Project name: [Klasifikasi]_Submission_Akhir_BMLP_Fiko_Rahardito_Baskoro

Project Summary

- Data Preparation: Import library, cek data, hapus/abaikan kolom tidak terpakai, lalu One-Hot Encoding pada kolom Marital_Status.
- Data Splitting: Membagi dataset menjadi latih dan uji, meskipun ada beberapa kali split (80:20, lalu 70:30) di kode.
- Model *K-Nearest Neighbors (KNN)* untuk klasifikasi.
- Visualisasi *decision boundary* setelah reduksi dimensi dengan **PCA** (2 komponen).
- Evaluasi dengan metrik **Accuracy, Precision, Recall, F1-Score**, dan **Confusion Matrix**.
- Hyperparameter Tuning: Menggunakan GridSearchCV untuk mencari kombinasi terbaik dari n_neighbors, weights, dan metric.
- Final Model: Model terbaik di-fit ulang, lalu hasilnya dibandingkan sebelum dan sesudah tuning.
- Meskipun pada kode ini hanya satu model utama (KNN), menambahkan model lain (misalnya *Random Forest* atau *Logistic Regression*) dapat memberikan perbandingan performa.
- Menggunakan *scatter plot* dan *contourf* area keputusan (*decision boundary*) untuk memudahkan interpretasi hasil klasifikasi di ruang 2 dimensi (PCA).
  Confusion Matrix juga divisualisasikan dengan ConfusionMatrixDisplay.
- Secara umum, kode masih terpadu dalam satu *notebook*. Pembuatan fungsi-fungsi terpisah dapat menurunkan duplikasi, terutama pada bagian *train_test_split* dan PCA yang dilakukan berulang.

Error Notes

Pada project yang diperiksa terjadi sebuah error saat penggunaan beberapa kali *train_test_split* (80:20 dan 70:30) menimbulkan inkonsistensi dalam evaluasi. dan saya mengatasinya dengan cara menggunakan *Pipeline* agar data latih dan data uji selalu konsisten.

Code Review

```
# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

pd.set_option('display.max_columns', None)

file_path = "/content/marketing_campaign.csv"
data = pd.read_csv(file_path)  # read_csv default is comma-delimited
print("Dataset Loaded Successfully!")
print(data.head())
```

RandomForestClassifier dan StandardScaler tidak dipakai selanjutnya. Anda bisa menghapusnya agar kode lebih rapi. seaborn as sns belum benar-benar dipakai untuk visualisasi (opsional). Dan terakhir pastikan format CSV Anda benar (apakah benar *comma-delimited* atau *tab-delimited*).

```
import pandas as pd

clustering_result_file = "/content/marketing_campaign.csv"
df_clustering = pd.read_csv(clustering_result_file)
print("Clustering Result Dataset Loaded Successfully!")
print(df_clustering.head())

data = pd.read_csv(file_path, sep='\t')
print(data.columns)

missing_values = data.isnull().sum()
missing_values[missing_values > 0]

data_encoded = pd.get_dummies(data, columns=['Marital_Status'], prefix='Marital')
print(data_encoded.head())
```

Pada bagian df_clustering tidak dimanfaatkan di kode selanjutnya. Bila tidak diperlukan, lebih baik dihapus. Dan untuk Membaca ulang data dengan sep='\t' dapat menimbulkan inkonsistensi jika file tidak benar-benar tab-delimited.

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report

k = 5
knn_model = KNeighborsClassifier(n_neighbors=k)
knn_model.fit(X_train, y_train)
y_pred = knn_model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

Pada bagian Normalisasi data (mis. StandardScaler) belum dilakukan, padahal KNN sensitif terhadap skala fitur.

```
from sklearn.decomposition import PCA
from matplotlib.colors import ListedColormap

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
X_train_pca, X_test_pca, y_train, y_test = train_test_split(X_pca, y,
                                                    test_size=0.3,
                                                    random_state=42)

knn_model = KNeighborsClassifier(n_neighbors=5)
knn_model.fit(X_train_pca, y_train)

plot_decision_boundaries(X_train_pca, y_train, knn_model,
                         "Decision Boundaries (Training Data)")
plot_decision_boundaries(X_test_pca, y_test, knn_model,
                         "Decision Boundaries (Test Data)")
```

Terjadi *split data* kedua (70:30), berbeda dengan 80:20 di atas. Ini membuat hasil evaluasi tidak konsisten.

Rekomendasi

1. Gunakan satu kali train_test_split atau *cross-validation* (CV) agar evaluasi lebih objektif.
2. Satukan PCA dan KNN dalam Pipeline atau make_pipeline. Kemudian lakukan GridSearchCV pada pipeline tersebut agar transformasi dan model selalu sinkron.
3. KNN sensitif terhadap skala fitur. Sebaiknya lakukan *MinMaxScaler* atau *StandardScaler* sebelum KNN.
4. Beberapa import dan variabel seperti df_clustering dapat dihilangkan untuk merapikan kode.
5. Jika dataset besar, bisa ditambah *progress bar* (tqdm) atau *logging* untuk memantau proses training/tuning.

Kesimpulan Akhir

Proyek ini sudah mempraktikkan tahapan penting *machine learning classification* (preprocessing, split, model training, tuning, hingga evaluasi). Metrik akurasi, precision, recall, dan F1-Score cukup lengkap. Namun, perlu diperbaiki dari segi *data consistency* (satu kali *split*), normalisasi fitur, dan penghapusan kode tidak relevan agar analisis makin optimal.