Titanic Data Analysis

Bahram Khanlarov

2022-08-08

Introduction

The sinking of the Titanic is one of the most infamous shipwrecks in history.

On April 15, 1912, during her maiden voyage, the widely considered "unsinkable" RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren't enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

We get the dataset from Kaggle platfoem through the link: https://www.kaggle.com/competitions/titanic/data We do basic data preparation and data exploration on this dataset.

Basic understanding of the data

We start reading our training and test set:

```
titanic_train <-read.csv("train.csv")
titanic_test <- read.csv("test.csv")
```

We can check the structure of the data using str():

```
str(titanic_train)
```

```
str(titanic_test)
```

The training set has 891 observations and 12 variables and the testing set has 418 observations and 11 variables. The traning set has 1 extra varible. Check which which one we are missing. I know we could see that in a very small dataset like this, but if its larger we want two compare them.

```
colnames_check <- colnames(titanic_train) %in% colnames(titanic_test)
colnames(titanic_train[colnames_check==FALSE])
```

```
## [1] "Survived"
```

As we can see we are missing the Survived in the test set. Which is correct because thats our challenge, we must predict this by creating a model.

```
#Use sapply(#object, class) to check the class of every column.
sapply(titanic_train, class)
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##   "integer"   "integer"   "integer" "character" "character"   "numeric"
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##   "integer"   "integer" "character"   "numeric" "character" "character"
```

We can see that the Survived and Pclass column are integers and Sex and Embarked character. But they are actually categorical variables. To convert them into categorical variables (or factors), use the factor() function.Survived is a nominal categorical variable, whereas Pclass is an ordinal categorical variable. For

an ordinal variable, we provide the order=TRUE and levels argument in the ascending order of the values(Pclass 3 < Pclass 2 < Pclass 1).

```
#change columns class
#Survived: from integer into factor
titanic_train$Survived = as.factor(titanic_train$Survived)
titanic_train$Sex = as.factor(titanic_train$Sex)
titanic_train$Embarked=as.factor(titanic_train$Embarked)
titanic_train$Pclass=factor(titanic_train$Pclass,order=TRUE, levels = c(3, 2, 1))
titanic_test$Sex = as.factor(titanic_test$Sex)
titanic_test$Embarked=as.factor(titanic_test$Embarked)
titanic_test$Pclass=factor(titanic_test$Pclass,order=TRUE, levels = c(3, 2, 1))
```

Let's look deeper into the training set, and check how many passengers that survived vs did not make it.

```
table(titanic_train$Survived)
```

```
##
##   0   1
## 549 342
```

Out of the 891 there are only 342 who survived it. Check also as proportions.

```
prop.table(table(titanic_train$Survived))
```

```
##
##         0         1
## 0.6161616 0.3838384
```

A little more than one-third of the passengers survived the disaster. Now see if there is a difference between males and females that survived vs males that passed away.

```
table(titanic_train$Sex, titanic_train$Survived)
```

```
##
##            0   1
##   female  81 233
##   male   468 109
```

```
prop.table(table(titanic_train$Sex, titanic_train$Survived),margin = 1)
```

```
##
##                   0         1
##   female 0.2579618 0.7420382
##   male   0.8110919 0.1889081
```

As we can see most of the female survived and most of the male did not make it.

Data Preparation

Now we need to clean the dataset to create our models. Note that it is important to explore the data so that we understand what elements need to be cleaned.

```
#missing data

is.na(titanic_train)
sum(is.na(titanic_train))
```

```
#This function shows us exactly how much values are missing in each column.
apply(titanic_train, MARGIN = 2, FUN = function(x) {sum(is.na(x))})
```

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

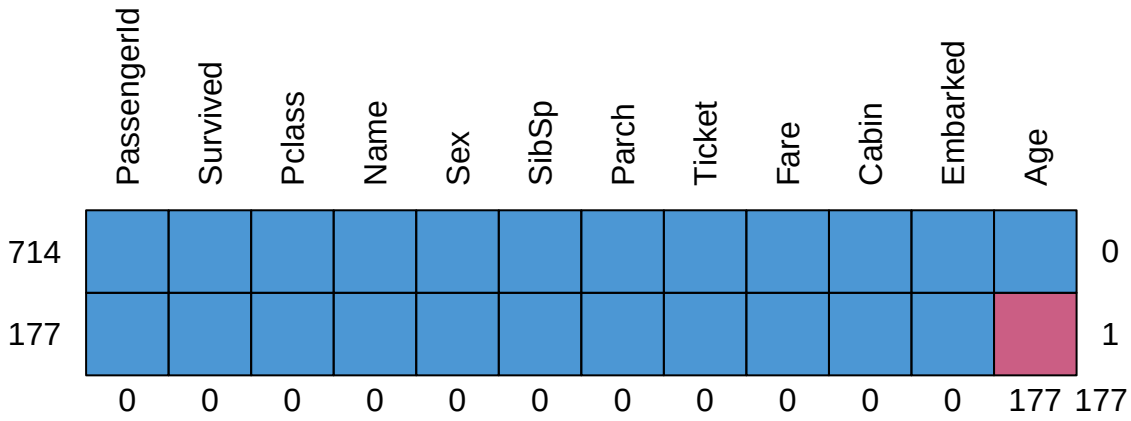
```
# Graphically check the missing data
library("mice")
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
missing_pattern <- md.pattern(titanic_train, rotate.names = TRUE)
```




```
colSums(is.na(titanic_test))
```

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

From the Training Data, the only thing missing is the age of 177 passengers of the 891 in the set. From the Test Data we are missing the age of 86 passengers and the Fare for 1 of the 418 in the dataset.

For the missing ages, it has been a common practice to use the median age, to replace missing age values. for both datasets, the returned value was 28 for the train_data and 27 for the test_data, so I will replace the missing values with these numbers.

```
#Filling missing values for Age
median(titanic_train$Age, na.rm=TRUE)
```

```
## [1] 28
```

```
median(titanic_test$Age, na.rm=TRUE)
```

```
## [1] 27
```

```
titanic_train$Age  <- ifelse(is.na(titanic_train$Age), 28, titanic_train$Age)
titanic_test$Age  <- ifelse(is.na(titanic_test$Age), 27, titanic_test$Age)
```

In the test data, there was one instances where the fare was missing. I found that there was 3rd Class passanger, named Thomas Storey, who was a 60 year old male, who embarked from Scottland that had a missing fare value. The rounded mean fare for 3rd class passangers that embarked from Scottland was 7.90, and I will replace it with that value.

```
titanic_test[!complete.cases(titanic_test$Fare),]
```

```
##     PassengerId Pclass               Name  Sex  Age SibSp Parch Ticket Fare
## 153        1044      3 Storey, Mr. Thomas male 60.5     0     0   3701   NA
##     Cabin Embarked
## 153              S
```

```
thrd_cl_fr <- subset(titanic_test, c(titanic_test$Pclass==3, titanic_test$Embarked=="S"))
m_fare <- round(median(thrd_cl_fr$Fare, na.rm=TRUE),2)
m_fare
```

```
## [1] 7.9
```

```
titanic_test$Fare <- ifelse(is.na(titanic_test$Fare), m_fare, titanic_test$Fare)
```

Running the code to check for NA for missing values after I cleaned the ages and the one Fare, I returned zero missing values. However I ran a table for the columns and found that the train_data had two rows with no Embark data. For this field the mode is “S”, so I will replace those two values with “S”.

```
sum(is.na(titanic_train))
```

```
## [1] 0
```

```
sum(is.na(titanic_test))
```

```
## [1] 0
```

```
table(titanic_train$Embarked)
```

```
##
##       C   Q   S
##   2 168  77 644
```

```
table(titanic_test$Embarked)
```

```
##
##   C   Q   S
## 102  46 270
```

```
m_embarked <-subset(titanic_train, titanic_train$Embarked==" ")
m_embarked
```

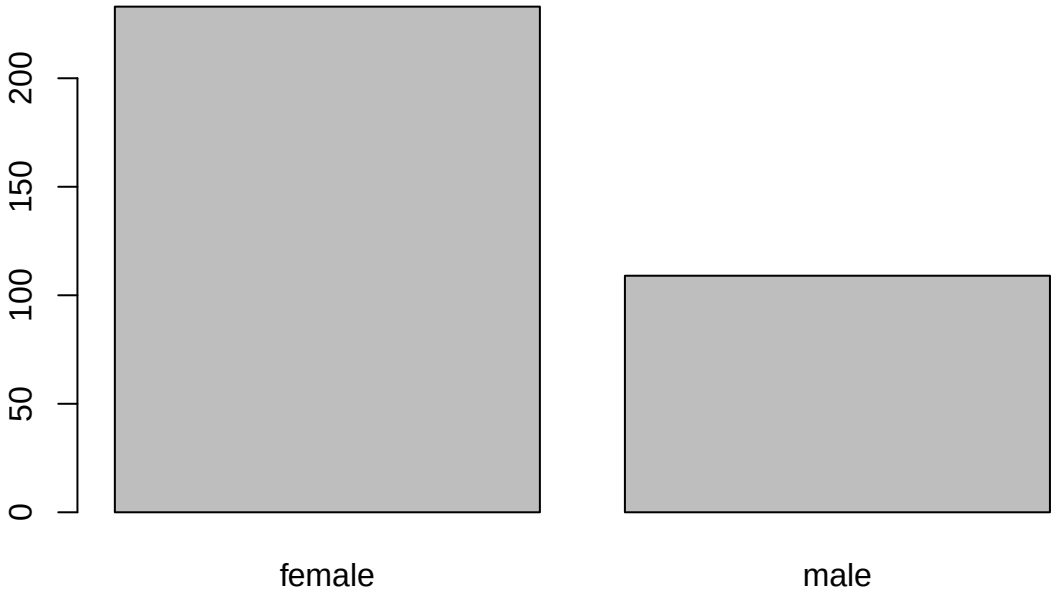
```
##  [1] PassengerId Survived    Pclass      Name        Sex         Age
##  [7] SibSp       Parch       Ticket      Fare        Cabin       Embarked
## <0 rows> (or 0-length row.names)
```

```
titanic_train[titanic_train$Embarked==" ", "Embarked"] <- "S"
```

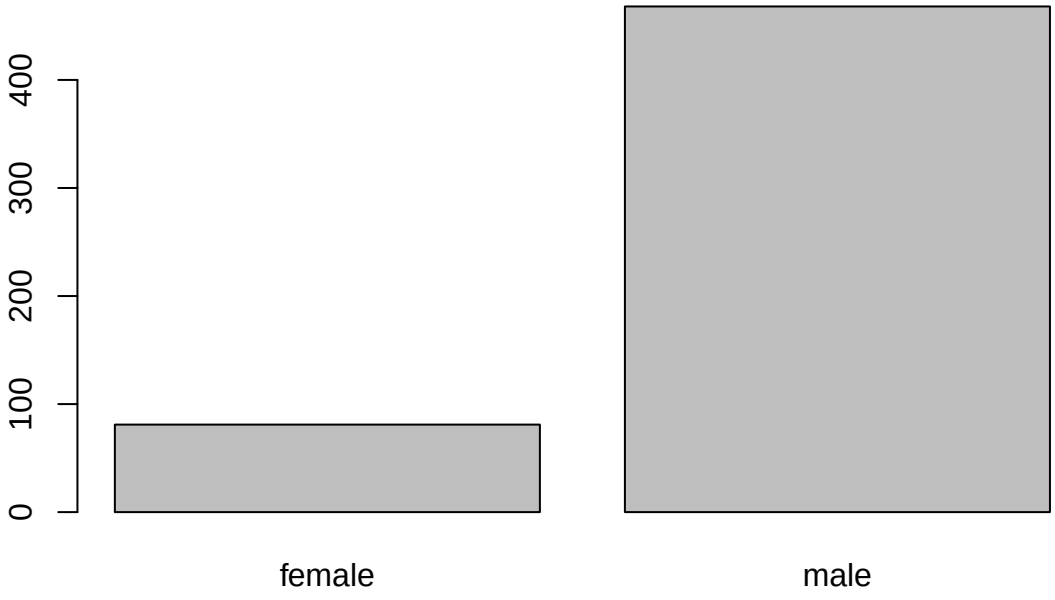
```
#separate data
titanic_survivor = titanic_train[titanic_train$Survived == 1, ]
titanic_nonsurvivor = titanic_train[titanic_train$Survived == 0, ]
```

Data Visualization

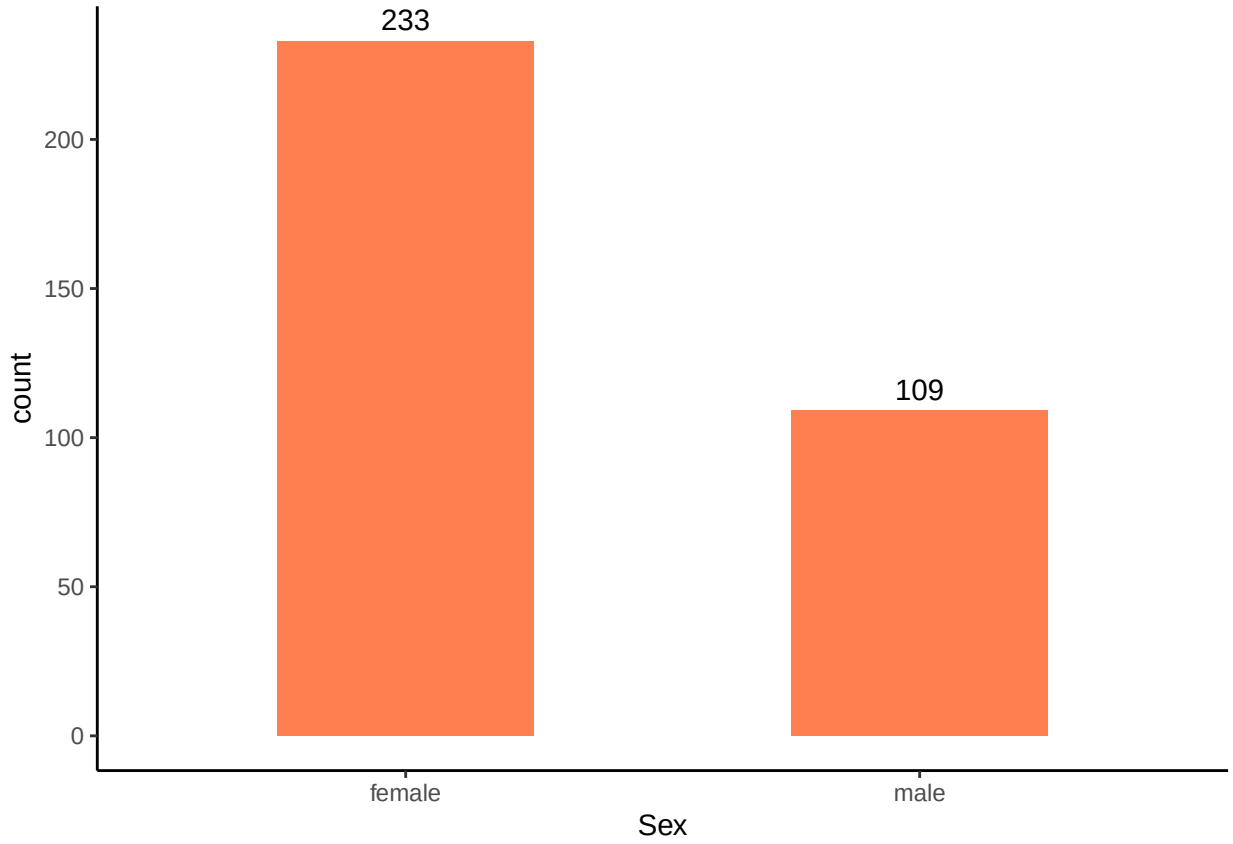
```r
#barchart
barplot(table(titanic_survivor$Sex))
```



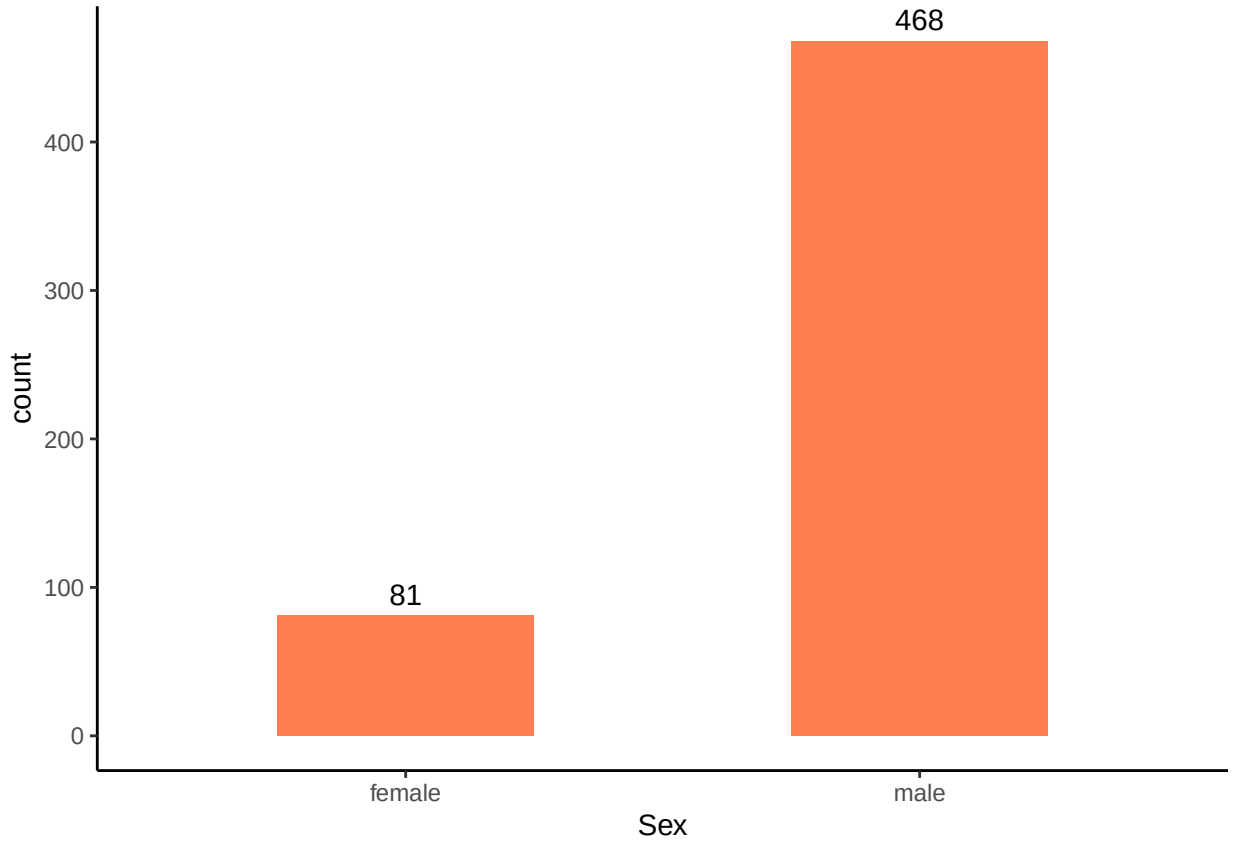

```r
barplot(table(titanic_nonsurvivor$Sex))
```



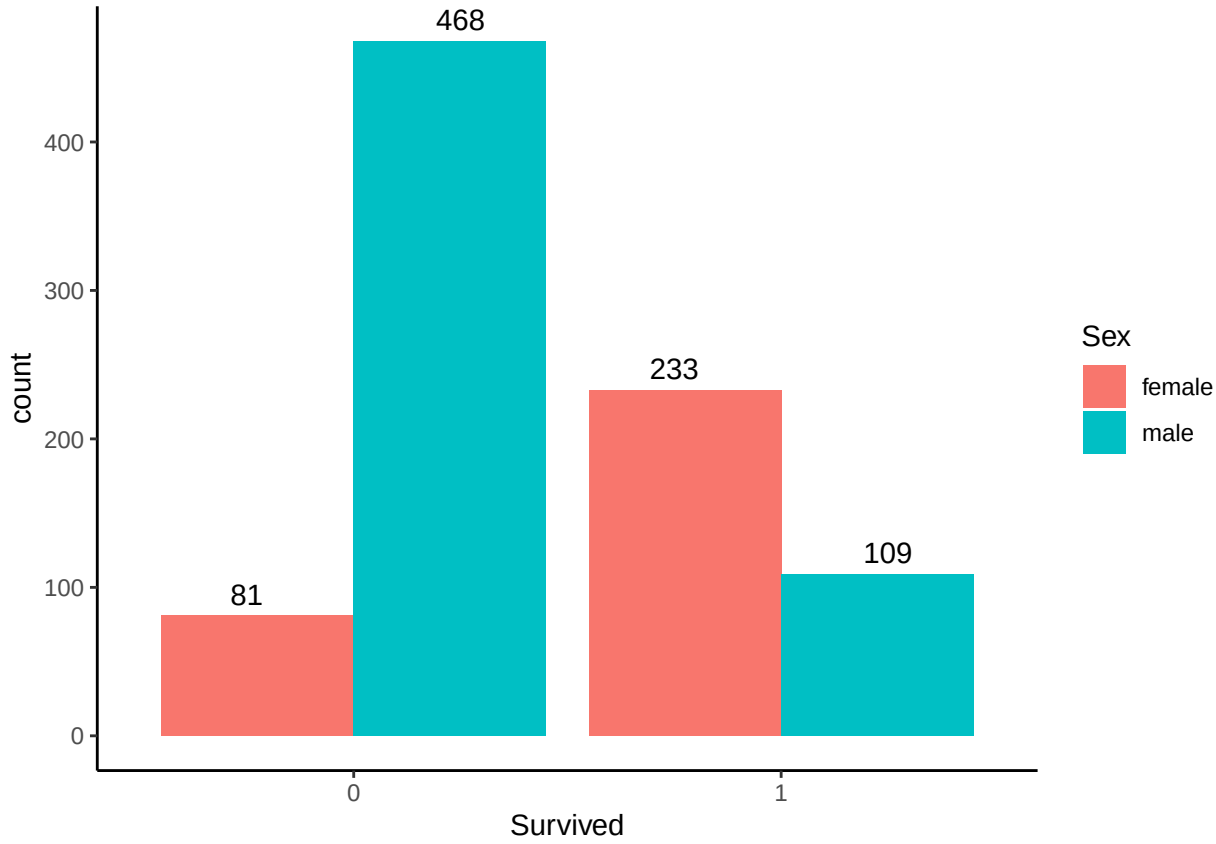

```r
## number of  survivals by Sex

library(ggplot2)
ggplot(titanic_survivor, aes(x = Sex)) +
  geom_bar(width=0.5, fill = "coral") +
  geom_text(stat='count', aes(label=stat(count)), vjust=-0.5) +
  theme_classic()
```



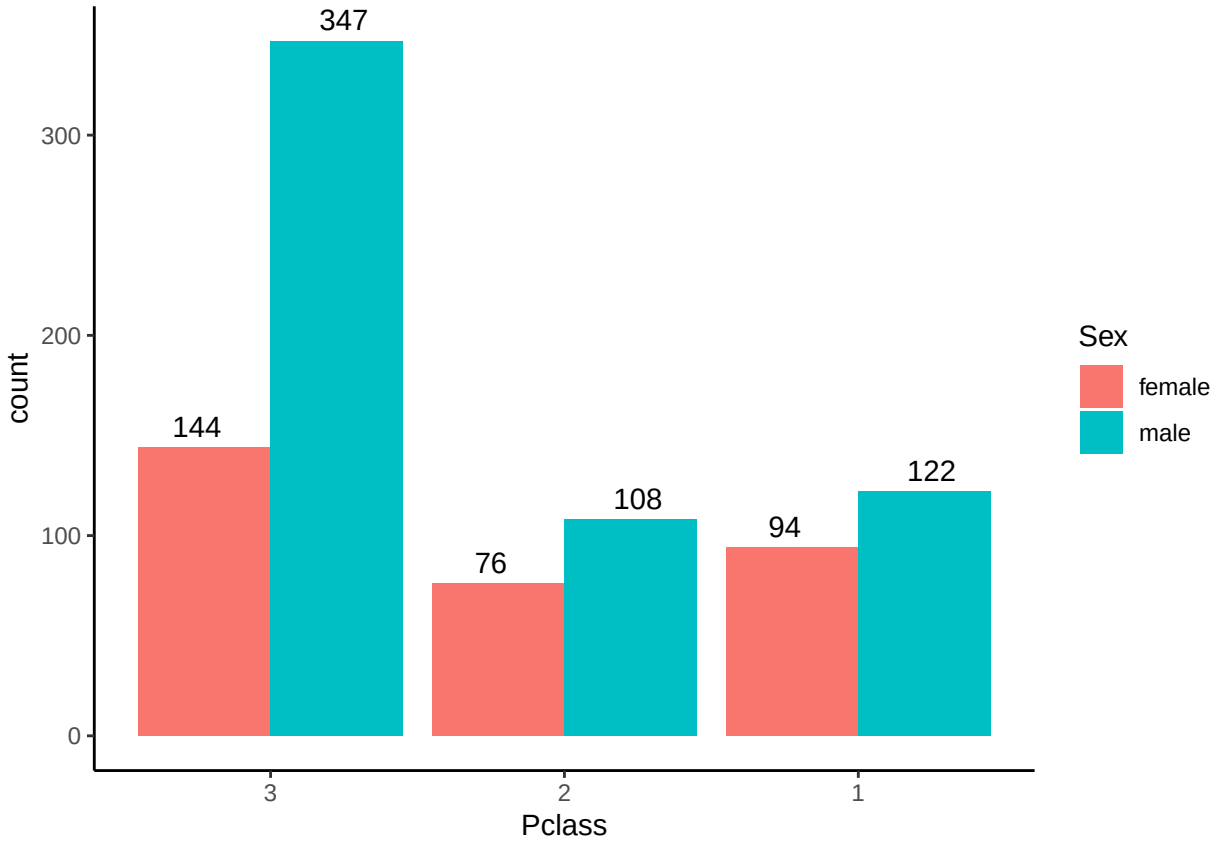

```
## number of  Non survivals by Sex

library(ggplot2)
ggplot(titanic_nonsurvivor, aes(x = Sex)) +
  geom_bar(width=0.5, fill = "coral") +
  geom_text(stat='count', aes(label=stat(count)), vjust=-0.5) +
  theme_classic()
```




```
ggplot(titanic_train, aes(x = Survived, fill=Sex)) +
 geom_bar(position = position_dodge()) +
 geom_text(stat='count',
           aes(label=stat(count)),
           position = position_dodge(width=1), vjust=-0.5)+
 theme_classic()
```

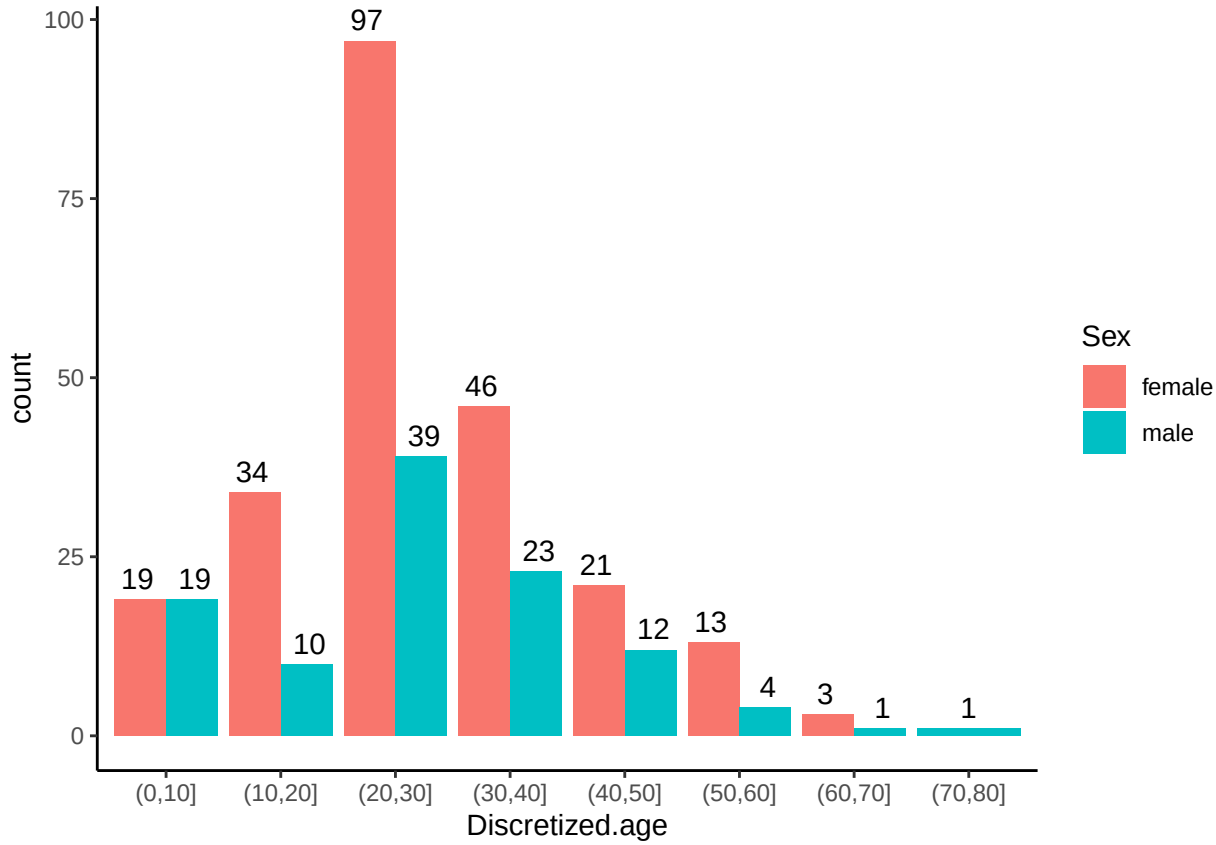



```
ggplot(titanic_train, aes(x = Pclass, fill=Sex)) +
 geom_bar(position = position_dodge()) +
 geom_text(stat='count',
           aes(label=stat(count)),
           position = position_dodge(width=1), vjust=-0.5)+
 theme_classic()
```



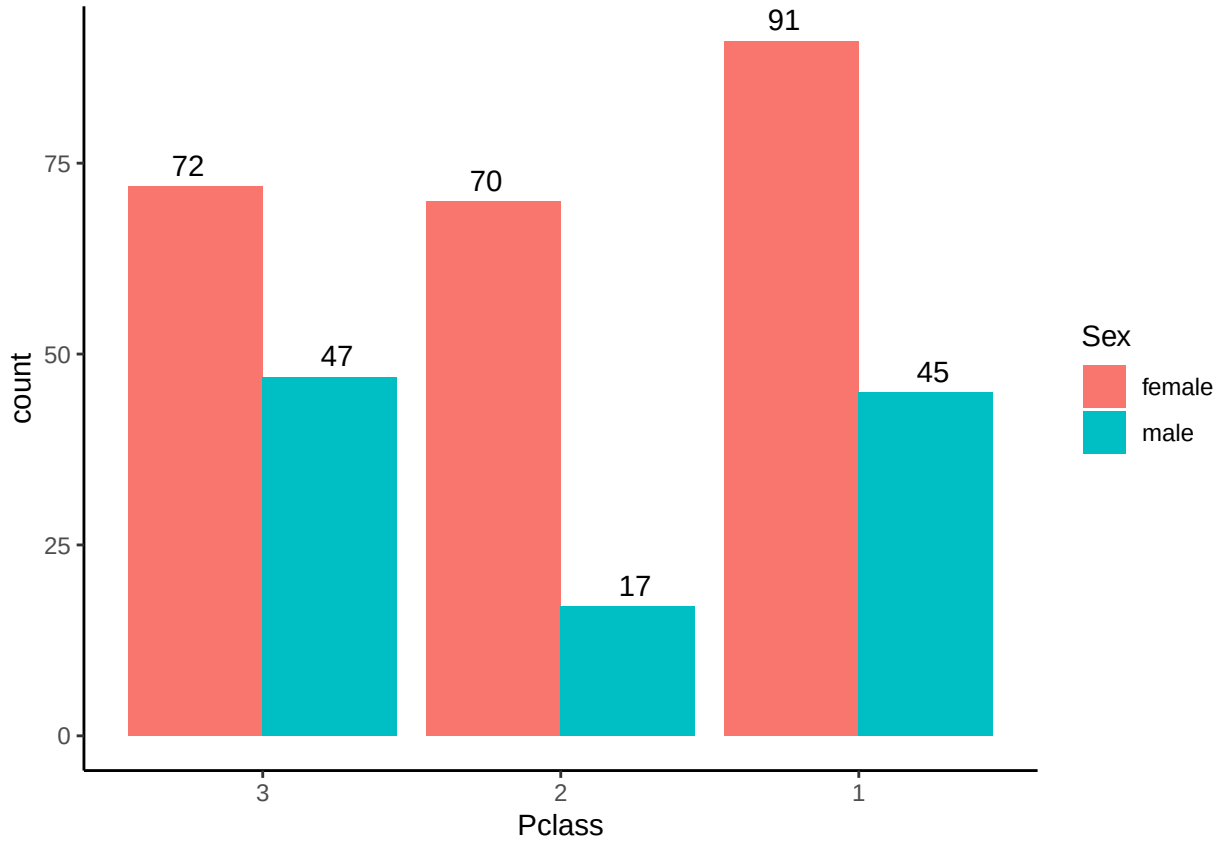
Here we have created a temporary attribute called Discretized.age which groups the ages with a span of 10 years. We discretize the age using the cut() function and specify the cuts in a vector. The temporary attribute it discarded after plotting. Most of the patients that died during hospitalization are in the age range from 70-80 years old.

```
#Discretize age to plot survival
titanic_survivor$Discretized.age = cut(titanic_survivor$Age, c(0,10,20,30,40,50,60,70,80,100))
# Plot discretized age
ggplot(titanic_survivor, aes(x = Discretized.age, fill=Sex)) +
  geom_bar(position = position_dodge()) +
  geom_text(stat='count', aes(label=stat(count)), position = position_dodge(width=1), vjust=-0.5)+
  theme_classic()
```

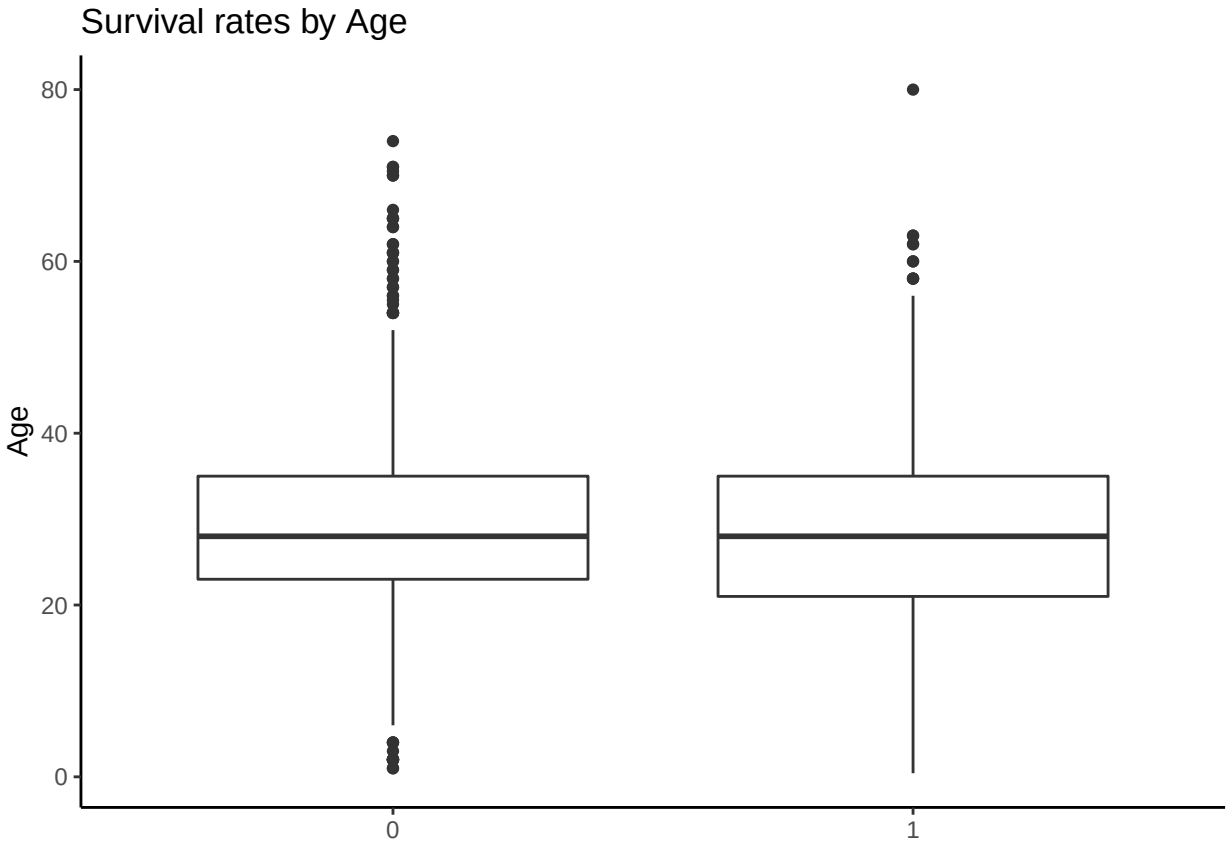



```
#data.frame$Discretized.age = NULL
```

```
ggplot(titanic_survivor, aes(x = Pclass, fill=Sex)) +
 geom_bar(position = position_dodge()) +
 geom_text(stat='count',
           aes(label=stat(count)),
           position = position_dodge(width=1), vjust=-0.5)+
 theme_classic()
```




```
# Boxplot
library(magrittr)
titanic_train %>%
  ggplot(aes(x = Survived, y = Age)) +
  geom_boxplot() +
  theme_classic() +
  labs(title = "Survival rates by Age", x = NULL)
```



Passengers who survived seems to have a lower median age.

Decision Tree Model

Now I am going to train a model to predict survivability and then test the model. The model will be saved and submitted to Kaggle. The file I send to Kaggle needs to have the Passenger ID and the prediction of whether or not that passenger survived.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tidyr::extract()   masks magrittr::extract()
## x dplyr::filter()    masks mice::filter(), stats::filter()
## x dplyr::lag()       masks stats::lag()
## x purrr::set_names() masks magrittr::set_names()
```

```
library(caret)
```

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

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
```

```
##
##     lift
```

```
library(rpart)
set.seed(123)  # for reproducibility
model1 <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked, data=titanic_train,meth
```

```
library(caret)
options(digits=4)
# assess the model's accuracy with train dataset by make a prediction on the train data.
Predict_model1_train <- predict(model1, titanic_train, type = "class")
#build a confusion matrix to make comparison
conMat <- confusionMatrix(as.factor(Predict_model1_train), as.factor(titanic_train$Survived))
#show confusion matrix
conMat$table
```

```
##           Reference
## Prediction   0   1
##          0 498  98
##          1  51 244
```

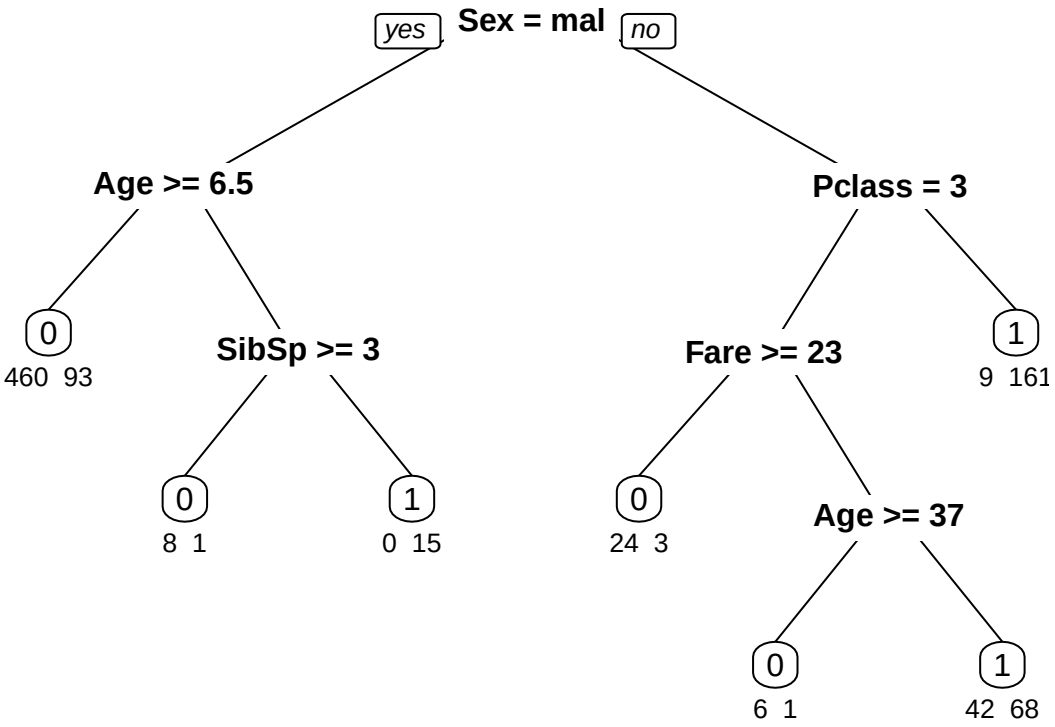
A brief assessment shows our model1's accuracy is 83.28%. It is not bad! Let us use this model to make a prediction on test dataset.

```
#show percentage of same values - accuracy
predict_train_accuracy <- conMat$overall["Accuracy"]
predict_train_accuracy
```

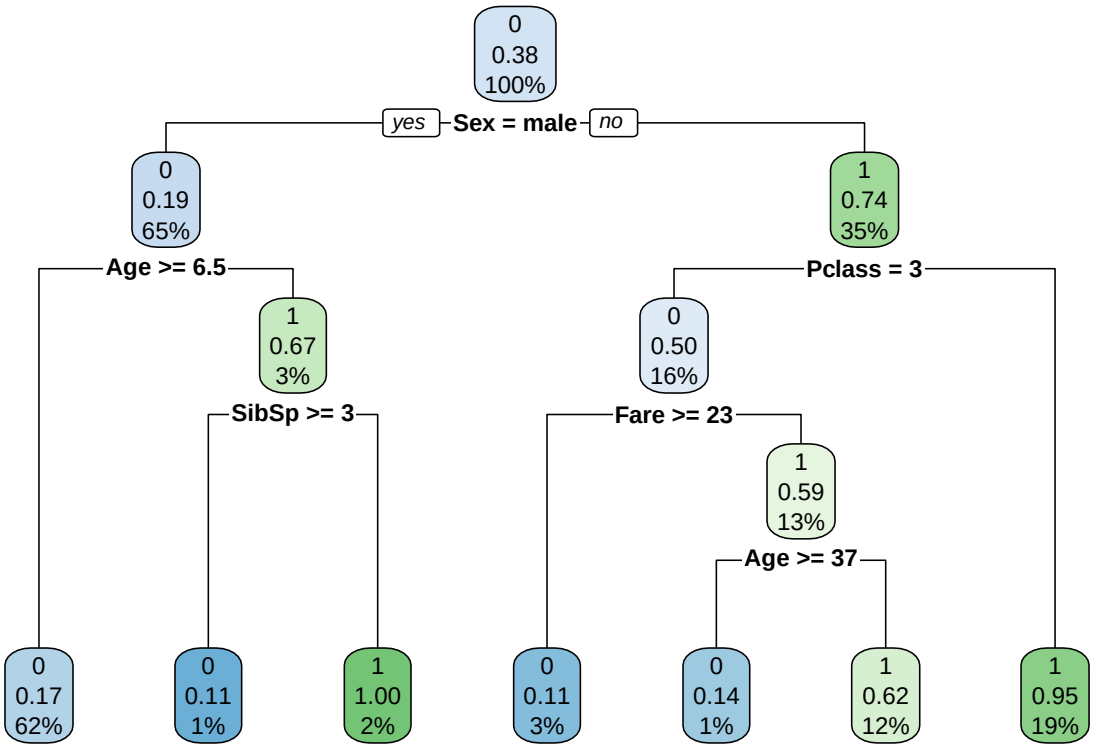
```
## Accuracy
##   0.8328
```

```
# The firs prediction produced by the first decision tree which only used one predictor Sex
Prediction1 <- predict(model1, titanic_test, type = "class")
```

```
# plot our full house classifier
library(rpart.plot)
prp(model1, type = 0, extra = 1, under = TRUE)
```




```
# plot our full house classifier
rpart.plot(model1)
```



Our prediction is produced. Let us submit to Kaggle for an evaluation. We need to convert our prediction into Kaggle's required format and save it into a file and name it as "Tree_Model1.CSV". Here, the importance is knowing the procedure.

```
# produce a submit with Kaggle required format that is only two attributes: PassengerId and Survived
submit1 <- data.frame(PassengerId = titanic_test$PassengerId, Survived = Prediction1)
# Write it into a file "Tree_Model1.CSV"
write.csv(submit1, file = "/Users/bkhan/Documents/Projects/titanic/Tree_Model1.csv", row.names = FALSE)
```

We check our prediction model's performance. We check our prediction's death and survive ratio on the test dataset and compare with the same ratio on the train dataset.

```
# Inspect prediction
summary(submit1$Survived)
```

```
##   0   1
## 272 146
```

```
prop.table(table(submit1$Survived, dnn="Test survive percentage"))
```

```
## Test survive percentage
##      0      1
## 0.6507 0.3493
```

```
#train survive ratio
prop.table(table(as.factor(titanic_train$Survived), dnn="Train survive percentage"))
```

```
## Train survive percentage
##      0      1
## 0.6162 0.3838
```

The result shows that among a total of 418 passengers in the test dataset, 272 passengers predicted non survived (with survived value 0), which counts as 65%, and 146 passengers predicted to be survived (with survived value 1) and which count as 35%. This is not too far from the ratio on the training dataset, which was 62% non survived and 38% survived.

```
print.data.frame(submit1)
```

```
##     PassengerId Survived
## 1           892        0
## 2           893        0
## 3           894        0
## 4           895        0
## 5           896        1
## 6           897        0
## 7           898        1
## 8           899        0
## 9           900        1
## 10          901        0
## 11          902        0
## 12          903        0
## 13          904        1
## 14          905        0
## 15          906        1
## 16          907        1
## 17          908        0
## 18          909        0
## 19          910        1
## 20          911        0
## 21          912        0
## 22          913        0
```

```
## 23          914         1
## 24          915         0
## 25          916         1
## 26          917         0
## 27          918         1
## 28          919         0
## 29          920         0
## 30          921         0
## 31          922         0
## 32          923         0
## 33          924         1
## 34          925         0
## 35          926         0
## 36          927         0
## 37          928         1
## 38          929         1
## 39          930         0
## 40          931         0
## 41          932         0
## 42          933         0
## 43          934         0
## 44          935         1
## 45          936         1
## 46          937         0
## 47          938         0
## 48          939         0
## 49          940         1
## 50          941         1
## 51          942         0
## 52          943         0
## 53          944         1
## 54          945         1
## 55          946         0
## 56          947         0
## 57          948         0
## 58          949         0
## 59          950         0
## 60          951         1
## 61          952         0
## 62          953         0
## 63          954         0
## 64          955         1
## 65          956         0
## 66          957         1
## 67          958         1
## 68          959         0
## 69          960         0
## 70          961         1
## 71          962         1
## 72          963         0
## 73          964         1
## 74          965         0
## 75          966         1
## 76          967         0
```

```
## 77          968        0
## 78          969        1
## 79          970        0
## 80          971        1
## 81          972        1
## 82          973        0
## 83          974        0
## 84          975        0
## 85          976        0
## 86          977        0
## 87          978        1
## 88          979        1
## 89          980        1
## 90          981        1
## 91          982        1
## 92          983        0
## 93          984        1
## 94          985        0
## 95          986        0
## 96          987        0
## 97          988        1
## 98          989        0
## 99          990        1
## 100         991        0
## 101         992        1
## 102         993        0
## 103         994        0
## 104         995        0
## 105         996        1
## 106         997        0
## 107         998        0
## 108         999        0
## 109        1000        0
## 110        1001        0
## 111        1002        0
## 112        1003        1
## 113        1004        1
## 114        1005        1
## 115        1006        1
## 116        1007        0
## 117        1008        0
## 118        1009        1
## 119        1010        0
## 120        1011        1
## 121        1012        1
## 122        1013        0
## 123        1014        1
## 124        1015        0
## 125        1016        0
## 126        1017        1
## 127        1018        0
## 128        1019        1
## 129        1020        0
## 130        1021        0
```

```
## 131        1022        0
## 132        1023        0
## 133        1024        0
## 134        1025        0
## 135        1026        0
## 136        1027        0
## 137        1028        0
## 138        1029        0
## 139        1030        1
## 140        1031        0
## 141        1032        0
## 142        1033        1
## 143        1034        0
## 144        1035        0
## 145        1036        0
## 146        1037        0
## 147        1038        0
## 148        1039        0
## 149        1040        0
## 150        1041        0
## 151        1042        1
## 152        1043        0
## 153        1044        0
## 154        1045        1
## 155        1046        0
## 156        1047        0
## 157        1048        1
## 158        1049        1
## 159        1050        0
## 160        1051        1
## 161        1052        1
## 162        1053        0
## 163        1054        1
## 164        1055        0
## 165        1056        0
## 166        1057        1
## 167        1058        0
## 168        1059        0
## 169        1060        1
## 170        1061        1
## 171        1062        0
## 172        1063        0
## 173        1064        0
## 174        1065        0
## 175        1066        0
## 176        1067        1
## 177        1068        1
## 178        1069        0
## 179        1070        1
## 180        1071        1
## 181        1072        0
## 182        1073        0
## 183        1074        1
## 184        1075        0
```

```
## 185        1076        1
## 186        1077        0
## 187        1078        1
## 188        1079        0
## 189        1080        0
## 190        1081        0
## 191        1082        0
## 192        1083        0
## 193        1084        0
## 194        1085        0
## 195        1086        0
## 196        1087        0
## 197        1088        1
## 198        1089        1
## 199        1090        0
## 200        1091        1
## 201        1092        1
## 202        1093        1
## 203        1094        0
## 204        1095        1
## 205        1096        0
## 206        1097        0
## 207        1098        1
## 208        1099        0
## 209        1100        1
## 210        1101        0
## 211        1102        0
## 212        1103        0
## 213        1104        0
## 214        1105        1
## 215        1106        0
## 216        1107        0
## 217        1108        1
## 218        1109        0
## 219        1110        1
## 220        1111        0
## 221        1112        1
## 222        1113        0
## 223        1114        1
## 224        1115        0
## 225        1116        1
## 226        1117        1
## 227        1118        0
## 228        1119        1
## 229        1120        0
## 230        1121        0
## 231        1122        0
## 232        1123        1
## 233        1124        0
## 234        1125        0
## 235        1126        0
## 236        1127        0
## 237        1128        0
## 238        1129        0
```

```
## 239        1130        1
## 240        1131        1
## 241        1132        1
## 242        1133        1
## 243        1134        0
## 244        1135        0
## 245        1136        0
## 246        1137        0
## 247        1138        1
## 248        1139        0
## 249        1140        1
## 250        1141        1
## 251        1142        1
## 252        1143        0
## 253        1144        0
## 254        1145        0
## 255        1146        0
## 256        1147        0
## 257        1148        0
## 258        1149        0
## 259        1150        1
## 260        1151        0
## 261        1152        0
## 262        1153        0
## 263        1154        1
## 264        1155        1
## 265        1156        0
## 266        1157        0
## 267        1158        0
## 268        1159        0
## 269        1160        1
## 270        1161        0
## 271        1162        0
## 272        1163        0
## 273        1164        1
## 274        1165        1
## 275        1166        0
## 276        1167        1
## 277        1168        0
## 278        1169        0
## 279        1170        0
## 280        1171        0
## 281        1172        1
## 282        1173        1
## 283        1174        1
## 284        1175        1
## 285        1176        1
## 286        1177        0
## 287        1178        0
## 288        1179        0
## 289        1180        0
## 290        1181        0
## 291        1182        0
## 292        1183        1
```

```
## 293        1184        0
## 294        1185        0
## 295        1186        0
## 296        1187        0
## 297        1188        1
## 298        1189        0
## 299        1190        0
## 300        1191        0
## 301        1192        0
## 302        1193        0
## 303        1194        0
## 304        1195        0
## 305        1196        1
## 306        1197        1
## 307        1198        0
## 308        1199        1
## 309        1200        0
## 310        1201        0
## 311        1202        0
## 312        1203        0
## 313        1204        0
## 314        1205        0
## 315        1206        1
## 316        1207        1
## 317        1208        0
## 318        1209        0
## 319        1210        0
## 320        1211        0
## 321        1212        0
## 322        1213        0
## 323        1214        0
## 324        1215        0
## 325        1216        1
## 326        1217        0
## 327        1218        1
## 328        1219        0
## 329        1220        0
## 330        1221        0
## 331        1222        1
## 332        1223        0
## 333        1224        0
## 334        1225        1
## 335        1226        0
## 336        1227        0
## 337        1228        0
## 338        1229        0
## 339        1230        0
## 340        1231        0
## 341        1232        0
## 342        1233        0
## 343        1234        0
## 344        1235        1
## 345        1236        0
## 346        1237        1
```

```
## 347        1238        0
## 348        1239        0
## 349        1240        0
## 350        1241        1
## 351        1242        1
## 352        1243        0
## 353        1244        0
## 354        1245        0
## 355        1246        1
## 356        1247        0
## 357        1248        1
## 358        1249        0
## 359        1250        0
## 360        1251        1
## 361        1252        0
## 362        1253        1
## 363        1254        1
## 364        1255        0
## 365        1256        1
## 366        1257        0
## 367        1258        0
## 368        1259        0
## 369        1260        1
## 370        1261        0
## 371        1262        0
## 372        1263        1
## 373        1264        0
## 374        1265        0
## 375        1266        1
## 376        1267        1
## 377        1268        1
## 378        1269        0
## 379        1270        0
## 380        1271        0
## 381        1272        0
## 382        1273        0
## 383        1274        1
## 384        1275        1
## 385        1276        0
## 386        1277        1
## 387        1278        0
## 388        1279        0
## 389        1280        0
## 390        1281        0
## 391        1282        0
## 392        1283        1
## 393        1284        0
## 394        1285        0
## 395        1286        0
## 396        1287        1
## 397        1288        0
## 398        1289        1
## 399        1290        0
## 400        1291        0
```

```
## 401        1292        1
## 402        1293        0
## 403        1294        1
## 404        1295        0
## 405        1296        0
## 406        1297        0
## 407        1298        0
## 408        1299        0
## 409        1300        1
## 410        1301        1
## 411        1302        1
## 412        1303        1
## 413        1304        1
## 414        1305        0
## 415        1306        1
## 416        1307        0
## 417        1308        0
## 418        1309        0
```