In The name of God

University Tehran

Engineering Facility

Electrical and Computer Engineering

# Pattern Recognition

## HW # 4

**Amin Fadaeinedjad**

**810195442**

Spring 98

## List

## Abstract

This Homework is about using different methods to learn a dataset and predict the test dataset in and we also compared it with the ready code in the scikit-learn library and also we are going see that what happened when we reduce the dimension of the data that we have and how the accuracy will threat them by the number of features that we have.

## Question 1

$$p(y|\theta_i) = \frac{1}{\sqrt{2\pi}s}\exp\left(-\frac{(y-\mu_i)^2}{2s^2}\right)$$

$$D_i = [y_1, y_2, \dots, y_n]^T \quad \theta_i = \begin{pmatrix}\mu_i \\ s\end{pmatrix}$$

$$r = \frac{\ln\big(p(D|\theta_1)\big)}{\ln\big(p(D|\theta_2)\big)} = \frac{\ln\left(\prod p(y_j|\theta_1)\right)}{\ln\left(\prod p(y_j|\theta_2)\right)} = \frac{\left(-n\ln\left(\sqrt{2\pi}s\right) - \sum \frac{(y_j-\mu_1)^2}{2s^2}\right)}{\left(-n\ln\left(\sqrt{2\pi}s\right) - \sum \frac{(y_j-\mu_2)^2}{2s^2}\right)}$$

$$\frac{\left(-n\,ln\left(\sqrt{2\pi}s\right) - \sum \frac{(y_j-\mu_1+\mu_2-\mu_2)^2}{2s^2}\right)}{\left(-n\,ln\left(\sqrt{2\pi}s\right) - \sum \frac{(y_j-\mu_2+\mu_1-\mu_1)^2}{2s^2}\right)}$$

$$\frac{-n\ln\sqrt{2\pi}s - \sum_{y_j\in\omega_1}\frac{(y_j-\mu_1)^2}{2s^2} - \sum_{y_j\in\omega_2}\frac{(y_j-\mu_1)^2}{2s^2}}{-n\ln\sqrt{2\pi}s - \sum_{y_j\in\omega_1}\frac{(y_j-\mu_1)^2}{2s^2} - \sum_{y_j\in\omega_2}\frac{(y_j-\mu_1)^2}{2s^2}} =$$

$$\sum_{y_j\in\omega_1}(y_j-\mu_1)^2 = \sum_{y_j\in\omega_2}(y_j-\mu_2)^2 = s^2 \qquad \sum_{y_j\in\omega_1} y_j = \mu_1 \quad \sum_{y_j\in\omega_2} y_j = \mu_2$$

$$\frac{\left(-n\,ln\sqrt{2\pi}s - \frac{1}{2} - \sum_{y_j\in\omega_2}\frac{(y_j-\mu_1)^2}{2s^2}\right)}{\left(-n\,ln\sqrt{2\pi}s - \frac{1}{2} - \sum_{y_j\in\omega_1}\frac{(y_j-\mu_2)^2}{2s^2}\right)} \to C = -n\ln\sqrt{2\pi}s \to$$

$$\frac{C - \frac{1}{2} - \sum_{y_j\in\omega_2}\frac{(y_j-\mu_1+\mu_2-\mu_2)^2}{2s^2}}{C - \frac{1}{2} - \sum_{y_j\in\omega_1}\frac{(y_j-\mu_2+\mu_1-\mu_1)^2}{2s^2}} = \frac{C - \frac{1}{2} - \sum_{y_j\in\omega_2}\frac{(y_j-\mu_2)^2}{2s^2} + \frac{(y_j-\mu_2)(\mu_2-\mu_1)}{s^2} + \frac{(\mu_2-\mu_1)^2}{2s^2}}{C - \frac{1}{2} - \sum_{y_j\in\omega_1}\frac{(y_j-\mu_1)^2}{2s^2} + \frac{(y_j-\mu_1)(\mu_1-\mu_2)}{s^2} + \frac{(\mu_1-\mu_2)^2}{2s^2}} =$$

$$\sum_{y_j\in\omega_1}(y_j-\mu_1)^2 = \sum_{y_j\in\omega_2}(y_j-\mu_2)^2 = s^2 \qquad \sum_{y_j\in\omega_1}(y_j-\mu_1) = \sum_{y_j\in\omega_2}(y_j-\mu_2) = 0 \to$$

$$\frac{C - 1 - \sum_{y_j\in\omega_2}\frac{(\mu_2-\mu_1)^2}{2s^2}}{C - 1 - \sum_{y_j\omega_1}\frac{(\mu_2-\mu_1)^2}{2s^2}} \to number\big(y_j \in \omega_1\big) = \eta_1 \;\; number\big(y_j \in \omega_2\big) = \eta_2$$

$$r(\theta_1,\theta_2) = \frac{C - 1 - \frac{\eta_2(\mu_2-\mu_1)^2}{2s^2}}{C - 1 - \frac{\eta_1(\mu_2-\mu_1)^2}{2s^2}} \to r\left(C - 1 - \frac{\eta_1(\mu_2-\mu_1)^2}{2s^2}\right) = \left(C - 1 - \frac{\eta_2(\mu_2-\mu_1)^2}{2s^2}\right)$$

$$(r-1)(C-1) = (r\eta_1 - \eta_2)\frac{(\mu_2-\mu_1)^2}{2s^2} \to \frac{(r-1)(C-1)}{r\eta_1-\eta_2} = \frac{(\mu_2-\mu_1)^2}{2s^2} \to$$

$$\frac{(r(\theta_1,\theta_2)-1)\left(-n\ln\sqrt{2\pi}s - 1\right)}{r\eta_1-\eta_2} = \frac{(\mu_2-\mu_1)^2}{2s^2}$$

$$\frac{(\mu_2-\mu_1)^2}{2s^2}=\frac{(r(\theta_1,\theta_2)-1)\left(1+n\ln\sqrt{2\pi}s\right)}{\eta_2-r\eta_1}$$

We can see that the fisher function is $\frac{(\mu_2-\mu_1)^2}{2s^2}$ and the $r(\theta_1,\theta_2)$ is the log likelihood retio

## Question 2

$$E(y|\omega_1)=\mu_1\;E(y|\omega_2)=\mu_2\;\;E((y-\mu_1)^2|\omega_1)=\sigma_1^2\;E((y-\mu_2)^2|\omega_2)=\sigma_2^2$$

$$J_1(w)=\left(\frac{(\mu_1-\mu_2)^2}{\sigma_1^2+\sigma_2^2}\right)\to y=w^Tx\to E(y|\omega_i)=E(w^Tx|\omega_i)=\mu_i=w^T\boldsymbol{\mu_i}$$

$$E((y-\mu_i)^2|\omega_i)=E((w^Tx-\mu_i)^T(w^Tx-\mu_i)|\omega_i)\to$$

$$E(x^Tww^Tx-\mu_i^Tw^Tx-x^Tw\mu_i+\mu_i^T\mu_i\big|\omega_i)=\sigma_i^2\to E(x^Tx|\omega_i)=\Sigma_i+\boldsymbol{\mu_i^2}$$

$$\frac{d\big(J_1(w)\big)}{dw}=\frac{d\big(J_1(w)\big)}{d\mu_1}\frac{d\mu_1}{dw}+\frac{d\big(J_1(w)\big)}{d\mu_2}\frac{d\mu_2}{dw}+\frac{d\big(J_1(w)\big)}{d\sigma_1^2}\frac{d\sigma_1^2}{dw}+\frac{d\big(J_1(w)\big)}{d\sigma_2^2}\frac{d\sigma_2^2}{dw}$$

$$\frac{dj}{d\mu_1}\boldsymbol{\mu_1}+\frac{dj}{d\mu_2}\boldsymbol{\mu_2}+\frac{dj}{d\sigma_1^2}\;2\Sigma_1w+\frac{dj}{d\sigma_2^2}2\Sigma_2w=\frac{dj}{dw}=0\to$$

$$w^*=-\frac{1}{2}\left(\frac{dj}{d\sigma_1^2}\Sigma_1+\frac{dj}{d\sigma_2^2}\Sigma_2\right)^{-1}\left(\frac{dj}{d\mu_1}\boldsymbol{\mu_1}+\frac{dj}{d\mu_2}\boldsymbol{\mu_2}\right)\to$$

$$\frac{dj}{d\sigma_i^2}=-\frac{(\mu_1-\mu_2)^2}{(\sigma_1^2+\sigma_2^2)^2}\quad\frac{dj}{\mu_1}=\frac{2(\mu_1-\mu_2)}{\sigma_1^2+\sigma_2^2}\quad\frac{dj}{d\mu_2}=-\frac{2(\mu_1-\mu_2)}{\sigma_1^2+\sigma_2^2}\to$$

$$w^*=\frac{1}{2}\left(\frac{(\mu_1-\mu_2)^2}{(\sigma_1^2+\sigma_2^2)^2}(\Sigma_1+\Sigma_2)\right)^{-1}\left(\frac{2(\mu_1-\mu_2)}{\sigma_1^2+\sigma_2^2}\boldsymbol{\mu_1}-\frac{2(\mu_1-\mu_2)}{\sigma_2^2+\sigma_2^1}\boldsymbol{\mu_2}\right)\to$$

$$w^*=\frac{(\sigma_1^2+\sigma_2^2)^2}{(\mu_1-\mu_2)^2}(\Sigma_1+\Sigma_2)^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})\frac{\mu_1-\mu_2}{\sigma_1^2+\sigma_2^2}\to$$

$$w^*=\frac{\sigma_2^2+\sigma_1^2}{\mu_1-\mu_2}(\mathit{\Sigma}_1+\mathit{\Sigma}_2)^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})\to\frac{\sigma_1^2+\sigma_2^2}{\mu_1-\mu_2}\;is\;constant$$

That means that the constant isn't important at all $w$ is a vector that only the direction is it that counts the constant that is multiplied in the vector isn't important because

$J(w)=J(kw)$ this shows that the scaler constant doesn't count.

$$w=(\Sigma_1+\Sigma_2)^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})$$

## Question 3

We need to only use the formulas to see the result

$$J = \frac{1}{n_1 n_2}\sum_{y_i \in Y_1}\sum_{y_j \in Y_2}(y_1 - y_2)^2$$

$$\frac{1}{n_1 n_2}\sum_{y_i \in Y_1}\sum_{y_j \in Y_2}(y_1^2 + y_2^2 - 2y_1 y_2) =$$

$$\frac{1}{n_1 n_2}\sum_{y_i \in Y_1}\sum_{y_j \in Y_2}(y_1^2) + \frac{1}{n_1 n_2}\sum_{y_i \in Y_1}\sum_{y_j \in Y_2}(y_2^2) - \frac{2}{n_1 n_2}\sum_{y_i \in Y_1}\sum_{y_j \in Y_2}(y_1 y_2) =$$

$$\frac{1}{n_1}\sum_{y_1 \in Y_1} y_1^2 + \frac{1}{n_2}\sum_{y_2 \in Y_2} y_2^2 - 2\frac{1}{n_1}\sum_{y_1 \in Y_1} y_1 \frac{1}{n_2}\sum_{y_2 \in Y_2} y_2 \rightarrow m_i = \frac{1}{n_i}\sum_{y_i \in Y_i} y_i$$

And we must know the definition of the scatter function

$$s = \sum_{x^q \in \omega_i}(x^q - \mu_i)^T(x^q - \mu) \rightarrow s_i = \sum_{y_i \in Y_i}(y_i - m_i)^2 \rightarrow$$

$$s_i = \sum_{y_i \in Y_i} y_i^2 - 2m_i \sum_{y_i \in Y_i} y_i + \sum_{y_i \in Y_i} m_i^2 \rightarrow number(y_i \in Y_i) = n_i$$

$$\frac{s_i}{n_i} = \frac{1}{n_i}\sum_{y_i \in Y_i} y_i^2 - 2m_i \frac{1}{n_i}\sum_{y_i \in Y_i} y_i + \frac{n_i}{n_i} m_i^2 = \frac{1}{n_i}\sum_{y_i \in Y_i} y_i^2 - m_i^2$$

$$\frac{s_1^2}{n_1} = \frac{1}{n_1}\sum_{y_1 \in Y_1} y_1^2 - m_1^2 \quad \frac{s_2^2}{n_2} = \frac{1}{n_2}\sum_{y_2 \in Y_2} y_2^2 - m_2^2$$

$$J = \left(\frac{s_1^2}{n_1} + m_1^2\right) + \left(\frac{s_2^2}{n_2} + m_2^2\right) - 2(m_1)(m_2) \rightarrow$$

$$J = (m_1^2 + m_2^2 - 2m_1 m_2) + \frac{1}{n_1}s_1^2 + \frac{1}{n_2}s_2^2$$

The within group can be written as the following equation

## Question 4

In this question are going use the feature selection to implement out classifier and we have 784 features and that is a large number we want to decrease the dimensions in order to make in better to generalizing the data and for better and faster classification.

What we have done is finding the best feature with the maximum accuracy and we started at 1 feature and every time we increased the dimension of features

The best features are:

Feature number: 182, 434, 610, 289, 40, 90, 275, 380, 745, 454, 417, 229, 545, 10, 355, 6, 553, 238, 431, 552, 743, 320, 0, 608, 326, 1, 293, 84, 11, 145, 295, 36, 404, 506, 635, 28, 265, 344, 267, 567, 13, 446, 27, 628, 294, 29, 775,283,319, 511, 531, 336, 174, 92, 241, 125, 756, 56, 140, 274.

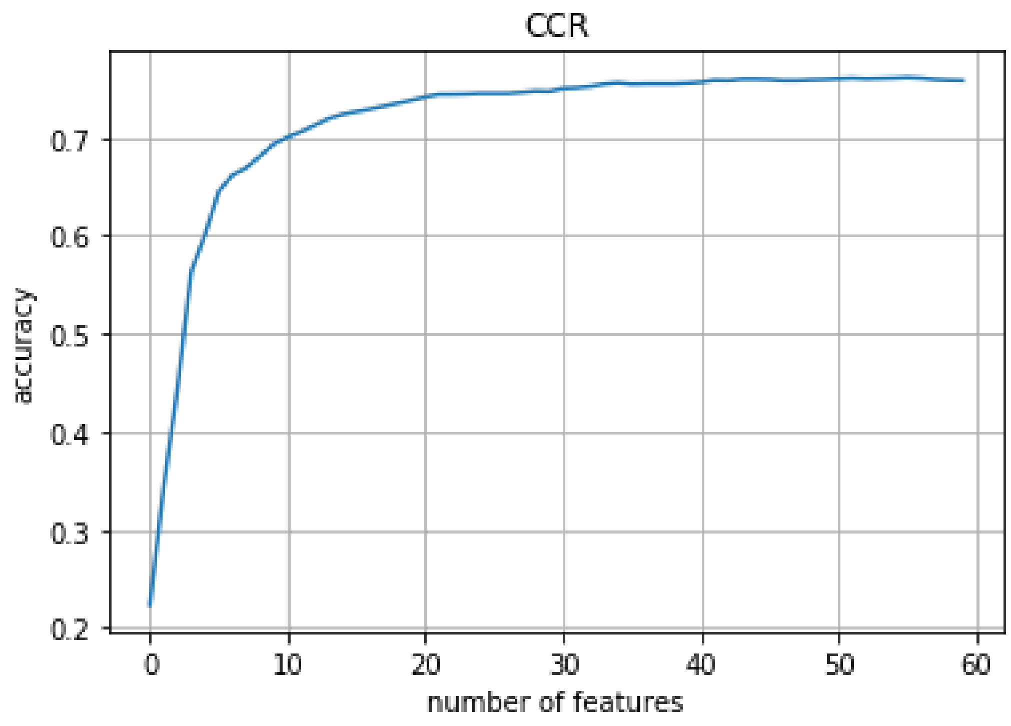

**Figure 1 Classification rate**

Because of the heavy calculation we only have done it until 60 features and the maximum accuracy was 76.1% for 56 features.

## Question 5

At first we need to know what algorithm we are using and how it works.

At first we need to calculate the covariance matrix.

$$\Sigma = \frac{1}{Q-1}\sum_{q}^{Q}(x^q-\mu)(x^q-\mu)^T \qquad \mu = \frac{1}{Q}\sum_{q=1}^{Q} x^q$$

$$\Sigma v_i = \lambda_i v_i \to eigen\ value = \{\lambda_i\}\ \ eigen\ vector = \{v_i\}$$

$$\Sigma v_i = \lambda_i v_i \quad \forall i = 1, \dots, n \to [\Sigma v_1, \Sigma v_2, \dots, \Sigma v_n] = [\lambda_1 v_1, \lambda_2 v_2, \dots, \lambda_n v_n]$$

$$\Sigma V = DV \to V = [v_1, v_2, \dots, v_n] \quad \& \, D = Diag([\lambda_1, \lambda_2, \dots, \lambda_n] \to sort$$

$$[\lambda_1, \lambda_2, \dots, \lambda_n] \to \lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n \geq 0 \to D$$

Now we can delete the $\lambda_i$ that are small $V_{new} = [v_1, v_2, \dots v_m] \quad D = Diag([\lambda_1, \dots \lambda_m])$

$$\tilde{x} = V^T x \to V\ the\ eigen\ vector\ \ x\ are\ the\ old\ data\ and\ \tilde{x}\ new\ data$$

Now we can use other classification do classify the new data with this algorithm we will have a better accuracy to classify this data

a) The Eigen value of this covariance matrix is the following figure of the **Logarithm** of the Eigen value of the covariance is Figure 2

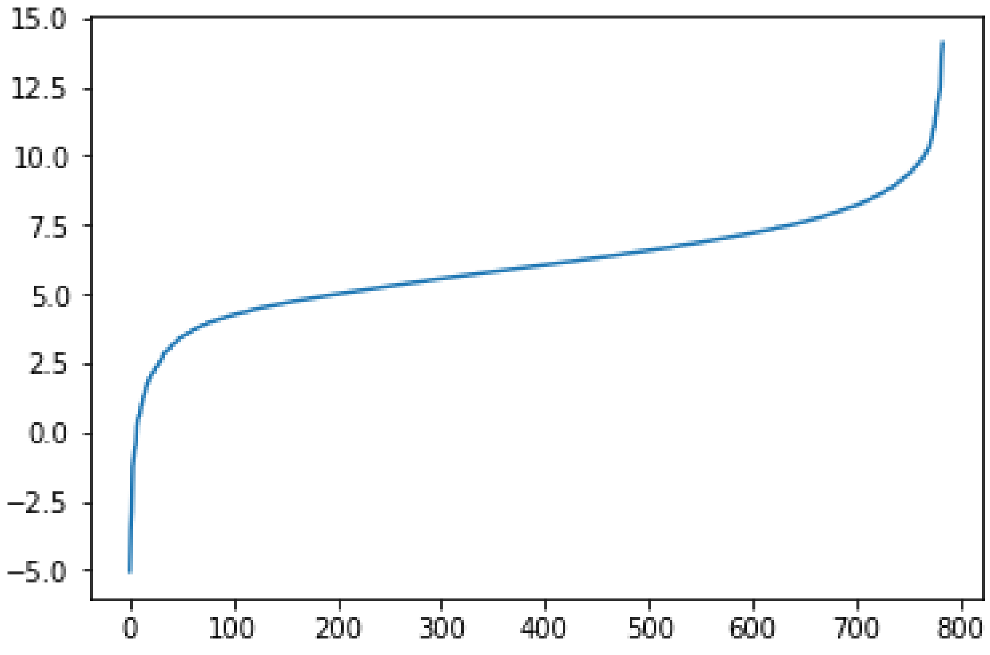

**Figure 2**

The plot of the Eigen value in decimal number is Figure 3

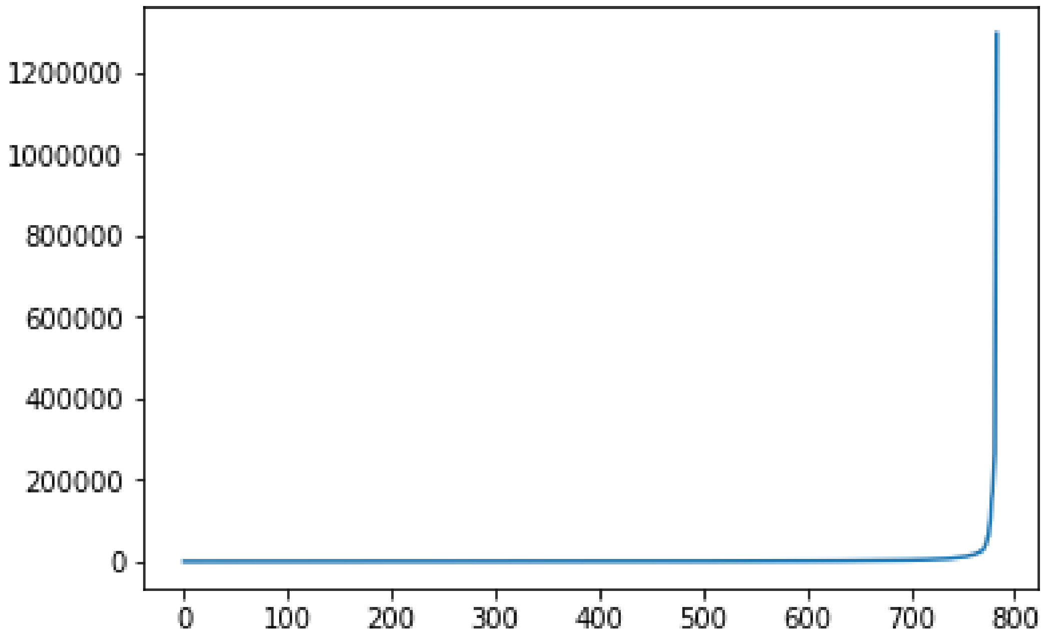

**Figure 3**

Which we have shown it in an increasing order.

The reason that we use the Eigen values to show the goodness of the classifier is that the trace of that matrix is a parameter that shows the goodness of it.

b) We have done the method from PCA and after that we plotted a diagram that shows the number of features and the accuracy of the classifier will be like Figure 4

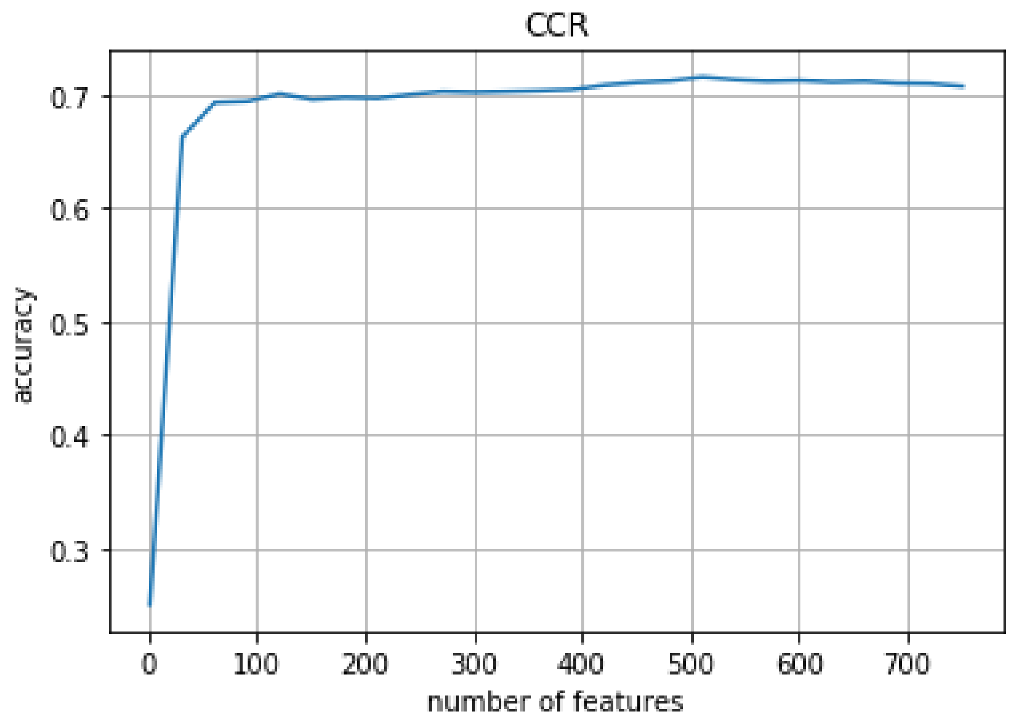

**Figure 4 CCR**

The point is that we haven’t done it with all the features we have used only a certain number of features that will add each other in the number of 30 and it starts from 1 to 31 to 61 to …

And Figure 4 was the result we can see that the classifier with near 50 features will work as accurate as the same classifier with 500 features and this shows that we can use only 50 and get the same result from a 780 feature classifier.

c) We have done this classification without PCA and the accuracy was 56.66% which is quite lower than when we use PCA so what we can get as a result that if we use PCA we can reduce the dimension of the data points but also increase the accuracy of the classification.

| Type | Accuracy | Dimension |
|---|---|---|
| Without PCA | 56% | High |
| PCA | 74% | Low |

So it is better to use PCA because it will give us a better accuracy and less need f memory to save all the data point of the test and the train data.

## Question 6

This method is like PCA but has some different cases at first you need to calculate the scatter matrix of the data set that we have.

$$S_i = \sum_{x^q \in \omega_i} (x^q - \mu)(x^q - \mu)^T \qquad S_w = \sum_{i=1}^{C} S_i$$

$$S_B = \sum_{i=1}^{C} (\mu_i - \mu_{total})(\mu_i - \mu_{total})^T$$

$$S_p \triangleq S_w^{-1} S_B \quad Separability\ matrix$$

$$S_p \rightarrow Eigen\ vector\ [v_1, v_2, \dots, v_n]\ Eigen\ value\ [\lambda_1, \dots, \lambda_n] \rightarrow sort$$

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$$

And just like the PCA we can get rid of some of the small Eigen value and Eigen vector and the way that we do that we sort it at first and delete the small Eigen values and their Eigen vectors as well, and after that we are going to have a new data points

$$J = trace(S_w^{-1} S_B) = \lambda_1 + \lambda_2 + \cdots + \lambda_n$$

$$\tilde{x} = V^T (x - \mu)$$

So in this case we are going to have a new data set and we are going use the other classifier to classify this new data.

The thing that LDA does is that it separates the data points and that will help us to classify the data and this could be done with dimension reduction and this means we can change the features to things that will better generalize the data points.

a) At this part we are going to plot the Eigen values of the matrix

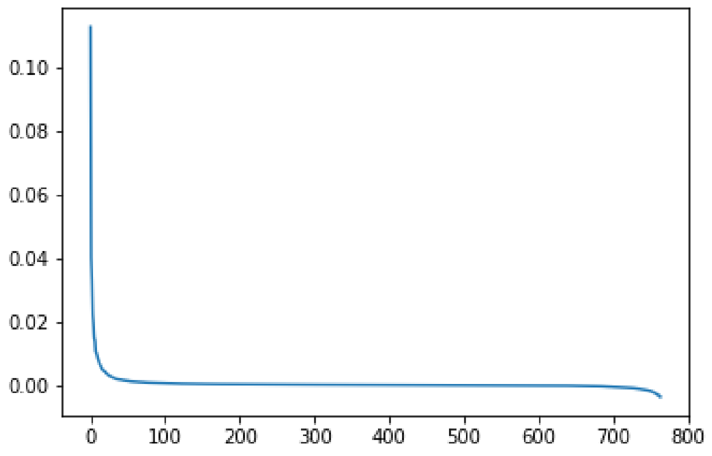

**Figure 5**

Figure 5 is the diagram of the Eigen values that are sorted in the decreasing way after that we are going to get rid of the small Eigen values.

b) In this we are going to plot the trace of the separable matrix and the result is like this

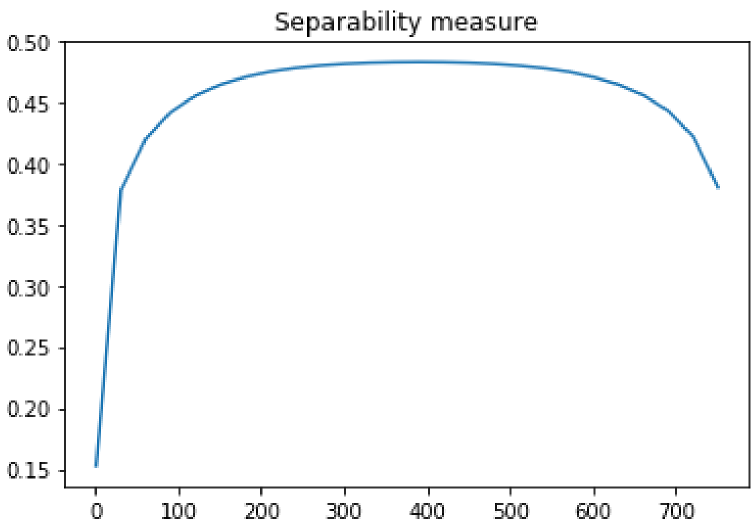

**Figure 6**

We can see from figure 6 that if we start to delete features at first o will increase but after a while the trace will start to decrease and that isn't what we want so it's better to choose a number of features that isn't too much and isn't too low.

c) We are going to see the different results for the dimension reduction in LDA

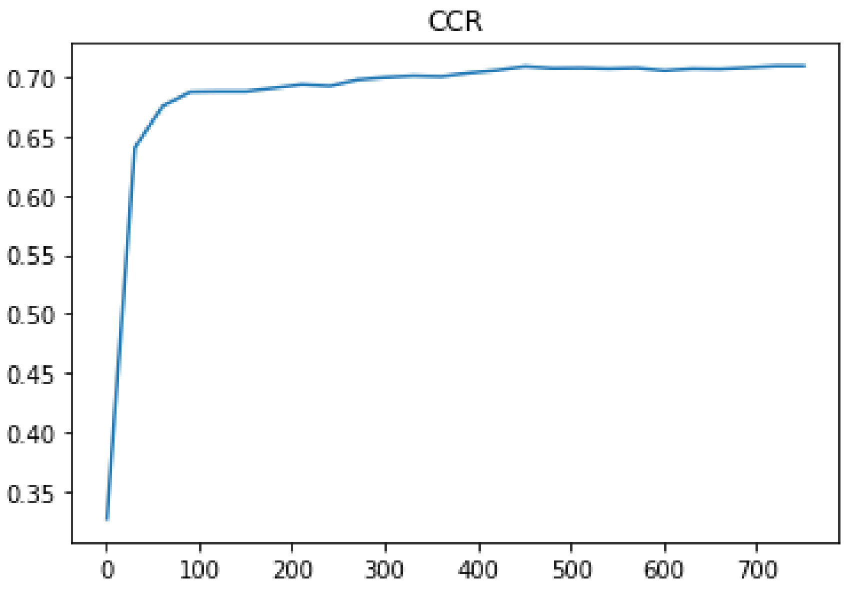

**Figure 7**

Figure 7 shows the accuracy of the classifier with number of features and shows that the accuracy will reach 70% near 100 features this means that if we have this amount of data there is no need to increase it because if we increase the number of features that we have we only will increase the memory that we need and that isn't good at all.

So we rather have about 100 features because it has a good accuracy and the number of features are reasonable and that is very important because in large sample data this will make some problems like memory link or making ram overflow and this method of LDA is a useful method

d) In the previous part we saw the different accuracy has accorded in the number of features that we use in this case for LDA and if we don't use LDA at all the accuracy will be about 56% and we know from the previous part that the accuracy of the LDA in the right number of features is about 70% and we also have dimension reduction in LDA.

| Type | Accuracy | Dimension |
|---|---|---|
| Without LDA | 56% | High |
| LDA | 70% | Low |

We can compare the result and see that in this case we have a better accuracy and a lower dimension data point which means faster calculation and better generalization.

## Question 7

In this part we are going to use predefined PCA and LDA functions to change the data with these two algorithms and we the following image is the result of this method

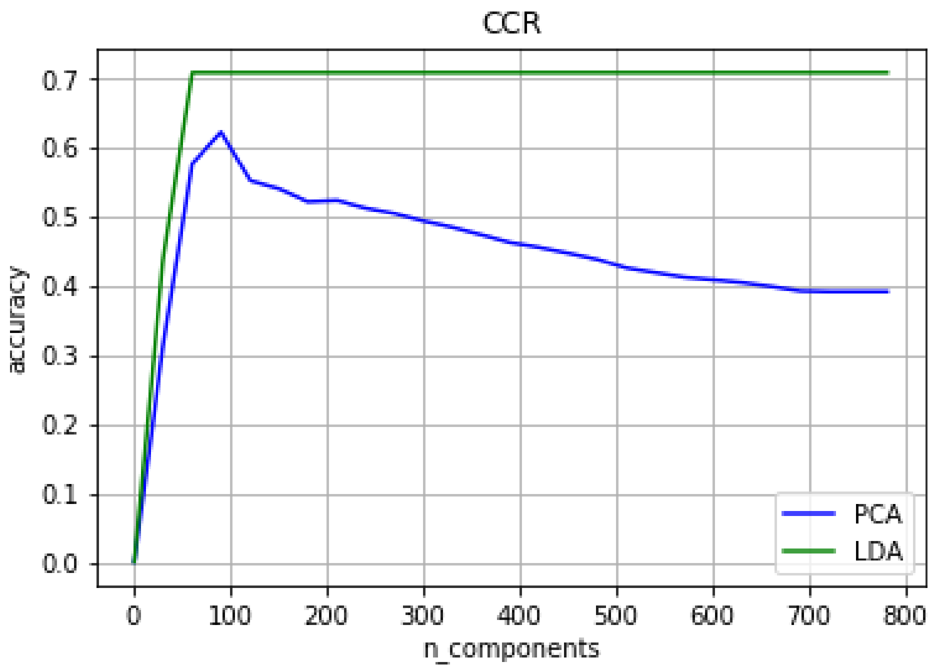

Figure 8

According to figure 8 we can see that LDA has a better performance than the LDA method

Both LDA and PCA are linear transformation techniques: LDA is a supervised whereas PCA is unsupervised – PCA ignores class labels.

We can picture PCA as a technique that finds the directions of maximal variance:

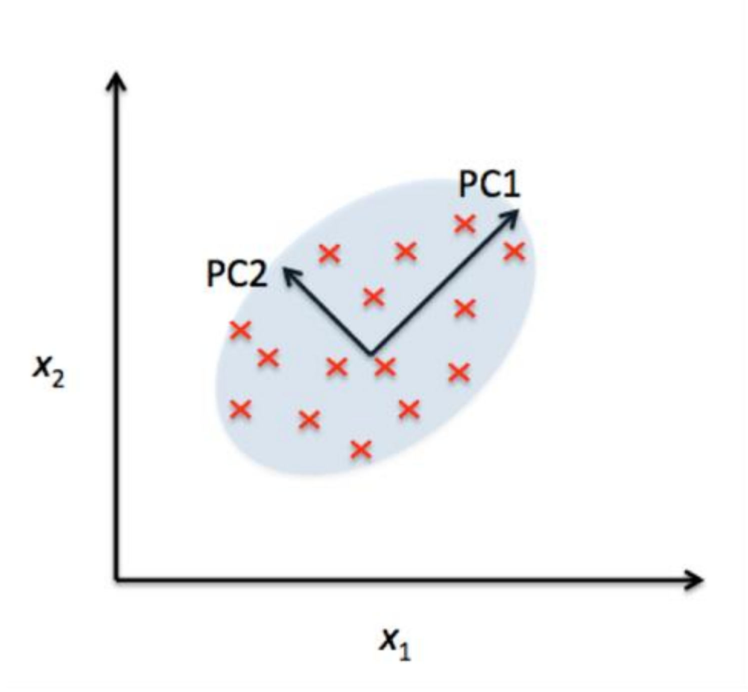

Figure 9

In contrast to PCA, LDA attempts to find a feature subspace that maximizes class severability (note that LD 2 would be a very bad linear discriminant in the figure above).

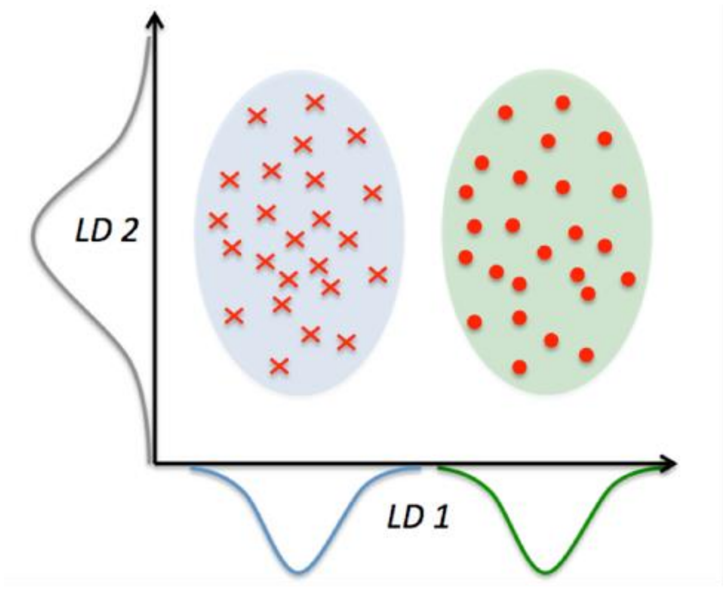

Figure 10

And according to the information above we can understand the data set that we have and we know that the following data has a better performance for LDA and so we can understand that the data set was better generalized with LDA method than the PCA method.

We can also find out that the PCA started to decrease the accuracy this means that we knew feature that we choose may have bad conditioning with each other but for LDA this data set is well generalized with this data set.

## Question 8

In this part we are going to us PCA to reduce the dimension of the image and there are 28 image in the Face folder and 14 images in the Test folder each face has two image in train and one in test the thing we want to do is to use PCA to help us in this condition.

More often than not, features are correlated. As an example, consider the case where we want to use the red, green and blue components of each pixel in an image to classify the image (e.g. detect dogs versus cats). Image sensors that are most sensitive to red light also capture some blue and green light. Similarly, sensors that are most sensitive to blue and green light also exhibit a certain degree of sensitivity to red light. As a result, the R, G, B components of a pixel are statistically correlated. Therefore, simply eliminating the R component from the feature vector, also implicitly removes information about the G and B channels. In other words, before eliminating features, we would like to transform the complete feature space such that the underlying uncorrelated components are obtained.

A) In the first part we want to plot the Recognition rate of this PCA to find the accuracy of the classification we have done was to find the accuracy for the number of features that we have transformed by the method of the PCA and the result was like figure 10

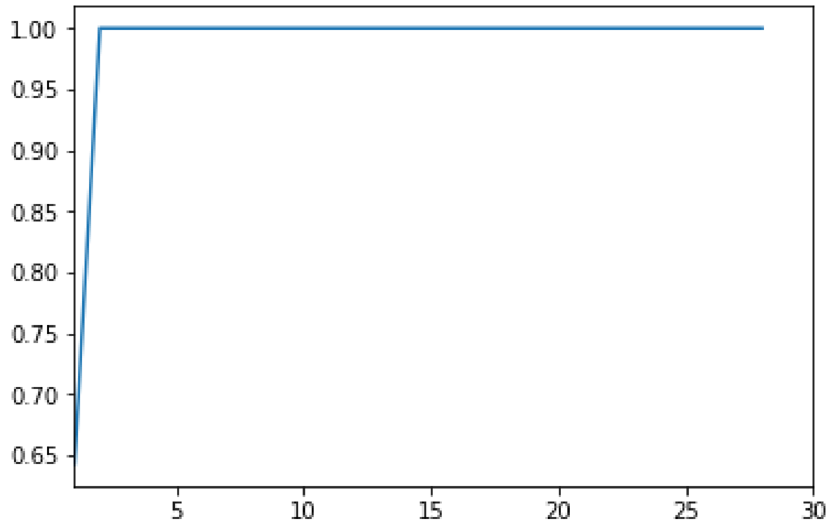

**Figure 11**

The following figure shows that the number of features that go above number 2 we will have an accurate recognition and that is quite strange because with only 2 features we have a classifier that has a 100% rating.

Know we should explain the result is so accurate is that the number of labels were about 14 and the images that we had didn't have a lot in common so if we use PCA to combine the feature with each other and by using test data that aren't a lot the same we can easily have a high rate of recognition but this happened because the test data weren't that scatter but if they were than we were going to have some problems with it.

b) We used a function get_error in our code to calculate the error from the classification

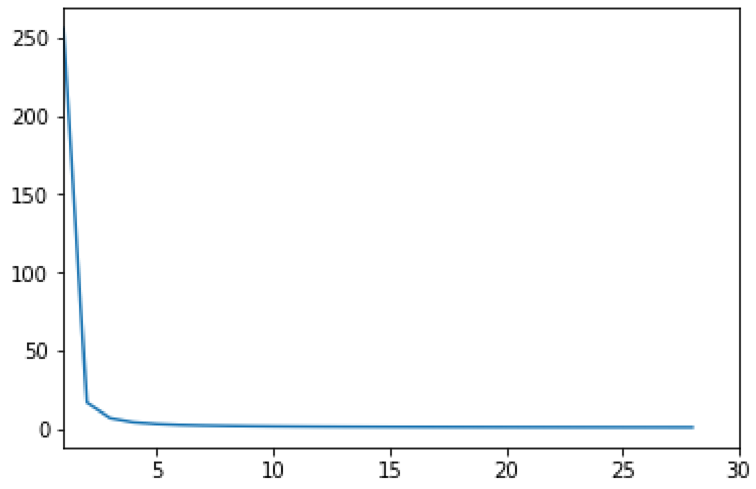

**Figure 12**

The following figure 12 we can see that the error of the classification is decreasing by the number of features that we have at that shows that our classification will have an almost no error when we have the number of 3 features and above and that explain the result just like the previous part that if the number of features pass a certain number than there is going to be no miss classification.

We can use also a pca_inverse function to reconstruct the image but we have used a function call get_error(img, test_data) that does the same thing and we can do the reconstruction with the data that we used PCA on.

## Process

In this Homework we are using different methods to classify our data set and we also used confusion and confidence matrix to give us a better view of the classification. We also compared the result when we change the parameters of the classification and when we changed the type of the classifier that we have

## Reference

Automatic speech recognition a deep learning approach

stackoverflow.com

youtube.com

https://sebastianraschka.com/faq/docs/lda-vs-pca.html

https://scholar.google.com/scholar?q=Forward+Algorithm+using+hidden+markov+model&hl=fa&as_sdt=0&as_vis=1&oi=scholart

https://eu.udacity.com/course/introduction-to-computer-vision--ud810

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.RadiusNeighborsClassifier.html#sklearn.neighbors.RadiusNeighborsClassifier

https://sebastianraschka.com/Articles/2014_kernel_density_est.html