Project Title: Diabetes prediction using KNN algorithm

About Dataset:

Introduction

The Diabetes prediction dataset is a collection of medical and demographic data from patients, along with their diabetes status (positive or negative). The data includes features such as age, gender, body mass index (BMI), hypertension, heart disease, smoking history, HbA1c level, and blood glucose level. This dataset can be used to build machine learning models to predict diabetes in patients based on their medical history and demographic information. This can be useful for healthcare professionals in identifying patients who may be at risk of developing diabetes and in developing personalized treatment plans. Additionally, the dataset can be used by researchers to explore the relationships between various medical and demographic factors and the likelihood of developing diabetes.

Key Features:

Number of records: 100000

Number of columns: 9

Target Variable : Diabetes

Prediction Accuracy : 97%

Code:

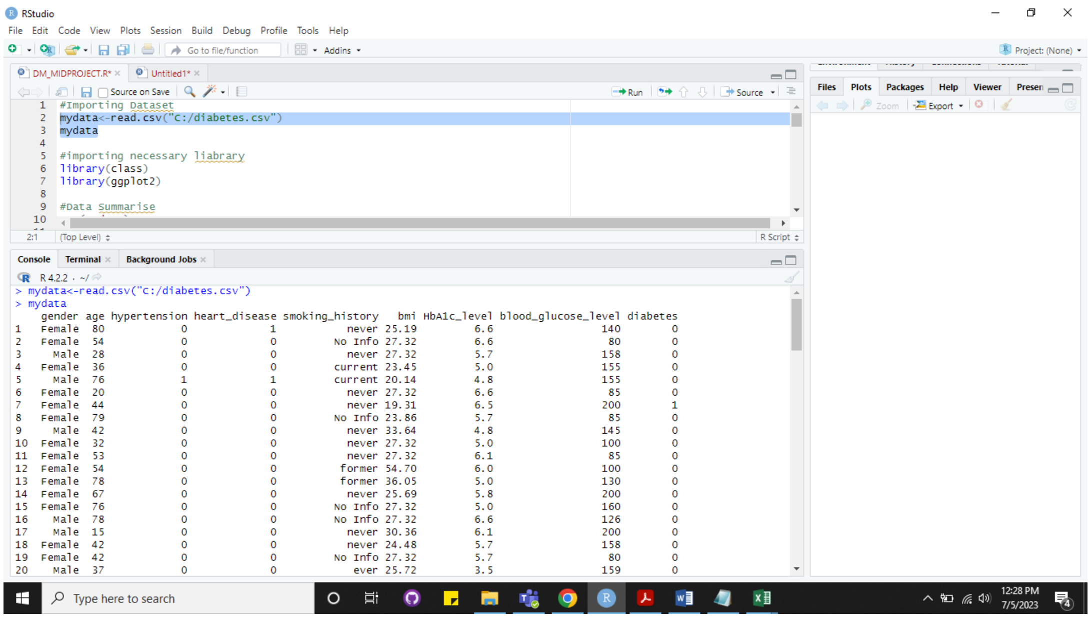
```
#Importing Dataset

mydata<-read.csv("C:/diabetes.csv")

mydata


#importing necessary liabrary

library(class)

library(ggplot2)


#Data Summarise

str(mydata)

summary(mydata)

attributes<- names(mydata)

attributes

dataType <- c(typeof(mydata$gender), typeof(mydata$age),
typeof(mydata$hypertension),typeof(mydata$heart_disease),

        typeof(mydata$bmi),  typeof(mydata$HbA1c_level), typeof(mydata$blood_glucose_level),

        typeof(mydata$diabetes))

dataType


head(mydata)


colSums(is.na(mydata)) #how many instances are missing



#Data Normalization/scaling

data_norm <- setdiff(names(mydata), c("gender", "smoking_history"))

mydata[data_norm] <- scale(mydata[data_norm])
```

```r
head(mydata)

head(data_norm)

data_norm

mydata[data_norm]

colSums(is.na(mydata[data_norm]))




# Setting predictor variables and the target variable

predictor_cols <- names(mydata[data_norm])[-ncol(mydata[data_norm])]

target_col <- names(mydata[data_norm])[ncol(mydata[data_norm])]


set.seed(123)

train_indices <- sample(1:nrow(mydata[data_norm]), round(0.7 * nrow(mydata[data_norm])))

train_data <- mydata[data_norm][train_indices, predictor_cols]

train_labels <- mydata[data_norm][train_indices, target_col]

test_data <- mydata[data_norm][-train_indices, predictor_cols]

test_labels <- mydata[data_norm][-train_indices, target_col]




# Set the value of k (number of neighbors) and distance measures


knn_with_distance_measure <- function(train_data, test_data, train_labels, k,

                                      distance_measure) {

  predicted_labels <- knn(train = train_data, test = test_data, cl = train_labels, k = k, prob =

                          TRUE, use.all = TRUE)
```

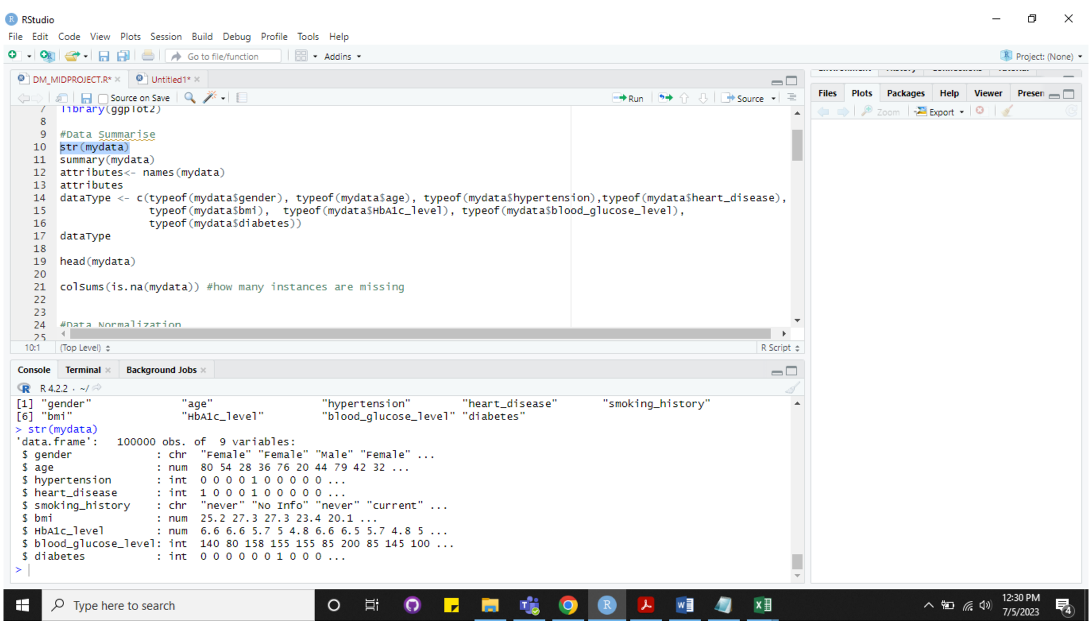
```
  return(predicted_labels)
}



# Set the values of k
k_values <- c(3, 5, 7)
# Initialize vectors to store accuracies
accuracies <- vector()
# Apply k-NN for each k value and distance measure
for (k in k_values) {
  # Apply k-NN with Euclidean distance
  euclidean_predictions <- knn_with_distance_measure(train_data, test_data, train_labels, k,
                                                     "euclidean")
  # Apply k-NN with Manhattan distance
  manhattan_predictions <- knn_with_distance_measure(train_data, test_data, train_labels, k,
                                                     "manhattan")
  # Apply k-NN with Maximum distance
  maximum_predictions <- knn_with_distance_measure(train_data, test_data, train_labels, k,
                                                   "maximum")
  # Evaluate the accuracy of the predictions
  accuracy_euclidean <- sum(euclidean_predictions == test_labels) / length(test_labels)
  accuracy_manhattan <- sum(manhattan_predictions == test_labels) / length(test_labels)
  accuracy_maximum <- sum(maximum_predictions == test_labels) / length(test_labels)
  # Store the accuracy
  accuracies <- c(accuracies, accuracy_euclidean, accuracy_manhattan, accuracy_maximum)
  # Print the accuracy for the current k value
  cat("Accuracy for k =", k, "\n")
  cat("Euclidean Distance:", accuracy_euclidean, "\n")
```

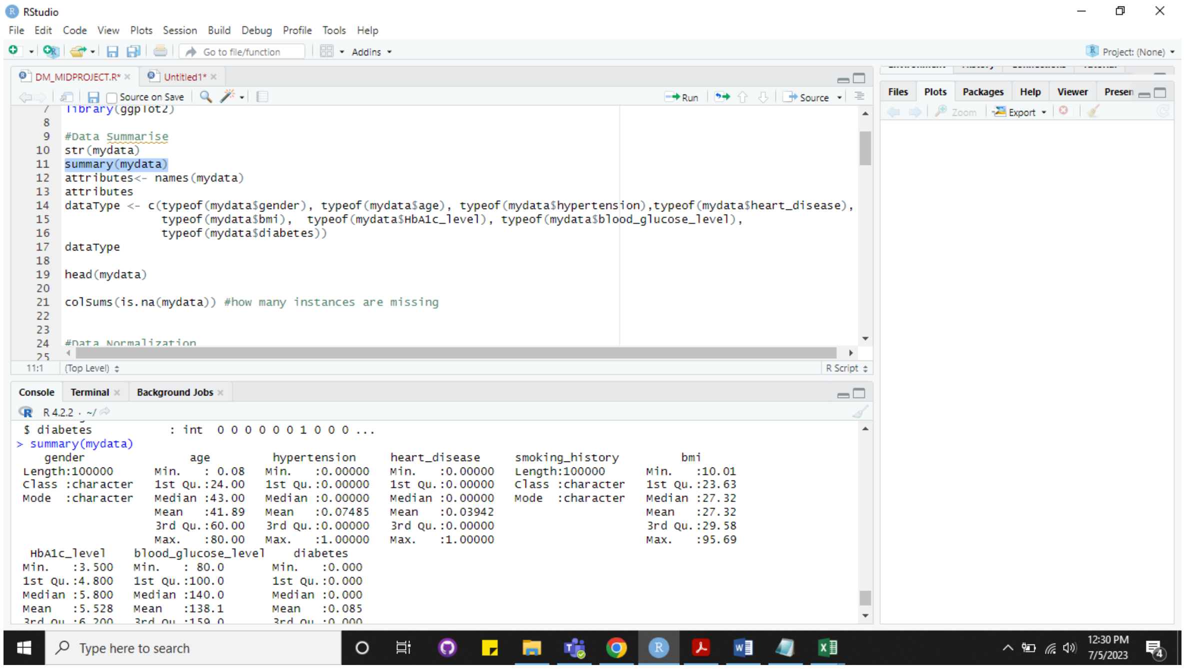
```r
  cat("Manhattan Distance:", accuracy_manhattan, "\n")
  cat("Maximum Distance:", accuracy_maximum, "\n")
  cat("\n")
}


# Create a data frame for accuracies
accuracy_df <- data.frame(Distance = rep(c("Euclidean", "Manhattan", "Maximum"),
                                  length(k_values)),
                          K = rep(k_values, each = 3),
                          Accuracy = accuracies)
accuracy_df

# Plot the accuracies
ggplot(accuracy_df, aes(x = K, y = Accuracy, color = Distance, group = Distance)) +
  geom_line() +
  geom_point() +
  labs(title = "Accuracy of k-NN with Different Distance Measures",
       x = "k",
       y = "Accuracy",
       color = "Distance Measure") +
  theme_minimal()
```

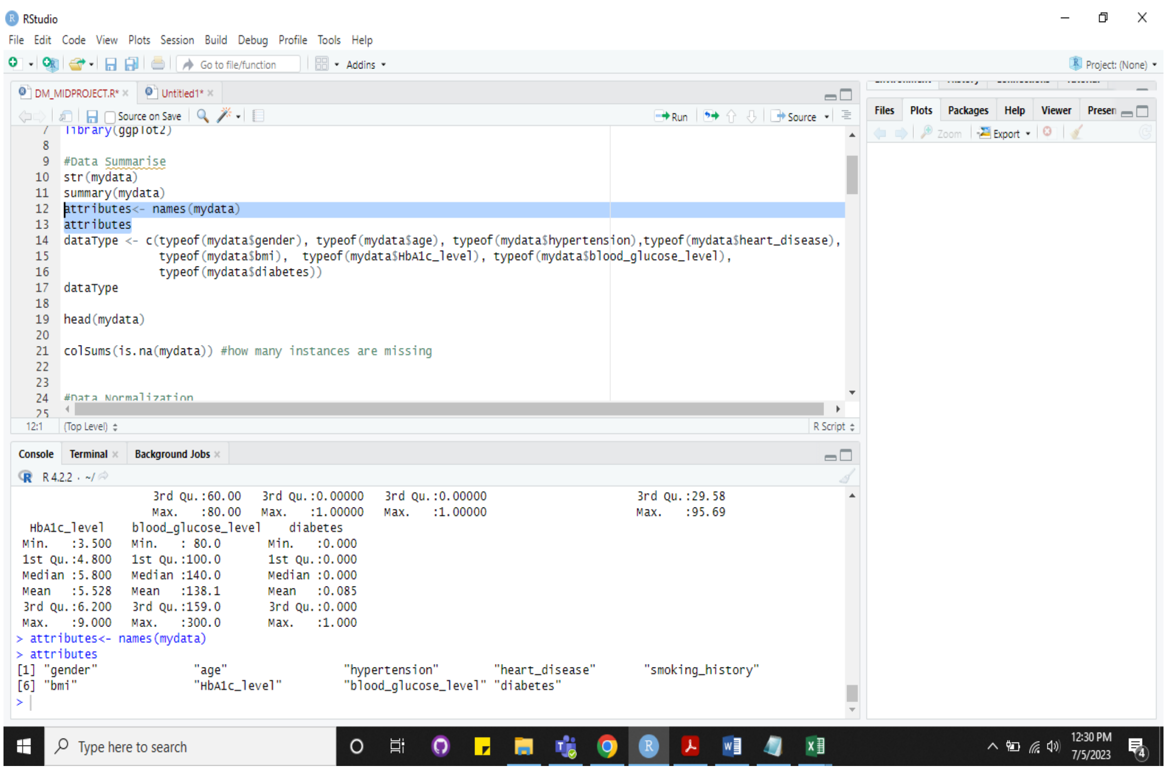
1. Importing Dataset

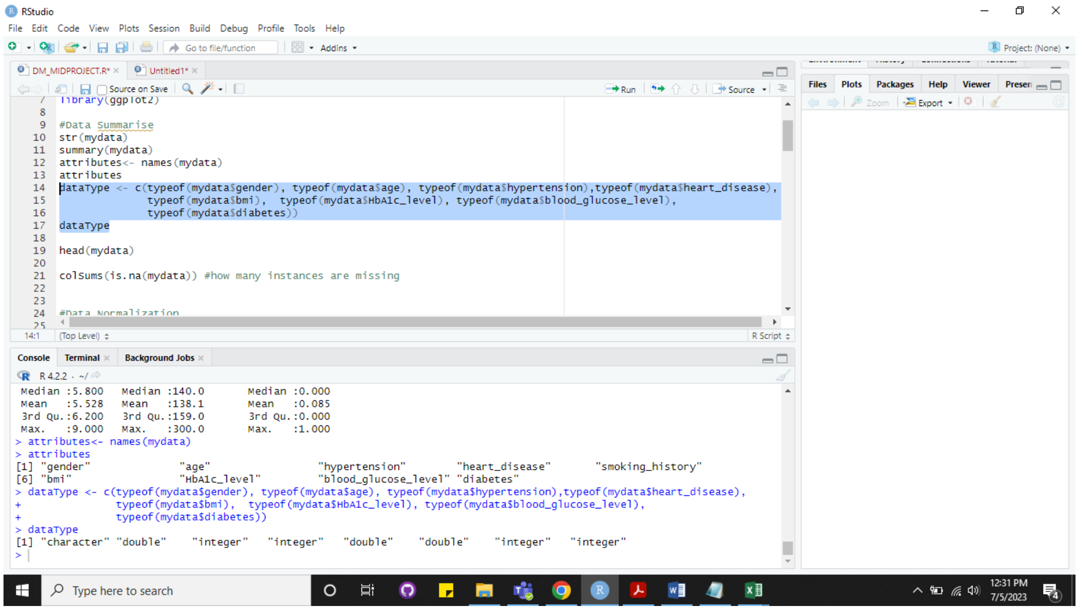


2. Summary

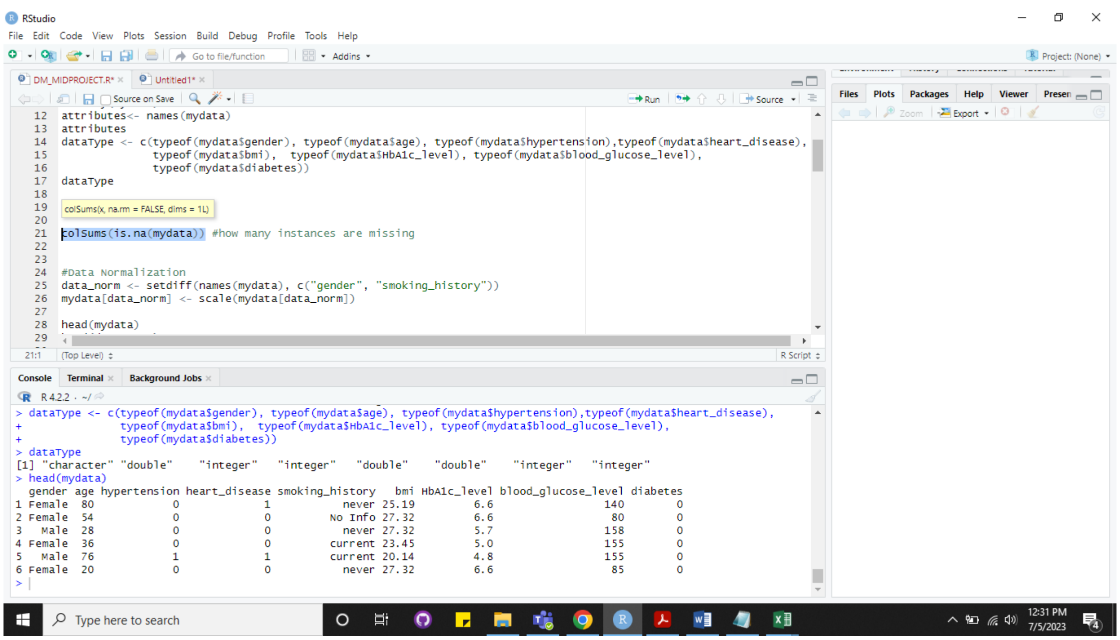



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
DM_MIDPROJECT.R*
Untitled1*
Source on Save
Run
Source
library(ggplot2)
#Data Summarise
str(mydata)
summary(mydata)
attributes<- names(mydata)
attributes
dataType <- c(typeof(mydata$gender), typeof(mydata$age), typeof(mydata$hypertension),typeof(mydata$heart_disease),
typeof(mydata$bmi), typeof(mydata$HbA1c_level), typeof(mydata$blood_glucose_level),
typeof(mydata$diabetes))
dataType
head(mydata)
colSums(is.na(mydata)) #how many instances are missing
#Data Normalization
(Top Level)
R Script
Console
Terminal
Background Jobs
R 4.2.2 · ~/
$ diabetes : int 0 0 0 0 0 0 1 0 0 0 ...
> summary(mydata)
gender age hypertension heart_disease smoking_history bmi
Length:100000 Min. : 0.08 Min. :0.00000 Min. :0.00000 Length:100000 Min. :10.01
Class :character 1st Qu.:24.00 1st Qu.:0.00000 1st Qu.:0.00000 Class :character 1st Qu.:23.63
Mode :character Median :43.00 Median :0.00000 Median :0.00000 Mode :character Median :27.32
Mean :41.89 Mean :0.07485 Mean :0.03942 Mean :27.32
3rd Qu.:60.00 3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:29.58
Max. :80.00 Max. :1.00000 Max. :1.00000 Max. :95.69
HbA1c_level blood_glucose_level diabetes
Min. :3.500 Min. : 80.0 Min. :0.000
1st Qu.:4.800 1st Qu.:100.0 1st Qu.:0.000
Median :5.800 Median :140.0 Median :0.000
Mean :5.528 Mean :138.1 Mean :0.085
Files Plots Packages Help Viewer
Project: (None)
Type here to search
12:30 PM
7/5/2023

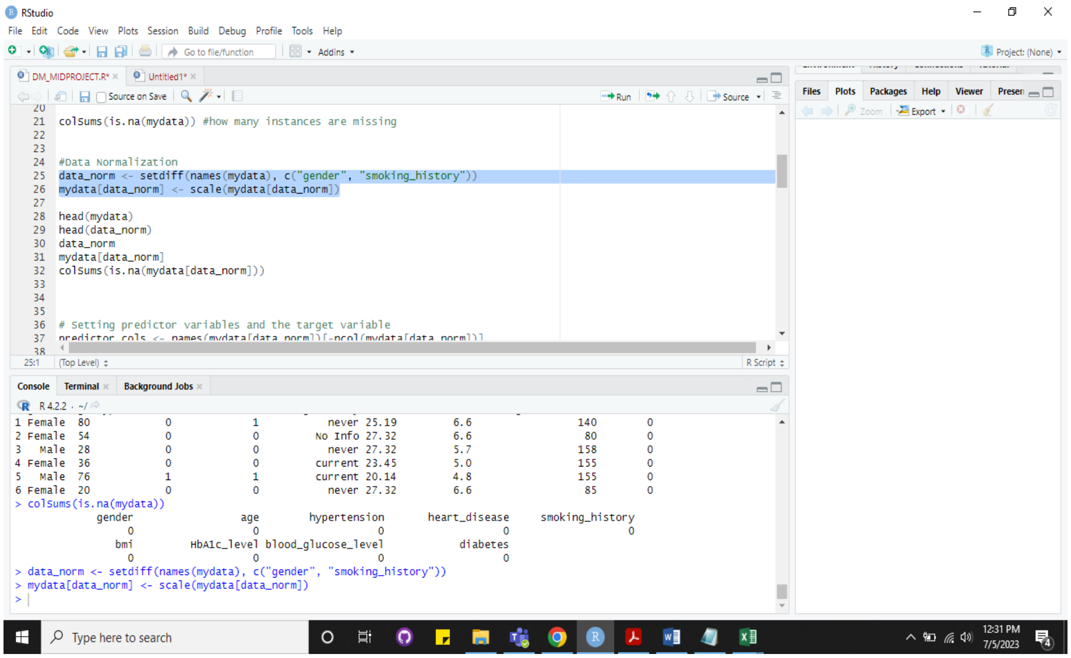

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
DM_MIDPROJECT.R*
Untitled1*
Source on Save
Run
Source
library(ggplot2)
#Data Summarise
str(mydata)
summary(mydata)
attributes<- names(mydata)
attributes
dataType <- c(typeof(mydata$gender), typeof(mydata$age), typeof(mydata$hypertension),typeof(mydata$heart_disease),
typeof(mydata$bmi), typeof(mydata$HbA1c_level), typeof(mydata$blood_glucose_level),
typeof(mydata$diabetes))
dataType
head(mydata)
colSums(is.na(mydata)) #how many instances are missing
#Data Normalization
(Top Level)
R Script
Console
Terminal
Background Jobs
R 4.2.2 · ~/
3rd Qu.:60.00 3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:29.58
Max. :80.00 Max. :1.00000 Max. :1.00000 Max. :95.69
HbA1c_level blood_glucose_level diabetes
Min. :3.500 Min. : 80.0 Min. :0.000
1st Qu.:4.800 1st Qu.:100.0 1st Qu.:0.000
Median :5.800 Median :140.0 Median :0.000
Mean :5.528 Mean :138.1 Mean :0.085
3rd Qu.:6.200 3rd Qu.:159.0 3rd Qu.:0.000
Max. :9.000 Max. :300.0 Max. :1.000
> attributes<- names(mydata)
> attributes
[1] "gender" "age" "hypertension" "heart_disease" "smoking_history"
[6] "bmi" "HbA1c_level" "blood_glucose_level" "diabetes"
Files Plots Packages Help Viewer
Project: (None)
Type here to search
12:30 PM
7/5/2023

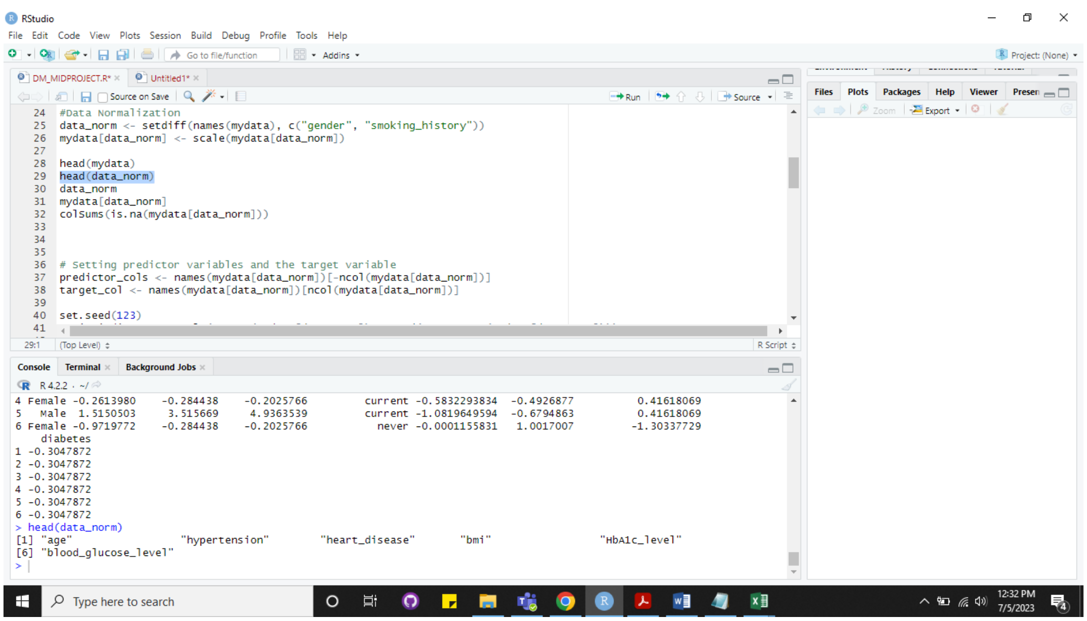


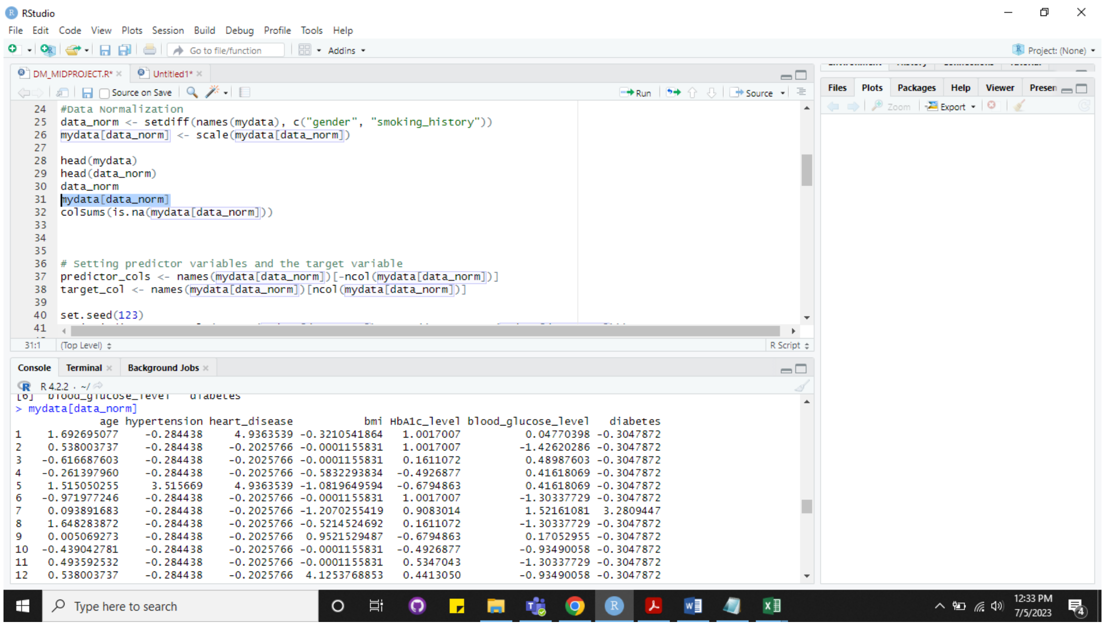
3. Looking for missing values

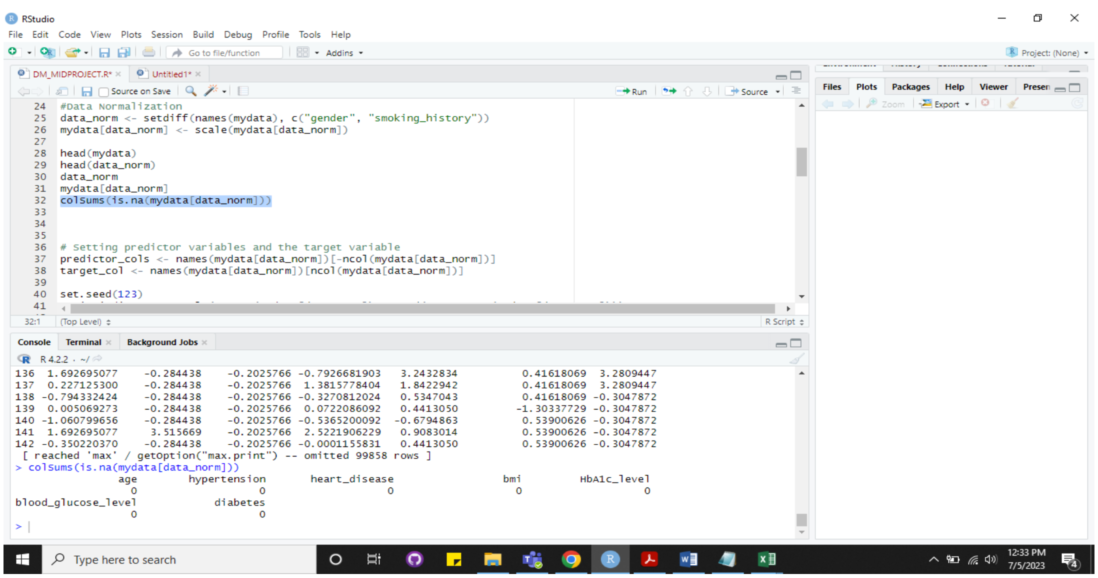


4. Data Normalization

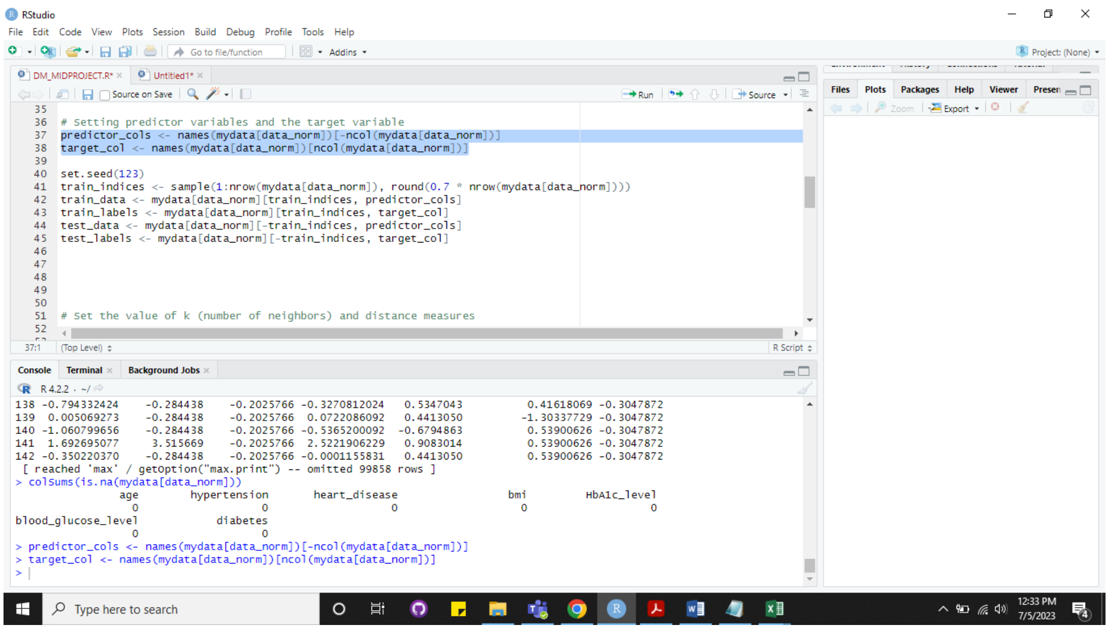




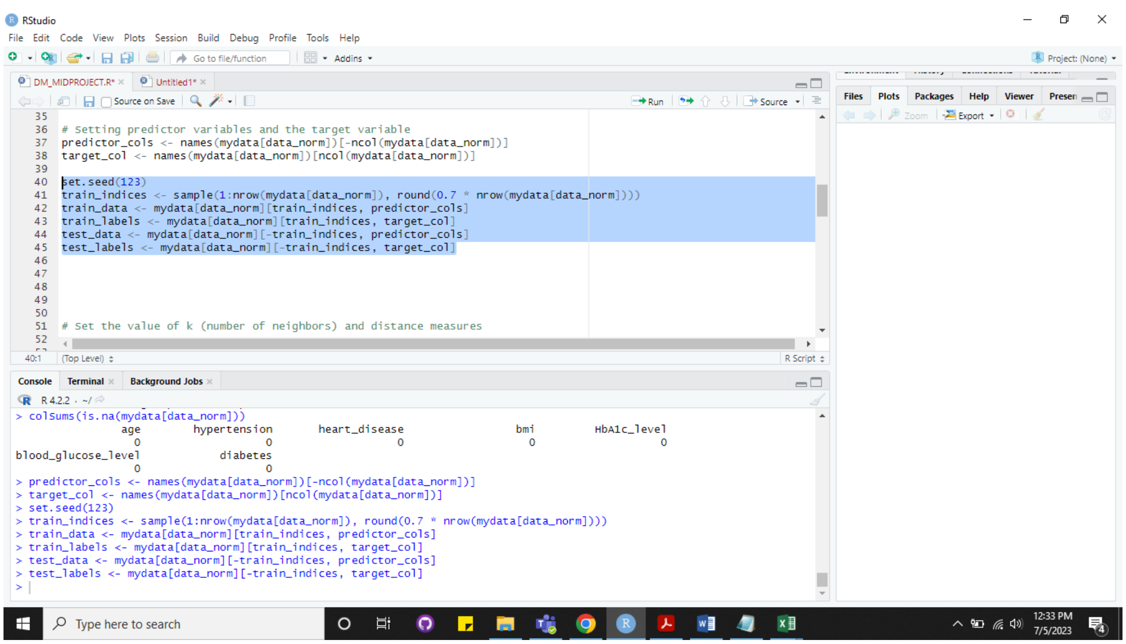

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
DM_MIDPROJECT.R*
Untitled1*
Source on Save
Run
Source
#Data Normalization
data_norm <- setdiff(names(mydata), c("gender", "smoking_history"))
mydata[data_norm] <- scale(mydata[data_norm])
head(mydata)
head(data_norm)
data_norm
mydata[data_norm]
colSums(is.na(mydata[data_norm]))
# Setting predictor variables and the target variable
predictor_cols <- names(mydata[data_norm])[-ncol(mydata[data_norm])]
target_col <- names(mydata[data_norm])[ncol(mydata[data_norm])]
set.seed(123)
31:1
(Top Level)
R Script
Console
Terminal
Background Jobs
R 4.2.2 · ~/
> mydata[data_norm]
age hypertension heart_disease bmi HbA1c_level blood_glucose_level diabetes
1 1.692695077 -0.284438 4.9363539 -0.3210541864 1.0017007 0.04770398 -0.3047872
2 0.538003737 -0.284438 -0.2025766 -0.0001155831 1.0017007 -1.42620286 -0.3047872
3 -0.616687603 -0.284438 -0.2025766 -0.0001155831 0.1611072 0.48987603 -0.3047872
4 -0.261397960 -0.284438 -0.2025766 -0.5832293834 -0.4926877 0.41618069 -0.3047872
5 1.515050255 3.515669 4.9363539 -1.0819649594 -0.6794863 0.41618069 -0.3047872
6 -0.971977246 -0.284438 -0.2025766 -0.0001155831 1.0017007 -1.30337729 -0.3047872
7 0.093891683 -0.284438 -0.2025766 -1.2070255419 0.9083014 1.52161081 3.2809447
8 1.648283872 -0.284438 -0.2025766 -0.5214524692 0.1611072 -1.30337729 -0.3047872
9 0.005069273 -0.284438 -0.2025766 0.9521529487 -0.6794863 0.17052955 -0.3047872
10 -0.439042781 -0.284438 -0.2025766 -0.0001155831 -0.4926877 -0.93490058 -0.3047872
11 0.493592532 -0.284438 -0.2025766 -0.0001155831 0.5347043 -1.30337729 -0.3047872
12 0.538003737 -0.284438 -0.2025766 4.1253768853 0.4413050 -0.93490058 -0.3047872
Files Plots Packages Help Viewer
Type here to search
12:33 PM
7/5/2023


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
DM_MIDPROJECT.R*
Untitled1*
#Data Normalization
data_norm <- setdiff(names(mydata), c("gender", "smoking_history"))
mydata[data_norm] <- scale(mydata[data_norm])
head(mydata)
head(data_norm)
data_norm
mydata[data_norm]
colSums(is.na(mydata[data_norm]))
# Setting predictor variables and the target variable
predictor_cols <- names(mydata[data_norm])[-ncol(mydata[data_norm])]
target_col <- names(mydata[data_norm])[ncol(mydata[data_norm])]
set.seed(123)
32:1
(Top Level)
R Script
Console
Terminal
Background Jobs
R 4.2.2 · ~/
136 1.692695077 -0.284438 -0.2025766 -0.7926681903 3.2432834 0.41618069 3.2809447
137 0.227125300 -0.284438 -0.2025766 1.3815778404 1.8422942 0.41618069 3.2809447
138 -0.794332424 -0.284438 -0.2025766 -0.3270812024 0.5347043 0.41618069 -0.3047872
139 0.005069273 -0.284438 -0.2025766 0.0722086092 0.4413050 -1.30337729 -0.3047872
140 -1.060799656 -0.284438 -0.2025766 -0.5365200092 -0.6794863 0.53900626 -0.3047872
141 1.692695077 3.515669 -0.2025766 2.5221906229 0.9083014 0.53900626 -0.3047872
142 -0.350220370 -0.284438 -0.2025766 -0.0001155831 0.4413050 0.53900626 -0.3047872
[reached 'max' / getOption("max.print") -- omitted 99858 rows]
> colSums(is.na(mydata[data_norm]))
age hypertension heart_disease bmi HbA1c_level
0 0 0 0 0
blood_glucose_level diabetes
0 0
Files Plots Packages Help Viewer
Type here to search
12:33 PM
7/5/2023

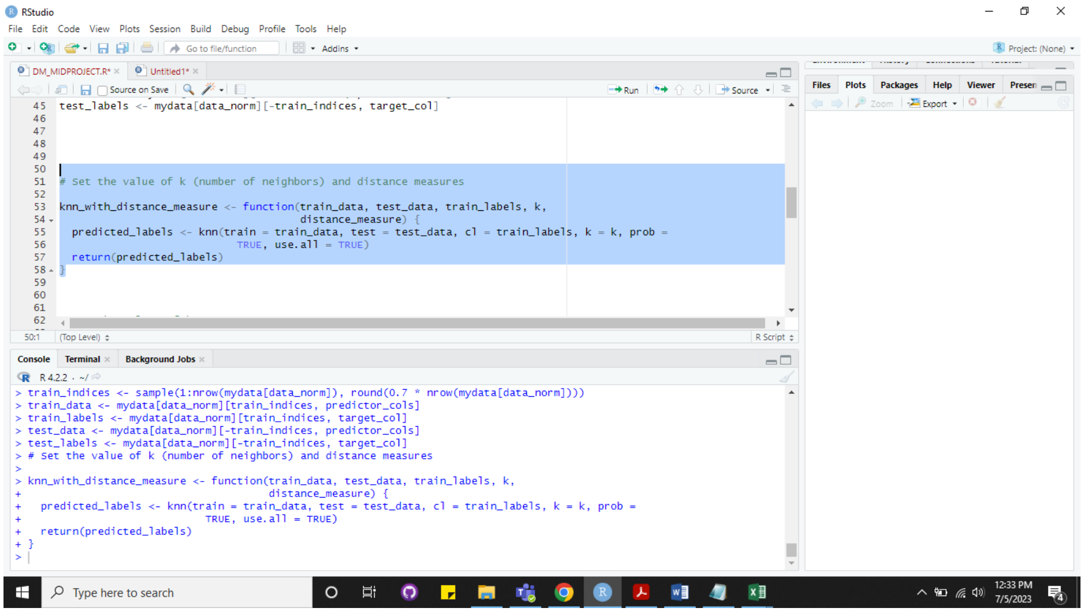
5. Setting predictor variables and target variable



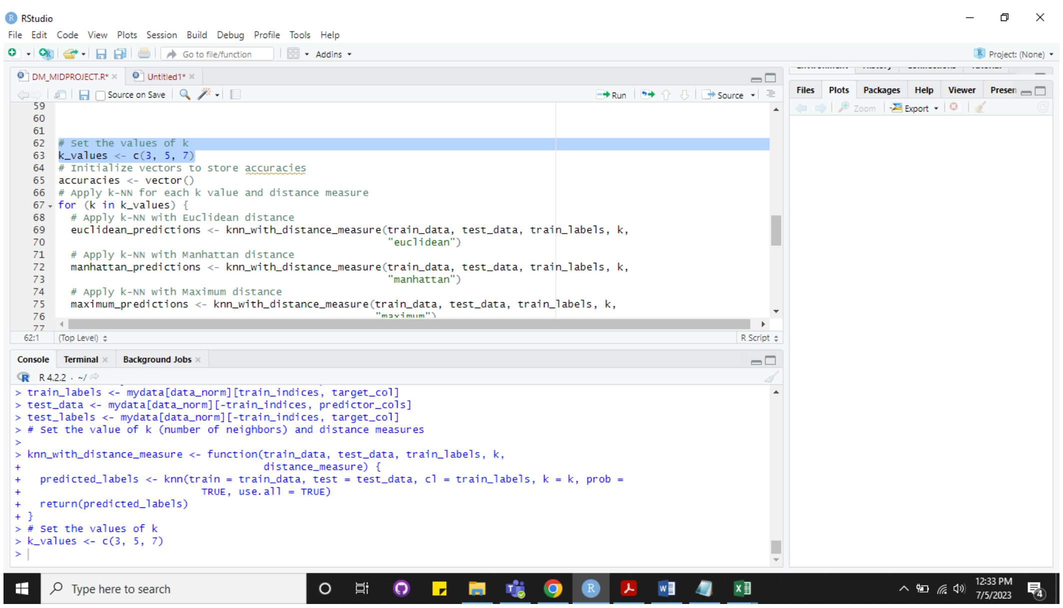
6. Splitting into Training and Test data



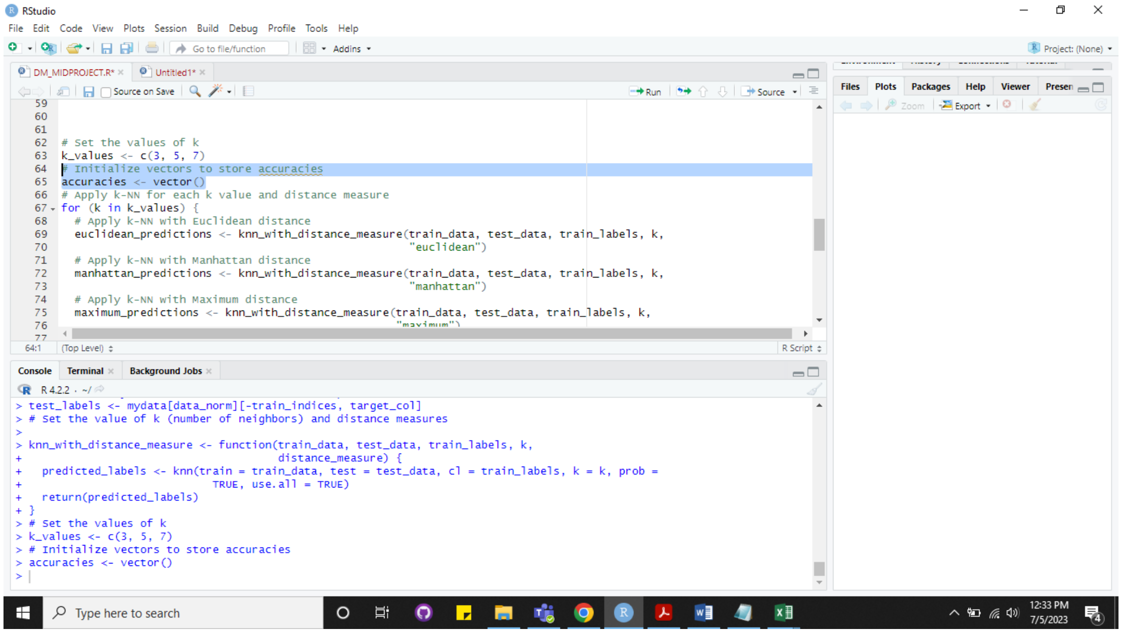
7. Implementing KNN algorithm



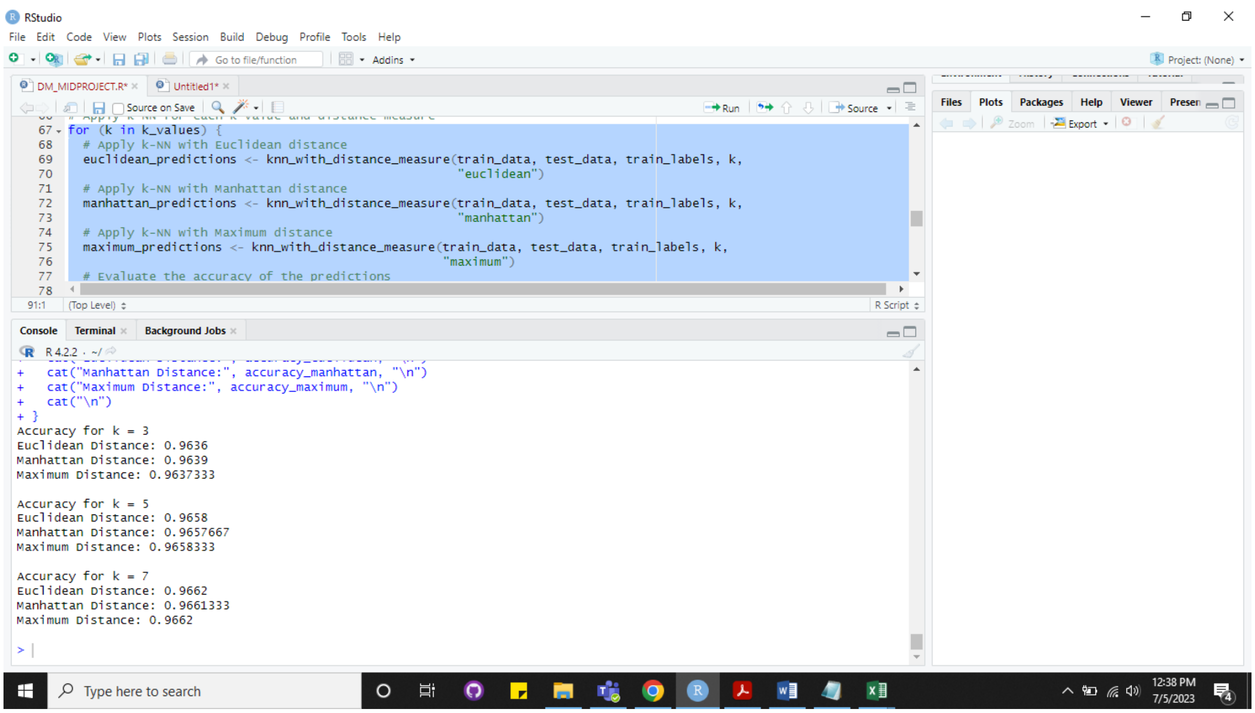
8. Initialize K values



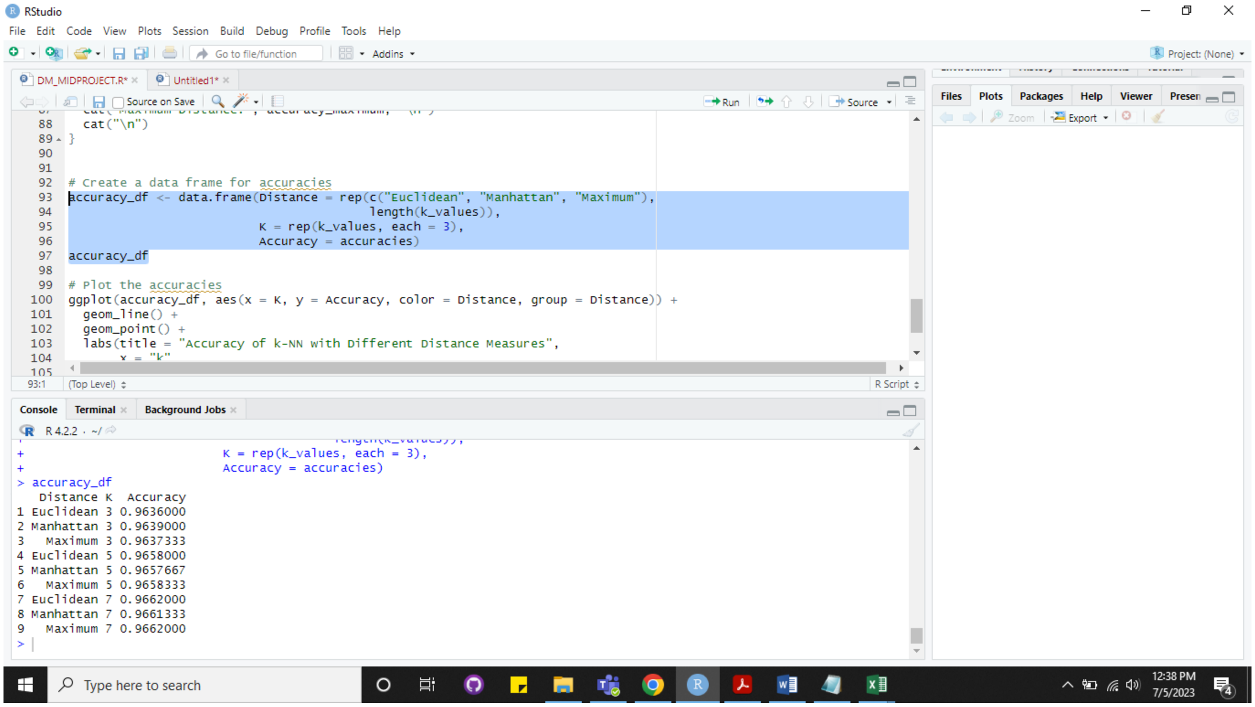
9. Initializing vectors to store accuracies



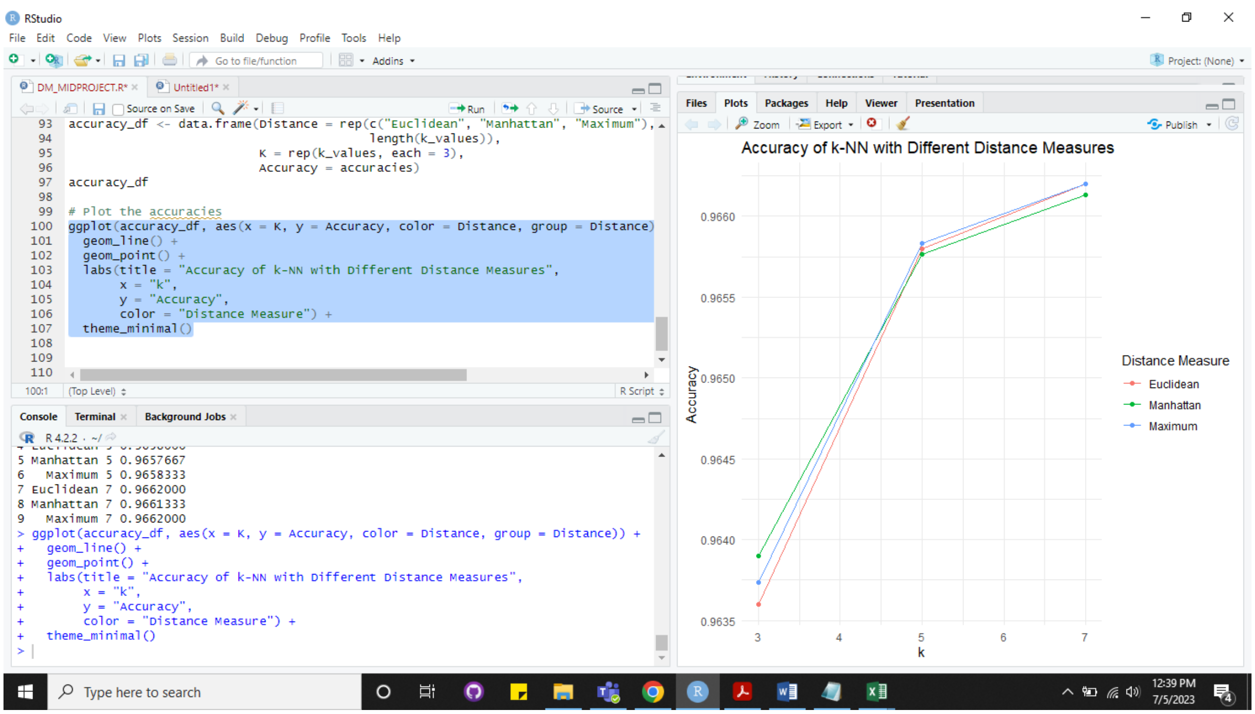
10. Apply k-NN for each k value and distance measure



11. Create Data Frame for accuracy



12. Plotting the accuracies



The graph above shows how the accuracy of the knn algorithm varies with different values of k on using the 3 distance measuring methods (Euclidean, Manhattan, and Maximum Dimension).

It can be seen that the accuracy was the highest when the value of k was 7. In this instance, Euclidean distance had the highest accuracy, followed by Maximum Dimension and then Manhattan.

When the value of k was increased to 5, the accuracies of all the distance method dropped, but the Maximum dimension was the highest, followed by Euclidean and then Manhattan.

Further, when the value was set to 3, the accuracy dropped. Also noticeable is that the Manhattan took the lead making the Euclidean last and Maximum the second last.

Among the three distance measures, Performance of Euclidean and Manhattan distance was giving the best in terms of accuracy. This indicates that for this particular dataset, this two distance measure was more suitable for distinguishing between the different attributes.