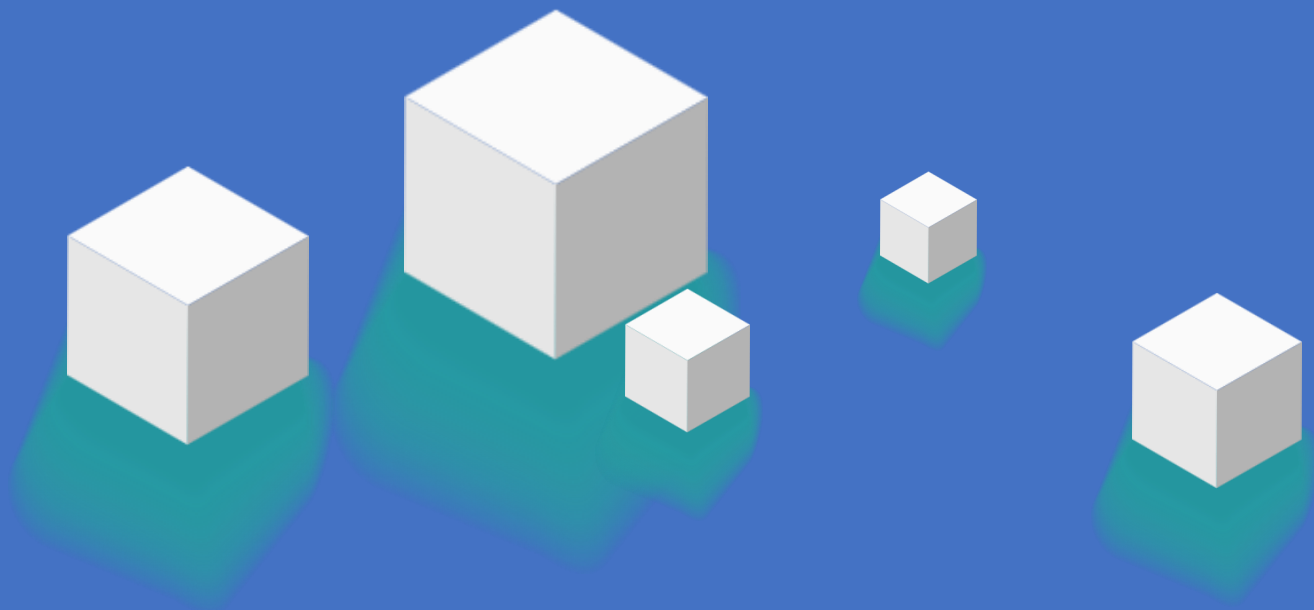


# 客户续保预测



# Case：客户续保预测

## Case：客户续保预测

寿险行业是一个高度依赖于客户忠诚度和持续支付保费的领域。准确预测客户是否续保可以帮助保险公司提前采取措施，减少流失率，比如提供更加个性化的服务或者优化产品设计。

训练集：train.csv 1000条

测试集：test.csv 200条

<https://www.kaggle.com/t/467370365f1747868eadbd65eeb970c5>

英文名称	含义描述
policy_id	每个保单的唯一标识符
age	客户的年龄，范围从18岁到70岁。
gender	客户的性别，分为“男”和“女”。
birth_region	客户的出生地
insurance_region	客户投保时所在的地区
income_level	客户的收入水平，分为“低”、“中”和“高”。
education_level	客户的最高教育程度，分为“高中”、“本科”、“硕士”和“博士”。
occupation	客户的职业，例如“销售”、“经理”、“设计师”、“工程师”、“医生”等。
marital_status	客户的婚姻状况，分为“单身”、“已婚”和“离异”。
family_members	客户的家庭成员数量，单身客户通常有1-2人，已婚客户通常有3-6人。
policy_type	保单的类型，例如“平安六福保”、“盛世福尊悦版”、“优悦版”等。
policy_term	保单的有效期限，分为“1年”、“5年”、“10年”和“20年”。
premium_amount	客户每年需要支付的保费金额
policy_start_date	保单开始生效的日期
policy_end_date	保单到期的日期，根据保单生效日期和保单期限计算。
claim_history	保单是否有过理赔记录，分为“是”和“否”。
renewal	是否续保，Yes 或者 No

# Case：客户续保预测

提交结果以CSV的格式，采用accuracy进行评价标准

工作中，可能还会用到其他评价标准：

$$F1 = 2 \frac{p \cdot r}{p + r} \text{ where } p = \frac{tp}{tp + fp}, \quad r = \frac{tp}{tp + fn}$$

比如F1值，可以平衡精确率和召回率

精确率 Precision: True Positive在all predicted positives的比例

召回率 Recall: True Positive在all actual positives的比例

policy_id	renewal
10001	Yes
10002	No
10003	Yes
10004	Yes
10005	Yes
10006	Yes
10007	No
10008	No
10009	Yes
10010	Yes
10011	Yes
10012	Yes
.....	.....
10200	Yes

policy\_result.csv，提交预测结果

# Case：客户续保预测

## Step1，数据预处理

对数据进行可视化探索

缺失值处理，异常值处理，唯一值统计

分类变量数值化

## Step2，模型选择（训练&评估）

随机森林, XGBoost, LightGBM

决策树、逻辑回归

## Step3，模型预测（提交预测结果）

To DO：数据预处理

policy_id	renewal
10001	Yes
10002	No
10003	Yes
10004	Yes
10005	Yes
10006	Yes
10007	No
10008	No
10009	Yes
10010	Yes
10011	Yes
10012	Yes
.....	.....
10200	Yes

policy\_result.csv，提交预测结果

# 数据探索

---

**Thinking:** 针对客户续保，都有哪些维度可以进行数据探索？

基本人口统计特征

- 年龄分布
- 性别差异
- 出生地区与投保所在地区的关联

经济状况

- 收入水平
- 教育背景
- 职业类型

家庭结构

- 婚姻状况
- 家庭成员数量

保单详情

- 保单类型
- 保费金额
- 保单期限
- 理赔历史

时间和地域因素

- 保单生效日期
- 投保所在地区

# 数据探索

```
# 年龄分布
```

```
plt.figure(figsize=(10, 6))
```

```
sns.histplot(df['age'], bins=50, kde=True)
```

```
plt.title('客户年龄分布')
```

```
plt.xlabel('年龄')
```

```
plt.ylabel('人数')
```

```
plt.show()
```

```
# 性别差异
```

```
gender_renewal =
```

```
df.groupby('gender')['renewal'].value_counts(normalize=True).unstack()
```

```
gender_renewal.plot(kind='bar', stacked=True, figsize=(8, 6), title='不同  
性别续保情况')
```

```
plt.ylabel('比例')
```

```
plt.show()
```

```
# 出生地区与投保所在地区的关联
```

```
region_cross = pd.crosstab(df['birth_region'], df['insurance_region'])
```

```
plt.figure(figsize=(12, 10))
```

```
sns.heatmap(region_cross, cmap='YlGnBu', annot=False, fmt='.0f',  
cbar_kws={'label': '人数'})
```

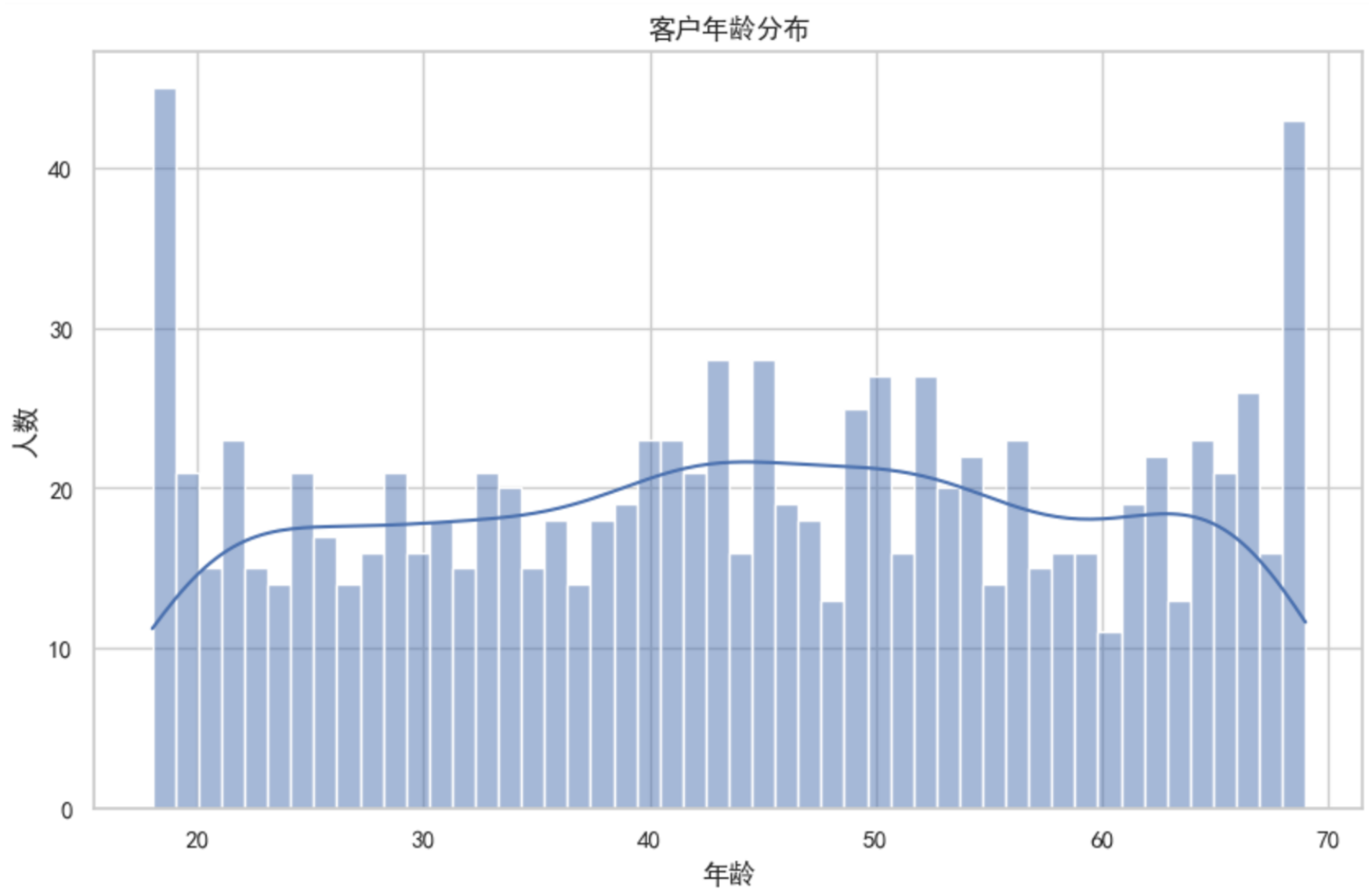
```
plt.title('出生地区与投保所在地区的热力图')
```

```
plt.xlabel('投保所在地区')
```

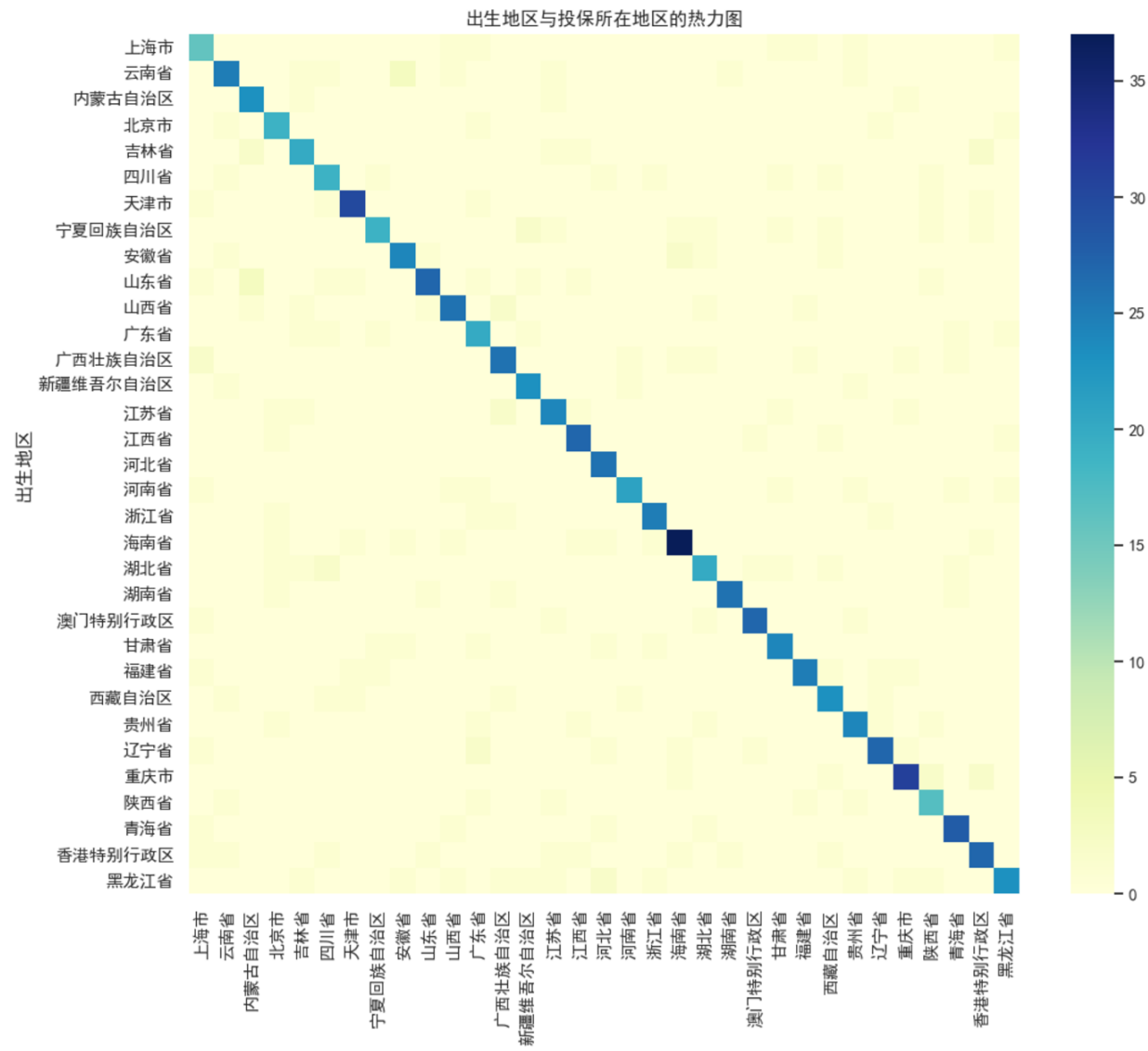
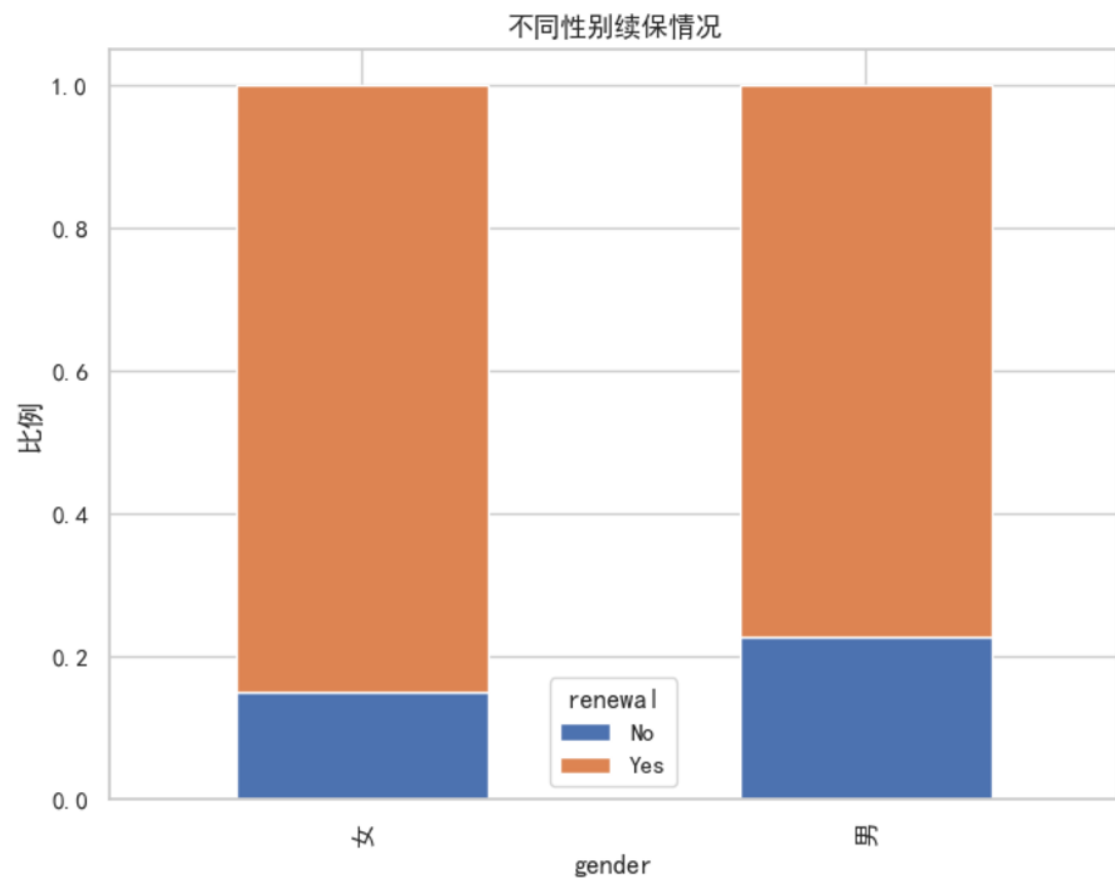
```
plt.ylabel('出生地区')
```

```
plt.show()
```

# 数据探索（人口特征）

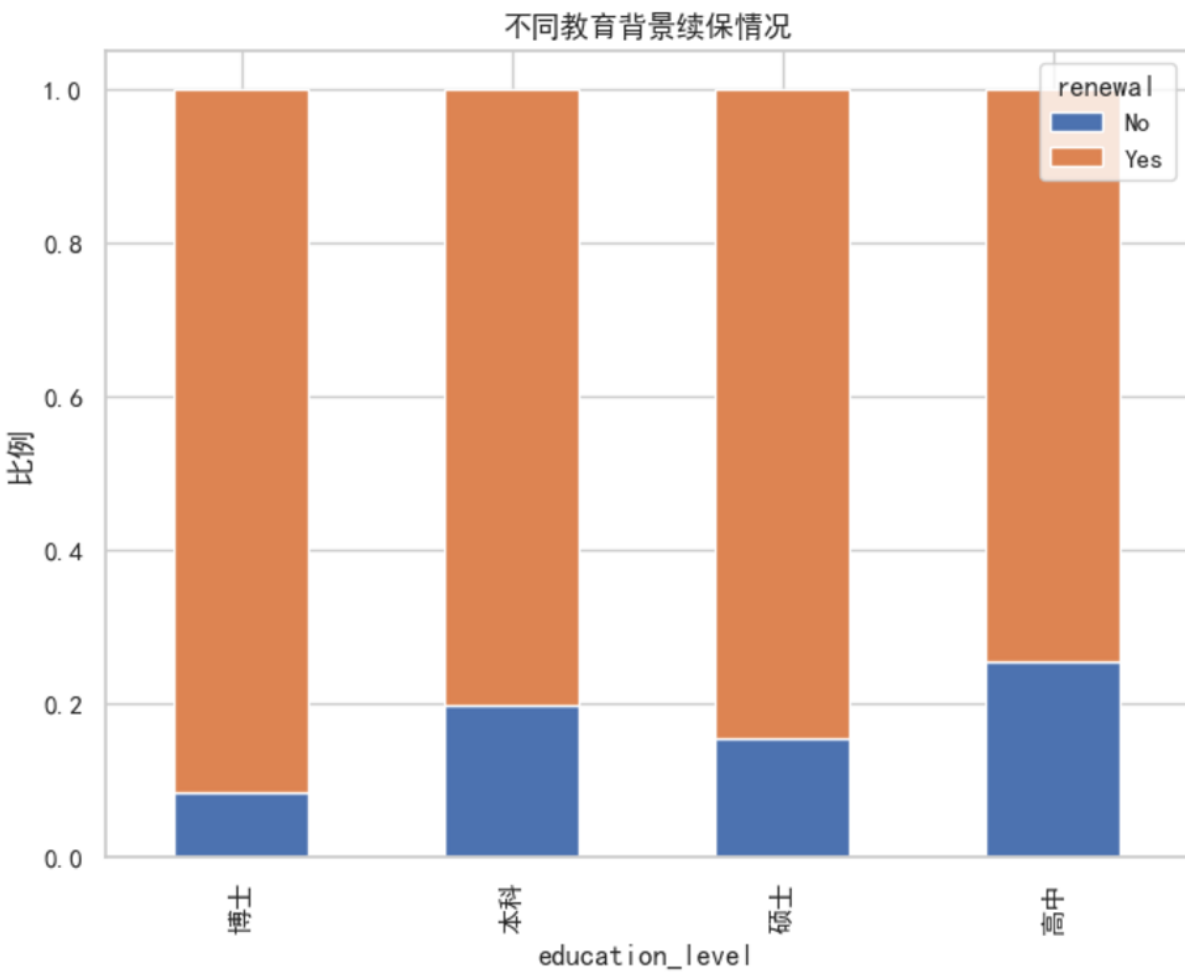
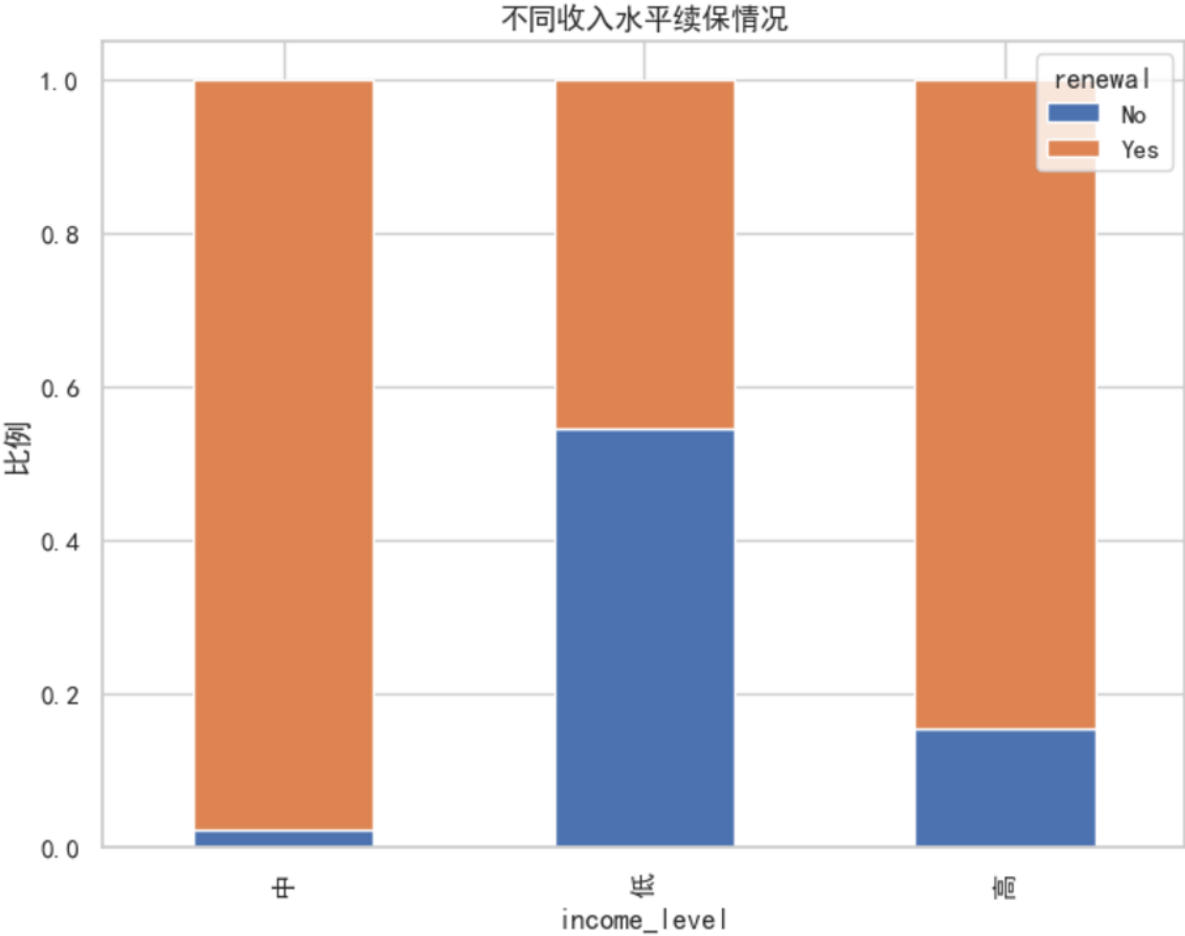


# 数据探索（人口特征）

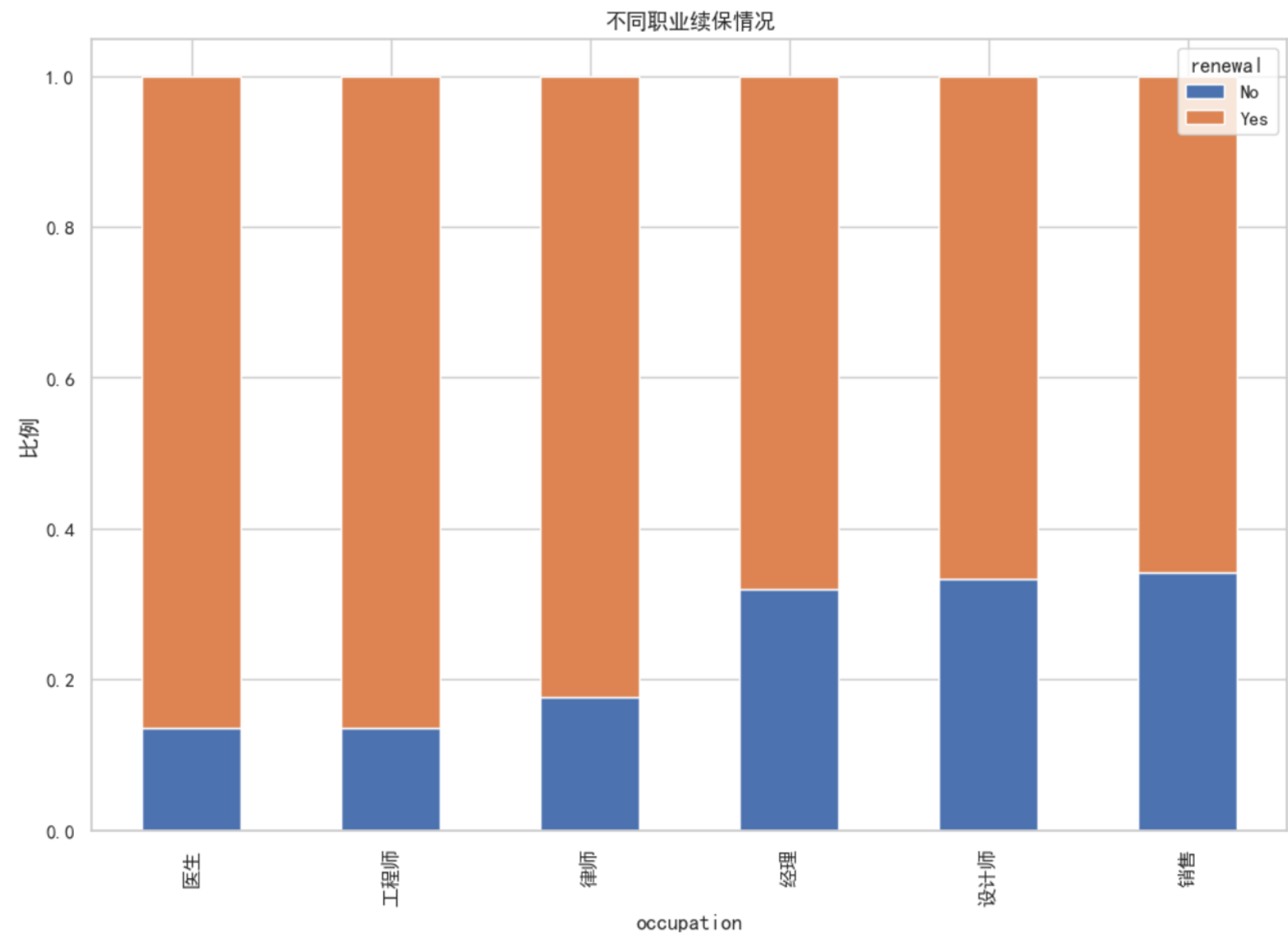




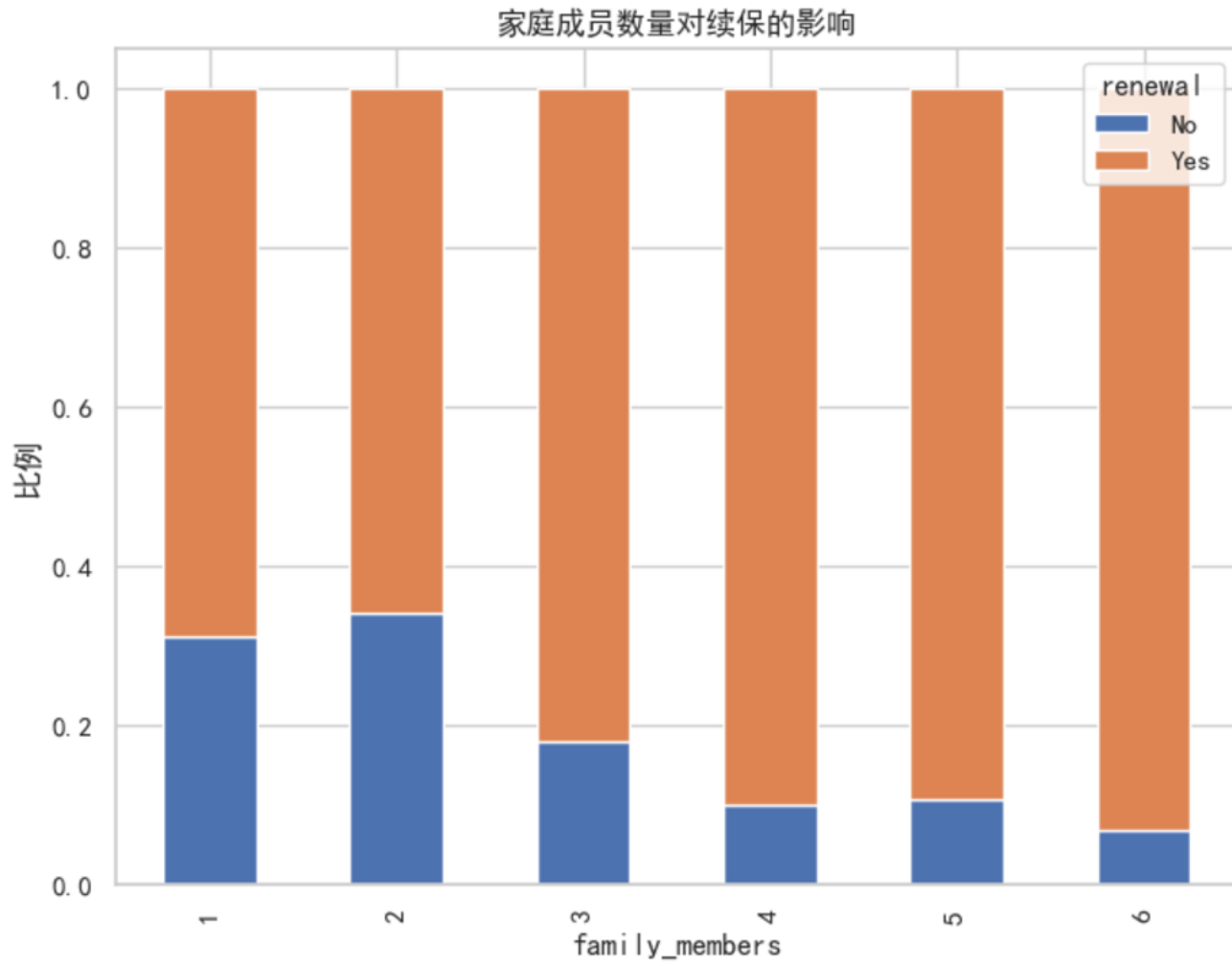
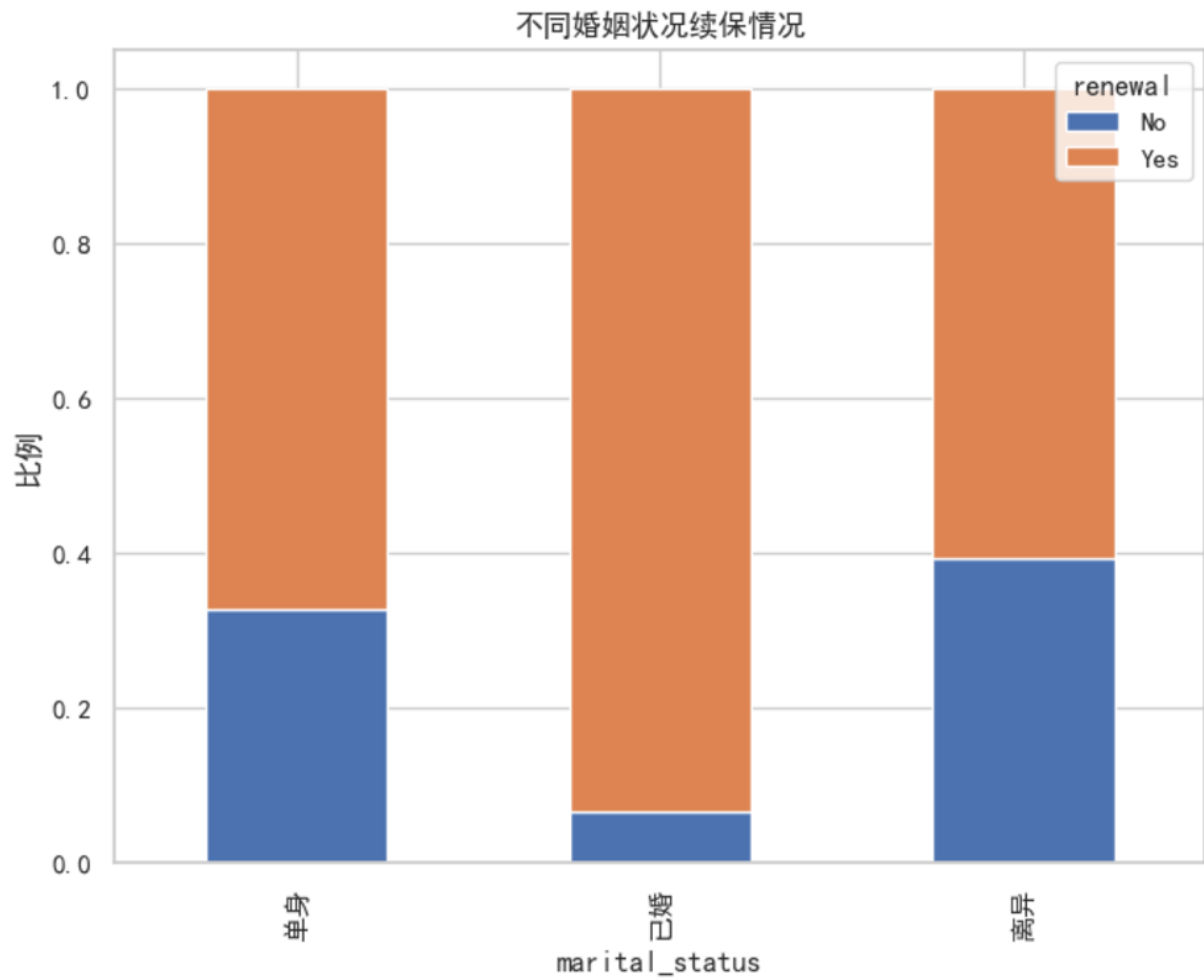
# 数据探索（经济情况）



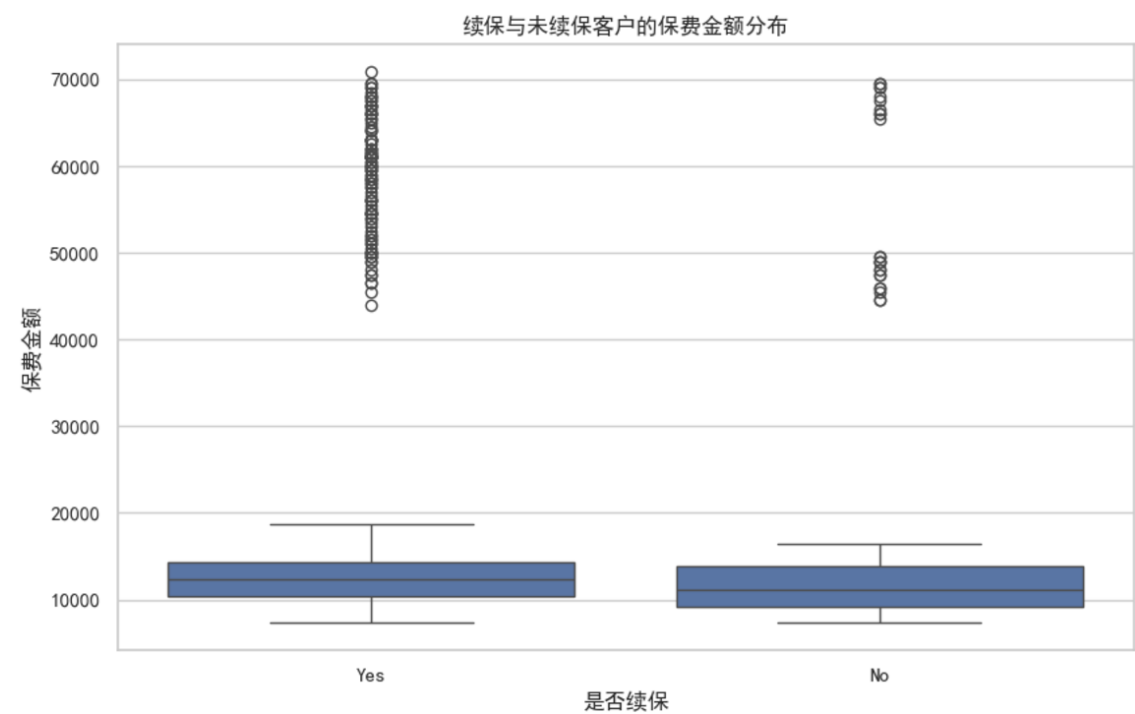
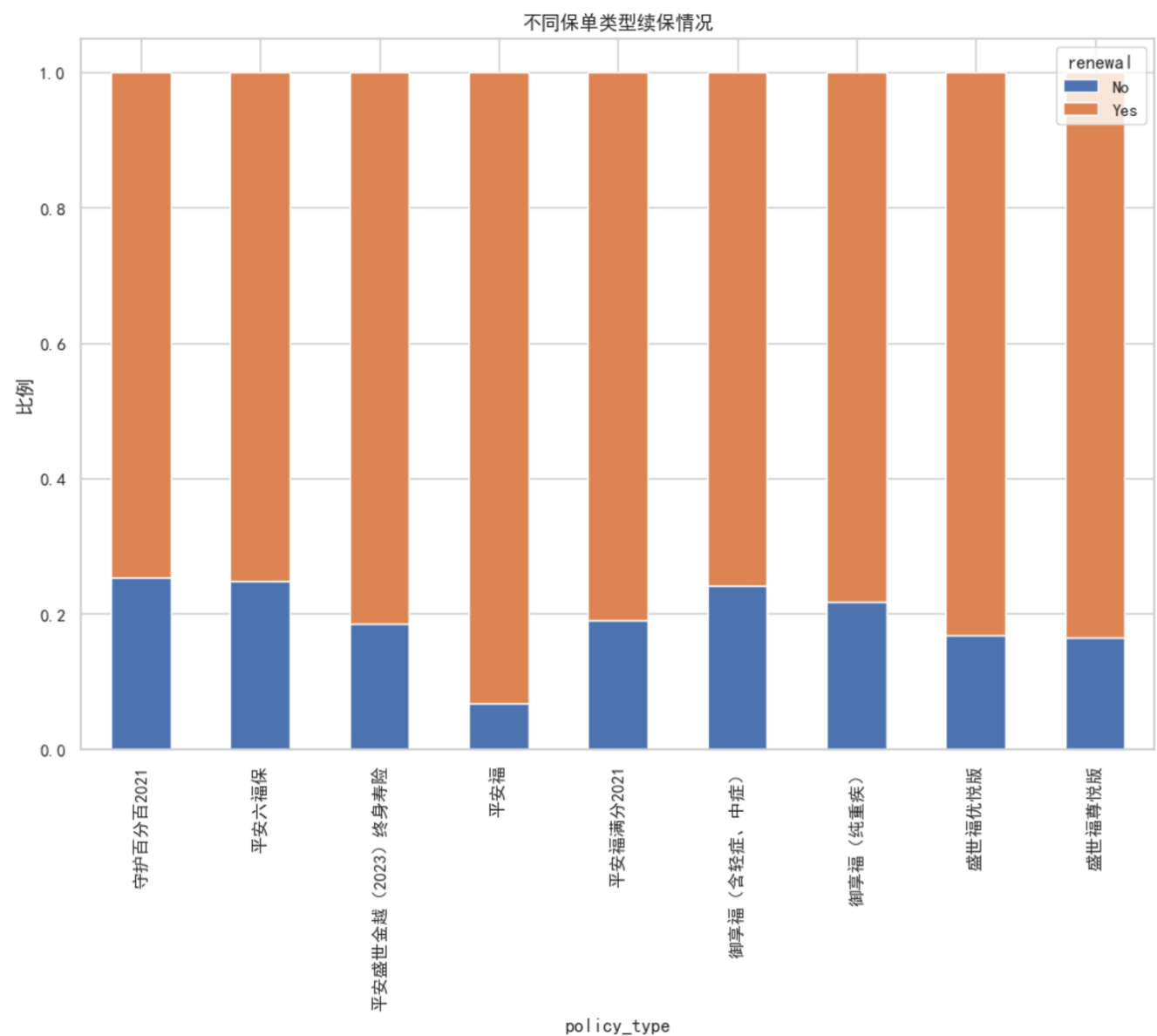
# 数据探索（经济情况）



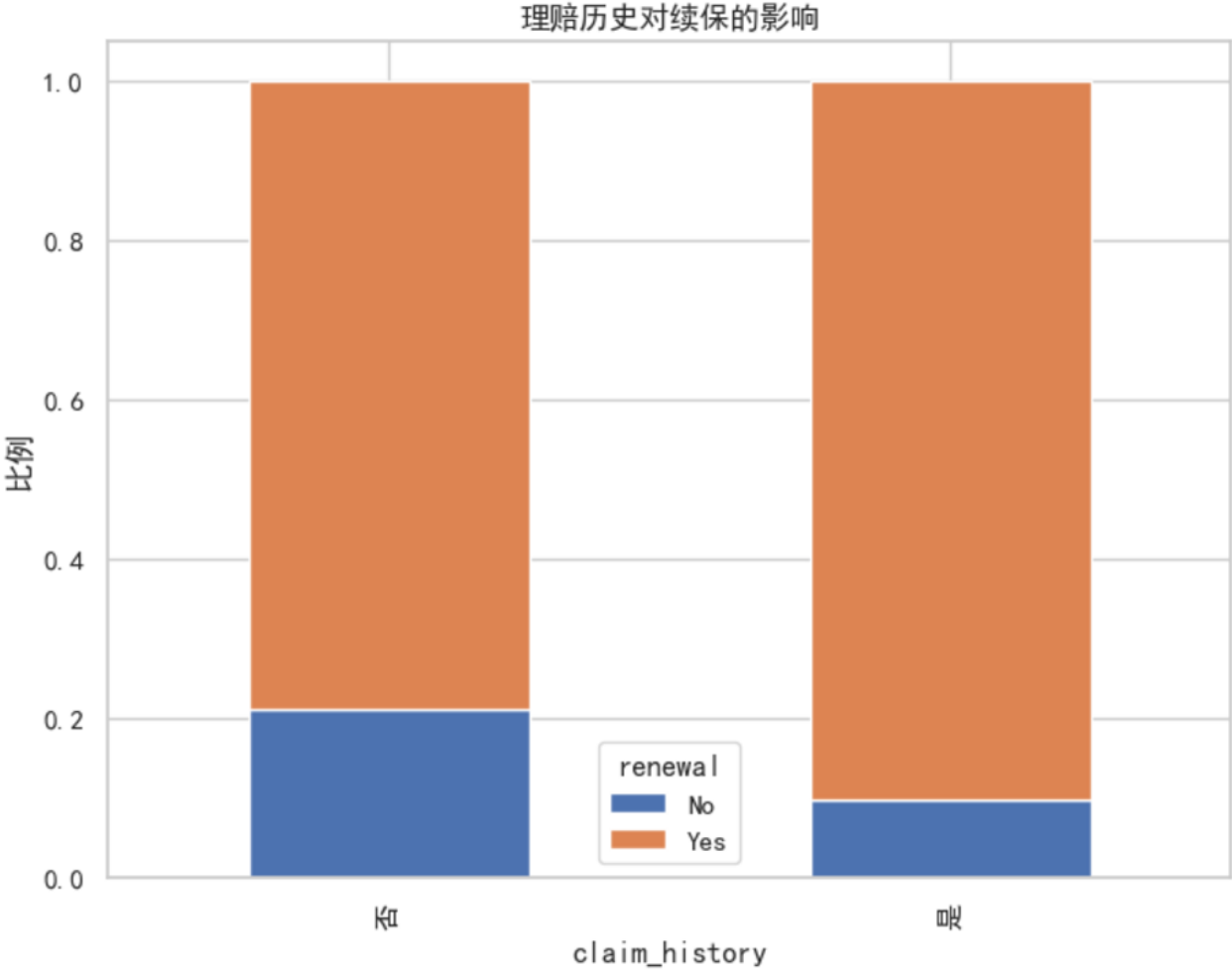
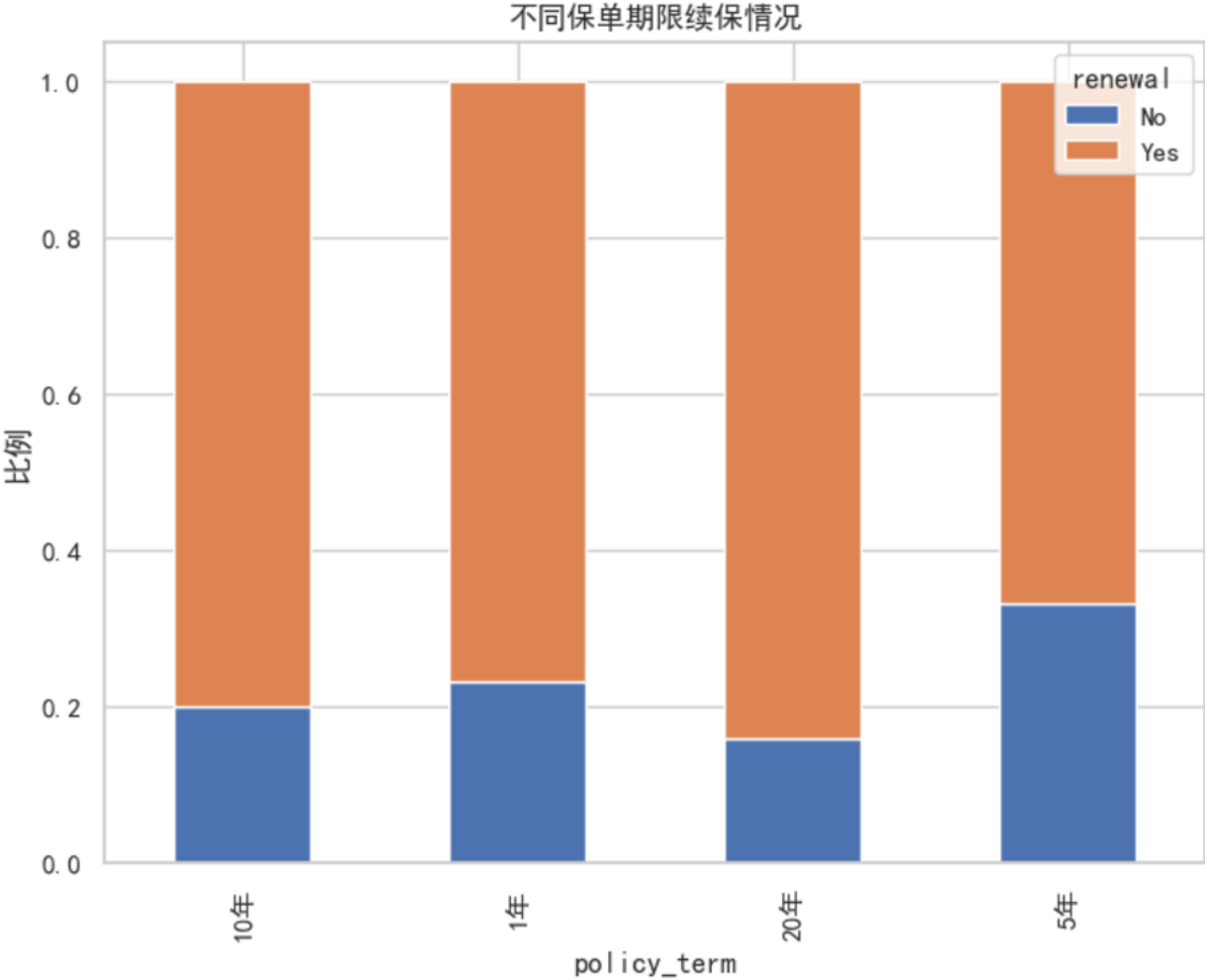
# 数据探索（家庭结构）



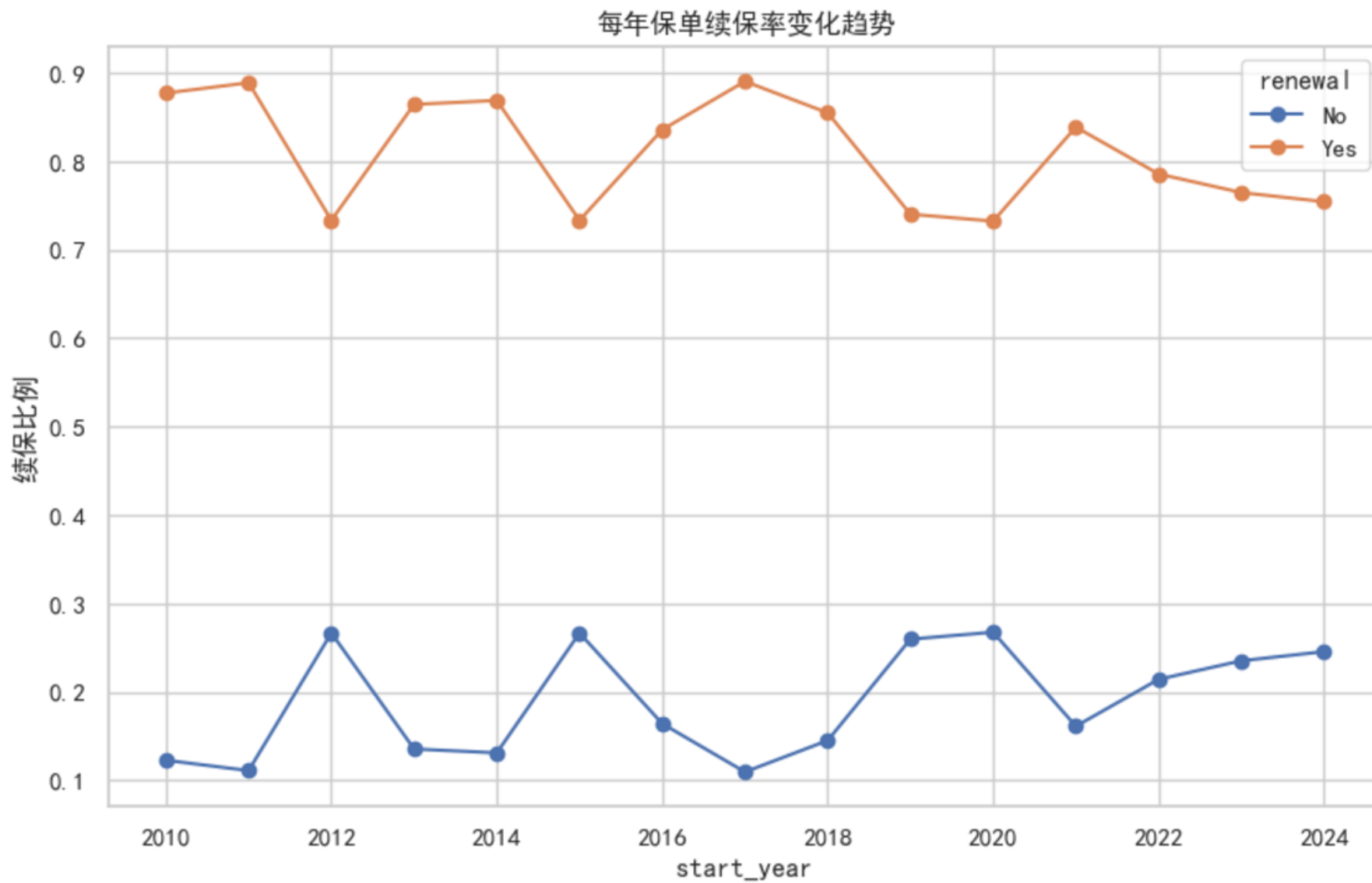
# 数据探索（保单详情）



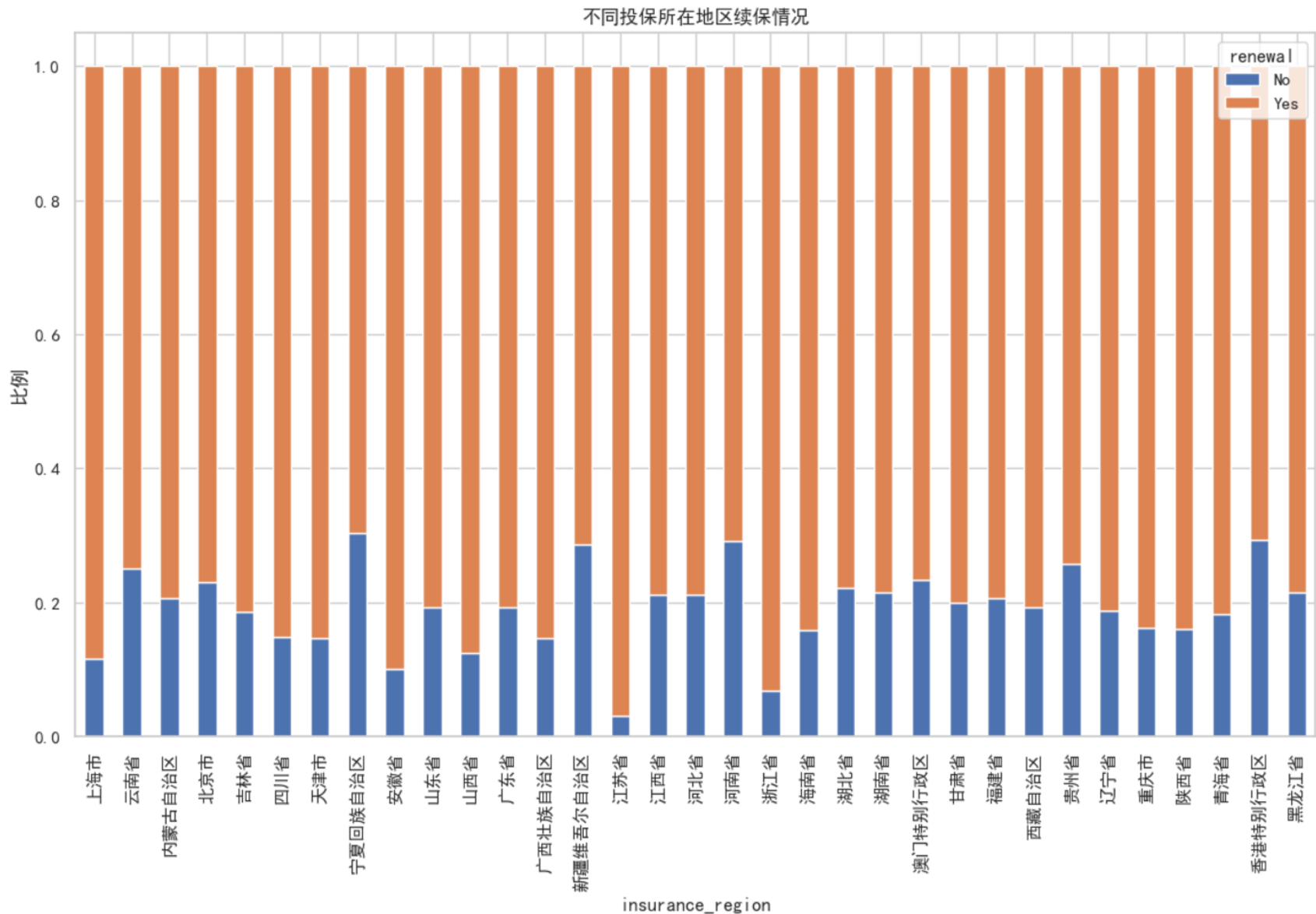
# 数据探索（保单详情）



# 数据探索（时间和地域因素）



# 数据探索（时间和地域因素）



# 数据预处理

---

```
# 查看分类变量
```

```
cate_cols = df.select_dtypes(include=['object']).columns
```

```
cate_cols
```

```
Index(['gender', 'birth_region', 'insurance_region', 'income_level',  
      'education_level', 'occupation', 'marital_status', 'policy_type',  
      'policy_term', 'claim_history', 'renewal'],  
      dtype='object')
```

```
# policy_term字段：去掉'年'字，并转换为整数
```

```
df['policy_term'] = df['policy_term'].str.replace('年', "").astype(int)
```



# 数据预处理

```
# 对income_level进行数值编码
income_map = {'低': 0, '中': 1, '高': 2}
df['income_level'] = df['income_level'].map(income_map)

# 对education_level进行数值编码
education_map = {'高中': 0, '本科': 1, '硕士': 2, '博士': 3}
df['education_level'] = df['education_level'].map(education_map)

# 对marital_status进行数值编码
marital_map = {'单身': 0, '已婚': 1, '离异': 2}
df['marital_status'] = df['marital_status'].map(marital_map)
```

```
# 对gender进行数值编码
gender_map = {'女': 0, '男': 1}
df['gender'] = df['gender'].map(gender_map)

# 对claim_map进行数值编码
renewal_map = {'Yes': 1, 'No': 0}
df['renewal'] = df['renewal'].map(renewal_map)

# 将分类变量转换为数值型
le = LabelEncoder()
cate_cols = ['birth_region', 'insurance_region',
             'occupation', 'policy_type']
for col in cate_cols:
    df[col] = le.fit_transform(df[col])
```

# 数据预处理

	policy_id	age	gender	birth_region	insurance_region	income_level	education_level	occupation	marital_status	family_members	policy_type	policy_term
0	10001	56	1	20	5	2	1	0	1	5	0	2
1	10002	69	1	31	31	2	3	0	0	1	7	2
2	10003	46	1	30	10	1	1	1	0	1	7	1
3	10004	32	0	16	16	1	2	2	1	6	2	1
4	10005	60	1	29	29	2	1	1	1	3	8	2
...	...	...	...	...	...	...	...	...	...	...	...	...
995	10996	60	0	16	16	2	2	0	2	6	1	1
996	10997	64	1	32	18	2	1	4	1	4	8	
997	10998	62	0	22	22	2	3	1	0	2	5	2
998	10999	35	1	27	27	1	2	0	0	2	8	1
999	11000	55	1	5	5	2	1	2	1	3	7	2

1000 rows × 17 columns

# 续保预测： 随机森林

```
import pandas as pd
df = pd.read_excel('./policy_clean.xlsx')
# 转换为自某一起始点的天数
df['policy_start_date'].min()
base_date = pd.to_datetime('2010-01-01')
df['policy_start_date_day'] = (df['policy_start_date'] -
base_date).dt.days
df[['policy_start_date', 'policy_start_date_day']]

df['policy_end_date_day'] = (df['policy_end_date'] -
base_date).dt.days
df[['policy_end_date', 'policy_end_date_day']]
```

policy_start_date			policy_end_date		
	policy_start_date	policy_start_date_day		policy_end_date	policy_end_date_day
0	2015-01-17	1842	0	2035-01-17	9147
1	2015-05-20	1965	1	2035-05-20	9270
2	2021-06-10	4178	2	2031-06-10	7830
3	2017-01-29	2585	3	2027-01-29	6237
4	2013-06-04	1250	4	2033-06-04	8555
...	...	...	...	...	...
995	2015-07-05	2011	995	2025-07-05	5664
996	2022-10-24	4679	996	2027-10-24	6505
997	2017-09-21	2820	997	2037-09-21	10125
998	2015-04-21	1936	998	2025-04-21	5589
999	2011-02-18	413	999	2031-02-18	7718
1000 rows × 2 columns			1000 rows × 2 columns		

# 续保预测：随机森林

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix

# 特征和目标变量
X = df.drop(['renewal', 'policy_start_date', 'policy_end_date'], axis=1)
y = df['renewal']

# 标准化特征
scaler = StandardScaler()
X_scaler = scaler.fit_transform(X)

# 划分训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(X_scaler, y, test_size=0.2, random_state=42)
```

以上代码都可以复用，用于 决策树、逻辑回归等

# 续保预测：随机森林

```
# 创建随机森林分类器
clf = RandomForestClassifier(n_estimators=100, random_state=42)

# 训练模型
clf.fit(X_train, y_train)

# 预测
y_pred = clf.predict(X_test)

# 评估模型
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Accuracy: 0.925

Confusion Matrix:

[[ 33 11]

[ 4 152]]

Classification Report:

	precision	recall	f1-score	support
0	0.89	0.75	0.81	44
1	0.93	0.97	0.95	156
accuracy		0.93		200
macro avg	0.91	0.86	0.88	200
weighted avg	0.92	0.93	0.92	200

# 续保预测：决策树

```
# 创建决策树分类器
clf = DecisionTreeClassifier(random_state=42, max_depth=4)

# 训练模型
clf.fit(X_train, y_train)

# 预测
y_pred = clf.predict(X_test)

# 评估模型
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Accuracy: 0.93

Confusion Matrix:

[[ 34 10]

[ 4 152]]

Classification Report:

	precision	recall	f1-score	support
0	0.89	0.77	0.83	44
1	0.94	0.97	0.96	156
accuracy			0.93	200
macro avg	0.92	0.87	0.89	200
weighted avg	0.93	0.93	0.93	200

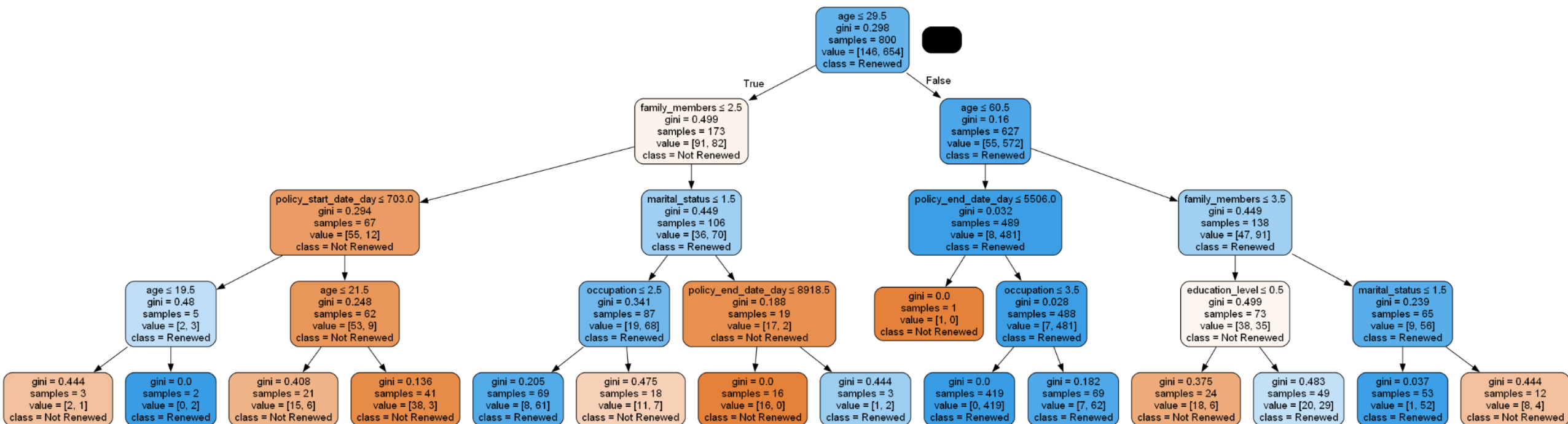
# 续保预测：决策树

# 可视化决策树

```
dot_data = export_graphviz(clf, out_file=None, feature_names=X.columns, class_names=['Not  
Renewed', 'Renewed'], filled=True, rounded=True, special_characters=True)
```

```
graph = pydotplus.graph_from_dot_data(dot_data)
```

```
Image(graph.create_png())
```



# 续保预测：决策树

```
from sklearn.tree import DecisionTreeClassifier, export_text
```

```
# 输出决策树的文本表示
```

```
tree_rules = export_text(clf, feature_names=list(X.columns))
```

```
print(tree_rules)
```

```
|--- age <= 29.50
|   |--- family_members <= 2.50
|       |--- policy_start_date_day <= 703.00
|           |--- age <= 19.50
|               |--- class: 0
|           |--- age > 19.50
|               |--- class: 1
|       |--- policy_start_date_day > 703.00
|           |--- age <= 21.50
|               |--- class: 0
|           |--- age > 21.50
|               |--- class: 0
|   |--- family_members > 2.50
|       |--- marital_status <= 1.50
|           |--- occupation <= 2.50
|               |--- class: 1
|           |--- occupation > 2.50
|               |--- class: 0
|       |--- marital_status > 1.50
|           |--- policy_end_date_day <= 8918.50
|               |--- class: 0
|           |--- policy_end_date_day > 8918.50
|               |--- class: 1
|--- age > 29.50
|   |--- age <= 60.50
|       |--- policy_end_date_day <= 5506.00
|           |--- class: 0
|       |--- policy_end_date_day > 5506.00
|           |--- occupation <= 3.50
|               |--- class: 1
|           |--- occupation > 3.50
|               |--- class: 1
|   |--- age > 60.50
|       |--- family_members <= 3.50
|           |--- education_level <= 0.50
|               |--- class: 0
|           |--- education_level > 0.50
|               |--- class: 1
|       |--- family_members > 3.50
|           |--- marital_status <= 1.50
|               |--- class: 1
|           |--- marital_status > 1.50
|               |--- class: 0
```



# 续保预测：逻辑回归

```
# 创建逻辑回归分类器
clf = LogisticRegression(random_state=42, max_iter=200) # 增加
max_iter以确保收敛

# 训练模型
clf.fit(X_train, y_train)

# 预测
y_pred = clf.predict(X_test)

# 评估模型
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Accuracy: 0.835

Confusion Matrix:

```
[[ 11  33]
```

```
 [  0 156]]
```

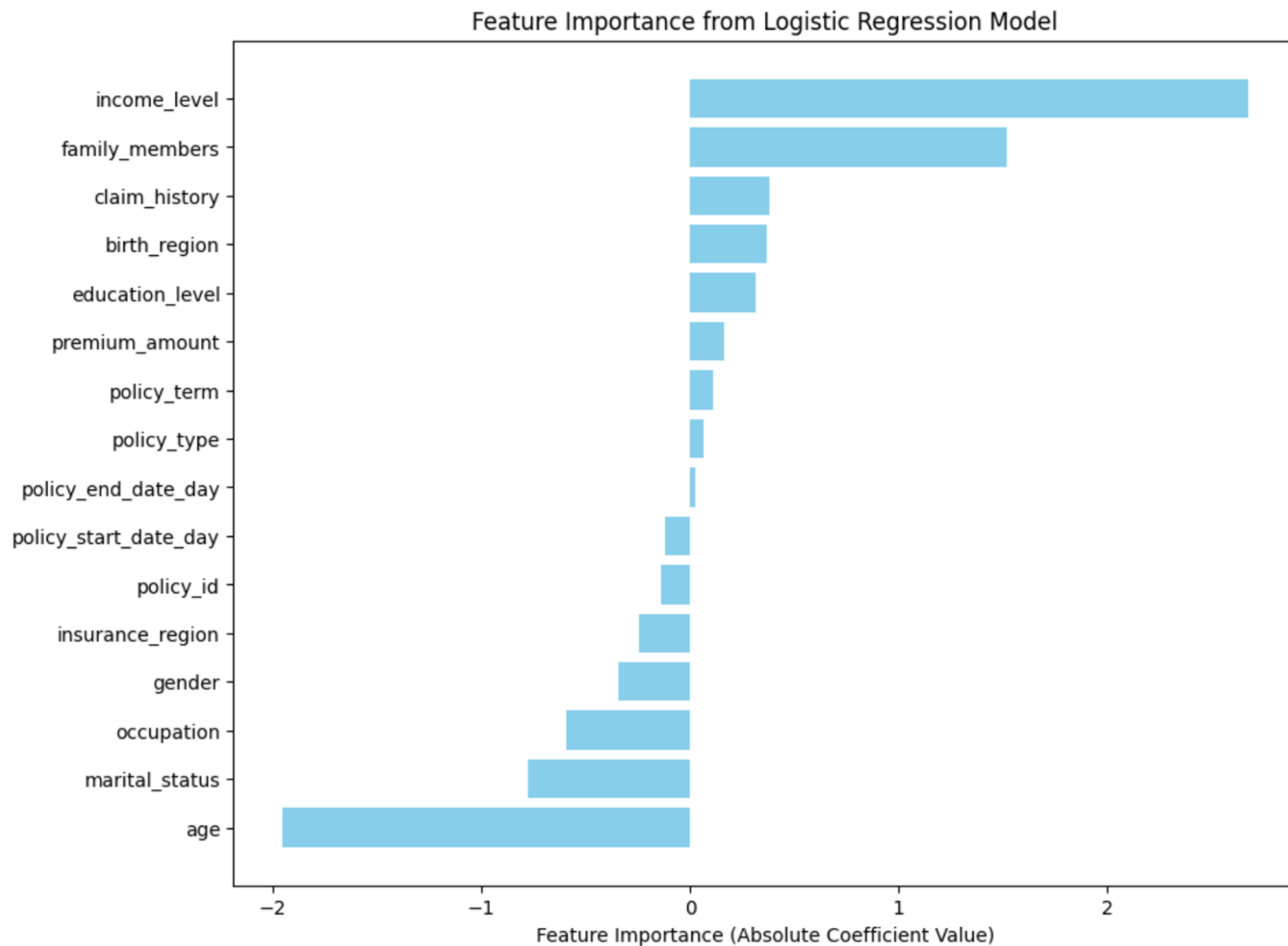
Classification Report:

	precision	recall	f1-score	support
0	1.00	0.25	0.40	44
1	0.83	1.00	0.90	156
accuracy		0.83		200
macro avg	0.91	0.62	0.65	200
weighted avg	0.86	0.83	0.79	200

# 续保预测：逻辑回归

```
# 获取特征重要性（通过系数）
feature_importance = clf.coef_[0] # 取第一个类别的系数绝对值
feature_names = X.columns
# 将特征重要性和特征名称组合成一个DataFrame
importances_df = pd.DataFrame({'Feature': feature_names, 'Importance': feature_importance})
# 按照重要性从大到小排序
importances_df = importances_df.sort_values(by='Importance', ascending=False)
# 绘制水平柱状图
plt.figure(figsize=(10, 8))
plt.barh(importances_df['Feature'], importances_df['Importance'], color='skyblue')
plt.xlabel('Feature Importance (Absolute Coefficient Value)')
plt.title('Feature Importance from Logistic Regression Model')
plt.gca().invert_yaxis() # 使得最重要的特征在顶部
plt.show()
```

# 续保预测：逻辑回归



# 模型洞察

# 常用模型对比

常用模型对比：

模型	优势	劣势
逻辑回归	简单、稳定 可解释性强 易于检测和部署	不适用于非线性问题 准确率不高
决策树	对数据质量要求低 可解释性强	易于过拟合 准确度不高
集成模型 (RF、GBDT)	准确度高 鲁棒性好 泛化能力强	可解释性差 计算量大 部署较困难

LR模型是评分卡模型最经常使用的算法

评分卡模型：

变量	分箱类别	分数
基准分	--	$A - B\theta_0$
$x_1$	1	$-B\theta_1w_{11}$
	2	$-B\theta_1w_{12}$
	...	...
	$k_1$	$-B\theta_1w_{1k_1}$
...	...	...
$x_n$	1	$-B\theta_nw_{n1}$
	2	$-B\theta_nw_{n2}$
	...	...
	$k_n$	$-B\theta_nw_{nk_n}$

$\theta_i$ 为模型中第i个变量的系数, $w_{ij}$  为第i个变量第j个分箱WOE值, 影响每个分箱分数的因素：参数B, 系数 $\theta_i$ , 该分箱WOE编码 $w_{ij}$

# 模型洞察

Thinking: 如何对决策树模型进行解读?

```
--- age <= 29.50
|--- family_members <= 2.50
|   |--- policy_start_date_day <= 703.00
|   |   |--- age <= 19.50
|   |   |   |--- class: 0
|   |   |--- age > 19.50
|   |   |   |--- class: 1
|   |--- policy_start_date_day > 703.00
|   |   |--- age <= 21.50
|   |   |   |--- class: 0
|   |   |--- age > 21.50
|   |   |   |--- class: 0
|--- family_members > 2.50
|   |--- marital_status <= 1.50
|   |   |--- occupation <= 2.50
|   |   |   |--- class: 1
|   |   |--- occupation > 2.50
|   |   |   |--- class: 0
|   |--- marital_status > 1.50
|   |   |--- policy_end_date_day <= 8918.50
|   |   |   |--- class: 0
|   |   |--- policy_end_date_day > 8918.50
|   |   |   |--- class: 1
|--- age > 29.50
|--- age <= 60.50
|   |--- policy_end_date_day <= 5506.00
|   |   |--- class: 0
|   |--- policy_end_date_day > 5506.00
|   |   |--- occupation <= 3.50
|   |   |   |--- class: 1
|   |   |--- occupation > 3.50
|   |   |   |--- class: 1
|--- age > 60.50
|   |--- family_members <= 3.50
|   |   |--- education_level <= 0.50
|   |   |   |--- class: 0
|   |   |--- education_level > 0.50
|   |   |   |--- class: 1
|   |--- family_members > 3.50
|   |   |--- marital_status <= 1.50
|   |   |   |--- class: 1
|   |   |--- marital_status > 1.50
|   |   |   |--- class: 0
```

# 模型洞察

我在做寿险续保预测，构建了决策树：

```
===
|--- age <= 29.50
| |--- family_members <= 2.50
| | |--- policy_start_date_day <= 703.00
| | | |--- age <= 19.50
| | | | |--- class: 0
| | | |--- age > 19.50
| | | | |--- class: 1
| | |--- policy_start_date_day > 703.00
| | | |--- age <= 21.50
| | | | |--- class: 0
| | | |--- age > 21.50
| | | | |--- class: 0
| |--- family_members > 2.50
| | |--- marital_status <= 1.50
| | | |--- occupation <= 2.50
| | | | |--- class: 1
| | | |--- occupation > 2.50
| | | | |--- class: 0
| | |--- marital_status > 1.50
```

```
| | | |--- policy_end_date_day <= 8918.50
| | | | |--- class: 0
| | | |--- policy_end_date_day > 8918.50
| | | | |--- class: 1
|--- age > 29.50
| |--- age <= 60.50
| | |--- policy_end_date_day <= 5506.00
| | | |--- class: 0
| | |--- policy_end_date_day > 5506.00
| | | |--- occupation <= 3.50
| | | | |--- class: 1
| | | |--- occupation > 3.50
| | | | |--- class: 1
| |--- age > 60.50
| | |--- family_members <= 3.50
| | | |--- education_level <= 0.50
| | | | |--- class: 0
| | | |--- education_level > 0.50
| | | | |--- class: 1
| | |--- family_members > 3.50
| | | |--- marital_status <= 1.50
```

```
| | | | |--- class: 1
| | | |--- marital_status > 1.50
| | | | |--- class: 0
===
policy_start_date_day: 距离 2010-01-01的天数
policy_end_date_day: 距离 2010-01-01的天数
class: 0代表不续保, 1代表续保
marital_status: '单身': 0, '已婚': 1, '离异': 2
occupation: 0: 医生, 1: 工程师, 2: 律师, 3: 经理, 4: 设计师, 5: 销售
education_level: '高中': 0, '本科': 1, '硕士': 2, '博士': 3
===
帮我解释这个决策树，告诉我寿险客户续保 有哪些特点
```

# 模型洞察

---

这个决策树模型根据一系列的条件来预测寿险客户是否会续保。让我们逐层分析，看看哪些因素对续保决策有影响。

## 1. 年龄 (age)

- 年龄小于等于29.5岁的客户：
  - 家庭成员数量(family\_members)少于或等于2.5时：
    - 如果保险开始日期(policy\_start\_date\_day)距离2010-01-01不超过703天（约2年），那么更年轻的客户( $\leq 19.5$ 岁)倾向于不续保；而较年长的客户( $> 19.5$ 岁)则倾向于续保。
    - 如果保险开始日期超过703天，那么更年轻的客户( $\leq 21.5$ 岁)和稍微年长一点的客户( $> 21.5$ 岁)都倾向于不续保。
  - 当家庭成员数大于2.5时：
    - 婚姻状况(marital\_status)为单身或已婚( $\leq 1.5$ )时，职业(occupation)为医生、工程师或律师( $\leq 2.5$ )的人倾向于续保，而其他职业则倾向于不续保。
    - 对于婚姻状况为离异( $> 1.5$ )的客户，如果保险结束日期(policy\_end\_date\_day)距离2010-01-01不超过8918.5天（约24.5年）则倾向于不续保；否则，倾向于续保。
- 年龄大于29.5岁的客户：
  - 年龄在60.5岁以下时，若保险结束日期早于或等于5506天（约15.1年），则倾向于不续保；否则，无论职业如何，都倾向于续保。
  - 年龄超过60.5岁时，家庭成员较少( $\leq 3.5$ )且教育水平较低( $\leq 0.5$ , 高中)的客户倾向于不续保；而教育水平较高( $> 0.5$ )的客户则倾向于续保。对于家庭成员较多( $> 3.5$ )的客户，如果是单身或已婚( $\leq 1.5$ )，则倾向于续保；反之，则倾向于不续保。



# 模型洞察

---

从上述决策树可以看出，影响续保的因素包括但不限于：

- **年龄**：年轻客户与老年客户的续保倾向有所不同。
- **家庭成员数量**：家庭规模大小会影响续保决策。
- **婚姻状态**：不同婚姻状态下的个人续保意愿存在差异。
- **职业**：某些职业可能比其他职业更有可能续保。
- **教育水平**：受教育程度较高的个体可能更倾向于续保。
- **保险起始/结束日期**：保险合同的时间长度也对续保决定有一定影响。

这些特征共同作用，帮助模型判断客户是否具有较高的续保可能性。请注意，这只是一个基于给定数据集的统计模型，实际应用中还需要考虑更多因素和个人情况。

# 模型洞察

在这个决策树中：年龄、家庭成员数量、婚姻状态、职业、教育水平、保险起始/结束日期 如何影响续保可能性，给我一些洞察结论

基于您提供的决策树，我们可以得出以下关于不同特征如何影响寿险客户续保可能性的洞察结论：

## 1. 年龄 (Age)

- 年轻客户（特别是29.5岁以下）与年长客户（尤其是60.5岁以上）在续保行为上表现出不同的模式。年轻客户中，家庭规模较小且保险开始日期较早的群体更可能不续保，而那些年龄稍大一些的年轻客户则更倾向于续保。
- 对于年龄较大的客户（超过60.5岁），如果他们的教育水平较高，那么他们更有可能续保；反之，则不太可能续保。

## 2. 家庭成员数量 (Family Members)

- 家庭成员较少的年轻人（ $\leq 2.5$ 人）通常在特定条件下（如保险开始日期较早）更可能选择不续保。
- 当家庭成员较多时，特别是在老年人群中（ $> 3.5$ 人），婚姻状况成为决定性因素之一。单身或已婚者更可能续保，而离异状态下的个人则倾向于不续保。

## 3. 婚姻状态 (Marital Status)

- 婚姻状况对续保的影响主要体现在家庭成员较多的老年群体中。单身或已婚人士比离异的人更可能续保。
- 在某些情况下，比如对于拥有较大家庭的年轻客户，婚姻状况也会影响续保决策，但这种影响是通过与其他变量结合来体现的。

## 4. 职业 (Occupation)

- 职业类型对年轻客户的续保意愿有显著影响。医生、工程师和律师等职业的年轻客户更倾向于续保，而其他职业类型的客户则相反。
- 对于年龄介于29.5至60.5岁之间的客户，职业不再是一个区分是否续保的关键因素。

## 5. 教育水平 (Education Level)

- 教育水平仅在老年客户群体中被用作区分续保与否的标准。高学历的老年人更可能续保。

## 6. 保险起始/结束日期 (Policy Start/End Date)

- 保险合同的时间长度对续保决策有重要影响。例如，保险开始日期较早（距离2010-01-01不超过703天）的年轻客户更可能选择不续保。
- 对于年龄较大的客户，如果保险结束日期远（距离2010-01-01超过8918.5天），则更可能续保。

# 模型洞察

## 综合来看，给我续保用户的特点

根据决策树分析，我们可以总结出以下续保用户的一些特点：

### 1. 年龄：

- 年轻客户（小于等于29.5岁）中，年龄稍大一些的客户（大于19.5岁）更可能续保。
- 对于年龄较大的客户（超过29.5岁），尤其是那些年龄在60.5岁以下且保险结束日期较晚的客户，倾向于续保。

### 2. 家庭成员数量：

- 家庭成员较多的老年客户（大于3.5人）如果教育水平较高，则更可能续保。
- 年轻客户中，家庭成员较少的情况下，保险开始日期较早的客户不太可能续保；而家庭成员较多的老年客户如果是单身或已婚状态，更可能续保。

### 3. 婚姻状态：

- 单身或已婚状态下的年轻和老年客户更可能续保。
- 离异状态下的老年客户不太可能续保。

### 4. 职业：

- 职业为医生、工程师或律师的年轻客户更可能续保。
- 对于年龄介于29.5至60.5岁的客户，职业类型不是续保的关键因素。
- 但对于年龄大于60.5岁的客户，无论职业如何，都倾向于续保。

# 模型洞察

---

## 5. 教育水平：

- 教育水平较高的老年客户更可能续保。

## 6. 保险起始/结束日期：

- 保险开始日期较晚（距离2010-01-01超过703天）的年轻客户更可能续保。
- 保险结束日期较晚（对于年龄较大的客户来说，距离2010-01-01超过8918.5天）的客户更可能续保。
- 对于年龄在60.5岁以下的客户，保险结束日期较晚（距离2010-01-01超过5506天）也增加了续保的可能性。

综上所述，续保用户的典型特征可以概括为：

- 年龄方面，中等年龄段（比如29.5岁以上但未达到老年）的客户以及特定条件下的年轻客户（年龄略大）。
- 拥有较大或适中的家庭规模。
- 婚姻状态稳定，如单身或已婚。
- 在某些年龄段内，从事专业性较强的职业（如医生、工程师、律师）。
- 教育背景较好，尤其是在老年群体中。
- 保险合同的有效期较长，特别是保险结束日期较远的客户。

这些特征反映了续保用户通常具有一定的财务稳定性、对未来规划的关注以及对保险产品价值的认可。保险公司可以根据这些特点来优化其产品和服务，以提高客户的续保率。



Thank You  
Using data to solve problems

