Logistic Regression

Wavelix

1 Requirment

Remove one class of wine samples from the dataset, retaining the other two classes to generate a new dataset. Consider the logistic regression model and cross-entropy loss function to address the binary classification problem of wine.

1. Write code to split the dataset into a training set and a test set in the ratio of (0.7, 0.3).
2. Write codes for Mini-batch update and Stochastic update to train the model.
3. Use the trained model to make predictions on the test set, and write code to evaluate the model's classification performance using Accuracy, recall, precision, and F1 score.
4. Analyze and conclude the difference between perceptron and logistic regression model.

2 Idea

Remove the last class in the dataset. Meanwhile, relabel the first class as 1 and the second as 0. We split the new dataset as follows:

	label 1	label 0	total
training set	41	50	91
test set	18	21	39
total	59	71	130

Now the number of instances in training set is 91, while in test set is 39.

We use the sigmoid function to calculate the output:

$$\sigma(z) = \frac{1}{1+\exp(-z)}$$

$$y = \sigma(z) = \sigma(w^T \bar{x})$$

Define the loss function:

$$\textbf{loss} = -\frac{1}{N}\sum_{i=1}^{N} t^{(i)} \log(y^{(i)} + \epsilon) - \frac{1}{N}\sum_{i=1}^{N}(1 - t^{(i)}) \log(1 - y^{(i)} + \epsilon)$$

Where ϵ is a nominal parameter to avoid math error.

For Mini-batch update, we randomly shuffle the training set and partition it into 9 mini-batches

(**batch size** = 10, ignore the last element in the training set). Then for every batches, we update the weight through the gradient:

$$\bigtriangledown l(W) = -\frac{1}{10} X^T (t - y)$$

For SGD update, we randomly shuffle and pick one sample in the training set for each epoch. The gradient is:

$$\bigtriangledown l(W) = -(x^{(i)})^T (t^{(i)} - y^{(i)})$$

Normalization and standardization are considered.

In the prediction part, we give the output by:

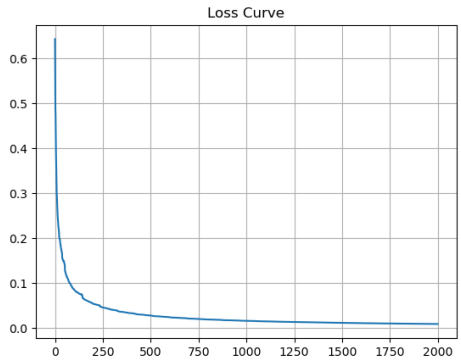
$$y = \begin{cases} 1 & \sigma(z) > 0.5 \\ 0 & \sigma(z) \leq 0.5 \end{cases}$$

3 Result

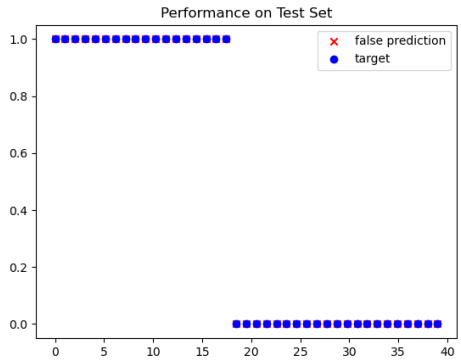
Here gives some examples of the results. More results can be found in the appendix.

3.1 SGD with standardization

epochs	learning rate	Accuracy	Recall	Precision	F1
2000	1e-1	1.00	1.00	1.00	1.00



(a) loss curve



(b) prediction

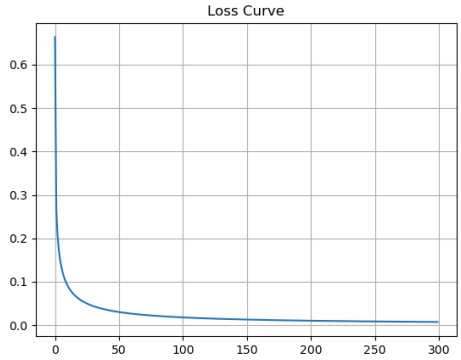
SGD updates faster than MBGD, while each steps are vulnerable to noise interference.

When Normalization or standardization is not applied, system is easy to diverge, so a smaller learning rate is needed, which leads to larger epochs. In this condition, the performance is not good.

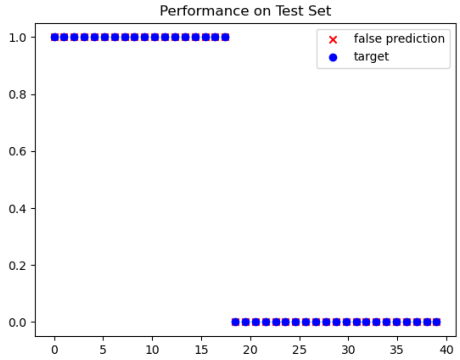
Normalization and standardization make the training easier to converge. More specificly, standardization performs better than normalization. Once normalization or standardization is applied, a larger learning rate is needed to accelerate the training. In this condition, the performance is better, and the model can always make correct prediction.

3.2 MBGD with standardization

epochs	learning rate	Accuracy	Recall	Precision	F1
300	1e-1	1.00	1.00	1.00	1.00



(c) loss curve



(d) prediction

The loss function of MBGD goes smoothly，and each step of MBGD can more accurately point to the optimal solution.

The effects of normalization and standardization is similar to SGD, and it should be noted that the standardization seems always perform better than normalization. Specificly, it is easier to find a set of hyper parameters to make the training converge fast and correct using standardization.

4 Analysis

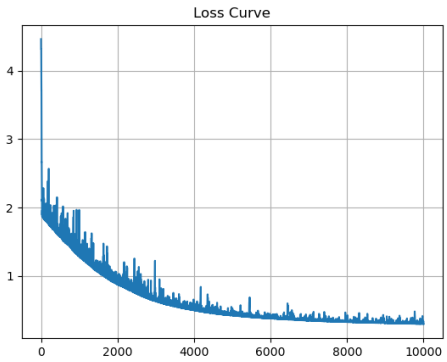
1. The perceptron uses a step function to produce binary outputs, while logistic regression uses a sigmoid function to produce continuous output values between 0 and 1.

2. The perceptron does not provide probabilistic information, and is simply based on a threshold. In other words, it only considers whether the prediction is right or wrong, while ignores how right or wrong it is. Logistic regression, however, provides a probability score for each input.

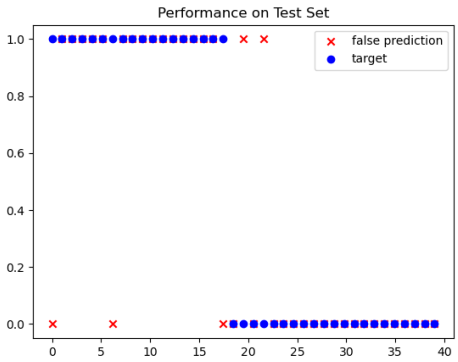
3. The perceptron only updades weights when the current prediction is wrong. Logistic regression, however, updades weights by looking for the prediction error, gradually refining the weights.

Though both of them have linear decision boundary, logistic regression tends to perform better in more complex cases. For wine.data, logistic regression have better prediction and is easier to converge, since in the last experiment, I failed to find a solution to make prediction strictly properly using the perceptron.

5 Appendix

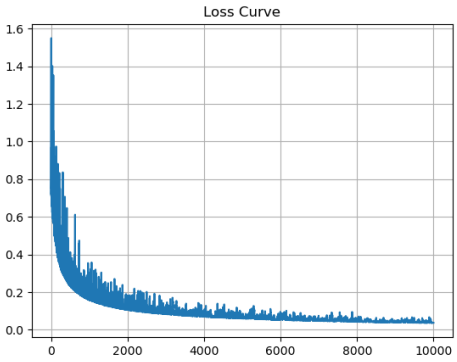


(e) loss curve

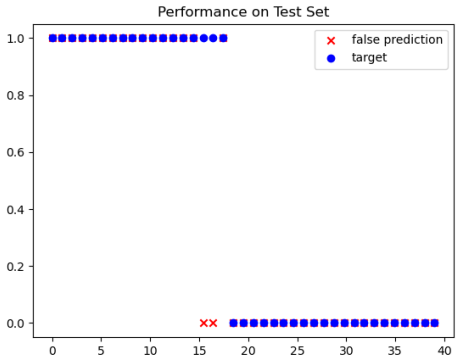


(f) prediction

图 1: SGD without normalization. lr=1e-6 Accuracy:0.87 Recall:0.83 Precision:0.88 F1:0.86

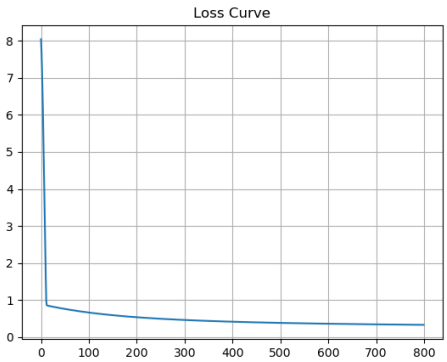


(a) loss curve

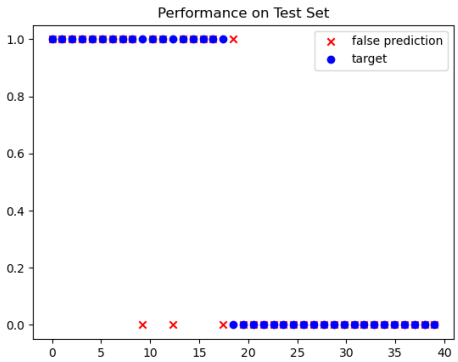


(b) prediction

图 2: SGD with normalization. lr=1e-1 Accuracy:0.95 Recall:0.89 Precision:1.00 F1:0.94

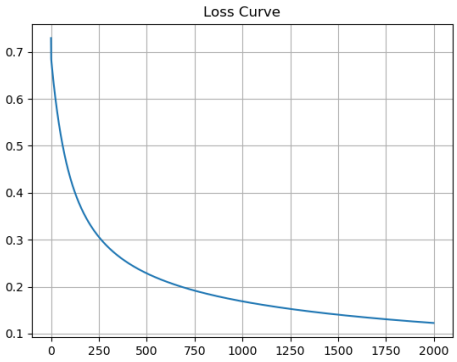


(a) loss curve

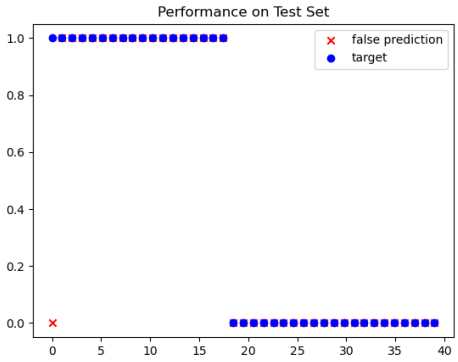


(b) prediction

图 3: MBGD without normalization. lr=1e-6 Accuracy:0.90 Recall:0.83 Precision:0.94 F1:0.88



(a) loss curve



(b) prediction

图 4: MBGD with normalization. lr=2e-2 Accuracy:0.97 Recall:0.94 Precision:1.00 F1:0.97