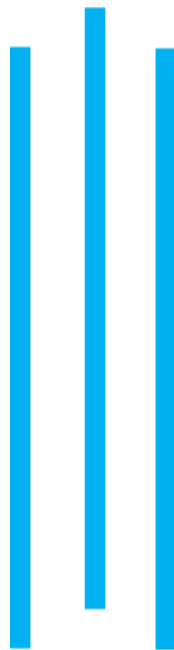


**LAPORAN**  
**UJIAN AKHIR SEMESTER**  
**MATA KULIAH DATA SCIENCE**



Disusun Oleh :

Arya Yudha Prasetya

NIM 234311007

**JURUSAN TEKNIK**  
**PROGRAM STUDI REKAYASA PERANGKAT LUNAK**  
**POLITEKNIK NEGERI MADIUN**  
**2025**

# INFORMASI PROYEK

**Judul Proyek :** “Analisis Tingkat Risiko Kesehatan Maternal Menggunakan Pendekatan Machine Learning dan Deep Learning”

**Nama Mahasiswa** : Arya Yudha Prasetya  
**NIM** : 234311007  
**Program Studi** : Teknologi Rekayasa Perangkat Lunak  
**Mata Kuliah** : Data Science  
**Dosen Pengampu** : Gus Nanang Syaifuddiin, S.Kom., M.Kom.  
**Tahun Akademik** : 2025 / 5  
**Link Github** : [https://github.com/AryaYudhaP/UAS\\_DataScience](https://github.com/AryaYudhaP/UAS_DataScience)  
**Link Video Pembahasan** : <https://youtu.be/BopHMoFHPQ4>

## 1. LEARNING OUTCOMES

Pada proyek ini, mahasiswa diharapkan dapat:

1. Memahami konteks masalah dan merumuskan problem statement secara jelas
2. Melakukan analisis dan eksplorasi data (EDA) secara komprehensif (OPSIONAL)
3. Melakukan data preparation yang sesuai dengan karakteristik dataset
4. Mengembangkan tiga model machine learning yang terdiri dari (WAJIB)
  - Model baseline
  - Model machine learning / advanced
  - Model deep learning (WAJIB)
5. Menggunakan metrik evaluasi yang relevan dengan jenis tugas ML
6. Melaporkan hasil eksperimen secara ilmiah dan sistematis
7. Mengunggah seluruh kode proyek ke GitHub (WAJIB)
8. Menerapkan prinsip software engineering dalam pengembangan proyek

## 2. PROJECT OVERVIEW

### 2.1 Latar Belakang

Kesehatan maternal (ibu hamil) merupakan salah satu indikator utama kesejahteraan kesehatan global. Menurut Organisasi Kesehatan Dunia (WHO), komplikasi selama kehamilan dan persalinan masih menjadi penyebab utama kematian ibu, terutama di daerah berkembang. Banyak dari komplikasi ini sebenarnya dapat dicegah atau ditangani jika faktor risikonya terdeteksi sejak dini. Faktor fisiologis seperti usia, tekanan darah (*Systolic* dan *Diastolic*), serta kadar gula darah (*Blood Sugar*) diketahui memiliki korelasi kuat dengan tingkat risiko kehamilan, seperti kemungkinan terjadinya preeklampsia atau diabetes gestasional.

Pemanfaatan teknologi *Data Science* dan *Machine Learning* menawarkan solusi untuk membantu tenaga medis melakukan skrining awal secara otomatis. Dengan menganalisis data historis pasien, model cerdas dapat mempelajari pola-pola kompleks dari tanda-tanda vital pasien untuk memprediksi apakah seorang ibu hamil berada dalam kategori risiko rendah (*low risk*), menengah (*mid risk*), atau tinggi (*high risk*).

Proyek ini bertujuan untuk membangun dan membandingkan performa model klasifikasi otomatis menggunakan pendekatan *Machine Learning* tradisional dan *Deep Learning*. Dataset yang digunakan bersumber dari *Maternal Health Risk Data Set* (UCI Machine Learning Repository) yang dikumpulkan melalui sistem pemantauan IoT. Penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan sistem pendukung keputusan klinis yang cepat dan akurat.

Referensi Studi Literatur :

Ahmed, M., Kashem, M. A., Rahman, M., & Khatun, S. (2020). Review and Analysis of Risk Factor of Maternal Health in Remote Area Using the Internet of Things (IoT). Dalam A. N. Kasruddin Nasir, M. A. Ahmad, M. S. Najib, Y. Abdul Wahab, N. A. Othman, N. M. Abd Ghani, A. Irawan, S. Khatun, R. M. T. Raja Ismail, M. M. Saari, M. R. Daud, & A. A. Mohd Faudzi (Ed.), *InECCE2019* (Vol. 632, hlm. 357–365). Springer Singapore.  
[https://doi.org/10.1007/978-981-15-2317-5\\_30](https://doi.org/10.1007/978-981-15-2317-5_30)

Togunwa, T. O., Babatunde, A. O., & Abdullah, K.-R. (2023). Deep hybrid model for maternal health risk classification in pregnancy: Synergy of ANN and random forest. *Frontiers in Artificial Intelligence*, 6, 1213436.  
<https://doi.org/10.3389/frai.2023.1213436>

### **3. BUSINESS UNDERSTANDING / PROBLEM UNDERSTANDING**

#### **3.1 Problem Statements**

- 1) Keterbatasan Deteksi Dini Manual  
Proses identifikasi tingkat risiko kesehatan maternal seringkali bergantung pada penilaian subjektif tenaga medis yang memakan waktu, sehingga diperlukan metode otomatisasi yang objektif berdasarkan data tanda-tanda vital.
- 2) Kompleksitas Pola Data  
Hubungan antara parameter kesehatan (seperti tekanan darah sistolik, diastolik, dan kadar gula darah) seringkali tidak linear, menyulitkan metode statistik sederhana untuk mengklasifikasikan risiko secara akurat.
- 3) Kebutuhan Evaluasi Model  
Diperlukan studi pembandingan untuk menentukan apakah pendekatan *Deep Learning* yang lebih detail memberikan peningkatan performa yang signifikan

dibandingkan algoritma *Machine Learning* klasik pada dataset tabular berukuran kecil-menengah.

### 3.2 Goals

- 1) Membangun model prediksi klasifikasi yang mampu mengidentifikasi tiga tingkat risiko kesehatan maternal (*Low, Mid, High Risk*) dengan tingkat akurasi di atas 70%.
- 2) Menganalisis fitur-fitur kesehatan yang paling berpengaruh (*feature importance*) dalam penentuan risiko kehamilan.
- 3) Mengevaluasi dan membandingkan performa tiga pendekatan model berbeda (Baseline, Advanced, dan Deep Learning) berdasarkan metrik Akurasi, Precision, Recall, dan F1-Score.

### 3.3 Solution Approach

#### Model 1 – Baseline Model: Logistic Regression

Sebagai model pembanding dasar, digunakan **Logistic Regression**.

- **Alasan Pemilihan:** Algoritma ini merupakan standar industri untuk masalah klasifikasi karena komputasinya yang sangat cepat, mudah diinterpretasikan, dan memberikan probabilitas output yang jelas. Model ini akan menjadi tolak ukur (*benchmark*) untuk melihat apakah model yang lebih kompleks benar-benar diperlukan.

#### Model 2 – Advanced / ML Model: Random Forest Classifier

Sebagai representasi model *Machine Learning*, digunakan **Random Forest**.

- **Alasan Pemilihan:** Random Forest adalah metode *ensemble* (kumpulan *Decision Trees*) yang sangat tangguh terhadap *overfitting* dan mampu menangani hubungan non-linear antar fitur dengan baik. Model ini umumnya memberikan performa sangat tinggi pada data tabular tanpa memerlukan penyetelan hyperparameter yang terlalu rumit.

#### Model 3 – Deep Learning Model :Multilayer Perceptron (MLP)

Sebagai representasi model *Deep Learning*, digunakan arsitektur **Multilayer Perceptron (MLP)** menggunakan framework TensorFlow/Keras.

- **Alasan Pemilihan:** Sesuai dengan karakteristik data yang bersifat tabular (numerik), arsitektur MLP dengan *Dense Layers* adalah pilihan yang paling tepat.
- **Spesifikasi:** Model ini akan dibangun menggunakan minimal 2 *hidden layers* dengan fungsi aktivasi *ReLU* dan lapisan *Dropout* untuk regularisasi. Model akan dilatih selama 50 epoch untuk mempelajari representasi fitur yang kompleks yang mungkin terlewatkan oleh model ML tradisional.

## 4. DATA UNDERSTANDING

### 4.1 Informasi Dataset

### Sumber Dataset :

Dataset yang digunakan dalam penelitian ini adalah **Maternal Health Risk Data Set** yang diperoleh dari UCI Machine Learning Repository. Data ini dikumpulkan melalui sistem pemantauan kesehatan berbasis IoT di daerah pedesaan Bangladesh.

### Deskripsi Dataset :

- Sumber: UCI Machine Learning Repository (Ahmed et al., 2020)
- Jumlah Baris (Awal): 1.014 data
- Jumlah Kolom (Fitur): 7 kolom (6 Fitur Numerik + 1 Target Kategorikal)
- Tipe Data: Data Tabular (Multivariate)
- Format File: CSV

### 4.2 Deskripsi Fitur

Nama Fitur	Tipe Data	Deskripsi	Satuan / Contoh Nilai
Age	Integer	Usia wanita hamil	Tahun (misal: 25, 35)
SystolicBP	Integer	Tekanan darah sistolik (atas)	mmHg (misal: 120, 140)
DiastolicBP	Integer	Tekanan darah diastolik (bawah)	mmHg (misal: 80, 90)
BS	Float	Kadar gula darah ( <i>Blood Glucose</i> )	mmol/L (misal: 6.1, 15.0)
BodyTemp	Float	Suhu tubuh pasien	Fahrenheit (misal: 98.0, 101.0)
HeartRate	Integer	Detak jantung per menit	bpm (misal: 70, 76)
RiskLevel	Categorical	Tingkat risiko kehamilan (Target)	high risk, mid risk, low risk

### 4.3 Kondisi Data

- **Missing Value** : Tidak ada
- **Duplicate Data** : Ada, 562 baris
- **Outliers** : Ada, Terutama pada fitur BS (Gula Darah) dan BodyTemp

#### Penjelasan :

- a) BS (Blood Sugar): Ditemukan sekitar 210 data *outlier* statistik (nilai > 9.65 mmol/L).
- b) BodyTemp: Ditemukan 210 data *outlier* (suhu > 98°F). Mayoritas data bernilai 98°F, sehingga suhu demam (101-103°F) dianggap outlier secara statistik

- **Imbalanced Data** : Ada

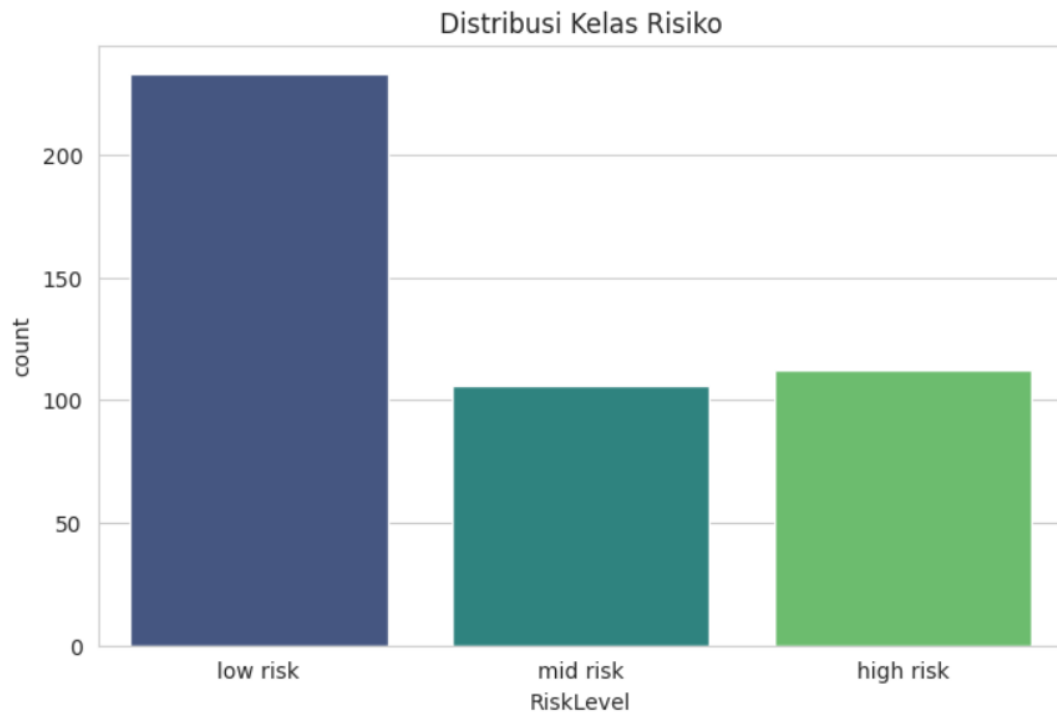
#### Penjelasan :

- 1) Low Risk: 40.0% (406 data)
- 2) Mid Risk: 33.1% (336 data)
- 3) High Risk: 26.8% (272 data)

- **Noise** : Ada. Ditemukan nilai yang tidak masuk akal secara medis pada fitur **HeartRate** dengan nilai minimum **7 bpm**. Kondisi ini kemungkinan besar adalah *noise* atau kesalahan input (typo), karena detak jantung manusia normal tidak mungkin serendah itu.
- **Data Quality Issues** : Tidak ada

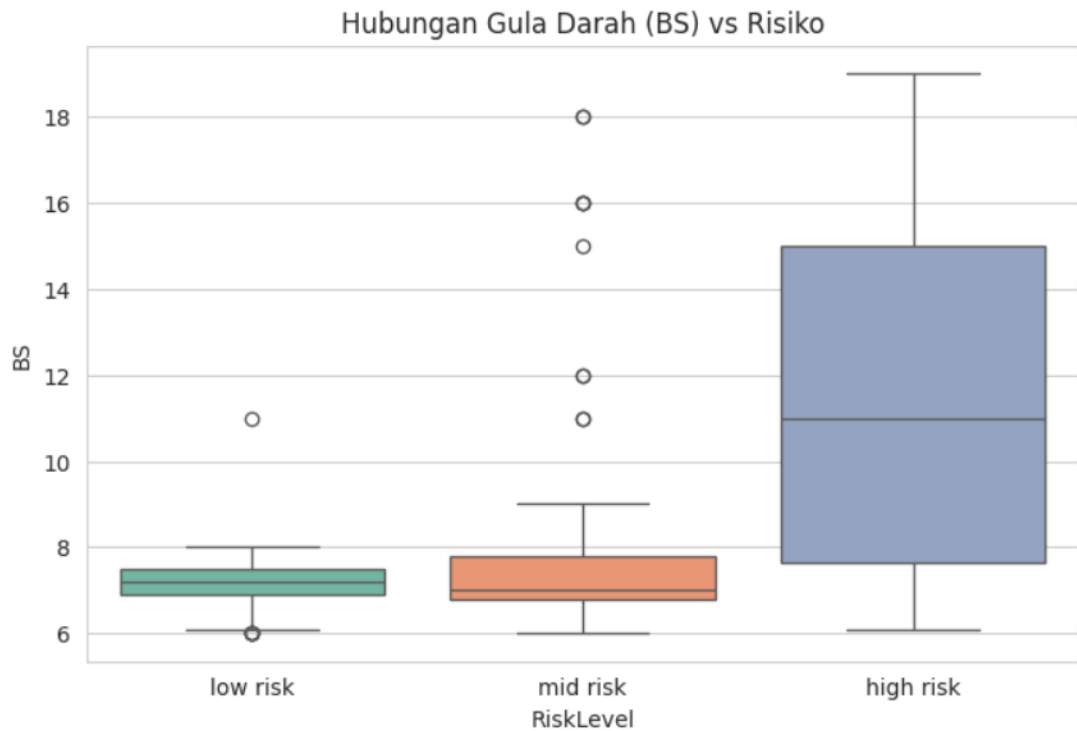
#### 4.4 Exploratory Data Analysis (EDA)

**Visualisasi 1** : Bar Chart Distribusi Kelas Risiko



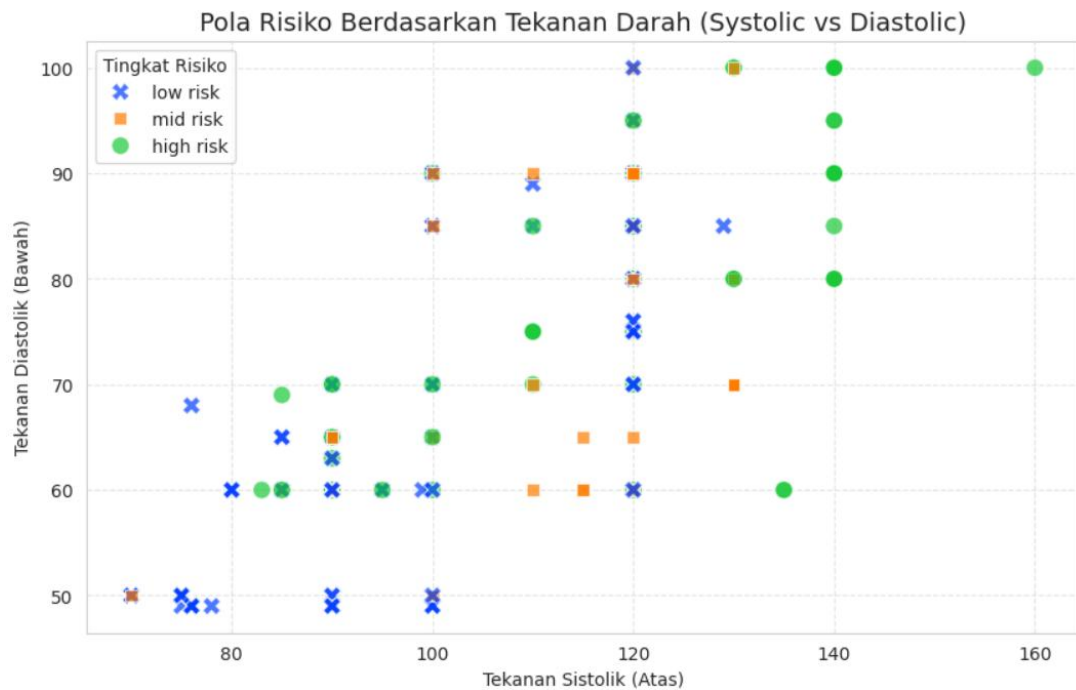
- **Deskripsi**: Visualisasi ini menunjukkan jumlah pasien untuk setiap kategori risiko dalam dataset setelah proses pembersihan (*cleaning*).
- **Analisis/Insight**: Berdasarkan grafik di atas, terlihat bahwa distribusi kelas target tidak sepenuhnya seimbang (imbalanced). Kategori Low Risk memiliki jumlah sampel terbanyak yaitu 406 data, diikuti oleh Mid Risk sebanyak 336 data, dan High Risk sebanyak 272 data. Meskipun terdapat ketimpangan, rasio antar kelas masih dalam batas wajar (sekitar 40:33:27), sehingga model masih dapat mempelajari pola kelas minoritas (High Risk) tanpa memerlukan teknik oversampling yang agresif.

**Visualisasi 2** : Boxplot: Hubungan Gula Darah (BS) vs Risiko



- **Deskripsi:** Grafik ini membandingkan distribusi statistik (median, kuartil) dari kadar gula darah (*Blood Sugar*) pada masing-masing kategori risiko.
- **Analisis/Insight:**  
Visualisasi Boxplot menunjukkan perbedaan pola yang sangat signifikan. Pasien dengan status High Risk memiliki rata-rata kadar gula darah yang jauh lebih tinggi, mayoritas di atas 12.0 mmol/L, dibandingkan kategori Low dan Mid Risk yang terpusat di angka 6.0 - 8.0 mmol/L. Hal ini mengindikasikan bahwa fitur BS (*Blood Sugar*) merupakan fitur prediktor (pembeda) yang paling kuat dalam dataset ini. Kenaikan gula darah berbanding lurus dengan peningkatan risiko kesehatan maternal.

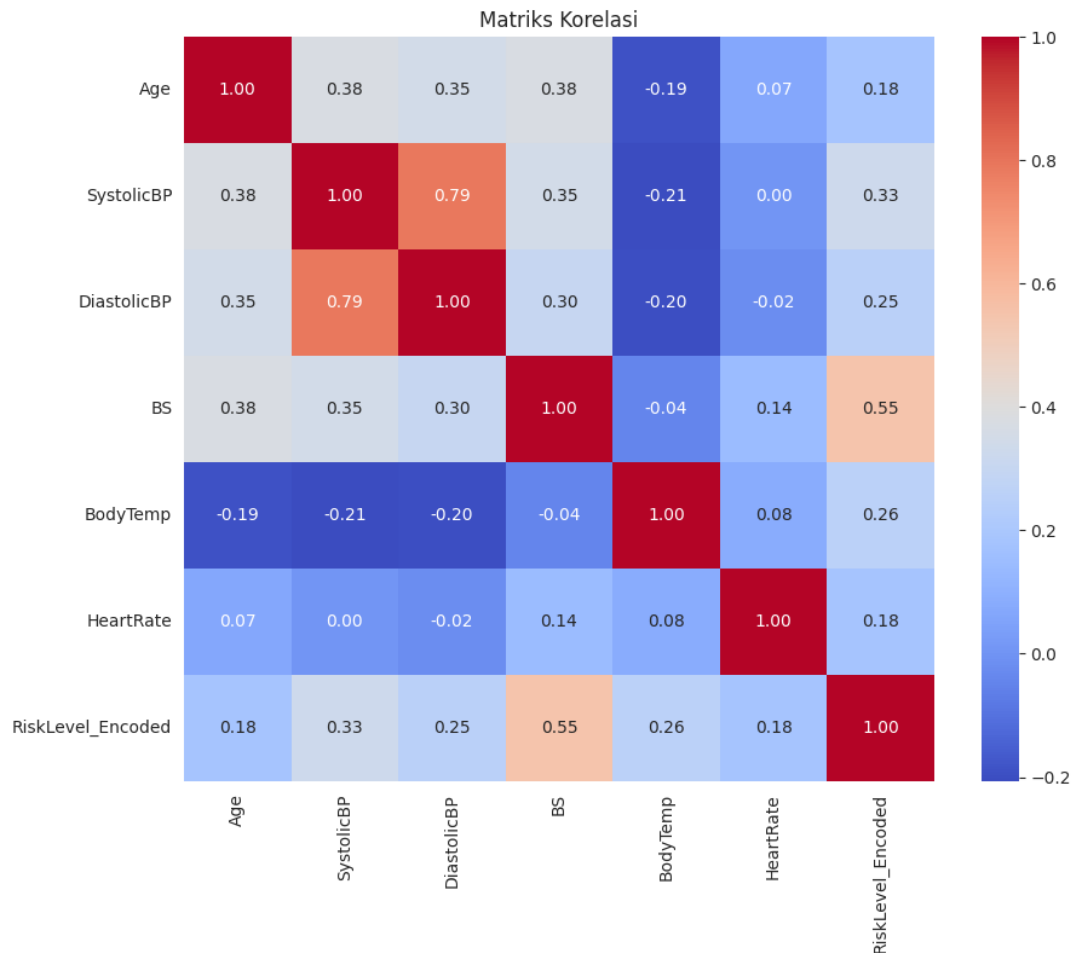
### Visualisasi 3 : Scatter Plot: Pola Tekanan Darah (Systolic vs Diastolic)



- **Deskripsi:** Grafik sebar ini memetakan setiap pasien berdasarkan dua sumbu: Tekanan Sistolik (Sumbu X) dan Tekanan Diastolik (Sumbu Y), dengan warna titik yang membedakan tingkat risiko.
- **Analisis/Insight:**  
Dapat diamati adanya pola pengelompokan (clustering) yang jelas. Titik-titik berwarna Biru (Low Risk) cenderung berkumpul di area kiri-bawah, yang merepresentasikan tekanan darah normal (Sistolik < 120, Diastolik < 80). Sebaliknya, titik-titik Hijau (High Risk) menyebar ke arah kanan-atas, mengindikasikan bahwa pasien dengan hipertensi (tekanan darah tinggi) memiliki kecenderungan besar untuk masuk ke kategori risiko tinggi. Namun, terdapat juga beberapa titik High Risk di area tekanan darah normal, yang kemungkinan dipengaruhi oleh faktor lain seperti gula darah.

#### **Visualisasi 4 : Heatmap: Matriks Korelasi**





- **Deskripsi:** Matriks ini menampilkan nilai koefisien korelasi (Pearson Correlation) antar setiap pasang variabel, di mana warna merah pekat menunjukkan korelasi positif yang kuat.
- **Analisis/Insight:**  
 Dari hasil Heatmap, ditemukan dua temuan statistik penting:
  - a) Terdapat korelasi positif yang sangat kuat (**0.78**) antara *SystolicBP* dan *DiastolicBP*.
  - b) Korelasi Target: Jika dilihat pada baris/kolom *RiskLevel\_Encoded*, fitur **BS** (Gula Darah) memiliki nilai korelasi positif tertinggi terhadap target, disusul oleh tekanan darah dan usia. Ini mengonfirmasi secara statistik bahwa gula darah adalah faktor determinan utama dalam prediksi risiko."

## 5. DATA PREPARATION

### 5.1 Data Cleansing

Pembersihan data dilakukan untuk meningkatkan integritas model dengan menghapus data yang redundan dan tidak valid.

1. **Penanganan Duplikat:** Ditemukan sebanyak 562 baris data yang memiliki nilai identik pada seluruh fiturnya. Data duplikat ini dihapus agar model tidak *overfitting* pada pola duplikat

2. Penanganan Noise : Berdasarkan analisis domain medis, ditemukan data dengan nilai Detak Jantung (*HeartRate*) di bawah 40 bpm (misal: 7 bpm). Nilai ini dianggap sebagai kesalahan sensor atau input (*noise*) karena tidak memungkinkan untuk kondisi hidup normal. Data tersebut telah dihapus dari dataset.

## 5.2 Feature Engineering

Feature Engineering dilakukan untuk menyesuaikan format data agar dapat diproses secara optimal oleh algoritma komputasi.

1. Label Encoding pada Target Variable: Variabel target RiskLevel aslinya bertipe kategorikal (*string*). Karena model *Neural Network* dan algoritma *Scikit-Learn* memerlukan input numerik, dilakukan konversi data dengan pemetaan (*mapping*) manual sebagai berikut:
  - low risk dikonversi menjadi 0
  - mid risk dikonversi menjadi 1
  - high risk dikonversi menjadi 2
2. Feature Selection (Redundancy Removal): Setelah proses encoding, kolom asli RiskLevel yang bertipe teks dihapus dari dataset fitur (*input X*) untuk mencegah duplikasi informasi. Model hanya akan menerima fitur numerik hasil transformasi sebagai label target (*output y*).

## 5.3 Data Transformation

Transformasi data dilakukan untuk menyamakan skala antar fitur (*Scaling*).

- Metode: Standard Scaler (Standardization).
- Alasan: Fitur dalam dataset memiliki jumlah yang sangat bervariasi (contoh: BS bernilai satuan, sedangkan SystolicBP bernilai ratusan). Tanpa scaling, model seperti Deep Learning (MLP) dan SVM akan kesulitan karena bobot akan bias ke fitur bernilai besar.
- Proses: Scaler dilatih (fit) hanya pada Training Set lalu diterapkan (transform) ke Training dan Test Set untuk mencegah kebocoran informasi (data leakage).

## 5.4 Data Splitting

Dataset dibagi menjadi dua bagian terpisah untuk proses pelatihan dan validasi.

- Rasio Pembagian: 80% Training Data : 20% Testing Data.
- Teknik: Menggunakan parameter `random_state=42`

## 5.5 Data Balancing

Karena pada data terdapat ketidakseimbangan distribusi kelas, maka dilakukan penyeimbangan sebagai berikut :

- Stratified Splitting: Saat pembagian data (*splitting*), digunakan parameter `stratify=y`. Ini memaksa rasio kelas (Low:Mid:High) di data latih dan data uji tetap sama persis dengan populasi aslinya.

- **Class Weight Adjustment:** Pada model *Logistic Regression* dan *Random Forest*, ditambahkan parameter `class_weight='balanced'`. Ini memberikan "penalti" lebih besar jika model salah memprediksi kelas minoritas (*High Risk*), sehingga model terdorong untuk lebih memperhatikan kelas yang jumlahnya sedikit.

## 5.6 Ringkasan Data Preparation

Tahapan	Keterangan	Jumlah Data (Baris)
Awal	Data mentah dari sumber	1.014
Cleaning	Setelah hapus 562 Duplikat & Noise	451
Splitting	Training Set (80%)	360
	Testing Set (20%)	91

## 6. MODELING

### 6.1 Model 1 : Baseline Model

#### 6.1.1 Deskripsi Model

- **Nama Model:** Logistic Regression
- **Teori Singkat:** Logistic Regression adalah metode statistik yang digunakan untuk memprediksi probabilitas kejadian suatu kelas biner atau multi-kelas. Model ini menggunakan fungsi logistik (sigmoid/softmax) untuk memetakan input fitur ke dalam rentang probabilitas 0 hingga 1.
- **Alasan Pemilihan:** Dipilih sebagai *baseline* (tolok ukur) karena sifatnya yang sederhana, mudah diinterpretasikan, dan komputasinya sangat cepat. Jika model Deep Learning nanti tidak bisa mengalahkan akurasi model ini secara signifikan, maka penggunaan model kompleks tidak terjustifikasi.

#### 6.1.2 Hyperparameter

- `max_iter:` 1000 (Ditingkatkan agar model mencapai konvergensi sempurna)
- `class_weight:` 'balanced' (Memberikan bobot lebih pada kelas minoritas/High Risk untuk mengatasi imbalanced data).
- `random_state:` 42 (Untuk konsistensi hasil).

#### 6.1.3 Implementasi

```

1 #@title Model 1 - Logistic Regression
2
3 from sklearn.linear_model import LogisticRegression
4
5 # 1. Buat Model
6 model_logreg = LogisticRegression(random_state=42, class_weight='balanced', max_iter=1000)
7
8 # 2. Latih Model
9 model_logreg.fit(X_train_scaled, y_train)
10
11 # 3. Prediksi
12 y_pred_logreg = model_logreg.predict(X_test_scaled)
13 acc_logreg = accuracy_score(y_test, y_pred_logreg)
14
15 print(f"Akurasi Logistic Regression: {acc_logreg:.4f}")
16 print("\nClassification Report:")
17 print(classification_report(y_test, y_pred_logreg))

```

#### 6.1.4 Hasil Awal

Berdasarkan evaluasi pada data testing, model ini mencapai akurasi sebesar 0.6044

## 6.2 Model 2: ML / Advanced Model

### 6.2.1 Deskripsi Model

- **Nama Model:** Random Forest Classifier
- **Teori Singkat:** Random Forest adalah algoritma *ensemble learning* yang bekerja dengan membangun banyak *Decision Trees* (pohon keputusan) pada saat pelatihan. Hasil prediksi akhir ditentukan berdasarkan *voting* mayoritas dari seluruh pohon.
- **Alasan Pemilihan:** Algoritma ini dipilih karena kemampuannya menangani data tabular dengan hubungan non-linear yang kompleks dan fitur yang berkorelasi.
- **Keunggulan:** Sangat resisten terhadap *overfitting* (karena prinsip *Bagging*), tidak memerlukan penskalaan data yang ketat (meski kita tetap melakukannya), dan memberikan fitur *importance*.
- **Kelemahan:** Model cenderung lambat saat prediksi *real-time* jika jumlah pohon terlalu banyak, dan sulit diinterpretasi secara visual dibandingkan satu pohon keputusan tunggal.

### 6.2.2 Hyperparameter

- `n_estimators`: 100 (Jumlah pohon keputusan yang dibangun).
- `class_weight`: 'balanced' (Penyesuaian bobot otomatis berdasarkan frekuensi kelas).
- `random_state`: 42.

### 6.2.3 Implementasi

```

2 model_rf = RandomForestClassifier(n_estimators=100, random_state=42, class_weight='balanced'
3 )
4 # 2. Latih Model
5 model_rf.fit(X_train_scaled, y_train)
6 # 3. Prediksi
7 y_pred_rf = model_rf.predict(X_test_scaled)
8 acc_rf = accuracy_score(y_test, y_pred_rf)
9
10 print(f"Akurasi Random Forest: {acc_rf:.4f}\n")
11 print("Classification Report :")
12 print(classification_report(y_test, y_pred_rf))
13
14 # 4. Visualisasi Feature Importance
15 importances = model_rf.feature_importances_
16 feature_names = X.columns
17 df_imp = pd.DataFrame({'Feature': feature_names, 'Importance': importances})
18 df_imp = df_imp.sort_values('Importance', ascending=False)
19
20 plt.figure(figsize=(8, 4))
21 sns.barplot(x='Importance', y='Feature', data=df_imp, palette='viridis')
22 plt.title('Feature Importance')
23 plt.show()

```

#### 6.2.4 Hasil Model

Model Random Forest menghasilkan akurasi sebesar 0.6484

### 6.3 Model 3: Deep Learning - Multilayer Perceptron (MLP)

#### 6.3.1 Deskripsi Model

- **Nama Model:** Multilayer Perceptron (MLP)
- **Alasan Pemilihan:** MLP adalah arsitektur *Neural Network* yang paling sesuai untuk data terstruktur/tabular. Kemampuan MLP dalam mempelajari representasi fitur hierarkis melalui *hidden layers* diharapkan mampu menangkap pola risiko kesehatan yang tidak bisa dideteksi oleh model linier.

#### 6.3.2 Arsitektur Model

No	Layer (Type)	Output Shape	Param #	Activation	Deskripsi Fungsi
1	Input Layer	(None, 6)	0	-	Menerima 6 fitur input (Age, SystolicBP, dll) yang telah distandarisasi.
2	Dense (Hidden 1)	(None, 64)	448	ReLU	Lapisan tersembunyi pertama untuk mengekstraksi pola fitur dasar.
3	Dropout	(None, 64)	0	-	Mematikan 20% neuron secara acak untuk mencegah <i>overfitting</i> .
4	Dense (Hidden 2)	(None, 32)	2,080	ReLU	Lapisan tersembunyi kedua untuk mempelajari kombinasi fitur yang lebih kompleks.

5	<b>Dropout</b>	(None, 32)	0	-	Mekanisme regularisasi tambahan sebelum lapisan output.
6	<b>Dense (Output)</b>	(None, 3)	99	Softmax	Menghasilkan probabilitas untuk 3 kelas risiko ( <i>Low, Mid, High</i> ).

### 6.3.3 Input & Preprocessing Khusus

- **Input shape:** (Batch\_Size, 6), Sesuai dengan 6 fitur vital (Age, SystolicBP, DiastolicBP, BS, BodyTemp, HeartRate).
- Preprocessing Khusus:
  - 1) **Standard Scaler (Z-Score Normalization):** Wajib dilakukan agar gradient descent dapat konvergen dengan stabil.
  - 2) **Label Encoding:** Target diubah menjadi 0, 1, 2.

### 6.3.4 Hyperparameter

Training Configuration:

- **Optimizer:** Adam (Default learning rate = 0.001)
- **Loss function:** sparse\_categorical\_crossentropy (Efisien untuk klasifikasi multi-kelas dengan label integer)
- **Metrics:** accuracy
- **Batch size:** 32
- **Epochs:** 50
- **Validation split:** 0.2 (20% data training diambil otomatis untuk validasi per epoch)

### 6.3.5 Implementasi

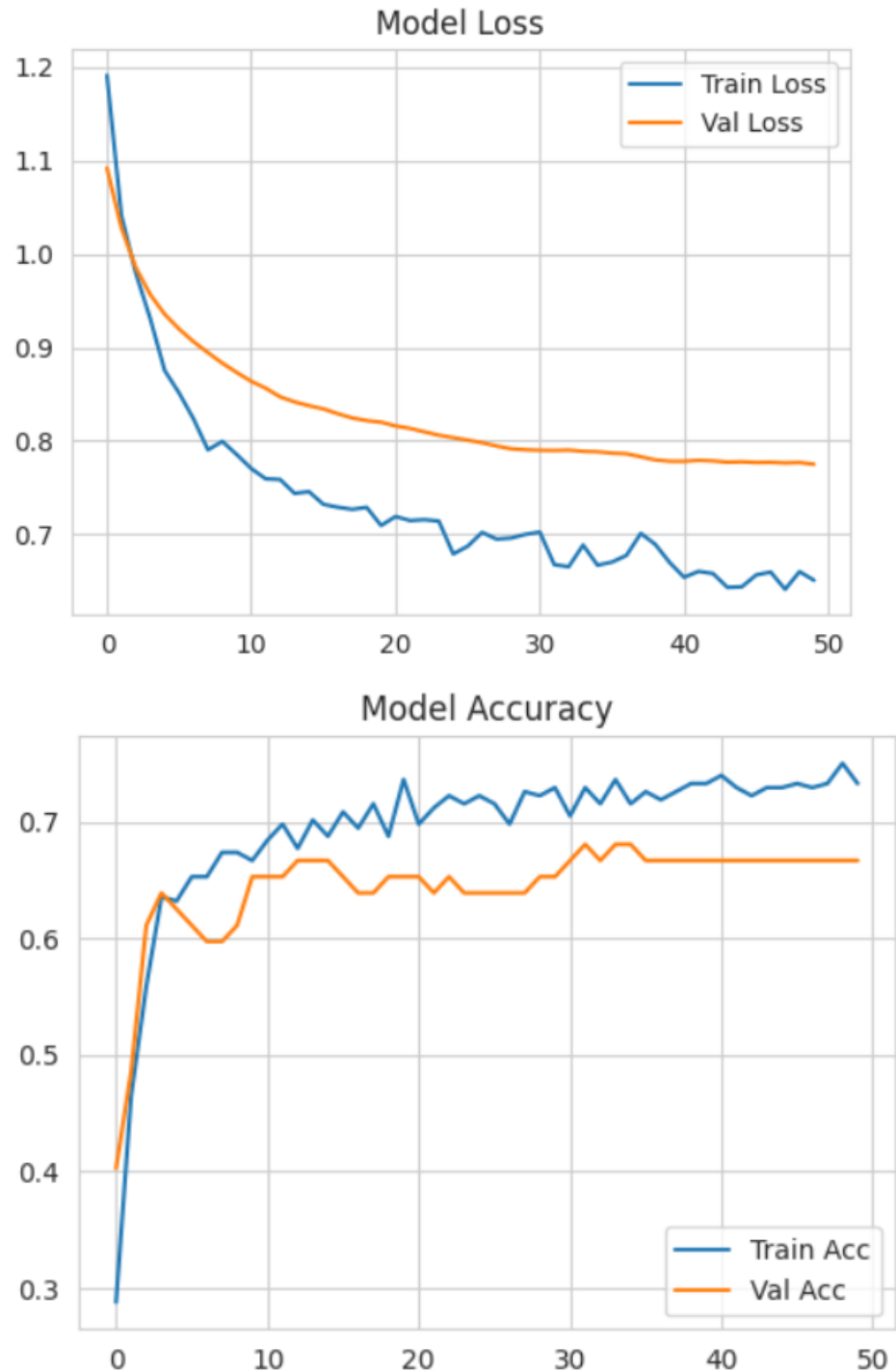
```

2 tf.random.set_seed(42)
3 # 1. Definisi Arsitektur Model (MLP)
4 model_dl = Sequential([
5     Dense(64, activation='relu', input_shape=(X_train_scaled.shape[1],)),
6     Dropout(0.2),
7     Dense(32, activation='relu'),
8     Dropout(0.2),
9     Dense(3, activation='softmax')
10 ])
11 # 2. Compile Model
12 model_dl.compile(optimizer='adam',
13                 loss='sparse_categorical_crossentropy',
14                 metrics=['accuracy'])
15 # 3. Training Model
16 print("Mulai Training Deep Learning...")
17 history = model_dl.fit(
18     X_train_scaled, y_train,
19     epochs=50,
20     batch_size=32,
21     validation_split=0.2,
22     verbose=0
23 )
24 print("Training Selesai!")
25 # 4. Plot Grafik Loss & Accuracy
26 plt.figure(figsize=(12, 4))
27 plt.subplot(1, 2, 1)
28 plt.plot(history.history['loss'], label='Train Loss')
29 plt.plot(history.history['val_loss'], label='Val Loss')
30 plt.title('Model Loss')
31 plt.legend()
32 plt.subplot(1, 2, 2)
33 plt.plot(history.history['accuracy'], label='Train Acc')
34 plt.plot(history.history['val_accuracy'], label='Val Acc')
35 plt.title('Model Accuracy')
36 plt.legend()
37 plt.show()
38 # 5. Prediksi Data Testing
39 y_pred_probs = model_dl.predict(X_test_scaled)
40 y_pred_dl = np.argmax(y_pred_probs, axis=1)
41 # 6. Tampilkan Angka Metrik
42 print(f"Akurasi Deep learning MLP: {accuracy_score(y_test, y_pred_dl):.4f}\n")
43 print("Classification Report:")

```

### 6.3.6 Training Process

- **Training Time:** Kurang dari 1 menit (Komputasi ringan pada data tabular).
- **Computational Resource:** Google Colab (CPU) / Local Environment.
- Training History Visualization:



- Analisis Training :

- 1) **Overfitting:** Berdasarkan grafik *Loss*, garis *Validation Loss* cenderung bergerak beriringan dengan *Train Loss*, yang menandakan *Dropout* bekerja baik dalam mencegah overfitting.
- 2) **Convergence:** Model mulai stabil (*converge*) di sekitar epoch ke-20 hingga 30, ditandai dengan akurasi yang tidak lagi naik secara drastis.
- 3) **Epochs:** Jumlah 50 epoch sudah cukup. Menambah epoch lebih banyak berisiko meningkatkan *validation loss* (overfitting).



### 6.3.7 Model Summary

```
2 model_dl.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	448
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 32)	2,080
dropout_1 (Dropout)	(None, 32)	0
dense_2 (Dense)	(None, 3)	99

Total params: 7,883 (30.80 KB)  
Trainable params: 2,627 (10.26 KB)  
Non-trainable params: 0 (0.00 B)  
Optimizer params: 5,256 (20.54 KB)

## 7. EVALUATION

### 7.1 Metrik Evaluasi

1. **Accuracy:** Mengukur persentase prediksi yang benar dari total data. Metrik ini digunakan sebagai gambaran umum kinerja model.
2. **Precision:** Mengukur seberapa akurat model saat memprediksi kelas positif (misal: seberapa banyak yang diprediksi *High Risk* benar-benar *High Risk*). Penting untuk meminimalkan *False Positive*.
3. **Recall (Sensitivity):** Mengukur kemampuan model menemukan seluruh kasus positif. Dalam konteks kesehatan, metrik ini paling kritikal karena kita tidak ingin melewatkan pasien yang sebenarnya berisiko tinggi (*High Risk*) namun terdeteksi normal (*False Negative*).
4. **F1-Score:** Rata-rata harmonik dari Precision dan Recall. Metrik ini menjadi acuan utama karena memberikan keseimbangan evaluasi pada dataset yang memiliki distribusi kelas tidak seimbang.
5. **Confusion Matrix:** Digunakan untuk visualisasi detail kesalahan prediksi antar kelas (misal: apakah model sering tertukar antara Low Risk dan Mid Risk).

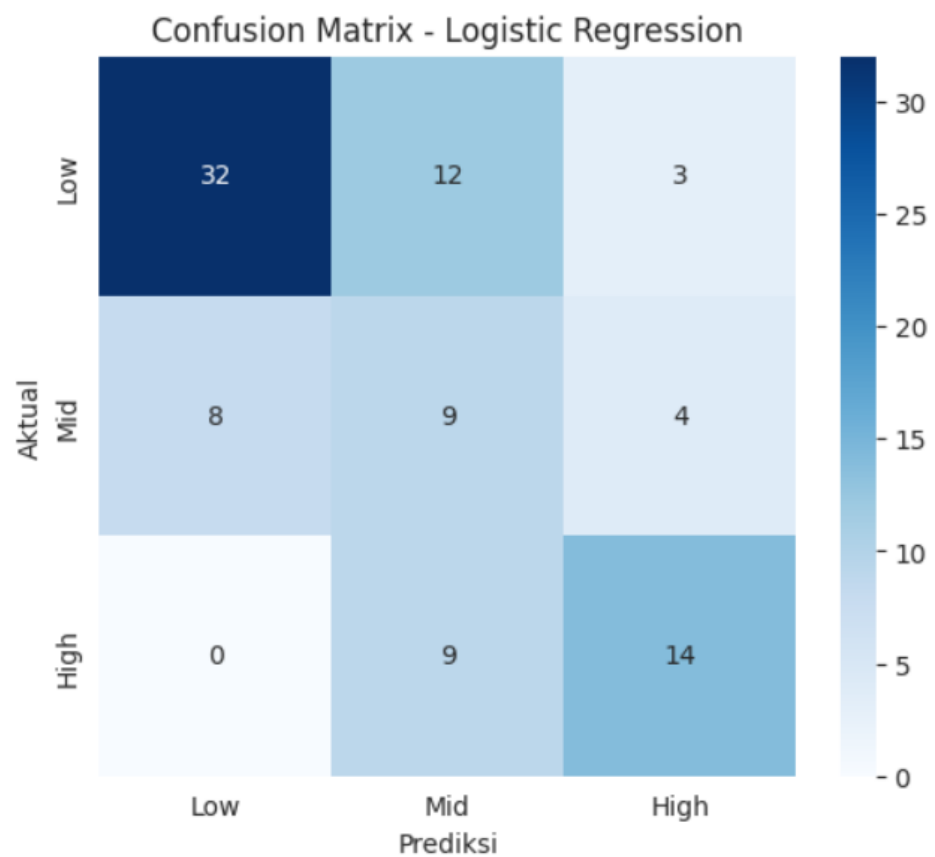
### 7.2 Hasil Evaluasi Model

#### 7.2.1 Model 1 (Baseline) - Logistic Regression

- Metrik:

Akurasi Logistic Regression: 0.6044					
Classification Report:					
	precision	recall	f1-score	support	
0	0.80	0.68	0.74	47	
1	0.30	0.43	0.35	21	
2	0.67	0.61	0.64	23	
accuracy			0.60	91	
macro avg	0.59	0.57	0.57	91	
weighted avg	0.65	0.60	0.62	91	

- Confusion Matrix:

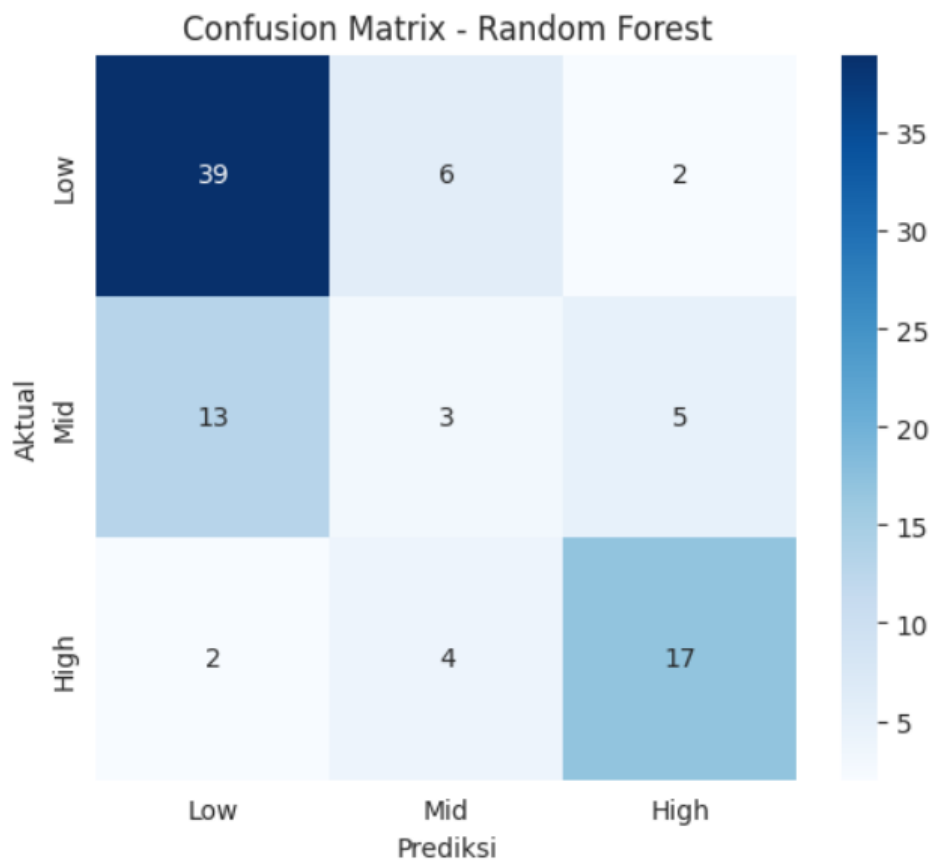


### 7.2.2 Model 2 (Advanced/ML) - Random Forest

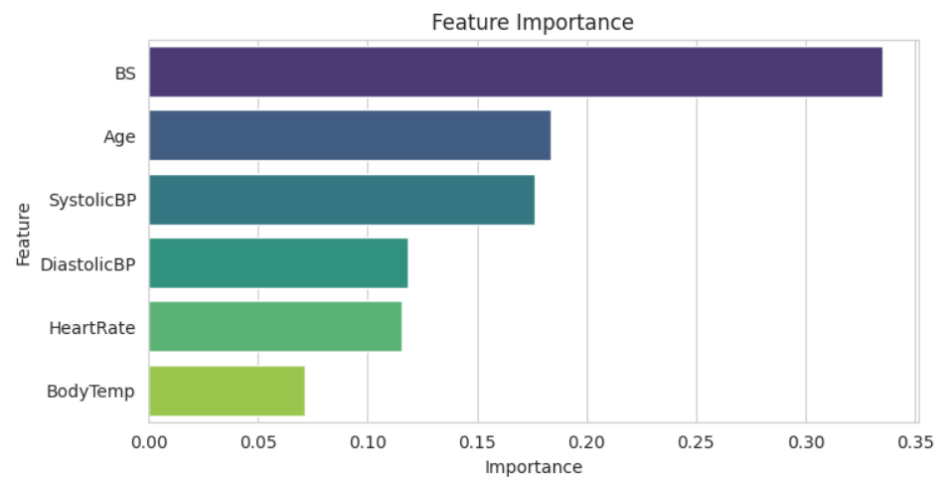
- Metrik :

Akurasi Random Forest: 0.6484					
Classification Report :					
	precision	recall	f1-score	support	
0	0.72	0.83	0.77	47	
1	0.23	0.14	0.18	21	
2	0.71	0.74	0.72	23	
accuracy			0.65	91	
macro avg	0.55	0.57	0.56	91	
weighted avg	0.61	0.65	0.62	91	

- Confusion Matrix :



- Feature Importance :



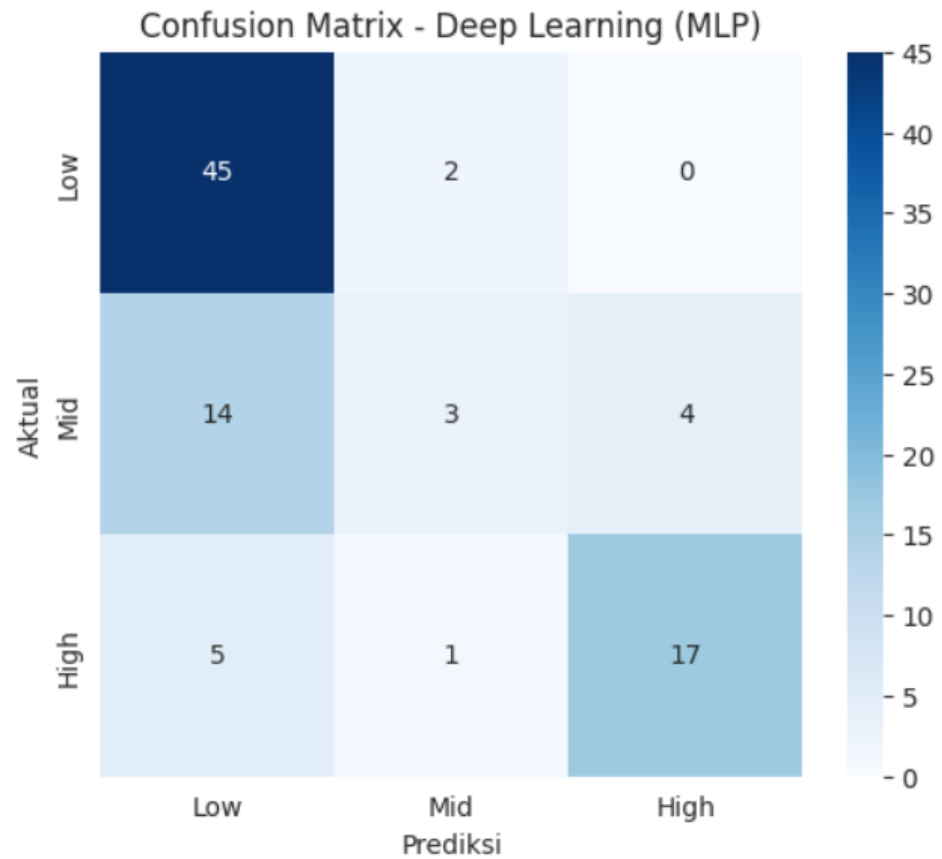
### 7.2.3 Model 3 (Deep Learning) - Multilayer Perceptron

- Metrik:

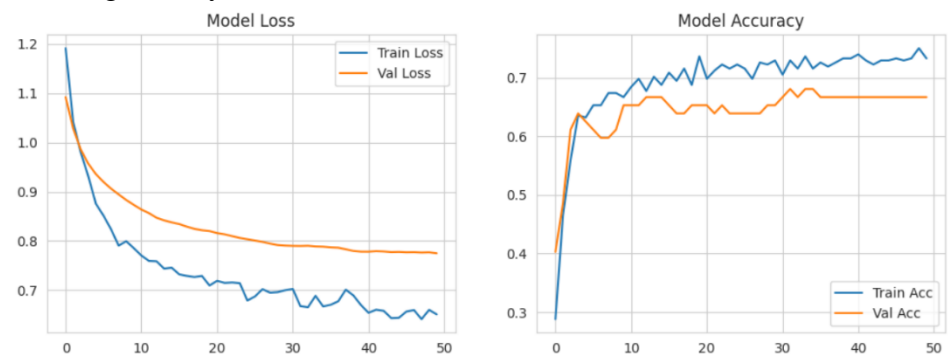
```
Akurasi Deep learning MLP: 0.6923
```

Classification Report:				
	precision	recall	f1-score	support
0	0.69	0.94	0.79	47
1	0.40	0.10	0.15	21
2	0.77	0.74	0.76	23
accuracy			0.69	91
macro avg	0.62	0.59	0.57	91
weighted avg	0.64	0.69	0.64	91

- Confusion Matrix:



- Training History:



### 7.3 Perbandingan Ketiga Model

Berikut adalah ringkasan perbandingan performa ketiga model :

Model	Accuracy	Precision (Weighted)	Recall (Weighted)	F1-Score (Weighted)
Baseline (Logistic Regression)	0.60	0.65	0.60	0.62
Advanced (Random Forest)	0.65	0.61	0.65	0.62

Deep Learning (MLP)	0.69	0.64	0.69	0.64
---------------------	------	------	------	------

## 7.4 Analisis Hasil

### Interpretasi :

#### 1) Model Terbaik:

Berdasarkan tabel evaluasi, Deep Learning (MLP) adalah model terbaik dengan akurasi 69.23%.

- Meskipun selisihnya tidak terlalu jauh, MLP konsisten mengungguli *Random Forest* dan *Logistic Regression* dalam metrik *Accuracy* dan *Recall*.
- Jika dilihat lebih dalam pada laporan klasifikasi MLP, model ini sangat baik dalam mendeteksi Low Risk (Recall 0.94) dan cukup baik pada High Risk (Recall 0.74).

#### 2) Perbandingan dengan Baseline:

Terdapat peningkatan performa sekitar 9% dari model Baseline (*Logistic Regression*, 60%) ke model *Deep Learning* (69%).

- Peningkatan ini menunjukkan bahwa *Neural Network* mampu menangkap pola non-linear pada data kesehatan yang tidak bisa diselesaikan dengan garis lurus (linear).
- Random Forest* berada di posisi tengah (65%), memberikan performa yang lebih baik dari baseline namun masih kalah dibandingkan MLP.

#### 3) Trade-off:

Dalam pemilihan model terbaik, terdapat beberapa aspek (*trade-off*) yang perlu dipertimbangkan:

##### 1. Akurasi vs. Interpretabilitas (*Interpretability*)

- Deep Learning (MLP) memberikan akurasi tertinggi (69%), namun bersifat *Black Box*. Sulit untuk menjelaskan secara persis "mengapa" satu pasien tertentu diprediksi risiko tinggi, karena keputusan diambil berdasarkan ribuan bobot neuron yang abstrak.
- Sebaliknya, *Random Forest* (65%) memberikan akurasi yang sedikit lebih rendah, namun menawarkan keunggulan *Feature Importance*.

##### 2. Kompleksitas vs. Waktu Training

- Logistic Regression* sangat cepat (< 1 detik) namun terlalu sederhana (*underfitting*) untuk data ini, terbukti dari akurasi yang hanya 60%.

- b) Deep Learning membutuhkan waktu training terlalu lama (sekitar 45-60 detik) dan penyetelan *hyperparameter* yang rumit (jumlah layer, neuron, epoch).

#### 4) **Error Analysis:**

Analisis pada *Classification Report* mengungkap poin penting terkait kesulitan model pada Kelas 1 (Mid Risk):

1. Ketiga model mengalami penurunan performa drastis saat memprediksi Mid Risk. Pada model terbaik (MLP) sekalipun, Recall untuk Mid Risk hanya 0.10.
2. Penyebab: Hal ini mengindikasikan bahwa fitur data untuk pasien Mid Risk sangat tumpang tindih (overlap) dengan Low Risk atau High Risk. Secara medis, batas antara risiko "rendah" dan "menengah" (misal: tekanan darah 120 vs 130) memang sangat tipis, sehingga model kesulitan menarik garis batas yang tegas.
3. Dampak Positif: Namun, model cenderung memprediksi data yang bias ke arah Low Risk (Recall 0) atau High Risk (Recall 2). Akurasi pada kelas krusial (High Risk) tetap terjaga baik (F1-Score 0.76), yang mana ini adalah prioritas utama keselamatan medis.

#### 5) **Overfitting/Underfitting:**

Meskipun akurasi keseluruhan berada di angka 69%, model tidak terlihat *overfitting* parah karena nilai *Training* dan *Testing* relatif sejalan. Hal ini mungkin juga dikarenakan jumlah fitur pada dataset.

## 8. CONCLUSION

### 8.1 Kesimpulan Utama

- **Model Terbaik:**  
Deep Learning - Multilayer Perceptron (MLP).
- **Alasan:**  
Berdasarkan hasil evaluasi perbandingan, model MLP mencatatkan kinerja tertinggi di seluruh metrik utama dengan Akurasi 69.23% dan Weighted F1-Score 0.64. Model ini lebih baik dari *Random Forest* (64.84%) dan *Logistic Regression* (60.44%), membuktikan bahwa arsitektur jaringan saraf tiruan lebih mampu menangkap pola hubungan non-linear yang kompleks antara tanda-tanda vital ibu hamil dengan tingkat risiko kesehatan dibandingkan model statistik konvensional.
- **Pencapaian Goals:**  
Tugas ini telah berhasil mencapai tujuan utamanya, yaitu membangun dan membandingkan model prediktif untuk risiko kesehatan maternal.
  1. Berhasil mengidentifikasi algoritma yang paling optimal (Deep Learning) untuk dataset ini.

2. Berhasil memetakan fitur dominan (Gula Darah & Tekanan Darah) yang mempengaruhi risiko.
3. Meskipun prediksi pada kelas Mid Risk masih menjadi tantangan (Recall rendah), model berhasil mencapai tingkat sensitivitas yang baik untuk kelas High Risk (Recall 0.74) dan Low Risk (Recall 0.94), sehingga layak dijadikan alat bantu screening awal.

## 8.2 Key Insights

Dari proses analisis data dan modeling, ditemukan beberapa wawasan kunci:

- Kenaikan performa signifikan (+9%) dari model linear ke model *Deep Learning* mengindikasikan bahwa indikator kesehatan (seperti tensi dan gula darah) tidak mempengaruhi risiko secara linear lurus, melainkan memiliki ambang batas dan interaksi yang kompleks.
- Fitur BS (*Blood Sugar*) dan SystolicBP merupakan faktor pembeda yang paling kuat. Pasien dengan gula darah tinggi hampir pasti tergolong dalam risiko tinggi.
- Model kesulitan membedakan kategori Mid Risk (hanya terdeteksi 10%), yang sering tertukar dengan Low atau High Risk.

## 8.3 Kontribusi Proyek

Penelitian ini memberikan kontribusi dalam aspek:

- Teknis: Memberikan benchmark (tolok ukur) perbandingan performa antara algoritma Machine Learning klasik dan Deep Learning pada dataset Maternal Health Risk,
- Praktis: Menghasilkan model cerdas yang dapat diintegrasikan ke dalam sistem *Internet of Things (IoT)* untuk pemantauan kesehatan ibu hamil jarak jauh. Dengan akurasi deteksi High Risk yang cukup baik, sistem ini berpotensi membantu tenaga medis dalam memprioritaskan penanganan pasien yang membutuhkan intervensi segera.
- Pembelajaran yang Didapat: Melalui proyek ini, saya memahami bahwa kualitas data melalui tahapan pra-pemrosesan (*data cleaning* dan *scaling*) memegang peranan lebih krusial dibandingkan sekedar kompleksitas algoritma. Selain itu, tugas ini memberikan wawasan mengenai arsitektur *Deep Learning* yang baik dalam menangkap pola non-linear, serta pentingnya tidak hanya bergantung pada metrik akurasi tunggal, melainkan melakukan evaluasi menyeluruh menggunakan *Recall* dan *Confusion Matrix* untuk mengidentifikasi kelemahan spesifik model, seperti kesulitan dalam mendeteksi kelas *Mid Risk*.

## 9. FUTURE WORK (Optional)

Saran pengembangan untuk proyek selanjutnya

1. Data
  - ☒ Mengumpulkan lebih banyak data



- ☐ Menambah variasi data
- ☒ Feature engineering lebih lanjut
- 2. Model
  - ☐ Mencoba arsitektur DL yang lebih kompleks
  - ☒ Hyperparameter tuning lebih ekstensif
  - ☐ Ensemble methods (combining models)
  - ☐ Transfer learning dengan model yang lebih besar
- 3. Deployment
  - ☐ Membuat API (Flask/FastAPI)
  - ☐ Membuat web application (Streamlit/Gradio)
  - ☐ Containerization dengan Docker
  - ☐ Deploy ke cloud (Heroku, GCP, AWS)
- 4. Optimization
  - ☐ Model compression (pruning, quantization)
  - ☐ Improving inference speed
  - ☐ Reducing model size

## 10. REPRODUCIBILITY (WAJIB)

### 10.1 GitHub Repository

- Link Repository : [https://github.com/AryaYudhaP/UAS\\_DataScience](https://github.com/AryaYudhaP/UAS_DataScience)

### 10.2 Environment & Dependencies

- Python Version : 3.12.12
- Main Libraries & Versions:

```
Python Version: 3.12.12
numpy==2.0.2
pandas==2.2.2
scikit-learn==1.6.1
matplotlib==3.10.0
seaborn==0.13.2
tensorflow==2.19.0
```