机器学习 Exercise1 实验报告

陈子康
U201713689

更改后的程序名为 test.py，使用的数据集为 data_normalized，均在文件夹内给出。

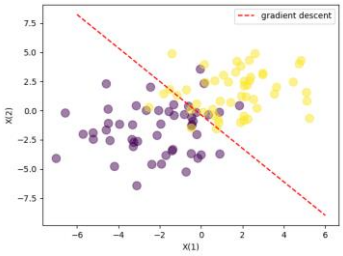
lr.py 运行结果

程序结果：

```
998 0.2775721848011017
999 0.2775695323944092
data is
tensor([[0.9116, 0.6356]])
tensor([[-0.0137, -0.0079]])
data is
tensor([0.2362])
tensor([0.0040])
Final gradient descend: [Parameter containing:
tensor([[0.9116, 0.6356]], requires_grad=True), Parameter containing:
tensor([0.2362], requires_grad=True)]
```

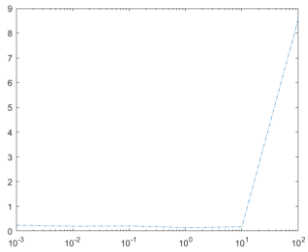
误差大小最后为：0.2775695323944092

图片：



更改学习率运行结果

将 lr 从 0.001 按十倍率取到 100，得到误差与 lr 的变化关系图：



数据为：
x=[0.001,0.01,0.1,1,10,100];
y=[0.2374141812324524,0.1960376352071762,0.20333701372146606,0.1418078988790512,0.1686791479587555,8.575901985168457];
由结果可以看出，随着学习率的增大，误差会在 10 左右较小，如果过大，则会损失信息，导致结果误差过大。
一般而言学习率取在 1 以内都是可以的。

对“Breast Cancer Wisconsin (Diagnostic) Data Set”进行处理

代码中需要改动的地方有：
1．将 X 改为从外部 csv 文件导入，注意预处理删去字符。读取 X 的维度 D 和数据集大小 N。
2．将标记 T 根据 N 进行改动，注意 N 为奇数时需要加一处理。

直接进行处理，可以发现当 lr 为除了 0.001 之外的值的时候，误差为 49，并且 1000 次训练中保持不变。而当 lr=0.001 时，有概率出现误差变化为：

```
988 35.67158508300781
989 24.63408851623535
990 5.423740386962891
991 8.77652359008789
992 5.191959381103516
993 43.7989501953125
994 42.21365737915039
995 38.595726013183594
```

说明结果误差并不稳定，模型无法正确进行分类。

通过与同学讨论可以得知，结果误差较大且不稳定可能是由于数据集数字本身较大，且不在同一个量级，因此预处理数据，使得每一列数据进行归一化处理。如此处理之后可以发现误差降到了 0.69，并且误差随着模型训练次数的增加而逐渐减小，符合预期。

此外，如果在模型中加入中间层，使得 D 维数据先映射到 10 个输出，而后 10 个输出映射到 1 个结果，误差会在 0.001 的量级上减小，最终的结果为：

```
994 0.693148672580719
995 0.693148672580719
996 0.693148672580719
997 0.693148672580719
998 0.693148672580719
999 0.693148672580719
data_is
```

可以发现模型最终误差为 0.693，可以接受。

将优化器改为 RMSprop，得到的结果误差相较而言会更小一点：

```
992 0.693143904209137
993 0.693143904209137
994 0.693143904209137
995 0.693143904209137
996 0.693143904209137
997 0.693143904209137
998 0.693143904209137
999 0.693143904209137
```