TUGAS PRAKTIKUM MANDIRI

Judul : Implementasi Principal Component Analysis (PCA) pada Dataset Breast Cancer untuk Reduksi Dimensi Data Medis

Nama Mahasiswa	: Al Hijir
Program studi	: Teknik Informatika, STT Terpadu Nurul Fikri, Depok
E-mail	: 0110224222@student.nurulfikri.ac.id

Abstrak

Principal Component Analysis (PCA) merupakan salah satu teknik reduksi dimensi yang banyak digunakan dalam analisis data berdimensi tinggi, khususnya pada bidang medis. Dataset medis umumnya memiliki banyak fitur numerik yang dapat menyebabkan kompleksitas tinggi dalam proses analisis. Oleh karena itu, diperlukan metode untuk menyederhanakan data tanpa menghilangkan informasi penting.

Pada praktikum ini, PCA diterapkan pada dataset Breast Cancer yang diperoleh dari Kaggle. Tahapan yang dilakukan meliputi preprocessing data, penghapusan atribut yang tidak relevan, standardisasi data, serta penerapan PCA untuk mereduksi dimensi dataset. Selanjutnya, dilakukan visualisasi hasil PCA menggunakan dua komponen utama untuk mengamati distribusi dan pemisahan data berdasarkan kelas diagnosis.

Hasil praktikum menunjukkan bahwa PCA mampu mereduksi dimensi data secara efektif dengan tetap mempertahankan sebagian besar variansi data. Visualisasi dua komponen utama juga memperlihatkan pemisahan yang cukup jelas antara data kanker jinak (benign) dan ganas (malignant). Dengan demikian, PCA dapat digunakan sebagai tahap awal yang efektif dalam pengolahan dan analisis data medis berdimensi tinggi.

1. Import Library

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

from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
```

Penjelasan:

- numpy digunakan untuk operasi numerik dan perhitungan matematis.
- pandas digunakan untuk membaca dan mengelola dataset dalam bentuk tabel (DataFrame).
- matplotlib.pyplot dan seaborn digunakan untuk visualisasi data.
- StandardScaler digunakan untuk menstandarkan data agar memiliki skala yang sama.
- PCA digunakan untuk melakukan reduksi dimensi menggunakan Principal Component Analysis.

2. Mengakses File di Google Drive

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

Penjelasan:

Digunakan untuk menghubungkan Google Drive agar file dataset

dapat diakses langsung di

lingkungan Google Colab.

3. Menentukan Path Folder

```
path = "/content/gdrive/MyDrive/praktikmML/praktikum12"
```

Penjelasan:

Variabel path digunakan untuk menentukan lokasi folder kerja tempat

dataset disimpan.

4. Load Dataset

```
data = pd.read_csv('/content/gdrive/MyDrive/PraktikumML/praktikum12/data/data.csv')

data.head()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compact
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	

5 rows × 33 columns

Penjelasan:

- Dataset Breast Cancer dimuat dari file data.csv.
- Fungsi head() digunakan untuk menampilkan 5 data teratas guna memastikan dataset berhasil dimuat dengan benar.

5. Cek Informasi Dataset

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   diagnosis               569 non-null    int64
 1   radius_mean             569 non-null    float64
 2   texture_mean            569 non-null    float64
 3   perimeter_mean          569 non-null    float64
 4   area_mean               569 non-null    float64
 5   smoothness_mean         569 non-null    float64
 6   compactness_mean        569 non-null    float64
 7   concavity_mean          569 non-null    float64
 8   concave points_mean     569 non-null    float64
 9   symmetry_mean           569 non-null    float64
 10  fractal_dimension_mean  569 non-null    float64
 11  radius_se               569 non-null    float64
 12  texture_se              569 non-null    float64
 13  perimeter_se            569 non-null    float64
 14  area_se                 569 non-null    float64
 15  smoothness_se           569 non-null    float64
```

Penjelasan:

- Digunakan untuk melihat struktur dataset, jumlah baris dan kolom, tipe data, serta mendeteksi adanya nilai kosong (missing values).

6. Preprocessing Data

```
data = data.drop(columns=['id'], errors='ignore')

data = data.dropna(axis=1, how='all')
```

Penjelasan:

- Kolom id tidak memiliki pengaruh terhadap analisis sehingga dihapus.
- Kolom yang seluruh nilainya kosong (misalnya Unnamed: 32) dihapus agar tidak menimbulkan error saat proses PCA.

7. Encoding Label

```
data['diagnosis'] = data['diagnosis'].map({
    'M': 1,   # Malignant
    'B': 0    # Benign
})
```

Penjelasan:

- Label diagnosis yang awalnya berupa kategori diubah menjadi nilai numerik.
- M (Malignant) diubah menjadi 1 dan B (Benign) menjadi 0.
- Langkah ini diperlukan agar data dapat diproses secara numerik.

8. Memisahkan Fitur dan Target

```
X = data.drop('diagnosis', axis=1)
y = data['diagnosis']
```

Penjelasan:

- X berisi seluruh fitur numerik.
- y berisi label diagnosis.
- PCA hanya diterapkan pada fitur (X), bukan pada label.

9. Standardisasi Data

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Penjelasan:

- PCA sangat sensitif terhadap skala data.
- Standardisasi dilakukan agar setiap fitur memiliki mean 0 dan standar deviasi 1.

- Hal ini memastikan setiap fitur memiliki kontribusi yang seimbang dalam PCA.

10. Implementasi PCA

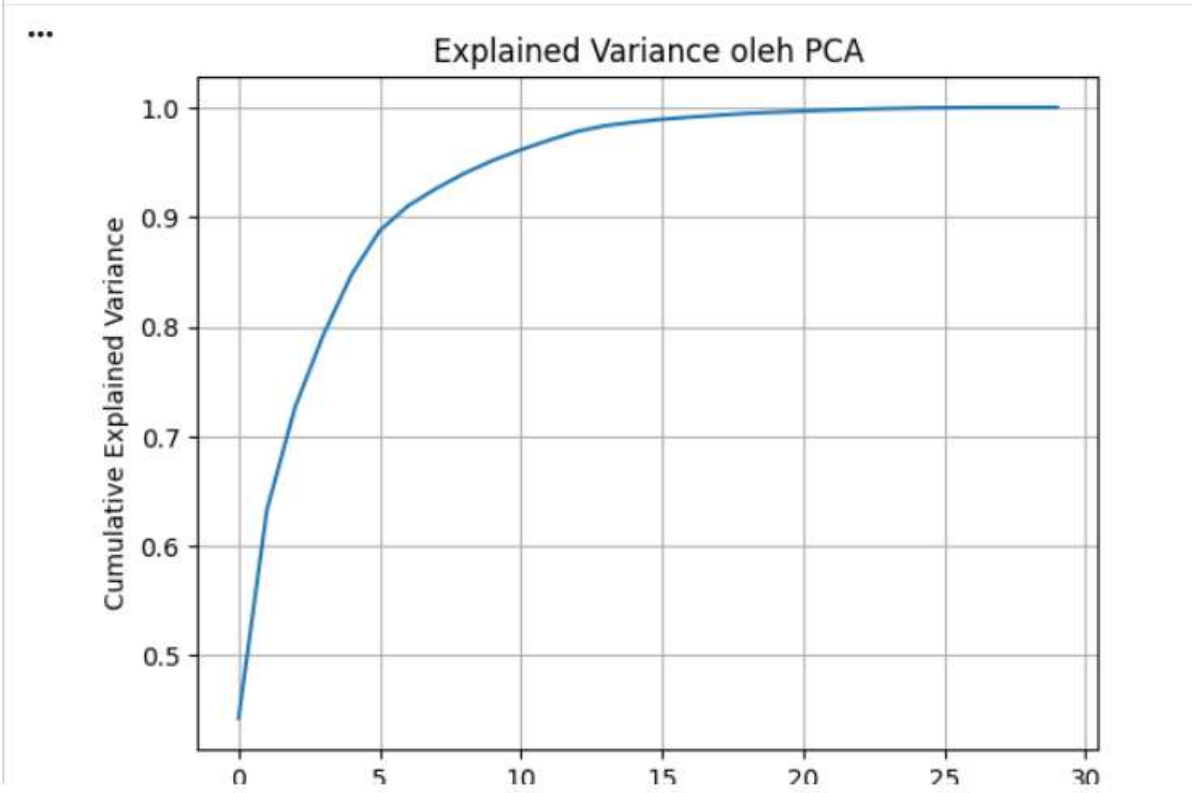
```
pca = PCA()
X_pca = pca.fit_transform(X_scaled)
```

Penjelasan:

- PCA diterapkan untuk mentransformasikan data ke ruang baru dengan komponen utama.
- Semua komponen dihitung untuk mengetahui seberapa besar variansi yang dijelaskan oleh masing-masing komponen.

11. Explained Variance Ratio

```
plt.figure()
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('Jumlah Komponen Utama')
plt.ylabel('Cumulative Explained Variance')
plt.title('Explained Variance oleh PCA')
plt.grid()
plt.show()
```



Penjelasan:

- Grafik ini menunjukkan akumulasi variansi yang dijelaskan oleh komponen utama.
- Digunakan untuk menentukan jumlah komponen optimal yang dapat mempertahankan sebagian besar informasi data.

12. PCA dengan 2 Komponen Utama

```
pca_2 = PCA(n_components=2)
X_pca_2 = pca_2.fit_transform(X_scaled)
```

Penjelasan:

- PCA diterapkan kembali dengan hanya 2 komponen utama.
- Tujuannya untuk visualisasi data dalam dua dimensi.

13. DataFrame Hasil PCA

```
pca_df = pd.DataFrame(
    X_pca_2,
    columns=['Principal Component 1', 'Principal Component 2']
)

pca_df['Diagnosis'] = y.values
```

Penjelasan:

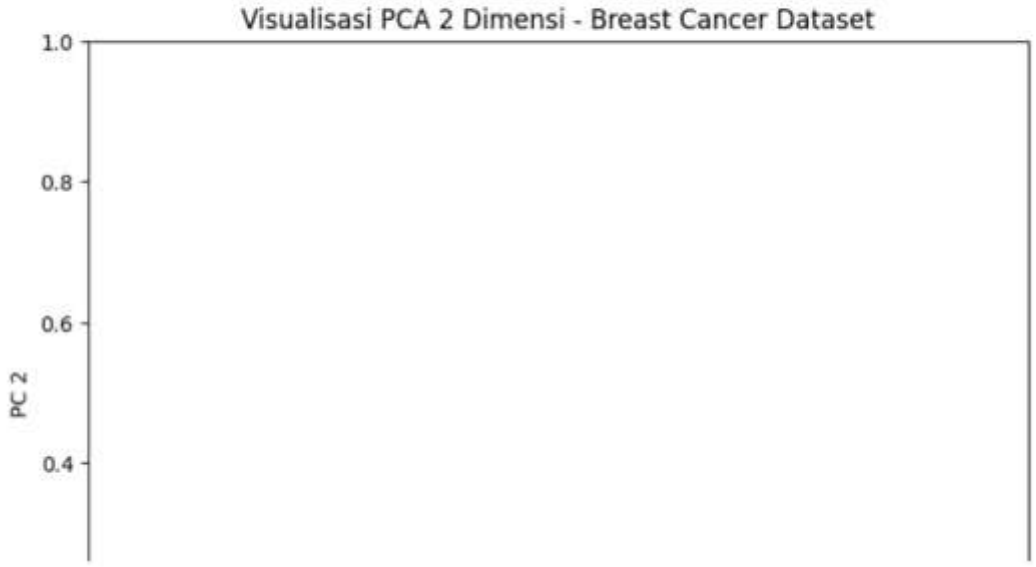
- Hasil PCA disimpan dalam DataFrame baru agar mudah dianalisis dan divisualisasikan.
- Label diagnosis ditambahkan kembali untuk membedakan kelas data.

14. Visualisasi PCA

```
plt.figure(figsize=(8,6))
sns.scatterplot(
    x='Principal Component 1',
    y='Principal Component 2',
    hue='Diagnosis',
    data=pca_df,
    palette='Set1'
)

plt.title('Visualisasi PCA 2 Dimensi - Breast Cancer Dataset')
plt.xlabel('PC 1')
plt.ylabel('PC 2')
plt.show()
```

```
/tmp/ipython-input-1165571932.py:2: UserWarning: Ignoring `palette` because no `hue` variable has
  sns.scatterplot(
```



Penjelasan:

- Scatter plot digunakan untuk memvisualisasikan hasil PCA dalam 2 dimensi.
- Warna berbeda menunjukkan kelas diagnosis.
- Visualisasi ini membantu melihat sejauh mana PCA mampu memisahkan data benign dan malignant.

15. Interpretasi PCA

```
print("Explained Variance PC1 & PC2:")
print(pca_2.explained_variance_ratio_)
print("Total Variance:", sum(pca_2.explained_variance_ratio_))
```

```
Explained Variance PC1 & PC2:
[0.44272026 0.18971182]
Total Variance: 0.6324320765155944
```

Penjelasan:

- Menampilkan persentase variansi yang dijelaskan oleh PC1 dan PC2.
- Total variansi menunjukkan seberapa besar informasi data yang masih dipertahankan setelah reduksi dimensi.