Computer Lab 5 Computational Statistics

Phillip Hölscher & Zijie Feng

23-2-2019

Contents

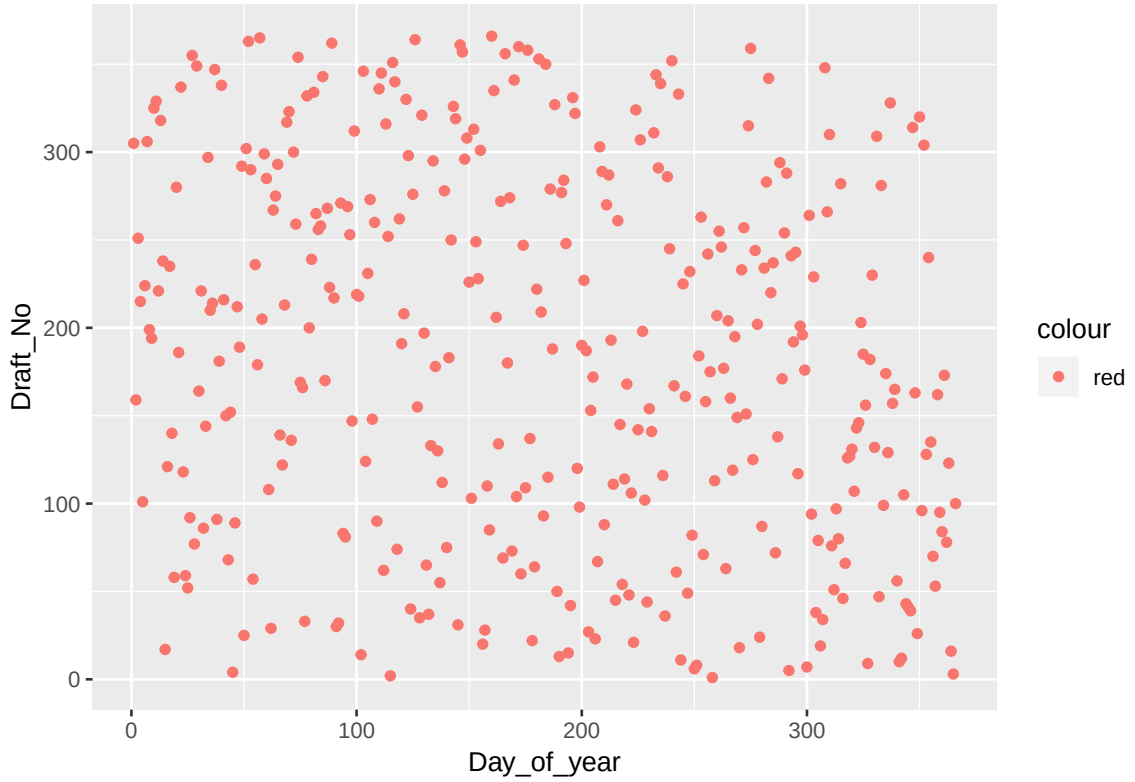
#Question 1: Hypothesis testing

In 1970, the US Congress instituted a random selection process for the military draft. All 366 possible birth dates were placed in plastic capsules in a rotating drum and were selected one by one. The first date drawn from the drum received draft number one, the second date drawn received draft number two, etc. Then, eligible men were drafted in the order given by the draft number of their birth date. In a truly random lottery there should be no relationship between the date and the draft number. Your task is to investigate whether or not the draft numbers were randomly selected. The draft numbers (`Y=Draft_No`) sorted by day of year (`X=Day_of_year`) are given in the file *lottery.xls*.

1. Make a scatterplot

of Y versus X and conclude whether the lottery looks random.

Scatterplot of Y versus X

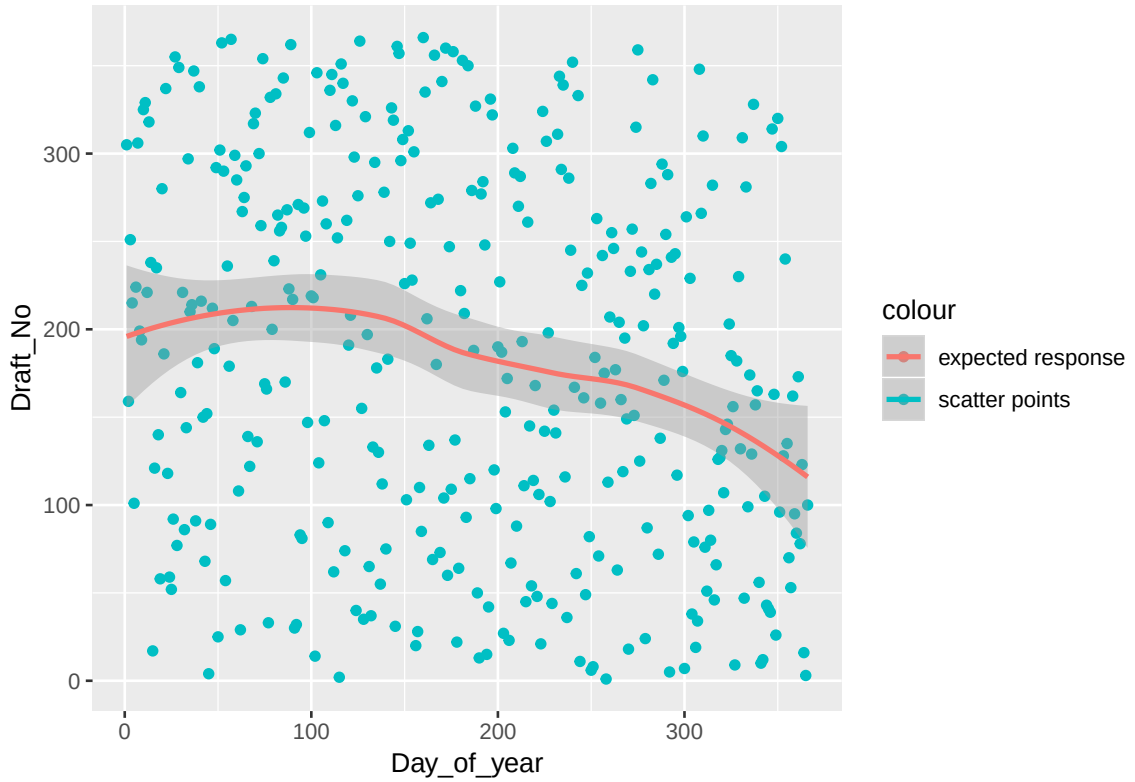


Since the dots are spaced over the whole area, it seems like the military draft is truly random.

2. Compute an estimate

$\hat{Y}$ of the expected response as a function of X by using a loess smoother (use `loess()`), put the curve $\hat{Y}$ versus X in the previous graph and state again whether the lottery looks random.

Scatterplot of $\hat{Y}$ versus X



It does look like the loess smoother decreases slowly after X (= Day of year) passes number 100. This could mean the draft is not truly random.

3. Check whether the lottery is random

To check whether the lottery is random, it is reasonable to use test statistics

$$T = \frac{\hat{Y}(X_b) - \hat{Y}(X_a)}{X_b - X_a}, \quad \text{where} \quad \mathrm{X_b} = \mathrm{argmax_X Y(X)}, \quad \mathrm{X_a} = \mathrm{argmin_X Y(X)}$$

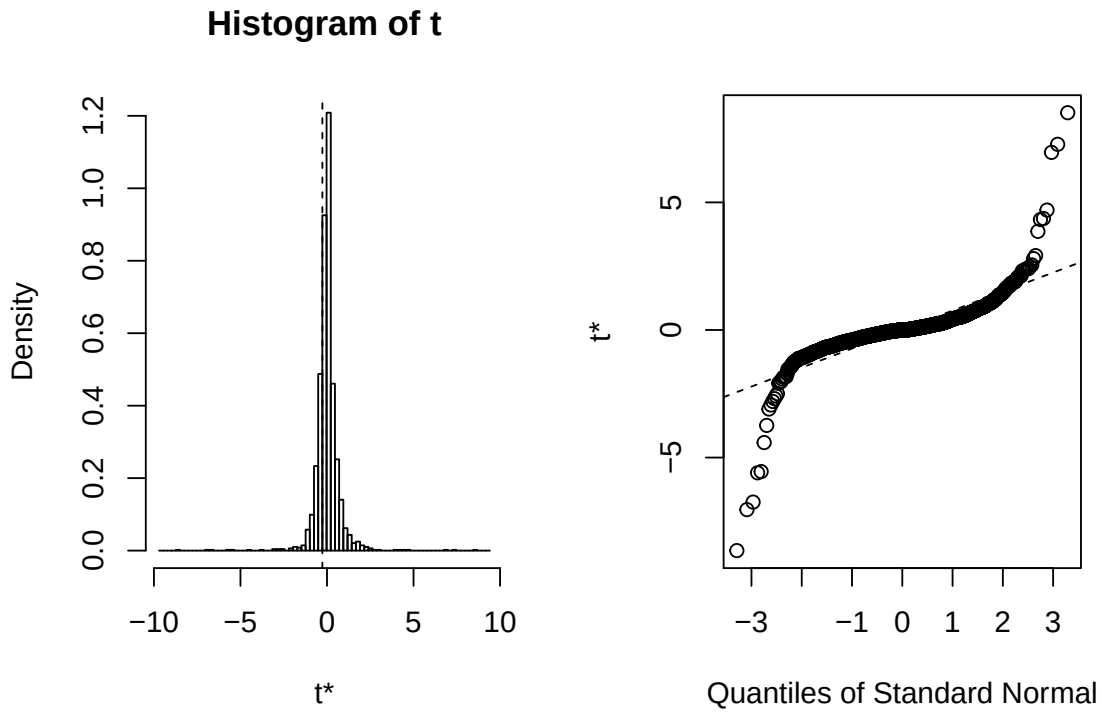
If this value is significantly greater than zero, then there should be a trend in the data and the lottery is not random. Estimate the distribution of T by using a non–parametric bootstrap with $B = 2000$ and comment whether the lottery is random or not. What is the p–value of the test?

```
# use non-parametric bootstrap
# need data = data1, statistic = loess & test stat, R = 2000
stats = function(data,vn){
  data<-data[vn,]
  # Y=Draft_No & X=Day_of_year
  # create the loess regression
  reg = loess(Draft_No ~ Day_of_year, data = data)
  # the X which owns the max/min Y in original data
  X_b = data$Day_of_year[which.max(data$Draft_No)]
  X_a = data$Day_of_year[which.min(data$Draft_No)]
  # Y_hat(X)
  fit_X_b = reg$fitted[X_b]
  fit_X_a = reg$fitted[X_a]
  # the given test statistic
```

```r
  T_stat = (fit_X_b - fit_X_a) / (X_b-X_a)
  return(T_stat)
}

#bootstrap
set.seed(123456)
myboot = boot(data = data1,
              statistic = stats,
              R = 2000)

# summary
myboot
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data1, statistic = stats, R = 2000)
##
##
## Bootstrap Statistics :
##       original   bias    std. error
## t1* -0.2671794 0.282995   0.7447472
```

```r
# plot distribution
plot(myboot, index = 1)
```



The test statistic T is around -0.267, which is smaller than 0. So the lottery is random. We used following

calculation to estimate the p-value:

$$\hat{p} = \frac{\sum T(X) \geqslant 0}{B}$$

```
## [1] 0.4865
```

Since p=value is larger than significant level $\alpha = 0.05$, we cannot reject the null hypothesis (it is not random).

4. Tests the hypothesis

Implement a function depending on *data* and B that tests the hypothesis - *H0*: Lottery is random versus versus

- *H1*: Lottery is non–random

by using a permutation test with statistics `T`. The function is to return the p–value of this test. Test this function on our data with 'B = 2000.

```
set.seed(123456)
permutation_test <- function(B, data1){
  n = dim(data1)[1]
  stat = numeric(B)
  for(b in 1:B){
    # randoming X
    Gb = sample(data1$Day_of_year, n)
    data11 <- data1
    data11$Day_of_year <- Gb
    stat[b] <- stats(data11,1:n)
  }
  return(stat)
}

stat <- permutation_test(B=2000, data1)

stat0 <- stats(data1,1:nrow(data1))
print (c( stat0 ,mean(abs(stat)>=abs(stat0)) ))
```

```
## [1] -0.2671794  0.1750000
```

Since the the p-value is still larger than 0.05, we cannot reject the null hypothesis either.

5.Make a crude estimate

Make a crude estimate of the power of the test constructed in Step 4:

(a) Generate (an obviously non{random) dataset with n = 366 observations by using same X as in the original data set and $Y(x) = \max(0, \min(\alpha x + \beta, 366))$, where $\alpha = 0, 1$ and $\beta \sim N(183, sd = 10)$.

(b) Plug these data into the permutation test with B = 200 and note whether it was rejected.

(c) Repeat Steps 5a-5b for $\alpha = 0.2, 0.3, ..., 10$.

What can you say about the quality of your test statistics considering the value of the power?

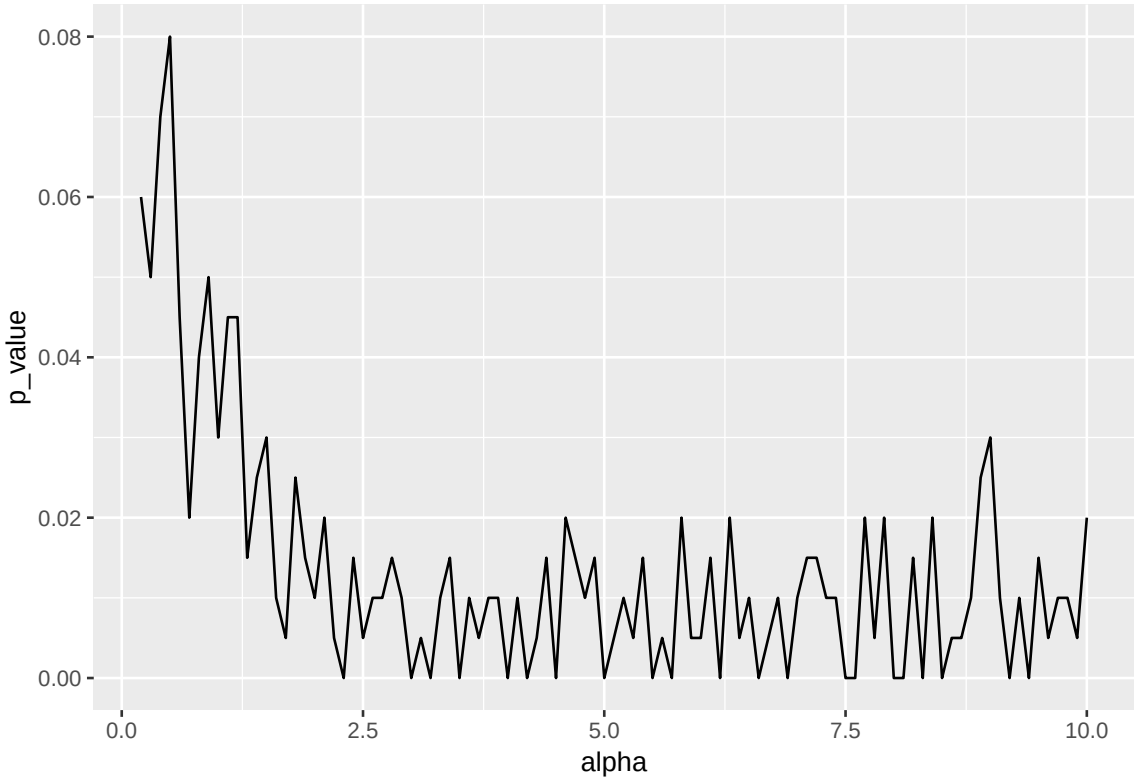
```
fun <- function(a){
  ## step (a)
  n <- 366
  b <- rnorm(n, mean=183, sd = 10)
```

```
  X <- data1$Day_of_year
  Y <- sapply(1:n,function(i){
    t <- min(a*X[i]+b[i],366)
    t <- max(0,t)
  })
  data15 <- data.frame(Day_of_year=X,Draft_No=Y)

  ## step (b)
  stat <- permutation_test(B=200, data15)

  stat0 <- stats(data15,1:n)
  return(c( stat0 ,mean(abs(stat)>=abs(stat0)) ))
}

fun(a=0.1)
```

```
## [1] 0.1082525 0.0650000
```

Since the p-value here is even smaller than 0.01, so we can reject the null hypothesis directly, which means the data is non-random.

```
## step (c)
result <- sapply(seq(0.2,10,0.1), fun)
result<- data.frame(alpha=seq(0.2,10,0.1), p_value=result[2,])
ggplot(data = result,aes(x=alpha,y=p_value))+
  geom_line()
```



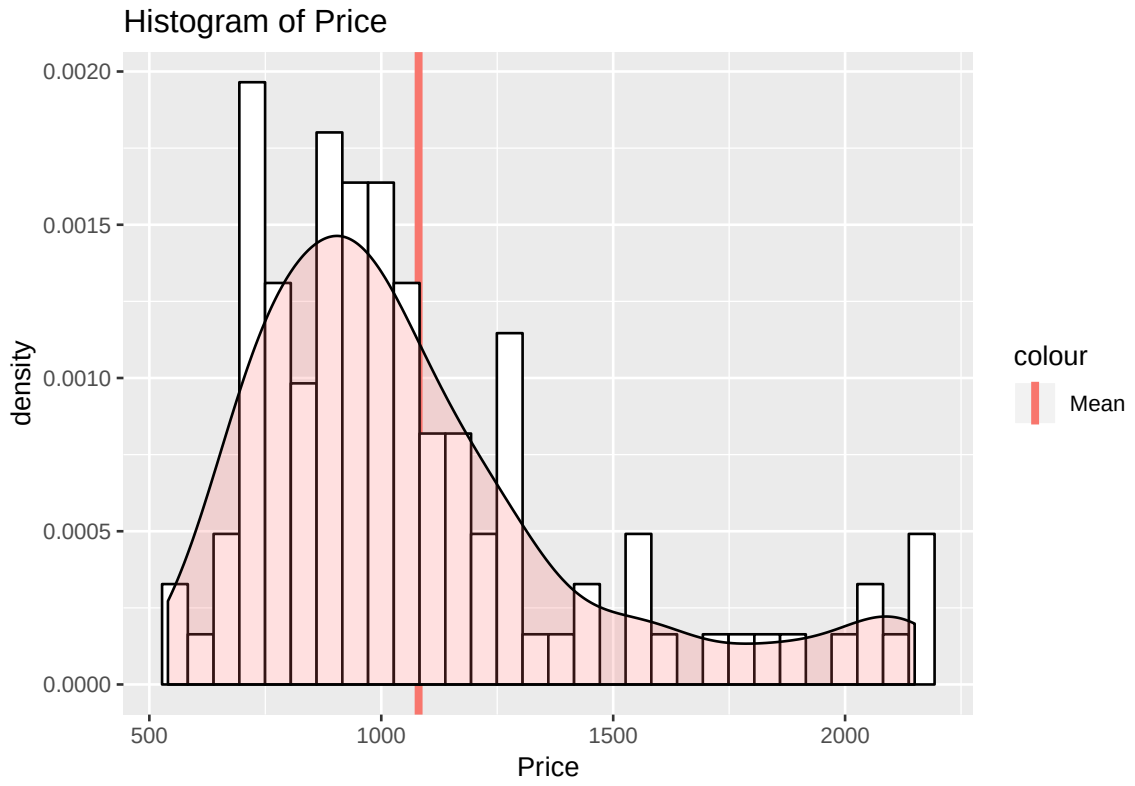
It seems that when α is larger, our test statistics is efficient to reject our null hypothesis (non-random).

Question 2: Bootstrap, jackknife and confidence intervals

The data you are going to continue analyzing is the database of home prices in Albuquerque, 1993. The variables present are `Price`; `SqFt`: the area of a house; `FEATS`: number of features such as dishwasher, refrigerator and so on; 'Taxes: annual taxes paid for the house. Explore the file *prices1.xls*.

1. Plot the histogram of Price.

Does it remind any conventional distribution? Compute the mean price.



The present distribution looks like a possion or gamma distribution.

The mean of the price is:

```
price_mean
```

```
## [1] 1080.473
```

2. Bootstrap

Estimate the distribution of the mean price of the house using bootstrap. Determine the bootstrap bias-correction and the variance of the mean price. Compute a 95% confidence interval for the mean price using bootstrap percentile, bootstrap BCa, and first-order normal approximation.

Estimate the distribution of the mean price of the house using bootstrap:

```
# function - estiamte mean
stat1 <- function(vec,vn){
    return(mean(vec[vn]))
```

```
}
B=1000

set.seed(123456)
res1 = boot(data2$Price, stat1, R=B)
res1
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data2$Price, statistic = stat1, R = B)
##
##
## Bootstrap Statistics :
##     original   bias    std. error
## t1* 1080.473 1.696191    36.15702
```

According to the slide, we know that the bias-corrected estimator is

$$T_1 := 2T(D) - \frac{1}{B}\sum_{i=1}^{B} T_i^* \ ,$$

and variance of estimator by bootstrap is

$$\widehat{Var[T(.)]} = \frac{1}{B-1}\sum_{i=1}^{B}(T(D_i^*) - \overline{T(D^*)})^2.$$

Determine the bootstrap bias-correction and the variance of the mean price:

```
# bias correcred estimator
2*res1$t0-mean(res1$t)
```

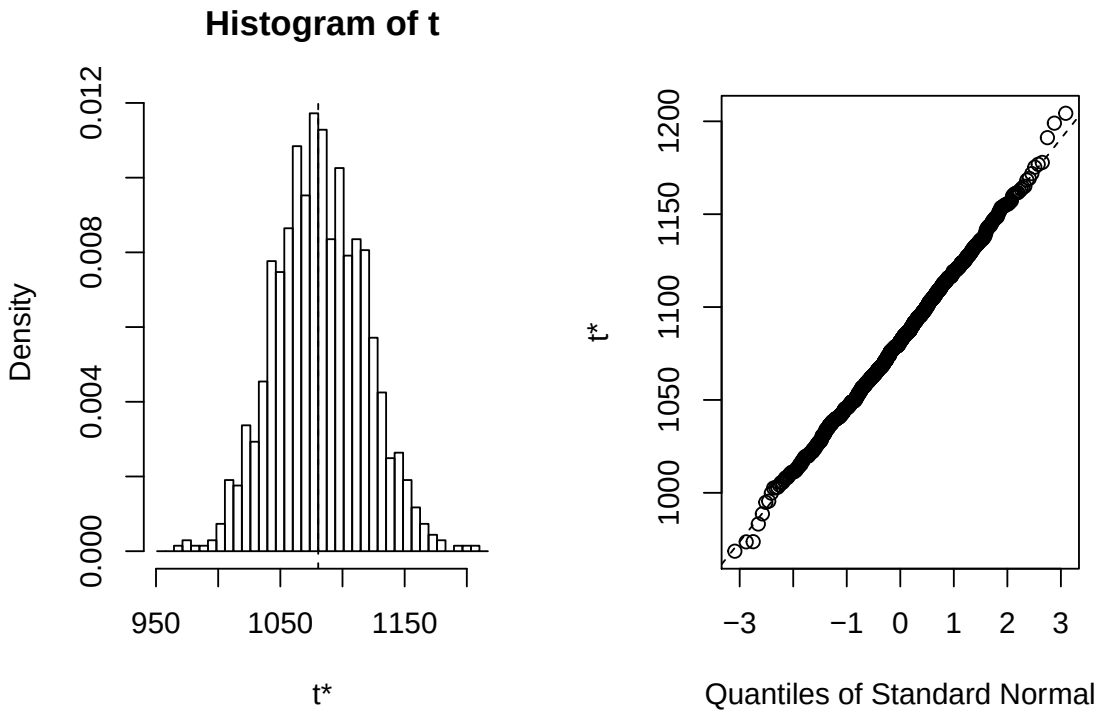
```
## [1] 1078.777
```

```
# variance of mean price (output of statistic)
var_boot <- 1/(B-1)*sum((res1$t-mean(res1$t))^2 )
var_boot
```

```
## [1] 1307.33
```

Compute a 95% confidence interval for the mean price using bootstrap percentile, bootstrap BCa, and first-order normal approximation

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = res1, type = c("perc", "bca", "norm"))
##
## Intervals :
## Level      Normal             Percentile            BCa
## 95%   (1008, 1150 )   (1012, 1155 )   (1014, 1155 )
## Calculations and Intervals on Original Scale
```



3. Jackknife

Estimate the variance of the mean price using the jackknife and compare it with the bootstrap estimate.

- Jackknife (n = B)

$$\widehat{Var[T(.)]} = \frac{1}{n(n-1)} \sum_{i=1}^{n} ((\mathrm{T_i^*}) - \mathrm{J(T)})^2,$$

where

$$T_i^* = nT(D) - (n-1)T(D_i^*)$$

and

$$J(T) = \frac{1}{n} \sum_{i=1}^{n} T_i^*$$

```
# other jackknife function
# library(bootstrap)
# res2 <- jackknife(x=data2$Price,stat1)
# mean(res2$jack.values)

n = nrow(data2)
constant = 1/(n*(n-1))

T_i_star = sapply(1:n, function(i){
  n * mean(data2$Price) - (n-1) * mean(data2[-i,1])
})
```

```r
J_T = (1/n) * sum(T_i_star)

Var_T_jackknife = constant * sum((T_i_star - J_T)^2)
Var_T_jackknife
```

```
## [1] 1320.911
```

boostrap	jackknife
1307.33	1320.911
It seems th	at bootstrap here is better than jackknife because of its smaller variance.

4. Compare the confidence intervals obtained

with respect to their length and the location of the estimated mean in these intervals.

	Len.CI	Esti.mean
percent	143.0140	1083.540
bca	141.9492	1084.496
norm	141.7329	1078.777

On the one hand, percentile has the largest confidence intervals, but BCa and normal approximation have similar confidence intervals. On the other hand, the interval-means of percentile and BCa are similar (≈ 1084), but of normal approximation is around 1079.

Appendix

```r
Sys.setlocale(locale = "English")
knitr::opts_chunk$set(echo = TRUE, out.height = "300px")
# libraries used in this lab
library(ggplot2)
#install.packages("boot") # to create booststarp
library(boot)
# scatterplot
rm(list=ls())
data1 = read.csv2("lottery.csv")

plot_11 = ggplot(data = data1,aes(x = Day_of_year, y = Draft_No)) +
  geom_point(aes(color = "red"))
plot_11
# loess(formula = Draft_No~Day_of_year, data = data1)

plot_11 +
  # geom_smooth()` using method = 'loess' and formula 'y ~ x'
  geom_smooth(method ="loess", formula = y~x, aes(color="blue")) +
  scale_color_discrete(labels=c("expected response",
                                "scatter points")
                       )
# use non-parametric bootstrap
# need data = data1, statistic = loess & test stat, R = 2000
stats = function(data,vn){
  data<-data[vn,]
  # Y=Draft_No & X=Day_of_year
  # create the loess regression
  reg = loess(Draft_No ~ Day_of_year, data = data)
  # the X which owns the max/min Y in original data
  X_b = data$Day_of_year[which.max(data$Draft_No)]
  X_a = data$Day_of_year[which.min(data$Draft_No)]
  # Y_hat(X)
  fit_X_b = reg$fitted[X_b]
  fit_X_a = reg$fitted[X_a]
  # the given test statistic
  T_stat = (fit_X_b - fit_X_a) / (X_b-X_a)
  return(T_stat)
}

#bootstrap
set.seed(123456)
myboot = boot(data = data1,
              statistic = stats,
              R = 2000)

# summary
myboot

# plot distribution
plot(myboot, index = 1)
# calculate the p-value
```

```r
# null hypothesis: T>=0, hv tendancy
p_val = mean(myboot$t>=0, na.rm = TRUE)
p_val
set.seed(123456)
permutation_test <- function(B, data1){
  n = dim(data1)[1]
  stat = numeric(B)
  for(b in 1:B){
    # randoming X
    Gb = sample(data1$Day_of_year, n)
    data11 <- data1
    data11$Day_of_year <- Gb
    stat[b] <- stats(data11,1:n)
  }
  return(stat)
}

stat <- permutation_test(B=2000, data1)

stat0 <- stats(data1,1:nrow(data1))
print (c( stat0 ,mean(abs(stat)>=abs(stat0)) ))
fun <- function(a){
  ## step (a)
  n <- 366
  b <- rnorm(n, mean=183, sd = 10)
  X <- data1$Day_of_year
  Y <- sapply(1:n,function(i){
    t <- min(a*X[i]+b[i],366)
    t <- max(0,t)
  })
  data15 <- data.frame(Day_of_year=X,Draft_No=Y)

  ## step (b)
  stat <- permutation_test(B=200, data15)

  stat0 <- stats(data15,1:n)
  return(c( stat0 ,mean(abs(stat)>=abs(stat0)) ))
}

fun(a=0.1)
## step (c)
result <- sapply(seq(0.2,10,0.1), fun)
result<- data.frame(alpha=seq(0.2,10,0.1), p_value=result[2,])
ggplot(data = result,aes(x=alpha,y=p_value))+
  geom_line()
rm(list = ls())
data2 = read.csv2("prices1.csv")
# calculate the mean of the price
price_mean = mean(data2$Price) # 1080.473

# create the plot - hist + distribution + mean
ggplot(data = data2, aes(x = Price)) +
  #geom_histogram(color = "#33CCFF", fill = "#33CCFF") +
```

```
  ggtitle("Histogram of Price") +
  geom_vline(aes(xintercept = price_mean, color = "Mean"),size=1.5) +
  geom_histogram(aes(y=..density..),
                 colour="black",
                 fill="white",
                 bins=30)+
  geom_density(alpha=.2, fill="#FF6666")
  # geom_density(kernel = "poisson") create a possion distribution
price_mean
# function - estiamte mean
stat1 <- function(vec,vn){
    return(mean(vec[vn]))
}
B=1000

set.seed(123456)
res1 = boot(data2$Price, stat1, R=B)
res1
# bias correcred estimator
2*res1$t0-mean(res1$t)

# variance of mean price (output of statistic)
var_boot <- 1/(B-1)*sum((res1$t-mean(res1$t))^2 )
var_boot

# default is a 95% confidence interval
ci <- boot.ci(res1, type = c("perc", "bca", "norm"))
print(ci)

plot(res1)
# other jackknife function
# library(bootstrap)
# res2 <- jackknife(x=data2$Price,stat1)
# mean(res2$jack.values)

n = nrow(data2)
constant = 1/(n*(n-1))

T_i_star = sapply(1:n, function(i){
  n * mean(data2$Price) - (n-1) * mean(data2[-i,1])
})

J_T = (1/n) * sum(T_i_star)

Var_T_jackknife = constant * sum((T_i_star - J_T)^2)
Var_T_jackknife
table = data.frame(boostrap = var_boot, jackknife= Var_T_jackknife)
knitr::kable(table)
percent<-ci$percent
bca<-ci$bca
norm<-ci$normal
intervals = rbind(percent[4:5],bca[4:5],norm[2:3])
Len.CI=intervals[,2]-intervals[,1]
```

```
Esti.mean=intervals[,1]+Len.CI/2
table2 <- data.frame(Len.CI,Esti.mean)
rownames(table2) <- c("percent","bca","norm")
knitr::kable(table2)
```