**《系统工程导论》黑箱建模作业1**

**一元线性回归（F检验）**

**【算法原理】**

在最小二乘的意义下，使得误差平方和最小：



得到：



显著性检验：原假设：X，Y 没有线性关系。若，认为原假设不成立，即X,Y存在线性关系；否则，接受原假设，认为X,Y不存在线性关系。

**【算法实现】**

第一步：求得回归直线

第二步：打印回归直线方程

第二步：进行显著性检验，是否满足线性关系；若满足，继续；否则：退出

第三步：求出置信区间；画图

【难点与收获】

本次作业使用了Python进行实现，在查询相关的统计函数（比如查询如何获得F检验的值）时花费了较多的时间。其余步骤实现较为顺利。

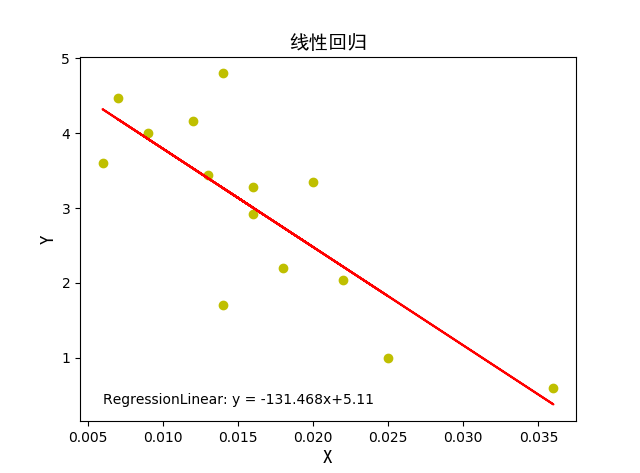
**【算法演示】**

数据：

从某矿石中取得14块样品，测得成分A和成分B的含量如下表所示。试分析矿石中成分A和成分B的含量之间是否存在线性关系。

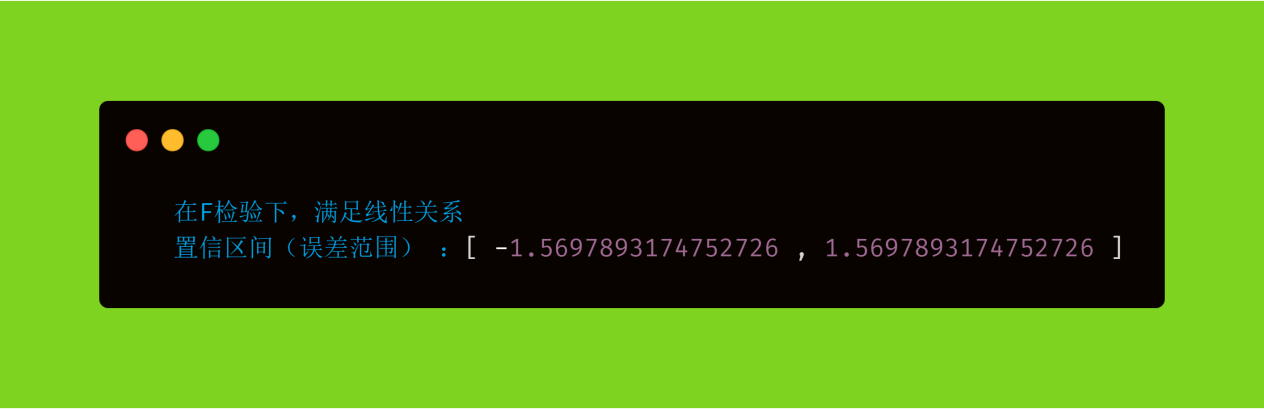
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **编号** | **成分A(x)** | **成分B(y)** | **编号** | **成分A(x)** | **成分B(y)** |
| **1** | 0.009 | 4.0 | **8** | 0.014 | 1.7 |
| **2** | 0.013 | 3.44 | **9** | 0.016 | 2.92 |
| **3** | 0.006 | 3.6 | **10** | 0.014 | 4.8 |
| **4** | 0.025 | 1.0 | **11** | 0.016 | 3.28 |
| **5** | 0.022 | 2.04 | **12** | 0.012 | 4.16 |
| **6** | 0.007 | 4.74 | **13** | 0.020 | 3.35 |
| **7** | 0.036 | 0.6 | **14** | 0.018 | 2.2 |

回归直线：

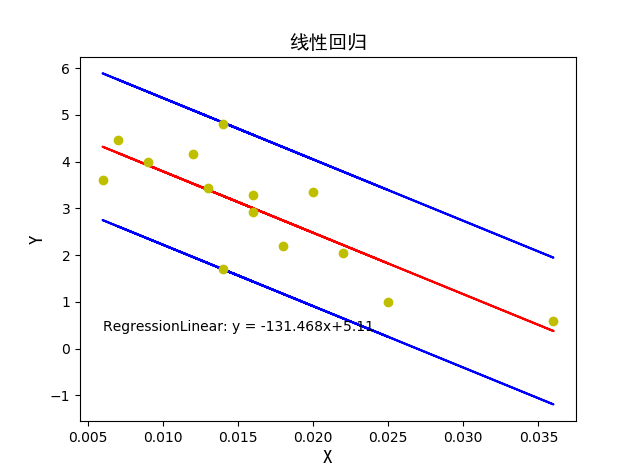


显著性检验结果：

输出：



打印：



【代码】

# -\*- coding: utf-8 -\*-  
# 张嘉玮  
# 20190403  
**import** numpy **as** np  
**from** scipy.stats **import** f  
**from** scipy.stats **import** norm  
**import** matplotlib.pyplot **as** plt  
  
**def linear\_regression1(**data,alpha**):** N **=** len**(**data**)** Y **=** np.array**([**data**[**i**][**0**] for** i **in** range**(**N**)])** X **=** np.array**([**data**[**i**][**1**] for** i **in** range**(**N**)])** Y\_ **=** np.mean**(**Y**)** X\_ **=** np.mean**(**X**)** L\_xx **=** np.sum**([**i**\***i **for** i **in** X**])-**np.square**(**np.sum**(**X**))/**N  
 L\_xy **=** np.sum**(**np.multiply**(**X,Y**)) -** np.sum**(**X**)\***np.sum**(**Y**)/**N  
 L\_yy **=** np.sum**([**j**\***j **for** j **in** Y**]) -** np.square**(**np.sum**(**Y**))/**N  
  
 b **=** L\_xy**/**L\_xx  
 a **=** Y\_ **-** b**\***X\_  
  
 # 打印出回归直线的方程  
 plt.figure**(**1**)** plt.plot**(**X,Y,**'yo')** Y\_pre **=** np.array**([**b**\***i**+**a **for** i **in** X**])** plt.plot**(**X,Y\_pre,**'r')** Text **= 'RegressionLinear: y = '+**str**(**round**(**b,3**))+'x+'+**str**(**round**(**a,3**))** plt.text**(**min**(**X**)**, min**(**Y\_pre**)**, Text, wrap**=True)** plt.title**(u'线性回归'**, fontproperties**='SimHei'**, fontsize**=**14**)** plt.xlabel**(u'X'**, fontproperties**='SimHei'**, fontsize**=**14**)** plt.ylabel**(u'Y'**, fontproperties**='SimHei'**, fontsize**=**14**)** plt.show**()** #F检验：根据F表，获得临界值  
 F **=** f.ppf**(**alpha,1,N**-**2**)** U **=** b**\***L\_xy  
 Q **=** L\_yy **-** U  
 F\_ **=** U**\*(**N**-**2**)/**Q  
  
 # 满足线性关系  
 **if** F\_ **>** F**:** print**(' 在F检验下，满足线性关系')** S\_a **=** np.sqrt**(**np.sum**([**i**\***i **for** i **in** Y**-**Y\_pre**])/(**N**-**2**))** sig **=** norm.ppf**(**1**-**alpha**/**2**)** Y\_max **=** Y\_pre**+**sig**\***S\_a  
 Y\_min **=** Y\_pre**-**sig**\***S\_a  
 print**(' 置信区间（误差范围） ：['**,**-**sig**\***S\_a,**','**,sig**\***S\_a,**']')** plt.figure**(**2**)** Text **= 'RegressionLinear: y = ' +** str**(**round**(**b, 3**)) + 'x+' +** str**(**round**(**a, 3**))** plt.text**(**min**(**X**)**, min**(**Y\_pre**)**, Text, wrap**=True)** plt.plot**(**X,Y\_pre,**'r')** plt.plot**(**X,Y\_max,**'b')** plt.plot**(**X,Y\_min,**'b')** plt.plot**(**X,Y,**'yo')** plt.title**(u'线性回归'**,fontproperties**='SimHei'**, fontsize**=**14**)** plt.xlabel**(u'X'**, fontproperties**='SimHei'**, fontsize**=**14**)** plt.ylabel**(u'Y'**, fontproperties**='SimHei'**, fontsize**=**14**)** plt.show**()** # 不满足线性关系  
 **else:** print**(' 在F检验下，不满足线性关系')**data**= [[**4,0.009**]**,**[**3.44,0.013**]**,**[**3.6,0.006**]**,**[**1.0,0.025**]**,**[**2.04,0.022**]**,**[**4.47,0.007**]**,**[**0.6,0.036**]**,**[**1.7,0.014**]**,**[**2.92,0.016**]**,**[**4.8,0.014**]**,**[**3.28,0.016**]**,**[**4.16,0.012**]**,**[**3.35,0.020**]**,**[**2.2,0.018**]]**linear\_regression1**(**data,0.05**)**