Assignment 3

From *Estimating Densities* and *Reducing Dimensionalities* to *Classifying Samples*

Homeworks Guidelines and Policies

- ***What you must hand in.*** It is expected that the students submit an assignment report (HW3_[student_id].pdf) as well as required source codes (.m or .py) into an archive file (HW3_[student_id].zip).
- ***Pay attention to problem types.*** Some problems are required to be solved *by hand* (shown by the icon), and some need to be implemented (shown by the icon).
  Please don't use implementation tools when it is asked to solve the problem by hand, otherwise you'll be penalized and lose some points.
- ***Don't bother typing!*** You are free to solve by-hand problems on a paper and include picture of them in your report. Here, cleanness and readability are of high importance. Images should also have appropriate quality.
- ***Reports are critical.*** Your work will be evaluated mostly by the quality of your report. Don't forget to explain what you have done, and provide enough discussions when it's needed.
- ***Appearance matters!*** In each homework, 5 points (out of a possible 100) belongs to compactness, expressiveness and neatness of your report and codes.
- ***Python is also allowable.*** By default, we assume you implement your codes in MATLAB. If you're using Python, you have to use equivalent functions when it is asked to use specific MATLAB functions.
- ***Be neat and tidy!*** Your codes must be separated for each question, and for each part. For example, you have to create a separate .m file for part b. of question 3. Please name it like p3b.m.
- ***Use bonus points to improve your score.*** Problems with bonus points are marked by the icon. These problems usually include uncovered related topics or those that are only mentioned briefly in the class.
- ***Moodle access is essential.*** Make sure you have access to Moodle because that's where all assignments as well as course announcements are posted on. Homework submissions are also done through Moodle.

- ***Assignment Deadline.*** Please submit your work **before the end of January 20th**.
- ***Delay policy.*** During the semester, students are given 7 free late days which they can use them in their own ways. Afterwards there will be a 25% penalty for every late day, and no more than three late days will be accepted.
- ***Collaboration policy.*** We encourage students to work together, share their findings, and utilize all the resources available. However you are not allowed to share codes/answers or use works from the past semesters. Violators will receive a zero for that particular problem.
- ***Any questions?*** If there is any question, please don't hesitate to contact me through **ali.the.special@gmail.com**.

1. Basic Density Estimation with Non-Parametric Methods (12 Pts.)

Keywords: *Density Estimation, Non-Parametric Methods, Histogram, Parzen Windows, Kernel Density Estimation, K-Nearest Neighbors*

Density Estimation is the problem of reconstructing an estimate of the probability density function using a set of observed data points. In other words, we observe $X_1, X_2, \ldots, X_n$ and we would like to recover the underlying probability density function generating this dataset. While a classical approach of density estimation is the **Histogram**, we often seek more complicated methods like **Parzen Windows** or **Kernel Density Estimation**.

Let's practice these methods on some routine density estimation problems. First, consider a set of observed data $D = \{0,1,1,1,1,1,2,2,3,3,3,4,5,5,5\}$.

a. Draw a histogram of the data, with the bins centred at {0,1,2,3,4,5} with the width of 1.
b. Sketch the Parzen window estimate of the unknown density function for $h_n = 1,2,4$.
c. Sketch the 3-nearest neighbor estimate of the density function.
d. Consider the following triangle kernel as the window function

$$K(u) = (1 - |u|)\ \delta(|u| \le 1)$$

where $u = x - x_i / h$. Find the kernel density estimates for $x = \{0,1,2,3,4,5\}$ and bandwidths of 2.

In a slightly different scenario, assume $f(x)$ is an unknown density function and the following samples are drawn from $f(x)$:

$$X = \{3,4,5,5,5,6,6,9,13,14,15,15,15,17,17\}$$

e. Estimate the density $p(x)$ at $y = 4,10,16$, assuming a bandwidth of $h = 4$ and standard kernel function

$$K(u) = \begin{cases} 1 & |u| < 1/2 \\ 0 & \text{otherwise} \end{cases}$$

f. Repeat the previous part using a Gaussian kernel $N(0,1)$.

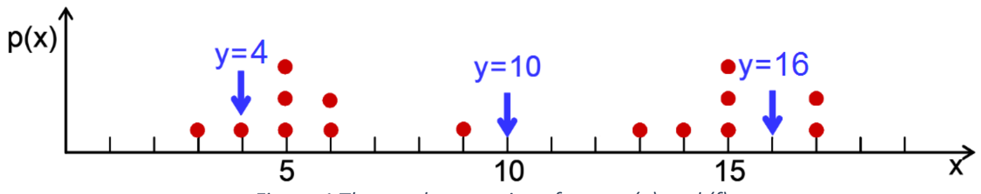


Figure 1 The toy dataset given for part (e) and (f).

2. PCA vs. LDA: Variation vs. Separation (15 Pts.)

Keywords: *Dimensionality Reduction, Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA)*

Before dealing with more challenging problems, we first warm up with several problems about **Principal Component Analysis** (**PCA**) and **Linear Discriminant Analysis** (**LDA**), the two most popular techniques in the realm of **Dimensionality Reduction**, to better comprehend their differences and master their theoretical characteristics.

First, Let X be a dataset with 10 samples, such that

$$X = \begin{pmatrix} 1 & 3 & 4 & 6 & 7 & 8 & 8 & 8 & 9 & 10 \\ 0 & 4 & 3 & 7 & 1 & 8 & 2 & 10 & 5 & 6 \\ 3 & 6 & 7 & 3 & 5 & 10 & 4 & 2 & 8 & 9 \end{pmatrix}$$

a. Find the samples mean.
b. Centre the samples to have zero mean.
c. Find the covariance matrix of the zero-mean samples. How can you describe the given samples by inspecting the obtained covariance matrix?
d. Find the Eigenvalues and Eigenvectors of the computed covariance matrix.
e. What would be the new basis of the given data?
f. Project the samples on the new basis.

Next, assume two classes w_1 and w_2, where

$$\text{samples belonging to } w_1\text{:} \begin{pmatrix} 0 & 2 & 2 & 3 & 4 & 5 & 7 & 8 & 9 & 9 \\ 3 & 1 & 2 & 1 & 4 & 5 & 2 & 3 & 5 & 6 \\ 7 & 6 & 10 & 8 & 2 & 0 & 4 & 3 & 5 & 5 \end{pmatrix}$$

$$\text{samples belonging to } w_2\text{:} \begin{pmatrix} 2 & 3 & 5 & 5 & 5 & 6 & 6 & 8 & 10 & 10 \\ 7 & 8 & 6 & 9 & 10 & 7 & 7 & 8 & 10 & 9 \\ 1 & 9 & 0 & 3 & 5 & 2 & 9 & 10 & 6 & 3 \end{pmatrix}$$

g. Find classes mean.
h. Find the covariance matrix of both classes.
i. Compute the within-class scatter matrix.
j. Compute the between-class scatter matrix.
k. Find the LDA projection.
l. Project the given samples of both classes onto the computed LDA projection.

Finally, assume a 3-category dataset with the samples given in Figure 2.

m. Draw the first principal component direction considering samples of class 1 and class 2.
n. Draw the first Fisher's linear discriminant direction considering samples of class 1 and class 2.
o. Repeat part a. considering samples of class 1 and class 3.
p. Repeat part b. considering samples of class 1 and class 3.
q. Repeat part a. considering all samples.
r. Repeat part b. considering all samples.

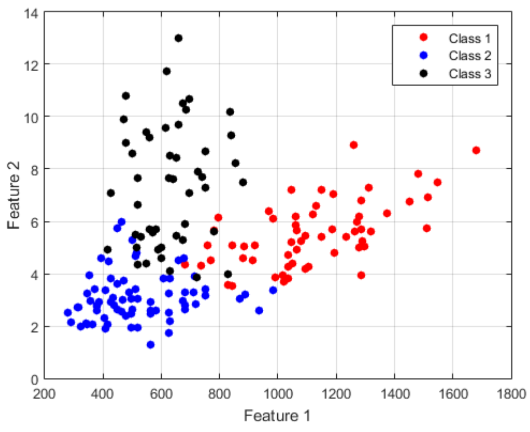


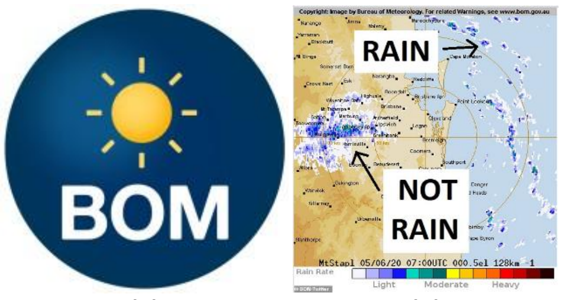
Figure 2 Distribution of the samples used in part (m) to part (r).

Note: In order to draw the required projection lines, you can use image "pca_lda_sandbox.png" (Figure 2) attached to this homework.

3. Rainy or Sunny? K-NN Answers (12 Pts.)

Keywords: *Classification Problem, K-Nearest Neighbors, Feature Selection, Missing Features*

The *Bureau of Meteorology*, or *BOM*, is an Australian governmental organization which is responsible for monitoring weather statistics and providing forecasts, warnings, and observations to the Australian public. With over seven thousand weather stations across the country, BOM records a variety of aspects of the weather in different regions, such as temperature, humidity, rainfall, air pressure, sunshine, wind speed and direction.



(a) **(b)**

Figure 3 For more than a century, BOM has been responsible for measuring and publishing weather statistics. (a) The organization logo. (b) A live map posted on BOM Twitter account.

A dataset of daily climate records gathered from different BOM stations between 2008 and 2017 is given, and the objective is to determine whether tomorrow is going to be rainy or not. First, remove those samples in which the target attribute has unassigned value. As for the remaining attributes with null values, you must set them to the median of the values of that attribute across the same exact month. To accomplish this, you can define a new column 'month' which contains the corresponding month according to the value of 'date'. Please note that you can also use the values of this column as a feature. Finally, partition the dataset into three sets 'train', 'test', and 'validation' according to appropriate proportions.

a. To prevent overfitting, we first aim to reduce the number of variables. One possible method is to detect highly correlated features and eliminate either of them from the dataset. Find the feature pairs, if any, with correlation coefficient over 0.95. Is it reasonable to keep these features in the dataset? If so, remove one of the features of each pair to your liking.
b. Next, determine the 10 best features of the dataset using sequential feature selection method.
c. Apply K–NN to the modified dataset. Find the best value of k using the validation set, and report the accuracy of the model on the test set.
d. Use the same model to forecast weather condition in the city of Rasht based on the city's similar records. You can treat the missing values similar to the previous part. Justify your observations.
e. Is it possible to determine the probability of rainfall using the same strategy? If so, how?

Note: There is no restriction on the usage of built-in functions and libraries.

4. A Few Bytes Less Please: PCA for Image Compression (16 Pts.)

Keywords: *Principal Component Analysis, Image Compression, Feature Selection*

Although **Principal Component Analysis** (**PCA**) is considered as a classical approach, yet it has numerous applications in the field of machine learning. In this problem, you are going to test PCA performance in an image processing application, i.e. **Image Compression**.

The approach considered here is based on the fact that subimages of an image can be presented using relatively few features. Therefore, instead of storing the original image, the values of the features in the subimages can be restored and then used for reconstruction.

a. First, you are working with the image "donald.png" as the input. Implement a function `patch_extract()` which takes an input image and returns a matrix of vectorised blocks of the image (known as patch) as its columns. More specifically, this function scans the input image and extracts non-overlapping 8×8 pixel blocks. Then it converts them into column vectors and construct a matrix. Here, since the image size is 448×600, the output would be a 64×4200 matrix of image blocks.

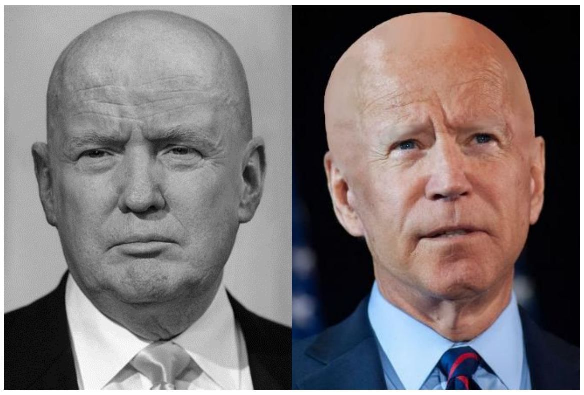


Figure 4 Two images are intended to be compressed here, one in grayscale and the other in RGB space.

b. Perform PCA analysis of the covariance matrix of the output obtained by the function `patch_extract()`. Find and report the first 20 largest eigenvalues and their corresponding eigenvectors. Also display the mean image and plot the eigenvectors (or "eigenimages") which correspond to the 8 largest eigenvalues.

c. Compress the subimages using the mean vector and the eigenvectors corresponding to the $k = 2,5,10,20$ largest eigenvalues.

d. Now try to reconstruct each subimage using its low-dimensional representation. Remember to include the mean value.

e. Now implement a function `patch_reconstruct()` to merge the reconstructed subimages and create a 448×600 pixel image again. Display the reconstructed and original images together and comment on the results. Also discuss on the effect of k.

f. Repeat the previous parts for the color image "joe.png".

Recommended MATLAB Functions: `cov()`, `eig()`, `reshape()`, `imagesc()`, `colormap()`, `subplot()`

5. Eyes ~~Never~~ Sometimes Tell Lies! (12 Pts.)

Keywords: *Density Estimation, Dimensionality Reduction, Principal Component Analysis (PCA), Generative Models, Smoothing Parameter (Bandwidth)*

Have you ever come across *This Person Does Not Exist* website? This is an online service based on generative adversarial networks (GAN) which generates a face image of a non existent identity each time the page refreshes (A wider range of similar models are accessible here). Unlike *discriminative models*, GANs and similar *generative models* can generate new samples based on the estimated densities of the input data.

Figure 5 Believe it or not, all these images do not exist!

The goal of this problem is to make use of the same strategy to generate fake images of digits. You are given a dataset of 1700 images of size 7×7, each depicting a handwritten digit.

a. Apply PCA to reduce the dimensionality of the data to 15. Specify the values of each 15 eigenvectors.
b. Find the best value for the bandwidth parameter. Clearly explain the procedure you used.
c. Generate 20 random samples from the estimated density of the data. Display these samples after converting them to 7×7 images.
d. Repeat the same process without the dimensionality reduction step, and compare the results.

Note: There is no restriction on the usage of built-in functions and libraries.

6. PCA and LDA in the Face Recognition Problem: Eigenfaces vs. Fisherfaces (20+10 Pts.)

Keywords: *Classification Problem, Face Recognition, Principal Component Analysis (PCA), Linear Discriminant Functions (LDA), K-Nearest Neighbor Classifier, Fisherfaces, Eigenfaces*

Matthew Turk and *Alex Pentland* were the first to apply **Principal Component Analysis** (**PCS**) to the problem of **Face Recognition**. Their approach – introduced in 1991 and commonly referred to as **Eigenfaces** – is considered as the first successful face recognition algorithm. **Fisherfaces**, on the other hand, is an enhancement of the Eigenfaces method which were proposed by *Joao Hespanha* in 1997. As might be expected, it applies **Fisher's Linear Discriminant Analysis** (**FLDA** or **LDA**) for the **Dimensionality Reduction**.

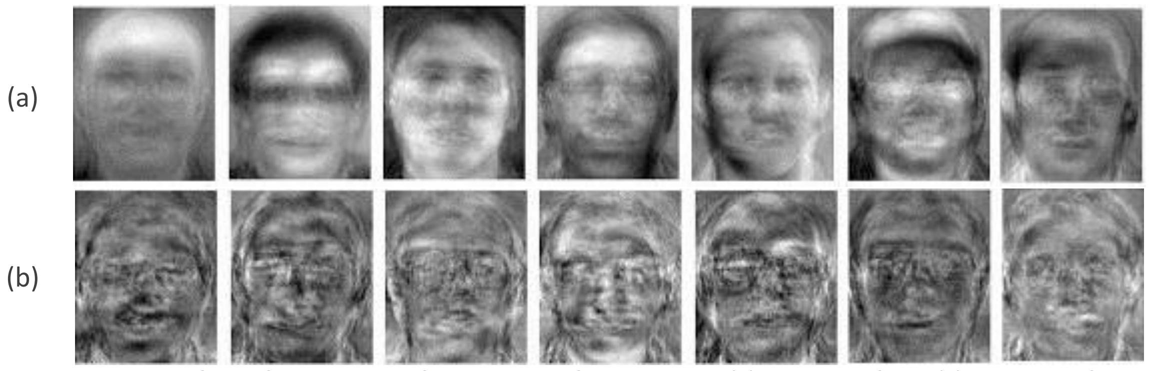


Figure 6 Examples of Eigenfaces and Fisherfaces extracted from a dataset. (a) Top 7 Eigenfaces. (b) Top 7 Fisherfaces.

In this problem, the goal is to implement Eigenfaces and Fisherfaces to address the problem of facial recognition. You will use the LFW dataset, which contains face images taken in natural conditions, Figure 7. The provided subset, however, is a smaller version of the main dataset which contains 894 cropped and grayscale images of size 80×60, representing 73 distinct individuals.

Figure 7 Images in the LFW dataset are of various degradations (pose, illumination, occlusion, etc.). Although the subset provided for this problem are cropped and converted to grayscale.

First, let's see how Eigenfaces algorithm deals with this dataset.

a. Write a function `[W,mu]=eigenfaces_train(trainset,v)` which takes $n \times d$ train set images matrix `trainset`, where $n = 821$ is the number of training images, and $d = 4800$ is the number of pixels in each image. This function performs Principal Component Analysis on the data and computes the top `v` eigenvectors. It return the $v \times d$ matrix of eigenvectors `W`, and a d–dimensional vector `mu` representing the mean of the training faces. Compute the top 50 eigenvectors.
b. Reshape each of the top 50 eigenvectors obtained in the previous part, and display them by appending them together into a 800×300 image (a grid of images containing 5 rows and 10 columns).
c. Select 10 random images from the train set, and show what each of these images would look like when using only the top v eigenvectors to reconstruct them, where $v = 1,\dots,10$.
Hint: This reconstruction procedure should project each image into a v–dimensional space, project that v–dimensional space back into a 4800 dimensional space, and finally reshape that 4800 vector into a 80×60 image.
d. Now let's test the Eigenfaces method performance. Implement a function with the header `testset_labels=eigenfaces_test(trainset,trainlabels,testset,W,mu,v)` which takes the train set matrix `trainset` as well as their labels vector `trainlabels`, test set matrix `testset` and outputs of PCA, `W` and `mu`, and the number of eigenvectors to use `v`. Your function must first project each image from `trainset` and `testset` onto the space spanned by the first `v` eigenvectors. Then it must classify each test image using the nearest neighbor (1–NN) classifier by considering L_2 distance in the lower dimensional space. It must return an m–dimensional vector `testset_labels` encoding the predicted class label for each test face, where $m = 40$ is the number of test images. Evaluate `eigenfaces_test` on the test set images for values $v = 1,\dots,50$, and plot the error rates as a function of `v`.
e. Repeat the previous part, but throw away the first 5 eigenvectors. In other words, use `v` eigenvectors starting with the 6th eigenvector. Produce a plot similar to the one in the previous step, and comment on the result. How do you explain the difference in recognition performance in comparison with the previous part?

Next, it's time to examine Fisherfaces.

⭐ f. Implement a function `[W,mu]=fisherfaces_train(trainset,trainlabels,c)` that takes the same $n \times d$ train images matrix `trainset` as well as their corresponding labels vector `trainlabels`, and the number of classes $c = 40$. Your function must do the following:
   - Compute the mean `mu` of the training data, and use PCA to compute the first $n - c$ principal components. Let this be W_{PCA}.
   - Use the obtained W_{PCA} to project the training images into a $n - c$ dimensional space.
   - Compute the between-class scatter matrix S_B and the within-class scatter matrix S_W on the $n - c$ dimensional space from the previous space.
   - Compute W_{FLD} by solving for the generalised eigenvectors of the $c - 1$ largest generalised eigenvalues for the problem $S_B w_i = \lambda_i S_W w_i$.
     Hint: In MATLAB, you can use the function `eig()` to solve for the generalised eigenvalues of S_B and S_W.
   - The fisher bases will be a $W = W_{FLD}\, W_{PCA}$, where W is $(c-1) \times d$ dimensional, W_{FLD} is $(c-1) \times (n-c)$ dimensional, and W_{PCA} is $(n-c) \times d$ dimensional.

g. As in the Eigenfaces method, reshape the top 50 Fisher bases you obtained in the previous part into images of the size 80×60 and stack them into one big 800×300 image.

h. Perform recognition on the test set with Fisherfaces. As in the Eigenfaces exercise, use a L_2 nearest neighbor classifier (1–NN), and evaluate results for each test images for values $v = 1,\ldots,50$. Plot the error rates as a function of `v`.

Hint: Images of the given dataset are stored in `double` format. It means that if you want to use MATLAB function `imshow()` to display them, first you need to convert them to 8-bit unsigned integers, i.e. `uint8`.

Useful Sources: [1], [2], [3]

Recommended MATLAB functions: `csvread()`, `colormap()`, `imagesc()`, `reshape()`, `eigs()`, `svds()`, `eig()`, `subplot()`

7. Some Explanatory Questions (8 Pts.)

Please answer the following questions as clear as possible:

a. Which one of the two introduced density estimation methods, Parzen window and K–NN, is more prone to outliers? Explain.

b. Determine the training error of a K–NN model, when $k = 1, 3, n$ (n is the size of training set).

c. Define an arbitrary two-category classification problem in 2D feature space in which selecting different distance metrics affect K–NN results. More specifically, you must determine number of samples as well as their feature values.

d. Under what circumstance(s) are the estimated densities obtained by Parzen window and K–NN methods the same?

e. Does it make any sense to apply PCA with the number of PCs greater than the dimensions? Justify your answer.

f. Explain in your own words how PCA could be used to address the problem of image noise removal.

g. When dealing with data of high dimensionality like images, PCA suffers from a numerical computational issue. Explain what it is and how it can be avoided.

Good Luck!
Ali Abbasi