华南理工大学

South China University of Technology

《机器学习》课程实验报告

学　　院　软件学院

专　　业　软件工程

组　　员　袁振宇

学　　号　201530613542

邮　　箱　ken1024448582@qq.com

指导教师　吴庆耀

提交日期　2017 年 12 月 15 日

1. **实验题目：逻辑回归、线性分类与随机梯度下降**
2. **实验时间：**2017 年 12 月 15 日
3. **报告人：袁振宇**
4. **实验目的：理解、模拟逻辑回归、线性分类**
5. **数据集以及数据分析：a9a 数据集，采用随机梯度下降分析**
6. **实验步骤：**

逻辑回归与随机梯度下降

1. 读取实验训练集和验证集。
2. 逻辑回归模型参数初始化，可以考虑全零初始化，随机初始化或者正态分布初始化。
3. 选择 Loss 函数及对其求导，过程详见课件 ppt。
4. 求得**部分样本**对 Loss 函数的梯度。
5. **使用不同的优化方法更新模型参数（NAG，RMSProp，AdaDelta 和 Adam）**。
6. 选择合适的阈值，将验证集中计算结果**大于阈值的标记为正类，反之为负类**。在验证集上测试并得到不同优化方法的 Loss 函数值，，和。
7. 重复步骤 4-6 若干次，**画出，，和随迭代次数的变化图**。

线性分类与随机梯度下降

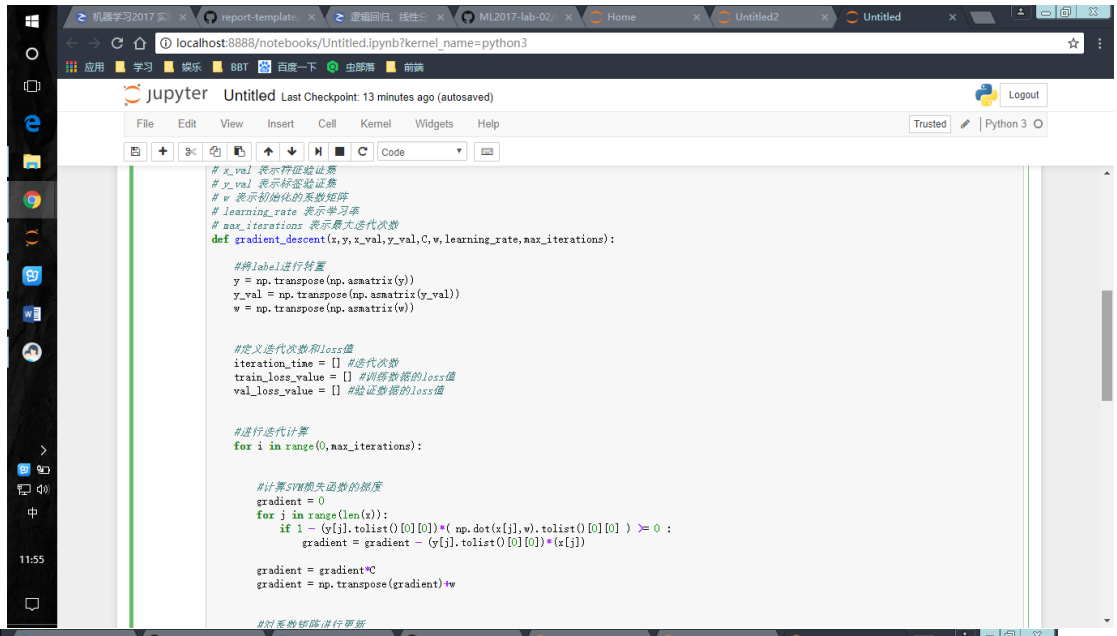
1. 读取实验训练集和验证集。
2. 支持向量机模型参数初始化，可以考虑全零初始化，随机初始化或者正态分布初始化。
3. 选择 Loss 函数及对其求导，过程详见课件 ppt。

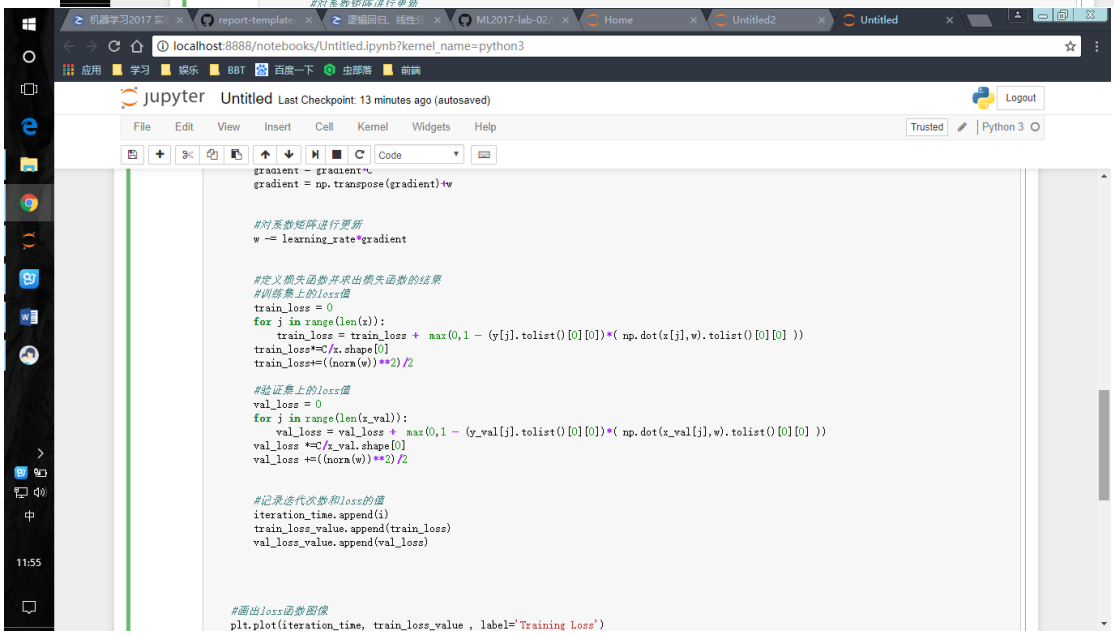
4. 求得**部分样本**对 *Loss* 函数的梯度。
5. **使用不同的优化方法更新模型参数（*NAG*，*RMSProp*，*AdaDelta* 和 *Adam*）**。
6. 选择合适的阈值，将验证集中计算结果**大于阈值的标记为正类，反之为负类**。在验证集上测试并得到不同优化方法的 *Loss* 函数值，，和。
7. 重复步骤 *4-6* 若干次，**画出，，和随迭代次数的变化图**。

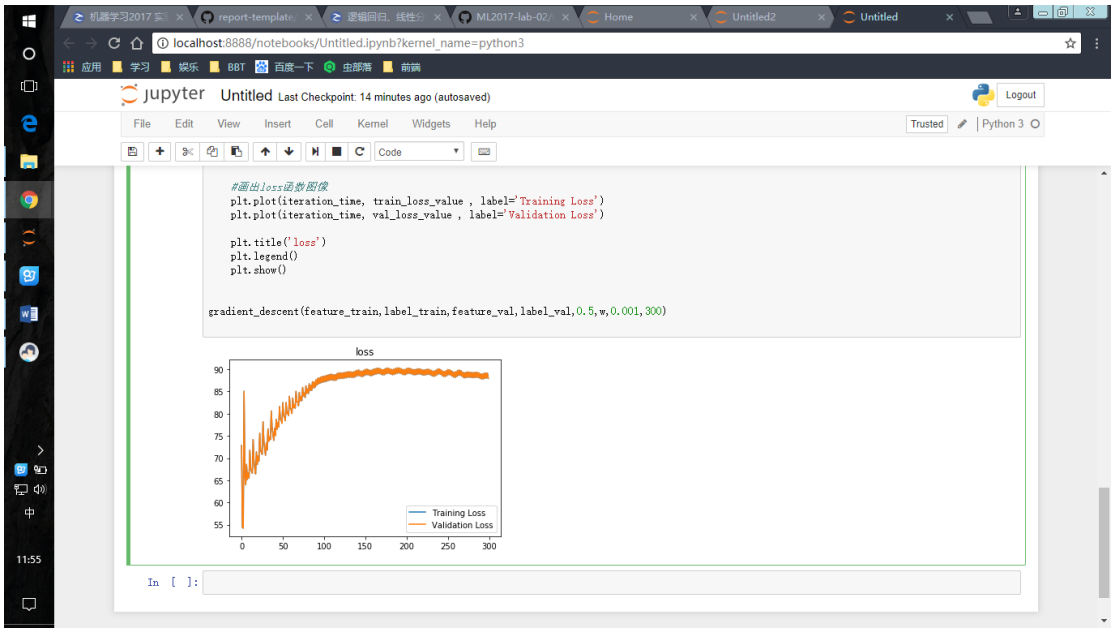
7. 代码内容:

线性回归：

In [10]:
```
#导入相应包
from sklearn.datasets import load_svmlight_file
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg.misc import norm

#读取相应数据
data,label=load_svmlight_file('a9a.t')
data = data.todense()

#切分数据集
feature_train,feature_val,label_train,label_val = train_test_split(data,label,test_size=0.25, random_state=42)
#向矩阵中加入偏置项的函数
def add_bias(matrix):
    bias = []
    #构造偏置项list
    for i in range(matrix.shape[0]):
        bias.append(1)
    #向矩阵中加入一列偏置项
    matrix = np.column_stack((matrix,bias))
    return matrix

#在特征集合中加入偏置项
feature_train = add_bias(feature_train)
feature_val = add_bias(feature_val)
#构造并初始化系数矩阵
w = np.ones((1,123))
#梯度下降算法函数
# x 表示特征训练集
# y 表示标签训练集
```


localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: 13 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Code
# x_val 表示特征验证集
# y_val 表示标签验证集
# w 表示初始化的系数矩阵
# learning_rate 表示学习率
# max_iterations 表示最大迭代次数
def gradient_descent(x,y,x_val,y_val,C,w,learning_rate,max_iterations):
#将label进行转置
y = np.transpose(np.asmatrix(y))
y_val = np.transpose(np.asmatrix(y_val))
w = np.transpose(np.asmatrix(w))
#定义迭代次数和loss值
iteration_time = [] #迭代次数
train_loss_value = [] #训练数据的loss值
val_loss_value = [] #验证数据的loss值
#进行迭代计算
for i in range(0,max_iterations):
#计算SVM损失函数的梯度
gradient = 0
for j in range(len(x)):
if 1 - (y[j].tolist()[0][0])*(np.dot(x[j],w).tolist()[0][0]) >= 0 :
gradient = gradient - (y[j].tolist()[0][0])*(x[j])
gradient = gradient*C
gradient = np.transpose(gradient)+w
#对系数矩阵进行更新


localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: 13 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Code
gradient = gradient*C
gradient = np.transpose(gradient)+w
#对系数矩阵进行更新
w -= learning_rate*gradient
#定义损失函数并求出损失函数的结果
#训练集上的loss值
train_loss = 0
for j in range(len(x)):
train_loss = train_loss + max(0,1 - (y[j].tolist()[0][0])*(np.dot(x[j],w).tolist()[0][0]))
train_loss*=C/x.shape[0]
train_loss+=((norm(w))**2)/2
#验证集上的loss值
val_loss = 0
for j in range(len(x_val)):
val_loss = val_loss + max(0,1 - (y_val[j].tolist()[0][0])*(np.dot(x_val[j],w).tolist()[0][0]))
val_loss *=C/x_val.shape[0]
val_loss +=((norm(w))**2)/2
#记录迭代次数和loss的值
iteration_time.append(i)
train_loss_value.append(train_loss)
val_loss_value.append(val_loss)
#画出loss函数图像
plt.plot(iteration_time, train_loss_value , label='Training Loss')



8. 模型参数的初始化方法：随机初始化

9.选择的 loss 函数及其导数:

逻辑回归 loss 函数

$$\cos t(y, h_\theta(x)) = \sum_{i=1}^{m}(-y_i \log \frac{1}{1+e^{-\theta^T x}} - (1-y_i)\log(1-\frac{1}{1+e^{-\theta^T x}}))$$

线性回归 loss 函数

$$\ell(\theta) = 1/2\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 = 1/2\sum_{i=1}^{m}(\theta^T x^{(i)} - y^{(i)})^2$$

10.实验结果和曲线图:（各种梯度下降方式分别填写此项）

超参数选择：

预测结果（最佳结果）:

loss 曲线图：

11.实验结果分析:

12.对比逻辑回归和线性分类的异同点：

线性回归：目的是进行预测，对 x 有对应的某个预测值，采用拟合函数，参

数计算方式是最小二乘法

逻辑回归：实质上是二分类，预测值是{0，1}，采用预测函数，参数计算方式是最大似然估计

共同点：都是对数据进行拟合，通过求参找出预测函数。

13.实验总结：

虽然对这两个知识点有一定理解，但是求参过程很模糊。