You:

Implement prinicpal component analysis from scratch, that can be used with either eigen decomposition and singular value decomposition.

ChatGPT:

Implementing Principal Component Analysis (PCA) from scratch involves a few steps, and it can be implemented using either Eigen Decomposition or Singular Value Decomposition (SVD). I'll guide you through both methods.

1. Standardize the Data

First, we standardize the dataset. Standardization involves subtracting the mean and dividing by the standard deviation for each feature.

2. Computing the Covariance Matrix (for Eigen Decomposition)

The next step is to compute the covariance matrix, which is used in the eigen decomposition method.

3. Eigen Decomposition Method

- Calculate the eigenvalues and eigenvectors of the covariance matrix.
- Sort the eigenvectors by decreasing eigenvalues and choose the top kk eigenvectors.

4. Singular Value Decomposition (SVD) Method

Alternatively, PCA can be performed using SVD:

- Compute the SVD of the data matrix.
- The principal components are given by the first kk right singular vectors.

5. Project Data onto Principal Components

Finally, project the data onto the selected principal components.

Here's the implementation in Python:

```
python
import numpy as np

def standardize_data(X):
    return (X - np.mean(X, axis=0)) / np.std(X, axis=0)

def pca_eigen(X, num_components):
    # Standardize the data
    X_std = standardize_data(X)
    # Covariance matrix
    covariance_matrix = np.cov(X_std.T)
```

```
    # Eigen decomposition
    eigenvalues, eigenvectors = np.linalg.eig(covariance_matrix)
    # Sort eigenvectors by eigenvalues in descending order
    idx = eigenvalues.argsort()[::-1]
    eigenvalues = eigenvalues[idx]
    eigenvectors = eigenvectors[:,idx]
    # Select top num_components eigenvectors
    eigenvectors = eigenvectors[:, :num_components]
    return X_std.dot(eigenvectors)

def pca_svd(X, num_components):
    # Standardize the data
    X_std = standardize_data(X)
    # Compute SVD
    U, S, Vt = np.linalg.svd(X_std)
    # Select top num_components eigenvectors (right singular vectors)
    V = Vt.T
    return X_std.dot(V[:, :num_components])

# Example usage
X = np.array([[1, 2], [3, 4], [5, 6]])  # Replace with your dataset
num_components = 1  # Number of components you want to keep

# Using Eigen Decomposition
pca_eigen_results = pca_eigen(X, num_components)

# Using SVD
pca_svd_results = pca_svd(X, num_components)
```

Replace X with your dataset and `num_components` with the number of principal components you wish to keep. This code will give you the PCA transformed data using both methods.
