生存分析（Survival Analysis）是统计学中用于分析“直到某个事件发生所需时间”的一种方法。如分析患者的生存时间，在医学、工程等领域都有广泛的应用。

要在一堆的数据中进行生存分析，核心的数据有三个：**事件、生存时间、删失。**其中事件指的是研究者关心发生的事件；生存事件指从设定的起始点到事件发生的时间；删失指的是在研究结束时事件仍未发生、或数据丢失等删失数据。

在这个 project 当中，主要介绍了三种生存分析的模型，分别为 Kaplan Meier 模型、Cox Proportional Hazards 模型 以及 Accelerated Failur Time 模型，再后文将一一分析。

本篇 report 主要介绍如何将参考代码改写到 Jupyter notebook 当中，以及解析各个模型的代码

一、 数据的读入

区别于在 databrick 上的数据读入，由于样例数据的样本量较小，只需要将文件读入到 pyspark 的 dataframe 当中，再对数据进行预处理

```
#在jupyter上进行生存分析，因没有databrick环境，且样例数据较小，直接将数据读取成dataframe储存在内存当中
schema = StructType([
  StructField('customerID', StringType()),
  StructField('gender', StringType()),
  StructField('seniorCitizen', DoubleType()),
  StructField('partner', StringType()),
  StructField('dependents', StringType()),
  StructField('tenure', DoubleType()),
  StructField('phoneService', StringType()),
  StructField('multipleLines', StringType()),
  StructField('InternetService', StringType()),
  StructField('onlineSecurity', StringType()),
  StructField('onlineBackup', StringType()),
  StructField('deviceProtection', StringType()),
  StructField('techSupport', StringType()),
  StructField('streamingTV', StringType()),
  StructField('streamingMovies', StringType()),
  StructField('Contract', StringType()),
  StructField('paperlessBilling', StringType()),
  StructField('paymentMethod', StringType()),
  StructField('monthlyCharges', DoubleType()),
  StructField('totalCharges', DoubleType()),
  StructField('Churn', StringType())
  ])
```

```
#将数据存入bronze层（原始数据）和silver层（处理后的结构化数据
#jupyter可能会区分大小写，严格要求符合数据中的列名，所以要进行列名的修改
bronze_df = spark.read.format('csv').schema(schema).option('header','true').load(file_path).withColumnRenamed("InternetService", "internetService") \
            .withColumnRenamed("Contract", "contract") \
            .withColumnRenamed("Churn", "churnString")
silver_df = bronze_df.withColumn('churn',
                   when(col('churnString') == 'Yes', 1)
                  .when(col('churnString') == 'No', 0)
                  .otherwise('Unknown')) \
            .drop('churnString') \
            .filter((col('contract') == 'Month-to-month') &
                    (col('internetService') != 'No'))
```

其中的 churn 列是我们将要带入模型的事件，它代表客户的流失。将事件列的值标记为二值变量，区分哪些数据是**真实事件**（**1**），哪些是**删失数据**（**0**）。此外过滤了 **contract='Month-to-month'** 和 **internetService!='No'** 的客户，这

意味着分析结果仅适用于这部分人群，不可泛化到其他客户群体。

```
telco_pd  =silver_df.toPandas()
```

将 PySpark DataFrame (**silver_df**) 转换为 Pandas DataFrame (**telco_pd**)，以方便使用 Python 生态的生存分析工具（如 **lifelines** 库）。

二、 Kaplan-Meier 模型

```
from lifelines import KaplanMeierFitter
from lifelines.utils import median_survival_times## 计算中位生存时间
from lifelines.statistics import pairwise_logrank_test## 对数秩检验（组间比较）
```

导入 **lifelines** 库中的 **Kaplan-Meier 生存曲线、中位生存时间计算** 和 **组间对数秩检验（Log-rank test）。**

其中 **Kaplan-Meier 生存曲线**是估计生存函数：计算在任意时间点 t 的生存概率 S(t)（即“个体在时间 t 之后仍未发生事件的概率”）并且能够有效利用未发生事件的样本（如客户未流失、患者未死亡）的信息，避免低估生存率。

生存概率 $S(t)$ 的乘积形式：

$$S(t) = \prod_{t_i \le t} \left(1 - \frac{d_i}{n_i}\right)$$

- d_i：时间 t_i 发生事件的人数。
- n_i：时间 t_i 之前仍处于风险中的人数（包括删失）。

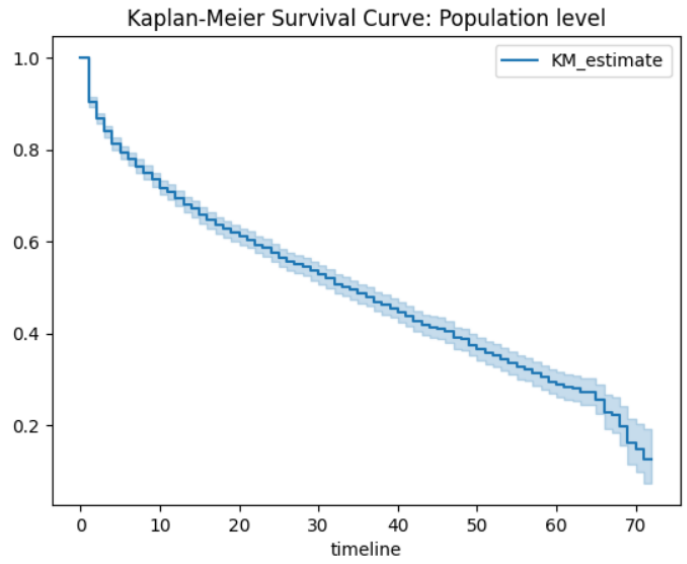
中位生存时间计算是找到生存概率降至 50%时对应的时间点，比平均值更适合

组间对数秩检验是检验不同组的生存曲线是否显著不同。不假设生存时间分布，仅比较事件发生的时序差异。

```
kmf = KaplanMeierFitter()#调用卡普兰-迈耶模型
T=telco_pd['tenure']#时间列
C=telco_pd['churn'].astype(float)#事件列
kmf.fit(T,C)
kmf.plot(title='Kaplan-Meier Survival Curve: Population level')
kmf.median_survival_time_#计算中位生存时间
```

在数据中的时间列是’tenure’列，事件列是处理过后的‘churn’列，将它们单独取出来，用输入进 Kaplan-Meier 模型当中，绘图并计算中位生存时间

```
np.float64(34.0)
```




```python
def plot_km(col):
    ax = plt.subplot(111)
    for r in telco_pd[col].unique():# 遍历该列的所有唯一值（如性别中的Male/Female）
    ix = telco_pd[col] == r # 获取当前分组（r）的布尔索引
    kmf.fit(T[ix], C[ix],label=r)# 拟合当前组的生存曲线
    kmf.plot(ax=ax)
```

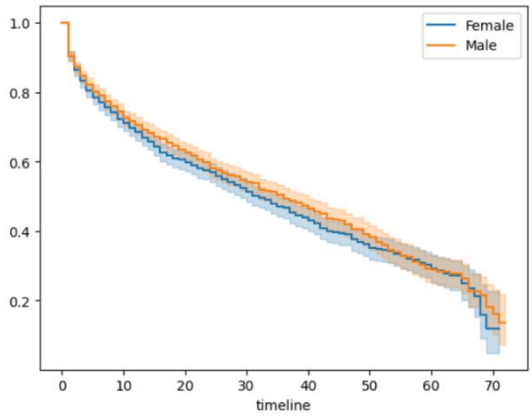
plot_km(col)函数用来绘制 KM 生存曲线，根据输入的 col 列数据进行分组，将不同组的生存概率绘制到同一张图上并进行比较。

```python
def print_logrank(col):#返回各组之间的对数秩检验结果（统计量、P值等），用于判断组间生存曲线是否显著不同。
  log_rank = pairwise_logrank_test(telco_pd['tenure'], telco_pd[col], telco_pd['churn'])
  return log_rank.summary
```

- **print_logrank(col) 函数**执行对数秩检验（Log-rank test），判断不同（由 col 列定义）的生存曲线是否存在显著差异。

例如比较 gender 列当中不同组的生存曲线：

```
plot_km('gender')
print_logrank('gender')
```



		test_statistic	p	-log2(p)
Female	**Male**	1.61011	0.204476	2.289995

- 其中数据表示：
- **test_statistic**：Log-rank 检验统计量（值越大，组间差异越显著）。
- **p**：p 值（若 **<0.05**，则拒绝原假设，认为组间生存曲线不同）。
- **-log2(p)**：p 值的对数变换（用于直观比较显著性）

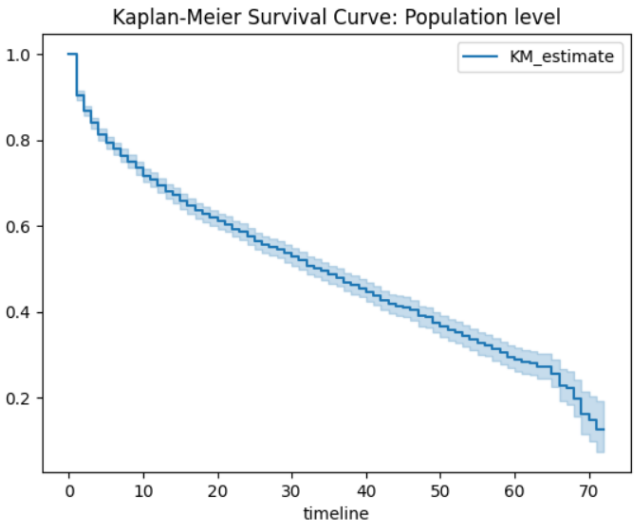
```
def get_survival_probs(col,val):#计算并返回指定分组（基于列 col 的特定值 val）的生存概率模型。
  ix = telco_pd[col] == val
  return kmf.fit(T[ix],C[ix],label=val)
```

get_survival_probs(col,val)函数用于计算特定列指定分组的生存模型

```
sp_internet_dsl = get_survival_probs('internetService','DSL')
pd.DataFrame(sp_internet_dsl.survival_function_at_times(range(0,10)))
```

如对 internet 列的 dsl 组进行生存分析，得到结果：

	DSL
0	1.000000
1	0.902698
2	0.864380
3	0.834702
4	0.810522
5	0.794352
6	0.783900
7	0.776362
8	0.768486
9	0.750833



三、 Cox Proportional Hazards 模型

```python
import pandas as pd
import numpy as np
from lifelines.fitters.coxph_fitter import CoxPHFitter
from lifelines.statistics import proportional_hazard_test
from lifelines import KaplanMeierFitter
```

导入 lifelines 库当中 **Cox 比例风险模型、比例风险假设检验、Kaplan-Meier 估计**

分析流程：

1、使用 **pandas** 加载数据，确保包含：生存时间列（如 **tenure**）、事件列（如 **churn**，1=事件发生，0=删失）、协变量列（如 **age**, **treatment**）

2、用 **Kaplan-Meier** 进行描述性分析

3、**Cox 模型**：探究变量对生存时间的影响。

Cox 比例风险模型作用：建立生存时间与协变量之间的关系，分析哪些因素（如年龄、治疗方式）会影响事件（如死亡、客户流失）发生的风险。属于半参数模型，不假设生存时间的具体分布。

- 模型公式：

$$h(t|X) = h_0(t)\exp(\beta_1 X_1 + \beta_2 X_2 + \dots)$$

  - $h(t|X)$：在协变量 X 下的风险函数。
  - $h_0(t)$：基线风险（所有协变量为0时的风险）。
  - $\exp(\beta)$：**风险比（Hazard Ratio, HR）**，解释协变量的影响。

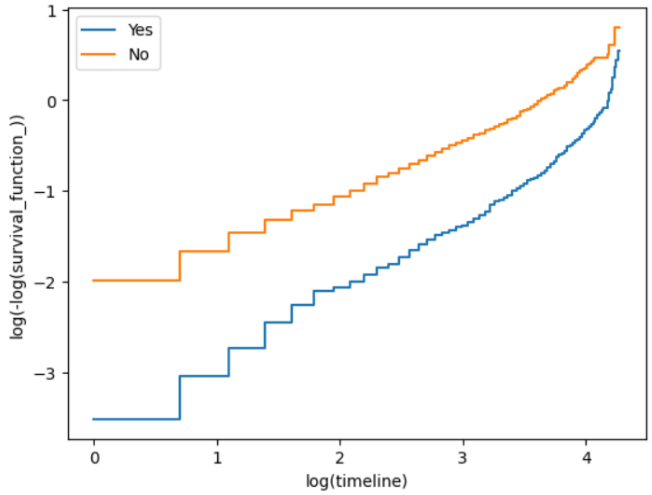
`proportional_hazard_test`：验证 Cox 模型的核心假设--协变量的风险比是否随时间恒定（即比例风险假设）。如果假设不成立，Cox 模型的结果可能不可靠。

检验方法

- 基于 **Schoenfeld残差** 的检验：
  - 原假设（H_0）：风险比恒定（假设成立）。
  - 若 `p < 0.05`，拒绝H_0，认为假设不成立。

```
kmf = KaplanMeierFitter()

T=telco_pd['tenure'] #时间列
C=telco_pd['churn'].astype(float) #事件列

kmf.fit(T,C)
import matplotlib.pyplot as plt
def plot_km_loglog(col):
  ax = plt.subplot(111)
  for r in telco_pd[col].unique():#按分不同事件进行绘图
    ix = telco_pd[col] == r
    kmf.fit(T[ix], C[ix],label=r)
    kmf.plot_loglogs(ax=ax)


plot_km_loglog('onlineBackup')
```

先用**使用 Kaplan-Meier 方法**绘制**对数-对数生存曲线（log-log survival plot）**，通过分组比较曲线，帮助验证 Cox 模型的适用性。如果曲线是平行的，则风险比例恒定，Cox 模型可直接应用；如果曲线不平行，则需进一步分析。如验证'onlineBackup'列，绘制图像：



可以观察到在一部分时间段中两条曲线是平行的，在这些时间段里风险比例恒定。

```
#对感兴趣的列进行编码
encode_cols = ['dependents','internetService','onlineBackup','techSupport','paperlessBilling']
#对数据框 telco_pd 中的分类变量进行 独热编码
encoded_pd = pd.get_dummies(telco_pd,
              columns=encode_cols,
              prefix=encode_cols,# 新列名前缀（使用原列名）
              drop_first=False)

encoded_pd.head()
survival_pd = encoded_pd[['churn','tenure','dependents_Yes','internetService_DSL','onlineBackup_Yes','techSupport_Yes']]
#将churn列转换为float类型
survival_pd.loc[:,'churn'] = survival_pd.loc[:,'churn'].astype('float')
```

在 Cox 比例风险模型分析前对感兴趣的分类变量进行**独热编码预处理，**将分类变量转换为二进制，解决多分类变量的参照组问题。（如 internetService_DSL，1=DSL，0=其他）

```
#将数据带入进cox模型
cph = CoxPHFitter(alpha=0.05)
cph.fit(survival_pd, 'tenure', 'churn')
#输出模型摘要
cph.print_summary()
```

再将处理后的数据输入 cox 模型当中，对每一个分类变量进行分析得到结果：

model	lifelines.CoxPHFitter
duration col	'tenure'
event col	'churn'
baseline estimation	breslow
number of observations	3351
number of events observed	1556
partial log-likelihood	-11315.95
time fit was run	2025-04-13 12:13:36 UTC

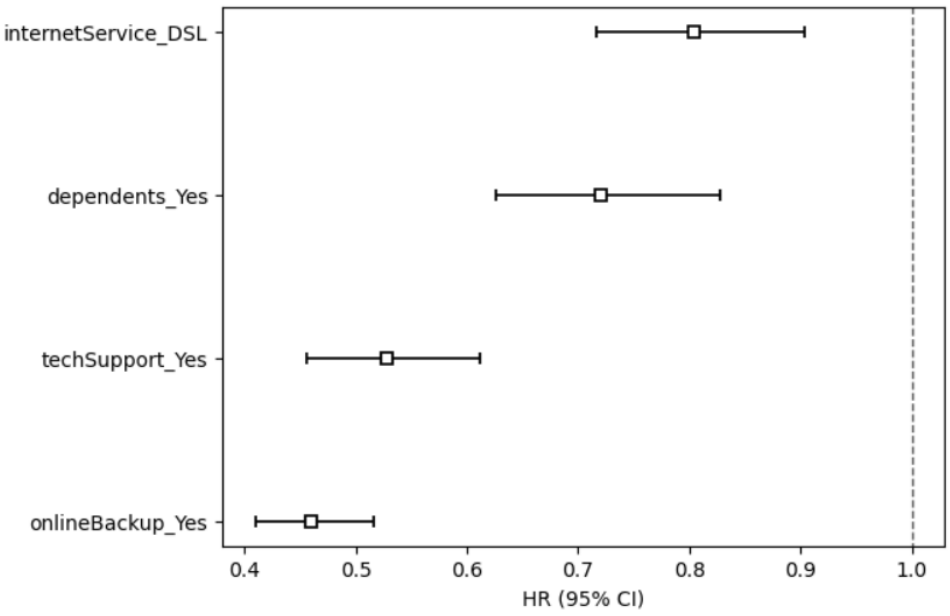
这是模型对数据的总体概括

	coef	exp(coef)	se(coef)	coef lower 95%	coef upper 95%	exp(coef) lower 95%	exp(coef) upper 95%	cmp to	z	p	-log2(p)
dependents_Yes	-0.33	0.72	0.07	-0.47	-0.19	0.63	0.83	0.00	-4.64	<0.005	18.12
internetService_DSL	-0.22	0.80	0.06	-0.33	-0.10	0.72	0.90	0.00	-3.68	<0.005	12.07
onlineBackup_Yes	-0.78	0.46	0.06	-0.89	-0.66	0.41	0.52	0.00	-13.13	<0.005	128.37
techSupport_Yes	-0.64	0.53	0.08	-0.79	-0.49	0.46	0.61	0.00	-8.48	<0.005	55.36

这是各个分类组的输出结果，其中列名 **coef** 表示回归系数，**exp(coef)**表示两组的风险比，**coef lower/upper 95%**是回归系数的置信区间，**exp(coef) lower/upper 95%**是风险比的置信区间

```
#绘制风险比
cph.plot(hazard_ratios=True)
```

得到结果：



直观展示协变量对风险的影响，若区间**不包含 1**，则影响显著。**HR>1**：增加风

险。**HR<1**：降低风险。

```
#检验比例风险假设是否成立
cph.check_assumptions(survival_pd,p_value_threshold=0.05)
```

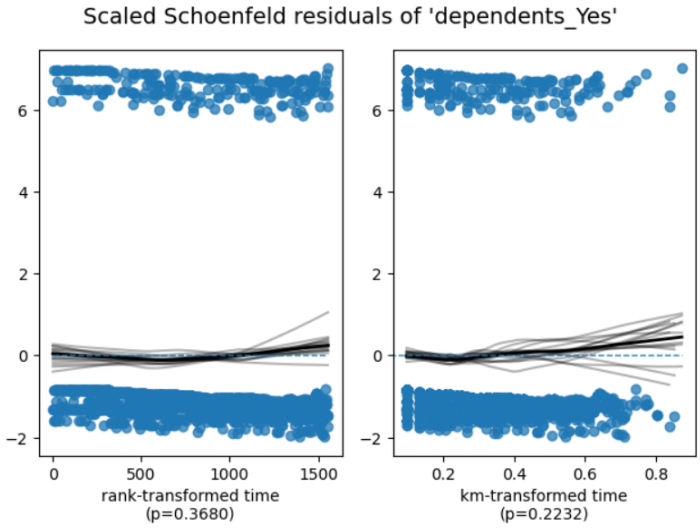
检验比例风险假设是否成立，输出结果：

		test_statistic	p	-log2(p)
dependents_Yes	**km**	1.48	0.22	2.16
	rank	0.81	0.37	1.44
internetService_DSL	**km**	20.98	<0.005	17.72
	rank	26.71	<0.005	22.01
onlineBackup_Yes	**km**	17.80	<0.005	15.31
	rank	17.47	<0.005	15.07
techSupport_Yes	**km**	8.09	<0.005	7.81
	rank	13.76	<0.005	12.23

由图可得变量 internetService_DSL、onlineBackup_Yes 、techSupport_Yes 违背了 non-proportional test

```
# 检验比例风险假设是否成立并绘图
cph.check_assumptions(survival_pd,p_value_threshold=0.05,show_plots=True)
```

与前面的代码相同，只是多了数据的绘图分析。以‘dependent_Yes’为例：



可以看到数据较为平行，不违反比例风险假设。

四、 Accelerated Failure Time 模型

```
from lifelines import WeibullAFTFitter,LogNormalAFTFitter,LogLogisticAFTFitter
from lifelines.fitters.coxph_fitter import CoxPHFitter
from lifelines.statistics import proportional_hazard_test
```

从 lifeline 中引入 **WeibullAFTFitter、LogNormalAFTFitter、LogLogisticAFTFitter 模型**

AFT 模型（加速失效时间模型，Accelerated Failure Time Model） 是生存分析中的一种参数化模型，用于研究协变量如何直接“加速”或“减速”事件发生的时间。

- **模型形式**：

$$\log(T) = \beta X + \sigma\epsilon$$

  - T：生存时间（如客户留存月数、设备故障时间）。
  - β：回归系数，解释为**时间比（Time Ratio, TR）**（`exp(coef)`）。
    - **TR > 1**：协变量延长生存时间（如治疗措施延长患者生存期）。
    - **TR < 1**：协变量缩短生存时间（如高负荷加速设备故障）。
  - ϵ：误差项（分布决定模型类型，如Weibull、对数正态等）。

分析流程：

1. 数据准备

2．选择分布类型

根据风险函数的可能形状选择模型：**WeibullAFTFitter**：风险单调递增/递减（适合设备故障）。**LogNormalAFTFitter**：风险先升后降（适合疾病复发）。**LogLogisticAFTFitter**：风险灵活变化（适合客户流失）。

3. 拟合模型

4. 解释结果

```
#对数据框 telco_pd 中的分类变量进行 独热编码
encode_cols = ['partner','multipleLines','internetService','onlineSecurity', 'onlineBackup','deviceProtection','techSupport','paymentMethod']

encoded_pd = pd.get_dummies(telco_pd,
               columns=encode_cols,
               prefix=encode_cols,
               drop_first=False)

encoded_pd.head()
survival_pd = encoded_pd[['churn','tenure','partner_Yes', 'multipleLines_Yes', \
                          'internetService_DSL','onlineSecurity_Yes','onlineBackup_Yes','deviceProtection_Yes','techSupport_Yes',\
                          'paymentMethod_Bank transfer (automatic)','paymentMethod_Credit card (automatic)' ]]
```

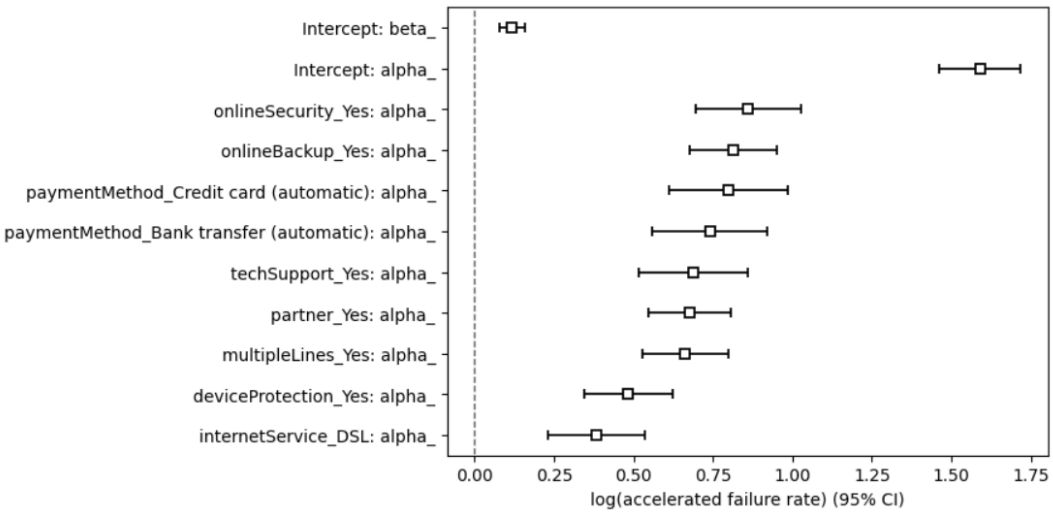
对数据框 telco_pd 中的分类变量进行 独热编码

```
#带入aft模型
aft = LogLogisticAFTFitter()
aft.fit(survival_pd, duration_col='tenure', event_col='churn')
# 将log化的数据处理，输出正确数据
print("Median Survival Time:{:.2f}".format(np.exp(aft.median_survival_time_)))
#评估模型结果并绘图
aft.print_summary()
aft.plot()
```

将数据带入到 aft 模型当中，其中 **Log-Logistic 分布特点**为风险函数可呈现**非单调性**（如先升后降），适合客户流失、疾病复发等场景。之后**计算并输出中位生存时间。**输出模型统计摘要，最后可视化协变量影响

得到结果:

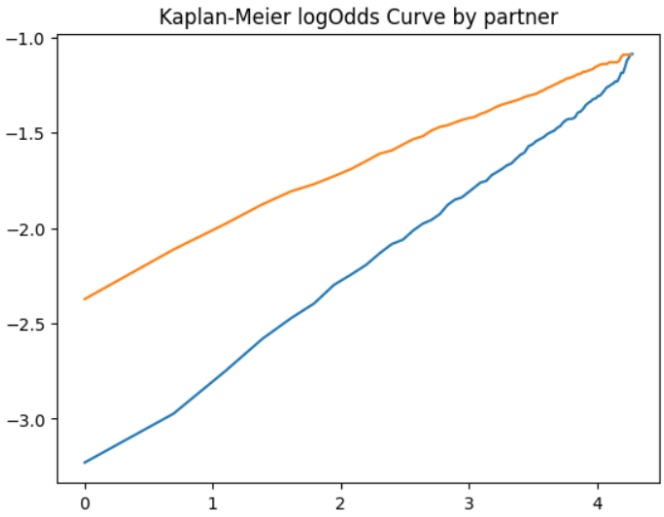
		coef	exp(coef)	se(coef)	coef lower 95%	coef upper 95%	exp(coef) lower 95%	exp(coef) upper 95%	cmp to	z	p	-log2(p)
alpha_	deviceProtection_Yes	0.48	1.62	0.07	0.35	0.62	1.41	1.86	0.00	6.88	<0.005	37.25
	internetService_DSL	0.38	1.47	0.08	0.23	0.53	1.26	1.71	0.00	4.98	<0.005	20.59
	multipleLines_Yes	0.66	1.94	0.07	0.53	0.80	1.70	2.22	0.00	9.64	<0.005	70.70
	onlineBackup_Yes	0.81	2.25	0.07	0.68	0.95	1.97	2.59	0.00	11.63	<0.005	101.50
	onlineSecurity_Yes	0.86	2.37	0.09	0.69	1.03	2.00	2.80	0.00	10.12	<0.005	77.60
	partner_Yes	0.68	1.97	0.07	0.55	0.81	1.73	2.24	0.00	10.21	<0.005	78.93
	paymentMethod_Bank transfer (automatic)	0.74	2.10	0.09	0.56	0.92	1.75	2.51	0.00	8.05	<0.005	50.07
	paymentMethod_Credit card (automatic)	0.80	2.22	0.10	0.61	0.99	1.84	2.68	0.00	8.36	<0.005	53.81
	techSupport_Yes	0.69	1.99	0.09	0.52	0.86	1.68	2.36	0.00	7.90	<0.005	48.37
	Intercept	1.59	4.91	0.07	1.46	1.72	4.32	5.58	0.00	24.47	<0.005	436.88
beta_	Intercept	0.12	1.13	0.02	0.08	0.16	1.08	1.17	0.00	5.71	<0.005	26.42



结果分析与先前提到的模型结果类似。

```
#Kaplan-Meier logOdds Curve用来检验AFT模型的两个假设：该模型是否遵循Proportional Odds假设、该模型是否有合适的特定分布
#如果图像中的线段平行，则假设1成立；如果图像都是笔直的，则假设2成立
from lifelines import KaplanMeierFitter
kmf = KaplanMeierFitter()

T=telco_pd['tenure'] #duration
C=telco_pd['churn'].astype(float) #event observed

kmf.fit(T,C)
import matplotlib.pyplot as plt

def plot_km_logOdds(col):
  ax = plt.subplot(111)
  for r in telco_pd[col].unique():
    ix = telco_pd[col] == r
    kmf.fit(T[ix], C[ix],label=r)
    sf = kmf.survival_function_
    sf['failureOdds'] = (np.log(1-sf))/sf
    sf['logTime'] = np.log(sf.index)
    plt.title(f'Kaplan-Meier logOdds Curve by {col}')
    plt.plot(sf['logTime'],sf['failureOdds'])
plot_km_logOdds('partner')
```

用 Kaplan-Meier logOdds Curve 用来检验 AFT 模型的两个假设：该模型是否遵循 Proportional Odds 假设、该模型是否有合适的特定分布。如果图像中的线段平行，则假设 1 成立；如果图像都是笔直的。则假设 2 成立



以 partner 这组的数据为例，可以观察到该组数据并不满足假设。

五、实例分析

```
import pandas as pd
import seaborn as sns
from ipywidgets import widgets, interact
from lifelines.fitters.coxph_fitter import CoxPHFitter
encode_cols = ['dependents','internetService','onlineBackup','techSupport','paperlessBilling']

encoded_pd = pd.get_dummies(telco_pd,
                columns=encode_cols,
                prefix=encode_cols,
                drop_first=False)

encoded_pd.head()
survival_pd = encoded_pd[['churn','tenure','dependents_Yes','internetService_DSL','onlineBackup_
survival_pd.loc[:,'churn'] = survival_pd.loc[:,'churn'].astype('float')
```

1. 对分类变量(dependents, internetService 等)进行独热编码

选择关键特征：客户流失状态(churn)、在网时长(tenure)、是否有家属(dependents)、网络服务类型等

将流失状态转换为数值类型用于建模

```
cph = CoxPHFitter(alpha=0.05)
cph.fit(survival_pd, 'tenure', 'churn')
```

2 模型构建

使用 Cox 比例风险模型(CoxPHFitter)，设置显著性水平 α=0.05

以"tenure"(在网时长)作为时间变量、以"churn"(流失状态)作为事件指标

```
cols = ['dependents_Yes', 'internetService_DSL', 'onlineBackup_Yes', 'techSupport_Yes', 'partner

# 创建交互式控件
widgets_dict = {}

for col in cols:
    if col == 'internal_rate_of_return':
        widgets_dict[col] = widgets.FloatSlider(
            value=0.10,
            min=0.0,
            max=1.0,
            step=0.01,
            description='Internal Rate of Return:',
            continuous_update=False
        )
    else:
        widgets_dict[col] = widgets.Dropdown(
            options=[('No', 0), ('Yes', 1)],
            value=0,
            description=col.replace('_', ' ') + ':',
            disabled=False
        )
```

```
def get_widget_values():
    widget_dict = {col: widget.value for col, widget in widgets_dict.items()}
    return pd.DataFrame(widget_dict, index=[0])  # 返回单行 DataFrame
```

3.交互式分析

创建可调节参数控件，包括：

- 客户特征(家属、网络服务、技术支持等)
- 内部收益率(IRR)参数(0-100%)

实现动态计算功能：

- 生存概率预测
- 预期月利润计算
- 净现值(NPV)计算

```
def get_payback_df():
    # 获取控件值（替换原 Databricks 的 get_widget_values）
    df = get_widget_values()  # 使用前文定义的 Jupyter 版本

    # 提取 IRR 并转换为月利率
    irr = df['internal_rate_of_return'].astype('float64')[0] / 12  # 注意列名去掉了空格

    # 假设 cph 是已拟合的 CoxPHFitter 模型
    if 'cph' not in globals():
        raise ValueError("请先定义并拟合 CoxPHFitter 模型 (cph)")

    # 预测生存函数
    survival_prob = cph.predict_survival_function(df)
    cohort_df = pd.concat([
        pd.DataFrame([1.00]),
        round(survival_prob, 2)
    ]).rename(columns={0: 'Survival Probability'})
```

4.1 生存概率分析

展示客户随时间推移的留存概率曲线

反映客户特征对长期留存的影响

用于预测客户生命周期价值

```
    # 计算各项指标
    cohort_df['Contract Month'] = cohort_df.index.astype('int')
    cohort_df['Monthly Profit for the Selected Plan'] = 30  # 固定值，可改为控件输入
    cohort_df['Avg Expected Monthly Profit'] = round(
        cohort_df['Survival Probability'] * cohort_df['Monthly Profit for the Selected Plan'], 2
    )
    cohort_df['NPV of Avg Expected Monthly Profit'] = round(
        cohort_df['Avg Expected Monthly Profit'] / ((1 + irr) ** cohort_df['Contract Month']), 2
    )
    cohort_df['Cumulative NPV'] = cohort_df['NPV of Avg Expected Monthly Profit'].cumsum()
    cohort_df['Contract Month'] = cohort_df['Contract Month'] + 1

    return cohort_df[
        ['Contract Month', 'Survival Probability',
         'Monthly Profit for the Selected Plan',
         'Avg Expected Monthly Profit',
         'NPV of Avg Expected Monthly Profit',
         'Cumulative NPV']
    ].set_index('Contract Month')
```

4.2 财务价值分析

月度利润：假设每位留存客户每月产生$30 利润

预期月度利润：生存概率×月度利润

净现值(NPV)：考虑资金时间价值的利润折现

- 使用可调节的内部收益率(IRR)参数
- 计算公式：NPV = 预期利润 / (1+月 IRR)^月份

累计 NPV：客户生命周期总价值的现值

```
def plot_cumulative_npv():
    payback_df = get_payback_df()
    selected_data = payback_df.iloc[[11, 23, 35]]['Cumulative NPV']
    months = ['12 Months', '24 Months', '36 Months']

    plt.figure(figsize=(10, 6))
    ax = sns.barplot(x=months, y=selected_data.values, palette="Blues_d")
    ax.set_title('Cumulative NPV Over Time')
    ax.set_ylabel('Cumulative NPV ($)')

    for p in ax.patches:
        ax.annotate(f"${p.get_height():.2f}",
                    (p.get_x() + p.get_width() / 2., p.get_height()),
                    ha='center', va='center',
                    xytext=(0, 10),
                    textcoords='offset points')

    plt.show()
```

5.1 累计 NPV 柱状图

- 展示 12、24、36 个月的关键时间点累计价值
- 直观比较不同时期的投资回报
- 包含具体数值标注，便于精确分析

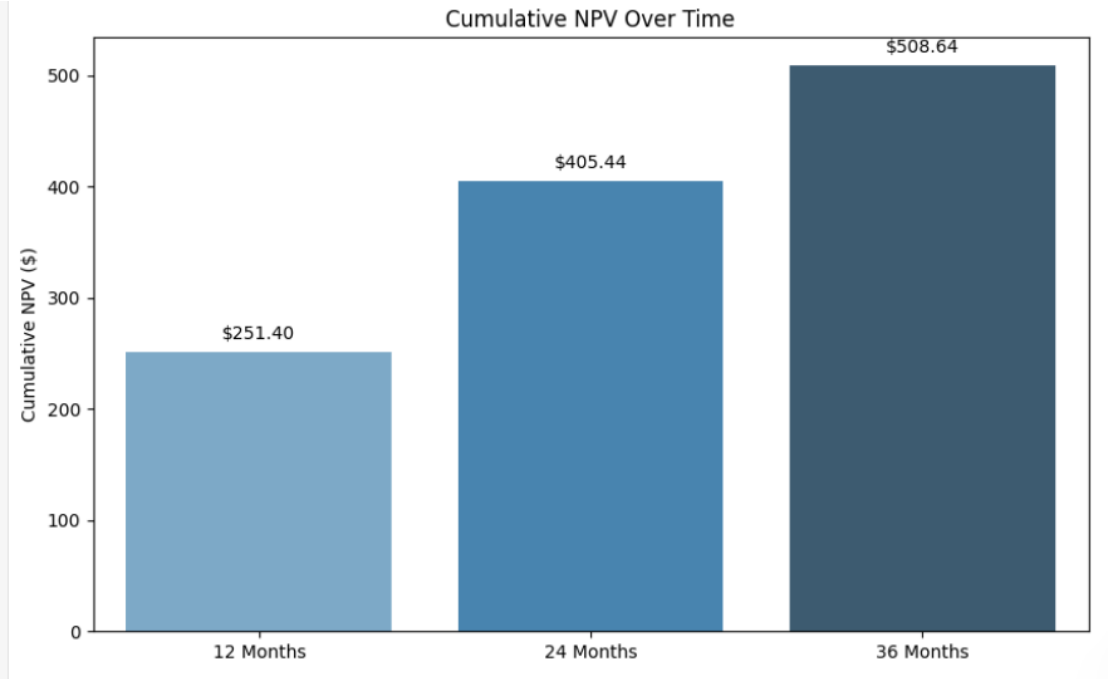
```
def plot_Survival_Probability():
    payback_df = get_payback_df()
    # 绘制折线图
    sns.lineplot(x=payback_df.index, y=payback_df['Survival Probability'])
    # 显示图形
    plt.show()
```

5.2 生存概率曲线图

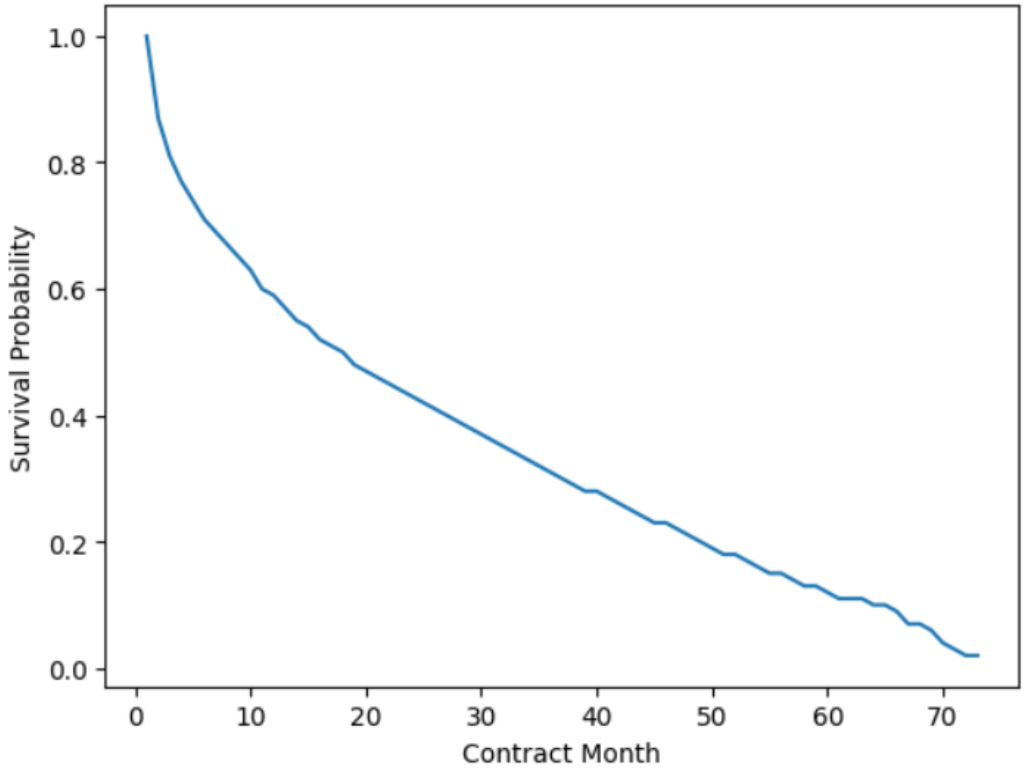
- 展示客户留存率随时间下降趋势
- 帮助识别客户流失高风险期
- 为客户保留策略提供时间参考

```
pd.options.display.max_rows = 25
display(get_payback_df()[0:25])

plot_cumulative_npv()

plot_Survival_Probability()
```

结果展示:



Contract Month	Survival Probability	Monthly Profit for the Selected Plan	Avg Expected Monthly Profit	NPV of Avg Expected Monthly Profit	Cumulative NPV
1	1.00	30	30.0	30.00	30.00
2	0.87	30	26.1	25.88	55.88
3	0.81	30	24.3	23.90	79.78
4	0.77	30	23.1	22.53	102.31
5	0.74	30	22.2	21.48	123.79
6	0.71	30	21.3	20.43	144.22
7	0.69	30	20.7	19.69	163.91
8	0.67	30	20.1	18.97	182.88
9	0.65	30	19.5	18.25	201.13
10	0.63	30	18.9	17.54	218.67
11	0.60	30	18.0	16.57	235.24
12	0.59	30	17.7	16.16	251.40
13	0.57	30	17.1	15.48	266.88
14	0.55	30	16.5	14.81	281.69



该实例展现了生存分析的实际应用价值：

1. **客户价值评估**：量化不同特征客户的生命周期价值
2. **营销决策支持**：

    - 确定可接受的客户获取成本(CAC)
    - 评估保留措施的投资回报率
3. **产品策略优化**:
    - 识别高价值特征组合(如有技术支持的客户)
    - 发现降低流失率的关键因素
4. **财务规划**:
    - 预测客户群产生的现金流
    - 考虑资金成本的投资回报分析