Tugas 6 Praktikum Mandiri

SYAHRI GHIFARI MAULIDI 0110222217

[1] Teknik Informatika, STT Terpadu Nurul Fikri, Depok

1.1 Import Library dan Membaca Dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.feature_extraction.text import TfidfVectorizer
import seaborn as sns
from sklearn.decomposition import PCA

# 1) Load data (samakan path ke file kamu)
csv_path = "../data/ramen-ratings.csv"
df = pd.read_csv(csv_path)
```

Dataset yang digunakan adalah **ramen-ratings.csv**. Dataset ini berisi informasi penilaian berbagai produk mie instan dari berbagai negara. Beberapa kolom penting yang digunakan:

- **Variety** – nama atau varian produk ramen
- **Style** – tipe kemasan (Cup, Pack, Bowl, dll.)
- **Country** – asal negara produk
- **Stars** – nilai rating rasa ramen (skala 0–5)
- **Brand** – nama merek (menjadi target klasifikasi*)*

Sebelum dilakukan pemodelan, dataset dibatasi pada **5 brand terpopuler** agar distribusi data lebih seimbang dan hasil klasifikasi mudah divisualisasikan.

```
display(df.head())
display(df.describe(include='all').T)
```

Langkah-langkah persiapan data meliputi:

- Membaca file ramen-ratings.csv
- Menampilkan ringkasan (head() dan describe())
- Menentukan fitur yang akan digunakan (Variety, Style, Country, Stars)
- Menetapkan kolom **Brand** sebagai *target*
- Menghapus baris yang memiliki nilai kosong
- Menyaring hanya **5 brand terbanyak** agar model terfokus pada data yang cuku

```
possible_cols = df.columns.tolist()
text_col = "Variety" if "Variety" in possible_cols else None
num_cols  = [c for c in ["Stars"] if c in possible_cols]
cat_cols  = [c for c in ["Style", "Country", "Top Ten"] if c in possible_cols]

feature_cols = []
if text_col: feature_cols.append(text_col)
feature_cols += num_cols + cat_cols

missing_needed = {"Variety","Stars","Style","Country"} - set(possible_cols)
print(f"\nKolom fitur yang dipakai: {feature_cols}")
if missing_needed:
    print(f"Catatan: Beberapa kolom umum tidak ada di CSV ini: {missing_needed}. Tidak masalah, lanjut pakai yang tersedia.")
```

1.2 Pembentukan Pipeline

Pipeline terdiri dari beberapa tahapan preprocessing:

1. **Teks (Variety)** → TfidfVectorizer

   Mengubah teks nama produk menjadi fitur numerik berbasis frekuensi kata.

2. **Numerik (Stars)** → SimpleImputer (median) + StandardScaler

   Menangani nilai hilang dan menstandarkan skala.

3. **Kategorik (Style, Country)** → OneHotEncoder

   Mengubah kategori menjadi representasi biner.

Seluruh proses ini digabung menggunakan **ColumnTransformer**, kemudian dihubungkan ke model **SVC(kernel='linear')** di dalam **Pipeline**.

1.3 Menyaring dataset

```
TOP_K = 5  # ubah ke None bila ingin semua brand
df = df.copy()
df = df[df["Brand"].notna()]
if TOP_K:
    top_brands = df["Brand"].value_counts().nlargest(TOP_K).index
    df = df[df["Brand"].isin(top_brands)]

print("Distribusi kelas (Brand)")
print(df["Brand"].value_counts())
```

Bagian kode ini berfungsi menyaring dataset agar hanya berisi 5 brand dengan jumlah data terbanyak. Hal ini dilakukan untuk menyeimbangkan distribusi kelas dan memudahkan proses klasifikasi menggunakan algoritma SVM..

2. Memisahkan fitur (X) dan target (y)

```
X = df[feature_cols].copy()
y = df["Brand"].astype("category")
class_names = list(y.cat.categories)
y = y.cat.codes
```

Kode ini digunakan untuk memisahkan fitur dan target, serta mengonversi label kategori ‘Brand’ menjadi bentuk numerik menggunakan astype('category') dan cat.codes. Hal ini dilakukan karena model SVM hanya dapat memproses data numerik

2.1 Inisialisasi list

```
transformers = []
if text_col:
    transformers.append(
        ("text", TfidfVectorizer(min_df=2, ngram_range=(1,2)), text_col)
    )

if num_cols:
    transformers.append(
        ("num", Pipeline([
            ("imputer", SimpleImputer(strategy="median")),
            ("scaler", StandardScaler())
        ]), num_cols)
    )

if cat_cols:
    transformers.append(
        ("cat", Pipeline([
            ("imputer", SimpleImputer(strategy="most_frequent")),
            ("ohe", OneHotEncoder(handle_unknown="ignore"))
        ]), cat_cols)
    )

preprocess = ColumnTransformer(transformers=transformers, remainder="drop")

model = Pipeline(steps=[
    ("prep", preprocess),
    ("clf", SVC(kernel="linear", random_state=42))  # linear seperti di slide
])
```

Bagian ini membangun pipeline preprocessing yang menyesuaikan perlakuan setiap jenis fitur. Data teks diubah dengan TF-IDF, data numerik dinormalisasi menggunakan StandardScaler, dan data kategorik diubah menjadi representasi biner menggunakan OneHotEncoder. Seluruh proses digabung dalam ColumnTransformer, lalu dihubungkan ke model SVM ber-kernel linear untuk membentuk pipeline klasifikasi yang utuh dan efisien.

2.2Membagi data latih dan data uji

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

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
acc = accuracy_score(y_test, y_pred)
print(f"Akurasi (test): {acc*100:.2f}%")
print("===== Classification Report =====")
print(classification_report(y_test, y_pred, target_names=class_names))
```

Kode ini berfungsi untuk melatih dan menguji model SVM. Dataset dibagi 80% untuk pelatihan dan 20% untuk pengujian menggunakan metode stratified split. Model dilatih menggunakan data latih, kemudian diuji pada data uji untuk menghitung akurasi serta menghasilkan laporan klasifikasi yang mencakup precision, recall, dan f1-score untuk setiap brand..

2.3 Membuat confusion matrix

```
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.title("Confusion Matrix - SVM (Brand)")
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")
plt.xticks(rotation=45, ha="right")
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```

Kode ini digunakan untuk menampilkan *Confusion Matrix* hasil klasifikasi menggunakan SVM. Heatmap memperlihatkan distribusi prediksi benar dan salah untuk setiap brand. Warna biru gelap di diagonal menunjukkan prediksi benar, sedangkan warna biru muda di luar diagonal menandakan kesalahan klasifikasi antar brand.