题目　Training Classifiers via SGD

姓名 骆克云 学号 MG1633052 邮箱 streamer.ky@foxmail.com 联系方式18115128082

(南京大学 计算机科学与技术系, 南京　210093)

1　实现细节

1.1　　读取文件，返回数据和标签（projectutil.py）

使用 Python 中的 Pandas 库读取 CSV 文件：将每一行数据按逗号分隔，最后一个作为标签，前面的作为数据

```
def get_sgd_data(name="dataset1-a9a", types="train"):
    """
    获取数据集
    :param name:数据名称：dataset1-a9a/covtype
    :param type:数据类型：train/test
    :return:数据，标签
    """
    file_train = project_dir + os.sep + "data/SGD/" + name + "-training.txt"
    file_test = project_dir + os.sep + "data/SGD/" + name + "-testing.txt"
    file = file_train if types == "train" else file_test
    data = []
    label = []
    with open(file, "r", encoding="utf-8") as f:
        for line in f:
            line = line.split(",")
            data.append(line[:-1])
            label.append(line[-1].strip("\n"))
    return data, label
```

1.2　　最大似然逻辑回归算法实现(assignment4/logisticregression.py)

SGD 的损失函数使用 Logistic Regression，正则项为 L1 范数，最大迭代次数为 5 次(可在初始化时调整)，正则项系数 λ 值为 0.001(可在初始化时调整)，梯度函数为：$J(\beta) = (\frac{1}{1+e^{-y_i\beta^T x_i}} - 1)y_i x_i + \lambda sign(\beta)$，权值更新函数为：$\beta^{k+1} = \beta^k - \alpha \cdot J(\beta^k)$，目标函数为：$\min_{\beta} \frac{1}{N}\sum_{i=1}^{N}\{\log(1+\exp^{-y_i\beta^T x_i})\} + \lambda\|\beta\|_1$，下降步长α动态调整， 权值β梯度下降。

相关算法如下：

1）初始化(__init__)：读取文件，包括文件的训练集和测试集的数据和标签(self.train, self.train_label, self.test, self.test_label)，确定迭代次数(self.max_iter=5), 正则化 λ 值，并对训练集和测试集数据进行收缩标准化，同时在训练集和测试集的第一列插入全 1 列， 权值β初始化为[0,1,1,…,1].

2）收缩标准化(all_scale,scale)：计算训练集数据的均值和标准差，若标准差为 0 则置为 1，然后对训练集数据和测试集数据中的每一个数据减去均值，然后除以标准差；

3）Sigmod 函数(def sigmod(self, x))：计算给定值的 Sigmoid 函数值;

4）梯度下降(def decent_vector(self, index, beta))：计算梯度 J(β)的值；

5）符号函数(def sign_vector(self, w))：对于向量中的每一个分量，若其大于零则置为 1，小于零置为-1，否则置为 0；

6）Sgd 方法(def sgd(self))：梯度下降中的主算法，引入一个步长基准 alpha0 = 8.0 / (len(self.train)), 在每次循环中，将训练数据索引打乱，对索引中的每一个值，步长α：alpha = 1.0 / (4.0 + i + j) + alpha0，权值β：beta = beta - alpha * (self.decent_vector(i, beta) + self.lambda_value * self.sign_vector(beta))，并保存到历史记录中；

7）错误率绘图统计(def plot_score(self, history_beta))：将历史权值β记录划分为 100 等分，取每份中的第一个数据进行错误率统计，绘制图表；

8）目标函数绘图统计(def plot_objective_function(self, history_beta))：将历史权值β记录划分为 100 等分，取每份中的第一个数据进行目标函数值统计，绘制图表；

9）标签分类(def score(self, prob)):如果概率值大于或等于 0.5，返回 1，否则返回-1；

10）错误率统计(def score_datasets(self, beta, datas, types))：将权值函数乘上测试集数据，运行上面的标签分类函数，计算与标准标签相同的个数，进而计算出准确率；

11）目标函数值计算(def objective_function(self, beta, data, label, lambda_value))：将权值β带入目标函数中，计算出目标函数值；

12）运行接口(def run(self)):运行 SGD 算法，得到权值β的历史记录值，运行错误率绘图统计和目标函数绘图统计方法，得到趋势图。

伪代码：

$\beta = [0, 1, 1, \dots, 1]$
$sigmod(x) = 1.0 / (1 + e^{-x})$
$J(\beta, i) = (sigmod(y_i \beta^T x_i) - 1) y_i x_i + \lambda sign(\beta)$
$\lambda = 0.001$

```
max_iter=5
indexes = np.array(len(train))
alpha0 = 8.0 / (len(train))
count = 0

repeat max_iter{
count += 1
np.random.shuffle(indexes)
for i in indexes:
    alpha = 1.0 / (4.0 + i + count) + alpha0
    β = β − alpha ∗ J(β, i)
}
```

1.3 Ridge Regression岭回归算法实现（assignment4/spectralclustering.py）

SGD 的损失函数使用 Ridge Regression, 正则项为 L2 范数，最大迭代次数为 5 次(可在初始化时调整)，正则项系数 λ 值为 0.3(可在初始化时调整)，梯度函数为：$J(\beta) = -x_i(y_i - \beta^T x_i) + \lambda\beta$,权值更新函数为：$\beta^{k+1} = \beta^k - \alpha \cdot J(\beta^k)$ ，目标函数为：$\min_{\beta} \frac{1}{N}\sum_{i=1}^{N}(y_i - \beta^T x_i)^2 + \lambda\|\beta\|_2^2$ 下降步长α动态调整， 权值β梯度下降。

相关算法如下：

1）初始化(__init__)：读取文件，包括文件的训练集和测试集的数据和标签(self.train, self.train_label, self.test, self.test_label)，确定迭代次数(self.max_iter=5), 正则化 λ 值，并对训练集和测试集数据进行收缩标准化，同时在训练集和测试集的第一列插入全 1 列， 权值β初始化为[0,1,1,…,1].

2）收缩标准化(all_scale,scale)：计算训练集数据的均值和标准差，若标准差为 0 则置为 1，然后对训练集数据和测试集数据中的每一个数据减去均值，然后除以标准差；

3）Sgd 方法(def sgd(self))：梯度下降中的主算法，引入一个迭代计数器 count = 0, 在每次循环中，将训练数据索引打乱，对索引中的每一个值，count += 1，步长α：alpha = 0.0008 / np.sqrt(count) + 0.0001，权值β：beta += alpha * (X[i]*(y[i] - beta.dot(X[i].T)) - self.lambda_value * beta)，并保存到历史记录中；

4）错误率绘图统计(def plot_score(self, history_beta))：将历史权值β记录划分为 100 等分，取每份中的第一个数据进行错误率统计，绘制图表；

5）目标函数绘图统计(def plot_objective_function(self, history_beta))：将历史权值β记录划分为 100 等分，取每份中的第一个数据进行目标函数值统计，绘制图表；

6）标签分类(def score(self, prob)):如果权值函数乘上测试集数据大于或等于 0，返回 1，否则返回-1；

7）错误率统计(def score_datasets(self, beta, datas, types))：对每个测试集中的数据，运行上面的标签分类函数，计算与标准标签相同的个数，进而计算出准确率；

8）目标函数值计算(def objective_function(self, beta, data, label, lambda_value))：将权值β带入目标函数中，计算出目标函数值；

9）运行接口(def run(self)):运行 SGD 算法，得到权值β的历史记录值，运行错误率绘图统计和目标函数绘图统计方法，得到趋势图。

伪代码：

$\beta = [0, 1, 1, \dots, 1]$
$J(\beta, i) = -x_i(y_i - \beta^T x_i) + \lambda\beta$
$\lambda = 0.3$

```
max_iter=5
indexes = np.array(len(train))
count = 0

repeat max_iter{
np.random.shuffle(indexes)
count += 1
for i in indexes:
    alpha = 0.0008 / np.sqrt(count) + 0.0001
    β = β − alpha ∗ J(β, i)
}
```

1.4 运行

对于每个文件（"dataset1-a9a", "covtype"），分别运行 LRWithSGD,RRWithSGD,SGDWithSklearn(sklearn 验证)得出结果。

```
def run():
  for file in ["dataset1-a9a", "covtype"]: #, "covtype" "dataset1-a9a",
      lr = LRWithSGD(file)
      lr.run()
      rr = RRWithSGD(file)
      rr.run()
      sgd = SGDWithSklearn(file)
      sgd.run()
```

2 结果

2.1 实验设置

数据来源：SGD,包含如下文件：[dataset1-a9a,covtype][--testing.txt,-training.txt]
数据预处理：projectutil.py 中 get_sgd_data(filename)可给定文件名返回数据及标签。
对于逻辑回归和岭回归的参数，都使用默认值。

2.2 实验结果

1. 逻辑回归：训练集/测试集错误率（表取 10 个时间点，图取 100 个时间点绘制成的折线图）

dataset1-a9a

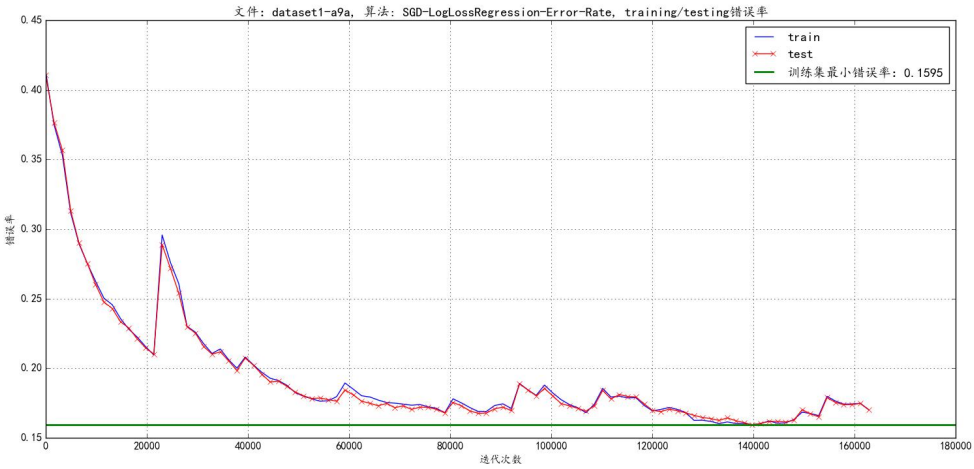
	0.0T	0.1T	0.2T	0.3T	0.4T	0.5T	0.6T	0.7T	0.8T	0.9T
测试集	0.4129	0.228,	0.2106,	0.1825	0.1771	0.1751	0.1879	0.1787	0.1618	0.1631
训练集	0.4104	0.2289	0.2101	0.183	0.173	0.173	0.1854	0.1797	0.1638	0.1626

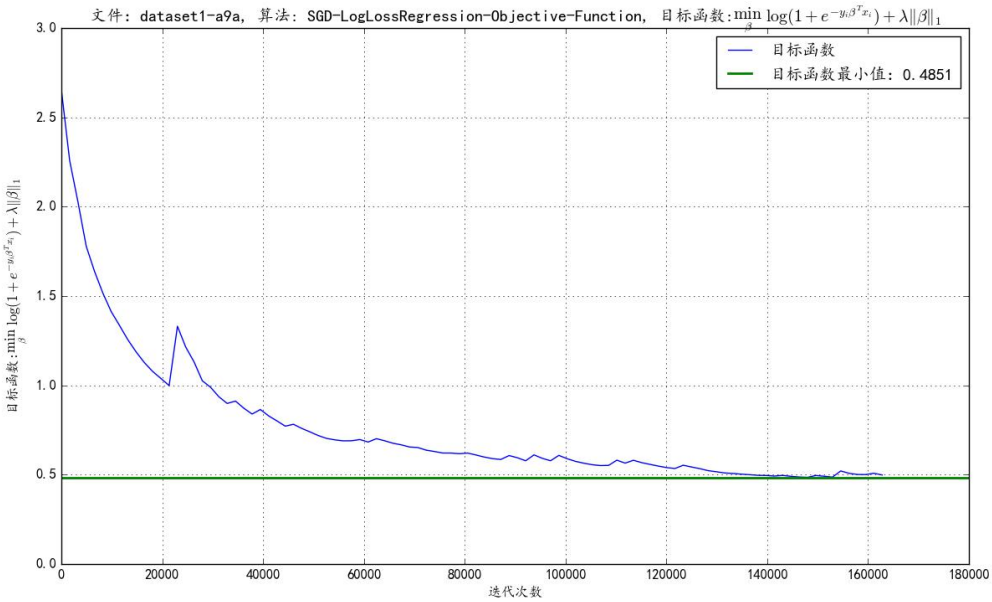
covtype

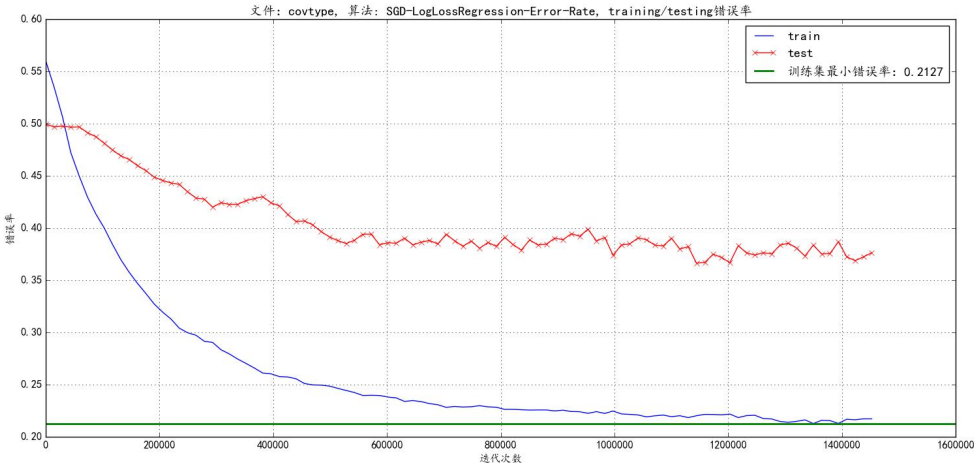
	0.0T	0.1T	0.2T	0.3T	0.4T	0.5T	0.6T	0.7T	0.8T	0.9T
测试集	0.5601	0.3576	0.2904	0.2556	0.2396	0.2285	0.2257	0.2215	0.2214	0.2148
训练集	0.4993	0.4656	0.42	0.4062	0.384	0.3825	0.3845	0.3848	0.3749	0.3807

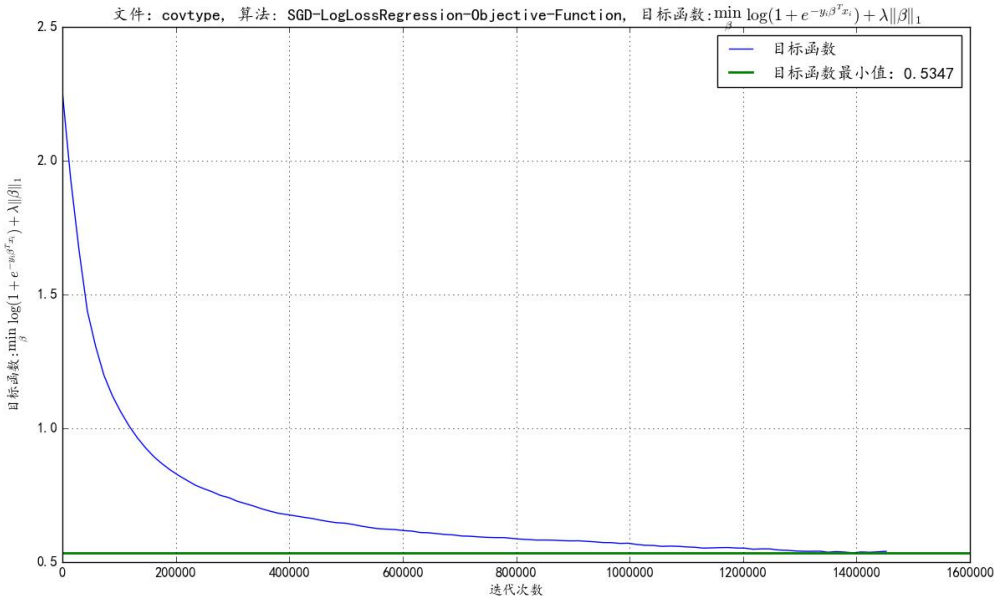
最优：文件：dataset1-a9a， 算法：SGD-LogLossRegression， 训练集准确率： 0.8405 测试集准确率： 0.8407；
文件：covtype， 算法：SGD-LogLossRegression， 训练集准确率： 0.7873 测试集准确率： 0.6164

验证：文件：dataset1-a9a， 算法：sklearn-sgd-logloss， 训练集正确率: 0.8421， 测试集正确率: 0.8424；
文件：covtype， 算法：sklearn-sgd-logloss， 训练集正确率: 0.8103， 测试集正确率: 0.5382









2． 岭回归：训练集/测试集错误率（表取 10 个时间点，图取 100 个时间点绘制成的折线图）

dataset1-a9a

	0.0T	0.1T	0.2T	0.3T	0.4T	0.5T	0.6T	0.7T	0.8T	0.9T
测试集	0.4315	0.2507	0.1767,	0.1623	0.1594	0.1565	0.1548	0.1558	0.1569	0.1568
训练集	0.4292	0.2508	0.1773	0.1626	0.1584	0.155	0.1542	0.154	0.1555	0.1569

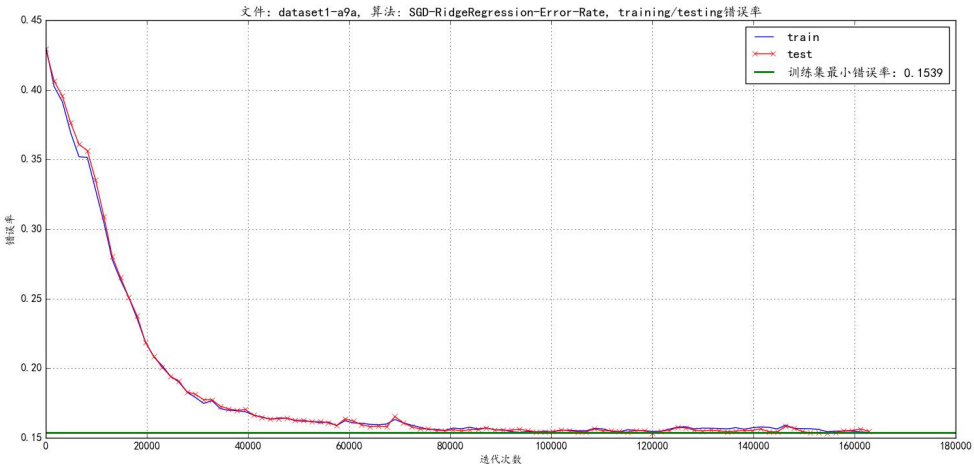
covtype

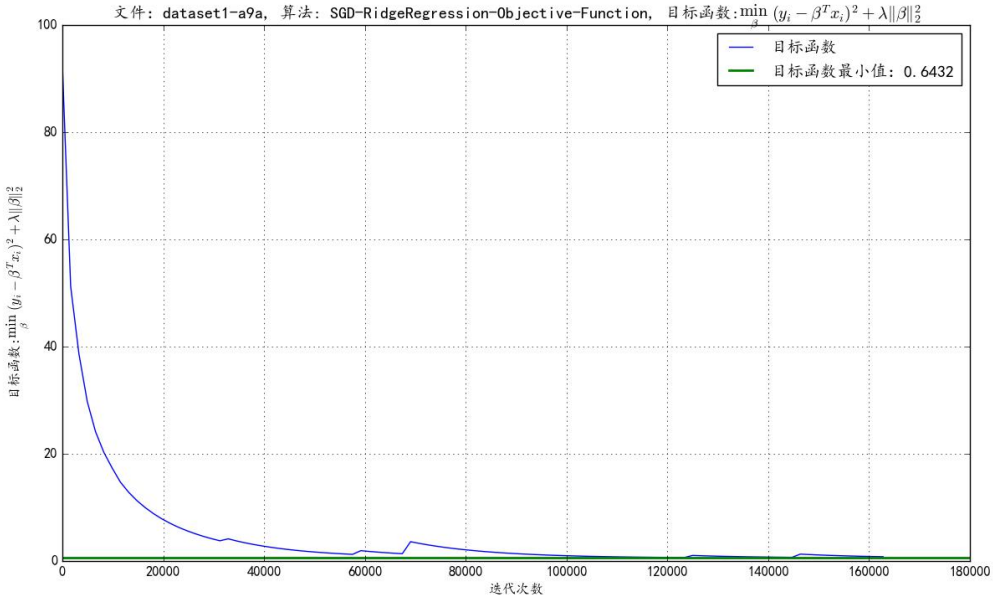
	0.0T	0.1T	0.2T	0.3T	0.4T	0.5T	0.6T	0.7T	0.8T	0.9T
测试集	0.5637	0.2194	0.2175	0.2165	0.215	0.2122	0.2217	0.2152	0.2219	0.2167
训练集	0.5025	0.4495	0.4483	0.4542	0.454	0.4448	0.4551	0.4466	0.4577	0.4357

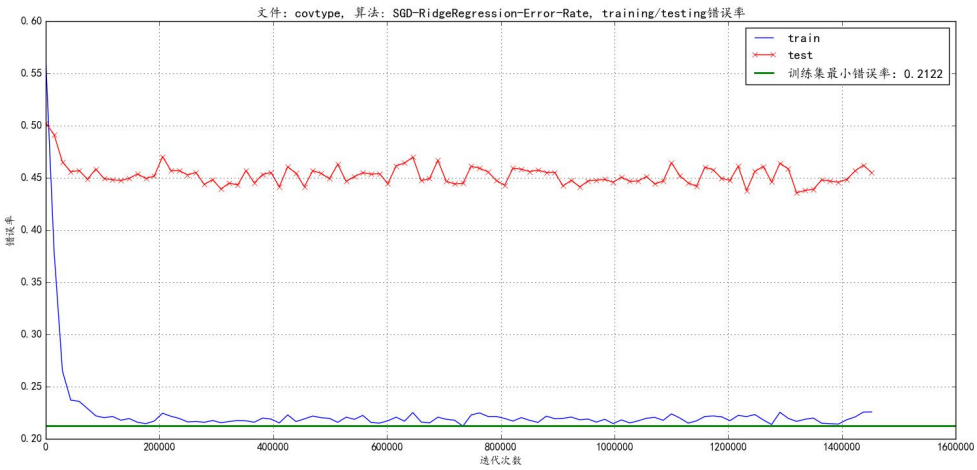
最优：文件：dataset1-a9a， 算法：SGD-RidgeRegression， 训练集准确率： 0.8461 测试集准确率： 0.8439；

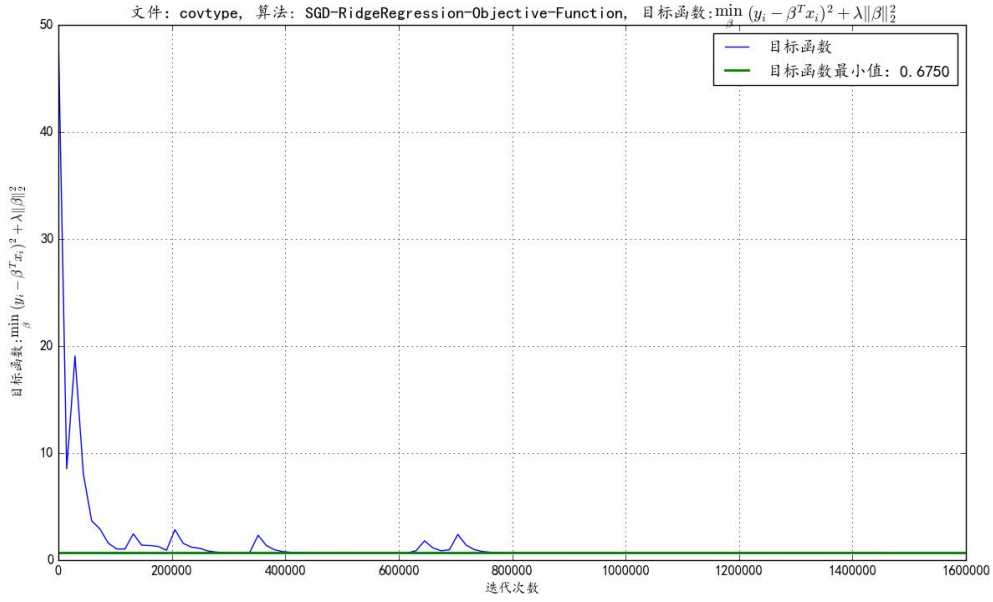
文件：covtype， 算法：SGD-RidgeRegression， 训练集准确率： 0.7878 测试集准确率： 0.5552

验证：文件：dataset1-a9a， 算法：sklearn-sgd-ridgeregression， 训练集正确率: 0.8455， 测试集正确率: 0.8476；
文件：covtype， 算法：sklearn-sgd-ridgeregression， 训练集正确率: 0.8062， 测试集正确率: 0.5505









3．结果反思

参数严重影响到最终分类效果的好坏，需要不停的尝试，选取最佳参数，随机梯度下降准确度下降，并不是全局最优。

注意：

1．最终提交的报告最好保存为 pdf 格式

2．压缩格式为 zip 格式，请勿使用需要安装特定软件才能打开的压缩方式

3．作业的文件夹目录请按照网页要求，代码、结果放在不同子文件夹中。作业网页上给出的数据不需要再次提交