数据特征分析

1、分布分析
2、对比分析
3、周期分析
4、贡献度分析
5、相关性分析

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:

```
# plt.rc('font', **{'family' : 'HiraginoSansGB-W3, PingFangSC-Regul
ar, Microsoft YaHei, SimHe'})
```

分布分析

1、定量数据的分布分析

In [3]:

```
np.random.seed(1)
data = np.random.normal(3000, 1000, size=100)
data = data[data>1800]
data
```

Out[3]:

```
array([4624.34536366, 2388.24358635, 2471.82824774, 1927.03137784,
       3865.40762932, 4744.81176422, 2238.7930991 , 3319.03909606,
       2750.62962452, 4462.10793704, 2677.58279599, 2615.94564533,
       4133.76944234, 1900.10873269, 2827.57179245, 2122.14158208,
       3042.21374672, 3582.81521372, 1899.38082279, 4144.72370984,
       3901.59072059, 3502.4943389 , 3900.85594926, 2316.27214083,
       2877.10977448, 2064.23056574, 2732.11192037, 3530.35546674,
       2308.33924827, 2603.24647314, 2312.82729988, 2154.7943585 ,
       2328.75386916, 2987.33540108, 1882.68965136, 3234.41569782,
       4659.80217711, 3742.04416058, 2808.16444764, 2112.37103592,
       2252.84170625, 4692.45460103, 3050.80775478, 2363.00435343,
       3190.91548467, 5100.25513648, 3120.15895248, 3617.20310971,
       3300.17031996, 2647.75015351, 1857.48180198, 2650.65727759,
       2791.10576663, 3586.62319118, 3838.98341387, 3931.1020813 ,
       3285.58732525, 3885.14116427, 2245.602059  , 4252.86815523,
       3512.92982042, 2701.9071649 , 3488.51814654, 2924.42828698,
       4131.62938745, 4519.81681642, 5185.57540653, 2495.53413705,
       3160.03706945, 3876.16892112, 3315.63494724, 2693.79598737,
       3827.97464261, 3230.09473536, 3762.01118031, 2777.67185739,
       2799.24193107, 3186.56139099, 3410.05164721, 3198.29972013,
       3119.00864581, 2329.33771371, 3377.56378632, 3121.82127099,
       4129.48390791, 4198.9178799 , 3185.15641748, 2624.71504991,
       2361.26959255, 3423.49435406, 3077.34006835, 2656.14632443,
       3043.59685683, 2379.99915605, 3698.03203407])
```

In [4]:

```
len(data)
```

Out[4]:

```
95
```

In [5]:

```
range_val = np.max(data) - np.min(data)      # 极差
range_val
```

Out[5]:

```
3328.0936045553017
```

In [6]:

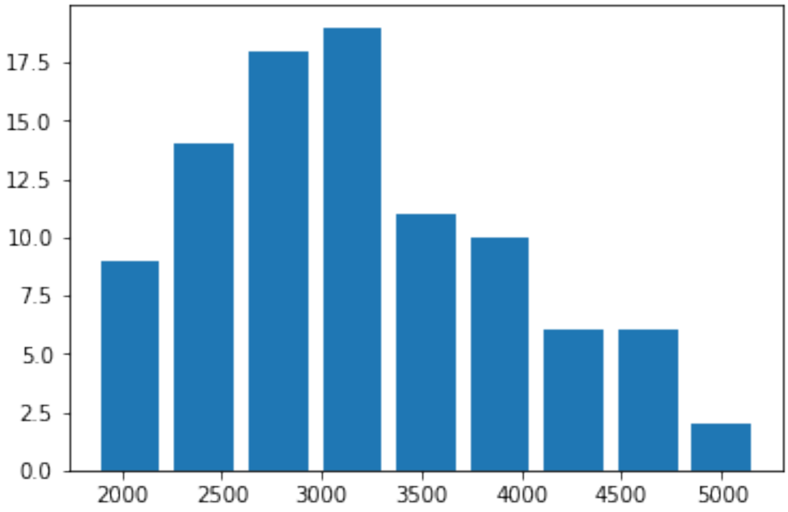
```
import math
bins = math.ceil(range_val/400)
bins
```

Out[6]:

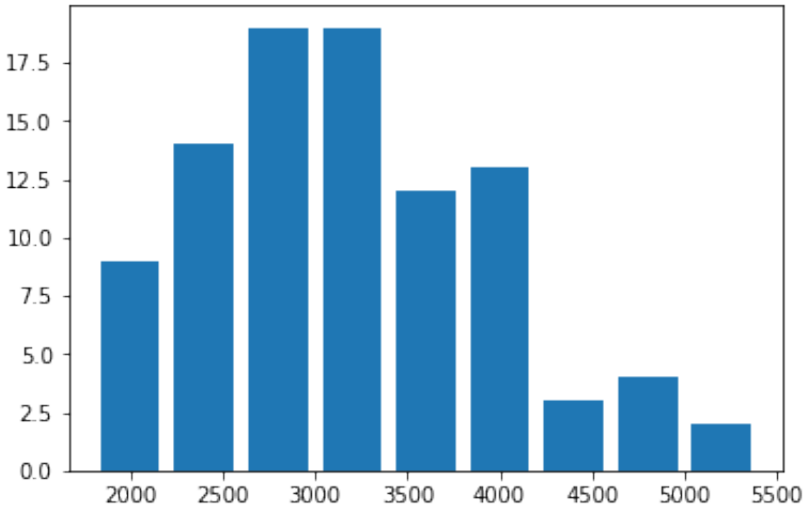
```
9
```

```
In [7]: plt.hist(data, bins=bins, rwidth=0.8)
```

```
Out[7]: (array([ 9., 14., 18., 19., 11., 10.,  6.,  6.,  2.]),
         array([1857.48180198, 2227.26998026, 2597.05815855, 2966.84633683
        ,
                3336.63451511, 3706.4226934 , 4076.21087168, 4445.99904997
        ,
                4815.78722825, 5185.57540653]),
         <a list of 9 Patch objects>)
```



```
In [8]: res = plt.hist(data, bins=bins, rwidth=0.8, range=(1800, 400*bins+1
        800))
```



```
In [9]: res[0]
```

```
Out[9]: array([ 9., 14., 19., 19., 12., 13.,  3.,  4.,  2.])
```

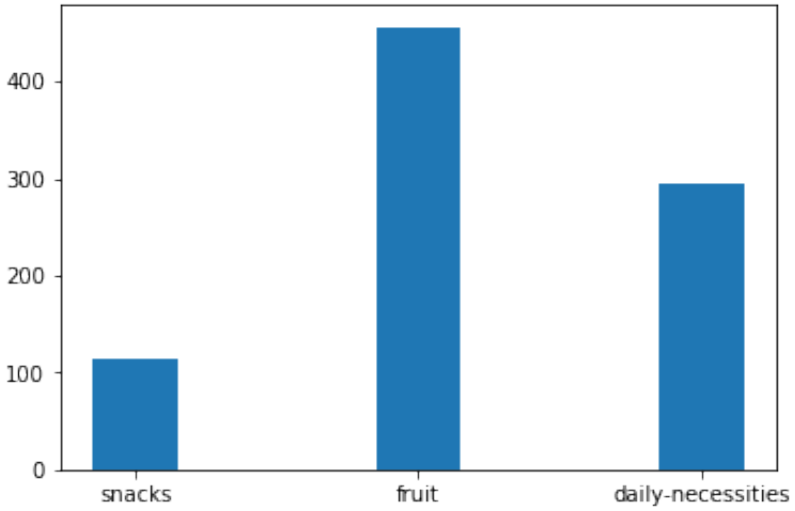
```
In [10]: res[1]
```

```
Out[10]: array([1800., 2200., 2600., 3000., 3400., 3800., 4200., 4600., 500
         0.,
                5400.])
```

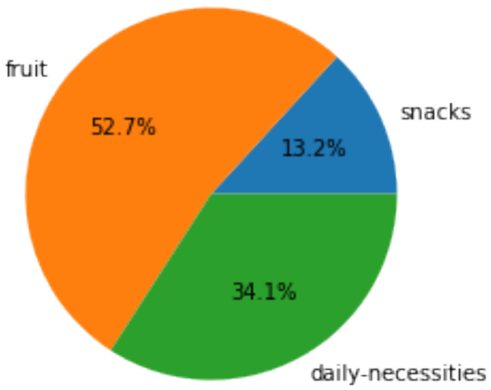
2、定性数据的分布分析

```
In [11]: data = [114, 456, 295]
         labels = ['snacks','fruit','daily-necessities']
```

```
In [12]: plt.bar(labels, data, width=0.3)
         plt.show()
```



```
In [13]: plt.axes(aspect=1)
         plt.pie(data, labels=labels, autopct='%.1f%%')
         plt.show()
```



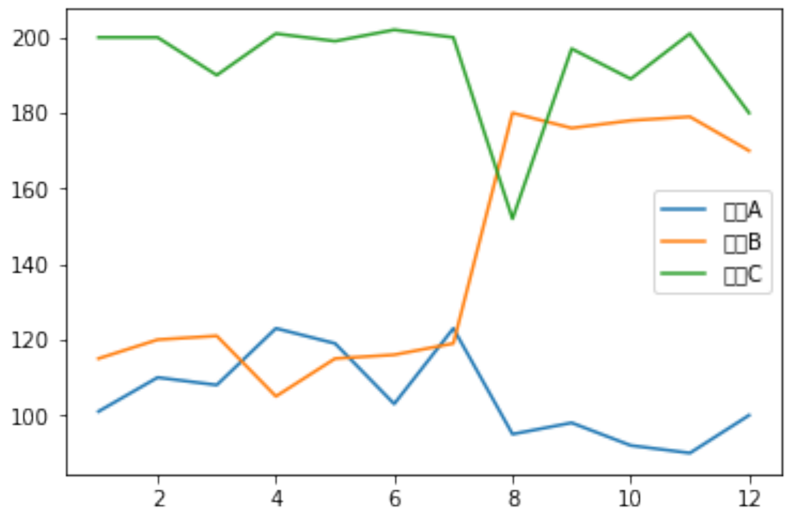
对比分析

In [14]:
```
data = {
    '部门A': [101, 110, 108, 123, 119, 103, 123, 95, 98, 92, 90, 100
],
    '部门B': [115, 120, 121, 105, 115, 116, 119, 180, 176, 178, 179,
170],
    '部门C': [200, 200, 190, 201, 199, 202, 200, 152, 197, 189, 201,
180]
}
df = pd.DataFrame(data, index=np.arange(1, 13))
df
```

Out[14]:

	部门A	部门B	部门C
1	101	115	200
2	110	120	200
3	108	121	190
4	123	105	201
5	119	115	199
6	103	116	202
7	123	119	200
8	95	180	152
9	98	176	197
10	92	178	189
11	90	179	201
12	100	170	180

In [15]:
```
res = df.plot()
plt.savefig('对比分析.png')
```



```
In [16]: res
```

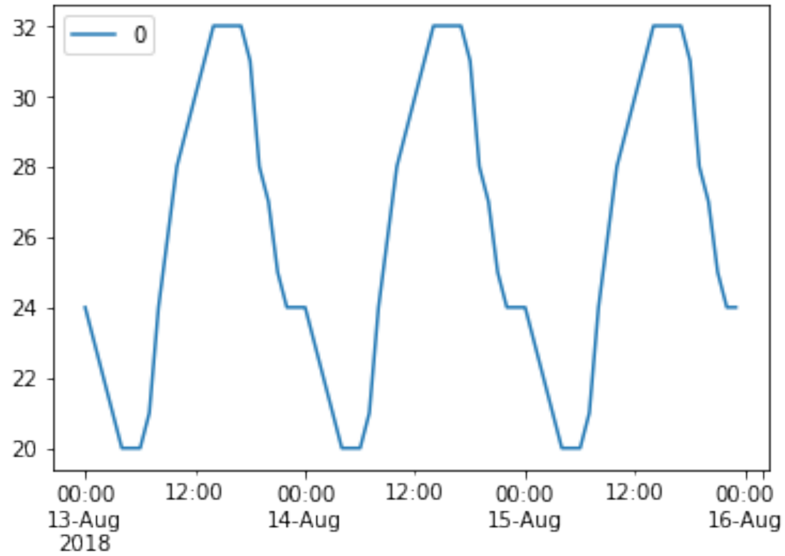
```
Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x119e40ba8>
```

周期性分析

```
In [17]: # 模拟三天的气温
         y = [24, 23, 22, 21, 20, 20, 20, 21, 24, 26, 28, 29, 30, 31, 32, 32
         , 32, 32, 31, 28, 27, 25, 24, 24]*3
         x = pd.date_range('2018-08-13', periods = 72, freq = 'H')
         df = pd.DataFrame(y, index=x)
         print(df.head())
         print(df.tail())
```

```
                      0
2018-08-13 00:00:00  24
2018-08-13 01:00:00  23
2018-08-13 02:00:00  22
2018-08-13 03:00:00  21
2018-08-13 04:00:00  20
                      0
2018-08-15 19:00:00  28
2018-08-15 20:00:00  27
2018-08-15 21:00:00  25
2018-08-15 22:00:00  24
2018-08-15 23:00:00  24
```

```
In [18]: df.plot()
         plt.show()
```



贡献度分析

In [19]:

```
# data = {
#     'profit': [1888, 1999, 2000, 334, 113, 1770, 124, 888, 503, 3
33]
# }
data = [1888, 1999, 2000, 334, 113, 1770, 124, 888, 503, 333]
index = ['服装', '手机', '家电', '玩具', '零食', '汽配', '图书', '办公',
'机票', '电脑']

df = pd.DataFrame(data, index=index, columns=['profit'])
df
```

Out[19]:

	profit
服装	1888
手机	1999
家电	2000
玩具	334
零食	113
汽配	1770
图书	124
办公	888
机票	503
电脑	333

In [20]:
```
data2 = {
    'profit': [1888, 1999, 2000, 334, 113, 1770, 124, 888, 503, 333
]
}
index2 = ['服装', '手机', '家电', '玩具', '零食', '汽配', '图书', '办公'
, '机票', '电脑']

df0 = pd.DataFrame(data2, index=index2)
df0
```

Out[20]:

	profit
服装	1888
手机	1999
家电	2000
玩具	334
零食	113
汽配	1770
图书	124
办公	888
机票	503
电脑	333

In [21]:
```
df2 = df.sort_values('profit', ascending=False)
```

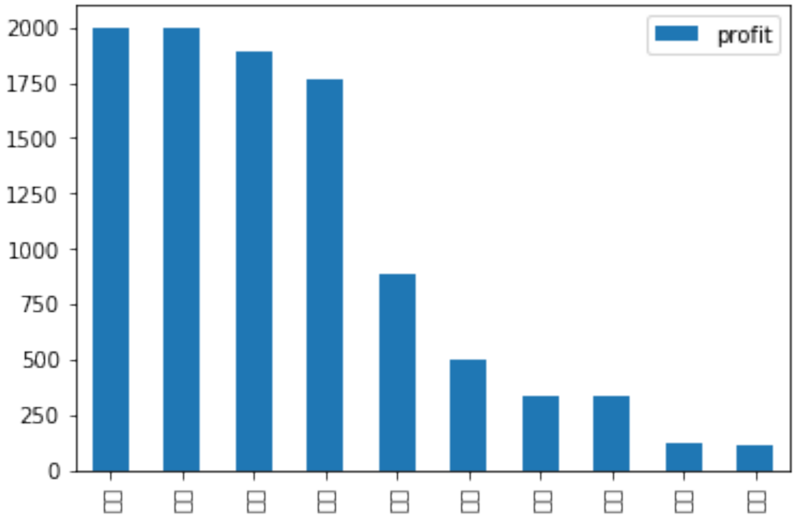
In [22]:
```
df2
```

Out[22]:

	profit
家电	2000
手机	1999
服装	1888
汽配	1770
办公	888
机票	503
玩具	334
电脑	333
图书	124
零食	113

In [23]:

```
df2.plot(kind='bar')
plt.show()
```



In [24]:

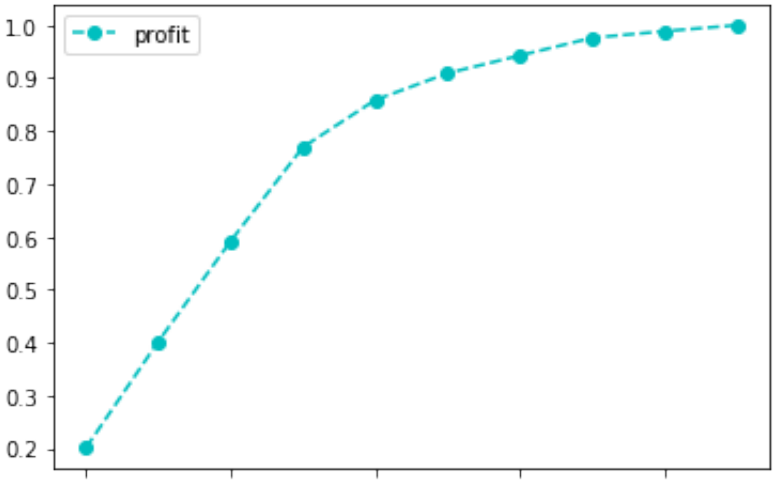
```
p = df2.cumsum()/df2.sum()
p
```

Out[24]:

	profit
家电	0.200965
手机	0.401829
服装	0.591539
汽配	0.769393
办公	0.858621
机票	0.909164
玩具	0.942725
电脑	0.976186
图书	0.988645
零食	1.000000

In [25]:

```
p.plot(style='c--o')
plt.show()
```



相关性分析

In [26]:

```
data = {
    'delivery-time': [12, 15, 15, 18, 18, 20, 20, 25, 25, 10, 10, 1
2],
    'minimum-delivery-amount': [15, 18, 18, 20, 20, 30, 30, 50, 50,
10, 10, 15],
    'sales-volume': [100, 200, 400, 400, 500, 600, 600, 700, 800, 9
00, 1000, 1000],
    'consumption-per-person': [100, 50, 80, 120, 60, 30, 200, 90, 4
0, 60, 58, 20],
    'grade': [1, 1, 1, 2, 2, 3, 3, 3, 4, 4, 5, 5]
}
df = pd.DataFrame(data)
df
```

Out[26]:

	delivery-time	minimum-delivery-amount	sales-volume	consumption-per-person	grade
0	12	15	100	100	1
1	15	18	200	50	1
2	15	18	400	80	1
3	18	20	400	120	2
4	18	20	500	60	2
5	20	30	600	30	3
6	20	30	600	200	3
7	25	50	700	90	3
8	25	50	800	40	4
9	10	10	900	60	4
10	10	10	1000	58	5
11	12	15	1000	20	5

corr() 查看相关性系数

In [27]:
```
df.corr()
```

Out[27]:

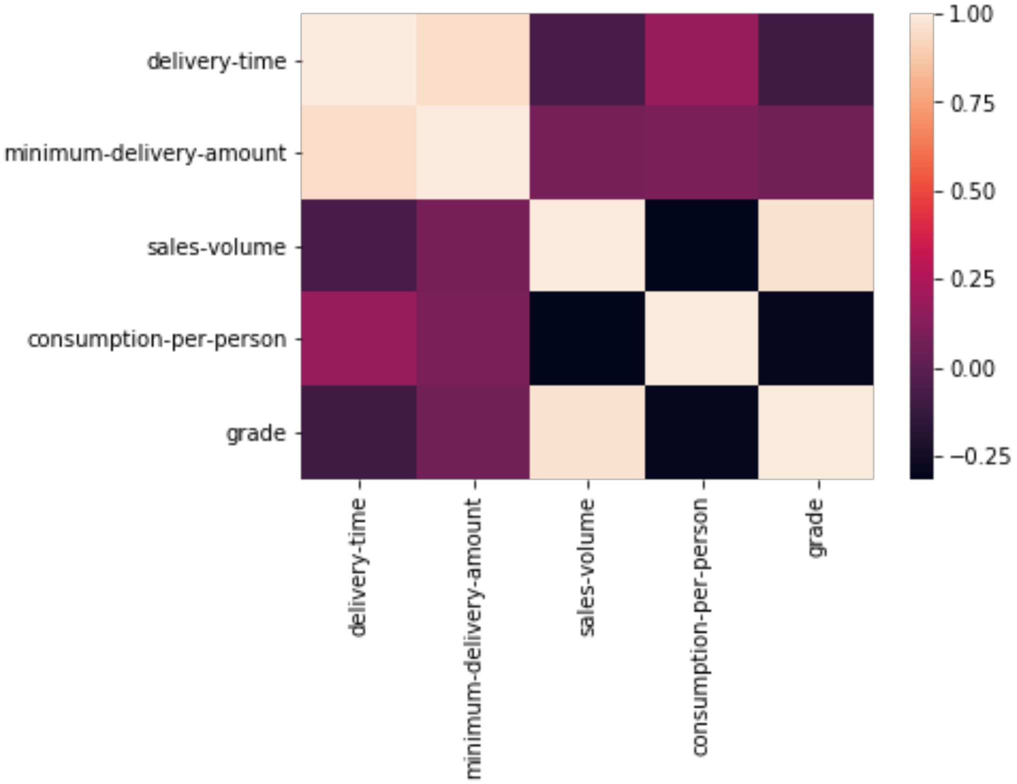
	delivery-time	minimum-delivery-amount	sales-volume	consumption-per-person	grade
delivery-time	1.000000	0.949072	-0.064532	0.187174	-0.090585
minimum-delivery-amount	0.949072	1.000000	0.082768	0.092060	0.070583
sales-volume	-0.064532	0.082768	1.000000	-0.314258	0.965059
consumption-per-person	0.187174	0.092060	-0.314258	1.000000	-0.295704
grade	-0.090585	0.070583	0.965059	-0.295704	1.000000

In [28]:
```
# df.corr?
```

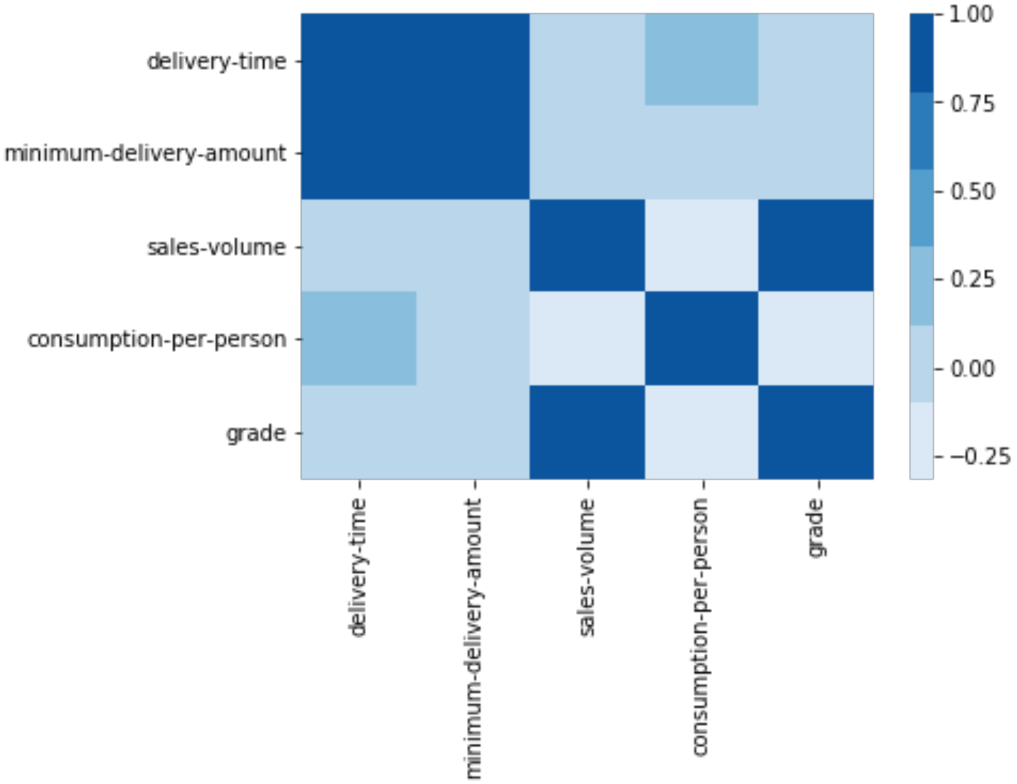
In [29]:
```
import seaborn as sns
```

In [30]:
```
corr = df.corr()
```

```
In [31]: sns.heatmap(corr)
         plt.show()
```



```
In [32]: sns.heatmap(corr, cmap=sns.color_palette('Blues'))
         plt.show()
```



```
In [33]: mask = np.zeros_like(corr, dtype=np.bool)
         mask
```

```
Out[33]: array([[False, False, False, False, False],
                [False, False, False, False, False],
                [False, False, False, False, False],
                [False, False, False, False, False],
                [False, False, False, False, False]])
```

```
In [34]: ind = np.triu_indices_from(mask)
         ind
```

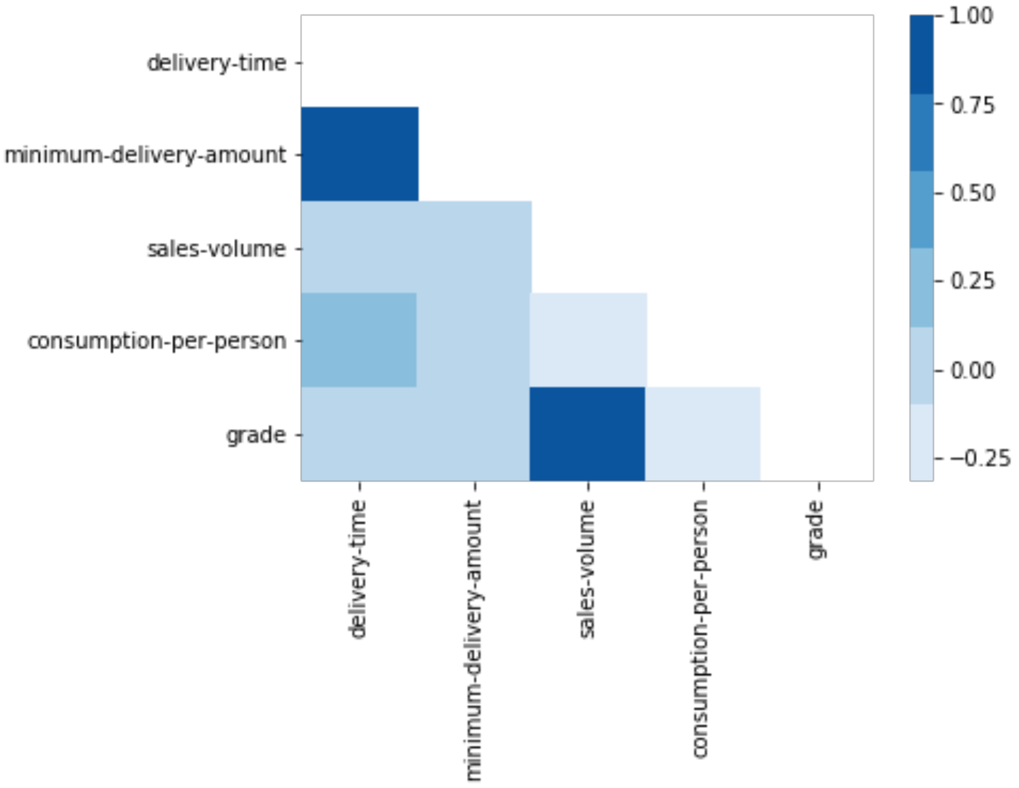
```
Out[34]: (array([0, 0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 3, 3, 4]),
          array([0, 1, 2, 3, 4, 1, 2, 3, 4, 2, 3, 4, 3, 4, 4]))
```

```
In [35]: mask[ind] = True
```

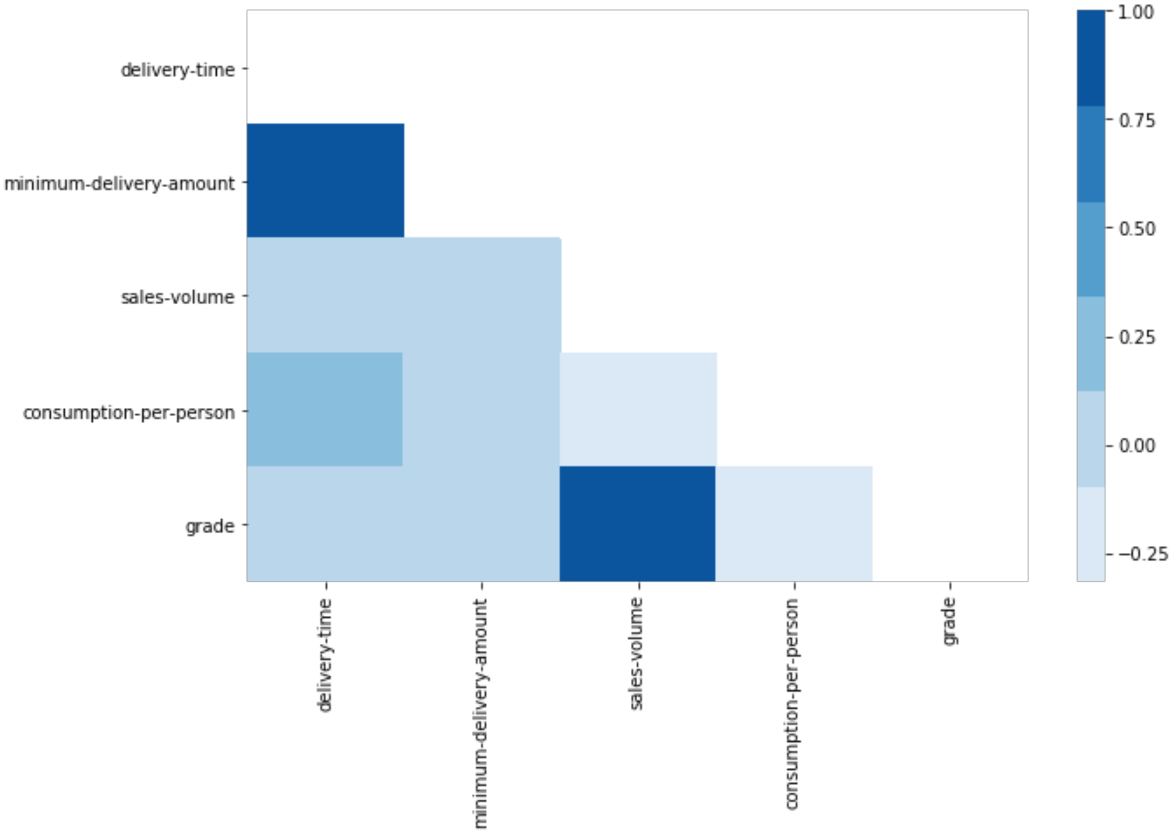
```
In [36]: mask
```

```
Out[36]: array([[ True,  True,  True,  True,  True],
                [False,  True,  True,  True,  True],
                [False, False,  True,  True,  True],
                [False, False, False,  True,  True],
                [False, False, False, False,  True]])
```

```
In [37]: sns.heatmap(corr, cmap=sns.color_palette('Blues'), mask=mask)
         plt.show()
```



```
In [38]: plt.figure(figsize=(10, 6))        # 设置图片大小
         sns.heatmap(corr, cmap=sns.color_palette('Blues'), mask=mask)
         plt.show()
```


delivery-time
minimum-delivery-amount
sales-volume
consumption-per-person
grade
delivery-time
minimum-delivery-amount
sales-volume
consumption-per-person
grade