final\_project

Diallo Mamadou Korka

2023-12-23

# Loading the libraries required for this project

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.2.3

library(tidyverse)

## Warning: package 'tibble' was built under R version 4.2.3

## Warning: package 'readr' was built under R version 4.2.3

## Warning: package 'dplyr' was built under R version 4.2.3

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.3 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ lubridate 1.9.2 ✔ tibble 3.2.1  
## ✔ purrr 1.0.1 ✔ tidyr 1.3.0  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force all conflicts to become errors

library(readr)  
library(corrplot)

## Warning: package 'corrplot' was built under R version 4.2.3

## corrplot 0.92 loaded

library(cowplot)

##   
## Attaching package: 'cowplot'  
##   
## The following object is masked from 'package:lubridate':  
##   
## stamp

library(viridis)

## Warning: package 'viridis' was built under R version 4.2.3

## Loading required package: viridisLite

## Warning: package 'viridisLite' was built under R version 4.2.3

library(tibble)  
library(caret)

## Warning: package 'caret' was built under R version 4.2.3

## Loading required package: lattice  
##   
## Attaching package: 'caret'  
##   
## The following object is masked from 'package:purrr':  
##   
## lift

library(rpart)  
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.2.3

library(randomForest)

## Warning: package 'randomForest' was built under R version 4.2.3

## randomForest 4.7-1.1  
## Type rfNews() to see new features/changes/bug fixes.  
##   
## Attaching package: 'randomForest'  
##   
## The following object is masked from 'package:dplyr':  
##   
## combine  
##   
## The following object is masked from 'package:ggplot2':  
##   
## margin

# Load the dataset

insurance\_data <- read\_csv("insurance.csv")

## Rows: 1338 Columns: 7  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (3): sex, smoker, region  
## dbl (4): age, bmi, children, charges  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

head(insurance\_data)

## # A tibble: 6 × 7  
## age sex bmi children smoker region charges  
## <dbl> <chr> <dbl> <dbl> <chr> <chr> <dbl>  
## 1 19 female 27.9 0 yes southwest 16885.  
## 2 18 male 33.8 1 no southeast 1726.  
## 3 28 male 33 3 no southeast 4449.  
## 4 33 male 22.7 0 no northwest 21984.  
## 5 32 male 28.9 0 no northwest 3867.  
## 6 31 female 25.7 0 no southeast 3757.

## Data Exploration

data\_shape <- dim(insurance\_data)  
  
# Print the shape of the dataset  
cat("Number of rows:", data\_shape[1], "\n")

## Number of rows: 1338

cat("Number of columns:", data\_shape[2], "\n")

## Number of columns: 7

#Data Cleaning ## cheack missing values

colSums(is.na(insurance\_data))

## age sex bmi children smoker region charges   
## 0 0 0 0 0 0 0

## check duplicate rows

duplicate\_rows <- sum(duplicated(insurance\_data))  
print(duplicate\_rows)

## [1] 1

## drop Duplicated rows

data <- distinct(insurance\_data)

#Exploring the dataset

# Age  
age\_mean <- mean(insurance\_data$age)  
age\_sd <- sd(insurance\_data$age)  
age\_min <- min(insurance\_data$age)  
age\_max <- max(insurance\_data$age)  
  
# Sex  
sex\_counts <- table(insurance\_data$sex)  
  
# BMI  
bmi\_mean <- mean(insurance\_data$bmi)  
bmi\_sd <- sd(insurance\_data$bmi)  
bmi\_min <- min(insurance\_data$bmi)  
bmi\_max <- max(insurance\_data$bmi)  
  
# Children  
children\_mean <- mean(insurance\_data$children)  
children\_sd <- sd(insurance\_data$children)  
children\_min <- min(insurance\_data$children)  
children\_max <- max(insurance\_data$children)  
  
# Smoker  
smoker\_counts <- table(insurance\_data$smoker)  
  
# Region  
region\_counts <- table(insurance\_data$region)  
  
# Charges  
charges\_mean <- mean(insurance\_data$charges)  
charges\_sd <- sd(insurance\_data$charges)  
charges\_min <- min(insurance\_data$charges)  
charges\_max <- max(insurance\_data$charges)  
  
# Print the results  
cat("Age:\n")

## Age:

cat("Mean:", age\_mean, "\n")

## Mean: 39.20703

cat("Standard Deviation:", age\_sd, "\n")

## Standard Deviation: 14.04996

cat("Min:", age\_min, ", Max:", age\_max, "\n\n")

## Min: 18 , Max: 64

cat("Sex:\n")

## Sex:

cat("Male:", sex\_counts["male"], "instances\n")

## Male: 676 instances

cat("Female:", sex\_counts["female"], "instances\n\n")

## Female: 662 instances

cat("BMI (Body Mass Index):\n")

## BMI (Body Mass Index):

cat("Mean:", bmi\_mean, "\n")

## Mean: 30.6634

cat("Standard Deviation:", bmi\_sd, "\n")

## Standard Deviation: 6.098187

cat("Min:", bmi\_min, ", Max:", bmi\_max, "\n\n")

## Min: 15.96 , Max: 53.13

cat("Children:\n")

## Children:

cat("Mean:", children\_mean, "\n")

## Mean: 1.094918

cat("Standard Deviation:", children\_sd, "\n")

## Standard Deviation: 1.205493

cat("Min:", children\_min, ", Max:", children\_max, "\n\n")

## Min: 0 , Max: 5

cat("Smoker:\n")

## Smoker:

cat("Yes:", smoker\_counts["yes"], "instances\n")

## Yes: 274 instances

cat("No:", smoker\_counts["no"], "instances\n\n")

## No: 1064 instances

cat("Region:\n")

## Region:

cat("Southwest:", region\_counts["southwest"], "instances\n")

## Southwest: 325 instances

cat("Southeast:", region\_counts["southeast"], "instances\n")

## Southeast: 364 instances

cat("Northwest:", region\_counts["northwest"], "instances\n")

## Northwest: 325 instances

cat("Northeast:", region\_counts["northeast"], "instances\n\n")

## Northeast: 324 instances

cat("Charges:\n")

## Charges:

cat("Mean:", charges\_mean, "\n")

## Mean: 13270.42

cat("Standard Deviation:", charges\_sd, "\n")

## Standard Deviation: 12110.01

cat("Min:", charges\_min, ", Max:", charges\_max, "\n")

## Min: 1121.874 , Max: 63770.43

#Exploring data Analysis

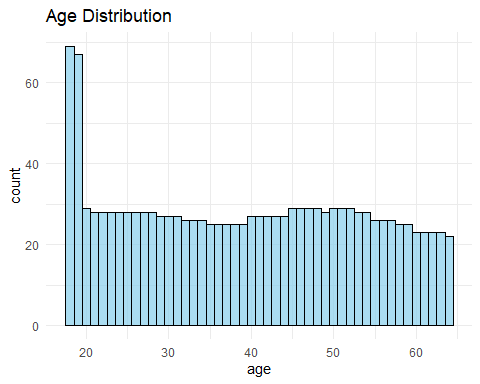
## information about the dataset

str(data)

## tibble [1,337 × 7] (S3: tbl\_df/tbl/data.frame)  
## $ age : num [1:1337] 19 18 28 33 32 31 46 37 37 60 ...  
## $ sex : chr [1:1337] "female" "male" "male" "male" ...  
## $ bmi : num [1:1337] 27.9 33.8 33 22.7 28.9 ...  
## $ children: num [1:1337] 0 1 3 0 0 0 1 3 2 0 ...  
## $ smoker : chr [1:1337] "yes" "no" "no" "no" ...  
## $ region : chr [1:1337] "southwest" "southeast" "southeast" "northwest" ...  
## $ charges : num [1:1337] 16885 1726 4449 21984 3867 ...

#### distribution on age

# Create a histogram  
ggplot(data, aes(x = age)) +  
 geom\_histogram(binwidth = 1, fill = "skyblue", color = "black", alpha = 0.7) +  
 labs(title = "Age Distribution") +  
 theme\_minimal()



# Define a function to categorize age  
age\_category <- function(age) {  
 age\_dict <- c('0-9', '10-19', '20-29', '30-39', '40-49', '50-59', '60-69', '70-79', '80-89', '90-99', '100-200')  
 return(age\_dict[findInterval(age, seq(0, 110, by = 10))])  
}  
  
# Apply the age\_category function to create a new column 'age\_category'  
insurance\_data$age\_category <- sapply(insurance\_data$age, age\_category)  
  
# Print counts by age category  
print("Age by category")

## [1] "Age by category"

by\_age <- table(insurance\_data$age\_category)  
print(by\_age)

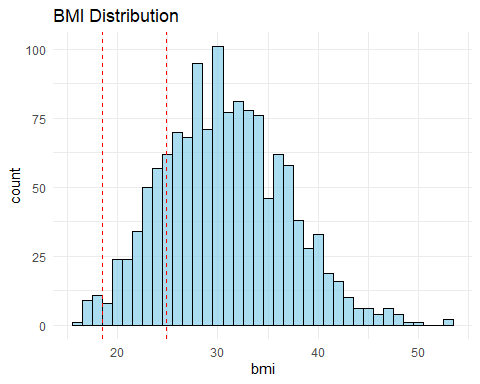
##   
## 10-19 20-29 30-39 40-49 50-59 60-69   
## 137 280 257 279 271 114

cat(rep('-', 80), '\n')

## - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

#BMI Distribution

ggplot(data, aes(x = bmi)) +  
 geom\_histogram(binwidth = 1, fill = "skyblue", color = "black", alpha = 0.7) +  
 geom\_vline(xintercept = 18.5, color = "red", linetype = "dashed") +  
 geom\_vline(xintercept = 24.9, color = "red", linetype = "dashed") +  
 labs(title = "BMI Distribution") +  
 theme\_minimal()



# Define a function to categorize BMI  
bmi\_category <- function(bmi) {  
 if (bmi < 18.5) {  
 return('under-weight')  
 } else if (bmi >= 18.5 && bmi <= 24.9) {  
 return('normal-weight')  
 } else if (bmi >= 25 && bmi <= 29.9) {  
 return('over-weight')  
 } else if (bmi >= 30.0) {  
 return('obese')  
 } else {  
 return('unknown') # Handle cases where BMI is not in any category  
 }  
}  
  
# Apply the bmi\_category function to create a new column 'cbmi'  
insurance\_data$cbmi <- sapply(insurance\_data$bmi, bmi\_category)  
  
# Print counts by BMI category  
print("BMI")

## [1] "BMI"

by\_bmi <- table(insurance\_data$cbmi)  
print(by\_bmi)

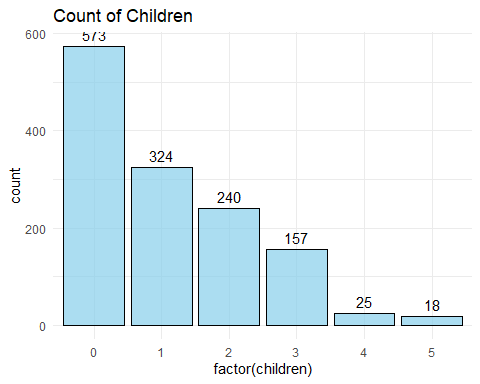
##   
## normal-weight obese over-weight under-weight unknown   
## 222 707 377 20 12

cat(rep('-', 80), '\n')

## - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# Create a count plot with labels  
ggplot(data, aes(x = factor(children))) +  
 geom\_bar(fill = "skyblue", color = "black", alpha = 0.7) +  
 geom\_text(stat = "count", aes(label = ..count..), vjust = -0.5) +  
 labs(title = "Count of Children") +  
 theme\_minimal()

## Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2 3.4.0.  
## ℹ Please use `after\_stat(count)` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



# Print counts by number of children  
print("Children")

## [1] "Children"

by\_children <- table(insurance\_data$children)  
print(by\_children)

##   
## 0 1 2 3 4 5   
## 574 324 240 157 25 18

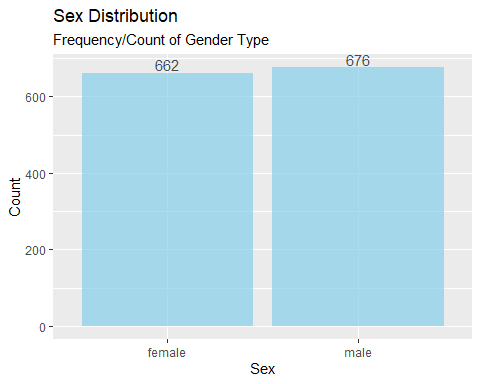
cat(rep('-', 80), '\n')

## - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

## Analysis on categorical features.

categorical feature consist of sex, smoker, and region. ### analysis on sex

# Create a count plot for sex distribution  
g <- ggplot(insurance\_data, aes(x = sex)) +  
 geom\_bar(stat = "count", fill = "skyblue", alpha = 0.7) +  
 geom\_text(stat = "count", aes(label = ..count.., vjust = -0.2), alpha = 0.7) +  
 ggtitle("Sex Distribution", subtitle = "Frequency/Count of Gender Type") +  
 labs(x = "Sex", y = "Count")  
  
# Display the plot  
print(g)



# Print counts by sex  
print("Gender")

## [1] "Gender"

by\_sex <- table(insurance\_data$sex)  
print(by\_sex)

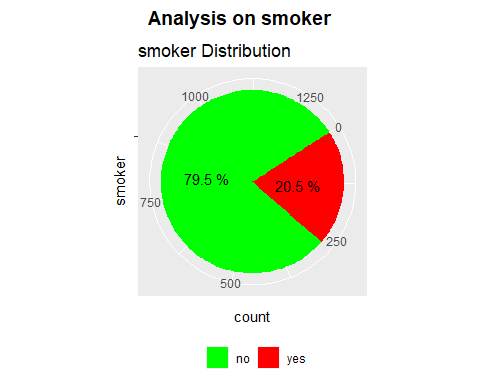
##   
## female male   
## 662 676

cat(rep('-', 80), '\n')

## - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

### analysis on smoker

# count plot  
# get freq/count  
smoker\_table <- table(data$smoker)  
fr <- as.data.frame.table(smoker\_table)  
# plot freq  
g <- ggplot(fr, aes(x="", y=Freq, fill = factor(Var1))) +   
 geom\_bar(width=1, stat="identity") +   
 geom\_text(aes(label = paste(round(Freq / sum(Freq) \* 100, 1), "%")),  
 position = position\_stack(vjust = 0.5)) +  
 ggtitle("smoker Distribution") +   
 labs(x="smoker", y="count") +   
 coord\_polar(theta = "y", start=1) +   
 theme(legend.position="bottom") +   
 guides(fill=guide\_legend(title="")) +   
 scale\_fill\_manual(values = c("green", "red"))  
   
p <- plot\_grid(g)  
title <- ggdraw() + draw\_label("Analysis on smoker", fontface = 'bold')  
plot\_grid(title, p, ncol = 1, rel\_heights = c(0.1,1))



# Print counts by smoker status  
print("Smoker")

## [1] "Smoker"

by\_smoker <- table(insurance\_data$smoker)  
print(by\_smoker)

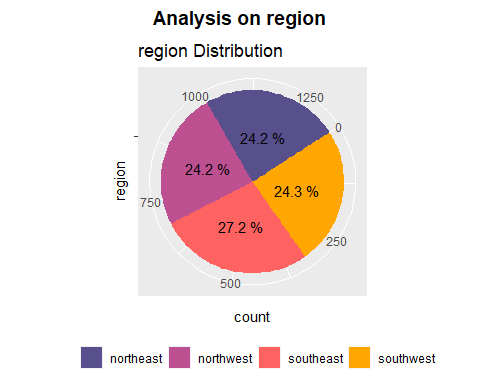
##   
## no yes   
## 1064 274

cat(rep('-', 80), '\n')

## - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

### analysis on region

# count plot  
# get freq/count  
region\_table <- table(data$region)  
fr <- as.data.frame.table(region\_table)  
# plot freq  
g <- ggplot(fr, aes(x="", y=Freq, fill = factor(Var1))) +   
 geom\_bar(width=1, stat="identity") +   
 geom\_text(aes(label = paste(round(Freq / sum(Freq) \* 100, 1), "%")),  
 position = position\_stack(vjust = 0.5)) +  
 ggtitle("region Distribution" ) +   
 labs(x="region", y="count") +  
 coord\_polar(theta = "y", start=1) +   
 theme(legend.position="bottom") +   
 guides(fill=guide\_legend(title="")) +   
 scale\_fill\_manual(values = c("#58508d", "#bc5090", "#ff6361", "#ffa600"))  
  
p <- plot\_grid(g)  
title <- ggdraw() + draw\_label("Analysis on region", fontface = 'bold')  
plot\_grid(title, p, ncol = 1, rel\_heights = c(0.1,1))



# Print counts by region  
print("Region")

## [1] "Region"

by\_region <- table(insurance\_data$region)  
print(by\_region)

##   
## northeast northwest southeast southwest   
## 324 325 364 325

cat(rep('-', 80), '\n')

## - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

#Processing the dataset ## Encoding.

encode <- function(x, order = unique(x)){  
 x <- as.numeric(factor(x, levels = order, exclude = NULL))  
 x  
}  
encoded\_df <- insurance\_data  
encoded\_df[["sex"]] <- encode(insurance\_data[["sex"]])  
encoded\_df[["smoker"]] <- encode(insurance\_data[["smoker"]])  
encoded\_df[["region"]] <- encode(insurance\_data[["region"]])  
head(encoded\_df)

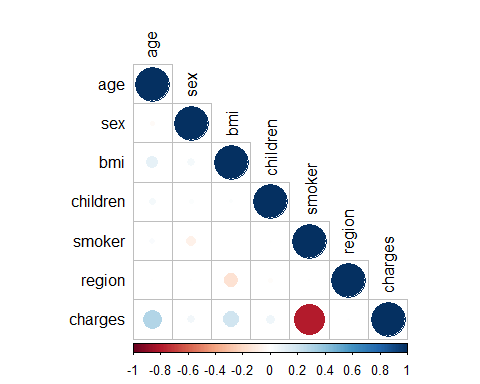
## # A tibble: 6 × 9  
## age sex bmi children smoker region charges age\_category cbmi   
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <chr>   
## 1 19 1 27.9 0 1 1 16885. 10-19 over-weight   
## 2 18 2 33.8 1 2 2 1726. 10-19 obese   
## 3 28 2 33 3 2 2 4449. 20-29 obese   
## 4 33 2 22.7 0 2 3 21984. 30-39 normal-weight  
## 5 32 2 28.9 0 2 3 3867. 30-39 over-weight   
## 6 31 1 25.7 0 2 2 3757. 30-39 over-weight

str(encoded\_df)

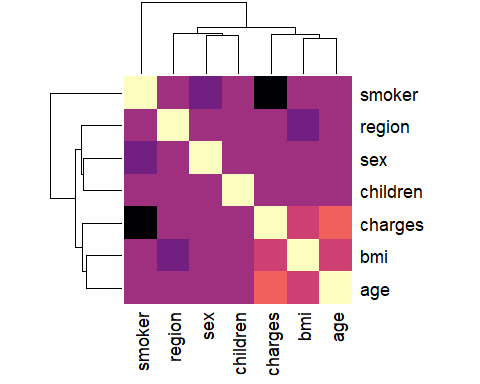
## spc\_tbl\_ [1,338 × 9] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ age : num [1:1338] 19 18 28 33 32 31 46 37 37 60 ...  
## $ sex : num [1:1338] 1 2 2 2 2 1 1 1 2 1 ...  
## $ bmi : num [1:1338] 27.9 33.8 33 22.7 28.9 ...  
## $ children : num [1:1338] 0 1 3 0 0 0 1 3 2 0 ...  
## $ smoker : num [1:1338] 1 2 2 2 2 2 2 2 2 2 ...  
## $ region : num [1:1338] 1 2 2 3 3 2 2 3 4 3 ...  
## $ charges : num [1:1338] 16885 1726 4449 21984 3867 ...  
## $ age\_category: chr [1:1338] "10-19" "10-19" "20-29" "30-39" ...  
## $ cbmi : chr [1:1338] "over-weight" "obese" "obese" "normal-weight" ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. age = col\_double(),  
## .. sex = col\_character(),  
## .. bmi = col\_double(),  
## .. children = col\_double(),  
## .. smoker = col\_character(),  
## .. region = col\_character(),  
## .. charges = col\_double()  
## .. )  
## - attr(\*, "problems")=<externalptr>

## correlation matrix

#corr <- cor(encoded\_df)  
#g <- corrplot(corr, type="lower", tl.col = "black")   
  
  
# Select only numeric columns from the dataframe  
numeric\_columns <- sapply(encoded\_df, is.numeric)  
numeric\_data <- encoded\_df[, numeric\_columns]  
  
# Calculate correlation matrix  
corr <- cor(numeric\_data)  
  
# Plot correlation matrix  
library(corrplot)  
g <- corrplot(corr, type = "lower", tl.col = "black")



gg <- heatmap(x=corr, col = magma(10), symm=T)

 # MODEL BUILDING

Spilt dataset to train and test

# Next, set a seed for reproducibility  
set.seed(123)  
  
# Split the data into a training set and a test set   
split\_index <- createDataPartition(encoded\_df$charges, p = 0.7, list = FALSE)  
train\_data <- encoded\_df[split\_index, ]  
test\_data <- encoded\_df[-split\_index, ]  
  
# Check the dimensions of the train set and test set  
dim(train\_data)

## [1] 938 9

dim(test\_data)

## [1] 400 9

### Decision Tree model

# build the decision tree   
decision\_tr <- rpart(smoker ~ age + sex + bmi + region, data = train\_data, method = "class")  
  
# make predictions on the test data  
predictions <- predict(decision\_tr, test\_data, type = "class")  
print("-------------------------------------------------------------------------------------------")

## [1] "-------------------------------------------------------------------------------------------"

# convert predictions and test data to factor and make the levels as same  
predictions <- as.factor(predictions)  
test\_data$smoker <- as.factor(test\_data$smoker)  
levels(predictions) <- levels(test\_data$smoker)  
  
print("-------------------------------------------------------------------------------------------")

## [1] "-------------------------------------------------------------------------------------------"

# evaluation of the the model   
# Confusion matrix to evalute Performance  
confusion\_matrix <- table(predicted = predictions, actual = test\_data$smoker)  
confusion\_matrix

## actual  
## predicted 1 2  
## 1 0 0  
## 2 77 323

print("-------------------------------------------------------------------------------------------")

## [1] "-------------------------------------------------------------------------------------------"

summary(decision\_tr)

## Call:  
## rpart(formula = smoker ~ age + sex + bmi + region, data = train\_data,   
## method = "class")  
## n= 938   
##   
## CP nsplit rel error xerror xstd  
## 1 0 0 1 0 0  
##   
## Node number 1: 938 observations  
## predicted class=2 expected loss=0.2100213 P(node) =1  
## class counts: 197 741  
## probabilities: 0.210 0.790

#Accuracy of the Decision tree model

accuracy\_of\_decision\_tree <- sum(diag(confusion\_matrix)) / sum(confusion\_matrix)  
print(paste("The accuracy of the Decision Tree model is : ", accuracy\_of\_decision\_tree))

## [1] "The accuracy of the Decision Tree model is : 0.8075"

###build the Random forest model

random\_forest\_model <- randomForest(as.factor(smoker) ~ ., data = train\_data, ntree = 500)  
  
# make predictions on the test data  
random\_forest\_predictions <- predict(random\_forest\_model, newdata = test\_data, type = "class")  
  
print("-------------------------------------------------------------------------------------------")

## [1] "-------------------------------------------------------------------------------------------"

# Evalute Performance  
confusion\_matrix <- table(predicted = random\_forest\_predictions, actual = test\_data$smoker)  
confusion\_matrix

## actual  
## predicted 1 2  
## 1 71 10  
## 2 6 313

print("-------------------------------------------------------------------------------------------")

## [1] "-------------------------------------------------------------------------------------------"

# Summary of the model  
summary(random\_forest\_model)

## Length Class Mode   
## call 4 -none- call   
## type 1 -none- character  
## predicted 938 factor numeric   
## err.rate 1500 -none- numeric   
## confusion 6 -none- numeric   
## votes 1876 matrix numeric   
## oob.times 938 -none- numeric   
## classes 2 -none- character  
## importance 8 -none- numeric   
## importanceSD 0 -none- NULL   
## localImportance 0 -none- NULL   
## proximity 0 -none- NULL   
## ntree 1 -none- numeric   
## mtry 1 -none- numeric   
## forest 14 -none- list   
## y 938 factor numeric   
## test 0 -none- NULL   
## inbag 0 -none- NULL   
## terms 3 terms call

# Calculate the accuracy of the model

accuracy\_Random <- sum(diag(confusion\_matrix)) / sum(confusion\_matrix)  
print(paste("Accuracy of the Random Forest Model is:", accuracy\_Random))

## [1] "Accuracy of the Random Forest Model is: 0.96"

### Compare the accuracy of three model

cat("Accuracy of Decision Tree Model:", (accuracy\_of\_decision\_tree))

## Accuracy of Decision Tree Model: 0.8075

cat("Accuracy of Random Forest Model:", (accuracy\_Random))

## Accuracy of Random Forest Model: 0.96