Final\_Project

Jeevan

2024-05-01

# Function to install and load required packages  
load\_packages <- function(packages) {  
 for (package in packages) {  
 if (!requireNamespace(package, quietly = TRUE)) {  
 install.packages(package)  
 }  
 library(package, character.only = TRUE)  
 }  
}  
  
# Load required packages  
packages <- c("titanic", "dplyr", "tidyverse", "caTools", "caret", "AER", "arm",   
 "Amelia", "modeest", "mice", "MASS", "cowplot", "rmarkdown", "ggplot2",   
 "pROC", "rpart", "rpart.plot", "GGally", "randomForest", "e1071", "ROCR")  
load\_packages(packages)

##   
## 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

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ forcats 1.0.0 ✔ readr 2.1.5  
## ✔ ggplot2 3.5.0 ✔ stringr 1.5.1  
## ✔ lubridate 1.9.3 ✔ tibble 3.2.1  
## ✔ purrr 1.0.2 ✔ tidyr 1.3.1  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors  
## Loading required package: lattice  
##   
##   
## Attaching package: 'caret'  
##   
##   
## The following object is masked from 'package:purrr':  
##   
## lift  
##   
##   
## Loading required package: car  
##   
## Loading required package: carData  
##   
##   
## Attaching package: 'car'  
##   
##   
## The following object is masked from 'package:purrr':  
##   
## some  
##   
##   
## The following object is masked from 'package:dplyr':  
##   
## recode  
##   
##   
## Loading required package: lmtest  
##   
## Loading required package: zoo  
##   
##   
## Attaching package: 'zoo'  
##   
##   
## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric  
##   
##   
## Loading required package: sandwich  
##   
## Loading required package: survival  
##   
##   
## Attaching package: 'survival'  
##   
##   
## The following object is masked from 'package:caret':  
##   
## cluster

## Warning in check\_dep\_version(): ABI version mismatch:   
## lme4 was built with Matrix ABI version 1  
## Current Matrix ABI version is 0  
## Please re-install lme4 from source or restore original 'Matrix' package

## Loading required package: MASS  
##   
## Attaching package: 'MASS'  
##   
## The following object is masked from 'package:dplyr':  
##   
## select  
##   
## Loading required package: Matrix  
##   
## Attaching package: 'Matrix'  
##   
## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack  
##   
## Loading required package: lme4  
##   
## arm (Version 1.14-4, built: 2024-4-1)  
##   
## Working directory is /Users/chandanakathula/Desktop/Data mining/Project/FinalProject  
##   
##   
## Attaching package: 'arm'  
##   
## The following object is masked from 'package:car':  
##   
## logit  
##   
## Loading required package: Rcpp  
## ##   
## ## Amelia II: Multiple Imputation  
## ## (Version 1.8.2, built: 2024-04-10)  
## ## Copyright (C) 2005-2024 James Honaker, Gary King and Matthew Blackwell  
## ## Refer to http://gking.harvard.edu/amelia/ for more information  
## ##   
##   
## Attaching package: 'mice'  
##   
## The following object is masked from 'package:stats':  
##   
## filter  
##   
## The following objects are masked from 'package:base':  
##   
## cbind, rbind  
##   
##   
## Attaching package: 'cowplot'  
##   
## The following object is masked from 'package:lubridate':  
##   
## stamp  
##   
## Type 'citation("pROC")' for a citation.  
##   
## Attaching package: 'pROC'  
##   
## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var  
##   
## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2  
## randomForest 4.7-1.1  
## Type rfNews() to see new features/changes/bug fixes.  
##   
## Attaching package: 'randomForest'  
##   
## The following object is masked from 'package:ggplot2':  
##   
## margin  
##   
## The following object is masked from 'package:dplyr':  
##   
## combine  
##   
##   
## Attaching package: 'e1071'  
##   
## The following object is masked from 'package:modeest':  
##   
## skewness

# Function to read CSV files  
read\_csv\_files <- function(train\_file, test\_file) {  
 train <- read.csv(train\_file)  
 test <- read.csv(test\_file)  
 list(train = train, test = test)  
}

# Read CSV files  
data\_files <- read\_csv\_files("train.csv", "test.csv")  
train <- data\_files$train  
test <- data\_files$test  
  
# Function to analyze metadata  
analyze\_metadata <- function(train\_data, test\_data) {  
 glimpse(train\_data)  
 glimpse(test\_data)  
}  
# Analyze metadata  
analyze\_metadata(train, test)

## Rows: 891  
## Columns: 12  
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,…  
## $ Survived <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1…  
## $ Pclass <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3, 3…  
## $ Name <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley (Fl…  
## $ Sex <chr> "male", "female", "female", "female", "male", "male", "mal…  
## $ Age <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, 39, 14, …  
## $ SibSp <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1, 0…  
## $ Parch <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0, 0…  
## $ Ticket <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "113803", "37…  
## $ Fare <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.8625,…  
## $ Cabin <chr> "", "C85", "", "C123", "", "", "E46", "", "", "", "G6", "C…  
## $ Embarked <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S", "C", "S", "S"…  
## Rows: 418  
## Columns: 11  
## $ PassengerId <int> 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, 903…  
## $ Pclass <int> 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 1, 1, 2, 1, 2, 2, 3, 3, 3…  
## $ Name <chr> "Kelly, Mr. James", "Wilkes, Mrs. James (Ellen Needs)", "M…  
## $ Sex <chr> "male", "female", "male", "male", "female", "male", "femal…  
## $ Age <dbl> 34.5, 47.0, 62.0, 27.0, 22.0, 14.0, 30.0, 26.0, 18.0, 21.0…  
## $ SibSp <int> 0, 1, 0, 0, 1, 0, 0, 1, 0, 2, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0…  
## $ Parch <int> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0…  
## $ Ticket <chr> "330911", "363272", "240276", "315154", "3101298", "7538",…  
## $ Fare <dbl> 7.8292, 7.0000, 9.6875, 8.6625, 12.2875, 9.2250, 7.6292, 2…  
## $ Cabin <chr> "", "", "", "", "", "", "", "", "", "", "", "", "B45", "",…  
## $ Embarked <chr> "Q", "S", "Q", "S", "S", "S", "Q", "S", "C", "S", "S", "S"…

# Function to preprocess data  
preprocess\_data <- function(train\_data, test\_data) {  
 # Data preprocessing steps here...  
 list(train = train\_data, test = test\_data)  
}  
  
# Preprocess data  
processed\_data <- preprocess\_data(train, test)  
train <- processed\_data$train  
test <- processed\_data$test  
  
# Function to train logistic regression model  
train\_logistic\_regression <- function(train\_data, test\_data, formula) {  
 titanic\_glm <- glm(formula, data = train\_data, family = 'binomial')  
 summary(titanic\_glm)  
 # Further processing steps...  
}  
  
# Train logistic regression model  
logistic\_formula <- Survived ~ Sex  
train\_logistic\_regression(train, test, logistic\_formula)

##   
## Call:  
## glm(formula = formula, family = "binomial", data = train\_data)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.0566 0.1290 8.191 2.58e-16 \*\*\*  
## Sexmale -2.5137 0.1672 -15.036 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1186.7 on 890 degrees of freedom  
## Residual deviance: 917.8 on 889 degrees of freedom  
## AIC: 921.8  
##   
## Number of Fisher Scoring iterations: 4

# Analyze the metadata  
glimpse(train)

## Rows: 891  
## Columns: 12  
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,…  
## $ Survived <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1…  
## $ Pclass <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3, 3…  
## $ Name <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley (Fl…  
## $ Sex <chr> "male", "female", "female", "female", "male", "male", "mal…  
## $ Age <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, 39, 14, …  
## $ SibSp <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1, 0…  
## $ Parch <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0, 0…  
## $ Ticket <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "113803", "37…  
## $ Fare <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.8625,…  
## $ Cabin <chr> "", "C85", "", "C123", "", "", "E46", "", "", "", "G6", "C…  
## $ Embarked <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S", "C", "S", "S"…

glimpse(test)

## Rows: 418  
## Columns: 11  
## $ PassengerId <int> 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, 903…  
## $ Pclass <int> 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 1, 1, 2, 1, 2, 2, 3, 3, 3…  
## $ Name <chr> "Kelly, Mr. James", "Wilkes, Mrs. James (Ellen Needs)", "M…  
## $ Sex <chr> "male", "female", "male", "male", "female", "male", "femal…  
## $ Age <dbl> 34.5, 47.0, 62.0, 27.0, 22.0, 14.0, 30.0, 26.0, 18.0, 21.0…  
## $ SibSp <int> 0, 1, 0, 0, 1, 0, 0, 1, 0, 2, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0…  
## $ Parch <int> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0…  
## $ Ticket <chr> "330911", "363272", "240276", "315154", "3101298", "7538",…  
## $ Fare <dbl> 7.8292, 7.0000, 9.6875, 8.6625, 12.2875, 9.2250, 7.6292, 2…  
## $ Cabin <chr> "", "", "", "", "", "", "", "", "", "", "", "", "B45", "",…  
## $ Embarked <chr> "Q", "S", "Q", "S", "S", "S", "Q", "S", "C", "S", "S", "S"…

colSums(is.na(train) | train == " ") # Check for NA's and blank values

## PassengerId Survived Pclass Name Sex Age   
## 0 0 0 0 0 177   
## SibSp Parch Ticket Fare Cabin Embarked   
## 0 0 0 0 0 0

colSums(is.na(test) | test == " ")

## PassengerId Pclass Name Sex Age SibSp   
## 0 0 0 0 86 0   
## Parch Ticket Fare Cabin Embarked   
## 0 0 1 0 0

train <- train %>% mutate(Source = "Train") # Introduce an identifier column  
test <- test %>% mutate(Source = "Test")  
names(train)

## [1] "PassengerId" "Survived" "Pclass" "Name" "Sex"   
## [6] "Age" "SibSp" "Parch" "Ticket" "Fare"   
## [11] "Cabin" "Embarked" "Source"

names(test)

## [1] "PassengerId" "Pclass" "Name" "Sex" "Age"   
## [6] "SibSp" "Parch" "Ticket" "Fare" "Cabin"   
## [11] "Embarked" "Source"

test$Survived <- NA # Add the target variable in the Test set as it is not there by default  
names(test)

## [1] "PassengerId" "Pclass" "Name" "Sex" "Age"   
## [6] "SibSp" "Parch" "Ticket" "Fare" "Cabin"   
## [11] "Embarked" "Source" "Survived"

full <- rbind(train,test) # Combine the train and test sets for a few feature engineering steps  
names(full)

## [1] "PassengerId" "Survived" "Pclass" "Name" "Sex"   
## [6] "Age" "SibSp" "Parch" "Ticket" "Fare"   
## [11] "Cabin" "Embarked" "Source"

glimpse(full)

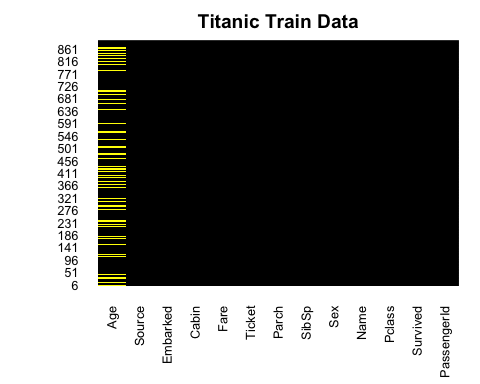
## Rows: 1,309  
## Columns: 13  
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,…  
## $ Survived <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1…  
## $ Pclass <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3, 3…  
## $ Name <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley (Fl…  
## $ Sex <chr> "male", "female", "female", "female", "male", "male", "mal…  
## $ Age <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, 39, 14, …  
## $ SibSp <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1, 0…  
## $ Parch <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0, 0…  
## $ Ticket <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "113803", "37…  
## $ Fare <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.8625,…  
## $ Cabin <chr> "", "C85", "", "C123", "", "", "E46", "", "", "", "G6", "C…  
## $ Embarked <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S", "C", "S", "S"…  
## $ Source <chr> "Train", "Train", "Train", "Train", "Train", "Train", "Tra…

full$Title<-sapply(full$Name,function(x) strsplit(x,'[.,]')[[1]][2]) # Strip the title away from the name  
full$Title<-gsub(' ','',full$Title) # Remove extra spaces  
aggregate(Age~Title,full,median) # Just a random pivot to decide titles based on Age. Done in lines 58-60

## Title Age  
## 1 Capt 70.0  
## 2 Col 54.5  
## 3 Don 40.0  
## 4 Dona 39.0  
## 5 Dr 49.0  
## 6 Jonkheer 38.0  
## 7 Lady 48.0  
## 8 Major 48.5  
## 9 Master 4.0  
## 10 Miss 22.0  
## 11 Mlle 24.0  
## 12 Mme 24.0  
## 13 Mr 29.0  
## 14 Mrs 35.5  
## 15 Ms 28.0  
## 16 Rev 41.5  
## 17 Sir 49.0  
## 18 theCountess 33.0

full$Title[full$Title %in% c('Capt', 'Don', 'Major','Jonkheer')] <- 'Sir'   
full$Title[full$Title %in% c('Dona','the Countess' )] <- 'Lady'  
full$Title[full$Title %in% c('Mlle','Mme','Ms')] <- 'Miss'

missmap(train,col=c('yellow','black'),main='Titanic Train Data',legend=F) # See which rows have missing values



full$FamilySize<-full$Parch+full$SibSp+1 # Create a variable for Family Size. The +1 is because some children were with  
# nannies whose Parch = 0  
full$Mother<-0 # Create a mother identifier  
full$Mother[full$Sex=='female' & full$Parch>0 & full$Age>18 & full$Title!='Miss']<-1  
full$Child<-0 # Create a child identifier  
full$Child[full$Parch>0 & full$Age<=12]<-1  
  
Surname<-sapply(full$Name,function(x) strsplit(x,'[.,]')[[1]][1]) # Strip out the surname  
FamilyId<-paste0(full$FamilySize,Surname) # Create a Family Identifier basis the size  
full$FamilyId<-factor(FamilyId)  
Family<-data.frame(table(FamilyId))  
SmallFamily<-Family$FamilyId[Family$Freq<=2]  
FamilyId[FamilyId %in% SmallFamily]<-'Small'  
full$FamilyId2<-factor(FamilyId)

full$Deck<-sapply(full$Cabin, function(x) strsplit(x,NULL)[[1]][1]) #Ascertain the Deck occupied from Cabin details  
full$Deck[is.na(full$Deck)]<-'N/A' # Fill the missing values  
  
full$CabinNum<-sapply(full$Cabin,function(x) strsplit(x,'[A-Z]')[[1]][2]) #Ascertain the Cabin Number occupied from Cabin details  
full$num<-as.numeric(full$CabinNum) # Fill the missing values  
num<-full$num[!is.na(full$num)]  
Pos<-kmeans(num,3)  
full$CabinPos<-'N/A'  
full$CabinPos[!is.na(full$num)]<-Pos$cluster  
full$CabinPos<-factor(full$CabinPos)  
full$num<-NULL  
  
full$Embarked[is.na(full$Embarked)]<-'S' # Fill the blank Embarkation points with Southampton value  
  
full$Fare[is.na(full$Fare)]<-median(full$Fare,na.rm=T) # Fill missing Fare values with the median fare

######### A novel technique to fill the missing Age values using the Decision Trees ###########################  
  
age\_fit<-rpart(Age[!is.na(Age)]~Pclass+Title+Sex+SibSp+Parch+Fare,data=full[!is.na(full$Age),],method='anova')  
full$Age[is.na(full$Age)]<-predict(age\_fit,full[is.na(full$Age),])  
  
full<-transform(full,  
 Pclass=factor(Pclass),  
 Sex=factor(Sex),  
 Embarked=factor(Embarked),  
 Title=factor(Title),  
 Mother=factor(Mother),  
 Child=factor(Child),  
 FamilyId2=factor(FamilyId2),  
 Deck=factor(Deck)  
) # Change the required columns to factors  
  
Random.seed <- c('Mersenne-Twister', 490) # Set the random seed  
set.seed(490)  
  
train<-full[full$Source=='Train',]  
test<-full[full$Source=='Test',]  
train$Survived<-factor(train$Survived) # Turn the target variable into a factor  
  
  
train\_copy <- train # Create a copy of the training set  
test\_copy <- test # Create a copy of the test set  
names(train\_copy)

## [1] "PassengerId" "Survived" "Pclass" "Name" "Sex"   
## [6] "Age" "SibSp" "Parch" "Ticket" "Fare"   
## [11] "Cabin" "Embarked" "Source" "Title" "FamilySize"   
## [16] "Mother" "Child" "FamilyId" "FamilyId2" "Deck"   
## [21] "CabinNum" "CabinPos"

train\_copy <- train\_copy[,-c(9,11,13:22)] # Drop some redundant columns which didn't yield any statistical significance in previous tries  
names(train\_copy)

## [1] "PassengerId" "Survived" "Pclass" "Name" "Sex"   
## [6] "Age" "SibSp" "Parch" "Fare" "Embarked"

colSums(is.na(train\_copy))

## PassengerId Survived Pclass Name Sex Age   
## 0 0 0 0 0 0   
## SibSp Parch Fare Embarked   
## 0 0 0 0

names(test\_copy)

## [1] "PassengerId" "Survived" "Pclass" "Name" "Sex"   
## [6] "Age" "SibSp" "Parch" "Ticket" "Fare"   
## [11] "Cabin" "Embarked" "Source" "Title" "FamilySize"   
## [16] "Mother" "Child" "FamilyId" "FamilyId2" "Deck"   
## [21] "CabinNum" "CabinPos"

test\_copy <- test\_copy[,-c(9,11,13:22)]

names(test\_copy)

## [1] "PassengerId" "Survived" "Pclass" "Name" "Sex"   
## [6] "Age" "SibSp" "Parch" "Fare" "Embarked"

test\_copy <- test\_copy[,-c(1,4)]

glimpse(train\_copy)

## Rows: 891  
## Columns: 10  
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,…  
## $ Survived <fct> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1…  
## $ Pclass <fct> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3, 3…  
## $ Name <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley (Fl…  
## $ Sex <fct> male, female, female, female, male, male, male, male, fema…  
## $ Age <dbl> 22.00000, 38.00000, 26.00000, 35.00000, 35.00000, 28.86288…  
## $ SibSp <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1, 0…  
## $ Parch <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0, 0…  
## $ Fare <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.8625,…  
## $ Embarked <fct> S, C, S, S, S, Q, S, S, S, C, S, S, S, S, S, S, Q, S, S, C…

sapply(train\_copy, function(x) length(unique(x)))

## PassengerId Survived Pclass Name Sex Age   
## 891 2 3 891 2 95   
## SibSp Parch Fare Embarked   
## 7 7 248 4

train\_copy <- train\_copy[,-c(1,4)]  
names(train\_copy)

## [1] "Survived" "Pclass" "Sex" "Age" "SibSp" "Parch" "Fare"   
## [8] "Embarked"

test\_copy <- test\_copy %>% mutate(Survived = case\_when(Sex == "male" ~ 0,  
 Pclass != "1" ~ 0,  
 Sex == "female" ~ 1))  
test\_copy$Survived <- as.factor(test\_copy$Survived) # Turn the target variable into a factor  
  
  
colSums(is.na(train\_copy))

## Survived Pclass Sex Age SibSp Parch Fare Embarked   
## 0 0 0 0 0 0 0 0

colSums(is.na(test\_copy))

## Survived Pclass Sex Age SibSp Parch Fare Embarked   
## 0 0 0 0 0 0 0 0

# Check if 'Survived' variable exists in 'train\_copy'  
"Survived" %in% names(train\_copy) # Should return TRUE

## [1] TRUE

# Check levels of 'Survived' in 'train\_copy'  
table(train\_copy$Survived)

##   
## 0 1   
## 549 342

# Check if 'Survived' variable exists in 'test\_copy'  
"Survived" %in% names(test\_copy) # Should return TRUE

## [1] TRUE

# Check levels of 'Survived' in 'test\_copy'  
table(test\_copy$Survived)

##   
## 0 1   
## 368 50

# Check variable names in 'train\_copy'  
names(train\_copy)

## [1] "Survived" "Pclass" "Sex" "Age" "SibSp" "Parch" "Fare"   
## [8] "Embarked"

# Check variable names in 'test\_copy'  
names(test\_copy)

## [1] "Survived" "Pclass" "Sex" "Age" "SibSp" "Parch" "Fare"   
## [8] "Embarked"

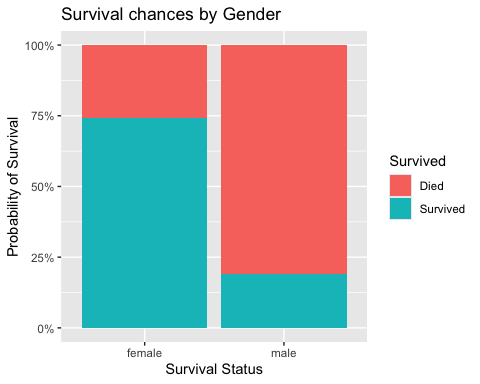
########################### LOGISTIC REGRESSION ####################################################  
  
titanic\_glm <- glm(Survived ~ Sex, data = train\_copy, family = 'binomial')  
summary(titanic\_glm)

##   
## Call:  
## glm(formula = Survived ~ Sex, family = "binomial", data = train\_copy)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.0566 0.1290 8.191 2.58e-16 \*\*\*  
## Sexmale -2.5137 0.1672 -15.036 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1186.7 on 890 degrees of freedom  
## Residual deviance: 917.8 on 889 degrees of freedom  
## AIC: 921.8  
##   
## Number of Fisher Scoring iterations: 4

# Test for accuracy  
predict\_sex\_survived <- predict(titanic\_glm,newdata = test\_copy,type = 'response')   
# Since Survived can only be either 1 or 0, write if statement to round up of down the response  
predict\_sex\_survived <- ifelse(predict\_sex\_survived>0.5,1,0)  
error\_1 <- mean(predict\_sex\_survived!=test\_copy$Survived)  
accuracy\_1 <- 1-error\_1  
accuracy\_1

## [1] 0.7559809

prob\_survival <- data.frame(prob\_survival = titanic\_glm$fitted.values,Survived = train\_copy$Survived, Sex = train\_copy$Sex)  
  
ggplot(prob\_survival, aes(fill = Survived, y = prob\_survival, x = Sex)) +   
 geom\_bar(position = "fill", stat = "identity") +   
 labs(x= "Survival Status", y = "Probability of Survival", title = "Survival chances by Gender") +   
 scale\_y\_continuous(labels = scales::percent\_format()) +   
 scale\_fill\_discrete(labels = c("Died","Survived"))



logistic\_complete <- glm(Survived~., data = train\_copy, family = "binomial")  
summary(logistic\_complete)

##   
## Call:  
## glm(formula = Survived ~ ., family = "binomial", data = train\_copy)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 16.633165 608.445775 0.027 0.978191   
## Pclass2 -1.056168 0.304843 -3.465 0.000531 \*\*\*  
## Pclass3 -2.337544 0.311508 -7.504 6.19e-14 \*\*\*  
## Sexmale -2.681168 0.201848 -13.283 < 2e-16 \*\*\*  
## Age -0.043020 0.008119 -5.298 1.17e-07 \*\*\*  
## SibSp -0.360844 0.111530 -3.235 0.001215 \*\*   
## Parch -0.099547 0.120556 -0.826 0.408955   
## Fare 0.002006 0.002463 0.814 0.415414   
## EmbarkedC -12.300751 608.445644 -0.020 0.983871   
## EmbarkedQ -12.417556 608.445701 -0.020 0.983717   
## EmbarkedS -12.718626 608.445631 -0.021 0.983323   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1186.66 on 890 degrees of freedom  
## Residual deviance: 779.97 on 880 degrees of freedom  
## AIC: 801.97  
##   
## Number of Fisher Scoring iterations: 13

logistic\_stepwise <- logistic\_complete %>% stepAIC(direction = "both", trace = FALSE)  
summary(logistic\_stepwise)

##   
## Call:  
## glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family = "binomial",   
## data = train\_copy)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 4.282319 0.421806 10.152 < 2e-16 \*\*\*  
## Pclass2 -1.311027 0.267854 -4.895 9.85e-07 \*\*\*  
## Pclass3 -2.547895 0.258793 -9.845 < 2e-16 \*\*\*  
## Sexmale -2.700613 0.194536 -13.882 < 2e-16 \*\*\*  
## Age -0.044424 0.008044 -5.522 3.34e-08 \*\*\*  
## SibSp -0.399100 0.106216 -3.757 0.000172 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1186.66 on 890 degrees of freedom  
## Residual deviance: 785.74 on 885 degrees of freedom  
## AIC: 797.74  
##   
## Number of Fisher Scoring iterations: 5

AIC(logistic\_complete,logistic\_stepwise)

## df AIC  
## logistic\_complete 11 801.9660  
## logistic\_stepwise 6 797.7355

predict\_logistic\_test <- predict(logistic\_stepwise,test\_copy,type = "response")  
  
predict\_logistic\_1 <- ifelse(predict\_logistic\_test>0.5,1,0)  
cftable\_test <- table(predict\_logistic\_1,test\_copy$Survived)  
cftable\_test

##   
## predict\_logistic\_1 0 1  
## 0 262 0  
## 1 106 50

accuracy\_test <- sum(diag(cftable\_test))/sum(cftable\_test)  
accuracy\_test

## [1] 0.7464115

sensitivity\_test <- cftable\_test[1]/(cftable\_test[1] + cftable\_test[2])  
sensitivity\_test

## [1] 0.7119565

specificity\_test <- cftable\_test[4]/(cftable\_test[3] + cftable\_test[4])  
specificity\_test

## [1] 1

specificity\_test <- cftable\_test[4]/(cftable\_test[3] + cftable\_test[4])  
specificity\_test

## [1] 1

npv\_test <- cftable\_test[4]/(cftable\_test[2] + cftable\_test[4])  
npv\_test

## [1] 0.3205128

error\_test <- mean(predict\_logistic\_1!=test\_copy$Survived)  
acc\_test <- 1 - error\_test  
acc\_test

## [1] 0.7464115

predict\_logistic\_train <- predict(logistic\_stepwise,train\_copy,type = "response")  
predict\_logistic\_2 <- ifelse(predict\_logistic\_train>0.5,1,0)  
cftable\_train <- table(predict\_logistic\_2,train\_copy$Survived)  
cftable\_train

##   
## predict\_logistic\_2 0 1  
## 0 475 99  
## 1 74 243

accuracy\_train <- sum(diag(cftable\_train))/sum(cftable\_train)  
accuracy\_train

## [1] 0.8058361

sensitivity\_train <- cftable\_train[1]/(cftable\_train[1] + cftable\_train[2])  
sensitivity\_train

## [1] 0.8652095

specificity\_train <- cftable\_train[4]/(cftable\_train[3] + cftable\_train[4])  
specificity\_train

## [1] 0.7105263

ppv\_train <- cftable\_train[1]/(cftable\_train[1] + cftable\_train[3])  
ppv\_train

## [1] 0.8275261

npv\_train <- cftable\_train[4]/(cftable\_train[2] + cftable\_train[4])  
npv\_train

## [1] 0.7665615

error\_train <- mean(predict\_logistic\_2!=train\_copy$Survived)  
acc\_train <- 1 - error\_train  
acc\_train

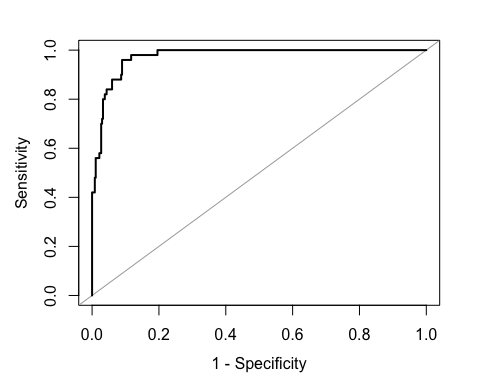
## [1] 0.8058361

roc\_test <- roc(test\_copy$Survived,predict\_logistic\_test)

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

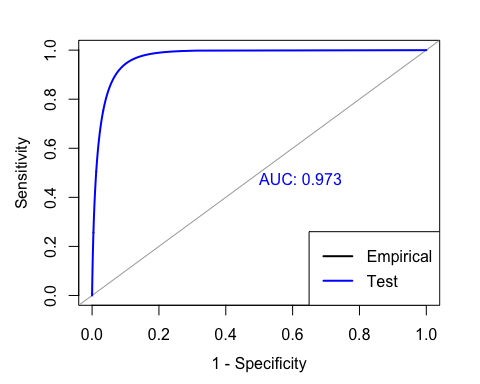
plot.roc(roc\_test, col=par("fg"),plot=TRUE,print.auc = FALSE, legacy.axes = TRUE, asp =NA)



roc\_test

##   
## Call:  
## roc.default(response = test\_copy$Survived, predictor = predict\_logistic\_test)  
##   
## Data: predict\_logistic\_test in 368 controls (test\_copy$Survived 0) < 50 cases (test\_copy$Survived 1).  
## Area under the curve: 0.9743

plot.new() # Initialize a new plot  
plot.roc(smooth(roc\_test), col="blue", plot=TRUE, print.auc = TRUE, legacy.axes = TRUE, asp = NA)  
legend("bottomright", legend=c("Empirical", "Test"), col=c(par("fg"), "blue"), lwd=2)

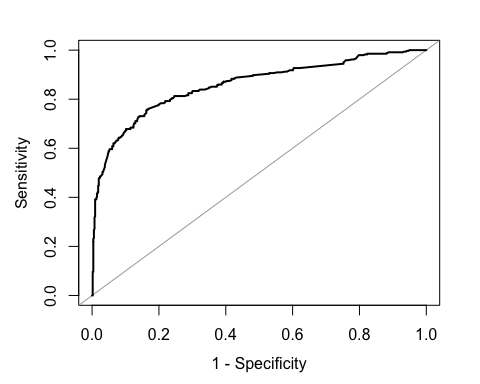


roc\_train <- roc(train\_copy$Survived,predict\_logistic\_train)

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

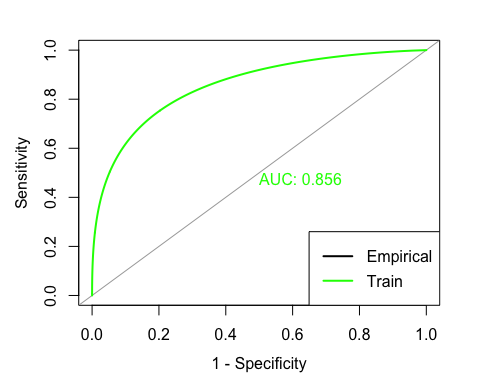
plot.roc(roc\_train, col=par("fg"),plot=TRUE,print.auc = FALSE, legacy.axes = TRUE, asp =NA)



roc\_test

##   
## Call:  
## roc.default(response = test\_copy$Survived, predictor = predict\_logistic\_test)  
##   
## Data: predict\_logistic\_test in 368 controls (test\_copy$Survived 0) < 50 cases (test\_copy$Survived 1).  
## Area under the curve: 0.9743

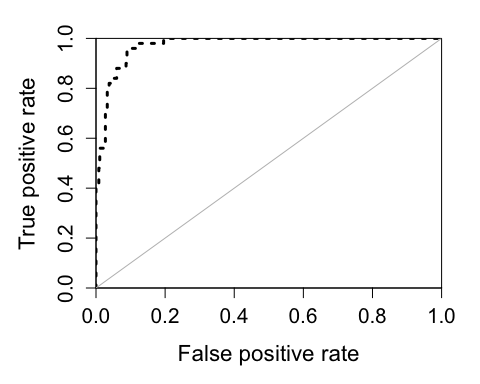
plot.new() # Initialize a new plot  
plot.roc(smooth(roc\_train), col="green", plot=TRUE, print.auc = TRUE, legacy.axes = TRUE, asp = NA)  
legend("bottomright", legend=c("Empirical", "Train"), col=c(par("fg"), "green"), lwd=2)



## The lines below attempted to plot both the ROC & AUC curves in the same curves  
  
pred <- prediction(predict\_logistic\_test,test\_copy$Survived)  
perf <- performance(pred,"tpr","fpr")  
par(mar=c(5,5,2,2),xaxs = "i",yaxs = "i",cex.axis=1.3,cex.lab=1.4)  
plot(perf,col="black",lty=3, lwd=3)  
auc <- performance(pred,"auc")  
auc

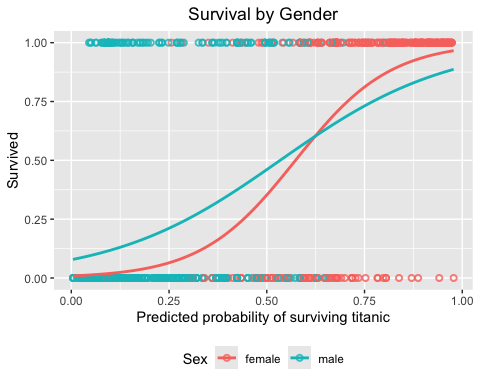
## A performance instance  
## 'Area under the ROC curve'

abline(0,1,col="grey")

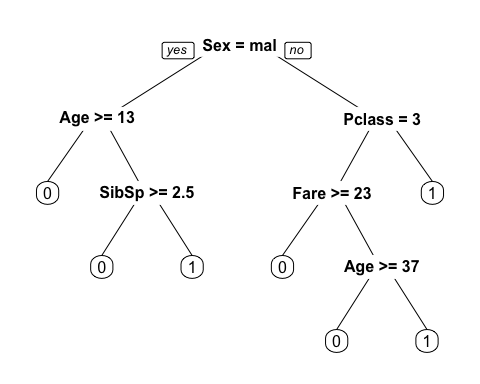


predicted\_data <- data.frame(probability\_of\_survival = logistic\_stepwise$fitted.value,Survived=train\_copy$Survived,Sex=train\_copy$Sex)  
# predicted\_data <- predicted\_data[order(predicted.data$probability.of.survival,decreasing = FALSE),]  
ggplot(data=predicted\_data,aes(x=probability\_of\_survival,y=as.numeric(Survived)-1,col=Sex))+  
 geom\_point(alpha=0.8,shape=1,stroke=1)+  
 xlab('Predicted probability of surviving titanic')+  
 ylab('Survived')+  
 stat\_smooth(method="glm", method.args=list(family="binomial"), se=FALSE,fullrange = TRUE)+  
 theme(legend.position = 'bottom',plot.title = element\_text(hjust = 0.5))+  
 ggtitle('Survival by Gender')

## `geom\_smooth()` using formula = 'y ~ x'



titanic\_cart <- rpart(formula = Survived ~., train\_copy, method = "class",  
 control = rpart.control(minsplit = 9, minbucket = 3,xval = 10))  
prp(titanic\_cart,roundint = F)



cart\_param <- titanic\_cart$cptable

cart\_param

## CP nsplit rel error xerror xstd  
## 1 0.44444444 0 1.0000000 1.0000000 0.04244576  
## 2 0.03070175 1 0.5555556 0.5555556 0.03574957  
## 3 0.01461988 5 0.4327485 0.4883041 0.03406141  
## 4 0.01000000 6 0.4181287 0.5058480 0.03452394

nrow(cart\_param)

## [1] 4

train\_error <- double(4)  
test\_error <- double(4)  
cv\_error <- double(4)  
for(i in 1:nrow(cart\_param)){  
 alpha <- cart\_param[i, 'CP']  
 train\_temp <- table(train\_copy$Survived, predict(prune(titanic\_cart, cp = alpha),  
 train\_copy,type = "class"))  
 train\_error[i] <- 1 - sum(diag(train\_temp))/sum(train\_temp)  
 cv\_error[i] <- cart\_param[i, "xerror"] \* cart\_param[i, "rel error"]  
 test\_temp <- table(test\_copy$Survived,predict(prune(titanic\_cart,cp = alpha),  
 test\_copy,type = "class"))  
 test\_error[i] <- 1 - sum(diag(test\_temp))/sum(test\_temp)  
}

train\_error

## [1] 0.3838384 0.2132435 0.1661055 0.1604938

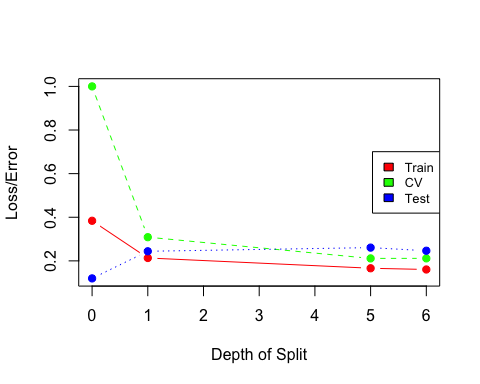
test\_error

## [1] 0.1196172 0.2440191 0.2607656 0.2464115

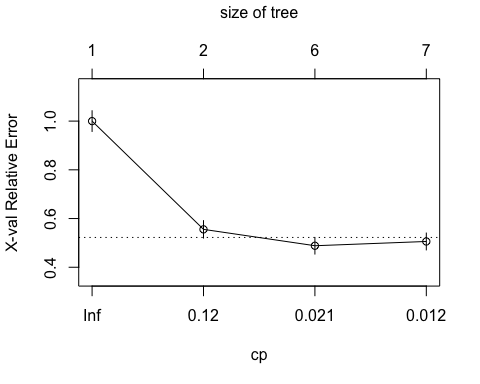
cv\_error

## [1] 1.0000000 0.3086420 0.2113129 0.2115095

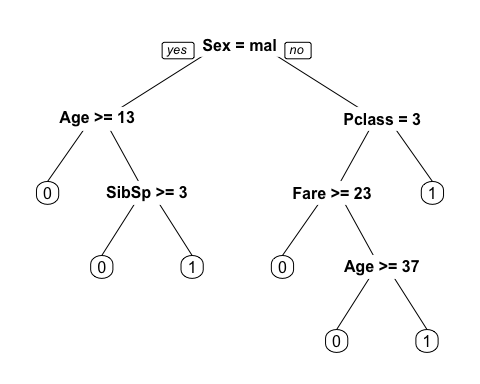
matplot(cart\_param[,"nsplit"],cbind(train\_error,cv\_error,test\_error),  
 pch = 19, col = c("red","green","blue"), type = "b",  
 ylab = "Loss/Error", xlab = "Depth of Split")  
  
legend("right", c('Train', 'CV', 'Test') ,col=seq\_len(3),cex=0.8,fill=c("red", "green", "blue"))



plotcp(titanic\_cart)



titanic\_prune <- prune(titanic\_cart,cp = cart\_param[4, 'CP'])  
prp(titanic\_prune)



titanic\_cfm\_cart <- table(test\_copy$Survived,predict(titanic\_prune, type = "class", newdata = test\_copy))  
titanic\_cfm\_cart

##   
## 0 1  
## 0 265 103  
## 1 0 50

cart\_accuracy <- sum(diag(titanic\_cfm\_cart))/sum(titanic\_cfm\_cart)  
cart\_accuracy

## [1] 0.7535885

cart\_train\_accuracy <- 1-train\_error[4]  
cart\_train\_accuracy

## [1] 0.8395062

cart\_test\_accuracy <- 1-test\_error[4]  
cart\_test\_accuracy

## [1] 0.7535885

cart\_cv\_accuracy <- 1-cv\_error[4]  
cart\_cv\_accuracy

## [1] 0.7884905

prune\_train\_acc <- 1-train\_error[4]  
prune\_train\_acc

## [1] 0.8395062

prune\_test\_acc <- 1-test\_error[4]  
prune\_test\_acc

## [1] 0.7535885

prune\_cv\_acc <- 1-cv\_error[4]  
prune\_cv\_acc

## [1] 0.7884905

control <- trainControl(method = "repeatedcv", number = 10, repeats = 10)  
  
logistic1 <- train(Survived ~ FamilySize+CabinPos+Deck+Pclass+Sex+Age+SibSp+Parch+Fare+Embarked+Title+Mother+Child, data = train, method = "glm", trControl = control, family = binomial(link = "logit"))

## prediction from rank-deficient fit; attr(\*, "non-estim") has doubtful cases

tree1 <- train(Survived ~ FamilySize+CabinPos+Deck+Pclass+Sex+Age+SibSp+Parch+Fare+Embarked+Title+Mother+Child, data = train, method = "rpart", trControl = control)

randomforest <- train(Survived ~ FamilySize+CabinPos+Deck+Pclass+Sex+Age+SibSp+Parch+Fare+Embarked+Title+Mother+Child, data = train, method = "rf", trControl = control)

grid <- expand.grid(size = 1, decay = 0.01)  
neuralnet <- train(Survived ~ FamilySize+CabinPos+Deck+Pclass+Sex+Age+SibSp+Parch+Fare+Embarked+Title+Mother+Child, data = train, method = "nnet", trControl = control,maxit = 1000, trace = FALSE, tuneGrid = grid )

acc\_rf <- paste0(round(max(randomforest$results$Accuracy),3)\*100,'%')  
cat('prediction accucary of random forest is ',acc\_rf,'.\n')

## prediction accucary of random forest is 82.5% .

acc\_nnet <- paste0(round(max(neuralnet$results$Accuracy),3)\*100,'%')  
cat('prediction accucary of neural net is ',acc\_nnet,'.\n')

## prediction accucary of neural net is 82% .

acc\_logistic1 <- paste0(round(max(logistic1$results$Accuracy),3)\*100,'%')  
cat('prediction accucary with logisitc regreesion is ',acc\_logistic1,'.\n')

## prediction accucary with logisitc regreesion is 81.5% .

acc\_tree1 <- paste0(round(max(tree1$results$Accuracy),3)\*100,'%')  
cat('prediction accucary with decision tree model is ',acc\_tree1,'.\n')

## prediction accucary with decision tree model is 81.1% .

logistic\_bayesian <- train(Survived ~ FamilySize+CabinPos+Deck+Pclass+Sex+Age+SibSp+Parch+Fare+Embarked+Title+Mother+Child, data = train, method = "bayesglm", trControl = control, family = binomial(link = "logit"))  
acc\_bayesian <- paste0(round(max(logistic\_bayesian$results$Accuracy),3)\*100,'%')

# Combine all the models and their accuracies into a data frame  
model\_accuracies <- data.frame(  
 Model = c("Logistic Regression", "Decision Tree", "Bayesian Logistic Regression",   
 "Random Forest", "Neural Network"),  
 Accuracy = c(acc\_logistic1, acc\_tree1, acc\_bayesian, acc\_rf, acc\_nnet)  
)  
  
# Print model accuracies  
cat("Model Accuracies:\n")

## Model Accuracies:

print(model\_accuracies)

## Model Accuracy  
## 1 Logistic Regression 81.5%  
## 2 Decision Tree 81.1%  
## 3 Bayesian Logistic Regression 82.5%  
## 4 Random Forest 82.5%  
## 5 Neural Network 82%