Lab5,Group15

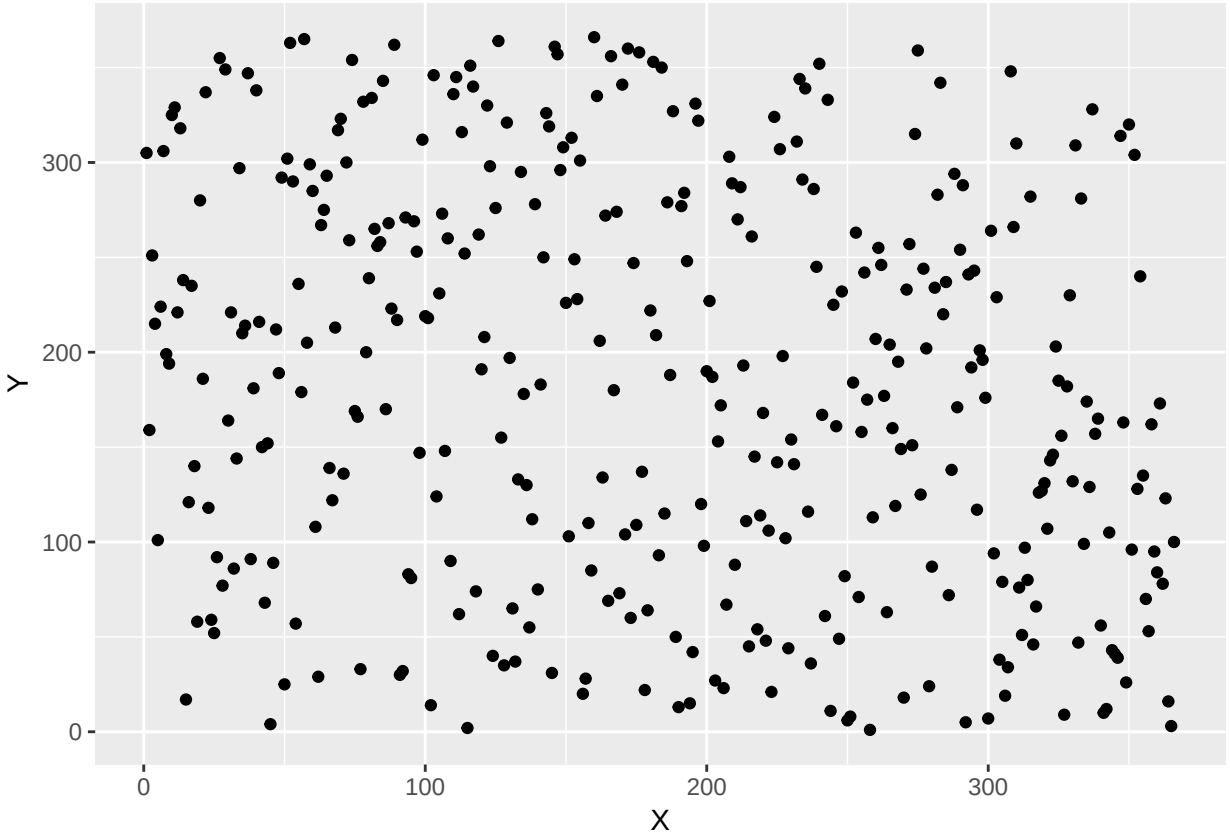
Naveen Gabriel ,Sridhar Adhikarla

2019-03-13

Contents

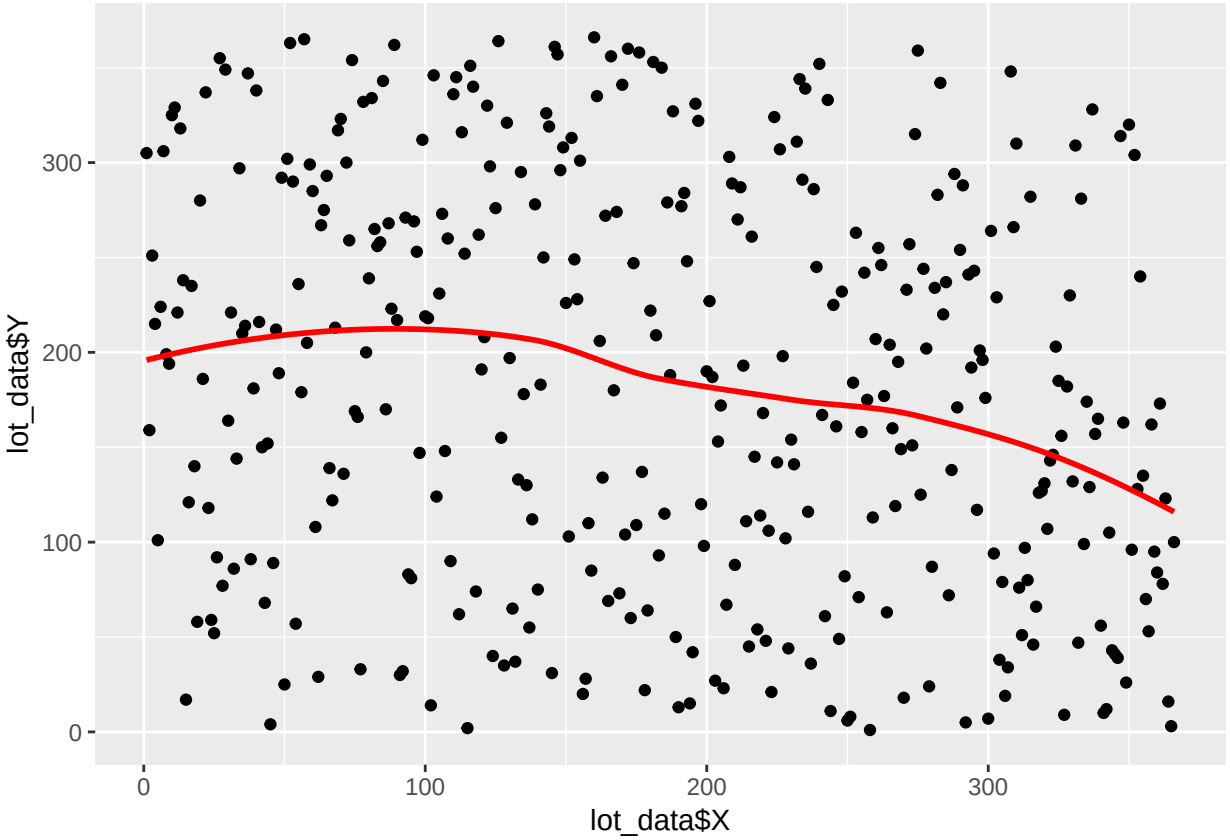
1 Hypothesis testing

1.1 Scatterplot of Y versus X



The data looks randomly distributed from this plot.

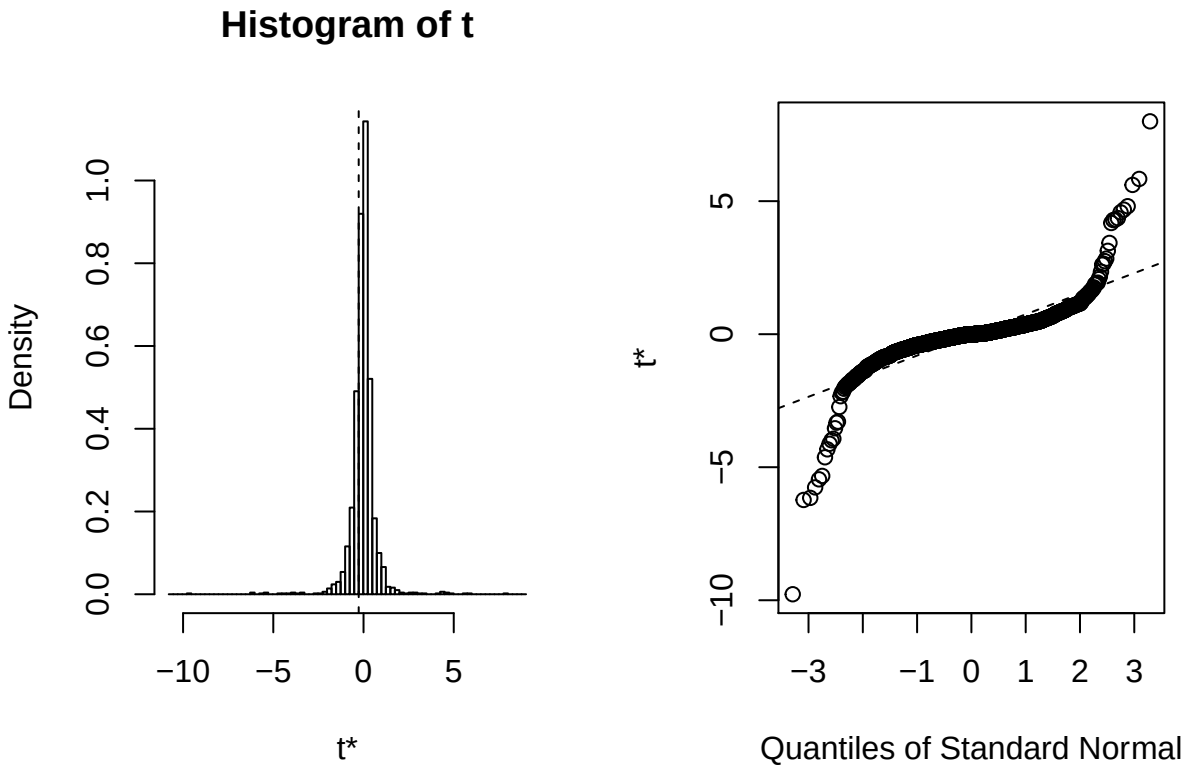
1.2 Estimate $\hat{Y}$

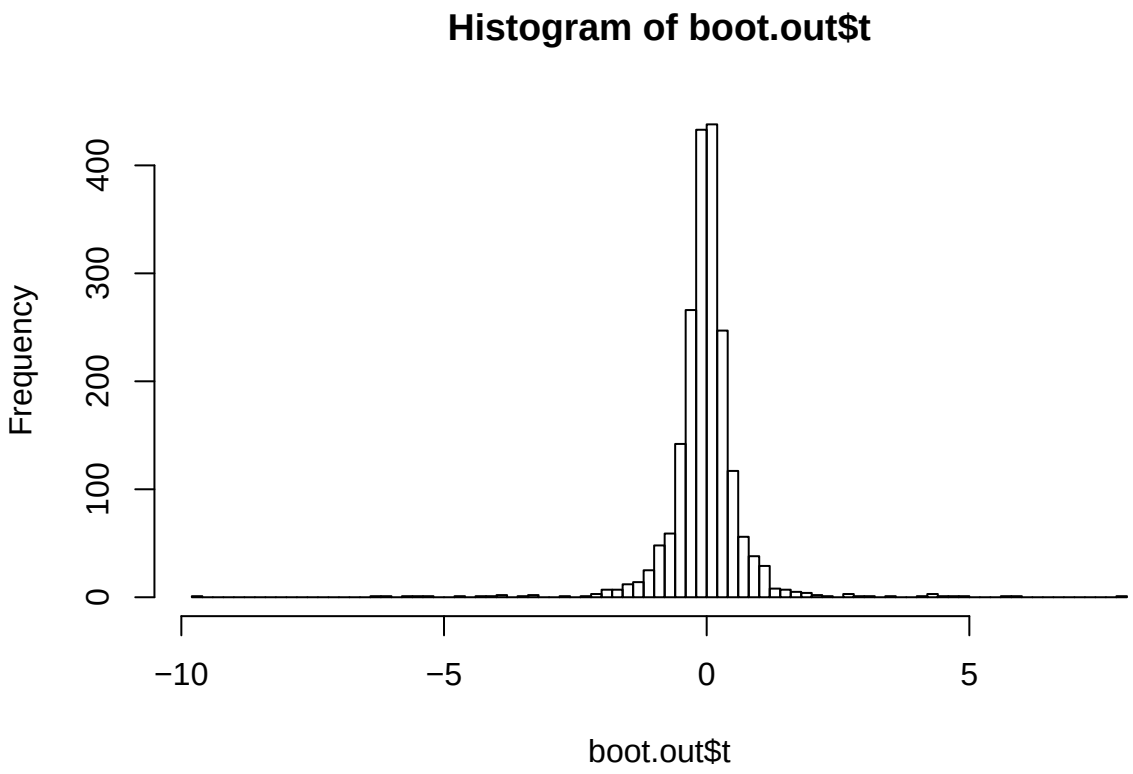


The smoothed line is not able to fit the data, this again indicates that the data is randomly distributed.

1.3 Test statistics to check if the lottery is random

```
The value of T-Test for the original dataset is : -0.2671794
```


Histogram of t
Density
0.0
0.2
0.4
0.6
0.8
1.0
−10
−5
0
5
t*
t*
5
0
−5
−10
−3
−1
0
1
2
3
Quantiles of Standard Normal




```
T is normally distributed with mean value,  -0.02835058  and standard deviation,  0.7746504
```

T looks normally distributed to me, with mean centered close to zero. The QQ plot also indicates that T is normally distributed as most of the values are close to the line.

This again indicates that the data was randomly distributed, as the mean value of T sampled 2000 times is also not significantly greater than 0.

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

CALL :
boot.ci(boot.out = boot.out, type = "norm")

Intervals :
Level      Normal
95%   (-2.0243,  1.0123 )
Calculations and Intervals on Original Scale
```

Since the 95% confidence interval is not significantly greater than 0, we can still say that the data is randomly distributed.

```
P-value :  0.5145972
```

In this case: H_0 -> T statistic is not significantly greater than 0. (null Hypothesis) H_a -> T statistic is significantly greater than 0. (alternative Hypothesis)

I am calculating the P-value with respect to 0, as this is the point that determines if our null Hypothesis (data is randomly distributed) is accepted or rejected. If the T value is significantly greater than 0 then the

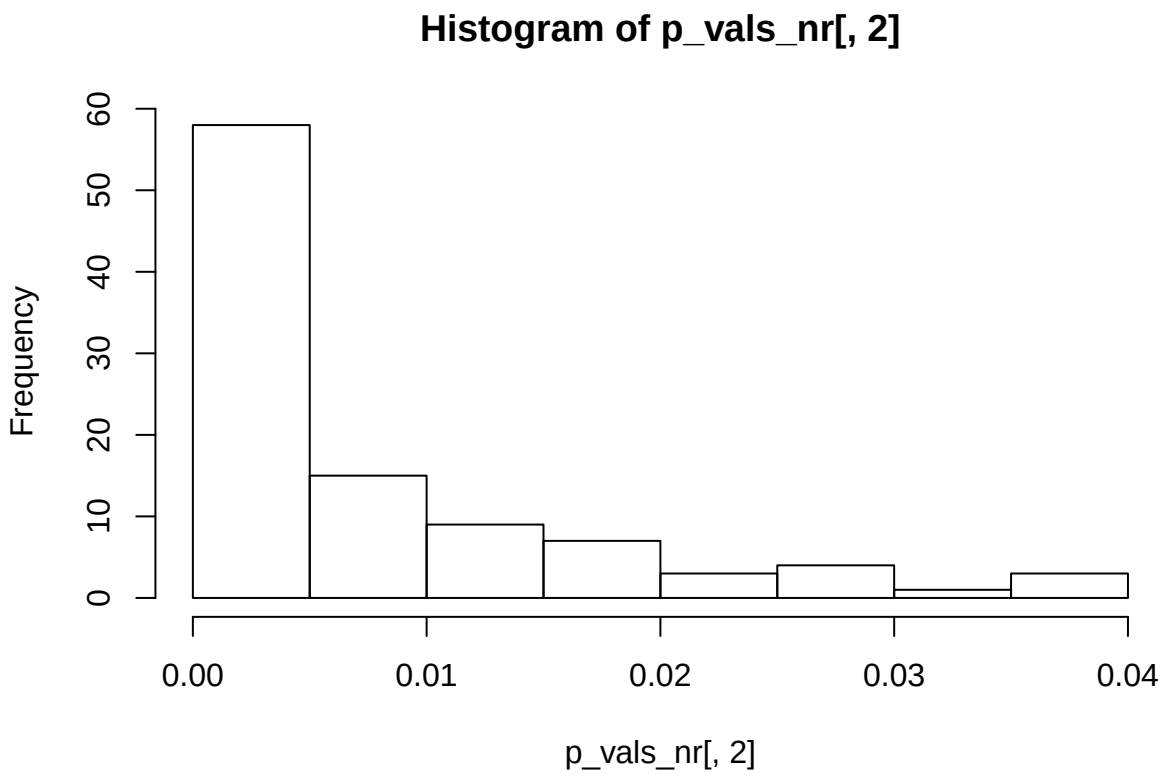
data is not random and null Hypothesis is rejected. So the probability that the T value of a random sample from the data will be smaller than zero is calculated here.

1.4 Implement a function depending on data and B that tests the hypothesis

```
P-value using Permutation Test :  0.453
```

Since P value calculated above is not significant (> than 0.05) hence we failed to reject the null hypothesis which implies that lottery is random.

1.5 Crude estimate of the power of the test constructed



This is the frequency plot showing the P-value's we got for different alpha values(0.1, 0.2, 0.3, ...10).

As we can see from the histogram, the maximum p-value we got from this non-random data is significant, so we can reject the null hypothesis for all the alpha values. This is what was expected and this proovs that the test is a good test to check if a random variable is from a random distribution or not.

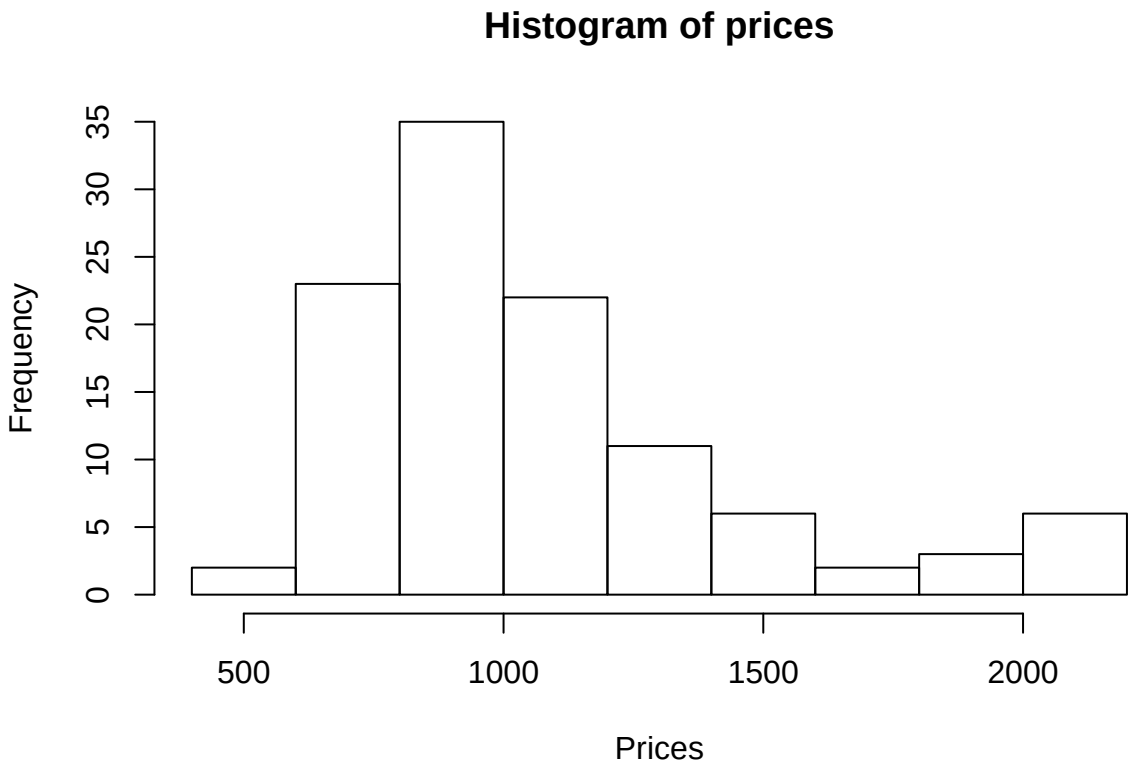
2 Bootstrap, jackknife and confidence intervals

2.1 Histogram of Price

```
Mean price is: 1080.473
```

From the distribution of histogram, it looks like gamma distribution

```
hist(prices1$Price, main = "Histogram of prices", xlab="Prices")
```

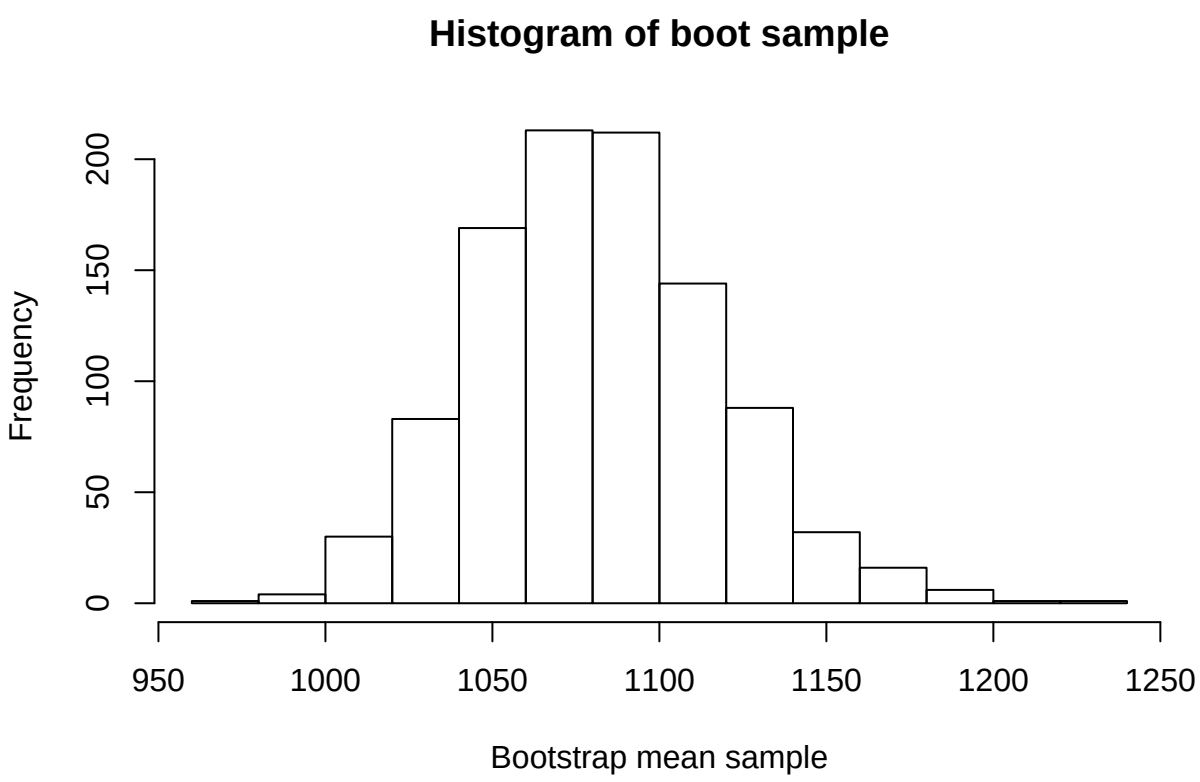


2.2 Bootstrap to estimate the parameter

The histogram plot of mean using non paramteric boot sample seems to be normal distribution according to central limit theorem. Bias corrected estimator is :

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

where T_i* is the mean of each bootstrap sample and T(D) is mean of actual data. After bias correction mean calculated by botostrap is nearly same to our true mean of the prices. With increase in bootstrap, the estimator, after bias correction, will come more closer to the actual value.




```
Bootstrap bias correction: 1079.511


Variance by boot strap sample: 1320.991


95% confidence interval for the mean price using bootstrap percentile:

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = bootsample, conf = 0.95, type = "perc")

Intervals :
Level     Percentile
95%   (1014, 1160 )
Calculations and Intervals on Original Scale


95% confidence interval for the mean price using bootstrap Bca:

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates
```

```
CALL :
boot.ci(boot.out = bootsample, conf = 0.95, type = "bca")

Intervals :
Level       BCa
95%   (1018, 1165 )
Calculations and Intervals on Original Scale
```

95% confidence interval for the mean price using first order normal approximation:

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = bootsample, conf = 0.95, type = "norm")

Intervals :
Level      Normal
95%   (1008, 1151 )
Calculations and Intervals on Original Scale
```

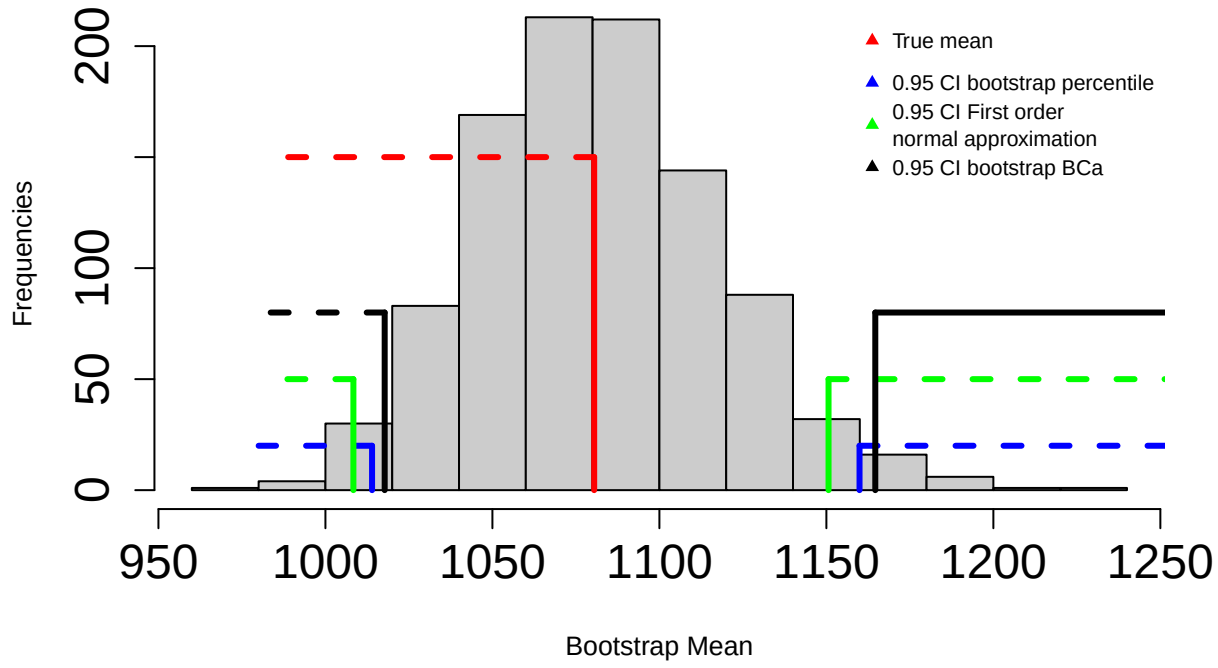
2.3 Variance of the mean price using the jackknife and compare with bootstrap

The variance of mean price by bootstrap is slightly higher than that estimated vy jacknife.

```
Variance using Jackknife: 1320.911
```

2.4 Compare the confidence intervals obtained

From the length and figure, it seems that the actual mean price is contained in all various CI but the length of CI under 1st normal approximation seems to be least which might suggest we can use it to say about location of mean with more confidence



Below table compares the length of three 95% confidence interval type .

Table 1: Comparison of various confidence interval

	Length
Percentile	145.9599
BCa	146.9064
1st order normal	142.2717

3 Appendix

```
knitr::opts_chunk$set(
    echo = TRUE,
    eval=TRUE,
    message = FALSE,
    warning = FALSE,
    comment = NA
)

#libraries
library(ggplot2) #To do Gelman-Rubin test
library(boot)
library(readxl)
set.seed(12345)
lottery_data = read_xls("lottery.xls")
lot_data = data.frame("X"=lottery_data$Day_of_year, "Y"=lottery_data$Draft_No)
#1
ggplot(lot_data) + geom_point(aes(X, Y))
#2
loessMod <- loess(Y ~ X, data=lot_data)
smoothed <- predict(loessMod)

ggplot() + geom_point(aes(lot_data$X, lot_data$Y)) +
  geom_line(aes(lot_data$X, smoothed), col="red", size=1)
#3
BootstrapT_Test = function(data_boot=lot_data, id){
  data_boot$boot_X = data_boot$X[id]
  data_boot$boot_Y = data_boot$Y[id]
  loessMod_boot <- loess(boot_Y ~ boot_X, data=data_boot)
  smoothed_boot <- predict(loessMod_boot)

  Xb = data_boot$boot_X[which.max(data_boot$boot_Y)]
  Xa = data_boot$boot_X[which.min(data_boot$boot_Y)]
  Ycap_Xb = smoothed_boot[Xb]
  Ycap_Xa = smoothed_boot[Xa]
  t_val = (Ycap_Xb - Ycap_Xa)/(Xb-Xa)
  return(t_val)
}
N = 2000
boot.out = boot(lot_data, BootstrapT_Test, N)

cat("The value of T-Test for the original dataset is :", boot.out$t0)
plot(boot.out)
hist(boot.out$t, breaks = 100)
cat("T is normally distributed with mean value, ", mean(boot.out$t), " and standard deviation, ", sd(boo
print(boot.ci(boot.out, type = "norm"))
cat("P-value : ", pnorm(0, mean = mean(boot.out$t), sd = sd(boot.out$t)))
#permutation test
permTest = function(boot_var){
  n = length(boot_var$t)
  new_diff = abs(boot_var$t) - abs(boot_var$t0)
  sum(new_diff>0)/n
```

```
}
N = 2000
boot.out_perm = boot(lot_data, BootstrapT_Test, N)
cat("P-value using Permutation Test : ", permTest(boot.out_perm))

#5
non_rand_data = lot_data
alphas_nr = seq(0.1, 10, 0.1)
beta_non_rand = rnorm(1, mean = 183, sd = 10)
p_vals_nr = matrix(0, nrow = length(alphas_nr), ncol = 2)
p_vals_nr[,1] = alphas_nr
for(j in 1:length(alphas_nr)){
  alpha_non_rand = alphas_nr[j]
  for(i in 1:366){
    non_rand_data$Y[i] = max(0, min(alpha_non_rand*i + beta_non_rand, 366))
  }
  N = 200
  boot.out_nr = boot(non_rand_data, BootstrapT_Test, N)
  p_vals_nr[j, 2] = permTest(boot.out_nr)
}
hist(p_vals_nr[,2])
knitr::opts_chunk$set(
    echo = TRUE,
    eval=TRUE,
    message = FALSE,
    warning = FALSE,
    comment = NA
)

library(boot)
prices1 <- read.csv2("prices1.csv")
n <- nrow(prices1)
samp_mean <- sum(prices1$Price)/n
cat("\n\nMean price is:",samp_mean)
hist(prices1$Price, main = "Histogram of prices", xlab="Prices")
calc_mean <- function(data,i) {
    data_new <- data[i,]$Price
    mean <- sum(data_new)/length(data_new)
    return(mean)
}

bootsample <- boot(prices1,statistic=calc_mean,R=1000)
hist(bootsample$t,main="Histogram of boot sample",xlab="Bootstrap mean sample",ylab="Frequency")

#Bias corrected estimator
mean_biascorrect <- 2*samp_mean - sum(bootsample$t)/1000

variance  <- sum((bootsample$t-mean_biascorrect)^2)/999
#Determine the bootstrap bias{correction and the variance of the mean price?


cat("\nBootstrap bias correction:", mean_biascorrect)
cat("\n\nVariance by boot strap sample:", variance)
```

```
cat("\n\n95% confidence interval for the mean price using bootstrap percentile: \n\n")
ci_perc <- boot.ci(bootsample,conf= 0.95,type="perc")
ci_perc

cat("\n\n95% confidence interval for the mean price using bootstrap Bca: \n\n")
ci_bca <- boot.ci(bootsample,conf= 0.95,type="bca")
ci_bca

cat("\n\n95% confidence interval for the mean price using first order normal approximation: \n\n")
ci_norm <- boot.ci(bootsample,conf= 0.95,type="norm")
ci_norm

x <- c()
for( i in 1:nrow(prices1)){
    T_i <- sum(prices1[-i,]$Price)/(n-1)
    x[i] <- (n*samp_mean - (n-1)*T_i)
}

x_mean <- sum(x)/n

v <- sum((x-x_mean)^2)/(n*(n-1))

cat("Variance using Jackknife:", v)
compare_data <- matrix(0,ncol=1,nrow=0)
compare_data <- rbind(compare_data,ci_perc$percent[5]-ci_perc$percent[4])
compare_data <- rbind(compare_data,ci_bca$bca[5]-ci_bca$bca[4])
compare_data <- rbind(compare_data,ci_norm$normal[3]-ci_norm$normal[2])
compare_data <- as.data.frame(compare_data)
colnames(compare_data) <- "Length"
rownames(compare_data) <- c("Percentile","BCa","1st order
normal")
{hist(bootsample$t,col=gray(0.8),main="",xlab="Bootstrap Mean",ylab="Frequencies",cex.axis=1.5,cex.lab=
segments(samp_mean,150,samp_mean,0,lwd=3, col="red");
segments(samp_mean,150,980,150,lwd=3,lty = 2,col="red");

segments(ci_perc$percent[4],20, ci_perc$percent[4],0,lwd=3, col="blue")
segments(ci_perc$percent[4],20, 980,20,lwd=3,lty = 2, col="blue")

segments(ci_perc$percent[5],20, ci_perc$percent[5],0,lwd=3, col="blue")
segments(ci_perc$percent[5],20,2000,20,lwd=3,lty = 2, col="blue")

segments(ci_norm$normal[2],50, ci_norm$normal[2],0,lwd=3, col="green")
segments(ci_norm$normal[2],50, 980,50,lwd=3,lty = 2,col="green")

segments(ci_norm$normal[3],50, ci_norm$normal[3],0,lwd=3, col="green")
segments(ci_norm$normal[3],50, 2000,50,lwd=3,lty = 2,col="green")

segments(ci_bca$bca[4],80, ci_bca$bca[4],0,lwd=3, col="black")
segments(ci_bca$bca[4],80, 980,80,lwd=3,lty = 2,col="black")

segments(ci_bca$bca[5],80, ci_bca$bca[5],0,lwd=3, col="black")
segments(ci_bca$bca[5],80, 2000,80,lwd=3, col="black")
```

```r
legend("topright",col=c("red","blue","green","black"),pch=17,legend=c("True mean","0.95 CI bootstrap per
normal approximation","0.95 CI bootstrap BCa"),bty="n",cex=0.7)

}
t <- knitr::kable(compare_data,format="latex",caption="Comparison of various confidence interval")
kableExtra::kable_styling(t,latex_options = "hold_position")
```