基于 RFM 分析的客户分群

一、实验目的：

1 掌握 RFM 分析方法和 k-means 聚类的方法，能够进行价值识别
2 掌握 Python 聚类的方法

二、知识准备：

RFM 模型是衡量客户价值和客户创利能力的重要工具和手段。在客户分类中，RFM 模型是一个经典的分类模型，利用通用交易环节中最核心的三个维度——最近消费(Recency)、消费频率(Frequency)、消费金额(Monetary)细分客户群体，从而分析不同群体的客户价值。

三、实验步骤：

1、提出问题，确定目标

对客户数据，探讨如何利用 KMeans 算法对客户群体进行细分，以及细分后如何利用 RFM 模型对客户价值进行分析，并识别出高价值客户。主要希望实现以下三个目标：
1）对客户进行群体分类
2）对不同的客户群体进行特征分析，比较各细分群体的客户价值
3）对不同价值的客户制定相应的运营策略

2、数据获取

```
data = pd.read_csv('实验2 数据data.csv',sep=',')
data
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

541909 rows × 8 columns

3、数据预处理

数据清洗：缺失值，异常值

变量转换、属性规约、标准化处理等

1）缺失值

```
#检查缺失值
data.isnull().any()
```

```
InvoiceNo      False
StockCode      False
Description     True
Quantity       False
InvoiceDate    False
UnitPrice      False
CustomerID      True
Country        False
dtype: bool
```

```
#对于关键字段缺失的数据进行清除数据记录处理
data = data.dropna(subset=['CustomerID'])
```

```
#处理之后
data.isnull().any()
```

```
InvoiceNo      False
StockCode      False
Description    False
Quantity       False
InvoiceDate    False
UnitPrice      False
CustomerID     False
Country        False
dtype: bool
```

```
data
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

406829 rows × 8 columns

2）新增变量

a、新增 money 存放数据记录中 Quantity 和 UnitPrice 的积

```
#计算每条数据的金额，存入新增的money列
data['money'] = data.loc[:,'Quantity']*data.loc[:,'UnitPrice']
data
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	money
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom	15.30
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	20.34
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom	22.00
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	20.34
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	20.34
...	...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France	10.20
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France	12.60
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France	16.60
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France	16.60
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France	14.85

b、新增 days 存放 Invoicedate 与日期 2010-01-01 的差距天数：（由于时间我转换为 Invoicedate 距离 2010 年 1 月 1 日的天数；R 越大证明近期消费也就越近！）

```
#计算 InvoiceDate 距2010-01-01的天数并存储到新增的 days 列中
days = []
days = (pd.to_datetime(data['InvoiceDate'])-pd.to_datetime('2010-01-01 00:00:00')).map(lambda x:x.days)
data['days'] = days
```

```
data
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	money	days
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom	15.30	334
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	20.34	334
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom	22.00	334
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	20.34	334
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	20.34	334
...	...	...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France	10.20	707
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France	12.60	707
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France	16.60	707
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France	16.60	707
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France	14.85	707

406829 rows × 10 columns

3）新建 DataFrame 结构的 data_RFM 用于存放整理后的数据集合。

```
#新建DataFrame结构的data_RFM
data_RFM = pd.DataFrame(columns=['CustomerID','R','F','M'])
data_RFM
```

CustomerID	R	F	M

4）根据原数据 data 进行分组统计构造 data_RFM 的数据集合

变量 R：（由于时间我转换为 Invoicedate 距离 2010 年 1 月 1 日的天数；R 越大证明近期消费也就越近！）

```
#按照'CustomerID'分组统计 'days' 的平均数目，作为RFM数据集的R（最近一次消费 (Recency)）
R = pd.DataFrame(data.groupby(['CustomerID'])['days'].max())
R
```

CustomerID	days
12346.0	382
12347.0	705
12348.0	632
12349.0	689
12350.0	397
...	...
18280.0	430
18281.0	527
18282.0	700
18283.0	704
18287.0	665

4372 rows × 1 columns

变量 F：

```
#按照'CustomerID'分组统计 'InvoiceNo' 的数目，作为RFM数据集的F（消费频率）
F = data.groupby(['CustomerID'])['InvoiceNo'].count()
F
```

```
CustomerID
12346.0      2
12347.0    182
12348.0     31
12349.0     73
12350.0     17
          ...
18280.0     10
18281.0      7
18282.0     13
18283.0    756
18287.0     70
Name: InvoiceNo, Length: 4372, dtype: int64
```

变量 M:

```
#按照'CustomerID','InvoiceNo','days'分组计算'money'的和，作为RFM数据集的M（消费金额）
M = data.groupby(['CustomerID'])['money'].sum()
```

```
M
```

```
CustomerID
12346.0        0.00
12347.0     4310.00
12348.0     1797.24
12349.0     1757.55
12350.0      334.40
              ...
18280.0      180.60
18281.0       80.82
18282.0      176.60
18283.0     2094.88
18287.0     1837.28
Name: money, Length: 4372, dtype: float64
```

5）将数据整合到 data_RFM 中

```
#将数据整合到data_RFM中
data_RFM['CustomerID'] = M.index.astype(object)
data_RFM['F'] = F.values
data_RFM['M'] = M.values
data_RFM['R'] = R.values
```

```
data_RFM
```

	CustomerID	R	F	M
0	12346.0	382	2	0.00
1	12347.0	705	182	4310.00
2	12348.0	632	31	1797.24
3	12349.0	689	73	1757.55
4	12350.0	397	17	334.40
...	...	...	...	...
4367	18280.0	430	10	180.60
4368	18281.0	527	7	80.82
4369	18282.0	700	13	176.60
4370	18283.0	704	756	2094.88
4371	18287.0	665	70	1837.28

4372 rows × 4 columns

6）data_RFM 数据处理

data_RFM 数据集基本情况：

```
data_RFM.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4372 entries, 0 to 4371
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   CustomerID  4372 non-null   object 
 1   R           4372 non-null   int64  
 2   F           4372 non-null   int64  
 3   M           4372 non-null   float64
dtypes: float64(1), int64(2), object(1)
memory usage: 136.8+ KB
```

进一步数据清理：

```
#清除金额小于等于0的无效数据，此类数据只有退货数据（为负值），不能作为RFM模型分析数据
data_RFM.drop(data_RFM[data_RFM['M']<=0].index,inplace=True)
```

4、数据探索性分析（可视化显示）

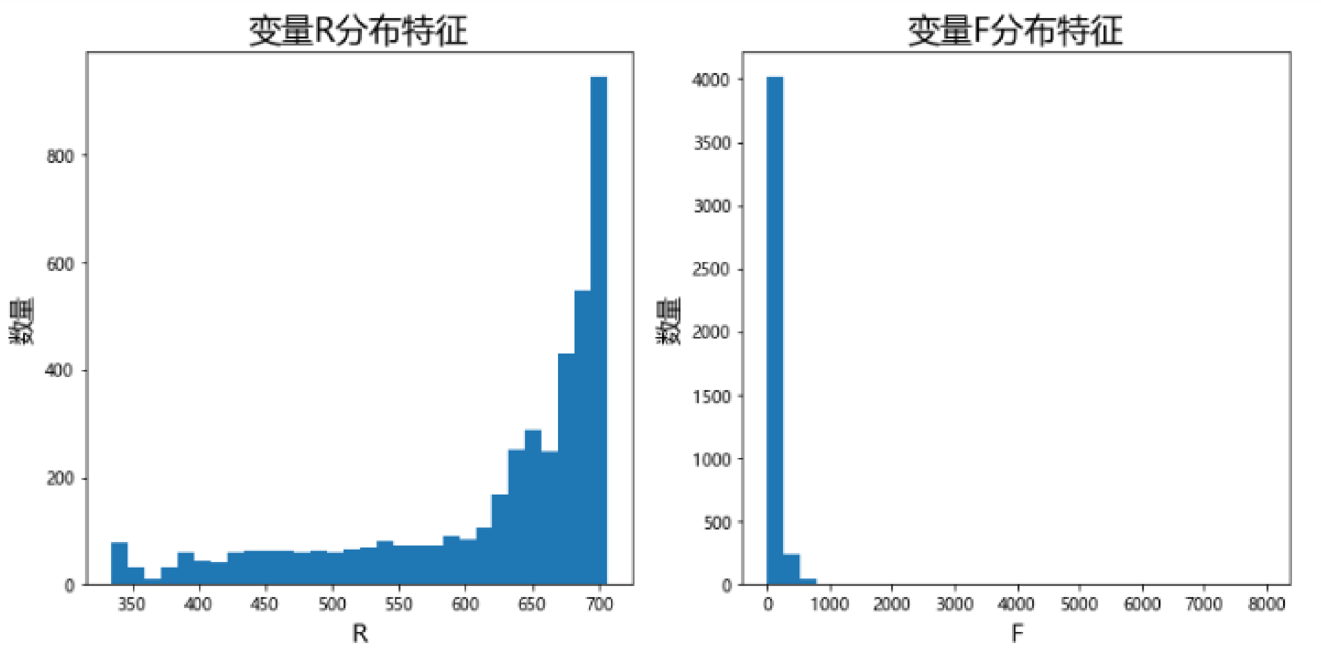
1）未标准化的变量分布情况，并存入到 k1 中（后续反标准化有用到）

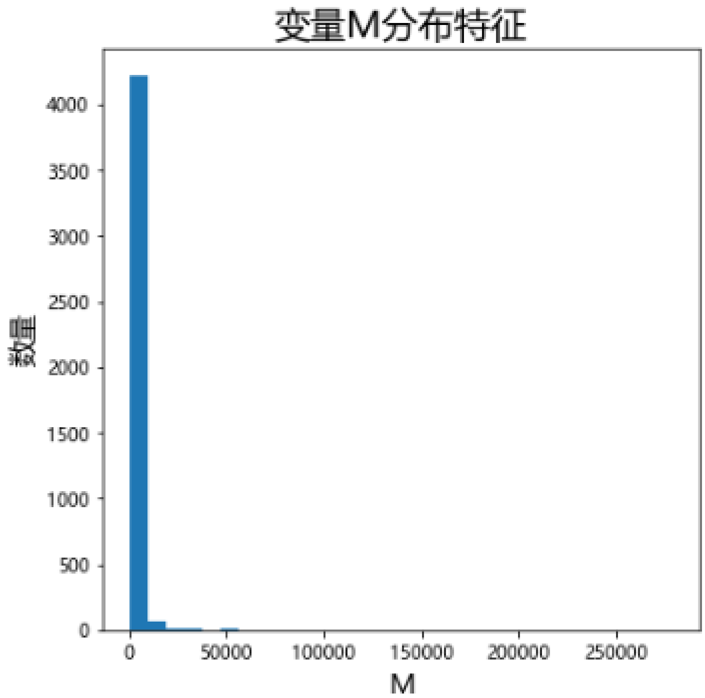
```
#未进行标准化的描述
data_RFM.describe()
```

	CustomerID	R	F	M
count	4322.000000	4322.000000	4322.000000	4.322000e+03
mean	15298.534475	617.123785	94.059695	1.923483e+03
std	1721.534033	99.137727	233.621415	8.263128e+03
min	12347.000000	334.000000	1.000000	1.776357e-15
25%	13812.250000	569.000000	18.000000	3.022925e+02
50%	15297.500000	658.000000	42.000000	6.575500e+02
75%	16777.750000	691.000000	102.750000	1.625740e+03
max	18287.000000	707.000000	7983.000000	2.794890e+05

```
k1 = data_RFM.describe()
```

```
#未进行标准化的三个变量的分布特征
import warnings
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
warnings.filterwarnings(action = 'ignore')
plt.figure(figsize=(13,13))
#R的分布
plt.subplot(2,2,1)
x_R = data_RFM['R']
plt.hist(x_R,bins=30)
#plt.yticks(())
plt.xlabel('R',fontsize=15)
plt.ylabel('数量',fontsize=15)
plt.title("变量R分布特征",fontsize=20)

#F的分布
plt.subplot(2,2,2)
x_F = data_RFM['F']
plt.hist(x_F,bins=30)
#plt.yticks(())
plt.xlabel('F',fontsize=15)
plt.ylabel('数量',fontsize=15)
plt.title("变量F分布特征",fontsize=20)

#M的分布
plt.subplot(2,2,3)
x_M = data_RFM['M']
plt.hist(x_M,bins=30)
#plt.yticks(())
plt.xlabel('M',fontsize=15)
plt.ylabel('数量',fontsize=15)
plt.title("变量M分布特征",fontsize=20)
```





在上图中，对于 R、F 两个变量的分布显示的较为明显，但是由于 M 变量的变量分布特性，在频率分布直方图中并不能很好的展示。从下图变量 M 的最大、最小值也可看出上述分布图的不足之处。

```
data_RFM['M'].max()
```

```
279489.02
```

```
data_RFM['M'].min()
```

```
1.77635683940025e-15
```

2）标准化（Z-score）处理

```
#标准化函数Z-score
def Zscore(x):
    x = (x-x.mean()) / np.std(x)
    return x
data_RFM.iloc[:,1:4] = Zscore(data_RFM.iloc[:,1:4])
```

标准化之后的数据集描述：

```
data_RFM.describe()
```

	CustomerID	R	F	M
count	4322.000000	4.322000e+03	4.322000e+03	4.322000e+03
mean	15298.534475	2.787888e-16	-3.364770e-17	-3.947311e-16
std	1721.534033	1.000116e+00	1.000116e+00	1.000116e+00
min	12347.000000	-2.856194e+00	-3.983816e-01	-2.328060e-01
25%	13812.250000	-4.854797e-01	-3.256058e-01	-1.962185e-01
50%	15297.500000	4.123652e-01	-2.228636e-01	-1.532204e-01
75%	16777.750000	7.452739e-01	3.720254e-02	-3.603693e-02
max	18287.000000	9.066842e-01	3.377196e+01	3.359474e+01

3）依照 3σ 原则去除离群点

```
#依照3σ原则去除离群点
k = data_RFM.describe()
std_M = k.loc['std','M']
std_F = k.loc['std','F']
std_R = k.loc['std','R']
import numpy as np
outlier_M = data_RFM[np.abs(data_RFM['M'])>3*std_M]  #变量M离群点，将离群数据保存到outlier_M
data_RFM.drop(outlier_M.index,inplace=True)            #去除离群点

data_RFM[np.abs(data_RFM['R'])>3*std_R]             #未发现离群点

outlier_F = data_RFM[np.abs(data_RFM['F'])>3*std_F]   #变量F离群点,将离群数据保存到outlier_F
data_RFM.drop(outlier_F.index,inplace=True)           #去除离群点
data_RFM
```

	CustomerID	R	F	M
1	12347	0.886508	0.376466	0.288849
2	12348	0.150073	-0.269954	-0.015280
3	12349	0.725098	-0.090155	-0.020084
4	12350	-2.220641	-0.329887	-0.192332
5	12352	0.543511	0.004025	-0.045760
...	...	...	...	...
4367	18280	-1.887732	-0.359853	-0.210947

```
#合并离群点数据
outlier = pd.concat([outlier_M,outlier_F])

#检测合并的离群点数据是否有重复数据
print(outlier.duplicated().any())
```

```
False
```

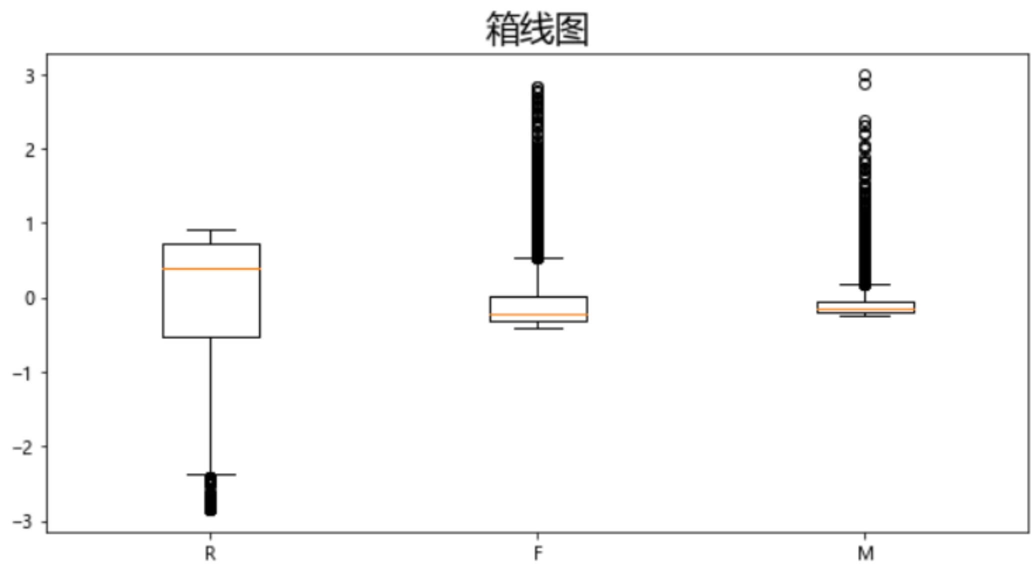
去除离群点后的数据描述（未进行标准化处理的，后续反标准化结果分析有用到）

```
data_RFM.describe()
```

	CustomerID	R	F	M
count	4260.000000	4260.000000	4260.000000	4.260000e+03
mean	15296.950000	615.913615	78.648357	1.363619e+03
std	1722.246113	99.339302	99.848036	2.074735e+03
min	12347.000000	334.000000	1.000000	1.776357e-15
25%	13809.750000	566.000000	17.750000	2.992275e+02
50%	15298.500000	657.000000	41.500000	6.458050e+02
75%	16777.250000	690.000000	99.000000	1.548585e+03
max	18287.000000	707.000000	696.000000	2.153590e+04

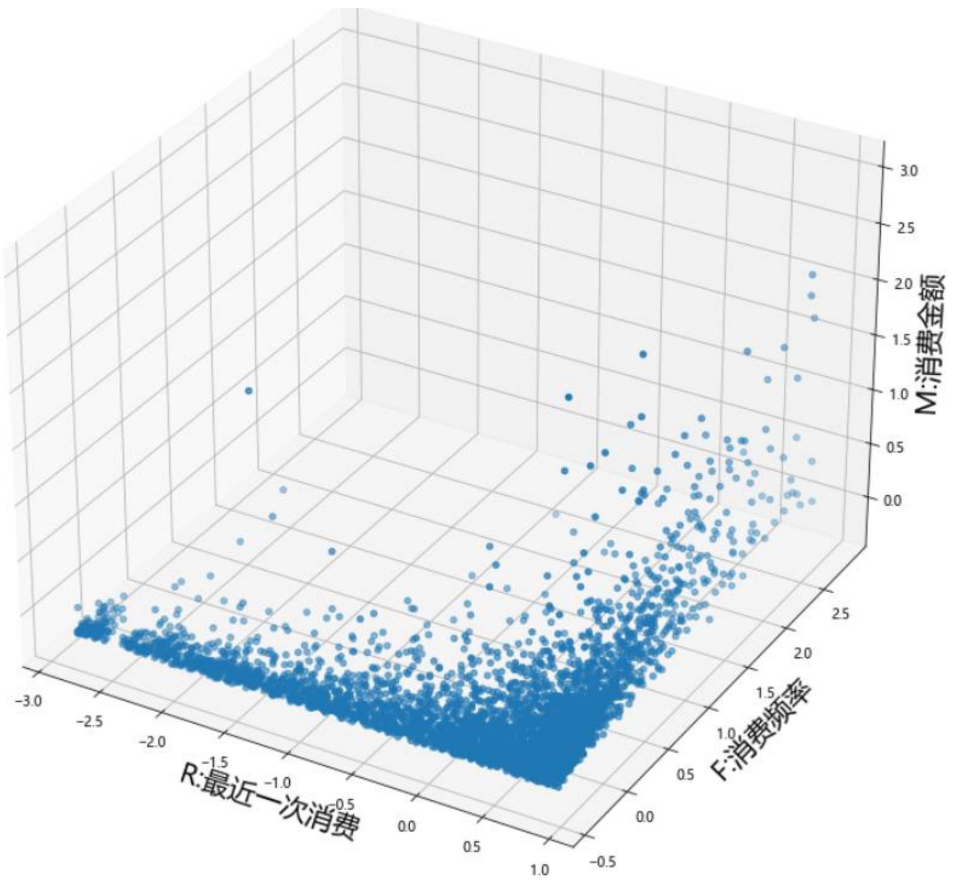
4）箱线图

```
#箱线图
import matplotlib.pyplot as plt

box_1, box_2, box_3 = data_RFM['R'], data_RFM['F'], data_RFM['M']

plt.figure(figsize=(10,5))#设置画布的尺寸
plt.title('箱线图',fontsize=20)#标题，并设定字号大小
labels = 'R','F','M'   #图例
plt.boxplot([box_1, box_2, box_3], labels = labels)
plt.show()#显示图像
```



5）变量分布

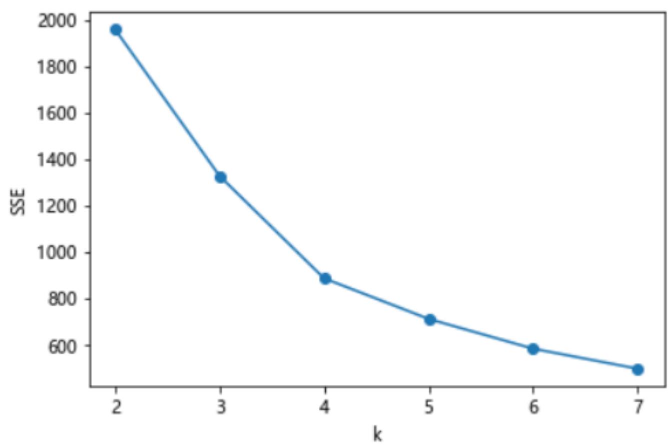
```
#根据三个变量特征，绘制三维散点图

from mpl_toolkits.mplot3d import Axes3D # 空间三维画图
#设置x、y、z轴
x=data_RFM['R']
y=data_RFM['F']
z=data_RFM['M']
#绘图
fig = plt.figure(figsize=(10,10))
ax = Axes3D(fig,auto_add_to_figure=False)
fig.add_axes(ax)
ax.scatter(x, y, z)
# 添加坐标轴
ax.set_xlabel('R:最近一次消费', fontdict={'size': 20})
ax.set_ylabel('F:消费频率', fontdict={'size': 20})
ax.set_zlabel('M:消费金额', fontdict={'size': 20})
plt.show()
```



5、建立模型和评价模型（聚成几类，效果好），聚类可视化

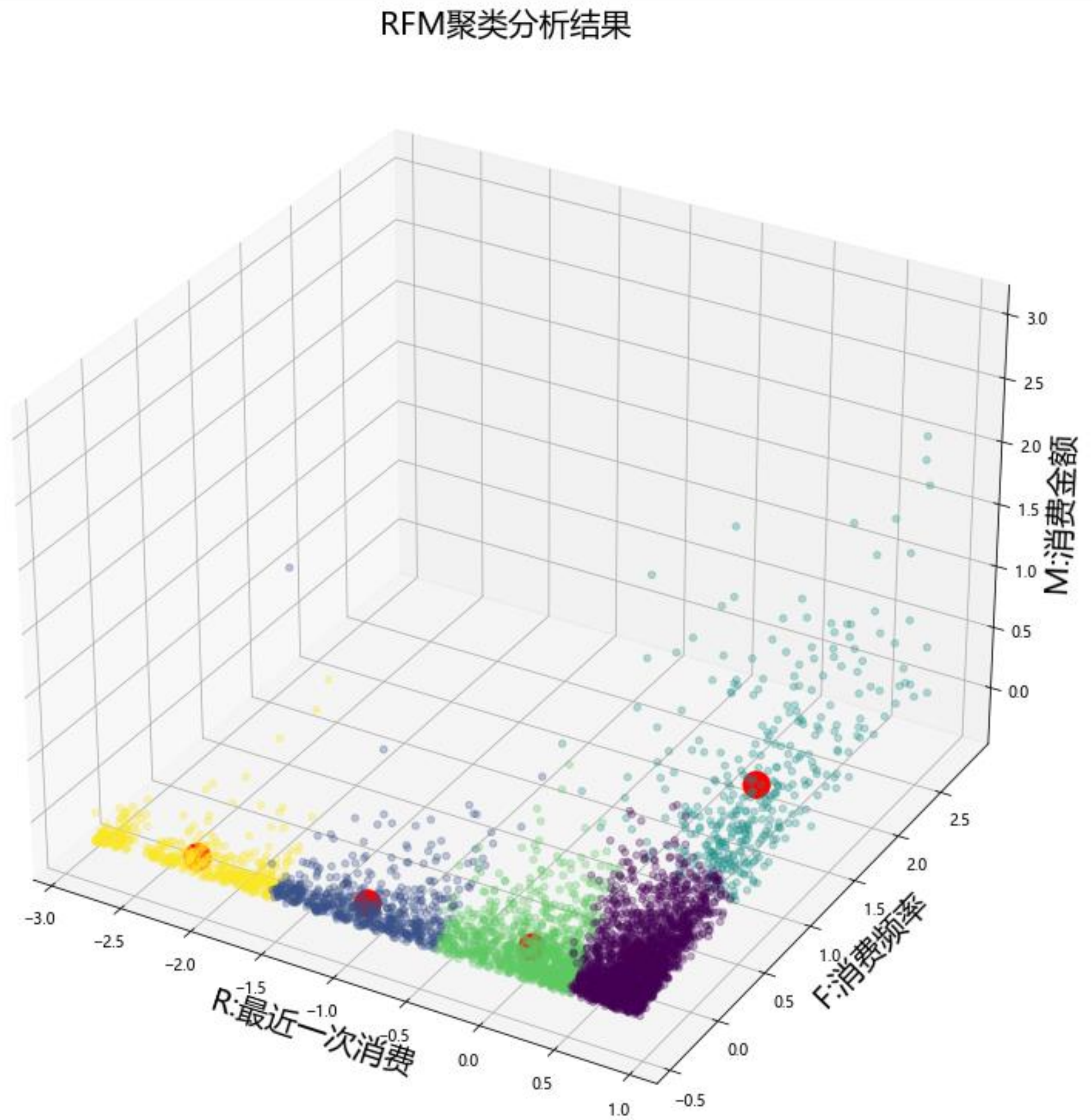
```
#碎石图
SSE = []
from sklearn.cluster import KMeans
for i in range(2,8):
    model = KMeans(n_clusters=i)
    model.fit(data_RFM.iloc[:,1:4])
    SSE.append(model.inertia_)

X = range(2,8)
plt.xlabel('k')
plt.ylabel('SSE')
plt.plot(X,SSE,'o-')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



1）聚类（上方碎石图显示聚类数为 4 或 5 效果较好）

```
# 聚类
from sklearn.cluster import KMeans
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']

model = KMeans(n_clusters = 5)  #指定聚类数
model.fit(data_RFM.iloc[:,1:4])

label_pred = model.labels_      #获取聚类标签
data_RFM['label'] = label_pred
#设置x、y、z轴
x=data_RFM['R']
y=data_RFM['F']
z=data_RFM['M']
#绘图
fig = plt.figure(figsize=(10,10))
ax = Axes3D(fig,auto_add_to_figure=False)
fig.add_axes(ax)
ax.scatter(x, y, z,c=label_pred,alpha=0.3)

centers = model.cluster_centers_ #获得中心点的坐标
ax.scatter(centers[0:5,0],centers[0:5,1],centers[0:5,2],c='red',s=300,alpha=1)#聚类质心
# 添加坐标轴
ax.set_xlabel('R:最近一次消费', fontdict={'size': 20})
ax.set_ylabel('F:消费频率', fontdict={'size': 20})
ax.set_zlabel('M:消费金额', fontdict={'size': 20})
ax.set_title('RFM聚类分析结果', fontdict={'size': 20})
plt.show()

print(centers)      #聚类质心数据
```

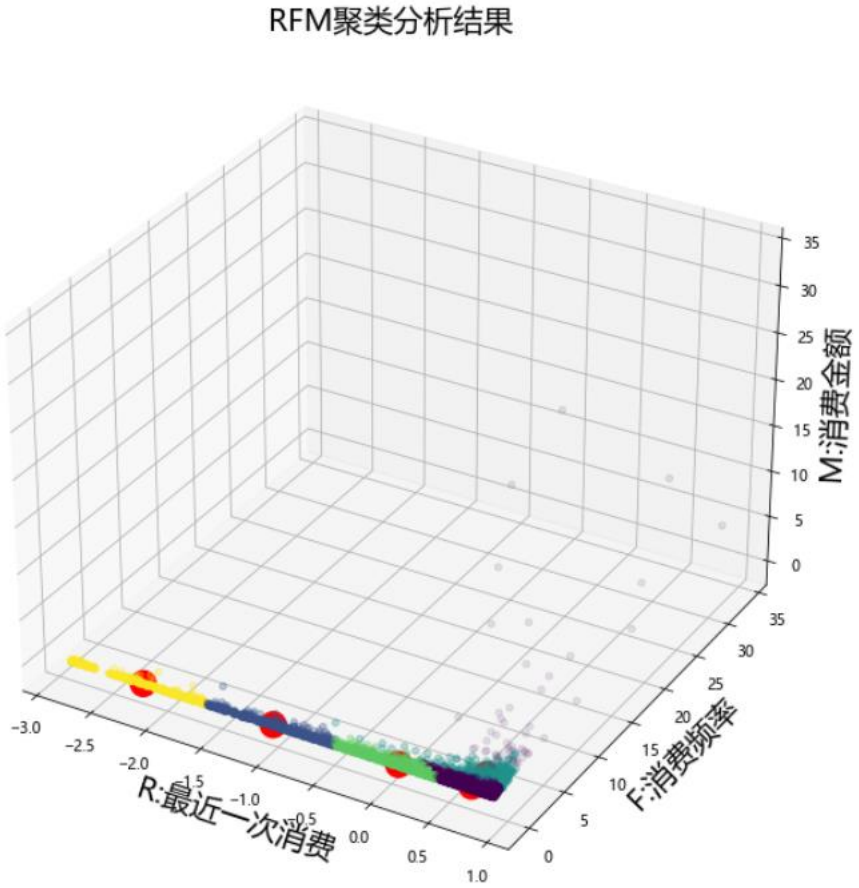


聚类质心数据：

```
[[ 0.07251632 -0.21892102 -0.12918667]
 [-1.02698256 -0.25681336 -0.16073191]
 [ 0.69642694 -0.08397428 -0.07421719]
 [ 0.74667726  1.08403191  0.45521573]
 [-2.1950712  -0.29845741 -0.18543799]]
```

2）添加清理的离群点数据，重新进行聚类结果展示：

```
#添加前部分清理掉的离群点数据，并在已经聚类完成的结果中展示
label_pred = model.labels_      #获取聚类标签
label_pred1 = outlier_result
data_RFM['label'] = label_pred
outlier['label'] = label_pred1  #离群点数据聚类标签

#设置x、y、z轴
x1 = data_RFM['R']
y1 = data_RFM['F']
z1 = data_RFM['M']
#离群点数据
x2 = outlier['R']
y2 = outlier['F']
z2 = outlier['M']
#绘图
fig = plt.figure(figsize=(8,8))
ax = Axes3D(fig,auto_add_to_figure=False)
fig.add_axes(ax)

ax.scatter(x1, y1, z1,c=label_pred,alpha=0.3)
ax.scatter(x2, y2, z2,c=label_pred1,alpha=0.1)

centers = model.cluster_centers_ #获得中心点的坐标
ax.scatter(centers[0:5,0],centers[0:5,1],centers[0:5,2],c='red',s=300,alpha=1)#聚类质心
# 添加坐标轴
ax.set_xlabel('R:最近一次消费', fontdict={'size': 20})
ax.set_ylabel('F:消费频率', fontdict={'size': 20})
ax.set_zlabel('M:消费金额', fontdict={'size': 20})
ax.set_title('RFM聚类分析结果', fontdict={'size': 20})
plt.show()
print(centers)      #聚类质心数据
```



3）反标准化：

```
import numpy as np

#k1 = data_RFM.describe()  #未标准化的数据描述
std_M = k1.loc['std','M']
std_F = k1.loc['std','F']
std_R = k1.loc['std','R']
mean_R = k1.loc['mean','R']
mean_F = k1.loc['mean','F']
mean_M = k1.loc['mean','M']

centers_zscore = np.zeros((5,3))
centers_zscore[:,0] = np.abs(centers[:,0]*std_R) + mean_R
centers_zscore[:,1] = np.abs(centers[:,0]*std_F) + mean_F
centers_zscore[:,2] = np.abs(centers[:,0]*std_M) + mean_M
```

```
#反标准化回去的聚类质心数据
centers_zscore
```

```
array([[  624.31288847,    111.00105994,   2522.69493338],
       [  718.93650199,    333.98481272,  10409.57117565],
       [  686.16596911,    256.75994113,   7678.147933  ],
       [  691.14767172,    268.49949221,   8093.37274354],
       [  834.73815602,    606.87533583,  20061.63665892]])
```

4）分析

聚类 1：变量 R 高于平均水平，变量 F、M 均低于平均水平，属于一般发展客户。
聚类 2：变量 F、M、R 均低于平均水平，属于一般保持客户。
聚类 3：变量 R 明显高于平均水平，变量 F、M 均稍低于平均水平，属于重要发展客户。
聚类 4：变量 R、F 和 M 均明显高于平均水平,属于重要价值客户。
聚类 5：变量 R 明显低于平均水平，变量 F、M 均低于平均水平，属于一般挽留客户。

```
data_RFM
```

	CustomerID	R	F	M	label
1	12347	0.886508	0.376466	0.288849	2
2	12348	0.150073	-0.269954	-0.015280	0
3	12349	0.725098	-0.090155	-0.020084	2
4	12350	-2.220641	-0.329887	-0.192332	4
5	12352	0.543511	0.004025	-0.045760	2
...	...	...	...	...	...
4367	18280	-1.887732	-0.359853	-0.210947	4
4368	18281	-0.909182	-0.372696	-0.223024	1
4369	18282	0.836067	-0.347010	-0.211432	2
4370	18283	0.876420	2.833717	0.020745	3
4371	18287	0.482982	-0.102998	-0.010433	2

4272 rows × 5 columns

```
len(np.unique(data_RFM['CustomerID']))
```

```
4272
```

客户总人数：4272；
聚类结果表格：

```
count = data_RFM.groupby('label')['label'].count()
```

```
centers
```

```
array([[ 0.07251632, -0.21892102, -0.12918667],
       [-1.02698256, -0.25681336, -0.16073191],
       [ 0.69642694, -0.08397428, -0.07421719],
       [ 0.74667726,  1.08403191,  0.45521573],
       [-2.1950712 , -0.29845741, -0.18543799]])
```

```
data = pd.DataFrame(columns=['聚类','聚类个数','ZR','ZF','ZM'])
```

```
data['聚类个数']= count.values
data['聚类']=['聚类1','聚类2','聚类3','聚类4','聚类5']
data.iloc[:,2:5] = centers
```

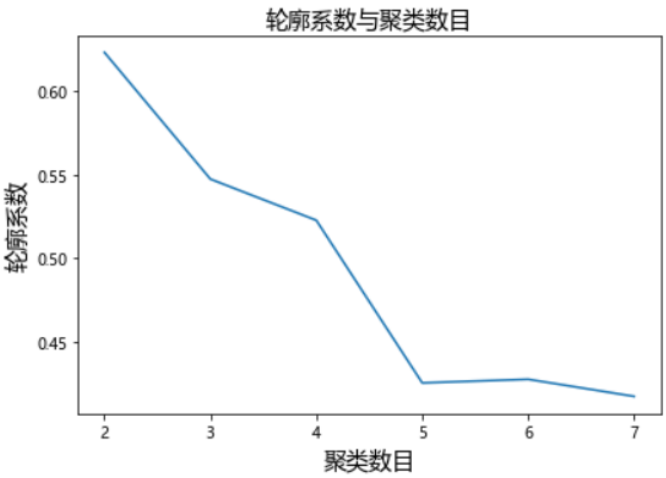
```
data
```

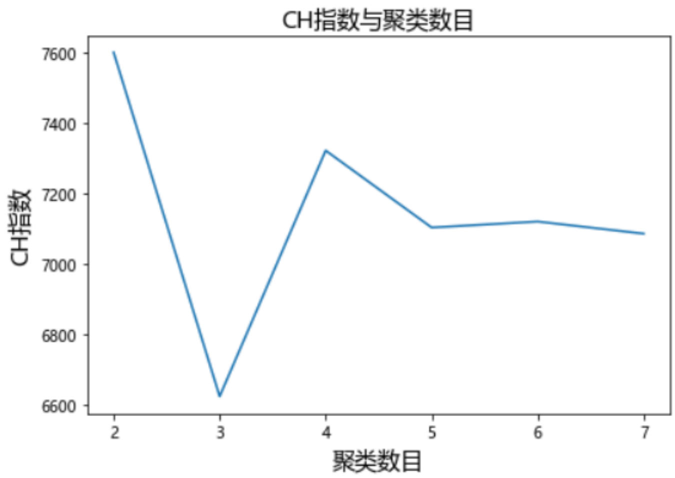
```
data
```

	聚类	聚类个数	ZR	ZF	ZM
0	聚类1	987	0.072516	-0.218921	-0.129187
1	聚类2	606	-1.026983	-0.256813	-0.160732
2	聚类3	1821	0.696427	-0.083974	-0.074217
3	聚类4	381	0.746677	1.084032	0.455216
4	聚类5	477	-2.195071	-0.298457	-0.185438

5）聚类模型评价

```
from sklearn.metrics import silhouette_score
from sklearn import metrics
import warnings
#from sklearn.cluster import KMeans
#import matplotlib.pyplot as plt
warnings.filterwarnings(action = 'ignore')
plt.figure(figsize=(15,10))
#聚类模型评价
silhouettescore = []
calinski_harabasz_score = []
for i in range(2,8):
    #轮廓系数
    kmeans = KMeans(n_clusters=i,random_state=100).fit(data_RFM.iloc[:,1:4])
    score1 = silhouette_score(data_RFM.iloc[:,1:4],kmeans.labels_)
    silhouettescore.append(score1)

    # CH指数
    score2 = metrics.calinski_harabasz_score(data_RFM.iloc[:,1:4],kmeans.labels_)
    calinski_harabasz_score.append(score2)

plt.subplot(2,2,1)
#plt.figure(figsize=(10,6))
plt.plot(range(2,8),silhouettescore,linewidth=1.5,linestyle='-')
plt.xlabel('聚类数目', fontdict={'size': 15})
plt.ylabel('轮廓系数', fontdict={'size': 15})
plt.title('轮廓系数与聚类数目', fontdict={'size': 15})

plt.subplot(2,2,2)
plt.plot(range(2,8),calinski_harabasz_score,linewidth=1.5,linestyle='-')
plt.xlabel('聚类数目', fontdict={'size': 15})
plt.ylabel('CH指数', fontdict={'size': 15})
plt.title('CH指数与聚类数目', fontdict={'size': 15})
plt.show()
```





分析：在上图中，聚类数目为 5 时轮廓系数图像的畸变程度最大，同时在 CH 指数与聚类数目图中，聚类为 5 类也是较高水平，因此选择聚为 5 类较好。

6、模型应用

对于上述 5 类客户均可采取进行会员升级，以此更好地服务客户，对于优质客户进行会员升级定制一站式服务，对于新用户以及一般客户也可通过会员升级的形式刺激消费，进而留住客户。

在老客户中，采取支持购物积分兑换，回馈优质老客户，应用到聚类 1、3、4、的客户效果应该会比较不错的，可以进一步刺激客户的消费热情。

交叉销售

交叉销售的销售策略，在销售产品的同时有额外的产品推荐和优惠，这对于新老客户或者是优质客户都有一定的效果，个人觉得对于吸引新用户（一般客户）效果会比较好，如聚类 1，2。