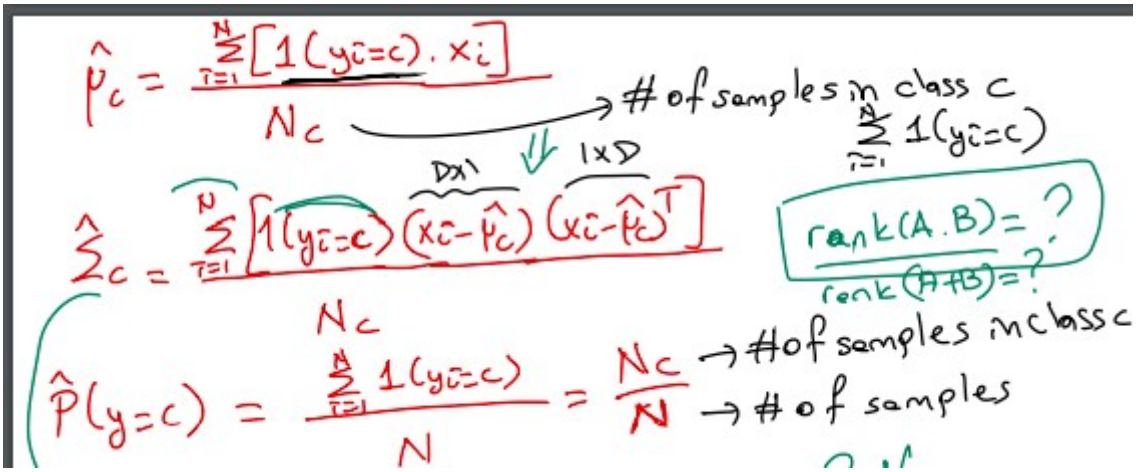
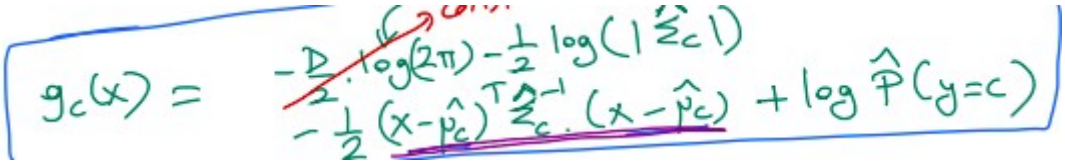
Engr421 Hw Report
Can Gözpınar 68965

2. To generate random data points first I stored the mean and covariance values in the matrix form using numpy's array method in my code then, used numpy's multivariate_normal method by passing it the mean, covariance and sample variables to generate data points.

3. I estimated the parameters using the equations we derived in the class for multivariate normal distribution which minimizes our squared error function. Sample mean and sample covariance maximizes our variables so I calculated those and stored them in parameters for later use. The three equations in red below are the ones I implemented in my code.



4. For calculating the confusion matrix, I first needed a goal function. I implemented my goal function using the equation we derived in the class which, I've provided below for your reference. I classified data points using the goal function formula I just mentioned in the given way. I calculated goal values for each class for every value then, assigned the class which returned the highest goal value to the point. After having classified my points using estimated parameters, I iterated through each point to check whether which class it actually belonged and in which class my estimated function predicted and stored them in an array which I named as confusionMatrix. Printing the confusion matrix displays the truth and predicted values of each class in a table form which is easier to read.



5. For drawing the decision boundaries, boundaries are the points that have the same value for the goal functions between two classes. In our case we had three classes therefore, decision boundaries are three lines which are g1(x) – g2(x) = 0 , g1(x) – g3(x) = 0 , g2(x) – g3(x) = 0. I used the numpy's meshgrid and linspace() function to represent the space of my plot and used contour function to draw the lines on my plot. Also I had to change the values of the discriminant values to np.nan at values that whenever the given goal value for a class was the smallest among the third so that we get three lines intersecting at the middle point since, our class size was three. For the contouring the background I used the contourf function. I used a logic to manipulate the discriminant values of classes. I kept only the values that the given class has the highest goal

function. In other words, I removed the mid and lowest valued values from discriminant matrices of each class so that when contouring using contourf, it colored only the coordinates that our program would classify as belonging the the same class as we colored.