Kaggle Titanic Survivor Prediction: Comparison of Machine Learning Methods

Jim Nelson

Friday, December 11, 2015

OBJECTIVES

1. Demonstrate proficiency in R.
2. Demonstrate ability to perform appropriate data transformations and creative feature engineering.
3. Compare performace of several R Machine Learning packages.
4. Submit models to Kaggle to assess performance and obtain challenge ranking.

INTRODUCTION

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.
One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

The goal of this study was to utilize the Kaggle Titanic Challenge dataset to perform and compare several machine learning predict analytic methods to predict which passengers survived the tragedy.

METHODS

Kaggle Titanic Competition

The crowdsourcing predicitve modeling competition website Kaggle was used as a platform for this study, providing a means to assess my skills compared to other participants. As of 12/11/15 there were 3946 participants with 2458 scripts. A good description of how Kaggle works is found here. The Titanic competitionis an active "Getting Started"competition in the "Knowledge" category which has been ongoing since September 2012. Kaggle also makes available a forum for competition discussionand a repository for scripts and notebooks used in the competition. All Kaggle rules and instructions were followed during this study.

Datasets

Datasets used in this study were obtained from the Kaggle website as CSV files. The "training" dataset consisted of 891 passengers with the following 14 variables: *PassengerId,Survived,Pclass,Name,Sex,Age, SibSp,Parch,Ticket,Fare,Cabin,Embarked.* The "test"dataset contained 418 passengers and the same variables with the exception of the*Survived* variable. A complete description of the variables is shown here. The two datasets were combined to facilitate data curation then seperated for predicitve modeling use.

Software and Computing Environment

The study was performed locally on a HP Pavilion 23 All-in-One with a 64 Bit OS running Windows 10 with 4.0 GB Ram and an AMD AP-5300 APU processor with Radeon HD graphics. The study was performed in R (R version 3.1.3 (2015-03-09) – "Smooth Sidewalk" Platform: x86_64-w64-mingw32/x64 (64-bit))Copyright (C) 2015 The R Foundation for Statistical Computing) using the open source R platform R Studio (Version 0.98.1102 - © 2009-2014 RStudio, Inc.)

The following R software packages with corresponding manuals and vignettes were obtained from the The Comprehensive R Archive Network (CRAN):

Data Manipulation and Visualization:
dplyr: A Grammar of Data Manipulation (v.0.4.3);
ggplot2: An Implementation of the Grammar of Graphics (v.1.01)

Logistic Regression Analysis:
glm2: Fitting Generalized Linear Models (v. 1.1.2)

Classification statistics and AUROC analysis:
caret: Classification and Regression Training (v.6.0-62); *pROC*: Display and Analyze ROC Curves (v. 1.8)

Recursive Partitioning Modeling:
rpart: Recursive Partitioning and Regression Trees (v.4.1-10);
rattle: Graphical User Interface for Data Mining in R (v.4.0.5);
rpart.plot: Plot 'rpart' Models: An Enhanced Version of 'plot.rpart'(v.1.5.3);
RColorBrewer: ColorBrewer Palettes (v 1.1-2)

Random Forest Modeling:
randomForest: Breiman and Cutler's Random Forests for Classification and Regression)

DATA CURATION

Load Datasets and R Packages

```
library(dplyr)
```

Load dplyr package for data transformation

```
##
## 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
```

```
library (ggplot2)
```

Load ggplot2 for data visualization

```
train <- read.csv ("train.csv",  header= TRUE)
  str(train)
```

Load the training and test datasets

```
## 'data.frame':    891 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       : Factor w/ 891 levels "Abbing, Mr. Anthony",..: 109 191 358 277 16 559 520 629 417 58
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ 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     : Factor w/ 681 levels "110152","110413",..: 524 597 670 50 473 276 86 396 345 133 ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : Factor w/ 148 levels "","A10","A14",..: 1 83 1 57 1 1 131 1 1 1 ...
##  $ Embarked   : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

```
test<- read.csv ("test.csv",  header= TRUE)
  str(test)
```

```
## 'data.frame':    418 obs. of  11 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : Factor w/ 418 levels "Abbott, Master. Eugene Joseph",..: 210 409 273 414 182 370 85 5
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 2 2 1 2 1 2 1 2 ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
##  $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ Ticket     : Factor w/ 363 levels "110469","110489",..: 153 222 74 148 139 262 159 85 101 270 ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : Factor w/ 77 levels "","A11","A18",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Embarked   : Factor w/ 3 levels "C","Q","S": 2 3 2 3 3 3 2 3 1 3 ...
```

Data Transformation and Verification

```
train$Survived<- factor(train$Survived)
```

Change the *Survived* variable from int to factor before combining

```
test<- mutate(test, Survived = "none")
```

Create *Survived* dummy variable in test set before combining

```
test <- mutate(test, dataset = "testset")
train <- mutate(train, dataset = "trainset")
```

Create sorting variable *dataset* before combining

```
titanic.combined <- rbind(test, train)
str(titanic.combined)
```

Combine training and test datasets for feature engineering

```
## 'data.frame':    1309 obs. of  13 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : Factor w/ 1307 levels "Abbott, Master. Eugene Joseph",..: 210 409 273 414 182 370 85
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 2 2 1 2 1 2 1 2 ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
##  $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ Ticket     : Factor w/ 929 levels "110469","110489",..: 153 222 74 148 139 262 159 85 101 270 ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : Factor w/ 187 levels "","A11","A18",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Embarked   : Factor w/ 4 levels "C","Q","S","": 2 3 2 3 3 3 2 3 1 3 ...
##  $ Survived   : chr  "none" "none" "none" "none" ...
##  $ dataset    : chr  "testset" "testset" "testset" "testset" ...
```

```
data<- tbl_df (titanic.combined)
```

Rename and create local data frame for simplicity

```
data$Pclass <- factor(data$Pclass)
data$dataset <- factor(data$dataset)
data$Survived<- factor(data$Survived)
```

Factorize *Pclass, dataset and Survived* variables

```
IDdups <- distinct(data, PassengerId)
dim(IDdups)
```

Check for duplicates

```
## [1] 1309   13
```

```
Namedups <- distinct(data, Name)
dim(Namedups)
```

```
## [1] 1307   13
```

Since there are only 1307 distinct names in the dataset, there may be 2 duplicates. However there are 1309 distinct Passenger ID's

```
filter(data, duplicated(Name))
```

```
filter(data, grepl('Kelly|Connolly', Name, Age ))
```

```
## Source: local data frame [7 x 13]
##
##   PassengerId Pclass                                     Name    Sex   Age
##         (int) (fctr)                                    (fctr) (fctr) (dbl)
## 1         892      3                        Kelly, Mr. James   male  34.5
## 2         898      3                     Connolly, Miss. Kate female  30.0
## 3         290      3                     Connolly, Miss. Kate female  22.0
## 4         301      3 Kelly, Miss. Anna Katherine "Annie Kate" female    NA
## 5         574      3                        Kelly, Miss. Mary female    NA
## 6         697      3                         Kelly, Mr. James   male  44.0
## 7         707      2            Kelly, Mrs. Florence "Fannie" female  45.0
## Variables not shown: SibSp (int), Parch (int), Ticket (fctr), Fare (dbl),
##   Cabin (fctr), Embarked (fctr), Survived (fctr), dataset (fctr)
```

The different age and ticket numbers of the potential duplicates indicate these are different with the same name not duplicates.

Data Exploration

```
summary(tbl_df(data))
```

Descriptive stats for the data

```
##   PassengerId   Pclass                                        Name
##  Min.   :   1   1:323   Connolly, Miss. Kate                     :   2
##  1st Qu.: 328   2:277   Kelly, Mr. James                         :   2
##  Median : 655   3:709   Abbott, Master. Eugene Joseph            :   1
##  Mean   : 655           Abelseth, Miss. Karen Marie              :   1
##  3rd Qu.: 982           Abelseth, Mr. Olaus Jorgensen            :   1
##  Max.   :1309           Abrahamsson, Mr. Abraham August Johannes:   1
##                         (Other)                                  :1301
##      Sex           Age            SibSp            Parch
##  female:466   Min.   : 0.17   Min.   :0.0000   Min.   :0.000
##  male  :843   1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.000
```

```
##                Median :28.00   Median :0.0000   Median :0.000
##                Mean   :29.88   Mean   :0.4989   Mean   :0.385
##                3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.000
##                Max.   :80.00   Max.   :8.0000   Max.   :9.000
##                NA's   :263
##       Ticket          Fare                      Cabin      Embarked
##  CA. 2343:  11   Min.   :  0.000                  :1014   C:270
##  1601    :   8   1st Qu.:  7.896   C23 C25 C27    :   6   Q:123
##  CA 2144 :   8   Median : 14.454   B57 B59 B63 B66:   5   S:914
##  3101295 :   7   Mean   : 33.295   G6             :   5    :  2
##  347077  :   7   3rd Qu.: 31.275   C22 C26        :   4
##  PC 17608:   7   Max.   :512.329   C78            :   4
##  (Other) :1261   NA's   :1         (Other)        : 271
##  Survived       dataset
##  0   :549   testset :418
##  1   :342   trainset:891
##  none:418
##
##
##
##
```

```
head(data)
```

```
## Source: local data frame [6 x 13]
##
##   PassengerId Pclass                                          Name    Sex
##         (int) (fctr)                                         (fctr) (fctr)
## 1         892      3                              Kelly, Mr. James   male
## 2         893      3              Wilkes, Mrs. James (Ellen Needs) female
## 3         894      2                     Myles, Mr. Thomas Francis   male
## 4         895      3                              Wirz, Mr. Albert   male
## 5         896      3 Hirvonen, Mrs. Alexander (Helga E Lindqvist) female
## 6         897      3                    Svensson, Mr. Johan Cervin   male
## Variables not shown: Age (dbl), SibSp (int), Parch (int), Ticket (fctr),
##   Fare (dbl), Cabin (fctr), Embarked (fctr), Survived (fctr), dataset
##   (fctr)
```

1. overall *Age* and *Cabin* variables are missing ~20% of values
2. *Fare* is missing 1 value
3. *Embarked* is missing 2 values

Visualize some potentially important features as a function of survival

```
trainset<-data%>% arrange(dataset)%>%slice(419:1309)
head (trainset)
glimpse(trainset)
```

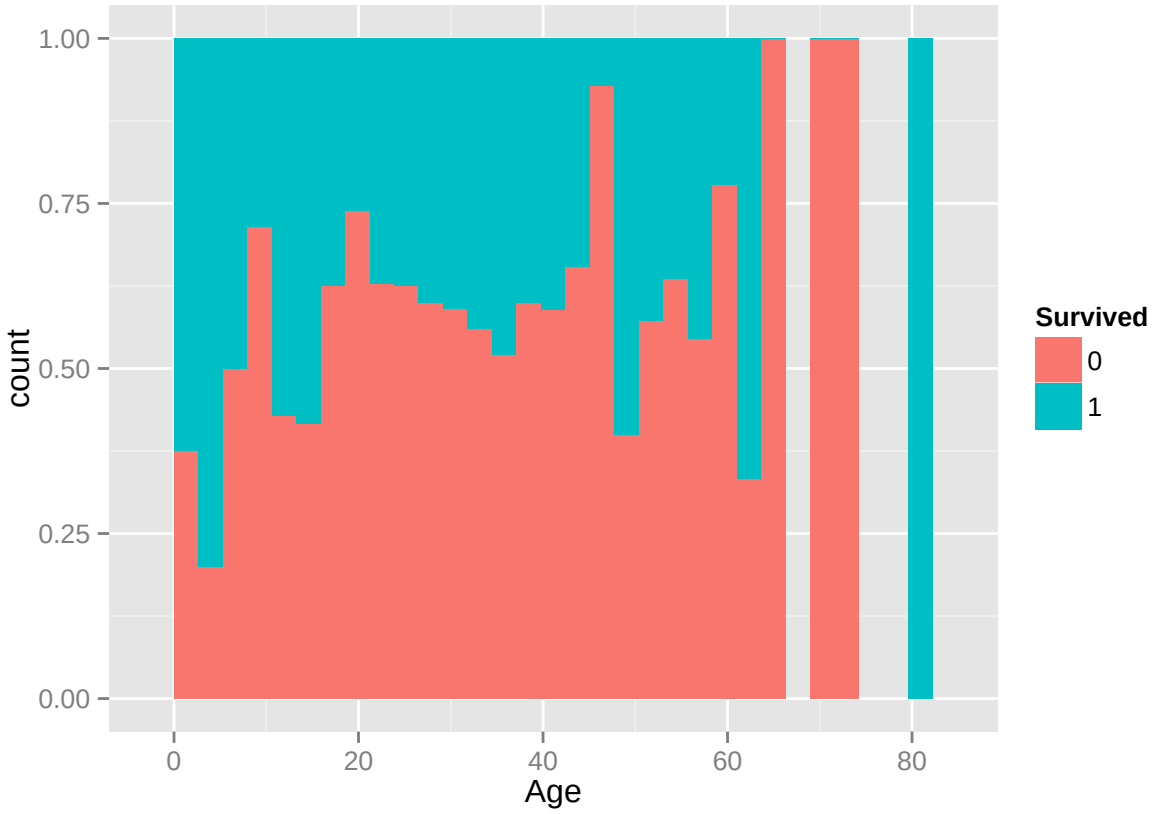
```
hist_Age <- ggplot(trainset, aes(x=Age, fill=Survived))
  hist_Age + geom_bar() # defaults to stacking
```

Age

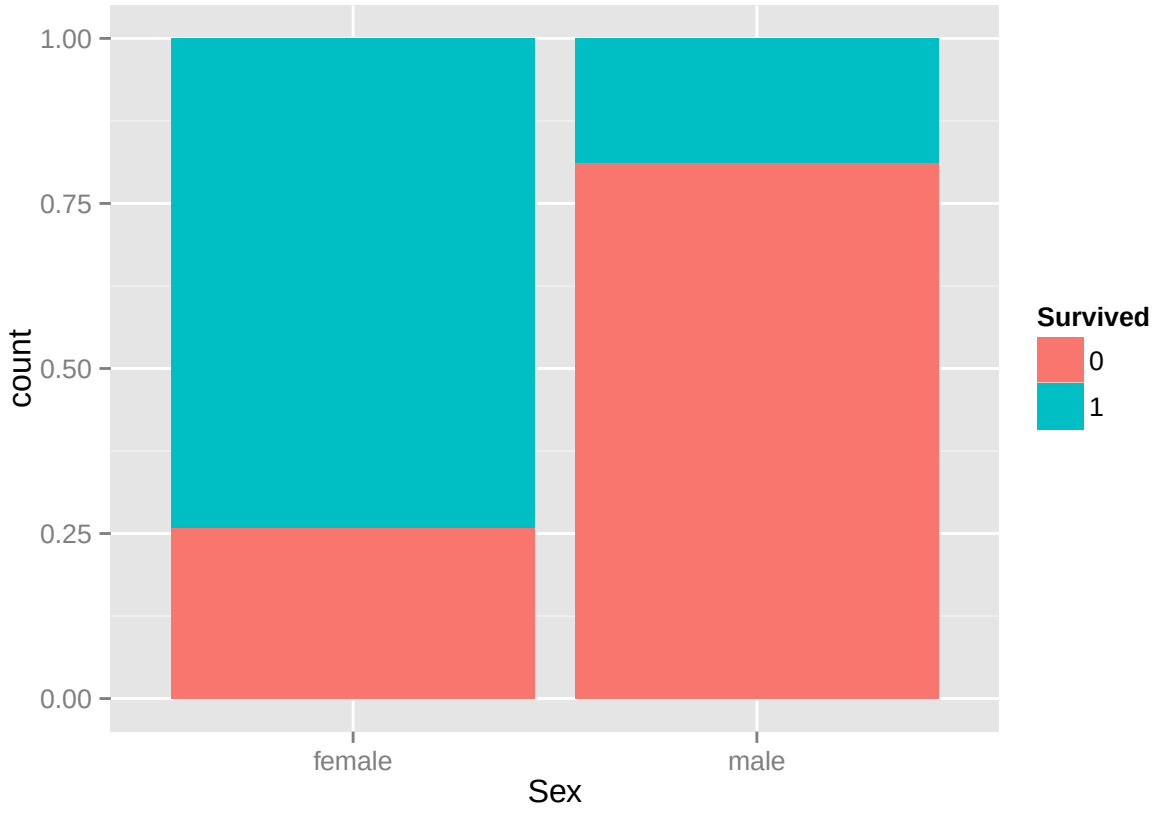
```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```


```
  hist_Age + geom_bar(position= "fill") #proportions
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```

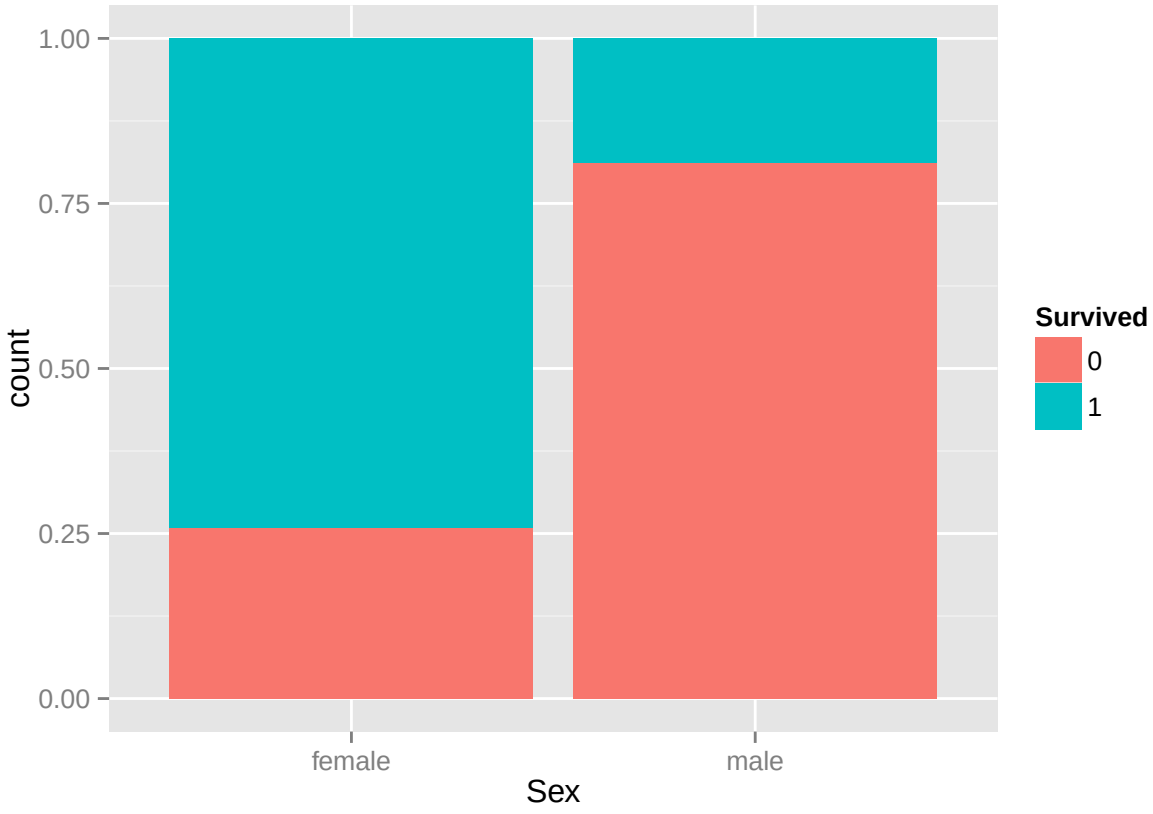



```
hist_Sex <- ggplot(trainset, aes(x=Sex, fill=Survived))
  hist_Sex + geom_bar(position= "fill") # defaults to stacking
```

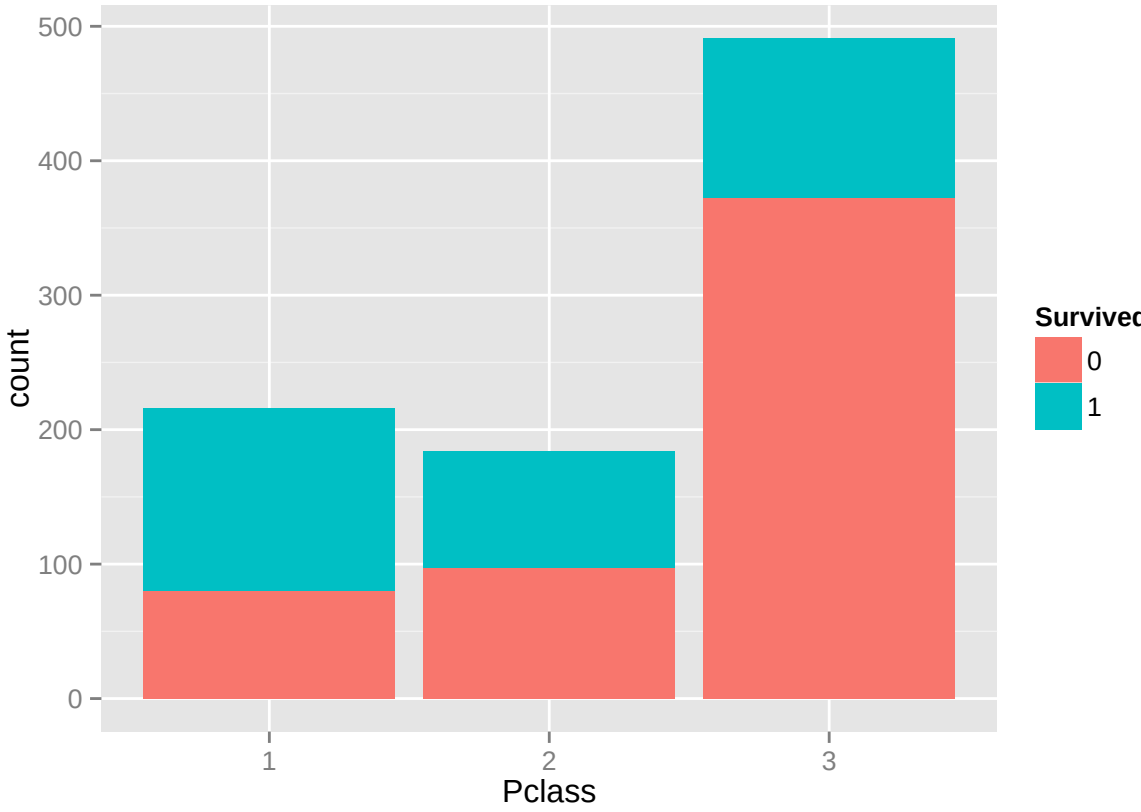


Sex

```
hist_Sex + geom_bar(position= "fill") #proportions
```

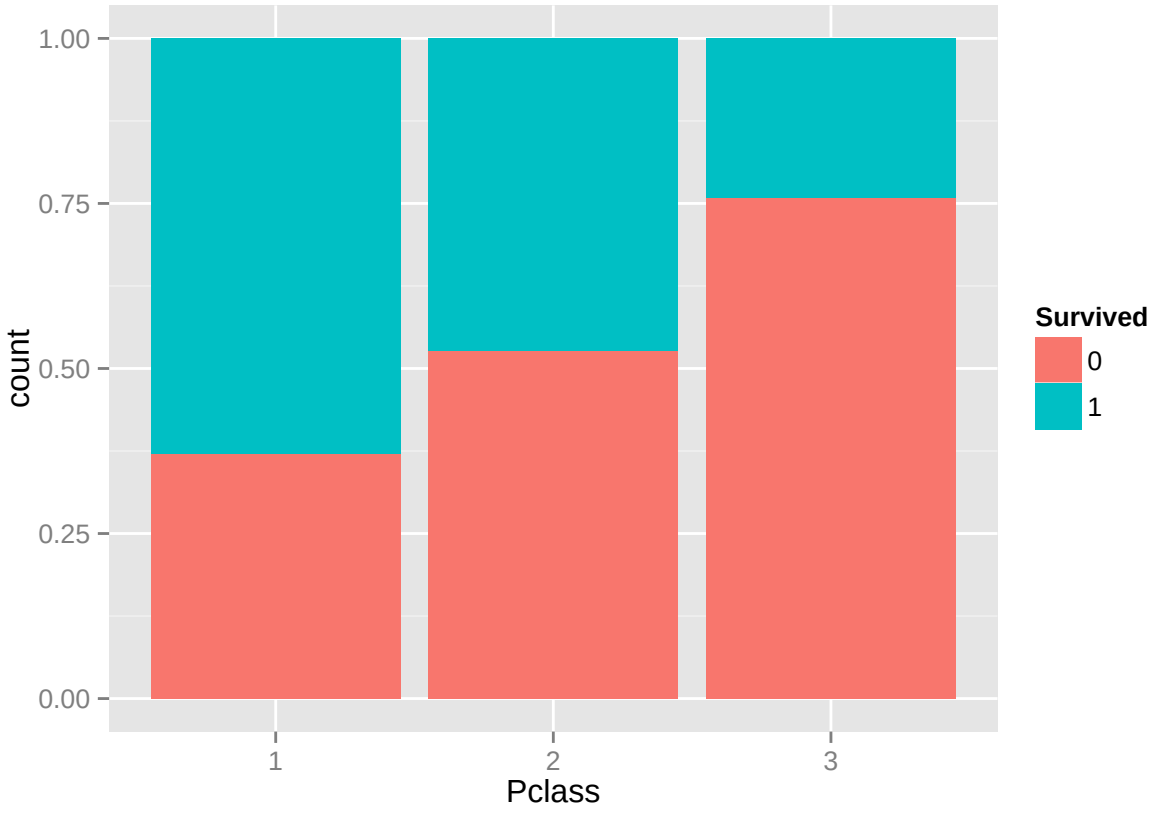



```
hist_Pclass <- ggplot(trainset, aes(x=Pclass, fill=Survived))
  hist_Pclass + geom_bar() # defaults to stacking
```



Pclass (cabin type)

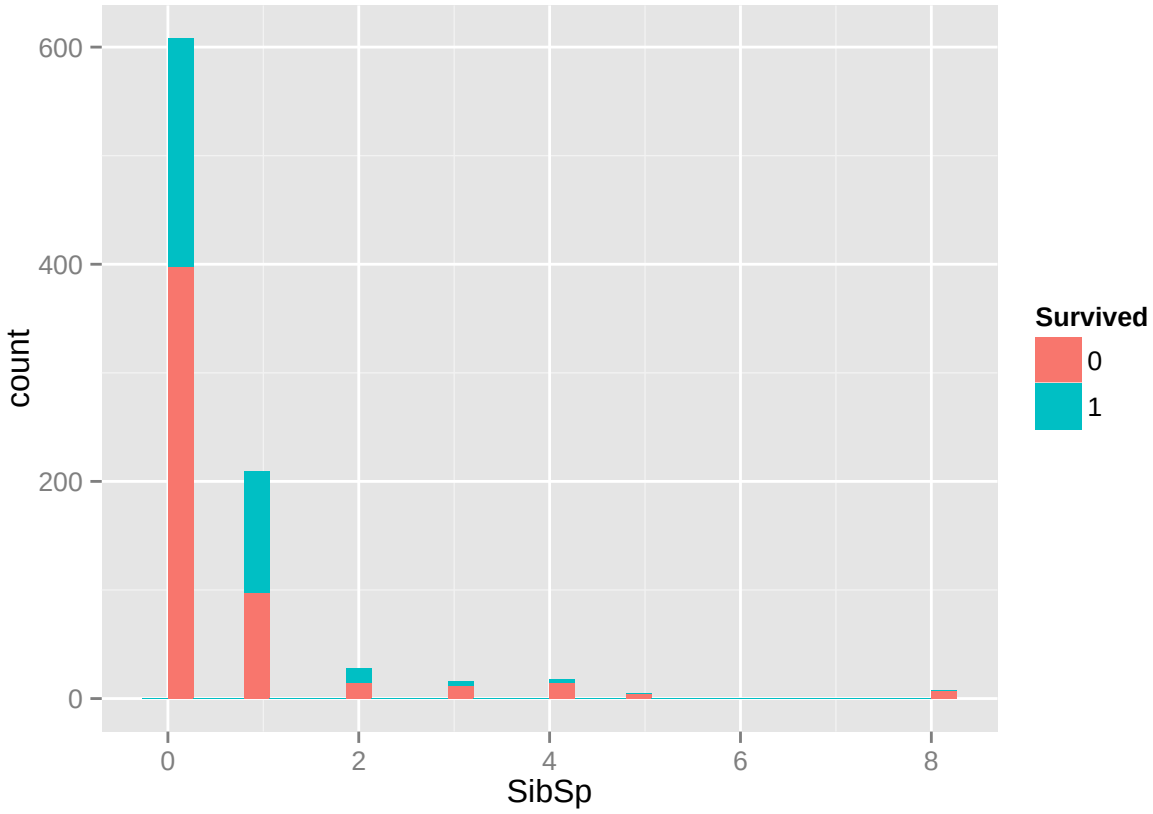
```
hist_Pclass + geom_bar(position= "fill") #proportions
```




```
hist_SibSp <- ggplot(trainset, aes(x=SibSp, fill=Survived, binwidth = .0005))
  hist_SibSp + geom_bar() # defaults to stacking
```

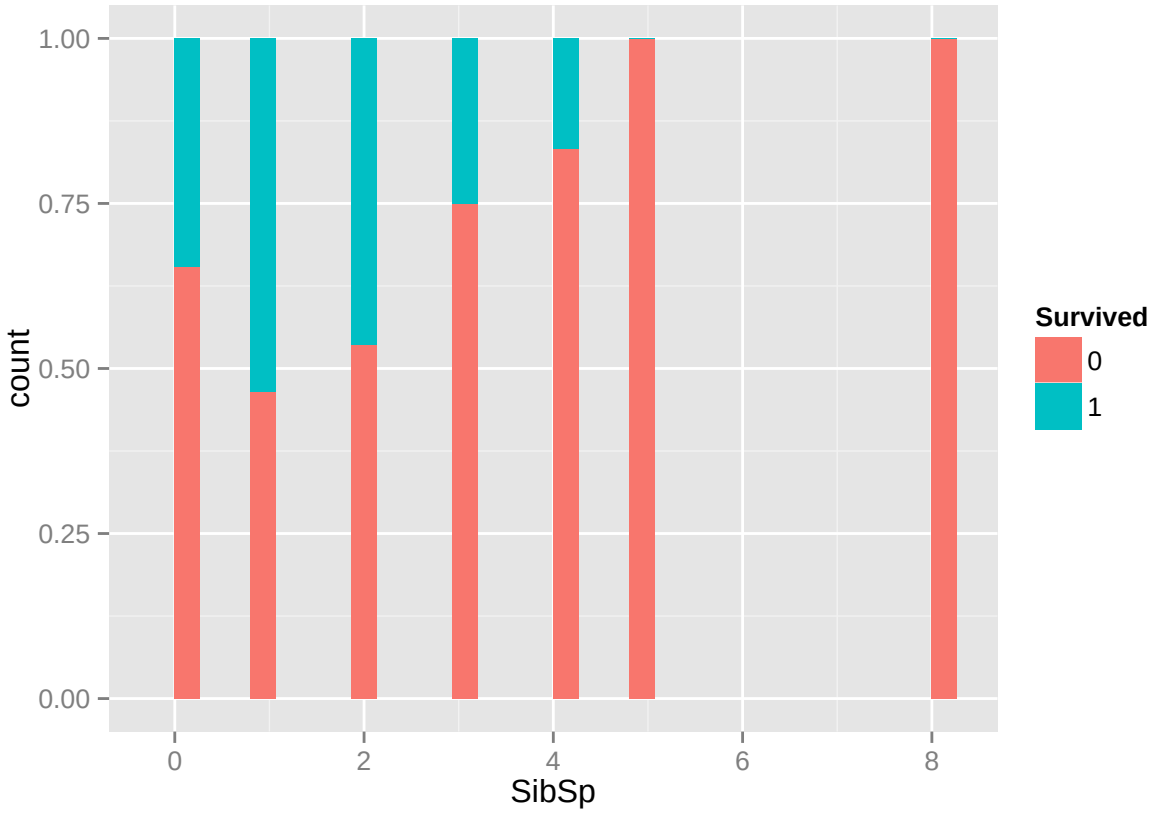
SibSp (no. siblings/spouse)

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```




```
hist_SibSp + geom_bar(position= "fill") #proportions
```

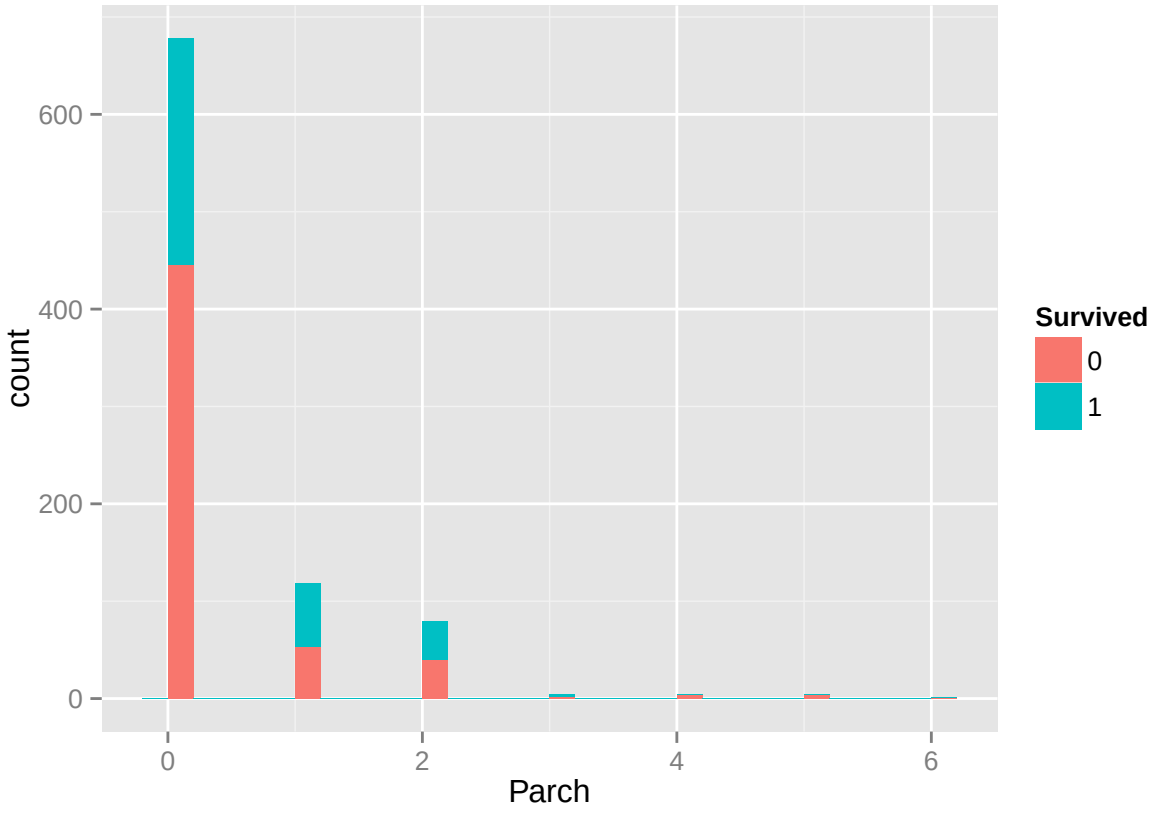
```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```




```
hist_Parch <- ggplot(trainset, aes(x=Parch, fill=Survived))
  hist_Parch + geom_bar() # defaults to stacking
```

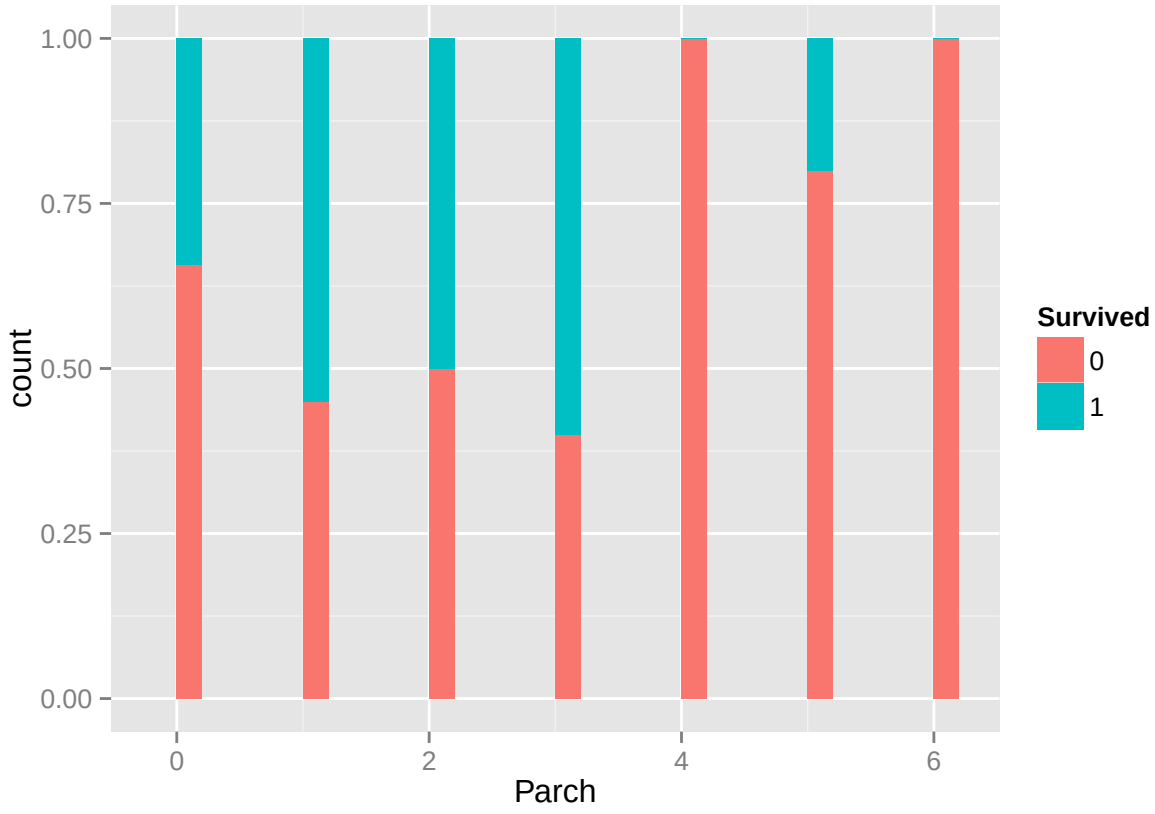
Parch (no. parents/children)

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```




```
hist_Parch + geom_bar(position= "fill") #proportions
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



Feature Engineering

Hypothesis 1: **data visualization suggests being a child and/or a female increased your odds of survival**

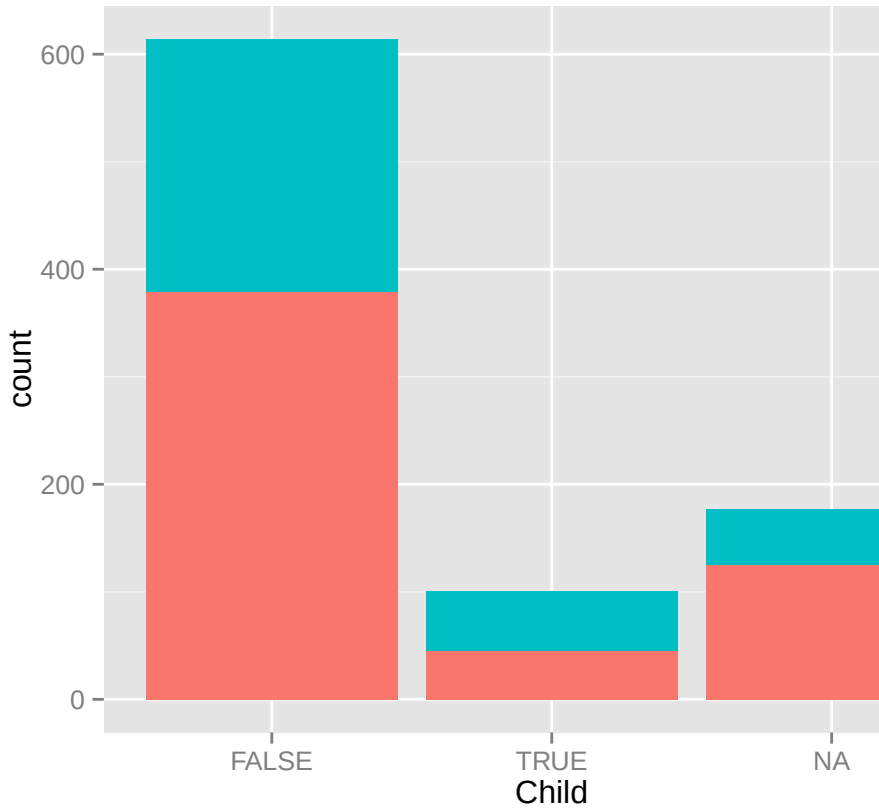
```
data <- data %>%
  mutate(Child = Age <=16)
data$Child <- factor(data$Child)
  glimpse (data)
```

Create feature *Child* from feature *Age* <= 16 yrs

```
## Observations: 1,309
## Variables: 14
## $ PassengerId (int) 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, ...
## $ Pclass      (fctr) 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 1, 1, 2, 1, 2, 2...
## $ Name        (fctr) Kelly, Mr. James, Wilkes, Mrs. James (Ellen Needs...
## $ Sex         (fctr) male, female, male, male, female, male, female, m...
## $ Age         (dbl) 34.5, 47.0, 62.0, 27.0, 22.0, 14.0, 30.0, 26.0, 18...
## $ SibSp       (int) 0, 1, 0, 0, 1, 0, 0, 1, 0, 2, 0, 0, 1, 1, 1, 1, 0,...
## $ Parch       (int) 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Ticket      (fctr) 330911, 363272, 240276, 315154, 3101298, 7538, 33...
```

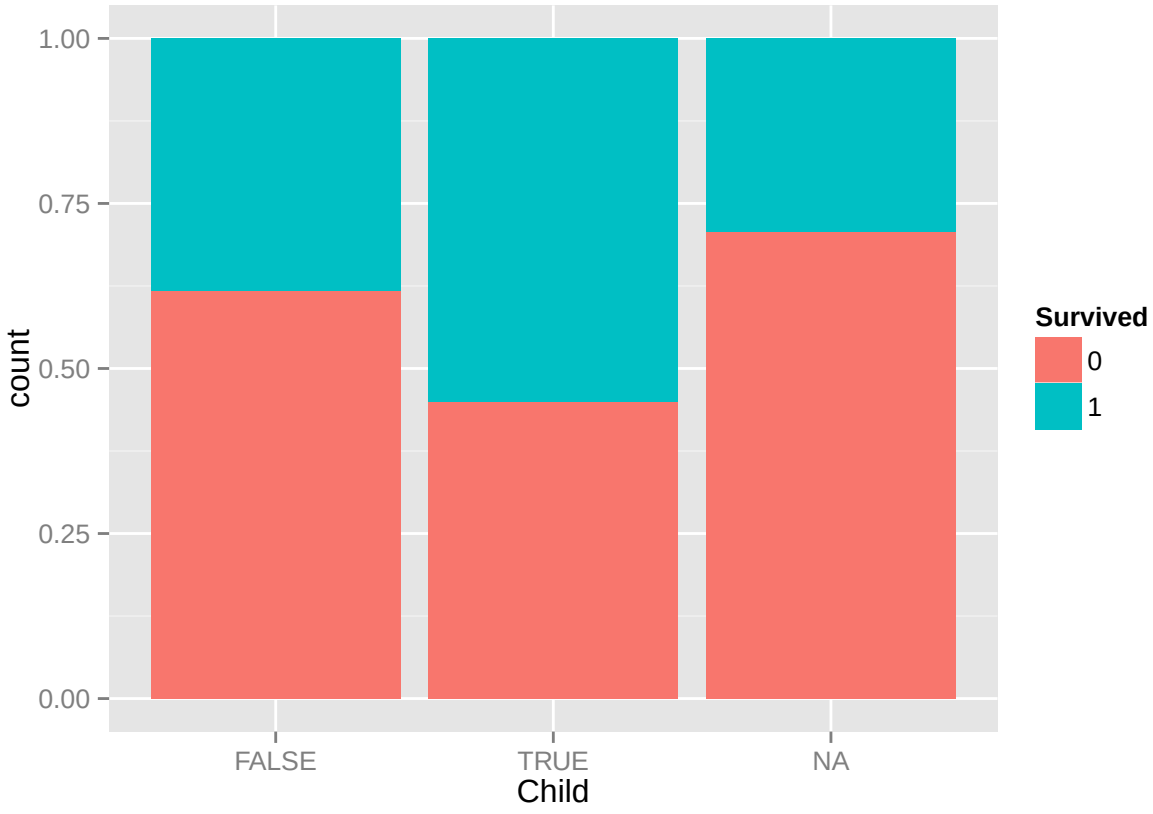
```
## $ Fare        (dbl) 7.8292, 7.0000, 9.6875, 8.6625, 12.2875, 9.2250, 7...
## $ Cabin       (fctr) , , , , , , , , , , , B45, , E31, , , , , , ,...
## $ Embarked    (fctr) Q, S, Q, S, S, S, Q, S, C, S, S, S, S, S, S, C, Q...
## $ Survived    (fctr) none, none, none, none, none, none, none, none, n...
## $ dataset     (fctr) testset, testset, testset, testset, testset, test...
## $ Child       (fctr) FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, F...
```

```
trainset<-data%>% arrange(dataset)%>%slice(419:1309)

hist_Child <- ggplot(trainset, aes(x=Child, fill=Survived))
  hist_Child + geom_bar() # defaults to stacking
```



Visualize survival as a function of Child

```
  hist_Child + geom_bar(position= "fill") #proportions
```



Hypothesis 2: **Did a persons Title effect survivability?**

```
Mr<-filter(data, grepl('Mr.' ,Name, fixed=TRUE ))
  Mr<-mutate(Mr, title = 'Mr')

Mrs<-filter(data, grepl('Mrs.', Name, fixed=TRUE ))
  Mrs<-mutate(Mrs, title = 'Mrs')

Miss<-filter(data, grepl('Miss.', Name, fixed=TRUE ))
  Miss<-mutate(Miss, title = 'Miss')

Master<-filter(data, grepl('Master.', Name, fixed=TRUE  ))
  Master<-mutate(Master, title = 'Master')

Dr <-filter(data, grepl('Dr.', Name, fixed=TRUE  ))
  Dr<-mutate(Dr, title = 'UCMale')

Rev<-filter(data, grepl('Rev.', Name, fixed=TRUE  ))
  Rev<-mutate(Rev, title = 'UCMale')

Ms<-filter(data, grepl('Ms.', Name, fixed=TRUE  ))
  Ms<-mutate(Ms, title = 'Mrs')
```

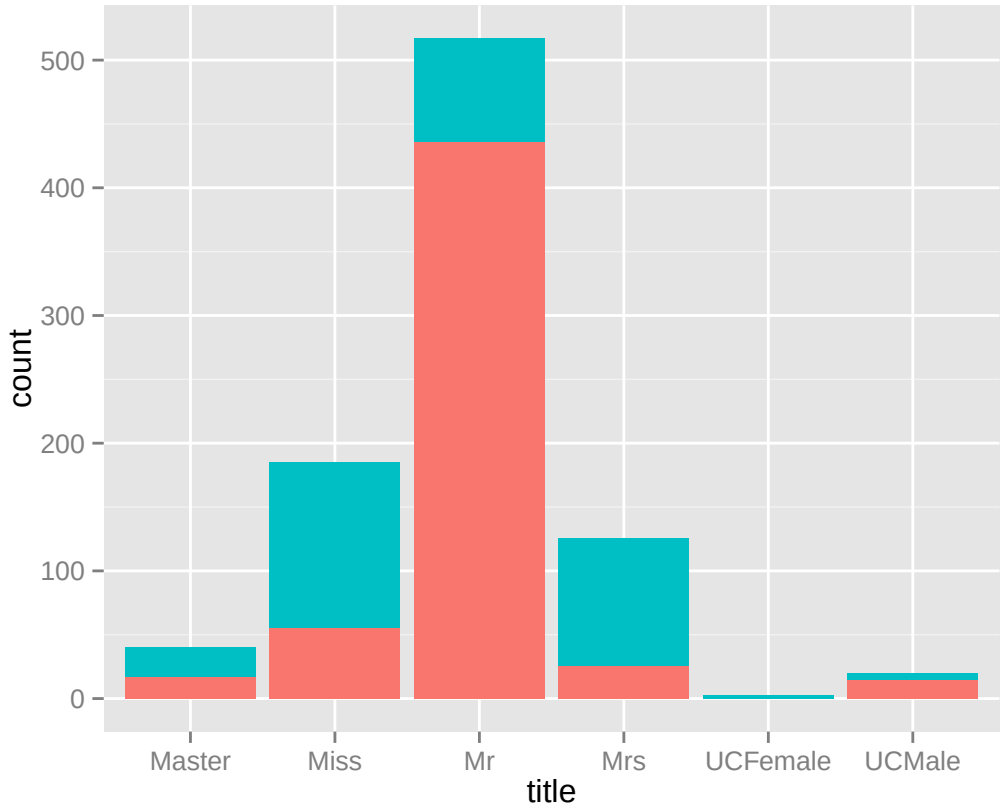
```
Major<-filter(data, grepl('Major.', Name, fixed=TRUE  ))
  Major<-mutate(Major, title = 'UCMale')

Col<-filter(data, grepl('Col.', Name, fixed=TRUE  ))
  Col<-mutate(Col, title = 'UCMale')

Dona<-filter(data, grepl('Dona.', Name, fixed=TRUE  ))
  Dona<-mutate(Dona, title = 'UCFemale')

Don<-filter(data, grepl('Don.', Name, fixed=TRUE  ))
  Don<-mutate(Don, title = 'UCMale')

Capt<-filter(data, grepl('Capt.', Name, fixed=TRUE  ))
  Capt<-mutate(Capt, title = 'UCMale')

Sir<-filter(data, grepl('Sir.', Name, fixed=TRUE  ))
  Sir<-mutate(Sir, title = 'UCMale')

Lady<-filter(data, grepl('Lady.', Name, fixed=TRUE  ))
  Lady<-mutate(Lady, title = 'UCFemale')

Mlle<-filter(data, grepl('Mlle.', Name, fixed=TRUE  ))
  Mlle<-mutate(Mlle, title = 'Miss')

Mme<-filter(data, grepl('Mme.', Name, fixed=TRUE  ))
  Mme<-mutate(Mme, title = 'Miss')

Ctss<-filter(data, grepl('Countess.', Name, fixed=TRUE  ))
  Ctss<-mutate(Ctss, title = 'UCFemale')

Jonk<-filter(data, grepl('Jonkheer.', Name, fixed=TRUE  ))
  Jonk<-mutate(Jonk, title = 'UCMale')

Dr<-Dr[-8, ] # remove the female Dr from 'Dr' df

FDr<-filter(data, grepl('Leader', Name, fixed=TRUE  ))
  FDr<-mutate(FDr, title = 'UCFemale')

# Create seperate title class, by sex, for people with titles indicative of the upper class
UCMale<- rbind(Dr, Rev, Sir, Major, Col, Capt, Don, Jonk)
UCFemale<- rbind(Lady, Dona, Ctss, FDr)

# combine "Ms" with "Mrs" and "Mme"/"Mlle" with Miss
Mrs<- rbind(Mrs, Ms)
Miss<- rbind(Miss, Mme, Mlle)

# combine all title into one variable "title"
tbl_df(alltitles<-rbind(Mr, Mrs, Miss, Master, UCMale, UCFemale))
  glimpse (alltitles)
  tail(alltitles)

# create dummy variable for data df
data<-mutate(data, title = "none")
```

```r
glimpse(data)

data<-arrange(data, PassengerId)
head(data)

alltitles<- arrange(alltitles, PassengerId)
head(alltitles)

# add new feature "title" to data df
data$title<-alltitles$title
  summary(data)

data$title <- factor(data$title)#factorize 'title'
```

Create new feature called *Title* based on the *Name* feature

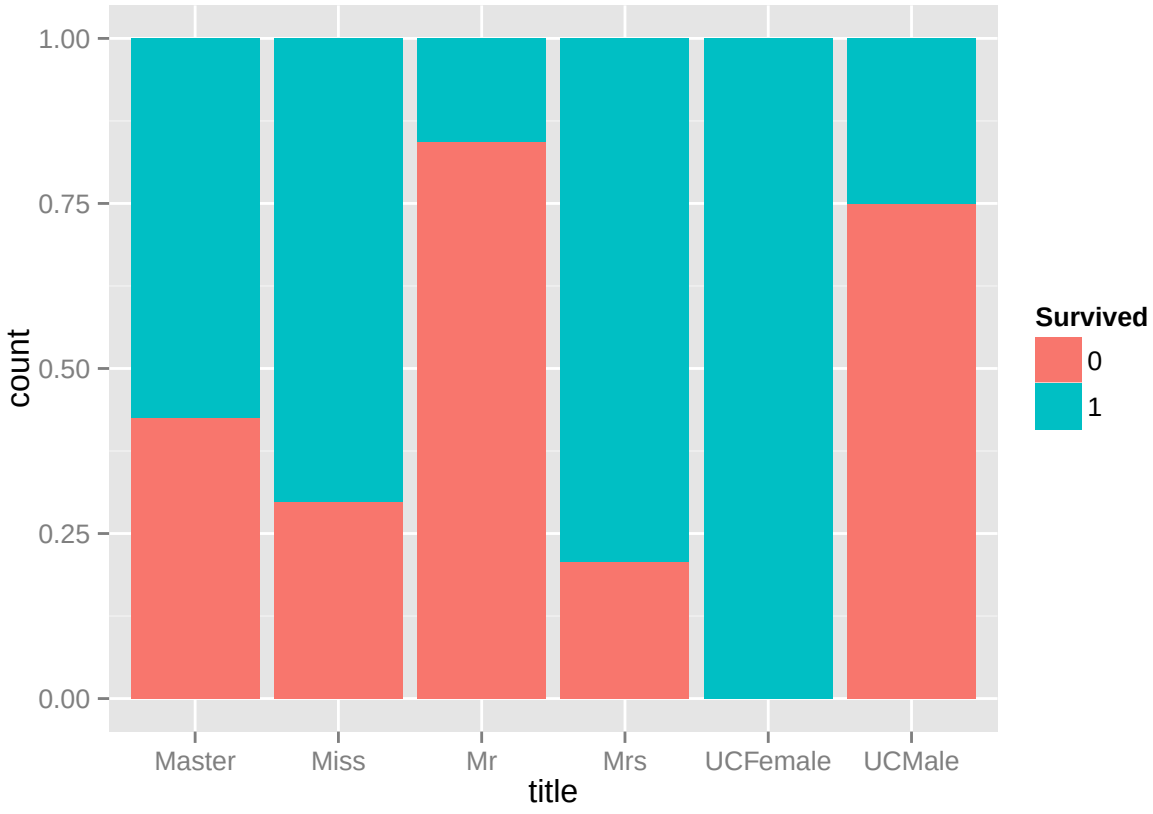
```r
trainset<-data%>% arrange(dataset)%>%slice(419:1309)
head (trainset)
glimpse(trainset)
```

```r
hist_title <- ggplot(trainset, aes(x=title, fill=Survived))
  hist_title + geom_bar() # defaults to stacking
```



Survival as a function of title

```
hist_title + geom_bar(position= "fill") #proportions
```




```
data%>%
  group_by(title)%>%
  filter(!is.na(Age))%>%
    summarise(min(Age))
```

Verify Age range for each title group

```
## Source: local data frame [6 x 2]
##
##      title min(Age)
##     (fctr)    (dbl)
## 1   Master     0.33
## 2     Miss     0.17
## 3       Mr    11.00
## 4      Mrs    14.00
## 5 UCFemale    33.00
## 6   UCMale    23.00
```

```
data%>%
  group_by(title)%>%
  filter(!is.na(Age))%>%
  summarise(max(Age))
```

```
## Source: local data frame [6 x 2]
##
##      title max(Age)
##     (fctr)    (dbl)
## 1   Master     14.5
## 2     Miss     63.0
## 3       Mr     80.0
## 4      Mrs     76.0
## 5 UCFemale     49.0
## 6   UCMale     70.0
```

```
under16<-filter(data, Age<=16)
under16%>%group_by(title)%>% summarise(n())
```

How many people with titles of "Mr" and "Mrs" are <=16

```
## Source: local data frame [4 x 2]
##
##    title   n()
##   (fctr) (int)
## 1 Master    53
## 2   Miss    61
## 3     Mr    17
## 4    Mrs     3
```

```
data%>%group_by(title)%>% summarise(n())
```

```
## Source: local data frame [6 x 2]
##
##      title   n()
##     (fctr) (int)
## 1   Master    61
## 2     Miss   263
## 3       Mr   757
## 4      Mrs   199
## 5 UCFemale     4
## 6   UCMale    25
```

```
is.na(data$Child[data$title=="Master"]<-TRUE)
is.na(data$Child[data$title=="Mr" ]<-FALSE)
is.na(data$Child[data$title=="Mrs" ]<-FALSE)
is.na(data$Child[data$title=="UCMale" ]<-FALSE)
is.na(data$Child[data$title=="UCFemale" ]<-FALSE)
is.na(data$Child[data$title=="Miss" ]<-FALSE)
```

Update Child feature based on above data;assume Miss is not a Child

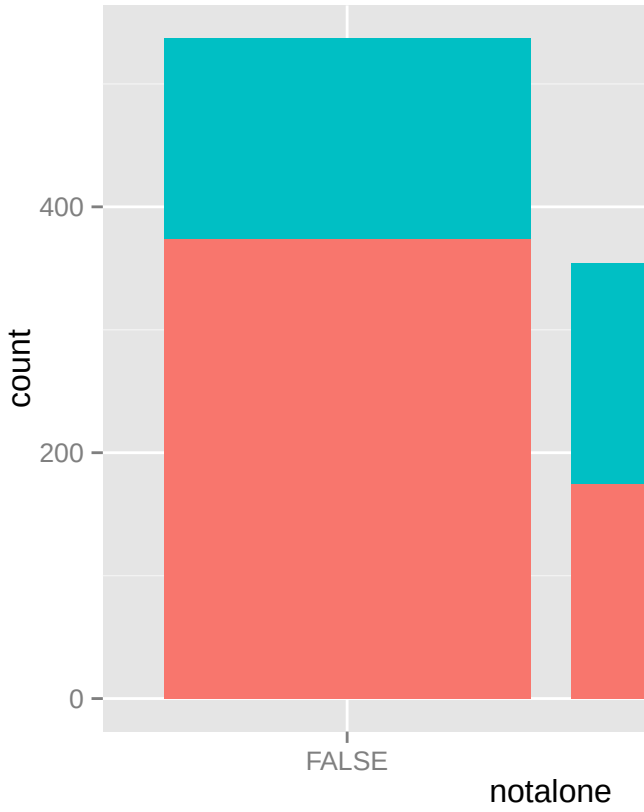
Hypothesis 3: **Data visualization suggests traveling alone decreased your odds of survival but also suggests families >=4 had decreased survival odds**

```
data<- data %>%
  mutate (familysize = SibSp  + Parch +1 ) %>%
  mutate(notalone = familysize >1)

data$notalone<- factor(data$notalone)
  glimpse (data)
```

Create 2 new categorical features *notalone* and *familysize*

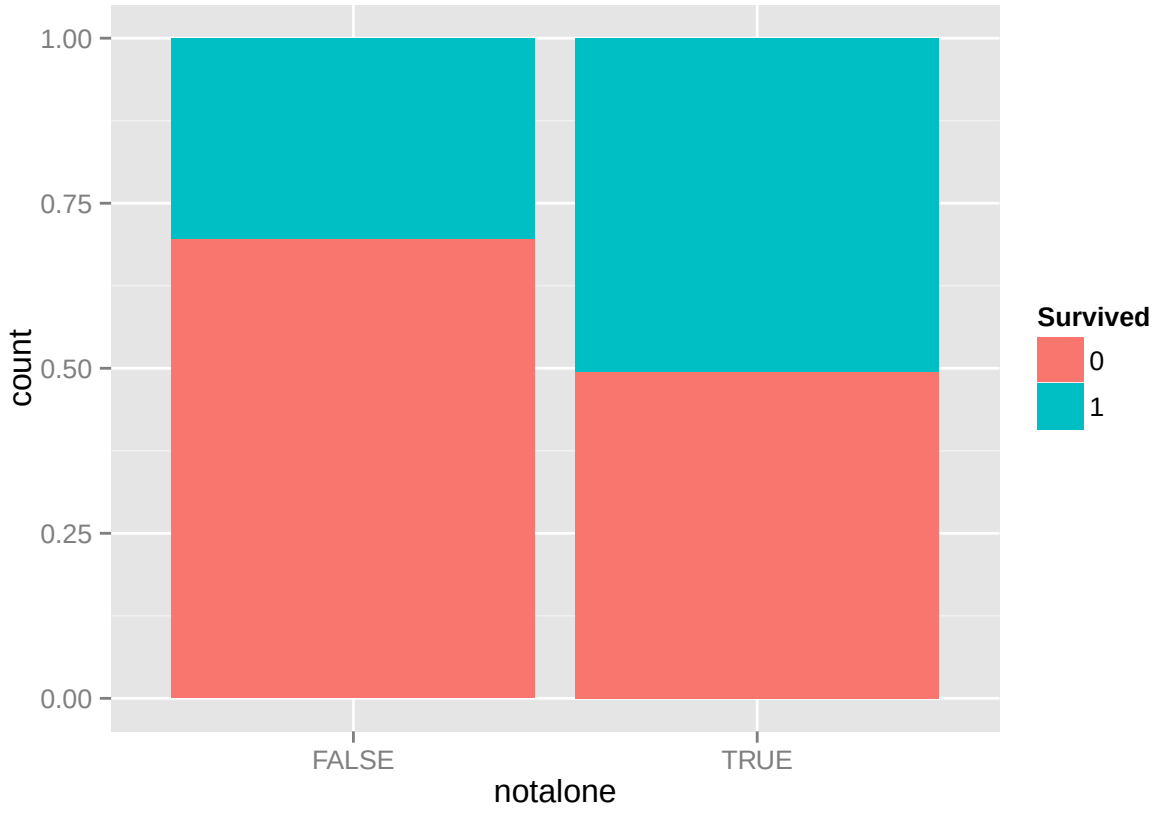
```
trainset<-data%>% arrange(dataset)%>%slice(419:1309)
head (trainset)
glimpse(trainset)
```

```
hist_notalone <- ggplot(trainset, aes(x=notalone, fill=Survived))
  hist_notalone + geom_bar() # defaults to stacking
```



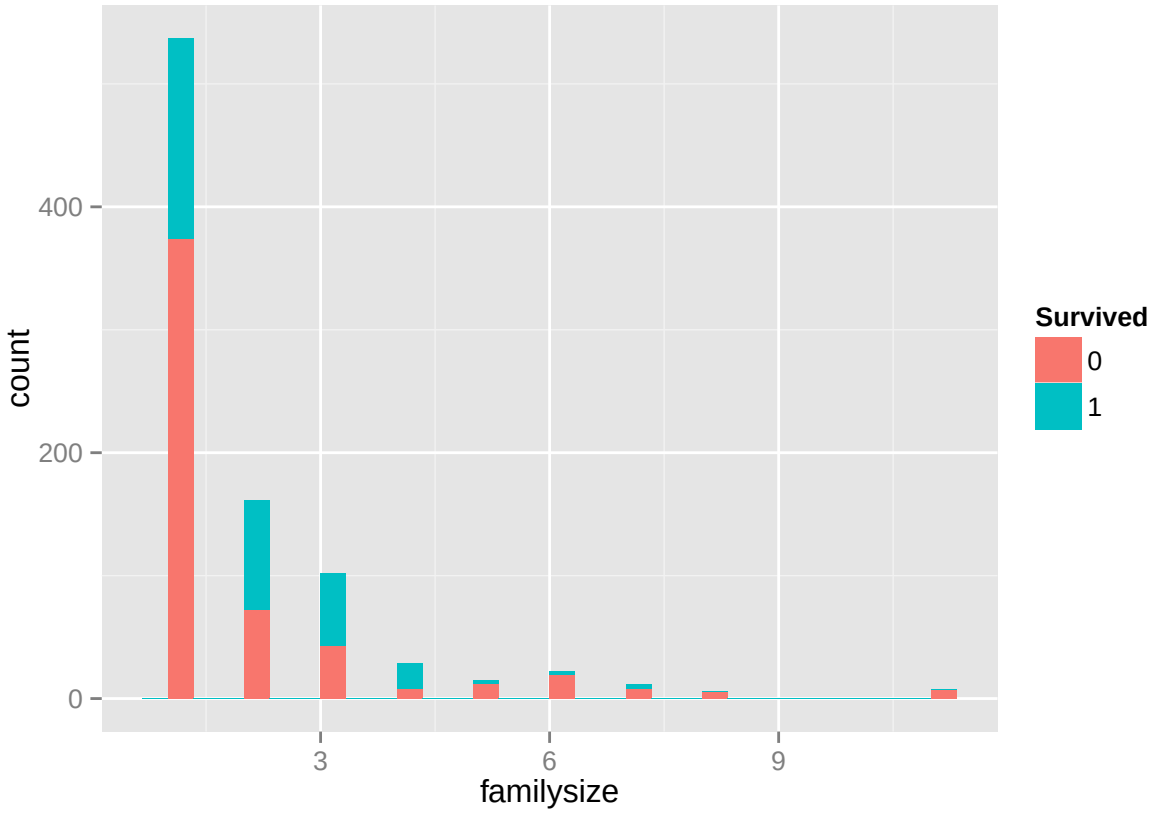
Visualize survival as a function of notalone and familysize

```
  hist_notalone + geom_bar(position= "fill") #proportions
```



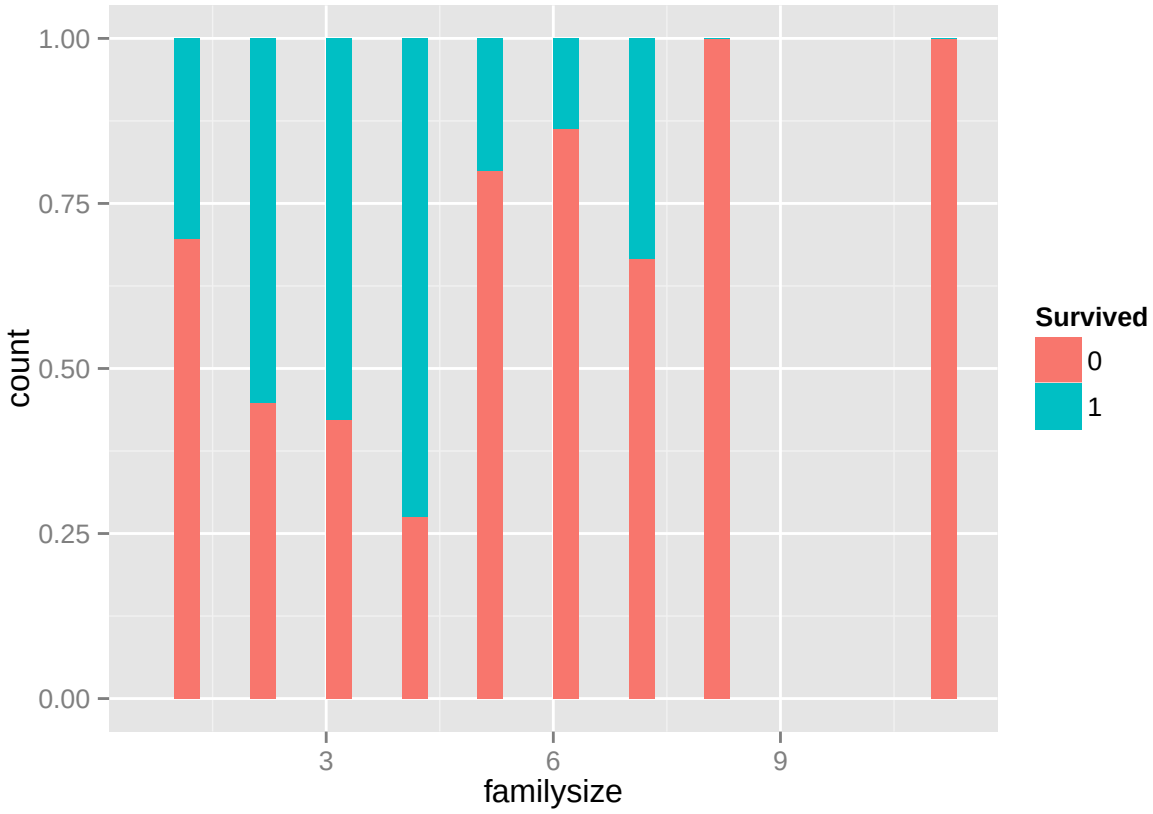

```
hist_familysize <- ggplot(trainset, aes(x=familysize, fill=Survived))
  hist_familysize + geom_bar() # defaults to stacking
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```




```
  hist_familysize + geom_bar(position= "fill") #proportions
```

```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



Hypothesis 4: **data visualization suggests that small families had increased odds of survival**

```
data$smallfamily[data$familysize >1 & data$familysize<=4] <-1
data$smallfamily[data$familysize == 1 | data$familysize>4 ] <-0
data$smallfamily <- factor(data$smallfamily)
```

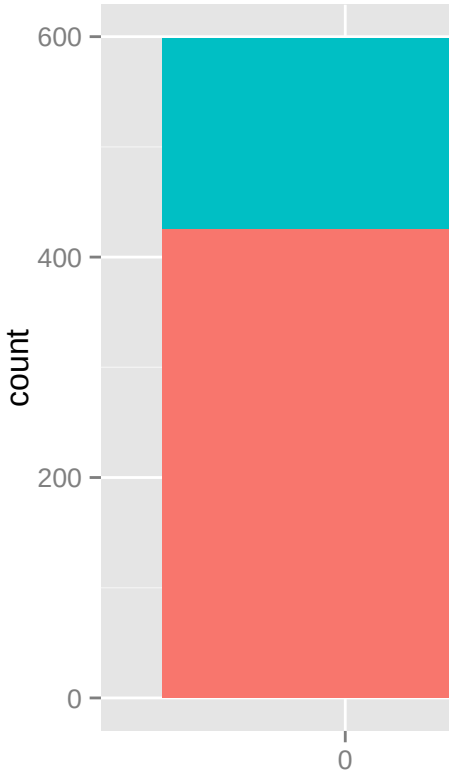
Create new categorical feature *smallfamily* from *familysize* >1 but <4 (ie between 2-4 people total)

```
data$thirdClass[data$Pclass ==3 ] <-1
data$thirdClass[data$Pclass ==1 | data$Pclass==2 ] <-0
data$thirdClass <- factor(data$thirdClass)
```

Create feature for just 3rd Class to test as a surrogate for *Pclass*

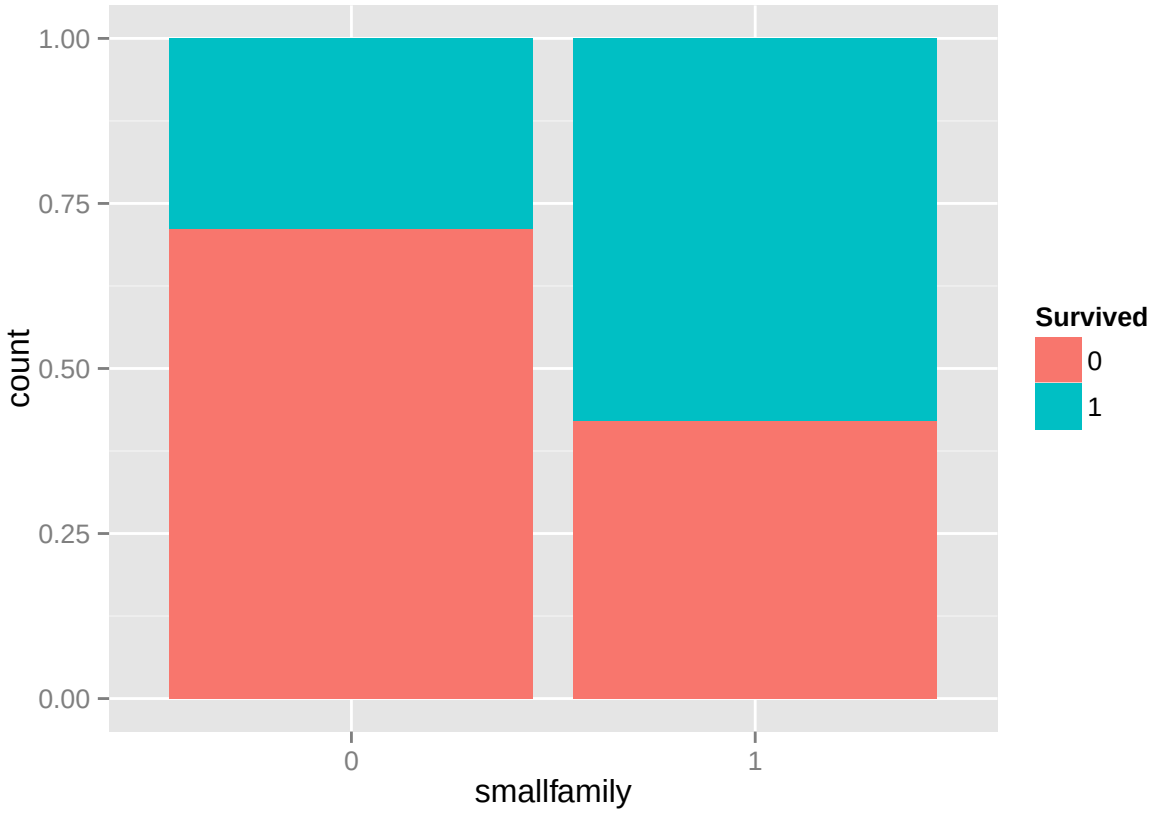
```
trainset<-data%>% arrange(dataset)%>%slice(419:1309)
head (trainset)
glimpse(trainset)
```

```
hist_smallfamily <- ggplot(trainset, aes(x=smallfamily, fill=Survived))
  hist_smallfamily + geom_bar() # defaults to stacking
```

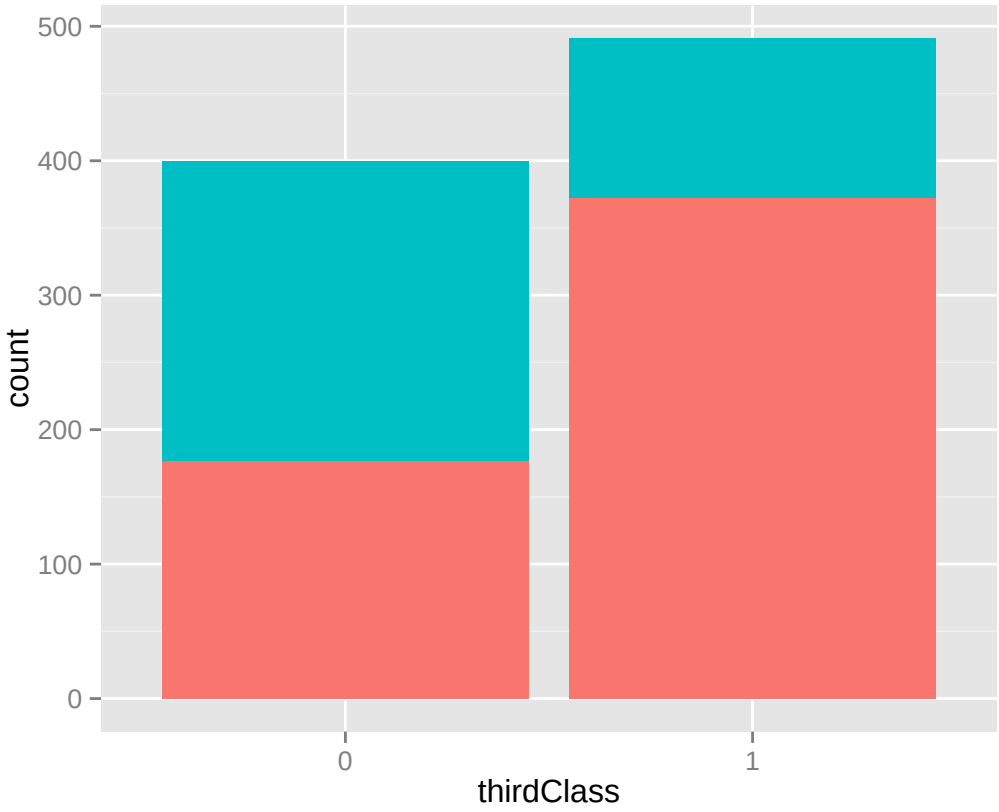


Visualize survival as a function of having a smallfamily or 3rd class cabin

```
  hist_smallfamily + geom_bar(position= "fill") #proportions
```

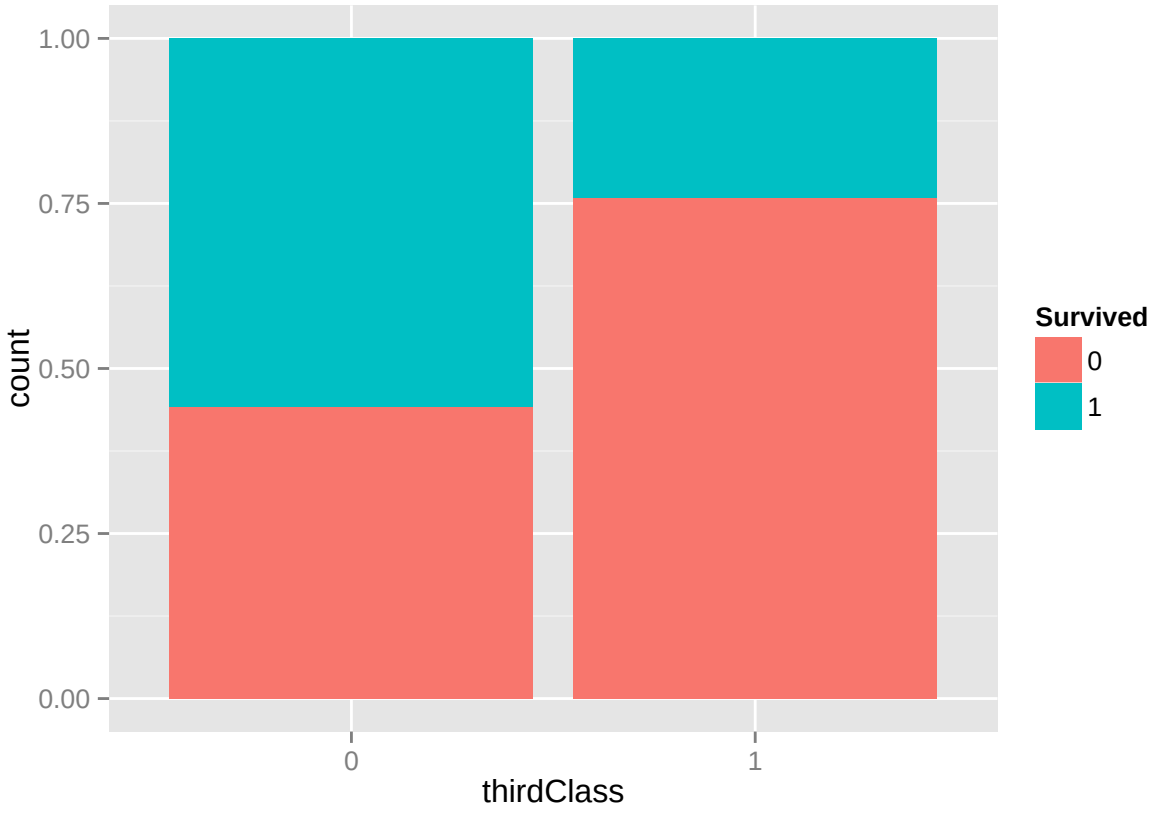



```
hist_thirdclass <- ggplot(trainset, aes(x=thirdClass, fill=Survived))
  hist_thirdclass + geom_bar() # defaults to stacking
```



Visualize *thirdClass*

```
hist_thirdclass+ geom_bar(position= "fill") #proportions
```




```
ageimp <- lm(Age~ Pclass+smallfamily+SibSp+title,  data= data)
summary(ageimp)
```

Impute value for *Age* based on logit model

```
##
## Call:
## lm(formula = Age ~ Pclass + smallfamily + SibSp + title, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.655  -7.641  -1.114   6.040  44.359
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    20.6258     1.8956  10.881  < 2e-16 ***
## Pclass2        -9.5378     0.9421 -10.124  < 2e-16 ***
## Pclass3       -13.0145     0.8575 -15.177  < 2e-16 ***
## smallfamily1   -3.5751     0.8029  -4.453 9.39e-06 ***
## SibSp          -0.9661     0.4250  -2.273   0.0232 *
## titleMiss      11.3967     1.7904   6.365 2.92e-10 ***
## titleMr        22.0297     1.7312  12.725  < 2e-16 ***
## titleMrs       26.5126     1.8398  14.410  < 2e-16 ***
```

```
## titleUCFemale  22.7595     5.7517   3.957 8.11e-05 ***
## titleUCMale    30.7582     2.8093  10.949  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.91 on 1036 degrees of freedom
##   (263 observations deleted due to missingness)
## Multiple R-squared:  0.4321, Adjusted R-squared:  0.4272
## F-statistic: 87.59 on 9 and 1036 DF,  p-value: < 2.2e-16
```

```
# assign imputed Age values for NAs in combined.df
for(i in 1:nrow(data)) {
  if(is.na(data[i, "Age"])) {
    data[i, "Age"] <- predict(ageimp, newdata = data[i, ])
  }
}
```

```
data<-arrange(data, desc(thirdClass))
  data<-arrange(data, SibSp)
  data<-arrange(data, Parch)

threemeanfare<-data[1:472, "Fare"]
summary(threemeanfare)
```

Impute missing fare value for passenger 1044 based on median cost of thirdclass single ticket

```
##       Fare
##  Min.   : 0.000
##  1st Qu.: 7.725
##  Median : 7.854
##  Mean   : 9.097
##  3rd Qu.: 8.050
##  Max.   :56.496
##  NA's   :1
```

```
arrange(data, PassengerId)
data[59,"Fare"]<-7.854
summary(data$Fare)
```

```
data<-arrange(data, dataset)
test<- data[1:418, ]
class(test)
```

Split data df into train and test datasets

```
## [1] "tbl_df"     "data.frame"
```

```
train<-data[419:1309, ]
train$Survived <- droplevels(train$Survived)
test$Survived <- droplevels(test$Survived)
str(test)
```

```
## Classes 'tbl_df' and 'data.frame':   418 obs. of  19 variables:
##  $ PassengerId: int  892 895 897 898 900 902 909 911 919 927 ...
##  $ Pclass     : Factor w/ 3 levels "1","2","3": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Name       : Factor w/ 1307 levels "Abbott, Master. Eugene Joseph",..: 210 414 370 85 5 191 22 21
##  $ Sex        : Factor w/ 2 levels "female","male": 2 2 2 1 1 2 2 1 2 2 ...
##  $ Age        : num  34.5 27 14 30 18 ...
##  $ SibSp      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Parch      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Ticket     : Factor w/ 929 levels "110469","110489",..: 153 148 262 159 101 196 120 121 122 116 .
##  $ Fare       : num  7.83 8.66 9.22 7.63 7.23 ...
##  $ Cabin      : Factor w/ 187 levels "","A11","A18",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Embarked   : Factor w/ 4 levels "C","Q","S","": 2 3 3 2 1 3 1 1 1 1 ...
##  $ Survived   : Factor w/ 1 level "none": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dataset    : Factor w/ 2 levels "testset","trainset": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Child      : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 1 1 1 1 1 ...
##  $ title      : Factor w/ 6 levels "Master","Miss",..: 3 3 3 2 4 3 3 4 3 3 ...
##  $ familysize : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ notalone   : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 1 1 1 1 1 ...
##  $ smallfamily: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ thirdClass : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

```
str(train)
```

```
## Classes 'tbl_df' and 'data.frame':   891 obs. of  19 variables:
##  $ PassengerId: int  3 5 6 13 15 20 23 27 29 30 ...
##  $ Pclass     : Factor w/ 3 levels "1","2","3": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Name       : Factor w/ 1307 levels "Abbott, Master. Eugene Joseph",..: 775 434 975 1150 1263 933 9
##  $ Sex        : Factor w/ 2 levels "female","male": 1 2 2 2 1 1 1 2 1 2 ...
##  $ Age        : num  26 35 29.6 20 14 ...
##  $ SibSp      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Parch      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Ticket     : Factor w/ 929 levels "110469","110489",..: 921 767 586 824 715 513 589 509 594 668 .
##  $ Fare       : num  7.92 8.05 8.46 8.05 7.85 ...
##  $ Cabin      : Factor w/ 187 levels "","A11","A18",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Embarked   : Factor w/ 4 levels "C","Q","S","": 3 3 2 3 3 1 2 1 2 3 ...
##  $ Survived   : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 2 1 2 1 ...
##  $ dataset    : Factor w/ 2 levels "testset","trainset": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Child      : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 1 1 1 1 1 ...
##  $ title      : Factor w/ 6 levels "Master","Miss",..: 2 3 3 3 2 4 2 3 2 3 ...
##  $ familysize : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ notalone   : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 1 1 1 1 1 1 1 ...
##  $ smallfamily: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ thirdClass : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

MACHINE LEARNING PREDICTIVE MODELING

Logistic Regression

```
library(glm2)
```

Load glm2

First perform univariate logistic regression for each important feature

```
agemodel <- glm(Survived ~ Age, family="binomial", data= train)
summary(agemodel)
```

Age

```
##
## Call:
## glm(formula = Survived ~ Age, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1168  -0.9940  -0.9293   1.3490   1.6396
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.145568   0.164363  -0.886   0.3758
## Age         -0.011205   0.005144  -2.178   0.0294 *
## ---
## 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: 1181.8  on 889  degrees of freedom
## AIC: 1185.8
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(agemodel), confint(agemodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 0.8645314 0.6262230 1.1935099
## Age         0.9888573 0.9788534 0.9988148
```

```
sexmodel <- glm(Survived ~ Sex, family="binomial", data= train)
summary(sexmodel)
```

Sex

```
##
## Call:
## glm(formula = Survived ~ Sex, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6462  -0.6471  -0.6471   0.7725   1.8256
##
## 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
```

```
exp(cbind(OR = coef(sexmodel), confint(sexmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                      OR      2.5 %    97.5 %
## (Intercept) 2.87654321 2.24473635 3.7245050
## Sexmale     0.08096732 0.05804709 0.1118353
```

```
Pclassmodel <- glm(Survived ~ Pclass, family="binomial", data= train)
summary(Pclassmodel)
```

Cabin class

```
##
## Call:
## glm(formula = Survived ~ Pclass, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4094  -0.7450  -0.7450   0.9619   1.6836
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.5306     0.1409   3.766 0.000166 ***
## Pclass2      -0.6394     0.2041  -3.133 0.001731 **
```

```
## Pclass3      -1.6704     0.1759  -9.496  < 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: 1083.1  on 888  degrees of freedom
## AIC: 1089.1
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(Pclassmodel), confint(Pclassmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 1.7000000 1.2934647 2.2491345
## Pclass2     0.5275925 0.3528265 0.7858201
## Pclass3     0.1881720 0.1328034 0.2648002
```

```
thirdclassmodel <- glm(Survived ~ thirdClass, family="binomial", data= train)
summary(thirdclassmodel)
```

3rdclass

```
##
## Call:
## glm(formula = Survived ~ thirdClass, family = "binomial", data = train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.277  -0.745  -0.745   1.081   1.684
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.2310     0.1007   2.295   0.0217 *
## thirdClass1  -1.3708     0.1457  -9.409   <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: 1093.0  on 889  degrees of freedom
## AIC: 1097
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(thirdclassmodel), confint(thirdclassmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 1.2598870 1.0348983 1.5360851
## thirdClass1 0.2539057 0.1903755 0.3371002
```

```
sibsmodel <- glm(Survived ~ SibSp, family="binomial", data= train)
summary(sibsmodel)
```

Sibs/spouse

```
##
## Call:
## glm(formula = Survived ~ SibSp, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9979  -0.9979  -0.9711   1.3682   1.4911
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.43815    0.07628  -5.744 9.23e-09 ***
## SibSp       -0.06864    0.06538  -1.050    0.294
## ---
## 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: 1185.5  on 889  degrees of freedom
## AIC: 1189.5
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(sibsmodel), confint(sibsmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 0.6452267 0.5552380 0.7488431
## SibSp       0.9336650 0.8170399 1.0575496
```

```
Parchmodel <- glm(Survived ~ Parch, family="binomial", data= train)
summary(Parchmodel)
```

Parents/children

```
##
## Call:
## glm(formula = Survived ~ Parch, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4705  -0.9533  -0.9533   1.4195   1.4195
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.55305    0.07689  -7.192 6.37e-13 ***
## Parch        0.20332    0.08462   2.403   0.0163 *
## ---
## 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: 1180.8  on 889  degrees of freedom
## AIC: 1184.8
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(Parchmodel), confint(Parchmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 0.5751925 0.4941937 0.6681342
## Parch       1.2254610 1.0388432 1.4493761
```

```
faremodel <- glm(Survived ~ Fare, family="binomial", data= train)
summary(faremodel)
```

Fare

```
##
## Call:
## glm(formula = Survived ~ Fare, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4906  -0.8878  -0.8531   1.3429   1.5942
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.941345   0.095130  -9.895  < 2e-16 ***
## Fare         0.015197   0.002232   6.810 9.78e-12 ***
```

```
## ---
## 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: 1117.6  on 889  degrees of freedom
## AIC: 1121.6
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(faremodel), confint(faremodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %   97.5 %
## (Intercept) 0.3901029 0.3228588 0.468872
## Fare        1.0153134 1.0110827 1.019971
```

```
embmodel <- glm(Survived ~ Embarked, family="binomial", data= train)
summary(embmodel)
```

Embarked

```
##
## Call:
## glm(formula = Survived ~ Embarked, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2700  -0.9065  -0.9065   1.3730   1.4750
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.2151     0.1552   1.386   0.1657
## EmbarkedQ    -0.6641     0.2805  -2.367   0.0179 *
## EmbarkedS    -0.8920     0.1762  -5.063 4.12e-07 ***
## Embarked     13.3510   378.5929   0.035   0.9719
## ---
## 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: 1157.0  on 887  degrees of freedom
## AIC: 1165
##
## Number of Fisher Scoring iterations: 12
```

```
exp(cbind(OR = coef(embmodel), confint(embmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                       OR        2.5 %    97.5 %
## (Intercept) 1.240000e+00 9.157592e-01 1.6846231
## EmbarkedQ   5.147563e-01 2.949021e-01 0.8879565
## EmbarkedS   4.098361e-01 2.895826e-01 0.5781591
## Embarked    6.284144e+05 9.768006e-21        NA
```

```
fsmodel <- glm(Survived ~ familysize, family="binomial", data= train)
summary(fsmodel)
```

Family size

```
##
## Call:
## glm(formula = Survived ~ familysize, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0601  -0.9767  -0.9767   1.3830   1.3924
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.51344    0.10643  -4.824 1.41e-06 ***
## familysize   0.02101    0.04233   0.496     0.62
## ---
## 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: 1186.4  on 889  degrees of freedom
## AIC: 1190.4
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(fsmodel), confint(fsmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 0.5984355 0.4854365 0.7370511
## familysize  1.0212362 0.9388141 1.1091056
```

```
smfammodel <- glm(Survived ~ smallfamily, family="binomial", data= train)
summary(smfammodel)
```

Small family

```
##
## Call:
## glm(formula = Survived ~ smallfamily, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3150  -0.8256  -0.8256   1.0458   1.5760
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.90115    0.09015  -9.996  < 2e-16 ***
## smallfamily1  1.21886    0.14891   8.185 2.72e-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: 1117.6  on 889  degrees of freedom
## AIC: 1121.6
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(smfammodel), confint(smfammodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                      OR     2.5 %    97.5 %
## (Intercept)  0.4061033 0.3395372 0.4835842
## smallfamily1 3.3833357 2.5308197 4.5384864
```

```
namodel <- glm(Survived ~ notalone, family="binomial", data= train)
summary(namodel)
```

Not alone

```
##
## Call:
## glm(formula = Survived ~ notalone, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -1.1870  -0.8506  -0.8506   1.1678   1.5442
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.83051    0.09385  -8.849  < 2e-16 ***
## notaloneTRUE  0.85311    0.14181   6.016 1.79e-09 ***
## ---
## 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: 1150.0  on 889  degrees of freedom
## AIC: 1154
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(namodel), confint(namodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                      OR     2.5 %    97.5 %
## (Intercept)  0.4358289 0.3617448 0.5227575
## notaloneTRUE 2.3469238 1.7790088 3.1024452
```

```
kidmodel <- glm(Survived ~ Child, family="binomial", data= train)
summary(kidmodel)
```

Child

```
##
## Call:
## glm(formula = Survived ~ Child, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3082  -0.9693  -0.9693   1.4009   1.4009
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.51145    0.07081  -7.223  5.1e-13 ***
## ChildTRUE    0.81373    0.32759   2.484    0.013 *
## ---
## 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: 1180.4  on 889  degrees of freedom
## AIC: 1184.4
```

```
##
## Number of Fisher Scoring iterations: 4
```

```
exp(cbind(OR = coef(kidmodel), confint(kidmodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 0.5996241 0.5214688 0.6883834
## ChildTRUE   2.2563157 1.1928235 4.3500385
```

```
titlemodel <- glm(Survived ~ title, family="binomial", data= train)
summary(titlemodel)
```

Title

```
##
## Call:
## glm(formula = Survived ~ title, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7766  -0.5838  -0.5838   0.8400   1.9254
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)     0.3023     0.3198   0.945  0.34462
## titleMiss       0.5579     0.3580   1.558  0.11915
## titleMr        -1.9855     0.3420  -5.806  6.4e-09 ***
## titleMrs        1.0448     0.3883   2.691  0.00713 **
## titleUCFemale  14.2638   509.6522   0.028  0.97767
## titleUCMale    -1.4009     0.6074  -2.306  0.02110 *
## ---
## 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:  879.37  on 885  degrees of freedom
## AIC: 891.37
##
## Number of Fisher Scoring iterations: 13
```

```
exp(cbind(OR = coef(titlemodel), confint(titlemodel))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                          OR        2.5 %    97.5 %
## (Intercept)   1.352941e+00 7.262372e-01 2.5710246
```

```
## titleMiss     1.747036e+00 8.569728e-01 3.5144644
## titleMr       1.373155e-01 6.928608e-02 0.2670172
## titleMrs      2.842809e+00 1.321888e+00 6.1015425
## titleUCFemale 1.565611e+06 2.342854e-23        NA
## titleUCMale   2.463768e-01 6.885592e-02 0.7712754
```

Multivariable logistic regression models

```
model1 <- (step(glm(Survived ~ Sex+smallfamily+notalone+Parch+Child+Age+Fare+thirdClass+SibSp, family="
```

```
## Start:  AIC=1454.9
## Survived ~ Sex + smallfamily + notalone + Parch + Child + Age +
##     Fare + thirdClass + SibSp
##
##               Df Deviance    AIC
## - SibSp        1   1435.1 1453.1
## <none>             1434.9 1454.9
## - Parch        1   1437.3 1455.3
## - Age          1   1438.7 1456.7
## - notalone     1   1449.3 1467.3
## - Fare         1   1450.2 1468.2
## - smallfamily  1   1454.9 1472.9
## - thirdClass   1   1456.9 1474.9
## - Child        1   1470.8 1488.8
## - Sex          1   1604.9 1622.9
##
## Step:  AIC=1453.1
## Survived ~ Sex + smallfamily + notalone + Parch + Child + Age +
##     Fare + thirdClass
##
##               Df Deviance    AIC
## <none>             1435.1 1453.1
## - Age          1   1438.7 1454.7
## - Parch        1   1439.3 1455.3
## - Fare         1   1450.4 1466.4
## - thirdClass   1   1457.2 1473.2
## - Child        1   1470.8 1486.8
## - notalone     1   1484.9 1500.9
## - smallfamily  1   1487.6 1503.6
## - Sex          1   1605.8 1621.8
```

```
summary(model1)
```

```
##
## Call:
## glm(formula = Survived ~ Sex + smallfamily + notalone + Parch +
##     Child + Age + Fare + thirdClass, family = "binomial", data = data3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7267  -0.9358   0.4093   0.9574   2.5887
```

```
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.358436   0.280357   8.412  < 2e-16 ***
## Sexmale      -1.998675   0.169093 -11.820  < 2e-16 ***
## smallfamily1  2.722453   0.408331   6.667 2.61e-11 ***
## notaloneTRUE -3.014996   0.460590  -6.546 5.91e-11 ***
## Parch         0.243947   0.119219   2.046 0.040736 *
## ChildTRUE     2.411606   0.440150   5.479 4.28e-08 ***
## Age          -0.011382   0.005993  -1.899 0.057528 .
## Fare          0.007666   0.002207   3.474 0.000513 ***
## thirdClass1  -0.742490   0.159400  -4.658 3.19e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1779.4  on 1307  degrees of freedom
## Residual deviance: 1435.1  on 1299  degrees of freedom
##   (1 observation deleted due to missingness)
## AIC: 1453.1
##
## Number of Fisher Scoring iterations: 5
```

```
exp(cbind(OR = coef(model1), confint(model1))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                       OR      2.5 %     97.5 %
## (Intercept)  10.57440452 6.15141385 18.4772524
## Sexmale       0.13551470 0.09655766  0.1875055
## smallfamily1 15.21760338 7.00032736 34.8836033
## notaloneTRUE  0.04904605 0.01930411  0.1180622
## Parch         1.27627708 1.01015334  1.6164572
## ChildTRUE    11.15185486 4.84722326 27.3582042
## Age           0.98868223 0.97707637  1.0003247
## Fare          1.00769519 1.00359002  1.0123018
## thirdClass1   0.47592752 0.34764674  0.6496234
```

```
model2 <- (step(glm(Survived ~ Sex+Pclass+smallfamily+notalone+Child+Age+Fare, family="binomial", data=
```

```
## Start:  AIC=735.26
## Survived ~ Sex + Pclass + smallfamily + notalone + Child + Age +
##     Fare
##
##               Df Deviance    AIC
## <none>             717.26 735.26
## - Fare         1   720.16 736.16
## - Age          1   723.00 739.00
## - Pclass       2   761.69 775.69
## - Child        1   762.44 778.44
## - smallfamily  1   769.45 785.45
## - notalone     1   771.52 787.52
## - Sex          1   981.38 997.38
```

```r
summary(model2)
```

```
##
## Call:
## glm(formula = Survived ~ Sex + Pclass + smallfamily + notalone +
##     Child + Age + Fare, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7582  -0.5538  -0.3924   0.5674   2.3918
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.427813   0.500811   6.845 7.67e-12 ***
## Sexmale      -3.185160   0.227752 -13.985  < 2e-16 ***
## Pclass2      -1.166416   0.317228  -3.677 0.000236 ***
## Pclass3      -2.125493   0.324109  -6.558 5.46e-11 ***
## smallfamily1  2.921151   0.476951   6.125 9.09e-10 ***
## notaloneTRUE -3.146940   0.507408  -6.202 5.58e-10 ***
## ChildTRUE     3.607242   0.582949   6.188 6.10e-10 ***
## Age          -0.021146   0.008961  -2.360 0.018288 *
## Fare          0.004102   0.002592   1.582 0.113538
## ---
## 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:  717.26  on 882  degrees of freedom
## AIC: 735.26
##
## Number of Fisher Scoring iterations: 5
```

```r
exp(cbind(OR = coef(model1), confint(model1))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                       OR      2.5 %     97.5 %
## (Intercept)  10.57440452 6.15141385 18.4772524
## Sexmale       0.13551470 0.09655766  0.1875055
## smallfamily1 15.21760338 7.00032736 34.8836033
## notaloneTRUE  0.04904605 0.01930411  0.1180622
## Parch         1.27627708 1.01015334  1.6164572
## ChildTRUE    11.15185486 4.84722326 27.3582042
## Age           0.98868223 0.97707637  1.0003247
## Fare          1.00769519 1.00359002  1.0123018
## thirdClass1   0.47592752 0.34764674  0.6496234
```

```r
model3 <- glm(Survived ~ Sex+Pclass+smallfamily+notalone+Child, family="binomial", data= train)
summary(model3)
```

```
##
## Call:
## glm(formula = Survived ~ Sex + Pclass + smallfamily + notalone +
##     Child, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4223  -0.5853  -0.4006   0.5829   2.2940
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.8790     0.2886   9.975  < 2e-16 ***
## Sexmale       -3.2902     0.2246 -14.649  < 2e-16 ***
## Pclass2       -1.1924     0.2722  -4.380 1.19e-05 ***
## Pclass3       -2.0711     0.2397  -8.639  < 2e-16 ***
## smallfamily1   2.6885     0.4409   6.097 1.08e-09 ***
## notaloneTRUE  -2.7625     0.4551  -6.070 1.28e-09 ***
## ChildTRUE      3.9591     0.5339   7.416 1.21e-13 ***
## ---
## 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:  727.05  on 884  degrees of freedom
## AIC: 741.05
##
## Number of Fisher Scoring iterations: 5
```

```
exp(cbind(OR = coef(model3), confint(model3))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                       OR       2.5 %       97.5 %
## (Intercept)  17.79594777 10.25959739  31.82652649
## Sexmale       0.03724506  0.02367239   0.05717013
## Pclass2       0.30348148  0.17661321   0.51418506
## Pclass3       0.12604883  0.07808756   0.20013173
## smallfamily1 14.70915263  6.46380786  36.89462655
## notaloneTRUE  0.06313329  0.02447978   0.14755872
## ChildTRUE    52.40895431 19.21329348 157.58077575
```

‘‘‘

```
model4 <- glm(Survived ~ Sex+Pclass+smallfamily+notalone+Child+Age+Fare, family="binomial", data= train)
summary(model4)
```

```
##
## Call:
## glm(formula = Survived ~ Sex + Pclass + smallfamily + notalone +
##     Child + Age + Fare, family = "binomial", data = train)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7582  -0.5538  -0.3924   0.5674   2.3918
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.427813   0.500811   6.845 7.67e-12 ***
## Sexmale      -3.185160   0.227752 -13.985  < 2e-16 ***
## Pclass2      -1.166416   0.317228  -3.677 0.000236 ***
## Pclass3      -2.125493   0.324109  -6.558 5.46e-11 ***
## smallfamily1  2.921151   0.476951   6.125 9.09e-10 ***
## notaloneTRUE -3.146940   0.507408  -6.202 5.58e-10 ***
## ChildTRUE     3.607242   0.582949   6.188 6.10e-10 ***
## Age          -0.021146   0.008961  -2.360 0.018288 *
## Fare          0.004102   0.002592   1.582 0.113538
## ---
## 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:  717.26  on 882  degrees of freedom
## AIC: 735.26
##
## Number of Fisher Scoring iterations: 5
```

```
exp(cbind(OR = coef(model4), confint(model4))) # odds ratios and 95% CI
```

```
## Waiting for profiling to be done...
```

```
##                        OR       2.5 %       97.5 %
## (Intercept)  30.80919780 11.69500463  83.55641803
## Sexmale       0.04137161  0.02613718   0.06390783
## Pclass2       0.31148131  0.16653805   0.57880136
## Pclass3       0.11937411  0.06297058   0.22508085
## smallfamily1 18.56264569  7.69127658  50.60253508
## notaloneTRUE  0.04298344  0.01478503   0.10966445
## ChildTRUE    36.86422479 12.25103453 122.09980831
## Age           0.97907641  0.96177099   0.99620500
## Fare          1.00411004  0.99941455   1.00983288
```

Classification statistics and AUROC analysis

```
library(caret)
```

Load caret and pROC packages

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

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
train$SurvivedYhat <- predict(model4, type = "response")  # generate yhat values on train df
train$SurvivedYhat <- ifelse(train$SurvivedYhat > 0.5, 1.0, 0.0)  # set binary prediction threshold
confusionMatrix(train$Survived,train$SurvivedYhat)  # run confusionMatrix to assess accuracy
```

Model performance

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 485  64
##          1  89 253
##
##                Accuracy : 0.8283
##                  95% CI : (0.8019, 0.8525)
##     No Information Rate : 0.6442
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6319
##  Mcnemar's Test P-Value : 0.05235
##
##             Sensitivity : 0.8449
##             Specificity : 0.7981
##          Pos Pred Value : 0.8834
##          Neg Pred Value : 0.7398
##              Prevalence : 0.6442
##          Detection Rate : 0.5443
##    Detection Prevalence : 0.6162
##       Balanced Accuracy : 0.8215
##
##        'Positive' Class : 0
##
```

```
auc(roc(train$Survived,train$SurvivedYhat))  # calculate AUROC curve
```

```
## Area under the curve: 0.8116
```

```
test$Survived <- predict(model4, newdata = test, type = "response")
test$Survived <- ifelse(test$Survived > 0.5, 1.0, 0.0)  # set binary prediction threshold

testSubmission <- data.frame(cbind(test$PassengerId, test$Survived))
colnames(testSubmission) <- c("PassengerId", "Survived")
```

Generate predicted values in test data for best model "model4"

```
# save csv file for submission
write.csv(testSubmission, "Submissionlogmodel4.csv", row.names = FALSE)
```

write csv file for submission

RESULTS: model3 (Sex+Pclass+smallfamily+notalone+Child); accuracy = 0.78947 placing 1547/3911 entries
model4 (Sex+Pclass+smallfamily+notalone+Child+Age+Fare); accuracy = 0.7790, worse than model3

Recursive Partitioning Models

```
library(rpart)
library(rattle)
```

Load necessary packages

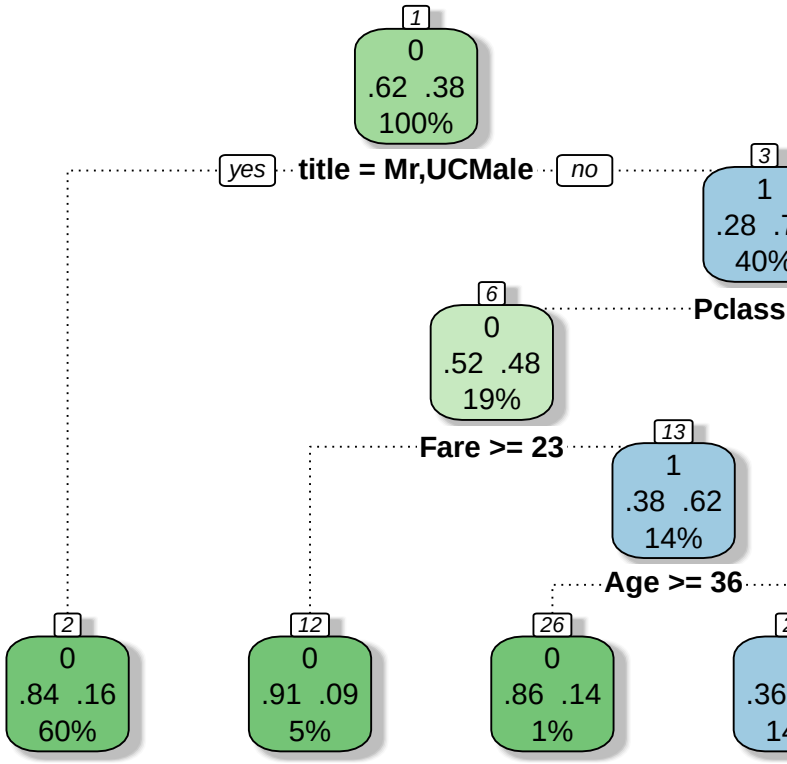
```
## Rattle: A free graphical interface for data mining with R.
## Version 4.0.5 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(rpart.plot)
library(RColorBrewer)
```

```
rpart4 <- rpart(Survived ~ Sex+Pclass+smallfamily+notalone+Child+Age+Fare+title, data = train, method =
```

Build the decision tree

```
fancyRpartPlot(rpart4)
```



Visualize the decision tree using rpart.plot

```
my_prediction <- predict(rpart4, test, type = "class")
```

Make prediction using the test set

```
my_solution <- data.frame(PassengerId = test$PassengerId, Survived = my_prediction)
```

Create a data frame with two columns for submission to Kaggle: PassengerId & Survived.

```
nrow(my_solution)
```

Check that my_solution has 418 entries

```
## [1] 418
```

Write csv file for submission write.csv(my_solution, file = "rpart4.csv", row.names = FALSE)

RESULTS: rpart2: (Sex+Pclass+smallfamily+notalone+Child) accuracy = 0.79426 placing 1425/3938

rpart3: (Sex+Pclass+smallfamily+notalone+Child+Age+Fare) accuracy = 0.79904 placing 1118/3938

rpart4: (Sex+Pclass+smallfamily+notalone+Child+Age+Fare+title) adding title to model omits:Sex+smallfamily+notalone+Child, accuracy = 0.77990 no improvement

Random Forest Models

```
library(randomForest)
```

Load randomForest package

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
set.seed(123)
```

Set seed for reproducibility

```
rf4 <- randomForest(Survived ~ (Sex+Pclass+smallfamily+notalone+Child), data = train, ntree = 1000, imp

round(importance(rf4), 1)
```

Apply the Random Forest Algorithm and check variable importance

```
##                0    1 MeanDecreaseAccuracy MeanDecreaseGini
## Sex         69.6 97.5                91.3            100.4
## Pclass      27.9 25.8                29.9             33.4
## smallfamily  3.4 23.8                24.0             15.4
## notalone     9.6 16.8                23.3              6.5
## Child       15.2 39.1                37.6             11.5
```

```
my_prediction<- predict(rf4, newdata=test)
```

Make a prediction using the test set

```
my_solution <- data.frame(PassengerId = test$PassengerId, Survived = my_prediction)
```

Create a data frame with two columns for submission to Kaggle: PassengerId & Survived.

```
nrow(my_solution)
```

Check that my_solution has 418 entries

```
## [1] 418
```

```
write.csv(my_solution, file = "rf1.csv", row.names = FALSE)
```

Write your solution to a csv file with the name my_solution.csv

RESULTS: rf2: (Sex+Pclass+smallfamily+notalone+Child+Age+Fare) accuracy = 0.80383 placing 911/3942 (Best overall model)

rf3: (Sex+Pclass+smallfamily+notalone+Child) accuracy = 0.79426 no improvement

rf4: (Sex+Pclass+smallfamily+Age+Fare) accuracy = 0.79904 no improvement

CONCLUSIONS

Participating in the Kaggle Titanic Competion was very rewarding and quite easy, in part because of all the resources that Kaggle provides, including datasets, scripts, notebooks, the discussion forum and for this competition several tutorials for both python and R. R is excellent open source software for data manipulation, visualization and machine learning analytics and R Studio is a very efficient and powerful platform for running R. In particular the R Markdown reporting tool, used for this report, is an excellent and efficient means of creating reports with embeded code and graphics. The relatively new *dplyr* R package for data manipulation was an improvement for this task over standard R scripting, however did not work as expected in several instances. The results of the predicitve modeling were very similar with all three methods, although the two methods which create decision trees to determine classification, recursive partitioning and random forests, were 2-3% more accurate than logistic regression. The best model was created using random forests and incorporated the features *Sex+Pclass+smallfamily+notalone+Child+Age+Fare*. This model scored 0.80383 placing 911/3942. The majority of the time involved in this project was spent preparing the data for analysis. In summary, R is a great tool for machine learning predicitve analytics. The results presented here suggest that it is most important to carefully prepare the data, including engineering additional features, and the three methods used in this study performed similarly although the random forest approach was best.

Titanic project Resources

Kaggle Titanic homepage and tutorials www.kaggle.com/c/titanic
www.datacamp.com/courses/kaggle-tutorial-on-machine-learing-the-sinking-of-the-titanic
www.youtube.com/watch?v=32o0DnuRjfg
www.youtube.com/watch?v=u6sahb7Hmog
rstudio-pubs-static.s3.amazonaws.com/98715_fcd035c75a9b431a84efca8b091a185f.html

dplyr resources www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf
www.youtube.com/watch?v=jWjqLW-u3hc&feature=youtu.be
www.youtube.com/watch?v=2mh1PqfsXVI
groups.google.com/forum/#!topic/manipulatr/Z46zwYXNh0g
stackoverflow.com/questions/22850026/filtering-row-which-contains-a-certain-string-using-dplyr
stackoverflow.com/questions/13520515/command-to-remove-row-from-a-data-frame

Logistic regression resources cran.r-project.org/web/packages/glm2/glm2.pdf
www.kaggle.com/eyebervil/titanic/titanic-simple-logit-with-interaction
cran.r-project.org/web/packages/caret/vignettes/caret.pdf
cran.r-project.org/web/packages/caret/caret.pdf
cran.r-project.org/web/packages/pROC/pROC.pdf
stats.stackexchange.com/questions/87234/aic-values-and-their-use-in-stepwise-model-selection-for-a-simple-linear-regress

Recursive partitioning method resources cran.r-project.org/web/packages/rpart/rpart.pdf
cran.r-project.org/web/packages/rpart/vignettes/longintro.pdf
campus.datacamp.com/courses/kaggle-r-tutorial-on-machine-learning/chapter-2-from-icebergs-to-trees?ex=1

Random Forest method resources cran.r-project.org/web/packages/randomForest/randomForest.pdf
campus.datacamp.com/courses/kaggle-r-tutorial-on-machine-learning/chapter-3-improving-your-predictions-through-random-ex=1

Error message: *Error in randomForest.default(y = train$Survived, x = train[, c("Sex", : Can't have empty classes in y.*
Solution: stackoverflow.com/questions/13495041/random-forests-in-r-empty-classes-in-y-and-argument-legth-0
Error message: Error in predict.randomForest(rf2, newdata = test, type = "response") : No forest component in the object
stat.ethz.ch/pipermail/r-help/2008-June/164878.html

Error Message: Error in predict.randomForest(rf2, newdata = test, type = "response") : Can't predict unsupervised forest.
Solution: stackoverflow.com/questions/17217951/how-can-i-drop-unused-levels-from-a-data-frame

R Markdown resources rmarkdown.rstudio.com/
www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf
www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf

Error message: Error: attempt to use zero-length variable name
Solution: stackoverflow.com/questions/31296908/knitr-running-script-without-warnings