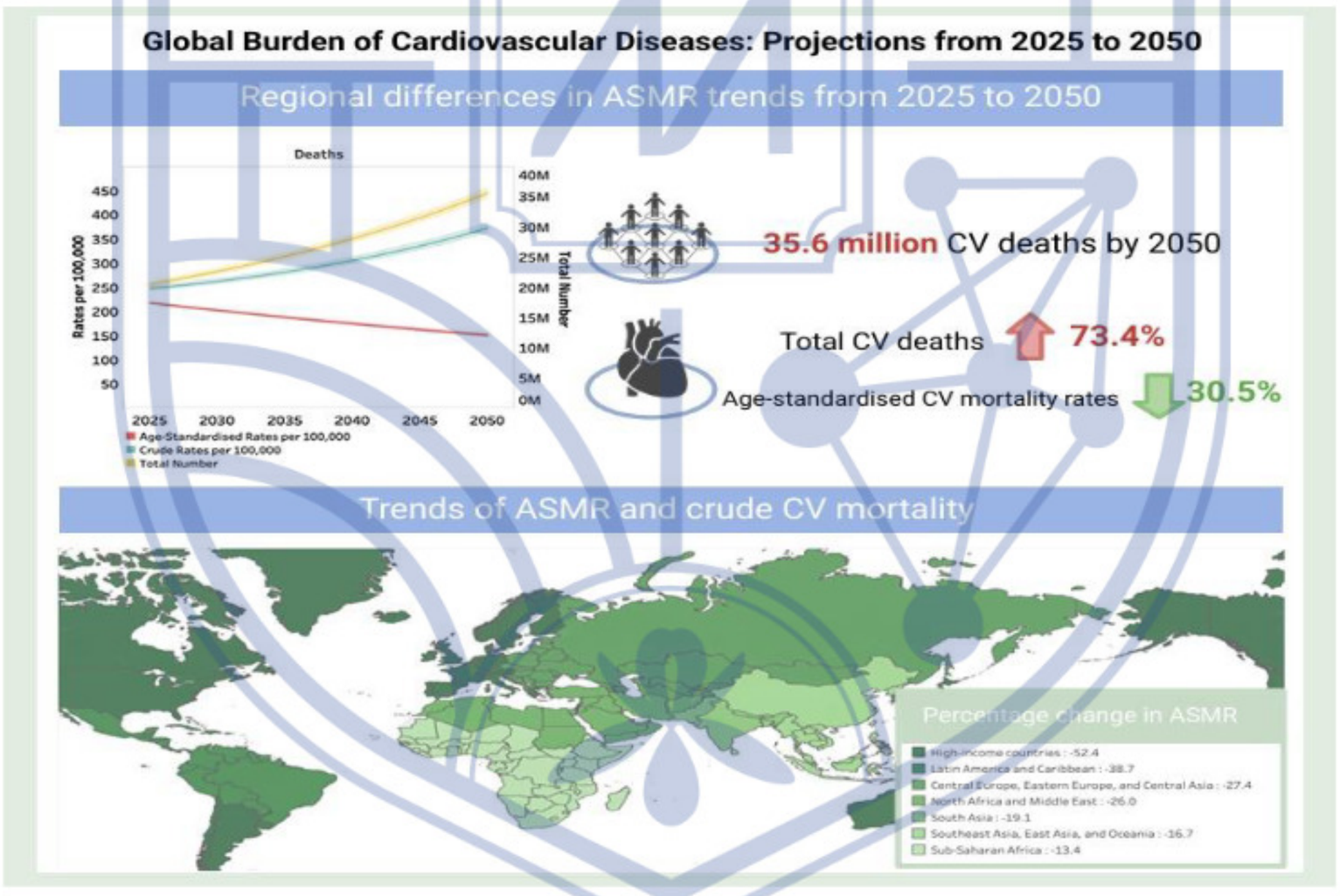
BAB II
KAJIAN LITERATUR

2.1 *Cardiovascular Disease* (CVD)

CVD merupakan penyakit yang menyerang sistem jantung dan pembuluh darah serta dapat berkembang secara bertahap dengan karakteristik yang kompleks dan sering kali tidak menunjukkan gejala yang signifikan pada tahap awal. Kondisi ini menyebabkan banyak kasus baru teridentifikasi ketika telah memasuki fase yang lebih serius, sehingga meningkatkan risiko terjadinya komplikasi dan kejadian klinis yang fatal. Minimnya gejala awal serta kompleksitas perkembangan penyakit menjadikan proses penilaian risiko sejak dini menjadi sulit dilakukan, sehingga diperlukan pendekatan analisis yang lebih akurat dan komprehensif untuk melakukan klasifikasi risiko penyakit secara mendalam serta mendukung upaya pencegahan dan pengelolaan yang efektif [12],[13].



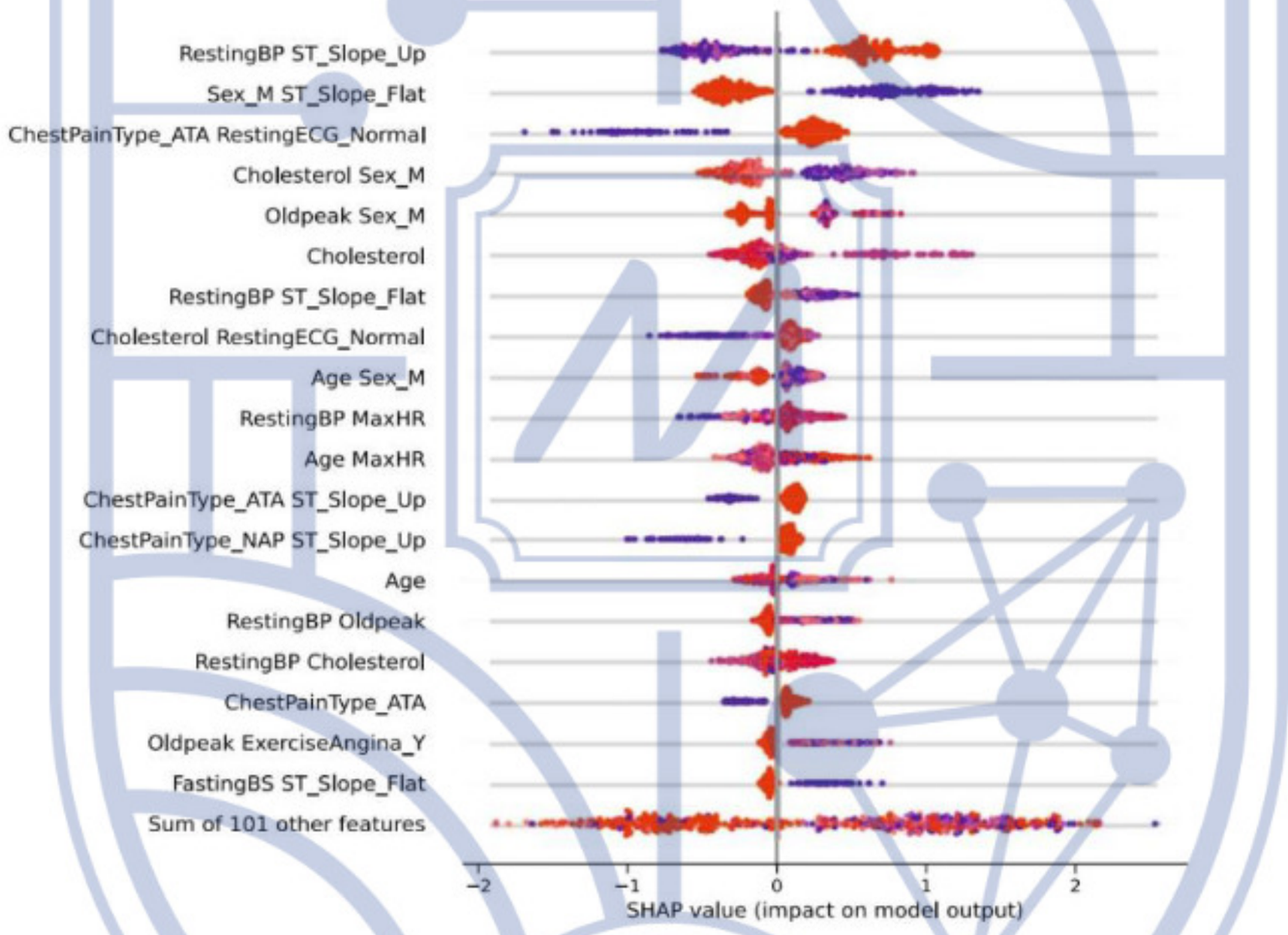
Gambar 2.1 Proyeksi Global CVD 2025–2050 [4]

Gambar 2.1 menunjukkan bahwa hasil proyeksi beban global CVD dari tahun 2025 hingga 2050, di mana jumlah kematian akibat CVD diperkirakan terus meningkat secara signifikan hingga mencapai sekitar 35,6 juta kematian pada tahun 2050 dengan kenaikan total sebesar 73,4%. Meskipun demikian, angka kematian yang telah disesuaikan berdasarkan usia (*age-standardized mortality rate/ASMR*) justru mengalami penurunan sebesar 30,5%, yang mengindikasikan adanya peningkatan dalam kualitas layanan

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

kesehatan serta upaya pencegahan penyakit. Selain itu, terdapat perbedaan tren antar wilayah di dunia, di mana negara dengan pendapatan tinggi menunjukkan penurunan angka kematian yang lebih signifikan dibandingkan negara berkembang, sehingga mencerminkan adanya ketimpangan dalam akses layanan kesehatan dan efektivitas penanganan penyakit kardiovaskular secara global [4].

CVD berkembang akibat interaksi berbagai faktor risiko yang bersifat kompleks, meliputi faktor demografis, klinis, dan gaya hidup. Faktor demografis mencakup usia dan jenis kelamin, sedangkan faktor klinis meliputi tekanan darah, kadar kolesterol, dan kadar gula darah. Adapun faktor gaya hidup meliputi kebiasaan merokok, pola makan, aktivitas fisik, dan konsumsi alkohol [14].



Gambar 2.2 *SHAP Value* (*impact on model output)* [14]

Gambar 2.2 menampilkan analisis menggunakan nilai *Shapley*, sebagian besar variabel yang berkontribusi dalam model merupakan hasil interaksi antar fitur. Sebagai contoh, interaksi antara tekanan darah saat istirahat (*resting blood pressure*) dengan kemiringan segmen ST (ST *slope*) menunjukkan pengaruh yang signifikan terhadap prediksi risiko penyakit jantung. Hasil analisis menunjukkan bahwa individu dengan kemiringan ST ke atas dan tekanan darah tinggi cenderung memiliki risiko yang lebih rendah, sedangkan pasien laki-laki dengan kemiringan ST datar memiliki kecenderungan risiko yang lebih tinggi terhadap penyakit kardiovaskular. Selain itu, interaksi antara jenis

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

kelamin laki-laki dan kadar kolesterol tinggi juga menunjukkan peningkatan probabilitas terjadinya penyakit kardiovaskular [14].

Selain itu, penyakit kardiovaskular juga memberikan dampak ekonomi yang sangat besar dan kompleks, baik pada tingkat individu, rumah tangga, maupun sistem pelayanan kesehatan nasional. Biaya yang timbul tidak hanya berupa biaya langsung seperti layanan rawat inap, pemeriksaan laboratorium, prosedur intervensi, serta konsumsi obat-obatan dalam jangka panjang, tetapi juga mencakup biaya tidak langsung seperti kehilangan produktivitas kerja, penurunan kualitas hidup, serta beban sosial yang harus ditanggung oleh keluarga pasien. Variasi biaya perawatan ini dipengaruhi oleh berbagai faktor, termasuk karakteristik demografis pasien seperti usia dan jenis kelamin, kondisi klinis seperti tingkat keparahan penyakit dan jumlah komorbiditas, serta faktor pelayanan kesehatan seperti lama rawat inap dan kompleksitas prosedur medis yang dilakukan. Oleh karena itu, pasien dengan kondisi yang lebih kompleks cenderung memerlukan biaya perawatan yang jauh lebih tinggi, sehingga meningkatkan beban ekonomi secara signifikan [15].

2.2 Klasifikasi CVD

Klasifikasi merupakan proses pengelompokan data ke dalam kategori atau kelas tertentu berdasarkan karakteristik yang dimiliki oleh data. Dalam bidang *machine learning*, klasifikasi digunakan untuk memprediksi atau menentukan label suatu data sehingga dapat membantu proses pengambilan keputusan secara lebih terstruktur dan sistematis [16].

Dalam melakukan klasifikasi terhadap risiko penyakit CVD memungkinkan pengelompokan tingkat risiko secara lebih sistematis, di mana ML digunakan untuk mengelompokkan individu ke dalam kategori tingkat risiko tertentu seperti *Low, Intermediary dan High* [14]. Metode *Convolutional Neural Network* (CNN) yang dipadukan dengan teknik *Synthetic Minority Over-Sampling Technique* (SMOTE) dapat dengan mudah diimplementasikan pada tugas klasifikasi, namun cenderung menunjukkan performa yang lebih rendah ketika dihadapkan pada data yang kompleks [11]. Di sisi lain, *Artificial Neural Network* (ANN) yang dikombinasikan dengan SMOTE mampu memberikan kinerja yang lebih optimal pada *dataset* berukuran besar dengan tingkat kompleksitas tinggi, akan tetapi pendekatan ini memerlukan sumber daya komputasi yang lebih besar [17].

Penelitian lainnya menggunakan fitur yang dipilih dalam pembuatan data yaitu atribut umum yang diambil dari basis data lembaga kesehatan cenderung memiliki data

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

yang *imbalanced. dataset* kemudian di seimbangkan dengan menggunakan teknik *over-sampling* seperti penambahan data berdasarkan penggabungan antar data minoritas yang sudah ada. Penggunaan teknik seperti SMOTE pada penelitian sebelumnya dengan metrik performa yang di pilih untuk evaluasi model adalah *confusion matrix, precision, recall* dan *accuracy.* Pada penelitian ini, SMOTE memiliki hasil yang sangat memuaskan dan memiliki nilai F1 cukup tinggi [18].

Penelitian pada klasifikasi CVD juga dilakukan dengan menggabungkan antara DNN dengan teknik SMOTE, dimana DNN efektif dalam mempelajari pola data yang kompleks dan *nonlinear*, sementara SMOTE digunakan untuk menghasilkan data baru dari kelas minoritas guna mengatasi *imbalanced*. *Dataset* yang digunakan berbasis sinyal *elektrokardiogram* (ECG) yang telah diverifikasi secara klinis, dengan distribusi awal yang tidak seimbang antara kelas normal dan CVD, kemudian diseimbangkan menggunakan SMOTE sehingga rasio kelas menjadi lebih proporsional sebelum proses pelatihan dan pengujian model. Evaluasi kinerja dilakukan menggunakan *confusion matrix* serta metrik *accuracy, precision, recall,* dan *F1-score*, yang menunjukkan bahwa setelah penerapan SMOTE, performa model DNN meningkat secara signifikan dengan nilai *accuracy* mencapai 97,62%, *precision* 97,66%, *recall* 97,62%, dan *F1-score* 97,64%. Penelitian ini merepresentasikan berdasarkan karakteristik sinyal medis ECG yang mencerminkan kondisi fisiologis jantung, sehingga mampu membantu model dalam mengklasifikasi pola penyakit kardiovaskular secara lebih akurat [19].

2.3 *Machine Learning*

Machine Learning (ML) merupakan salah satu cabang dari kecerdasan buatan (*Artificial Intelligence*) yang berfokus pada pengembangan sistem komputer agar mampu belajar secara otomatis dari data dan meningkatkan kinerjanya tanpa harus diprogram secara eksplisit. Pendekatan ini memungkinkan komputer untuk membangun model berdasarkan data historis sehingga dapat mengenali pola, melakukan prediksi, serta mendukung proses pengambilan keputusan secara mandiri. Konsep utama dalam ML adalah *learning from experience*, di mana data digunakan sebagai sumber pengetahuan, sehingga semakin banyak data yang dipelajari maka model yang dihasilkan diharapkan semakin akurat [20],[21].

Secara umum, proses dalam ML meliputi beberapa tahapan utama, yaitu pengumpulan data yang relevan dengan permasalahan, *preprocessing* data berupa pembersihan dan transformasi data, pemilihan atau ekstraksi fitur untuk memperoleh

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

informasi yang penting, pelatihan model (*training*) menggunakan data latih, serta pengujian model (*testing*) menggunakan data uji untuk mengevaluasi kinerja model dalam memprediksi data baru. Berdasarkan metode pembelajarannya, *machine learning* dibagi menjadi tiga jenis utama, yaitu *supervised learning*, *unsupervised learning*, dan *reinforcement learning*. *Supervised learning* menggunakan data berlabel dan banyak diterapkan pada tugas klasifikasi serta *regression*, *unsupervised learning* digunakan untuk menemukan pola tersembunyi pada data tanpa label seperti *clustering*, sedangkan *reinforcement learning* berbasis interaksi antara agen dan lingkungan melalui mekanisme *trial and error* untuk memperoleh hasil yang optimal. Dalam konteks klasifikasi penyakit, khususnya penyakit kardiovaskular, pendekatan *supervised learning* menjadi metode yang paling banyak digunakan karena mampu mempelajari hubungan antara variabel input dan *output* secara optimal sehingga menghasilkan prediksi yang lebih akurat [20],[21].

Selain itu, ML melibatkan berbagai algoritma seperti *Decision Tree, Support Vector Machine* (SVM)*, Naïve Bayes, K-Nearest Neighbor* (KNN), dan *Neural Network*, di mana setiap algoritma memiliki karakteristik yang berbeda sesuai dengan jenis permasalahan yang dihadapi. Dalam penerapannya, ML memiliki keunggulan seperti kemampuan dalam mengolah data dalam jumlah besar, menemukan pola yang kompleks, serta meningkatkan akurasi prediksi. Hal ini menjadikan ML banyak diterapkan dalam bidang kesehatan untuk menganalisis data medis yang kompleks dan berskala besar. Pendekatan ini mampu mengidentifikasi pola tersembunyi yang sulit dideteksi secara manual, sehingga dapat mendukung proses diagnosis, prediksi risiko penyakit, serta pengambilan keputusan klinis yang lebih akurat, termasuk pada penyakit kardiovaskular. Namun demikian, ML juga memiliki keterbatasan seperti ketergantungan pada kualitas data, risiko *overfitting*, serta kebutuhan sumber daya komputasi yang cukup tinggi, sehingga pemilihan metode dan pengolahan data yang tepat menjadi faktor penting dalam menghasilkan model yang optimal [20],[22].

2.4 Data Preprocessing

Data *preprocessing* merupakan tahapan penting dalam proses *machine learning* yang bertujuan untuk meningkatkan kualitas data sebelum digunakan dalam pemodelan. Tahapan ini meliputi proses pembersihan data, transformasi data, serta reduksi data sehingga data menjadi lebih terstruktur, konsisten, dan siap digunakan dalam analisis. Proses *preprocessing* sangat berpengaruh terhadap performa model, karena kualitas data yang baik akan menghasilkan model yang lebih akurat dan stabil [23].

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

1. *Data Cleaning*

   Data cleaning merupakan proses pembersihan data dari berbagai permasalahan seperti *missing value*, data duplikat, inkonsistensi, dan *outlier*. Data yang tidak bersih dapat menyebabkan bias dan menurunkan akurasi model, sehingga perlu dilakukan perbaikan sebelum tahap analisis. Beberapa teknik dalam data *cleaning* yaitu menghapus data duplikat, menangani *missing value*, memperbaiki inkonsistensi format data. Proses ini penting karena data yang tidak lengkap atau tidak akurat dapat menunjukkan hasil yang tidak valid dalam *machine learning*. *Mean imputation* merupakan salah satu teknik penanganan *missing value* pada data numerik dengan cara mengganti nilai yang hilang menggunakan nilai rata-rata (*mean*) dari atribut tersebut. Teknik ini digunakan untuk mempertahankan jumlah data agar tidak berkurang akibat penghapusan baris data yang memiliki nilai kosong. *Mean imputation* banyak digunakan karena sederhana dan mampu menjaga distribusi data tetap stabil, khususnya pada *dataset* dengan jumlah *missing value* yang relatif kecil. Namun demikian, penggunaan *mean imputation* juga dapat mengurangi variasi data karena nilai yang digunakan berasal dari rata-rata atribut [24],[25].

2. *Data Transformation*

   Data transformation merupakan proses mengubah data ke dalam format yang sesuai untuk analisis atau pemodelan. Transformasi dilakukan agar data dapat diproses oleh algoritma *machine learning* yang umumnya membutuhkan data dalam bentuk numerik. Beberapa teknik dalam data *transformation* yaitu Normalisasi dan standarisasi data, *Encoding* data kategorikal, *Scaling* nilai fitur. Transformasi data membantu meningkatkan performa model dengan memastikan setiap fitur memiliki skala dan representasi yang sesuai. *One-Hot Encoding* merupakan teknik transformasi data kategorikal ke dalam bentuk numerik dengan membuat representasi biner untuk setiap kategori. Pada metode ini, setiap kategori akan diubah menjadi kolom baru yang berisi nilai 1 jika data termasuk dalam kategori tersebut dan 0 jika tidak. Teknik ini banyak digunakan dalam *machine learning* karena sebagian besar algoritma hanya dapat memproses data numerik. Dengan menggunakan *One-Hot Encoding*, informasi kategori dapat direpresentasikan tanpa memberikan hubungan ordinal antar kategori [26],[27].

3. *Data Reduction*

   Data reduction adalah proses mengurangi jumlah fitur atau dimensi data tanpa menghilangkan informasi penting. Tujuannya adalah untuk meningkatkan efisiensi

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

komputasi dan mengurangi kompleksitas model. Beberapa teknik *data reduction* yaitu *Feature selection*, *Dimensionality reduction*, Penghapusan fitur tidak relevan. Dengan data yang lebih ringkas, model dapat bekerja lebih cepat dan menghindari *overfitting* [28].

2.5 *Deep Learning*

Deep Learning (DL) merupakan salah satu cabang dari *machine learning* yang menggunakan arsitektur jaringan saraf tiruan (*Artificial Neural Networks*) dengan banyak lapisan (*Deep Neural Networks*) untuk mempelajari representasi data secara bertingkat. Pendekatan ini memungkinkan model untuk secara otomatis mengekstraksi fitur dari data tanpa memerlukan proses ekstraksi fitur secara manual, sehingga mampu menangani permasalahan yang kompleks dengan tingkat akurasi yang tinggi. DL dirancang untuk meniru cara kerja otak manusia dalam memproses informasi, di mana setiap lapisan dalam jaringan berfungsi untuk mempelajari tingkat abstraksi yang berbeda dari data, mulai dari fitur sederhana hingga fitur yang lebih kompleks [29].

Secara umum, DL bekerja dengan menggunakan struktur jaringan saraf yang terdiri dari beberapa lapisan utama, yaitu *input layer*, *hidden layer*, dan *output layer*. Berbeda dengan *neural network* tradisional, DL memiliki lebih banyak *hidden layer* sehingga mampu mempelajari pola yang lebih kompleks dan *nonlinier*. Proses pembelajaran dalam DL dilakukan melalui algoritma *backpropagation*, yang digunakan untuk menyesuaikan bobot pada setiap *neuron* berdasarkan *error* antara hasil prediksi dan nilai aktual, sehingga model dapat meningkatkan performanya secara bertahap selama proses pelatihan menggunakan data dalam jumlah besar. Kemampuan ini menjadikan DL sangat efektif dalam mengolah berbagai jenis data seperti citra, teks, suara, maupun data medis. Salah satu arsitektur utama dalam DL adalah DNN, yaitu jaringan saraf tiruan yang memiliki lebih dari satu *hidden layer*, sehingga mampu memodelkan hubungan *nonlinier* yang kompleks antar variabel dan sangat cocok digunakan dalam analisis data medis, termasuk klasifikasi risiko penyakit kardiovaskular [29],[30].

Dalam implementasinya, DL telah banyak digunakan dalam berbagai bidang seperti kesehatan, pengolahan citra, dan analisis sinyal medis. Dalam perkembangannya, metode ini menjadi salah satu pendekatan yang paling dominan dalam *machine learning* karena kemampuannya dalam menyelesaikan permasalahan kompleks seperti pengenalan gambar, pengolahan bahasa alami (*Natural Language Processing*), serta analisis data kesehatan. Keunggulan utama DL terletak pada kemampuannya dalam meningkatkan akurasi prediksi

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

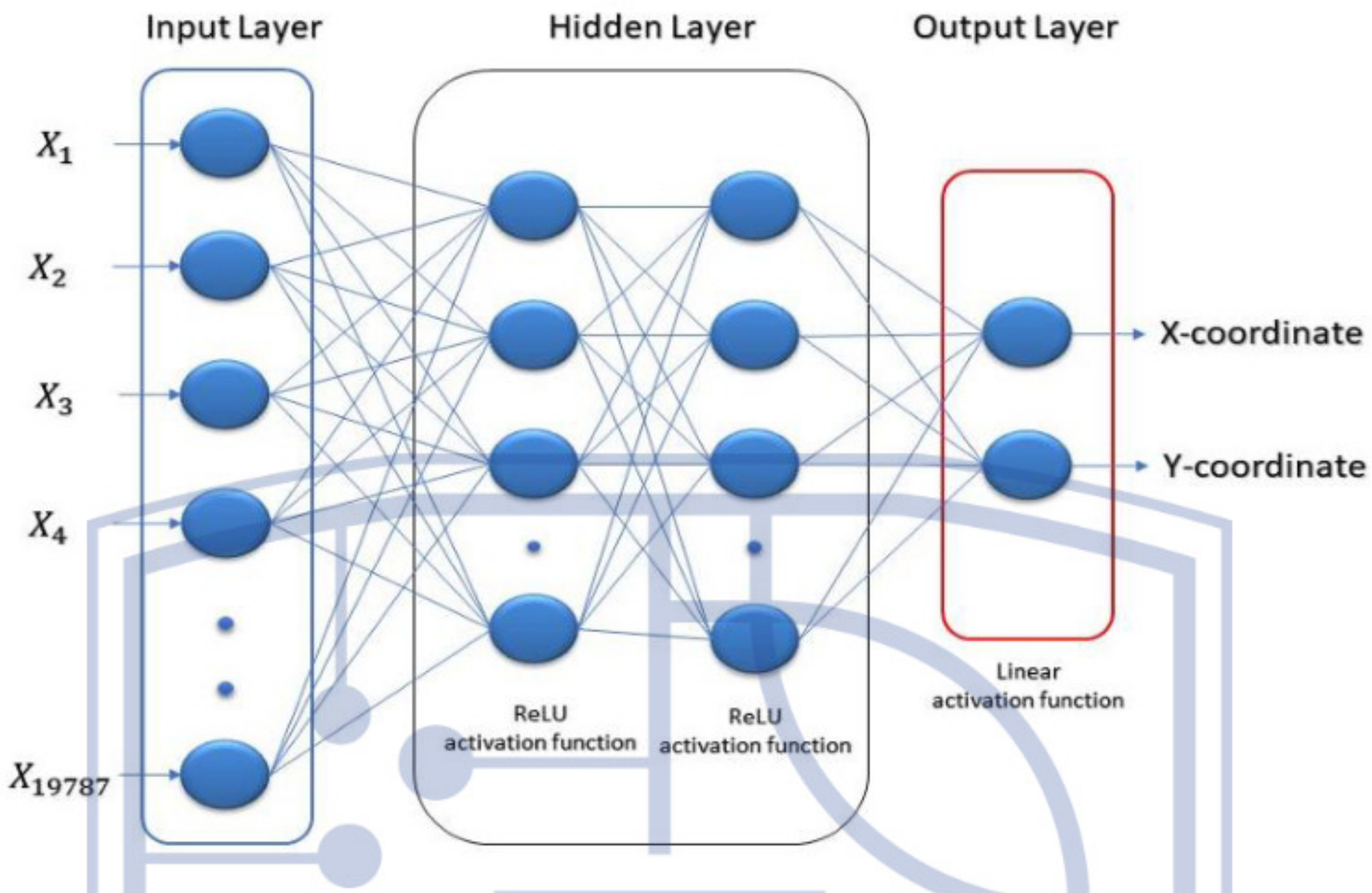
serta menangani data dalam jumlah besar dengan tingkat kompleksitas yang tinggi. Beberapa arsitektur DL yang populer antara lain *Convolutional Neural Network* (CNN), *Recurrent Neural Network* (RNN), dan DNN. Meskipun demikian, DL juga memiliki keterbatasan, seperti kebutuhan data dalam jumlah besar, waktu pelatihan yang relatif lama, serta kebutuhan sumber daya komputasi yang tinggi. Oleh karena itu, pemilihan metode DL perlu disesuaikan dengan karakteristik data dan tujuan penelitian agar menghasilkan model yang optimal [29],[30]. Salah satu pendekatan DL yang banyak digunakan dalam tugas klasifikasi adalah DNN yang akan dibahas secara lebih rinci pada bagian selanjutnya.

2.6 *Deep Neural Network* (DNN)

DNN merupakan bagian dari *deep learning* dengan memanfaatkan jaringan saraf berlapis banyak untuk mempelajari pola kompleks dari data dalam jumlah besar melalui proses pembelajaran bertahap pada setiap lapisannya. Kemampuan ini memungkinkan DNN dalam menangkap hubungan *nonlinier* antar variabel secara lebih mendalam, sehingga banyak digunakan dalam berbagai tugas klasifikasi, termasuk di bidang kesehatan. Namun demikian, efektivitas kinerja DNN sangat bergantung pada kualitas serta distribusi data yang digunakan selama proses pelatihan, sehingga dalam konteks klasifikasi CVD, permasalahan ketidakseimbangan data menjadi aspek penting yang perlu diperhatikan agar model dapat menghasilkan akurasi yang optimal. Dalam implementasinya, kemampuan tersebut didukung oleh struktur DNN yang terdiri dari beberapa lapisan utama yaitu *input layer*, *hidden layer*, dan *output layer*. *Input layer* berfungsi sebagai penerima data awal, kemudian data tersebut diproses melalui satu atau lebih *hidden layer* sebelum menghasilkan *output* pada *output layer*. Setiap lapisan terdiri dari *neuron* yang saling terhubung melalui bobot (*weights*) dan bias sebagai parameter utama dalam proses pembelajaran [31]. Setiap *neuron* dalam jaringan melakukan operasi matematis berupa kombinasi *linear* antara *input* dan bobot yang kemudian ditransformasikan menggunakan fungsi aktivasi. Fungsi aktivasi yang umum digunakan antara lain sigmoid, ReLU, dan fungsi lainnya yang bersifat *non-linear*.

Arsitektur DNN bersifat *feedforward*, di mana aliran data bergerak dari input menuju *output* tanpa adanya umpan balik. Dengan adanya banyak lapisan, DNN mampu membentuk representasi fitur yang lebih abstrak pada setiap *layer* sehingga meningkatkan kemampuan model dalam memahami pola data yang kompleks.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.



Gambar 2.3 Arsitektur Model DNN [24]

Berdasarkan arsitektur yang digunakan pada gambar 2.3, DNN terdiri dari tiga komponen utama yaitu *input layer*, *hidden layer*, dan *output layer*. *Input layer* berfungsi sebagai tempat masuknya data berupa fitur-fitur yang digunakan dalam penelitian, sedangkan *hidden layer* terdiri dari beberapa lapisan *neuron* yang saling terhubung secara penuh (*fully connected*) dan menggunakan fungsi aktivasi ReLU untuk mempelajari pola *nonlinier* dalam data. *Output layer* berfungsi untuk menghasilkan prediksi akhir model. Proses pembelajaran dilakukan melalui tahapan *forward propagation*, perhitungan *error* menggunakan fungsi *loss*, serta *backpropagation* untuk memperbarui bobot jaringan. Dengan mekanisme ini, model mampu meningkatkan akurasi prediksi secara bertahap. Selain itu, performa model juga dipengaruhi oleh beberapa *hyperparamater* seperti jumlah *layer*, jumlah *neuron*, *learning rate*, *epoch*, *batch size*, fungsi aktivasi, dan *optimizer* yang digunakan [32].

Kekuatan utama DNN terletak pada kemampuannya dalam mempelajari pola data yang kompleks melalui proses pembelajaran berlapis. Kemampuan ini dibentuk melalui serangkaian tahapan pemrosesan internal yang terstruktur dan berulang. Dalam konteks cara kerja DNN, proses tersebut mencakup beberapa komponen utama, yaitu penentuan arsitektur jaringan, proses *feedforward*, serta *backpropagation* dalam pembaruan bobot. Setiap tahapan saling berkontribusi dalam menghasilkan representasi fitur yang semakin

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

abstrak dan bermakna, sehingga memungkinkan DNN melakukan prediksi dan klasifikasi secara efektif [32].

Proses kerja DNN terdiri dari beberapa tahapan utama yang masing-masing dijelaskan beserta rumus matematisnya sebagai berikut [32].

1. *Input Layer*

   Input layer berfungsi sebagai tempat masuknya data berupa fitur-fitur yang digunakan dalam penelitian. Data direpresentasikan dalam bentuk *vector*, di mana setiap *neuron* pada lapisan ini mewakili satu fitur dari *dataset* dan akan diteruskan ke lapisan berikutnya tanpa mengalami perubahan.

   $x = (x_1, x_2, \dots, x_n)$ .....(2.1)

   Keterangan: xi menyatakan nilai dari setiap fitur ke-i yang digunakan sebagai input dalam model, sedangkan n menunjukkan jumlah keseluruhan fitur yang digunakan dalam proses pembelajaran.

2. *Hidden Layer*

   Hidden layer terdiri dari satu atau lebih lapisan *neuron* yang saling terhubung secara penuh. Pada lapisan ini, setiap *neuron* melakukan operasi matematis berupa kombinasi *linear* antara input, bobot, dan bias, yang kemudian ditransformasikan menggunakan fungsi aktivasi *nonlinier* seperti *Rectified Linear Unit* (ReLU) untuk mempelajari pola kompleks dalam data.

   Forward propagation merupakan proses mengalirkan data dari *input layer* menuju *output layer* melalui *hidden layer*. Pada tahap ini, setiap *neuron* menghitung *output* berdasarkan bobot, bias, dan fungsi aktivasi.

   - Kombinasi *Linear*

     Pada setiap *neuron*, dilakukan operasi kombinasi *linear* antara input, bobot, dan bias untuk menghasilkan nilai awal (z).

     $z^{\{(l)\}} = W^{\{(l)\}a^{\{(l-1)\}}} + b^{\{(l)\}}$ .....(2.2)

     Keterangan: z merupakan hasil dari proses kombinasi *linear* antara input dengan bobot dan bias, di mana *W* adalah bobot yang menghubungkan antar *neuron*, *a* adalah *output* dari *layer* sebelumnya, dan b adalah bias yang ditambahkan pada proses perhitungan.

   - Fungsi aktivasi

     Nilai hasil kombinasi *linear* kemudian ditransformasikan menggunakan fungsi aktivasi untuk memperkenalkan sifat *nonlinier* dalam model.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

$$a^{\{(l)\}} = f\left(z^{\{(l)\}}\right) \quad \text{.....(2.3)}$$

Salah satu fungsi aktivasi yang umum digunakan adalah *Rectified Linear Unit* (ReLU), yang didefinisikan sebagai:

$$f(x) = max(0, x) \quad \text{.....(2.4)}$$

Fungsi ReLU akan menghasilkan nilai nol jika input bernilai negatif, dan mempertahankan nilai input jika bernilai positif. Hal ini membantu model dalam mempelajari pola kompleks secara lebih efisien.

Keterangan: a merupakan *output* yang dihasilkan setelah melalui fungsi aktivasi, f adalah fungsi aktivasi yang digunakan untuk memberikan sifat *nonlinier* pada model, dan z adalah nilai hasil kombinasi *linear* sebelum dilakukan aktivasi.

3. *Output Layer*

   Output layer berfungsi untuk menghasilkan prediksi akhir dari model. Pada kasus klasifikasi multikelas, *output layer* biasanya menggunakan fungsi aktivasi *softmax* untuk menghasilkan probabilitas dari setiap kelas. Kelas dengan nilai probabilitas tertinggi akan dipilih sebagai hasil prediksi.

$$y_i = \frac{e^{zi}}{\sum_{j=1}^{k} e^{zj}} \quad \text{.....(2.5)}$$

   Keterangan: y topi i merupakan probabilitas untuk setiap kelas ke-i yang dihasilkan oleh model, z adalah nilai input pada *output layer*, dan k menunjukkan jumlah kelas yang digunakan dalam proses klasifikasi.

4. Perhitungan Error (*Loss Function*)

   Setelah diperoleh *output* prediksi, langkah selanjutnya adalah menghitung *error* menggunakan fungsi *loss* untuk mengukur selisih antara nilai prediksi dan nilai aktual.

$$L = -\sum_{\{i=1\}_i^{\{n\}y}} \backslash log(\hat{}\{y\}_i) \quad \text{.....(2.6)}$$

   Keterangan: L merupakan nilai *loss* yang digunakan untuk mengukur kesalahan prediksi model, yi adalah nilai label sebenarnya dari data, dan y topi i adalah nilai hasil prediksi yang dihasilkan oleh model.

5. *Backpropagation*

   Backpropagation merupakan proses untuk menghitung *gradien error* terhadap setiap bobot dalam jaringan. *Gradien* ini digunakan untuk mengetahui arah perubahan bobot agar *error* dapat diminimalkan.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

$$\frac{\partial L}{\partial w} \quad \text{.....(2.7)}$$

Keterangan: turunan L terhadap w merupakan *gradien* yang menunjukkan seberapa besar perubahan nilai *loss* terhadap perubahan bobot, sedangkan w adalah bobot yang digunakan dalam jaringan.

6. Update Bobot

Bobot jaringan diperbarui menggunakan algoritma optimasi seperti *Gradien*t *Descent* atau Adam dengan tujuan meminimalkan nilai *loss*.

$$w = w - \eta \frac{\partial L}{\partial w} \quad \text{.....2.8)}$$

Keterangan: w merupakan bobot yang akan diperbarui selama proses pelatihan, eta adalah *learning rate* yang mengatur besar langkah perubahan bobot, dan *gradien* digunakan sebagai acuan untuk memperbaiki nilai bobot agar *error* semakin kecil.

Selain tahapan di atas, terdapat beberapa *Activation Function* yang umum digunakan dalam arsitektur DNN, yaitu [32]:

1. *Rectified Linear Unit* (ReLU) adalah fungsi aktivasi *nonlinear* yang hanya mengaktifkan *neuron* ketika *output* dari transformasi linier lebih besar dari nol. Dengan kata lain, *neuron* hanya "aktif" jika hasilnya positif. Fungsi ini membantu jaringan saraf fokus pada informasi yang relevan.
2. *Softmax* digunakan untuk menghitung nilai probabilitas untuk setiap kelas dalam tugas klasifikasi. Probabilitas yang dihasilkan berada pada rentang 0 hingga 1, dimana kelas dengan probabilitas tertinggi menjadi kelas yang diprediksi sebagai *output*. Proses ini adalah langkah umum yang digunakan dalam berbagai model DNN untuk klasifikasi. Fungsi ini digunakan sebagai fungsi aktivasi pada lapisan terakhir untuk klasifikasi multikelas.
3. *Sigmoid* merupakan fungsi aktivasi yang diaplikasikan dalam arsitektur DNN pada tugas klasifikasi CVD. Fungsi ini mentransformasikan *output* menjadi nilai dengan rentang 0 hingga 1, sehingga memudahkan interpretasi probabilitas terhadap suatu kelas. *Output* yang mendekati 1 merepresentasikan probabilitas tinggi untuk kelas positif, sedangkan *output* mendekati 0 menunjukkan probabilitas tinggi untuk kelas negatif. Penggunaan fungsi *sigmoid* lebih optimal pada lapisan akhir dalam klasifikasi biner.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

Performa model DNN juga dipengaruhi oleh beberapa *Hyperparameter* yang ditentukan sebelum proses pelatihan di mulai, berikut *hyperparamater* yang digunakan [32]:

1. *Learning Rate*

   Learning rate adalah *hyperparamater* kunci dalam proses *training* yang menentukan seberapa besar bobot model diperbarui setiap iterasi. Pemilihan *learning rate* yang tidak sesuai dapat berdampak besar: jika terlalu kecil maka proses pelatihan akan berjalan lambat dan jika terlalu besar maka pelatihan bisa menjadi tidak stabil dan konvergensi model terganggu.

2. *Epoch*

   Epoch merupakan satu putaran penuh ketika seluruh *dataset* yang digunakan sebagai *input* telah diproses sepenuhnya oleh model melalui proses *forward propagation* dan *backward propagation*. Dalam satu *epoch* data melewati seluruh lapisan jaringan saraf untuk dilakukan pembelajaran secara menyeluruh sebelum kemudian diulang pada siklus berikutnya. Selama satu *epoch*, *dataset* pelatihan biasanya dibagi menjadi beberapa *batch* (kelompok data yang lebih kecil). Model akan memproses satu *batch* pada satu waktu, menghitung *gradien*, dan melakukan pembaruan bobot. Setelah semua *batch* selesai diproses, satu *epoch* dinyatakan selesai. Jumlah *batch* dalam satu *epoch* tergantung pada ukuran *batch* yang ditentukan sebagai *hyperparamater*. Penentuan jumlah *epoch* sangat memengaruhi kinerja model. Jumlah *epoch* yang terlalu sedikit dapat menyebabkan model belum mencapai konvergensi atau belum optimal sedangkan jumlah *epoch* yang terlalu banyak dapat menyebabkan *overfitting*, yaitu kondisi di mana model terlalu mempelajari data pelatihan sehingga performa pada data baru menurun. Oleh karena itu, pemilihan jumlah *epoch* perlu dilakukan secara hati-hati dan biasanya dipantau melalui nilai *loss* atau akurasi pada data validasi.

3. *Batch size*

   Batch size adalah istilah dalam pembelajaran mesin yang mengacu pada jumlah data pelatihan yang diproses secara bersamaan dalam satu iterasi. Nilai ini termasuk salah satu *hyperparamater* penting yang perlu disesuaikan untuk mendukung performa pelatihan model secara optimal. Pemilihan ukuran *batch* tidak bisa dilakukan secara sembarangan karena jumlah data pelatihan yang digunakan turut memengaruhi penentuan *batch size* yang optimal. Ukuran *batch* yang terlalu besar memang dapat mempercepat proses konvergensi namun sering kali menghasilkan solusi akhir yang kurang optimal sedangkan penggunaan *batch size* yang terlalu kecil dapat

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

menyebabkan proses pelatihan menjadi lebih *noisy* karena model mungkin gagal menangkap pola dan variasi distribusi data secara menyeluruh. *Batch size* biasanya dipilih dalam kelipatan dua, misalnya 32, 64, 128, hingga 256 karena sistem perangkat keras dan memori cenderung bekerja lebih efisien dengan ukuran *array* berbasis pangkat dua. Oleh karena itu pemilihan *batch size* yang tepat menjadi salah satu faktor kunci dalam mengoptimalkan performa model dan durasi pelatihan.

4. *Optimizer*

Optimizer dalam kecerdasan buatan adalah proses penting yang bertujuan untuk menyesuaikan parameter model seperti bobot dan bias agar menurunkan nilai fungsi kerugian. Dengan pendekatan ini maka proses pelatihan model menjadi lebih efisien karena memungkinkan permbaruan bobot dan penyesuaian *learning rate* secara dinamis pada jaringan saraf sehingga kesalahan prediksi dapat di kurangi. Penggunaan algoritma *Optimizer* yang tepat sangat berpengaruh terhadap performa akhir model. Salah satu algoritma yang banyak digunakan adalah Adam (*Adaptive Moment Estimation)* karena kemampuannya dalam mempercepat konvergensi dan menangani data dengan *noise*. Namun Adam memiliki keterbatasan dalam regularisasi bobot khususnya terkait implementasi *weight decay* yang tercampur dengan *update parameter*. Untuk mengatasi kelemahan tersebut digunakanlah AdamW yaitu varian dari Adam yang memisahkan *eksplisit* proses *weight decay* dari perhitungan *gradien*. AdamW terbukti mampu menghasilkan pelatihan yang lebih stabil dan akurat. Dalam implementasinya sangat penting untuk menentukan nilai *learning rate* yang optimal.

2.7 *Imbalanced Data* (Ketidakseimbangan Data)

Ketidakseimbangan data (*class imbalanced*) merupakan salah satu masalah yang sering muncul, di mana distribusi jumlah data pada setiap kelas tidak seimbang dan kelas mayoritas memiliki jumlah sampel yang jauh lebih besar dibandingkan kelas minoritas sehingga menyebabkan model *machine learning* cenderung bias terhadap kelas mayoritas dan menurunkan kemampuan dalam mendeteksi kelas minoritas [33].Pada *dataset* CVD yang digunakan dalam penelitian ini, ketidakseimbangan data terlihat dari perbedaan jumlah data pada setiap kategori risiko, yaitu *Low*, *Intermediary*, dan *High* [8]. Kondisi ini menyebabkan salah satu kelas menjadi dominan sementara kelas lainnya kurang terwakili, sehingga distribusi data menjadi tidak seimbang dan kelas minoritas sering diabaikan oleh algoritma pembelajaran mesin karena dominasi kelas mayoritas. Selain itu, ketidakseimbangan data pada *dataset* CVD juga terbukti menghambat kemampuan model

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

dalam melakukan prediksi yang akurat terhadap kasus minoritas, sehingga berdampak pada penurunan kualitas diagnosis secara keseluruhan [34].

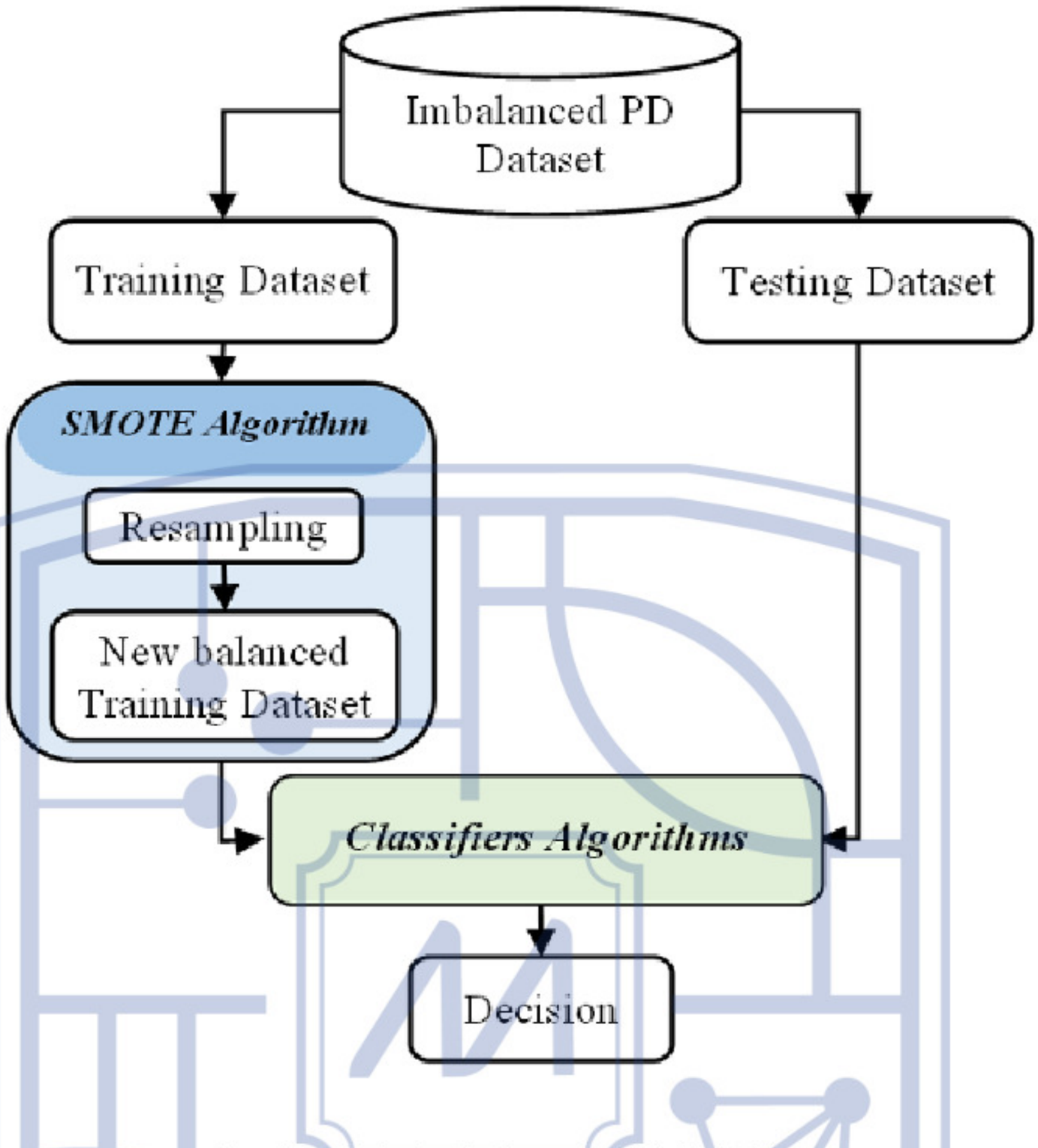
Model yang dilatih pada *dataset* yang tidak seimbang cenderung menghasilkan prediksi yang kurang akurat terutama pada kelas minoritas sehingga menjadi permasalahan penting dalam klasifikasi risiko penyakit karena dapat memengaruhi pengambilan keputusan, namun berbagai pendekatan seperti penyeimbangan data menggunakan teknik *oversampling* berbasis data sintetis telah dikembangkan untuk meningkatkan representasi kelas minoritas, menyeimbangkan distribusi data, serta meningkatkan kinerja model dan kemampuan *generalisasi* dalam proses klasifikasi [35].

Pada teknik *resampling*, proses difokuskan pada pemanfaatan dan pembangkitan ulang sampel dari data yang sudah ada untuk mengatasi ketidakseimbangan kelas, di mana metode SMOTE menghasilkan sampel sintetis melalui interpolasi antar data minoritas menggunakan konsep *k-nearest neighbors* sehingga mampu menciptakan distribusi data yang lebih seimbang tanpa sekadar duplikasi, dan berbagai penelitian menunjukkan bahwa pendekatan berbasis SMOTE mampu meningkatkan performa model dibandingkan teknik *resampling* lain dalam beberapa skenario klasifikasi [36].

2.8 *Synthetic Minority Oversampling Technique* (SMOTE)

SMOTE merupakan teknik *oversampling* yang bekerja dengan cara menghasilkan data sintetis baru pada kelas minoritas berdasarkan kedekatan antar data dalam ruang fitur. Berbeda dengan metode *oversampling* konvensional yang hanya melakukan duplikasi data, SMOTE membentuk sampel baru melalui proses interpolasi antara data minoritas dan tetangga terdekatnya, sehingga menghasilkan distribusi data yang lebih seimbang dan representatif, serta mengurangi risiko *overfitting* akibat penggandaan data yang sama, sehingga model dapat mempelajari pola pada kelas minoritas secara lebih optimal [37].

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.



Gambar 2.4 Arsitektur SMOTE [38]

Gambar 2.4 menunjukkan alur penerapan metode SMOTE dalam proses klasifikasi pada *dataset* yang tidak seimbang (*imbalanced dataset*). Proses dimulai dari *dataset* awal yang memiliki distribusi kelas tidak seimbang, kemudian *dataset* tersebut dibagi menjadi dua bagian yaitu data pelatihan (*training dataset*) dan data pengujian (*testing dataset*). Selanjutnya, metode SMOTE diterapkan hanya pada data pelatihan melalui proses *resampling* untuk menghasilkan data sintetis pada kelas minoritas sehingga terbentuk *dataset* pelatihan yang lebih seimbang (*balanced training dataset*). Setelah itu, *dataset* yang telah diseimbangkan digunakan dalam proses pelatihan model menggunakan algoritma klasifikasi, sementara data pengujian yang tidak mengalami perubahan digunakan untuk menguji performa model agar tetap merepresentasikan kondisi data sebenarnya. Tahap akhir dari proses ini adalah menghasilkan keputusan (*decision*) berdasarkan hasil klasifikasi yang diperoleh, di mana penerapan SMOTE diharapkan mampu meningkatkan kinerja model dalam menangani permasalahan ketidakseimbangan data [38].

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

SMOTE merupakan salah satu teknik oversampling yang digunakan untuk mengatasi permasalahan ketidakseimbangan data (*imbalanced dataset*) dalam proses klasifikasi. Ketidakseimbangan data terjadi ketika jumlah data pada kelas mayoritas jauh lebih besar dibandingkan kelas minoritas, sehingga model cenderung bias terhadap kelas mayoritas dan mengabaikan pola penting pada kelas minoritas. Untuk mengatasi permasalahan tersebut, SMOTE menghasilkan data sintetis baru pada kelas minoritas berdasarkan kedekatan antar data dalam ruang fitur, sehingga distribusi data menjadi lebih seimbang tanpa melakukan duplikasi data secara langsung. Lebih lanjut, proses penerapan SMOTE tidak hanya terbatas pada tahap *resampling*, tetapi juga melibatkan serangkaian tahapan sistematis dalam pembangkitan data sintetis yang bertujuan untuk memastikan bahwa data yang dihasilkan tetap merepresentasikan karakteristik distribusi asli kelas minoritas. Oleh karena itu, untuk memberikan pemahaman yang lebih mendalam mengenai mekanisme kerja SMOTE, tahapan-tahapan penerapannya perlu dijelaskan secara rinci [39],[40],[12]:

1. Identifikasi ketidakseimbangan data

   Tahap awal dalam penerapan SMOTE adalah melakukan analisis distribusi kelas pada *dataset*. Ketidakseimbangan terjadi ketika proporsi jumlah data antar kelas tidak seimbang secara signifikan, misalnya perbandingan 80:20 antara kelas mayoritas dan minoritas. Kondisi ini dapat menyebabkan model klasifikasi memiliki kecenderungan untuk memprediksi kelas mayoritas karena jumlah datanya lebih dominan, sehingga performa model terhadap kelas minoritas menjadi rendah, terutama pada metrik seperti *recall* dan *F1-score* .

2. Pembagian *dataset* (*training* dan *testing*)

   dataset kemudian dibagi menjadi dua bagian utama, yaitu data pelatihan (*training*) dan data pengujian (*testing*). Penerapan SMOTE hanya dilakukan pada data pelatihan, sedangkan data pengujian dibiarkan dalam kondisi asli tanpa perubahan. Hal ini bertujuan untuk menghindari terjadinya *data leakage*, yaitu kondisi di mana informasi dari data pengujian secara tidak sengaja digunakan dalam proses pelatihan, sehingga dapat menyebabkan hasil evaluasi model menjadi tidak valid atau terlalu optimistis.

3. Pemilihan data kelas minoritas

   Pada tahap ini, seluruh data yang termasuk dalam kelas minoritas diidentifikasi dan dipisahkan dari *dataset*. Data minoritas ini menjadi fokus utama dalam proses SMOTE karena tujuan utama metode ini adalah menambah jumlah data pada kelas minoritas agar distribusi *dataset* menjadi lebih seimbang. Dengan meningkatkan jumlah data

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

minoritas, model diharapkan dapat mempelajari pola yang lebih representatif dari kelas tersebut

4. Pencarian tetangga terdekat (K-*Nearest Neighbors*)

   Setiap data pada kelas minoritas akan dicari sejumlah *k* tetangga terdekatnya menggunakan metode K-*Nearest Neighbors* (KNN) berdasarkan jarak dalam ruang fitur (biasanya menggunakan *Euclidean distance*). Nilai *k* ditentukan sebagai parameter (misalnya k = 5). Proses ini bertujuan untuk menemukan data yang memiliki karakteristik paling mirip sehingga dapat digunakan sebagai dasar dalam pembangkitan data sintetis. Kedekatan antar data menunjukkan kemiripan pola yang penting untuk menjaga konsistensi distribusi data baru yang akan dihasilkan.

5. Pemilihan tetangga secara acak

   Dari kumpulan *k* tetangga terdekat yang telah diperoleh, satu atau lebih tetangga dipilih secara acak. Pemilihan secara acak ini bertujuan untuk meningkatkan variasi data sintetis yang dihasilkan, sehingga tidak hanya terbentuk pola yang seragam. Variasi ini penting agar model tidak mengalami *overfitting* terhadap data tertentu dan tetap mampu melakukan generalisasi terhadap data baru

6. Pembangkitan data sintetis

   Tahap inti dari SMOTE adalah proses pembangkitan data sintetis menggunakan teknik interpolasi antara data minoritas dan tetangga terdekatnya. Proses ini dinyatakan dengan persamaan berikut:

   $$x_{\{baru\}} = x_i + \lambda\left(x_{\{tetangga\}} - x_i\right), \backslash quad\ 0 < \lambda < 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.9)$$

   Keterangan: x_i menyatakan data minoritas asli yang menjadi titik awal dalam proses pembangkitan data sintetis, sedangkan $x_{\{tetangga\}}$ merupakan data dari tetangga terdekat yang dipilih berdasarkan kedekatan jarak dalam ruang fitur. Selanjutnya, λ adalah nilai acak dalam rentang 0 hingga 1 yang digunakan untuk mengatur *proporsi* interpolasi antara kedua data tersebut, dan $x_{\{baru\}}$ merupakan data sintetis yang dihasilkan dari proses kombinasi antara data minoritas asli dengan data tetangga terdekat. Melalui persamaan tersebut, data baru dibentuk di antara dua titik data dalam ruang fitur. Hal ini membuat data sintetis tetap berada dalam distribusi yang sama dengan data asli, sehingga lebih representatif dibandingkan metode duplikasi data. Pendekatan ini juga membantu mengurangi risiko *overfitting* karena data yang dihasilkan memiliki variasi baru.

7. Pengulangan proses *oversampling*

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

Melalui persamaan tersebut, data baru dibentuk di antara dua titik data dalam ruang fitur. Hal ini membuat data sintetis tetap berada dalam distribusi yang sama dengan data asli, sehingga lebih representatif dibandingkan metode duplikasi data. Pendekatan ini juga membantu mengurangi risiko *overfitting* karena data yang dihasilkan memiliki variasi baru.

8. Pembentukan *dataset* seimbang

   Setelah proses *oversampling* selesai, akan diperoleh *dataset* pelatihan yang memiliki distribusi kelas yang lebih seimbang. *dataset* ini memungkinkan model untuk mempelajari pola dari kedua kelas secara lebih adil dan tidak bias terhadap kelas mayoritas. *dataset* yang seimbang juga berkontribusi pada peningkatan performa model, terutama pada kemampuan mendeteksi kelas minoritas.

9. Pelatihan Model (*Training*)

   dataset yang telah diseimbangkan kemudian digunakan dalam proses pelatihan model klasifikasi, seperti DNN. Dengan distribusi data yang lebih seimbang, model dapat mempelajari hubungan antar fitur secara lebih optimal, termasuk pola yang sebelumnya sulit dikenali pada kelas minoritas. Hal ini berdampak pada peningkatan akurasi dan *stabilitas* model dalam melakukan prediksi.

10. Evaluasi Model (*Testing*)

    Tahap terakhir adalah evaluasi model menggunakan data pengujian yang tidak mengalami proses SMOTE. Evaluasi dilakukan menggunakan metrik seperti *accuracy*, *precision*, *recall*, dan F1-*score*. Penggunaan data asli pada tahap pengujian bertujuan untuk memastikan bahwa model benar-benar mampu melakukan generalisasi terhadap data baru dan tidak hanya menghafal data hasil *oversampling*.

2.9 Evaluasi Model

Evaluasi model klasifikasi tidak hanya berfokus pada nilai performa secara keseluruhan, tetapi juga perlu mempertimbangkan distribusi prediksi benar dan salah pada setiap kelas, sehingga diperlukan suatu metode evaluasi yang mampu merepresentasikan hal tersebut secara komprehensif. *Confusion matrix* merupakan metode yang digunakan untuk menampilkan sekaligus mengevaluasi hasil prediksi model, sehingga dapat membantu dalam menganalisis tingkat akurasi serta kesalahan prediksi yang dihasilkan. Dengan menggunakan alat ini, kelebihan dan keterbatasan model dalam melakukan klasifikasi dapat diidentifikasi secara lebih objektif. Pada kasus klasifikasi multikelas, *confusion matrix* tidak hanya menampilkan istilah TP, FP, FN, dan TN secara umum

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

seperti pada klasifikasi biner, melainkan setiap kelas memiliki nilai TP, FP, FN, dan TN masing-masing yang ditentukan berdasarkan pendekatan *one-vs-all* [41].

Tabel 2.1 *Confusion Matrix* [41]

Data Aktual	**Data Prediksi**		
	HIGH	*INTERMEDIARY*	*LOW*
HIGH	a	b	c
INTERMEDIARY	d	e	f
LOW	g	h	i

Keterangan:

1. Matriks di atas menunjukkan hasil perbandingan antara data aktual dan data hasil klasifikasi (prediksi) yang dihasilkan oleh model. Baris merepresentasikan kelas aktual, sedangkan kolom menunjukkan kelas hasil prediksi.
2. Elemen pada diagonal utama yaitu a,e,i, merupakan jumlah data yang diklasifikasikan dengan benar oleh model untuk masing-masing kelas, sehingga disebut sebagai *True Positive* (TP) pada setiap kelas.
3. Elemen di luar diagonal utama (b,c,d,f,g,h) menunjukkan jumlah kesalahan klasifikasi, yaitu kondisi ketika data diprediksi ke kelas yang tidak sesuai dengan label aslinya.
4. Dalam klasifikasi *multiclass*, evaluasi kinerja model dilakukan menggunakan pendekatan *one-vs-rest*, di mana setiap kelas secara bergantian dianggap sebagai kelas positif, sedangkan kelas lainnya dianggap sebagai kelas negatif.

   Dalam konteks *multiclass*, perhitungan dilakukan untuk setiap kelas dengan pendekatan *one-vs-all*, sebagai berikut:

1. Kelas *HIGH*
   - *True Positive* (TP) = a
   - *False Negative* (FN) = b+c
   - *False Positive* (FP) = d+g
   - *True Negative* (TN) = e+f+h+i
2. Kelas *INTERMEDIARY*
   - *True Positive* (TP) =e
   - *False Negative* (FN) = d+f
   - *False Positive* (FP) = b+h
   - *True Negative* (TN) = a+c+g+i

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

3. Kelas *LOW*

   – *True Positive* (TP) = i
   – *False Negative* (FN) = g+h
   – *False Positive* (FP) =c+f
   – *True Negative* (TN) = a+b+d+e

Berdasarkan hasil perhitungan *confusion matrix*, selanjutnya dilakukan evaluasi performa model menggunakan metrik *accuracy*, *precision*, *recall*, dan *F1-score* untuk memperoleh gambaran kinerja model secara menyeluruh.

Precision mengukur tingkat ketepatan model dalam memprediksi suatu kelas tertentu. Metrik ini menunjukkan seberapa banyak data yang diprediksi sebagai positif benar-benar merupakan data positif. Berikut adalah persamaan nya:

$$Precision = \frac{TP}{TP + FP} \quad \text{.....(2.10)}$$

Recall mengukur kemampuan model dalam mendeteksi seluruh data yang termasuk dalam suatu kelas. Metrik ini menunjukkan seberapa banyak data positif yang berhasil diidentifikasi dengan benar oleh model. Berikut adalah persamaannya:

$$Recall = \frac{TP}{TP + FN} \quad \text{.....(2.11)}$$

F1-score merupakan rata-rata harmonik antara *precision* dan *recall*. Metrik ini digunakan untuk memberikan keseimbangan antara *precision* dan *recall*, terutama ketika terdapat ketidakseimbangan data. Berikut adalah persamaannya:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad \text{.....(2.12)}$$

Accuracy merupakan metrik yang digunakan untuk mengukur tingkat ketepatan model secara keseluruhan, yaitu proporsi jumlah prediksi yang benar dibandingkan dengan total seluruh data. Nilai *accuracy* yang tinggi menunjukkan bahwa model mampu melakukan klasifikasi dengan baik. Berikut adalah persamaannya:

$$Accuracy = \frac{TP\ (a, e, i)}{Total\ data\ (a + b + c + d + e + f + g + h + i)} \times\ 100\% \quad \text{.....(2.13)}$$

2.10 Penelitian Terdahulu

Penelitian mengenai klasifikasi penyakit kardiovaskular telah banyak dilakukan dengan menggunakan metode *machine learning* dan *deep learning* untuk meningkatkan

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

performa prediksi risiko penyakit. Beberapa penelitian juga menunjukkan bahwa metode DNN mampu mengenali pola hubungan nonlinier pada data medis dengan hasil klasifikasi yang cukup baik. Selain itu, penerapan teknik SMOTE dinilai efektif dalam mengatasi permasalahan ketidakseimbangan data sehingga performa model klasifikasi dapat meningkat. Oleh karena itu, penelitian-penelitian terdahulu yang berkaitan dengan metode klasifikasi, penggunaan DNN, dan penerapan SMOTE perlu dikaji sebagai dasar pendukung dalam penelitian ini [5],[6],[9].

Tabel 2.2 Penelitian Terdahulu Mengenai CVD

No.	Judul, Peneliti, Tahun	Metode	*Dataset* yang digunakan	Hasil/Temuan
1.	***Auxiliary Prediction of Cardiovascular Diseases Based on Deep Learning***, Zhongzheng Zhang, Junnan Zhu, 2024	KNN, *Random Forest* (RF), XGBoost, GBDT, dan *Deep Neural Network* (DNN) + SMOTE untuk menangani *imbalance*	*Dataset* dari Kaggle, 70.000 data, 11 fitur + 1 target (data objektif, pemeriksaan, subjektif)	Model terbaik adalah DNN dengan *Accuracy* 85% dan AUC 83.2% setelah data *augmentation*. SMOTE terbukti meningkatkan performa model, dan DNN menunjukkan kinerja paling baik dibanding metode lain.
2.	***Detection of Cardiovascular Disease using Machine Learning Classification Models***, Hana H. Alalawi, Manal S. Alsuwat, 2021	SVM, *Decision Tree, Logistic Regression,* KNN, Naïve Bayes, *Random Forest,* ANN, *Voting Classifier*, *Gradien*t	2 *dataset*: *Cardiovascular Disease Dataset* & *Heart Disease Dataset* (IEEE *Dataport*, ±1190 data, 10 fitur + 1 target)	Random Forest memberikan hasil terbaik pada *Heart Disease Dataset* dengan akurasi 94%, sedangkan *Gradien*t *Boosting* terbaik pada CVD *dataset* dengan

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

		Boosting		akurasi sekitar 70–74%. Model ML efektif untuk membantu diagnosis dini penyakit jantung.
3.	***Improving Cardiovascular Disease Prediction With Deep Learning and Correlation-Aware SMOTE***, Maria Trigka, Elias Dritsas, 2025	*Deep Learning* (MLP, CNN, RNN, LSTM, *Autoencoder*) + SMOTE dan *Enhanced (Correlation-Aware)* SMOTE	*Dataset* CVD dengan 20.000 data, 11 fitur (usia, tekanan darah, kolesterol, glukosa, BMI, dll) + 1 target	*Enhanced* SMOTE meningkatkan performa model secara signifikan dibanding tanpa SMOTE dan SMOTE biasa. Model terbaik adalah CNN dengan *Accuracy* 91%, AUC 0.90, *Precision* 0.89, *Recall* 0.86, dan *F1-score* 0.87.
4.	***A Computational Intelligence Approach Using SMOTE and Deep Neural Network (DNN)***, Madhusmita Sahu, Rasmita Dash, 2022	*Deep Neural Network* (DNN) yang dikombinasikan dengan SMOTE untuk mengatasi data imbalance	*Dataset* ASTER *(Advanced Spaceborne Thermal Emission and Reflection Radiometer)*, data tutupan hutan di Jepang dengan 4 kelas jenis pohon	Kombinasi SMOTE + DNN mampu meningkatkan performa klasifikasi dan mengatasi ketidakseimbangan data. Model mencapai akurasi sebesar 94% serta efektif untuk identifikasi jenis

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

				hutan dan pemetaan wilayah hutan skala besar.
5.	***Effective Class-Imbalance Learning Based on SMOTE and Convolutional Neural Networks***, Javad Hassannataj Joloudari, Abdolreza Marefat, Mohammad Ali Nematollahi, Solomon Sunday Oyelere, Sadiq Hussain, 2023	*Convolutional Neural Network* (CNN) yang dikombinasikan dengan SMOTE, *oversampling, undersampling*, serta metode utama SMOTE + *Normalization* + CNN	24 *dataset imbalanced* dari KEEL *Repository*, termasuk *Breast Cancer Dataset* dan Z-Alizadeh Sani *Dataset*	Model SMOTE + *Normalization* + CNN menunjukkan performa terbaik dengan akurasi hingga 99,08%. Kombinasi SMOTE dan CNN terbukti lebih efektif dibanding metode lain serta cocok untuk klasifikasi biner dengan data tidak seimbang. Eksperimen dilakukan sebanyak 100 kali untuk memastikan kestabilan hasil.

Penelitian terdahulu pada Tabel 2.2 menunjukkan bahwa metode *machine learning* dan *deep learning* telah banyak digunakan dalam klasifikasi penyakit kardiovaskular. Beberapa algoritma yang sering digunakan antara lain SVM, *Random Forest*, *Decision Tree*, CNN, dan DNN. Selain itu, beberapa penelitian juga menerapkan teknik SMOTE untuk mengatasi ketidakseimbangan data (*imbalanced data*) sehingga performa model klasifikasi dapat meningkat.

Berdasarkan penelitian terdahulu, kombinasi metode *deep learning* dengan teknik SMOTE menunjukkan hasil yang cukup baik dalam meningkatkan akurasi klasifikasi. Model DNN dan CNN menjadi metode yang paling banyak digunakan karena mampu

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

mempelajari pola data secara lebih kompleks dibandingkan metode *machine learning* tradisional. Selain itu, penggunaan teknik *oversampling* seperti SMOTE terbukti efektif dalam meningkatkan kemampuan model untuk mengenali kelas minoritas pada *dataset* yang tidak seimbang.

Meskipun demikian, penelitian yang secara khusus mengombinasikan DNN dengan SMOTE pada klasifikasi penyakit kardiovaskular masih relatif terbatas. Oleh karena itu, penelitian ini dilakukan untuk menganalisis performa metode DNN dengan penerapan SMOTE dalam meningkatkan hasil klasifikasi penyakit kardiovaskular.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.