LAPORAN TUGAS AKHIR

IMPLEMENTASI XGBOOST UNTUK DIAGNOSIS PENYAKIT JANTUNG: STUDI KASUS PADA DATASET MEDIS

Disusun Oleh:

230411100082 - Abdillah Fikrul Azhari

230411100087 - Zanuar Rikza Aditiya

230411100092 - Nabiilah Rizqi Amalia

230411100104 - Moch Sigit Aringga

Dosen Pengampu:

Eka Mala Sari Rochman, S.Kom., M.Kom.

NIP: 198407162008122001

PRODI TEKNIK INFORMATIKA

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS TRUNOJOYO MADURA

2025

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Allah SWT yang telah melimpahkan rahmat, taufik, serta hidayah-Nya sehingga kami dapat menyelesaikan laporan penelitian yang berjudul "Implementasi XGBoost untuk Diagnosis Penyakit Jantung: Studi Kasus pada Dataset Medis" ini dengan baik dan tepat waktu. Laporan ini disusun sebagai salah satu bentuk pemenuhan tugas akhir pada mata kuliah Kecerdasan Komputasional di Program Studi Teknik Informatika, Fakultas Teknik, Universitas Trunojoyo Madura.

Penelitian ini dilatarbelakangi oleh pentingnya pemanfaatan teknologi dalam bidang kesehatan, khususnya dalam upaya membantu proses diagnosis penyakit jantung secara lebih cepat dan akurat. Dengan kemajuan algoritma machine learning seperti XGBoost, diharapkan pengambilan keputusan medis dapat didukung secara lebih objektif berdasarkan data yang ada. Melalui penelitian ini, kami berupaya menerapkan model prediktif berbasis XGBoost pada dataset medis guna mengetahui seberapa efektif metode ini dalam mengidentifikasi potensi penyakit jantung.

Kami menyadari bahwa penyusunan laporan ini tidak lepas dari bantuan, bimbingan, serta dukungan dari berbagai pihak. Oleh karena itu, dengan segala kerendahan hati, kami menyampaikan ucapan terima kasih yang sebesar-besarnya kepada Ibu Eka Mala Sari Rochman, S.Kom., selaku dosen pengampu mata kuliah *Kecerdasan Komputasional*, yang telah memberikan arahan, bimbingan, serta masukan yang sangat berharga dari awal hingga akhir penyusunan laporan ini. Teman-teman seperjuangan di Program Studi Teknik Informatika, atas segala bantuan, semangat, serta kebersamaan yang sangat berarti dalam menjalani proses pembelajaran dan penyusunan tugas akhir ini. Seluruh pihak yang tidak dapat kami sebutkan satu per satu, namun telah memberikan dukungan, baik secara langsung maupun tidak langsung, selama proses penyusunan laporan ini berlangsung.

Kami menyadari sepenuhnya bahwa laporan ini masih jauh dari sempurna, baik dari segi isi maupun penyajiannya. Oleh karena itu, kami sangat mengharapkan kritik dan saran yang membangun dari para pembaca demi perbaikan di masa mendatang. Harapan kami, semoga laporan ini dapat memberikan kontribusi positif bagi perkembangan ilmu pengetahuan, khususnya dalam bidang Kecerdasan Komputasional dan penerapannya di dunia medis.

Akhir kata, kami ucapkan terima kasih dan berharap semoga Allah SWT senantiasa melimpahkan rahmat, berkah, dan hidayah-Nya kepada kita semua. Amin.

Bangkalan, 24 Juni 2025

Penulis

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Penyakit jantung merupakan salah satu penyebab utama kematian di seluruh dunia. Menurut data World Health Organization (WHO), jutaan orang meninggal setiap tahunnya akibat penyakit jantung, Sedangkan di Indonesia, Data Riset Kesehatan Dasar (Riskesdas) tahun 2013-2018 menunjukkan penyakit jantung memiliki tren meningkat dari 0,5% menjadi 1,5%. Untuk biaya perawatan dengan BPJS, penyakit jantung menjadi klaim pembiayaan terbesar pada tahun 2021 yaitu sebesar Rp. 7,7 Triliun (Budianto & Syarief, 2023).

Dalam beberapa tahun terakhir, perkembangan teknologi informasi dan kecerdasan buatan (Artificial Intelligence) telah memberikan kontribusi yang signifikan di berbagai bidang, termasuk sektor kesehatan. Salah satu pendekatan yang banyak digunakan adalah *machine learning*, yakni suatu metode komputasional yang memungkinkan sistem untuk belajar dari data dan membuat prediksi atau keputusan tanpa diprogram secara eksplisit. Metode ini dapat dimanfaatkan dalam menganalisis data medis guna mendeteksi pola dan indikator yang relevan terhadap suatu penyakit.

XGBoost (Extreme Gradient Boosting) merupakan salah satu metode *supervised learning* yang digunakan untuk klasifikasi dan regresi. XGBoost bekerja dengan menggabungkan berbagai pengklasifikasi lemah menjadi model yang lebih kuat, melalui pelatihan berurutan menggunakan hasil klasifikasi sebelumnya, yang disebut residuals atau error. Keunggulan XGBoost terletak pada kemampuannya dalam menangani data besar, mengurangi risiko overfitting. Dengan kelebihan tersebut, XGBoost menjadi salah satu pilihan algoritma yang potensial dalam membangun sistem prediksi diagnosis penyakit jantung berbasis data medis (Sausan, Pratiwi, & Mufidah, 2024).

Penelitian ini bertujuan untuk mengimplementasikan algoritma XGBoost dalam proses diagnosis penyakit jantung menggunakan dataset medis sebagai studi kasus. Melalui penelitian ini, diharapkan dapat diketahui sejauh mana tingkat akurasi dan efektivitas XGBoost dalam mengidentifikasi pasien yang berisiko terkena penyakit jantung. Hasil dari penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan sistem yang lebih efisien.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disampaikan, rumusan masalah yang dapat diidentifikasi adalah sebagai berikut:

1. Apa saja teori yang digunakan dalam mengerjakan proyek implementasi XGBoost dalam prediksi penyakit jantung?
2. Bagaimana merancang dan mengimplementasikan prediksi dengan menggunakan metode XGBoost untuk mendukung prediksi terhadap penyakit jantung?

1.3 Tujuan

Penelitian ini bertujuan:

1. Menganalisis teori-teori yang relevan yang digunakan dalam proyek implementasi metode XGBoost untuk sistem pendukung keputusan prediksi diagnosis penyakit jantung.
2. Merancang dan mengimplementasikan sistem pendukung keputusan dengan Metode XGBoost untuk mendukung prediksi diagnosis penyakit jantung.

Dengan mencapai tujuan ini, diharapkan dapat meningkatkan akurasi dan efisiensi dalam proses diagnosis penyakit jantung, serta memberikan kontribusi pada pengembangan sistem dalam konteks kesehatan.

1.4 Manfaat

Penelitian ini diharapkan memberikan kontribusi signifikan pada bidang kesehatan, khususnya dalam prediksi diagnosis penyakit jantung, dengan melibatkan implementasi Metode Penyakit jantung. Manfaat dari penelitian ini meliputi:

1. Memberikan solusi yang efisien dan akurat bagi tenaga medis dalam mendiagnosis penyakit jantung, sehingga dapat meningkatkan kualitas perawatan pasien.
2. Mendorong pengembangan teknologi kesehatan dengan memanfaatkan Metode XGBoost sebagai alat evaluasi yang dapat mempertimbangkan berbagai kriteria resiko penyakit jantung.

Melalui manfaat-manfaat tersebut, penelitian ini diharapkan dapat memberikan sumbangan yang berarti dalam upaya meningkatkan kesejahteraan pasien dan pengembangan sistem kesehatan yang berkelanjutan.

BAB II

DASAR TEORI

2.1 Penyakit Jantung

Penyakit jantung merupakan salah satu penyakit yang dianggap sebagai penyebab utama kematian sesorang. Penyakit jantung merupakan kumpulan beberapa kondisi yang memengaruhi kesehatan jantung manusia. Beberapa kondisi tersebut antara lain penyakit pembuluh darah seperti serangan jantung, stroke, gagal jantung dan aritmia [4]. Kondisi jantung yang terganggu termasuk kardiovaskular, jantung koroner, dan serangan jantung, seringkali dipicu oleh gaya hidup yang tidak sehat. Faktor faktor seperti kebiasaan merokok, pola makan yang tidak baik, konsumsi alkohol, kafein berlebih, begadang, serta stres memberikan dampak negatif pada kesehatan jantung. Disamping itu faktor faktor fisiologis seperti kolesterol, obesitas, darah tinggi juga turut memperburuk resiko individu terhadap penyakit ini. Pengobatan penyakit jantung tersedia dalam beberapa metode antara lain operasi, penyinaran, kemoterapi (Sausan, Pratiwi, & Mufidah, 2024).

Menurut data dari World Health Organization (WHO), penyakit jantung menyumbang sekitar 31% dari total kematian global. Tingginya angka kejadian serta serta sulitnya deteksi dini penyakit jantung mendorong perlunya pendekatan teknologi untuk membantu tenaga medis dalam pencegahan dan pengobatan yang lebih efektif.Diagnosis yang cepat dan tepat dapat meningkatkan peluang pencegahan komplikasi serius pada pasien dengan resiko tinggi (Sausan, Pratiwi, & Mufidah, 2024). Metode konvensional, seperti konsultasi medis dan tes laboratorium cenderung memerlukan biaya yang besar, sehingga pendekatan alternatif yang efesien sangat diperlukan. Dalam konteks ini, teknologi khususnya *machine learning* memiliki peran yang sangat penting. Dengan menggunakan kemampuan *machine learning* membuat, memproses, dan menganalisis data pada jumlah besar, sistem prediksi berbasis teknologi ini mampu membuat prediksi yang seksama dan bisa diandalkan pada saat yang lebih singkat. Teknologi ini memungkinkan tenaga medis untuk dapat lebih mudah mengenali pola dan gejala yang berhubungan dengan risiko penyakit jantung, yang pada akhirnya membantu tenaga medis pada pengambilan keputusan yang lebih tepat terkait kesehatan pasien (Sausan, Pratiwi, & Mufidah, 2024).

2.2 Metode XGBoost

XGBoost Classifier adalah algoritma pembelajan mesin yang diterapkan untuk regresi dan klasifikasi pada dataset besar. Algoritma ini menggunakan pohon keputusan dangkal yang dibangun secara berurutan untuk memberikan hasil yang akurat. Sebagai pengembangan dari *Gradient Boosting,* XGBoost adalah metode esemble berbasis decision tree yang dirancang untuk mempercepat waktu proses, bahkan pada data berukuran besar. XGBoost bekerja dengan menggabungkan berbagai klasifikasi lemah menjadi model yang lebih kuat, melalui pelatihan berurutan menggunakan klasifikasi sebelumnya, yang disebut residuals atau eror (Sausan, Pratiwi, & Mufidah, 2024). Rumus XGBoost memperkanalkan regularisasi dalam fungsi objektif untuk mencegah *overvitting*, dengan fungsi objektif yang di diefinisikan dengan persamaan :

$$0 \; - \; \sum_{i=1}^{n} L(\, yi, fxi)) \; + \; \sum_{k=1}^{t} R(fk) \; + \; C$$

Pada persamaan diatas terdapat bebarapa jenis penjelasan sebagai berikut :

a. L(yi,fxi)) : Fungsi kerugian (loss function) yang berfungsi mengukur tinbgkat akurasi model dalam memprediksi data.
b. R (Fk) : Istilah regularisasi yang berfungsi mencegah *overfitting*, diformulasikan sebagai $aH \; + \; \frac{1}{2} \, n \; + \; \sum_{j=1}^{H} W_j^2 \;\; dengan$:
    - a : menyatakan tingkat komplektifitas pada daun
    - H : menunjukkan jumlah daun pada model
    - n : mengindikasikan parameter pinalti
    - $W_j^2 : \; mengacu \, pada \, output \, yang \, dihasilkan \, setipa \, simpul \, daun$
c. C : Konstanta yang diabaikan secara efektif.

XGBoost bekerja dengan membangun model *ensemble* yang terdiri dari banyak pohon keputusan yang digabungkan secara bertahap untuk memperbaiki kesalahan prediksi dari model sebelumnya. Setiap pohon dalam *ensemble* ini berkontribusi dalam mengurangi error secara bertahap menggunakan gradient boosting.

2.3 Metode SMOTE

SMOTE (*syntetic Minority Oversampling Technique)* adalah teknik yang digunakan untuk mengatasi masalah ketidak seimbangan data dalam mesin learning. Ketidakseimbangan ini terjadi ketika salah satu kelas dalam dataset memiliki jumlah sampel yang lebih sedikit dibandingkan kelas lainnya. SMOTE meningkatkan representasi kelas minoritas dengan menghasilkan data sintetis, sehingga membuat dataset menjadi lebih seimbang. SMOTE bekerja dengan prinsip interpolasi, dimana data sintetis dihasilkan dengan membuat sampel baru yang berada diantara dua sampel kelas minoritas yang ada. Prosesnya dimulai dengan memilih sampel dari kelas minoritas, lalu mengidentifikasi beberapa model terdekatnya (*k-nearest neighborrs*) menggunakan metrik jarak, seperti jarak *Euclidean*. Setelah itu ,salah satu tetangga terdekat dipilih secra acak, dan sampel baru dibuat menggunakan interpolasi linear antara dua titik. Data sintetik ini dihitung dengan persamaan :

$$x_{synthetic} = x_i + \lambda \, . \, (x_j \; - \; x_i)$$

di mana $x_{synthetic}$ merupakan data sintetik yang dihasilkan, xi menunjukkan data dari kelas minoritas yang dipilh sbecara acak, xj menunjukkan tetangga terdekat dari xi, dan λ merupakan faktor pengalia acak dalam interval [0,1], yang menentukan posisi interpolasi antara xi dan xj (Sidiq et al., 2025).

2.4 Metode Evaluasi

Evaluasi model dilakukan untuk mengukur performa prediksi dengan membandingkan hasil prediksi model terhadap nilai aktual pada data testing. Evaluasi dimulai dengan menghitung *confusion matrix*, yang menunjukkan jumlah prediksi benar dan salah untuk masing masing kelas, termasuk T*rue positive (TP), False Positive (FP), True Negative (TN), False Negative (FN)*. *Confusion matrix* berfungsi sebagai dasar analisis kinerja model klasifikasi dengan menyajikan visualisasi komprehensif tetntang kemampuan model dalam membedakan antar kelas dan mengidentifikasi pola kesalahan prediksi.

Berdasarkan *confusion matrix,* bebrapa metrik kinerja dihitung, seperti akurasi, *precision,recall,* dan F1-*score.* Selain itu, analisis *Receiver Operating Characteristic (ROC) Curve* dilakukan untuk mengevaluasi *trade-off* antara tingkat deteksi positif (*True Positive Rate*) dan tingkat kesalahan positif (*False Positive Rate*).Area Under the Curve (AUC) digunakan sebagai indikator kemampuan model untuk membedakan antara kelas positif dan negatif (Sidiq et al., 2025). Hasil evaluasi dibandingkan dengan model baseline untuk menilai keunggulan pendekatan yang diusulkan. *confusion matrix* mencakup beberapa perhitungan :

1. Presisi

   Presisi adalah metrik untuk menghitung proporsi data yang benar-benar positif dari seluruh prediksi yang dinyatakan positif. Perhitungan presisi dapat dilakukan menggunakan Persamaan :

$$Presisi = \frac{TP}{FN + TP}$$

2. Recall

   Recall ialah metrik untuk menghitung proporsi data yang diprediksi benar positif dari seluruh data yang seharusnya positif. Recall dihitung menggunakan Persamaan :

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

3. F1 Score

   F1 Score ialah adalah metrik yang digunakan untuk menghitung rata-rata harmonis antara presisi dan recall. Perhitungan F1 Score dilakukan dengan menggunakan Persamaan :

$$F1\ Score = 2 \times \frac{Recall \times Presisi}{Recall + Presisi}$$

4. Akurasi

   Akurasi ialah metrik untuk mengukur sejauh mana model dapat melakukan klasifikasi dengan benar. Perhitungan akurasi dapat dilakukan menggunakan Persamaan :

$$Akurasi = \frac{TN + TP}{TP + TN + FP + FN}$$

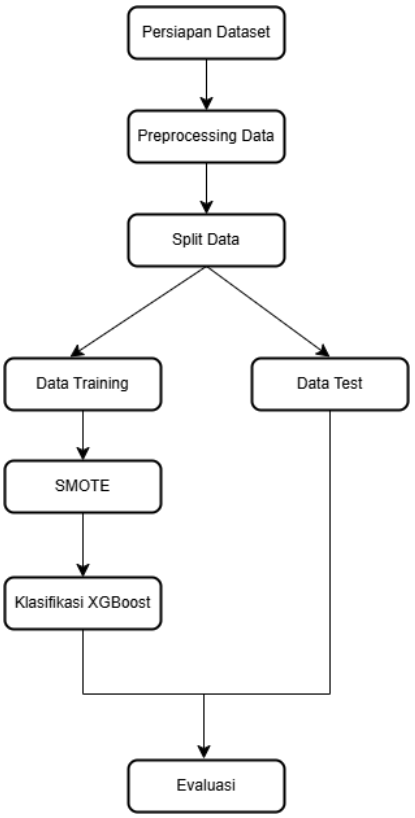
BAB III

METODE

Penelitian ini bertujuan untuk membangun sistem prediksi penyakit jantung dengan memanfaatkan pendekatan machine learning berbasis eksperimen. Data yang digunakan dalam penelitian ini bersumber dari Kaggle, sebuah platform yang menyediakan kumpulan data open-source untuk keperluan penelitian dan pengembangan model pembelajaran mesin. Dataset yang digunakan memuat informasi pasien terkait kondisi kesehatan jantung, yang terdiri atas sejumlah fitur numerik. Data ini kemudian diproses menggunakan bahasa pemrograman Python dengan dukungan berbagai library, terutama library XGBoost yang digunakan sebagai metode klasifikasi utama dalam penelitian ini. Selanjutnya, dilakukan evaluasi terhadap performa model untuk mengukur efektivitas metode XGBoost dalam memprediksi penyakit jantung.

3.1 Prosedur Penelitian

Metodologi dalam penelitian ini disusun secara sistematis untuk menunjang proses analisis dan implementasi algoritma XGBoost Classifier dalam menyelesaikan permasalahan klasifikasi. Alur lengkap prosedur penelitian disajikan pada Gambar 1.



Gambar 1. Diagram Langkah-Langkah Analisis

Gambar 1 menunjukkan tahapan analisis yang diterapkan dalam penelitian ini, dengan tujuan mengimplementasikan metode klasifikasi XGBoost untuk prediksi penyakit jantung, sehingga dapat mendukung proses diagnosis dini. Diagram dimulai dengan tahap persiapan dataset, yaitu proses pengumpulan data dari sumber yang relevan, seperti Kaggle. Selanjutnya, dilakukan pra-pemrosesan data yang mencakup penanganan nilai duplikat, *missing value, encoding* data kategori. Setelah itu, data dibagi menjadi dua bagian, yaitu data training dan data test. Pada data training, dilakukan teknik SMOTE untuk melakukan oversampling terhadap kelas minoritas, kemudian dilanjutkan dengan proses pelatihan model menggunakan algoritma XGBoost. Model yang telah dilatih kemudian diuji menggunakan data test, dan pada tahap akhir dilakukan proses evaluasi kinerja model menggunakan metrik seperti akurasi, precision, recall, dan F1-score untuk menilai efektivitas model dalam memprediksi penyakit jantung.

3.2 Teknik Pengumpulan Dataset

Penelitian ini menggunakan data numerik yang berkaitan dengan diagnosis penyakit jantung, yang diambil dari situs Kaggle. Dataset ini berjudul "UCI Heart Disease Data", yang tersedia secara online di laman :
https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data .

Atribut dalam dataset memiliki nilai numerik dan kategori yang dikumpulkan dari empat lokasi berbeda terkait diagnosis penyakit jantung. Jumlah observasi dari masing-masing lokasi adalah 304 observasi dari Cleveland, 293 dari Hungarian, 123 dari Switzerland, dan 200 dari Long Beach VA, sehingga totalnya mencakup 920 observasi. Dataset yang digunakan berjumlah 920 dengan 14 fitur yang dijelaskan lebih lanjut dalam tabel 1.

Tabel 1. Atribut Dataset

No	Atribut	Kategori	Deskripsi
1.	Umur	-	Menunjukkan usia individu
2.	Jenis Kelamin	female, male	Menunjukkan jenis kelamin individu
3.	Cp (Jenis Nyeri Dada)	asymptomatic, atypical angina, non angina, typical angina	Mengindikasikan jenis nyeri dada yang dialami oleh individu
4.	Trestbps (Tekanan Darah)	-	Menunjukkan nilai tekanan darah dalam satuan mmHg
5.	Chol (Kolesterol Serum)	-	Menunjukkan kadar kolesterol serum dalam mg/dl
6.	Fbs (Gula Darah Puasa)	false, true	Membandingkan kadar gula darah puasa seseorang terhadap batas 120 mg/dl

7.	Restecg	lv hypertrophy, normal, st-t abnormality	Menunjukkan hasil elektrokardiogram
8.	Thalach (Detak Jantung Maksimum)	-	Menunjukkan detak jantung maksimum yang dicapai individu
9.	Exang (Angina yang Dipicu Oleh Latihan)	false, true	Menunjukkan apakah individu mengalami angina yang dipicu oleh latihan
10.	Oldpeak (Depresi ST yang Diinduksi Oleh Latihan)	-	Menunjukkan nilai depresi ST yang dapat berupa bilangan integer atau float
11.	Slope (Slope dari Segmen ST Puncak Latihan)	downsloping, flat, upsloping	Menunjukkan bentuk kemiringan segmen ST pada puncak latihan
12.	Ca (Jumlah Pembuluh Darah Utama yang Diberi Warna)	0-3 values	Menunjukkan jumlah pembuluh darah yang terlihat melalui fluoroskopi
13.	Thal	fixed defect, normal, reversable defect	Menunjukkan tipe thalassemia pada individu
14.	num	0 = tidak ada penyakit 1-4 = tingkat keparahan penyakit	Menunjukan diagnosis

3.3 Pra-Pemrosesan Data

Pada tahap pra-pemrosesan, dilakukan beberapa langkah untuk mempersiapkan dataset agar dapat digunakan dalam proses pelatihan model. Langkah pertama adalah menghapus atribut yang tidak diperlukan yaitu atribut id dan dataset, selanjutnya adalah memeriksa adanya data duplikat. Hasil pengecekan menunjukkan bahwa terdapat 2 data yang duplikat, sehingga tidak dilakukan penghapusan baris pada tahap ini.

Tahap berikutnya adalah menangani data yang memiliki nilai kosong (missing value). Proses ini dibedakan berdasarkan tipe data. Untuk kolom-kolom yang bersifat numerik, nilai kosong diimputasi dengan menggunakan nilai rata-rata (mean), sedangkan untuk kolom kategorikal, nilai kosong diisi menggunakan nilai yang paling sering muncul (modus). Setelah proses imputasi selesai, kolom-kolom kategorikal dikonversi ke dalam bentuk numerik menggunakan teknik *One Hot Encoding*, agar dapat diproses oleh algoritma pembelajaran mesin yang umumnya hanya menerima input numerik. Seluruh atribut numerik dan hasil encoding dari atribut kategorikal kemudian digabungkan kembali menjadi satu

kesatuan DataFrame yang merepresentasikan fitur lengkap dari dataset yang telah dibersihkan.

Selanjutnya, dilakukan transformasi pada kolom target (num) yang awalnya bersifat multi kelas dengan lima label (0 hingga 4). Untuk menyederhanakan permasalahan menjadi klasifikasi biner, kelas 1, 2, 3, dan 4 digabung menjadi satu kelas (positif penyakit jantung) dan diberi label 1, sedangkan kelas 0 dipertahankan sebagai label 0 (negatif penyakit jantung). Dataset yang telah bersih dan siap ini kemudian dibagi menjadi dua bagian, yaitu data latih (80%) dan data uji (20%) menggunakan metode pembagian acak. Namun, karena distribusi kelas target tidak seimbang (jumlah data pada kelas 1 lebih banyak dibanding kelas 0), dilakukan proses penyeimbangan data menggunakan teknik **SMOTE (Synthetic Minority Oversampling Technique)**. SMOTE bekerja dengan menciptakan contoh sintetis dari kelas minoritas pada data latih, sehingga model yang dilatih menjadi lebih adil dan tidak bias terhadap kelas mayoritas. Seluruh langkah ini dilakukan untuk memastikan bahwa data yang digunakan dalam pelatihan model XGBoost memiliki kualitas yang baik dan representatif terhadap permasalahan klasifikasi penyakit jantung yang ingin diselesaikan.

3.4 Pembagian Data (Train-Test Split)

Setelah proses pra-pemrosesan selesai dilakukan, data dibagi menjadi dua subset, yaitu data latih (training set) dan data uji (testing set). Pembagian ini bertujuan agar model dapat dilatih menggunakan data latih, dan kemudian dievaluasi kinerjanya menggunakan data uji yang belum pernah dilihat sebelumnya oleh model. Dengan demikian, dapat dinilai seberapa baik model dapat melakukan generalisasi terhadap data baru. Pembagian dilakukan dengan menggunakan fungsi train_test_split dari library *scikit-learn*, dengan rasio 80:20, artinya 80% data digunakan sebagai data latih dan 20% sisanya digunakan sebagai data uji. Proses ini juga disertai dengan pengaturan random_state agar hasil pembagian bersifat konsisten.

Namun sebelum proses pelatihan dilakukan, ditemukan bahwa distribusi kelas target pada data latih bersifat tidak seimbang, di mana jumlah data pada kelas positif (mengidap penyakit jantung) lebih banyak dibandingkan dengan kelas negatif (tidak mengidap penyakit jantung). Ketidakseimbangan ini berpotensi menyebabkan model menjadi bias terhadap kelas mayoritas. Oleh karena itu, diterapkan metode SMOTE (Synthetic Minority Oversampling Technique) pada data latih untuk menyeimbangkan jumlah sampel antar kelas. Penting untuk dicatat bahwa SMOTE hanya diaplikasikan pada data latih saja, agar informasi dari data uji tidak “bocor” ke dalam proses pelatihan dan menghindari *data leakage*. Dengan demikian, model dapat belajar dari data yang telah seimbang dan tetap diuji secara adil terhadap data yang merepresentasikan kondisi dunia nyata.

3.5 SMOTE

Dalam proses pelatihan model pembelajaran mesin, keseimbangan jumlah data antara kelas target sangat penting untuk mencegah bias model terhadap kelas mayoritas. Pada dataset ini, setelah dilakukan transformasi label target menjadi klasifikasi biner (0 = tidak mengidap penyakit jantung, 1 = mengidap penyakit jantung), ditemukan ketidakseimbangan jumlah data antara kedua kelas. Jumlah data pada kelas 1 jauh lebih banyak dibandingkan

dengan kelas 0. Ketidakseimbangan ini berpotensi menyebabkan model lebih sering memprediksi kelas yang dominan, sehingga mengurangi akurasi dalam mendeteksi kasus dari kelas minoritas.

Untuk mengatasi masalah ini, diterapkan metode **SMOTE (Synthetic Minority Oversampling Technique)** pada data latih. SMOTE bekerja dengan cara membuat data sintetis (buatan) untuk kelas minoritas berdasarkan tetangga terdekat dari sampel yang ada. Teknik ini tidak hanya menyalin data yang ada, tetapi menghasilkan sampel baru yang menyerupai data minoritas secara alami, sehingga mengurangi risiko overfitting.

Proses SMOTE hanya diterapkan pada data latih untuk mencegah *data leakage*. Sebelum dilakukan SMOTE, jumlah data latih adalah **736 baris**, dengan **kelas 0 sebanyak 336 sampel** dan **kelas 1 sebanyak 400 sampel**. Setelah dilakukan SMOTE, jumlah data latih meningkat menjadi **800 baris**, dengan distribusi yang seimbang yaitu **kelas 0 sebanyak 400 sampel** dan **kelas 1 sebanyak 400 sampel**. Dengan distribusi yang seimbang ini, model dapat belajar secara lebih adil antara kedua kelas dan diharapkan menghasilkan performa yang lebih baik saat diuji pada data yang belum pernah dilihat sebelumnya.

3.6 Model XGBoost

Setelah data selesai melalui tahap pra-pemrosesan dan penyeimbangan kelas dengan SMOTE, langkah selanjutnya adalah membangun dan melatih model klasifikasi menggunakan algoritma XGBoost (Extreme Gradient Boosting). XGBoost merupakan salah satu algoritma *ensemble learning* berbasis *gradient boosting* yang terkenal karena efisiensi, akurasi, dan kemampuannya menangani data dalam skala besar serta menangani fitur yang tidak seimbang. Untuk memperoleh performa terbaik dari model, dilakukan proses optimasi hiperparameter menggunakan metode Bayesian Optimization dengan bantuan pustaka hyperopt. Dalam proses ini, dilakukan eksplorasi terhadap beberapa parameter penting XGBoost seperti:

1. learning_rate : tingkat pembelajaran model, yaitu seberapa besar kontribusi setiap pohon individu terhadap prediksi akhir. Nilai learning_rate yang kecil membuat proses pembelajaran menjadi lebih lambat namun stabil, dan cenderung menghasilkan generalisasi yang lebih baik. Sebaliknya, nilai yang terlalu besar dapat mempercepat pelatihan tetapi berisiko menyebabkan overfitting.
2. max_depth : kedalaman maksimum setiap pohon keputusan yang dibentuk. Semakin besar nilai max_depth, semakin kompleks struktur pohon yang dihasilkan, sehingga model dapat menangkap pola yang lebih rumit. Namun, nilai yang terlalu besar dapat menyebabkan model belajar terlalu detail pada data pelatihan dan kehilangan kemampuan generalisasi.
3. n_estimator : jumlah pohon keputusan, yang akan dibangun secara bertahap dalam proses boosting. Setiap pohon dibangun berdasarkan kesalahan dari pohon sebelumnya. Penambahan jumlah pohon dapat meningkatkan akurasi, namun juga menambah waktu komputasi dan potensi overfitting jika tidak dikontrol dengan baik.
4. subsample : proporsi data latih yang digunakan untuk membentuk setiap pohon.

5. colsample_bytree : Menentukan proporsi fitur (kolom) yang digunakan untuk setiap pohon.
6. gamma : Adalah parameter regularisasi yang mengontrol ambang minimum pengurangan loss yang dibutuhkan agar pemisahan (split) pada node dilakukan.
7. min_child_weight : Bobot minimum yang diperlukan untuk membuat sebuah daun pada pohon individu.
8. reg_lambda : parameter regularisasi L2 yang digunakan untuk mencegah overfitting dengan menambahkan penalti terhadap besarnya bobot dalam model.

BAB IV

IMPLEMENTASI

4.1. Hasil Coding

Membaca dan menghapus kolom yang tidak dibutuhkan:

Dataset heart.csv dibaca menggunakan Pandas, lalu dilakukan pengecekan nilai kosong dan data duplikat, Duplikasi ini dihapus menggunakan drop_duplicates(), dan hasilnya disimpan ke dalam variabel no_dup_data untuk digunakan pada tahap selanjutnya.

```
dataFramePenyakitJantung = pd.read_csv("./dataset/heart.csv")

# Tidak ada missing value
# print(f"Missing Value: \n{ dataFramePenyakitJantung.isnull().sum() }")

# duplicate data

# dataFramePenyakitJantung[dataFramePenyakitJantung.duplicated(keep="first")]

dataFramePenyakitJantung.duplicated().sum()
dataFramePenyakitJantung[dataFramePenyakitJantung.duplicated(keep=False)].sort_values(by=list(dataFramePenyakitJantung.columns))
no_dup_data = dataFramePenyakitJantung.drop_duplicates()

no_dup_data
```

Preprocessing Data:

- Mendeteksi data duplikat, ditemukan data duplikat sebanyak 2.

```
# duplikat 2
print(dataFramePenyakitJantung.duplicated().sum())
dataFramePenyakitJantung = dataFramePenyakitJantung.drop_duplicates()
print(dataFramePenyakitJantung.duplicated().sum())
```

✓ 0.0s

```
2
0
```

- lalu menangani missing value dengan cara melakukan imputasi ke dalam missing value untuk atribut yang numeric diisi dengan rata rata untuk atribut yang kategorikal di isi dengan modus.

```
# Melakukan imputasi ke missing value
numerical_cols = dataFramePenyakitJantung.select_dtypes(include=np.number).columns.tolist()
categorical_cols = dataFramePenyakitJantung.select_dtypes(include='object').columns.tolist()

imputer_numerical = SimpleImputer(strategy='mean')

imputer_numerical.fit(dataFramePenyakitJantung[numerical_cols])
df_imputed_num = imputer_numerical.transform(dataFramePenyakitJantung[numerical_cols])


imputer_categorical = SimpleImputer(strategy='most_frequent')

imputer_categorical.fit(dataFramePenyakitJantung[categorical_cols])

df_imputed_cat = imputer_categorical.transform(dataFramePenyakitJantung[categorical_cols])

imputed_df = pd.DataFrame(df_imputed_num, columns=numerical_cols, index=dataFramePenyakitJantung.index)
imputed_df[categorical_cols] = df_imputed_cat
```

- lalu melakukan encoding di atribut kategorikal menggunakan One Hot Encoding memecah setiap kategori menjadi fitur, lalu melakukan perubahan dari data multiclass menjadi binary class dengan cara merubah setiap num = 1, 2, 3, 4 menjadi 1

```
# Mengubah data kategorikal menjadi numeric menggunakan One-Hot Encoder
encoder = OneHotEncoder(handle_unknown='ignore', sparse_output=False)

encoder.fit(df_imputed_cat)

df_encoded_cat = encoder.transform(df_imputed_cat)


encoded_feature_names = encoder.get_feature_names_out(categorical_cols)
categorical_df = pd.DataFrame(df_encoded_cat, columns=encoded_feature_names, index=dataFramePenyakitJantung.index)
num_df = pd.DataFrame(df_imputed_num, columns=numerical_cols, index=dataFramePenyakitJantung.index)

df_processed = pd.concat([num_df, categorical_df], axis=1)

df_processed.to_csv("./data_after_preprocess.csv")
```

Pembagian Data (Train-Test Split)

- Membagi dataset menjadi data latih (80% dari total data) untuk melatih model, dan data uji (sisanya 20%) untuk menguji performanya.

```
# Split Data
X_train, X_test, y_train, y_test = model_selection.train_test_split(X_resampled, y_resampled, test_size=0.2)
```

Penyeimbangan Data dengan SMOTE:

Dataset yang telah telah di split kita ambil data train yang berjumlah 80% (dari datase yang telah bersih) kemudian diseimbangkan menggunakan metode SMOTE untuk mengatasi masalah ketidakseimbangan kelas dengan k-neighbor nya 5 . Proses ini meningkatkan jumlah data pada kelas minoritas sehingga kedua kelas (0 dan 1) memiliki jumlah sampel yang sama, yaitu 734 sampel, dengan total data menjadi 794 baris.

```python
# K = 5, untuk menghasilkan akurasi yang lebih baik
smote = SMOTE(random_state=42, k_neighbors=5)

X_resampled, y_resampled = smote.fit_resample(X_train, y_train)

print(f"Data train Sebelum di Oversampling: {X_train.shape[0]} baris")
print(f"Data train Sebelum di Oversampling: {X_resampled.shape[0]} baris")

print()
print("Jumlah Class 0 dan 1, sebelum di oversampling")
print(y_train.value_counts())
print()
print("Jumlah Class 0 dan 1, setalah di oversampling")
print(y_resampled.value_counts())
```

Tuning Parameter Model XGBoost dengan GridSearchCV

Untuk memperoleh performa terbaik dari algoritma XGBoost, dilakukan pencarian parameter optimal menggunakan GridSearchCV. Parameter yang diuji meliputi n_estimators, learning_rate, max_depth, subsample, colsample_bytree, dan parameter regularisasi seperti reg_alpha dan reg_lambda. Proses tuning menggunakan 5-fold cross-validation dan metrik evaluasi berupa akurasi. Hasil dari pencarian parameter terbaik kemudian disimpan dalam model akhir yang digunakan untuk evaluasi.

```python
param_grid_xgboost = {
    'n_estimators': [300, 500],
    'learning_rate': [0.01, 0.05, 0.1, 0.2],
    'max_depth': [3, 5, 7, 9],
    'subsample': [0.7, 0.8, 0.9, 1.0],
    'colsample_bytree': [0.7, 0.8, 0.9, 1.0],
    'gamma': [0.2, 0.5],
    'reg_alpha': [0, 0.001, 0.01, 0.1],
    'reg_lambda': [1, 0.1],
    'min_child_weight': [1, 3, 5],
}


model_xgb = xgb.XGBClassifier(
    objective='binary:logistic',
    eval_metric='logloss',
    use_label_encoder=False,
    n_estimators=100,      # Jumlah pohon
    learning_rate=0.1,     # Laju pembelajaran
    max_depth=5,           # Kedalaman maksimum pohon
    subsample=0.8,         # Fraksi sampel untuk setiap pohon
    colsample_bytree=0.5,  # Fraksi fitur untuk setiap pohon
    gamma=0.1,             # Minimum loss reduction
    random_state=42
)

grid_search = GridSearchCV(
    estimator=model_xgb,
    param_grid=param_grid_xgboost,
    scoring='accuracy', # Metrik untuk optimasi
    cv=5,               # Jumlah fold untuk cross-validation
    n_jobs=-1,          # Gunakan semua core CPU
    verbose=1
)

model_xgb.fit(X_train, y_train)

grid_search.fit(X_train, y_train)
best_xgb_model = grid_search.best_estimator_
```

Prediksi pada Data Uji

Menguji performa model XGBoost dengan memprediksi data uji. Hasil prediksi dibandingkan dengan label asli untuk menghitung akurasi dan menampilkan classification report. Evaluasi dilakukan pada model awal dan model terbaik dari hasil Bayesian Optimization berdasarkan akurasi nya.

```
y_pred = model_xgb.predict(X_test)
y_pred_proba = model_xgb.predict_proba(X_test)[:, 1]

y_pred_best = best_xgb_model.predict(X_test)
y_pred_proba_best = best_xgb_model.predict_proba(X_test)[:, 1]
```

```
print("model xgb")
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")

print("\nClassification Report:")
print(classification_report(y_test, y_pred))

print("Best Model XGB")
accuracy = accuracy_score(y_test, y_pred_best)
print(f"Accuracy: {accuracy:.4f}")

print("\nClassification Report:")
print(classification_report(y_test, y_pred_best))
```

```
model xgb
Accuracy: 0.8333

Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.83      0.85        36
           1       0.81      0.83      0.82        30

    accuracy                           0.83        66
   macro avg       0.83      0.83      0.83        66
weighted avg       0.83      0.83      0.83        66

Best Model XGB
Accuracy: 0.8182

Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.83      0.83        36
           1       0.80      0.80      0.80        30

    accuracy                           0.82        66
   macro avg       0.82      0.82      0.82        66
weighted avg       0.82      0.82      0.82        66
```

Menyimpan Model XGBoost Terbaik ke dalam menjadi file .pkl

Menyimpan model XGBoost terbaik hasil Grid Search ke dalam file .pkl menggunakan modul pickle, agar dapat digunakan kembali tanpa perlu melatih ulang.

```
nama_model = "klasifikasi_penyakit_jantung_xgboost.pkl"

with open(nama_model, 'wb') as file:
    pickle.dump(best_xgb_model, file)
```

Membaca Model yang Telah Disimpan dan Menyiapkan Data Pasien Baru

Memuat kembali model XGBoost dari file .pkl yang telah disimpan sebelumnya. Selanjutnya, dibuat data pasien baru dalam bentuk dictionary, lalu dikonversi menjadi DataFrame untuk keperluan Uji Coba Model.

```python
import pickle
import pandas as pd

nama_model = "klasifikasi_penyakit_jantung_xgboost.pkl"

with open(nama_model, 'rb') as file:
    model_xgb = pickle.load(file)

data_pasien_baru = {
    'age': [58, 54],
    'sex': [0, 1],
    'cp': [2, 0],
    'trestbps': [130, 124],
    'chol': [260, 266],
    'fbs': [0, 0],
    'restecg': [1, 0],
    'thalach': [150, 109],
    'exang': [0, 1],
    'oldpeak': [1.5, 2.2],
    'slope': [1, 1],
    'ca': [0, 1],
    'thal': [2, 3]
}

new_df = pd.DataFrame(data_pasien_baru)
new_df
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	58	0	2	130	260	0	1	150	0	1.5	1	0	2
1	54	1	0	124	266	0	0	109	1	2.2	1	1	3

```python
model_xgb.predict(new_df)
```

```
array([1, 0])
```

4.2. User Interface (Aplikasi Streamlit)

Sebuah user interface (UI) berbasis web dikembangkan untuk mempermudah penggunaan model. UI ini memungkinkan pengguna memasukkan 13 fitur data pasien dan mendapatkan hasil prediksi secara instan.

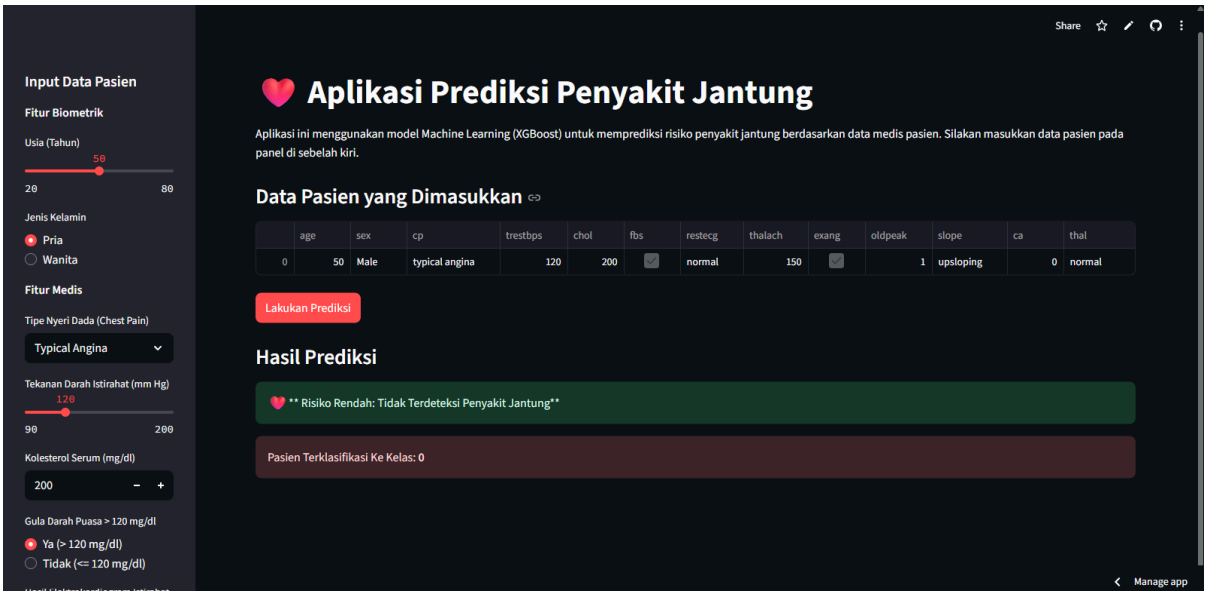
Kode Implementasi UI :

```python
import streamlit as st
import pandas as pd
import joblib
import numpy as np # Import numpy jika Anda menggunakan np.array() atau sejenisnya

st.set_page_config(page_title="Prediksi Risiko Penyakit Jantung", layout="centered")


# --- Muat Model yang Sudah Dilatih ---
# Pastikan file model_klasifikasi_xgboost.pkl ada di direktori yang sama
# atau berikan path lengkap ke file tersebut.
try:
    model = joblib.load('./klasifikasi_penyakit_jantung_xgboost.pkl')
    st.success("Model berhasil dimuat!")
except FileNotFoundError:
    st.error("Error: File model 'model_klasifikasi_xgboost.pkl' tidak ditemukan.")
    st.info("Pastikan Anda sudah melatih model XGBoost dan menyimpannya dengan nama tersebut.")
    st.stop() # Hentikan aplikasi jika model tidak ditemukan

# --- JIKA ANDA MENGGUNAKAN SCALER/ENCODER SAAT PELATIHAN, MUAT JUGA DI SINI ---
# Contoh:
# try:
#     scaler = joblib.load('scaler_data.pkl') # Ganti dengan nama file scaler Anda
#     st.success("Scaler berhasil dimuat!")
# except FileNotFoundError:
#     st.warning("Peringatan: File scaler tidak ditemukan. Jika model Anda dilatih dengan data yang diskalakan, prediksi mungkin tidak akurat.")
#     scaler = None # Atur None jika tidak ada scaler

# --- Judul Aplikasi Streamlit ---
st.title("🫀 Prediksi Risiko Penyakit Jantung")
st.markdown("Aplikasi ini memprediksi risiko penyakit jantung berdasarkan input fitur pasien.")

st.write("---")

# --- Form Input untuk Fitur Pasien ---
st.header("Input Data Pasien")

# Mengumpulkan input dari pengguna
col1, col2, col3 = st.columns(3)
```

```python
with col1:
    age = st.slider("Usia", 1, 120, 50)
    sex = st.selectbox("Jenis Kelamin", options=[(0, "Perempuan"), (1, "Laki-laki")], format_func=lambda x: x[1])[0] # 0: Perempuan, 1: Laki-laki
    cp = st.selectbox("Tipe Nyeri Dada (cp)", options=[0, 1, 2, 3], help="0: Asymptomatic, 1: Atypical Angina, 2: Non-Anginal Pain, 3: Typical Angina")
    trestbps = st.number_input("Tekanan Darah Istirahat (trestbps)", 80, 200, 120, help="Tekanan darah sistolik saat istirahat (mm Hg)")

with col2:
    chol = st.number_input("Kolesterol Serum (chol)", 100, 600, 200, help="Kolesterol serum dalam mg/dl")
    fbs = st.selectbox("Gula Darah Puasa > 120 mg/dl (fbs)", options=[(0, "Tidak"), (1, "Ya")], format_func=lambda x: x[1])[0] # 0: False, 1: True
    restecg = st.selectbox("Hasil Elektrokardiogram Saat Istirahat (restecg)", options=[0, 1, 2], help="0: Normal, 1: ST-T wave abnormality, 2: Left ventricular hypertrophy")
    thalach = st.number_input("Detak Jantung Maksimal Tercapai (thalach)", 70, 220, 150, help="Detak jantung maksimal yang tercapai saat uji stres")

with col3:
    exang = st.selectbox("Angina Akibat Olahraga (exang)", options=[(0, "Tidak"), (1, "Ya")], format_func=lambda x: x[1])[0] # 0: No, 1: Yes
    oldpeak = st.number_input("Depresi ST Akibat Olahraga Relatif Terhadap Istirahat (oldpeak)", 0.0, 6.0, 1.0, step=0.1, help="Penurunan ST yang diinduksi olahraga relatif terhadap istirahat")
    slope = st.selectbox("Kemiringan Segmen ST Puncak Latihan (slope)", options=[0, 1, 2], help="0: Upsloping, 1: Flat, 2: Downsloping")
    ca = st.selectbox("Jumlah Pembuluh Darah Besar (ca)", options=[0, 1, 2, 3, 4], help="Jumlah pembuluh darah besar (0-3) yang diwarnai oleh fluoroskopi")
    thal = st.selectbox("Thalassemia (thal)", options=[0, 1, 2, 3], help="0: Normal, 1: Fixed defect, 2: Reversable defect, 3: Other types (rarely used)") # Biasanya 1,2,3, tapi contoh data ada 0


# --- Tombol Prediksi ---
if st.button("Prediksi Risiko Penyakit Jantung"):
    # Kumpulkan semua input ke dalam dictionary
    input_data = {
        'age': [age],
        'sex': [sex],
        'cp': [cp],
        'trestbps': [trestbps],
        'chol': [chol],
        'fbs': [fbs],
        'restecg': [restecg],
        'thalach': [thalach],
        'exang': [exang],
        'oldpeak': [oldpeak],
        'slope': [slope],
        'ca': [ca],
        'thal': [thal]
    }

    # Buat DataFrame dari input
    # Penting: Pastikan urutan kolom sesuai dengan saat model dilatih!
    # Urutan ini harus sama dengan X_train Anda.
    input_df = pd.DataFrame(input_data)
```

```python
# --- LAKUKAN PRA-PEMROSESAN YANG SAMA DENGAN DATA PELATIHAN ---
# Jika Anda menggunakan scaler atau encoder, terapkan di sini.
# Contoh jika Anda menggunakan StandardScaler:
# if scaler:
#     input_processed = scaler.transform(input_df)
# else:
#     input_processed = input_df

input_processed = input_df # Untuk contoh ini, asumsi tidak ada pra-pemrosesan kompleks

# --- Buat Prediksi ---
prediction = model.predict(input_processed)
prediction_proba = model.predict_proba(input_processed)

st.write("---")
st.header("Hasil Prediksi")

if prediction[0] == 1:
    st.error("⚠️ Pasien Diprediksi **BERISIKO TINGGI** Terkena Penyakit Jantung")
    st.write(f"Terklasifikasi pada kelas: **{prediction[0]}**")
    st.write(f"Probabilitas risiko tinggi: **{prediction_proba[0][1]*100:.2f}%**")
else:
    st.write(f"Terklasifikasi pada kelas: **{prediction[0]}**")
    st.success("✅ Pasien Diprediksi **BERISIKO RENDAH** Terkena Penyakit Jantung")
    st.write(f"Probabilitas risiko rendah: **{prediction_proba[0][0]*100:.2f}%**")
```

Tangkapan Layar User Interface:



4.3 Evaluasi Model

Setelah proses pelatihan model XGBoost selesai dilakukan menggunakan parameter hasil optimasi, langkah selanjutnya adalah melakukan evaluasi kinerja model terhadap data uji. Evaluasi dilakukan untuk mengukur kemampuan generalisasi model dalam mengklasifikasikan data yang belum pernah dilihat sebelumnya. Salah satu metrik utama yang digunakan adalah akurasi, yang menunjukkan persentase prediksi yang benar dari total data uji. Model XGBoost yang dibangun menghasilkan akurasi sebesar 0.8207, yang menunjukkan bahwa sebagian besar prediksi model terhadap data uji sudah tepat. Classification report digunakan sebagai alat evaluasi yang memberikan gambaran menyeluruh mengenai performa model klasifikasi terhadap masing-masing kelas. Laporan ini mencakup metrik penting yaitu **precision**, **recall**, **f1-score**, dan **support** untuk setiap kelas, yaitu kelas 0 (tidak mengidap penyakit jantung) dan kelas 1 (mengidap penyakit jantung).

Hasil classification report menunjukkan bahwa model XGBoost memiliki kemampuan klasifikasi yang baik pada kedua kelas. Untuk kelas 0, model menghasilkan precision sebesar

0,77, recall sebesar 0,78, dan f1-score sebesar 0,78. Hal ini menunjukkan bahwa model cukup akurat dalam mengidentifikasi pasien yang tidak mengidap penyakit jantung, dengan kesalahan prediksi yang relatif rendah. Sementara itu, pada kelas 1, model mencatatkan precision sebesar 0,85, recall sebesar 0,85, dan f1-score sebesar 0,85. Nilai-nilai ini menunjukkan bahwa model sangat baik dalam mengenali pasien yang mengidap penyakit jantung, dengan keseimbangan antara ketepatan prediksi dan kemampuan mendeteksi kasus yang sebenarnya positif.

Nilai rata-rata metrik berdasarkan pendekatan macro average dan weighted average berada pada kisaran 0,84 hingga 0,85. Nilai macro average mengindikasikan bahwa model memiliki performa yang konsisten pada kedua kelas tanpa mempertimbangkan jumlah sampel, sedangkan weighted average memberikan gambaran performa model dengan mempertimbangkan proporsi data pada masing-masing kelas. Dari hasil tersebut, dapat disimpulkan bahwa model XGBoost yang digunakan dalam penelitian ini mampu memberikan prediksi yang akurat dan seimbang dalam mendeteksi keberadaan penyakit jantung, baik untuk kelas mayoritas maupun minoritas.

BAB V

HASIL DAN KESIMPULAN

5.1 Hasil

Implementasi model XGBoost untuk diagnosis penyakit jantung dilakukan setelah serangkaian tahap *preprocessing* dan optimasi. Data diperoleh dari Kaggle ("UCI Heart Disease Data") dan melewati beberapa tahapan penting:

Penanganan Duplikat dan Nilai Hilang: ditemukan 2 data duplikat. Nilai kosong pada kolom numerik diisi dengan nilai rata-rata (*mean*), sementara kolom kategorikal diisi dengan nilai yang paling sering muncul (*modus*).

age	trestbps	chol	thalch	oldpeak	ca	num	sex	cp	fbs	restecg	exang	slope	thal
63.0	145.000000	233.0	150.000000	2.300000	0.000000	0.0	Male	typical angina	True	lv hypertrophy	False	downsloping	fixed defect
67.0	160.000000	286.0	108.000000	1.500000	3.000000	2.0	Male	asymptomatic	False	lv hypertrophy	True	flat	normal
67.0	120.000000	229.0	129.000000	2.600000	2.000000	1.0	Male	asymptomatic	False	lv hypertrophy	True	flat	reversable defect
37.0	130.000000	250.0	187.000000	3.500000	0.000000	0.0	Male	non-anginal	False	normal	False	downsloping	normal
41.0	130.000000	204.0	172.000000	1.400000	0.000000	0.0	Female	atypical angina	False	lv hypertrophy	False	upsloping	normal
...	...	...	...	...	...	...	...	...	...	...	...	...	...
54.0	127.000000	333.0	154.000000	0.000000	0.676375	1.0	Female	asymptomatic	True	st-t abnormality	False	flat	normal
62.0	132.137369	139.0	137.542294	0.880841	0.676375	0.0	Male	typical angina	False	st-t abnormality	False	flat	normal
55.0	122.000000	223.0	100.000000	0.000000	0.676375	2.0	Male	asymptomatic	True	st-t abnormality	False	flat	fixed defect
58.0	132.137369	385.0	137.542294	0.880841	0.676375	0.0	Male	asymptomatic	True	lv hypertrophy	False	flat	normal
62.0	120.000000	254.0	93.000000	0.000000	0.676375	1.0	Male	atypical angina	False	lv hypertrophy	True	flat	normal

Encoding Data Kategorikal: Kolom-kolom kategorikal dikonversi ke dalam bentuk numerik menggunakan teknik One-Hot Encoding.

age	trestbps	chol	thalch	oldpeak	ca	num	sex_Female	sex_Male	cp_asymptomatic	...	restecg_normal	restecg_st-t abnormality	exang_False	exang_True	slope_downsloping	slope_flat	slope_upsloping	thal_fixed defect	thal_normal	thal_reversable defect
63.0	145.0	233.0	150.0	2.3	0.0	0.0	0.0	1.0	0.0	...	0.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0
67.0	160.0	286.0	108.0	1.5	3.0	2.0	0.0	1.0	1.0	...	0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0
67.0	120.0	229.0	129.0	2.6	2.0	1.0	0.0	1.0	1.0	...	0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	1.0
37.0	130.0	250.0	187.0	3.5	0.0	0.0	0.0	1.0	0.0	...	1.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0
41.0	130.0	204.0	172.0	1.4	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0

Transformasi Target Biner: Kolom target 'num' yang awalnya memiliki lima label (0 hingga 4) diubah menjadi klasifikasi biner, di mana kelas 1,2,3,4 digabung menjadi kelas 1 (mengidap penyakit jantung) dan kelas 0 tetap sebagai 0 (tidak mengidap penyakit jantung).

num
0
1
1
0
0
...
1
0
1
0
1

Pembagian Data (*Train-Test Split*): Data dibagi menjadi 80% data latih dan 20% data uji.

Penyeimbangan Kelas dengan SMOTE: Karena adanya ketidakseimbangan pada kelas target (jumlah data kelas 1 lebih banyak dari kelas 0) , teknik SMOTE (Synthetic Minority Oversampling Technique) diterapkan pada data latih saja. Sebelum SMOTE, data latih memiliki 336 sampel untuk kelas 0 dan 400 sampel untuk kelas 1. Setelah SMOTE, kedua kelas seimbang dengan masing-masing 400 sampel, menjadikan total data latih 800 baris.

```
Data train Sebelum di Oversampling: 736 baris
Data train Sebelum di Oversampling: 800 baris

Jumlah Class 0 dan 1, sebelum di oversampling
num
1    400
0    336
Name: count, dtype: int64

Jumlah Class 0 dan 1, setalah di oversampling
num
1    400
0    400
Name: count, dtype: int64
```

Pemodelan dan Optimasi *Hyperparameter*: Model klasifikasi dibangun menggunakan algoritma XGBoost. Untuk mendapatkan kinerja terbaik, optimasi *hyperparameter* dilakukan menggunakan Bayesian Optimization dengan *library* Hyperopt. Parameter yang dioptimalkan meliputi learning_rate, max_depth, n_estimators, subsample, colsample_bytree, gamma, min_child_weight, dan reg_lambda.

Setelah proses optimasi, model XGBoost dievaluasi pada data uji yang belum pernah dilihat sebelumnya. Hasil evaluasi menunjukkan kinerja model sebagai berikut:

Akurasi Model XGBoost: Model XGBoost yang dibangun menghasilkan akurasi sebesar 0.8207. Ini mengindikasikan bahwa sebagian besar prediksi model terhadap data uji sudah tepat.

Laporan Klasifikasi (*Classification Report*):

```
Best Model XGB
Accuracy: 0.8207

Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.78      0.78        73
           1       0.85      0.85      0.85       111

    accuracy                           0.82       184
   macro avg       0.81      0.81      0.81       184
weighted avg       0.82      0.82      0.82       184
```

Hasil ini menunjukkan kemampuan klasifikasi yang baik pada kedua kelas, dengan model sangat baik dalam mengenali pasien yang mengidap penyakit jantung (kelas 1) dan cukup akurat dalam mengidentifikasi pasien yang tidak mengidap penyakit jantung (kelas 0)

5.2 Kesimpulan

Berdasarkan hasil implementasi dan evaluasi, dapat disimpulkan bahwa **model XGBoost yang dibangun dan dioptimalkan mampu memberikan prediksi yang akurat dan seimbang dalam mendeteksi keberadaan penyakit jantung**. Akurasi keseluruhan model sebesar **0.8207** menunjukkan kinerja yang solid dalam mengklasifikasikan data penyakit jantung yang belum pernah dilihat sebelumnya.

Penerapan *preprocessing* data yang komprehensif, termasuk penanganan nilai hilang, *One-Hot Encoding* untuk fitur kategorikal, transformasi target biner, dan terutama penggunaan teknik **SMOTE** untuk mengatasi ketidakseimbangan kelas pada data latih, terbukti krusial dalam membangun model yang tidak bias dan memiliki kinerja yang baik untuk kedua kelas (mayoritas dan minoritas).

Optimisasi *hyperparameter* menggunakan **Bayesian Optimization** juga berperan penting dalam menyetel model XGBoost sehingga mencapai kinerja optimal. Model XGBoost yang dihasilkan menunjukkan *precision*, *recall*, dan F1-*score* yang baik untuk kedua kelas, menegaskan kemampuannya dalam membedakan antara pasien yang mengidap dan tidak mengidap penyakit jantung secara efektif. Model yang telah disimpan (klasifikasi_penyakit_jantung_xgboost.pkl) juga dapat dimuat kembali dan digunakan untuk prediksi pada data pasien baru, serta telah diintegrasikan ke dalam antarmuka pengguna berbasis Streamlit untuk kemudahan akses.

DAFTAR PUSTAKA

Budianto, Akhmad Ghiffary, and Akhmad Syarief. "Analisis Pengaruh Pengurangan Dimensi Data Pada Keakuratan Prediksi Penyakit Jantung dengan Menggunakan SVM Linear." *Jurnal Ilmiah Teknik dan Manajemen Industri* 3.1 (2023): 80-91.

Sausan, Pratiwi, D. M., & Mufidah, L. (2024). *Perbandingan metode Decision Tree Classifier dan XGBoost Classifier dalam memprediksi penyakit jantung*. CENTIVE: Conference on Electrical Engineering, Informatics, Industrial Technology, and Creative Media.

Sidiq, S., Alfian, A., & Mabrur, N. S. (2025). Pengembangan model prediksi risiko diabetes menggunakan pendekatan AdaBoost dan teknik *oversampling* SMOTE. *Jurnal Ilmiah Informatika dan Ilmu Komputer (JIMA-ILKOM)*, 4(1), 13–23.

A. U. Dullah, A. Y. Darmawan, D. A. A. Pertiwi, and Jumanto, "Extreme Gradient Boosting Model with SMOTE for Heart Disease Classification," *Universitas Negeri Semarang*, Jan. 2025.

G. J. Kurniarwan, C. Dewi, and M. A. Rahman, "Penerapan Machine Learning Extreme Gradient Boosting dalam Klasifikasi Potensi Tsunami Berdasarkan Data Gempa Bumi," *J. Pengemb. Tek. Inf. dan Ilmu Komput.*, vol. 9, no. 2, pp. xx–xx, Feb. 2025.

Hasanah, S. H. *Application of Machine Learning for Heart Disease Classification Using Naive Bayes*. Jurnal Matematika MANTIK, vol. 8, no. 1, 2022, pp. 68–77.

Azizah, S. R., et al. *Kombinasi Seleksi Fitur Berbasis Filter dan Wrapper Menggunakan Naive Bayes pada Klasifikasi Penyakit Jantung*. Jurnal Teknologi Informasi dan Ilmu Komputer, vol. 10, no. 6, 2023, pp. 1361–1368.

Surya, A. A., and Yamasari, Y. *Penerapan Algoritma Naïve Bayes (NB) untuk Klasifikasi Penyakit Jantung*. Journal of Informatics and Computer Science (JINACS), vol. 5, no. 3, 2024, pp. 447–455.

Am, A. N., Nurkholifah, M., and Oktorina, F. K. *Analisa Penyakit Jantung Menggunakan Algoritma Naïve Bayes*. Journal of System and Computer Engineering, vol. 4, no. 1, 2023, pp. 26–36.

Akil, I., and Chaidir, I. *Classification of Heart Disease Diagnoses Using Gaussian Naïve Bayes*. Komputasi: Jurnal Ilmiah Ilmu Komputer dan Matematika, vol. 21, no. 2, 2024, pp. 31–36.

Selayanti, N., et al. *Penerapan Machine Learning Algoritma Random Forest Untuk Prediksi Penyakit Jantung*. Prosiding Seminar Nasional Sains Data, vol. 4, no. 1, 2024, pp. 895–906.

Filemon, J. K., and Senjaya, W. F. *Prediksi Penyakit Jantung Berdasarkan Indikator-Indikator Kesehatan*. Jurnal STRATEGI-Jurnal Maranatha, vol. 6, no. 2, 2024, pp. 272–283.

Agusyul, A. Y. *Prediksi Penyakit Jantung Menggunakan Algoritma Random Forest*. Jurnal Minfo Polgan, vol. 12, no. 2, 2023.

Oktafiani, R., Hermawan, A., and Avianto, D. *Max Depth Impact on Heart Disease Classification: Decision Tree and Random Forest*. Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 8, no. 1, 2024, pp. 160–168.

Alfajr, N. H., and Defiyanti, S. *Prediksi Penyakit Jantung Menggunakan Metode Random Forest dan Penerapan Principal Component Analysis (PCA)*. Jurnal Informatika dan Teknik Elektro Terapan, vol. 12, no. 3S1, 2024.

Amelia, Y. *Perbandingan Metode Machine Learning untuk Mendeteksi Penyakit Jantung*. Idealis: Indonesia Journal Information System, 2023.

Rahman, H., Agusman, R., and Sutabri, T. *Model Prediksi Penyakit Jantung Menggunakan Machine Learning*. Jurnal Ilmiah Betrik, 2024.

Sausan, P., Pratiwi, D. M., and Mufidah, L. *Perbandingan Metode Decision Tree Classifier dan XGBoost Classifier dalam Memprediksi Penyakit Jantung*. CENTIVE: Conference on Electrical Engineering, Informatics, Industrial Technology, and Creative Media, 2024.

Dullah, A. U., Darmawan, A. Y., Pertiwi, D. A. A., and Jumanto. *Extreme Gradient Boosting Model with SMOTE for Heart Disease Classification*. JISKA (Jurnal Informatika Sunan Kalijaga), 2025.

Darmawan, D. B., Azahwa, I. R., Saputra, R. W., Septiadi, R., and Rosyani, P. *Klasifikasi Penyakit Jantung Menggunakan Extreme Gradient Boosting (XGBoost)*. JRIIN: Jurnal Riset Informatika dan Inovas, 2025.