Second Assestment (roll 12)

Kabita Joshi

2024-05-31

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
#QN-6)
#a)

# Load necessary libraries
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

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

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

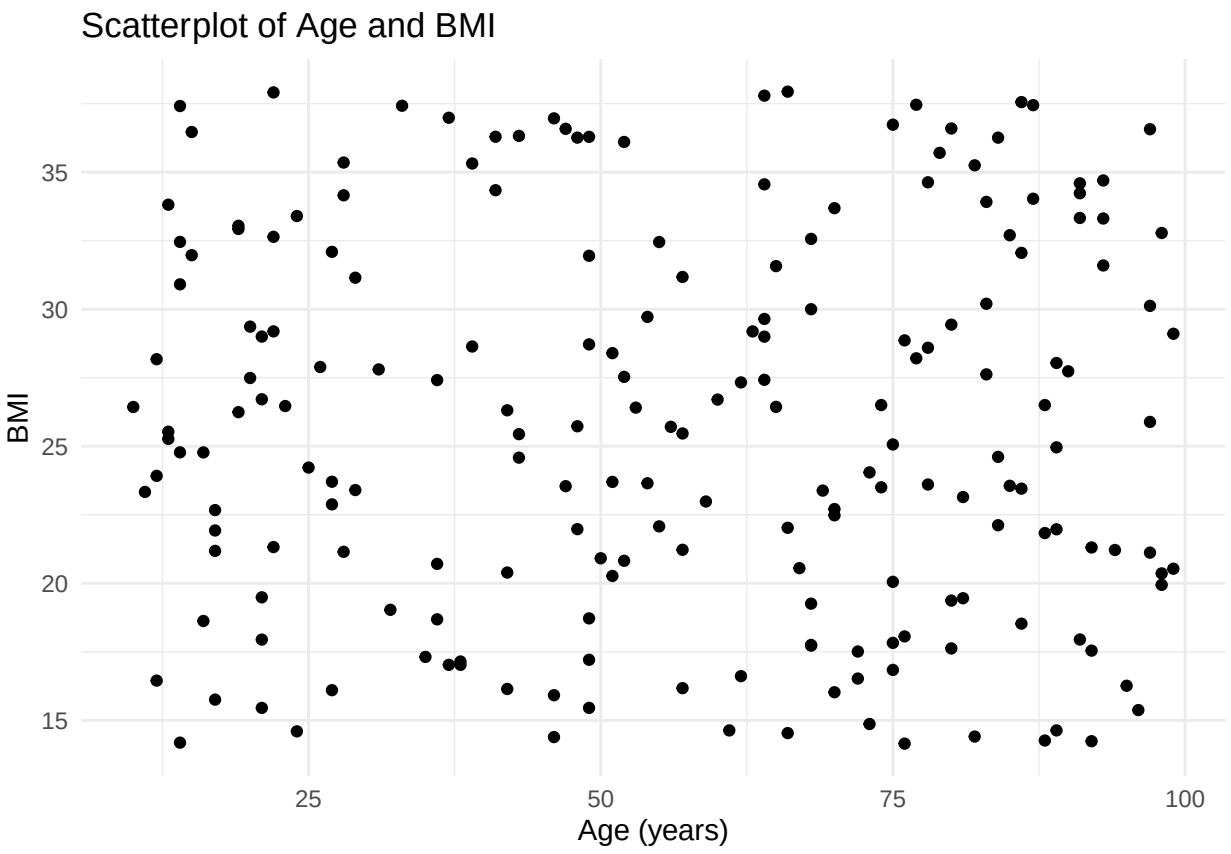
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Set random seed using the 12 roll number (e.g., 123456)
set.seed(12)

# Generate the dataset
n <- 200

data <- data.frame(
  age = sample(10:99, n, replace = TRUE),
```

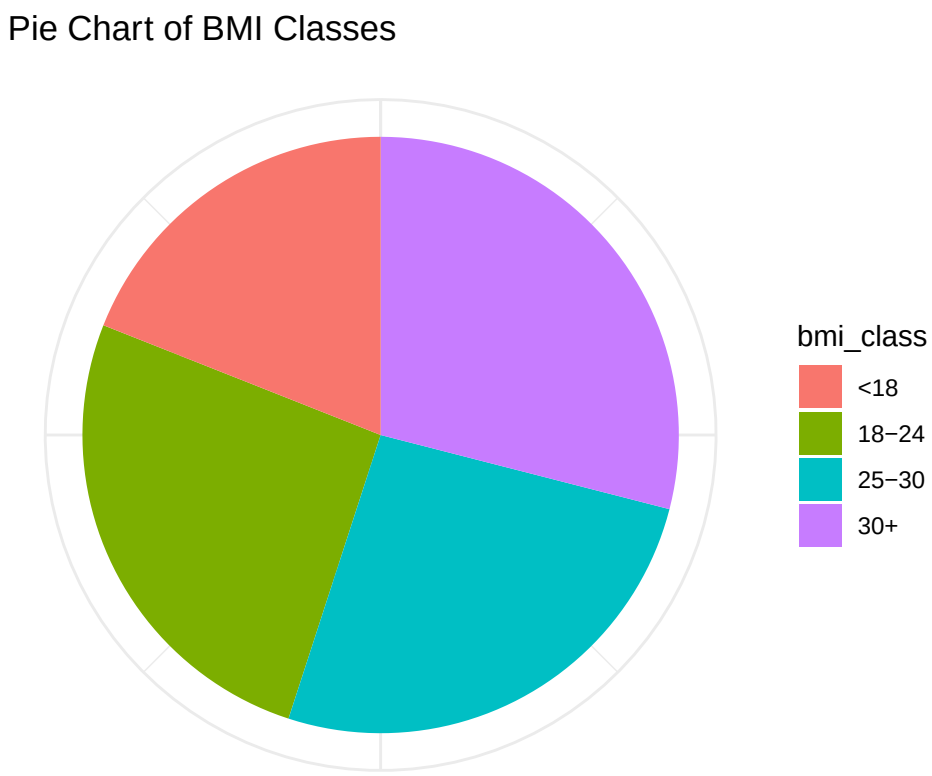
```
  sex = sample(c("Male", "Female"), n, replace = TRUE),
  educational_level = sample(c("No education", "Primary", "Secondary", "Beyond Secondary"), n, replace =
  socioeconomic_status = sample(c("Low", "Middle", "High"), n, replace = TRUE),
  bmi = runif(n, 14, 38)
)

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

```
##   age    sex educational_level socioeconomic_status      bmi
## 1  75 Female      No education               Middle 17.82718
## 2  99 Female      No education                  Low 29.10356
## 3  89   Male           Primary                 High 14.63461
## 4  55 Female      No education                 High 22.07759
## 5  78   Male      No education               Middle 23.60422
## 6  78   Male      No education               Middle 28.59271
```

```
 #b)

# Scatterplot of Age and BMI
ggplot(data, aes(x = age, y = bmi)) +
  geom_point() +
  labs(title = "Scatterplot of Age and BMI", x = "Age (years)", y = "BMI") +
  theme_minimal()
```

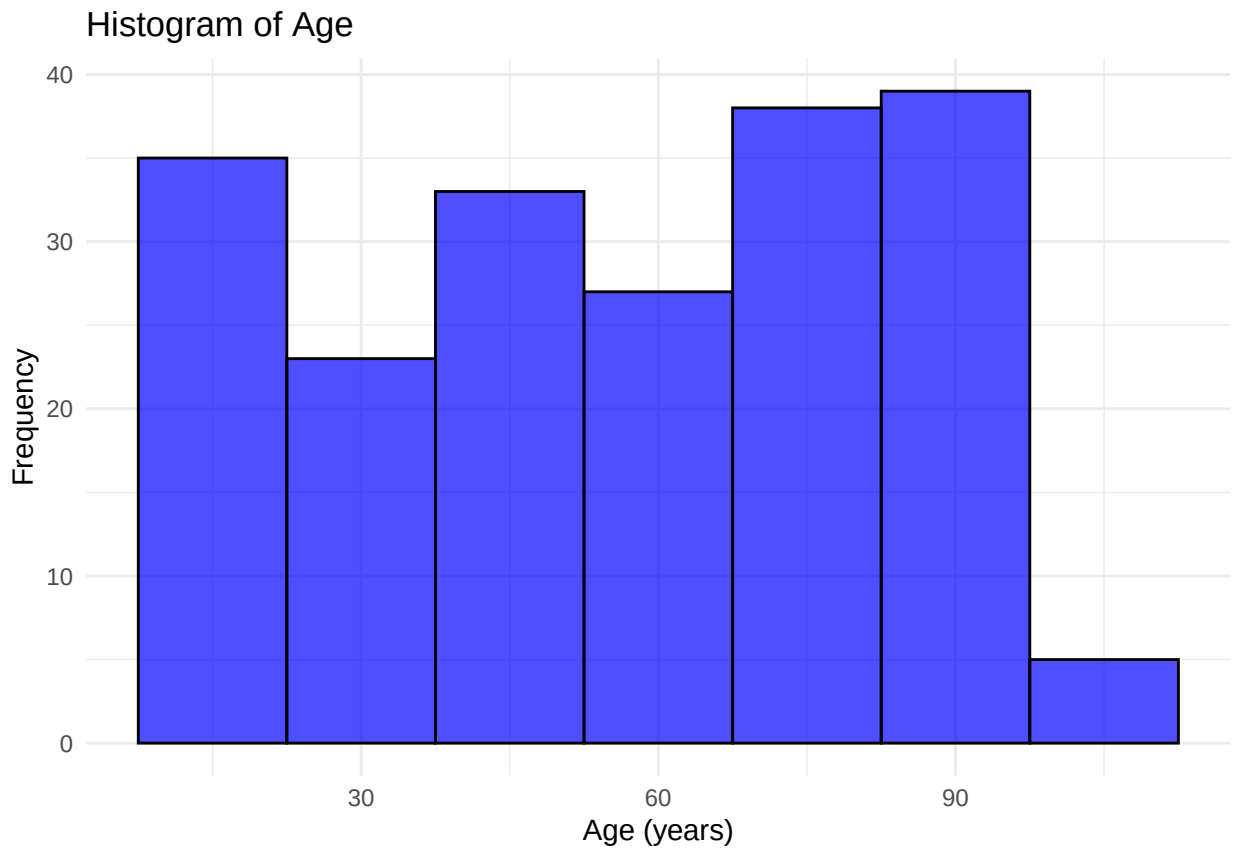


```r
#c)# Classify BMI into categories
data <- data %>%
  mutate(bmi_class = case_when(
    bmi < 18 ~ "<18",
    bmi >= 18 & bmi < 24 ~ "18-24",
    bmi >= 24 & bmi < 30 ~ "25-30",
    bmi >= 30 ~ "30+"
  ))

# Count the number of cases in each BMI class
bmi_counts <- data %>%
  group_by(bmi_class) %>%
  summarize(count = n())

# Pie chart of BMI classes
ggplot(bmi_counts, aes(x = "", y = count, fill = bmi_class)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(title = "Pie Chart of BMI Classes", x = "", y = "") +
  theme_minimal() +
  theme(axis.ticks = element_blank(), axis.text = element_blank())
```




```r
#d)
# Histogram of Age with bin size 15
ggplot(data, aes(x = age)) +
```

```
  geom_histogram(binwidth = 15, fill = "blue", color = "black", alpha = 0.7) +
  labs(title = "Histogram of Age", x = "Age (years)", y = "Frequency") +
  theme_minimal()
```



Scatter plot of Age and BMI The scatter plot will show the relationship between age and BMI. Each point represents an individual, plotting their age against their BMI. Looking for any patterns or trends, there is no such specific large clusters or outliers , it suggests that there is no strong correlation between age and BMI in this dataset.

Pie Chart of BMI Classes The pie chart will illustrate the distribution of individuals across different BMI categories (<18, 18-24, 25-30, 30+). The segment <18 represents least proportion of individual in this BMI category wherease 30+ includes highest number of individual.

Histogram of Age The histogram will show the distribution of ages within the sample, with a bin size of 15 years.In these 15 years age 90 are the age groups that are prevelant in the dataset.

```
#QN-8
# Load necessary libraries
library(ggplot2)
library(psych)
```

```
## Warning: package 'psych' was built under R version 4.3.3
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

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

# a) Perform PCA on the first four variables (Sepal.Length, Sepal.Width, Petal.Length, Petal.Width)
iris_pca <- prcomp(iris[, 1:4], center = TRUE, scale. = TRUE)

# Display PCA summary
summary(iris_pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.7084 0.9560 0.38309 0.14393
## Proportion of Variance 0.7296 0.2285 0.03669 0.00518
## Cumulative Proportion  0.7296 0.9581 0.99482 1.00000
```

```
# b) Compute eigenvalues and interpret using Kaiser's criteria
eigenvalues <- iris_pca$sdev^2
print(eigenvalues)
```

```
## [1] 2.91849782 0.91403047 0.14675688 0.02071484
```

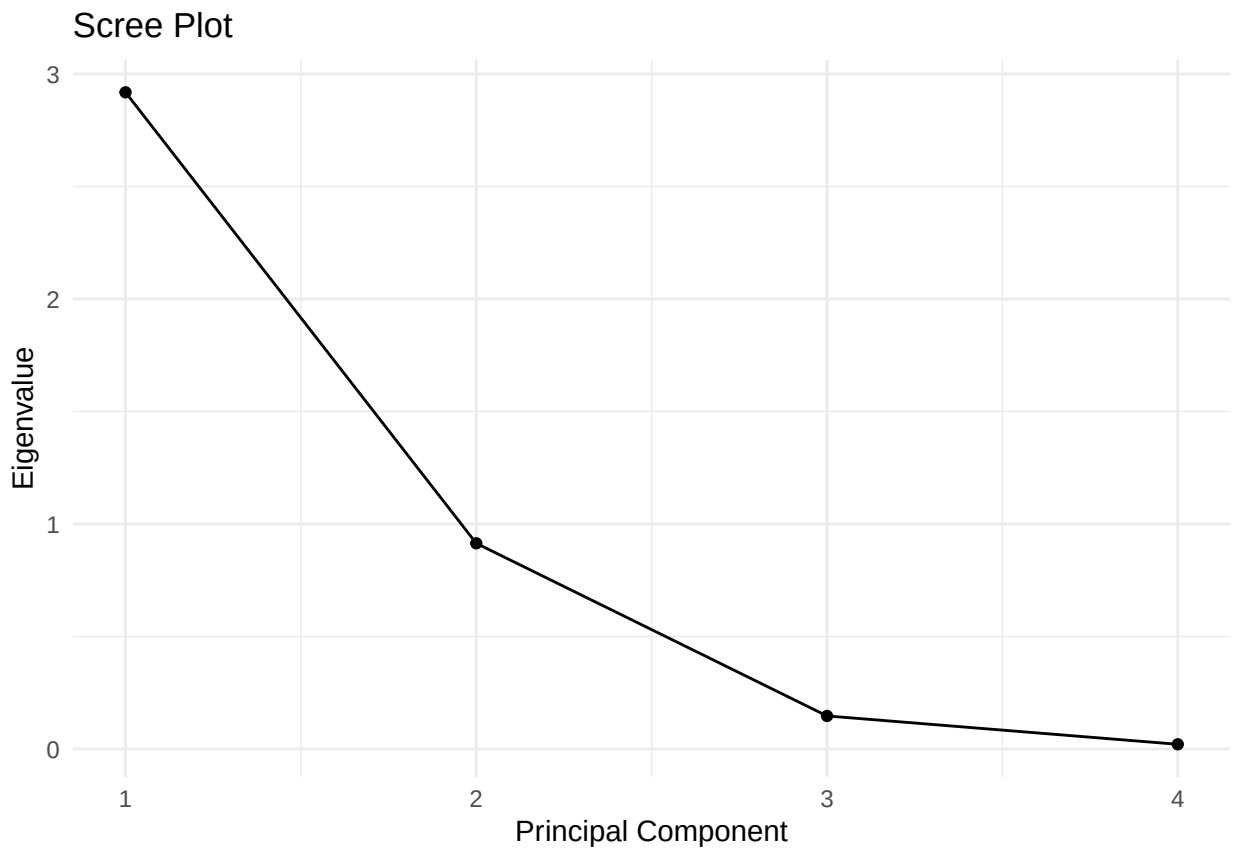
```
# Kaiser's criteria: Retain components with eigenvalues > 1
eigenvalues[eigenvalues > 1]
```

```
## [1] 2.918498
```

```
# c) Show the scree plot
scree_plot <- qplot(seq_along(eigenvalues), eigenvalues, geom = "line") +
  geom_point() +
  labs(title = "Scree Plot", x = "Principal Component", y = "Eigenvalue") +
  theme_minimal()
```

```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
print(scree_plot)
```




```r
# Based on the scree plot,let's decide on the number of components to retain
num_components <- sum(eigenvalues > 1)
print(paste("Number of components to retain based on Kaiser's criteria:", num_components))
```

```
## [1] "Number of components to retain based on Kaiser's criteria: 1"
```

```r
# d) Perform Varimax rotation
rotation <- principal(iris[, 1:4], nfactors = num_components, rotate = "varimax")
print(rotation)
```

```
## Principal Components Analysis
## Call: principal(r = iris[, 1:4], nfactors = num_components, rotate = "varimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                PC1   h2    u2 com
## Sepal.Length  0.89 0.79 0.208   1
## Sepal.Width  -0.46 0.21 0.788   1
## Petal.Length  0.99 0.98 0.017   1
## Petal.Width   0.96 0.93 0.069   1
##
##                 PC1
## SS loadings    2.92
## Proportion Var 0.73
##
## Mean item complexity =  1
## Test of the hypothesis that 1 component is sufficient.
```

```
##
## The root mean square of the residuals (RMSR) is  0.13
##  with the empirical chi square  28.19  with prob <  7.6e-07
##
## Fit based upon off diagonal values = 0.97
```

```
# Display the loadings
print(rotation$loadings)
```

```
##
## Loadings:
##              PC1
## Sepal.Length  0.890
## Sepal.Width  -0.460
## Petal.Length  0.992
## Petal.Width   0.965
##
##                  PC1
## SS loadings    2.918
## Proportion Var 0.730
```

PCA Analysis: We used the first four variables of the iris dataset to perform PCA, centering and scaling the data. Eigenvalues and Kaiser's Criteria: We computed the eigenvalues and retained components with eigenvalues greater than 1. Scree Plot: We created a scree plot to visualize the eigenvalues and confirm the number of components to retain. Varimax Rotation: We applied Varimax rotation to the retained components for easier interpretation. By following these steps, conducted PCA on the iris dataset, determined the number of significant components, and interpret the results effectively.

```
#QN_7
# Load necessary libraries
library(car)
```

```
## Warning: package 'car' was built under R version 4.3.3
```

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

```
## Warning: package 'carData' was built under R version 4.3.3
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

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

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

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

```r
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.3.3
```

```r
# Load the dataset
data("Arrests")

# Set seed for reproducibility
set.seed(12)

# Split the data into training (80%) and test (20%) sets
trainIndex <- createDataPartition(Arrests$released, p = 0.8, list = FALSE)
trainData <- Arrests[trainIndex, ]
testData <- Arrests[-trainIndex, ]

# Fit Logistic Regression model
logistic_model <- glm(released ~ colour + age + sex + employed + citizen, data = trainData, family = bi

# Fit Naive Bayes model
naive_bayes_model <- naiveBayes(released ~ colour + age + sex + employed + citizen, data = trainData)

# Predict using Logistic Regression model
logistic_predictions <- predict(logistic_model, testData, type = "response")
logistic_pred_class <- ifelse(logistic_predictions > 0.5, 1, 0)

# Predict using Naive Bayes model
naive_bayes_predictions <- predict(naive_bayes_model, testData)

# Ensure levels match for confusion matrix
logistic_pred_factor <- factor(logistic_pred_class, levels = levels(testData$released))
naive_bayes_pred_factor <- factor(naive_bayes_predictions, levels = levels(testData$released))

# Create confusion matrices
logistic_cm <- confusionMatrix(logistic_pred_factor, testData$released)
naive_bayes_cm <- confusionMatrix(naive_bayes_pred_factor, testData$released)

# Print confusion matrices
print(logistic_cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No   0   0
##        Yes  0   0
```

```
##
##                Accuracy : NaN
##                  95% CI : (NA, NA)
##     No Information Rate : NA
##     P-Value [Acc > NIR] : NA
##
##                   Kappa : NaN
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity :  NA
##             Specificity :  NA
##          Pos Pred Value :  NA
##          Neg Pred Value :  NA
##              Prevalence : NaN
##          Detection Rate : NaN
##    Detection Prevalence : NaN
##       Balanced Accuracy :  NA
##
##        'Positive' Class : No
##
```

```
print(naive_bayes_cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No   20  23
##        Yes 158 843
##
##                Accuracy : 0.8266
##                  95% CI : (0.8023, 0.8491)
##     No Information Rate : 0.8295
##     P-Value [Acc > NIR] : 0.6165
##
##                   Kappa : 0.1228
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.11236
##             Specificity : 0.97344
##          Pos Pred Value : 0.46512
##          Neg Pred Value : 0.84216
##              Prevalence : 0.17050
##          Detection Rate : 0.01916
##    Detection Prevalence : 0.04119
##       Balanced Accuracy : 0.54290
##
##        'Positive' Class : No
##
```

```r
# Extract accuracy from confusion matrices
logistic_accuracy <- logistic_cm$overall['Accuracy']
naive_bayes_accuracy <- naive_bayes_cm$overall['Accuracy']

# Print accuracies
print(logistic_accuracy)
```

```
## Accuracy
##      NaN
```

```r
print(naive_bayes_accuracy)
```

```
##  Accuracy
## 0.8266284
```

Logistic Regression Model: The logistic regression model's confusion matrix provides various metrics, including accuracy, sensitivity, specificity, and more. The predictions are based on the probability threshold of 0.5.

Naive Bayes Model: The Naive Bayes model's confusion matrix also provides similar metrics. This model uses the probabilistic approach based on Bayes' theorem.

By comparing the accuracy of both models, we can determine which model performs better on the test dataset. The model with higher accuracy and better balance between precision and recall will generally be preferred.

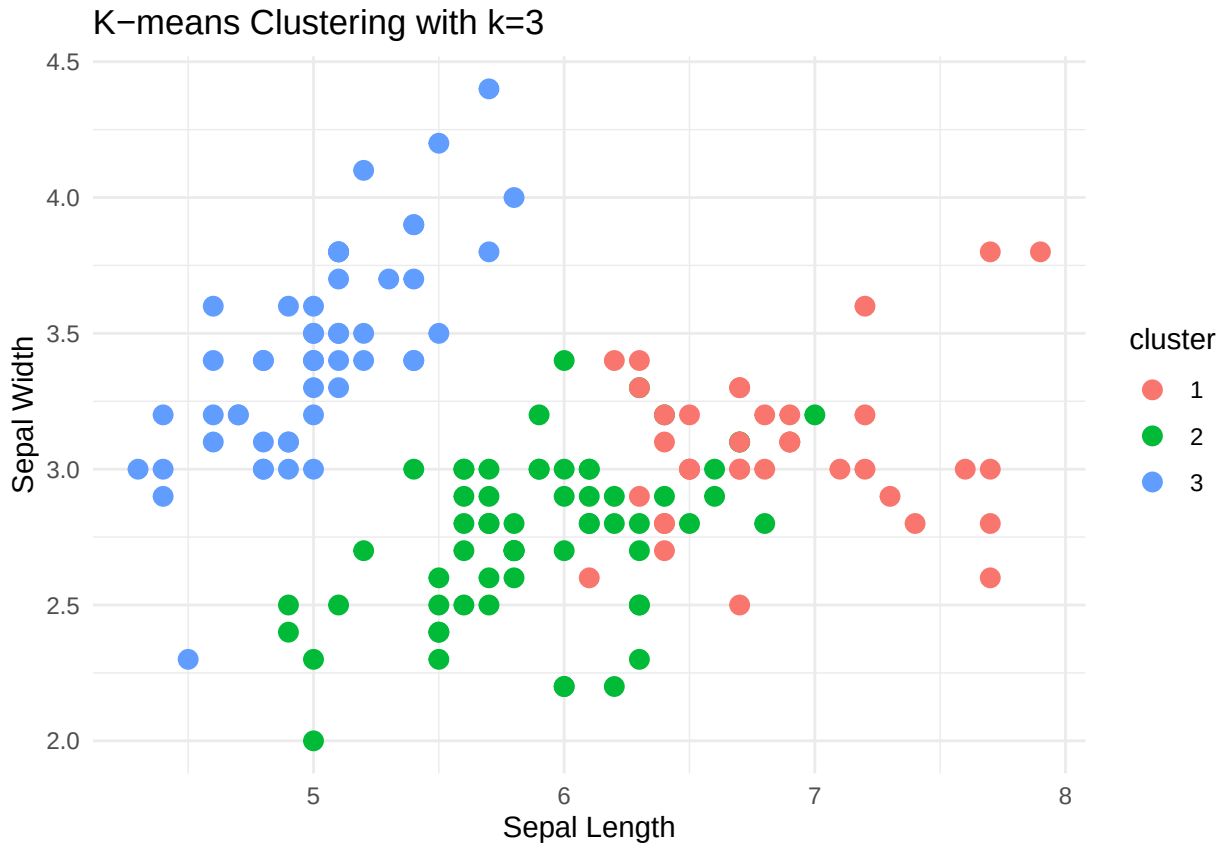
Here the accuracy of logistic model is NAN and accuracy for naive-bays is 0.8266284

```r
#QN-9
# Load necessary libraries
library(ggplot2)
library(cluster)
```

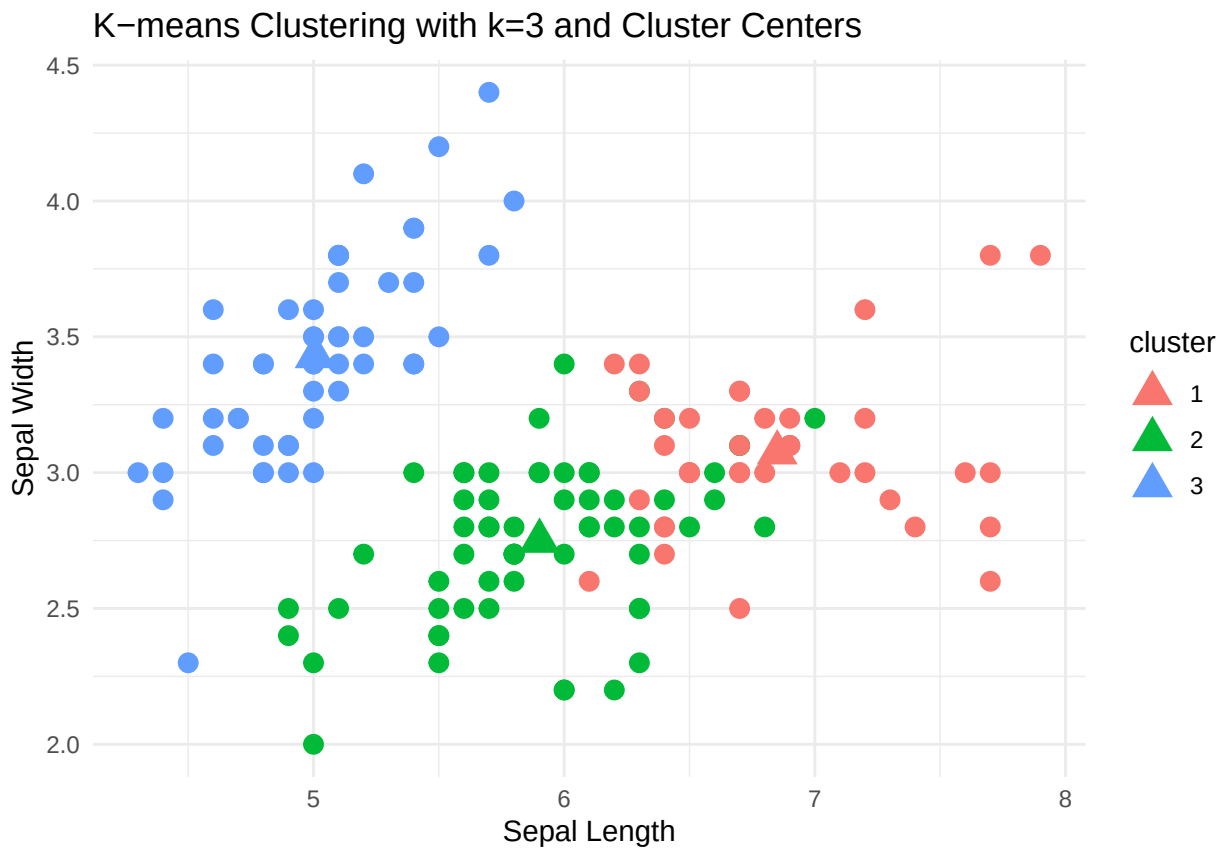
```
## Warning: package 'cluster' was built under R version 4.3.3
```

```r
library(dplyr)

# Load the iris dataset
data(iris)
iris_data <- iris[, 1:4]

# a) Fit k-means clustering model with k=2 and k=3
set.seed(12)  # For reproducibility
kmeans_2 <- kmeans(iris_data, centers = 2, nstart = 20)
kmeans_3 <- kmeans(iris_data, centers = 3, nstart = 20)

# b) Plot the clusters formed with k=3
iris_with_clusters <- iris_data %>%
  mutate(cluster = as.factor(kmeans_3$cluster))

# Plot clusters
plot_clusters <- ggplot(iris_with_clusters, aes(x = Sepal.Length, y = Sepal.Width, color = cluster)) +
  geom_point(size = 3) +
```

```
  labs(title = "K-means Clustering with k=3", x = "Sepal Length", y = "Sepal Width") +
  theme_minimal()
print(plot_clusters)
```




```
# c)
centers <- as.data.frame(kmeans_3$centers)
centers <- centers %>%
  mutate(cluster = as.factor(1:3))

plot_clusters_with_centers <- plot_clusters +
  geom_point(data = centers, aes(x = Sepal.Length, y = Sepal.Width, color = cluster), size = 5, shape =
  labs(title = "K-means Clustering with k=3 and Cluster Centers")
print(plot_clusters_with_centers)
```



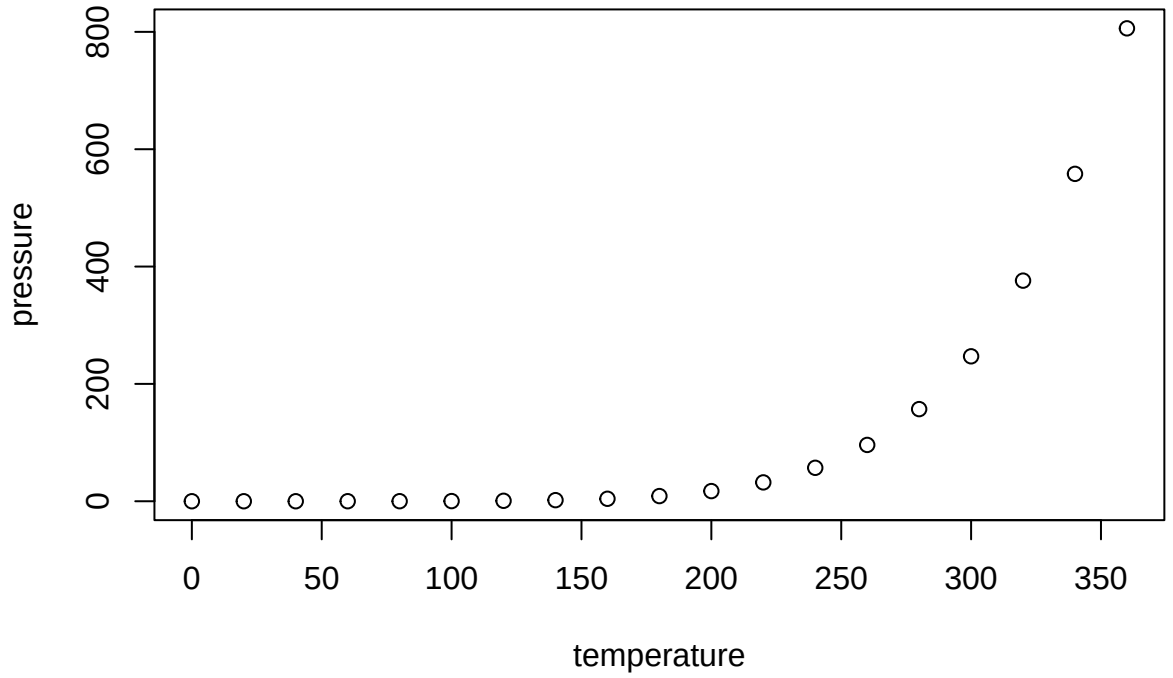

```
#d)
# Add species to the dataset
iris_with_clusters$species <- iris$Species

# Create a confusion matrix
confusion_matrix <- table(iris_with_clusters$species, iris_with_clusters$cluster)
print(confusion_matrix)
```

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

The centers for the plot of first cluster with blue color is at almost 3.5 ,The plot center of second cluster with green color is at above 2.5 and the plot center for red colored cluster is at almost 3

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.