来源：数据分析不是个事儿

文 | 李启方

用户精细化分类也可以称做用户画像，是目前很常见的一种运营手段，目的是为了更好的服务不同性质的客户，提高每个环节的转化率，最大程度挖掘客户价值，创造利润。

那么如何构建用户画像，或者说构建精细化的运营体系，这个过程的数据工作其实就是：

- 画像相关数据的整理和集中
- 找到同业务场景强相关数据
- 对数据进行分类和标签化（定量 to 定性）
- 依据业务需求引入外部数据
- 按照业务需求进行筛选客户（DMP 的作用）

本次案例就来分享如何通过数据挖掘的手段对用户进行精细化分类，以保险行业为例。

一、客户细分

客户细分根据客户的分类维度进行细分，保险行业的分类的维度一般包括 5 类，分别是社会特征因素、自然属性因素、行为特征因素、态度偏好因素和生活状态与个性因素。

前三者属于事前分类维度，是表露在外的因素，即通过接触就可知道的因素；后两者是属于事后分类维度，需要通过调研才能了解，反应客户内在本质的区别。对客户细分，我们往往通过事后分类维度做客户分类，以保证分类的深入性，再通过事前分类维度进行描述与验证，以保证分类客户的差异性和可接触性。这里用 Python 读取调研的 Excel 数据，将事后分类维度取出来并查看类型，发现 9 个维度都是数字类型并且部分维度之间似乎存在一定的相关性，这种相关性可能会造成重叠信息的扩大化，增加分类偏差，因此先对这 9 个维度进行因子分析。

```
import pandas as pd
import prince  # 对应分析
from scipy.cluster import hierarchy  # 层次聚类
from statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm  # 方差分析
import matplotlib.pyplot as plt
from factor_analyzer import FactorAnalyzer, calculate_kmo, calculate_bartlett_sphericity  # 因子分析
%matplotlib inline

# 读取数据
X = pd.read_csv('./Desktop/第6章保险公司客户分类数据.csv', engine='python',
                encoding='utf-8-sig')            # 注意设置encoding参数，不然中文字体会乱码
x = X.iloc[:, -9:]
x.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 712 entries, 0 to 711
Data columns (total 9 columns):
对自己的生活很满意         712 non-null int64
为享受而产生的浪费是必要的    712 non-null int64
买房子前要先有车          712 non-null int64
不惜金钱和时间装修房子       712 non-null int64
买衣服都买便宜的          712 non-null int64
休息时经常进行户外活动       712 non-null int64
尝试生活充满变化          712 non-null int64
喜欢独自享受安静的生活       712 non-null int64
下班后尽快回家           712 non-null int64
dtypes: int64(9)
memory usage: 50.1 KB
```

1.1 因子分析

因子分析是将多个实测变量转换为少数几个综合指标（或称潜变量），它反映一种降维的思想。通过降维将相关性高的变量聚在一起，从而减少需要分析的变量的数量，而减少问题分析的复杂性。

因子分析的前提是具有一定的相关性，因此必须通过了 **kmo 和 bartlett 球形度检验**的数据才能进行因子分析。

因子分析前，首先进行 KMO 检验和巴特利球体检验，KMO 检验系数>0.5，（巴特利特球体检验的 x2 统计值的显著性概率）P 值<0.05 时，问卷才有结构效度，才能进行因子分析，因子分析主要是你自己做了一份调查问卷，你要考量这份问卷调查来的数据信度和效度如何，能不能对你想要调查的东西起代表性作用，说得很通俗不知道能不能理解。

球形检验主要是用于检验数据的分布，以及各个变量间的独立情况。简单一点说，按照理想情况，如果我们有一个变量，那么所有的数据都在一条线上。如果有两个完全独立的变量，则所有的数据在两条垂直的线上。如果有三条完全独立的变量，则所有的数据在三条相互垂直的线上。如果有 n 个变量，那所有的数据就会在 n 条相互垂直的线上，在每个变量取值范围大致相等的情况下，所有数据分布就像在一个球形体里面。想象一下万剑穿心的情形，大抵就是那个样子。如果不对数据分布进行球形检验，在做因素分析的时候就会违背因素分析的假设——各个变量在一定程度上相互独立。

```
#因子分析适用性检验
kmo = calculate_kmo(x)  # kmo值要大于0.7
bartlett = calculate_bartlett_sphericity(x)  # bartlett球形度检验p值要小于0.05
print('kmo:{},bartlett:{}'.format(kmo[1], bartlett[1]))
kmo:0.7164804529238993,bartlett:2.40899758533843e-221
```

通过了适用性检验后进行因子分析：

```
# 使用主成分分析法，9个因子维度拟合
fa = FactorAnalyzer(rotation=None, n_factors=9, method='principal')
fa.fit(x)
fa_9_sd = fa.get_factor_variance()
fa_9_df = pd.DataFrame(
    {'特征值': fa_9_sd[0], '方差贡献率': fa_9_sd[1], '方差累计贡献率': fa_9_sd[2]})

#各个因子的特征值以及方差贡献率
print(fa_9_df)
    特征值    方差贡献率 方差累计贡献率
0  2.779391  0.308821  0.308821
1  1.302006  0.144667  0.453489
2  1.157716  0.128635  0.582124
3  1.034961  0.114996  0.697119
4  0.659054  0.073228  0.770348
5  0.597498  0.066389  0.836736
6  0.571816  0.063535  0.900271
7  0.486350  0.054039  0.954310
8  0.411207  0.045690  1.000000
```

查看 9 个公因子的特征值以及方差贡献率，一般选择方差累计贡献率大于 0.8 的公因子，而文中选择了特征值大于 1 的公因子，即方差累计贡献率为 0.697 的前 4 个公因子。接着根据 4 个公因子重新拟合。

```
#公因子数设为4个，重新拟合
fa_4 = FactorAnalyzer(rotation=None, n_factors=4, method='principal')
fa_4.fit(x)

#查看公因子提取度
print(fa_4.get_communalities())
[0.74797278 0.67110816 0.74233283 0.73808703 0.66567718 0.77689271
 0.69607415 0.61523932 0.62069047]
```

查看公因子的提取度，发现当使用 4 个公因子时，4 个公因子对 9 个维度的解释率都超过 0.6，说明提取的 4 个公因子对原始维度有一定的解释力。

接着查看 4 个公因子的因子载荷，看看是否需要旋转。

```
#查看因子载荷
print(fa_4.loadings_)
[[ 0.41758577 -0.04630919  0.69667789  0.29341146]
 [ 0.5458075  -0.36656161  0.20321007  0.44445539]
 [ 0.64142608 -0.2555359  -0.50646667  0.09538522]
 [ 0.65217692 -0.36074267 -0.3699891   0.21383425]
 [ 0.54104332  0.4901776  -0.28576184 -0.22586587]
 [ 0.58866609 -0.09674514  0.23956478 -0.60300418]
 [ 0.64268497 -0.19004547  0.23352467 -0.43861045]
 [ 0.48559443  0.56693683 -0.10335096  0.21757431]
 [ 0.42690002  0.60240529  0.18274222  0.20532898]]
```

以第一个维度为例，我们发现 4 个公因子对原始的第一个维度的解释程度分别为：0.418，-0.046，0.697，0.293，表明公因子 1 与公因子 3 之间存在一定的相关性，达不到因子分析的既定效果，因此需要进行旋转，使得各个公因子具有差异化的特征。

```
#使用最大方差法旋转因子
fa_4_rotate = FactorAnalyzer(
    rotation='varimax', n_factors=4, method='principal')
fa_4_rotate.fit(x)

#查看z旋转后的因子载荷
print(fa_4_rotate.loadings_)
[[-0.06853212  0.15285685  0.20333033  0.8237522 ]
 [ 0.4912233   0.00448056  0.05911173  0.6529116 ]
 [ 0.83076037  0.1490026   0.16448475 -0.05397252]
 [ 0.83463555  0.07159505  0.1317772   0.13776588]
 [ 0.21485233  0.67063434  0.31903511 -0.26073327]
 [ 0.08489552  0.09023972  0.87007017  0.06723208]
 [ 0.20807596  0.06273611  0.78272561  0.1902192 ]
 [ 0.15963168  0.76368161 -0.01467611  0.07957422]
 [-0.06747767  0.74047386  0.03975419  0.25740106]]
```

还是以第一个维度为例，我们发现经过最大方差法旋转之后，4 个公因子对原始的第一个维度的解释程度分别为：-0.069，0.153，0.203，0.824，即公因子 4 对第一个维度的解释力较大。旋转后 4 个公因子在原始维度上被明显的区别出来，即 4 个公因子具有差异性的特征。

```
# 以因子最大值的因子为因子类型
facotor_result = pd.DataFrame(
    fa_4_rotate.fit_transform(x), columns=list('1234'))
facotor_result['因子类型'] = facotor_result.idxmax(axis=1)

#查看结果
print(facotor_result.head())
         1         2         3         4 因子类型
0 -0.011259  1.616764  1.398204  0.938451    2
1 -0.179805 -0.855833 -0.061380 -2.978798    3
2 -1.105833  0.729183 -0.664757 -0.664573    2
3  1.524168 -0.264195 -0.187541 -1.982722    1
4  0.437858 -0.520359 -0.091027 -0.654643    1

#将因子类型合并到原数据中
result = pd.concat([X.iloc[:, :-9], facotor_result['因子类型']], axis=1)
print(result.head())
问卷编号  是否购买车险  性别  年龄  城市  学历  家庭月收入 职业  汽车价格  决策时间  ...  投保渠道 保险
0     1       1   1   1   6   2      1  5     1     1  ...     4       3
1     2       1   2   1   7   2      1  1     1     1  ...     1       3
2     3       1   1   1   6   1      2  6     1     1  ...     4       4
3     4       1   1   3   5   2      2  6     3     1  ...     2       4
4     5       1   1   3   5   2      1  2     3     3  ...     2       3

     保费金额  索赔经历  一站式服务考虑程度  网上投保考虑程度  产品个性化考虑程度  选择保险公司的考虑因素
0   870.0     2          1          1          7          4    1     2
1  1199.7     1          5          4          5          2    1     3
2  1764.0     2          4          4          4          6    2     2
3  1770.0     2          4          4          4          4    2     1
4  1377.0     2          3          4          4          3    1     1
```

1.2 聚类分析

经过因子分析之后，我们把所有的客户分成了具有差异性特征的 4 类客户(代表了 9 个事后分类维度)，接着我们通过因子类型以及保费金额两个维度进行聚类分析。

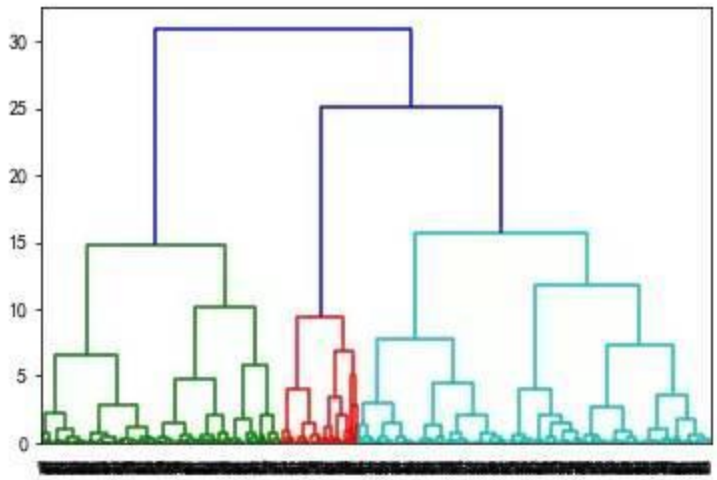
聚类分析是针对数据的相似性和差异性将一组数据分为几个类别。常用的聚类分析方法有 kmeans、DBSCAN 以及层次聚类。这里使用的是层次聚类，因为层次聚类对数据的类型要求不高而且事先不需要知道分为几类，缺点在于计算量大。

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。

打个比方，你作为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，例如，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

```
# 因子类型以及保费金额的量纲不一致，需进行标准化处理
result['因子类型'] = result['因子类型'].astype('int64')
result['Z 因子类型'] = (result['因子类型']-result['因子类型
'].mean())/result['因子类型'].std()
result['Z 保费金额'] = (result['保费金额']-result['保费金额
'].mean())/result['保费金额'].std()
result = result.set_index(result['问卷编号'])
#层次聚类分析
Z = hierarchy.linkage(result[['Z 保费金额', 'Z 因子类型']],
                      method='ward', metric='euclidean')
hierarchy.dendrogram(Z, labels=result.index)
# 看效果图，分为 5 类比较合适，即高度大概在 13 左右
label = hierarchy.cut_tree(Z, height=13)
label = label.reshape(label.size,)
result['细分类型'] = list(label)
```



通过层次聚类分析，我们将所有客户分为了 5 类。接下来我们通过方法分析检验分类的效果。我们发现无论是保费金额还是因子类型，通过层次聚类分组后，p 值都小于 0.05，即组间存在显著性差异，聚类效果良好。

```
#使用方差分析检验
聚类效果
a = anova_lm(ols('保费金额~C(细分类型)', data=result[['保费金额', '
细分类型']]).fit())[:1]
b = anova_lm(ols('因子类型~C(细分类型)', data=result[['因子类型', '
细分类型']]).fit())[:1]
f_oneway_result = pd.concat([a.iloc[:, 3:], b.iloc[:, 3:]])
f_oneway_result['列名'] = ['保费金额', '因子类型']
print(f_oneway_result)
                F         PR(>F)       列名
C(细分类型)  306.108565  1.157673e-152  保费金额
C(细分类型)  742.643495  1.999808e-251  因子类型
```

通过单因素方差分析，我们知道细分类型各组间存在显著性差异，那么这种差异怎么表现出来呢？对于类别型数据我们用占比比较，对于数值型数据我们用均值进行比较，然后根据保费金额以及因子类别对细分类型命名。

```
#保费金额使用均值比较，因子类别使用占比比较
nor = pd.crosstab(result['细分类型'], result['因子类型'],
                  normalize=0)  # normalize = 0 按行求占比
mean = result.groupby('细分类型')['保费金额'].mean()
result_xf = pd.concat([nor, mean], axis=1)
print(result_xf)
         1         2         3         4         保费金额
细分类型
0     0.000000  0.603774  0.396226  0.000000  1481.796226
1     0.657407  0.342593  0.000000  0.000000  2098.268056
2     0.000000  0.000000  0.578947  0.421053  2779.996241
3     0.000000  0.000000  0.000000  1.000000  1708.326829
4     0.530864  0.259259  0.197531  0.012346  3780.096296
#各细分类型命名
result['细分类型'] = result['细分类型'].map(
```

```
    {0: '低端居家型客户', 1: '中端享受型客户', 2: '中端外向型客户', 3:
'中端自信型客户', 4: '高端享受型客户'})
```

二、目标客户选取

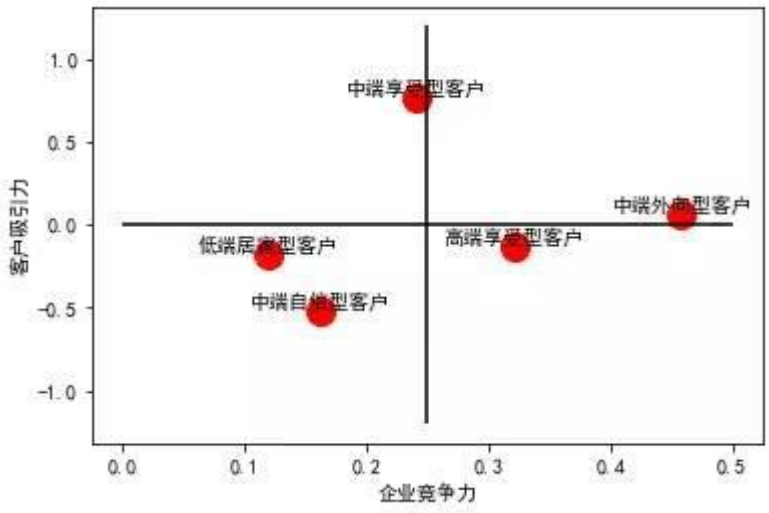
细分客户之后，要选取目标客户。选取目标客户主要从两个维度来度量，客户吸引力和企业竞争力。

企业吸引力主要体现在各个保险公司拥有各个细分类型客户的数量，即市场占有率。

客户吸引力包括两个方面，一是客户规模，二是保费金额，根据其公司需要，按权重 6：4 进行计算，得出客户吸引力。

```
#统计客户吸引力和企业竞争力
result_final = pd.DataFrame()
result_final['客户数量'] = result.groupby('细分类型')['问卷编号
'].count()
result_final['保费金额'] = result.groupby('细分类型')['保费金额
'].mean()
result_final['客户规模'] = result_final['客户数量
']/result_final['客户数量'].sum()
result_final['客户规模标准化'] = (
    result_final['客户规模']-result_final['客户规模
'].mean())/result_final['客户规模'].std()
result_final['保费金额标准化'] = (
    result_final['保费金额']-result_final['保费金额
'].mean())/result_final['保费金额'].std()
result_final['客户吸引力'] = 0.6*result_final['客户规模标准化
']+0.4*result_final['保费金额标准化']
result2 = pd.crosstab(result['细分类型'], result['保险公司的选择
'], normalize=0)
result2.columns = ['甲', '乙', '丙', '丁']
result_final['企业竞争力'] = result2['甲']
print(result_final)

                客户数量     保费金额     客户规模     客户规模标准化     保费金
```

```
额标准化    客户吸引力    企业竞争力
细分类型
中端享受型客户   216  2098.268056  0.303371  1.477388 -0.291968
0.769645  0.240741
中端外向型客户   133  2779.996241  0.186798 -0.188688  0.441347
0.063326  0.458647
中端自信型客户   123  1708.326829  0.172753 -0.389420 -0.711415
-0.518218  0.162602
低端居家型客户   159  1481.796226  0.223315  0.333215 -0.955087
-0.182106  0.119497
高端享受型客户    81  3780.096296  0.113764 -1.232494  1.517124
-0.132647  0.320988
#矩阵分析图
plt.rcParams['font.sans-serif'] = 'Simhei'
plt.rcParams['axes.unicode_minus'] = False
plt.subplot(1, 1, 1)
plt.scatter(result_final['企业竞争力'],
           result_final['客户吸引力'], s=200, c='r', marker='o')
plt.hlines(y=0, xmin=0, xmax=0.5)
plt.vlines(x=0.25, ymin=-1.2, ymax=1.2)
plt.xlabel('企业竞争力')
plt.ylabel('客户吸引力')
for a, b, c in zip(result_final['企业竞争力'], result_final['客户
吸引力'], result_final.index):
    plt.text(a, b, c, ha='center', va='bottom', fontsize=10)
```



由上图可知，中端外向型客户是甲公司的首选客户，其次是中端享受型客户和高端享受型客户，而低端居家型客户和中端自信型客户在资源不足的情况下暂可放弃。

三、目标客户定位

目标客户定位包括两部分，一是目标客户长什么样，也就是用户画像，通过事前分类维度描述目标客户画像；二是目标客户需求是什么，针对需求进行精准营销。分析过程依然是先进行方差分析，通过方差分析的维度用均值或者占比表现差异性，最后通过对应分析展示效果。

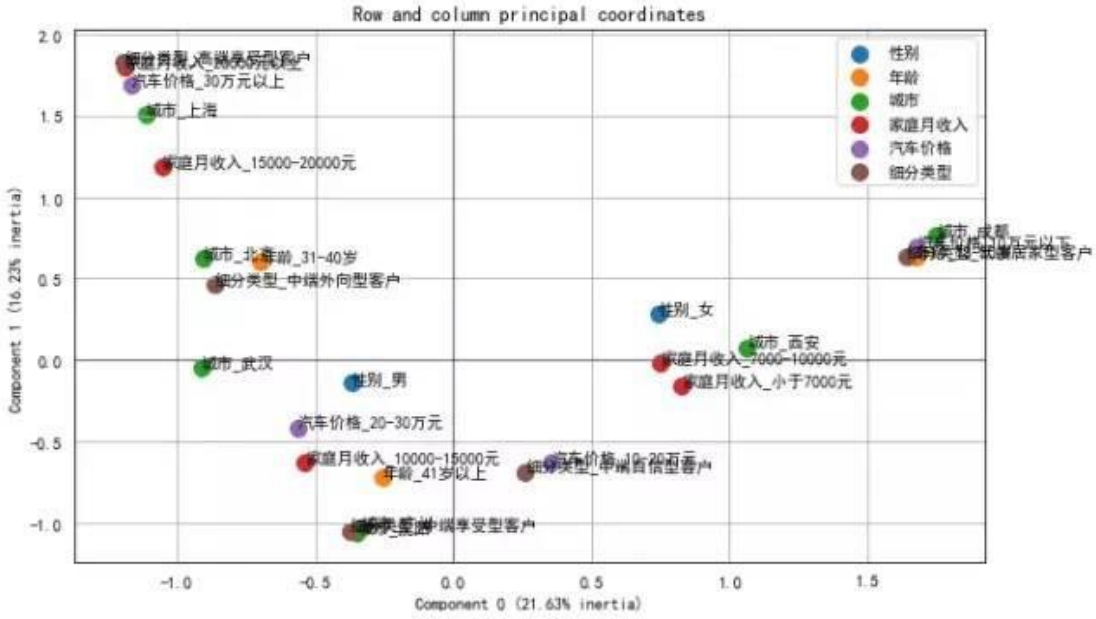
3.1 目标客户画像

事前分类维度一共由 6 个，分别是城市、年龄、性别、家庭月收入、汽车价格、学历以及职业。通过方差分析发现，学历以及职业在细分类型组间没有显著性差异，故忽略这两个维度继续分析。

```
#事前分类维度方差分析
result['职业'] = result['职业'].replace(' ', '6').astype('int64')
target_sd = []
for i in ['性别', '年龄', '城市', '家庭月收入', '汽车价格', '学历',
'职业']:
    formula = '(' + str(i) + '~' + 'C(' + '细分类型)' + ')'
    a = anova_lm(ols(formula, data=result[[i, '细分类型
']]).fit())[:1]
    target_sd.append(pd.DataFrame(
        {'c': str(i), 'F': a['F'], 'PR(>F)': a['PR(>F)']}))

target_result = pd.concat(target_sd)
target_result = target_result[target_result['PR(>F)'] < 0.05]

#具有显著性差异的维度结果展示
print(target_result)
```

```
                c            F         PR(>F)
C(细分类型)     性别    57.940193   2.614665e-42
C(细分类型)     年龄   553.274636  4.801252e-216
C(细分类型)     城市  3629.629395   0.000000e+00
C(细分类型)  家庭月收入   268.460859  3.193752e-140
C(细分类型)   汽车价格   901.193079  7.780527e-276
#具有显著性差异的维度命名
Y = result[['性别', '年龄', '城市', '家庭月收入', '汽车价格', '细分类型']]
Y['性别'] = Y['性别'].map({1: '男', 2: '女'})
Y['年龄'] = Y['年龄'].map({1: '18-30 岁', 2: '31-40 岁', 3: '41 岁以上'})
Y['城市'] = Y['城市'].map(
    {1: '北京', 2: '上海', 3: '武汉', 4: '沈阳', 5: '广州', 6: '西安', 7: '成都'})
Y['家庭月收入'] = Y['家庭月收入'].map(
    {1: '小于 7000 元', 2: '7000-10000 元', 3: '10000-15000 元', 4: '15000-20000 元', 5: '20000 元以上'})
Y['汽车价格'] = Y['汽车价格'].map(
    {1: '10 万元以下', 2: '10-20 万元', 3: '20-30 万元', 4: '30 万元以上'})

# 多因子对应分析
mca = prince.MCA(n_components=2, n_iter=10, random_state=1)
mca = mca.fit(Y)
ax = mca.plot_coordinates(
    X=Y,
    ax=None,
    figsize=(10, 6),
    show_row_points=False,
    show_column_points=True,
    column_points_size=100,
    show_column_labels=True,
    legend_n_cols=1
)
```

从对应分析效果图可知，甲公司首选目标客户中端外向型客户，主要分布在北京。武汉的分布比例也比其他细分类型要高，年龄集中在 31-40 之间，性别为男，家庭月收入在 1.5 万到 2 万元之间，汽车价格在 20-30 万元间。



3.2 目标客户需求分析

目标客户关注哪些需求，如何分析？依然是选定各细分类型客户，然后对各个维度进行方差分析，通过方差分析检验后的维度用均值或者占比进行比较。而未通过方差分析的维度，则直接用中端外向型客户进行各个维度的比较，数值型用均值，类型用占比。

具体见下：

```
#事前分类维度方差分析
result['职业'] = result['职业'].replace(' ', '6').astype('int64')
target_sd = []
for i in ['性别', '年龄', '城市', '家庭月收入', '汽车价格', '学历',
'职业']:
    formula = '(' + str(i) + '~' + 'C(' + '细分类型)' + ')'
    a = anova_lm(ols(formula, data=result[[i, '细分类型
']]).fit())[:1]
    target_sd.append(pd.DataFrame(
        {'c': str(i), 'F': a['F'], 'PR(>F)': a['PR(>F)']}))

target_result = pd.concat(target_sd)
target_result = target_result[target_result['PR(>F)'] < 0.05]
```

```
#具有显著性差异的维度结果展示
print(target_result)
                c            F         PR(>F)
C(细分类型)     性别     57.940193   2.614665e-42
C(细分类型)     年龄    553.274636  4.801252e-216
C(细分类型)     城市   3629.629395   0.000000e+00
C(细分类型)  家庭月收入    268.460859  3.193752e-140
C(细分类型)   汽车价格    901.193079  7.780527e-276
#具有显著性差异的维度命名
Y = result[['性别', '年龄', '城市', '家庭月收入', '汽车价格', '细分类型']]
Y['性别'] = Y['性别'].map({1: '男', 2: '女'})
Y['年龄'] = Y['年龄'].map({1: '18-30 岁', 2: '31-40 岁', 3: '41 岁以上'})
Y['城市'] = Y['城市'].map(
    {1: '北京', 2: '上海', 3: '武汉', 4: '沈阳', 5: '广州', 6: '西安', 7: '成都'})
Y['家庭月收入'] = Y['家庭月收入'].map(
    {1: '小于 7000 元', 2: '7000-10000 元', 3: '10000-15000 元', 4: '15000-20000 元', 5: '20000 元以上'})
Y['汽车价格'] = Y['汽车价格'].map(
    {1: '10 万元以下', 2: '10-20 万元', 3: '20-30 万元', 4: '30 万元以上'})

# 多因子对应分析
mca = prince.MCA(n_components=2, n_iter=10, random_state=1)
mca = mca.fit(Y)
ax = mca.plot_coordinates(
    X=Y,
    ax=None,
    figsize=(10, 6),
    show_row_points=False,
    show_column_points=True,
    column_points_size=100,
    show_column_labels=True,
    legend_n_cols=1
)
```

由上面的分析可知:

- 甲公司的目标客户中端外向型客户在选择保险公司考虑的因素中，比较关注服务网点多、亲朋推荐和信任销售人员，其中尤其关注亲朋的推荐。
- 在满意度分析中，发现中端外向型客户对目前购买的车险并不满意，满意度只有 1.5%，还有很大提升空间，不满意的具体原因还需进一步调研。
- 中端外向型客户车险平均保费在 2780 元，比其他细分客户更注重产品个性化，所以可以研究下定价策略和一些个性化产品。

总结

最后，市场上用户画像的方法很多，许多企业也提供用户画像服务，将用户画像提升到很有逼格一件事。金融企业是最早开始用户画像的行业，由于拥有丰富的数据，金融企业在进行用户画像时，对众多纬度的数据无从下手，总是认为用户画像数据纬度越多越好，画像数据越丰富越好，某些输入的数据还设定了权重甚至建立了模型，搞的用户画像是一个巨大而复杂的工程。但是费力很大力气进行了画像之后，却发现只剩下了用户画像，和业务相聚甚远，没有办法直接支持业务运营，投入精力巨大但是回报微小，可以说是得不偿失，无法向领导交代。

事实上，用户画像涉及数据的纬度需要业务场景结合，既要简单干练又要和业务强相关，既要筛选便捷又要方便进一步操作。用户画像需要坚持三个原则，分别是人口属性和信用信息为主，强相关信息为主，定性数据为主。

1、信用信息和人口属性为主

描述一个用户的信息很多，信用信息是用户画像中重要的信息，信用信息是描述一个人在社会中的消费能力信息。任何企业进行用户画像的目的是寻找目标客户，其必须是具有潜在消费能力的用户。信用信息可以直接证明客户的消费能力，是用户画像中最重要和基础的信息。一句戏言，所有的信息都是信用信息就是这个道理。其包含消费者工作、收入、学历、财产等信息。

定位完目标客户之后，企业需要触达客户，人口属性信息就是起到触达客户的作用，人口属性信息包含姓名、性别，电话号码，邮件地址，家庭住址等信息。这些信息可以帮助联系客户，将产品和服务推销给客户

2、采用强相关信息，忽略弱相关信息

强相关信息就是同场景需求直接相关的信息，其可以是因果信息，也可以是相关程度很高的信息。

如果定义采用 0 到 1 作为相关系数取值范围的化，0.6 以上的相关系数就应该定义为强相关信息。例如在其他条件相同的前提下，35 岁左右人的平均工资高于平均年龄为 30 岁的人，计算机专业毕业的学生平均工资高于哲学专业学生，从事金融行业工作的平均工资高于从事纺织行业的平均工资，上海的平均工资超过海南省平均工资。从这些信息可以看出来人的年龄、学历、职业、地点对收入的影响较大，同收入高低是强相关关系。简单的将，对信用信息影响较大的信息就是强相关信息，反之则是弱相关信息。

用户其他的信息，例如用户的身高、体重、姓名、星座等信息，很难从概率上分析出其对消费能力的影响，这些弱相关信息，这些信息就不应该放到用户画像中进行分析，对用户的信用消费能力影响很小，不具有较大的商业价值。

用户画像和用户分析时，需要考虑强相关信息，不要考虑弱相关信息，这是用户画像的一个原则。

3、将定量的信息归类为定性的信息

用户画像的目的是为产品筛选出目标客户，定量的信息不利于对客户进行筛选，需要将定量信息转化为定性信息，通过信息类别来筛选人群。

例如可以将年龄段对客户进行划分，18 岁-25 岁定义为年轻人，25 岁-35 岁定义为中青年，36-45 定义为中年人等。可以参考个人收入信息，将人群定义为高收入人群，中等收入人群，低收入人群。参考资产信息也可以将客户定义为高、中、低级别。定性信息的类别和方式方法，金融可以从自身业务出发，没有固定的模式。

将企业各类定量信息，集中在一起，对定性信息进行分类，并进行定性化，有利与对用户进行筛选，快速定位目标客户，是用户画像的另外一个原则。

4、用户画像的方法介绍，不要太复杂

需要结合业务需求进行用户画像，从实用角度出发，比如这里我们将用户画像信息分成五类信息。分别是社会特征因素、自然属性因素、行为特征因素、态度偏好因素和生活状态与个性因素。它们基本覆盖了业务需求所需要的强相关信息，

结合外部场景数据将会产生巨大的商业价值。特别复杂的用户画像纬度例如八个纬度，十个纬度信息都不利于商业应用，其他具有价值的信息，基本上都可以归纳到这五个纬度。过于复杂用户画像这个工作，对商业意义也不太大。