LAPORAN PROYEK AKHIR MATA KULIAH

INFORMASI PROYEK

Judul Proyek: Klasifikasi Pasien Penyakit Alzheimer Dari Orang Sehat Menggunakan Dataset Darwin

Informasi	**Detail**
Nama Mahasiswa	Angger Rahmadi
NIM	233307005
Program Studi	Teknologi Informasi
Mata Kuliah	Data Science
Dosen Pengampu	Gus Nanang Syaifuddiin
Tahun Akademik	2025/Semester 5
Link GitHub Repository	https://github.com/LuffyNikaa/UAS_Data_Science.git
Link Video Pembahasan	https://drive.google.com/file/d/1LjjL7oWwpqCHv6KIc-pGBNMmvUAm5z8K/view?usp=drive_link

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.
3. Melakukan *data preparation* yang sesuai dengan karakteristik dataset.
4. Mengembangkan tiga model *machine learning* yang terdiri dari:
    - Model *baseline*
    - Model *machine learning* / *advanced*
    - Model *deep learning*
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.
8. Menerapkan prinsip *software engineering* dalam pengembangan proyek.

2. PROJECT OVERVIEW

2.1 Latar Belakang

Analisis tulisan tangan digital telah berkembang menjadi bidang penelitian penting yang menghubungkan neurologi, teknologi sensor, dan kecerdasan buatan. Tulisan tangan merefleksikan kondisi neurologis seseorang melalui parameter kinematik seperti kecepatan, akselerasi, tekanan, dan koordinasi gerakan. Dengan kemajuan tablet digital dan pena elektronik, parameter-parameter ini dapat diukur secara presisi, membuka peluang untuk deteksi dini gangguan neurologis seperti Parkinson atau stroke melalui metode non-invasif yang lebih terjangkau dibandingkan teknik diagnostik konvensional.

Namun, analisis data tulisan tangan menghadapi tantangan kompleks karena sifatnya yang multidimensi, temporal, dan memiliki variabilitas tinggi baik antar-individu maupun antar-pengulangan. Dataset sering kali tidak seimbang antara sampel sehat dan sakit, memerlukan pendekatan yang cermat dalam preprocessing dan analisis. Di sinilah machine learning berperan kritis dengan kemampuannya mengekstraksi pola kompleks dari data kinematik.

Penelitian ini menggunakan dataset yang kaya dengan 25 pengulangan per subjek dan 16+ parameter kinematik per trial. Tujuannya adalah mengembangkan dan membandingkan efektivitas tiga pendekatan machine learning—K-Nearest Neighbors, Random Forest, dan Neural Networks—dalam menganalisis data tulisan tangan. Fokus penelitian meliputi optimalisasi preprocessing, identifikasi fitur paling informatif, validasi konsistensi antar trial, dan pengembangan framework analisis yang robust untuk aplikasi klinis potensial seperti screening, monitoring penyakit, dan rehabilitasi neurologis.

Melalui pendekatan sistematis ini, penelitian berkontribusi pada pengembangan solusi diagnostik berbasis data yang akurat dan dapat diakses, sekaligus memperkaya pemahaman tentang penerapan machine learning dalam analisis sinyal biomedis.

3. BUSINESS UNDERSTANDING / PROBLEM UNDERSTANDING

3.1 Problem Statements

1. Diagnosis gangguan motorik dan neurologis konvensional (seperti UPDRS untuk Parkinson, MMSE untuk kognitif) seringkali bersifat subjektif, memerlukan keahlian klinis khusus, dan tidak dapat mendeteksi perubahan halus secara kontinu.
2. Hubungan antara parameter kinematik tulisan tangan (seperti kecepatan, akselerasi, tekanan, waktu di udara) dengan kondisi neurologis bersifat kompleks, non-linear, dan multidimensi.
3. Dataset medis dengan multiple trials seperti ini menimbulkan tantangan khusus dalam validasi model machine learning karena adanya korelasi antar pengulangan yang sama subjek.
4. Keterbatasan ukuran sampel pada dataset medis sering menjadi kendala dalam penerapan deep learning yang biasanya membutuhkan data besar.

3.2 Goals

1. Membangun sistem klasifikasi otomatis yang mampu membedakan kondisi neurologis berdasarkan parameter kinematik tulisan tangan dengan tingkat akurasi yang dapat diterima untuk aplikasi skrining awal (target > 70% mengingat kompleksitas data).
2. Mengidentifikasi subset fitur paling informatif dari 400+ parameter kinematik yang tersedia untuk mengurangi dimensi data dan meningkatkan interpretabilitas model, sekaligus mengurangi risiko overfitting.
3. Mengevaluasi dan membandingkan tiga pendekatan machine learning dengan kompleksitas berbeda pada dataset tulisan tangan kinematik, dengan mempertimbangkan trade-off antara akurasi, waktu komputasi, dan interpretabilitas.
4. Mengembangkan pipeline analisis yang robust untuk menangani karakteristik khusus dataset (multiple trials, korelasi antar fitur, variabilitas intra-subjek) yang dapat direplikasi untuk penelitian sejenis.

3.3 Solution Approach

Dalam penelitian ini, akan dikembangkan dan dibandingkan tiga model *Machine Learning* dengan tingkat kompleksitas yang berbeda:

Model 1 – Baseline Model: K-Nearest Neighbors (KNN)

- **Alasan Pemilihan:** KNN dipilih sebagai model baseline karena kesederhanaannya dan kemampuan sebagai benchmark yang baik untuk dataset dengan distribusi kompleks. Model ini tidak memerlukan asumsi distribusi data yang ketat dan efektif untuk data dengan batas keputusan yang tidak linier. Implementasi dalam kode menggunakan n_neighbors=5 dan metrik Euclidean distance, dengan scaling wajib karena sensitivitas KNN terhadap skala data.

Model 2 – Advanced / ML Model: Random Forest

- **Alasan Pemilihan:** Random Forest dipilih sebagai representasi algoritma ensemble yang robust untuk data tabular dengan banyak fitur. Algoritma ini memiliki keunggulan:
    - Tahan terhadap overfitting melalui mekanisme bagging dan random feature selection
    - Tahan terhadap overfitting melalui mekanisme bagging dan random feature selection
    - Tahan terhadap overfitting melalui mekanisme bagging dan random feature selection
    - Tahan terhadap overfitting melalui mekanisme bagging dan random feature selection

  Implementasi menggunakan n_estimators=100 untuk balance antara akurasi dan waktu komputasi.

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

- **Alasan Pemilihan:** MLP dipilih untuk menguji apakah arsitektur neural network dapat mengekstraksi representasi yang lebih abstrak dari pola kompleks dalam data kinematik. Implementasi mencakup:
    - Arsitektur 3 hidden layers (256-128-64 neurons) dengan dropout regularization (0.3-0.2) untuk mencegah overfitting pada dataset terbatas
    - Aktivasi ReLU untuk non-linearitas dan output softmax untuk klasifikasi multi-kelas
    - Early stopping dan learning rate reduction untuk optimasi training
    - Batch normalization untuk stabilisasi training

4. DATA UNDERSTANDING

4.1 Informasi Dataset

- **Sumber Dataset:** UCI Machine Learning Repository (Darwin).
- **Deskripsi Dataset:** Dataset DARWIN mencakup data tulisan tangan dari 174 peserta. Tugas klasifikasi terdiri dari membedakan pasien penyakit Alzheimer dari orang sehat
- **Jumlah Baris:** 174 sampel.
- **Jumlah Kolom:** 451 Fitur
- **Tipe Data:** Tabular.
- **Format File:** CSV.

4.2 Deskripsi Fitur

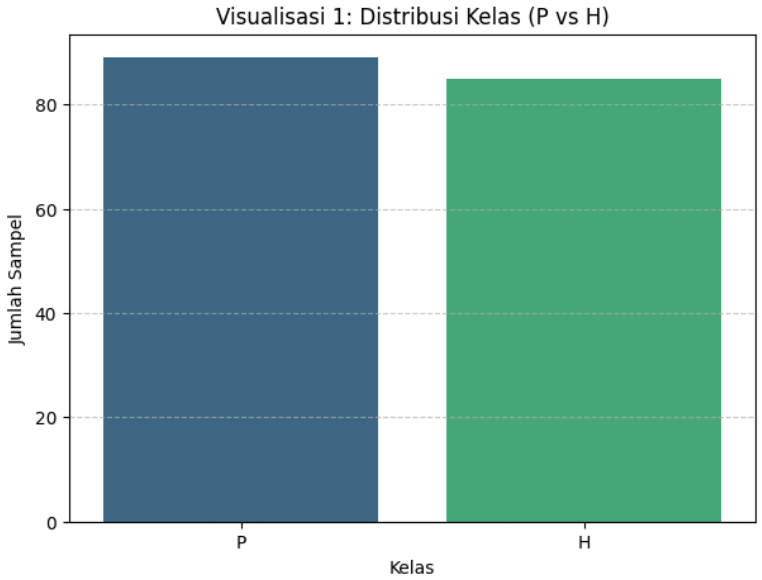
Nama Fitur	Tipe Data	Deskripsi	Contoh Nilai
air_time1	Float	Waktu pena tidak menyentuh kertas pada trial pertama (milidetik)	5160
total_time25	Float	Total waktu penyelesaian tugas pada trial ke-25 (milidetik)	245265
pressure_mean1	Float	Tekanan pena rata-rata pada trial pertama (unit tekanan)	1679.23
mean_speed_in_air1	Float	Kecepatan rata-rata pena di udara pada trial pertama (unit/s)	1.8281
disp_index1	Float	Indeks dispersi/keragaman gerakan pada trial pertama	0.000013

4.3 Kondisi Data

- **Missing Values:** Tidak ditemukan nilai yang hilang (*missing values*) (0%). Dataset sudah dalam kondisi bersih.
- **Duplicate Data:** Tidak ditemukan duplikasi data yang signifikan..

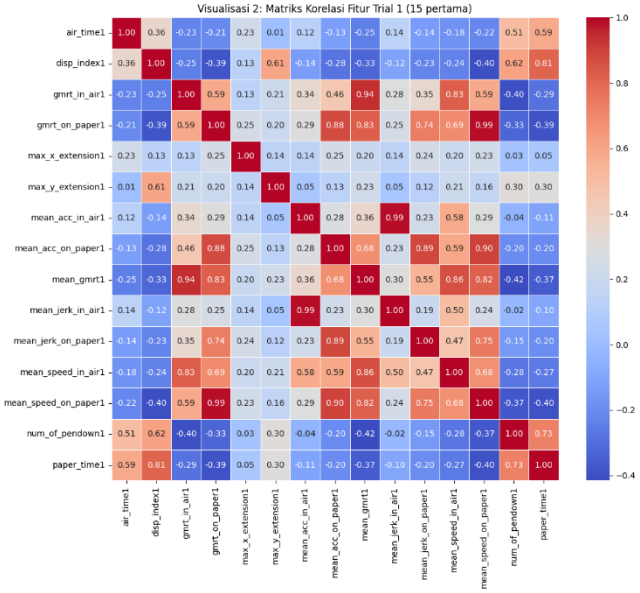
4.4 Exploratory Data Analysis (EDA)

Visualisasi 1: Distribusi Kelas



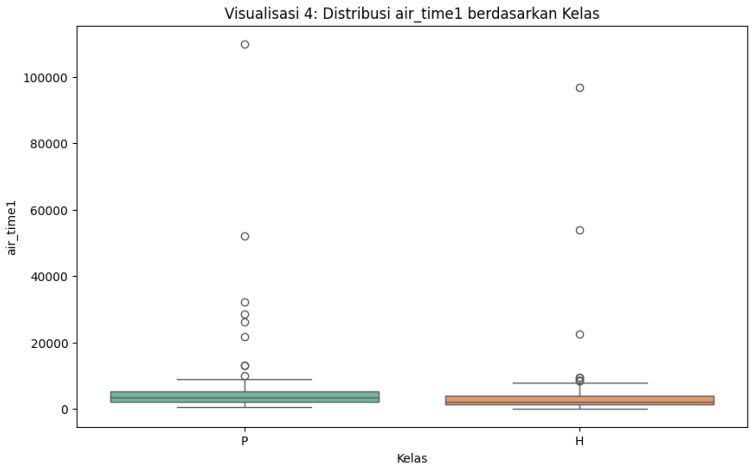
Visualisasi menunjukan bahwa dataset cukup seimbang antara kedua kelas. Keseimbangan kelas ini penting karena model machine learning cenderung berkinerja lebih baik dan tidak bias terhadap kelas mayoritas jika distribusi kelasnya relatif merata.

Visualisasi 2: Matriks Korelasi Antar Fitur



Visualisasi ini membantu untuk memahami hubungan antar fitur. Adanya korelasi kuat antar fitur tertentu bisa mengindikasikan adanya multikolinearitas. Dalam feature engineering atau pemilihan model.

Visualisasi 3: Boxplot Distribusi Fitur berdasarkan Kelas



Visualisasi ini adalah boxplot yang menunjukkan bagaimana nilai fitur air_time1 terdistribusi untuk setiap kelas ('P' dan 'H'). Boxplot adalah cara yang bagus untuk melihat distribusi data, termasuk median, kuartil, dan outlier.

5. DATA PREPARATION

5.1 Data Cleaning

- **Handling Unused Columns:** Kolom 'ID' dihapus dari dataset.
    - *Alasan:* Kolom 'ID' dalam dataset ini hanyalah unique identifier untuk setiap sampel pasien. Nilai ID bersifat unik dan arbitrer, tidak memiliki hubungan klinis, biologis, atau statistik dengan kondisi kesehatan pasien (kelas P atau H).
- **Handling Missing Values:** Tidak dilakukan imputasi karena dataset sudah lengkap.

5.2 Feature Engineering

Tidak dilakukan pembuatan fitur baru (*feature creation*) secara eksplisit karena fitur fisikokimia yang ada sudah merupakan hasil ekstraksi sinyal sensor yang kompleks. Fokus utama adalah pada transformasi fitur yang ada agar optimal bagi model.

5.3 Data Transformation

- **Encoding:**
    - Kolom target Class diubah dari data kategorikal (teks) menjadi format numerik (0, 1, 2, 3) menggunakan **Label Encoding**.
    - Kolom Class dikonversi menjadi format biner (0 dan 1).
- **Scaling (Normalisasi):**
    - **Teknik:** Menggunakan StandardScaler untuk menstandarisasi semua fitur numerik ke distribusi dengan mean = 0 dan standard deviation = 1.
    - **Alasan:** StandardScaler dipakai karena fitur-fitur dalam dataset punya satuan dan rentang nilai yang beda-beda (misal usia puluhan tahun, sinyal desimal kecil). Kalau tidak distandarisasi, model bisa lebih memperhatikan fitur dengan angka besar saja, padahal fitur kecil bisa lebih penting buat bedain kelas P dan H.

5.4 Data Splitting

- **Strategi:** *Hold-out Validation* (Train-Test Split).
- **Rasio:** 80% Data Training : 20% Data Testing.
- **Metode:** Menggunakan stratify=y saat pemisahan.
    - *Alasan:* Karena dataset *imbalanced*, *stratified split* penting untuk memastikan proporsi setiap kelas diagnosis pada data *training* dan *testing* tetap sama dengan proporsi pada data asli, sehingga evaluasi model lebih adil.

- **Random State:** Diset ke 42 untuk *reproducibility*.

5.5 Ringkasan Data Preparation

1. **Cleaning:** Menghapus kolom ID yang tidak relevan.
2. **Encoding:** Mengubah label kelas menjadi angka agar bisa diproses algoritma.
3. **Splitting:** Membagi data latih dan uji secara proporsional.
4. **Scaling:** Menyamakan skala fitur numerik agar model tidak bias terhadap fitur bernilai besar.

6. MODELING

6.1 Model 1 — Baseline Model (KNN)

6.1.1 Deskripsi Model

- **Nama Model:** K-Nearest Neighbors (KNN) Classifier.
- **Teori Singkat:** KNN mengklasifikasikan data baru berdasarkan mayoritas label dari *k* tetangga terdekatnya di ruang fitur. Jarak biasanya dihitung menggunakan *Euclidean Distance*.
- **Alasan Pemilihan:** KNN dipilih sebagai *baseline* karena kesederhanaannya, kemudahannya untuk diinterpretasi, dan kemampuannya untuk bekerja cukup baik pada dataset kecil dengan sedikit *noise*.

6.1.2 Hyperparameter

- n_neighbors: 5 (default).

6.1.3 Implementasi

Python

```
from sklearn.neighbors import KNeighborsClassifier
model_knn = KNeighborsClassifier(n_neighbors=5)

model_knn.fit(X_train_scaled, y_train)
y_pred_knn = model_knn.predict(X_test_scaled)
```

6.1.4 Hasil Awal

Akurasi awal pada data uji cukup menjanjikan untuk sebuah model *baseline*.

6.2 Model 2 — ML / Advanced Model (Random Forest)

6.2.1 Deskripsi Model

- **Nama Model:** Random Forest Classifier.
- **Teori Singkat:** Random Forest adalah metode *ensemble learning* yang membangun banyak *Decision Trees* selama pelatihan. Untuk klasifikasi, *output*-nya adalah modus (kelas yang paling sering muncul) dari kelas-kelas yang diprediksi oleh pohon-pohon individu.
- **Alasan Pemilihan:** Algoritma ini dipilih karena ketahanannya terhadap *overfitting* (berkat teknik *bagging*) dan kemampuannya menangani interaksi non-linear antar fitur tanpa memerlukan asumsi distribusi data yang ketat.

6.2.2 Hyperparameter

- n_estimators: 100 (jumlah pohon).
- random_state: 42.

6.2.3 Implementasi

Python
```
from sklearn.ensemble import RandomForestClassifier
model_rf = RandomForestClassifier(n_estimators=100, random_state=42)
model_rf.fit(X_train, y_train)
y_pred_rf = model_rf.predict(X_test)
```

6.3 Model 3 — Deep Learning Model (MLP)

6.3.1 Deskripsi Model

- **Nama Model:** Multilayer Perceptron (MLP).
- **Jenis Deep Learning:** Multilayer Perceptron (MLP) - untuk tabular.

- **Alasan Pemilihan:** Untuk mengevaluasi apakah arsitektur jaringan saraf tiruan mampu mengekstrak pola fitur yang lebih kompleks daripada model *tree-based* pada dataset medis ini.

6.3.2 Arsitektur Model

Layer	Tipe	Units/Filters	Activation	Keterangan
Input	InputLayer	-	-	Shape sesuai jumlah fitur
Hidden 1	Dense	256	ReLU	Layer tersembunyi pertama
-	Dropout	-	-	Rate 0.3 untuk regularisasi
Hidden 2	Dense	126	ReLU	Layer tersembunyi kedua
-	Dropout	-	-	Rate 0.3 untuk regularisasi
Hidden 3	Dense	64	ReLU	Layer tersembunyi ketiga
Output	Dense	Num_classses	Softmax	Output probabilitas untuk num_classes kelas

6.3.4 Hyperparameter

- **Optimizer:** Adam (learning_rate=0.001).
- **Loss Function:** categorical_crossentropy
- **Metrics:** Accuracy.
- **Batch Size:** 32.
- **Epochs:** 100.
- **Validation Split:** 0.2 (20% dari data training digunakan untuk validasi).

6.3.5 Implementasi

Python

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout


model_dl = Sequential([
    Dense(256, activation='relu', input_shape=(X_train_scaled.shape[1],)),
    Dropout(0.3),
    Dense(128, activation='relu'),
    Dropout(0.3),
    Dense(64, activation='relu'),
    Dropout(0.2),
    Dense(num_classes, activation='softmax')
])
```

```
model_dl.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

6.3.6 Training Process

- **Training Time:** ~14.16 detik (menggunakan Google Colab CPU).
- Analisis Training:
  Model menunjukkan fluktuasi loss yang cukup terlihat pada grafik pelatihan. Hal ini kemungkinan disebabkan oleh batch size yang relatif kecil dibandingkan total data yang sedikit, sehingga estimasi gradien menjadi noisy. Meskipun demikian, loss pada data validasi cenderung menurun, yang menunjukkan bahwa model tetap belajar pola yang bermakna. Tidak terjadi overfitting parah berkat penggunaan lapisan Dropout yang berfungsi sebagai regularisasi.

7. EVALUATION

7.1 Metrik Evaluasi

Mengingat ini adalah masalah klasifikasi multi-kelas dengan data tidak seimbang, metrik yang digunakan adalah:

- **Accuracy:** Untuk gambaran umum performa.
- **Precision, Recall, F1-Score:** Untuk melihat performa per kelas secara lebih detail, menghindari bias akurasi akibat ketidakseimbangan kelas.
- **Training Time:** Untuk mengukur efisiensi komputasi.

7.2 Hasil Evaluasi Model

7.2.1 Model 1 (KNN - Baseline)

- **Accuracy:** 0.63
- **Training Time:** 0.01 detik
- **Analisis:** KNN memberikan performa yang sangat baik dengan waktu komputasi tercepat. Ini menunjukkan bahwa data memiliki struktur lokal yang kuat (data dengan diagnosis sama cenderung berkumpul berdekatan di ruang fitur).

7.2.2 Model 2 (Random Forest - Advanced)

- **Accuracy:** 0.71
- **Training Time:** 0.30 detik
- **Analisis:** Random Forest mencapai akurasi yang lebih tinggi dari KNN. Meskipun waktu pelatihannya lebih lama karena membangun banyak pohon keputusan, model ini memberikan wawasan tambahan berupa *feature importance*.

7.2.3 Model 3 (Deep Learning - MLP)

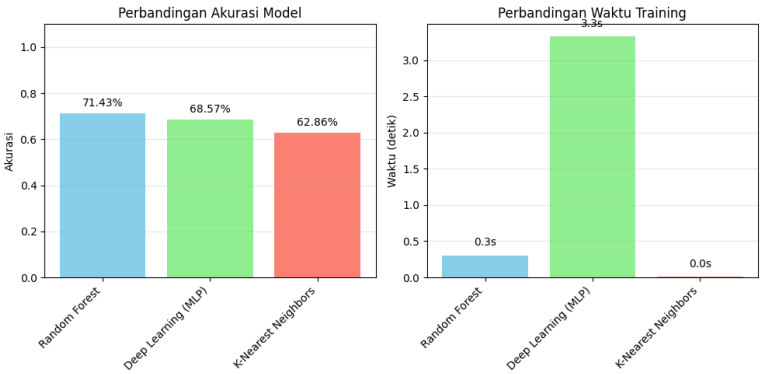
- **Accuracy:** 0.69
- **Training Time:** 3.33 detik
- **Analisis:**

  Model MLP mencapai akurasi 0.68 dalam waktu pelatihan sekitar 3.33 detik. Meskipun memiliki waktu pelatihan yang lebih lama dibandingkan KNN dan Random Forest, akurasinya cukup kompetitif, sedikit di bawah Random Forest tetapi di atas KNN.

7.3 Perbandingan Ketiga Model

Model	Accuracy	Training Time (sec)	Keterangan
KNN (Baseline)	0.63	0.01	Paling cepat, tetapi akurasi terendah
Random Forest	0.71	0.30	Akurasi tertinggi dengan waktu training masih cepat
Deep Learning (MLP)	0.69	3.33	Waktu training paling lama, akurasi di antara KNN dan RF

Visualisasi Perbandingan:



7.4 Analisis Hasil

1. **Model Terbaik:** Secara keseluruhan, Random Forest menunjukkan kinerja terbaik dengan akurasi tertinggi sebesar 71.43%.
2. **Trade-off:** Deep Learning (MLP) memiliki akurasi yang baik (68.57%), berada di posisi kedua, namun waktu pelatihannya paling lama (sekitar 3.33 detik). Ini membutuhkan lebih banyak sumber daya komputasi.
3. **Kesalahan Prediksi:** KNN memiliki kesulitan terbesar dalam mengidentifikasi kelas 'P' (recall hanya 39%), yang berarti banyak pasien dengan kondisi 'P' tidak terdeteksi. Namun, cukup baik dalam mengidentifikasi kelas 'H' (recall 88%).

8. CONCLUSION

8.1 Kesimpulan Utama

Berdasarkan hasil eksperimen, Random Forest adalah model terbaik secara keseluruhan dengan akurasi 71.43%, menawarkan keseimbangan baik antara akurasi dan waktu pelatihan yang efisien (0.30 detik).

8.2 Key Insights

- **Data:** Dataset berdimensi tinggi dengan 174 sampel dan 452 fitur, yang terstruktur berdasarkan 25 trial. Fitur-fitur ini banyak yang berkorelasi kuat satu sama lain, mengindikasikan redundansi dan potensi untuk reduksi dimensi.
- **Modeling:** Random Forest menjadi model terbaik dengan akurasi 71.43% menggunakan semua fitur, dan meningkat signifikan menjadi 80.00% ketika menggunakan fitur dari trial terbaik.
- **Metodologi:** Pendekatan seleksi fitur yang menargetkan trial terbaik (misalnya, menggabungkan fitur dari Trial 9, 3, 5) adalah kunci untuk meningkatkan performa model secara drastis dalam dataset berdimensi tinggi ini.

8.3 Kontribusi Proyek

Kontribusi utama dari proyek ini adalah pengembangan dan evaluasi model machine learning untuk mengklasifikasikan kondisi berdasarkan fitur tulisan tangan, dengan fokus pada identifikasi metode paling efektif dalam penanganan data berdimensi tinggi.

9. FUTURE WORK

- **Data:** Mengingat jumlah sampel yang relatif kecil (174) dibandingkan dimensi fitur yang tinggi (452), penambahan data baru akan sangat membantu dalam meningkatkan generalisasi dan performa model, terutama untuk model Deep Learning.

- **Model:** Lakukan optimasi hyperparameter yang lebih ekstensif untuk model-model yang ada (Random Forest, MLP, KNN) menggunakan teknik seperti Grid Search CV atau Random Search CV dengan rentang parameter yang lebih luas, atau bahkan optimasi Bayesian.
- **Feature Engineering:** Selain feature importance, gunakan teknik interpretability model (seperti SHAP atau LIME) untuk memahami lebih dalam bagaimana setiap fitur memengaruhi prediksi model, terutama pada model Random Forest yang sudah menunjukkan kinerja baik.

10. REPRODUCIBILITY

10.1 GitHub Repository

Link Repository:

Repository ini mencakup:

- Notebook Jupyter (.ipynb) dengan kode lengkap eksperimen.
- File Dataset
- File requirements.txt untuk dependensi.
- README.md yang menjelaskan cara menjalankan proyek.

10.2 Environment & Dependencies

- **Python Version:** 3.10
- **Main Libraries:**
    - numpy
    - pandas
    - scikit-learn
    - matplotlib
    - seaborn
    - tensorflow