基于聚类算法完成航空公司客户价值分析

链接：<https://zhuanlan.zhihu.com/p/430231028>

**一、任务描述**

信息时代的来临使得企业营销焦点从产品中心转变成客户中心。具体地，对不同的客户进行分类管理，给予不同类型的客户制定优化的个性化服务方案，采取不同的营销策略。将有限的营销资源集中于高价值的客户，实现企业利润最大化。因此，如何对客户进行合理的分类成为了管理中亟需解决的关键问题之一。航空公司能够获取到客户的多种信息与行为数据，我们需要根据这些数据来实现以下目标：

1. 借助航空公司数据，对客户进行分类
2. 对不同类别的客户进行特征分析，比较不同类别客户的价值
3. 对不同价值的客户类别进行个性化服务，制定相应的营销策略

**1 数据集**

使用Kaggle数据竞赛平台提供的某航空公司的客户信息，其中数据中包括62988个客户样本，44种属性，具体的每种属性对应的含义如下图所示。

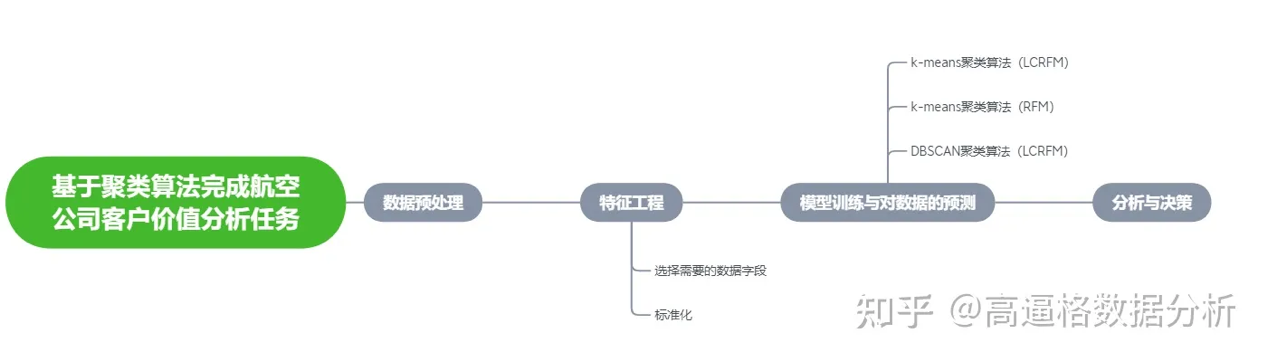


**2 库**

matplotlib、sklearn、pandas、numpy

**3 方法**

基于聚类算法完成航空客户分析任务主要流程有：数据预处理、模型训练与对数据的预测、分析与决策。详细步骤见下图



**二、 实战**

**1 数据预处理**

**1.1导入库**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import datetime

import sklearn.preprocessing

import sklearn.cluster

**1.2 读取数据**

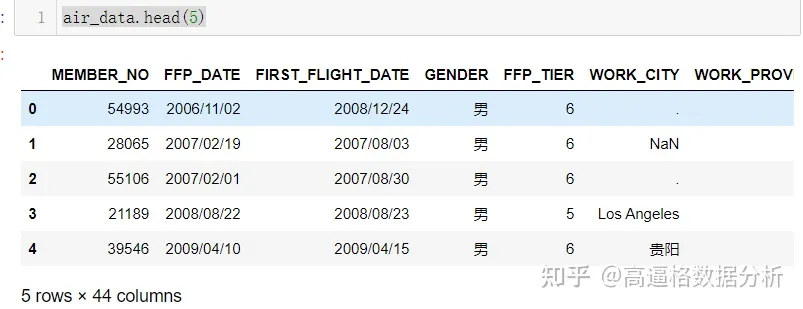
数据是以 CSV 文件的形式存储的，每行代表一个客户，每列代表一个属性字段。

air\_data\_path = './datasets/air\_data.csv'

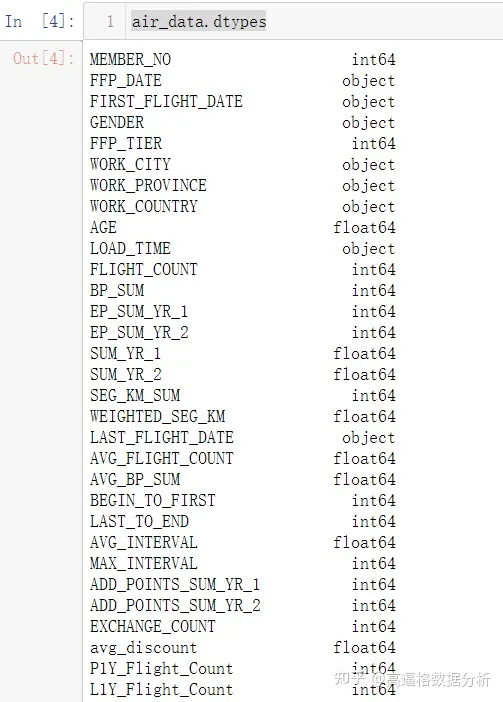
air\_data = pd.read\_csv(air\_data\_path)

print(air\_data.shape)

air\_data.head(5)

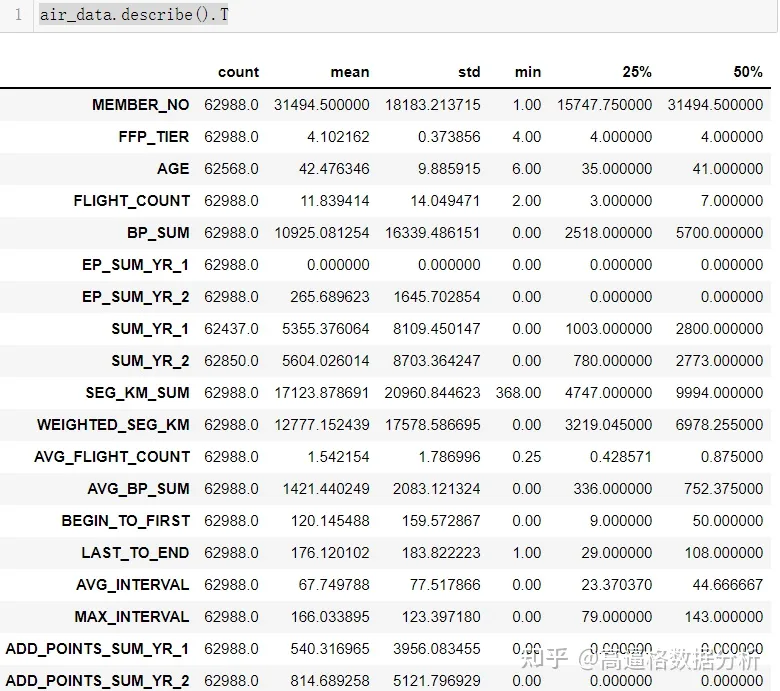


air\_data.dtypes



使用 pandas 中 DataFrame 的 describe() 函数表述数据的基本统计信息，对于数值型数据，输出结果指标包括count， mean，std，min，max以及第25百分位，中位数（第50百分位）和第75百分位

air\_data.describe().T



检查数据中是否有重复的会员 ID。

dup = air\_data[air\_data['MEMBER\_NO'].duplicated()]

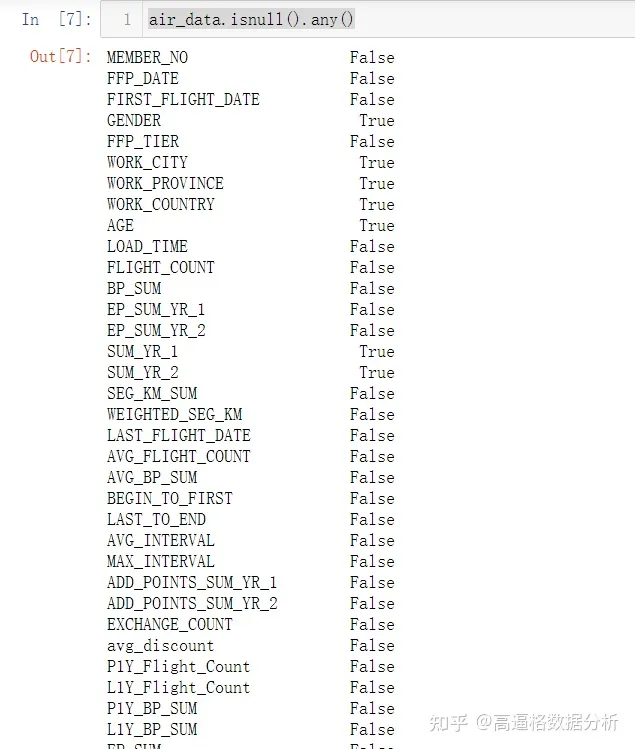
if len(dup) != 0:

print("There are duplication in the data:")

print(dup)

统计数据集中缺失值情况。有时数据集会有缺失的数据，这些缺失值在读入 pandas 的 DataFrame 里之后通常会表示为 None、NaN 等，如果忘记处理这些内容，可能会导致程序无法正常进行下去。isnull 函数返回布尔值 DataFrame ，代表数据中的每一个元素是不是 None、NaN 等。 any() 函数返回布尔值数组，每一个元素代表 DataFrame 的一列是否含有 True。

air\_data.isnull().any()



可以看到数据中的某些属性是有缺失值的，即 True 对应的几个属性。 其中 SUM\_YR\_i 是第 i 年的总消费，这些属性缺失或为 0 可能是因为数据采集出错或此客户不存在乘机记录。因此我们需要丢弃这些样本。

在 DataFrame 的一列数据上调用 notnull() 函数，返回布尔值数组，True 代表不为空， False 代表为空。

boolean\_filter = air\_data['SUM\_YR\_1'].notnull() & air\_data['SUM\_YR\_2'].notnull()

air\_data = air\_data[boolean\_filter]

filter\_1 = air\_data['SUM\_YR\_1'] != 0

filter\_2 = air\_data['SUM\_YR\_2'] != 0

air\_data = air\_data[filter\_1 | filter\_2]

print(air\_data.shape)

**2 特征工程**

**2.1 RFM模型**  
我们需要根据给定的44个属性来对客户价值进行价值分析，也就是对不同的客户进行分类，特别地，对于客户价值分析的一个经典模型是 RFM 模型。

* Recency: 最近消费时间间隔。
* Frequency: 客户消费频率。
* Monetary Value: 客户总消费金额。

使用 RFM 模型主要根据其中的RFM三个特征来对用户进行分析，将客户群体细分为重要保持客户、重要发展客户、重要挽留客户、一般客户、低价值客户五类。  
**2.2 变体 - LRFMC 模型**  
考虑到商用航空行业与一般商业形态的不同，国内外航空公司在RFM模型的基础上，还加上了 L 客户关系时长以及 C 客户所享受的平均折扣率这两个特征用于客户分群与价值分析，得到航空行业的LRFMC模型：

* Length of Relationship: 客户关系时长，反映可能的活跃时长。
* Recency: 最近消费时间间隔，反映当前的活跃状态。
* Frequency: 客户消费频率，反映客户的忠诚度。
* Mileage: 客户总飞行里程，反映客户对乘机的依赖性。
* Coefficient of Discount: 客户所享受的平均折扣率，侧面反映客户价值高低。

LRFMC 对应到数据集的字段：

* L = LOAD\_TIME - FFP\_DATE
* R = LAST\_TO\_END
* F = FLIGHT\_COUNT
* M = SEG\_KM\_SUM
* C = avg\_discount

增加一个表示关系长度（L）的属性（列）：L = LOAD\_TIME - FFP\_DATE

load\_time = datetime.datetime.strptime('2014/03/31', '%Y/%m/%d')

ffp\_dates = [datetime.datetime.strptime(ffp\_date, '%Y/%m/%d') for ffp\_date in air\_data['FFP\_DATE']]

length\_of\_relationship = [(load\_time - ffp\_date).days for ffp\_date in ffp\_dates]

air\_data['LEN\_REL'] = length\_of\_relationship

#移除我们不关心的属性（列），即只保留 LRFMC 模型需要的属性。

features = ['LEN\_REL','FLIGHT\_COUNT','avg\_discount','SEG\_KM\_SUM','LAST\_TO\_END']

data = air\_data[features]

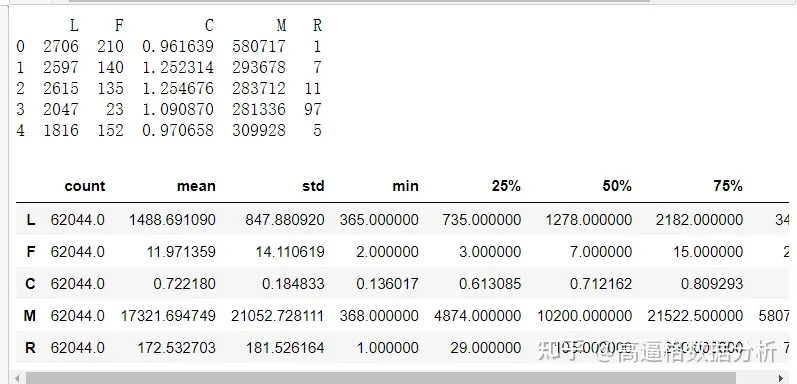
features = ['L','F','C','M','R']

data.columns = features

#预览前 5 行数据，并查看数据的元数据

print(data.head(5))

data.describe().T



我们可以看到，不同属性的取值范围差异很大，例如 ∈[365,3437]L∈[365,3437]、 ∈[0.136017,1.5]C∈[0.136017,1.5]。 这种情况会导致模型在学习的时候可能会对不同属性有着错误的重要性衡量。 因此我们要对这个情况进行处理，让不同属性的取值范围一致，即数据的标准化。 标准化方法有极大极小标准化、标准差标准化等方法，此处采用标准差标准化的方法对数据进行处理。\

**3 标准化**

对特征进行标准化，使得各特征的均值为 0、方差为 1。下一个代码块等同于以下语句

data = (data - data.mean(axis=0))/(data.std(axis=0))

ss = sklearn.preprocessing.StandardScaler(with\_mean=True, with\_std=True)#标准化

data = ss.fit\_transform(data) #数据转换

data = pd.DataFrame(data, columns=features)

data\_db = data.copy()

#描述标准化后的数据的元数据

data.describe().T



**三、 模型训练与预测**

我们想要将将客户群体细分为重要保持客户、重要发展客户、重要挽留客户、一般客户、低价值客户五类，于是我们可以用 k-means 聚类算法进行聚类，类别的数量可以人为控制。 前面我们已经通过数据处理得到了我们 LRFMC 模型需要的特征，接下来就使用 k-means 聚类算法来分析数据。这里我们使用机器学习库 scikit-learn 中现有的 kmeans 函数的实现。

**1 K-means 聚类算法**

目标是把 n 个观测样本划分成 k 个群体（cluster），每个群体都有一个中心（mean）。每个样本仅属于其中一个群体，即与这个样本距离最近的中心的群体。 符号: Si 是一个群体, mi 是群体 Si 里的样本的中心, xi 是一个样本点。

Assignment step (expectation step): 把每个样本分配给距离最近的中心的群体

Update step (maximization step): 根据当前的样本及其所属群体，重新计算各群体的中心

num\_clusters = 5 #设置类别为5

km = sklearn.cluster.KMeans(n\_clusters=num\_clusters, n\_jobs=4) #模型加载

km.fit(data) #模型训练

查看模型学习出来的 5 个群体的中心，以及 5 个群体所包含的样本个数

r1 = pd.Series(km.labels\_).value\_counts()

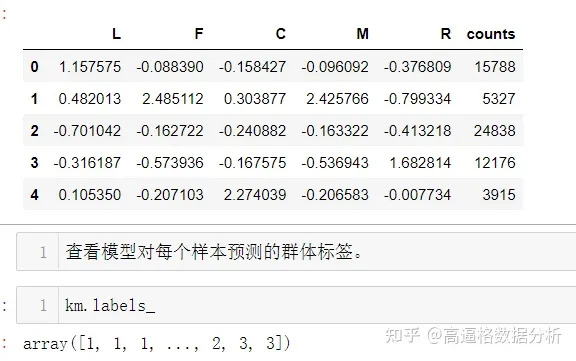
r2 = pd.DataFrame(km.cluster\_centers\_)

r = pd.concat([r2, r1], axis=1)

r.columns = list(data.columns) + ['counts']

r

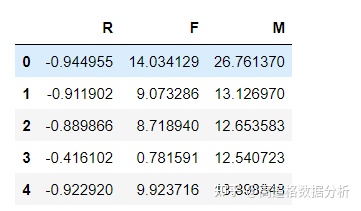
km.labels\_



**2 尝试使用RFM模型**

data\_rfm = data[['R','F','M']]

data\_rfm.head(5)



km.fit(data\_rfm) #模型对只包含rfm数据集训练

km.labels\_

r1 = pd.Series(km.labels\_).value\_counts()

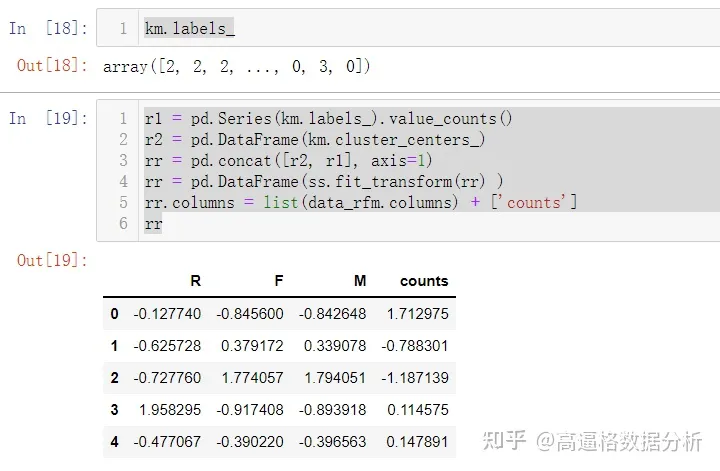
r2 = pd.DataFrame(km.cluster\_centers\_)

rr = pd.concat([r2, r1], axis=1)

rr = pd.DataFrame(ss.fit\_transform(rr) )

rr.columns = list(data\_rfm.columns) + ['counts']

rr



**3 分析与决策**

利用雷达图对模型学习出的 5 个群体的特征进行可视化分析。

import numpy as np

import matplotlib.pyplot as plt

from matplotlib.patches import Circle, RegularPolygon

from matplotlib.path import Path

from matplotlib.projections.polar import PolarAxes

from matplotlib.projections import register\_projection

from matplotlib.spines import Spine

from matplotlib.transforms import Affine2D

def radar\_factory(num\_vars, frame='circle'):

# 计算得到evenly-spaced axis angles

theta = np.linspace(0, 2\*np.pi, num\_vars, endpoint=False)

class RadarAxes(PolarAxes):

name = 'radar'

# 使用1条线段连接指定点

RESOLUTION = 1

def \_\_init\_\_(self, \*args, \*\*kwargs):

super().\_\_init\_\_(\*args, \*\*kwargs)

# 旋转绘图，使第一个轴位于顶部

self.set\_theta\_zero\_location('N')

def fill(self, \*args, closed=True, \*\*kwargs):

"""覆盖填充，以便默认情况下关闭该行"""

return super().fill(closed=closed, \*args, \*\*kwargs)

def plot(self, \*args, \*\*kwargs):

"""覆盖填充，以便默认情况下关闭该行"""

lines = super().plot(\*args, \*\*kwargs)

for line in lines:

self.\_close\_line(line)

def \_close\_line(self, line):

x, y = line.get\_data()

# FIXME: x[0], y[0] 处的标记加倍

if x[0] != x[-1]:

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

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

line.set\_data(x, y)

def set\_varlabels(self, labels):

self.set\_thetagrids(np.degrees(theta), labels)

def \_gen\_axes\_patch(self):

# 轴必须以（0.5，0.5）为中心并且半径为0.5

# 在轴坐标中。

if frame == 'circle':

return Circle((0.5, 0.5), 0.5)

elif frame == 'polygon':

return RegularPolygon((0.5, 0.5), num\_vars,

radius=.5, edgecolor="k")

else:

raise ValueError("unknown value for 'frame': %s" % frame)

def \_gen\_axes\_spines(self):

if frame == 'circle':

return super().\_gen\_axes\_spines()

elif frame == 'polygon':

# spine\_type 必须是'left'/'right'/'top'/'bottom'/'circle'.

spine = Spine(axes=self,

spine\_type='circle',

path=Path.unit\_regular\_polygon(num\_vars))

# unit\_regular\_polygon 给出以1为中心的半径为1的多边形

#（0，0），但我们希望以（0.5，

# 0.5）的坐标轴。

spine.set\_transform(Affine2D().scale(.5).translate(.5, .5)

+ self.transAxes)

return {'polar': spine}

else:

raise ValueError("unknown value for 'frame': %s" % frame)

register\_projection(RadarAxes)

return theta

**4 LCRFM模型作图**

N = num\_clusters

theta = radar\_factory(N, frame='polygon')

data = r.to\_numpy()

fig, ax = plt.subplots(figsize=(5, 5), nrows=1, ncols=1,

subplot\_kw=dict(projection='radar'))

fig.subplots\_adjust(wspace=0.25, hspace=0.20, top=0.85, bottom=0.05)

# 去掉最后一列

case\_data = data[:, :-1]

# 设置纵坐标不可见

ax.get\_yaxis().set\_visible(False)

# 图片标题

title = "Radar Chart for Different Means"

ax.set\_title(title, weight='bold', size='medium', position=(0.5, 1.1),

horizontalalignment='center', verticalalignment='center')

for d in case\_data:

# 画边

ax.plot(theta, d)

# 填充颜色

ax.fill(theta, d, alpha=0.05)

# 设置纵坐标名称

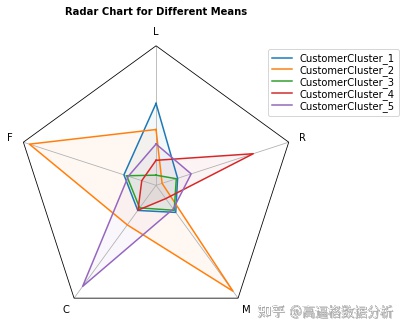
ax.set\_varlabels(features)

# 添加图例

labels = ["CustomerCluster\_" + str(i) for i in range(1,6)]

legend = ax.legend(labels, loc=(0.9, .75), labelspacing=0.1)

plt.show()



**5 RFM模型作图**

theta = radar\_factory(3, frame='polygon')

data = rr.to\_numpy()

fig, ax = plt.subplots(figsize=(5, 5), nrows=1, ncols=1,

subplot\_kw=dict(projection='radar'))

fig.subplots\_adjust(wspace=0.25, hspace=0.20, top=0.85, bottom=0.05)

# 去掉最后一列

case\_data = data[:, :-1]

# 设置纵坐标不可见

ax.get\_yaxis().set\_visible(False)

# 图片标题

title = "Radar Chart for Different Means"

ax.set\_title(title, weight='bold', size='medium', position=(0.5, 1.1),

horizontalalignment='center', verticalalignment='center')

for d in case\_data:

# 画边

ax.plot(theta, d)

# 填充颜色

ax.fill(theta, d, alpha=0.05)

# 设置纵坐标名称

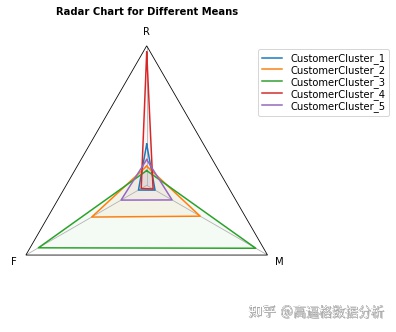
ax.set\_varlabels(['R','F','M'])

# 添加图例

labels = ["CustomerCluster\_" + str(i) for i in range(1,6)]

legend = ax.legend(labels, loc=(0.9, .75), labelspacing=0.1)

plt.show()



我们可以看到当使用RFM对用户进行划分时，我们可以进行考虑的参数（纬度）较少，只有RFM三个纬度，不能很好地对客户进行全方面的分析.

**6 DBSCAN模型对LCRFM特征进行计算**

除了Kmeans聚类算法外，我们还可以使用DBSCAN等聚类算法进行建模。

from sklearn.cluster import DBSCAN

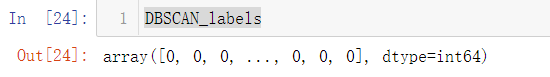
# db = DBSCAN(eps=10,min\_samples=2).fit(data\_db)

# Kagging debug

db = DBSCAN(eps=10,min\_samples=2).fit(data\_db.sample(10000))

DBSCAN\_labels = db.labels\_

DBSCAN\_labels



**7 根据LCRFM结果进行分析**

我们这里是对Kmeans算法使用LCRFM模型进行分析： 对应实际业务对聚类结果进行分值离散转化，对应1-5分，其中属性值越大，分数越高：



* （1）重要保持客户 平均折扣率高(C↑)，最近有乘机记录(R↓)，乘机次数高(F↑)或里程高(M↑)：

这类客户机票票价高，不在意机票折扣，经常乘机，是最理想的客户类型。 公司应优先将资源投放到他们身上，维持这类客户的忠诚度。

* （2）重要发展客户 平均折扣率高(C↑)，最近有乘机记录(R↓)，乘机次数低(F↓)或里程低(M↓)：

这类客户机票票价高，不在意机票折扣，最近有乘机记录，但总里程低，具有很大的发展潜力。 公司应加强这类客户的满意度，使他们逐渐成为忠诚客户。

* （3）重要挽留客户 平均折扣率高(C↑)，乘机次数高(F↑)或里程高(M↑)，最近无乘机记录(R↑)：

这类客户总里程高，但较长时间没有乘机，可能处于流失状态。 公司应加强与这类客户的互动，召回用户，延长客户的生命周期。

* （4）一般客户 平均折扣率低(C↓)，最近无乘机记录(R↑)，乘机次数高(F↓)或里程高(M↓)，入会时间短(L↓)：

这类客户机票票价低，经常买折扣机票，最近无乘机记录，可能是趁着折扣而选择购买，对品牌无忠诚度。 公司需要在资源支持的情况下强化对这类客户的联系。

* （5）低价值客户 平均折扣率低（C↓），最近无乘机记录（R↑），乘机次数高（F↓）或里程高（M↓），入会时间短（L↓）：

这类客户与一般客户类似，机票票价低，经常买折扣机票，最近无乘机记录，可能是趁着折扣而选择购买，对品牌无忠诚度。

**8 结果分析**

* 群体1的L属性最大
* 群体2的L、C属性最小
* 群体3的C属性上最大
* 群体4的M、F属性属性最大，R属性最小
* 群体5的R属性最大，F、M属性最小

其中每项指标的实际业务意义为：

* L：加入会员的时长。越大代表会员资历越久
* R：最近一次乘机时间。越大代表越久没乘机
* F：乘机次数。越大代表乘机次数越多
* M：飞行总里程。越大代表总里程越多
* C：平均折扣率。越大代表折扣越弱，0表示0折免费机票，10代表无折机票

重要保持客户：客户群4

重要发展客户：客户群3

重要挽留客户：客户群1

一般客户：客户群2

低价值客户：客户群5

**9 决策**

重要发展客户、重要保持客户、重要挽留客户这三类客户其实也对应着客户生命周期中的发展期、稳定器、衰退期三个时期。从客户生命周期的角度讲，也应重点投入资源召回衰退期的客户。  
一般而言，数据分析最终的目的是针对分析结果提出并开展一系列的运营/营销策略，以期帮助企业发展。在本实例中，运营策略有三个方向：  
提高活跃度：提高一般客户、低价值客户的活跃度。将其转化为优质客户  
提高留存率：与重要挽留客户互动，提高这部分用户的留存率  
提高付费率：维系重要保持客户、重要发展客户的忠诚度，保持企业良好收入  
每个方向对应不同的策略，如会员升级、积分兑换、交叉销售、发放折扣券等手段，此处不再展开。

**10优化思路**

数据预处理：对缺失值、异常值进行预处理 特征工程：RFM模型、LRFMC模型、标准

聚类算法模型：K-means、DBSCAN算法