LAPORAN PRAKTIKUM
DATA SCIENCE

DATA HIGH DIMENSIONALITY REDUCTION (t-SNE)

LAURA VEGAWANI PASARIBU

11422009

ST Teknologi Rekayasa Perangkat Lunak

INSTITUT TEKNOLOGI DEL
FAKULTAS VOKASI

t-SNE

t-distributed stochastic neighbord embending(t-SNE) adalah sebuah algoritma dalam masin learning untuk mengurangi jumlah dimensi dataset yang akan digunakan. t-SNE merupakan teknik yang bersifat non-linear yang cocok untuk masalah high-dimensional data kedalam 2 atau 3 dimensi sehingga dapat divisualisasikan menggunakan scatter plot. t-SNE memodelkan objek dengan dimensi yang tinggi berdasarkan tingkat kemiripan data point melalui nilai jarak (distance).

Mnist Dataset

1. Perhatikan details dari dataset mnist yang digunakan seperti dibawah ini :

```
from sklearn.datasets import fetch_openml
import pandas as pd
mnist = fetch_openml('mnist_784', version=1, parser='auto')
d = mnist.data
l = mnist.target
df = pd.DataFrame(d)
df['label'] = l
print(df.head())
```

   Penjelasan :
   Kode ini bertujuan untuk mengunduh, memproses, dan menampilkan dataset MNIST yang berisi gambar digit tulisan tangan. Pada code menggunakan fetch_openml dari sklearn.dataset untuk mengambil dataset dari OpenMl. Data piksel diubah menjadi bentuk table menggunakan pandas.DataFrame agar lebih mudah diolah.
2. Dari dataset tersebut diketahui jumlah baris dan feature yang digunakan

```
   pixel1  pixel2  pixel3  pixel4  pixel5  pixel6  pixel7  pixel8  pixel9  \
0       0       0       0       0       0       0       0       0       0
1       0       0       0       0       0       0       0       0       0
2       0       0       0       0       0       0       0       0       0
3       0       0       0       0       0       0       0       0       0
4       0       0       0       0       0       0       0       0       0

   pixel10  ...  pixel776  pixel777  pixel778  pixel779  pixel780  pixel781  \
0        0  ...         0         0         0         0         0         0
1        0  ...         0         0         0         0         0         0
2        0  ...         0         0         0         0         0         0
3        0  ...         0         0         0         0         0         0
4        0  ...         0         0         0         0         0         0

   pixel782  pixel783  pixel784  label
0         0         0         0      5
1         0         0         0      0
2         0         0         0      4
3         0         0         0      1
4         0         0         0      9

[5 rows x 785 columns]
```

3. Pada tahap ini dataset dengan jumlah baris 5 dan kolom 785 akan dilakukan Standar scalling. Jelaskan perubahan atau transisi dari step 2 yang menghasilkan (70000, 784)!

```
from sklearn.preprocessing import StandardScaler

standardized_data = StandardScaler().fit_transform(d)
print(standardized_data.shape)
```

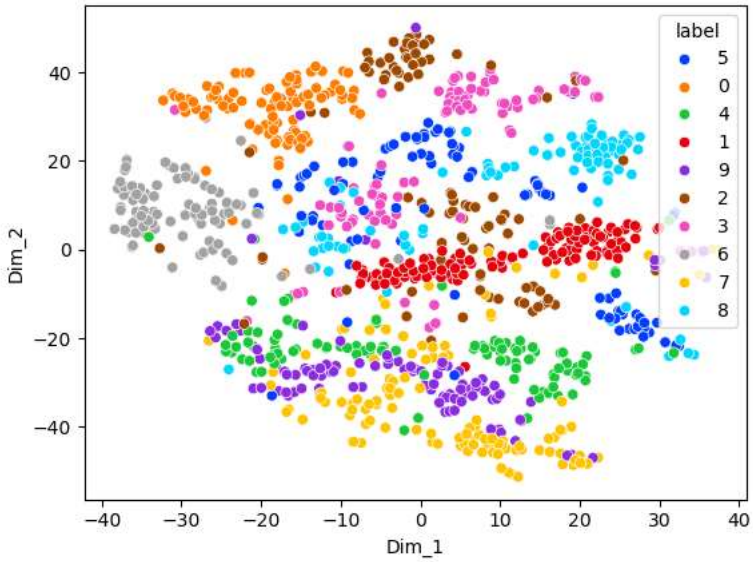
```
(70000, 784)
```

   Penjelasan :
   Pada step 2 dataset memiliki 785 kolom (784 fitur pikes + 1 kolom label), jumlah baris 70.000, label angka (0-9) masih termasuk dalam dataset namun pada step 3 dataset

mengalami standarisasi menggunakan StandardScaler(), yang menesuaikan setiap fitur sehingga memiliki rat-rata 0 dan standar deviasi 1. Label tidak termasuk dalam scaling karena hanya fitur numerik (piksel) yang distandarisasi. Hasilnya array dengan dimensi (70.000, 784), yang dimana hanya fitur piksel yang diproses, tanpa kolom label. Pada step 3 kolom label dihapus sebelum proses standardisasi, karena tidak relevan untuk normalisasi fitur numerik, StandardScaler() hanya bekerja pada fitur numerik (784 kolom piksel), sehingga label harus dipisahkan sebelum transformasi.

4. Perhatikan standarisasi yang digunakan label dan data!

```
import numpy as np
import pandas as pd
import seaborn as sns #impor seaborn dengan alias 'sns'
import matplotlib.pyplot as plt
from sklearn.manifold import TSNE

data_1000 = standardized_data[0:1000, :]
labels_1000 = l[0:1000]

model = TSNE(n_components=2, random_state=0)
tsne_data = model.fit_transform(data_1000)
tsne_data = np.vstack((tsne_data.T, labels_1000)).T

#Buat DataFrame
tsne_df = pd.DataFrame(data=tsne_data, columns=("Dim_1", "Dim_2", "label"))

#Plot menggunakan seaborn
tsne_df["label"] = tsne_df["label"].astype(str)
sns.scatterplot(data=tsne_df, x="Dim_1", y="Dim_2", hue="label", palette="bright")
plt.show()
```



Penjelasan :

Visualisasi hasil dari proses dimensi reduksi menggunakan algoritma t-SNE (t-Distributed Stochastic Neighbor Embedding) pada data 1000 sampel. t-SNE adalah teknik visualisasi data dimensi tinggi ke dalam ruang berdimensi rendah, biasanya 2D atau 3D, sehingga struktur data dapat diamati dengan lebih baik. Pada visualisasi ini, setiap titik merepresentasikan satu sampel data, dan warna titik menunjukkan label kelas dari sampel tersebut. Dapat dilihat bahwa data terbagi ke dalam beberapa klaster yang berbeda, yang mengindikasikan adanya struktur atau pola tertentu dalam data.

Tugas

1. Silahkan melakukan dimensionality reduction dengan dataset DataMahasiswa pada praktikum sebelumnya
2. Jelaskan langkah-langkah dan perubahan data yang terjadi selama proses dimensionality reduction!
3. Silahkan melakukan perhitungan untuk :
   a. Ambilah sampling dari dataset tersebut 15 baris pertama

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.manifold import TSNE
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler

# Membaca dataset dengan penanganan error
file_path = r"C:\Users\User\Data Science\W05S02\Data_Mahasiswa.csv"

try:
    df = pd.read_csv(file_path, sep=None, engine='python', encoding="utf-8", on_bad_lines="skip")
    print("File berhasil dibaca!")
except Exception as e:
    print(f"Error saat membaca file: {e}")
    exit()

# Pilih hanya kolom numerik
df_numeric = df.select_dtypes(include=["number"])

# Ambil 15 Baris Pertama (Sampling)
df_numeric = df_numeric.head(15)
```

   b. The t-SNE algorithm finds the similarity measure between pairs of instances in higher and lower dimensional space. After that, it tries to optimize two similarity measures.

```python
# Normalisasi Data
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data_imputed)

# Jalankan t-SNE
try:
    tsne = TSNE(n_components=2, perplexity=5, init="random", random_state=42)
    data_tsne = tsne.fit_transform(data_scaled)
except Exception as e:
    print(f"Error saat menjalankan t-SNE: {e}")
    exit()
```

   c. t-SNE models a point being selected as a neighbor of another point in both higher and lower dimensions. It starts by calculating a pairwise similarity between all data points in the highdimensional space using a Gaussian kernel. The points far apart have a lower probability of being picked than the points close together.
   Jawab :
   Terjadi secara internal dalam fit_transform(data_scaled), di mana Gaussian kernel digunakan untuk menghitung probabilitas bahwa satu titik akan dipilih sebagai tetangga titik lain.
   d. Map higher-dimensional data points onto lower-dimensional space while preserving the pairwise similarities.

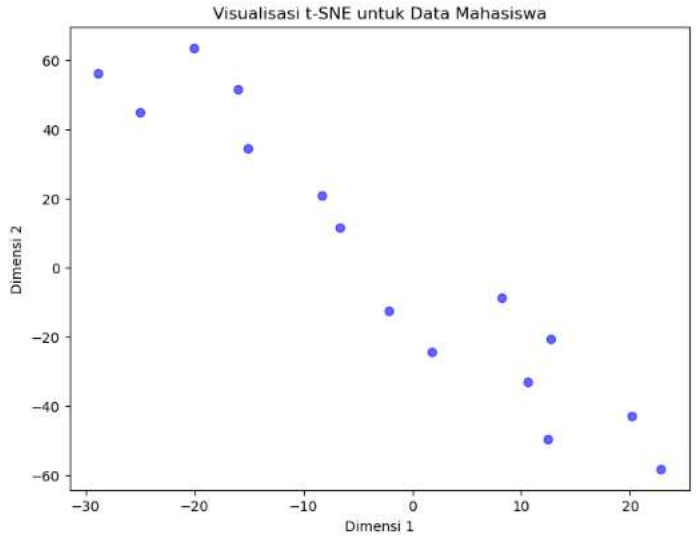
```python
data_tsne = tsne.fit_transform(data_scaled)
```

e. It is achieved by minimizing the divergence between the original high-dimensional and lowerdimensional probability distribution. The algorithm uses gradient descent to minimize the divergence. The lower-dimensional embedding is optimized to a stable state.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.manifold import TSNE
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler

# Membaca dataset dengan penanganan error
file_path = r"C:\Users\User\Data Science\W05S02\Data_Mahasiswa.csv"

try:
    df = pd.read_csv(file_path, sep=None, engine='python', encoding="utf-8", on_bad_lines="skip")
    print("File berhasil dibaca!")
except Exception as e:
    print(f"Error saat membaca file: {e}")
    exit()

# Pilih hanya kolom numerik
df_numeric = df.select_dtypes(include=["number"])

# Ambil 15 Baris Pertama (Sampling)
df_numeric = df_numeric.head(15)

# Cek & Tangani Nilai N/A
print("Jumlah Nilai sebelum imputasi:\n", df_numeric.isnull().sum())

imputer = SimpleImputer(strategy="mean")
data_imputed = imputer.fit_transform(df_numeric)

# Normalisasi Data
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data_imputed)

# Jalankan t-SNE
try:
    tsne = TSNE(n_components=2, perplexity=5, init="random", random_state=42)
    data_tsne = tsne.fit_transform(data_scaled)
except Exception as e:
    print(f"Error saat menjalankan t-SNE: {e}")
    exit()

# Konversi ke DataFrame
tsne_df = pd.DataFrame(data_tsne, columns=["Dim_1", "Dim_2"])
print("\nHasil Transformasi t-SNE:")
print(tsne_df)

# Visualisasi
plt.figure(figsize=(8, 6))
plt.scatter(tsne_df["Dim_1"], tsne_df["Dim_2"], c='blue', alpha=0.6)
plt.xlabel("Dimensi 1")
plt.ylabel("Dimensi 2")
plt.title("Visualisasi t-SNE untuk Data Mahasiswa")
plt.show()
```

```
File berhasil dibaca!
Jumlah Nilai sebelum imputasi:
 Programming Skill    0
dtype: int64

Hasil Transformasi t-SNE:
        Dim_1      Dim_2
0  -25.031668  44.869057
1   12.455816 -49.568264
2  -16.037170  51.530304
3  -20.067163  63.598545
4   -6.614944  11.520743
5   22.847986 -58.080223
6   10.629889 -32.900421
7    8.244394  -8.717136
8   -8.302712  20.885338
9  -28.883282  56.250847
10  12.744308 -20.472231
11  20.163904 -42.711205
12   1.768849 -24.200495
13  -2.133908 -12.311777
14 -15.140038  34.591721
```

Output



Penerapan t-SNE pada dataset DataMahasiswa berhasil mereduksi dimensi tinggi menjadi dua dimensi untuk memudahkan analisis dan visualisasi. Proses ini diawali dengan pengambilan sampel 15 baris pertama, penanganan nilai NaN dengan imputasi mean, serta normalisasi data menggunakan standarisasi Z-score. t-SNE bekerja dengan menghitung kesamaan antar data di dimensi tinggi menggunakan Gaussian kernel, lalu memetakannya ke dimensi rendah dengan mempertahankan hubungan lokal antar titik melalui t-Student distribution. Proses optimasi dilakukan menggunakan gradient descent untuk menghasilkan representasi yang stabil. Hasil visualisasi menunjukkan pola distribusi data yang lebih jelas, membantu dalam mengidentifikasi struktur tersembunyi dan klaster dalam dataset. Metode ini sangat berguna dalam memahami hubungan antar data yang sulit terlihat dalam dimensi tinggi.

4. Silahkan melakukan perhitungan manual untuk 3a s/d 3e dan visualisasikan dalam bentuk gambar!

Nama : Laura Veganovi Pinariu
NIM : 11921009
Kelas : 93 TRPL 1

Perhitungan Manual

3 a) Sampling 15 baris pertama

ID	T1	T2	T3
1	2	5	8
2	3	6	9
...	...	...	...
15	5	8	7

3 b) (menangani nan dan normalisasi)

Normalisasi Z-score

$$X_{scaled} = \frac{X - M}{\sigma}$$

F1 (M = 3.5, σ = 1.5)

$$X_{scaled}(1) = \frac{2 - 3.5}{1.5} = -1.0$$

$$X_{scaled}(2) = \frac{3 - 3.5}{1.5} = -0.33$$

c) perhitungan Probabilitas Gaussian kernel (3c)

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)}{\sum_{k \neq i} \exp\left(-\|x_i - x_k\|^2 / 2\sigma^2\right)}$$

$x_1 = (-1.0, 0.5)$

$x_2 = (-0.33, 1.2)$

Euclidean : $[x_1 - x_2^2)$

$= (-1.0 + 0.33)^2 + (0.5 - 1.2)^2$

$= 0.45 + 0.49$

$= 0.94$

Probabilitas :

$$p_{2|1} = \frac{e^{-0.94/2}}{\sum_{k \neq 1} e^{-\|x_1 - x_k\|^2/2}}$$

d) Mapping ke dimensi berenah (Distribusi t - Student)

$$q_{j|i} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq i} (1 + \|y_i - y_k\|^2)^{-1}}$$

e) Optimasi dengan gradient Descendent (3e)

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{j|i} - q_{j|i})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

$$y_i^{t+1} = y_i^t - \eta \frac{\delta C}{\delta y_i^t}$$

```python
# Visualisasi
plt.figure(figsize=(8, 6))
plt.scatter(tsne_df["Dim_1"], tsne_df["Dim_2"], c='blue', alpha=0.6)
plt.xlabel("Dimensi 1")
plt.ylabel("Dimensi 2")
plt.title("Visualisasi t-SNE untuk Data Mahasiswa")
plt.show()
```