Advanced Database Management System Lab

Subject Code: MCAL13

A Practical Journal Submitted in Fulfillment of

the Degree of

MASTER

In

COMPUTERAPPLICATION

Year 2023-2024

By

Mr. KAMBLE TEJAS GUNAJI ANITA

(Application Id: - 74610)

Semester- 1

Under the Guidance of

Prof. Dr. Sujatha Iyer

Institute of Distance and Open Learning

Vidya Nagari, Kalina, Santacruz East – 400098.

Universityof Mumbai

PCP Center

[Satish Pradhan Dyanasadhana College, Thane]

Institute of Distance and Open Learning,

Vidyanagari, Kalina, Santacruz (E) -400098

CERTIFICATE

This to certify that, **Mr. KAMBLE TEJAS GUNAJI ANITA** appearing **Master in Computer Application (Semester I) Application ID: 74610** has satisfactory completed the prescribed practical of **MCAL13-Advanced Database Management System Lab** as laid down by the University of Mumbai for the academic year 2023-24

Teacher in charge | Examiners | Coordinator
IDOL, MCA
University of Mumbai

Date: -

Place: -

INDEX

PRACTICAL 1

DISTRIBUTED DATABASE

RANGE Partitioning in mysql

Aim: Implementation of Data partitioning through Range.

```
C:\Program Files (x86)\MySQL\MySQL Server 5.1\bin\mysql.exe
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 5.1.28-rc-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use mca;
Database changed
mysql> CREATE TABLE tr1 (id INT, name VARCHAR(50), purchased
    -> DATE)
    -> PARTITION BY RANGE( YEAR(purchased) ) (
    -> PARTITION p0 VALUES LESS THAN (1990),
    -> PARTITION p1 VALUES LESS THAN (1995),
    -> PARTITION p2 VALUES LESS THAN (2000),
    -> PARTITION p3 VALUES LESS THAN (2005),
    -> PARTITION p4 VALUES LESS THAN (2010),
    -> PARTITION p5 VALUES LESS THAN (2015)
    -> );
Query OK, 0 rows affected (0.30 sec)
```

```
mysql>
mysql>  INSERT INTO tr1 VALUES
    -> (1, 'desk organiser', '2003-10-15'),
    -> (2, 'alarm clock', '1997-11-05'),
    -> (3, 'chair', '2009-03-10'),
    -> (4, 'bookcase', '1989-01-10'),
    -> (5, 'exercise bike', '2014-05-09'),
    -> (6, 'sofa', '1987-06-05'),
    -> (7, 'espresso maker', '2011-11-22'),
    -> (8, 'aquarium', '1992-08-04'),
    -> (9, 'study desk', '2006-09-16'),
    -> (10, 'lava lamp', '1998-12-25');
Query OK, 10 rows affected (0.05 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> select * from tr1;
+------+---------------+------------+
| id   | name          | purchased  |
+------+---------------+------------+
|    4 | bookcase      | 1989-01-10 |
|    6 | sofa          | 1987-06-05 |
|    8 | aquarium      | 1992-08-04 |
|    2 | alarm clock   | 1997-11-05 |
|   10 | lava lamp     | 1998-12-25 |
|    1 | desk organiser | 2003-10-15 |
|    3 | chair         | 2009-03-10 |
|    9 | study desk    | 2006-09-16 |
|    5 | exercise bike | 2014-05-09 |
|    7 | espresso maker | 2011-11-22 |
+------+---------------+------------+
10 rows in set (0.02 sec)

mysql>
```

```
mysql> SELECT PARTITION_NAME, TABLE_ROWS FROM
    -> INFORMATION_SCHEMA.PARTITIONS WHERE
    -> TABLE_NAME='tr1';
+----------------+------------+
| PARTITION_NAME | TABLE_ROWS |
+----------------+------------+
| p0             |          2 |
| p1             |          1 |
| p2             |          2 |
| p3             |          1 |
| p4             |          2 |
| p5             |          2 |
+----------------+------------+
6 rows in set (0.27 sec)

mysql>
```

PRACTICAL 2

ANALYTICAL QUERIES

Aim: Implementation of Analytical queries like Roll_UP, CUBE, First, Last, Rank AND Dense Rank.

```
SQL> CREATE TABLE emp(
  2  empno NUMBER(4) CONSTRAINT pk_emp PRIMARY KEY,
  3  ename VARCHAR2(10),
  4  job VARCHAR2(10),
  5  mgr NUMBER(4),
  6  hiredate DATE,
  7  sal NUMBER(7,2),
  8  comm NUMBER(7,2),
  9  deptno NUMBER(2));

Table created.

SQL> INSERT INTO emp VALUES(1,'Hema','Developer',2,'22-May-2020',25000,2000,4)
  2  ;

1 row created.

SQL> INSERT INTO emp VALUES(2,'Ram','Developer',2,'20-April-2019',45000,2300,4);

1 row created.

SQL> INSERT INTO emp VALUES(3,'Vrudhi','Tester',4,'20-April-2019',30000,8000,5);

1 row created.
```

```
SQL> INSERT INTO emp VALUES(4,'Rahul','Tester',4,'5-November-2018',50000,8000,5);

1 row created.
```

```
SQL> SELECT * FROM emp;

     EMPNO ENAME      JOB               MGR HIREDATE          SAL       COMM
---------- ---------- ---------- ---------- --------- ---------- ----------
    DEPTNO
----------
         1 Hema       Developer           2 22-MAY-20      25000       2000
         4

         2 Ram        Developer           2 20-APR-19      45000       2300
         4

         3 Vrudhi     Tester              4 20-APR-19      30000       8000
         5


     EMPNO ENAME      JOB               MGR HIREDATE          SAL       COMM
---------- ---------- ---------- ---------- --------- ---------- ----------
    DEPTNO
----------
         4 Rahul      Tester              4 05-NOV-18      50000       8000
         5
```

```
SQL> SELECT empno,sal,sum(sal) as Totalsal from emp group by rollup(empno,sal);

     EMPNO        SAL   TOTALSAL
---------- ---------- ----------
         1      25000      25000
         1                 25000
         2      45000      45000
         2                 45000
         3      30000      30000
         3                 30000
         4      50000      50000
         4                 50000
                          150000

9 rows selected.

SQL>
```

```
SQL> SELECT empno,sal,sum(sal) as Totalsal from emp group by cube(empno,sal);

     EMPNO        SAL   TOTALSAL
---------- ---------- ----------
                          150000
                30000      30000
                50000      50000
                25000      25000
                45000      45000
         1                 25000
         1      25000      25000
         2                 45000
         2      45000      45000
         3                 30000
         3      30000      30000

     EMPNO        SAL   TOTALSAL
---------- ---------- ----------
         4                 50000
         4      50000      50000

13 rows selected.
```

RANK

```
SQL> SELECT empno,deptno,sal,RANK() OVER (PARTITION BY deptno ORDER BY sal) AS myrank FROM emp;

     EMPNO      DEPTNO         SAL      MYRANK
---------- ---------- ---------- ----------
         1          4      25000          1
         2          4      45000          2
         3          5      30000          1
         4          5      50000          2
```

```
SQL> SELECT empno,deptno,sal, DENSE_RANK() OVER (Partition By deptno ORDER By sal) as myrank FROM emp;

     EMPNO      DEPTNO         SAL      MYRANK
---------- ---------- ---------- ----------
         1          4      25000          1
         2          4      45000          2
         3          5      30000          1
         4          5      50000          2
```

DENSE_RANK

```
SQL> SELECT * FROM (SELECT empno,deptno,sal, DENSE_RANK() OVER (Partition By deptno ORDER By sal DESC) as myrank FROM emp)WHERE myrank<=2;

     EMPNO      DEPTNO         SAL      MYRANK
---------- ---------- ---------- ----------
         2          4      45000          1
         1          4      25000          2
         4          5      50000          1
         3          5      30000          2

SQL>
```

FIRST AND LAST

```
SQL> SELECT empno,deptno,sal, MIN(sal) KEEP (DENSE_RANK FIRST ORDER By sal) OVER (Partition By deptno) as lowest,MAX(sal) KEEP (DENSE_RANK LAST ORDER By sal) OVER (Part
ition By deptno) As highest FROM emp ORDER By deptno,sal;

    EMPNO    DEPTNO       SAL    LOWEST   HIGHEST
--------- --------- --------- --------- ---------
        1         4     25000     25000     45000
        2         4     45000     25000     45000
        3         5     30000     30000     50000
        4         5     50000     30000     50000
```

LAG

```
SQL> SELECT deptno,empno,ename,job,sal, LAG(sal,1,0) OVER (PARTITIOn By deptno ORDER BY sal) As sal_prev FROM emp;

    DEPTNO      EMPNO ENAME      JOB               SAL   SAL_PREV
---------- ---------- ---------- --------- ---------- ----------
         4          1 Hema       Developer      25000          0
         4          2 Ram        Developer      45000      25000
         5          3 Vrudhi     Tester         30000          0
         5          4 Rahul      Tester         50000      30000
```

LEAD

```
SQL> SELECT empno,ename,job,sal, LEAD(sal,1,0) OVER (ORDER By sal) As sal_next, LEAD(sal,1,0) OVER (ORDER By sal)-sal As sal_diff FROM emp;

     EMPNO ENAME      JOB              SAL   SAL_NEXT   SAL_DIFF
---------- ---------- ---------- ---------- ---------- ----------
         1 Hema       Developer       25000      30000       5000
         3 Vrudhi     Tester          30000      45000      15000
         2 Ram        Developer       45000      50000       5000
         4 Rahul      Tester          50000          0     -50000
```

```
SQL> SELECT deptno,empno,ename,job,sal, LEAD(sal,1,0) OVER (PARTITION By deptno ORDER BY sal) As sal_next FROM emp;

    DEPTNO      EMPNO ENAME      JOB              SAL   SAL_NEXT
---------- ---------- ---------- ---------- ---------- ----------
         4          1 Hema       Developer       25000      45000
         4          2 Ram        Developer       45000          0
         5          3 Vrudhi     Tester          30000      50000
         5          4 Rahul      Tester          50000          0

SQL>
```

PRACTICAL 3

Aim: Implementation of Abstract Data Type & Reference

Customer_reltab

The Customer_reltab table has the following definition:

```
CREATE TABLE Customer_reltab ( CustNo NUMBER
                    NOT NULL,
 CustName          VARCHAR2(200) NOT NULL,
 Street          VARCHAR2(200) NOT NULL,
 City            VARCHAR2(200) NOT NULL,
 State           CHAR(2) NOT NULL,
 Zip             VARCHAR2(20) NOT NULL,
 Phone1            VARCHAR2(20),
 Phone2            VARCHAR2(20),
 Phone3            VARCHAR2(20),
 PRIMARY KEY (CustNo));
```

```
SQL> desc customer_reltab;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CUSTNO                                    NOT NULL NUMBER
 CUSTNAME                                  NOT NULL VARCHAR2(200)
 STREET                                    NOT NULL VARCHAR2(200)
 CITY                                      NOT NULL VARCHAR2(200)
 STATE                                     NOT NULL CHAR(2)
 ZIP                                       NOT NULL VARCHAR2(20)
 PHONE1                                             VARCHAR2(20)
 PHONE2                                             VARCHAR2(20)
 PHONE3                                             VARCHAR2(20)
```

PurchaseOrder_reltab

The PurchaseOrder_reltab table has the following definition:

```
CREATE TABLE PurchaseOrder_reltab (
  PONo       NUMBER, /* purchase order no */
  Custno      NUMBER references Customer_reltab, /* Foreign KEY referencing
                                  customer */
  OrderDate DATE, /* date of order */
  ShipDate DATE, /* date to be shipped */
  ToStreet VARCHAR2(200), /* shipto address */
  ToCity      VARCHAR2(200),
  ToState     CHAR(2),
  ToZip       VARCHAR2(20),
  PRIMARY KEY(PONo));
```

```
SQL> CREATE TABLE PurchaseOrder_reltab (
  2     PONo          NUMBER,
  3     Custno        NUMBER references Customer_reltab,
  4     OrderDate     DATE,
  5     ShipDate      DATE,
  6     ToStreet      VARCHAR2(200),
  7     ToCity        VARCHAR2(200),
  8     ToState       CHAR(2),
  9     ToZip         VARCHAR2(20),
 10     PRIMARY KEY(PONo));

Table created.
```

Stock_reltab

The Stock_reltab table has the following definition:

CREATE TABLE Stock_reltab (
StockNo NUMBER PRIMARY KEY,
Price NUMBER,
TaxRate NUMBER);

```
SQL> CREATE TABLE Stock_reltab (
  2    StockNo       NUMBER PRIMARY KEY,
  3    Price         NUMBER,
  4    TaxRate       NUMBER);

Table created.
```

LineItems_reltab

The LineItems_reltab table has the following definition:

CREATE TABLE LineItems_reltab (

LineItemNo NUMBER,
PONo NUMBER REFERENCES PurchaseOrder_reltab,
StockNo NUMBER REFERENCES Stock_reltab, Quantity

NUMBER,

Discount NUMBER,
PRIMARY KEY (PONo, LineItemNo));

```
SQL> CREATE TABLE LineItems_reltab (
  2    LineItemNo            NUMBER,
  3    PONo                  NUMBER REFERENCES PurchaseOrder_reltab,
  4    StockNo               NUMBER REFERENCES Stock_reltab,
  5    Quantity              NUMBER,
  6    Discount              NUMBER,
  7    PRIMARY KEY (PONo, LineItemNo));

Table created.
```

Inserting Values Under the Relational Model

In our application, statements like these insert data into the tables:INSERT INTO Stock_reltab VALUES(1004, 6750.00, 2);

INSERT INTO Stock_reltab VALUES(1011, 4500.23, 2); INSERT INTO Stock_reltab VALUES(1534, 2234.00, 2);

INSERT INTO Stock_reltab VALUES(1535, 3456.23, 2);

```
INSERT INTO Customer_reltab
  VALUES (1, 'Jean Nance', '2 Avocet Drive',
        'Redwood Shores', 'CA', '95054',
        '415-555-1212', NULL, NULL);
INSERT INTO Customer_reltab
  VALUES (2, 'John Nike', '323 College Drive','Edison',
        'NJ', '08820',
        '609-555-1212', '201-555-1212', NULL);
INSERT INTO PurchaseOrder_reltab
  VALUES (1001, 1, SYSDATE, '10-MAY-1997',NULL,
         NULL, NULL, NULL);
INSERT INTO PurchaseOrder_reltab
  VALUES (2001, 2, SYSDATE, '20-MAY-1997',
        '55 Madison Ave', 'Madison', 'WI', '53715');
```

INSERT INTO LineItems_reltab VALUES(01, 1001, 1534, 12, 0);

INSERT INTO LineItems_reltab VALUES(02, 1001, 1535, 10, 10);

INSERT INTO LineItems_reltab VALUES(01, 2001, 1004, 1, 0); INSERT INTO LineItems_reltab VALUES(02, 2001, 1011, 2, 1);

Querying Data Under the Relational Model The application can execute queries like these:

```
SELECT C.CustNo, C.CustName, C.Street, C.City, C.State,
        C.Zip, C.phone1, C.phone2, C.phone3,
        P.PONo, P.OrderDate,
        L.StockNo, L.LineItemNo, L.Quantity, L.Discount
 FROM Customer_reltab C,
        PurchaseOrder_reltab P,
        LineItems_reltab L
 WHERE C.CustNo = P.CustNo
```

```
 AND P.PONo = L.PONo AND
        P.PONo = 1001;
```

Get the Total Value of Purchase Orders

```
SELECT P.PONo,
    SUM(S.Price * L.Quantity) FROM
PurchaseOrder_reltab P,
    LineItems_reltab L,
    Stock_reltab S
WHERE P.PONo= L.PONo
 AND L.StockNo = S.StockNo
GROUP BYP.PONo;
```

Get the Purchase Order and Line Item Data for Stock Item 1004

```
SELECT P.PONo, P.CustNo,
    L.StockNo, L.LineItemNo, L.Quantity, L.Discount FROM
PurchaseOrder_reltab P,
    LineItems_reltab L
WHERE P.PONo = L.PONo
AND L.StockNo = 1004;
```

Updating Data Under the Relational Model. The applicationcan execute statements like these to update the data:

```
UPDATE LineItems_reltab SET Quantity =
        20 WHERE
        PONo = 1001
 AND StockNo = 1534;
```

Deleting Data Under the Relational Model

```
DELETE
  FROM LineItems_reltab
  WHERE PONo = 1001;

DELETE
  FROM PurchaseOrder_reltab
  WHERE PONo = 1001;
```

PRACTICAL 4

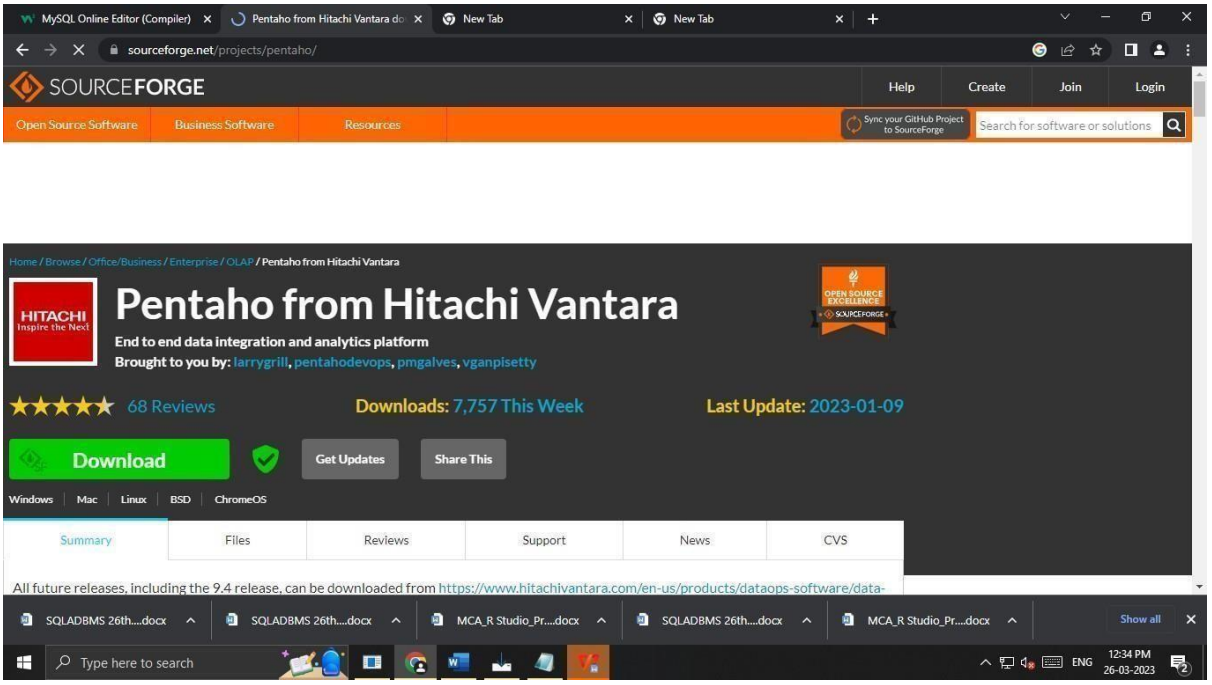
Aim: To study ETL process

* Installation steps for Pentaho Data Integration Software

Step 1: Download Pentaho Data Integration Software. The first thing we need is the Pentaho Data Integration software that we'll be working with

You can download the set up file from link

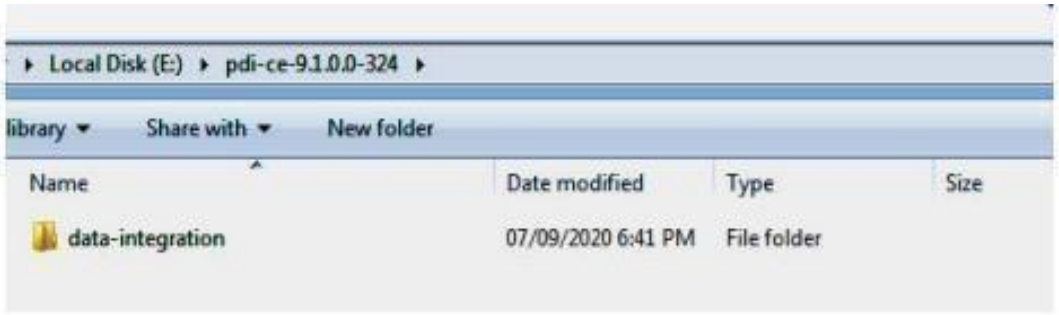
https://sourceforge.net/projects/pentaho/.



Press the "Download" button.

It will start downloading zip file on your computer. Once the downloading is finished, extract the files into a folder you want to.

Your folder should look something like this:



Step 2: Install the Java Dependencies, if Required.

To run Pentaho Data Integration, Java Runtime Environment and Java Development Kit are required. To check if you already have these installed, go to this path in your file explorer:

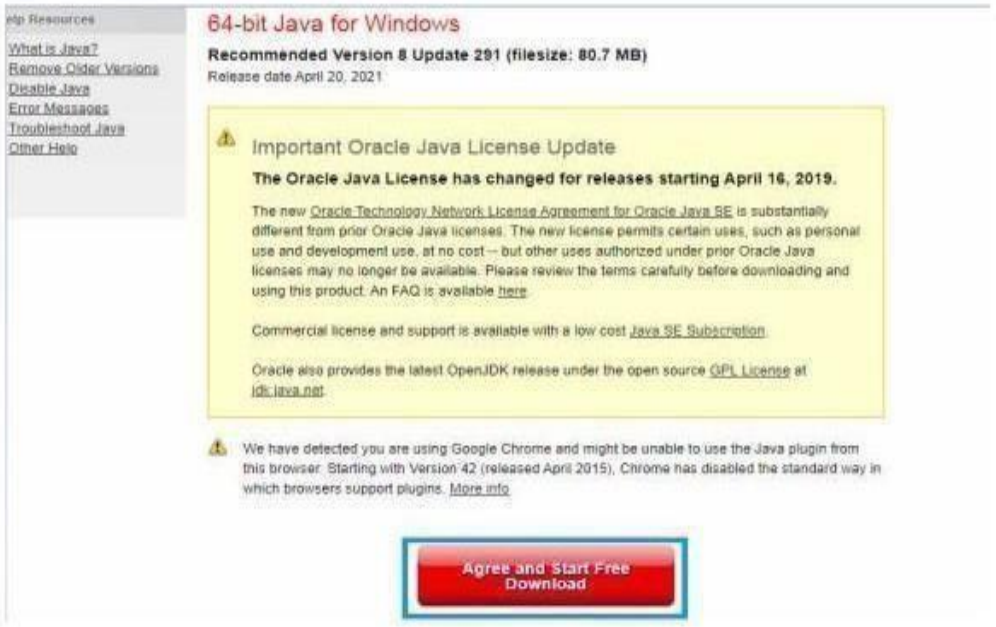
C:\Program Files\Java

Or: C:\Program Files (x86)\Java If this folder exists

and you see folders that look like:

Then you have the required files. If this folder doesn't exist or you don't see one or both of these folders, then you need to download JRE and/or JDK. To download JRE, go to this link https://java.com/en/download/ and press "Download."

Your page should look like this:



The installation window will look something like this:

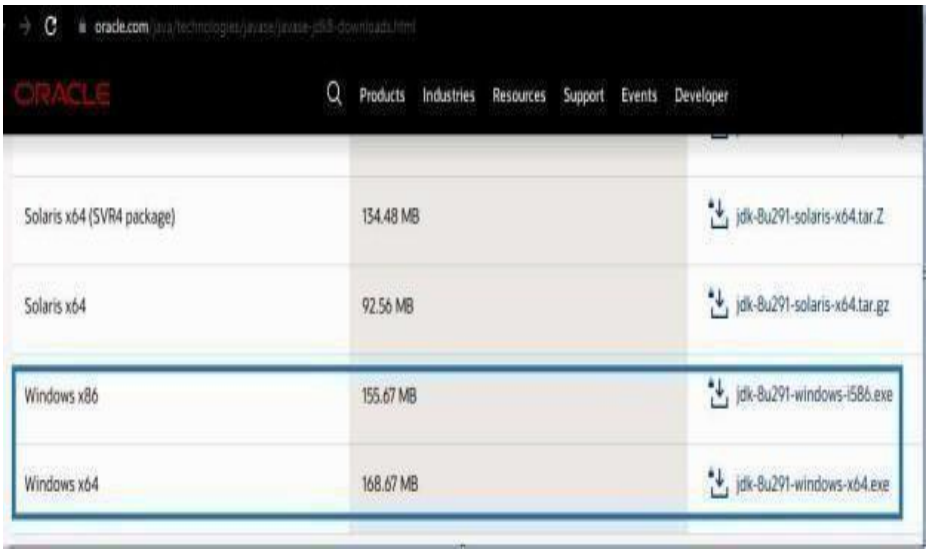
Follow the instructions until finished.

Next, download the JDK from this link
https://www.oracle.com/java/technologies/javase/javase-jdk8- downloads.html.

Please note that there have been substantial changes to the Oracle JDK licensing agreement. Details are available at Oracle Technology Network License Agreement for Oracle Java SE.

There will be a list of different operating systems to choose from. Scroll until you find Windows.

If you're unsure about which version (x64 or x86) your Windows is, select x86.

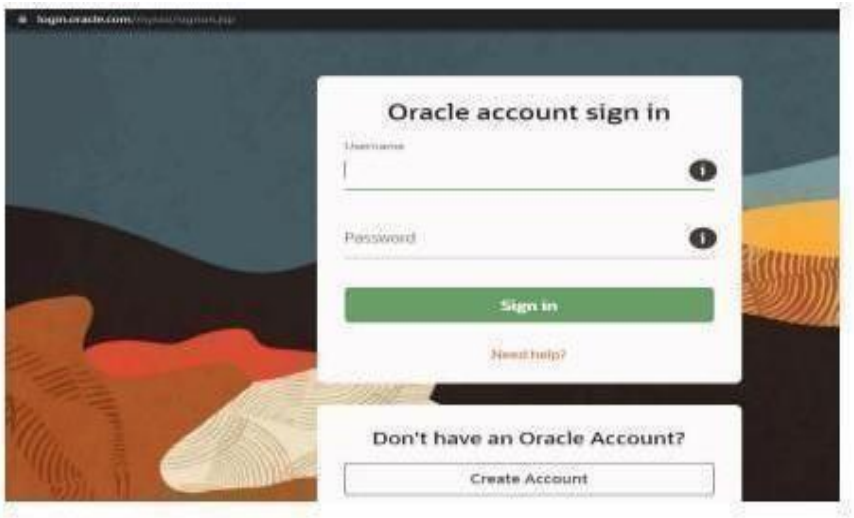


It will open following window

Press “Download”.

If you’re not logged in to Oracle, then you will be prompted to log in.

If you don’t have an Oracle account, you need to create one in order to download the JDK.

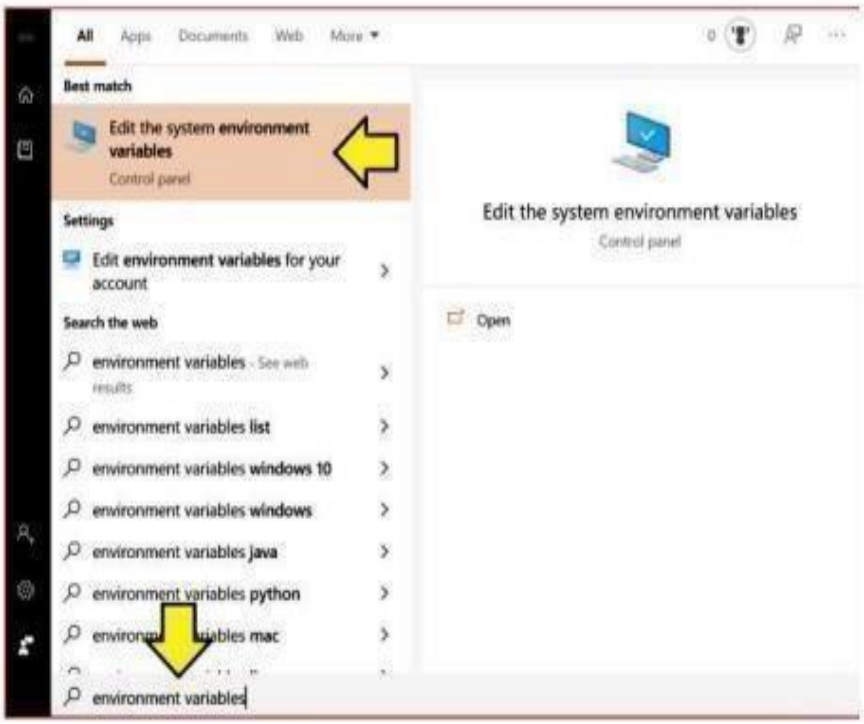


The installation setup will look like this:

Graphics:

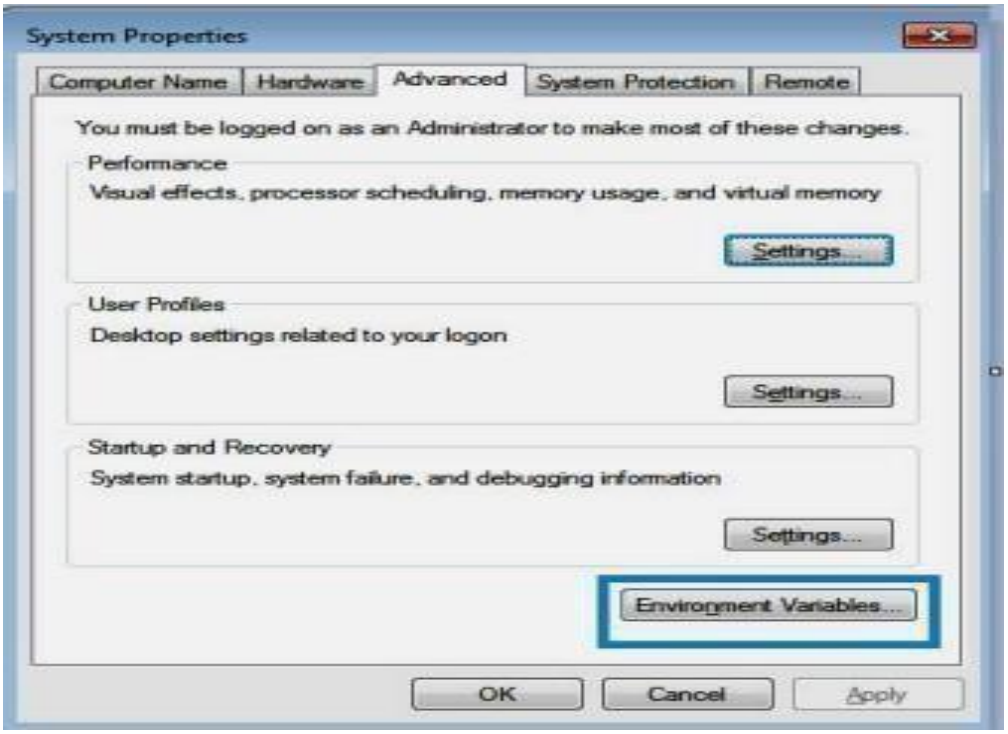
Hitachi Video Management Platform (VMP) has been designed from the ground up to meet the challenges of data storage and processing that new video systems present.

Step 3: Set Up the Environment Variables There are three environment variables that need to be set up. To open the environment variables menu type in “environment variables” in the Windows search bar like this:

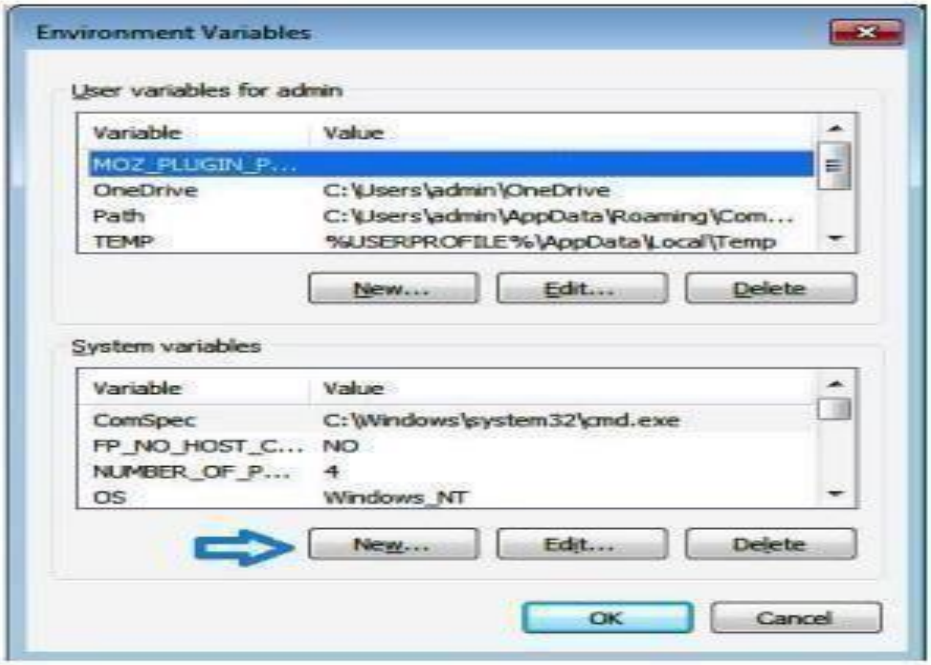


Click the “Edit the system environment variables” option.

That will open the “System Properties” window. Under Advanced tab …Click the “Environment Variables.” button at the bottom.



That will open a window that looks like this:

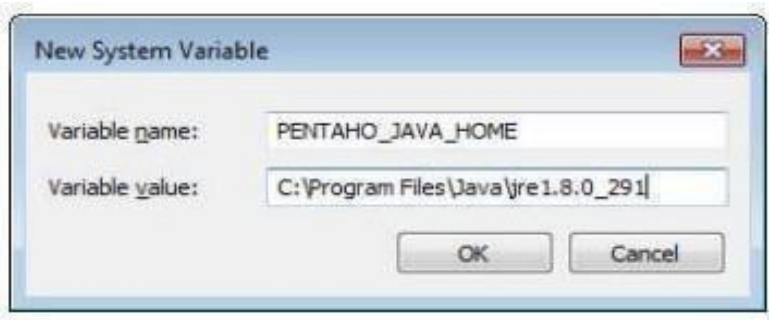


We need to add three new System variables.

Click the “New…” button under “System variables” and enter the following:

Make sure your variable value file path is the same one on your computer.

Press “OK” and then enter two more.

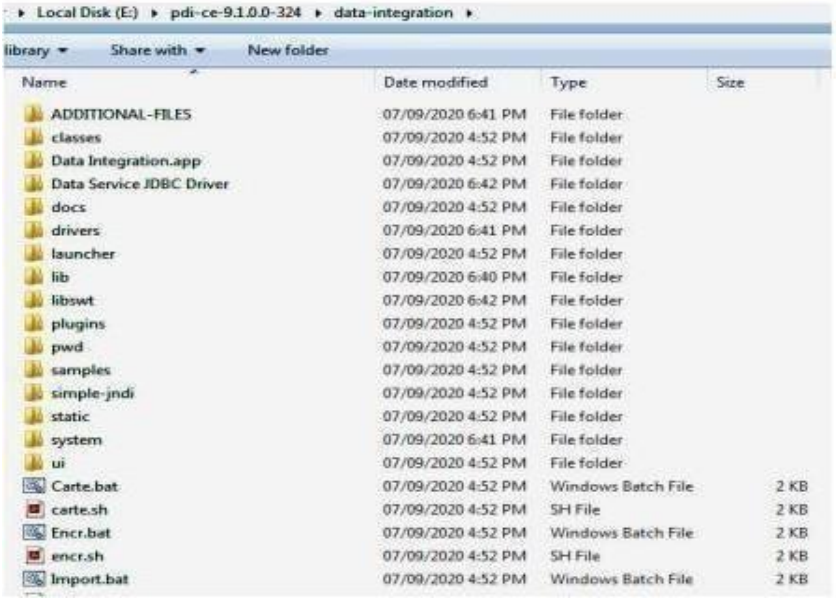


Press ‘’OK’’.

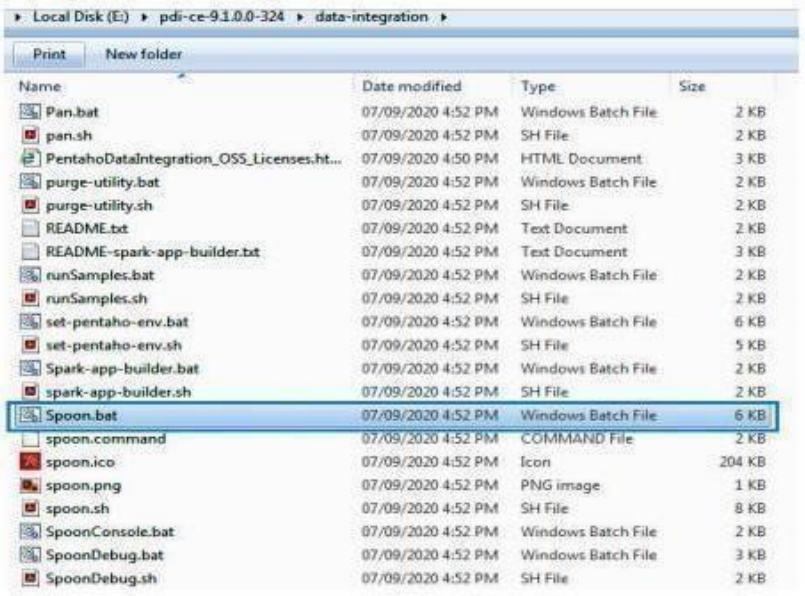
Press ‘’OK’’ and close all the previous windows by pressing “OK.”

Step 4: Open the Pentaho Data Integration App Now that Java is installed successfully and the environment variables are also set, we can start running the Pentaho Data Integrationapp.

The data integration folder that you downloaded earlier will looklike this:



The file that runs the app is called “Spoon.bat”.



Double click this file to open the Pentaho Data Integration app.

Now you can start using this app by pressing “New transformation” or “New job.”

PRACTICAL 5

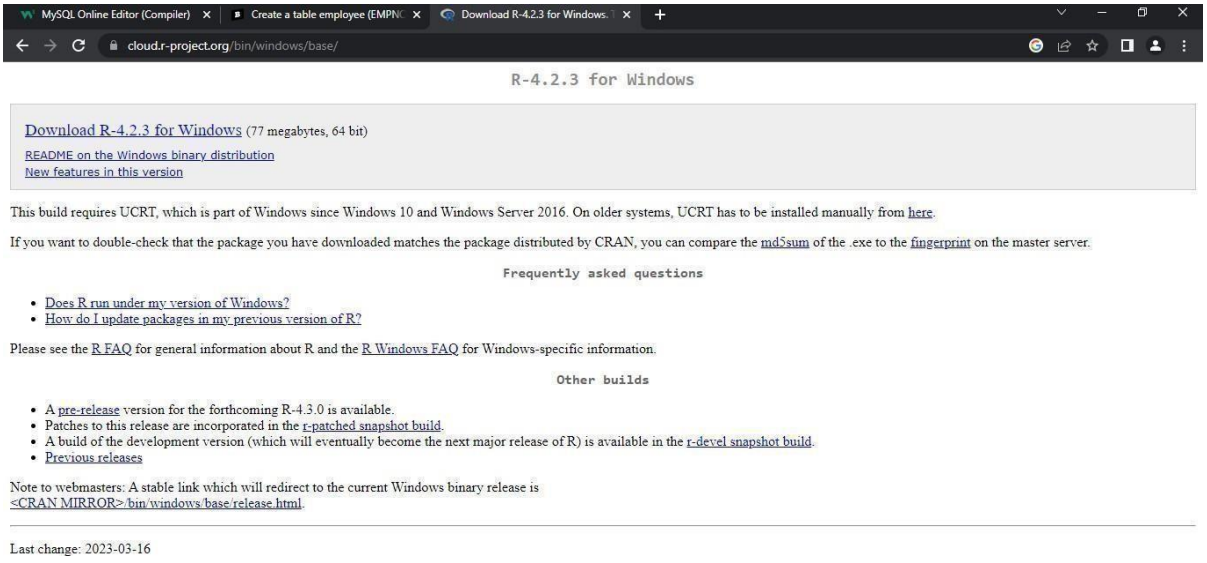
EXPERIMENT 1

Aim: installation of R

* R Installation in Windows

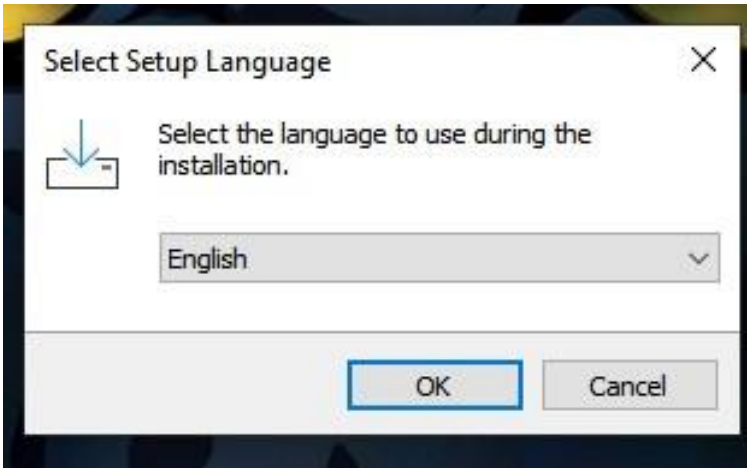
Steps used to install the R in Windows are as follows:

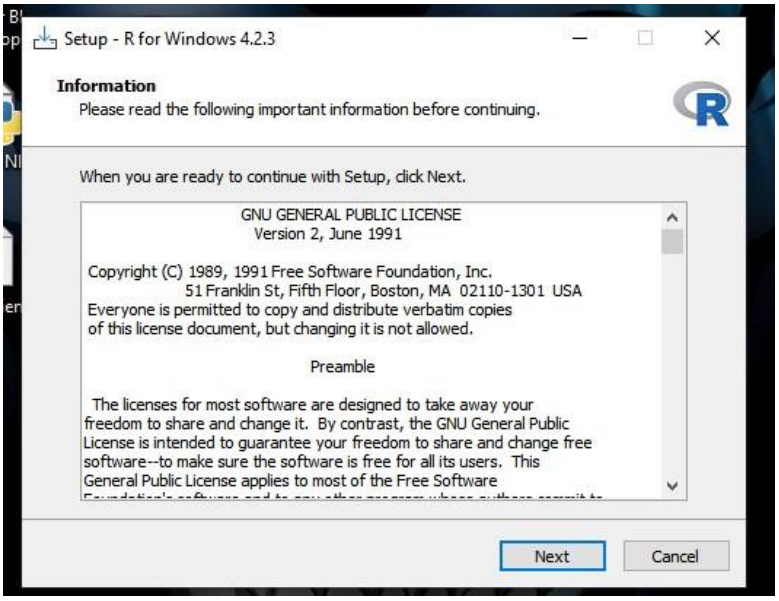
Step 1: First, we have to download the R setup from https://cloud.rproject.org/bin/windows/base/.



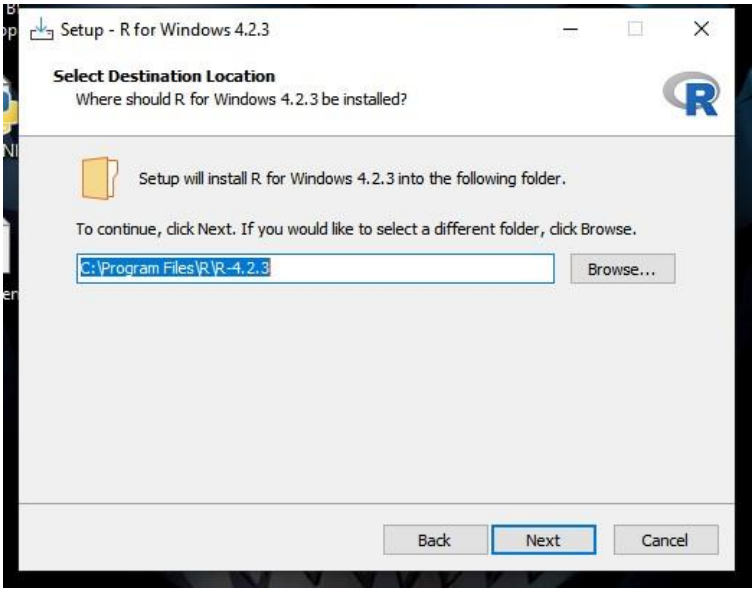
Step 2:

When we click on Download R- 4.1.0 for windows, our downloading will start. Once the downloading is finished, we have to run the setup of R as follows:

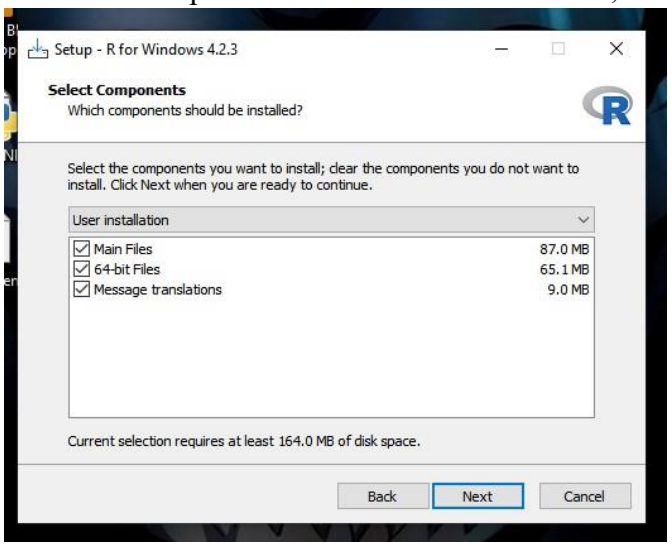




1) Select the path where we want to download the R and click Next

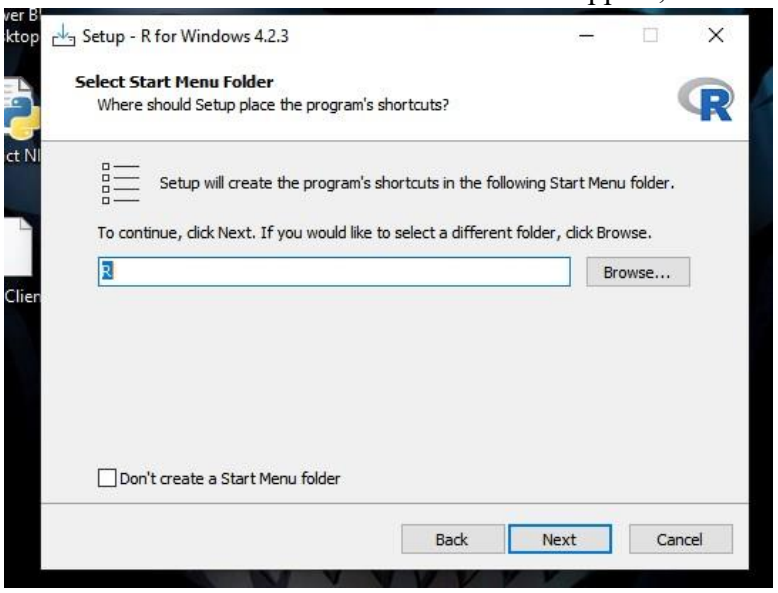


2) Select all components which we want to install, and then click Next.



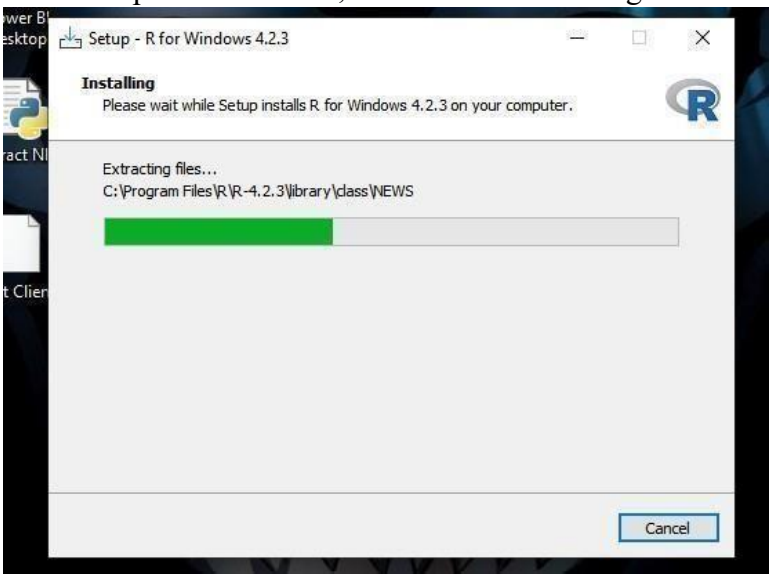
3) Now, we have to select either (customized startup) or (accept the default), and then click Next.

4) Now Select Start Menu Folder window will appear, click Next



Click Next

5) When we proceed to Next, installation of R will get started:



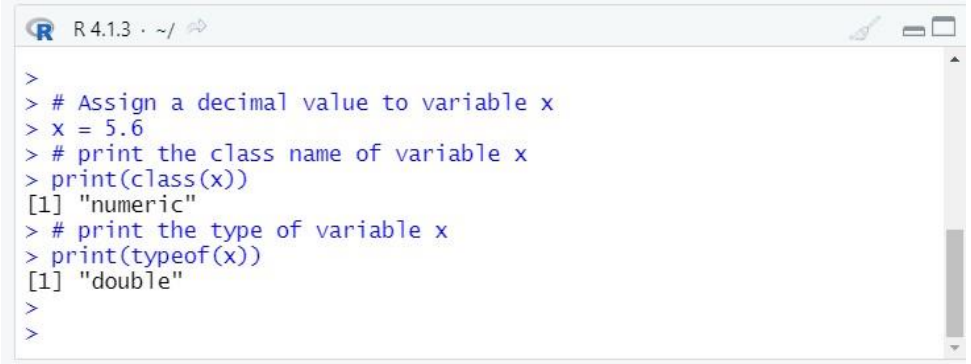
6) Finally, we will click on Finish.

R has been successfully installed.

EXPERIMENT 2

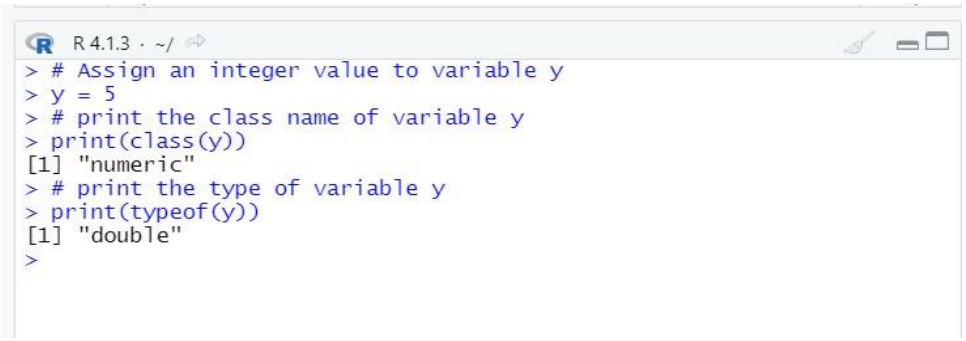
Aim: datatype in R programming

1. **R program to illustrate Numeric data** #
   Assign a decimal value to variable x x
   = 5.6
   # print the class name of variable x print(class(x))
   # print the type of variable x print(typeof(x))



   Output:
   [1] "numeric"
   [1] "double"

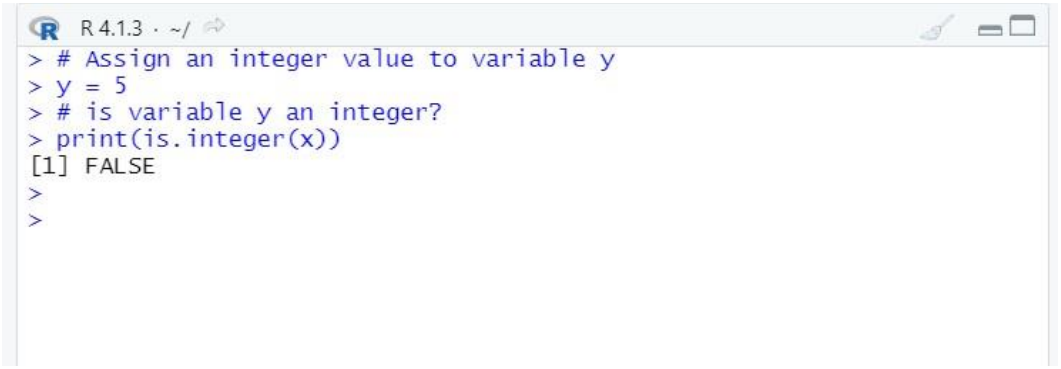
2. **R program to illustrate Numeric datatype**
   # Assign an integer value to variable y
   y = 5
   # print the class name of variable y print(class(y))
   # print the type of variable y print(typeof(y))



   Output:
   [1] "numeric"
   [1] "double"

3. **R program to illustrate Numeric datatype**

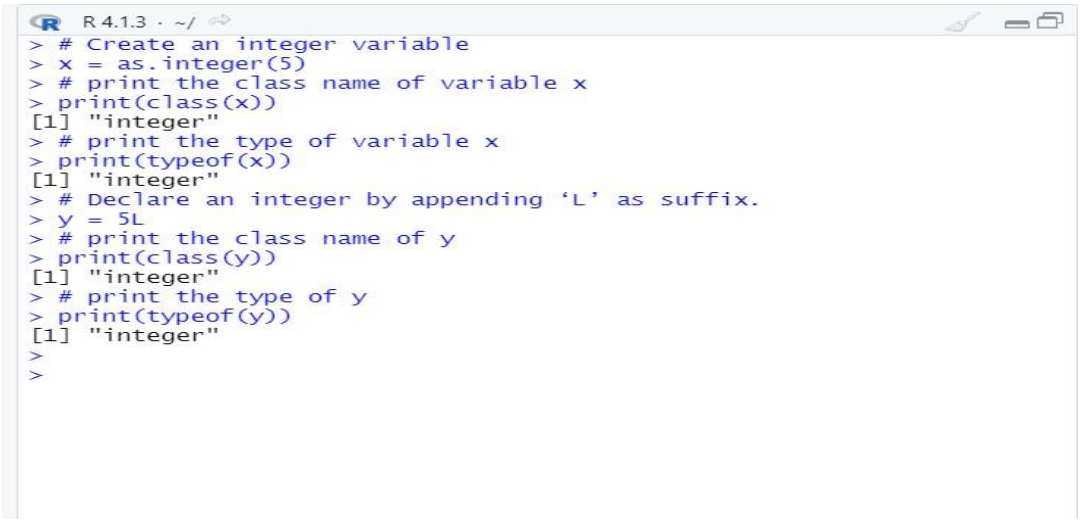
```
# Assign an integer value to variable y
y = 5
# is variable y an integer?
print(is.integer(x))
```



Output:
[1] FALSE

4. **R program to illustrate integer data type** #

```
Create an integer variable x = as.integer(5)
# print the class name of variable x print(class(x))
 # print the type of variable x print(typeof(x))
 # Declare an integer by appending ‘L’ as suffix. y
= 5L
# print the class name of
y
print(class(y)) # print the
type of y print(typeof(y))
```

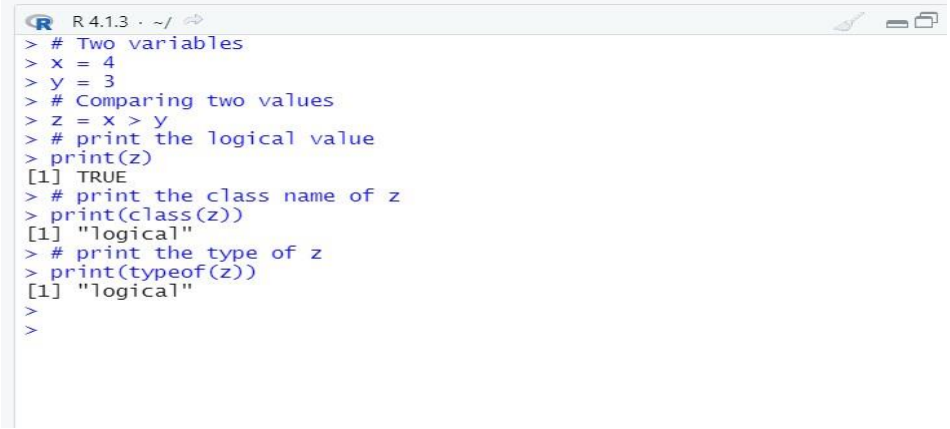


Output:

[1] "integer"
[1] "integer"
[1] "integer"
[1] "integer"

5. R program to illustrate logical data type

```
# Two variables x = 4
y = 3
# Comparing two values
z = x > y # print the
logical value print(z)
# print the class name of z print(class(z))
# print the type of z
print(typeof(z))
```



Output:
[1] TRUE
[1] "logical"
[1] "logical"

6. R program to illustrate complex datatype

```
# Assign a complex value to variable x x = 4
+ 3i
# print the class name of variable x print(class(x))
# print the type of variable x print(typeof(x))
```

```
R 4.1.3 · ~/
> # Assign a complex value to variable x
> x = 4 + 3i
> # print the class name of variable x
> print(class(x))
[1] "complex"
> # print the type of variable x
> print(typeof(x))
[1] "complex"
>
>
```

Output:
[1] "complex"
[1] "complex"

7. R program to illustrate character data type

```
# Assign a character value to char char
 = "Mumbai University"
# print the class name of char
 print(class(char))
# print the type of char
print(typeof(char))
```

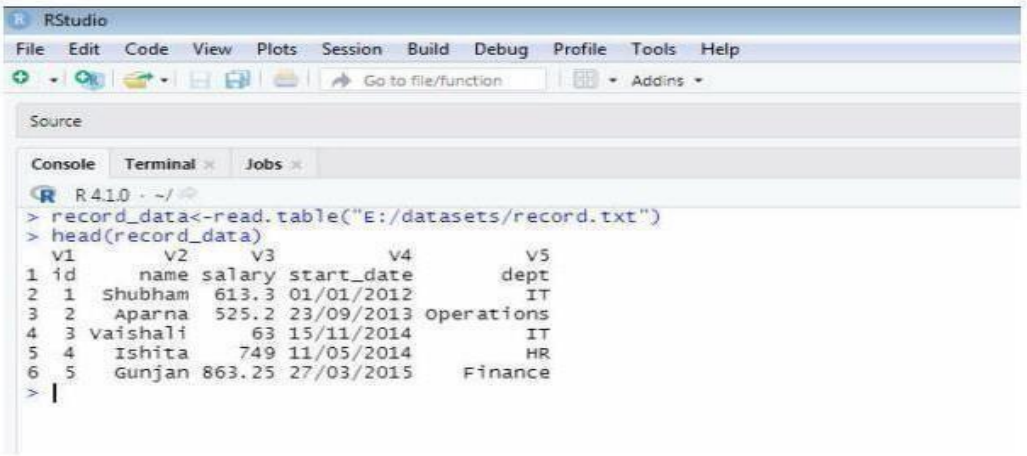
```
R 4.1.3 · ~/
> # Assign a character value to char
> char = "Mumbai University"
> # print the class name of char
> print(class(char))
[1] "character"
> # print the type of char
> print(typeof(char))
[1] "character"
>
>
```

Output:
[1] "character"
[1] "character"

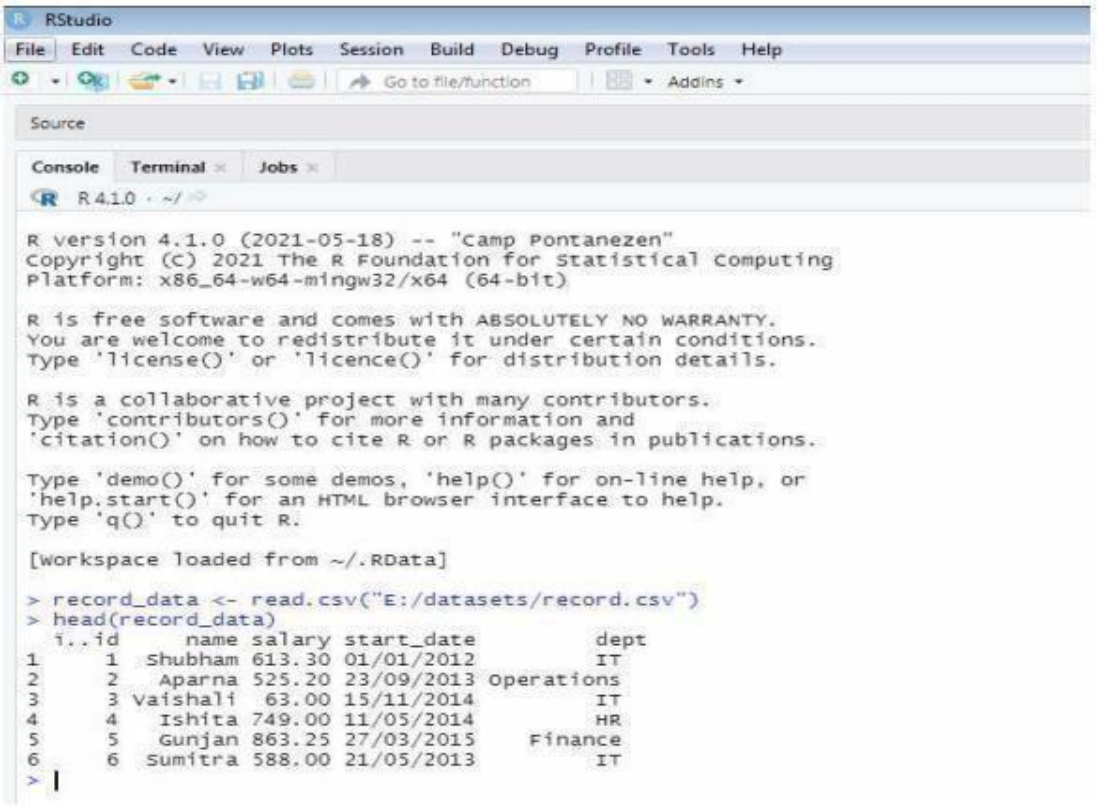
Experiment 3
Aim: Reading and Writing data to and from R.

Reading data files with read.table()
The read.table() function is one of the most common used functions for readingdata into R. It has following arguments.
The function read.table() can be used to read the data frame.
We have kept record.txt and record.csv files under datasets folder inside E: drive.

>record_data<- read.table("E:/datasets/record.txt")
>head(record_data)
#returns first n rows of the data



Similarly, read.csv() function can be used to read data from csv files.
>record_data<- read.csv("E:/datasets/record.csv")
>head(record_data) #returns first n rows of thedata

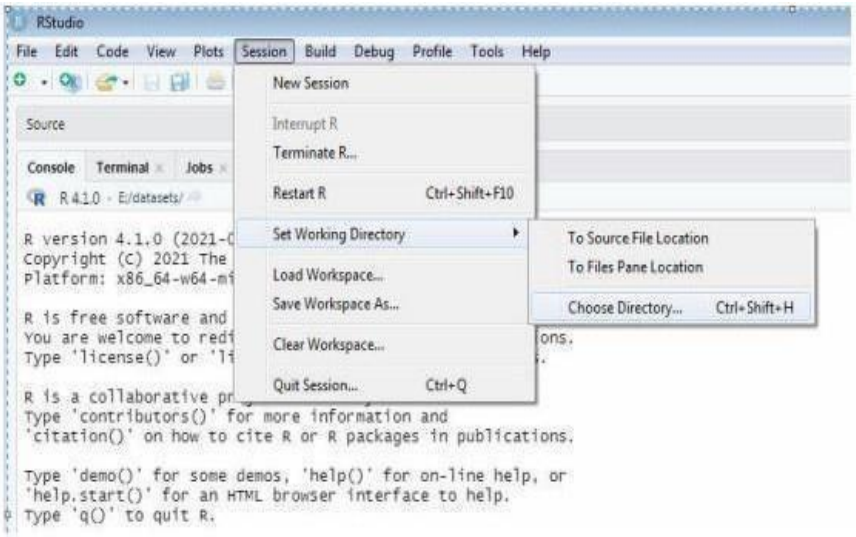


***Writing Data to a File**

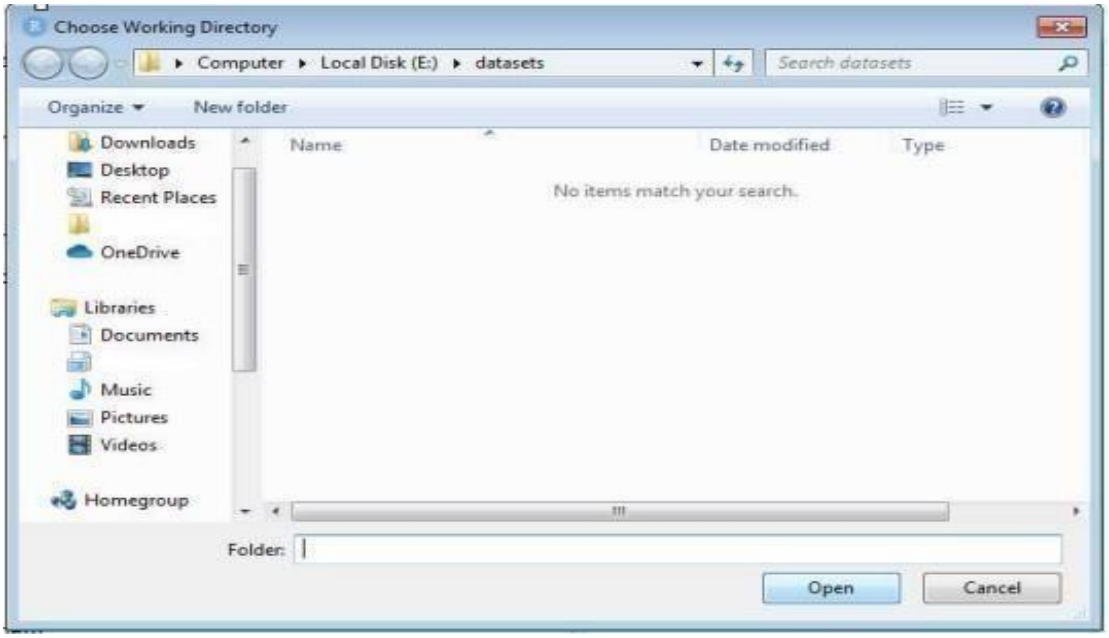
After working with a dataset, we might like to save it for future use. Before we do this, let's first set up a working directory so we know where we can find all our data sets and files later.

Setting up a Directory

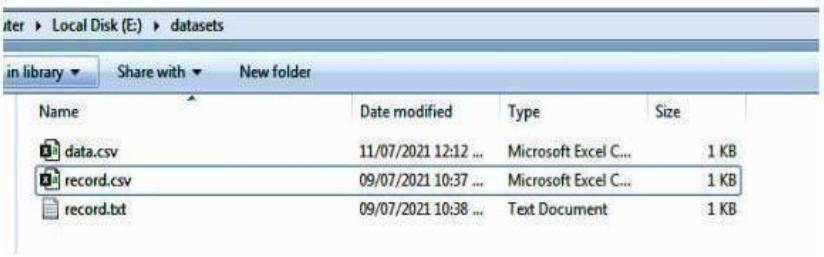
From RStudio, use the menu to change your working directory under Session > Set Working Directory > Choose Directory



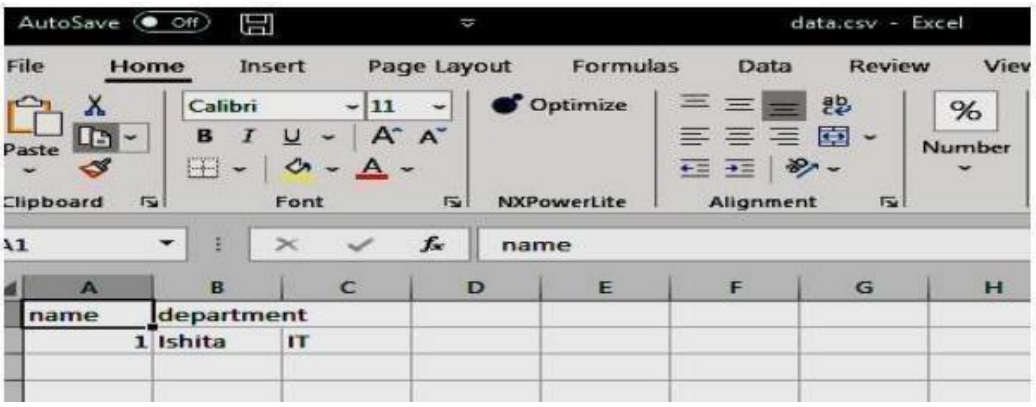
Click Open.



Alternatively, you can use the setwd() function to assign working directory.
> setwd("E:/datasets")
To check your current working directory, type
> getwd()
In R, we can write data easily to a file, using the write.table() command.
x<-data.frame(name ="Ishita", department = "IT") write.table(x, file ="data.csv", sep = ",")



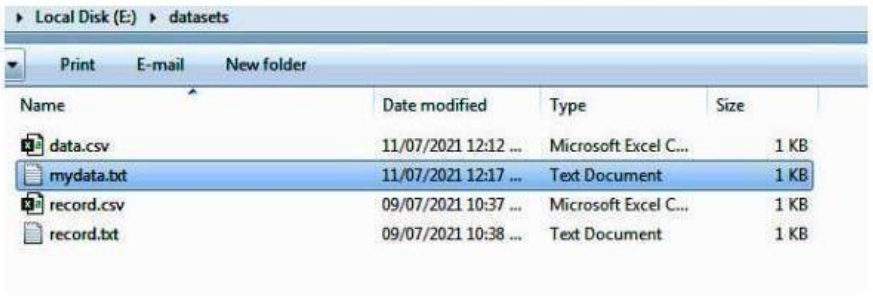
By going to this location E:/datasets,you should see a data.csv file.



y<-data.frame(name ="Ankit", department = "HR")
write.table(y,"E:/datasets/mydata.txt", sep = "\t")

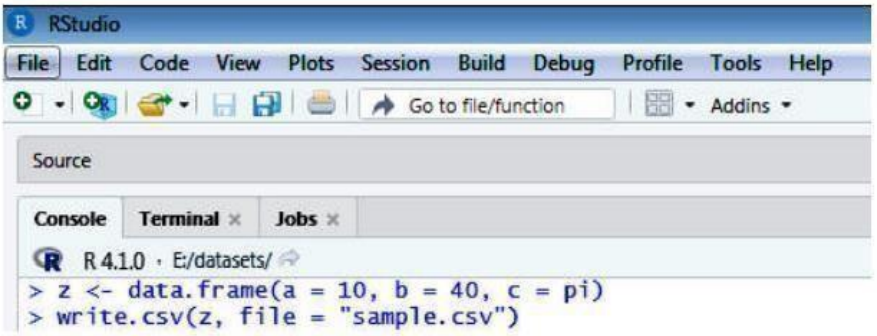
```
> y<-data.frame(name ="Ankit", department = "HR")
> write.table(y, "E:/datasets/mydata.txt", sep = "\t")
>
```

Now, let's check whether R created the file mydata.txt under E:/datasets folder or not.

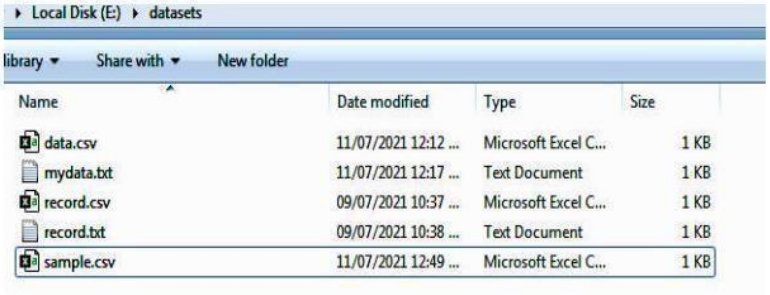


By going to this location E:/datasets, you should see a mydata.txt file.

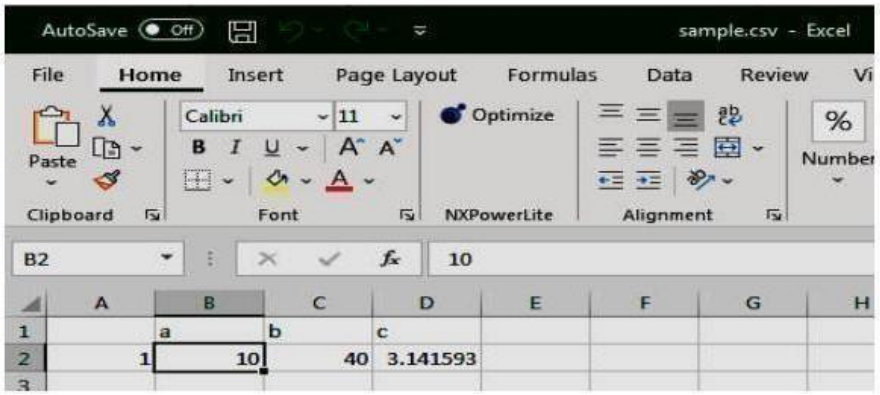
z <- data.frame(a = 10, b = 40, c = pi) write.csv(z, file = "sample.csv")



Now, let's check whether R created the file sample.csv under E:/datasets folder or not.



By going to this location E:/datasets,you should see a sample.csv file.



PRACTICAL 6

Aim: Data preprocessing in R.

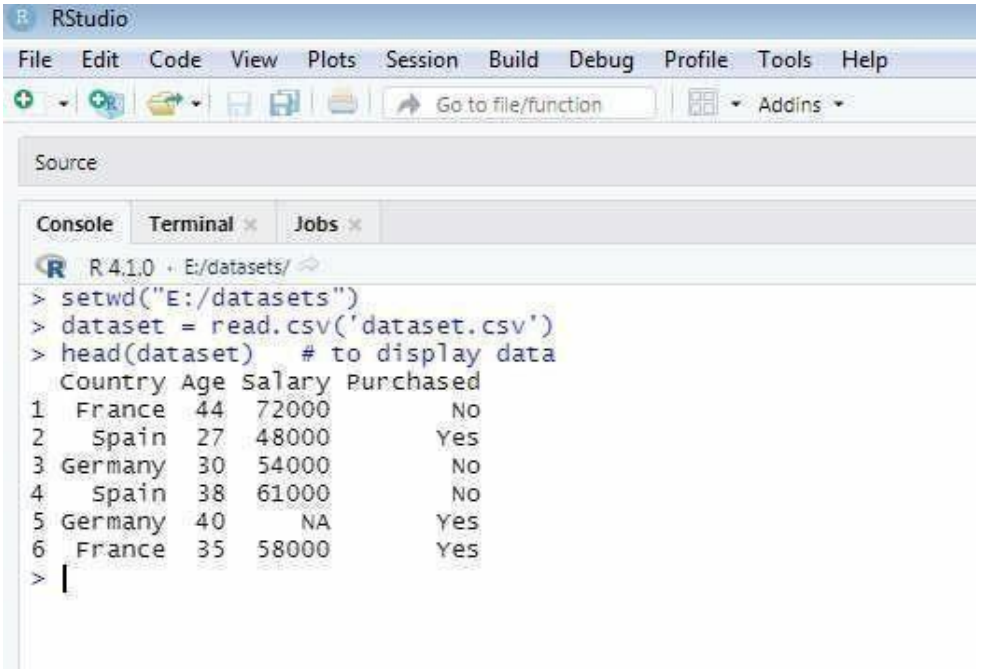
❖ **Data Preprocessing in R**

	A	B	C	D	E	F
1	Country	Age	Salary	Purchased		
2	France	44	72000	No		
3	Spain	27	48000	Yes		
4	Germany	30	54000	No		
5	Spain	38	61000	No		
6	Germany	40		Yes		
7	France	35	58000	Yes		
8	Spain		52000	No		
9	France	48	79000	Yes		
10	Germany	50	83000	No		
11	France	37	67000	Yes		
12						

dataset.csv file

❖ **Importing the Dataset**

Here, first we will change the working directory to E:/datasets (where dataset.csv is stored)



```
> setwd("E:/datasets")
> dataset = read.csv('dataset.csv')
> head(dataset)   # to display data
  Country Age Salary Purchased
1  France  44  72000        No
2   Spain  27  48000       Yes
3 Germany  30  54000        No
4   Spain  38  61000        No
5 Germany  40     NA       Yes
6  France  35  58000       Yes
>
```

To display all 7 rows from csv file

```
> head(dataset,10)
   Country Age Salary Purchased
1   France  44  72000        No
2    Spain  27  48000       Yes
3  Germany  30  54000        No
4    Spain  38  61000        No
5  Germany  40     NA       Yes
6   France  35  58000       Yes
7    Spain  NA  52000        No
8   France  48  79000       Yes
9  Germany  50  83000        No
10  France  37  67000       Yes
> 
```

❖ *Dealing with Missing Values*

```
dataset$Age = ifelse(is.na(dataset$Age),ave(dataset$Age, FUN = function(x)
na.rm =
```

```
dataset$Salary = ifelse(is.na(dataset$Salary), ave(dataset$Salary, FUN =
mean(x, na.rm = 'TRUE')),
```

The above code checks for missing values in the Age and Salary columns and update the missing cells with the column-wise average.

- *dataset$column_header:*

  Selects the column in the dataset specified after $ (Age andSalary).

- *is.na(dataset$column_header):*

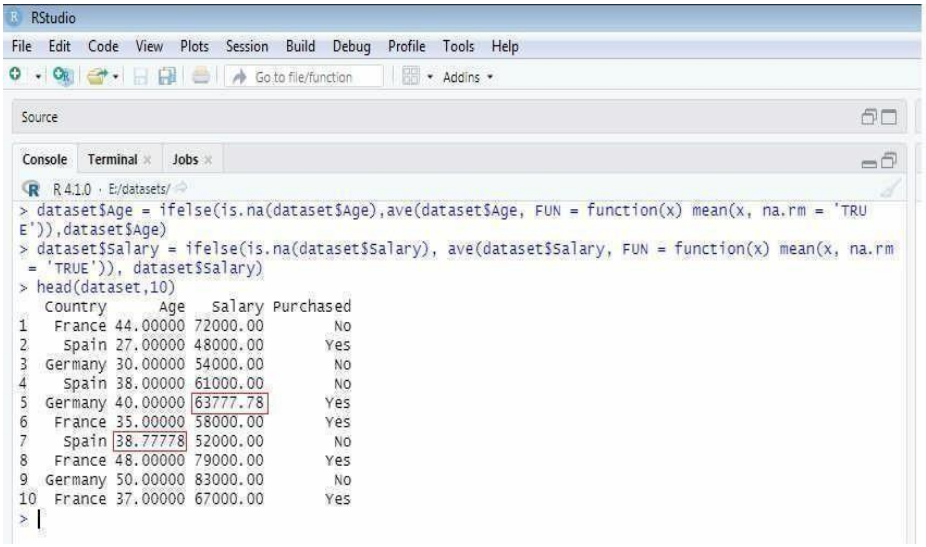
  This method returns true for all the cells in the specified columnwith no values.

- *ave(dataset$column_header, FUN = function(x) mean(x, na.rm* **=**

  'TRUE')):

  This method calculates the average of the column passed asargument.

Output:



Since we don't want decimal places for Age, we will round it up using thefollowing code.

```
dataset$Age =
```

The argument 0 in the round function means no decimal places.

After executing the above code block , the dataset would look like what's shown below :

```
> dataset$Age = as.numeric(format(round(dataset$Age, 0))
> head(dataset,10)
   Country Age   Salary Purchased
1   France  44 72000.00        No
2    Spain  27 48000.00       Yes
3  Germany  30 54000.00        No
4    Spain  38 61000.00        No
5  Germany  40 63777.78       Yes
6   France  35 58000.00       Yes
7    Spain  39 52000.00        No
8   France  48 79000.00       Yes
9  Germany  50 83000.00        No
10  France  37 67000.00       Yes
>
```

Dealing withCategorical Data

❖

Categorical variables represent types of data which may be divided into groups. Examples of categorical variables are race, sex, age group, educational level etc.

In our dataset, we have categorical features 'Purchased'. In R we can use the factor method to convert texts into numerical codes.

```
dataset$Purchased = factor(dataset$Purchased, levels = c('No','Yes'), labels =
```

- **factor(dataset$olumn_header, levels = c(), labels = c()) :**

the factor method converts the categorical features in the specifiedcolumn to factors or numerical codes.

- **levels:**

  The categories in the column passed as a vector. Example c('No','Yes')

- **labels:**

  The numerical codes for the specified categories in the same order.Example c(0,1))

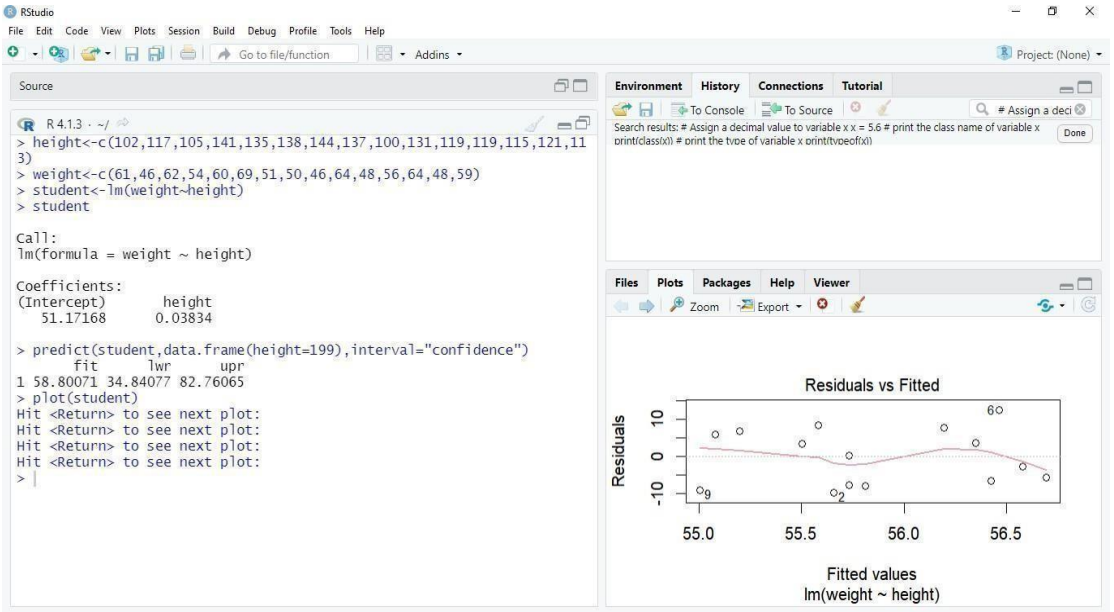
Output:

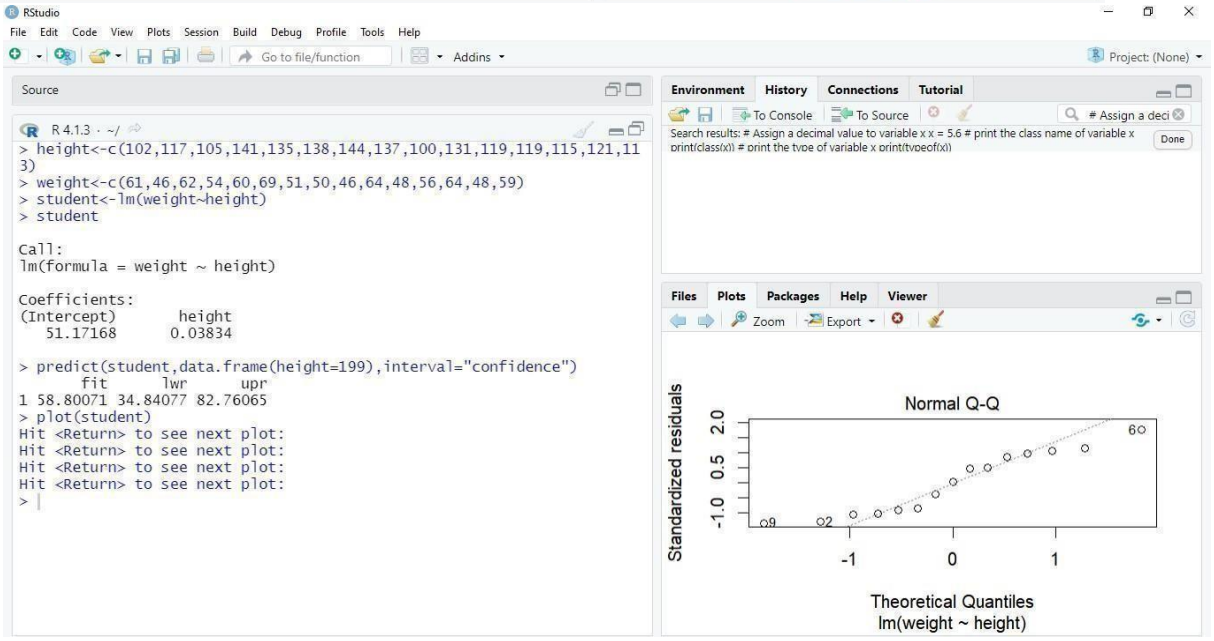
```
> dataset$Purchased = factor(dataset$Purchased, levels = c('No','Yes'),  labels = c(0,1))
> head(dataset,10)
   Country Age   Salary Purchased
1   France  44 72000.00         0
2    Spain  27 48000.00         1
3  Germany  30 54000.00         0
4    Spain  38 61000.00         0
5  Germany  40 63777.78         1
6   France  35 58000.00         1
7    Spain  39 52000.00         0
8   France  48 79000.00         1
9  Germany  50 83000.00         0
10  France  37 67000.00         1
```

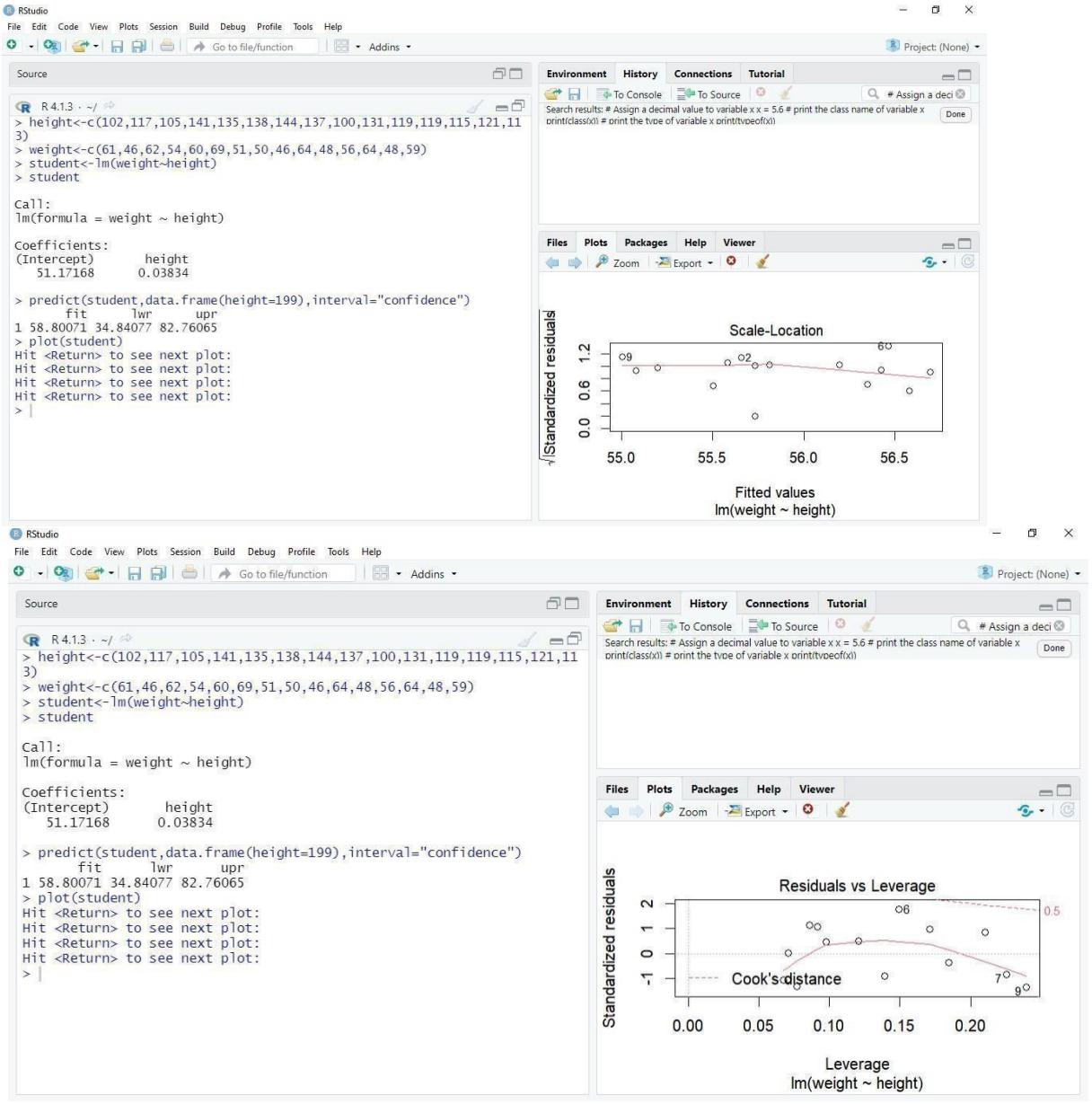
PRACTICL 7

Aim: To implement Simple linear regression

```
height<-
c(102,117,105,141,135,138,144,137,100,131,119,119,115,121,113)
weight<-c(61,46,62,54,60,69,51,50,46,64,48,56,64,48,59) student<-
lm(weight~height) student
predict(student,data.frame(height=199),interval="confidence")
plot(student)
```

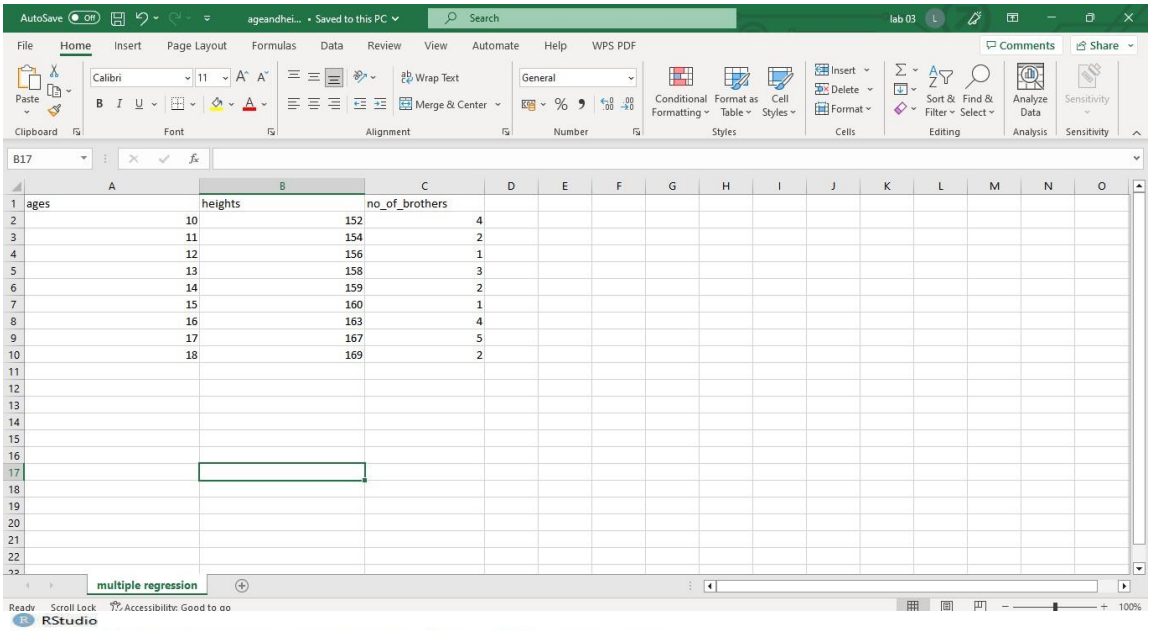




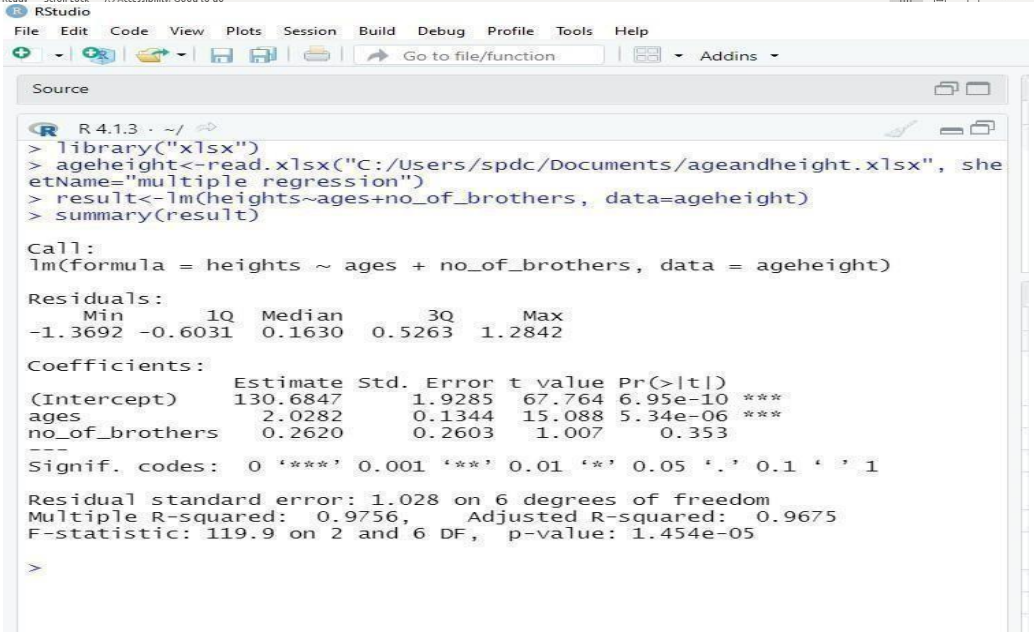


Aim: To implement multiple linear regression

Program: In this program, ages, number of brothers and heights of people are recorded in an excel file "ageandheight.xls" and the relationship between heights (dependent variable) and two independent variables – ages and number of brothers is studied. This relationship between heights and ages, number of brothers can be expressed as a linear equation: Heights = (m1*ages) + (m2* no_of_brothers) + c. M1 and m2 are the co-efficients and c is the intercept.



	A	B	C
1	ages	heights	no_of_brothers
2	10	152	4
3	11	154	2
4	12	156	1
5	13	158	3
6	14	159	2
7	15	160	1
8	16	163	4
9	17	167	5
10	18	169	2

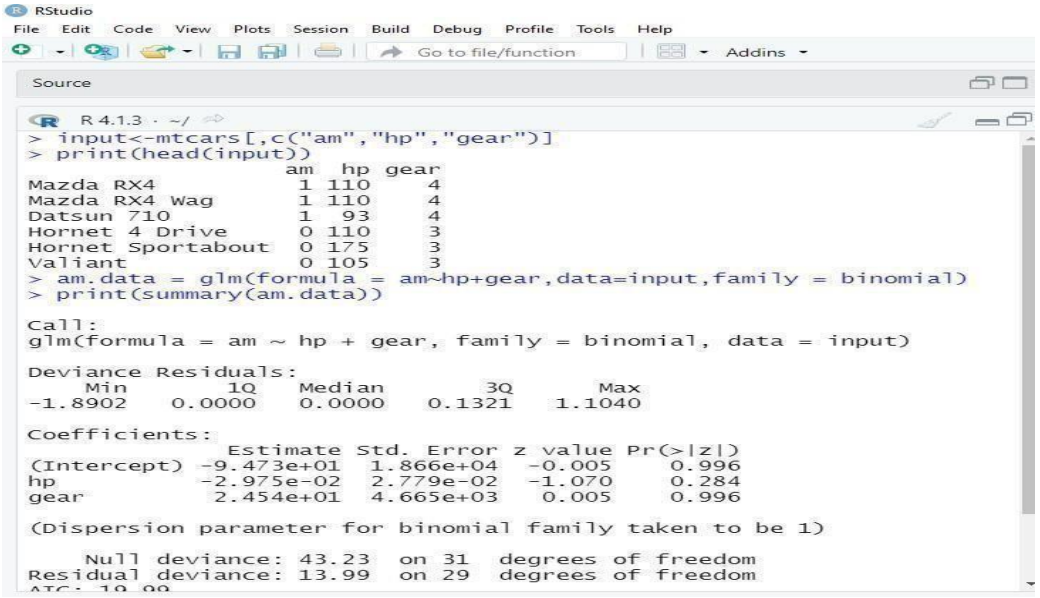


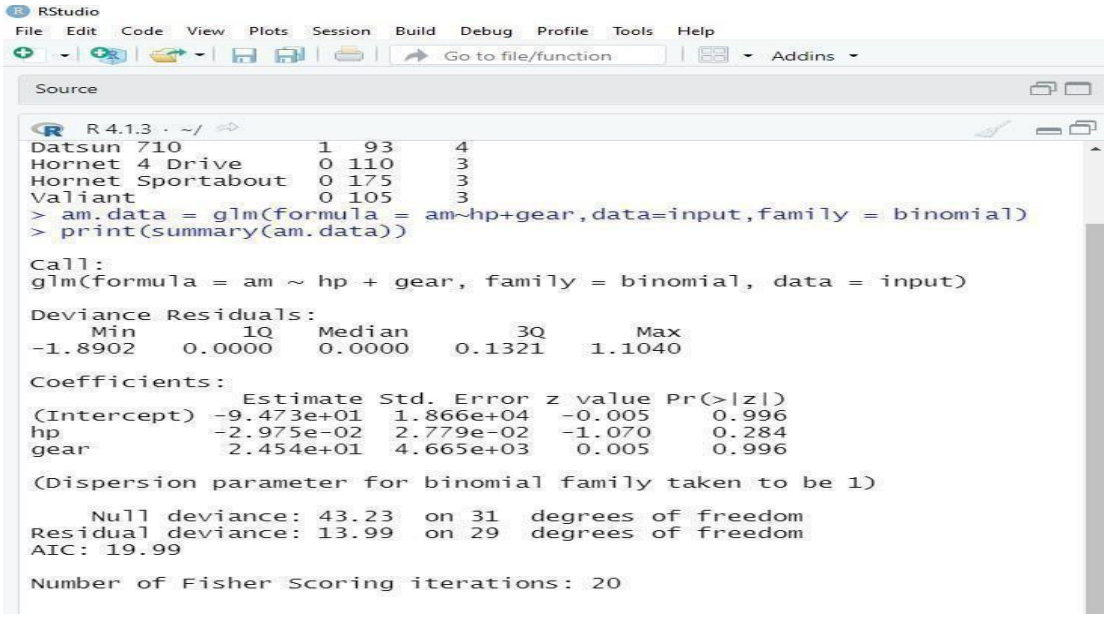
```
> library("xlsx")
> ageheight<-read.xlsx("C:/Users/spdc/Documents/ageandheight.xlsx", she
etName="multiple regression")
> result<-lm(heights~ages+no_of_brothers, data=ageheight)
> summary(result)

Call:
lm(formula = heights ~ ages + no_of_brothers, data = ageheight)

Residuals:
    Min      1Q  Median      3Q     Max
-1.3692 -0.6031  0.1630  0.5263  1.2842

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    130.6847     1.9285  67.764 6.95e-10 ***
ages             2.0282     0.1344  15.088 5.34e-06 ***
no_of_brothers   0.2620     0.2603   1.007    0.353
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.028 on 6 degrees of freedom
Multiple R-squared:  0.9756,    Adjusted R-squared:  0.9675
F-statistic: 119.9 on 2 and 6 DF,  p-value: 1.454e-05

>
```

Aim: To implement Logistic regression

```
> input<-mtcars[,c("am","hp","gear")]
> print(head(input))
```

> am.data = glm(formula =am~hp+gear,data=input,family =binomial) > print(summary(am.data))





Aim: Performing K Nearest Neighbour on Dataset (KNN)

Installing Packages

install.packages("e1071")

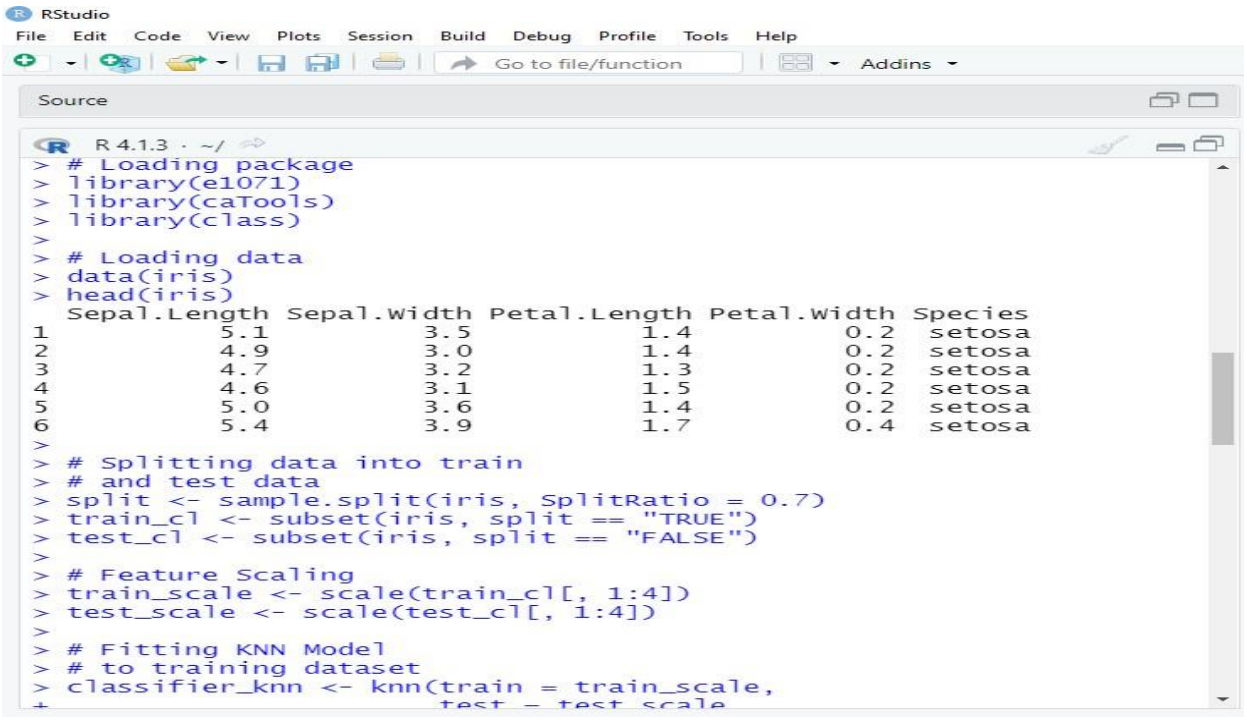
install.packages("caTools")

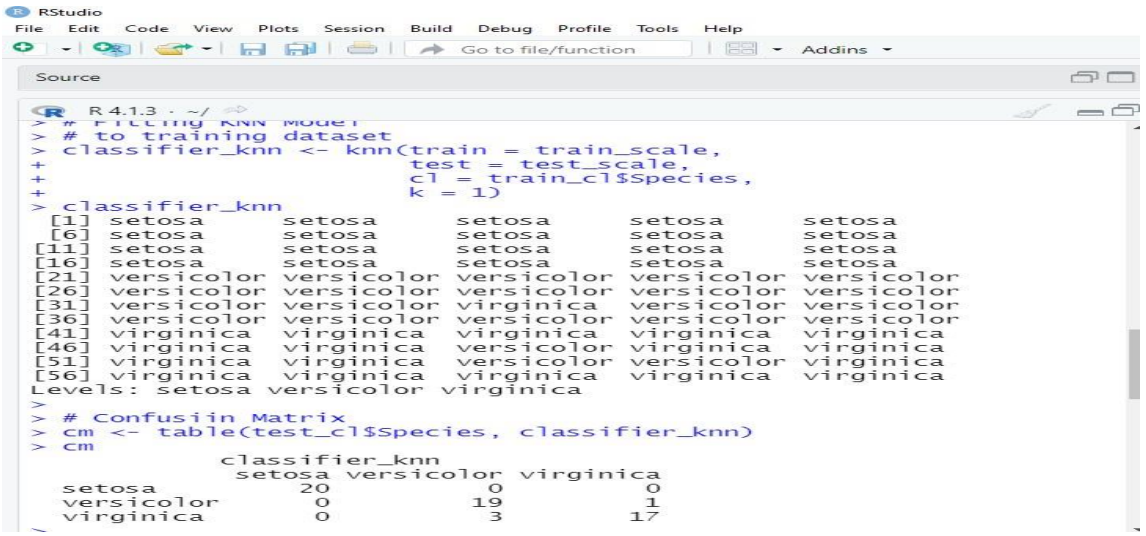
install.packages("class")

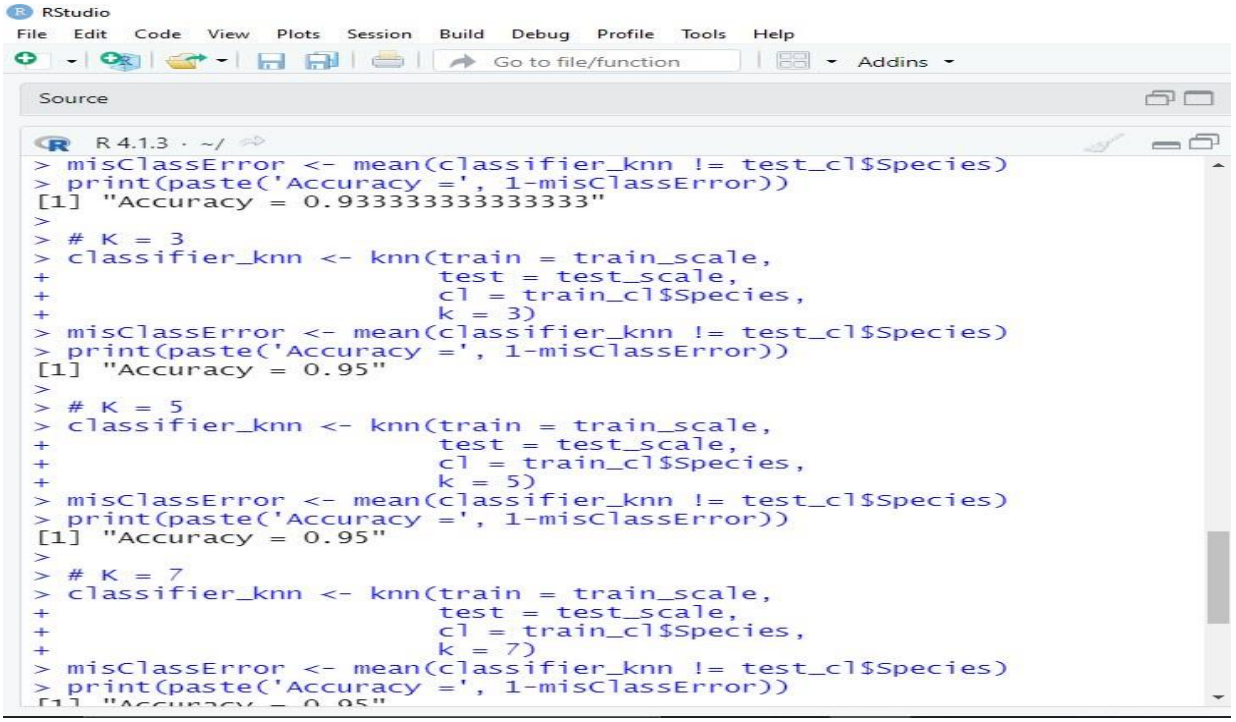
```
# Loading package
library(e1071)
library(caTools)
library(class)

# Loading data
data(iris)
head(iris)

# Splitting data into train
# and test data
split <- sample.split(iris, SplitRatio = 0.7)
train_cl <- subset(iris, split == "TRUE") test_cl
<- subset(iris, split == "FALSE")

# Feature Scaling train_scale <-
scale(train_cl[, 1:4]) test_scale <-
scale(test_cl[, 1:4])

# Fitting KNN Model # to training
dataset classifier_knn <- knn(train =
train_scale,
                              test = test_scale, cl =
                              train_cl$Species, k =
                              1)
classifier_knn

# Confusiin Matrix
cm <- table(test_cl$Species, classifier_knn) cm
```

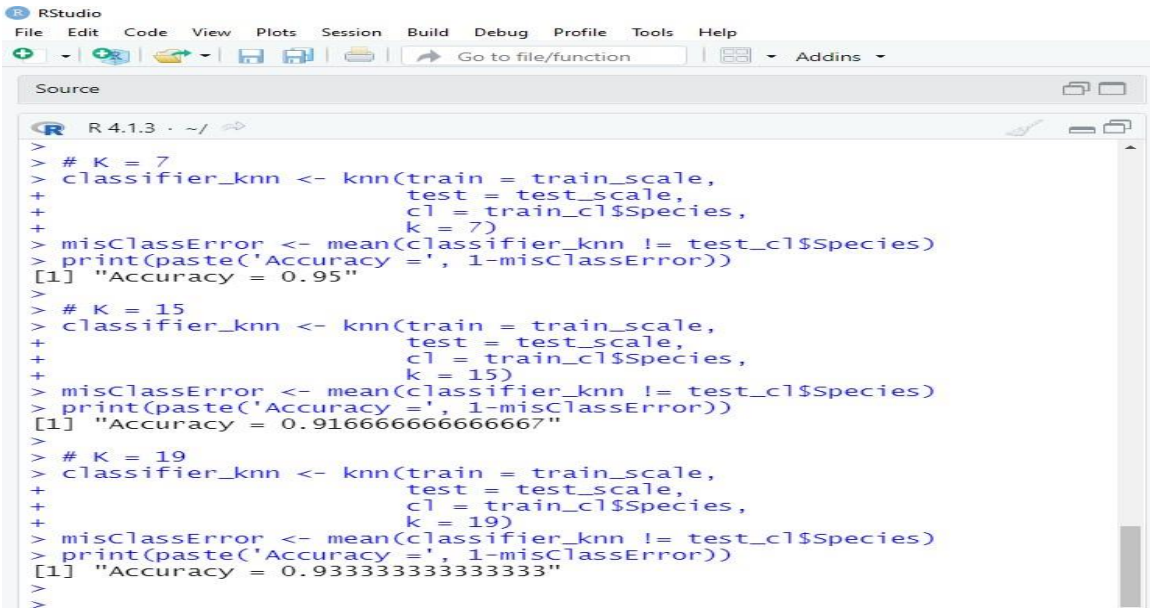
```
# Model Evaluation - Choosing K #
Calculate out of Sample error
misClassError <- mean(classifier_knn != test_cl$Species) print(paste('Accuracy
=', 1-misClassError))

# K = 3
classifier_knn <- knn(train = train_scale,
                      test = test_scale, cl =
                      train_cl$Species, k =
                      3)
misClassError <- mean(classifier_knn != test_cl$Species) print(paste('Accuracy =', 1-
misClassError))

# K = 5
classifier_knn <- knn(train = train_scale,
                      test = test_scale, cl =
                      train_cl$Species, k =
                      5)
misClassError <- mean(classifier_knn != test_cl$Species)
print(paste('Accuracy =', 1-misClassError))

# K = 7
classifier_knn <- knn(train = train_scale, test = test_scale,
                      cl = train_cl$Species,
                      k = 7)
misClassError <- mean(classifier_knn != test_cl$Species) print(paste('Accuracy
=', 1-misClassError))

# K = 15
classifier_knn <- knn(train = train_scale,
```

```
                                        test = test_scale, cl =
                                        train_cl$Species, k =
                                        15)
misClassError <- mean(classifier_knn != test_cl$Species) print(paste('Accuracy
=', 1-misClassError))


# K = 19
classifier_knn <- knn(train = train_scale,
                                        test = test_scale, cl =
                                        train_cl$Species, k =
                                        19)

misClassError <- mean(classifier_knn != test_cl$Species) print(paste('Accuracy
=', 1-misClassError))
```




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
R 4.1.3 · ~/
> # to training dataset
> classifier_knn <- knn(train = train_scale,
+ test = test_scale,
+ cl = train_cl$Species,
+ k = 1)
> classifier_knn
[1] setosa setosa setosa setosa setosa
[6] setosa setosa setosa setosa setosa
[11] setosa setosa setosa setosa setosa
[16] setosa setosa setosa setosa setosa
[21] versicolor versicolor versicolor versicolor versicolor
[26] versicolor versicolor versicolor versicolor versicolor
[31] versicolor versicolor virginica versicolor versicolor
[36] versicolor versicolor versicolor versicolor versicolor
[41] virginica virginica virginica virginica virginica
[46] virginica virginica versicolor virginica virginica
[51] virginica virginica versicolor versicolor virginica
[56] virginica virginica virginica virginica virginica
Levels: setosa versicolor virginica
> # Confusiin Matrix
> cm <- table(test_cl$Species, classifier_knn)
> cm
classifier_knn
setosa versicolor virginica
setosa 20 0 0
versicolor 0 19 1
virginica 0 3 17


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
R 4.1.3 · ~/
> misClassError <- mean(classifier_knn != test_cl$Species)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.933333333333333"
> # K = 3
> classifier_knn <- knn(train = train_scale,
+ test = test_scale,
+ cl = train_cl$Species,
+ k = 3)
> misClassError <- mean(classifier_knn != test_cl$Species)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.95"
> # K = 5
> classifier_knn <- knn(train = train_scale,
+ test = test_scale,
+ cl = train_cl$Species,
+ k = 5)
> misClassError <- mean(classifier_knn != test_cl$Species)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.95"
> # K = 7
> classifier_knn <- knn(train = train_scale,
+ test = test_scale,
+ cl = train_cl$Species,
+ k = 7)
> misClassError <- mean(classifier_knn != test_cl$Species)
> print(paste('Accuracy =', 1-misClassError))



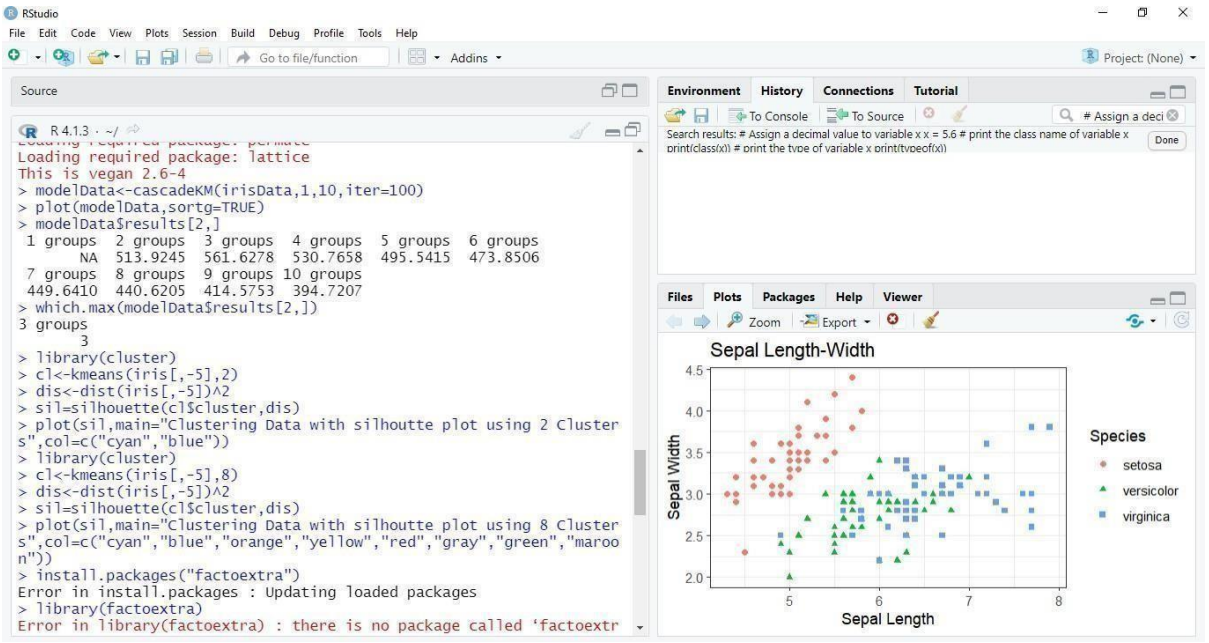

```
> # K = 7
> classifier_knn <- knn(train = train_scale,
+                       test = test_scale,
+                       cl = train_cl$Species,
+                       k = 7)
> misClassError <- mean(classifier_knn != test_cl$Species)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.95"
>
> # K = 15
> classifier_knn <- knn(train = train_scale,
+                       test = test_scale,
+                       cl = train_cl$Species,
+                       k = 15)
> misClassError <- mean(classifier_knn != test_cl$Species)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.916666666666667"
>
> # K = 19
> classifier_knn <- knn(train = train_scale,
+                       test = test_scale,
+                       cl = train_cl$Species,
+                       k = 19)
> misClassError <- mean(classifier_knn != test_cl$Species)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.933333333333333"
>
>
```

PRACTICAL 8

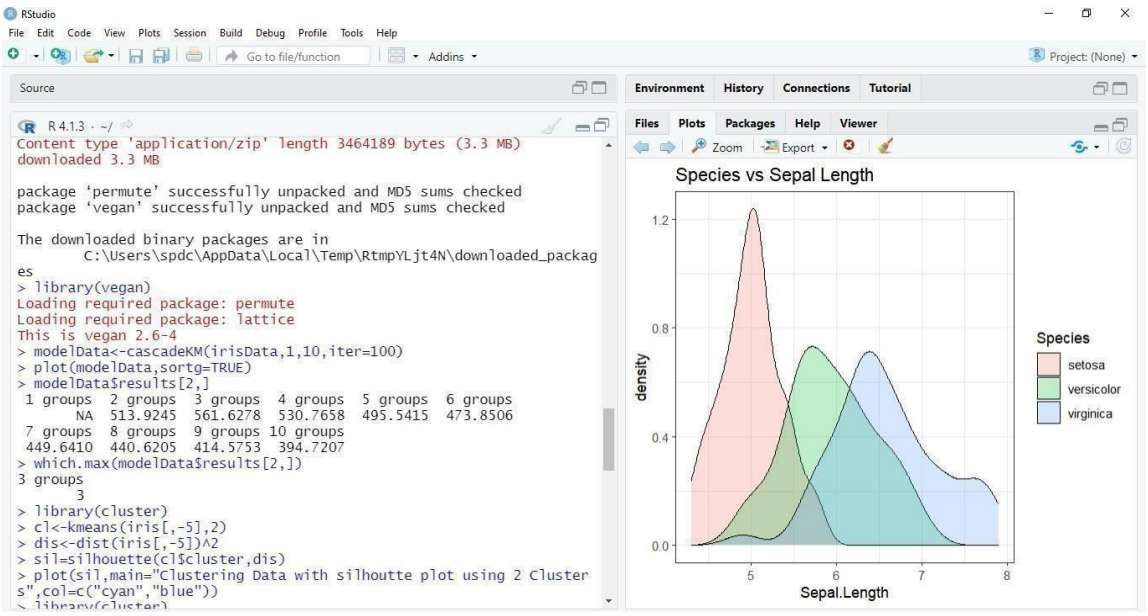
Aim: To implement K means clustering

```
install.packages("ggplot2")
library(ggplot2)
scatter <-
ggplot(data=iris,aes(x=Sepal.Length,y=Sepal.Width)) scatter +
geom_point(aes(color=Species,shape=Species))+
theme_bw()+
  xlab("Sepal Length")+ylab("Sepal Width")+
ggtitle("Sepal Length-Width")
ggplot(data=iris,aes(Sepal.Length,fill=Species))+
theme_bw()+
  geom_density(alpha=0.25)+
labs(x="Sepal.Length",title="Species vs Sepal Length") vol
<- ggplot(data=iris,aes(x=Sepal.Length))
vol + stat_density(aes(ymax=..density..,ymin=-
..density..,fill=Species,color=Species),geom="ribbon",position="identity")+
facet_grid(.~Species)+coord_flip()+theme_bw()+labs(x="Sepal Length",title="Species vs
Sepal Length")
vol <- ggplot(data=iris,aes(x=Sepal.Width)) vol
+ stat_density(aes(ymax=..density..,ymin=-
..density..,fill=Species,color=Species),geom="ribbon",position="identity")+
facet_grid(.~Species)+coord_flip()+theme_bw()+labs(x="Sepal Width",title="Species vs
Sepal Width") irisData <- iris[,1:4] totalwSS<-c() for(i in 1:15)
{clusterIRIS<- kmeans(irisData,centers = i)
totalwSS[i] <-clusterIRIS$tot.withinss}
plot(x=1:15,y=totalwSS,type="b",xlab="Number of Clusters",ylab="Within groups sum-of-
squares") install.packages("NbClust")
library(NbClust)
par(mar=c(2,2,2,2))
```

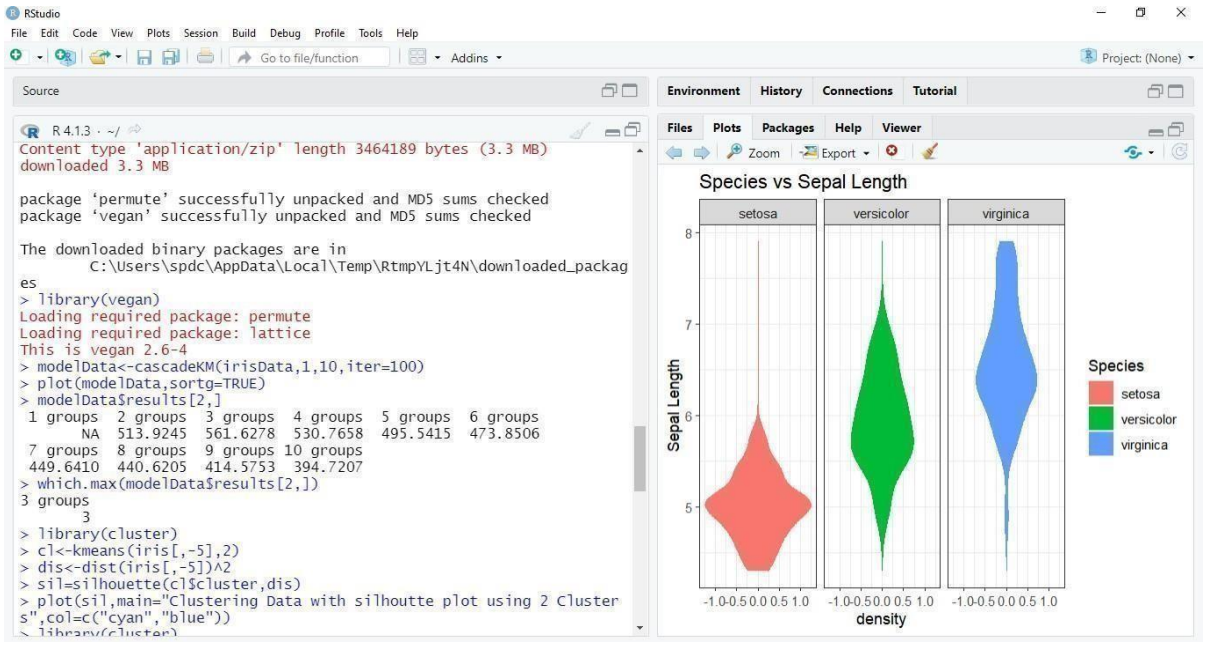
```
nb<-NbClust(irisData,method="kmeans")
hist(nb$Best.nc[1,],breaks=15,main="Histogram for Number of
Clusters") install.packages("vegan") library(vegan)
modelData<-
cascadeKM(irisData,1,10,iter=100)
plot(modelData,sortg=TRUE)
modelData$results[2,]
which.max(modelData$results[2,])
library(cluster) cl<-kmeans(iris[,-5],2) dis<-
dist(iris[,-5])^2 sil=silhouette(cl$cluster,dis)
plot(sil,main="Clustering Data with silhoutte plot using 2
Clusters",col=c("cyan","blue")) library(cluster) cl<-kmeans(iris[,-5],8) dis<-dist(iris[,-
5])^2 sil=silhouette(cl$cluster,dis)
plot(sil,main="Clustering Data with silhoutte plot using 8
Clusters",col=c("cyan","blue","orange","yellow","red","gray","green","maroon"
)) install.packages("factoextra") library(factoextra)
install.packages("clustertend") library(clustertend) genx<-function(x){
runif(length(x),min(x),(max(x)))} random_df<-apply(iris[,-5],2,genx)
random_df<-as.data.frame(random_df) iris[,-5]<-scale(iris[,-5])
random_df<-scale(random_df) res<-get_clust_tendency(iris[,-5],n=nrow(iris)-
1,graph=FALSE) res$hopkins_stat hopkins(iris[,-5],n=nrow(iris)-1)
res<-get_clust_tendency(random_df,n=nrow(random_df)-1,graph=FALSE)
res$hopkins_stat
```

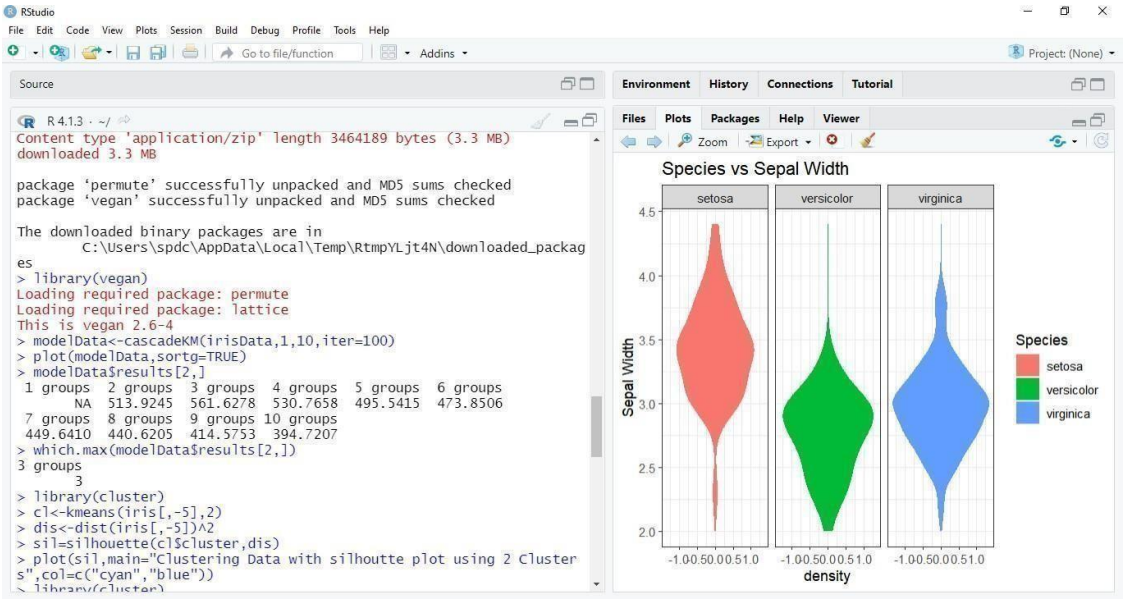


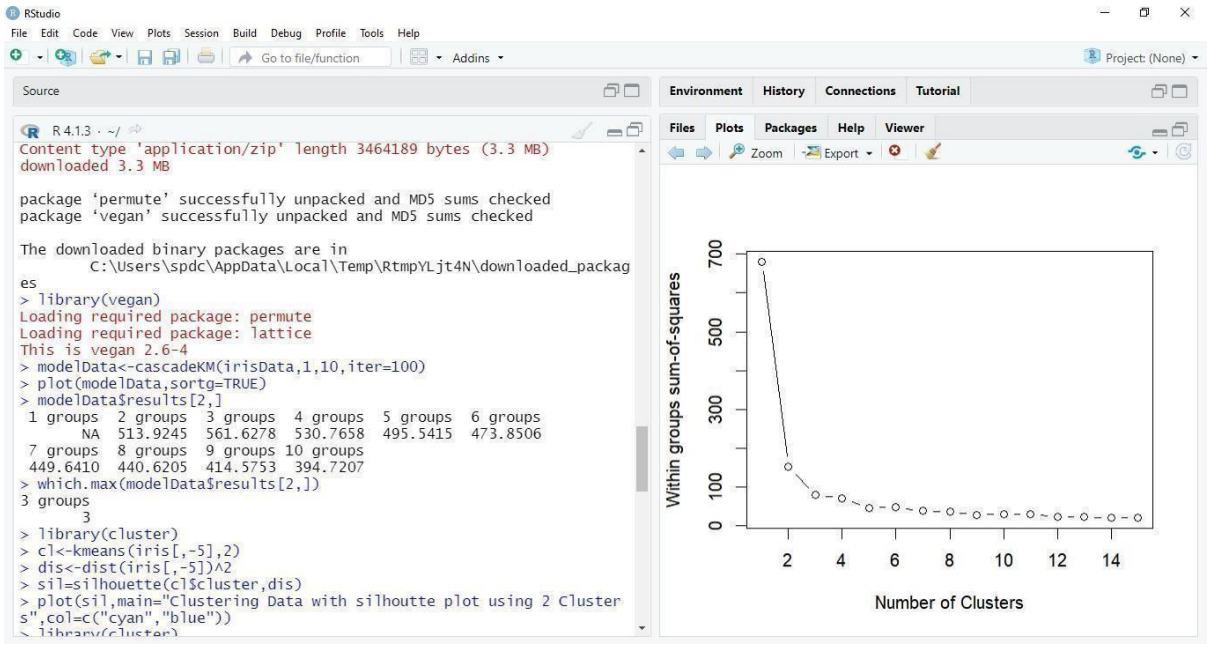

```
Loading required package: lattice
This is vegan 2.6-4
> modelData<-cascadeKM(irisData,1,10,iter=100)
> plot(modelData,sortg=TRUE)
> modelData$results[2,]
 1 groups  2 groups  3 groups  4 groups  5 groups  6 groups
       NA  513.9245  561.6278  530.7658  495.5415  473.8506
 7 groups  8 groups  9 groups 10 groups
 449.6410  440.6205  414.5753  394.7207
> which.max(modelData$results[2,])
3 groups
       3
> library(cluster)
> cl<-kmeans(iris[,-5],2)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 2 Clusters",col=c("cyan","blue"))
> library(cluster)
> cl<-kmeans(iris[,-5],8)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 8 Clusters",col=c("cyan","blue","orange","yellow","red","gray","green","maroon"))
> install.packages("factoextra")
Error in install.packages : Updating loaded packages
> library(factoextra)
Error in library(factoextra) : there is no package called 'factoextr
```

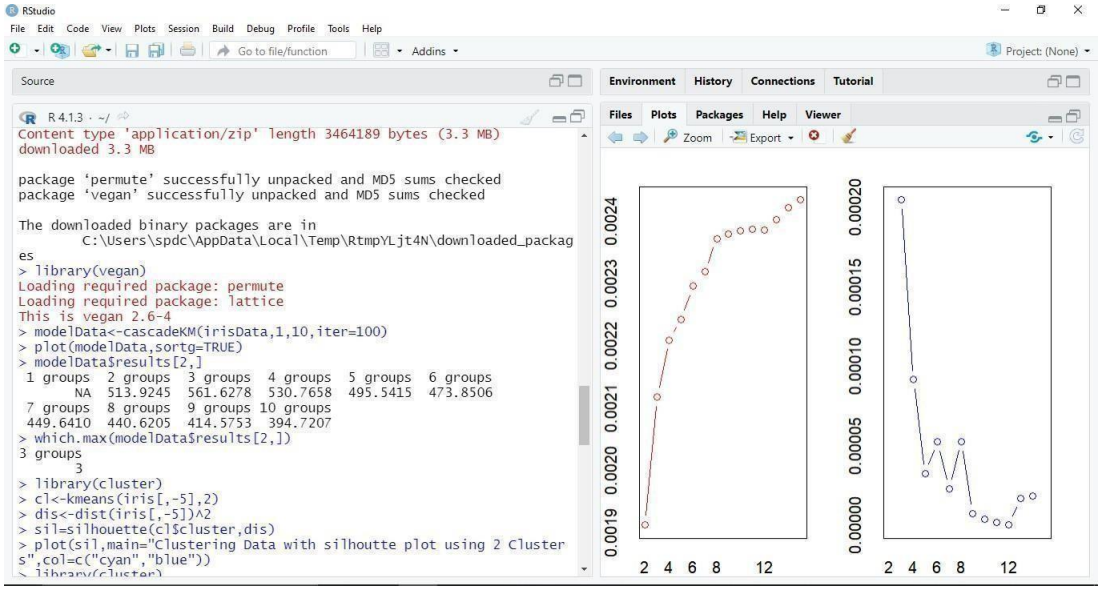


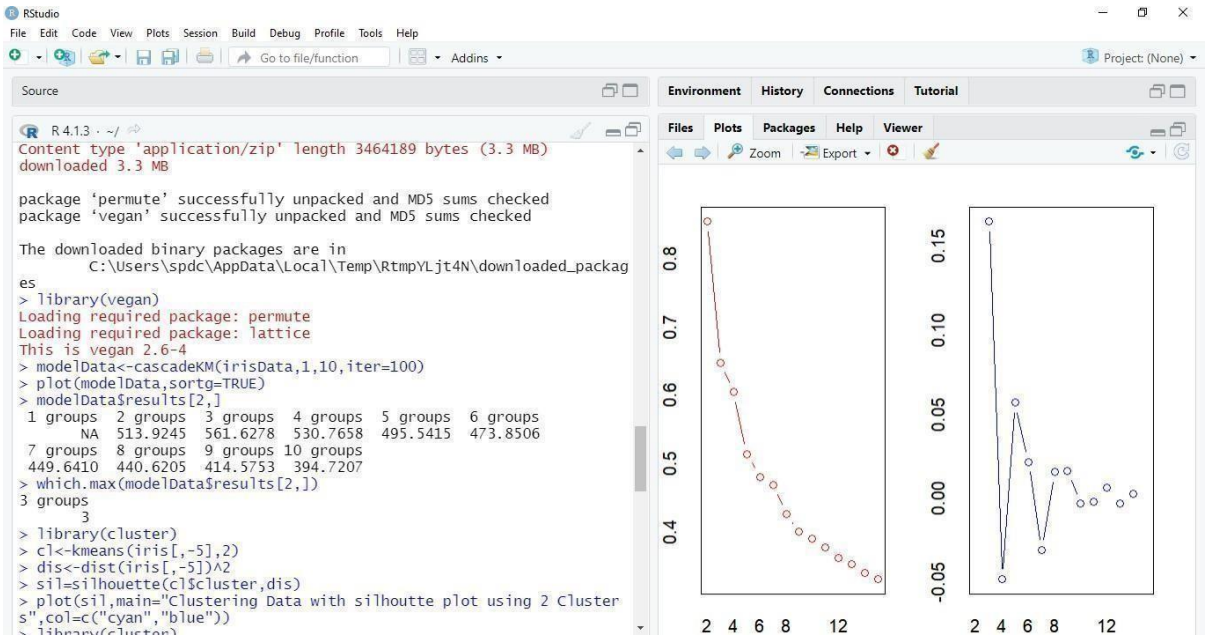

```
Content type 'application/zip' length 3464189 bytes (3.3 MB)
downloaded 3.3 MB

package 'permute' successfully unpacked and MD5 sums checked
package 'vegan' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\spdc\AppData\Local\Temp\RtmpYLjt4N\downloaded_packages
> library(vegan)
Loading required package: permute
Loading required package: lattice
This is vegan 2.6-4
> modelData<-cascadeKM(irisData,1,10,iter=100)
> plot(modelData,sortg=TRUE)
> modelData$results[2,]
 1 groups  2 groups  3 groups  4 groups  5 groups  6 groups
       NA  513.9245  561.6278  530.7658  495.5415  473.8506
 7 groups  8 groups  9 groups 10 groups
 449.6410  440.6205  414.5753  394.7207
> which.max(modelData$results[2,])
3 groups
       3
> library(cluster)
> cl<-kmeans(iris[,-5],2)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 2 Clusters",col=c("cyan","blue"))
```

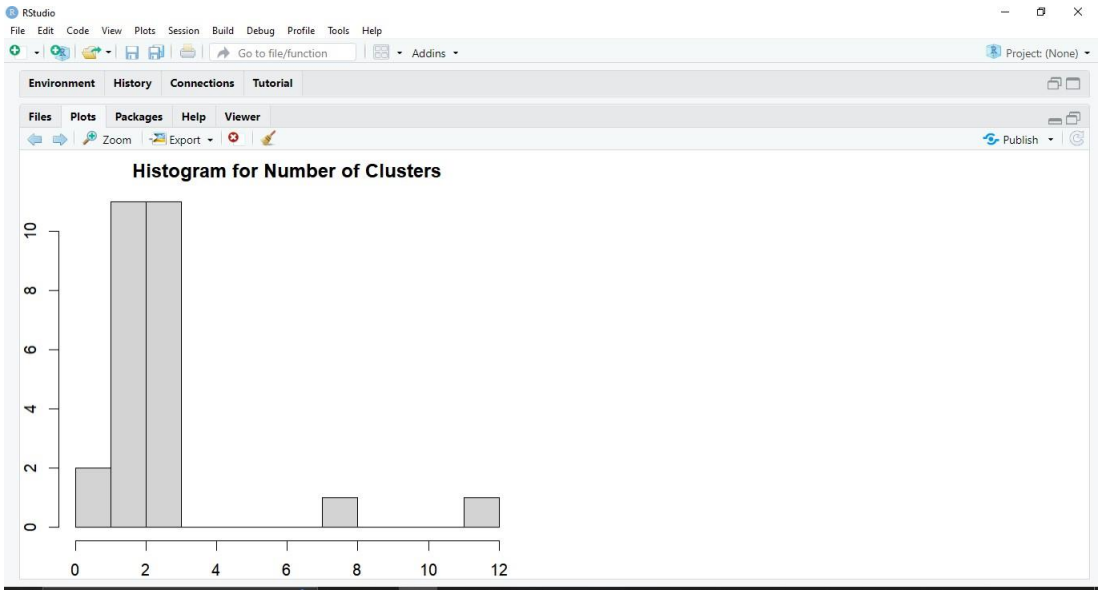

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Source
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom Export
Species vs Sepal Length
setosa
versicolor
virginica
Sepal Length
density
Species
setosa
versicolor
virginica
R 4.1.3 · ~/
Content type 'application/zip' length 3464189 bytes (3.3 MB)
downloaded 3.3 MB
package 'permute' successfully unpacked and MD5 sums checked
package 'vegan' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\spdc\AppData\Local\Temp\RtmpYLjt4N\downloaded_packag
es
> library(vegan)
Loading required package: permute
Loading required package: lattice
This is vegan 2.6-4
> modelData<-cascadeKM(irisData,1,10,iter=100)
> plot(modelData,sortg=TRUE)
> modelData$results[2,]
1 groups 2 groups 3 groups 4 groups 5 groups 6 groups
NA 513.9245 561.6278 530.7658 495.5415 473.8506
7 groups 8 groups 9 groups 10 groups
449.6410 440.6205 414.5753 394.7207
> which.max(modelData$results[2,])
3 groups
3
> library(cluster)
> cl<-kmeans(iris[,-5],2)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 2 Cluster
s",col=c("cyan","blue"))

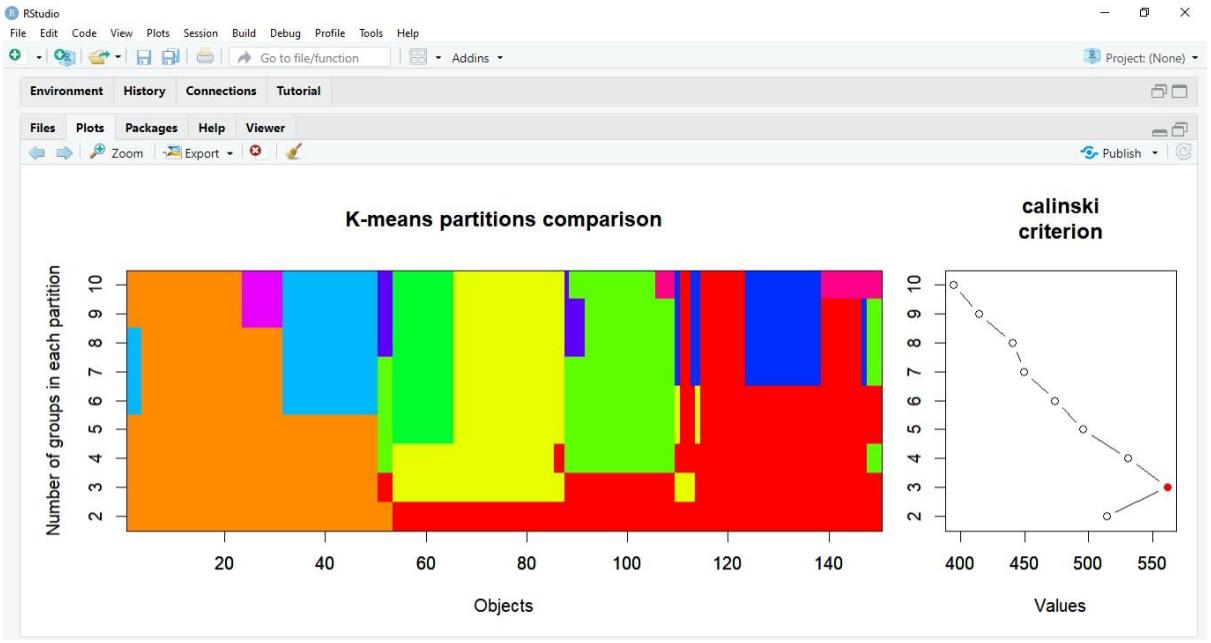

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Source
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom Export
Species vs Sepal Width
setosa
versicolor
virginica
Sepal Width
density
Species
setosa
versicolor
virginica
R 4.1.3 · ~/
Content type 'application/zip' length 3464189 bytes (3.3 MB)
downloaded 3.3 MB
package 'permute' successfully unpacked and MD5 sums checked
package 'vegan' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\spdc\AppData\Local\Temp\RtmpYLjt4N\downloaded_packag
es
> library(vegan)
Loading required package: permute
Loading required package: lattice
This is vegan 2.6-4
> modelData<-cascadeKM(irisData,1,10,iter=100)
> plot(modelData,sortg=TRUE)
> modelData$results[2,]
1 groups 2 groups 3 groups 4 groups 5 groups 6 groups
NA 513.9245 561.6278 530.7658 495.5415 473.8506
7 groups 8 groups 9 groups 10 groups
449.6410 440.6205 414.5753 394.7207
> which.max(modelData$results[2,])
3 groups
3
> library(cluster)
> cl<-kmeans(iris[,-5],2)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 2 Cluster
s",col=c("cyan","blue"))

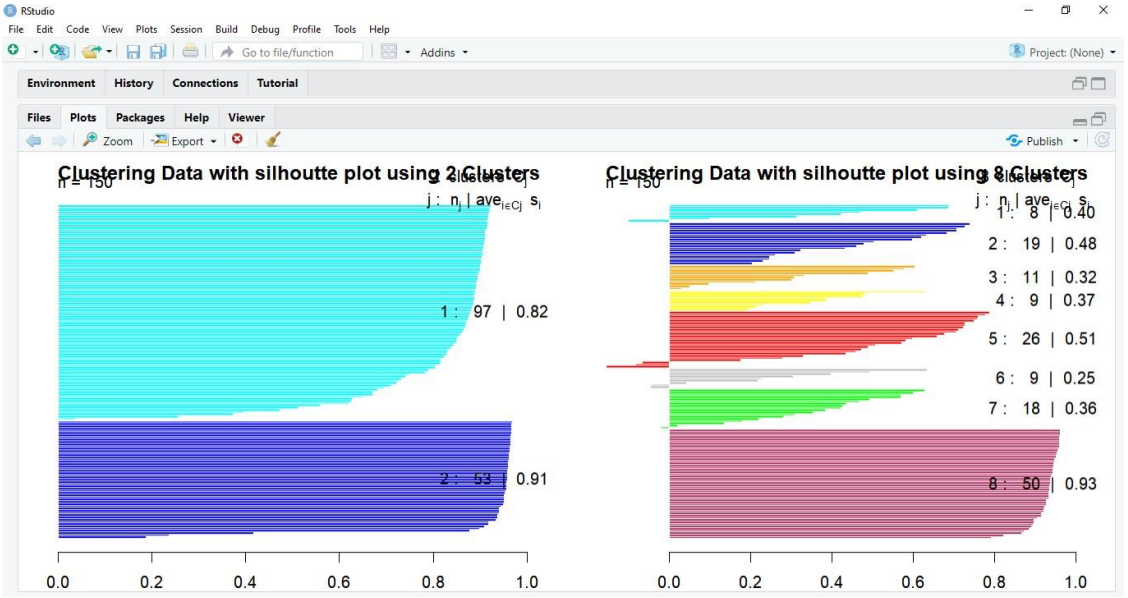

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Source
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom Export
R 4.1.3 · ~/
Content type 'application/zip' length 3464189 bytes (3.3 MB)
downloaded 3.3 MB
package 'permute' successfully unpacked and MD5 sums checked
package 'vegan' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\spdc\AppData\Local\Temp\RtmpYLjt4N\downloaded_packag
es
> library(vegan)
Loading required package: permute
Loading required package: lattice
This is vegan 2.6-4
> modelData<-cascadeKM(irisData,1,10,iter=100)
> plot(modelData,sortg=TRUE)
> modelData$results[2,]
1 groups 2 groups 3 groups 4 groups 5 groups 6 groups
NA 513.9245 561.6278 530.7658 495.5415 473.8506
7 groups 8 groups 9 groups 10 groups
449.6410 440.6205 414.5753 394.7207
> which.max(modelData$results[2,])
3 groups
3
> library(cluster)
> cl<-kmeans(iris[,-5],2)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 2 Cluster
s",col=c("cyan","blue"))
Within groups sum-of-squares
0 100 300 500 700
2 4 6 8 10 12 14
Number of Clusters


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Source
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom Export
R 4.1.3 · ~/
Content type 'application/zip' length 3464189 bytes (3.3 MB)
downloaded 3.3 MB
package 'permute' successfully unpacked and MD5 sums checked
package 'vegan' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\spdc\AppData\Local\Temp\RtmpYLjt4N\downloaded_packag
es
> library(vegan)
Loading required package: permute
Loading required package: lattice
This is vegan 2.6-4
> modelData<-cascadeKM(irisData,1,10,iter=100)
> plot(modelData,sortg=TRUE)
> modelData$results[2,]
1 groups 2 groups 3 groups 4 groups 5 groups 6 groups
NA 513.9245 561.6278 530.7658 495.5415 473.8506
7 groups 8 groups 9 groups 10 groups
449.6410 440.6205 414.5753 394.7207
> which.max(modelData$results[2,])
3 groups
3
> library(cluster)
> cl<-kmeans(iris[,-5],2)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 2 Cluster
s",col=c("cyan","blue"))
0.0019 0.0020 0.0021 0.0022 0.0023 0.0024
2 4 6 8 12
0.00000 0.00005 0.00010 0.00015 0.00020
2 4 6 8 12


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Source
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom Export
R 4.1.3 · ~/
Content type 'application/zip' length 3464189 bytes (3.3 MB)
downloaded 3.3 MB
package 'permute' successfully unpacked and MD5 sums checked
package 'vegan' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\spdc\AppData\Local\Temp\RtmpYLjt4N\downloaded_packag
es
> library(vegan)
Loading required package: permute
Loading required package: lattice
This is vegan 2.6-4
> modelData<-cascadeKM(irisData,1,10,iter=100)
> plot(modelData,sortg=TRUE)
> modelData$results[2,]
1 groups 2 groups 3 groups 4 groups 5 groups 6 groups
NA 513.9245 561.6278 530.7658 495.5415 473.8506
7 groups 8 groups 9 groups 10 groups
449.6410 440.6205 414.5753 394.7207
> which.max(modelData$results[2,])
3 groups
3
> library(cluster)
> cl<-kmeans(iris[,-5],2)
> dis<-dist(iris[,-5])^2
> sil=silhouette(cl$cluster,dis)
> plot(sil,main="Clustering Data with silhoutte plot using 2 Cluster
s",col=c("cyan","blue"))
0.4 0.5 0.6 0.7 0.8
2 4 6 8 12
-0.05 0.00 0.05 0.10 0.15
2 4 6 8 12


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom Export Publish
Histogram for Number of Clusters
0 2 4 6 8 10
0 2 4 6 8 10 12


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom
Export
Publish
K-means partitions comparison
Number of groups in each partition
Objects
calinski criterion
Values


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Environment History Connections Tutorial
Files Plots Packages Help Viewer
Zoom
Export
Publish
Clustering Data with silhoutte plot using 2 Clusters
n = 150
2 clusters Cj
j : nj | avei∈Cj si
1 : 97 | 0.82
2 : 53 | 0.91
Clustering Data with silhoutte plot using 8 Clusters
n = 150
8 clusters Cj
j : nj | avei∈Cj si
1 : 8 | 0.40
2 : 19 | 0.48
3 : 11 | 0.32
4 : 9 | 0.37
5 : 26 | 0.51
6 : 9 | 0.25
7 : 18 | 0.36
8 : 50 | 0.93