The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Xiaoxuan Peng

Supervisor:
Mingkui Tan

Student ID：
201730683321

Grade:
Undergraduate

October 26, 2019

Logistic Regression and Support Vector Machine

Abstract—The experiment intends to use logistic regression and support vector machine to show the differences between logistic regression and linear classification.

I. INTRODUCTION

Logistic regression is used to predict the probability that y=0 and y=1, which helps us to make some predictions. Actually, logistic regression is not a regression task but a classification task. Support vector machine is a model to solve linear classification problem. In this experiment, I implement these two models using a9a dataset of LIBSVM Data.

II. METHODS AND THEORY

A. Logistic Function

$$g(z) = \frac{1}{1+e^{-z}}$$

$$g(-z) = \frac{1}{1+e^{z}} = 1 - g(z)$$

If $z \to +\infty$, then $g(z) \to 1$

If $z \to -\infty$, then $g(z) \to 0$

B. Logistic Regression

Assume the labels are binary: $y_i \in \{0,1\}$

$$h_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w^T x}) = \frac{1}{1+e^{-\mathbf{w^T x}}}$$

Probability:

$$p = \begin{cases} h_{\mathbf{w}}(\mathbf{x_i}) & y_i = 1 \\ 1 - h_{\mathbf{w}}(\mathbf{x_i}) & :y_i = 0 \end{cases}$$

C. Log-likehood Loss Function

In Logistic Regression, I use the Log-likehood Loss Function to evaluate the model.

$$J(\mathbf{w}) = -\frac{1}{n}[\sum_{i=1}^{n} y_i \log h_{\mathbf{w}}(\mathbf{x_i}) + (1-y_i)\log(1-h_{\mathbf{w}}(\mathbf{x_i}))]$$

D. Gradient of the Log-likehood Loss Function

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -y \cdot \frac{1}{h_{\mathbf{w}}(\mathbf{x})} \cdot \frac{\partial h_{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{w}} + (1-y) \cdot \frac{1}{1-h_{\mathbf{w}}(\mathbf{x})} \frac{\partial h_{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{w}} = -y \cdot \frac{1}{h_{\mathbf{w}}(\mathbf{x})} \cdot \frac{\partial g(\mathbf{w^T x})}{\partial \mathbf{w}} + (1-y) \cdot \frac{1}{1-h_{\mathbf{w}}(\mathbf{x})} \frac{\partial g(\mathbf{w^T x})}{\partial \mathbf{w}} = (-\frac{xy}{h_{\mathbf{w}}(\mathbf{x})} + \frac{\mathbf{x}(1-y)}{1-h_{\mathbf{w}}(\mathbf{x})}) \cdot g(\mathbf{w}^T\mathbf{x}) \cdot [1 - g(\mathbf{w}^T\mathbf{x})] = (h_{\mathbf{w}}(\mathbf{x}) - y)\mathbf{x}$$

Notes:

$$g(z) = \frac{1}{1+e^{-z}}$$

$$g'(z) = \frac{e^{-z}}{(1+e^{-z})^2} = g(z)[1-g(z)]$$

E. Gradient Descent in Logistic Regression

For a sample:

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = (h_{\mathbf{w}}(\mathbf{x}) - y)\mathbf{x}$$

$$\mathbf{w} := \mathbf{w} - \alpha(h_{\mathbf{w}}(\mathbf{x}) - y)\mathbf{x}$$

For all samples:

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{n}\sum_{i=1}^{n} (h_{\mathbf{w}}(\mathbf{x_i}) - y)\mathbf{x_i}$$

$$\mathbf{w} := \mathbf{w} - \frac{1}{n}\sum_{i=1}^{n} \alpha(h_{\mathbf{w}}(\mathbf{x_i}) - y_i)\mathbf{x_i}$$

F. Support Vector Machine

Choose normalization such that $\mathbf{w}^T\mathbf{x}_+ + b = +1$ and $\mathbf{w}^T\mathbf{x}_- + b = -1$ for the positive and negative support vector respectively.

The margin is given by:

$$\frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot (\mathbf{x}_+ - \mathbf{x}_-) = \frac{\mathbf{w}^T(\mathbf{x}_+ - \mathbf{x}_-)}{\|\mathbf{w}\|} = \frac{(1-b) - (-1-b)}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$

Learning the SVM can be formulated as an optimization:

$$\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|}$$

$$s.t. \mathbf{w}^T\mathbf{x}_i + b = \begin{cases} \geq 1 & y_i = +1 \\ \leq -1 & y_i = -1 \end{cases}$$

Equivalently, we get:

$$\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$$

$$s.t. y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad \text{i=1,2,…,n}$$

G.Hinge Loss in SVM

$$\xi_i = max(0,1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))$$

The optimization problem becomes:

$$\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} + \frac{C}{n}\sum_{i=1}^{n} max(0,1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)$$

H.Minibatch Stochastic Gradient Descent

Gradient Computation:

$$g_{\mathbf{w}}(\mathbf{x}_i) = \frac{\partial \xi_i}{\partial \mathbf{w}} = \begin{cases} -y_i\mathbf{x}_i & 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 0 \\ 0 & 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) < 0 \end{cases}$$

$$g_b(\mathbf{x}_i) = \frac{\partial \xi_i}{\partial b} = \begin{cases} -y_i & 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 0 \\ 0 & 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) < 0 \end{cases}$$

Mini-Batch Stochastic Gradient Descent

$$grad_{\mathbf{w}}L(\mathbf{w},b) = \mathbf{w} + \frac{C}{|\mathcal{S}_k|}\sum_{i\in \mathcal{S}_k} g_{\mathbf{w}}(\mathbf{x}_i)$$

$$grad_b L(\mathbf{w},b) = \frac{C}{|\mathcal{S}_k|}\sum_{i\in \mathcal{S}_k} g_b(\mathbf{x}_i)$$

Notes: We randomly choose a subset $\mathcal{S}_k$, the size of $\mathcal{S}_k$ is $|\mathcal{S}_k|$.

Update:

$$\mathbf{w} = \mathbf{w} - \alpha \cdot grad_{\mathbf{w}}L(\mathbf{w},b)$$
$$b = b - \alpha \cdot grad_{\mathbf{w}}L(\mathbf{w},b)$$

Notes: α is the learning rate.

III. EXPERIMENT

A.Dataset

This dataset is a9a in LIBSVM Data, including 32561/16281(testing) samples and each sample has 123/123 (testing) features.

B. Implementation

(1) Initialization

The dataset is split into two parts, 80% for training set and 20% for validation set, we don't generate test set here.

I use random normal distribution to initialize parameter $\mathbf{w}$:

$$\mathbf{w} \sim N(\mu = 0, \sigma^2 = 1)$$

(2) Parameters

a) There are three parameters in Logistic Regression. The detailed information about there three parameters is listed below.

Table 1-Parameters in Logistic Regression

Parameters	Value
Number of epochs	200
Learning rate	0.0008
Batch size	32

b) There are four parameters in Linear Classification(SVM). The detailed information about these four parameters is listed below.

Table 2-Parameters in Linear Classification(SVM)

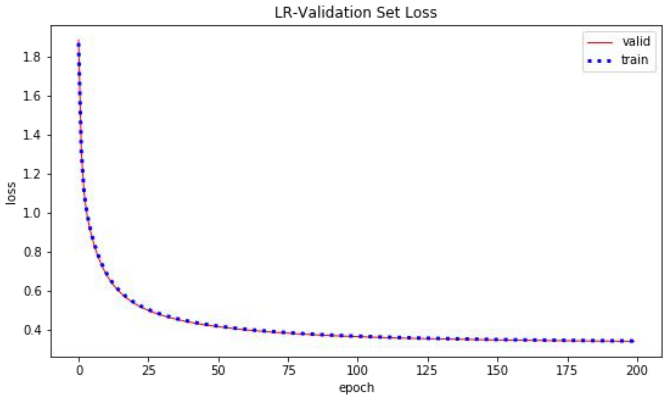
Parameters	Value
Number of epochs	200
Learning rate	0.0003
Batch size	32
C	0.05

(3) Results

a) Logistic Regression

Figure 1 shown below shows the loss on the validation set.

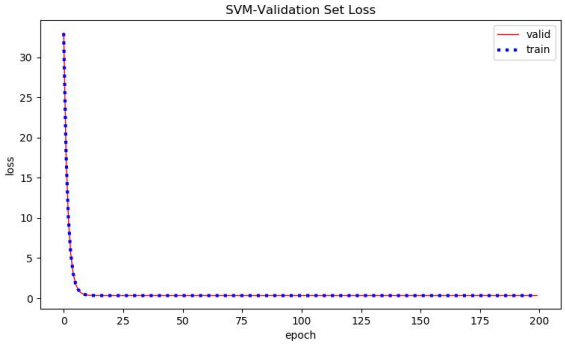
Figure 1-Logistic Regression loss on validation set



b) Linear Classification (SVM)

Figure 2 shown below shows the loss on the validation set.

Figure 2-SVM loss on validation set



c) Changing Parameters

Figure 3 shown below shows the Logistic Regression result, I make learning rate much larger here. Learning rate = 0.3. Other parameters are the same. The result is not converged.

Figure 3- Logistic Regression loss on validation set(Large LR)

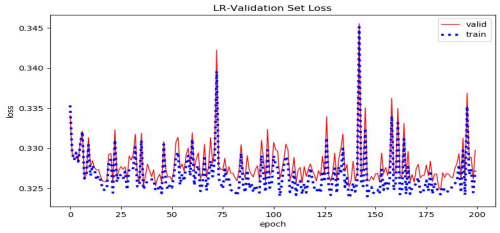


Figure 4 shown below shows the Logistic Regression result, I make learning rate much smaller here. Learning rate = 1e-6. Other parameters are the same. It is very slow for it to converge.

Figure 4- Logistic Regression loss on validation set(Small LR)

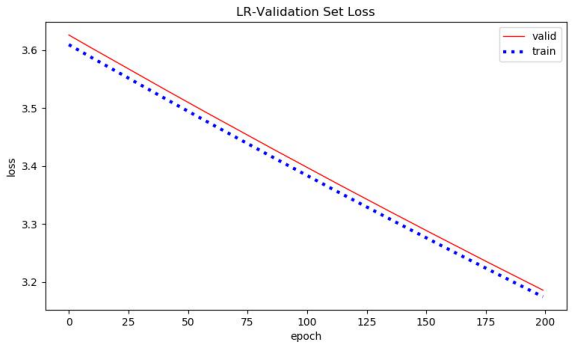


Figure 5 shown below shows the Linear Classification result. I make learning rate much larger here. Learning rate = 0.3. Other parameters are the same. It is not converged.

Figure 5- SVM loss on validation set(Large LR)

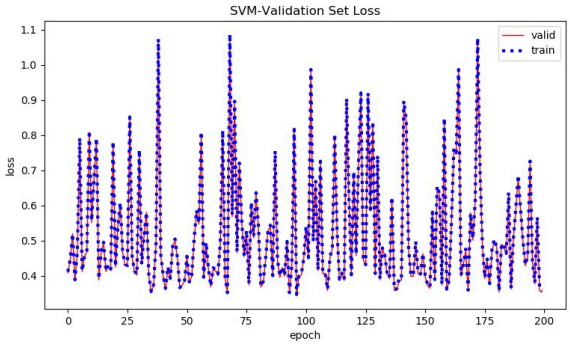


Figure 6 shown below shows the Linear Classification result. I make learning rate much larger here. Learning rate = 1e-6. Other parameters are the same. It is very slow for it to converge.

Figure 6- SVM loss on validation set(Small LR)

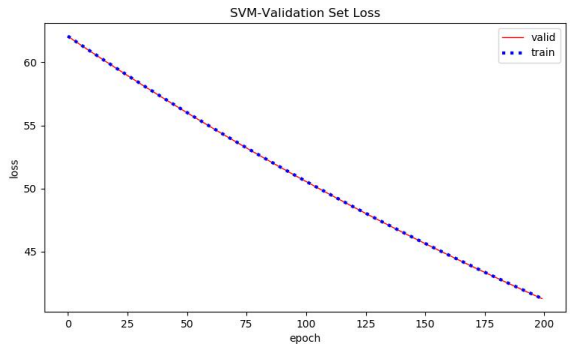
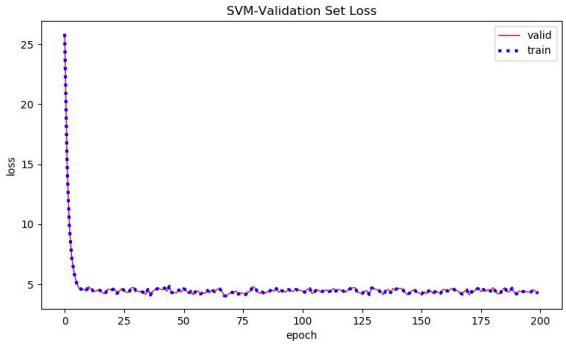


Figure 7 shown below shows the Linear Classification result. I make the parameter C much larger here. C = 20. Other parameters are the same. Large C makes constraints hard to ignore.

Figure 7- SVM loss on validation set(Large C)



IV. CONCLUSION

(1) There are many loss functions. In lab1, I use least square loss function. In this experiment, I use Log-likehood loss function in Logistic Regression and Hinge loss function in linear classification, exactly SVM.

(2) Parameters play an important role in each model. In both models, if I choose a much larger BATCH_SIZE, the iterations run faster and if I choose a much smaller BATCH_SIZE, the iterations run slower. Also, if I make the learning rate much larger, the loss function may not converge and if I make the learning rate much smaller, the loss function will be very slow to converge. As for SVM, the parameter C is also important. If I use a larger C, it makes constraints hard to ignore and if I use a smaller C, it allows constraints to be easily ignored