Practical No: 1

Aim: Install, configure and run Hadoop and HDFS

Description:

Hadoop Installation.

Step 1: downlaod java jdk first .the package size 168.67MB

Windows x64	168.67 MB	jdk-8u291-windows-x64.exe

hadoop-2.10.1-src.tar.gz	16-05-2021 17:16	WinRAR archive	43,967 KB
hqbhjb.txt	06-05-2021 08:23	Text Document	1 KB
jdk-8u291-windows-x64.exe	16-05-2021 17:16	Application	1,72,731 KB
LogisticRegressionGFG.png	23-05-2021 17:04	PNG File	4 KB

Step 2: download Hadoop binaries from the official website. The binary package size is about 342 MB.

Download

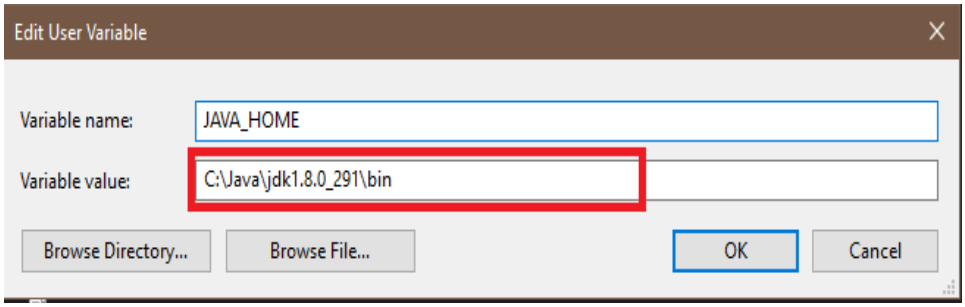
Hadoop is released as source code tarballs with corresponding binary tarballs for convenience. The downloads are distributed via mirror sites and should be checked for tamperi using GPG or SHA-512.

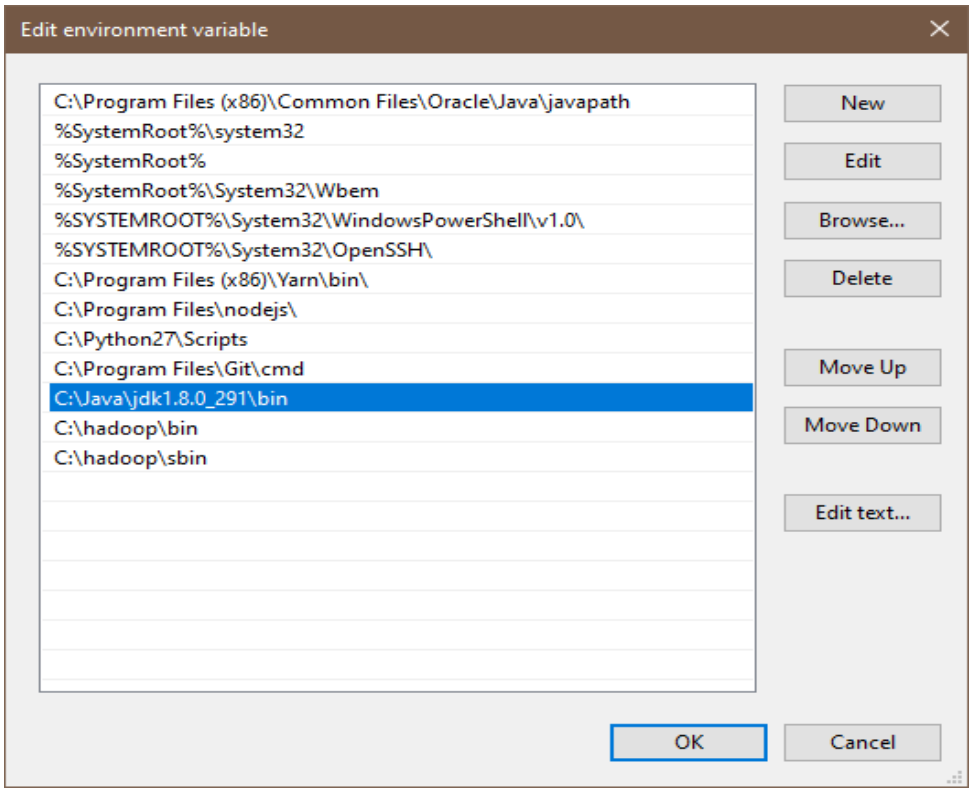
Version	Release date	Source download	Binary download	Release notes
3.2.2	2021 Jan 9	source (checksum signature)	binary (checksum signature)	Announcement
2.10.1	2020 Sep 21	source (checksum signature)	binary (checksum signature)	Announcement
3.1.4	2020 Aug 3	source (checksum signature)	binary (checksum signature)	Announcement
3.3.0	2020 Jul 14	source (checksum signature)	binary (checksum signature) binary-aarch64 (checksum signature)	Announcement

Step 3: After finishing the file download, we should unpack the package using 7zip int two steps. First, we should extract the hadoop-3.2.1.tar.gz library, and then, we shouldunpack the extracted tar file:

Name	Date modified	Type	Size
hadoop-3.3.0.tar.gz	12-05-2021 08:51	WinRAR archive	4,89,013 KB
wavelets_0.3-0.2.tar.gz	12-05-2021 08:27	WinRAR archive	114 KB
govind.data	12-05-2021 08:24	DATA File	283 KB

Step 4: When the “Advanced system settings” dialog appears, go to the “Advanced”tab and click on the “Environment variables” button located on the bottom of the dialog.





Step 5: Check the version of java

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19041.928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>javac
Usage: javac <options> <source files>
where possible options include:
  -g                         Generate all debugging info
  -g:none                    Generate no debugging info
  -g:{lines,vars,source}     Generate only some debugging info
  -nowarn                    Generate no warnings
  -verbose                   Output messages about what the compiler is doing
  -deprecation               Output source locations where deprecated APIs are used
  -classpath <path>          Specify where to find user class files and annotation process
  -cp <path>                 Specify where to find user class files and annotation process
  -sourcepath <path>         Specify where to find input source files
  -bootclasspath <path>      Override location of bootstrap class files
  -extdirs <dirs>            Override location of installed extensions
  -endorseddirs <dirs>       Override location of endorsed standards path
  -proc:{none,only}          Control whether annotation processing and/or compilation is d
  -processor <class1>[,<class2>,<class3>...] Names of the annotation processors to run; by
ss
```

```
C:\Users\hp>java -version
java version "1.8.0_291"
Java(TM) SE Runtime Environment (build 1.8.0_291-b10)
Java HotSpot(TM) 64-Bit Server VM (build 25.291-b10, mixed mode)
```

Step 6: Configuration core-site.xml

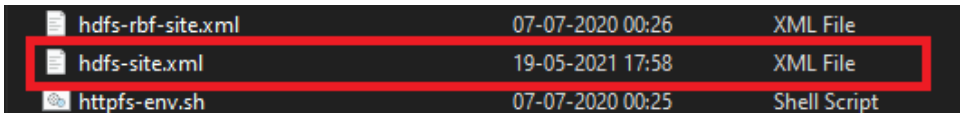
container-executor.cfg	07-07-2020 01:03	CFG File
core-site.xml	19-05-2021 17:57	XML File
hadoop-env.cmd	19-05-2021 17:57	Windows Comma...

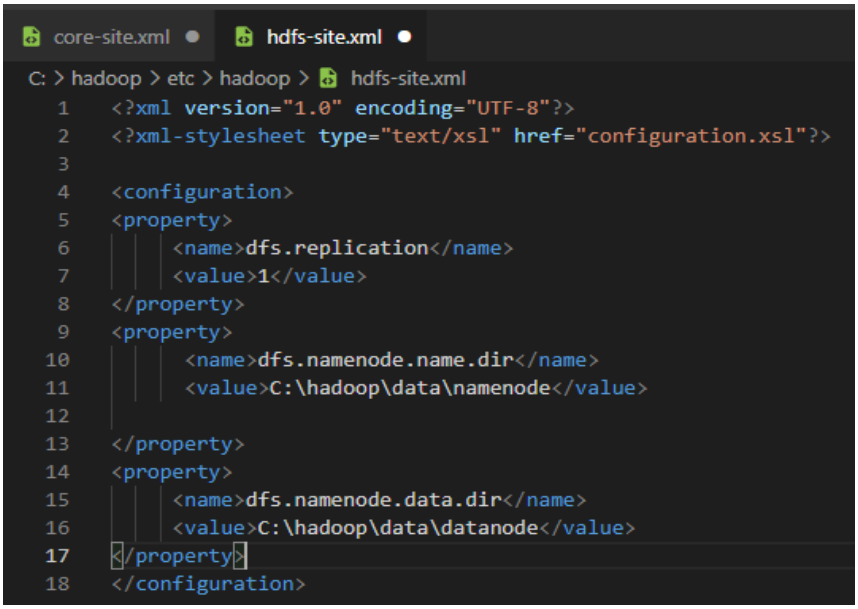
core-site.xml

C: > hadoop > etc > hadoop > core-site.xml

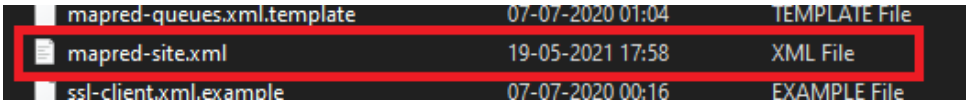
```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>

<property>
        <name>fs.deafultFS</name>
        <value>hdfs://localhost:9000</value>
<property>
</configuration>
```

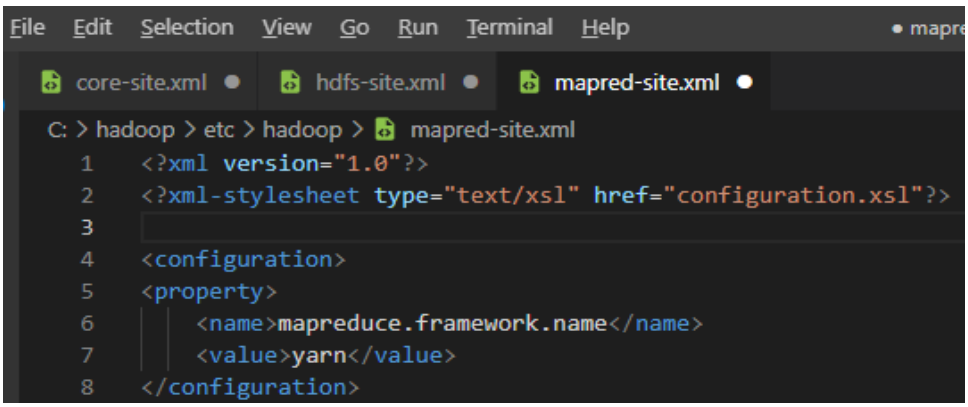
Step 7: Configuration core-site.xml.



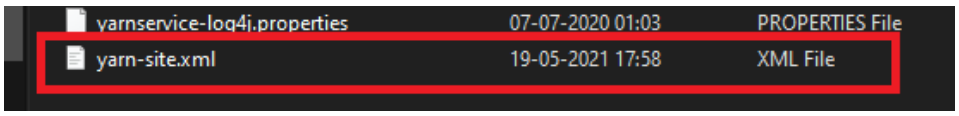


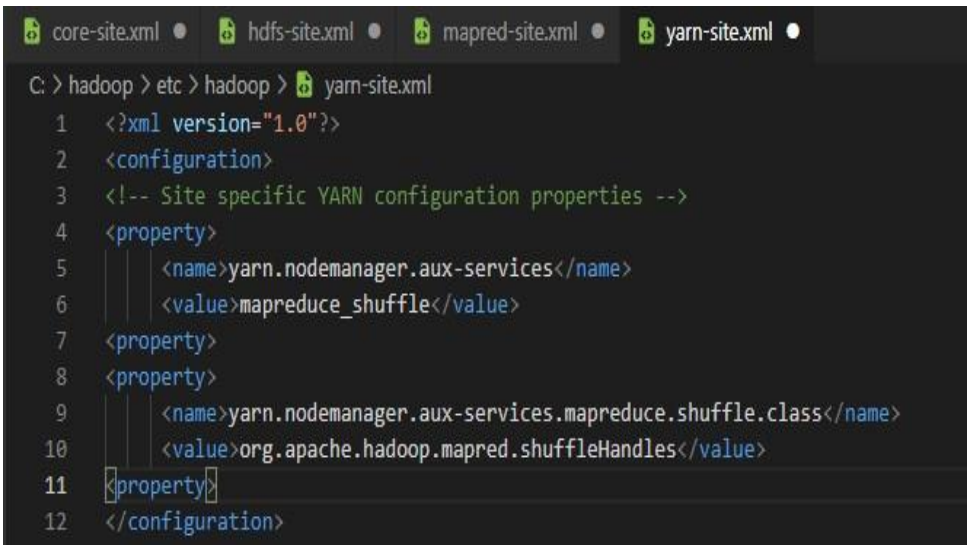
Step 8: Configuration core-site.xml



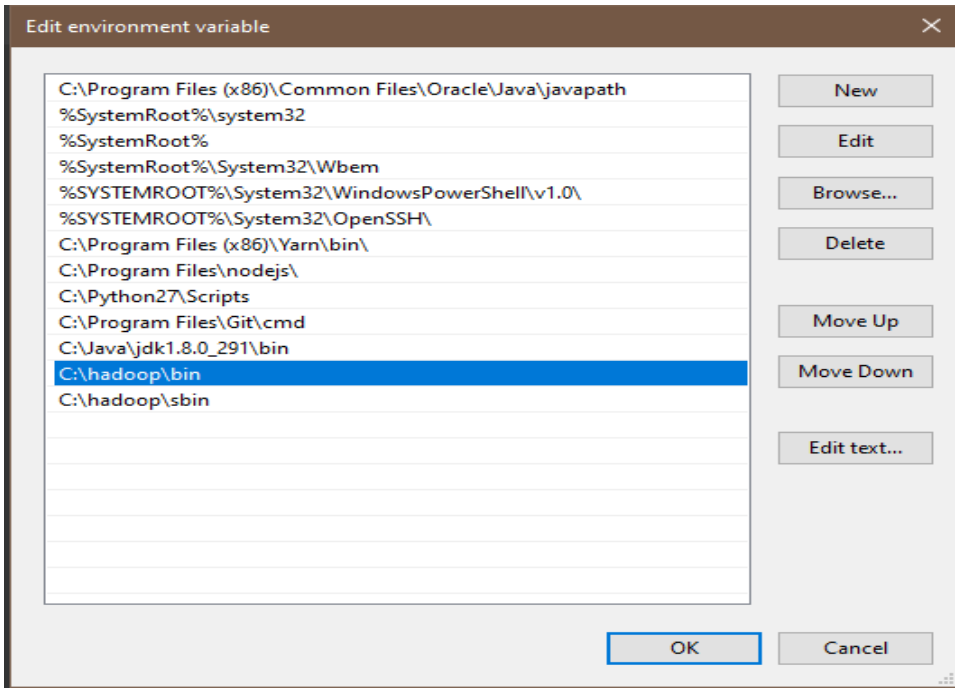


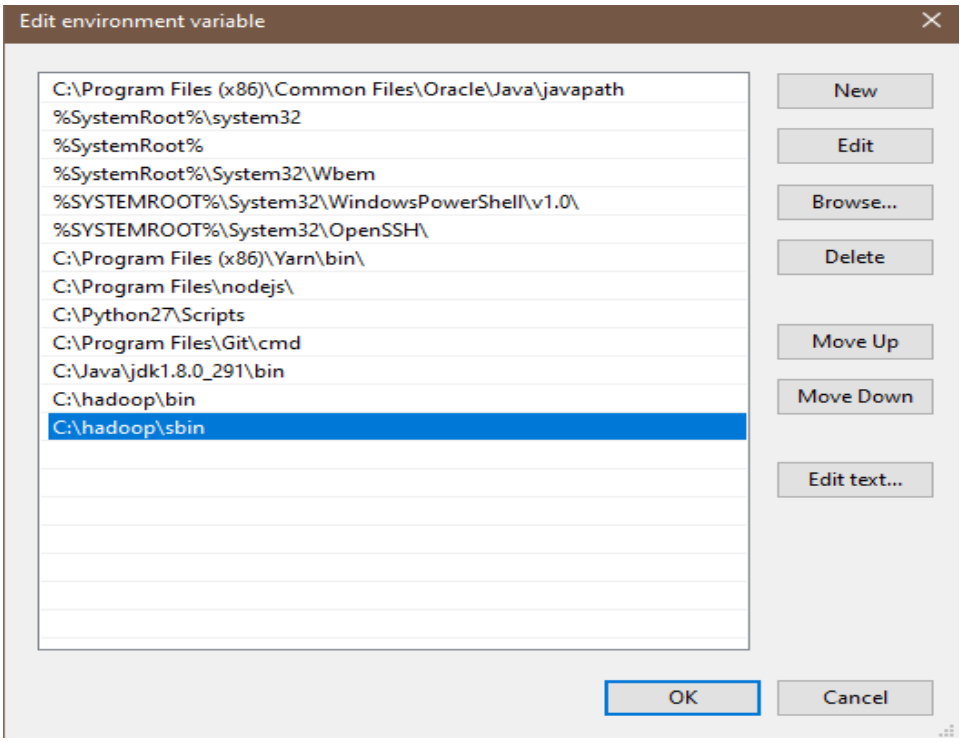
Step 9: Configuration core-site.xml.



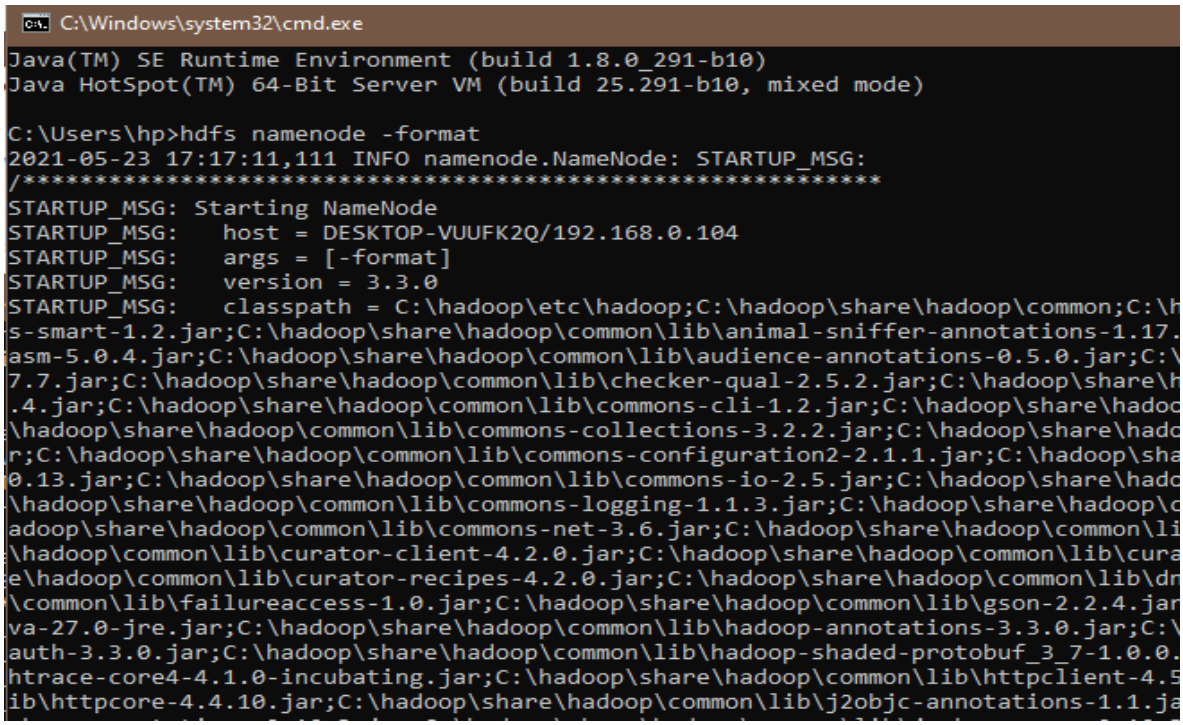


Step 10: When the “Advanced system settings” dialog appears, go to the “Advanced”tab and click on the “Environment variables” button located on the bottom of the dialog.





Step 11: let's check Hadoop install Successfully.



```
Apache Hadoop Distribution
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
2021-05-23 17:19:33,116 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = DESKTOP-VUUFK2Q/192.168.0.104
STARTUP_MSG:   args = []
STARTUP_MSG:   version = 3.3.0
STARTUP_MSG:   classpath = C:\hadoop\etc\hadoop;C:\hadoop\share\hadoop\common;C:\hadoop\share\hadoop\common\lib\accessor
s-smart-1.2.jar;C:\hadoop\share\hadoop\common\lib\animal-sniffer-annotations-1.17.jar;C:\hadoop\share\hadoop\common\lib\
asm-5.0.4.jar;C:\hadoop\share\hadoop\common\lib\audience-annotations-0.5.0.jar;C:\hadoop\share\hadoop\common\lib\avro-1.
7.7.jar;C:\hadoop\share\hadoop\common\lib\checker-qual-2.5.2.jar;C:\hadoop\share\hadoop\common\lib\commons-beanutils-1.9
.4.jar;C:\hadoop\share\hadoop\common\lib\commons-cli-1.2.jar;C:\hadoop\share\hadoop\common\lib\commons-codec-1.11.jar;C:
\hadoop\share\hadoop\common\lib\commons-collections-3.2.2.jar;C:\hadoop\share\hadoop\common\lib\commons-compress-1.19.ja
r;C:\hadoop\share\hadoop\common\lib\commons-configuration2-2.1.1.jar;C:\hadoop\share\hadoop\common\lib\commons-daemon-1.
0.13.jar;C:\hadoop\share\hadoop\common\lib\commons-io-2.5.jar;C:\hadoop\share\hadoop\common\lib\commons-lang3-3.7.jar;C:
\hadoop\share\hadoop\common\lib\commons-logging-1.1.3.jar;C:\hadoop\share\hadoop\common\lib\commons-math3-3.1.1.jar;C:\h
adoop\share\hadoop\common\lib\commons-net-3.6.jar;C:\hadoop\share\hadoop\common\lib\commons-text-1.4.jar;C:\hadoop\share
\hadoop\common\lib\curator-client-4.2.0.jar;C:\hadoop\share\hadoop\common\lib\curator-framework-4.2.0.jar;C:\hadoop\shar
e\hadoop\common\lib\curator-recipes-4.2.0.jar;C:\hadoop\share\hadoop\common\lib\dnsjava-2.1.7.jar;C:\hadoop\share\hadoop
```

```
Apache Hadoop Distribution
        at com.ctc.wstx.sr.StreamScanner.throwParseError(StreamScanner.java:491)
        at com.ctc.wstx.sr.StreamScanner.throwParseError(StreamScanner.java:475)
        at com.ctc.wstx.sr.BasicStreamReader.reportWrongEndElem(BasicStreamReader.java:3365)
        at com.ctc.wstx.sr.BasicStreamReader.readEndElem(BasicStreamReader.java:3292)
        at com.ctc.wstx.sr.BasicStreamReader.nextFromTree(BasicStreamReader.java:2911)
        at com.ctc.wstx.sr.BasicStreamReader.next(BasicStreamReader.java:1123)
        at org.apache.hadoop.conf.Configuration$Parser.parseNext(Configuration.java:3347)
        at org.apache.hadoop.conf.Configuration$Parser.parse(Configuration.java:3141)
        at org.apache.hadoop.conf.Configuration.loadResource(Configuration.java:3034)
        ... 9 more
```

Step 12: Let check bin

```
C:\Windows\system32\cmd.exe

C:\Users\hp>cd C:\hadoop\sbin

C:\hadoop\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\hadoop\sbin>
```

Practical No: 2

Aim: **Classification using SVM**

Requirement:
R tool

Code:

```
getwd()

read.csv()

ds=read.csv("E:/Shiva_big_data_pract/Shivam/socialnetworking.csv",TRUE ",")

ds

ds=ds[3:5]

ds

install("catools")

library(caTools)

set.seed(123)

split=sample.split(ds$Purchased, SplitRatio=0.75)

training_set=(subset(ds, split == TRUE))

test_set =(subset(ds, split == FALSE))

ds

test_set[-3]=scale(test_set[-3])

training_set[-3]=scale(training_set[-3])

test_set[-3] = scale(test_set[-3])

test_set[-3]

install.packages('e1071')

library('e1071')

classifier = svm(formula = Purchased ~ ., data = training_set , type  = 'C-classification', kernal ='linear')

classifier

y_pred=predict(classifier, newdata=test_set[-3])

y_pred

cm=table(test_set[, 3],y_pred)

cm
```

Output:

```
>
> y_pred=predict(classifier, newdata=test_set[-3])
> y_pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46  48  52  66  69  74  75  82  84  85  86  87  89
  0   0   0   0   0   1   1   1   0   0   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   1   0   0
103 104 107 108 109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175 176 193 199 200 208 213 224 226
  0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   1   1   0
228 229 230 234 236 237 239 241 255 264 265 266 273 274 281 286 292 299 302 305 307 310 316 324 326 332 339 341
  1   0   0   1   1   0   1   1   1   0   1   1   1   1   1   1   1   0   1   0   1   0   0   1   0   1   0   1
343 347 353 363 364 367 368 369 372 373 380 383 389 392 395 400
  0   1   1   0   0   1   1   0   1   0   1   1   1   1   0   1
Levels: 0 1
>
>
> cm=table(test_set[, 3],y_pred)
> cm
   y_pred
     0  1
  0 58  6
  1  4 32
```

set = training_set

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid_set = expand.grid(X1, X2)

colnames(grid_set) = c('Age', 'EstimatedSalary')

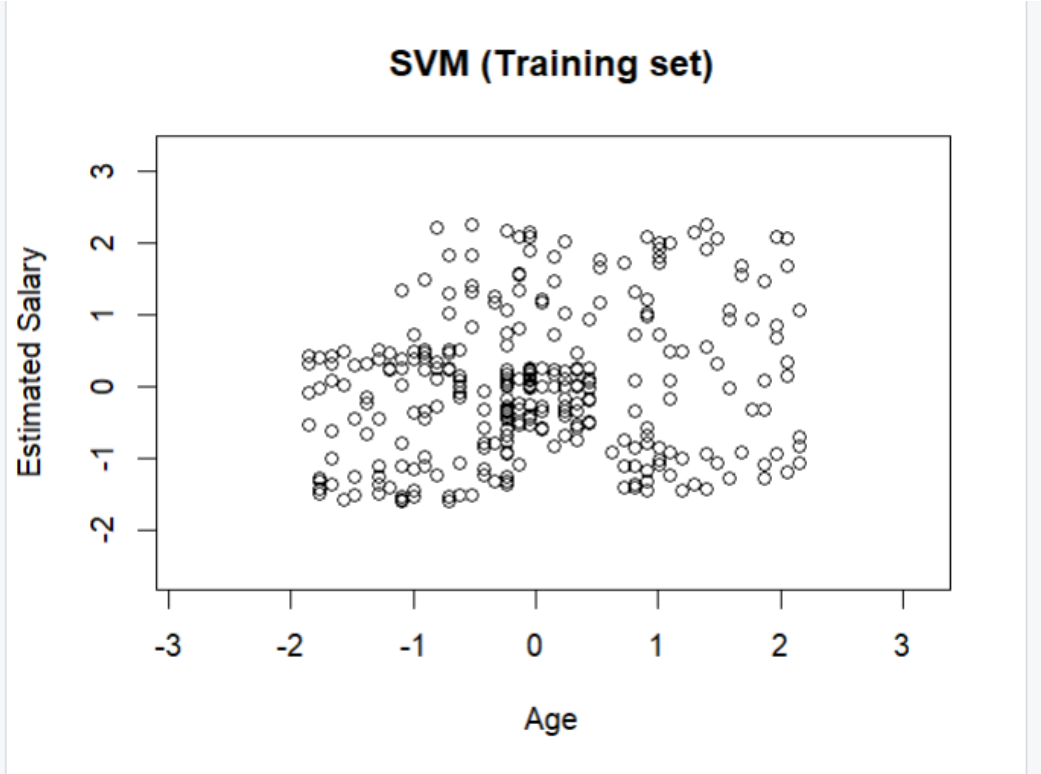
y_grid = predict(classifier, newdata = grid_set)

plot(set[, -3],

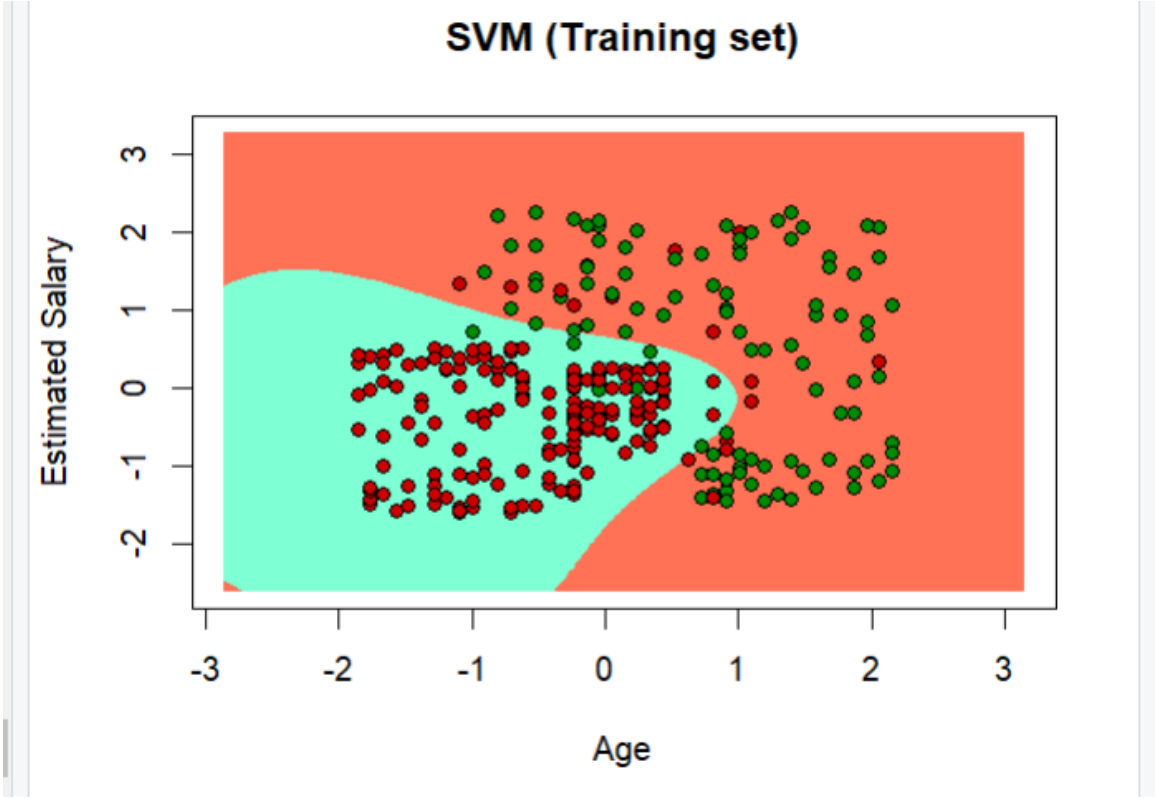
main = 'SVM (Training set)',

xlab = 'Age', ylab = 'Estimated Salary',

xlim = range(X1), ylim = range(X2))




```
contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add = TRUE)
  points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'coral1', 'aquamarine'))
  points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))
```


SVM (Training set)
Estimated Salary
Age
-3
-2
-1
0
1
2
3

Practical 3:

Aim: write a program in R of Naive baye's theorem.

Requirement:
R tool

Code:

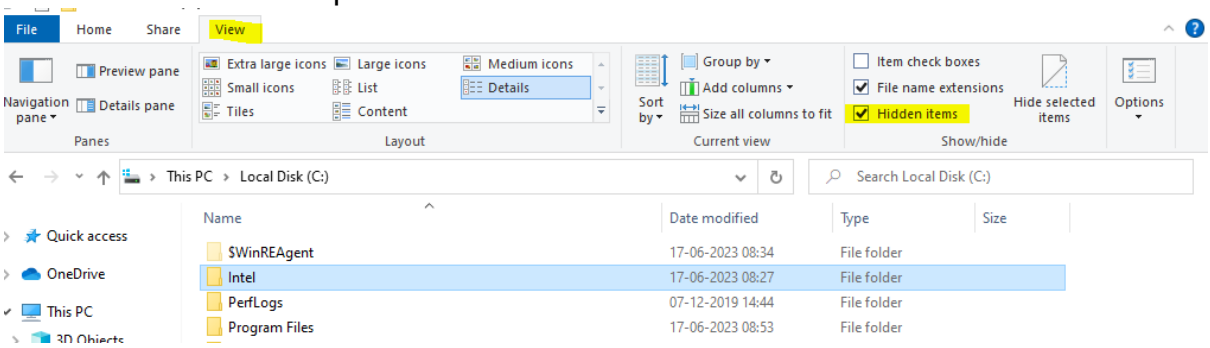
```
data(iris)

str(iris)


install packages("e1071")

install packages("caTools")

install packages("caret")


library(e1071)

library(caTools)

library(caret)


split <- sample.split(iris,SplitRatio=0.7)

train_c1 <-subset(iris,split=="TRUE")

test_c1 <- subset(iris,split == "FALSE")

train_scale <- scale(train_c1[, 1:4])

test_scale <- scale(test_c1[,1:4])


set.seed(120)

classifier_c1 <- naiveBayes(Species ~ ., data = train_c1)

classifier_c1


y_pred <- predict(classifier_c1, newdata= test_c1)

cm <- table(test_c1$Species, y_pred)

cm


confusionMatrix(cm)
```

Output:

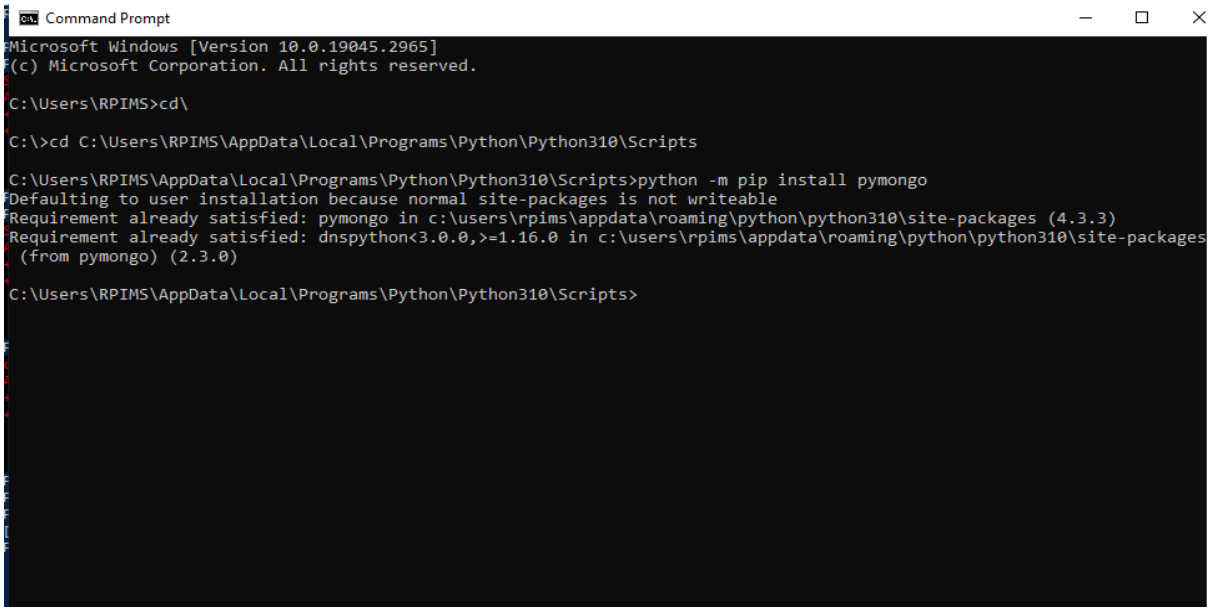
```
R Console
  virginica       0          1         19

Overall Statistics

               Accuracy : 0.9333
                 95% CI : (0.838, 0.9815)
    No Information Rate : 0.3667
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: setosa Class: versicolor Class: virginica
Sensitivity                 1.0000            0.9444           0.8636
Specificity                 1.0000            0.9286           0.9737
Pos Pred Value              1.0000            0.8500           0.9500
Neg Pred Value              1.0000            0.9750           0.9250
Prevalence                  0.3333            0.3000           0.3667
Detection Rate              0.3333            0.2833           0.3167
Detection Prevalence        0.3333            0.3333           0.3333
Balanced Accuracy           1.0000            0.9365           0.9187
>
```

Install python package:

1. You will need to make the hidden folder visible: go to “C:” drive on top click on tab “view”
2. Select “hidden Items” option:



3. Go to the below path: C:\Users*Your Name*\AppData\Local\Programs\Python\Python36-32\Scripts
4. Set the below path in command prompt and then use the below command: python -m pip install pymongo

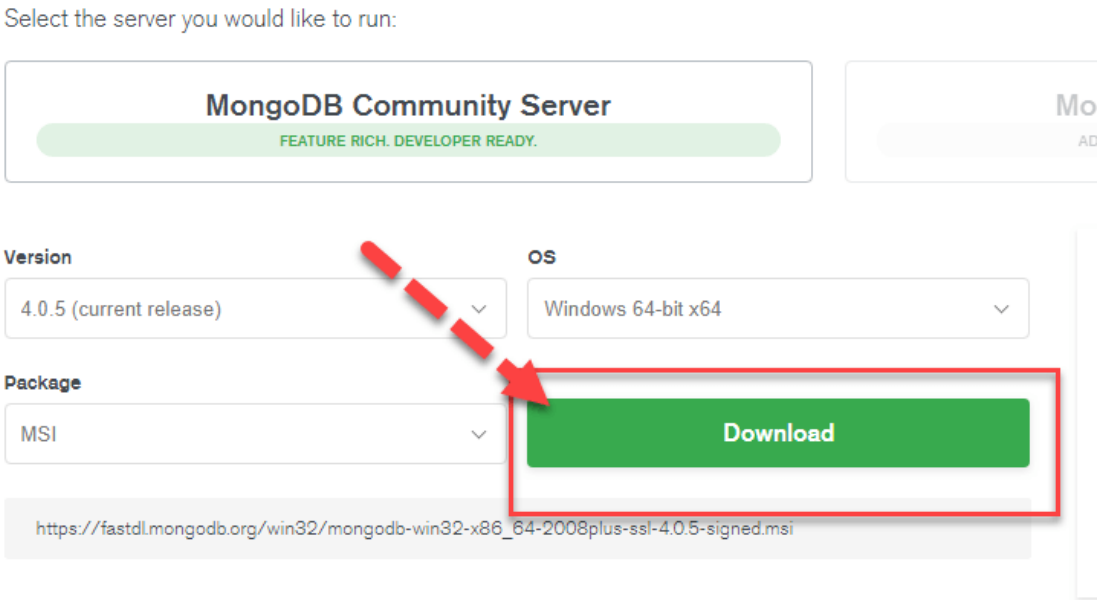


Practical :4

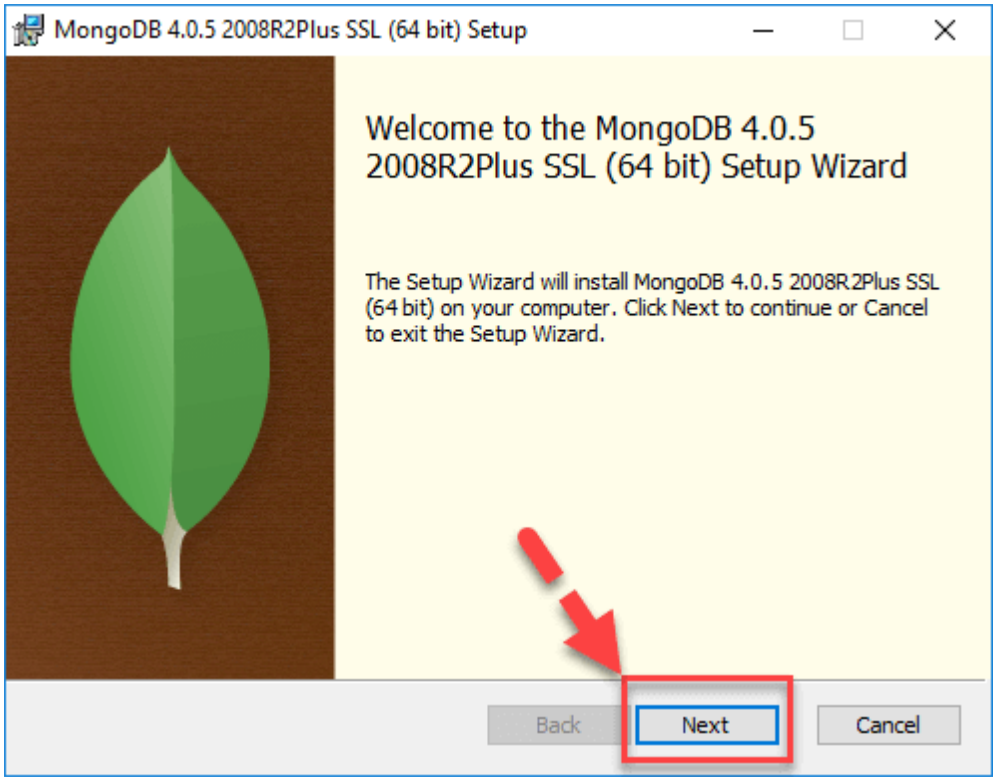
Aim: Implement an application that stores big data in Hbase / MongoDB and manipulate it using R / Python

Requirement:
a. Python Package: PyMongo
b. Mongo Database

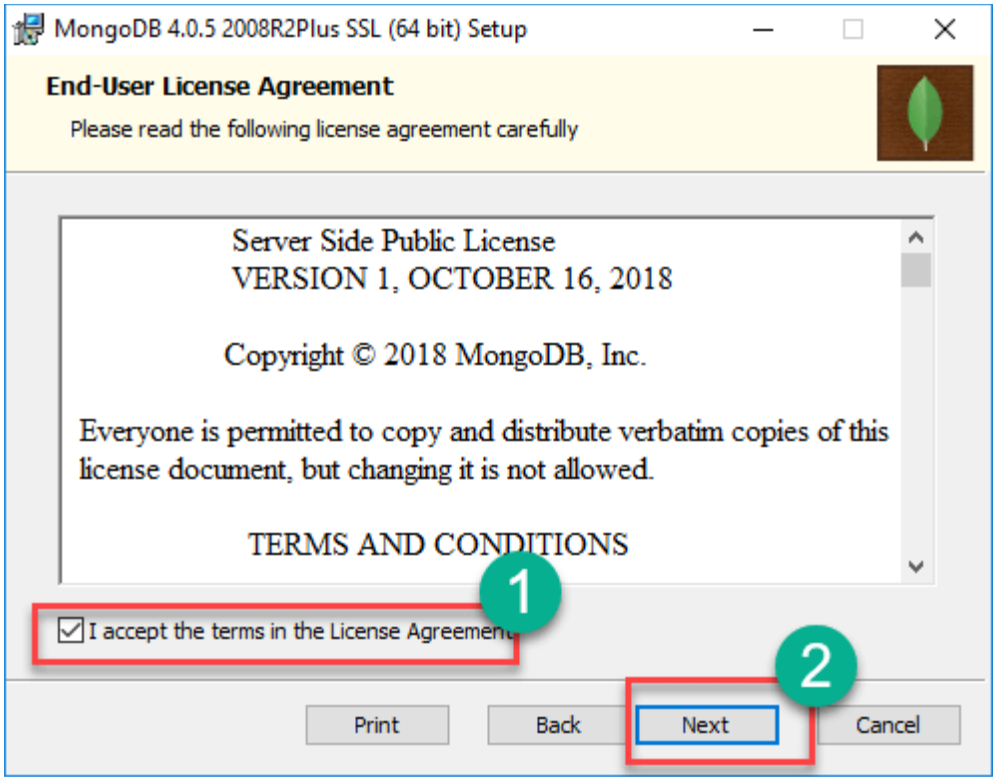
Step A: Install Mongo database
Step 1) Go to (https://www.mongodb.com/download-center/community) and Download MongoDB Community Server. We will install the 64-bit version for Windows.



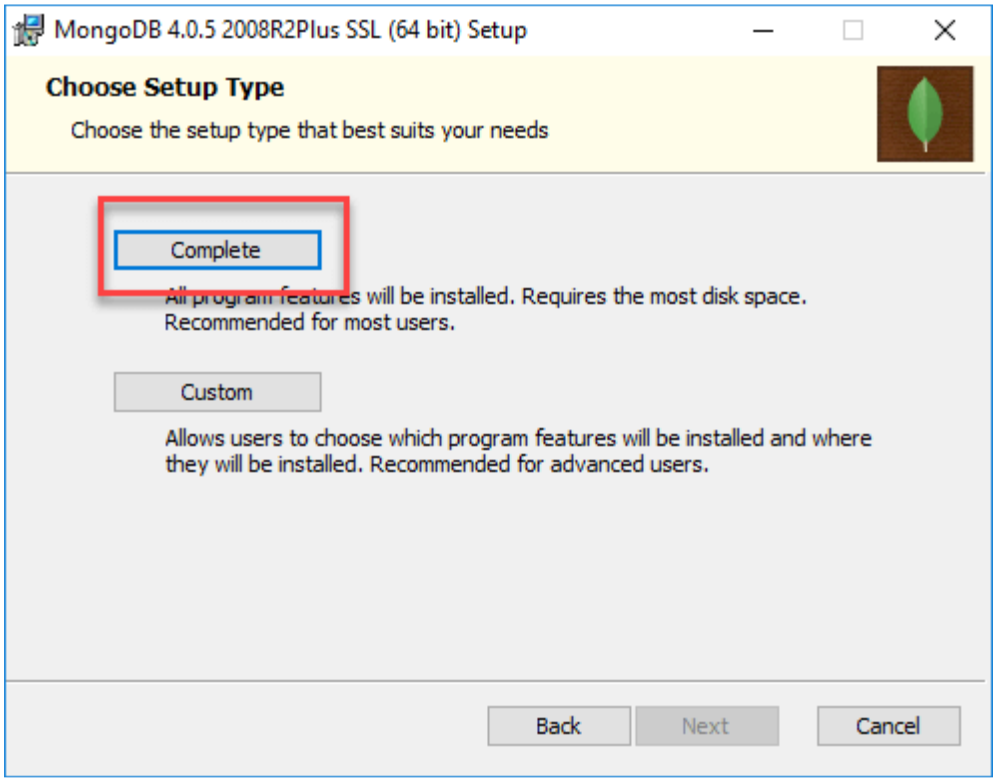
Step 2) Once download is complete open the msi file. Click Next in the start up screen



Step 3)
1. Accept the End-User License Agreement
2. Click Next

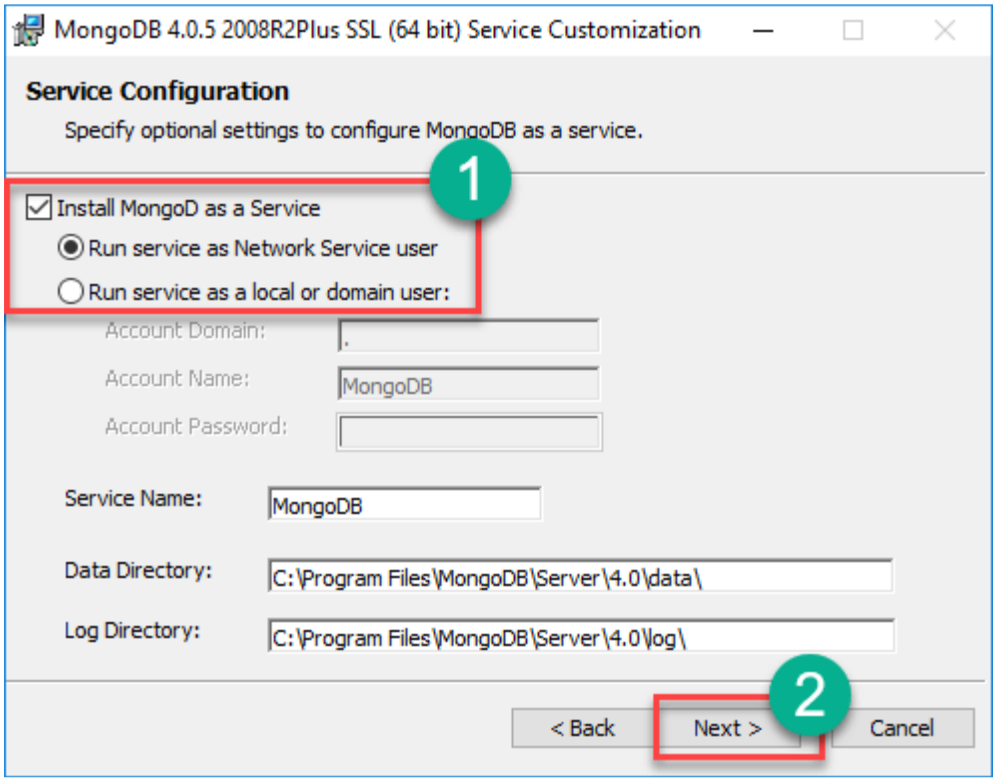


Step 4) Click on the "complete" button to install all of the components. The custom option can be used to install selective components or if you want to change the location of the installation.

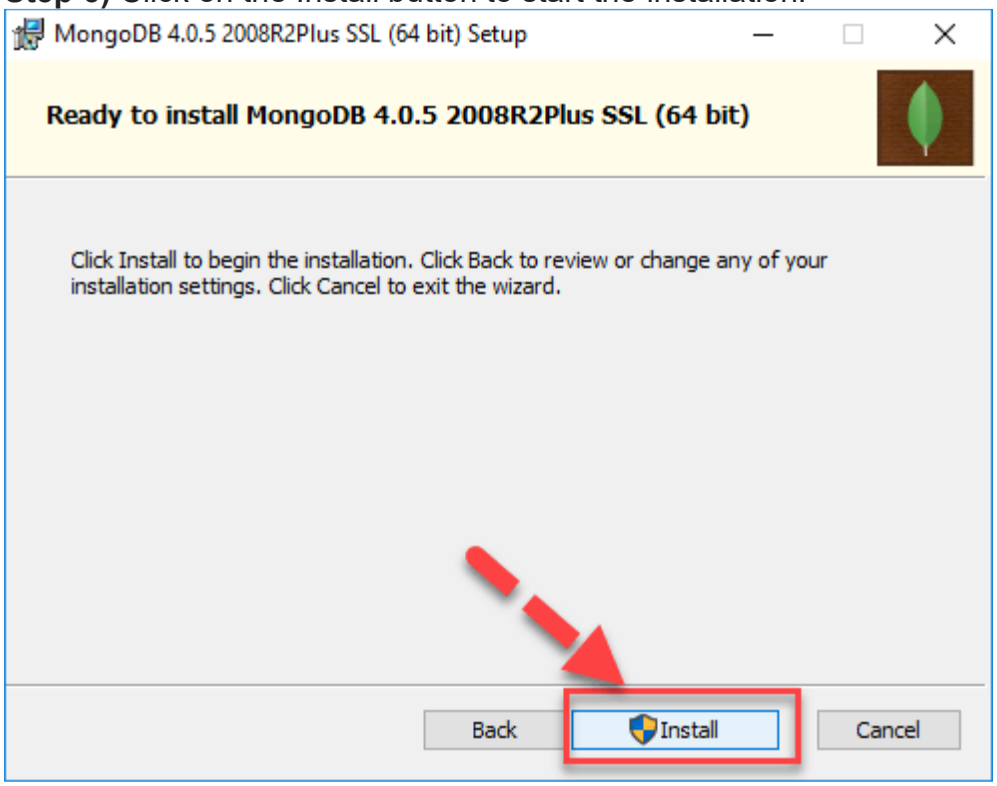


Step 5)

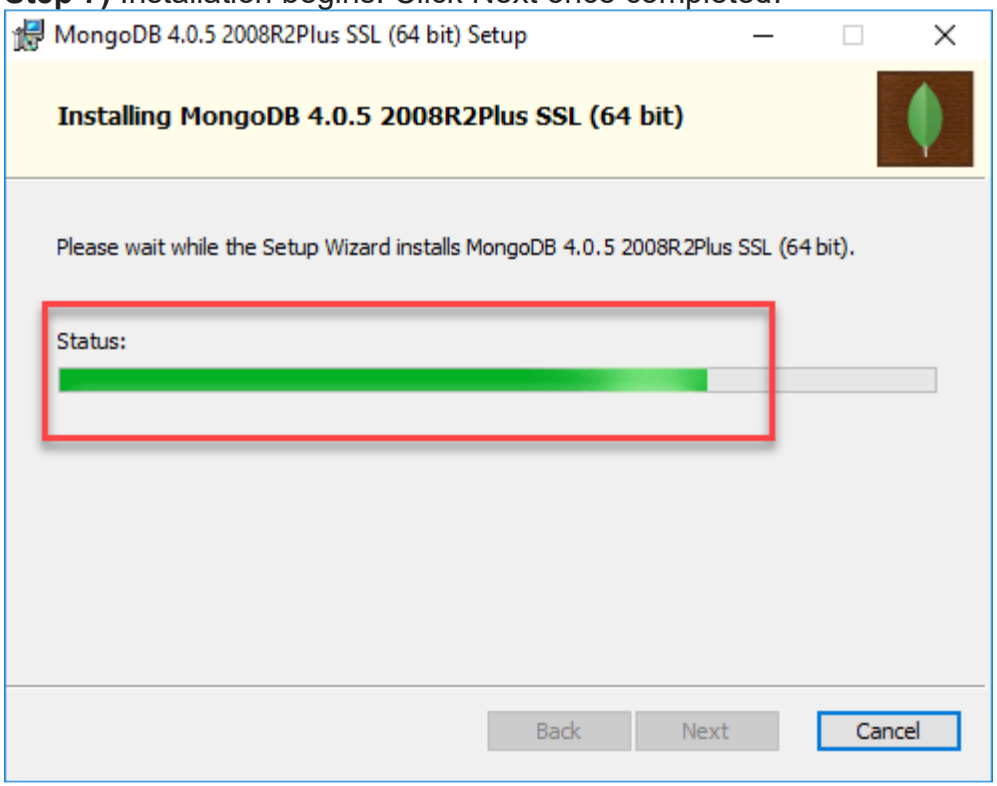
1. Select “Run service as Network Service user”. make a note of the data directory, we’ll need this later.
2. Click Next



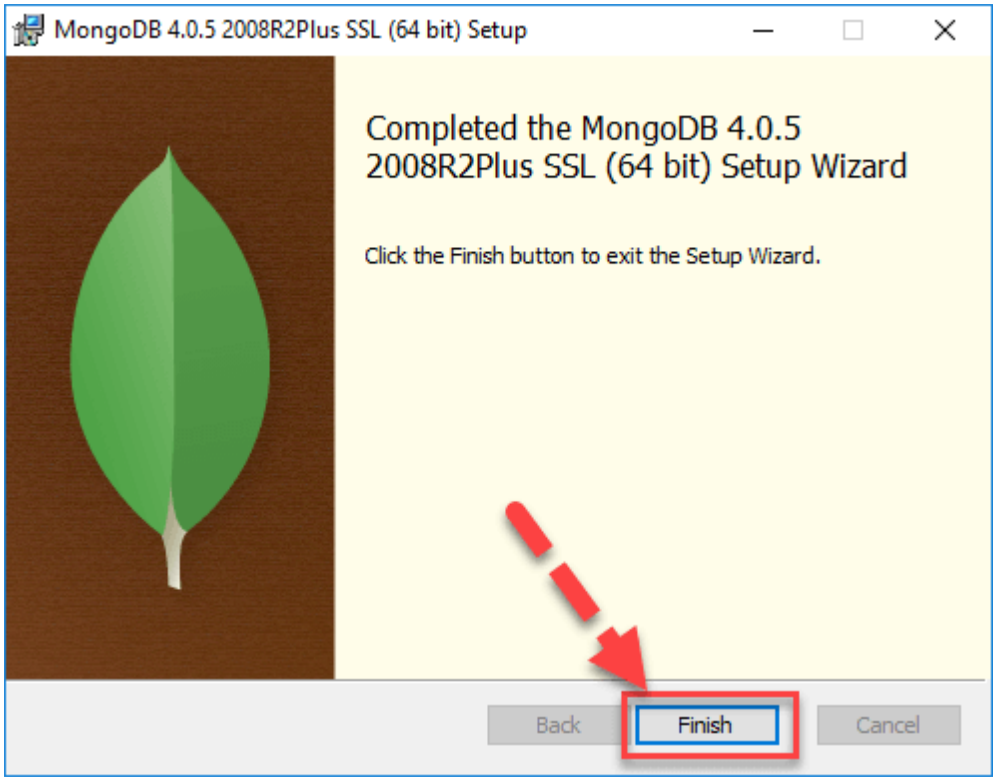
Step 6) Click on the Install button to start the installation.



Step 7) Installation begins. Click Next once completed.



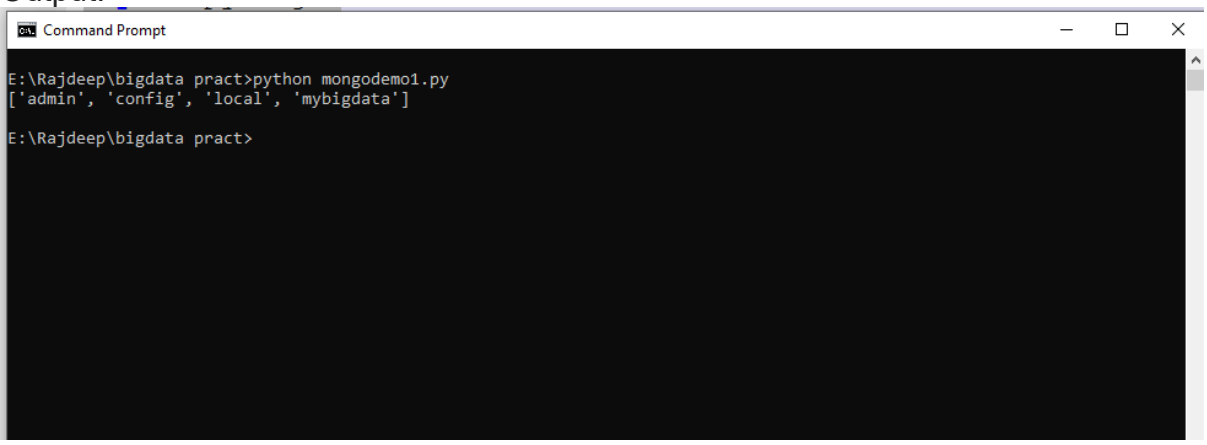
Step 8) Click on the Finish button to complete the installation



Program 1: Displaying the database name:

```
import pymongo
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mybigdata"]
print(myclient.list_database_names())
```

Output:



Program 2: Creating collection:

```
import pymongo
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mybigdata"]
mycol=mydb["student"]
print(mydb.list_collection_names())
```

Output:

```
Command Prompt

E:\Rajdeep\bigdata pract>python mongodemo2.py
['student']

E:\Rajdeep\bigdata pract>
```

Program 3: Inserting Data

```
import pymongo
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mybigdata"]
mycol=mydb["student"]
mydict={"name":"vai", "address":"bhy"}
x=mycol.insert_one(mydict)
print("Data inserted !")
```

Output:

```
Command Prompt

E:\Rajdeep\bigdata pract>python mongodemo3.py
Data inserted !

E:\Rajdeep\bigdata pract>
```

Program 4: Insert Multiple data into Collection

```
import pymongo
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mybigdata"]
mycol=mydb["student"]
mylist=[{"name":"Ganesh", "address":"Mumbai"}, {"name":"Varun",
"address":"Mumbai"},
{"name":"Prasoon", "address":"Pune"}, {"name":"Satish", "address":"Pune"},]
x=mycol.insert_many(mylist)
print("Data inserted !")
```

Output:

```
Command Prompt

E:\Rajdeep\bigdata pract>python mongodemo5.py

E:\Rajdeep\bigdata pract>
```

Program 5: Displaying the collection data:

```
import pymongo
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mybigdata"]
mycol = mydb["student"]

myquery = { "name": "Vai" }

mydoc = mycol.find(myquery)

for x in mydoc:
  print(x)
```

Output:

```
Command Prompt

E:\Rajdeep\bigdata pract>python mongodemo4.py
{'_id': ObjectId('648e7643d46bff955ea3f978'), 'name': 'vai', 'address': 'bhy'}
{'_id': ObjectId('648e7652d8e7314fba232ad8'), 'name': 'vai', 'address': 'bhy'}
{'_id': ObjectId('648e7672e703e178e6aa8ec7'), 'name': 'vai', 'address': 'bhy'}

E:\Rajdeep\bigdata pract>
```

Practical 5:

K means clustering.

Aim: Read a datafile grades_km_input.csv and apply k-means clustering.

Requirement:

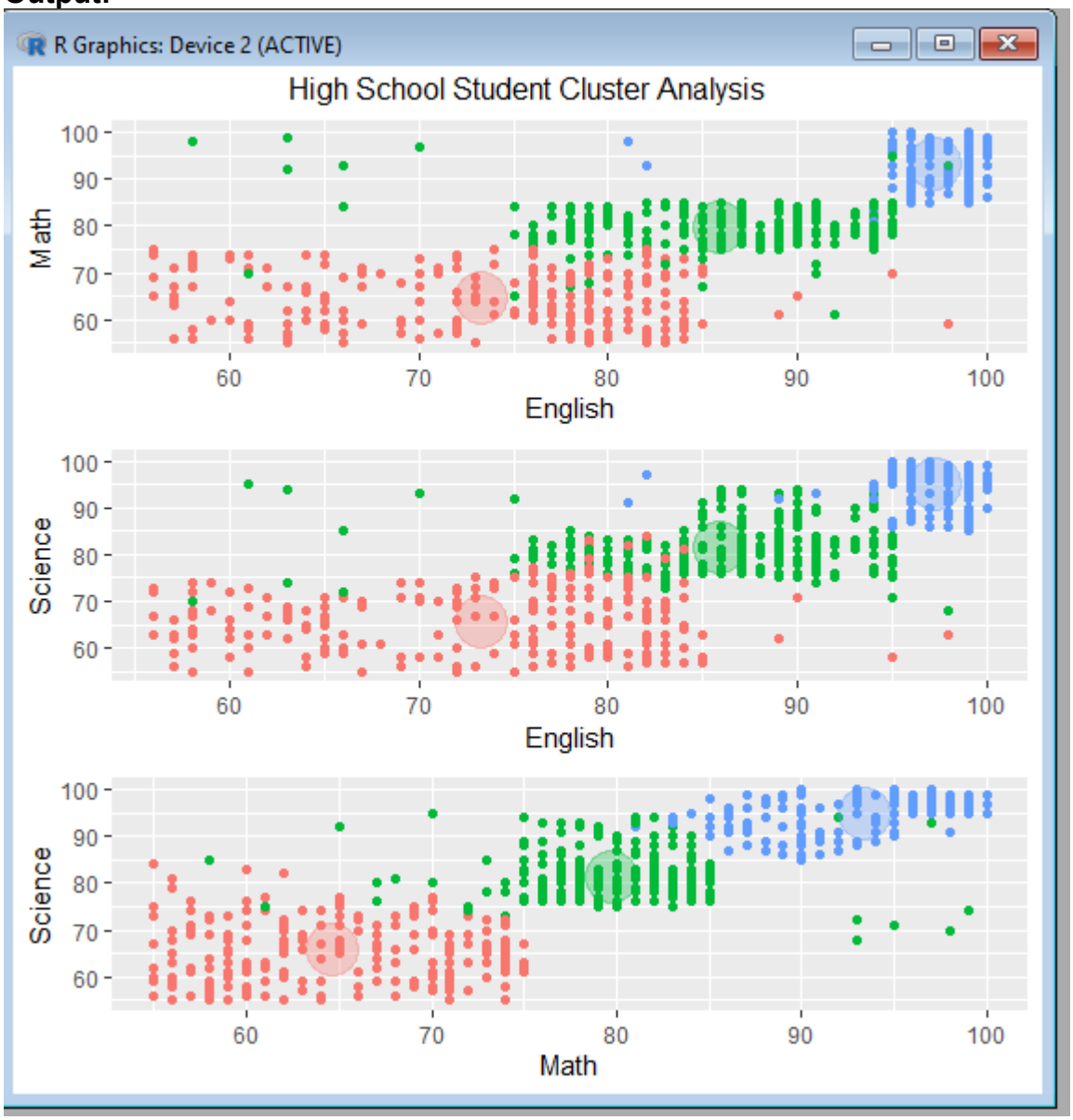
R tool

Code:

```
install.packages("plyr")
install.packages("ggplot2")
install.packages("cluster")
install.packages("lattice")
install.packages("grid")
install.packages("gridExtra")

library(plyr)
library(ggplot2)
library(cluster)
library(lattice)
library(grid)
library(gridExtra)

grade_input=as.data.frame(read.csv("E:/Rajdeep/bigdata
pract/dataset/grades_km_input.csv"))

kmdata_orig=as.matrix(grade_input[, c ("Student","English","Math","Science")])
kmdata=kmdata_orig[,2:4]
kmdata[1:10,]
wss=numeric(15)

for(k in 1:15)wss[k]=sum(kmeans(kmdata,centers=k,nstart=25)$withinss)
plot(1:15,wss,type="b",xlab="Number of Clusters",ylab="Within sum of square")
km = kmeans(kmdata,3,nstart=25)
km

c( wss[3] , sum(km$withinss))
df=as.data.frame(kmdata_orig[,2:4])
df$cluster=factor(km$cluster)
centers=as.data.frame(km$centers)

g1=ggplot(data=df, aes(x=English, y=Math, color=cluster )) +
geom_point() + theme(legend.position="right") +
geom_point(data=centers,aes(x=English,y=Math, color=as.factor(c(1,2,3))),size=10,
alpha=.3, show.legend =FALSE)

g2=ggplot(data=df, aes(x=English, y=Science, color=cluster )) +
geom_point () +geom_point(data=centers,aes(x=English,y=Science,
color=as.factor(c(1,2,3))),size=10, alpha=.3, show.legend=FALSE)

g3 = ggplot(data=df, aes(x=Math, y=Science, color=cluster )) +
```

```
geom_point () + geom_point(data=centers,aes(x=Math,y=Science,
color=as.factor(c(1,2,3))),size=10, alpha=.3, show.legend=FALSE)

tmp=ggplot_gtable(ggplot_build(g1))
grid.arrange(arrangeGrob(g1 + theme(legend.position="none"),g2 +
theme(legend.position="none"),g3 + theme(legend.position="none"),top ="High
School Student Cluster Analysis" ,ncol=1))
```

Output:



Practical 6:

a. Simple Linear regression

Aim: Create your own data for years of experience and salary in lakhs and apply linear regression model to predict the salary

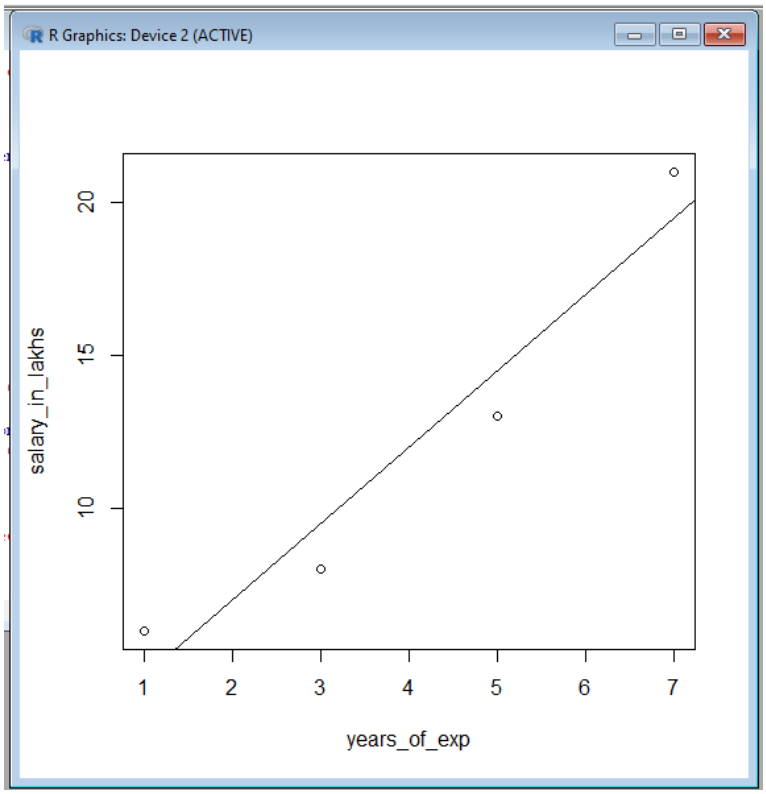
Requirement:

R tool

Code:

```
years_of_exp = c(7,5,1,3)
salary_in_lakhs = c(21,13,6,8)
employee.data = data.frame(years_of_exp, salary_in_lakhs)
employee.data

model <- lm(salary_in_lakhs ~ years_of_exp, data = employee.data)
summary(model)

plot(salary_in_lakhs ~ years_of_exp, data = employee.data)
abline(model)
```

Output:



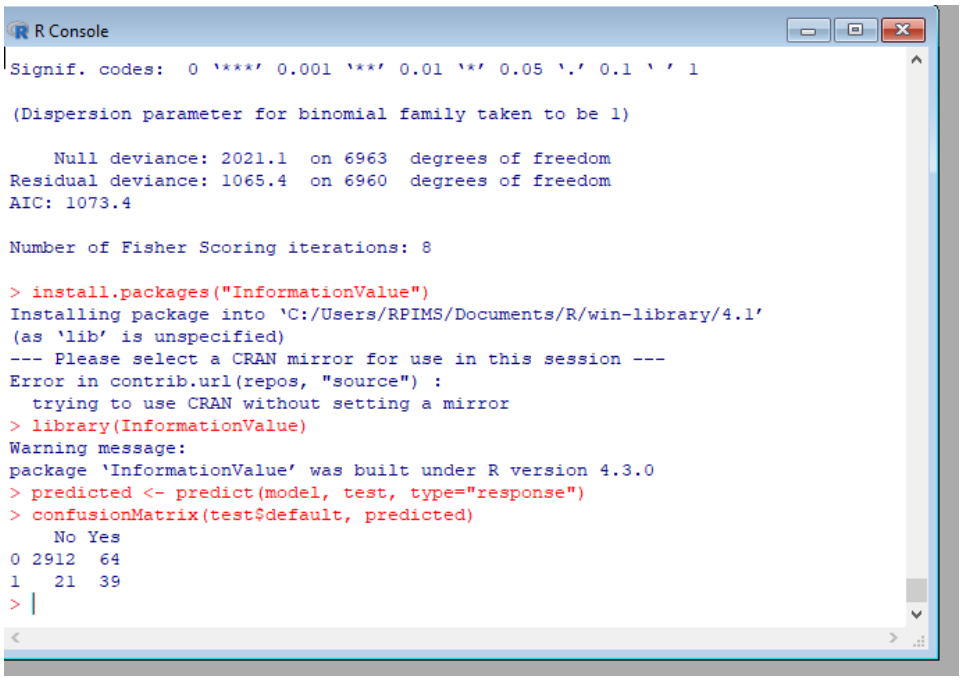
b.: Logistic regression:

Aim: Take the in-built data from ISLR package and apply generalized logistic regression to find whether a person would be defaulter or not; considering input as student, income and balance.

Code:

```
install.packages("ISLR")
library(ISLR)
data <- ISLR::Default
print (head(ISLR::Default))
summary(data)
nrow(data)
set.seed(1)
sample <- sample(c(TRUE, FALSE), nrow(data), replace=TRUE, prob=c(0.7,0.3))
print (sample)
train <- data[sample, ]
test <- data[!sample, ]
nrow(train)
nrow(test)
model <- glm(default~student+balance+income, family="binomial", data=train)
summary(model)
install.packages("InformationValue")
library(InformationValue)
predicted <- predict(model, test, type="response")
confusionMatrix(test$default, predicted)
```

Output:



```
R Console
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2021.1  on 6963  degrees of freedom
Residual deviance: 1065.4  on 6960  degrees of freedom
AIC: 1073.4

Number of Fisher Scoring iterations: 8

> install.packages("InformationValue")
Installing package into 'C:/Users/RPIMS/Documents/R/win-library/4.1'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
Error in contrib.url(repos, "source") :
  trying to use CRAN without setting a mirror
> library(InformationValue)
Warning message:
package 'InformationValue' was built under R version 4.3.0
> predicted <- predict(model, test, type="response")
> confusionMatrix(test$default, predicted)
    No Yes
0 2912  64
1   21  39
>
```

Practical 7:

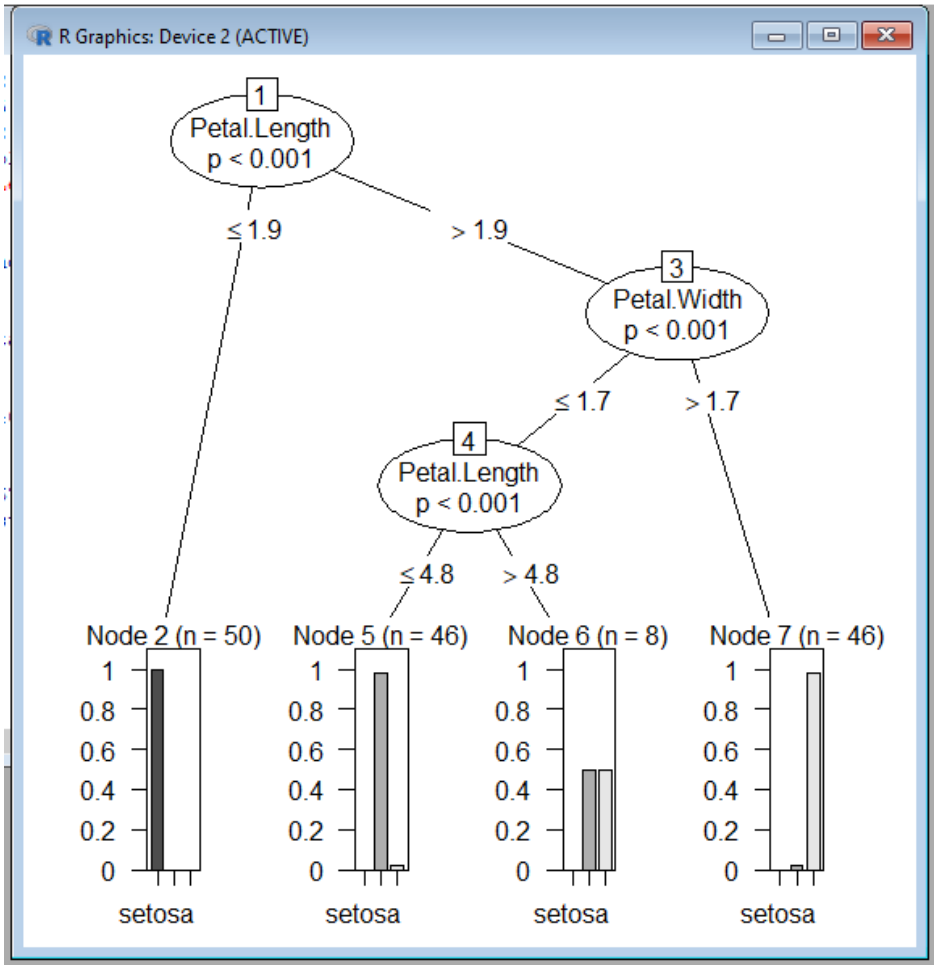
Aim: Implement Decision tree classification techniques.

Requirement:
R tool

Code:

```
library("party")
print(head(readingSkills))

str(iris)
iris_ctree <- ctree(Species ~ Sepal.Width + Sepal.Length + Petal.Length +
Petal.Width, data=iris

print (iris_ctree)
plot(iris_ctree)
```

Output:



Practical 8:

Apriori algorithm

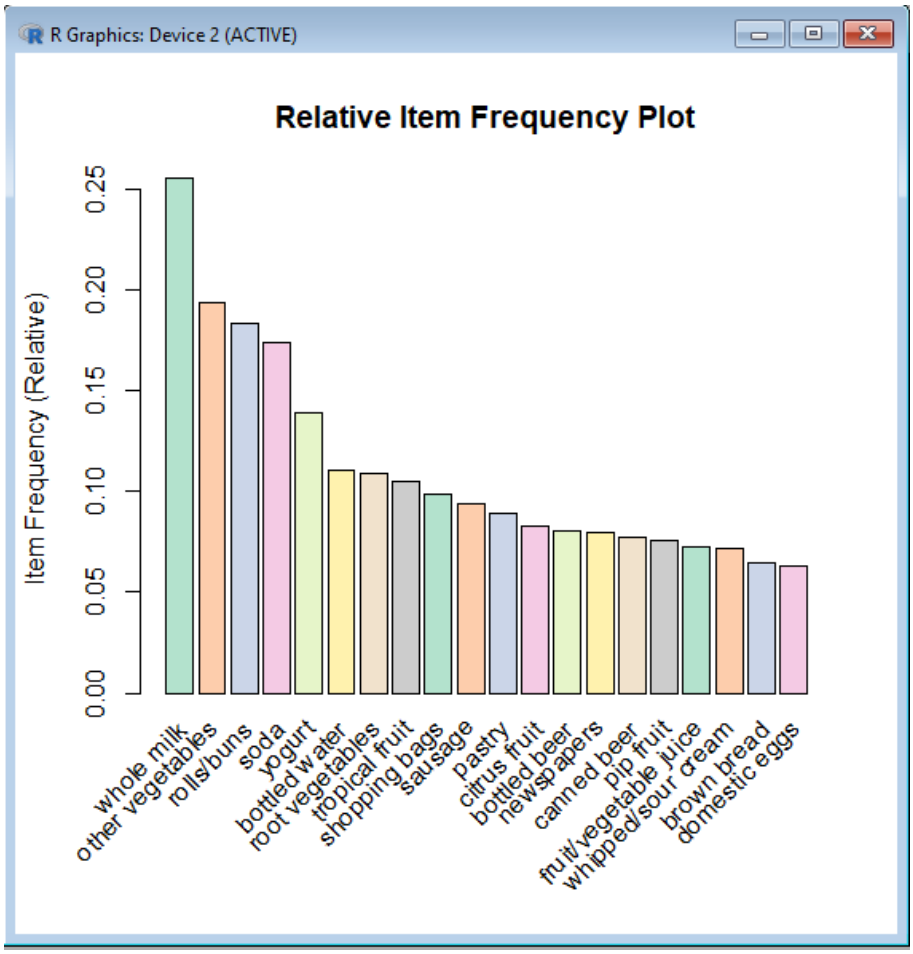
Aim: Perform Apriori algorithm using Groceries dataset from the R arules package.

Requirement:
R tool

Code:

```
library(arules)
library(arulesViz)
library(RColorBrewer)

data(Groceries)
Groceries

summary(Groceries)
class(Groceries)

rules = apriori(Groceries, parameter = list(supp = 0.02, conf = 0.2))
summary (rules)

inspect(rules[1:10])

arules::itemFrequencyPlot(Groceries, topN = 20,
col = brewer.pal(8, 'Pastel2'),
main = 'Relative Item Frequency Plot',
type = "relative",
ylab = "Item Frequency (Relative)")

itemsets = apriori(Groceries, parameter = list(minlen=2, maxlen=2,support=0.02,
target="frequent itemsets"))
summary(itemsets)
inspect(itemsets)
itemsets_3 = apriori(Groceries, parameter = list(minlen=3, maxlen=3,support=0.02,
target="frequent itemsets"))
summary(itemsets_3)


inspect(itemsets_3)
```

Output:




```
R Console
      3        3        3        3        3        3

summary of quality measures:
    support           count
 Min.   :0.02227   Min.   :219.0
 1st Qu.:0.02250   1st Qu.:221.2
 Median :0.02272   Median :223.5
 Mean   :0.02272   Mean   :223.5
 3rd Qu.:0.02295   3rd Qu.:225.8
 Max.   :0.02318   Max.   :228.0

includes transaction ID lists: FALSE

mining info:
      data ntransactions support confidence
 Groceries          9835    0.02          1
                                                                    $
 apriori(data = Groceries, parameter = list(minlen = 3, maxlen = 3, support = 0$
>
>
> inspect(itemsets_3)
    items                                          support    count
[1] {root vegetables, other vegetables, whole milk} 0.02318251 228
[2] {other vegetables, whole milk, yogurt}          0.02226741 219
>
```