

**The Experiment Report of**

***Machine Learning***

**College Software College**

**Subject Software Engineering**

**Members**   **庄仁鑫**

**Student ID 201530613979**

**E-mail 529266221@qq.com**

**Tutor**   **谭明奎**

**Date submitted** **2017. 12 .**

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

**2. Time: 2017-12-09-2017-12-15**

**3. Reporter: 庄仁鑫**

**4. Purposes:** Compare and understand the difference between gradient descent and stochastic gradient descent.

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

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

1. **Data sets and data analysis:**

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

1. **Experimental steps:**

**Logistic Regression and Stochastic Gradient Descent：**

Load the training set and validation set.

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

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

Calculate gradient G toward loss function from partial samples.

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

Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Predict under validation set and get the different optimized method loss ，

Repeate step 4 to 6 for several times, and drawing graph of  with the number of iterations.

**Linear Classification and Stochastic Gradient Descent:**

Load the training set and validation set.

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

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

Calculate gradient toward loss function from partial samples.

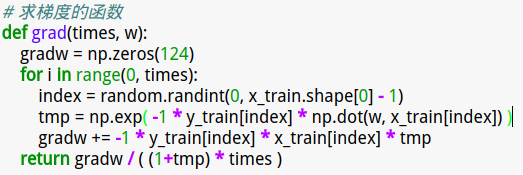
Update model parameters using different optimized methods(NAG，RMSProp，AdaDelta and Adam).

Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Predict under validation set and get the different optimized method loss ，

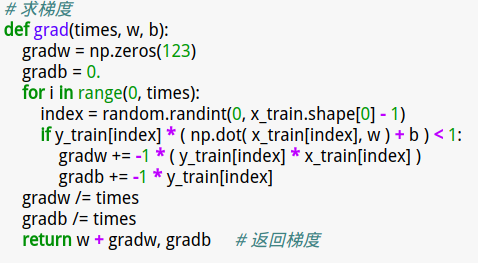
Repeate step 4 to 6 for several times, and drawing graph of  with the number of iterations.

1. **Code:**

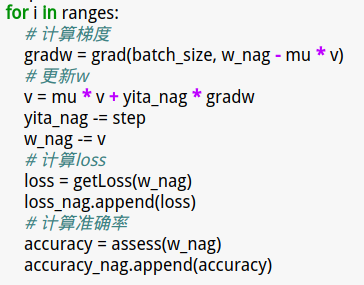
The gradient computing function of Logistic Regression is:



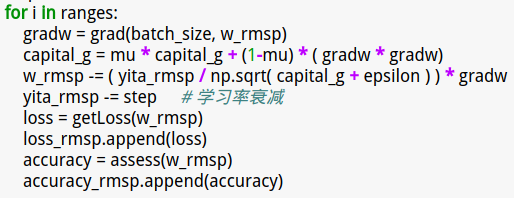
The gradient computing function of Support Vector Machine is:



The gradient descent process of NAG is:



The gradient descent process of RMSProp is:



The gradient descent process of AdaDelta is:

The gradient descent process of Adamis:

(Fill in the contents of 8-11 respectively for logistic regression and linear classification)

1. **The initialization method of model parameters:**

**For Logistic Regression and Stochastic Gradient Descent：**

**Randomly initialization**

w\_nag = np.random.rand(124) # 模型参数初始化

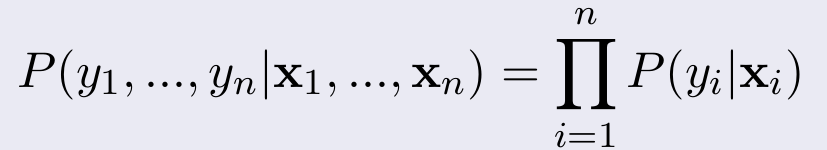
w\_rmsp = np.random.rand(124)

**For** **Linear Classification and Stochastic Gradient Descent:**

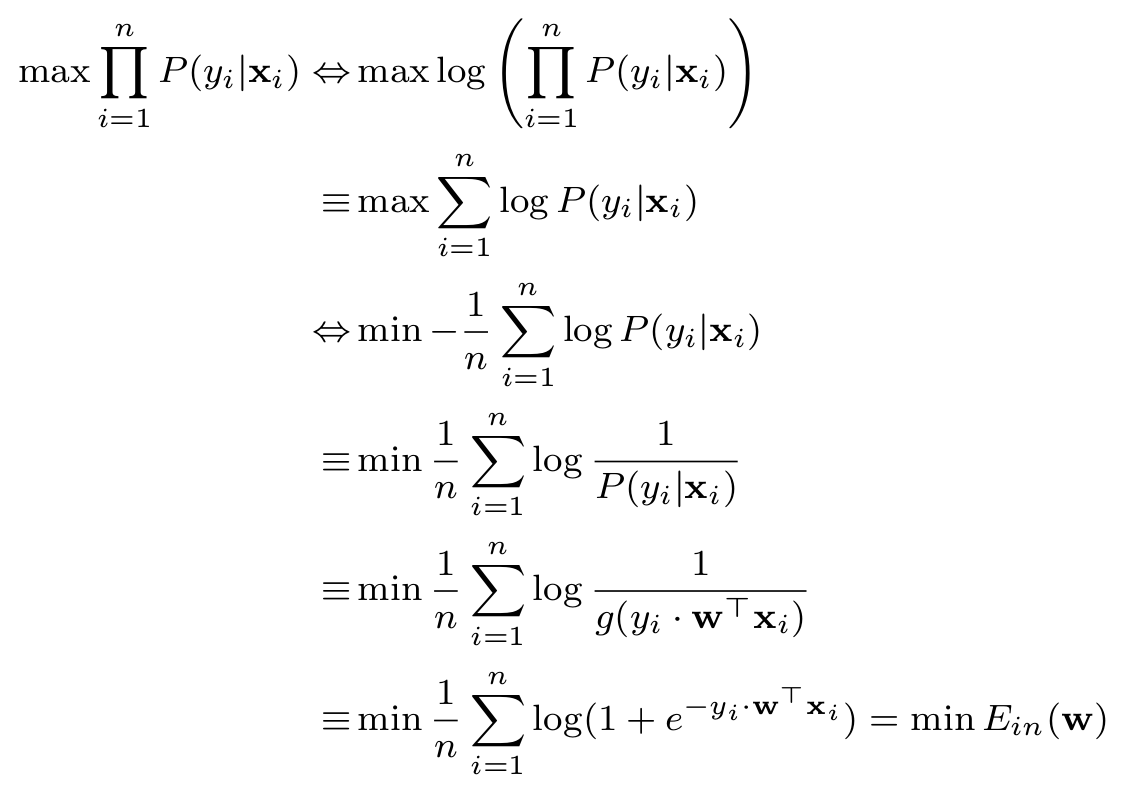
**Randomly Initialization**

1. **The selected loss function and its derivatives:**

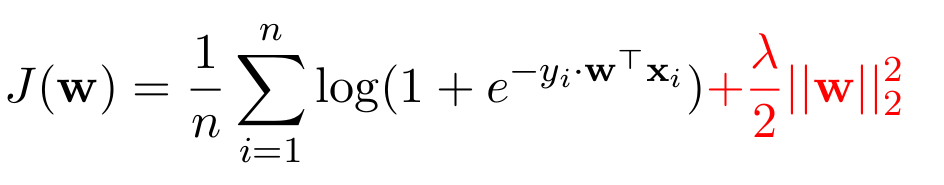
**For Logistic Regression and Stochastic Gradient Descent：**

We want to maximize the likelyhood of training data: 

Instead, turning it into a minimization problem helps:



Then, to avoid overfitting, we add a regularizer to the target function , so the target function that we want to minimize became:

 Its derivative is as follow:

1. **Experimental results and curve:**(Fill in this content for various methods of gradient descent respectively)

**For Logistic Regression and Stochastic Gradient Descent：**

**# NAG**

## Hyper-parameter selection:

1. yita\_nag = 0.001 # 学习率

lambda\_nag = 0.9 # 正则项的权重

batch\_size = 1 # 求梯度的样本数

The accuracy rate is not high enough. After 300 iterations it’s about 75%.

1. yita\_nag = 0.001 # 学习率，即梯度的加权值

batch\_size = 1 # 求梯度的样本数

lambda\_nag = 1 # 正则项的权重

Lambda\_nag is increased to 1. After 150 iterations the accuracy rate began decreasing slightly. After 300 iterations the accuracy rate reached approximately 76.5%.

## Predicted Results (Best Results):

yita\_nag = 0.001 # 学习率，即梯度的加权值

batch\_size = 1 # 求梯度的样本数

lambda\_nag = 1 # 正则项的权重

The accuracy rate after 300 iterations is approximately 76.5%.

**# RMSProp**

## Hyper-parameter selection:

1. yita\_rmsp = 0.01

batch\_size = 2\*\*4

lambda\_rmsp = 1

The loss curve is quite smooth, but the accuracy rate after 300 iterations is relatively small, 75.6%.

1. batch\_size is changed into 2\*\*2=4. The loss curve is not so smooth as that of batch\_size=16. But after 300 iterations the accuracy rate reached 77.5%.
2. After promoting the lambda\_rmsp to be 1.1 the loss curve decreases faster and after 300 iterations the accuracy rate reached 79.5%, which is quite satisfactory.
3. When the yita\_rmsp is set to be 0.05, the loss curve vibrates obviously and the final accuracy rate only reached 74.8%.
4. yita\_rmsp = 0.01

batch\_size = 2\*\*2

lambda\_rmsp = 1.1

The curve is descenting and after 300 iterations the accuracy rate reached 80%.

## Predicted Results (Best Results):

yita\_rmsp = 0.01

batch\_size = 2\*\*2

lambda\_rmsp = 1.1

After 300 iterations the accuracy rate reached 80%.

**# AdaDelta**

## Hyper-parameter selection:

## Predicted Results (Best Results):

**# Adam**

## Hyper-parameter selection:

## Predicted Results (Best Results):

## Loss curve:

**11. Results analysis:**

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

**13. Summary:**