单因子数据探索-对每列值进行分析

In [91]:

```
import pandas as pd
import numpy as np
df = pd.read_csv(’./data/HR.csv’)
```

In [92]:

```
df
```

Out[92]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	time_spenc
0	0.38	0.53	2	157	
1	0.80	0.86	5	262	
2	0.11	0.88	7	272	
3	0.72	0.87	5	223	
4	0.37	0.52	2	159	
...	...	...	...	...	
14997	0.11	0.96	6	280	
14998	0.37	0.52	2	158	
14999	NaN	0.52	2	158	
15000	NaN	999999.00	2	158	
15001	0.70	0.40	2	158	

15002 rows × 10 columns

第一列：满意度satisfaction_level分析

In [3]:

```
sl_s = df['satisfaction_level']
sl_s
```

Out[3]:

```
0        0.38
1        0.80
2        0.11
3        0.72
4        0.37
         ...
14997    0.11
14998    0.37
14999     NaN
15000     NaN
15001    0.70
Name: satisfaction_level, Length: 15002, dtype: float64
```

In [4]:

```
# 异常值分析
sl_s.isnull()
```

Out[4]:

```
0        False
1        False
2        False
3        False
4        False
         ...
14997    False
14998    False
14999     True
15000     True
15001    False
Name: satisfaction_level, Length: 15002, dtype: bool
```

In [5]:

```
# 异常值个数
sl_s[sl_s.isnull()]
```

Out[5]:

```
14999   NaN
15000   NaN
Name: satisfaction_level, dtype: float64
```

In [6]:

```
# 查看异常值数据
df[sl_s.isnull()]
```

Out[6]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	time_spenc
14999	NaN	0.52	2	158	
15000	NaN	999999.00	2	158	

丢掉空值

In [7]:

```
# 丢弃异常值
sl_s = sl_s.dropna()
sl_s
```

Out[7]:

```
0        0.38
1        0.80
2        0.11
3        0.72
4        0.37
         ...
14995    0.37
14996    0.37
14997    0.11
14998    0.37
15001    0.70
Name: satisfaction_level, Length: 15000, dtype: float64
```

In [8]:

```
# 查看均值
sl_s.mean()
```

Out[8]:

```
0.6128393333333333
```

In [9]:

```
# 标准差
sl_s.std()
```

Out[9]:

```
0.24862338135944925
```

In [10]:

```
# 最大值
sl_s.max()
```

Out[10]:

```
1.0
```

In [11]:

```
# 最小值
sl_s.min()
```

Out[11]:

```
0.09
```

In [12]:

```
# 中位数
sl_s.median()
```

Out[12]:

```
0.64
```

In [13]:

```
# 下四分位数
sl_s.quantile(q=0.25)
```

Out[13]:

```
0.44
```

In [14]:

```
# 上四分位数
sl_s.quantile(q=0.75)
```

Out[14]:

```
0.82
```

In [15]:

```
# 偏态系数-偏度
# 负偏，均值偏小，大部分数比均值大
sl_s.skew()
```

Out[15]:

```
-0.47643761717258093
```

In [16]:

```
# 峰态系数-峰度
sl_s.kurt()
```

Out[16]:

```
-0.6706959323886252
```

直方图分布分析

In [17]:

```
# 获取这个指标离散化的分布
# 获取分布的数字,bins指切分大小
# 直方图数字查看，每两个间隔有多少值
np.histogram(sl_s.values,bins=np.arange(0.0,1.1,0.1))
```

Out[17]:

```
(array([ 195, 1214,  532,  974, 1668, 2146, 1973, 2074, 2220, 2004],
       dtype=int64),
 array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ]))
```

第二列：过去评估last_evaluation分析

In [18]:

```
le_s = df['last_evaluation']
le_s
```

Out[18]:

```
0             0.53
1             0.86
2             0.88
3             0.87
4             0.52
            ...
14997         0.96
14998         0.52
14999         0.52
15000    999999.00
15001         0.40
Name: last_evaluation, Length: 15002, dtype: float64
```

In [19]:

```
le_s[le_s.isnull()]
```

Out[19]:

```
Series([], Name: last_evaluation, dtype: float64)
```

In [20]:

```
le_s.mean()
```

Out[20]:

```
67.37373216904412
```

In [21]:

```
le_s.std()
```

Out[21]:

```
8164.407523745649
```

In [22]:

```
le_s.median()
```

Out[22]:

```
0.72
```

In [23]:

```
le_s.max()
```

Out[23]:

```
999999.0
```

In [24]:

```
le_s.min()
```

Out[24]:

```
0.36
```

In [25]:

```
le_s.skew()
```

Out[25]:

```
122.48265175204614
```

In [26]:

```
le_s.kurt()
```

Out[26]:

```
15001.999986807796
```

In [27]:

```
le_s[le_s>1]
```

Out[27]:

```
15000    999999.0
Name: last_evaluation, dtype: float64
```

筛选掉在计算范围外的异常值

In [28]:

```
q_low = le_s.quantile(q=0.25)
q_high = le_s.quantile(q=0.75)
# 四分位间距
q_interval = q_high - q_low
# k为系数
k = 1.5
# 筛选异常值
le_s = le_s[le_s<q_high+k*q_interval][le_s>q_low-k*q_interval]
le_s
```

Out[28]:

```
0        0.53
1        0.86
2        0.88
3        0.87
4        0.52
         ...
14996    0.53
14997    0.96
14998    0.52
14999    0.52
15001    0.40
Name: last_evaluation, Length: 15001, dtype: float64
```

In [29]:

```
np.histogram(le_s.values,bins=np.arange(0.0,1.1,0.1))
```

Out[29]:

```
(array([   0,    0,    0,  179, 1390, 3396, 2234, 2062, 2752, 2988],
       dtype=int64),
 array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ]))
```

In [30]:

```
le_s.mean()
```

Out[30]:

```
0.7160675954936337
```

In [31]:

```
le_s.std()
```

Out[31]:

```
0.17118464250786233
```

In [32]:

```
le_s.median()
```

Out[32]:

```
0.72
```

In [33]:

```
le_s.max()
```

Out[33]:

```
1.0
```

In [34]:

```
le_s.min()
```

Out[34]:

```
0.36
```

In [35]:

```
le_s.skew()
```

Out[35]:

```
-0.02653253746872579
```

In [36]:

```
le_s.kurt()
```

Out[36]:

```
-1.2390454655108427
```

第三列：工程数量number_project分析

In [38]:

```
np_s = df['number_project']
np_s
```

Out[38]:

```
0        2
1        5
2        7
3        5
4        2
        ..
14997    6
14998    2
14999    2
15000    2
15001    2
Name: number_project, Length: 15002, dtype: int64
```

In [39]:

```
np_s[np_s.isnull()]
```

Out[39]:

```
Series([], Name: number_project, dtype: int64)
```

In [40]:

```
np_s.mean()
```

Out[40]:

```
3.8026929742700974
```

In [41]:

```
np_s.std()
```

Out[41]:

```
1.232732779200601
```

In [42]:

```
np_s.median()
```

Out[42]:

```
4.0
```

In [43]:

```
np_s.max()
```

Out[43]:

```
7
```

In [44]:

```
np_s.min()
```

Out[44]:

```
2
```

In [45]:

```
np_s.skew()
```

Out[45]:

```
0.3377744235231047
```

In [46]:

```
np_s.kurt()
```

Out[46]:

```
-0.49580962709450604
```

结构分析--静态结构分析

In [47]:

```
# 计算每个数字出现多少次
np_s.value_counts()
```

Out[47]:

```
4    4365
3    4055
5    2761
2    2391
6    1174
7     256
Name: number_project, dtype: int64
```

In [48]:

```
# 计算构成和每个数的比例
np_s.value_counts(normalize=True)
```

Out[48]:

```
4    0.290961
3    0.270297
5    0.184042
2    0.159379
6    0.078256
7    0.017064
Name: number_project, dtype: float64
```

In [49]:

```
np_s.value_counts(normalize=True).sort_index()
```

Out[49]:

```
2    0.159379
3    0.270297
4    0.290961
5    0.184042
6    0.078256
7    0.017064
Name: number_project, dtype: float64
```

第四列：平均每月工作时间average_monthly_hours分析

In [50]:

```
amh_s = df['average_monthly_hours']
amh_s
```

Out[50]:

```
0        157
1        262
2        272
3        223
4        159
        ...
14997    280
14998    158
14999    158
15000    158
15001    158
Name: average_monthly_hours, Length: 15002, dtype: int64
```

In [51]:

```
amh_s.mean()
```

Out[51]:

```
201.0417277696307
```

In [52]:

```
amh_s.std()
```

Out[52]:

```
49.94181527437925
```

In [53]:

```
amh_s.median()
```

Out[53]:

```
200.0
```

In [54]:

```
amh_s.max()
```

Out[54]:

```
310
```

In [55]:

```
amh_s.min()
```

Out[55]:

```
96
```

In [56]:

```
amh_s.skew()
```

Out[56]:

```
0.053224587799163044
```

In [57]:

```
amh_s.kurt()
```

Out[57]:

```
-1.1350158577565719
```

In [58]:

```
q_low = amh_s.quantile(q=0.25)
q_high = amh_s.quantile(q=0.75)
# 四分位间距
q_interval = q_high - q_low
# k为系数
k = 1.5
# 筛选异常值
amh_s = amh_s[amh_s<q_high+k*q_interval][amh_s>q_low-k*q_interval]
amh_s
```

Out[58]:

```
0        157
1        262
2        272
3        223
4        159
        ...
14997    280
14998    158
14999    158
15000    158
15001    158
Name: average_monthly_hours, Length: 15002, dtype: int64
```

In [59]:

```
# 连续值的直方图分布分析，分成10份
np.histogram(amh_s.values,bins=10)
```

Out[59]:

```
(array([ 367, 1240, 2736, 1722, 1628, 1712, 1906, 2240, 1127,  324],
       dtype=int64),
 array([ 96. , 117.4, 138.8, 160.2, 181.6, 203. , 224.4, 245.8, 267.2,
        288.6, 310. ]))
```

In [60]:

```
# 左闭右开
np.histogram(amh_s.values,bins=np.arange(amh_s.min(),amh_s.max()+10,10))
```

Out[60]:

```
(array([ 168,  171,  147,  807, 1153, 1234, 1075,  824,  818,  758,  751,
         738,  856,  824,  987, 1002, 1045,  935,  299,  193,  131,   86],
       dtype=int64),
 array([ 96, 106, 116, 126, 136, 146, 156, 166, 176, 186, 196, 206, 216,
        226, 236, 246, 256, 266, 276, 286, 296, 306, 316], dtype=int64))
```

In [61]:

```
# 计算每个值数量
# 左开右闭
amh_s.value_counts(bins=np.arange(amh_s.min(),amh_s.max()+10,10))
```

Out[61]:

```
(146.0, 156.0]     1277
(136.0, 146.0]     1159
(256.0, 266.0]     1063
(236.0, 246.0]     1006
(156.0, 166.0]      995
(246.0, 256.0]      987
(126.0, 136.0]      886
(216.0, 226.0]      873
(266.0, 276.0]      860
(166.0, 176.0]      832
(226.0, 236.0]      814
(176.0, 186.0]      813
(186.0, 196.0]      761
(196.0, 206.0]      755
(206.0, 216.0]      731
(276.0, 286.0]      319
(95.999, 106.0]     187
(286.0, 296.0]      164
(116.0, 126.0]      162
(106.0, 116.0]      162
(296.0, 306.0]      128
(306.0, 316.0]       68
Name: average_monthly_hours, dtype: int64
```

第五列：在公司的时间time_spend_company分析

In [62]:

```
tsc_s = df['time_spend_company']
tsc_s
```

Out[62]:

```
0        3
1        6
2        4
3        5
4        3
        ..
14997    4
14998    3
14999    3
15000    3
15001    2
Name: time_spend_company, Length: 15002, dtype: int64
```

In [63]:

```
tsc_s.value_counts().sort_index()
```

Out[63]:

```
2     3245
3     6445
4     2557
5     1473
6      718
7      188
8      162
10     214
Name: time_spend_company, dtype: int64
```

In [64]:

```
tsc_s.mean()
```

Out[64]:

```
3.498066924410079
```

第六列：工作事故Work_accident分析

In [65]:

```
wa_s = df['Work_accident']
wa_s
```

Out[65]:

```
0        0
1        0
2        0
3        0
4        0
        ..
14997    0
14998    0
14999    0
15000    0
15001    0
Name: Work_accident, Length: 15002, dtype: int64
```

In [66]:

```
wa_s.value_counts()
```

Out[66]:

```
0    12833
1     2169
Name: Work_accident, dtype: int64
```

In [67]:

```
# 事故率，在0-1的取值，事故率和均值是相等的
wa_s.mean()
```

Out[67]:

```
0.14458072257032395
```

第七列：最近离职left分析

In [68]:

```
l_s = df['left']
l_s
```

Out[68]:

```
0        1
1        1
2        1
3        1
4        1
        ..
14997    1
14998    1
14999    1
15000    1
15001    1
Name: left, Length: 15002, dtype: int64
```

In [69]:

```
l_s.value_counts()
```

Out[69]:

```
0    11428
1     3574
Name: left, dtype: int64
```

第八列：过去5年的提升promotion_last_5years分析

In [70]:

```
pl5_s = df['promotion_last_5years']
pl5_s
```

Out[70]:

```
0        0
1        0
2        0
3        0
4        0
        ..
14997    0
14998    0
14999    0
15000    0
15001    0
Name: promotion_last_5years, Length: 15002, dtype: int64
```

In [71]:

```
pl5_s.value_counts()
```

Out[71]:

```
0    14683
1      319
Name: promotion_last_5years, dtype: int64
```

第十列：工资salary分析

In [72]:

```
s_s = df['salary']
s_s
```

Out[72]:

```
0           low
1        medium
2        medium
3           low
4           low
          ...
14997       low
14998       low
14999       low
15000       low
15001       nme
Name: salary, Length: 15002, dtype: object
```

In [73]:

```
s_s.value_counts()
```

Out[73]:

```
low       7318
medium    6446
high      1237
nme          1
Name: salary, dtype: int64
```

In [75]:

```
s_s = s_s.where(s_s!='nme').dropna()
s_s
```

Out[75]:

```
0           low
1        medium
2        medium
3           low
4           low
          ...
14996       low
14997       low
14998       low
14999       low
15000       low
Name: salary, Length: 15001, dtype: object
```

In [76]:

```
s_s.value_counts()
```

Out[76]:

```
low       7318
medium    6446
high      1237
Name: salary, dtype: int64
```

第九列：部门department分析

In [77]:

```
d_s = df['department']
d_s
```

Out[77]:

```
0          sales
1          sales
2          sales
3          sales
4          sales
          ...
14997    support
14998    support
14999    support
15000       sale
15001       sale
Name: department, Length: 15002, dtype: object
```

In [78]:

```
d_s.value_counts(normalize=True)
```

Out[78]:

```
sales          0.275963
technical      0.181309
support        0.148647
IT             0.081789
product_mng    0.060125
marketing      0.057192
RandD          0.052460
accounting     0.051127
hr             0.049260
management     0.041994
sale           0.000133
Name: department, dtype: float64
```

In [80]:

```
d_s = d_s.where(d_s!='sale')
d_s
```

Out[80]:

```
0           sales
1           sales
2           sales
3           sales
4           sales
          ...
14997     support
14998     support
14999     support
15000         NaN
15001         NaN
Name: department, Length: 15002, dtype: object
```

In [81]:

```
d_s = d_s.dropna()
d_s
```

Out[81]:

```
0           sales
1           sales
2           sales
3           sales
4           sales
          ...
14995     support
14996     support
14997     support
14998     support
14999     support
Name: department, Length: 15000, dtype: object
```

简单对比分析

In [93]:

```
# 先剔除异常值
df = df.dropna(axis=0,how=’any’)
df
```

Out[93]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	time_spenc
0	0.38	0.53	2	157	
1	0.80	0.86	5	262	
2	0.11	0.88	7	272	
3	0.72	0.87	5	223	
4	0.37	0.52	2	159	
...	...	...	...	...	
14995	0.37	0.48	2	160	
14996	0.37	0.53	2	143	
14997	0.11	0.96	6	280	
14998	0.37	0.52	2	158	
15001	0.70	0.40	2	158	

15000 rows × 10 columns

In [95]:

```
df = df[df['salary']!='nme']
df
```

Out[95]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	time_spen
0	0.38	0.53	2	157	
1	0.80	0.86	5	262	
2	0.11	0.88	7	272	
3	0.72	0.87	5	223	
4	0.37	0.52	2	159	
...	...	...	...	...	
14994	0.40	0.57	2	151	
14995	0.37	0.48	2	160	
14996	0.37	0.53	2	143	
14997	0.11	0.96	6	280	
14998	0.37	0.52	2	158	

14999 rows × 10 columns

In [96]:

```
# 对部门进行对比，先分组，再聚合
df.groupby('department').mean()
```

Out[96]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	time_
department					
IT	0.618142	0.716830	3.816626	202.215974	
RandD	0.619822	0.712122	3.853875	200.800508	
accounting	0.582151	0.717718	3.825293	201.162973	
hr	0.598809	0.708850	3.654939	198.684709	
management	0.621349	0.724000	3.860317	201.249206	
marketing	0.618601	0.715886	3.687646	199.385781	
product_mng	0.619634	0.714756	3.807095	199.965632	
sales	0.614447	0.709717	3.776329	200.911353	
support	0.618300	0.723109	3.803948	200.758188	
technical	0.607897	0.721099	3.877941	202.497426	

In [97]:

```
# 对表中某几个字段进行对比
df.loc[:,['last_evaluation','department']].groupby('department').mean()
```

Out[97]:

	last_evaluation
department	
IT	0.716830
RandD	0.712122
accounting	0.717718
hr	0.708850
management	0.724000
marketing	0.715886
product_mng	0.714756
sales	0.709717
support	0.723109
technical	0.721099

In [98]:

```
# 使用自己定义的函数进行对比，这里计算极差：最大值-最小值
df.loc[:,['average_monthly_hours','department']].groupby('department')['average_monthly_hours'].
apply(lambda x:x.max()-x.min())
```

Out[98]:

```
department
IT             212
RandD          210
accounting     213
hr             212
management     210
marketing      214
product_mng    212
sales          214
support        214
technical      213
Name: average_monthly_hours, dtype: int64
```

In [ ]:

```

```