Assignment #1

s

Tushar Chopra

tushar.chopra@uwaterloo.ca

1. Consider two classes described by the covariance matrices below (assume zero mean)

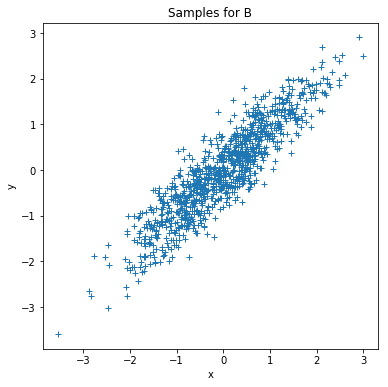
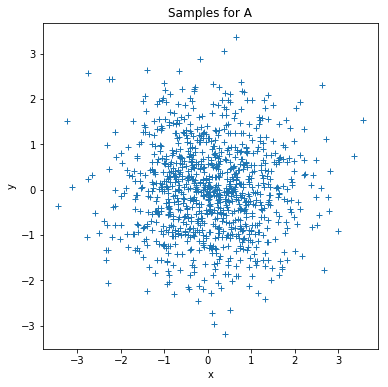
a.∑=[ [1,0] [0,1] ] b.∑=[ [1,0.9] [0.9,1] ]

1. For each matrix generate 1000 data samples and plot them on separate figures.
2. For each case calculate first standard deviation contour as a function of the mean, eigenvalues, and eigenvectors. Show your calculation (Hint: consider distribution whitening from the tutorial). You may use pre-existing functions for Eigen computation.  Plot each contour on the respective plots from part (a).
3. Calculate sample covariance matrices for each class using the data generated in part  (a). Do not use a Python/Matlab function for computing the covariance.
4. Compare the given covariance matrix for each class with the corresponding sample  covariance matrix generated in (b).

Explanations: -

1. As, the covariance and mean of both the classes are given. We used np.random.multivariate\_normal function which is used to generate random samples with Gaussian distribution of mean 0 and covariance of a and b.

Results:-



Distribution for class a Distribution for class b.

1. YOLO

2. Consider a 2D problem with 3 classes where each class is described by the following priors, mean vectors, and covariance matrices.

P(C1) = 0.2

μ1 = [3 2] T𝑇

∑1= [[1 −1] [ −1 2]]

P(C2) = 0.3

μ2 = [5 4] T

∑2= [[1 −1] [−1 7]]

P(C3) = 0.5

μ3 = [2 5] T𝑇

∑3= [[0.5 0.5] [0.5 3]]

1. Create a program to plot the decision boundaries for a ML and MAP classifier. Plot the means and first standard deviation contours for each class. Discuss the differences between the decision boundaries.
2. Generate a 3000 sample dataset using the prior probabilities of each class. For both the ML and MAP classifiers: classify the generated dataset, calculate a confusion matrix, and calculate the experimental P(ε). Discuss the results.

Explanations: -

1. The prior probability, mean and covariance matrix were given for 3 classes (C1, C2, C3). To get the decision boundary we generated numpy mesh-grid, and for every point(x) on mesh-grid, we calculated likelihood (Conditional probability of observation 𝒙 given class 𝐶 i.e P(x/Ck)). The equation of likelihood is given below.

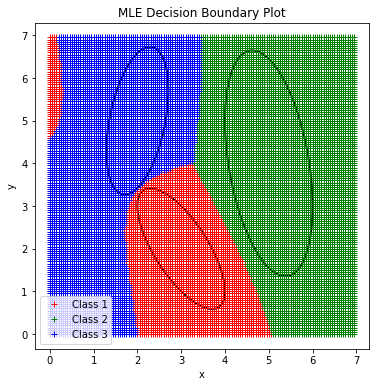
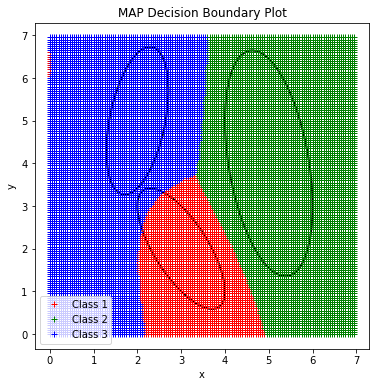
Likelihood or P(x/Ck) = \_\_\_\_1\_\_\_\_ \* exp(-(0.5)(*x*−*m*)*TS*−1(*x*−*m*))

(*2\*pi*)*l/2*|*S*|*1/2*

Now, the boundary of MAP classifier is defined as “i”, if P(x|Ci)\*P(Ci) > P(x|Cj)\*P(Cj) and vise verse, that is, the point with the higher value of P(x|Ck)\*P(Ck) is classified in k class. Similarly, the boundary of ML classifier is defined as “i”, if P(x|Ci) > P(x|Cj), that is, the point with the higher value of P(x|Ck) is classified in k class. For all the classes we have given P(Ck) and by using the equation of Likelihood, we get the class of new point(x) using MAP and ML classifiers.

YOLOOOOOOOOO

Results:-

1. YOLOOOOOOOOOOOO

3. The MNIST dataset contains a set of images containing the digits 0 to 9. Each image in the data set is a 28x28 image. The data is divided into two sets of images: a training set and a testing set. The MNIST dataset can be downloaded from http://yann.lecun.com/exdb/mnist/. Use only the training set to perform this part.

1. Program PCA that takes X(DxN) and returns Y(dxN) where N is the number of samples, D is the number of input features, and d is the number of features selected by the PCA algorithm. Note that you must compute the PCA computation method by yourself. You may use pre-existing functions for Eigen computation.
2. Propose a suitable d using proportion of variance (POV) =95%.
3. Program PCA reconstruction that takes 𝑌 (dxN) and returns 𝑋 (DxN) (i.e., a  reconstructed image). For different values of d= {1, 2, 3, 4, ..., 784} reconstruct all samples and calculate the average mean square error (MSE). Plot MSE (y-axis) versus d (x-axis). Discuss the results.
4. Reconstruct a sample from the class of number ‘5’ and show it as a ‘png’ image for d= {1, 10, 50, 250, 784}. Discuss the results.
5. For the values of d= {1, 2, 3, 4, ..., 784} plot eigenvalues (y-axis) versus d (x-axis). Discuss the results.

Explanations: -

1. Algorithm PCA: -
2. 𝝁 = sample mean of 𝑋, Σ = sample covariance of
3. 𝑋\_hat = subtract sample mean 𝝁 from each column sample (𝑋\_hat has zero mean)
4. Find eigenvectors and eigenvalues of Σ
5. 𝑊 = Using 𝑑 (𝑑 < 𝐷) eigenvectors with largest eigenvalues form the mapping function as a 𝐷 × 𝑑 matrix, each column corresponds to an eigenvector
6. The transformed samples will be 𝑌 = *W*T𝑋\_hat

The MNIST data is used from the link given in the question(<http://yann.lecun.com/exdb/mnist/)>. In code, “pca\_from\_no\_dimentions()” is used to calculate PCA which gives data in lower number of dimensions. It takes 2 arguments, one is dataset (“data”) and other is the required number of dimentions (“dimentions”). After execution, it returns lower dimention data, eigen vectors of data which have largest eigen values, eigen values and mean.

Results:-

The MNIST data is used from the link given in the question(<http://yann.lecun.com/exdb/mnist/)>. In code, “pca\_from\_no\_dimentions()” is used to calculate PCA which gives data in lower number of dimensions. It takes 2 arguments, one is dataset (“data”) and other is the required number of dimentions (“dimentions”). After execution, it returns lower dimention data, eigen vectors of data which have largest eigen values, eigen values and mean.