Assignment 4

Computational Intelligence, SS2023

Team Members		
Last name	First name	Matriculation Number
Carlos Franco	Verde Arteaga	
Nedžma	Mušović	
Laura Maria	Höber	

Contents

2 Expectation Maximization (EM)

The aim of this subtask was to implement and apply the EM algorithm. It is an iterative approach to determining the GMM parameters α_k, μ_k and Σ_k by following these four principial steps:

1. Initialization
There are multiple methods which can be applied for the parameters initialisation. In this task the initial values for the parameters of interest (for t = 0) were selected according to the lecture materials as follows:

- α_k is initially uniformly distributed with the respect to number of components K: $\alpha_k = \frac{1}{K}$
- For μ_k we select k random data samples from the data set.
- And the covariance matrix Σ_k is set to the overall covariance matrix of the data set and is initially identical for all components.

2. E-step
In the expectation step the posterior probability for each data sample x_n is computed as follows:

$$r_k^n = \frac{\alpha_k^t \cdot \mathcal{N}(x_n|\mu_k^t, \Theta_k^t)}{\sum_{k'=0}^{K} \alpha_{k'}^t \cdot \mathcal{N}(x_n|\mu_{k'}^t, \Theta_{k'}^t)} = P(k|x_n, \Theta^t)$$

3. M-Step
The maximization step updates the model parameters with the respect to the newly computed posterior probability for the next iteration and increases the iteration variable ($t = t{+}1$) repsectively.

$$\begin{aligned}
\mu_k^{t+1} &= \frac{1}{N_k} \sum_{n=0}^{N} r_k^n \cdot x_n \\
\Sigma_k^{t+1} &= \frac{1}{N_k} \sum_{n=0}^{N} r_k^n \cdot (x_n - \mu_k) \cdot (x_n - \mu_k)^T \\
\alpha_k^{t+1} &= \frac{N_k}{N}
\end{aligned}$$

4. Likelihood evaluation
In the last step the log likelihood of the estimated parameters given data gets calculated and compared with the likelihood from the previous iteration. If the convergence gets observed, the parameters are considered suitable and the algorithm terminates. Otherwise, the it continues with the next iteration.

$$logP(\mathcal{X}|\Theta^{t+1}) = \sum_{n=1}^{N} log \sum_{k=1}^{K} \alpha_k^{t+1} \cdot \mathcal{N}(x_n|\mu_k^t, \Theta_k^t)$$

2.1 EM for 2-dimensional features

In this part the data samples of iris flowers are classified, by applying the, as previously described, implemented and initialized algorithm for estimating corresponding parameters for GMM components on the two features "sepal length" and "petal length". This corresponds to scenario 1.

2.1.1 K = 3

Applying the EM algorithm on the first two features of the provided dataset and assuming the default number of components (K = 3) resulted in the following plots:

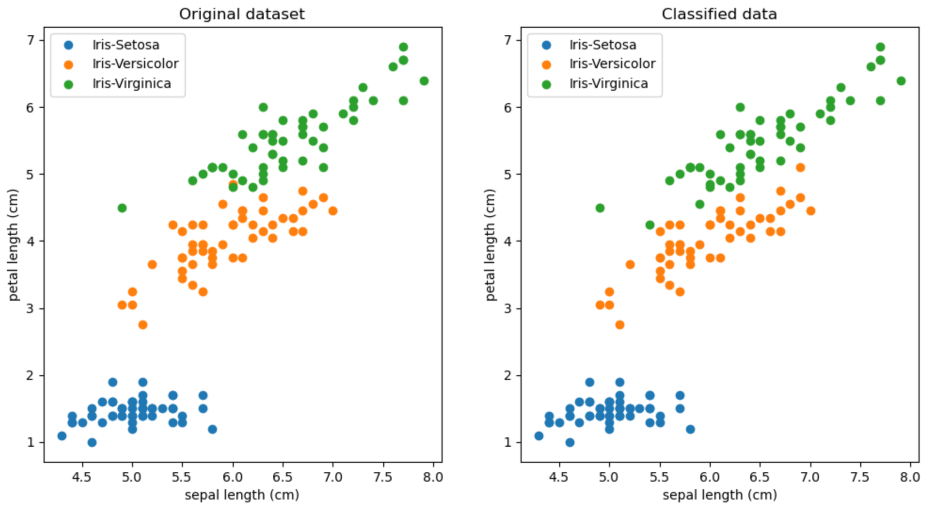

Figure 1: Classified data with 3 components in scenario 1

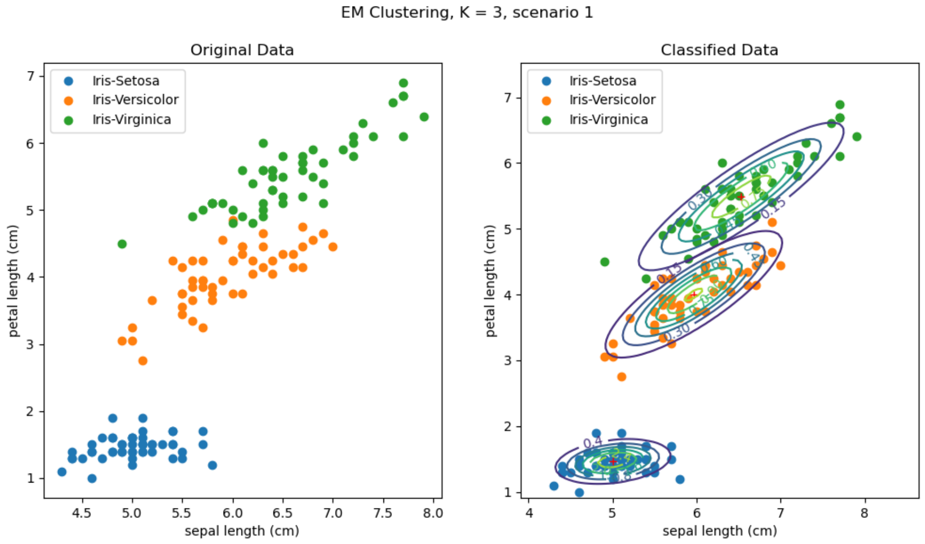

Figure 2: Classified data with 3 components and the GMM plot in scenario 1

As it can be observed in the previous plot, the EM does not completely mirror the original data labels and some borderline data samples (between the green and orange classes) are missclassified. Nevertheless, it can be claimed that the EM alorithm reliably classifies the data set.

The parameters of the corresponding GMM components seem to be fairly well estimated. The plotted variance mostly covers the components variances except for the data samples on the edges.

However, it is worth mentioning, that the performance of the EM algorithm highly depends on parameters initialisation, and given the randomness in mean values, not every run leads to the best local minima and thus the best results. That is why, the algorithm should be run multiple times, and the results with the highest likelihood selected as the most correct one.

2.1.2 K = 2, 5

The EM algorithm significantly relies on the number of components taken into account upon data processing. Thus changing the value of K while dealing with the same data set, has a large impact on the results. That impact for the K = 2 and K = 5 case is shown in the following plots:

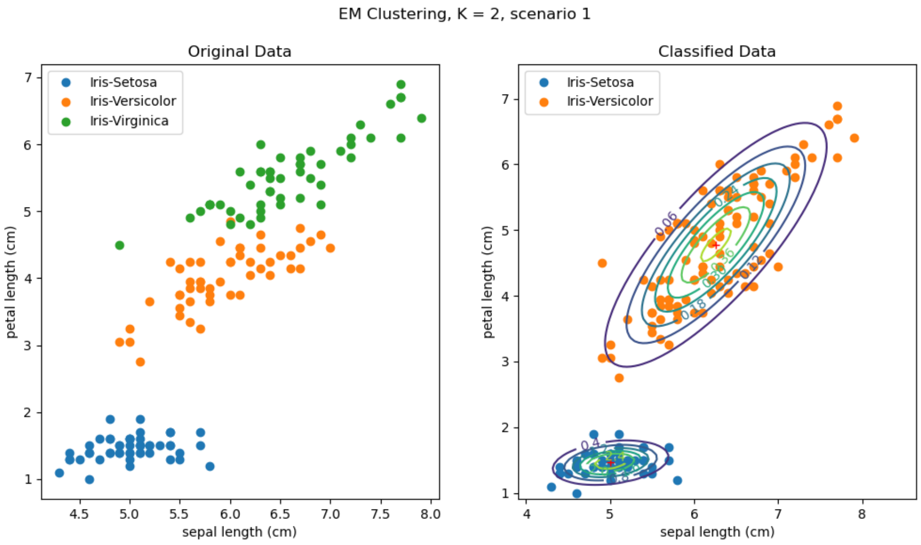

Figure 3: Classified data with 2 components in scenario 1

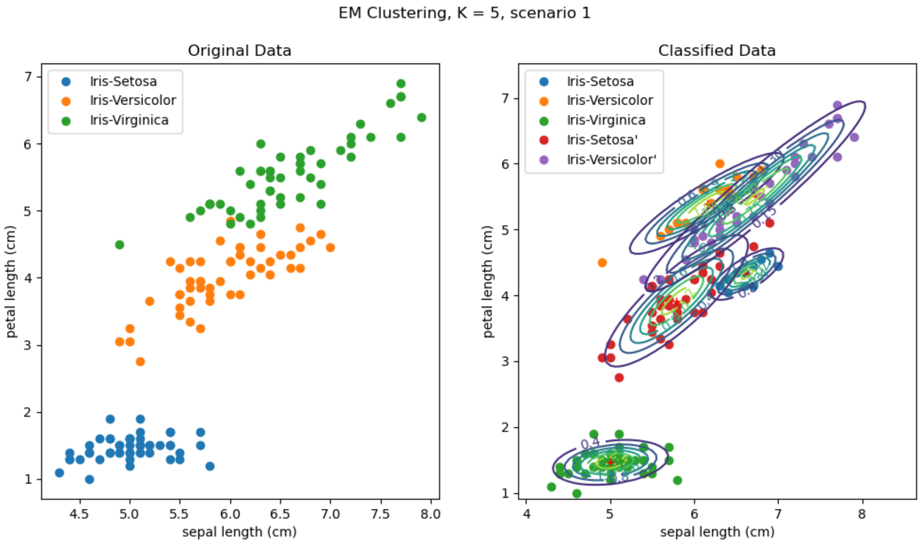

Figure 4: Classified data with 5 components in scenario 1

As expected, when considering only 2 clusters, the EM algorithm, even visually intuitive, merges the orange and green cluster, while the EM expecting 5 clusters divides the merged cluster (in the case of K = 2) into 4 subclusters without affecting the visually clearly separated cluster at the bottom.

2.1.3 Log-likelihood evaluation

The behvaiour of log-likelihood functions as our termination criterion over iterations for the observed K variances, is given in the following plots respectively:

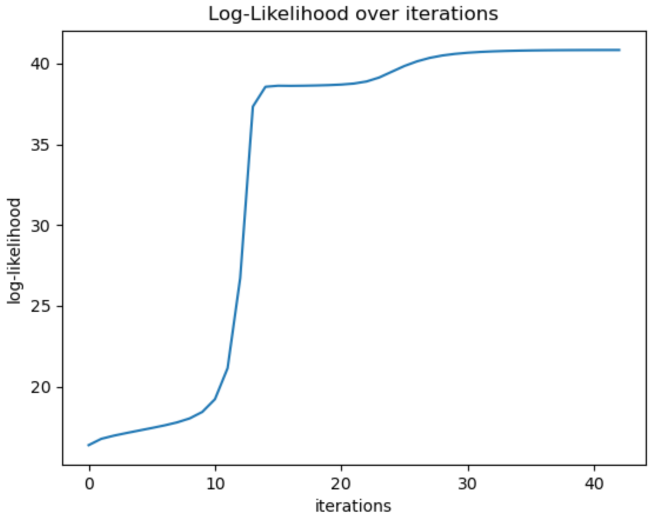

Figure 5: Log likelihood function over iterations for K = 3

As expected, the log likelihood gets increased over the iterations until it converges to a relatively high value recognized as the termination sign for the algorithm. As shown in the Figure 5, the convergence gets achieved after 40 iterations. The likelihood may visually appear steady for more the last 10 iterations in this plot, because of a very low tolerance (tol = 0.001).

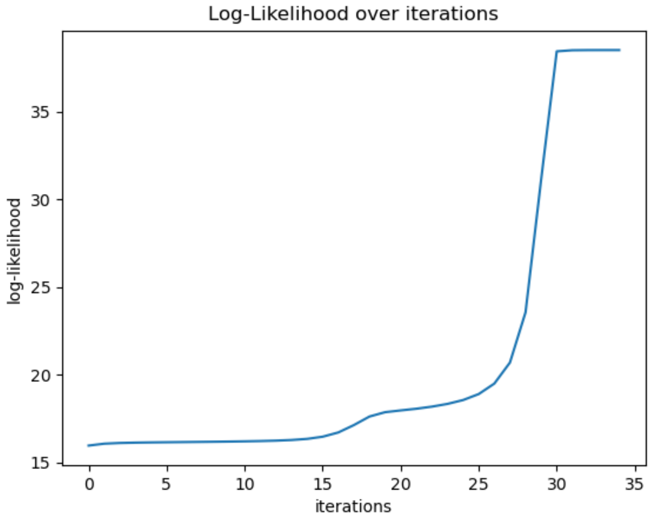

Figure 6: Log likelihood function over iterations for K = 2

Compared to the case K = 3, the EM converged noticeably faster, after 30 iterations. The achieved log-likelihood function is however, as expected, lower.

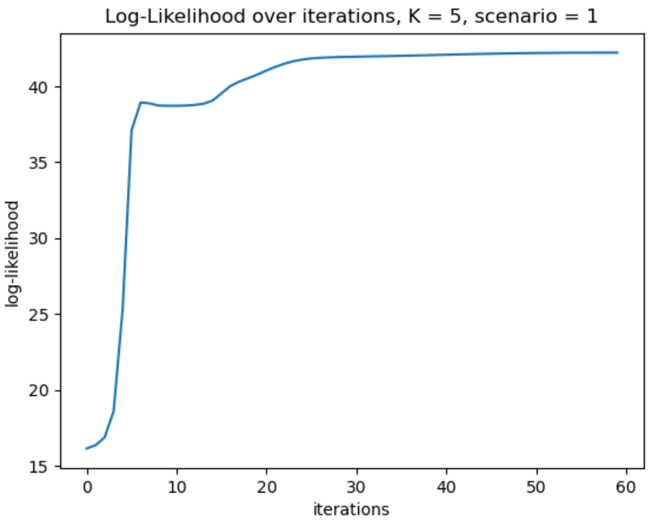

Figure 7: Log likelihood function over iterations for K = 5

Through the introduced complexity of the parameter space in the case of K = 5, the convergence is achieved only after 60 iterations, as shown in the Figure 7.

2.2 EM for 4-dimensional features

Now, all 4 features "sepal length', 'sepal width', 'petal length' and 'petal width' are used for parameters estimation. For the easier visualisation and comparision, only the two features (as in scenario 1) are represented in the following plot:

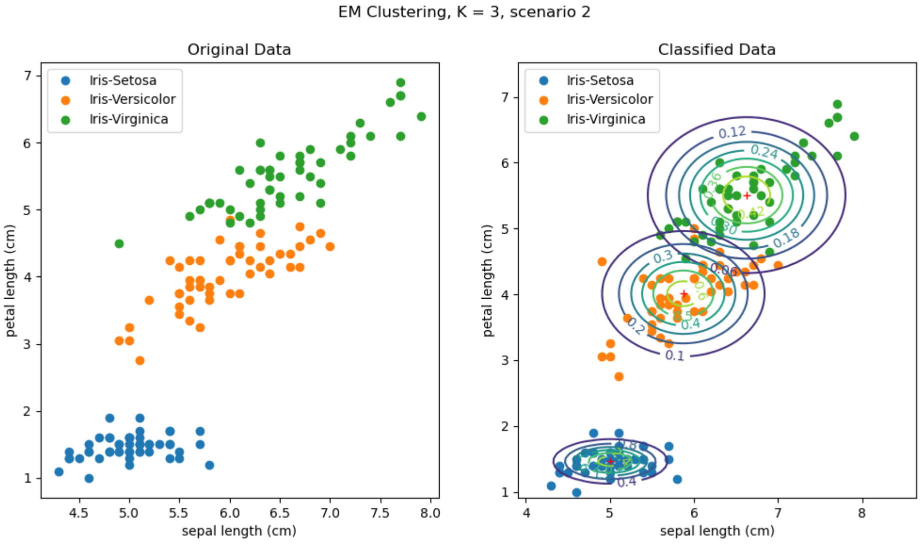

Figure 8: Classified data with 5 components in scenario 1

Including all four features in the GMM parameters estimation leads to fairly reliable results again. There are less (but still present) missclassifications to be observed at the borderline between the green and orange cluster, implying a comparably similar efficiency of both methods for the given data set.

However, it is worth mentioning that using four features in the GMM parameters estimation increases the stability of the system with the respect to the randomized initial parameters values.

The figure 9 shows the log likelihood function over iterations in the scenario 2. We can observe a relatively quick convergence compared to scenario 1, as the increased number of features may provide the algorithm with more useful information in the estimation process.

Additionally, the EM was adjusted such that it diagonalise iteratively computed covariance matrices and thus decorrelates features, which contributes to the overall simplicity and the faster computations. However, the plotted GMM is always axis aligned, as the non-diagonal elements get removed and the information does not get capured, as in the scenario 1.

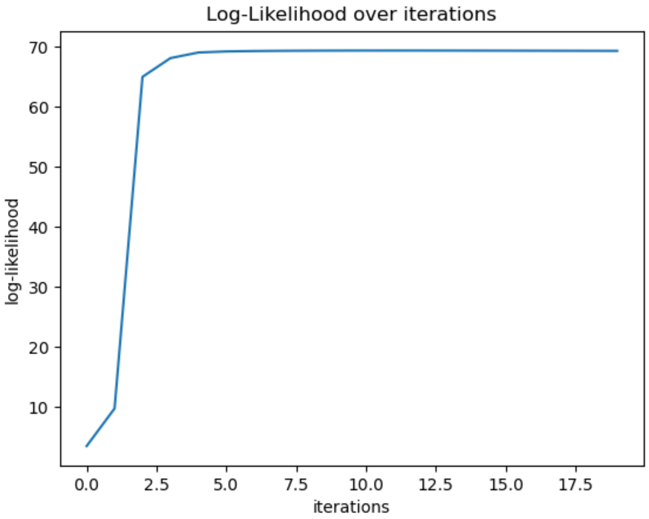

Figure 9: Log likelihood function over iterations for K = 3, scenario 2

3 K-Means

The K-means algorithm is a simplification of the EM algorithm, i.e. each sample is only modelled by one component k which can be described only by it's mean μ_k, the weights α_k are uniform and can be neglected and the covariance matrix is spherical and equal for all clusters. To find K clusters among the data samples, the essential steps consist of:

1. Initializing the cluster centers μ_k by taking K random samples out of the data.
2. Assign a class label Y_k to each data sample, based on the closest center to the sample, i.e. the smalles euclidean distance:
   $Y_n = x_n | k = armin_{k'}[(x_n - \mu_{k'}^t)^T(x_n - \mu_{k'}^t)] \quad \forall k = 1, ..., K$
3. Update the cluster centers by averaging over all samples, that were assigned the same class label:
   $\mu_k^{t+1} = \frac{1}{|Y_k|} \sum_{x_n \in Y_k} x_n$
4. Evaluate the cumulative distance of all samples to their assigned center:
   $J^t = \sum_{k=1}^{K} \sum_{x_n \in Y_n} (x_n - \mu_{k'}^t)^T(x_n - \mu_{k'}^t)$
   If $|J^t - J^{t-1}| >\in$ the algorithm converged. Otherwise repeat steps 2-4.

3.1 K-Means for 2-dimensional features

In this section the data samples of iris flowers are classified based on the two features "sepal length" and "petal length". The training data $X = x_1, ..., x_N$ is therefore a vector of 2-dimensional features vectors x_n and the vector of targets is known as $\mathbf{t}$. This corresponds to scenario 1.

3.1.1 K = 3

First, the number of clusters is chosen as K = 3, which corresponds to the number of possible class labels within the data. While k-means in itself doesn't provide any information about the label for each cluster, because it is an unsupervised algorithm. However, here a label got assigned to each cluster and it's respective data points based on the most common target value among the data samples, that were assigned to that center.

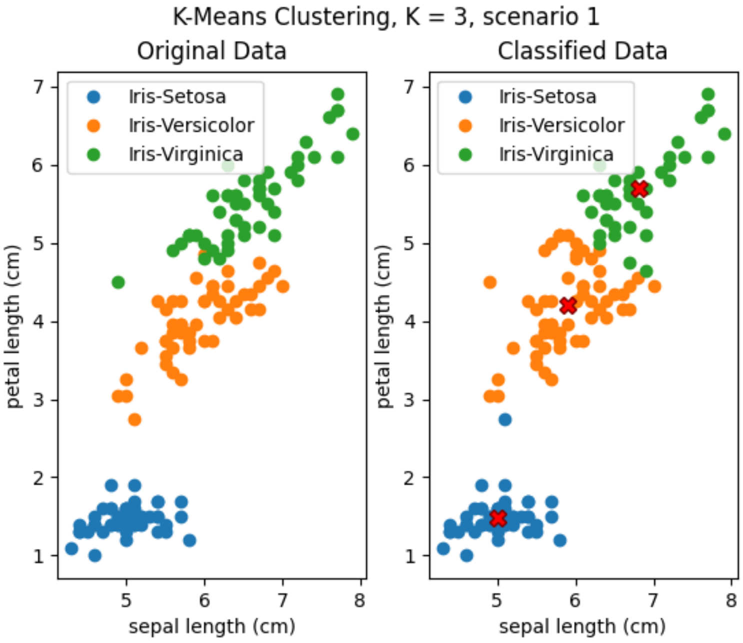

Figure 10: Classified data with 3 clusters

The clusters fit the data fairly well. The blue cluster only has one misclassification, while green cluster looses some more points to the orange one. Between the gren and blue cluster it can also be seen, that the decision boundaries between the clusters are linear, because the euclidean distance is used to determine which center is closest to each sample.

3.1.2 K = 2, 5

Next, a K of 2 and 5 were also tested on the data. In this case, the labels got assigned to the centers the same way as before, but in case multiple centers get assigned the same label one of them gets slightly different markers, so the clusters can still be differentiated in the plot. They were not given a separate color(as stated in the exercise sheet), so it's still visible, which class the samples belong to.

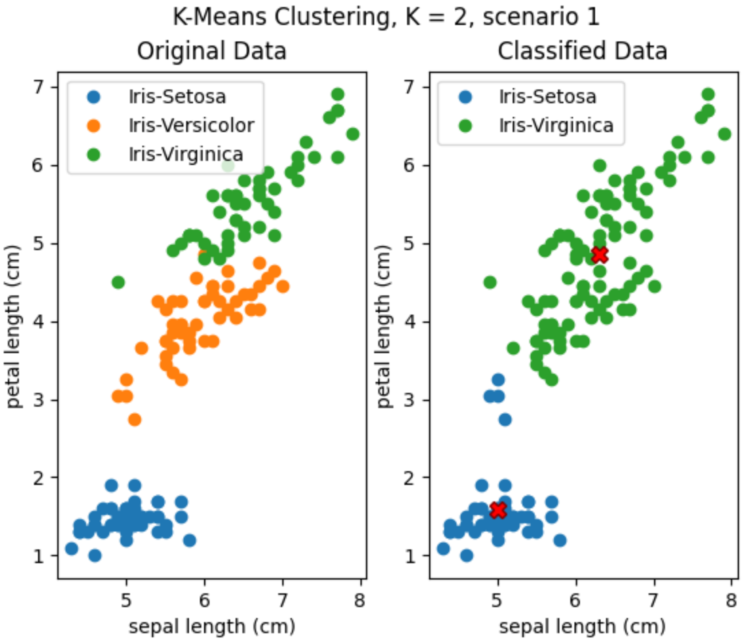

Figure 11: Classified data with 2 clusters

As expected, the bigger more coherent part of the data distribution gets clustered into one class, while the small blue cluster at the bottom still stays separate. This means, that it the cluster number is chosen smaller than the actual number of existing classes, the overlapping/more similar classes are detected as one. Therefore, this is not a good representation of all the information contained in the data.

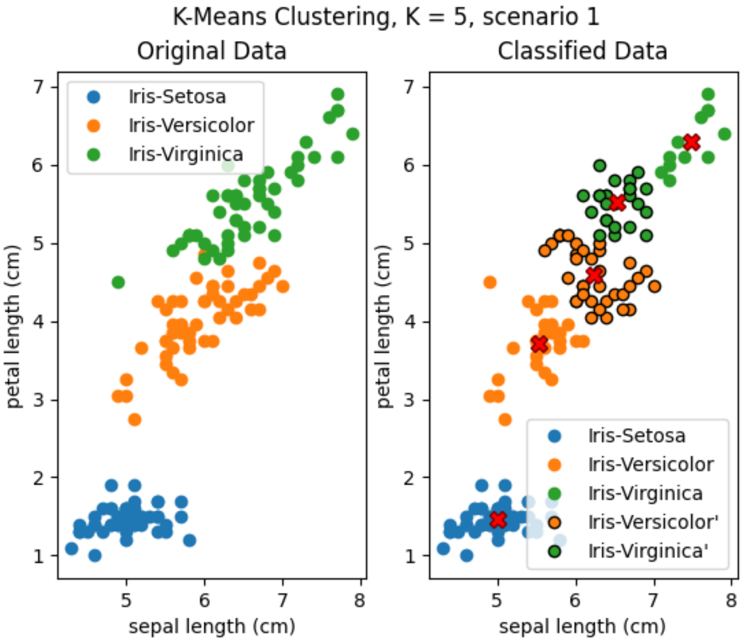

Figure 12: Classified data with 5 clusters

For a cluster number that is higher than the actual number of classes, the bigger clusters get separated into multiple parts. Two of the clusters received the same label, so they received a black border to stay distinguishable. The one misclassified sample of the blue cluster is now corrected and the differentiation between the green and orange clusters got slightly better.

3.1.3 Cumulative distances

For a K = 3,2,5 the cumulative distances of all samples to their respective centers was calculated in every iteration. Once a tolerance of 0.1 or the maximum number of iteration of 20 was reached, the algorithm is halted.

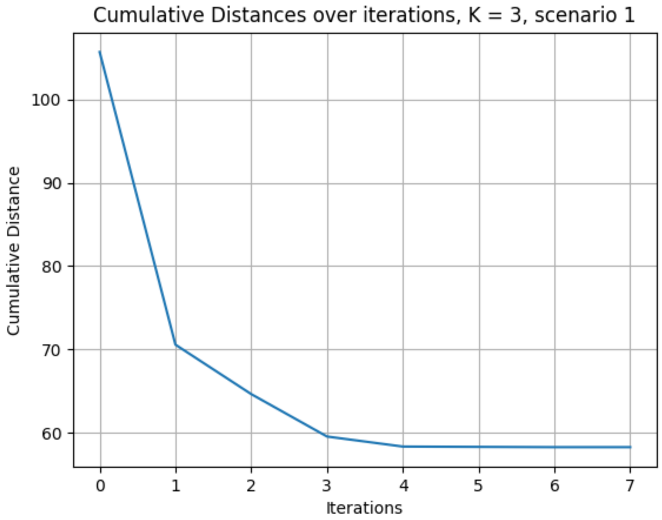

Figure 13: Cumulative distance for scenario 1 with 3 clusters

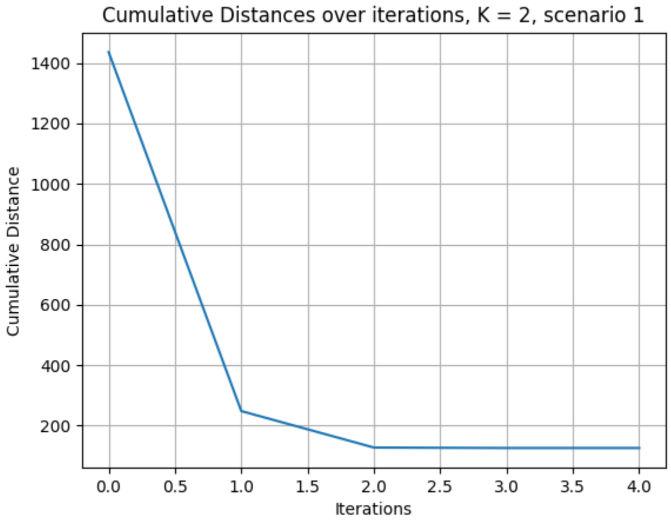

Figure 14: Cumulative distance for scenario 1 with 2 clusters

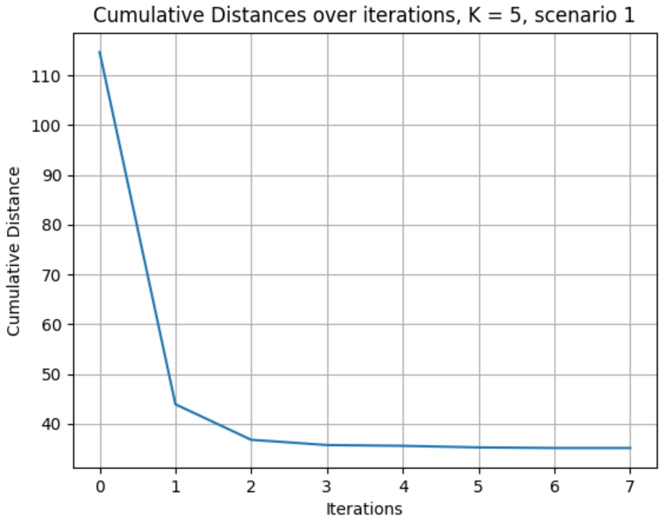

Figure 15: Cumulative distance for scenario 1 with 5 clusters

From figures 13 to 15 above it's noticeable, that the cumulative distance decreases exponentially with iterations. For a lower cluster number the algorithm converges slightly faster, after already 4 iterations, while the other 2 need 7 iterations. However, the final cumulative distance for the lower cluster number K = 2 is still very high, with a value of about 100, while the highest cluster number K = 5 reaches the lowest final cumulative distance with a value of about 35.

That was to be expected, because the more clusters there are, the more possibilities the samples have to find the closest center. So, separating the clusters into sub-clusters, that still belong to the same class can be an effective way of improving the classification and therefore compensating the simplified form of the k-means clusters with respect to the EM clusters.

3.2 K-Means for 4-dimensional features

Now, all 4 features "sepal length', 'sepal width', 'petal length' and 'petal width' are used to find the cluster centers. Therefore the centers are also 4-dimensional, but do visualize the results, only the feature values for sepal and petal length are used again.

Figure 16 shows, that the results do not vastly differ from the previous approaches. There are only slight improvements in the classification of the border values between the orange and green cluster, but apart from one additional correctly classified point the result is the same as in figure 12

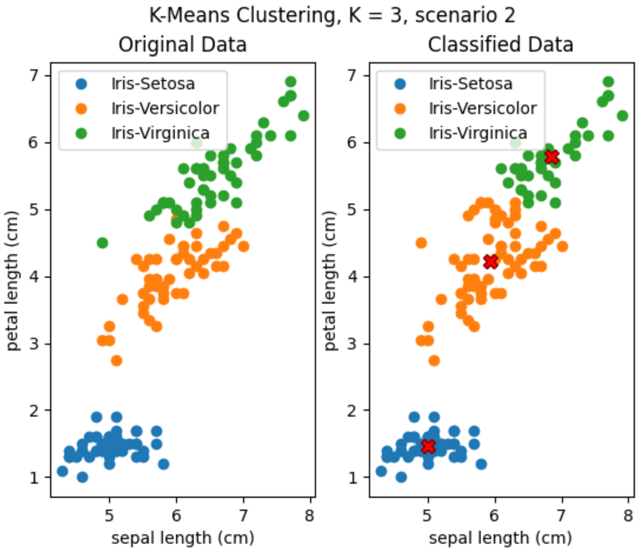

Figure 16: Classified data with 3 clusters and 4-dimensional data

Because the cumulative distance doesn't factor in the number of dimensions, the converging value stays higher than for scenario 1 with the same cluster number of 3. So, the more dimensions, the higher the overall distance gets. Although, in this case it needs 3 less iterations to converge.

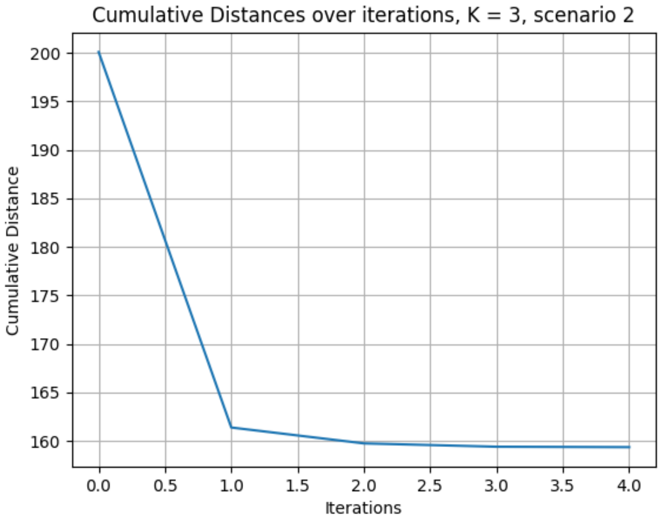

Figure 17: Cumulative distance for scenario 2 with 3 clusters

4 Principal Component Analysis

4.1 Pre-processing the data with PCA

Principal component analysis is used in this case to reduce the dimensionality of the data, while loosing as little information as possible. As visible in figure 18 and 19 the distribution of the data changed. But, what can be notices, is that the clusters, especially the orange and green one, can be better distinguished from each other.

For the EM algorithm, the reach a good result, only with minor misclassifications between the green and orange clusters, similar to the results without PCA. For the Log-Likelihood the curve in figure 19 converges to a slightly lower value of about 33 % instead of about 40 % reached in figures 5, 6 and 7.

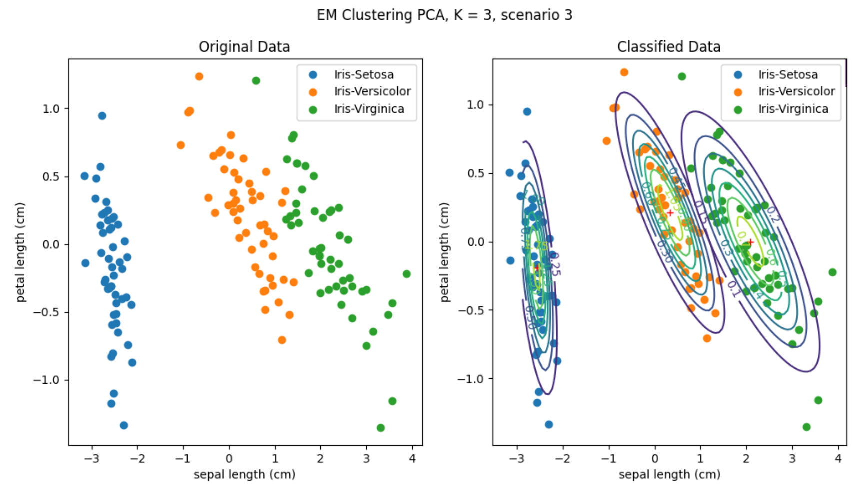

Figure 18: Clusters obtained with EM with a cluster number of K = 3 and PCA applied

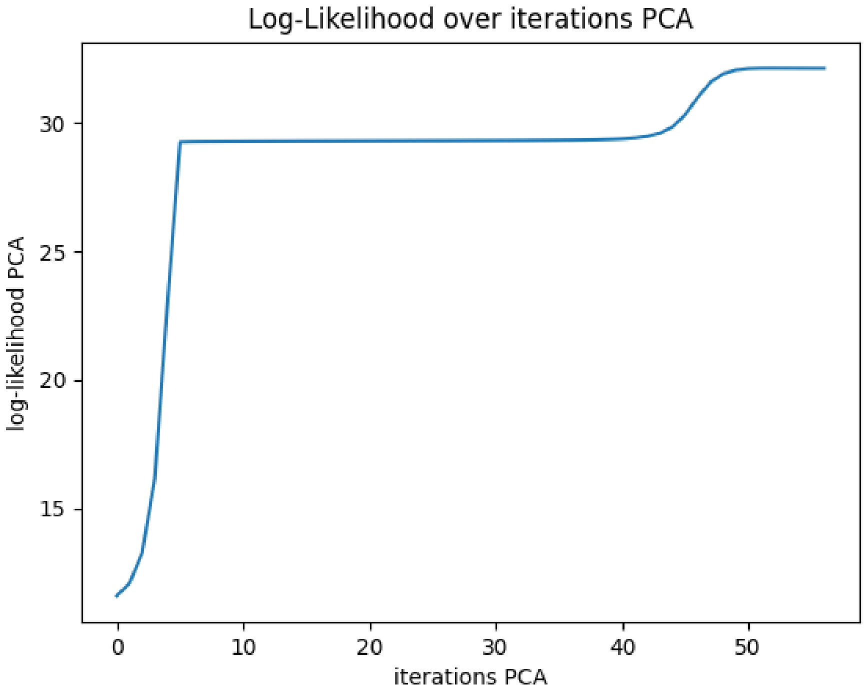

Figure 19: Log likelihood of classification with EM with a cluster number of 3, after PCA was applied

For the K-means algorithm, figure 24 also shows very promising results, with slightly more misclassifications than for the EM algorithm between the orange and green cluster. The cumulative distance of figure 25 shows a similar behavior like the one shown in figure 13 for K=3 without PCA. Both converge after 7 iterations and with PCA apllied, the eventual distance is only slightly higher, with about 70 instead of 60.

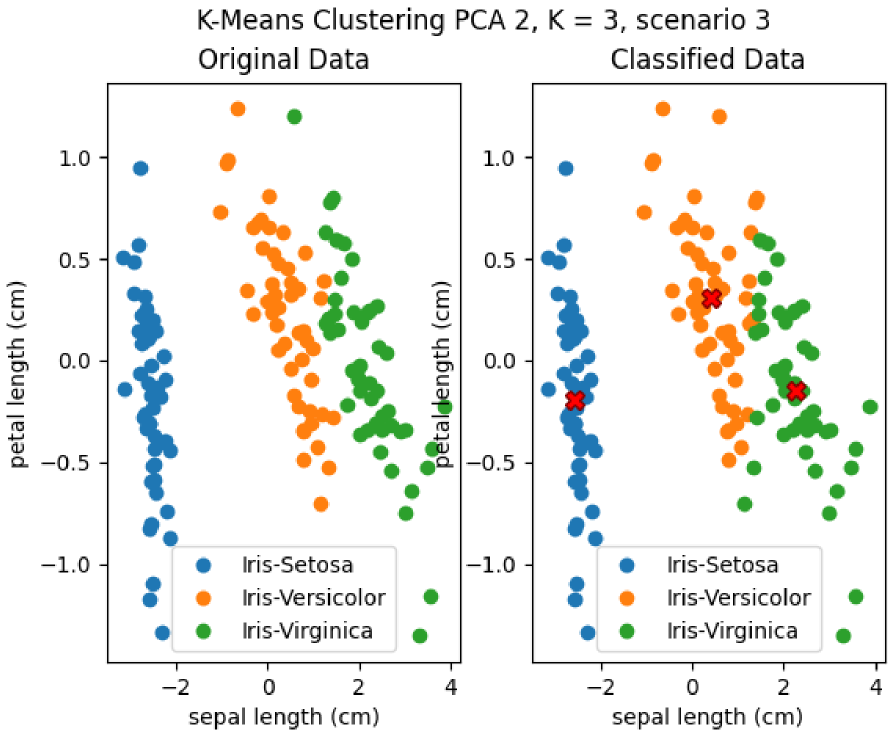

Figure 20: Clusters obtained with K-means with a cluster number of K = 3 and PCA applied

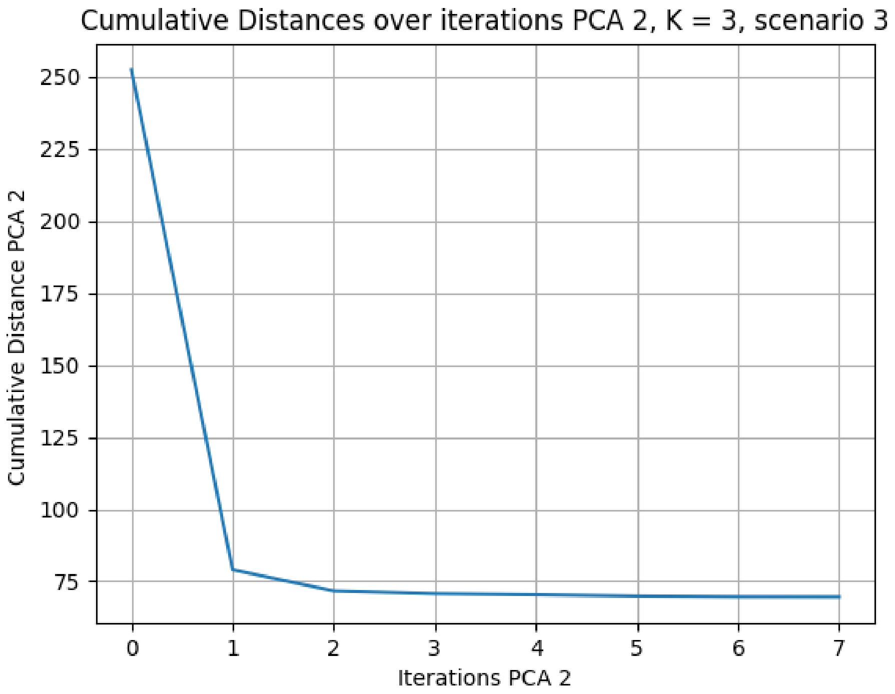

Figure 21: Cumulative distance per iteration with K-means and a cluster number of 3, after PCA was applied

4.1.1 How much of the variance in the data is explained in this way?

By calculating the ration between the total variance over the data with all 4 dimensions and the variance of the data, that was reduced to 2 dimensions it can be checked how much of the original variance is maintained after applying PCA. In our case an explained variance of 97,68 % is maintained, so most information within the data is still present, while only half of it is needed. This shows how much more efficient classification can be made by applying PCA.

4.1.2 PCA with whitening

The transformed data has zero mean and a identity co-variance matrix. Again apply EM and k-Means to the so pre-processed data and visualize the results

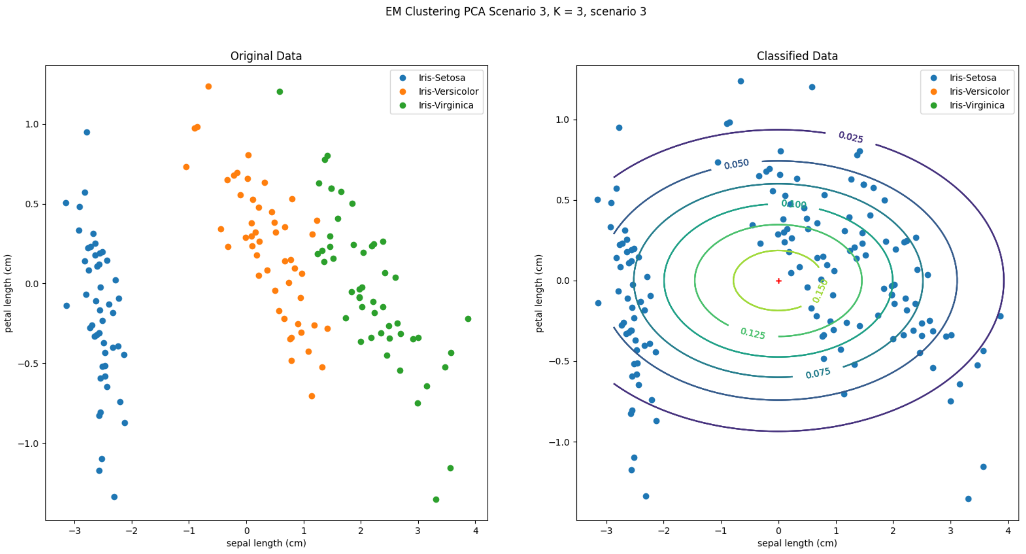

Figure 22: Clusters obtained with EM Scenario 3,transformed data has zero mean and a identity co-variance matrix

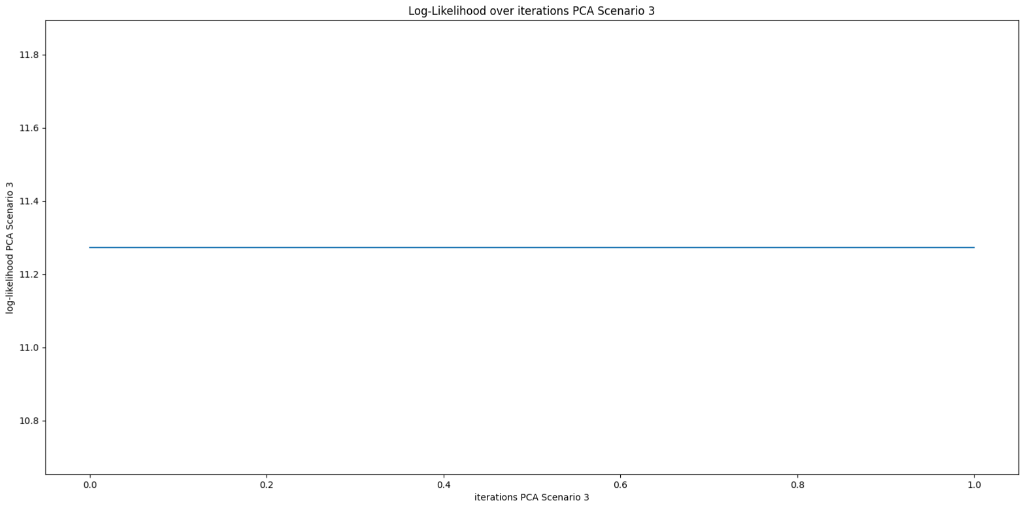

Figure 23: Log likelihood of classification with EM Scenario 3,transformed data has zero mean and a identity co-variance matrix

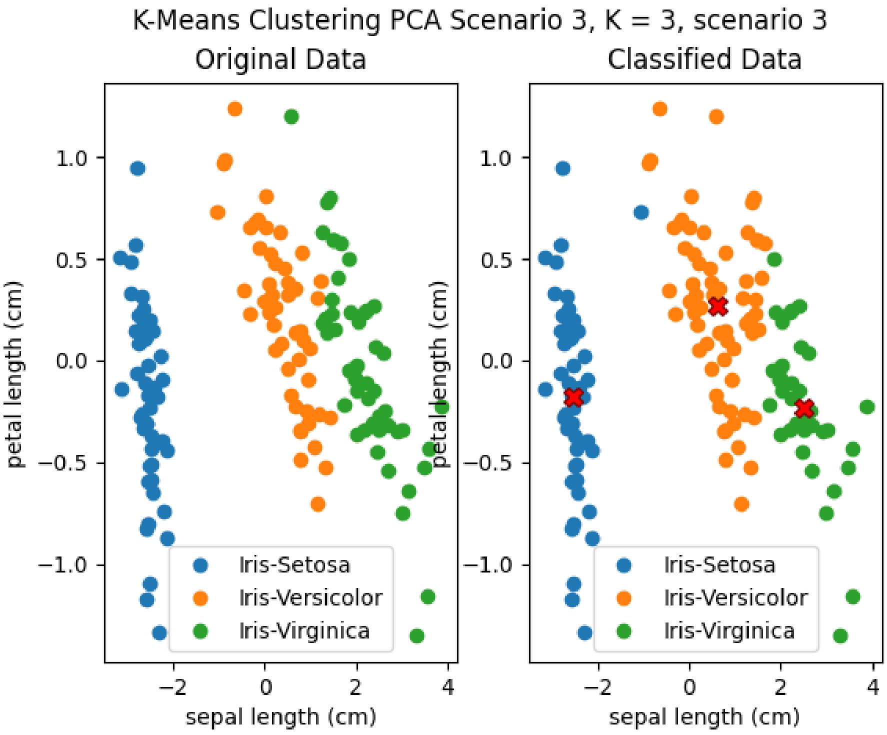

Figure 24: Clusters obtained with K-means Scenario 3,transformed data has zero mean and a identity co-variance matrix

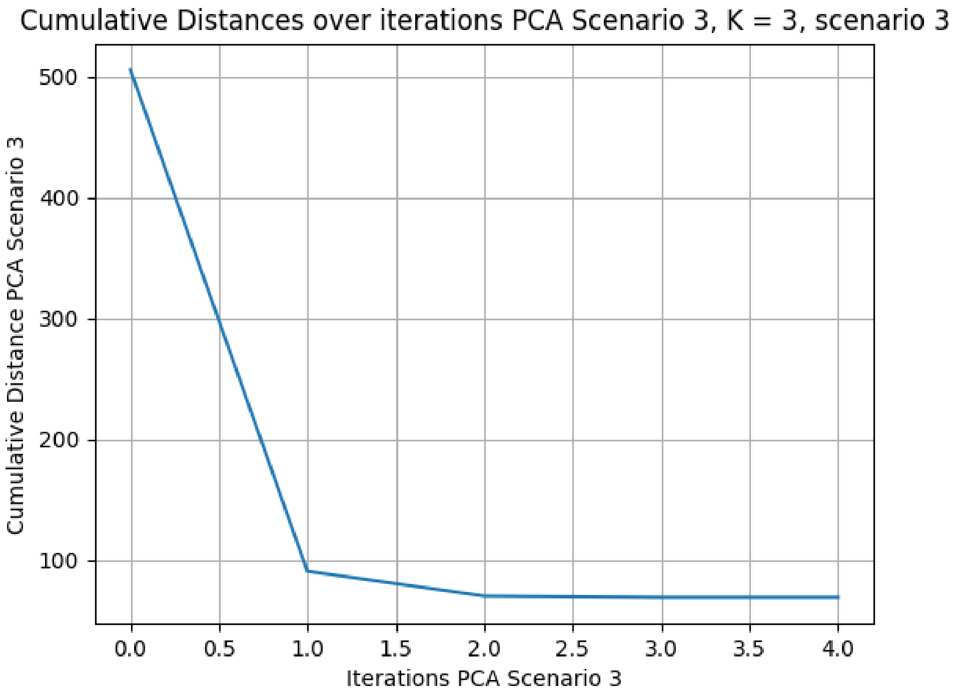

Figure 25: Cumulative distance per iteration Scenario 3,transformed data has zero mean and a identity co-variance matrix