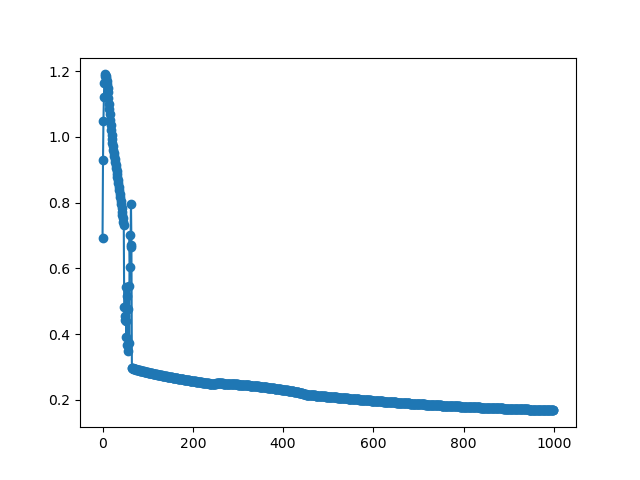
**SE3332 Lab1**

**521021910197 高健翔**

1. LR
2. 参数选择与运行时间

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学习率 | 循环次数 | 运行时间t/s | 训练集结果 | 测试集结果 |
| 0.01 | 100000 | 5.979 | 0.9525 | 0.92 |
| 0.01 | 10000 | 0.5852 | 0.925 | 0.885 |
| 0.01 | 1000 | 0.05813 | 0.675 | 0.65 |
| 0.02 | 100000 | 6.394 | 0.955 | 0.905 |
| 0.02 | 10000 | 0.6299 | 0.925 | 0.875 |
| 0.02 | 1000 | 0.06149 | 0.8775 | 0.8 |
| 0.005 | 100000 | 5.973 | 0.955 | 0.93 |
| 0.005 | 10000 | 0.6027 | 0.825 | 0.79 |
| 0.005 | 1000 | 0.05453 | 0.8775 | 0.81 |

1. 最终选择rate=0.001 循环次数为100000(其loss能够收敛，某些效果较好的参数loss的收敛性不太好)



Loss值(每100次记录一个loss)

训练集准确性：0.9375

测试集准确性：0.89

运行时间为：6.1471s

1. W初始值的影响—上面的实验w都是从0开始，下面令w的初始值处于随机状态

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rate 0.001 / epoch 100000 | 准确率1 / (train, test) | 2 | 3 | Average |
| (-0.001, 0.001) | (0.9375, 0.89) | (0.9375, 0.89) | (0.9375, 0.89) | (0.9375, 0.89) |
| (-0.01, 0.01) | (0.9475, 0.88) | (0.9475, 0.88) | (0.9375, 0.89) | (0.9442, 0.887) |
| (-0.1, 0.1) | (0.9475, 0.88) | (0.9375, 0.89) | (0.9475, 0.88) | (0.9442, 0.887) |

对于训练次数足够大的情况，初始值的影响似乎并不是很大

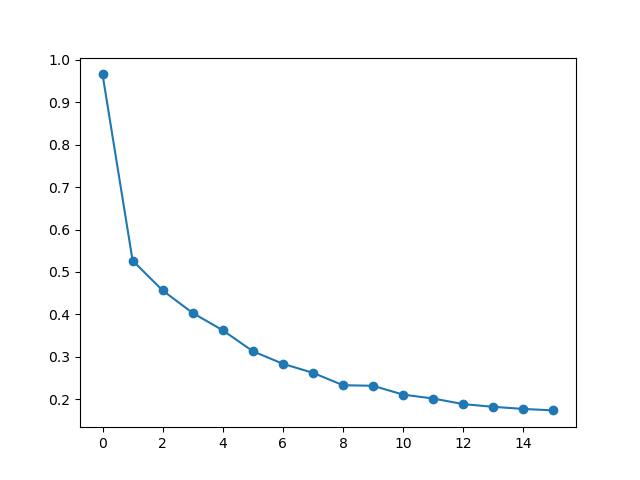
1. SVM

实验中发现当学习率设置稍大就会造成loss不收敛，因此下面只显示学习率小的情况。

1. 参数选择与运行时间

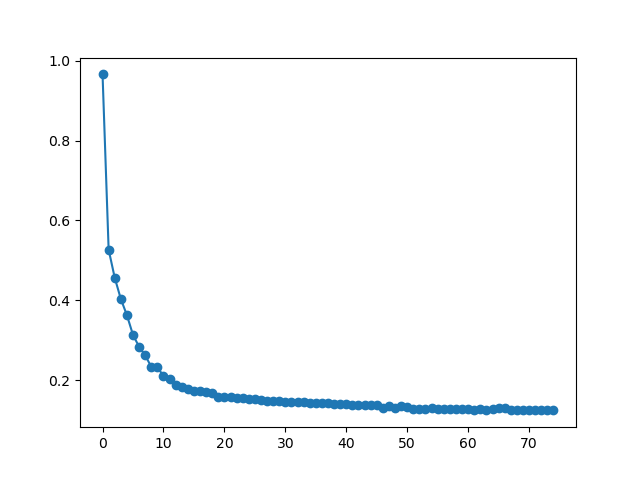
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学习率 | 循环次数 | 运行时间t/s | 训练集 | 测试集 |
| 0.00001 | 1200 | 7.757 | 0.935 | 0.92 |
| 0.00001 | 6000 | 32.21 | 0.9525 | 0.915 |
| 0.00001 | 12000 | 63.21 | 0.96 | 0.915 |
| 0.000005 | 1500 | 9.679 | 0.9425 | 0.925 |
| 0.000005 | 7500 | 42.11 | 0.9525 | 0.92 |
| 0.000005 | 15000 | 79.41 | 0.9525 | 0.905 |

1. 选择测试集上效果最好的 学习率=0.000005 循环次数=1500（每100次记录一个loss）



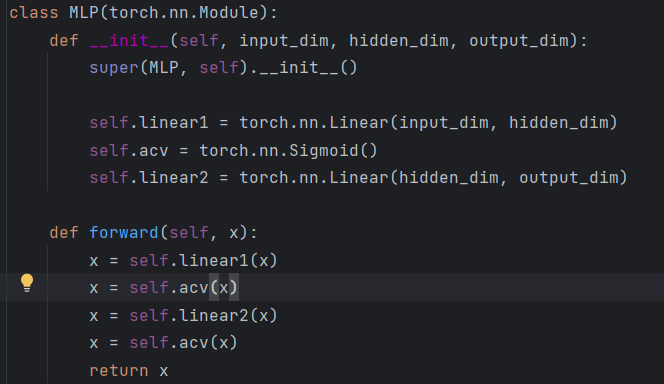
Loss值

事实上，当循环次数增加时，loss仍然有收敛迹象，但是在测试集上效果却并不是很好，猜测出现了过拟合



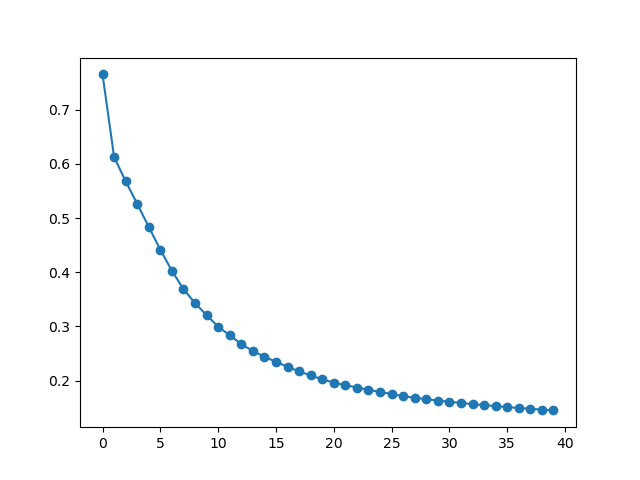
7500次的loss

1. MLP use pytorch
2. 参数选择与运行时间



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学习率 | 循环次数 | 隐藏层节点数 | 运行时间t/s | 训练集 | 测试集 |
| 0.001 | 8000 | 64 | 29.83 | 0.9325 | 0.92 |
| 0.001 | 20000 | 64 | 73.84 | 0.9475 | 0.93 |
| 0.0001 | 20000 | 64 | 72.32 | 0.87 | 0.82 |
| 0.001 | 8000 | 10 | 21.69 | 0.8825 | 0.85 |
| 0.001 | 20000 | 10 | 55.94 | 0.94 | 0.905 |
| 0.0001 | 20000 | 10 | 54.10 | 0.805 | 0.77 |

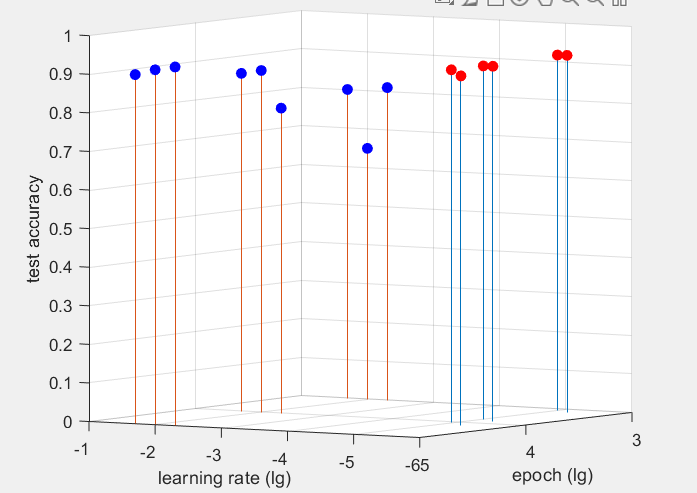
1. 选取测试集效果最好的参数 学习率=0.001 循环次数=20000 隐藏层节点数=64



Loss(每500次记录一个loss)

可以看出loss还是一直在收敛的

1. 总结
2. LR与SVM的可视化比较(MLP涉及到节点数，故不在图像上绘制)



1. 简单分析

选取每一种方法中效果最好的进行比较(在表格中标红)，可以发现在效果相似的情况下LR的时间效率最好，SVM的较差

分析：逻辑回归算法相对简单，容易实现，计算成本低；但是只适用于线性数据，对于异常值比较敏感(本次数据集的最后一列对于线性回归的影响是很大的)

SVM：在更高维的空间表现良好，在非线性数据中，也可以通过核技巧来处理；但是对大规模数据集计算开销很大，需要消耗较多资源

MLP：具有高度可配置性，包括中间层的层数，神经元的数量等，可以适应不同的任务，可以使用GPU加速(实验的时候没有配置好)；然而在小数据集中容易过拟合，且对于参数的选择比较困难