Done by : Anu Aniyammal S

Mentor Name : Arvind

Abstract: Data Manipulation and exploration in R

ACD\_ANR\_Project1.1

Problem Statement: **(Part I )**

There are 20 files with .dat extention. You have to read all the files in to single dataframe

Dataset :



Solution : **s**et the directory and read files using read.delim

Source code with **comments**  :

library(dplyr) **### For manipulation in data**

library(plyr)

setwd("C:/Users/eanuani/Documents/DAT File") **### Set the directory**

**### list the files**

filesnames = list.files(path="C:/Users/eanuani/Documents/DAT File",pattern="\*.dat") length(filesnames)

**### To append row by row by skipping the attributes column in 20 dat files**

for (file in filesnames)

{

**# if the merged dataset doesn't exist, create it**

if (!exists("dataset")){

dataset <- read.delim(file, header=TRUE, skip=9 ,sep="\t")

}

**# if the merged dataset does exist, append to it**

if (exists("dataset")){

temp\_dataset <-read.delim(file, header=TRUE, skip=9 ,sep="\t")

dataset<-rbind(dataset, temp\_dataset)

rm(temp\_dataset)

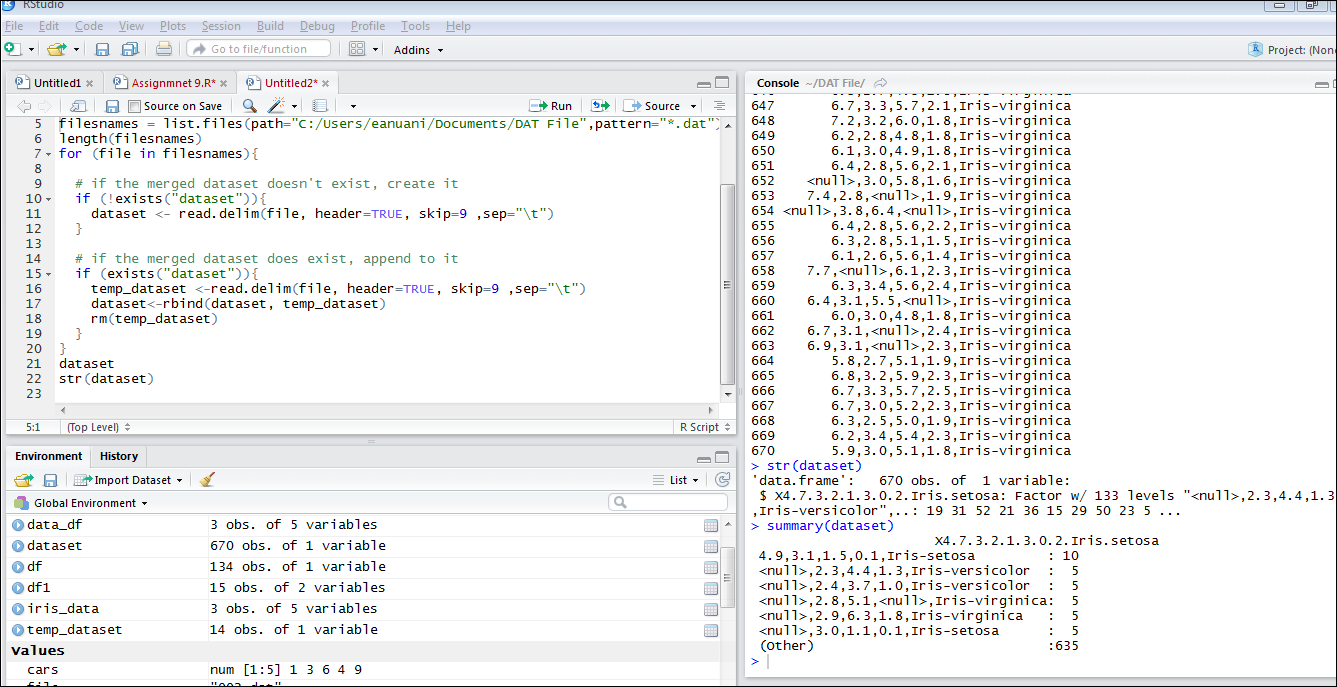
}

}

Dataset **### Final output**

str(dataset)

Screenshot for the solution after running code :



Problem Statement: **(Part II )**

The data is present in xml format, with file name, iris.xml. Your task is to read the XML data and store it in the data frame df.

Dataset :



Solution : **s**et the directory and read files using Xml

Package and Xmltodataframe command

Source code with **comments**  :

Install.packages(XML) **### Install XML package**

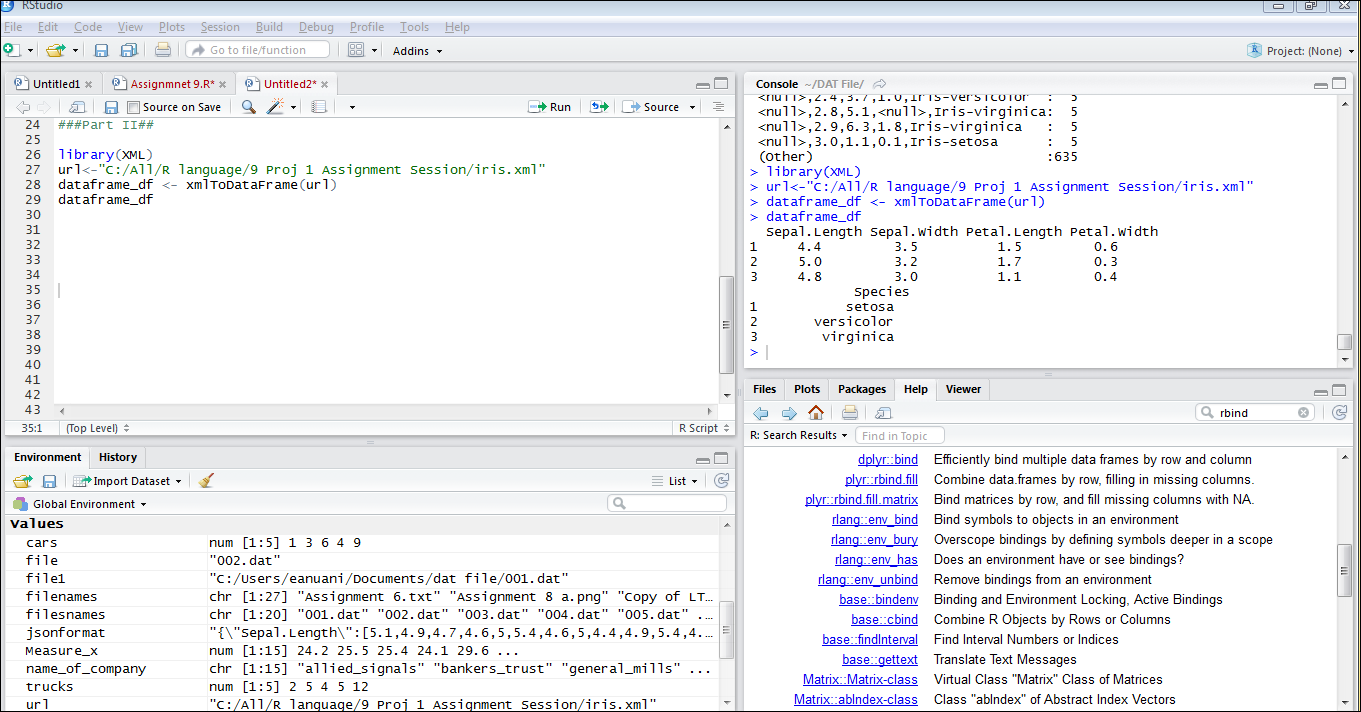
library(XML) **### Call the package**

url<-"C:/All/R language/9 Proj 1 Assignment Session/iris.xml" **### Set the path to file**

dataframe\_df <- xmlToDataFrame(url) **### importing data into a data frame**

dataframe\_df **### Final Output**

Screenshot for the solution after running code :



Problem Statement: **(Part III )**

Convert the iris data into the JSON format and read the data in JSON format and convert it into dataframe “iris\_data”.

Dataset : datframe\_df ouput from Part II

Solution : JSON installation and conversion using

commands

Source code with **comments**  :

Install.packages(rjson) ) **### Install XML package**

library(rjson) **### Call the package**

jsonformat<- toJSON(dataframe\_df) **### Conversion to json format**

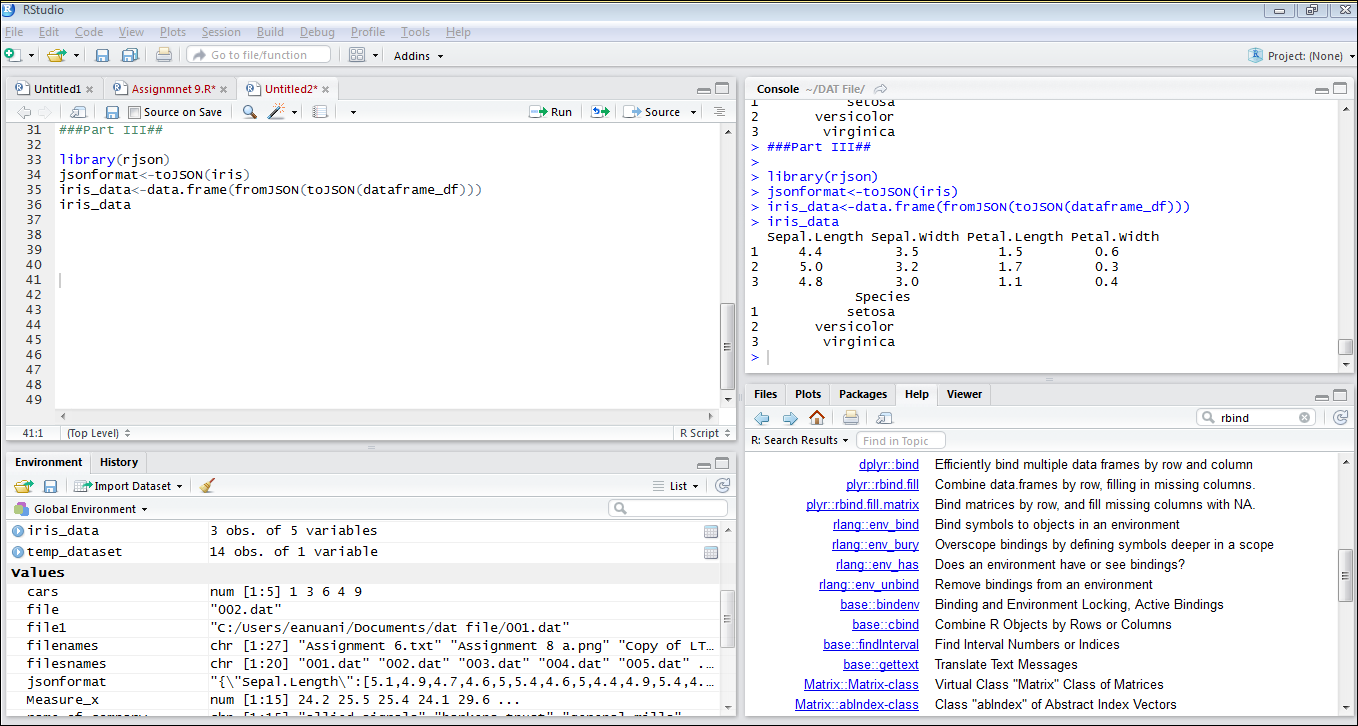
iris\_data1<-fromJSON( jsonformat)

iris\_data<-data.frame(iris\_data1,stringsAsFactors = F) **### Conversion to dataframe**

iris\_data **### Final Output**

str(iris\_data)

Screenshot for the solution after running code :



Problem Statement: **(Part IV )**

Use dplyr function on the data iris\_data. Implement select, match, filter, arrange, rename, and mutate function on the iris\_data.

* **Implement select**

Dataset : iris\_data ouput from Part III

Solution : using dplyr library

Source code with **comments**  :

str(iris\_data **) ###it projects your sorted data**

glimpse(iris\_data ) **###it projects same as your dataframe without any order**

iris\_data

select(iris\_data,3) **###select 3rd variable in dataframe**

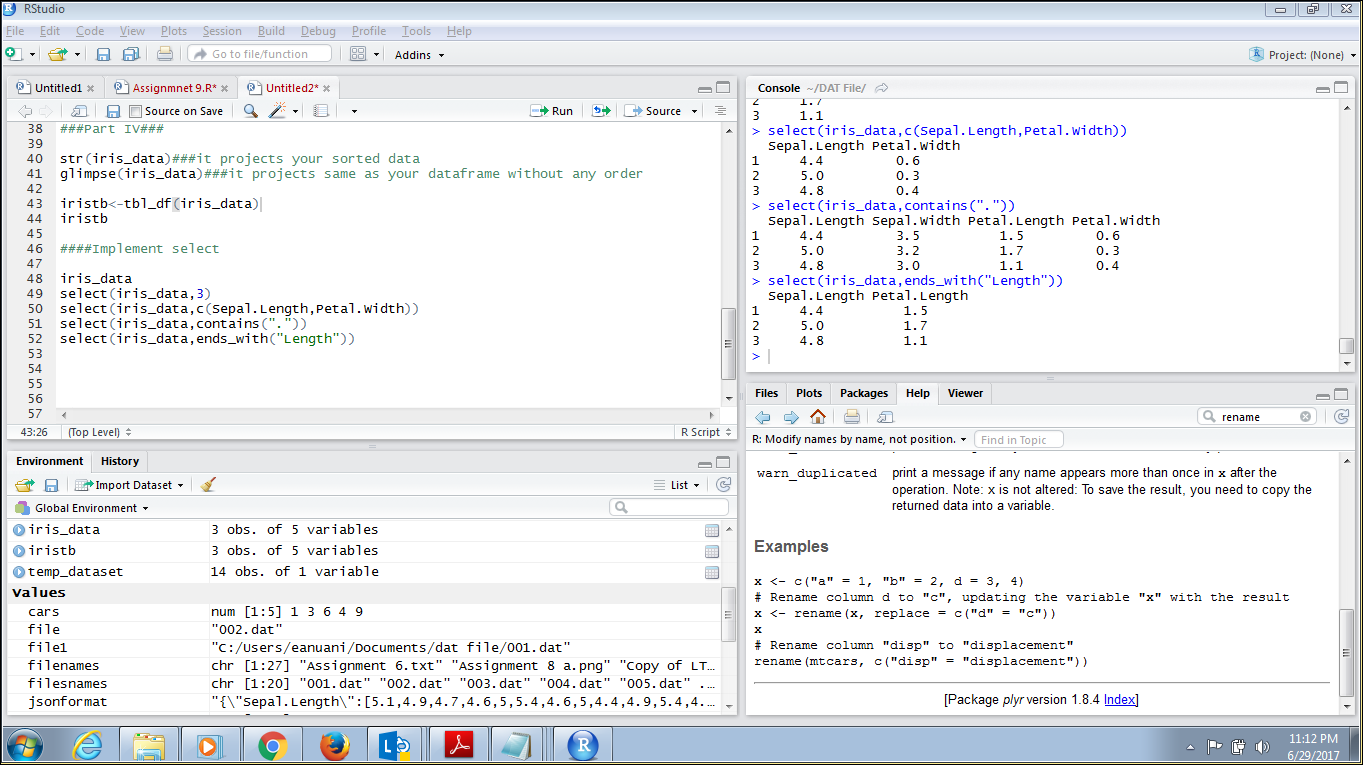
select(iris\_data,c(Sepal.Length,Petal.Width)) **###select from selective variables**

select(iris\_data,contains(".")) **###select from selective variables**  **,variables containing (.)**

select(iris\_data,ends\_with("Length")) **###select from selective variables**  **,variables containing Length**

select(iris\_data,-Petal.Length) **###Drop petal.length**

Screenshot for the solution after running code :



* **Implement filter**

Dataset : iris\_data ouput from Part III

Solution : using dplyr library

Source code with **comments**  :

**###Conversion of factors into numeric**

iris\_data$Petal.Width<-as.numeric(iris\_data$Petal.Width)

iris\_data$Petal.Width<-c(0.6,0.3,0.4)

iris\_data$Sepal.Width<-as.numeric(iris\_data$Sepal.Width)

iris\_data$Sepal.Width<-c(3.5,3.2,3.0)

iris\_data$Petal.Length<-as.numeric(iris\_data$Petal.Length)

iris\_data$Petal.Length<-c(1.5,1.1,1.7)

iris\_data$Sepal.Length<-as.numeric(iris\_data$Sepal.Length)

iris\_data$Sepal.Length<-c(4.4,5.0,4.8)

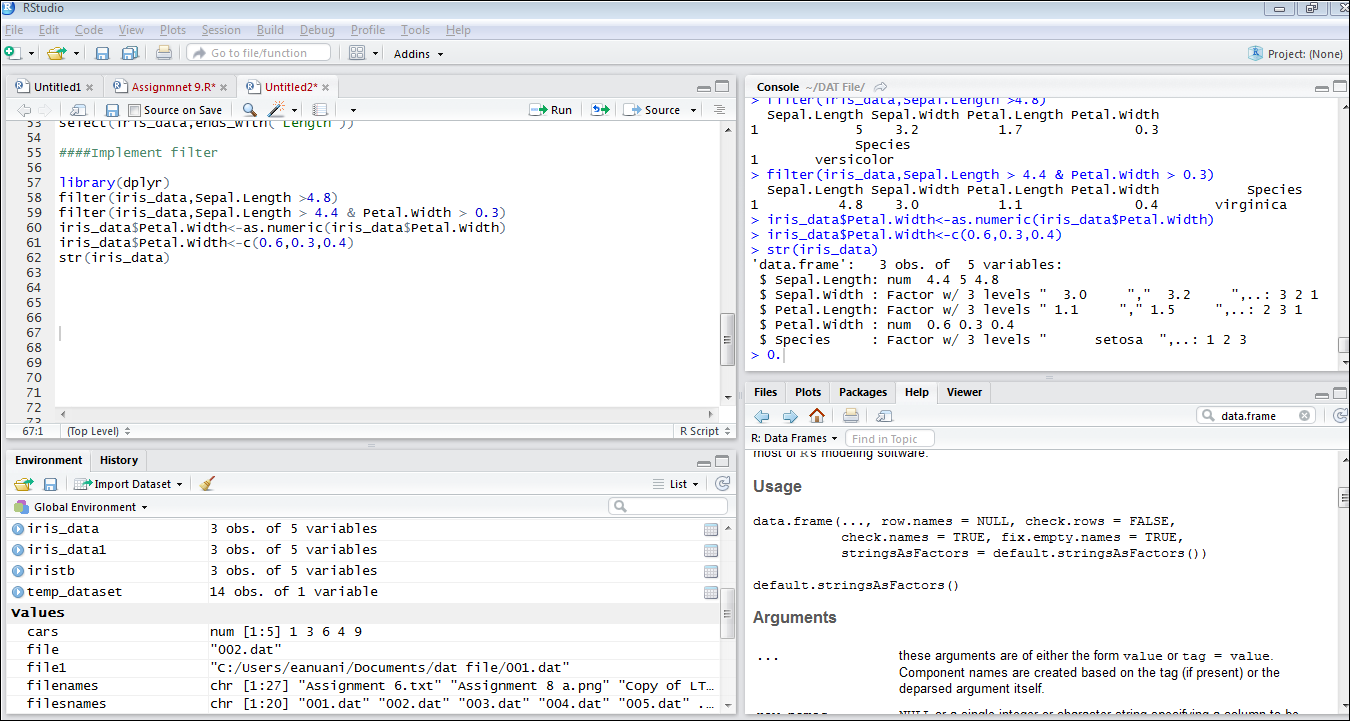
**####Implement filter**

library(dplyr) **### Call the package**

filter(iris\_data,Sepal.Length >4.8)  **### Filtering Sepal.length greater than 4.8**

filter(iris\_data,Sepal.Length > 4.4 & Petal.Width > 0.3) **### using logical operators**

Screenshot for the solution after running code :



* **Implement mutate**

Dataset : iris\_data ouput from Part III

Solution : using dplyr library

Source code with **comments**  :

**###Conversion of factors into numeric**

iris\_data$Petal.Width<-as.numeric(iris\_data$Petal.Width)

iris\_data$Petal.Width<-c(0.6,0.3,0.4)

iris\_data$Sepal.Width<-as.numeric(iris\_data$Sepal.Width)

iris\_data$Sepal.Width<-c(3.5,3.2,3.0)

iris\_data$Petal.Length<-as.numeric(iris\_data$Petal.Length)

iris\_data$Petal.Length<-c(1.5,1.1,1.7)

iris\_data$Sepal.Length<-as.numeric(iris\_data$Sepal.Length)

iris\_data$Sepal.Length<-c(4.4,5.0,4.8)

**####Implement mutate**

iris\_data %>%

mutate(iris\_data,

Sepal.Area = Sepal.Width \* Sepal.Length,

Petal.Area = Petal.Width \* Petal.Length,

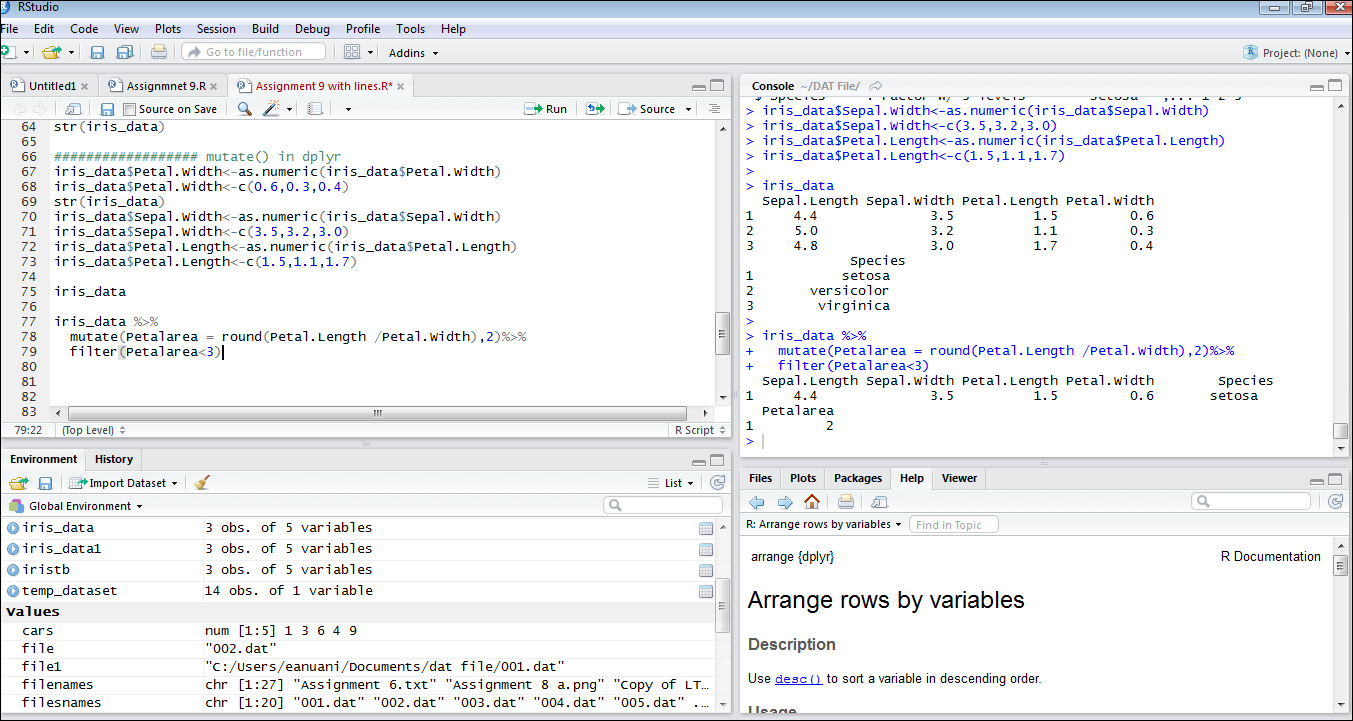
Area.Ratio = Petal.Area / Petal.Area) **####creating new variables using existing one)**

iris\_data %>%

mutate(Dimensions = round(Petal.Length /Petal.Width),2)%>%

filter(Dimensions<3)

Screenshot for the solution after running code :



* **Implement match**

Dataset : iris\_data ouput from Part III

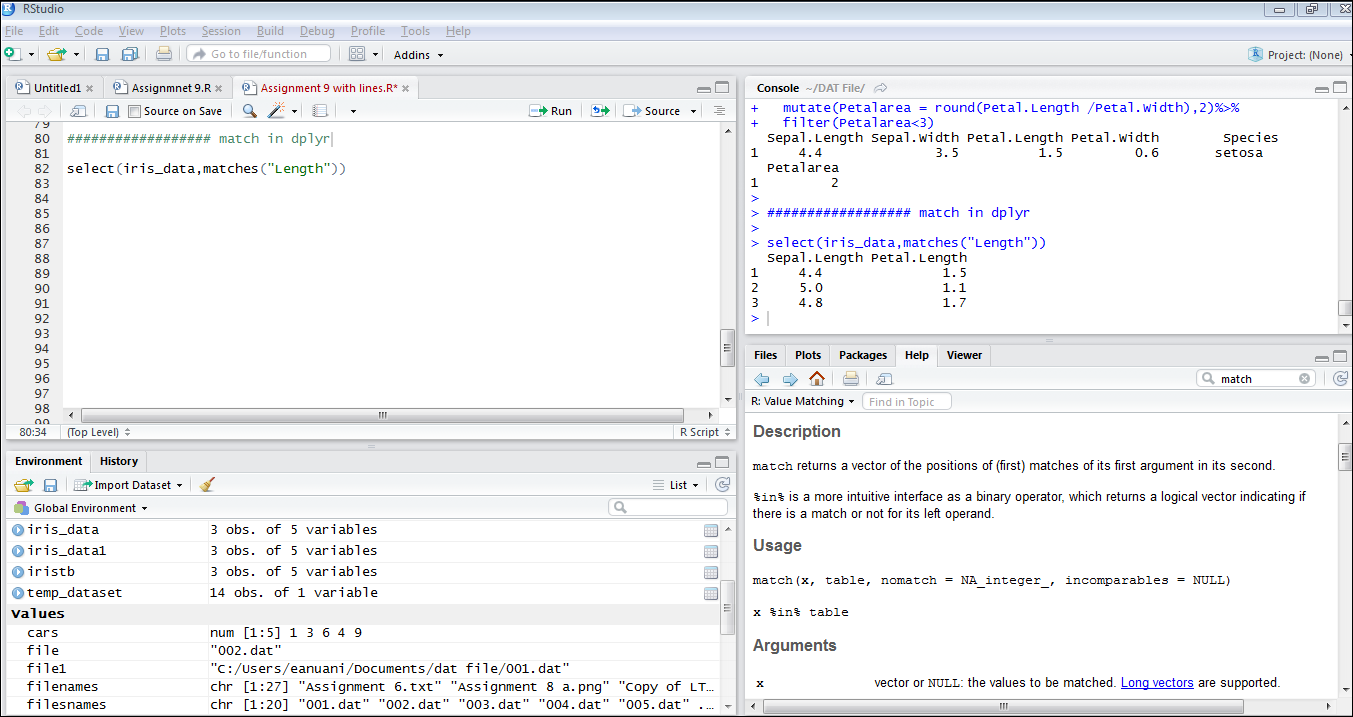
Solution : using dplyr library

Source code with **comments**  :

**################## match in dplyr**

select(iris\_data,matches("Length"))

Screenshot for the solution after running code :



* **Implement rename**

Dataset : iris\_data ouput from Part III

Solution : using dplyr library

Source code with **comments**  :

**################## rename in dplyr**

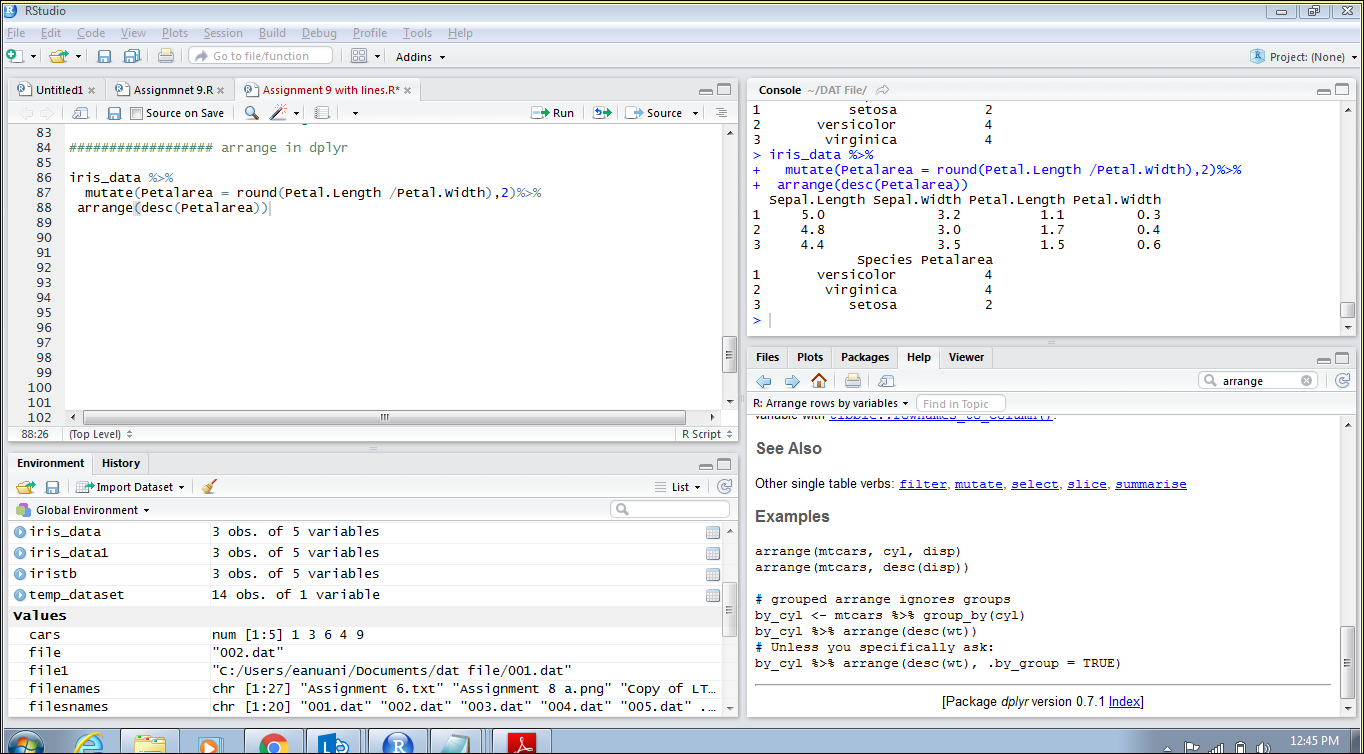
**### using select**

select(iris\_data, Species, PetalLength = Petal.Length, PetalWidth = Petal.Width)

**### using rename**

rename(iris\_data,c("Species"= "Speciesnew"))

Screenshot for the solution after running code :



* **Implement arrange**

Dataset : iris\_data ouput from Part III

Solution : using dplyr library

Source code with **comments**  :

**################## arrange in dplyr**

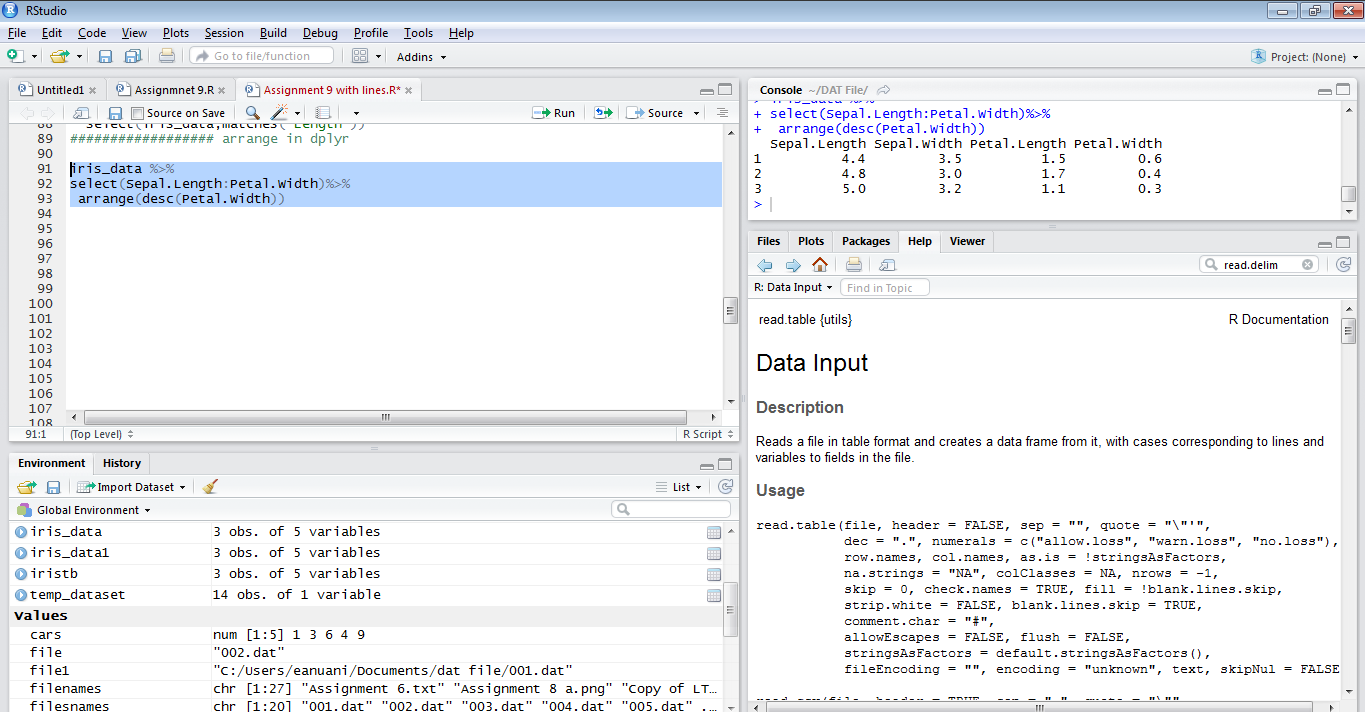
**### sort in descending order**

iris\_data %>%

select(Sepal.Length:Petal.Width)%>%

arrange(desc(Petal.Width))

Screenshot for the solution after running code :



Problem Statement: **(Part V )**

Print the summary of iris\_data

Dataset : iris\_data ouput from Part III

Solution : using dplyr library

Source code with **comments**  :

**################## summary of iris\_data**

iris\_data%>%

mutate(iris\_data,

Sepal.Area = Sepal.Width \* Sepal.Length, **### Creating new variable using mutate**

Petal.Area = Petal.Width \* Petal.Length,

Area.Ratio = Petal.Area / Petal.Area)%>%

summarise(mean = mean(Sepal.Length), **### Using pipe operator and using all arguments**

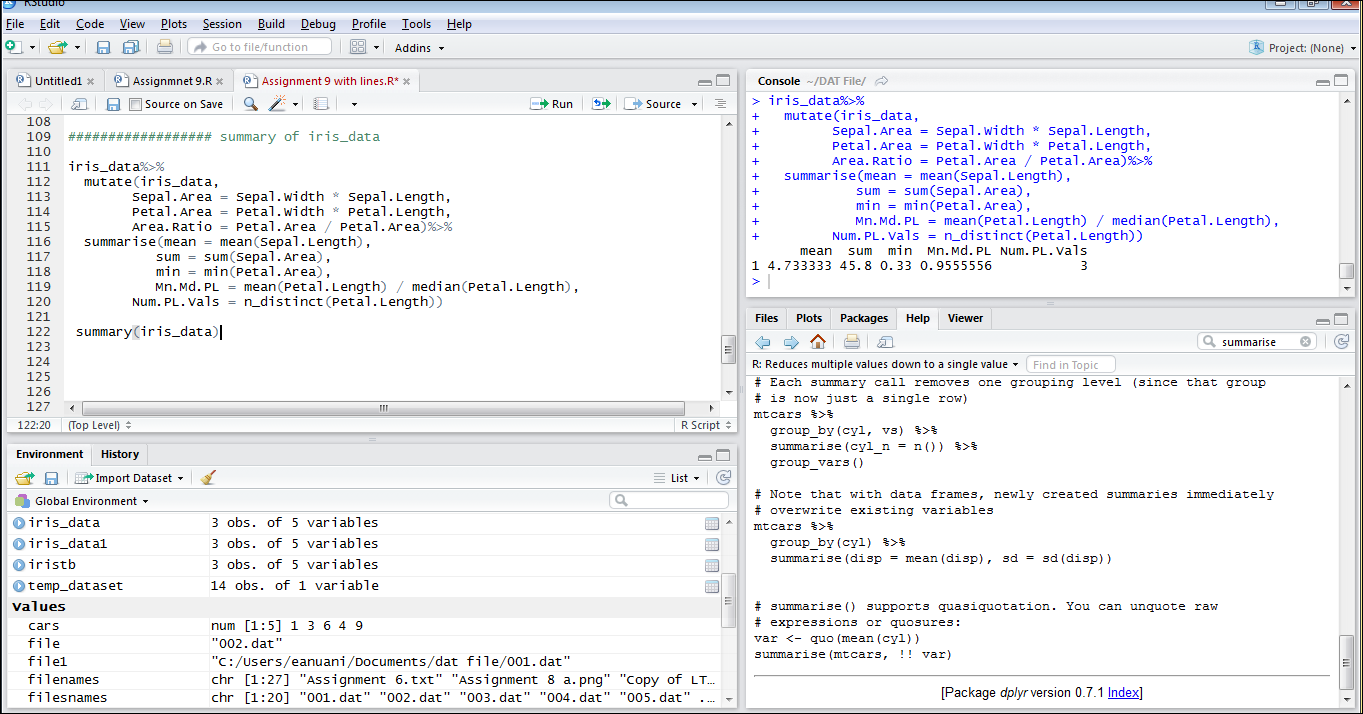
sum = sum(Sepal.Area),

min = min(Petal.Area),

Mn.Md.PL = mean(Petal.Length) / median(Petal.Length),

Num.PL.Vals = n\_distinct(Petal.Length))

Screenshot for the solution after running code :



THANK YOU