Pattern Recognition

Face Recognition

30th April, 2025

Team

Name	ID
Ahmed Ashraf	21010040
Ahmed Osama	21010037
Saifullah Mousaad	21010651

Data Manipulation

After reading the dataset, a flattening operation is done to produce the vector containing 10304 entries with each representing a pixel in the image.

Principal Components Analysis (PCA)

Approach

The steps to construct the PCA reduced space was as follows:

- Build the covariance matrix for the training set.
- Calculate the eigenvalues and eigenvectors for the covariance matrix.
- Based on alpha value (variance to be retained), save the eigenvectors whose eigenvalues preserve the required variance as the new principal components for the new space.
- Construct matrix “U” representing the principal components (10304, k_components), dot product with the training data matrix with the mean subtracted (denoted as “the empirical mean”) (samples, 10304) ⇒ return projection of the data matrix (samples, k_components).
- Provide functions to project any matrix to the reduced space and to reconstruct it into its original space with the addition of the empirical mean.

Results

The following table represents The number of principal components for each alpha:

Principal Components		
Alpha	Principal Components	Quality
0.80	36	- Training set: Some details of the faces are unrecognizable - Test set: Faces are very bad

Principal Components		
Alpha	Principal Components	Quality
0.85	52	- Training: Some faces became more recognizable - Test: still bad
0.90	76	- Training: Faces are now someway accepted but not yet that good
0.95	117	- Training: Most images became reasonable and look like the original
0.99	174	- Training: Almost all faces are now the same as the original - Test: Faces are still noisy, the difference between test and training sets is huge.

Analysis

- PCA provides a mediocre-quality technique for space reduction to save memory and operating time on the dataset.
- As expected, the idea is all about compromise, PCA saves a huge amount of memory and time with some losses that can go too severe to be used in the required operations. Hence, it's not that preferable.
- With alpha increasing, the amount of variance preserved ⇒ the less the loss is and the better the images are, but as mentioned, this comes with more data to be saved.
- Centering the data is a crucial step to ensure obtaining the highest-variance components and adding the mean ensures that images preserve good quality compared with the original.

- The training samples are reconstructed almost perfectly while the test samples are reconstructed badly as the components and the mean stored are for the training set.

K-Means Clustering

Approach

- Hard K-Means was implemented.
- Initial centroids were selected randomly from the dataset.
- Looping until convergence or reaching max iterations.
- Assign points to clusters based on Sum Square Error (SSE).
- Calculating new centroids by evaluating clusters mean values.
- Handling empty clusters by
- Choose the point that contributes most to SSE
- Choose a point from the cluster with the highest SSE
- Returning clusters, centroids and clusters points indices

Results

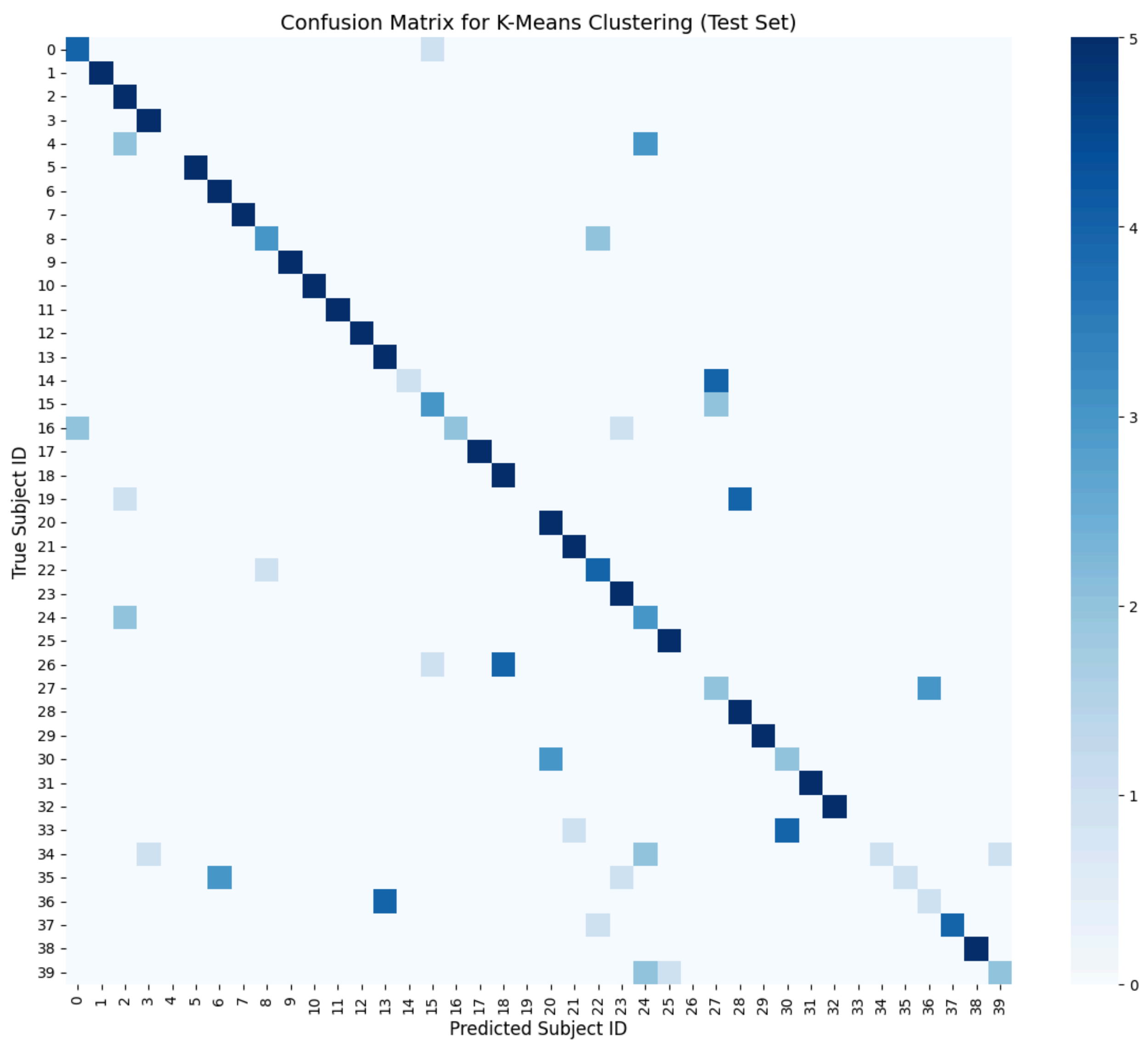
With PCA

Accuracy	**K = 20**	**K = 40**	**K = 60**
Alpha = 0.8	0.44	0.67	0.8050
Alpha = 0.85	0.3750	0.5950	0.7250
Alpha = 0.9	0.44	0.6450	0.7900
Alpha = 0.95	0.4550	0.6250	0.78
Alpha = 0.99	0.3950	0.6650	0.79

With Autoencoder

- Accuracy of training set: 0.7300

- Test Accuracy: 0.6550

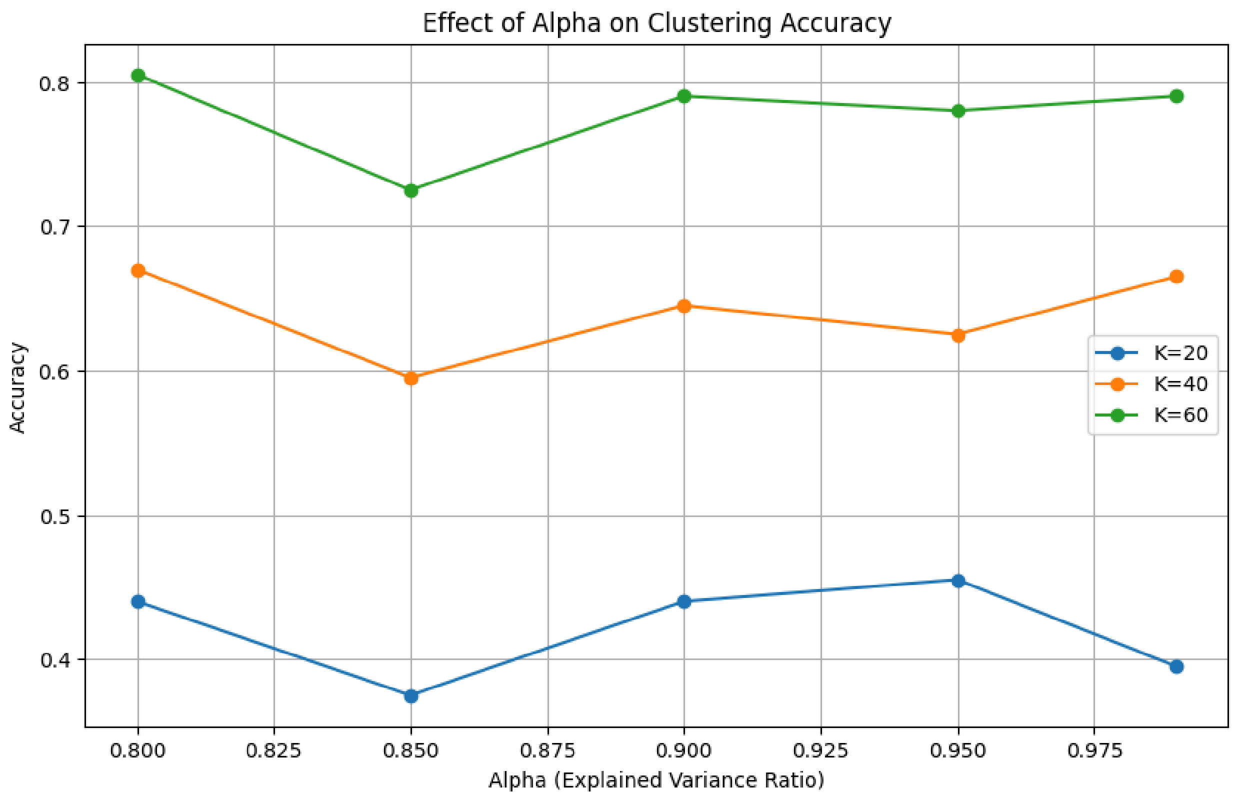


Evaluating K-Means on test set

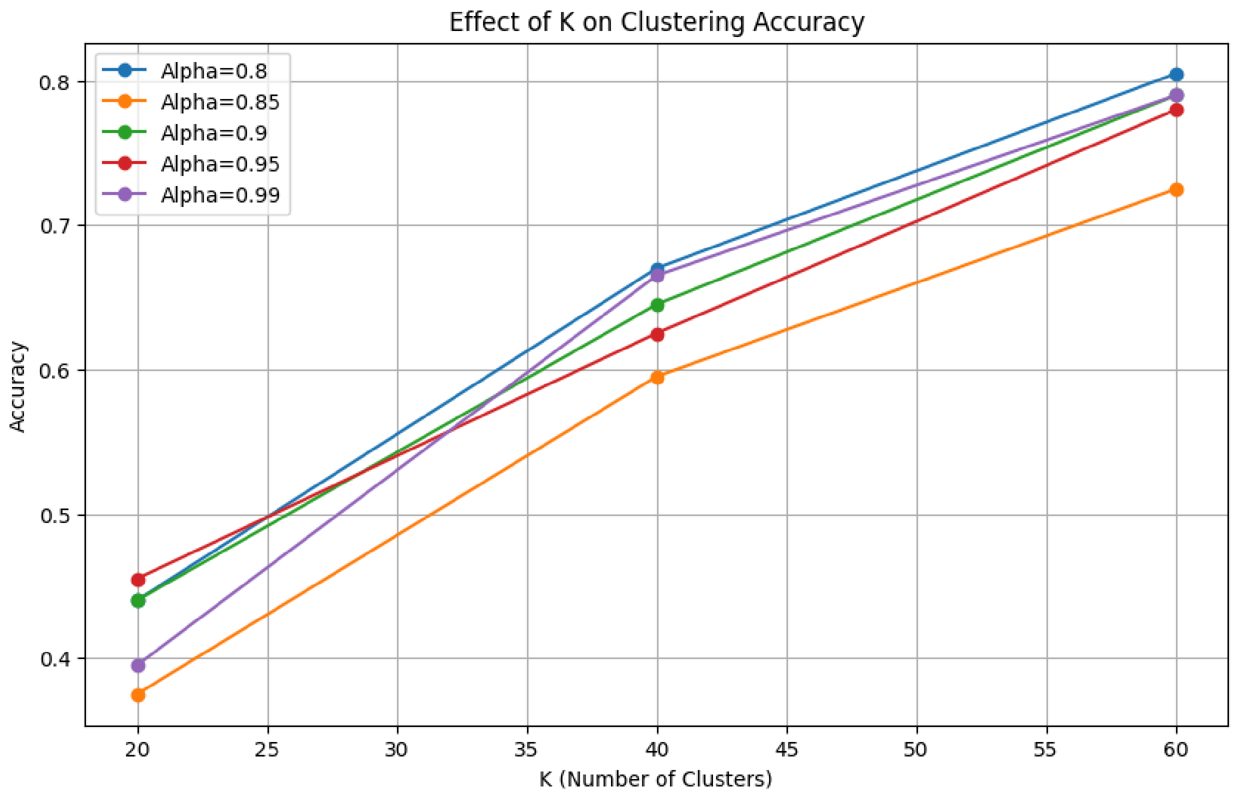
Test Accuracy: 0.7150, Test F1-Score: 0.6706

Analysis

Effect of Alpha and K on K-Means Accuracy



As alpha increases (e.g., from 0.800 to 0.975), PCA retains more principal components, increasing the dimensionality of the reduced space.
K-Means relies on Euclidean distances, which become less meaningful in higher dimensions due to the "curse of dimensionality."
In high-dimensional spaces, distances between points tend to become more uniform, making it harder for K-Means to distinguish meaningful clusters.



Increasing the number of clusters increases clustering accuracy and reduces the probability of clustering points wrong.

Gaussian Mixture Model

Approach

- GMM was implemented using Expectation-Maximization (EM).
- Initial means were selected randomly from the dataset.
- Initialized covariances as scaled identity matrices and weights equally.
- Looping until convergence or reaching max iterations:
  - Assign points to clusters based on probabilistic responsibilities (E-step).
  - Update means, covariances, and weights using weighted means (M-step).
- Convergence checked using log-likelihood improvement.
- Due to Numerical instability we worked in log space.

Results

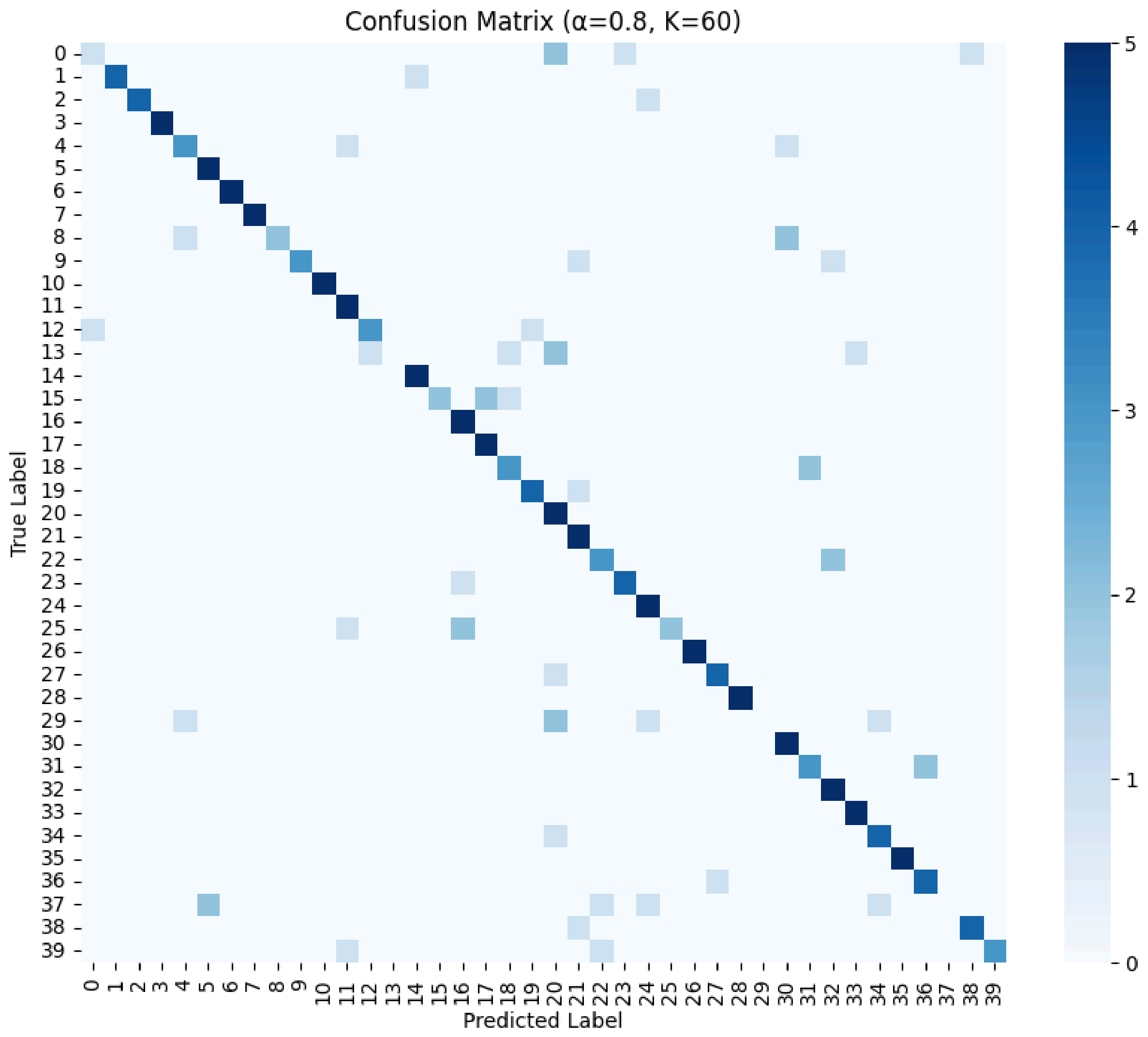
With PCA

Accuracy	**K = 20**	**K = 40**	**K = 60**
Alpha = 0.8	0.345	0.575	0.73
Alpha = 0.85	0.275	0.565	0.71
Alpha = 0.9	0.27	0.44	0.565
Alpha = 0.95	0.225	0.32	0.53

With Autoencoder

- Accuracy of training set: 0.73

- Test Accuracy: 0.65

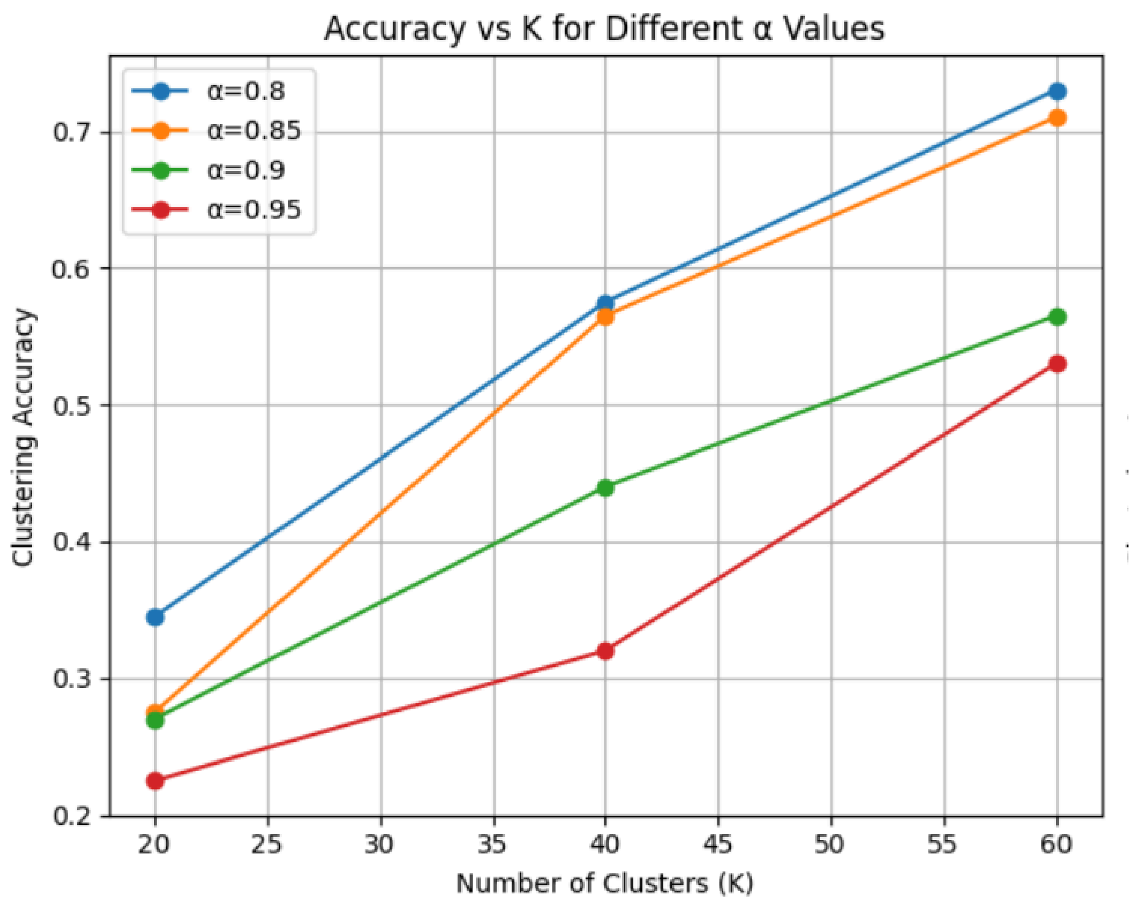


Evaluating GMM on test set

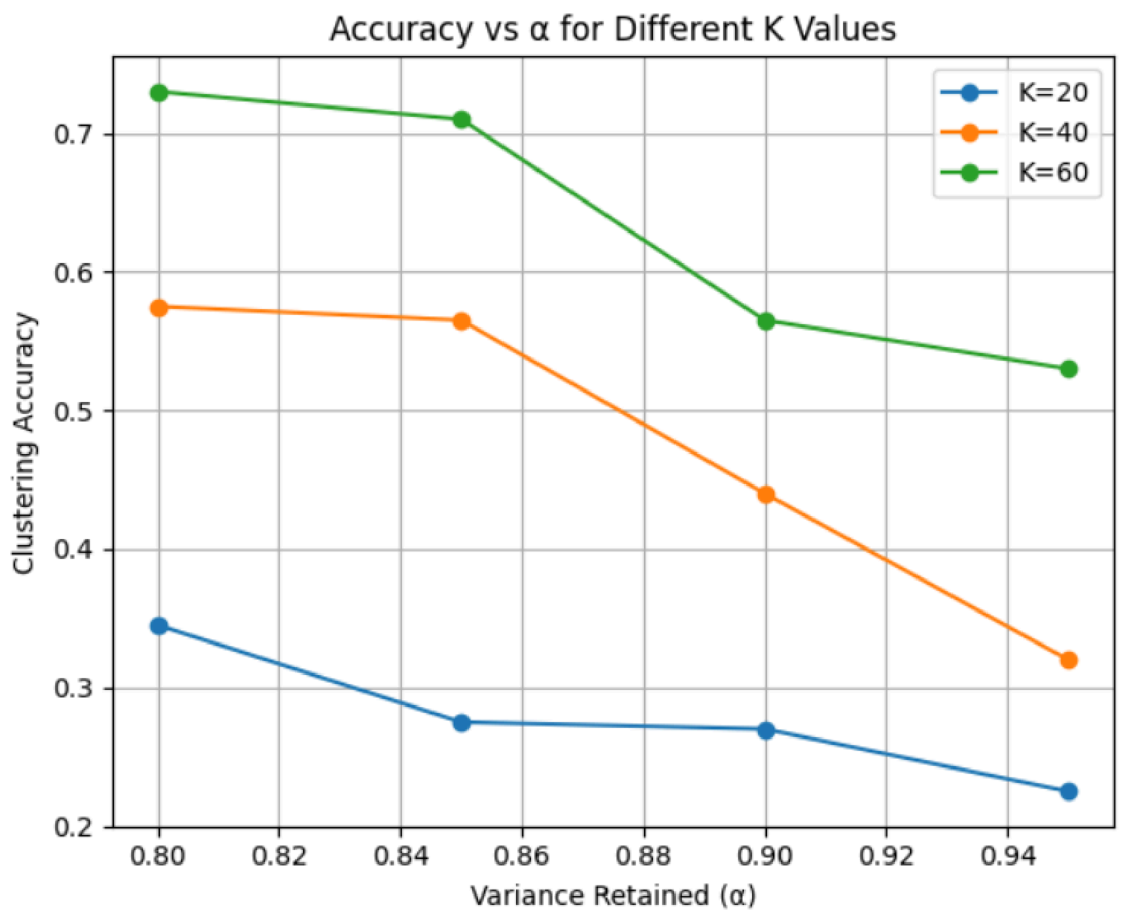
Test Accuracy: 0.75, Test F1-Score: 0.7164

Analysis

Effect of Alpha and K on K-Means Accuracy



For all α values, clustering accuracy gets better as K increases
All lines trend upward as K increases, showing that adding more clusters improves accuracy
As K increases, the model has more components, so it can better capture the different patterns or groups in the data, leading to better clustering and higher accuracy.



For all K values, accuracy goes down as α increases

All lines show a downward trend as α increases, meaning keeping more variance in PCA doesn't help the GMM cluster better in this case.

a higher α means keeping more variance, so the data has more dimensions. This might include noise or less important details that make it harder for the GMM to find clear clusters, lowering accuracy.

GMM models each cluster as a Gaussian distribution, which involves estimating means and covariances. In higher dimensions (higher α), the covariance matrices become harder to estimate accurately, especially if the data points are spread out or noisy. This can lead to overlapping clusters, where points are assigned to the wrong group, reducing accuracy.

GMM vs K-means

Comparison between k-means and GMM performance on test set

On PCA:

	Accuracy	F1-Score
GMM	0.75	0.7164
K-Means	0.715	0.7106

On autoencoder:

	Accuracy	F1-Score
GMM	0.64	0.7414
K-Means	0.665	0.6159

With PCA (Training set)

	K-means			GMM		
Accuracy	K = 20	K = 40	K = 60	K = 20	K = 40	K = 60
Alpha = 0.8	0.44	0.67	0.8050	0.345	0.575	0.73
Alpha = 0.85	0.3750	0.5950	0.7250	0.275	0.565	0.71
Alpha = 0.9	0.44	0.6450	0.7900	0.27	0.44	0.565
Alpha = 0.95	0.4550	0.6250	0.78	0.225	0.32	0.53