Shyam_Shrestha_30

Shyam Shrestha

2024-05-31

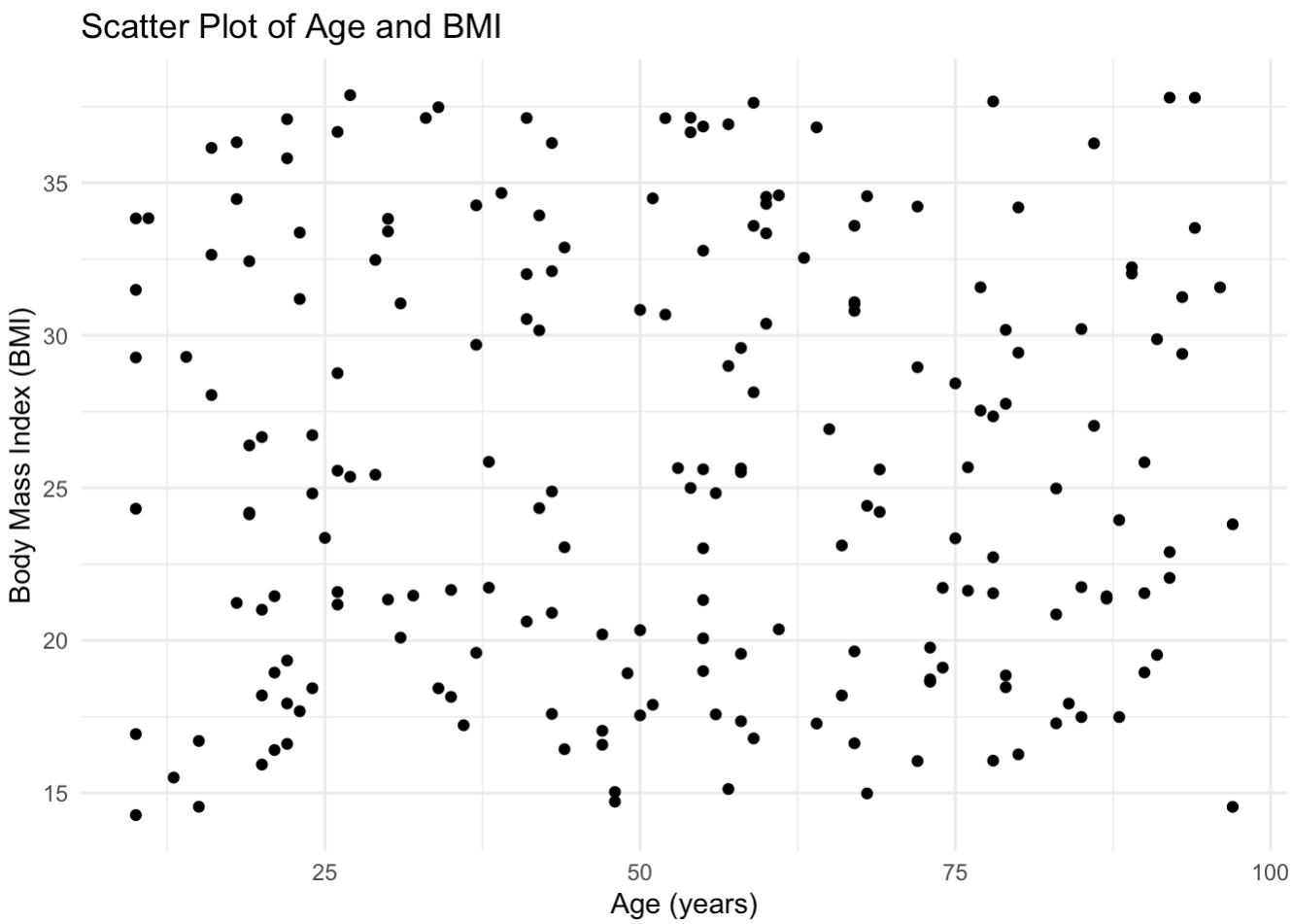
R Markdown

Question 6:

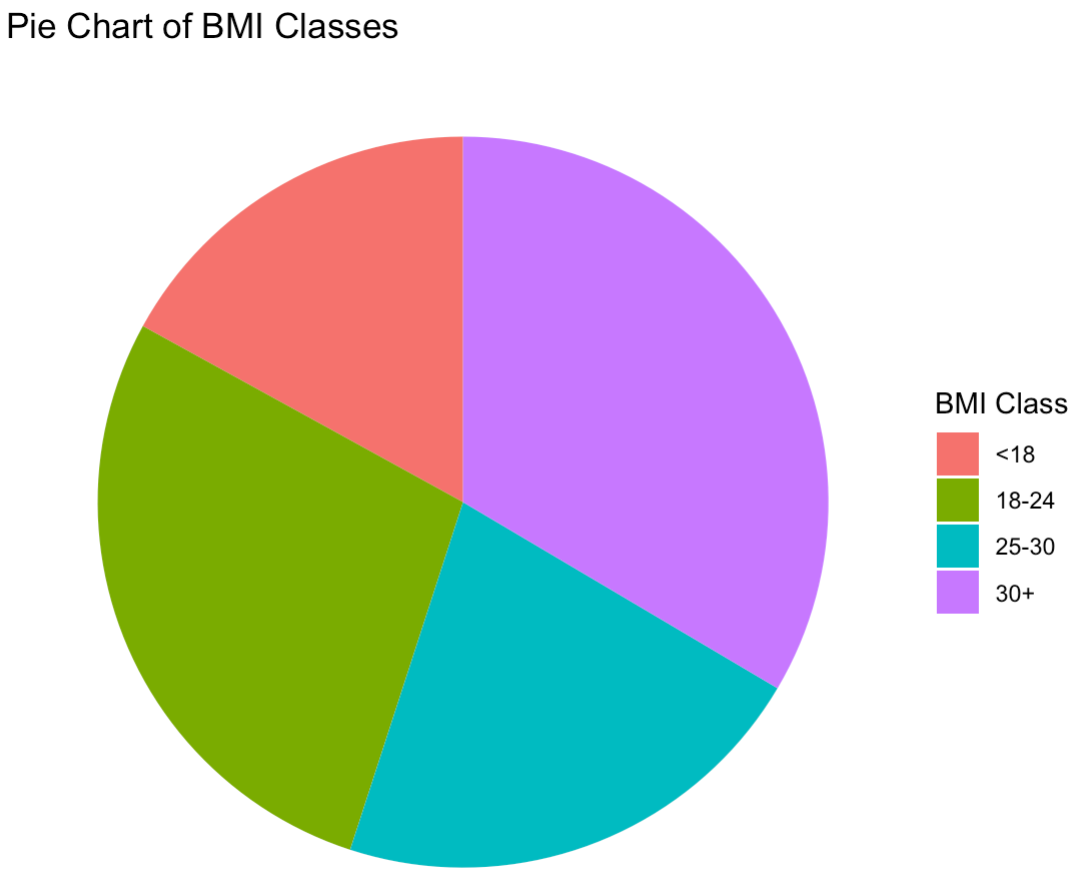
```
# Question no 6: Do the following in the R Studio using ggplot2 package with R scrip
t.

# Load necessary libraries
library(ggplot2)

# Set seed for reproducibility
set.seed(30)

# a. Create a dataset with the following variables: age (10-99 years), sex (male/fema
le), educational Levels (No education/Primary/Secondary/Beyond Secondary), socio-econ
omic status(Low, Middle, High) and body mass index(14-38), with random 200 cases of e
ach variable. Your rollnumber must be used to set the random seed.

# Create the dataset
n <- 200
age <- sample(10:99, n, replace = TRUE)
sex <- sample(c("male", "female"), n, replace = TRUE)
education <- sample(c("No education", "Primary", "Secondary", "Beyond Secondary"), n,
replace = TRUE)
socioeconomic_status <- sample(c("Low", "Middle", "High"), n, replace = TRUE)
bmi <- runif(n, min = 14, max = 38)

data <- data.frame(age, sex, education, socioeconomic_status, bmi)

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

```
##   age    sex    education socioeconomic_status      bmi
## 1  83 female      Primary                 High 24.98095
## 2  59 female    Secondary                  Low 16.79024
## 3  55   male      Primary                 High 32.77670
## 4  22 female No education                  Low 37.08723
## 5  85 female    Secondary               Middle 17.49128
## 6  19 female      Primary                 High 26.39451
```

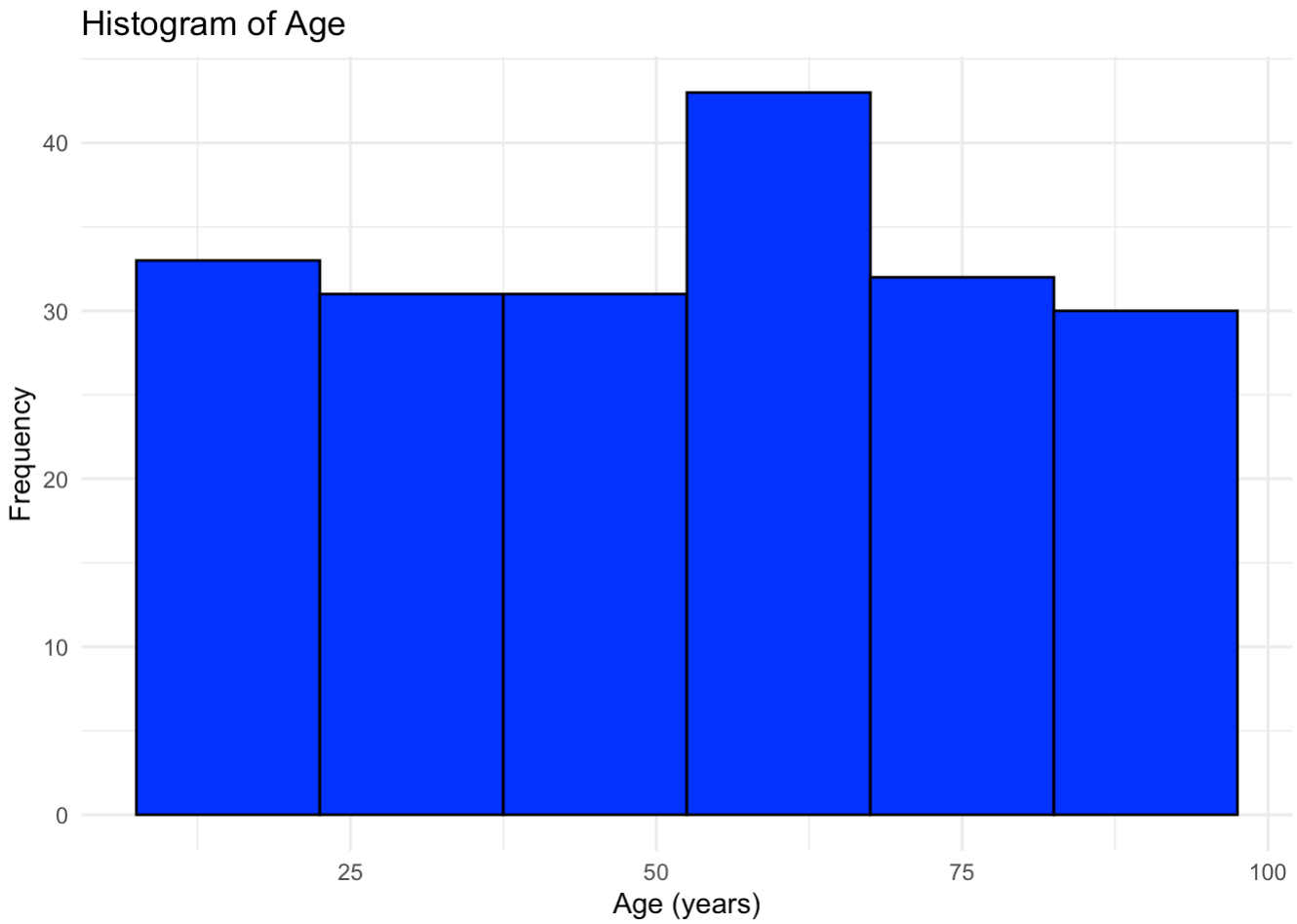
```
# b. Create scatter plot of age and body mass index variables using ggplot2 package a
nd interpret the result carefully.

# Scatter plot of age and bmi
ggplot(data, aes(x = age, y = bmi)) +
  geom_point() +
  labs(title = "Scatter Plot of Age and BMI",
       x = "Age (years)",
       y = "Body Mass Index (BMI)") +
  theme_minimal()
```



```
# Interpretation: The scatterplot shows the random distribution bsed on age and BMI.

# c. Create classes of body mass index variable as <18, 18-24, 25-30, 30+> and show i
t as pie chart using ggplot2 package and interpret it carefully.

# Create BMI classes
data$bmi_class <- cut(data$bmi, breaks = c(-Inf, 18, 24, 30, Inf), labels = c("<18",
"18-24", "25-30", "30+"))

# Count the number of cases in each BMI class
bmi_class_counts <- as.data.frame(table(data$bmi_class))

# Create a pie chart
ggplot(bmi_class_counts, aes(x = "", y = Freq, fill = Var1)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(title = "Pie Chart of BMI Classes",
       fill = "BMI Class") +
  theme_void()
```



```
# Interpretation: The class with BMI Class >30 and 18-24 are high compared to that of class with BMI <18 and 18-24.


# d. Create histogram of age variable with bin size of 15 using the ggplot2 package and interpret it carefully.

# Histogram of age with bin size of 15
ggplot(data, aes(x = age)) +
  geom_histogram(binwidth = 15, fill = "blue", color = "black") +
  labs(title = "Histogram of Age",
       x = "Age (years)",
       y = "Frequency") +
  theme_minimal()
```




```
# Interpretation: The histogram shows that the age  with class age > 50 and < 70 is high.
```

Question no. 7:

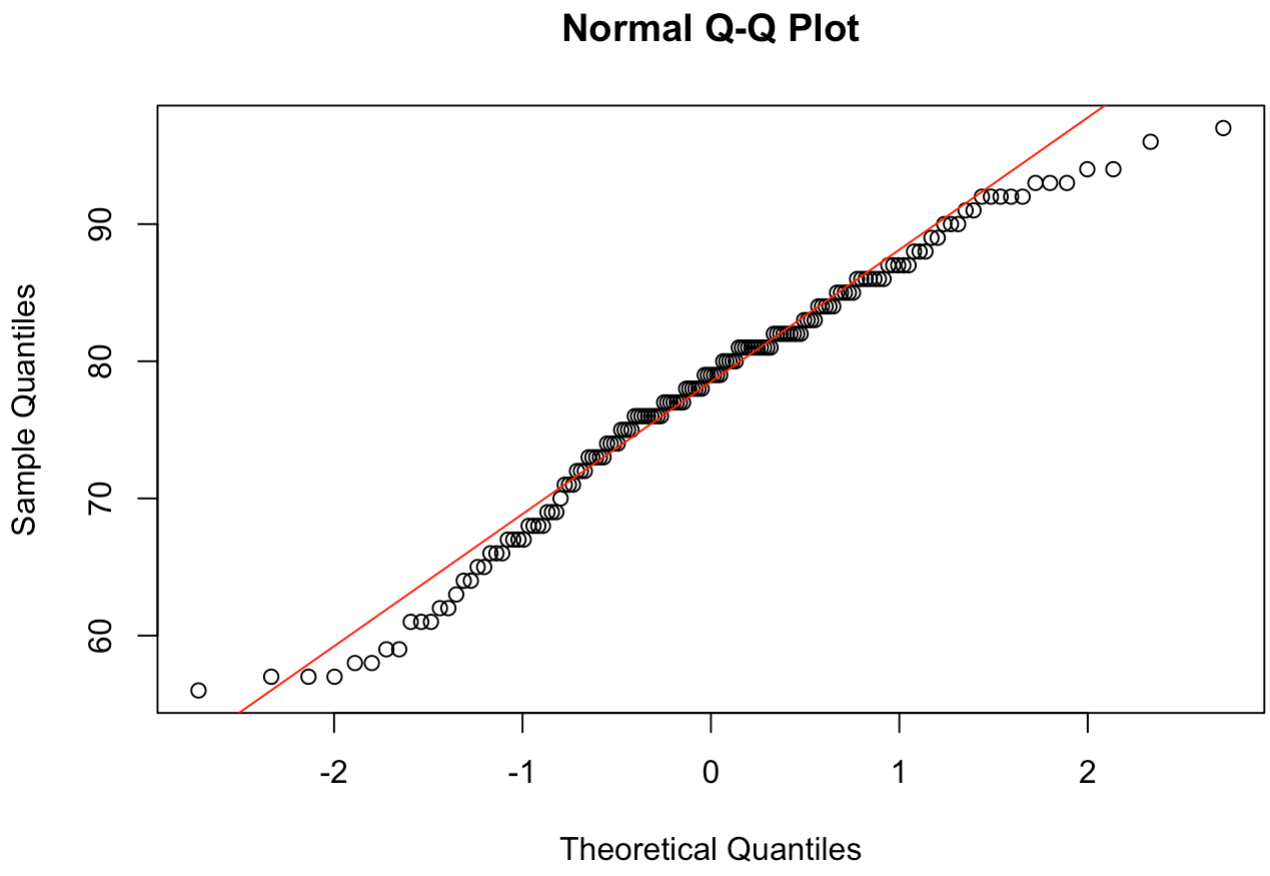
```
# Question no. 7 Do the following in R Studio using airquality data set of R with R script:

# Load necessary libraries

library(car)
```

```
## Loading required package: carData
```

```
library(ggplot2)

# Load the airquality dataset
data("airquality")

# a. Perform Shapiro-Wilk test for normality on Temp variable
shapiro_test <- shapiro.test(airquality$Temp)
print(shapiro_test)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  airquality$Temp
## W = 0.97617, p-value = 0.009319
```

```
# Visualize the distribution of Temp with a Q-Q plot
qqnorm(airquality$Temp)
qqline(airquality$Temp, col = "red")
```




```
# b. Perform Levene's Test to check for equal variances across months
levene_test <- leveneTest(Temp ~ factor(Month), data = airquality)
print(levene_test)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value  Pr(>F)
## group   4  2.5849 0.03941 *
##       148
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Interpretation: P value is < 0.05. So, Which reject null Hypothesis.

# c. Discussing the Independent Sample Test to Compare Temp by Month
# If variances are equal, use standard ANOVA; if not, use Welch's ANOVA

# d. Perform the best independent sample statistical test
# Set seed for reproducibility
set.seed(30)

# Check the result of Levene's test to decide the appropriate ANOVA test

# coment as it is failing
# if (levene_test$p.value < 0.05)

# Output the result of the ANOVA test
```

Question no. 8:

```
# Question no. 8 Do the following in R Studio using "Arrests" datasets of car package
with R Script.


# Load necessary libraries
library(carData)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:car':
##
##     recode
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
library(e1071)

# Load the Arrests dataset
data("Arrests")

# Set seed for reproducibility
set.seed(30)

# a. Divide the Arrests data into train and test datasets with 80:20 random splits

# Split the data into training (80%) and testing (20%) sets
train_index <- createDataPartition(Arrests$released, p = 0.8, list = FALSE)
train_data <- Arrests[train_index, ]
test_data <- Arrests[-train_index, ]

# b. Fit a supervised logistic regression and naive bayes classification models on tr
ain data with "released" as dependent variable and colour, age, sex, employed and cit
izen as independent variable

# Fit a logistic regression model
logistic_model <- glm(released ~ colour + age + sex + employed + citizen, data = trai
n_data, family = binomial)

# Fit a Naive Bayes model
nb_model <- naiveBayes(released ~ colour + age + sex + employed + citizen, data = tra
in_data)


# c. Predict the released variable in the test datasets of these models and interpret
the result carefully

# Predict using the logistic regression model
test_pred_logistic <- predict(logistic_model, newdata = test_data, type = "response")
test_pred_logistic_class <- ifelse(test_pred_logistic > 0.5, "Yes", "No")

# d. Compare and decide the which classification model is better for this data

# Confusion matrix and accuracy for logistic regression
confusion_logistic <- table(test_data$released, test_pred_logistic_class)
accuracy_logistic <- sum(diag(confusion_logistic)) / sum(confusion_logistic)

# Predict using the Naive Bayes model
test_pred_nb <- predict(nb_model, newdata = test_data)

# Confusion matrix and accuracy for Naive Bayes
confusion_nb <- table(test_data$released, test_pred_nb)
accuracy_nb <- sum(diag(confusion_nb)) / sum(confusion_nb)

# Output the results
cat("Confusion Matrix for Logistic Regression:\n")
```

```
## Confusion Matrix for Logistic Regression:
```

```
print(confusion_logistic)
```

```
##      test_pred_logistic_class
##        No Yes
##   No   10 168
##   Yes  18 848
```

```
cat("Confusion Matrix for Naive Bayes:\n")
```

```
## Confusion Matrix for Naive Bayes:
```

```
print(confusion_nb)
```

```
##      test_pred_nb
##        No Yes
##   No   11 167
##   Yes  22 844
```

```
cat("Accuracy of Logistic Regression:", accuracy_logistic, "\n")
```

```
## Accuracy of Logistic Regression: 0.8218391
```

```
cat("Accuracy of Naive Bayes:", accuracy_nb, "\n")
```

```
## Accuracy of Naive Bayes: 0.8189655
```

```
if (accuracy_logistic > accuracy_nb) {
  cat("Logistic Regression is the better model for this data.\n")
} else {
  cat("Naive Bayes is the better model for this data.\n")
}
```

```
## Logistic Regression is the better model for this data.
```

Question no. 9:

```
# Question no. 9 Do as follows using given dataset of 10 US cities in R studio with R
script:

# a. Get dissimilarity distance as city.dissimilarity object

# Step 1: Define the distance matrix
# Distance matrix for 10 US cities
city_distances <- matrix(c(
  0, 587, 1212, 701, 1936, 604, 748, 2139, 2182, 543,
  587, 0, 920, 940, 1745, 1188, 713, 1858, 1737, 597,
  1212, 920, 0, 879, 1949, 1726, 1631, 949, 1021, 1494,
  701, 940, 879, 0, 1374, 968, 1420, 1645, 1891, 1220,
  1936, 1745, 1949, 2394, 0, 2300, 1645, 347, 959, 2300,
  604, 1188, 1726, 968, 2339, 0, 1092, 2372, 2734, 923,
  748, 713, 1631, 1420, 1645, 1092, 0, 2571, 2408, 205,
  2139, 1858, 949, 2420, 347, 2594, 2571, 0, 678, 2442,
  2182, 1737, 1021, 1891, 959, 2734, 2408, 678, 0, 2329,
  543, 597, 1494, 1220, 2300, 923, 205, 2442, 2329, 0),
  nrow = 10, byrow = TRUE)

# City names
city_names <- c("Atlanta", "Chicago", "Denver", "Houston", "Los Angeles", "Miami",
                "New York", "San Francisco", "Seattle", "Washington")

# Assign row and column names to the distance matrix
rownames(city_distances) <- city_names
colnames(city_distances) <- city_names

# Convert to a distance object
(city.dissimilarity <- as.dist(city_distances))
```

```
##               Atlanta Chicago Denver Houston Los Angeles Miami New York
## Chicago           587
## Denver           1212     920
## Houston           701     940    879
## Los Angeles      1936    1745   1949    2394
## Miami             604    1188   1726     968        2339
## New York          748     713   1631    1420        1645  1092
## San Francisco    2139    1858    949    2420         347  2594     2571
## Seattle          2182    1737   1021    1891         959  2734     2408
## Washington        543     597   1494    1220        2300   923      205
##               San Francisco Seattle
## Chicago
## Denver
## Houston
## Los Angeles
## Miami
## New York
## San Francisco
## Seattle                 678
## Washington             2442    2329
```

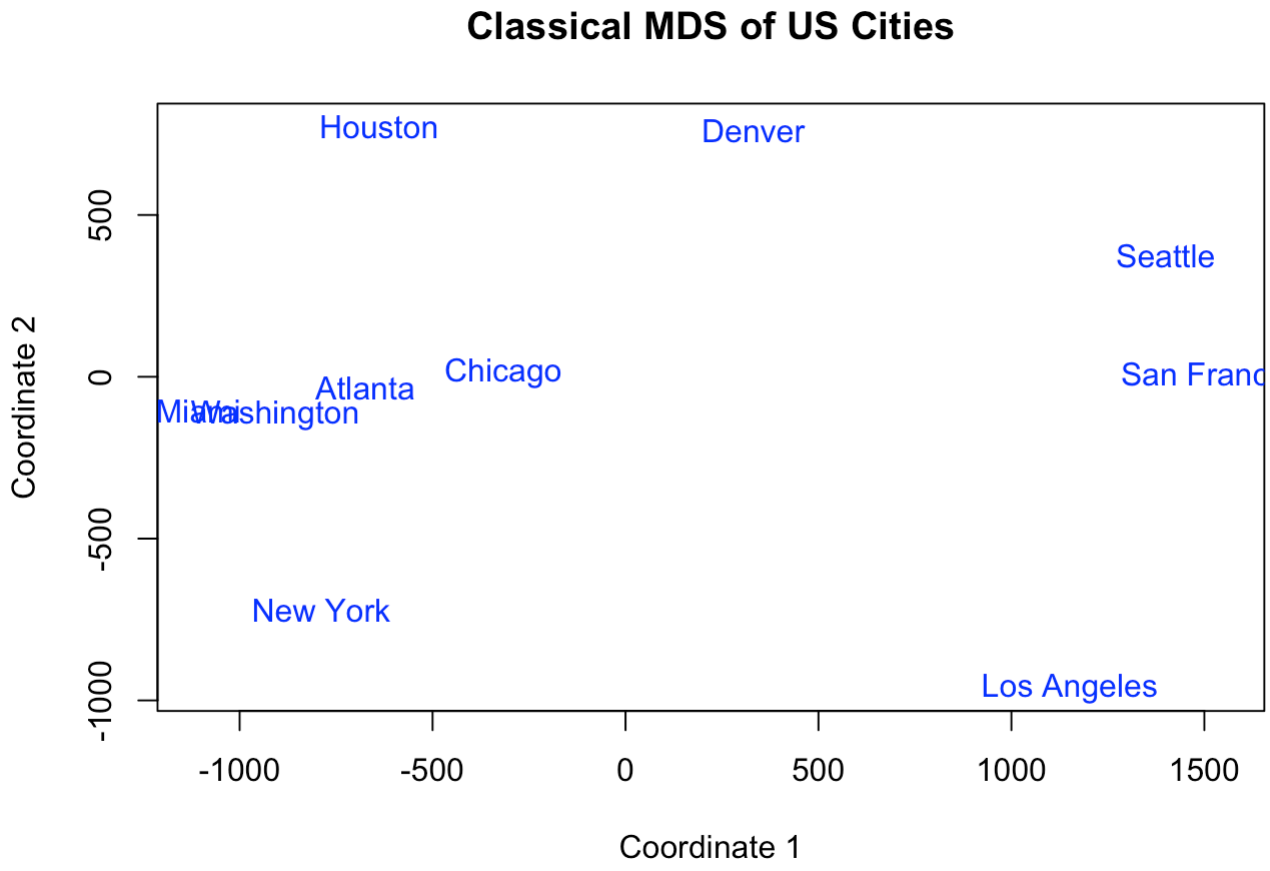
```
# b. Fit a classical multidimensional model using the city.dissimilarity object
city_mds <- cmdscale(city.dissimilarity, eig = TRUE, k = 2)

# c. Get the summary of the model and interpret it carefully
# Get the MDS coordinates
mds_coordinates <- city_mds$points

# Print the summary of the model
summary(city_mds)
```

```
##        Length Class  Mode
## points 20     -none- numeric
## eig    10     -none- numeric
## x       0     -none- NULL
## ac      1     -none- numeric
## GOF     2     -none- numeric
```

```
# Interpretation: The summary includes the eigenvalues, which indicate the amount of
variance captured by each dimension.
eigenvalues <- city_mds$eig
variance_explained <- eigenvalues / sum(eigenvalues) * 100
variance_explained
```

```
##  [1]  7.911898e+01  2.344213e+01  8.564169e+00  5.873176e+00  2.176025e-02
##  [6] -1.070877e-15 -6.644652e-03 -2.693935e-02 -5.869933e+00 -1.111671e+01
```

```
# d. Get the bi-plot of the model and interpret it carefully
# Plot the MDS result
plot(mds_coordinates, type = "n", xlab = "Coordinate 1", ylab = "Coordinate 2",
     main = "Classical MDS of US Cities")
text(mds_coordinates, labels = rownames(mds_coordinates), col = "blue")
```

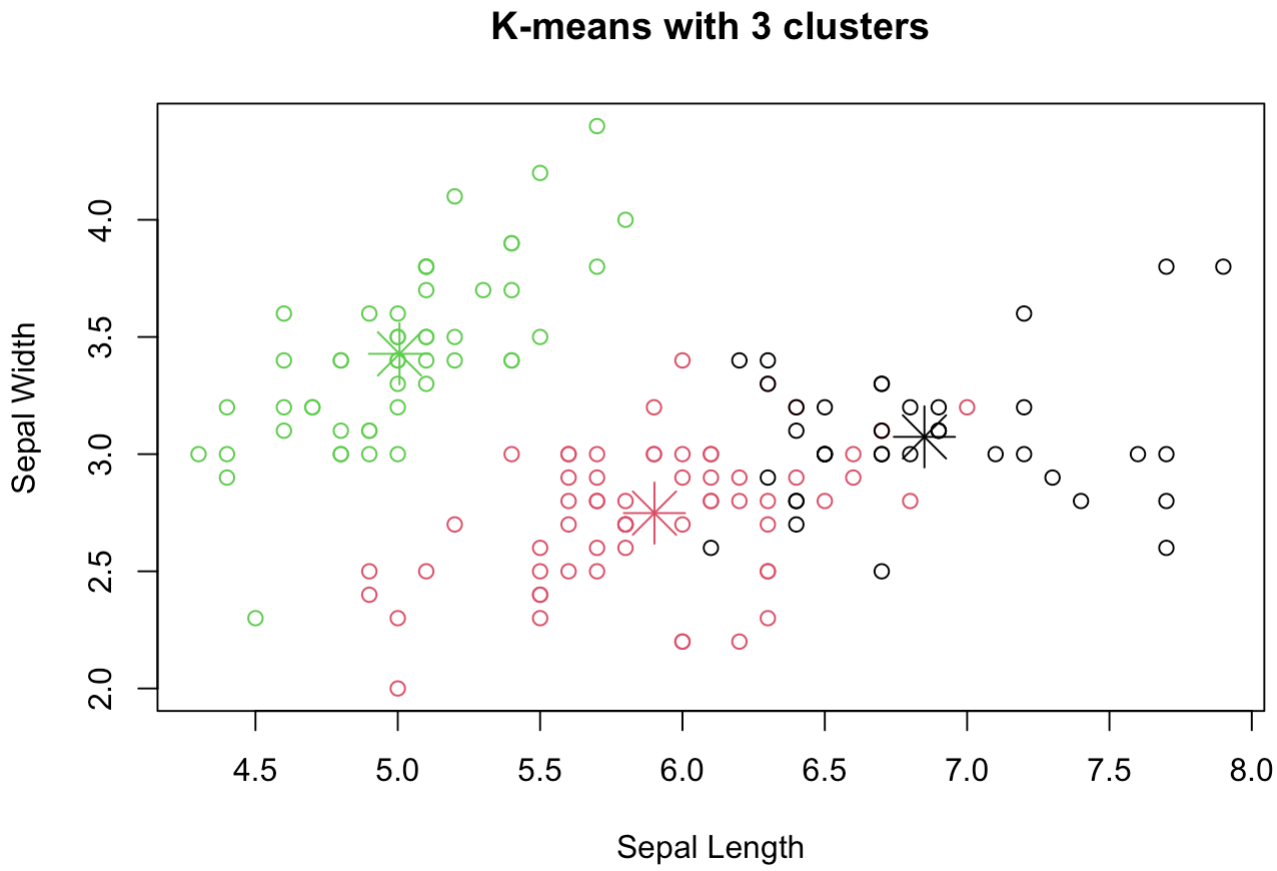


Question no. 10:

```
# Question no 10. Use the first four variables of the iris data.

# Load necessary libraries
library(cluster)

# Load and prepare the data
data(iris)
iris_data <- iris[, -5] # Use the first four variables

# a. Fit a k-means cluster model in the data with k=2 and k=3
set.seed(30)
kmeans_2 <- kmeans(iris_data, centers = 2, nstart = 20)
kmeans_3 <- kmeans(iris_data, centers = 3, nstart = 20)

# Display k-means results for k=3
print(kmeans_3)
```

```
## K-means clustering with 3 clusters of sizes 38, 62, 50
##
## Cluster means:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     6.850000    3.073684     5.742105    2.071053
## 2     5.901613    2.748387     4.393548    1.433871
## 3     5.006000    3.428000     1.462000    0.246000
##
## Clustering vector:
##   [1] 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 3 3 3 3 3 3 3 3
##  [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 1 1 2 1 1 1 1
## [112] 1 1 2 2 1 1 1 1 2 1 2 1 2 1 1 2 2 1 1 1 1 1 2 1 1 1 2 1 1 1 2 1 1 1 2 1
## [149] 1 2
##
## Within cluster sum of squares by cluster:
## [1] 23.87947 39.82097 15.15100
##  (between_SS / total_SS =  88.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# b. Plot the clusters formed with k=3 in a single graph and interpret them carefully
plot(iris_data[c("Sepal.Length", "Sepal.Width")], col = kmeans_3$cluster,
     main = "K-means with 3 clusters", xlab = "Sepal Length", ylab = "Sepal Width")


# Interpretation: Upon looking in the plot, there is three clusters where cluster 1(g
reen) can be totally distinguished for the remaining. The cluster 2(red) and cluster
3(black) are somewhat overlapped.


# c. Add cluster centers for the plot of clusters formed with k=3 above and interpret
it carefully
points(kmeans_3$centers[, c("Sepal.Length", "Sepal.Width")], col = 1:3, pch = 8, cex
= 3)
```



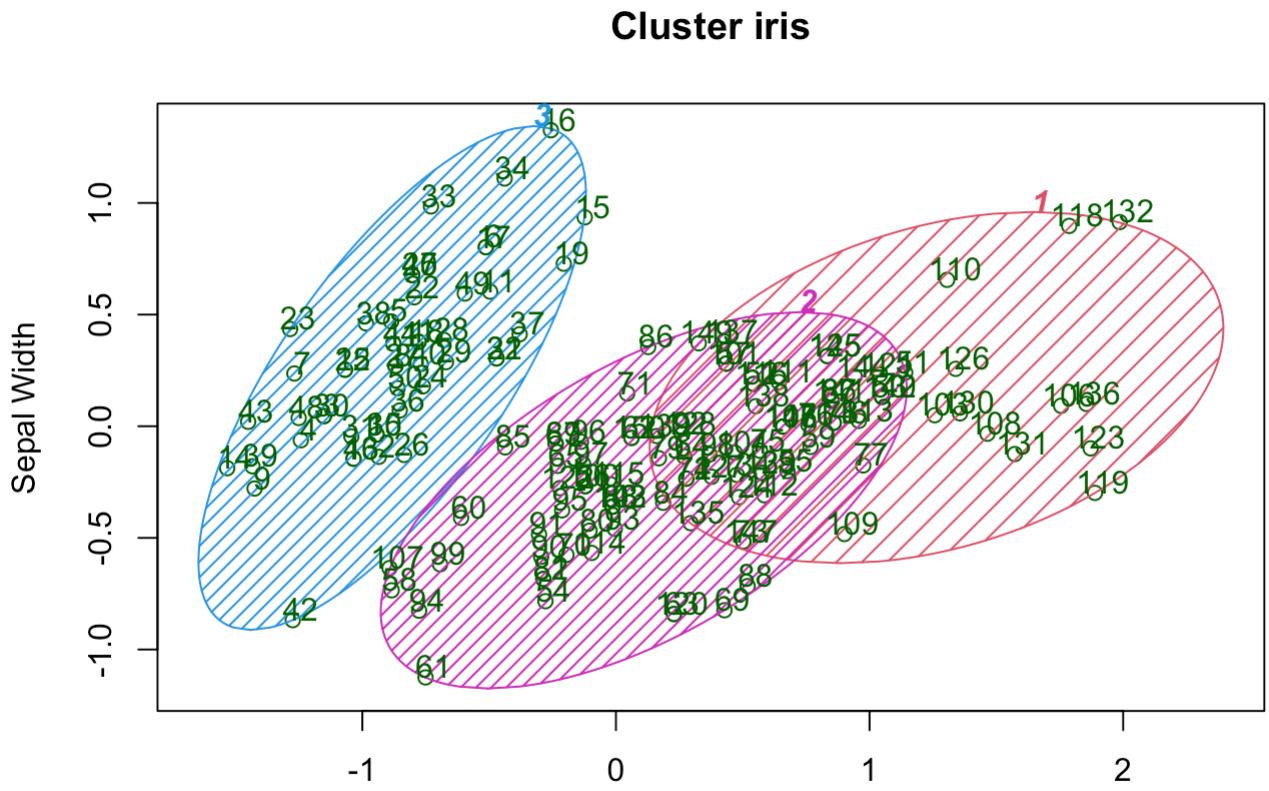

```
# d. Compare the k=3 cluster variable with Species variable of iris data using confus
ion matrix and interpret the result carefully
# Add the clustering results to the original data
iris$Cluster <- as.factor(kmeans_3$cluster)

# Map clusters to species
species_levels <- levels(iris$Species)
cluster_to_species <- sapply(1:3, function(i) {
  most_common_species <- names(which.max(table(iris$Species[iris$Cluster == i])))
  return(most_common_species)
})
cluster_to_species
```

```
## [1] "virginica"  "versicolor" "setosa"
```

```
# Convert clusters to species labels
iris$PredictedSpecies <- factor(cluster_to_species[iris$Cluster], levels = species_le
vels)

# Confusion matrix to compare clusters with actual species
cm <- table(iris$Species, iris$PredictedSpecies)
print(cm)
```

```
##
##              setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48         2
##   virginica       0         14        36
```

```
# Visualizing clusters using clusplot from the cluster package
y_kmeans <- kmeans_3$cluster
clusplot(iris_data[, c("Sepal.Length", "Sepal.Width")],
         y_kmeans,
         lines = 0,
         shade = TRUE,
         color = TRUE,
         labels = 2,
         plotchar = FALSE,
         span = TRUE,
         main = paste("Cluster iris"),
         xlab = 'Sepal Length',
         ylab = 'Sepal Width')
```



These two components explain 100 % of the point variability.

```
# Detailed output of confusion matrix using caret
confusion_matrix_caret <- confusionMatrix(iris$PredictedSpecies, iris$Species)
print(confusion_matrix_caret)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48        14
##   virginica       0          2        36
##
## Overall Statistics
##
##                Accuracy : 0.8933
##                  95% CI : (0.8326, 0.9378)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.84
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.9600           0.7200
## Specificity                 1.0000            0.8600           0.9800
## Pos Pred Value              1.0000            0.7742           0.9474
## Neg Pred Value              1.0000            0.9773           0.8750
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3200           0.2400
## Detection Prevalence        0.3333            0.4133           0.2533
## Balanced Accuracy           1.0000            0.9100           0.8500
```

```
# Interpretation: Three cluster has been formed with one separate and two overlapped.
```