1)b)Explain Multiple Regression and implement in R Programming

Ans)

1. Multiple Regression

→ Multiple regression is a statistical technique that can be used to analyze the relationship between single dependent variable and several independent variable

→ The goal of multiple regression is to determine the strength and direction of the relationship between the independent variable and dependent variable.

→ In R, multiple regression can be performed using this 'lm()' function, which stands for "linear model"

Syntax:

lm(formula, data)

where 'formula' is an R formula specifying the model and 'data' is the dataset containing the variable used in model

For example, to fit a multiple regression model with two independent variables x1 and x2 and a dependent variable y, the formula would be:

lm(y ~ x1 + x2, data)

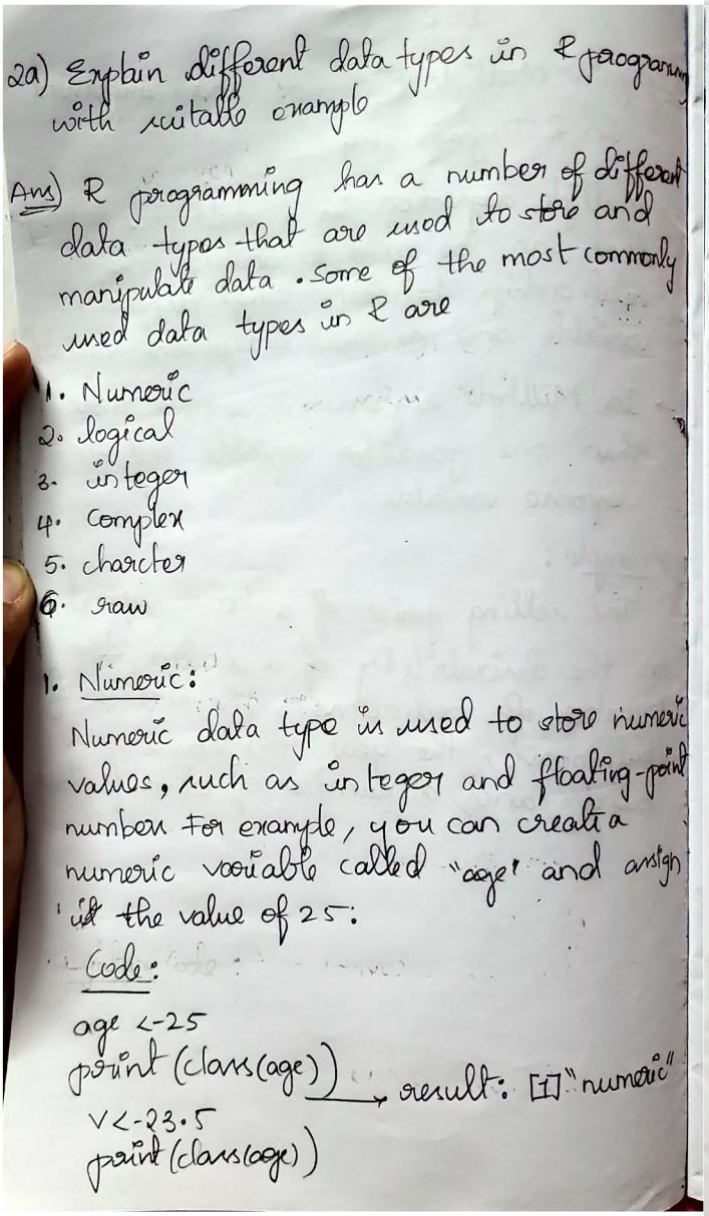
Program

```
input <- mtcars[, c("mpg", "disp", "hp", "wt")]
print(head(input))
model <- lm(mpg ~ disp+hp+wt, data=input)
print(model)
plot(x=input$wt, y=input$mpg,
     xlab="weight", ylab="milage",
     xlim=c(2.5,5), ylim=c(15,30),
     main="weight vs milage")

xdisp <- coef(model)[2]
xhp <- coef(model)[3]
xwt <- coef(model)[4]
print(xdisp)
print(xhp)
print(xwt)
```

2)a)Explain different data types in R Programming with suitable example

Ans)



2a) Explain different data types in R programming with suitable example

Ans) R programming has a number of different data types that are used to store and manipulate data. Some of the most commonly used data types in R are

1. Numeric
2. logical
3. integer
4. Complex
5. charcter
6. raw

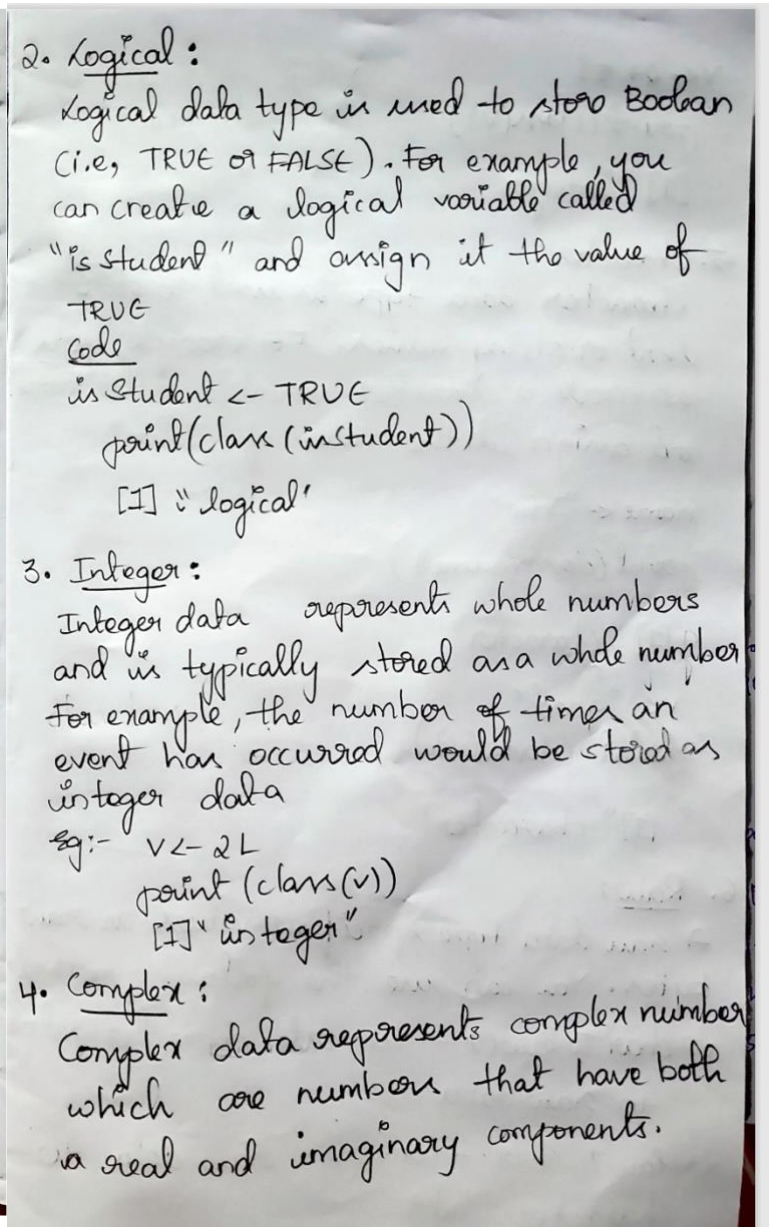
1. Numeric:

Numeric data type is used to store numeric values, such as integer and floating-point number For example, you can create a numeric variable called "age" and assign it the value of 25:

Code:

```
age <-25
print (class(age))
```
→ result: [1] "numeric"

```
V<-23.5
print (class(age))
```



2. Logical:

Logical data type is used to store Boolean (i.e, TRUE or FALSE). For example, you can create a logical variable called "is student" and assign it the value of TRUE

Code

```
is student <- TRUE
print(class (instudent))
```

[1] : logical'

3. Integer:

Integer data represents whole numbers and is typically stored as a whole number For example, the number of times an event has occurred would be stored as integer data

eg:-
```
V<- 2L
print (class(v))
```
[1] " integer"

4. Complex:

Complex data represents complex number which are numbers that have both a real and imaginary components.

```
v<- 2+ 5i
print(class(v))
[1] "integer"
```

5. Character:
character data type is used to store text or string values. For example, you can create a character variable called "name" and assign it the value of "John"

```
name <- "John"
print (class (name))
[1] "character"
v<- "good"
print (class (v))
[1] "character"
```

6. Raw:
A raw data type specifies values as raw bytes. You can use the following method
1. charToRaw() - Converts character data to raw data
2. RawTo char - Converts raw data to character data

```
Eg: v<- charToRaw
    print (class (v))
    [1] "raw"
```

3)a)What is data frame in R programming? Explain how to import datasets in R Programming?

In R programming, a data frame is a table-like data structure that is used to store and manipulate data. Each row in a data frame represents an observation and each column represents a variable. Data frames are similar to tables in a relational database, and they are the most commonly used data structure in R for working with data.

There are several ways to import datasets into R, here are a few examples:

1. `read.csv()` function: This function is used to read a CSV (Comma Separated Values) file and create a data frame. For example, you can import a CSV file called "data.csv" using the following command:

Copy code

```
data <- read.csv("data.csv")
```

2. `read.table()` function: This function is used to read a text file and create a data frame. For example, you can import a text file called "data.txt" using the following command:

Copy code

```
data <- read.table("data.txt", sep = "\t", header = TRUE)
```

3. `read_csv()`, `read_tsv()`, `read_delim()`,`read_excel()`,`read_sas()`,`read_stata()`,`read_spss()`,`read_json()`,`read_sql()`: These functions are provided by the `readr` package and are used to read different types of files like CSV, TSV, Excel, SAS, Stata, SPSS and SQL files. For example, you can import a CSV file called "data.csv" using the following command:

Copy code

```
library(readr)
data <- read_csv("data.csv")
```

6. `RMySQL` and `ROracle` package to connect to SQL databases and import data.

file.choose(): It opens a menu to choose a csv file from the desktop.
header: It is to indicate whether the first row of the dataset is a variable name or not. Apply T/True if the variable name is present else put F/False.

4)b) Write program in R to implement K-means Algorithm

ans)

Here is an example of how to implement the K-means algorithm in R:

Copy code

```r
# load the required libraries
library(tidyverse)
library(cluster)

# load the dataset
data <- iris

# set the number of clusters (k)
k <- 3

# run the k-means algorithm
kmeans_model <- kmeans(data[,1:4], k)

# view the results
print(kmeans_model)
```

In this example, we first load the `tidyverse` and `cluster` libraries, which are required for the k-means algorithm. Next, we load the `iris` dataset and set the number of clusters (k) to 3. Then, we run the k-means algorithm on the first 4 columns of the `iris` dataset and store the result in the `kmeans_model` object. Finally, we print the results of the k-means algorithm to view the cluster centers, within-cluster sum of squares, and other information.

You can adjust this example to suit your specific dataset and needs.

(or)

```
# Installing Packages
install.packages("ClusterR")
install.packages("cluster")
# Loading package
library(ClusterR)
library(cluster)
iris_1 <- iris[, -5]
set.seed(240) # Setting seed
kmeans.re <- kmeans(iris_1, centers = 3, nstart = 20)
kmeans.re
kmeans.re$cluster
cm <- table(iris$Species, kmeans.re$cluster)
Cm
plot(iris_1[c("Sepal.Length", "Sepal.Width")])
plot(iris_1[c("Sepal.Length", "Sepal.Width")],
   col = kmeans.re$cluster)
plot(iris_1[c("Sepal.Length", "Sepal.Width")],
   col = kmeans.re$cluster,
   main = "K-means with 3 clusters")
kmeans.re$centers


kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")]
points(kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")],
   col = 1:3, pch = 8, cex = 3)
y_kmeans <- kmeans.re$cluster
clusplot(iris_1[, c("Sepal.Length", "Sepal.Width")],
        y_kmeans,
        lines = 0,
        shade = TRUE,
        color = TRUE,
        labels = 2,
        plotchar = FALSE,
        span = TRUE,
        main = paste("Cluster iris"),
        xlab = 'Sepal.Length',
        ylab = 'Sepal.Width')
```

Short Questions

1)Difference between Linear Regression and Multiple Linear Regression

Ans)

Linear Regression	Multiple Linear Regression
A technique for predicting a continuous target variable using one predictor variable	A technique for predicting a continuous target variable using multiple predictor variables
It models the relationship between one independent variable and one dependent variable	It models the relationship between multiple independent variables and one dependent variable
It is used for simple linear relationship	It is used for complex relationships between variables
It is represented by a straight line equation	It is represented by a multiple variable equation
It is used to analyze the relationship between two variables	It is used to analyze the relationship between multiple variables and the target variable

2)What is the function used in Linear Regression

In Linear Regression, the main function used is the "lm()" function. It stands for "linear model" and is used to fit a linear model to a given dataset. The basic syntax for the "lm()" function is as follows:

Copy code

```
lm(formula, data)
```

Where "formula" is the equation representing the linear relationship between the predictor variable(s) and the response variable, and "data" is the dataset containing the variables.

For example, if you have a dataset called "mydata" with a predictor variable called "x" and a response variable called "y", you can use the following code to fit a linear regression model:

Copy code

```
model <- lm(y ~ x, data = mydata)
```

This creates a linear model object "model" that represents the relationship between "x" and "y" in the "mydata" dataset.

3)Explain lists and vectors in R Programming

- In R programming, a "list" is a type of data structure that can hold a collection of items, which can be of different types (e.g. numbers, strings, other lists, etc.).
- Lists are created using the function `list()`, and items can be added, removed, or accessed using the `[]` operator.
- A "vector" is a type of data structure that can hold a collection of items, but all of the items must be of the same type (e.g. all numbers, all strings, etc.).
- Vectors are created using the `c()` function, and items can be accessed using the `[]` operator.
- Both lists and vectors are important and widely used data structures in R programming. They are often used to store and organize data, and are also used as inputs and outputs for many R functions.

4)What is Debugging in R Programming

Ans)

Debugging in R programming refers to the process of identifying and resolving errors or bugs in the code. It involves locating and correcting the source of the problem in the code to make sure that the program runs correctly and produces the expected results. Debugging tools such as the traceback(), browser(), and debug() functions can be used to help identify the source of the error and step through the code line by line to find the problem.

5)Write in the difference between supervised and unsupervised learning

Supervised Learning	Unsupervised Learning
The learning process is guided by labeled examples	The learning process is not guided by labeled examples
The model is trained on labeled data and learns to predict the output from input	The model is trained on unlabeled data and finds patterns and structure in the data
The goal is to make accurate predictions on new, unseen data	The goal is to find hidden structure or patterns in the data
The input and output variables are known	The input variables are known but the output variables are unknown
Examples: Regression, classification	Examples: Clustering, association rule mining, anomaly detection

6)What is Lazy Learner in KNN algorithm

Ans)

In the K-nearest neighbours (KNN) algorithm, a "lazy learner" is a type of learning algorithm that does not build a model during the training phase. Instead, it simply stores the training data and waits until the testing phase to make predictions. In other words, it delays the generalization process until new data is encountered. When a new data point is encountered, the algorithm compares it to the stored training data and makes a prediction based on the k-nearest neighbours. This is different from "eager learners" that builds a model during the training phase, which is then used to make predictions.

7)Define Time series Analysis in Machine Learning

Ans)

Time series analysis is a technique used to analyze and model time-based data in order to understand and make predictions about trends and patterns over time. It is a subfield of statistics and machine learning that focuses on working with data that is collected over time. Time series data can be found in a wide range of fields, such as finance, economics, engineering, and many others. Time series analysis allows to understand the underlying mechanisms that drive the data, detect patterns, and forecast future values. It uses various statistical methods and models such as ARIMA, Exponential smoothing, and other techniques to extract meaningful insights from time-based data in machine learning.

8)Explain working principle of KNN Algorithm

Ans)

The K-Nearest Neighbours (KNN) algorithm works by identifying the k number of data points in the training set that are closest to a new data point, and then classifying the new data point based on the majority class of those k nearest neighbours. The distance metric used to determine the "closeness" of the data points can be any metric such as Euclidean distance, Manhattan distance or Minkowski distance. The KNN algorithm is a simple yet powerful method for classification and regression problems.