CPEN 355 Homework 2

Aloysio Kouzak Campos da Paz
Student ID 58687526

Collaboration statement I completed the assignment with the help of the internet and ChatGPT for understanding the concepts behind the assignment and for generating Python code.

Here is a link to the chat I had with ChatGPT:
https://chat.openai.com/share/a2efffc7-d5d1-4de7-85bb-a267e0331e4f

1 Logistic Regression

We will use the following equation for solving part a and b of question 1:

$$\sigma_\Theta(x_1, x_2) = P(\hat{y} = 1|x_1, x_2) = \frac{1}{1 + e^{-f_\Theta(x1,x2)}} \tag{1}$$

Where

$$f_\Theta(x1, x2) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2 \tag{2}$$

We will simplify notation by saying P(y) = $P(\hat{y} = 1|x_1, x_2)$

1.1 Question 1 a

We begin by setting up the logistic model plugging in the values of theta we want to start with:

$$\theta_0 = -1, \theta_1 = -1.5, \theta_2 = 0.5 \tag{3}$$

$$P(\hat{y} = 1|x_1, x_2) = \frac{1}{1 + e^{1+1.5*x_1-0.5*x_2}} \tag{4}$$

No we find the cross entropy loss function:

$$L(y, P(y)) = -y * ln(P(y)) - (1 - y) * ln(1 - P(y)) \tag{5}$$

$$L(y, P(y)) = -y * ln(1/(1 + e^{1+1.5*x_1-0.5*x_2})) - (1 - y) * ln(1 - 1/(1 + e^{1+1.5*x_1-0.5*x_2})) \tag{6}$$

1.2 Question 1 b

We will use gradient descent to update θ_0, θ_1 and θ_2 for ONE iteration.

We will do batch gradient descent, therefore we will use all x_1 and x_2 values in the training set.

Sample ID	x_1	x_2	y
1	2.35	1.87	0
2	2.15	2.01	0
3	2.09	1.42	0
4	2.78	2.32	0
5	3.02	0.98	1
6	2.44	0.75	1
7	2.87	0.68	1
8	2.18	0.92	1

Table 1: Training data

Now, for each of the 8 pairs in our training data of x1 and x2 values, we calculate the probability that $P(\hat{y} = 1|x_1, x_2)$

For example, for the pair $x1 = 2.35, x2 = 1.87$, we have

$$P(\hat{y} = 1|x_1 = 2.35, x_2 = 1.87) = 1/(1 + e^{1+1.5*2.35-0.5*1.87}) = 0.0269 \tag{7}$$

Then, we use each value we found for $P(\hat{y} = 1|x_1, x_2)$, which are 8 values in total, to calculate the gradient of the loss function with respect to θ_0, θ_1 and θ_2.

For example, for the first data points $x1 = 2.35, x2 = 1.87$, we have y = 0, so we find the following three derivatives:

$$\frac{dL}{d\theta_0} = P(\hat{y} = 1|x_1 = 2.35, x_2 = 1.87) - y = 0.0269 - 0 = 0.0269$$
$$\frac{dL}{d\theta_1} = (P(\hat{y} = 1|x_1 = 2.35, x_2 = 1.87) - y) * x_1 = 0.0269 * 2.35 = 0.631$$
$$\frac{dL}{d\theta_2} = (P(\hat{y} = 1|x_1 = 2.35, x_2 = 1.87) - y) * x_2 = 0.0269 * 1.87 = 0.052$$

We repeat this for each pair of x1, x2 and calculate 8 different derivatives for each theta. Then we add all 8 derivatives with respect to θ_0 to each other to come up with a single value representing $\frac{dL}{d\theta_0}$. We also add all derivatives with respect to θ_1, and all derivatives with respect to θ_2. I solved this using Python, so it's faster to calculate.

We find that:

$$\frac{dL}{d\theta_0} = -3.8367...$$
$$\frac{dL}{d\theta_1} = -10.1291...$$
$$\frac{dL}{d\theta_2} = -3.0753...$$

Finally, we can update the values for θ_0, θ_1 and θ_2 by using the gradients we added up, and a learning rate $\alpha = 0.1$.

$$new\theta_0 = \theta_0 - \alpha * \frac{dL}{d\theta_0} = -1 - 0.1 * (-3.8367) = -0.61633...$$
$$new\theta_1 = \theta_1 - \alpha * \frac{dL}{d\theta_1} = -1.5 - 0.1 * (-10.1291) = -0.48709...$$
$$new\theta_2 = \theta_2 - \alpha * \frac{dL}{d\theta_2} = 0.5 - 0.1 * (-3.0753) = 0.80753...$$

We can use the new θ_0, θ_1 and θ_2 to calculate the new loss of the model, using the equation (5) shown in question 1a: $L(y, P(y)) = -y * log(P(y)) - (1 - y) * log(1 - P(y))$

For each pair of x1 and x2 values in our training data, we calculate a probability $P(\hat{y} = 1|x_1, x_2)$. Then we plug this probability in the loss equation (5). We will obtain 8 values for loss (one loss for each x1,x2 pair). We sum the 8 loss values up to obtain one loss value.

The summed loss for the new thetas we found using ONE iteration of this algorithm is:

$$L(y, P(y)) = 8.3130$$

We can use the new values for θ_0, θ_1 and θ_2 to repeat the algorithm and find new thetas again and again.

We use Python to repeat this algorithm until the loss converges, meaning that the difference between a new loss and the old loss is 0.000001.

$$FinalTheta : [1.67066084, 9.32557632, -18.46849794]$$

$$Loss : 0.01300383305932358$$

$$TotalIterations : 12532$$

1.3 Question 1 c

We will visualize the training set and the model's decision hyperplane at initialization, after one iteration, and after convergence.

We derive the decision boundary line by setting

$$f_\Theta(x1, x2) = 0$$

Therefore we solve for x2 in the equation

$$0 = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2$$

$$x_2 = boundary = -(\theta_0 + \theta_1 * x_1)/\theta_2$$

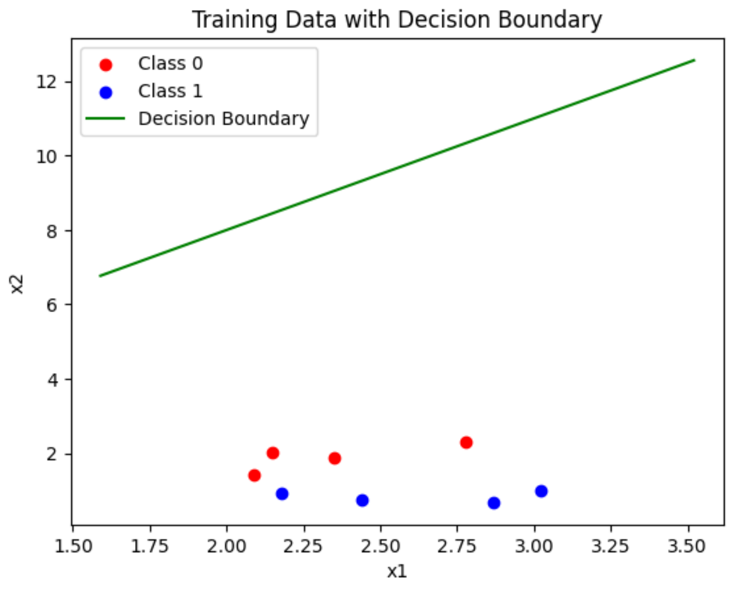

Figure 1: Training data and decision boundary at initialization with theta 0 = -1, theta 1 = -1.5, and theta 2 = 0.5

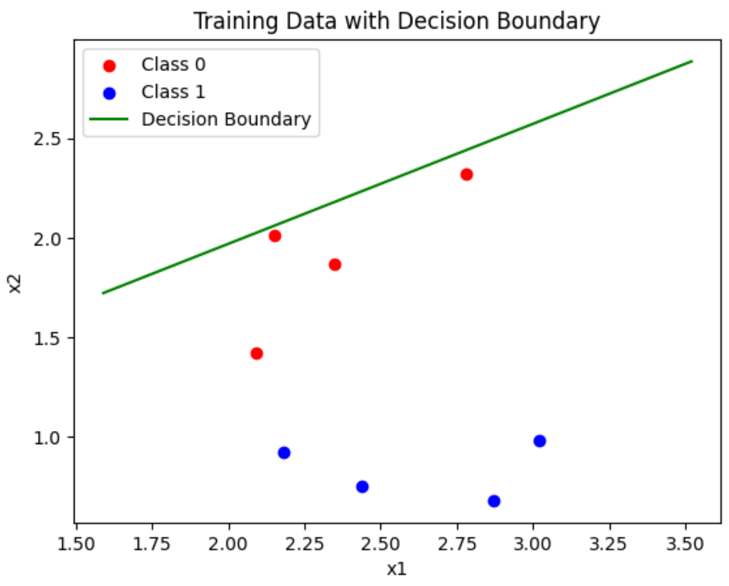

Figure 2: Training data and decision boundary after one iteration, with theta 0 = -0.61633, theta 1 = -0.48709, and theta 2 = 0.80753

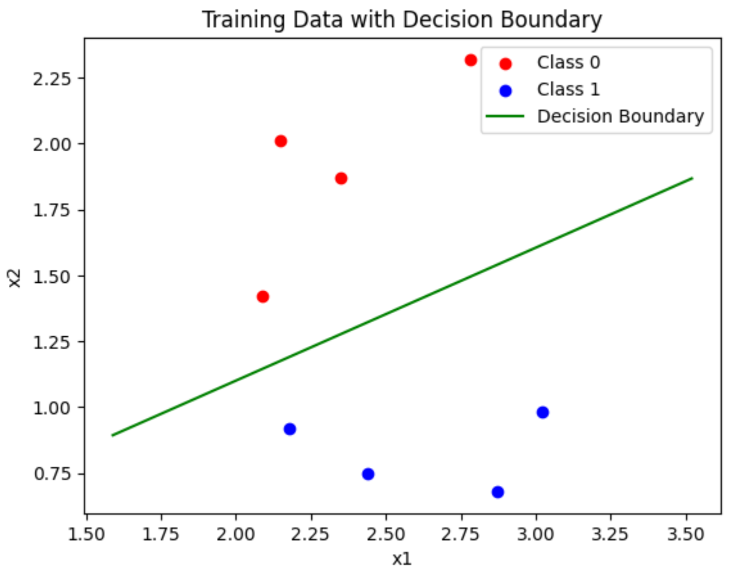

Figure 3: Training data and decision boundary after convergence, with theta 0 = 1.6706..., theta 1 = 9.3255..., and theta 2 = -18.4684...

1.4 Question 1 d

We will now use the library called sklearn, which has a model for Logistic Regression, to train their model using our training data. Then we will compare the results of sklearn's model with our own model from part b.

Training sklearn's logistic regression model, we find:

$$\theta_0 = 0.7904..., \theta_1 = 0.3863..., \theta_2 = -1.2830... \tag{8}$$

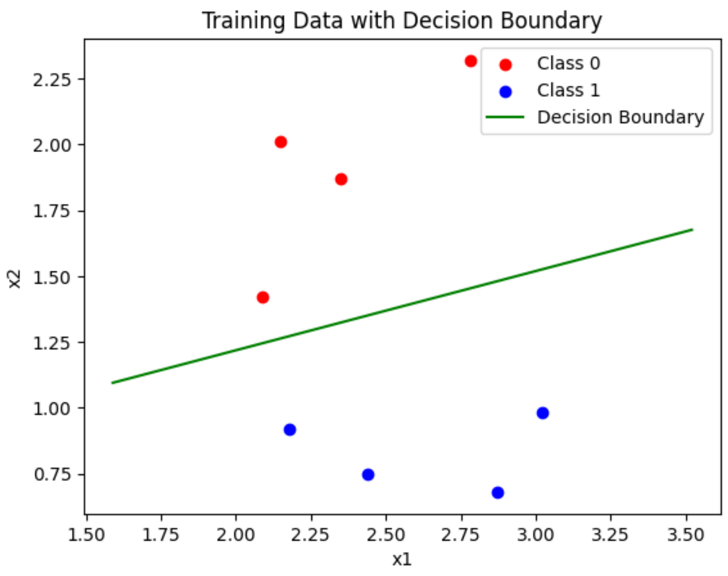

Figure 4: Training data and decision boundary using Sklearn's model, with theta 0 = 0.7904..., theta 1 = 0.3863..., and theta 2 = -1.2830...

1.5 Question 1 e

We will use sklearn's model and our own model on the testing data and compare their accuracy, precision and recall.

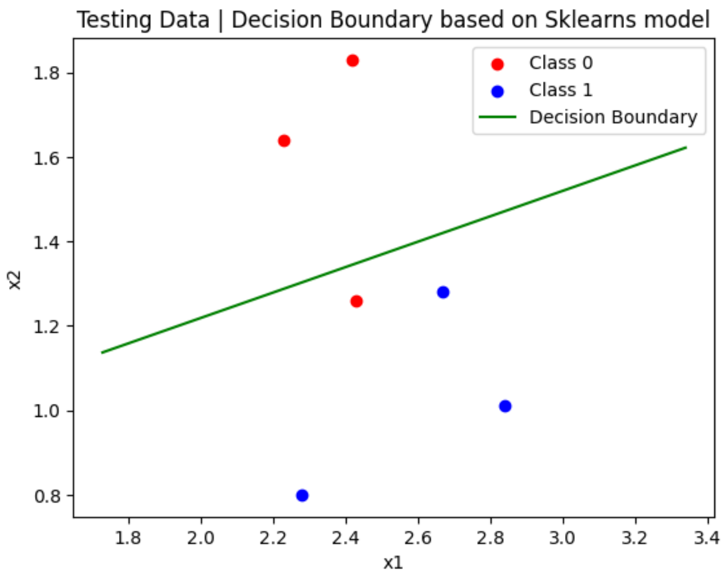

Figure 5: Testing data and decision boundary using Sklearn's model

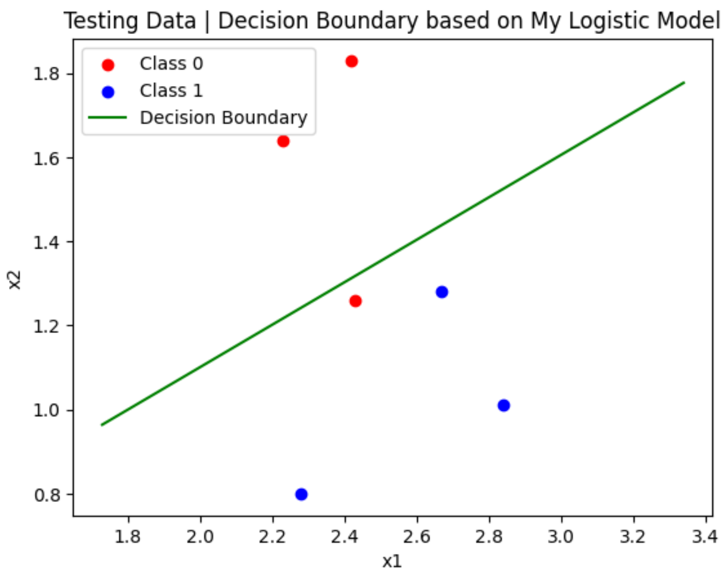

Figure 6: Testing data and decision boundary using my model

Now we will calculate the precision, accuracy and recall of both models. We need to count the number of samples that are true positive, true negative, false positive and false negative based on the decision boundary.

Let's call class 0, the red class on the plot, the positive; let's call class 1, the blue class on the plot, the negative;

Let's start with the model we trained:

The number of true positive (TP) samples for the model we trained are the red samples above the decision boundary:

$$TP_{ourmodel} = 3$$

The number of true negative (TN) samples for the model we trained are the blue samples under the decision boundary:

$$TN_{ourmodel} = 3$$

The number of false positive (FP) samples for our model are the blue samples above the decision boundary:

$$FP_{ourmodel} = 0$$

The number of false negative (FN) samples for our model are the red samples under the decision boundary:

$$FN_{ourmodel} = 1$$

The accuracy for the model is calculated with the formula

$$Accuracy_{ourmodel} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{3 + 3}{3 + 3 + 0 + 1} = 0.857...$$

$$Precision_{ourmodel} = \frac{TP}{TP + FP} = \frac{3}{3 + 0} = 1$$

$$Recall_{ourmodel} = \frac{3}{3 + 1} = 0.75$$

Now we repeat the same procedure to count the sample points that are TP, TN, FP, FN for the sklearn model. We find

$$TP_{sklearn} = 3$$
$$TN_{sklearn} = 3$$
$$FP_{sklearn} = 0$$
$$FN_{sklearn} = 1$$

$$Accuracy_{sklearn} = 0.857...$$

$$Precision_{sklearn} = 1$$

$$Recall_{sklearn} = 0.75$$

2 Short Answer Questions

2.1 Question 2a

False, because the loss function only reaches zero as $f(x1, x2) - > infinity$.

We can see this by looking at the cross entropy loss equation:

$$L(y, P(y)) = -y * ln(P(y)) - (1 - y) * ln(1 - P(y)) \tag{9}$$

Loss = 0 only when: P(y) reaches 0 if y = 0, or P(y) = 1 if y = 1.

Remember that

$$P(\hat{y} = 1|x_1, x_2) = \frac{1}{1 + e^{-f_\Theta(x1,x2)}} \tag{10}$$

$P(\hat{y} = 1|x_1, x_2)$ will only reach 1 when $e^{-f(x1,x2)}$ reaches 0, which will only happen if $f(x1, x2)$ goes to infinity.

2.2 Question 2b

The model **accuracy remains the same**, because when we multiply the theta parameters of the model by 2, we have the same decision boundary.

We can see this by looking at the equation we used for drawing a decision boundary

$$x_2 = boundary = -(\theta_0 + \theta_1 * x_1)/\theta_2$$

If θ_0, θ_1, and θ_2 all double, we can factor out the 2 from θ_0 and θ_1 in the numerator, and divide with the factor of 2 from θ_2 in the denominator. So the decision boundary remains the same.

2.3 Question 2c

We can't determine the effect of doubling the theta parameters on the loss of the model without more information on whether $f_\Theta(x1, x2)$ is positive or negative to start with.

If θ_0, θ_1, and θ_2 all double, our equation $f_\Theta(x1, x2) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2$ will double as well since we can factor out the 2 from each theta and have

$$f_\Theta(x1, x2) = 2 * \theta_0 + 2 * \theta_1 * x_1 + 2 * \theta_2 * x_2$$
$$f_\Theta(x1, x2) = 2 * (\theta_0 + \theta_1 * x_1 + \theta_2 * x_2)$$

Since $f_\Theta(x1, x2)$ doubled, we will have a different probability function:

$$P(\hat{y} = 1|x_1, x_2) = \frac{1}{1 + e^{-2*f_\Theta(x1,x2)}} \tag{11}$$

If $f_\Theta(x1, x2)$ is positive, doubling $f_\Theta(x1, x2)$ will make $P(\hat{y} = 1|x_1, x_2)$ closer to 1. If $f_\Theta(x1, x2)$ is negative, doubling the $f_\Theta(x1, x2)$ will make $P(\hat{y} = 1|x_1, x_2)$ closer to 0.

Assume y = 0. Then, our loss is:

$$L(y, P(y)) = -ln(1 - \frac{1}{1 + e^{-2*f_\Theta(x1,x2)}}) \tag{12}$$

If $f_\Theta(x1, x2)$ is positive and we doubled the parameters, $\frac{1}{1+e^{-2*f_\Theta(x1,x2)}}$ is closer to 1. Therefore, the loss increases.

If $f_\Theta(x1, x2)$ is negative and we doubled the parameters, $\frac{1}{1+e^{-2*f_\Theta(x1,x2)}}$ is closer to 0. Therefore, the loss decreases.

The effect on the loss will depend on whether $f_\Theta(x1, x2)$ was negative or positive before doubling, so an increase or decrease of loss depends on the situation.