

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**   **黄登**

**学 号 201530211588**

**邮 箱 384813529@qq.com**

**指导教师**  **吴庆耀**

**提交日期** **2017年12月7日**

## 1. 实验题目: 线性回归、线性分类与梯度下降

## 2. 实验时间：2017年12月 7日

## 3. 报告人: 黄登

## 4. 实验目的:

进一步理解线性回归和梯度下降的原理。

在小规模数据集上实践。

体会优化和调参的过程。

## 5. 数据集以及数据分析：

线性回归使用的是LIBSVM Data中的Housing数据，包含506个样本，每个样本有13个属性。

线性分类使用的是LIBSVM Data中的australian数据，包含690个样本，每个样本有14 个属性。

## 6. 实验步骤:

*线性回归和梯度下降*

1. 读取实验数据，使用sklearn库的[load\_svmlight\_file](http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_svmlight_file.html)函数读取数据。
2. 将数据集切分为训练集和验证集，本次实验不切分测试集。使用[train\_test\_split](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html)函数切分数据集。
3. 线性模型参数初始化，可以考虑全零初始化，随机初始化或者正态分布初始化。
4. 选择Loss函数及对其求导，过程详见课件ppt。
5. 求得**所有样本**对Loss函数的梯度G。
6. 取梯度的负方向，记为D。
7. 更新模型参数，Wt = Wt-1 + ηD。η为学习率，是人为调整的超参数。
8. 在训练集上测试并得到Loss函数值，在验证集上测试并得到Loss函数值。
9. 重复步骤5-8若干次，**画出和随迭代次数的变化图**。

*线性分类和梯度下降*

1. 读取实验数据，使用sklearn库的[load\_svmlight\_file](http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_svmlight_file.html)函数读取数据。
2. 将数据集切分为训练集和验证集，本次实验不切分测试集。使用[train\_test\_split](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html)函数切分数据集。
3. 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。
4. 选择Loss函数及对其求导，过程详见课件ppt。
5. 求得**所有样本**对Loss函数的梯度G。
6. 取梯度的负方向，记为D。
7. 更新模型参数，Wt = Wt-1 + ηD。η为学习率，是人为调整的超参数。
8. 选择合适的阈值，将计算结果**大于阈值的标记为正类，反之为负类**。在训练集上测试并得到Loss函数值，在验证集上测试并得到Loss函数值。
9. 重复步骤5-8若干次，**画出和随迭代次数的变化图**。

## 7. 代码内容:

Linear Regression

|  |
| --- |
| def gradient\_descent(x, y, w=None, alpha=0.001, iterations=50000):  x0 = np.ones((len(y), 1))  x = hstack([x0, x])  if w == None:  w = np.zeros(x.shape[1])    loss\_history = [0] \* iterations  m = len(y)    for iteration in range(iterations):  h = x.dot(w)  loss = h - y    gradient = x.T.dot(loss) / m  w = w - alpha \* gradient  loss = loss\_fn(x, y, w)  loss\_history[iteration] = loss    return w, loss\_history |

LinearSVM Classification

|  |
| --- |
| class LinearClassifier:  def \_\_init\_\_(self):  self.w = None  self.le = LabelEncoder()    def train(self, X, y, learning\_rate=1e-3, reg=1e-5, num\_iters=2000, batch\_size=200, verbose=False):  num\_train, dim = X.shape  y = self.le.fit\_transform(y)    num\_classes = int(np.max(y)) + 1  if self.w is None:  print(dim, num\_classes)  self.w = 0.001 \* np.random.randn(dim, num\_classes)    loss\_history = []  for i in range(num\_iters):  batch\_idx = np.random.choice(num\_train, batch\_size, replace=True)  X\_batch = X[batch\_idx]  y\_batch = y[batch\_idx]    loss, grad = self.loss(X\_batch, y\_batch, reg)    loss\_history.append(loss)    self.w += -1 \* learning\_rate \* grad    if verbose and i % 100 == 0:  print('iteration %d / %d: loss %f' % (i, num\_iters, loss))    return loss\_history    def predict(self, X):  y\_pred = np.zeros(X.shape[1])  scores = X.dot(self.w)  y\_pred = np.argmax(scores, axis=1)  y\_pred = self.le.inverse\_transform(y\_pred)  return y\_pred |

## 8. 选择的评估方法（留出法，交叉验证，k折交叉验证等）:

Regression:

1. RMSE(root-mean-square error, 均方根误差)
2. R2\_score

Classification:

1. Accuracy score

## 9. 模型参数的初始化方法:

Regression: 全零初始化

Classification: 随机初始化

## 选择的loss函数及其导数:

Regression:

Classification:

## 11.实验结果和曲线图:

## 超参数选择（η,epoch等）：

Regression: η = 1e-5

Classification: η = 1e-3

## 评估结果（根据选择的评估方法）：

Regression:

调用sklearn.linear\_model中的 LinearRegression对训练集训练，得到

rmse = 5.01179783065

r2\_score = 0.700417094174

本模型的结果为

rmse = 4.75843882965

r2\_score = 0.729940833545

其中，rmse更小，说明拟合效果更好

Classification：

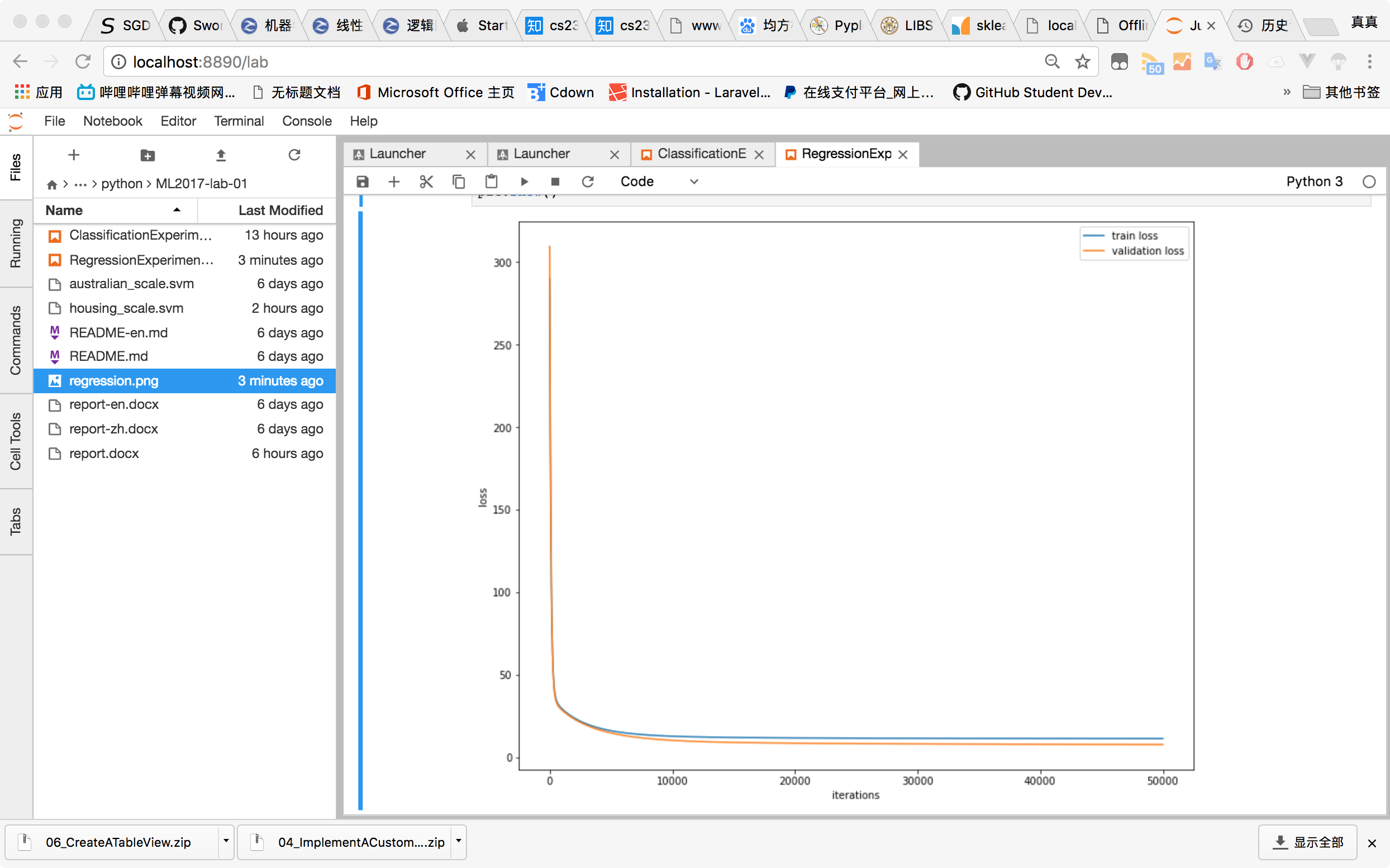
调用sklearn.svm中的 LinearSVC进行预测，准确率在0.86413043478260865左右（受训练集划分影响），本模型也能达到0.85688405797101452，基本准确。

## 预测结果（最佳结果）：

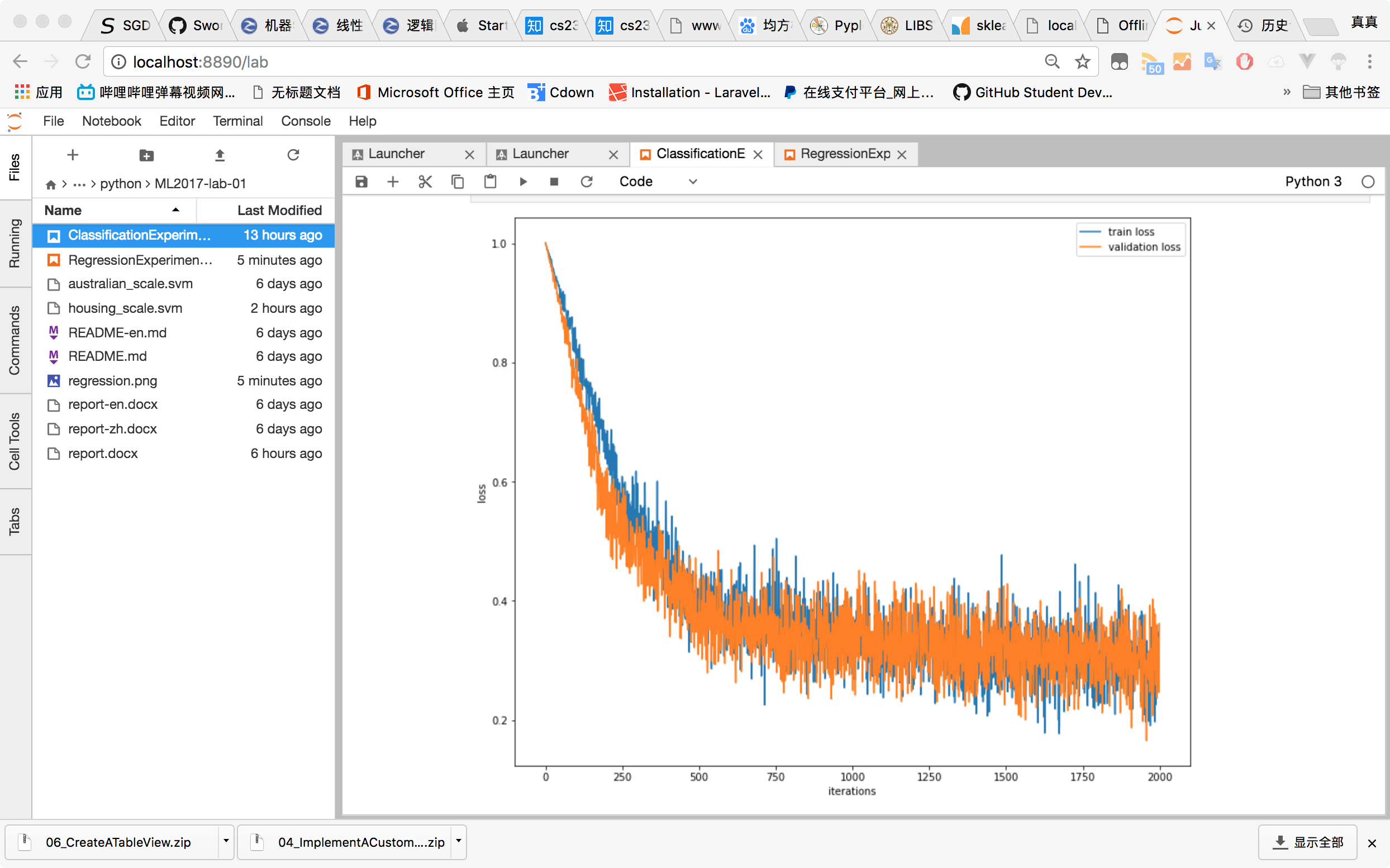
Classification：accuracy = 0.85688405797101452

## loss曲线图：

Regression:



Classification:



## 实验结果分析:

与sklearn中的model的结果相近。

## 对比线性回归和线性分类的异同点：

相同点：

1. 使用线性方程
2. 用到梯度下降来寻找最优参数

不同点：

1. 回归是拟合的过程，分类是寻找分割平面的过程
2. Loss函数不同

## 14.实验总结：

对线性回归和svm有了更深入的认识，锻炼了动手能力。复习了大一所学的矩阵运算知识。对自己平时调用的库有了更深层次的了解。