Project name: Submission [Klasifikasi] Submission Akhir BMLP_Muhammad Fikry Rizal.

Project Summary

- Pipeline lengkap dari data loading, preparation, split data, training 5 model berbeda (KNN, Decision Tree, Random Forest, SVM, Naive Bayes), evaluasi dengan berbagai metrik dan visualisasi confusion matrix.
- Implementasi komparasi 5 model klasifikasi untuk memberikan perbandingan performa yang komprehensif.
- Penggunaan heatmap confusion matrix untuk setiap model memudahkan interpretasi hasil.
- Kode perlu dimodularisasi dan dioptimasi dengan menghapus unused imports dan membuat fungsi-fungsi terpisah.
- Belum ada implementasi preprocessing data yang dapat mempengaruhi performa model.

Error Notes

Pada project yang diperiksa terjadi sebuah error saat ada potensi data leakage karena tidak ada preprocessing dan validasi dan Kolom cluster perlu divalidasi keberadaannya sebelum splitting dan saya mengatasinya dengan cara membuat preprosesing dan validasi data dan kolom clusteringnya sebelum daa dipisah atau spliting

Code Review

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler, MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
```

Sebaiknya perlu menghapus import yang tidak digunakan (LabelEncoder, StandardScaler, MinMaxScaler) lalu Kelompokkan import berdasarkan kategori agar kode lebih rapih dan mudah dimengerti

```
X = data.drop(columns=['cluster'])
y = data['cluster']

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

print(f"Training set shape: X_train={X_train.shape}, y_train={y_train.shape}")
print(f"Test set shape: X_test={X_test.shape}, y_test={y_test.shape}")
```

Perlu untuk menambahkan preprocessing (scaling/normalization) serta Implementasi stratified split untuk

imbalanced data dan Cek distribusi kelas

```
knn = KNeighborsClassifier().fit(X_train, y_train)
dt = DecisionTreeClassifier().fit(X_train, y_train)
rf = RandomForestClassifier().fit(X_train, y_train)
svm = SVC().fit(X_train, y_train)
nb = GaussianNB().fit(X_train, y_train)
```

Sebaiknya dibuat fungsi training terpisah dan Tambahkan hyperparameter tuning serta Implementasi cross-validation

```
ef evaluate_model(model, X_test, y_test):
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    results = {
        'Confusion Matrix': cm,
        'Accuracy': accuracy_score(y_test, y_pred),
        'Precision': precision_score(y_test, y_pred, average='weighted'),
        'Recall': recall_score(y_test, y_pred, average='weighted'),
        'F1-Score': f1_score(y_test, y_pred, average='weighted')
    }
    return results

results = {
    'K-Nearest Neighbors (KNN)': evaluate_model(knn, X_test, y_test),
    'Decision Tree (DT)': evaluate_model(dt, X_test, y_test),
    'Random Forest (RF)': evaluate_model(rf, X_test, y_test),
    'Support Vector Machine (SVM)': evaluate_model(svm, X_test, y_test),
    'Naive Bayes (NB)': evaluate_model(nb, X_test, y_test)
}
```

Perlu untuk menambahkan error handling, metrics lain (ROC/AUC) dan Implementasi logging hasil evaluasi

```
summary_df = pd.DataFrame(columns=['Model', 'Accuracy', 'Precision', 'Recall', 'F1-Score'])
rows = []
for model_name, metrics in results.items():
    rows.append({
        'Model': model_name,
        'Accuracy': metrics['Accuracy'],
        'Precision': metrics['Precision'],
        'Recall': metrics['Recall'],
        'F1-Score': metrics['F1-Score']
    })
summary_df = pd.DataFrame(rows)


models = [knn, dt, rf, svm, nb]
models_name = ['K-Nearest Neighbors (KNN)', 'Decision Tree (DT)', 'Random Forest (RF)',
               'Support Vector Machine (SVM)', 'Naive Bayes (NB)']

for i, model in enumerate(models):
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
                xticklabels=model.classes_, yticklabels=model.classes_)
    plt.title(f'Confusion Matrix for {models_name[i]}')
    plt.xlabel('Predicted Labels')
```

```
    plt.ylabel('True Labels')
    plt.show()
```

Sebaiknya dibuat fungsi visualisasi terpisah dan juga tambahkan normalisasi pada confusion matrix

Rekomendasi:

- Terapkan Normalisasi/Standarisasi sebelum training
- Implementasi Cross-Validation
- Tambahkan Hyperparameter Tuning dengan GridSearchCV
- Modularisasi kode dengan fungsi-fungsi terpisah
- Explorasi model lain (XGBoost, LightGBM) untuk meningkatkan performa

Kesimpulan Akhir:
Proyek ini sudah mencakup pipeline machine learning classification yang lengkap, namun masih memerlukan peningkatan dalam hal preprocessing, error handling, modularisasi kode, dan optimasi parameter model untuk hasil dan performa yang lebih baik