lab05

Shipeng Liu,Dongwei Ni

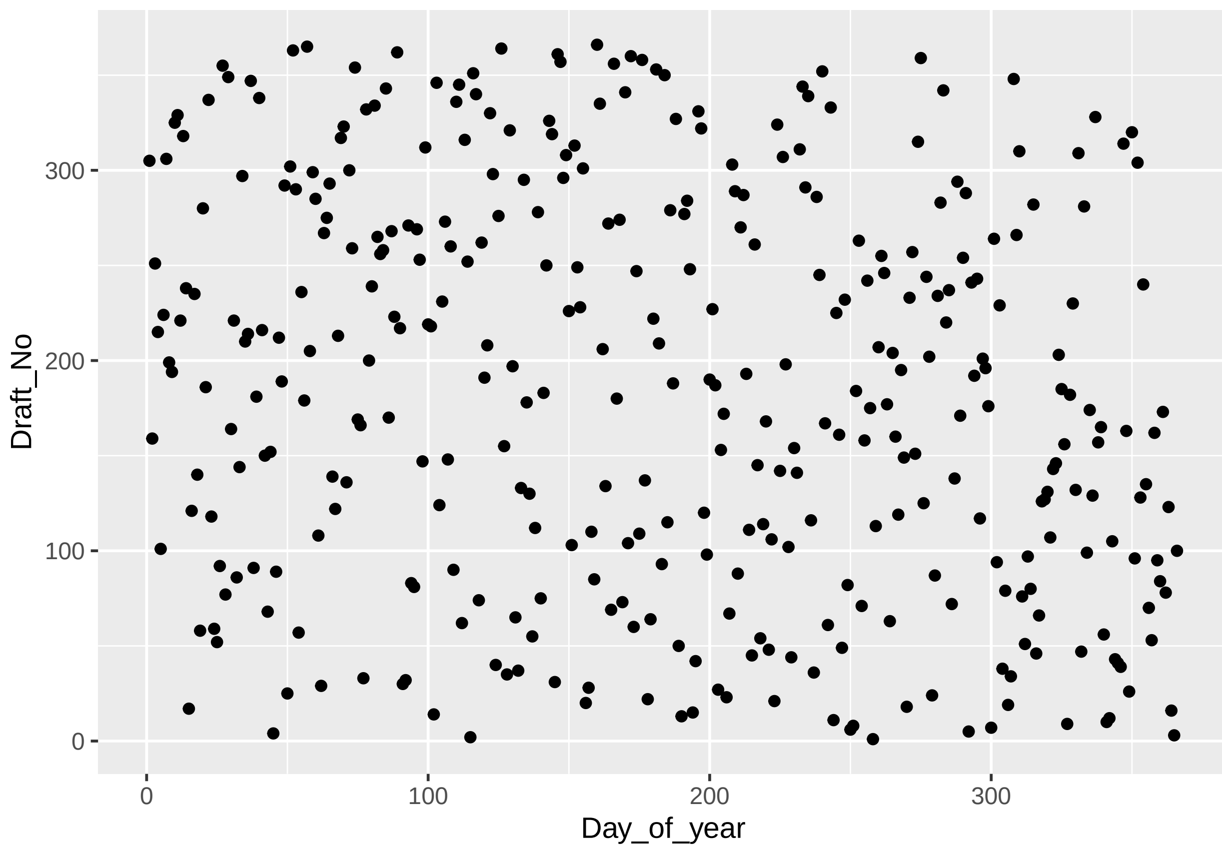
2023-01-22

Question 1: Hypothesis testing

```
set.seed(12345)
lottery=read.csv("lottery.csv",header =TRUE,sep =";")
```

Task 1:Make a scatter plot

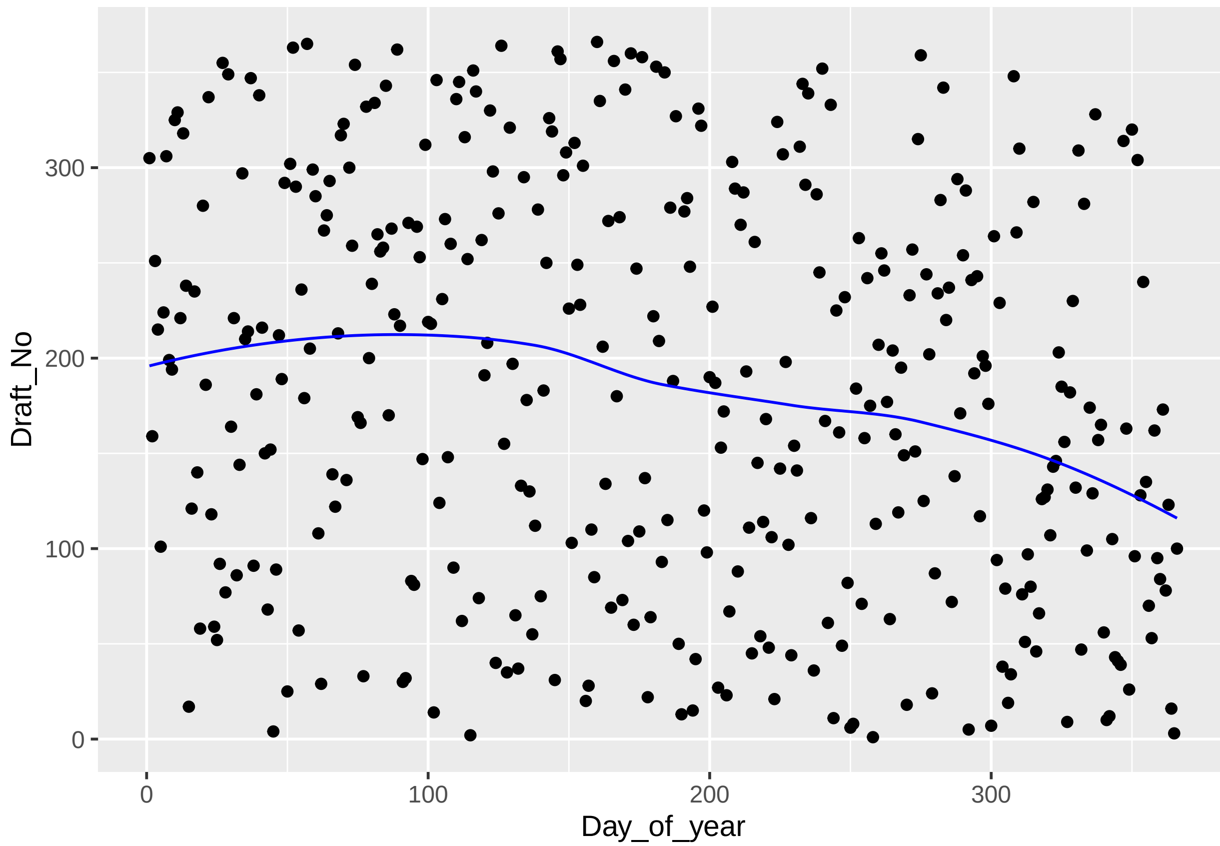
```
ggplot(data=lottery)+geom_point(aes(x=Day_of_year,y=Draft_No))
```



The lottery looks random,X and Y seems independent.

Task 2:Plot a loess smoother curve

```
fit=loess(lottery$Draft_No~lottery$Day_of_year,data=lottery)
Y_predict=predict(fit,data=lottery)
lottery1=lottery%>%mutate(predict_Y=Y_predict)
ggplot(data=lottery1)+geom_point(aes(x=Day_of_year,y=Draft_No))+
  geom_line(aes(x=Day_of_year,y=predict_Y),color="blue")
```



We use loess smoother to fit a curve and plot it in the previous graph,now it seems there are negative correlation between Day_of_year and Draft_No.

Task 3:Estimate the distribution of T using non–parametric bootstrap

```
stat=function(data,vn){
  data=as.data.frame(data[vn,])
  fit_bootstarp=loess(Draft_No~Day_of_year,data=data)
  y_predict_bs=predict(fit_bootstarp,data=data)

  y_predict_max=max(y_predict_bs)
  y_predict_min=min(y_predict_bs)

  x_max=data$Day_of_year[which.max(y_predict_bs)]
  x_min=data$Day_of_year[which.min(y_predict_bs)]

  t=((y_predict_max-y_predict_min)/(x_max-x_min))
  return(t)
```

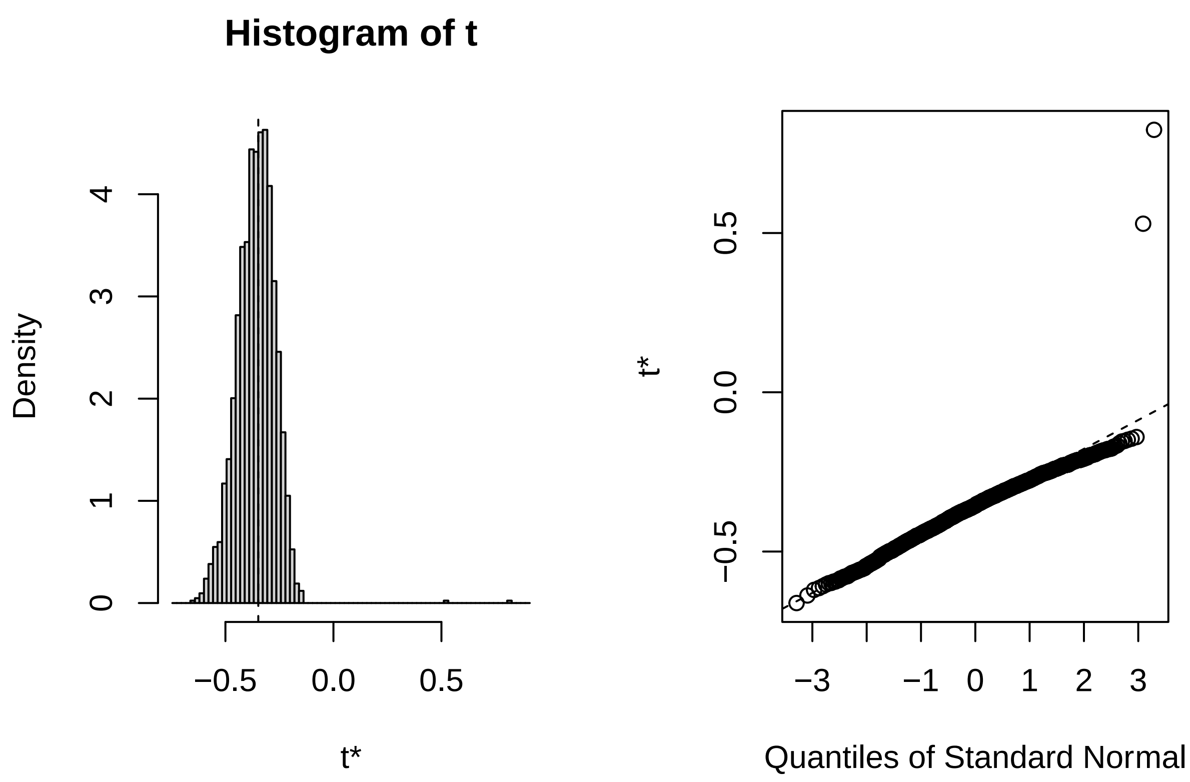
```r
}

res=boot(lottery,stat,R=2000)
boot.ci(res)
```

```
## Warning in boot.ci(res): bootstrap variances needed for studentized intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = res)
##
## Intervals :
## Level      Normal              Basic
## 95%   (-0.5146, -0.1600 )   (-0.4860, -0.1553 )
##
## Level     Percentile            BCa
## 95%   (-0.5405, -0.2099 )   (-0.5249, -0.1977 )
## Calculations and Intervals on Original Scale
```

```r
plot(res)
```



t=0 is outside of a significant portion of distribution of T,so the lottery is not random.

```r
p_value=mean(res$t<=0)
p_value
```

```
## [1] 0.999
```

Value of p is 0.999,where $P(t <= 0) = p$.

Task 4:implement permutation test

```
B=2000
permu_test=function(data,B){
  Ms=stat(data,1:nrow(data))
  sta=numeric(B)
  n=dim(data)[1]
  for(b in 1:B){
    GB=sample(data$Day_of_year,n)
    data=data%>%mutate(Day_of_year=GB)
    sta[b]=stat(data,1:n)
  }
  #Ms=stat(lottery,1:nrow(lottery))
  #calculate p value
  #test is two-sided
  p_value_permut=mean(abs(sta)>=abs(Ms))
  return(p_value_permut)
}

res_permu=permu_test(lottery,B)
res_permu
```

```
## [1] 0.154
```

The p-value is around 0.15,so We can't reject H0,that lottery is random.

Task 5:Study the power

```
repeat_power=function(alpha){
  new_dataset=as.data.frame(lottery$Day_of_year)
  colnames(new_dataset)=c("Day_of_year")
  new_dataset=new_dataset%>%
    mutate(Draft_No=NA)
  for(i in 1:nrow(new_dataset)){
    new_dataset$Draft_No[i]=max(0,min(alpha*new_dataset$Day_of_year[i]+
                                        rnorm(1,183,sd=10),366))
  }

  res_permu=permu_test(new_dataset,200)
  return(res_permu)
}

power_data=sapply(seq(0.01,1,0.01),FUN=repeat_power)
power=1-(sum(power_data[power_data>0.05])/length(power_data))
cat("The power is:",power)
```

```
## The power is: 0.9901
```

In this case,

H0:Lottery is random

H1:Lottery is non–random

When the alpha is 0.07,the p-value is 0.01,There is only 1% chance that the data distribution is random, so we reject the null hypothesis.

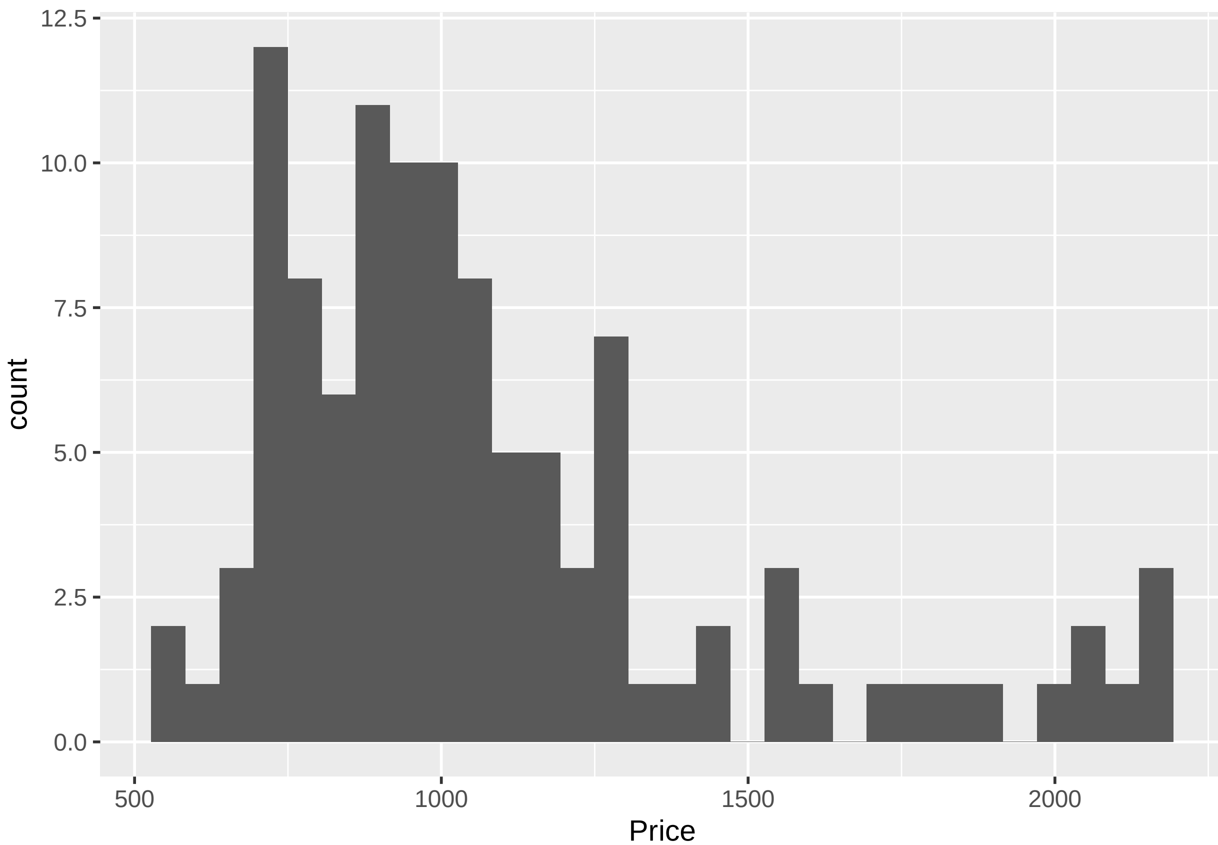
Consider when α become larger,The distribution of data will gradually become like non-random.Finally we got the power=0.98505,hence the quality of the test statistics is good.

Assignment 2:Bootstrap, jackknife and confidence intervals

```
price=read.csv("prices1.csv",header =TRUE,sep =";")
```

Task 1:Histogram and mean price

```
ggplot(data=price)+geom_histogram(aes(x=Price),bins = 30)
```




```
cat("The mean price is :",mean(price$Price))
```

```
## The mean price is : 1080.473
```

Does it remind any conventional distribution? The distribution looks likes gamma distribution.

Task 2

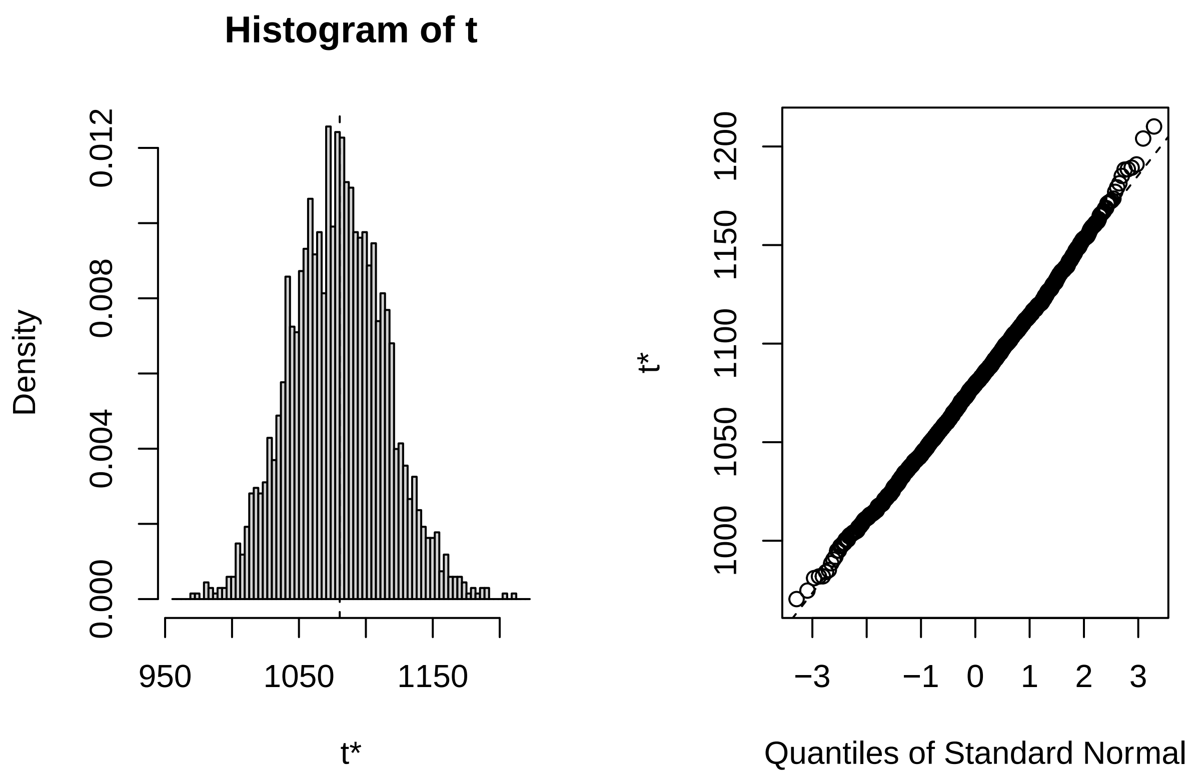
Estimate the distribution of the mean price of the house using bootstrap

```
stat_mean_price=function(data,vn){
  data=data[vn,]
  t=mean(data$Price)
  return(t)
}

res=boot(price,stat_mean_price,R=2000)
res
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = price, statistic = stat_mean_price, R = 2000)
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 1080.473 -0.9710045    35.31229
```

```
plot(res)
```



Bootstrap bias–correction and the variance of the mean price

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

```
cat("The variance of the mean price:",35.64^2,
    "\nThe bias correction:",2*mean(price$Price)-mean(res$t))
```

```
## The variance of the mean price: 1270.21
## The bias correction: 1081.444
```

Compute a 95% confidence interval

```
boot.ci(res)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = res)
##
## Intervals :
## Level      Normal              Basic
## 95%   (1012, 1151 )   (1009, 1148 )
##
## Level     Percentile            BCa
## 95%   (1013, 1152 )   (1015, 1157 )
## Calculations and Intervals on Original Scale
```

The 95% confidence interval for the mean price using: bootstrap percentile:(1013,1152)

bootstrap BCa:(1015,1157)

first–order normal approximation:(1012,1151)

Task 3:Estimate the variance of the mean price using the jackknife and compare it with the bootstrap estimate

Jackknife (n = B):

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

where $T_i^* = nT(D) - (n-1)T(D_i^*)$ $J(T) = \frac{1}{n}\sum_{i=1}^{n} T_i^*$

```
jackknife=function(data,B){
  res=c()
  for(i in 1:B){
    res=c(res,mean(data$Price[-i]))
  }
  return(res)
}

res_jackknife=jackknife(price,nrow(price))
n=nrow(price)
Ti_star=n*mean(price$Price)-(n-1)*res_jackknife
var_jackknife=sum(((Ti_star)-mean(Ti_star))^2)/(n*(n-1))
```

```r
cat("The variance of the mean price using the jackknife:",var_jackknife,
    "\nThe variance of the mean price using the bootstrap:",35.64^2)
```

```
## The variance of the mean price using the jackknife: 1320.911
## The variance of the mean price using the bootstrap: 1270.21
```

Compare to the bootstrap estimate,the variance of the mean price using the jackknife is larger,because Jackknife tend to overestimate variance.

Task 4:Compare the confidence intervals obtained with respect to their length and the location of the estimated mean in these intervals

```r
a=mean(price$Price)-1.96*sqrt(var_jackknife)
b=mean(price$Price)+1.96*sqrt(var_jackknife)
cat("The confidence intervals (95%) of means of price using jackknife is:(",a,",",b,")")
```

```
## The confidence intervals (95%) of means of price using jackknife is:( 1009.238 , 1151.708 )
```

```r
ic_res=data.frame("confidence_intervals"=c("(1009,1153)","(1009,1151)",
                                           "(1010,1152)","(1013,1158)"),
           "length"=c(1153-1009,1151-1009,1152-1010,1158-1013),
           "estimated_mean"=c(((1153+1009)/2),((1151+1009)/2),
                              ((1152+1010)/2),((1158+1013)/2)))

rownames(ic_res)=c("Normal","Basic","Percentile","BCa")
ic_res
```

```
##            confidence_intervals length estimated_mean
## Normal              (1009,1153)    144         1081.0
## Basic               (1009,1151)    142         1080.0
## Percentile          (1010,1152)    142         1081.0
## BCa                 (1013,1158)    145         1085.5
```

Appendix

```r
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(dplyr)
library(boot)
set.seed(12345)
lottery=read.csv("lottery.csv",header =TRUE,sep =";")
ggplot(data=lottery)+geom_point(aes(x=Day_of_year,y=Draft_No))
fit=loess(lottery$Draft_No~lottery$Day_of_year,data=lottery)
Y_predict=predict(fit,data=lottery)
lottery1=lottery%>%mutate(predict_Y=Y_predict)
ggplot(data=lottery1)+geom_point(aes(x=Day_of_year,y=Draft_No))+
  geom_line(aes(x=Day_of_year,y=predict_Y),color="blue")

stat=function(data,vn){
  data=as.data.frame(data[vn,])
  fit_bootstarp=loess(Draft_No~Day_of_year,data=data)
  y_predict_bs=predict(fit_bootstarp,data=data)

  y_predict_max=max(y_predict_bs)
```

```r
  y_predict_min=min(y_predict_bs)

  x_max=data$Day_of_year[which.max(y_predict_bs)]
  x_min=data$Day_of_year[which.min(y_predict_bs)]

  t=((y_predict_max-y_predict_min)/(x_max-x_min))
  return(t)
}

res=boot(lottery,stat,R=2000)
boot.ci(res)
plot(res)
p_value=mean(res$t<=0)
p_value
B=2000
permu_test=function(data,B){
  Ms=stat(data,1:nrow(data))
  sta=numeric(B)
  n=dim(data)[1]
  for(b in 1:B){
    GB=sample(data$Day_of_year,n)
    data=data%>%mutate(Day_of_year=GB)
    sta[b]=stat(data,1:n)
  }
  #Ms=stat(lottery,1:nrow(lottery))
  #calculate p value
  #test is two-sided
  p_value_permut=mean(abs(sta)>=abs(Ms))
  return(p_value_permut)
}

res_permu=permu_test(lottery,B)
res_permu

repeat_power=function(alpha){
  new_dataset=as.data.frame(lottery$Day_of_year)
  colnames(new_dataset)=c("Day_of_year")
  new_dataset=new_dataset%>%
    mutate(Draft_No=NA)
  for(i in 1:nrow(new_dataset)){
    new_dataset$Draft_No[i]=max(0,min(alpha*new_dataset$Day_of_year[i]+
                                        rnorm(1,183,sd=10),366))
  }

  res_permu=permu_test(new_dataset,200)
  return(res_permu)
}

power_data=sapply(seq(0.01,1,0.01),FUN=repeat_power)
power=1-(sum(power_data[power_data>0.05])/length(power_data))
cat("The power is:",power)
price=read.csv("prices1.csv",header =TRUE,sep =";")
ggplot(data=price)+geom_histogram(aes(x=Price),bins = 30)
```

```r
cat("The mean price is :",mean(price$Price))
stat_mean_price=function(data,vn){
  data=data[vn,]
  t=mean(data$Price)
  return(t)
}

res=boot(price,stat_mean_price,R=2000)
res
plot(res)
cat("The variance of the mean price:",35.64^2,
    "\nThe bias correction:",2*mean(price$Price)-mean(res$t))
boot.ci(res)
jackknife=function(data,B){
  res=c()
  for(i in 1:B){
    res=c(res,mean(data$Price[-i]))
  }
  return(res)
}

res_jackknife=jackknife(price,nrow(price))
n=nrow(price)
Ti_star=n*mean(price$Price)-(n-1)*res_jackknife
var_jackknife=sum(((Ti_star)-mean(Ti_star))^2)/(n*(n-1))
cat("The variance of the mean price using the jackknife:",var_jackknife,
    "\nThe variance of the mean price using the bootstrap:",35.64^2)
a=mean(price$Price)-1.96*sqrt(var_jackknife)
b=mean(price$Price)+1.96*sqrt(var_jackknife)
cat("The confidence intervals (95%) of means of price using jackknife is:(",a,",",b,")")
ic_res=data.frame("confidence_intervals"=c("(1009,1153)","(1009,1151)",
                                    "(1010,1152)","(1013,1158)"),
           "length"=c(1153-1009,1151-1009,1152-1010,1158-1013),
           "estimated_mean"=c(((1153+1009)/2),((1151+1009)/2),
                              ((1152+1010)/2),((1158+1013)/2)))

rownames(ic_res)=c("Normal","Basic","Percentile","BCa")
ic_res
```