BAB I
PENDAHULUAN

1.1 Latar Belakang

Diabetes mellitus merupakan salah satu penyakit kronis yang ditandai dengan tingginya kadar gula dalam darah. Glukosa merupakan sumber energi utama bagi tubuh. Namun, pada penyandang diabetes, pankreas tidak mampu memproduksi insulin dengan efektif. Akibatnya, sel-sel tubuh tidak dapat menyerap dan mengolah glukosa menjadi energi sehingga dapat menimbulkan berbagai gangguan kesehatan dan jika tidak ditangani dengan baik, diabetes dapat menimbulkan berbagai komplikasi kesehatan seperti gagal ginjal, kerusakan saraf, penyakit jantung, dan sebagainya (Alodokter, 2024).

Menurut data *International Diabetes Federation* (IDF), jumlah penyandang diabetes di dunia pada tahun 2021 mencapai 537 juta orang dan diperkirakan akan meningkat menjadi 643 juta pada tahun 2030 serta 783 juta pada tahun 2045. Indonesia sendiri menempati peringkat kelima negara dengan jumlah penyandang diabetes terbanyak, yakni 19.5 juta kasus pada tahun 2021 dan diprediksi meningkat hingga 28.6 juta kasus pada tahun 2045. Data dari Kementerian Kesehatan RI juga menunjukkan peningkatan prevalensi diabetes pada penduduk usia di atas 15 tahun, dari 10.9% pada tahun 2018 menjadi 11.7% pada tahun 2023. Peningkatan ini menunjukkan bahwa diabetes telah menjadi masalah kesehatan global yang membutuhkan strategi pencegahan dan penanganan yang lebih efektif (Katadata, 2024).

Secara global, jumlah penderita diabetes meningkat pesat dari 200 juta pada tahun 1990 menjadi lebih dari 800 juta pada tahun 2022. Prevalensi penyakit ini mengalami lonjakan lebih cepat di negara-negara berpenghasilan rendah dan menengah. Pada tahun 2022, sekitar 14% orang dewasa berusia 18 tahun ke atas tercatat menderita diabetes, meningkat signifikan dari 7% pada tahun 1990. Selain itu, lebih dari separuh (59%) penderita diabetes berusia 30 tahun ke atas tidak mengonsumsi obat untuk mengelola penyakit mereka. Kondisi ini menunjukkan perlunya pendekatan yang lebih komprehensif dalam deteksi dini, pencegahan, dan pengelolaan diabetes guna mengurangi risiko komplikasi serta meningkatkan kualitas hidup penderita (The Guardian, 2024).

Banyak penyandang diabetes tidak menyadari bahwa dirinya sedang mengidap diabetes karena gejala atau ciri-ciri diabetes yang seringkali terlalu umum, ringan dan berkembang secara perlahan sehingga sulit untuk dikenali. *International Diabetes Federation* (IDF) melaporkan bahwa dari 589 juta orang dewasa yang mengidap diabetes di seluruh dunia, sekitar 252 juta di antaranya tidak menyadari bahwa mereka memiliki kondisi tersebut. Banyak orang hanya menyadari dan melakukan pemeriksaan setelah komplikasi sudah mulai muncul (IDF, 2025). Oleh karena itu, diperlukan skrining dini berbasis *data mining* untuk membantu deteksi awal diabetes dan mencegah risiko komplikasi yang lebih berat.

Berbagai penelitian telah menerapkan metode *machine learning* untuk mendeteksi diabetes secara dini. Menurut penelitian oleh Setyawan et al., (2025), Logistic Regression memiliki keunggulan dalam hal interpretabilitas dan efisiensi komputasi, tetapi kurang mampu menangani hubungan *non-linear*. Sementara itu, Hanif et al., (2025) menunjukkan bahwa Random Forest unggul dalam menangani data kompleks dan interaksi antar fitur, namun hasilnya sulit diinterpretasikan. Di sisi lain, Saputra (2023) menemukan bahwa Decision Tree unggul dalam hal kecepatan komputasi dan interpretasi yang mudah namun cenderung rentan terhadap *overfitting*.

Untuk mengatasi keterbatasan model tunggal tersebut pendekatan *Voting Classifier* menjadi alternatif yang efektif. Metode ini menggabungkan model *machine learning* baik yang sama maupun yang berbeda konsep untuk melakukan klasifikasi (Manconi et al., 2022). Keunggulan utama dari pendekatan ini adalah kemampuannya dalam mengurangi bias dan varians yang sering menjadi masalah pada model individu sehingga diharapkan dapat memberikan hasil yang lebih baik dibandingkan dengan model tunggal. *Hard voting* melakukan prediksi berdasarkan kelas yang paling banyak muncul dari prediksi kelas yang dihasilkan oleh model tunggal (Oliveira et al., 2022). Sedangkan *soft voting* merupakan kombinasi dari beberapa algoritma pengklasifikasi dimana keputusan dibuat berdasarkan keputusan individu yang digabungkan dengan nilai probabilitas untuk menentukan bahwa data termasuk dalam kelas tertentu. Dalam metode *soft voting ensemble*, prediksi dibobot berdasarkan kepentingan pengklasifikasi dan menggabungkannya untuk mendapatkan jumlah probabilitas terbobot. Label target dengan jumlah probabilitas terbesar dipilih karena memiliki nilai *voting* terbesar (Sherazi et al., 2021).

Beberapa penelitian menunjukkan bahwa metode *ensemble* seperti *Hard Voting* dan *Soft Voting* memberikan hasil yang lebih akurat dibandingkan dengan model tunggal. Dalam penelitian oleh Saputra (2023), model *Soft Voting* menunjukkan performa yang lebih unggul dibandingkan dengan tiga model tunggal yaitu Decision Tree, Support Vector Machine dan K-Nearest Neighbors. Kemudian sebuah penelitian dari Bangladesh (Sunny et al., 2024) menunjukkan bahwa klasifikasi *Soft Voting* mencapai akurasi yang mengesankan yaitu sebesar 99,50% pada kumpulan data diabetes Irak dengan menggunakan algoritma Random Forest, XGBoost, dan Gradient Boosting. Selain itu, penelitian oleh AMALIANA et al., (2024) menunjukkan bahwa *Soft Voting* unggul dalam hal presisi, sementara *Hard Voting* lebih baik dalam akurasi dan spesifisitas. Penerapan seleksi fitur khususnya *Recursive Feature Elimination with Cross-Validation* (RFECV) juga terbukti meningkatkan performa model. Hal ini dibuktikan dengan penelitian oleh Nurhalisa (2024) yang menemukan bahwa RFECV dapat meningkatkan akurasi SVM dan mempertahankan akurasi XGBoost dalam mengklasifikasikan kelayakan air minum. Selain itu, penelitian ini mengatakan bahwa RFECV lebih unggul dibandingkan dengan *feature selection* menggunakan SelectKBest dengan mempertahankan 5 fitur.

Berdasarkan hasil penelitian terdahulu, dapat disimpulkan bahwa kombinasi antara metode *ensemble learning* (*Hard Voting* dan *Soft Voting*) dengan teknik seleksi fitur RFECV merupakan pendekatan yang berpotensi dapat meningkatkan performa klasifikasi diabetes. Oleh karena itu, pada penelitian ini diterapkan metode *ensemble*, yaitu *Hard Voting* dan *Soft Voting* yang dikombinasikan dengan *Recursive Feature Elimination with Cross-Validation* (RFECV) sebagai teknik seleksi fitur, dengan tujuan menghasilkan model klasifikasi yang lebih optimal untuk skrining awal diabetes serta mendukung upaya deteksi dan penanganan penyakit diabetes sejak dini.

1.2 Rumusan Masalah

Berdasarkan latar belakang, penelitian ini memiliki beberapa rumusan masalah, yaitu sebagai berikut:

1) Bagaimana perbandingan efektifitas *Hard Voting* dan *Soft Voting* dalam melakukan klasifikasi penyakit Diabetes Mellitus?
2) Sejauh mana *Feature Selection* bisa meningkatkan akurasi model klasifikasi diabetes pada metode *Soft Voting* dan *Hard Voting*?

1.3 Tujuan Penelitian

Berikut tujuan penelitian ini:

1) Untuk membandingkan efektivitas *Hard Voting* dan *Soft Voting* dalam melakukan klasifikasi dini Diabetes Mellitus.
2) Untuk menganalisis pengaruh *Feature Selection* dalam meningkatkan akurasi model klasifikasi Diabetes.

1.4 Manfaat Penelitian

Adapun manfaat dari penelitian ini sebagai berikut:

1) Memberikan kontribusi pada bidang kesehatan dalam meningkatkan akurasi klasifikasi diabetes mellitus dengan menggunakan metode *ensemble learning*.
2) Menentukan metode yang lebih optimal dalam menganalisis data kesehatan khususnya pada klasifikasi penyakit diabetes.
3) Berpotensi dalam pengembangan sistem keputusan khususnya bagi Puskesmas Balangnipa Sinjai untuk memberikan rekomendasi kepada pasien yang berisiko diabetes agar segera melakukan pemeriksaan sebelum kondisinya semakin parah.

1.5 Ruang Lingkup

Untuk memperjelas ruang lingkup penelitian ini, terdapat beberapa batasan yang akan diterapkan, antara lain:

1) Penelitian ini difokuskan pada perbandingan metode *Soft Voting* dan *Hard Voting* dalam klasifikasi penyakit Diabetes Mellitus dengan menggunakan tiga model dasar yaitu Logistic Regression, Random Forest, dan Decision Tree.

2) Data yang digunakan berasal dari dua sumber, yaitu data rekam medis pasien dari Puskesmas Balangnipa, Kabupaten Sinjai, serta dataset publik yang bersumber dari Kaggle yang dikombinasikan untuk memperkuat variasi dan jumlah sampel penelitian.
3) Objek penelitian terbatas pada klasifikasi penyakit diabetes mellitus dengan dua kelas utama, yaitu diabetes dan non-diabetes.
4) Proses seleksi fitur menggunakan metode *Recursive Feature Elimination with Cross-Validation* (RFECV).
5) Optimasi bobot pada model *Soft Voting* dilakukan menggunakan *Particle Swarm Optimization* (PSO)
6) Penelitian ini hanya difokuskan pada evaluasi performa model menggunakan metrik *accuracy, precision, recall*, dan *F1-score*.

1.6 Landasan Teori

1.6.1 Diabetes Mellitus

Diabetes adalah penyakit kronis yang disebabkan oleh pankreas yang tidak memproduksi insulin dalam jumlah yang cukup (bahkan tidak sama sekali) atau tubuh tidak dapat menggunakan insulin yang diproduksinya secara efektif. Diabetes yang tidak terkontrol dapat menyebabkan hiperglikemia, atau gula darah tinggi, yang seiring waktu dapat menyebabkan kerusakan serius pada banyak sistem tubuh, terutama saraf dan pembuluh darah. Jumlah penderita diabetes sangat meningkat berdasarkan analisis global yang diterbitkan dalam Lancet yang menemukan bahwa tingkat diabetes pada orang dewasa meningkat dua kali lipat dari sekitar 7% menjadi sekitar 14% antara tahun 1990 hingga 2022, dengan peningkatan terbesar di negara-negara berpenghasilan rendah dan menengah (The Guardian, 2024).

Diabetes mellitus terdiri atas beberapa jenis yaitu diabetes tipe 1, diabetes tipe 2, diabetes gestasional, serta tipe diabetes lainnya/sekunder. Diabetes tipe 1 disebabkan oleh destruksi autoimun sel beta pankreas yang mengakibatkan kekurangan insulin absolut. Biasanya muncul pada anak-anak atau remaja dan pasien memerlukan terapi insulin seumur hidup. Kadang-kadang dikenal sebagai diabetes yang bergantung insulin atau diabetes juvenil. Destruksi beta-sel ini dapat berdampak cepat pada anak-anak atau lebih lambat pada orang dewasa. Pasien sering memiliki autoantibodi spesifik sebagai penanda imunitas. Kemudian diabetes tipe 2 yang merupakan tipe diabetes yang paling umum yang disebabkan oleh resistensi insulin, yaitu ketika sel tidak berhasil menggunakan insulin dengan baik. Pengobatan dapat dilakukan dengan merubah gaya hidup seperti diet sehat, berolahraga dan jika diperlukan mengonsumsi obat-obat oral atau terapi insulin. Diabetes tipe 2 mungkin saja tidak disadari karena gejalanya yang sangat umum. Pengukuran glukosa darah naik dapat membantu diagnosis. Sedangkan Diabetes Gestasional terjadi selama kehamilan akibat resistensi insulin dipicu hormon kehamilan. Biasanya didiagnosis pada trimester kedua atau ketiga kehamilan. Setelah melahirkan, kadar gula biasanya kembali normal namun wanita dengan

diabetes gestasional memiliki risiko lebih tinggi mengalami diabetes tipe 2 di kemudian hari. Salah satu fenomena yang memicu peningkatan kasus diabetes gestasional adalah adanya berbagai faktor risiko seperti obesitas, riwayat keluarga dengan diabetes, usia ibu yang lebih tua, serta gaya hidup (Afifah, 2025). Kemudian ada tipe diabetes lainnya/sekunder yaitu diabetes yang terjadi akibat penyakit lain atau akibat penggunaan obat-obatan tertentu yang mempengaruhi sekresi atau kerja insulin, seperti gangguan pankreas, penggunaan kortikosteroid, dan gangguan hormon lainnya. Diabetes dapat berdampak buruk apabila tidak segera diobati, seperti kerusakan mata atau bahkan dapat mengalami kebutaan, kerusakan saraf, stroke, serangan jantung dan masalah sirkulasi yang dapat menyebabkan luka yang sulit sembuh bahkan harus diamputasi hingga dapat menyebabkan kematian. Menurut penelitian Siallagan dan Fitriyani (2021), ada beberapa gejala diabetes mellitus yang perlu diwaspadai, antara lain:

a. *Polydipsia* (cepat haus)
b. *Polyuria* (banyak buang air kecil)
c. *Polyphagia* (cepat lapar)
d. *Sudden Weight Loss* (penurunan berat badan)
e. *Weakness* (rasa lelah dan lemah pada tubuh)
f. *Visual Blurring* (pandangan kabur/buram)
g. *Delayed Healing* (pemulihan luka yang lama atau sering infeksi)
h. *Acanthosis Nigricans* (warna kulit gelap)

Selain gejala yang dialami, ada beberapa pula faktor resiko yang dialami oleh penderita Diabetes Mellitus, antara lain:

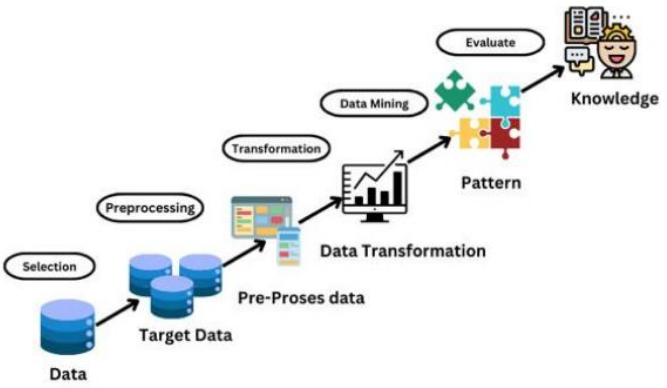
i. Obesitas (kegemukan)
j. Hipertensi
k. Riwayat Keluarga Diabetes Mellitus
l. Dislipedimia
m. Umur
n. Riwayat Persalinan
o. Faktor Genetik
p. Alkohol dan Rokok

1.6.2 *Data Mining*

Data mining merupakan bagian dari proses *Knowledge Discovery in Database* (KDD) merupakan aktivitas yang berkaitan dengan pengumpulan data, pemakaian data historis untuk menemukan pengetahuan, informasi, keteraturan, pola atau hubungan dalam data yang berukuran besar. *Output* dalam *data mining* dapat dipergunakan sebagai alternatif dalam pengambilan keputusan atau memperbaiki pengambilan keputusan di masa yang akan datang (Buulolo, 2020).

Kemampuan *data mining* dalam menggali informasi dalam cakupan yang sangat besar ini menjadi kelebihan yang tidak perlu diragukan. Teknologi seperti ini biasanya digunakan untuk memprediksi berbagai hal dalam kehidupan, dimana

data mining mengotomatisasi proses pencarian informasi di dalam basis data yang besar dan menemukan pola-pola yang tidak diketahui sebelumnya (Kalimah, 2022). Data dapat bersumber seperti dari *databases*, data *warehouses*, situs web, repositori informasi lainnya, atau data yang dialirkan ke dalam sistem secara dinamis. Proses dalam menemukan pengetahuan tersebut dicakup dalam beberapa tahap, seperti yang ditunjukkan pada gambar berikut.



Gambar 1 Tahapan *Data Mining (Binus, 2021)*

a. *Data Original*
   Merupakan data asli yang diperoleh dari pengambilan data, tanpa dilakukan proses seleksi data dan pembersihan serta transformasi data (Buani, 2024).
b. *Data Selection*
   Data selection merupakan tahapan melakukan identifikasi data, pemilihan data yang relevan agar dapat dijadikan data untuk melakukan penelitian, sehingga model yang terbentuk lebih optimal (Buani, 2024).
c. *Data Cleaning*
   Data cleaning merupakan proses membuang data yang duplikasi, memeriksa data yang tidak konsisten, dan memperbaiki kesalahan pada data, seperti kesalahan penulisan, dan memberikan nilai atau menghapus data yang kosong (Buani, 2024).
d. *Data Transformation*
   Proses transformasi data menjadi data tertentu yang digunakan atau dapat dibaca oleh algoritma. Contohnya adalah merubah data *Type Object* menjadi *Numbering* (Buani, 2024).
e. *Data Mining*
   Data mining adalah proses penentuan pola atau penentuan model yang terbaik sehingga hasil dari proses KDD akurat serta dapat dipercaya (Buani, 2024).
f. Evaluasi
   Tahapan evaluasi bertujuan untuk melakukan evaluasi terhadap model atau pola yang telah terbentuk dari proses sebelumnya yaitu proses *data mining*

sehingga hasil dari pola/model tersebut memiliki hasil akurasi yang dapat dipercaya (Buani, 2024).

Data mining dibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan, seperti pada gambar berikut.

Gambar 2 Pengelompokan *Data Mining* (Handoko et al., 2018)

- Deskripsi, suatu cara yang digunakan untuk menggambarkan suatu pola dan kecenderungan dalam data.
- Estimasi, hampir sama dengan klasifikasi namun variabel target estimasi lebih ke arah numerik daripada ke arah kategori.
- Prediksi, hampir sama dengan klasifikasi dan estimasi, prediksi memberikan gambaran nilai dari hasil yang akan ada dimasa mendatang.
- Klasifikasi merupakan pengelompokan data menjadi beberapa bagian atau kelas dengan target variabel kategori. Contohnya seperti penggolongan diabetes yaitu positif atau negatif, kemudian penggolongan berat badan yaitu gemuk, sedang, atau kurus dll.
- Klastering, merupakan pengelompokan hasil dari pengamatan yang memiliki kemiripan objek.
- Asosiasi, metode yang digunakan untuk menemukan pola atau hubungan antar item dalam sekumpulan data.

1.6.3 Klasifikasi

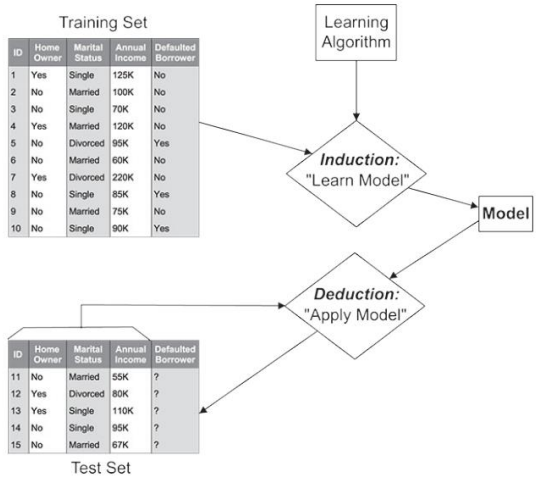
Klasifikasi adalah proses pengkategorian yang dilakukan dalam sekumpulan data kemudian membaginya ke dalam kelas-kelas tertentu. Klasifikasi memberikan penilaian objek data untuk memasukkannya ke dalam kelas tertentu dari jumlah kelas yang tersedia. Klasifikasi dapat didefinisikan sebagai pekerjaan yang melakukan pelatihan/pembelajaran terhadap fungsi target yang memetakan setiap set atribut ke satu jumlah label kelas yang tersedia (Finda, 2024). Berikut ilustrasi proses klasifikasi (Washilaturrizqi et al., 2023).

Gambar 3 Proses Klasifikasi

Sebuah klasifikasi juga disebut sekumpulan *record* dan bisa juga disebut kasus atau sampel. Sebuah tuple (x,y) adalah pemetaan *record*, dimana x sebagai himpunan atribut dan y sebagai label kelas. Klasifikasi dibagi menjadi dua tahap antara lain:

a. Tahap pelatihan (*training*), bagian dari data yang kategori datanya diketahui dan dimasukkan untuk membangun dan melatih model.
b. Tahap pengujian (*testing*), bagian dari data yang terbentuk diuji dengan bagian data lainnya yang tidak diberikan pada model untuk dipelajari untuk mengetahui akurasi model.

Model klasifikasi dibangun dari sebuah dataset input dengan sebuah algoritma pembelajaran yang harus sesuai dengan data input dan memprediksi dengan benar label kelas dari *record* yang belum pernah terlihat sebelumnya. Berikut ilustrasi pendekatan umum untuk pembentukan model klasifikasi.



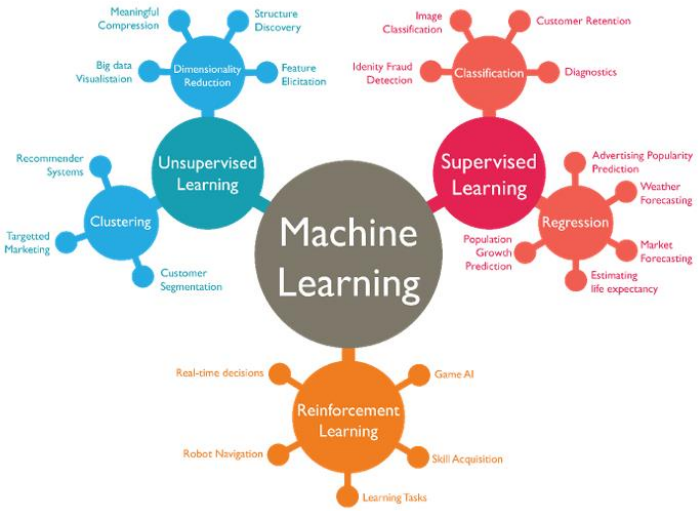
Gambar 4 Pembangunan model klasifikasi (Tan et al., 2019)

1.6.4 *Machine Learning*

Machine Learning atau pembelajaran mesin merupakan pendekatan dalam kecerdasan buatan yang merupakan disiplin ilmu yang mencakup perancangan dan pengembangan algoritma yang memungkinkan komputer untuk mengembangkan perilaku yang didasarkan pada data empiris, seperti data dari basis data, hal ini merupakan teknik yang digunakan untuk mengembangkan mesin otomatis berdasarkan eksekusi pada algoritma dan kumpulan aturan yang terdefinisikan.

Machine learning dilengkapi sejumlah aturan program yang dijalankan oleh algoritma, oleh karena itu teknik *machine learning* dapat dikategorikan sebagai instruksi yang dijalankan dan dipelajari secara otomatis untuk menghasilkan output yang optimal. Hal ini dilakukan secara otomatis tanpa ada campur tangan manusia. Semua dilakukan secara otomatis untuk mengubah data menjadi beberapa pola dan diinputkan ke dalam sistem untuk mendeteksi masalah secara otomatis (Rahmadini et al., 2023).

Machine learning banyak digunakan untuk menggantikan atau menirukan perilaku manusia untuk menyelesaikan masalah dengan mempelajari pola dari data yang masuk ke dalam mesin. Proses dalam *machine learning* melibatkan algoritma yang dirancang untuk mengenali pola dalam data dan membuat prediksi atau keputusan berdasarkan pola. Keberhasilan model dalam menghasilkan rekomendasi atau prediksi bergantung pada kualitas, jumlah dan relevansi data terhadap masalah yang ingin diselesaikan (Alpaydin, 2020). *Machine learning* terbagi atas 3 jenis utama yaitu *Supervised Learning*, *Unsupervised Learning* dan *Reinforcement Learning*.



Gambar 5 Jenis *machine learning* (Yakip, 2020)

a. *Supervised Learning*

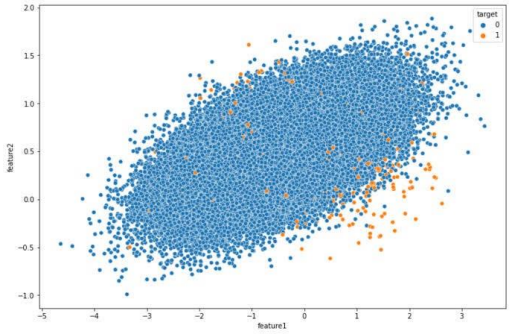
Supervised learning adalah tipe *machine learning* di mana kita mempunyai variabel *input* dan variabel *output* dan menggunakan satu algoritma atau lebih untuk mempelajari fungsi pemetaan dari *input* ke *output*. Tujuan dari metode ini adalah agar algoritma dapat belajar dengan membandingkan *output* aktualnya dengan *output* yang diajarkan untuk menemukan kesalahan, dan memodifikasi model yang sesuai. Oleh karena itu, *supervised learning* menggunakan pola untuk memprediksi nilai label pada data yang tidak berlabel tambahan.

b. *Unsupervised Learning*
Unsupervised learning mempelajari bagaimana sistem mewakili pola tertentu dengan cara mencerminkan struktur statistik dari seluruh kumpulan pola yang diinput. Pendekatan *unsupervised learning* tidak menggunakan *data training* untuk melakukan prediksi maupun klasifikasi. Berdasarkan model matematisnya, algoritma ini tidak memiliki variabel target. Salah satu tujuan dari algoritma ini adalah mengelompokkan objek yang hampir sama dalam suatu area tertentu. Hal ini tentu berbeda dengan *supervised learning* yang mana dari data *input* pasti ada *output* berupa "ya" atau "tidak". Pada *unsupervised learning* lebih menjelaskan data atau menyimpulkan data tanpa adanya *output*.

c. *Reinformancement Learning*
Reinformancement learning mengambil tindakan yang sesuai untuk memaksimalkan keputusan yang diambil dalam kondisi tertentu. *Reinformancement learning* berbeda dengan *supervised learning* yang data pelatihannya memiliki kunci jawaban sehingga model dilatih dengan jawaban yang benar itu sendiri. Sedangkan *reinforcement learning* tidak ada jawaban tetapi mesin dalam hal ini memutuskan apa yang harus dilakukan untuk menyelesaikan tugas yang diberikan.

1.6.5 *Imbalance Data*

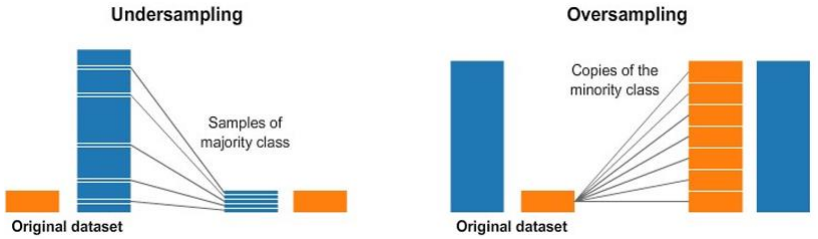
Ketidakseimbangan kelas (*class imbalance*) merupakan permasalahan umum yang sering ditemukan pada dataset dunia nyata. Kondisi ini terjadi ketika jumlah sampel pada satu kelas jauh lebih banyak dibandingkan dengan kelas lainnya. Model klasifikasi yang dilatih menggunakan data tidak seimbang cenderung bias terhadap kelas mayoritas sehingga kelas minoritas sering salah diklasifikasikan. Misalnya, pada data kesehatan, umumnya terdapat lebih banyak data pasien dengan hasil negatif dibandingkan dengan hasil positif (Abdullah-All-Tanvir et al., 2023). Berikut contoh ketidakseimbangan data antar 2 kelas.



Gambar 6 Contoh *Imbalance data* antar dua kelas (Topre, 2023)

Penggunaan dataset yang tidak seimbang dapat membuat performa model *machine learning* menjadi tidak maksimal dan tidak akurat (Khuat, 2020). Untuk mengatasi permasalahan ketidakseimbangan kelas pada data set, dapat digunakan

metode sampling yang terbagi menjadi dua teknik yaitu teknik *oversampling* dan teknik *undersampling*. Teknik *Oversampling* bekerja dengan cara menambahkan data pada kelas minoritas sehingga jumlah datanya menjadi sama. Sedangkan *undersampling* bekerja dengan cara menghapus data pada kelas mayoritas sehingga jumlah datanya menjadi sama. Kedua teknik ini memiliki kekurangan dan kelebihan masing-masing. Adapun gambaran teknik *undersampling* dan *oversampling* sebagai berikut.



Gambar 7 *Undersampling* dan *Oversampling* (Badr, 2019)

Salah satu teknik *undersampling* yang sering digunakan adalah *Edited Nearest Neighbour* (ENN). Metode ini berfokus pada identifikasi instans yang redundan atau *noisy* dalam kelas mayoritas. ENN bekerja dengan menghapus data yang tidak konsisten terhadap label mayoritas di lingkungannya. Prinsip kerja ENN didasarkan pada algoritma K-Nearest Neighbors (KNN), dengan tahapan umum sebagai berikut:

1. Menentukan K-Nearest Neighbors (KNN)
   Setiap instans dalam dataset dihitung jaraknya terhadap instans lain menggunakan metrik jarak tertentu. Nilai *k* tetangga terdekat kemudian ditentukan untuk setiap instans.
2. Membandingkan label kelas
   Label kelas setiap instans dibandingkan dengan label *k* tetangganya. Jika mayoritas tetangga memiliki label berbeda, maka instans tersebut dianggap tidak konsisten dengan lingkungannya.
3. Menghapus instans mayoritas yang tidak konsisten
   Instans dari kelas mayoritas yang berbeda label dengan mayoritas tetangga terdekatnya akan dihapus dari dataset. Proses ini membantu mengurangi pengaruh data *noisy* dan memperjelas batas keputusan antar kelas.

1.6.6 Random Forest

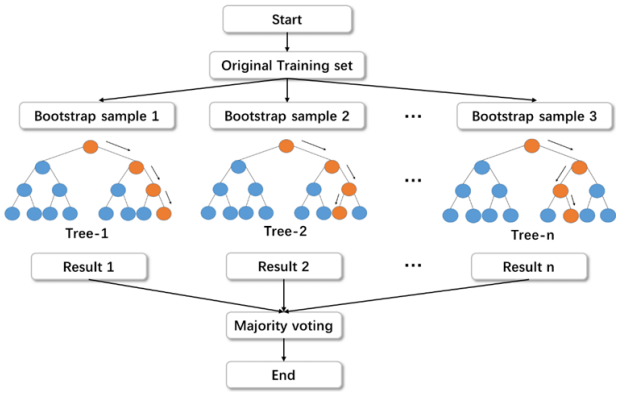
Random forest adalah salah satu metode di dalam *machine learning* yang digunakan untuk proses klasifikasi terutama untuk data dalam jumlah yang besar. Random forest adalah pengembangan dari metode *Classification and Regression Tree* (CART) dengan menerapkan agregasi *bootstrap* dan metode pemilihan fitur secara acak (Nugroho et al., 2019).

Random forest diawali dengan teknik dasar data mining yaitu decision tree, dimana kumpulan decision tree tersebut digunakan untuk mengklasifikasi data ke

suatu kelas. Metode ini menciptakan berbagai pohon (*tree*) dalam sebuah hutan (*forest*) sehingga semakin meningkatkan keandalan model. Random forest dapat mengurangi risiko *overfitting* yang sering terjadi pada decision tree individual dengan melakukan *averaging* prediksi dari beberapa *tree* yang dibagun pada subset data yang berbeda (Khoeruddin et al., 2023).

Random forest membangun sejumlah pohon keputusan secara paralel pada tahap pelatihan, kemudian menggabungkan hasil prediksi setiap pohon melalui mekanisme pemungutan suara mayoritas (*majority voting*) untuk melakukan klasifikasi (Hanif et al., 2025). Proses pembentukan pohon dilakukan dengan metode *random sampling* menggunakan teknik *bootstrap* yang menghasilkan beberapa subset data latih dari dataset asli dengan pengambilan sampel ulang secara acak. Karena pengambilan data dilakukan secara acak dan data yang telah dipilih dikembalikan lagi ke kumpulan awal, maka suatu data dapat muncul lebih dari sekali dalam subset pelatihan. Setiap subset data latih tersebut digunakan untuk membangun model decision tree secara terpisah melalui teknik *bagging*. Berbeda dengan decision tree tunggal, random forest memilih subset fitur secara acak pada setiap *node*, sehingga mampu mengurangi korelasi antar pohon dan meningkatkan kemampuan generalisasi model.

Prediksi akhir diperoleh dengan menggabungkan hasil prediksi dari semua pohon melalui mekanisme pemungutan suara (*majority voting*). Proses yang dimulai dari pengambilan sampel hingga tahap *voting* dapat dilihat pada gambar berikut.



Gambar 8 Proses Random Forest (Guo et al., 2021)

Proses konstruksi random forest adalah sebagai berikut:

1. Memilih *n* subset data pelatihan (*sub-training set*) secara random beserta set data uji yang sesuai dari dataset asli.
2. Pada setiap node pohon keputusan yang akan dipecah, secara acak pilih *m* atribut. Dari *m* atribut tersebut, pilih satu atribut berdasarkan metode *information gain* sebagai atribut pemisah.
3. Ulangi langkah 2 hingga pohon tidak dapat dipecah lagi.

4. Ulangi langkah 1–3 untuk membangun sejumlah besar pohon keputusan sehingga terbentuk sebuah random forest.

Gambar tersebut mengilustrasikan bagaimana random forest bisa menghasilkan hasil prediksi dengan menggunakan *majority voting*. Metode random forest memiliki 2 tahapan. Tahap pertama adalah pembentukan *forest* dan tahap kedua adalah *voting* hasil klasifikasi (Willy et al., 2021).

$$forest = \{h(x, \theta_k), k = 1,2, \dots, n\} \quad (1)$$

Dimana:

h = Hipotesa atau klasifikasi

x = *Input vector*

θ_k = *Random parameters*

Persamaan ini menggambarkan bahwa random forest tersusun atas sekumpulan decision tree, dimana parameter θ_k diperoleh melalui proses *bootstrap sampling* sehingga setiap pohon memiliki karakteristik yang berbeda. Selanjutnya $\{h(x, \theta_k), k = 1,2, \dots, n\}$ menunjukkan bahwa forest terdiri dari n buah pohon keputusan. Setiap pohon dilatih menggunakan sampel data dan subset fitur yang berbeda. Langkah selanjutnya adalah melakukan *voting* dengan menggunakan persamaan berikut.

$$C_{rf}(x) = \underset{Y}{argmax} \sum_{k=1}^{n} I\ (h(x, \theta_k) = Y) \quad (2)$$

Dimana:

$C_{rf}(x)$ = *Class* hasil klasifikasi Random Forest

$\underset{Y}{argmax}$ = Fungsi yang memilih kelas *Y* yang memberikan suara terbanyak

n = Jumlah pohon dalam forest

$I(\dots)$ = Fungsi indikator bernilai 1 jika kondisi dalamnya benar dan 0 jika salah

C_n = *Class* prediksi dari *tree* ke-*n* pada Random Forest

$h(x, \theta_k)$ = Prediksi kelas dari pohon ke- k untuk input x

Y = Target kelas

Persamaan tersebut menjelaskan proses penentuan label kelas menggunakan mekanisme *majority voting*. Setiap pohon ke- k memberikan prediksi x yang dinotasikan dengan $h(x, \theta_k)$, dimana θ_k merupakan parameter acak hasil *bootstrapping* pada pohon tersebut. Fungsi $I(\dots)$ digunakan untuk menghitung banyaknya pohon yang memilih kelas tertentu Y. Fungsi bernilai 1 jika prediksi pohon ke- k sama dengan kelas Y, dan bernilai 0 apabila tidak sama. Dengan

demikian, penjumlahan $\sum_{k=1}^{n} I\ (h(x, \theta_k) = Y)$ menghasilkan total suara untuk kelas Y dari seluruh pohon dalam random forest. Selanjutnya, $\underset{Y}{argmax}$ memilih kelas Y yang memperoleh jumlah suara terbanyak.

Random forest memiliki beberapa keunggulan, antara lain kemampuan untuk menangani data yang mengandung *noise*, kecepatan kinerja dan kontrol terhadap *overfitting* (Guo et al., 2021). Keunggulan ini dibuktikan oleh sebuah penelitian terdahulu yang dimana random forest mencapai akurasi tertinggi yaitu 97,20% dibandingkan 2 algoritma lainnya dalam melakukan prediksi diabetes.

1.6.7 Logistic Regression

Logistic regression merupakan salah satu metode klasifikasi dalam *machine learning* yang digunakan untuk memodelkan probabilitas dari suatu variabel dependen kategorikal, khususnya biner. Pada model ini, variabel dependen atau respon hanya memiliki dua kemungkinan nilai, misalnya 1 (positif/ya/sukses) dan 0 (negatif/tidak/gagal). Tujuan utama dari regresi logistik adalah memperkirakan peluang suatu data atau observasi masuk ke dalam salah satu kelas berdasarkan variabel prediktor yang dimilikinya (Setyawan et al., 2025).

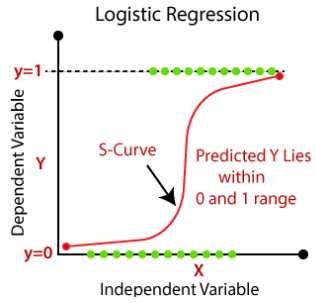
Berbeda dengan regresi linier yang menghasilkan *output* kontinu, regresi logistik menggunakan fungsi logistik atau fungsi sigmoid untuk memetakan kombinasi linier dari variabel input menjadi nilai probabilitas dalam rentang 0 hingga Fungsi sigmoid ini dirumuskan sebagai:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

Dengan:

$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + .. \beta_n X_n$

Di mana β_0 merupakan intersep, $\beta_1, \beta_2, \dots, \beta_n$ adalah koefisien regresi yang diperoleh dari data latih, dan $X_1, X_2, \dots, X_n$ adalah variabel independent atau predictor. Fungsi sigmoid ini memungkinkan model untuk mengonversi output linier menjadi probabilitas sehingga lebih sesuai untuk kasus klasifikasi biner (Setyawan et al., 2025). Berikut ilustrasi model logistic regression.

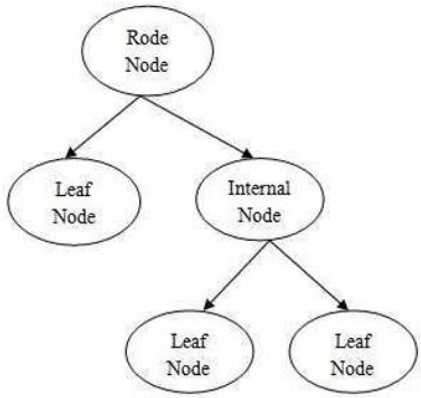


Gambar 9 Ilustrasi Logistic Regression

Logistic Regression biner banyak digunakan dalam berbagai bidang karena mampu menjelaskan hubungan antara variabel independen dengan variabel dependen kategorikal. Model ini tidak hanya memberikan hasil klasifikasi, tetapi juga memberikan probabilitas sebagai ukuran keyakinan model terhadap prediksi (Britanthia et al., 2020).

1.6.8 Decision Tree

Decision tree adalah suatu algoritma yang popular untuk prediksi dan klasifikasi data yang merupakan suatu struktur diagram alur berbentuk pohon yang terdiri dari simpul (*node*) dan rusuk (*edge*) (Nasrullah, 2021). Di setiap *node*, salah satu fitur data akan dievaluasi untuk membagi pengamatan dalam proses pelatihan atau untuk membuat titik data tertentu mengikuti jalur tertentu saat membuat prediksi. Metode ini dibangun dengan mengevaluasi fitur yang berbeda secara rekursif dan menggunakan fitur yang terbaik untuk membagi data pada setiap *node* (Thorn, 2020). Simpul pada sebuah pohon dibagikan menjadi tiga bagian, yaitu simpul akar (*root node*), simpul percabangan/internal (*branch/internal node*) dan simpul daun (*leaf node*) yang menyimpan informasi label kelas. Berikut ini struktur dari decision tree.



Gambar 10 Struktur Decision Tree (Zhou et al., 2020)

a. *Root node* adalah simpul teratas dalam struktur pohon atau *node* yang memulai grafik. Pada node ini tidak ada *input* dan bisa tidak mempunyai *output* atau mempunyai *output* lebih dari satu.
b. *Internal node*, memproses subset data dan membaginya ke dalam dua atau lebih cabang berdasarkan nilai fitur tertentu.
c. *Leaf node* merupakan node akhir atau hasil akhir dari decision tree, pada *node* ini hanya terdapat satu *input* dan tidak mempunyai *output*.

Decision tree merupakan representasi sederhana dari teknik klasifikasi untuk sejumlah kelas berhingga, dimana *internal node* maupun *root node* ditandai dengan nama atribut, rusuk-rusuknya diberi label nilai atribut yang mungkin dan *leaf node* ditandai dengan kelas-kelas yang berbeda (Nasrullah, 2021).

Terdapat beberapa algoritma turunan decision tree yang umum digunakan antara lain ID3, C4.5, dan CART (*Classification and Regression Tree*). Algoritma ID3 menggunakan ukuran information gain berdasarkan entropy untuk menentukan atribut terbaik dalam pemisahan data. Algoritma C4.5 mengembangkan ID3 dengan memperkenalkan gain ratio untuk mengatasi bias terhadap fitur dengan banyak kategori. Sementara itu, CART menggunakan ukuran Gini Index sebagai kriteria pemisahan data.

Pemilihan atribut terbaik pada setiap node dilakukan dengan menghitung tingkat ketidakmurnian (*impurity measure*) dari data. Salah satu ukuran *impurity* yang digunakan oleh CART adalah Gini Index, yang dapat dihitung dengan rumus berikut (Zhang et al., 2023):

$$Gini(S) = 1 - \sum_{i=1}^{c} p_i^2 \tag{4}$$

Dimana:

S = Himpunan data pada suatu node

c = Jumlah kelas

p_i = Proporsi kelas ke-i di dalam node

Algoritma ini mampu menyederhanakan data yang kompleks menjadi struktur hierarkis yang lebih mudah dimengerti. Salah satu keunggulan utama dari algoritma decision tree adalah kemampuannya dalam menginterpretasikan masalah kompleks dan menyajikan solusi secara intuitif dalam proses pengambilan keputusan. Hal ini membuatnya menjadi pilihan yang sering digunakan dalam penelitian dan aplikasi di berbagai bidang (Mellina et al., 2023).

1.6.9 *Particle Swarm Optimization* (PSO)

Particle Swarm Optimization (PSO) adalah salah satu dari algoritma metaheuristik yang sering digunakan untuk mencari solusi optimal dalam masalah optimisasi. Metode terinsipirasi dari perilaku sosial kelompok-kelompok alami, seperti kawanan burung atau ikan. Dalam kawanan tersebut meteka tidak memiliki pemimpin sehingga mereka akan menyebar secara acak untuk menemukan lokasi makanan. Sebagai contoh, kawanan burung pergi mencari makanan secara acak, kemudian burung yang paling dekat dengan sumber makanan akan mengirimkan sinyal kepada kawanannya untuk terbang menuju sumber makanan tersebut. Dalam metode ini, burung disebut sebagai partikel, sinyal disebut sebagai posisi dan kecepatan serta makanan disebut sebagai solusi (Soepangkat et al., 2020).

Populasi dibentuk secara acak dengan nilai terkecil dan terbesar yang telah ditetapkan sebagai batas bawah dan batas atas. Setiap partikel menemukan solusi dengan melakukan penyesuaian terhadap posisi terbaik pribadinya (*pbest*) dan penyesuaian terhadap posisi terbaik dari seluruh populasi partikel (*gbest*) selama proses iterasi pencarian berlangsung.

1.6.10 *Ensemble Learning*

Ensemble Learning merupakan salah satu pendekatan dalam *Machine Learning* yang mengombinasikan hasil prediksi dari beberapa model untuk memperoleh performa yang lebih baik. Teknik ini biasanya menghasilkan akurasi yang lebih tinggi dibandingkan dengan penggunaan model tunggal, sekaligus mampu menurunkan tingkat kesalahan serta meningkatkan stabilitas prediksi (Mienye et al., 2022). Beberapa metode yang umum digunakan dalam *Ensemble Learning* di antaranya adalah *Bagging, Boosting, Stacking*, dan *Voting* (Finda et al., 2024).

Salah satu bentuk *Ensemble Learning* yang cukup populer adalah *Voting Ensemble*, yaitu teknik yang menjalankan beberapa model berbeda pada dataset yang sama, kemudian menggabungkan hasil prediksi dari setiap model untuk memperoleh keputusan akhir yang lebih akurat dan konsisten. *Voting Ensemble* sendiri terbagi menjadi dua jenis, yaitu *Hard Voting* dan *Soft Voting* (Peppes et al., 2021).

A. *Soft Voting*

Soft voting merupakan sebuah pendekatan *ensemble learning* yang digunakan untuk menggabungkan hasil prediksi dari beberapa model pembelajaran mesin yang berbeda dalam proses klasifikasi (Oliveira et al., 2022). Dalam metode klasifikasi ini, pada prinsipnya setiap model dasar (*base classifier*) menghasilkan *ouput* berupa probabilitas terhadap kelas target kemudian nilai rata-rata probabilitas dari setiap kelas yang diberikan oleh setiap model akan digunakan untuk menentukan kelas dengan rata-rata probabilitas tertinggi (Manconi et al., 2022).

Penentuan probabilitas tiap model

Pada metode *soft voting*, keluaran tiap model bukan berupa kelas 0 atau 1 tetapi berupa probabilitas suatu sampel termasuk ke dalam kelas tertentu. Oleh karena itu, sebelum dilakukan proses ensemble, masing-masing model terlebih dahulu menghasilkan probabilitas untuk setiap data. Probabilitas ini diperoleh dengan mekanisme yang berbeda pada tiap jenis algoritma seperti berikut.

- Logistic regression

Logistic regression menghasilkan probabilitas menggunakan fungsi sigmoid dengan menghitung terlebih dahulu skor linier nya seperti berikut.

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + .. \beta_n X_n \quad (5)$$

Dimana β adalah bobot yang diperoleh dari proses *training* menggunakan algoritma optimasi seperti lbfgs, liblinear dan saga sedangkan X adalah fitur-fitur yang digunakan. Kemudian skor ini dimasukkan ke fungsi sigmoid untuk menjadi nilai probabilitas dengan rumus berikut.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (6)$$

Berikut contoh perhitungan probabilitas untuk logistic regression:

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + .. \beta_n X_n$$
$$= -3.3357415 + \big(-0.30 * (-0.761684)\big) + \big(1.15 * (-0.663943)\big) + (0.78 * (-0.185065)) + \big(1.65 * (-0.164448)\big) + \big(1.58 * (-0.161474)\big) + \big(1.59 * (-0.167375)\big) \big(0.62 * (-0.224699)\big) + \big(0.38 * (-0.204772)\big) + \big(0.51 * (-0.149045)\big) + (0.91 * (-0.135148)) + (1.52 * (-0.145376)) + (0.79 * (-0.227787)) + (1.14 * (-0.191611)) + \big(0.32 * (-0.097562)\big) + \big(0.78 * (-0.098161)\big) + (0.42 * 1.525683)$$
$$= -55147$$

Kemudian nilai z tersebut dimasukkan ke fungsi sigmoid

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$
$$= \frac{1}{1 + e^{-(-55147)}}$$
$$= 0.004$$

Probabilitas yang dihasilkan oleh fungsi sigmoid adalah probabilitas untuk kelas positif atau pada penelitian ini adalah kelas diabetes. Sedangkan untuk kelas non-diabetes, probabilitasnya adalah 1 dikurang dengan hasil dari fungsi sigmoid.

- Random forest

Random forest terdiri dari banyak decision tree yang dilatih menggunakan subset data dan jafitur yang berbeda. Setiap tree menghasilkan probabilitas kelas berdasarkan distribusi data pada *leaf node*. Probabilitas akhir model merupakan rata-rata dari probabilitas yang dihasilkan seluruh tree. Setelah seluruh pohon memberikan prediksinya, random forest menentukan probabilitas suatu kelas berdasarkan proporsi jumlah pohon yang memilih kelas tersebut.

Contoh:

Jika terdapat 500 pohon dan untuk sampel tertentu hanya 10 pohon yang memprediksi kelas 1 sementara 490 lainnya memprediksi kelas 0, maka probabilitas nya seperti berikut.

$$P(1) = \frac{10}{500} = 0.02\ (2\%) \qquad P(0) = \frac{490}{500} = 0.98\ (98\%)$$

- Decision tree

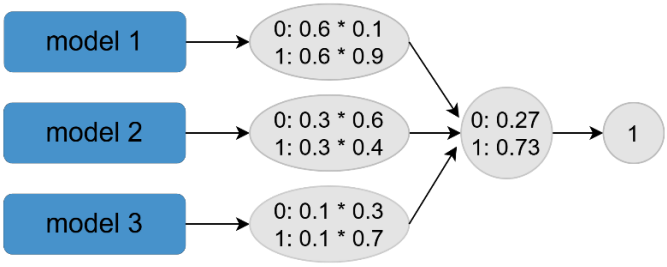
Berbeda dengan random forest yang menggabungkan banyak pohon, decision tree hanya terdiri dari satu pohon tunggal. Penentuan probabilitas dilakukan secara langsung berdasarkan komposisi kelas pada *leaf node* tempat sampel tersebut berakhir. Ketika sebuah sampel melewati serangkaian pemisahan fitur dan akhirnya berhenti di sebuah *leaf*, maka probabilitas kelas ditentukan dari proporsi data pelatihan yang ada pada *leaf* tersebut.

Contoh:

sebuah *leaf* mengandung 25 sampel kelas 0 dan 5 sampel kelas 1. Maka probabilitasnya seperti berikut.

$$P(1) = \frac{5}{30} = 0.17\ (17\%) \qquad P(0) = \frac{25}{30} = 0.83\ (83\%)$$

Probabilitas prediksi dengan rata-rata tertinggi dianggap sebagai pemenang dan dipilih untuk melakukan prediksi pada contoh baru. Berikut gambaran struktur *soft voting*.



Gambar 11 *Soft Voting Classifier* (Manconi et al., 2022)

Setiap model tidak hanya menghasilkan skor prediksi tetapi juga setiap model akan diberi bobot. Kemudian, probabilitas ini dijumlahkan untuk setiap kelas dan diambil rata-ratanya dan dikalikan dengan bobot setiap model untuk mendapatkan nilai akhir probabilitas atau skor untuk setiap kelas. Bobot yang diberikan pada setiap pengklasifikasi untuk melakukan prediksi p dapat diterapkan dengan menggunakan persamaan berikut (Kumar et al., 2020).

$$\hat{y} = arg\ max_i \sum_{j=1}^{m} w_j p_{ij} \tag{7}$$

Dimana:

m = jumlah model

w_j = bobot yang diberikan ke pengklasifikasi ke-j.

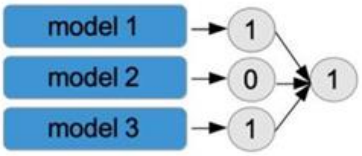
p_{ij} = probabilitas prediksi model

Soft voting memberikan hasil dan kinerja yang lebih baik karena mengandalkan rata-rata probabilitas dan dapat mengatasi kelemahan pengklasifikasi dasar individual dan meningkatkan hasil secara keseluruhan dengan menggabungkan beberapa model prediksi (Sherazi et al., 2021).

B. *Hard Voting*

Hard voting adalah salah satu pendekatan *ensemble learning* yang digunakan untuk meningkatkan hasil klasifikasi dengan cara menggabungkan prediksi dari beberapa model (*base classifiers*) sama dengan *soft voting* hanya berbeda dengan cara melakukan prediksi nya. Dalam metode ini, setiap model melakukan klasifikasi dengan memutuskan hasil prediksi akhir berdasarkan suara mayoritas dari seluruh model. Dengan kata lain, kelas yang paling banyak dipilih oleh *base classifiers* akan

menjadi hasil akhir prediksi (Morgan-Benita et al., 2022). Berikut gambaran cara kerja *hard voting*.



Gambar 12 *Hard Voting Classifier* (Manconi et al., 2022)

Jadi, jika dalam skenario terdapat 3 model, dua diklasifikasian sebagai “1” dan satu diklasifikasikan sebagai “0”, maka kelas yang diprediksi akan menjadi “1” karena memiliki mayoritas suara dari klasifikasi (Oliveira et al., 2022). *Hard voting* dapat diilustrasikan dengan persamaan berikut (Cholis, 2023).

$$\hat{y} = mode\{h_1(x), h_2(x), \ldots, h_n(x)\} \quad (8)$$

Dimana:

$h_1(x), h_2(x), \ldots, h_n(x)$ = hasil prediksi masing-masing model

$mode\{\}$ = Mayoritas suara

Penerapan *Voting Ensemble* sangat bermanfaat pada kondisi ketika terdapat perbedaan hasil prediksi antar model. Dengan mengombinasikan hasil prediksi, risiko kesalahan dari satu model dapat diminimalkan karena prediksi yang lebih tepat dari model lain akan ikut berkontribusi dalam keputusan akhir.

1.6.11 *Recursive Feature Elimination with Cross-Validation* (RFECV)

Feature selection adalah proses untuk memilih subset dari fitur-fitur asli yang paling berpengaruh dalam proses pembentukan model. Tujuannya adalah untuk menghasilkan subset optimal dari fitur yang relevan tanpa menambah kompleksitas model atau algoritma (Pameka et al., 2024). Seleksi fitur memiliki 3 kategori algoritma yaitu metode *filter, wrapper* dan *embedded*.

1. Metode *Filter*
   Metode ini menerapkan ukuran statistik untuk memberikan skor pada setiap fitur. Fitur-fitur diberikan peringkat berdasarkan skor dan dipilih untuk disimpan atau dihapus dari kumpulan data.
2. Metode *Wrapper*
   Metode ini mempertimbangkan pemilihan kombinasi fitur yang dibentuk, diuji serta dibandingkan satu sama lain dengan bantuan model prediktif. Model tersebut digunakan untuk menilai setiap kombinasi fitur dan memberikan skor berdasarkan tingkat akurasi yang dihasilkan. Salah satu contoh metode ini adalah *Recursive Feature Elimination* (RFE).
3. Metode *Embedded*

Metode ini mempelajari fitur mana yang paling berkontribusi terhadap akurasi model saat model sedang dibuat. Jenis metode ini yang paling umum adalah metode regularisasi.

Salah satu metode pemilihan fitur *wrapper* adalah *Recursive Feature feature Elimination* (RFE). RFE bekerja dengan cara mengurangi fitur yang tidak saling terkait atau lemah dan dilakukan terus menerus secara rekursif hingga fitur terbaik didapatkan untuk digunakan dalam membangun model. Fitur yang tersisa tersebut akan digunakan untuk membangun model dan dihitung nilai akurasinya. Fitur diperingkatkan secara relatif sesuai dengan urutan eliminasi (Pratama et al., 2022).

RFE kemudian diperluas dengan memilih kinerja terbaik berdasarkan skor *cross-validation*. *Cross-validation* merupakan komponen kedua dari RFECV yang mengevaluasi model dengan membagi menjadi beberapa subset atau *fold*. Model dilatih pada beberapa *fold* kemudian diuji pada *fold* yang tersisa untuk memastikan bahwa model tidak *overfit* terhadap data *training*. Dengan menggabungkan RFE dengan *cross-validation*, fitur-fitur yang tidak relevan bukan hanya dihilangkan secara berurutan tetapi juga dapat mengevaluasi kinerja model pada setiap penghapusan fitur menggunakan teknik *cross-validation* (Harif et al., 2024). Teknik ini memberi kesempatan pada seluruh set data untuk menjadi testing set sebanyak *k*-1, dimana dataset *k* adalah jumlah partisi pada *cross validation*. Ini dapat memastikan bahwa fitur-fitur yang dipilih adalah fitur yang dapat memberikan konsistensi pada kinerja model di berbagai subset data (Wibowo et al., 2024).

Manfaat dari *feature selection* antara lain, mengurangi dimensi data, menghilangkan fitur yang tidak relevan atau berpengaruh, meningkatkan pemahaman terhadap data, serta berpotensi untuk meningkatkan akurasi algoritma dengan fokus pada fitur yang paling penting (Pameka et al., 2024). Keunggulan ini telah dibuktikan oleh beberapa penelitian, salah satunya adalah penelitian oleh Wibowo et al (2024) yang melakukan optimalisasi seleksi fitur RFECV dalam credit scoring menggunakan ANN dan menghasilkan bahwa RFECV terbukti efektif dalam meningkatkan kinerja model.

1.6.12 *Confusion Matrix*

Confusion matrix merupakan metode pengujian yang digunakan untuk menilai performa suatu algoritma *machine learning*. Hal ini dilakukan dengan membandingkan prediksi yang dihasilkan oleh algoritma terhadap data sebenarnya yang telah diketahui sebelumnya (Antika et al., 2023). *Confusion matrix* memberikan gambaran jelas tentang sejauh mana algoritma dapat memprediksi dengan benar setiap kelas atau label pada dataset yang berisi nilai *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) (Heydarian et al., 2022). Berikut adalah tabel dari *confusion matrix*:

Aktual	**Prediksi**	
	Positif	Negatif
Positif	*True Positive* (TP)	*False Positive* (FP)

Negatif	*False Negative* (FN)	*True Negative* (TN)

Table 1 Confusion matrix

Dimana:
TP = sampel bernilai positif dan diprediksi sebagai positif.
FP = sampel bernilai negatif dan diprediksi sebagai positif.
FN = sampel bernilai positif dan diprediksi sebagai negatif.
TN = sampel bernilai negatif dan diprediksi sebagai negatif.

Penilaian menggunakan *confusion matrix* memberikan nilai *Accuracy, Recall, Precision,* dan *F1-Score*. Perhitungannya dapat dijelaskan sebagai berikut (Kumari et al., 2021).

1. *Accuracy*, memberikan gambaran jelas mengenai seberapa baik model dalam mengklasifikasikan secara keseluruhan. Ini menggambarkan tingkat keakuratan model dalam melakukan prediksi. Berikut persamaan untuk menghitung nilai *accuracy* (Kumari et al., 2021):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

2. *Precision*, metrik yang menilai seberapa akurat prediksi positif dibandingkan dengan total prediksi positif yang dibuat oleh model. Dalam kata lain, precision memberi informasi tentang seberapa akurat model dalam memprediksi kelas positif. Rumus *precision* ditulis pada persamaan berikut (Kumari et al., 2021):

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

3. *Recall*, mengukur kemampuan model dalam mendeteksi kasus positif. Dalam kata lain, *recall* memberi informasi tentang seberapa akurat model dalam menemukan kelas positif. Berikut rumus untuk menghitung *recall* (Kumari et al., 2021):

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

4. *F1-score*, digunakan untuk mengukur keseimbangan antara *recall* dan *precision*. Semakin tinggi nilai *f1-score*, semakin baik model dalam melakukan klasifikasi yang seimbang antara *precision* dan *recall*. Berikut rumus persamaan untuk menghitung *f1-score* (Antika et al., 2023):

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{12}$$

BAB II
METODOLOGI PENELITIAN

2.1 Waktu dan Lokasi Penelitian

Penelitian ini dilakukan sejak disetujuinya judul proposal pada bulan April 2025. Pelaksanaan penelitian dilakukan di Lab *Ubiquitous Computing and Networking* (UBICON), Departemen Teknik Informatika, Fakultas Teknik, Universitas Hasanuddin. Adapun lokasi pengambilan data penelitian pada bulan Juni 2025 di Puskesmas Balangnipa yang berlokasi di Jl. Dr Sutomo No.14 Kelurahan Balangnipa, Sinjai, Sulawesi Selatan.

Gambar 13 Lokasi Puskesmas Balangnipa Sinjai

2.2 Instrumen Penelitian

Penelitian ini menggunakan beberapa perangkat berikut:

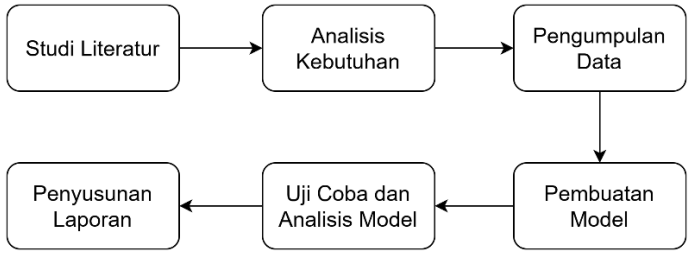
a. Perangkat keras (*Hardware*)
   MSI GF63 Thin 11SC, 11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz 2.69 GHz, RAM 8GB
b. Perangkat lunak (*Software*)
   - Google Collabolatory
   - Kaggle
   - Windows 11 Home 64 bit
   - Microsoft Excel
   - Microsoft Word

- Microsoft Edge
- Google Chrome
- Google Drive

c. Bahasa pemrograman:
   Python v3.11.13
d. Algoritma yang digunakan dalam penelitian ini antara lain:
   - Random Forest
   - Logistic Regression
   - Decision Tree
e. *Library*
   - Pandas v2.2.3, digunakan untuk mengolah, serta menganalisis data.
   - Scikit-learn v1.2.2, digunakan untuk preprocessing data serta modelling dengan machine learning.
   - Numpy v1.26.4, digunakan untuk melakukan komputasi numerik atau operasi matematika.
   - Matplotlib v3.7.2, digunakan untuk melakukan visualisasi data.

2.3 Tahapan Penelitian

Penelitian ini dilakukan melalui beberapa tahapan yang disusun secara sistematis mulai dari pengumpulan referensi hingga evaluasi hasil model. Tahapan ini bertujuan untuk memastikan setiap langkah penelitian berjalan terarah dan menghasilkan model klasifikasi yang optimal. Secara umum, alur penelitian dapat dilihat pada gambar berikut.



Gambar 14 Tahapan Penelitian

2.3.1 Studi Literatur

Tahap awal penelitian dilakukan dengan meninjau literatur yang relevan, termasuk jurnal ilmiah, buku, dan artikel penelitian terdahulu mengenai klasifikasi diabetes mellitus. Tujuannya adalah untuk memahami konsep dasar *machine learning*, metode *ensemble* seperti *Hard Voting* dan *Soft Voting*, serta teknik seleksi fitur RFECV. Studi literatur ini juga membantu mengidentifikasi kesenjangan penelitian yang menjadi dasar dilakukannya studi ini.

2.3.2 Analisis Kebutuhan

Tahap ini fokus pada identifikasi kebutuhan penelitian, baik terkait data maupun metode yang akan digunakan. Analisis mencakup pemahaman masalah utama, tujuan penelitian, variabel yang terlibat, serta indikator keberhasilan model. Selain itu, keterbatasan penelitian sebelumnya dianalisis untuk menentukan arah dan kontribusi penelitian ini.

2.3.3 Pengumpulan Data

Data penelitian berasal dari dua sumber, yaitu rekam medis pasien Puskesmas Balangnipa Sinjai (2024–2025) dan dataset publik dari Kaggle. Kedua data digabungkan untuk memperluas variasi dan jumlah sampel, sehingga model dapat belajar lebih baik. Data yang dikumpulkan meliputi informasi demografis, hasil pemeriksaan klinis, serta indikator terkait diabetes mellitus.

2.3.4 Pembuatan Model

Pada tahap inti penelitian, dilakukan pembangunan model klasifikasi menggunakan Logistic Regression, Random Forest, dan Decision Tree. Ketiga model kemudian digabungkan dengan metode *ensemble learning* yaitu *Hard Voting* dan *Soft Voting*, di mana bobot probabilitas pada *soft voting* dioptimasi menggunakan *Particle Swarm Optimization* (PSO). Selain itu, dilakukan seleksi fitur menggunakan RFECV untuk memilih fitur yang paling berpengaruh terhadap performa model.

2.3.5 Uji Coba dan Analisis Model

Evaluasi dilakukan untuk mengukur performa setiap model sebelum dan sesudah penerapan RFECV. Metode evaluasi menggunakan *Accuracy, Precision, Recall, F1-Score*, dan divisualisasikan dengan *Confusion Matrix*. Hasil ini digunakan untuk membandingkan efektivitas metode *Hard Voting* dan *Soft Voting*.

2.3.6 Penyusunan Laporan

Tahap terakhir adalah penyusunan laporan penelitian dalam bentuk skripsi, yang mendokumentasikan seluruh proses penelitian mulai dari studi literatur, pengumpulan dan pengolahan data, analisis, hingga hasil evaluasi model secara sistematis dan sesuai format akademik.

2.4 Teknik Pengumpulan Data

Data yang digunakan adalah data sekunder dari Puskesmas Balangnipa Sinjai berupa rekam medis pasien tahun 2024–2025. Data diperoleh dalam format *Excel* (.xls) dengan total 39.542 baris dan terdiri dari 85 fitur, mulai dari informasi umum hingga identitas pasien. Fitur yang tidak relevan atau tidak informatif dihapus, sehingga hanya tersisa variabel yang berkaitan dengan gejala atau tanda umum diabetes. Berikut adalah rincian data yang digunakan dalam penelitian ini:

Parameter	Indikasi
Umur	Umur pasien yang terdaftar pada data.
Tinggi Badan	Digunakan untuk pengukuran IMT. (cm)
Berat Badan	Digunakan untuk pengukuran IMT. (kg)
Indeks Massa Tubuh (IMT)	$IMT = \frac{Berat\ Badan\ (kg)}{Tinggi\ Badan\ (m)^2}$
Keluhan Utama dan Keluhan Tambahan	Kategori ini terdiri atas banyak macam keluhan pasien yang kemudian dianalisis untuk melihat gejala yang relevan dengan gejala diabetes.
ICD-X 1	Kolom ini berisi kode medis berdasarkan diagnosa pasien yang dimana khusus untuk pasien diabetes kode diagnosanya diawali dengan "E1" yang selanjutnya akan menjadi label target.

Table 2 Fitur data puskesmas

Selain data sekunder dari Puskesmas Balangnipa, ditambahkan pula data yang diperoleh dari dataset publik berjudul "*Early Stage Diabetes Risk Prediction Dataset*" (UCl Repository, 2020) yang tersedia pada platform Kaggle. Dataset ini berisi informasi gejala klinis pasien diabetes yang telah terstruktur, dengan 17 atribut utama, antara lain.

No.	Fitur	Keterangan
1.	*Age*	Umur dari orang-orang/pasien yang ikut serta dalam kuisioner penelitian
2.	*Gender*	Jenis kelamin pasien
3.	*Polyuria*	Sering buang air kecil dengan frekuensi lebih dari 20 kali/hari
4.	*Polydipsia*	Haus yang berlebihan hingga mencapai lebih dari 3L/hari
5.	*Sudden Weight Loss*	Penurunan berat badan secara drastis
6.	*Weakness*	Daya tahan tubuh mulai menurun
7.	*Polyphagia*	Peningkatan nafsu makan yang berlebihan
8.	*Genital trhush*	Infeksi jamur dan bakteri
9.	*Visual Blurring*	Sakit kepala dan penglihatan yang mulai buram

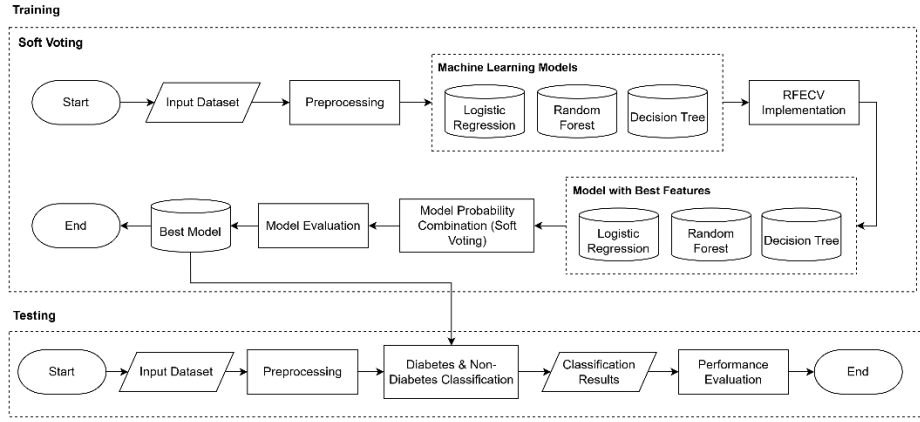
10.	*Itching*	Rasa gatal yang tidak berkesudahan
11.	*Irritability*	Pembengkakan pada tangan serta kaki
12.	*Delayed healing*	Luka yang sulit sembuh
13.	*Partial paresis*	Lemas pada badan
14.	*Muscle stiffness*	Sulit bergerak atau melemahnya kinerha otot
15.	*Alopecia*	Rambut rontok pada area tertentu sehingga membuat pitak
16.	*Obesity*	Tingkat obesitas pasien tergantung IMT
17.	*Class*	Kelas yang diklasifikasikan yaitu positif atau negatif diabetes

Table 3 Fitur data Kaggle

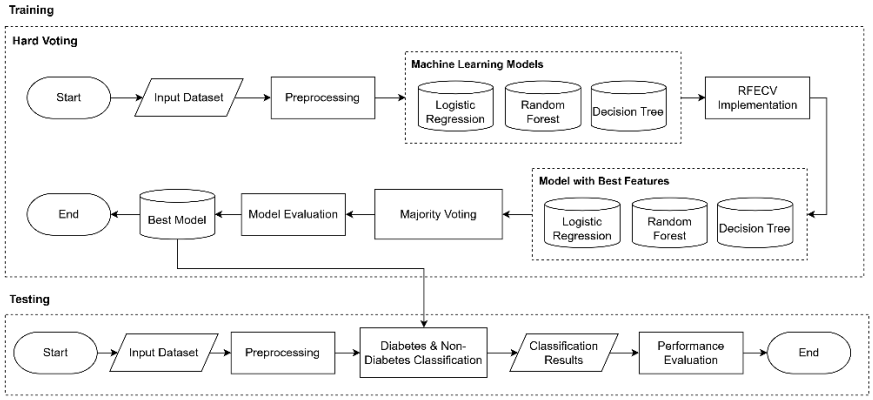
Dataset ini digunakan untuk meningkatkan proporsi kelas pasien pada data lokal, sehingga data menjadi lebih seimbang.

2.5 Perancangan dan Implementasi Sistem

Gambar berikut menampilkan *flowchart* perancangan sistem yang akan dibangun untuk melakukan klasifikasi awal diabetes. Secara garis besar, proses penelitian ini dibagi menjadi 2 yaitu pembangunan base model kemudian *voting classifier* (*Hard Voting* dan *Soft Voting*).



Gambar 15 Usulan Sistem *Soft Voting*



Gambar 16 Usulan Sistem *Hard Voting*

2.5.1 Input Data

Data awal yang digunakan dalam sistem berasal dari rekam medis pasien Puskesmas Balangnipa Sinjai pada periode 2024–2025. Selain data lokal, penelitian ini juga menggunakan dataset publik dari Kaggle untuk menambah jumlah kasus positif diabetes. Penelitian ini tidak hanya berfokus pada pasien yang sudah terdiagnosis, tetapi juga mencakup pasien dengan gejala awal yang berpotensi menjadi indikasi diabetes atau dasar rekomendasi pemeriksaan. Pendekatan ini memastikan analisis lebih komprehensif, mencakup kasus terkonfirmasi maupun risiko potensial yang belum terdeteksi. Rekap data pasien di Puskesmas dilakukan setiap bulan melalui sistem website resmi, dan seluruh rekap bulanan dari 2024 hingga 2025 digabung menjadi satu dataset utuh untuk keperluan penelitian.

2.5.2 *Preprocessing*

Setelah data terkumpul, proses selanjutnya adalah *preprocessing* data. Tahap ini bertujuan untuk memastikan bahwa data yang akan digunakan dalam proses pemodelan memiliki kualitas yang baik, bebas dari kesalahan serta dalam format yang sesuai dengan kebutuhan algoritma *machine learning*. Dengan adanya tahap *preprocessing*, model yang dibangun diharapkan dapat memberikan hasil yang baik. Berikut tahapan preprocessing yang dilakukan.

1. Data Lokal Puskesmas Balangnipa
   a. *Data Integration*
      Data rekam medis pasien dari Puskesmas Balangnipa Sinjai dikumpulkan dalam bentuk rekap bulanan dari April 2024 hingga Mei 2025. Langkah pertama adalah menggabungkan seluruh data bulanan menjadi satu dataset agar memudahkan pengolahan pada tahap berikutnya.
   b. *Duplicate Handling*
      Setelah penggabungan, dilakukan pengecekan data duplikat, karena satu pasien mungkin memiliki beberapa catatan dengan nomor rekam medis

yang sama akibat pemeriksaan berulang. Baris duplikat dihapus untuk mencegah bias pada proses pelatihan model.

c. *Feature Extraction*
Kolom keluhan, yang terdiri dari "keluhan utama" dan "keluhan tambahan", berisi deskripsi gejala pasien dalam bentuk teks bebas. Untuk memudahkan analisis oleh algoritma machine learning, dilakukan ekstraksi fitur dengan mengubah teks menjadi variabel biner (Ya/Tidak) yang mewakili gejala umum diabetes. Contohnya, jika pasien menyebut “sering buang air kecil”, kolom *Polyuria* diisi “Ya”; jika tidak, diisi “Tidak”. Fitur yang dihasilkan mencakup *Polyuria, Polydipsia, Sudden Weight Loss, Weakness, Polyphagia, Genital Thrush, Visual Blurring, Itching, Irritability, Delayed Healing, Partial Paresis, Muscle Stiffness, Alopecia,* dan *Obesity*.

d. *Label Validation*
Setelah ekstraksi fitur, dilakukan pemeriksaan ulang label diabetes untuk memastikan konsistensi antara diagnosis dan kondisi pasien. Beberapa pasien yang berlabel diabetes (ICD-X = E1) ternyata tidak memiliki gejala atau faktor risiko terkait, kemungkinan karena sudah terdiagnosis sebelumnya. Data seperti ini dihapus agar model tidak belajar dari informasi yang tidak representatif.

e. *Dimension Reduction*
Dilakukan penghapusan fitur yang tidak relevan atau bersifat administratif, seperti nama pasien, tanggal lahir, nomor rekam medis, alamat, dan informasi poli/ruangan. Tujuannya adalah menyederhanakan dataset, mengurangi kompleksitas, mencegah overfitting, dan meningkatkan akurasi serta efisiensi model.

f. *Data Cleaning*
Beberapa langkah pembersihan dilakukan, antara lain:
- Menangani missing values dengan menghapus baris yang tidak lengkap atau mengisinya menggunakan metode statistik (median atau modus).
- Mengidentifikasi dan menyesuaikan outlier yang tidak wajar atau ekstrem.
- Membersihkan format string, misalnya mengubah kolom Umur Tahun dari format “40 tahun” menjadi nilai numerik agar dapat diproses oleh sistem.

g. *Feature Engineering*
Tahap ini bertujuan untuk menambahkan fitur baru yang meningkatkan kemampuan model dalam mengenali pola data. Ditambahkan fitur *Obesity*, berdasarkan nilai IMT pasien, di mana IMT ≥ 25 dikategorikan “Ya” dan IMT < 25 dikategorikan “Tidak”. Selain itu, dibuat fitur Class sebagai variabel target, dengan kode ICD-X yang diawali E1 dikategorikan "Positif" dan kode lainnya sebagai "Negatif".

2. Dataset Publik Kaggle
   a. *Data Transformation*

Setelah proses pembersihan data selesai, tahap berikutnya adalah transformasi data untuk menyamakan tipe data numerik dan kategorikal antara dataset lokal dan eksternal. Beberapa atribut numerik pada dataset lokal, seperti Umur Tahun, Tinggi Badan, Berat Badan, dan IMT, tersimpan dalam format string yang menyertakan satuan, misalnya “40 tahun”. Selain itu, terdapat perbedaan format antar dataset, seperti kolom Age dibandingkan Umur Tahun, dan label diabetes antara ICD-X dan kolom class berbentuk biner (Yes/No). Oleh karena itu, penamaan fitur, tipe data, dan kategori label disesuaikan agar kedua dataset dapat diproses secara konsisten.

3. Integrasi data sekunder dan data publik
   a. *Encoding Variabel Categorical*
      Beberapa variabel bersifat kategorikal, seperti gejala diabetes berisi Ya/Tidak atau kolom gender berisi Perempuan/Laki-laki. Karena algoritma machine learning hanya menerima data numerik, variabel ini dikonversi menggunakan Label Encoding dari pustaka Scikit-Learn, sehingga setiap kategori diberi label angka. Misalnya, kolom Gender dikodekan menjadi “0” untuk Perempuan dan “1” untuk Laki-laki.
   b. *Data Filtering*
      Pada tahap ini, dilakukan penyaringan data berdasarkan usia, di mana pasien yang digunakan untuk pelatihan model berusia di atas 10 tahun. Langkah ini sesuai dengan temuan Alodokter (2025) yang menyebutkan bahwa diagnosis diabetes tipe 1 di Indonesia paling banyak terjadi pada kelompok usia 10–14 tahun, sementara diabetes tipe 2 pada anak biasanya terdiagnosis pada saat pubertas atau usia lebih dewasa.
   c. *Data Normalization*
      Normalisasi dilakukan untuk menyeimbangkan skala antar fitur, karena beberapa variabel memiliki rentang nilai yang berbeda. Contohnya, fitur Umur Tahun bernilai puluhan, sedangkan fitur gejala seperti *Polyuria* atau *Polydipsia* hanya bernilai biner (0 atau 1). Pada penelitian ini, normalisasi dilakukan menggunakan metode Standardization dengan rumus tertentu agar seluruh fitur berada pada skala yang seragam.

$$Z = \frac{X - \mu}{\sigma} \tag{13}$$

Dimana:
X = Nilai asli data
μ = Rata-rata dari kolom tersebut
σ = Standar deviasi
Hasil dari proses ini membuat setiap fitur memiliki rata-rata 0 dan standar deviasi 1. Dengan demikian, model tidak akan berat sebelah terhadap fitur yang memiliki skala lebih besar.

2.5.3 Split Data

Setelah tahap preprocessing selesai, dataset dibagi menjadi dua bagian, yaitu *training* set untuk melatih model dan testing set untuk menguji performanya. Pemisahan ini bertujuan agar evaluasi model lebih objektif dan tidak terpengaruh oleh data yang digunakan saat pelatihan. Proses pembagian dilakukan pada data yang sudah dinormalisasi, sehingga skala fitur tetap konsisten di seluruh subset.

Dalam penelitian ini, pembagian dilakukan dengan proporsi 80% untuk data latih dan 20% untuk data uji, menggunakan fungsi *train_test_split()* dari pustaka *Scikit-Learn*.

2.5.4 *Undersampling Data*

Dataset yang digunakan dalam penelitian ini mengalami ketidakseimbangan kelas, di mana jumlah pasien non-diabetes jauh lebih banyak dibandingkan pasien diabetes. Kondisi ini, yang disebut imbalance problem, dapat membuat model machine learning lebih fokus pada kelas mayoritas sehingga kinerja model dalam mendeteksi kelas minoritas dalam hal ini pasien diabetes menjadi rendah, terutama pada metrik recall.

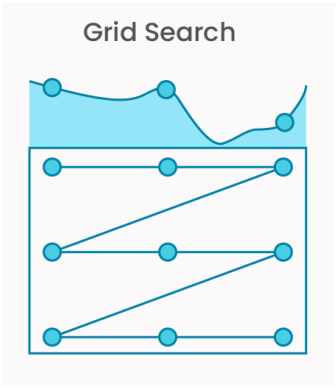
Untuk mengatasi masalah tersebut, diterapkan metode Edited Nearest Neighbors (ENN). Metode ini bekerja dengan menghapus data yang tidak konsisten dengan tetangga terdekatnya berdasarkan algoritma K-NN. Selain menyeimbangkan distribusi kelas, ENN juga membersihkan data dari sampel yang tumpang tindih atau kurang representatif, sehingga kualitas data latih meningkat dan lebih informatif bagi model. Proses ENN hanya dilakukan pada data latih agar distribusi data uji tetap sesuai kondisi aslinya.

2.5.5 *Hyperparameter Tuning*

Setelah data *training* dan *testing* hasil *undersampling* tersedia, langkah selanjutnya adalah melakukan *hyperparameter tuning* pada data *training*. *Hyperparameter* merupakan parameter yang ditentukan sebelum proses pelatihan dimulai dan berfungsi untuk meningkatkan performa algoritma klasifikasi, sehingga dapat memberikan pengaruh signifikan terhadap kinerja model (Elgeldawi et al., 2021).

Pengaturan *hyperparameter* sangat penting karena memengaruhi cara model belajar serta tingkat kompleksitas model yang terbentuk. Pemilihan nilai parameter yang tepat dapat meningkatkan akurasi dan mencegah *overfitting* (model terlalu menyesuaikan data training) atau *underfitting* (model gagal mengenali pola pada data) (Maulid, 2024).

Pada penelitian ini, *hyperparameter tuning* dilakukan untuk mencari kombinasi terbaik dari tiga algoritma yang digunakan, yaitu Logistic Regression, Random Forest, dan Decision Tree, dengan memanfaatkan metode Grid Search Cross Validation (GridSearchCV) dari pustaka *Scikit-Learn*. Gambar berikut menunjukkan visualisasi proses Grid Search (Kumar, 2024).



Gambar 17 Grid Search

Grid Search bekerja dengan mengeksplorasi semua kombinasi *hyperparameter* yang mungkin. Setiap kombinasi dievaluasi dengan menguji kinerja model pada berbagai bagian dataset untuk mengukur akurasinya. Dari seluruh kombinasi, Grid Search akan memilih konfigurasi yang memberikan hasil terbaik. Karena sifatnya yang menyeluruh, metode ini membutuhkan waktu yang cukup lama, terutama jika jumlah hyperparameter meningkat (Shah, 2025).

Pada penelitian ini, pemilihan kombinasi parameter didasarkan pada nilai recall, karena tujuan utama adalah mengevaluasi kemampuan model dalam mendeteksi pasien yang berisiko diabetes. Berikut ini adalah parameter yang diterapkan pada masing-masing model:

a. Logistic Regression

Hyperparameters	**Keterangan**
penalty	Jenis regularisasi yang digunakn untuk mengontrol kompleksitas model (L1 atau L2)
C	Nilai konstanta regularisasi yang mengontrol kekuatan penalty (semakin kecil nilainya, semakin kuat regularisasi)
solver	Metode optimisasi yang digunakan untuk mencari parameter model (liblinear, saga dan lbfgs)

Table 4 Hyperparameter logistic regression

b. Random Forest

Hyperparameters	**Keterangan**
n_estimators	Jumlah pohon keputusan yang dibentuk
max_depth	Kedalaman maksimum dari setiap pohon
min_samples_split	Jumlah minimum sampel yang dibutuhkan untuk memecah node

min_samples_leaf	Jumlah minimum sampel yang harus ada pada node daun
max_features	Jumlah maksimal fitur di setiap split
bootstrap	Metode pengambilan sampel acak saat pembentukan pohon

Table 5 Hyperparameter random forest

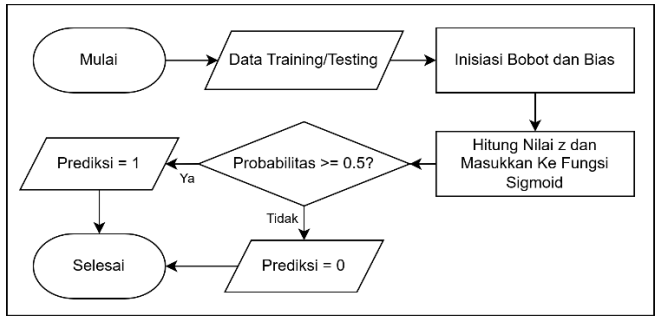
c. Decision Tree

Hyperparameters	**Keterangan**
criterion	Metode pengukuran pemisahan data, yaitu Gini Index atau entropy
max_depth	Kedalaman maksimum pohon keputusan
min_samples_split	Jumlah minimum sampel yang dibutuhkan untuk memecah node
min_samples_leaf	Jumlah minimum sampel yang harus ada pada node daun
max_features	Jumlah maksimal fitur di setiap split

Table 6 Hyperparameter decision tree

2.5.6 Klasifikasi Logistic Regression

Setelah melalui *preprocessing* sehingga didapatkan hasil split *data training* dan *data testing* selanjutnya dilakukan proses klasifikasi untuk melatih model dalam melakukan diagnosis awal diabetes. Klasifikasi dilakukan dengan menggunakan tiga algoritma yang salah satunya adalah logistic regression. Algoritma ini bekerja dengan memodelkan probabilitas hasil kategoris misalkan ya/tidak, sukses/gagal. Berikut tahapan alur proses klasifikasi logistic regression.



Gambar 18 Klasifikasi Logistic Regression

Diagram alur ini menggambarkan proses klasifikasi diagnosis awal diabetes menggunakan algoritma logistic regression. Setelah data *training* dan *testing* disiapkan, tahap pertama yang dilakukan model adalah menginisialisasi parameter

berupa bobot dan bias. Parameter ini menjadi titik awal proses perhitungan yang nantinya diperbarui selama pelatihan untuk menyesuaikan keputusan model.

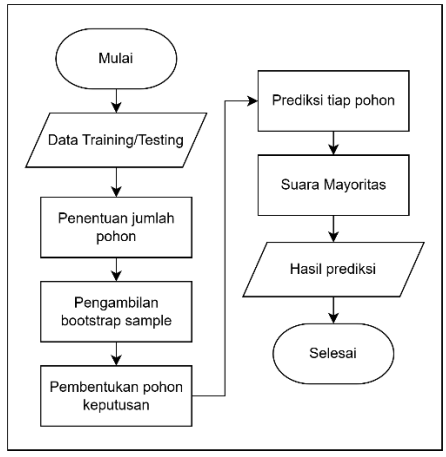
Setiap fitur pada data pasien kemudian dikalikan dengan bobot masing-masing, di mana bobot tersebut merepresentasikan tingkat pengaruh fitur terhadap kemungkinan pasien terindikasi diabetes atau non-diabetes. Hasil perkalian seluruh fitur dengan bobotnya dijumlahkan bersama nilai bias sehingga menghasilkan nilai *z*, yaitu kombinasi linear dari fitur.

Nilai *z* selanjutnya dimasukkan ke dalam fungsi sigmoid untuk dikonversi menjadi probabilitas dengan rentang 0 hingga 1. Probabilitas ini menggambarkan besarnya peluang seorang pasien menderita diabetes berdasarkan pola yang dipelajari model dari data training.

Pada tahap pengambilan keputusan, Logistic Regression menggunakan nilai ambang (*threshold*) 0.5. Jika probabilitas yang dihasilkan bernilai ≥ 0.5, maka model mengklasifikasikan pasien sebagai diabetes (prediksi = 1). Sebaliknya, jika probabilitas < 0.5, pasien dikategorikan sebagai non-diabetes (prediksi = 0).

2.5.7 Klasifikasi Random Forest

Algoritma selanjutnya yang digunakan dalam proses klasifikasi adalah random forest. Algoritma ini bekerja dengan cara menggabungkan banyak pohon keputusan (decision tree) untuk menghasilkan prediksi yang lebih stabil. Adapun diagram alur yang memperlihatkan cara kerja random forest sebagai berikut.



Gambar 19 Klasifikasi Random Forest

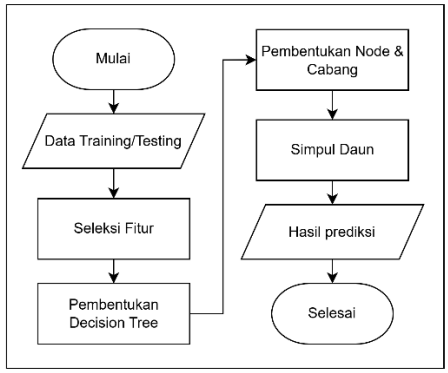
Klasifikasi menggunakan random forest dimulai dengan pembentukan banyak pohon keputusan, di mana setiap pohon dilatih melalui *bootstrap sampling*. Teknik ini memilih sampel data secara acak dari dataset asli dengan pengembalian, sehingga setiap pohon mendapatkan subset data yang berbeda dan memiliki struktur yang unik, meskipun algoritma dasarnya tetap pohon keputusan.

Setiap pohon kemudian melakukan prediksi secara terpisah pada data uji. Hasil prediksi individu ini kemudian digabungkan dalam mekanisme *majority voting*,

di mana keputusan akhir ditentukan oleh kelas yang paling banyak dipilih oleh seluruh pohon. Dengan demikian, jika mayoritas pohon memprediksi pasien termasuk kategori diabetes, klasifikasi akhir adalah diabetes; sebaliknya, jika mayoritas memprediksi non-diabetes, hasil akhirnya adalah non-diabetes.

2.5.8 Klasifikasi Decision Tree

Algoritma ketiga selain logistic regression dan random forest adalah decision tree. Algoritma ini mampu menghasilkan model klasifikasi sederhana serta interpretatif. Model ini membangun representasi data dalam bentuk pohon keputusan yang terdiri dari *node*, cabang dan simpul daun. Struktur pohon inilah yang menjadi dasar dalam menentukan kelas dari setiap data baru yang dimasukkan. Proses klasifikasi ini dapat dilihat pada gambar berikut.



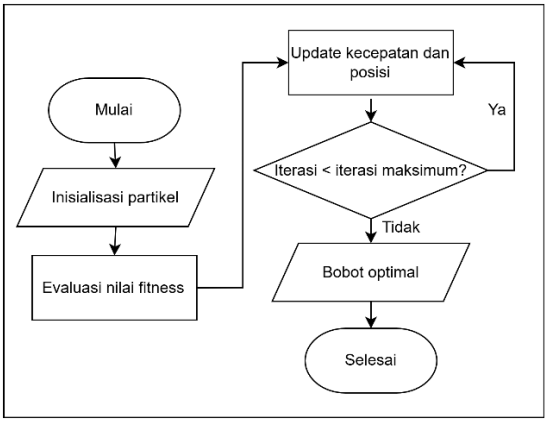
Gambar 20 Klasifikasi Decision Tree

Data yang akan digunakan selanjutnya menjalani seleksi fitur, yaitu proses untuk menentukan atribut mana saja yang paling relevan dalam membangun pohon keputusan. Tahap ini penting karena pemilihan fitur yang tepat akan memungkinkan pohon membuat pembagian data yang lebih akurat.

Setelah fitur terpilih, model masuk ke tahap pembentukan pohon keputusan, di mana algoritma menentukan fitur terbaik yang mampu memisahkan data ke dalam kelas yang berbeda secara optimal. Algoritma kemudian membuat node berdasarkan fitur tersebut dan membentuk cabang sesuai dengan nilai yang dimiliki fitur. Data diarahkan mengikuti cabang yang sesuai hingga terbentuk simpul daun, yang berisi label kelas akhir, yaitu diabetes atau non-diabetes.

2.5.9 Optimasi Bobot PSO

Sebelum memasuki evaluasi dengan *ensemble learning*, dilakukan terlebih dahulu optimasi bobot khususnya untuk metode *soft voting*. Berikut tahapan umum proses pencarian bobot optimal dengan menggunakan PSO.



Gambar 21 Optimasi Bobot PSO

Flowchart tersebut menggambarkan alur kerja algoritma *Particle Swarm Optimization* (PSO) untuk menentukan bobot optimal dalam perhitungan *soft voting*. Proses diawali dengan inisialisasi partikel, di mana partikel-partikel dibentuk secara acak dan masing-masing merepresentasikan solusi potensial, lengkap dengan parameter posisi dan kecepatan sesuai batas yang ditetapkan.

Setelah itu, setiap partikel dievaluasi menggunakan nilai *fitness* yang menunjukkan kualitas solusi yang diusulkan. Berdasarkan hasil ini, dilakukan pembaruan posisi dan kecepatan, yang memperhitungkan *personal best* (pbest), posisi terbaik partikel itu sendiri, dan *global best* (gbest), posisi terbaik seluruh partikel, sehingga partikel bergerak menuju solusi yang lebih optimal. Posisi adalah letak tiap partikel yang dimana satu partikel terdiri atas satu kandidat kombinasi bobot. Contohnya partikel ke-1 adalah [0.45, 0.35, 0.20] kemudian partikel ke-2 adalah [0.60, 0.25, 0.15], dst. Satu partikel terdiri atas 3 kandidat bobot yang jika dijumlahkan hasilnya 1 sesuai dengan jumlah model atau algoritma yang digunakan. Sedangkan kecepatan pada PSO menunjukkan besarnya perubahan nilai bobot dari satu iterasi ke iterasi berikutnya. Pada penelitian ini, digunakan 30 partikel untuk menemukan bobot optimal sesuai dengan penelitian oleh (Istikomah, et al., 2017) yang membuktikan bahwa 30 partikel memberikan bobot paling optimal diantara 5 – 50 partikel yang diuji coba.

Proses ini berlangsung iteratif hingga tercapainya jumlah iterasi maksimum. Setelah berhenti, partikel dengan nilai fitness tertinggi akan dipilih sebagai bobot optimal untuk *soft voting*.

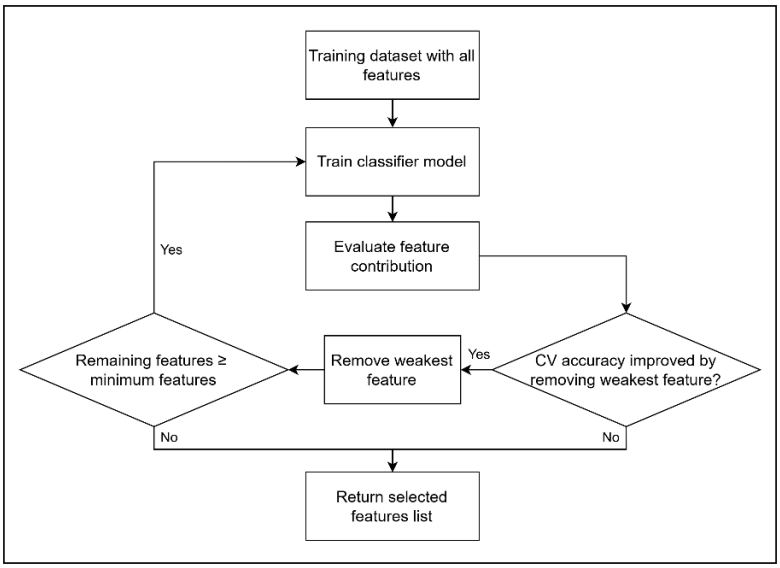
2.5.10 Penerapan *Feature Selection*

Setelah model berhasil dibuat, tahap berikutnya adalah menentukan fitur-fitur yang paling berpengaruh terhadap kinerja model klasifikasi menggunakan metode *Recursive Feature Elimination with Cross-Validation* (RFECV). RFECV merupakan pengembangan dari metode RFE, di mana proses seleksi fitur dilakukan secara bertahap dan dievaluasi melalui *cross-validation* sehingga hasil yang diperoleh

lebih stabil dan tidak terlalu tergantung pada dataset tertentu. Secara umum, proses RFECV meliputi empat langkah utama (Shi et al., 2024):

a. Melatih model dasar menggunakan dataset dengan seluruh fitur asli.
b. Menilai kepentingan masing-masing fitur.
c. Membentuk subset fitur baru dengan menghapus fitur-fitur yang kurang penting melalui proses *cross-validation*.
d. Mengulangi langkah a hingga c sampai diperoleh jumlah fitur yang optimal.

Gambar berikut menunjukkan mekanisme kerja RFECV yang digunakan dalam penelitian ini, mulai dari proses pelatihan model hingga dihasilkan jumlah fitur optimal berdasarkan hasil *cross-validation*.



Gambar 22 Cara kerja RFECV

Dalam penelitian ini, RFECV digunakan untuk mengidentifikasi fitur-fitur yang paling relevan dalam klasifikasi diabetes. Metode ini membantu menyederhanakan model dan meningkatkan efisiensi komputasi tanpa menurunkan akurasi secara signifikan.

2.5.11 Implementasi *Voting Classifier*

Voting classifier digunakan sebagai tahap terakhir dalam proses klasifikasi untuk menghasilkan prediksi yang lebih stabil dan akurat. Metode ini bekerja dengan menggabungkan prediksi dari beberapa model dasar sehingga kinerja klasifikasi lebih baik dibandingkan penggunaan satu model saja. Penelitian ini menerapkan dua jenis *voting classifier*, yaitu *hard voting* dan *soft voting*, dengan tiga algoritma dasar yaitu logistic regression, decision tree, dan random forest. Ketiga algoritma dipilih karena memiliki karakteristik yang berbeda dan saling melengkapi. Proses implementasinya dilakukan melalui beberapa tahapan:

a. *Training base model*
   Setiap algoritma dilatih secara terpisah menggunakan data hasil *undersampling*, kemudian dilakukan pelatihan ulang menggunakan data hasil seleksi fitur melalui metode RFECV.
b. Model terbaik dari ketiga algoritma digabungkan menggunakan dua pendekatan yaitu *hard voting*, di mana prediksi akhir ditentukan berdasarkan mayoritas kelas dari ketiga model, dan *soft voting*, di mana prediksi akhir diperoleh dari rata-rata probabilitas kelas yang dihasilkan tiap model.

Metode *Voting Classifier* ini diharapkan mampu mengoptimalkan kinerja prediksi dengan memanfaatkan kelebihan dari masing-masing *base model*.