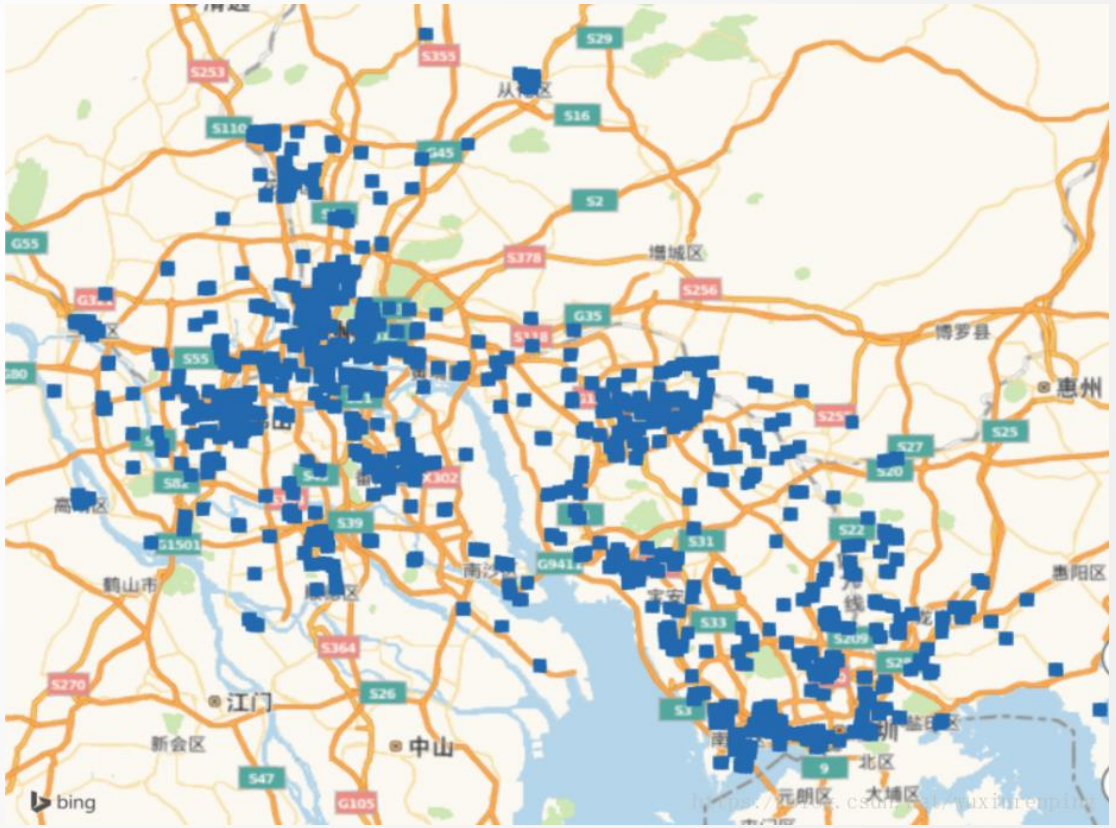
实战练习 - 众包任务

“拍照赚钱”是移动互联网下的一种自助式服务模式。用户下载APP，注册成为APP会员，然后从APP上领取需要拍照的任务（比如上超市去检查某种商品的上架情况），赚取APP对任务所标定的酬金。

这种基于移动互联网的自助式劳务众包平台，为企业提供各种商业检查和信息搜集，相比传统的市场调查方式可以大大节省调查成本，而且有效地保证了调查数据真实性，缩短了调查的周期。因此APP成为该平台运行的核心，而APP中的任务定价又是其核心要素。如果定价不合理，有的任务就会无人问津，而导致商品检查的失败。



左边是一个已结束项目的任务数据，包含了每个任务的位置、定价和完成情况（“1”表示完成，“0”表示未完成）；

右边是会员信息数据，包含了会员的位置、信誉值、参考其信誉给出的任务开始预订时间和预订限额，原则上会员信誉越高，越优先开始挑选任务，其配额也就越大（任务分配时实际上是根据预订限额所占比例进行配发）

表1 已结束项目任务数据

任务号码	任务gps纬度	任务gps经度	任务标价	任务执行情况
A0001	22.56614225	113.9808368	66	0
A0002	22.68620526	113.9405252	65.5	0
A0003	22.57651183	113.957198	65.5	1
A0004	22.56484081	114.2445711	75	0
A0005	22.55888775	113.9507227	65.5	0
A0006	22.55899906	114.2413174	75	0
A0007	22.54900371	113.9722597	65.5	1
……	……	……	……	……

表2 会员信息数据

会员编号	会员位置(GPS)	预订任务限额	预订任务开始时间	信誉值
B0001	22.947097 113.679983	114	6:30:00	67997.3868
B0002	22.577792 113.966524	163	6:30:00	37926.5416
B0003	23.192458 113.347272	139	6:30:00	27953.0363
B0004	23.255965 113.31875	98	6:30:00	25085.6986

问题：

（1） 对每一个任务，计算该任务在5公里范围内的任务数量总和，记为Z1。

（2） 对每一个任务，计算该任务在5公里范围内的任务平均价格，记为Z2。

（3） 对每一个任务，计算该任务在5公里范围内的会员个数，记为Z3。

（4） 对每一个任务，计算该任务在5公里范围内的会员信誉平均值，记为Z4。

（5） 对每一个任务，计算该任务在5公里范围内的会员可预订任务限额总和，记为Z5。

图7-1所示圆圈代表任务，三角形代表会员，分布在同一个区域上，位置均由经度和纬度确定。以某个任务为圆心，5公里范围为半径，作一个圆，如图中所示。该任务在5公里范围内有4个任务（包括自身）、2个会员。对该任务来讲，则：

Z1=4；

Z2=对应4个任务定价的平均值

Z3=2

Z4=对应2个会员信誉值的平均值

Z5=对应2个会员预订限额的总和

本案例的关键是在计算任务之间、任务与会员之间的距离，从而确定每个任务在5公里范围内具体包括哪些任务和会员，进而就可以计算其指标值了。

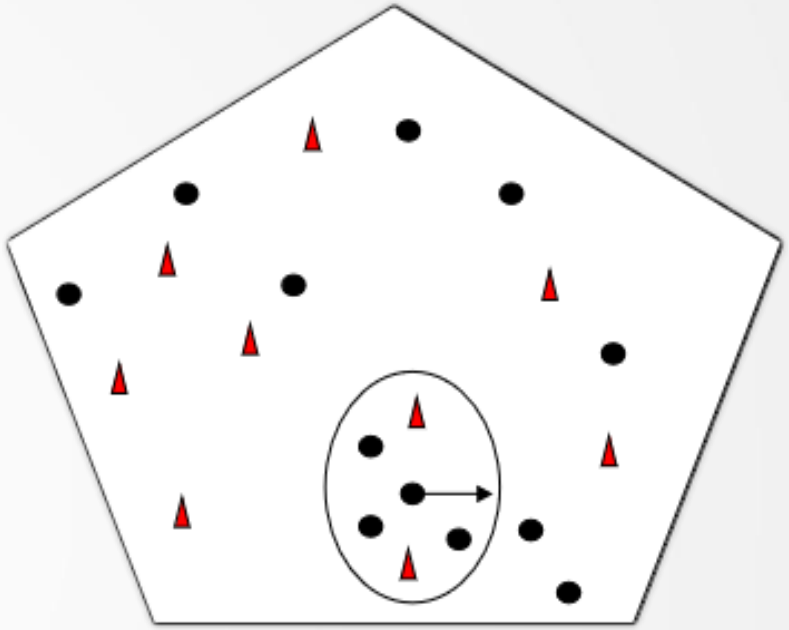


图7-1

设定A点（纬度，经度）和B点（纬度，经度），则两点之间的距离可以用以下公式进行计算：

$$\Delta = 111.199\left[(\varphi_1 - \varphi_2)^2 + (\lambda_1 - \lambda_2)^2 COS^2\left(\frac{\varphi_1 + \varphi_2}{2}\right)\right]^{\frac{1}{2}}$$

其中距离的单位为：公里。

1.小试牛刀：计算第0个任务到第1个任务、第0个任务到第0个会员之间的距离。

这里计算比较简单，在获得给定两个任务、给定一个任务和一个会员的经纬度数据之后，直接利用经纬度距离公式计算即可，属于点对点的计算。示例代码如下：

```
import pandas as pd    #导入pandas库
import numpy as np     #导入nmypy库
import math            #导入数学函数模

A=pd.read_excel('已结束项目任务数据.xls')
B=pd.read_excel('会员信息数据.xlsx')

A_W0=A.iloc[0,1]  #第0个任务的维度
A_J0=A.iloc[0,2]  #第0个任务的经度
A_W1=A.iloc[1,1]  #第1个任务的维度
A_J1=A.iloc[1,2]  #第1个任务的经度
B_WJ=B.iloc[0,1]
I=B_WJ.find(' ',0,len(B_WJ))
B_W0=float(B_WJ[0:I])        #第0个会员的维度
B_J0=float(B_WJ[I:len(B_WJ)]) #第0个会员的经度

#第0个任务到第1个任务之间的距离
d1=111.19*math.sqrt((A_W0-A_W1)**2+(A_J0-
A_J1)**2*math.cos((A_W0+A_W1)*math.pi/180)**2);
#第0个任务到第0个会员之间的距离
d2=111.19*math.sqrt((A_W0-B_W0)**2+(A_J0-
B_J0)**2*math.cos((A_W0+B_W0)*math.pi/180)**2);
print('d1= ',d1)
print('d2= ',d2)
```

执行结果如下：

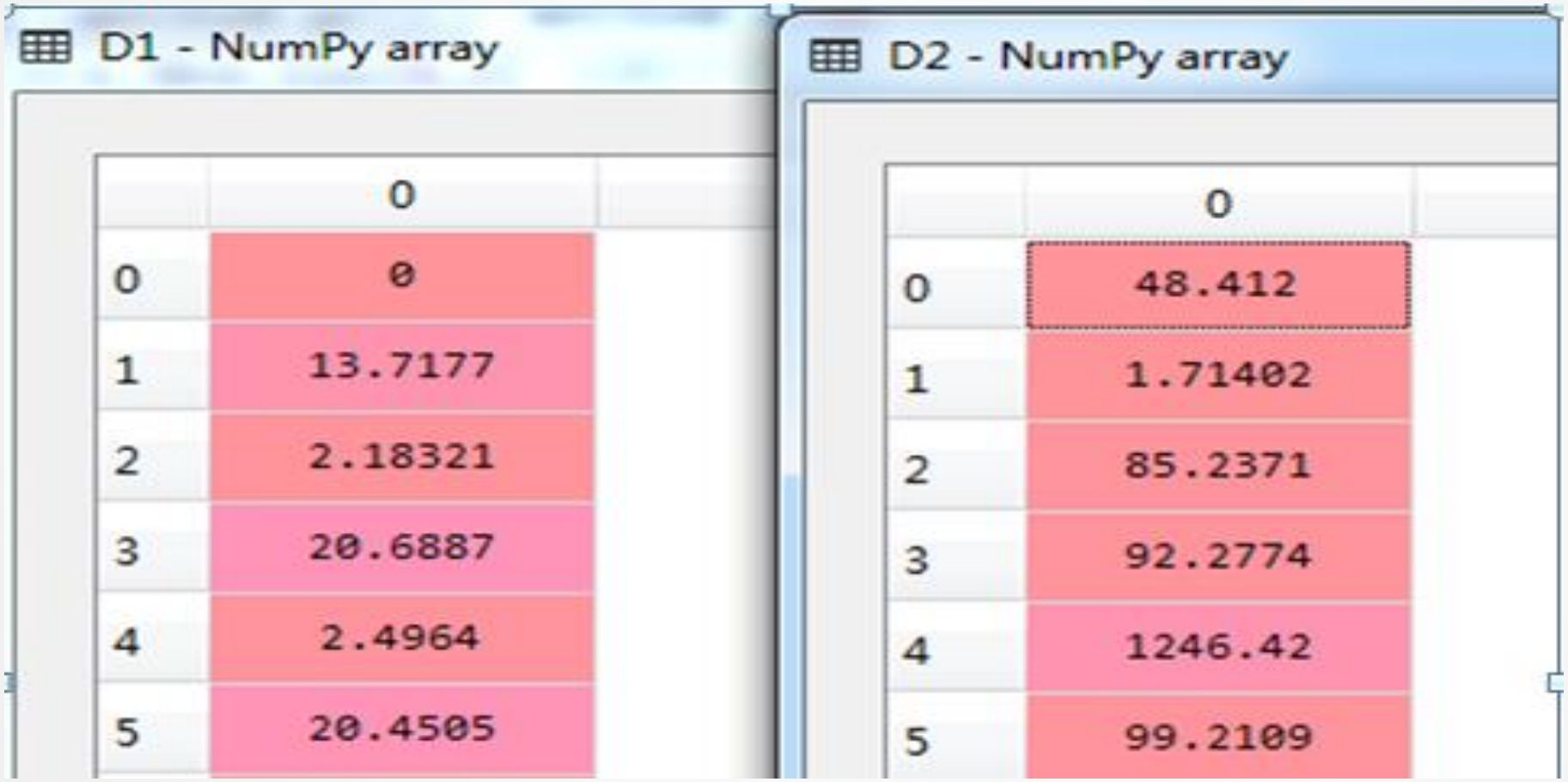
```
d1=  13.71765563354376
d2=  48.41201229628393
```

2.小步进阶：第0个任务与所有任务、所有会员之间的距离。

在前面点对点计算基础上拓展到了点对线的计算，即第0个任务点与所有任务点（线）、第0个任务点与所有会员点（线）之间的距离计算，事实上在前面点对点计算基础上增加一个循环即可实现。示例代码如下：

```
import pandas as pd    #导入pandas库
import numpy as np     #导入nmypy库
import math            #导入数学函数模
A=pd.read_excel('附件一：已结束项目任务数据.xls')
B=pd.read_excel('附件二：会员信息数据.xlsx')
A_W0=A.iloc[0,1]  #第0个任务的维度
A_J0=A.iloc[0,2]  #第0个任务的经度
# 预定义数组D1，用于存放第0个任务与所有任务之间的距离
# 预定义数组D2，用于存放第0个任务与所有会员之间的距离
D1=np.zeros((len(A)))
D2=np.zeros((len(B)))
for t in range(len(A)):
    A_Wt=A.iloc[t,1]  #第t个任务的维度
    A_Jt=A.iloc[t,2]  #第t个任务的经度
    #第0个任务到第t个任务之间的距离
    dt=111.19*math.sqrt((A_W0-A_Wt)**2+(A_J0-A_Jt)**2*math.cos((A_W0+A_Wt)*math.pi/180)**2);
    D1[t]=dt
    for k in range(len(B)):
        B_WJ=B.iloc[k,1]
        I=B_WJ.find(' ',0,len(B_WJ))
        B_Wk=float(B_WJ[0:I])       #第k个会员的维度
        B_Jk=float(B_WJ[I:len(B_WJ)])  #第k个会员的经度
        #第0个任务到第k个会员之间的距离
        dk=111.19*math.sqrt((A_W0-B_Wk)**2+(A_J0-B_Jk)**2*math.cos((A_W0+B_Wk)*math.pi/180)**2);
D2[k]=dk
```

执行结果（部分）如图7-2所示。



3.初步落成：对第0个任务计算指标Z1、Z2、Z3、Z4、Z5。

只需在前面点对线计算结果基础上，根据案例分析中的指标计算方法进行计算即可。示例代码如下：

```
import pandas as pd    #导入pandas库
import numpy as np     #导入nmypy库
import math          #导入数学函数模
A=pd.read_excel('附件一：已结束项目任务数据.xls')
B=pd.read_excel('附件二：会员信息数据.xlsx')
A_W0=A.iloc[0,1]  #第0个任务的维度
A_J0=A.iloc[0,2]  #第0个任务的经度
# 预定义数组D1，用于存放第0个任务与所有任务之间的距离
# 预定义数组D2，用于存放第0个任务与所有会员之间的距离
D1=np.zeros((len(A)))
D2=np.zeros((len(B)))
for t in range(len(A)):
    A_Wt=A.iloc[t,1]  #第t个任务的维度
    A_Jt=A.iloc[t,2]  #第t个任务的经度
#第0个任务到第t个任务之间的距离
    dt=111.19*math.sqrt((A_W0-A_Wt)**2+(A_J0-A_Jt)**2*math.cos((A_W0+A_Wt)*math.pi/180)**2);
    D1[t]=dt
```

```
for k in range(len(B)):
        B_WJ=B.iloc[k,1]
        I=B_WJ.find(' ',0,len(B_WJ))
       B_Wk=float(B_WJ[0:I])        #第k个会员的
维度
      B_Jk=float(B_WJ[I:len(B_WJ)])  #第k个会员
的经度
#第0个任务到第k个会员之间的距离
  dk=111.19*math.sqrt((A_W0-
B_Wk)**2+(A_J0-
B_Jk)**2*math.cos((A_W0+B_Wk)*math.pi/18
0)**2);
    D2[k]=dk
```

```
Z1=len(D1[D1<=5])
Z2=A.iloc[D1<=5,[3]].mean()[0]
Z3=len(D2[D2<=5])
Z4=B.iloc[D2<=5,[2,4]].sum()[0]
Z5=B.iloc[D2<=5,[2,4]].sum()[1]/Z3
print('Z1= ',Z1)
print('Z2= ',Z2)
print('Z3= ',Z3)
print('Z4= ',Z4)
print('Z5= ',Z5)
```

执行结果如下：

```
Z1=  18  Z2=  66.1944444444  Z3=  45  Z4= 548.0  Z5=  1302.32711556
```

4.任务完成：计算所有任务的Z1、Z2、Z3、Z4、Z5。

前面介绍了第0个任务点与所有任务（线）、所有会员（线）之间的计算，在此基础上利用循环即可实现所有任务与所有任务、所有会员之间的指标计算。示例代码如下：

```
import pandas as pd    #导入pandas库
import numpy as np     #导入nmypy库
import math            #导入数学函数模
A=pd.read_excel('附件一：已结束项目任务数据.xls')
B=pd.read_excel('附件二：会员信息数据.xlsx')
# 预定义，存放所有任务的指标Z1、Z2、Z3、Z4、Z5
Z=np.zeros((len(A),5))
for q in range(len(A)):
A_Wq=A.iloc[q,1]  #第q个任务的维度
A_Jq=A.iloc[q,2]  #第q个任务的经度
# 预定义数组D1，用于存放第q个任务与所有任务之间的距离
  # 预定义数组D2，用于存放第q个任务与所有会员之间的距离
  D1=np.zeros((len(A)))
    D2=np.zeros((len(B)))
  for t in range(len(A)):
    A_Wt=A.iloc[t,1]  #第t个任务的维度
    A_Jt=A.iloc[t,2]  #第t个任务的经度
    #第q个任务到第t个任务之间的距离
    dt=111.19*math.sqrt((A_Wq-A_Wt)**2+(A_Jq-A_Jt)**2*math.cos((A_Wq+A_Wt)*math.pi/180)**2);
    D1[t]=dt
```

```
for k in range(len(B)):
        B_WJ=B.iloc[k,1]
        I=B_WJ.find(' ',0,len(B_WJ))
        B_Wk=float(B_WJ[0:I])
#第k个会员的维度
        B_Jk=float(B_WJ[I:len(B_WJ)])
 #第k个会员的经度
 #第q个任务到第k个会员之间的距离
  dk=111.19*math.sqrt((A_Wq-
B_Wk)**2+(A_Jq-
B_Jk)**2*math.cos((A_Wq+B_Wk)*math.pi/18
0)**2);
   D2[k]=dk
```

```
Z1=len(D1[D1<=5])
Z2=A.iloc[D1<=5,[3]].mean()[0]
Z3=len(D2[D2<=5])
Z4=B.iloc[D2<=5,[2,4]].sum()[0]
Z5=B.iloc[D2<=5,[2,4]].sum()[1]/Z3
Z[q,0]=Z1
Z[q,1]=Z2
Z[q,2]=Z3
Z[q,3]=Z4
Z[q,4]=Z5
```

执行结果（部分）如图7-3所示。

Z - NumPy array

	0	1	2	3	4
0	18	66.1944	45	548	1302.33
1	9	68.3889	43	152	124.27
2	24	65.875	60	679	1014.04
3	2	75	3	3	0.371233
4	36	65.9722	72	756	853.326
5	2	75	5	9	8.19198
6	31	65.9516	62	655	958.302
7	31	65.9194	70	764	887.689
8	19	65.7105	25	126	64.3293

本案例在指标计算过程中，详细介绍了如何由简单到复杂的程序演化计算过程，也体现了由点到线，再到面的的编程思想。点：即第0个任务与第1个任务、第0个会员之间距离的点对点的计算。线：即第0个任务与所有任务、所有会员之间的由点到线的计算。面：即所有任务与所有任务、所有会员之间的线到面的计算。这种由简单到复杂的程序演化编程思想，对编程具有非常重要的作用。

- 如何根据5公里内的相关数据来确定最佳的任务价格呢?

1）基于给定的**数据文件**，对每一个任务，计算以下指标（12个）：

Z1 = 5公里范围内的其他任务量
Z2 = 5公里范围内的任务平均价格
Z3 = 5公里范围内会员可预订任务量
Z4 = 5公里范围内会员平均信誉值
Z5 = 5公里范围内会员个数
Z6 = 5公里范围内6:30发布的可预订任务量
Z7 = 5公里范围内6:33-6:45发布的可预订任务量
Z8 = 5公里范围内6:48-7:03发布的可预订任务量
Z9 = 5公里范围内7:06-7:21发布的可预订任务量
Z10 = 5公里范围内7:24-7:39发布的可预订任务量
Z11 = 5公里范围内7:42-7:57发布的可预订任务量
Z12 = 5公里范围内8:00发布的可预订任务量

- 如何根据5公里内的相关数据来确定最佳的任务价格呢?

2）对以上计算的12个指标进行**相关性分析**。如果多个指标之间存在较强的相关性，请对这12个指标进行主成分分析，并提取主成分（要求累计贡献率在90%以上），同时写出每个主成分的表达式。

3）针对**被执行的任务**，以提取的主成分作为自变量，任务定价作为因变量，研究自变量与因变量之间的关系，线性还是非线性？如果是线性的，则构建多元线性回归模型；如果是非线性的，则构造BP神经网络回归模型。同时对模型进行训练。

4）针对**未被执行的任务**，以提取的主成分作为自变量，利用上一步选择的模型，重新预测其任务定价

- 如何根据5公里内的相关数据来确定最佳的任务价格呢?

5）针对**所有任务**，以Z1~Z12和原来的定价作为自变量（共13个），执行情况为因变量。训练支持向量机分类模型

6）针对**未执行任务**，以Z1~Z12和重新预测的任务定价作为自变量（共13个），利用上一步训练好的支持向量机分类模型进行预测，获得其执行情况。

7）模型评价：即针对**未执行任务**，重新预测的任务定价比原来的任务定价增加了多少？被支持向量机预测为执行的任务有多少，即增加了多少个任务被执行了？

- 如何根据5公里内的相关数据来确定最佳的任务价格呢?

第九次作业提交要求:

- 必须用Jupyter Notebook完成，保存为ipynb格式提交
- 提交文件名为“Python数据分析第9次作业+班级+姓名.ipynb”
- 提交到刘顿同学的邮箱 810792334@qq.com
- 提交的截止时间为2019年5月7日下午17:00

评分标准:

（1）模型预测为“执行”的任务数增加数量 >= 30;

（2）模型重新预测的任务定价总额增加额度 <= 50;