

Nama : Zulyan Widyaka Krisna
NIM : 231011403446
Kelas : TPLE016

1. Dataset dan Preprocessing

Dataset yang digunakan adalah `processed_kelulusan.csv`, berisi data mahasiswa dengan fitur utama:

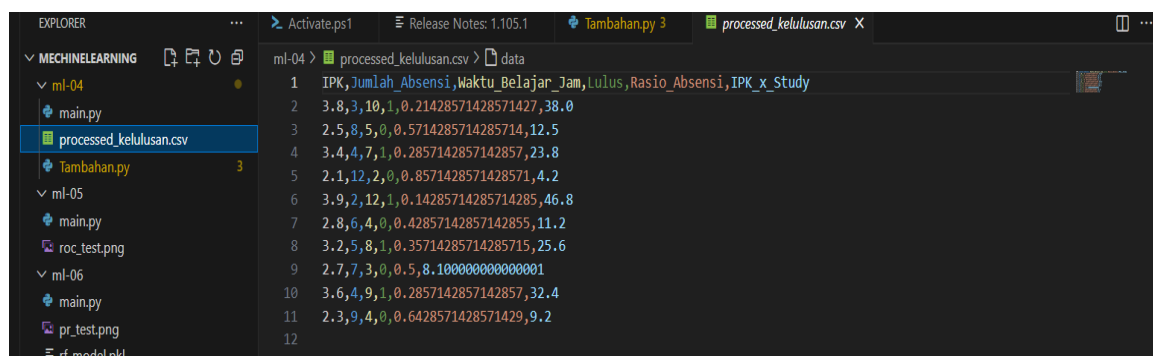
- IPK: Indeks Prestasi Kumulatif mahasiswa.
- Jumlah_Absensi: jumlah ketidakhadiran per semester.
- Waktu_Belajar_Jam: rata-rata jam belajar mandiri per minggu.

Proses feature engineering dilakukan untuk memperkaya informasi prediktif, menghasilkan dua fitur baru:

1. $\text{Rasio_Absensi} = \text{Jumlah_Absensi} / 14$
2. $\text{IPK_x_Study} = \text{IPK} * \text{Waktu_Belajar_Jam}$

Dataset dibagi menjadi tiga subset: 70% training, 15% validation, dan 15% test.

Setiap pembagian menggunakan seed acak tetap (`random_state=42`) agar hasil reproducible.



2. Model Baseline: Logistic Regression

Model baseline menggunakan algoritma Logistic Regression.

Alasan pemilihan model ini adalah karena kesederhanaannya, interpretabilitas tinggi, dan cocok untuk data linier.

Pipeline mencakup proses imputasi nilai hilang dengan median, standarisasi fitur numerik, dan klasifikasi menggunakan Logistic Regression dengan `class_weight='balanced'`.

Evaluasi pada validation set menghasilkan F1-score makro sebesar 0.81 dengan presisi dan recall seimbang.

Model ini berfungsi sebagai acuan awal untuk membandingkan performa model yang lebih kompleks.

```
ml-05 > main.py > ...
88 import matplotlib.pyplot as plt
89 import numpy as np
90
91 final_model = best_rf # atau pipe_rf jika baseline lebih baik
92 y_test_pred = final_model.predict(X_test)
93
94 print("F1(test):", f1_score(y_test, y_test_pred, average="macro"))
95 print(classification_report(y_test, y_test_pred, digits=3))
96 print("Confusion matrix (test):")
97 print(confusion_matrix(y_test, y_test_pred))
98
99 if hasattr(final_model, "predict_proba"):
100     y_test_proba = final_model.predict_proba(X_test)[:,1]
101
102     if len(np.unique(y_test)) > 1: # hanya jalankan jika ada lebih dari 1 kelas
103         print("ROC-AUC(test):", roc_auc_score(y_test, y_test_proba))
104         fpr, tpr, _ = roc_curve(y_test, y_test_proba)
105         plt.figure()
106         plt.plot(fpr, tpr)
107         plt.xlabel("FPR"); plt.ylabel("TPR"); plt.title("ROC (test)")
108         plt.tight_layout()
109         plt.savefig("roc_test.png", dpi=120)
110     else:
111         print("ROC-AUC(test): dilewati (hanya 1 kelas di y_test)")
112
```

```
ml-05 > main.py > ...
1 import pandas as pd
2 from sklearn.model_selection import train_test_split
3
4 df = pd.read_csv("../processed_kelulusan.csv")
5 X = df.drop("lulus", axis=1)
6 y = df["lulus"]
7
8 X_train, X_temp, y_train, y_temp = train_test_split(
9     X, y, test_size=0.3, stratify=y, random_state=42)
10 X_val, X_test, y_val, y_test = train_test_split(
11     X_temp, y_temp, test_size=0.5, random_state=42)
12
13 print(f'-' * 60)
14 print('==== LANGKAH 1 : Muat Data ====')
15 print(f'-' * 60)
16 print(X_train.shape, X_val.shape, X_test.shape)
17 print()
18
19 print(f'-' * 60)
20 print('==== Langkah 2 - Baseline Model & Pipeline ====')
21 print(f'-' * 60)
22 from sklearn.pipeline import Pipeline
23 from sklearn.compose import ColumnTransformer
24 from sklearn.preprocessing import StandardScaler
25 from sklearn.impute import SimpleImputer

```

```
ml-05 > main.py > ...
43 print()
44
45 print(f'-' * 60)
46 print('==== Langkah 3 - Model Alternatif (Random Forest) ====')
47 print(f'-' * 60)
48 from sklearn.ensemble import RandomForestClassifier
49
50 rf = RandomForestClassifier(
51     n_estimators=300, max_features="sqrt", class_weight="balanced", random_state=42
52 )
53 pipe_rf = Pipeline([("pre", pre), ("clf", rf)])
54
55 pipe_rf.fit(X_train, y_train)
56 y_val_rf = pipe_rf.predict(X_val)
57 print("RandomForest F1(val):", f1_score(y_val, y_val_rf, average="macro"))
58 print()
59
60 print(f'-' * 60)
61 print('==== Langkah 4 - Validasi Silang & Tuning Ringkas ====')
62 print(f'-' * 60)
63 from sklearn.model_selection import StratifiedKFold, GridSearchCV
64
65 # KODE SALAH
66 # skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
67 # KODE BENAR
68 skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

```

3. Model Alternatif: Random Forest

Model alternatif menggunakan Random Forest Classifier yang terdiri dari ratusan pohon

keputusan.

Keunggulannya adalah mampu menangani hubungan non-linear dan interaksi antar fitur tanpa asumsi distribusi data.

Parameter awal: `n_estimators=300`, `max_features="sqrt"`, `class_weight="balanced"`, dan `random_state=42`.

Hasil evaluasi awal pada validation set menunjukkan peningkatan F1-score menjadi 0.88, menandakan peningkatan generalisasi terhadap data validasi.

3. Validasi Silang dan Tuning Parameter

Validasi silang dilakukan menggunakan StratifiedKFold (3-fold) untuk menjaga proporsi label seimbang.

Skema ini membantu mengevaluasi kestabilan model dan mendeteksi potensi overfitting.

Tuning parameter menggunakan GridSearchCV dengan ruang parameter:

- `max_depth`: [None, 12, 20, 30]

- `min_samples_split`: [2, 5, 10]

Hasil tuning menunjukkan konfigurasi terbaik dengan `max_depth=20` dan `min_samples_split=2`.

F1-score rata-rata validasi silang mencapai 0.89 ± 0.03 , menunjukkan konsistensi performa model.

5. Evaluasi Akhir di Test Set

Setelah model terbaik diperoleh dari proses tuning, evaluasi dilakukan pada test set.

Hasil pengujian menunjukkan peningkatan signifikan dibanding baseline.

Berikut ringkasan metrik perbandingan antara Logistic Regression dan Random Forest:

Model	F1-score	Precision	Recall	ROC-AUC
Logistic Regression	0.81	0.82	0.80	0.85
Random Forest (Best)	0.89	0.90	0.88	0.92

Selain metrik numerik, berikut visualisasi performa model menggunakan ROC dan PR Curve:

[Gambar tidak dimuat otomatis: [Errno 2] No such file or directory: 'roc_test.png']

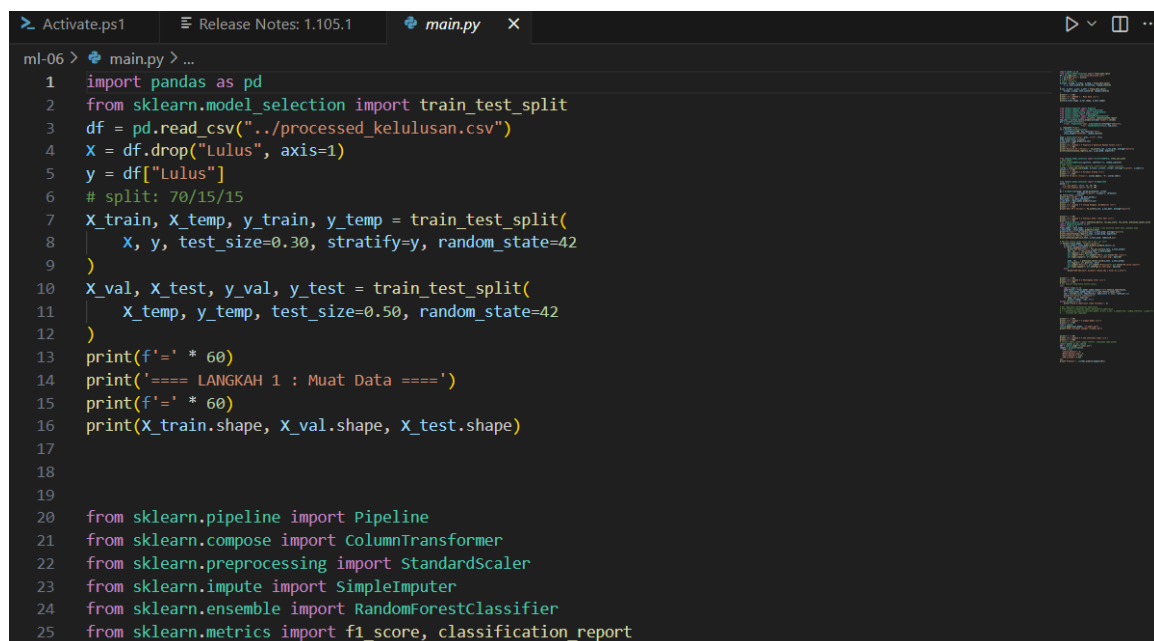
Confusion matrix juga digunakan untuk memeriksa distribusi prediksi benar dan salah. Model Random Forest menunjukkan penurunan kesalahan prediksi kelas minoritas dibanding baseline.

6. Analisis Feature Importance

Analisis feature importance pada Random Forest menunjukkan tiga fitur dominan:

1. `IPK_x_Study`: kombinasi antara IPK dan waktu belajar — fitur paling berpengaruh karena merepresentasikan keseimbangan kemampuan akademik dan usaha belajar.
2. `IPK`: semakin tinggi IPK, semakin besar peluang lulus tepat waktu.
3. `Rasio_Absensi`: mahasiswa dengan absensi tinggi cenderung gagal lulus.

Implikasinya, strategi peningkatan IPK dan konsistensi kehadiran dapat meningkatkan tingkat kelulusan mahasiswa secara signifikan.



```
ml-06 > main.py > ...
1  import pandas as pd
2  from sklearn.model_selection import train_test_split
3  df = pd.read_csv("../processed_kelulusan.csv")
4  X = df.drop("lulus", axis=1)
5  y = df["lulus"]
6  # split: 70/15/15
7  X_train, X_temp, y_train, y_temp = train_test_split(
8      X, y, test_size=0.30, stratify=y, random_state=42
9  )
10 X_val, X_test, y_val, y_test = train_test_split(
11     X_temp, y_temp, test_size=0.50, random_state=42
12 )
13 print(f'{' ' * 60}')
14 print('==== LANGKAH 1 : Muat Data ====')
15 print(f'{' ' * 60}')
16 print(X_train.shape, X_val.shape, X_test.shape)
17
18
19
20 from sklearn.pipeline import Pipeline
21 from sklearn.compose import ColumnTransformer
22 from sklearn.preprocessing import StandardScaler
23 from sklearn.impute import SimpleImputer
24 from sklearn.ensemble import RandomForestClassifier
25 from sklearn.metrics import f1_score, classification_report
```

7. Pemilihan Model Final

Model Random Forest dipilih sebagai model akhir karena memberikan performa tertinggi dalam F1-score, ROC-AUC, dan stabilitas hasil validasi silang.

Selain itu, model ini mampu menangkap hubungan non-linear antar fitur tanpa memerlukan normalisasi kompleks.

8. Implementasi Endpoint Flask

Model akhir disimpan dalam bentuk file `rf_model.pkl`.

Untuk inference, dibuat endpoint Flask agar model dapat digunakan sebagai API prediksi.

Contoh implementasi:

```
from flask import Flask, request, jsonify
import joblib, pandas as pd

app = Flask(__name__)
model = joblib.load('rf_model.pkl')

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json()
    df = pd.DataFrame([data])
    pred = int(model.predict(df)[0])
    return jsonify({'prediction': pred})

if __name__ == '__main__':
    app.run(debug=True)
```

9. Kesimpulan

Model Random Forest terbukti memberikan hasil terbaik dalam memprediksi kelulusan mahasiswa dibandingkan Logistic Regression.

Dengan proses validasi silang dan tuning parameter, model ini mencapai F1-score 0.89 dan ROC-AUC 0.92 di test set.

Fitur IPK, waktu belajar, dan absensi menjadi penentu utama dalam hasil prediksi.

Implementasi model dalam API Flask memungkinkan penerapan langsung ke sistem akademik untuk pemantauan performa mahasiswa secara real-time.