Maths coursework

binara

2024-04-07

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ purrr     1.0.2
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

#question 1 #a

```
red_balls<-rep(5,5)
green_balls <-rep(10,3)
bag<-c(red_balls,green_balls)
bag_size=length(bag)

blue_balls<-rep(15,2)
yellow_balls<-rep(20,4)
bag2<-c(blue_balls,yellow_balls)
bag2_size=length(bag2)



output <- expand.grid(bag, bag2)


output$sum <- output$Var1 + output$Var2


values <- unique(output$sum)

# Print the possible values
values
```

```
## [1] 20 25 30
```

```
#In the first bag we have 5 red balls label with 5 and 3 green balls labelled
with 10 and the next bags has 2 blue balls labelled with with 15 and 4 yellow
balls with labelled 20.If we draw a red ball  from the first bag and a blue
ball from the second bag, the sum is 5 + 15 = 20,If we draw a red ball from
the first bag and a yellow ball from the second bag, the sum is 5 + 20 =
25,If we draw a green ball from the first bag and a blue ball from the second
bag, the sum is 10 + 15 = 25, If we draw a green from the first bag and a
yellow ball  from the second bag, the sum is 10 + 20 = 30, these outputs are
the values for X.
```

##b)

```
# Get the unique values of the sum and their counts
count <- table(output$sum)

# Calculate the probabilities
total_outcomes <- sum(count)

pmf <-count / total_outcomes

# Print the probability mass function
pmf
```

```
##
##        20        25        30
## 0.2083333 0.5416667 0.2500000
```

##c)

```
# Calculate the expected value (mean) of X
E_X <- sum(values * pmf)  # Multiply each possible value by its probability

# Calculate the variance of X
Var_X <- sum((values - E_X)^2 * pmf)  # Square the deviation from the mean

# Print the results
cat("Expected value E(X) =", E_X, "\n")
```

```
## Expected value E(X) = 25.20833
```

```
cat("Variance Var(X) =", Var_X, "\n")
```

```
## Variance Var(X) = 11.41493
```

##d)

```
output$Y <- 2*output$sum-3

count <- table(output$Y)
```

```r
# Calculate the probabilities
total_outcomes <- sum(count)
pmf_Y <- count / total_outcomes

# Print the probability mass function of Y
print(pmf_Y)
```

```
##
##        37        47        57
## 0.2083333 0.5416667 0.2500000
```

##e)

```r
# Calculate the cdf of Y
cdf_Y <- cumsum(pmf_Y)

# Create a data frame with possible values of Y and corresponding cdf
cdf_table_Y <- data.frame(y = names(pmf_Y), cdf = cdf_Y)

# Print the cdf of Y
print(cdf_table_Y)
```

```
##     y       cdf
## 37 37 0.2083333
## 47 47 0.7500000
## 57 57 1.0000000
```

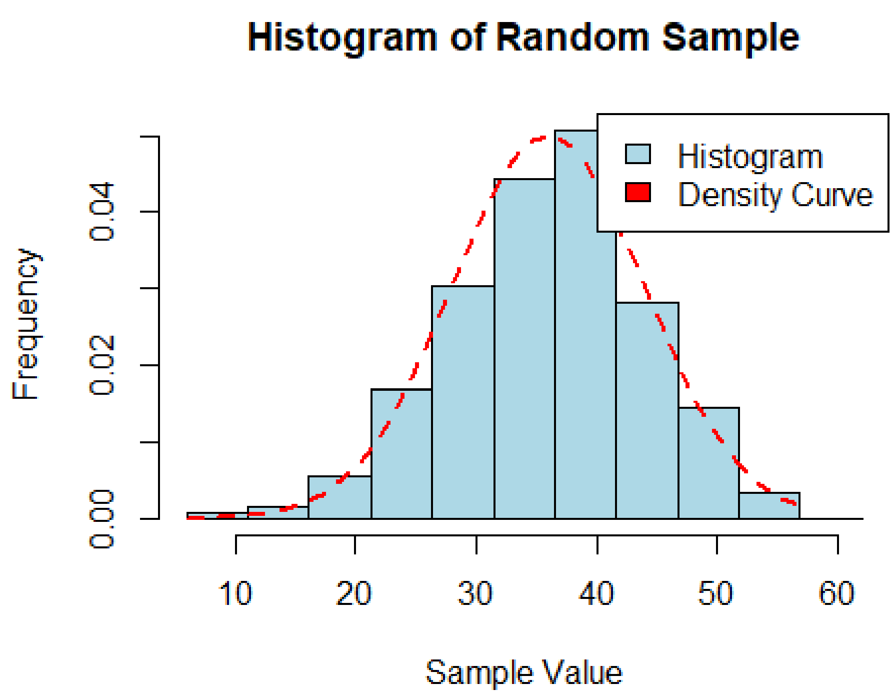
##f)

```r
y_values<-c(37,47,57)
y_cdf <- c(0.2083333, 0.7500000, 1.0000000)


index<-which(y_values==37)

pmf_y<-y_cdf[index]

pmf_y
```

```
## [1] 0.2083333
```

```r
#question 2

#a)
# Set mean and standard deviation
mean <- 36
sd <- 8

# Generate random sample of size 500
sample <- rnorm(n = 500, mean = mean, sd = sd)
```

```
# Calculate bin ranges (assuming you'll find min and max values in the
sample)
min_val <- min(sample)
max_val <- max(sample)
bin_width <- (max_val - min_val) / 10
bins <- seq(from = min_val, to = max_val + bin_width, by = bin_width)

# Create histogram
hist(sample, breaks = bins, col = "lightblue", main = "Histogram of Random
Sample", xlab = "Sample Value", ylab = "Frequency", freq =F )
#b)
# Overlay density curve
x <- seq(from = min_val, to = max_val, length.out = 100)  # 100 points for
smooth curve
density <- dnorm(x, mean = mean, sd = sd)
lines(x, density, col = "red", lwd = 2, lty = 2, legend = "Density Curve")

## Warning in plot.xy(xy.coords(x, y), type = type, ...): "legend" is not a
## graphical parameter

legend("topright", c("Histogram", "Density Curve"), fill =c("lightblue",
"red"))
```

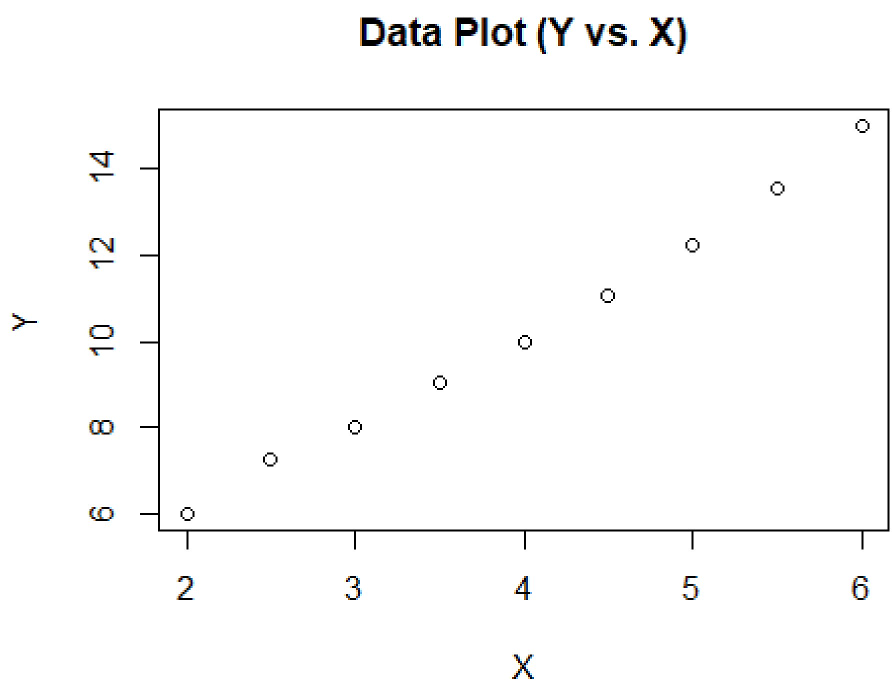



```
#c)
# histogram creates a bell curve centered around the mean 36. that means many
points fall near to the mean value.
 # desity curve which represents the probability density function(PDF) also
```

```
creates a bell curve, since the histogram is bell shaped density curve should
have to be bell shaped. Peek of the curve represents the mean value.
```

#question 3 #a

```
X<- c(2,2.5,3,3.5,4,4.5,5,5.5,6)
Y<- c(6,7.25,8,9.0625,10,11.0625,12.25,13.5625,15)
df <- as.data.frame(cbind(X,Y))

plot(df$X, df$Y, main = "Data Plot (Y vs. X)", xlab = "X", ylab = "Y")
```



#b

```
# Calculate Pearson's correlation coefficient
correlation<- cor(X, Y)
cat("Pearson's coefficient for the data set:",correlation,"\n")

## Pearson's coefficient for the data set: 0.9970374
```

#c

```
# The value got for the pearson's correlation coefficient is 0.9970374, as
this value is very close to +1 this has a positive linear correlation. It
suggest a strong relation between X and Y
```

#d

```
last_sixX <- X[4:9]
last_sixY <- Y[4:9]
```

```r
cor2 <- cor(last_sixX,last_sixY)
cat("Pearson's coefficient for the last six values of the data
set:",cor2,"\n")

## Pearson's coefficient for the last six values of the data set: 0.9970583

# The last six data points have slightly more consistent linear trend.
because the coefficent value as increase by very small amount.
```

#e

```r
X1<- 2*X-1
X2<-X^2

cor_of_X1y<- cor(X1,Y)
cor_of_X2y<- cor(X2,Y)
cor_of_X1y       #when X1=2X-1  r(X1,y)

## [1] 0.9970374

cor_of_X2y       #when X2=X^2 r(X2,y)

## [1] 0.9964801

#r(x,y) and r(x1,y) are similar because x to x1 is a linear transformation.

#r(x,y) and r(x2,y) are not similar because it is bot a linear
transformation, squaring x can tends to positive correlation or negative
correlation since the pearson's correlation coefficent decrease, negative
correlation has occured.
```

#question 4 #a

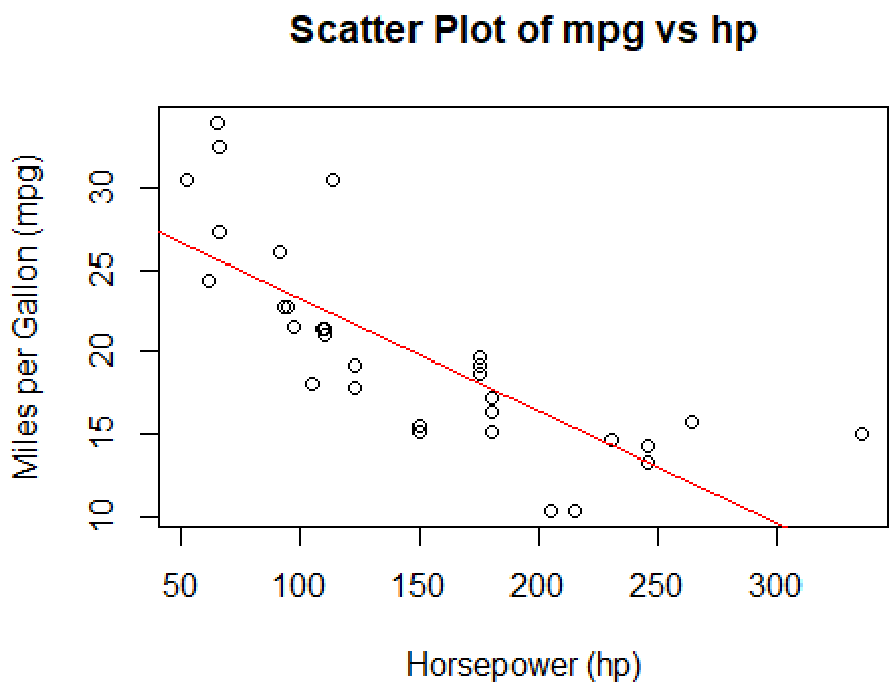
```r
# Load the mtcars dataset
data(mtcars)

# first ten rows of the data  set
head(mtcars, 10)

##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
```

#b

```r
summary(mtcars$mpg)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.40   15.43   19.20   20.09   22.80   33.90

summary(mtcars$hp)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    52.0    96.5   123.0   146.7   180.0   335.0
```

#c,#d

```r
plot(mtcars$mpg ~ mtcars$hp, main = "Scatter Plot of mpg vs hp", xlab =
"Horsepower (hp)", ylab = "Miles per Gallon (mpg)")



model <- lm(mpg ~ hp, data = mtcars)
summary(model)  # Print model summary

##
## Call:
## lm(formula = mpg ~ hp, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.7121 -2.1122 -0.8854  1.5819  8.2360
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.09886    1.63392  18.421  < 2e-16 ***
## hp          -0.06823    0.01012  -6.742 1.79e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.863 on 30 degrees of freedom
## Multiple R-squared:  0.6024, Adjusted R-squared:  0.5892
## F-statistic: 45.46 on 1 and 30 DF,  p-value: 1.788e-07

# Fitted values
fitted_values <- predict(model)

#reggersion line
abline(model, col = "red")  # Red line represents the fitted model
```

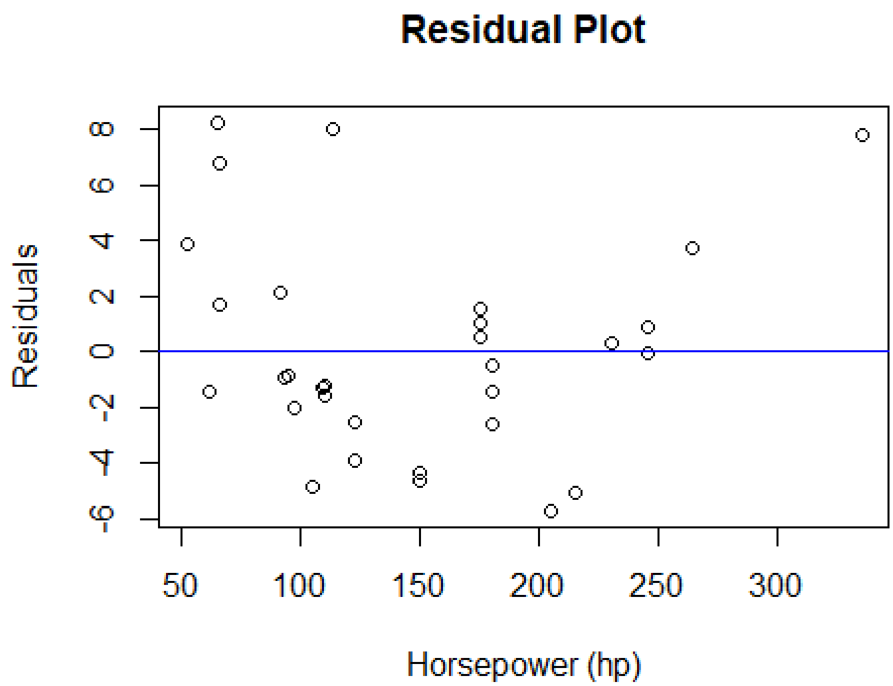


#e

```
intercept<- coef(model)[1]
gradient<-coef(model)[2]
#Fitted equation
cat("Fitted equation: y =", round(intercept, 2), "+", round(gradient, 2),
"x\n")# Extract coefficients

## Fitted equation: y = 30.1 + -0.07 x

cat("for every one unit increase in horse power, mpg is expected to decrease
by",gradient,"\n")# it decrease because of the negative gradient

## for every one unit increase in horse power, mpg is expected to decrease by
-0.06822828
```

#f

```
plot(mtcars$hp, residuals(model), main = "Residual Plot", xlab = "Horsepower
(hp)", ylab = "Residuals")
abline(h = 0, col = "blue")
```




```
# Ideally, residuals should be randomly scattered around the horizontal line
(no pattern). This plot suggests no major deviations from the linear model
assumption. However, a more thorough analysis might be needed for
confirmation.
```

#g

```
new_hp <- 110
predicted_mpg <- predict(model, newdata = data.frame(hp = new_hp))
cat("\nPredicted mpg for hp =", predicted_mpg)

##
## Predicted mpg for hp = 22.59375
```

#question 5 part A #a

```
# Define the mean and standard deviation
mean <- 5.50
sd <- 1.20

#90th percentile
p90 <- qnorm(0.90, mean = mean, sd = sd)
cat("The 90th percentile of customer spending is:", p90, "\n")

## The 90th percentile of customer spending is: 7.037862
```

#b

```r
#25th percentile
p25 <- qnorm(0.25, mean = mean, sd = sd)
cat("The 25th percentile of customer spending is:", p25, "\n")

## The 25th percentile of customer spending is: 4.690612
```

#c

```r
# median value/ 50th percentile
median <- qnorm(0.50, mean = mean, sd = sd)
cat("The median value of customer spending is:", median, "\n")

## The median value of customer spending is: 5.5
```

#d

```r
percentage_above_7 <- (1 - pnorm(7, mean = mean, sd = sd)) * 100
cat("The percentage of customers who spend more than $7.00 is:",
percentage_above_7, "%\n")

## The percentage of customers who spend more than $7.00 is: 10.56498 %
```

#question 5 part B #a

```r
# Parameters for the binomial distribution[^1^][1]
n <- 50   # size
p <- 0.05  # infection rate


# The binomial distribution is suitable here
```

#b

```r
probaForfewer3 <- pbinom(2, size=n, prob=p)
cat("Probability that fewer than 3 individuals are infected in the first
scenario:", probaForfewer3, "\n")

## Probability that fewer than 3 individuals are infected in the first
scenario: 0.5405331
```

#c

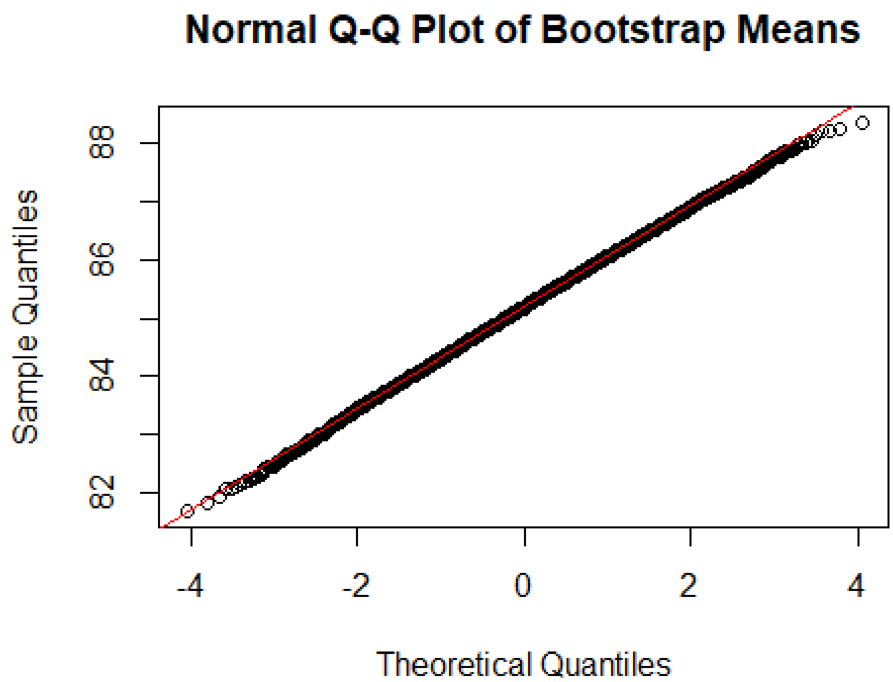
```r
mean <- n * p
variance<- n * p * (1 - p)
cat("Mean of X in the first scenario:", mean, "\n")

## Mean of X in the first scenario: 2.5

cat("Variance of X in the first scenario:", variance, "\n")

## Variance of X in the first scenario: 2.375
```

#d

```r
# Still the experment is doing for infected or not and for large sample size therefore still binomial distribution is okay.
n_new <- 200  # new size
p_new <- 0.02  # new infection rate
```

#question 6

```r
# Define the dataset
exam_scores <- c(82, 88, 75, 94, 90, 85, 78, 91, 86, 89, 92, 80, 87, 79, 84, 77, 83, 81, 76, 93, 88, 85, 89, 90, 82, 86, 75, 91, 79, 84, 78, 95, 88, 87, 93, 86, 82, 89, 90, 80)


set.seed(123)


calc_mean <- function(data, indices) {
  mean(data[indices])
}


bootstrapped_means <- replicate(20000, calc_mean(exam_scores, sample(length(exam_scores), replace = TRUE)))


lower_ci <- quantile(bootstrapped_means, 0.1 / 2)
upper_ci <- quantile(bootstrapped_means, 1 - 0.1 / 2)


cat("90% Bootstrap Percentile Confidence Interval for the Mean:", lower_ci,upper_ci,"\n")

## 90% Bootstrap Percentile Confidence Interval for the Mean: 83.75 86.6

qqnorm(bootstrapped_means, main = "Normal Q-Q Plot of Bootstrap Means")
qqline(bootstrapped_means, col = "red")
```


Normal Q-Q Plot of Bootstrap Means
Sample Quantiles
82
84
86
88
-4
-2
0
2
4
Theoretical Quantiles