ETC5242Assignment

Sahinya Akila

9/4/2021

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(survival)
library(survminer)
```

```
## Loading required package: ggpubr
```

```r
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```r
library(knitr)
library(ggplot2)
## Remove the line break in the file name!
churn_dat <- read_csv("https://raw.githubusercontent.com/square/pysurvival/master/pysurvival/datasets/cl
```

```
## Rows: 2000 Columns: 14

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): product_travel_expense, product_payroll, product_accounting, compan...
## dbl (9): product_data_storage, csat_score, articles_viewed, smartphone_notif...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

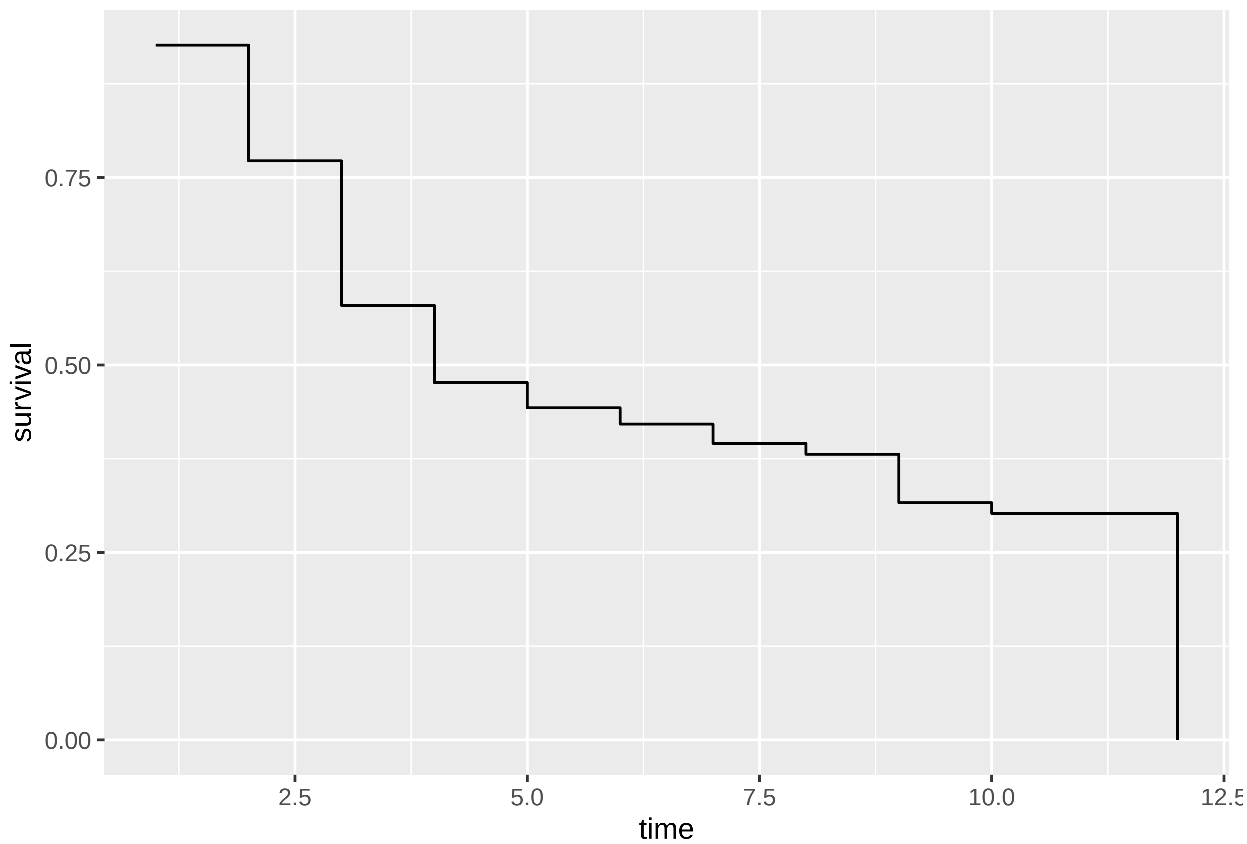
```
churn_dat <- churn_dat %>% filter(months_active > 0) %>% select(c(company_size, months_active, churned))
```

```
km_model <- function(time, event){
  dataset <- data_frame(time, event)

  km_data <- dataset %>%
    group_by(time, event) %>%
    summarise(died = n()) %>%
    ungroup() %>%
    mutate(risk = nrow(dataset) - accumulate(died, `+`) + died) %>%
    filter(event == 1) %>%
    mutate(probability = 1 - died/risk,
           survival = accumulate(probability, `*`))
  return(km_data %>% select(time, survival))
}

km_survive <- km_model(churn_dat$months_active, churn_dat$churned)
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## Please use `tibble()` instead.
```

```
## `summarise()` has grouped output by 'time'. You can override using the `.groups` argument.
```

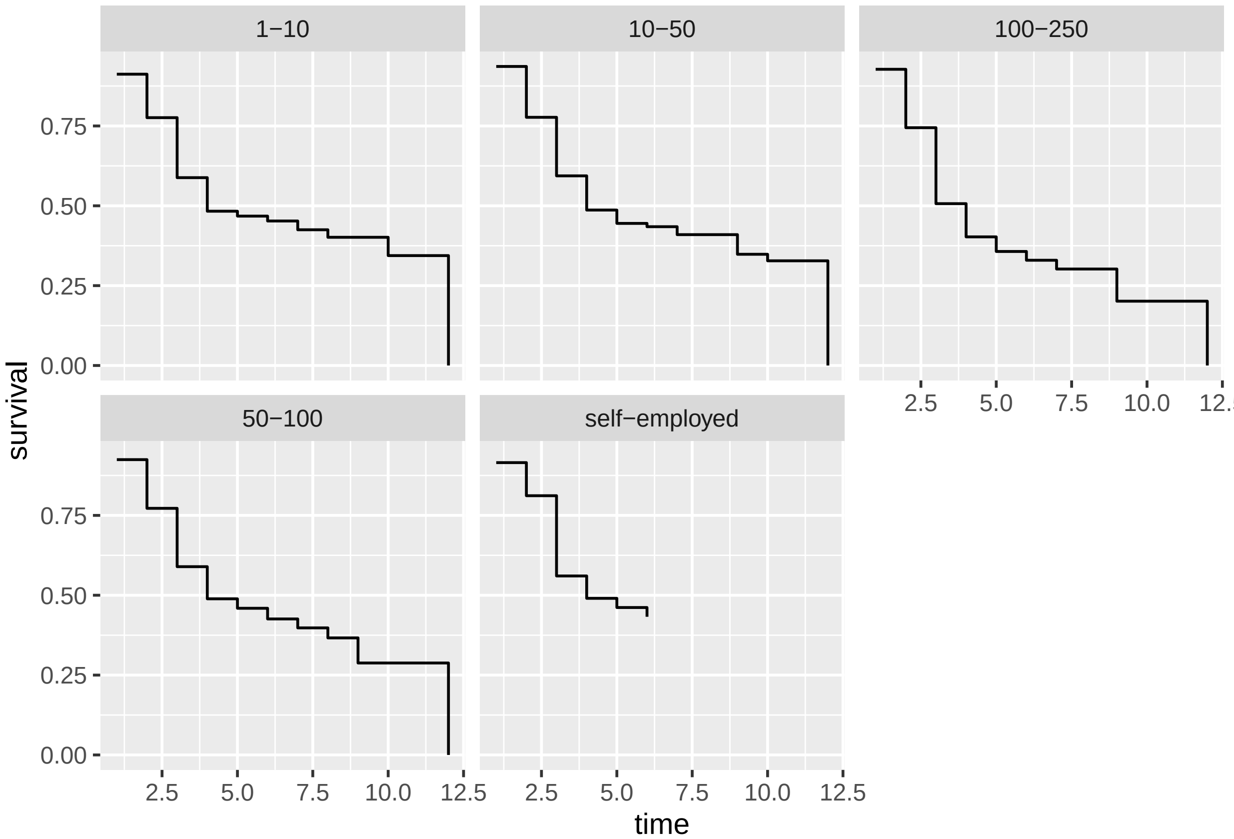
```
km_survive %>%
  ggplot(aes(time, survival)) +
  geom_step()
```




```
company_km_model <- data.frame(time = double(), survival = double(), company_size = character())
for(size in unique(churn_dat$company_size)){
  filtered <- churn_dat %>% filter(company_size == size)
  final_model <- km_model(filtered$months_active, filtered$churned) %>% mutate(company_size = size)
  company_km_model <- rbind(company_km_model, final_model)
}
```

```
## `summarise()` has grouped output by 'time'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'time'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'time'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'time'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'time'. You can override using the `.groups` argument.
```

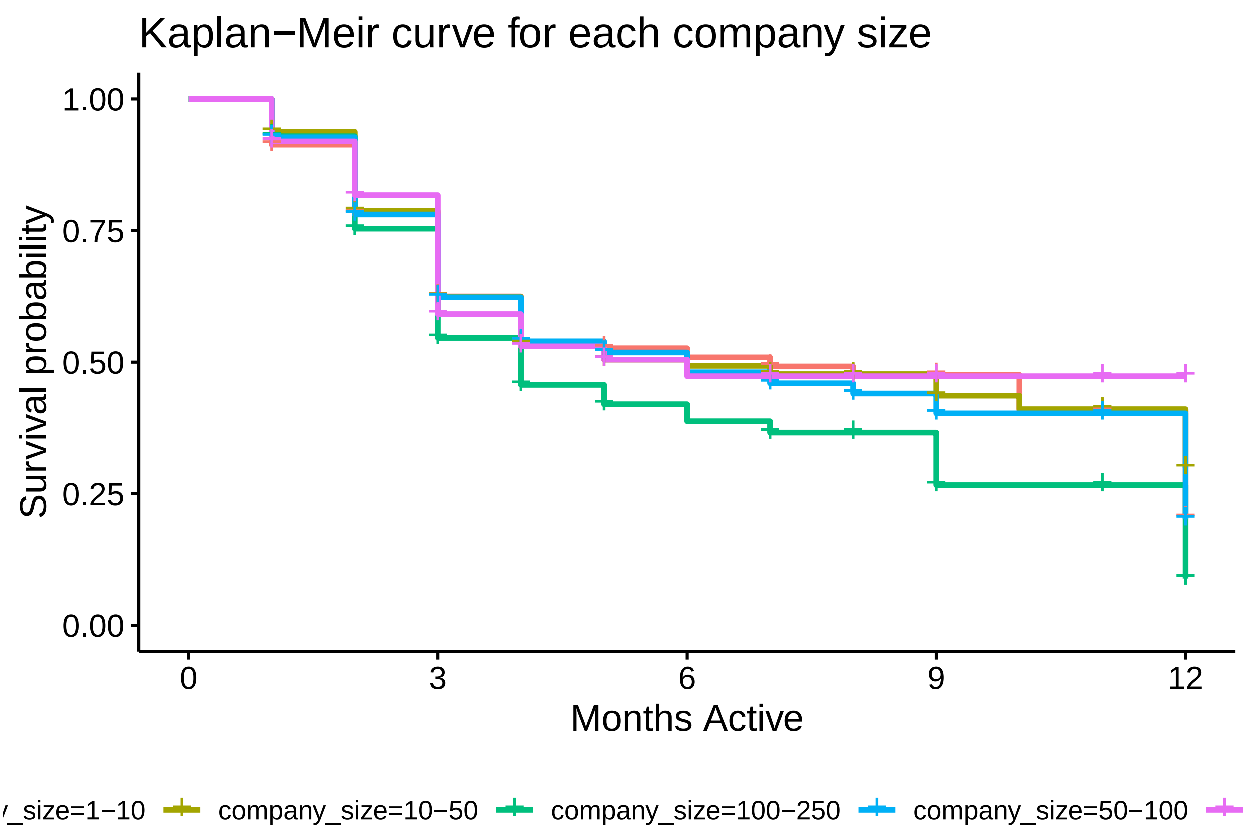
```
company_km_model %>%
  ggplot(aes(time, survival)) +
  geom_step() +
  facet_wrap(~company_size)
```



Question 2

Compute the Kaplan-Meir curve and use this to estimate the median churn time.

```
fit <- survfit(Surv(months_active, churned) ~ company_size,
                  data = churn_dat)

g2 <- ggsurvplot(fit,conf.int = FALSE)

g2$plot +
  labs(x = "Months Active",
       title = "Kaplan-Meir curve for each company size") +
  theme(legend.position="bottom")
```




```r
median_function <- function(fit){
  index <- which.min(abs(fit$surv - 0.5))
  median <- fit$time[index]
  return(median)
}
```

```r
for (size in unique(churn_dat$company_size)){
  temp_data <- churn_dat %>% filter(company_size == size)
  name <- size
  assign(name, survfit(Surv(months_active, churned) ~ company_size, data = temp_data))
}
```

```r
company_median <- data_frame(company_size = unique(churn_dat$company_size), median = c(NA, NA, NA, NA, N

for (i in 1:length(company_median$company_size)){
  company_median$median[i] <- median_function(get(company_median$company_size[i]))
}

company_median %>%
  knitr::kable(
  caption = "Medians for different company sizes") %>%
  kable_styling(c("hover", "striped"))
```

The table above demonstrates the median churn time estimated for different company size. - Company size of 1-10 have the highest estimated median of 7 months. - Company size of 100-250 have the lowest estimated median of 4 months. - The rest of the company sizes have the same estimated median of 5 months.

Table 1: Medians for different company sizes

company_size	median
10-50	5
100-250	4
50-100	5
1-10	7
self-employed	5

Use a non-parametric bootstrap to construct 90% confidence intervals for the median of each company size

```
bootstrapmedian <- function(df_median, df){
  bootstrap <- tibble(experiment = rep(1:1000, each = nrow(df)),
                      ind = sample(1:nrow(df), size = nrow(df)*1000, replace = TRUE),
                      timestar = df$months_active[ind],
                      churnstar = df$churned[ind])

  bias <- bootstrap %>%
    group_by(experiment) %>%
    summarise(delta = median_function(df_median) - median_function(survfit(Surv(timestar, churnstar) ~ 

  ci <- median_function(df_median) + quantile(bias$delta, c(0.05, 0.95))

  return(ci)
}
```

```
company_median_ci <- data_frame(company_size = unique(churn_dat$company_size), median = c(NA, NA, NA, NA

for (i in 1:length(company_median_ci$company_size)){
  ci <- bootstrapmedian(get(company_median_ci$company_size[i]), churn_dat %>% filter(company_size == co
  company_median_ci$median[i] <- median_function(get(company_median_ci$company_size[i]))
          company_median_ci$lci[i] <- ci[1]
          company_median_ci$uci[i] = ci[2]
}
company_median_ci %>%
  knitr::kable(
  caption = "estimated mean under 90% CI") %>%
  kable_styling(c("hover", "striped"))
```

\begin{table}

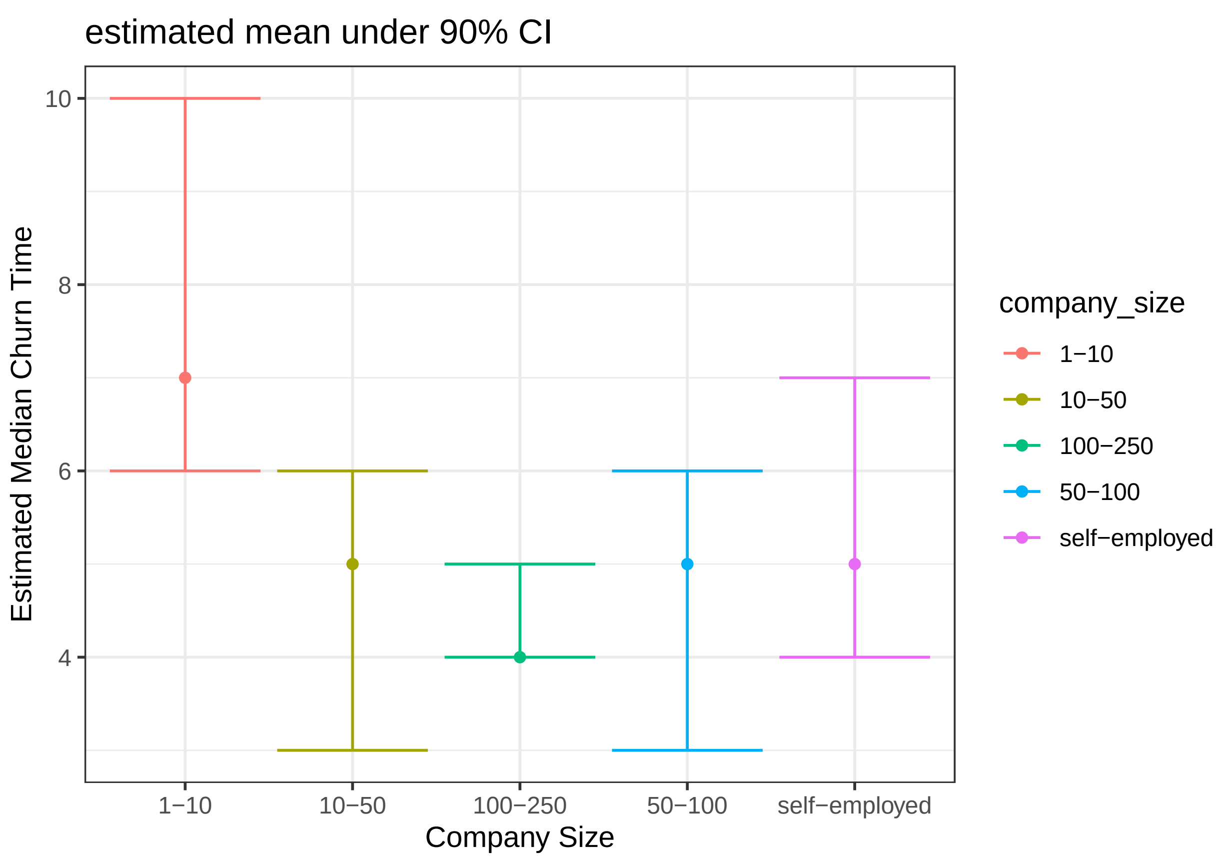
\caption{estimated mean under 90% CI}

company_size	median	lci	uci
10-50	5	3	6
100-250	4	4	5
50-100	5	3	6
1-10	7	6	10
self-employed	5	4	7

\end{table}

Make a plot that shows that estimate of the median and the corresponding confidence interval on the same axes

```
ggplot(company_median_ci,
       aes(x = company_size,
           y = median,
           colour = company_size)) +
  geom_errorbar(aes(ymax = uci, ymin = lci)) +
  geom_point() +
  theme_bw() +
  labs(x = "Company Size",
       y = "Estimated Median Churn Time",
       title = "estimated mean under 90% CI")
```



Q3 - Use a nonparametric bootstrap to re-sample the data and construct 90% confidence intervals for the survival curve at each time.

```
mean_ci(Surv(churn_dat$months_active, churn_dat$churned)) %>%
  knitr::kable(
  caption = "Minimum and Maximum value in the survival curve") %>%
  kable_styling(c("hover", "striped"))
```

```
is_in_ci <- function(max_y, true_max, n, n_sim = 1000) {
bootstrap <- tibble(experiment = rep(1:n_sim,
```

Table 2: Minimum and Maximum value in the survival curve

y	ymin	ymax
2.219356	2.111533	2.32718

```
each = n),
ystar = runif(n * n_sim, 0, max_y))
bias <- bootstrap %>%
group_by(experiment) %>%
summarise(delta = max_y - max(ystar))
int <- max_y + quantile(bias$delta, c(0.05, 0.95))
return((true_max > int[1]) & (true_max < int[2]))
}
```

Check the coverage

```
true_max <- 2.32718

experiments <- tibble(experiment = rep(1:1000,
each = 100),
draw = runif(100*1000,0,true_max))
test <- experiments %>%
group_by(experiment) %>%
summarise(max_y = max(draw)) %>%
mutate(is_in = map_dbl(max_y,
~is_in_ci(.x, true_max, 100)))
mean(test$is_in)
```

```
## [1] 0.911
```

Question 3

Choose company size of 50-100

```
q3_company <- churn_dat %>%
  filter(company_size == "50-100")

q3_fit <- survfit(Surv(months_active, churned) ~1,
                  data = q3_company)
```

Use a nonparametric bootstrap to re-sample the data and construct 90% confidence intervals for the survival curve at each time.

```
bootstrap_time <- tibble(experiment = rep(1:1000, each =672),
                        ind = sample(1:672, size = 672*1000, replace = TRUE),
                                    months_active = q3_company$months_active[ind],
```

```r
                                          churned = q3_company$churned[ind])
bias_time <- bootstrap_time %>%
  group_by(experiment) %>%
  summarise(delta = q3_fit$surv - survfit(Surv(months_active, churned) ~1)$surv)
```

```
## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length
```

```
## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## Warning in q3_fit$surv - survfit(Surv(months_active, churned) ~ 1)$surv: longer
## object length is not a multiple of shorter object length

## ‘summarise()‘ has grouped output by ’experiment’. You can override using the ‘.groups‘ argument.
```

```r
lowerCIs <- q3_fit$surv + quantile(bias_time$delta, 0.05)
upperCIs <- q3_fit$surv + quantile(bias_time$delta, 0.95)
```

```r
Month <- c(1:11)

time_50_100_CIs <- data.frame(Month, q3_fit$surv, lowerCIs, upperCIs) %>%
  rename("Probability" = q3_fit.surv, "Lower Confidence Interval" = lowerCIs, "Upper Confidence Interval

kable(time_50_100_CIs,
      caption = "90% confidence intervals for the survival curve at each time for company size 50-100")
  kable_styling(c("hover", "striped"))
```

\begin{table}

\caption{90% confidence intervals for the survival curve at each time for company size 50-100}

Month	Probability	Lower Confidence Interval	Upper Confidence Interval
1	0.9270833	0.8784632	0.9783529
2	0.7805394	0.7319192	0.8318090
3	0.6231333	0.5745132	0.6744029
4	0.5395180	0.4908978	0.5907876
5	0.5183604	0.4697403	0.5696300
6	0.4807375	0.4321173	0.5320071
7	0.4598358	0.4112157	0.5111054
8	0.4404062	0.3917860	0.4916758
9	0.4026571	0.3540369	0.4539267
10	0.4026571	0.3540369	0.4539267
11	0.2013285	0.1527084	0.2525981

\end{table}

Compute simultaneous coverage for the entire survival function.

Q4

log-rank test

```r
q4_comp <- churn_dat %>%
 mutate(comp_hyp = case_when(company_size == "50-100" ~ 1, company_size== "100-250" ~ 2, TRUE ~ 0))

q4_comp <- q4_comp %>%
  filter(comp_hyp == 1| comp_hyp == 2)
```

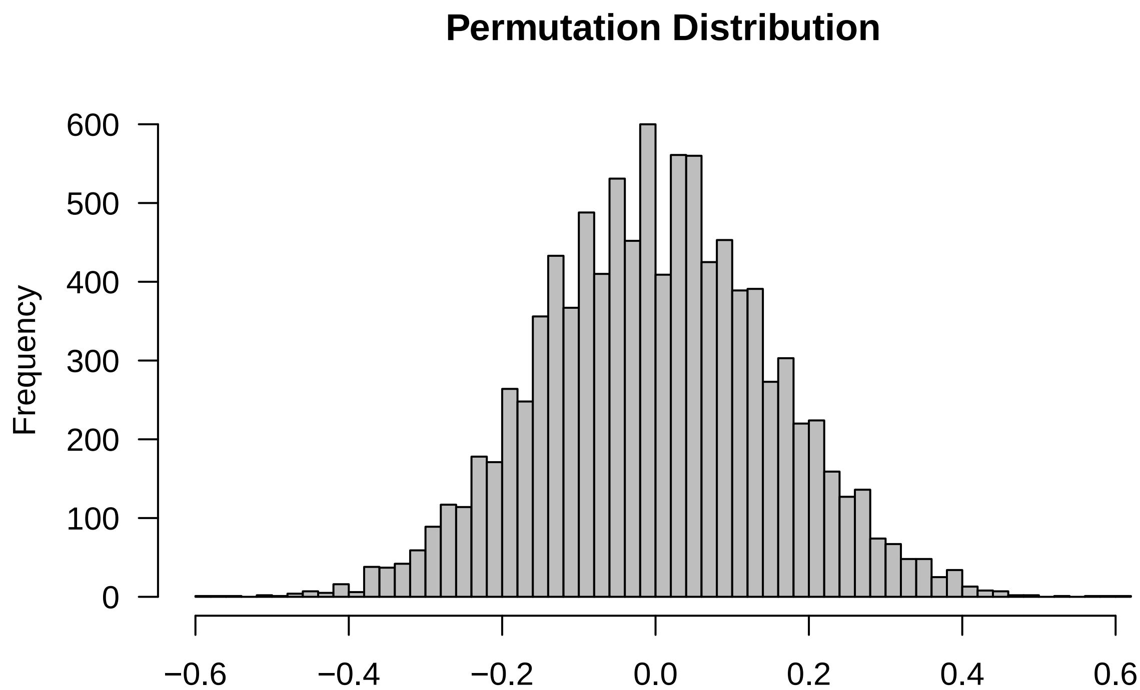
```r
survdiff(Surv(months_active, churned) ~ comp_hyp, data=q4_comp)
```

```
## Call:
## survdiff(formula = Surv(months_active, churned) ~ comp_hyp, data = q4_comp)
##
##              N Observed Expected (O-E)^2/E (O-E)^2/V
## comp_hyp=1 672      313      332      1.14      5.26
## comp_hyp=2 240      135      116      3.27      5.26
##
##  Chisq= 5.3  on 1 degrees of freedom, p= 0.02
```

```r
treatment <- q4_comp$churned
outcome <- q4_comp$months_active

#Difference in means
original <- diff(tapply(outcome, treatment, mean))
mean(outcome[treatment==1])-mean(outcome[treatment==0])
```

```
## [1] -1.896937
```

```r
#Permutation test
permutation.test <- function(treatment, outcome, n){
  distribution=c()
  result=0
  for(i in 1:n){
    distribution[i]=diff(by(outcome, sample(treatment, length(treatment), FALSE), mean))
  }
  result=sum(abs(distribution) >= abs(original))/(n)
  return(list(result, distribution))
}

test1 <- permutation.test(treatment, outcome, 10000)
hist(test1[[2]], breaks=50, col='grey', main="Permutation Distribution", las=1, xlab='')
abline(v=original, lwd=3, col="red")
```




```
test1[[1]]
```

```
## [1] 0
```

```
#Compare to t-test
t.test(outcome~treatment)
```

```
##
##  Welch Two Sample t-test
##
## data:  outcome by treatment
## t = 13.702, df = 842.56, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.625195 2.168678
## sample estimates:
## mean in group 0 mean in group 1
##        4.823276        2.926339
```

Question 5

fit a Weibull distribution to the survival data to estimate the mean and the median of the churn time for each company size

```
size1 <- churn_dat %>%
  filter(company_size == "10-50")

size2 <- churn_dat %>%
  filter(company_size == "50-100")

size3 <- churn_dat %>%
  filter(company_size == "100-250")

size4 <- churn_dat %>%
  filter(company_size == "self-employed")
```

```
fit_q5 <- function(dat){
  fit <- survreg(Surv(months_active, churned) ~ 1, data = dat, dist = "weibull" )
  rweibull_shape <- 1 / fit$scale ## Approximately 3
  rweibull_scale <- exp(coef(fit)) ## approximately 7

print(rweibull_scale*log(2)^(1/rweibull_shape))
print(rweibull_scale*gamma(1+(1/rweibull_shape)))

}
```

```
weibull1<- fit_q5(size1)
```

```
## (Intercept)
##    5.691844
## (Intercept)
##    6.790794
```

```
weibull2<- fit_q5(size2)
```

```
## (Intercept)
##    5.561796
## (Intercept)
##    6.611866
```

```
weibull3<- fit_q5(size3)
```

```
## (Intercept)
##    4.701316
## (Intercept)
##    5.448748
```

```r
weibull4<- fit_q5(size4)
```

```
## (Intercept)
##    6.226591
## (Intercept)
##    7.924842
```

```r
weibull <- data.frame(weibull1, weibull2, weibull3, weibull4)

kable(weibull)
```

	weibull1	weibull2	weibull3	weibull4
(Intercept)	6.790794	6.611866	5.448748	7.924842