Nithin S
221IT085

IT302 Lab Assignment 3

Q1. Using standard normal distribution, write a R/Python code for any normal
random variable, to the probability it is
(a) Within 1 standard deviation of its mean.
(b)Within 2 standard deviations of its mean.
(c) Within 3 standard deviations of its mean.

Code

```
prob_within_1_sd <- pnorm(1) - pnorm(-1)
prob_within_2_sd <- pnorm(2) - pnorm(-2)
prob_within_3_sd <- pnorm(3) - pnorm(-3)

cat("Probability within 1 standard deviation:", prob_within_1_sd, "\n")
cat("Probability within 2 standard deviations:", prob_within_2_sd, "\n")
cat("Probability within 3 standard deviations:", prob_within_3_sd, "\n")
```

Output

```
> prob_within_1_sd <- pnorm(1) - pnorm(-1)
> prob_within_2_sd <- pnorm(2) - pnorm(-2)
> prob_within_3_sd <- pnorm(3) - pnorm(-3)
>
> cat("Probability within 1 standard deviation:", prob_within_1_sd, "\n")
Probability within 1 standard deviation: 0.6826895
> cat("Probability within 2 standard deviations:", prob_within_2_sd, "\n")
Probability within 2 standard deviations: 0.9544997
> cat("Probability within 3 standard deviations:", prob_within_3_sd, "\n")
Probability within 3 standard deviations: 0.9973002
>
```

Q2. The length of a long-distance telephone calls follows normal distribution with mean of 20 minutes and standard deviation of 5 minutes. Suppose a sample of 60 telephone calls is used to reflect on the population of all longdistance calls. Write a R/Python code to find the following. (a) About how many calls are between 16 and 18 minutes? (b) How many minutes would include within 3 standard deviations of its mean? (c) A 12-minute call would be how many standard deviations below the mean?

Code

```
mean_call <- 20
std_dev_call <- 5
sample_size <- 60

prob_between_16_18 <- pnorm(18, mean = mean_call, sd = std_dev_call) - pnorm(16, mean = mean_call, sd = std_dev_call)
calls_between_16_18 <- prob_between_16_18 * sample_size

lower_bound_3sd <- mean_call - 3 * std_dev_call
upper_bound_3sd <- mean_call + 3 * std_dev_call

z_score_12_min <- (12 - mean_call) / std_dev_call

cat(" (a) Estimated number of calls between 16 and 18 minutes:", calls_between_16_18, "\n")
cat(" (b) Minutes range within 3 standard deviations:", lower_bound_3sd, " mins to", upper_bound_3sd,
    " mins i.e., it includes", (upper_bound_3sd - lower_bound_3sd), "minutes.\n")

cat(" (c) A 12-minute call is", abs(z_score_12_min), "standard deviations below the mean.\n")
```

Output

```
> mean_call <- 20
> std_dev_call <- 5
> sample_size <- 60
>
> prob_between_16_18 <- pnorm(18, mean = mean_call, sd = std_dev_call) - pnorm(16, mean = mean_call, sd = std_dev_call)
> calls_between_16_18 <- prob_between_16_18 * sample_size
>
> lower_bound_3sd <- mean_call - 3 * std_dev_call
> upper_bound_3sd <- mean_call + 3 * std_dev_call
>
> z_score_12_min <- (12 - mean_call) / std_dev_call
>
> cat(" (a) Estimated number of calls between 16 and 18 minutes:", calls_between_16_18, "\n")
 (a) Estimated number of calls between 16 and 18 minutes: 7.963372
> cat(" (b) Minutes range within 3 standard deviations:", lower_bound_3sd, " mins to", upper_bound_3sd,
+     " mins i.e., it includes", (upper_bound_3sd - lower_bound_3sd), "minutes.\n")
 (b) Minutes range within 3 standard deviations: 5  mins to 35  mins i.e., it includes 30 minutes.
>
> cat(" (c) A 12-minute call is", abs(z_score_12_min), "standard deviations below the mean.\n")
 (c) A 12-minute call is 1.6 standard deviations below the mean.
>
```

Q3. Use Monte Carlo simulations in R/Python to simulate the following scenario. It’s my 28th birthday, and my friends bought me a cake with 28 candles on it. I make a wish and try to blow them out. Every time, I blow out a random number of candles between 1 and the number that remains, including one and that other number. How many times, on average, do I blow before all the candles are extinguished?

Code

```r
num_simulations <- 10000
initial_candles <- 28

blow_out_candles <- function() {
  candles_left <- initial_candles
  blows <- 0

  while (candles_left > 0) {
    blow <- sample(1:candles_left, 1)
    candles_left <- candles_left - blow
    blows <- blows + 1
  }

  return(blows)
}

results <- replicate(num_simulations, blow_out_candles())
average_blows <- mean(results)
cat("On average, it takes", average_blows, "blows to extinguish all the candles.\n")
```

Output

```
> num_simulations <- 100000
> initial_candles <- 28
>
> blow_out_candles <- function() {
+   candles_left <- initial_candles
+   blows <- 0
+
+   while (candles_left > 0) {
+     blow <- sample(1:candles_left, 1)
+     candles_left <- candles_left - blow
+     blows <- blows + 1
+   }
+
+   return(blows)
+ }
>
> results <- replicate(num_simulations, blow_out_candles())
> average_blows <- mean(results)
> cat("On average, it takes", average_blows, "blows to extinguish all the candles.\n")
On average, it takes 3.93242 blows to extinguish all the candles.
>
```

Q4. Use Monte Carlo simulations in R/Python to simulate the following scenario. Two teams, say Cavs and Warriors, are playing a seven-game championship series. The first to win four games, therefore, win the series. The teams are equally good, so they each have a 50-50 chance of winning each game. If the Cavs lose the first game, what is the probability that they win the series?

Code

```r
simulate_series <- function() {
  cavs_wins <- 0
  warriors_wins <- 1
  while (cavs_wins < 4 && warriors_wins < 4) {
    if (runif(1) < 0.5) {
      cavs_wins <- cavs_wins + 1
    } else {
      warriors_wins <- warriors_wins + 1
    }
  }
  return(cavs_wins == 4)
}

# Monte Carlo Simulation
num_simulations <- 1000000
cavs_win_count <- sum(replicate(num_simulations, simulate_series()))
probability_cavs_win <- cavs_win_count / num_simulations

cat("Probability that the Cavs win the series after losing the first game:", probability_cavs_win, "\n")
```

Output

```
> simulate_series <- function() {
+   cavs_wins <- 0
+   warriors_wins <- 1
+   while (cavs_wins < 4 && warriors_wins < 4) {
+     if (runif(1) < 0.5) {
+       cavs_wins <- cavs_wins + 1
+     } else {
+       warriors_wins <- warriors_wins + 1
+     }
+   }
+   return(cavs_wins == 4)
+ }
>
> # Monte Carlo Simulation
> num_simulations <- 1000000
> cavs_win_count <- sum(replicate(num_simulations, simulate_series()))
> probability_cavs_win <- cavs_win_count / num_simulations
>
> cat("Probability that the Cavs win the series after losing the first game:", probability_cavs_win, "\n")
Probability that the Cavs win the series after losing the first game: 0.34344
>
```

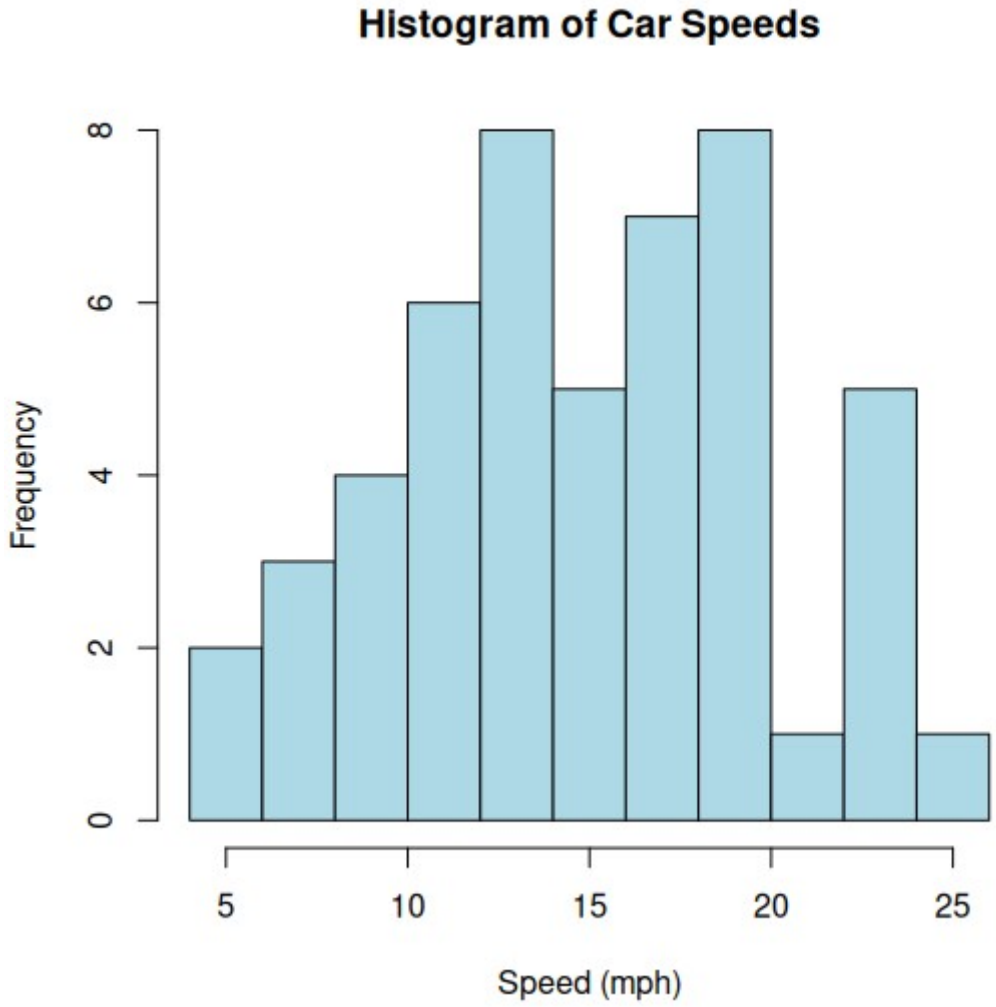
Q5. Consider the cars dataset in R. This data frame has 50 observations on two variables: speed and distance. For speed data using R/Python, find
(a) Mean, variance, standard deviation.
(b) Median, mode, standard error.
(c) Draw the histogram.

Code

```r
data(cars)

mean_speed <- mean(cars$speed)
var_speed <- var(cars$speed)
sd_speed <- sd(cars$speed)
median_speed <- median(cars$speed)

get_mode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}
mode_speed <- get_mode(cars$speed)
se_speed <- sd_speed/sqrt(length(cars$speed))

hist(cars$speed,
     main="Histogram of Car Speeds",
     xlab="Speed (mph)",
     ylab="Frequency",
     col="lightblue",
     breaks=10)

summary_stats <- data.frame(
  Statistic = c("Mean", "Variance", "Standard Deviation", "Median", "Mode", "Standard Error"),
  Value = c(mean_speed, var_speed, sd_speed, median_speed, mode_speed, se_speed)
)
print("Summary Statistics:")
print(summary_stats)
```

Output

```
> data(cars)
>
> mean_speed <- mean(cars$speed)
> var_speed <- var(cars$speed)
> sd_speed <- sd(cars$speed)
> median_speed <- median(cars$speed)
>
> get_mode <- function(v) {
+   uniqv <- unique(v)
+   uniqv[which.max(tabulate(match(v, uniqv)))]
+ }
> mode_speed <- get_mode(cars$speed)
> se_speed <- sd_speed/sqrt(length(cars$speed))
>
> hist(cars$speed,
+      main="Histogram of Car Speeds",
+      xlab="Speed (mph)",
+      ylab="Frequency",
+      col="lightblue",
+      breaks=10)
>
> summary_stats <- data.frame(
+   Statistic = c("Mean", "Variance", "Standard Deviation", "Median", "Mode", "Standard Error"),
+   Value = c(mean_speed, var_speed, sd_speed, median_speed, mode_speed, se_speed)
+ )
> print("Summary Statistics:")
[1] "Summary Statistics:"
> print(summary_stats)
          Statistic      Value
1              Mean 15.4000000
2          Variance 27.9591837
3 Standard Deviation  5.2876444
4            Median 15.0000000
5              Mode 20.0000000
6    Standard Error  0.7477858
>
```


Histogram of Car Speeds
Frequency
0
2
4
6
8
5
10
15
20
25
Speed (mph)