Titanic - Machine Learning from Disaster

uông Thị Thanh Thủy - Trần Gia Bảo - Hoàng Thị Cẩm Tú - Lê Kha

2021/01/24

1. GIỚI THIỆU

Nhóm đã chọn đề tài giải quyết và dự đoán về sự sống sót trên tàu Titanic - Titanic - Machine Learning from Disaster.

Nhóm đã sử dụng Megan Risdal làm nguồn cảm hứng và xây dựng dựa trên nó. Nhóm thực hiện một số kỹ thuật tính năng và nhiều hình ảnh hóa dữ liệu minh họa trong quá trình thực hiện. Sau đó, nhóm sẽ sử dụng Logistic Regression,randomForest,Classification tree,Support Vector Machine,Linear Discriminent , để tạo một mô hình dự đoán sự sống sót trên tàu Titanic. Tập lệnh có ba phần như sau: - Kỹ thuật tính năng (Feature engineering) - Thiếu giá trị (Missing value imputation) - Dự đoán (Prediction)

1.1 Load libraries and check the data

```
library('caret')
```

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

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

```
library('MASS')
library('e1071') # classification algorithm
library('grid')
library('gridExtra')
library('pscl')
```

```
## Classes and Methods for R developed in the
## Political Science Computational Laboratory
## Department of Political Science
## Stanford University
## Simon Jackman
## hurdle and zeroinfl functions by Achim Zeileis
```

```
library('rpart') # classification algorithm
library('ggplot2') # visualization
library('ggthemes') # visualization
library('scales') # visualization
library('dplyr') # data manipulation
```

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

```
## The following object is masked from 'package:gridExtra':
##
##     combine

## The following object is masked from 'package:MASS':
##
##     select

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

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

```
library('mice') # imputation
```

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

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

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

```
library('randomForest') # classification algorithm
```

```
## randomForest 4.6-14

## 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:gridExtra':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library('plyr') # feature correlation
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
library('corrplot') # feature correlation plotting
```

```
## corrplot 0.84 loaded
```

Các gói đã được thêm vào, bây giờ thêm các bảng có liên quan với train, test

```
#load data
train <- read.csv('C:/Users/uongt/Downloads/titanic/train.csv', na.strings = c("N/A","DIV/0!",""), stri
test  <- read.csv('C:/Users/uongt/Downloads/titanic/test.csv', na.strings = c("N/A","DIV/0!",""), string
full <- bind_rows(train,test) # Tạo tập dữ liệu mới với cả train và test
str(full) # xem cấu trúc dữ liệu
```

```
## 'data.frame':    1309 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  NA "C85" NA "C123" ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

Chúng ta đã biết về các biến của mình, loại lớp của chúng và một vài quan sát đầu tiên về mỗi biến. Biết rằng nhóm đang làm việc với 1309 quan sát của 12 biến và 1630 quan sát của 2 biến. Để làm cho mọi thứ rõ ràng hơn một chút vì một vài tên biến không hiển thị 100%, đây là những gì chúng ta phải giải quyết:

```
       full
```

Name Description PassengerId ID duy nhất của hành khách Survived Sống sót [1] hoặc đã chết [0] Pclass Hạng hành khách [1,2,3] Name Tên và chức danh của hành khách Sex Giới tính của hành khách Age Tuổi của hành khách SibSp Số anh chị em/vợ/chồng trên tàu Parch Số lượng cha mẹ/trẻ em trên tàu Ticket Số vé Fare Giá tiền Cabin Số Cabin Embarked Cảng Lên tàu

```
# gán các giá trị bị mất thành NA
full[full==""] <- NA
a<- apply(full,2,is.na)
summary(a)
```

```
##  PassengerId      Survived         Pclass           Name
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:1309      FALSE:891       FALSE:1309      FALSE:1309
##                  TRUE :418
##     Sex             Age            SibSp           Parch
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:1309      FALSE:1046      FALSE:1309      FALSE:1309
##                  TRUE :263
##    Ticket           Fare           Cabin          Embarked
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:1309      FALSE:1308      FALSE:295       FALSE:1307
##                  TRUE :1         TRUE :1014      TRUE :2
```

```
apply(a,2,sum)
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0         418           0           0           0         263
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           1        1014           2
```

2.FEATURE ENGINEERING

Bước thứ hai là bước quan trọng nhất! Mặc dù đã có rất nhiều tính năng, nhưng vẫn cần bổ sung các giá trị còn thiếu và cũng tìm kiếm các mối tương quan và tính năng có thể ảnh hưởng đến sự sống còn của hành khách.

2.1 Tên của hành khách

Biến đầu tiên mà làm việc là "name" của hành khách vì có thể chia nhỏ nó thành các biến có ý nghĩa bổ sung cung cấp các dự đoán hoặc được sử dụng để tạo thêm các tính năng mới. Ví dụ, "title" hành khách được chứa trong biến tên hành khách, chúng ta có thể sử dụng "surname" để đại diện cho gia đình.

```
# Trích xuất tiêu đề từ tên
full$Title<- gsub("^.*, (.*?)\\..*$", "\\1", full$Name)%>%
  gsub("[[:space:]]", "", .)

# tạo một bảng hiển thị tất cả các tổ hợp tiêu đề
table(full$Sex,full$Title)
```

```
##
##          Capt Col Don Dona  Dr Jonkheer Lady Major Master Miss Mlle Mme  Mr Mrs
##   female    0   0   0    1   1        0    1     0      0  260    2   1   0 197
##   male      1   4   1    0   7        1    0     2     61    0    0   0 757   0
```

```
##
##           Ms Rev Sir theCountess
##   female   2   0   0           1
##   male     0   8   1           0
```

=> Xem xét sự phân bố Chức danh (Title) cho mỗi giới tính (Sex)

Có những chức danh với lượng người chia sẽ chúng rất thấp. Nhiệm vụ là tổng hợp các chức danh hiếm trong các nhóm phụ của riêng chúng.

```
officer_title <- c('Capt','Col','Major')
community_title <- c('Dr','Sir')
rare_title <- c('Dona', 'Lady', 'the Countess', 'Don', 'Rev', 'Jonkheer')
full$Title[full$Title == 'Mlle']        <- 'Miss'
full$Title[full$Title == 'Ms']          <- 'Miss'
full$Title[full$Title == 'Mme']         <- 'Mrs'
full$Title[full$Title %in% rare_title]  <- 'Rare Title'
full$Title[full$Title %in% officer_title]  <- 'Crew'
full$Title[full$Title %in% community_title]  <- 'Member'
```

Danh sách các Title bây giờ có vẻ khái quát hơn.

```
table(full$Sex,full$Title)
```

```
##
##           Crew Master Member Miss  Mr Mrs Rare Title theCountess
##   female    0      0      1  264   0 198          2           1
##   male      7     61      8    0 757   0         10           0
```

```
# Trích xuất họ
full$Surname <- tolower(sapply(full$Name,function(x) {strsplit(x, split = '[,.]')[[1]][1]}))
```

2.2 Khả năng sinh tồn của các gia đình lớn

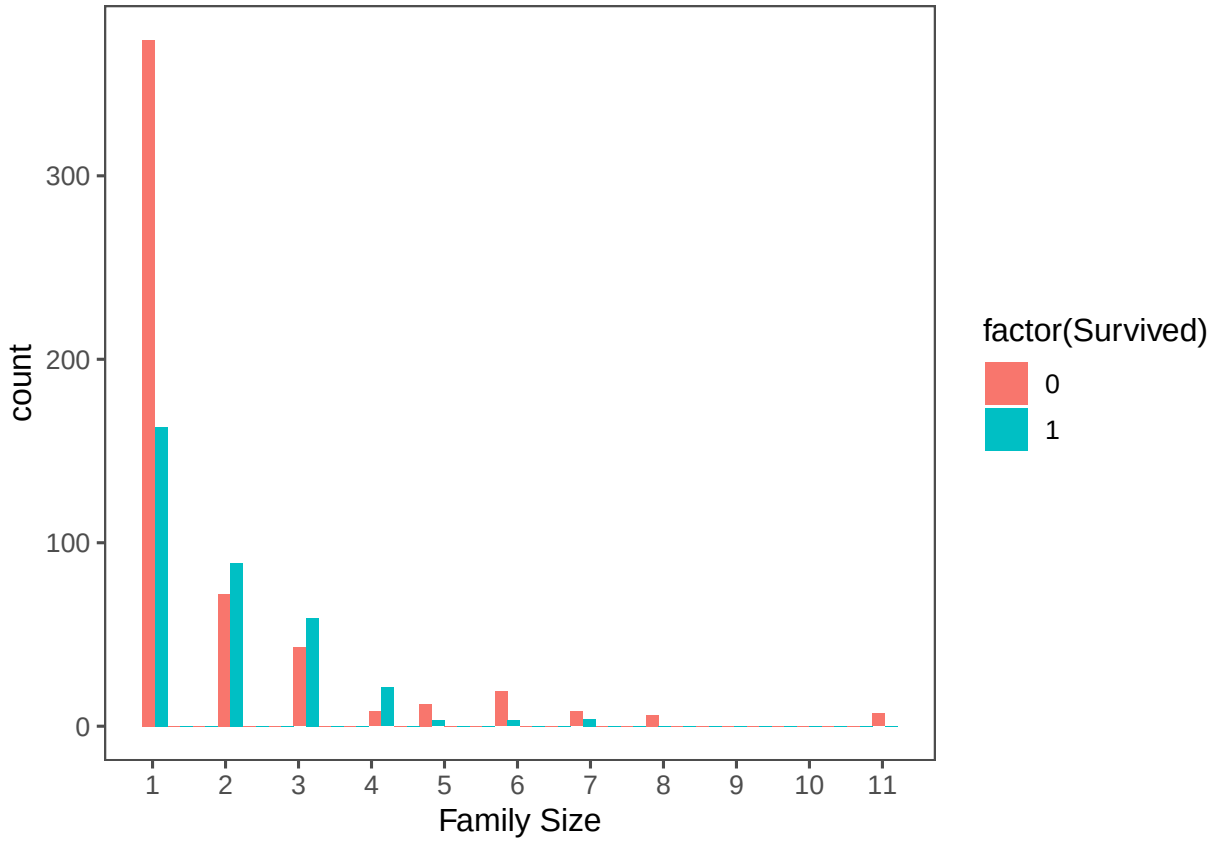
Chúng ta đã tách tên hành khách thành một số biến mới. Bây giờ, chúng ta có thể tạo một số biến gia đình mới. Đầu tiên sẽ tạo một biến quy mô gia đình dựa trên số anh chị em / vợ / chồng Và số con cái / cha mẹ. Dưới đây là tổng hợp các quy mô gia đình và kiểm tra tỷ lệ sống sót của họ.

```
full$Fsize <- full$SibSp + full$Parch + 1 # Hành khách + anh chị em/vợ/chồng + cha mẹ/con cái
full$IsAlone[full$Fsize==1] <- 'Alone' # Hành khách có đi du lịch 1 mình
full$IsAlone[full$Fsize!=1] <- 'Not Alone'
full$IsAlone <- factor(full$IsAlone)
full$Family <- paste(full$Surname, full$Fsize, sep='_') # Họ của những gia đình
```

Xem tỷ lệ sống sót của họ

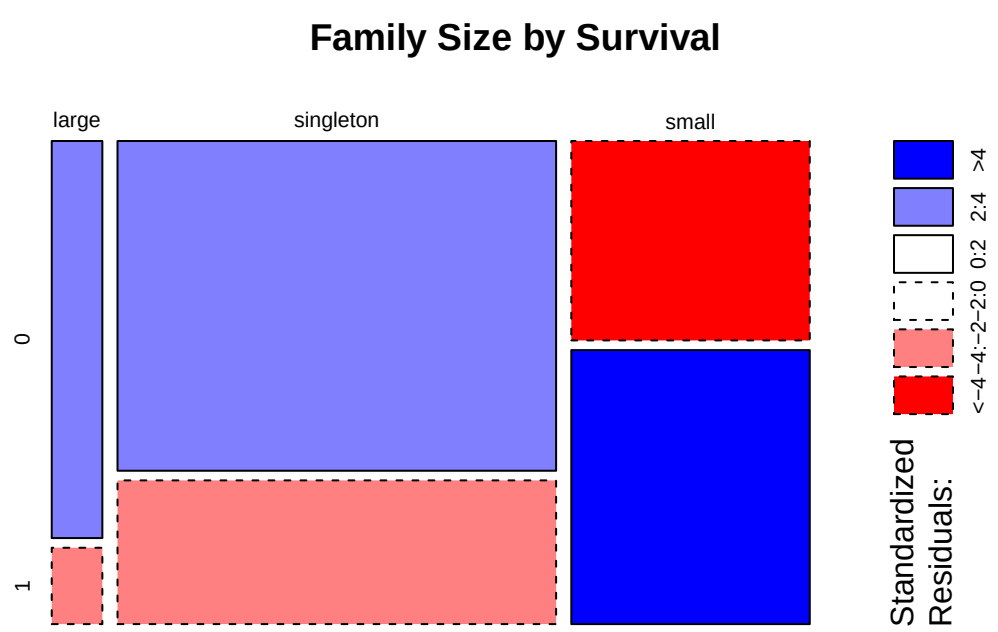
```
full <- full[order(full$PassengerId),] #sắp xếp dữ liệu
ggplot(full[1:891,], aes(x = Fsize, fill = factor(Survived))) +
  geom_bar(stat='bin', position='dodge') +
  scale_x_continuous(breaks=c(1:11)) +
  labs(x = 'Family Size') +
  theme_few()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Chúng ta có thể thấy rằng khả năng sống sót thấp đối với những người độc thân và những người có quy mô gia đình trên 4. Chúng ta có thể thu gọn biến này thành ba cấp độ sẽ hữu ích vì tương đối ít gia đình lớn. Bây giờ chúng ta sẽ tạo một biến quy mô gia đình tùy ý.

```
full$FsizeD[full$Fsize == 1] <- 'singleton'
full$FsizeD[full$Fsize < 5 & full$Fsize > 1] <- 'small'
full$FsizeD[full$Fsize > 4] <- 'large'
mosaicplot(table(full$FsizeD, full$Survived), main='Family Size by Survival', shade=TRUE) # Biểu đồ
```



Biểu đồ vẫn cho thấy sự sống thấp của người độc thân và gia đình lớn. Bây giờ, nhóm sẽ đo sự sinh sống theo biên độ tuổi, nhưng có 263 bị thiếu giá trị tuổi. Chúng ta sẽ giải quyết việc thiếu giá trị tuổi.

2.3 Nếu đi du lịch với bạn

Một chi tiết là có những vé trùng lặp. Điều này cho thấy mọi người đã đi du lịch cùng nhau mà không cần phải là họ hàng, gia đình của nhau. Những vé này cũng có giá vé giống hệt nhau, có nghĩa là giá vé nên được chia cho số lượng người mua nó. Điều này sẽ giúp chúng ta có cái nhìn tổng quan hơn về giá vé dựa trên các tính năng khác nhau.

```
n_occur <- data.frame(table(full$Ticket))
full <- merge(full,n_occur, by.x="Ticket", by.y="Var1", x.all=T)
full$Fare <- full$Fare / full$Freq
```

2.4 Những chữ cái này trong cột Cabin

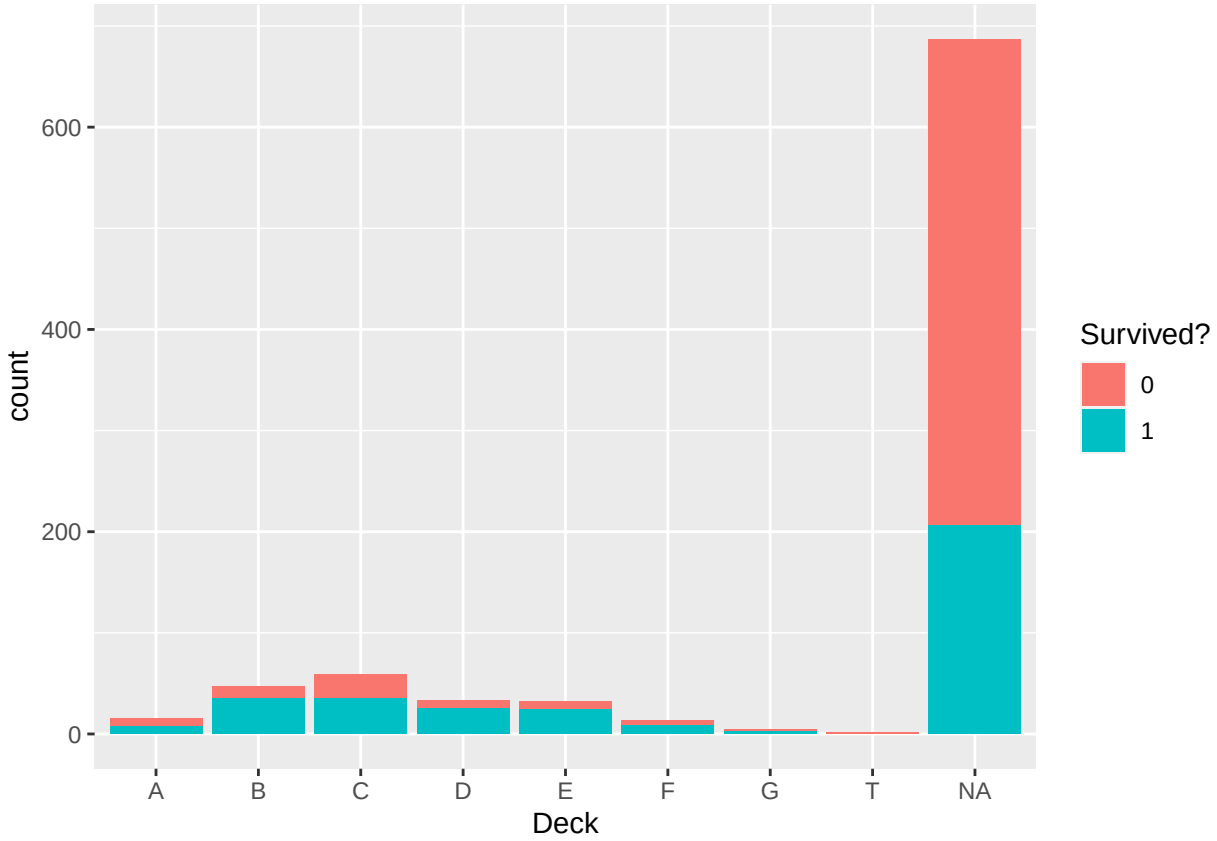
Các giá trị Cabin chỉ ra rằng có ba tham số. Tham số đầu tiên luôn là một chữ cái. Mỗi chữ cái tương ứng với "deck" mà căn phòng có thể được tìm thấy. Chúng ta phải điều tra xem nếu nằm trên một "deck" nhất định sẽ làm tăng cơ hội sống sót của họ.

```
full$Deck<-factor(sapply(full$Cabin, function(x) {strsplit(x, NULL)[[1]][1]}))
```

Quan sát biểu đồ Desk/Survived

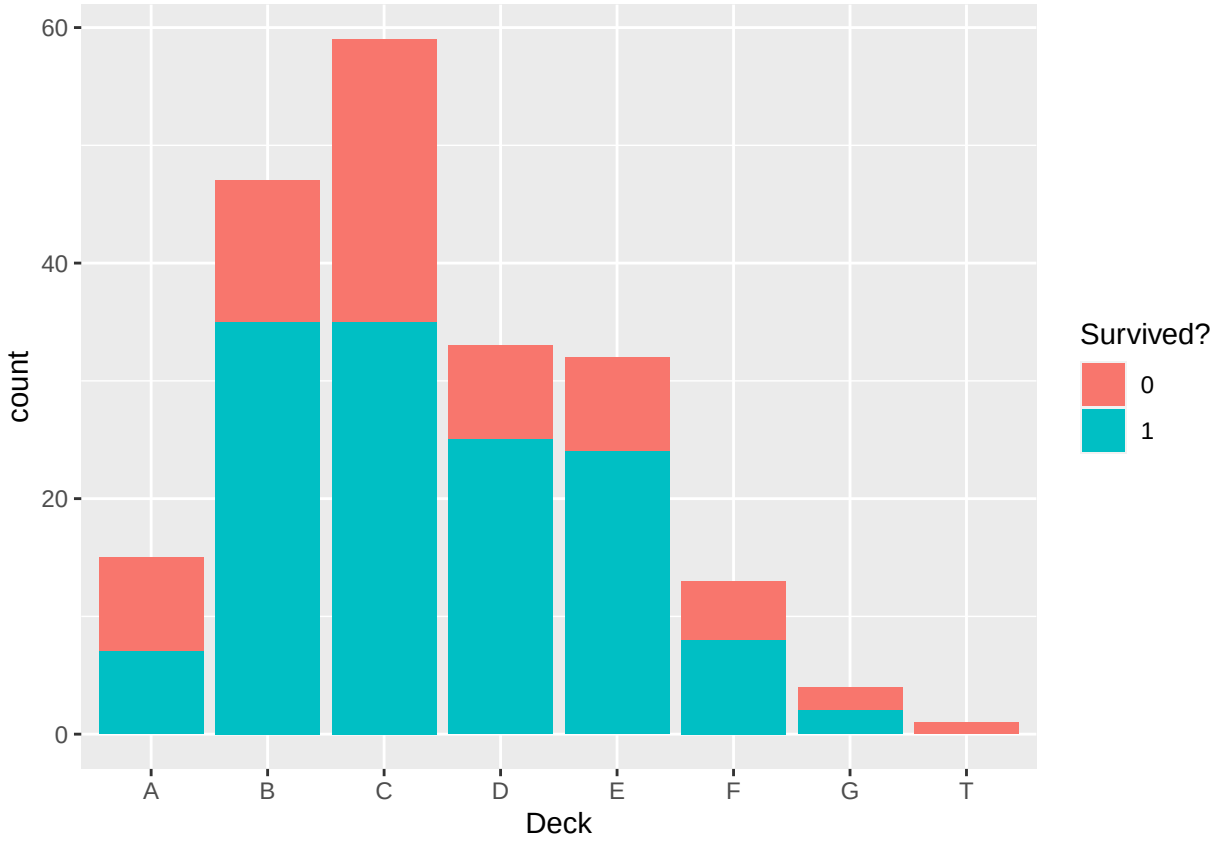
```
full <- full[order(full$PassengerId),]
full$Deck[full$Deck=='']<-NA
full$Deck <- factor(full$Deck)
```

```
full$Deck <- addNA(full$Deck)
ggplot(full[1:891,],aes(x=Deck,fill=factor(Survived)))+geom_bar()+scale_fill_discrete("Survived?")
```



Có vẻ như có rất nhiều giá trị bị thiếu. Hãy xem dữ liệu không có các giá trị bị thiếu này.

```
levels(full$Deck)[9] <- "TT"
train <- full[1:891,] # Look only at Data with known survival
ggplot(train[train$Deck!='TT',],aes(x=Deck,fill=factor(Survived)))+geom_bar()+scale_fill_discrete("Surv
```



Dường như có một số mối tương quan, nhưng với quá nhiều giá trị bị thiếu, sẽ không có ý nghĩa gì để đưa ra kết luận

3. GIÁ TRỊ BỊ MẤT

Trước khi tiếp tục với kỹ thuật tính năng, phải xử lý các giá trị bị thiếu. Thiếu các giá trị trong Age, Fare, Embarked và Deck

3.1 các giá trị bị thiếu trong Fare và Embarked

Khi kiểm tra các giá trị bị thiếu trong cột Fare và thấy rằng hàng 1044 có một Fare bị thiếu. Đây là hành khách từ hạng ba, khởi hành từ cảng S. Chúng ta sẽ cung cấp cho anh ta một Fare tương ứng với Fare trung bình cho trường hợp này.

```
full$Fare[1044] <- median(full[full$Pclass == '3' & full$Embarked == 'S', ]$Fare, na.rm = TRUE)
```

Nhìn vào Embarked, các hàng có số 62 và 830 không có giá trị cho Embarked. Hãy xem những hành khách này đã trả bao nhiêu cho vé của họ và họ sẽ được xếp ở đâu theo class và Fare của họ

```
full$Fare[full$PassengerId==62][1]
```

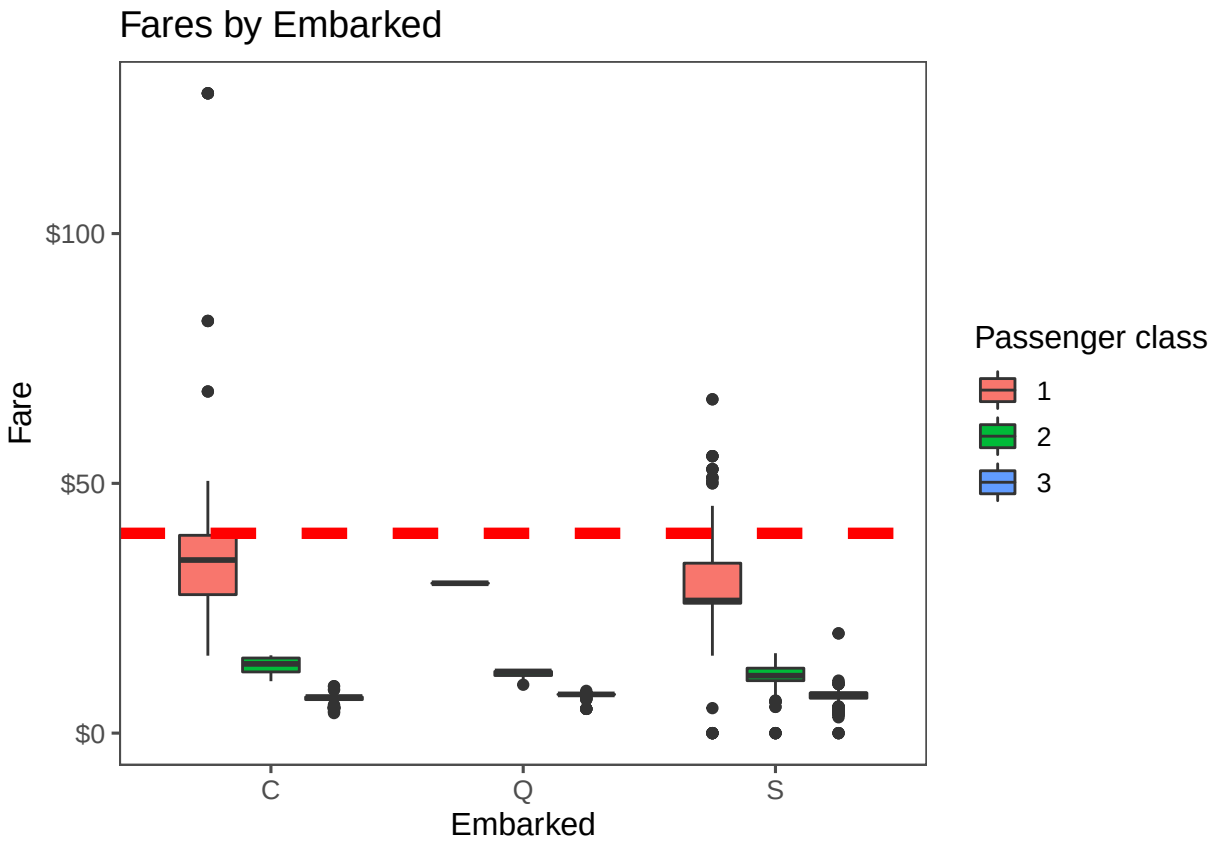
```
## [1] 40
```

```
full$Fare[full$PassengerId==830][1]
```

```
## [1] 40
```

Có vẻ như cả hai hành khách đều trả số tiền như nhau - 40 đô la. Hãy kiểm tra nơi này so với giá vé trung bình cho mỗi cảng.

```
embark_fare <- full %>% filter(PassengerId != 62 & PassengerId != 830)
ggplot(embark_fare, aes(x = Embarked, y = Fare, fill = factor(Pclass))) +
  geom_boxplot() +
  geom_hline(aes(yintercept=40), colour='red', linetype='dashed', lwd=2) +
  scale_y_continuous(labels=dollar_format()) +
  scale_fill_discrete("Passenger class") +
  labs(title= "Fares by Embarked") +
  theme_few()
```



Giá vé mà những hành khách này phải trả gần nhất với mức trung bình của hạng 1 ở cảng C

```
full$Embarked[c(62, 830)] <- 'C'
```

3.2 các giá trị bị thiếu trong Age (tuổi).

có khá nhiều giá trị Age bị thiếu trong dữ liệu. Chúng ta sẽ làm khác đi một chút trong việc áp đặt các giá trị tuổi bị thiếu. Chúng ta sẽ tạo một mô hình dự đoán độ tuổi dựa trên các biến khác.

```
sum(is.na(full$Age))
```

```
## [1] 263
```

Bước đầu tiên là tính toán các biến số và sau đó sử dụng mice để dự đoán Age

```
full$PassengerId <- factor(full$PassengerId)
full$Pclass <- factor(full$Pclass)
full$Sex <- factor(full$Sex)
full$Embarked <- factor(full$Embarked)
full$Title <- factor(full$Title)
full$Surname <- factor(full$Surname)
full$Family <- factor(full$Family)
full$FsizeD <- factor(full$FsizeD)

set.seed(129)
mice_mod <- mice(full[, !names(full) %in% c('PassengerId','Name','Ticket',
            'Cabin','Family','Surname','Survived')], method='rf')
```

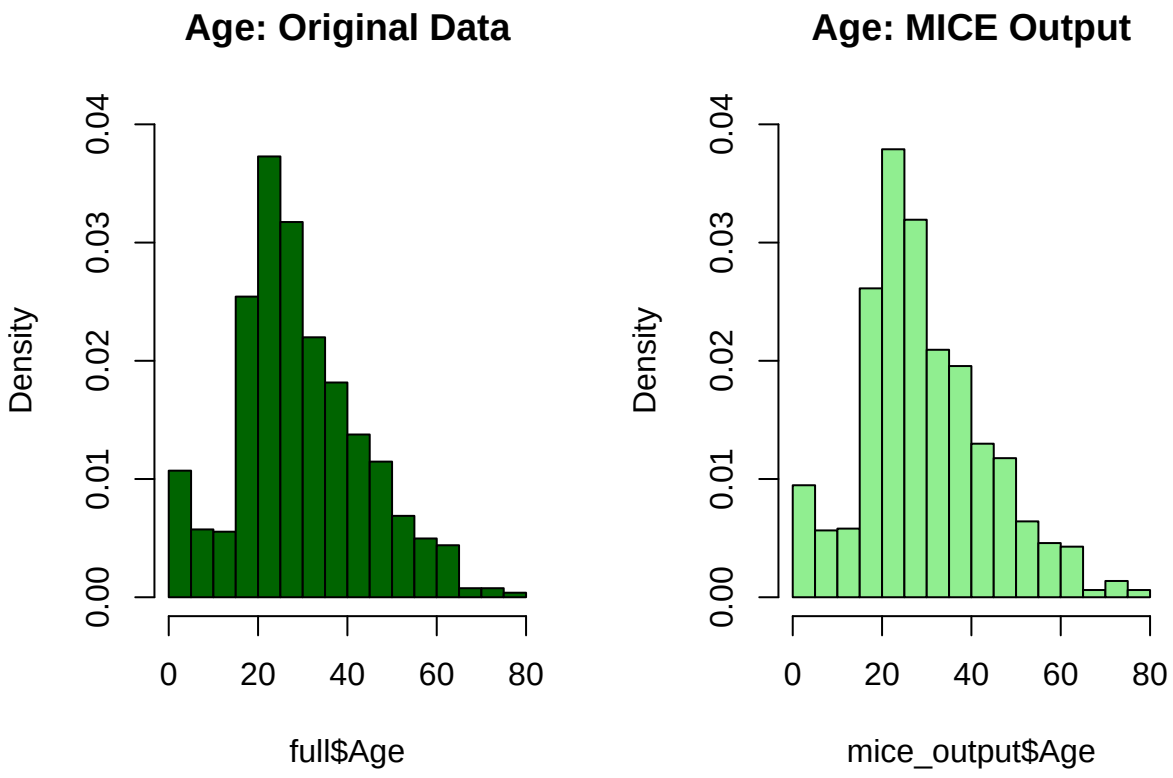
```
##
##  iter imp variable
##   1   1  Age
##   1   2  Age
##   1   3  Age
##   1   4  Age
##   1   5  Age
##   2   1  Age
##   2   2  Age
##   2   3  Age
##   2   4  Age
##   2   5  Age
##   3   1  Age
##   3   2  Age
##   3   3  Age
##   3   4  Age
##   3   5  Age
##   4   1  Age
##   4   2  Age
##   4   3  Age
##   4   4  Age
##   4   5  Age
##   5   1  Age
##   5   2  Age
##   5   3  Age
##   5   4  Age
##   5   5  Age

## Warning: Number of logged events: 25
```

```
mice_output <- complete(mice_mod)
```

Hãy xem độ tuổi được quy định có tuân theo mô hình hiện có không?

```
par(mfrow=c(1,2))
hist(full$Age, freq=F, main='Age: Original Data',
  col='darkgreen', ylim=c(0,0.04))
hist(mice_output$Age, freq=F, main='Age: MICE Output',
  col='lightgreen', ylim=c(0,0.04))
```



gán các giá trị cho những giá trị còn thiếu.

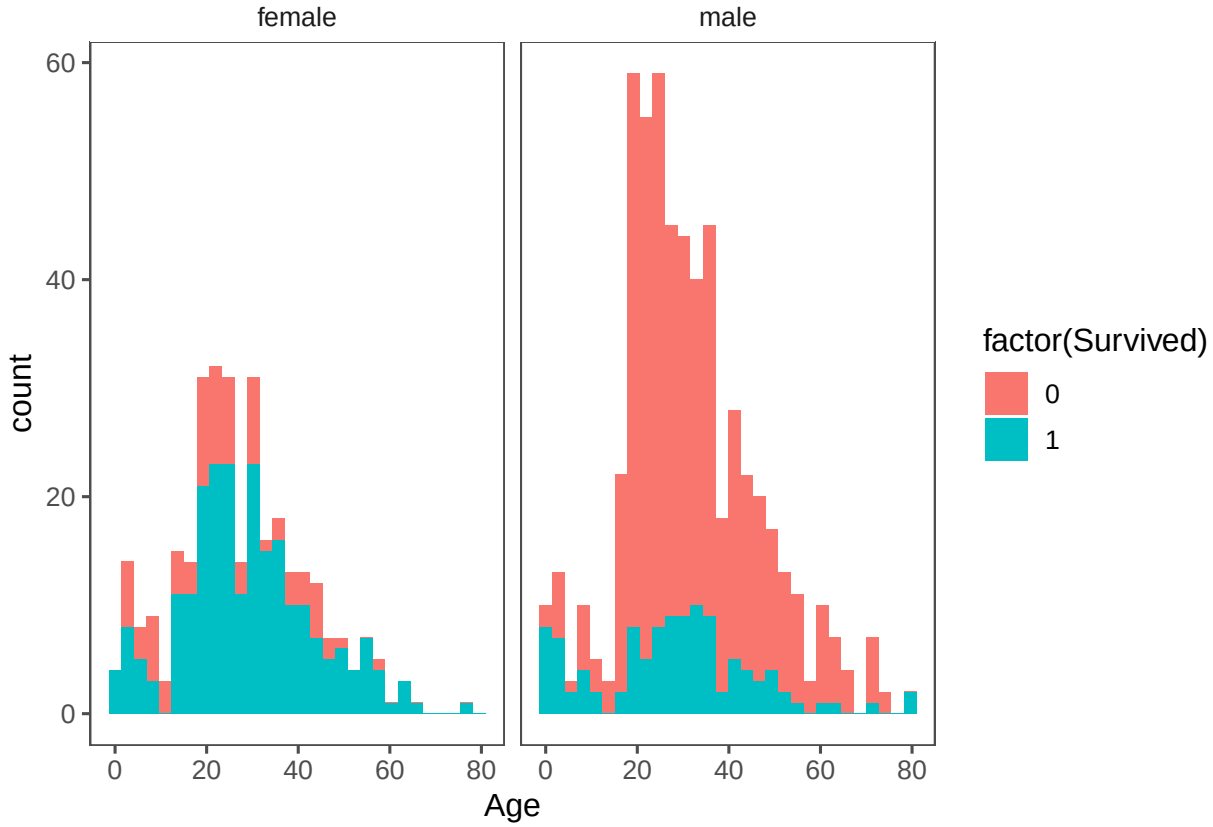
```
full$Age <- mice_output$Age
```

3.3 Feature Engineering vòng 2

Bây giờ chúng ta đã biết tuổi của mọi người, chúng ta có thể tạo một vài biến phụ thuộc độ tuổi mới: Child và Mother. Một đứa trẻ sẽ chỉ đơn giản là người dưới 18 tuổi và một người mẹ là hành khách là 1) nữ, 2) trên 18 tuổi, 3) có nhiều hơn 0 trẻ em (không đùa!), Và 4) không có danh hiệu 'Miss'. Trước hết, hãy xem liệu có mối quan hệ giữa Age, Survived và Sex hay không.

```
ggplot(full[1:891,], aes(Age, fill = factor(Survived))) +
  geom_histogram() +
  facet_grid(.~Sex) +
  theme_few()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Nếu là phụ nữ và trẻ em sẽ có cơ hội sống sót cao hơn so với nam giới

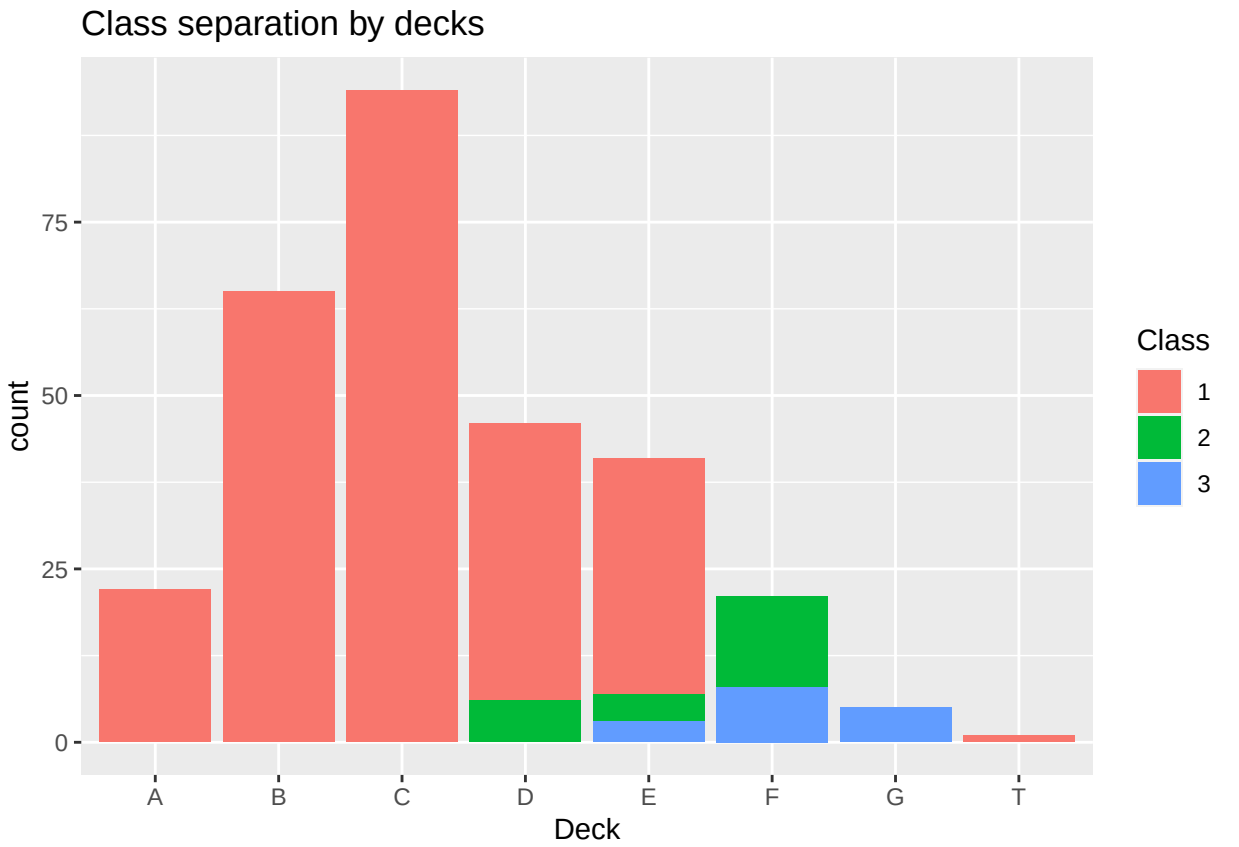
```
full$Child[full$Age < 18] <- 'Child'
full$Child[full$Age >= 18] <- 'Adult'
full$Mother <- 'Not Mother'
full$Mother[full$Sex == 'female' & full$Parch > 0 & full$Age > 18 & full$Title != 'Miss'] <- 'Mother'
full$Child  <- factor(full$Child)
full$Mother <- factor(full$Mother)
```

4.FEATURE ENGINEERING VÒNG 3

tìm hiểu thêm về các giá trị bị thiếu của Desk. Có mối liên hệ nào giữa Giới tính, Tuổi tác không?

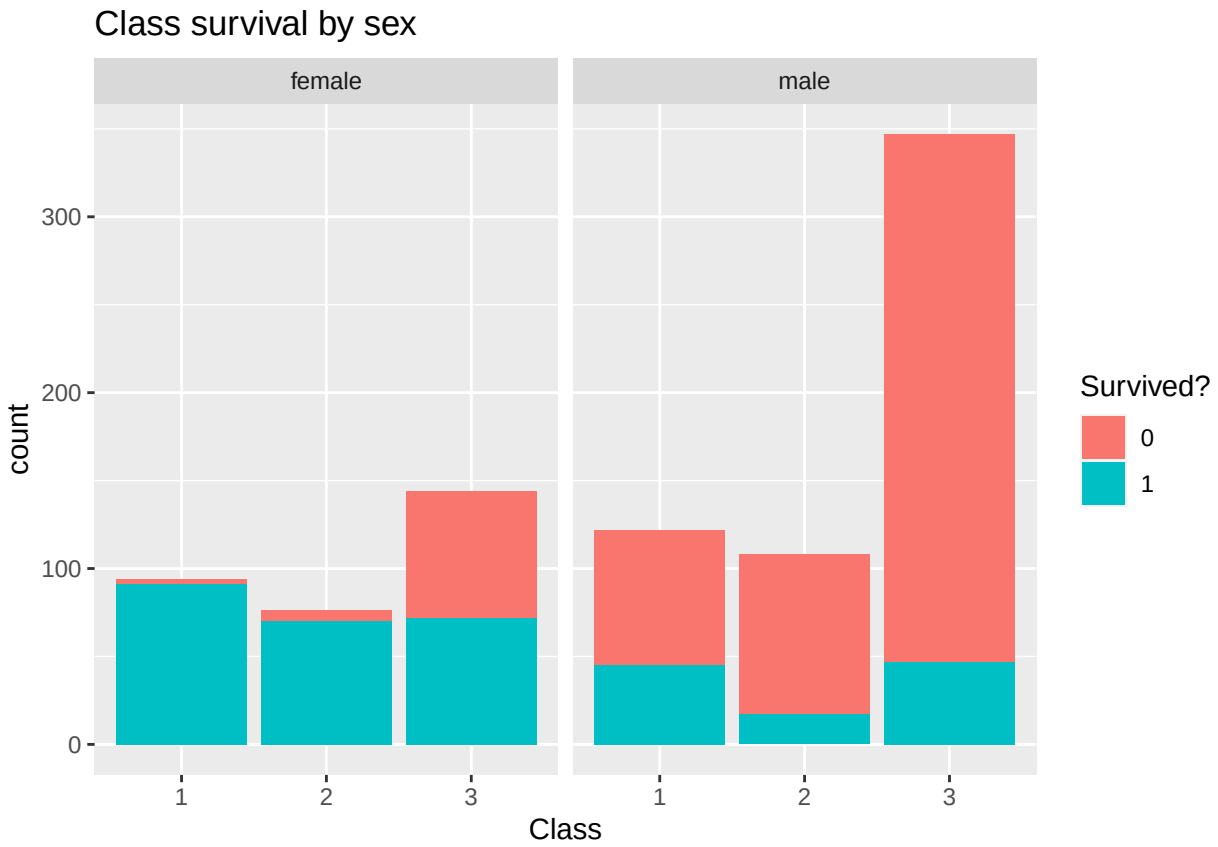
4.1 Class có phải là một yếu tố quan trọng cho sự sống còn?

```
ggplot(full[full$Deck!='TT',],aes(x=factor(Deck),fill=factor(Pclass))) +
        geom_bar() +
        scale_fill_discrete("Class") +
        labs(title= "Class separation by decks",x = 'Deck')
```



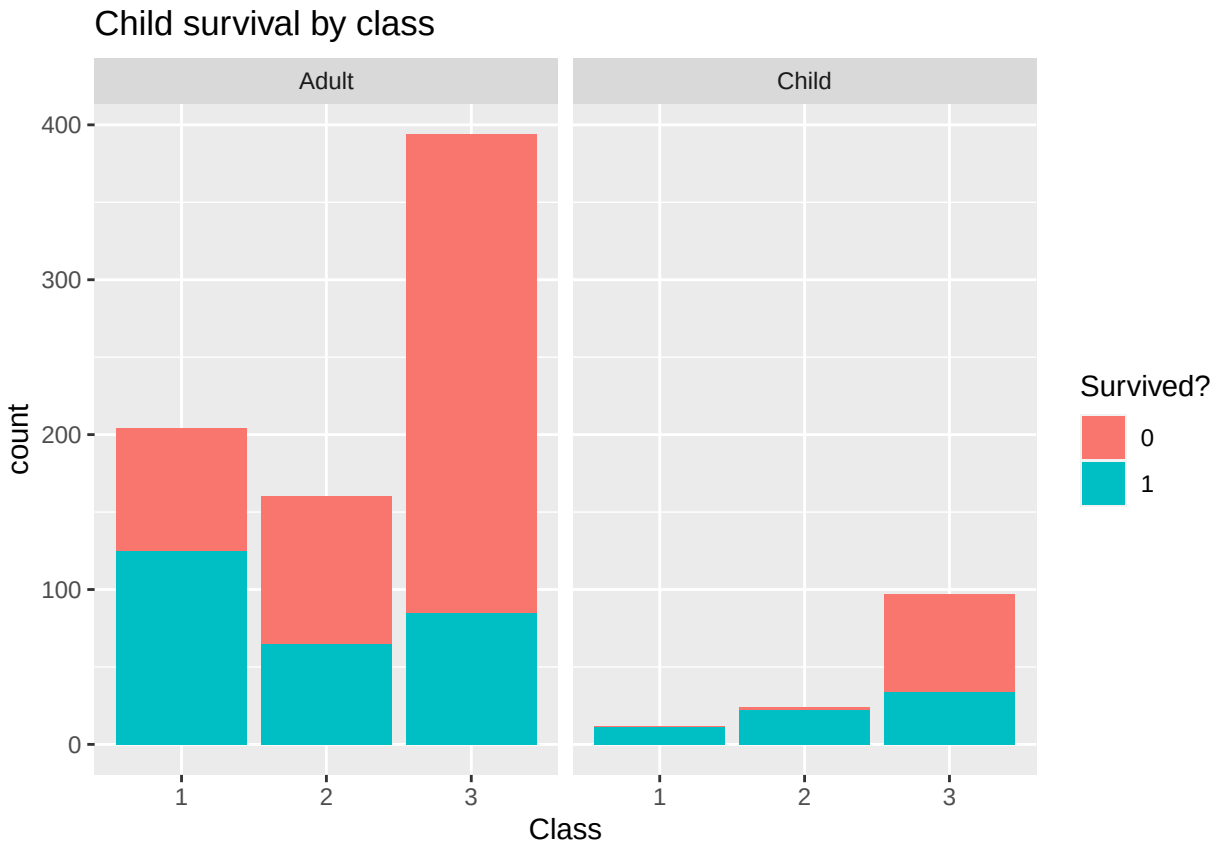
Lớp 1 được xếp trên các sàn từ A đến E, Lớp 2 được xếp trên các sàn D, E, F và Lớp 3 được xếp trên các sàn E, F, G. Bộ bài T đã được sinh sống bởi một nhóm nhỏ từ Lớp 1. số lượng giá trị bị thiếu trong cột Desk sẽ khiến mọi giả định dễ bị bác bỏ. Tuy nhiên, chúng ta biết chắc chắn rằng những người từ lớp 3 đã ở phần dưới của con tàu. bây giờ kiểm tra xem khả năng sống sót của họ có phần nào phụ thuộc vào class và sex của họ hay không.

```
ggplot(full[1:891,],aes(x=factor(Pclass),fill=factor(Survived))) +
        geom_bar() +
        scale_fill_discrete("Survived?") +
        labs(title= "Class survival by sex",x = 'Class') +
        facet_grid(.~Sex)
```



Chúng ta biết rằng phụ nữ có cơ hội sống sót cao hơn, nhưng phụ nữ thuộc Nhóm 3 không có tỷ lệ sống sót cao.

```
ggplot(full[1:891,],aes(x=factor(Pclass),fill=factor(Survived))) +
        geom_bar() +
        scale_fill_discrete("Survived?") +
        labs(title= "Child survival by class",x = 'Class') +
        facet_grid(.~Child)
```

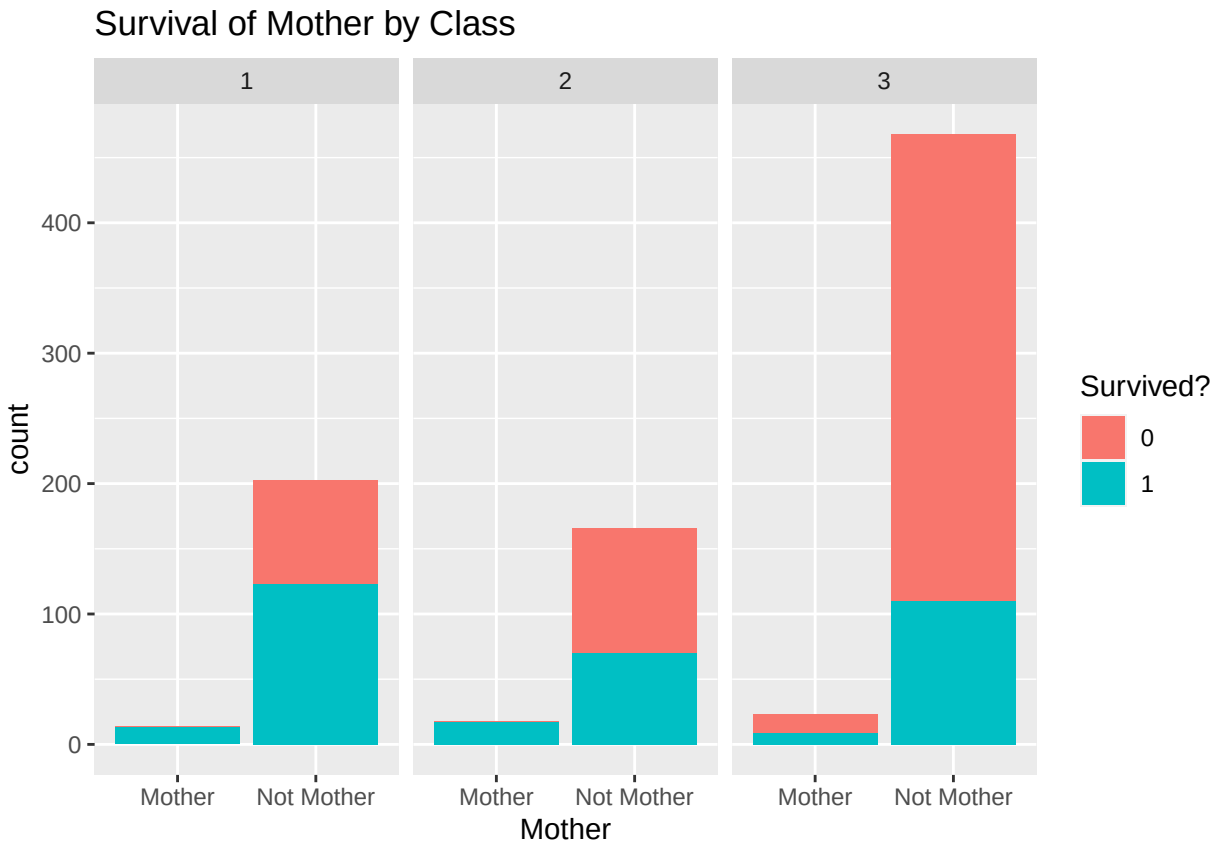


Từ 2 biểu đồ cuối cùng, người ta có thể dễ dàng nhận thấy rằng nếu là phụ nữ hoặc trẻ em từ lớp 1 và lớp 2, có cơ hội sống sót rất cao! Phụ nữ hoặc trẻ em từ lớp 3 có cơ hội sống sót ngang bằng với nam giới.

```
full$ChildFrom12 <- 'Not from'
full$ChildFrom12[full$Child=='Child'&full$Pclass==1] <- 'From'
full$ChildFrom12[full$Child=='Child'&full$Pclass==2] <- 'From'
full$ChildFrom12 <- factor(full$ChildFrom12)
full$FemaleFrom12 <- 'Not from'
full$FemaleFrom12[full$Sex=='female'&full$Pclass==1] <- 'From'
full$FemaleFrom12[full$Sex=='female'&full$Pclass==2] <- 'From'
full$FemaleFrom12 <- factor(full$FemaleFrom12)
```

chúng ta sẽ tìm thấy sự class survival đối với những người phụ nữ là Mẹ

```
ggplot(full[1:891,],aes(x=Mother,fill=factor(Survived))) +
        geom_bar() +
        scale_fill_discrete("Survived?") +
        labs(title= "Survival of Mother by Class",x = 'Mother') +
            facet_grid(.~Pclass)
```



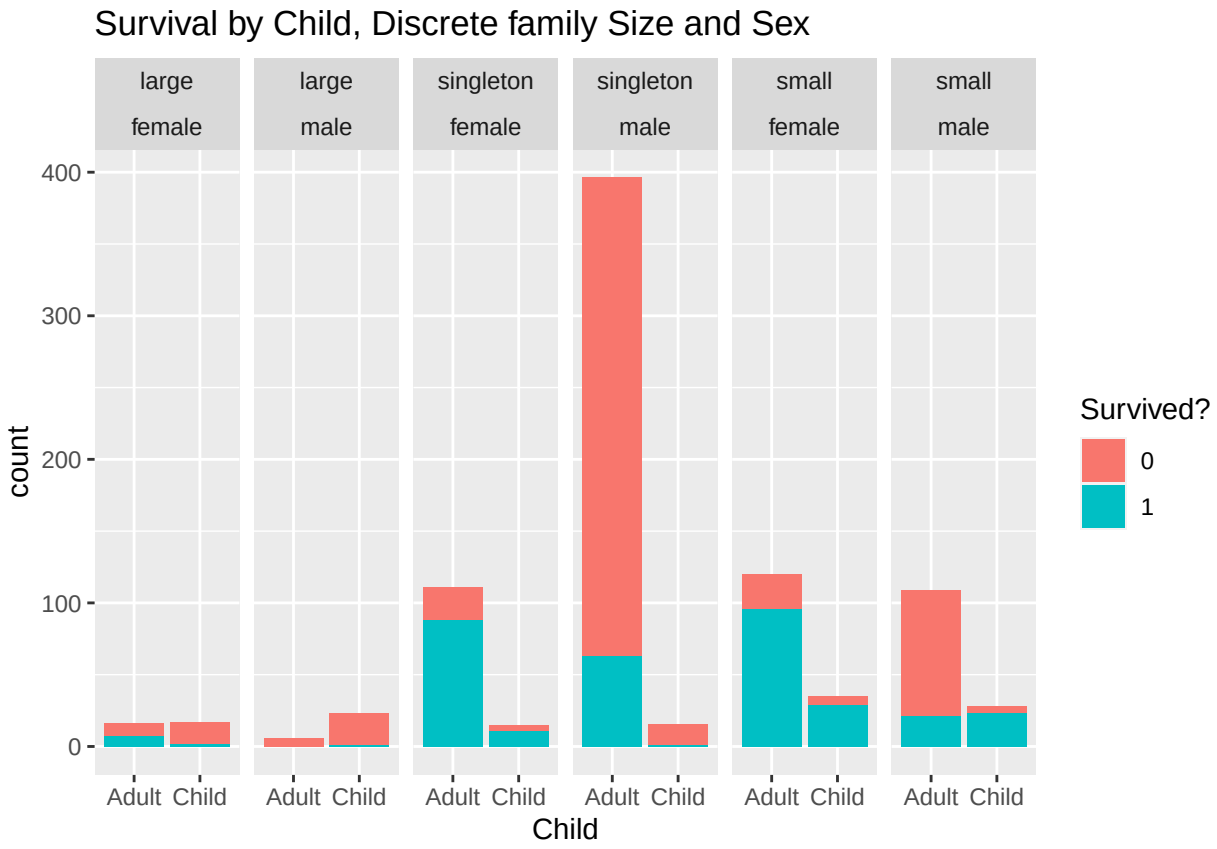
có một mối quan hệ chặt chẽ với sự sống còn tùy thuộc vào class nếu chúng ta có thể có nhiều giá trị Desk hơn để có thể thông báo rằng những người ở Desk thấp hơn tỉ lệ sống sót không cao.

```
full$MotherFrom12 <- 'Not from'
full$MotherFrom12[full$Mother=='Mother'&full$Pclass==1] <- 'From'
full$MotherFrom12[full$Mother=='Mother'&full$Pclass==2] <- 'From'
full$MotherFrom12 <- factor(full$MotherFrom12)
```

4.2 quy mô của gia đình

có mối quan hệ giữa quy mô gia đình, con cái và giới tính

```
ggplot(full[1:891,],aes(x=factor(Child),fill=factor(Survived))) +
        geom_bar() +
        scale_fill_discrete("Survived?") +
        labs(title= "Survival by Child, Discrete family Size and Sex",x = 'Child') +
        facet_grid(.~FsizeD+Sex)
```



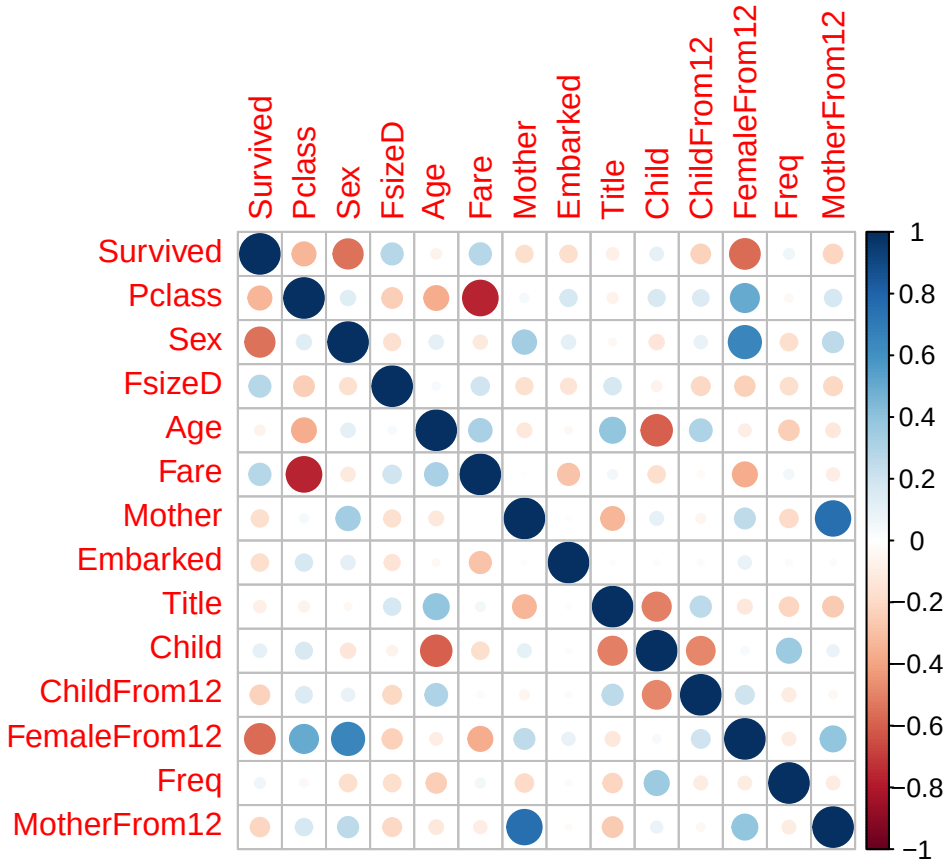

```
full$ChildSaved <- 'Not saved'
full$ChildSaved[full$Child=='Child'&full$Sex=='female'&full$FsizeD!='large'] <- 'Saved'
full$ChildSaved[full$Child=='Child'&full$Sex=='male'&full$FsizeD=='small'] <- 'Saved'
full$ChildSaved <- factor(full$ChildSaved)
```

4.3 Tính năng tương quan

hình dung mối tương quan giữa các đối tượng để có một số thông tin chi tiết về các đối tượng đủ mạnh cho mô hình dự đoán.

```
corr_data <- full[1:891,]
## chuyển đổi sang kiểu số
corr_data$Embarked <- revalue(corr_data$Embarked,
            c("S" = 1, "Q" = 2, "C" = 3))
corr_data$Sex <- revalue(corr_data$Sex,
            c("male" = 1, "female" = 2))
corr_data$Title <- revalue(corr_data$Title,
            c("Mr" = 1, "Master" = 2,"Crew" = 3,
            "Mrs" = 4,"Member" = 5,"Miss" = 6, "Rare Title" = 7))
corr_data$FsizeD <- revalue(corr_data$FsizeD,
            c("small" = 1, "singleton" = 2, "large" = 3))
corr_data$Child <- revalue(corr_data$Child,
            c("Adult" = 1, "Child" = 2))
corr_data$Mother <- revalue(corr_data$Mother,
            c("Mother" = 1, "Not Mother" = 2))
corr_data$Mother <- as.numeric(corr_data$Mother)
```

```
corr_data$FsizeD <- as.numeric(corr_data$FsizeD)
corr_data$Child <- as.numeric(corr_data$Child)
corr_data$Sex <- as.numeric(corr_data$Sex)
corr_data$Embarked <- as.numeric(corr_data$Embarked)
corr_data$Title <- as.numeric(corr_data$Title)
corr_data$Pclass <- as.numeric(corr_data$Pclass)
corr_data$Survived <- as.numeric(corr_data$Survived)
corr_data$Freq <- as.numeric(corr_data$Freq)
corr_data$Age <- as.numeric(corr_data$Age)
corr_data$ChildFrom12 <- as.numeric(revalue(corr_data$ChildFrom12,
            c("From"=1,"Not from" = 2)))
corr_data$FemaleFrom12 <- as.numeric(revalue(corr_data$FemaleFrom12,
            c("From"=1,"Not from" = 2)))
corr_data$MotherFrom12 <- as.numeric(revalue(corr_data$MotherFrom12,
            c("From"=1,"Not from" = 2)))
corr_data <-corr_data[,c("Survived", "Pclass", "Sex",
            "FsizeD", "Age", "Fare", "Mother",
            "Embarked","Title","Child","ChildFrom12",
            "FemaleFrom12","Freq","MotherFrom12")]
mcorr_data <- cor(corr_data)
corrplot(mcorr_data,method="circle")
```



5. PREDICTION

Cuối cùng, chúng ta dự đoán xem ai sống sót trong số các hành khách trên tàu Titanic dựa trên các biến số đã cẩn thận sắp xếp và xử lý các giá trị còn thiếu. Đối với điều này, nhóm sẽ dựa vào thuật toán phân loại Logistic Regression,randomForest,Classification tree,Support Vector Machine,Linear Discriminant

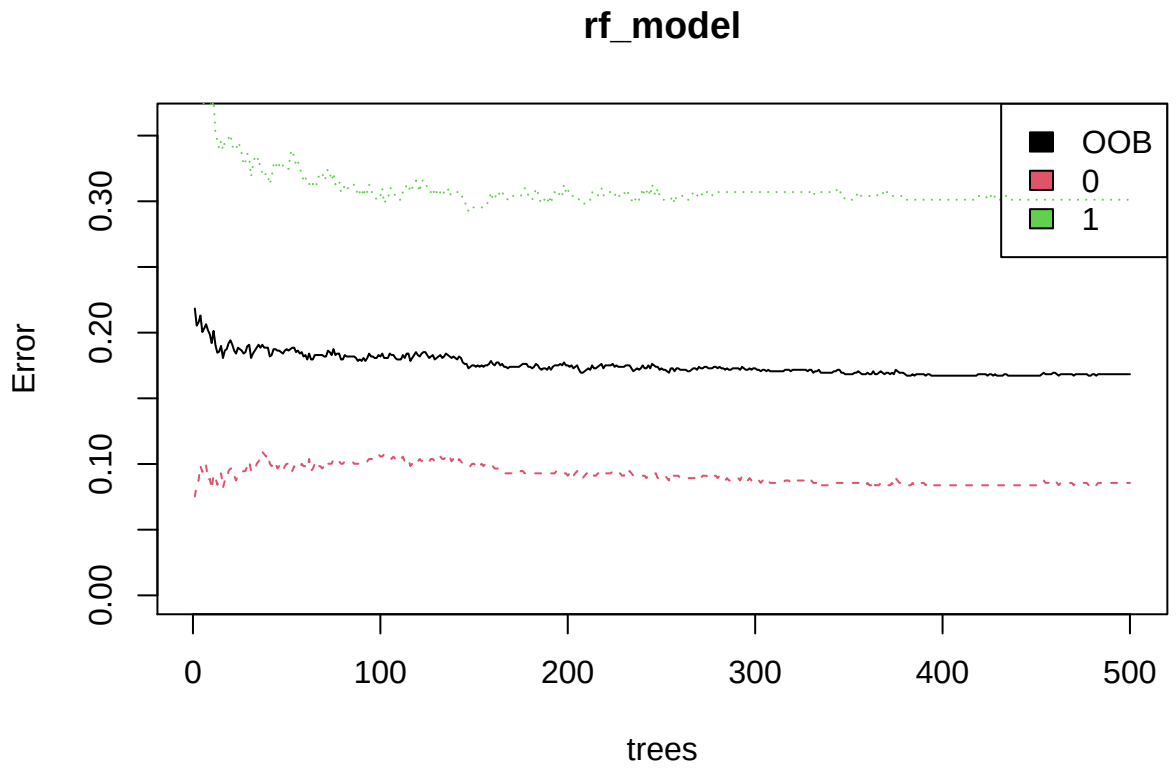
5.1 Xây dựng mô hình

tách tệp gốc thành dữ liệu đào tạo và kiểm tra.

```
train <- full[1:891,]
test <- full[892:1309,]
```

sử dụng randomForest trên tập huấn luyện. nhóm sẽ không sử dụng Tuổi, Desk vì số lượng giá trị bị thiếu. Các tham số đã chọn hoạt động tốt và đạt độ chính xác của mô hình 83,16%

```
set.seed(156)
rf_model <- randomForest(factor(Survived) ~ Pclass + Sex + Fare + Embarked + Title +
            FsizeD + Freq + ChildSaved + FemaleFrom12 + ChildFrom12, data = train, trees=500)
plot(rf_model, ylim=c(0,0.36))
legend('topright', colnames(rf_model$err.rate), col=1:3, fill=1:3)
```




```
rf.fitted = predict(rf_model)
ans_rf = rep(NA,891)
for(i in 1:891){
  ans_rf[i] = as.integer(rf.fitted[[i]]) -1
}
# Result
table(ans_rf)
```

```
## ans_rf
```

```
##   0   1
## 605 286
```

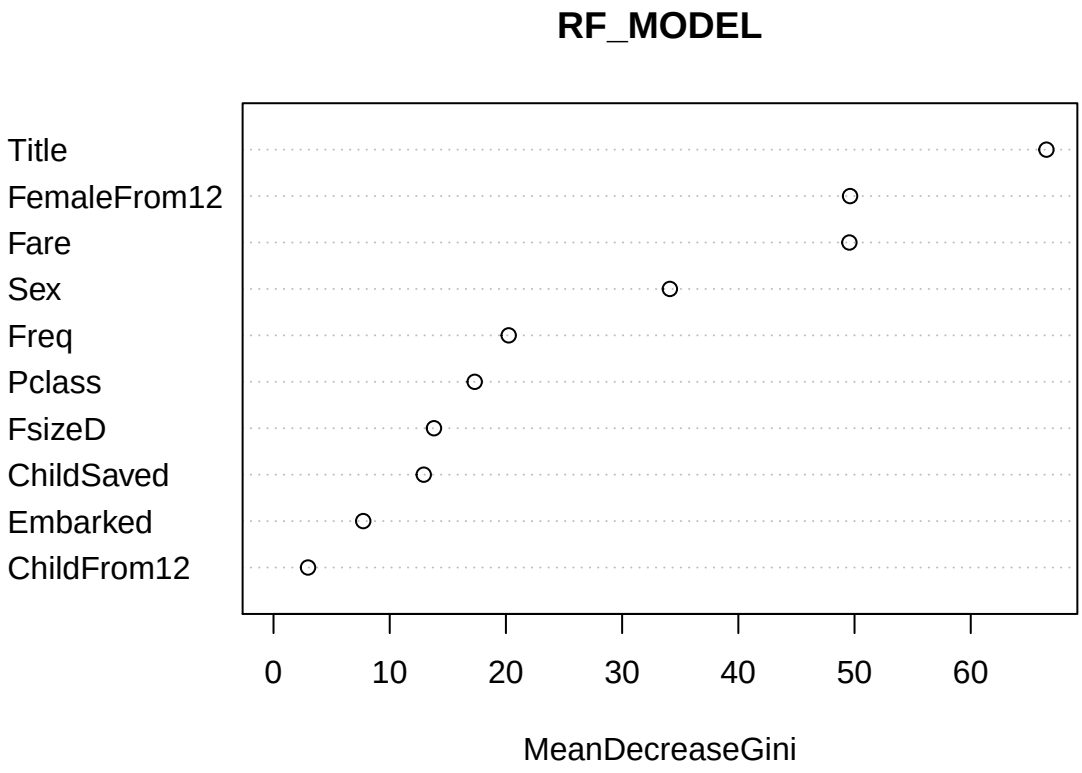
```
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = factor(Survived) ~ Pclass + Sex + Fare +      Embarked + Title + FsizeD + Fr
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 16.84%
## Confusion matrix:
##     0   1 class.error
## 0 502  47   0.0856102
## 1 103 239   0.3011696
```

```
mean(ans_rf == train$Survived)
```

```
## [1] 0.8316498
```

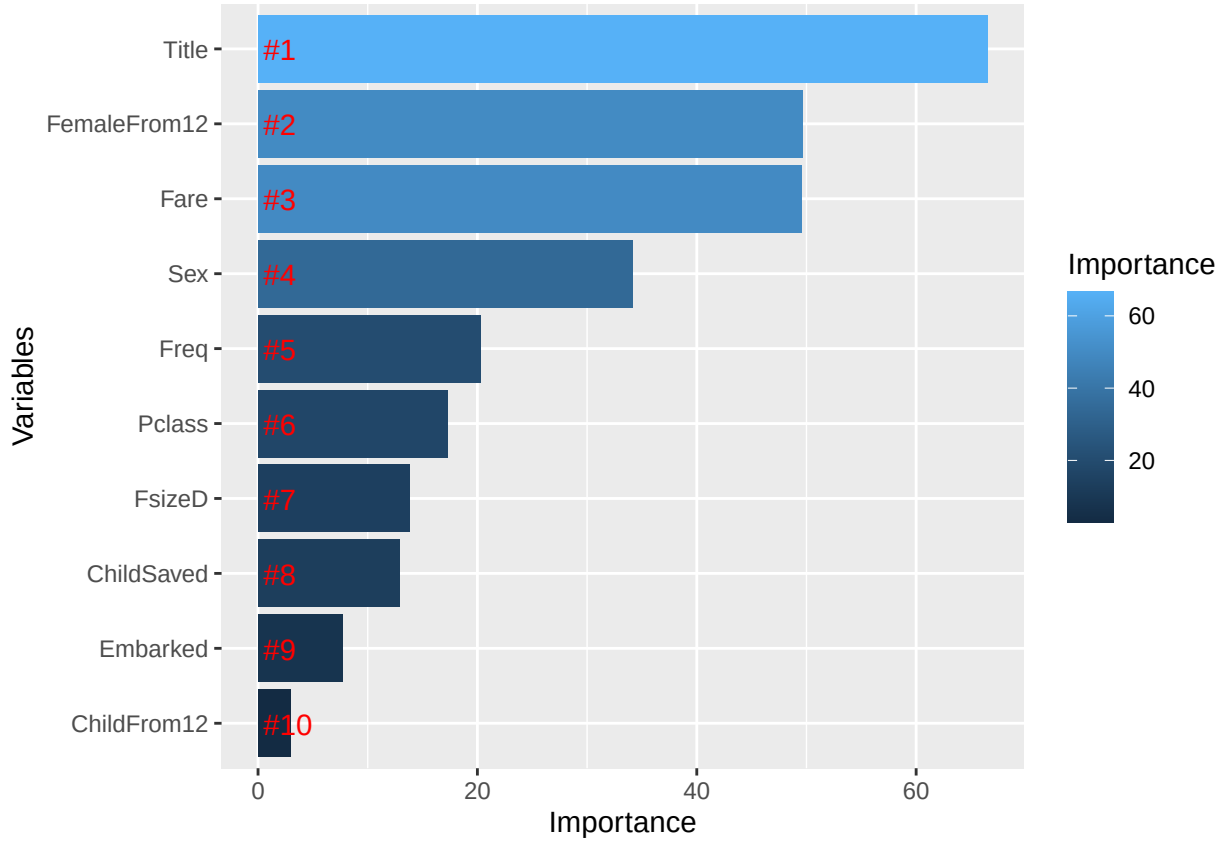
```
varImpPlot(rf_model, main = "RF_MODEL")
```



```
# Var importancies
importance    <- importance(rf_model)
varImportance <- data.frame(Variables = row.names(importance),
                            Importance = round(importance[ ,'MeanDecreaseGini'],2))

# var imp
rankImportance <- varImportance %>%
  mutate(Rank = paste0('#',dense_rank(desc(Importance))))

# Graph importancies
ggplot(rankImportance, aes(x = reorder(Variables, Importance),
                           y = Importance, fill = Importance)) +
  geom_bar(stat='identity') +
  geom_text(aes(x = Variables, y = 0.5, label = Rank),
            hjust=0, vjust=0.55, size = 4, colour = 'red') +
  labs(x = 'Variables') +
  coord_flip()
```



sử dụng Logistic Regression trên tập huấn luyện. nhóm vẫn sẽ không sử dụng Tuổi, Desk vì số lượng giá trị bị thiếu. Các tham số đã chọn hoạt động tốt và đạt độ chính xác của mô hình 76,87%. Độ chính xác thấp hơn mô hình RandomForest

```
glm_model1 <- glm(factor(Survived) ~ 1, data = train, family = 'binomial')
glm_model2 <- glm(factor(Survived) ~ ., data = train, family = 'binomial')
```

```
## Warning: glm.fit: algorithm did not converge
```

```r
glm_model <- step(glm_model1, scope = list(lower = glm_model1, upper = glm_model2 ),
                  direction = "both")
```

```
## Start:  AIC=1188.66
## factor(Survived) ~ 1

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, : using
## the 204/891 rows from a combined fit

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, :
## glm.fit: algorithm did not converge

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, :
## glm.fit: algorithm did not converge

##                 Df Deviance    AIC
## + FemaleFrom12   1  187.472 1118.4
## + Title          7  176.781 1119.7
## + Sex            1  190.637 1121.6
## + Age            1  249.826 1180.8
## + ChildFrom12    1  252.373 1183.3
## + Child          1  253.853 1184.8
## + ChildSaved     1  253.853 1184.8
## + Freq           1  254.222 1185.2
## + IsAlone        1  254.477 1185.4
## + MotherFrom12   1  254.561 1185.5
## + Mother         1  255.743 1186.7
## + SibSp          1  255.872 1186.8
## + FsizeD         2  254.325 1187.3
## + Embarked       2  255.015 1188.0
## + Fsize          1  257.435 1188.4
## <none>              259.698 1188.7
## + Fare           1  258.269 1189.2
## + Pclass         2  256.590 1189.5
## + Parch          1  259.397 1190.3
## + Deck           7  249.270 1192.2
## + Ticket       141   50.640 1261.6
## + Cabin        146   53.046 1274.0
## + Surname      157   53.413 1296.4
## + Family       163   43.002 1298.0
## + PassengerId  203    0.000 1335.0
## + Name         203    0.000 1335.0
##
## Step:  AIC=886.94
## factor(Survived) ~ FemaleFrom12

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, : using
## the 204/891 rows from a combined fit

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, :
## glm.fit: algorithm did not converge

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, :
## glm.fit: algorithm did not converge
```

```
##                 Df Deviance     AIC
## + Age            1   179.51  880.98
## + Child          1   180.60  882.06
## + ChildSaved     1   180.60  882.06
## + ChildFrom12    1   180.96  882.43
## + Title          7   171.75  885.22
## + IsAlone        1   185.38  886.85
## <none>               882.94  886.94
## + Sex            1   186.08  887.55
## + FsizeD         2   184.23  887.70
## + SibSp          1   186.39  887.86
## + Embarked       2   184.91  888.38
## + Fsize          1   186.92  888.39
## + MotherFrom12   1   187.14  888.60
## + Fare           1   187.14  888.61
## + Mother         1   187.41  888.88
## + Parch          1   187.42  888.89
## + Freq           1   187.44  888.91
## + Pclass         2   186.78  890.25
## + Deck           7   181.75  895.22
## + Ticket       141    23.87 1005.33
## + Cabin        146    29.15 1020.62
## + Surname      157    24.14 1037.61
## + Family       163    22.96 1048.43
## + PassengerId  202     0.00 1103.47
## + Name         202     0.00 1103.47
## - FemaleFrom12   1  1186.66 1188.66
##
## Step:  AIC=871.84
## factor(Survived) ~ FemaleFrom12 + Age

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, : using
## the 204/891 rows from a combined fit

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, :
## glm.fit: algorithm did not converge

## Warning in add1.glm(fit, scope$add, scale = scale, trace = trace, k = k, :
## glm.fit: algorithm did not converge

##                 Df Deviance     AIC
## <none>               865.84  871.84
## + ChildFrom12    1   177.54  871.87
## + Child          1   178.03  872.35
## + ChildSaved     1   178.03  872.35
## + Fare           1   178.03  872.36
## + Embarked       2   176.58  872.91
## + Freq           1   178.95  873.28
## + IsAlone        1   179.01  873.34
## + Parch          1   179.19  873.52
## + SibSp          1   179.23  873.55
## + Sex            1   179.32  873.64
## + MotherFrom12   1   179.41  873.74
## + Mother         1   179.48  873.81
```

```
## + Fsize          1   179.51  873.83
## + FsizeD         2   177.85  874.18
## + Pclass         2   179.30  875.62
## + Title          7   170.25  876.58
## + Deck           7   172.82  879.15
## - Age            1   882.94  886.94
## + Ticket       141    16.27  990.60
## + Cabin        146    25.27 1009.60
## + Surname      157    17.36 1023.69
## + Family       163    16.05 1034.38
## + PassengerId  201     0.00 1094.33
## + Name         201     0.00 1094.33
## - FemaleFrom12   1  1183.31 1187.31
```

```
glm.fitted = predict(glm_model)
ans_glm = rep(NA,length(glm.fitted))
for(i in 1:length(glm.fitted)){
  ans_glm[i] = as.integer(glm.fitted[[i]])
}
# Result
table(ans_glm)
```

```
## ans_glm
##  -2  -1   0   1   2   3
##  15 416 290   1  88  81
```

```
mean(ans_glm != train$Survived)
```

```
## [1] 0.7687991
```

Với thuật toán Classification tree,độ chính xác của mô hình khá thấp chỉ 61,61%

```
dt_model <- rpart(factor(Survived) ~ Pclass + Sex + Fare + Embarked + Title +
            FsizeD + Freq + ChildSaved + FemaleFrom12 + ChildFrom12, data = train, method = "class")

dt.fitted = predict(dt_model)
ans_dt = rep(NA,length(dt.fitted))
for(i in 1:length(dt.fitted)){
  ans_dt[i] = as.integer(dt.fitted[[i]])
}
# Result
table(ans_dt)
```

```
## ans_dt
##    0
## 1782
```

```
mean(ans_dt == train$Survived)
```

```
## [1] 0.6161616
```

Support Vector Machine là mô hình cho độ chính xác khá cao 81%. Nhưng vẫn thấp hơn Random Forest là 83,16%

```
new_train <- full[which(is.na(full$Survived) ==FALSE),]
svm_model <- svm(factor(Survived) ~ Pclass + Sex + Fare + Embarked + Title +
            FsizeD + Freq + ChildSaved + FemaleFrom12 + ChildFrom12, data = train )

svm.fitted = predict(svm_model,new_train[,-1])
ans_svm = rep(NA,length(svm.fitted))
for(i in 1:length(svm.fitted)){
  ans_svm[i] = as.integer(svm.fitted[[i]])
}
# Result
table(ans_svm)
```

```
## ans_svm
##   1   2
##  98 106
```

```
mean(ans_svm != train$Survived)
```

```
## Warning in ans_svm != train$Survived: longer object length is not a multiple of
## shorter object length
```

```
## [1] 0.8103255
```

Mô hình Linear Discriminent cho ra độ chính xác trung bình là 79,34%.

```
lda_model <- MASS::lda(factor(Survived) ~ Pclass + Sex + Fare + Embarked + Title +
            FsizeD + Freq + ChildSaved + FemaleFrom12 + ChildFrom12, data = train)

lda.fitted = predict(lda_model)
ans_lda = rep(NA,length(lda.fitted))
for(i in 1:length(lda.fitted)){
  ans_lda[i] = as.integer(lda.fitted[[i]]) -1
}
```

```
## Warning in ans_lda[i] <- as.integer(lda.fitted[[i]]) - 1: number of items to
## replace is not a multiple of replacement length

## Warning in ans_lda[i] <- as.integer(lda.fitted[[i]]) - 1: number of items to
## replace is not a multiple of replacement length

## Warning in ans_lda[i] <- as.integer(lda.fitted[[i]]) - 1: number of items to
## replace is not a multiple of replacement length
```

```
# Result
table(ans_lda)
```

```
## ans_lda
## -2 -1  0
##  1  1  1
```

```r
mean(ans_lda != train$Survived)
```

```
## [1] 0.7934905
```

Vậy trong năm thuật toán phân loại Logistic Regression,randomForest,Classification tree,Support Vector Machine,Linear Discriminent. Thì Randomforest cho ra độ chính xác cao nhất là 83,16%

5.2 dự đoán

```r
# Dự đoán bằng cách sử dụng test
prediction <- predict(rf_model, test)

solution <- data.frame(PassengerID = test$PassengerId, Survived = prediction)

# Viết vào file
write.csv(solution, file = 'C:/Users/uongt/Downloads/titanic/LRpredict.csv', row.names = F)
```

6. KẾT LUẬN

Tất cả các mô hình đều cho độ chính xác khá gần nhau. Tuy nhiên,có hai mô hình có độ chính xác rõ ràng cho tập dữ liệu titanic. Mô hình Support Vector Machine (SVM) và mô hình RandomForest cho độ chính xác tốt nhất.
Tuy nhiên, nhóm vẫn chưa nhận được độ chính xác từ 84% trở lên. và độ chính xác được Kaggle xác nhận chỉ là 78,23% - ở mức trung bình.