Exercise 0: What are the new correctly classified instances, and the new confusion matrix? Briefly explain the reason behind your observation. [1 mark]

Linear Regression

a b <-- classified as

overall True Positive(700) TP(605) FN(95) | a = good

overall True Negative(300) FP(153) TN(147) | b = bad

X Y

X = Overall Predicted good(a) class = 758

Y = Overall Predicted bad(b) class = 242

K means K=999

a b <-- classified as

overall True Positive(700) TP(700) FN(0) | a = good credit score

overall True Positive(300) FP(300) TN(0) | b = bad credit score

X Y

X = Overall Predicted good(a) class = 1000

Y = Overall Predicted bad(b) class = 0

False Positive meaning predicted positive class instead of negative class, here it means predicted good instead of bad which is a very important indicator which tells the classifier is doing something wrong. Second it fails to classify any instance as bad as overall predicted bad(b) class is 0. Thus, it is a bad classifier. Here we are using K factor of KNN which indicate number of neighboring samples it will consider before assigning a data sample a class here if we set K=999 then all the samples will be used to assign a data point a class.

Exercise 1: Try (by varying the values of weights in the code), and describe the effect of the following actions on the classifier (with a brief explanation): [0.5 mark]

- changing the first weight dimension, i.e., $w0$?
- changing the second two parameters, $w1$, $w2$?
- negating all the weights?
- scaling the the second two weights up or down (e.g. dividing or multiplying both $w1$ and $w2$ by 10)?

1. When we change the first weight i.e. w0 or the intercept from extreme negative to extreme positive we observe that the probability region defining the dots and crosses into different class expand or condense.

2. If we change the w1 and w2 then angle of intercept change and it is centered around 0.

3. when we negate all the weights [-0,5,-0] the probability region inverses the red crosses are in red region and vice-versa. If we consider P(A).P(1 - A) as the initial likelihood then negating the weights will change the likelihood to P(1-A).P(A).

4. As we increase (scale up) the weights of w1 and w2 the probability region describing in-between probabilities decreases and the separable line becomes more evident. But if we decrease (scale down) both w1 and w2 then it becomes difficult to predict in which class dots and cross lies.

Exercise 1: (a) How many times did your loop execute? (b) Report your classifier weights that first get you at least 75% train accuracy. [0.25 + 0.25 = 0.5 mark]

a) the loop executed 8000 times

b) accuracy: 0.75 --- weights = [-0.579 1.000 0.789]

Exercise 2: (a) Notice that we used the negative of the gradient to update the weights. Explain why. (b) Given the gradient at the new point, determine (in terms of "increase" or "decrease", no need to give values), what we should do to each of $w0$, $w1$ and $w2$ to further improve the loss function. [0.5 + 0.5 = 1 mark]

a) Because we used gradient of negative log likelihood which is `-np.matmul(X.T, Xprob) + 2. * LAMBDA * w` which tells us the direction of gradient descent, to attain global minima, we use the negative of the gradient to update the weights, `weights = weights - step_size * gradientLL`.

b) Since we have move in opposite direction of gradient descent to improve the loss function, we have to decrease the value of w0 and increase the values of w1 and w2

Exercise 3: Suppose we wanted to turn this step into an algorithm that sequentially takes such steps to get to the optimal solution. What is the effect of "step-size"? Your answer should be in terms of the trade-offs, i.e., pros and cos, in choosing a big value for step-size versus a small value for step-size. [0.25 + 0.25 = 0.5 mark]

Step-size or the learning rate governs the Gradient Descent. Consider an example of a person on a hill, and he must reach the sea. With a high learning rate, we can cover more ground each step, but we risk overshooting the lowest point. With a very low learning rate, we can confidently move in the direction of the negative gradient since we are recalculating it so frequently and it becomes time consuming. A low learning rate is more precise, but calculating the gradient is time-consuming, so it will take us a very long time to get to the bottom.

Exercise 4: Answer this question either by doing a bit of research, or looking at some source codes, or thinking about them yourself: (a) Argue why choosing a constant value for step_size, whether big or small, is not a good idea. A better idea should be to change the value of step-size along the way. Explain whether the general rule should be to decrease the value of step-size or increase it. (b) Come up with the stopping rule of the gradient descent algorithm which was described from a high level at the start of this section. That is, come up with (at least two) rules that if either one of them is reached, the algorithm should stop and report the results. [0.5 + 0.5 = 1 mark]

a) A constant value of step_size is not a good idea, let alone the size being small or big. The general idea should be at the start it should be moderate (not to large and not to small) and then the step_size should decrease at constant rate. This is because if we do the opposite of this then we may not be able to find the global minima as we have the chances of overshooting and if we keep the step-size small then it will take a very long time to reach. The approach described above will attain global minima in less time.

b) We can employ two stopping rules. First, if the current error calculated using gradient descent is bigger than previous then one should stop. Second, limiting the number of iterations to a certain value with decreasing step-size which tends to zero

Exercise 5: Provide one advantage and one disadvantage of using exhaustive search for optimisation, especially in the context of training a model in machine learning. [0.25+0.25 = 0.5 mark]

If we have a large dataset with large sample size and large dimensions, then using exhaustive search for optimization is expensive both in terms of computation resource and time this is the biggest disadvantage of exhaustive search. The only advantage of exhaustive search is that it will give best accuracy as it is a brute force technique, it will combine all possible combination to find one.

Exercise 6: (a) What is the best value for K? Explain how you got to this solution. (b) Interpret the main trends in the plot. In partiular, specify which parts correspond to under-fitting and which part with over-fitting. [0.5+0.5 = 1 mark]

a) The best value of K = 6. The elbow method helps in deciding the value of K, the graph is a plot of error rate and different values of K, we can see at K=6 the error rate was lowest.

b) The graph is a plot of increasing K values vs error rate, which describe the best K to pick in for KNN. This graph is particularly following elbow method, when the value of K is between 0 and 5 the error rate decreases this is where underfitting can be seen at K=6 the error value is lowest around 0.25 and after this as the value of K increases the error rate increases, this is where overfitting can be seen.

Exercise 7: Given what you learned from this entire lab(!), provide one advantage and one disadvantage of using MaxEnt vs KNN. [0.25+0.25 = 0.5 mark]

MaxEnt/Linar regression Advantage: Simple to use, and different apporaches such as exhaustive search or gradient descent can be used Disadvantage: Non applicable for nonlinear problem

K Nearest Neighbour Advantage: Applicable to nonlinear problems Disadvantage: For large dataset KNN is expensive (time and compute resources) to apply as it is an instance-based learning and is model free.