

BAB II

LANDASAN TEORI

2.1 Pengertian Penyakit Jantung

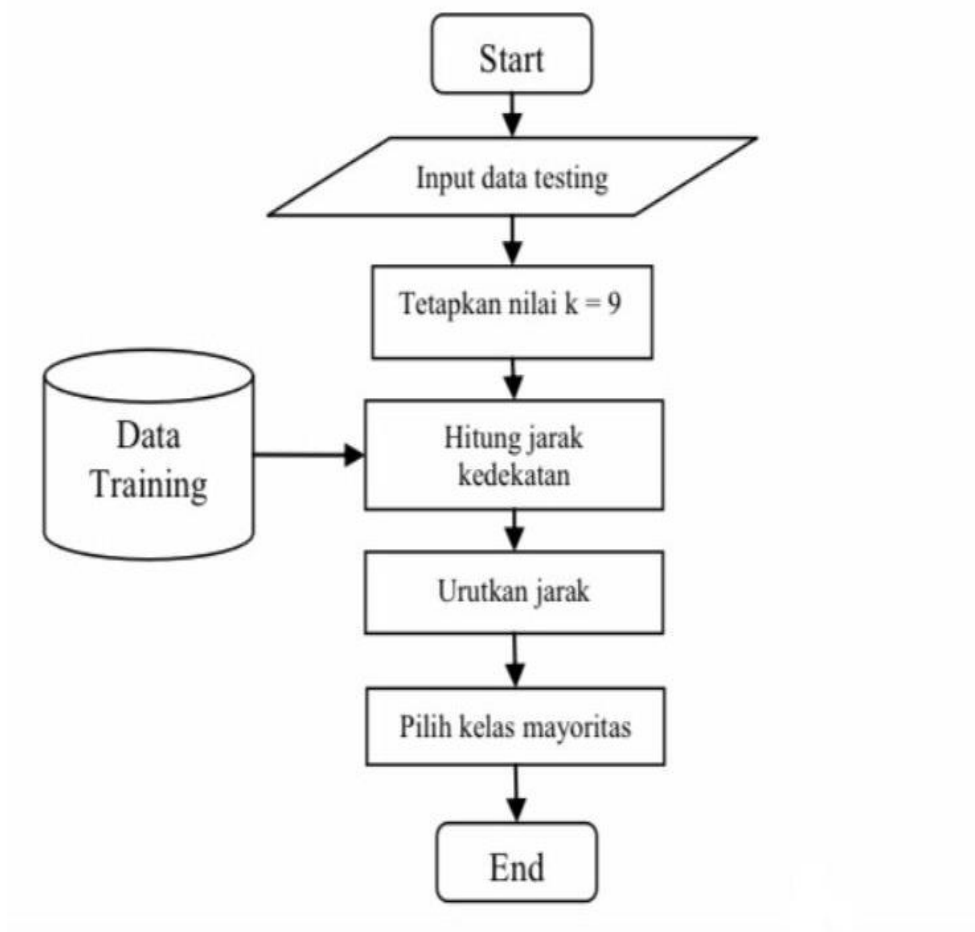
Penyakit jantung merupakan penyebab kematian utama di Indonesia. Menurut Organisasi Kesehatan Dunia (WHO), 31% kematian di dunia disebabkan oleh penyakit jantung. Kematian akibat penyakit ini meningkat seiring dengan berkembangnya gaya hidup tidak sehat di masyarakat. Penyakit jantung merupakan gangguan yang terjadi pada sistem pembuluh darah besar sehingga menyebabkan jantung dan peredaran darah tidak berfungsi sebagaimana mestinya. Penyakit-penyakit yang berhubungan dengan organ jantung dan pembuluh darah antara lain adalah gagal jantung, jantung koroner, dan jantung rematik (Sutanto, 2010). Selain karena bertambahnya usia atau penuaan, ada beberapa faktor risiko yang meningkatkan angka kejadian penyakit jantung. Faktor risiko tersebut antara lain menderita tekanan darah tinggi, menderita diabetes melitus, memiliki kadar kolesterol melebihi normal dan berat badan berlebih serta obesitas. Data Riskesdas Depkes RI menemukan prevalensi obesitas pada kelompok umur dewasa sebesar 15.4 % dan overweight sebesar 13.5 % (Akbarollah et al., 2023).

2.2 Algoritma K-Nearest Neighbor

Variabel eksternal berpengaruh langsung terhadap manfaat dan kenyamanan yang dialami pengguna. Variabel eksternal yang berhubungan dengan fungsi sistem seperti mouse, layar sentuh, menu, mempengaruhi persepsi kemudahan penggunaan, peningkatan penggunaan teknologi, dan selain itu pelatihan individu juga mempengaruhi kegunaan. Semakin banyak pelatihan yang diterima seseorang, semakin mudah penggunaannya. Tujuan dari algoritma KNN adalah untuk mengklasifikasikan objek baru berdasarkan atribut dan sampel pelatihan. Pengklasifikasi tidak menggunakan model apa pun untuk pencocokan dan hanya didasarkan pada memori. Diberikan titik query, akan ditemukan sejumlah k obyek atau (titik training) yang paling dekat dengan titik query. Klasifikasi menggunakan voting terbanyak diantara klasifikasi dari k obyek. Algoritma KNN

menggunakan klasifikasi ketetanggaan sebagai nilai prediksi dari query instance yang baru. Algoritma metode KNN sangatlah sederhana, bekerja berdasarkan jarak terpendek dari query instance ke training sample untuk menentukan KNN-nya. Nilai k yang terbaik untuk algoritma ini tergantung pada data. Secara umum, nilai k yang tinggi akan mengurangi efek noise pada klasifikasi, tetapi membuat batasan antara setiap klasifikasi menjadi semakin kabur. Nilai k yang bagus dapat dipilih dengan optimasi parameter, misalnya dengan menggunakan cross-validation. Kasus khusus dimana klasifikasi diprediksikan berdasarkan training data yang paling dekat (dengan kata lain, $k=1$) disebut algoritma Nearest Neighbor (Dhany, 2021).

Untuk lebih jelasnya algoritma K-NN dapat dilihat pada gambar 2.1 berikut.



Gambar 2. 1 Algoritma KNN

Algoritma K-NN bekerja sebagai berikut.

1. Tentuka parameter K
2. Hitung jarak antara data yang akan dievaluasi dengan semua pelatihan
3. Urutan jarak yang terbentuk (urut naik)
4. Tentukan jarak terdekat sampai urutan K
5. Pasangkan kelas yang bersesuaian
6. Cari jumlah kelas dari tetangga yang terdekat dan tetapkan kelas tersebut sebagai kelas data yang akan dievaluasi.

Persamaan KNN:

$$d_i = \sqrt{\sum_{i=1}^p (x_{2i} - x_{1i})^2} \dots (1)$$

Keterangan:

- x1 = Sampel Data
- x2 = Data Uji / Testing
- I = Variabel Data
- d = Jarak
- p = Dimensi Data

Dataset Penyakit Jantung

8 faktor kesehatan dalam kumpulan data yang digunakan pada penelitian ini diuraikan di bawah ini.

1. *Age* – Umur
2. *Sex* – Jenis Kelamin
 - a. 0 = Perempuan
 - b. 1 = Laki-laki
3. *Systolic Blood Pressure* – Tekanan darah
4. *Cholesterol* – Kolesterol
5. *Thalach* – Denyut nadi maksimal
6. *Oldpeak* – *ST depression induced by exercise relative to rest*
7. *Slope* – *gradient* dari puncak ST segmen

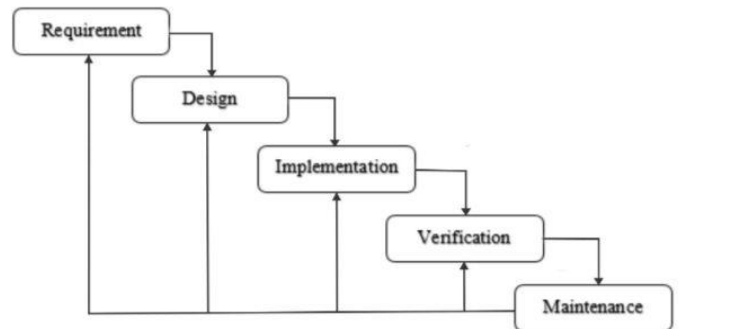
- a. 1 = *unsloping*
 - b. 2 = *flat*
 - c. 3 = *Downsloping*
8. *Cardio* – Diagnosis penyakit jantung
- a. 0 sehat
 - b. 1 penyakit jantung

2.3 Metode Waterfall

Metode air terjun atau yang sering disebut metode waterfall seing dinamakan siklus hidup klasik (*classic life cycle*), nama model ini sebenarnya adalah “Linear Sequential Model” dimana hal ini menggambarkan pendekatan yang sistematis dan juga berurutan pada pengembangan perangkat lunak, dimulai dengan spesifikasi kebutuhan pengguna lalu berlanjut melalui tahapan-tahapan perencanaan (*planning*), permodelan (*modelling*), konstruksi (*contruction*), serta penyerahan sistem ke para pengguna (*deployment*), yang diakhiri dengan dukungan pada perangkat lunak lengkap yang dihasilkan (Pressman, 2010). Model waterfall pertama kali diperkenalkan pada tahun 1970 oleh Winston Royce, sehingga sering dianggap kuno, namun merupakan model yang paling banyak digunakan dalam rekayasa perangkat lunak (SE). Saat ini, model waterfall merupakan model pengembangan perangkat lunak yang umum digunakan. Model pengembangan ini mengikuti pendekatan sistematis dan berurutan. Disebut waterfall karena langkah yang dilakukan harus menunggu langkah sebelumnya selesai dan dijalankan secara berurutan. Model pengembangan ini linier dari tahap awal pengembangan sistem, atau tahap perencanaan, hingga tahap akhir pengembangan sistem, atau tahap pemeliharaan. Langkah selanjutnya tidak dijalankan sampai langkah sebelumnya selesai dan tidak dapat dikembalikan atau diulangi ke langkah sebelumnya..

2.3.1 Tahapan Metode Waterfall

Tahapan dari metode waterfall dapat dilihat pada gambar 2.2 di bawah ini.



Gambar 2. 2 Tahapan Metode *Waterfall*

1. *Requirement*

Pada tahap ini, pengembang sistem memerlukan komunikasi yang bertujuan untuk memahami perangkat lunak yang diharapkan pengguna dan keterbatasan perangkat lunak. Informasi dapat diperoleh melalui wawancara, diskusi atau survei langsung. Data dianalisis untuk memberikan informasi yang dibutuhkan pengguna.

2. *Design*

Pada fase ini, pengembang membuat desain sistem yang membantu menentukan persyaratan perangkat keras dan sistem dan juga mendefinisikan arsitektur sistem secara keseluruhan.

3. *Implementation*

Pada fase ini, sistem pertama-tama dikembangkan menjadi program-program yang disebut unit-unit kecil, yang kemudian diintegrasikan pada fase-fase berikutnya. Setiap perangkat dikembangkan dan diuji fungsionalitasnya, yang disebut pengujian unit.

4. *Verification*

Pada fase ini, sistem diperiksa dan diuji untuk melihat apakah sistem memenuhi seluruh atau sebagian persyaratan sistem. Pengujian dapat diklasifikasikan menjadi pengujian unit (dilakukan pada modul kode tertentu), pengujian sistem

(untuk melihat bagaimana sistem bereaksi ketika semua modul terintegrasi), dan pengujian penerimaan (dilakukan dengan pelanggan atau atas namanya untuk memverifikasi bahwa semua kebutuhan pelanggan terpenuhi).

5. *Maintenance*

Ini adalah langkah terakhir dari metode waterfall. Perangkat lunak yang telah selesai dioperasikan dan dipelihara. Pemeliharaan melibatkan perbaikan kesalahan yang tidak ditemukan pada langkah sebelumnya..

2.3.2 Kelebihan Metode *Waterfall*

Berikut adalah kelebihan dari metode waterfall.

1. Kualitas dari sistem yang dihasilkan akan baik, karena pelaksanaannya dilakukan secara bertahap.
2. Proses pengembangan model fase one by one, sehingga meminimalis kesalahan yang mungkin akan terjadi.
3. Dokumen pengembangan sistem sangat terorganisir, karena setiap fase harus terselesaikan dengan lengkap sebelum melangkah ke fase berikutnya (F.Supandi 2018).

2.3.3 Kekurangan Metode *Waterfall*

Berikut adalah kekurangan dari metode waterfall.

1. Waktu pengembangan yang lama dan mahal.
2. Diperlukan pengelolaan yang baik karena proses pengembangan tidak dapat diulang sampai produk diproduksi.
3. Bug kecil menjadi masalah besar jika tidak terdeteksi sejak awal pengembangan, sehingga mempengaruhi langkah selanjutnya.
4. Pada kenyataannya jarang sekali mengikuti tatanan yang berurutan (*sequence*) seperti pada teori. Seringkali pengulangan (*loop*) justru menimbulkan permasalahan baru.

2.4 UML (*Unified Modelling Language*)



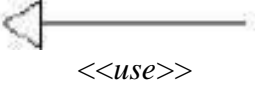

Unified Modeling Language (UML) merupakan kumpulan diagram-diagram yang sudah memiliki standar untuk membangun perangkat lunak berbasis objek (Teguh

Prihandoyo, 2018). UML memiliki banyak diagram *use case*, *activity diagram*, dan *class diagram*.

2.4.1 Use Case Diagram

Use Case Diagram merupakan diagram yang harus dibuat terlebih dahulu pada saat memodelkan perangkat lunak berorientasi objek. Tabel tersebut mencantumkan simbol-simbol yang digunakan untuk membuat diagram *use case* ini, antara lain sebagai berikut..



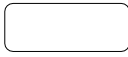
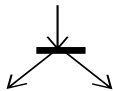



Tabel 2. 1 *Use Case Diagram*

Simbol	Nama	Keterangan
	Aktor	Merupakan Penggunaan dari sistem Penamaan actor menggunakan kata benda.
	<i>Use case</i>	Merupakan pekerjaan yang dilakukan oleh aktor. Penamaan <i>usecase</i> dengan katakerja.
	Asosiasi	Hubungan antara actor dengan <i>usecase</i> .
	<i>Include</i>	Hubungan antara <i>use case</i> dengan <i>usecase</i> , <i>include</i> menyatakan bahwa sebelum pekerjaan dilakukan harus mengerjakan pekerjaan lain terlebih dahulu.
	<i>Extends</i>	Hubungan antara <i>use case</i> dengan <i>use case</i> , <i>extends</i> menyatakan bahwa jika pekerjaan yang dilakukan merupakan fungsional tambahan jika suatu kondisi terpenuhi.

2.4.2 Activity Diagram

Menjelaskan alur kerja atau fungsi suatu sistem atau proses bisnis. Simbol yang digunakan dalam operasi. Skema kerjanya ditunjukkan pada tabel 2.2 di bawah ini.

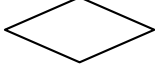
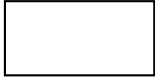
Tabel 2. 2 Activity Diagram

Simbol	Keterangan
	<i>Start Point</i> , diletakkan pada pojok kiri atas dan merupakan awal aktivitas.
	<i>End Point</i> , akhir aktivitas.
	<i>Activities</i> , menggambarkan suatu proses/kegiatan bisnis.
	<i>Fork</i> / percabangan, digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan parallel menjadi satu.
	<i>Join</i> (penggabungan) atau <i>rake</i> , digunakan untuk menunjukkan adanya dekomposisi.
	<i>Decision Points</i> , menggambarkan pilihan untuk pengambilan keputusan, <i>True</i> dan <i>False</i> .
	<i>Swimline</i> , pembagian <i>activity diagram</i> untuk menunjukkan siapa melakukan apa.

2.4.3 Class Diagram

Class diagram adalah jenis diagram UML yang digunakan untuk merepresentasikan kelas dan paket dalam suatu sistem yang akan digunakan nantinya. Dengan demikian diagram ini dapat memberikan gambaran tentang sistem dan hubungan-hubungan yang terdapat pada sistem yang dibuat. Nama skema kelas ditunjukkan pada Tabel 2.3..

Tabel 2. 3 *Class Diagram*

Simbol	Nama	Keterangan
	<i>N-Ary Association</i>	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	<i>Class</i>	Himpunan dari objek-objek yang berbagai atribut serta operasi yang sama.

2.5 Penelitian Terdahulu

Penelitian terdahulu ini mempunyai peranan yang penting, karena dijadikan sebagai bahan refleksi dan referensi dalam penelitian ini. Penelitian sebelumnya telah melibatkan beberapa materi. Penelitian sebelumnya disajikan pada Tabel 2.4 sebagai berikut..

Tabel 2. 4 Penelitian Terdahulu

No.	Judul	Peneliti	Hasil
1	Analisis Penyakit Jantung Menggunakan Metode KNN Dan Random Forest.	(Sahar, 2020)	Tujuan penelitian ini adalah menganalisis penyakit jantung berdasarkan usia pasien dan nyeri dada (angina). K-nearest neighbour (KNN) dan metode hutan acak. Penggunaan model perhitungan Precision, Recall, F1 Score dan Precision. Berdasarkan hasil percobaan yang dilakukan pada penelitian ini, ditunjukkan bahwa

			metode K-Nearest Neighbor dapat digunakan dalam klasifikasi data penyakit jantung. Akurasi 93% dengan metode KNN dan 72% dengan metode Random Forest.
2	Penerapan Algoritma Klasifikasi Nearest Neighbor (K-Nn) Untuk Mendeteksi Penyakit Jantung	(Lestari, 2014)	Pada penelitian ini algoritma K-NN $k = 9$ diterapkan pada data pasien untuk mendeteksi penyakit jantung. Kedekatan data latih dan data uji digunakan untuk menentukan kelas data uji. Untuk mengukur kinerja algoritma dilakukan dengan menggunakan fusion matriks dan kurva ROC diperoleh nilai akurasi sebesar 70% dan tergolong baik karena memiliki nilai AUC sebesar 0,875.
3	Penerapan Algoritma K-Nearest Neighbord Untuk Prediksi Kematian Akibat	(Reza et al., 2022)	Pada penelitian ini diterapkan algoritma prediksi K-Nearest Neighbor terhadap

	Penyakit Jantung	Gagal	<p> kematian akibat gagal jantung, dengan kesimpulan bahwa penerapan algoritma KNN pada data gagal jantung dilakukan dengan cara menghitung secara manual dengan Microsoft Excel kemudian menghitung jaraknya. dari data tersebut dengan menguji data latih dengan $k=7$ tetangga terdekat yang memberikan data uji kelas kematian yaitu no. Setelah itu dilakukan pengujian akurasi menggunakan aplikasi speedminer dan bahasa pemrograman Python. Pengujian dengan rapidminer dilakukan dengan mengubah nilai k, akurasi tertinggi diperoleh dengan $k = 7$ dengan akurasi 94,92%. Kemudian, pengujian dengan bahasa pemrograman Python memberikan akurasi 68 persen. </p>
--	---------------------	-------	---

4	<p>Klasifikasi Penyakit Gagal Jantung Menggunakan Algoritma <i>K-Nearest Neighbor</i>.</p>	<p>(Maskuri et al., 2022)</p>	<p>Pada penelitian ini, algoritma K-Nearest Neighbor diterapkan pada dataset gagal jantung. Data gagal jantung diperoleh dengan menggunakan uji akurasi speed miner. Dataset yang dapat digunakan berisi 918 data dan 12 atribut. Kemudian hasil implementasi algoritma KNN dibuat pada aplikasi matangminer dengan perubahan nilai k, dan diperoleh hasil akurasi tertinggi dengan nilai k = 9 dengan akurasi 70,65%, nilai presesinya adalah 75%. , saat kembali. menghasilkan 70,73%..</p>
---	--	-------------------------------	---