TUGAS STUDI KASUS PEMBELAJARAN MESIN

Clustering dan Approximate Nearest Neighbor (ANN)

Deskripsi Umum:

Tugas kali ini adalah mengerjakan **studi kasus analisis data dan clustering** menggunakan *unsupervised learning* dengan langkah-langkah sebagai berikut:

1. **Preprocessing data**
    - Tangani *missing values* (imputasi mean/median/modus sesuai jenis data)
    - Normalisasi atau standarisasi data
    - Buat minimal satu fitur baru hasil kombinasi fitur lama
2. **Clustering**
    - Terapkan **K-Means** dan **DBSCAN**
    - Bandingkan hasil clustering menggunakan:
        - **Silhouette Score**
        - **Davies–Bouldin Index**
3. **Approximate Nearest Neighbor (ANN)**
    - Gunakan **Annoy** untuk mencari tetangga terdekat dari beberapa *query points* hasil clustering
    - Tampilkan *output* berupa:
        - Index *query point*
        - Daftar tetangga terdekat yang ditemukan
        - Nilai jarak kemiripan

Tugas 1 — House Prices Dataset

Untuk mahasiswa dengan nomor absen 1, 4, 7, dst.

- **Dataset:** House Prices - Advanced Regression Techniques
- **Deskripsi:** Dataset ini berisi atribut rumah (luas, tipe bangunan, kondisi, lokasi, dsb.) yang dapat digunakan untuk eksplorasi fitur, penanganan missing values, dan clustering rumah dengan karakteristik mirip.
- **Langkah tambahan:**
    - Fokus pada subset fitur numerik terlebih dahulu.
    - Coba buat fitur baru seperti “TotalArea = GrLivArea + TotalBsmtSF”.

Tugas 2 — Credit Card Dataset

Untuk mahasiswa dengan nomor absen 2, 5, 8, dst.

- **Dataset:** Credit Card Dataset for Clustering
- **Deskripsi:** Dataset ini berisi data perilaku pengguna kartu kredit. Gunakan untuk menemukan kelompok pelanggan berdasarkan pola penggunaan.
- **Langkah tambahan:**
    - Tangani missing values dan normalisasi fitur numerik.
    - Buat fitur baru seperti rasio antara *BALANCE* dan *PURCHASES*.

Tugas 3 — Heart Disease Dataset

Untuk mahasiswa dengan nomor absen 3, 6, 10, dst.

- **Dataset:** Heart Disease Dataset (UCI)
- **Deskripsi:** Dataset medis untuk melihat pengelompokan pasien berdasarkan fitur kesehatan seperti tekanan darah, kolesterol, umur, dan lain-lain.
- **Langkah tambahan:**
    - Tangani nilai kosong (jika ada).
    - Buat fitur gabungan seperti "CholAge = cholesterol × age".

Output yang Diharapkan

Untuk setiap studi kasus, laporan dan notebook Colab harus memuat:

1. Penjelasan singkat dataset (jumlah sampel, fitur, tipe data).
2. Proses preprocessing (missing values, normalisasi, pembuatan fitur baru).
3. Hasil clustering KMeans dan DBSCAN, lengkap dengan:
    - Nilai Silhouette dan Davies–Bouldin
    - Visualisasi 2D (PCA/TSNE opsional)
4. Implementasi Annoy:
    - Pemilihan 3–5 titik query secara acak
    - Output index dan tetangga terdekat dengan nilai jaraknya

Pengumpulan: notebook ipynb link dimasukkan pada laporan tugas berformat pdf. Dikumpulkan pada LMS

Template Code

1. Library

```
# Import library yang diperlukan
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans, DBSCAN
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score, davies_bouldin_score
from annoy import AnnoyIndex
import matplotlib.pyplot as plt
import seaborn as sns
```

2. Upload Dataset ke Colab (Tuliskan catatan singkat tentang kondisi awal dataset dan kolom mana yang memiliki missing values)

```
# Upload atau load dataset dari Kaggle
# (ubah path sesuai dataset masing-masing)

df = pd.read_csv('/content/dataset.csv')

df.info()
df.describe()
df.head()
```

3. Data Preprocessing (Catat kolom apa yang memiliki missing values dan fitur baru apa yang Anda buat)

```
# Tangani missing values
df.fillna(df.mean(), inplace=True)

# (Opsional) Buat fitur baru, contoh:
#   df['NewFeature']   =   df['Feature1']   /
df['Feature2']

# Normalisasi data numerik
num_df                                      =
df.select_dtypes(include=['float64','int64'])
scaler = StandardScaler()
```

```
X_scaled = scaler.fit_transform(num_df)
```

4. Clustering dengan Kmeans dan DBScan

```
# KMeans

# DBSCAN

# Evaluasi
print("KMeans                          Silhouette:",
silhouette_score(X_scaled, labels_kmeans))
print("KMeans                       Davies-Bouldin:",
davies_bouldin_score(X_scaled, labels_kmeans))
print("DBSCAN                          Silhouette:",
silhouette_score(X_scaled, labels_dbscan))
print("DBSCAN                       Davies-Bouldin:",
davies_bouldin_score(X_scaled, labels_dbscan)
```

5. Visualisasi Clustering (2D)

```
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

plt.figure(figsize=(6,5))
plt.scatter(X_pca[:,0],        X_pca[:,1],
c=labels_kmeans, cmap='viridis')
plt.title("KMeans                Clustering
Visualization")
plt.xlabel("PCA 1")
plt.ylabel("PCA 2")
plt.show()
```

6. ANN

```
dim = X_scaled.shape[1]
ann = AnnoyIndex(dim, 'euclidean')

# Build index
…

# Pilih query point (misal index ke-42)
query_idx = 42
```

```
print(f"Query Point Index: {query_idx}")
print("Nearest Neighbors and Distances:")
for n, d in zip(neighbors, distances):
    print(f"Neighbor Index: {n} | Distance:
{d:.4f} | Cluster (KMeans): {labels_kmeans[n]}")
```

7. Tulis kesimpulan singkat:
   a. Perbedaan hasil KMeans dan DBSCAN, mana yang lebih baik diantara kedua model ini dan jelaskan jawaban anda
   b. Nilai metrik terbaik (Silhouette, DBI).
   c. Hasil query Annoy: apakah tetangga yang ditemukan termasuk dalam cluster yang sama? Jelaskan jawaban anda.