实验 2：基于回归分析的大学综合得分预测实验报告

陈泓宇 2022/7/4

实验任务

本次实验的任务是：根据所给大学综合情况及得分数据，使用线性回归模型，进行得分预测。

数据分析

在本次实验给出的数据中，有共 9879 组数据，每个数据对应有一个标签（blueWins）和 38 个特征。其中红队与蓝队特征各 19 项。

数据集划分

将原始数据集以 8：2 划分为训练集和测试集。

结果分析

对线性回归模型的系数进行分析

实验过程

数据集划分

同上次实验一样，使用 sklearn.model_selection 中的 train_test_split 将数据按 8：2 分成训练集与测试集。

```
from sklearn.model_selection import train_test_split
all_y = Y.values
all_x = X.values
#print(type(all_y))

x_train, x_test, y_train, y_test = train_test_split(all_x, all_y, test_size = 0.2, random_state = 2022)

all_y.shape, all_x.shape, x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

数据预处理

划分好的数据集是 numpy.ndarry 格式的数据，为了方便之后的矩阵运算，需要将训练集的数据转为 torch.tensor

```
import torch
x_tensor = torch.tensor(x_train,dtype=torch.float)
y_tensor = torch.tensor(y_train, dtype = torch.float)
```

在需要求解的线性模型中，除了具有每个特征的权重系数，还有一个常数像项，在计算之前可以在原数据的特征矩阵 X 后加上一列单位矩阵，将它作为 feature 矩阵进行系数求解。

```
feature = torch.cat((x_tensor,e_tensor), 1) #链接偏移量与x（特征）矩阵
feature
```

模型求解

根据课堂上给出的公式（最小二乘法）：

$$\beta = (X'X)^{-1}X'Y$$

求出各项系数

```
b_pre = torch.mm(torch.mm(torch.inverse(torch.mm(torch.t(feature),feature)),torch.t(feature)),y_tensor.view(1600,1))
print(list(b_pre))
```

```
[44] ✓ 0.1s
... [tensor([-0.0610]), tensor([0.0004]), tensor([-0.0001]), tensor([-0.0068]), tensor([0.0002]), tensor([-0.0058]),
tensor([-0.0023]), tensor([-0.0025]), tensor([65.0603])]
```

在得出的系数中，系数为正数的，说明大学的综合得分与该项正相关，反之则负相关

结果分析与改进的尝试

将所得系数与测试集数据相乘，可以得到测试集的预测数据

```
test_ans = torch.mm(test_feature,b_pre)
```

带入 loss 公式：

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

得到 loss 的值为 8977

```
loss = torch.tensor(y_test).view([400,1]) - test_ans
loss = torch.mm(torch.t(loss),loss)
loss
```

```
✓ 0.1s
tensor([[8977.9963]], dtype=torch.float64)
```

计算均方根误差 RMSE

$$\sqrt{\text{MSE}} = \sqrt{\text{loss}/\textit{预测个数}}$$

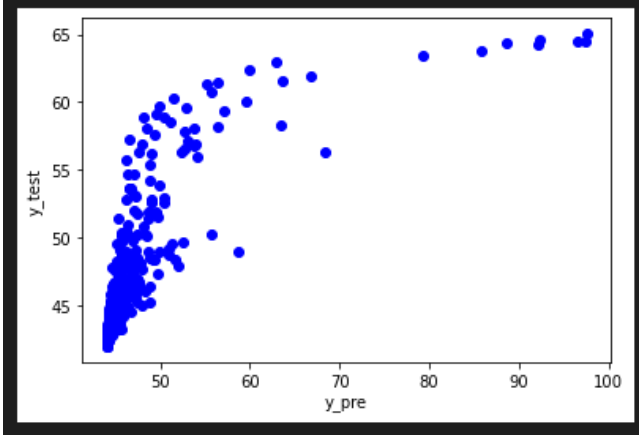
```
RMSE = (loss.item()/len(lable))**0.5
RMSE
```

```
✓ 0.2s
2.368807228654319
```

由于没有对比的 loss 及 RMSE 作为参考，我使用 matplotlib 中的 pyplot 将预测值与参考值画了下来

```python
from matplotlib import pyplot as plt
from scipy.linalg import expm,logm
x = y_test
y = test_ans
plt.xlabel("y_pre")
plt.ylabel("y_test")
plt.plot(x,y,"ob")
plt.show()
```

✓ 0.3s

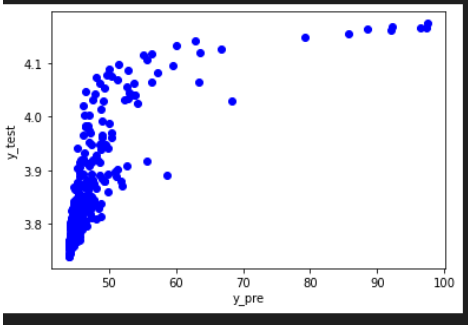


可以看到这个图像并不是很符合“线性”。

我尝试使用对数使数据看起来更线性但是失败了

```python
from matplotlib import pyplot as plt
from scipy.linalg import expm,logm
x = y_test
y = torch.log(test_ans)
plt.xlabel("y_pre")
plt.ylabel("y_test")
plt.plot(x,y,"ob")
plt.show()
```

✓ 0.1s



由于时间问题我尝试将训练集的标签数据取指数后重新计算系数，使用以下模型

$$e^{(y-y_{min})} = k/e^{y_{min}}x + b$$

但是失败了

```python
lable = torch.exp(y_tensor + torch.ones(y_tensor.shape) * (0 - y_tensor.min()))
```

```
test_ans = torch.log(torch.mm(test_feature,b_pre))
```

```
tensor([[51.5780],
        [    nan],
        [48.4016],
        [    nan],
        [    nan],
        [49.7454],
        [    nan],
        [    nan],
        [    nan],
        [    nan],
        [    nan],
        [    nan],
        [51.3573],
        [    nan],
        [    nan],
        [50.3888],
        [43.9595],
        [    nan],
```