Image Reconstruction

PCA Analysis - Part A

Definition

Principal Component Analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

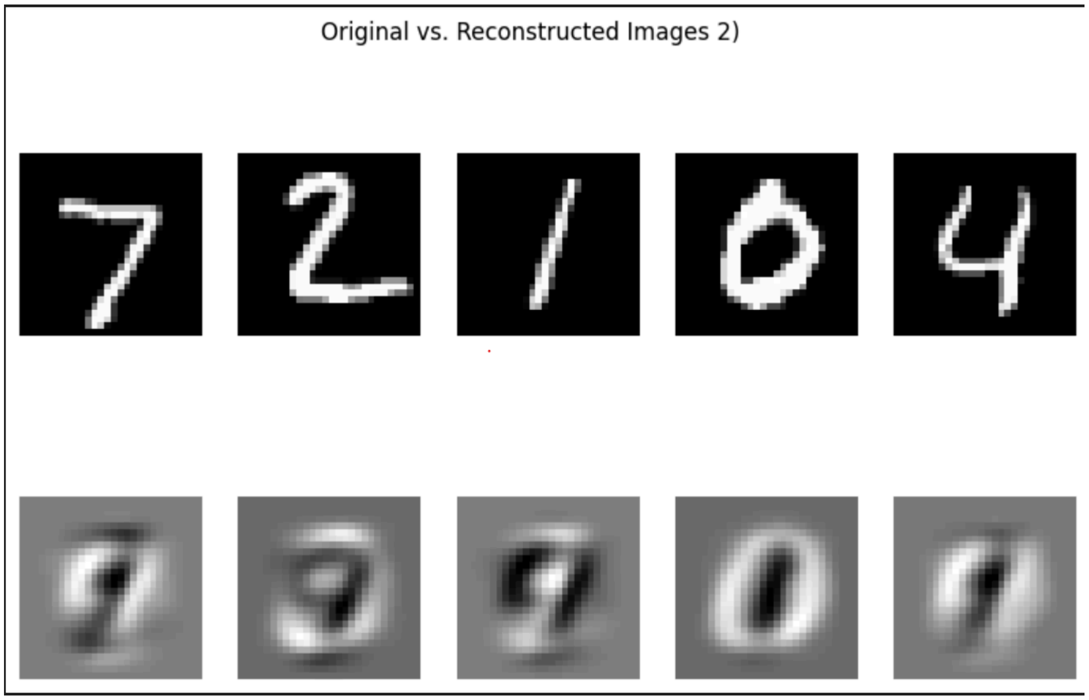
Following are the steps taken to get it:
1. Get the vectorized data as list of numpy arrays (each array is 1d).
2. Get the covariance matrix of the training list of vectors.
3. Get eigenvalues and eigen vectors using the above covariance matrix.
4. Sort the eigenvectors based on 'k' value and get a list for each dimentsion in component list.
5. get dot product of top eigenvector (for a dimension of latent variable) with training data.
6. get dot product of above matrix with transpose of top eigenvector (reconstructed data).
7. get the mean_squared_error from above two variables (training data & reconstructed data).
8. Top eigenvector list can be used to reconstruct the image as given below.

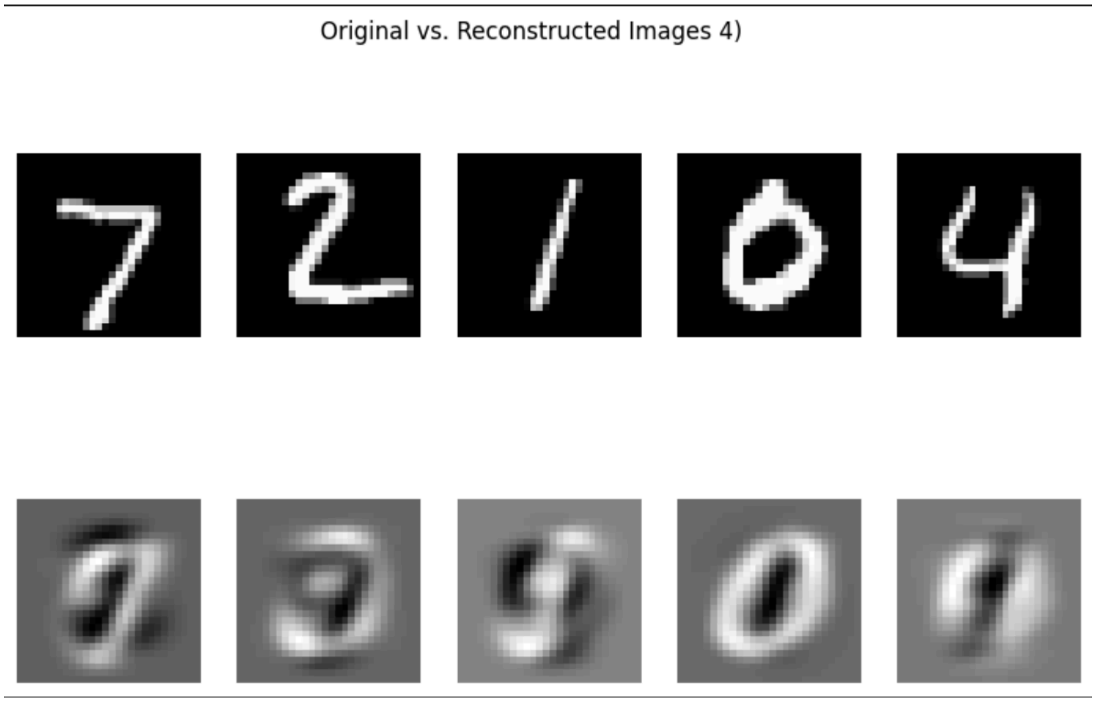
Mean Squared Errors

Following are the mean errors got from 6 models:

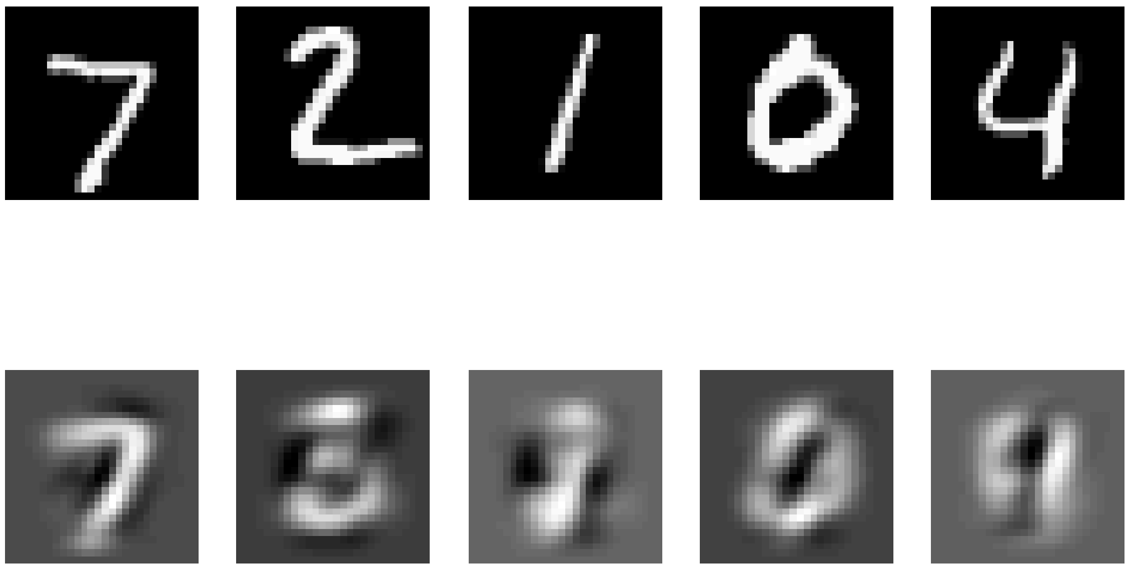
1. Performing PCA reconstruction with 2 principal components... MSE: 0.089463
2. Performing PCA reconstruction with 4 principal components... MSE: 0.076454
3. Performing PCA reconstruction with 8 principal components... MSE: 0.044232
4. Performing PCA reconstruction with 16 principal components... MSE: 0.028734
5. Performing PCA reconstruction with 32 principal components... MSE: 0.017572
6. Performing PCA reconstruction with 64 principal components... MSE: 0.009391

Image Reconstruction examples

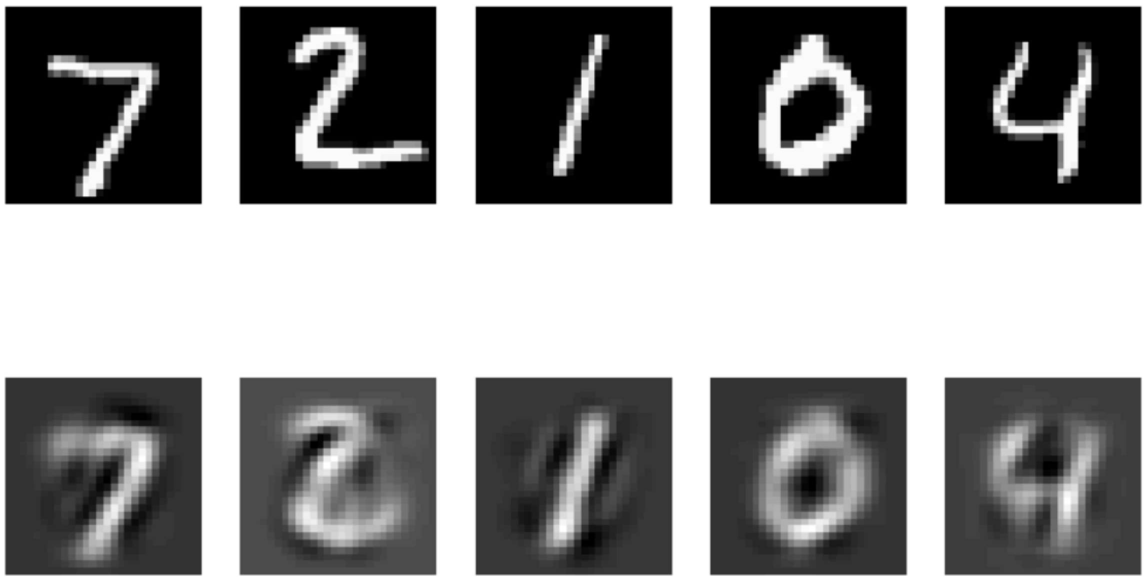




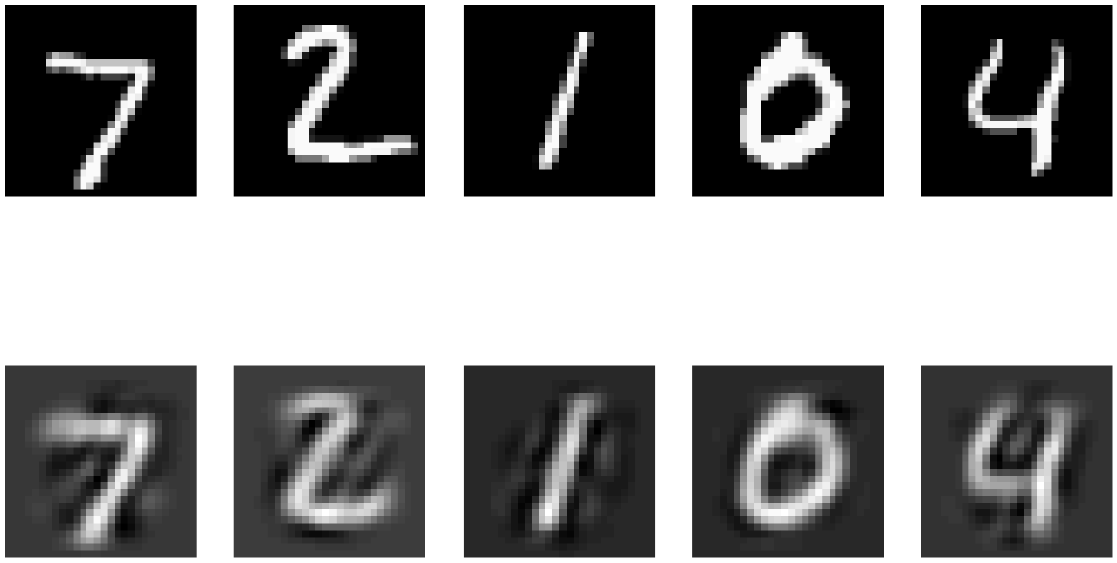
Original vs. Reconstructed Images 8)



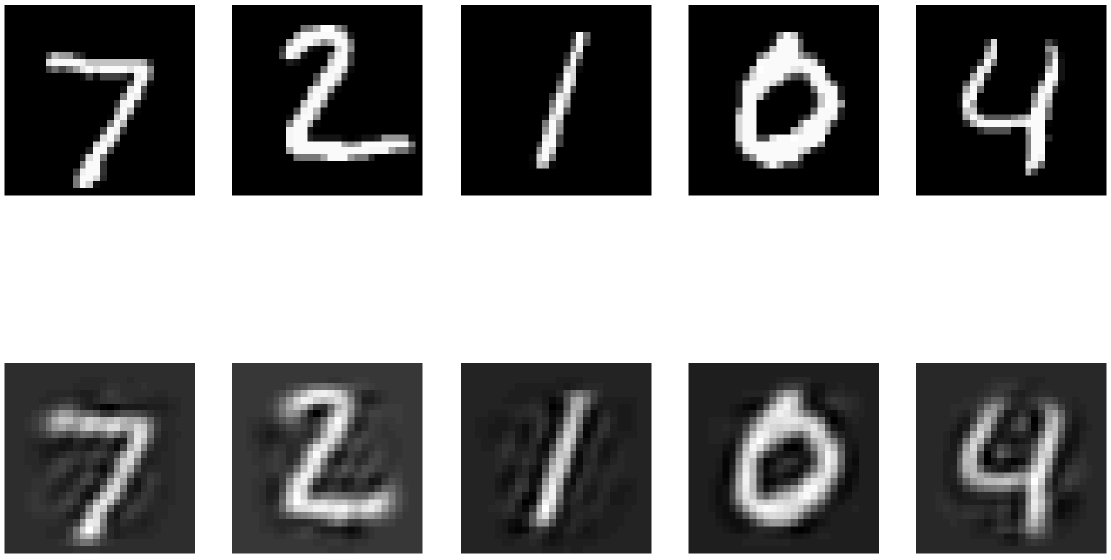
Original vs. Reconstructed Images 16)



Original vs. Reconstructed Images 32)



Original vs. Reconstructed Images 64)



Probabilistic PCA Analysis - Part B

Definition

Probabilistic PCA is a dimensionality reduction technique that models the data as a sample from a Gaussian distribution. It is a probabilistic model that is closely related to PCA, but adds a latent variable to the model. The latent variable is a random variable that is not observed, but is inferred from the observed data.

Following are the steps taken to get it

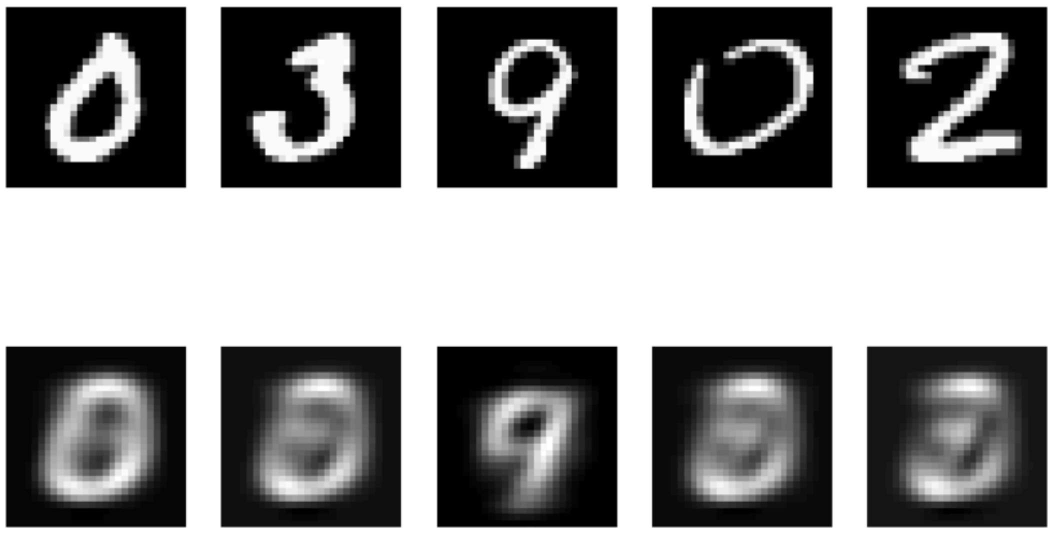
1. Get the vectorized data as list of numpy arrays (each array is 1d).
2. Center the data by subtracting the mean from each data point.
3. Get the covariance matrix on the centered data.
4. Get eigenvalues and eigen vectors using the above covariance matrix.
5. Sort the eigenvectors based on 'k' value and get a list for each dimentsion in component list.
6. get dot product of top eigenvector (for a dimension of latent variable) with centered data.
7. get dot product of above matrix with transpose of top eigenvector and add the mean to it (reconstructed data).
8. get the mean_squared_error from above two variables (training data & reconstructed data).
9. Top eigenvector list can be used to reconstruct the image as given below.

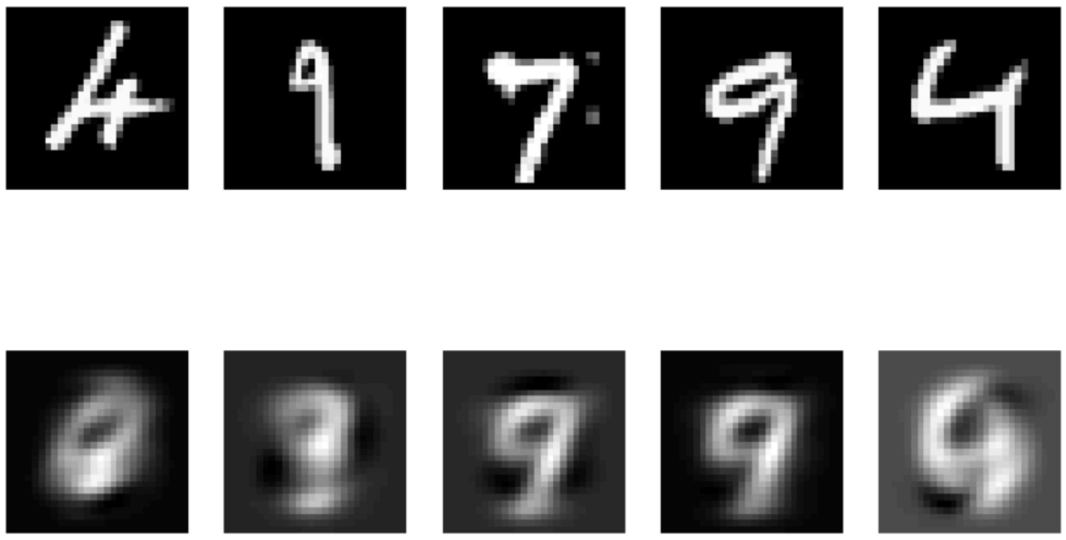
Mean Squared Errors

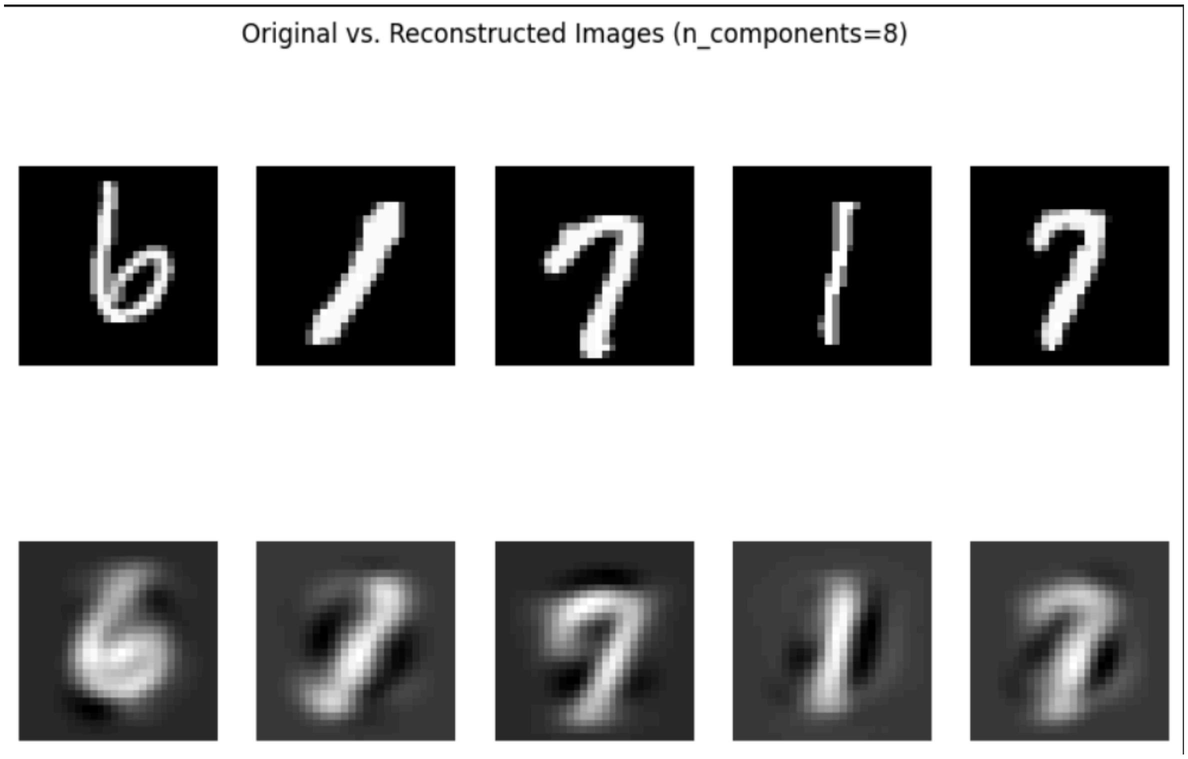
Following are the mean errors got from 6 models:

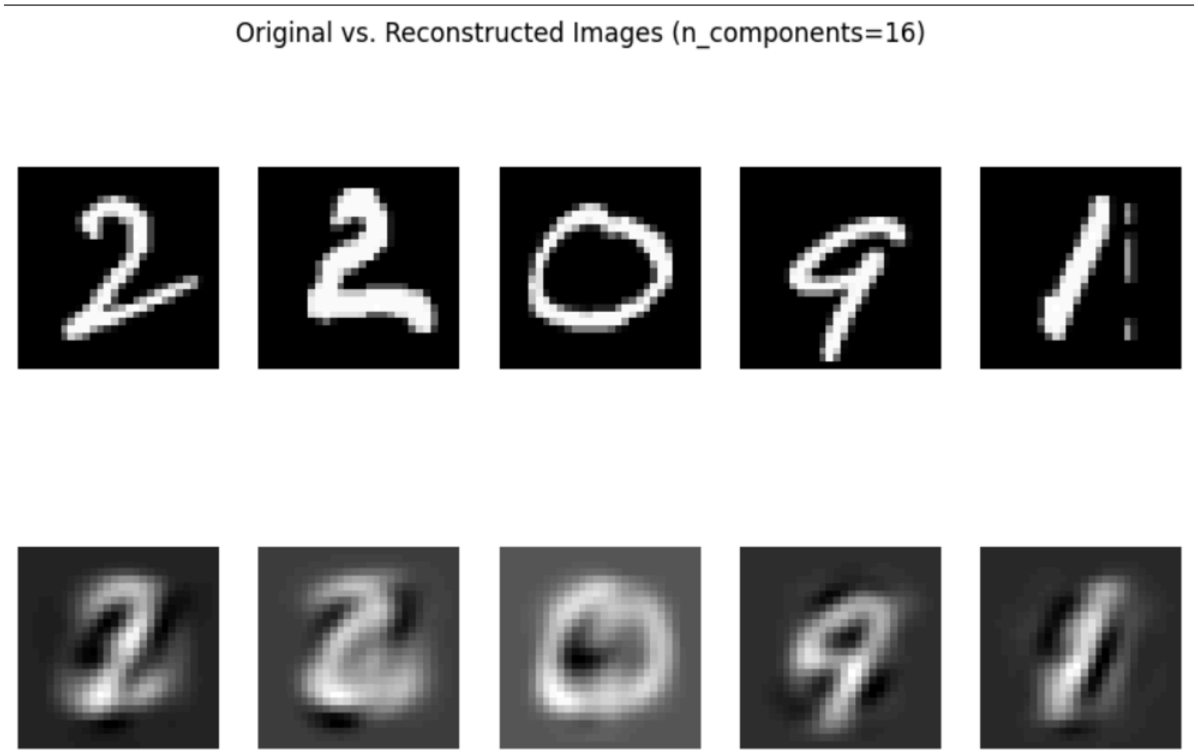
1. Performing Probabilistic PCA reconstruction with 2 principal components... Train MSE: 0.055953
2. Performing Probabilistic PCA reconstruction with 4 principal components... Train MSE: 0.048179
3. Performing Probabilistic PCA reconstruction with 8 principal components... Train MSE: 0.037865
4. Performing Probabilistic PCA reconstruction with 16 principal components... Train MSE: 0.027293
5. Performing Probabilistic PCA reconstruction with 32 principal components... Train MSE: 0.017243
6. Performing Probabilistic PCA reconstruction with 64 principal components... Train MSE: 0.009284

Image Reconstruction examples

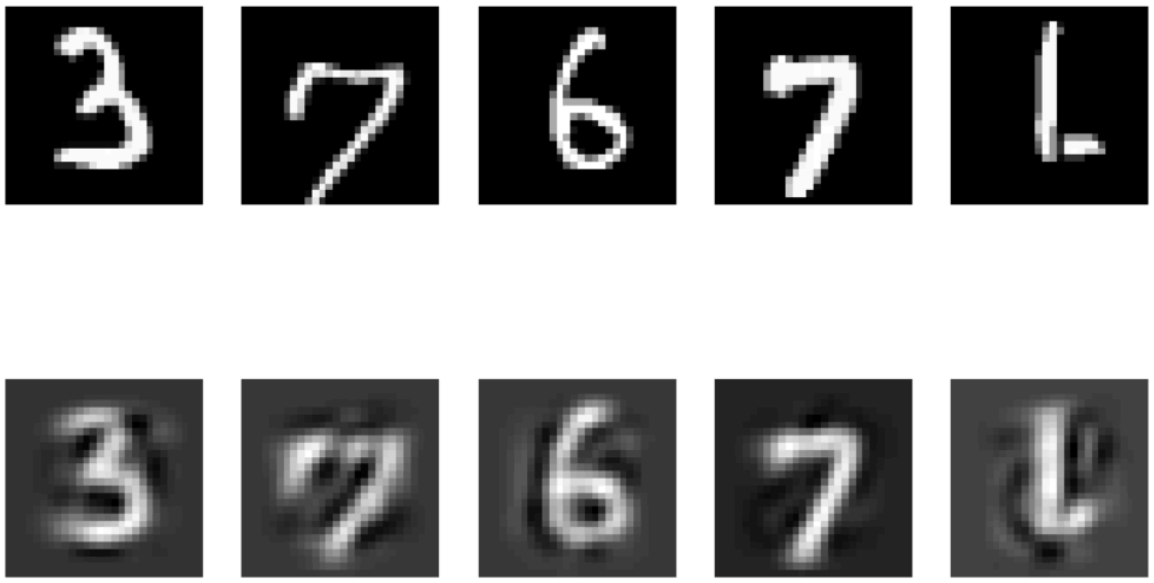

Original vs. Reconstructed Images (n_components=2)


Original vs. Reconstructed Images (n_components=4)

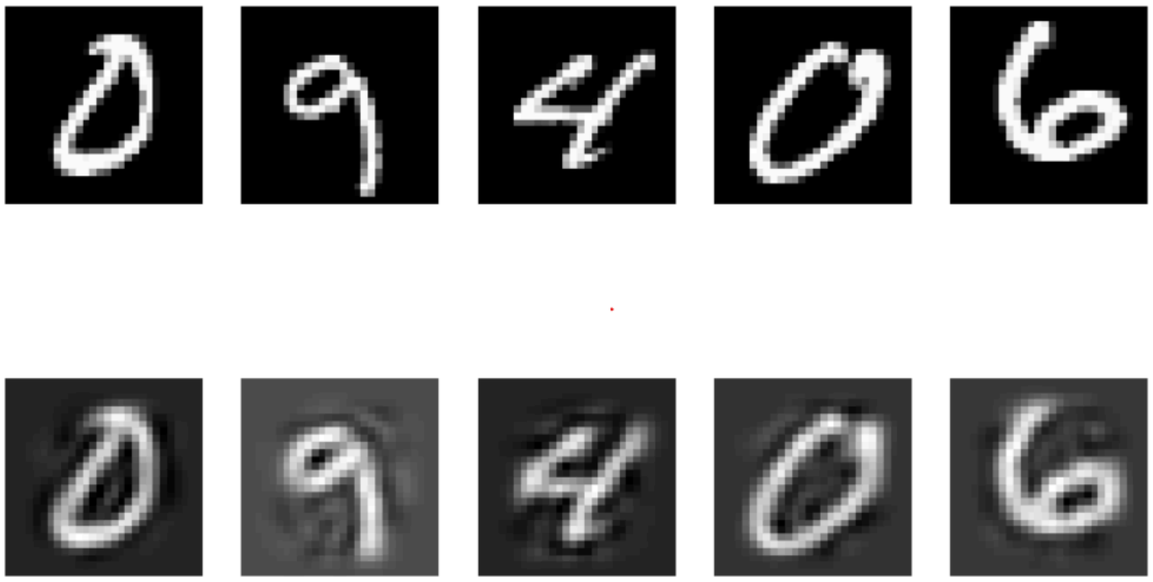

Original vs. Reconstructed Images (n_components=8)


Original vs. Reconstructed Images (n_components=16)

Original vs. Reconstructed Images (n_components=32)



Original vs. Reconstructed Images (n_components=64)



Variational Autoencoders - Part C

Definition

Variational autoencoders (VAEs) are neural networks that learn to encode input data into a compact representation, called a latent space. Unlike traditional autoencoders, VAEs introduce probabilistic elements, enabling them to generate diverse outputs. They find applications in data compression, generation, and anomaly detection in machine learning.

Following are steps to train it:

1. **Data Preparation:** - Load, normalize, and reshape MNIST images.
2. **Encoder Network:** - Design neural network with probabilistic layers for input-to-latent mapping.
3. **Decoder Network:** - Create a mirrored network for latent-to-output mapping.
4. **Loss Function:** - Define loss balancing reconstruction error and latent space divergence.
5. **Training:** - Train VAE on normalized dataset, optimize parameters to minimize defined loss.
6. **Sampling from Latent Space:** - Generate random samples from learned latent space.
7. **Decoding Test Examples:** - Encode MNIST test examples with trained encoders.
8. **Reconstruction:** - Decode latent representations to reconstruct images.
9. **Evaluation:** - Assess reconstruction quality on the test set.

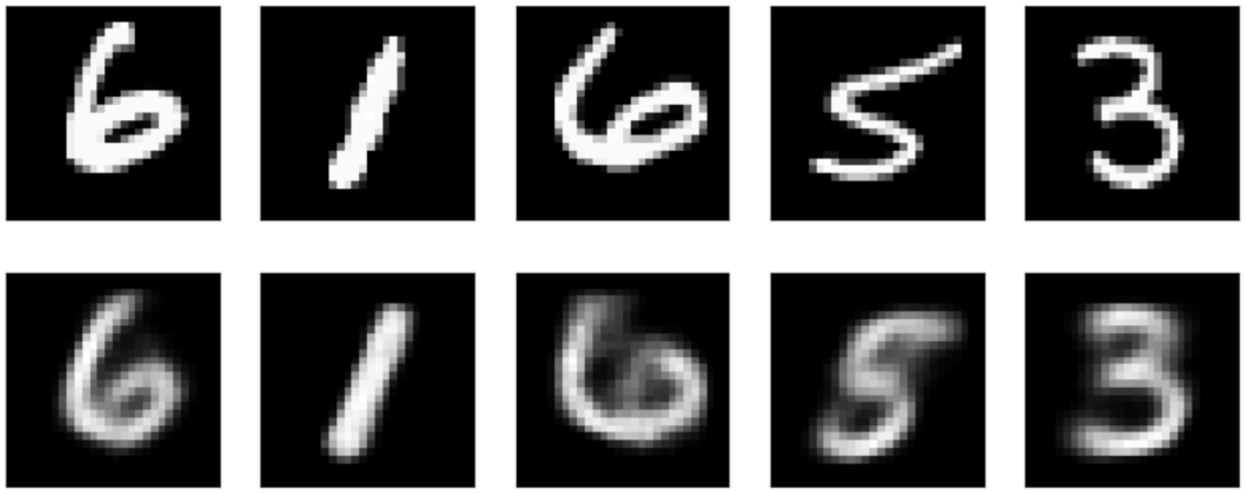
Mean Squared errors

Following are the errors obtained after training same model (with principal component varying) in 10 epochs:

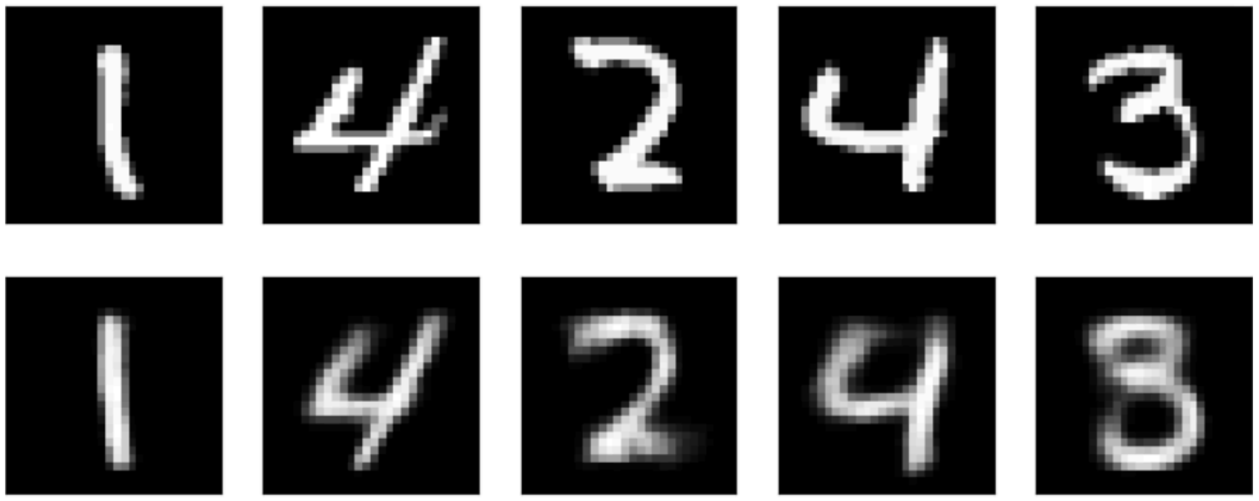
1. Performing Probabilistic PCA reconstruction with 2 principal components... Train MSE: 0.0397
2. Performing Probabilistic PCA reconstruction with 4 principal components... Train MSE: 0.0272
3. Performing Probabilistic PCA reconstruction with 8 principal components... Train MSE: 0.0139
4. Performing Probabilistic PCA reconstruction with 16 principal components... Train MSE: 0.0167
5. Performing Probabilistic PCA reconstruction with 32 principal components... Train MSE: 0.0086
6. Performing Probabilistic PCA reconstruction with 64 principal components... Train MSE: 0.0089

Image Reconstruction examples

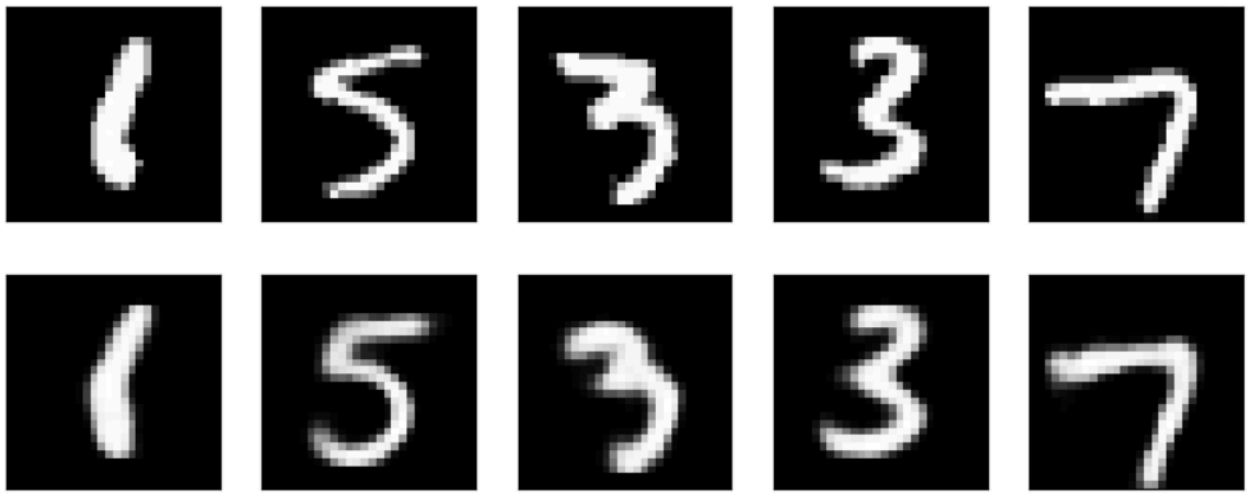
Original vs Reconstructed (latent_space = 2)



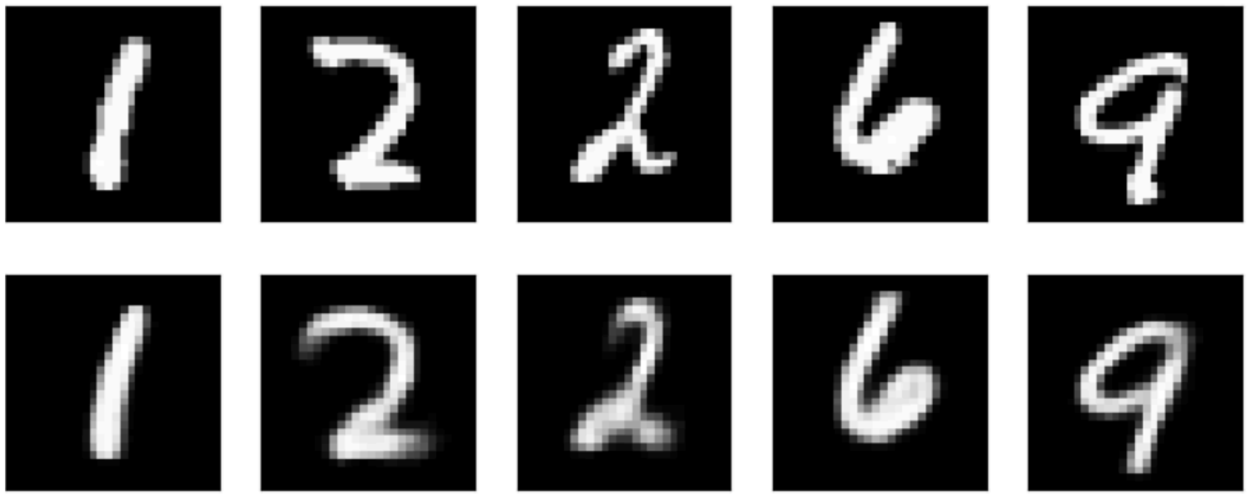
Original vs Reconstructed (latent_space = 4)



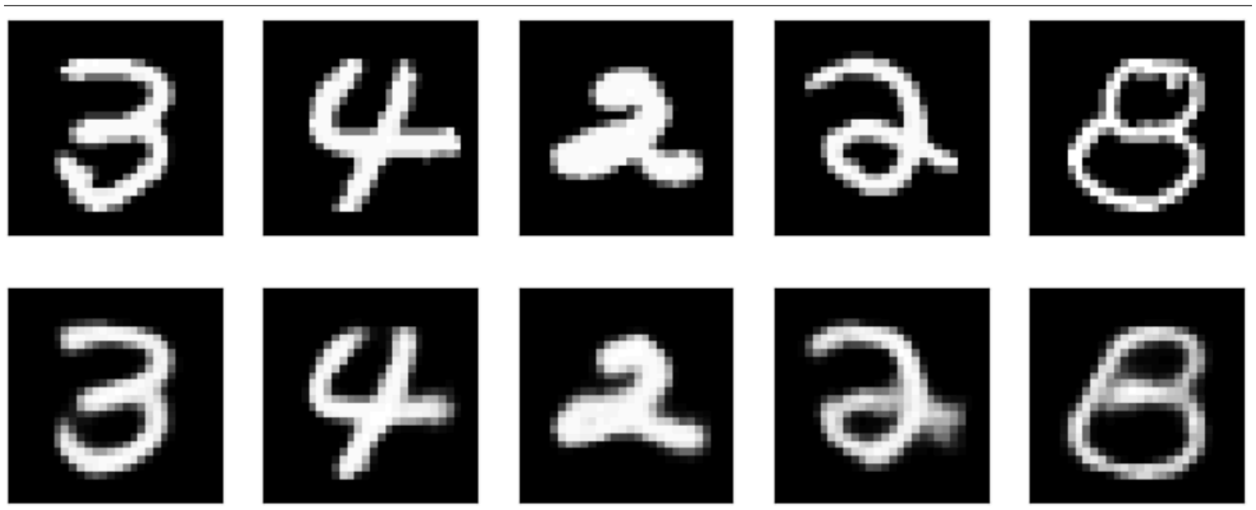
Original vs Reconstructed (latent_space = 8)



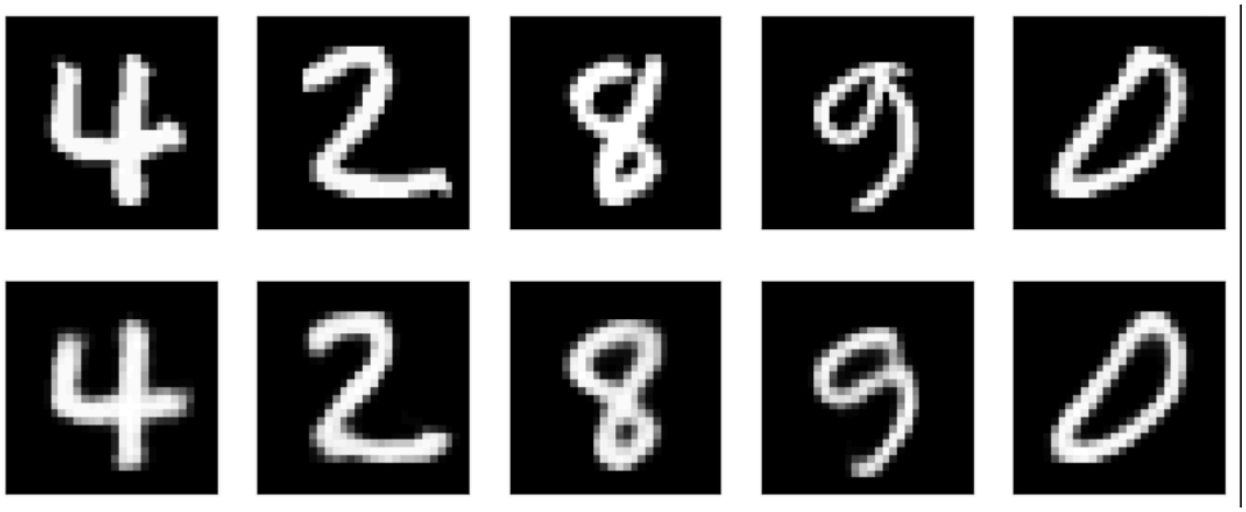
Original vs Reconstructed (latent_space = 16)



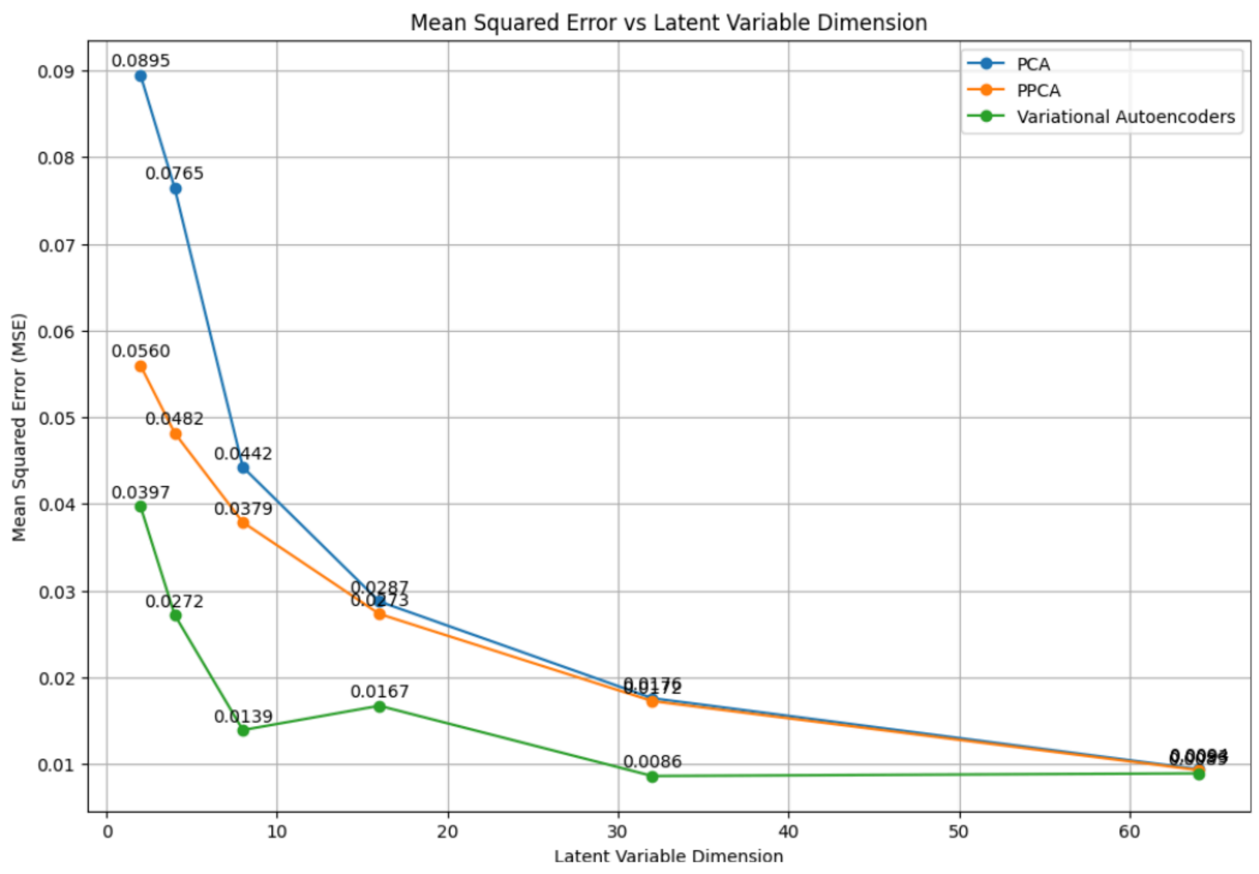
Original vs Reconstructed (latent_space = 32)



Original vs Reconstructed (latent_space = 64)



Graph Plotting - Part D


Mean Squared Error vs Latent Variable Dimension
Mean Squared Error (MSE)
Latent Variable Dimension
PCA
PPCA
Variational Autoencoders