(1) Using R execute the basic commands, array, list and frames.

Array

```
# Create two vectors of different lengths.

vector1 <- c(5,9,3)

vector2 <- c(10,11,12,13,14,15)

# Take these vectors as input to the array.

array1 <- array(c(vector1,vector2),dim = c(3,3,2))

array1

column.names <- c("COL1","COL2","COL3")

row.names <- c("ROW1","ROW2","ROW3")

matrix.names <- c("Matrix1","Matrix2")

# Take these vectors as input to the array.

array2 <- array(c(vector1,vector2),dim = c(3,3,2),dimnames =
list(row.names,column.names,matrix.names))

array2

# Print the third row of the second matrix of the array.

array2[3,,2]

# Print the element in the 1st row and 3rd column of the 1st matrix.

array2[1,3,1]

# Print the 2nd Matrix.

array2[,,2]

# Create two vectors of different lengths.
```

```
vector3 <- c(9,1,0)
vector4 <- c(6,0,11,3,14,1,2,6,9)
array2 <- array(c(vector3,vector4),dim = c(3,3,2))
array2
# create matrices from these arrays.
matrix1 <- array1[,,2]
matrix2 <- array2[,,2]
matrix1
matrix2
# Add the matrices.
array3 <- matrix1 + matrix2
array3
# Take these vectors as input to the array.
new.array <- array(c(vector1,vector2),dim = c(3,3,2))
new.array
# Use apply to calculate the sum of the rows across all the matrices.
result <- apply(new.array, c(1), sum)
result
```

list

```
# Create a list containing strings, numbers, vectors and a logical
values.
list1 <- list("Red", "Green", c(21,32,11), TRUE, 51.23, 119.1)
list1
```

```
# Create a list containing a vector, a matrix and a list.
list2 <- list(c("Jan","Feb","Mar"), matrix(c(3,9,5,1,-2,8), nrow = 2),
              list("green",12.3))
# Give names to the elements in the list.
names(list2) <- c("1st Quarter", "A_Matrix", "A Inner list")
# Show the list.
list2
# Access the first element of the list.
list2[1]
# Access the thrid element. As it is also a list, all its elements will be printed.
list2[3]
# Access the list element using the name of the element.
list2$A_Matrix
# Add element at the end of the list.
list2[4] <- "New element"
list2[4]
# Remove the last element.
list2[4] <- NULL
# Print the 4th Element.
list2[4]
# Update the 3rd Element.
list2[3] <- "updated element"
```

```
list2[3]
# Create two lists.
list1 <- list(1,2,3)
list2 <- list("Sun","Mon","Tue")
# Merge the two lists.
mergelist <- c(list1,list2)
# Print the merged list.
mergelist
# Convert the lists to vectors.
v1 <- unlist(list1)
v1
v2 <- unlist(list2)
v2
```

and frames.

```
# Create the data frame.
emp <- data.frame(
  emp_id = c (1:5),
  emp_name = c("Dhiraj","raj","raju","murli","sinha"),
  salary = c(5000,6000,6050,7000,8500),
  start_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15",
"2014-05-11",
                         "2015-03-27")),
  stringsAsFactors = FALSE)
```

```
# Print the data frame.
emp
str(emp)
summary(emp)
result <- data.frame(emp$emp_name,emp$salary)
result
# Extract first two rows.
result <- emp[1:2,]
result
# Extract 3rd and 5th row with 2nd and 4th column.
frame1 <- emp[c(3,5),c(2,4)]
frame1
# Add the "dept" coulmn.
emp$dept <- c("IT","Operations","IT","HR","Finance")
frame2 <- emp
frame2
# Create the second data frame
empnew <-        data.frame(
  emp_id = c (6:8),
  emp_name = c("kumar","Pranab","mishra"),
  salary = c(578.0,722.5,632.8),
  start_date = as.Date(c("2013-05-21","2013-07-30","2014-06-17")),
  dept = c("IT","Operations","Fianance"),
```

```
  stringsAsFactors = FALSE)
# Bind the two data frames.
empfinal <- rbind(frame2,empnew)
empfinal
```

(2) Create a Matrix using R and Perform the operations addition, inverse, transpose and multiplication operations.

```
A <- matrix(c(2,3,-2,1,2,2),3,2)
A
B <- matrix(c(1,4,-2,1,2,1),3,2)
B
C <- A + B
C
A <- matrix(c(4,4,-2,2,6,2,2,8,4),3,3)
inverse <- solve(A)
inverse
transpose <- t(A)
transpose
A <- matrix(c(2,3,-2,1,2,2),3,2)
D <- matrix(c(2,-2,1,2,3,1),2,3)
multiply <- A %*% D
multiply
```

(3) Using R Execute the statistical functions: mean, median, mode, quartiles, range, inter quartile range histogram

```
# Create a vector.
x <- c(12,7,3,4.2,18,2,54,-21,8,-10)
# Find Mean.
result <- mean(x)
result
result <-  mean(x,trim = 0.3)
result
# Create a vector.
x <- c(12,7,3,4.2,18,2,54,-21,8,-5,NA)
# Find mean.
result <-  mean(x)
result
# Find mean dropping NA values.
result <-  mean(x,na.rm = TRUE)
result
# Create the vector.
x <- c(12,7,3,4.2,18,2,54,-21,8,-5)
# Find the median.
median <- median(x)
median
```

```
# Create the function.
getmode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]}
# Create the vector with numbers.
v <- c(2,1,2,3,1,2,3,4,1,5,5,3,2,3)
# Calculate the mode using the user function.
result <- getmode(v)
print(result)
# Create the vector with characters.
charv <- c("o","it","the","it","it")
# Calculate the mode using the user function.
result <- getmode(charv)
print(result)
duration <- c(1,2,3,0,5,6,7,9,1,11,12,23,34,4,45,56)
#find quartiles
quantile(duration)
#find range
range <- max(duration) - min(duration)
range
#find inter quartiles
IQR(duration)
```

#find histogram

hist(duration)

(4). Using R import the data from Excel / .CSV file and Perform the above functions.

(5). Using R import the data from Excel / .CSV file and Calculate the standard deviation, variance, co-variance.

(6). Using R import the data from Excel / .CSV file and draw the skewness.

(7). Import the data from Excel / .CSV and perform the hypothetical testing.

(8). Import the data from Excel / .CSV and perform the Chi-squared Test.

(9). Using R perform the binomial and normal distribution on the data.

Problem

Suppose there are twelve multiple choice questions in an English class quiz. Each question has five possible answers, and only one of them is correct. Find the probability of having four or less correct answers if a student attempts to answer every question at random.

#binomial distribution

Solution

Since only one out of five possible answers is correct, the probability of answering a question correctly by random is 1/5=0.2. We can find the probability of having exactly 4 correct answers by random attempts as follows.

```
dbinom(4, size=12, prob=0.2)
```

To find the probability of having four or less correct answers by random attempts, we apply the function dbinom with $x = 0,\ldots,4$.

```
dbinom(0, size=12, prob=0.2) +
+ dbinom(1, size=12, prob=0.2) +
+ dbinom(2, size=12, prob=0.2) +
+ dbinom(3, size=12, prob=0.2) +
+ dbinom(4, size=12, prob=0.2)
```

Alternatively, we can use the cumulative probability function for binomial distribution pbinom.

```
pbinom(4, size=12, prob=0.2)
```

Answer

The probability of four or less questions answered correctly by random in a twelve question multiple choice quiz is 92.7%.

Create a sample of 50 numbers which are incremented by 1.

dbinom()

```
x <- seq(0,50,by = 1)
```

Create the binomial distribution.

```
y <- dbinom(x,50,0.5)
```

```
# Give the chart file a name.
png(file = "dbinom.png")
# Plot the graph for this sample.
plot(x,y)
```

pbinom()

```
# Probability of getting 26 or less heads from a 51 tosses of a coin.
x <- pbinom(26,51,0.5)
x
```

qbinom()

```
# How many heads will have a probability of 0.25 will come out when a coin is tossed 51 times.
x <- qbinom(0.25,51,1/2)
x
```

rbinom()

```
# Find 8 random values from a sample of 150 with probability of 0.4.
x <- rbinom(8,150,.4)
x
```

#normal distribution

Problem

Assume that the test scores of a college entrance exam fits a normal distribution. Furthermore, the mean test score is 72, and the standard deviation is 15.2. What is

the percentage of students scoring 84 or more in the exam?

Solution

We apply the function pnorm of the normal distribution with mean 72 and standard deviation 15.2. Since we are looking for the percentage of students scoring higher than 84, we are interested in the *upper tail* of the normal distribution.

```
pnorm(84, mean=72, sd=15.2, lower.tail=FALSE)
```

Answer

The percentage of students scoring 84 or more in the college entrance exam is 21.5%.

dnorm()

Create a sequence of numbers between -10 and 10 incrementing by 0.1.

x <- seq(-10, 10, by = .1)

Choose the mean as 2.5 and standard deviation as 0.5.

y <- dnorm(x, mean = 2.5, sd = 0.5)

Give the chart file a name.

png(file = "dnorm.png")

plot(x,y)

pnorm()

Create a sequence of numbers between -10 and 10 incrementing by 0.2.

x <- seq(-10,10,by = .2)

Choose the mean as 2.5 and standard deviation as 2.

y <- pnorm(x, mean = 2.5, sd = 2)

```
# Give the chart file a name.
png(file = "pnorm.png")
# Plot the graph.
plot(x,y)
```

qnorm()

```
# Create a sequence of probability values incrementing by 0.02.
x <- seq(0, 1, by = 0.02)
# Choose the mean as 2 and standard deviation as 3.
y <- qnorm(x, mean = 2, sd = 1)
# Give the chart file a name.
png(file = "qnorm.png")
# Plot the graph.
plot(x,y)
```

rnorm()

```
# Create a sample of 50 numbers which are normally distributed.
y <- rnorm(50)
# Give the chart file a name.
png(file = "rnorm.png")
# Plot the histogram for this sample.
hist(y, main = "Normal DIstribution")
```

(10). Perform the Linear Regression using R.

Simple Linear Regression

lm() Function

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
```

```
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
# Apply the lm() function.
relation <- lm(y~x)
relation
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
# Apply the lm() function.
relation <- lm(y~x)
summary(relation)
```

predict() Function

```
# The predictor vector.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
# The resposne vector.
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
# Apply the lm() function.
relation <- lm(y~x)
relation
# Find weight of a person with height 170.
a <- data.frame(x = 170)
a
result <-  predict(relation,a)
result
# Create the predictor and response variable.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
```

```
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x)
relation
# Give the chart file a name.
png(file = "linearregression.png")
# Plot the chart.
plot(y,x,col = "blue",main = "Height & Weight Regression",
abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab =
"Height in cm")
```

Multiple Linear Regression

lm() Function

```
input <- mtcars[,c("mpg","disp","hp","wt")]
head(input)
input <- mtcars[,c("mpg","disp","hp","wt")]
# Create the relationship model.
model <- lm(mpg~disp+hp+wt, data = input)
# Show the model.
model
# Get the Intercept and coefficients as vector elements.
cat("# # # # The Coefficient Values # # # ","\n")
a <- coef(model)[1]
a
Xdisp <- coef(model)[2]
Xhp <- coef(model)[3]
Xwt <- coef(model)[4]
```

Xdisp

Xhp

Xwt

Logistic Linear Regression

glm() function

```
# Select some columns form mtcars.

input <- mtcars[,c("am","cyl","hp","wt")]

head(input)

input <- mtcars[,c("am","cyl","hp","wt")]

am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)

summary(am.data)
```

(11). Compute the Least squares means using R.

(12). Compute the Linear Least Square Regression.