INFORMASI PROYEK

Judul Proyek: Penerapan Metode Machine Learning untuk Klasifikasi Gangguan Hati Menggunakan Dataset Liver Disorders.

Nama Mahasiswa	:	Diaz Azkha Varissa
NIM	:	233307043
Program Studi	:	D-III Teknologi Informasi
Mata Kuliah	:	Data Science
Dosen Pengampu	:	Gus Nanang Syaifuddiin, S.Kom., M.Kom.
Tahun Akademik	:	2025
Link GitHub Repository	:	https://github.com/dzav97/UAS_DataScience-LD.git
Link Video Pembahasan	:	

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

Konsumsi alkohol berlebih dapat berdampak pada kesehatan, termasuk meningkatkan risiko gangguan fungsi hati dan penurunan kualitas hidup jika tidak terdeteksi sejak dini. Beberapa parameter laboratorium seperti GGT serta enzim hati lain (misalnya ALT/AST) sering digunakan sebagai indikator adanya stres atau kerusakan pada hati, sementara MCV juga kerap meningkat pada konsumsi alkohol berat kronis. Karena hubungan antara konsumsi alkohol dan perubahan biomarker ini tidak selalu linear serta bisa dipengaruhi faktor lain (misalnya pola hidup dan kondisi metabolik), pendekatan analitik berbasis data dapat membantu menangkap pola yang lebih kompleks dibandingkan analisis manual semata.

Dalam konteks analitik data, hasil tes darah dapat dimanfaatkan untuk membangun model yang memprediksi pola konsumsi alkohol atau mengelompokkan individu berdasarkan tingkat risikonya. Dataset Liver Disorders (BUPA) dari UCI Machine Learning Repository menyediakan data hasil tes darah (MCV, ALKPHOS, SGPT, SGOT, GAMMAGT) beserta estimasi konsumsi alkohol harian (drinks) pada 345 individu laki-laki. Dataset ini relevan sebagai studi kasus karena fitur-fiturnya merepresentasikan indikator klinis yang umum dan

mudah diukur, sehingga hasil pemodelan berpotensi mendukung proses skrining awal, pemantauan kesehatan, maupun pengambilan keputusan berbasis bukti. Melalui proyek ini, model machine learning dan deep learning akan dibandingkan untuk melihat pendekatan mana yang paling efektif dan stabil dalam memprediksi target, sekaligus memastikan proses eksperimen dapat direplikasi melalui dokumentasi kode dan konfigurasi lingkungan.

Referensi:
Liver Disorders [Dataset]. (2016). UCI Machine Learning Repository. https://doi.org/10.24432/C54G67\
Anton, R. F., et al. (2022). Biomarkers and Clinical Laboratory Detection of Acute and Chronic Ethanol Use. Clinical Chemistry, 68(5), 635–649.

3. BUSINESS UNDERSTANDING/PROBLEM UNDERSTANDING

3.1 Problem Statements

Berdasarkan karakteristik dataset Liver Disorders dan tujuan proyek ini, maka pernyataan masalah dapat dirumuskan sebagai berikut:

1. Dataset Liver Disorders berisi data numerik hasil tes darah dan konsumsi alkohol yang perlu dimanfaatkan untuk memprediksi label selector sebagai indikasi gangguan hati secara akurat.
2. Dataset memiliki distribusi kelas yang tidak seimbang sehingga diperlukan proses preprocessing dan pemilihan model yang tepat agar performa klasifikasi tidak bias.
3. Dibutuhkan perbandingan performa antara beberapa pendekatan model, mulai dari model sederhana hingga model yang lebih kompleks, untuk mengetahui metode terbaik dalam menyelesaikan permasalahan klasifikasi ini.
4. Diperlukan model deep learning yang mampu mempelajari pola dan hubungan kompleks antar fitur numerik pada data tabular.

3.2 Goals

Tujuan dari proyek ini adalah:

1. Membangun model machine learning untuk mengklasifikasikan kondisi gangguan hati pada dataset Liver Disorders dengan performa yang baik.
2. Mengukur dan membandingkan performa tiga pendekatan model, yaitu model baseline, model machine learning lanjutan, dan model deep learning.
3. Menentukan model terbaik berdasarkan metrik evaluasi klasifikasi seperti accuracy, precision, recall, dan F1-score.
4. Menghasilkan pipeline pemodelan yang reproducible, sehingga eksperimen dapat dijalankan kembali dengan hasil yang konsisten.

3.3 Solution Approach

Pada proyek ini digunakan tiga model dengan tingkat kompleksitas yang berbeda, sesuai dengan ketentuan tugas:

1. Model 1 – Baseline Model (Logistic Regression)

   Logistic Regression dipilih sebagai model baseline karena merupakan algoritma klasifikasi yang sederhana, mudah diinterpretasikan, dan sering

digunakan sebagai pembanding awal pada permasalahan klasifikasi biner. Model ini bekerja dengan memodelkan probabilitas kelas menggunakan fungsi sigmoid dan memberikan gambaran performa dasar sebelum menggunakan model yang lebih kompleks.

Sebagai baseline, Logistic Regression membantu menentukan apakah peningkatan performa pada model lanjutan benar-benar signifikan dibandingkan pendekatan sederhana.

2. Model 2 – Advanced / Machine Learning Model (Random Forest)

   Random Forest dipilih sebagai model machine learning lanjutan karena merupakan metode ensemble yang mampu menangkap hubungan non-linear antar fitur. Model ini bekerja dengan membangun banyak decision tree dan menggabungkan hasil prediksinya sehingga lebih robust terhadap noise dan overfitting dibandingkan single decision tree.

   Random Forest juga cocok untuk dataset tabular numerik seperti Liver Disorders dan sering memberikan performa yang lebih baik dibandingkan model linear pada data dengan pola kompleks.

3. Model 3 – Deep Learning Model (Multilayer Perceptron (MLP) / Neural Network)

   Karena dataset Liver Disorders merupakan data tabular numerik, model deep learning yang digunakan adalah Multilayer Perceptron (MLP). Model ini terdiri dari:

   - Input layer sesuai jumlah fitur,
   - Minimal dua hidden layer dengan fungsi aktivasi non-linear,
   - Output layer untuk klasifikasi biner.

   MLP dipilih karena mampu mempelajari representasi fitur yang lebih kompleks dibandingkan model machine learning tradisional. Model ini akan dilatih dengan:

   - Minimal 10 epochs,
   - Pencatatan training time,
   - Visualisasi grafik loss dan accuracy per epoch,
   - Evaluasi hasil prediksi pada data uji (test set).

   Penggunaan MLP diharapkan dapat menunjukkan perbedaan performa antara pendekatan deep learning dan model machine learning konvensional pada permasalahan klasifikasi gangguan hati..

4. DATA UNDERSTANDING

4.1 Informasi Dataset

Sumber Dataset: UCI Machine Learning Repository
URL: https://archive.ics.uci.edu/dataset/60/liver+disorders
Deskripsi Dataset:
Dataset Liver Disorders (BUPA) merupakan dataset tabular yang berisi data hasil pemeriksaan laboratorium dan informasi konsumsi alkohol yang digunakan untuk mengklasifikasikan gangguan hati.

- Jumlah baris (rows): 345
- Jumlah kolom (features): 7
- Tipe data: Tabular
- Ukuran dataset: ± 0.03 MB
- Format file: CSV

4.2 Deskripsi Fitur

Dataset Liver Disorders terdiri dari enam fitur input dan satu fitur target, sebagaimana ditunjukkan pada Tabel berikut:

Nama Fitur	Tipe Data	Deskripsi	Contoh Nilai
mcv	Continuous	Mean Corpuscular Volume	85, 90
alkphos	Continuous	Alkaline Phosphotase	92, 120
sgpt	Continuous	Alanine Aminotransferase	20, 35
sgot	Continuous	Aspartate Aminotransferase	15, 40
gammagt	Continuous	Gamma-Glutamyl Transpeptidase	25, 75
drinks	Continuous	Jumlah konsumsi alkohol per hari	0, 5
selector	Categorical	Label target (1 atau 2)	1, 2

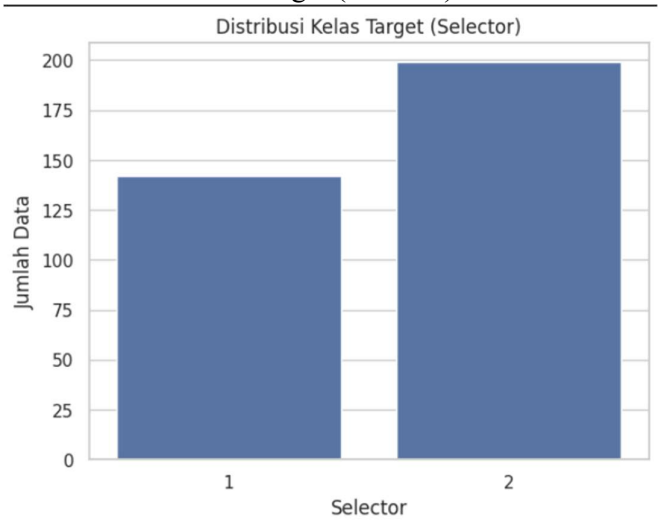
4.3 Kondisi Data

Berdasarkan hasil eksplorasi awal, kondisi dataset dapat dijelaskan sebagai berikut:

- Missing Values: Tidak terdapat missing value pada dataset.
- Duplicate Data: Ditemukan sebanyak 4 data duplikat pada dataset.
- Outliers: Terdapat indikasi outlier pada beberapa fitur numerik seperti gammagt dan drinks.
- Imbalanced Data: Dataset memiliki distribusi kelas yang tidak seimbang, di mana kelas 2 lebih dominan dibandingkan kelas 1.
- Noise: Kemungkinan terdapat noise akibat variasi biologis dan gaya hidup individu.
- Data Quality Issues: Tidak terdapat masalah kualitas data yang signifikan, dataset dapat langsung digunakan setelah preprocessing dasar.

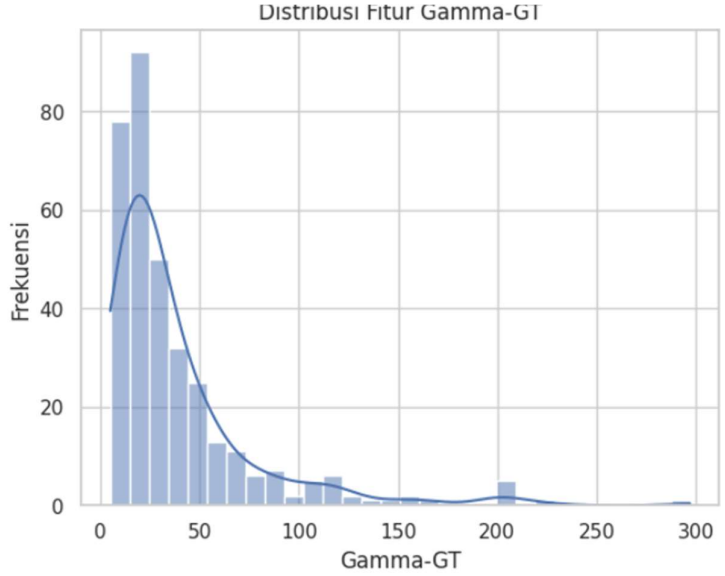
4.4 Exploratory Data Analysis (EDA)

- Visualisasi 1: Distribusi Kelas Target (Selector)



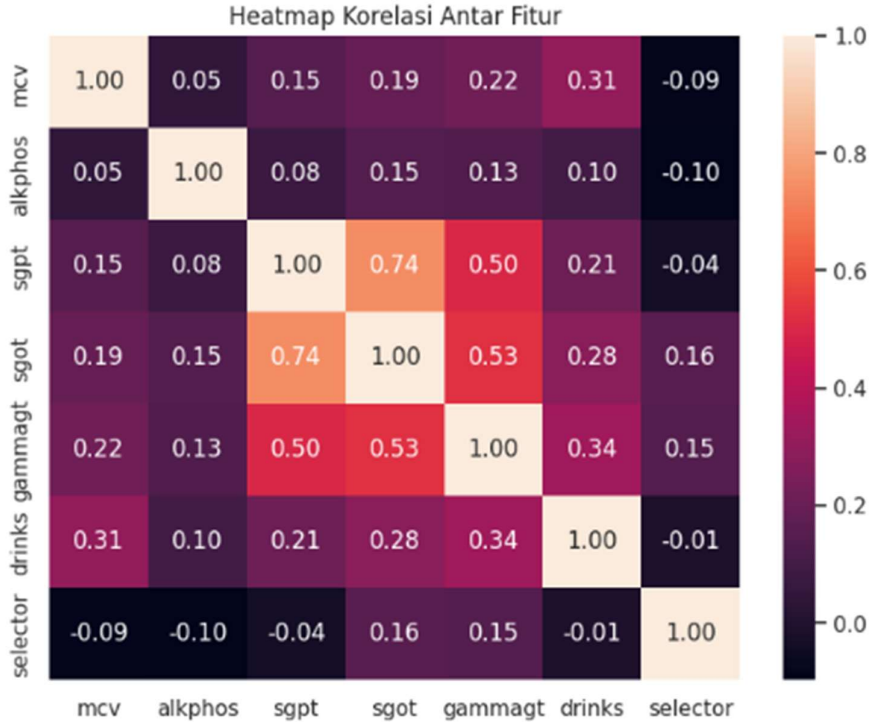
Distribusi label menunjukkan bahwa kelas 2 memiliki jumlah data yang lebih banyak dibandingkan kelas 1, sehingga dataset bersifat imbalanced. Hal ini perlu diperhatikan pada tahap pemodelan.

- Visualisasi 2: Distribusi Fitur Gamma-GT



Distribusi fitur gammagt bersifat skewed ke kanan, menunjukkan adanya nilai ekstrim (outliers) yang berpotensi memengaruhi performa model.

- Visualisasi 3: Heatmap Korelasi Antar Fitur



Beberapa fitur memiliki korelasi positif sedang, seperti antara sgpt, sgot, dan gammagt. Informasi ini berguna untuk memahami hubungan antar fitur sebelum pemodelan.

5. DATA PREPARATION

5.1 Data Cleaning

a. Handling Missing Values

Berdasarkan hasil data understanding, tidak ditemukan missing values pada seluruh fitur dataset Liver Disorders. Oleh karena itu, tidak dilakukan proses imputasi pada tahap ini.

Alasan:

Karena tidak terdapat nilai kosong, data dapat langsung digunakan tanpa risiko distorsi nilai akibat imputasi.

b. Removing Duplicates

Ditemukan sebanyak 4 data duplikat pada dataset. Data duplikat tersebut dihapus untuk mencegah bias pada proses pelatihan model.

Alasan:

Data duplikat dapat memberikan bobot berlebih pada pola tertentu dan memengaruhi hasil evaluasi model.

c. Handling Outliers

Berdasarkan analisis EDA, terdapat indikasi outliers pada beberapa fitur numerik seperti gammagt dan drinks. Namun, outliers tersebut masih dianggap relevan secara medis karena dapat merepresentasikan kondisi ekstrem pada pasien.

Keputusan:

Outliers tidak dihapus, tetapi akan ditangani secara tidak langsung melalui proses scaling pada tahap transformasi data.

d. Data Type Conversion

Seluruh fitur pada dataset sudah bertipe numerik (integer), sehingga tidak diperlukan konversi tipe data tambahan.

5.2 Feature Engineering

Pada proyek ini tidak dilakukan pembuatan fitur baru karena seluruh fitur yang tersedia sudah relevan secara domain dan jumlah fitur relatif sedikit.

Namun, dilakukan feature selection implisit dengan:

- Memisahkan fitur input (X) dan label target (y)
- Menggunakan seluruh fitur numerik sebagai prediktor

Alasan:

Dataset berskala kecil dan seluruh fitur memiliki keterkaitan langsung dengan kondisi hati, sehingga penghapusan fitur berpotensi menghilangkan informasi penting.

5.3 Data Transformation

a. Label Encoding (Target)

Label target selector memiliki nilai 1 dan 2, sehingga dilakukan mapping menjadi:

- $1 \rightarrow 0$
- $2 \rightarrow 1$

Alasan: Model machine learning dan deep learning umumnya membutuhkan label dalam bentuk numerik biner (0 dan 1).

b. Feature Scaling
Dilakukan Standardization (StandardScaler) pada fitur numerik.
Alasan:
- Beberapa model (Logistic Regression dan Neural Network) sensitif terhadap skala fitur
- StandardScaler membuat fitur memiliki mean 0 dan standar deviasi 1
- Membantu proses training menjadi lebih stabil dan cepat

5.4 Data Splitting

Dataset dibagi menggunakan stratified split untuk mempertahankan proporsi kelas pada data latih dan data uji. Adapun strategi pembagian datanya yaitu:

- Training set: 80%
- Test set: 20%
- Random state: 42 (untuk reproducibility)

Alasan: Stratified split digunakan karena dataset memiliki distribusi kelas yang tidak seimbang.

5.5 Data Balancing

Pada tahap ini tidak dilakukan data balancing seperti SMOTE atau undersampling.
Alasan:

- Selisih jumlah kelas tidak terlalu ekstrem
- Digunakan stratified split untuk menjaga proporsi kelas
- Evaluasi performa model akan memperhatikan metrik selain accuracy (precision, recall, F1-score)

Namun, penggunaan teknik balancing tetap dipertimbangkan sebagai pengembangan lanjutan.

5.6 Ringkasan Data Preparation

Tahap data preparation dilakukan dengan menghapus 4 data duplikat untuk mencegah bias, sementara tidak ditemukan missing values sehingga tidak diperlukan imputasi. Outliers pada beberapa fitur numerik tidak dihapus karena masih relevan secara medis dan ditangani melalui proses scaling. Label target dimapping menjadi bentuk biner (0 dan 1), kemudian fitur dinormalisasi menggunakan StandardScaler. Dataset dibagi menggunakan stratified split dengan proporsi 80% data latih dan 20% data uji tanpa menerapkan teknik data balancing.

5. Split features & target + train/test split

```
X = df.drop("selector", axis=1)
y = df["selector"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    stratify=y,
    random_state=42
)

print("Train shape:", X_train.shape, "Test shape:", X_test.shape)
```

```
Train shape: (272, 6) Test shape: (69, 6)
```

6. Feature scaling (untuk Logistic Regression & MLP)

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled  = scaler.transform(X_test)

print("Scaled train shape:", X_train_scaled.shape)
print("Scaled test shape :", X_test_scaled.shape)
```

```
Scaled train shape: (272, 6)
Scaled test shape : (69, 6)
```

```
def eval_cls(y_true, y_pred, title="Model"):
    acc = accuracy_score(y_true, y_pred)
    prec = precision_score(y_true, y_pred, zero_division=0)
    rec = recall_score(y_true, y_pred, zero_division=0)
    f1 = f1_score(y_true, y_pred, zero_division=0)

    print(f"\n=== {title} ===")
    print(f"Accuracy : {acc:.4f}")
    print(f"Precision: {prec:.4f}")
    print(f"Recall   : {rec:.4f}")
    print(f"F1-score : {f1:.4f}")

    print("\nClassification report:")
    print(classification_report(y_true, y_pred, digits=4))

    print("Confusion matrix:")
    print(confusion_matrix(y_true, y_pred))

    return {"accuracy": acc, "precision": prec, "recall": rec, "f1": f1}
```

6. MODELING

6.1 Model 1 — Baseline Model

6.1.1 Deskripsi Model

Nama Model: Logistic Regression

Logistic Regression merupakan algoritma klasifikasi linear yang memodelkan probabilitas suatu kelas menggunakan fungsi sigmoid. Model ini menghitung kombinasi linear dari fitur input dan mengubahnya menjadi probabilitas kelas antara 0 dan 1, sehingga cocok untuk permasalahan klasifikasi biner.

Alasan Pemilihan:

Logistic Regression dipilih sebagai model baseline karena sederhana, mudah diinterpretasikan, dan sering digunakan sebagai pembanding awal untuk mengetahui performa dasar sebelum menerapkan model yang lebih kompleks.

6.1.2 Hyperparameter

Parameter yang digunakan:

- C (regularization): 1.0
- solver: 'lbfgs'
- max_iter: 100

6.1.3 Implementasi

```
model_baseline = LogisticRegression(
    C=1.0,
    solver="lbfgs",
    max_iter=100
)
model_baseline.fit(X_train_scaled, y_train)

t1 = time.perf_counter()
train_time_lr = t1 - t0

y_pred_baseline = model_baseline.predict(X_test_scaled)
```

6.1.4 Hasil Awal

Model Logistic Regression mampu melakukan klasifikasi pada dataset Liver Disorders dengan performa yang cukup baik sebagai baseline. Hasil evaluasi awal menunjukkan bahwa model ini dapat digunakan sebagai pembanding terhadap model yang lebih kompleks.

6.2 Model 2 — ML / Advanced Model

6.2.1 Deskripsi Model

Nama Model: Random Forest Classifier

Teori Singkat:

Random Forest merupakan metode ensemble yang membangun banyak decision tree menggunakan subset data dan fitur yang berbeda. Prediksi akhir diperoleh melalui voting mayoritas dari seluruh pohon, sehingga model menjadi lebih stabil dan robust.

Alasan Pemilihan:

Random Forest dipilih karena mampu menangkap hubungan non-linear antar fitur, cocok untuk data tabular numerik, dan sering memberikan performa lebih baik dibandingkan model linear.

Keunggulan:

- Robust terhadap noise dan overfitting
- Dapat menangani hubungan non-linear
- Tidak terlalu sensitif terhadap outliers

Kelemahan:

- Kurang interpretatif dibandingkan model linear

- Waktu training lebih lama dibanding baseline

6.2.2 Hyperparameter

Parameter yang digunakan:

- n_estimators: 100
- max_depth: 10
- random_state: 42

6.2.3 Implementasi

```
model_advanced = RandomForestClassifier(
    n_estimators=100,
    max_depth=10,
    random_state=42
)
model_advanced.fit(X_train, y_train)

t1 = time.perf_counter()
train_time_rf = t1 - t0

y_pred_advanced = model_advanced.predict(X_test)
```

6.2.4 Hasil Model

Random Forest menunjukkan performa yang lebih baik dibandingkan model baseline karena kemampuannya dalam menangkap pola non-linear pada data.

6.3 Model 3 — Deep Learning Model

6.3.1 Deskripsi Model

Nama Model: Multilayer Perceptron (MLP)
Jenis Deep Learning:
Multilayer Perceptron (MLP) – untuk data tabular
Alasan Pemilihan:
Dataset Liver Disorders merupakan data tabular numerik, sehingga arsitektur MLP sangat sesuai untuk mempelajari hubungan kompleks antar fitur tanpa memerlukan struktur spasial atau sekuensial.

6.3.2 Arsitektur Model

Deskripsi Layer:

- Input Layer: jumlah neuron sesuai jumlah fitur (6)
- Dense Layer: 64 neuron, activation = ReLU
- Dropout Layer: 0.3
- Dense Layer: 32 neuron, activation = ReLU
- Dropout Layer: 0.3
- Output Layer: 1 neuron, activation = Sigmoid

6.3.3 Input & Preprocessing Khusus

Input shape: (6,)
Preprocessing khusus:

- Standardization menggunakan StandardScaler
- Tidak dilakukan data augmentation karena data bersifat tabular

6.3.4 Hyperparameter

```
history = model_mlp.fit(
    X_train_scaled, y_train,
    validation_split=0.2,
    epochs=20,              # >= 10
    batch_size=16,
    callbacks=[early_stopping],
    verbose=1
)
```

6.3.5 Implementasi

```
tf.random.set_seed(42)
np.random.seed(42)

input_dim = X_train_scaled.shape[1]

model_mlp = keras.Sequential([
    keras.layers.Input(shape=(input_dim,)),
    keras.layers.Dense(64, activation="relu"),
    keras.layers.Dropout(0.3),
    keras.layers.Dense(32, activation="relu"),
    keras.layers.Dropout(0.3),
    keras.layers.Dense(1, activation="sigmoid")
])

model_mlp.compile(
    optimizer=keras.optimizers.Adam(learning_rate=0.001),
    loss="binary_crossentropy",
    metrics=["accuracy"]
)

model_mlp.summary()
```

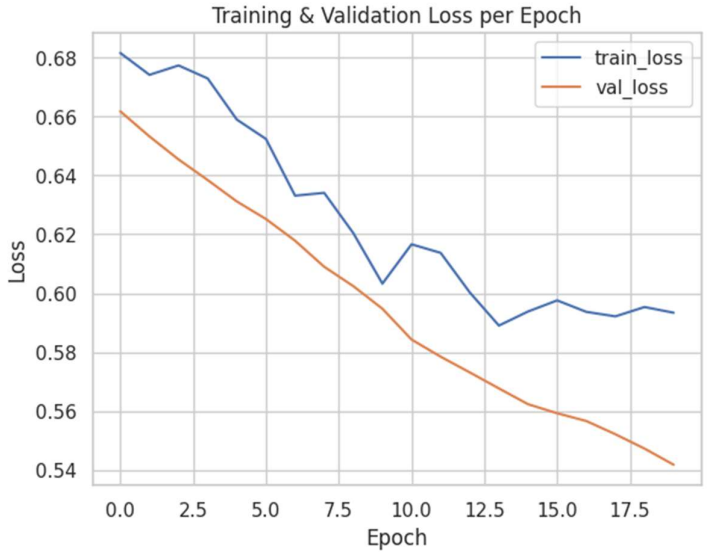
6.3.6 Training Process

Training Time: Waktu training total model Multilayer Perceptron (MLP) adalah 22,22 detik, dihitung dari awal proses training hingga selesai pada epoch ke-50.

Computational Resource: Proses training dilakukan menggunakan CPU pada lingkungan Local / Google Colab, tanpa akselerasi GPU. Hal ini memadai karena ukuran dataset relatif kecil (341 data).

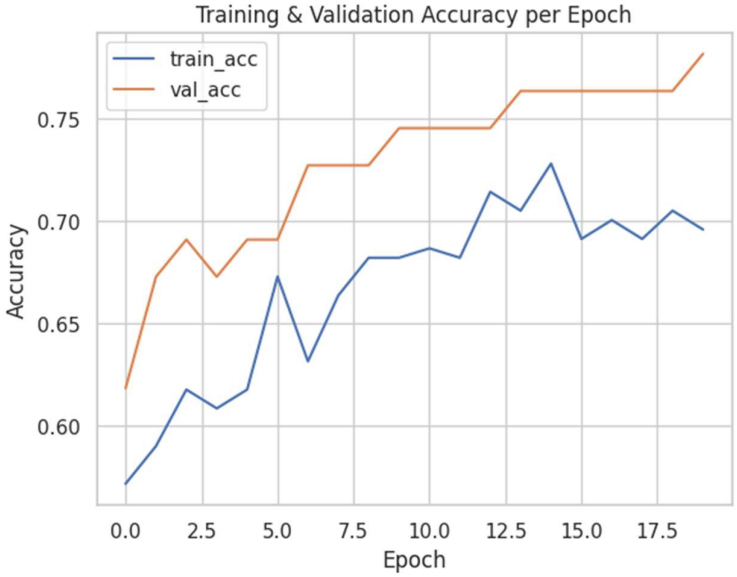
Training History Visualization

a. Training & Validation Loss per Epoch



Berdasarkan grafik training vs validation loss, terlihat bahwa nilai training loss menurun dari sekitar 0,68 di awal menjadi sekitar 0,59 di akhir epoch, meskipun sempat berfluktuasi kecil di beberapa titik. Sementara itu, validation loss turun lebih stabil dari sekitar 0,66 hingga sekitar 0,54 dan konsisten berada di bawah training loss. Pola ini menunjukkan bahwa model benar-benar belajar dan semakin mampu meminimalkan kesalahan pada data latih maupun data validasi, serta tidak menunjukkan gejala overfitting, karena pada kasus overfitting biasanya validation loss justru berhenti turun atau naik ketika training loss terus turun. Kondisi validation loss yang lebih kecil dari training loss juga sering terjadi ketika training dibuat lebih "berat", misalnya karena penggunaan dropout, regularisasi, atau augmentasi, sehingga performa di training terlihat sedikit lebih rendah dibanding validasi.

b. Training & Validation Accuracy per Epoch



Pada grafik training vs validation accuracy, training accuracy meningkat dari sekitar 0,57 menjadi sekitar 0,70 dengan sedikit naik-turun, sedangkan validation accuracy meningkat lebih konsisten dari sekitar 0,62 hingga mencapai sekitar 0,78 dan tetap lebih tinggi dibanding training accuracy. Hal ini menandakan kemampuan generalisasi model cukup baik karena akurasi validasi ikut membaik seiring waktu, bukan stagnan atau menurun. Perbedaan di mana akurasi validasi lebih tinggi dari training bisa disebabkan oleh faktor seperti dropout/augmentasi yang hanya aktif saat training atau karena data validasi relatif lebih mudah, namun untuk memastikan tidak ada masalah seperti data leakage, tetap disarankan mengecek proses preprocessing (fit hanya pada train) dan menguji performa pada test set terpisah.

Analisis Training

a. Apakah model mengalami overfitting?
   Tidak. Grafik menunjukkan train loss dan val loss sama-sama turun sampai akhir epoch, bahkan val loss terus membaik tanpa ada kenaikan di bagian akhir. Selain itu, val accuracy juga meningkat dan tidak mengalami penurunan, sehingga tidak ada tanda klasik overfitting (biasanya val loss naik dan val acc turun saat train terus membaik).

b. Apakah model sudah converge?
   Belum sepenuhnya. Indikasinya, validation loss masih terus menurun dan validation accuracy masih meningkat hingga epoch terakhir. Kalau sudah converge, biasanya perbaikan metrik validasi mulai sangat kecil atau datar (plateau) selama beberapa epoch.

c. Apakah perlu lebih banyak epoch?
   Ya, masih bisa dicoba, karena tren metrik validasi masih membaik di akhir training.

6.3.7 Model Summary

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, 1)	33

Total params: 2,561 (10.00 KB)
Trainable params: 2,561 (10.00 KB)
Non-trainable params: 0 (0.00 B)

7. EVALUATION

7.1 Metrik Evaluasi

Karena permasalahan yang diselesaikan merupakan klasifikasi biner, maka metrik evaluasi yang digunakan adalah:

- Accuracy: Mengukur proporsi prediksi yang benar dari seluruh data uji.
- Precision: Mengukur ketepatan prediksi kelas positif, yaitu seberapa banyak prediksi positif yang benar.
- Recall: Mengukur kemampuan model dalam menemukan seluruh data positif yang sebenarnya.
- F1-Score: Merupakan rata-rata harmonik dari precision dan recall, digunakan untuk memberikan gambaran performa model secara seimbang, terutama pada data yang tidak seimbang.
- Confusion Matrix: Digunakan untuk melihat distribusi prediksi benar dan salah pada masing-masing kelas.

7.2 Hasil Evaluasi Model

7.2.1 Model 1 Baseline (Logistic Regression)

```
=== Logistic Regression (Baseline) ===
Accuracy : 0.6957
Precision: 0.7021
Recall   : 0.8250
F1-score : 0.7586

Classification report:
              precision    recall  f1-score   support

           0     0.6818    0.5172    0.5882        29
           1     0.7021    0.8250    0.7586        40

    accuracy                         0.6957        69
   macro avg     0.6920    0.6711    0.6734        69
weighted avg     0.6936    0.6957    0.6870        69
```

7.2.2 Model 2 Advanced / Machine Learning (Random Forest)

```
=== Random Forest (Advanced) ===
Accuracy : 0.7391
Precision: 0.7292
Recall   : 0.8750
F1-score : 0.7955

Classification report:
              precision    recall  f1-score   support

           0     0.7619    0.5517    0.6400        29
           1     0.7292    0.8750    0.7955        40

    accuracy                         0.7391        69
   macro avg     0.7455    0.7134    0.7177        69
weighted avg     0.7429    0.7391    0.7301        69
```

7.2.3 Model 3 Deep Learning (MLP)

```
=== MLP (Deep Learning) ===
Accuracy : 0.7826
Precision: 0.7660
Recall   : 0.9000
F1-score : 0.8276

Classification report:
              precision    recall  f1-score   support

           0     0.8182    0.6207    0.7059        29
           1     0.7660    0.9000    0.8276        40

    accuracy                         0.7826        69
   macro avg     0.7921    0.7603    0.7667        69
weighted avg     0.7879    0.7826    0.7764        69
```

7.3 Perbandingan Ketiga Model

model	train_time_sec	accuracy	precision	recall	f1
MLP (Deep Learning)	4.654213131999995	0.782608695652174	0.7659574468085106	0.9	0.8275862068965517
Random Forest	0.5745210530000122	0.7391304347826086	0.7291666666666666	0.875	0.7954545454545454
Logistic Regression	0.027003971999988607	0.6956521739130435	0.7021276595744681	0.825	0.7586206896551724

7.4 Analisis Hasil

1. Model Terbaik: Berdasarkan tabel evaluasi, MLP (Deep Learning) adalah model terbaik karena menghasilkan akurasi tertinggi (0,7826) dan F1-score tertinggi (0,8276), serta recall paling tinggi (0,90).

2. Perbandingan dengan Baseline: Jika baseline dianggap Logistic Regression, maka terjadi peningkatan yang jelas pada dua model lain. Dibanding Logistic Regression (akurasi 0,6957, F1 0,7586), Random Forest meningkat menjadi akurasi 0,7391 (naik ~0,0435) dan F1 0,7955 (naik ~0,0369). Peningkatan terbesar ada pada MLP, yaitu akurasi 0,7826 (naik ~0,0870) dan F1 0,8276 (naik ~0,0690). Selain itu, recall juga naik dari 0,825 (LR) → 0,875 (RF) → 0,90 (MLP), menunjukkan model yang lebih kompleks lebih baik dalam mendeteksi kelas positif.

3. Trade-off: LR paling cepat & sederhana (0,027s) tapi performa terendah. RF sedang (0,57s) performa bagus. MLP performa terbaik tapi training paling lama (4,65s) dan paling kompleks.

4. Error Analysis: Karena recall > precision, model cenderung lebih sering false positive (memprediksi positif padahal negatif), terutama pada data yang nilainya mirip antar kelas.

5. Overfitting/Underfitting: Tidak terlihat overfitting. LR cenderung underfitting (model terlalu sederhana). RF & MLP lebih pas.

8. CONCLUSION

8.1 Kesimpulan Utama

Berdasarkan hasil evaluasi, model terbaik pada penelitian ini adalah MLP (Deep Learning) karena menghasilkan performa tertinggi dengan akurasi sekitar 0,783, F1 sekitar 0,828, dan recall 0,90, sehingga paling baik dalam mendeteksi kelas positif serta paling seimbang secara keseluruhan dibanding model lainnya.

Selain itu, goals pada Section 3.2 dinyatakan tercapai karena proses yang direncanakan telah dilakukan, yaitu membangun model klasifikasi dan membandingkan beberapa algoritma melalui pengujian tiga model, evaluasi menggunakan metrik utama (accuracy, precision, recall, dan F1), lalu menentukan model terbaik berdasarkan hasil evaluasi tersebut.

8.2 Key Insights

Insight dari Data

- Data memiliki pola yang bisa dipelajari model (terlihat dari metrik yang cukup baik).
- Ada kemungkinan overlap antar kelas sehingga beberapa sampel sulit diprediksi.
- Ketidakseimbangan/karakteristik kelas membuat model lebih fokus menaikkan recall (lebih "berani" memprediksi positif).

Insight dari Modeling

- Model yang lebih kompleks (RF, MLP) memberi performa lebih baik daripada model linear (LR).
- MLP unggul pada recall dan F1, namun butuh waktu training paling lama.

8.3 Kontribusi Proyek

Manfaat Praktis

Proyek ini dapat digunakan sebagai contoh penerapan data science dalam bidang kesehatan, khususnya untuk klasifikasi gangguan hati berbasis data laboratorium. Model yang dihasilkan dapat berfungsi sebagai alat bantu skrining awal untuk mendukung analisis data medis, meskipun tidak menggantikan diagnosis profesional.

Pembelajaran yang Didapat

Melalui proyek ini, diperoleh pemahaman menyeluruh mengenai proses data science end-to-end, mulai dari data understanding, preprocessing, pemodelan machine learning dan deep learning, hingga evaluasi dan interpretasi hasil. Selain itu, proyek ini memberikan pengalaman dalam membandingkan performa dan efisiensi berbagai jenis model pada data tabular nyata.

9. FUTURE WORK

- Mengumpulkan lebih banyak data
  Dataset Liver Disorders memiliki jumlah data yang relatif kecil, sehingga penambahan data dapat membantu meningkatkan kemampuan generalisasi model.
- Menambah variasi data

Variasi data dari latar belakang pasien yang lebih beragam dapat membuat model lebih robust terhadap berbagai kondisi.

- Feature engineering lebih lanjut
  Pengembangan fitur baru, seperti rasio antar enzim hati atau transformasi fitur tertentu, berpotensi meningkatkan performa model.

10. REPRODUCIBILITY

10.1 GitHub Repository

Link Repository: https://github.com/dzav97/UAS_DataScience-LD.git
Repository ini mencakup:

- Notebook Jupyter (.ipynb)
- File Dataset (Exasens.csv)
- File requirements.txt
- README.md

10.2 Environment & Dependencies

a. Python Version: 3.10
b. Main Libraries:
   - numpy
   - pandas
   - scikit-learn
   - matplotlib
   - seaborn
   - tensorflow (Keras)