PENERAPAN METODE STACKING ENSEMBLE PADA KLASIFIKASI PENYAKIT DIABETES MELITUS

PROPOSAL SKRIPSI

Oleh :

Abdul Rahem Faqih

220411100029

PROGRAM STUDI S1 TEKNIK INFORMATIKA

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS TRUNOJOYO MADURA

2025

KATA PENGANTAR

DAFTAR ISI

KATA PENGANTAR ..... **i**

DAFTAR ISI ..... **ii**

DAFTAR GAMBAR ..... **v**

DAFTAR TABEL ..... **vi**

ABSTRAK ..... **vii**

BAB I PENDAHULUAN **1**

1.1 Latar Belakang 1

1.2 Perumusan Masalah 4

1.2.1. Permasalahan 4

1.2.2. Metode Usulan 4

1.2.3. Pertanyaan Penelitian 4

1.3 Tujuan dan Manfaat 5

1.3.1. Tujuan Penelitian 5

1.3.2. Manfaat Penelitian 5

1.4 Batasan Masalah 5

1.5 Sistematika Penulisan 5

BAB II TINJAUAN PUSTAKA **7**

2.1 Diabetes Melitus 27

2.2 Data Mining 7

2.3 Machine Learning 7

2.4 *Supervised Learning* 7

2.5 *Min-max Normalization* 8

2.6 *Synthetic Minority Oversampling Technique* (SMOTE) 9

2.7 Klasikasi 9

2.8 *Stacking* 10

2.9 *Logistic Regression* (*LR*) 11
2.10 *K-Nearest Neighbor* (*KNN*) 13
2.11 *Support Vector Machine (SVM)* 14
2.12 *Artificial Neural Network* (*ANN*) 16
2.12.1. Arsitektur Backpropagation 17
2.12.2. Fungsi Aktivasi 23
2.13 *Grid Search CV* 23
2.14 Evaluasi 24
2.14.1. Cross Validation 24
2.14.2. Confusion Matrix 25
2.15 Penelitian Terkait ..... **Error! Bookmark not defined.**
BAB III METODOLOGI PENELITIAN 33
3.1 Alur Sistem 33
3.2 Pengumpulan Data 33
3.2.1. Sumber Data 34
3.2.2. Variabel Penelitian 34
3.3 Prepocessing 35
3.3.1. Data Cleaning 35
3.3.2. Tranformasi Data 35
3.3.3. Data Normalization 36
3.3.4. Data Balancing 36
3.3.5. Data Split 36
3.4 Modelling 37
3.4.1. Cross Validation 37
3.4.2. HyperParameter Tuning (GridSearchCV) 37
3.4.3. K-Nearest Neighbor 38

3.4.4. Logistic Regression 40
3.4.5. Support Vector Machine 41
3.4.6. Artificial Neural Network 43
3.4.7. Stacking Ensemble 45
3.5 Evaluasi 48
3.6 Skenario Pengujian 48
DAFTAR PUSTAKA 49

DAFTAR GAMBAR

Gambar 2.1 Konsep Penggabungan Metode dengan Stacking 11
Gambar 2.2 Fungsi Logistik..... 12
Gambar 2. 3 Hyperplane Support Vector Machine (SVM) 14
Gambar 2.4 Arsitektur Multi Layer Network 17
Gambar 2.5 Arsitektur Algoritma Backpropagation..... 18
Gambar 2.6 Grafik Fungsi Aktivasi Sigmoid 23
Gambar 2. 7 Grid Search CV 24
Gambar 2.8 Contoh Penggunaan Cross Validation..... 25
Gambar 2. 9 Confusion Matrix 25
Gambar 3. 1 Alur Sistem..... 33
Gambar 3. 2 Algoritma Normalisasi Data MinMax..... **Error! Bookmark not defined.**
Gambar 3. 3 Pembagian CrossValidation pada data training 37
Gambar 3. 4 Flowchart Klasifikai Menggunakan KNN 39
Gambar 3. 5Algoritma Klasifikasi Menggunakan LR 40
Gambar 3. 6 Algoritma Klasifikasi Menggunakan SVM..... 42
Gambar 3. 7 Algoritma Klasifikasi Menggunakan ANN..... 44
Gambar 3. 8 Algoritma Klasifikasi Stacking Ensemble 46

DAFTAR TABEL

Tabel 2. 1 Fungsi Kernel pada SVM 14
Tabel 2. 2 Penelitian Terkait 30
Tabel 3. 1 Sample Data 34
Tabel 3. 2 Atribut data yang digunakan 34
Tabel 3. 3 Kombinasi Nilai Hyperparameter 38
Tabel 3. 4 Contoh Dataset Dari Hasil Prediksi Base-Learners 47
Tabel 3. 5 Skenario Pengujian 48

ABSTRAK

BAB I
PENDAHULUAN

1.1 Latar Belakang

Klasifikasi didefinisikan sebagai proses komputasi yang termasuk ke dalam teknik machine learning dengan mengelompokkan suatu data menjadi satu kelas atau label berdasarkan atribut-atribut yang terdapat di dalam data tersebut [1]. Machine Learning merupakan metode yang menerapkan kecerdasan buatan untuk membantu kemajuan sistem komputer dengan kemampuan menganalisis data serta menentukan akurasi terkait data yang sedang diolah [2]. Dalam proses klasifikasi, data akan dibagi menjadi data training untuk mengetahui data yang dikaitkan dengan label atau kelas dan data testing sebagai sampel uji penentuan label atau kelas [3]. Model klasifikasi akan memperlihatkan kumpulan fitur yang saling berhubungan pada diagnosa dan melihat pola atau hubungan guna membangun model yang dapat mengenali pola atau hubungan yang dicari sehingga dapat diklasifikasikan untuk menghasilkan nilai dengan akurasi yang tinggi [4]. Hasil dari klasifikasi akan menunjukkan keefektivitasan Machine Learning dalam menemukan hubungan antara fitur dengan target dalam data [5].

Berbagai penelitian terkait klasifikasi penyakit diabetes menggunakan algoritma machine learning tunggal telah dilakukan dengan hasil yang beragam. Tsehay Admassu Assegie dkk. menggunakan metode K-Nearest Neighbor (KNN) dan menghasilkan tingkat akurasi sebesar 82,5%[6]. Penelitian serupa juga dilakukan oleh Erlin dkk. dengan judul "Early Detection of Diabetes Using Machine Learning with Logistic Regression Algorithm" dimana metode Logistic Regression yang diterapkan menghasilkan akurasi sebesar 82%, precision 81%, recall 79%, dan F1-score 80%[7]. Selanjutnya, Dimas Aryo Anggoro dan Dian Novitaningrum melakukan penelitian dengan judul "Comparison of Accuracy Level of Support Vector Machine (SVM) and Artificial Neural Network (ANN) Algorithms in Predicting Diabetes Mellitus Disease" menggunakan dataset dari National Institute of Diabetes and Digestive and Kidney Diseases dengan 768 data, dimana hasil perbandingan menunjukkan bahwa metode ANN memperoleh tingkat akurasi tertinggi sebesar 85,20% sedangkan metode SVM mencapai akurasi 83,54% [8].

Meskipun berbagai algoritma tunggal telah menunjukkan hasil yang cukup baik, namun masih terdapat ruang untuk peningkatan performa klasifikasi, algoritma tunggal cenderung mengalami overfitting ketika terlalu mempelajari karakteristik khusus dari data training, yang menyebabkan performa buruk pada data yang belum pernah dilihat sebelumnya [9]. Untuk mengatasi keterbatasan tersebut dan meningkatkan performa kinerja klasifikasi, diperlukan pendekatan yang dapat mengoptimalkan kinerja klasifikasi dengan menggabungkan kekuatan dari berbagai algoritma [9].

Stacking adalah teknik ensemble learning yang kuat yang mengoptimalkan keuntungan dari berbagai algoritma pembelajaran sambil meminimalkan kelemahan individu mereka [10]. Stacking Ensemble menggunakan meta-learner yang secara optimal menggabungkan prediksi dari beberapa base learner yang masing-masing menangkap pola dan hubungan yang berbeda dalam data [11]. Dengan menggabungkan model dasar dengan karakteristik berbeda, stacking dapat menangkap pola data yang lebih luas, yang mungkin tidak dapat dikenali oleh satu model saja [10]. Cara kerja metode Stacking Ensemble adalah dengan melatih beberapa model dasar (base learners) pada data training, kemudian prediksi dari model-model tersebut digunakan sebagai input untuk melatih model tingkat kedua (meta-learner) yang akan menghasilkan prediksi akhir, sehingga dapat mengurangi variance model, meningkatkan ketahanan, dan mencapai akurasi prediktif yang lebih baik daripada model tunggal[11].

Beberapa penelitian telah membuktikan keunggulan metode Stacking Ensemble dalam meningkatkan akurasi klasifikasi. Alfredo Daza dkk. dengan judul "Stacking ensemble approach to diagnosing the disease of diabetes" untuk melakukan prediksi diabetes menggunakan teknik penggabungan metode. Penelitian ini menggunakan Ensemble Stacking dengan beberapa algoritma seperti Decision Tree, Gradient Boosting, K-Nearest Neighbors, Random Forest, Logistic Regression, Support Vector Machines, dan Bayesian Networks. Hasil penelitian ini menunjukkan bahwa penggabungan metode menggunakan konsep Stacking Ensemble menghasilkan tingkat akurasi paling tinggi dibandingkan algoritma tunggal dengan nilai akurasi sebesar 91,5%, sensitivitas 91,6%, F1-score 91,49%, dan presisi 91,5% [10]. Metode Ensemble Stacking juga digunakan pada studi kasus

yang berbeda oleh Agusviyanda dkk. dengan judul "Stacking Ensemble Machine Learning Model for Early Detection of Chronic Kidney Disease in Indonesia" menggunakan Adaboost, XGBoost, LightGBM sebagai base model dan Logistic Regression sebagai meta model. Hasilnya adalah Ensemble Stacking memiliki akurasi yang tinggi dengan angka hasil evaluasi yang didapatkan untuk akurasi sebesar 99,40% [12]. Penelitian lainnya dilakukan oleh Isteaq Kabir Sifat dan Md. Kaderi Kibria dengan judul "Optimizing hypertension prediction using ensemble learning approaches" menggunakan dataset hipertensi dari studi potong lintang yang dilakukan di Hawassa City, Ethiopia dengan 612 peserta dan 27 fitur. Penelitian ini menggunakan Ensemble Stacking dengan kombinasi algoritma LR, ANN, RF, LGBM, dan XGB dengan Logistic Regression sebagai meta-learner. Hasil penelitian menunjukkan bahwa model stacking ensemble mencapai akurasi sebesar 96,32%, presisi 95,48%, recall 97,51%, F1-score 96,48%, dan AUC 0,971, yang menunjukkan performa lebih baik dibandingkan dengan lima model pembelajaran mesin individual lainnya [13]

Untuk melakukan penelitian ini, data yang digunakan adalah data rekam medis pasien Diabetes Melitus. Diabetes adalah kondisi kronis ketika tubuh tidak mampu memproduksi insulin yang memadai atau tidak dapat memanfaatkan hormon insulin secara efektif yang berfungsi membawa glukosa ke dalam sel-sel tubuh dan memungkinkan glukosa masuk serta menjadi sumber energi [14]. Diabetes mellitus yang umumnya disebut sebagai "diabetes" merupakan penyakit kronis yang berkaitan dengan tingginya kadar glukosa (gula) dalam darah[14] .Pemilihan data diabetes melitus sebagai objek penelitian didasarkan pada karakteristik dataset yang memiliki fitur-fitur medis hasil uji laboratorium dengan nilai diskret yang dapat dikategorikan. Dataset diabetes melitus memiliki berbagai atribut seperti kadar glukosa darah, tekanan darah, indeks massa tubuh, dan faktor-faktor lainnya yang secara langsung mempengaruhi kondisi kesehatan pasien. Fitur-fitur tersebut memiliki hubungan yang kuat dengan label atau kelas target, yaitu apakah pasien terkena diabetes atau tidak, sehingga sangat cocok untuk dianalisis menggunakan metode klasifikasi[6].

Berdasarkan uraian di atas, penelitian ini akan mengimplementasikan metode Stacking Ensemble pada studi kasus klasifikasi penyakit diabetes melitus.

Adapun tujuan utama penelitian ini adalah mengeksplorasi dan menemukan arsitektur Stacking Ensemble yang paling unggul dengan cara mencari kombinasi yang optimal antara *base learners* dan *meta-learner*. Pendekatan yang digunakan adalah dengan menguji secara bergantian beberapa algoritma yang telah digunakan pada penelitian sebelumnya, seperti seperti K-Nearest Neighbor (KNN) [6], Logistic Regression (LR) [7], Artificial Neural Network (ANN) [8] , Support Vector Machine (SVM) [8] sebagai *meta-learner*.

1.2 Perumusan Masalah

1.2.1. Permasalahan

Algoritma machine learning tunggal dalam klasifikasi diabetes melitus memiliki keterbatasan dalam hal akurasi. Setiap algoritma tunggal rentan terhadap overfitting, serta memiliki kecenderungan terhadap pola data tertentu. Hal ini menyebabkan performa klasifikasi yang tidak optimal dan tidak konsisten ketika diterapkan pada data yang berbeda. Selain itu, algoritma tunggal tidak memiliki cara untuk menutupi kelemahan yang dimiliki oleh masing-masing metode. Oleh karena itu, diperlukan pendekatan yang dapat mengatasi keterbatasan algoritma tunggal untuk meningkatkan performa metrik evaluasi dalam klasifikasi diabetes melitus.

1.2.2. Metode Usulan

Metode yang diusulkan untuk penelitian ini adalah Stacking Ensemble dikarenakan algoritma ini memiliki kinerja yang dapat diunggulkan untuk melakukan klasifikasi pada beberapa studi kasus penyakit seperti diabetes. Stacking Ensemble terdiri dari beberapa metode yang digabungkan untuk melakukan klasifikasi penyakit diabetes melitus. Penelitian ini akan menggunakan Stacking Ensemble dengan beberapa algoritma antara lain: K-Nearest Neighbor, Logistic Regresion, Arificial Neural Network dan Support Vector Machine. Klasifikasi penyakit diabetes melitus ini berdasarkan dari hasil rekam medis pasien penyakit diabetes melitus.

1.2.3. Pertanyaan Penelitian

Berdasarkan latar belakang yang telah diuraikan, pertanyaan penelitian yang akan dijawab dalam penelitian ini adalah bagaimana perbandingan performa metrik

evaluasi metode *Stacking Ensemble* dengan beberapa metode tunggal yang digunakan pada klasifikasi penyakit diabetes melitus berdasarkan data rekam medis pasien.

1.3 Tujuan dan Manfaat

1.3.1. Tujuan Penelitian

Tujuan penelitian ini adalah untuk melakukan analisis terhadap perbandingan hasil metrik evaluasi dari stacking ensemble dengan metode tunggal pada studi kasus klasifikasi penyakit diabetes melitus.

1.3.2. Manfaat Penelitian

Hasil penelitian ini diharapkan dapat memberikan kontribusi berupa pemahaman mengenai penerapan metode Stacking Ensemble untuk klasifikasi penyakit diabetes melitus, sekaligus menjadi landasan bagi penelitian selanjutnya di bidang yang sama.

1.4 Batasan Masalah

Dalam mempertimbangkan lingkup dam tujuan penelitian, terdapat sejumlah pertimbangan yang membatasi, antara lain:

1. Penelitian ini menggunakan data rekam medis pasien diabetes yang bersumber dari Puskesmas Pulo Lor, Jombang, dengan total data sebanyak 367 baris (belum fix)
2. Fitur yang digunakan didalam dataset untuk penelitian ini adalah umur, jenis kelamin, IMT, LP, Sistolik, Diastolik, Hba1c, GDP, GD2PP.
3. Teradapat 2 kelas yaitu Diabetes Melitus dan Non-Diabetes Melitus
4. Model *Stacking Ensemble* dibatasi pada empat algoritma, yaitu K-Nearest Neighbor (KNN), Logistic Regression (LR), Artificial Neural Network (ANN), dan Support Vector Machine (SVM). Keempat algoritma ini akan digunakan sebagai *base learner* dan diuji secara bergantian sebagai *meta-learner*.

1.5 Sistematika Penulisan

Sistematika penulisan proposal dalam penelitian ini disusun sebagai berikut.

BAB I PENDAHULUAN

Bab ini membahas latar belakang masalah yang meliputi ruang lingkup masalah, solusi yang diusulkan untuk mengatasi permasalahan, tujuan dan manfaat dari penerapan solusi tersebut, batasan – batasan, serta penyusunan proposal yang sistematis.

BAB II TINJAUAN PUSTAKA

Teori – teori dasar yang digunakan dalam penelitian tugas akhir ini disajikan dalam bab ini yang meliputi Diabetes, klasifikasi, *preprocessing*, KNN, Logistic Regression, SVM, ANN, Stacking Ensemble, *Grid Search*, *K-Fold Cross Validation*, evaluasi model, serta penelitian terkait.

BAB III METODOLOGI PENELITIAN

Pada bab ini secara mendalam mengulas tentang dataset yang digunakan sekaligus arsitektur sistem dan tahapan yang akan diterapkan dalam penelitian.

BAB IV HASIL DAN PEMBAHASAN

Pada bab ini berisi pembahasan atau penjelasan yang lebih rinci mengenai implementasi model klasifikasi penyakit diabetes melitus berdasarkan data rekam medis pasien menggunakan metode tunggal dan stacking ensemble yang telah diusulkan pada bab sebelumnya dan hasil pengujian dari model yang dihasilkan. Pengujian dilakukan untuk mengukur seberapa tepat model dalam menghasilkan *output* dengan skenario uji yang diusulkan.

BAB V PENUTUP

Bab ini menyajikan rangkuman atau kesimpulan dari implementasi yang telah dilakukan, serta saran rekomendasi agar penelitian ini bisa dikembangkan pada penelitian mendatang.

BAB II
TINJAUAN PUSTAKA

2.1 Data Mining

Data mining merupakan sebuah pendekatan untuk menggali pengetahuan berharga berupa pola-pola informatif yang tersembunyi dari kumpulan data berskala besar. Tujuan utamanya adalah untuk mengekstraksi nilai tambah dari data yang sebelumnya tidak diketahui secara manual. Informasi yang digali dapat berupa hubungan antar variabel, pola-pola signifikan, hingga pembentukan model yang dapat digunakan untuk tujuan prediksi, seperti melakukan klasifikasi suatu penyakit berdasarkan atribut-atribut yang ada. Dengan kata lain, data mining adalah proses analisis data secara otomatis atau semi-otomatis untuk menemukan pola dan aturan yang bermakna. Penting untuk dipahami bahwa data mining bukanlah proses tunggal, melainkan sebuah tahap inti dalam serangkaian proses yang lebih luas yang dikenal sebagai *Knowledge Discovery in Databases* (KDD). Proses KDD berkaitan dengan keseluruhan metodologi untuk mengubah data mentah menjadi pengetahuan yang dapat ditindaklanjuti [15]. Dalam proses KDD basis data yang bervolume besar dilibatkan dengan tahapan seleksi data, preprocessing data, modeling, dan evaluasi pola yang telah dihasilkan [4].

2.2 Machine Learning

Machine learning merupakan subset dari artificial intelligence yang memungkinkan komputer untuk belajar dan membuat keputusan tanpa perlu diprogram secara eksplisit untuk setiap tugas spesifik [16]. Machine learning terbagi menjadi beberapa paradigma utama, yaitu supervised learning, unsupervised learning, dan reinforcement learning[16]. Supervised learning menjadi paradigma yang paling umum digunakan dalam aplikasi praktis karena kemampuannya untuk memprediksi output berdasarkan data berlabel[17].

2.3 *Supervised Learning*

Supervised learning adalah teknik dalam Machine Learning (ML) yang melibatkan penggunaan data yang memiliki label melatih sebuah model. Algoritma pada *supervised learning* akan belajar dari hubungan data *input* dengan *output* menggunakan data yang berisikan pasangan antara *input-output* yang sudah

diketahui [18]. Secara umum terdapat langkah-langkah utama yang terdapat pada proses pembelajaran model *supervised learning* yaitu proses *training* dan *testing*. Klasifikasi dan regresi adalah dua metode utama dalam *supervised learning* yang digunakan untuk memprediksi *output* berdasarkan data *input*[18]

2.4 *Min-max Normalization*

Min-Max Normalization merupakan salah satu teknik preprocessing data yang paling umum digunakan dalam machine learning untuk mengubah skala data ke dalam rentang tertentu, biasanya antara 0 dan 1 [19]. Teknik ini juga dikenal sebagai feature scaling atau min-max scaling yang berfungsi untuk menormalkan rentang variabel independen atau fitur dari data [20]. Normalisasi data merupakan tahapan penting dalam preprocessing karena rentang nilai dari data mentah yang bervariasi secara luas dapat menyebabkan fungsi objektif dalam beberapa algoritma machine learning tidak bekerja dengan baik tanpa normalisasi [19].

Min-Max Normalization bekerja dengan melakukan transformasi linear pada data asli untuk mengkonversi nilai-nilai floating-point dari rentang alami mereka ke dalam rentang standar [20]. Proses ini memastikan bahwa semua fitur memiliki skala yang sama, sehingga tidak ada fitur tunggal yang mendominasi proses pembelajaran karena memiliki rentang nilai yang lebih besar. Hal ini sangat penting terutama pada algoritma yang menghitung jarak antar titik seperti Euclidean distance, karena jika salah satu fitur memiliki rentang nilai yang luas, maka jarak akan dipengaruhi secara dominan oleh fitur tersebut[21].

$$x'_{i,n} = \frac{x_{i,n} - \min(x_i)}{\max(x_i) - \min(x_i)}(nMax - nMin) + nMin \qquad (2.1)$$

Keterangan

$x'_{i,n}$	:	nilai yang telah dinormalisasi untuk fitur ke-i, data ke-n
$x_{i,n}$	:	nilai asli untuk fitur ke-i, data ke-n yang akan dinormalisasi
$\min(x_i)$	:	nilai minimum dari fitur ke-i dalam dataset
$\max(x_i)$	:	nilai maksimum dari fitur ke-i dalam dataset
nMin	:	batas bawah rentang normalisasi yang diinginkan
nMax	:	batas atas rentang normalisasi yang diinginkan

2.5 *Synthetic Minority Oversampling Technique* (SMOTE)

Ketidakseimbangan data terjadi ketika satu kelas memiliki jumlah sampel yang jauh lebih banyak dibandingkan kelas lainnya, sehingga model cenderung kurang efektif dalam mengenali pola dari kelas minoritas [40]. Salah satu metode yang dapat digunakan untuk mengatasi masalah ini adalah Synthetic Minority Oversampling Technique (SMOTE). Teknik ini bekerja dengan menghasilkan sampel sintetis baru untuk kelas minoritas dengan menambahkan data berdasarkan pola dari sampel yang sudah ada [41].

Algoritma SMOTE bekerja dengan menghitung perbedaan antara suatu sampel kelas minoritas dan tetangga terdekatnya [42], lalu mengalikannya dengan bilangan acak antara 0 hingga 1 [43]. Hasilnya digunakan untuk membentuk vektor baru yang memiliki karakteristik serupa dengan data asli [42]. Dengan cara ini, SMOTE membantu menyeimbangkan jumlah data di setiap kelas tanpa sekadar menduplikasi data yang sudah ada [43]. Persamaan dari SMOTE ditunjukkan sebagai persamaan (17) berikut.

$$x_{\text{new}} = x_i + (\widehat{x_\iota} - x_i) * \delta \tag{2.1}$$

Keterangan

x_{new}	:	nilai vektor baru
x_i	:	vektor dari fitur pada kelas minoritas
$\widehat{x_\iota}$	:	*nearest neighbors* untuk x_i
δ	:	angka acak antara 0 hingga 1

2.6 Klasikasi

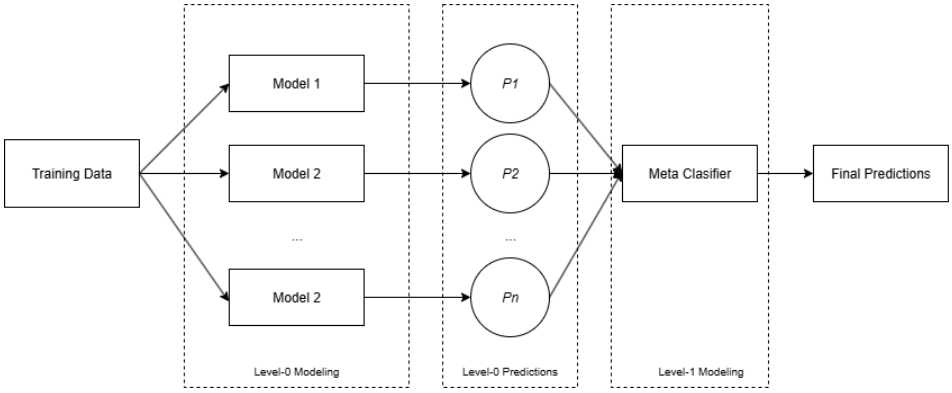
Klasifikasi merupakan suatu metode untuk mengembangkan sebuah model yang memiliki kemampuan dalam mengenali pola-pola pembeda antar konsep atau kelas di dalam data [4]. Dalam prosesnya, klasifikasi secara aktif mencari dan memperhatikan kombinasi fitur-fitur yang paling relevan untuk dapat memisahkan data sesuai dengan kelas target yang telah ditetapkan [4]. Keandalan sebuah model klasifikasi diukur berdasarkan tingkat akurasinya; nilai akurasi yang tinggi mengindikasikan bahwa hasil klasifikasi dapat dipercaya, sedangkan akurasi yang rendah menunjukkan bahwa hasilnya patut dipertanyakan [3]. Secara praktis,

klasifikasi adalah tugas mengevaluasi suatu objek data untuk kemudian menempatkannya ke dalam salah satu dari kelas-kelas yang telah tersedia [3]. Proses ini dapat juga dianalogikan sebagai kegiatan melatih suatu fungsi yang memetakan sekumpulan atribut ke sebuah label kelas yang telah didefinisikan sebelumnya [22].

2.7 *Stacking*

Stacking (*Stacked Generalization*) merupakan konsep penggabungan metode yang termasuk ke dalam bagian *ensemble learning*. Penggabungan metode ini dilakukan untuk meningkatkan tugas prediksi atau klasifikasi secara keseluruhan. Metode ini melibatkan penggunaan model lain untuk memprediksi kinerja model utama, sehingga dapat mengatasi masalah *overfitting* dan meningkatkan *robustness*. Konsep penggabungan metode dengan *stacking* tidak hanya mengandalkan satu model, tetapi memanfaatkan kekuatan dari berbagai model untuk menghasilkan prediksi yang lebih akurat. Konsep utama dari *stacking* adalah bahwa setiap model dasar memiliki kelebihan dan kekurangan yang berbeda. Dengan melakukan kombinasi prediksi dari berbagai model dasar, *meta-model* atau model lanjutan dapat belajar untuk menilai dan memanfaatkan kelebihan serta kekurangan tersebut dengan tepat [23].

Selain itu, *stacking* memungkinkan model lanjutan untuk mengoptimalkan hasil prediksi dengan mempertimbangkan kontribusi masing-masing model dasar. Dengan cara ini, pendekatan ini tidak hanya meningkatkan akurasi, tetapi juga dapat mengurangi *overfitting*, karena *meta-model* belajar dari kombinasi yang lebih beragam[24]. Berikut merupakan gambar konsep penggabungan metode dengan *stacking* pada Gambar



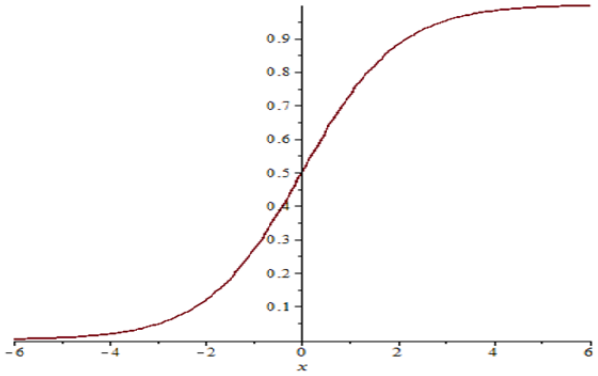
Gambar 2.1 Konsep Penggabungan Metode dengan *Stacking*

Model stacking terdiri dari dua lapisan utama yang bekerja sama untuk meningkatkan akurasi prediksi. Bagian lapisan dasar terdiri dari beberapa model berbeda yang dilatih secara independen pada *dataset* yang sama, sehingga setiap model dapat menangkap pola unik dalam data. Setelah model-model dasar menghasilkan prediksi atau *output* yang akan digunakan sebagai *input* untuk lapisan meta. Di lapisan ini *meta-model* dilatih untuk menggabungkan prediksi dari model dasar dengan tujuan mempelajari hubungan antara prediksi tersebut dan label sebenarnya. Dengan cara ini *meta-model* dapat menimbang kontribusi masing-masing model dasar dan memperbaiki kesalahan yang mungkin terjadi, sehingga menghasilkan kinerja keseluruhan yang lebih baik daripada model individu [25].

2.8 *Logistic Regression* (*LR*)

Logistic Regression (LR) termasuk ke dalam model linier yang digunakan untuk pendekatan klasifikasi dibandingkan dengan regresi [26]. Metode ini digunakan ketika variabel target (y) memiliki skala dalam bentuk kategorik atau biner (terdiri dari 2 kategori) atau lebih. Metode LR berdasarkan fungsi logistik dengan berdasarkan pada persamaan (5), sehingga apapun perkiraan yang terjadi selalu berada pada nilai 0<y<1 atau selalu berada pada nilai 0 hingga 1[27], [28].

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$



Gambar 2.2 Fungsi Logistik

Berikut merupakan langkah-langkah dan rumus persamaan yang digunakan pada metode *Logistic Regression* antara lain [28]:

1. Menentukan bobot awal secara acak untuk membentuk model awal dari LR, jumlah bobot yang digunakan harus sama jumlahnya dengan jumlah fitur data yang digunakan.

$$f(x) = \frac{1}{1 + \exp^{-(w_0 + w_1 x_1 + \cdots + w_d x_d)}} \tag{2.3}$$

Keterangan

$f(x)$: Prediksi dari data input ke x

x_d : Input data dengan dimensi d yaitu $\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \ldots, \mathrm{x}_\mathrm{d}$

w_d : Bobot atau *coefficient* dari masing-masing input data dengan dimensi d yaitu $\mathrm{w}_1, \mathrm{w}_2, \mathrm{w}_3, \ldots, \mathrm{w}_\mathrm{d}$

w_0 : *Intercept* atau bias

2. Melakukan beberapa perhitungan nilai pada setiap *epoch* dan interasi, nilai yang dihitung antara lain: Menghitung hasil prediksi dari setiap data pelatihan berdasarkan model yang dibentuk menggunakan persamaan 6. Menghitung selisih *error* dan *update* bobot menggunakan persamaan 7

$$w_j = w_j + n\left(y^i - f\left(x^i\right)\right) f\left(x^i\right)\left(1 - f\left(x^i\right)\right) x_j^i \tag{2.4}$$

Keterangan

w_j : Bobot ke j, dimana $\mathrm{j} = 1, 2, 3, \ldots \mathrm{d}$ dimana d adalah jumlah input data

y^i : Label sebenarnya dari data ke-i

$f(x^i)$: Prediksi dari data ke-i

x_j^i : Input data ke-j dari data pelatihan ke-i

n : *Learning rate*

2.9 *K-Nearest Neighbor* (*KNN*)

K-Nearest Neighbors (KNN) adalah metode ML yang digunakan dalam klasifikasi dan regresi. KNN merupakan algoritma yang berdasarkan prinsip bahwa objek-objek yang serupa cenderung berada dalam jarak yang dekat di dalam ruang fitur. Dalam klasifikasi, KNN mengidentifikasi k tetangga terdekat dari objek yang akan diprediksi dan objek tersebut akan diklasifikasikan berdasarkan mayoritas kelas dari tetangga-tetangganya. Proses perhitungan dalam algoritma KNN menggunakan *Euclidean distance* yang merupakan metode untuk menentukan seberapa dekat dua titik variabel. Semakin dekat dan serupa kedua titik tersebut, semakin kecil jarak yang diukur di antara keduanya [29]. Metode KNN ini mempunyai langkah-langkah dalam penerapannya antara lain:

1. Menentukan nilai K
2. Mengukur jarak antara data latih dan data uji merupakan langkah penting dalam proses ini. Salah satu metode yang umum digunakan adalah menghitung jarak menggunakan rumus *Euclidean distance*, dengan menggunakan formula yang dinyatakan pada persamaan dibawah

$$d(\mathrm{p},\mathrm{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2.5}$$

Keterangan

$d(\mathrm{p},\mathrm{q})$: Jarak Euclidean

n : Jumlah dimensi atau fitur dari data

p_i : Nilai dari fitur ke-i

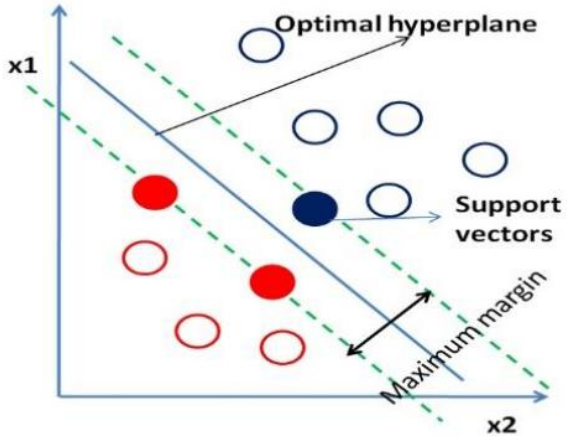
q_i : Nilai dari fitur ke-i

3. Melakukan pengurutan dari jarak yang telah terbentuk
4. Menentukan jarak terdekat sampai dengan urutan nilai K
5. Menyandingkan kelas yang sesuai
6. Menghitung jumlah kelas dari tetangga terdekat dan menetapkan kelas tersebut

sebagai kelas yang akan di evaluasi untuk data tersebut.

2.10 *Support Vector Machine (SVM)*

Support Vector Machine (SVM) merupakan algoritma yang tergolong ke kategori supervised learning. SVM bekerja dengan cara mencari pemisah (garis, bidang atau hyperplane) yang optimal untuk melakukan pemisah pada dua kelas data atau lebih. Hyperplane adalah pemisah terbaik antara kedua kelas atau lebih pada data yang ditentukan dari hasil pengukuran pada titik terdekat masing- masing kelas. Jika data berada paling dekat dengan hyperplane terbaik maka disebut sebagai support vector [30]. SVM dapat melakukan klasifikasi data yang terpisah baik secara linier (lenearly separable) dan non-linier (nonlinier separable). Hyperplane SVM dapat dilihat pada Gambar 2. *3* dibawah ini [31].



Gambar 2. 3 Hyperplane Support Vector Machine (SVM)

Proses pembelajaran pada SVM akan bergantung pada dot product dari data yang telah dilakukan transformasi ke ruang baru yang memiliki dimensi lebih tinggi. Bentuk transformasi data ke ruang baru ini akan sangat sulit dipahami secara mudah, perhitungan dot product ini dapat digantikan dengan fungsi kernel pada SVM yang dapat mendefinisikan secara implisit bentuk transformasi setiap dot product [32]. Proses ini pada SVM disebut sebagai kernel trick, terdapat beberapa fungsi kernel yang dapat digunakan Tabel 2. *1* dibawah ini.

Tabel 2. 1 Fungsi Kernel pada SVM

Jenis Kernel	Fungsi Persamaan
Linear	$K(x_i, x) = x_i^T x$
Polynomial	$K(x_i, x) = (\gamma x_i^T x + r)^p, \gamma > 0$

Radial Basis Function (RBF)	$K(x_i, x) = \exp(-\gamma\|x - x_i\|^2)$

Berikut merupakan langkah-langkah dan persamaan yang digunakan pada metode SVM antara lain [32]:

1. Menentukan nilai awal untuk α, C, epsilon, gamma dan lambda
2. Memasukkan data *training* yang akan digunakan
3. Menghitung dot product untuk setiap data dengan cara menggunakan fungsi kernel
4. Menghitung nilai matriks hessian menggunakan persamaan 2.6

$$D_{ij} = Y_i Y_j \left(K(X_i, X_j) + \lambda^2\right) \quad (2.6)$$

Keterangan

D_ij : Elemen matriks hessian ke-ij

Y_i : Kelas data ke-i

Y_j : Kelas data ke-j

λ : Lamda sebagai parameter regulasi

5. Menghitung nilai *error* dengan persamaan 2.7

$$E_i = \sum_j^n a_j D_{ij} \quad (2.7)$$

Keterangan

E_i : Nilai *error* data ke-i

a_j : Nilai alpha ke-j

D_ij : Nilai dari matriks hessian

6. Menghitung nilai $\delta\alpha_i$ dengan persamaan 2.8

$$\delta\alpha_i = \min[\max[y(1 - E_i) - \alpha_i], C - \alpha_i] \quad (2.7)$$

Keterangan

α_i : Nilai alpha ke-i

y : Nilai dari gamma yang telah ditentukan

E_i : Rata-rata nilai *error*

C Batas Nilai alpha

7. Menghitung nilai alpha baru dengan persamaan 2.8

$$\alpha_i = \alpha_i + \delta\alpha_i \tag{2.7}$$

Keterangan

α_i : Nilai alpha ke-i

$\delta\alpha_i$: Nilai dari gamma yang telah ditentukan

E_i : Nilai delta alfa ke-i

8. Mengulangi langkah 4 hingga mencapai iterasi maksimum atau hingga $\max(|\delta\alpha_i|) < \varepsilon$, yakni ketika konvergensi tercapai atau perubahan sudah tidak signifikan
9. Setelah memperoleh nilai support vector, langkah selanjutnya menghitung nilai bias dengan persamaaan 2.8

$$b = -\frac{1}{2}\left(\sum_{i=0}^{n} \alpha_i y_i K(x_i, x^-) + \sum_{i=0}^{n} \alpha_i y_i K(x_i, x^+)\right) \tag{2.8}$$

10. Menghitung nilai fungsi f(x) menggunakan persamaan dibawha ini

$$f(x) = \sum_{i=0}^{n} \alpha_i y_i K(x_i, x) + b \tag{2.9}$$

Keterangan

α_i: : Nilai alpha ke-i

y_i : Nilai data dari kelas ke-i

m : Jumlah data dari *support vector*

$K(x, x_i)$: Fungsi kernel yang digunakan

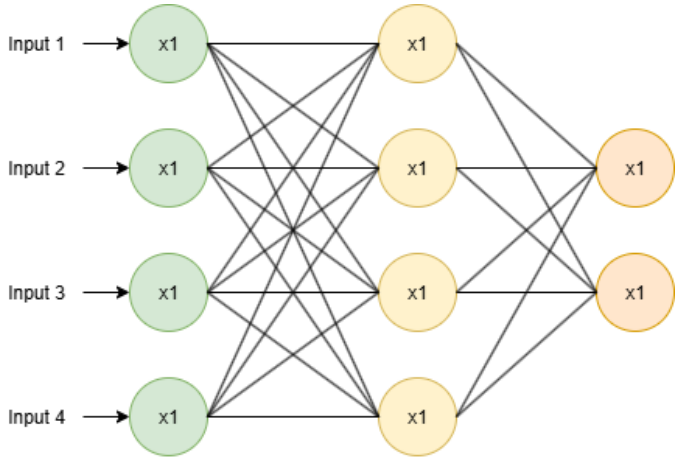
b : Nilai bias

11. Menentukan kelas. Jika f(x) > 0, maka akan diberi label +1, dan jika f(x) 0 maka akan diberi label -l

2.11 *Artificial Neural Network* (*ANN*)

Artificial Neural Network (ANN) adalah sebuah metode dalam kecerdasan buatan yang meniru atau terinspirasi dari mekanisme kerja otak manusia dalam memproses informasi. ANN memiliki struktur yang terbentuk dari unit atau *neuron* buatan yang saling terhubung satu sama lain. *Neuron* yang saling terhubung ini

mirip dengan struktur jaringan *neuron* dalam otak. Masing-masing *neuron* memiliki bobot yang berbeda kemudian informasi ANN ini dilatih dengan menggunakan data dan algoritma tertentu untuk menyesuaikan bobot-bobot tersebut sehingga mampu membuat prediksi atau klasifikasi berdasarkan pola yang ada dalam data proses menggunakan fungsi aktivasi yang digunakan [33]. ANN memiliki beberapa arsitektur jaringan yang dikategorikan secara umum menjadi 3 mulai dari *single layer network*, *multi-layer network* dan *competitive network* [34]. Berikut merupakan arsitektur jaringan *multi layer network* pada Gambar 2.3[33].



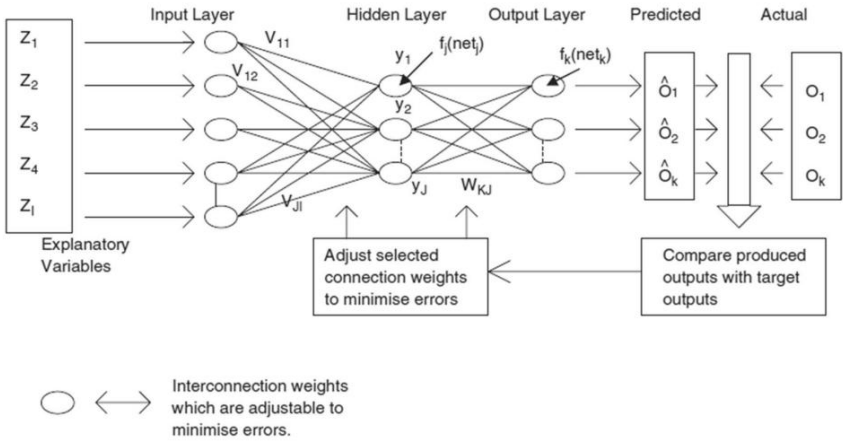
Gambar 2.4 Arsitektur Multi Layer Network

Multi-layer network terdiri dari beberapa *layer neuron* yang terhubung dari *layer* lainnya yaitu *input layer*, *hidden layer* dan *output layer*[35]. Konsep dasar pada arsitektur *multi-layer network* adalah dengan menghubungkan neuron-neuron dalam 3 lapisan utama. Setiap *neuron* dalam lapisan tersembunyi berfungsi untuk mengekstrak fitur dari *input*, sementara lapisan *output* menghasilkan prediksi akhir berdasarkan representasi yang dibangun oleh *layer* sebelumnya [36]. Selama proses pelatihan *multi-layer network* berlangsung akan melibatkan algoritma *Backpropagation*. Algoritma ini digunakan untuk *multi-layer network* agar dapat memperbarui bobot setiap koneksi yang dimiliki oleh *neuron* berdasarkan dari kesalahan yang dihitung pada *output*, kemudian melalui proses propagasi mundur bobot akan diperbarui[37] .

2.11.1. Arsitektur Backpropagation

Backpropagation merupakan salah satu algoritma yang termasuk ke dalam *neural network*. Algoritma *backpropagation* menghubungkan setiap *neuron* yang

terdapat pada masing-masing *layer* untuk melakukan perhitungan bobot untuk menghasilkan sebuah *output*. Algoritma yang paling banyak digunakan dalam pembelajaran jaringan saraf adalah *Backpropagation* karena kemampuannya dalam mengurangi tingkat kesalahan dengan menyesuaikan bobot berdasarkan perbedaan antara keluaran dan target yang diinginkan [38]. Berikut merupakan gambar dari arsitektur algoritma *Backpropagation* pada Gambar 2.4 [39].



Gambar 2.5 Arsitektur Algoritma Backpropagation

Backpropagation terdiri dari tiga tahap dalam proses pelatihannya: fase maju (forward propagation), fase mundur (backward propagation), dan fase modifikasi bobot. Pada fase maju pola masukan dihitung dari input hingga mencapai output. Selanjutnya pada fase mundur, setiap unit output menerima nilai kesalahan untuk dihitung dan pada fase modifikasi bobot kesalahan tersebut dioptimalkan. Proses ini diulang hingga kondisi penghentian terpenuhi [40]. Berikut merupakan tahapan dalam pelatihan Backpropagation antara lain [33]:

1. Menentukan nilai learning rate, epoch, dan nilai toleransi error. Proses pembelajaran akan berhenti jika mencapai epoch maksimum dan memenuhi toleransi error yang telah ditentukan.
2. Menentukan bobot secara acak dengan nilai yang kecil.
3. Jika kondisi pemberhentian belum terpenuhi maka ulangi langkah 4-11.

Fase I: Forward Propagation

4. Setiap neuron masukan $(X_1, X_2, \dots, X_i)$ dimana i adalah jumlah neuron masukan. Setiap neuron masukan mendapat bobot v_{ij}dan menuju ke setiap neuron lapisan tersembunyi.

5. Setiap neuron tersembunyi $(Z_1, Z_2, \dots, Z_j)$ di mana j adalah jumlah neuron tersembunyi. Menjumlahkan bobot setiap neuron masukan yang menuju ke Setiap neuron lapisan tersembunyi ditunjukkan dengan persamaan di bawah ini.

$$Z_{net_j} = V_{0j} + \sum_{i=1}^{n} X_i V_{ij} \tag{2.11}$$

Keterangan

Z_{net_j} : Hasil perhitungan pada neuron hidden ke-j

$\text{v}_{0\text{j}}$: Bias pada neuron hidden layer ke-j

x_i : Nilai neuron input ke-i

v_ij : Bobot yang menyambungkan antara neuron input ke-i dan neuron hidden layer ke$-j$

6. Hasil Z_{net_j} dikalikan dengan fungsi aktivasi untuk memperoleh *output* dari *hidden layer* kemudian dikirim ke *output layer*.

$$Z_j = f\left(Z_{\text{net}_j}\right) = \frac{1}{1 + e^{-Z_{\text{net}_j}}} \tag{2.12}$$

Keterangan

Z_j : Hasil aktivasi output di neuron hidden ke-j

Z_{net_j} : Fungsi aktivasi pada nilai Z_{net_j}

7. Setiap output layer $(\text{Y}_1, \text{Y}_2, \dots, \text{Y}_\text{k})$ di mana k adalah jumlah neuron output. Menjumlahkan bobot ouput dari hidden layer Z_j

$$Y_{net_k} = W_{0k} + \sum_{j=1}^{p} Z_j W_{jk} \tag{2.13}$$

Keterangan

Y_{net_k} : Hasil perhitungan pada neuron *hidden*

$\text{W}_{0\text{k}}$: Bias pada neuron hidden layer ke-j

Z_j : Nilai neuron input ke-j

W_jk : Bobot yang menyambungkan antara neuron hidden ke-j dan neuron output layer ke$-$k

8. Hasil Y_{ink} dikalikan dengan fungsi aktivasi untuk memperoleh output dari

output layer.

$$Y_k = f(Y_{net_k}) = \frac{1}{1 + e^{-Y_{net_k}}} \quad (2.14)$$

Keterangan

Y_k : Hasil aktivasi output pada neuron output layer

$f(Y_{ink})$: Fungsi aktivasi pada nilai Y_{ink}

Fase II: Backward Propagation

9. Setiap output layer $(Y_1, Y_2, \dots, Y_k)$ di mana k adalah jumlah neuron output. Memperoleh pola dari hasil training input, kemudian melakukan perhitungan nilai error dari ouput layer (Y_k)

$$\delta_k = (t_k - Y_k) f'(Y_{net_k})$$
$$\delta_k = (t_k - Y_k) Y_k (1 - Y_k) \quad (2.15)$$

Keterangan

δ_k : Perbandingan nilai kesalahan antara ouput layer dengan target

t_k : Target output ke-k

Y_{net_k} : Hasil perhitungan output layer

Y_k : Hasil aktivasi output di neuron output layer ke-k

f' : Turunan dari fungsi aktivasi

Kemudian menghitung koreksi bobot antara hidden layer dengan output layer W_{jk}

$$\Delta W_{jk} = \alpha \delta_k Z_j \quad (2.16)$$

Keterangan

ΔW_{jk} : Faktor kesalahan bobot yang menyambungkan neuron hidden layer ke-j dengan neuron output layer ke-k

a : Nilai *learning rate*

δ_k : Nilai Kesalahan dari neuron output layer ke-k

Z_j : Hasil aktivasi output di neuron hidden ke-j

Menghitung koreksi bias (digunakan untuk memperbarui nilai W_{ok}):

$$\Delta W_{0k} = \alpha \delta_k \quad (2.17)$$

Keterangan

ΔW_{0k} : Faktor kesalahan bias pada neuron output layer ke-k

α : Nilai *learning rate*

δ_k : Nilai kesalahan dari neuron output layer ke-k

10. Menghitung nilai error pada unit hidden δ_{net_j}

$$\delta_{net_j} = \sum_{k=1}^{m} \delta_k W_{jk} \tag{2.18}$$

Keterangan

δ_{net_j} : Total kesalahan dari lapisan output yang telah dikalikan dengan bobot koneksinya, untuk neuron ke-j.

δ_k : Nilai kesalahan dari neuron output layer ke-k

W_{jk} : Bobot yang menyambungkan antara neuron hidden ke-j dan neuron ouput layer ke-k

Melakukan perkalian nilai error pada unit hidden menggunakan turunan rumus fungsi aktivasi untuk menghitung nilai error.

$$\begin{aligned} \delta_j &= \delta_{net_j} f'\left(Z_{net_j}\right) \\ \delta_j &= \delta_{net_j} Z_j\left(1 - Z_j\right) \end{aligned} \tag{2.20}$$

Keterangan

δ_j : Selisih nilai kesalahan dari neuron hidden layer ke-j

δ_{net_j} : Total kesalahan dari lapisan output yang telah dikalikan dengan bobot koneksinya, untuk neuron ke-j.

Z_{net_j} : Hasil perhitungan pada neuron hidden ke-j

Z_j : Hasil aktivasi output di neuron hidden layer ke-j

f' : Turunan dari fungsi aktivasi

Kemudian menghitung koreksi bobot antara input layer dengan hidden layer W_{jk}

$$\Delta v_{ij} = \alpha \delta_j x_i \tag{2.21}$$

Keterangan

Δv_{ij} : Nilai faktor kesalahan bobot yang menyambungkan neuron input layer ke-i dengan neuron hidden layer ke-j

α : Nilai learning rate

δ_j : Selisih nilai kesalahan dari neuron hidden layer ke-j

x_i : Nilai neuron input ke-i

Menghitung koreksi bias (digunakan untuk memperbarui nilai v_{oj})

$$\Delta v_{0j} = \alpha \delta_j \quad (2.22)$$

Keterangan

Δv_{0j} : Nilai faktor kesalahan bias pada neuron hidden layer ke-j

α : Nilai learning rate

δ_j : Selisih nilai kesalahan dari neuron hidden layer ke-j

11. Melakukan pembaruan bobot dan bias pada setiap layer. Setiap unit output Y_k memperbarui bias dan bobotnya

$$W_{jk}(\text{baru}) = W_{jk}(\text{lama}) + \Delta w_{jk} \quad (2.23)$$

Keterangan

$W_{jk}(\text{baru})$: Bobot baru yang menyambungkan neuron *hidden layer* dengan *output layer* ke-*j*

$W_{jk}(\text{lama})$: Bobot lama yang menyambungkan neuron *hidden layer* dengan *output layer* ke-k

Δw_{jk} : Nilai faktor kesalahan yang menyambungkan neuron *hidden layer* ke-j dengan neuron *output layer* ke-k

Setiap unit *hidden* Z_j memperbarui bias dan bobotnya.

$$V_{ij}(\text{baru}) = V_{ij}(\text{lama}) + \Delta V_{ij} \quad (2.24)$$

Keterangan

$V_{ij}(\text{baru})$: Bobot baru yang menyambungkan neuron *input layer* dengan *hidden layer* ke-j

$V_{ij}(\text{lama})$: Bobot lama yang menyambungkan neuron *input layer* dengan *hidden layer* ke-j

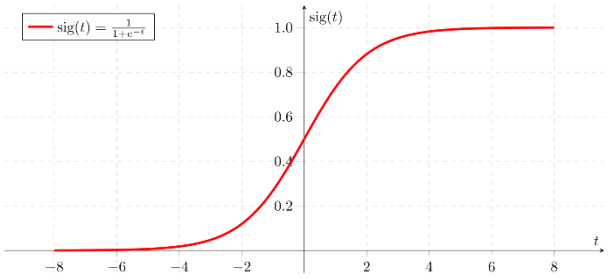
ΔV_{ij} : Nilai fakyang menyambungkan neuron *input layer* ke-i dengan neuron *hidden layer* ke-j

2.11.2. Fungsi Aktivasi

Fungsi aktivasi adalah komponen dalam Jaringan Saraf Tiruan yang digunakan untuk melakukan kalkulasi kompleks di *hidden layer* dan kemudian mentransfer hasilnya ke *output layer*[41]. Fungsi utamanya adalah untuk memperkenalkan properti non-linear pada jaringan saraf, yang memungkinkan model mempelajari pola data yang sangat rumit. Selain itu, pemilihan fungsi aktivasi yang tepat juga berperan penting untuk mempercepat proses pelatihan (*training*) serta meningkatkan akurasi model secara keseluruhan[41]. Fungsi aktivasi *Sigmoid* sering digunakan ketika menggunakan algoritma *Backpropagation* pada proses kasus yang membutuhkan *output* dua kelas yang berbeda [42].

$$y = f(x) = \frac{1}{1 + e^{-x}} \tag{2.25}$$

Fungsi aktivasi *Sigmoid* ini memiliki nilai antara 0 sampai 1. Berikut merupakan gambar yang dapat mempresentasikan grafik dari hasil fungsi aktivasi *Sigmoid* pada Gambar 2.6 [42].



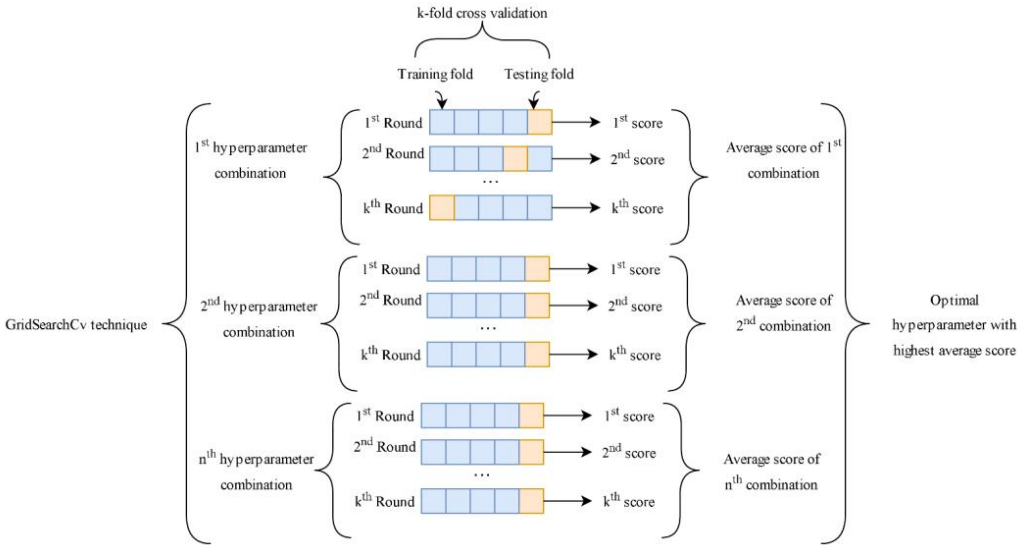
Gambar 2.6 Grafik Fungsi Aktivasi Sigmoid

2.12 *Grid Search CV*

Grid Search Cross-Validation atau GridSearchCV merupakan salah satu metode yang sering digunakan untuk penyetelan hyperparameter. GridSearchCV adalah teknik yang membantu menemukan parameter-parameter yang menghasilkan performa terbaik untuk model tertentu [43]. Metode ini mengadopsi proses *cross-validation* untuk mengevaluasi kinerja setiap kombinasi hiperparameter pada data latih guna menemukan parameter terbaiknya[44]. GridSearch akan mencoba semua kombinasi yang mungkin dari beberapa parameter yang ditetapkan dan memberikan jumlah kombinasi untuk dilakukan pencarian kombinasi terbaik. Dan CV akan melakukan pelatihan sebanyak *Fold*

cross-validation yang ditentukan[44]

Berikut merupakan gambar cara kerja GridSearch mencari kombinasi *hyperparameter* terbaik[44].

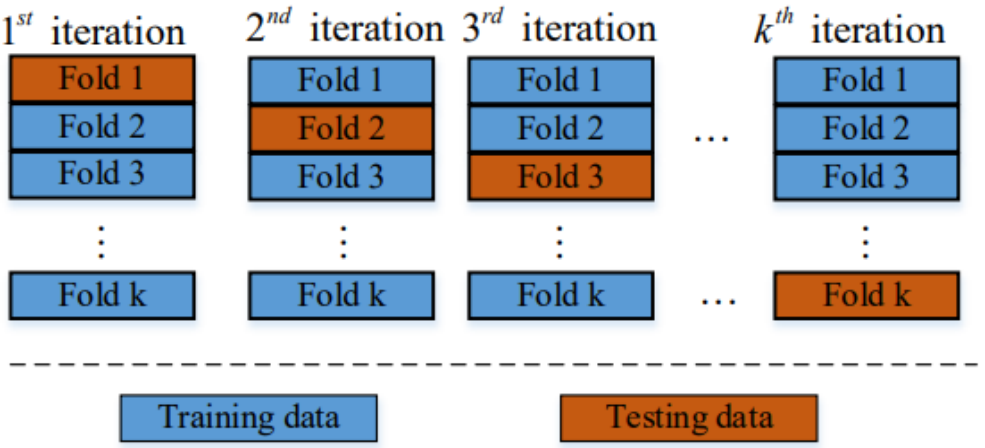


Gambar 2. 7 Proscs pencarjan hyperparameter terbaik GridSeachCV

2.13 Evaluasi

2.13.1. Cross Validation

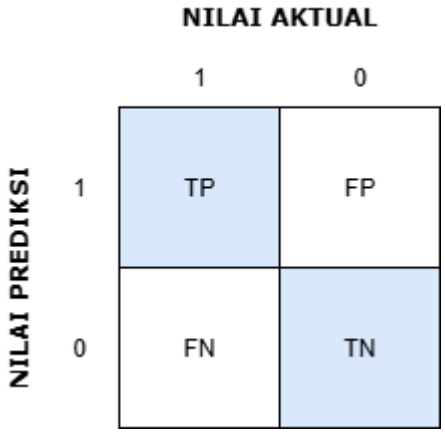
K-Fold Cross Validation merupakan salah satu metode yang umum digunakan untuk menguji kinerja model *machine learning*[45]. Prinsip kerjanya adalah dengan membagi kumpulan data menjadi K bagian yang sama besar. Proses pengujian ini kemudian dijalankan secara berulang sebanyak K kali[45]. Pada setiap putarannya, satu bagian akan difungsikan sebagai data uji, sementara sisa bagian lainnya digunakan sebagai data latih[45]. Nantinya, skor performa akhir dari model diambil dari nilai rata-rata yang diperoleh dari keseluruhan putaran pengujian tersebut[45]. Gambar 2.*8* merupakan ilustrasi dari K-Fold Cross Validation[45].



Gambar 2.8 K-Fold Cross Validation

2.13.2. Confusion Matrix

Confusion matrix adalah sebuah metode untuk mengukur kinerja model klasifikasi dengan cara menyajikan perbandingan antara hasil prediksi model dan nilai aktualnya. Matriks ini merangkum prediksi yang benar, yang terdiri dari True Positives (TP) dan True Negatives (TN), serta prediksi yang tidak tepat, yaitu False Positives (FP) dan False Negatives (FN). Berdasarkan nilai-nilai dalam matriks yang berisi label aktual dan prediksi ini, dapat dihitung berbagai metrik evaluasi lain seperti accuracy, precision, recall, dan F1-Score untuk menilai performa supervised learning [46]. Berikut merupakan bentuk dasar dari *confusion matrix* terlampir pada Gambar 2.9[46].



Gambar 2. 9 Confusion Matrix

Berikut merupakan beberapa keterangan istilah atau singkatan yang terdapat pada gambar 2.9 tentang *confusion matrix* antara lain[46]:

True Positive (TP)	:	Kondisi dimana model dengan benar mengidentifikasi data kelas positif
True Negative (TN)	:	Kondisi dimana model dengan benar mengidentifikasi data kelas negatif
False Positive (FP)	:	Kondisi dimana model salah mengidentifikasi data kelas negatif sebagai kelas positif
False Negative (FN)	:	Kondisi di mana model salah mengidentifikasi data kelas positif sebagai kelas negatif.

Hasil dari *confusion matrix* ini dapat digunakan untuk mengukur evaluasi dari hasil klasifikasi antara lain[46]:

a. Accuracy

Accuracy (akurasi) digunakan untuk mengukur sejauh mana model klasifikasi memprediksi dengan benar seluruh kelas data, baik kelas positif maupun negatif. Untuk menghitung akurasi dinyatakan pada persamaan 2.26.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2.26}$$

b. Precision

Precision (presisi) adalah metrik evaluasi dalam konteks klasifikasi yang mengukur sejauh mana model memprediksi dengan tepat kasus positif. Dalam kata lain, presisi mengukur proporsi dari hasil prediksi positif yang benar dari semua prediksi positif yang dilakukan oleh model. Untuk menghitung presisi dinyatakan pada persamaan 2.27.

$$Precision = \frac{TP}{TP + FP} \tag{2.27}$$

c. Recall

Recall juga dikenal sebagai sensitivitas atau *True Positive Rate (TPR)*, adalah metrik evaluasi dalam konteks klasifikasi yang mengukur sejauh mana model dapat mendeteksi atau menangkap semua kasus positif yang sebenarnya. *Recall* mengukur proporsi dari kasus positif yang diprediksi dengan benar dari seluruh kasus positif yang sebenarnya. Untuk menghitung *recall* dinyatakan pada persamaan 2.28.

$$Recall = \frac{TP}{TP + FN} \tag{2.28}$$

d. F1-Score

F1-Score adalah metrik evaluasi yang menggabungkan antara *precision* dan *recall*. *F1-Score* memberikan suatu nilai tunggal yang mencerminkan keseimbangan antara *precision* dan *recall* sehingga berguna ketika ingin mempertimbangkan kedua aspek ini secara bersamaan. Untuk menghitung *F1-Score* dinyatakan pada persamaan 34.

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.29}$$

2.14 Diabetes Melitus

2.14.1. Pengertian Diabetes Melitus

Diabetes melitus adalah suatu kondisi yang disebabkan oleh tingginya kadar gula dalam darah. Pankreas berfungsi untuk menghasilkan insulin yang berperan menurunkan kadar gula darah, namun pada penderita diabetes, proses ini terganggu. Kadar gula darah yang tidak normal dapat berdampak negatif pada berbagai organ tubuh dan sistem saraf. Kondisi ini tidak hanya memengaruhi kesehatan perorangan, tetapi juga telah menjadi masalah kesehatan global karena dampaknya yang signifikan dan prevalensinya yang terus meningkat [47] Diabetes melitus dapat diklasifikasikan menjadi dua tipe utama, yaitu tipe 1 dan tipe 2. Diabetes tipe 1 disebabkan oleh respons autoimun, di mana sistem kekebalan tubuh merusak sel-sel pankreas yang memproduksi insulin, sehingga mengakibatkan produksi insulin menurun drastis. Sementara itu, diabetes tipe 2 disebabkan oleh kombinasi faktor genetik dan lingkungan. Faktor genetik berkaitan dengan gangguan sekresi insulin dan resistensi insulin. Faktor lingkungan yang dapat memicu kondisi ini antara lain kelebihan berat badan, pola makan tidak sehat, kurangnya aktivitas fisik, stres, dan proses penuaan. Tipe 2 merupakan jenis yang lebih umum ditemukan di masyarakat dan sering dikaitkan dengan pola hidup[48], [49].

2.14.2. Korelasi Umur dengan Diabetes

Umur menjadi salah satu pemicu seseorang menderita penyakit diabetes

yang tidak dapat diubah[50]. Semakin bertambah usia seseorang, semakin besar pula potensi diri terdiagnosa diabetes[50]. Seseorang dengan umur lebih dari 45 tahun memiliki kecenderungan 4x lebih besar menderita kadar gula darah rendah dibandingkan seseorang dengan umur kurang dari 45 tahun sehingga potensi menderita diabetes tipe I lebih besar[50]. Pada diabetes tipe II menunjukan seseorang memiliki potensi hingga 9x lebih besar ketika berusia lebih dari 45 tahun keatas[50]. Keadaan ini disebabkan karena faktor penurunan kemampuan metabolisme glukosa dalam tubuh[50].

2.14.3. Korelasi Jenis Kelamin dengan Diabetes

Penyakit diabetes dapat menyerang siapa saja, baik laki-laki maupun perempuan, namun berdasarkan penelitian, perempuan lebih sering terdiagnosis menderita diabetes dibandingkan laki-laki[51]. Perbedaan risiko ini dijelaskan oleh beberapa faktor hormonal dan komposisi tubuh yang unik pada perempuan[51]. Hormon estrogen dan progesteron yang dimiliki perempuan memiliki kemampuan untuk meningkatkan respons tubuh terhadap insulin[51]. Akan tetapi, saat perempuan memasuki masa menopause, produksi kedua hormon ini menurun drastis, yang menyebabkan respons insulin dalam tubuh ikut melemah[51]. Penurunan hormon estrogen ini juga memicu peningkatan cadangan lemak tubuh, terutama di daerah perut, yang dapat menyebabkan resistensi insulin[51]. Faktor-faktor inilah yang menyebabkan perempuan memiliki risiko lebih tinggi untuk terkena diabetes melitus

2.14.4. Korelasi Indeks Massa Tubuh dengan Diabetes

Obesitas merupakan salah satu faktor risiko paling umum yang dapat memicu terjadinya diabetes melitus[52]. Berdasarkan penelitian, ditemukan kesimpulan bahwa pasien dengan Indeks Massa Tubuh (IMT) atau Body Mass Index (BMI) yang lebih tinggi memiliki kemungkinan lebih besar untuk menderita diabetes[52].Hasil studi menunjukkan bahwa nilai rata-rata BMI pada kelompok penderita diabetes secara signifikan lebih tinggi dibandingkan dengan kelompok non-diabetik[52]. Rata-rata BMI pada pasien diabetes adalah $27.06\ \mathrm{kg/m^2}$, sedangkan pada kelompok sehat adalah $22.47\ \mathrm{kg/m^2}$. Perbedaan ini secara statistik sangat signifikan, yang menegaskan bahwa penderita diabetes cenderung memiliki

BMI yang lebih tinggi dibandingkan individu sehat.

2.14.5. Korelasi Lingkar Pinggang dengan Diabetes

Berdasarkan penelitian, obesitas abdominal yang diukur melalui lingkar pinggang merupakan faktor risiko yang lebih kuat untuk diabetes tipe 2 dibandingkan Indeks Massa Tubuh (IMT)[53]. Pengukuran lingkar pinggang adalah metode yang efektif dan mudah untuk mengidentifikasi individu yang berisiko. Sebuah studi menetapkan bahwa lingkar pinggang ≥94 cm adalah batas ukur paling prediktif untuk risiko tinggi diabetes tipe 2, dengan tingkat sensitivitas 84.4% dan spesifisitas 78.2%[53].

2.14.6. Korelasi Tekanan Darah dengan Diabetes

Terdapat hubungan dua arah yang sangat erat antara hipertensi (tekanan darah tinggi) dan diabetes, yang sering digambarkan sebagai hubungan "ayam dan telur"[54]. Hipertensi tidak hanya lebih umum terjadi pada penderita diabetes, tetapi diabetes juga lebih sering ditemukan pada penderita hipertensi[54]. Diperkirakan sekitar 50% hingga 80% penderita diabetes tipe 2 juga menderita hipertensi[54]. Data epidemiologis secara konsisten menunjukkan bahwa hipertensi merupakan faktor risiko kuat untuk perkembangan diabetes; sebuah studi menemukan bahwa diabetes tipe 2 hampir 2,5 kali lebih mungkin berkembang pada pasien dengan hipertensi dibandingkan dengan mereka yang memiliki tekanan darah normal[54].

2.14.7. Korelasi HbA1c dengan Diabetes

Glycated Haemoglobin (HbA1c) merupakan indikator penting untuk kontrol glikemik jangka panjang karena kemampuannya untuk mencerminkan riwayat glikemik kumulatif selama dua hingga tiga bulan sebelumnya[55]. Jumlah HbA1c yang terbentuk berbanding lurus dengan rata-rata kadar glukosa dalam plasma artinya, semakin tinggi rata-rata glukosa darah, semakin tinggi pula kadar HbA1c[55]. Berdasarkan korelasi yang kuat ini, American Diabetes Association (ADA) merekomendasikan HbA1c dengan nilai ≥6.5% sebagai salah satu kriteria untuk mendiagnosis diabetes[55]. Secara umum, rentang HbA1c untuk non-diabetes adalah 4.0%-5.6%[55].

2.14.8. Korelasi Gula Darah dengan Diabetes

Kadar gula darah merupakan indikator dalam mendiagnosis diabetes.

Tingkat gula darah ini sendiri dipengaruhi oleh berbagai faktor, seperti perubahan berat badan, pola konsumsi makanan, dan aktivitas fisik harian. Kondisi gula darah yang tinggi secara berkelanjutan dapat menjadi pemicu diabetes melitus. Untuk kondisi puasa (GDP), batas diagnosis diabetes adalah ≥126 mg/dL. Sementara itu, untuk tes Gula Darah 2 Jam Postprandial (GD2PP) selama Tes Toleransi Glukosa Oral (TTGO), seseorang didiagnosis menderita diabetes jika kadar glukosa darahnya ≥200 mg/dL.

2.14.9. Penelitian Terkait

Penelitian-penelitian terdahulu menjadi acuan penting yang memberikan wawasan, metode, dan kerangka kerja untuk mendukung studi ini. Kontribusinya membantu dalam perumusan masalah, penyusunan metodologi, dan validasi hasil yang kuat. Oleh karena itu, Tabel 2.2 menyajikan judul dan ringkasan dari studi-studi relevan yang telah dilakukan oleh para ahli sebelumnya.

Tabel 2. 2 Penelitian Terkait

Peneliti, Tahun	Judul	Metode	Hasil Penelitian
Erlin dkk. 2022 [7]	*Early Detection of Diabetes Using Machine Learning with Logistic Regression Algorithm*	*Logistic Regression*	Setelah hyperparameter tuning dengan grid search, model mencapai akurasi 82%, precision 81%, recall 79%, dan F1-score 80%
Tsehay Admassu Assegie dkk. 2022 [6]	*Early Prediction of Gestational Diabetes with Parameter-Tuned K-Nearest Neighbor*	K-Nearest Neighbor (KNN)	Model KNN dengan hyperparameter tuning mencapai akurasi 82.5% untuk prediksi diabetes tahap awal

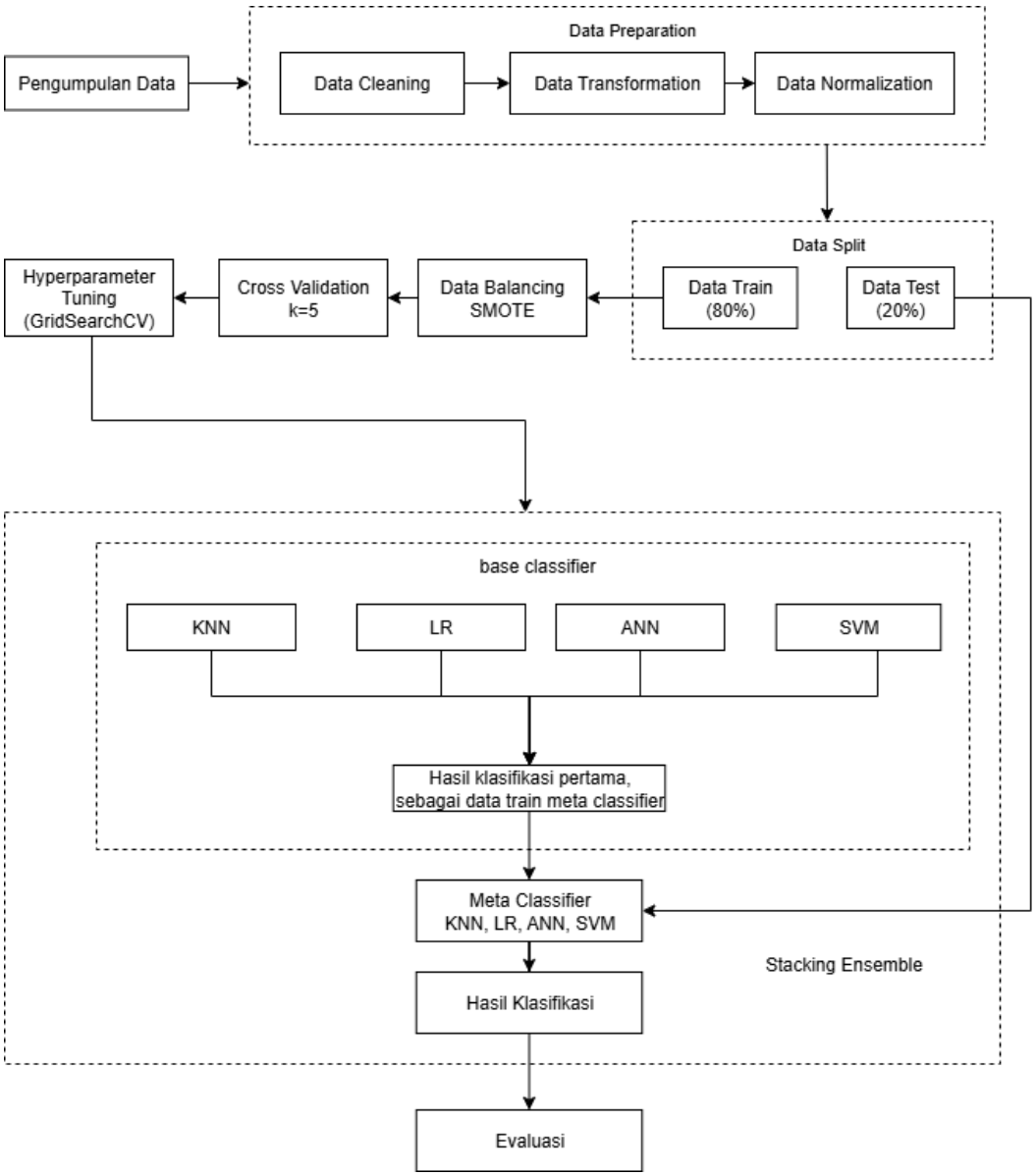
	Classifier		
Dimas Aryo Anggoro dan Dian Novitaningrum. 2020 [8]	*Comparison of Accuracy Level of Support Vector Machine (SVM) and Artificial Neural Network (ANN) Algorithms in Predicting Diabetes Mellitus Disease*	Support Vector Machine (SVM) dan Artificial Neural Network (ANN)	Setelah normalisasi data, ANN menghasilkan akurasi 85.20%, lebih tinggi dibandingkan SVM dengan akurasi 83.54%
Vikas Chaurasia dan Saurabh Pal. 2021 [56]	*Stacking-Based Ensemble Framework and Feature Selection Technique for the Detection of Breast Cancer*	Stacking Ensemble (SVM, KNN, Naive Bayes, Perceptron dengan Logistic Regression sebagai meta model)	Stacking ensemble mencapai akurasi tertinggi 99.968%, lebih unggul dibandingkan model tunggal SVM (98.958%), KNN (98.768%), Naive Bayes (98.652%)
I Kadek Yogi Prayoga dkk. 2025 [57]	Klasifikasi Penyakit Demam Berdarah Menggunakan Algoritma Stacking Ensemble Learning	Stacking Ensemble (KNN, Naive Bayes, C4.5 dengan Logistic Regression sebagai meta model)	Stacking ensemble mencapai akurasi 84,15%, precision 85,28%, recall 92,62%, dan F1-Score 88,70%

Alfredo Daza dkk. 2024 [10]	Stacking Ensemble Approach to Diagnosing the Disease of Diabetes	Stacking Ensemble (Decision Tree, Gradient Boosting, KNN, Random Forest, Logistic Regression, SVM, Bayesian Networks)	Model Stacking dengan oversampling mencapai akurasi 91,5%, sensitivitas 91,6%, F1-score 91,49%, dan precision 91,5%

BAB III
METODOLOGI PENELITIAN

3.1 Alur Sistem

Alur sistem dalam penelitian ini menggambarkan alur kerja dan tahapan-tahapan yang dilakukan dari awal hingga akhir. Diagram alur dari sistem tersebut disajikan pada *Gambar 3. 1*



Gambar 3. 1 Alur Sistem

3.2 Pengumpulan Data

Dalam tahap ini akan mengumpulkan data yang berperan sebagai objek dari penelitian ini untuk di olah menggunakan metode terpilih hingga menghasilkan

performa model yang optimal.

3.2.1. Sumber Data

Pada penelitian ini, data yang digunakan adalah data primer yang berfungsi untuk membangun model klasifikasi. Data ini merupakan kumpulan rekam medis pasien dari Puskesmas Pulo Lor, Jombang, dengan total data sebanyak 367 baris. Atribut yang akan digunakan antara lain adalah umur, jenis kelamin, Lingkar Perut, tekanan darah (tensi), HbA1C, dan gula darah acak (GDA). Variabel target dari data ini akan dibagi menjadi dua kelas, yaitu non-Diabetes Melitus untuk pasien yang tidak terjangkit diabetes dan Diabetes Melitus untuk pasien yang terjangkit diabetes.

Tabel 3. 1 Sample Data

Umur	JK	IMT	LP	Sistolik	Diastolik	Hba1c	GDP	GD2PP	Diabetes
73	L	27,39	77	117	70	7,1	149	149	Diabetes Melitus
60	L	20,28	71	136	77	8,4	179	175	Diabetes Melitus
75	L	18,49	66	159	86	6,2	80	217	Non-Diabetes Melitus
70	P	24,46	72	149	98	5,7	91	178	Non-Diabetes Melitus
61	P	25,44	73	130	71	8,2	103	149	Diabetes Melitus

3.2.2. Variabel Penelitian

Pada Tabel 3. 1 memperlihatkan seluruh atribut pada data yang akan digunakan untuk membangun model klasifikasi. Berikut penjelasan mengenai atribut data dari penelitian dan penjelasan definisi dari masing - masing atribut dijabarkan pada Tabel 3. 2 berikut.

Tabel 3. 2 Atribut data yang digunakan

Atribut	Tipe Data	Keterangan

Umur	Numerik	Menjelaskan umur dari pasien
JK	Kategori	Menjelaskan mengenai jenis kelamin dari pasien
IMT	Numerik	Menjelaskan mengenai index massa tubuh pasien
LP	Numerik	Menjelaskan mengenai lingkar perut dari pasien
Sistolik	Numerik	Menjelaskan mengenai sistolik dari pasien
Diastolik	Numerik	Menjelaskan mengenai diastolik dari pasien
Hba1c	Numerik	Menjelaskan mengenai tingkat Hba1c dalam darah dari pasien
GDP	Numerik	Menjelaskan mengenai tingkat gula darah puasa dari pasien
GD2PP	Numerik	Menjelaskan mengenai tingkat gula darah setelah makan dari pasien
Diabetes	Kategori	Status diagnosis pasien

3.3 Prepocessing

3.3.1. Data Cleaning

Pemeriksaan awal pada data penelitian dilakukan untuk mendeteksi adanya nilai yang hilang (*missing value*). Hasil deteksi menunjukkan tidak ada *missing value* pada dataset, sehingga tidak diperlukan tindakan penghapusan baris. Dengan demikian, integritas data untuk pengukuran kinerja model tetap terjaga.

Selanjutnya, dilakukan juga deteksi data duplikat. Data yang teridentifikasi sebagai duplikat akan dihapus untuk menghindari redundansi. Langkah ini penting karena data duplikat tidak memberikan informasi baru bagi model, namun dapat menambah ukuran dataset yang berpotensi memperlambat waktu komputasi.

3.3.2. Tranformasi Data

Algoritma klasifikasi dalam penelitian ini berbasis operasi matematis, sehingga atribut yang bersifat kategorikal perlu diubah menjadi format numerik agar dapat diproses. Selain itu, konversi ini juga merupakan prasyarat untuk menerapkan teknik *oversampling* yang bekerja dengan menghitung jarak antar data.

Untuk melakukan transformasi tersebut, digunakan fungsi LabelEncoder() yang akan mengubah setiap nilai unik pada data kategori menjadi representasi angka. Sebagai contoh, atribut 'Jenis Kelamin' (*gender*) yang bersifat kategorikal

akan dikonversi menjadi format numerik untuk dapat diolah oleh model.

3.3.3. Data Normalization

Pada penelitian ini, data dinormalisasi menggunakan metode min-max scaler agar memiliki rentang nilai yang seragam antara 0 dan 1. Proses normalisasi ini difokuskan pada fitur-fitur numerik kontinu seperti Umur, IMT, LP (Lingkar Perut), Sistolik, Diastolik, Hba1c, dan GDP, GD2PP. Untuk fitur atau kolom lainnya tidak lakukan normalisasi data dikarenakan data berupa numerik kategorikal seperti pada fitur jenis kelamin. Adapun langkah-langkah normalisasi data menggunakan *min-max scaler* dirincikan dalam *flowchart* pada **Error! Reference source not found.**

3.3.4. Data Balancing

Pada tahap ini akan melakukan sampling data menggunakan metode SMOTE. Parameter yang digunakan untuk melakukan SMOTE pada penelitian ini yaitu data hasil normalisasi dan label dari data tersebut. SMOTE bekerja dengan menghitung perbedaan antara suatu sampel kelas minoritas dan tetangga terdekatnya [58], kemudian mengalikan selisih tersebut dengan angka acak antara 0 dan 1 [59]. Nilai yang diperoleh selanjutnya digunakan untuk membuat data sintetis yang memiliki karakteristik mirip dengan data asli [58]

3.3.5. Data Split

Dalam tahap ini, dataset diabetes melitus akan dibagi menjadi dua bagian yang independen, yaitu data latih (*training set*) dan data uji (*testing set*). Proporsi pembagian yang diterapkan dalam penelitian ini adalah 80% untuk data latih dan 20% untuk data uji, data yang digunakan pada saat training memiliki proporsi data yang lebih banyak jika dibandingkan testing[10]. Dengan total data yang digunakan sebanyak 367 baris, maka distribusinya adalah 294 baris data (80%) dialokasikan sebagai data latih untuk membangun model, dan 73 baris data sisanya (20%) digunakan sebagai data uji untuk mengevaluasi performa model.

3.3.6. Data Balancing

Pada tahap ini akan melakukan sampling data menggunakan metode SMOTE. Parameter yang digunakan untuk melakukan SMOTE pada penelitian ini yaitu data latih dan label dari data latih. SMOTE bekerja dengan menghitung

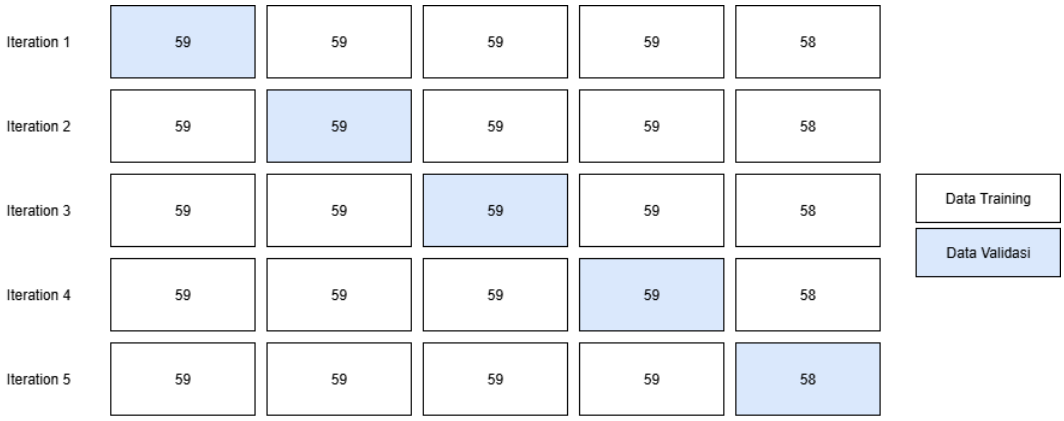
perbedaan antara suatu sampel kelas minoritas dan tetangga terdekatnya [58], kemudian mengalikan selisih tersebut dengan angka acak antara 0 dan 1 [59]. Nilai yang diperoleh selanjutnya digunakan untuk membuat data sintetis yang memiliki karakteristik mirip dengan data asli [58]

3.4 Modelling

Tahap pemodelan (*modelling*) merupakan tahap inti dari penelitian ini. Pada bab ini, akan dijelaskan secara rinci proses-proses yang dilakukan untuk membangun model klasifikasi, mulai dari penerapan teknik *Cross-Validation* dan *Hyperparameter Tuning*, hingga pelatihan beberapa algoritma individual yang pada akhirnya akan digabungkan menggunakan metode *Stacking Ensemble*

3.4.1. Cross Validation

Setelah data dipisahkan, data latih selanjutnya diproses menggunakan metode Cross-Validation dengan nilai K=5. Langkah ini bertujuan untuk mengevaluasi dan menemukan kombinasi parameter terbaik selama proses pelatihan model. Sebagaimana diilustrasikan pada Gambar 3. *1*, metode ini bekerja dengan membagi data latih menjadi lima bagian (atau *fold*) yang berukuran sama. Pelatihan kemudian dilakukan sebanyak lima putaran (iterasi), di mana pada setiap putaran, satu *fold* secara bergantian digunakan sebagai data validasi, sementara empat *fold* sisanya digunakan sebagai data training.



Gambar 3. 2 Pembagian CrossValidation pada data training

3.4.2. HyperParameter Tuning (GridSearchCV)

Untuk meningkatkan performa dari setiap model, dilakukan tahap optimasi menggunakan Hyperparameter Tuning. Proses ini memanfaatkan metode GridSearchCV untuk secara sistematis mencari kombinasi *hyperparameter* yang

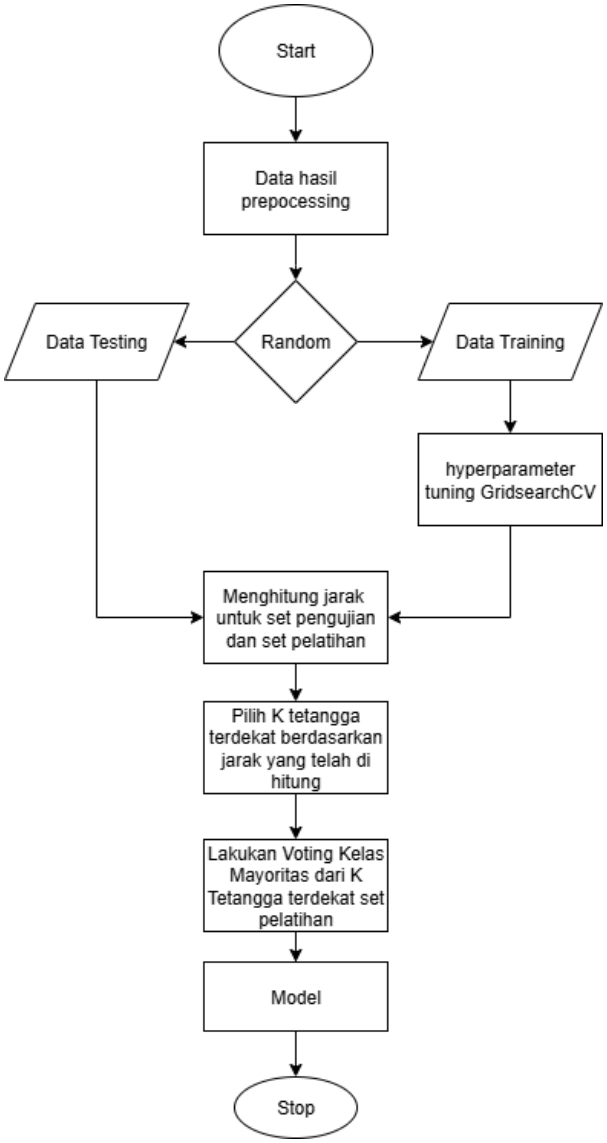
paling optimal bagi masing-masing algoritma yang digunakan. GridSearchCV bekerja dengan menjalankan proses Cross-Validation berulang kali untuk setiap kemungkinan kombinasi *hyperparameter* yang diberikan. Kombinasi yang menghasilkan skor rata-rata Cross-Validation tertinggi akan dipilih sebagai konfigurasi model terbaik. Daftar lengkap mengenai kombinasi *hyperparameter* yang diuji untuk setiap metode dirincikan pada Tabel 3. *3*

Tabel 3. 3 Kombinasi Nilai Hyperparameter

No	Metode	Hyperparameter	Value
1	K-Nearest Neighbor	K	[3,5,7,9,11,13,15]
		Metric	[Euclidean, Manhattan]
2	Logistic Regression	Penalty	[l1,l2]
		Solver	[lgbfs, liblinear]
		C	[0.001, 0.01, 0.1, 1, 100, 1000]
3	Support Vector Machine	C	[0.1, 1, 10, 100, 1000]
		Gamma	[scale, auto]
		Kernal	[rbf, linear]
4	Artificial Neural Network	Epoch	[500, 1000, 2000]
		Learning Rate	[0.001, 0.01, 0.1]
		Activation	[Tanh, Sigmoid, ReLu]
		Neuron Hidden	[1-50]

3.4.3. K-Nearest Neighbor

Dalam penelitian ini, K-Nearest Neighbor (KNN) digunakan sebagai salah satu model klasifikasi. Berikut adalah rincian langkah-langkah kerja algoritma KNN yang diterapkan, sesuai dengan visualisasi pada



Gambar 3. 3 Tahapan Klasifikasi Menggunakan KNN

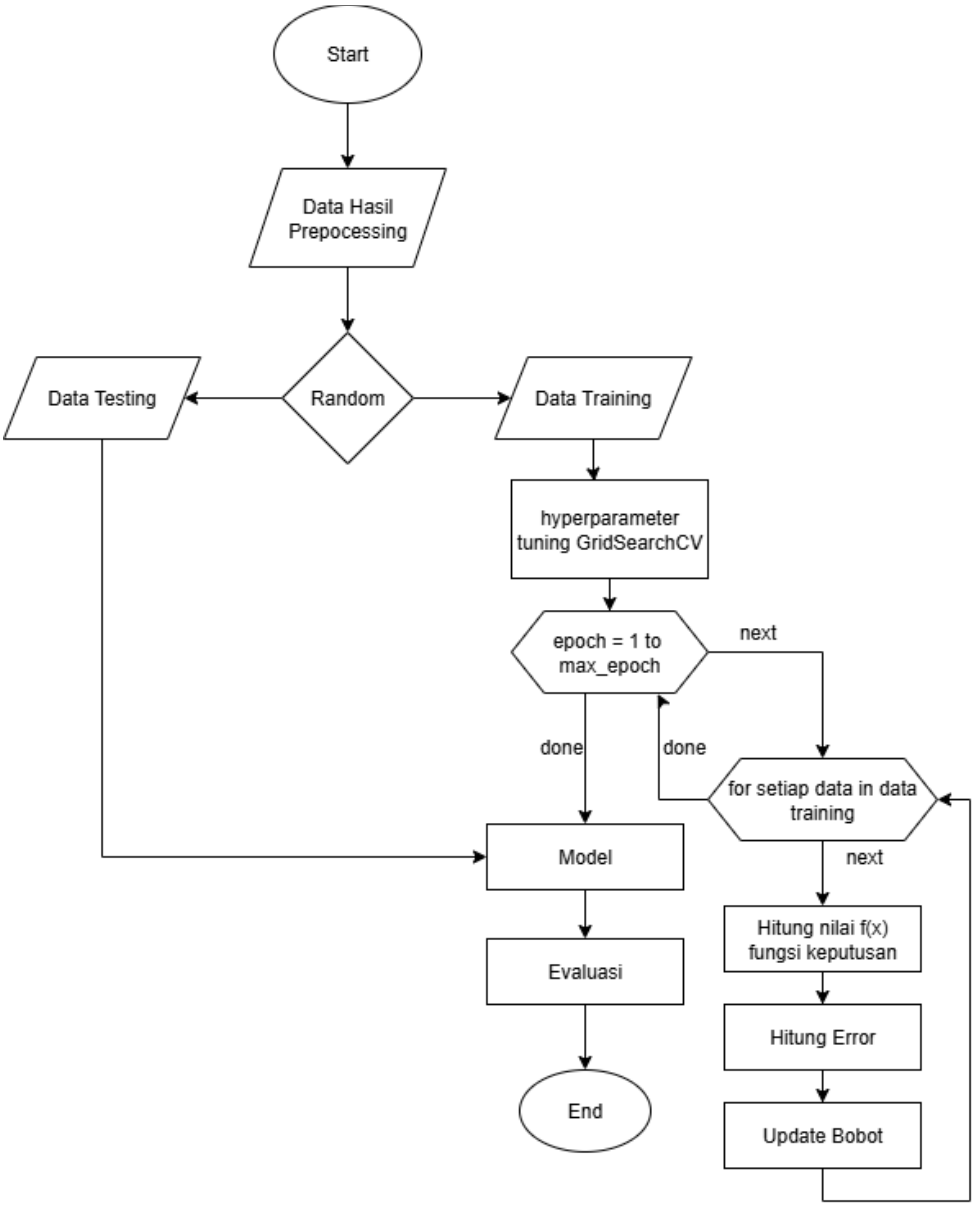
Berikut merupakan tahapan perhitungan untuk klasifikasi data menggunakan metode KNN antara lain:

1. Setelah melalui tahap pra-pemrosesan, Data kemudian dibagi secara acak menjadi dua bagian: Data Training dan Data Testing.
2. Melakukan optimasi *hyperparameter* untuk menemukan konfigurasi parameter yang paling efektif bagi model menggunakan GridSearchCV
3. Untuk setiap data pada Data Testing, dihitung jarak (misalnya, Jarak Euclidean) ke semua data yang ada pada Data Training.
4. Berdasarkan jarak yang telah dihitung, dipilih K tetangga terdekat dari set pelatihan untuk setiap data uji.

5. Dilakukan voting kelas mayoritas dari K tetangga terdekat tersebut. Kelas yang paling sering muncul akan menjadi hasil prediksi untuk data uji yang bersangkutan.
6. Kumpulan hasil prediksi untuk seluruh data uji ini merupakan keluaran akhir dari Model.

3.4.4. Logistic Regression

Metode selanjutnya yang digunakan pada penelitian ini adalah LR. Terdapat beberapa langkah-langkah yang digunakan untuk melakukan klasifikasi menggunakan metode LR sesuai dengan flowchart pada Gambar 3. *4*



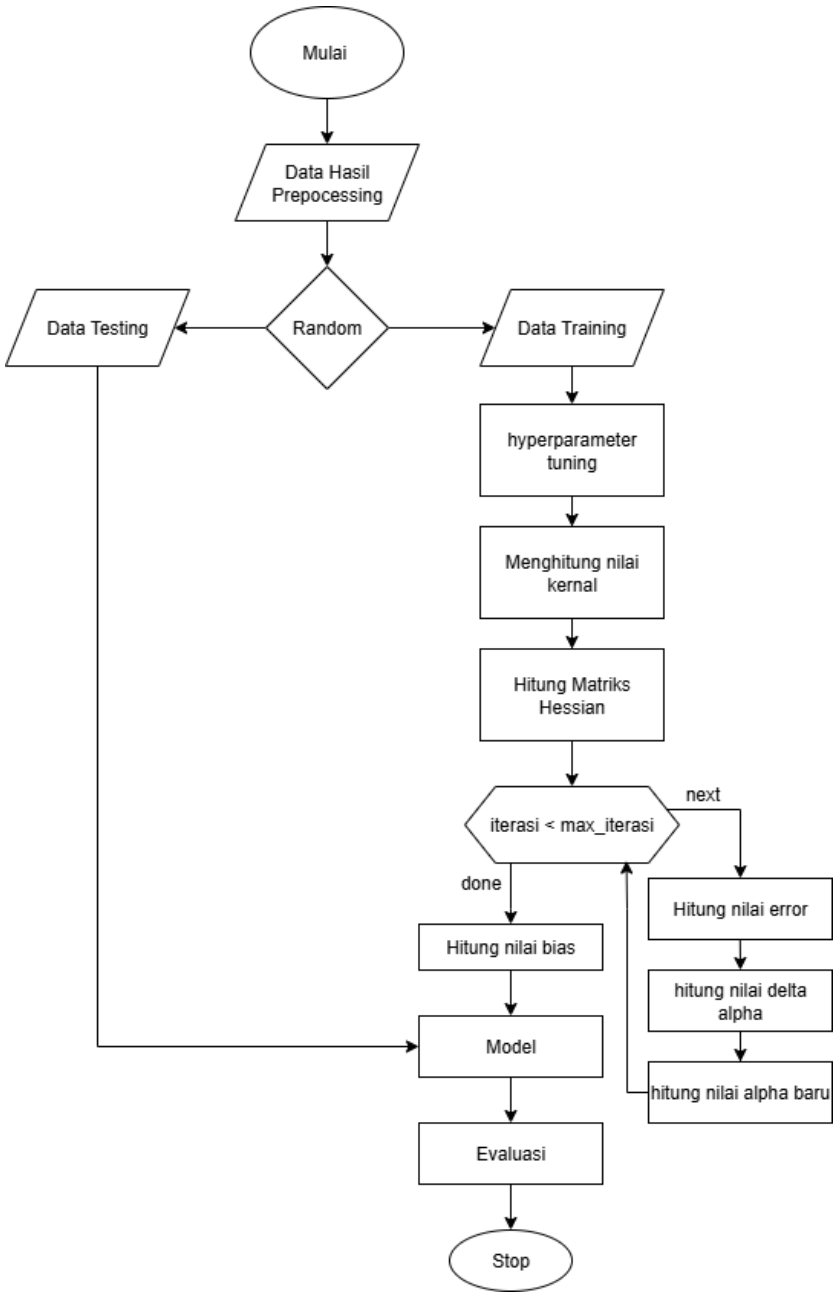
Gambar 3. 4 Tahapan Klasifikasi Menggunakan LR

Berikut merupakan tahapan perhitungan untuk klasifikasi data menggunakan metode LR antara lain:

1. Setelah melalui tahap pra-pemrosesan, dataset dibagi secara acak menjadi dua bagian: Data Training dan Data Testing.
2. Melakukan optimasi *hyperparameter* untuk menemukan konfigurasi parameter yang paling efektif bagi model menggunakan GridSearchCV
3. Setiap epoch dan iterasi yang dilakukan pada saat training akan menghitung f(x), nilai error dan update bobot.
4. Langkah ke-3 dilakukan secara berulang dengan setiap epoch memiliki iterasi sama jumlahnya dengan banyak data yang digunakan. Proses training ini akan berlangsung selama belum memenuhi jumlah epoch maksimal yang ditentukan.
5. Setelah semua epoch selesai, proses pelatihan berhenti dan nilai bobot terakhir akan membentuk Model final yang siap digunakan.
6. Untuk melakukan klasifikasi pada Data Testing, Model tersebut digunakan untuk menghitung nilai f(x) untuk setiap data uji. Hasilnya digunakan untuk menentukan kelas: jika nilai f(x) lebih besar atau sama dengan 0,5, data diklasifikasikan sebagai kelas positif (1), dan jika nilainya lebih kecil dari 0,5, data diklasifikasikan sebagai kelas negatif (0).
7. Hasil prediksi pada Data Testing kemudian dievaluasi untuk mengukur performa model.

3.4.5. Support Vector Machine

Metode selanjutnya yang digunakan pada penelitian ini adalah SVM. Terdapat beberapa langkah-langkah yang digunakan untuk melakukan klasifikasi menggunakan metode SVM sesuai dengan flowchart pada Gambar 3. *5*



Gambar 3. 5 Tahapan Klasifikasi Menggunakan SVM

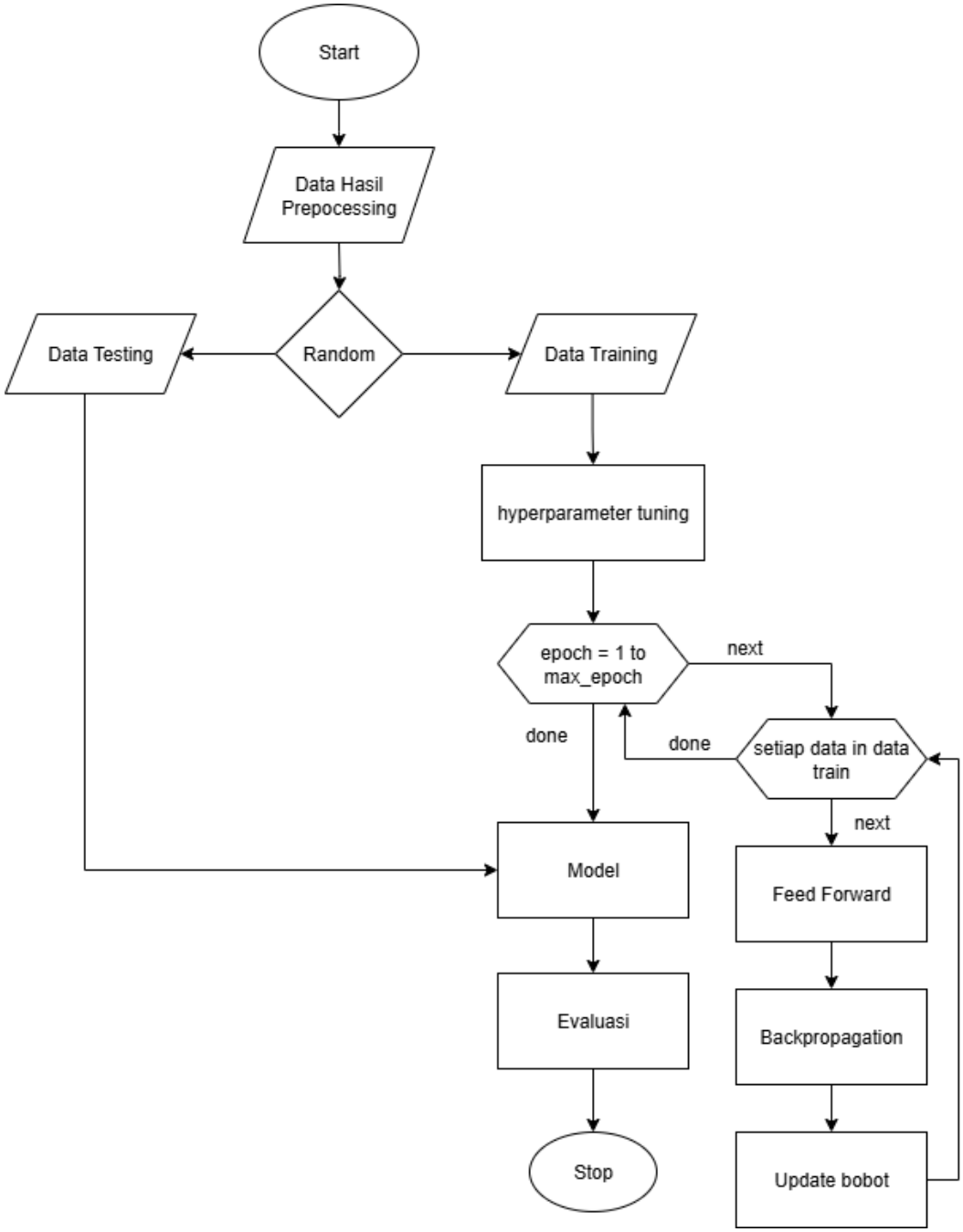
Berikut merupakan tahapan perhitungan untuk klasifikasi data menggunakan metode SVM antara lain:

1. Setelah melalui tahap pra-pemrosesan, dataset dibagi secara acak menjadi dua bagian: Data Training dan Data Testing.
2. Melakukan optimasi *hyperparameter* untuk menemukan konfigurasi parameter yang paling efektif bagi model menggunakan GridSearchCV
3. Menghitung nilai dot product setiap data training menggunakan fungsi kernel.
4. Menghitung nilai matriks hessian berdasarkan nilai kernel.

5. Proses pelatihan optimisasi kemudian dilakukan secara berulang (*iterasi*). Di setiap iterasi, tiga langkah berikut dijalankan secara berurutan:
    a. Mengitung nilai error.
    b. Menghitung nilai delta alpha.
    c. Menghitung nilai alpha baru
6. Menghitung nilai bias, dengan melakukan perhitungan terlebih dahulu untuk nilai bobot dot product pada kelas positif dan kelas negatif.
7. Jika nilai α_i dan b sudah diketahui, sebuah Model final terbentuk. Untuk memprediksi Data Testing, pertama-tama dilakukan perhitungan *dot product* (atau kernel) antara data testing dengan data training. Hasilnya kemudian digunakan untuk menghitung fungsi keputusan f(x).
8. Hasil dari fungsi keputusan f(x) digunakan untuk menentukan kelas: jika nilainya lebih dari atau sama dengan 1, data diklasifikasikan sebagai kelas positif (1), dan jika nilainya kurang dari atau sama dengan -1, data diklasifikasikan sebagai kelas negatif (0).
9. Hasil prediksi pada Data Testing dievaluasi untuk mengukur performa model

3.4.6. Artificial Neural Network

Metode selanjutnya yang digunakan pada penelitian ini adalah ANN. Terdapat beberapa langkah-langkah yang digunakan untuk melakukan klasifikasi menggunakan metode ANN sesuai dengan flowchart pada Gambar 3. *6*



Gambar 3. 6 Tahapan Klasifikasi Menggunakan ANN

Berikut merupakan tahapan perhitungan untuk klasifikasi data menggunakan metode ANN antara lain:

1. Setelah melalui tahap pra-pemrosesan, dataset dibagi secara acak menjadi dua bagian: Data Training dan Data Testing.
2. Melakukan optimasi *hyperparameter* untuk menemukan konfigurasi parameter yang paling efektif bagi model menggunakan GridSearchCV

3. Proses pelatihan dimulai, di mana fase *Forward Propagation* dan *Backward Propagation* berikut (langkah 4 sampai 13) diulang untuk setiap data dalam *Data Training* selama jumlah *epoch* yang ditentukan.

Fase I: Forward Propagation

4. Melakukan penjumlahan bobot dari input ke hidden.
5. Mengaplikasikan fungsi aktivasi pada hasil penjumlahan tersebut untuk menghasilkan output dari neuron di *hidden layer*.
6. Menjumlahkan bobot dari *hidden layer* ke *output layer*.
7. Mengaplikasikan fungsi aktivasi pada hasil penjumlahan tersebut untuk menghasilkan ouput dari neuron di *ouput layer*.

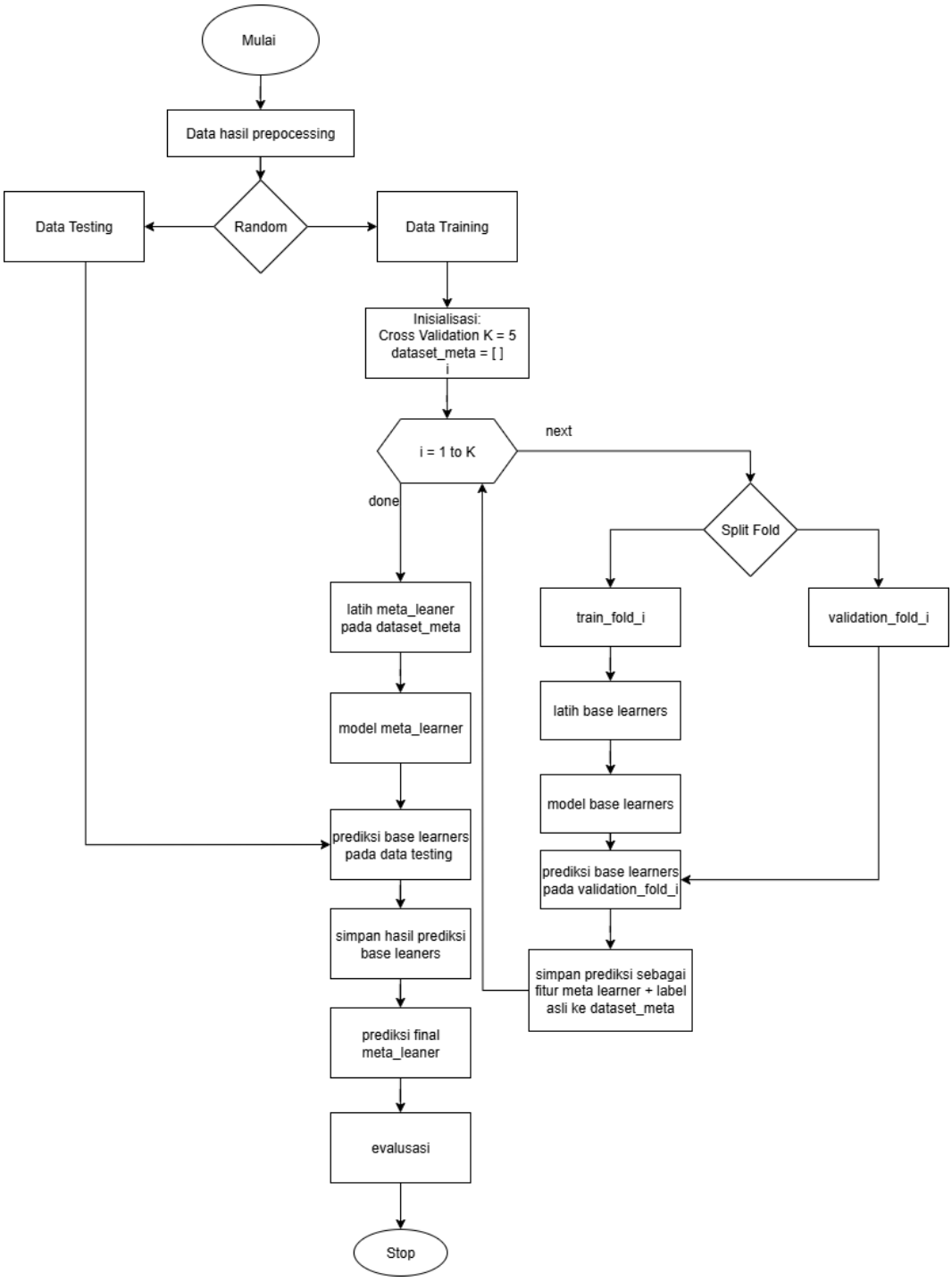
Fase II: Backward Propagation

8. Setelah nilai output didapatkan pada ouput layer, kemudian melakukan perhitungan nilai error pada unit output layer.
9. Menghitung nilai koreksi yang diperlukan untuk bobot dan bias antara *hidden layer* dan *output layer*.
10. Menghitung nilai error pada neuron hidden layer.
11. Menghitung nilai koreksi yang diperlukan untuk bobot dan bias antara *input layer* dan *hidden layer*.
12. Memperbarui bobot dan bias pada koneksi dari *hidden layer* ke *output layer*.
13. Memperbarui (*update*) bobot dan bias pada koneksi dari *input layer* ke *hidden layer*.
14. Setelah proses training selesai, lakukan prediksi pada Data Testing dengan menjalankan kembali langkah *Forward Propagation* (langkah 4 sampai 7) menggunakan bobot dan bias final yang telah diperbarui.
15. Hasil prediksi pada data uji kemudian dievaluasi untuk mengukur performa model, dan proses pun selesai.

3.4.7. Stacking Ensemble

Penelitian ini akan melakukan penggabungan metode yang telah dipilih sebelumnya antara lain: KNN, LR, ANN dan SVM. Konsep penggabungan metode yang digunakan adalah Stacking. Cara kerja metode penggabungan ini adalah dengan cara melakukan penumpukan model pada beberapa level yaitu level-0 dan level-1 untuk membuat suatu hasil klasifikasi data. Berikut merupakan tahapan

untuk konsep penggabungan metode menggunakan Stacking pada flowchart *Gambar 3. 7* dibawah ini.



Gambar 3. 7 Algoritma Klasifikasi Stacking Ensemble

Berikut merupakan tahapan-tahapan yang digunakan pada konsep penggabungan Stacking antara lain:

1. Setelah melalui tahap pra-pemrosesan, dataset dibagi secara acak menjadi dua

bagian: Data Training dan Data Testing.

2. Pada Data Training, proses dimulai dengan inisialisasi *Cross-Validation* (K=5) dan pembuatan sebuah dataset kosong bernama dataset_meta. Selanjutnya, dilakukan perulangan sebanyak 5 kali. Di setiap perulangan, data training dibagi menjadi train_fold_i dan validation_fold_i. *Base learners* dilatih menggunakan train_fold_i, lalu hasilnya digunakan untuk memprediksi validation_fold_i. Prediksi ini beserta label aslinya disimpan ke dalam dataset_meta.
3. Setelah perulangan selesai, *model meta-learner* dilatih menggunakan dataset_meta yang telah terbentuk. Dataset ini berisi prediksi dari *base learners* sebagai fitur dan label asli sebagai target. Berikut merupakan contoh dataset dari hasil prediksi yang disimpan di model base-learners pada Tabel 3. *4*

Tabel 3. 4 Contoh Dataset Dari Hasil Prediksi Base-Learners

pred_knn	**pred_lr**	**pred_ann**	**pred_svm**	**diabetes**
1	0	1	1	1
1	1	1	0	1
1	0	0	0	0

4. Dataset yang disajikan pada Tabel 3. *4* akan berfungsi sebagai data latih untuk model *meta-classifier*. Dalam penelitian ini, akan dilakukan eksperimen dengan menguji empat algoritma berbeda secara bergantian untuk mengisi peran sebagai *meta-classifier*, yaitu Logistic Regression (LR), KNN, ANN, dan SVM.
5. Selanjutnya, *model base learners* yang sudah ada digunakan untuk membuat prediksi pada Data Testing. Hasil dari prediksi ini kemudian disimpan.
6. *Model meta_learner* yang telah dilatih pada tahap sebelumnya digunakan untuk membuat prediksi akhir, dengan input berupa hasil prediksi *base learners* pada Data Testing.
7. Hasil prediksi akhir dari *meta-learner* kemudian dievaluasi dengan membandingkannya dengan label asli dari Data Testing untuk mengukur performa model.

3.5 Evaluasi

Evaluasi model dilakukan untuk mengukur performa dalam mengklasifikasikan apakah seorang pasien terkena penyakit diabetes atau tidak. Proses evaluasi menggunakan *confunsion matrix* untuk menghasilkan beberapa nilai pengujian antara lain *accuracy, precision, recall* dan yang terakhir *F1-Score.* Evaluasi ini akan menilai seberapa besar hasil kinerja masing-masing metode tunggal serta pada saat metode tunggal tersebut digabungkan menggunakan *stacking ensemble.*

3.6 Skenario Pengujian

Skenario pengujian dalam penelitian ini dirancang untuk mengevaluasi performa dari dua pendekatan. Pertama, pengujian dilakukan pada setiap algoritma secara individual. Kedua, pengujian juga dilakukan pada model gabungan (*ensemble*) yang dibangun menggunakan konsep stacking. Rincian lengkap mengenai skenario pengujian ini disajikan pada Tabel 3. *5*

Tabel 3. 5 Skenario Pengujian

No	Skenario	Keterangan
1	Skenario 1	Melakukan pengujian pada masing masing metode tunggal yaitu KNN*, Logistic Regression*, ANN, dan SVM. Pengujian ini diserta dengan *hyperparamater tuning* menggunakan *gridsearch*
2	Skenario 2	Melakukan pengujian pada stacking ensemble dengan meta-classifier secara bergatian yaitu KNN, Logistic Regression, ANN dan SVM. Pengujian ini disertai dengan *hyperparamater tuning* menggunakan *gridsearch*

DAFTAR PUSTAKA

[1] Y. Religia, A. Nugroho, and W. Hadikristanto, "Comparative Analysis of Optimization Algorithms in Random Forest for Classification of Bank Marketing Data," *Jurnal RESTI*, vol. 5, no. 1, pp. 187–192, Feb. 2021, doi: 10.29207/resti.v5i1.2813.

[2] A. H. Permana, F. R. Umbara, and F. Kasyidi, "Klasifikasi Penyakit Jantung Tipe Kardiovaskular Menggunakan Adaptive Synthetic Sampling dan Algoritma Extreme Gradient Boosting," *Building of Informatics, Technology and Science (BITS)*, vol. 6, no. 1, pp. 499–508, Jun. 2024, doi: 10.47065/bits.v6i1.5421.

[3] N. Novianti, M. Zarlis, and P. Sihombing, "Penerapan Algoritma Adaboost Untuk Peningkatan Kinerja Klasifikasi Data Mining Pada Imbalance Dataset Diabetes," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 6, no. 2, p. 1200, Apr. 2022, doi: 10.30865/mib.v6i2.4017.

[4] A. Kharis Pratama *et al.*, "KLASIFIKASI DATA GEMPA BUMI DI PULAU JAWA MENGGUNAKAN ALGORITMA EXTREME GRADIENT BOOSTING," *Jurnal Mahasiswa Teknik Informatika*, vol. 7, no. 4, pp. 2923–2929, 2023, doi: 10.36040/jati.v7i4.7296.

[5] M. Alkhalaf, P. Yu, J. Shen, and C. Deng, "A review of the application of machine learning in adult obesity studies," *Applied Computing and Intelligence*, vol. 2, no. 1, pp. 32–48, 2022, doi: 10.3934/aci.2022002.

[6] T. A. Assegie, T. Suresh, R. Purushothaman, S. Ganesan, and N. K. Kumar, "Early Prediction of Gestational Diabetes with Parameter-Tuned K-Nearest Neighbor Classifier," *Journal of Robotics and Control (JRC)*, vol. 4, no. 4, pp. 452–457, 2023, doi: 10.18196/jrc.v4i4.18412.

[7] Y. Nora Marlim, L. Suryati, and N. Agustina, "Early Detection of Diabetes Using Machine Learning with Logistic Regression Algorithm," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi* |, vol. 11, no. 2, 2022, doi: https://doi.org/10.22146/jnteti.v11i2.3586.

[8] D. A. Anggoro and D. Novitaningrum, "Comparison of accuracy level of support vector machine (SVM) and artificial neural network (ANN) algorithms in predicting diabetes mellitus disease," *ICIC Express Letters*, vol. 15, no. 1, pp. 9–18, Jan. 2021, doi: 10.24507/icicel.15.01.9.

[9] V. Aceña, I. Martín de Diego, R. R. Fernández, and J. M. Moguerza, "Minimally overfitted learners: A general framework for ensemble learning," *Knowl Based Syst*, vol. 254, Oct. 2022, doi: 10.1016/j.knosys.2022.109669.

[10] A. Daza, C. F. Ponce Sánchez, G. Apaza-Perez, J. Pinto, and K. Zavaleta Ramos, "Stacking ensemble approach to diagnosing the disease of diabetes," *Inform Med Unlocked*, vol. 44, Jan. 2024, doi: 10.1016/j.imu.2023.101427.

[11] L. Nguyen Van and G. Lee, "Optimizing Stacked Ensemble Machine Learning Models for Accurate Wildfire Severity Mapping," *Remote Sens (Basel)*, vol. 17, no. 5, Mar. 2025, doi: 10.3390/rs17050854.

[12] Agusviyanda, Hamdani, M. K. Anam, Agustin, and M. I. Wibowo, "STACKING ENSEMBLE MACHINE LEARNING MODEL FOR EARLY DETECTION OF CHRONIC KIDNEY DISEASE IN INDONESIA," *Rabit : Jurnal Teknologi dan Sistem Informasi Univrab*, vol. 10, no. 2, pp. 1244–1252, Jul. 2025, doi: 10.36341/rabit.v10i2.6501.

[13] I. K. Sifat and M. K. Kibria, "Optimizing hypertension prediction using ensemble learning approaches," *PLoS One*, vol. 19, no. 12, Dec. 2024, doi: 10.1371/journal.pone.0315865.

[14] A. W. Mucholladin, F. Abdurrachman Bachtiar, and M. T. Furqon, "Klasifikasi

Penyakit Diabetes menggunakan Metode Support Vector Machine," 2021. [Online]. Available: http://j-ptiik.ub.ac.id
[15] A. H. Nst, I. Rasyid Munthe, and A. Putra Juledi, "Implementasi Data Mining Algoritma Apriori untuk Meningkatkan Penjualan," *Jurnal Teknik Informatika Unika St. Thomas (JTIUST*, vol. 6, Jun. 2021.
[16] I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," May 01, 2021, *Springer*. doi: 10.1007/s42979-021-00592-x.
[17] J. Gui *et al.*, "A Survey on Self-supervised Learning: Algorithms, Applications, and Future Trends," *IEEE Trans Pattern Anal Mach Intell*, Jul. 2024, [Online]. Available: http://arxiv.org/abs/2301.05712
[18] H. H. Rashidi, N. Tran, S. Albahra, and L. T. Dang, "Machine learning in health care and laboratory medicine: General overview of supervised learning and Auto-ML," Jul. 01, 2021, *John Wiley and Sons Inc*. doi: 10.1111/ijlh.13537.
[19] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Appl Soft Comput*, vol. 97, Dec. 2020, doi: 10.1016/j.asoc.2019.105524.
[20] I. Izonin, R. Tkachenko, N. Shakhovska, B. Ilchyshyn, and K. K. Singh, "A Two-Step Data Normalization Approach for Improving Classification Accuracy in the Medical Diagnosis Domain," *Mathematics*, vol. 10, no. 11, Jun. 2022, doi: 10.3390/math10111942.
[21] M. M. Ahsan, M. A. P. Mahmud, P. K. Saha, K. D. Gupta, and Z. Siddique, "Effect of Data Scaling Methods on Machine Learning Algorithms and Model Performance," *Technologies (Basel)*, vol. 9, no. 3, Sep. 2021, doi: 10.3390/technologies9030052.
[22] D. P. Utomo and M. Mesran, "Analisis Komparasi Metode Klasifikasi Data Mining dan Reduksi Atribut Pada Data Set Penyakit Jantung," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 4, no. 2, p. 437, Apr. 2020, doi: 10.30865/mib.v4i2.2080.
[23] A. Mohammed and R. Kora, "A comprehensive review on ensemble deep learning: Opportunities and challenges," Feb. 01, 2023, *King Saud bin Abdulaziz University*. doi: 10.1016/j.jksuci.2023.01.014.
[24] M. A. Alemayehu, S. D. Kebede, A. D. Walle, D. N. Mamo, E. B. Enyew, and J. B. Adem, "A stacked ensemble machine learning model for the prediction of pentavalent 3 vaccination dropout in East Africa," *Front Big Data*, vol. 8, 2025, doi: 10.3389/fdata.2025.1522578.
[25] B. Sunarko *et al.*, "Edu Komputika Journal Penerapan Stacking Ensemble Learning untuk Klasifikasi Efek Kesehatan Akibat Pencemaran Udara," *Edu Komputika*, vol. 10, no. 1, 2023, [Online]. Available: http://journal.unnes.ac.id/sju/index.php/edukom
[26] Y. Nora Marlim, L. Suryati, and N. Agustina, "Deteksi Dini Penyakit Diabetes Menggunakan Machine Learning dengan Algoritma Logistic Regression," 2022.
[27] F. Reviantika, Y. Azhar, and G. I. Marthasari, "Analisis Klasifikasi SMS Spam Menggunakan Logistic Regression," APIC, 2021. [Online]. Available: https://www.cnbcindonesia.com
[28] I. A. Siradjuddin, C. V. Angkoso, and K. E. Pramana, *Buku Pembelajaran Mesin dan Implementasinya Dengan Menggunakan Python*. Sleman: DeepPublish, 2024.
[29] R. Rahim and A. S. Ahmar, "Cross-Validation and Validation Set Methods for Choosing K in KNN Algorithm for Healthcare Case Study," *JINAV: Journal of Information and Visualization*, vol. 3, no. 1, pp. 57–61, Jul. 2022, doi: 10.35877/454ri.jinav1557.
[30] C. Z. V. Junus, T. Tarno, and P. Kartikasari, "KLASIFIKASI MENGGUNAKAN METODE SUPPORT VECTOR MACHINE DAN RANDOM FOREST UNTUK DETEKSI AWAL RISIKO DIABETES MELITUS," *Jurnal Gaussian*, vol. 11,

no. 3, pp. 386–396, Jan. 2023, doi: 10.14710/j.gauss.11.3.386-396.
[31] O. H. Rahman, G. Abdillah, and A. Komarudin, “Klasifikasi Ujaran Kebencian pada Media Sosial Twitter Menggunakan Support Vector Machine,” *Jurnal RESTI*, vol. 5, no. 1, pp. 17–23, Feb. 2021, doi: 10.29207/resti.v5i1.2700.
[32] S. Rahma Yustihan and P. Pandu Adikara, “Analisis Sentimen berbasis Aspek terhadap Data Ulasan Rumah Makan menggunakan Metode Support Vector Machine (SVM),” 2021. [Online]. Available: http://j-ptiik.ub.ac.id
[33] D. Finaliamartha, D. Supriyadi, and G. F. Fitriana, “PENERAPAN METODE JARINGAN SYARAF TIRUAN BACKPROPAGATION UNTUK PREDIKSI TINGKAT KEMISKINAN DI PROVINSI JAWA TENGAH”, doi: 10.25126/jtiik.202294806.
[34] “Artificial Neural Networks Pengenalan Pola Pasword Angka Menggunakan Metode Heteroassociative Memory.”
[35] K. H. Suradiradja, “Algoritme Machine Learning Multi-Layer Perceptron dan Recurrent Neural Network untuk Prediksi Harga Cabai Merah Besar di Kota Tangerang,” *Faktor Exacta*, vol. 14, no. 4, p. 194, Jan. 2022, doi: 10.30998/faktorexacta.v14i4.10376.
[36] A. F. Hardiyanti and D. Fitrianah, “Perbandingan Algoritma C4.5 dan Multilayer Perceptron untuk Klasifikasi Kelas Rumah Sakit di DKI Jakarta,” *Jurnal Telekomunikasi dan Komputer*, vol. 11, no. 3, p. 198, Dec. 2021, doi: 10.22441/incomtech.v11i3.10632.
[37] S. Sunardi, A. Yudhana, and G. Z. Muflih, “Sistem Prediksi Curah Hujan Bulanan Menggunakan Jaringan Saraf Tiruan Backpropagation,” *JURNAL SISTEM INFORMASI BISNIS*, vol. 10, no. 2, pp. 155–162, Dec. 2020, doi: 10.21456/vol10iss2pp155-162.
[38] … | Hamdani, H. Arifin, and Z. Septiarini, “Expert System of Dengue Disease Using Artificial Neural Network Classifier,” 2022.
[39] M. A. Rahman, “A deep learning framework for football match prediction,” *SN Appl Sci*, vol. 2, no. 2, Feb. 2020, doi: 10.1007/s42452-019-1821-5.
[40] P. Silitonga, A. Bustamam, H. Muradi, W. Mangunwardoyo, and B. E. Dewi, “Comparison of dengue predictive models developed using artificial neural network and discriminant analysis with small dataset,” *Applied Sciences (Switzerland)*, vol. 11, no. 3, pp. 1–16, Feb. 2021, doi: 10.3390/app11030943.
[41] A. Muis, E. M. Zamzami, and E. B. Nababan, “Convolutional Neural Network Activation Function Performance on Image Recognition of The Batak Script,” *Sinkron*, vol. 9, no. 1, pp. 182–195, Jan. 2024, doi: 10.33395/sinkron.v9i1.13192.
[42] E. Saraswati, Y. Umaidah, and A. Voutama, “Penerapan Algoritma Artificial Neural Network untuk Klasifikasi Opini Publik Terhadap Covid-19,” *Generation Journal*, vol. 5, no. 2, pp. 2580–4952, 2021, doi: https://doi.org/10.29407/gj.v5i2.16125.
[43] S. Risal, F. Apriyadi, A. Sumardin, A. D. Achmad, and A. N. Puteri, “Enhancing Stroke Prediction with Logistic Regression and Support Vector Machine Using Oversampling Techniques,” *Jurnal RESTI*, vol. 9, no. 3, pp. 646–658, Jun. 2025, doi: 10.29207/resti.v9i3.6431.
[44] Z. M. Alhakeem, Y. M. Jebur, S. N. Henedy, H. Imran, L. F. A. Bernardo, and H. M. Hussein, “Prediction of Ecofriendly Concrete Compressive Strength Using Gradient Boosting Regression Tree Combined with GridSearchCV Hyperparameter-Optimization Techniques,” *Materials*, vol. 15, no. 21, Nov. 2022, doi: 10.3390/ma15217432.
[45] I. K. Nti, O. Nyarko-Boateng, and J. Aning, “Performance of Machine Learning Algorithms with Different K Values in K-fold CrossValidation,” *International Journal of Information Technology and Computer Science*, vol. 13, no. 6, pp. 61–71, Dec. 2021, doi: 10.5815/ijitcs.2021.06.05.

[46] P. L. Romadloni, B. dhi Kusuma, and W. M. Baihaqi, “KOMPARASI METODE PEMBELAJARAN MESIN UNTUK IMPLEMENTASI PENGAMBILAN KEPUTUSAN DALAM MENENTUKAN PROMOSI JABATAN KARYAWAN,” *JATI*, vol. 6, no. 2, Sep. 2022, doi: https://doi.org/10.36040/jati.v6i2.5238.
[47] A. F. Daru, M. B. Hanif, and E. Widodo, “Improving Neural Network Performance with Feature Selection Using Pearson Correlation Method for Diabetes Disease Detection,” 2021.
[48] A. B. Abbas *et al.*, “A case–control study to evaluate hematological indices in blood of diabetic and non-diabetic individuals in Ibb City, Yemen,” *Sci Rep*, vol. 13, no. 1, Dec. 2023, doi: 10.1038/s41598-023-43973-3.
[49] A. Mansoori *et al.*, “Prediction of type 2 diabetes mellitus using hematological factors based on machine learning approaches: a cohort study analysis,” *Sci Rep*, vol. 13, no. 1, Dec. 2023, doi: 10.1038/s41598-022-27340-2.
[50] N. Susanti *et al.*, “HUBUNGAN USIA PADA KEJADIAN DIABETES MELLITUS TIPE-2 DENGAN PENDEKATAN STEPWISE,” vol. 5, no. 2, 2024, doi: https://doi.org/10.31004/jkt.v5i2.28312.
[51] R. Arania *et al.*, “HUBUNGAN ANTARA USIA, JENIS KELAMIN, DAN TINGKAT PENDIDIKAN DENGAN KEJADIAN DIABETES MELLITUS DI KLINIK MARDI WALUYO LAMPUNG TENGAH,” 2021.
[52] B. J. Patel, D. N. Mehta, A. Vaghani, and K. Patel, “Correlation of Body Mass Index (BMI) with Saliva and Blood Glucose Levels in Diabetic and Non-Diabetic Patients,” *J Pharm Bioallied Sci*, vol. 15, no. 6, pp. S1204–S1207, Jul. 2023, doi: 10.4103/jpbs.jpbs_159_23.
[53] R. Siren, J. G. Eriksson, and H. Vanhanen, “Waist circumference a good indicator of future risk for type 2 diabetes and cardiovascular disease,” *BMC Public Health*, vol. 12, no. 1, 2012, doi: 10.1186/1471-2458-12-631.
[54] G. Jia and J. R. Sowers, “Hypertension in Diabetes: An Update of Basic Mechanisms and Clinical Disease,” Nov. 01, 2021, *Lippincott Williams and Wilkins*. doi: 10.1161/HYPERTENSIONAHA.121.17981.
[55] S. I. Sherwani, H. A. Khan, A. Ekhzaimy, A. Masood, and M. K. Sakharkar, “Significance of HbA1c test in diagnosis and prognosis of diabetic patients,” Jul. 03, 2016, *Libertas Academica Ltd.* doi: 10.4137/Bmi.s38440.
[56] V. Chaurasia and S. Pal, “Stacking-Based Ensemble Framework and Feature Selection Technique for the Detection of Breast Cancer,” *SN Comput Sci*, vol. 2, no. 2, Apr. 2021, doi: 10.1007/s42979-021-00465-3.
[57] I. Kadek, Y. Prayoga, I. Made, G. Sunarya, and P. H. Suputra, “Klasifikasi Penyakit Demam Berdarah Menggunakan Algoritma Stacking Ensemble Learning”, [Online]. Available: https://doi.org/10.31598
[58] M. A. Nugraha, M. I. Mazdadi, A. Farmadi, Muliadi, and T. H. Saragih, “Penyeimbangan Kelas SMOTE dan Seleksi Fitur Ensemble Filter pada Support Vector Machine untuk Klasifikasi Penyakit Liver,” *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 10, no. 6, pp. 1273–1284, Dec. 2023, doi: 10.25126/jtiik.2023107234.
[59] L. N. I. Afida, F. A. Bachtiar, and I. Cholissodin, “Klasifikasi Aktivitas Manusia Menggunakan Metode Long Short-Term Memory,” *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 11, no. 2, pp. 357–368, Aug. 2024, doi: 10.25126/jtiik.20241127060.