lab5

group12

11/29/2021

Question 1 Hypothesis testing

Using losse()

Using `loess()`function and get estimate $\hat{Y}$, and calculate the statistics

$$T = \frac{\hat{Y}(X_b) - \hat{Y}(X_a)}{X_b - X_a}, \quad whereX_b = argmax_x\hat{Y}(X), X_a = argmin_x\hat{Y}(X)$$

The T-statistic is below. Since it is significantly different from 0, We can not say the lottery is random.

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

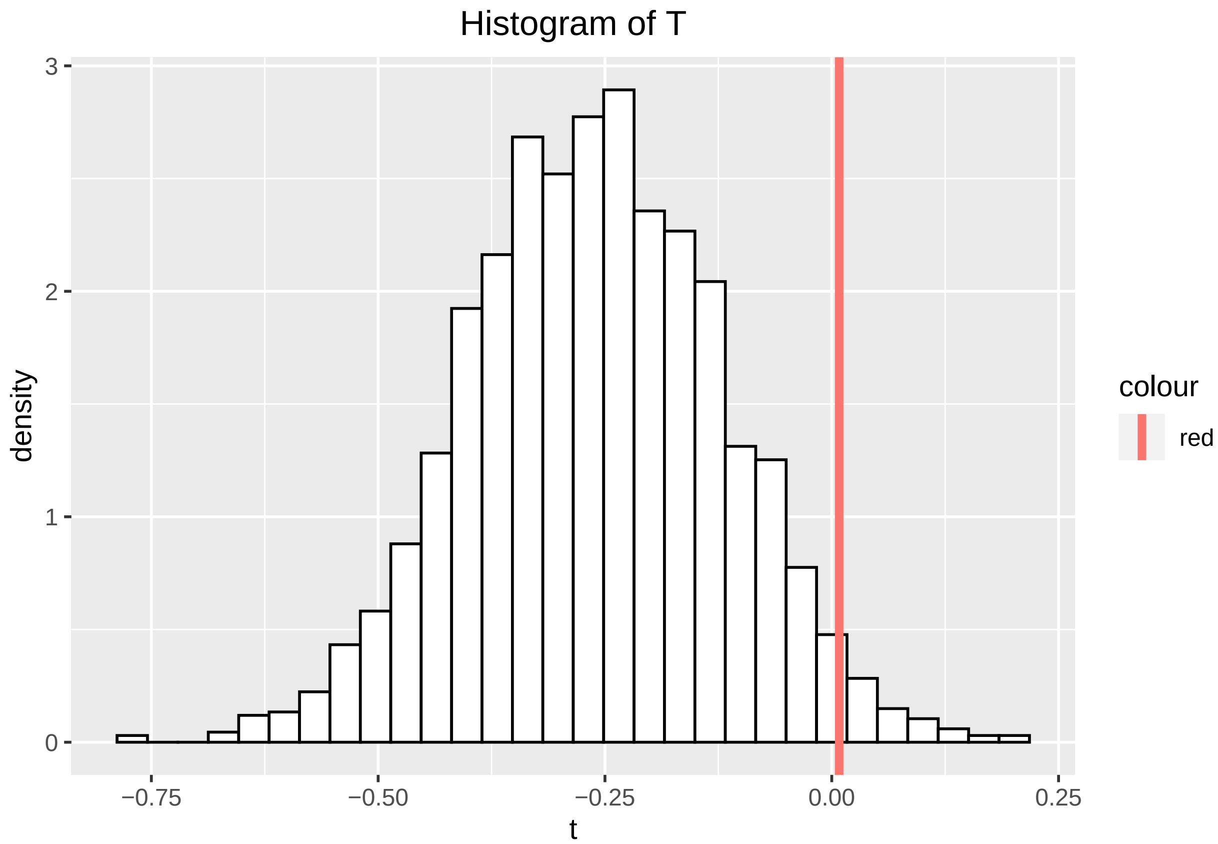
Distribution of T

Using the bootstrap algorithm(from course pdf files), we have the plot where the red line means 95% significance.

```
## [1] "summary for bootstrap"
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = df, statistic = stats, R = 2000)
##
##
## Bootstrap Statistics :
##       original      bias    std. error
## t1* -0.2671794 0.005376575   0.1400503
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

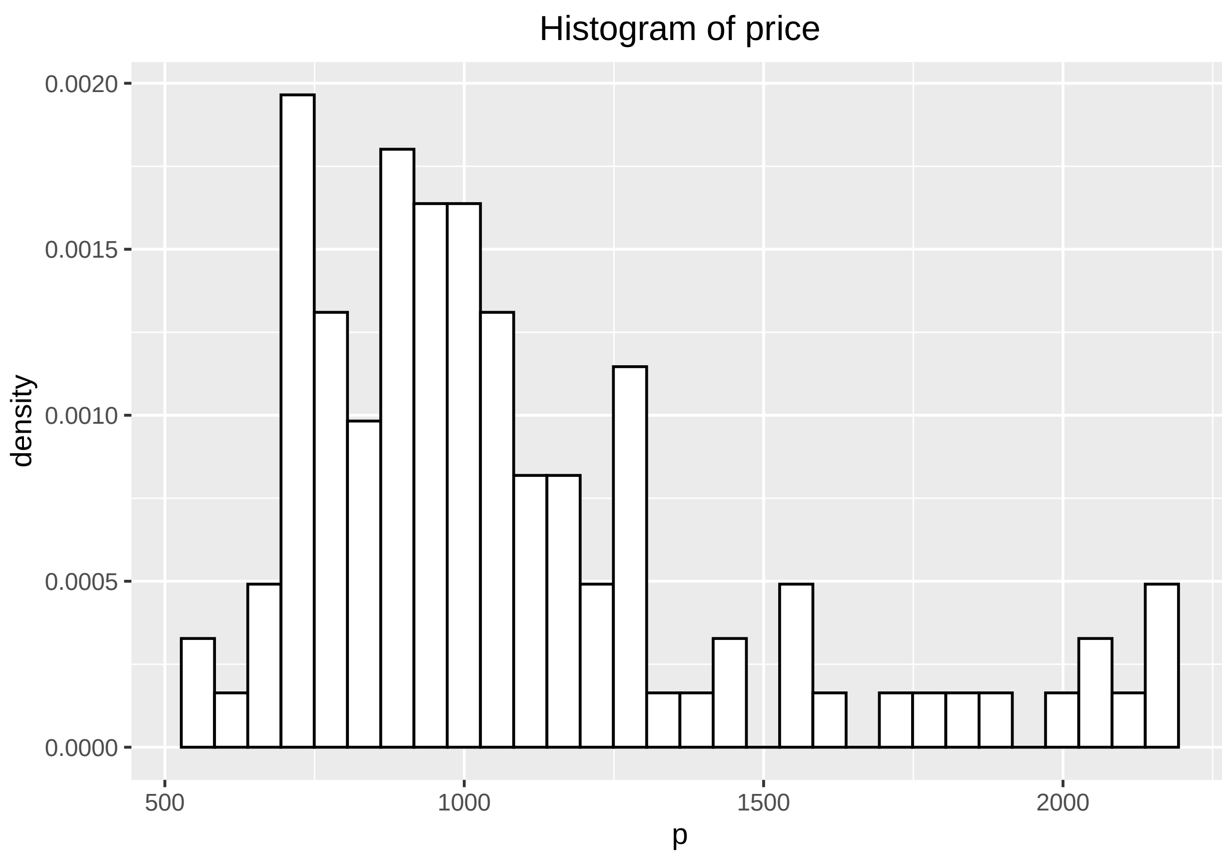


Since x = 0 is in the right side of red line, so we accept that T is not significantly different from 0, so the lottery is random.

Question 2 Bootstrap, jackknife and confifidence intervals

1 Plot and show mean value

This plot can not remind us any familiar distribution.




```
## [1] "mean value of peice is 1080.47272727273"
```

2 Compute some statistics

From the course document, we know that

$$bias - correction = T_1 = 2T(D) - \frac{\sum_{i=1}^{B} T_i^*}{B} variance\ of\ estimator = Var[\hat{T}(.)] = \frac{\sum_{i=1}^{B}(T(D^*) - \overline{T(D^*)})^2}{B-1}$$

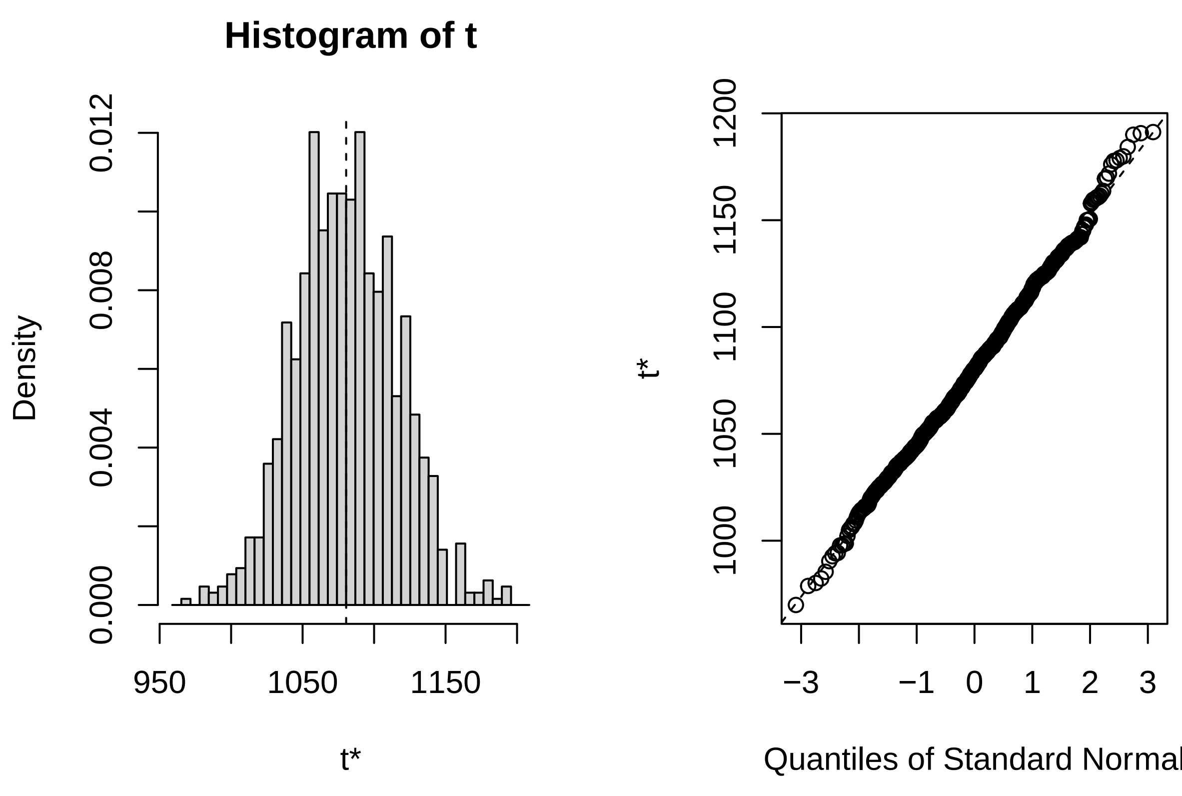
```
## [1] "summary of bootstrap"

##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data$Price, statistic = stat1, R = B)
##
##
## Bootstrap Statistics :
##     original    bias    std. error
## t1* 1080.473 0.3807182    35.67683

## [1] "                                                             "
```

```
## [1] "bias-correction : 1080.09200909091"
```

```
## [1] "variance : 1272.83631634602"
```

```
## [1] "                                                                    "
```

```
## 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%   (1010, 1150 )   (1014, 1150 )   (1016, 1160 )
## Calculations and Intervals on Original Scale
```



3 About estimate mean

```
## [1] "estimated mean is :1080.85344545455"
```

The estimated mean is 1080.853, it located in all confidence intervals.

4 jackknife

First, we have the knowledge from the course documents.

$$V\hat{ar[T(.)]} = \frac{1}{n(n-1)}\sum_{i=1}^{n}((\mathrm{T_i^*}) - \mathrm{J(T)})^2, \text{where } \mathrm{T_i^* = nT(D) - (n-1)T(D_i^*)} \text{and } \mathrm{J(T)} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i=1}}^{\mathrm{n}}\mathrm{T_i^*}$$

The variance of mean price is showed below and the comparasion is in the table.

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

```
##   boostrap jackknife
## 1 1272.836  1320.911
```

Appendix

```
library(boot)
library(ggplot2)
data <- read.csv2('lottery.csv')
##################################################################################
# Q1
##################################################################################
df <- data.frame(x = data[,4],y = data[,5])
los <- loess(y~x,data = df)
y_hat <- los[['fitted']]
Xb = df$x[which.max(df$y)]
Xa = df$x[which.min(df$y)]
T_ <- (predict(los,Xb)-predict(los,Xa))/(Xb-Xa)
## print(T_)
stats <- function(data,vec){
 datatemp<-data[vec,]
  los = loess(y ~ x, data = datatemp)
  Xb = data$x[which.max(data$y)]
  Xa = data$x[which.min(data$y)]
  y_Xb = predict(los,newdata=Xb)
  y_Xa = predict(los,newdata=Xa)
  T_stat = (y_Xb - y_Xa) / (Xb-Xa)
  return(T_stat)
}
# non-parametric bootstrap
set.seed(12345)
# dt=data1[order(data1$Draft_No),]
myboot = boot(data = df,
              statistic = stats,
              R = 2000)
## print('summary for bootstrap')
## myboot
# plot distribution
# plot(myboot, index = 1)
df <- data.frame(t=myboot$t)
```

```r
per95 = sort(myboot$t)[1950]
p1 <- ggplot(data = df, aes(x = t)) +
  ggtitle("Histogram of T") +
  geom_histogram(aes(y=..density..),
                 colour="black",
                 fill="white",
                 bins=30) +
  geom_vline(aes(xintercept = per95, color = "red"),size=1.5)+
  theme(plot.title = ggplot2::element_text(hjust=0.5))
#################################################################################
# Q2 Jackknife
#################################################################################
rm(list=ls())
data <- read.csv2('prices1.csv')
df <- data.frame(p = data[,1])
p2 <- ggplot(data = df, aes(x = p)) +
  ggtitle("Histogram of price") +
  geom_histogram(aes(y=..density..),
                 colour="black",
                 fill="white",
                 bins=30) +
  theme(plot.title = ggplot2::element_text(hjust=0.5))
## print(paste0('mean value of peice is ', mean(df[,1])))
##########################################################################
#q2
##########################################################################
stat1 <- function(vec,vn){
  return(mean(vec[vn]))
}
B=1000
set.seed(12345)
res1 = boot(data$Price, stat1, R=B)
## print('summary of bootstrap')
## res1
## print('                                                    ')
## print(paste0('bias-correction : ',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 )
## print(paste0('variance : ',var_boot))
## print('                                                    ')
# default is a 95% confidence interval
ci <- boot.ci(res1, type = c("perc", "bca", "norm"))
## print(ci)
## plot(res1)
##########################################################################
#q3
##########################################################################
## print(paste0('estimated mean is :',mean(res1$t)))
##########################################################################
#q4
##########################################################################
n = nrow(data)
constant = 1/(n*(n-1))
```

```r
T_i = sapply(1:n, function(i){
  n * mean(data$Price) - (n-1) * mean(data[-i,1])
})
J_T = (1/n) * sum(T_i)
Var_jac = constant * sum((T_i - J_T)^2)
## Var_jac
table = data.frame(boostrap = var_boot, jackknife= Var_jac)
## print(table)
```