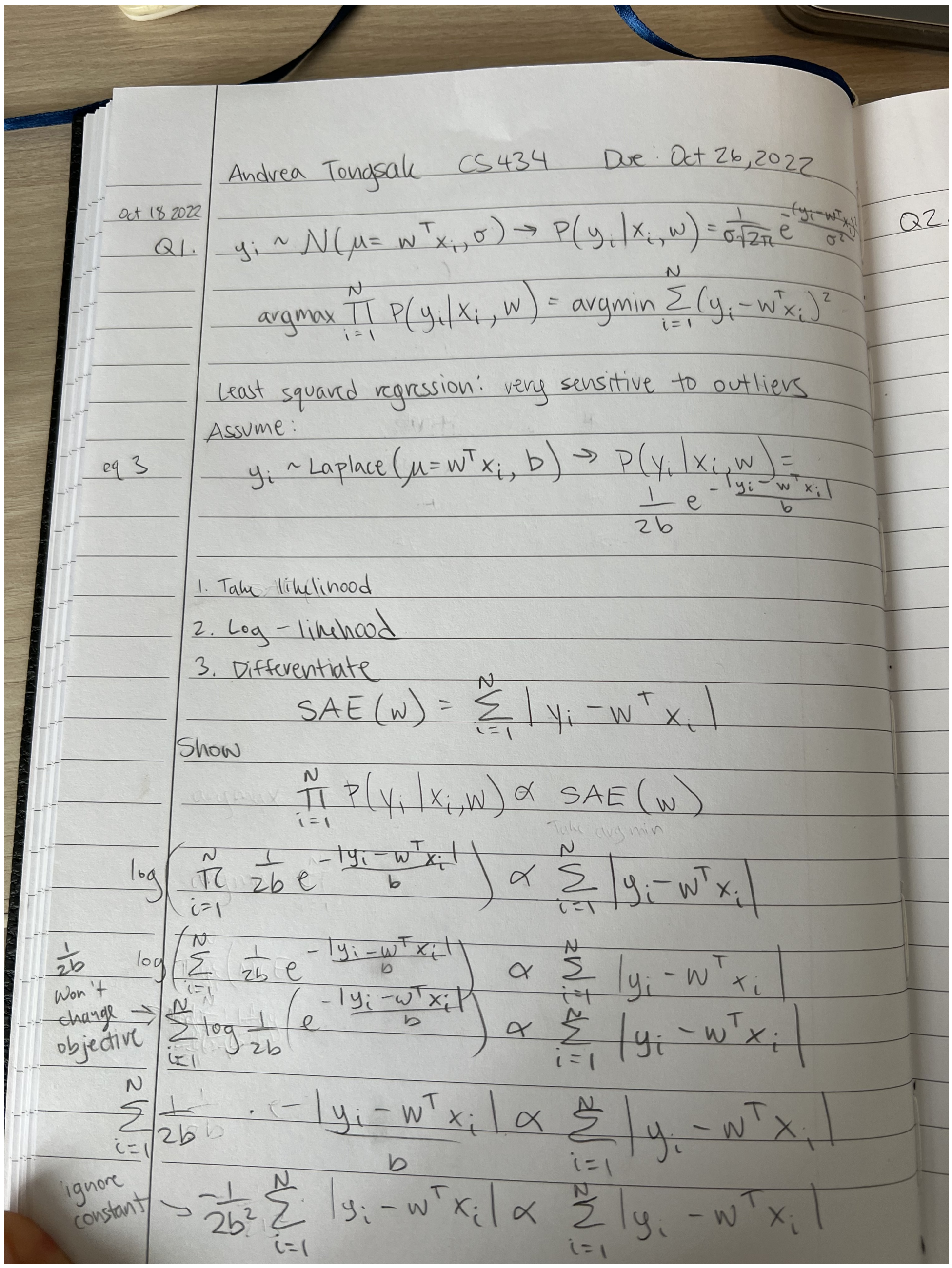
CS434: HW2

Andrea Tongsak

October 25, 2022

1 Written Exercises: Linear Regression and Precision/Recall

1. Math below in image.



Andrea Tongsak CS434 Due: Oct 26, 2022

Oct 18 2022

Q1. $y_i \sim \mathcal{N}(\mu = w^T x_i, \sigma) \rightarrow P(y_i | x_i, w) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y_i - w^T x_i)^2}{\sigma^2}}$

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

Least squared regression: very sensitive to outliers

Assume:

eq 3 $\quad y_i \sim \text{Laplace}(\mu = w^T x_i, b) \rightarrow P(y_i | x_i, w) = \frac{1}{2b} e^{-\frac{|y_i - w^T x_i|}{b}}$

1. Take likelihood
2. Log-likelihood
3. Differentiate

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

Show

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

Take argmin

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

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

$\frac{1}{2b}$ won't change objective $\rightarrow$

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

$$\sum_{i=1}^{N} \frac{1}{2b} \cdot -\frac{|y_i - w^T x_i|}{b} \propto \sum_{i=1}^{N} |y_i - w^T x_i|$$

ignore constant $\rightarrow$

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

Q2.

Q2.

true negative: where $P(y|x)$ is less $<= t$, and $y = 0$

false positive: any entry where $P(y|x)$ is greater than the threshold t, and y is 0.

false negative: where $P(y|x)$ is less $<= t$, and $y = 1$

true positive: where $P(y|x) > t$ is the positive class and $y = 1$

Compute Recall and Precision at specific t

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

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

$t = 0$

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

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

$t = 0.2$

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

$$\text{precision} = \frac{8}{8+6} = \frac{8}{14} = 0.5714$$

$t = 0.4$

$$\text{recall} = \frac{6}{6+2} = \frac{6}{8} = \frac{3}{4} = 0.75$$

$$\text{precision} = \frac{6}{6+3} = \frac{6}{9} = \frac{2}{3}$$

$t = 0.6$

recall $= \dfrac{6}{6+2} = \dfrac{6}{8}$

precision $= \dfrac{6}{6+1} = \dfrac{6}{7}$

$t = 0.8$

recall $= \dfrac{4}{4+4} = \dfrac{4}{8} = \dfrac{1}{2}$

precision $= \dfrac{4}{4+1} = \dfrac{4}{5}$

$t = 1$

recall $= \dfrac{0}{0+8} = 0$

precision $= \dfrac{0}{0+0} =$ undefined ?

2 Implementating Logistic Regression for Tumor Diagnosis

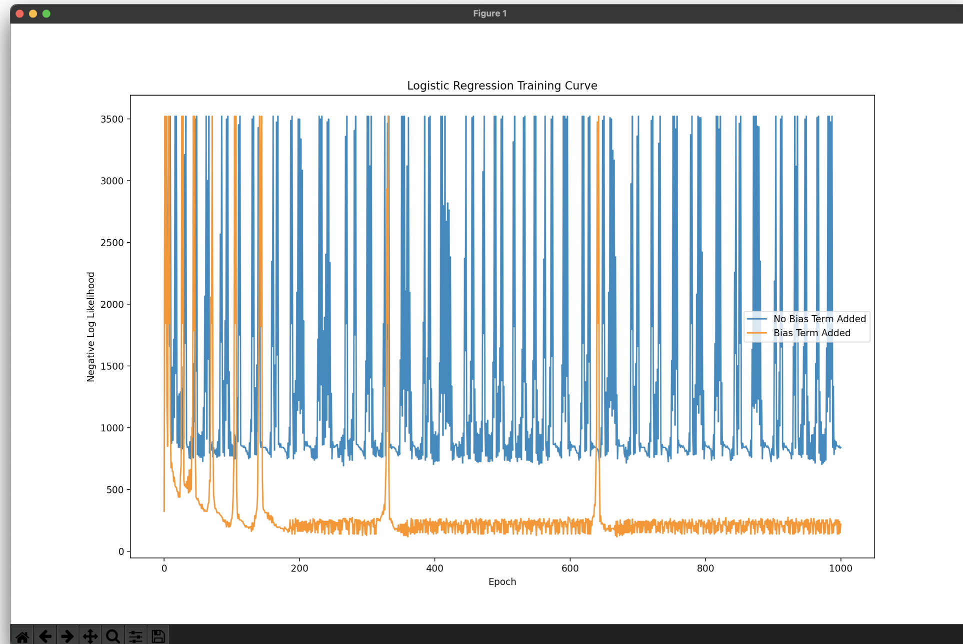
5. logistic and calculateNegativeLogLikelihood implemented in .py code.

   step size is 0.00001
   No Bias Term
   weight vector: [-0.2464, 0.8677, 0.2008, 0.2785, -0.6761, -0.3325, 0.4337, -0.3499]
   training accuracy: 86.27%
   Added Bias Term
   Learned weight vector: [-3.4144, 0.08, 0.4192, 0.2177, 0.2745, -0.2522, 0.0561, 0.2724, -0.0528]
   training accuracy: 96.35%

   Yes, it did make a significant difference, adding bias increased the training accuracy.

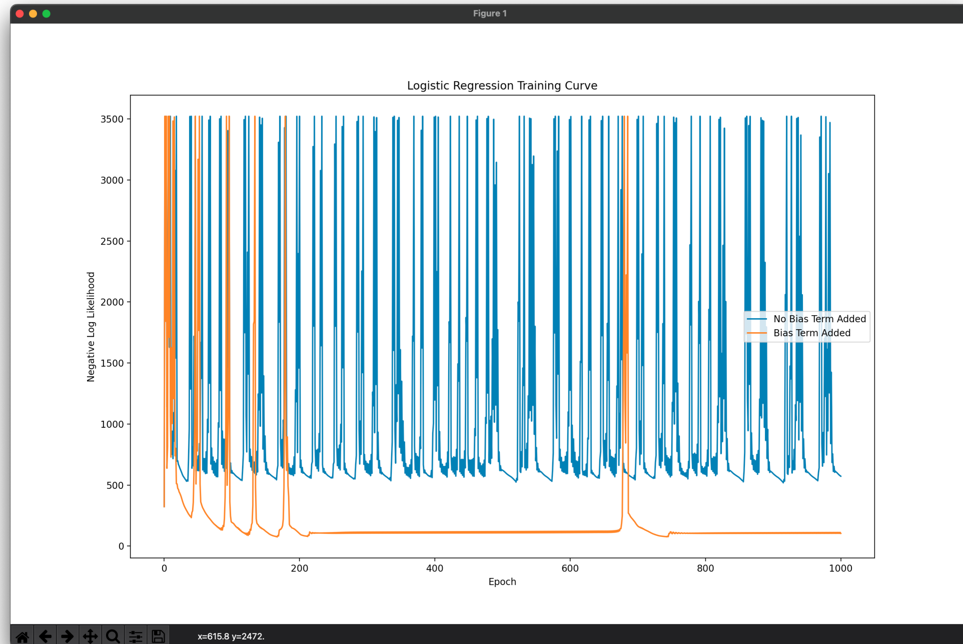
6. Step size: 1
   Training logistic regression model (No Bias Term) Learned weight vector: [-428.358, 1497.0248, 338.2895, 590.5005, -1150.9271, -544.9648, 802.157, -543.3729]
   Train accuracy: 84.33%

   Training logistic regression model (Added Bias Term)
   Learned weight vector: [-4044.4709, 248.0085, -92.2857, 199.2664, 224.8464, 28.6599, 137.0, 107.3912, 321.708]
   Train accuracy: 95.92%



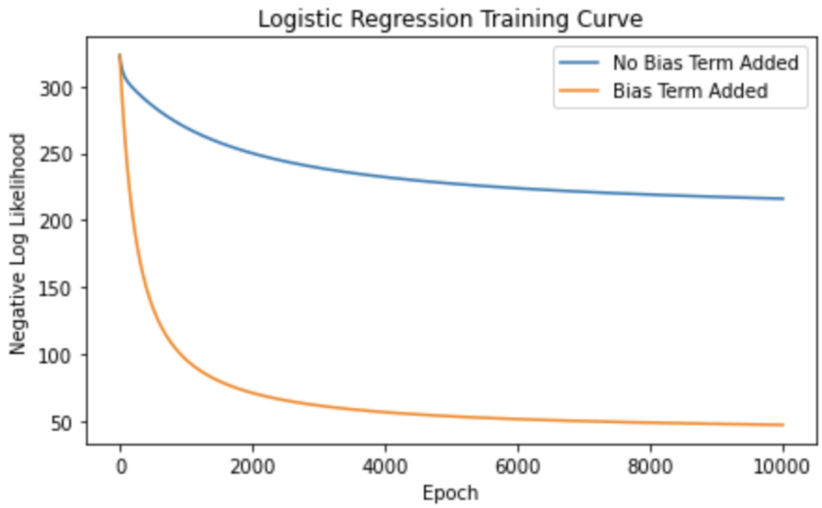
Step size: 0.1
Training logistic regression model (No Bias Term)
Learned weight vector: [-55.2898, 157.863, 39.656, 60.5772, -121.6459, -61.4181, 80.5727, -57.1643]
Train accuracy: 85.84%

Training logistic regression model (Added Bias Term)
Learned weight vector: [-403.9028, 23.6033, -8.7606, 19.6447, 22.0517, 3.9549, 13.6063, 10.4134, 32.2611]
Train accuracy: 95.49%



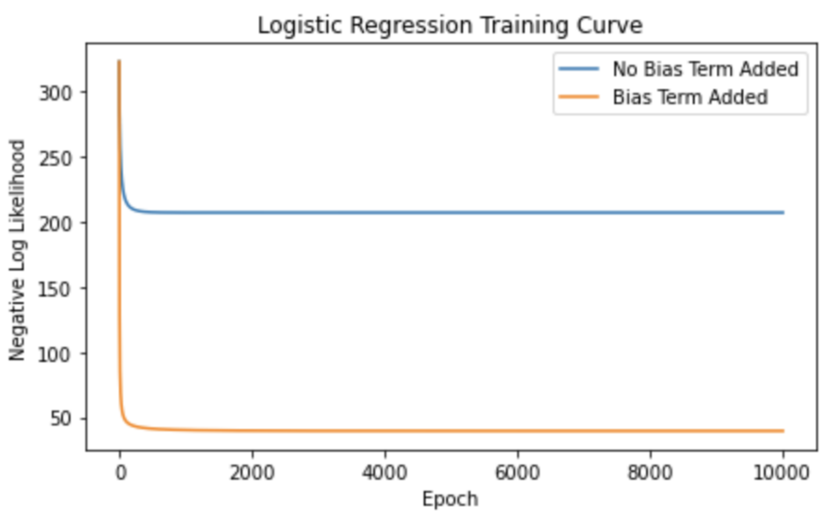
Step size: 0.01
Training logistic regression model (No Bias Term)
Learned weight vector: [-3.7844, 14.4811, 3.1256, 5.811, -11.3729, -5.3475, 8.1695, -5.574]
Train accuracy: 84.55%

Training logistic regression model (Added Bias Term)
Learned weight vector: [-41.87, 2.5453, -0.8747, 2.0604, 2.22, 0.3034, 1.3241, 1.0198, 3.3427]
Train accuracy: 95.28%



Step size: 0.00001
Training logistic regression model (No Bias Term) 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)
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%



I noticed that the step size get smaller, the bias and no bias lines begin diverging significantly. Going from 0.0001 and 0.00001 can show where the lines start to diverge.

7. Evaluating Cross Validation: Cross validation isn't a good indicator of how a program may actually perform on the Kaggle leaderboard. The standard deviation will increase as k increases.

8. For my final submission, I ended up adding an epislon value of 0.00001 to the trainLogistic function, however, I could have improved it if I had more patience and time!

3 Debriefing

1 7 hours

2 easier-moderate

3 some discussion, but mainly to confirm understanding.

4 80%

5 Just will spend time on studying the midterm after this is turned in.