Computational Statistics Lab 5

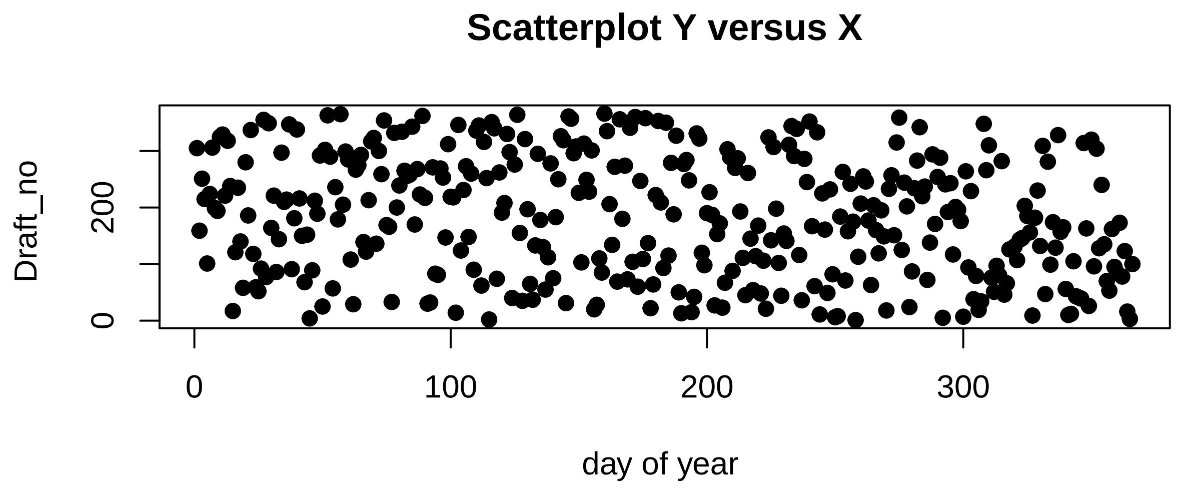
Martynas Lukosevicius, Alejo Perez Gomez, Zahra Jalil Pour

04/12/2020

Question 1: Hypothesis testing

1.

For this question we are required to make a scatter plot and conclude whether the lottery looks random.

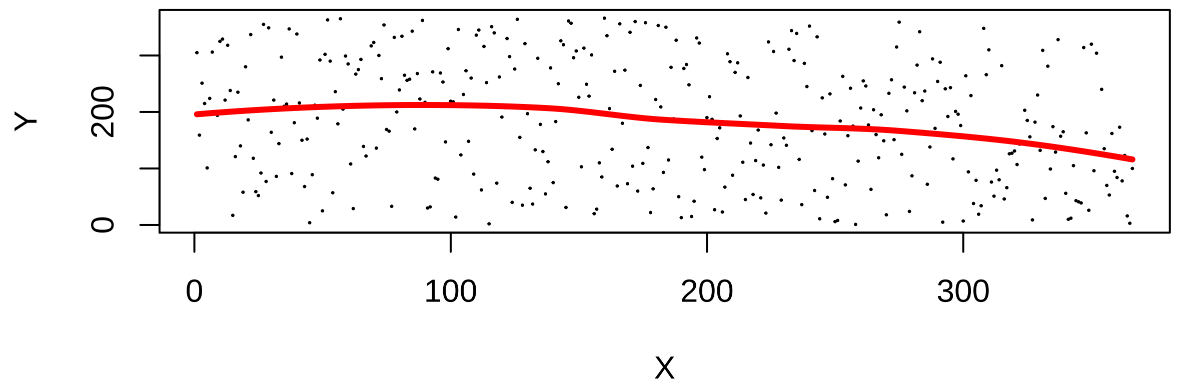


By judging based on our plot, lottery looks random.

2.

An estimate $\hat{Y}$ will be computed using function *loess()* and afterwards we will plot $\hat{Y}(X)$ adding it to the previous plot.

```r
lw1 <- loess(Y ~ X, data = new_df)
new_df$Y_hat <- predict(lw1, new_df$Day_of_year)
plot(Y ~ X, data=new_df,pch=19,cex=0.1)
lines(X,lw1$fitted,col="red",lwd=3)
```



From estimates we can see that there is some pattern in the data, however it is hardly visible.

3.

For this section we will use the statistic given, from where we intuit it stands for a slope in $\hat{Y}(X)$

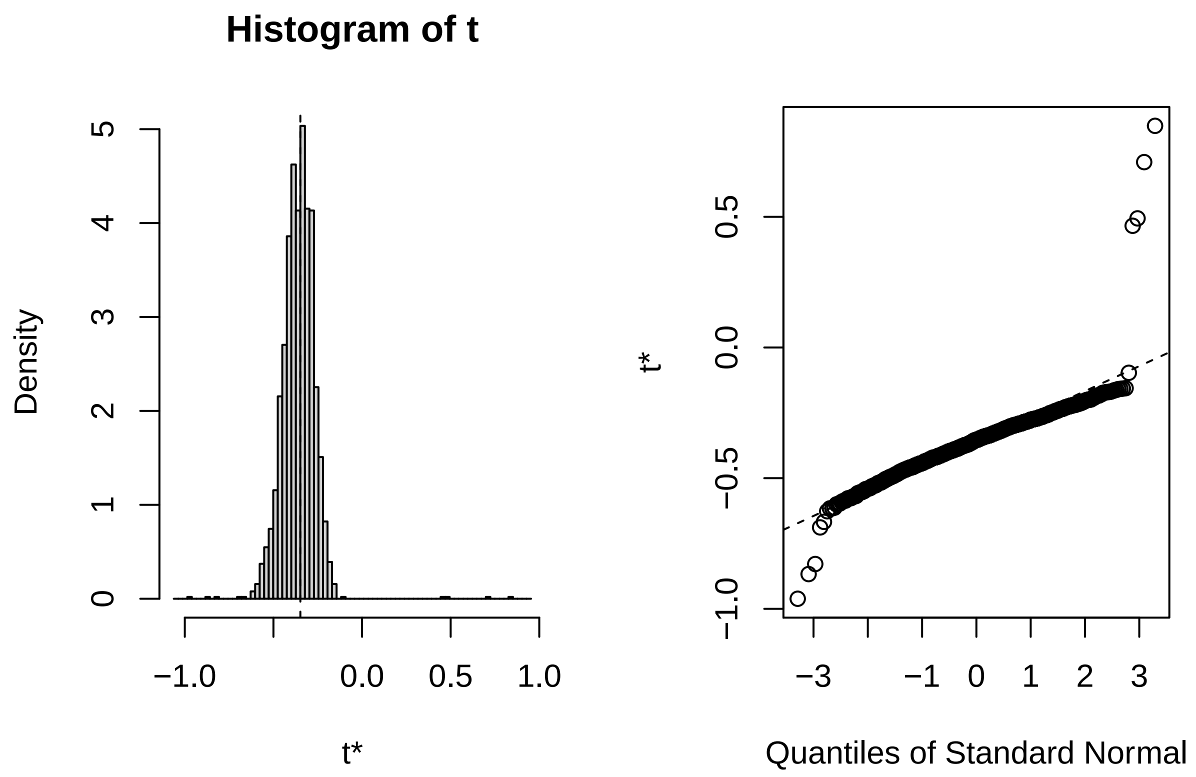
```
statis <- function(data, model){
  index_max <- which.max(model$fitted)
  index_min <- which.min(model$fitted)

  X_a <- data$Day_of_year[index_min]
  Y_hat_a <- model$fitted[index_min]

  X_b <- data$Day_of_year[index_max]
  Y_hat_b <- model$fitted[index_max]
  statistic <- (Y_hat_b - Y_hat_a)/(X_b - X_a)
  return(statistic)
}

library(boot)
fc<- function(data, index){
  data <- as.data.frame(data[index, ])
  loes <- loess(Draft_No ~ Day_of_year, data)
  return(statis(data,loes))
}


# bootstrap
results <- boot(new_df, statistic = fc, R=2000)
plot(results)
```



```
# h0:0, ha != 0
#data before bootstrap is not centered so we calculate distance from h0 to mean of distribution
diff <- abs(0-mean(results$t))
p_value <- (sum(results$t > (mean(results$t) + diff)) + sum(results$t < (mean(results$t) - diff)))/2000
```

For this hypothesis testing let our hypothesis be.

$H_0 : t = 0$ - lottery is random

$H_a : t \neq 0$ - lottery is not random

We will consider two sided p-value: 0.0035. That is because we account for values that would reject the null hypothesis above and below $t = 0$.

As p-value results in less than 0.05 that lead us reject null hypothesis, meaning that lottery is not random.

4.

Permutation test function will be used in this occasion to evaluate the next hypothesis analysis with $B = 2000$:

$H_0 : t = 0$ - lottery is random

$H_a : t \neq 0$ -lottery is not random

```
permutation_test <- function(data, B){

  origin_loes <- loess(Draft_No ~ Day_of_year, data)
  t_origin <- statis(data, origin_loes)

  stat= numeric(B)
  n = dim(data)[1]
  for(b in 1:B){
    perm_data <- data.frame(data)
    perm_data$Day_of_year = sample(data$Day_of_year, n)
    loes_h1 <- loess(Draft_No ~ Day_of_year, perm_data)
    stat[b] <- statis(perm_data,loes_h1)
  }
  # statistic from original dat

  p_value <- sum(abs(stat) >= abs(t_origin))/B
  return(p_value)
}
```

The permutation test when B = 2000 results in a p-value = 0.149. p-value > 0.05, meaning that we have no evidence to reject null hypothesis - we cant say that lottery is not random

5.

A crude estimate of the power of the test constructed in Step 4 will be made.

```
new_y <- function(x, alpha){
  set.seed(12345)
  beta <- rnorm(1, 183, 10)
  y <- max(0, min(alpha * x + beta, 366) )
  return(y)
}


### 5-3
new_alpha <-seq(0.1, 10, 0.1)
p_val <- c()
for( i in 1:length(new_alpha)){
  y <- sapply(new_df$Day_of_year,new_y, alpha = new_alpha[i])
  new_dataset <- data.frame(Day_of_year = new_df$Day_of_year, Draft_No = y )
  p_val[i]<-permutation_test(new_dataset,200)
}
```

For all alpha p-values are 0

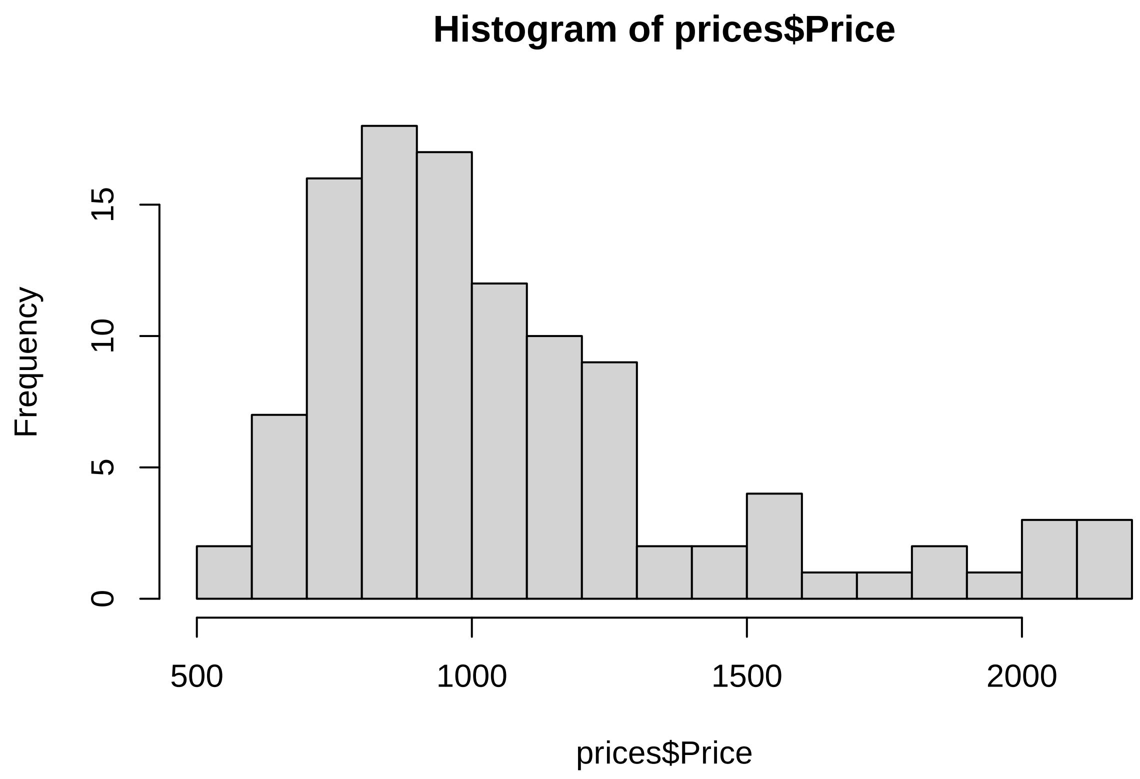
The computed power of the test is: 1-type 2 error. type 2 error is a probability of failing to reject H_0 when H_a is true. We know that our generated data samples are not random. The amount of rejected $H_0 : 0$. As a result type 2 error is: 0, and power of the test is: 1.

Question 2: Bootstrap, jackknife and confidence intervals

In this section we are required to use bootstrapping and jackknife techniques to estimate some unknown parameters at population level based on the sampling knowledge comprised in confidence intervals.

1.

Histogram of the Data and distribution-based approach.



It resembles a Gamma distribution with shape = 3, scale = 2.

2.

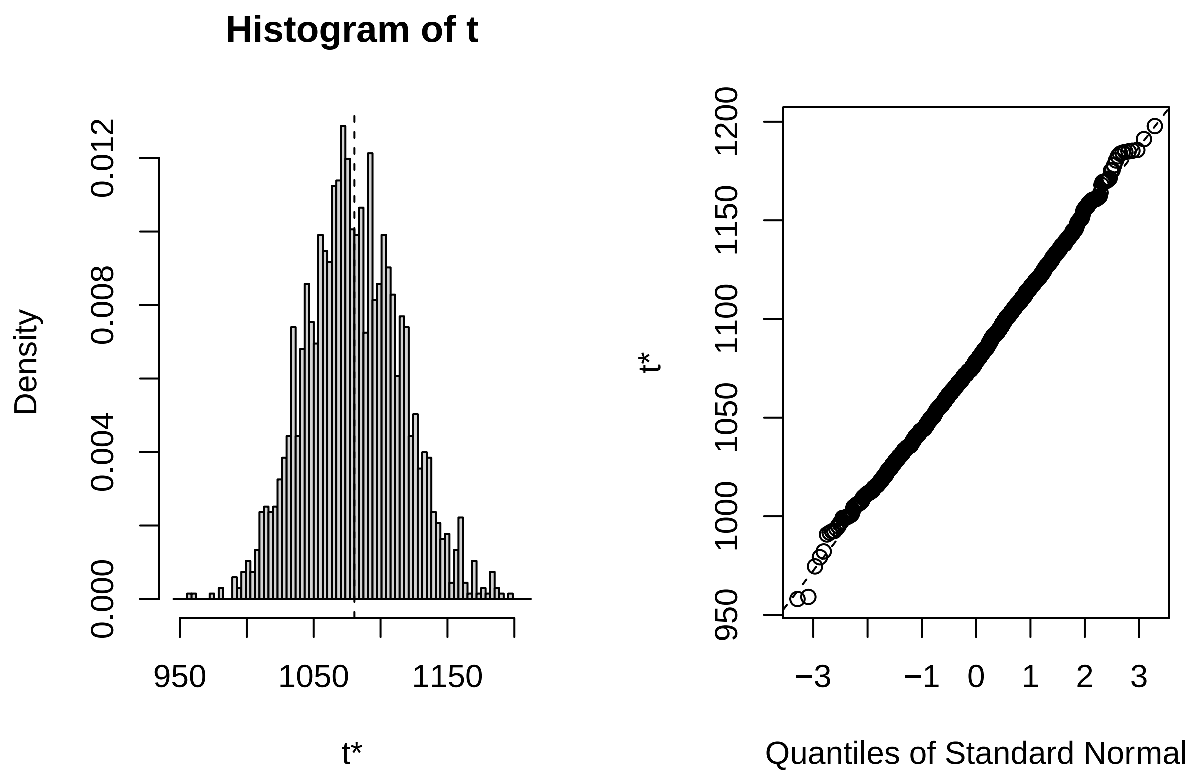
```
## Mean
mean_price <- mean(prices$Price)

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


mean_stat <- function(data, vn){
  data<- as.data.frame(data[vn, ])
  data_mean <- as.numeric(colMeans(data['Price']))
  data_mean
}

ans <- boot(data = prices, statistic = mean_stat, R=2000)

plot(ans)
```



```
# CI percentil
ciperc <- boot.ci(ans, index = 1, type=c('perc'))

# CI bca

cibca <- boot.ci(ans, index =1, type=c('bca'))

# CI normal

cinorm <- boot.ci(ans, index =1, type=c('norm'))

## Bias-correction

T_bias_correction <- 2*mean_price - mean(ans$t)

## variance
boot_var <- (1/(ans$R-1))* sum((ans$t-mean(ans$t))^2)

conf_int <- rbind(c(ciperc$percent[4:5]), c(cibca$bca[4:5]), c(cinorm$normal[2:3]))
row.names(conf_int) <- c("percentile", "BCa", "first-order normal")
colnames(conf_int) <- c("low", "high")
```

After applying the aforementioned techniques we encountered this parameter estimations.

To calculate the Bias-Correction we applied the ensuing formula. Note B will be our number of sub-samplings, T stands for our Data statistic and T_i^* our statistic for sub-samplings.

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

In order to calculate the variance, the following formula will be used:

$$Var[\hat{T}(\mathring{u})] = \frac{1}{B-1}\sum_{i=1}^{B}(T(D_i)^* - \bar{T}(D^*)^2))$$

Bootstrap bias-correction: 1081.1160955. Variance - 1280.567486.

95% confidence intervals:

	low	high
percentile	1012.310	1152.771
BCa	1017.472	1160.455
first-order normal	1010.979	1151.253

3.

For this section the same thing as in the previous point will be calculated by using jackknife instead. This estimation will be ruled by this mathematical expression:

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

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

```r
#3 jackknife

jackknife <- function(data, R, func){
  R <- ifelse(length(data) <= R, length(data), R)
  t <- vector()
  for (i in 1:R) {
    t[i] <- func(data[-i])
  }

  Ti <- R*func(data) - (R-1)*t
  var <- (1/(R*(R-1)))*sum((Ti-mean(t))^2)
  results <- list(R=R, t=t, t_mean = mean(t), variance = var)
  return(results)
}

res <- jackknife(prices$Price, 2000, mean)
```

Variance using jackknife: 1320.9110441, the difference obtained between jackknife and bootstrap is: 40.343558. As can we see the jackknife method tends to overestimate variance.

4.

In the fourth point we will compare the confdence intervals obtained with respect to their length and the location of the estimated mean in such intervals.

```r
location <- function(high,low, mean){
  length <- high-low
  loc <- (mean-low) /length
  res <- list(length = length, location = loc)
  return(res)
}

loc_ciperc <- location(ciperc$percent[5],ciperc$percent[4], mean_price)
loc_ciBCA <- location(cibca$bca[5],cibca$bca[4], mean_price)
loc_cinorm <- location(cinorm$normal[3],cinorm$normal[2], mean_price)

conf_int_comp <- rbind(c(ciperc$percent[4:5], loc_ciperc$length, loc_ciperc$location),
                  c(cibca$bca[4:5],loc_ciBCA$length, loc_ciBCA$location),
                  c(cinorm$normal[2:3], loc_cinorm$length, loc_cinorm$location))
row.names(conf_int_comp) <- c("percentile", "BCa", "first-order normal")
colnames(conf_int_comp) <- c("low", "high", "length", "location of mean")
```

The table below compares confidence intervals:

	low	high	length	location of mean
percentile	1012.310	1152.771	140.4607	0.4852763
BCa	1017.472	1160.455	142.9831	0.4406176
first-order normal	1010.979	1151.253	140.2747	0.4954135

The mean location shows the portion of interval length from the beginning of the interval until the mean.