

销售收入汇总实训（任务五）

读取开店计划表（I01）与新开店月营收模型（S01，两级列索引），匹配出每家新店的销售收入和成本预算（S02），提取数值列计算合计（S03），最后可视化特征数据与合计费用之间的关系。

1. 新开店销售收入和成本预算（读取数据）

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

#中文图表显示设置（避免坐标轴中文乱码）
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

#设置数据源文件路径
file = 'budget_data.xlsx'
df_I01 = pd.read_excel(file, sheet_name='I01') #开店计划表
#注意，打开营收模型表时，columns上有两级索引，用header参数设置
df_S01 = pd.read_excel(file, sheet_name='S01', header=[0,1], index_col=0) #新开店月;
#将营收模型表的行列转置（使用 df.T 属性可以实现数组的转置），reset_index()对索引重置
df_S01 = df_S01.T.reset_index()
df_S01.head()
```

Out[1]:

	店 模 型	选 址 评 级	销售 收 入	销售 成 本	原材 料	直接 人 工	租 赁 费	折 旧 费	广 告 促 销 费	制 服 费	维 修 费	水 电 费	其 他 杂 费
0	大型店	高	250000	100000	30000	25000	20000	8000	3000	1000	2000	5000	6000
1	大型店	中	150000	63000	18900	15750	12600	5040	1890	630	1260	3150	3780
2	大型店	低	80000	40000	12000	10000	8000	3200	1200	400	800	2000	2400
3	中型店	高	180000	75600	22680	18900	15120	6048	2268	756	1512	3780	4536
4	中型店	中	120000	52800	15840	13200	10560	4224	1584	528	1056	2640	3168

2. 将新开店计划与新店月度营收模型进行匹配

```
In [2]: df_S02 = pd.merge(df_I01,df_S01,on=['店规模','选址评级'], how='left')
df_S02
```

Out[2]:

	店名	地址	开店日期	建设期	面积	店规模	选址评级	销售收入	销售成本	原材料	直接人工	租赁费	折旧费	广促费
0	11号店	江苏省南京市	2019年3月	2019年2月	1000	大型店	中	150000	63000	18900	15750	12600	5040	18900
1	12号店	安徽省合肥市	2019年5月	2019年3月	500	中型店	中	120000	52800	15840	13200	10560	4224	15840
2	13号店	重庆市	2019年6月	2019年4月	600	中型店	高	180000	75600	22680	18900	15120	6048	22680
3	14号店	山西省太原市	2019年8月	2019年7月	300	小型店	低	50000	22000	6600	5500	4400	1760	6600
4	15号店	云南省昆明市	2019年10月	2019年9月	200	小型店	中	100000	46000	13800	11500	9200	3680	13800
5	16号店	山东省济南市	2019年11月	2019年9月	950	大型店	高	250000	100000	30000	25000	20000	8000	30000

3. 计算每个店的销售收入和

```
In [3]: df_S03 = df_S02.iloc[:,[0,7,8,9,10,11,12,13,14,15,16,17]]
df_S03
```

Out[3]:

	店名	销售收入	销售成本	原材料	直接人工	租赁费	折旧费	广告促销费	制服费	维修费	水电费	其他杂费
0	11号店	150000	63000	18900	15750	12600	5040	1890	630	1260	3150	3780
1	12号店	120000	52800	15840	13200	10560	4224	1584	528	1056	2640	3168
2	13号店	180000	75600	22680	18900	15120	6048	2268	756	1512	3780	4536
3	14号店	50000	22000	6600	5500	4400	1760	660	220	440	1100	1320
4	15号店	100000	46000	13800	11500	9200	3680	1380	460	920	2300	2760
5	16号店	250000	100000	30000	25000	20000	8000	3000	1000	2000	5000	6000

```
In [4]: df_S03['合计'] =df_S03.iloc[:,1:].sum(axis=1)
df_S03
```

C:\Users\10730\AppData\Local\Temp\ipykernel_56816\621790412.py:1: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.

Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
df_S03['合计'] =df_S03.iloc[:,1:].sum(axis=1)
```

Out[4]:

	店名	销售收入	销售成本	原材料	直接人工	租赁费	折旧费	广告促销费	制服费	维修费	水电费	其他杂费	
0	11号店	150000	63000	18900	15750	12600	5040	1890	630	1260	3150	3780	27
1	12号店	120000	52800	15840	13200	10560	4224	1584	528	1056	2640	3168	22
2	13号店	180000	75600	22680	18900	15120	6048	2268	756	1512	3780	4536	33
3	14号店	50000	22000	6600	5500	4400	1760	660	220	440	1100	1320	9
4	15号店	100000	46000	13800	11500	9200	3680	1380	460	920	2300	2760	19
5	16号店	250000	100000	30000	25000	20000	8000	3000	1000	2000	5000	6000	41

In [5]: df_S03 = pd.concat([df_S03,df_S03,df_S03])
df_S03

Out[5]:

	店名	销售收入	销售成本	原材料	直接人工	租赁费	折旧费	广告促销费	制服费	维修费	水电费	其他杂费	
0	11号店	150000	63000	18900	15750	12600	5040	1890	630	1260	3150	3780	27000
1	12号店	120000	52800	15840	13200	10560	4224	1584	528	1056	2640	3168	22000
2	13号店	180000	75600	22680	18900	15120	6048	2268	756	1512	3780	4536	33000
3	14号店	50000	22000	6600	5500	4400	1760	660	220	440	1100	1320	9000
4	15号店	100000	46000	13800	11500	9200	3680	1380	460	920	2300	2760	19000
5	16号店	250000	100000	30000	25000	20000	8000	3000	1000	2000	5000	6000	49000
0	11号店	150000	63000	18900	15750	12600	5040	1890	630	1260	3150	3780	27000
1	12号店	120000	52800	15840	13200	10560	4224	1584	528	1056	2640	3168	22000
2	13号店	180000	75600	22680	18900	15120	6048	2268	756	1512	3780	4536	33000
3	14号店	50000	22000	6600	5500	4400	1760	660	220	440	1100	1320	9000
4	15号店	100000	46000	13800	11500	9200	3680	1380	460	920	2300	2760	19000
5	16号店	250000	100000	30000	25000	20000	8000	3000	1000	2000	5000	6000	49000
0	11号店	150000	63000	18900	15750	12600	5040	1890	630	1260	3150	3780	27000
1	12号店	120000	52800	15840	13200	10560	4224	1584	528	1056	2640	3168	22000
2	13号	180000	75600	22680	18900	15120	6048	2268	756	1512	3780	4536	33000

	店名	销售收入	销售成本	原材料	直接人工	租赁费	折旧费	广告促销费	制服费	维修费	水电费	其他杂费	
	店												
3	14号店	50000	22000	6600	5500	4400	1760	660	220	440	1100	1320	9
4	15号店	100000	46000	13800	11500	9200	3680	1380	460	920	2300	2760	19
5	16号店	250000	100000	30000	25000	20000	8000	3000	1000	2000	5000	6000	49

4. 可视化特征数据与合计费用之间的关系

```
In [6]: #获取标签数据
labeldata = list(df_S03['合计'])
#获取特征数据
traindata = []
for i in range(18):
    value = df_S03.iloc[i,1:12]
    traindata.append(list(value))
#转换List类型为ndarray类型
print(traindata)
traindata = np.array(traindata)
labeldata = np.array(labeldata)
print(traindata)
print(labeldata)
```

```

[[np.int64(150000), np.int64(63000), np.int64(18900), np.int64(15750), np.int64(12600), np.int64(5040), np.int64(1890), np.int64(630), np.int64(1260), np.int64(3150), np.int64(3780)], [np.int64(120000), np.int64(52800), np.int64(15840), np.int64(13200), np.int64(10560), np.int64(4224), np.int64(1584), np.int64(528), np.int64(1056), np.int64(2640), np.int64(3168)], [np.int64(180000), np.int64(75600), np.int64(22680), np.int64(18900), np.int64(15120), np.int64(6048), np.int64(2268), np.int64(756), np.int64(1512), np.int64(3780), np.int64(4536)], [np.int64(50000), np.int64(22000), np.int64(6600), np.int64(5500), np.int64(4400), np.int64(1760), np.int64(660), np.int64(220), np.int64(440), np.int64(1100), np.int64(1320)], [np.int64(100000), np.int64(46000), np.int64(13800), np.int64(11500), np.int64(9200), np.int64(3680), np.int64(1380), np.int64(460), np.int64(920), np.int64(2300), np.int64(2760)], [np.int64(250000), np.int64(100000), np.int64(30000), np.int64(25000), np.int64(20000), np.int64(8000), np.int64(3000), np.int64(1000), np.int64(2000), np.int64(5000), np.int64(6000)], [np.int64(150000), np.int64(63000), np.int64(18900), np.int64(15750), np.int64(12600), np.int64(5040), np.int64(1890), np.int64(630), np.int64(1260), np.int64(3150), np.int64(3780)], [np.int64(120000), np.int64(52800), np.int64(15840), np.int64(13200), np.int64(10560), np.int64(4224), np.int64(1584), np.int64(528), np.int64(1056), np.int64(2640), np.int64(3168)], [np.int64(180000), np.int64(75600), np.int64(22680), np.int64(18900), np.int64(15120), np.int64(6048), np.int64(2268), np.int64(756), np.int64(1512), np.int64(3780), np.int64(4536)], [np.int64(50000), np.int64(22000), np.int64(6600), np.int64(5500), np.int64(4400), np.int64(1760), np.int64(660), np.int64(220), np.int64(440), np.int64(1100), np.int64(1320)], [np.int64(100000), np.int64(46000), np.int64(13800), np.int64(11500), np.int64(9200), np.int64(3680), np.int64(1380), np.int64(460), np.int64(920), np.int64(2300), np.int64(2760)], [np.int64(250000), np.int64(100000), np.int64(30000), np.int64(25000), np.int64(20000), np.int64(8000), np.int64(3000), np.int64(1000), np.int64(2000), np.int64(5000), np.int64(6000)], [np.int64(150000), np.int64(63000), np.int64(18900), np.int64(15750), np.int64(12600), np.int64(5040), np.int64(1890), np.int64(630), np.int64(1260), np.int64(3150), np.int64(3780)], [np.int64(120000), np.int64(52800), np.int64(15840), np.int64(13200), np.int64(10560), np.int64(4224), np.int64(1584), np.int64(528), np.int64(1056), np.int64(2640), np.int64(3168)], [np.int64(180000), np.int64(75600), np.int64(22680), np.int64(18900), np.int64(15120), np.int64(6048), np.int64(2268), np.int64(756), np.int64(1512), np.int64(3780), np.int64(4536)], [np.int64(50000), np.int64(22000), np.int64(6600), np.int64(5500), np.int64(4400), np.int64(1760), np.int64(660), np.int64(220), np.int64(440), np.int64(1100), np.int64(1320)], [np.int64(100000), np.int64(46000), np.int64(13800), np.int64(11500), np.int64(9200), np.int64(3680), np.int64(1380), np.int64(460), np.int64(920), np.int64(2300), np.int64(2760)], [np.int64(250000), np.int64(100000), np.int64(30000), np.int64(25000), np.int64(20000), np.int64(8000), np.int64(3000), np.int64(1000), np.int64(2000), np.int64(5000), np.int64(6000)]]

```

```

[[150000 63000 18900 15750 12600 5040 1890 630 1260 3150
 3780]
 [120000 52800 15840 13200 10560 4224 1584 528 1056 2640
 3168]
 [180000 75600 22680 18900 15120 6048 2268 756 1512 3780
 4536]
 [ 50000 22000 6600 5500 4400 1760 660 220 440 1100
 1320]
 [100000 46000 13800 11500 9200 3680 1380 460 920 2300
 2760]
 [250000 100000 30000 25000 20000 8000 3000 1000 2000 5000
 6000]
 [150000 63000 18900 15750 12600 5040 1890 630 1260 3150
 3780]
 [120000 52800 15840 13200 10560 4224 1584 528 1056 2640
 3168]

```

```

[180000  75600  22680  18900  15120  6048  2268  756  1512  3780
 4536]
[ 50000  22000   6600   5500   4400  1760   660  220   440  1100
 1320]
[100000  46000  13800  11500   9200  3680  1380  460   920  2300
 2760]
[250000 100000  30000  25000  20000  8000  3000  1000  2000  5000
 6000]
[150000  63000  18900  15750  12600  5040  1890   630  1260  3150
 3780]
[120000  52800  15840  13200  10560  4224  1584   528  1056  2640
 3168]
[180000  75600  22680  18900  15120  6048  2268  756  1512  3780
 4536]
[ 50000  22000   6600   5500   4400  1760   660  220   440  1100
 1320]
[100000  46000  13800  11500   9200  3680  1380  460   920  2300
 2760]
[250000 100000  30000  25000  20000  8000  3000  1000  2000  5000
 6000]]
[276000 225600 331200  94000 192000 450000 276000 225600 331200  94000
192000 450000 276000 225600 331200  94000 192000 450000]

```

```

In [7]: def dataShow(traindata, labeldata):
        fig = plt.figure(figsize=(20, 5))
        colors = ['red', 'green', 'blue']
        titles = ['销售收入', '销售成本', '原材料']
        count = 1
        for i in range(3):
            ax1 = fig.add_subplot(1, 3, count)
            ax1.scatter(traindata[0:15,i], labeldata[:15], color=colors[i], marker='o')
            ax1.set_xlabel(titles[i])
            count = count+1
        plt.show()

        dataShow(traindata, labeldata)

```

