

**The Experiment Report of**

***Machine Learning***

**College Software College**

**Subject Software Engineering**

**Members**  **Xuanyi Zhang**

**Student ID 201530613658**

**E-mail 839921303@qq.com**

**Tutor**  **Qingyao Wu**

**Date submitted** **2017.12.07**

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

**2. Time:** 2017/12/02

**3. Reporter:**Xuanyi Zhang

**4. Purposes:**

(1) Further understand of linear regression and gradient descent.

(2) Conduct some experiments under small scale dataset.

(3) Realize the process of optimization and adjusting parameters.

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

Linear Regression uses [Housing](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html" \l "housing" \t "https://www.zybuluo.com/chenyaofo/note/_blank) in [LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/" \t "https://www.zybuluo.com/chenyaofo/note/_blank), including 506 samples and each sample has 13 features. You are expected to download scaled edition. After downloading, you are supposed to divide it into training set, validation set.   
 Linear classification uses [australian](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html" \l "australian" \t "https://www.zybuluo.com/chenyaofo/note/_blank) in [LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/" \t "https://www.zybuluo.com/chenyaofo/note/_blank), including 690 samples and each sample has 14 features. You are expected to download scaled edition. After downloading, you are supposed to divide it into training set, validation set.

1. **Experimental steps:**

Linear Regression and Gradient Descent:

1. Load the experiment data. You can use [load\_svmlight\_file](http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_svmlight_file.html" \t "https://www.zybuluo.com/chenyaofo/note/_blank) function in sklearn library.
2. Divide dataset. You should divide dataset into training set and validation set using [train\_test\_split](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html" \t "https://www.zybuluo.com/chenyaofo/note/_blank) function. Test set is not required in this experiment.
3. Initialize linear model parameters. You can choose to set all parameter into zero, initialize it randomly or with normal distribution.
4. Choose loss function and derivation: Find more detail in PPT.
5. Calculate gradient G  toward loss function from all samples.
6. Denote the opposite direction of gradient G as D.
7. Update model: 1512389151(1)is learning rate, a hyper-parameter that we can adjust.
8. Get the loss 1512389177(1) under the training set and 1512389201(1) by validating under validation set.
9. Repeat step 5 to 8 for several times, and drawing graph of  1512389177(1)  as well as 1512389201(1)  with the number of iterations.

Linear Classification and Gradient Descent:

1. Load the experiment data.
2. Divide dataset into training set and validation set.
3. Initialize SVM model parameters. You can choose to set all parameter into zero, initialize it randomly or with normal distribution.
4. Choose loss function and derivation: Find more detail in PPT.
5. Calculate gradient G toward loss function from all samples.
6. Denote the opposite direction of gradient G as D.
7. Update model: 1512389151(1) is learning rate, a hyper-parameter that we can adjust.
8. Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Get the loss 1512389177(1) under the training set and  by validating 1512389201(1) under validation set.
9. Repeat step 5 to 8 for several times, and drawing graph of  1512389177(1) as well as 1512389201(1) with the number of iterations.
10. **Code:**

Linear Regression and Gradient Descent:

from sklearn.model\_selection import train\_test\_split

from sklearn.externals.joblib import Memory

from sklearn.datasets import load\_svmlight\_file

import matplotlib.pyplot as plt

import numpy as np

#读取数据

mem = Memory("./mycache")

@mem.cache

def get\_data():

data = load\_svmlight\_file("F:/大三上/机器学习/实验一/housing\_scale.txt")

return data[0], data[1]

X,y= get\_data()

X = X.toarray()

#切分数据集

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

#初始化

param = np.zeros(X\_train.shape[1])

#选择Loss函数及对其求导

def L(X,y,param):

return (( y - np.dot(X,param) ) \*\* 2).sum() / 2

def Gra(X,y,param):

return np.dot(X.T, np.dot(X, param) - y)

L\_train = []

L\_validation = []

learning\_rate = 0.00009

num = 30

#求得所有样本对Loss函数的梯度的平均值

for i in range(num):

G = Gra(X,y,param)

#更新模型参数

param = param - learning\_rate \* G

#在训练集上测试并得到Loss函数值

L\_train.append(L(X\_train,y\_train,param))

L\_validation.append(L(X\_test,y\_test,param))

#画出随迭代次数的变化图

plt.title('Loss Curve')

plt.plot(range(num),L\_train,label="Train Loss")

plt.plot(range(num),L\_validation,label="Validation Loss")

plt.xlabel("Iteration")

plt.ylabel("Loss")

plt.legend()

plt.show()

(Fill in the contents of 8-12 respectively for linear regression and linear classification)

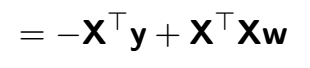
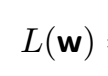
1. **Selection of validation (hold-out, cross-validation, k-folds cross-validation, etc.):**

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

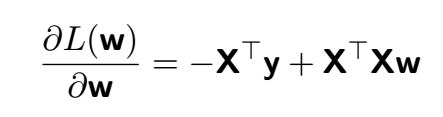
Set all parameter into zero

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

Loss function:

****

Its derivative:

****

**11. Experimental results and curve:**

## Hyper-parameter selection (η, epoch, etc.):

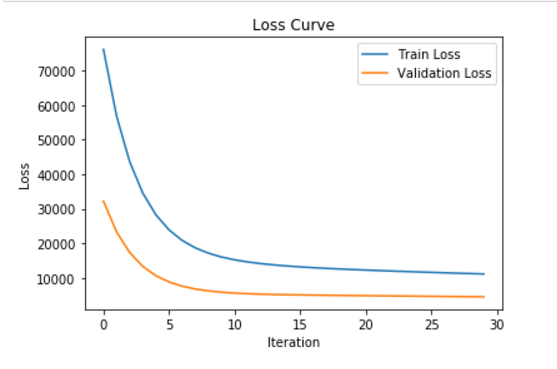
learning-rate = 0.00009

epoch = 30

## Assessment Results (based on selected validation):

## Predicted Results (Best Results):

## Loss curve:



**12. Results analysis:**

From the Loss Curve Chart, we can conclude that if learning\_rate=0.00009, the result is relatively good. As the number of iterations increases, the loss becomes lower. The trend at first is steep and it begins to be gentle.

Linear Classification and Gradient Descent:

from sklearn.model\_selection import train\_test\_split

from sklearn.datasets import load\_svmlight\_file

from sklearn.externals.joblib import Memory

from matplotlib import pyplot as plt

import numpy as np

#读取数据

mem = Memory("./mycache")

@mem.cache

def get\_data():

data = load\_svmlight\_file("F:/大三上/机器学习/实验一/australian\_scale.txt")

return data[0], data[1]

X,y= get\_data()

X = X.toarray()

#在训练集前添加一列1

X = np.c\_[np.ones(len(X)), X]

#切分数据集

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

#初始化

W = np.zeros(X\_train.shape[1])

#选择Loss函数及对其求导

def hinge\_loss(x):

return max(0,1 - x)

def L(X,y,W,C):

return (W \*\* 2).sum() / 2 + C \* sum(map(hinge\_loss, y \* (np.dot(X, W))))

def Gra(X,y,W,C):

temp = np.zeros(len(W))

for i in range(X.shape[0]):

if y[i] \* np.dot(X[i,:], W) < 1:

temp += -y[i] \* X[i,:]

return W + C \* temp

L\_train = []

L\_validation = []

num=150

C=0.9

learning\_rate = 0.0001

#求得所有样本对Loss函数的梯度的平均值

for i in range(num):

G = Gra(X,y,W,C)

#更新模型参数

W = W - learning\_rate \* G

#在训练集上测试并得到Loss函数值

L\_train.append(L(X\_train,y\_train,W,C))

L\_validation.append(L(X\_test,y\_test,W,C))

#画出随迭代次数的变化图

plt.title('Loss Curve')

plt.plot(range(num),L\_train,label="Train Loss")

plt.plot(range(num),L\_validation,label="Validation Loss")

plt.xlabel("Iteration")

plt.ylabel("Loss")

plt.legend()

plt.show()

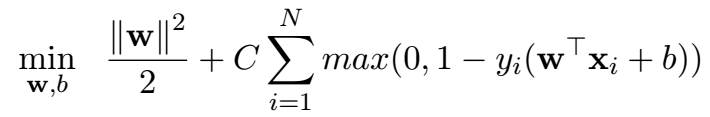
**8. Selection of validation (hold-out, cross-validation, k-folds cross-validation, etc.):**

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

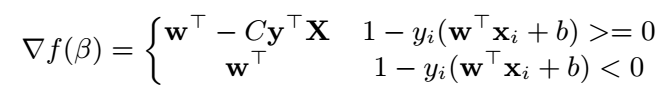
Set all parameter into zero

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

loss function:

****

Its derivative:



**11. Experimental results and curve:**

## Hyper-parameter selection (η, epoch, etc.):

learning\_rate = 0.0001

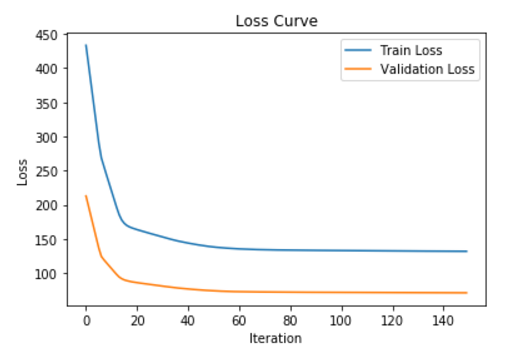
C = 0.9

epoch = 150

## Assessment Results (based on selected validation):

## Predicted Results (Best Results):

## Loss curve:



1. **Results analysis:**

From the Loss Curve Chart, we can conclude that if learning\_rate=0.0001, and C = 0.9, the result is relatively good. As the number of iterations increases, the loss becomes lower. The trend at first is steep and it begins to be gentle.

1. **Similarities and differences between linear regression and linear classification:**

Similarities: They are both the fitting of models.

Differences:The classification is more like some x in a certain region corresponds to y. The model of the regression is more inclined to x in a very small area, or generally one x corresponds to a y.

**14. Summary:**

Linear regression is often used to analysis the relationship between the variables x and y. The y value of the classification problem is often a discrete number. When modeling a problem, be sure to consider good needs and make your model better correspond to realistic problems.