LAB01：Regression

小组编号：

小组成员 1：吴其平

小组成员 2：

小组成员 3：

小组成员 4：

实验报告需要包含的内容如下章节一所示

在提交实验报告时请**将代码以附件形式随报告一起提交**

请在实验报告中标注实验报告的各个部分是由哪位小组成员完成的

实验报告的格式无限制，页数限制在 10 页内，不包含第一页

一、实验报告

1.1. 数据准备

使用 pandas 读入“Concrete_Data.xls”文件，并将数据按行随机打乱。本数据集共有 1030 个样本，每个样本有 9 个属性，其中前 8 个属性是输入特征，最后一个属性是输出。

1.2. 数据清洗

首先检查数据集中是否有缺失的数据，若有则将此样本去除。本数据集无缺失数据。

其次分出输入和输出、训练集和测试集。样本的前 8 个属性是输入特征，最后一个属性是输出。由于数据已经按行随机打乱，因此可以使用数据集的前 70%作为训练集，后 30%作为测试集，共得到 721 个训练样本和 309 个测试样本。

接着进行数据的标准化。对每个特征计算训练集上的均值$mean$和标准差std，将每个样本按如下方式进行标准化：

$$x' = \frac{x - mean}{std}$$

为方便后续计算，将 dataframe 对象转为 ndarray 类型。

最后对样本进行特征升维，添加特征$x_0 = 1$，此时样本的特征维度变为 9。

1.3. 模型搭建

对本实验的符号说明如下：

1. n表示样本的维度
2. m表示样本的数量
3. X表示样本输入集，为$(n+1) \times m$的矩阵
4. θ表示参数，为$(n+1) \times 1$的向量
5. y表示样本输出集，为$m \times 1$的向量
6. $x^{(i)}$表示第i个样本

7. x_j表示样本的第j个特征

8. α表示学习率

9. λ表示正则化系数

基础模型使用的假设函数为线性函数，其表达式如下：

$$h_\theta\left(x^{(i)}\right) = \sum_{j=0}^{n} \theta_j x_j^{(i)} = \theta^T x^{(i)}$$

其向量化形式：

$$h_\theta(X) = X^T\theta$$

使用的$loss$如下：

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)} - h_\theta\left(x^{(i)}\right)\right)^2 = \frac{1}{2m}\sum_{i=1}^{m}(y^{(i)} - \theta^T x^{(i)})^2$$

其向量化形式：

$$J(\theta) = \frac{1}{2m}(y - X^T\theta)^T(y - X^T\theta)$$

计算梯度$grad$：

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)} = \frac{1}{m}\sum_{i=1}^{m}(\theta^T x^{(i)} - y^{(i)})x_j^{(i)}$$

其向量化形式：

$$\frac{\partial J(\theta)}{\partial \theta} = \frac{1}{m}X(X^T\theta - y)$$

每次更新参数θ：

$$\theta_j = \theta_j - \alpha\frac{\partial J(\theta)}{\partial \theta_j}\ (j = 0,1,\dots,n)$$

其向量化形式：

$$\theta = \theta - \alpha \frac{\partial J(\theta)}{\partial \theta}$$

由于向量化形式在计算时速度更快，因此本实验的代码全部使用向量化形式。

1.4. 模型训练测试

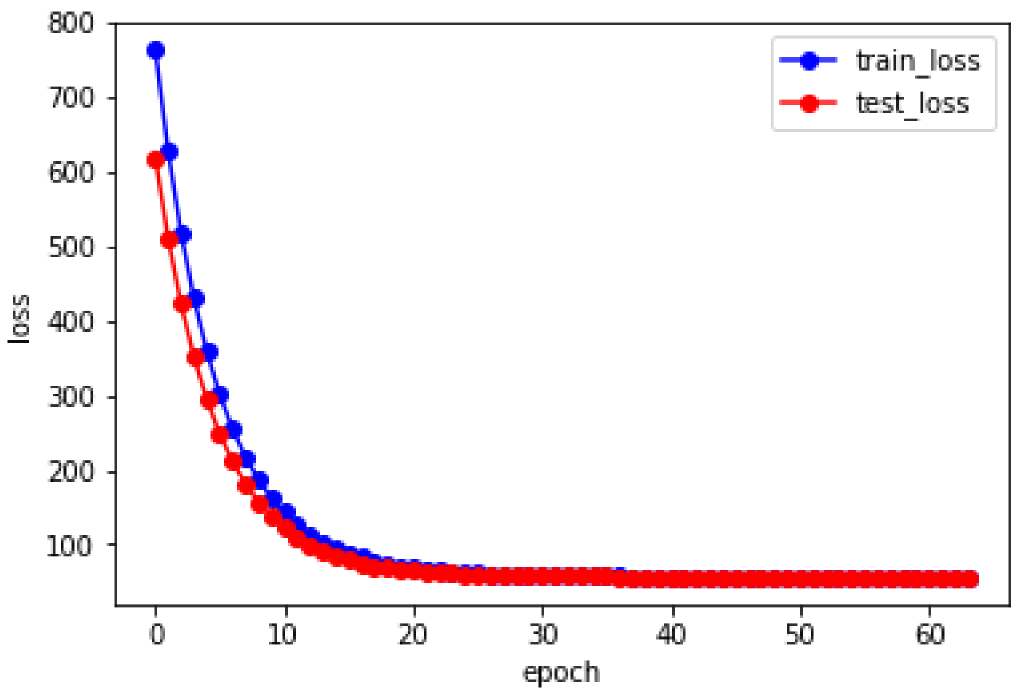
首先初始化θ。本实验使用 np. random. rand 函数随机生成$[0,1)$区间上的均匀分布随机值，维度为 9。

接着进行 64 个 epoch 的训练，设定学习率$\alpha = 0.1$。

将训练完的模型在测试集上进行测试，得到$loss$为 55. 9976。

1.5. 结果可视化

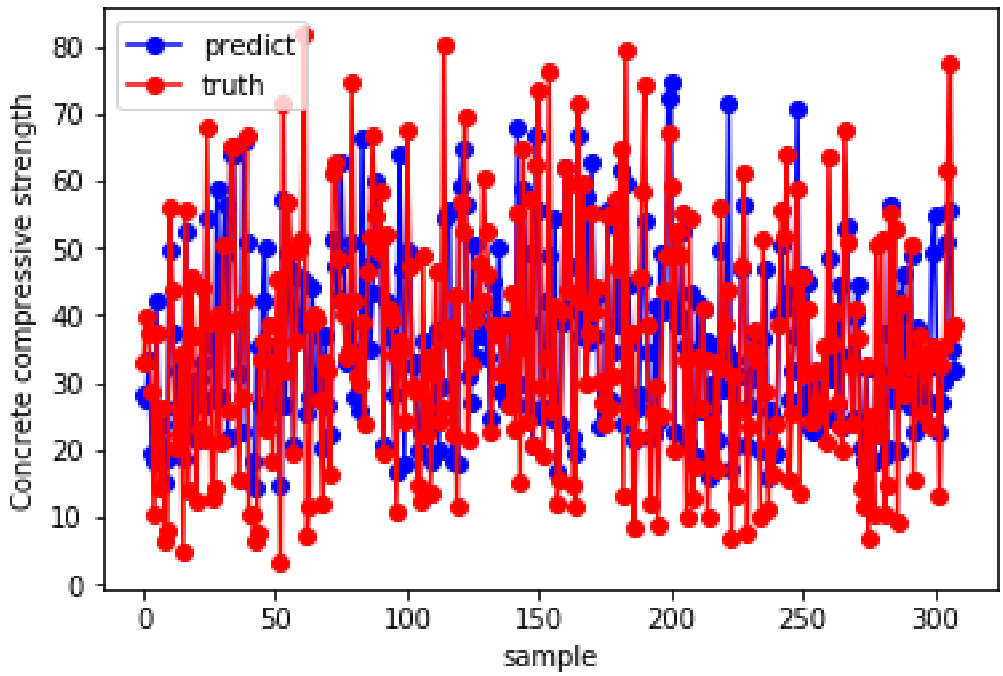
训练过程中模型在训练集和测试集上$loss$的变化如下图所示：



其中蓝色点是模型在训练集上的$loss$，红色点是型在测试集上的$loss$。由此

图可见模型成功收敛。

模型在测试集上的预测值和测试集的真实值如下图所示：



其中蓝色点是模型的预测值，红色点是真实值。由此图可见模型对大多数测试样本的预测都已较为准确。

1.6. 模型优化

基础模型在测试集上的$loss$为 55.9976。由于输出只是 9 个输入特征线性组合，因此其表达能力有限。于是希望对模型进行优化以得到更好的预测结果。

尝试使用如下三种优化方法。

优化 1：使用正则化。

在计算$loss$时加入正则项，此处使用θ的$L2$范式：

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(y^{(i)} - \theta^T x^{(i)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n}\theta_j^2$$

其向量化形式如下：

$$J(\theta) = \frac{1}{2m}(y - X^T\theta)^T(y - X^T\theta) + \frac{\lambda}{2}\theta^T\theta$$

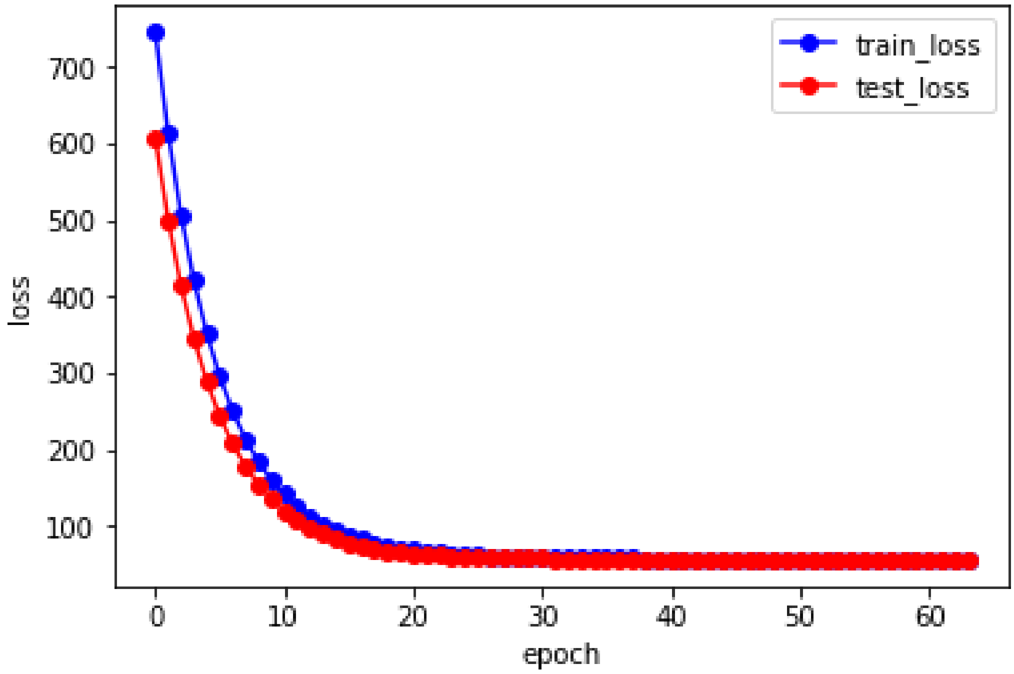
在计算梯度时也加入正则项，需要注意的是参数更新时只对θ_1到θ_n进行正则化，而不对θ_0进行正则化：

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(\theta^T x^{(i)} - y^{(i)})x_0^{(i)}$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(\theta^T x^{(i)} - y^{(i)})x_j^{(i)} + \lambda\theta_j \ (j = 1,2,\dots,n)$$

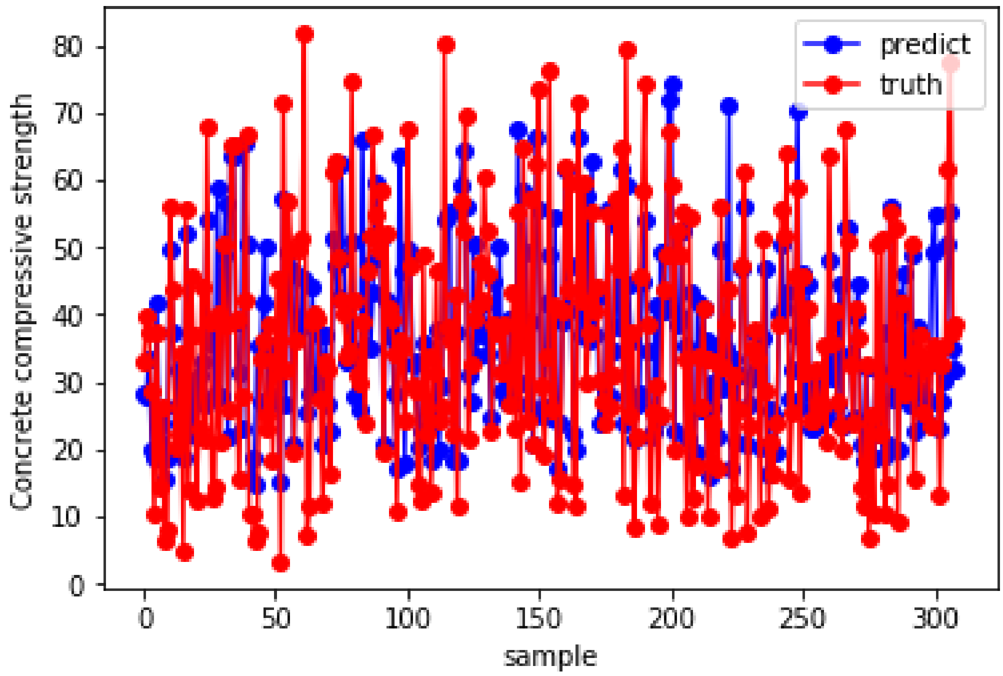
训练 64 个 epoch，设定学习率$\alpha = 0.1$。尝试不同正则化系数λ，分别进行训练并取使得模型在训练集上$loss$最小的λ作为最终的λ，得到$\lambda = 0.01$。

训练过程中模型在训练集和测试集上$loss$的变化如下图所示：



由此图可见模型成功收敛。进行测试后，得到在测试集上的$loss$为 56.0237，高于基础模型。

模型在测试集上的预测值和测试集的真实值如下图所示：



分析后认为，正则化的作用是防止过拟合，而基础模型中并未出现过拟合的现象，因此使用正则化的没有提升效果。

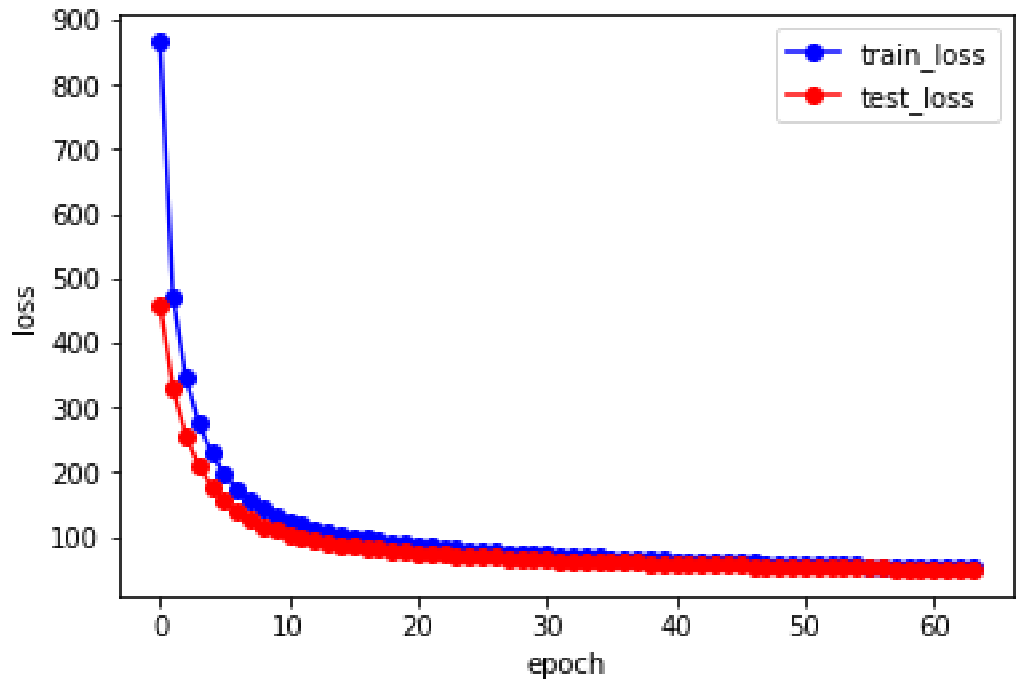
优化 2：增加特征。

在基础模型中，样本的特征维度仅为 9，并不能很好表示样本的特征。如果增加样本的特征维度，便能表达样本的更多信息。

本模型使用不同特征相乘作为新的特征，即$x_{j_1}x_{j_2}\,(j_1,j_2 = 1,2,\dots,n \wedge j_1 \neq j_2)$。因此样本的特征维度变为 37。

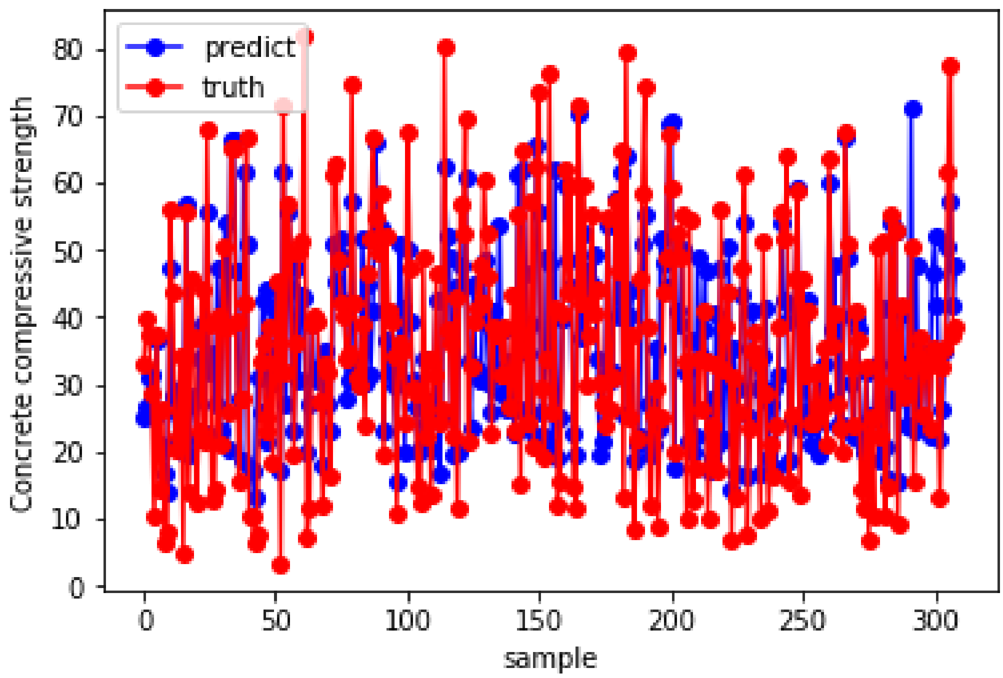
训练 64 个 epoch，设定学习率$\alpha = 0.1$。尝试不同正则化系数λ，分别进行训练并取使得模型在训练集上$loss$最小的λ作为最终的λ，得到$\lambda = 0.01$。

训练过程中模型在训练集和测试集上$loss$的变化如下图所示：



由此图可见模型成功收敛。进行测试后，得到在测试集上的$loss$为49.3766，低基础模型。

模型在测试集上的预测值和测试集的真实值如下图所示：



可见样本升维对模型的提升效果明显。

优化 3：标准方程法。

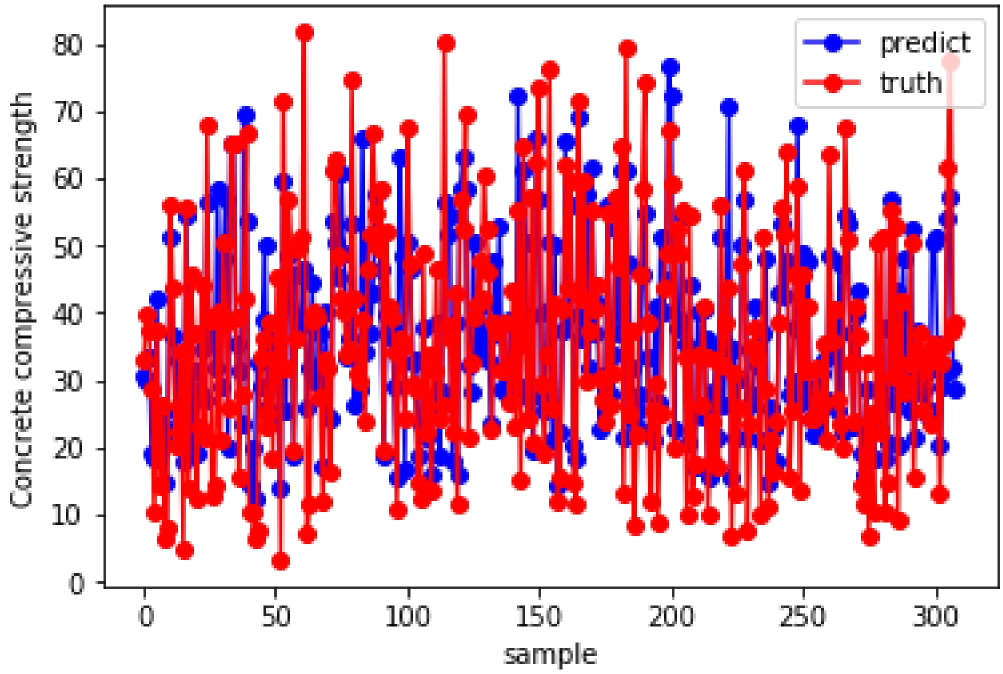
线性回归问题除了使用梯度下降算法外，还可以使用标准方程的方法，其公式如下：

$$\theta = (X^T X)^{-1} X^T y$$

由于本实验的训练集样本个数和样本维度都不是很大，因此可以使用此方法进行求解。

直接通过公式计算出θ，并在测试集上测试，得到$loss$为 54.8387。

模型在测试集上的预测值和测试集的真实值如下图所示：



可见使用标准方程方法得到的模型比基础模型也有所提升。

说明：本实验所有回归算法模型均手写实现，未使用 sklearn 库中封装好的算法模型。