# Installation guide for R and RStudio

## Step 1 – Install R

1. Download the R installer from https://cran.r-project.org/

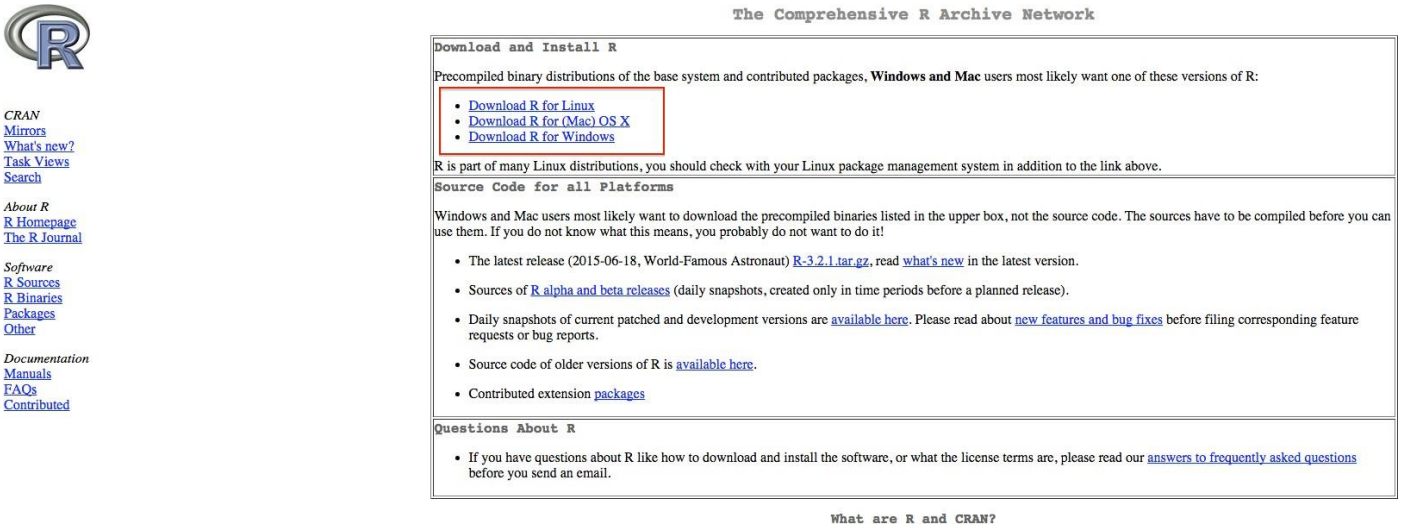



Figure 1. Screenshot of http://cran.csiro.au/

2. Run the installer. Default settings are fine. If you do not have admin rights on your laptop, thenask you local IT support. In that case, it is important that you also ask them to give you full permissions to the R directories. Without this, you will not be able to install additional packages later

## Step 2 – Install RStudio

1. Download RStudio: https://www.rstudio.com/products/rstudio/download/

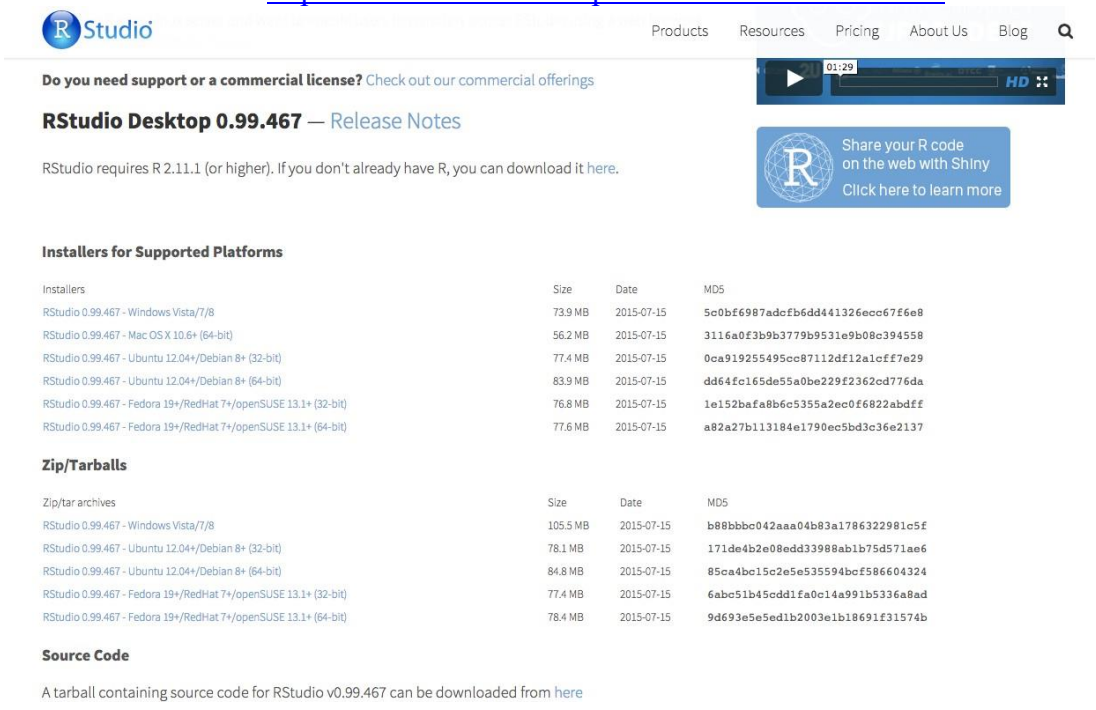



Figure 2. Download RStudio on https://www.rstudio.com/products/rstudio/download/

2. Once the installation of R has completed successfully (and not before), run the RStudio installer.

3. If you do not have administrative rights on your laptop, step 2 may fail. Ask your IT Support or download a pre-built zip archive of RStudio which doesn't need installing. The link for this is towards the bottom of the download page, highlighted in Image 2.

    a. Download the appropriate archive for your system (Windows/Linux only – the Mac version can be installed into your personal "Applications" folder without admin rights).

    b. Double clicking on the zip archive should automatically unpack it on most Windows machines.

## Step 3 – Check that R and RStudio are working

1. Open RStudio. It should open a window that looks similar to image 3 below.

2. In the left hand window, by the '>'sign, type '4+5'(without the quotes) and hit enter. An outputline reading '[1] 9' should appear. This means that R and RStudio are working.

3. If this is not successful, contact us or your local IT support for further advice

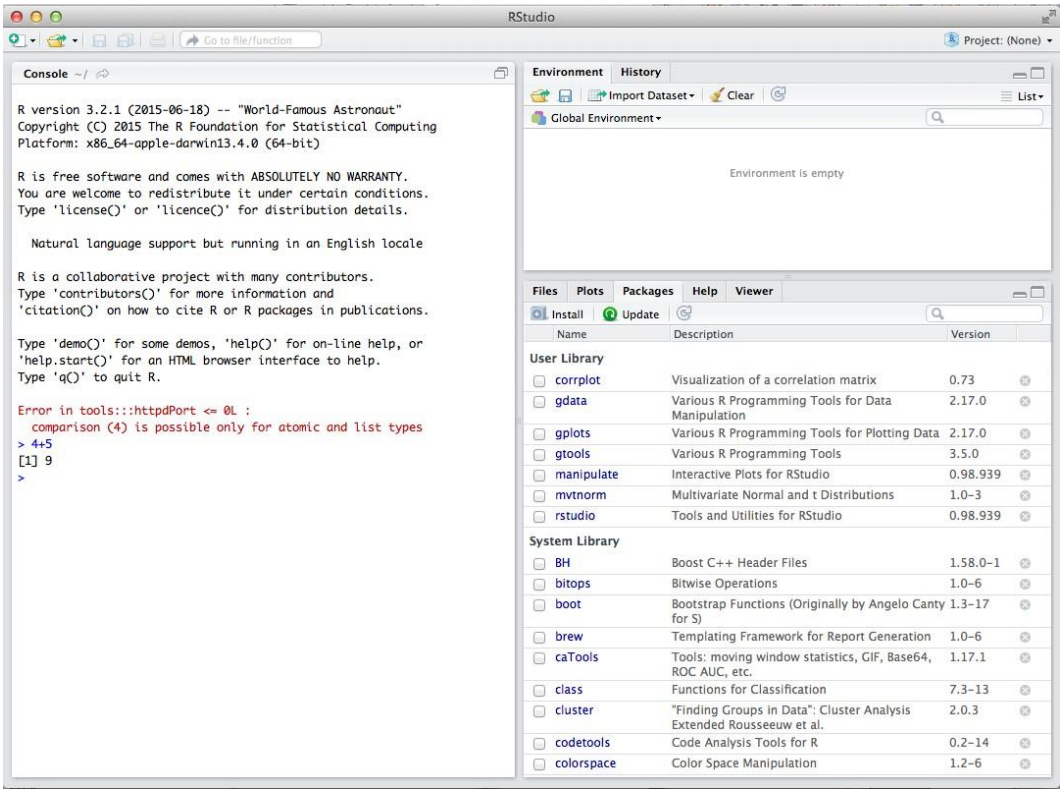


Figure 3. Running R with RStudio

## Step 4 – Install R packages required for the workshop

1. Click on the tab ' Packages' then 'Install' as shown in Image 4. Or Tools --> Install packages.
2. Install the following packages: mixOmics version 6.1.0, mvtnorm, RColorBrewer, corrplot, igraph (see Image 4). For apple mac users, if youare unable to install the mixOmics imported library rgl, you will need to install the XQuartz software first https://www.xquartz.org/

3. Check that the packages are installed by typing ‘library(mixOmics)’ (without thequotes) in the prompt and press enter (see Image 5).
4. Then type ‘sessionInfo()’ and check that mixOmics version 6.1.0 has been installed(image 6).

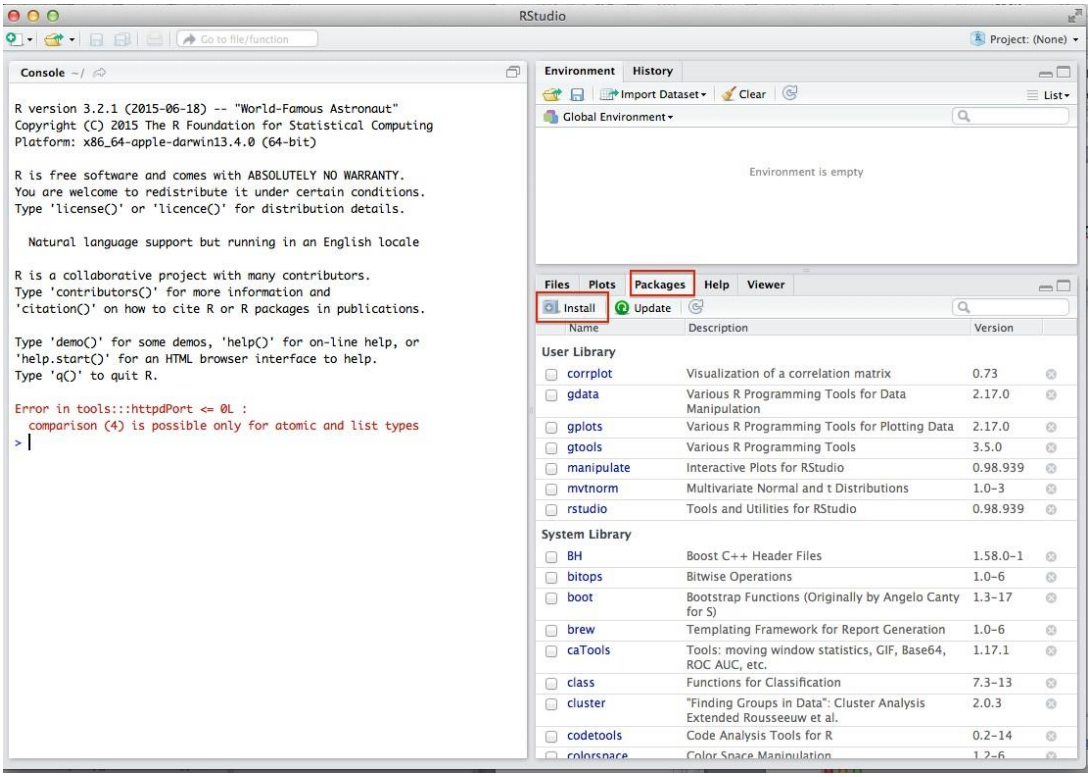



Figure 4. Click on Install to install R packages.

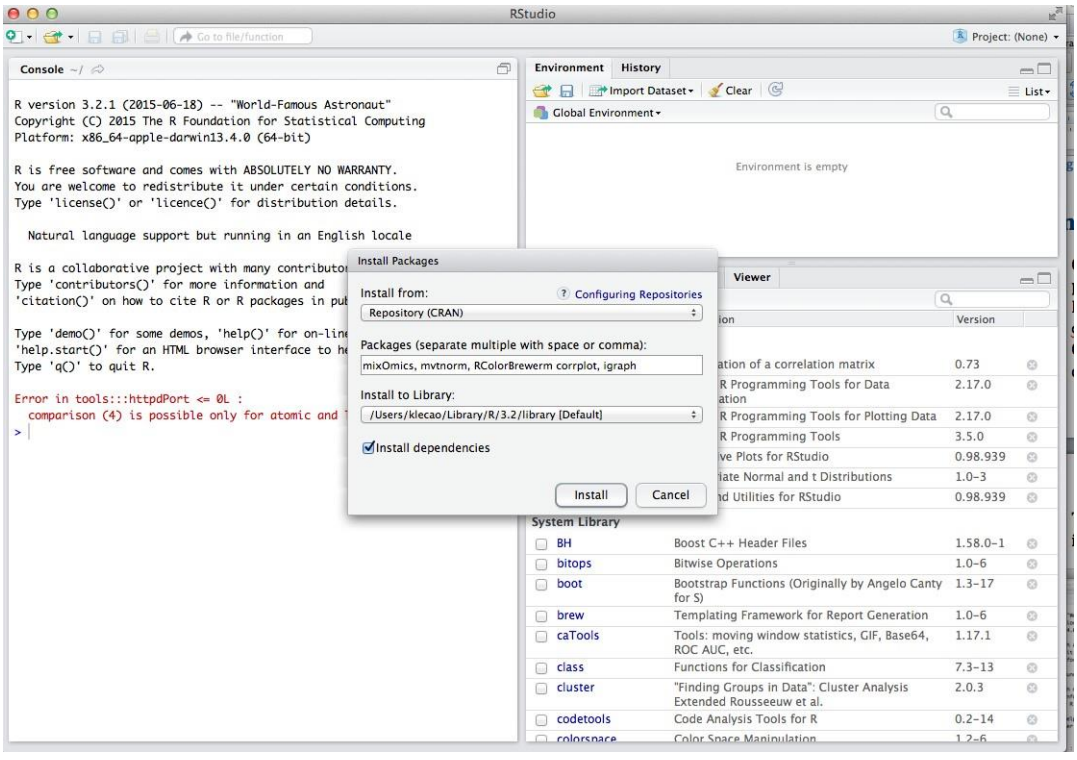



Figure 5. Specify the list of packages to be installed

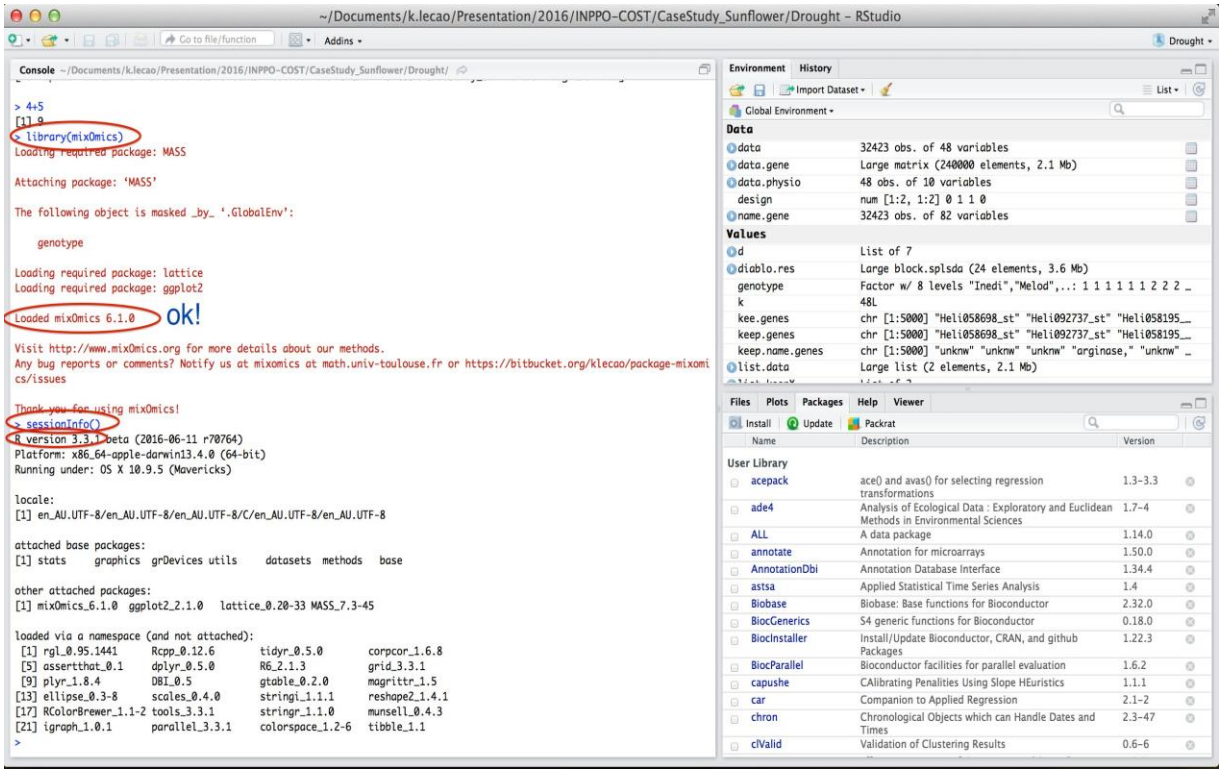


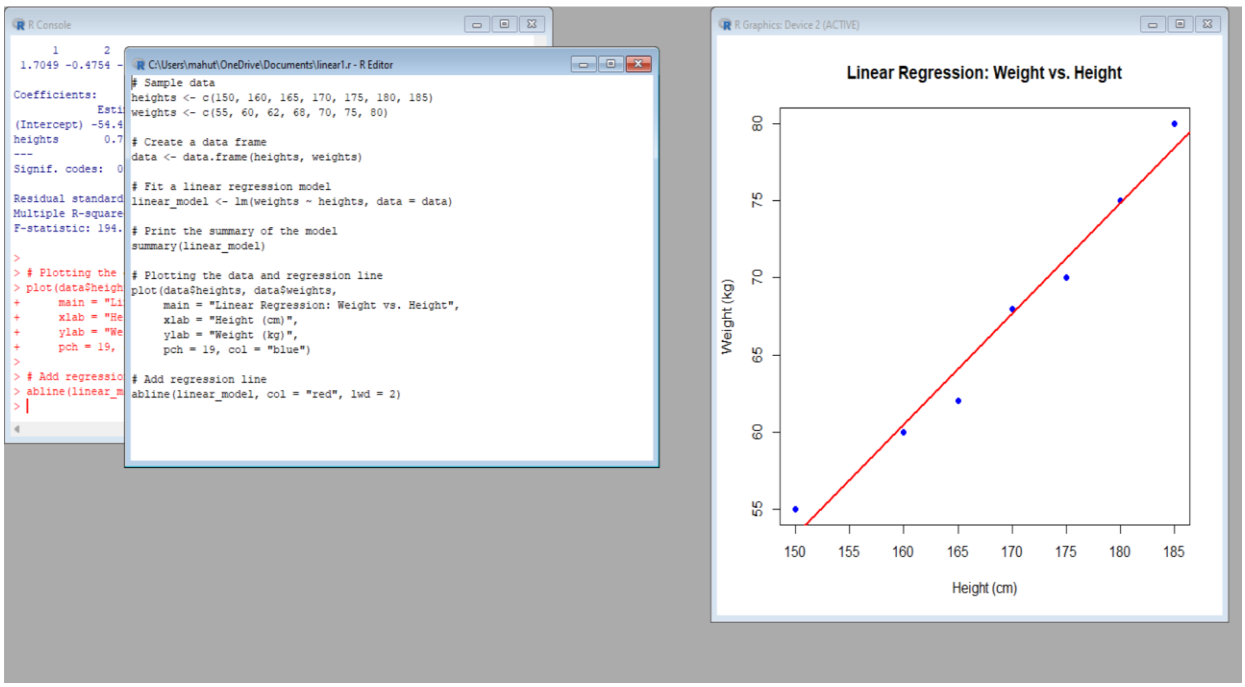

Figure 6. Check that the package mixOmics is installed and has the version 6.1.0.

**Exp:7**

## Implement Linear and Logistic Regressiona

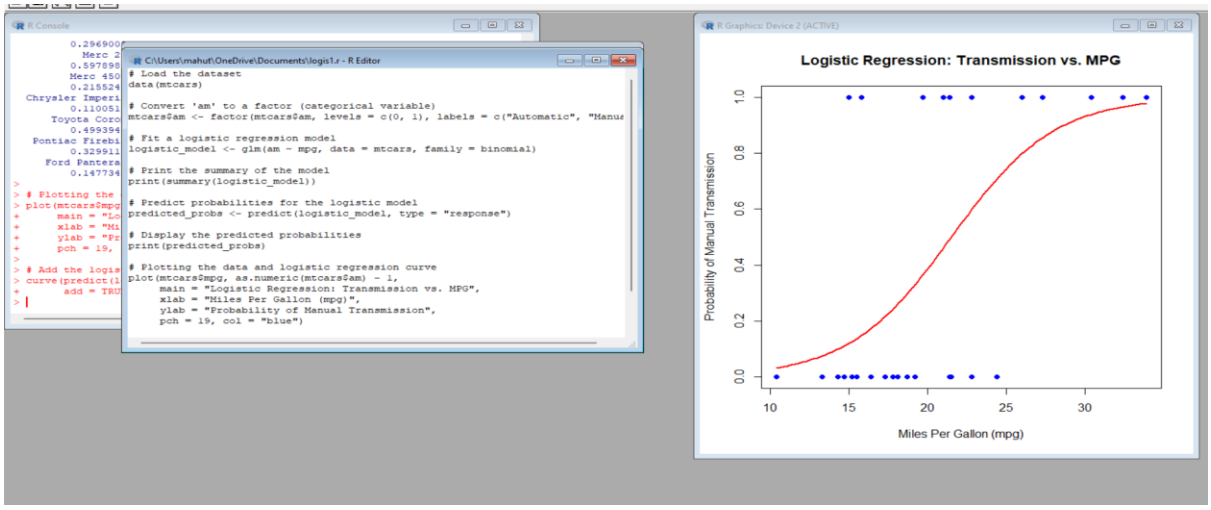
**a)**Linear regression

```
# Sample data
heights <- c(150, 160, 165, 170, 175, 180, 185)
weights <- c(55, 60, 62, 68, 70, 75, 80)

# Create a data frame
data <- data.frame(heights, weights)

# Fit a linear regression model
linear_model <- lm(weights ~ heights, data = data)

# Print the summary of the model
print(summary(linear_model))

# Plotting the data and regression line
plot(data$heights, data$weights,
     main = "Linear Regression: Weight vs. Height",
     xlab = "Height (cm)",
     ylab = "Weight (kg)",
     pch = 19, col = "blue")

# Add regression line
abline(linear_model, col = "red", lwd = 2)
```



**b) Logistic regression**

```
# Load the dataset
data(mtcars)

# Convert 'am' to a factor (categorical variable)
mtcars$am <- factor(mtcars$am, levels = c(0, 1), labels = c("Automatic", "Manual"))

# Fit a logistic regression model
logistic_model <- glm(am ~ mpg, data = mtcars, family = binomial)

# Print the summary of the model
print(summary(logistic_model))

# Predict probabilities for the logistic model
predicted_probs <- predict(logistic_model, type = "response")

# Display the predicted probabilities
print(predicted_probs)

# Plotting the data and logistic regression curve
plot(mtcars$mpg, as.numeric(mtcars$am) - 1,
     main = "Logistic Regression: Transmission vs. MPG",
     xlab = "Miles Per Gallon (mpg)",
     ylab = "Probability of Manual Transmission",
     pch = 19, col = "blue")

# Add the logistic regression curve
curve(predict(logistic_model, data.frame(mpg = x), type = "response"),
      add = TRUE, col = "red", lwd = 2)
```

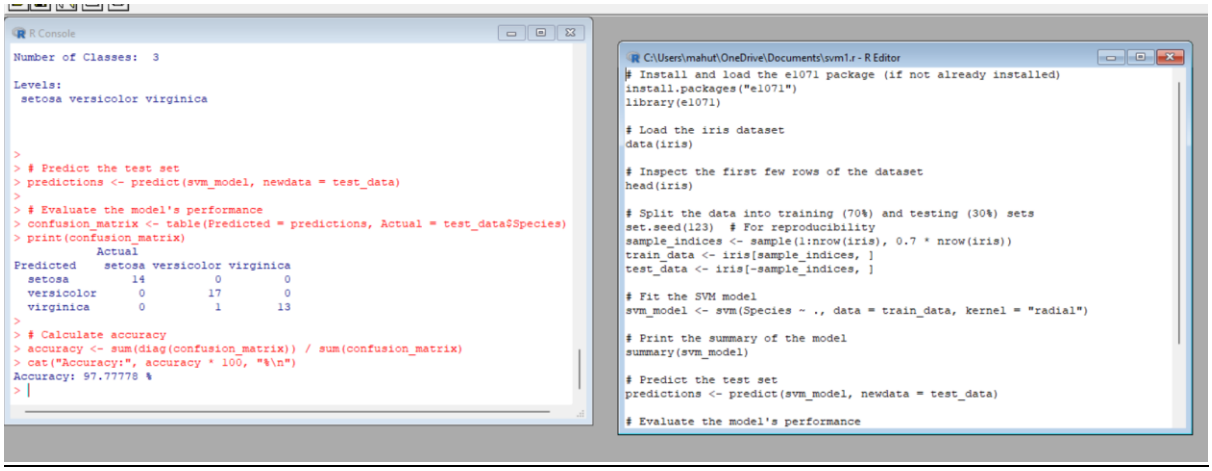


**Exp8:**

# Implement SVM/Decision tree classification techniques

## a) SVM IN R

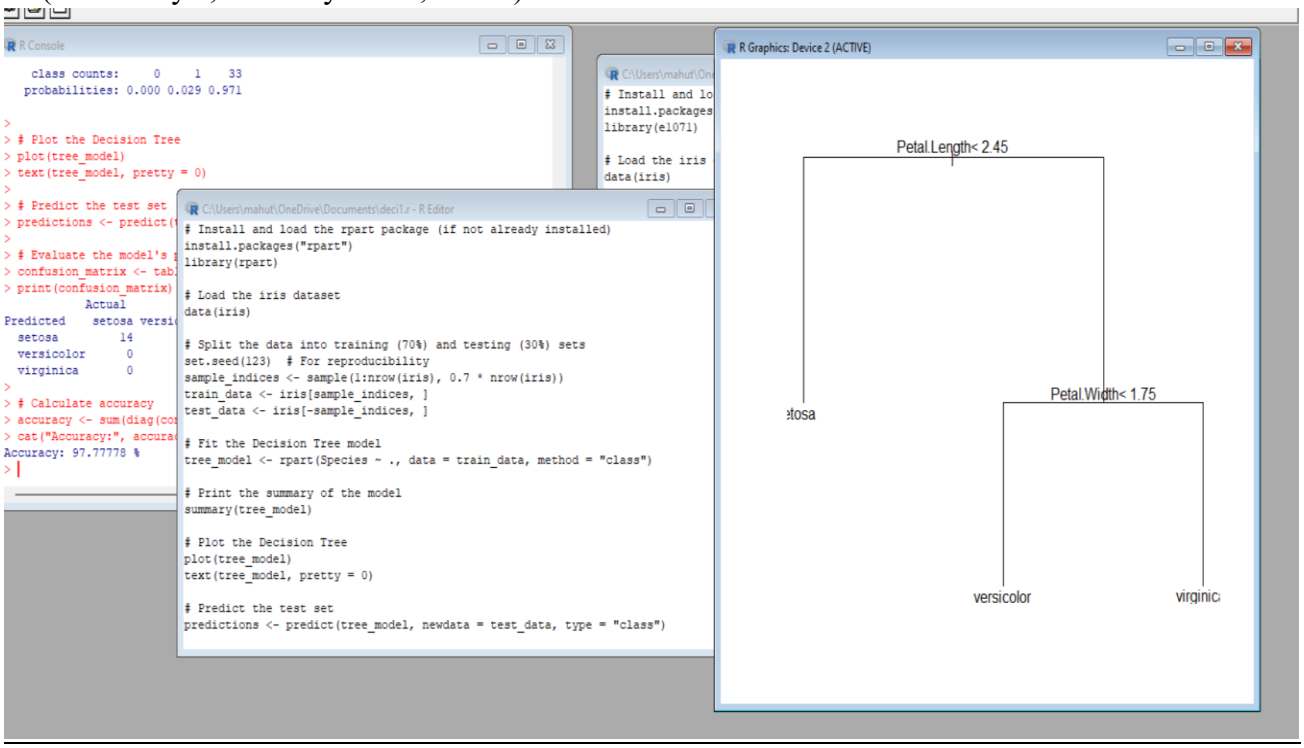
```
# Install and load the e1071 package (if not already installed)
install.packages("e1071")
library(e1071)

# Load the iris dataset
data(iris)

# Inspect the first few rows of the dataset
head(iris)

# Split the data into training (70%) and testing (30%) sets
set.seed(123)  # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_indices, ]
test_data <- iris[-sample_indices, ]

# Fit the SVM model
svm_model <- svm(Species ~ ., data = train_data, kernel = "radial")

# Print the summary of the model
summary(svm_model)

# Predict the test set
predictions <- predict(svm_model, newdata = test_data)

# Evaluate the model's performance
confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
print(confusion_matrix)

# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy:", accuracy * 100, "%\n")
```




**b) Decision tree in R**

```
# Install and load the rpart package (if not already installed)
install.packages("rpart")
library(rpart)

# Load the iris dataset
data(iris)

# Split the data into training (70%) and testing (30%) sets
set.seed(123)  # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_indices, ]
test_data <- iris[-sample_indices, ]

# Fit the Decision Tree model
tree_model <- rpart(Species ~ ., data = train_data, method = "class")

# Print the summary of the model
summary(tree_model)

# Plot the Decision Tree
plot(tree_model)
text(tree_model, pretty = 0)

# Predict the test set
predictions <- predict(tree_model, newdata = test_data, type = "class")

# Evaluate the model's performance
confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
print(confusion_matrix)
```

# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy:", accuracy * 100, "%\n")



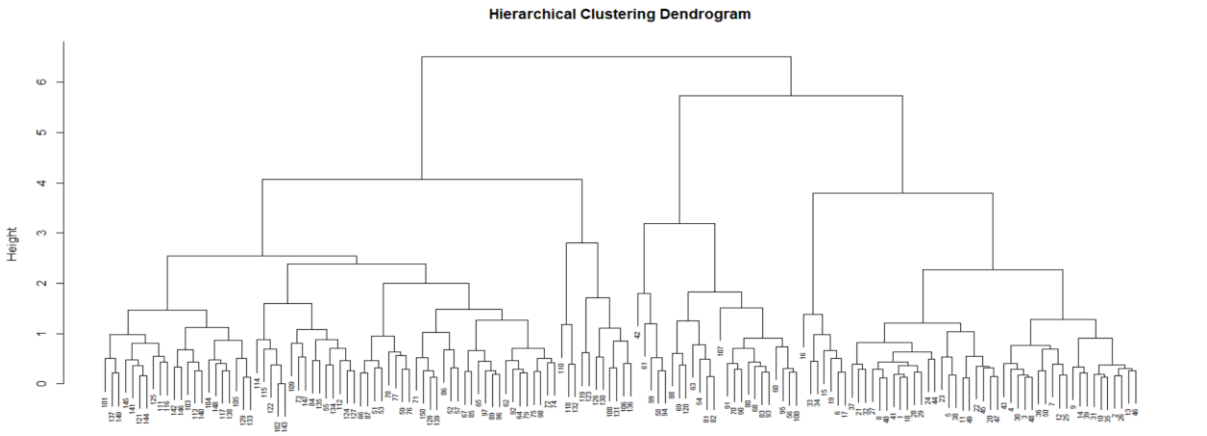
**Exp:9**

## Implement clustering techniques – Hierarchical and K-Means

### a) HIERARCHIAL CLUSTERING

```
# Load the iris dataset
data(iris)

# Use only the numeric columns for clustering (exclude the Species column)
iris_data <- iris[, -5]

# Standardize the data
iris_scaled <- scale(iris_data)

# Compute the distance matrix
distance_matrix <- dist(iris_scaled, method = "euclidean")

# Perform hierarchical clustering using the "complete" linkage method
hc_complete <- hclust(distance_matrix, method = "complete")

# Plot the dendrogram
plot(hc_complete, main = "Hierarchical Clustering Dendrogram", xlab = "", sub = "", cex =
0.6)

# Cut the tree to form 3 clusters
clusters <- cutree(hc_complete, k = 3)

# Print the cluster memberships
print(clusters)

# Add the clusters to the original dataset
iris$Cluster <- as.factor(clusters)

# Display the first few rows of the updated dataset
head(iris)
```

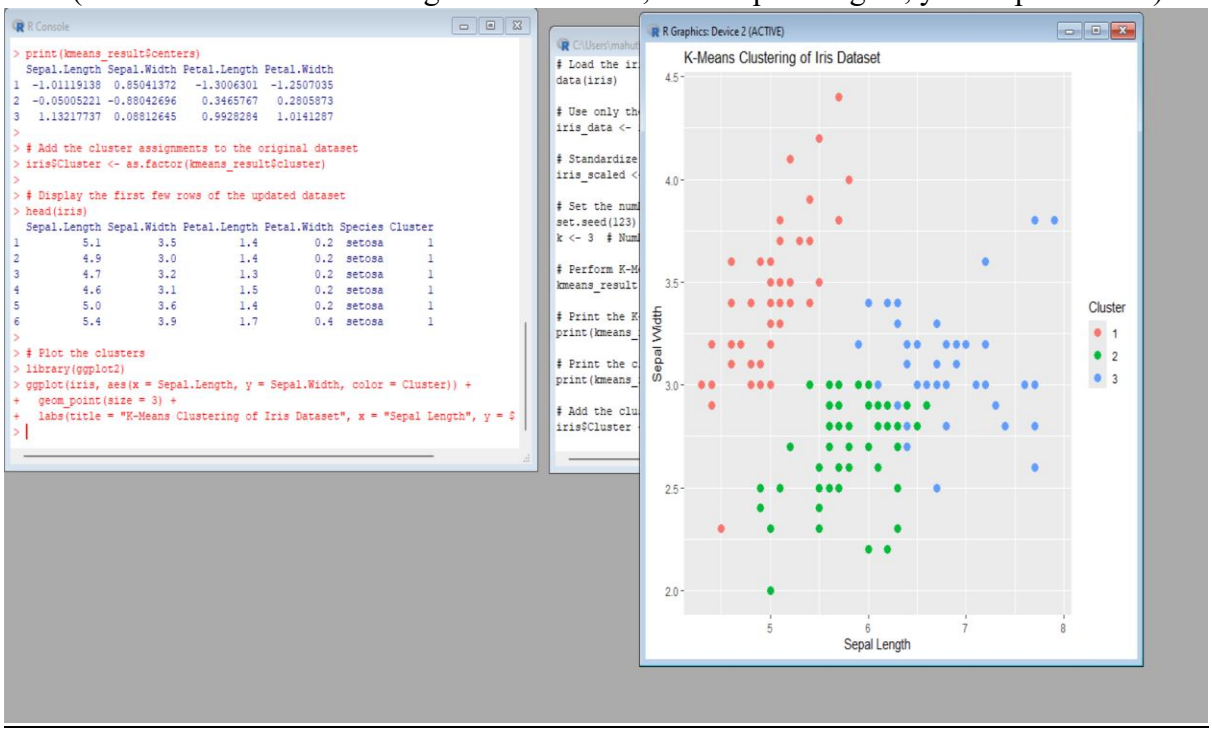
```
R Console

>
> # Cut the tree to form 3 clusters
> clusters <- cutree(hc_complete, k = 3)
>
> # Print the cluster memberships
> print(clusters)
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [38] 1 1 1 1 2 1 1 1 1 1 1 1 1 3 3 3 2 3 2 3 2 3 2 2 3 2 3 3 3 3 2 2 2 3 3 3 3
 [75] 3 3 3 3 3 2 2 2 2 3 3 3 3 2 3 2 2 3 2 2 2 3 3 3 2 2 3 3 3 3 3 3 2 3 3 3 3
[112] 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[149] 3 3
>
> # Add the clusters to the original dataset
> iris$Cluster <- as.factor(clusters)
>
> # Display the first few rows of the updated dataset
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species Cluster
1          5.1         3.5          1.4         0.2  setosa       1
2          4.9         3.0          1.4         0.2  setosa       1
3          4.7         3.2          1.3         0.2  setosa       1
4          4.6         3.1          1.5         0.2  setosa       1
5          5.0         3.6          1.4         0.2  setosa       1
6          5.4         3.9          1.7         0.4  setosa       1
>
```




## b) K-MEANS CLUSTERING

```
# Load the iris dataset
data(iris)

# Use only the numeric columns for clustering (exclude the Species column)
iris_data <- iris[, -5]
```

```
# Standardize the data
iris_scaled <- scale(iris_data)

# Set the number of clusters
set.seed(123)  # For reproducibility
k <- 3  # Number of clusters

# Perform K-Means clustering
kmeans_result <- kmeans(iris_scaled, centers = k, nstart = 25)

# Print the K-Means result
print(kmeans_result)

# Print the cluster centers
print(kmeans_result$centers)

# Add the cluster assignments to the original dataset
iris$Cluster <- as.factor(kmeans_result$cluster)

# Display the first few rows of the updated dataset
head(iris)

# Plot the clusters
library(ggplot2)
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Cluster)) +
  geom_point(size = 3) +
  labs(title = "K-Means Clustering of Iris Dataset", x = "Sepal Length", y = "Sepal Width")
```

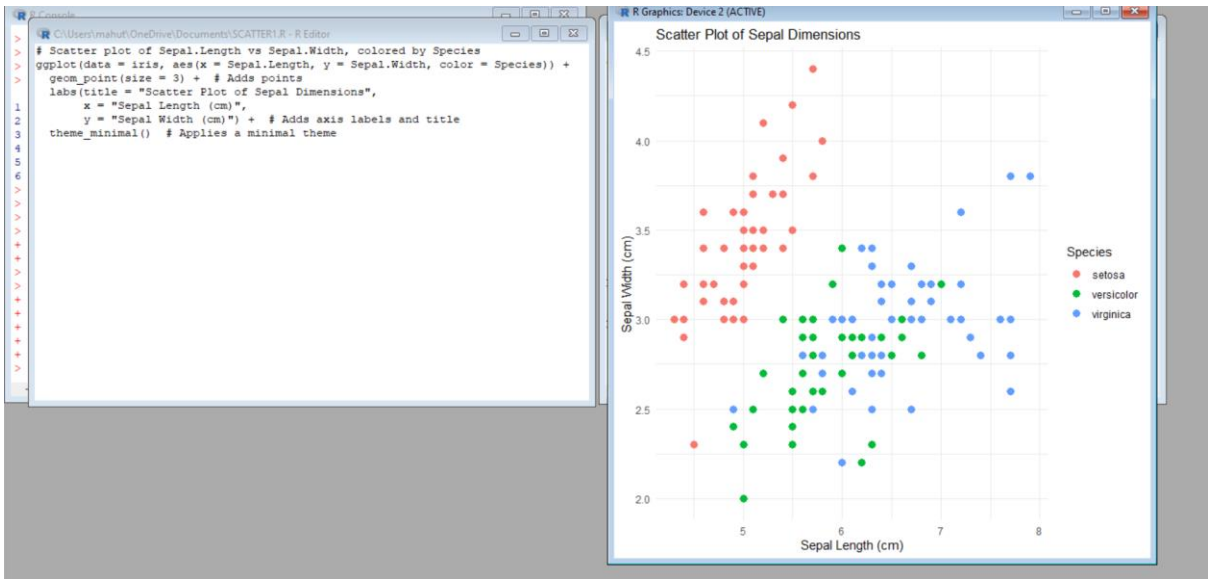


**Exp:10**

## VISUALIZE DATA USING ANY PLOTTING FRAMEWORK

### 1) SCATTER PLOT

```
# Install ggplot2 (if not already installed)
install.packages("ggplot2")

# Load the ggplot2 package
library(ggplot2)

# Scatter plot of Sepal.Length vs Sepal.Width, colored by Species
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point(size = 3) +  # Adds points
  labs(title = "Scatter Plot of Sepal Dimensions",
       x = "Sepal Length (cm)",
       y = "Sepal Width (cm)") +  # Adds axis labels and title
  theme_minimal()  # Applies a minimal theme
```

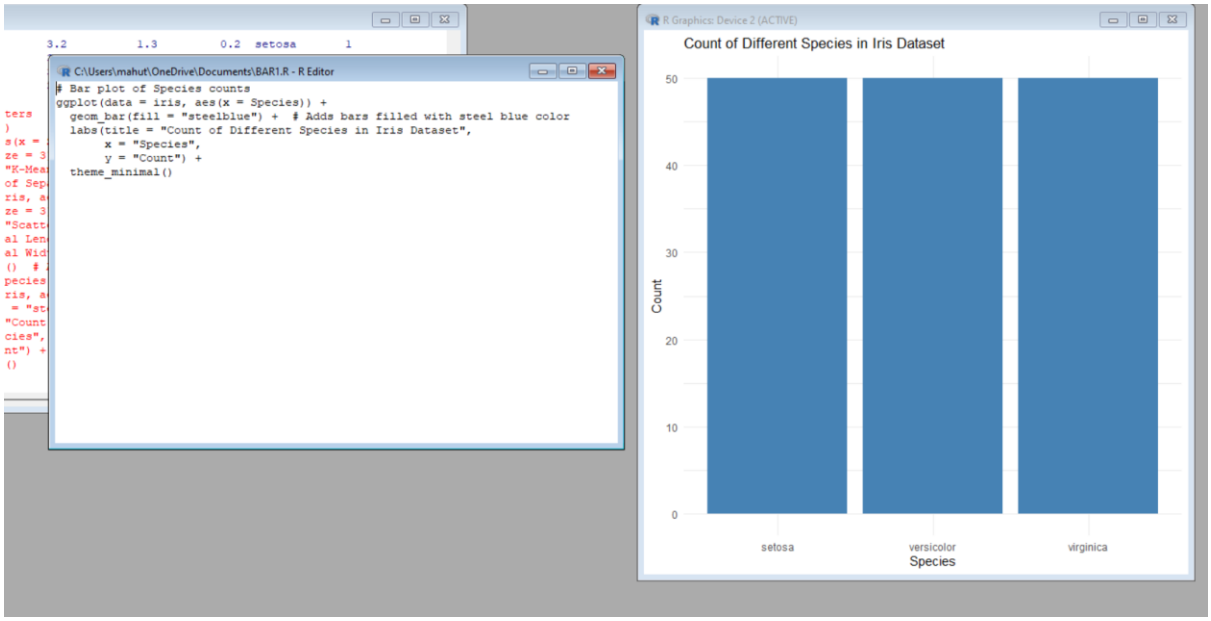



---

### 2) BAR CHART

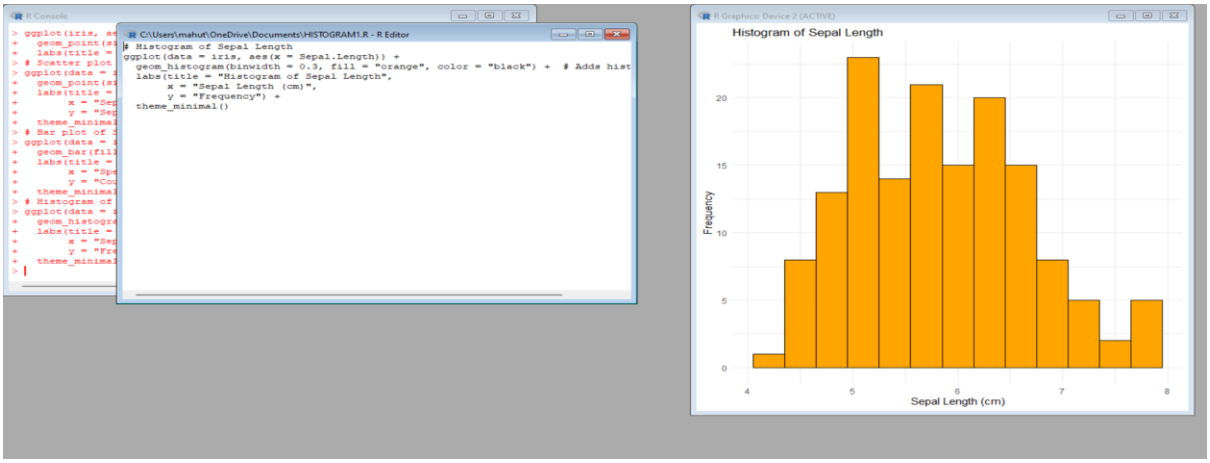
```
# Install ggplot2 (if not already installed)
install.packages("ggplot2")

# Load the ggplot2 package
library(ggplot2)

# Bar plot of Species counts
ggplot(data = iris, aes(x = Species)) +
  geom_bar(fill = "steelblue") +  # Adds bars filled with steel blue color
```

```
labs(title = "Count of Different Species in Iris Dataset",
     x = "Species",
     y = "Count") +
theme_minimal()
```




## 3) HISTOGRAM

```
# Install ggplot2 (if not already installed)
install.packages("ggplot2")

# Load the ggplot2 package
library(ggplot2)

# Histogram of Sepal Length
ggplot(data = iris, aes(x = Sepal.Length)) +
  geom_histogram(binwidth = 0.3, fill = "orange", color = "black") +  # Adds
histogram bars
  labs(title = "Histogram of Sepal Length",
       x = "Sepal Length (cm)",
       y = "Frequency") +
  theme_minimal()
```




**4)BOX PLOT**

```
# Install ggplot2 (if not already installed)
install.packages("ggplot2")

# Load the ggplot2 package
library(ggplot2)

# Box plot of Sepal Length for each Species
ggplot(data = iris, aes(x = Species, y = Sepal.Length, fill = Species)) +
  geom_boxplot() +  # Adds box plot
  labs(title = "Box Plot of Sepal Length by Species",
       x = "Species",
       y = "Sepal Length (cm)") +
  theme_minimal()
```