# 归一化其实由一个缺点

|  |
| --- |
|  |
|  |

## 归一化这种方法不太通用

# 标准化

|  |
| --- |
|  |

## 和归一化对异常值的处理对比

|  |
| --- |
|  |
|  |

## 使用到的api模块

|  |
| --- |
|  |

## 使用方法

|  |
| --- |
|  |

## 使用场景

|  |
| --- |
|  |

## 实例,还是使用dating.txt的数据.代码如下

### dating.txt的内容

|  |
| --- |
| milage,Liters,Comsumtime,target  40920,8.326976,0.953952,3  14488,7.153469,1.673904,2  26052,1.441871,0.805124,1  75136,13.147394,0.428964,1  28344,1.669788,0.134296,1  30920,8.326976,0.953952,3  24488,7.153469,1.673904,2  20052,1.441871,0.805124,1  45136,13.147394,0.428964,1  18344,1.669788,0.134296,1  20920,8.326976,0.953952,3  10488,7.153469,1.673904,2  20052,1.441871,0.805124,1  55136,13.147394,0.428964,1  68344,1.669788,0.134296,1  30920,8.326976,0.953952,3  24488,7.153469,1.673904,2  29052,1.441871,0.805124,1  45136,13.147394,0.428964,1  58344,1.669788,0.134296,1 |

### 代码如下

|  |
| --- |
| # 标准化 import pandas as pd from sklearn.preprocessing import StandardScaler  # 1.加载数据 data = pd.read\_csv("./dating.txt") # 2.获取数据的前三列,因为第四列是目标值不能作为特征值 data = data.iloc[:,:3] # print(data) # 3.创建转换器 transfer = StandardScaler() # 4.调用标准化方法 ret\_data = transfer.fit\_transform(data) print(ret\_data) |

### 效果

|  |
| --- |
| [[ 0.37194576 0.44990013 0.29573441]  [-1.12171426 0.18312874 1.67200507]  [-0.468238 -1.11527928 0.01123265]  [ 2.305476 1.54571769 -0.70784025]  [-0.33871815 -1.06346729 -1.27113187]  [-0.19314959 0.44990013 0.29573441]  [-0.55661892 0.18312874 1.67200507]  [-0.80729521 -1.11527928 0.01123265]  [ 0.61018996 1.54571769 -0.70784025]  [-0.9038135 -1.06346729 -1.27113187]  [-0.75824494 0.44990013 0.29573441]  [-1.3477524 0.18312874 1.67200507]  [-0.80729521 -1.11527928 0.01123265]  [ 1.1752853 1.54571769 -0.70784025]  [ 1.92166324 -1.06346729 -1.27113187]  [-0.19314959 0.44990013 0.29573441]  [-0.55661892 0.18312874 1.67200507]  [-0.2987094 -1.11527928 0.01123265]  [ 0.61018996 1.54571769 -0.70784025]  [ 1.35656789 -1.06346729 -1.27113187]] |

# 这个方法比归一化常用