华南理工大学

South China University of Technology

# The Experiment Report of

# *Machine Learning*


**College** Software College

**Subject** Software Engineering

**Members** 张超弦

**Student ID** 201530613597

**E-mail** 864858406@qq.com

**Tutor** 谭明奎


**Date submitted** 2017. 12 . 7

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

**2. Time:** 2017.12.7

**3. Reporter:**张超弦

**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. Please download the training set and validation set.

**6. Experimental steps:**

**For 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. Predict under validation set and get the different optimized method loss $L_{NAG}$ , $L_{RMSProp}$ , $L_{AdaDelta}$ and $L_{Adam}$.

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

**For 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. Predict under validation set and get the different optimized method loss ，，

and .

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

**7. Code:**

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

```
def train(X_train, Y_train, theta, learning_rate, optimization, optimizer_params, t):
    gradient = 1.0/X_train.shape[0] * np.dot(X_train.transpose(), sigmoid(X_train.dot(theta))-Y_train)
    if optimization == 'initial':
        theta -= learnign_rate * gradient
    elif optimization == 'NAG':
```

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

```
def train(X_train, Y_train, theta, learning_rate, optimization, optimizer_params, t, C=0.9):
    X =  (1 - Y_train * X_train.dot(theta) < 0)
    Y = Y_train.copy()
    Y[X] = 0
    E_gradient = -np.dot(X_train.transpose(), Y)
    gradient = theta + C * E_gradient
    if optimization == 'initial':
        theta -= learning_rate * gradient
    elif optimization == 'NAG':
```

**8. The initialization method of model parameters:**set all parameter into zero

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

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

```
def loss_function(X, Y, theta):
    loss = -1.0 / X.shape[0] * (Y * np.log(sigmoid(X.dot(theta)) ) +  (1 - Y) * np.log(1-sigmoid(X.dot(theta)) ) ) .sum()
    return loss
```

Its derivatives：

```
gradient = 1.0/X_train.shape[0] * np.dot(X_train.transpose(), sigmoid(X_train.dot(theta))-Y_train)
```

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

```
def loss_function(X, Y, theta, C):
    E_loss = 1 - Y * X.dot(theta)
    E_loss[E_loss<0] = 0
    loss = 0.5 * np.dot(theta.transpose(), theta).sum() + C * E_loss.sum()
    return loss/X.shape[0]
```

Its derivatives：

```
gradient = theta + C * E_gradient
```

**10. Experimental results and curve:**

Hyper-parameter selection:

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

```
optimizer_params={
    'M': 0.9,
    'rms_decay': 0.02,
    'epsilon':0.001
}
learning_rate = 0.01
epoch = 400
```

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

```
optimizer_params={
    'M': 0.9,
    'epsilon':0.001
}
learning_rate = 0.02
C = 0.009
epoch = 400
threshold = 0
```

Predicted Results (Best Results):

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

L_NAG = 0.373198362808,L_train = 0.373198362808

L_RMSprop =0.340664761664,L_train = 0.34604996324

L_AdaDelta = 0.289465393327, L_train = 0.297315333076

L_Adam = 0.328599858407,L_train = 0.329528302198

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

L_NAG = 0.00689797403325,L_train = 0.00725080738128

L_RMSprop =0.00790272455593,L_train = 0.0076196356937

L_AdaDelta = 0.00747940518307, L_train = 0.00735721642494

L_Adam =0.00731326475838,L_train =0.00742805693266

**11.Results analysis:** For Logistic Regression and Stochastic Gradient Descent, use AdaDelta is a little worse than other methods,but it is the best methods for linear classification.

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

Similarities: Both of them are used to predict regression.

Differences: The output of linear regression are continuous values, but in logistic regression ,Y is only 0 or 1

**13. Summary:**

In this experiment, I further understood the problems of logistic

regression, linear classification and stochastic gradient descent, and learned the optimization algorithm through looking up the data online.
I learned more about machine learning through this experiment and know more about the beauty of it !