MTH209

Homework 2

Group 9

Dataset

Link for the data set used in the solution - https://www.kaggle.com/datasets/nitingoyal8/jio-cinema-userbase-dataset?resource=download

We perform all the tests on the dataset of Jio Cinema monthly revenue per user data.

```
library(ggplot2)

# Load dataset
a <- read.csv("jio_cinema_users_extended.csv")
head(a)
```

```
  User.ID Subscription.Type Monthly.Revenue  Join.Date Last.Payment.Date
1 JIO1000             Basic           14.60 2020-11-22        2023-11-06
2 JIO1001           Premium            8.83 2022-04-05        2024-08-08
3 JIO1002              Free            3.07 2020-02-01        2023-09-24
4 JIO1003              Free           12.00 2023-08-07        2024-05-29
5 JIO1004              Free            1.54 2023-12-24        2024-11-11
6 JIO1005             Basic            4.69 2021-04-16        2024-01-03
    Country Age Gender   Device Plan.Duration..Months.
1     India  44  Other Smart TV                      1
2        UK  32   Male   Mobile                     12
3        UK  43 Female   Mobile                     12
4        UK  51 Female   Mobile                      1
5    Canada  59  Other   Laptop                     12
6 Australia  44   Male   Laptop                     12
```

```r
##Complete Dataset
```

```r
data <- unlist(a[[3]], use.names = FALSE)
print(length(data))
```

```
[1] 500
```

```r
head(data)
```

```
[1] 14.60  8.83  3.07 12.00  1.54  4.69
```

The data set is of size 500.

Problem 1

Hypothesis Test

We test the hypothesis:

$$H_0 : F = F_0, \text{where } F_0 \text{ is the standard normal distribution,}$$

against the alternative hypothesis:

$$H_a : F \neq F_0$$

Method 1 : Using Cramer-von Mises test

Standardizing the data

```r
library(ggplot2)

# Standardizing data
nu <- mean(data)
sig <- sd(data)
Sdata <- (data - nu) / sig
```

CVM Test Statistic Function

```r
Tn <- function(x) {
  x<-sort(x)
  s<-0
  for(i in 1:500){
    s<-s + (i/500 - pnorm(x[i]))^2
  }
  return(s)
}
```

Bootstrapping Process

```r
set.seed(42)  # For reproducibility
T <- numeric(1000)
for (i in seq_len(1000)) {
  y <- sample(Sdata, size = length(Sdata), replace = TRUE)
  T[i] <- Tn(y)
}
```

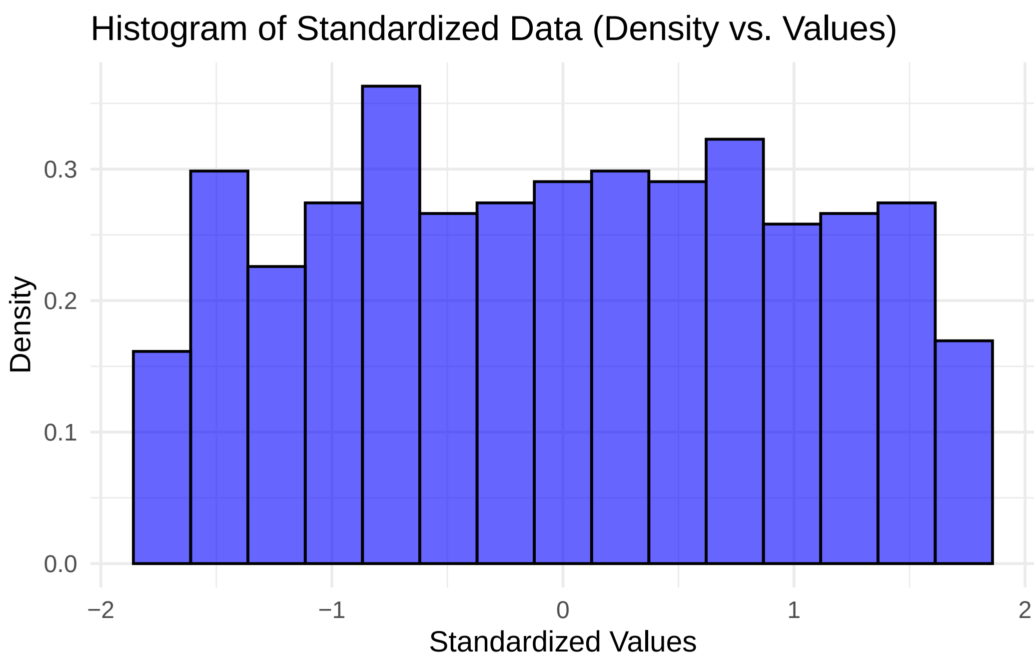
Calculating p-value

```r
p_value <- mean(T > Tn(Sdata))
p_value
```

```
[1] 0.714
```

Visualization

```
# Histogram with density on y-axis
ggplot(data.frame(Sdata), aes(x = Sdata, y = after_stat(density))) +
  geom_histogram(bins = 15, fill = "blue", alpha = 0.6, color = "black") +
  labs(title = "Histogram of Standardized Data (Density vs. Values)",
       x = "Standardized Values", y = "Density") +
  theme_minimal()
```



Conclusion

If the p-value is small (typically < 0.02) (p-values in the range of 0.02-0.05 are likely to follow the null hypothesis) , we say that most likely the test does not favor the null hypothesis suggesting that the data **does not follow a normal distribution**. Otherwise, it is most likely that **the test favors the null hypothesis** and follows normal distribution.

Method 2

Kolmogorov-Smirnov (KS) Test

To test this hypothesis, we use the **Kolmogorov-Smirnov (KS) test**, which is based on the **Kolmogorov-Smirnov statistic:**

$$D_n = \sup_x |F_n(x) - F_0(x)|$$

where:
- ($F_n(x)$) is the *empirical distribution function (EDF)* of the sample,
- ($F_0(x)$) is the cumulative distribution function (CDF) of the standard normal distribution, and
- ($\sup_x$) represents the *supremum* (maximum absolute difference) over all values of (x).

The p-value of the KS test determines whether we accept or reject (H0). If the p-value is less than **0.02**, we reject the null hypothesis, indicating that the sample **most likely does not follow a standard normal distribution**. Otherwise, we accept the null hypothesis.

```r
# Load the data
a <- read.csv("jio_cinema_users_extended.csv")
data <- a[3]
data <- unlist(data, use.names = FALSE)

# Standardizing the data
nu <- mean(data)
sig <- sqrt(var(data))
Sdata <- (data - nu) / sig  # Standardized Data

# Kolmogorov Smirnov Test Statistic function
Tn<-function(x){
  x <- sort(x)
  s <- 0
  n <- length(x)

  # Calculate the Kolmogorov Test statistic
  for(i in 1:n){
    s <- max(s,abs(i/n - pnorm(x[i])))
  }
  return(s)
}
```

```r
# For storing test statistics of bootstrapped samples
T <- numeric(1e3)

# Bootstrapping for KS
for(i in 1:1e3){
  y <- sample(Sdata, size = 500, prob = rep(1, 500), replace = TRUE)
  T[i] <- Tn(y)
}

# Calculating p-value for KS
observed_Tn <- Tn(Sdata)
p <- sum(T > observed_Tn) / 1e3

# Print the p-value
print(p)
```

```
[1] 0.858
```

Question 2

Hypothesis Test

For each statistic (mean, median, mode), we test the following hypotheses:

Null Hypothesis ((H_0))

$$H_0 : F_{\text{statistic}} = F_0, \quad \text{where } F_0 \text{ is the standard normal distribution.}$$

This means that after normalization, the sample statistic follows a normal distribution.

Alternative Hypothesis ((H_a))

$$H_a : F_{\text{statistic}} \neq F_0$$

This means that after normalization, the sample statistic follows a normal distribution.

normalization, the sample statistic does not follow a normal distribution.

Application to Different Statistics

This hypothesis applies separately to:

- **Sample Mean**
- **Sample Median**
- **Sample Mode**

$$H_0 : F_{\text{mode}(X)} = F_0, \quad H_a : F_{\text{mode}(X)} \neq F_0$$

For statistic (**mean and median**), we test:

```
set.seed(42)

# Resampling and calculating mean & median 500 times
means <- numeric(500)
medians <- numeric(500)

for(i in 1:500){
  y <- sample(data, size = 500, replace = TRUE)
  means[i] <- mean(y)
  medians[i] <- median(y)
}

# Standardizing Mean (Correct CLT Scaling)
Smean <- (means - mean(data)) * sqrt(500) / sd(data)

# Standardizing Median (Using PDF Estimation at Median)
fhat <- density(data)$y[which.min(abs(density(data)$x - median(data)))]  # Estimate PDF at me
Smedian <- (medians - median(data)) * sqrt(500) / (1 / (2 * fhat))  # Corrected scaling


#Now we have to check weather Smean,Smedian follow N(0,1) whcih we will do in
#the same way in which we verified for Sdata


# Kolmogorov-Smirnov Test Statistic function
Tn <- function(x) {
  x <- sort(x)
  s <- 0
  n <- length(x)
```

```r
  for(i in 1:n){
    s <- max(s, abs(i/n - pnorm(x[i])))
  }
  return(s)
}

# Bootstrapping for KS Test (Efficient Vectorized)
T1 <- replicate(1000, Tn(sample(Smean, 500, replace = TRUE)))  # Tn's for means
T2 <- replicate(1000, Tn(sample(Smedian, 500, replace = TRUE)))  # Tn's for medians

# Observed KS Test Statistic
TnMean <- Tn(Smean)
TnMedian <- Tn(Smedian)

#p-value calculation
pMean <- sum(T1 > TnMean) / 1000
pMedian <- sum(T2 > TnMedian) / 1000

# Print the p-values
print(paste("p-value for Mean:", pMean))
```

```
[1] "p-value for Mean: 0.898"
```

```r
print(paste("p-value for Median:", pMedian))
```

```
[1] "p-value for Median: 0.528"
```

Since, these p-values turn out be >0.02 , **the scaled sample median and mean most likely follow the normal distribution**.

For **mode** we use density estimation to estimate the mode:

```r
get_mode <- function(data) {
  # Define the Uniform Kernel function
  uniform_kernel <- function(x) {
    if (abs(x) <= 1) return(0.5) else return(0)
  }

  # Kernel Density Estimation (Uniform Kernel)
  kde_uniform <- function(x, data, bandwidth) {
```

```r
    n <- length(data)
    kde_values <- sapply(x, function(xi) {
      sum(sapply((xi - data) / bandwidth, uniform_kernel)) / (n * bandwidth)
    })
    return(kde_values)
  }

  # Define range for KDE estimation (before normalization)
  x_vals <- seq(min(data), max(data), length.out = 1000)

  # Compute KDE with Uniform Kernel
  bandwidth <- 5  # Adjust bandwidth based on data range
  density_vals <- kde_uniform(x_vals, data, bandwidth)

  # Estimate Mode: Find x corresponding to max KDE value
  mode_index <- which.max(density_vals)
  estimated_mode <- x_vals[mode_index]

  return(estimated_mode)
}

sample_mode = get_mode(data)
# Print Estimated Mode
print(paste("Estimated Mode (Before Normalization):", sample_mode))
```

```
[1] "Estimated Mode (Before Normalization): 5.9261961961962"
```

```r
set.seed(42)  # For reproducibility

set.seed(123)  # For reproducibility

# Function to estimate mode using Kernel Density Estimation
estimate_mode <- function(x) {
  dens <- density(x)
  return(dens$x[which.max(dens$y)])
}




n <- 100  # Sample size per iteration
```

```r
num_simulations <- 1000  # Number of samples

mode_values <- numeric(num_simulations)

for (i in 1:num_simulations) {
  sample_data <- sample(data, n, replace = TRUE)  # Sampling from given data
  mode_values[i] <- estimate_mode(sample_data)
}

# Normalize the mode values
mode_values <- (mode_values - mean(mode_values)) / sd(mode_values)


# Normality tests
shapiro.test(mode_values)  # Shapiro-Wilk test
```

```
	Shapiro-Wilk normality test

data:  mode_values
W = 0.94388, p-value < 2.2e-16
```

```r
ks.test(mode_values, "pnorm")  # Kolmogorov-Smirnov test
```

```
	Asymptotic one-sample Kolmogorov-Smirnov test

data:  mode_values
D = 0.12081, p-value = 4.213e-13
alternative hypothesis: two-sided
```

Conclusion :

Since the p-value for Mode is too low, it is most likely that the null hypothesis is rejected i.e. the mode doesn't follow normal distribution