# Project reports

# SYS-828


## Mohammadamin Abbasnejad


**November 2018**

## 1 Introduction

Over recent years identifying an individual has played an important role in biometric systems while this task has always been a difficult task. Specially in security purposes recognition of persons (face) is very important. This is the core problem of recognition systems, recognition of individuals by computers. To tackle this problem we need developing systems to recognize persons accurately, it can be performed based on physical traits specific to each individual such as fingerprints, eye recognition or facial recognition. In this project we will focus on facial recognition, a field of study particularly interesting by its relative simplicity of use. Indeed, unlike ocular recognition or fingerprint analysis, face recognition requires fewer constraints such as a device in contact with the subject to identify. This feature of facial recognition makes it a popular area for researchers.

## 1.2. **Application domain**

The capability of automatically recognizing faces from images has opened up multitude of possibilities I different domains. It can be from surveillance cameras placed in public places for criminal activity detections to advertisements, game and entertainment industry and even medical devices industry.

In robotics and smart cars interacting with computers has gained a lot of interests, effectively recognizing the person can lead to higher security in cars and also comfortability. For instance, this technology can capable scientists to find behaviour of persons according to muscles movements of face or eyes and apply this techniques in cars for alarming systems.

## I.3. problematic

Face recognition presents multiple challenges, during image acquisition images can be taken under uncontrolled conditions. The most challenging problems in image acquisition is changing brightness, the pose of the face, accessories (glasses, caps, hats), beards or the quality of the image. Especially when we are dealing in the wild and the real time scenarios makes this the process recognition much more challenging.

## I.4. objectives

In this project, we will focus on extracting the characteristics of samples from a facial image database and optimization of two classification algorithms. Our goal will be to determine what is the, or what are the combinations of approaches to obtain the best performances of classification of the faces according to the complexity (and by extension of the computation time). we will only focus on representation and classification steps, using images of already segmented faces. For the representation of the data, we will start from a representation of the images in the form of a vector of the gray levels of each pixel on which we will compare 2 methods of reduction of the dimension: principal component analysis (PCA) and linear discriminant analysis (LDA).

Next, we will experiment with two methods of data classification, one of generative type, K-Nearest Neighbor or K-NN, and the other of discriminative type, Support Vector Model or SVM. The difference between the two types of classification is as follows: the generative models first try to find an optimal model of the classes of the samples of the database. The classification of a new sample is then made by comparing it to all the classes and assigning it to the one closest to it. Discriminatory models will try to define decision boundaries for optimal separation of classes. The classification of a new sample is then based on the position of this sample in the feature space with respect to these decision boundaries.

In summary the main objective of this laboratory is to compare the K-NN and SVM classification methods and to analyze the influence of dimension reduction methods on the performance of these classifiers.

### I.5. Structure of the document

I will see in detail the various techniques used for the extraction of the characteristics of the images (PCA and LDA) and then classification of the data (K-NN and SVM) and using CNNs. To write this report I tried to visualize parameters to gain better insight from data visualization, in each experiment I tried to conclude by presenting the experimental results and providing a critical analysis of the different techniques.

## 2 Extraction of characteristics

### 2.1 Principal Components Analysis

PCA (Principal Components Analysis) is a mathematical method for reducing the dimensionality of data. It consists of breaking down a set of data into a series of decorrelated variables called principle components by projecting it into a lower-dimensional orthogonal subspace whose axes are chosen so as to maintain a maximum level of variance of the data. A common use of PCA is the visualization and analysis of a large database, made possible by projecting it into a small space of 1.2 or 3 dimensions.

## 2.2. Linear Discriminative Analysis

Linear Discriminative Analysis (LDA), like PCA, is a mathematical method for reducing dimensionality. However, in LDA, instead of looking at the magnitude of the variance and keeping the components that will maximize it, we are interested in the classes present in our database and we look for projection axes that will separate them optimally by maximizing inter-class dispersion.

## 3 Methodology

The experimental protocol for the comparison of PCA and LDA techniques is as follows: after extracting a representation of the images in the form of a grayscale vector, we will apply the PCA and LDA transformations. In each case, we will first analyze the effect of the two transformations on the dispersion of the data, in the second step, we will reconstruct the faces from these new representations with a variable number of characteristics and we will analyze the quality of the images obtained.

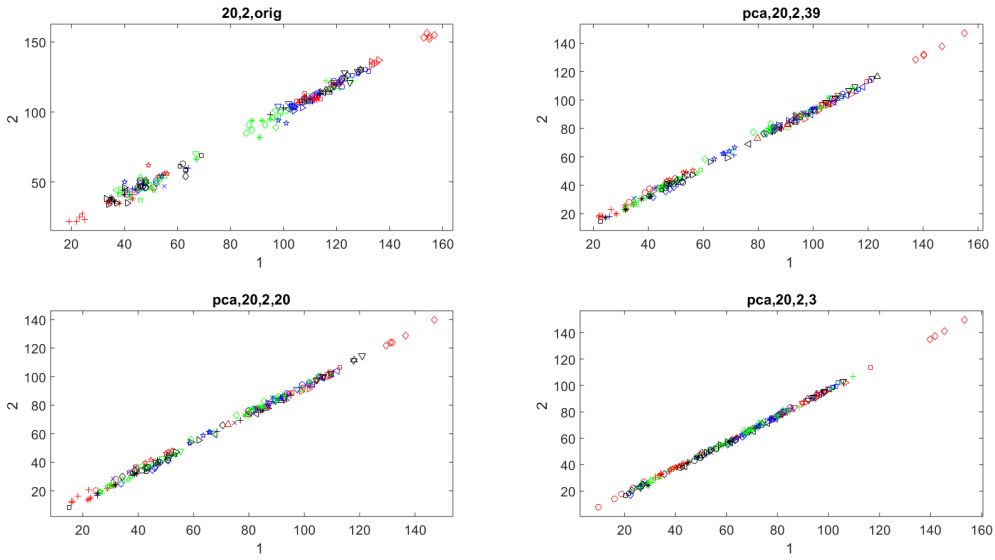
### 3.1 PCA

I begin our comparison by comparing the PCA and LDA in extraction techniques. Itry to apply PCA and LDA on our data, find and visualize different parameters and try to reconstruct our original dataset by PCA and LDA operator, then I will make comparison about both approaches, benefits and disadvantages. As in slides and explanation of the assignments all procedures were clearly explained I directly go through experimental results and I tried to visualize parameters because I believe that they worth a thousand words. We need to

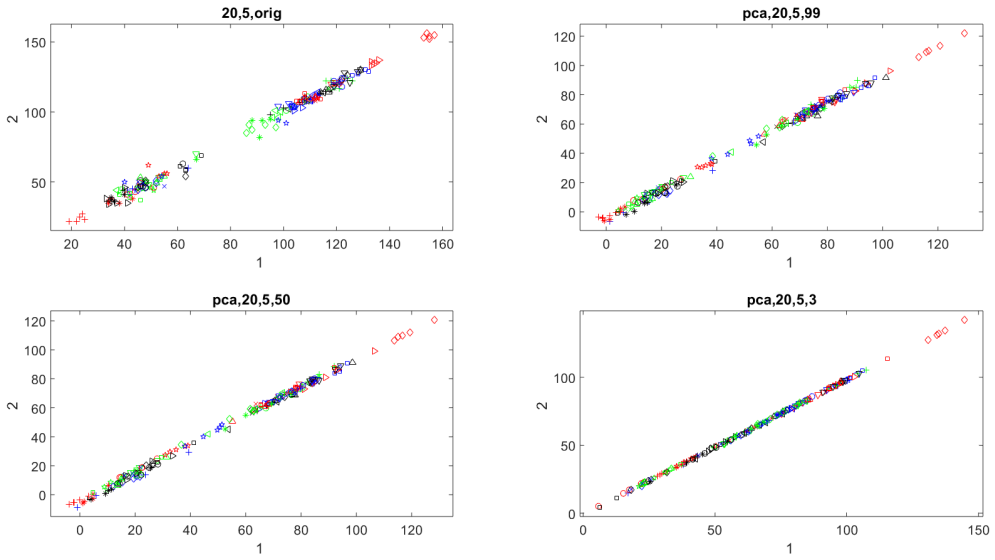
implement this experiments both for showing the effective of PCA and LDA and then use them to explore their impacts on classifications,

PCA

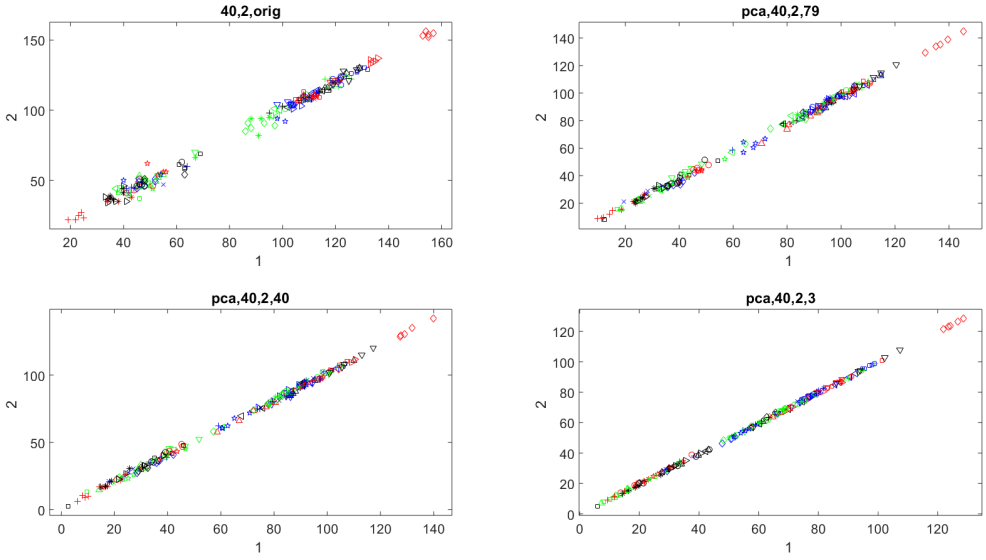
Figure.1. Visualization of PCA number of images per individual, number of selected components.



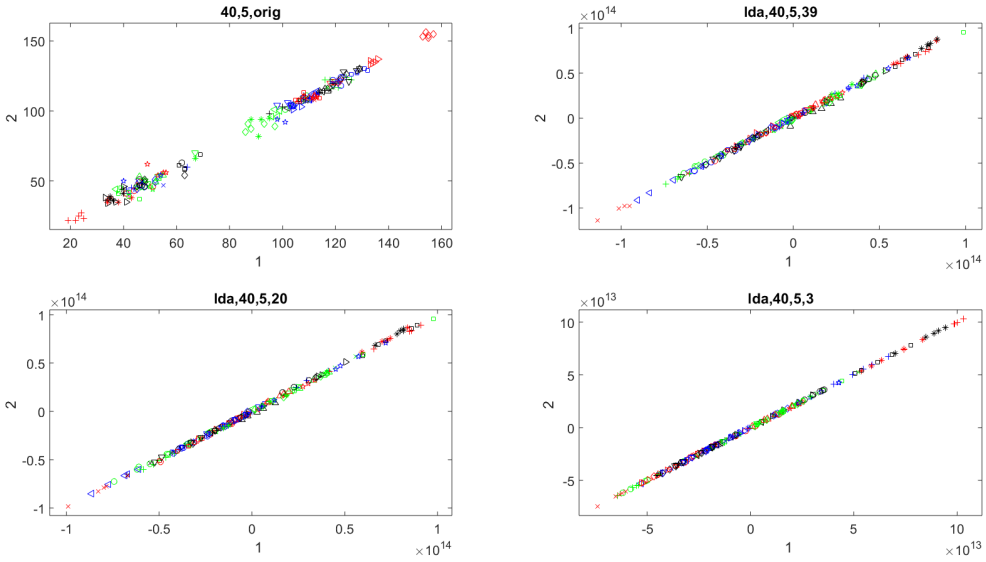

(20-2-2)



(20-2-5)

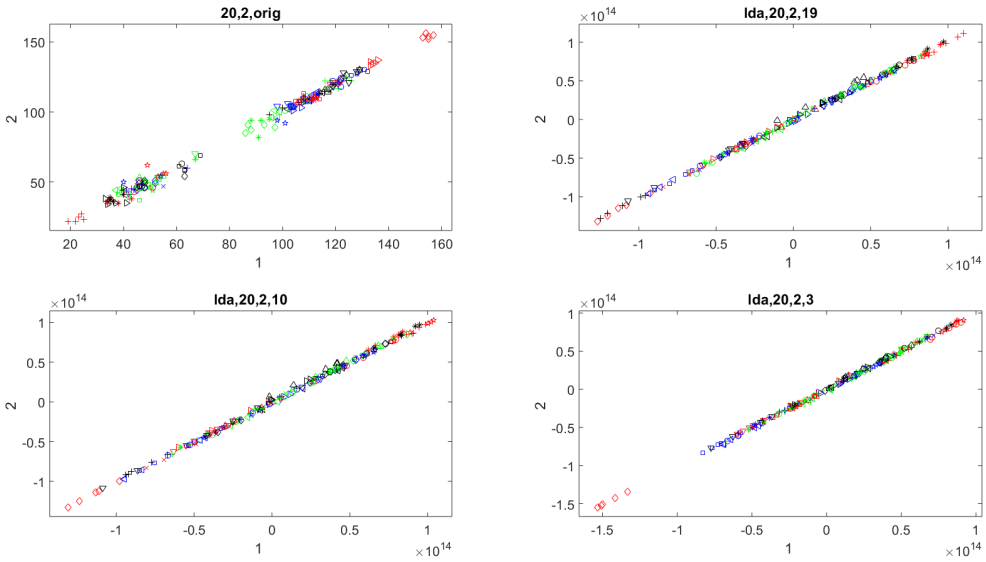

40,2,orig
pca,40,2,79
pca,40,2,40
pca,40,2,3

(40-2)

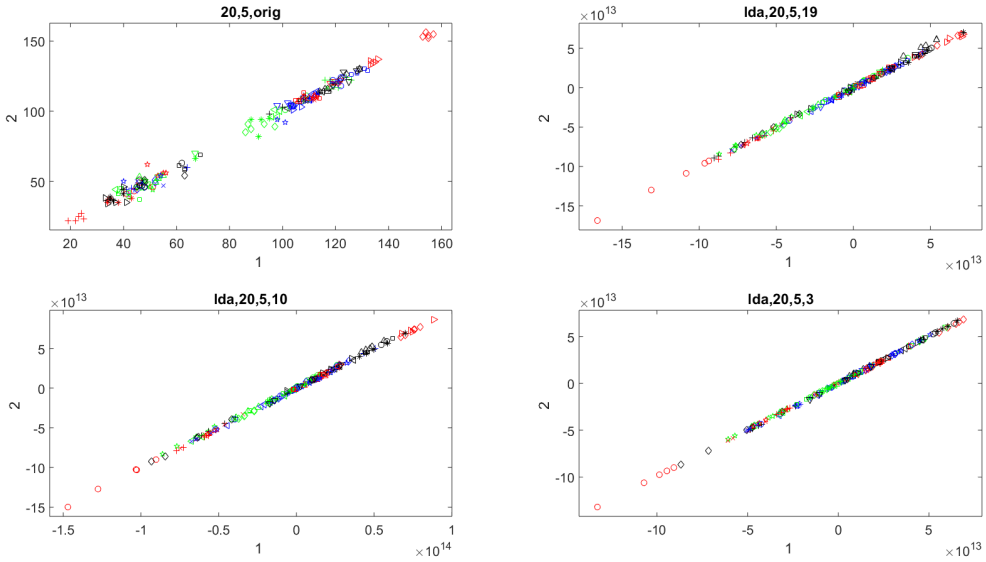

40,5,orig
lda,40,5,39
lda,40,5,20
lda,40,5,3

(40-5)

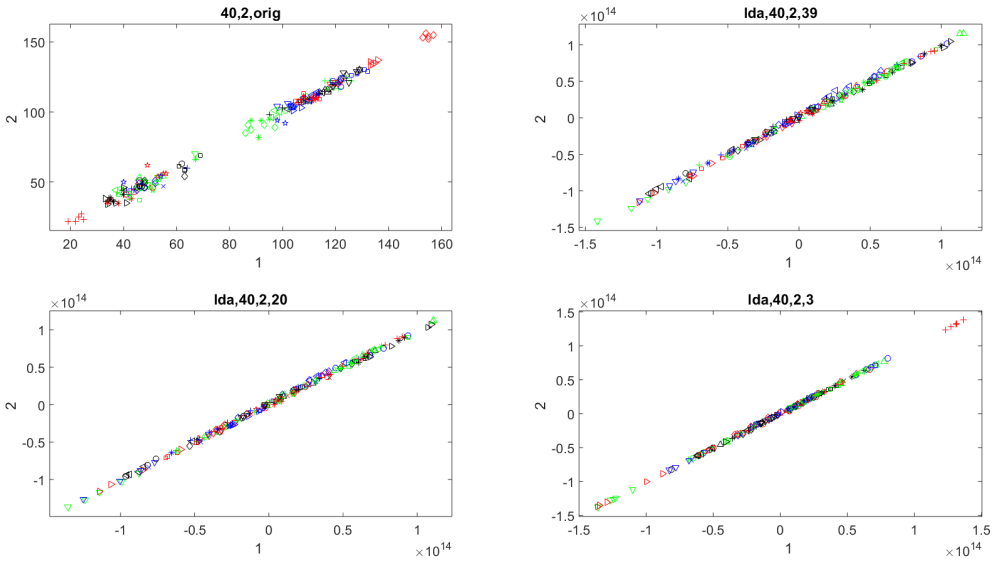
LDA


20,2,orig
lda,20,2,19
lda,20,2,10
lda,20,2,3

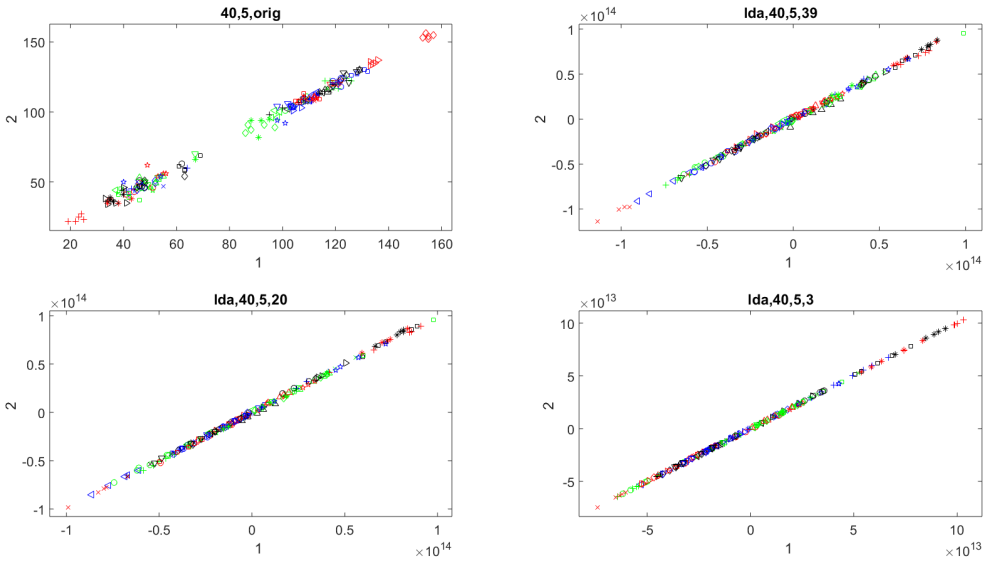
(20-2)


20,5,orig
lda,20,5,19
lda,20,5,10
lda,20,5,3

(20-5)




(40-2)




(40-5)

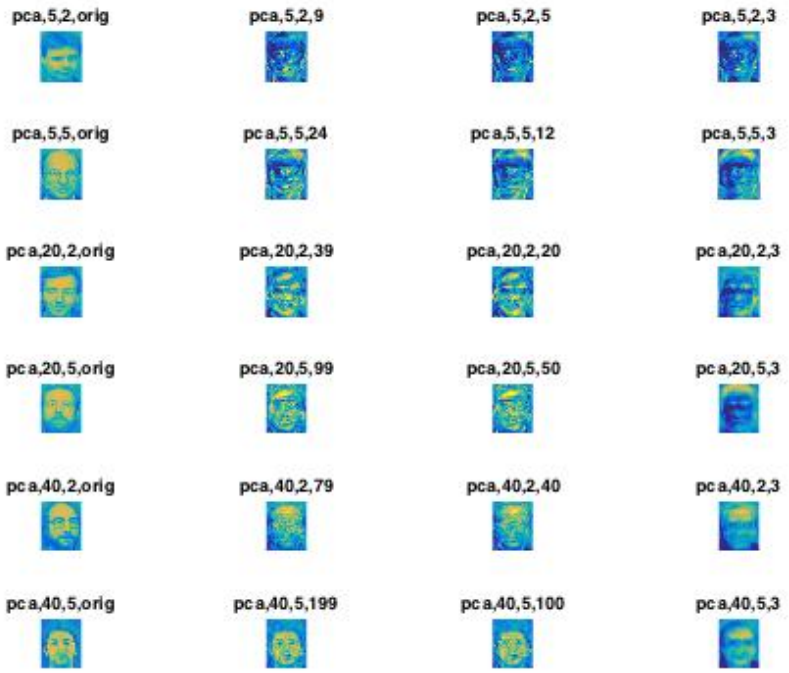
The main advantage of PCA is that it is possible to control the amount of original information retained. In fact, during the selection phase of the eigenvectors on which we want to project our data, these are classified according to the magnitude of the variability after projection and it is therefore possible to keep only a minimal number of vectors, in order to have a final

retained variability greater than or equal to a threshold. This ability to limit the loss of information makes PCA provide better results when reconstructing images from the feature vector. However, focusing on the vectors that maximize the magnitude of the variability makes it possible that the most discriminating vectors are not retained during dimensionality reduction. Moreover, since PCA does not use class data, we see that this method is not advantageous for classification tasks and it is precisely on this type of problem that LDA stands out.

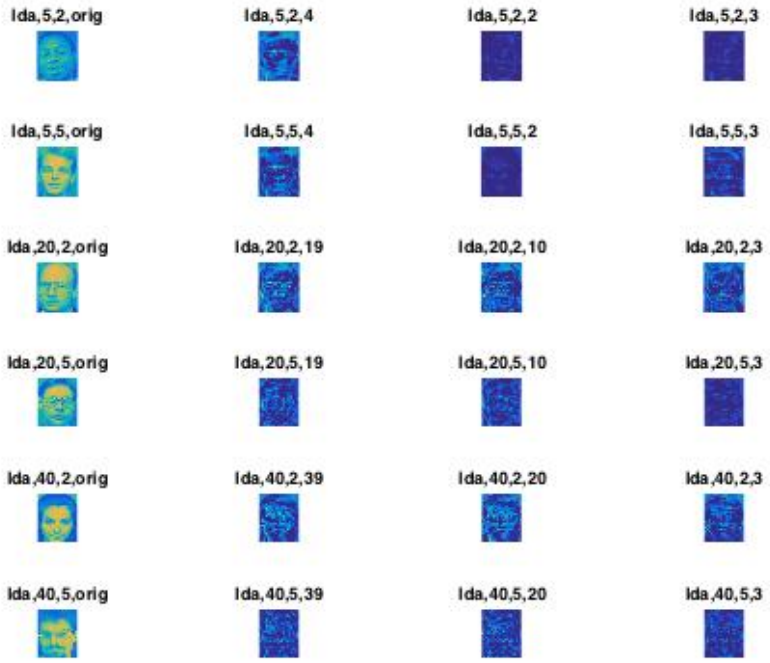
Indeed, whereas PCA is limited to reducing the space of the characteristics, LDA goes further and uses class information in the process of selection of the eigenvectors. Instead of trying to select the eigenvectors that will maximize the magnitude of the variability of the dataset, LDA will identify the most discriminating eigenvectors in order to maximize inter-class distance and minimize intra-class distance.

While PCA seeks to reduce the size of the data by looking for projection axes limiting the loss of variability, LDA will look for projection axes allowing optimal separation of classes.

Face reconstruction after PCA application. Figure.2




reconstruction after LDA application of face. Figure.3



In Figs. 2 and 3, we can see original images compared to reconstructed images with a variable number of components retained. The images are illustrated according to: method, total number of individuals, number of images per individual, number of selected components. It is very clear that with a number of main components and sufficient samples, the images reconstructed with PCA are closer to the original images than the images reconstructed with LDA.

Since PCA and LDA have different mechanisms and benefits, it is not uncommon to see the two methods combined. PCA is then used first to reduce dimensions, and LDA is used later to optimize class separation.

if LDA is particularly efficient for the resolution of classification problems, the most discriminating vectors are not necessarily the most information-carrying vectors and the loss of information after projection greater than with PCA makes LDA less efficient. when reconstructing data.

## 4. Comparing K-NN and SVM

In the second stage of our project we are interested to see the influence of PCA and LDA on the performance of K-NN and SVM classifiers, I will test the two classifiers with the original data, then the reduced data with PCA and finally with the reduced data with LDA.
For the comparison of K-NN and SVM classification techniques, we will take these steps approach:

1) The first step is to split our database into 2 parts: a training set and a test set. The training set will be used to find the optimal parameters of our classification models during the training phase, which will be used in the final model in the test phase.
2) In the learning phase, we will look for the optimal parameters for our classifiers using the cross-validation method. This method consists of subdividing the training database into a training database and a validation database. The classifier is trained with the samples from the training base and then its performances are tested with the samples from the validation database. This is repeated a predefined number of times by varying the samples of the validation database.
3) Finally, the last step, the test step, is to train our classifier models using the optimal parameters found in the previous step and the complete training base, and then test its performance using the test base to determine the performance of our classifier.

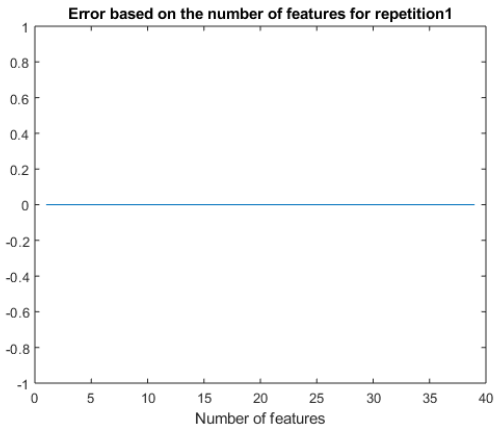
### 4.1. Performance indicators

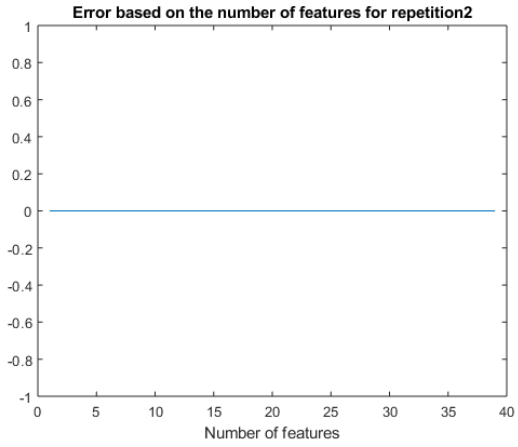
The performance indicators used to compare PCA and LDA are the number of features relevant to the classification and quality of the reconstructed images.
To compare KNN with SVM, we will use several metrics:
Time: I will compare the time spent for both the learning phase and the testing phase
Error rate: I will compare the error rates after cross-validation as well as the test error rates
Confusion Matrix: will analyze the complexity of each of the models and analyze the matrices of confusion

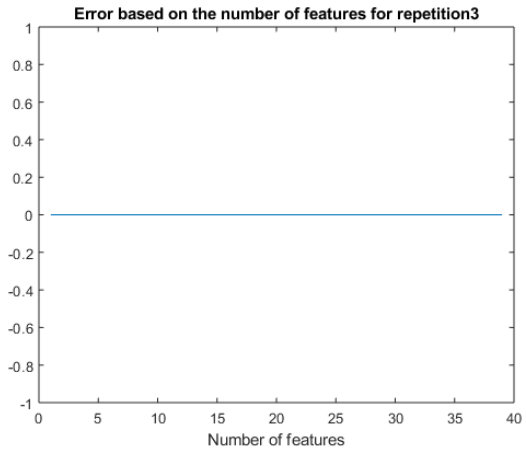
### 4.2 Selecting of number of features

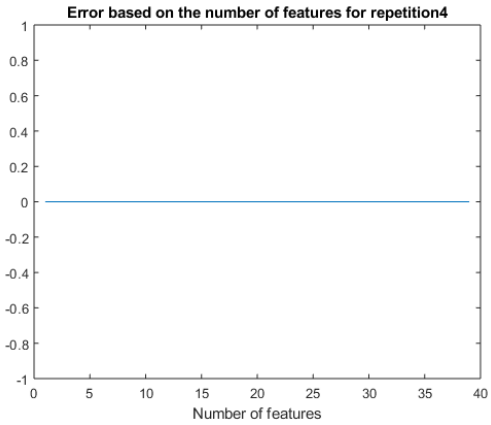
By analyzing the variation of the error of a 1NN classifier according to the number of LDA characteristics, we see that the validation error is constant around 0%. This means that with

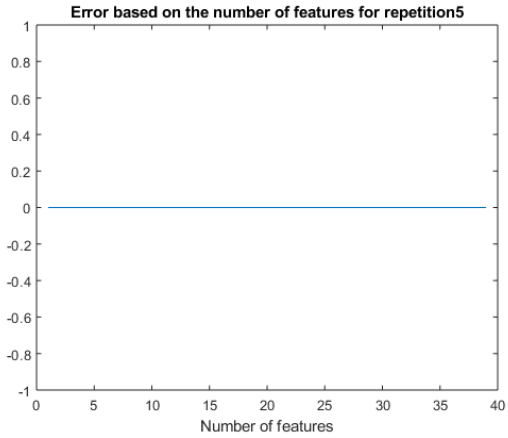
only a single LDA characteristic, it is already possible to separate the characteristics appropriately.



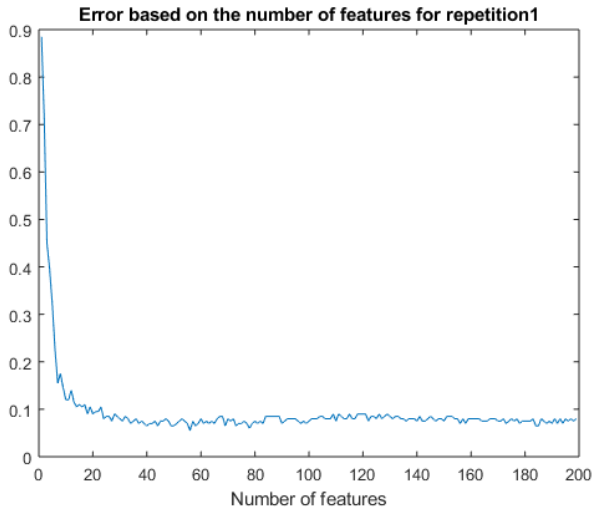



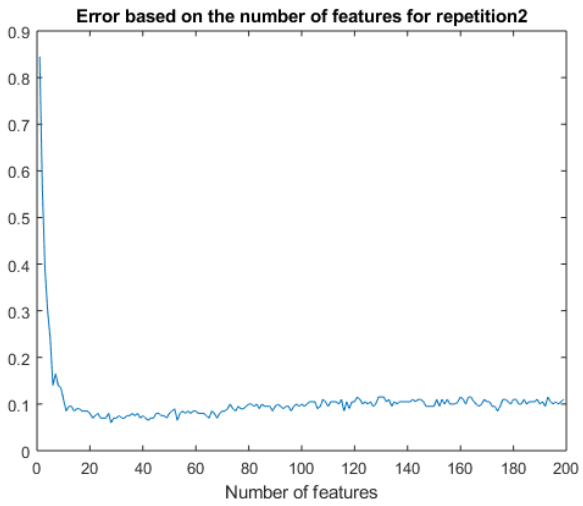



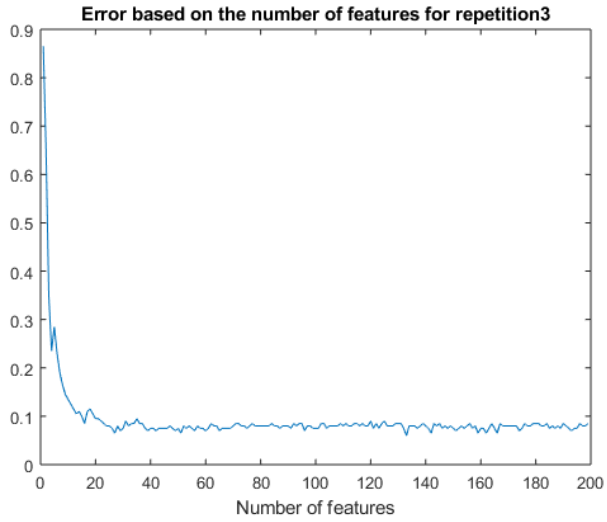


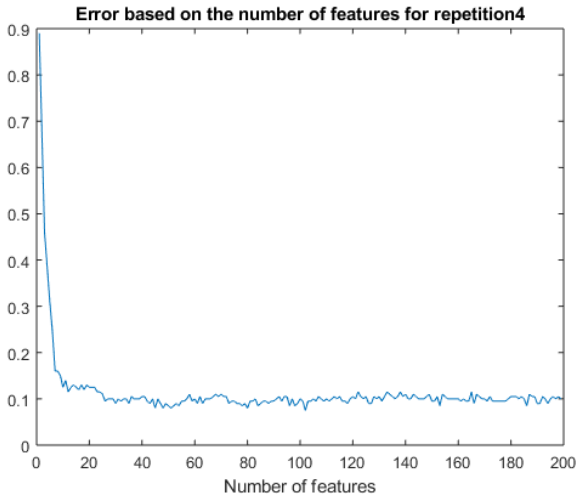




By analyzing the variation of the error of a 1NN classifier according to the number of PCA characteristics, we obtain the following curves:




Error based on the number of features for repetition2
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 20 40 60 80 100 120 140 160 180 200
Number of features


Error based on the number of features for repetition3
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 20 40 60 80 100 120 140 160 180 200
Number of features


Error based on the number of features for repetition4
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0 20 40 60 80 100 120 140 160 180 200
Number of features

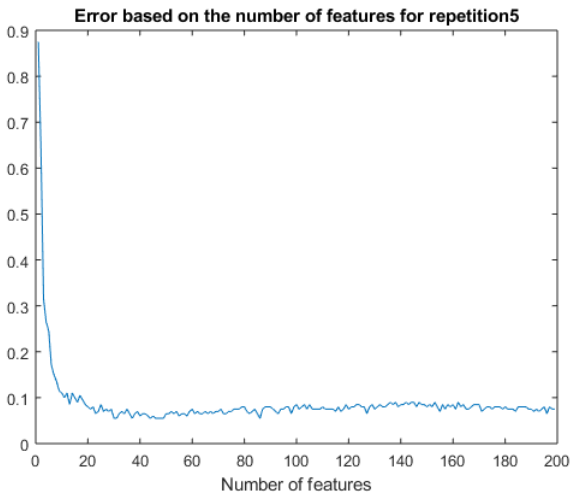
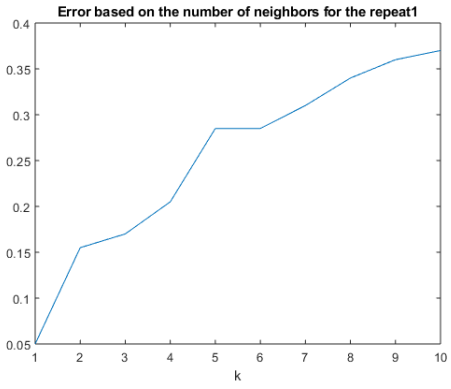
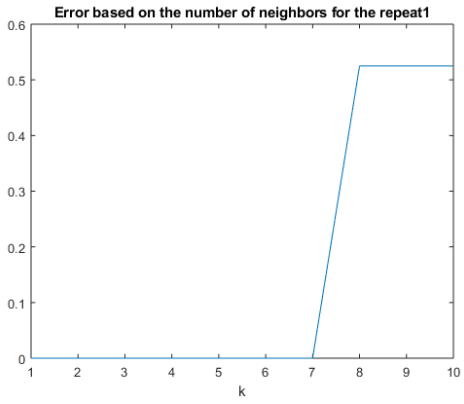


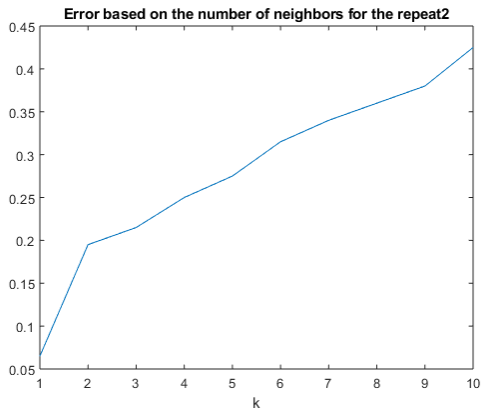

Figure.4. Validation error of a 1NN classifier based on the number of PCA characteristics

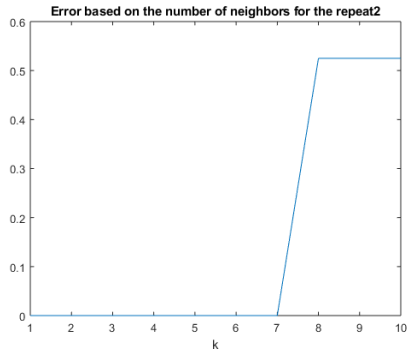
As can be seen form curves when we increase the number of PCA characteristics the error decreases and then stabilizes around a constant number. This means that after a certain point, the following characteristics only represent noise. The average value obtained for the number of optimal characteristics for PCA is around **35**.

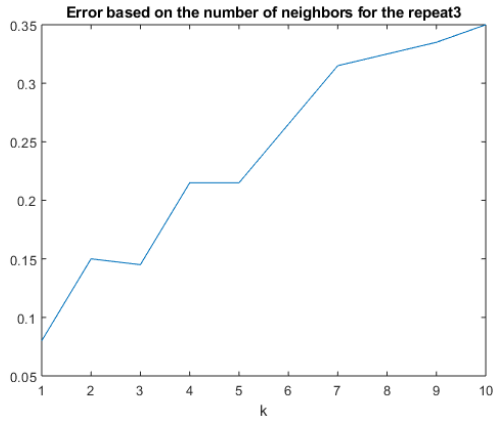
I made 5 replications and calculated the performance for kNN.

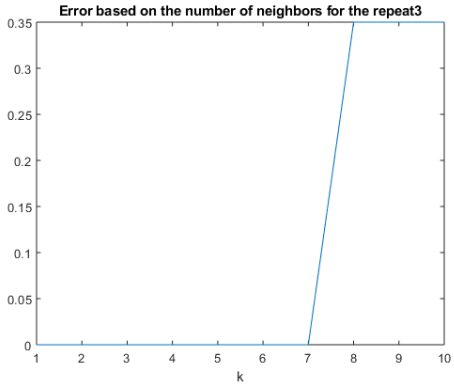


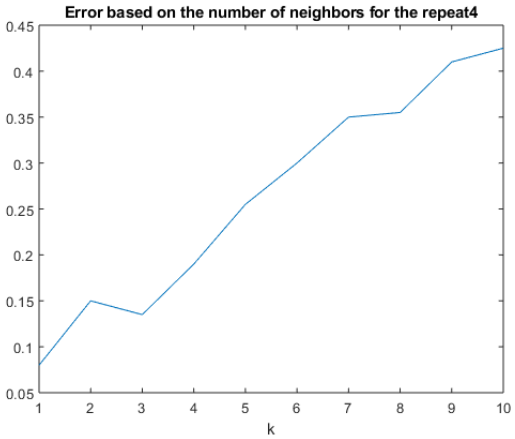



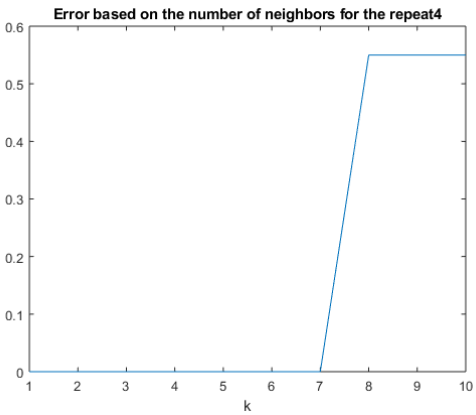

Error based on the number of neighbors for the repeat2
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
1 2 3 4 5 6 7 8 9 10
k


Error based on the number of neighbors for the repeat2
0.6
0.5
0.4
0.3
0.2
0.1
0
1 2 3 4 5 6 7 8 9 10
k


Error based on the number of neighbors for the repeat3
0.35
0.3
0.25
0.2
0.15
0.1
0.05
1 2 3 4 5 6 7 8 9 10
k


Error based on the number of neighbors for the repeat3
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
1 2 3 4 5 6 7 8 9 10
k


Error based on the number of neighbors for the repeat4
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
1 2 3 4 5 6 7 8 9 10
k


Error based on the number of neighbors for the repeat4
0.6
0.5
0.4
0.3
0.2
0.1
0
1 2 3 4 5 6 7 8 9 10
k

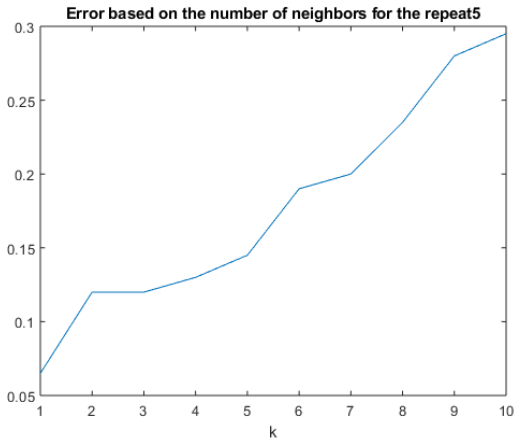
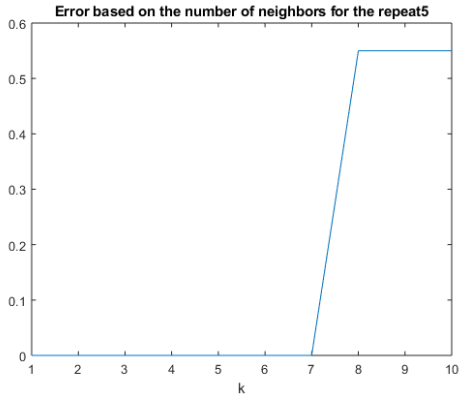



Figure: Error change in PCA as a function of K




Error change in LDA as a function of K

Plot a graph representing the training error with respect to k.

| Best number for K | K=1 |
|---|---|

In our experimental phase, we found that the optimal value of K for the K-NN classification method was always k = 1, both for PCA and LDA, and that higher values resulted in an increase in the error rate. A possible interpretation of this phenomenon is that the higher K is, the greater the probability of including samples of another class when classifying a new sample. This can be explained by the fact that PCA and especially LDA have a strong grouping function with very little overlap near borders. Indeed, if class boundaries were less clear, a higher K value would be required to obtain good classification accuracy.

## 4.4 Choice of hyperparameters

An analysis of the data showed us that applying of the PCA and LDA methods, has a direct impact on C and γ. We found that trying exponentially growing sequences of C and γ is a practical way to identify good parameters. After applying PCA due to scale of values we selected the parameters as fallows:

| $\gamma$ | 0.1 | 1 | 10 |
|---|---|---|---|

| C | 0.1 | 1 | 10 | 100 |
|---|---|---|---|---|

After applying LDA, we encountered with a more complicated problem. Indeed, the scale of

values were much more higher: the intra-class difference is very small and the inter-class difference is by comparison very large. This makes the initialization of γ and C very difficult because the classes are so well separated that one arrives very quickly in an overfitting situation during the training phase, which is characterized by error rates of 0% in validation and which leads to the selection of non-optimal values of C and γ.
By increasing the values of C and γ , we fixed the values as fallows:

| $\gamma$ | $10^{23}$ | $0^{24}$ | 10 |
|---|---|---|---|

| C | $10^{12}$ | $10^{13}$ | $10^{14}$ |
|---|---|---|---|

| Classifier | Dimensionality Reduction Technique | Training Time | Learning Error | Classification Time | Test Error | Final Classification |
|---|---|---|---|---|---|---|
| K-NN | | 18.23 | 6.9 | 0.5 | 5.6 | 94.4 |
| K-NN | PCA | 10.4 | 7.1 | 0.25 | 4.9 | 95.1 |
| K-NN | LDA | 19.1 | 0 | 0.9 | 4.2 | 95.8 |
| SVM | | 418 | 8.1 | 1.8 | 5.3 | 94.7 |
| SVM | PCA | 221 | 7.9 | 4.1 | 6.1 | 93.9 |
| SVM | LDA | 94 | 3 | 3.8 | 5.9 | 94.1 |
| Combination | | - | - | 30.2 | 5.8 | 94.2 |
| Combination | PCA | - | - | 10.4 | 6.1 | 93.9 |
| Combination | LDA | - | - | 5 | 5.2 | 94.8 |

According to different configuration we can gain these information:

The first thing that comes to mind is the learning times of SVM are generally longer than for K-NN. It can be explained that number of classes (40) requires the learning and optimization of 40 SVM classifiers 'One Against All' and the relatively small number of samples (5 per class, 200 for learning and 200 for testing) explains the speed of execution of K-NN.
SVM is theoretically the classification method that generalizes the best because, as a generative model, it is less dependent on training data than K-NN. However, we get slightly lower classification performance on test data than K-NN. This can be explained by the optimization problems of the kernel parameters.

The performances of K-NN on the basis of non-reduced dimension learning are quite close to those on the basis of learning after PCA reduction. However, we see that K-NN generalizes better with a PCA-reduced basis. This can be explained by the fact that the characteristics not retained by PCA represented only noise that could negatively influence the distances calculated by K-NN.

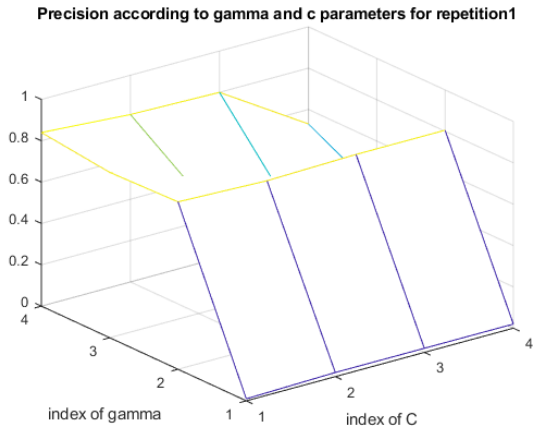
The performance of classifiers combined by majority voting system shows that the simple combination of classifiers does not necessarily obtain better results than all the classifiers taken unitarily. This is particularly the case when one has a majority of inefficient classifiers who will negatively influence the final classification by their vote. By making comparison between the KNN and SVM classification methods we can highlight some interesting points.

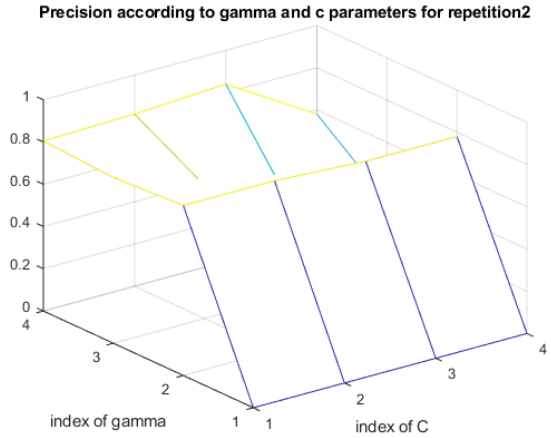
First of all, KNN has the advantage of simplicity of implementation, the only parameter of this method to define and optimize is the variable k, which represents the number of nearest neighbors. To classify a sample, it is enough for a given k to calculate the distance between this sample and the rest of the samples and to determine the class according to the class of the k nearest neighbors.

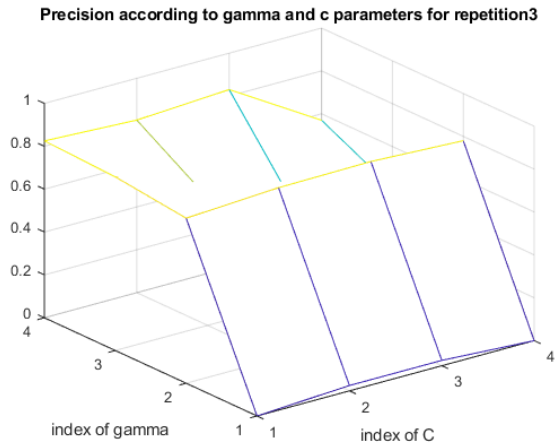
The disadvantage of KNN is that it is difficult to apply to large databases because the model requires saving all the samples for the calculation of distances to classify each new sample. Moreover, even if the Euclidean distance is the most used, it is not necessarily the best and if it becomes interesting to test several distances to improve the performance of the model it takes the complexity and the time of the phase of "Learning" of k.

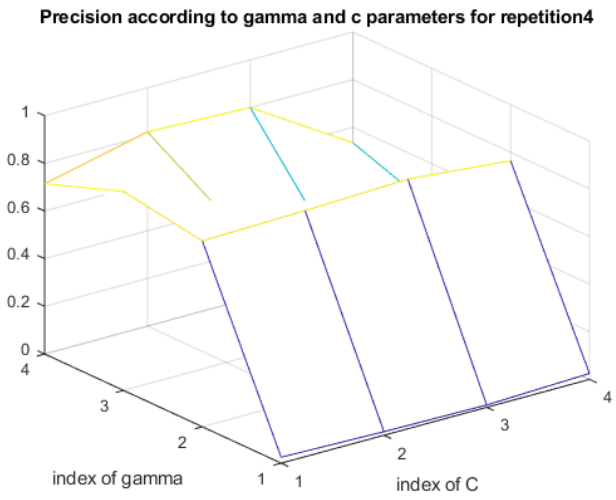
SVM is a more complex method to understand and implement than KNN but we can take advantage of kernels, thus we have more flexibility. This makes it possible to work in spaces of larger size that are more conducive to separating two classes that would not be linearly separable in their original space.

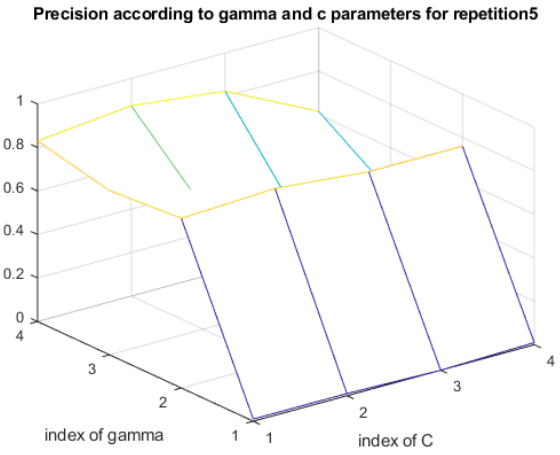
The dark side of SVM is that the final performance of the classifier is very dependent on the parameters of the kernel, for example C and gamma in the case of a Gaussian kernel. In the quantitative analysis of the method we have notably seen the difficulty of defining parameters that do not lead to an overfitting situation on the training data. In addition, SVM is only a binary classifier. To use it in the context of multi-class data, it is therefore necessary to build a set of binary classifier by adopting "one-on-one" or "one-against-all" approaches. This increases the processing time and the overall complexity of the model.

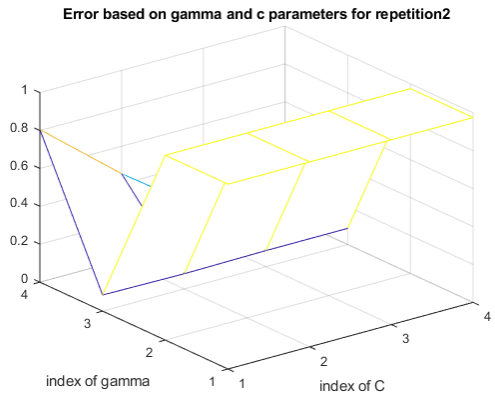

Precision according to gamma and c parameters for repetition1
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C

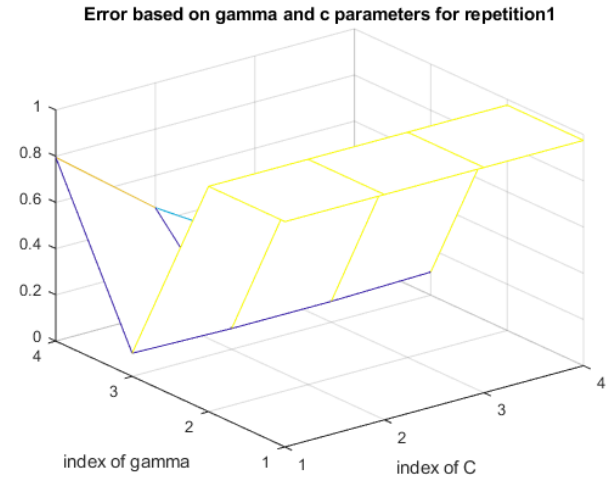

Precision according to gamma and c parameters for repetition2
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C

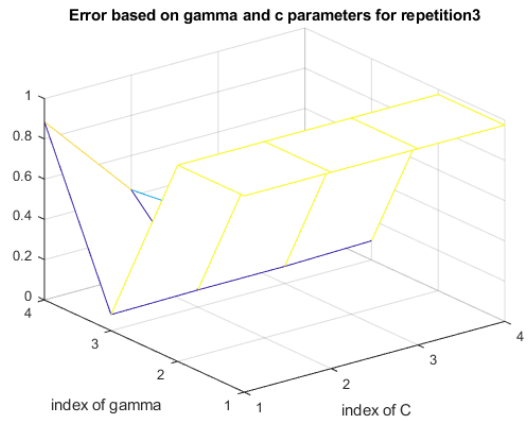

Precision according to gamma and c parameters for repetition3
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C

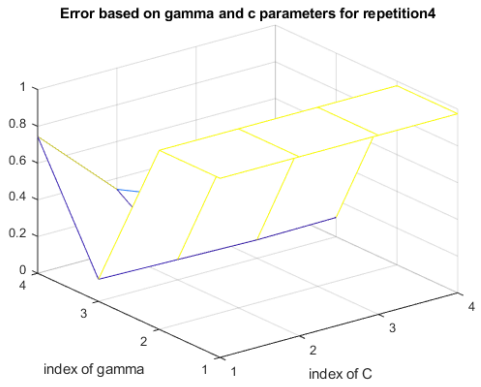

Precision according to gamma and c parameters for repetition4
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C

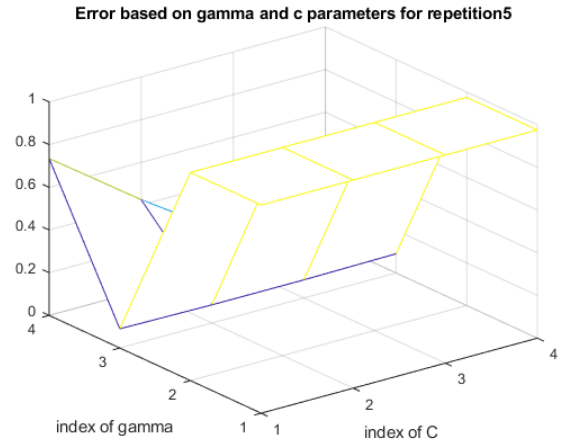

Precision according to gamma and c parameters for repetition5
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C


Error based on gamma and c parameters for repetition2
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C


Error based on gamma and c parameters for repetition1
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C

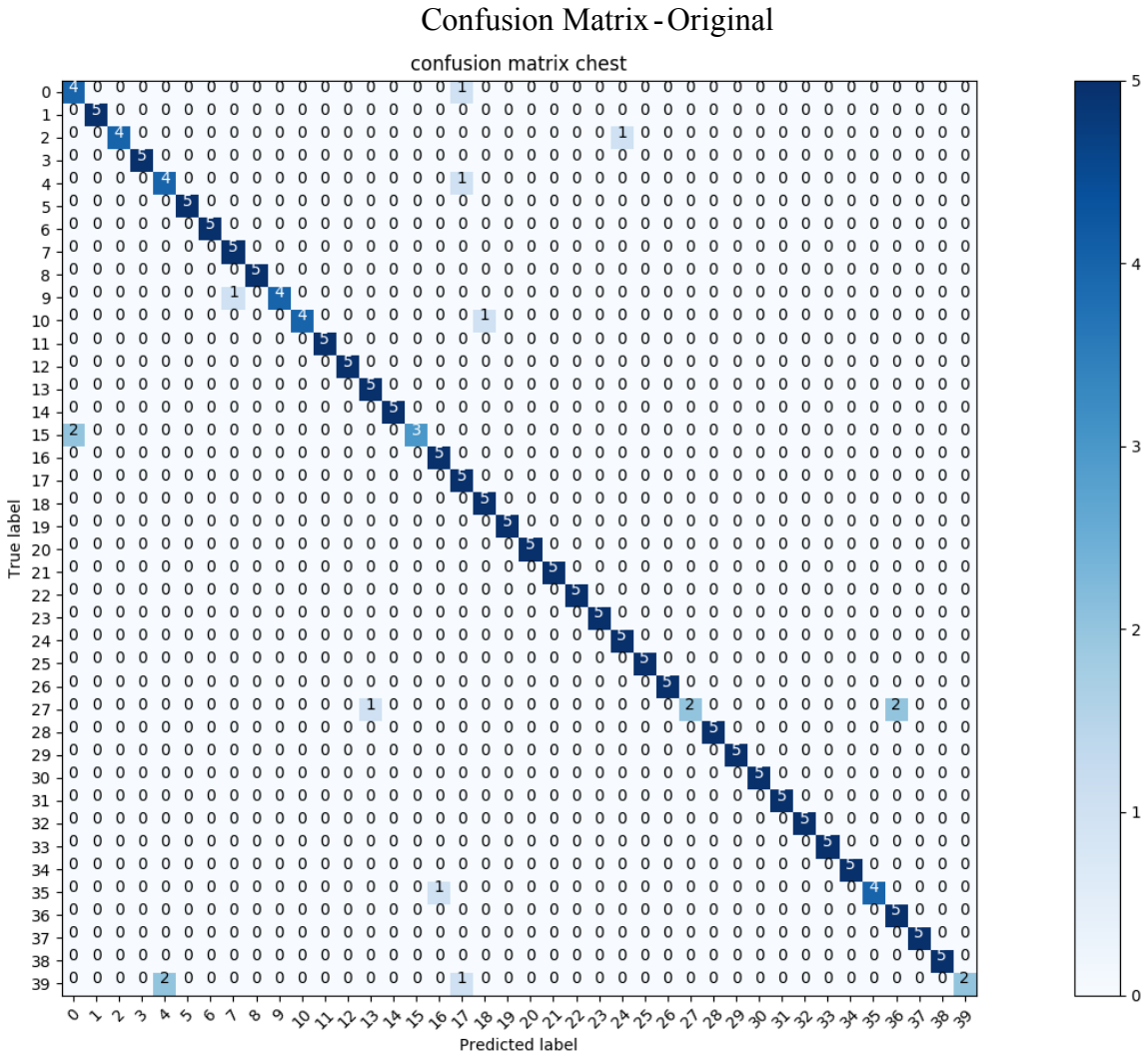

Error based on gamma and c parameters for repetition3
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C


Error based on gamma and c parameters for repetition4
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C

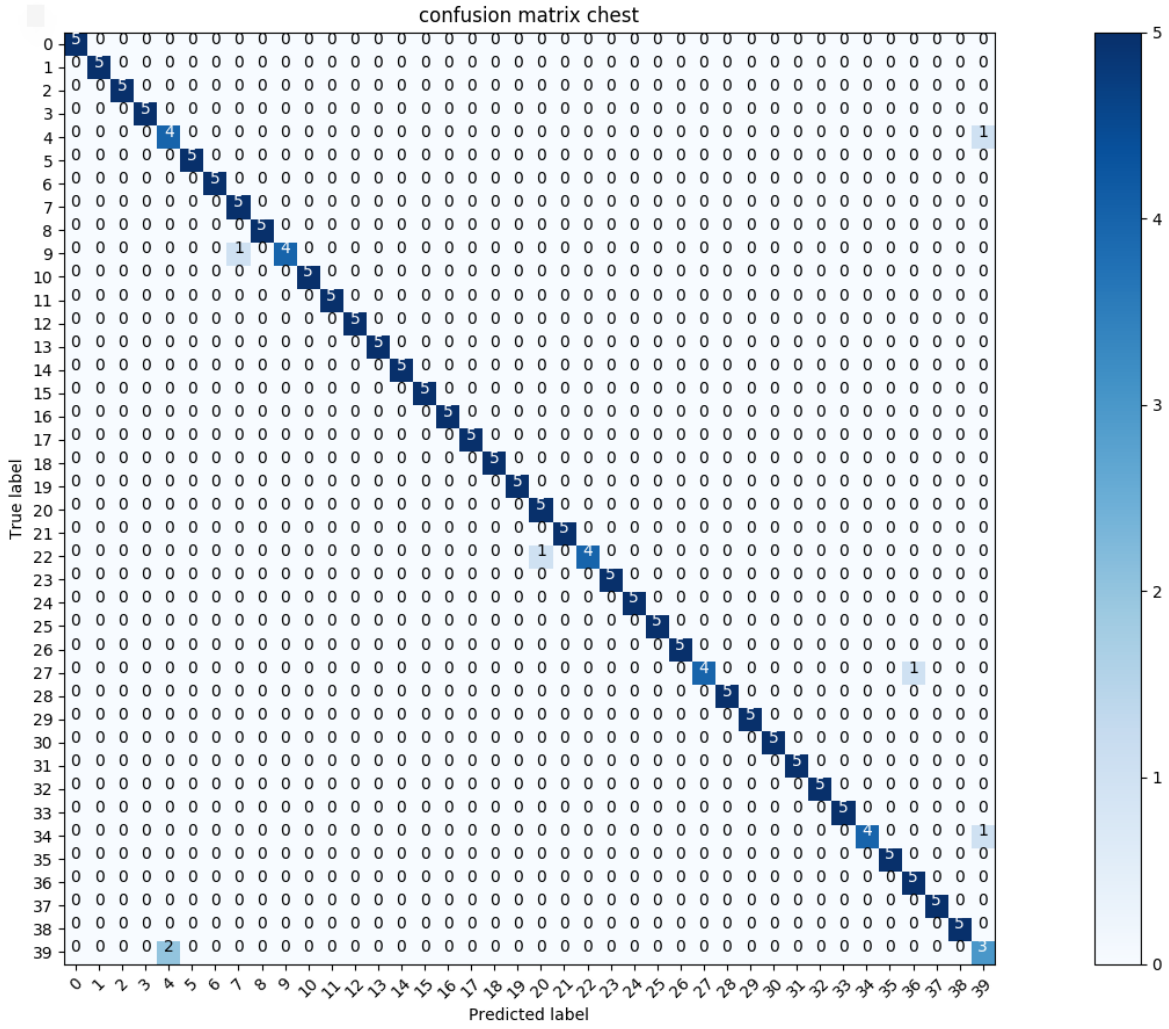

Error based on gamma and c parameters for repetition5
1
0.8
0.6
0.4
0.2
0
4
3
2
1
index of gamma
1
2
3
4
index of C

## 4.5 Confusion Matrix

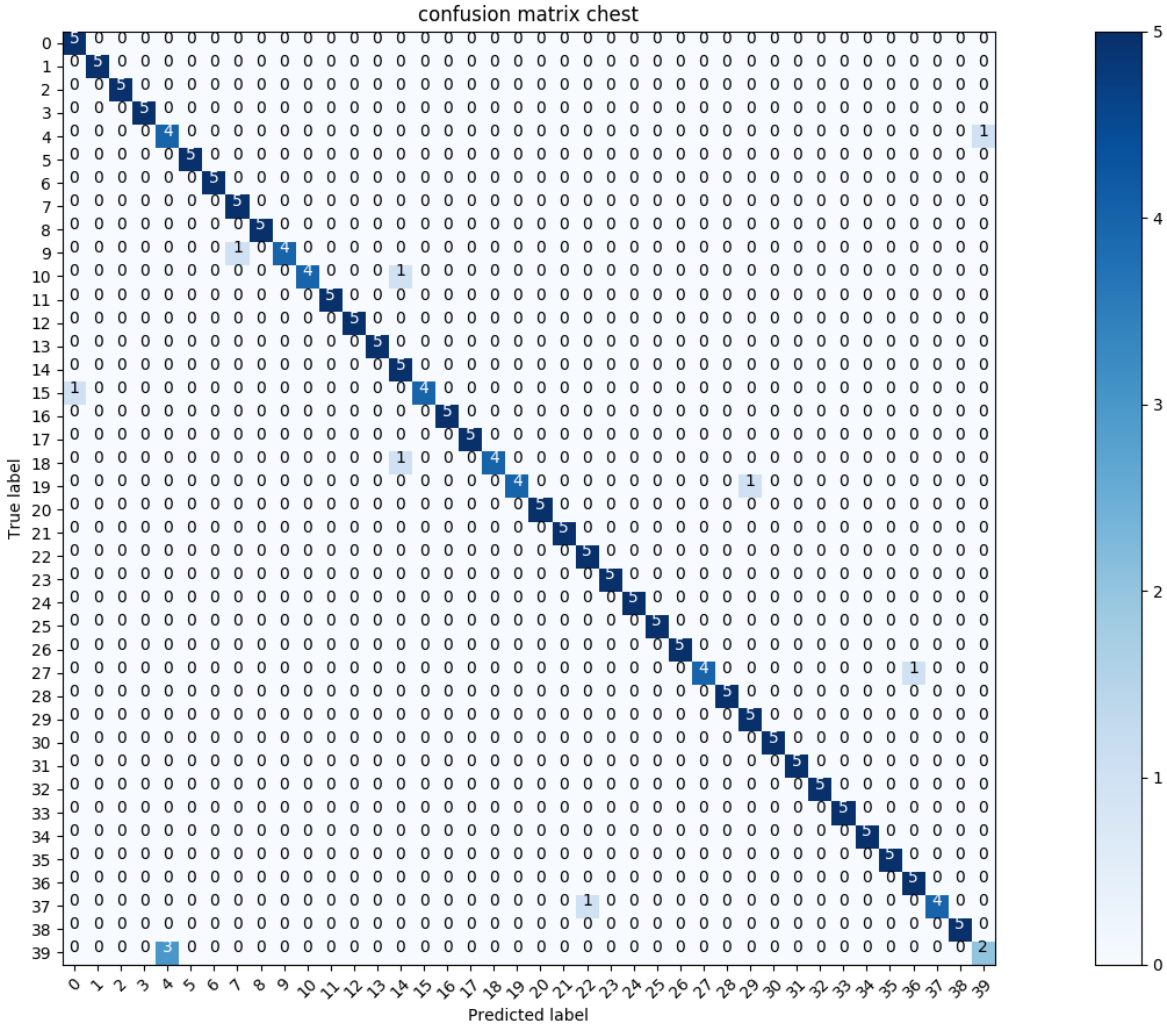
In this part we illustrate some of the confusion matrixes, the confusion matrix describes that performance of classification model, The diagonal elements represent the number of points for which the predicted label is equal to the true label, while off-diagonal elements are those that are mislabeled by the classifier.



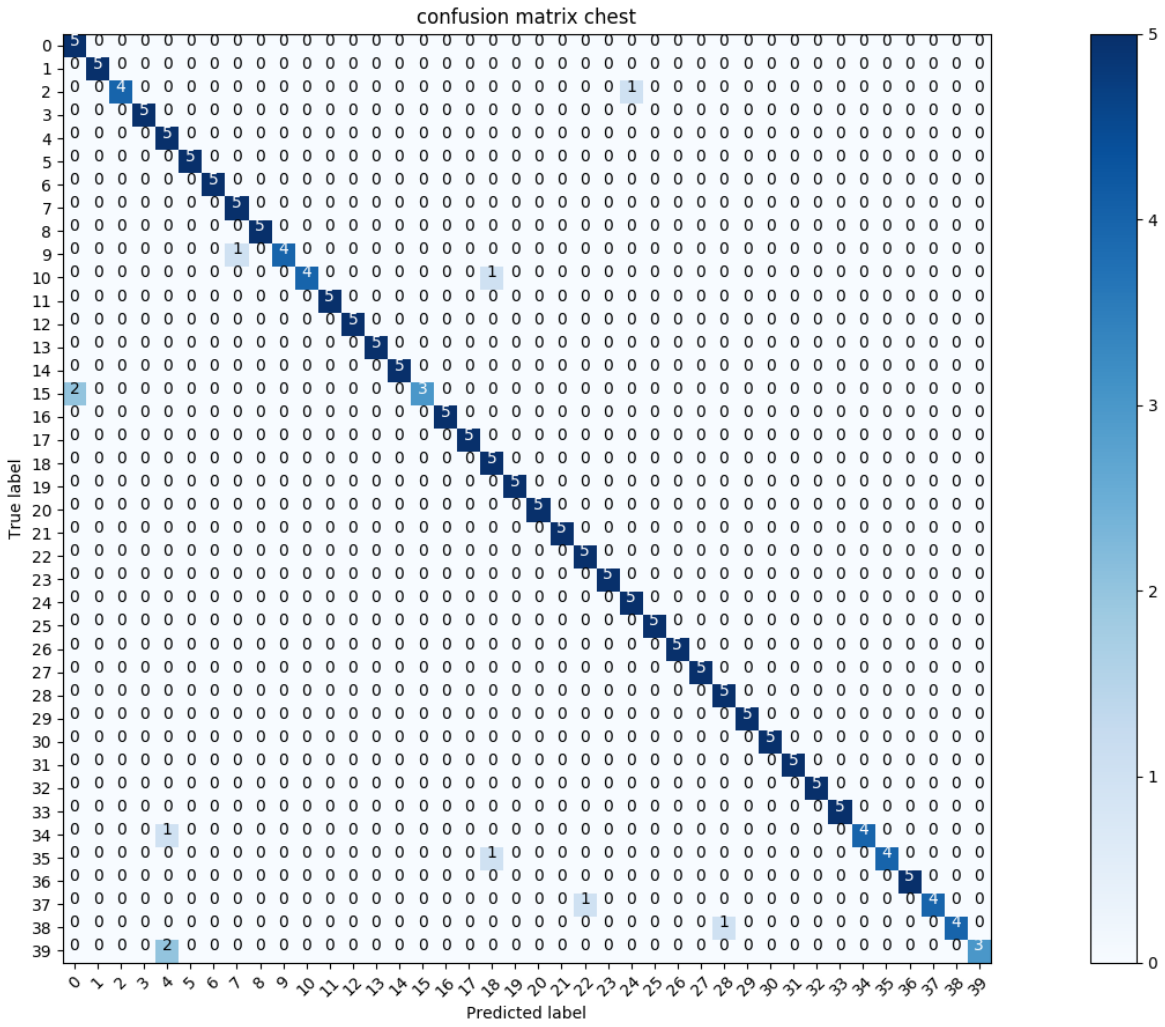
Confusion Matrix – K-NN by LDA



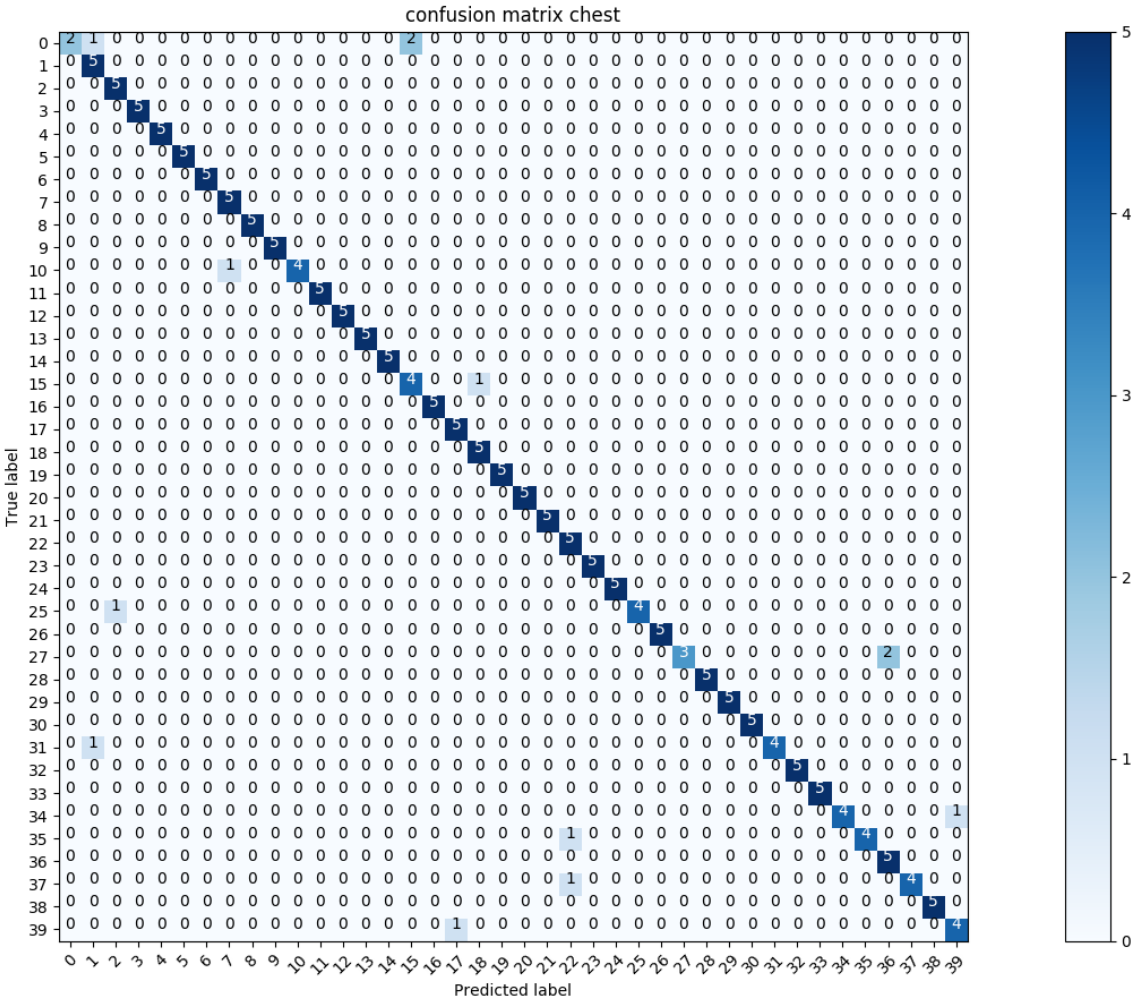
Confusion Matrix – K-NN by LDA



Confusion Matrix - SVM by LDA



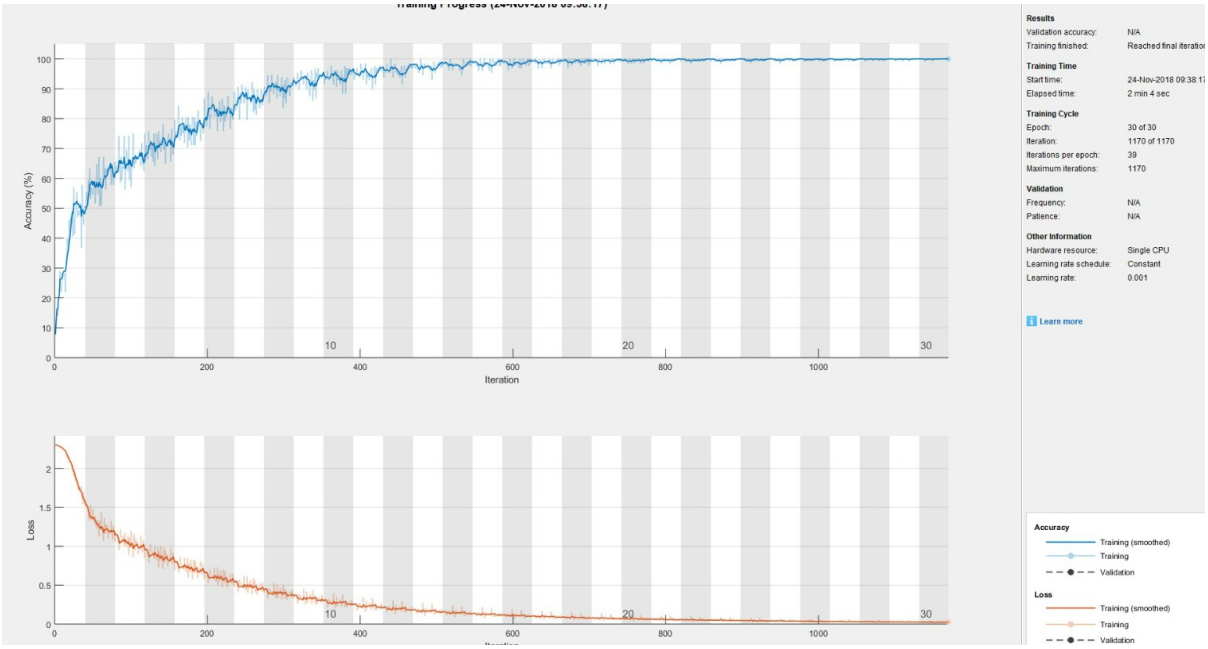
Confusion Matrix - Combination by PCA



## 5. Experiments with CNN

### ADAM

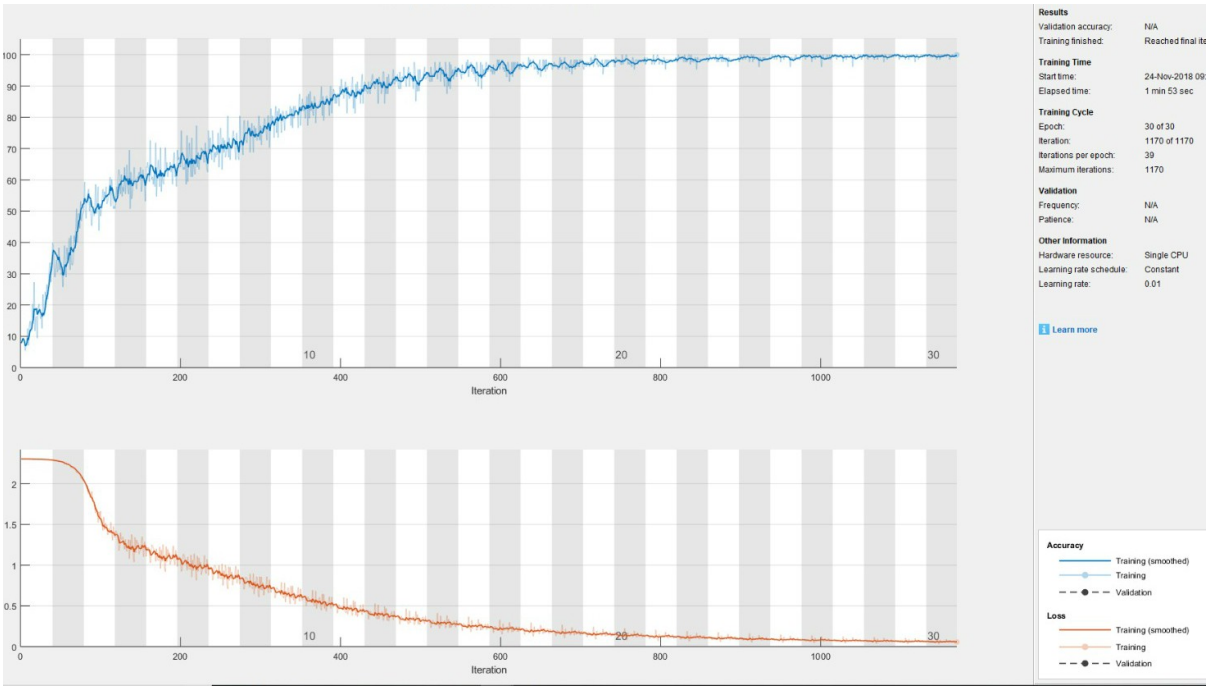
### ACC (ADAM) : 0.9856




| Epoch | Iteration | Accuracy | Loss |
|---|---|---|---|
| 1 | 1 | 7.81% | 2.3026 |
| 2 | 50 | 54.69% | 1.4782 |
| 3 | 100 | 57.03% | 1.1930 |
| 4 | 150 | 72.66% | 0.8085 |
| 6 | 200 | 82.81% | 0.5848 |
| 7 | 250 | 90.63% | 0.4471 |
| 8 | 300 | 91.41% | 0.3719 |
| 9 | 350 | 93.75% | 0.3724 |
| 11 | 400 | 95.31% | 0.2553 |
| 12 | 450 | 95.31% | 0.2084 |
| 13 | 500 | 97.66% | 0.1495 |
| 15 | 550 | 99.22% | 0.1298 |
| 16 | 600 | 98.44% | 0.1109 |
| 17 | 650 | 96.88% | 0.1588 |
| 18 | 700 | 100.00% | 0.0693 |
| 20 | 750 | 98.44% | 0.0661 |
| 21 | 800 | 99.22% | 0.0582 |
| 22 | 850 | 100.00% | 0.0384 |
| 24 | 900 | 100.00% | 0.0490 |
| 25 | 950 | 98.44% | 0.0671 |
| 26 | 1000 | 100.00% | 0.0306 |
| 27 | 1050 | 100.00% | 0.0256 |

| | | | |
|---|---|---|---|
| 29 | 1100 | 99.22% | 0.0377 |
| 30 | 1150 | 100.00% | 0.0231 |
| 30 | 1170 | 100.00% | 0.0269 |

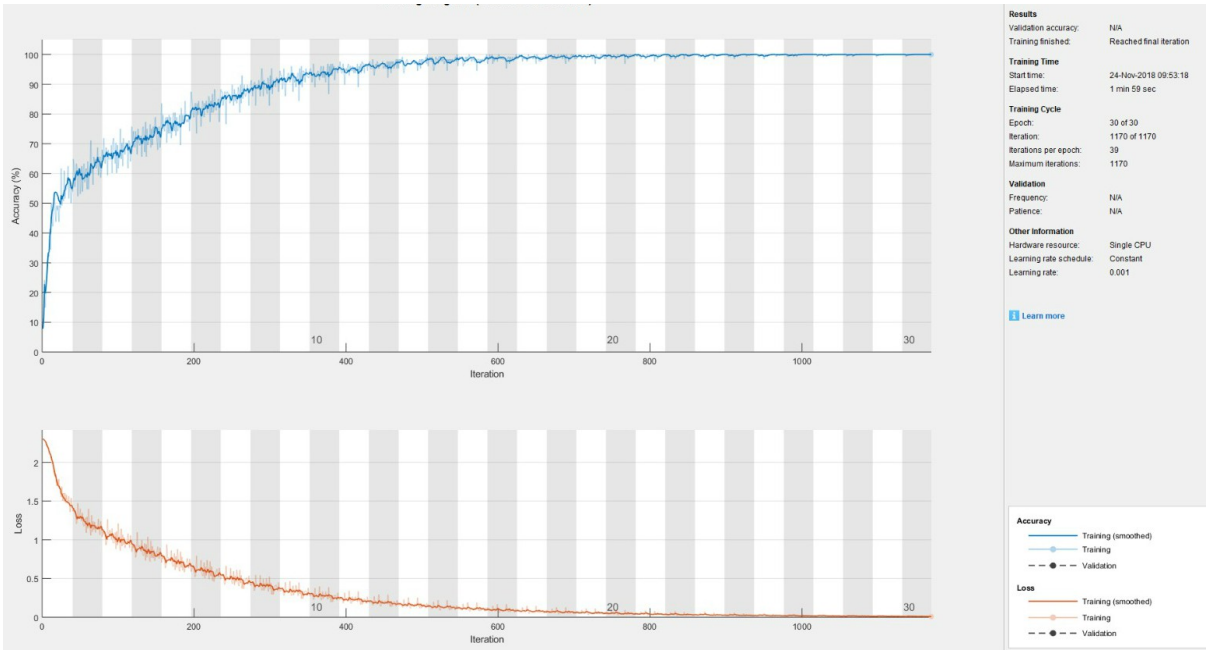
SGDM
Acc SGDM: 0.9770




| | | | |
|---|---|---|---|
| 1 | 1 | 7.81% | 2.3026 |
| 2 | 50 | 33.59% | 2.2735 |
| 3 | 100 | 48.44% | 1.6613 |
| 4 | 150 | 64.06% | 1.1803 |
| 6 | 200 | 64.06% | 1.0499 |
| 7 | 250 | 76.56% | 0.8391 |
| 8 | 300 | 77.34% | 0.6981 |
| 9 | 350 | 77.34% | 0.7084 |
| 11 | 400 | 87.50% | 0.4902 |
| 12 | 450 | 91.41% | 0.3839 |
| 13 | 500 | 92.19% | 0.2986 |
| 15 | 550 | 93.75% | 0.2583 |
| 16 | 600 | 97.66% | 0.2010 |
| 17 | 650 | 92.97% | 0.2642 |
| 18 | 700 | 97.66% | 0.1448 |
| 20 | 750 | 96.88% | 0.1314 |
| 21 | 800 | 97.66% | 0.1232 |
| 22 | 850 | 98.44% | 0.1009 |
| 24 | 900 | 100.00% | 0.1051 |
| 25 | 950 | 97.66% | 0.1483 |
| 26 | 1000 | 99.22% | 0.0743 |
| 27 | 1050 | 100.00% | 0.0603 |

| | | | |
|---|---|---|---|
| 29 | 1100 | 99.22% | 0.0769 |
| 30 | 1150 | 100.00% | 0.0524 |
| 30 | 1170 | 100.00% | 0.0566 |

Rmsprop
Acc rmsprop :0.9846

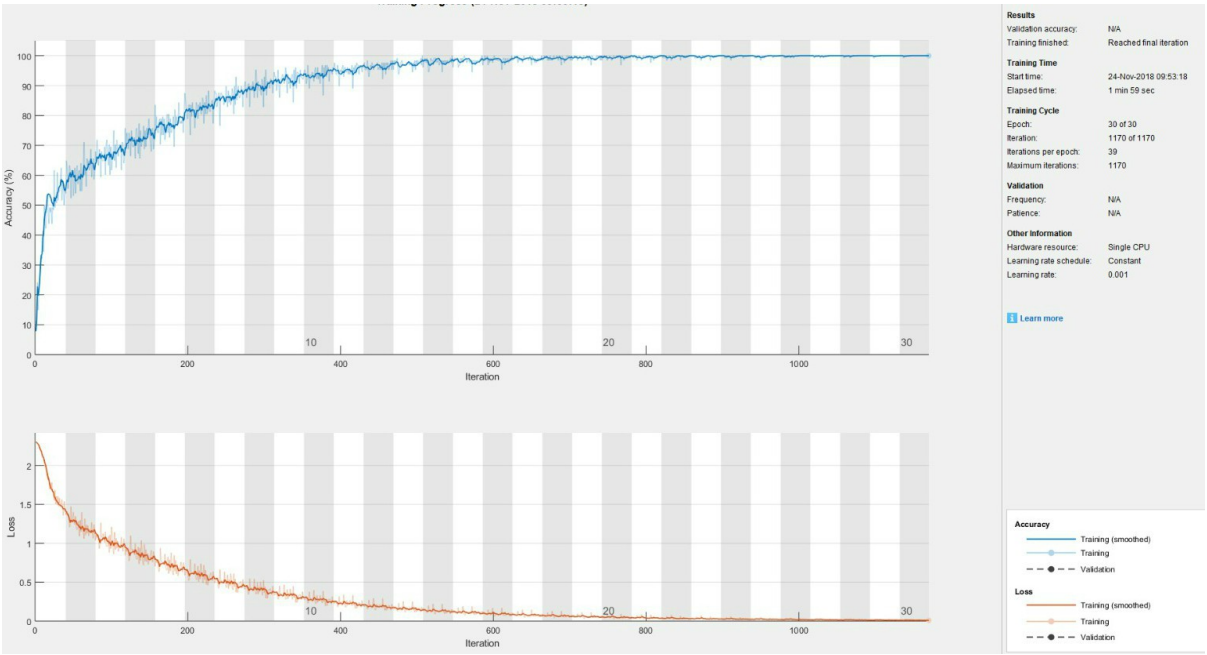



| | | | |
|---|---|---|---|
| 1 | 1 | 7.81% | 2.3026 |
| 2 | 50 | 53.13% | 1.3992 |
| 3 | 100 | 61.72% | 1.2097 |
| 4 | 150 | 74.22% | 0.8094 |
| 6 | 200 | 78.91% | 0.6004 |
| 7 | 250 | 89.84% | 0.4312 |
| 8 | 300 | 92.97% | 0.3646 |
| 9 | 350 | 92.19% | 0.3773 |
| 11 | 400 | 95.31% | 0.2470 |
| 12 | 450 | 96.09% | 0.1747 |
| 13 | 500 | 96.88% | 0.1309 |
| 15 | 550 | 99.22% | 0.1096 |
| 16 | 600 | 99.22% | 0.1007 |
| 17 | 650 | 96.88% | 0.1463 |
| 18 | 700 | 99.22% | 0.0463 |
| 20 | 750 | 99.22% | 0.0512 |
| 21 | 800 | 100.00% | 0.0341 |
| 22 | 850 | 100.00% | 0.0255 |
| 24 | 900 | 100.00% | 0.0248 |
| 25 | 950 | 98.44% | 0.0500 |
| 26 | 1000 | 100.00% | 0.0144 |
| 27 | 1050 | 100.00% | 0.0107 |

| | | | |
|---|---|---|---|
| 29 | 1100 | 100.00% | 0.0233 |
| 30 | 1150 | 100.00% | 0.0107 |
| 30 | 1170 | 100.00% | 0.0085 |

Acc with 3 layer on Test set :97.80%

Train: 4000. Test:1000 Images

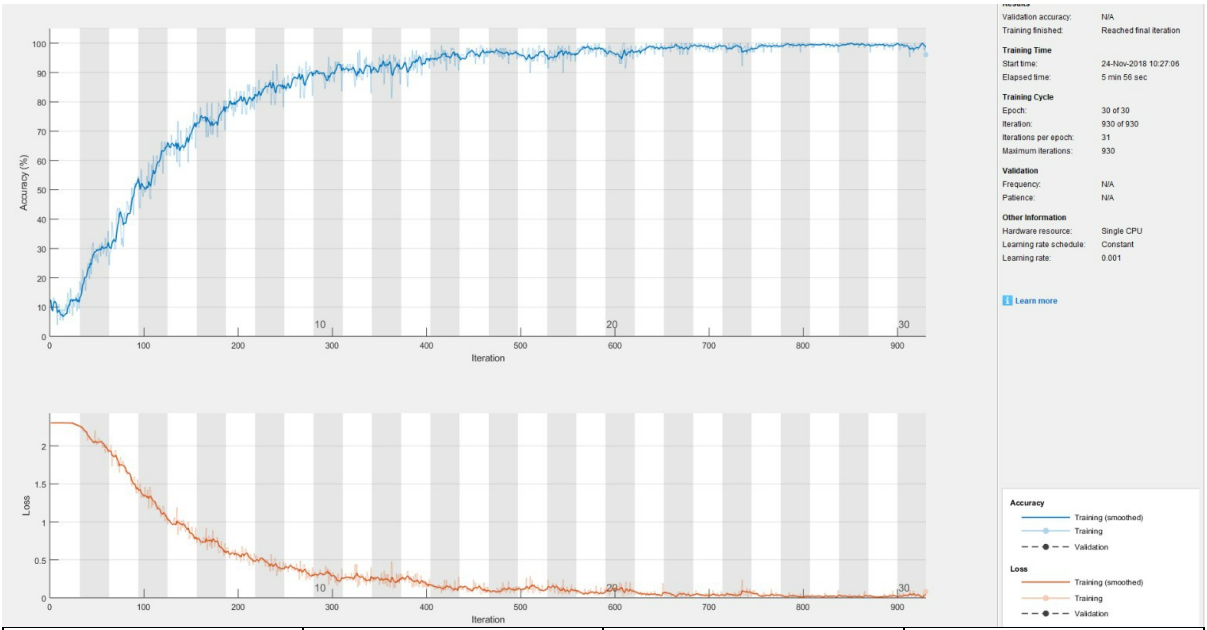



**ACC with 4 Layr on Test set :98.80**

Train: 4000, Test: 1000 imges

| | | | |
|---|---|---|---|
| 1 | 1 | 50.78% | 2.3026 |
| 2 | 50 | 50.78% | 1.3865 |
| 4 | 100 | 71.88% | 0.6865 |
| 5 | 150 | 89.84% | 0.2491 |
| 7 | 200 | 92.97% | 0.1623 |
| 9 | 250 | 94.53% | 0.1064 |
| 10 | 300 | 99.22% | 0.0389 |
| 12 | 350 | 100.00% | 0.0148 |
| 13 | 400 | 100.00% | 0.0180 |
| 15 | 450 | 100.00% | 0.0120 |
| 17 | 500 | 100.00% | 0.0079 |
| 18 | 550 | 99.22% | 0.0223 |
| 20 | 600 | 99.22% | 0.0179 |
| 21 | 650 | 100.00% | 0.0021 |
| 23 | 700 | 100.00% | 0.0050 |
| 25 | 750 | 100.00% | 0.0009 |
| 26 | 800 | 100.00% | 0.0003 |
| 28 | 850 | 100.00% | 0.0002 |
| 30 | 900 | 100.00% | 0.0002 |
| 30 | 930 | 100.00% | 0.0002 |

ACC with 5 Layer on Test set : 98.30%




| | | | |
|---|---|---|---|
| 1 | 1 | 12.50% | 2.3026 |
| 2 | 50 | 28.13% | 2.0149 |
| 4 | 100 | 50.78% | 1.3160 |
| 5 | 150 | 70.31% | 0.8124 |
| 7 | 200 | 80.47% \| | 0.6009 |
| 9 | 250 | 84.38% | 0.4570 |
| 10 | 300 | 89.84% | 0.3341 |
| 12 | 350 | 95.31% | 0.1685 |
| 13 | 400 | 96.88% | 0.1229 |
| 15 | 450 | 95.31% | 0.2402 |
| 17 | 500 | 95.31% | 95.31% |
| 18 | 550 | 95.31% | 0.1008 |
| 20 | 600 | 96.09% | 0.1002 |
| 21 | 650 | 98.44% | 0.0474 |
| 23 | 700 | 96.88% | 0.0658 |
| 25 | 750 | 100.00% | 0.0213 |
| 26 | 800 | 100.00% | 0.0051 |
| 28 | 850 | 100.00% | 0.0062 |
| 30 | 900 | 100.00% | 0.0074 |
| 30 | 930 | 96.09% | 0.0815 |

As can be seen we conducted experiments by different optimization techniques and different layers, in our experiment we obtained the best accuracy with ADAM optimization technique. By adding new layer I expected to have better results, although the performance improved when we added fourth layer for the fifth it did not have positive impact so we can conslide that it will not be effective always.

I achieved the best performances in the CNNs, although accuracies were so close to each other, the best one in our task was with ADAM algorithms. The choice of number of layers and optimization algorithm is completely dependent on our data and our dataset.

## 6. Conclusion

In this project we performed some experiments in representation and classification of our data from both traditional (PCA and LDA) and novel approaches (CNNs).

There types of feature extraction were discussed, PCA (Principal Component Analysis) and LDA (Linear Discriminative Analysis) and CNN. If PCA offers better results in image reconstruction from a feature vector, LDA is more efficient at extracting discriminant characteristics and is more suitable for classification tasks. However, the two methods can also be used in combination: PCA is applied first to reduce the size of the dataset, and LDA is then responsible for finding a space in which class separation is optimal.

Once the characteristics of our data were extracted, two classifiers were tested: one of the category of generative classifiers, K-NN, and the other of the class of discriminant classifiers, SVM. While K-NN is simpler to implement, SVM offers more advanced classification options and better accuracy through the use of kernels to project data into a higher-dimensional space that is more conducive to separating data. However, this also represents one of its weaknesses, since the initialization and determination of optimal kernel parameters is a challenging task for obtaining high performance.

In the light of the analysis of the various techniques explored, it is not surprising to note that the use of dimension reduction techniques such as PCA and LDA allows better grouping of data and better classification results. The best results for the classification of faces in our dataset are obtained by the combination of LDA data extraction and a K-NN model classification. This can be explained by the difficulties encountered in finding the optimal parameters of the SVM models because of the overfitting situation following the application of LDA.

The most important thing that I learnt from the CNN part of project was that the choice of parameter updater rule such as Adam or RMSProp , SGD) is generally dependent on the model and the dataset.