COVER PAGE

Nama Lengkap: Happy Victor Jayata Karundeng

NIM: 2802540692

Judul Proyek: Pengembangan Model Machine Learning untuk Prediksi Gagal Jantung (Deployment Menggunakan Streamlit)

Pendahuluan

Latar Belakang dan Tujuan Proyek

Penyakit jantung adalah salah satu penyebab utama kematian secara global. Deteksi dini dan diagnosis yang akurat merupakan kunci untuk menyelamatkan nyawa dan mengurangi biaya perawatan kesehatan. Namun, proses diagnosis seringkali rumit dan bergantung pada interpretasi berbagai faktor klinis.

Machine learning (ML) menawarkan potensi besar untuk menganalisis pola kompleks dari data pasien dan memberikan prediksi risiko yang akurat.

Tujuan dari proyek ini adalah:

1. Mengembangkan dan mengevaluasi beberapa model *machine learning* untuk memprediksi kemungkinan gagal jantung pada pasien.
2. Mengidentifikasi model dengan performa terbaik berdasarkan metrik evaluasi yang rellevan.
3. Menerapkan (melakukan *deployment*) model terbaik tersebut ke dalam sebuah aplikasi web yang interaktif dan mudah digunakan oleh pemangku kepentingan, baik teknis maupun non-teknis.

Permasalahan yang Ingin Diselesaikan

Proyek ini bertujuan untuk menyelesaikan permasalahan berikut:

- **Kesulitan Deteksi Dini:** Memberikan alat bantu (tool) yang dapat mengidentifikasi individu berisiko tinggi lebih awal.
- **Aksesibilitas Diagnosis:** Menciptakan sebuah aplikasi yang dapat diakses oleh praktisi kesehatan atau bahkan pasien untuk melakukan penilaian risiko awal secara cepat.
- **Kompleksitas Data:** Menerjemahkan 11 variabel klinis yang kompleks menjadi satu keluaran (output) yang dapat ditindaklanjuti, yaitu "Risiko Rendah" atau "Risiko Tinggi".

Sumber Dataset dan Deskripsi Singkat

Dataset yang digunakan adalah **"Heart Failure Prediction Dataset"** yang bersumber dari Kaggle. Dataset ini menggabungkan lima dataset penyakit jantung poppuler lainnya dan telah dibersihkan (ternyata jenis penyakit jantung ada banyak, dalma project kali ini hanya menggabungkan beberapa saja)

Dataset ini terdiri dari 11 fitur klinis yang digunakan untuk memprediksi variabel target, HeartDisease (gagal jantung).

Dataset dan Definisi Masalah

Jumlah Fitur dan Ukuran Dataset

- **Jumlah Baris (Sampel):** 918
- **Jumlah Kolom (Fitur + Target):** 12
- **Fitur (Features):** 11 fitur yang terdiri dari data demografis dan hasil tes klinis, seperti Age, Sex, ChestPainType, RestingBP, Cholesterol, MaxHR, dan Oldpeak.
- **Target:** 1 variabel target, yaitu HeartDisease.

Variabel Target dan Tipe Task AI

- **Variabel Target:** HeartDisease. Ini adalah variabel biner di mana:
    - 1: Pasien menderita penyakit jantung.
    - 0: Pasien tidak menderita penyakit jantung.
- **Tipe Task AI: Supervised Learning** untuk **Binary Classification** (Klasifikasi Biner). Tujuannya adalah melatih model untuk memetakan 11 fitur input ke salah satu dari dua kelas target.

Langkah Pre-processing

Sebelum melatih model, data mentah harus melalui beberapa langkah *pre-processing* untuk memastikan kualitas dan kompatibilitas dengan algoritma ML:

1. **Penanganan Nilai Janggal (Anomalies):** Ditemukan bahwa 172 baris data memiliki nilai Cholesterol sebesar 0, yang secara medis tidak mungkin. Baris-baris ini dianggap sebagai data yang hilang atau tidak akurat dan dihapus dari dataset.
2. **Encoding Fitur Kategorikal:** Algoritma ML memerlukan input numerik. Oleh karena itu, fitur kategorikal (Sex, ChestPainType, RestingECG, ExerciseAngina, ST_Slope) diubah menggunakan One-Hot Encoding.
3. **Scaling Fitur Numerik:** Fitur numerik (Age, RestingBP, Cholesterol, MaxHR, Oldpeak) memiliki skala yang berbeda-beda. StandardScaler diterapkan untuk menormalisasi fitur-fitur ini, sehingga memiliki mean 0 dan standar deviasi 1.
4. **Pipeline:** Seluruh langkah *pre-processing* (Encoding dan Scaling) digabungkan ke dalam ColumnTransformer dan Pipeline untuk memastikan proses yang konsisten dan menghindari kebocoran data (*data leakage*).

Pengembangan Model

Algoritma yang Digunakan dan Alasan Pemilihan

Berdasarkan analisis performa pada *notebook* referensi, kami memilih tiga model teratas untuk dikembangkan lebih lanjut:

1. **Support Vector Machine (SVM) / SVC:**
    - **Alasan:** Sangat efektif dalam ruang berdimensi tinggi (setelah *one-hot encoding*) dan fleksibel berkat penggunaan *kernel* (misalnya, rbf) untuk menangani hubungan non-linear.
    - **Parameter:** probability=True, kernel='rbf'.
2. **Random Forest Classifier:**
    - **Alasan:** Merupakan model *ensemble* yang kuat, menggabungkan banyak *decision tree* untuk mengurangi *overfitting* dan meningkatkan akurasi. Model ini juga dapat menangani interaksi kompleks antar fitur.
    - **Parameter:** n_estimators=200, random_state=42.
3. **Extra Trees Classifier (Extremely Randomized Trees):**
    - **Alasan:** Mirip dengan Random Forest, tetapi menambahkan lebih banyak keacakan dalam pemilihan *threshold* pemisahan, yang seringkali dapat mengurangi varians model dan memberikan performa yang kompetitif.
    - **Parameter:** n_estimators=200, random_state=42.

Proses Pelatihan dan Pembagian Data

1. **Pembagian Data:** Dataset yang telah bersih dibagi menjadi dua set:
    - **Data Latih (Training Set):** 80% dari data, digunakan untuk melatih model.
    - **Data Uji (Test Set):** 20% dari data, digunakan untuk mengevaluasi performa model pada data yang belum pernah dilihat sebelumnya.
2. **Validasi Silang (Cross-Validation):** Selama fase eksperimen, *K-Fold Crross-Validation* digunakan pada data latih untuk mendapatkan estimasi performa model yang lebih stabil dan menghindari bias dari satu pembagian *train-test split* saja.
3. **Pelatihan Model Akhir:** Setelah evaluasi, ketiga model (SVM, RF, Extra Trees) dilatih kembali menggunakan **keseluruhan data latih** untuk membangun model final yang akan di-*deploy*.

Hasil Evaluasi

Metrik utama yang digunakan adalah **Akurasi**, yang mengukur proporsi prediksi yang benar (baik positif maupun negatif) dari total prediksi.

Berikut adalah perbandingan performa akurasi dari model-model yang dievaluasi pada **data uji (test set)**:

Model	Akurasi (Test Set)
Support Vector Machine (SVM)	84.74%
Random Forest	83.05%
Extra Trees	82.20%

Berdasarkan tabel di atas, **Support Vector Machine (SVM)** memberikan akurasi tertinggi, meskipun performa Random Forest dan Extra Trees sangat kompetitif dan hampir identik.

Proses Deployment

Untuk membuat model ini dapat diakses dan digunakan oleh *stakeholders* (CEO, CTO, dokter, analis), kami membangun sebuah aplikasi web interaktif.

Platform yang Digunakan

- **Framework Aplikasi: Streamlit**
    - **Alasan:** Streamlit dipilih karena kemudahannya dalam mengubah skrip data Python menjadi aplikasi. Ini memungkinkan *deployment* cepat tanpa memerlukan pengetahuan *front-end* (HTML, CSS, JavaScript) yang mendalam.
- **Platform Hosting (Opsional):** Streamlit Cloud
    - **Alasan:** Platform-platform ini menawarkan *hosting* gratis yang dapat terhubung langsung ke repositori GitHub, memungkinkan *deployment* dan pembaruan yang berkelanjutan (CI/CD) secara otomatis.

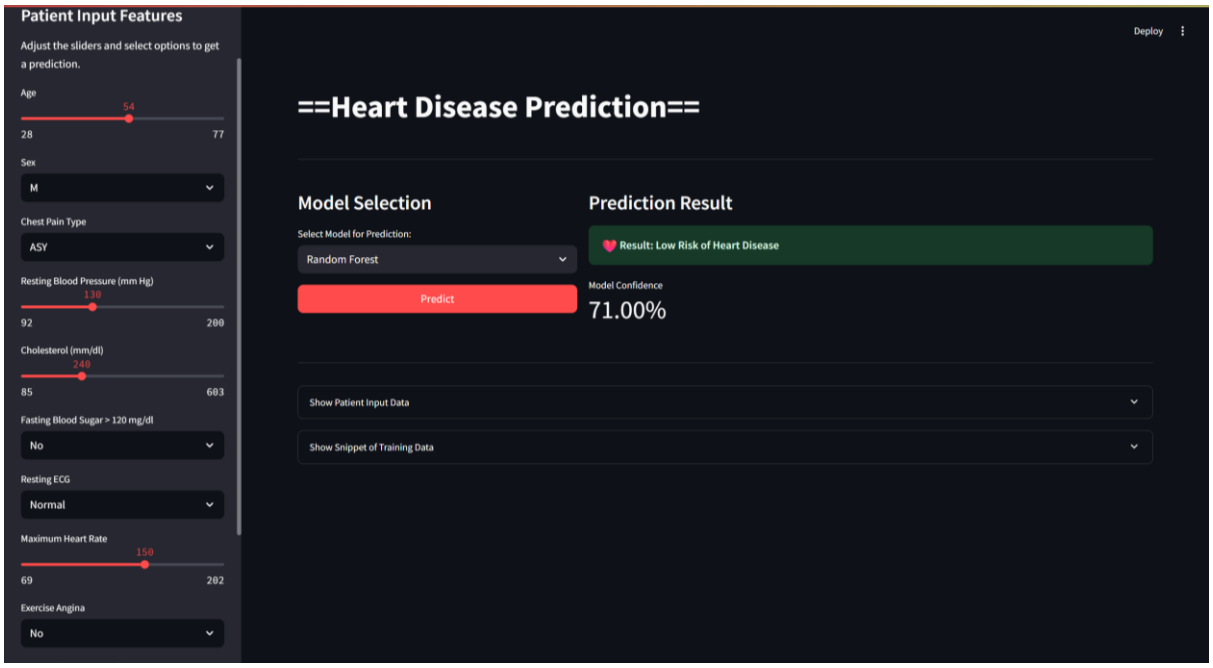
Langkah-langkah Deployment

1. **Serialisasi Pipeline:** *Pipeline* yang berisi ColumnTransformer (preprocessor) dan model yang telah dilatih (misalnya, SVM) disimpan ke dalam satu file menggunakan joblib atau pickle.Namun, dalam project kali ini, saya memutuskan untuk langusng memasukkan model dan parameternya dan membuat pipeline langsung tanpa membuat file berbentuk .pkl maupun .joblib.
   Hal ini saya lakukan dengan tujuan mempermudah deployment dan memudahkan update model (model dapat langsung belajar jika datasetnya diubah) hanya dengan menjalankan ulang server streamlit.
2. **Pembuatan Aplikasi (app.py):** Sebuah skrip Python (app.py) dibuat menggunakan *library* Streamlit.
3. **Antarmuka Pengguna (UI):**
   - Judul dan deskripsi ditambahkan menggunakan st.title() dan st.markdown().
   - *Sidebar* (st.sidebar) dibuat untuk menampung semua 11 fitur input dari pengguna.
   - st.slider digunakan untuk input numerik (misalnya, Age, RestingBP).
   - st.selectbox digunakan untuk input kategorikal (misalnya, Sex, ChestPainType).
4. **Logika Backend:**
   - Saat aplikasi dimulai, *pipeline* model yang telah disimpan dimuat ke dalam memori menggunakan st.cache_resource agar tidak perlu dimuat ulang setiap kali ada interaksi.
   - Input dari pengguna di *sidebar* dikumpulkan ke dalam sebuah DataFrame Pandas.
   - Sebuah tombol "Predict" (st.button) dibuat.
5. **Penyajian Hasil:**
   - Ketika tombol ditekan, data input pengguna dimasukkan ke dalam *pipeline* model.
   - Model menghasilkan prediksi (0 atau 1) dan probabilitasnya (predict_proba).
   - Hasil diterjemahkan ke format yang mudah dibaca (misalnya, "Risiko Tinggi" dengan st.error atau "Risiko Rendah" dengan st.success) dan ditampilkan di halaman utama.

6. **Manajemen Dependensi:** File requirements.txt dibuat untuk mendaftar semua *library* yang diperlukan (misalnya, streamlit, pandas, scikit-learn).
7. **Peluncuran (Lokal):** Aplikasi dijalankan secara lokal menggunakan perintah streamlit run app.py.
8. **Peluncuran (Publik):** Kode proyek diunggah ke repositori GitHub. Akun Streamlit Cloud ditautkan ke repositori tersebut untuk men-*deploy* aplikasi secara publik.

Screenshot Hasil Aplikasi Berjalan

Aplikasi ini menyediakan antarmuka yang bersih di mana *stakeholder* dapat memasukkan data pasien di *sidebar* sebelah kiri dan menerima hasil prediksi secara instan di halaman utama.



(Contoh: Screenshot yang menunjukkan sidebar dengan slider dan tombol 'Predict', serta hasil prediksi di sebelahnya)

Link Aplikasi

Aplikasi yang telah di-*deploy* dapat diakses melalui tautan berikut:

https://mlops----uts-bh7trnyjmycjftprrwkvxk.streamlit.app/

Kesimpulan

Kendala dan Solusi Selama Pengerjaan

1. **Kendala:** Data Cholesterol memiliki banyak nilai 0 yang tidak realistis.
    - **Solusi:** Baris-baris ini diidentifikasi dan dihapus dari dataset selama *pre-processing* untuk meningkatkan kualitas data latih.
2. **Kendala:** Fitur memiliki tipe data dan skala yang beragam (kategorikal dan numerik dengan rentang berbeda).
    - **Solusi:** Penerapan ColumnTransformer yang menggabungkan OneHotEncoder dan StandardScaler memastikan semua data diproses secara konsisten dan seragam.
3. **Kendala:** Memilih model terbaik dari banyak kandidat.
    - **Solusi:** Eksperimen yang sistematis dengan 10-Fold Cross-Validation dilakukan untuk membandingkan performa model secara objektif, yang mengarah pada pemilihan SVM, Random Forest, dan Extra Trees.

Analisis Hasil Deployment

Deployment menggunakan Streamlit berhasil mentransformasi model *machine learning* yang kompleks menjadi alat bantu keputusan yang praktis.

- **Untuk Stakeholder Non-Teknis (CEO, Manajer):** Aplikasi ini memberikan gambaran langsung tentang nilai bisnis dari proyek ML, mengubah data mentah menjadi wawasan yang dapat ditindaklanjuti.
- **Untuk Stakeholder Teknis (CTO, Tim Medis):** Aplikasi ini berfungsi sebagai *prototype* (MVP - Minimum Viable Product) yang valid dan dapat digunakan untuk pengujian lebih lanjut, atau bahkan diintegrasikan ke dalam sistem rekam medis elektronik (EHR) yang ada.

Rencana Perbaikan di Masa Depan

1. **Peningkatan Akurasi:** Mengeksplorasi teknik *feature engineering* yang lebih canggih dan melakukan *hyperparameter tuning* yang lebih ekstensif (misalnya, menggunakan GridSearchCV).
2. **Interpretasi Model (Explainable AI):** Mengintegrasikan *library* seperti **SHAP** atau **LIME** ke dalam aplikasi Streamlit untuk menunjukkan fitur apa yang paling berkontribusi terhadap prediksi (misalnya, "Risiko tinggi karena Oldpeak tinggi dan ChestPainType ASY").
3. **Memperkuat infrastruktur:** Jika digunakan dalam skala besar, aplikasi dapat dipindahkan dari Streamlit Cloud ke infrastruktur yang lebih *scalable* seperti AWS atau GCP menggunakan Docker dan Kubernetes.

Lampiran

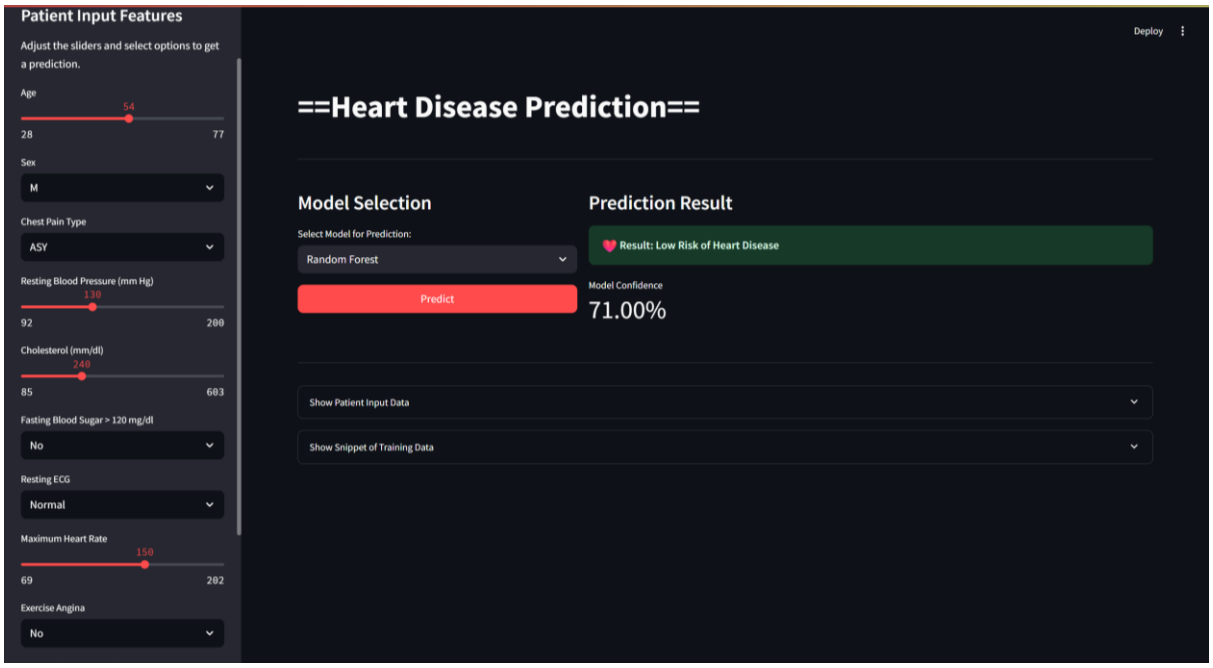
Link Deployment

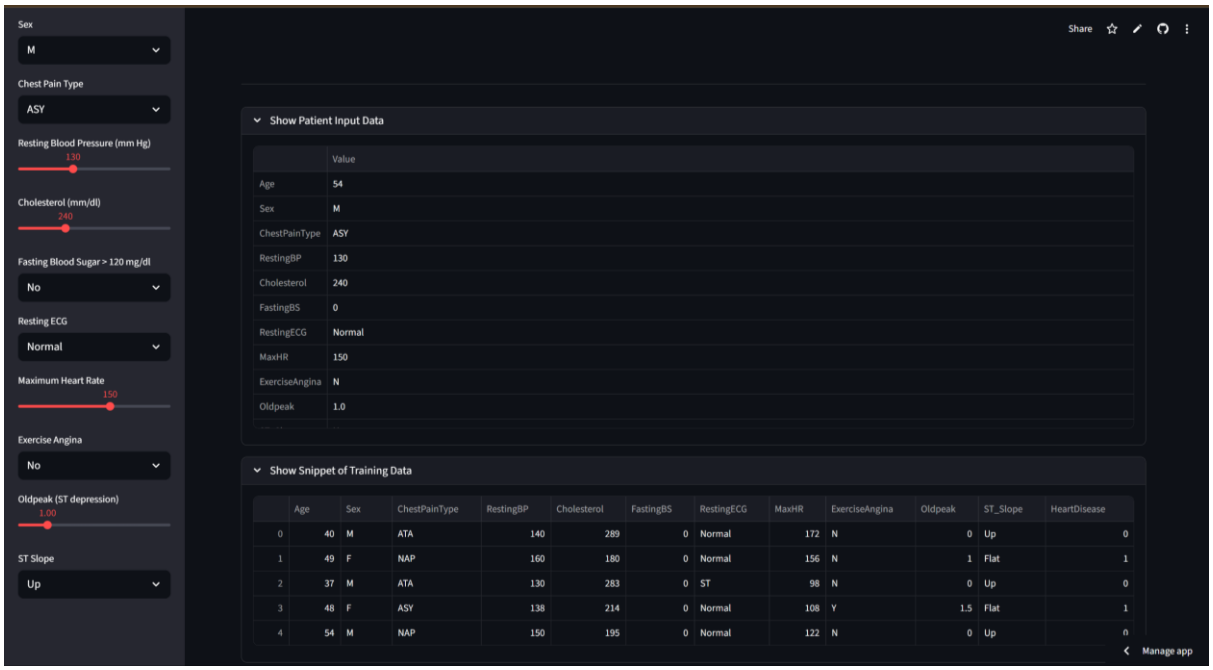
- **URL Aplikasi:** https://mlops----uts-bh7trnyjmycjftprrwkvxk.streamlit.app/

Link GitHub Repository

- **URL Kode Sumber:** https://github.com/happyvictor-Vesper/MLOPS----UTS

Cuplikan Tampilan/Screenshot





Video Demonstrasi Aplikasi

https://youtu.be/mt8UkPotAZI