**《机器学习》课程小作业（一）**

**报 告**

|  |  |
| --- | --- |
| **学院** | 深圳国际研究生院 |
| **班级** | 深数据硕212班 |
| **姓名** | 陈文硕 |
| **学号** | 2021214480 |
| **日期** | 2021年10月5日 |

# 预处理

## 标准化

将训练集与测试集分别存放至project/train与project/test文件夹下

将数据进行标准化至区间

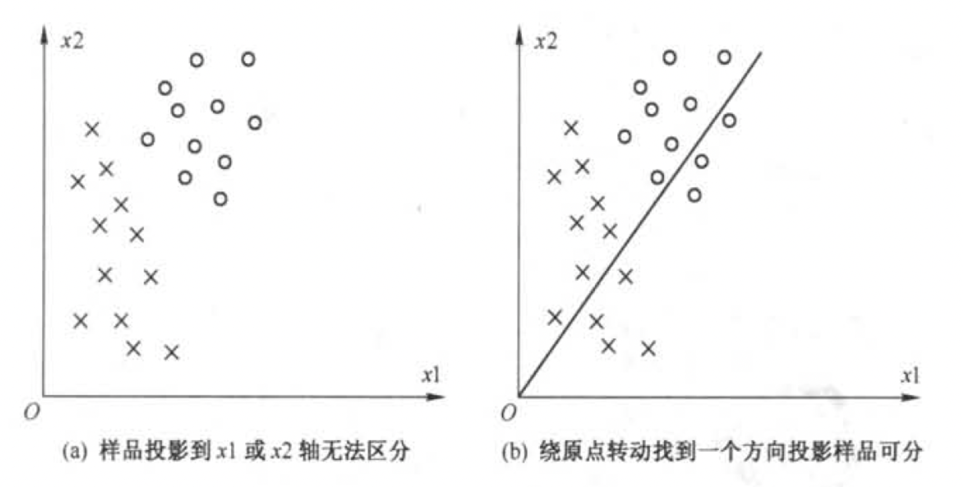
## 主成分分析

在最初的实验中发现，原训练集的样本矩阵不可逆，说明部分数据线性相关。为了让为非奇异矩阵，可以使用PCA的方法进行降维。

我们选取scikit-learn中的PCA工具包（由于作业中并没有考察相关知识，故直接调包），设置主成分个数为‘mle’，即程序会自动根据主成分的方差选取主成分的个数。最终发现，降至96维是合理的。将PCA的降维与之前手动选取特征（选取前37维特征）对比，发现前者在实验中约有2%的准确率提升。由于scikit-learn的PCA库中会自动对数据集进行标准化，故对于标准化，无需再单独编写函数。

# 实验一：FLD

## 用TrainingSet-1计算FLD的判别函数，并利用TestSet-1计算错误率

**Fisher线性判别法**

其核心思想是通过将样本投影到空间中的某个平面，使类内距离尽可能小，类间距离尽可能大。经过推导，优化问题转为

s.t.

再利用拉格朗日乘数法

图 1 FLD图示[1]

其中为两类样本的均值向量。

|  |  |
| --- | --- |
|  |  |
| 79.90% | 79.63% |

但通过上述仿射变换只能确定将样本投影至哪一个平面。还需要确定平移的阈值。

常用的确定方法有三种，经实验，我们采取以下方法效果最优: （用TrainingSet-1和TrainingSet-2作为训练集）将训练集通过归一化以及将数据降至96维后，通过FLD方法，使用TrainingSet-1训练模型，在TestingSet-1测试正确率为77.58%，Error rate为22.42%。

表格 1 不同阈值的正确率

## 总结与思考

在确定阈值的时候，最开始想了好久并不理解为什么和的关系是“负”号，原因是超平面的平移方向与样本点仿射变换的方向相反。

# 实验二：感知机

## 2.1 分别用“Fixed increment rule”和“Variable increment rule”训练模型

经查阅文献[2]，Fixed increment rule和Variable increment rule定义如下：

对于

当为恒定值时，称为*fixed increment rule*，当为变量时，称为*variable increment rule*。

于是我们用BGD (它遵循fixed increment rule)以及Adam (它遵循variable increment rule)分别训练模型

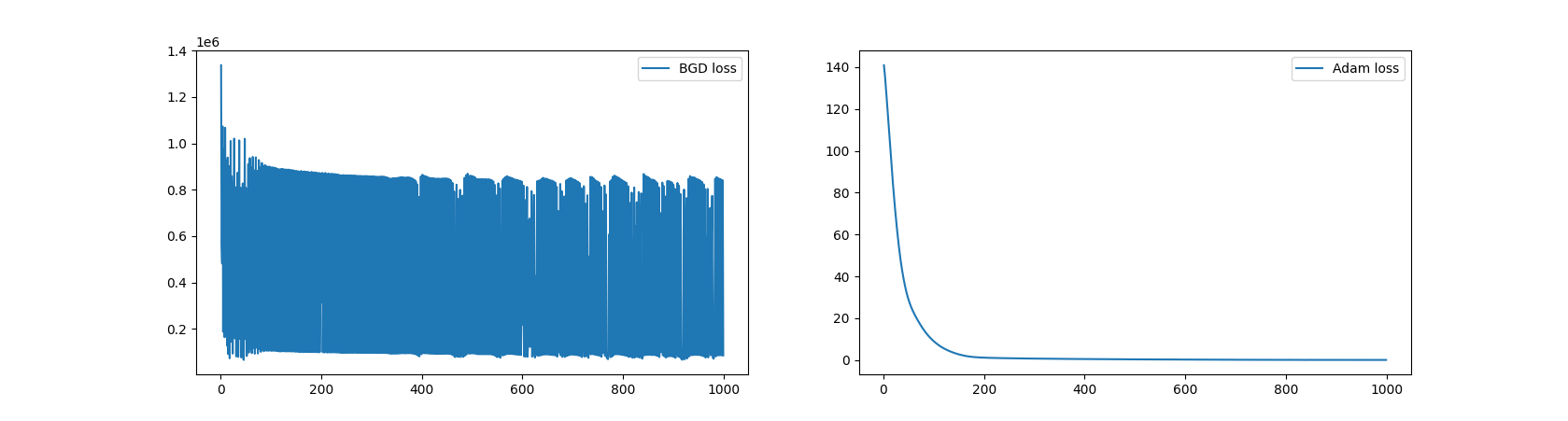


图 2 不同优化器训练的损失函数变化过程 lr=1e-2, n\_split=5, n\_iter=1000

可以看到，使用BGD训练时，震荡剧烈，loss在1000附近依然会震荡，然而采用Adam优化器训练时，迭代次数在500左右就接近收敛。

## 2.2 对比两种优化算法，画出学习曲线 (learning curve)

画出学习曲线1如图所示。

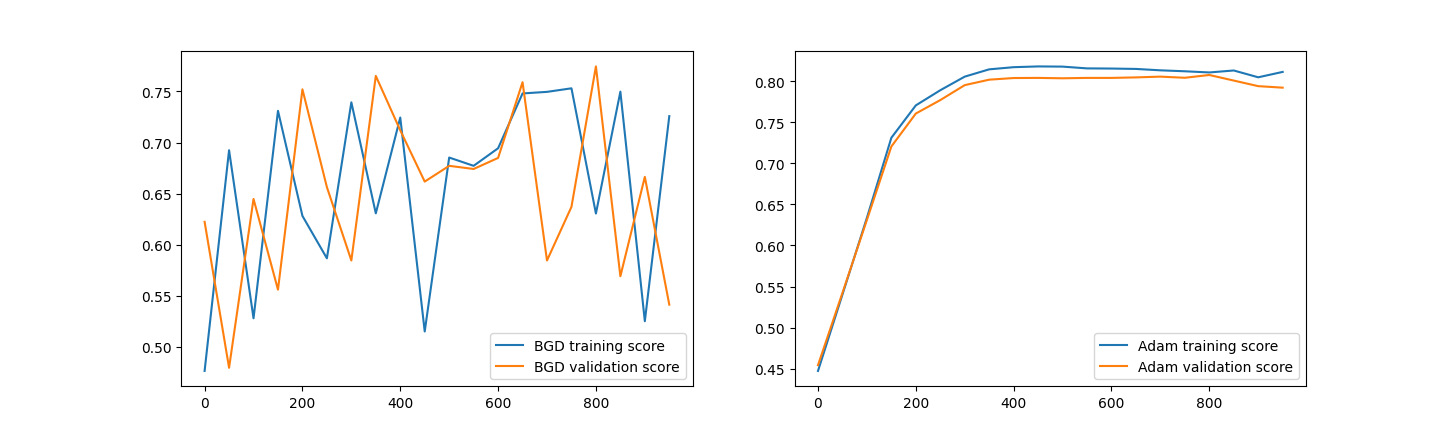


图 3 不同优化器的学习曲线 lr=1e-2, n\_iter=1000, n\_splits=5

可以看到，Adam优化器的速度远远快于BGD的速度。对于BGD (Fixed increment rule)，我们设定迭代10万次以后，能在测试集上达到78%的正确率(Adam收敛时约79%)。（这一实验在Apple M1芯片上跑了6小时，可惜没有记录下其对应的learning curve）

1. 我并没有找到学习曲线（learning curve）的官方定义，wikipedia的learning curve (machine learning)的词条中记录为“In machine learning, a learning curve (or training curve) plots the optimal value of a model's loss function for a training set against this loss function evaluated on a validation data set with same parameters as produced the optimal function.”，而在scikit-learn的官方文档[3]中，learning curve绘制的是不同cross-validation在同一迭代次数下的training score的平均以及validation score的平均。

## 2.3 用Training-Set2分别测试TestSet-1和TestSet-2，对比测试结果

|  |  |  |
| --- | --- | --- |
|  | TestSet-1 | TestSet-2 |
| Error Rate | 22.15% | 20.00% |

图 4 测试结果。n\_split=5, n\_iter=3000, r=5e-3，在2879次迭代时收敛 (loss=0)。

## 2.4 用Training-Set-1训练，并用TestSet-1测试

实验发现Error Rate为24.43%。

## 2.5 总结与思考

**对于2.1～2.2：**

相比Fixed increment rule，Variable increment rule的训练方式可以更为高效。在某一梯度方向的时候可以连续地迅速下降。

**对于2.2：**

1. 强制收敛的方法有多种，可以固定一定迭代次数后终止循环，也可以设置一个更大的阈值。
2. Learning Curve的定义并不唯一。本报告参考scikit-learn中的plotting\_learning\_curves的demo[3]。每50 epoch绘制一次training score与validation score。
3. 参考上述文档，当利用cross-validation时，每一轮的score等于cross-validation的平均。

**对于2.3-2.4：**

Training-Set-1有5000个样本，Traing-Set-2有1475个样本，自然用Training-Set-1训练的时候效果会更好。但这一现象并不能表明训练集越大越好。如果是50000个与14750个之间对比，14750数量的样本或许会更好，因为它的泛化能力更强。

# 实验三：罗杰斯特回归

## 3.1～3.2 用TrainingSet-1计算training error和cross validation，以及test error

我们设置参数n\_iter=1000, 1e-2, n\_splits=5, optimizer=“Adam”，得到training eror: 0.1984 validation error (use 20% training set as test set cross-validation): 0.2194 test error: 0.2133

## 3.3 根据测试集绘制ROC curve如图

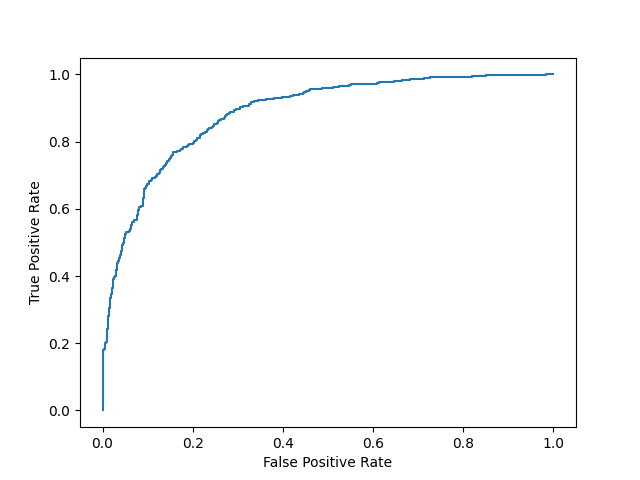


图 5 ROC曲线。Logistic Regression

## 3.4 分析判别死亡的主要成分

参考[4]，主成分分析后确定特征权重的公式为对的加权。输出前10项因素如图所示。可以发现，apache\_3是判断人生死最相关的因素。（由于scikit-learn有PCA工具包，故没有用推荐的statsmodel，此结果写入至表格*feature\_weigh.csv*，见附件）



# 实验四：K近邻算法

## 4.1 描述KNN库的基本参数和寻找最近邻的算法。

寻求最近邻的算法有KD-Tree。

构建：维空间数据集，其中，

1. 开始构造根节点，由根节点生成深度为1的左右子结点，左边对应小于切分点的子区域，右子结点对英语坐标大于切分点的子区域。
2. 重复选择为切分的坐标，
3. 直到两个子区域构建完成。

查找：

1. 从根结点开始递归访问直到叶子结点
2. 递归向上回退，维护最近结点，在区域中寻求最近结点
3. 回退到根结点输出维护的最近结点，此结点即位1-Nearest-Neighbor

在scikit-learn中，KNN的常见参数有

表格 2 sk-learn中KNN分类器常见参数

|  |  |  |
| --- | --- | --- |
| 参数名 | 参数可选值 | 作用 |
| n\_neighborsint | int, default=5 | k的值 |
| weights | ‘uniform’：每个点权重相等  ‘distance’：越近权重越大 | 指定权重函数 |
| algorithm | ‘auto’：自动选择  ‘ball\_tree’  ‘kd\_tree’ | 选择搜索方法 |

## 4.2 选择参数训练KNN

分别对weights的不同方案进行实验，

在weights=‘uniform’时：

对k 取1～79分别进行交叉检验，发现k=27时效果最好，正确率为68.20%；

在weight=‘distance’时：

发现k=23时效果最好，正确率为68.28%。

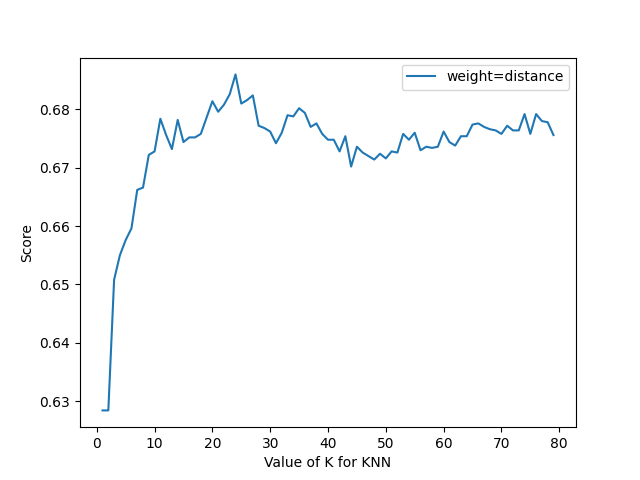
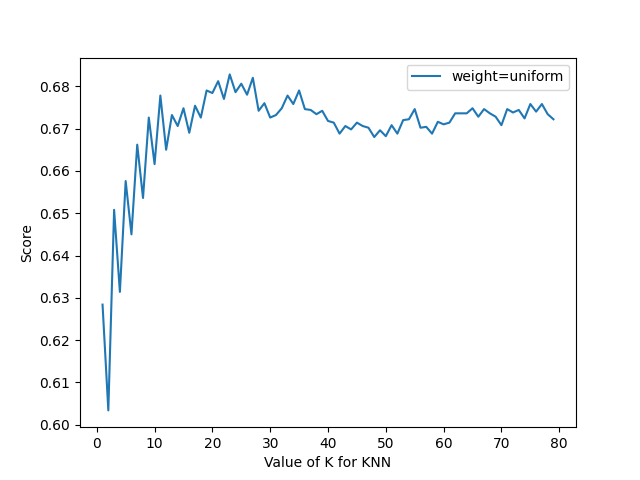


图6 K取不同的值对正确率的影响

但是事实上，如果我们设置weight=‘distance’，将在k=62时在验证集上达到最高的正确率。validation score为79.90%（上述实验采用TrainingSet-1与TrainingSet-2）

我们仅采用TrainingSet-1作为训练集，TestingSet-1作为测试集正确率为68.92%。

## 4.3 观察与讨论

1. 在训练KNN模型时，需要提前观测数据集，如果数据过大，务必要进行归一化或者标准化，保证在进行距离运算时不会越界。
2. KNN的训练集发生变化时，K也有必要重新设定。

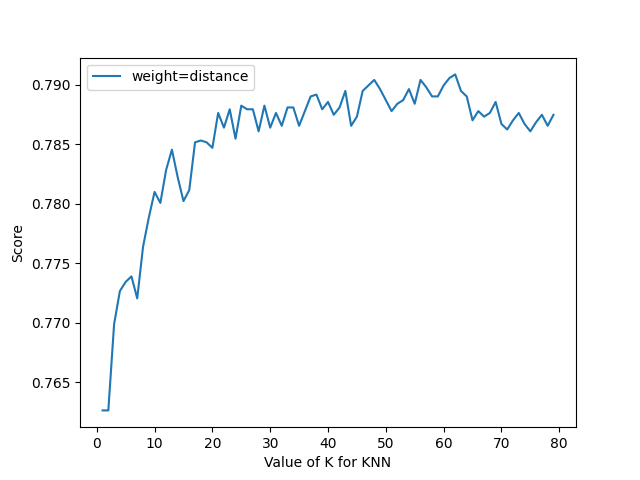
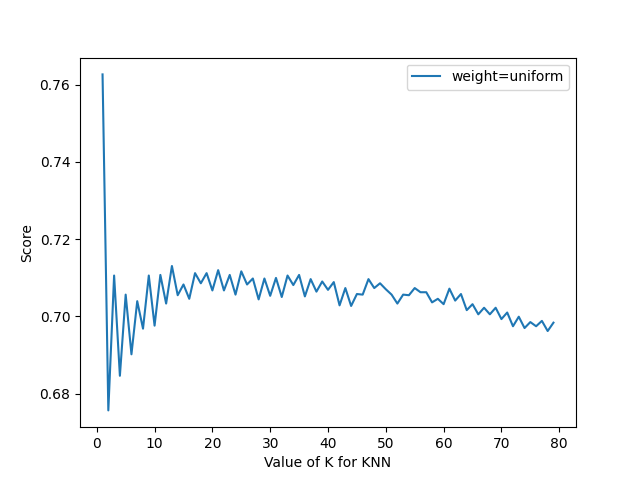


图7 当训练集发生改变时的对照

在之前的实验中，weight选择不同形式时对k的取值看似并没有很大的影响，但当我们改变训练集时，如将TrainingSet-1与TrainingSet-2均作为训练集，发现uniform下的k取1最优，distance下取62最优，且其正确率竟与前面的其他线性分类模型相持平。当训练集最大时，依照距离加权比等权求距离更优。

# 参考文献

1. https://blog.csdn.net/u014568921/article/details/45846531
2. Anestis Gkanogiannis and Theodore Kalamboukis, A modified and fast Perceptron learning rule and its use for Tag Recommendations in Social Bookmarking Systems, Department of Informatics Athens University of Economics and Business, Athens, Greece
3. https://scikit-learn.org/stable/auto\_examples/model\_selection/plot\_learning\_curve.html#sphx-glr-auto-examples-model-selection-plot-learning-curve-py
4. https://blog.csdn.net/lzw790222124/article/details/120262798