Principal Component Analysis sebagai Teknik Reduksi Dimensi pada Data Penjualan Rumah di Kawasan Jabodetabek

Adhimas Aryo Bimo - 13523052
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13523052@std.stei.itb.ac.id

***Abstrak*— Dalam era di mana data berlimpah dan kompleksitasnya terus meningkat, tidak jarang kita dihadapkan dengan dataset berdimensi tinggi. Salah satu cara untuk menurunkan dimensi data tanpa mengurangi informasi yang relevan adalah dengan menggunakan metode PCA (Principal Component Analysis). PCA menawarkan dua pendekatan utama, yaitu melalui Singular Value Decomposition (SVD) dan Matriks Kovarians. Penelitian ini menggunakan kedua pendekatan tersebut untuk mereduksi dimensi dataset penjualan rumah di kawasan Jabodetabek, sekaligus membandingkan performa model yang dilatih dengan data hasil reduksi dan data asli. Hasil penelitian menunjukkan bahwa PCA berhasil mereduksi fitur numerik dari 111 menjadi hanya 15 komponen utama. Model yang dilatih menggunakan data hasil reduksi mampu menghasilkan waktu eksekusi yang lebih cepat, dengan peningkatan efisiensi sebesar 12,9%. Namun, ada kompromi dalam bentuk sedikit peningkatan kesalahan sebesar 0,22%.**

***Kata Kunci*— PCA, SVD, Rumah, Reduksi Dimensi**

I. PENDAHULUAN

Dalam era digital yang semakin berkembang, data telah menjadi salah satu aset paling berharga dalam berbagai bidang, termasuk ekonomi, pendidikan, dan properti. Salah satu tantangan utama dalam analisis data adalah tingginya dimensi data yang sering kali menghambat proses analisis dan interpretasi. Dalam konteks analisis atas penjualan rumah seperti pada data penjualan rumah di Jabodetabek, terdapat banyak variabel atau fitur yang saling berkaitan, seperti lokasi, harga, fasilitas, dan lain-lain. Banyaknya fitur ini dapat menyebabkan masalah redundansi dan *overfitting* dalam model prediksi.

Gambar 1.1 Penjualan Rumah
Sumber :
https://radarbogor.jawapos.com/bogor/2475377944/viral-tren-gua-tunjukin-rumah-sudah-jadi-ternyata-ini-sosok-pria-yang-tawarkan-rumah-di-bogor

Principal Component Analysis (PCA) adalah salah satu teknik reduksi dimensi yang dapat digunakan untuk menyederhanakan dataset tanpa kehilangan informasi yang signifikan. PCA bekerja dengan mentransformasikan variabel yang berpotensi berkolerasi menjadi sekumpulan variabel yang lebih kecil yang disebut komponen utama. PCA dikembangkan pertama kali oleh Karl Pearson pada tahun 1901 dan semakin berkembangnya teknologi PCA makin populer digunakan karena PCA sangat efektif dalam memvisualisasikan dan mengeksplorasi kumpulan data berdimensi tinggi, atau data dengan banyak fitur, karena dapat dengan mudah mengidentifikasi tren, pola, atau *outlier*.

Makalah ini bertujuan untuk memahami penggunaan PCA dalam mereduksi dimensi dataset penjualan rumah di Jabodetabek. Proses ini meliputi standarisasi data, menghitung matriks kovarian, mencari vektor eigen dan nilai eigen, memilih komponen utama, dan proyeksi data ke ruang baru. Selain itu, makalah ini juga akan menampilkan performa model *machine learning* XGBRegressor dalam memprediksi harga rumah dan dibandingkan dengan data dalam dimensi penuh menggunakan metrik MAPE.

Melalui makalah ini diharapkan pembaca mampu memahami cara kerja PCA dan penerapan PCA dalam analisis data properti serta kelebihan dan keterbatasan dalam konteks pengambilan keputusan berbasis data.

II. LANDASAN TEORI

A. Matriks

Matriks merupakan susunan angka-angka atau elemen yang disusun dalam bentuk persegi panjang dengan baris dan kolom. Matriks sering digunakan dalam matematika, aljabar linear, dan berbagai bidang seperti komputer dan statistik.

Secara formal, matriks A dengan ukuran $m \times n$ didefinisikan sebagai berikut:

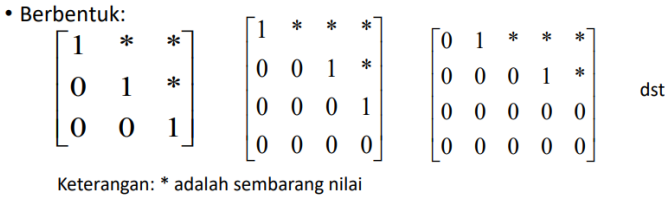
$$A = [a_{ij}] = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Gambar 2.1 Matriks
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-01-Review-Matriks-2023.pdf

Terdapat beberapa jenis matriks, diantaranya Matriks Kovarians dan Matriks Eselon Baris.

B. Matriks Eselon Baris

Matriks Eselon Baris merupakan matriks yang memiliki 1 utama (*leading one*) pada setiap baris, keuali baris yang seluruhnya nol.



Gambar 2.2 Matriks Eselon Baris
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-02-Matriks-Eselon-2023.pdf

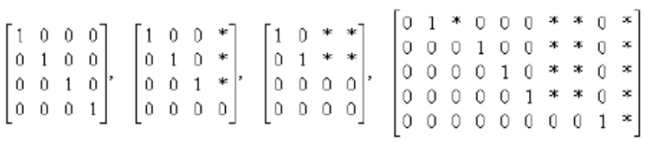
Matriks Eselon Baris memiliki beberapa sifat, di antaranya:

1. Jika sebuah baris tidak terdiri dari seluruhnya nol, maka bilangan tidak nol pertama di dalam baris tersebut adalah 1 (disebut 1 utama / *leading one*).

2. Jika ada baris yang seluruhnya nol, maka semua baris itu dikumpulkan pada bagian bawah matriks.

3. Di dalam dua baris berurutan yang tidak seluruhnya nol, maka 1 utama pada baris yang e=lebih rendah terdapat lebih jauh ke kanan daripada 1 utama pada baris yang lebih tinggi.

Matriks Eselon memiliki jenis lain, yakni Matriks Eselon Baris Tereduksi. Matriks ini memiliki sifat yang sama dengan Matriks Eselon Baris, tetapi terdapat tambahan sifat yakni Setiap kolom yang memiliki 1 utama memiliki nol di tempat lain.



Gambar 2.3 Matriks Eselon Baris Tereduksi
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-02-Matriks-Eselon-2023.pdf

B. Kovarians dan Matriks Kovarians

Kovarians adalah ukuran statistik yang menujukkan hubungan linear antara dua variabel. Kovarian menunjukkan arah hubungan (positif atau negatif) antara variabel. Secara matematis kovarians dapat dituliskan sebagai berikut.

$$Cov(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

Keterangan :

- X_i dan Y_i : Nilai ke-i dari variabel X dan Y,
- $\bar{X}$ dan $\bar{Y}$: Rata-rata dari X dan Y,
- n : jumlah sampel

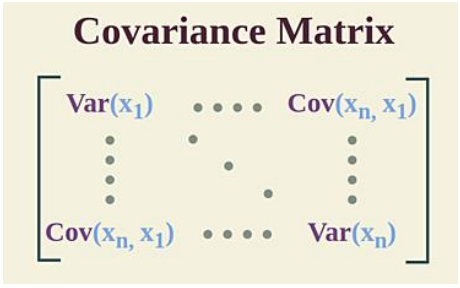
Jika nilai kovarians menghasilkan nilai yang positif, maka variabel bergerak ke arah yang sama (jika X meningkat, Y juga meningkat). Sebaliknya jika negatif maka variabel bergerak ke arah yang berlawanan (jika X meningkat, Y menurun). Sedangkan, jika menghasikan nilai 0, maka tidak ada hubungan linear antara X dan Y.

Matriks yang digunakan untuk menggambar nilai kovarians antara dua item dalam vektor acak disebut Matriks Kovarians. Matriks ini digunakan untuk menganalisis hubungan antar semua fitur dalam dataset. Jika dataset X memiliki n sampel dan p fitur, maka matriks kovarians berbentuk $p \times p$ dengan elemen $Cov(X_j, X_k)$ pada baris ke-j dan kolom ke-k. Persamaan dari matriks kovarians sebagai berikut.

$$Cov(X) = \frac{1}{n-1}(X - \mu)^{\top}(X - \mu)$$

Keterangan :

- X : Matriks data berukuran $n \times p$ (baris adalah sampel, kolom adalah fitur),
- μ : Vektor rata-rata dari setiap fitur,
- $(X - \mu)$: Matriks hasil pengurangan setiap elemen data dengan rata-rata fitur.



Gambar 2.4 Matriks Kovarians
Sumber :
https://www.geeksforgeeks.org/covariance-matrix/

B. Eliminasi Gauss

Eliminasi Gauss adalah metode dalam aljabar linear yang digunakan untuk menyelesaikan sistem persamaan linear. Metode ini bekerja dengan mengubah matriks augmented dari sistem persamaan linear menjadi matriks eselon baris.

Operasi baris ini mencakup penukaran baris, pengalian baris dengan bilangan non-nol, dan menambah kelipatan suatu baris ke baris lain.

$$[A \mid \mathbf{b}] = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array}\right]$$

Gambar 2.5 Matriks Augmented
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_n \end{bmatrix} \sim \text{OBE} \sim \begin{bmatrix} 1 & * & * & \dots & * & * \\ 0 & 1 & * & \dots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

Gambar 2.6 Metode Eliminasi Gauss
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-03-Sistem-Persamaan-Linier-2023.pdf

Setelah matriks dalam bentuk segitiga atas, solusi sistem persamaan linear dapat diperoleh dengan menggunakan subttusi mundir (*bakward subtitution*).

Dalam konteks Principal Component Analysis (PCA), Eliminasi Gmemiliki peran penting dalam manipulasi matriks untuk menghitung nilai eigen dan vektor eigen.

D. Eigenvector dan Eigenvalues

Eigenvector dan Eigenvalues saling berhubungan di aljabar liena. Kedua istilah tersebut digunakan dalam transformasi linear. Eigenvalues (Nilai Eigen) merupakan nilai skalar yang menunjukkan seberapa suat vektor berubah skalanya setelah diterapkan transformasi linear. Sedangkan Eigenvector (Vektor Eigen) merupakan vektor non-nol yang tidak berubah arahnya ketika dilakukan transformasi linear.

Nilai Eigen dan Vektor Eigen memenuhi persamaan berikut:

$$Av = \lambda v$$

Keterangan :
- A : Matriks persegi ($n \times n$),
- v : Vektor eigen, yaitu vektor non-nol,
- λ : Nilai eigen , yaitu skalar.

Konsep vektor eigen dan nilai eigen sangat penting dalam penggunaan PCA. PCA bertujuan untuk memahami data dengan merangkum pola utama dalam dataset yang memiliki banyak dimensi (fitur). Di sini, matriks A adalah matriks kovarians data, yang menunjukkan hubungan antar fitur. Vektor eigen mewakili arah dengan varians maksimum dalam data, sedangkan nilai eigen menunjukkan besarnya varians tersebut. Dengan memilih beberapa vektor eigen utama (yang memiliki nilai eigen terbesar), PCA mampu mereduksi dimensi data tanpa kehilangan informasi yang signifikan

Dengan kata lain, vektor eigen adalah kompas yang menunjukkan arah utama dalam data, dan nilai eigen adalah alat ukur yang memberitahu seberapa penting arah itu. Bersama-sama, mereka memungkinkan kita untuk merangkum data kompleks ke dalam bentuk yang lebih sederhana namun tetap bermakna. Tanpa vektor eigen dan nilai eigen, PCA tidak akan mampu menyaring pola tersembunyi yang tersebar dalam data kita. Mereka adalah jantung dari reduksi dimensi, membuka jalan bagi wawasan baru dalam dunia analisis data..

E. SVD

Singular Value Decomposition (SVD) merupakan salah satu metode dekomposisi pada aljabar linear. Dalam SVD, matriks A (berukuran $m \times n$) diuraikan menjadi tiga komponen utama:

$$A = U\sum V^{\top}$$

Keterangan :
- A : Matriks yang akan didekomposisi
- U : Matriks ortogonal ($m \times m$) yang kolomnya adalah vektor eigen dari $AA^{\top}$
- $\sum$: Matriks diagonal ($m \times n$) yang berisi nilai singular. Nilai singular ini adalah akar kuadrat dari nilai eigen matriks $A^{\top}A$
- $V^{\top}$: Transpos dari matriks ortogonal V ($n \times n$), yang kolomnya adalah vektor eigen dari $A^{\top}A$

Selain menggunakan matriks kovarians, PCA juga dapat dilakukan dengan menghitung SVD. Metode SVD lebih sering digunakan dalam implementasi PCA modern karena lebih efisien untuk dataset besar dan menghindari pembentukkan matriks kovarians. Walaupun begitu, baik pendekatan menggunakan SVD maupun matriks kovarians keduanya memungkinkan PCA untuk mereduksi dimensi data sambil tetap mempertahankan informasi utama.

F. PCA

Principal Component Analysis (PCA) adalah salah satu teknik analisis yang digunakan dalam bidang statistika untuk mereduksi dimensi dataset multivariabel namun tetap mempertahankan sebanyak mungkin informasi.

PCA mencari pola dalam data dengan mengidentifikasi arah varians maksimum yang disebut sebagai komponen utama (*principal components).*

Terdapat beberapa langkah untuk melakukan PCA.

1. Standarisasi Data
   Data harus distandarisasi dengan mengubah rata-rata fitur menjadi 0 dan varians satu. berikut persamaan standarisasi.

   $$z = \frac{x - \mu}{\sigma}$$

   Keterangan :
   - x : Nilai Asli,
   - μ : Nilai rata-rata fitur,
   - σ : Standar deviasi fitur

2. Hitung Matriks Kovarians / Hitung SVD
   a. Jika menggunakan Matriks Kovarians setelah menghitung matriks tersebut hitung nilai eigen dan vektor eigen. Nilai eigen menunjukkan jumlah varians yang dijelaskan oleh setiap komponen utama, sedangkan vektor eigen memberikan arah

komponen utama. Lalu urutkan nilai eigen dari yang terbesar ke terkecil untuk membentuk komponen utama.

b. Jika menggunakan SVD, hitung SVD dan didapatkan 3 komponen matriks. Nilai singular dalam $\sum$ berhubungan dengan varians data. dengan memeriksa nilai singular terbesar, dapat dipilih komponen utama yang menjelaskan varians terbesar.

3. Proyeksi Data ke Ruang Baru
Data diproyeksikan ke ruang dimensi rendah dengan mengalikan data dengan vektor eigen yang dipilih:

$$Z = X \cdot V_k$$

Keterangan :
- Z : Data dalam ruang dimensi rendah
- X : Nilai yang telah distandarisasi
- V_k (Metode matriks kovarians) : Matriks yang berisi k vektor eigen pertama
- V_k (Metode SVD) : submatriks yang terdiri dari k kolom pertama dari V (vektor singular kanan)

Data juga dapat diproyeksikan dengan seperti ini (khusus SVD):

$$Z = U\sum_k$$

Keterangan :
- U : Matriks singular kiri
- $\sum_k$: Submatriks dari matriks singular kanan

G. Metrik Evaluasi

Untuk menguji apakah informasi yang disimpan setelah melakukan PCA masih terjaga atau tidak, diperlukan percobaan prediksi menggunakan *machine learning*. Metrik yang digunakan untuk evaluasi model adalah MAPE.

Mean Absolute Percentage Error (MAPE) adalah metrik evaluasi yang digunakan untuk mengukur kesalahan prediksi dalam model regresi. MAPE mengukur rata-rata kesalahan absolut dalam bentuk persentase terhadap nilai sebenarnya.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

Keterangan :
- y_i : Nilai sebenarnya pada data ke-i,
- $\hat{y}_i$: Prediksi model pada data ke-i,
- n : Jumlah sampel

III. METODE DAN HASIL

A. Persiapan Eksperimen dan Modul

Percobaan ini dilakukan menggunakan bahasa pemrograman Python versi 3.12 Beberapa pustaka (modul) yang digunakan dalam eksperimen adalah sebagai berikut.

- Pandas : Digunakan untuk analisis data
- Matplotlib : Digunakan untuk visualisasi data
- Scikit-Learn : Digunakan untuk pelatihan model
- XGBoost : Digunakan untuk menggunakan model XGBRegressor
- Numpy : Digunakan untuk manipulasi matriks

Untuk dapat melihat secara keseluruhan kode serta hasil yang digunakan dalam makalah ini. Kode dapat diakses pada github

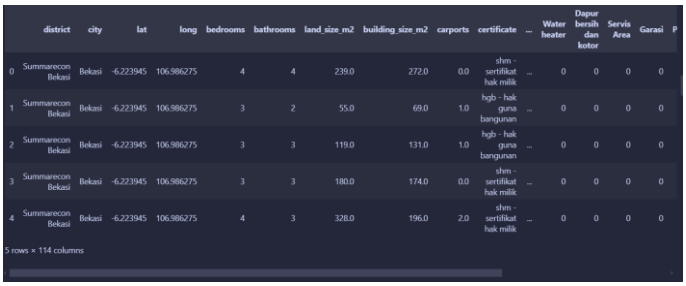
B. Persiapan Dataset

Dalam percobaan ini, akan digunakan data daftar harga rumah di Jabodetabek yang dapat diakses di kaggle. Berikut merupakan ringkasan singkat datasest.

Tabel 3.1 Deskripsi Dataset

Nama	Daftar Harga Rumah Jabodetabek
Sumber Dataset	Kaggle
Jumlah Sample	3553 sampel
Jumlah Fitur	114 kolom (111 numerik, 3 kategori)
Target	fitur numerik pada kolom “price_in_rp”

Perlu digaris bawahi, data yang digunakan telah mengalami pemrosesan agar dapat disesuaikan dengan proses pelatihan model *machine learning* dan implementasi PCA. berikut merupakan tampilan data yang telah dilakukan pemrosesan.



	district	city	lat	long	bedrooms	bathrooms	land_size_m2	building_size_m2	carports	certificate	...	Water heater	Dapur bersih dan kotor	Servis Area	Garasi	P
0	Summarecon Bekasi	Bekasi	-6.223945	106.986275	4	4	239.0	272.0	0.0	shm - sertifikat hak milik	...	0	0	0	0	
1	Summarecon Bekasi	Bekasi	-6.223945	106.986275	3	2	55.0	69.0	1.0	hgb - hak guna bangunan	...	0	0	0	0	
2	Summarecon Bekasi	Bekasi	-6.223945	106.986275	3	3	119.0	131.0	1.0	hgb - hak guna bangunan	...	0	0	0	0	
3	Summarecon Bekasi	Bekasi	-6.223945	106.986275	3	3	180.0	174.0	0.0	shm - sertifikat hak milik	...	0	0	0	0	
4	Summarecon Bekasi	Bekasi	-6.223945	106.986275	4	3	328.0	196.0	2.0	shm - sertifikat hak milik	...	0	0	0	0	

5 rows × 114 columns

Gambar 3.1 Tampilan Sebagian Dataset
Sumber :Olahan Penulis

C. Implementasi PCA menggunakan SVD

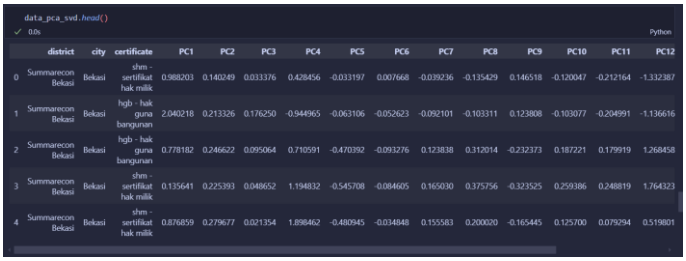
Pada 114 kolom di dataset, 3 di antaranya merupakan tipe kategori sehingga perlu dikecualikan dalam perhitungan SVD. Berikut merupakan kode untuk melakukan PCA dengan metode SVD.

```python
def pca_with_svd(data, n_components=None):
    """

    Melakukan PCA menggunakan Singular Value Decomposition (SVD) pada kolom numerik.

    Parameter:
    - data: pd.DataFrame
        DataFrame input.
    - n_components: int atau None
        Jumlah komponen utama yang akan disimpan. Jika None, maka semua komponen akan disimpan.

    Keluaran:
    - dict
        Sebuah dictionary yang berisi:
        - 'pca_svd': Data yang diproyeksikan menggunakan SVD,
        - 'explained_variance': Rasio varians yang dijelaskan,
        - 'components': Komponen utama (vektor singular kanan).
    """
    start_time = time.time()
    # Memilih kolom numerik
    numerical_data = data.select_dtypes(include=[np.number])
    if numerical_data.empty:
        raise ValueError("No numerical data found for PCA.")

    # Standardisasi data
    standardized_data = (numerical_data - numerical_data.mean()) / numerical_data.std()

    # Dekompisisi SVD
    U, Sigma, Vt = np.linalg.svd(standardized_data, full_matrices=False)

    # Menerapkan jumlah komponen yang diinginkan
    n_components = n_components or standardized_data.shape[1]

    # Proyeksi data ke ruang komponen utama
    pca_svd = U[:, :n_components] @ np.diag(Sigma[:n_components])

    # Menmasukkan PCA ke dalam DataFrame
    pca_column_names = [f"PC{i+1}" for i in range(n_components)]

    # Membuat DataFrame baru
    pca_df = pd.DataFrame(pca_svd, columns=pca_column_names, index=data.index)
    non_numerical_data = data.drop(columns=numerical_data.columns)  # Menyimpan kolom non-numerik

    # Menggabungkan data
    updated_data = pd.concat([non_numerical_data, pca_df], axis=1)
    execution_time = time.time() - start_time

    print(f"PCA with SVD completed in {execution_time:.2f} seconds.")

    return updated_data
```

Gambar 3.2 PCA dengan SVD
Sumber : Olahan Penulis

Kode ini mengacu pada algoritma PCA dengan SVD pada landasan teori yang menjelaskan mengenai algoritma PCA (baca II.F). Pada percobaan ini, hasil dekomposisi akan mereduksi 111 numerik kolom menjadi 15 kolom.

Kode dimulai dengan memilih fitur yang bertipe numerik terlebih dahulu. Lalu, data distandarisasi agar nilai semua fitur memiliki skala yang sama dan tidak ada fitur yang mendominasi. Setelah itu, data didekomposisi menggunakan SVD. Terkahir,data yang telah didekomposisi diproyeksikan ulang ke komponen utama. Berikut hasil PCA menggunakan SVD.



`data_pca_svd.head()` — 0.0s — Python

	district	city	certificate	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC12
0	Summarecon Bekasi	Bekasi	shm - sertifikat hak milik	0.988203	0.140249	0.033376	0.428456	-0.033197	0.007668	-0.039236	-0.135429	0.146518	-0.120047	-0.212164	-1.332387
1	Summarecon Bekasi	Bekasi	hgb - hak guna bangunan	2.040218	0.213326	0.176250	-0.944965	-0.063106	-0.052623	-0.092101	-0.103311	0.123808	-0.103077	-0.204991	-1.136616
2	Summarecon Bekasi	Bekasi	hgb - hak guna bangunan	0.778182	0.246622	0.095064	0.710591	-0.470392	-0.093276	0.123838	0.312014	-0.232373	0.187221	0.179919	1.268458
3	Summarecon Bekasi	Bekasi	shm - sertifikat hak milik	0.135641	0.225393	0.048652	1.194832	-0.545708	-0.084605	0.165030	0.375756	-0.323525	0.259386	0.248819	1.764323
4	Summarecon Bekasi	Bekasi	shm - sertifikat hak milik	0.876859	0.279677	0.021354	1.898462	-0.480945	-0.034848	0.155583	0.200020	-0.165445	0.125700	0.079294	0.519801

Gambar 3.3 Hasil PCA dengan SVD
Sumber : Olahan Penulis

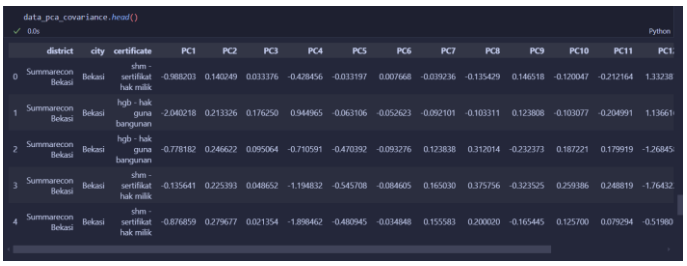
Nilai pada data hasil PCA merepresentasikan posisi data dalam ruang komponen utama. Komponen utama menangkap variansi terbesar dalam data, dengan masing-masing nilai mencerminkan kontribusi data terhadap arah variansi tersebut.

D. Implementasi PCA menggunakan Matriks Kovarians

```python
def pca_with_covariance(data, n_components=None):
    """
    Melakukan PCA menggunakan matriks kovariansi pada kolom numerik.

    Parameter:
    - data: pd.DataFrame
        DataFrame input.
    - n_components: int atau None
        Jumlah komponen utama yang akan disimpan. Jika None, maka semua komponen akan disimpan.

    Keluaran:
    - pd.DataFrame
        DataFrame yang diperbarui dengan data numerik digantikan oleh komponen utama (PCA).
    """
    start_time = time.time()
    # Memilih kolom numerik
    numerical_data = data.select_dtypes(include=[np.number])
    if numerical_data.empty:
        raise ValueError("No numerical data found for PCA.")

    # Standardisasi data
    standardized_data = (numerical_data - numerical_data.mean()) / numerical_data.std()

    # Membuat matriks kovariansi
    covariance_matrix = np.cov(standardized_data.T)

    # Menghitung eigenvalues dan eigenvectors
    eigenvalues, eigenvectors = np.linalg.eig(covariance_matrix)
    eigenvalues = np.real(eigenvalues)
    eigenvectors = np.real(eigenvectors)

    # Mengurutkan eigenvalues dan eigenvectors
    sorted_indices = np.argsort(eigenvalues)[::-1]
    eigenvalues = eigenvalues[sorted_indices]
    eigenvectors = eigenvectors[:, sorted_indices]

    # Menentukan jumlah komponen yang diinginkan
    n_components = n_components or standardized_data.shape[1]

    # Memilih n komponen teratas
    selected_eigenvectors = eigenvectors[:, :n_components]

    # Proyeksi data ke ruang komponen utama
    pca_covariance = standardized_data @ selected_eigenvectors

    # Membuat nama kolom untuk komponen utama
    pca_column_names = [f"PC{i+1}" for i in range(n_components)]

    # Membuat DataFrame baru untuk komponen utama
    pca_df = pd.DataFrame(pca_covariance, index=data.index)
    pca_df.columns = pca_column_names

    # Menyimpan kolom non-numerik
    non_numerical_data = data.drop(columns=numerical_data.columns)

    # Menggabungkan kembali data non-numerik dan komponen utama
    updated_data = pd.concat([non_numerical_data, pca_df], axis=1)
    execution_time = time.time() - start_time

    print(f"PCA with covariance completed in {execution_time:.2f} seconds.")

    return updated_data
```

Gambar 3.4 PCA dengan Matriks Kovarians
Sumber : Olahan Penulis

Sama seperti pada metode SVD, kode ini mengacu pada algoritma PCA dengan SVD pada landasan teori yang menjelaskan mengenai algoritma PCA (baca II.F) dan juga mereduksi 111numerik kolom menjadi 15 kolom. Berikut hasil PCA menggunakan Matriks Kovarians.



`data_pca_covariance.head()` — 0.0s — Python

	district	city	certificate	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC1
0	Summarecon Bekasi	Bekasi	shm - sertifikat hak milik	-0.988203	0.140249	0.033376	-0.428456	-0.033197	0.007668	-0.039236	-0.135429	0.146518	-0.120047	-0.212164	1.33238
1	Summarecon Bekasi	Bekasi	hgb - hak guna bangunan	-2.040218	0.213326	0.176250	0.944965	-0.063106	-0.052623	-0.092101	-0.103311	0.123808	-0.103077	-0.204991	1.13661
2	Summarecon Bekasi	Bekasi	hgb - hak guna bangunan	-0.778182	0.246622	0.095064	-0.710591	-0.470392	-0.093276	0.123838	0.312014	-0.232373	0.187221	0.179919	-1.26845
3	Summarecon Bekasi	Bekasi	shm - sertifikat hak milik	-0.135641	0.225393	0.048652	-1.194832	-0.545708	-0.084605	0.165030	0.375756	-0.323525	0.259386	0.248819	-1.76432
4	Summarecon Bekasi	Bekasi	shm - sertifikat hak milik	-0.876859	0.279677	0.021354	-1.898462	-0.480945	-0.034848	0.155583	0.200020	-0.165445	0.125700	0.079294	-0.51980

Gambar 3.5 Hasil PCA dengan Matriks Kovarians
Sumber : Olahan Penulis

Untuk membandingkan kompleksitas kedua algoritma, dilakukan *benchmarking* proses PCA menggunakan kedua metode tersebut. Berikut merupakan hasil perbandingan kedua algoritma.

```
data_pca_svd = pca_with_svd(data_pipe, n_components=15)
data_pca_covariance = pca_with_covariance(data_pipe, n_components=15)
✓ 0.1s

PCA with SVD completed in 0.10 seconds.
PCA with covariance completed in 0.05 seconds.
```

Gambar 3.6 Waktu Eksekusi
Sumber : Olahan Penulis

Ternyata terlihat bahwa PCA menggunakan SVD lebih lambat dua kali lipat dibandingkan dengan menggunakan Matriks Kovarians. Hal ini akan dijelaskan lebih lanjut pada bagian pembahasan.

Selain itu, pada kedua jenis metode, nilai yang dihasilkan sama tetapi terdapat perbedaan tanda pada beberapa kolom komponen utama. Hal ini juga akan dijelaskan lebih lanjut pada bagian pembahasan.

E. Pelatihan dan Evaluasi Model

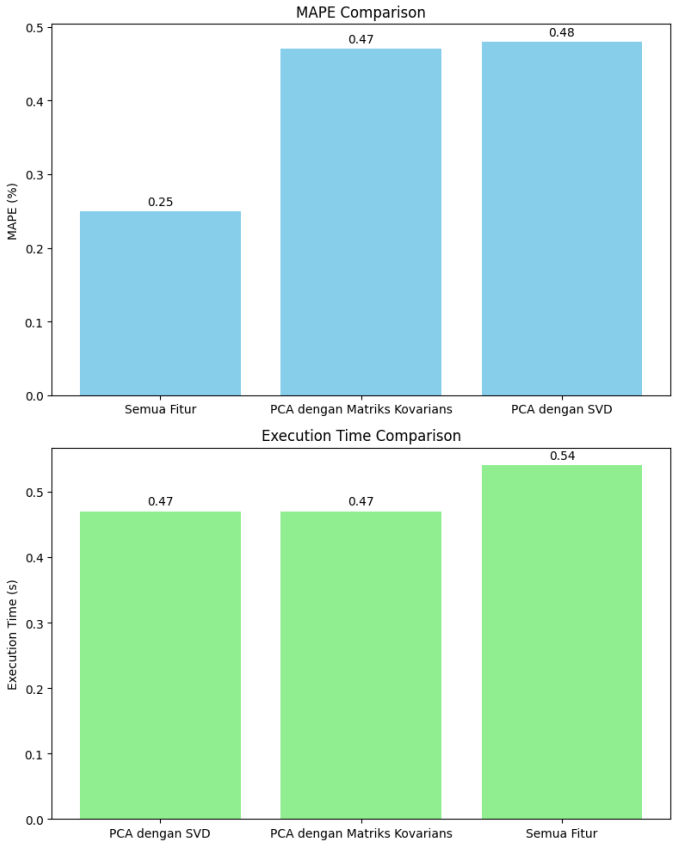
Percobaan ini akan menggunakan salah satu model dengan jenis *ensemble learning* berbasis tree (*Gradient Boosting Decision Tree*) dengan nama XGBoost. Penggunaan XGBoost pada percobaan ini karena model ini memiliki efisiensi yang lebih baik dan cepat dalam melakukan pelatihan. Kelas yang menjadi target pelatihan berjenis numerik, maka dari itu akan digunakan XGBRegressor sebagai model latih dan prediksi.

```python
# Semua fitur
cat_columns = X_train.select_dtypes(include='category').columns.tolist()
start = time.time()
model = XGBRegressor(random_state=42, n_jobs=-1, enable_categorical=True)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
mape = mean_absolute_percentage_error(y_test, y_pred)
print(f"MAPE (semua fitur): {mape:.2f}%")
print(f"Execution time (semua fitur): {time.time() - start:.2f} seconds")

# PCA dengan SVD
start = time.time()
model = XGBRegressor(random_state=42, n_jobs=-1, enable_categorical=True)
model.fit(svd_train, y_train)
y_pred = model.predict(svd_test)
mape = mean_absolute_percentage_error(y_test, y_pred)
print(f"MAPE (PCA dengan SVD): {mape:.2f}%")
print(f"Execution time (PCA dengan SVD): {time.time() - start:.2f} seconds")

# PCA dengan matriks kovariansi
start = time.time()
model = XGBRegressor(random_state=42, n_jobs=-1, enable_categorical=True)
model.fit(cov_train, y_train)
y_pred = model.predict(cov_test)
mape = mean_absolute_percentage_error(y_test, y_pred)
print(f"MAPE (PCA dengan matriks kovariansi): {mape:.2f}%")
print(f"Execution time (PCA dengan matriks kovariansi): {time.time() - start:.2f} seconds")
```

Gambar 3.7 Kode Pelatihan XGB
Sumber : Olahan Penulis

Model akan dilatih dengan 3 metode, yakni dengan (1) data yang tidak dilakukan PCA, (2) data yang telah direduksi oleh PCA metode SVD, dan (3) data yang telah direduksi oleh PCA metode.

Parameter yang digunakan oleh model mencakup "random_state = 42" agar model dapat diproduksi ulang tanpa mengubah hasil, "n_jobs = -1" agar model dapat menggunakan seluruh *core* CPU dalam melakukan pelatihan, dan "enable_categorical = True" agar model dapat melatih data kategori. Dalam hal ini data kategori tersebut berupa fitur "district", "city", "certificate". Selanjutnya akan ditampilkan evaluasi dari model.



Gambar 3.8 MAPE dan Execution Time
Sumber : Olahan Penulis

Pelatihan model menggunakan seluruh fitur menghasilkan skor MAPE yang paling rendah (semakin rendah semakin baik). Namun, waktu eksekusi yang digunakan dalam melakukan latih dan prediksi paling lambat dibandingkan dengan kedua metode lainnya.

Terlihat pula evaluasi model menggunakan metode PCA. Hasil evaluasi tersebut menunjukkan skor yang lebih buruk dibandingkan tidak dilakukan PCA. Secara metode, PCA menggunakan Matriks Kovarians dan PCA menggunakan SVD tidak memberikan perbedaan sama sekali pada skor MAPE maupun waktu eksekusi.

IV. PEMBAHASAN

A. Pengaruh PCA terhadap Akurasi Model

Reduksi dimensi menggunakan PCA berhasil mengurangi jumlah fitur dari 111 numerik hanya menjadi 15 fitur numerik. Namun, hasil evaluasi menunjukkan bahwa penggunaan data yang telah direduksi menghasilkan skor MAPE yang lebih tinggi. Hal ini menandakan bahwa, meskipun PCA mampu menangkap variansi terbesar dalam data, akan ada beberapa informasi-informasi yang hilang selama proses reduksi. Ini menjadi salah satu *trade off* dalam penggunaan PCA untuk menghasilkan prediksi.

B. Perbandingan Eksekusi Waktu Proses PCA

PCA dengan SVD lebih lambat dua kali lipat dibandingkan PCA menggunakan Matriks Kovarians. Hal ini disebabkan oleh kompleksitas algoritma dan jumlah operasi yang terjadi dalam

masing-masing metode.

Ketika menggunakan metode SVD, matriks data asli akan dipecah menjadi 3 matriks , yaitu matriks U, $\sum$, dan $V^{\top}$. Proses dekomposisi ini memerlukan operasi yang sangat besar. Jika dihitung kompleksitasnya, kompleksitas SVD adalah $O(min(m,n) \times m \times n)$ dengan m adalah jumlah sampel dan n adalah jumlah fitur. Dibandingkan dengan kompleksitas matriks kovarians yang hanya $O(n^2 \times m)$ yang jauh lebih kecil dibandingkan dengan kompleksitas SVD.

C. Perbedaan Tanda Hasil PCA

Perbedaan tanda pada hasil PCA yang dilakukan pada metode PCA dan Matriks Kovarians disebabkan oleh sifat vektor eigen dan vektor singular yang tidak memiliki orientasi tanda unik. Dengan kata lain, vektor eigen dapat berupa negatif atau positif dan kedua hasil tersebut tetap valid secara matematis.

Perbedaan tanda ini tidak memengaruhi hasil PCA karena PCA bertujuan untuk menangkap variansi terbesar dalam data, dan tanda komponen tidak memengaruhi arah atau jumlah variansi yang dijelaskan. Selain itu, proyeksi data ke ruang komponen utama tetap valid terlepas dari orientasi tanda.

D. Pengaruh PCA terhadap Waktu Eksekusi Model

Pada percobaan, waktu eksekusi model dalam melakukan pelatihan dan prediksi menggunakan data yang telah direduksi jauh lebih cepat dibandingkan dengan data asli. Hal ini karena fitur yang direduksi membuat kompleksitas komputasi model dalam melihat pola dapat dikurangi, sehingga waktu pelatihan menjadi lebih singkat. PCA sangat bermanfaat pada situasi jika dataset memiliki fitur yang sangat besar atau terdapat beberapa fitur yang memiliki korelasi tinggi. Namun, walaupun eksekusi lebih cepat, perlu diperhatikan mengenai informasi yang hilang ketika melakukan reduksi dimensi data.

V. KESIMPULAN

Percobaan ini menunjukkan bahwa implementasi PCA, baik menggunakan pendekatan SVD atau Matriks Kovarians, mampu menurunkan dimensi data secara efektif dan menghasilkan proses komputasi model *machine learning* yang lebih cepat. Dalam hal kecepatan komputasi, PCA berbasis Matriks Kovarians menunjukkan performa yang lebih unggul 2 kali lebih cepat dibandingkan dengan PCA berbasis SVD. Meski demikian, walaupun PCA berhasil mempercepat proses latih model sebesar 12,9%, hasil evaluasi mengindikasikan bahwa pengurangan dimensi ini mengakibatkan peningkatan kesalahan dalam prediksi sebesar 0,22% jika dibandingkan dengan penggunaan seluruh fitur asli.

Lebih lanjut, walaupun ditemukan variasi tanda pada komponen utama yang dihasilkan oleh kedua metode PCA tersebut, perbedaan ini tidak berdampak signifikan terhadap hasil analisis, mengingat karakteristik matematis dari vektor eigen dan vektor singular. PCA masih merupakan metode yang efektif untuk menangani permasalahan dimensi tinggi pada dataset yang besar, khususnya ketika waktu pemrosesan menjadi pertimbangan utama. Namun, perlu adanya pertimbangan yang cermat antara efisiensi waktu dan potensi hilangnya informasi untuk setiap kasus penerapan tertentu.

REFERENSI

[1] D. C. Lay, *Linear Algebra and Its Applications*. Pearson, 2012.

[2] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016.

[3] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed. New York, NY, USA: Springer, 2002.

[4] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. Baltimore, MD, USA: Johns Hopkins University Press, 2013.

[5] Scikit-learn Developers, "Principal Component Analysis (PCA)," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/decomposition.html#pca. [Accessed: Dec. 2024].

[6] R. Munir, "Aljabar Geometri," [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/algeo.htm. [Accessed: Jan. 1, 2025].

[7] Ryonlunar, "Makalah Algeo," GitHub repository. [Online]. Available: https://github.com/ryonlunar/makalah-algeo/tree/main. [Accessed: Jan. 2, 2025].

[8] "Daftar Harga Rumah Jabodetabek," Kaggle. [Online]. Available: https://www.kaggle.com/datasets/nafisbarizki/daftar-harga-rumah-jabodetabek. [Accessed: Dec. 2024].

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 26 Desember 2024

Adhimas Aryo Bimo dan 13523052