《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　2 0 2 1 2 1 4 4 8 0

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[0,1]区间

$$x^j = \frac{x^j - \min(x^j)}{std(v^j)}$$

主成分分析

在最初的实验中发现，原训练集的样本矩阵X不可逆，说明部分数据线性相关。为了让X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$y_i = \boldsymbol{w}^T \boldsymbol{x}_i$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

$$\max \boldsymbol{w}^T \boldsymbol{S}_b \boldsymbol{w}$$
$$\text{s.t. } \boldsymbol{w}^T \boldsymbol{S}_w \boldsymbol{w} = c \neq 0$$

再利用拉格朗日乘数法

$$\boldsymbol{w}^* = \boldsymbol{S}_w^{-1}(\boldsymbol{m}_1 - \boldsymbol{m}_2)$$

其中$\boldsymbol{m}_1, \boldsymbol{m}_2$为两类样本的均值向量。

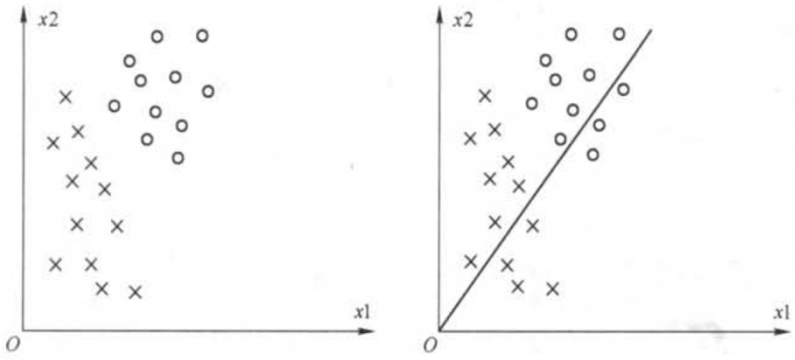

图 1 FLD 图示[1]

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m。

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: $w_0 = -\frac{1}{2}(\widetilde{m_1} + \widetilde{m_2})$（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%。

$w_0 = -\frac{1}{2}(\widetilde{m_1} + \widetilde{m_2})$	$w_0 = -\widetilde{m}$
79.90%	79.63%

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w_0和m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

$$w^k = w^{k-1} + lr^{k-1}\sum y_i x_i$$

当lr为恒定值时，称为 *fixed increment rule*，当lr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

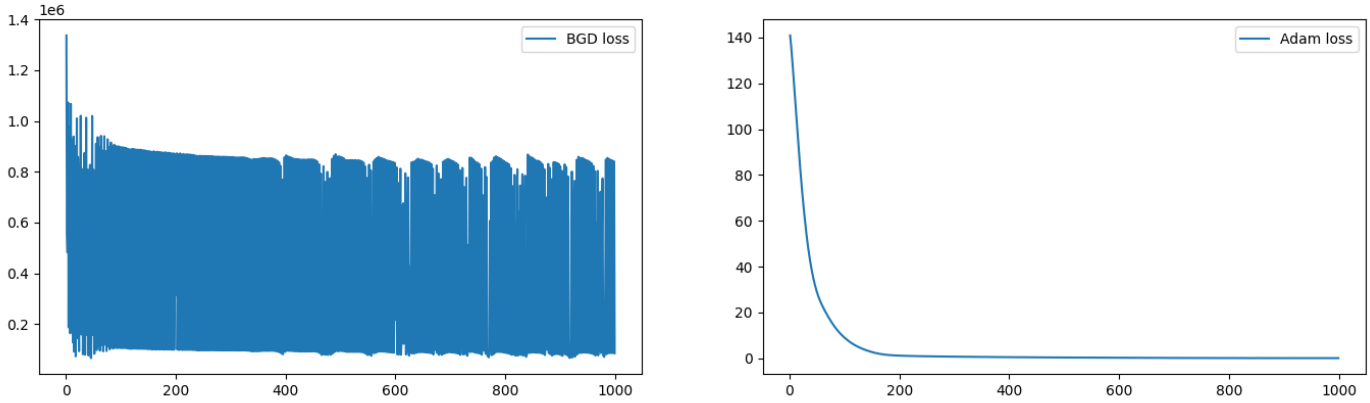

图 2 不同优化器训练的损失函数变化过程 lr=1e-2, n_split=5, n_iter=1000

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。

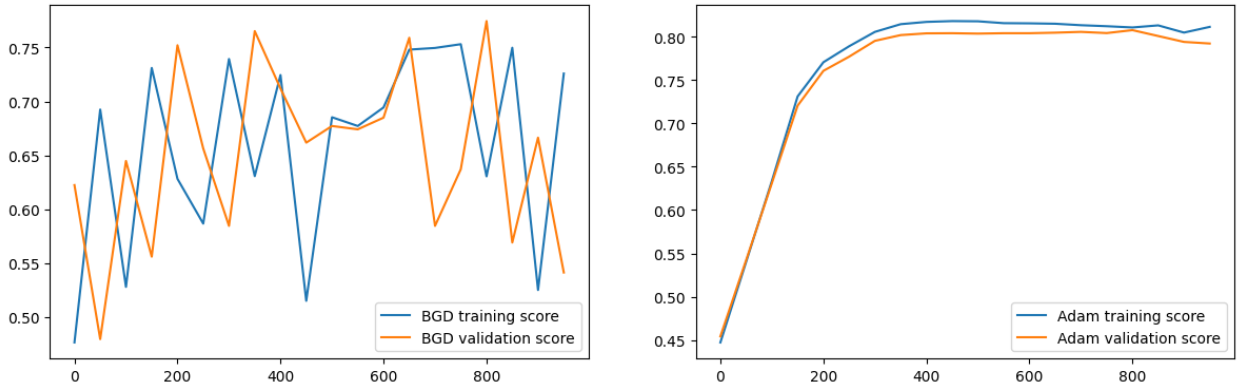

图 3 不同优化器的学习曲线 lr=1e-2, n_iter=1000, n_splits=5

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

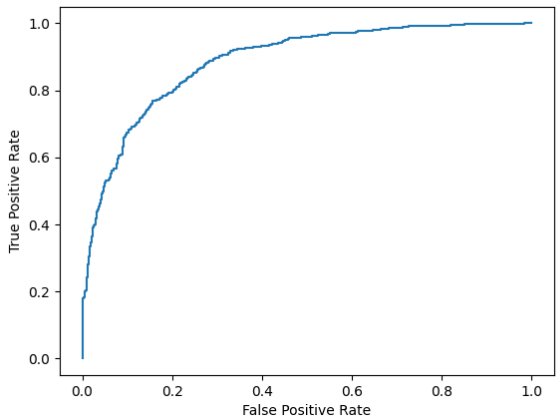

图 5 ROC 曲线。Logistic Regression

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$components \cdot explained_variance_ratio_$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）

1	0.773379	8	apache_3j	None	numeric	The APACHE III-J sub-diagnosis code which best describes the reason for the ICU admission
2	0.090228	7	apache_2	None	numeric	The APACHE II diagnosis for the ICU admission
3	0.044561	59	d1_platele	10^9/L	numeric	The highest platelet count for the patient during the first 24 hours of their unit stay
4	0.022396	19	map_apac	Millimetres	numeric	The mean arterial pressure measured during the first 24 hours which results in the highest APACHE III score
5	0.01376	15	glucose_ap	mmol/L	numeric	The glucose concentration measured during the first 24 hours which results in the highest APACHE III score
6	0.007389	49	h1_sysbp_	Millimetres	numeric	The patient's highest systolic blood pressure during the first hour of their unit stay, either non-invasively or invasively measured
7	0.006994	35	d1_sysbp_	Millimetres	numeric	The patient's highest systolic blood pressure during the first 24 hours of their unit stay, either non-invasively or invasively measu
8	0.005455	50	h1_sysbp_	Millimetres	numeric	The patient's lowest systolic blood pressure during the first hour of their unit stay, either non-invasively or invasively measured
9	0.005034	55	d1_glucos	mmol/L	numeric	The lowest glucose concentration of the patient in their serum or plasma during the first 24 hours of their unit stay
10	0.004841	36	d1_sysbp_	Millimetres	numeric	The patient's lowest systolic blood pressure during the first 24 hours of their unit stay, either non-invasively or invasively measu

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：k维空间数据集$T = \{x_1, x_2, \dots x_N\}$，其中$x_i = \left\{x_i^1, x_i^2, \dots, x_i^k\right\}^T$，$i = 1,2, \dots N$

(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x^1小于切分点的子区域，右子结点对英语坐标x^1大于切分点的子区域。

(2) 重复选择x^l为切分的坐标，$l = j(modk) + 1$

(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%。

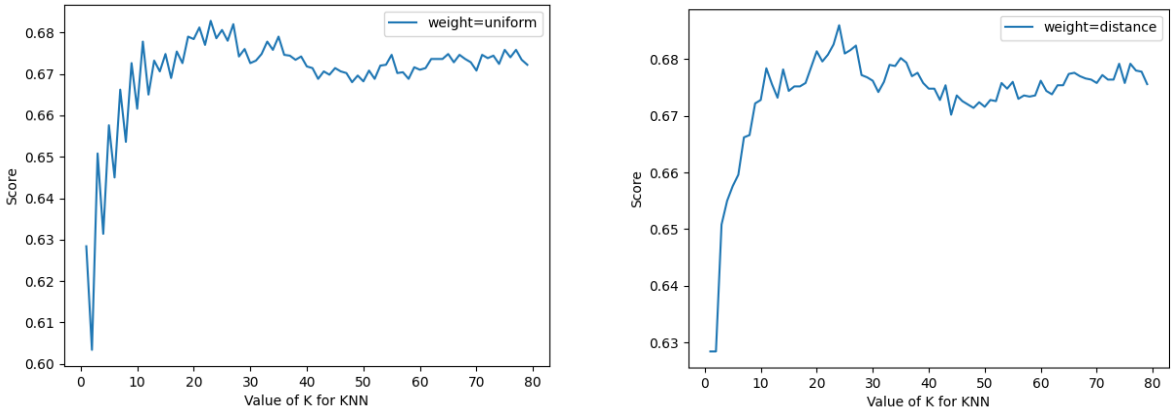

图 6 K 取不同的值对正确率的影响

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。

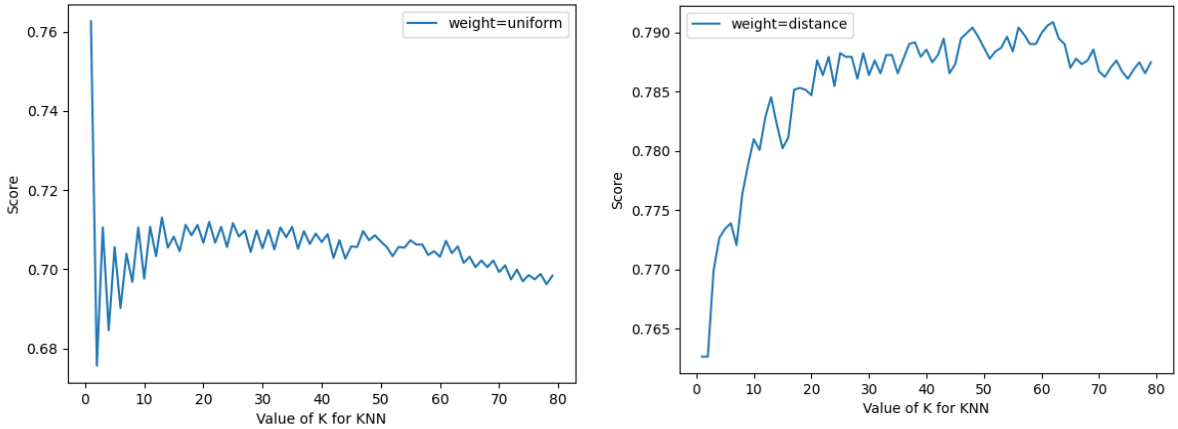

图 7 当训练集发生改变时的对照

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