2020 机器学习

实验报告 2

姓名：刘言政

学号：20307130167

一．任务描述

1. 手写以下三种线性回归模型：

   （1） 普通的标准线性回归模型

   （2） Ridge 回归（岭回归）模型

   （3） Lasso 回归模型

2. 在 Boston 房价数据集上测试模型的回归效果。

3. 以可视化的方式，与 python 现有库的回归模型分析比较。

二．数据描述

1. 来源：https://www.kaggle.com/vikrishnan/boston-house-price

   也可以直接通过 sklearn.dataset.load_boston()导入。

2. 简介数据集共包含 506 条数据，每条数据可以看做一个 14×1 的向量。

   前 13 个数值表示房子相关的 13 个属性，即 x；最后一个为房价，即 y。x 包含 13 个属性：CRIM、ZN、INDUS、CHAS、NOX、AGE、DIS、RAD、TAX、RTRATIO、B-1000、LSTAT、MEDV，含义可参照上述链接。

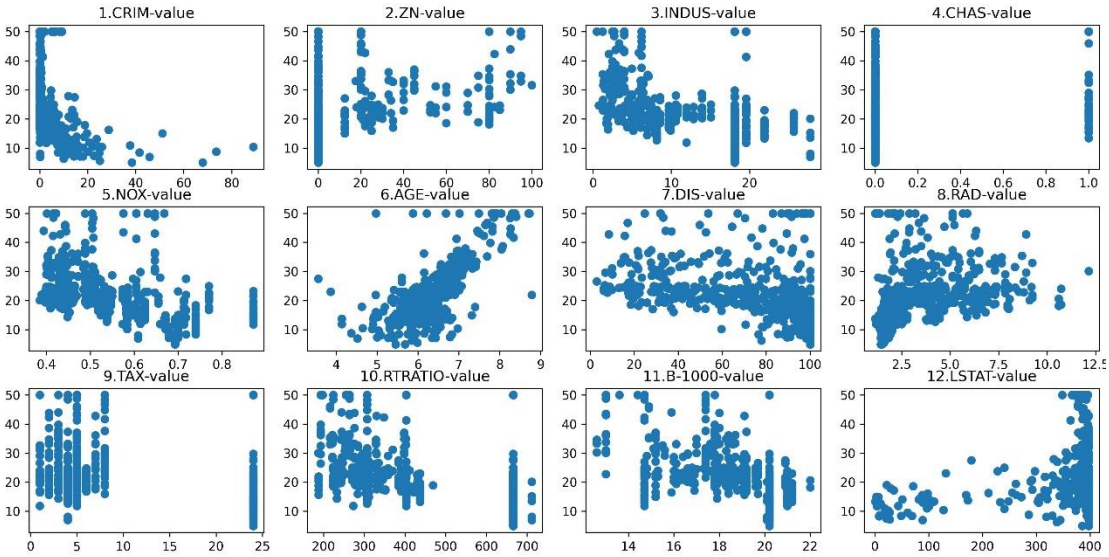
   我们的模型目标即，根据这 13 个属性，预测 y 的值

三．数据集预处理

1. 数据集可视化

   该部分只是针对数据集的可视化处理，用于直观感受 x 的每个属性和 y 的相关性。与后面的算法并无直接关联。

   功能通过 pre_vsl.py 下的函数实现。也可直接运行该文件。



2. 载入并划分数据集

数据集的载入在 pre_work.py 文件下实现。

```
def load_dataset():
    # 获取数据集
    # x 表示房屋属性，共 13 项，y 表示房价
    train_x, train_y = datasets.load_boston(return_X_y = True)
    return get_array(train_x,train_y)
```

该函数实现数据集的导入，同时会将数组转换为 numpy.array 的形式。

```
def load_trainset():
    X, Y = load_dataset()
    return X[0:450], Y[0:450]

def load_testset():
    X, Y = load_dataset()
    return X[451:506], Y[451:506]
```

此处两个函数分别载入训练集和测试集，实现了原数据集的划分。

四．算法介绍

1. Linear 回归

普通的线性回归方法。通过最小化损失函数：

$$J(a,b)=\sum_{i=1}^{n}(f(x^{(i)})-y^{(i)})^2=\sum_{i=1}^{n}(ax^{(i)}+b-y^{(i)})^2$$

来实现线性回归。由于该问题已存在数学上的最优解，即：

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

所以线性回归的实现相较简单，直接代入公式即可。

```
class Linear:
    def __init__(self):
        pass

    def fit(self, x, y):
        m = x.shape[0]
        self.x = np.concatenate((np.ones((m,1)),x),axis=1)
        self.y = copy.copy(y)
        self.w =
np.linalg.inv(np.transpose(self.x).dot(self.x)).dot(np.transpo
se(self.x)).dot(self.y)

    def predict(self, x):
        m = x.shape[0]
        X = np.concatenate((np.ones((m,1)),x),axis=1)
        y = X.dot(self.w)
        return y
```

2. Ridge 回归

线性回归在数据集较小的情况下，尤其是数据集不足以概括全部数据特征的情况下，由于矩阵不满秩，存在较明显的缺陷。因此，ridge 回归改善了损失函数：

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

与普通的线性回归一样，该回归方程也存在数学上的最优解，公式可直接给出：

$$w = (X^T X + \lambda I)^{-1} X^T y$$

λ为岭系数。

具体的实现如下：

```
class Ridge:
    def __init__(self):
        pass

    def fit(self, x, y, lmd = 0.2):
        m = x.shape[0]
        self.x = np.concatenate((np.ones((m,1)),x),axis=1)
        self.y = copy.copy(y)
        idty =  np.identity(len(self.x[0]))
        self.w =
np.linalg.inv(np.transpose(self.x).dot(self.x)-
lmd*idty).dot(np.transpose(self.x)).dot(self.y)

    def predict(self, x):
        m = x.shape[0]
        X = np.concatenate((np.ones((m,1)),x),axis=1)
        y = X.dot(self.w)
        return y
```

3. Lasso 回归

Lasso 回归是改善普通线性回归的缺陷的另一种方式，其损失函数如下：

$$\text{Cost}(w) = \sum_{i=1}^{N} \left(y_i - w^T x_i\right)^2 + \lambda \|w\|_2^2$$

与前两者不同的是，lasso 回归的损失函数并非处处可导，这也就意味着不能通过前两者相似的方法，直接求出数学上的最优解。所以我们使用梯度下降法来逼近最优解：

```
class Lasso:
    def __init__(self):
        pass

    def fit(self, x, y, lmd = 0.2, learning_rate = 0.000005,
epochs = 50000):
        m = x.shape[0]
        self.x = np.concatenate((np.ones((m,1)),x),axis=1)
```

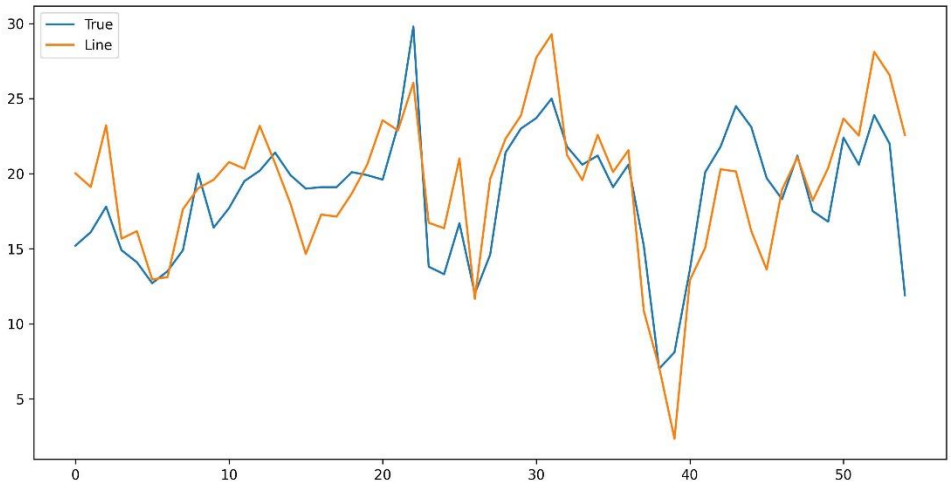
```
        self.y = copy.copy(y)

        xMAT = np.mat(self.x)
        yMAT = np.mat(self.y.reshape(-1,1))

        self.w = np.ones(self.x.shape[1]).reshape(-1,1)

        for i in range(epochs):
            gradient = xMAT.T * (xMAT*self.w - yMAT)/m +
lmd*np.sign(self.w)
            self.w = self.w -  learning_rate*gradient


    def predict(self, x):
        m = x.shape[0]
        X = np.concatenate((np.ones((m,1)),x),axis=1)
        y = X.dot(self.w)
        return y
```

此处，有三个参数可以调整。实验过程中发现，学习率过高时，回归模型无法正常工作，数组中会出现 Nan（not a number）的情况。当学习率小于 0.00001 时，在 Boston 数据集上才能够正常工作。所以此处选择了默认学习率为 0.000005。关于 lmd 的分析，将在后续部分提到。

五．实验结果及分析

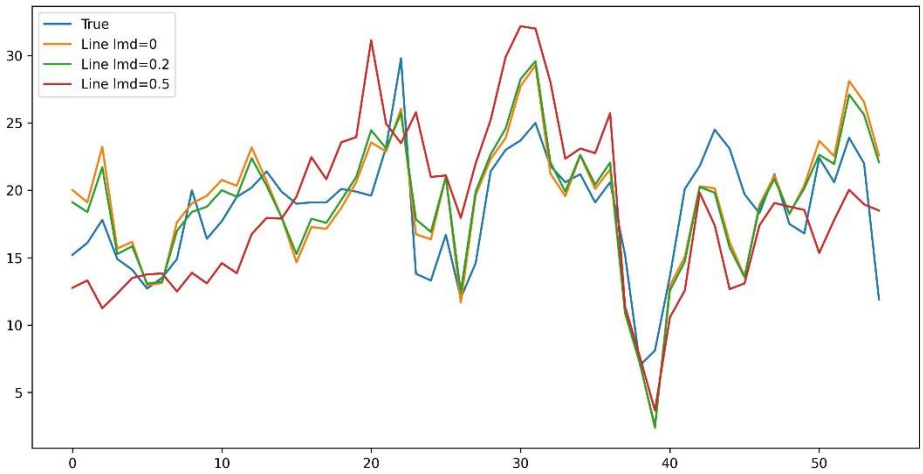
1. 回归结果可视化

（1） 线性回归



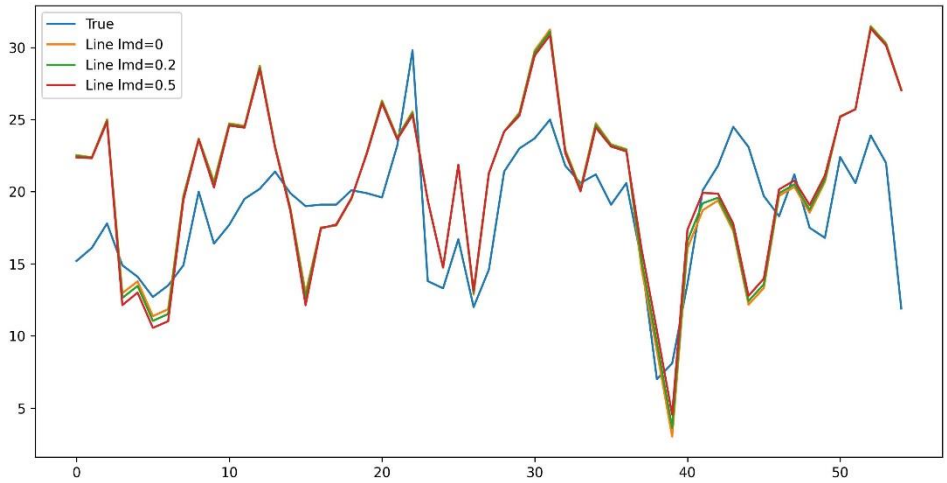
回归结果较好，符合真实数据集的变化趋势，与真实值接近。

（2） Ridge 回归



整体上回归结果较好。对比不同的 lmd（λ）参数，当 lmd 取值为 0.2 时，回归结果的 score 最高。故模型中 lmd 参数默认值设置为 0.2。

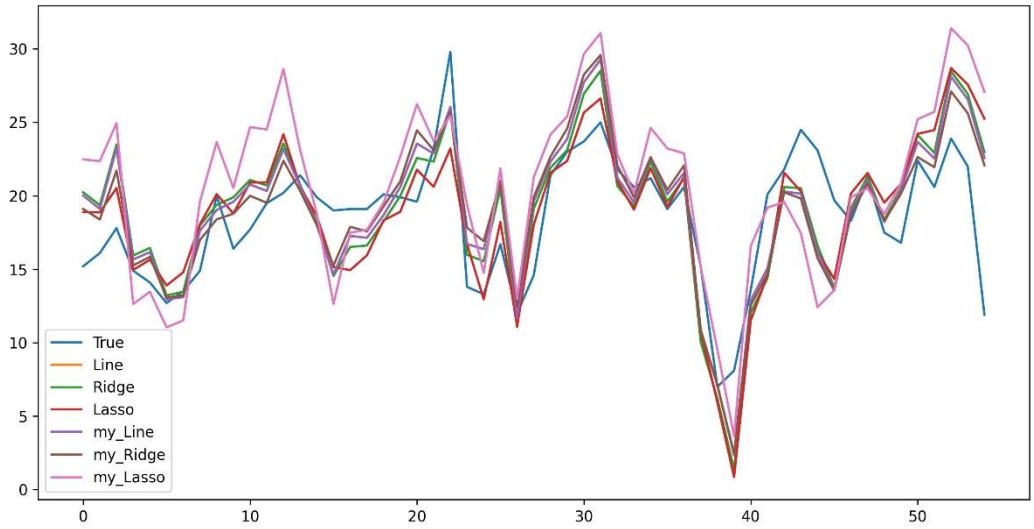
（3） Lasso 回归



回归结果与前两者相比略逊一筹。一部分原因在于该方法使用了梯度下降的方法，与前两者有本质上的差异，且在参数的调整上还有较大的提升空间。Lmd 的取值对回归结果的影响并不大。模型中默认值取 0.2。

2. 与 python 已有函数库的模型对比

相关代码在 test.py 文件下。



直观上来看，6 个回归模型的优劣并不明显。所以我们求助于 sklearn 的模型打分机制：

```
model score:
line:0.37205273544068374
ridge:0.34416584474343337
lasso:0.30566596479736396
my_line:0.37205273544067863
my_ridge:0.4053997139387757
my_lasso:-0.35540624942966836
```

可以看到：

（1） 普通的线性回归模型，官方库和自写的模型差异极小，可以忽略不计。

（2） 自写的 ridge 回归模型在 Boston 数据集上的回归效果最优，远超其他模型。

（3） 自写的 lasso 回归模型的回归效果最差，远低于其他模型。

六．总结

1. 线性回归模型是一种常用的简单机器学习算法，在线性相关性较高的场景下具有很好的回归效果。但在数据集较小的情况下不可行。
2. Ridge 回归和 lasso 回归都是对普通线性回归缺陷做出改进的回归模型。其中 ridge 回归存在理论上的最优解，但 lasso 回归由于损失函数不可导的原因，只能使用梯度下降法等方法逼近最优解。
3. 三种回归模型在 Boston 房价数据集上均具有良好的表现。