Python 与线性回归实践篇

#机器学习

线性回归是机器学习中最为基础的算法之一，因为 NumPy 库的存在，使用 Python 可以简单快速的编写出线性回归代码。
为了方便，先写一篇实践篇，下一篇我们将根据结论来回到理论。

SKLearn 的线性回归

Python 的 SKLearn 库中提供了线性回归模型，能够实现在给定的一组数据中拟合一条直线来预测输出值，使用方法非常简单。

```
#!/usr/bin/python

# 导入 NumPy 库和 LinearRegression 库
import numpy as np
from sklearn.linear_model import LinearRegression

# 随机假定两组数据，a 代表房屋的面积和楼层，b 代表房价
a = [[100, 10],[80, 2],[120, 7],[75, 6],[60, 11],[43, 1],[140, 5],[132, 9],[63, 9],[55, 13],[74, 1],[44, 8],[88, 12]]
b = [[120],[92],[143],[87],[60],[50],[167],[147],[80],[60],[90],[57],[99]]

# 创建一个回归模型
model = LinearRegression()

# 拟合数据
model.fit(a, b)

# 给定一组预测参数，120 平米，5 楼的一栋房屋
model.predict([[120, 5]])
```

最终的输出为：

```
Output: array([[140.97248685]])
```

给定的这一组数据的预测值已经被输出了，我们用一组已知的值来测试一下拟合的模型：

```
# 这是一组已知的参数，真实值为 87
model.predict([[75, 6]])
```

输出为：

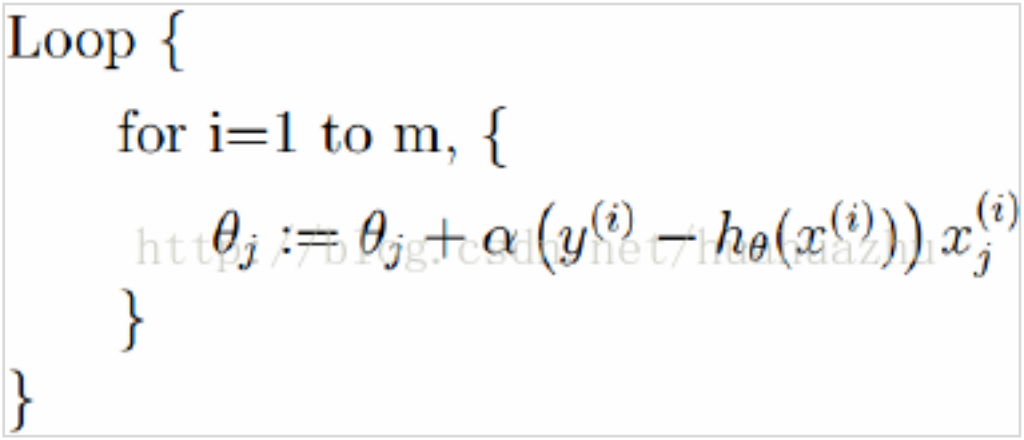
```
Output: array([[87.97877034]])
```

可以看到，我们的预测值非常接近真实值。

随机梯度下降

SKLearn 的线性回归模型使用的是最小二乘法来拟合数据，接下来我们来试一试使用随机梯度下降来拟合数据。

首先回顾一下随机梯度下降的公式：



和批量梯度下降不同的是，随机梯度下降每次只针对一个特征或一组数据进行梯度下降，用 Python 实现的代码如下：

```
# 将数据转换成 nparray
npA = np.array(a)
npB = np.array(b)
m, n = npA.shape

# 提供学习速率和初始权重
```

```
# 这里的学习速率取值已经经过优化，若取其他值例如 0.001，会导致结果无法收敛
alpha = 0.00005
theta = np.zeros(n)

# 设置一个最大循环次数防止无限循环
loop_time = 10000
loop = 0

while loop < loop_time:
    loop += 1

    for i in range(m):
        # 计算损失函数
        cost = np.dot(theta, npA[i]) - npB[i]

        # 在随机梯度下降中，每次只针对一个特征进行或一组数据梯度下降
        theta = theta - alpha * cost * npA[i]

print(np.dot(np.array([[88, 12]]), theta))
```

最终输出结果是：

```
# 我们使用了一组已知数据，真实值是 99，下面是预测值：
Output: [100.89734569]
```