**实验1报告**

自硕21 崔晏菲 2021210976

**1. FLD**

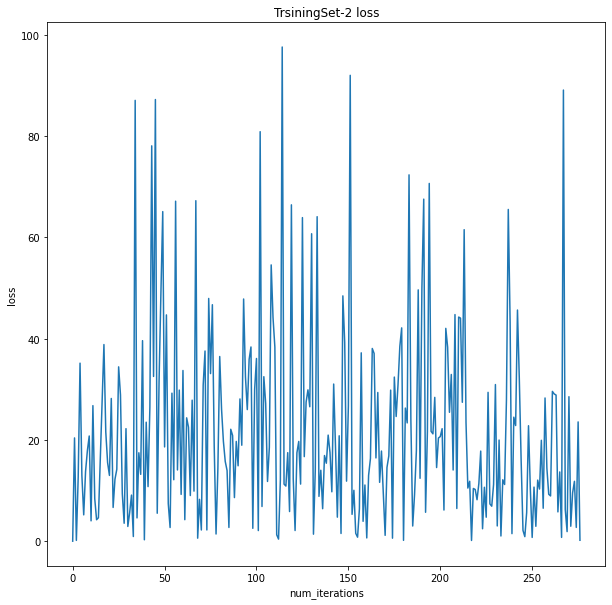
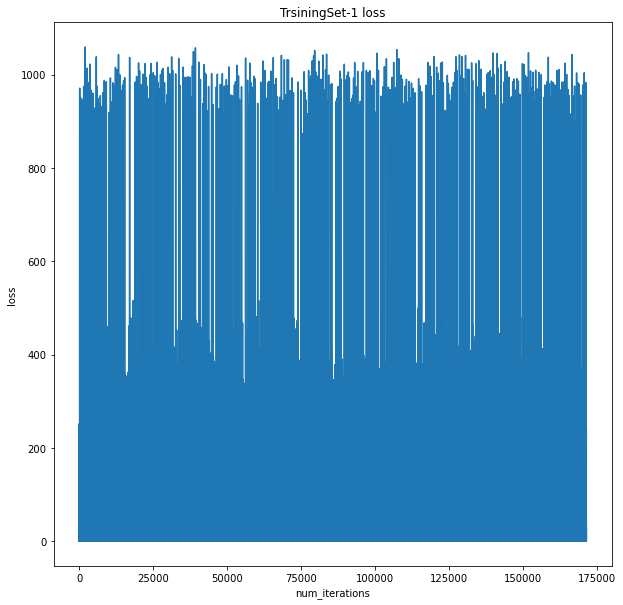
一开始，数据没有做标准化预处理，FLD就像是什么都没学到一般，一股脑把几乎所有数据都预测成同一类，测试准确率只有；后来我在训练的时候给数据加了标准化预处理，效果显著提升，测试准确率达到了；而我用sklearn的API，测试准确率高达

我们可以看到，显然，数据是线性不可分的，并且数据每个维度的分布相差极大。

**2. Perceptron**

(1) 因为数据是线性不可分的，所以无法收敛

(2) 我加了一个迭代次数，迭代超过500次就会强行停止。得到迭代过程中的loss变化折线图如下：



可以看到，训练集1的数据相差很大，基本上并不服从同一分布；训练集2的数据相差就比较小，因此loss的波动也比较小。

(3) 将在训练集2训练的感知器用在两个测试集中，得到的结果如下：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 准确率 | 精确度 | 特异度 |
| TestSet-1 | 0.7739289 | 0.7745455 | 0.773309 |
| TestSet-2 | 0.82 | 0.8309859 | 0.8012048 |

可见，在两个测试集上表现的都很好。测试集1稍微不如测试集2可以理解，因为二者分布不同。

(4) 在训练集1训练并在测试集1测试，得到的测试准确率为0.5041，精确度为0.0127，特异度为0.9982。

可以看到，在训练集1上做训练，很难学到正确的参数，这是因为训练集1的数据比较脏，不同样本的值相差过大。

**3. LR**

(1) 我进行了10折交叉验证。在首先对样本办进行标准化处理后使用逻辑回归。得到的结果如下：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Fold-K | 训练损失 | 训练准确率 | 训练精确度 | 训练特异度 | 验证损失 | 验证准确率 | 验证精确度 | 验证特异度 |
| 1 | 3.573767 | 0.658222 | 0.96427 | 0.339683 | 3.62811 | 0.66 | 0.968627 | 0.338776 |
| 2 | 3.575359 | 0.657778 | 0.966013 | 0.336961 | 3.471063 | 0.664 | 0.960784 | 0.355102 |
| 3 | 3.428801 | 0.658 | 0.968192 | 0.335147 | 3.572037 | 0.65 | 0.94902 | 0.338776 |
| 4 | 3.414714 | 0.655111 | 0.965142 | 0.332426 | 3.372873 | 0.666 | 0.964706 | 0.355102 |
| 5 | 3.503063 | 0.656889 | 0.962527 | 0.338776 | 3.415847 | 0.666 | 0.984314 | 0.334694 |
| 6 | 3.540796 | 0.659111 | 0.96122 | 0.344671 | 4.327175 | 0.674 | 0.992157 | 0.342857 |
| 7 | 3.398005 | 0.658889 | 0.969499 | 0.335601 | 3.213831 | 0.626 | 0.941176 | 0.297959 |
| 8 | 3.615521 | 0.658 | 0.96427 | 0.339229 | 3.635693 | 0.666 | 0.972549 | 0.346939 |
| 9 | 3.479314 | 0.658 | 0.96427 | 0.339229 | 2.779886 | 0.648 | 0.980392 | 0.302041 |
| 10 | 3.444151 | 0.657111 | 0.96732 | 0.33424 | 3.891436 | 0.654 | 0.952941 | 0.342857 |
| 平均 | 3.497349 | 0.657711 | 0.965272 | 0.337596 | 3.530795 | 0.6574 | 0.966667 | 0.33551 |

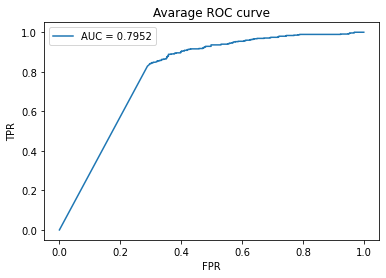
可以看到，准确率还是比较可观的，比单层感知器要强很多。

(2) 在测试集1上的结果如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fold-K | 测试损失 | 测试准确率 | 测试精确度 | 测试特异度 |
| 1 | 4.05664 | 0.661805 | 0.963437 | 0.361818 |
| 2 | 4.045298 | 0.661805 | 0.965265 | 0.36 |
| 3 | 3.955103 | 0.667274 | 0.967093 | 0.369091 |
| 4 | 3.895093 | 0.661805 | 0.965265 | 0.36 |
| 5 | 3.973652 | 0.661805 | 0.961609 | 0.363636 |
| 6 | 4.090437 | 0.666363 | 0.967093 | 0.367273 |
| 7 | 3.84828 | 0.656335 | 0.965265 | 0.349091 |
| 8 | 4.109348 | 0.66454 | 0.965265 | 0.365455 |
| 9 | 3.890492 | 0.661805 | 0.965265 | 0.36 |
| 10 | 3.96553 | 0.662716 | 0.965265 | 0.361818 |
| 平均 | 3.982987 | 0.662625 | 0.965082 | 0.361818 |

可以看到，模型在训练集和测试集和验证集上的表现非常接近，说明没有发生过拟合。

(3) 计算得到的平均ROC曲线为：



(4)我用了两种不同的方法进行了特征重要性排序。

(4.1)直接比较系数法。因为我的数据都是进行标准化之后的，所以可以通过直接比较各项参数对应的系数绝对值大小来判断重要性。得到的最重要的前10名如下：

1. intercept

2. apache\_4a\_icu\_death\_prob

3. gcs\_motor\_apache

4. gcs\_eyes\_apache

5. d1\_temp\_min

6. gcs\_verbal\_apache

7. ventilated\_apache

8. d1\_spo2\_min

9. intubated\_apache

(4.2)比较p值法。通过使用StatsModels自带的分析工具，我们可以比较每项参数对应的p值大小来确定重要性排序。得到的最重要的前10名如下：

1. age

2. ventilated\_apache

3. apache\_4a\_icu\_death\_prob

4. d1\_spo2\_min

5. wbc\_apache

6. solid\_tumor\_with\_metastasis

7. bun\_apache

8. d1\_hemaglobin\_max

9. d1\_temp\_min

10. d1\_heartrate\_max