# HW1

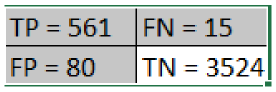
mrodr120

October 2019

## Problem 3

### Part 3.1-Logistic Regression

1. We are learning **1296** parameters
2. The L2-norm of our weight vector is **34.85**
3. Our train set accuracy is **97.72%**
4. Train set Precision, Recall, F1 for **positive class**.
    (a) Precision = 87.52%
    (b) Recall = 97.39%
    (c) F1 = 92.19%
5. Train set Precision, Recall, F1 for **negative class**.
    (a) Precision = 99.58%
    (b) Recall = 97.78%
    (c) F1 = 98.67%
6. Confusion Matrix for **train set**



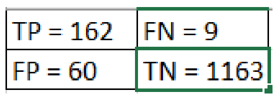
| | |
|---|---|
| TP = 561 | FN = 15 |
| FP = 80 | TN = 3524 |

Figure 1: Confusion matrix for train set

7. Test set accuracy is **95.05%**
8. Test set Precision, Recall, F1 for **positive & negative class**.
    (a) Positive class
        i. Precision = 72.97%

        ii. Recall = 94.74%
        iii. F1 = 82.44%
    (b) Negative class
        i. Precision = 99.23%
        ii. Recall = 95.09%
        iii. F1 = 97.12%

9. Confusion matrix for **test set**



| | |
|---|---|
| TP = 162 | FN = 9 |
| FP = 60 | TN = 1163 |

Figure 2: Confusion matrix for test set

10. Plot of log-likelihood per iteration

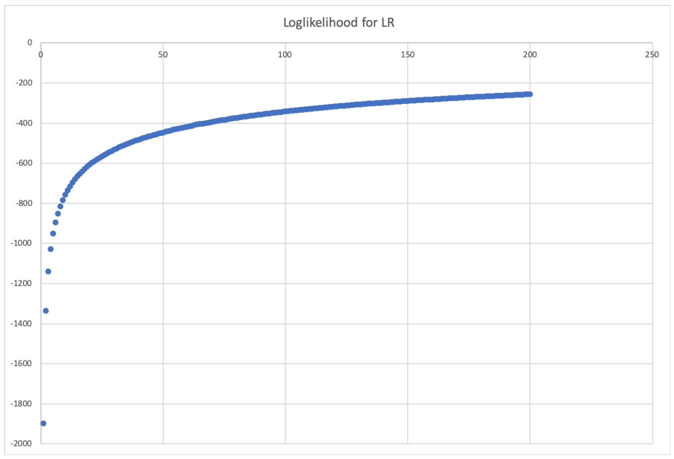


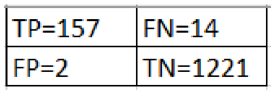
Figure 3: Plot of log-likelihood per iteration

11. As we repeat computing the log-likelihood each iteration, we can see a logarithmic increase. This makes sense since we are taking the log of the weights. We are reaching a plateau, which means eventually the changes in our log-likelihood will be minimal in which is when we would have found our optimal weights for classifying instances correctly.

12. We could initialize the weight vector to be random weights every time, this will help in case we are in a local minimum every time we run the program since we are starting from the same point every time, when trying to optimize our weights. Or we could change the probability threshold to favor the class that shows up more often in our train set, so that more instances get classified to that class.

## Part 3.2-Logistic Regression With Bias

1. We are learning **1297** parameters
2. Our train set accuracy is **99.69%**; our test set accuracy is **95.05%**
3. Confusion Matrix for **test set**



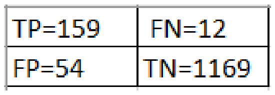
| | |
|---|---|
| TP=157 | FN=14 |
| FP=2 | TN=1221 |

Figure 4: Confusion matrix for test set with bias

4. The accuracy on with a Bias term is more accurate than without in both training and testing. Which means that the bias term is helping with accurately shifting the weights to classify instances correctly. Secondly, the Precision, Recall, and F1 measure are all higher in with a Bias than without in both testing and training. Therefore, we observed that L.R. with a bias is a better performer, by these observations.

## Part 3.3-Logistic Regression with Regularization

1. We are learning **1296** parameters
2. Our train set accuracy is **96.36%**; our test set accuracy is **95.27%**
3. Confusion Matrix for **test set with regularization**



| | |
|---|---|
| TP=159 | FN=12 |
| FP=54 | TN=1169 |

Figure 5: Confusion matrix for test set with regularization

4. The performance for the training set decreases while the performance for the test set increases from normal logistic regression to one with Regularization. Because regularization is to avoid overfitting it makes a more general classification thus mis-classifying more training data but does better when classifying new data.
5. (a) Changes to weight vector for different values of $\lambda$
   (b) if lambda is 0, then the L2-norm of the weight vector is the same in as in 3.1.
   (c) As lambda increases, the norm decreases this is because in extra term when regularizing will grow much large and thus create a smaller weight vector. When taking the norm of that vector, the values will

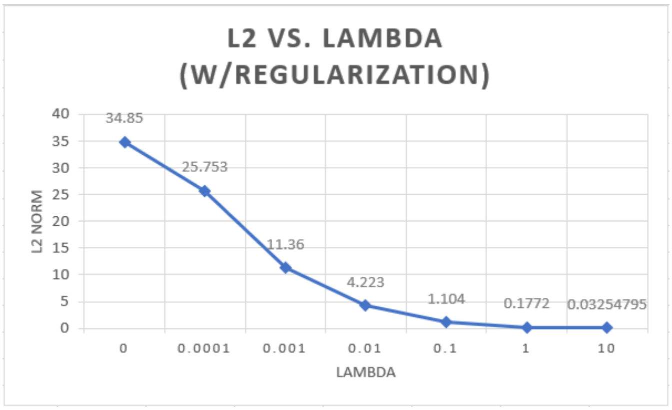


Figure 6: Plot of L2 norm of weight vector vs lambda

be really small. In other words, we are penalizing higher weights a lot and making our weights smaller to help with generalization in the testing phase.

6. (a) Changes to accuracy for different values of $\lambda$

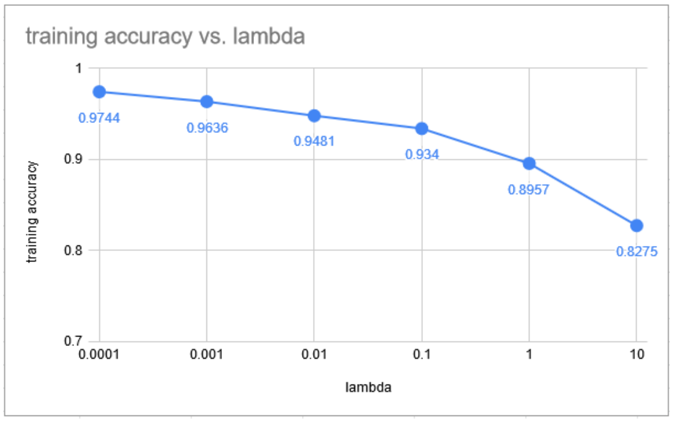


Figure 7: Plot of training accuracy vs lambda

(b) As lambda increases, the accuracy decreases. This is because we are generalizing more as we train on the training set. The weights are dropping for the reason above (6.c), so that in the test set we have better luck with instances we have not yet seen.