# Homework 2

Tuesday, October 29, 2024 1:30 PM

## Part 1) Linear Regression and Precision/Recall

### Problem 1)

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

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

$\text{argmax} \prod_{i=1}^{N} P(y_i|x_i, w) = \text{argmin} \sum_{i=1}^{N} (y_i - w^T x_i)^2$

$= \prod_{i=1}^{N} P(y_i|x_i, w)$

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

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

we know that $\sum_{i=1}^{N} |y_i - w^T x_i|$ will increase proportionaly but when multiplyed by $-\frac{1}{b}$ it will decrease. when having this as the exponent of $e$ this still stays the same. This means that if we fix b, the likelihood is minimized when $\sum_{i=1}^{N} |y_i - w^T x_i|$ is also minimized. This means that w also minimizes $\sum_{i=1}^{N} |y_i - w^T x_i|$

### Problem 2)

$\text{Recall} = \frac{\#\text{True Positives}}{\#\text{True Positives} + \#\text{False Negatives}}$

$\text{Precission} = \frac{\#\text{True Positives}}{\#\text{True Positives} + \#\text{False Positives}}$

| y | P(y\|x) | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 |
|---|---|---|---|---|---|---|---|
| 0 | 0.1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0.1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0.25 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0.25 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0.3 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0.33 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0.4 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0.52 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0.55 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0.7 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0.8 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0.85 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0.9 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0.9 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0.95 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1.0 | 1 | 1 | 1 | 1 | 1 | 0 |

$t = 0$

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

$\text{Precision} = \frac{8}{8+8} = 0.5$

$t = 0.2$

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

$\text{Precision} = \frac{8}{8+6} = 0.5714$

$t = 0.4$

$\text{Recall} = \frac{6}{6+2} = 0.75$

$\text{Precision} = \frac{6}{6+3} = 0.6\overline{6}$

$t = 0.6$

$\text{Recall} = \frac{6}{6+2} = 0.75$

$\text{Precision} = \frac{6}{6+1} = 0.8571$

$t = 0.8$

$\text{Recall} = \frac{4}{4+4} = 0.5$

$\text{Precision} = \frac{4}{4+1} = 0.8$

$t = 1$

$\text{Recall} = \frac{0}{0+8} = 0$

$\text{Precision} \frac{0}{0+0} = ?$ or Devide by zero

Question 4)

```
2024-10-29 19:41:15 INFO     Training logistic regression model (No Bias Term)
2024-10-29 19:41:19 INFO     Learned weight vector: [-0.2464, 0.8677, 0.2008, 0.2785, -0.6761, -0.3325, 0.4337, -0.3499]
2024-10-29 19:41:19 INFO     Train accuracy: 86.27%
2024-10-29 19:41:19 INFO
```

Question 5)

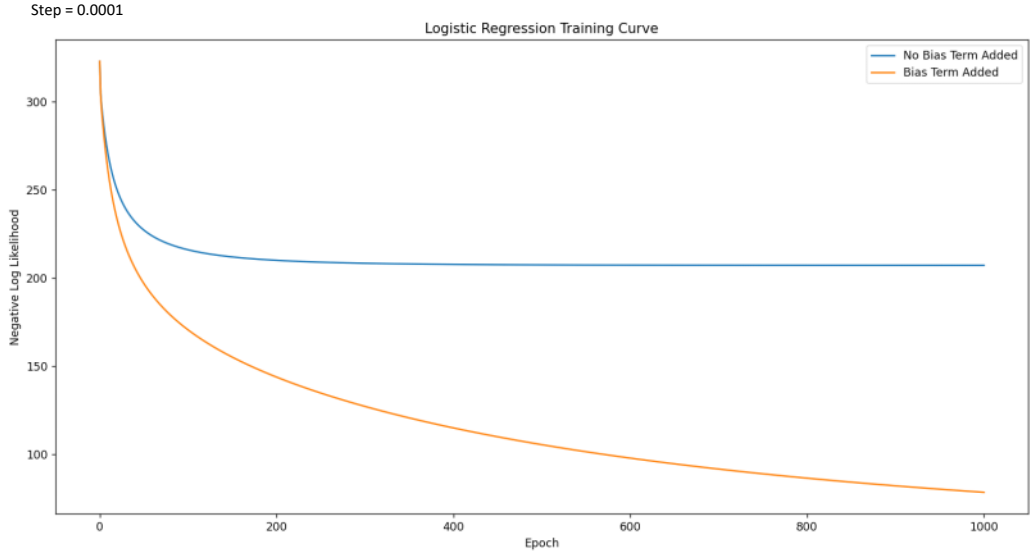
```
2024-10-29 19:41:15 INFO     Training logistic regression model (No Bias Term)
2024-10-29 19:41:19 INFO     Learned weight vector: [-0.2464, 0.8677, 0.2008, 0.2785, -0.6761, -0.3325, 0.4337, -0.3499]
2024-10-29 19:41:19 INFO     Train accuracy: 86.27%
2024-10-29 19:41:19 INFO
----------------------------------------------------------------------

2024-10-29 19:41:19 INFO     Training logistic regression model (Added Bias Term)
2024-10-29 19:41:24 INFO     Learned weight vector: [-3.4144, 0.08, 0.4192, 0.2177, 0.2745, -0.2522, 0.0561, 0.2724, -0.0528]
2024-10-29 19:41:24 INFO     Train accuracy: 96.35%
```

It seems like it makes somewhat of a difference, we can see this in the train accuracy within the training accuracy and also it reflects with the learned weight vector.

Question 6)

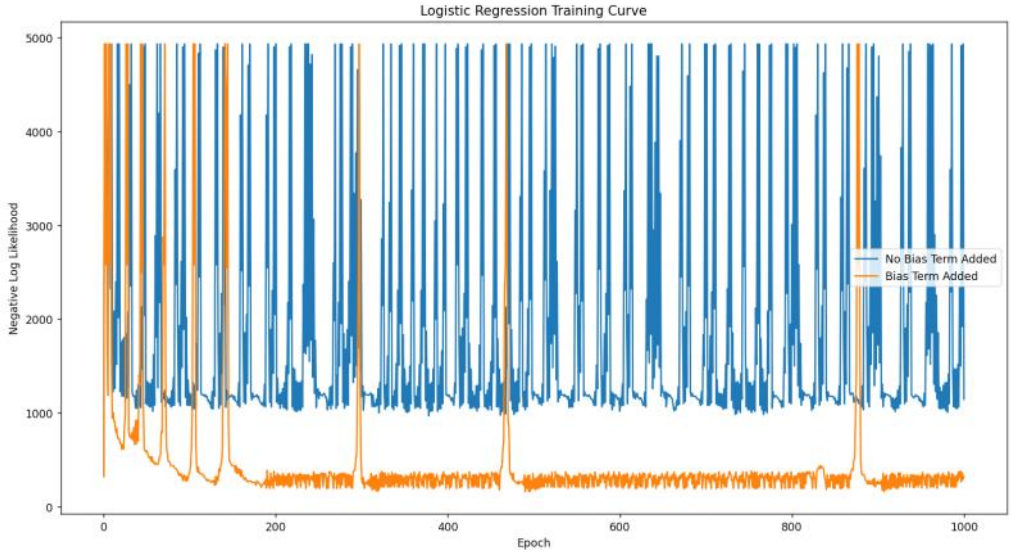
Step = 0.0001




```
2024-10-29 19:41:15 INFO     Training logistic regression model (No Bias Term)
2024-10-29 19:41:19 INFO     Learned weight vector: [-0.2464, 0.8677, 0.2008, 0.2785, -0.6761, -0.3325, 0.4337, -0.3499]
2024-10-29 19:41:19 INFO     Train accuracy: 86.27%
2024-10-29 19:41:19 INFO
----------------------------------------------------------------------

2024-10-29 19:41:19 INFO     Training logistic regression model (Added Bias Term)
2024-10-29 19:41:24 INFO     Learned weight vector: [-3.4144, 0.08, 0.4192, 0.2177, 0.2745, -0.2522, 0.0561, 0.2724, -0.0528]
2024-10-29 19:41:24 INFO     Train accuracy: 96.35%
```

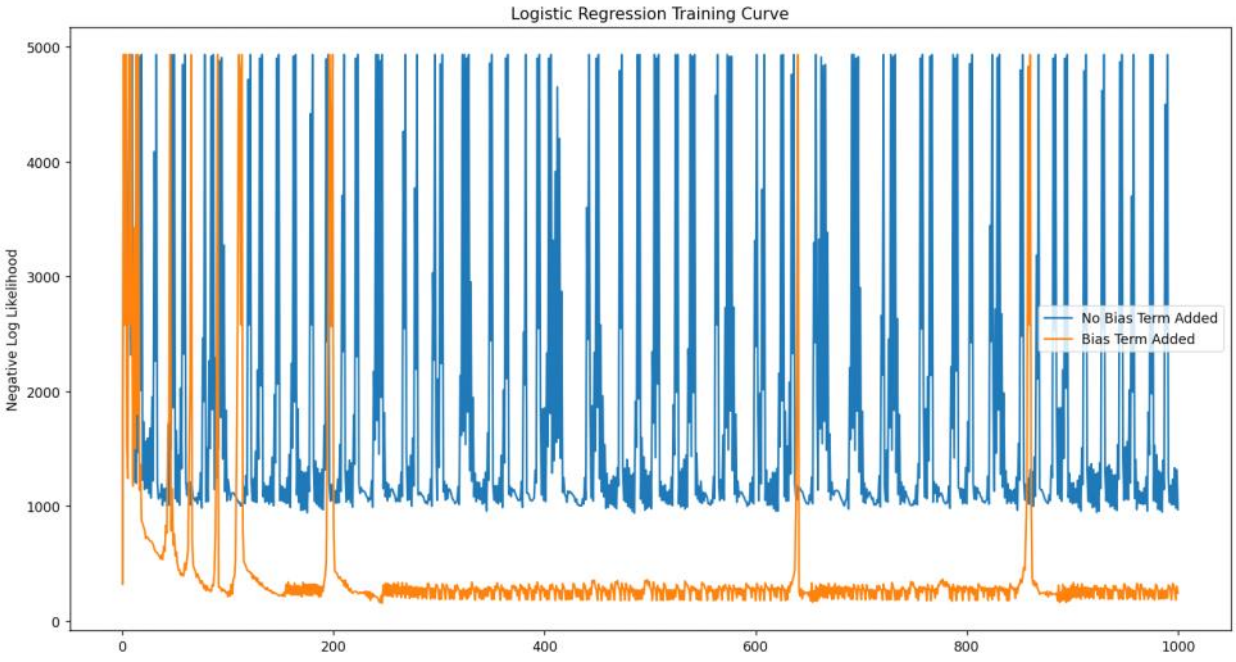
Step = 1




```
2024-10-29 20:30:22 INFO     Training logistic regression model (No Bias Term)
2024-10-29 20:30:27 INFO     Learned weight vector: [-783.2642, 1627.7124, 457.0358, 600.1099, -1331.4154, -703.5553, 806.0741, -606.114]
2024-10-29 20:30:27 INFO     Learned weight vector: [-783.2642, 1627.7124, 457.0358, 600.1099, -1331.4154, -703.5553, 806.0741, -606.114]
2024-10-29 20:30:27 INFO     Train accuracy: 84.76%
2024-10-29 20:30:27 INFO
----------------------------------------------------------------------

2024-10-29 20:30:27 INFO     Training logistic regression model (Added Bias Term)
2024-10-29 20:30:32 INFO     Learned weight vector: [-4327.7176, 255.1665, -85.8565, 205.8328, 238.9805, 36.3716, 138.0318, 125.7803, 342.294]
2024-10-29 20:30:32 INFO     Train accuracy: 95.71%
```

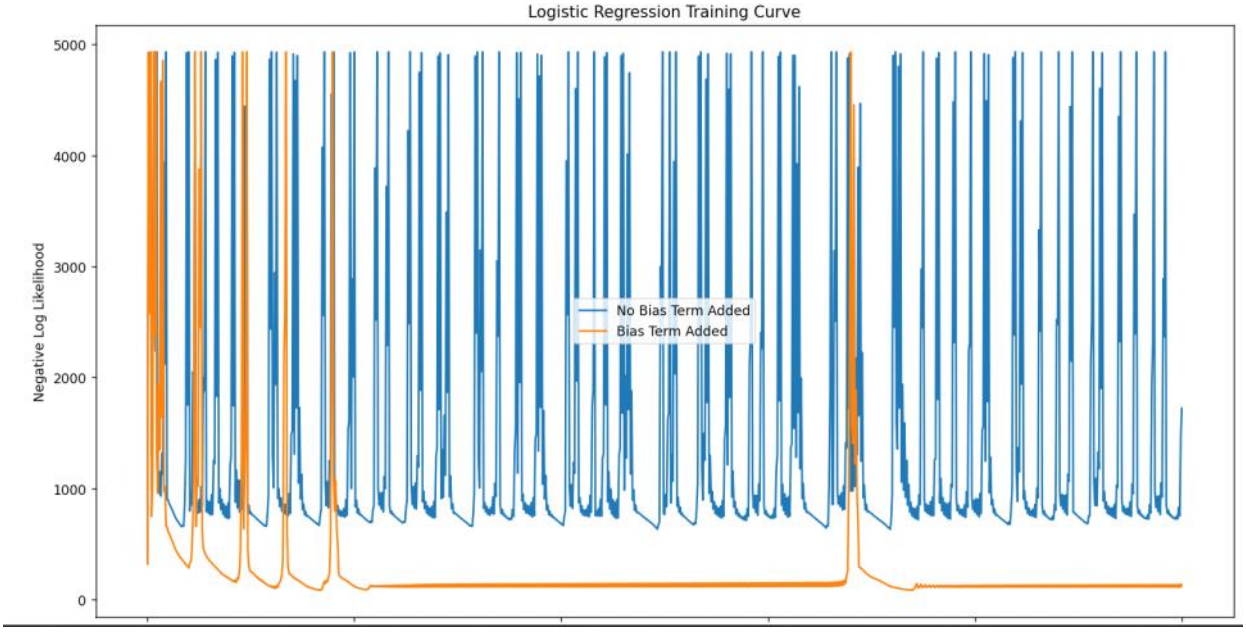
Step = 0.1




```
2024-10-29 23:04:22 INFO     Training logistic regression model (No Bias Term)
2024-10-29 23:04:27 INFO     Learned weight vector: [-44.0695, 137.2765, 22.4725, 48.9776, -113.9293, -57.7692, 71.3121, -49.2553]
2024-10-29 23:04:27 INFO     Train accuracy: 86.48%
2024-10-29 23:04:27 INFO
----------------------------------------------------------------------

2024-10-29 23:04:27 INFO     Training logistic regression model (Added Bias Term)
2024-10-29 23:04:33 INFO     Learned weight vector: [-416.7573, 32.3044, -4.1042, 26.2385, 28.0513, 8.978, 18.4862, 16.1763, 35.5794]
2024-10-29 23:04:33 INFO     Train accuracy: 96.14%
```

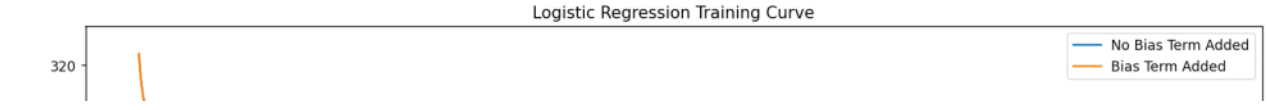
Step = 0.01

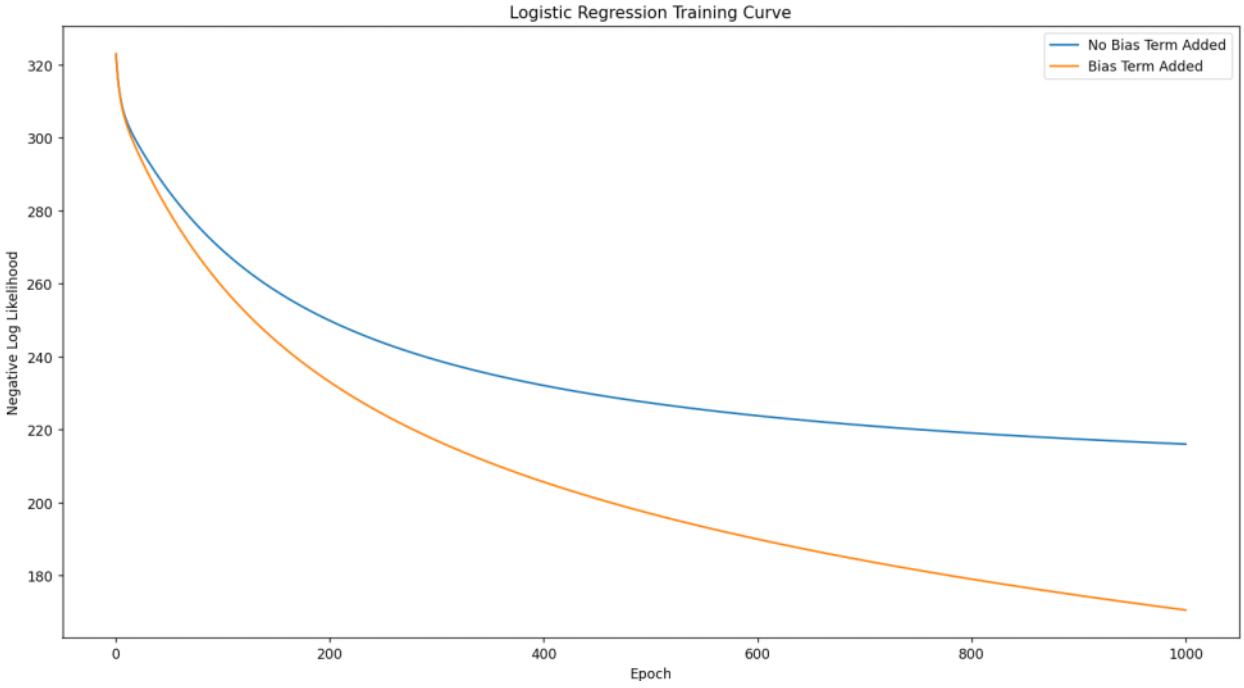



```
2024-10-29 23:07:36 INFO     Training logistic regression model (No Bias Term)
2024-10-29 23:07:41 INFO     Learned weight vector: [-5.612, 11.6849, 0.1929, 3.4828, -12.1849, -6.4571, 5.6635, -6.1002]
2024-10-29 23:07:41 INFO     Train accuracy: 74.25%
2024-10-29 23:07:41 INFO
----------------------------------------------------------------------

2024-10-29 23:07:41 INFO     Training logistic regression model (Added Bias Term)
2024-10-29 23:07:46 INFO     Learned weight vector: [-41.3642, 3.1533, -0.3764, 2.5165, 2.5392, 0.8474, 1.8014, 1.507, 3.446]
2024-10-29 23:07:46 INFO     Train accuracy: 97.0%
```

Step = 0.00001






```
2024-10-29 23:10:46 INFO     Training logistic regression model (No Bias Term)
2024-10-29 23:10:51 INFO     Learned weight vector: [-0.2371, 0.4447, 0.2724, 0.1993, -0.3705, -0.2156, 0.2652, -0.212]
2024-10-29 23:10:51 INFO     Train accuracy: 85.62%
2024-10-29 23:10:51 INFO
----------------------------------------------------------------------

2024-10-29 23:10:51 INFO     Training logistic regression model (Added Bias Term)
2024-10-29 23:10:56 INFO     Learned weight vector: [-0.7407, -0.17, 0.4121, 0.2636, 0.2022, -0.3106, -0.1542, 0.2571, -0.1773]
2024-10-29 23:10:56 INFO     Train accuracy: 90.77%
```

It seems that when the step size is very high we seem to have more erratic behavior, this is also reflected in the rain accuracy when looking at the step sizes. With high step sizes we see lower train accuracy then that of lower step sizes on average. This is for a few reasons there is for a few reasons but the main one is that when having a large step size we will be stepping over the intended direction of w-> like we learned in class meaning that the line will be moving back in forth with a w in the negative and positive but when stepping it get further away. We also see this line fallowing a log line somewhat.

Question 7)

```
2024-10-30 21:51:15 INFO     Running cross-fold validation for bias case:
2024-10-30 21:51:20 INFO     2-fold Cross Val Accuracy -- Mean (stdev): 95.49% (1.502%)
2024-10-30 21:51:30 INFO     3-fold Cross Val Accuracy -- Mean (stdev): 96.13% (1.419%)
2024-10-30 21:51:46 INFO     4-fold Cross Val Accuracy -- Mean (stdev): 96.36% (1.1%)
2024-10-30 21:52:07 INFO     5-fold Cross Val Accuracy -- Mean (stdev): 96.1% (2.095%)
2024-10-30 21:52:53 INFO     10-fold Cross Val Accuracy -- Mean (stdev): 96.28% (3.886%)
2024-10-30 21:54:32 INFO     20-fold Cross Val Accuracy -- Mean (stdev): 96.25% (3.931%)
2024-10-30 21:59:14 INFO     50-fold Cross Val Accuracy -- Mean (stdev): 96.03% (6.582%)
```

In this one it seems that the cross Val accuracy between is less comparable between running 2 fold cross-val and 50 fold cross val. Additionally we see that as we go up in the fold cross we can see a incrase to the stand deviation %. These 2 facts combined means that our data could in general not be very good at generalizing beyond a certain data characteristic although 6% is on the lower side. It seems with this problem that there is a fewer number of hyperparameters we can touch to increase or decrease the score.

Question 8)

In the end I went with a step size of 0.00021 due to my computer being a brick and taking 20 mins to run the code a single time. The number of limiters stayed that same. Other then that nothing really was touched to the code.

## Debriefing

1 - I spent like 6 hours on this one it wasn't too bad.
2 - Easy/moderate
3 - All alone
4 - 95% of the topics
5 - Thank you for the class so far It has been a great learning experience.