1. 数据分析背景

随着经济的不断发展，高铁、动车等铁路不断兴建，现代的人们出行方式也更加多元化。但这也无形之中给航空公司带来了较大的冲击。航空公司之间的竞争也十分的激烈。以民航为例。民航的竞争除了三大航空公司之间的竞争之外，还将加入新崛起的各类小型航空公司、民营航空公司，甚至国外航空巨头。航空产品生产过剩，产品同质化特征愈加明显，于是航空公司从价格、服务间的竞争逐渐转向对客户的竞争。

随着人们消费水平的提高，客户问题越来越受到关注，客户的流失对利润增长造成了非常大的负面影响。客户与航空公司的关系越长，航空公司的利润就越高。因此，对于企业来说，企业的营销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是客户分类，通过客户分类，区分客户价值，企业针对不同类型的客户进行个性化的服务方案，采取不同的营销策略，将有限的资源集中用于高价值的客户，实现企业利润最大化目标。那么。分析航空公司客户数据，对客户进行分类，提高客户流失率是当务之急。

基于此背景，通过分析某一航空公司的客户数据，建立合理的客户评计模型，对客户进行分群，分析不同的客户群的客户价值，并制定相应的营销策略。

2. 数据获取方法

数据说明：

只针对了所提供的的数据进行分析和建立模型。

数据获取代码：

```
import pandas as pd s_data = pd.read_csv('air_data.csv',header=0)  # header=0 默认数据第一行为列名 s_data.shape # 统计一个 dataframe 的信息 (62988,44)
```

3. 数据探索

3.1 探索目标与探索发现

3.1.1 探索数据属性

```
In [4]: import pandas as pd
        s_data = pd.read_csv('air_data.csv',header=0)
        s_data
```

Out[4]:

	MEMBER_NO	FFP_DATE	FIRST_FLIGHT_DATE	GENDER	FFP_TIER	WORK_CITY	WORK_PROVINCE	WORK_COUNTRY	AGE	LOAD_TIME	...	ADD_Po
0	54993	2006/11/2	2008/12/24	男	6	.	北京	CN	31.0	2014/3/31	...	
1	28065	2007/2/19	2007/8/3	男	6	NaN	北京	CN	42.0	2014/3/31	...	
2	55106	2007/2/1	2007/8/30	男	6	.	北京	CN	40.0	2014/3/31	...	
3	21189	2008/8/22	2008/8/23	男	5	Los Angeles	CA	US	64.0	2014/3/31	...	
4	39546	2009/4/10	2009/4/15	男	6	贵阳	贵州	CN	48.0	2014/3/31	...	
...	...	...	...	...	...	...	...	...	...	...	...	...
62983	18375	2011/5/20	2013/6/5	女	4	广州	广东	CN	25.0	2014/3/31	...	
62984	36041	2010/3/8	2013/9/14	男	4	佛山	广东	CN	38.0	2014/3/31	...	
62985	45690	2006/3/30	2006/12/2	女	4	广州	广东	CN	43.0	2014/3/31	...	
62986	61027	2013/2/6	2013/2/14	女	4	广州	广东	CN	36.0	2014/3/31	...	
62987	61340	2013/2/17	2013/2/17	女	4	上海	.	CN	29.0	2014/3/31	...	

62988 rows × 44 columns

图 1

3.1.2 head()探索

```
In [7]: import pandas as pd
        s_data = pd.read_csv('air_data.csv',header=0)
        s_data.head()
```

Out[7]:

	MEMBER_NO	FFP_DATE	FIRST_FLIGHT_DATE	GENDER	FFP_TIER	WORK_CITY	WORK_PROVINCE	WORK_COUNTRY	AGE	LOAD_TIME	...	ADD_Point_S
0	54993	2006/11/2	2008/12/24	男	6	.	北京	CN	31.0	2014/3/31	...	39
1	28065	2007/2/19	2007/8/3	男	6	NaN	北京	CN	42.0	2014/3/31	...	12
2	55106	2007/2/1	2007/8/30	男	6	.	北京	CN	40.0	2014/3/31	...	15
3	21189	2008/8/22	2008/8/23	男	5	Los Angeles	CA	US	64.0	2014/3/31	...	
4	39546	2009/4/10	2009/4/15	男	6	贵阳	贵州	CN	48.0	2014/3/31	...	22

5 rows × 44 columns

图 2

head()函数默认显示前 5 行，可以大致观察到表中的数据格式

3.1.3 info()探索

```
s_data.info()  # 输出 dataframe 的一些总体信息


'''  运行结果
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62988 entries, 0 to 62987
Data columns (total 44 columns):
```

```
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
0   MEMBER_NO               62988 non-null  int64
1   FFP_DATE                62988 non-null  object
2   FIRST_FLIGHT_DATE       62988 non-null  object
    3   GENDER                  62985 non-null  object
......
 42  Ration_L1Y_BPS          62988 non-null  float64
43  Point_NotFlight          62988 non-null  int64
dtypes: float64(12), int64(24), object(8) memory
usage: 21.1+ MB
```

3.1.4 describe()探索

```
s_data.describe()  # 显示数值（int or float）的基本统计学特征
```

```
In [8]: import pandas as pd
        s_data = pd.read_csv('air_data.csv',header=0)
        s_data.describe()
```

Out[8]:

	MEMBER_NO	FFP_TIER	AGE	FLIGHT_COUNT	BP_SUM	EP_SUM_YR_1	EP_SUM_YR_2	SUM_YR_1	SUM_YR_2	SEG_KM_SUM
count	62988.000000	62988.000000	62568.000000	62988.000000	62988.000000	62988.0	62988.000000	62437.000000	62850.000000	62988.000000
mean	31494.500000	4.102162	42.476346	11.839414	10925.081254	0.0	265.689623	5355.376064	5604.026014	17123.878691
std	18183.213715	0.373856	9.885915	14.049471	16339.486151	0.0	1645.702854	8109.450147	8703.364247	20960.844623
min	1.000000	4.000000	6.000000	2.000000	0.000000	0.0	0.000000	0.000000	0.000000	368.000000
25%	15747.750000	4.000000	35.000000	3.000000	2518.000000	0.0	0.000000	1003.000000	780.000000	4747.000000
50%	31494.500000	4.000000	41.000000	7.000000	5700.000000	0.0	0.000000	2800.000000	2773.000000	9994.000000
75%	47241.250000	4.000000	48.000000	15.000000	12831.000000	0.0	0.000000	6574.000000	6845.750000	21271.250000
max	62988.000000	6.000000	110.000000	213.000000	505308.000000	0.0	74460.000000	239560.000000	234188.000000	580717.000000

8 rows × 36 columns

图 3

其中各特征所对应的含义如下：

count：数量统计，此列共有多少有效值

mean：均值 unipue：不同的值有多少个

std：标准差 min：最小值 50%：二分之一分位数 max：最大值

通过 include 参数显式指定包含的数据类型，可以查看非数值特征的统计数据。

```
s_data.describe(include='object')
```

```
s_data.describe(include='object')
```

	FFP_DATE	FIRST_FLIGHT_DATE	GENDER	WORK_CITY	WORK_PROVINCE	WORK_COUNTRY	LOAD_TIME
count	62988	62988	62985	60719	59740	62962	62988
unique	3068	3406	2	3309	1183	118	1
top	2011/1/13	2013/2/16	男	广州	广东	CN	2014/3/31
freq	184	96	48134	9385	17507	57748	62988

图 4

3.1.5 value_counts()

探索目标：'GENDER' 探索代码：

```
gender=s_data['GENDER']  # 拿到数据列 print(gender.isnull().sum())  # 查看有多少空值 gender.value_counts(normalize=True) # 查看某列有多少种不同的值 normalize=True 表示
百分比的形式
```

运行结果：

```
    3
10  男    0.764214
    女    0.235786
    Name: GENDER, dtype: float64
```

图 5

男性占比 76.4% 女性占比 23.6%

4. 数据预处理

航空公司客户原始数据存在少量的缺失值和异常值，需要清洗后才能用于分析 。

- 通过对数据观察发现原始数据中存在票价为空值，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。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造成。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。

处理代码：

```
# 数据处理，除去票价为空 & 交集 当两个票价均为非空时，才算除去了空值数据
pro_data = s_data[s_data['SUM_YR_1'].notnull() & s_data['SUM_YR_2'].notnull()]
# 保证票价非0或者平均折扣非0且总里程>0  | 并集
con1 = pro_data['SUM_YR_1'] != 0   con2 = pro_data['SUM_YR_2'] != 0
con3 = (pro_data['avg_discount'] != 0) & (pro_data['SEG_KM_SUM'] >
0) pro_data = pro_data[con1 | con2 | con3]
pro_data # 处理之后的数据
```

运行结果：

	MEMBER_NO	FFP_DATE	FIRST_FLIGHT_DATE	GENDER	FFP_TIER	WORK_CITY	WORK_PROVINCE	WORK_COUNTRY
0	54993	2006/11/2	2008/12/24	男	6	.	北京	CN
1	28065	2007/2/19	2007/8/3	男	6	NaN	北京	CN
2	55106	2007/2/1	2007/8/30	男	6	.	北京	CN
3	21189	2008/8/22	2008/8/23	男	5	Los Angeles	CA	US
4	39546	2009/4/10	2009/4/15	男	6	贵阳	贵州	CN
...	...	...	...	...	...	...	...	...
62974	11163	2005/5/8	2005/8/26	男	4	NaN	NaN	CN
62975	30765	2008/11/16	2013/11/30	男	4	TAIPEI	NaN	TW
62976	10380	2010/7/8	2011/6/21	男	4	贵阳市	贵州省	CN
62977	16372	2012/12/20	2012/12/20	男	4	桃园	NaN	TW
62978	22761	2011/4/14	2011/4/14	男	4	汕头	广东省	CN

图6 处理之前：(62988,44) 处理之后：(62292,44)

5. 模型构建

5.1 模型介绍

一谈到用户分类模型，最常见的是RFM模型。RFM模型利用通用交易环节中最核心的三个维度--最近消费、消费频率、消费金额细分客户群体。从而分析不用群体的客户价值。但是该模型并不适用与所有的行业。对于航空行业来说，直接使用M（消费总额）并不能反映客户的真实价值，因为“长途低等舱”可能没有“短途高等舱”价值高，但是花费的金额却与短途的相差不大甚至更高。因此，在RFM模型的基础上，增加两个指标用于客户分群与价值分析。

其中 LRFMC 的含义分别如下图所示：

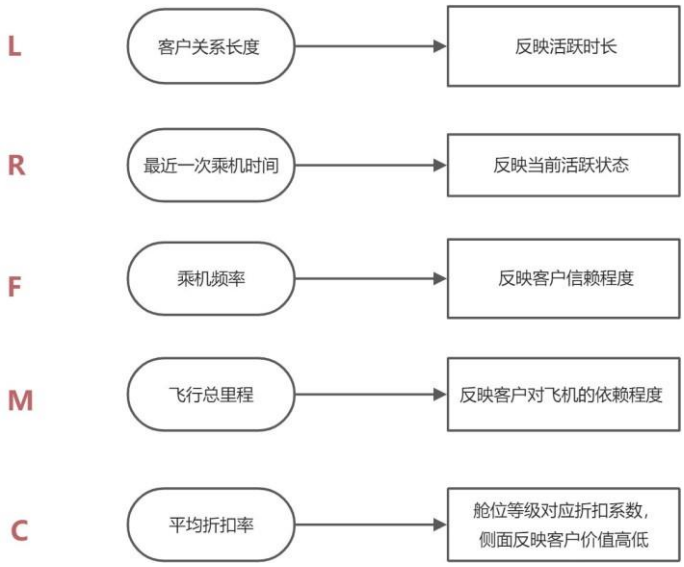

图 7

针对航空公司的 LRMFC 模型，若采用传统的 RFM 模型分析的话，如下图所示，虽然能够识别出最具价值的客户，但是细分出来的客户群太多，针对性营销成本会很大。因此，需要采用 K-Means 聚类的方法识别客户的额价值。

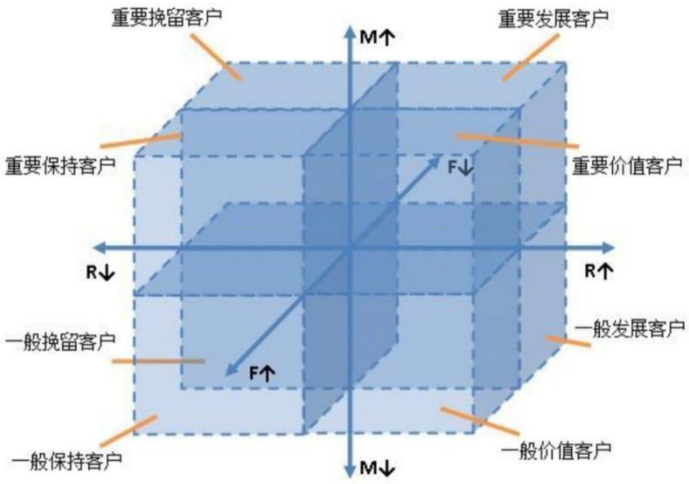

图 8

5.2 LRFMC 指标

原始数据中属性太多，根据 LPFMV 模型，选择与其指标相关的 6 个属性：'LOAD_TIME','FFP_DATE','LAST_TO_END','FLIGHT_COUNT','SEG_KM_SUM','avg_discount'。删除与其不相关、弱相关或冗余的属性。

```
# 重置索引
mode_data = pro_data.reset_index(drop=True)
```

```
# 只选择需要的列
mode_data = mode_data[
  ['LOAD_TIME','FFP_DATE','LAST_TO_END','FLIGHT_COUNT','SEG_KM_SUM','avg_discount']
         ]
mode_data.head()
```

	LOAD_TIME	FFP_DATE	LAST_TO_END	FLIGHT_COUNT	SEG_KM_SUM	avg_discount
0	2014/3/31	2006/11/2	1	210	580717	0.961639
1	2014/3/31	2007/2/19	7	140	293678	1.252314
2	2014/3/31	2007/2/1	11	135	283712	1.254676
3	2014/3/31	2008/8/22	97	23	281336	1.090870
4	2014/3/31	2009/4/10	5	152	309928	0.970658

图 9

LRFMC 中的指标需要自己构建。其中各项指标解释如下：

- L（客户关系长度）= LOAD_TIME - FFP_DATE

  会员入会时间距观测时间结束的月数=观测结束时间-入会时间 （单位：月）

- R = LAST_TO_END

  客户最后一次乘坐本公司飞机距观测时间结束的月数=最后一次乘机时间至观测时间结束的时长（单位：月）

- F = FLIGHT_COUNT

  客户在观测时间内乘坐本公司飞机的次数=观测时间内的飞行次数（单位：次）

- M = SEG_KM_SUM

  客户在穿刺时间内在本公司累计飞行里程=观测时间内的总飞行里数（单位：公里）

- 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）代码：

```
mode_data['LOAD_TIME'] = pd.to_datetime(mode_data['LOAD_TIME']) mode_data['FFP_DATE'] = pd.to_datetime(mode_data['FFP_DATE']) mode_data['TIME'] = mode_data['LOAD_TIME'] - mode_data['FFP_DATE'] # 得到时间差 mode_data['TIME'] = mode_data['TIME'].dt.days # 变成整数形式 final_data = mode_data[['TIME','LAST_TO_END','FLIGHT_COUNT','SEG_KM_SUM','avg_discount']] final_data.columns = ['L','R','F','M','C'] # 修改列名 final_data.head() # 显示前五列
```

运行结果截图：

	L	R	F	M	C
0	2706	1	210	580717	0.961639
1	2597	7	140	293678	1.252314
2	2615	11	135	283712	1.254676
3	2047	97	23	281336	1.090870
4	1816	5	152	309928	0.970658

图 10

5.3 标准化处理

利用 sklearn 进行标准化处理

```
# 5.标准化处理
from sklearn import preprocessing final_data.describe() data_scaled = preprocessing.scale(final_data) # 标准化处理 data_scaled.mean(axis=0) # 均值 data_scaled.std(axis=0) # 方差 data = pd.DataFrame(data_scaled) # 将 array 转变成 dataframe data.columns = ['L','R','F','M','C'] # 重命名列名 data.head()
```

5.4 K-Means 聚类

5.4.1 K-Means 简介

k-means :基于欧式距离的聚类算法,认为两个点距离越近，就归为一类算法步骤：

1. 选择初始化的 k 个样本作为初始聚类中心 $a = a_1, a_2, \ldots a_k$ ；
2. 针对数据集中每个样本 x_i 计算它到 k 个聚类中心的距离并将其分到距离最小的聚类中心所对应的类中；
3. 针对每个类别 a_j ，重新计算它的聚类中心 $a_j = \frac{1}{|c_i|}\sum_{x \in c_i} x$ （即属于该类的所有样本的质心）；
4. 重复上面 2 3 两步操作，直到达到某个中止条件（迭代次数、最小误差变化等）。

5.4.2 探索最佳的 K 值

```
# 聚类
from sklearn.cluster import KMeans import matplotlib.pyplot as
plt
```

```
 x = data option = []
for i in range(1,11):
    kmean = KMeans(n_clusters=i) # 分为 i 类 kmean.fit(x)
option.append(kmean.inertia_) # inertia 每个簇内到其质心的距离相加，叫 inertia
# 中文和负号的正常显示
plt.rcParams['font.sans-serif'] = [u'SimHei'] plt.rcParams['axes.unicode_minus']
= False
 plt.plot(range(1,11),option)
plt.tick_params(color='w',labelcolor='w')
plt.grid(True) plt.title('探索最佳的 k 值
',color='w')
plt.show()
```

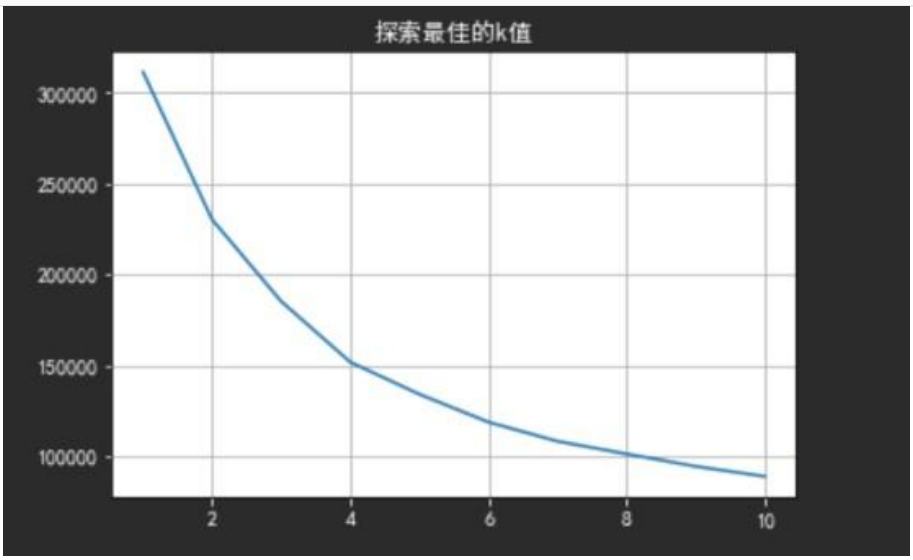

图 11

这里为了后续操作方便，利用手肘法，曲线趋于平缓大致在 5 处，所以选取聚类数为 5。

5.4.3 模型训练

```
from sklearn.cluster import KMeans kmeans = KMeans(n_clusters=5, random_state=666) #
此时 random=666 得到的 inertia_比
较小
x = data_kmeans.fit(x) # 实例化训练 r =
pd.concat([x,pd.Series(kmeans.labels_,index=x.index)],axis=1) # 主要是为了方便
后面可视化
r.columns = list(x.columns) + ['class'] # 将数据被判定的类别加入到 x_data 中
print(r.head()) # 显示前五个
df =
pd.DataFrame(kmeans.cluster_centers_,index=[0,1,2,3,4],columns=['L','R','F','M','C'
]) # cluster_centers_分类簇的均值向量,聚类中心 index=[0,1,2,3,4]不能改成中文，否则
后面 cnt 就不能正常显示了，需要 cnt 显示之后再修改行名
df['cnt'] = pd.Series(kmeans.labels_).value_counts() # 每个样本所属簇的标记，进行统计
df.index = ['客户群 1','客户群 2','客户群 3','客户群 4','客户群 5'] # 修改行名
df
```

	L	R	F	M	C	class
0	1.437348	-0.947618	14.054669	26.798874	1.296485	1
1	1.308775	-0.914518	9.087297	13.146445	2.868708	1
2	1.330007	-0.892451	8.732484	12.672433	2.881479	1
3	0.660011	-0.418022	0.784689	12.559423	1.995475	1
4	0.387530	-0.925551	9.938846	13.919344	1.345267	1

图 12

	L	R	F	M	C	聚类个数
客户群**1**	-0.314357	1.682992	-0.572820	-0.535733	-0.173725	12180
客户群**2**	0.484846	-0.801683	2.487390	2.428475	0.310922	5342
客户群**3**	0.049311	-0.003508	-0.230177	-0.234103	2.178057	4230
客户群**4**	-0.699413	-0.414472	-0.160683	-0.160515	-0.256791	24762
客户群**5**	1.162589	-0.376998	-0.086097	-0.093985	-0.156076	15778

图 13

6.数据分析 6.1

可视化聚类结果

```
for i in range(5):
    # r['class'] 是 x 数据对应被分好的类别
    # kind='kde' 密度图（核密度估计） layout 布局
    x[r['class'] == i].plot(kind='kde',linewidth=2, subplots=True,
sharex=False,layout=(1,x.shape[1]),figsize=(16,2))      plt.legend() plt.show()
# 每一行是一个类别
```

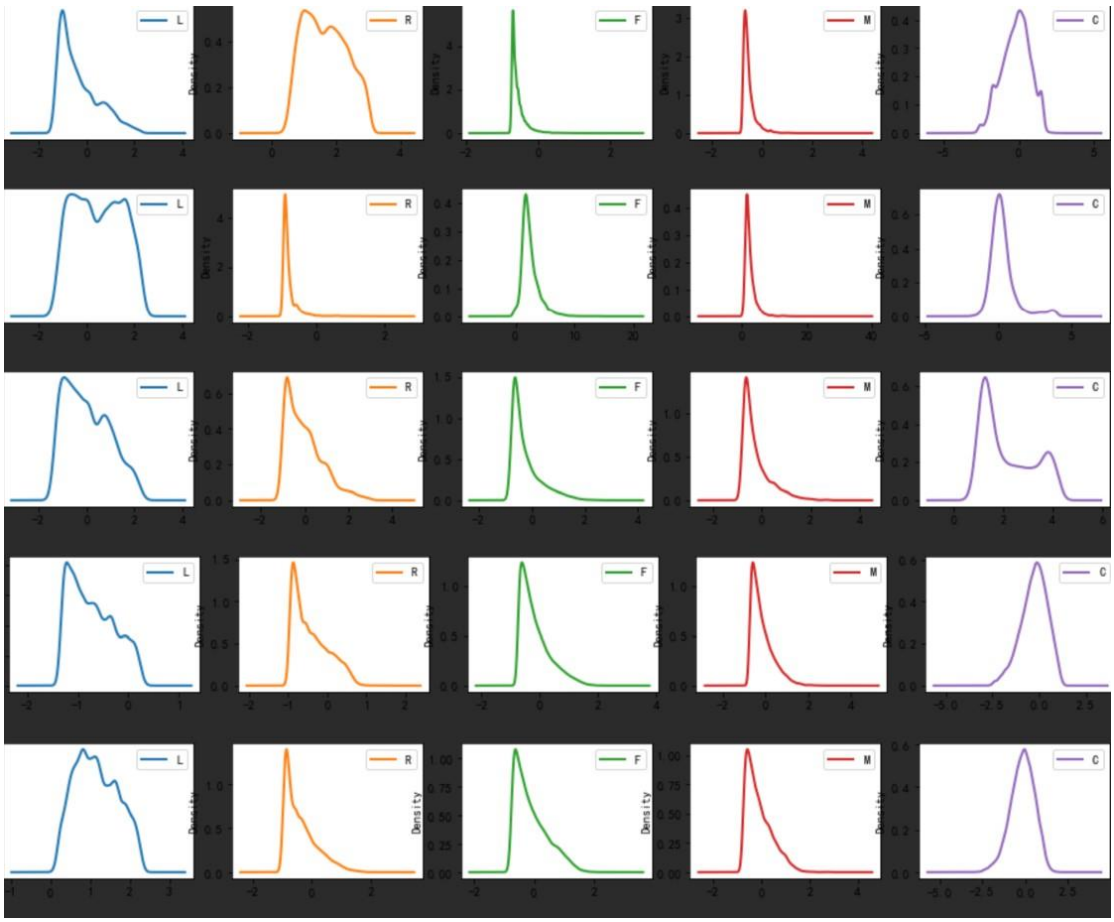

图 14

6.2 雷达图

```
# 绘制雷达图
fig = plt.figure(figsize=(10,8)) # 设置画布大小 ax
= fig.add_subplot(111,polar=True)
five = df.values # 聚类中心，最后一列是每一类的总个数
for i,v in enumerate(five):
    # 设置雷达图的角度
    angles = np.linspace(0, 2 * np.pi, 5, endpoint=False)     labels =
np.array(['L','R','F','M','C'])      center = np.concatenate((v[:-1],[v[0]]))   #
数据拼接最后一列和第一列，将雷达图
封闭
    angles = np.concatenate((angles,[angles[0]]))
labels = np.concatenate((labels,[labels[0]]))
    # 绘制折线图
    ax.plot(angles,center,'o--',linewidth=2,label="客户群%d : %d" % (i + 1, v[-1]))
     ax.fill(angles,center,alpha=0.3)      ax.set_thetagrids(angles * 180 /
np.pi,labels,color="w",fontsize=15)      ax.set_ylim(min_c - 0.2, max_c + 0.2)
plt.title("LRFCM 雷达展示",color="w",fontsize=20)      ax.grid(True)
plt.legend(loc='upper right',bbox_to_anchor=(1.3, 1.0), ncol=1, fancybox=True,
shadow=True) plt.show()
```

concatenate():使雷达图的数据是环形封闭的，第一个参数是原则，元组中的每个元素是一个数组。能够完成多个数组的拼接。

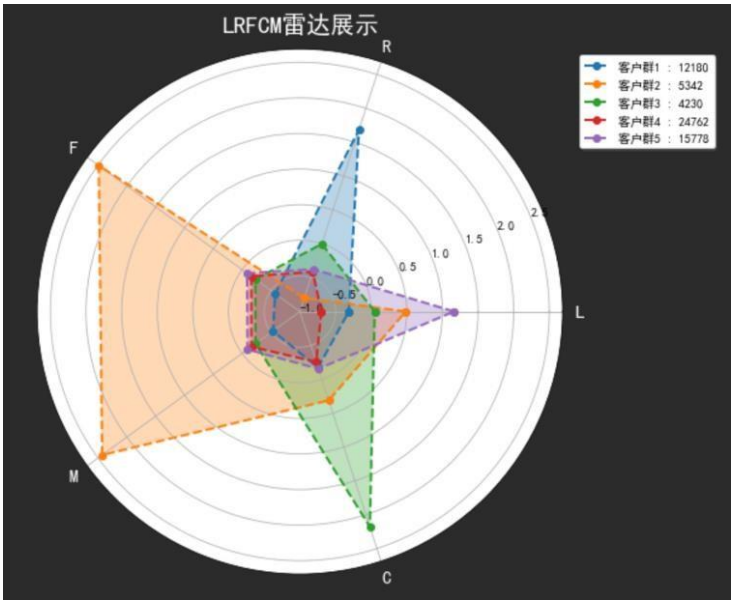

图 15

```
# 绘制饼图
t = df.iloc[:,5].values  # 客户群的数量 ex = [0.01,0.1,0.01,0.01,0.01] # 突出客户群 2 plt.pie(t,explode=ex,labels=df.index,autopct='%1.1f%%',
shadow=True,
startangle=90,colors=["orange","red","skyblue","pink","yellow"])
plt.tick_params(color='w',labelcolor='w') plt.show()
```

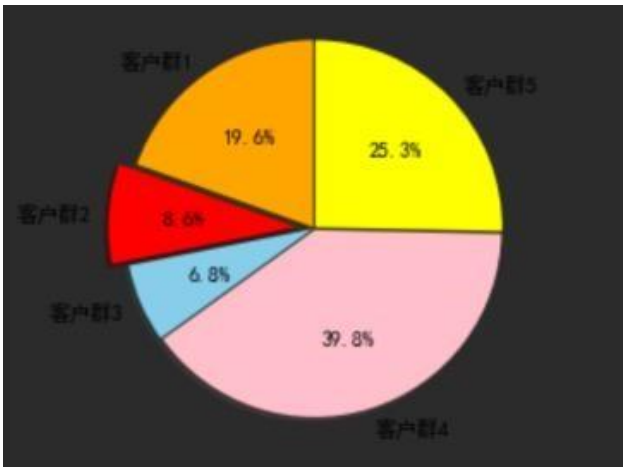

图 16

通过分析雷达图可知：

表 1

群类别	优势特征			弱势特征	
客户群 1	R			F	M
客户群 2	F	M	L	R	
客户群 3	C				
客户群 4				L	R
客户群 5	L			R	

根据上述特征分析说明每个客户群都有着显著不同的表现特征，基于该特征描述，这里将五个客户群依次定义为：重要保持客户、重要发展客户、挽留客户、低价值客户。

- 重要保持客户（客户群 2)：他们的 F（乘坐次数）以及 M（里程）远高于其他用户群体；但是他们的 R（最近乘坐本公司航班）低，他们对航空公司的贡献最大，所占比例比较小。但是他们的 C（平均折扣率）力度不够大。航空公司应该优先将资源投放到他们身上，对该客户群进行差异化的管理，提高客户的忠诚度和满意度，改善服务质量，尽可能延长客户的消费时间。
- 重要发展客户（客户群 3)：他们的 C（平均折扣率）远远高于其他 4 个用户群体，其他特征（R、F、M、L）均较为平衡，占比较低。这类用户可以认为是航空公司最具潜在价值的客户。虽然他们当前对公司利润做出的贡献不大，但却具有潜力。公司应该想办法增加这类客户群体的消费次数，加强对客户的沟通以及售后反馈，提高用户的满意度，使之成为公司的忠实客户。

- 重要挽留客户（客户群 1&5)：客户群 5 的 R（最近乘坐本公司航班）仅次于客户群 3（重要发展客户)，但是他们的 F（乘坐次数）以及 M（里程）比较低。即该类客户选择该公司的频率不高，具有较大的价值变动。而客户群 1 的 L 入会时间；由于客户衰退的原因各不相同，所以掌握客户的最新消息，维持与客户的互动就显得尤为重要。公司也应该根据客户的消费时间，消费次数的变化，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来延长客户的生命周期。

- 低价值客户（客户群 4）：该类客户五项指标均比较低。猜测这类客户一般是公司机票打着促销时才会乘坐。

其中重要发展客户、重要保持客户、重要挽留客户可归结为客户生命周期的发展期、稳定器和衰退期三个阶段。

因此各类用户的价值排名为：

表 2

客户群	排名	标签
客户群 2	1	重要保持客户
客户群 3	2	重要发展客户
客户群 1	3	重要挽留客户
客户群 5		
客户群 4	4	低价值客户

7. 分析总结

根据对各个客户群进行特征分析，采用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意见：

（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，并享受相应的高级服务。有效期快结束时，根据相关评价方法确定用户是否有资格继续作为精英会员，然后对该客户进行相应的升级或降级。

然而，由于许多客户并没有意识到活着根本不了解会员升级或保级的时间与要求，经常在评价期过后才发现自己其实只差一点就可以升级或保级，使之前的里程积累白白损失，由于这种损失给客户造成的不满导致客户直接放弃本公司的消费。

因此，航空公司可以对会员升级或保级评价的时间点前，对这些接近但未达到要求的较高消费者进行适当的提醒或采取一些促销活动，刺激他们通过消费达到相应的标准。这样既可以获得利益，也可以提高客户的满意度。

（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通过消费积累的里程来兑换免票或者升舱。各航空公司都有一个首次兑换的标准，这个标志会高于正常的标准。而且有的公司会在年末对积累里程进行处理，这种情况会导致不了解情况旅客的不满或者流失。可以采取对那些接近首次兑换标准的会员对他们进行提醒或促销，使他们达到标准。一旦实现首次兑换，他们转移的成本就大大提高了，从而增加客户黏度。

8. 参考资料

LRFCM 模型

航空客户价值分析