华南理工大学

South China University of Technology

The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Canguang Li

Supervisor:
Mingkui Tan

Student ID：
201530611937

Grade:
Undergraduate

December 14, 2017

Logistic Regression, Linear Classification and Stochastic Gradient Descent

Abstract—In this experiment, logistic regression and linear classification algorithms were implemented using mini-batch, stochastic gradient descent and the model parameters were updated with four different optimization methods-NAG, RMSProp, AdaDelta and Adam.

I. INTRODUCTION

Through implementing logistic regression and linear classification with stochastic gradient descent, this experiment explored the difference between logistic regression and linear classification as well as the difference between gradient descent and stochastic gradient descent. What's more, this experiment explored the effect of four difference optimization methods-NAG, RMSProp, AdaDelta and Adam.

II. METHODS AND THEORY

A. Logistic regression

The logistic regression model function used in the experiment is

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

The loss function used is

$$J(W) = \frac{\lambda}{2}\|W\|_2^2 + \frac{1}{n}\sum_{i=1}^{n}\log\left(1+e^{-y_i W^T x_i}\right)$$

Its gradient is

$$G(W) = \lambda W - \frac{1}{n}\sum_{i=1}^{n}\frac{y_i \cdot x_i}{1+e^{y_i \cdot W^T x_i}}$$

B. Linear classification

The linear classification model function used in the experiment is

$$f(X) = W^T X + b$$

The loss function used is

$$\text{Loss} = \frac{\|W\|^2}{2} + \frac{C}{n}\sum_{i=1}^{n}\max\left(0, 1-y_i(W^T x_i + b)\right)$$

Denote

$$g_W(x_i) = \begin{cases} -y_i x_i & 1-y_i(W' x_i + b) >= 0 \\ 0 & 1-y_i(W' x_i + b) < 0 \end{cases}$$

$$g_b(x_i) = \begin{cases} -y_i & 1-y_i(W' x_i + b) >= 0 \\ 0 & 1-y_i(W' x_i + b) < 0 \end{cases}$$

then the gradient is

$$\frac{\partial \text{Loss}}{\partial W} = W + \frac{C}{n}\sum_{i=1}^{n} g_{W(x_i)}$$

$$\frac{\partial \text{Loss}}{\partial b} = \frac{C}{n}\sum_{i=1}^{n} g_b(x_i)$$

C. Optimization methods

Four different optimization methods used in this experiment is NAG, RMSProp, AdeDelta and Adam.

The update step of NAG is

$$g_t \leftarrow \nabla J(W_{t-1} - \gamma v_{t-1})$$
$$v_t \leftarrow \gamma v_{t-1} + \eta g_t$$
$$W_t \leftarrow W_{t-1} - v_t$$

The update step of RMSProp is

$$g_t \leftarrow \nabla J(W_{t-1})$$
$$G_t \leftarrow \gamma G_t + (1-\gamma) g_t \odot g_t$$
$$W_t \leftarrow W_{t-1} - \frac{\eta}{\sqrt{G_t + \epsilon}} \odot g_t$$

The update step of AdaDelta is

$$g_t \leftarrow \nabla J(W_{t-1})$$
$$G_t \leftarrow \gamma G_t + (1-\gamma) g_t \odot g_t$$
$$\Delta W_t \leftarrow -\frac{\sqrt{\Delta_{t-1} + \epsilon}}{\sqrt{G_t + \epsilon}} \odot g_t$$
$$W_t \leftarrow W_{t-1} + \Delta W_t$$
$$\Delta_t \leftarrow \gamma \Delta_{t-1} + (1-\gamma)\Delta W_t \odot \Delta W_t$$

The update step of Adam is

$$g_t \leftarrow \nabla J(W_{t-1})$$
$$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$$
$$G_t \leftarrow \gamma G_t + (1-\gamma) g_t \odot g_t$$
$$\alpha \leftarrow \eta \frac{\sqrt{1-\gamma^t}}{1-{\beta_1}^t}$$
$$W_t \leftarrow W_{t-1} - \alpha \frac{m_t}{\sqrt{G_t + \epsilon}}$$

III. EXPERIMENT

A. Dataset

Experiment uses a9a of LIBSVM Data, including 32561/16281(testing) samples and each sample has 123/123(testing) features.

B. Implementation

The parameters(of logistic regression and linear classification) were initialized as follow:

TABLE1
Initialized parameters

	NAG	RMSProp	AdaDelta	Adam
ϵ	-	1e-8	1e-8	1e-8
v_t(vetor)	0	-	-	-
G_t(vetor)	-	0	0	0
m_t(vetor)	-	-	-	0
Δ_t(vetor)	-	-	0	-
η	0.1	0.1	-	0.1
γ	0.9	0.9	0.9	0.9
β_1	-	-	-	0.9

λ/C	1	1	1	1
W(vetor)	0	0	0	0

After exploring, some super parameters(of logistic regression and linear classification) were as follow:

TABLE2

Adjusted super parameters

	NAG	RMSProp	AdaDelta	Adam
η	0.01	0.01	-	0.01
γ	0.9	0.9	0.9999	0.9
β_1	-	-	-	0.9

The implementation of four optimization method were as follow:

```
def NAG(X, y, W, params):
    W_ = W - params['GAMMA'] * params['Vt']

    gradient = compute_gradient(X, y, W_)
    params['Vt'] = params['GAMMA'] * params['Vt'] \
        + params['ETA'] * gradient
    W = W - params['Vt']

    return W, params
```

Figure 1 Implementation of NAG

```
def RMSProp(X, y, W, params):
    EPSILON = 1e-8

    gradient = compute_gradient(X, y, W)
    params['Gt'] = params['GAMMA'] * params['Gt'] \
        + (1 - params['GAMMA']) * (gradient * gradient)
    W = W - (params['ETA'] / np.sqrt(params['Gt'] \
        + EPSILON)) * gradient

    return W, params
```

Figure 2 Implementation of RMSProp

```
def AdaDelta(X, y, W, params):
    EPSILON = 1e-8
    GAMMA = params['GAMMA']

    gradient = compute_gradient(X, y, W)
    params['Gt'] = GAMMA * params['Gt'] \
        + (1 - GAMMA) * (gradient * gradient)
    Delta_W = -(np.sqrt(params['Delta_t'] \
        + EPSILON) / np.sqrt(params['Gt'] + EPSILON)) * gradient
    params['Delta_t'] = GAMMA * params['Delta_t'] \
        + (1 - GAMMA) * (Delta_W * Delta_W)
    W = W + Delta_W

    return W, params
```

Figure 3 Implementation of AdaDelta

```
def Adam(X, y, W, params):
    EPSILON = 1e-8
    BETA1 = params['BETA1']
    GAMMA = params['GAMMA']

    gradient = compute_gradient(X, y, W)
    params['mt'] = BETA1 * params['mt'] \
        + (1 - BETA1) * gradient
    params['Gt'] = GAMMA * params['Gt'] \
        + (1 - GAMMA) * (gradient * gradient)
    ALPHA = params['ETA'] * \
        math.sqrt(1 - GAMMA ** params['iteration']) \
        / (1 - BETA1 ** params['iteration'])
    W = W - ALPHA * (params['mt'] \
        / np.sqrt(params['Gt'] + EPSILON))

    return W, params
```

Figure 4 Implementation of Adam

As for logistic regression, I assigned 100 to batch number, 0.5 to threshold and 100 to iteration time.And the loss curve was as figure 5:

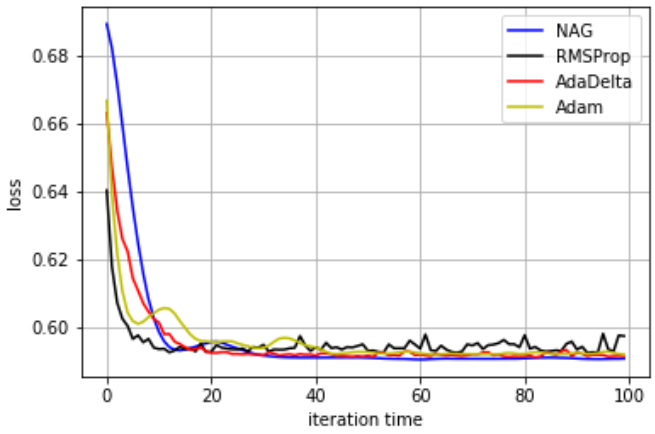

Figure 5 Loss curve of four optimization method

As for linear regression, I assigned 100 to batch number, 0 to threshold and 100 to iteration time.And the loss curve was as figure 4:

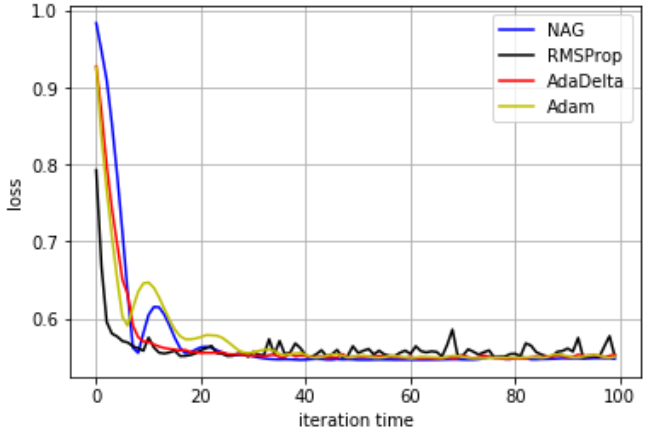

Figure 6 Loss curve of four optimization method

IV. CONCLUSION

Through implementing logistic regression and linear classification with stochastic gradient descent, I learned that logistic regression is not a model to do regression but a model to classify. Logistic regression and linear classification all can solve two-classification problem. Besides, I learned that the difference between gradient descent and stochastic gradient descent. Gradient descent uses all the samples to compute the gradient each iteration but stochastic gradient descent uses just one samples to compute the gradient each iteration. Gradient descent performs well but slowly, while stochastic gradient descent performs not very well but fast. To balance, this experiment used some samples to compute gradient each iteration(mini-batch stochastic gradient descent), this method performs well and fast. As for four optimization methods, I learned that AdaDelta converge slower than other three methods under the same conditions. And RMSProp as well as Adam often oscillates slightly when they trend to converge. But all the optimization methods used in this experiment all can make the loss curve converge to a similar value.