You:

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

ChatGPT:

Principal Component Analysis (PCA) is a technique used for dimensionality reduction and feature extraction. You can implement PCA from scratch using either eigen decomposition or singular value decomposition (SVD). I'll provide a Python code example for both methods.

First, let's import the necessary libraries:

```
python
import numpy as np
```

PCA using Eigen Decomposition

Here's a PCA implementation using eigen decomposition:

```
python
def pca_eigen_decomp(X, num_components):
    # Center the data
    mean = np.mean(X, axis=0)
    centered_X = X - mean

    # Calculate the covariance matrix
    cov_matrix = np.cov(centered_X, rowvar=False)

    # Eigen decomposition
    eigenvalues, eigenvectors = np.linalg.eig(cov_matrix)

    # Sort eigenvalues and corresponding eigenvectors
    sorted_indices = np.argsort(eigenvalues)[::-1]
    eigenvalues = eigenvalues[sorted_indices]
    eigenvectors = eigenvectors[:, sorted_indices]

    # Select the top 'num_components' eigenvectors
    top_eigenvectors = eigenvectors[:, :num_components]

    # Project the data onto the selected eigenvectors
    pca_result = np.dot(centered_X, top_eigenvectors)

    return pca_result
```

PCA using Singular Value Decomposition (SVD)

Here's a PCA implementation using SVD:

```
python
def pca_svd(X, num_components):
    # Center the data
    mean = np.mean(X, axis=0)
    centered_X = X - mean

    # Perform SVD
    U, S, Vt = np.linalg.svd(centered_X, full_matrices=False)

    # Select the top 'num_components' singular values and corresponding columns of U
    U_reduced = U[:, :num_components]

    # Project the data onto the selected components
    pca_result = np.dot(centered_X, U_reduced)

    return pca_result
```

You can use either of these functions by passing your data matrix X and the number of principal components you want to retain `num_components`. These functions will return the PCA-transformed data. For example:

```
python
# Generate some sample data
data = np.random.rand(100, 3)

# Perform PCA using eigen decomposition or SVD
num_components = 2
pca_result_eigen = pca_eigen_decomp(data, num_components)
pca_result_svd = pca_svd(data, num_components)

print("PCA Result (Eigen Decomposition):\n", pca_result_eigen)
print("PCA Result (SVD):\n", pca_result_svd)
```

Make sure to choose the appropriate method based on your needs (eigen decomposition or SVD).
