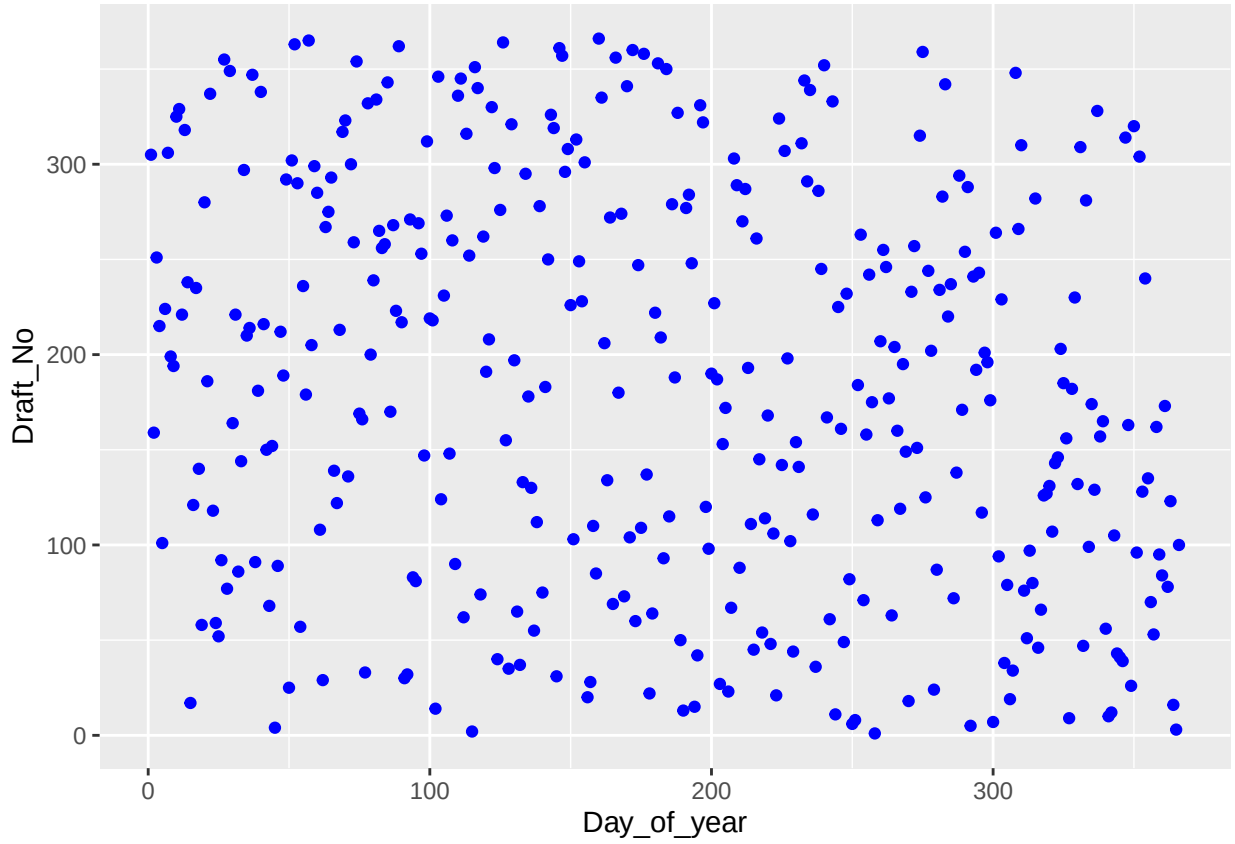
Computational Statistics Lab3 Group 29

Jin Yan (jinya425), Yaning Wang (yanwa579)

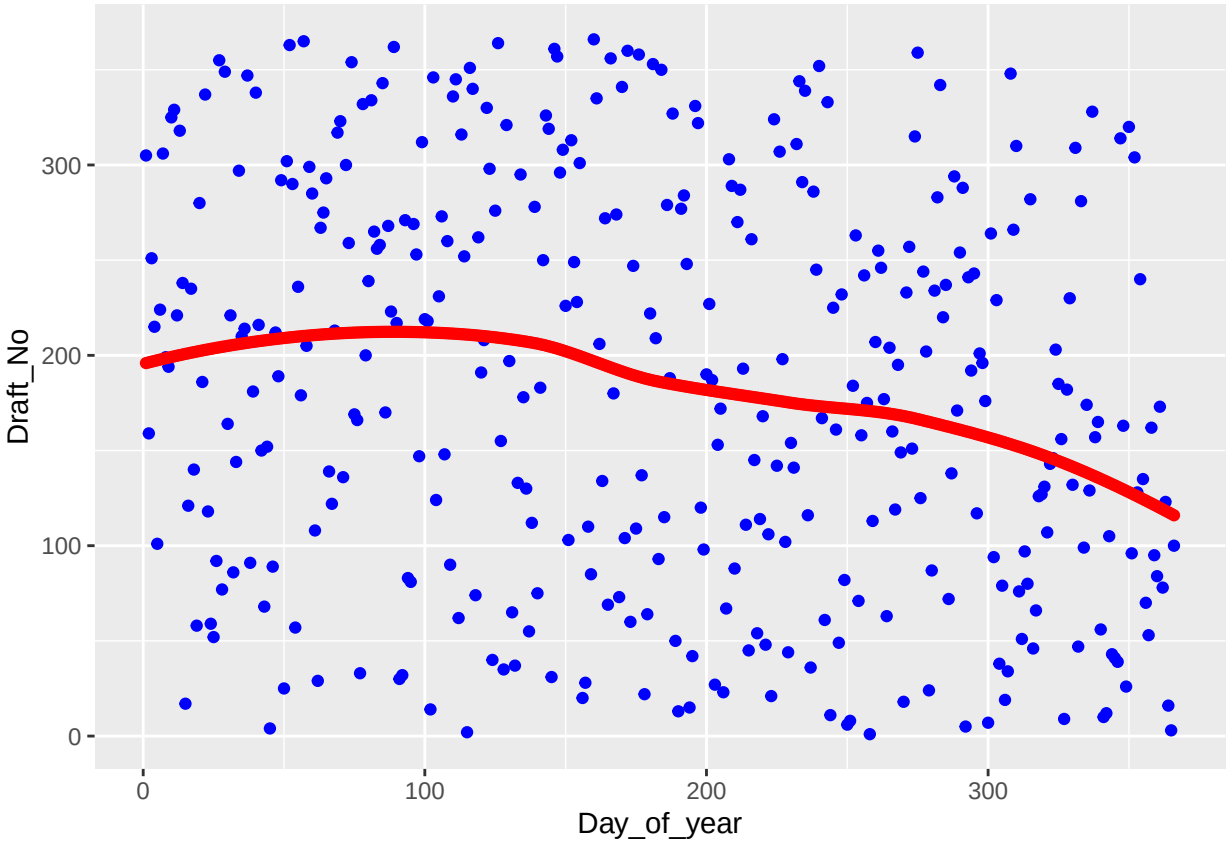
Question 1: Hypothesis testing

TASK ONE



Q: Conclude whether the lottery looks random. From the plot, we can see the x and y axes have no apparent relationship. The lottery looks random. But we need more analysis.

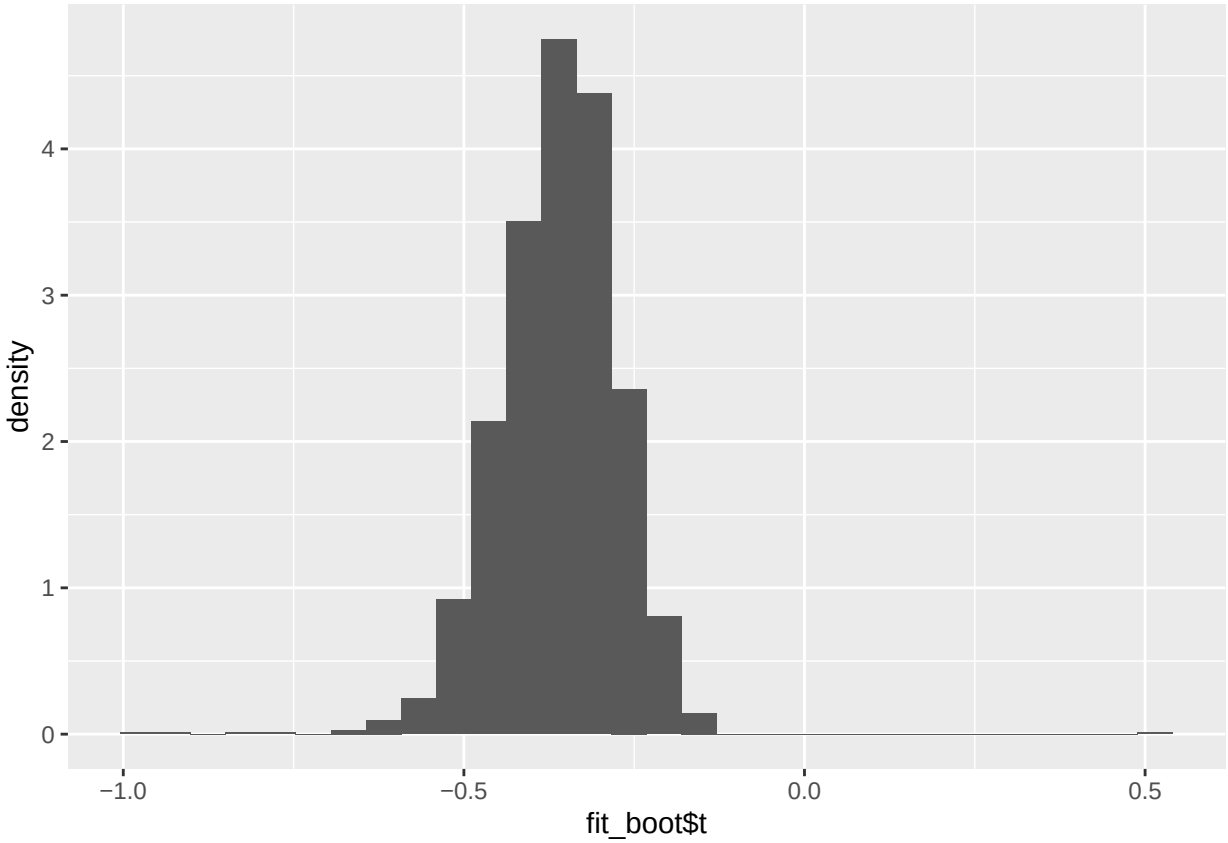
TASK TWO



From the plot, we can see a decreasing line. There is a negative correlation between Draft_No and Day_of_year. So, the lottery looks not random.

TASK THREE

```
## The t value is -0.3479163
```




```
## Value of p is 0.9995
```

TASK FOUR

```
## The p value is 0.142
```

TASK FIVE

```
## The p value is 0.815
```

```
## The power is  1
```

From the P value, we can conclude that power of our test is pretty good.

Question 2: Bootstrap, jackknife and confidence intervals

TASK ONE

It is like a Gamma distribution.

mean_price is 1080.473

TASK TWO

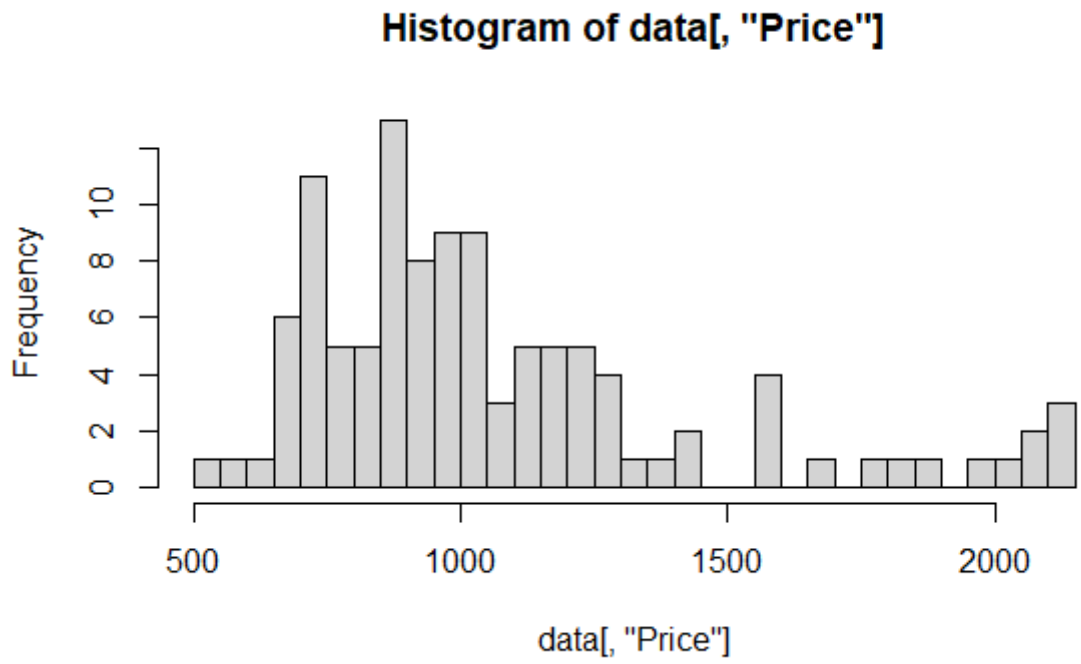
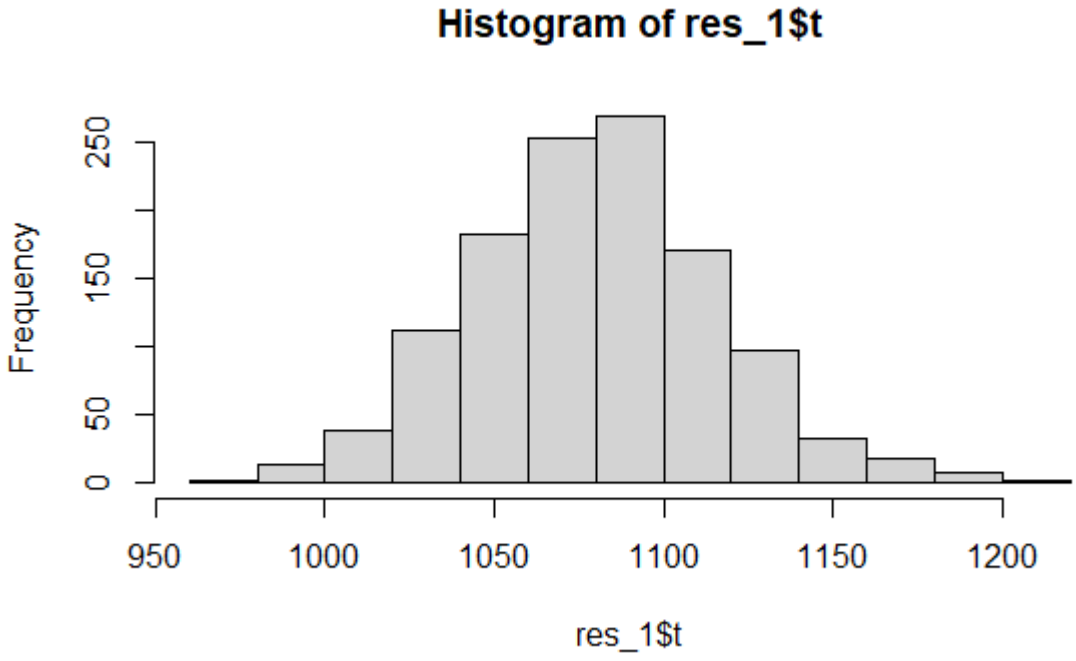


Figure 1: plot1



From the histogram, we can see the distribution is like the normal distribution.

The bias corrected estimator:1081.00375

The variance of estimator:1333.18489305084

95% intervals are shown below

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1200 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = res_1, type = c("perc", "bca", "norm"))
##
## Intervals :
## Level      Normal              Percentile            BCa
## 95%   (1009, 1153 )   (1012, 1158 )   (1016, 1163 )
## Calculations and Intervals on Original Scale
```

TASK THREE The variance of bootstrap method is around 1334, whereas the value of jacknife method is around 1320. Obviously, the mean second distribution more cluster.

TASK FOUR After the calculating, we found that the length for intervals are "144", "146", "147". They corresponds to the arguments of "type", "norm", "perc" , "bca" respectively. That means when we use the normal approximation to the distribution, we can get the shorter 95% interval.

In order to get the location of estimator in intervals, we can use the formula "(estimator - lower_bound_normal) /upper_bound_normal - lower_bound_normal" to the location of estimator in one interval. The followings are the results corresponding to "norm","perc","bca" respectively.

location_nor = 0.500026 location_perc = 0.4726284 location_bca = 0.4422024

From the above data, we can see that the bias estimator is nearest to the center of the interval that is obtained with the argument "norm".

APPENDIX

```
#Question 1: Hypothesis testing

#1
lottery<-read.csv2("lottery.csv")
lottery<-lottery%>%select(c(4,5))
ggplot(lottery,aes(x=Day_of_year,y=Draft_No))+
  geom_point(colour="blue")
#2
fit<-loess(Draft_No~Day_of_year,lottery)
pred_y<-predict(fit)
lottery<-lottery%>%mutate(pred_Draft_No=pred_y)
ggplot(lottery)+
  geom_point(aes(x=Day_of_year,y=Draft_No),colour="blue")+
  geom_point(aes(x=Day_of_year,y=pred_Draft_No),colour="red")
#3
fit<-loess(Draft_No~Day_of_year,lottery)
pred_y<-predict(fit)
lottery<-lottery%>%mutate(pred_Draft_No=pred_y)
x_a<-lottery[which.min(lottery$pred_Draft_No),1]
x_b<-lottery[which.max(lottery$pred_Draft_No),1]
y_b<-max(lottery$pred_Draft_No)
```

```r
y_a<-min(lottery$pred_Draft_No)
t0<-(y_b-y_a)/(x_b-x_a)
cat("The t value is",t0)
#This t value is not zero, and there is no standard what is significantly different from zero, so we ca
B<-2000
stat1<-function(data,index){
  data<-as.data.frame(data[index,])
  fit<-loess(Draft_No~Day_of_year,data)
  pred_y<-predict(fit)
  data<-data%>%mutate(pred_Draft_No=pred_y)
  x_a<-data[which.min(data$pred_Draft_No),1]
  x_b<-data[which.max(data$pred_Draft_No),1]
  y_b<-max(data$pred_Draft_No)
  y_a<-min(data$pred_Draft_No)
  t<-(y_b-y_a)/(x_b-x_a)
}
fit_boot<-boot(lottery,stat1,B)
df<-data.frame(fit_boot$t)
ggplot(data=df,aes(x=fit_boot$t))+
  geom_histogram(aes(y=..density..),bins=30)
t_0 <- fit_boot$t[fit_boot$t <= 0]
cat("Value of p is", length(t_0) / length(fit_boot$t), "\n")

#4
pvalue<-function(data,B){
  fit<-loess(Draft_No~Day_of_year,data)
  pred_y<-predict(fit)
  data<-data%>%mutate(pred_Draft_No=pred_y)
  x_a<-data[which.min(data$pred_Draft_No),1]
  x_b<-data[which.max(data$pred_Draft_No),1]
  y_b<-max(data$pred_Draft_No)
  y_a<-min(data$pred_Draft_No)
  t0<-(y_b-y_a)/(x_b-x_a)
  t<-c()
  #new
  for(i in 1:B){
    n<-dim(data)[1]
    data_x<-sample(1:n,size=n)
    data[,1]<-data_x
    fit<-loess(Draft_No~Day_of_year,data)
    pred_y<-predict(fit)
    data<-data%>%mutate(pred_Draft_No=pred_y)
    x_a<-data[which.min(data$pred_Draft_No),1]
    x_b<-data[which.max(data$pred_Draft_No),1]
    y_b<-max(data$pred_Draft_No)
    y_a<-min(data$pred_Draft_No)
    t[i]<-(y_b-y_a)/(x_b-x_a)
  }
  p_value <- length(t[abs(t) >= abs(t0)]) / B

  return(p_value)
}
```

```r
cat("The p value is",pvalue(lottery,2000),"\n")

#5

##1
#Generate dataset
set.seed(12345)
B<-200
x<-c(1:366)
n<-366
y<-c()
for (i in 1:n){
  norm<-rnorm(1,183,10)
  y[i]<-max(0,min((0.01*x[i]+norm),366))
}
data<-data.frame(Day_of_year=x,Draft_No=y)

##2

p_val<-pvalue(data,B)
cat("The p value is",p_val)

#The typical value to reject null hypothesis is 0.05, so in this situation, we can not reject null hypo

##3
set.seed(12345)
alpha<-seq(0.01,1,by=0.01)
p_val<-c()
y_sum<-data.frame()
for(a in alpha){
  for (i in 1:n){
    norm<-rnorm(1,183,10)
    y[i]<-max(0,min((a*x[i]+norm),366))
    data<-data.frame(Day_of_year=x,Draft_No=y)
  }
  p_val[i]<-pvalue(data,B)
}
power <- length(p_val[p_val < 0.025]) / length(p_val)
cat("The power is ",power)


#Question 2: Bootstrap, jackknife and confidence intervals
data <- read.csv("prices1.csv",sep = ";")
hist(data[,"Price"],breaks = 50)
mean_price <- mean(data[,"Price"])
mean_price
# It is like a Gamma distribution.
##############################################################################

# This function is used to tell the boot function we need get what statistics
# from resampled sample
library(boot)
mean_function <- function(data, indices) {return(mean(data[indices]))}
```

```r
B = 1200
set.seed(12345)
res_1 <- boot(data[,"Price"],mean_function,R = B)
res_1



# the estimation of the distribution of the mean price of the house
hist(res_1$t)
# from the histogram, we can see the distribution is like the normal
#distribution.


# According to the formula, we can get the bias corrected estimator
T <- 2 * res_1$t0 - mean(res_1$t)
print(paste0("the bias corrected estimator:", T ,sep = " "))


# Similarly, use function to get the variance of estimator

temp <- c()
for(i in 1:B){
  temp_2 <- (res_1$t[i] - mean(res_1$t))**2
  temp <- sum(temp + temp_2)
}

variance <- temp / (B - 1)
print(paste0("the variance of estimator:", variance ,sep = " "))


#find a 95% confidence interval
confi_interval <- boot.ci(res_1,type = c("perc","bca","norm"))
confi_interval

##############################################################################
##############################################################################

# calculate the variance of mean price
n <- nrow(data)

Ti <- c()
for(i in 1:n){
  ti <- n * mean(data$Price) - (n - 1) * mean(data[-i,"Price"])
  Ti <- c(Ti,ti)
}


variance_jacknife <- 1 / n / (n - 1) * sum((Ti - (1 / n) * sum(Ti))^2)
print(paste0("the variance of estimator:", variance_jacknife ,sep = " "))

# The variance of bootstrap method is around 1334, whereas the value of jacknife
# method is around 1320. Obviously, the mean second distribution more cluster.
```

```r
##############################################################################
##############################################################################
##############################################################################

lower_bound_normal <- 1009
upper_bound_normal <- 1153

lower_bound_perc <- 1012
upper_bound_perc <- 1158

lower_bound_bca <- 1016
upper_bound_bca <- 1163

interval_nor <- upper_bound_normal - lower_bound_normal
interval_perc <- upper_bound_perc - lower_bound_perc
interval_bca <- upper_bound_bca - lower_bound_bca
print(interval_nor)
print(interval_perc)
print(interval_bca)

# After the calculating, we found that the length for intervals are
# "144", "146", "147". They corresponds to the arguments of "type", "norm",
# "perc", "bca" respectively. That means when we use the normal approximation to
# the distribution, we can get the shorter 95% interval.

location_nor <- (T - lower_bound_normal) / interval_nor
location_perc <- (T - lower_bound_perc) / interval_perc
location_bca <- (T - lower_bound_bca) / interval_bca

print(location_nor)
print(location_perc)
print(location_bca)

# In order to get the location of estimator in intervals, we can use the formula
# "(estimator - lower_bound_normal) /upper_bound_normal - lower_bound_normal" to
# the location of estimator in one interval. The followings are the results
# corresponding to "norm","perc","bca" respectively.
#
# location_nor = 0.500026
# location_perc = 0.4726284
# location_bca = 0.4422024
#
# From the above data, we can see that the bias estimator is nearest to the
# center of the interval that is obtained with the argument "norm".
```