航空公司客户价值分析

A. 背景

信息时代的来临使得企业营销焦点从产品中心转变为客户中心，客户关系管理成为企业的核心问题。客户关系管理的关键问题是客户分类，通过客户分类，区分无价值客户、高价值客户，企业针对不同价值的客户指定优化的个性化服务方案，采取不同营销策略，将有限的营销资源集中于高价值客户，实现企业利润最大化目标。准确的客户分类结果是企业优化营销资源分配的重要依据，客户分类越来越成为客户关系管理中亟待解决的关键问题之一。

B. 分析过程

本文的目标是客户价值识别，即通过航空公司客户说是识别不同价值的客户。识别客户价值应用最广泛的模型是通过三个指标（最近消费时间间隔（Recency）、消费频率（Frequency）和消费金额（Monetary））来进行客户细分，识别出高价值的客户，简称 RFM 模型。

在 RFM 模型中，消费金额表示在一段时间内，客户购买该企业产品金额的总和。由于航空票价受到运输距离、舱位等级等多种因素影响，同样消费金额的不同旅客对航空公司的价值是不同的。例如，一位购买长航线、低等级舱位票的旅客与一位购买短航线、高等级舱位票的旅客相比，后者对于航空公司而言价值可能更高。因此，这个指标并不适用于航空公司的客户价值分析。我们选择客户在一定时间内累计的飞行里程 M 和客户在一定时间内乘坐舱位所对应的折扣系数的平均值 C 两个指标代替消费金额。此外，考虑航空公司会员入会时间的长短在一定程度上能够影响客户价值，所以在模型中增加客户关系长度 L，作为区分客户的另一指标。

本文将客户管理长度 L、消费时间间隔 R、消费频率 F、飞行里程 M 和折扣系数的平均值 C 五个指标作为航空公司十年客户价值指标，记为 LRFMC 模型，如表B.1所示。

针对航空公司 LRFMC 模型，如果采用传统 RFM 模型分析的属性分箱方法，如图B.1所示（依据属性的平均值进行划分），虽然也能识别出最有价值的客户，但是细分的客户群太多，提高了针对性营销的成本。因

表 B.1. 指标含义

模型	航空公司 LRFMC 模型
L	会员入会时间距观测窗口结束的月数
R	客户最近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

此，本课题采用三种方法识别客户价值。通过对航空公司客户价值的 LRFMC 模型的五个指标进行 K-Means 聚类、层次聚类以及 DBSCAN 聚类，比较三种聚类算法的效果，应用于实际中，识别出最有价值的客户。

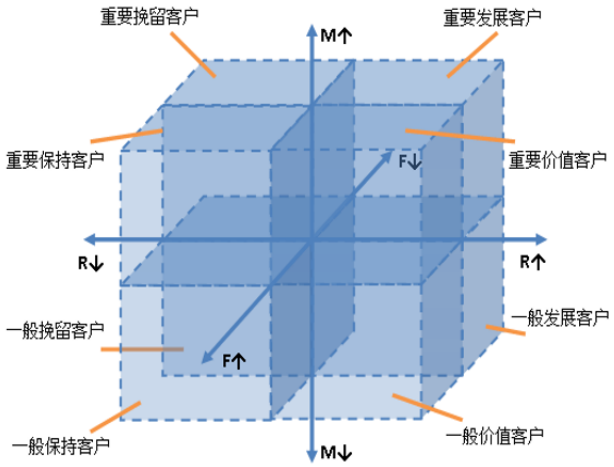

图 B.1. RFM 模型分析

航空货运信息挖掘主要包括以下步骤。

1）从航空公司的数据源中进行选择性抽取与新增数据同时分别形成历史数据和增量数据。

2）对步骤 1）中形成的两个数据集进行数据探索分析与预处理，包括数据缺失值与异常值的探索分析，数据的属性规约、清洗和变换。

3）利用步骤 2）中形成的已完成数据预处理的建模数据，基于旅客价值 LRFMC 模型进行客户分群，对各个客户群进行特征分析，识别出有价值的客户。

4）针对模型结果得到不同价值的客户，采用不同的营销手段，提供定制化的服务。

本文航空公司客户价值分析的总体流程如图B.2所示。

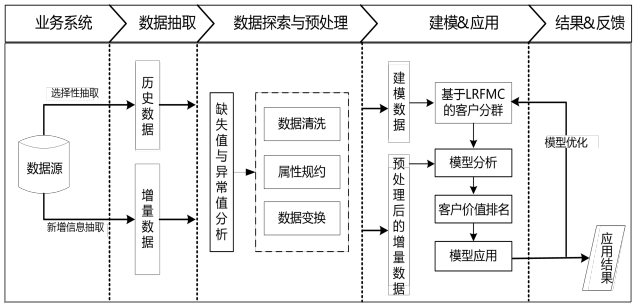

图 B.2. 航空客运数据挖掘建模总体流程

B.1. 数据抽取

从航空公司系统内的客户基本信息、乘机信息以及积分信息等详细数据中，根据末次飞行日期，抽取 2014-04-01 至 2014-03-31 内所有乘客的详细数据，总共有 62988 条记录。其中包含了会员卡号、入会时间、性别、年龄、会员卡级别、工作地城市、工作地所在省份、工作地所在国家、观测窗口结束时间、观测窗口乘机积分、飞行公里数、飞行次数、飞行时间、乘机时间间隔和平均折扣率等 44 个属性。

B.2. 数据探索分析

本文的探索分析是对数据进行缺失值分析与异常值分析，分析出数据的规律以及异常值。通过对数据观察发现原始数据中存在票价为空值，售价最小值为 0、折扣率最小值为 0、总飞行公里数大于 0 的记录。票价为空值的数据可能是客户不存在乘机记录造成，其他的数据可能是客户乘坐 0 折机票或者积分兑换产生的。

查找每列属性观测值中空值个数、最大值与最小值的 Python 代码如下所示。

数据探索

```
#对数据进行基本的探索
#返回缺失值个数以及最大最小值
import pandas as pd
#航空原始数据,第一行为属性标签
datafile='./data/air_data.csv'
#数据探索结果表
resultfile='./temp/explore.csv'
#读取原始数据，指定 UTF-8 编码（需要用文本编辑器将数据装换为 UTF-8 编码）
data=pd.read_csv(datafile,encoding='utf-8')
#包括对数据的基本描述，percentiles 参数是指定计算多少的分位数表（如 1/4 分位数、中位数等
#T 是转置，转置后更方便查阅
explore=data.describe(percentiles=[],include='all').T
#describe() 函数自动计算非空值数，需要手动计算空值数
explore['null']=len(data)-explore['count']
explore=explore[['null','max','min']]
#表头重命名
explore.columns=[u'空值数',u'最大值',u'最小值']
'''
这里只选取部分探索结果。
describe()函数自动计算的字段有count（非空值数）、unique（唯一值数）、top（频数最高者）、freq（最高频数）、mean（平均值）、std（方差）、min（最小值）、50%（中位数）、max（最大值）
'''
#导出结果
explore.to_csv(resultfile,encoding='utf-8-sig')
```

B.3. 数据预处理

本文主要采用数据清洗、属性规约与数据变换的预处理方法。

B.3.1 数据清洗

通过数据探索分析，发现数据中存在缺失值，票价最小值为 0、折扣率最小值为 0、总飞行公里数大于 0 的记录。由于原始数据量大，这类数据所占比例较小，对于问题影响不大，因此对其进行丢弃处理。具体处理方法如下。

1）丢弃票价为空的记录

2）丢弃票价为 0、平均折扣率不为 0、总飞行公里数大于 0 的记录

使用 Pandas 对满足清洗条件的数据进行丢弃，处理方法：满足清洗条件的一行数据全部丢弃，其 Python 代码如下。

数据清洗

```
import numpy as np
import pandas as pd
airline_data=pd.read_csv('./data/air_data.csv')
print('原始数据的形状为：',airline_data.shape)
#数据清洗结果表
cleanfile="./temp/data_cleaned.csv"
#去除票价为空的记录
exp1=airline_data["SUM_YR_1"].notnull()
exp2=airline_data["SUM_YR_2"].notnull()
exp=exp1&exp2
airline_notnull=airline_data.loc[exp,:]
print('删除缺失记录后数据的形状为：',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)
airline=airline_notnull[(index1|index2)&index3]
print('删除异常记录后数据的形状为：',airline.shape)
#导出结果
airline_notnull.to_csv(cleanfile,encoding='utf-8-sig')
airline_notnull.head(5)
```

运行结果：原始数据的性状为：(62988, 44)；删除缺失记录后数据的形状为：(62299, 44)；数据清洗后数据的形状为：(62044, 44)。

B.3.2 属性规约

原始数据中属性太多，根据航空公司客户价值 LRFMC 模型，选择与 LRFMC 指标相关的 6 个属性：FFP_DATE、LOAD_TIME、FLIGH_COUNT、AVG_DISCOUNT、SEG_KM_SUM、LAST_TO_END。删除与其不相关、弱相关或冗余的属性，例如，会员卡号、性别、工作地城市、工作地所在省份、工作地所在国家和年龄等属性。其 Python 代码如下。

	MEMBER_NO	FFP_DATE	FIRST_FLIGHT_DATE	GENDER	FFP_TIER	WORK_CITY	WO
0	54993	2006/11/02	2008/12/24	男	6	.	
1	28065	2007/02/19	2007/08/03	男	6	NaN	
2	55106	2007/02/01	2007/08/30	男	6	.	
3	21189	2008/08/22	2008/08/23	男	5	Los Angeles	
4	39546	2009/04/10	2009/04/15	男	6	贵阳	

5 rows × 44 columns

图 B.3. 数据清洗后的结果（前五行）

选取并构建 LRFMC 模型的特征

```
#选取需求特征
airline_selection=airline[["FFP_DATE","LOAD_TIME",
    "FLIGHT_COUNT","LAST_TO_END",
    "avg_discount","SEG_KM_SUM"]]
#构建 L 特征
L=pd.to_datetime(airline_selection["LOAD_TIME"])-\
    pd.to_datetime(airline_selection["FFP_DATE"])
L=L.astype("str").str.split().str[0]
L=L.astype("int")/30
#合并特征
airline_features=pd.concat([L,airline_selection.iloc[:,2:]],axis = 1)
print('构建的LRFMC特征前5行为：\n',airline_features.head())
```

运行结果如图B.4所示：

构建的LRFMC特征前5行为：

	0	FLIGHT_COUNT	LAST_TO_END	avg_discount	SEG_KM_SUM
0	90.200000	210	1	0.961639	580717
1	86.566667	140	7	1.252314	293678
2	87.166667	135	11	1.254676	283712
3	68.233333	23	97	1.090870	281336
4	60.533333	152	5	0.970658	309928

图 B.4. 构建的 LRFMC 特征前 5 行

B.3.3 数据变换

数据变换是将数据转换成“适当的”格式，以适应挖掘任务及算法的需要。本文中主要采用的数据变换方式为属性构造和数据标准化。

由于原始数据中并没有直接给出 LRFMC 五个指标，需要通过原始数据提取这五个指标，具体的计算方式如下。

(1)L = LOAD_TIME - FFP_DATE

会员人会时间距观测窗口结束的月数 = 观测窗口的结束时间–入会时间 [单位：月]

(2) R = LAST_TO_END

客户最近一次乘坐公司飞机距观测窗口结束的月数 = 最后一次乘机时间至观察窗口末端时长 [单位：月]

(3) F = FLIGHT_COUNT

客户在观测窗口内乘坐公司飞机的次数 = 观测窗口的飞行次数 [单位：次]

(4) M = SEG_KM_SUM

客户在观测时间内在公司累计的飞行里程 = 观测窗口的总飞行公里数 [单位：公里]

(5)C = AVG_DISCOUNT

客户在观测时间内乘坐舱位所对应的折扣系数的平均值 = 平均折扣率 [单位：无]

其 Python 代码如下：

标准化 LRFMC 模型的特征

```
from sklearn.preprocessing import
    StandardScaler
data = StandardScaler().fit_transform(
    airline_features)
np.savez('./temp/airline_scale.npz',
    data)
print('标准化后LRFMC五个特征为：\n',
    data[:5,:])
```

运行结果如B.5所示：为了消除数量级数据带来的影响，需要对数据进行标准化处理。标准差标准化处理的 Python 代码如下所示。

标准差标准化

```
import pandas as pd
import warnings
```

```
构建的LRFMC特征前5行为：
          0  FLIGHT_COUNT  LAST_TO_END  avg_discount  SEG_KM_SUM
0  90.200000           210            1      0.961639      580717
1  86.566667           140            7      1.252314      293678
2  87.166667           135           11      1.254676      283712
3  68.233333            23           97      1.090870      281336
4  60.533333           152            5      0.970658      309928
```

图 B.5. 运行结果

```
warnings.filterwarnings('ignore')
#需要进行标准化的数据文件
datafile = '../data/zscoredata.xls'
#标准差化后的数据存储路径文件;
zscoredfile = '../temp/zscoreddata.xls'
#标准化处理
data = pd.read_excel(datafile)
#简洁的语句实现了标准化变换，类似地可以实现任何想要的变换
data = (data - data.mean(axis=0))/(data
    .std(axis=0))
#表头重命名
data.columns = ['Z'+i for i in data.
    columns]
#数据写入
data.to_excel(zscoredfile, index=False)
```

标准差标准化处理后，形成 ZL、ZR、ZF、ZM、ZC 这 5 个属性的数据，如表B.2所示。

表 B.2. 标准化处理后的数据集

ZL	ZR	ZF	ZM	ZC
1.690	0.140	-0.636	0.069	-0.337
1.690	-0.322	0.852	0.844	-0.554
...	...	...	...	...
0.575	-0.212	-0.423	-0.187	-0.662
0.529	-0.377	0.144	0.617	0.042

B.4. 模型构建

客户价值分析模型构建主要有两个部分构成，第一部分根据航空公司客户 5 个指标的数据，对客户进行聚类分群。第二部分结合业务对每个客户群进行特征分析，分析其客户价值，并对每个客户群进行排名。

B.4.1 K-Means 聚类

采用 K-Means 聚类算法对客户数据进行客户分群，聚成 5 类（需要结合业务的理解与分析来确定客户的类别数量）。

K-Means 聚类算法位于 Scikit-Learn 库下的聚类子库（sklearn.cluster），其 Python 代码如下所示。

K-Means 聚类算法

```python
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
#读取标准化后的数据
airline_scale=np.load('./temp/
    airline_scale.npz')['arr_0']
#确定聚类中心数
k =5
#构建模型，随机种子设为 123
kmeans_model=KMeans(n_clusters=k,
    random_state=123)
#模型训练
fit_kmeans=kmeans_model.fit(
    airline_scale)
#查看聚类结果
kmeans_cc=kmeans_model.cluster_centers_
print('各类聚类中心为：\n',kmeans_cc)
kmeans_labels=kmeans_model.labels_
print('各样本的类别标签为：\n',
    kmeans_labels)
r1=pd.Series(kmeans_model.labels_).
    value_counts()
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)
```

运行结果如图B.6所示：

```
各类聚类中心为：
 [[ 1.16094184e+00 -8.66355853e-02 -3.77438378e-01 -1.56893014e-01
  -9.45420456e-02]
 [ 4.83551752e-01  2.48315495e+00 -7.99413281e-01  3.09787292e-01
   2.42425727e+00]
 [ 4.07521844e-02 -2.32407891e-01 -2.32993527e-03  2.16900461e+00
  -2.36767589e-01]
 [-3.13072314e-01 -5.73910449e-01  1.68707882e+00 -1.75466654e-01
  -5.36725346e-01]
 [-7.00318428e-01 -1.60626736e-01 -4.15128221e-01 -2.58203510e-01
  -1.60330905e-01]]
各样本的类别标签为：
 [1 1 1 ... 4 3 3]
最终每个类别的数目为：
 4    24611
0    15730
3    12111
1     5337
2     4255
dtype: int64
```

图 B.6. 运行结果

对数据进行聚类分群的结果如表B.3所示。值得注意的是，由于 K-Means 聚类是随机选择类标号，因此重复此实验得到结果中的类标号可能与此不同；另外，由于算法的精度问题，重复实验得到的聚类中心也可能略有不同。

表 B.3. 客户聚类结果

聚类类别	聚类中心				
	ZL	ZR	ZF	ZM	ZC
客户群 1	-0.312	1.692	-0.574	-0.537	-0.184
客户群 2	0.485	-0.800	2.483	2.423	0.314
客户群 3	-0.701	-0.417	-0.158	-0.157	-0.271
客户群 4	0.007	0.005	-0.251	-0.260	2.073
客户群 5	1.163	-0.378	-0.085	-0.093	-0.162

B.4.2 层次聚类

层次聚类（Hierarchical Clustering）是聚类算法的一种，通过计算不同类别数据点间的相似度来创建一棵有层次的嵌套聚类树。在聚类树中，不同类别的原始数据点是树的最低层，树的顶层是一个聚类的根节点。创建聚类树有自下而上合并和自上而下分裂两种方法。

层次聚类的原理是首先把每个对象作为一个簇，然后通过计算簇与簇的距离，合并距离最小的两个簇，也就是说每次结果的总簇数将减一，直到只有一个簇为止。

例如，假设你是一家公司的人力资源部经理，你可以把所有的雇员组织成较大的簇，如主管、经理和职员；然后你可以进一步划分为较小的簇，职员簇可以进一步划分为子簇；高级职员，一般职员和实习人员。所有的这些簇形成了层次结构，可以很容易地对各层次上的数据进行汇总或者特征化。

接下来使用层次聚类算法来看看本案例的聚类效果，其 Python 代码如下：

层次聚类算法

```
from sklearn.cluster import
    AgglomerativeClustering
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
pca=PCA(n_components=2)
newdata=pca.fit_transform(data)

hicl=AgglomerativeClustering(n_clusters
    =5)
hicl_pre=hicl.fit_predict(newdata
    [:500])

#可视化
plt.figure()
plt.scatter(newdata[:500][hicl_pre
    ==0,0],newdata[:500][hicl_pre==0,1],
    c="g",alpha=1,marker="s")
plt.scatter(newdata[:500][hicl_pre
    ==1,0],newdata[:500][hicl_pre==1,1],
    c="r",alpha=1,marker="*")
plt.scatter(newdata[:500][hicl_pre
    ==2,0],newdata[:500][hicl_pre==2,1],
    c="b",alpha=1,marker="d")
plt.scatter(newdata[:500][hicl_pre
    ==3,0],newdata[:500][hicl_pre==3,1],
    c="k",alpha=1,marker="^")
plt.scatter(newdata[:500][hicl_pre
    ==4,0],newdata[:500][hicl_pre==4,1],
    c="y",alpha=1,marker="o")
plt.rcParams['font.family'] = ['sans-
    serif']
plt.rcParams['font.sans-serif'] = ['
    SimHei']
plt.rcParams['axes.unicode_minus'] =
    False
plt.xlabel("主成分1")
plt.ylabel("主成分2")
plt.title("层次聚类")
plt.show()
```

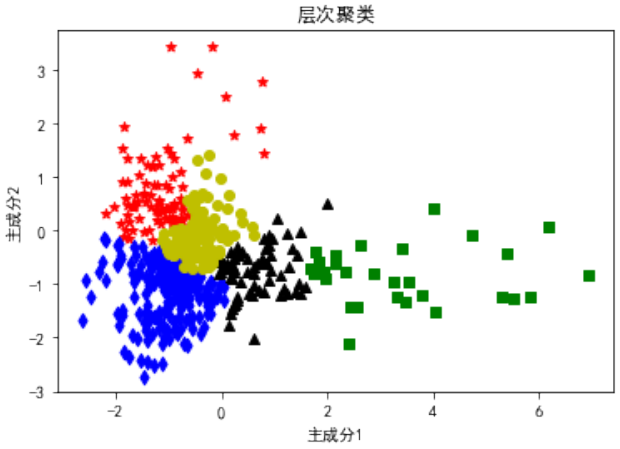

图 B.7. 层次聚类结果

使用层次聚类算法进行聚类的结果如图B.7所示，可以看出，图中绿色和红色的大部分区域的点较为离散，然后我们可以画出它的层次聚类树看看结果如何。画出层次聚类树的 Python 代码如下。

绘制层次聚类树

```
from scipy.cluster.hierarchy import
    dendrogram,linkage
z=linkage(newdata[:500],method='ward',
    metric='euclidean')
fig=plt.figure(figsize=(30,12))
irisdn=dendrogram(z)
plt.axhline(y=10,color='k',linestyle='
    solid',label='five class')
```

```
plt.axhline(y=20,color='g',linestyle='
    dashdot',label='four class')
plt.title("层次聚类树")
plt.xlabel("ID")
plt.ylabel("距离")
plt.legend(loc=1)
plt.show()
```

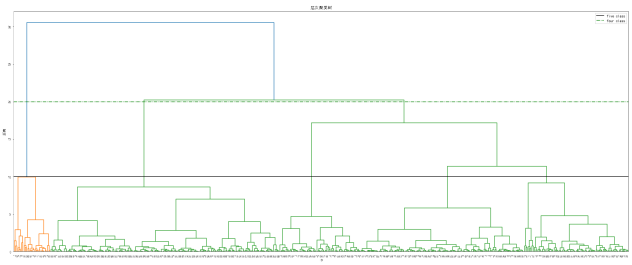

图 B.8. 层次聚类树

从图B.8中可以看出，若自顶向下分裂，在虚线处可以被分为四类，在实现处被分为了 5 类，它不用我们事先制定需要分成几类，而是根据程序自行的划分。而在实际应用中，层次聚类更适用于数据量小的情况，并不适用于本案例中数据量较大的情况。

B.4.3 DBSCAN 聚类

DBSCAN（Density-Based Spatial Clustering of Applications with Noise）算法，是一个具有代表性的基于密度的聚类算法，它的算法核心是，通过某个点 r 邻域内样本点的数量来衡量该点所在空间的密度。用通俗点的语言来解释就是，通过让中心点来发展“下线”的方法，来形成一个个簇。

DBSCAN 算法仅需要两个参数：领域半径 r 和最少点数目 MinPts。当邻域半径 R 内的点的个数大于最少点数目 MinPts 时，就是密集。所有的点都访问一次之后，形成的簇中，就有三种类型的点：核心点、边界点和噪声点。邻域半径 R 内样本点的数量大于等于 MinPts 的点叫做核心点。不属于核心点但在某个核心点的邻域内的点叫做边界点。既不是核心点也不是边界点的是噪声点。因此，DBSCAN 算法可以对任意形状的稠密数据集进行聚类，而 K-Means 之类的聚类算法一般只适用于凸数据集。同时可以在聚类的同时发现异常点，而对数据集中的异常点不敏感。

接下来看看 DBSCAN 算法对本案例的聚类效果，其 Python 代码如下。

DBSCAN 聚类算法

```
from sklearn.cluster import DBSCAN
dbscan=DBSCAN()
dbscan.fit(data)
print(dbscan.labels_)
for i in range(0,500):
    if dbscan.labels_[i]==-1:
        c1=plt.scatter(newdata[:500][i
            ,0],newdata[:500][i,1],c='r'
            ,marker="+")
    elif dbscan.labels_[i]==0:
        c2=plt.scatter(newdata[:500][i
            ,0],newdata[:500][i,1],c='g'
            ,marker="o")
    elif dbscan.labels_[i]==1:
        c3=plt.scatter(newdata[:500][i
            ,0],newdata[:500][i,1],c='b'
            ,marker="*")

plt.legend([c1,c2,c3],['类1','类2','类3
    '])
plt.title("使用DBSCAN聚类")
plt.show()
```

代码运行的结果如图B.9所示。

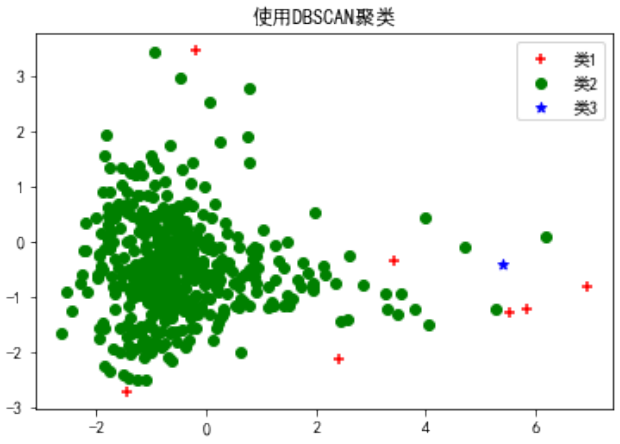

图 B.9. DBSCAN 算法聚类结果

可以看出，使用 DBSCAN 算法聚类的结果并不好，其实，从它的原理来思考，数据集的密度不均匀、聚类间距差相差很大时，聚类的质量一般较差。同时，调试参数比较复杂时，主要需要对距离阈值 Eps，邻域样本数阈值 MinPts 进行联合调参，不同的参数组合对最后的聚类效果有较大影响，因次，需要反复多次试验才有可能达到较为理想的效果。

B.4.4 客户价值分析

综上三种算法的聚类效果，最终选择 K-Means 算法进行客户价值分析，并绘制聚类雷达图，其 Python 代码如下。

```
import pandas as pd
#导入 K 均值聚类算法
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt
#待聚类的数据文件
inputfile = './temp/zscoreddata.xls'
#需要进行的聚类类别数
k = 5
#读取数据并进行聚类分析
data = pd.read_excel(inputfile)
#调用 k-means 算法，进行聚类分析
kmodel = KMeans(n_clusters=k)
#训练模型
kmodel.fit(data)
#标签
labels = data.columns
#添加多第一个标签
labels = labels.append(pd.Index([data.
    columns[0]]))
plot_data = kmodel.cluster_centers_
#指定颜色
color = ['b', 'g', 'r', 'c', 'y']
angles = np.linspace(0, 2*np.pi, k,
    endpoint=False)
plot_data = np.concatenate((plot_data,
    plot_data[:,[0]]), axis=1)
angles = np.concatenate((angles, [
    angles[0]]))
fig = plt.figure()
#polar 参数
ax = fig.add_subplot(111, polar=True)
#画线
for i in range(len(plot_data)):
    ax.plot(angles, plot_data[i], 'o-',
        color = color[i], label = u'客
        户群'+str(i), linewidth=2)
ax.set_rgrids(np.arange(0.01, 3.5, 0.5)
    , np.arange(-1, 2.5, 0.5),
    fontproperties="SimHei")
ax.set_thetagrids(angles * 180/np.pi,
    labels)
plt.legend(loc = 4)
plt.title('客户特征分析雷达图')
plt.show()
```

其运行结果，如图B.10所示。

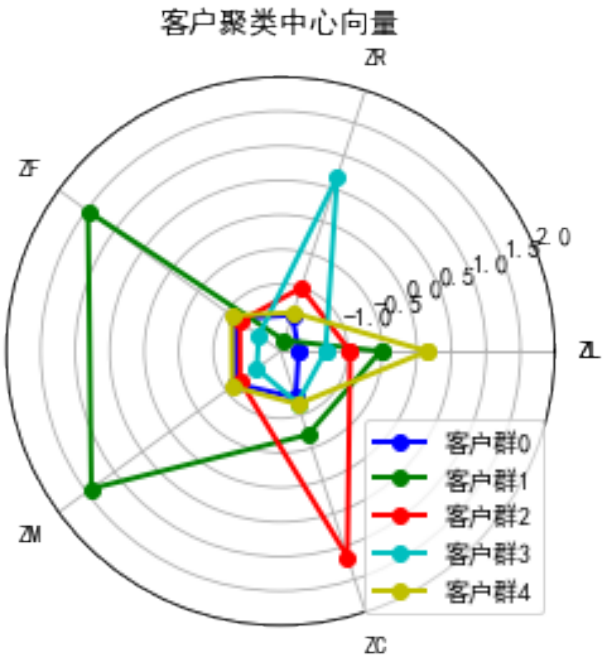

图 B.10. 客户群特征分析图

其中，客户群 0 在 L、C 属性上最小；客户群 1 在 F、M 属性上最大，在 R 属性上最小；客户群 2 在 C 属性上最大；客户群 3 在 R 属性上最大，在 F、M 属

性上最小；客户群 4 在 L 属性上最大。结合业务分析，通过比较各个指标在群间的大小对某一个群的特征进行评价分析。例如客户群 1 在 F、M 属性最大，在 R 指标最小，因此可以说 F、M、R 在客户群 1 是优势特征。以此类推，F、M、R 在客户群 3 上是劣势特征。从而总结出每个群的优势和弱势特征。

综上说明每个客户群的都有显著不同的表现特征，基于该特征描述，本案例定义五个等级的客户类别：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。他们之间的区别如图B.11所示，其中每种客户类别的特征如下：

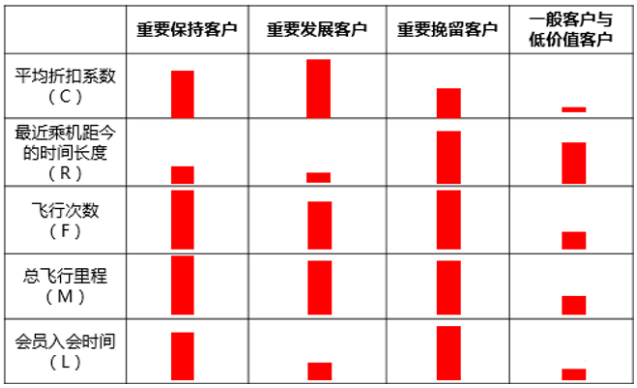

图 B.11. 客户类别的特征分析

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。

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。

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。

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。

其中，重要发展客户、重要保持客户、重要挽留客户这三类重要客户分别可以归入客户生命周期管理的发展期、稳定期、衰退期三个阶段。

根据每种客户类型的特征，对各类客户群进行客户价值排名，其结果如表B.4所示。针对不同类型的客户群提供不同的产品和服务，提升重要发展客户的价值、稳定和延长重要保持客户的高水平消费、防范重要挽留客户的流失并积极进行关系恢复。

表 B.4. 客户群价值排名

客户群	排名	排名含义
客户群 1	1	重要保持客户
客户群 2	2	重要发展客户
客户群 4	3	重要挽留客户
客户群 0	4	一般客户
客户群 3	5	低价值客户

C. 模型应用

根据对各个客户群进行特征分析，采取下面的一些营销手段和策略，为航空公司的价值客户群管理提供参考。

C.1. 会员的升级与保级

航空公司的会员可以分为白金卡会员、金卡会员、银卡会员、普通卡会员，其中非普通卡会员可以统称为航空公司的精英会员。虽然各个航空公司都有自己的特点和规定，但会员制的管理方法是大同小异的。成为

精英会员一般都是要求在一定时间内（如一年）积累一定的飞行里程或航段，达到这种要求后就会在有效期内（通常为两年）成为精英会员，并享受相应的高级别服务。有效期快结束时，根据相关评价方法确定客户是否有资格继续作为精英会员，然后对该客户进行相应地升级或降级。

C.2. 首次兑换

航空公司常旅客计划中最能够吸引客户的内容就是客户可以通过消费积累的里程来兑换免票或免费升舱等。各个航空公司都有一个首次兑换标准，也就是当客户的里程或航段积累到一定程度时才可以实现第一次兑换，这个标准会高于正常的里程兑换标准。但是很多公司的里程积累随着时间会进行一定地削减，例如有的公司会在年末对该年积累的里程进行折半处理。这样会导致许多不了解情况的会员白白损失自己好不容易积累的里程，甚至总是难以实现首次兑换。同样，这也会引起客户的不满或流失。可以采取的措施是从数据库中提取出接近但尚未达到首次兑换标准的会员，对他们进行提醒或促销，使他们通过消费达到标准。一旦实现了首次兑换，客户在本公司进行再次消费兑换就比在其他公司进行兑换要容易许多，在一定程度上等于提高了转移的成本。另外，在一些特殊的时间点（如里程折半的时间点）之前可以给客户一些提醒，这样可以增加客户的满意度。

C.3. 交叉销售

通过发行联名卡等与非航空类企业的合作，使客户在其他企业的消费过程中获得本公司的积分，增强与公司的联系，提高他们的忠诚度。例如，可以查看重要客户在非航空类合作伙伴处的里程积累情况，找出他们习惯的里程积累方式（是否经常在合作伙伴处消费、更喜欢消费哪些类型合作伙伴的产品），对他们进行相应促销。

客户识别期和发展期为客户关系打下基石，但是这两个时期带来的客户关系是短暂的、不稳定的。企业要获取长期的利润，必须具有稳定的、高质量的客户。保持客户对于企业是至关重要的，不仅因为争取一个新客户的成本远远高于维持老客户的成本，更重要的是客户流失会造成公司收益的直接损失。因此，在这一时期，航空公司应该努力维系客户关系，使之处于较高的水准，最大化生命周期内公司与客户的互动价值，并使这样的高水平尽可能延长。对于这一阶段的客户，主要应该通过提供优质的服务产品和提高服务水平来提高客户的满意度。通过对旅客数据库的数据挖掘、进行客户细分，可以获得重要保持客户的名单。这类客户一般所乘航班的平均折扣率（C）较高，最近乘坐过本公司航班（R 低）、乘坐的频率（F）或里程（M）也较高。他们是航空公司的价值客户，是最理想的客户类型，对航空公司的贡献最大，所占比例却比较小。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。