Pada tahap ini, data diperiksa dan dibersihkan dari duplikasi, nilai kosong, atau anomali lainnya. Selain itu, dilakukan normalisasi untuk menyamakan skala setiap atribut serta seleksi fitur untuk fokus pada atribut yang paling berpengaruh. Proses ini memastikan bahwa data berada dalam kondisi terbaik untuk menghasilkan prediksi yang akurat.

3. Menentukan Nilai K

   Menentukan nilai K berarti memilih jumlah tetangga terdekat yang akan digunakan untuk mengklasifikasikan data uji. Nilai K yang kecil membuat model lebih peka terhadap perubahan kecil atau noise pada data, sedangkan nilai K yang besar dapat mengurangi sensitivitas terhadap pola lokal. Pemilihan nilai K yang optimal biasanya dilakukan melalui proses validasi untuk memastikan hasil terbaik bagi *dataset* yang digunakan.

4. Menghitung Jarak

   Pada tahap ini, jarak antara data uji dengan setiap data pelatihan dihitung untuk menentukan tetangga terdekat. Metode yang sering digunakan

mencakup *Euclidean Distance* untuk jarak lurus, *Manhattan Distance* untuk jarak berbasis grid, atau *Minkowski Distance* sebagai generalisasi keduanya. Langkah ini adalah dasar untuk mengidentifikasi data pelatihan mana yang paling relevan dengan data uji.

5. Klasifikasi Data

   Data uji diklasifikasikan berdasarkan kelas mayoritas dari tetangga terdekat yang telah ditemukan. Setelah tetangga terdekat ditentukan, jumlah kelas dalam tetangga tersebut dihitung, dan kelas dengan frekuensi tertinggi dipilih sebagai prediksi untuk data uji. Proses ini memungkinkan KNN untuk memanfaatkan hubungan antar data dalam membuat prediksi.

6. Evaluasi Hasil

   Langkah terakhir adalah mengevaluasi performa model untuk memastikan hasil yang dihasilkan memenuhi harapan. Metrik seperti akurasi, presisi, *recall*, dan F1-score digunakan untuk menilai kemampuan model dalam mengklasifikasikan data dengan benar. Evaluasi ini membantu memastikan bahwa model bekerja sesuai kebutuhan dan memberikan hasil yang dapat diandalkan.

Untuk menghitung kedekatan antar data, KNN menggunakan rumus berikut:

$$d = \sqrt{\sum_{i=1}^{n} (X_{test,i} - X_{train,i})^2}$$

Keterangan

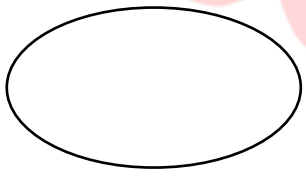
1. d: Jarak antara data uji (XtestX_{test}Xtest) dan data pelatihan (XtrainX_{train}Xtrain).
2. Xtest,i: Nilai fitur ke-iii dari data uji.

3. Xtrain,i: Nilai fitur ke-iii dari data pelatihan.
4. n: Jumlah fitur dalam *dataset* (dimensi data).
5. (Xtest,i−Xtrain,i)2: Perbedaan kuadrat antara nilai fitur ke-iii pada data uji dan data pelatihan.
6. Akar kuadrat untuk mendapatkan jarak Euclidean dalam bentuk standar.

b. *Flowchart*

Flowchart adalah representasi visual dari sebuah proses atau alur kerja yang menggunakan simbol-simbol standar untuk menggambarkan langkah-langkah secara sistematis. *Flowchart* digunakan untuk mempermudah pemahaman terhadap suatu proses, mengidentifikasi langkah-langkah utama, dan menunjukkan hubungan antar langkah dalam urutan logis (Rusydy et al., 2024). Alat ini sering digunakan dalam berbagai bidang, seperti pemrograman, manajemen proyek, dan analisis sistem, untuk merancang, menganalisis, atau mendokumentasikan proses yang kompleks. *Flowchart* dibuat untuk menjelaskan proses pemecahan masalah secara terperinci melalui langkah-langkah yang ringkas, terorganisir, dan mudah dimengerti, dengan menggunakan simbol-simbol baku.

Tabel 2.1 *Flowchart*

Bentuk Simbol	Nama Simbol	Fungsi
	Terminator	Menunjukkan awal dan akhir dari proses
	Input/Output	Menunjukkan proses input (misal memasukkan data) atau

		output (misal menampilkan hasil prediksi)
	Proses	Menunjukkan sesuatu aktivitas/proses yang harus dilakukan
	Arah Aliran	Menunjukkan arah urutan langkah dari satu proses ke proses lainnya

c. *Hypertext Preprocessor* (PHP)

PHP adalah bahasa pemrograman server-side yang sering digunakan untuk membuat *website* dinamis. Bahasa ini dirancang untuk memudahkan pengelolaan data dan *int*eraksi dengan pengguna, seperti mengolah data yang dikirimkan melalui formulir atau mengatur sesi pengguna. Kemampuannya untuk secara dinamis menghasilkan halaman web menjadikan PHP salah satu pilihan favorit dalam pembuatan aplikasi web (Sotnik et al., 2023).

PHP juga merupakan perangkat lunak open source, yang berarti dapat diakses, digunakan, dan dimodifikasi secara bebas tanpa biaya. Kemampuannya untuk bekerja di berbagai platform sistem operasi, seperti Windows, Linux, dan macOS, serta *int*egrasinya dengan *database* seperti MySQL, Oracle, dan PostgreSQL, membuat PHP menjadi alat yang fleksibel dan serbaguna dalam pengembangan web.

d. *Hypertext Markup Language* (HTML)

HTML (*HyperText Markup Language*) adalah bahasa yang dirancang untuk menyusun struktur dan isi sebuah halaman web. HTML menggambarkan halaman web sebagai teks terorganisir yang terdiri dari berbagai elemen, atribut, dan hubungan hierarkis yang menentukan isi dan cara tampilan elemen-elemen tersebut di peramban (Gur et al., 2023).

Selain itu, jika terjadi kesalahan dalam penggunaan HTML, hanya bagian yang bermasalah yang terpengaruh, tanpa memengaruhi keseluruhan fungsi halaman, menjadikannya bahasa yang andal dan praktis untuk pengembangan web.

e. *Cascading Style Sheets* (CSS)

CSS (*Connectome Spatial Smoothing*) adalah metode yang dirancang untuk meningkatkan keakuratan dan sensitivitas peta konektivitas otak dengan resolusi tinggi. Teknik ini bekerja dengan menerapkan kernel smoothing pada kedua ujung koneksi dalam matriks konektivitas, sehingga menghasilkan matriks yang dihaluskan secara spasial (Mansour L et al., 2022).

CSS mendukung berbagai perangkat dan resolusi layar, memungkinkan pembuatan desain yang responsif dan lebih adaptif untuk meningkatkan pengalaman pengguna. Hal ini menjadikan CSS sebagai komponen utama dalam menciptakan antarmuka web yang menarik dan efisien.

f. MySQL

MySQL adalah sistem manajemen basis data relasional yang memanfaatkan SQL (*Structured Query Language*) sebagai bahasa utama untuk mengelola dan mengakses data dalam *database*. MySQL banyak digunakan dalam pengembangan

aplikasi web karena kemampuannya yang efisien dalam mengolah data serta skalabilitasnya yang tinggi (Sotnik et al., 2023).

Walaupun terdapat berbagai sistem manajemen basis data lainnya, MySQL tetap menjadi salah satu pilihan utama berkat keunggulannya. MySQL menyediakan berbagai alat bantu yang memudahkan administrator dalam mengelola dan memantau *database* melalui antarmuka berbasis web. Selain itu, MySQL mendukung akses oleh banyak pengguna sekaligus, memungkinkan mereka untuk memanipulasi data secara bersamaan tanpa menimbulkan konflik, menjadikannya solusi yang andal untuk berbagai kebutuhan aplikasi, dari yang sederhana hingga yang kompleks.

g. *Python*

Python adalah bahasa pemrograman yang dapat digunakan untuk berbagai keperluan, termasuk pengelolaan data mining. Dengan dukungan library yang beragam, Python mampu menangani seluruh proses data mining, mulai dari pengumpulan dan pembersihan data hingga analisis dan visualisasi. Ketika digunakan bersama platform seperti Google Colab, Python menyediakan fitur lengkap, termasuk akses penyimpanan *dataset*, pengolahan data, serta pembuatan visualisasi data yang menarik dalam bentuk diagram atau grafik (Chromiński et al., 2021).

2.1.4 Teori Pengujian

Pada bab ini, penulis akan menguraikan teori-teori yang relevan sebagai landasan dalam pengujian yang berkaitan dengan topik pembahasan. Penjelasan ini diharapkan dapat memberikan pemahaman yang mendalam mengenai kerangka konseptual yang mendasari penelitian ini.

a. *Confusion Matrix*

Confusion matrix adalah alat analisis statistik yang digunakan dalam klasifikasi untuk mengevaluasi kinerja model pembelajaran mesin. Matriks ini menunjukkan bagaimana prediksi model dibandingkan dengan label sebenarnya, dengan mengukur distribusi hasil yang benar dan salah. Confusion Matrix tradisional digunakan dalam klasifikasi multi-kelas untuk mengidentifikasi tingkat kesalahan, seperti True Positives (TP), True Negatives (TN), False Positives (FP), dan False Negatives (FN) (Heydarian et al., 2022).

Tabel 2.2 Perhitungan *Confusion Matrix*

	Aktual Positif	Aktual Negatif
Prediksi Positif	TP	FP
Prediksi Negatif	FN	TN

Contoh menggunakan model untuk memprediksi apakah seseorang berisiko terkena *stroke* atau tidak.

1. TP = memprediksi individu beresiko terkena *stroke*, dan prediksi tersebut benar.
2. FP = memprediksi individu berisiko terkena *stroke*, tetapi kenyataannya tidak.
3. TN = memprediksi individu tidak berisiko terkena *stroke*, dan kenyataannya memang tidak.
4. FN = memprediksi individu tidak berisiko terkena *stroke*, tetapi kenyataannya mereka berisiko.

Berikut pengertian dari *Accuracy, Precision,* dan *Recall* :

1. *Accuracy*

   Accuracy mengukur tingkat keberhasilan model dalam membuat prediksi yang benar. Nilai ini dihitung dengan membagi jumlah prediksi yang benar oleh total jumlah data.

   Rumus:

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

2. *Precision*

   Precision mengindikasikan sejauh mana prediksi positif yang dibuat oleh model benar-benar akurat. Ini dihitung dengan membandingkan jumlah prediksi positif yang benar dengan total prediksi positif.

   Rumus:

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

3. *Recall*

   Recall mengukur kemampuan model dalam mengidentifikasi semua kasus positif yang ada. Ini dihitung dengan membagi jumlah prediksi positif yang benar dengan total jumlah data yang sebenarnya positif.

   Rumus:

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

2.2 Kajian Pustaka

2.2.1 Ibnu Rusydy, Ismet Canbulat, Chengguo Zhang, Chunchen Wei, Alison McQuillan, Geoenvironmental Disasters, Vol 11, no 28, hal 1–26, 2024

Jurnal ini membahas pentingnya pengembangan design *flowchart* untuk evaluasi probabilistik kestabilan lereng batuan. Masalah utama yang diangkat adalah

kompleksitas dalam memahami kestabilan lereng akibat heterogenitas sifat batuan, struktur geologi, dan dampak gempa. Untuk mengatasi hal ini, penulis mengusulkan *int*egrasi empat metode utama, yaitu teknik klasifikasi massa batuan, analisis kinematik, *limit equilibrium* (LE), dan metode numerik. Penelitian ini menunjukkan bagaimana ketidakpastian aleatorik (karakteristik bawaan) dan epistemik (kekurangan data atau informasi) dapat diatasi dengan pendekatan probabilistik.

Melalui studi kasus di Indonesia, jurnal ini membuktikan bahwa penggabungan metode-metode tersebut memberikan hasil yang lebih andal dalam analisis kestabilan lereng. Selain itu, penulis memperkenalkan formula baru untuk menghitung probabilitas kegagalan total dengan mempertimbangkan semua metode yang digunakan. Hasilnya, metode probabilistik ini dapat membantu dalam desain lereng yang lebih aman dan optimal, terutama di wilayah rawan gempa.

2.2.2 Izzeddin Gur, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Google Research, Vol 1, no 2, hal 1–19, 2023

Jurnal ini mengeksplorasi kemampuan *large language models* (LLM) dalam memahami struktur HTML untuk aplikasi seperti navigasi web otomatis, web *crawling*, dan pengisian formulir otomatis. LLM, terutama model berbasis T5, menunjukkan kinerja luar biasa dalam tiga tugas utama: klasifikasi semantik elemen HTML, generasi deskripsi dari elemen HTML, dan navigasi otomatis situs web berbasis HTML. Dibandingkan dengan model khusus lainnya, LLM memiliki efisiensi data hingga 192 kali lebih tinggi dan menunjukkan transfer pembelajaran yang unggul dari korpus NLP standar ke tugas-tugas berbasis HTML.

Penelitian ini juga memperkenalkan *dataset* HTML skala besar yang dihasilkan dari *CommonCrawl* untuk mendukung pelatihan model. Hasil menunjukkan bahwa LLM

mampu memahami struktur dan konten HTML secara efektif, memungkinkan otomatisasi web yang lebih cerdas. Penulis menyimpulkan bahwa arsitektur *encoder-decoder* seperti T5 sangat ideal untuk aplikasi ini, membuka peluang baru untuk *int*egrasi LLM dalam *age*n web otonom.

2.2.3 Svitlana Sotnik, Volodymyr Manakov, Vyacheslav Lyashenko, IJAISR, Vol 7, no 1, hal 11–17, 2023

Jurnal ini membahas peran penting PHP dan MySQL dalam pengembangan proyek web modern. Fokusnya adalah pada fitur utama PHP seperti kesederhanaan, kompatibilitas, dan skalabilitas, yang membuatnya tetap populer di kalangan pengembang web meskipun banyak bahasa pemrograman pesaing. Penulis juga mengulas beberapa kerangka kerja PHP terkemuka, seperti *CodeIgniter* dan *CakePHP*, serta manfaatnya dalam menciptakan aplikasi dinamis. MySQL, sebagai sistem manajemen basis data, juga diuraikan karena perannya yang penting dalam mendukung proyek-proyek web yang dinamis dan *int*eraktif.

Melalui studi kasus, jurnal ini menunjukkan aplikasi PHP dan MySQL dalam berbagai bidang seperti pendidikan, bisnis, dan medis. Contoh-contoh ini mencakup forum mahasiswa, sistem informasi pemesanan makanan, dan platform visualisasi data medis berbasis web. Jurnal ini menyoroti bahwa kombinasi PHP dan MySQL sangat relevan untuk memenuhi kebutuhan pengembangan web modern yang *int*eraktif dan berbasis data.

2.2.4 Harmayani, Dicky Apdilah, Mapilindo, Oktopanda, Jeperson Hutahaean, Yayasan Kita Menulis, Cetakan 1, hal 1–90, 2021

Buku ini berfungsi sebagai panduan dasar tentang komputer, yang mencakup definisi, fungsi, jenis, dan komponen sistem komputer. Penulis menjelaskan elemen

utama komputer, seperti perangkat keras, perangkat lunak, dan pengguna (*brainware*). Selain itu, buku ini juga menyajikan perkembangan teknologi komputer dari generasi pertama hingga era kecerdasan buatan. Setiap bab memberikan wawasan terperinci, mulai dari sejarah perkembangan komputer hingga penerapan perangkat lunak populer seperti *Microsoft Word* dan *Excel*.

Penulis bertujuan untuk membantu pembaca, terutama pemula dan mahasiswa, memahami konsep dasar dan aplikasi komputer dalam kehidupan sehari-hari. Bab-bab dalam buku ini juga mencakup aspek teknis seperti perangkat keras, perangkat lunak, serta fungsi dan kegunaannya. Buku ini diharapkan dapat menjadi referensi yang bermanfaat untuk memahami dasar-dasar teknologi komputer.

2.2.5 Sina Mansour L, Caio Seguin, Robert E. Smith, Andrew Zalesky, NeuroIm*age*, Vol 250, hal 118930, 2022

Jurnal ini membahas teknik *Connectome Spatial Smoothing* (CSS) yang dirancang untuk meningkatkan akurasi dan reliabilitas peta konektom spasial resolusi tinggi dari *diffusion MRI*. CSS menggunakan kernel *smoothing* pada dua ujung setiap koneksi untuk mengurangi *noise*, artefak, dan *misalignment*. Penelitian ini menunjukkan bahwa CSS mampu meningkatkan identifikasi, sensitivitas, dan kekuatan statistik konektom, baik pada resolusi tinggi maupun atlas otak skala besar. Penulis juga mengevaluasi parameter smoothing yang optimal untuk metode ini.

Hasil penelitian menunjukkan bahwa penerapan CSS meningkatkan reliabilitas peta konektom, terutama untuk aplikasi resolusi tinggi. Selain itu, CSS juga membantu analisis statistik pada konektom berbasis atlas otak. Dengan efisiensi komputasi yang ditingkatkan, teknik ini diharapkan menjadi bagian penting dari alur kerja pemetaan konektom di masa depan.

2.2.6 WHO, Geneva, "Avoiding Heart Attacks and *Strokes*", 2020

Buku ini dirancang oleh WHO untuk memberikan panduan praktis tentang pencegahan serangan jantung dan *stroke*. Dalam buku ini dijelaskan penyebab utama dari penyakit ini, seperti pola hidup tidak sehat, tekanan darah tinggi, diabetes, dan kadar lemak darah yang tinggi. Buku ini memberikan langkah-langkah pencegahan sederhana, termasuk berhenti merokok, makan makanan sehat, dan tetap aktif secara fisik. Selain itu, pembaca diberikan informasi tentang tanda-tanda awal serangan jantung dan *stroke*, serta tindakan darurat yang harus dilakukan.

WHO juga menyoroti pentingnya deteksi dini dan perubahan gaya hidup untuk mengurangi risiko. Buku ini menjelaskan bagaimana keluarga dan masyarakat dapat memainkan peran penting dalam mendorong kebiasaan hidup sehat. Didesain untuk pembaca umum, buku ini menyarankan cara mudah untuk mencegah penyakit kardiovaskular yang dapat mengancam jiwa, sambil mempromosikan gaya hidup yang lebih baik untuk kesehatan jangka panjang.

2.2.7 Chromiński, Kornel Benko, Ľubomír Hernández-Figueroa, Zenón José González-Domínguez, José Daniel Rodríguez-del-Pino, Juan Carlos, 2021

Jurnal "Python Fundamentals" memberikan pengantar komprehensif tentang pemrograman Python, yang dirancang untuk pembaca dari berbagai latar belakang. Python diperkenalkan sebagai bahasa pemrograman yang kuat, elegan, dan mudah dipahami. Buku ini menguraikan dasar-dasar pemrograman Python seperti variabel, tipe data, dan operasi dasar hingga konsep lanjutan seperti tuples, dictionaries, kontrol perulangan, dan *functions*. Penulis juga menyoroti fleksibilitas Python untuk berbagai aplikasi, dari pengolahan data hingga pengembangan web.

Selain itu, buku ini menekankan pentingnya pendekatan praktis melalui contoh kode dan penjelasan langkah-demi-langkah. Fitur seperti struktur data mutabel dan tidak mutabel, komentar kode, serta teknik *debugging* juga dijelaskan untuk membantu pembaca mengembangkan keterampilan pemrograman yang efektif. Buku ini cocok bagi pemula maupun pengembang yang ingin memperdalam pemahaman mereka tentang Python.

2.2.8 Heydarian, MohammadrezaDoyle, Thomas E. Samavi, Reza, 2022

Jurnal "Multi-Label Confusion Matrix" membahas pengembangan *Multi-Label Confusion Matrix* (MLCM) untuk menilai kinerja *algoritma* klasifikasi multi-label dalam pembelajaran mesin. Dalam klasifikasi *multi-label*, setiap *instance* dapat memiliki lebih dari satu label, sehingga metode evaluasi tradisional seperti matriks kebingungan (*confusion matrix*) tidak dapat digunakan. MLCM dirancang untuk mengatasi kekurangan ini dengan memberikan analisis rinci tentang distribusi prediksi yang benar dan salah, termasuk jumlah prediksi positif palsu (*false positives*) dan negatif palsu (*false negatives*) untuk setiap label. Penulis menguji metode ini pada dua *dataset* nyata: sinyal ECG 12-lead dengan sembilan kelas dan poster film dengan delapan belas kelas.

Hasil penelitian menunjukkan bahwa MLCM memberikan wawasan lebih jelas tentang kinerja *algoritma* dibandingkan dengan metode evaluasi saat ini seperti *hamming loss*, presisi, *recall*, dan *F1-score*. Sebagai contoh, MLCM mampu menunjukkan distribusi kesalahan prediksi antar kelas, sesuatu yang tidak dapat ditangkap oleh metode satu-lawan-semua (*one-vs-rest*) konvensional. Penulis menyoroti bahwa MLCM tidak hanya meningkatkan evaluasi kinerja model tetapi juga dapat digunakan untuk memandu pengaturan parwameter dan identifikasi kelas dengan tingkat kebingungan tinggi, sehingga membantu peningkatan kualitas model secara keseluruhan.

2.2.9 Fitriana Sholekhah, Adinda Dwi Putri, Rahmaddeni, Lusiana Efrizoni, MALCOM Vol. 4, No. 2, hal. 507–514, April 2024

Jurnal ini membandingkan *algoritma Naïve Bayes* dan *K-Nearest Neighbors* (KNN) dalam klasifikasi sindrom metabolik. Penelitian dilakukan dengan *dataset* publik yang diambil dari Kaggle, mencakup 2041 data dengan 15 fitur yang menjelaskan berbagai aspek sindrom metabolik. Hasil penelitian menunjukkan bahwa KNN menghasilkan akurasi tertinggi sebesar 82% dengan proporsi pembagian data 50:50, sementara *Naïve Bayes* memiliki akurasi sebesar 79%. Penelitian ini bertujuan untuk memberikan wawasan tentang *algoritma* klasifikasi mana yang lebih efektif untuk kasus sindrom metabolik.

Penulis juga membahas pentingnya pembagian data menjadi tiga skenario (50:50, 60:40, dan 70:30) untuk mendapatkan hasil yang lebih reliabel. Dari hasil perbandingan tersebut, *algoritma* KNN menunjukkan performa yang lebih baik dibandingkan *Naïve Bayes* dalam skenario tertentu, tetapi *Naïve Bayes* tetap menjadi metode yang lebih sederhana dan cepat untuk implementasi dalam *dataset* besar. Penelitian ini merekomendasikan pengembangan model lebih lanjut dengan *algoritma* lain untuk diagnosis sindrom metabolik.

2.2.10 Dinda Ulfatul Maula Rachmad, Hardian Oktavianto, Miftahur Rahman, Jurnal Smart Teknologi Vol. 3, No. 4, hal. 405–412, Mei 2022

Jurnal ini membahas perbandingan *algoritma K-Nearest Neighbor* (KNN) dan *Gaussian Naïve Bayes* dalam klasifikasi penyakit *stroke*. Dengan *dataset* medis, penelitian ini menunjukkan bahwa *Gaussian Naïve Bayes* memiliki akurasi tertinggi sebesar 74,45%, presisi 74,01%, dan recall 75,71%. Sementara itu, KNN menghasilkan akurasi sebesar 68,30%, presisi 67,20%, dan recall 73,34%. Berdasarkan hasil tersebut,

Gaussian *Naïve Bayes* terbukti lebih unggul dalam klasifikasi penyakit *stroke* dibandingkan KNN.

Selain itu, jurnal ini juga menguraikan pentingnya *algoritma* klasifikasi untuk membantu praktisi medis dalam mendiagnosis *stroke* lebih cepat dan akurat. Penelitian ini memberikan wawasan penting mengenai bagaimana *algoritma Gaussian Naïve Bayes* lebih baik dalam menangani *dataset* dengan karakteristik tertentu, terutama pada penyakit yang memiliki gejala kompleks seperti *stroke*.

2.2.11 Alphinda Rahma Safira Nisa', Hendro Nugroho, Gusti Eka Yuliastuti, SNESTIK Prosiding, hal. 177–184, Maret 2023

Jurnal ini mengimplementasikan metode *Naïve Bayes* untuk diagnosis dini penyakit *stroke* dengan *dataset* dari Rumah Sakit Haji Surabaya. *Dataset* mencakup 100 data dengan 10 variabel, seperti usia, hipertensi, diabetes, dan kadar gula darah. Penelitian ini menggunakan teknik validasi silang *10-fold* untuk mengevaluasi performa *algoritma*, menghasilkan akurasi tertinggi sebesar 82%. Selain itu, percobaan dengan teknik *oversampling* dan *undersampling* menunjukkan akurasi masing-masing sebesar 71% dan 64%.

Penelitian ini menyimpulkan bahwa *Naïve Bayes* lebih efektif pada *dataset* seimbang dibandingkan dengan teknik penanganan data tidak seimbang. Dengan hasil ini, *algoritma Naïve Bayes* direkomendasikan sebagai alat diagnosis berbasis data mining untuk penyakit *stroke*, karena implementasinya yang sederhana dan akurasinya yang cukup tinggi.

2.2.12 Mohammad Sholikhul Fiqri, Henny Dwi Bhakti, JATI Vol. 8, No. 4, hal. 7305–7308, Agustus 2024

Jurnal ini membahas klasifikasi potensi penyakit diabetes mellitus tipe II pada pasien menggunakan *algoritma* KNN (*K-Nearest Neighbor*). *Dataset* yang digunakan diambil dari Kaggle, dengan 4303 data pasien yang terdiri dari 17 variabel dan satu kelas target. Penelitian menunjukkan bahwa KNN dengan nilai K=1 menghasilkan akurasi tertinggi sebesar 85%, sementara KNN dengan K=3 menghasilkan akurasi sebesar 75%. Penelitian ini menyimpulkan bahwa KNN dengan K=1 lebih efektif untuk klasifikasi penyakit diabetes tipe II.

Jurnal ini juga mencatat bahwa proses klasifikasi mencakup *preprocessing* data untuk menghilangkan *noise* dan memastikan konsistensi *dataset*. Selain itu, hasil klasifikasi diukur menggunakan matriks kebingungan (*Confusion Matrix*) untuk mengevaluasi akurasi model. Penelitian ini menjadi langkah awal untuk pengembangan sistem diagnosis berbasis data mining dalam mendeteksi dini penyakit diabetes.

2.2.13 Yufis Azhar, Aidia Khoiriyah Firdausy, Putri Juli Amelia, S*INT*ECH Journal Vol. 5, No. 2, hal. 191–194, Oktober 2022

Jurnal ini membandingkan beberapa *algoritma* klasifikasi data mining untuk prediksi penyakit *stroke*, termasuk *Logistic Regression*, *Random Forest*, *Support Vector Machine* (SVM), dan KNN. Penelitian dilakukan dengan *dataset stroke* yang dibagi menjadi data latih dan uji menggunakan validasi split. Hasil menunjukkan bahwa *Logistic Regression* memiliki akurasi tertinggi sebesar 75,65% pada data seimbang, sementara pada data tidak seimbang, *Logistic Regression*, *Random Forest*, SVM, dan KNN mencapai akurasi hingga 98,63%.

Penelitian ini menyoroti pentingnya teknik data mining dalam mengidentifikasi penyakit *stroke* dan memberikan wawasan tentang *algoritma* yang paling efektif berdasarkan karakteristik *dataset.* Dengan hasil tersebut, penelitian ini merekomendasikan penggunaan *algoritma Logistic Regression* dan *Random Forest* untuk prediksi penyakit *stroke* yang lebih akurat dan efisien.

2.2.14 Md. Monirul Islam, Sharmin Akter, Md. Rokunojjaman, Jahid Hasan Rony, Al Amin, Vol. 1, No. 2, hal. 17–22, Desember 2021

Jurnal "*Int*ernational Journal of Electronics and Communications System" membahas analisis prediksi penyakit *stroke* menggunakan *algoritma machine learning* dan teknik fitur untuk meningkatkan akurasi klasifikasi. *Dataset* yang digunakan mencakup 5110 observasi dengan 12 atribut, seperti usia, hipertensi, penyakit jantung, dan status merokok. Penelitian ini menerapkan beberapa *algoritma*, termasuk *Random Forest*, *Logistic Regression*, *Decision Tree Classifier*, dan KNN (*K-Nearest Neighbor*), dengan hasil terbaik dicapai oleh *algoritma Random Forest*. Model ini menghasilkan akurasi tertinggi dengan nilai *precision*, *recall*, dan *F1-score* masing-masing sebesar 96%.

Penelitian ini juga memperkenalkan aplikasi berbasis cloud yang memungkinkan pengguna untuk memprediksi resiko *stroke* berdasarkan data pribadi mereka, seperti BMI dan kadar gula darah. Aplikasi ini dirancang agar ramah pengguna, memberikan peringatan dini kepada pasien potensial dengan data yang dimasukkan melalui perangkat seluler. Studi ini menunjukkan bagaimana *int*egrasi *algoritma machine learning* dengan sistem berbasis aplikasi dapat membantu mencegah dampak fatal *stroke* melalui diagnosis dini dan pengambilan keputusan yang lebih cepat.

2.2.15 Ida Bagus Ary Indra Iswara, Ida Bagus Gede Anandita, Maria Dahul, Journal of Computer Networks, Architecture and High Performance Computing, Vol. 6, No. 3, hal. 1415–1424, Juli 2024

Jurnal "*Comparative Analysis of Naïve Bayes and K-Nearest Neighbor (KNN) Algorithms in Stroke Classification*" membahas perbandingan kinerja dua *algoritma* klasifikasi, yaitu *Naïve Bayes* dan *K-Nearest Neighbor*, dalam mendeteksi penyakit *stroke*. Penelitian ini menggunakan *dataset* dari Kaggle yang terdiri dari 4.909 data dengan berbagai atribut seperti usia, jenis kelamin, hipertensi, penyakit jantung, status menikah, jenis pekerjaan, tempat tinggal, kadar gula darah, BMI, dan status merokok. Data tersebut diolah melalui beberapa tahap, mulai dari validasi, *preprocessing*, hingga pembagian data untuk pelatihan dan pengujian model menggunakan RapidMiner.

Hasil penelitian menunjukkan bahwa *algoritma* K-Nearest Neighbor (KNN) memberikan rata-rata akurasi terbaik sebesar 95,59%, sedangkan Naïve Bayes mencapai rata-rata akurasi 91,67%. Dengan demikian, KNN dinilai lebih unggul dalam klasifikasi penyakit *stroke* pada *dataset* yang digunakan. Studi ini menegaskan bahwa pemilihan *algoritma* yang tepat sangat penting dalam aplikasi machine learning untuk deteksi dini *stroke*, sehingga dapat membantu pengambilan keputusan medis secara lebih efektif dan efisien.

BAB III

METODOLOGI PENELITIAN

3.1 Analisa Kebutuhan

Penelitian ini bertujuan untuk membandingkan *algoritma Naïve Bayes* dengan *K-Nearest Neighbors* (KNN) dalam menganalisis *dataset* pasien *stroke*. Fokus utama penelitian ini adalah untuk mengidentifikasi pola-pola yang berkaitan dengan kejadian *stroke*. Diharapkan penerapan *algoritma Naïve Bayes* dan KNN dapat meningkatkan efektivitas deteksi dini *stroke*, sehingga dapat mendukung upaya pencegahan dan penanganan yang lebih tepat.

3.1.1 Analisa Kebutuhan Pemakai

Rendahnya kesadaran masyarakat terhadap kondisi kesehatan, khususnya terkait *stroke*, menyebabkan perlunya aplikasi yang dapat membantu pengguna dalam memprediksi resiko *stroke*. Berikut adalah beberapa kebutuhan yang diperlukan dalam aplikasi tersebut:

1. Menyajikan hasil prediksi
2. Memberikan hasil prediksi yang tepat dan dapat diandalkan
3. Memiliki antarmuka yang *user-friendly*
4. Aplikasi yang ringan dan tidak memberatkan perangkat

3.1.2 Analisa Kebutuhan Aplikasi

Berdasarkan analisis kebutuhan pengguna, dikembangkan aplikasi yang dirancang untuk memenuhi kebutuhan tersebut. Berikut ini adalah tabel yang menggambarkan kebutuhan aplikasi:

Tabel 3.1 Analisis kebutuhan aplikasi

No.	Analisis Kebutuhan Aplikasi Saya mengharapkan sistem dapat:	Keterangan
1	Memberikan prediksi resiko *stroke*	V
2	Memberikan rekomendasi gaya hidup sehat	V
3	Memberikan edukasi tentang *stroke*	V
4	Tampilan aplikasi mudah digunakan dan dipahami	V
5	Menampilkan gejala *stroke*	V
6	Memberikan saran jika terkena *stroke*	V

3.2 Pembahasan Metode Naïve Bayes dan KNN

3.2.1 *Algoritma* naïve bayes

Algoritma naïve bayes dapat diaplikasikan menggunakan python dengan total 500 data, terdiri melalui 250 data positif *stroke* dan 250 data negative *stroke*.

```python
# 1. Membaca dataset
df = pd.read_csv("balanced.csv")

# 2. Mapping kolom kategorikal
mappings = {
    "gender": {"Male": 1, "Female": 0, "Other": 2},
    "ever_married": {"Yes": 1, "No": 0},
    "work_type": {
        "Private": 0,
        "Self-employed": 1,
        "Govt_job": 2,
        "children": 3,
        "Never_worked": 4,
    },
    "Residence_type": {"Urban": 1, "Rural": 0},
    "smoking_status": {
        "never smoked": 0,
        "formerly smoked": 1,
        "smokes": 2,
        "Unknown": 3,
    },
}

for col, mapping in mappings.items():
    df[col] = df[col].map(mapping)

# 3. Hapus baris dengan nilai kosong
df = df.dropna()
```

Gambar 3.1 Mempersiapkan data (bagian 1)

```python
# 4. Cek distribusi kelas
print("Distribusi awal data stroke:\n", df['stroke'].value_counts())

# 5. Buat dataset seimbang
stroke_1 = df[df['stroke'] == 1]
stroke_0 = df[df['stroke'] == 0]

# Ambil sample terkecil dari dua kelas
sample_n = min(len(stroke_1), len(stroke_0))

# Ambil data secara acak dan seimbang
df_balanced = pd.concat([
    stroke_1.sample(sample_n, random_state=42),
    stroke_0.sample(sample_n, random_state=42)
]).sample(frac=1, random_state=42).reset_index(drop=True)

# 6. Pisahkan fitur dan label
X = df_balanced.drop('stroke', axis=1)
y = df_balanced['stroke']

# 7. Bagi data menjadi training dan testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 8. Melatih model Naive Bayes
model = GaussianNB()
model.fit(X_train, y_train)
```

Gambar 3.2 Mempersiapkan data (bagian 2)

Gambar 3.1 dan Gambar 3.2 merupakan step awal dari mempersiapkan data yang akan digunakan. Berikut penjelasannya:

1. Membaca data dari *file balanced.csv* ke dalam *DataFrame* untuk digunakan dalam proses *preprocessing* dan pelatihan model.
2. Mengubah nilai-nilai kategorikal menjadi representasi numerik agar bisa diproses oleh *algoritma machine learning*.
3. Menghapus seluruh baris yang memiliki nilai kosong (*missing values*) untuk menjaga kualitas data dan mencegah *error* saat pelatihan.
4. Menampilkan jumlah data untuk masing-masing kelas *stroke* (0 dan 1) sebelum diseimbangkan, guna melihat apakah terdapat ketidakseimbangan kelas.

5. Mengambil sampel data dari masing-masing kelas dalam jumlah yang sama untuk membentuk *dataset* yang seimbang antara *stroke* dan non-*stroke*.
6. Memisahkan kolom *stroke* sebagai label dan sisanya sebagai atribut untuk digunakan dalam pelatihan model.
7. Membagi data menjadi 80% untuk pelatihan (*training*) dan 20% untuk pengujian (*testing*) agar model bisa dievaluasi setelah dilatih.
8. Melatih *algoritma Naive Bayes* menggunakan data *training* yang telah disiapkan untuk membuat model klasifikasi.

```
# 9. Prediksi terhadap data test
y_pred = model.predict(X_test)

# 10. Evaluasi model
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# 11. Tampilkan 10 hasil prediksi
sample_data = X_test.iloc[:10].copy()
sample_data['Aktual'] = y_test.iloc[:10].values
sample_data['Prediksi'] = y_pred[:10]
sample_data.reset_index(drop=True, inplace=True)

print("\nHasil Lengkap Prediksi 10 Data Testing:")
print(sample_data)
```

Gambar 3.3 Prediksi menggunakan *naïve bayes*

9. Menggunakan model yang telah dilatih untuk memprediksi label (*stroke*) pada data uji, guna melihat performa model di data yang belum pernah dilihat sebelumnya.
10. Menampilkan *confusion matrix* dan *classification report* untuk mengevaluasi performa model, termasuk metrik seperti akurasi, presisi, *recall*, dan *F1-score*.

11. Menampilkan 10 data pertama dari data uji, beserta label sebenarnya (Aktual) dan hasil prediksi dari model (Prediksi), untuk melihat perbandingan langsung antara prediksi dan kenyataan.

```
Confusion Matrix:
[[31 12]
 [ 6 35]]

Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.72      0.78        43
           1       0.74      0.85      0.80        41

    accuracy                           0.79        84
   macro avg       0.79      0.79      0.79        84
weighted avg       0.79      0.79      0.78        84


Hasil Lengkap Prediksi 10 Data Testing:
     id  gender    age  hypertension  heart_disease  ever_married  work_type  Residence_type  avg_glucose_level   bmi  smoking_status  Aktual  Prediksi
0  2458       0  78.00             0              0             1          0               0             235.63  32.3               0       1         1
1  36338      0  39.00             1              0             1          0               0              58.09  39.2               2       1         0
2  10281      0  51.00             1              0             1          1               0             176.34  28.4               0       0         1
3  36236      1  80.00             1              0             1          0               1             240.09  27.0               0       1         1
4  25783      0   0.48             0              0             0          3               0              94.06  14.8               3       0         0
5  68074      1  54.00             0              0             1          0               0             100.47  50.2               1       0         0
6  62681      0  38.00             1              0             1          0               1             137.94  41.8               0       0         0
7  62019      1  54.00             0              0             1          2               0              87.85  31.1               2       1         0
8  60491      0  78.00             0              0             1          0               1              58.57  24.2               3       1         1
9  19508      0  26.00             0              0             0          0               1             116.68  18.7               1       0         0
```

Gambar 3.4 Hasil prediksi

Gambar 3.4 menampilkan hasil prediksi terhadap data uji yang digunakan. Terlihat bahwa pada baris ke-1, ke-2, dan ke-7 terdapat perbedaan antara nilai aktual dan hasil prediksi, yang menunjukkan bahwa model belum sepenuhnya akurat dalam memprediksi kasus *stroke*.

Selain itu, Python juga dapat dimanfaatkan untuk menganalisis faktor-faktor yang paling berpengaruh terhadap resiko penyakit *stroke*.

```python
feature_names = X.columns.tolist()
# Ambil rata-rata tiap fitur per kelas
means = model.theta_
# Hitung selisih rata-rata antar kelas
pengaruh = means[1] - means[0]  # stroke=1 - stroke=0
# Buat DataFrame hasil analisis
pengaruh_df = pd.DataFrame({
    'Atribut': feature_names,
    'Pengaruh': pengaruh
})
# 12. Urutkan berdasarkan besarnya pengaruh (positif atau negatif)
pengaruh_df = pengaruh_df.sort_values(by='Pengaruh', ascending=False).reset_index(drop=True)

print("\nFaktor Paling Berpengaruh terhadap Stroke:")
print(pengaruh_df)
```

Gambar 3.5 Faktor paling berpengaruh menggunakan python

12. Berfungsi untuk mengurutkan fitur berdasarkan nilai Pengaruh tertinggi ke terendah. Artinya, fitur yang paling membedakan antara *stroke* positif dan negatif akan muncul di atas. *Reset index* berguna agar *DataFrame* tampil rapih (mulai dari 0, 1, 2, ...), dan tidak menyimpan index lama yang mungkin acak setelah pengurutan.

```
Faktor Paling Berpengaruh terhadap Stroke:
              Atribut    Pengaruh
0                 age   26.995037
1   avg_glucose_level   25.159436
2                 bmi    1.715275
3        ever_married    0.248566
4        hypertension    0.219378
5       heart_disease    0.124211
6              gender    0.054934
7      Residence_type    0.011905
8      smoking_status   -0.284065
9           work_type   -0.457975
```

Gambar 3.6 Faktor paling berpengaruh menggunakan python

Berdasarkan gambar 3.6, faktor yang paling mempengaruhi terhadap *stroke* adalah

1. Usia (26.99) : Faktor dengan pengaruh tertinggi. Semakin tua usia seseorang, semakin tinggi resiko terkena *stroke*. Hal ini sejalan dengan fakta medis bahwa resiko *stroke* meningkat secara signifikan seiring bertambahnya umur.
2. Kadar Gula Darah (25.15) : Merupakan faktor terbesar kedua. Tingkat *avg_glucose_level* yang tinggi menunjukkan bahwa hiperglikemia berperan besar dalam memicu *stroke*, terutama pada penderita diabetes atau sindrom metabolik.
3. BMI (1.71) : Indeks massa tubuh menunjukkan adanya kaitan antara obesitas dan resiko *stroke*. Berat badan berlebih meningkatkan tekanan darah dan memperburuk sirkulasi, yang berkontribusi terhadap resiko *stroke*.

4. Status Pernikahan (0.25) : *ever_married* memiliki pengaruh sedang. Mungkin mencerminkan stabilitas sosial atau ekonomi, atau bisa juga sebagai *proxy* dari faktor usia.
5. Hipertensi (0.22) : Tekanan darah tinggi adalah salah satu faktor risiko utama *stroke* secara klinis. Nilai ini memperkuat pentingnya kontrol tekanan darah dalam pencegahan *stroke*.
6. Penyakit Jantung (0.12) : Riwayat penyakit jantung memiliki kontribusi terhadap *stroke*, karena gangguan kardiovaskular dapat menyebabkan penyumbatan aliran darah ke otak.
7. Jenis Kelamin (0.05) : Memiliki pengaruh yang kecil. Namun, beberapa studi menyebutkan laki-laki memiliki resiko *stroke* lebih tinggi pada usia muda dibanding perempuan.
8. Tipe Tempat Tinggal (0.01) : *Residence_type* memiliki pengaruh sangat kecil terhadap resiko *stroke*. Lingkungan urban atau rural tampaknya bukan penentu utama dalam model ini.
9. Status Merokok (-0.28) : Status merokok memberikan pengaruh negatif dalam model ini. Secara medis, merokok tetap merupakan faktor resiko *stroke* yang signifikan.
10. Jenis Pekerjaan (-0.46) : Faktor pekerjaan memiliki pengaruh negatif tertinggi, meskipun secara klinis korelasinya bisa bervariasi.

3.2.2 *Algoritma K-Nearest Neighbors* (KNN)

Algoritma KNN dapat diaplikasikan menggunakan python dengan total 500 data, terdiri atas 250 data positif *stroke* dan 250 data negatif *stroke*.

```
# 9. Melatih model K-Nearest Neighbors
model = KNeighborsClassifier(n_neighbors=5)  # k = 5 bisa disesuaikan
model.fit(X_train, y_train)
```

Gambar 3.7 Melatih model KNN

Gambar 3.7 berfungsi untuk memuat dan melatih model KNN dengan menggunakan data *training*

```
# 13. Analisis KNN untuk 1 data uji
test_index = 0
single_test = X_test[test_index].reshape(1, -1)
distances, indices = model.kneighbors(single_test, n_neighbors=5, return_distance=True)
print(f"\nAnalisis KNN untuk Data Uji ke-{test_index}:")
print(f"Prediksi Model      : {y_pred[test_index]}")
print(f"Label Sebenarnya    : {y_test.iloc[test_index]}")
print("\nTetangga Terdekat (k=5):")
print("Jarak\t\tLabel")

# Tampilkan jarak dan label dari tetangga terdekat
for i, (dist, idx) in enumerate(zip(distances[0], indices[0])):
    print(f"{dist:.4f}\t\t{y_train.iloc[idx]}")
```

Gambar 3.8 Menganalisa model KNN untuk 1 data uji

```
Analisis KNN untuk Data Uji ke-0:
Prediksi Model      : 1
Label Sebenarnya    : 1

Tetangga Terdekat (k=5):
Jarak           Label
1.8195          1
2.2249          1
2.6315          0
2.6491          0
2.7335          1
```

Gambar 3.9 Hasil dari analisis

Berdasarkan gambar 3.9, dapat diketahui bahwa prediksi *stroke* pada data uji ke-0 ditentukan oleh mayoritas tetangga terdekat yang memiliki label *stroke*. Dari 5 tetangga terdekat, 3 di antaranya merupakan penderita *stroke*, sehingga model memutuskan klasifikasi sebagai *stroke*.

3.3 Pembahasan *Algoritma Naïve Bayes* dan *K-Nearest Neighbors* (KNN)

3.3.1 Pengumpulan Data

Tujuan pengumpulan data dalam penelitian ini adalah untuk memperoleh informasi yang dibutuhkan dalam proses klasifikasi penyakit *stroke*. Data yang dikumpulkan mencakup rekam medis pasien, hasil pemeriksaan diagnostik, serta faktor-faktor seperti usia, jenis kelamin, riwayat keluarga, dan kebiasaan hidup yang dapat memengaruhi resiko *stroke*. Dengan data yang lengkap dan akurat, model yang dikembangkan diharapkan dapat memprediksi kemungkinan terjadinya *stroke* dengan lebih tepat.

Selain itu, pengumpulan data juga bertujuan untuk memastikan bahwa informasi yang diperoleh digunakan dengan cara yang sah dan etis, dengan persetujuan pasien, serta hanya untuk kepentingan penelitian. Data yang terkumpul akan menjadi dasar dalam pengembangan sistem yang dapat mendeteksi risiko *stroke* sejak dini dan memberikan rekomendasi pengobatan yang lebih tepat bagi pasien.

3.3.2 Preprocessing Data

Dataset yang digunakan pada penelitian ini diambil dari www.kaggle.com. *Dataset* ini juga terdiri dari lebih dari 5000 data yang mencakup berbagai atribut terkait faktor resiko *stroke*. Sebelum digunakan dalam analisis, *dataset* ini melalui tahap *preprocessing* untuk membersihkan dan menyiapkan data, yang mencakup penanganan data yang hilang, normalisasi, dan penghapusan outlier. Berikut adalah penjelasan mengenai atribut-atribut dalam *dataset* ini.

Tabel 3.2 Keterangan data

Atribut	Keterangan	Variabel
Age	Menunjukkan usia pasien (dalam tahun)	*Ineteger*
Gender	Menunjukkan jenis kelamin pasien	*Categorical*
Hypertension	Menunjukkan apakah pasien memiliki riwayat hipertens	*Integer*
Heart disease	Menunjukkan apakah pasien memiliki riwayat penyakit jantung	*Integer*
Ever married	Menunjukkan apakah pasien sudah menikah	*Integer*
Smoking	Menunjukkan apakah pasien merokok	*Integer*
Physical Activity	Menjelaskan apakah pasien aktif secara fisik	*Real*
BMI	Indeks massa tubuh pasien	*Real*
Stroke	Menunjukkan apakah pasien mengalami *stroke*	*Integer*

Berikut keterangan data:

1. *Gender*: Menunjukkan jenis kelamin individu (1 = Pria, 0 = Wanita)

   Min : 0

Max : 1

2. *Age*: Menunjukkan usia individu

   Min : 8

   Max : 82

3. *Hypertension*: Menunjukkan apakah individu memiliki riwayat hipertensi (1 = Ya, 0 = Tidak)

   Min : 0

   Max : 1

4. *Heart disease*: Menunjukkan apakah individu memiliki riwayat penyakit jantung (1 = Ya, 0 = Tidak)

   Min : 0

   Max : 1

5. *Ever married*: Menunjukkan apakah individu sudah menikah (1 = Ya, 0 = Belum menikah)

   Min : 0

   Max : 1

6. *Work type*: Menunjukkan jenis pekerjaan individu (1 = Bekerja, 0 = Tidak bekerja)

   Min : 0

   Max : 1

7. *Residence type*: Menunjukkan jenis tempat tinggal individu (1 = Perkotaan, 0 = Pedesaan)

   Min : 0

   Max : 1

8. *Average Glucose Level*: Menunjukkan kadar rata-rata gula darah individu

Min : 77.59

Max : 228.69

9. BMI: Menunjukkan indeks massa tubuh individu

   Min : 17.6

   Max : 54.66

10. *Stroke*: Menunjukkan apakah individu mengidap *Stroke* (1 = Ya, 0 = Tidak)

    Min : 0

    Max : 1

Tabel 3.3 *Dataset*

Gender	*Age*	*Hypertension*	Heart Desease	*Ever married*	*Work type*	*Residence type*	*AVG Glucose Level*	BMI	*Stroke*
1	67	0	1	1	1	1	228.69	36.6	1
0	61	0	0	1	0	0	202.21	0	1
1	80	0	1	1	1	0	105.92	32.5	1
0	82	1	1	0	1	0	84.03	26.5	1
0	69	0	0	0	1	1	94.39	22.8	1
0	52	0	0	1	1	1	77.59	17.7	0
1	58	1	0	1	1	1	87.96	39.2	0
0	8	0	0	0	1	1	110.89	17.6	0
1	72	0	1	1	0	0	97.53	29.4	0
0	74	1	0	1	0	1	205.84	54.6	0

3.3.3 Menentukan Nilai K

Dalam *algoritma K-Nearest Neighbor* (KNN), nilai K merujuk pada jumlah tetangga terdekat yang digunakan untuk mengklasifikasikan data. Pemilihan nilai K yang tepat sangat penting untuk mencapai keseimbangan antara akurasi dan kompleksitas model. Nilai K yang terlalu kecil bisa membuat model terlalu sensitif terhadap *noise* dalam data (*overfitting*), sementara nilai K yang terlalu besar bisa mengurangi kemampuan model untuk menangkap pola yang penting dan menyebabkan model terlalu sederhana (*underfitting*). Untuk menentukan nilai K yang optimal, biasanya digunakan teknik cross-validation untuk menguji berbagai nilai K dan memilih yang memberikan kinerja terbaik pada data yang belum pernah dilihat sebelumnya.

Tabel 3.4 Pengaruh pemilihan nilai K terhadap akurasi model KNN

Nilai K	Akurasi (%)	Deskripsi
1	90	*Overfitting* - Terlalu sensitif terhadap *noise*
3	92	Kinerja terbaik, model cukup fleksibel
5	91	Model stabil, tidak terlalu kompleks
7	85	*Underfitting* - Model terlalu sederhana
9	82	*Underfitting* - Tidak menangkap pola yang penting

3.3.3 Menghitung Jarak

Dalam analisis data terkait penyakit *stroke*, perhitungan jarak digunakan untuk mengukur kedekatan antara pasien berdasarkan faktor-faktor resiko medis mereka, seperti usia, tekanan darah, dan kolesterol. Salah satu cara umum untuk menghitung jarak antar pasien adalah dengan menggunakan jarak *Euclidean*, yang mengukur seberapa mirip dua pasien berdasarkan atribut medis mereka. Perhitungan jarak ini berguna untuk

mengklasifikasikan pasien dengan resiko *stroke* yang serupa dan membantu dalam menentukan kelompok resiko mereka (tinggi atau rendah).

Rumus:

$$d(X,Y) = \sqrt{(x1-y1)2+(x2-y2)2+(x3-y3)2+\cdots+(xn-yn)2}$$

Keterangan:

1. $x = (x1, x2, x3,..., x_n)$ adalah data pasien pertama (misalnya, Pasien 1) dengan beberapa atribut medis (seperti usia, tekanan darah, dll).
2. $Y = (y1, y2, y3,...,y_n)$) adalah data pasien kedua (misalnya, Pasien 2) dengan atribut medis yang sama.
3. $d(X,Y)$ adalah jarak *Euclidean* antara kedua pasien, yang menggambarkan tingkat kedekatan resiko *stroke* mereka.

Langkah-langkah:

$$d(X,Y) = \sqrt{(55-60)2+(130-140)2+(200-210)2}$$

$$d(X,Y) = \sqrt{(-5)2+(-10)2+(-10)2}$$

$$d(X,Y) = \sqrt{25+100+100} = \sqrt{255} = 15$$

Jadi, jarak Euclidean antara Pasien 1 dan Pasien 2 dihitung sebesar 15.

Berikut adalah tabel lengkap yang menunjukkan jarak Euclidean antara Pasien 2:

Tabel 3.5 Data pasien

Pasien	Usia	Tekanan Darah	Kolestrol	Jarak Euclidean ke P-2
P-1	55	130	200	15

P-2	60	140	210	15
P-3	50	120	180	-

3.3.4 Klasifikasi Data

Klasifikasi data merupakan suatu proses dalam pembelajaran mesin (*machine learning*) yang bertujuan untuk membagi atau mengelompokkan data ke dalam kategori atau label tertentu berdasarkan fitur atau atribut yang dimilikinya. Tujuan utama dari klasifikasi adalah untuk memprediksi kategori atau label dari data baru, dengan menggunakan model yang telah dilatih sebelumnya menggunakan data yang sudah ada (data latih).

1. Akurasi yang mengukur sejauh mana model dapat membuat prediksi label yang benar, yaitu berapa proporsi prediksi yang benar dari seluruh prediksi yang dibuat.

   Rumus:

$$Akurasi = \frac{Jumlah\ Prediksi\ Benar}{Jumlah\ Total\ Prediksi} = \frac{TP+TN}{TP+TN+FP+FN}$$

   1. TP (*True Positive*): Jumlah prediksi benar untuk kelas positif
   2. TN (*True Negative*): Jumlah prediksi benar untuk kelas negatif
   3. FP (*False Positive*): Jumlah prediksi salah untuk kelas positif
   4. FN (*False Negative*): Jumlah prediksi salah untuk kelas negatif

   Contoh pada perhitungan dari akurasi:

   Data:

   - TP = 50
   - TN = 30

- FP = 10
- FN = 5

$$Akurasi = \frac{50+30}{50+30+10+5}$$

$$Akurasi = \frac{80}{95}$$

Akurasi = 0.8421 atau 84.21%

Hasil pada perhitungan akurasi mendapatkan nilai 0.8421 atau sekitar 84.21%

Kelebihan: Mudah dihitung dan dipahami

Kekurangan: Akurasi bisa menyesatkan jika data tidak seimbang (misalnya, jika jumlah data untuk satu kelas jauh lebih banyak daripada kelas lainnya).

2. *Precision* (Presisi) mengukur seberapa akurat model dalam memprediksi kelas positif. Ini penting ketika biaya kesalahan dalam memprediksi kelas positif sangat tinggi (misalnya dalam deteksi spam, deteksi penyakit, dll).

   Rumus:

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

   1. TP (*True Positive*): Jumlah prediksi benar untuk kelas positif
   2. FP (*False Positive*): Jumlah prediksi salah untuk kelas positif

   Untuk dilakukannya perhitungannya, seperti:

   - TP = 55
   - FP = 60

$$Precision = \frac{55}{55+60} = \frac{55}{115} \approx 0.478$$

Jadi pada *precision* dari perhitungan terhadap data tersebut sekitar 0.478 atau sekitar 47.8%

*Int*erpretasi: *Precision* menunjukkan proporsi prediksi yang benar-benar positif di antara semua prediksi yang diklasifikasikan sebagai positif

Kelebihan: Berguna mau meminimalkan jumlah *false positive* (kesalahan dalam mengklasifikasikan negatif sebagai positif)

3. *Recall* (Sensitivitas atau *True Positive Rate*) mengukur seberapa bagus model dalam menemukan semua contoh kelas positif. Ini penting ketika biaya kesalahan dalam memprediksi kelas negatif sangat tinggi (misalnya dalam diagnosis penyakit, kita mau mendeteksi sebanyak mungkin kasus positif).

   Rumus:

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

   1. TP (*True Positive*): Jumlah prediksi benar untuk kelas positif
   2. FN (*False Negative*): Jumlah prediksi salah untuk kelas positif

   Data yang mau di lakukan perhitungan:

   - TP = 80
   - FP = 20

$$Precision = \frac{80}{80 + 20} = \frac{80}{100} = 0.8$$

   Jadi *recall* pada perhitungan tersebut adalah 0.8 atau 80%. Dengan *recall* sebesar 80%, model ini berhasil mendeteksi 80% dari semua kasus yang seharusnya positif.

*Int*erpretasi: *Recall* menunjukkan proporsi kasus positif yang berhasil ditemukan oleh model

Kelebihan: Berguna ketika mau meminimalkan jumlah *false negative* (kesalahan dalam mengklasifikasikan positif sebagai negatif).

4. *F1-Score* adalah harmonik rata-rata antara *precision* dan *recall*. *F1-Score* digunakan ketika mau mengimbangkan antara *precision* dan *recall*, terutama dalam kasus di mana keduanya penting, tetapi *trade-off* antara keduanya diperlukan.

   Rumus:

$$F1 - Score = 2 + \frac{Precision\ x\ Recall}{Precision + Recall}$$

   Data yang akan dihitung dapat diambil dari hasil perhitungan *recall* dan *precision*:

   - *Precision* = 0.8
   - *Recall* = 0.75

   Kemudian di hitung:

$$F1 - Score = 2 + \frac{0.8\ x\ 0.75}{0.8 + 0.75}$$

   Langkah pertama, dikalikan terlebih dahulu:

$$0.8\ x\ 0.75 = 0.6$$

   Kemudian jumlahkan:

$$0.8 + 0.75 = 1.55$$

   Lalu dibagi hasil dari perkaliannya dengan jumlah *precision* dan *recall*:

$$\frac{0.6}{1.55} \approx 0.387$$

   Step selanjutnya, dikalikan dengan 2:

$$F1 - Score = 2\ x\ 0.387 \approx 0.774$$

Untuk hasil penilaian pada *precision* = 0.8 dan *recall* = 0.75 adalah 0.774 atau sekitar 77.4%

*Int*erpretasi: *F1-Score* memberikan nilai antara 0 dan 1, dengan 1 berarti model memiliki precision dan recall yang sempurna. F1 lebih rendah jika *precision* dan *recall* sangat berbeda

Kelebihan: Bagus digunakan ketika memiliki data yang tidak seimbang atau ketika kita perlu menyeimbangkan antara *precision* dan *recall*

5. *Confusion Matrix* adalah tabel yang menunjukkan jumlah prediksi yang benar dan salah untuk setiap kelas. Ini memberikan gambaran yang lebih jelas tentang seberapa baik model bekerja, terutama dalam konteks klasifikasi dengan banyak kelas.

   Confusion matrix untuk dua kelas (misalnya positif dan negatif):

Tabel 3.6 *Confusion matrix* untuk dua kelas

Actual/Predicted	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Dari tabel *confusion matrix* diatas, kita dapat menghitung semua metrik evaluasi seperti *precision*, *recall*, dan *F1-score*.

Menghitung *confusion matrix* berdasarkan dari *precision*, *recall*, dan *F1-Score*:

a. *Precision*

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{80}{80 + 30} = \frac{80}{110} \approx 0.727$$

Precision = 0.727 atau 72.7%

b. *Recall*

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{80}{80 + 20} = \frac{80}{100} \approx 0.8$$

Recall = 0.8 atau 80%

c. *F1-Score*

$$Precision = 2\ x\ \frac{0.727\ x\ 0.8}{0.727 + 0.8} = 2\ x\ \frac{0.5816}{1.527} \approx 0.762$$

F1-Score = 0.762 atau 76.2%

Kesimpulan yang dapat diambil dari perihitungan diatas ialah *confusion matrix* menghitung *precision* sebesar 76.7%, *recall* sebesar 80%, dan *F1-Score* sebesar 76.2%. *Precision* dan *recall* memberikan gambaran yang berbeda, yang dimana *precision* menunjukkan tingkat kesalahan dalam memprediksi positif, sementara *recall* menunjukkan seberapa banyak kasus positif yang berhasil dideteksi oleh model. *F1-Score* merupakan metrik yang berguna untuk menyeimbangkan *precision* dan *recall* terutama ketika keduanya penting untuk digunakan dalam mendeteksi penyakit.

6. ROC *Curve* dan AUC (*Area Under the Curve*)

   1. ROC *Curve* (*Receiver Operating Characteristic Curve*) adalah grafik yang menggambarkan kinerja model klasifikasi dengan menunjukkan *True Positive Rate* (*Recall*) terhadap *False Positive Rate* (*FPR*) pada berbagai *threshold* klasifikasi.
   2. AUC (*Area Under the Curve*) mengukur area di bawah ROC *curve*. AUC memberikan gambaran keseluruhan tentang kinerja model klasifikasi, di mana nilai AUC yang lebih tinggi menunjukkan bahwa model memiliki kemampuan lebih baik dalam membedakan antara kelas positif dan negatif.

Rumus AUC: AUC dihitung berdasarkan *int*egral dari ROC *curve*, namun pada dasarnya AUC = Probabilitas bahwa model akan memberikan peringkat lebih tinggi untuk contoh positif dibandingkan contoh negatif.

*Int*erpretasi:

a. AUC = 0.5: Model tidak lebih baik dari tebakan acak
b. AUC = 1: Model sempurna dalam membedakan kelas positif dan negatif.

Rumus pada UAC:

$$AUC = \frac{1}{2}\,(TPR_1 + TPR_2)\;x\;(FPR_2 + FPR_1)$$

Dimana:

1. TPR_1 dan TPR_2 adalah nilai TPR pada 2 titik *threshold* yang berbeda
2. FPR_1 dan FPR_2 adalah nilai FPR pada 2 titik *threshold* yang berbeda

Data yang akan dihitung:

- TPR (*True Positive Rate*) / *recall*:

$$TPR = \frac{TP}{TP + FN} = \frac{80}{80 + 20} = \frac{80}{100} = 0.8$$

- FPR (*False Positive Rate*):

$$FPR = \frac{FP}{FP + TN} = \frac{30}{30 + 70} = \frac{30}{100} = 0.3$$

Kemudian dihitung dengan menggunakan rumus UAC:

$$AUC = \frac{1}{2}\,(0 + 0.8)\;x\;(0.3 - 0) = \frac{1}{2}\;x\;0.8\;x\;0.3 = 0.12$$

AUC menghitung lebih akurat dengan metode numerik dan menghitung untuk banyak titik di ROC *curve* dan bukan hanya dua titik saja.

7. *Log Loss* (*Logarithmic Loss*) adalah metrik evaluasi yang mengukur seberapa baik probabilitas yang diprediksi oleh model mendekati label yang sebenarnya. Ini lebih cocok untuk model yang menghasilkan probabilitas klasifikasi (misalnya regresi logistik), namun juga dapat digunakan untuk *algoritma* lain.

   Rumus:

$$Log\ Loss = -\frac{1}{N} + \sum_{i=1}^{N} (y_i \log(p_i) + (1 - y_i) \log\ (1 - p_i))$$

   Keterangan:

   a. y_i adalah label yang sebenarnya (0 atau 1)

   b. p_i adalah probabilitas yang diprediksi untuk kelas positif

   c. N adalah jumlah total sampel

   *Int*erpretasi: Semakin kecil nilai *log loss*, semakin baik model. *Log Loss* menganggap probabilitas prediksi sebagai bagian dari evaluasi, bukan hanya hasil klasifikasi biner.

Menggabungkan data:

Tabel 3.7 Perhitungan *Log Loss*

Data	y_i (Label Asli)	p_i (Probabilitas Prediksi)
1	1	0.9
2	0	0.1
3	1	0.6
4	0	0.2

Data pertama:

$$Log\ Loss\ Data\ 1 = 1\ x\ log(0.9) + (1-1)\ x log(1-0.9) = log(0.9)$$

$$= -0.1054$$

Data kedua:

$$Log\ Loss\ Data\ 2 = 0\ x\ log(0.1) + (1-0)\ x log(1-0.1) = log(0.9)$$

$$= -0.1054$$

Data ketiga:

$$Log\ Loss\ Data\ 3 = 1\ x\ log(0.6) + (1-1)\ x log(1-0.6) = log(0.6)$$

$$= -0.5108$$

Data keempat:

$$Log\ Loss\ Data\ 4 = 0\ x\ log(0.2) + (1-0)\ x log(1-0.2) = log(0.8)$$

$$= -0.2231$$

Total *Log Loss*:

$$Total\ Log\ Loss = (-0.1054) + (-0.1054) + (-0.5108) + (-0.2231)$$

$$= -0.9447$$

Membagi hasil penjumlahan dengan jumlah Data $N = 4$:

$$Log\ Loss = -\frac{1}{4}\ x\ (-0.9447) = 0.2362$$

Nilai pada *Log Loss* untuk pada data ini adalah 0.2362 artinya model memprediksi dengan bagus, tetapi masih ada ruang untuk perbaikan dalam memprediksi probabilitas yang lebih akurat untuk setiap kelas.

8. *K-fold Cross Validation* teknik yang digunakan untuk menilai kinerja model pada *dataset* yang terbatas. Data dibagi menjadi K *subset* (*folds*), dan model dilatih K kali, dengan setiap *fold* digunakan sekali sebagai data uji dan sisanya sebagai data latih.

Langkah-langkah *K-fold Cross Validation*:

1. Data dibagi menjadi K bagian (*fold*)
2. Model dilatih pada K-1 bagian dan di uji pada bagian yang tersisa
3. Proses ini diulang K kali dengan masing-masing *fold* menjadi data uji
4. Hasil evaluasi dari setiap iterasi diambil rata-ratanya untuk mendapatkan estimasi kinerja yang lebih *robust*

Rumus:

$$Rata - rata\ Akurasi = \frac{1}{K} + \sum_{i=1}^{K} Akurasi\ pada\ iterasi\ i$$

Perhitungan pada *K-fold Cross Validation*:

Langkah 1, membagi *dataset* menjadi 5 *folds*:

- Iterasi 1: pada *fold* 2, 3, 4, 5; uji pada *fold* 1
- Itarasi 2: pada *fold* 1, 3, 4, 5; uji pada *fold* 2
- Iterasi 3: pada *fold* 1, 2, 4, 5; uji pada *fold* 3
- Iterasi 4: pada *fold* 1, 2, 3, 5; uji pada *fold* 4
- Iterasi 5: pada *fold* 1, 2, 3, 4; uji pada *fold* 5

Langkah 2, akurasi setiap *fold*:

- Iterasi 1: 85%
- Iterasi 2: 88%
- Iterasi 3: 90%
- Iterasi 4: 87%
- Iterasi 5: 92%

Langkah 3, menghitung rata-rata akurasi:

$$Rata - rata\ Akurasi = \frac{85\% + 88\% + 90\% + 87\% + 92\%}{5} = 88.4\%$$

Kelebihan: Mengurangi bias evaluasi karena data digunakan lebih dari satu kali untuk pelatihan dan pengujian.

3.3.5 Evaluasi Hasil

Dalam klasifikasi model pembelajaran mesin, seperti *K-Nearest Neighbors* (KNN), sangat penting untuk menilai seberapa baik model dalam memprediksi data yang tidak terlihat (data uji). Evaluasi ini tidak hanya melihat akurasi, tetapi juga melibatkan berbagai metrik lainnya untuk memastikan model berfungsi dengan baik sesuai tujuan yang diinginkan. Berikut adalah langkah-langkah yang dapat diambil untuk melakukan evaluasi hasil setelah model klasifikasi seperti KNN dilatih dan diuji.

1. Setelah model KNN dilatih dengan data latih (*training data*), model diuji menggunakan data uji (testing data) untuk mengukur kinerjanya. Data uji harus berbeda dari data latih, agar model bisa diuji pada data yang belum pernah dilihat sebelumnya.

   Langkah-langkah:

   a. Pisahkan data menjadi dua bagian: data latih dan data uji
   b. Gunakan data latih untuk melatih model
   c. Uji model dengan data uji untuk menghitung metrik evaluasi

2. *Confusion Matrix* adalah alat yang berguna untuk mengevaluasi hasil klasifikasi dengan memberikan gambaran lebih jelas tentang bagaimana model mengklasifikasikan data. Matriks ini menunjukkan jumlah prediksi yang benar dan salah untuk setiap kelas.

   Confusion matrix untuk klasifikasi dua kelas:

Tabel 3.8 *Confusion matrix* untuk klasifikasi dua kelas

	Prediksi Positif (1)	Prediksi Negatif (0)
Label Positif (1)	*True Positive* (TP)	*False Negative* (FN)
Label Negatif (0)	*False Positive* (FP)	*True Negative* (TN)

Penjelasan:

a. TP (True Positive): Prediksi benar untuk kelas positif

b. TN (True Negative): Prediksi benar untuk kelas negatif

c. FP (False Positive): Prediksi salah untuk kelas positif

d. FN (False Negative): Prediksi salah untuk kelas negatif

Dari *confusion matrix*, kita bisa menghitung berbagai metrik evaluasi.

3. *Matrix* Kinerja Model

a. Akurasi

Akurasi mengukur seberapa banyak prediksi yang benar dibandingkan dengan total prediksi yang dibuat. Meskipun ini metrik yang paling umum, akurasi bisa tidak terlalu menggambarkan performa model dengan baik jika data yang digunakan tidak seimbang.

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

b. *Precision* (Presisi)

Precision mengukur berapa banyak prediksi positif yang benar-benar positif.

$$Precision = \frac{TP}{TP + FP}$$

c. *Recall* (Sensitivitas)

Recall mengukur seberapa banyak kasus positif yang berhasil ditemukan oleh model.

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

d. *F1-Score*

F1-Score adalah rata-rata harmonis antara *precision* dan *recall*, memberikan keseimbangan antara kedua metrik tersebut. Ini berguna ketika keduanya penting jika mau menyeimbangkan keduanya.

$$F1 - Score = 2\ x\ \frac{Precision\ x\ Recall}{Precision + Recall}$$

e. ROC *Curve* dan AUC (*Area Under the Curve*)

Receiver Operating Characteristic (ROC) *curve* menggambarkan hubungan antara *True Positive Rate* (*Recall*) dan *False Positive Rate* pada berbagai *threshold* klasifikasi.

AUC (*Area Under the Curve*) mengukur luas area di bawah kurva ROC. AUC yang lebih tinggi menunjukkan bahwa model lebih baik dalam membedakan antara kelas positif dan negatif.

- AUC = 0.5: Model tidak lebih baik dari tebakan acak
- AUC = 1: Model sempurna

f. *Log Loss*

Log Loss mengukur seberapa baik probabilitas yang diprediksi model mendekati nilai yang sebenarnya. Ini lebih sering digunakan jika model menghasilkan probabilitas klasifikasi (misalnya regresi logistik atau SVM dengan probabilitas).

$$Log\ Loss = -\frac{1}{N} + \sum_{i=1}^{N}(y_i \log(p_i) + (1 - y_i)\log\ (1 - p_i))$$

Di mana:

- y_i adalah label yang sebenarnya (0 atau 1)
- p_i adalah probabilitas yang diprediksi untuk kelas positif
- N adalah jumlah total sampel

*Int*erpretasi: Semakin kecil nilai *Log Loss*, semakin bagus model.

4. *Cross-Validation* untuk Menghindari *Overfitting* dan *Underfitting*

   Untuk mendapatkan estimasi yang lebih baik tentang kinerja model, terutama jika data terbatas, gunakan *cross-validation* (terutama *K-fold cross-validation*). Teknik ini membagi data menjadi beberapa bagian (*folds*), kemudian melatih dan menguji model dengan data yang berbeda-beda, sehingga bisa mendapatkan gambaran kinerja model yang lebih stabil.

   Langkah-langkah *K-fold cross-validation*:

   - Bagi data menjadi **K** bagian (*folds*)
   - Latih model menggunakan *K-1 folds*, uji dengan fold yang tersisa
   - Ulangi langkah ini untuk semua fold, agar setiap fold menjadi data uji satu kali
   - Hitung rata-rata nilai evaluasi dari semua *fold*

5. Visualisasi Kinerja Model

   Untuk memahami kinerja model secara lebih mendalam, ada nya kita bisa menggunakan visualisasi berikut:

   - Visualisasi *Confusion Matrix*, menampilkan *confusion matrix* dalam bentuk *heatmap* untuk lebih jelas melihat distribusi prediksi benar dan salah.

- ROC *Curve*, memvisualisasikan ROC curve untuk mengevaluasi *trade-off* antara *recall* dan *false positive rate*.
- *Precision-Recall Curve*, menggambarkan hubungan antara *precision* dan *recall* pada berbagai threshold.

6. Perbandingan dengan Model Lain

   Evaluasi hasil model juga bisa dilakukan dengan cara membandingkan model KNN dengan model klasifikasi lainnya (misalnya *Decision Trees*, *Support Vector Machines* (SVM), atau *Random Forest*). Dengan membandingkan beberapa model, bisa mendapatkan gambaran mana yang bekerja lebih baik untuk masalah yang dihadapi.

7. Penyempurnaan Model (*Hyperparameter Tuning*)

   Setelah mengevaluasi model, mungkin akan menemukan bahwa model bisa lebih bagus lagi. Beberapa langkah untuk memperbaiki model adalah:

   a. Pemilihan nilai k yang optimal, gunakan *Grid Search* atau *Random Search* untuk mencari nilai KKK terbaik.
   b. Eksperimen dengan metrik jarak, cobalah berbagai metrik jarak (*Euclidean*, *Manhattan*, *Minkowski*, dll).
   c. Normalisasi Data, pastikan data telah dinormalisasi atau distandarisasi agar model tidak bias terhadap fitur dengan skala yang lebih besar.

8. Kesimpulan Evaluasi Hasil

   Evaluasi hasil dari model klasifikasi melibatkan pengukuran kinerja menggunakan berbagai metrik seperti akurasi, *precision*, *recall*, *F1-score*, AUC, dan *log loss*. Teknik seperti *cross-validation* bisa digunakan untuk memberikan gambaran yang lebih stabil tentang performa model. Dengan menggunakan visualisasi seperti ROC *curve* dan *confusion matrix*, agar bisa

lebih memahami kelebihan dan kekurangan model. Jika hasil model tidak memadai, langkah perbaikan bisa dilakukan melalui peningkatan data, pemilihan fitur, atau penyesuaian parameter model. Semua evaluasi ini membantu dalam keputusan pengembangan model lebih lanjut.

3.4 CRISP-DM

Dalam penelitian ini, penulis menggunakan metode CRISP-DM (*Cross Industry Standard Process for Data Mining*) sebagai acuan utama dalam membangun model klasifikasi. CRISP-DM merupakan metodologi standar yang banyak digunakan dalam proyek data mining karena memberikan panduan langkah demi langkah, mulai dari memahami permasalahan hingga penerapan solusi.

Metode ini terdiri dari enam tahap utama: *Business Understanding*, *Data Understanding*, *Data Preparation*, *Modeling*, *Evaluation*, dan *Deployment*. Keenam tahap ini tidak kaku dan dapat dilakukan secara iteratif, artinya peneliti bisa kembali ke tahap sebelumnya apabila diperlukan.

1. *Business Understanding*

   Tahap pertama adalah memahami secara jelas masalah yang ingin diselesaikan dan tujuan akhir penelitian. Permasalahan yang diangkat adalah *stroke*. *Stroke* merupakan salah satu penyakit berbahaya yang dapat menyebabkan kematian atau kecacatan permanen. Banyak kasus *stroke* tidak terdeteksi sejak dini karena masyarakat kurang mengenali faktor risiko dan gejala awal. Akibatnya, tindakan pencegahan sering terlambat dilakukan. Tujuan penelitian ini, sebagai berikut.

   1. Mengembangkan model prediksi yang dapat mengidentifikasi resiko *stroke* berdasarkan data pasien.

2. Membandingkan performa dua *algoritma* populer dalam klasifikasi, yaitu *K-Nearest Neighbor* (KNN) dan *Naïve Bayes*, untuk menentukan metode yang paling efektif.
3. Memberikan rekomendasi faktor resiko yang perlu diwaspadai, sehingga dapat mendukung upaya pencegahan.

Alasan mengenai pentingnya tahap ini, memastikan bahwa semua proses selanjutnya selaras dengan tujuan penelitian. Tanpa pemahaman yang jelas, hasil yang diperoleh mungkin tidak relevan atau tidak dapat digunakan.

2. *Business Understanding*

Setelah masalah dipahami, langkah berikutnya adalah mengumpulkan dan mempelajari data yang akan digunakan. Sumber data ini diperoleh dari platform Kaggle dengan judul *Healthcare Dataset Stroke Data*. *Dataset* ini memuat informasi kesehatan pasien, meliputi data demografis, riwayat medis, dan gaya hidup. Berikut adalah ringkasan *dataset*:

1. Jumlah data memiliki 5.110 baris x 12 kolom
2. Atribut target adalah *stroke* (1 = positif, 0 = tidak)
3. Distribusi kelasnya memiliki:
    - 4.861 pasien (95,13%) tidak terkena *stroke*
    - 249 pasien (4,87%) terkena *stroke*
4. Untuk nilai yang kosong hanya meliputi pada bagian bmi yang memiliki 201 data kosong (3,39%), sementara kolom lain lengkap.

Atribut yang ada pada *dataset*:

1. *Gender*: Jenis kelamin (*Female* dan *Male*)
2. *Age*: Usia pasien (0,08 – 82 tahun)
3. *Hypertension*: Riwayat hipertensi (0 = tidak, 1 = ya)
4. *Heart_disease*: Riwayat penyakit jantung (0 = tidak, 1 = ya)
5. *Ever_married*: Status pernikahan (*Yes* dan *No*)
6. *Work_type*: Jenis pekerjaan (*Private*, *Self-employeed*, *Govt_job*, *children*, *Never-worked*)
7. *Residence_type*: Tipe tempat tinggal (*Urban* dan *Rural*)
8. *Avg_glucose_level*: Kadar gula darah rata-rata (55,12 – 271,74 mg/dL)
9. *Bmi*: Indeks massa tubuh (10,30 – 97,60)
10. *Stroke*: Label target

Pentingnya tahap ini karena dapat membantu peneliti memahami karakteristik data, menemukan masalah seperti *missing values*, dan mengetahui distribusi target sebelum melanjutkan ke tahap persiapan data.

3. *Data Preparation* (Persiapan data)

Pada tahap ini, peniliti dapat mempersiapkan data agar layak dan optimal digunakan oleh *algoritma machine learning*. Langkah-langkah yang dilakukan, sebagai berikut.

1. Pembersihan data (*Data Cleaning*) bertujuan:
    1. Menghapus baris yang memiliki nilai kosong pada kolom bmi.

2. Tujuannya untuk menghindari bias hasil dan error saat pemodelan.

2. Transformasi data kategorikal, bertujuan:
    1. Mengubah data kategorikal menjadi numerik dengan teknik *encoding* (misalnya *Label Encoding* atau *One-Hot Encoding*).
    2. Contoh: *gender* → *Male* = 1, *Female* = 0.
3. Penyeimbangan data (*Data balancing*), bertujuan:
    1. Menggunakan *undersampling* pada kelas mayoritas (tidak terkena *stroke*) sehingga proporsi data menjadi seimbang (250 pasien *stroke* : 250 pasien non-*stroke*).
    2. Hal ini mencegah model terlalu condong memprediksi kelas mayoritas.
4. Pembagian *dataset* (*Data splitting*), bertujuan:
    1. 80% data digunakan untuk pelatihan (*training set*).
    2. 20% data digunakan untuk pengujian (*testing set*).

Pentingnya pada tahap ini, jika data tidak disiapkan dengan baik, hasil model bisa menyesatkan. Pembersihan dan penyeimbangan data sangat berpengaruh terhadap akurasi dan keadilan (*fairness*) model.

4. *Modeling* (Pembuatan Model)

Pada tahap ini, data yang telah dipersiapkan digunakan untuk membangun model klasifikasi. *Algoritma* yang digunakan, sebagai berikut.

1. *Naïve Bayes*

1. Menggunakan prinsip probabilitas untuk menentukan kelas target.
2. Asumsi setiap atribut tidak saling bergantung.
3. Cocok untuk *dataset* dengan fitur yang bervariasi.

2. *K-Nearest Neighbor* (KNN)
    1. Mengklasifikasikan data berdasarkan kedekatan (*similarity*) dengan K tetangga terdekat.
    2. Menggunakan jarak *Euclidean* untuk mengukur kedekatan.
    3. Dilakukan pengujian dengan berbagai nilai K untuk menemukan performa terbaik.

Pentingnya dibagian tahap ini karena pemilihan *algoritma* yang tepat mempengaruhi kemampuan sistem dalam membuat prediksi yang akurat. Menggunakan dua *algoritma* memungkinkan perbandingan yang objektif.

5. *Evaluation* (Evaluasi Model)

Model yang telah dibuat diuji menggunakan metrik evaluasi untuk mengukur kinerjanya:

1. *Accuracy*, persentase prediksi benar dari seluruh data uji.
2. *Precision*, tingkat ketepatan prediksi positif.
3. *Recall*, kemampuan model mendeteksi seluruh kasus positif.
4. *Confusion Matrix*, menampilkan detail prediksi benar dan salah per kelas.

Evaluasi membantu memastikan bahwa model tidak hanya akurat secara keseluruhan, tetapi juga efektif dalam mendeteksi kasus *stroke* yang jumlahnya relatif sedikit.

6. *Deployment* (Penerapan Model)

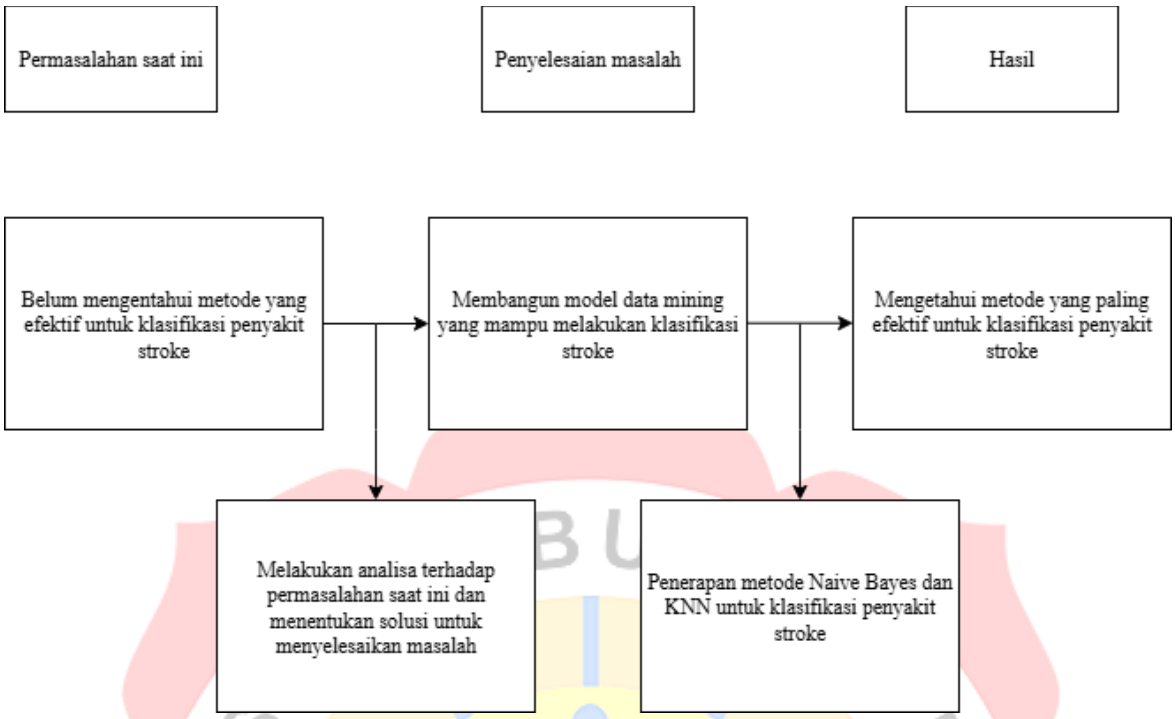
   Tahap akhir adalah penerapan model ke dalam bentuk aplikasi berbasis web.

   1. Aplikasi ini memungkinkan pengguna memasukkan data kesehatan mereka dan mendapatkan prediksi resiko *stroke* secara instan.
   2. Hasil prediksi disertai informasi edukatif tentang pencegahan *stroke*.
   3. Antarmuka dibuat sederhana agar dapat digunakan oleh masyarakat umum maupun tenaga medis.

   Hasil penelitian tidak hanya berhenti pada analisis, tetapi benar-benar memberikan manfaat nyata kepada pengguna.

3.5 Kerangka Pemikiran

Kerangka pemikiran adalah pola atau alur logis yang digunakan untuk menyelesaikan suatu permasalahan. Kerangka ini berfungsi sebagai panduan dalam merancang langkah-langkah sistematis yang diperlukan untuk mencapai tujuan tertentu. Biasanya, kerangka pemikiran disajikan dalam bentuk diagram atau narasi yang menggambarkan keterkaitan antara berbagai elemen yang relevan dengan permasalahan yang dihadapi.



Gambar 3.10 Kerangka Pemikiran

Kerangka pemikiran pada gambar 3.10 menunjukkan alur proses penyelesaian masalah dalam klasifikasi penyakit *stroke* menggunakan metode *naïve bayes* dan KNN. Berikut adalah penjelasan berdasarkan elemen-elemen dalam kerangka tersebut:

1. Permasalahan saat ini

   Belum mengetahui metode yang efektif untuk mengklasifikasikan penyakit *stroke*. Pada tahap ini, masalah utama yang dihadapi adalah tidak adanya metode yang terbukti efektif dalam melakukan klasifikasi penyakit *stroke* berdasarkan data medis. Oleh karena itu, diperlukan pendekatan yang dapat memprediksi risiko *stroke* pada individu berdasarkan atribut kesehatan tertentu.

2. Penyelesaian masalah

   Membangun model *data mining* yang mampu melakukan klasifikasi terhadap data medis pada penyakit *stroke*. Pada tahap ini, akan dibangun model data mining menggunakan *algoritma naïve bayes* dan KNN untuk menyelesaikan

masalah. Langkah pertama adalah merancang model klasifikasi dengan mengolah *dataset* yang berisi data kesehatan dan pola-pola yang relevan dengan resiko *stroke*. Metode *naïve bayes* dan KNN akan diterapkan untuk menentukan pola yang dapat memberikan prediksi akurat terkait resiko *stroke*.

3. Hasil

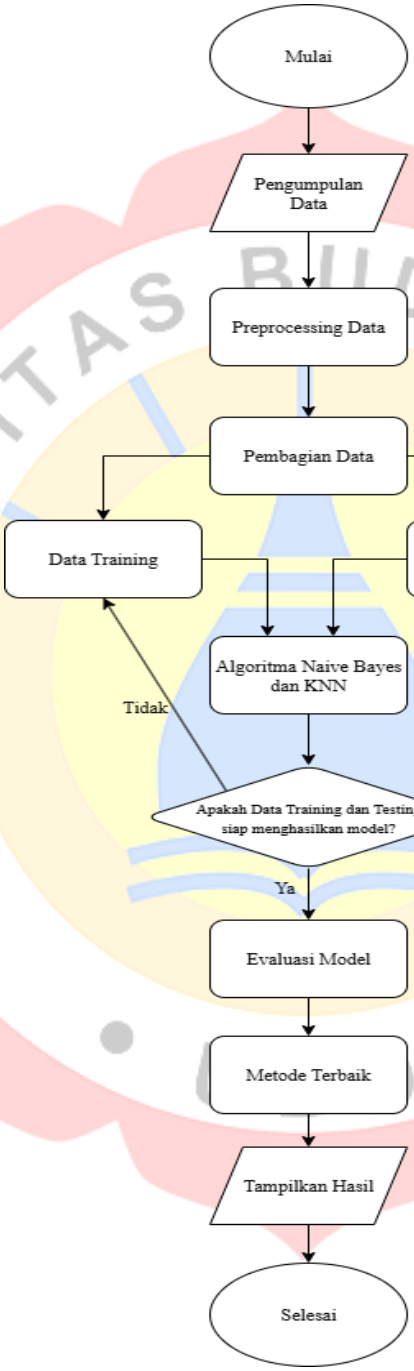
   Mengetahui metode yang paling efektif untuk mengklasifikasikan penyakit *stroke*. Setelah model dilatih dan diuji, diharapkan akan diperoleh metode klasifikasi yang paling efektif. Dengan membandingkan kinerja *algoritma naïve bayes* dan KNN, hasilnya akan menunjukkan *algoritma* mana yang lebih baik dalam mengklasifikasikan resiko *stroke* berdasarkan *dataset* yang digunakan.

4. Tahapan

   - Melakukan analisis terhadap permasalahan saat ini dan menentukan solusi untuk menyelesaikan masalah. Tahapan awal melibatkan pemahaman mendalam terhadap *dataset* medis terkait *stroke*, seperti data demografi, riwayat kesehatan, dan faktor risiko lainnya. Selanjutnya dilakukan evaluasi untuk memilih *algoritma* yang paling sesuai dalam mengklasifikasikan resiko *stroke*. Dalam hal ini, *algoritma naïve bayes* dan KNN dipilih sebagai pendekatan yang sesuai.
   - Penerapan metode *naïve bayes* dan KNN untuk mengklasifikasikan penyakit *stroke*. Pada tahap ini, *algoritma naïve bayes* akan digunakan untuk mempelajari *dataset* dan menghitung probabilitas resiko *stroke* berdasarkan atribut tertentu. Sementara itu, *algoritma* KNN akan mengklasifikasikan resiko *stroke* dengan menghitung kedekatan (*similarity*) antara individu yang akan diprediksi dengan data-data pada

dataset. Setelah implementasi, hasil dari kedua metode ini akan dibandingkan untuk menentukan efektivitasnya.

3.6 Perancangan *Flowchart*



Gambar 3.11 *Flowchart* naïve bayes dan KNN

Berikut adalah penjelasan mengenai *flowchart* pada gambar 3.11 untuk pengembangan model klasifikasi penyakit *stroke* menggunakan *algoritma Naïve Bayes* dan KNN:

1. Mulai: proses awal dimulai yang merupakan awal dari alur kerja pengembangan model untuk klasifikasi penyakit *stroke*.
2. Pengambilan data: pada tahap ini, dilakukan pengambilan data yang akan digunakan untuk pengembangan model. Data yang digunakan diambil dari sumber yang terpercaya seperti *dataset* penyakit *stroke* yang tersedia di situs penyedia data terkenal, www.kaggle.com.
3. *Preprocessing data*: Setelah pengambilan data selesai, tahap selanjutnya adalah melakukan *preprocessing* terhadap data yang diambil. Proses ini melibatkan analisis dan pembersihan data untuk memastikan bahwa data yang digunakan siap untuk pelatihan model. Tahap ini mencakup pembersihan data, seperti menghapus nilai yang inkonsisten atau hilang. Jumlah data yang awalnya banyak akan disaring, sehingga hanya data yang relevan dan bersih yang akan diproses lebih lanjut.
4. Pembagian data: setelah *preprocessing* selesai, data akan dibagi menjadi dua subset: *data training* yang digunakan untuk melatih model dan *data testing* yang digunakan untuk menguji model yang telah dibuat.
5. *Algoritma Naïve Bayes* dan KNN: pada tahap ini, *algoritma Naïve Bayes* dan KNN diterapkan pada *data training* untuk melatih model klasifikasi penyakit *stroke*. *Algoritma Naïve Bayes* akan menghitung probabilitas resiko *stroke* berdasarkan fitur yang ada, sementara KNN akan mengklasifikasikan data berdasarkan kedekatan antara data yang diuji dengan data dalam *dataset*.
6. Evaluasi model: setelah *algoritma* diterapkan, evaluasi model dilakukan menggunakan data testing untuk menilai seberapa baik model dapat memprediksi

penyakit *stroke* pada data yang belum pernah dilihat sebelumnya. Evaluasi dilakukan dengan mengukur akurasi, *precision*, *recall*, dan menggunakan metode *10-fold cross-validation*. Evaluasi model memberikan dua kemungkinan hasil:

(1) Ya: jika hasil evaluasi menunjukkan model memiliki performa yang memadai (misalnya akurasi atau metrik lainnya memuaskan), maka proses dilanjutkan ke tahap berikutnya.

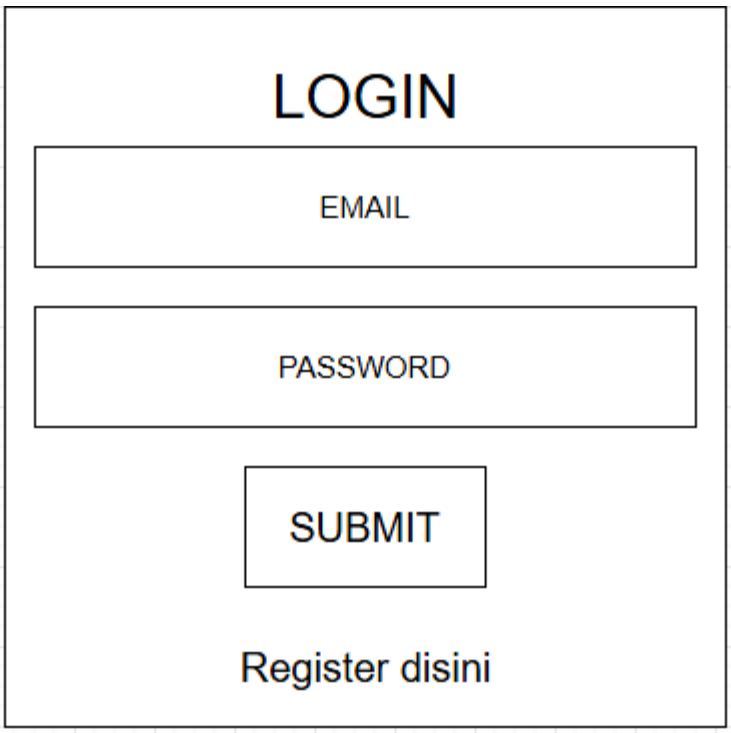
(2) Tidak: jika hasil evaluasi kurang memuaskan (misalnya akurasi terlalu rendah), maka model perlu diperbaiki dengan mengulang proses dari tahap *preprocessing* atau pembagian data.

7. Tampilkan hasil: mengetahui metode terbaik antara *Naïve Bayes* dan KNN untuk mengklasifikasikan penyakit *stroke*, dengan membandingkan akurasi dan performa dari kedua *algoritma*.
8. Metode terbaik: proses pembuatan model selesai, yang berarti model yang telah dibangun siap digunakan untuk melakukan prediksi penyakit *stroke* pada data baru.
9. Selesai: proses pembuatan model selesai yang berarti model yang telah dibangun siap digunakan untuk melakukan prediksi penyakit *stroke* pada data baru.

3.7 Perancangan Layar, Menu, *Database*

Pada tahap ini akan diperlihatkan sketsa atau model desain antarmuka dari *website* yang sedang dalam proses pengembangan.

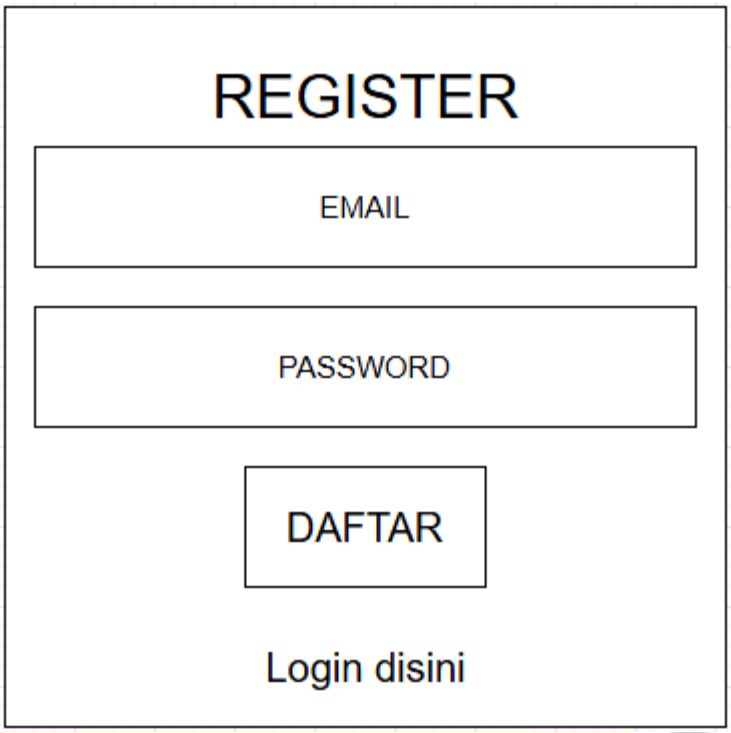
3.7.1 Tampilan Halaman *Login*



Gambar 3.12 Tampilan halaman *login*

Berikut adalah tampilan pertama pada halaman *login* yang berfungsi untuk mengidentifikasi pengguna. Pengguna diwajibkan untuk memasukkan *email* dan *password* pada kolom yang tersedia. Setelah menekan tombol submit, pengguna akan diarahkan ke halaman utama. Bagi yang belum memiliki akun, dapat memilih opsi registrasi untuk menuju halaman pendaftaran. Selain itu, akan muncul notifikasi jika terjadi kesalahan dalam memasukkan *email* atau *password*.

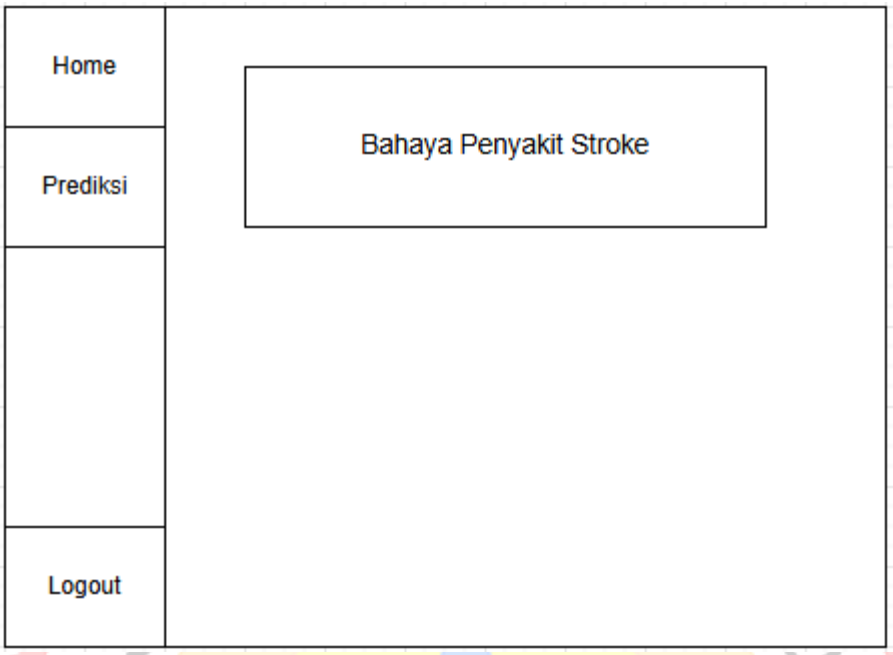
3.7.2 Tampilan Halaman *Register*



Gambar 3.13 Tampilan halaman *register*

Berikut adalah tampilan pada halaman registrasi untuk pendaftaran akun pengguna. Pengguna diwajibkan untuk mengisi kolom *email* dan *password* yang tersedia. Setelah menekan tombol *register*, pengguna akan diarahkan ke halaman *login*. Bagi pengguna yang sudah memiliki akun, dapat memilih opsi *login* untuk masuk ke halaman *login* dan melanjutkan ke halaman utama. Selain itu, akan muncul notifikasi jika *email* yang dimasukkan sudah terdaftar atau sama dengan yang sudah ada.

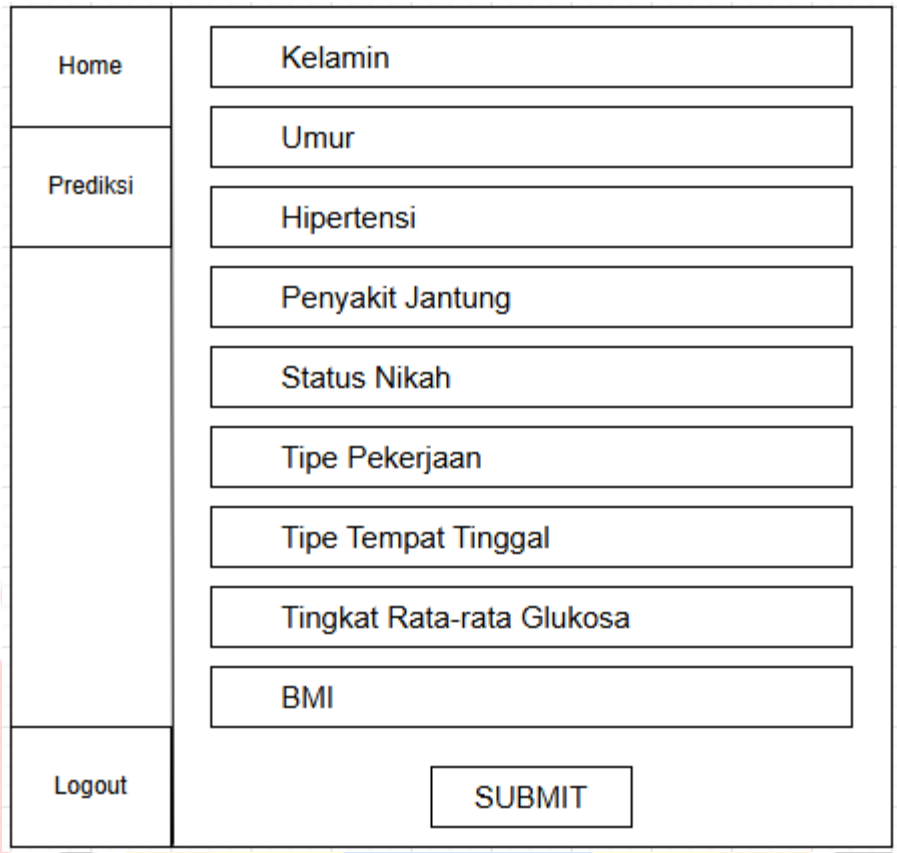
3.7.3 Tampilan halaman utama



Gambar 3.14 Tampilan halaman utama

Setelah pengguna berhasil *login* ke dalam sistem, tampilan ini dirancang untuk memberikan akses yang mudah fitur-fitur utama yang tersedia dalam *website*. Pada bagian kiri terdapat *home* yang berfungsi untuk mengarahkan pengguna ke halaman utama. Prediksi yang digunakan untuk mengarahkan pengguna ke halaman prediksi diabetes, dimana pengguna dapat memasukkan data untuk melakukan klasifikasi apakah mereka berisiko terkena diabetes atau tidak. *Logout* yang memungkinkan pengguna keluar dari sesi saat ini dan pengguna akan diarahkan ke halaman *login*. Pada bagian tengah terdapat informasi mengenai bahaya diabetes pada perempuan seperti bahaya terkena diabetes, ciri ciri terkena *stroke*, solusi mencegah *stroke* serta dampak dari diabetes jika tidak dikelola dengan baik. Pada bagian kanan akan terdapat gambar yang menampilkan diabetes.

3.7.4 Tampilan Halaman *Input Data*



Gambar 3.15 Tampilan halaman *input data*

Berikut adalah tampilan halaman untuk memasukkan data. Pengguna dapat mengisi informasi terkait faktor-faktor yang mempengaruhi resiko *stroke*. Setelah data dimasukkan, pengguna dapat menekan tombol prediksi di bawahnya untuk memproses data. Sistem akan menerapkan *algoritma Naive Bayes* atau KNN berdasarkan input yang diberikan dan menghasilkan prediksi mengenai apakah pengguna beresiko terkena *stroke*. Selain itu, hasil prediksi akan disertai dengan saran, apakah pengguna terindikasi positif *stroke* atau tidak. Hasil tersebut juga akan mencakup tingkat akurasi dari *algoritma* yang digunakan melalui metode *10-fold cross-validation*.

3.7.5 Perancangan *Database*

Pada bab ini akan dijelaskan mengenai *database* yang dirancang dan digunakan untuk mendukung *website* dalam menyimpan data-datanya.

a. Tabel user

Tabel 3.9 Keterangan user

Nama *Field*	*Type field*
ID	*Int*(11)
Email	*Varchar*(50)
Password	*Varchar*(50)

Keterangan:

ID : menampilkan jumlah data, di mana ID berfungsi sebagai *primary key* dan memiliki sifat *auto increment*. Tipe data *int*(11) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 11 digit.

Email : menampilkan *email* pengguna yang telah terdaftar. Tipe data *Varchar*(50) menunjukkan bahwa kolom ini menyimpan teks dengan panjang maksimum 50 karakter.

Password : menampilkan kata sandi pengguna yang telah terdaftar. Tipe data *Varchar*(50) menunjukkan bahwa kolom ini menyimpan teks dengan panjang maksimum 50 karakter, di mana kata sandi juga di-hash menggunakan *algoritma* MD5.

b. Tabel *dataset*

Tabel 3.10 Keterangan *dataset*

Nama *Field*	*Type field*
ID	*Int*(11)
Kelamin	*Int*(3)
Umur	*Int*(3)
Hipertensi	*Int*(3)
Penyakit Jantung	*Int*(3)
Status Nikah	*Int*(3)
Tipe Pekerjaan	*Int*(3)
Tipe Tempat Tinggal	*Int*(3)
Tingkat Rata-rata Glukosa	*Float*
BMI	*Float*
Hasil	*Int*(1)

Keterangan :

ID : menampilkan jumlah data, di mana ID berfungsi sebagai *primary key* dan bersifat auto increment. Tipe data *int*(11) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 11 digit.

Kelamin : menampilkan jenis kelamin pengguna, di mana 1 untuk laki-laki dan 2 untuk perempuan. Tipe data *int*(3) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 3 digit.

Umur : menampilkan usia pengguna berdasarkan *dataset* yang digunakan. Tipe data *int*(3) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 3 digit.

Hipertensi : menampilkan status hipertensi pengguna, di mana 1 berarti memiliki hipertensi dan 0 berarti tidak. Tipe data *int*(3) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 3 digit.

Penyakit Jantung : menampilkan status penyakit jantung pengguna, di mana 1 berarti memiliki penyakit jantung dan 0 berarti tidak. Tipe data *int*(3) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 3 digit.

Status Nikah : menampilkan status pernikahan pengguna, di mana 1 berarti menikah dan 0 berarti belum menikah. Tipe data *int*(3) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 3 digit.

Tipe Pekerjaan : menampilkan jenis pekerjaan pengguna, di mana 1 berarti *self-employed* dan 0 berarti *private*. Tipe data *int*(3) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 3 digit.

Tipe Tempat Tinggal : Menampilkan jenis tempat tinggal pengguna, di mana 1 berarti *urban* dan 0 berarti *rural*. Tipe data *int*(3) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 3 digit.

Tingkat Rata-rata Glukosa : menampilkan tingkat rata-rata glukosa darah pengguna. Tipe data *float* menunjukkan bahwa kolom ini digunakan untuk menyimpan angka desimal.

BMI : menampilkan nilai *Body Mass Index* (BMI) pengguna. Tipe data *float* menunjukkan bahwa kolom ini digunakan untuk menyimpan angka desimal.

Hasil : menampilkan hasil prediksi apakah individu beresiko terkena penyakit tertentu, seperti *stroke* atau diabetes, dengan 0 berarti negatif dan 1 berarti positif. Tipe data *int*(1) menunjukkan bahwa kolom ini menyimpan angka *int*eger dengan panjang maksimal 1 digit.

3.8 Jadwal Penelitian

Tabel 3.11 Tabel jadwal penelitian

No	Kegiatan	2025											
		April				Mei				Juni			
		1	2	3	4	1	2	3	4	1	2	3	4
1	Pengumpulan Data												
2	Analisis Data												
3	Design Aplikasi												
4	Coding Program												
5	Testing												
6	Penulisan Paper												