CS434_HW2-3

Lyon Kee

October 28, 2023

1 Written Exercises: Linear Regression and Precision/Recall [5pts]

1.1 Least Absolute Error Regression

1.1.1 Q1 Linear Model with Laplace Error [2pts]

$$y_i \sim Laplace(\mu = w^T x_i, b) \rightarrow P(y_i|x_i, w) = \frac{1}{2b} e^{-\frac{|y_i - w^T x_i|}{b}}$$

$$\mathcal{L}(\theta) = \prod_{i=1}^{N} \frac{1}{2b} e^{-\frac{|y_i - w^T x_i|}{b}}$$

$$\mathcal{LL}(\theta) = N \log \frac{1}{2b} - \frac{1}{b} \sum_{i=1}^{N} |y_i - w^T x_i|$$

$$\therefore \text{To maximize } \mathcal{L}(\theta) \text{, we need to minimize } |y_i - w^T x_i|$$

$$SAE(w) = \sum_{i=1}^{N} |y_i - w^T x_i|$$

1.2 Recall and Precision

1.2.1 Q2 Computing Recall and Precision [3pts]

t = 0:

$$Recall = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalseNegatives}} = \frac{8}{8+0} = 1$$

$$Precision = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalsePositives}} = \frac{8}{8+8} = 0.5$$

t = 0.2:

$$Recall = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalseNegatives}} = \frac{8}{8+0} = 1$$
$$Precision = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalsePositives}} = \frac{8}{8+6} = 0.571$$

t = 0.4:

$$Recall = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalseNegatives}} = \frac{6}{6+2} = 0.75$$
$$Precision = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalsePositives}} = \frac{6}{6+3} = 0.667$$

t = 0.6:

$$Recall = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalseNegatives}} = \frac{6}{6+2} = 0.75$$
$$Precision = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalsePositives}} = \frac{6}{6+1} = 0.857$$

t = 0.8:

$$Recall = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalseNegatives}} = \frac{4}{4+4} = 0.5$$
$$Precision = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalsePositives}} = \frac{4}{4+1} = 0.8$$

t = 1:

$$Recall = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalseNegatives}} = \frac{0}{0+8} = 0$$
$$Precision = \frac{\text{\#TruePositives}}{\text{\#TruePositives} + \text{\#FalsePositives}} = \frac{0}{0+0} = N/A$$

2 Implementing Logistic Regression for Tumor Diagnosis [20pts]

2.1 Implementing Logistic Regression

2.1.1 Q3 Negative Log Likelihood [2pt]

2.1.2 Q4 Gradient Descent for Logistic Regression [5pt]

```
Training logistic regression model (No Bias Term; step_size=1)
Learned weight vector:
[-341.33, 1627.2351, 452.8004, 651.8745, -1098.5961, -497.7727, 866.9366, -506.6063]
Train accuracy: 74.68%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=0.1)
Learned weight vector:
[-62.4709, 148.2739, 34.3275, 60.0743, -118.1612, -61.9209, 84.7914, -60.126]
Train accuracy: 85.84%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=0.01)
Learned weight vector:
[-4.7255, 16.5512, 3.9401, 6.4157, -12.4888, -5.9264, 8.8469, -5.9882]
Train accuracy: 84.33%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=0.0001)
Learned weight vector:
[-0.2464, 0.8677, 0.2008, 0.2785, -0.6761, -0.3325, 0.4337, -0.3499]
Train accuracy: 86.27%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=1e-05)
Learned weight vector:
[-0.2371, 0.4447, 0.2724, 0.1993, -0.3705, -0.2156, 0.2652, -0.212]
Train accuracy: 85.62%
--------------------------------------------------------------------------------------
```

2.2 Playing with Logistic Regression on This Dataset

2.2.1 Q5 Adding A Dummy Variable [1pt]

    \begin{verbatim}

```
Training logistic regression model (No Bias Term; step_size=1)
Learned weight vector:
```

```
[-341.33, 1627.2351, 452.8004, 651.8745, -1098.5961, -497.7727, 866.9366, -506.6063]
Train accuracy: 74.68%
---------------------------------------------------------------------------
Training logistic regression model (Added Bias Term; step_size=1)
Learned weight vector:
[-4323.392, 257.8921, -84.8401, 212.1003, 239.9524, 30.2812, 142.2181, 118.2768
    , 343.3145]
Train accuracy: 95.92%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=0.1)
Learned weight vector:
[-62.4709, 148.2739, 34.3275, 60.0743, -118.1612, -61.9209, 84.7914, -60.126]
Train accuracy: 85.84%
---------------------------------------------------------------------------
Training logistic regression model (Added Bias Term; step_size=0.1)
Learned weight vector:
[-411.5745, 23.8918, -9.2874, 20.1712, 23.1029, 2.9321, 13.0412, 10.4698, 33.0279]
Train accuracy: 95.28%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=0.01)
Learned weight vector:
[-4.7255, 16.5512, 3.9401, 6.4157, -12.4888, -5.9264, 8.8469, -5.9882]
Train accuracy: 84.33%
---------------------------------------------------------------------------
Training logistic regression model (Added Bias Term; step_size=0.01)
Learned weight vector:
[-40.0357, 3.0216, -0.2534, 2.4309, 2.4355, 0.8877, 1.8328, 1.5013, 3.2737]
Train accuracy: 97.0%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=0.0001)
Learned weight vector:
[-0.2464, 0.8677, 0.2008, 0.2785, -0.6761, -0.3325, 0.4337, -0.3499]
Train accuracy: 86.27%
---------------------------------------------------------------------------
Training logistic regression model (Added Bias Term; step_size=0.0001)
Learned weight vector:
[-3.4144, 0.08, 0.4192, 0.2177, 0.2745, -0.2522, 0.0561, 0.2724, -0.0528]
Train accuracy: 96.35%
--------------------------------------------------------------------------------------
Training logistic regression model (No Bias Term; step_size=1e-05)
Learned weight vector:
[-0.2371, 0.4447, 0.2724, 0.1993, -0.3705, -0.2156, 0.2652, -0.212]
```

```
Train accuracy: 85.62%
--------------------------------------------------------------------------------
Training logistic regression model (Added Bias Term; step_size=1e-05)
Learned weight vector:
[-0.7407, -0.17, 0.4121, 0.2636, 0.2022, -0.3106, -0.1542, 0.2571, -0.1773]
Train accuracy: 90.77%
-------------------------------------------------------------------------------------------------
```

Yes, there is a significantly meaningful difference in both the weight vector and the accuracy. Adding the bias term allows for a shift in the line which can better accommodate for a "best fit" with the ability to not go through the origin.

2.2.2 Q6 Learning Rates / Step Sizes. [2pt]

We do observe a downward curve from each plot, and it is safe to say that the negative log-likelihood will continue to drop for some learning rates such as 0.01 and 0.00001 however for larger learning rates, it seems to have converged due to the large learning rate, it is unable to easily shift towards the minima without a smaller learning rate. However, increasing max_iter also increases the computational power needed, thus we have to find a trade-off between trying out different step_sizes of smaller intervals or running for more iterations to obtain globa minima.

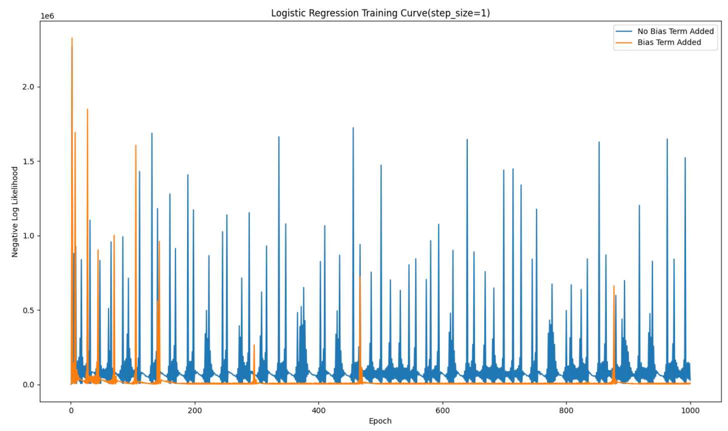

Figure 1: Logistic Regression Training Curve for step size = 1
We observe that a step size or learning rate of 1 is too large and it would constantly jump over the globa minima. this is why we see lines going up and then down when it goes from a low value to large, we observe it leaving the minima due to a large step, which bounces the loss to the other side of the minima, which we observe it as another peak of -LL($/theta$). The same applies to both bias and non-bias as the bias only shifts the plot to not go through the origin, but we also observe that it follows another path due to this change in the graph.

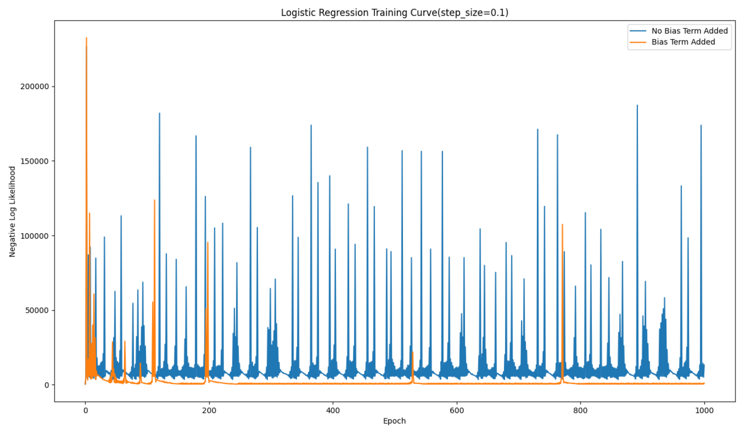

Figure 2: Logistic Regression Training Curve for step size = 0.1
This is the same as a step_size of 1, bouncing over the minima.

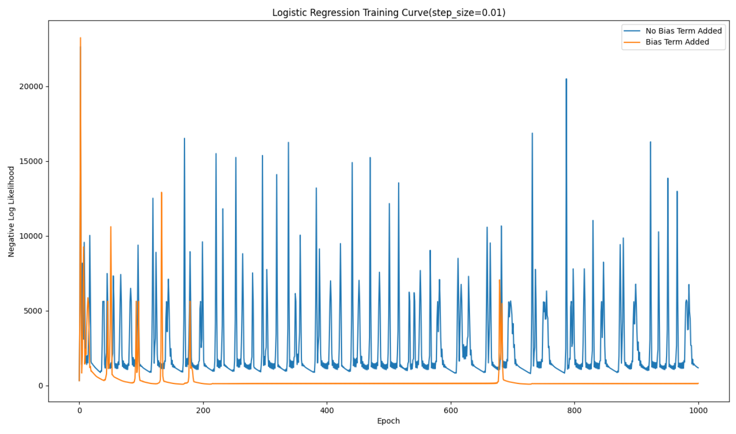

Figure 3: Logistic Regression Training Curve for step size = 0.01
This is the same as a step_size of 0.1, but we observe less bouncing over the minima.

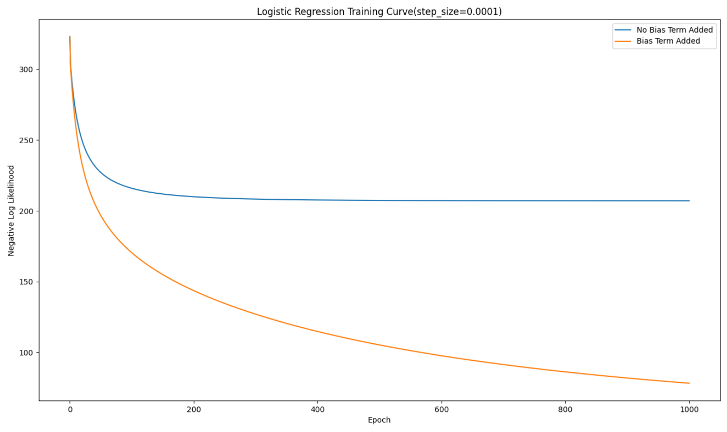

Figure 4: Logistic Regression Training Curve for step size = 0.0001
This is around the point where we have a step_size that is just right so that it keeps approaching the global minima. We observe that without the bias term, we are unable to reach a low global minima like the biased plot this is due to a modeling error. It seems like blue has converged and orange has yet to converge and with higher iterations, it will continue to reach a lower -LL.

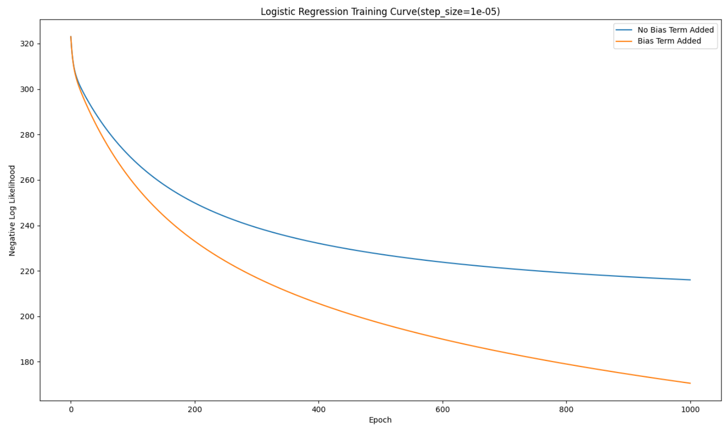

Figure 5: Logistic Regression Training Curve for step size = 0.00001
However, we see that such a small learning rate and running for only 1000 iterations will not allow either biased or unbiased to reach a minima yet. With more iterations, this could result in a better than 0.0001 plot but we will definitely need more iterations.

2.2.3 Q7 Evaluating Cross Validation [2pt]

I had 3 attempts, my first attempt had a step of 0.00001 and a public score of 0.92241 and on k-fold we observe that it is around the lowest mean +- the standard deviation which is around k = 2. It is also within the mean +- standard deviation for k-fold of 10, 20, 50.
My second attempt had an issue where I took the weight of the best k-fold of step size 0.36 so it is not a valid submission because I only trained it on half of a set and I took the best fold.
In my third attempt, I had a public score of 0.94827, this is by using values I observed did well on training data testing all values given by np.linspace(0.005,0.00005, 100), where the best mean resulted in step size = 0.0029 with a mean of 97.22% and a standard deviation of 0.8376%. we observe that this is nowhere near the public score I got, but it had low deviation with high accuracy thus making it a good model. It is hard to notice any trend except for the best mean and deviation are usually around folds of 3,4,5,10, I ran my final hyperparameter search on only k-folds of 5 and 10 to save time.

2.3 Playing with Logistic Regression on This Dataset

3 Debriefing (required in your report)

3.1 Approximately how many hours did you spend on this assignment?

finished this in 2 days, I'd say roughly 40 hours including computational time of running while I sleep.

3.2 Would you rate it as easy, moderate, or difficult?

The assignment was pretty straightforward and easy except for predicting y because there wasn't any skeleton code that I could use. I struggled with the numerical instability with the negative log-likelihood, even now when I'm submitting it, I have only addressed half of the issue. so, overall it's moderate.

3.3 Did you work on it mostly alone or did you discuss the problems with others?

I worked on it alone. Referencing lecture slides.

3.4 How deeply do you feel you understand the material it covers (0%–100%)?

65%

3.5 Any other comments?

Nope