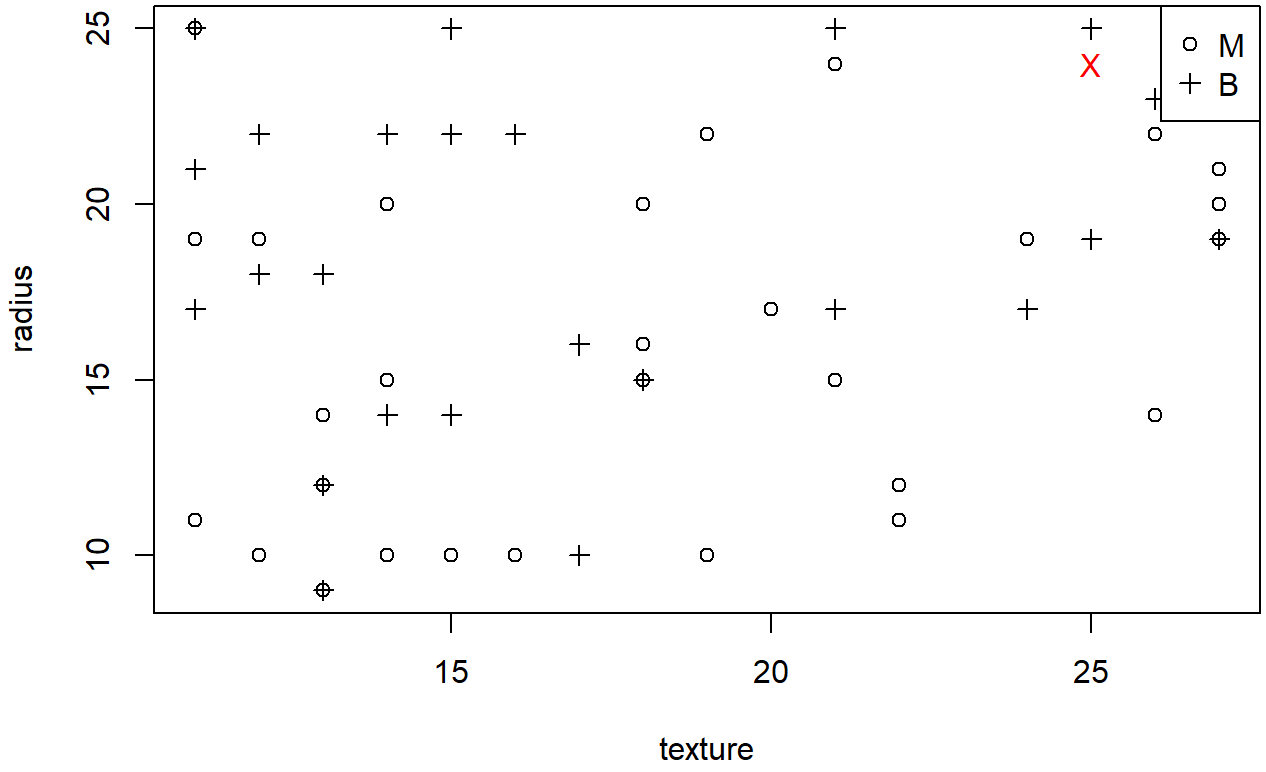
```
######################## K -  Nearest(KNN) Neighbours#############################################################
library(caret)
```

```
## Loading required package: ggplot2
```

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

```
library(FNN)

# Load the data

cancer.df <- read.csv("Cancer.csv")

# Splitting the data into training and validation sets
set.seed(1881)
train.index <- sample(nrow(cancer.df), 0.6 * nrow(cancer.df))
train.df <- cancer.df[train.index, ]
valid.df <- cancer.df[-train.index, ]

# Defining the new patient data
new.df <- data.frame(radius = 25, texture = 24, perimeter = 140, area = 1100, smoothness = 0.050,
                     compactness = 0.100, symmetry = 0.250, fractal_dimension = 0.065)
```

```
# Scatter plot for radius Vs texture
plot(radius ~ texture, data = train.df, pch = ifelse(train.df$diagnosis_result == "M", 1, 3))
# The new patient data has (radius = 25, texture = 24), so let's plot it using red color in "X"
text(25,24, "X", col = "red")
# Displaying the Legend in topright which indicates the data points with diagnosis_result=="M" as 'o' and diagnosis_result=
="B" as '+'
legend("topright", c("M", "B"), pch = c(1, 3), cex = 1)
```




```
# Normalizing the data
norm_values <- preProcess(train.df[, 1:8], method = c("center", "scale"))
#preProcess calculates the mean and standard deviation
train.norm.df <- predict(norm_values, train.df[, 1:8])
valid.norm.df <- predict(norm_values, valid.df[, 1:8])
cancer.norm.df <- predict(norm_values, cancer.df[, 1:8])
new.norm_df <- predict(norm_values, new.df)
```

```
#K-Nearest Neighbors

#Calculating the nearest 1 to 14 data points accuracy to find optimal 'k' value inorder to find whether the new patient has malignant tumor or not.

k_values <- 1:14
knn_results <- data.frame(k = k_values, accuracy = rep(0, length(k_values)))

for (k in k_values) {
  knn_pred <- knn(train.norm.df, valid.norm.df, cl = as.factor(train.df$diagnosis_result), k = k)
  accuracy <- confusionMatrix(knn_pred, as.factor(valid.df$diagnosis_result))$overall["Accuracy"]
  knn_results[k, "accuracy"] <- accuracy
}

# Printing the accuracy results for all 14 nearest data points
print(knn_results)
```

```
##     k accuracy
## 1   1    0.675
## 2   2    0.550
## 3   3    0.725
## 4   4    0.675
## 5   5    0.750
## 6   6    0.750
## 7   7    0.725
## 8   8    0.750
## 9   9    0.725
## 10 10    0.750
## 11 11    0.725
## 12 12    0.725
## 13 13    0.700
## 14 14    0.675
```

```
#From the above results we can see that K=5,6,10 has highest accuracy as 0.750
#So let's consider the optimal value of K which is 5.

# k-Nearest Neighbors for the new data point
knn_pred_new <- knn(cancer.norm.df, new.norm_df, cl = as.factor(cancer.df$diagnosis_result), k = 5)

# Printing the neighbors for the new data point
neighbors <- row.names(cancer.df)[attr(knn_pred_new, "nn.index")]
print(neighbors)
```

```
## [1] "96" "11" "87" "71" "7"
```

```
# Now let's determine which type of tumor does the new patient has by plotting the contingency table between neighbors

neighbor_labels <- cancer.df[, "diagnosis_result"]
contingency_table <- table(neighbors, cancer.df[neighbors, "diagnosis_result"])
print(contingency_table)
```

```
##
## neighbors M
##        11 1
##        7  1
##        71 1
##        87 1
##        96 1
```

```
#From the contingency table we can see that all the nearest neighbors of new patient data point belong to Malignant tumor.

#The new patient has a malignant tumor.
```

```
######################## Naive Bayes#################################################################

library(e1071)

#Load the data
data <- read.csv("cancernb.csv")

# Preprocessing the data as the whole data is in character converting it into factor
data$radius <- as.factor(data$radius)
data$texture <- as.factor(data$texture)
data$perimeter <- as.factor(data$perimeter)
data$area <- as.factor(data$area)
data$smoothness <- as.factor(data$smoothness)
data$compactness <- as.factor(data$compactness)
data$symmetry <-  as.factor(data$symmetry)
data$fractal_dimension <- as.factor(data$fractal_dimension)
data$diagnosis_result <- as.factor(data$diagnosis_result)
```

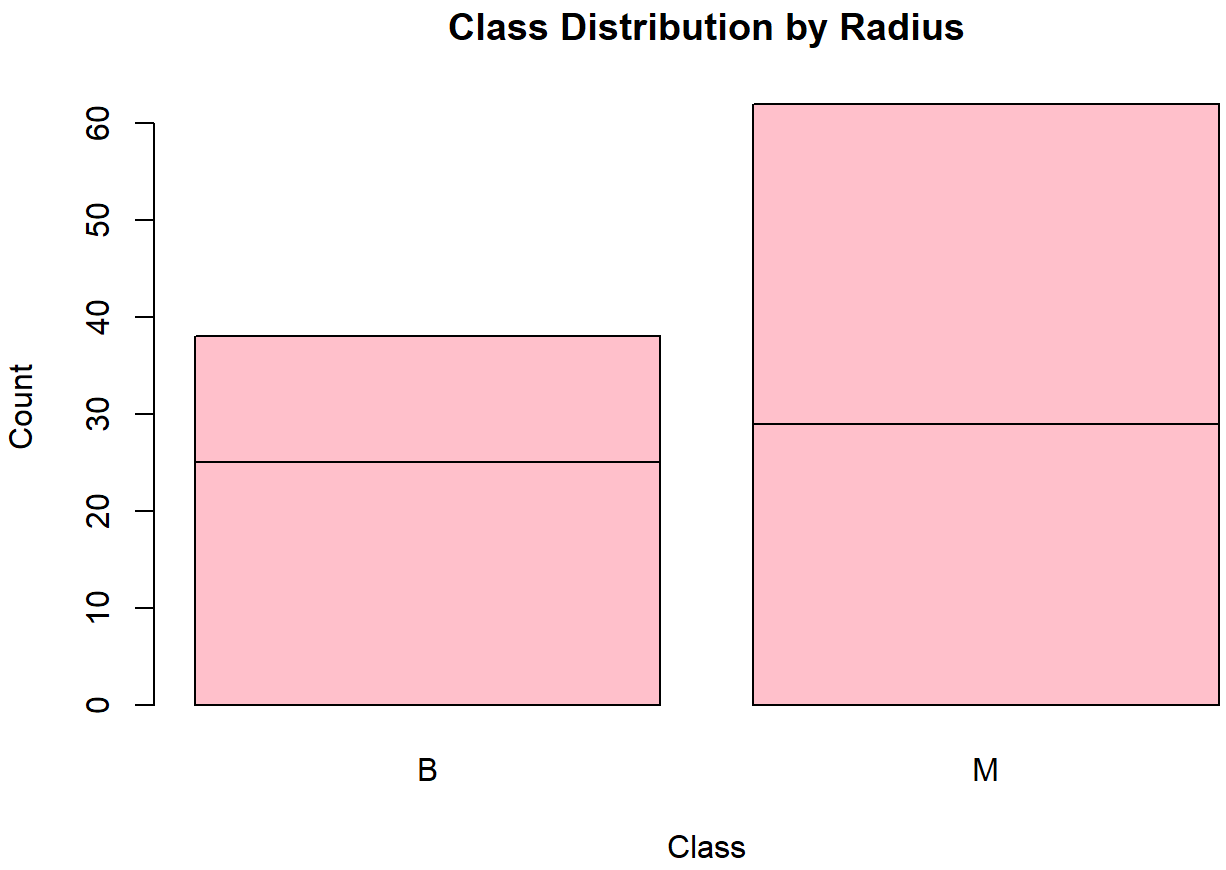
```
# Splitting the data into training and validation sets
set.seed(123)  # Set a seed for reproducibility
train.rows <- sample(rownames(data), dim(data)[1]*0.6)
train.data <- data[train.rows, ]
valid.rows <- setdiff(rownames(data), train.rows)
valid.data <- data[valid.rows, ]
```

```
# Train a Naive Bayes classifier
model <- naiveBayes(radius ~ diagnosis_result + texture, data = train.data)

# Make predictions on the test data
predictions <- predict(model, valid.data, type = "class")

# Check and align levels if necessary
if (!identical(levels(predictions), levels(valid.data$diagnosis_result))) {
  levels(predictions) <- levels(valid.data$diagnosis_result)
}

# Evaluate the model's performance
confusion_matrix <- confusionMatrix(predictions, valid.data$diagnosis_result)
print(confusion_matrix)
```

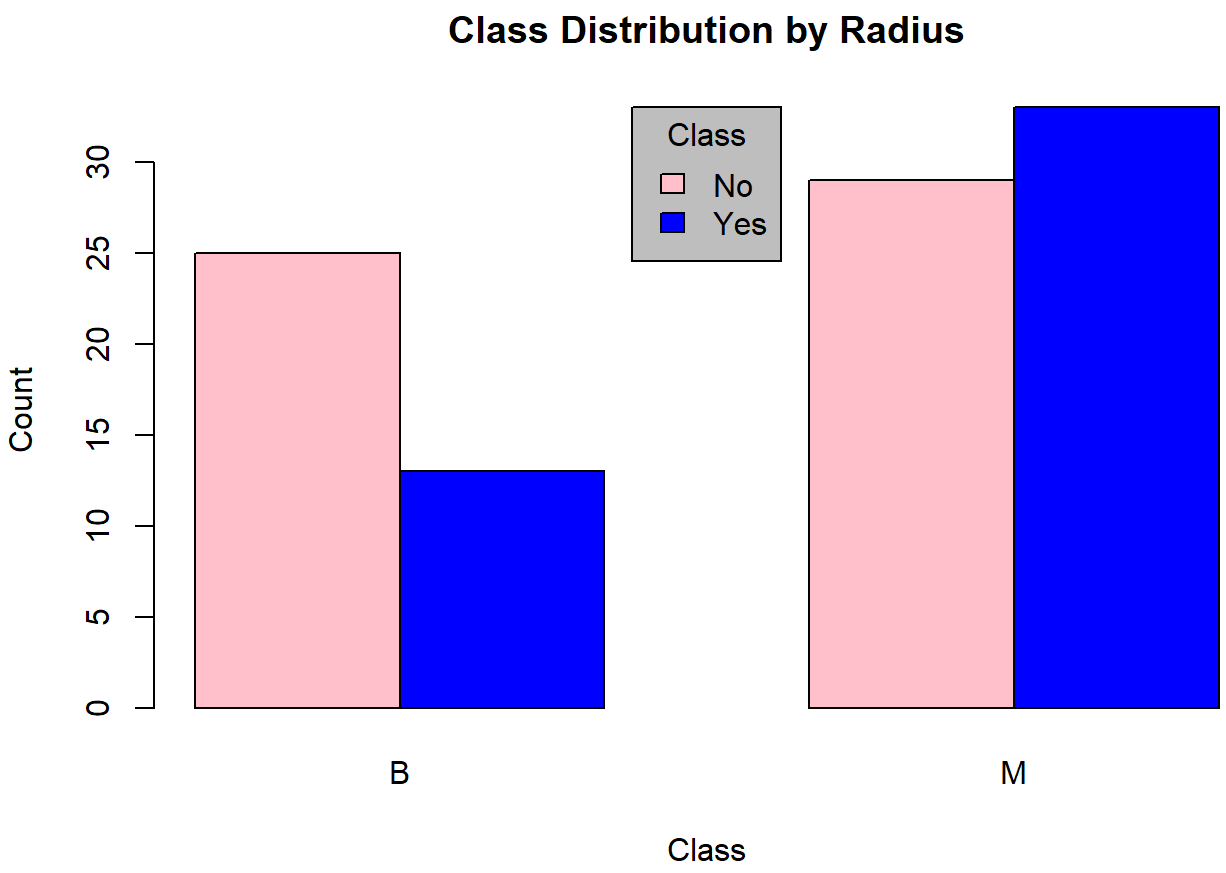
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 13  0
##          M  0 27
##
##                Accuracy : 1
##                  95% CI : (0.9119, 1)
##     No Information Rate : 0.675
##     P-Value [Acc > NIR] : 1.486e-07
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.000
##             Specificity : 1.000
##          Pos Pred Value : 1.000
##          Neg Pred Value : 1.000
##              Prevalence : 0.325
##          Detection Rate : 0.325
##    Detection Prevalence : 0.325
##       Balanced Accuracy : 1.000
##
##        'Positive' Class : B
##
```

```
#The confusion matrix and statistics show that:
#1. The value 13 suggests that there were 13 instances of class 'B' (benign) correctly classified as 'B'.
#2. The value 27suggests that there were 27 instances of class 'M' (malignant) correctly classified as 'M'.
#3. There are no mis-classifications (off-diagonal elements are all 0).

#The overall accuracy of the model is 1. This means that 100% of the instances in the validation set were correctly classifi
ed.The confidence interval for accuracy ranges from 0.9119 to 1.

#The p-value is very close to zero (1.486e-07), indicating that the observed accuracy is significantly better than the No In
formation Rate.

#The positive class is 'B' (benign) in this case
```

```
# Bar chart to visualize the class distribution by variables “radius” versus “diagnosis_result”.
class_distribution <- table(data$radius, data$diagnosis_result)
barplot(class_distribution, main = "Class Distribution by Radius", xlab = "Class", ylab = "Count", col = "pink")
```


Class Distribution by Radius
Count
0
10
20
30
40
50
60
B
M
Class

```
# Create a stacked bar chart to visualize class distribution by "Radius"
class_distribution_by_radius <- table(data$radius, data$diagnosis_result)


# Set the legend background color to be gray
legend_bg_color <- "gray"

# Set the legend border width and line type to 1 (with border)
legend_box_width <- 1
legend_box_line_type <- 1

# Create the plot with the adjusted legend properties
barplot(class_distribution_by_radius, beside = TRUE, col = c("pink", "blue"),
        main = "Class Distribution by Radius", xlab = "Class", ylab = "Count",
        legend.text = rownames(class_distribution_by_radius),
        args.legend = list(title = "Class", x = "top", bg = legend_bg_color,
                           box.lwd = legend_box_width, box.lty = legend_box_line_type))
```




```
#From the charts we can find that the count of patients with Malignant tumor is high compared to patients with Benign tumor irrespective of radius.
#The patients with Radius "Yes" and Malignant tumor are higher compared to all other patients with Radius "No" or tumor as "Benign"
#Next comes Patients with Radius "No" and Benign tumor.
```