AC50001 Introduction to Data Mining and Machine Learning   
**AC50001.2 Assignment: Classification and Clustering***Author: Vladislavs Ignatjevs (120015095)*

During last week I was working on Classification and Clustering assignment in attempt to create a system capable of dimension reduction using LDA, PCA and clustering using K-Means, and classification using SVM with a linear kernel, SVM with RBF kernel and Neural Network classifier. All these created components were tested/trained on limited MNIST handwritten digit database (only 3 numbers, 100 samples each).

**Q1. Principal Component Analysis (PCA)**

Applying PCA on data was the first classification problem I attempted to implement. PCA is a good way to compress data in regards to pattern recognition improvement. PCA transforms data to new, linearly uncorrelated coordinate system. It has its advantages and disadvantages. For example, if speaking about advantages, PCA is fast and easy to implement. However, if we overwrite data after applying PCA, it will not be possible to get the exact original data back, since PCA clears up the data that it considers unnecessary. As the result, we might lose significant information during the classification process. Apart from that, there are situations where we cannot apply PCA, since it works on linear spaces. For example, if the data is manifold structured, PCA will not perform an analysis in respect to expected results. The other drawback of PCA I found out comes from inability of Linear PCA to provide the information about class labels. LDA, on the other hand, solves this problem.

At the beginning I spent lots of time because I understood the task in a wrong way. Rather than applying PCA to all number data, I attempted to apply PCA on every number separately. This gave interesting results when plotted results. That was what I believe an attempt to reduce dimensions from 100 to 2 and create an interesting interpretation of most common features for numbers one, five and eight. Due to this being unsuitable for the task. I am not attaching the result of these application here, since it is not applicable for the task given, but I backed up the unused code into folder called “Unused”.

PCA can be summarized in the following steps:

1. Subtracting the mean for individual dimensions
2. Calculating covariance matrix
3. Calculating Eigen values and Eigen vectors
4. Transformation into new data

For the task allocated, I started with importing the data set and concatenating all the data sets into one larger matrix and transposing it:

%combine all data

all\_data = [digit\_one digit\_five digit\_eight]';

Then, I calculated the mean for this matrix and subtracted it from individual dimensions:

%substract the mean for individual dimenstions

all\_data = all\_data - repmat(meanAll, size(all\_data,1),1);

After doing that, it was possible to calculate covariance matrix (to see how much data vary from the mean in respect to each other). Then it was possible to calculate Eigen values and Eigen vectors from the covariance matrix calculate in the previous step. Then, by multiplying the matrix containing all data by specific Eigen vector’s values it made it possible to calculate PCA score:

score = all\_data\*v(:,1:2);

This made it ready for the data to get picked up for clustering.

**K-Means method for clustering.**

For data clustering I have used K-Means method because it was more familiar to me than the others. Using the following code I applied k-means to cluster the data into 3 clusters:

%do k-means

[idx, c] = kmeans(pca\_score,3);

%dim1

x1 = min(pca\_score(:,1)):0.01:max(pca\_score(:,1));

%dim2

x2 = min(pca\_score(:,2)):0.01:max(pca\_score(:,2));

%calculate mesh grid

[x1G,x2G] = meshgrid(x1,x2);

x\_grid = [x1G(:),x2G(:)]; % Defines a fine grid on the plot

% Assigns each node in the grid to the closest centroid

idx\_2\_region = kmeans(x\_grid,3,'MaxIter',1,'Start',c);

**Results:**

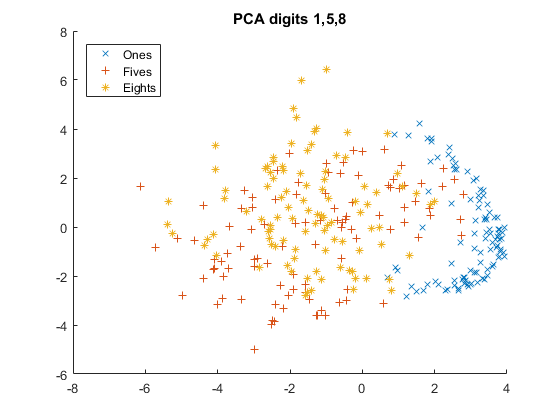


Figure 1: data after PCA projection

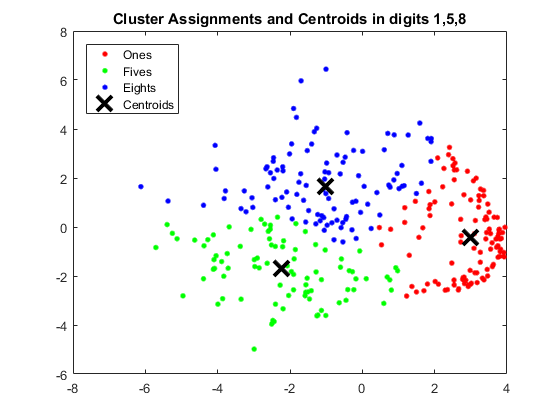


Figure 2: Data after PCA projection with Centroids

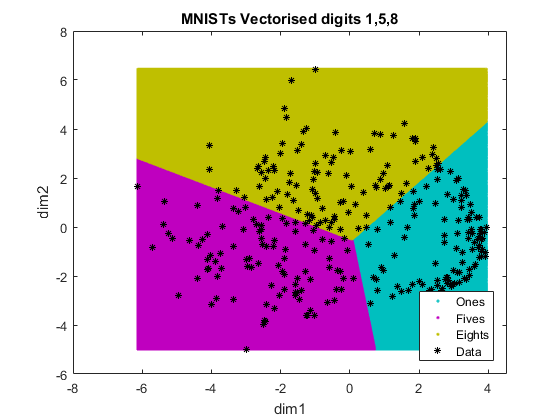


Figure 3: a more visualised plot of PCA projection on data, divided by regions for each of the classes

From the figures above we can see how PCA made it possible to classify all the data we had in the matrix. We can clearly see that due to classification some of the data got lost. For example, from the first figure, we can see that “Fives” data is spread all over the figure, but then classifier choses data “Eights” over “Fives”. Same for projection of “Ones”, if we compare figure one with figure 2 and figure 3, we can see that there is more “ones” data shown on figure 2 and figure 3 than in original projection on figure 1. I believe this happens in the result of implication of PCA discussed above. However, in respect of PCA, I believe that good results were achieved, especially if taking in mind, that it performed dimensionality reduction on data from 100 dimensions into 2 dimensions.

**Q2. Linear Discriminant Analysis (LDA)**

From what I learned, there are two types of Linear Discriminant Analysis applications. These are:

***Linear Discriminant Analysis on 2-class classification problem.***

The summary of steps for 2-class LDA:

1. Calculating means for each class
2. Calculating covariance matrices for each class
3. Calculating “within class scatter matrix” by summing up class covariance matrices
4. Computing Projection Vector by multiplying inverted “within class scatter matrix” on transposed difference of class means

***Linear Discriminant Analysis on multi-class classification problem.***

The summary of steps for n-class LDA:

1. Calculating means for each class and then calculate the mean of the class means
2. Calculating covariance matrices for each of the classes
3. Calculating “within class scatter matrix” by summing up all the covariance matrices
4. Calculating “between class scatter matrix”:
   1. for each of the classes:  
      multiplying number of size of the class by the difference of class means and mean of all class means, multiplied by the difference of class means and mean of all class means transposed.
   2. Summing up the results of above calculation from each class
5. Computing Eigen vectors (projection vectors) and Eigen values from inverted “within class scatter mask” multiplied by between class scatter mask
6. Calculating score by multiplying the matrix containing all data by specific Eigen vector’s values

If comparing LDA to PCA it is essential to mention that unlike PCA, LDA can pick up class labels, it can be used for jobs that PCA is not capable of doing. However, unlike expectations LDA is not guaranteed to perform better than PCA.

For the task allocated, I had to use 3-class LDA. I followed the steps I mentioned above to produce the 3-class LDA, but got stuck on the problem of calculating inverted “within-class scatter matrix”. Due to the data we have, the result of summing up class covariance matrices produced a matrix that had many zero values in it. It is obvious that it makes no sense to attempt to calculate inverse of 0 value. I solved the problem by adding bias to “within class scatter matrix” (I have picked some small number and to the “within-class scatter matrix“) in attempt to produce the matrix that would be suitable for inverting:

%addding small number to avoid inv on zero

dc=0.00001\*eye(size(sw));

sw\_new=sw+dc;

inverted\_SW=inv(sw\_new)\*sb;

It finally made it possible to calculate projection vectors and plot the results:

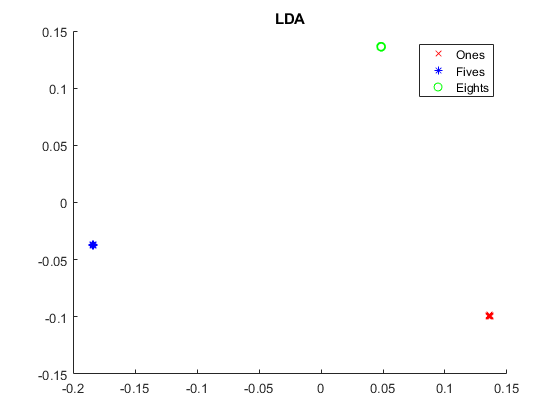
% computing the projection vectors:

[v1,d] = eig(inverted\_SW);

% calculating score

score = (all\_data\*v1(:,1:2));

**Results:**



From what we can see on the figure, LDA managed to classify classes and distinguishes the data. Compared to PCA, LDA treats data in slightly different way. By looking at PCA centroids, I believe that PCA performed better than LDA in this particular problem. Also, I believe that there is a bug somewhere in my LDA implementation, the results are not what I expected.

**Q3. Two-class problem classification**

For this question, I had to separate data for “Fives” from the rest of the data and treat it as two-class classification problem.

**Support Vector Machine (SVM)**

**SVM with linear kernel**

**SVM with RBF (radial basis function) kernel**

**Neural Network classifier**