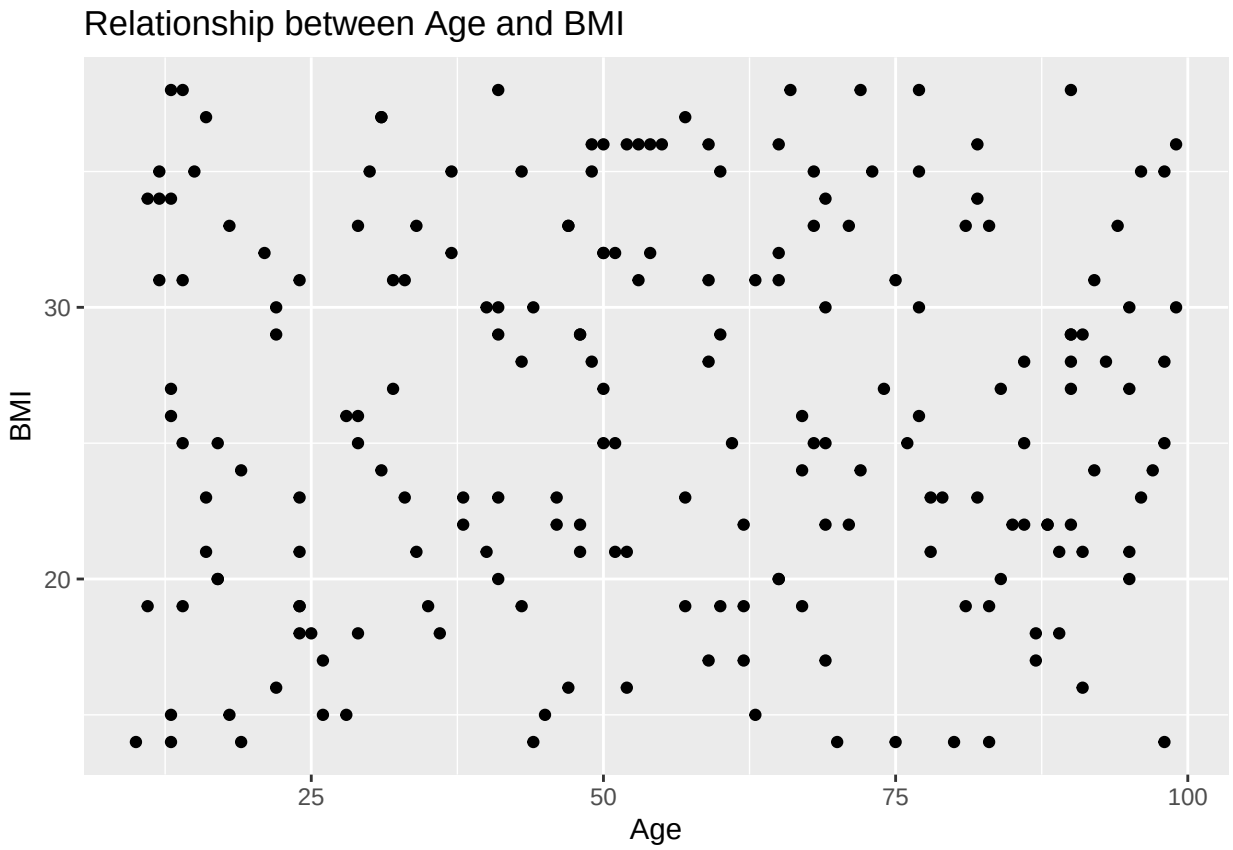
33-Second-Term.R

Suman Paudel

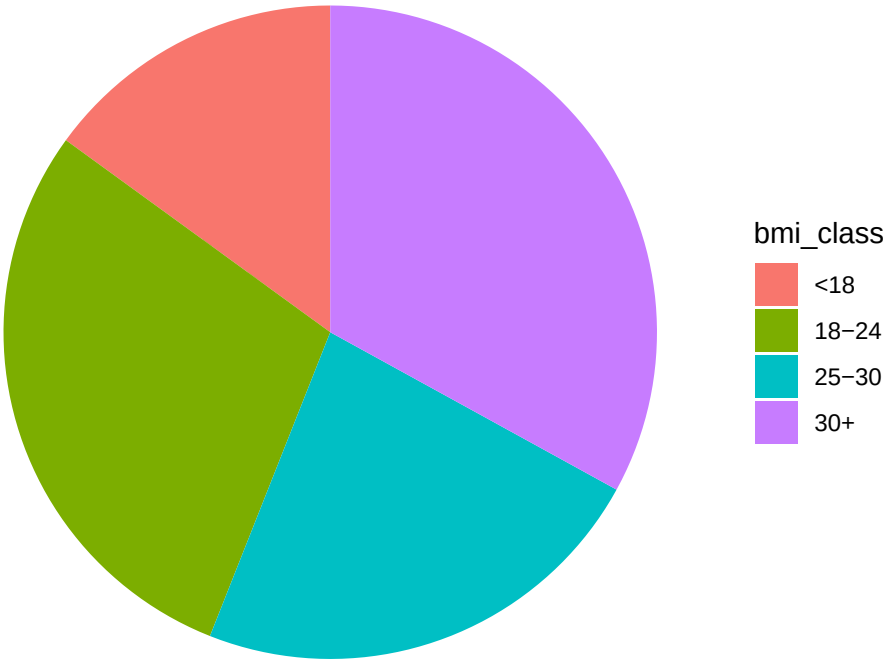
2024-05-31

```r
# Question No 6

library(ggplot2)
set.seed(33)

# a
age <- sample(10:99, 200, replace = TRUE)
sex <- sample(c("Male", "Female"), 200, replace = TRUE)
education <- sample(c("No education", "Primary", "Secondary", "Beyond secondary"), 200, replace = TRUE)
socioeconomic_status <- sample(c("Low", "Middle", "High"), 200, replace = TRUE)
bmi <- sample(14:38, 200, replace = TRUE)

# b

ggplot(data = data.frame(age, bmi), aes(x = age, y = bmi)) +
  geom_point() +
  labs(x = "Age", y = "BMI", title = "Relationship between Age and BMI")
```

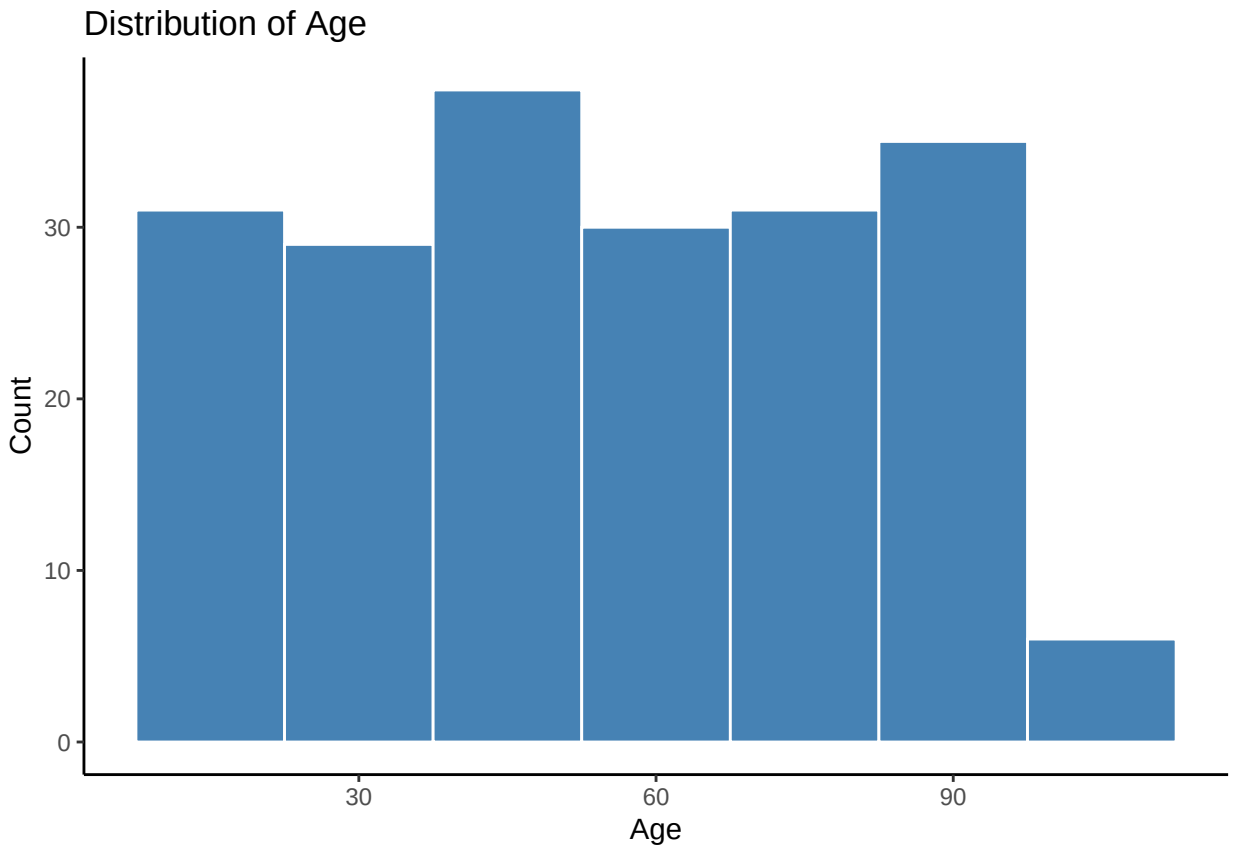



```r
# There seems to be nor trend at all and the data is scattered all over

# c
bmi_class <- cut(bmi, breaks = c(0, 18, 24, 30, Inf), labels = c("<18", "18-24", "25-30", "30+"))

# pie chart
ggplot(data.frame(bmi_class), aes(x = "", fill = bmi_class)) +
  geom_bar(width = 1) +
  coord_polar("y", start = 0) +
  labs(title = "Distribution of BMI Classes") +
  theme_void() +
  theme(legend.position = "right")
```

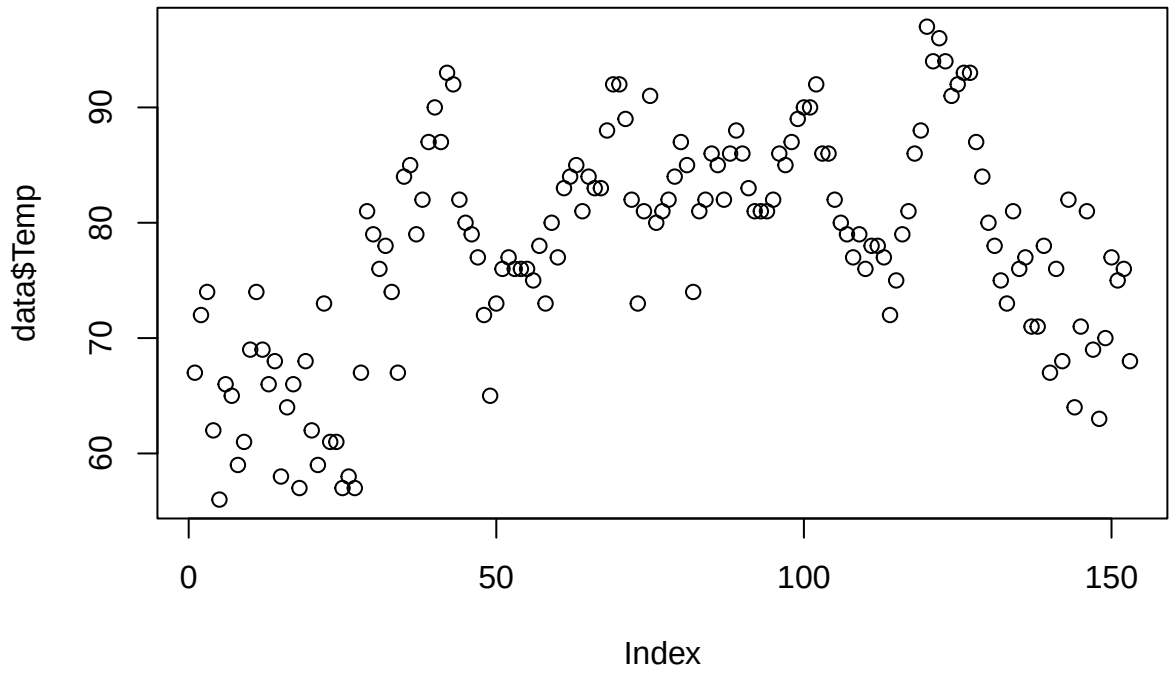
Distribution of BMI Classes



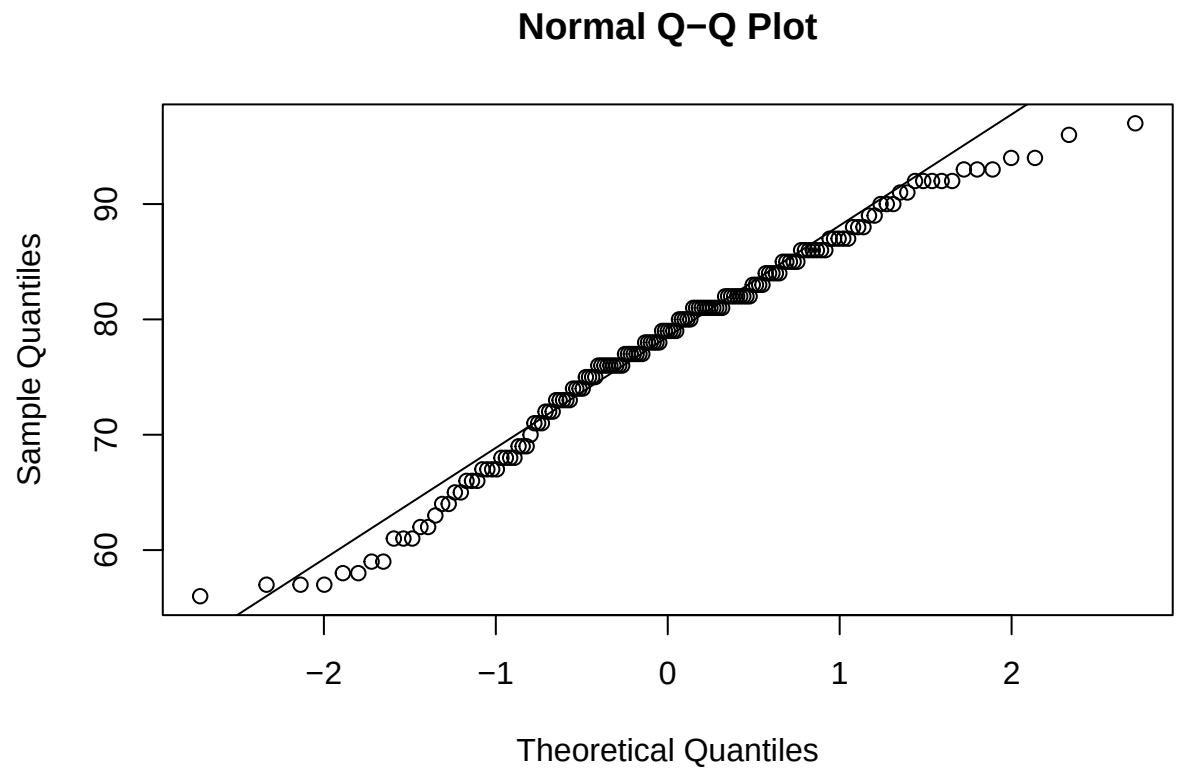

```
# here first I cut the bin to 5 classes then using ggplot I created pie chart.
# to create pie chart first I created bar then using coord_polar(), I created pie
# chart

# d

ggplot(data.frame(age), aes(x = age)) +
  geom_histogram(binwidth = 15, fill = "steelblue", color = "white") +
  labs(x = "Age", y = "Count", title = "Distribution of Age") +
  theme_classic()
```




```
# Question No 7
# Airquality dataset


data <- airquality

# a
# to perform goodness of fit test we have to do following
# check scatterplot
plot(data$Temp)
```




```
# qqplot(data$Temp)
qqnorm(data$Temp)
qqline(data$Temp)
```




```
ks.test(airquality$Temp, "pnorm")
```

```
## Warning in ks.test.default(airquality$Temp, "pnorm"): ties should not be
```

```
## present for the one-sample Kolmogorov-Smirnov test

##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  airquality$Temp
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```
# even though visually data suggests that it follows normal distribution but
# using ks.test we conclude that data doesnot follows it.
# used ks.test because sample size > 100

# var.test(airquality$Temp~airquality$Month, data = airquality)

summary(aov(airquality$Temp~airquality$Month),data=airquality)
```

```
##                   Df Sum Sq Mean Sq F value   Pr(>F)
## airquality$Month   1   2413  2413.0   32.52 6.03e-08 ***
## Residuals        151  11205    74.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# now that the pvalue is < 0.05 we need to do another test

library(car)
```

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

```
leveneTest(airquality$Temp, airquality$Month)
```

```
## Warning in leveneTest.default(airquality$Temp, airquality$Month):
## airquality$Month coerced to factor.

## 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
```

```
# now that p value is less than 0.05 we can conclude that there is no equal variance.

# b

leveneTest(Temp ~ as.factor(Month), data = airquality)
```

```
## 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
```

```r
# since p value is less than 0.05 we have to use another test



# 8
library(car)
library(e1071)
# a
data <- Arrests
ind <- sample(2, nrow(data),
              replace = T, prob = c(0.7, 0.3))
train <- data[ind==1,]
test <- data[ind==2,]

# b

# Fit the logistic regression model
logistic_model <- glm(released ~ ., data = train, family = "binomial")

# Fit the Naive Bayes model
nb_model <- naiveBayes(released ~ ., data = train)

# c
# Make predictions on the test data
logistic_pred <- predict(logistic_model, newdata = test, type = "response")
nb_pred <- predict(nb_model, newdata = test)


logistic_pred_class <- ifelse(logistic_pred > 0.5, 1, 0)
logistic_conf_matrix <- table(Predicted = logistic_pred_class, Actual = test$released)
logistic_conf_matrix
```

```
##          Actual
## Predicted   No  Yes
##         0   18   11
##         1  243 1294
```

```r
logistic_accuracy <- sum(diag(logistic_conf_matrix)) / sum(logistic_conf_matrix)
print(paste("Logistic Regression Accuracy:", logistic_accuracy))
```

```
## [1] "Logistic Regression Accuracy: 0.837803320561941"
```

```r
# True Negatives (TN): 21
# These are the cases where the actual outcome was "No" and
# the logistic regression model correctly predicted "No".
# False Negatives (FN): 31
# These are the cases where the actual outcome was "Yes" but
# the logistic regression model incorrectly predicted "No".
# False Positives (FP): 231
# These are the cases where the actual outcome was "No" but
# the logistic regression model incorrectly predicted "Yes".
# True Positives (TP): 1290
```

```r
# These are the cases where the actual outcome was "Yes" and
# the logistic regression model correctly predicted "Yes".


nb_conf_matrix <- table(Predicted = nb_pred, Actual = test$release)
nb_conf_matrix
```

```
##          Actual
## Predicted   No  Yes
##       No    49   43
##       Yes  212 1262
```

```r
nb_accuracy <- sum(diag(nb_conf_matrix)) / sum(nb_conf_matrix)
print(paste("Naive Bayes Accuracy:", nb_accuracy))
```

```
## [1] "Naive Bayes Accuracy: 0.837164750957854"
```

```r
# True Negatives (TN): 46
# These are the cases where the actual outcome was "No" and
# the Naive Bayes model correctly predicted "No".
# False Negatives (FN): 81
# These are the cases where the actual outcome was "Yes" but
# the Naive Bayes model incorrectly predicted "No".
# False Positives (FP): 206
# These are the cases where the actual outcome was "No" but
# the Naive Bayes model incorrectly predicted "Yes".
# True Positives (TP): 1240
# These are the cases where the actual outcome was "Yes" and
# the Naive Bayes model correctly predicted "Yes".

# d
# Based on the performance metrics calculated from the confusion matrices,
# the logistic regression model appears to be the better performing model
# compared to the Naive Bayes model.
# Accuracy of LR Model is 83% which is slightly better than Naive Bayes Algorithms


# 9

# 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, 2182, 2234, 597,
  1212, 920, 0, 879, 1949, 1726, 1631, 949, 1021, 1494,
  701, 940, 879, 0, 2394, 968, 1420, 2420, 2442, 597,
  1936, 1745, 1949, 2394, 0, 2300, 1645, 347, 403, 2339,
  604, 1188, 1726, 968, 2300, 0, 781, 2372, 2420, 1121,
  748, 713, 1631, 1420, 1645, 781, 0, 1923, 1960, 688,
  2139, 2182, 949, 2420, 347, 2372, 1923, 0, 214, 2571,
  2182, 2234, 1021, 2442, 403, 2420, 1960, 214, 0, 2534,
```

```
  543, 597, 1494, 597, 2339, 1121, 688, 2571, 2534, 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        2300
## New York          748     713   1631    1420        1645   781
## San Francisco    2139    2182    949    2420         347  2372     1923
## Seattle          2182    2234   1021    2442         403  2420     1960
## Washington        543     597   1494     597        2339  1121      688
##               San Francisco Seattle
## Chicago
## Denver
## Houston
## Los Angeles
## Miami
## New York
## San Francisco
## Seattle                 214
## Washington             2571    2534
```

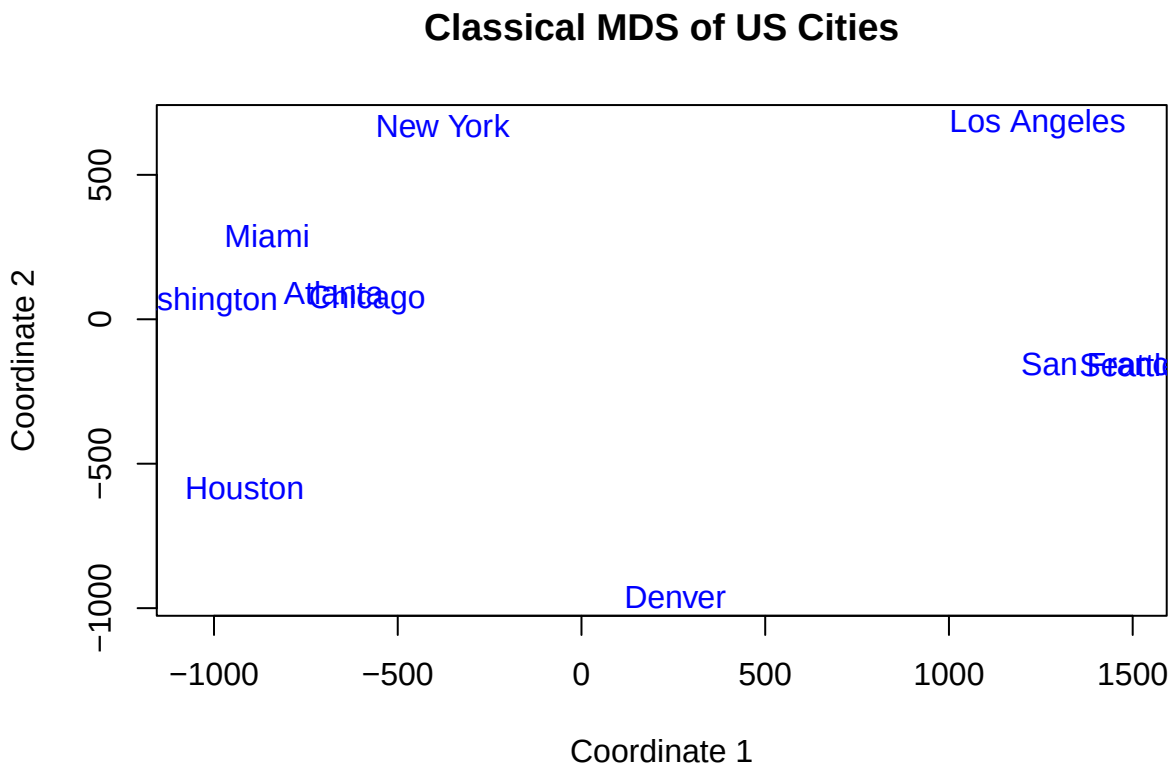
```
# b. Fit a classical multidimensional model using the city.dissimilarity object
city_mds <- cmdscale(city.dissimilarity, eig = TRUE, k = 2)


# c. Get the summary of the model and interpret it carefully
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 given 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]  8.234796e+01  1.986524e+01  8.496239e+00  2.816187e+00  1.534806e+00
##  [6]  5.687274e-02 -3.985221e-15 -3.274654e-01 -3.353130e+00 -1.143670e+01
```

```r
# 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")
```




```r
# If there are distinct clusters of points, it suggests that those cities might be
# grouped together geographically. As if plot we see few clusters of city with distance apart.
# one from east coast and another from west coast, the Denver is just outlier.

# 10
set.seed(33)
iris_data <- iris[,-5]

kmeans_2 <- kmeans(iris_data, centers = 2, nstart = 20)
kmeans_3 <- kmeans(iris_data, centers = 3, nstart = 20)

summary(kmeans_2)
```

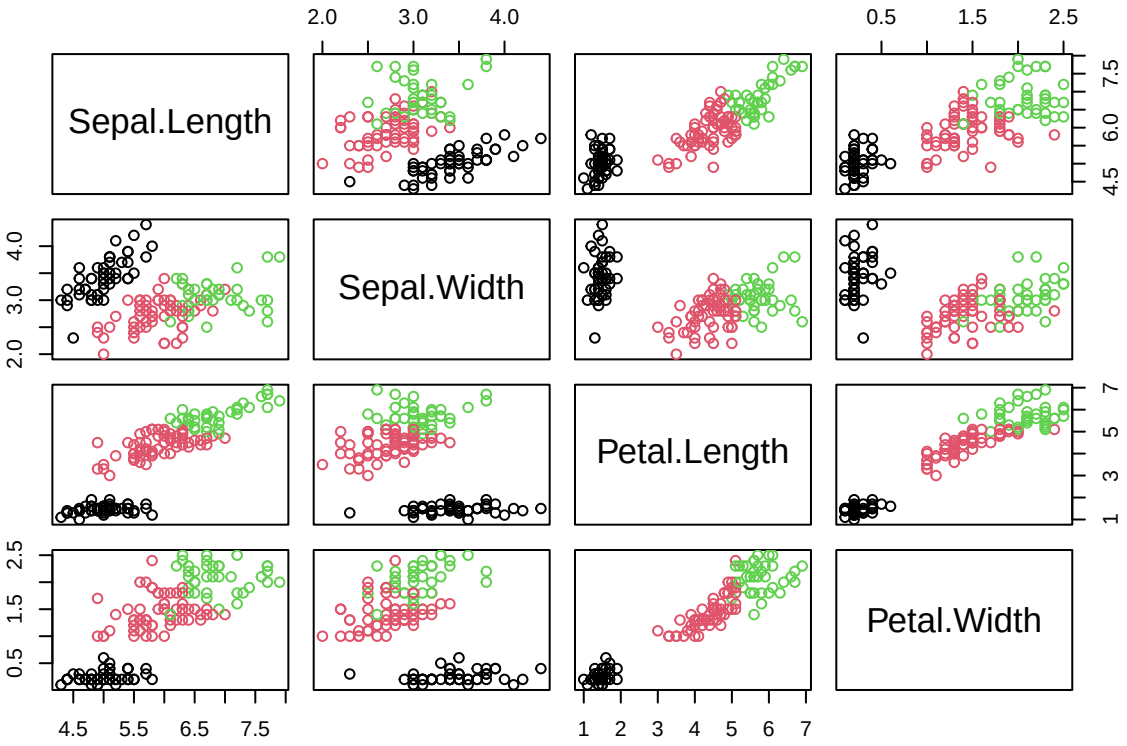
```
##              Length Class  Mode
## cluster      150    -none- numeric
## centers        8    -none- numeric
## totss          1    -none- numeric
## withinss       2    -none- numeric
## tot.withinss   1    -none- numeric
```

```
## betweenss      1    -none- numeric
## size           2    -none- numeric
## iter           1    -none- numeric
## ifault         1    -none- numeric
```

```
summary(kmeans_3)
```

```
##              Length Class  Mode
## cluster      150    -none- numeric
## centers       12    -none- numeric
## totss          1    -none- numeric
## withinss       3    -none- numeric
## tot.withinss   1    -none- numeric
## betweenss      1    -none- numeric
## size           3    -none- numeric
## iter           1    -none- numeric
## ifault         1    -none- numeric
```

```
plot(iris_data, col = kmeans_3$cluster)
points(kmeans_3$centers, col = 1:3, pch = 8, cex = 2)
```




```
# data is ploted for each variable from 1 to 3 columns

cm <- table(iris$Species,
            kmeans_3$cluster)
cm
```

```
##
```

```
##              1  2  3
##   setosa     50  0  0
##   versicolor  0 48  2
##   virginica   0 14 36
```

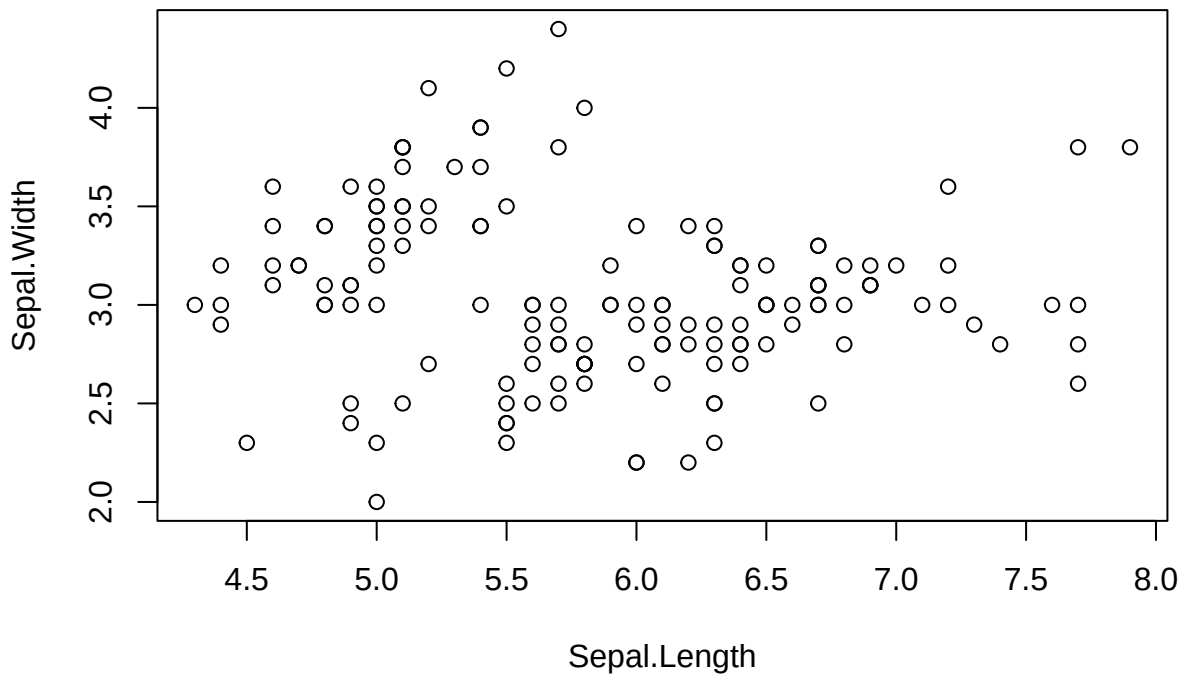
```
# the accuracy is 43%
(accuracy <-
    sum(diag(cm))/sum(cm))
```

```
## [1] 0.8933333
```

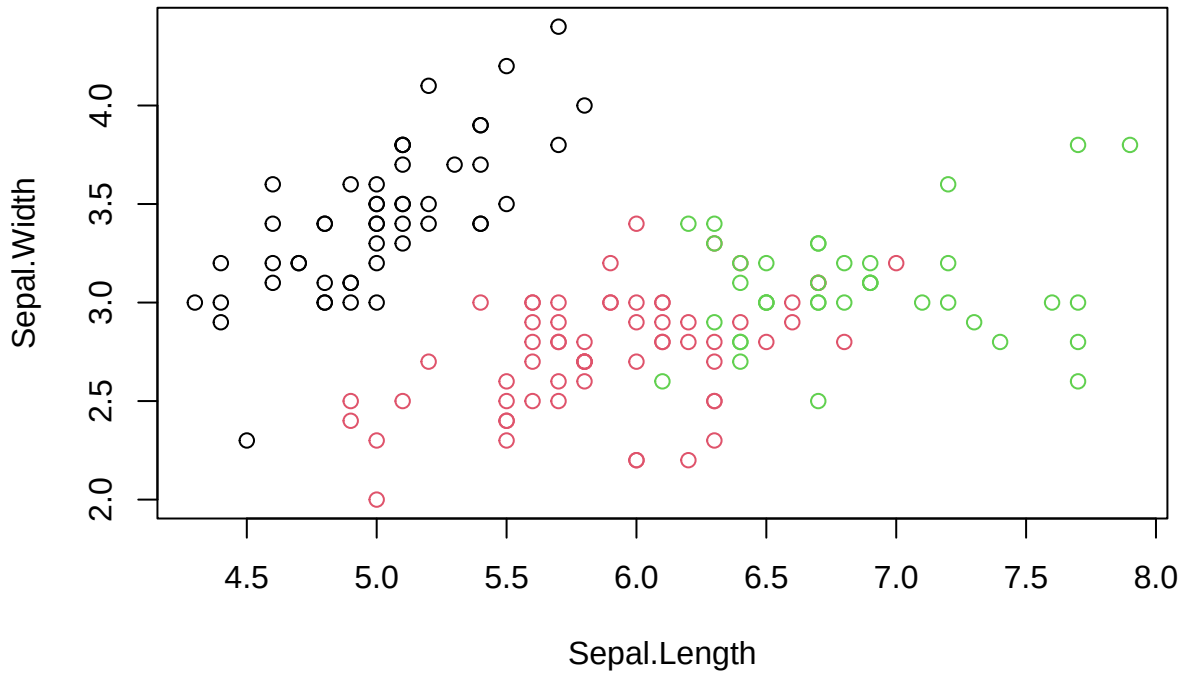
```
#
# 49 setosa flowers were assigned to cluster 1.
# 2 versicolor flowers were assigned to cluster 2.
# 36 virginica flowers were assigned to cluster 2.
# It shows that the algorithm was able to correctly identify all the
# setosa flowers and most of the versicolor and virginica flowers,
# but it had some difficulty distinguishing between versicolor and virginica flowers.
(mce <- 1 - accuracy)
```

```
## [1] 0.1066667
```

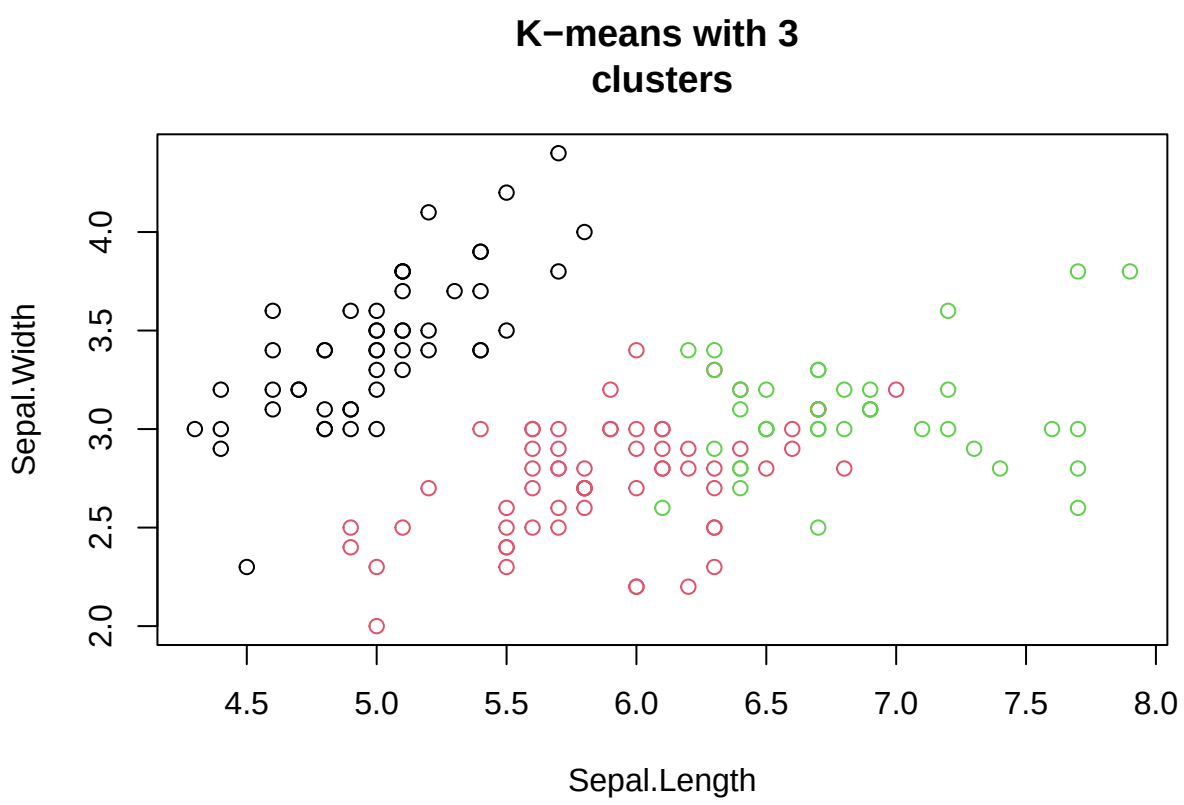
```
plot(iris_data[c("Sepal.Length",
             "Sepal.Width")])
```




```
plot(iris_data[c("Sepal.Length",
                 "Sepal.Width")],
       col = kmeans_3$cluster)
```




```
plot(iris_data[c("Sepal.Length",
                 "Sepal.Width")],
       col = kmeans_3$cluster,
       main = "K-means with 3
clusters")
```



```
# we can see that there is 3 differenct cluster
```