BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Berdasarkan sumber pustaka yang digunakan berasal dari jurnal-jurnal terkait dengan penelitian sebelumnya sebagai referensi guna menjadi panduan maupun pendukung dalam melakukan penulisan ini. Adapun sebagai sumber pustaka tersebut yaitu sebagai berikut:

Amelia *et al*. (2022) menyampaikan Implementasi Algoritma *Support Vector Machine* (SVM) Untuk Prediksi Penyakit Stroke Dengan Atribut Berpengaruh, dalam penelitian ini dilakukan untuk memprediksi adanya penyakit Strok dengan menggunakan Algoritma *Support Vector Machine* (SVM) yang digunakan untuk mengklasifikasikan himpunan data menggunakan Metode *Confusion Matrix*. Pengujian algoritma SVM ini menggunakan Kernel *linear* untuk mendapatkan hasil terbaik. algoritma *Support Vector Machine* (SVM) dengan *Relief-f*. Data yang menggunakan 3.426 Baris dan lima kolom. Dengan hasil pengujian mendapatkan akurasi data sebesar 100%.

Wafa *et al.*(2022) dalam penelitiannya "Prediksi Penyakit Diabetes Menggunakan Algoritma *Support Vector Machine* (SVM)" mengungkapkan bahwa hasil yang didapatkan dari penelitiannya tersebut dengan menggunakan metode *algoritma Support Vektor Machine Radial Basis Function (RBF)* mendapatkan hasil akurasi sebesar 91%. Pengujiannya menggunakan *Confusion Matrix* dan peramalan *Mean Square Error* dengan menggunakan *K-fold* berkelipatan 10. Jumlah record yang digunakan sebanyak 1.017 data dengan 8 atribut. Dalam

penelitiannya tersebut bertujuan untuk menentukan apakah seseorang dapat terkena penyakit diabetes atau tidak.

Dalam penelitian yang dilakukan oleh Hasanah & Nurmalitasari. (2023) yang berjudul "Perbandingan Tingkat Akurasi Algoritma *Support Vector Machines* (SVM) dan C45 dalam Prediksi Penyakit Jantung". Menyebutkan bahwa hasil dari penelitian yang telah dilakukan dengan menggunakan metode algoritma *Support Vektor Machine* (SVM) dan algoritma C.45 mendapatkan hasil perbandingan akurasi sebesar 87% pada metode algoritma *Support Vektor Machine* (SVM) sedangkan untuk hasil akurasi menggunakan metode algoritma C.45 mendapatkan hasil akurasi sebesar 82%. Dengan penggunaan data sebanyak 1.190 dataset dan terdapat 12 atribut.

Penelitian yang dilakukan oleh Tino, et al., (2023) akurasi yang didapatkan dari algoritma SVM lebih tinggi dibandingkan dengan *Neural Network*, dilihat dari hasil *AUC* 0.833, *CA* 0.83, *F1* 0.835, *Accuracy* 0.835 serta *Review* 0.835. Dari percobaan kedua model tersebut, model SVM memiliki hasil lebih tinggi dengan melakukan teknik klasifikasi yang menghasilkan akurasi sebesar 83%, sedangkan untuk algoritma *Neural Network* lebih berguna untuk menyelesaikan masalah yang berkaitan penyakit jantung dengan nilai akurasi yang didapatkan sebesar 82%.

Selanjutnya, penelitian yang pernah dilakukan oleh Lumbanraja,*et al*.(2022) memaparkan penelitiannya yang berjudul "Implementasi *Support Vector Machine* (Svm) Untuk Klasifikasi Pederita *Diabetes Mellit*us". Dengan menggunakan dataset sebanyak 101.766, dari data tersebut dilakukan proses pembersihan data sehingga data yang didapatkan setelah pembersihan menjadi 84.900. Hasil dari

penelitiannya untuk proses klasifikasi penderita *Diabetes Mellitus* yang menggunakan metode *Support Vektor Machine* (SVM) didapatkan hasil rata-rata akurasi terbesar didapatkan dari *kernel Gaussian,* yaitu sebesar 82,76%, hasil rata-rata *sensitivity* sebesar 27.47% dan hasil *specificity* sebesar 76.02%. Penelitian terdahulu dapat dilihat pada Tabel 2.1.

Tabel 2. 1 Penelitian terdahulu

NO	PENELITI	TOPIK	METODE	HASIL
1.	Amelia et al. (2022)	IMPLEMENTASI ALGORITMA *SUPPORT VECTOR MACHINE* (SVM) UNTUK PREDIKSI PENYAKIT STROKE DENGAN ATRIBUT BERPENGARUH	Algoritma Support Vektor Machine (SVM) dengan *Relief-f,* menggunakan kernel *linier*.	Hasil yang disimpulkan pada penelitian prediksi penyakit Strok dengan menggunakan Algoritma SVM adalah model prediksi yang dibangun menggunakan algoritma SVM dengan *Relief-f*. Menggunakan 2.398 data training dan 1.028 data *testing* menghasilkan akurasi data sebesar 100%. Metode yang digunakan menggunakan Rasio dan *Confusion Matrix* dengan total jumlah data sebanyak 3.426 baris dan lima kolom. Penerapan Algoritma SVM dengan *Relif-f* menggunakan Kernel *Linear* untuk mendapatkan hasil terbaik.
2.	Wafa, et al., (2022)	PREDIKSI PENYAKIT DIABETES MENGGUNAKAN ALGORITMA *SUPPORT VECTOR MACHINE* (SVM)	SVM kernel *radial basis function* (RBF) dan *forward selection.*	Performa yang dihasilkan dari metode *Support Vektor Machine* dengan kernel *radial basis function* dan *forward selection* memperoleh hasil 91.2% untuk *accuracy*, 93.0% untuk *precision*, 94.3% untuk *recall*, dan 93.7% untuk *f1-score*.

3.	Lumbanraja, et al., (2022)	IMPLEMENTASI *SUPPORT VECTOR MACHINE* (SVM) UNTUK KLASIFIKASI PEDERITA *DIABETES MELLIT*US	1. *K-Fold Cross Validation* 2. *support vector machine* untuk klasifikasi penderita *diabetes* menggunakan R *Shiny*. ada 3 *kernel* dari model algoritma SVM, yaitu *Kernel Linear, Kernel Gaussian,* dan *Kernel Polynomial.*	Dalam prediksi untuk klasifikasi penderita diabetes, menunjukkan bahwa hasil dari pengklasifikasian tersebut mendapatkan hasil akurasi sebesar 82,76% dari kernel *Gaussian*. Sedangkan untuk hasil rata-rata *sensitivity* sebanyak 27,47% dan hasil *specificity* sebanyak 76,02%.
4.	Hasanah & Nurmalitasari, (2023)	PERBANDINGAN TINGKAT AKURASI ALGORITMA *SUPPORT VECTOR MACHINE* (SVM) DAN C45 DALAM PREDIKSI PENYAKIT JANTUNG	1. Algoritma *Support Vektor Machine* menggunakan tahap evaluasi AUC, CA, F1, *Precission, Recall, Confussion Matrix* dan ROC analysis. 2. Algoritma C45 menggunakan tahap evaluasi AUC, CA, F1, *Precission, Recall, Confussion Matrix* dan ROC analysis.	Dalam Perbandingan Tingkat Akurasi Algoritma *Support Vector Machines* (SVM) Dan C45 Dalam Prediksi Penyakit Jantung. Mendapatkan hasil akurasi sebesar 87%, AUC 0.93, CA 0.88, F1 0.086, Precision 0.86 dan Recall 0.87 pada metode algoritma *Support Vektor Machine* (SVM), sedangkan untuk hasil akurasi menggunakan metode algoritma C.45 mendapatkan hasil akurasi sebesar 82%, AUC 0.82, CA 0.82, F1 0.81, Precision 0.83 dan Recall 0.79.
5.	Tino, et al., (2023)	PERBANDINGAN ALGORITMA *SUPPORT VECTOR MACHINES (SVM)* DAN *NEURAL NETWORK* UNTUK KLASIFIKASI PENYAKIT JANTUNG	1. Algoritma *Support Vector Machines* (SVM). 2. Algoritma *Neural Network.*	Akurasi yang didapatkan dari algoritma SVM lebih tinggi dibandingkan dengan *Neural Network* , dilihat dari hasil AUC 0.833, CA 0.83, F1 0.835, *Accuracy* 0.835 serta *Review* 0.835. Dari percobaan kedua model tersebut, model SVM memiliki hasil lebih tinggi dengan melakukan teknik klasifikasi yang menghasilkan akurasi sebesar 83%, sedangkan untuk algoritma *Neural Network* lebih berguna untuk menyelesaikan masalah yang berkaitan

				penyakit jantung dengan nilai akurasi yang didapatkan sebesar 82%.
6.	Afifa (2023)	IMPLEMENTASI METODE *SUPPORT VEKTOR MACHINE* (SVM) DALAM PREDIKSI PENYAKIT JANTUNG DENGAN MENGGUNAKAN *PYTHON* 3	*Support Vektor Machine* menggunakan *kernel non-linear, radial basis function* (RBF).	Dalam penelitian ini menghasilkan akurasi sebesar 71.52% atau 72% dengan menggunakan fungsi kernel RBF *non linear*. Didapatkan juga hasil *precision sebesar* 81%, *recall* 67%, dan *f1-Score* 73%.

2.2 Dasar Teori

2.2.1 *Support Vektor Machine* (SVM)

Menurut Munawarah, *et al.* (2016) didalam jurnal Biantong, *et al*.(2019) *Support Vektor machine* (SVM) dikembangkan oleh Boser, Guyon dan Vapnik. SVM pertama kali di representasikan pada tahun 1992 di *Annual Workshop on Computation Learning Theory*. Dasar dari SVM sesungguhnya dapat dijelaskan secara sederhana yaitu dengan berusaha mencari sebuah *hyperplane* terbaik yang berfungsi untuk memisahkan dua buah *class* pada *input space*. *Hyperplane* adalah sebuah garis lurus ataupun bidang mendatar yang memisahkan kelas-kelas.

Menurut Burges didalam jurnal Biantong, *et al.* (2019) proses dari pengklasifikasian data menggunakan *Support Vektor Machine* dapat dibedakan atas 2 berdasarkan *kernel* yang digunakan antara lain:

1) *Support Vektor Machine* (SVM) Kernel *Linear*

Honakan, *et al.* (2018) didalam jurnal Biantong, *et al.*(2019) menjabarkan bahwa kernel *linear* digunakan saat data yang diklasifikasikan dapat dipisahkan

dengan sebuah garis /*hyperplane* dengan mudah dan menggunakan pembagian yang jelas. Perhitungan kernel *linear* dapat didefinisikan pada persamaan:

$$K(x, y) = x \times y \qquad (2.1)$$

Keterangan:

K(x, y) : hasil perhitungan *kernel*

x : data ke-x

y : data ke-y

2) *Support Vektor Machine* (SVM) dengan menggunakan kernel *Non-Linear*

Nugroho, et al. (2003) didalam jurnal Biantong, Furqon & Soebroto (2019) menyebutkan bahwa SVM *non linear* merupakan sebuah cara memisahkan data-data yang sifatnya *non-linear* menjadi *linear* dengan menggunakan sebuah fungsi kernel. Fungsi kernel di metode ini sering disebut juga sebagai kernel *trick*. Kernel *trick* adalah *kernel* yang berfungsi mengelompokkan data dari dimensi rendah ke data dimensi tinggi, ada beberapa macam fungsi kernel *non linear* yang digunakan pada sebuah aplikasi untuk mengatasi masalah seperti:

1. Kernel *Polynomial*

$$K(x, y) = (x \times y + 1)^p \qquad (2.2)$$

2. Kernel *Gaussian RBF*

$$K(x, y) = exp\left(-\gamma \left\| x_i - x_j \right||^2\right) \qquad (2.3)$$

3. Kernel *Sigmoid*

$$K(x, y) = \tanh\left(K_x \times y - \delta\right) \qquad (2.4)$$

Keterangan:

p : pangkat (*degree of*)

σ : nilai *sigma*

δ : nilai *delta*

Langkah-langkah proses pengklasifikasian menggunakan algoritma *Support Vektor Machine* (Munawarah, *et al*., 2016).

7. Menghitung kernel SVM proses pertama yaitu menentukan *dot product* setiap data dengan memasukkan fungsi kernel baik kernel *linear* ataupun *non linear*.
8. Menghitung nilai matriks *Hessian*, fungsi dari matriks ini adalah untuk melakukan indentifikasi optimum relatif fungsi. Untuk menghitung matriks *Hessian*, dengan menggunakan persamaan 2.5 berikut:

$$D_{ij} = y_i \times y_j(K(x_i \times x_j) + \lambda^2 \qquad (2.5)$$

Keterangan:

D_{ij} : elemen matriks *Hessian* ke-ij

y_i : kelas data ke-i

y_i : kelas data ke-j

λ : batas teoritis kelas yang akan diturunkan

9. Proses perhitungan *sequential training* SVM. proses perhitungan *sequential training* SVM memiliki beberapa tahapan yaitu:
   a. Menghitung nilai *error* (E_i), dengan menggunakan persamaan 2.6 berikut:

$$E_i = \sum_{j=1}^{i} \alpha_i D_{ij} \qquad (2.6)$$

   b. Menghitung nilai *Delta Alpha*

$$\delta\alpha_i = min\{max[\gamma(1 - E_i]; -\alpha_i]; C - \alpha_i\} \quad (2.7)$$

c. Menghitung nilai *alpha* baru

$$new\ \alpha_i = \alpha_i + \delta\alpha_i \quad (2.8)$$

10. Menghitung nilai bobot/*weight* (w) untuk kelas positif dan negatif, proses perhitungannya dapat menggunakan persamaan 2.9 dan 2.10.

$$w \times x^+ = K(x \times y^+) \times \max(\alpha_i{}^+) \times y_i \quad (2.9)$$

$$w \times x^- = K(x \times y^-) \times \max(\alpha_i{}^-) \times y_i \quad (2.10)$$

Keterangan:

$w \times x^+$: bobot kelas positif

$w \times x^-$: bobot kelas negatif

$K(x \times y^+)$: kernel kelas positif

$K(x \times y^-)$: kernel kelas negatif

11. Menghitung nilai *bias* (b), proses menghitung nilai bias dapat menggunakan persamaan 2.11.

$$b = -\frac{1}{2}(w_i^+ + w_i^-) \quad (2.11)$$

12. Proses perhitungan untuk pengujian SVM dapat dilakukan dengan beberapa tahapan yaitu:

a. Perhitungan kernel data *testing* setelah mendapatkan nilai α, w dan b dari proses *training* SVM, langkah pertama yang digunakan dalam proses pengujian tersebut menggunakan perhitungan semua *dot product kernel* antar data *testing* dan data *training* yang berdasarkan hitungan *kernel* yang digunakan pada sebelumnya saat proses *training* SVM.

Langkah selanjutnya adalah melakukan perhitungan $\sum \alpha_i x_i$ ($K(x,x_i)$) terhadap seluruh data *testing* yang menggunakan persamaan 2.12:

$$\sum \alpha_i \times x_i(K(x, x_i)) \quad (2.12)$$

b. Melakukan perhitungan fungsi $f(x)$, perhitungan ini memiliki tujuan untuk mencari nilai dari proses klasifikasi setiap data *testing* dengan menggunakan fungsi *sign*. Perhitungan fungsi klasifikasi dapat menggunakan persamaan 2.13:

$$f(x) = sign(\sum_{i=1}^{m} \alpha_i y_i \times K(x, x_1) + b) \quad (2.13)$$

3) Kelebihan dan kekurangan dari SVM

a) Kelebihan

SVM memiliki beberapa kelebihan dalam penggunaannya antara lain;

1. Memiliki kemampuan untuk generalisasi yang tinggi.
2. Mampu menghasilkan sebuah model klasifikasi yang baik meskipun dilatih dengan menggunakan data yang relatif sedikit (dibandingkan dengan ruang masalah yang harus diselesaikan) hanya dengan sebuah pengaturan parameter yang sederhana.
3. Relatif mudah untuk melakukan implementasi karena penentuan *support vektor* dapat dirumuskan dalam masalah QP (*Quadratic Programming*).

b) Kekurangan

Selain memiliki beberapa kelebihan, SVM juga memiliki beberapa kekurangan antara lain;

1. Sulit untuk mengaplikasikan sebuah himpunan data dengan jumlah sampel dan dimensi yang sangat besar.

2. Hanya untuk formulasi penyelesaian masalah klasifikasi dua kelas. Walaupun dapat digunakan untuk menyelesaikan masalah klasifikasi multi kelas, tetapi masing-masing *multi class* SVM juga mempunyai kekurangan. (Suyanto, 2017).

2.2.2 Data Mining

Menurut Sucahyo (2003) didalam jurnal Biantong, *et al.*(2019) data mining merupakan sebuah metode statistik, kecerdasan buatan dan *Machine Learning* dapat menganalisis secara otomatis data dalam jumlah yang besar dan kompleks untuk mendapatkan suatu pola yang tidak disadari keberadaannya. Data mining juga dikenal dengan nama *Knowledge Discovery in Databases* (KDD). Kemunculan data mining ini dilatar belakangi dengan masalah yang kerap kali dialami pada saat ini yaitu adanya data explosion pada banyak bidang dan organisasi karena data yang terkumpul setiap hari, bulan, atau bahkan tahun.

Didalam jurnal Biantong *et al.* (2019), Kantardsic (2003) mengungkapkan bahwa prosedur yang digunakan pada proses data mining dapat melalui beberapa tahapan seperti:

1) Menentukan masalah dan rumusan hipotesis. Pada tahap ini dilakukan untuk menentukan variabel-variabel terkait dan membuat rumusan hipotesis.
2) Pengumpulan data adalah tahapan pengumpulan data yang terkait dan data yang akan dihasilkan.
3) Preprocessing data merupakan tahapan yang memiliki fungsi pembersihan data (*cleaning*). Pembersihan data dilakukan untuk melakukan pembersihan terhadap *outlier*, penanganan *missing value*, dan transformasi data.

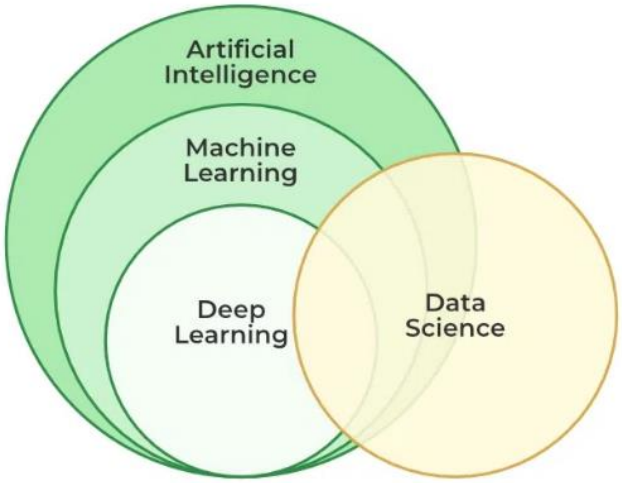
Sehingga data *training* mempunyai rentang yang baik, dan memudahkan dalam proses pengukuran dan perhitungannya.

4) Perkiraan model pada tahap ini akan dilakukan pemilihan teknik data mining yang disesuaikan dengan tugas yang dilakukan.
5) Menafsirkan model dan mengambil sebuah kesimpulan yang berfungsi untuk menafsirkan sebuah model sebagai pendukung kesimpulan yang akan didapatkan.

2.2.3 *Machine Learning*

Machine Learning merupakan cabang dari kecerdasan buatan yang mengembangkan algoritma dengan pembelajaran pola yang tersembunyi dari sekumpulan data yang akan digunakan untuk prediksi pada model data baru yang sama, kecuali secara eksplisit diprogram untuk tiap tugas. *Machine Learning* tradiosional melakukan penggabungan data dengan alat statistic untuk melakukan prediksi *output* yang bisa digunakan membuat wawasan yang bisa ditindaklanjuti. *Machine Learning* digunakan untuk pengenalan gambar, serta ucapan sampai pemprosesan Bahasa alami, sistem rekomendasi, deteksi penipuan, untuk pengoptimalan portofolio, tugas otomatis, dan lain sebagainya. Selain itu, *Machine Learning* juga digunakan untuk menyalakan kendaraan otonom, *drone*, serta robot. *Machine Learning* membuat pembelajaran yang lebih cerdas dan mudah beradaptasi dengan lingkungan. Tugas umum dari *Machine Learning* ini adalah memberikan sebuah rekomendasi, *Machine Learning* digunakan untuk melakukan pengembangan sistem yang dapat menganalisis data dari pengguna serta memberikan saran ataupun rekomendasi berdasarkan pola yang ditemukan dalam

data. *Machine Learning* berkaitan erat dengan data *mining* dan data *science*. *Machine Learning* menerima sebagai masukan dan menggunakan algoritma untuk merumuskan sebuah jawaban. Pembelajaran mesin dapat dilihat pada gambar 2.1 serta perbedaan mengenai *Machine Learning* dan *Traditional programming* dapat dilihat pada tabel 2.2.



Gambar 2. 1 Pembelajaran Mesin

Tabel 2. 2 Perbedaan *machine learning dan traditional programming*

Machine Learning	*Traditional programming*	*Artificial Intelligence (*AI)
Machine Learning merupakan bagian dari kecerdasan buatan (AI) berfokus pada pembelajaran data untuk menggembangkan algoritma sehingga dapat digunakan untuk melakukan prediksi.	*Traditional programming*, kode berbasis pada aturan tertulis oleh pengembang tergantung pernyataan masalah yang dialami.	*Artificial intelligence*, usaha untuk meningkatkan sebuah kemampuan mesin sehingga mesin tersebut dapat menjalankan tugas yang pada dasarnya memerlukan kecerdasan manusia.
Sistem yang dilatih dengan menggunakan data historis yang kemudian dapat digunakan untuk melakukan prediksi pada data yang baru.	Berbasis pada aturan dan deterministic, tidak mempunyai fitur pembelajaran mandiri seperti *Machine Learning* ataupun AI.	AI melibatkan banyak Teknik yang berbeda termasuk juga dengan *Machine Learning* dan *deep learning*, dan *traditional programming*.
Dapat menemukan pola serta wawasan dalam dataset yang besar yang memungkinkan sulit bagi manusia untuk menemukannya.	Sepenuhnya bergantung pada kecerdasan para pengembangnya, sehingga mempunyai kemampuan yang terbatas.	AI memakai gabungan antara data dan rule pre-defined, yang memberikan keunggulan yang besar dalam penyelesaian tugas kompleks.

digunakan untuk berbagai tugas berbasis AI seperti menjawab Pertanyaan *Chatbot*, mobil yang dikendarai sendiri., dll.	*Traditional programming* yang digunakan untuk membangun sebuah sistem perangkat lunak yang memiliki fungsi tertentu.	Mencakup banyak aplikasi berbeda termasuk dengan pemprosesan Bahasa alami, visi komputer, dan robotika.

Siklus Hidup *Machine Learning*

Dalam siklus hidup proyek *Machine Learning* melibatkan serangkaian langkah-langkah seperti berikut:

1. *Definition problems*: Mempelajari permasalahan, pada langkah ini melibatkan pemahaman terkait dengan permasalahan bisnis dan definisi tujuan model.
2. *Data collection*: *Data Collection* atau pengumpulan data digunakan untuk mengumpulkan data yang relevan yang akan diperlukan untuk pemodelan, data tersebut dapat berasal dari berbagai sumber seperti *database*, API, ataupun *web scraping*.
3. *Data preparation*: selanjutnya adalah pengecekan data dengan benar dalam format yang diinginkan sehingga dapat digunakan oleh model unutk menemukan sebuah pola yang tersembunyi. Dengan menggunakan langkah-langkah:
    a. Ekstraksi Data: Ekstraksi data untuk mengidentifikasi dan mengambil informasi yang akan diperlukan dari sumber data. Melibatkan pemilihan kolom tertentu, penggabungan data dari beberapa sumber, atau pemilihan data dalam rentang waktu tertentu.
    b. Pembersihan Data (*Data Cleaning*): mengidentifikasi dan menangani nilai-nilai yang hilang, duplikat, atau anomali dalam data. tahap ini

meliputi penanganan *missing values*, deteksi dan penanganan *outlier*, serta menghapus duplikat.

c. Transformasi Data: Transformasi data mengubah format atau struktur data agar sesuai dengan kebutuhan analisis atau model. Proses ini meliputi normalisasi data, pembuatan fitur baru, atau konversi jenis data.
d. Pemilihan Fitur (*Feature Selection*): pemilihan fitur atau atribut yang paling relevan untuk tugas *Machine Learning*. Hal ini membantu untuk mengurangi dimensi data dan fokus pada informasi yang paling penting.
e. Pemisahan Data (*Data Splitting*): pemisahan data dikategorikan menjadi data pelatihan dan data pengujian. data pelatihan digunakan untuk melatih model, sedangkan data pengujian digunakan untuk menguji kinerja model pada data yang tidak pernah dilihat sebelumnya.

4. Pemilihan model (*Model Selection*): pemilihan algoritma pembelajaran mesin yang sesuai yang cocok untuk permasalahan. Langkah ini membutuhkan pengetahuan tentang kekuatan dan kelemahan algoritma yang berbeda.
5. Pelatihan Model (*Model Training*): data pelatihan digunakan untuk melatih model. Pada proses ini melibatkkan beberapa penyesuaian parameter model sehingga dapat membuat sebuat prediksi.
6. Evaluasi Model: model dapat dievaluasi pada dataset uji untuk penentuan akurasi serta kinerja dengan menggunakan teknik yang berbeda seperti *classification*, F1-*score*, *precision*, *recall*, Kurva ROC, *Mean Square error*, *absolute error*, dan lain sebagainnya.

7. Optimasi Model (*Model Optimization*): menyesuaikan model serta parameter untuk meningkatkan kinerjanya. Hal ini dapat melibatkan penalaan ulang, penambahan fitur, ataupun penyetelan parameter.
8. *Deployment*: model dapat digunakan pada lingkungan produksi untuk membuat prediksi pada sebuah data yang baru. Pada langkah ini memerlukan integrasi model ke dalam sistem perangkat lunak yang ada atau membuat sistem baru untuk model.
9. *Monitoring and Maintenance*: pemantauan kinerja model di lingkungan produksi dan melakukan tugas pemeliharaan sesuai kebutuhan. Yang melibatkan pemantauan penyimpangan data, melatih ulang model sesuai kebutuhan, dan memperbarui model saat data baru tersedia. (Gupta, 2023).

2.2.4 Penyakit Jantung

Berdasarkan perkembangannya penyakit dapat dibagi menjadi 2 yaitu; Akut dan Kronis. Sedangkan berdasarkan sifat penularannya dibagi menjadi; penyakit menular dan tidak menular. Proses timbulnya penyakit karena adanya sebuah interaksi antara agen penyakit dan manusia (*host*) dan lingkungan sekitarnya. Untuk penyakit yang menular, proses timbulnya penyakit tersebut diakibatkan adanya sebuah interaksi antara agen penyakit (mikroorganisme hidup), manusia serta lingkungan. Sedangkan untuk penyakit yang tidak menular, penyakit timbul diakibatkan interaksi antara agen penyakit (*non living agen*t), manusia dan lingkungan. Penyakit tidak menular dapat besifat akut ataupun bersifat kronis. (Darmawan, 2016).

Penyakit kardiovaskuler atau *cardiovascullar disease* (CVD) merupakan kumpulan gangguan yang terjadi didalam sistem jantung dan pembuluh darah. Yang termasuk kedalam penyakit CVD antara lain; penyakit jantung koroner, penyakit jantung kongenital, penyakit jantung rematik, penyakit arteri perifer, trombosis vena dalam, penyakit serebrovaskular, dan juga emboli pulmonal. Penyumbatan terjadi pada aliran darah dari jantung ke otak dan jantung secara akut sehingga mengakibatkan terjadinya serangan jantung dan stroke. (WHO, 2017).

Menurut Rilantono (2013) didalam jurnal Fadlilah, *et al.*(2019) menyebutkan bahwa Penyakit Kardiovaskular adalah penyakit yang ideal untuk melakukan pencegahan. Pada tahun 1981 sampai dengan 2004, di Amerika dan beberapa negara maju, angka kematian yang disebabkan oleh penyakit kardiovaskular mengalami penurunan 50%. Penurunan jumlah tersebut merupakan hasil dari usaha yang dilakukan pencegahan untuk mengontrol faktor risiko. Penghitungan faktor risiko tersebut memiliki tujuan untuk mengetahui seberapa jauh tingkat risiko yang dimiliki individu untuk terkena penyakit kardiovaskular. Sehinga adanya pengetahuan mengenai skor risiko ini diharapkan dapat dilakukan pencegahan primer dalam meningkatkan penurunan faktor risiko.

Menurut Ghani, *et al.* (2016) menyebutkan bahwa terdapat beberapa faktor risiko dominan yang berpengaruh terhadap penyakit jantung koroner dari yang terbesar sampai terkecil kekuatan yang saling berhubungan yaitu:

1. Hipertensi.
2. Gangguan mental emosional.
3. Diabetes mellitus.

4. Stroke.
5. Umur ≥ 40 tahun.
6. Riwayat merokok.
7. Jenis kelamin perempuan.
8. Tingkat pendidikan tidak sekolah hingga tamat SD.
9. Obesitas sentral.
10. Serta tingkat sosial ekonomi miskin atau rendah.

2.2.5 Implementasi

Implementasi dalam Kamus Besar Indonesia menjelaskan bahwa implementasi merupakan pelaksanaan ataupun penerapan. Istilah implementasi berkaitan dengan sebuah proses kegiatan yang dilakukan untuk memperoleh suatu tujuan tertentu. Implementasi adalah penempatan ide, suatu konsep atau bahkan kebijakan atau bahkan dapat berupa inovasi. Didalam suatu kegiatan praktis yang memberikan sebuah dampak yang berupa perubahan pengetahuan, keterampilan ataupun nilai dan sikap. Implementasi dapat berupa sebuah aspek yang penting dalam setiap proses kebijakan dan merupakan suatu bentuk usaha untuk mencapai sebuah tujuan tertentu dengan sarana dan prasarana tertentu serta dalam urutan waktu tertentu. Pada intinya implementasi kebijakan adalah upaya untuk mencapai tujuan yang telah ditentukan dengan program-program agar terpenuhi pelaksanaan kebijakan tersebut. (Hernita Ulfatimah, 2020).

2.2.6 *Python*

Python merupakan sebuah bahasa pemprograman yang interpretatif yang digunakan di berbagai *platform* dengan aturan atau teori rancangan terfokus kepada

tingkat keterbacaan sebuah kode dan salah satu bahasa pemrograman yang memiliki hubungan dengan Data *Science, Machine Learning* atau bahkan *Internet of Thing* (IoT). *Python* memiliki kelebihan yaitu bersifat interpretatif bahkan banyak digunakan untuk *prototyping, scripting* dalam penanganan infrastruktur, sampai pembuatan sebuah *website* berukuran besar. Bahasa *python* termasuk kedalam 3 bahasa pemprograman yang berguna untuk beberapa tahun kedepan. Adapun beberapa pustaka (*library*) yang digunakan untuk data *science* dan *Machine Learning* yaitu: *Scikit-Learn, TensorFlow, pytorch*. Bahasa *python* mempunyai sebuah kurva pembelajaran (*learning-curve*) yang pandai, bahkan sesuai untuk diperlajari sebagai bahasa pemprograman pertama, dengan berbagai kemudahan dalam membaca dan mempelajari sintaksnya. (Farobi, 2021).

Python secara umum merupakan sebuah pemprograman berorientasi objek, pemprograman imperatif, dan pemprograman fungsional. Atau memiliki istilah yang disebut bahasa pemprograman multi-paradigma. *Python* digunakan untuk berbagai kepentingan pengembangan perangkat lunak dan juga *python* dapat berjalan di berbagai platform sistem operasi. Adapun beberapa platform yang mendukung *python* antara lain: *Linux/Unix, Windows, Mac OS X, Java Virtual Machine, OS/2, Amiga, Palm,* dan *Symbian* (produk nokia). *Python* juga memiliki beberapa fitur dan kelebihan yang dimiliki yaitu:

1) Memiliki koleksi *library* (pustaka) yang banyak. Itu artinya bahwa tersedia modul-modul yang siap untuk digunakan untuk berbagai kebutuhan, seperti pembuatan *game* sampai dengan *Artificial Intelligence* (misalnya *TensorFlow*).

2) Mempunyai struktur bahasa yang jelas dan mudah dimengerti, sederhana dan simpel.
3) Berorientasi prosedural dan objek sekaligus (*multi-paradigma*).
4) Memiliki sistem pengelolaan memori otomatis (*garbage collection*) atau seperti java.
5) Memiliki sifat modular sehingga memudahkan dalam pengembangan dengan membuat modul-modul baru, baik dibangun menggunakan bahasa *python* ataupun C/C++. (Enterprise, 2019).