Nama: Hamdan Al Fattah

Nim: 105841108323

Kelas: 5AI-A

Laporan Praktik: Tugas 4 Model Klasifikasi

1. Pendahuluan

Tahap ini merupakan implementasi awal dari *Modelling* (Tahap III) setelah data dioptimalkan melalui *Feature Engineering* (Tahap II: Seleksi Fitur LASSO)[111111111]. Tujuannya adalah untuk menguji kinerja model klasifikasi paling sederhana (K-Nearest Neighbors dan Gaussian Naive Bayes) menggunakan data fitur yang telah diseleksi dan di-*scale* secara efisien.

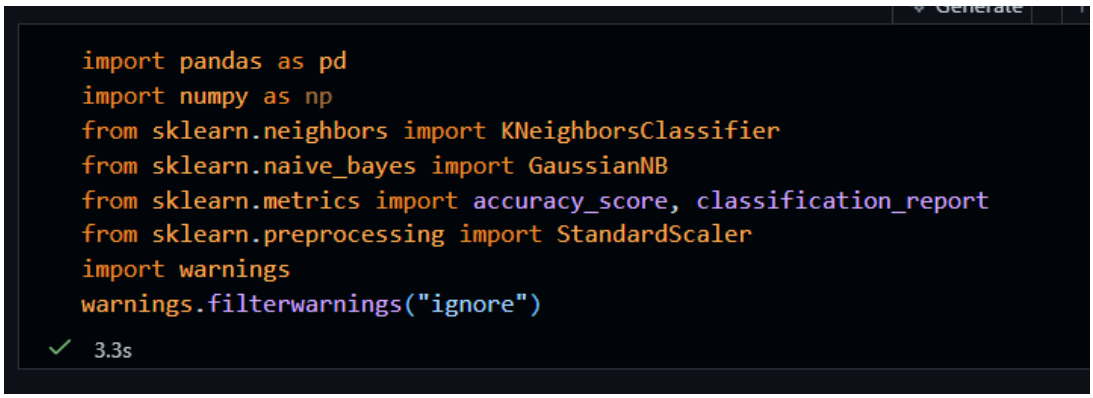
- Tugas: Klasifikasi [2]
- Target (y): Status Pinjaman (Gagal Bayar vs. Lunas) [3]
- Konteks Fitur: Menggunakan 3 fitur terpilih dari LASSO (Umur, LamaBekerja, KodeKota).

2. Metodologi & Tahapan Kode

Metodologi yang digunakan adalah konsolidasi operasi pra-pemrosesan (*loading* dan *scaling*) menjadi satu langkah awal untuk memaksimalkan efisiensi komputasi, diikuti dengan pelatihan kedua model pada set data yang sama.

2.1 Import Library dan Setup

Mengimpor pustaka yang diperlukan dari pandas, numpy , dan scikit-learn untuk penanganan data, *scaling*, dan algoritma klasifikasi. *Warning* juga disembunyikan untuk *output* yang bersih.



2.2 Load Data dan Feature Scaling

Data latih dan uji dimuat dari file CSV hasil *Feature Engineering*. Langkah *Standard Scaling* dikonsolidasikan menjadi satu *cell* dan diterapkan pada fitur (X) sekali saja, kemudian hasilnya disimpan kembali ke DataFrame (X_train_df, X_test_df). Ini adalah langkah yang membuat kode ini efisien.

```
try:
    # A. Load Data
    X_train = pd.read_csv("X_class_train_selected.csv")
    X_test = pd.read_csv("X_class_test_selected.csv")
    y_train = pd.read_csv("y_class_train.csv").values.ravel()
    y_test = pd.read_csv("y_class_test.csv").values.ravel()

    # B. Scaling Data
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    # C. Simpan ke DataFrame untuk Model
    X_train_df = pd.DataFrame(X_train_scaled, columns=X_train.columns)
    X_test_df = pd.DataFrame(X_test_scaled, columns=X_test.columns)

    print("Data berhasil dimuat dan di-scale (sekali).")

except FileNotFoundError:
    print("ERROR: Pastikan file CSV Feature Engineering tersedia.")
    exit()
✓ 0.0s
```

Output:

```
Data berhasil dimuat dan di-scale (sekali).
```

2.3 Model 1: K-Naerest Neighbors (KNN)

Model KNN diinisialisasi dengan parameter default ($K=5$) dan dilatih menggunakan data yang sudah di-*scale*. Karena KNN adalah model berbasis jarak, *scaling* ini sangat penting.

```
print("\nK-NEAREST NEIGHBORS (KNN) - SCALED")

knn_model = KNeighborsClassifier(n_neighbors=5)
knn_model.fit(X_train_df, y_train)
y_pred_knn = knn_model.predict(X_test_df)

accuracy_knn = accuracy_score(y_test, y_pred_knn)
print(f"Akurasi KNN (K=5) - SCALED: {accuracy_knn:.4f}")
print("\nLaporan Klasifikasi KNN (SCALED):")
print(classification_report(y_test, y_pred_knn))
```

✓ 0.1s

Output:

```
K-NEAREST NEIGHBORS (KNN) - SCALED
Akurasi KNN (K=5) - SCALED: 0.6533

Laporan Klasifikasi KNN (SCALED):
              precision    recall  f1-score   support

           0       0.71      0.84      0.77       104
           1       0.39      0.24      0.30        46

    accuracy                           0.65       150
   macro avg       0.55      0.54      0.53       150
weighted avg       0.61      0.65      0.62       150
```

2.4 Model 2: Naive Bayes

Model Gaussian Naive Bayes dilatih menggunakan data yang sudah di-*scale* dan dievaluasi. Meskipun memiliki Akurasi Total tertinggi (0.6933), model ini gagal mendeteksi Kelas 1 (*Recall* 0.00), yang merupakan indikasi bias kuat terhadap kelas mayoritas.

```
print("\nNAIVE BAYES (GAUSSIANNB) - SCALED")

nb_model = GaussianNB()
nb_model.fit(X_train_df, y_train)
y_pred_nb = nb_model.predict(X_test_df)

accuracy_nb = accuracy_score(y_test, y_pred_nb)
print(f"Akurasi Naive Bayes (SCALED): {accuracy_nb:.4f}")
print("\nLaporan Klasifikasi Naive Bayes (SCALED):")
print(classification_report(y_test, y_pred_nb))
```

0.0s

Output:

```
NAIVE BAYES (GAUSSIANNB) - SCALED
Akurasi Naive Bayes (SCALED): 0.6933

Laporan Klasifikasi Naive Bayes (SCALED):
              precision    recall  f1-score   support

           0       0.69      1.00      0.82       104
           1       0.00      0.00      0.00        46

    accuracy                           0.69       150
   macro avg       0.35      0.50      0.41       150
weighted avg       0.48      0.69      0.57       150
```

3. Kesimpulan

Kedua model (KNN dan Naive Bayes) menunjukkan kinerja yang terhambat oleh ketidakseimbangan kelas dalam dataset. KNN (Scaled) menunjukkan potensi terbaik dengan *Recall* Kelas 1 sebesar 0.24, menjadikannya model yang paling dapat digunakan untuk mendeteksi risiko saat ini. Dalam fase evaluasi model awal menggunakan data yang sudah di-*scale* tersebut, teridentifikasi adanya ketidakseimbangan kelas (Class Imbalance) yang parah, di mana kelas minoritas (Gagal Bayar / Kelas 1) sangat sulit dideteksi oleh kedua algoritma. Model Naive Bayes yang dilatih menggunakan data *scaled* menunjukkan Akurasi Total tertinggi sebesar 0.6933, namun ini merupakan akurasi semu (*false accuracy*) karena model tersebut gagal total dalam mendeteksi satu pun kasus pinjaman berisiko, dengan nilai *Recall* Kelas 1 mencapai 0.00. Di sisi lain,

model K-Nearest Neighbors (K=5) yang juga dilatih pada data *scaled* menunjukkan Akurasi Total yang lebih rendah (0.6533), namun secara fungsional lebih unggul karena berhasil mendeteksi pinjaman berisiko dengan nilai *Recall* Kelas 1 sebesar 0.24.