華南理工大學

South China University of Technology

The Experiment Report of *Machine Learning*

College	**Software College**
Subject	**Software Engineering**
Members	Mo Junwen
Student ID	201530612545
E-mail	545096186@qq.com
Tutor	Tan Mingkui
Date submitted	2017. 12 .11

1. Topic:Logistic Regression, Linear Classification and Stochastic Gradient Descent

2. Time: 2017.12.11

3. Reporter:Mo Junwen

4. Purposes:

(1)Compare and understand the difference between gradient descent and stochastic gradient descent.

(2)Compare and understand the differences and relationships between Logistic regression and linear classification.

(3)Further understand the principles of SVM and practice on larger data.

5. Data sets and data analysis:

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

6. Experimental steps:

Logistic Regression and Stochastic Gradient Descent

(1)Load the training set and validation set.

(2)Initalize logistic regression model parameters, you can consider initalizing zeros, random numbers or normal distribution.

(3)Select the loss function and calculate its derivation, find more detail in PPT.

(4)Calculate gradient toward loss function from partial samples.

(5)Update model parameters using different optimized methods(NAG，RMSProp，AdaDelta and Adam).

(6)Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative.

(7)Predict under validation set and get the different optimized method loss L_{NAG}，$L_{RMSprop}$，$L_{AdaDelta}$ and L_{Adam} .

(8)Repeat step 4 to 6 for several times, and drawing graph of L_{NAG}，$L_{RMSprop}$，$L_{AdaDelta}$ and L_{Adam} and with the number of iterations.

Linear Classification and Stochastic Gradient Descent

(1)Load the training set and validation set.

(2)Initalize SVM model parameters, you can consider initalizing zeros, random numbers or normal distribution.

(3)Select the loss function and calculate its derivation, find more detail in PPT.

(4)Calculate gradient toward loss function from partial samples.

(5)Update model parameters using different optimized methods(NAG，RMSProp，AdaDelta and Adam).

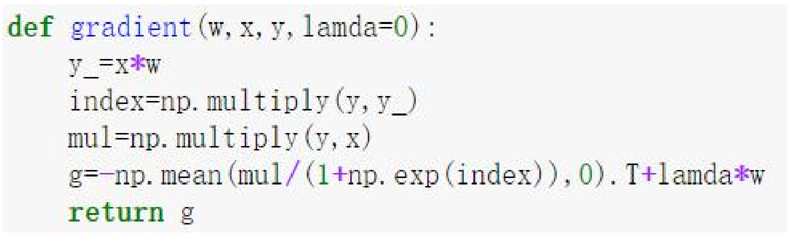
(6)Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative.

(7)Predict under validation set and get the different optimized method loss L_{NAG}，$L_{RMSprop}$，$L_{AdaDelta}$ and L_{Adam} and .

(8)Repeate step 4 to 6 for several times, and drawing graph of L_{NAG}, $L_{RMSprop}$, $L_{AdaDelta}$ and L_{Adam} and with the number of iterations.

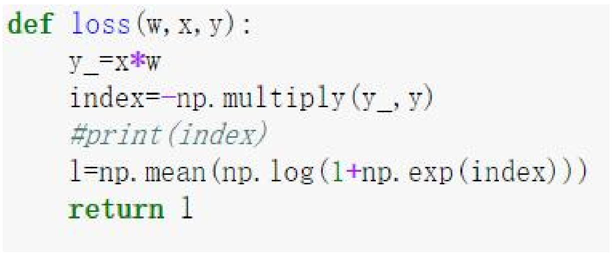
7. Code:

Logistic regression gradient:



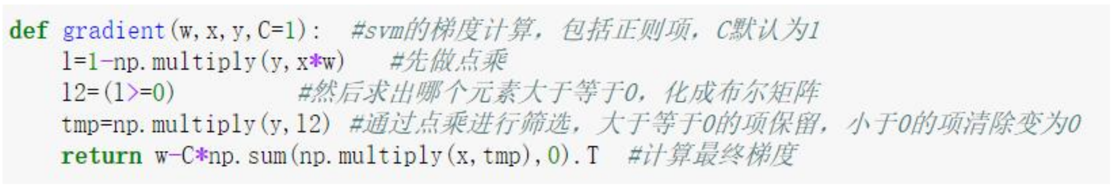
```
def gradient(w, x, y, lamda=0):
    y_=x*w
    index=np.multiply(y, y_)
    mul=np.multiply(y, x)
    g=-np.mean(mul/(1+np.exp(index)), 0).T+lamda*w
    return g
```

Logistic regression loss function:



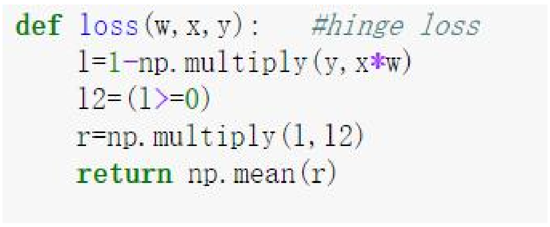
```
def loss(w, x, y):
    y_=x*w
    index=-np.multiply(y_, y)
    #print(index)
    l=np.mean(np.log(1+np.exp(index)))
    return l
```

Linear Classification gradient:



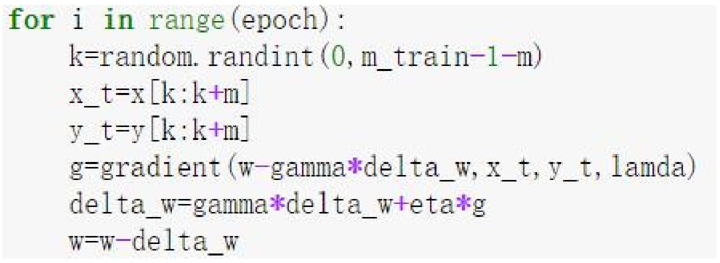
```
def gradient(w,x,y,C=1):  #svm的梯度计算，包括正则项，C默认为1
    l=1-np.multiply(y,x*w)    #先做点乘
    l2=(l>=0)             #然后求出哪个元素大于等于0，化成布尔矩阵
    tmp=np.multiply(y,l2) #通过点乘进行筛选，大于等于0的项保留，小于0的项清除变为0
    return w-C*np.sum(np.multiply(x,tmp),0).T  #计算最终梯度
```

Linear Classification loss function:



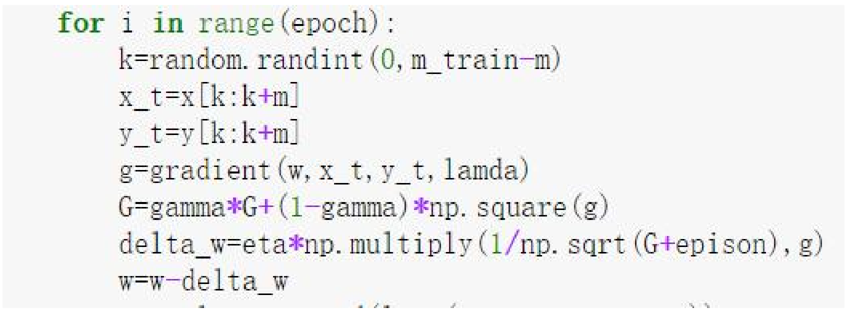
```
def loss(w, x, y):    #hinge loss
    l=1-np.multiply(y, x*w)
    l2=(l>=0)
    r=np.multiply(l, l2)
    return np.mean(r)
```

NAG main code:



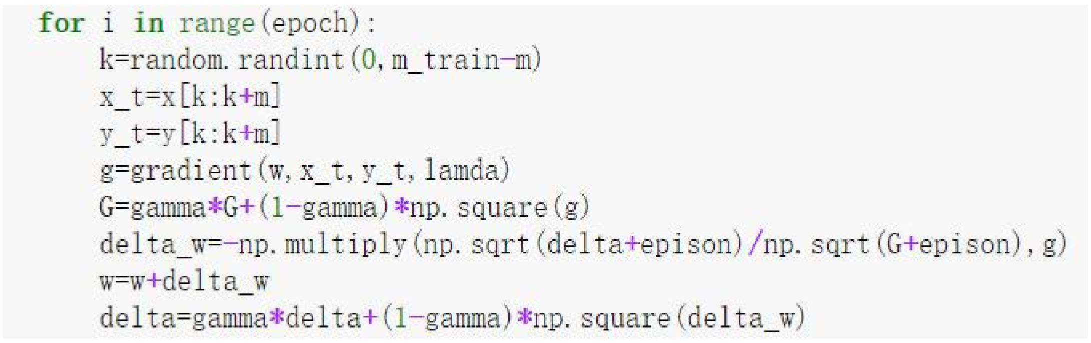
```
for i in range(epoch):
    k=random.randint(0,m_train-1-m)
    x_t=x[k:k+m]
    y_t=y[k:k+m]
    g=gradient(w-gamma*delta_w,x_t,y_t,lamda)
    delta_w=gamma*delta_w+eta*g
    w=w-delta_w
```

RMSprop main code:



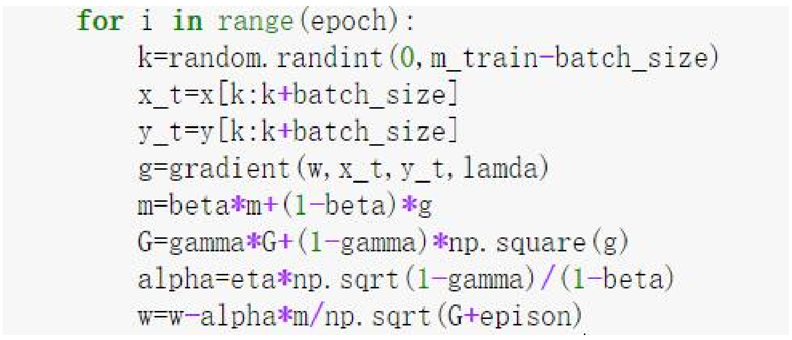
```
    for i in range(epoch):
        k=random.randint(0,m_train-m)
        x_t=x[k:k+m]
        y_t=y[k:k+m]
        g=gradient(w,x_t,y_t,lamda)
        G=gamma*G+(1-gamma)*np.square(g)
        delta_w=eta*np.multiply(1/np.sqrt(G+epison),g)
        w=w-delta_w
```

AdaDelta main code:



```
  for i in range(epoch):
      k=random.randint(0,m_train-m)
      x_t=x[k:k+m]
      y_t=y[k:k+m]
      g=gradient(w,x_t,y_t,lamda)
      G=gamma*G+(1-gamma)*np.square(g)
      delta_w=-np.multiply(np.sqrt(delta+epison)/np.sqrt(G+epison),g)
      w=w+delta_w
      delta=gamma*delta+(1-gamma)*np.square(delta_w)
```

Adam main code:



```
     for i in range(epoch):
         k=random.randint(0,m_train-batch_size)
         x_t=x[k:k+batch_size]
         y_t=y[k:k+batch_size]
         g=gradient(w,x_t,y_t,lamda)
         m=beta*m+(1-beta)*g
         G=gamma*G+(1-gamma)*np.square(g)
         alpha=eta*np.sqrt(1-gamma)/(1-beta)
         w=w-alpha*m/np.sqrt(G+epison)
```

8. The initialization method of model parameters:

Logistic Regression:random numbers or normal distribution.

Linear Classification:random numbers or normal distribution.

9. The selected loss function and its derivatives:

Logistic Regression:

Loss function:

$$J(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\log(1+e^{-y_i\cdot\mathbf{w}^\top\mathbf{x}_i})+\frac{\lambda}{2}||\mathbf{w}||_2^2$$

Derivatives:

$$\frac{\partial \mathrm{J}}{\partial w} = \frac{1}{n}\sum_{i=1}^{n}\frac{-y_i x_i}{1+e^{-y_i w^T x_i}} + \lambda w$$

Linear Classification:

Loss function:

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

Derivatives:

$$\frac{1}{2}\cdot\frac{\partial(\|\mathbf{w}\|^2)}{\partial\mathbf{w}} = \mathbf{w}$$

if $1-y_i(\mathbf{w}^\top\mathbf{x}_i+b) >= 0$:

$$\begin{aligned} g_{\mathbf{w}}(\mathbf{x}_i) &= \frac{\partial(-y_i(\mathbf{w}^\top\mathbf{x}_i+b))}{\partial\mathbf{w}} \\ &= -\frac{\partial(y_i\mathbf{w}^\top\mathbf{x}_i)}{\partial\mathbf{w}} \\ &= -y_i\mathbf{x}_i \end{aligned}$$

if $1-y_i(\mathbf{w}^\top\mathbf{x}_i+b) < 0$:

$$g_{\mathbf{w}}(\mathbf{x}_i) = 0$$

$$g_{\mathbf{w}}(\mathbf{x}_i) = \begin{cases} -y_i\mathbf{x}_i & 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b) >= 0 \\ 0 & 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b) < 0 \end{cases}$$

$$\frac{\partial f(\mathbf{w}, b)}{\mathbf{w}} = \mathbf{w} + C\sum_{i=1}^{N} g_{\mathbf{w}}(\mathbf{x}_i)$$

10. Experimental results and curve:

Logistic Regression:

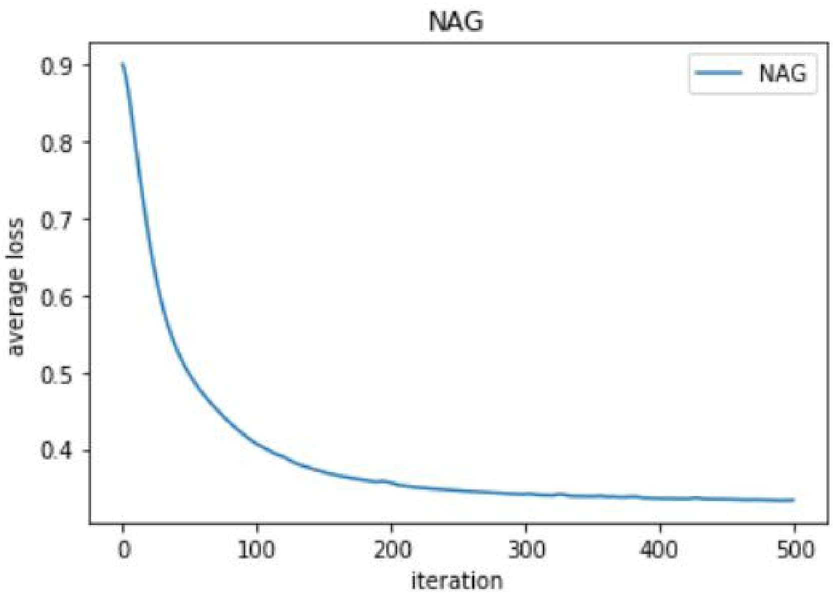
(1)NAG

Hyper-parameter selection:

η =0.05 λ =0.001 γ =0.9 batch_size=256 epoch=500

Predicted Results (Best Results):84.61%

Loss curve:



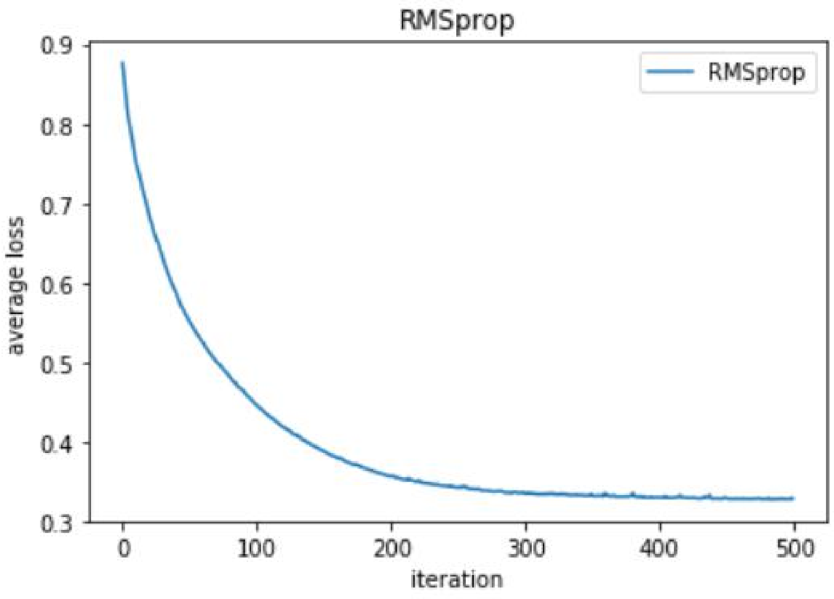
(2) RMSprop

Hyper-parameter selection:

η =0.01 λ =0.001 γ =0.9 batch_size=256 epoch=500

Predicted Results (Best Results):84.97%

Loss curve:



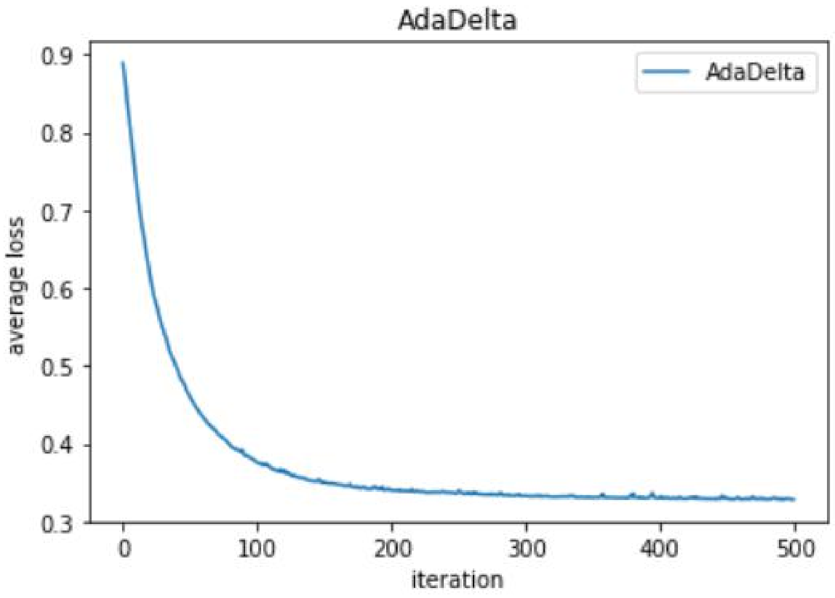
(3) AdaDelta

Hyper-parameter selection:

λ =0.001 γ =0.4 batch_size=256 epoch=500

Predicted Results (Best Results):85.03%

Loss curve:



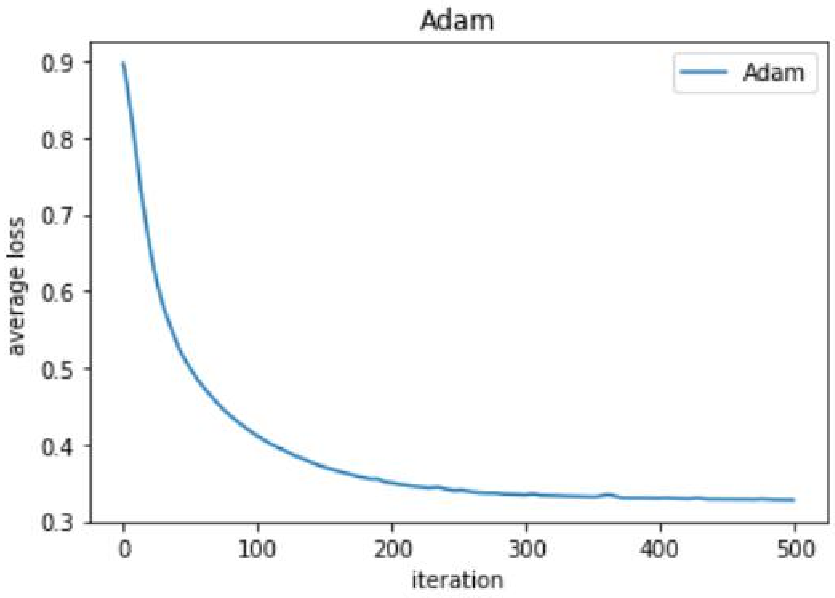
(4) Adam

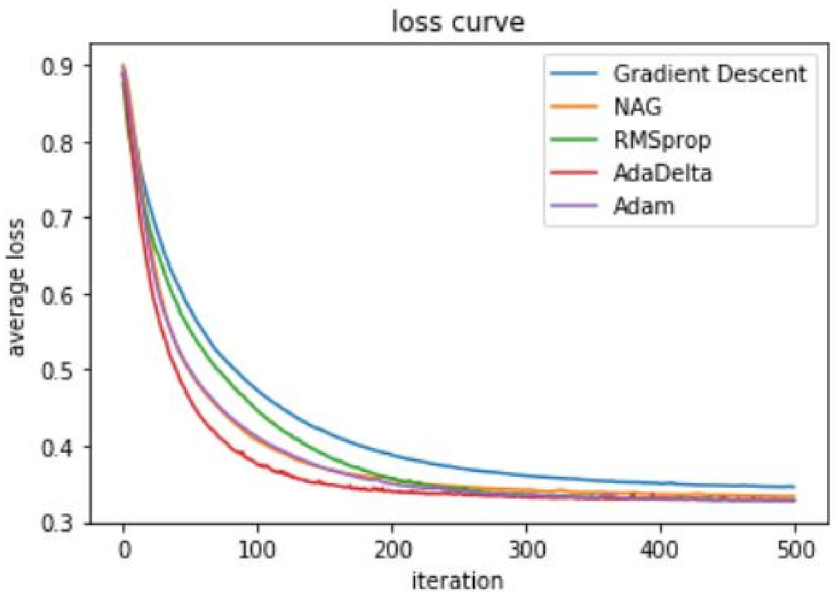
Hyper-parameter selection:

η =0.005 λ =0.001 γ =0.99 β =0.9 batch_size=256 epoch=500

Predicted Results (Best Results):84.96%

Loss curve:





Linear Classification:

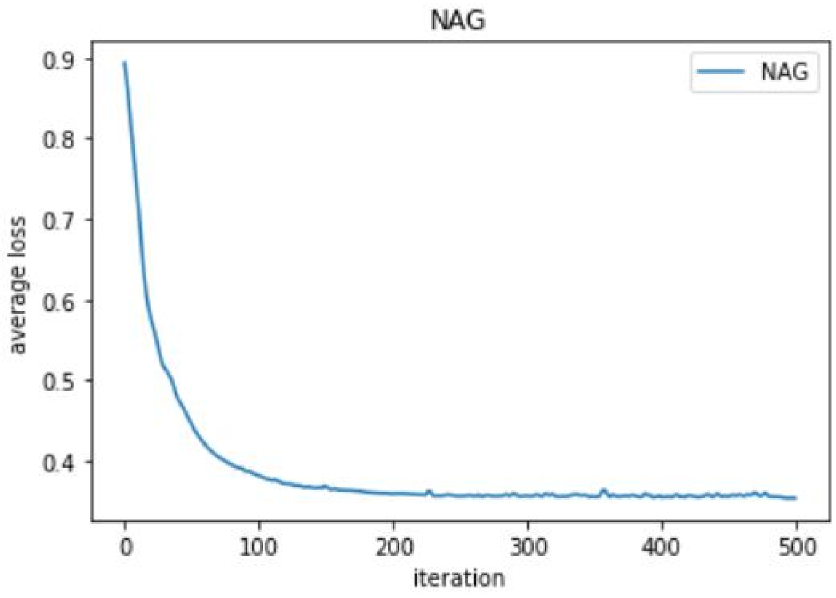
(1)NAG

Hyper-parameter selection:

η =0.0003 C=1 γ =0.9 batch_size=256 epoch=500

Predicted Results (Best Results):84.61%

Loss curve:



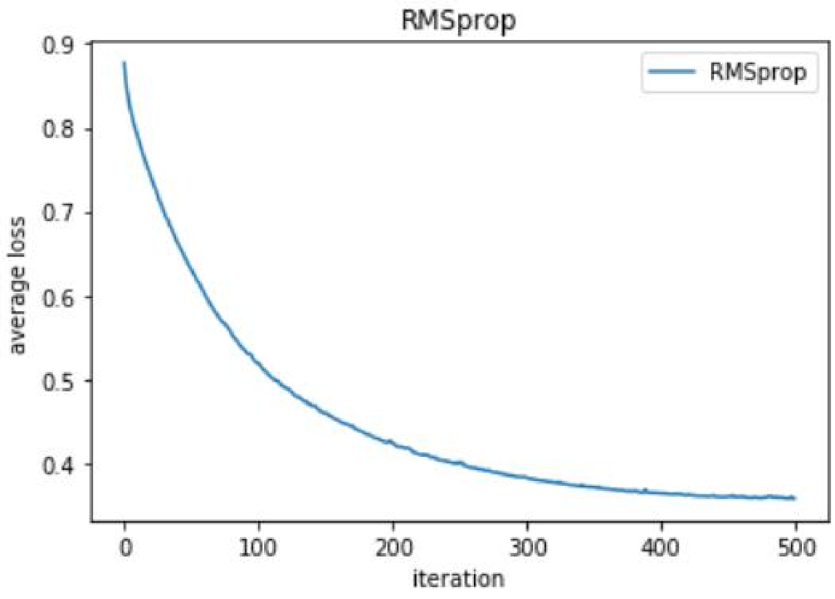
(2)RMSprop

Hyper-parameter selection:

η =0.005 C=1 γ =0.9 batch_size=256 epoch=500

Predicted Results (Best Results):84.78%

Loss curve:



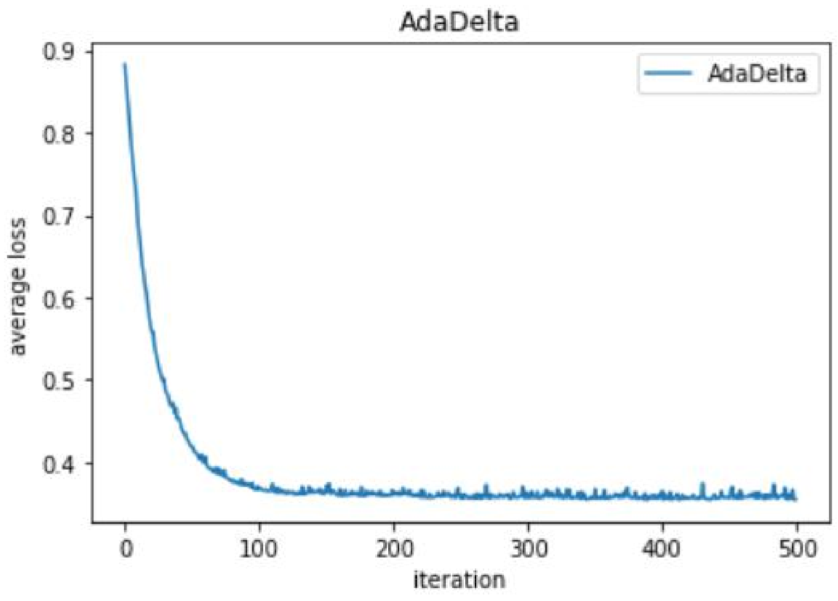
(3)AdaDelta

Hyper-parameter selection:

C=0.9 γ =0.3 batch_size=256 epoch=500

Predicted Results (Best Results):84.95%

Loss curve:



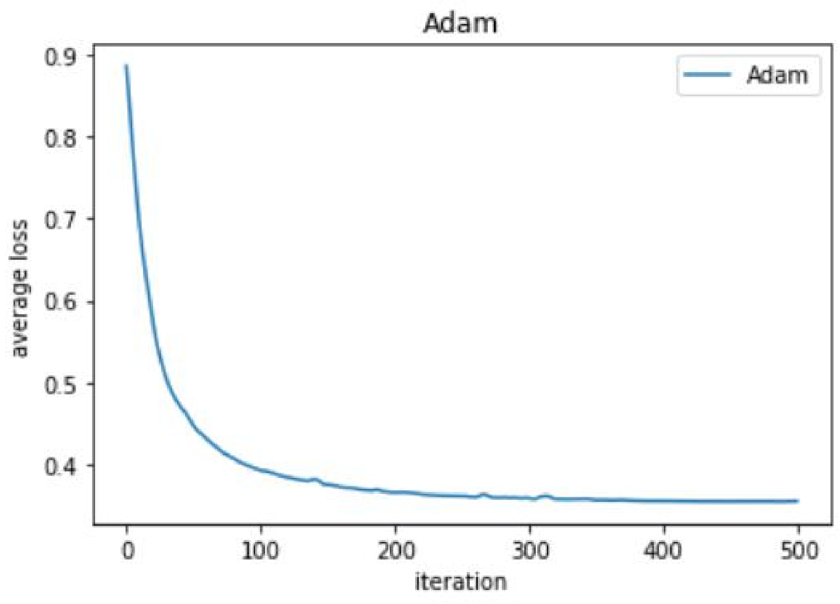
(4)Adam

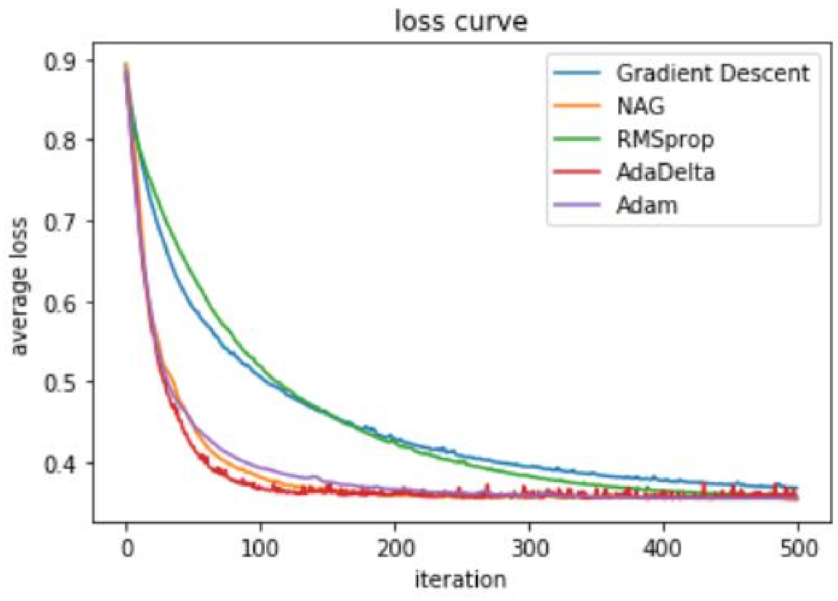
Hyper-parameter selection:

η =0.01 C=1 γ =0.999 β =0.9 batch_size=256 epoch=500

Predicted Results (Best Results):84.78%

Loss curve:





11. Results analysis:

In this experiment, we compare logistic regression and linear classification. We can find in the loss curve that the loss of the logistic regression is more stable. And their accuracy is similar. We also compare different stochastic gradient descent in this experiment. We can find that AdaDelta converge fastest. Adam and the NAG are similar in the convergence rate. RMSprop converge a little faster than the mini-batch gradient descent.

12. Similarities and differences between logistic regression and linear classification：

Similarities: They are both used in classification problem. When the threshold in linear classification is 0 and the threshold in logistic regression is 0.5, logistic regression and linear classification have similar meanings.

Differences: The loss and goal functions of them are different. Logistic regression can be used to solve multiclass classification problem while svm is hard to do that.

13. Summary:

In this experiment, we use the logistic regression and linear classification to solve the classification problem. Also, we try different stochastic gradient descent and compare them. The stochastic gradient descent can solve the memory error and speed up the training process but it will make the loss unstable. Different algorithms have different convergence rates. Adadelta converge fastest and it doesn’t require the learning rate but its loss is unstable. Adam and NAG also run fast. And RMSprop is a little faster than the gradient descent.