

PCA (Principal Component Analysis)

Adinda Putri - 13523071

PCA merupakan algoritma untuk mereduksi dimensi dari data, dengan cara mentransformasikan variabel-variabel menjadi lebih sedikit tetapi mengandung informasi penting dari variabel-variabel awal. PCA menawarkan trade-off antara accuracy dan simplicity. **Principal components** merupakan variabel-variabel baru yang didapat dari kombinasi linear dari variabel-variabel awal. Kombinasi tersebut dilakukan sedemikian rupa sehingga variabel-variabel baru, yaitu principal components, tidak saling berkorelasi dan sebagian besar informasi dari variabel-variabel awal dipadatkan ke dalam principal component pertama. Prinsipnya, data 10 dimensi akan menghasilkan 10 principal component. Akan tetapi PCA sebisa mungkin memadatkan informasi pada principal component pertama, lalu memadatkan informasi sisanya di principal component kedua, dan seterusnya.

Cara Kerja

1. **Standardization and Centering Data:** Tahap ini dilakukan untuk menyeimbangkan sensitivitas tiap variabel dan mengurangi dominasinya.
2. **Covariance Matrix Computation:** Covariance matrix merupakan matriks simetris $p \times p$ (p adalah dimensi data) dengan entrinya terdiri dari covariances antarvariabel awal. Data 3 dimensi dengan 3 variabel x , y , dan z memiliki covariance matrix sebagai berikut.

$$\begin{bmatrix} Cov(x, x) & Cov(x, y) & Cov(x, z) \\ Cov(y, x) & Cov(y, y) & Cov(y, z) \\ Cov(z, x) & Cov(z, y) & Cov(z, z) \end{bmatrix}$$

Gambar 1. Covariance matrix

Sumber: Penulis

Diagonal utama covariance matrix berisi variance dari masing-masing variabel awal karena $Cov(a, a) = Var(a)$. Lalu, karena $Cov(a, b) = Cov(b, a)$, maka matriks bersifat simetris terhadap diagonal utama. Covariance pada entri bernilai positif jika dua variabel meningkat atau menurun

bersamaan (berkorelasi), dan bernilai negatif jika yang satu meningkat ketika yang lainnya menurun (berkorelasi terbalik)

3. Eigen Decomposition and Identifying Principal Components: Eigenvectors dan eigenvalues selalu berpasangan, artinya setiap eigenvector mempunyai eigenvalue. Jumlah keduanya selalu sama dengan dimensi data. Eigenvectors dari covariance matrix merupakan **arah sumbu** dimana terdapat variance tertinggi (informasi terbanyak) sehingga disebut principal components. Sementara itu eigenvalues merupakan koefisien dari eigenvector yang menunjukkan besar variance pada setiap principal component. Dengan mengurutkan eigenvector berdasarkan eigenvalues-nya, dari besar ke kecil, didapatkan principal components terurut berdasarkan signifikansinya.

$$v1 = \begin{bmatrix} 0.6778736 \\ 0.7351785 \end{bmatrix} \quad \lambda_1 = 1.284028$$
$$v2 = \begin{bmatrix} -0.7351785 \\ 0.6778736 \end{bmatrix} \quad \lambda_2 = 0.04908323$$

Gambar 2. Eigen Decomposition

Sumber: Penulis

Dari gambar di atas, didapatkan bahwa eigenvector dari principal component pertama (PC1) adalah v1 sedangkan dari principal component kedua (PC2) adalah v2. Dalam bentuk persentase, dapat dihitung bahwa PC1 dan PC2 mengandung secara berturut-turut 96% dan 4% dari variance data.

4. Feature Selection and Creating a Feature Vector

Jika akan dipilih p variabel (eigenvectors) dari n variabel, dimensi data akan menjadi p . Pada gambar langkah 3 di atas, Jika v1 dan v2 akan diambil dua-duanya, maka dapat dibentuk **feature vector** sebagai berikut.

$$\begin{bmatrix} 0.6778736 & -0.7351785 \\ 0.7351785 & 0.6778736 \end{bmatrix}$$

Gambar 3. Feature Vector v1 dan v2

Sumber: Penulis

Atau, jika hanya v1 yang akan diambil, feature vector menjadi

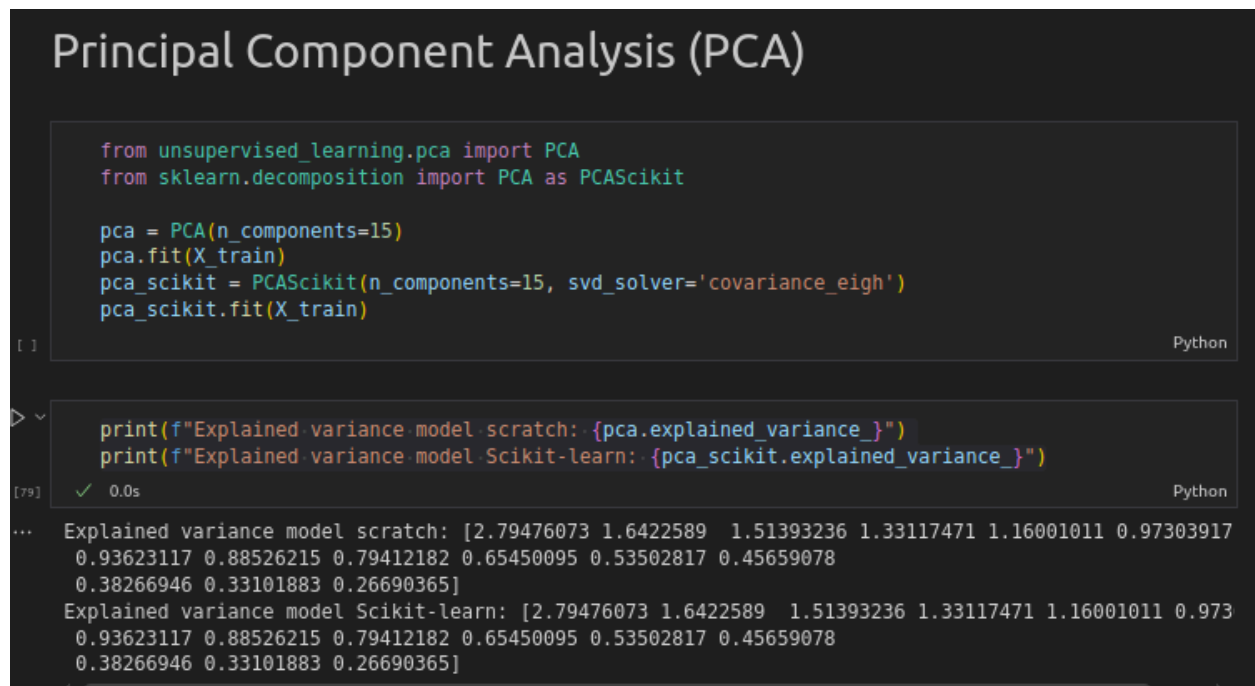
$$\begin{bmatrix} 0.6778736 \\ 0.7351785 \end{bmatrix}$$

Gambar 4. Feature Vector v1

Sumber: Penulis

5. **Data Projection:** Hingga langkah 4, data awal tidak berubah. Algoritma hanya menghasilkan feature vector. Untuk melakukan dimensionality reduction pada data, dilakukan operasi berikut.

Perbandingan model dari scratch dengan dari Scikit-Learn



The screenshot shows a Jupyter Notebook titled "Principal Component Analysis (PCA)". It contains two code cells. The first cell initializes and trains a PCA model from scratch using NumPy and SciPy. The second cell prints the explained variance for both the scratch model and the Scikit-Learn PCA model. The output shows that both models have identical explained variance values.

```
from unsupervised_learning.pca import PCA
from sklearn.decomposition import PCA as PCA_Scikit

pca = PCA(n_components=15)
pca.fit(X_train)
pca_scikit = PCA_Scikit(n_components=15, svd_solver='covariance_eigh')
pca_scikit.fit(X_train)
```

```
print(f"Explained variance model scratch: {pca.explained_variance}")
print(f"Explained variance model Scikit-learn: {pca_scikit.explained_variance}")
```

Output:

```
Explained variance model scratch: [2.79476073 1.6422589 1.51393236 1.33117471 1.16001011 0.97303917
0.93623117 0.88526215 0.79412182 0.65450095 0.53502817 0.45659078
0.38266946 0.33101883 0.26690365]
Explained variance model Scikit-learn: [2.79476073 1.6422589 1.51393236 1.33117471 1.16001011 0.973
0.93623117 0.88526215 0.79412182 0.65450095 0.53502817 0.45659078
0.38266946 0.33101883 0.26690365]
```

Gambar 5. Inisialisasi, Training, dan Perbandingan Masing-Masing Model

Sumber: Penulis

Seperti terlihat pada hasil di atas, nilai explained variance yang dihasilkan kedua model PCA identik. Hasil ini membuktikan bahwa logika inti dari implementasi model dari scratch, yang meliputi perhitungan matriks kovarians, ekstraksi eigenvalue, dan eigenvector, dan sorting principal components, sudah benar.

Referensi:

[1] *Step by Step Explanation of Principal Component Analysis (PCA)*, Built In, 2023. [Daring]. Tersedia: <https://builtin.com/data-science/step-step-explanation-principal-component-analysis>. [Diakses: 2 September 2025].

[2] R. Krimsony, *PCA from Scratch in Python with MNIST Dataset*, GitHub, 2021. [Daring]. Tersedia: https://github.com/redwankarimsony/PCA-from-Scratch-in-Python/blob/main/PCA_with_MNIST_Dataset.ipynb. [Diakses: 2 September 2025].