****

航空公司客户价值分析实验报告

|  |  |
| --- | --- |
| 学生姓名 | 谭哲文 |
| 学 号 | 8202191123 |
| 专业班级 | 计科2105 |
| 指导教师 | 王 斌 |
| 学 院 | 计算机学院 |
| 完成时间 | 2023.05.24 |

目录

1、实验概述 ..1

（1）实验目的 1

（2）实验内容及要求 1

（3）主要知识点，重点难点............................................................................1

2、需求分析 2

（1）背景分析 2

（2）具体实现思路 2

3、总体设计................................................................................................................2

（1）基本功能 2

4、详细设计与实现....................................................................................................2

（1）模块（函数）的设计与实现.....................................................................3

5、运行结果与客户价值.......................…………………………………........................……....7

（1）运行结果....................................................................................................7

（2）客户价值分析............................................................................................10

6、结束语..................................................................................................................14

7、源代码..................................................................................................................14

1. **实验概述**
2. **实验目的**

重点结合航空公司客户价值分析的案例介绍K-Means聚类算法在客户价值分析中的应用。此外，介绍基于RFM客户价值分析模型的不足，使用K-Means算法构建航空客户价值分析LRFMC模型，详细的描述数据分析的整个过程。

1. **实验内容及要求**

1.基本要求

（1）了解RFM模型的基本原理。

（2）掌握K-Means算法的基本原理与使用方法。

（3）比较不同类别客户的客户价值，制定相应的营销策略。

2.实验内容

（1）处理数据缺失值与异常值。

（2）构建航空客户价值分析的关键特征。

（3）标准化LRFMC 5个特征。

（4）构建K-Means聚类模型。

（5）评价K-Means聚类模型。

1. **主要知识点、重点与难点**

1.主要知识点

（1）RFM模型的基本原理。

（2）K-Means算法的基本原理与使用方法。

（3）比较不同类别客户的客户价值，制定相应的营销策略。

2.重点

（1）航空客户价值分析的步骤和流程。

（2）RFM模型的基本原理。

（3）K-Means算法的基本原理与使用方法。

（4）比较不同类别客户的客户价值。

3.难点

（1）RFM模型的基本原理。

（2）KMeans算法的基本原理与使用方法。

1. **需求分析**
2. **背景分析**

在面向客户制定运营策略、营销策略时，希望能够针对不同的客户推行不同的策略，实现精准化运营，以期获取最大的转化率。客户关系管理是精准化运营的基础，而客户关系管理的核心是客户分类。通过客户分类，对客户群体进行细分，区别出低价值客户、高价值客户，对不同的客户群体开展不同的个性化服务，将有限的资源合理地分配给不同价值的客户，实现效益最大化。

本章将使用航空公司客户数据，结合RFM模型，采用K-Means聚类算法，对客户进行分群，比较不同类别客户的客户价值，从而制定相应的营销策略。

1. **总体设计**
2. **基本功能**
3. 读取数据：将csv文件中的数据读取，方便下一步操作
4. 数据预处理：

A.丢弃票价为空的记录。

B.保留票价不为0的，或者平均折扣率不为0且总飞行公里数大于0的记录。

C.丢弃年龄大于100的记录。

1. 构造LRFMC模型的五个属性：其中RFMC均可直接获取，L通过计算获得。
2. 将五种属性合并，并进行标准化
3. 构建模型，进行训练
4. 绘制客户信息雷达图
5. 将客户数据分类保存
6. **详细设计与实现**
7. **模块（函数）的设计与实现**
8. 读取数据：

datafile = 'C:/Users/Dell/Desktop/python/air\_data.csv'  # 航空原始数据路径

cleanedfile = 'data\_cleaned.csv'  # 数据清洗后保存的文件路径

# 读取数据

airline\_data = pd.read\_csv(datafile,encoding = 'utf-8')

print('原始数据的规格为：',airline\_data.shape)

从air\_data.csv文件中读取客户信息，并新建一个data\_cleaned.csv文件用来保存下一步数据预处理后的数据

1. 数据预处理并保存：

# 去除票价为空的记录

# loc：Selection by Label函数，即为按标签取数据

airline\_notnull = airline\_data.loc[airline\_data['SUM\_YR\_1'].notnull() &

                                   airline\_data['SUM\_YR\_2'].notnull(),:]

print('删除缺失记录后数据的规格为：',airline\_notnull.shape)

# 只保留票价非零的，或者平均折扣率不为0且总飞行公里数大于0的记录。

index1 = airline\_notnull['SUM\_YR\_1'] != 0

index2 = airline\_notnull['SUM\_YR\_2'] != 0

index3 = (airline\_notnull['SEG\_KM\_SUM']> 0) & (airline\_notnull['avg\_discount'] != 0)

index4 = airline\_notnull['AGE'] > 100  # 去除年龄大于100的记录

airline = airline\_notnull[(index1 | index2) & index3 & ~index4]

print('数据清洗后数据的规格为：',airline.shape)

airline.to\_csv(cleanedfile)  # 保存处理后的数据

airline = pd.read\_csv(cleanedfile, encoding = 'utf-8')

数据处理主要去除三种数据：票价为空的、均折扣率为0且总飞行公里数为0的记录、年龄大于100的，这类数据对于数据分析作用不大，直接舍弃。

将处理后的数据保存在第一步设定好的文件里

1. 构造模型所需数据：

# 选取需求属性

# 入会时间，观测窗口结束时间，最后一次乘机至观测窗口结束的时长，观测期间飞行次数，观测期间飞行里程数，平均折扣率

airline\_selection = airline[['FFP\_DATE','LOAD\_TIME','LAST\_TO\_END',

                                     'FLIGHT\_COUNT','SEG\_KM\_SUM','avg\_discount']]

print('筛选的属性前10行为：\n',airline\_selection.head(10))

# 构造属性L

# pandas.to\_datetime（arg，errors =‘raise’，utc = None，format = None，unit = None ）自动识别时间格式

# astype 数据类型转换

L = pd.to\_datetime(airline\_selection['LOAD\_TIME']) - pd.to\_datetime(airline\_selection['FFP\_DATE'])

print('未处理的L的前5行是：\n',L.head(5))

# split：对字符串进行切片，观察之前数据发现，当前L的第一个字符串即为天数

L = L.astype('str').str.split().str[0]

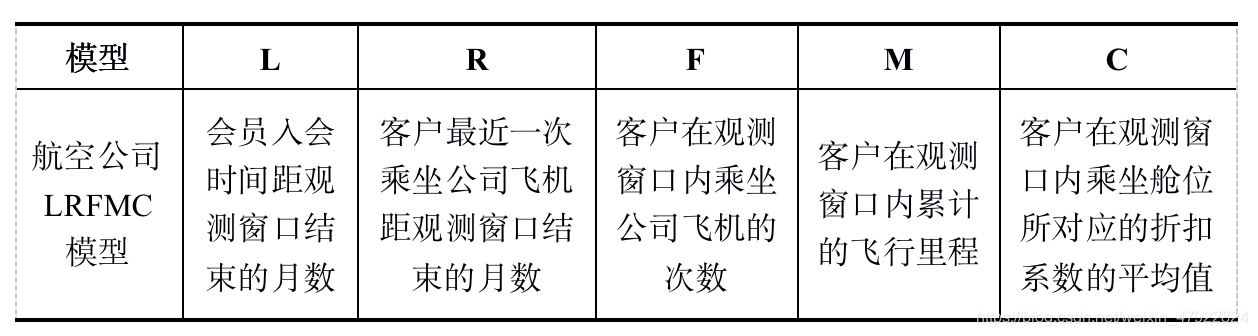
print('取数据的L的前5行是：\n',L.head(5))

L = L.astype('int')/30

print('观测期间的月数（L）前5行是：\n',L.head(5))



源文件包含信息如上图所示，与模型所需数据对比发现：



仅有L需要重新计算，剩余四个属性可以直接读取。

计算L的方法也很简单，将窗口日期相减，得到总天数，除以30，得到大致的月数。

至此，所有属性均准备好，可以进行下一步工作。

1. 合并属性

# 合并属性

# pd.concat(obj,axis = 1),数据连接函数，obj可以是数组，列表或者数据框；

# axis = 0 表示按行连接（行增加连接），axis = 1 表示按列连接（列增加连接）

# iloc[:,:]前面的冒号取行数，后面的冒号取列数

airline\_features = pd.concat([L,airline\_selection.iloc[:,2:]],axis = 1)

airline\_features.columns = ['L','R','F','M','C']

print('定义的LRFMC属性前10行为：\n',airline\_features.head(10))

将准备好的五个属性合并，按列连接。

1. 属性数据标准化

# 数据标准化：消除数据量级的影响

# StandardScaler():通过删除平均值并缩放到单位方差来标准化特征

# 通过 fit\_transform() 对训练集进行归一化

data = StandardScaler().fit\_transform(airline\_features)

# 保存为二进制文件

np.savez('airline\_scale.npz',data)

print('LRFMC五个属性经标准化后前10个值为：\n',data[:10,:])

# 读取标准化后的数据

airline\_scale = np.load('airline\_scale.npz')['arr\_0']

因为各个属性的量级不同，直接进行模型训练会有较大的偏差，因此先进行数据标准化，处理为同一量级再进行下一步工作。

1. 建立模型，进行训练

k = 5  # 确定聚类中心数

# 构建模型，随机种子设为666

kmeans\_model = KMeans(n\_clusters=k, random\_state=666)

fit\_kmeans = kmeans\_model.fit(airline\_scale)  # 模型训练

# 查看聚类结果

kmeans\_cc = kmeans\_model.cluster\_centers\_  # 聚类中心

print('各类聚类中心为：\n',kmeans\_cc)

kmeans\_labels = kmeans\_model.labels\_  # 样本的类别标签

print('各样本的类别标签为：\n',kmeans\_labels)

r1 = pd.Series(kmeans\_model.labels\_).value\_counts()  # 统计不同类别样本的数目

print('最终每个类别的数目为：\n',r1)

调用KMeanns函数，进行聚类模型的建立与训练。

查看聚类结果，进行检查，观察个聚类数据的分布情况。

# 输出聚类分群的结果

cluster\_center = pd.DataFrame(kmeans\_model.cluster\_centers\_,

             columns = ['ZL','ZR','ZF','ZM','ZC'])   # 将聚类中心放在数据框中

cluster\_center.index = pd.DataFrame(kmeans\_model.labels\_ ).\

                  drop\_duplicates().iloc[:,0]  # 将样本类别作为数据框索引

print(cluster\_center)

将训练后的聚类结果储存在数据框中，方便下一步绘图时使用。

1. 雷达图的设计与绘制：

# 客户分群雷达图

labels = ['ZL','ZR','ZF','ZM','ZC']

legen = ['客户群' + str(i + 1) for i in cluster\_center.index]  # 客户群命名，作为雷达图的图例

lstype = ['-','--',(0, (3, 5, 1, 5, 1, 5)),':','-.']

kinds = list(cluster\_center.iloc[:, 0])

# 由于雷达图要保证数据闭合，因此再添加L列，并转换为 np.ndarray

cluster\_center = pd.concat([cluster\_center, cluster\_center[['ZL']]], axis=1)

centers = np.array(cluster\_center.iloc[:, 0:])

# 分割圆周长，并让其闭合

n = len(labels)

angle = np.linspace(0, 2 \* np.pi, n, endpoint=False)

angle = np.concatenate((angle, [angle[0]]))

如上设计雷达图，但要注意的是，雷达图实现的点直接的连线，为保证雷达图是闭合的，需要最后增加一类数据，实现起点和终点之间的连线。

# 绘图

fig = plt.figure(figsize = (8,6))

ax = fig.add\_subplot(111, polar=True)  # 以极坐标的形式绘制图形

plt.rcParams['font.sans-serif'] = ['SimHei']  # 用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False  # 用来正常显示负号

# 画线

for i in range(len(kinds)):

    ax.plot(angle, centers[i], linestyle=lstype[i], linewidth=2, label=kinds[i])

# 添加属性标签

ang = angle \* 180 / np.pi

ax.set\_thetagrids(ang[:-1], labels)

#ax.set\_thetagrids(angle \* 180 / np.pi, labels)

plt.title('客户特征分析雷达图')

plt.legend(legen)

plt.show()

plt.close

调用matplotlib.pyplot进行雷达图的绘制。

1. 将客户数据分类保存：

with open('data\_cleaned.csv','r',encoding='utf-8')as fp:

    reader=csv.reader(fp)

    t=0

    for rows in enumerate(reader):

        if kmeans\_labels[t]==0:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/python/一般客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==1:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/python/低价值客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==2:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/python/重要挽留客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==3:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/python/重要保持客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==4:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/python/重要发展客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

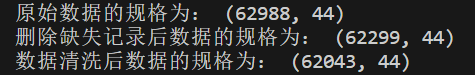
                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

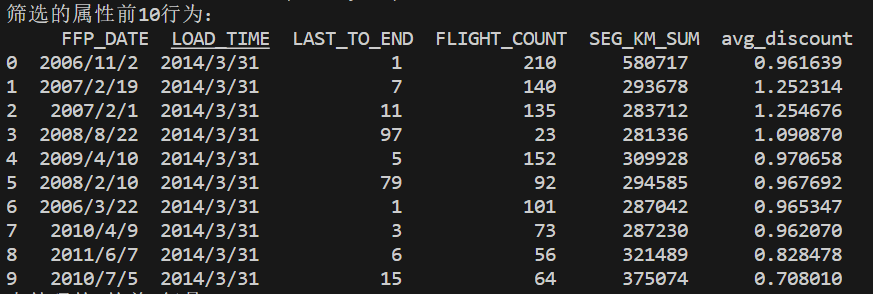
            t=t+1

1. **运行结果与客户价值分析**
2. **运行结果展示**
3. 数据预处理



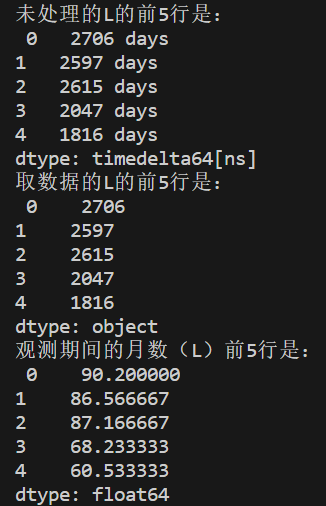
可以看到处理后删减了945条无用数据

1. 筛选六种数据进行五种属性的构建



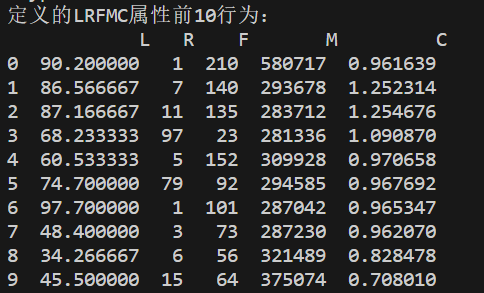
前两列数据用来计算L

1. L的计算

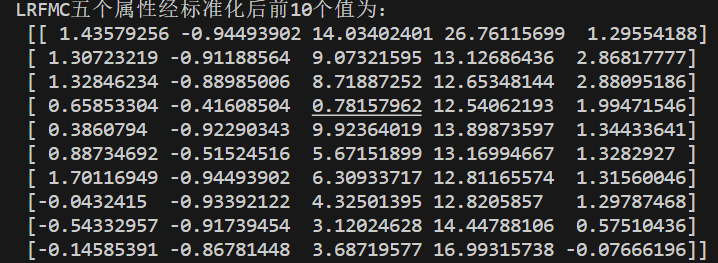


天数相减，除以30，得到大致的月数（L）

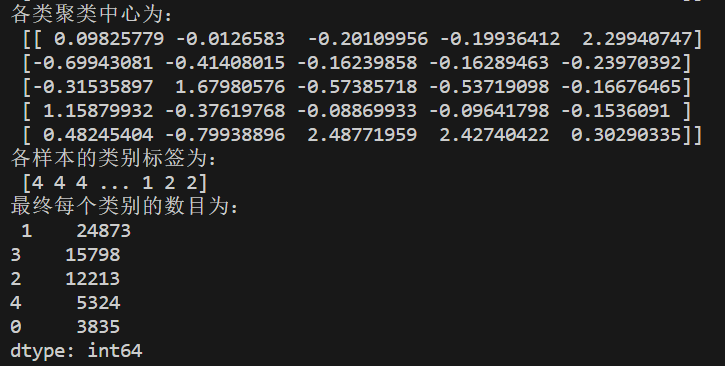
1. LRFMC模型的五个属性



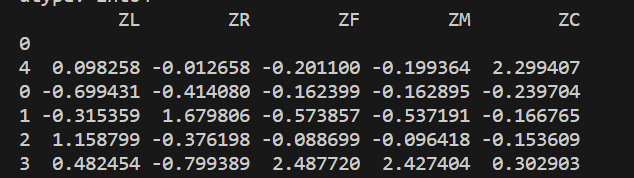
1. 将属性进行标准化



1. 进行模型训练（聚类中心数为5，其余结果下一部分分析）

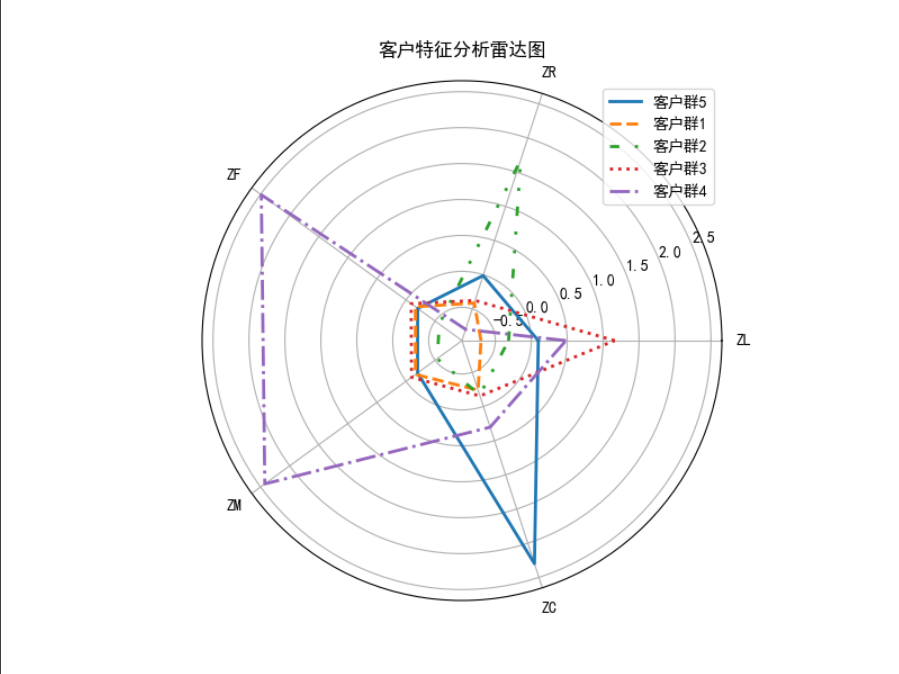


1. 输出聚类结果



利用上述数据进行雷达图的绘制。

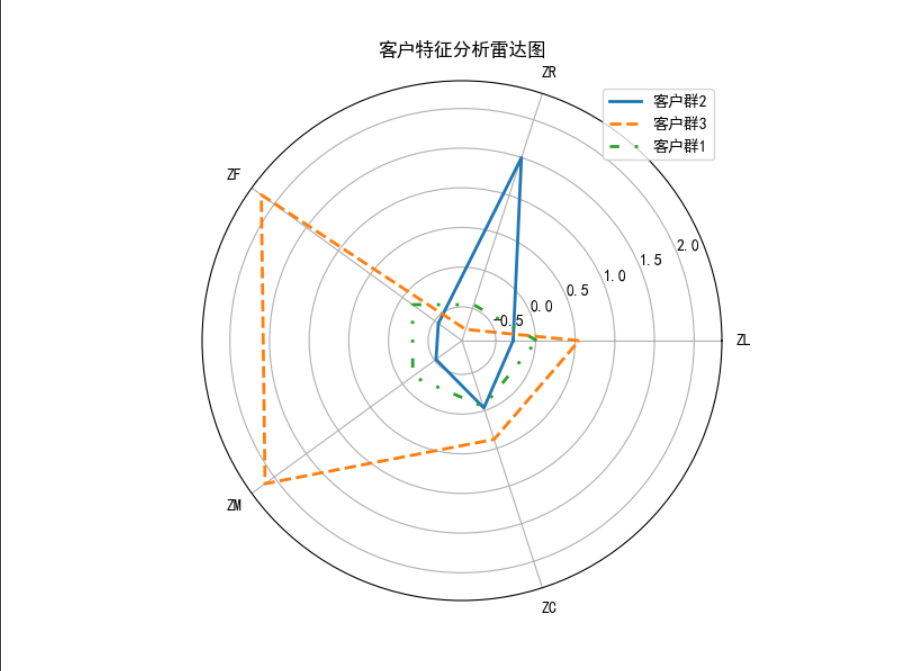
1. 雷达图的绘制（k = 5）



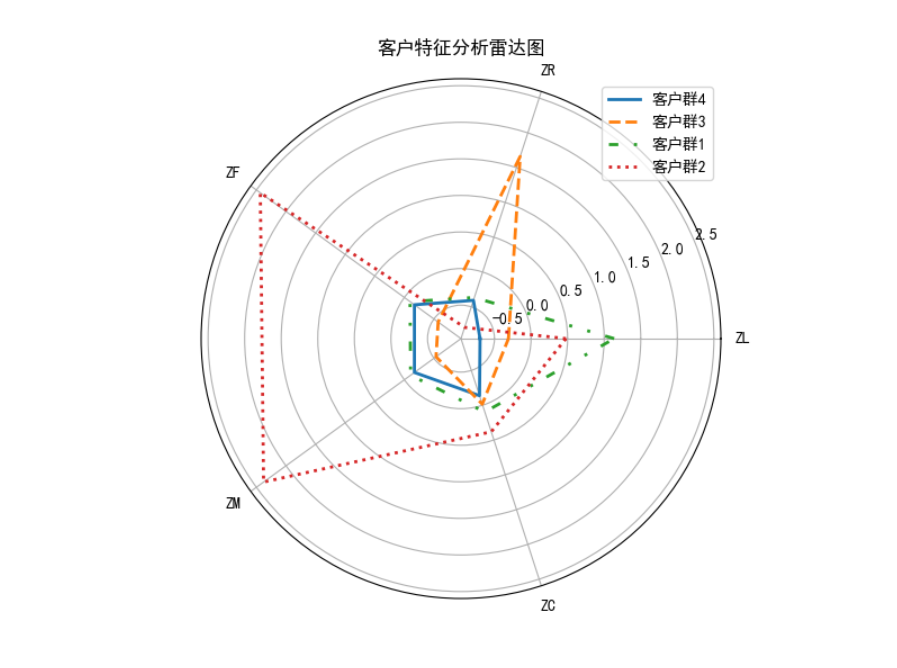
1. **客户价值分析**
2. k值的确立

更改代码中k的值可以得出k为不同值时的结果

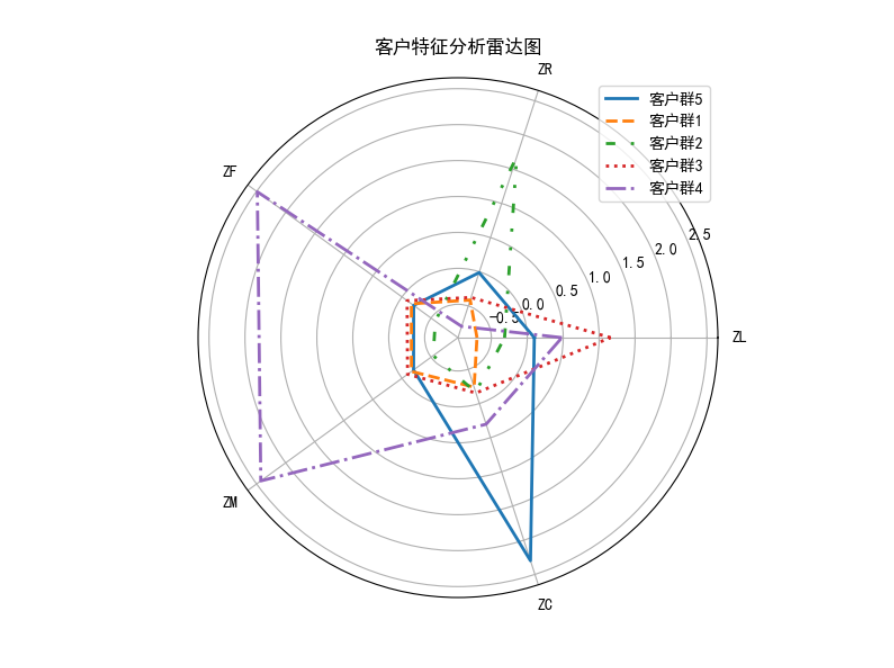
K = 3



K = 4



K = 5



观察可得，当k = 5时，聚类分析的结果更为多样，分类也更为细致。

1. 雷达图数据分析：

由雷达图中可知：

客群1在L,R，F,M,C处都小，说明客群1是新入会员较多客户群。

客群2在R处最大，在其他特征处都较小，属于入会时间短的低价值客户群。

客群3在L处最大，在R处值较小，其他特征适中，说明客群3属于入会时间长，飞行频率高的高价值客户。

客群4在F，M处最大，且在R处最小，说明客户群体4频繁乘机并且最近也有乘机记录。

客群5在C处最大，在F,M处的值较小，说明客群5是偏好坐高级舱的客户群（一般航班舱位等级越高，折扣系数越大）。

根据以上分析，将客户分为五类：重要保持客户，重要发展客户，重要挽留客户，一般客户，低价值客户。

具体特点如下：

（1）重要保持客户：这类客户的平均折扣率（C）较高（一般所乘航班的舱位等级较高），最近乘坐过本公司航班（R）低，乘坐的次数（F）或里程（M）较高。他们是航空公司的高价值客户，是最为理想的客户类型，对航空公司的贡献最大，所占比例却较小。航空公司应该优先将资源投放到他们身上，对他们进行差异化管理和一对一营销，提高这类客户的忠诚度与满意度，尽可能延长这类客户的高水平消费。

（2）重要发展客户：这类客户的平均折扣率（C）较高，最近乘坐过本公司航班（R）低，但乘坐次数（F）或乘坐里程（M）较低。这类客户入会时长（L）短，他们是航空公司的潜在价值客户。虽然这类客户的当前价值并不是很高，但却有很大的发展潜力。航空公司要努力促使这类客户增加在本公司的乘机消费和合作伙伴处的消费，也就是增加客户的钱包份额。通过客户价值的提升，加强这类客户的满意度，提高他们转向竞争对手的转移成本，使他们逐渐成为公司的忠诚客户。

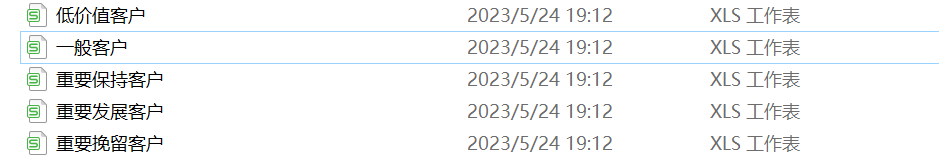
（3）重要挽留客户：这类客户过去所乘航班的平均折扣率（C），乘坐次数（F）或者里程（M）较高，但是较长时间已经没有乘坐本公司的航班（R）高或是乘坐频率变小。他们客户价值变化的不确定性很高。由于这些客户衰退的原因各不相同，所以掌握客户的最新信息，维持与客户的互动就显得尤为重要。航空公司应该根据这些客户的最近消费时间，消费次数的变化情况，推测客户消费的异动状况，并列出客户名单，对其重点联系，采取一定的营销手段，延长客户的生命周期。

（4）一般与低价值客户。这类客户所乘航班的平均折扣率（C）很低，较长时间没有乘坐过本公司航班（R）高，乘坐的次数（F）或里程（M）较低，入会时长（L）短。他们是航空公司的一般用户与低价值客户，可能是在航空公司机票打折促销时，才会乘坐本公司航班。

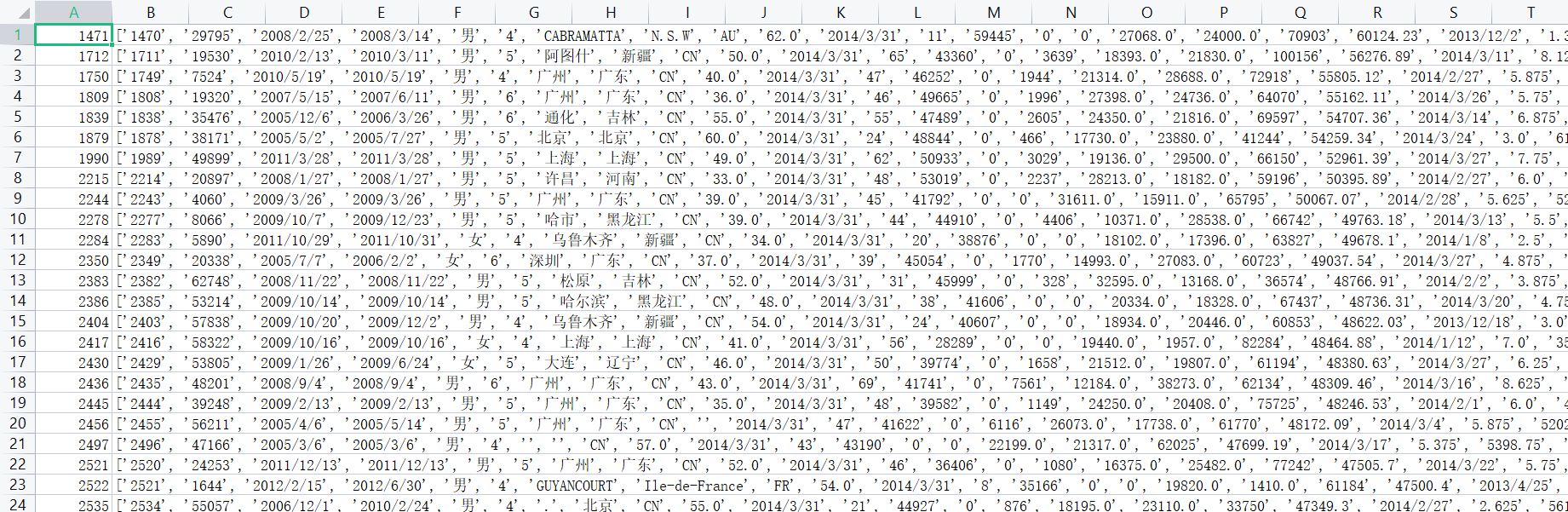
根据上述分类标准，分类如下：

|  |  |
| --- | --- |
| 客户群 | 客户类型 |
| 4 | 重要保持客户 |
| 5 | 重要发展客户 |
| 3 | 重要挽留客户 |
| 1 | 一般客户 |
| 2 | 低价值客户 |

1. 将客户信息分类保存：



将不同类型的客户信息分别保存在不同的文件中，方便之后使用。



1. **结束语**

【1】通过这次实验，我了解并使用到K-means聚类算法。在这次实验中我遇到了：

（1）ValueError: The number of FixedLocator locations, usually from a call to set\_ticks, does not match the number of ticklabels这个错误,经过长时间摸索发现了data = np.concatenate((data, [data[0]])) angles = np.concatenate((angles, [angles[0]]))这两句代码造成的，因为要线条闭合，所以在后面进行了拼接，网上有些人的解决方案是注释掉，但会出现一个问题，那就是雷达图不闭合。正确改法是：在radar\_labels中重复第一个元素。

（2）另一个错误是：AttributeError: ‘Text’ object has no property ‘frac’，经过一些时间的摸索，发现这个错误是在新版本的matplotlib中这个 ‘Text’ object移除了’frac’属性。解决方法：去掉变量。

【2】总之，数据分析不易，是综合性很强的事物，需要自身不断摸索琢磨！

1. **源代码**

import numpy as np

import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans  # 导入kmeans算法

import matplotlib.pyplot as plt

import csv

"""

    L:会员入会时间距观测窗口结束的月数

    R:客户最后一次乘坐公司飞机距观测窗口结束的月数

    F:客户在观测窗口内乘坐公司飞机的次数

    M:客户在观测窗口内累计的飞行里程

    C:客户在观测窗口内乘坐舱位所对应的折扣系数的平均值

"""

datafile = 'C:/Users/Dell/Desktop/python/air\_data.csv'  # 航空原始数据路径

cleanedfile = 'data\_cleaned.csv'  # 数据清洗后保存的文件路径

# 读取数据

airline\_data = pd.read\_csv(datafile,encoding = 'utf-8')

print('原始数据的规格为：',airline\_data.shape)

# 去除票价为空的记录

# loc：Selection by Label函数，即为按标签取数据

airline\_notnull = airline\_data.loc[airline\_data['SUM\_YR\_1'].notnull() &

                                   airline\_data['SUM\_YR\_2'].notnull(),:]

print('删除缺失记录后数据的规格为：',airline\_notnull.shape)

# 只保留票价非零的，或者平均折扣率不为0且总飞行公里数大于0的记录。

index1 = airline\_notnull['SUM\_YR\_1'] != 0

index2 = airline\_notnull['SUM\_YR\_2'] != 0

index3 = (airline\_notnull['SEG\_KM\_SUM']> 0) & (airline\_notnull['avg\_discount'] != 0)

index4 = airline\_notnull['AGE'] > 100  # 去除年龄大于100的记录

airline = airline\_notnull[(index1 | index2) & index3 & ~index4]

print('数据清洗后数据的规格为：',airline.shape)

airline.to\_csv(cleanedfile)  # 保存处理后的数据

airline = pd.read\_csv(cleanedfile, encoding = 'utf-8')

# 选取需求属性

# 入会时间，观测窗口结束时间，最后一次乘机至观测窗口结束的时长，观测期间飞行次数，观测期间飞行里程数，平均折扣率

airline\_selection = airline[['FFP\_DATE','LOAD\_TIME','LAST\_TO\_END',

                                     'FLIGHT\_COUNT','SEG\_KM\_SUM','avg\_discount']]

print('筛选的属性前10行为：\n',airline\_selection.head(10))

# 构造属性L

# pandas.to\_datetime（arg，errors =‘raise’，utc = None，format = None，unit = None ）自动识别时间格式

# astype 数据类型转换

L = pd.to\_datetime(airline\_selection['LOAD\_TIME']) - pd.to\_datetime(airline\_selection['FFP\_DATE'])

print('未处理的L的前5行是：\n',L.head(5))

# split：对字符串进行切片，观察之前数据发现，当前L的第一个字符串即为天数

L = L.astype('str').str.split().str[0]

print('取数据的L的前5行是：\n',L.head(5))

L = L.astype('int')/30

print('观测期间的月数（L）前5行是：\n',L.head(5))

# 合并属性

# pd.concat(obj,axis = 1),数据连接函数，obj可以是数组，列表或者数据框；

# axis = 0 表示按行连接（行增加连接），axis = 1 表示按列连接（列增加连接）

# iloc[:,:]前面的冒号取行数，后面的冒号取列数

airline\_features = pd.concat([L,airline\_selection.iloc[:,2:]],axis = 1)

airline\_features.columns = ['L','R','F','M','C']

print('定义的LRFMC属性前10行为：\n',airline\_features.head(10))

# 数据标准化：消除数据量级的影响

# StandardScaler():通过删除平均值并缩放到单位方差来标准化特征

# 通过 fit\_transform() 对训练集进行归一化

data = StandardScaler().fit\_transform(airline\_features)

# 保存为二进制文件

np.savez('airline\_scale.npz',data)

print('LRFMC五个属性经标准化后前10个值为：\n',data[:10,:])

# 读取标准化后的数据

airline\_scale = np.load('airline\_scale.npz')['arr\_0']

k = 5  # 确定聚类中心数

# 构建模型，随机种子设为666

kmeans\_model = KMeans(n\_clusters=k, random\_state=666)

fit\_kmeans = kmeans\_model.fit(airline\_scale)  # 模型训练

# 查看聚类结果

kmeans\_cc = kmeans\_model.cluster\_centers\_  # 聚类中心

print('各类聚类中心为：\n',kmeans\_cc)

kmeans\_labels = kmeans\_model.labels\_  # 样本的类别标签

print('各样本的类别标签为：\n',kmeans\_labels)

r1 = pd.Series(kmeans\_model.labels\_).value\_counts()  # 统计不同类别样本的数目

print('最终每个类别的数目为：\n',r1)

# 输出聚类分群的结果

cluster\_center = pd.DataFrame(kmeans\_model.cluster\_centers\_,

             columns = ['ZL','ZR','ZF','ZM','ZC'])   # 将聚类中心放在数据框中

cluster\_center.index = pd.DataFrame(kmeans\_model.labels\_ ).\

                  drop\_duplicates().iloc[:,0]  # 将样本类别作为数据框索引

print(cluster\_center)

# 客户分群雷达图

labels = ['ZL','ZR','ZF','ZM','ZC']

legen = ['客户群' + str(i + 1) for i in cluster\_center.index]  # 客户群命名，作为雷达图的图例

lstype = ['-','--',(0, (3, 5, 1, 5, 1, 5)),':','-.']

kinds = list(cluster\_center.iloc[:, 0])

# 由于雷达图要保证数据闭合，因此再添加L列，并转换为 np.ndarray

cluster\_center = pd.concat([cluster\_center, cluster\_center[['ZL']]], axis=1)

centers = np.array(cluster\_center.iloc[:, 0:])

# 分割圆周长，并让其闭合

n = len(labels)

angle = np.linspace(0, 2 \* np.pi, n, endpoint=False)

angle = np.concatenate((angle, [angle[0]]))

# 绘图

fig = plt.figure(figsize = (8,6))

ax = fig.add\_subplot(111, polar=True)  # 以极坐标的形式绘制图形

plt.rcParams['font.sans-serif'] = ['SimHei']  # 用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False  # 用来正常显示负号

# 画线

for i in range(len(kinds)):

    ax.plot(angle, centers[i], linestyle=lstype[i], linewidth=2, label=kinds[i])

# 添加属性标签

ang = angle \* 180 / np.pi

ax.set\_thetagrids(ang[:-1], labels)

#ax.set\_thetagrids(angle \* 180 / np.pi, labels)

plt.title('客户特征分析雷达图')

plt.legend(legen)

plt.show()

plt.close

with open('data\_cleaned.csv','r',encoding='utf-8')as fp:

    reader=csv.reader(fp)

    t=0

    for rows in enumerate(reader):

        if kmeans\_labels[t]==0:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/labcode/一般客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==1:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/labcode/低价值客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==2:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/labcode/重要挽留客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==3:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/labcode/重要保持客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1

        elif kmeans\_labels[t]==4:

            row =rows

            # print(row)

            filename=row[1]

            path=r'C:/Users/Dell/Desktop/labcode/重要发展客户.csv'

            with open (path,'a',newline='',encoding='utf-8')as fp:

                    writer=csv.writer(fp)

                    writer.writerow(row)

            fp.close()

            t=t+1