Logistic Regression Classifier

Ethan Seefried
CS 542
Colorado State University

1 Introduction

Using the same datasets given in the Naive Bayes assignment, I was tasked with creating a logistic regression classifier, in which could predict the class of a movie. This model, given a lexicon would be capable of training on mini-batches, where groups of documents could be trained at a time. This, along with the other parameters of the learning rate, and amount of epochs were able to be changed, to fine-tune the model.

2 Theory

To train the model, we can use linear algebra, and a gradient descent approach. To begin the gradient descent, we need a matrix x, which contains the documents currently being trained, with a bias layer, as well as a vector y, which contains the classes for each document. It's important to know the shapes for these, and it turns out that x is the shape $(m, |F| + 1)$, where m is the number of documents in the training (mini-batch), and F is the number of features. Notice we add one to the shape, to account for the bias. For the shape of Y, we will see that it will be $(m,)$. With our matrix and vector, we can find the loss for the model, as well as the gradients, which will be used to update the weights. To find the loss we can use

$$L_{CE}\left(\hat{y}, y\right) = -[ylog\hat{y} + (1-y)log(1-\hat{y})], \tag{1}$$

where,

$$\hat{y} = \sigma(x \cdot \theta). \tag{2}$$

For this equation (2), σ is the traditional sigmoid function,

$$S(x) = \frac{1}{1+e^{-x}}, \tag{3}$$

and θ is a vector containing the weights and bias. Using these equations, we can calculate the sum of the gradients as an average gradient for each document in the current mini-batch,

$$\Delta L = \frac{1}{m}\left(x^T \cdot (\hat{y} - y)\right). \tag{4}$$

Finally, we can update the weights using,

$$\theta_{t+1} = \theta_t - \eta \Delta L. \tag{5}$$

For this model, to compute the statistics and analyze the data, we would use the same metrics as the previous assignment; accuracy, precision, recall, and the F1 score [1].

3 Results

To test the model, I modified a python dictionary using multiple methods to select a lexicon that would produce the highest accuracy. The methods I ended up testing were; binary word features, document statistics, in which I counted the words per document, and a lexicon count feature, which is what I ultimately got the best results with. Using accuracy as a measurement, I received a 47.5% accuracy (worse than a coin flip) and a 53% accuracy for the document statistics test and binary word features respectively. I believe these poor results were due to how I was choosing the words, for the binary classifier, with too many words being in both the positive and negative class. For the document length, I expected better results, but after scrolling through the documents, I believed them to be roughly similar lengths for each class. I tried 4 main lexicons, first I tested for punctuation (?, or !), before testing for a lexicon of 11 words that I created and a lexicon of 20 words. After settling on a lexicon count method, I tested multiple lexicons, and began fine-tuning the parameters. I chose to use accuracy as my main testing method, as I believed that it was the most important for the classifier to predict the correct class.

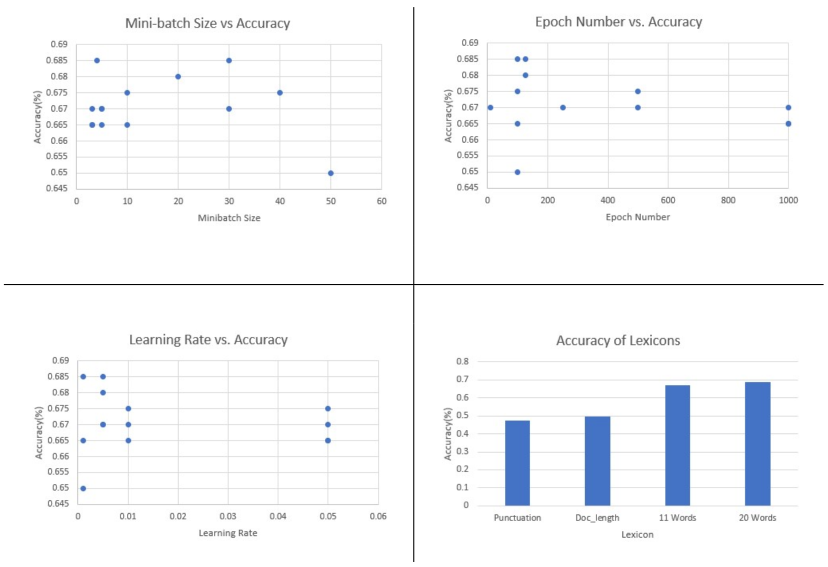

Table 1: Accuracy metrics for hyper-parameters

4 Conclusion

Using python and numpy, I created a logistic regression classifier, to predict the class of a movie review. To train the model, I used linear algebra and performed a gradient descent on documents in mini-batches. To perform gradient descent, the loss of each document was calculated, before taking the derivative and updating weights. This process could be ran for different amount of epochs, with varying learning rates and mini-batch sizes. After training on 5 small documents, the model was tested on 200 documents, of varying lengths. After testing multiple feature selection methods, I came up with a 20 word lexicon, that achieved 68.5% accuracy.

References

[1] Ethan Seefried. Naive bayes classifier, September 2022.