PSTAT175 - Final Project - Part 1 - Recurrence

Zifeng(Robin) Zhan, Calder Glass, Kotaro Ito

2025-04-29

We used a significance level of $\alpha = 0.05$ as the critical value.

```
library(survival)
```

```
## Warning: package 'survival' was built under R version 4.4.3
```

```
library(survminer)
```

```
## Warning: package 'survminer' was built under R version 4.4.3

## Loading required package: ggplot2

## Loading required package: ggpubr

## Warning: package 'ggpubr' was built under R version 4.4.3

##
## Attaching package: 'survminer'

## The following object is masked from 'package:survival':
##
##     myeloma
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
## Warning: package 'purrr' was built under R version 4.4.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.4     v tibble    3.2.1
## v purrr     1.0.4     v tidyr     1.3.1
## v readr     2.1.5
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
data("colon")
```

```
## Warning in data("colon"): data set 'colon' not found
```

```r
View(colon)

sum(is.na(colon))
```

```
## [1] 82
```

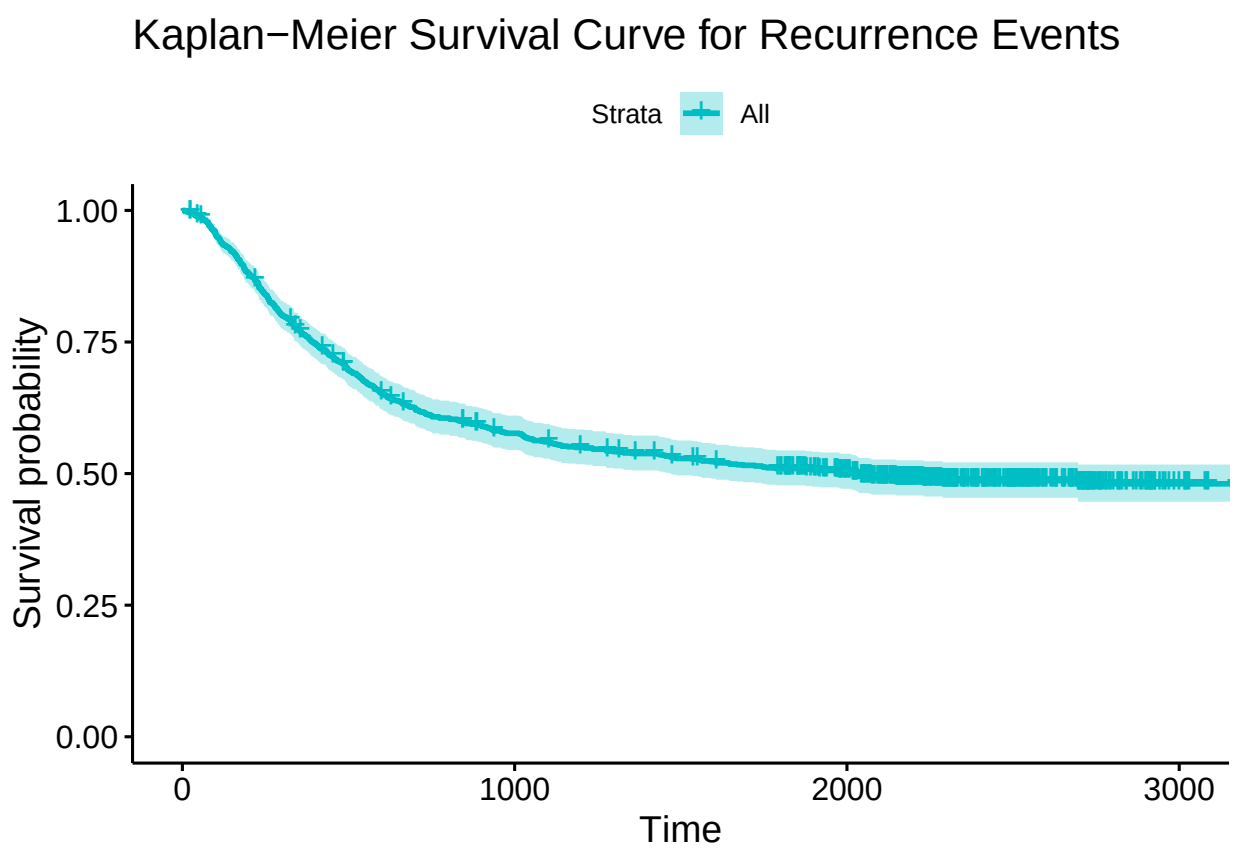
```r
# There are 82 missing observations - accounts for 4.4% of the data, safe enough to remove those observ

colon = tibble(colon)

colon <- colon %>%
  drop_na()
```

Two additional covariates will be introduced, start and stop, in order to build a counting process model later on.

```r
# Factor some of the covariates first in order to do the cox ph tests:

colon <- colon %>%
  mutate_at(c("sex", "obstruct", "perfor", "adhere", "differ", "etype", "node4", "surg", "extent"), fac

# Preparing the split
colon$start = 0
colon$stop = colon$time
```

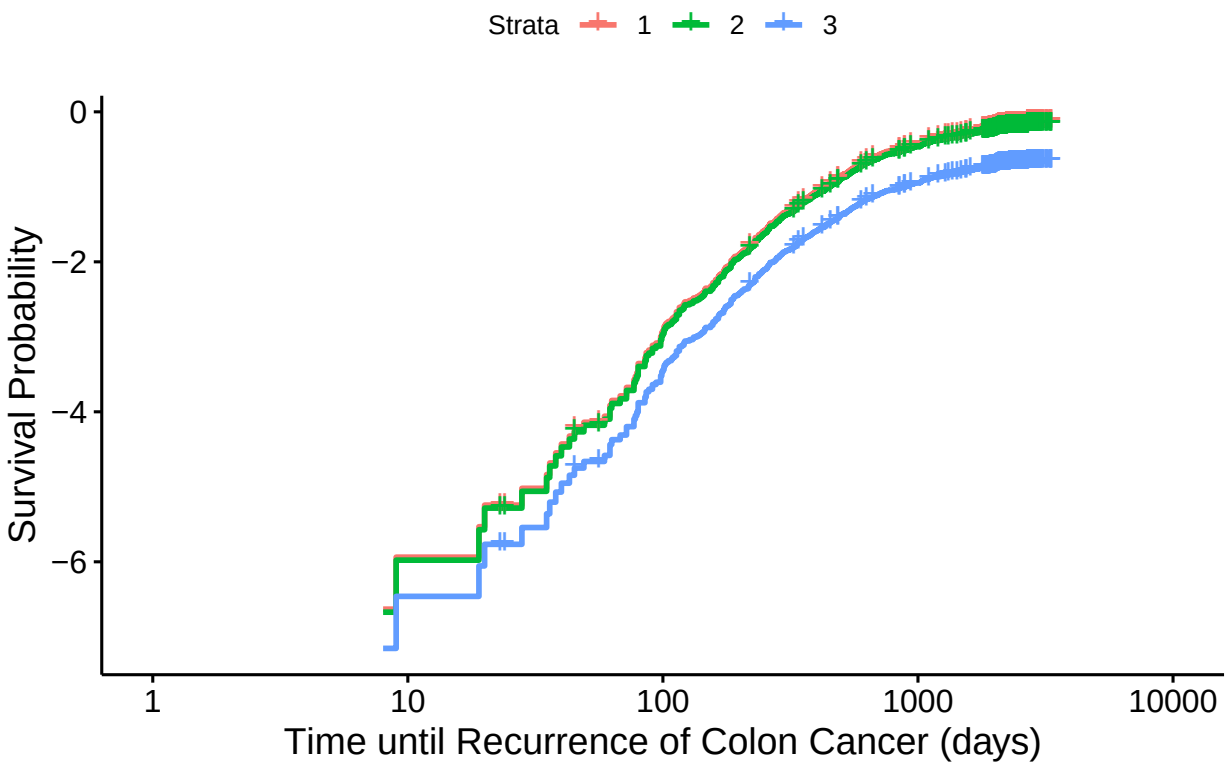
```r
# data for recurrence event
recurrence_data <- colon[colon$etype == 1, ]

# survival object
surv_object_recurrence <- Surv(time = recurrence_data$time,
                               event = recurrence_data$status)
```

```r
# Fit the Kaplan-Meier model for recurrence events
km_fit_recurrence <- survfit(surv_object_recurrence ~ 1)

# Plot the Kaplan-Meier survival curve for recurrence events with a title
ggsurvplot(km_fit_recurrence,
           data = recurrence_data,
           palette = "#00BFC4", # Customize the color
           title = "Kaplan-Meier Survival Curve for Recurrence Events")
```



Then we checked if the Cox Proportional Hazards assumption was violated or not in terms of the treatment covariate.

```r
recurrence_obj = coxph(Surv(time, status) ~ rx, data = recurrence_data)
recurrence_fit = survfit(recurrence_obj, newdata = data.frame(rx= c("Obs", "Lev", "Lev+5FU")))
ggsurvplot(recurrence_fit, data = recurrence_data, fun = "cloglog", conf.int = FALSE, title = "Comparis
```



The survival curves are fairly parallel, so it's reasonable to assume that the Cox Proportional Hazards Assumption for the recurrence portion of the data is not violated by the treatment covariate.

Covariates that are either binaries or have very few levels, 3 to 4 maximum, are factored so that the effect of each level of the covariates can be summarized.

```
# Factor some of the covariates first in order to do the cox ph tests:

colon <- colon %>%
  mutate_at(c("sex", "obstruct", "perfor", "adhere", "differ", "etype", "node4", "surg", "extent"), fac
```

AIC:

Now that the treatment covariate has been confirmed to not violate the Cox Proportional Hazards Assumption and the recurrence subset of colon cancer data has been cleaned, the next step is to find the model with the most significant covariates under the AIC criterion.

For the AIC tests, the covariates *study* and *id* are not included. The *id* covariate is the same as the observation number, it doesn't have contextual significance to the event of relapse or death from colon cancer. The *study* covariate is not included as all of the subjects are from the same study.

```
# Level 1:
# Construct a list of covariates to put into the models:
recurrence_covariates = c("obstruct", "adhere", "nodes", "node4", "differ", "extent", "surg", "perfor",

# building a model per covariate by pasting the given covariate into the formula

# the set_names function helps to clear up which AIC value corresponds to which model when performing t
```

```
recurrence_models = map(recurrence_covariates, \(v) coxph(as.formula(paste("Surv(time, status) ~ rx + "

aic_lvl1 = map_dbl(recurrence_models, AIC) |>
  sort()

aic_lvl1
```

```
##    node4    nodes   extent   differ   adhere     surg obstruct   perfor
## 5666.763 5677.508 5713.851 5728.289 5732.669 5733.290 5733.868 5735.037
##      sex      age
## 5735.202 5735.275
```

The model with the *node4* covariate, the binary variable for whether the patient had more than 4 positive lymph nodes, had the lowest AIC.

Since the *nodes* covariate and *node4* covariate are closely related, the *nodes* covariate will be skipped.

Therefore, forward selection proceeds with the above covariate.

```
# Level 2:
# Construct a list of covariates to put into the models:
recurrence_covariates2 = c("obstruct", "adhere", "differ", "extent", "surg", "perfor", "sex", "age")

# Build a model per covariate by pasting the given covariate into the formula.

# The set_names function helps to clear up which AIC value corresponds to which model when performing t

recurrence_models2 = map(recurrence_covariates2, \(v) coxph(as.formula(paste("Surv(time, status) ~ rx +

aic_lvl2 = map_dbl(recurrence_models2, AIC) |>
  sort()

aic_lvl2
```

```
##   extent obstruct   differ     surg   adhere   perfor      sex      age
## 5650.922 5663.826 5663.897 5664.489 5664.654 5666.367 5667.227 5668.279
```

The model with the *extent* covariate, the description of the local spread of the tumor, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 3:
# Construct a list of covariates to put into the models:
recurrence_covariates3 = c("obstruct", "adhere", "differ", "surg", "perfor", "sex", "age")

# building a model per covariate by pasting the given covariate into the formula

# the set_names function helps to clear up which AIC value corresponds to which model when performing t

recurrence_models3 = map(recurrence_covariates3, \(v) coxph(as.formula(paste("Surv(time, status) ~ rx +

aic_lvl3 = map_dbl(recurrence_models3, AIC) |>
  sort()

aic_lvl3
```

```
##     surg   differ obstruct   adhere   perfor      sex      age
## 5647.952 5649.310 5649.831 5650.401 5651.370 5651.535 5652.557
```

The model with the *surg* covariate, the time from initial surgery to registration in the study, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 4:
# list of covariates to put into the models
recurrence_covariates4 = c("obstruct", "adhere", "differ", "perfor", "sex", "age")

# building a model per covariate by pasting the given covariate into the formula

# the set_names function helps to clear up which AIC value corresponds to which model when performing t

recurrence_models4 = map(recurrence_covariates4, \(v) coxph(as.formula(paste("Surv(time, status) ~ rx +

aic_lvl4 = map_dbl(recurrence_models4, AIC) |>
  sort()

aic_lvl4
```

```
##   differ obstruct   adhere   perfor      sex      age
## 5646.470 5647.117 5647.543 5648.347 5648.412 5649.503
```

The model with the *differ* covariate, the description of the removed cancer cells from the colon, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 5:
# list of covariates to put into the models
recurrence_covariates5 = c("adhere", "obstruct", "perfor", "sex", "age")

# building a model per covariate by pasting the given covariate into the formula

# the set_names function helps to clear up which AIC value corresponds to which model when performing t

recurrence_models5 = map(recurrence_covariates5, \(v) coxph(as.formula(paste("Surv(time, status) ~ rx +

aic_lvl5 = map_dbl(recurrence_models5, AIC) |>
  sort()

aic_lvl5
```

```
## obstruct   adhere      sex   perfor      age
## 5645.417 5646.611 5646.921 5647.033 5647.914
```

The model with the *obstruct* covariate, the binary variable for whether the cancer had adhered to other organs, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```r
# Level 6:
# list of covariates to put into the models
recurrence_covariates6 = c("adhere", "perfor", "sex", "age")

# building a model per covariate by pasting the given covariate into the formula

# the set_names function helps to clear up which AIC value corresponds to which model when performing t

recurrence_models6 = map(recurrence_covariates6, \(v) coxph(as.formula(paste("Surv(time, status) ~ rx +

aic_lvl6 = map_dbl(recurrence_models6, AIC) |>
  sort()

aic_lvl6
```

```
##   adhere      sex   perfor      age
## 5645.561 5646.087 5646.407 5647.091
```

None of the AICs shown above are less than the previous model, so the chosen model has the following covariates: *obstruct*, *surg*, *extent*, *node4*, and *differ*.

Next, the model was summarized in order to conclude relationships between the different levels of covariates, such as the treatment covariate and the differentiation covariate.

```r
summary(coxph(Surv(time, status) ~ rx + node4 + extent + surg + differ + obstruct, data = recurrence_da
```

```
## Call:
## coxph(formula = Surv(time, status) ~ rx + node4 + extent + surg +
##     differ + obstruct, data = recurrence_data)
##
##   n= 888, number of events= 446
##
##               coef exp(coef) se(coef)      z Pr(>|z|)
## rxLev     -0.01639   0.98374  0.11097 -0.148   0.8826
## rxLev+5FU -0.50013   0.60645  0.12187 -4.104 4.06e-05 ***
## node41     0.83424   2.30307  0.09982  8.358  < 2e-16 ***
## extent2    0.24399   1.27634  0.53027  0.460   0.6454
## extent3    0.79137   2.20642  0.50568  1.565   0.1176
## extent4    1.29347   3.64541  0.54350  2.380   0.0173 *
## surg1      0.22603   1.25362  0.10408  2.172   0.0299 *
## differ2   -0.03295   0.96759  0.16406 -0.201   0.8408
## differ3    0.27195   1.31252  0.19103  1.424   0.1546
## obstruct1  0.21014   1.23385  0.11790  1.782   0.0747 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##           exp(coef) exp(-coef) lower .95 upper .95
## rxLev        0.9837     1.0165    0.7914    1.2228
## rxLev+5FU    0.6065     1.6489    0.4776    0.7701
## node41       2.3031     0.4342    1.8938    2.8007
## extent2      1.2763     0.7835    0.4514    3.6086
## extent3      2.2064     0.4532    0.8189    5.9446
## extent4      3.6454     0.2743    1.2564   10.5772
```

```
## surg1        1.2536     0.7977    1.0223    1.5373
## differ2      0.9676     1.0335    0.7015    1.3345
## differ3      1.3125     0.7619    0.9026    1.9086
## obstruct1    1.2339     0.8105    0.9793    1.5546
##
## Concordance= 0.662  (se = 0.013 )
## Likelihood ratio test= 128.8  on 10 df,   p=<2e-16
## Wald test            = 130.4  on 10 df,   p=<2e-16
## Score (logrank) test = 138.3  on 10 df,   p=<2e-16
```

From the likelihood ratio test, the p-value is less than $2e-16$, which is much less than the critical value/significance level of 0.05.

The hazard rate for patients who took the treatment with just Levamisole is 1.163% less hazardous than taking no treatment at all. Those who took Fluoracil in addition to Levamisole benefited with a hazard ratio of 0.6065, 39.35% less hazardous than no treatment at all.

Patients who had more than 4 positive lymph nodes had over double the hazard rate of those who didn't.

As the spread of the tumor developed from muscles to continguous structures, the hazard ratio to those who only had submucosa development increased to as high as 3.64 times as likely to suffer a recurrence of colon cancer.

Patients with a long time from their initial surgery to registration in the study had a 25% greater hazard rate than those with a shorter time interval.

Patients whose removed cancer cells were "moderately differentiated" had a 3.24% lower hazard rate than patients whose cancer cells were "well differentiated", while those with "poorly differentiated" cells had 31.25% higher hazard rate compared to same base group.

Patients whose colons were obstructed by a tumor had a 23.39% higher hazard rate compared to those who were obstruction-free.

With the following exceptions of the treatment level that included Fluoracil and Levamisole, the node level of patients who had more than 4 positive lymph nodes, and the long time interval level between initial surgery to registering for the study, all of the other covariates' levels had 95% confidence intervals which contained the baseline 1. This suggests that the most significant levels of covariates in their effect on the hazard rate of the recurrence of colon cancer are Levamisole + Fluoracil as a treatment, over 4 positive lymph nodes, spread of cancer to the contiguous structures, and a long time between initial surgery to registration for the study is the best fit for the recurrence data.

Counting Process Model:

The data is looking at the time between recurrences of colon cancer and death. As a result, it can't be treated as independent intervals like the gap model because there is a relationship between the recurrence of colon cancer and death from colon cancer. Thus, the counting process model is the best model to use.

Earlier in the report, the colon covariates *start* and *stop* were created. They are needed in order to separate the recurrence of colon cancer and death from colon cancer into new rows.

```
# Builds the start and stop covariates for the subset of the colon cancer data where only the recurrenc
recurrence_data = recurrence_data %>%
  mutate(start = 0,
         stop = time)
# Established the subset of the colon cancer data where only the deaths from colon cancer are noted
```

```r
death_data = colon[colon$etype == 2,]

# Modifying the death subset to join the recurrence subset by id. The mutation is so the end time for t

death_data = death_data %>%
  left_join(recurrence_data %>% select(id, recurrence_time = time), by = "id") %>%
  mutate(start = recurrence_time,
         stop = time)

# The two subsets are then combined by the id covariate from earlier. They are then arranged in order o

colon_counting = bind_rows(recurrence_data, death_data) %>%
  arrange(id) %>%
  select(-recurrence_time) %>%
  mutate(start = if_else(start == stop, 0, start))
```

Now the colon dataset counts the time between the beginning to the first recurrence, the time between the recurrence to the next recurrence or death.

Next, a cox proportional hazards model of this dataset will be constructed with the new time variables in the Survival object. Then, AIC + forward selection will find the most significant covariates for this model.

Similar to the AIC process for the marginal model for the recurrence data, the study covariate will not be included. However, the *id* covariate will be included, but as a cluster in order to compress multiple *id* rows that have the same *id* into one subject/*id*.

```r
# Level 1:
# list of covariates to put into the models
colon_covariates = c("obstruct", "adhere", "nodes", "node4", "differ", "extent", "surg", "perfor", "sex

# building a model per covariate by pasting the given covariate into the formula

# the set_names function helps to clear up which AIC value corresponds to which model when performing t

colon_models = map(colon_covariates, \(v) coxph(as.formula(paste("Surv(start, stop, status) ~ rx + clus

colon_aic_lvl1 = map_dbl(colon_models, AIC) |>
  sort()

colon_aic_lvl1
```

```
##    node4    nodes   extent   differ   adhere     surg obstruct   perfor
## 12078.67 12096.23 12184.40 12223.57 12229.57 12231.17 12234.77 12239.16
##      sex      age
## 12240.89 12241.37
```

The model with the *node4* covariate, the binary variable for whether the patient had more than 4 positive lymph nodes, had the lowest AIC.

Since the *nodes* covariate and *node4* covariate are closely related, the *nodes* covariate will be skipped.

Therefore, forward selection proceeds with the above covariate.

```
# Level 2:
# list of covariates to put into the models
colon_covariates2 = c("obstruct", "adhere", "differ", "extent", "surg", "perfor", "sex", "age")

# building a model per covariate by pasting the given covariate into the formula

# the set_names function helps to clear up which AIC value corresponds to which model when performing t

colon_models2 = map(colon_covariates2, \(v) coxph(as.formula(paste("Surv(start, stop, status) ~ rx + clu

colon_aic_lvl2 = map_dbl(colon_models2, AIC) |>
  sort()

colon_aic_lvl2
```

```
##   extent     surg   adhere obstruct   differ   perfor      age      sex
## 12038.26 12068.37 12069.54 12071.37 12071.49 12077.78 12080.24 12080.31
```

The model with the *extent* covariate, the description of the local spread of the tumor, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 3:
# Construct a list of covariates to put into the models:
colon_covariates3 = c("obstruct", "adhere", "differ", "surg", "perfor", "sex", "age")

# Build a model per covariate by pasting the given covariate into the formula.

# The set_names function helps to clarify up which AIC value corresponds to which model when performing

colon_models3 = map(colon_covariates3, \(v) coxph(as.formula(paste("Surv(start, stop, status) ~ rx + clu

colon_aic_lvl3 = map_dbl(colon_models3, AIC) |>
  sort()

colon_aic_lvl3
```

```
##     surg   adhere   differ obstruct   perfor      age      sex
## 12026.71 12032.99 12033.07 12034.78 12038.63 12039.68 12040.07
```

The model with the *surg* covariate, the time from initial surgery to registration in the study, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 4:
# Construct a list of covariates to put into the models:
colon_covariates4 = c("obstruct", "adhere", "differ",  "perfor", "sex", "age")

# Build a model per covariate by pasting the given covariate into the formula.

# The set_names function helps to clarify up which AIC value corresponds to which model when performing
```

```
colon_models4 = map(colon_covariates4, \(v) coxph(as.formula(paste("Surv(start, stop, status) ~ rx + clu

colon_aic_lvl4 = map_dbl(colon_models4, AIC) |>
  sort()

colon_aic_lvl4
```

```
##   adhere   differ obstruct   perfor      age      sex
## 12021.81 12022.21 12023.73 12027.16 12028.28 12028.45
```

The model with the *adhere* covariate, the binary variable of whether the cancer had adhered to other organs, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 5:
# Construct a list of covariates to put into the models:
colon_covariates5 = c("obstruct", "differ",  "perfor", "sex", "age")

# Build a model per covariate by pasting the given covariate into the formula.

# The set_names function helps to clarify up which AIC value corresponds to which model when performing

colon_models5 = map(colon_covariates5, \(v) coxph(as.formula(paste("Surv(start, stop, status) ~ rx + clu

colon_aic_lvl5 = map_dbl(colon_models5, AIC) |>
  sort()

colon_aic_lvl5
```

```
## obstruct   differ   perfor      sex      age
## 12018.62 12018.72 12023.02 12023.58 12023.65
```

The model with the *obstruct* covariate, the binary variable for whether the cancer had adhered to other organs, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 6:
# Construct a list of covariates to put into the models:
colon_covariates6 = c("differ",  "perfor", "sex", "age")

# Build a model per covariate by pasting the given covariate into the formula.

# The set_names function helps to clarify up which AIC value corresponds to which model when performing

colon_models6 = map(colon_covariates6, \(v) coxph(as.formula(paste("Surv(start, stop, status) ~ rx + clu

colon_aic_lvl6 = map_dbl(colon_models6, AIC) |>
  sort()

colon_aic_lvl6
```

```
##   differ      age   perfor      sex
## 12015.23 12020.16 12020.21 12020.52
```

The model with the *differ* covariate, the description of the removed cancer cells from the colon, had the lowest AIC.

Therefore, forward selection proceeds with the above covariate.

```
# Level 7:
# Construct a list of covariates to put into the models:
colon_covariates7 = c("perfor", "sex", "age")

# Build a model per covariate by pasting the given covariate into the formula.

# The set_names function helps to clarify up which AIC value corresponds to which model when performing

colon_models7 = map(colon_covariates7, \(v) coxph(as.formula(paste("Surv(start, stop, status) ~ rx + cl

colon_aic_lvl7 = map_dbl(colon_models7, AIC) |>
  sort()

colon_aic_lvl7
```

```
##      age   perfor      sex
## 12016.87 12016.90 12017.10
```

None of the models at this level have an AIC lower than the previous model, so the counting process model after AIC criterion has the following covariates, besides treatment and id: *node4*, *extent*, *surg*, *adhere*, *obstruct*, and *differ*