第十一章 生存分析和删失数据

生存分析 (survival analysis) 的研究对象是尚未发生的活动。例如，在为其五年的医学研究中，部分患者可能存活超过 5 年，我们希望研究这些未被保留的数据，它们称为**删失数据 (censored data)**，这些数据携带了有用的信息；或者，对于取消订阅的客户，在我们研究的时段内没有取消，我们不知道将来何时会取消，则这些客户取消订阅事件的数据也是删失的

11.1 生存时间和删失时间

对于每个研究对象，我们可以得到它们的**生存时间 (survival time)** 和 **删失时间 (censoring time)**，分别记为 T 和 C，其中删失时间又称为**失败时间 (failure time)** 或者 **事件时间 (event time)**。生存时间意味着此时我们感兴趣的事情发生了，删失时间则意味着此时数据丢失 (例如病人退出或未复发)

定义随机变量：

$$Y = \min(T, C)$$

为事件的发生，指示变量：

$$\delta = \begin{cases} 1, & if\ T \leq C \\ 0, & if T\ > C \end{cases}$$

现在我们可以获得观测变量对 (Y, δ)

11.2 删失的具体描述

为了分析删失数据，我们首先要对数据删失的原因进行假设。例如，患者提前退出研究的可能是因为它病得非常厉害 (若无此假设，可能高估生存时间)，男性的患者在重病时更倾向于退出研究 (若无此假设，可能错误分析性别影响)

生存分析的一个重要假设是删失机制与事件发生独立，即删失不会倾向于某些数据 (大体重被删失违反这一假设)。判断独立性主要取决于数据搜集的过程，后续研究假设独立性成立

本章重点介绍右删失，即 $T \geq Y$ 时发生删失，这保证删失发生时间至少与观测时间 Y 一样大。左删失和区间删失的思想同上

11.3 The Kaplan–Meier 生存曲线

生存曲线 (survival curve 或 **生存函数 (survival function)** 的定义如下：

$$S(t) = \Pr(T > t)$$

这是个递减函数，量化了在时间 t 时幸存下来的概率。$S(t)$ 的值越大，表明 t 之前发生事件的概率越小

用 `BrainCancer` 数据集举例，当我们试图估计 $S(20)$ 时，我们计算 $t=20$ 时仍然存活患者的比例。然而，我们若用下面两种方式来处理删失患者：

1. 将删失患者认为不存活。可能低估了生存率
2. 只考虑非删失患者，即缩小分母。没有用上删失患者的信息

采用下面的方法处理删失数据。数据集中，在观测时间内死亡的人数记为 K，则对于所有的死者，它们的唯一死亡时间点为 $d_1<d_2<\cdots<d_k$，记录 q_k 为第 k 个时间点时的死亡人数。对于 $k=1,\ldots,K$，令 r_k 表示在某个死亡时间点 d_k 前存在于研究中的人数，它们称为风险患者，风险患者的集合被称为**风险集 (risk set)**

根据总概率定律：

$$\begin{aligned}\Pr(T>d_k)=\Pr(T>d_k|T>d_{k-1})\Pr(T>d_{k-1})+\\ \Pr(T>d_k|T\le d_{k-1})\Pr(T\le d_{k-1})\end{aligned}$$

而事件 $\{T>d_k|T\le d_{k-1}\}$ 不可能发生，因此：

$$S(d_K)=\Pr(T>d_k)=\Pr(T>d_k|T>d_{k-1})\Pr(T>d_{k-1})$$

代入公式 $S(t)=\Pr(T>t)$，我们有：

$$S(d_K)=\Pr(T>d_k|T>d_{k-1})S(d_{k-1})$$

这意味着：

$$S(d_k)=\Pr(T>d_k|T>d_{k-1})\ldots\Pr(T>d_2|T>d_1)\Pr(T>d_1)$$

为了估计每一项的值，我们有：

$$\hat{\Pr}(T>d_j|T>d_{j-1})=(r_j-q_j)/r_j$$

这表示在时间点 d_j 后，幸存人数的比例，则**Kaplan-Meier 估计器 (Kaplan–Meier estimator)** 估计的生存曲线如下：

$$\hat{\mathrm{S}}(d_k)=\prod_{j=1}^{k}\left(\frac{r_j-q_j}{r_j}\right)$$

对于在两个死亡时间点 d_k,d_{k+1} 中的时间 t，我们令 $\hat{S}(t)=\hat{S}(d_k)$，因此 Kaplan-Meier 生存曲线呈阶梯状

11.4 对数秩检验

为了检验存在删失数据的两组生存曲线的风险是否有显著差异 (不能用平均值比较)，我们引入**对数秩检验 (log-rank test)**，它是一种按时间顺序检验事件的方法

设两组样本在死亡时间 d_k 前的未死亡且在试验中患者数分别为 r_{k1},r_{k2}，d_k 时的死亡人数分别为 q_{k1},q_{k2}。对于每个死亡时间 d_k，若没有同时死亡患者，则 q_{k1} 和 q_{k2} 中一个为 1，另一个为

0。注意到 $r_{k1}+r_{k2}=r_k, q_{k1}+q_{k2}=q_k$。在每个死亡时间点，我们作形如下表的 2×2 表格：

	Group 1	Group 2	Total
Died	q_{1k}	q_{2k}	q_k
Survived	$r_{1k}-q_{1k}$	$r_{2k}-q_{2k}$	r_k-q_k
Total	r_{1k}	r_{2k}	r_k

对数秩检验的核心思路如下：为了检验假设 $H_0: E(X)=\mu$，我们对随机变量 X 构造一个检验统计量：

$$W=\frac{X-\mu}{\mathrm{Var}(X)}$$

构造对数秩统计量我们则计算 $X=\sum_{k=1}^{K} q_{k1}$，其中 q_{k1} 通过查上面的表得到。若两组数据中没有显著差异，则下公式成立：

$$\mu_k=\frac{r_{k1}}{r_k}q_k$$

因此 $X=\sum_{k=1}^{K} q_{k1}$ 的期望 $\mu=\sum_{k=1}^{K}\frac{r_{k1}}{r_k}q_k$。进一步地 q_{k1} 的方差：

$$\mathrm{Var}(q_{k1})=\frac{q_k(r_{k1}/r_k)(1-r_{k1}/r_k)(r_k-q_k)}{r_k-1}$$

尽管 q_{ki} 可能相关，我们依然估计：

$$\mathrm{Var}(\sum_{k=1}^{K} q_{k1})\approx\sum_{k=1}^{K}\mathrm{Var}(q_{k1})=\sum_{k=1}^{K}\frac{q_k(r_{k1}/r_k)(1-r_{k1}/r_k)(r_k-q_k)}{r_k-1}$$

这一步我们得到了检验统计量中的 $\mathrm{Var}(X)$，因此最终：

$$W=\frac{\sum_{k=1}^{K}(q_{k1}-\mu_k)}{\sqrt{\sum_{k=1}^{K}\mathrm{Var}\left(q_{k1}\right)}}=\frac{\sum_{k=1}^{K}\left(q_{k1}-\frac{q_k}{r_k}r_{k1}\right)}{\sqrt{\sum_{k=1}^{K}\frac{q_k(r_{k1}/r_k)(1-r_{k1}/r_k)(r_k-q_k)}{r_k-1}}}.$$

当样本量较大时，对数秩检验得到的统计量 W 大致呈现标准正态分布，因此计算零假设的 p 值可以比较两组生存曲线之间的差异

11.5 回归模型与生存数据

考虑回归模型在生存数据中的应用。每对观测值 (Y,δ)，其中 $Y=\min(T,C)$，指示变量 δ 在 $T\le C$ 时为 1，否则为 0；特征向量 $X\in R^p$，表示有 p 个特征的向量，预测目标为真实生存时间 T

我们的目的是预测 T 而不是 Y，因此需要模仿之前的操作使用顺序结构

11.5.1 风险函数

风险函数 (hazard function)，也称为**及时风险率 (hazard rate)** 定义为：

$$h(t) = \lim_{\Delta t \to 0} \frac{\Pr(t < T \leq T + \Delta t | T > t)}{\Delta t}$$

它衡量了在时间 t 时，个体未发生事件的条件下，瞬间发生事件的速率。将生存数据建模为协变量函数的关键方法在于风险函数

由条件概率公式得：

$$\begin{aligned} h(t) &= \lim_{\Delta t \to 0} \frac{\Pr\left((t < T \leq t + \Delta t) \cap (T > t)\right)/\Delta t}{\Pr(T > t)} \\ &= \lim_{\Delta t \to 0} \frac{\Pr(t < T \leq t + \Delta t)/\Delta t}{\Pr(T > t)} \\ &= \frac{f(t)}{S(t)}, \end{aligned}$$

其中：

$$f(t) = \lim_{\Delta t \to 0} \frac{\Pr(t < T \leq t + \Delta t)}{\Delta t}$$

表示 T 的**概率密度函数 (probability density function)**，也即死亡时间 T 的瞬时变化率，第一步消去交集的过程是显然的

> 上式的三个等号实际上是表述 T 分布的等效方式

第 i 对观测值的似然函数为：

$$\begin{aligned} L_i &= \begin{cases} f(y_i) & \text{if the } i\text{th observation is not censored} \\ S(y_i) & \text{if the } i\text{th observation is censored} \end{cases} \\ &= f(y_i)^{\delta_i} S(y_i)^{1-\delta_i}. \end{aligned}$$

其中 y_i 表示某个时间，$f(y_i)$ 表示概率密度函数，$S(y_i)$ 表示生存函数，δ_i 是当前时间的删失指示变量，值为 1 时表示事件发生

这个公式的直觉是：若 $Y = y_i$ 时，第 i 个观测值未被删失，则似然函数应为事件刚好发生在 y_i 的概率，即密度函数 $f(y_i)$；如果第 i 个观测值被删失，则说明其至少存活到 y_i，则似然函数应为个体至少活到 y_i 的概率，即生存函数 $S(y_i)$

假设 n 对观测值独立，极大似然函数的形式则是：

$$L = \prod_{i=1}^{n} f(y_i)^{\delta_i} S(y_i)^{1-\delta_i} = \prod_{i=1}^{n} h(y_i)^{\delta_i} S(y_i),$$

得到似然函数后，我们就可以对原函数进行估计了。可采用两种形式：

1. 假设生存函数是形如 $f(t) = \lambda e^{-\lambda t}$ 的指数形式，或者来自 Γ 或者 Weibull 分布族
2. 用非参数化的估计，即类似于 Kaplan-Meier 估计器的阶梯形式

使用风险函数来估计生存函数比直接用概率密度函数好，这是因为前者反应了生存函数的变化率，更能展示协变量与生存函数的关系。我们可以假设一个指数型的风险函数：

$$h(t|x_i) = e^{\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}}$$

其中指数函数保证风险函数始终为正。上式的 $\beta_0, \beta_1, \ldots, \beta_p$ 可用极大似然法估计，然而指数函数形式的风险函数在每时每刻的值恒定，不符合风险逐渐增大的实际 (即生存函数的导数的导数为 0，实际应为正数)

11.5.2 比例风险

比例风险假设 (proportional hazards assumption) 指出：

$$h(t|x_i) = h_0(t) \exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)$$

其中 $h_0(t) \geq 0$ 是一个未知函数，也即**基础风险 (baseline hazard)**。基础风险衡量了当 $x_{i1} = \cdots = x_{ip} = 0$ 时的风险。“比例风险”的名字来源于 $h_0(t)$ 对每一类特征向量 $X_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ 的图像成比例 (平移且不交叉)。项 $\exp(\sum_{j=1}^{p} x_{ij}\beta_j)$ 被称为关于特征 $X_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ 的**相对风险 (relative risk)**

未指明的 $h_0(t)$ 实际上提供了任意满足数据的函数，这使得风险函数的形式相当灵活，这里关键假设是协变量 x_{ij} 是以指数形式影响风险率 $\exp(\sum_{j=1}^{p} x_{ij}\beta_j)$ 的

> 比例风险假设可能在实际上并不容易成立，因为它意味着不同组的风险比例恒定，生存曲线永不交叉。然而事实上，随着时间的变化，风险比可能变动，生存曲线可能交叉

Cox 比例风险模型 (Cox's proportional hazards model) 提供了一种在无法使用极大似然法估计不确定 $h_0(t)$ 条件下获得参数 $\beta = (\beta_1, \ldots, \beta_p)^T$ 的估计，我们同样使用时间顺序的方法来实现这种估计。首先，假设每个个体的死亡发生在各不相同的时间，且 $\delta_i = 1$，因此 y_i 是真正的死亡时间。则对于 y_i 时刻的第 i 对观测值其风险函数值为 $h(y_i|x_i) = h_0 \exp(\sum_{j=1}^{p} x_{ij}\beta_j)$，$y_i$ 时的总体风险 (处于风险集中，即为发生死亡的其他个体的风险) 为：

$$\sum_{i':y_{i'} \geq y_i} h_0(y_i) \exp\left(\sum_{j=1}^{p} x_{i'j}\beta_j\right)$$

则第 i 对观测值相对于总体的风险函数值为：

$$\frac{h_0(y_i) \exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}{\sum_{i':y_{i'} \geq y_i} h_0(y_i) \exp\left(\sum_{j=1}^{p} x_{i'j}\beta_j\right)} = \frac{\exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}{\sum_{i':y_{i'} \geq y_i} \exp\left(\sum_{j=1}^{p} x_{i'j}\beta_j\right)}$$

因为 Cox 模型是成比例的，第 i 个个体的失败概率通过上式给出 (表示在所有仍然存活的个体中，第 i 个成为下一个死亡者的概率)。在这里，基础风险函数 $h_0(y_i)$ 被抵消了

偏极大似然 (partial likelihood) 函数即是这些相对于所有个体发生风险的比例的乘积：

$$PL(\beta)=\prod_{i:\delta_i=1}\frac{\exp\left(\sum_{j=1}^{p}x_{ij}\beta_j\right)}{\sum_{i':y_{i'}\geq y_i}\exp\left(\sum_{j=1}^{p}x_{i'j}\beta_j\right)}$$

偏似然函数忽略了 $h_0(t)$ 的具体形式，但可以证明其提供了对 β 的良好估计。此外，一些和回归模型有关的其他信息，如零假设 $H_0:\beta_j=0$ 也可以被验证并显示

> 偏似然函数一般是对不好直接使用极大似然估计的模型使用的，为了最大化偏极大似然函数 (没有封闭解)，我们需要用到第四章中的迭代算法

当协变量只有一个，且取值为二元时，我们使用对数秩检验和直接用 Cox 模型检测 β 等价。一般来说，如果单纯比较两组生存时间的差异，使用对数秩检验更便捷；若需要量化影响，则使用 Cox 模型可以提供调整协变量的估计

一些关于 Cox 模型的细节如下：

- Cox 比例风险模型的指数项中没有 β_0，这是因为它放入了 $h_0(t)$ 中
- 假设死亡时间是唯一的，即无同时死亡；若有，则偏似然函数形式发生改变，计算更复杂
- 偏极大似然不是极大似然，是对极大似然的近似
- 通常，我们只关注 β 的估计，对 $h_0(t)$ 估计的数学过程超出本书范围

11.5.3 Brain Cancer 数据集研究实例

本节用比例风险模型拟合 `BrainCancr` 数据集。结果显示，男性比女性在任意时间发生癌症的风险高出 $e^{0.18}=1.2$ 倍，但是 p 值为 0.61，表明结果并不显著

医学中的一种指数 (Karnofsky 指数) 在模型中的系数 β_i 为 -0.05，$p=0.0027$，表明越高的指数显示越低的患病风险，结果显著

11.5.4 Publication 数据集研究实例

本节拟合 `Publication` 数据集，目的是研究论文发表时间与临床医学上的各种协变量的关系。首先用 Kaplan-Meier 生存曲线拟合了关于阴性实验结果和阳性实验结果的论文发表时间，并用对数秩检验检验其显著性，对数秩检验表明无显著差异

接着，考虑 Cox 风险比例模型，对比阴性实验结果和阳性实验结果论文发表时间的差异，此时检验认为差异显著。出现两种不同结果的原因是只有后者考虑了全部协变量影响 (调整其他协变量一致)

11.6 Cox 模型的收缩方法

受到损失函数+惩罚项的启发，我们可对 Cox 模型的偏极大似然函数使用收缩方法：

$$-\log\left(\prod_{i:\delta_i=1}\frac{\exp\left(\sum_{j=1}^{p}x_{ij}\beta_j\right)}{\sum_{i':y_{i'}\geq y_i}\exp\left(\sum_{j=1}^{p}x_{i'j}\beta_j\right)}\right)+\lambda P(\beta)$$

此处我们还需要考虑 $P(\beta) = \sum_{j=1}^{p} \beta_j^2$ 对损失函数的影响，当然也可改为 $P(\beta) = \sum_{j=1}^{p} |\beta_j|$ 惩罚项，二者分别对于岭回归和 lasso 回归

将 lasso 惩罚项应用到 Cox 模型对 `Publication` 数据集的拟合中。随着正则化强度的加强，模型在验证集中的偏似然误差 (partial likelihood deviance) 呈现 U 型曲线，其值是负对数部分似然的两倍。这显示了模型复杂度和误差之间的平衡关系

在测试集中，应用 Cox 模型存在两个问题：

1. 删失数据影响我们对其真实生存曲线的估计
2. Cox 模型不是用特征 X 来估计某个个体的生存时间 T，而是用 X 对它的生存曲线 $S(t|X)$ 进行建模，是时间 t 的函数

因此，为了评估模型，我们对每个测试观测值定义其风险评分：

$$\text{risk}_i = \text{budget}_i \cdot \hat{\beta}_{\text{budget}} + \text{impact}_i \cdot \hat{\beta}_{\text{impact}}$$

其中 $\hat{\beta}_{\text{budget}}$ 和 $\hat{\beta}_{\text{impact}}$ 是训练集中估计出来的两个特征参数，我们用这个两个值的特征组合来表示“风险”，例如高风险组中包含这对值最大的观测值。在 `publication` 中数据集中，观测结果被分为高中低三层，这三层之间的生存曲线存在明显区分，排序正确

11.7 其他主题

11.7.1 生存曲线的曲边形面积

类比二分类问题使用的 ROC 曲线，得到 AUC (曲线下面积) 来获得分类器效果比较的方法，我们试图分析生存曲线的曲线下面积的意义

考虑设计每对观测值的风险评分 $\hat{\eta}_i = \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_p x_{ip}$，我们可能会尝试判断风险评分和实际生存时间 t_i 都符合预测的观测对有多少来评价模型性能 (以风险评分较大，且实际生存时间较短为判断正例)。然而删失数据将会导致我们无法获得实际的生存时间 $t_{i'}$

因此，需要用**Harrell's 一致性系数 (Harrell's concordance index)**，也被称为**C 系数 (C-index)** 来计算满足 $\hat{\eta}_{i'} > \hat{\eta}_i$ 且 $\hat{y}_i > y_{i'}$ 的比例：

$$C = \frac{\sum_{i,i':y_i>y_{i'}} I(\hat{\eta}_{i'} > \hat{\eta}_i)\delta_{i'}}{\sum_{i,i':y_i>y_{i'}} \delta_{i'}}$$

其中指示变量 $I(\hat{\eta}_{i'} > \hat{\eta}_i)$ 在 $\hat{\eta}_{i'} > \hat{\eta}_i$ 满足时为 1。这个指数的分子表示所有满足实际生存时间较长的个体中，模型也能正确排序的观测对数量；分母表示所有满足实际生存时间较长的个体中，实际没有被删失的观测对数量

> 当 C 越接近 1 时，表明模型效果越好；若 $C = 0.5$，则模型等同于随机猜测

11.7.2 时间刻度的选择

时间刻度的选择至关重要。例如，考察协变量对患者治疗效果的影响，我们可以选择两种时间零作为开始时间：

1. 以患者的出生日期作为时间零点。此时 $h_0(t)$ 隐含量患者的年龄，即年龄被时间刻度吸收，无需额外调整年龄
2. 以治疗日期作为时间零点。此时年龄是独立于时间变量的协变量，需要额外加入模型

不同的选择取决于研究背景，包括我们是否要单独研究年龄的影响

11.7.3 随时间变化的协变量

比例风险模型的强大之处在于它能处理随时间变化的协变量。例如，测量血压时，我们不在将血压值视为静态特征 x_i，而是视为一个随时间变化的动态特征 $x_i(t)$

因为偏似然本身就是随着时间序列构造的，因此它处理随时间变化的协变量更加直接。我们只需要将计算 $PL(\beta)$ 式子中的 $x_{ij}, x_{i'j}$ 改成 $x_{ij}(y_i), x_{i'j}(y_i)$。一个实例是比较心脏移植患者能否获得更长的生存期，若使用固定的协变量表示移植状态，则会忽略患者必须活得足够长才能接受心脏移植的事实。因此，需要用随时间变化的协变量来建模移植状态，即患者在 t 前接受移植，则 $x_i(t) = 1$，否则 $x_i(t) = 0$

11.7.4 比例风险假设的检验

Cox 模型依赖于比例风险假设。在定性特征情况下，我们可以绘制每个级别的对数风险函数，若假设成立，则这些函数应该只差一个常数；对于定量的特征，我们可以用特征分层方法来实现验证

> Cox 模型对违反此假设的数据依然具有相当的稳健性

11.7.5 生存树

决策树的思想可以用于生存数据分析，同理，也可以用随机森林的方式来优化

11.8 实验：生存分析

本次实验导入的基础包：

```
from matplotlib import pyplot as plt
import numpy as np
import pandas as pd
from ISLP.models import ModelSpec as MS
from ISLP import load_data
```

新增包：

```
from lifelines import \
     (KaplanMeierFitter,
      CoxPHFitter)
from lifelines.statistics import \
     (logrank_test,
```

```
      multivariate_logrank_test)
from ISLP.survival import sim_time
```

11.8.1 BrainCancer数据集

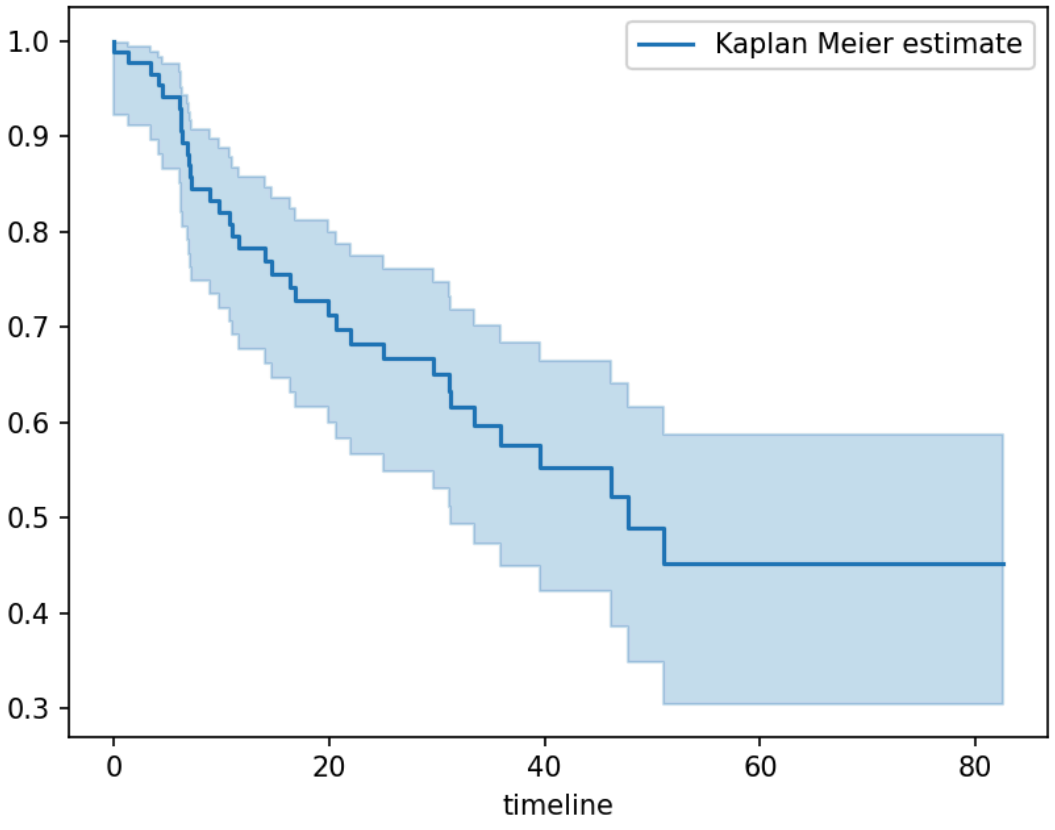
首先加载数据。注意我们需要知道 `status` 列的数字1和0哪个表示删失，一般来说0表示删失：

```
#导入数据
BrainCancer = load_data('BrainCancer')
print(BrainCancer.columns)
print(BrainCancer['sex'].value_counts()) #对某列进行分类计数
print(BrainCancer['diagnosis'].value_counts())
```

我们来查看数据的生存曲线：

```
#生成Kaplan-Meier生存曲线
fig, ax = plt.subplots()
km = KaplanMeierFitter()
km_brain = km.fit(BrainCancer['time'], BrainCancer['status']) #指定时间列和状态列
km_brain.plot(label='Kaplan Meier estimate', ax=ax)
```

> 注意，在默认下，生成的是90%置信区间；可通过 `alpha` 参数设置为1减去置信度来更改

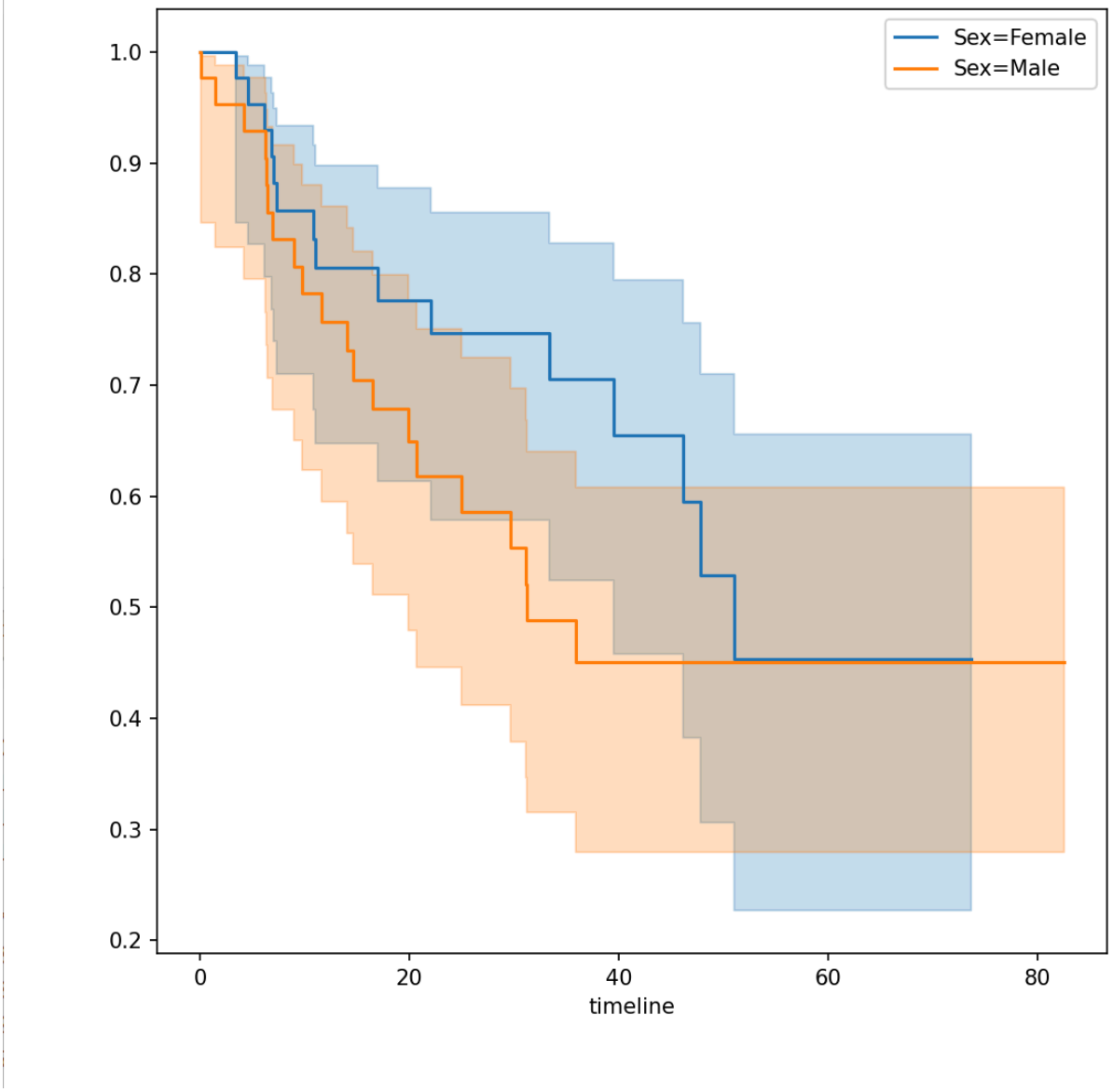


> 注意`KaplanMeierFitter()`估计器需要指定数据集的时间列和状态列

`.groupby()`方法类似于`SQL`语句中的`GROUP BY`，可以对数据框中的某列进行分组(传入列表可以实现多级分组)。利用这个特性，我们可以绘制不同性别的生存曲线：

```
#用对数秩检验检验男性和女性生存曲线差异
fig, ax = plt.subplots(figsize=(8, 8))
by_sex = {}
for sex, df in BrainCancer.groupby('sex'):
    by_sex[sex] = df
    km_sex = km.fit(df['time'], df['status'])
    km_sex.plot(label='Sex=%s' % sex, ax=ax)
print(  #对数秩检验
    logrank_test(by_sex['Male']['time'],
                 by_sex['Female']['time'],
                 by_sex['Male']['status'],
                 by_sex['Female']['status'])
)
```

上面的代码也显示了对数秩检验的结果，这需要用到 `logrank_test()` 函数



尝试使用 `CoxPHFitter()` 估计器拟合Cox模型，这里只使用 `sex` 作为唯一的协变量：

```
#使用Cox模型
coxph = CoxPHFitter #这里是缩写
sex_df = BrainCancer[['time', 'status', 'sex']] #构造要用的数据
model_df = MS(['time', 'status', 'sex'],
              intercept=False).fit_transform(sex_df) #Cox模型不应该携带截距项
cox_fit = coxph().fit(model_df,
                      'time', #生存时间列的名称
                      'status') #生存状态列的名称，用于删失变量标注
print(cox_fit.summary[['coef', 'se(coef)', 'p']]) #输出总结表中的三列
```

需要注意，传入 `CoxPHFitter()` 估计器的 `model_df` 不包括截距项

使用 `log_likelihood_ratio_test()` 将带有协变量的Cox模型与无协变量的生存函数进行比较：

```
#将有协变量sex与无协变量的风险比例模型的似然比进行比较
print(cox_fit.log_likelihood_ratio_test())
```

结果中的p值不支持性别对生存曲线影响的差异

> 分数测试与对数秩检验在统计上等价

现在我们尝试拟合所有的协变量，在此之间除去空值：

```
#重新拟合所有协变量模型
cleaned = BrainCancer.dropna()
all_MS = MS(cleaned.columns, intercept=False)
all_df = all_MS.fit_transform(cleaned)
fit_all = coxph().fit(all_df,
                      'time',
                      'status')
print(fit_all.summary[['coef', 'se(coef)', 'p']])
```

现在，我们希望绘制不同诊断分类下的生存曲线图。我们先定义了一个`representative()`函数，对特征进行处理。对于分类特征，我们使用其众数；对于定量特征，我们使用其均值，这样我们可以固定非诊断分类的其他特征的值：

```
#绘制不同诊断特征的生存曲线图
levels = cleaned['diagnosis'].unique() #提取诊断特征唯一值
def representative(series):
    if hasattr(series.dtype, 'categories'): #检测到分类变量
        return pd.Series.mode(series) #返回众数
    else:
        return series.mean() #返回均值
modal_data = cleaned.apply(representative, axis=0) #对每一列应用函数，计算代表值；生成一行数据
modal_df = pd.DataFrame(
    [modal_data.iloc[0] for _ in range(len(levels))]) #生成一个有levels数量行的dataframe
modal_df['diagnosis'] = levels #添加一列levels
print(modal_df)
```

在规定非诊断分类后，我们在新数据框中加入了`levels`作为分类

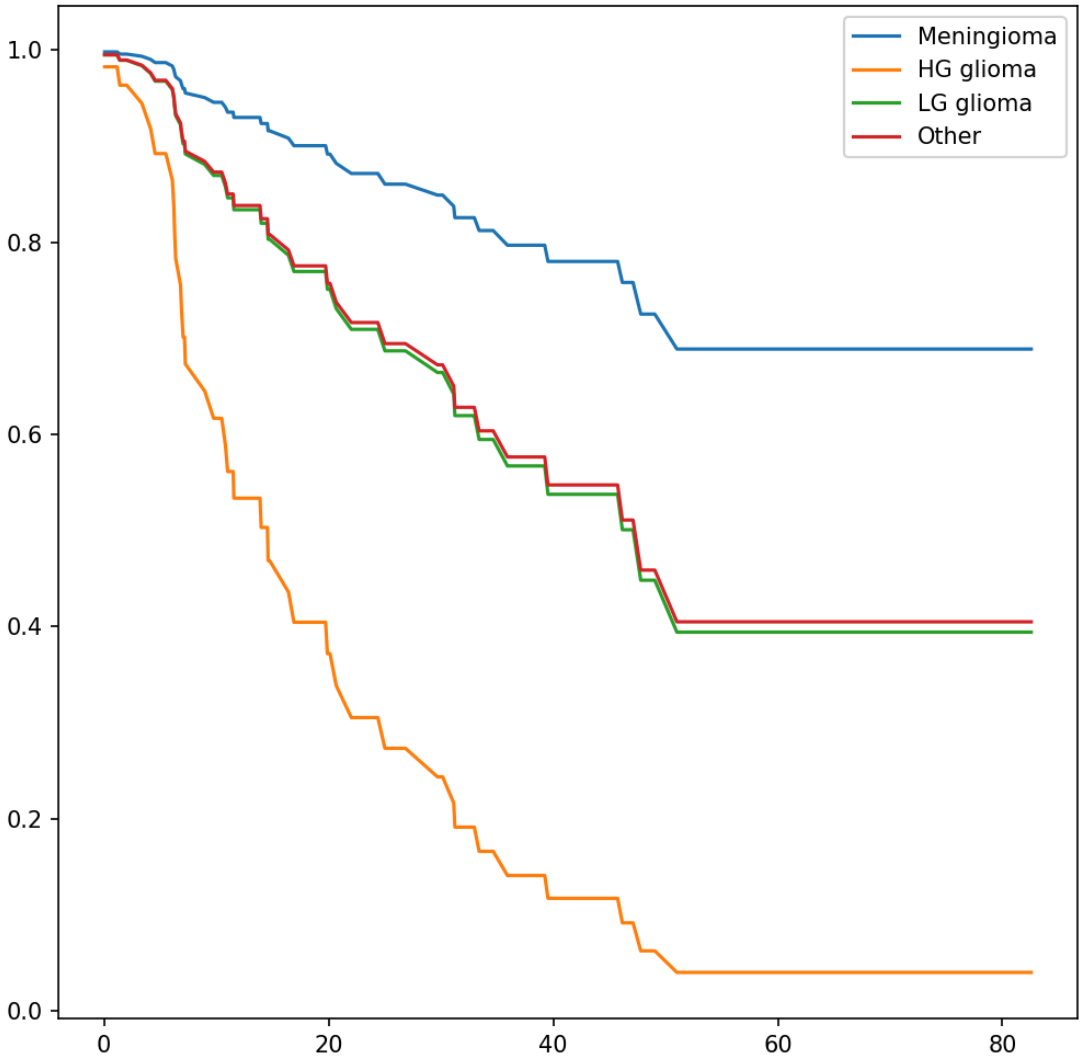
现在，我们用Cox模型来预测这四类随着时间变化的生存曲线，并绘制成图像：

```
#生成构造矩阵
modal_X = all_MS.transform(modal_df) #获取modal Dataframe的构造矩阵
modal_X.index = levels
print(modal_X)

#获取生存曲线的系数估计
```

```
predicted_survival = fit_all.predict_survival_function(modal_X)
print(predicted_survival)

#显示图像
fig, ax = plt.subplots(figsize=(8, 8))
predicted_survival.plot(ax=ax) #为了图像的清晰，不显示置信区间
plt.show()
```

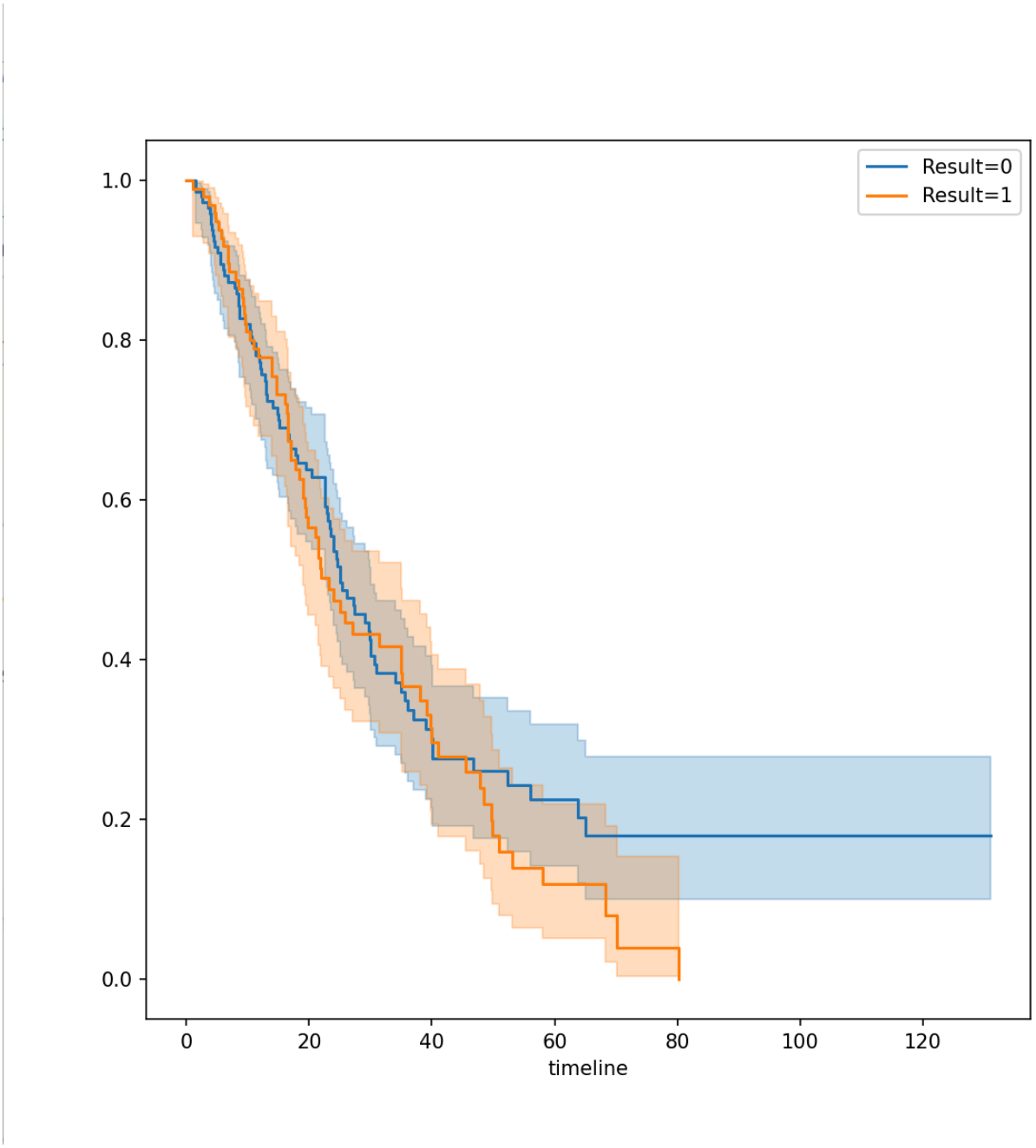


11.8.2 Publication数据集

`Publication`数据集显示了医学论文出版时间与协变量的关系。生成一个Kaplan-Meier曲线，曲线以`posrse`为协变量：

```
#生成关于posres协变量的生存曲线
fig, ax = plt.subplots(figsize=(8,8))
Publication = load_data('Publication')
```

```
by_result = {}
km = KaplanMeierFitter()
for result, df in Publication.groupby('posres'):
    by_result[result] = df
    km_result = km.fit(df['time'], df['status'])
    km_result.plot(label='Result=%d' % result, ax=ax)
plt.show()
```



p值和图像都暗示阳性结果与研究发表时间不存在差异

然而，当我们考虑所有变量时，p值表明它们之间存在差异：

```
#使用Cox模型
coxph = CoxPHFitter
posres_df = MS(['posres',
                'time',
                'status'],
```

```
                  intercept=False).fit_transform(Publication)
posres_fit = coxph().fit(posres_df,
                         'time',
                         'status')
print(posres_fit.summary[['coef', 'se(coef)', 'p']])
```

删除`mech`变量后的模型：

```
#删除mech后的模型
model = MS(Publication.columns.drop('mech'),
           intercept=False)
print(coxph().fit(model.fit_transform(Publication),
            'time',
            'status').summary[['coef', 'se(coef)', 'p']])
```

p值指出许多因素都会造成发表时间的差异

11.8.3 Call Center数据

本节采用练习8探讨的关于生存曲线与风险函数之间的关系，生成了研究用户拨打中心电话响应时间的模拟数据。若用户在未接听的情况下提前挂断电话，则认为数据发生删失。这个数据的协变量包括接线员数量，选择的话务中心(A, B, C)以及每天的时间(早晨，中午和晚上)，我们生成的数据关于时间是平均分布的：

```
#生成数据
rng = np.random.default_rng(10)
N = 2000
Operators = rng.choice(np.arange(5, 16),
                       N,
                       replace=True)
Center = rng.choice(['A', 'B', 'C'],
                    N,
                    replace=True)
Time = rng.choice(['Morn', 'After', 'Even'],
                  N,
                  replace=True)
D = pd.DataFrame({'Operators': Operators,
                  'Center': pd.Categorical(Center),
                  'Time': pd.Categorical(Time)})
```

这里用到的`pd.Categorical`可将列表数据转换为分类数据类型，便于后续数据分析

我们可以构造模型的设计矩阵了：

```
#获取设计矩阵
model = MS(['Operators',
            'Center',
```

```
             'Time'],
            intercept=False)
X = model.fit_transform(D)
print(X[:5]) #显示前五行
```

注意到每一个分类变量的第一列在`X`中实际上是被删除的(A, B, C中没有A)，这是独热编码的一个特性，即某一类被视为基类

现在我们指定系数和风险函数：

```
#系数和风险函数
true_beta = np.array([0.04, -0.3, 0, 0.2, -0.2])
true_linpred = X.dot(true_beta)
hazard = lambda t: 1e-5 * t
```

这里的真实系数可以这样理解：

- 每增加一个操作员，等待时间下降$e^{0.04} = 1.041$，即人数越多，等待时间越小
- 操作中心B的等待时间比操作中心A的等待时间长$e^{-0.3} = 0.74$

`sim_time()`是来自`ISLP.survival`中的一个函数，能够帮助我们依据Cox模型中的线性预测器和累积风险来模拟数据。具体来说，生存函数$S(t)$与累计风险$H(t)$之间的关系满足$S(t) = \exp\left(-H(t)\right)$，这意味着生存函数是累计风险的指数函数的负值。Cox模型中的累计风险函数需要具体形式，这里我们提供了一个：

```
cum_hazard = lambda t: 1e-5 * t**2 / 2
```

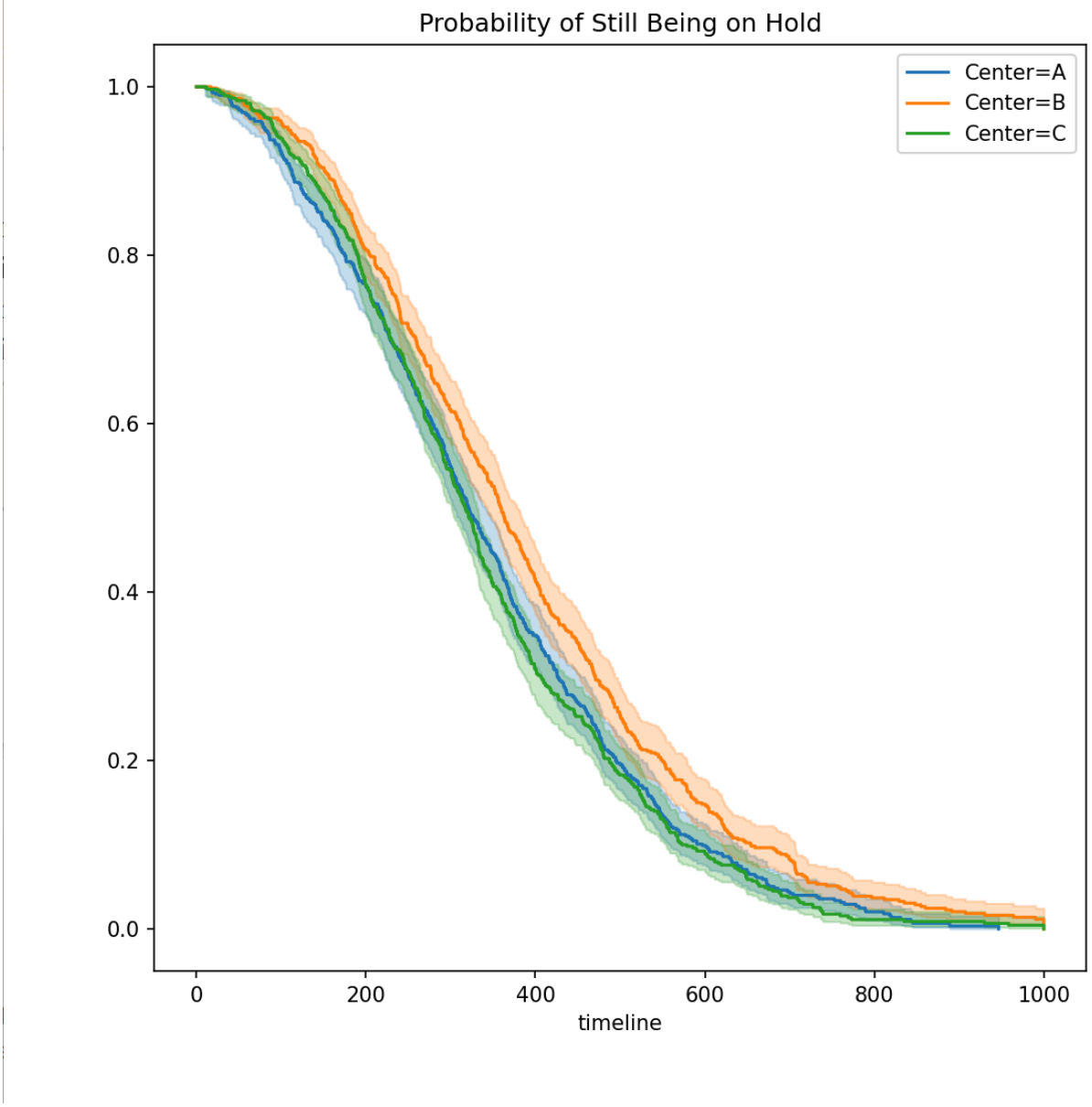
函数`sim_time()`采用线性预测器、累计风险函数和随机数生成器。此处最长等待时间设置为1ks，90%变量未删失：

```
#设置时间
W = np.array([sim_time(l, cum_hazard, rng)
              for l in true_linpred])
D['Wait time'] = np.clip(W, 0, 1000)

#模拟删失变量
D['Failed'] = rng.choice([1, 0],
                         N,
                         p=[0.9, 0.1])
print(D[:5])
print("Mean: ", D['Failed'].mean())
```

现在，我们绘制Kaplan-Meier生存曲线：

```
#绘制生存曲线
fig, ax = plt.subplots(figsize=(8, 8))
by_center = {}
for center, df in D.groupby('Center'):
    by_center[center] = df
    km_center = km.fit(df['Wait time'], df['Failed'])
    km_center.plot(label='Center=%s' % center, ax=ax)
ax.set_title('Probability of Still Being on Hold')
plt.show()
```



使用 `multivariate_logrank_test()` 检验曲线之间的差异是否具有统计学意义：

```
print(multivariate_logrank_test(D['Wait time'],
                                D['Center'],
                                D['Failed'])) #执行多变量对数秩检验
```

多变量对数秩检验多变量情况下生存曲线的差异情况，即单变量的扩展。结果显示差异显著

我们现在限定单变量来检测差异情况：

```
#查看Center之间的差异
X = MS(['Wait time', 'Failed', 'Center'],
       intercept=False).fit_transform(D)
F = coxph().fit(X, 'Wait time', 'Failed')
print(F.log_likelihood_ratio_test())

#查看Time之间的差异
X = MS(['Waite time', 'Failed', 'Center'],
       intercept=False).fit_transform(D)
F = coxph().fit(X, 'Wait time', 'Failed')
print(F.log_likelihood_ratio_test())
```

最后，我们用Cox模型来拟合数据集：

```
#拟合Cox模型
X = MS(D.columns,
       intercept=False).fit_transform(D)
fit_queuing = coxph().fit(
                  X,
                 'Wait time',
                 'Failed')
print(fit_queuing.summary[['coef', 'se(coef)', 'p']])
```