BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Sebelumnya

Penelitian sebelumnya memiliki signifikansi yang tinggi sebagai landasan untuk melaksanakan penelitian yang sedang dilakukan, serta dapat mencegah duplikasi hasil penelitian. Penyusunan proyek penelitian ini didasarkan pada beberapa penelitian sebelumnya yang relevan, yang memiliki keterkaitan dengan metode dan konsep serupa. Berikut adalah rangkuman dari *State of The Art* yang telah peneliti kumpulkan pada Tabel 2. 1.

Tabel 2. 1 *State of The Art* Penelitian

No	Peneliti	Metode	Kelebihan	Kekurangan
1	Aida dkk, 2024 [10]	Klasifikasi Risiko Penyakit Serangan Jantung Dengan Menggunakan Algoritma C4.5	• Didapatkan hasil akurasi sebesar 83,98%. • Model yang dihasilkan mudah diinterpretasikan dan digunakan untuk pengambilan keputusan.	• Memerlukan waktu yang lama untuk melatih model. • Memerlukan banyak data training untuk menghasilkan model yang akurat.
2	Fauzi dkk, 2023 [8]	Penerapan Algoritma *K-Nearest Neighbor* Dalam Klasifikasi Penyakit Jantung	• Didapatkan hasil akurasi sebesar 92,78%. • Mampu menangani data non-linear. • Sederhana dan mudah dipahami.	• Rentan terhadap data yang tidak seimbang. • Kinerja dapat terpengaruh oleh skala fitur, sehingga normalisasi data sering diperlukan.
3	Bowo dkk, 2023 [9]	Implementasi Metode *Naive Bayes* Untuk Klasifikasi Penderita	• Didapatkan tingkat akurasi sebesar 86,84%. • Mudah diimplementasikan	• Kinerja menurun jika fitur-fitur tidak independen. • Tidak dapat menangani data

		Penyakit Jantung	dan memiliki waktu komputasi yang cepat.	dengan interaksi kompleks antara fitur.
4	Hidayat dkk, 2023 [6]	Klasifikasi Penyakit Jantung Menggunakan *Random Forest Classifier*	• Didapatkan hasil evaluasi dengan akurasi sebesar 94,0%. • Memberikan hasil prediksi yang cukup akurat dan stabil.	• Memerlukan banyak data training untuk menghasilkan model yang akurat. • Memakan waktu dan sumber daya komputasi yang signifikan.
5	Ibnu dkk, 2023 [11]	Klasifikasi Penyakit Jantung Menggunakan Algoritma Analisis Diskriminan Linier	• Didapatkan tingkat akurasi sebesar 82,12% • Relatif mudah diimplementasikan dan cepat untuk dilatih.	• Kinerja menurun jika data tidak memenuhi asumsi distribusi normal. • Tidak efektif untuk dataset dengan hubungan non-linier antara fitur.

2.2 Dasar Teori

Pada tahap ini akan menjelaskan hasil tinjauan pustaka yang digunakan dalam penelitian ini.

2.2.1 Penyakit Jantung

Dilansir dari primayahospital.com yang direview oleh (dr. Andriga, 2023), penyakit jantung merupakan suatu kondisi medis yang melibatkan kerusakan atau gangguan pada jantung atau pembuluh darah yang memberikan pasokan darah ke jantung. Penyakit ini juga terkait dengan masalah medis lain yang mempengaruhi fungsi pembuluh darah.

2.2.1.1 Jenis Penyakit Jantung

Terdapat sejumlah jenis penyakit jantung yang memiliki karakteristik berbeda sesuai dengan bagian dari organ jantung yang terpengaruh dan risiko yang ditimbulkan, diantaranya:

a. Penyakit jantung koroner

   Penyakit jantung koroner adalah penyakit yang terjadi akibat penyempitan atau pengerasan arteri koroner yang menghambat aliran darah ke jantung.

b. Aritmia

   Aritma adalah gangguan irama detak jantung yang dapat berupa detak yang terlalu cepat, terlalu lambat, atau tidak teratur.

c. Penyakit jantung bawaan

   Penyakit jantung bawaan adalah kelainan struktural pada jantung yang terjadi sejak lahir atau sebelum lahir.

d. Gagal jantung

   Gagal jantung adalah kondisi di mana jantung tidak mampu memompa darah dengan efektif, yang dapat mempengaruhi fungsi organ tubuh lain yang membutuhkan pasokan darah.

e. Penyakit jantung perifer

   Penyakit jantung perifer adalah kondisi di mana pembuluh darah yang mengalirkan darah ke kaki dan tangan mengalami penyumbatan.

f. Penyakit jantung rematik

   Penyakit jantung rematik adalah penyakit yang menyebabkan kerusakan jaringan jantung akibat infeksi bakteri, terutama pada katup jantung.

g. Penyakit katup jantung

   Penyakit katup jantung adalah kondisi di mana terdapat gangguan pada fungsi katup jantung, seperti penyempitan atau ketidakmampuan untuk menutup sepenuhnya.

h. Kardiomiopati

   Kardiomiopati adalah kelainan pada otot jantung yang disebabkan oleh berbagai faktor, seperti genetik, infeksi, atau penggunaan obat-obatan tertentu.

2.2.1.2 Gejala Penyakit Jantung

Penyakit jantung juga dapat menyebabkan berbagai gejala, tergantung pada jenis dan tingkat keparahannya. Berikut adalah beberapa gejala umum penyakit jantung:

a. Sensasi tidak nyaman atau nyeri dada, seperti terbakar, tertekan, atau diremas, yang dapat menjalar ke rahang, lengan, dan punggung.
b. Sesak napas baik saat beraktivitas maupun saat istirahat.
c. Detak jantung yang tidak teratur.
d. Mudah merasa lelah.
e. Pusing, terutama saat berdiri atau bergerak cepat.
f. Bengkak pada kaki.
g. Terdengar suara desingan atau gemuruh di jantung.
h. Penurunan nafsu makan.

2.2.1.3 Penyebab Penyakit Jantung

Penyebab penyakit jantung bervariasi, tergantung pada kondisi yang dialami oleh individu tersebut. Berikut adalah beberapa penyebab yang umum:

a. Tekanan darah tinggi (hipertensi).
b. Kadar kolesterol tinggi.
c. Diabetes atau kondisi pra-diabetes.
d. Pola makan tidak sehat.
e. Kurangnya aktivitas fisik.
f. Pertambahan usia.
g. Stress.

2.2.1.4 Mendiagnosis Penyakit Jantung

Untuk mendiagnosis penyakit jantung secara medis, serangkaian pemeriksaan dan tes umumnya diperlukan. Berikut adalah beberapa di antaranya:

a. Elektrokardiogram (EKG), untuk merekam aktivitas listrik jantung dan memeriksa irama detak jantung.
b. Ekokardiogram, menggunakan gelombang suara untuk mendapatkan gambaran detail tentang struktur dan fungsi jantung.
c. Tes stres jantung, untuk mengevaluasi fungsi jantung saat aktivitas meningkat, seperti berjalan di treadmill atau bersepeda statis.
d. Angiografi koroner menggunakan sinar-X dan cairan kontras, untuk memeriksa kondisi arteri koroner.
e. Tes darah, untuk memeriksa kadar kolesterol, gula darah, dan enzim jantung yang dilepaskan saat terjadi kerusakan pada jantung.
f. Pencitraan resonansi magnetik (MRI) atau tomografi terkomputasi (*CT scan*) [12].

2.2.2 Dataset Yang Digunakan

Dataset adalah kumpulan data yang terstruktur dan terorganisir, yang digunakan sebagai input untuk melatih dan menguji model *machine learning*. Dalam penelitian ini, dataset yang digunakan adalah dataset dari data medis penyakit jantung yang sudah tersedia dalam website *machine learning open-source repository*. Pemilihan dataset ini dilakukan dengan mempertimbangkan karakteristik dan relevansinya terhadap populasi yang diteliti dan ketersediaan variabel yang dibutuhkan untuk analisis. Dataset ini akan digunakan untuk melatih model *Random Forest*, dengan tujuan mengidentifikasi faktor risiko utama dan memprediksi kemungkinan seseorang menderita penyakit jantung dengan tingkat akurasi yang baik.

2.2.2.1 *Heart Disease Cleveland UCI*

Dataset ini diunggah oleh *Cherngs* dalam website *kaggle.com* pada tahun 2020 [13]. Dataset ini berupa file *Comma Separated Values* (CSV) yang berisi beberapa informasi medis dari pasien yang telah menjalani evaluasi untuk penyakit jantung di *Cleveland Clinic Foundation*, Ohio, Amerika Serikat. Dataset ini sudah dibersihkan oleh pengunggah dan juga menjadi salah satu dataset yang paling sering digunakan oleh peneliti di bidang *machine learning* yang berfokus pada pasien penyakit jantung. Pada Tabel 2. 2 adalah isian beberapa baris data awal dan akhir dari dataset *Heart Disease Cleveland UCI*.

Tabel 2. 2 Isian Baris Data dari Dataset Pertama

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	condition
69	1	0	160	234	1	2	131	0	0.1	1	1	0	0
69	0	0	140	239	0	0	151	0	1.8	0	2	0	0
66	0	0	150	226	0	0	114	0	2.6	2	0	0	0
65	1	0	138	282	1	2	174	0	1.4	1	1	0	1
64	1	0	110	211	0	2	144	1	1.8	1	0	0	0
. . .													
40	1	3	152	223	0	0	181	0	0	0	0	2	1
39	1	3	118	219	0	0	140	0	1.2	1	0	2	1
35	1	3	120	198	0	0	130	1	1.6	1	0	2	1
35	0	3	138	183	0	0	182	0	1.4	0	0	0	0
35	1	3	126	282	0	2	156	1	0	0	0	2	1

2.2.2.2 *Heart Disease CDC Survey Health Status of U.S. Residents*

Dataset ini diunggah oleh *Kamil Pytlak* dalam website *kaggle.com* pada tahun 2022 [14]. Dataset ini berupa file *Comma Separated Values* (CSV) yang mencakup data dari survei tahunan CDC BRFSS (*Behavioral Risk Factor Surveillance System*) pada tahun 2020 dengan informasi seperti demografis, perilaku kesehatan, dan status kesehatan dari lebih dari 300.000 orang dewasa di Amerika Serikat. Dataset ini dapat digunakan untuk analisis dan klasifikasi penderita penyakit jantung. Pada Tabel 2. 3 adalah isian beberapa baris data awal dan akhir dari dataset *Heart Disease CDC Survey Health Status of U.S. Residents*.

Tabel 2. 3 Isian Baris Data dari Dataset Kedua

HeartDisease	BMI	Smoking	AlcoholDrinking	Stroke	PhysicalHealth	MentalHealth	DiffWalking	Sex	AgeCategory	Race	Diabetic	PhysicalActivity	GenHealth	SleepTime	Asthma	KidneyDisease	SkinCancer
No	16.6	Yes	No	No	3.0	30.0	No	Female	55-59	White	Yes	Yes	Very good	5.0	Yes	No	Yes
No	20.34	No	No	Yes	0.0	0.0	No	Female	80 or older	White	No	Yes	Very good	7.0	No	No	No
No	26.58	Yes	No	No	20.0	30.0	No	Male	65-69	White	Yes	Yes	Fair	8.0	Yes	No	No
No	24.21	No	No	No	0.0	0.0	No	Female	75-79	White	No	No	Good	6.0	No	No	Yes
No	23.71	No	No	No	28.0	0.0	Yes	Female	40-44	White	No	Yes	Very good	8.0	No	No	No
Yes	28.87	Yes	No	No	6.0	0.0	Yes	Female	75-79	Black	No	No	Fair	12.0	No	No	No
No	21.63	No	No	No	15.0	0.0	No	Female	70-74	White	No	Yes	Fair	4.0	Yes	No	Yes
No	31.64	Yes	No	No	5.0	0.0	Yes	Female	80 or older	White	Yes	No	Good	9.0	Yes	No	No
No	26.45	No	No	No	0.0	0.0	No	Female	80 or older	White	No, borderline diabetes	No	Fair	5.0	No	Yes	No
No	40.69	No	No	No	0.0	0.0	Yes	Male	65-69	White	No	Yes	Good	10.0	No	No	No
Yes	34.3	Yes	No	No	30.0	0.0	Yes	Male	60-64	White	Yes	No	Poor	15.0	Yes	No	No
No	28.71	Yes	No	No	0.0	0.0	No	Female	55-59	White	No	Yes	Very good	5.0	No	No	No
No	28.37	Yes	No	No	0.0	0.0	Yes	Male	75-79	White	Yes	Yes	Very good	8.0	No	No	No
No	28.15	No	No	No	7.0	0.0	Yes	Female	80 or older	White	No	No	Good	7.0	No	No	No
No	29.29	Yes	No	No	0.0	30.0	Yes	Female	60-64	White	No	No	Good	5.0	No	No	No
No	29.18	No	No	No	1.0	0.0	No	Female	50-54	White	No	Yes	Very good	6.0	No	No	No
No	26.26	No	No	No	5.0	2.0	No	Female	70-74	White	No	No	Very good	10.0	No	No	No
No	22.59	Yes	No	No	0.0	30.0	Yes	Male	70-74	White	No, borderline diabetes	Yes	Good	8.0	No	No	No
No	29.86	Yes	No	No	0.0	0.0	Yes	Female	75-79	Black	Yes	No	Fair	5.0	No	Yes	No
No	18.13	No	No	No	0.0	0.0	No	Male	80 or older	White	No	Yes	Excellent	8.0	No	No	Yes
No	21.16	No	No	No	0.0	0.0	No	Female	80 or older	Black	No, borderline diabetes	No	Good	8.0	No	No	No
No	28.9	No	No	No	2.0	5.0	No	Female	70-74	White	Yes	No	Very good	7.0	No	No	No
No	26.17	Yes	No	No	0.0	15.0	No	Female	45-49	White	No	Yes	Very good	6.0	No	No	No
No	25.82	Yes	No	No	0.0	30.0	No	Male	80 or older	White	Yes	Yes	Fair	8.0	No	No	No
No	25.75	No	No	No	0.0	0.0	No	Female	80 or older	White	No	Yes	Very good	6.0	No	No	Yes
									...								
No	26.63	No	No	No	0.0	0.0	No	Female	45-49	Hispanic	No	No	Very good	6.0	No	No	No
No	23.62	No	No	No	0.0	0.0	No	Male	25-29	Hispanic	No	Yes	Good	7.0	No	No	No
No	30.9	No	No	No	0.0	0.0	No	Female	18-24	Hispanic	No	Yes	Excellent	7.0	No	No	No
No	26.45	No	No	No	3.0	2.0	No	Female	18-24	Hispanic	No	No	Fair	8.0	No	No	No
No	20.36	No	No	No	30.0	0.0	Yes	Female	55-59	Hispanic	Yes	Yes	Fair	8.0	No	No	No
No	24.37	Yes	No	No	0.0	0.0	No	Male	35-39	Hispanic	No	No	Good	8.0	No	No	No
No	27.98	No	No	No	0.0	0.0	No	Female	50-54	Hispanic	No	No	Good	8.0	No	No	No
No	42.57	No	No	No	0.0	0.0	Yes	Female	60-64	Hispanic	No	No	Good	7.0	No	No	No
No	26.63	No	No	No	0.0	0.0	No	Female	25-29	Hispanic	No	No	Very good	8.0	No	No	No
No	23.38	Yes	No	No	30.0	0.0	Yes	Female	70-74	Hispanic	No	Yes	Fair	5.0	No	No	No
No	30.67	No	No	No	0.0	8.0	No	Female	25-29	Hispanic	No	Yes	Very good	7.0	No	No	No
Yes	37.12	Yes	No	No	0.0	0.0	No	Male	35-39	Hispanic	No	Yes	Very good	7.0	No	No	No
No	31.89	Yes	No	No	30.0	30.0	Yes	Female	55-59	Hispanic	No	No	Fair	4.0	No	No	No
No	33.28	No	No	No	0.0	0.0	No	Female	30-34	Hispanic	No	Yes	Excellent	8.0	No	No	No
No	26.58	Yes	No	No	0.0	0.0	Yes	Male	75-79	Hispanic	No	Yes	Good	6.0	No	No	No
No	31.93	No	Yes	No	0.0	0.0	No	Male	65-69	Hispanic	No	Yes	Good	7.0	No	No	No
Yes	33.2	Yes	No	No	0.0	0.0	No	Female	60-64	Hispanic	Yes	Yes	Very good	8.0	Yes	No	No
No	36.54	No	No	No	7.0	0.0	No	Male	30-34	Hispanic	No	No	Good	9.0	No	No	No
No	23.38	No	No	No	0.0	0.0	No	Female	60-64	Hispanic	No	Yes	Excellent	6.0	No	No	No
No	22.22	No	No	No	0.0	0.0	No	Female	18-24	Hispanic	No	Yes	Excellent	8.0	No	No	No
Yes	27.41	Yes	No	No	7.0	0.0	Yes	Male	60-64	Hispanic	Yes	No	Fair	6.0	Yes	No	No
No	29.84	Yes	No	No	0.0	0.0	No	Male	35-39	Hispanic	No	Yes	Very good	5.0	Yes	No	No
No	24.24	No	No	No	0.0	0.0	No	Female	45-49	Hispanic	No	Yes	Good	6.0	No	No	No
No	32.81	No	No	No	0.0	0.0	No	Female	25-29	Hispanic	No	No	Good	12.0	No	No	No
No	46.56	No	No	No	0.0	0.0	No	Female	80 or older	Hispanic	No	Yes	Good	8.0	No	No	No

2.2.2.3 *Heart Disease Hungarian UCI*

Dataset ini diunggah oleh *Bima Rakajati* dalam website *github.com* pada tahun 2024 [15]. Dataset ini berupa file *Comma Separated Values* (CSV) yang berisi beberapa informasi medis dari pasien yang telah menjalani evaluasi untuk penyakit jantung di *Hungarian Institute of Cardiology*, Budapest, Hungaria. Dataset ini juga sudah dibersihkan oleh pengunggah. Pada Tabel 2. 4 adalah isian beberapa baris data awal dan akhir dari dataset *Heart Disease Hungarian UCI.*

Tabel 2. 4 Isian Baris Data dari Dataset Ketiga

age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	target
40.0	1.0	2.0	140.0	289.0	0.0	0.0	172.0	0.0	0.0	0.0
49.0	0.0	3.0	160.0	180.0	0.0	0.0	156.0	0.0	1.0	1.0
37.0	1.0	2.0	130.0	283.0	0.0	1.0	98.0	0.0	0.0	0.0
48.0	0.0	4.0	138.0	214.0	0.0	0.0	108.0	1.0	1.5	3.0
54.0	1.0	3.0	150.0	251.0	0.0	0.0	122.0	0.0	0.0	0.0
. . .										
48.0	0.0	2.0	133.0	308.0	0.0	1.0	139.0	0.0	2.0	0.0
36.0	1.0	2.0	120.0	166.0	0.0	0.0	180.0	0.0	0.0	0.0
48.0	1.0	3.0	110.0	211.0	0.0	0.0	138.0	0.0	0.0	0.0
47.0	0.0	2.0	140.0	257.0	0.0	0.0	135.0	0.0	1.0	0.0
53.0	1.0	4.0	130.0	182.0	0.0	0.0	148.0	0.0	0.0	0.0

2.2.3 Transformasi Data

Transformasi data adalah serangkaian teknik yang digunakan untuk mengubah struktur, format, atau nilai data untuk meningkatkan kualitas dan konsistensi data. Transformasi ini mencakup konversi data dari satu format ke format lain, penggantian data yang tidak sesuai, dan pengubahan skala atau distribusi data. Transformasi ini mencakup berbagai teknik yang membantu meningkatkan kualitas data dan kinerja model *machine learning.*

2.2.3.1 Tujuan Transformasi Data

Transformasi data membantu dalam mengatasi berbagai masalah yang mungkin timbul dari data mentah dan memastikan bahwa model yang dibuat akurat dan andal. Secara lebih spesifik, transformasi data bertujuan untuk:

1. Meningkatkan Kualitas Data

   Menghilangkan *noise* dan inkonsistensi dalam data untuk menghasilkan dataset yang lebih bersih dan lebih dapat diandalkan. Data yang berkualitas tinggi akan membantu dalam menghasilkan model yang lebih akurat dan dapat diandalkan.

2. Skalabilitas

   Menyesuaikan skala fitur sehingga semua fitur memberikan kontribusi yang setara dalam proses pemodelan, menghindari dominasi fitur dengan skala yang lebih besar. Ini memastikan bahwa model *machine learning* dapat mempelajari pola dari semua fitur dengan baik.

3. Peningkatan Kinerja Model

   Mempermudah algoritma *machine learning* untuk mempelajari pola yang relevan dari data dengan mengubah data menjadi format yang lebih sesuai. Transformasi data membantu dalam meningkatkan efisiensi dan efektivitas algoritma, yang pada akhirnya meningkatkan kinerja model.

2.2.3.2 *Encoding* Fitur Kategorikal

Encoding fitur kategorikal adalah teknik mengubah data kategorikal menjadi format numerik yang dapat digunakan oleh algoritma *machine learning*. Data kategorikal adalah data yang nilainya merupakan kategori atau label yang tidak memiliki hubungan numerik langsung, seperti "*Male*" atau "*Female*" untuk jenis kelamin, atau "*Red*," "*Blue*," dan "*Green*" untuk warna. Ada beberapa teknik untuk melakukan encoding fitur kategorikal diantaranya:

a. *Label Encoding*

   Setiap kategori dalam fitur diberikan label numerik yang unik. Misalnya, untuk fitur warna dengan kategori "*Red*," "*Blue*," dan "*Green*," Label *Encoding* akan mengubahnya menjadi 0, 1, dan 2. Teknik ini mudah diterapkan dan digunakan.

b. *One-Hot Encoding*

Mengubah setiap kategori menjadi kolom biner terpisah. Setiap kolom mewakili satu kategori dan diisi dengan 1 atau 0, tergantung pada apakah sampel tersebut termasuk dalam kategori itu atau tidak. Misalnya, fitur warna dengan kategori "*Red*," "*Blue*," dan "*Green*" akan diubah menjadi tiga kolom: "*Red*," "*Blue*," dan "*Green*," dengan nilai biner.

c. *Binary Encoding*

Kombinasi dari *Label Encoding* dan *One-Hot Encoding*. Dalam teknik ini, label pertama diubah menjadi biner dan kemudian dipecah menjadi kolom biner terpisah. Teknik ini mengurangi jumlah kolom yang dihasilkan dibandingkan dengan *One-Hot Encoding*, terutama jika jumlah kategori sangat besar.

d. *Target Encoding*

Mengubah kategori menjadi nilai rata-rata target (label) untuk setiap kategori. Teknik ini dapat memberikan informasi yang lebih kaya karena mempertimbangkan hubungan antara fitur kategorikal dan target.

e. *Frequency Encoding*

Teknik yang mengubah kategori menjadi frekuensi kemunculannya dalam data. Teknik ini menyederhanakan data kategorikal berdasarkan frekuensinya, memberikan informasi tambahan tentang seberapa umum setiap kategori dalam dataset [16].

2.2.4 *Sampling Data*

Sampling Data adalah teknik pra-pemrosesan data yang digunakan untuk mengatasi ketidakseimbangan kelas dalam dataset. Ketidakseimbangan kelas terjadi ketika jumlah contoh dalam satu kelas jauh lebih banyak dibandingkan dengan kelas lainnya. Ketidakseimbangan kelas dapat menyebabkan model *machine learning* untuk cenderung memprediksi kelas mayoritas dan mengabaikan kelas minoritas. Ini mengakibatkan kinerja model yang buruk pada kelas minoritas.

Proses ini dilakukan dengan menyesuaikan kumpulan data sehingga kelas-kelas memiliki distribusi yang seimbang. *Sampling Data* dapat dilakukan dengan berbagai cara, seperti mengurangi jumlah sampel dari kelas mayoritas atau menambah jumlah sampel dari kelas minoritas. Selain itu, modifikasi data dapat dilakukan secara acak, di mana penambahan sampel bisa dilakukan dengan menduplikasi sampel yang sudah ada atau membuat sampel sintetis berdasarkan sampel yang ada [17].

2.2.4.1 Keuntungan *Sampling Data*

Sampling Data adalah teknik penting dalam pra-pemrosesan data yang memberikan berbagai keuntungan signifikan, terutama saat mengatasi ketidakseimbangan kelas dalam dataset. Berikut adalah keuntungan utama dari Sampling Data:

1. Peningkatan akurasi kelas minoritas

   Ketika dataset memiliki ketidakseimbangan kelas yang signifikan, model cenderung mengabaikan kelas minoritas. Dengan menggunakan teknik sampling seperti SMOTE, jumlah contoh dari kelas minoritas ditingkatkan. Ini memungkinkan model untuk belajar lebih banyak tentang karakteristik kelas minoritas, sehingga meningkatkan akurasi prediksi untuk kelas tersebut.
2. Generalisasi yang lebih baik

   Model yang dilatih pada dataset yang seimbang lebih mungkin untuk belajar pola yang berlaku umum dan tidak hanya spesifik pada data pelatihan. Ini meningkatkan kemampuan model untuk membuat prediksi yang akurat pada data baru.

2.2.4.2 SMOTE (*Synthetic Minority Over-sampling Technique*)

SMOTE (*Synthetic Minority Over-sampling Technique*) adalah salah satu teknik *oversampling* yang digunakan untuk mengatasi masalah ketidakseimbangan kelas dalam dataset. SMOTE bekerja dengan menghasilkan sampel sintetis baru dari kelas minoritas daripada hanya menduplikasi sampel yang sudah ada. Teknik

ini membantu meningkatkan representasi kelas minoritas sehingga model *machine learning* dapat belajar lebih baik dan memberikan hasil yang lebih akurat.

Berikut adalah cara kerja dari SMOTE:

1. Pemilihan Sampel Minoritas

   SMOTE memulai dengan memilih sampel minoritas secara acak dari dataset.
2. Pencarian *K-Nearest Neighbors*

   Untuk setiap sampel minoritas yang dipilih, SMOTE mencari *k-nearest neighbors* (tetangga terdekat) dari kelas yang sama. Nilai k biasanya ditentukan berdasarkan parameter yang diatur sebelumnya (misalnya, k=5).
3. Pembuatan Sampel Sintetis

   SMOTE kemudian menghasilkan sampel sintetis baru dengan memilih salah satu dari k tetangga terdekat dan menciptakan sampel baru di sepanjang garis lurus yang menghubungkan sampel minoritas asli dengan tetangganya. Proses ini dapat dijelaskan dengan persamaan berikut:

   $$x\ Sintetis = x\ Asli + \lambda \times (x\ Tetangga - x\ Asli) \qquad (2.\ 1)$$

   Di mana $x\ Sintetis$ adalah sampel sintetis yang dihasilkan, $x\ Asli$ adalah sampel minoritas asli, $x\ Tetangga$ adalah salah satu tetangga terdekat, dan λ adalah bilangan acak antara 0 dan 1 yang memastikan variasi dalam sampel sintetis yang dihasilkan [18].

2.2.5 *Machine Learning*

Machine learning merupakan salah satu bentuk kecerdasan buatan yang memungkinkan aplikasi perangkat lunak untuk memprediksi *output* tanpa memerlukan pemrograman khusus. Fokus utama *machine learning* adalah pengembangan program komputer yang mampu mengakses data dan menggunakan informasi tersebut untuk belajar secara mandiri. Algoritma dalam teknik *machine learning* menggunakan data historis, seperti pengalaman dan instruksi, sebagai masukan untuk memprediksi nilai keluaran baru.

Proses *machine learning* dimulai dengan pengamatan data dan pola yang ada. Kemudian, perangkat akan membuat keputusan dan menghasilkan *output* yang lebih baik di masa depan. Selain itu, *machine learning* dapat membantu dalam pembuatan model yang dapat memproses dan menganalisis data kompleks untuk menghasilkan hasil yang tepat dan akurat.

2.2.5.1 Cara Kerja *Machine Learning*

Machine learning menggunakan algoritma dan model statistik untuk menganalisis data, mengenali pola serta hubungan, dan menghasilkan prediksi atau keputusan berdasarkan analisis tersebut. Terdapat beberapa tahapan dalam proses pembelajaran mesin, antara lain:

1. Pengumpulan data

   Tahap pertama dalam proses *machine learning* adalah mengumpulkan data yang akan digunakan untuk melatih model pembelajaran mesin. Data ini dapat diperoleh dari berbagai sumber, seperti sensor, *database*, dan lainnya.
2. Pra-pemprosesan data (*pre-processing*)

   Pada tahap ini, data harus dibersihkan dan diproses sebelumnya agar sesuai dengan format yang tepat untuk analisis. Hal ini mungkin melibatkan penghapusan nilai yang hilang, penskalaan data, konversi variabel kategorikal menjadi variabel numerik dan *sampling data* jika diperlukan.
3. Pemilihan model

   Pemilihan model ini tergantung pada karakteristik masalah yang ingin diselesaikan dan jenis data yang akan dianalisis. Beberapa contoh model pembelajaran mesin yang umum meliputi regresi, klasifikasi, dan pengelompokan.
4. Melatih model

   Model yang telah dipilih akan dilatih menggunakan data yang telah diproses sebelumnya. Selama proses pelatihan, model akan diekspos dengan data dan menyesuaikan parameter-parameter internalnya untuk meningkatkan kinerjanya.

5. Evaluasi

   Model tersebut dievaluasi menggunakan kumpulan data terpisah untuk menilai sejauh mana kinerjanya. Evaluasi ini membantu dalam menentukan apakah model tersebut akurat dan dapat diandalkan dalam melakukan prediksi atau pengambilan keputusan.

6. Penerapan

   Setelah model dilatih dan dievaluasi, model tersebut dapat diterapkan atau digunakan dalam aplikasi dunia nyata. Ini dapat melibatkan integrasi model ke dalam sistem yang lebih besar atau pengembangan antarmuka pengguna yang memungkinkan pengguna berinteraksi dengan model dan menerima prediksi atau rekomendasi.

2.2.5.2 *Supervised Learning*

Supervised learning atau pembelajaran terarah, merupakan salah satu kategori jenis algoritma dalam *machine learning*. Dalam *supervised learning*, model dilatih menggunakan dataset yang terdiri dari *input* yang telah diketahui dan *output* yang diinginkan. Tujuan utama adalah untuk mengembangkan model yang dapat memprediksi *output* yang tepat untuk input baru berdasarkan pola yang dipelajari dari dataset pelatihan.

Salah satu algoritma yang termasuk *supervised learning* adalah algoritma *Random Forest*. *supervised learning* berfokus pada pembelajaran hubungan atau fungsi pemetaan dari input ke output berdasarkan contoh pasangan input-output yang diberikan. *Random Forest* dapat digunakan untuk menganalisis berbagai fitur pasien dan memberikan hasil klasifikasi berupa prediksi apakah seorang pasien menderita penyakit jantung atau tidak [19].

2.2.5.3 Klasifikasi Dalam *Machine Learning*

Klasifikasi dalam *machine learning* mengacu pada kemampuan algoritma untuk mengidentifikasi pola dalam data yang telah diajarkan dan menggunakan pola tersebut untuk menentukan kelas atau kategori dari data baru. Proses ini secara fundamental terbagi menjadi dua tahap utama, yaitu:

1. Pelatihan Model

   Tahap pelatihan melibatkan penggunaan dataset pelatihan, yang merupakan kumpulan data dengan contoh *input* (fitur) dan *output* (label) yang diketahui, untuk mengajarkan model tentang hubungan antara *input* dan *output*. Selama proses pelatihan, algoritma *machine learning* secara iteratif menyesuaikan parameter internalnya untuk meminimalkan perbedaan antara prediksi model dan hasil aktual. Tujuannya adalah untuk membuat model yang dapat secara akurat memetakan *input* ke *output* yang diinginkan, sehingga dapat mengklasifikasikan data baru dengan tepat.
2. Inferensi atau Prediksi

   Setelah model dilatih dan dievaluasi menggunakan dataset pengujian, model tersebut dapat digunakan untuk inferensi, atau membuat prediksi, pada data baru. Dalam tahap ini, model menggunakan pengetahuan yang diperoleh selama pelatihan untuk menentukan kelas atau kategori untuk klasifikasi dari *input* baru tanpa memerlukan jawaban yang diketahui sebelumnya.

2.2.5.4 Evaluasi Model *Machine Learning*

Evaluasi model adalah langkah penting dalam proses pengembangan *machine learning*, yang menentukan seberapa baik model memprediksi data baru. Metrik yang biasa digunakan untuk mengevaluasi model klasifikasi ialah:

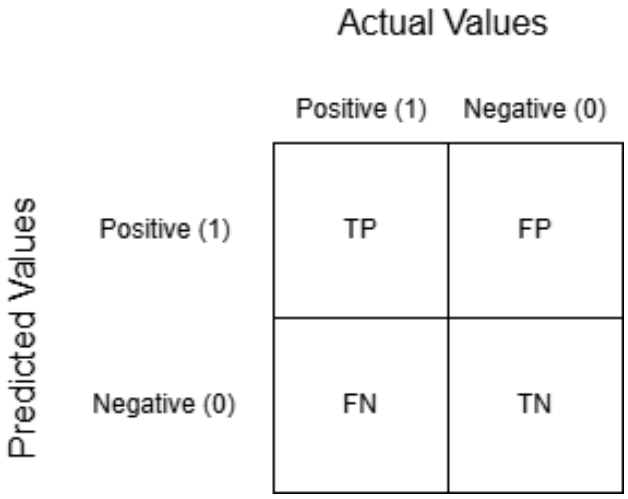
1. *Confusion Matrix*

   Confusion matrix atau matriks kebingungan adalah tabel yang digunakan untuk menggambarkan kinerja model klasifikasi pada set data uji yang nilai sebenarnya diketahui.

Confusion matrix terdiri dari empat komponen utama:

a. *True Positives* (TP): jumlah positif yang diklasifikasikan dengan benar oleh model.
b. *False Positives* (FP): jumlah negatif yang diklasifikasikan salah sebagai positif oleh model.
c. *True Negatives* (TN): jumlah negatif yang diklasifikasikan dengan benar oleh model.
d. *False Negatives* (FN): jumlah positif yang diklasifikasikan salah sebagai negatif oleh model.

Pada Gambar 2. 1 adalah contoh dari tabel *confusion matrix* atau matriks kebingungan.



Gambar 2. 1 *Confusion Matrix*

2. Akurasi (*Accuracy*)

Akurasi merupakan rasio prediksi yang benar terhadap total jumlah prediksi. Berikut persamaan untuk menghitung nilai akurasi:

$$Akurasi = \frac{True\ Positives + True\ Negatives}{Total\ Prediksi} \qquad (2.\ 2)$$

3. Presisi (*Precision*)

   Presisi adalah rasio prediksi positif yang benar terhadap total prediksi positif. Berikut persamaan untuk menghitung nilai presisi:

$$Presisi = \frac{True\ Positives}{True\ Positivies + False\ Positives} \quad (2.\ 3)$$

4. Sensitivitas (*Recall*)

   Sensitivitas mengukur rasio prediksi positif yang benar terhadap semua kasus aktual positif. Berikut persamaan untuk menghitung nilai sensitivitas:

$$Sensitivitas = \frac{True\ Positives}{True\ Positivies + False\ Negatives} \quad (2.\ 4)$$

5. Spesifisitas (*Specificity*)

   Spesifisitas mengukur rasio prediksi negatif yang benar terhadap semua kasus aktual negatif. Berikut persamaan untuk menghitung nilai spesifisitas:

$$Spesifisitas = \frac{True\ Negatives}{True\ Negatives + False\ Positives} \quad (2.\ 5)$$

6. *F1-Score*

   F1-Score merupakan rata-rata harmonik dari presisi dan sensitivitas, memberikan keseimbangan antara kedua metrik tersebut. Berikut persamaan untuk menghitung nilai *F1-Score*:

$$F1 - Score = 2\ x\ \frac{Presisi\ x\ Recall}{Presisi + Recall} \quad (2.\ 6)$$

Evaluasi model memberikan wawasan tentang seberapa baik model bekerja dan membantu dalam memilih model terbaik untuk tugas yang diberikan. Selain itu, evaluasi dapat membantu mengidentifikasi masalah dan memandu untuk peningkatan model lebih lanjut [20].

2.2.6 Algoritma *Random Forest*

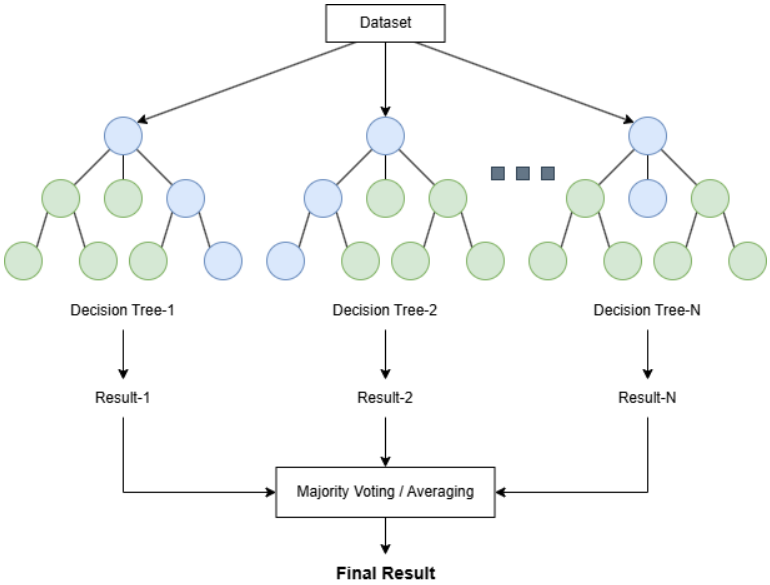
Random Forest adalah algoritma dalam *machine learning* yang digunakan untuk klasifikasi dan prediksi dataset terlabel dalam jumlah besar. Menurut (Breiman 2001) dalam bukunya berjudul "*Random Forests*", *Random Forest* adalah suatu algoritma yang mengkombinasikan beberapa pohon keputusan atau *decision tree* yang dibuat secara acak dengan tujuan menghasilkan klasifikasi atau prediksi yang lebih akurat. Sementara itu, menurut (Cutler et al. 2007) dalam bukunya yang berjudul "*Random Forests for Classification in Ecology*", *Random Forest* adalah suatu algoritma yang menggunakan teknik *ensemble learning* dengan menggabungkan beberapa *decision tree atau* pohon keputusan yang dibuat secara acak untuk meningkatkan akurasi klasifikasi dan regresi pada data yang cukup kompleks [7].

Algoritma ini dapat digunakan untuk mengklasifikasikan sebuah data baru berdasarkan input data sebelumnya. Klasifikasi dan prediksi ini dilakukan melalui penggabungan *tree* dalam *decision tree* dengan cara training dataset yang dimiliki. Algoritma *Random Forest* menggunakan *decision tree* atau pohon keputusan untuk melangsungkan proses seleksi, di mana *tree* atau pohon dari *decision tree* akan dibagi secara rekursif berdasarkan data pada kelas yang sama. Dalam hal ini, penggunaan jumlah *tree* dapat memengaruhi hasil akurasi yang didapat. Penentuan klasifikasi dan prediksi dengan *Random Forest* dilakukan berdasarkan hasil voting dari *tree* yang terbentuk.

2.2.6.1 Cara Kerja *Random Forest*

Cara kerja *Random Forest* adalah dengan membangun banyak pohon keputusan (*decision trees*) dari subset acak dari dataset asli. Setiap pohon keputusan dalam hutan ini menghasilkan prediksi berdasarkan subset data dan fitur yang dipilih secara acak. Setelah semua pohon selesai dibuat dan dilatih, hasil prediksi dari setiap pohon tersebut digabungkan. Untuk klasifikasi, *Random Forest* menggunakan metode voting mayoritas kelas yang paling sering diprediksi oleh pohon-pohon tersebut menjadi hasil akhir. *Random Forest* biasanya dilatih dengan

teknik *ensemble learning*. *Ensemble* secara sederhana berarti menggabungkan beberapa model, dalam algoritma ini yang digabungkan adalah *decision tree*. Dengan demikian, kumpulan model digunakan untuk membuat sebuah prediksi daripada hanya menggunakan satu model individu. Teknik *ensemble* yang biasa digunakan untuk melatih *Random Forest* adalah dengan metode *bagging* [6]. Pada Gambar 2. 2 adalah ilustrasi sederhana untuk proses kerja dari algoritma *Random Forest*.



Gambar 2. 2 Cara Kerja Random Forest

2.2.6.2 *Decision Tree*

Decision tree pertama kali dikenalkan oleh J.R. Quinlan pada tahun 1986. *Decision tree* atau pohon keputusan adalah model prediktif yang digunakan dalam *machine learning* untuk membuat keputusan atau prediksi berdasarkan serangkaian aturan *if-then-else* yang diterapkan pada fitur data input. Struktur *decision tree* menyerupai diagram alir, dengan *node internal* mewakili pengujian pada fitur, cabang mewakili hasil pengujian, dan daun (*leaf nodes*) mewakili prediksi akhir atau keputusan [21].

Dalam *Random Forest*, *decision tree* atau pohon keputusan berfungsi sebagai konstituen dasar dari model *ensemble*. *Random Forest* membangun banyak *decision tree* atau pohon keputusan yang dilatih secara independen pada sampel dataset yang berbeda-beda melalui proses *bootstrapping*. Dengan menggabungkan prediksi dari banyak *decision tree* atau pohon keputusan, *Random Forest* bertujuan untuk mengurangi varians dan *overfitting* yang sering terjadi pada *decision tree* atau pohon keputusan tunggal, sehingga menghasilkan model yang lebih stabil dan akurat.

2.2.6.3 CART (*Classification and Regression Trees*)

CART (*Classification and Regression Trees*) adalah salah satu algoritma pembentuk *decision tree*. *Random Forest* merupakan hasil pengembangan dari algoritma CART yang menggunakan metode *bagging* dan *random sampling with replacement*. Dalam klasifikasi, CART membagi data ke dalam kelas-kelas yang berbeda. Berikut adalah struktur dari CART:

1. *Root Node*

   Node teratas dari pohon keputusan dan mewakili fitur terbaik yang membagi data paling baik. Pemilihan fitur ini didasarkan pada kriteria pemisahan tertentu seperti *gini impurity* (untuk klasifikasi).
2. *Internal Nodes*

   Node yang berada di antara *root node* dan *leaf nodes*. Setiap *internal node* mewakili pengujian pada fitur tertentu dan membagi dataset berdasarkan hasil pengujian tersebut. Setiap cabang dari *node* menunjukkan salah satu hasil pengujian (misalnya, nilai lebih besar atau lebih kecil dari *threshold* tertentu).
3. *Leaf Nodes*

   Node terminal yang memberikan prediksi akhir atau keputusan. Setiap *leaf node* mewakili salah satu kelas target dalam kasus klasifikasi.

Dalam proses pembentukan CART, pada setiap *node*, fitur terbaik dipilih untuk membagi data. Pemilihan ini didasarkan pada kriteria tertentu seperti *gini impurity* untuk klasifikasi. *Gini impurity* adalah ukuran seberapa sering elemen

dalam dataset yang dipilih secara acak akan salah klasifikasi jika elemen tersebut diberi label berdasarkan distribusi kelas di dataset tersebut. Semakin rendah nilai *gini impurity*, semakin baik pembagian data tersebut. Berikut persamaannya:

$$Gini(t) = 1 - \sum_{i=1}^{c} Pi^2 \qquad (2.\ 7)$$

Di mana Pi adalah proporsi sampel dari kelas i pada node t, dan C adalah jumlah kelas.

Selanjutnya dataset dibagi menjadi dua subset berdasarkan fitur yang dipilih dan nilai pemisah (*threshold*). Proses ini diulang secara rekursif untuk setiap subset, membentuk struktur pohon dengan banyak cabang hingga mencapai node terminal (*leaf node*) atau memenuhi kriteria berhenti. Pada *leaf node*, prediksi akhir untuk sampel yang jatuh ke dalam node tersebut diberikan berdasarkan mayoritas kelas [22].

2.2.6.4 *Bagging*

Metode *bagging* yang merupakan singkatan dari *Bootstrap Aggregating* adalah suatu teknik dalam *machine learning* yang digunakan untuk meningkatkan kinerja model dengan menggabungkan beberapa model yang dilatih secara independen dengan menggunakan sampel acak yang diambil dengan penggantian dari dataset pelatihan yang tersedia.

Bagging berfungsi sebagai teknik *ensemble* dalam algoritma *Random Forest*. Berikut adalah langkah-langkah yang terlibat dalam *bagging:*

1. Pengambilan Sampel Bootstrap

   Pengambilan sampel dengan pengembalian (*sampling with replacement*) atau bootstrap sampling. Misalnya, jika dataset asli memiliki n sampel, beberapa subset data pelatihan berukuran n dibuat dengan mengacak dan memilih sampel dari dataset asli secara acak dengan pengembalian. Ini berarti beberapa sampel dalam dataset asli dapat muncul beberapa kali dalam satu subset atau tidak muncul sama sekali.

2. Pelatihan Model Independen

   Setiap subset bootstrap digunakan untuk melatih pohon keputusan. Pada setiap *node* dalam pohon, subset acak dari fitur dipertimbangkan untuk pemisahan. Ini meningkatkan keragaman antara pohon-pohon keputusan dan mengurangi korelasi antar pohon.

3. Penggabungan Prediksi

   Untuk klasifikasi, setiap pohon dalam hutan memberikan suara untuk kelas tertentu, dan kelas yang menerima suara terbanyak dipilih sebagai prediksi akhir (*majority voting*) [23].

2.2.6.5 Parameter Penting Dalam *Random Forest*

Parameter digunakan dalam *Random Forest* untuk meningkatkan kinerja, kecepatan dan kemampuan prediksi dalam model. Berikut adalah parameter untuk memaksimalkan kinerja dari model *Random Forest*:

a. Jumlah pohon (`n_estimators`)

   Menentukan jumlah pohon keputusan (*decision trees*) dalam hutan (*forest*). Semakin banyak pohon, biasanya model akan lebih stabil dan akurat, tetapi pada titik tertentu, peningkatan jumlah pohon memberikan manfaat yang semakin berkurang.

b. Kedalaman maksimum pohon (`max_depth`)

   Kedalaman maksimum setiap pohon keputusan. Kedalaman yang lebih besar dapat meningkatkan kemampuan model untuk mempelajari data dengan lebih detail tetapi juga dapat menyebabkan *overfitting*.

c. Jumlah fitur untuk dipertimbangkan pada setiap split (`max_features`)

   Jumlah maksimal fitur yang dipertimbangkan saat membagi sebuah *node* dalam *Random Forest*. Memilih lebih sedikit fitur dapat meningkatkan kecepatan pelatihan dan mengurangi *overfitting*.

d. Jumlah sampel minimum untuk membagi *node* (`min_samples_split`)

   Jumlah sampel minimum yang diperlukan untuk membagi *node internal*. Nilai yang lebih tinggi mencegah pembentukan node yang memiliki sampel

terlalu sedikit, yang dapat mengurangi *overfitting* tetapi juga dapat membuat model kurang fit dengan data pelatihan.

e. Jumlah sampel minimum pada *leaf node* (`mini_sample_leaf`)

   Jumlah sampel minimum yang diperlukan untuk menjadi *node* daun. Nilai yang lebih tinggi mengurangi risiko *overfitting* dengan membuat struktur pohon yang lebih sederhana.

f. Kriteria (`criterion`)

   Metode yang digunakan untuk membagi node dalam setiap pohon. Untuk tugas klasifikasi, mengubah kriteria dari "*gini*" menjadi "*entropy*" atau sebaliknya dapat mempengaruhi cara pohon dibangun dan tingkat akurasi yang didapat juga kemungkinan berbeda.

g. Metode pemilihan sampel *bootstrap* (`bootstrap`)

   Menggunakan sampel *bootstrap* (*default* adalah *true*) membantu dalam menambah keragaman pada pohon yang dibangun, yang dapat meningkatkan kinerja model. Namun, dalam beberapa kasus, menggunakan seluruh dataset (*false*) dapat memberikan hasil yang lebih baik, tergantung pada data spesifik yang digunakan.

h. *Random State* (`random_state`)

   Mengontrol penentuan angka acak, memastikan konsistensi dan reproduksibilitas hasil model. Proses pengambilan sampel dan pemilihan fitur acak tetap konsisten, sehingga perubahan nilai pada parameter ini dapat mempengaruhi struktur pohon dan akurasi model [24].

2.2.7 Bahasa Pemrograman dan Pustaka Pendukung

Bahasa pemrograman dan pustaka pendukung merupakan komponen inti dalam ekosistem *machine learning*, menyediakan alat dan fungsi yang diperlukan untuk mengolah data, mengembangkan model, dan melakukan evaluasi. Dengan adanya berbagai pilihan bahasa pemrograman dan pustaka yang tersedia, penting untuk memahami karakteristik dan keunggulan masing-masing untuk dapat memilih kombinasi yang paling sesuai dengan kebutuhan proyek.

2.2.7.1 *Python*

Python adalah bahasa pemrograman yang dianggap sebagai pilihan yang paling cocok untuk berbagai proyek kecerdasan buatan (AI), termasuk dalam *machine learning*. *Python* mempunyai ekosistem perpustakaan yang bagus dan lengkap. Perpustakaan dalam *Python* memungkinkan pelaksanaan *continuous data processing* yang diperlukan oleh *machine learning*, karena memberikan kemampuan untuk mengakses, memanipulasi, dan mengubah data dengan sangat efisien. Bahasa pemrograman *python* juga sangat fleksibel karena menyediakan opsi penggunaan OOP atau *scripting*, tanpa perlu perlu *re-compile* pada kode sumber, dan juga dapat diintegrasikan dengan bahasa lain untuk optimalisasi tujuan tertentu. *Python* pada *machine learning* dapat dijalankan di semua platform, termasuk *Windows, MacOS, Linux, Unix*, dan platform lainnya.

2.2.7.2 *Scikit-learn*

Scikit-learn adalah pustaka *Python open-source* yang menyediakan berbagai alat untuk *machine learning* dan data mining. *Scikit-learn* menyediakan akses ke berbagai algoritma *machine learning*, termasuk prediksi, klasifikasi, regresi, clustering, dan reduksi dimensi, serta alat untuk pemilihan model, pra-pemrosesan data, dan evaluasi model. Fitur utama dari *Scikit-learn* adalah API yang konsisten dan desain orientasi objek yang memudahkan penggunaan dan integrasi algoritma-algoritma *machine learning* ke dalam aplikasi *Python*.

2.2.7.3 *Pandas*

Pandas adalah pustaka *Python* yang menyediakan struktur data dan alat analisis data yang cepat, fleksibel, dan ekspresif. *Pandas* memudahkan berbagai operasi data, seperti manipulasi data, agregasi, dan visualisasi. Struktur data utama dalam *Pandas* adalah *DataFrame*, yang memungkinkan penyimpanan dan manipulasi data dengan cara yang efisien dan intuitif.

2.2.7.4 *Matplotlib & Seaborn*

Matplotlib adalah pustaka *plotting* untuk *Python* yang menyediakan fungsi untuk membuat berbagai jenis plot dan visualisasi data. *Seaborn* membangun di atas *Matplotlib* dan menyediakan antarmuka tingkat tinggi untuk membuat visualisasi statistik yang menarik dan informatif. Kedua pustaka ini sering digunakan bersama untuk memvisualisasikan hasil analisis dan model *machine learning*, memungkinkan peneliti dan analis untuk menyajikan temuan mereka dengan cara yang mudah dipahami.