Analysing Titanic Data set

Ayush Bhargava

January 9, 2018

Table of content

Introduction

This is my attempt at analysing Titanic data set, which is a very famous data set at KAGGLE. In this analysis i will first explore the dataset, try to understand dependent and independent variable. Then i will deal with missing values in the data set and perform feature engineering to extract as much information from given features as possible.Thereafter i will present visualizations of each features to gather some insights about their distribution and finally build predictive model and evaluate their performance.

Data Source

Train and test data can be dowloaded from here:- https://www.kaggle.com/c/titanic/data Both train and test data sets are in standard csv format and has following features

Table 1: Features in data set

Variable	Definition	Key
Survived	Surival	0 = No, 1 = Yes
pclass	Passenger class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	sex	
Age	age in years	
sibps	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	ticket number	
fare	passenger fare	
cabin	cabin number	
embarked	Port of embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Data import

Loading Required Libraries

```
# data wrangling
library(tidyverse)
library(forcats)
library(stringr)

#data imputation
library(mice)

# data assessment/visualizations
library(tm)
library(data.table)
library(pander)
library(ggplot2)
library(scales)
library(grid)
library(gridExtra)
library(corrplot)
library(VIM)
library(knitr)
library(vcd)
library(caret)

# model
library(xgboost)
library(MLmetrics)
library('randomForest')
library('rpart')
library('rpart.plot')
library('car')
library('e1071')
```

Setting up working directory and getting data

```
setwd("C:/Users/ichbi/Desktop/kaggle/Titanic")
train <- read_csv('train.csv')
test <- read_csv('test.csv')
```

Here we are using read_csv as it is faster than read.csv and doesnot convert string as factor by default. Next we will sneak peak the train data set and do some preprocessing. we will convert some variables in to factor variable for ease of analysis

Train data preprocessing

```
#sneak peak
glimpse(train)
```

```
## Observations: 891
```

```
## Variables: 12
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ Survived    <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0,...
## $ Pclass      <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3,...
## $ Name        <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bra...
## $ Sex         <chr> "male", "female", "female", "female", "male", "mal...
## $ Age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, ...
## $ SibSp       <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4,...
## $ Parch       <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1,...
## $ Ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "1138...
## $ Fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, ...
## $ Cabin       <chr> NA, "C85", NA, "C123", NA, NA, "E46", NA, NA, NA, ...
## $ Embarked    <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S", "C", ...
```

```r
#converting in to factors
train <- train %>% setNames(tolower(names(.))) %>% mutate(survived=factor(survived, levels = c("0","1")

#looking at the summary
summary(train)
```

```
##   passengerid    survived pclass      name               sex
##  Min.   :  1.0   0:549    1:216   Length:891         female:314
##  1st Qu.:223.5   1:342    2:184   Class :character   male  :577
##  Median :446.0            3:491   Mode  :character
##  Mean   :446.0
##  3rd Qu.:668.5
##  Max.   :891.0
##
##       age            sibsp           parch            ticket
##  Min.   : 0.42   Min.   :0.000   Min.   :0.0000   Length:891
##  1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000   Class :character
##  Median :28.00   Median :0.000   Median :0.0000   Mode  :character
##  Mean   :29.70   Mean   :0.523   Mean   :0.3816
##  3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##  Max.   :80.00   Max.   :8.000   Max.   :6.0000
##  NA's   :177
##       fare           cabin             embarked       set
##  Min.   :  0.00   Length:891         C    :168    Length:891
##  1st Qu.:  7.91   Class :character   Q    : 77    Class :character
##  Median : 14.45   Mode  :character   S    :644    Mode  :character
##  Mean   : 32.20                      NA's:  2
##  3rd Qu.: 31.00
##  Max.   :512.33
##
```

```r
#percentage survived
surv_pct <- train %>% count(survived) %>% mutate(pct=n/sum(n)) %>% setNames(c('survived','count','percen
```

There is no survived column in the test set and our goal is to build a predictive model on train set and predict survival of passengers in the test set.

Test data preprocessing

```r
#sneak peak
glimpse(test)
```

```
## Observations: 418
## Variables: 11
## $ PassengerId <int> 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, ...
## $ Pclass      <int> 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 1, 1, 2, 1, 2, 2,...
## $ Name        <chr> "Kelly, Mr. James", "Wilkes, Mrs. James (Ellen Nee...
## $ Sex         <chr> "male", "female", "male", "male", "female", "male"...
## $ 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      <chr> "330911", "363272", "240276", "315154", "3101298",...
## $ Fare        <dbl> 7.8292, 7.0000, 9.6875, 8.6625, 12.2875, 9.2250, 7...
## $ Cabin       <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "B...
## $ Embarked    <chr> "Q", "S", "Q", "S", "S", "S", "Q", "S", "C", "S", ...
```

```r
#converting in to factors
test <- test %>% setNames(tolower(names(.))) %>% mutate(pclass=factor(pclass), sex=factor(sex),embarked=

#looking at the summary
summary(test)
```

```
##   passengerid     pclass      name               sex           age
##  Min.   : 892.0   1:107   Length:418         female:152   Min.   : 0.17
##  1st Qu.: 996.2   2: 93   Class :character   male  :266   1st Qu.:21.00
##  Median :1100.5   3:218   Mode  :character                Median :27.00
##  Mean   :1100.5                                           Mean   :30.27
##  3rd Qu.:1204.8                                           3rd Qu.:39.00
##  Max.   :1309.0                                           Max.   :76.00
##                                                           NA's   :86
##      sibsp            parch           ticket               fare
##  Min.   :0.0000   Min.   :0.0000   Length:418         Min.   :  0.000
##  1st Qu.:0.0000   1st Qu.:0.0000   Class :character   1st Qu.:  7.896
##  Median :0.0000   Median :0.0000   Mode  :character   Median : 14.454
##  Mean   :0.4474   Mean   :0.3923                      Mean   : 35.627
##  3rd Qu.:1.0000   3rd Qu.:0.0000                      3rd Qu.: 31.500
##  Max.   :8.0000   Max.   :9.0000                      Max.   :512.329
##                                                       NA's   :1
##     cabin             embarked     set
##  Length:418         C:102    Length:418
##  Class :character   Q: 46    Class :character
##  Mode  :character   S:270    Mode  :character
##
##
##
##
```

- For doing feature engineering and preprocessing for modeling and there after predicting, we will comine train and test set so that features are consistent amoung two data sets
- Just before building models we will again divide the train and test set

Merging test and train data sets

```r
#bind_row will row bind and will fill NA values in survived column for test set passengers
full <- bind_rows(train,test)
summary(full)
```

```
##   passengerid     survived    pclass       name                sex
##  Min.   :   1   0   :549   1:323   Length:1309        female:466
##  1st Qu.: 328   1   :342   2:277   Class :character   male  :843
##  Median : 655   NA's:418   3:709   Mode  :character
##  Mean   : 655
##  3rd Qu.: 982
##  Max.   :1309
##
##       age            sibsp            parch           ticket
##  Min.   : 0.17   Min.   :0.0000   Min.   :0.000   Length:1309
##  1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.000   Class :character
##  Median :28.00   Median :0.0000   Median :0.000   Mode  :character
##  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
##       fare            cabin            embarked       set
##  Min.   :  0.000   Length:1309        C   :270   Length:1309
##  1st Qu.:  7.896   Class :character   Q   :123   Class :character
##  Median : 14.454   Mode  :character   S   :914   Mode  :character
##  Mean   : 33.295                      NA's:  2
##  3rd Qu.: 31.275
##  Max.   :512.329
##  NA's   :1
```

```r
glimpse(full)
```

```
## Observations: 1,309
## Variables: 13
## $ passengerid <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ survived    <fctr> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0...
## $ pclass      <fctr> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3...
## $ name        <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bra...
## $ sex         <fctr> male, female, female, female, male, male, male, m...
## $ age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, ...
## $ sibsp       <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4,...
## $ parch       <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1,...
## $ ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "1138...
## $ fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, ...
## $ cabin       <chr> NA, "C85", NA, "C123", NA, NA, "E46", NA, NA, NA, ...
## $ embarked    <fctr> S, C, S, S, S, Q, S, S, S, C, S, S, S, S, S, S, Q...
## $ set         <chr> "train", "train", "train", "train", "train", "trai...
```

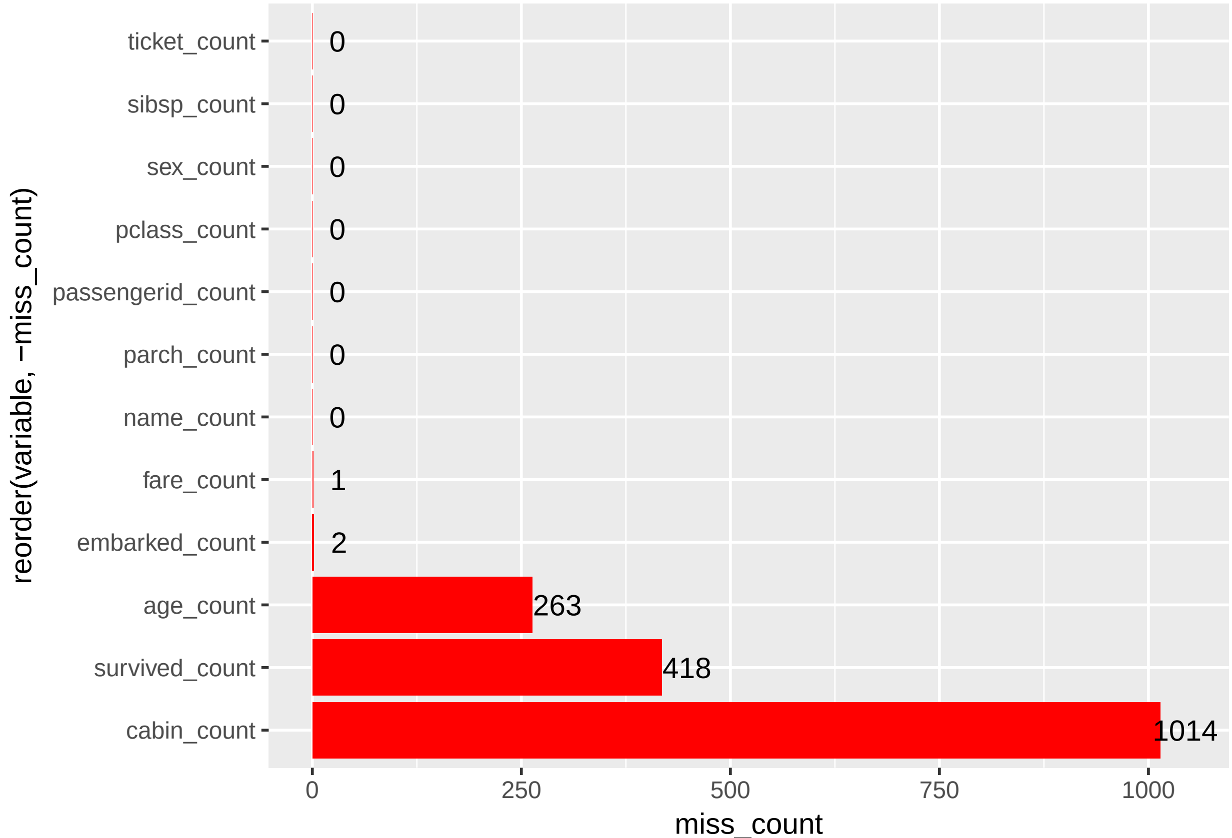
Missing Value analysis

Next we will look if there are any missing values in our data set. first we will build a table with count and percentage of missing values for every feature in our data set and later we will visualize the same data

```r
#Creating table of count of missing values in features of our full dataset
miss_values <- full %>% summarise_all(funs(count=sum(is.na(.)),percentage=sum(is.na(.))/n()))
miss_count <- miss_values[,1:12] %>% gather(variable, miss_count)
miss_count
```

```
## # A tibble: 12 x 2
##    variable          miss_count
##    <chr>                  <int>
##  1 passengerid_count          0
##  2 survived_count           418
##  3 pclass_count               0
##  4 name_count                 0
##  5 sex_count                  0
##  6 age_count                263
##  7 sibsp_count                0
##  8 parch_count                0
##  9 ticket_count               0
## 10 fare_count                 1
## 11 cabin_count             1014
## 12 embarked_count             2
```

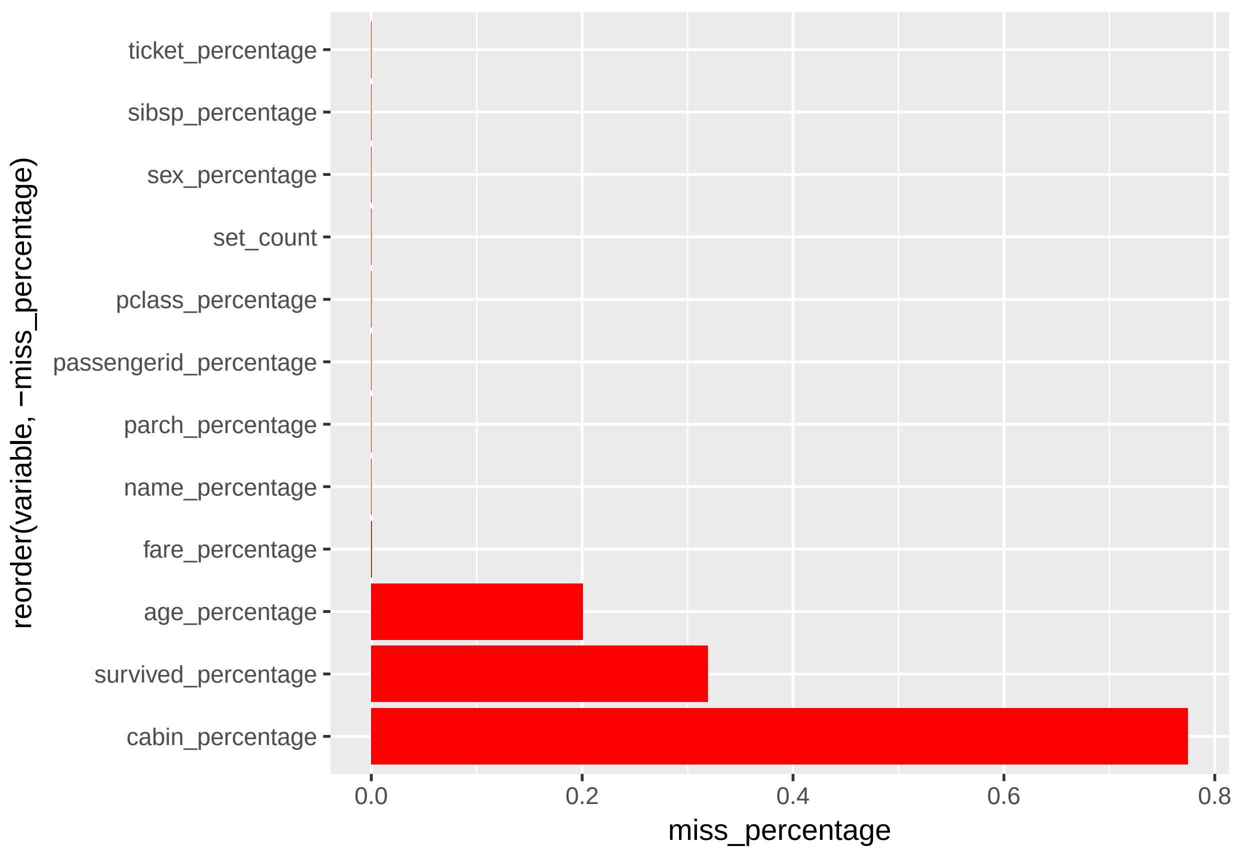
```r
#visualizing
miss_count %>% ggplot(aes(x=reorder(variable,-miss_count),y=miss_count)) + geom_bar(stat='identity',fil
```




```r
#Creating table of percentage of missing values in features of our full dataset
miss_percentage <- miss_values[,13:24] %>% gather(variable, miss_percentage)
miss_percentage
```

```
## # A tibble: 12 x 2
##    variable                miss_percentage
##    <chr>                             <dbl>
##  1 set_count                      0
##  2 passengerid_percentage         0
##  3 survived_percentage            0.319
##  4 pclass_percentage              0
##  5 name_percentage                0
##  6 sex_percentage                 0
##  7 age_percentage                 0.201
##  8 sibsp_percentage               0
##  9 parch_percentage               0
## 10 ticket_percentage              0
## 11 fare_percentage                0.000764
## 12 cabin_percentage               0.775
```

```r
#Visualizing
miss_percentage %>% ggplot(aes(x=reorder(variable,-miss_percentage),y=miss_percentage)) + geom_bar(stat=
```



- Missing value in "cabin" is greater than 80%, we can not do much about this feature
- We will impute values for other missing variables in the next section

Missing value Imputation

Embarkement :-

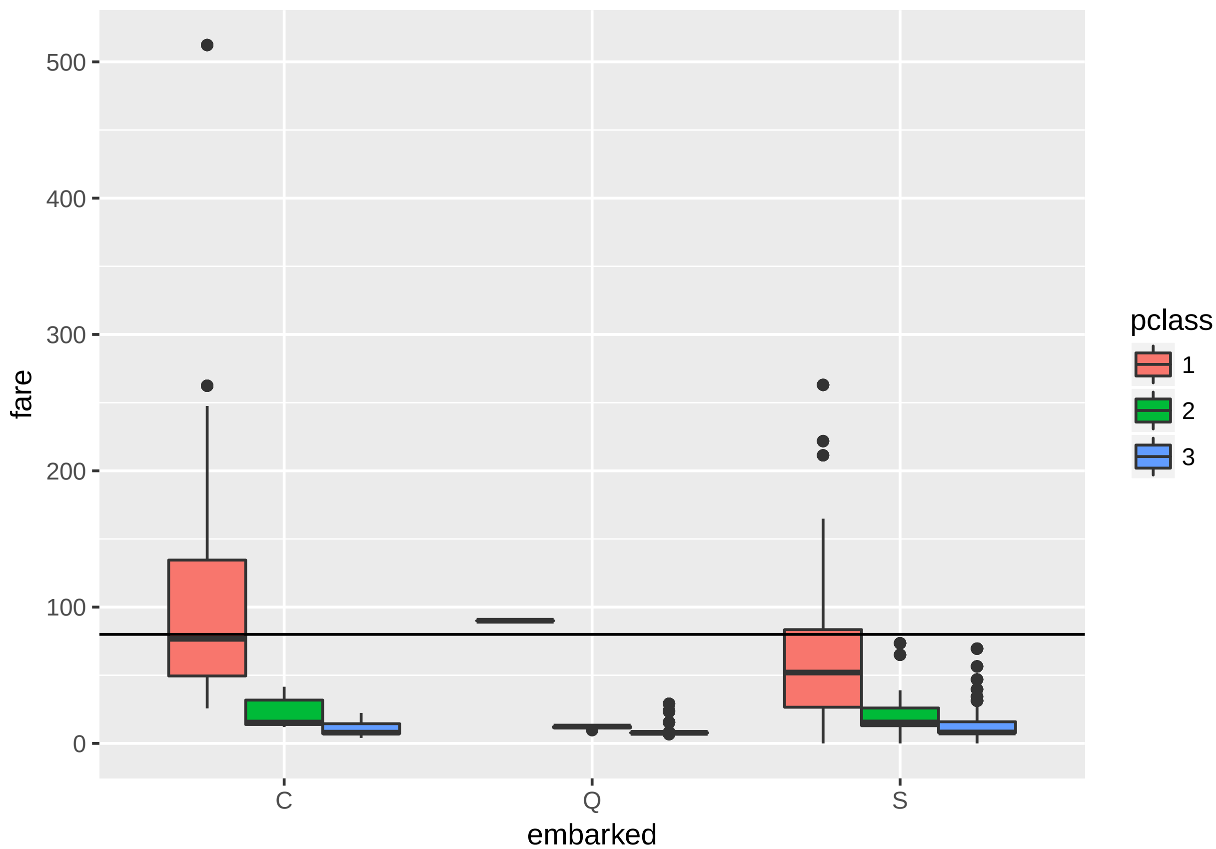
This variable could be related to fare and pclass variable, lets look at the whole feature space of these missing points

```
glimpse(full %>% filter(is.na(embarked)))
```

```
## Observations: 2
## Variables: 13
## $ passengerid <int> 62, 830
## $ survived    <fctr> 1, 1
## $ pclass      <fctr> 1, 1
## $ name        <chr> "Icard, Miss. Amelie", "Stone, Mrs. George Nelson ...
## $ sex         <fctr> female, female
## $ age         <dbl> 38, 62
## $ sibsp       <int> 0, 0
## $ parch       <int> 0, 0
## $ ticket      <chr> "113572", "113572"
## $ fare        <dbl> 80, 80
## $ cabin       <chr> "B28", "B28"
## $ embarked    <fctr> NA, NA
## $ set         <chr> "train", "train"
```

Now we know that these are 1st class females who paid 80 fare. lets look at relationship between pclass vs fare vs embarkemnt and see if we could decipher any useful information about emabarkment of these passengers

```
#subseting the non missing embarked
embark_full <- full %>% filter(!is.na(embarked))

#plotting
ggplot(embark_full, aes(x=embarked,y=fare, fill=pclass))+geom_boxplot()+geom_hline(yintercept = 80)
```



It is clearly visible in the plot that passenger of 1st class embarked from "C" paid a fare of 80 on average, thus it is safe to impute "c" for these missing embarked passengers

Imputing value and getting the imputed data set

```
full <- full %>% mutate(embarked=factor(ifelse(is.na(full$embarked),'C', as.character(embarked))))
```

Fare :-

lets look at the whole feature space of the missing point

```
glimpse(full %>% filter(is.na(fare)))
```

```
## Observations: 1
## Variables: 13
## $ passengerid <int> 1044
## $ survived    <fctr> NA
## $ pclass      <fctr> 3
## $ name        <chr> "Storey, Mr. Thomas"
## $ sex         <fctr> male
## $ age         <dbl> 60.5
## $ sibsp       <int> 0
## $ parch       <int> 0
## $ ticket      <chr> "3701"
## $ fare        <dbl> NA
## $ cabin       <chr> NA
```

```
## $ embarked    <fctr> S
## $ set         <chr> "test"
```

embarked = s, pclass=3, since only one value is missing lets plug with the median in this category

Imputing value and getting the imputed data set

```
impte_value <- full %>% group_by(embarked, pclass) %>% summarise(mean_fare=median(fare,na.rm=T)) %>% fil
full <- full %>% mutate(fare=ifelse(is.na(full$fare),impte_value,fare))
```

Age :-

We saw earlier there are many missing points in age feature, thus we will use mice imputation, which is a neat implementation of data imputation. more can be read about mice imputation here

```
#creating dataset for mice imputation
  mice_data <- full[,!names(full) %in% c('passengerid','name','ticket','cabin','survived')]
 set.seed(120)
 imputed_data <- mice(mice_data, method = 'rf')
```

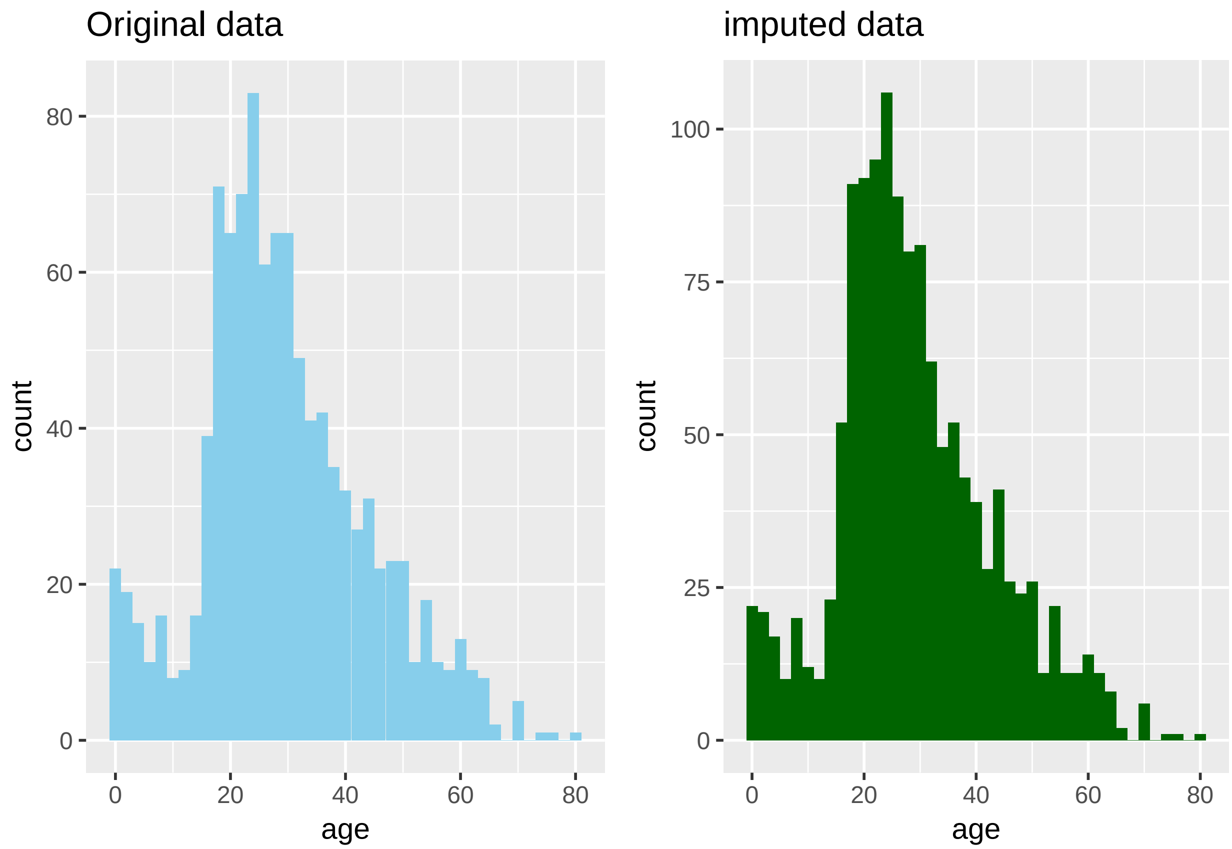
```
##
##  iter imp variable
##   1   1  age
##   1   2  age
##   1   3  age
##   1   4  age
##   1   5  age
##   2   1  age
##   2   2  age
##   2   3  age
##   2   4  age
##   2   5  age
##   3   1  age
##   3   2  age
##   3   3  age
##   3   4  age
##   3   5  age
##   4   1  age
##   4   2  age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

```
imputed_data <- mice::complete(imputed_data)
```

lets compare age distribution in original and imputed data set

```
#comparing age distribution in original and imputed data
 par_1 <- ggplot(full, aes(x=age)) + geom_histogram(fill='skyblue', binwidth = 2) + labs(title="Origina
```

```
 par_2 <- ggplot(imputed_data, aes(x=age)) + geom_histogram(fill='darkgreen', binwidth = 2) + labs(titl
grid.arrange(par_1, par_2, ncol=2)
```



Age distribution in imputed dataset seems is in line with the original dataset, lets replace age in original data set with imputed value

```
#finally replacing age variable in full data set
full$age <- imputed_data$age
```

Feature Engineering

There are features in data set which contains more information that could potentailly reduce bais in our prediction models. This information is not implicitly visible and has to be extracted. This process is feature engineering.

Creating Title

From the name variable we can extract title of a passenger to add more information for training our model

```
full$title <- gsub("^.*,[[:blank:]](.*?)\\..*$", "\\1", full$name)
table(full$title)
```

```
##
##        Capt          Col          Don         Dona           Dr
##           1            4            1            1            8
##    Jonkheer         Lady        Major       Master         Miss
```

```
##            1            1            2           61          260
##         Mlle          Mme           Mr          Mrs           Ms
##            2            1          757          197            2
##          Rev          Sir the Countess
##            8            1            1
```

```r
rare <- c('the Countess','Capt', 'Col', 'Don',
            'Dr', 'Major', 'Rev', 'Sir', 'Jonkheer')
full$title[full$title == 'Mlle'] <- 'Miss'
full$title[full$title == 'Ms']   <- 'Miss'
full$title[full$title == 'Mme']  <- 'Mrs'
full$title[full$title == 'Lady'] <- 'Miss'
full$title[full$title == 'Dona'] <- 'Miss'
full$title[full$title %in% rare] <- "rare"
table(full$title)
```

```
##
## Master   Miss     Mr    Mrs   rare
##     61    266    757    198     27
```

```r
full$title <- factor(full$title)
```

We have grabbed the title from the name and categorised them on the basis of occurance in the data set to avoid outliers. we will explore the surivival on the basis of title in the next section

Family size

We will explore Whether size of family matter in the survival. we will create a family size variable by adding no of siblings (sibps) and no of parents variable(parch)

```r
full$familysize <- full$sibsp +full$parch +1
full$FamilySized[full$familysize == 1] <- 'Single'
full$FamilySized[full$familysize < 5 & full$familysize >= 2] <- 'Small'
full$FamilySized[full$familysize >= 5] <- 'Big'
full$FamilySized=as.factor(full$FamilySized)
```

To avoid outliers we have created categories of family size

Age groups

To analyse age variable lets categorise this variable too

```r
full <- full %>% mutate(agegroup=case_when(age<13 ~"children", (age >= 13 & age< 18) ~ "adolsents", (age
```

Ticket groups

Ticket variable does not seem to provide useful information, to use this variable we will do some text mining and find pattern in the ticket number and finally creating groups based on pattern.

It was found that there are ticket numbers which are numeric with 3,4,5,6,7 digits and than there are ticket numbers which starts with certain letters, we have created a variable ticket group based on these observed pattern.

```r
#Ticket
full <- full %>% mutate(ticketgroup= tolower(ticket)) %>% mutate(ticketgroup=removePunctuation(ticketgro
```

Exploratory Analysis

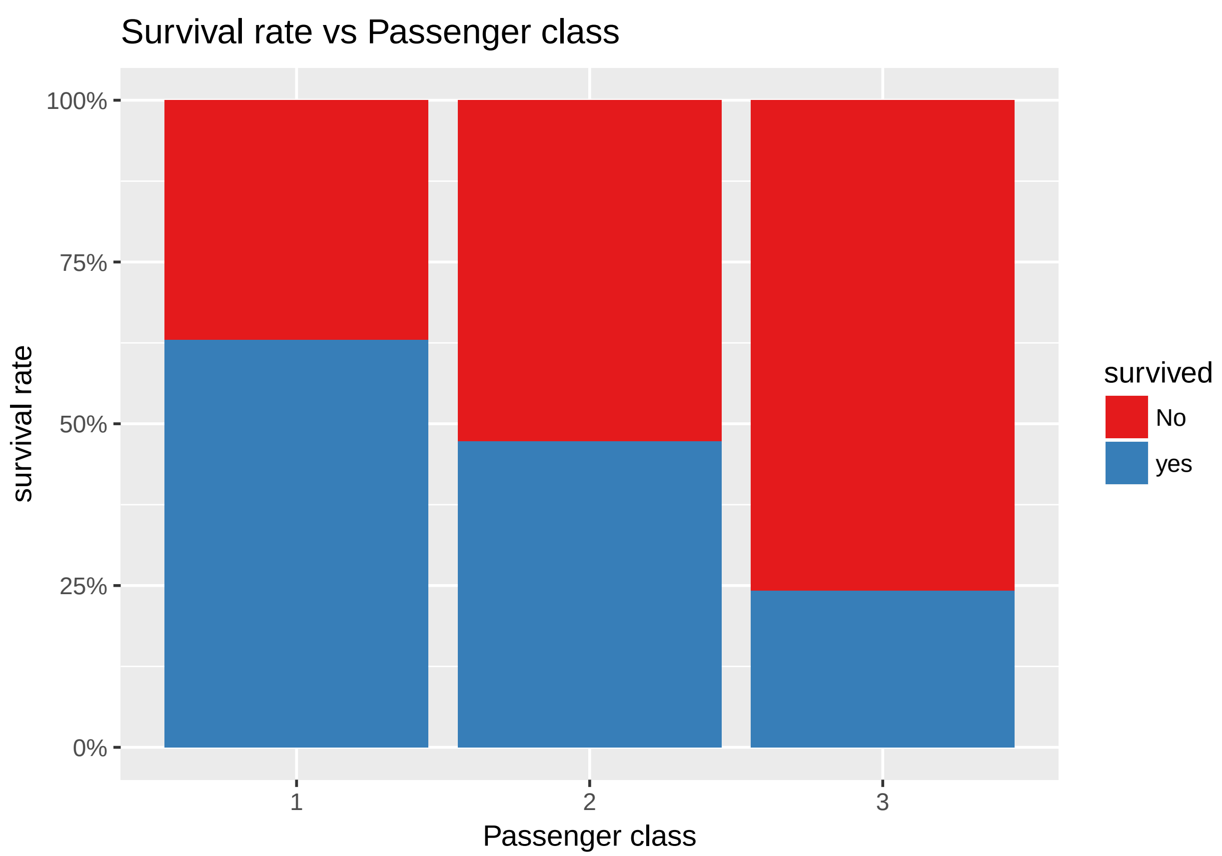
survival vs other relevant variables

Pclass

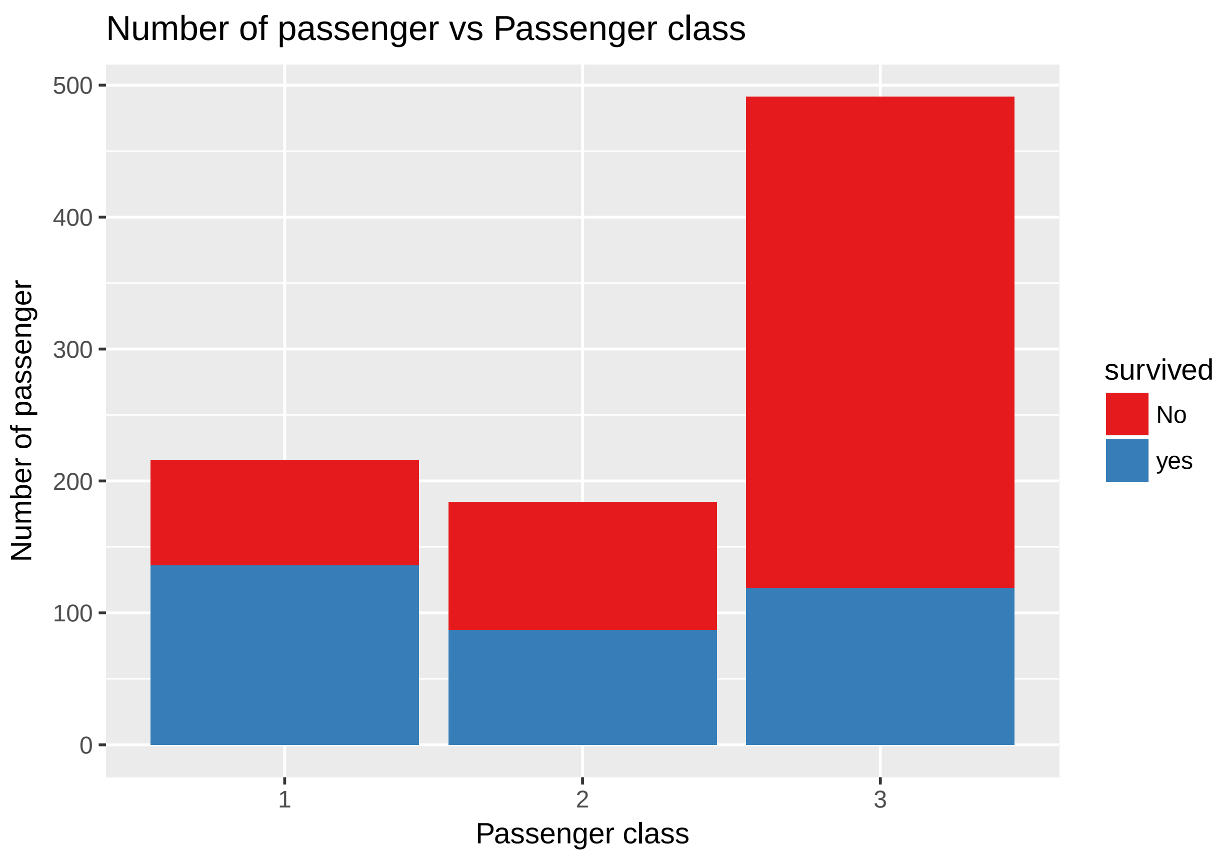
```
summary <- full %>% filter(set=='train') %>% group_by(pclass) %>% summarise(passenger=n(),survived=sum(a
kable(summary)
```

pclass	passenger	survived	survival_rate
1	216	136	63
2	184	87	47
3	491	119	24

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

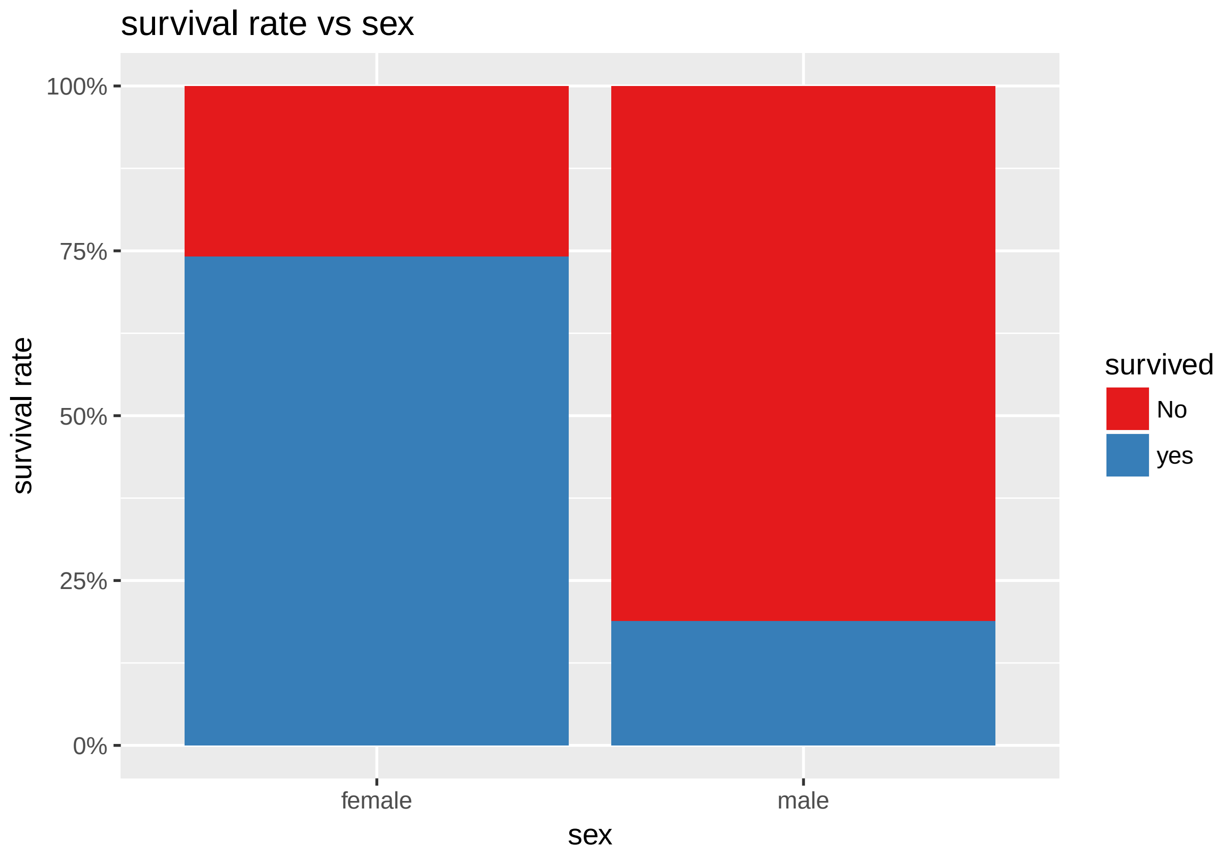


Sex

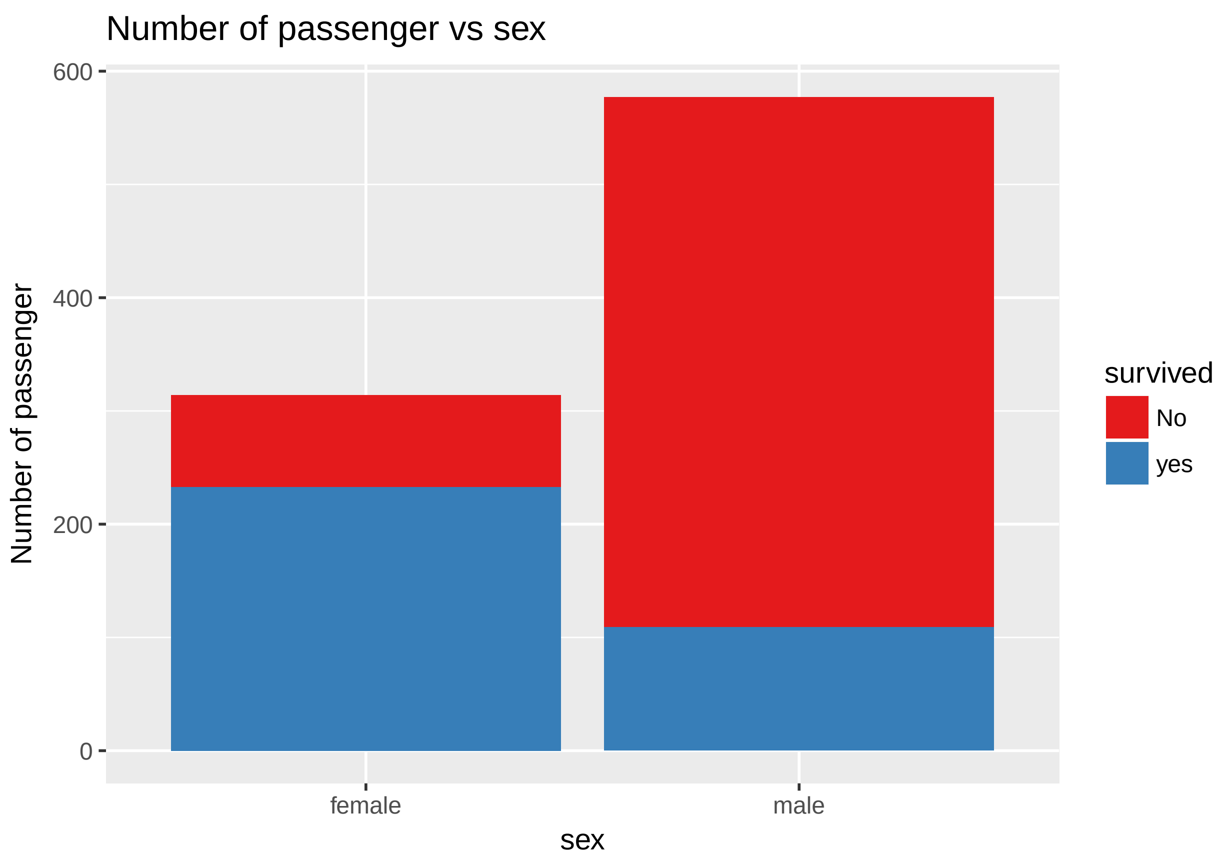
```
summary <- full %>% filter(set=='train') %>% group_by(sex) %>% summarise(passenger=n(),survived=sum(as.n
kable(summary)
```

sex	passenger	survived	survival_rate
female	314	233	74
male	577	109	19

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

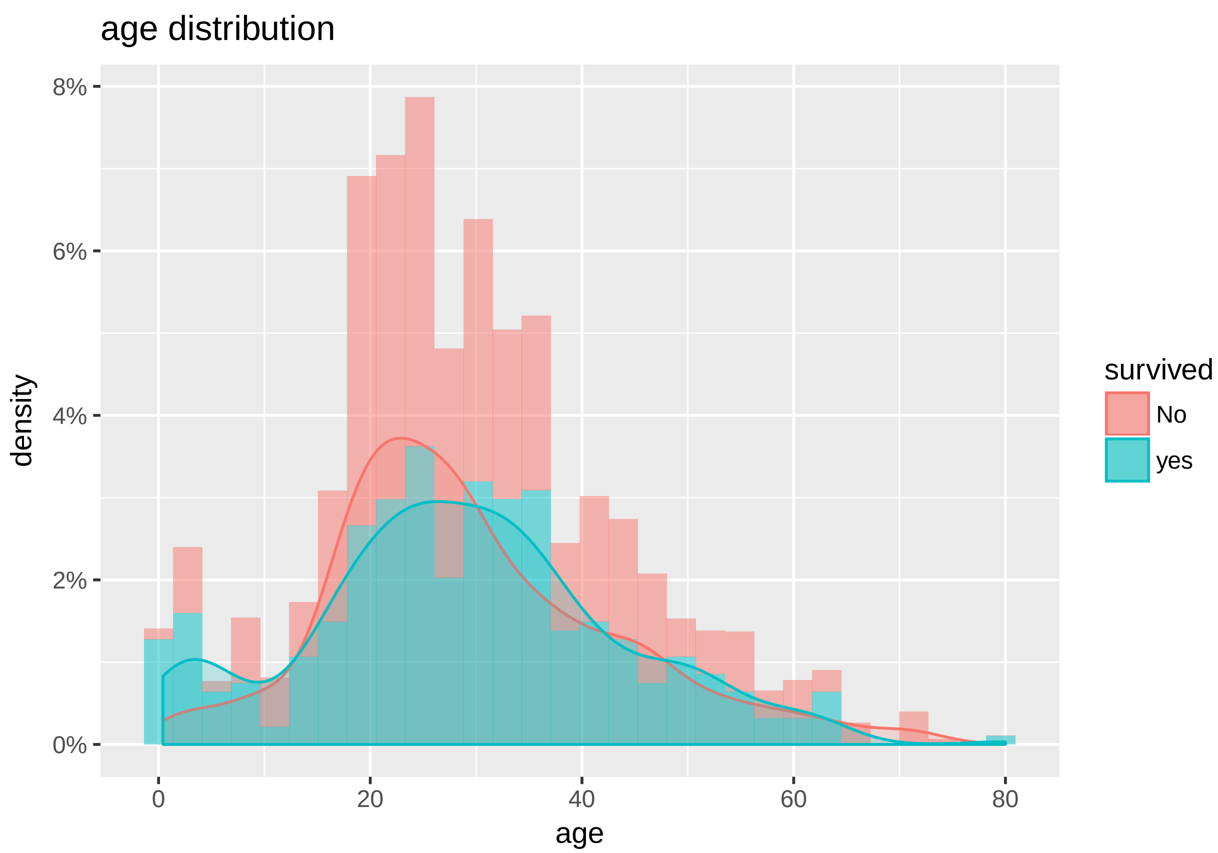



```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

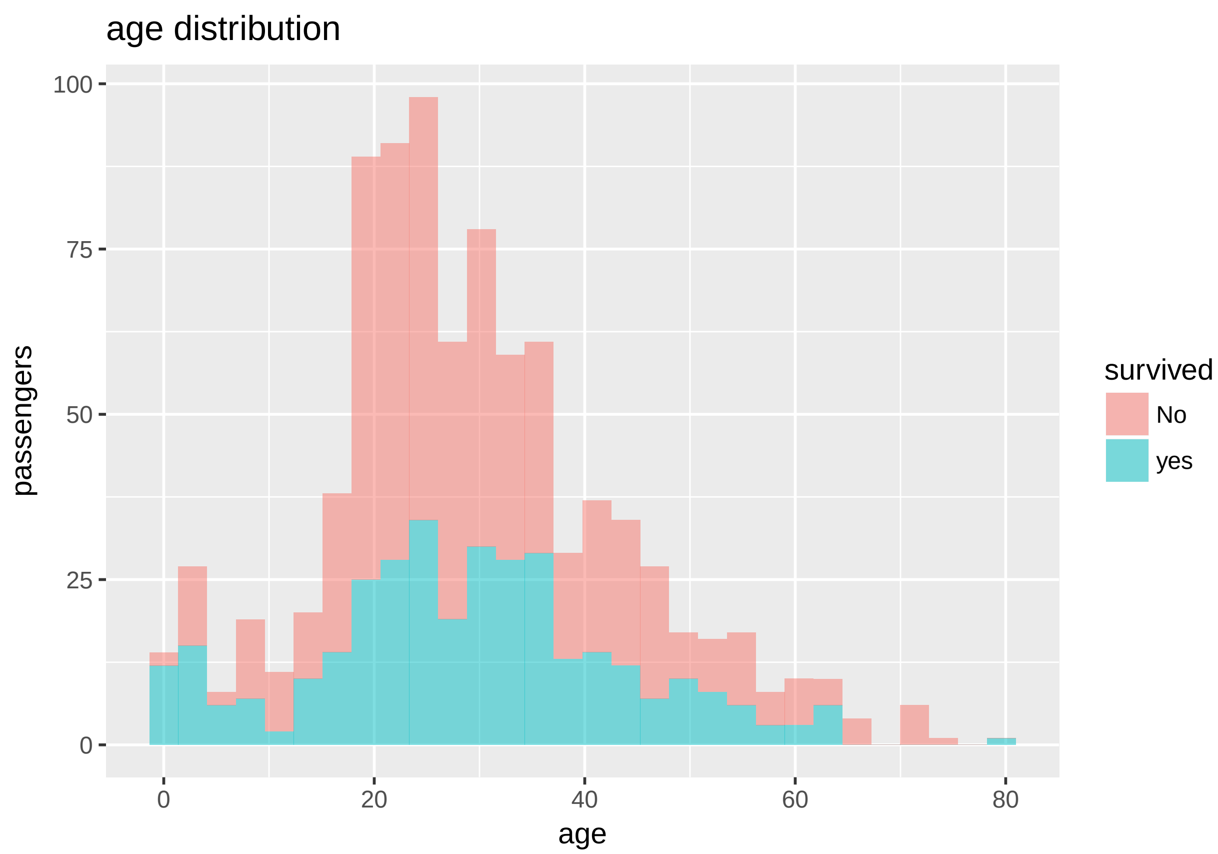


Age

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

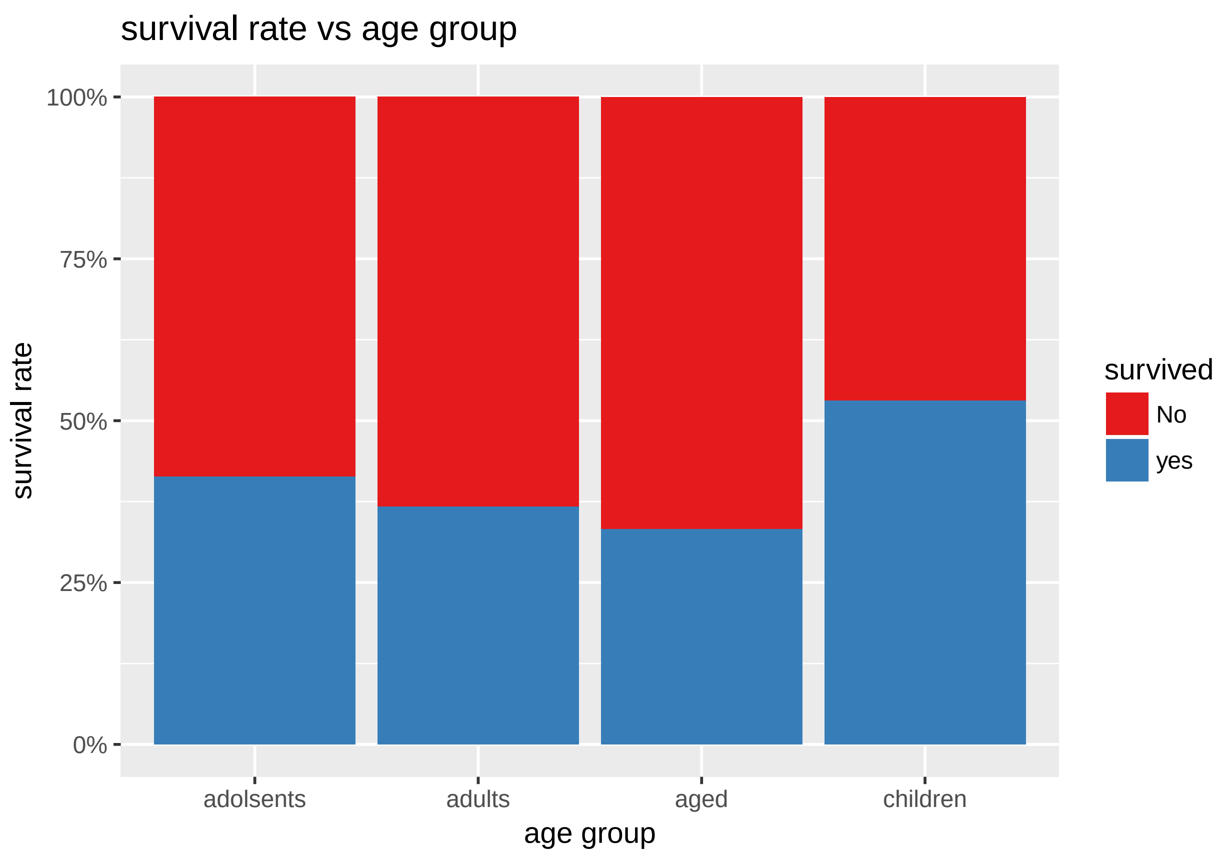


Agegroup

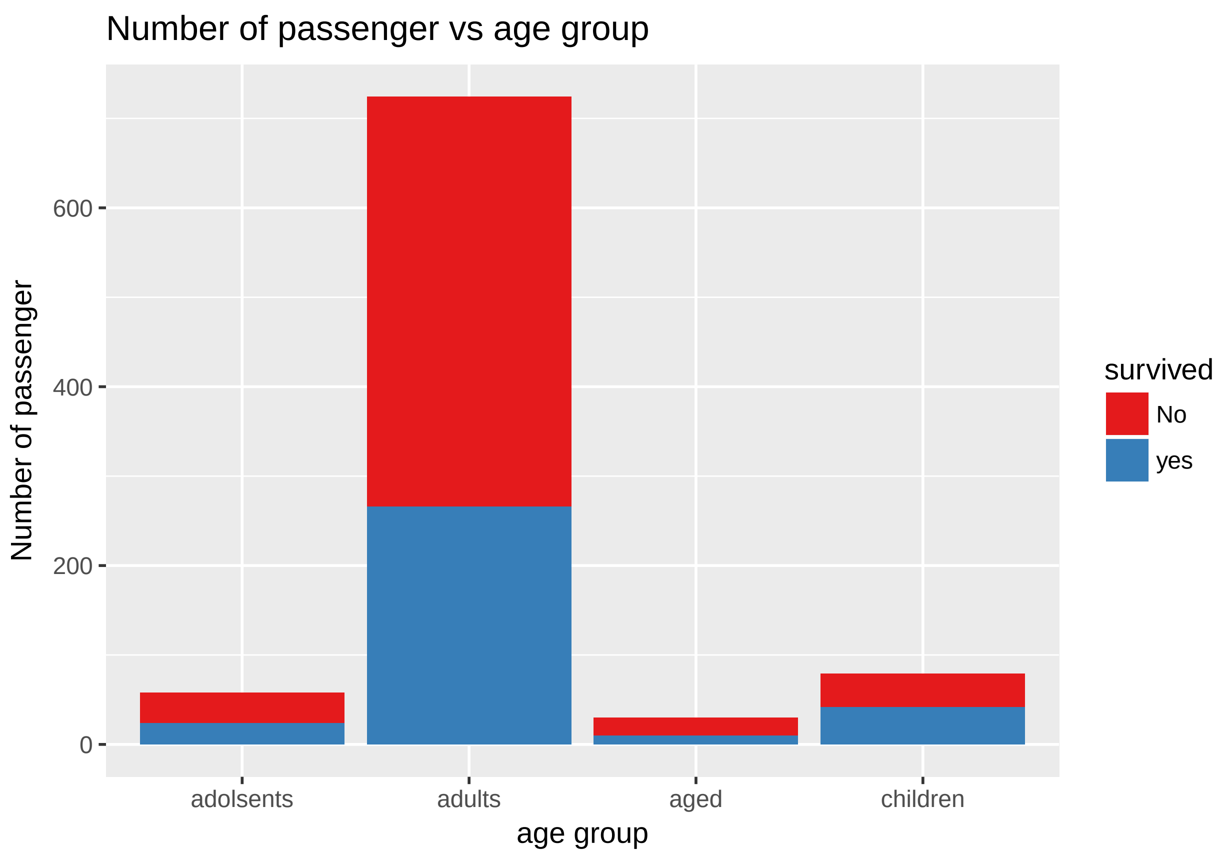
```
summary <- full %>% filter(set=='train') %>% group_by(agegroup) %>% summarise(passenger=n(),survived=su
kable(summary)
```

agegroup	passenger	survived	survival_rate
adolsents	58	24	41
adults	724	266	37
aged	30	10	33
children	79	42	53

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

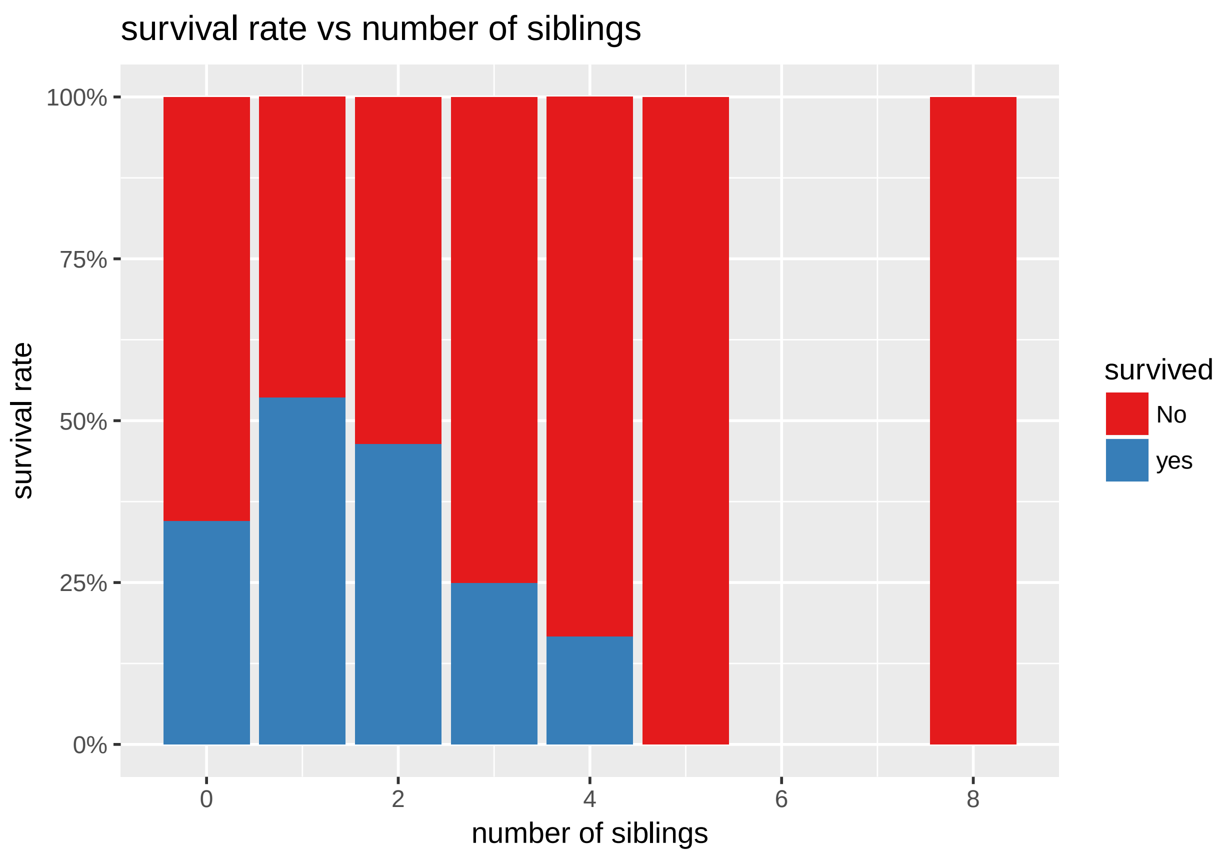


Sibsp

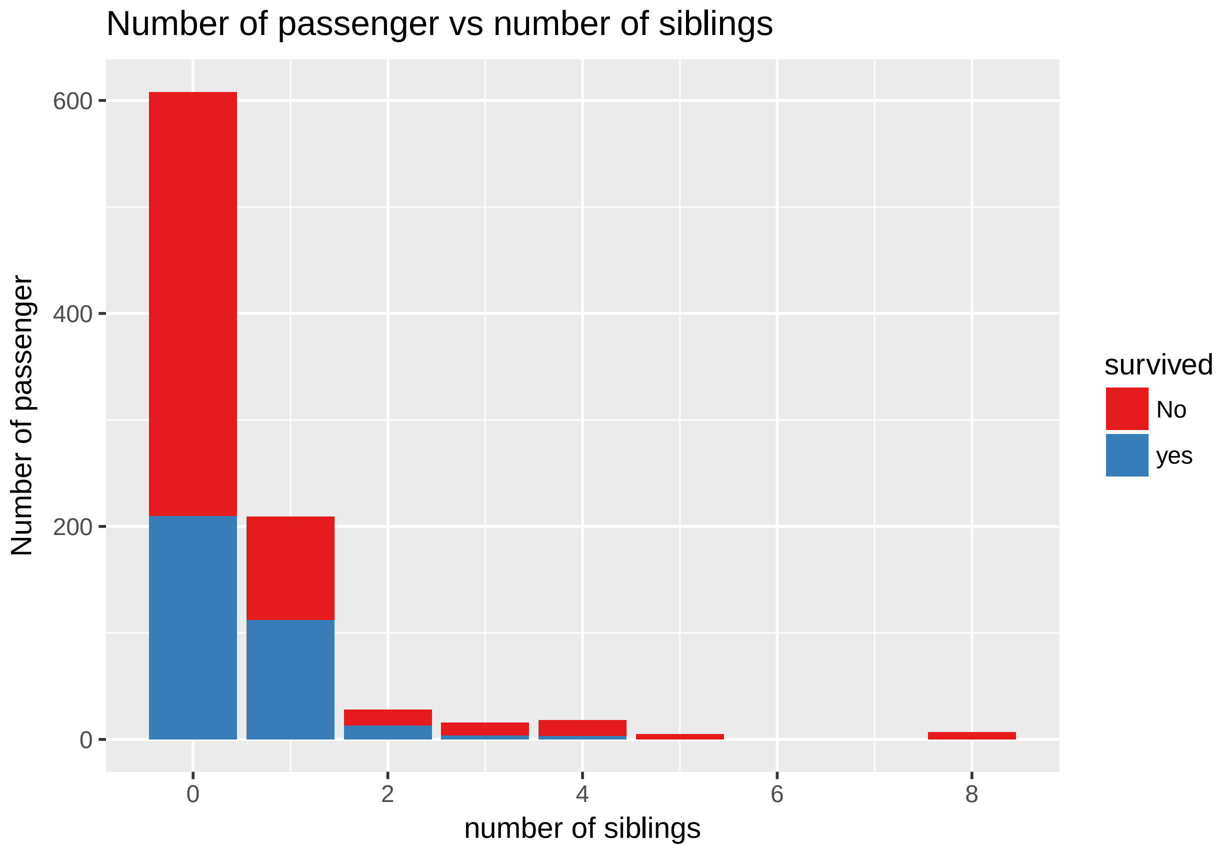
```
summary <- full %>% filter(set=='train') %>% group_by(sibsp) %>% summarise(passenger=n(),survived=sum(a
kable(summary)
```

sibsp	passenger	survived	survival_rate
0	608	210	35
1	209	112	54
2	28	13	46
3	16	4	25
4	18	3	17
5	5	0	0
8	7	0	0

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tr
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

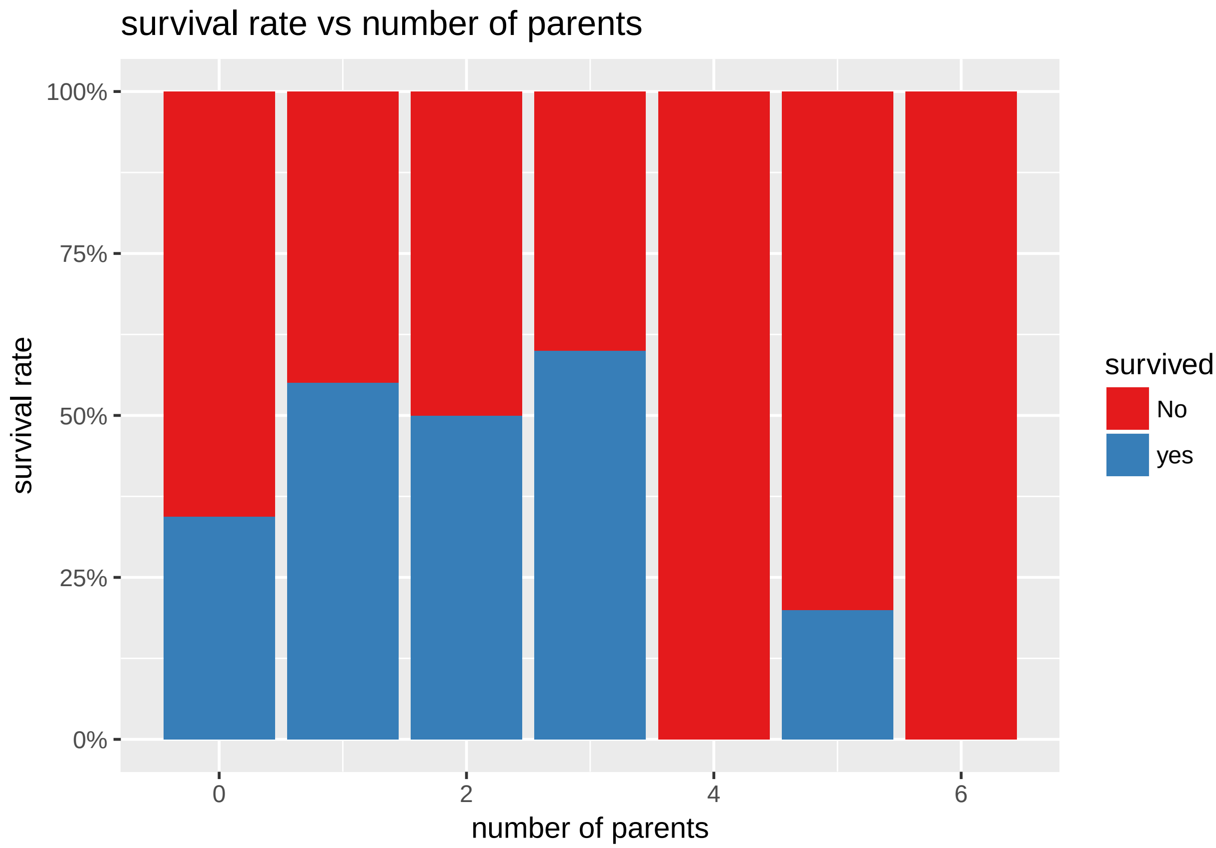


Parch

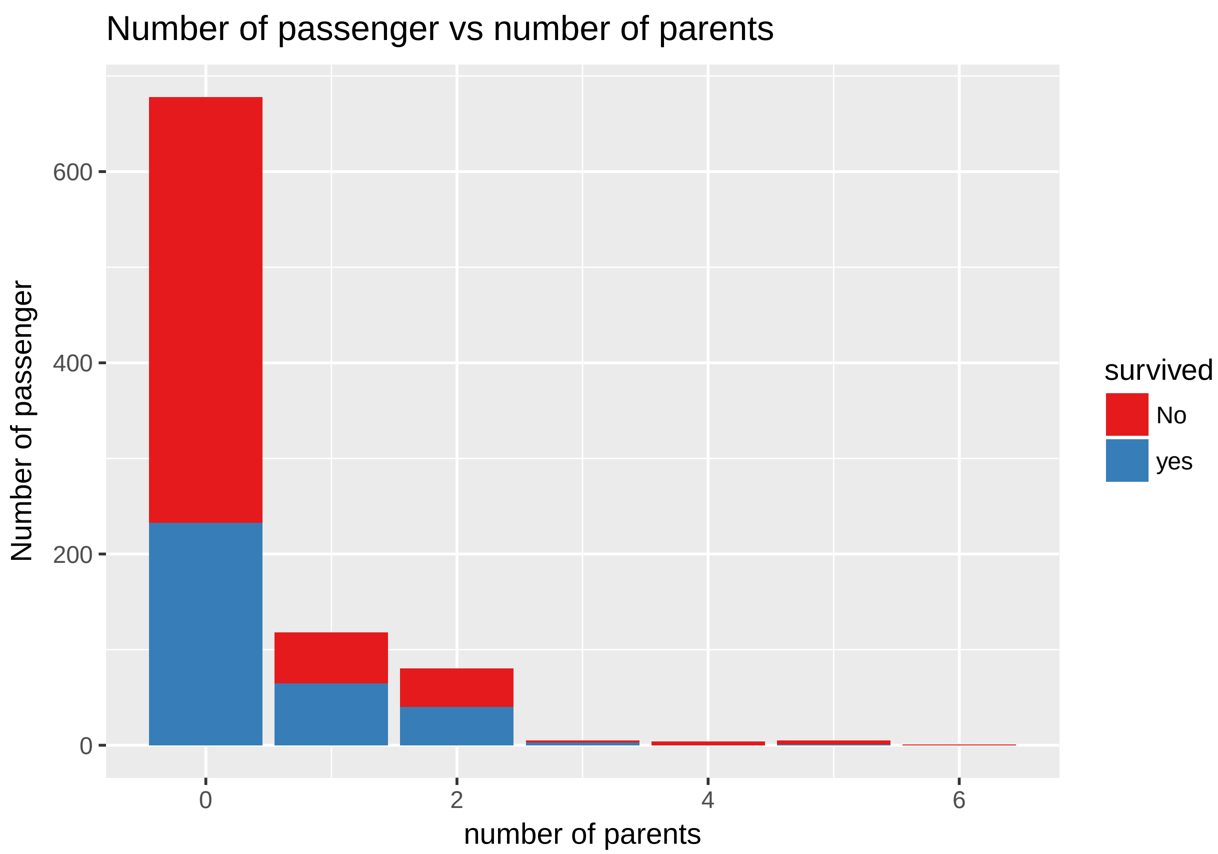
```
summary <- full %>% filter(set=='train') %>% group_by(parch) %>% summarise(passenger=n(),survived=sum(a
kable(summary)
```

parch	passenger	survived	survival_rate
0	678	233	34
1	118	65	55
2	80	40	50
3	5	3	60
4	4	0	0
5	5	1	20
6	1	0	0

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

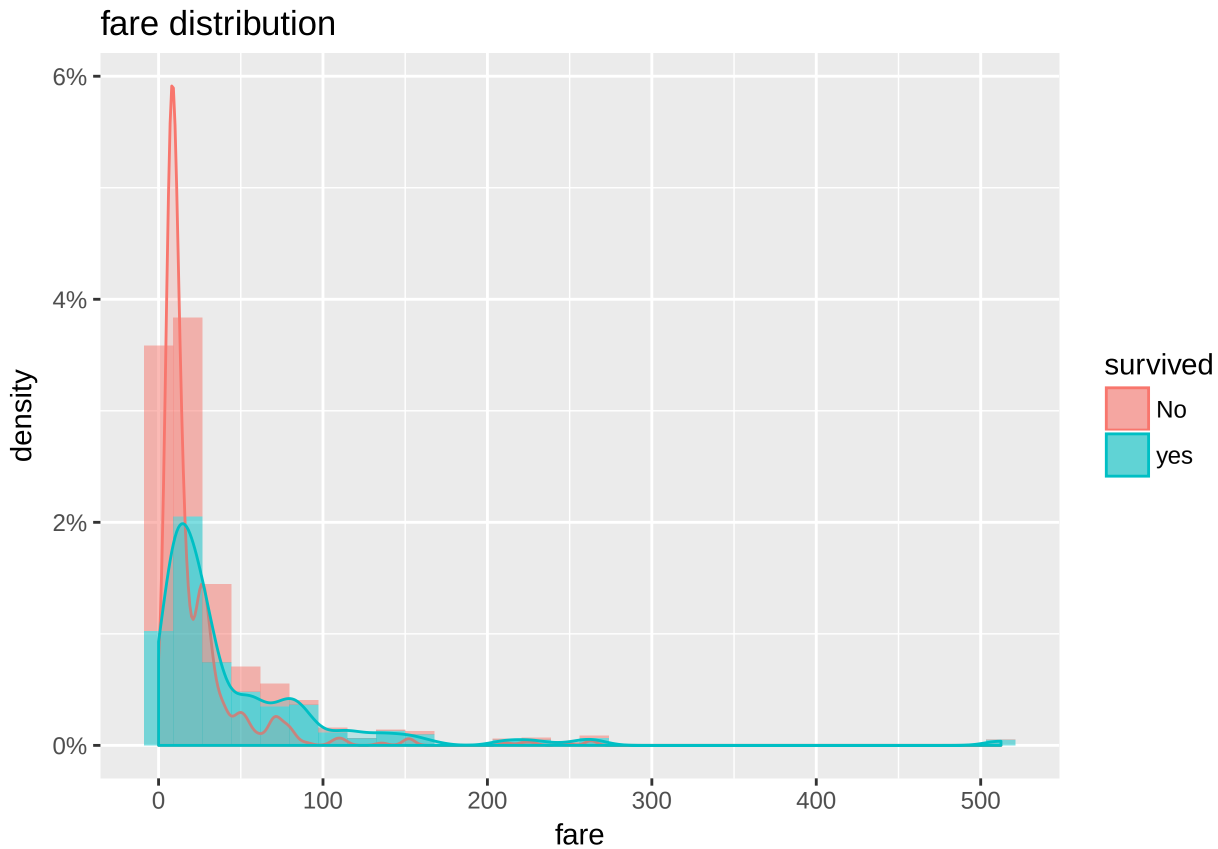



```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

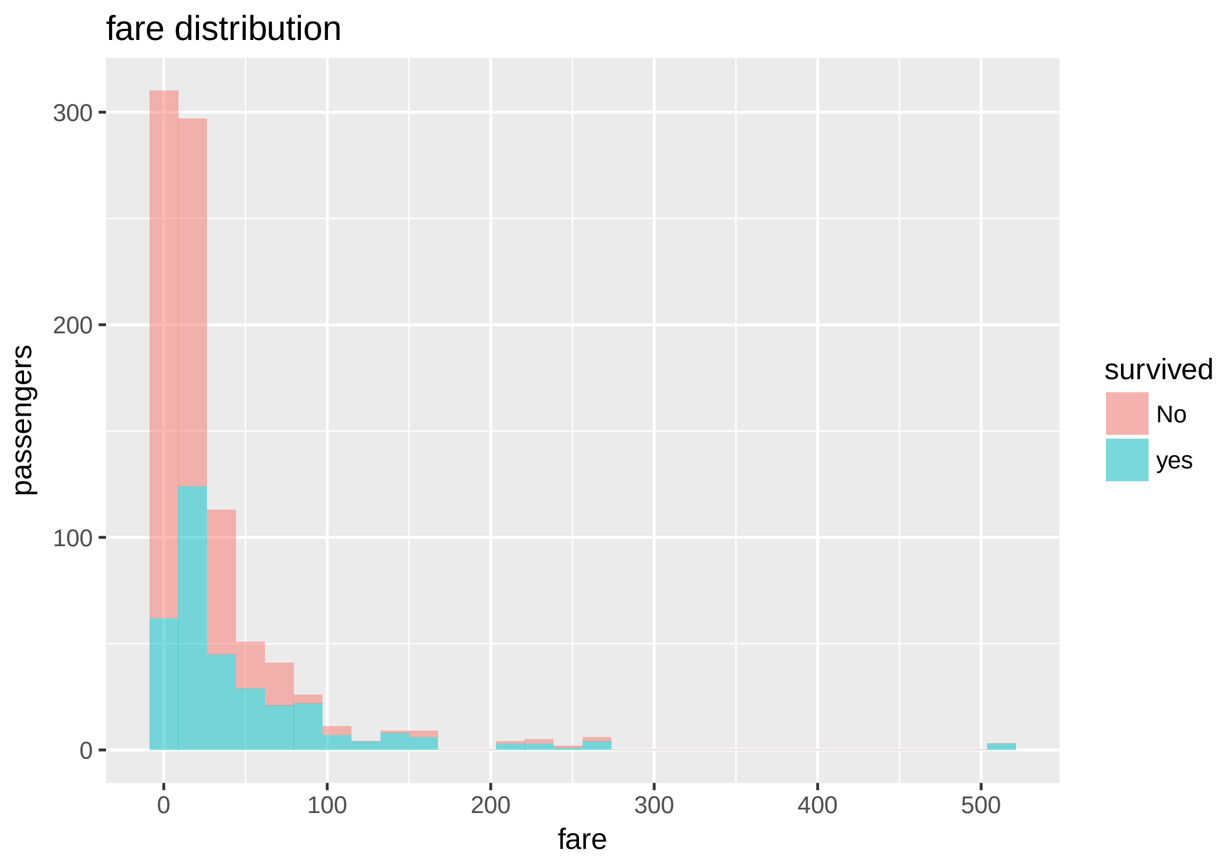


Fare

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

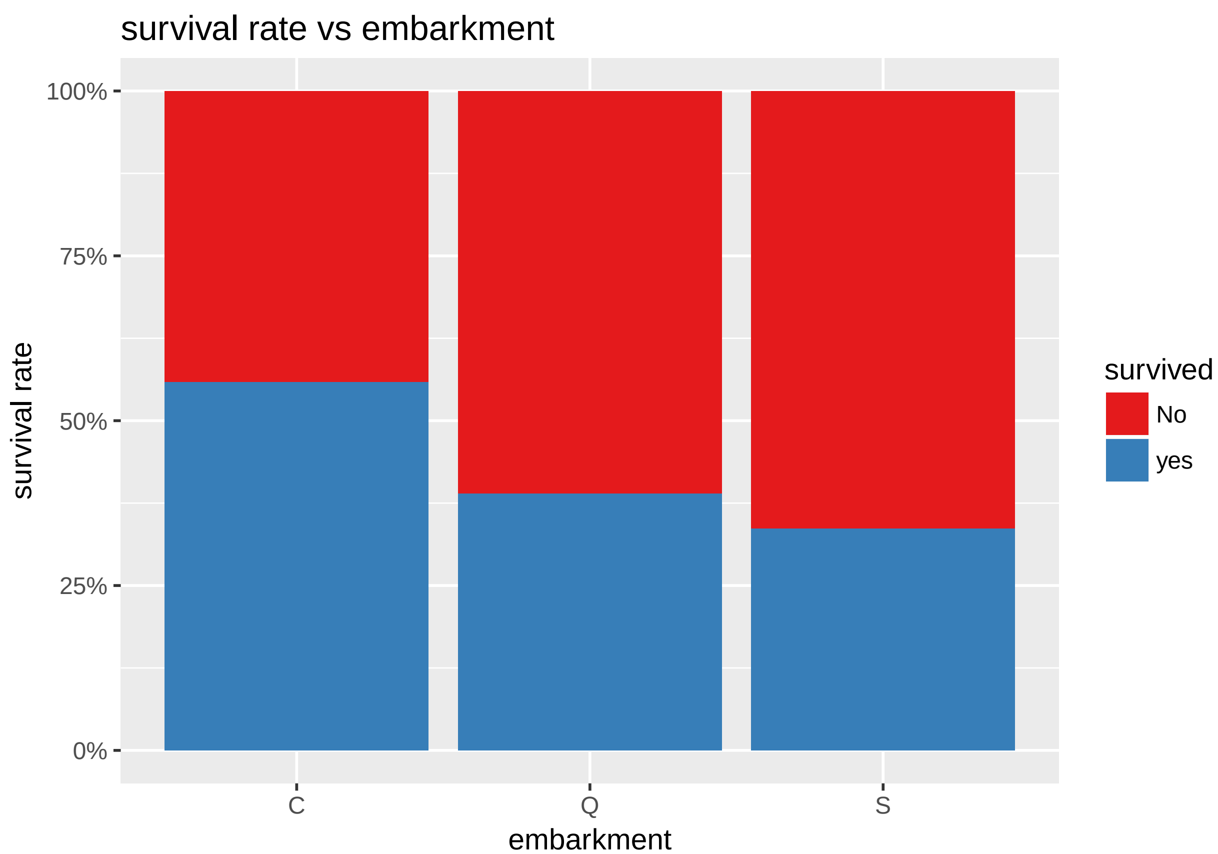


Embarked

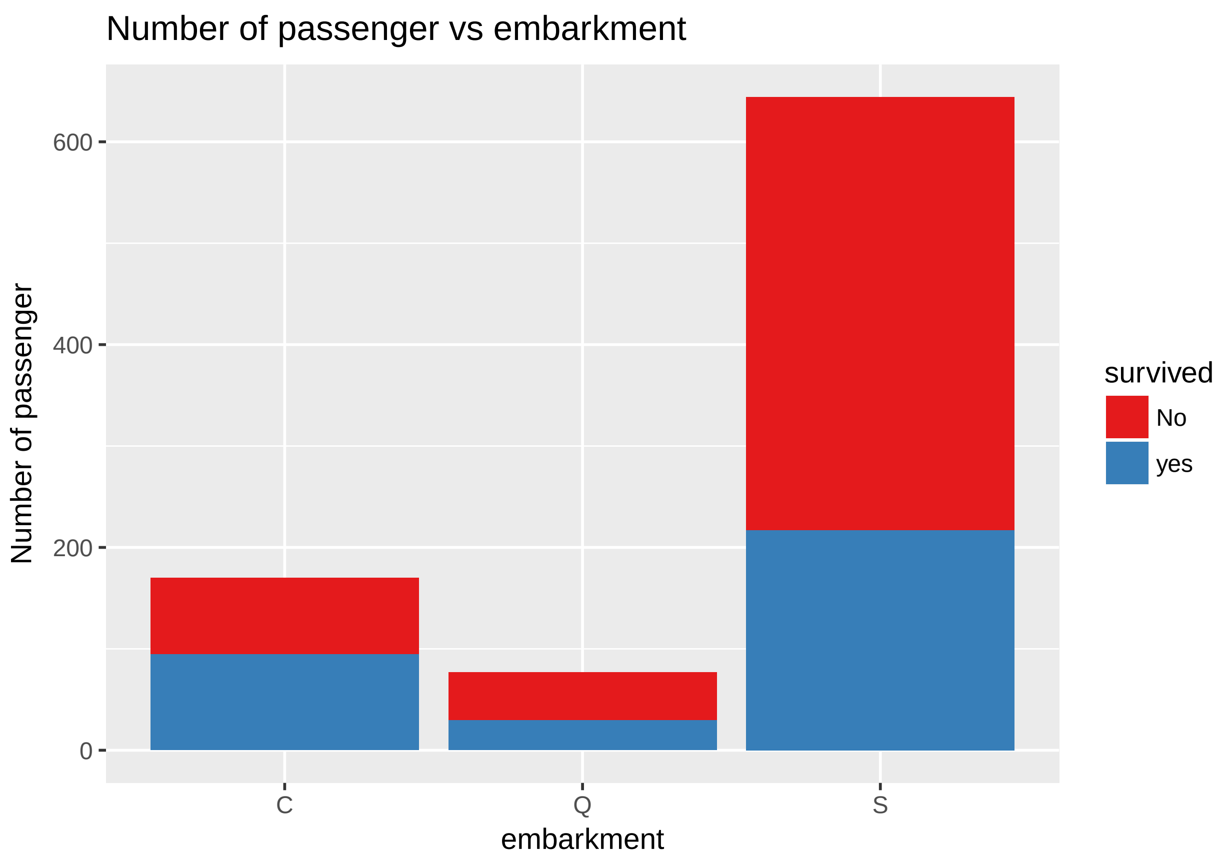
```
summary <- full %>% filter(set=='train') %>% group_by(embarked) %>% summarise(passenger=n(),survived=su
kable(summary)
```

embarked	passenger	survived	survival_rate
C	170	95	56
Q	77	30	39
S	644	217	34

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

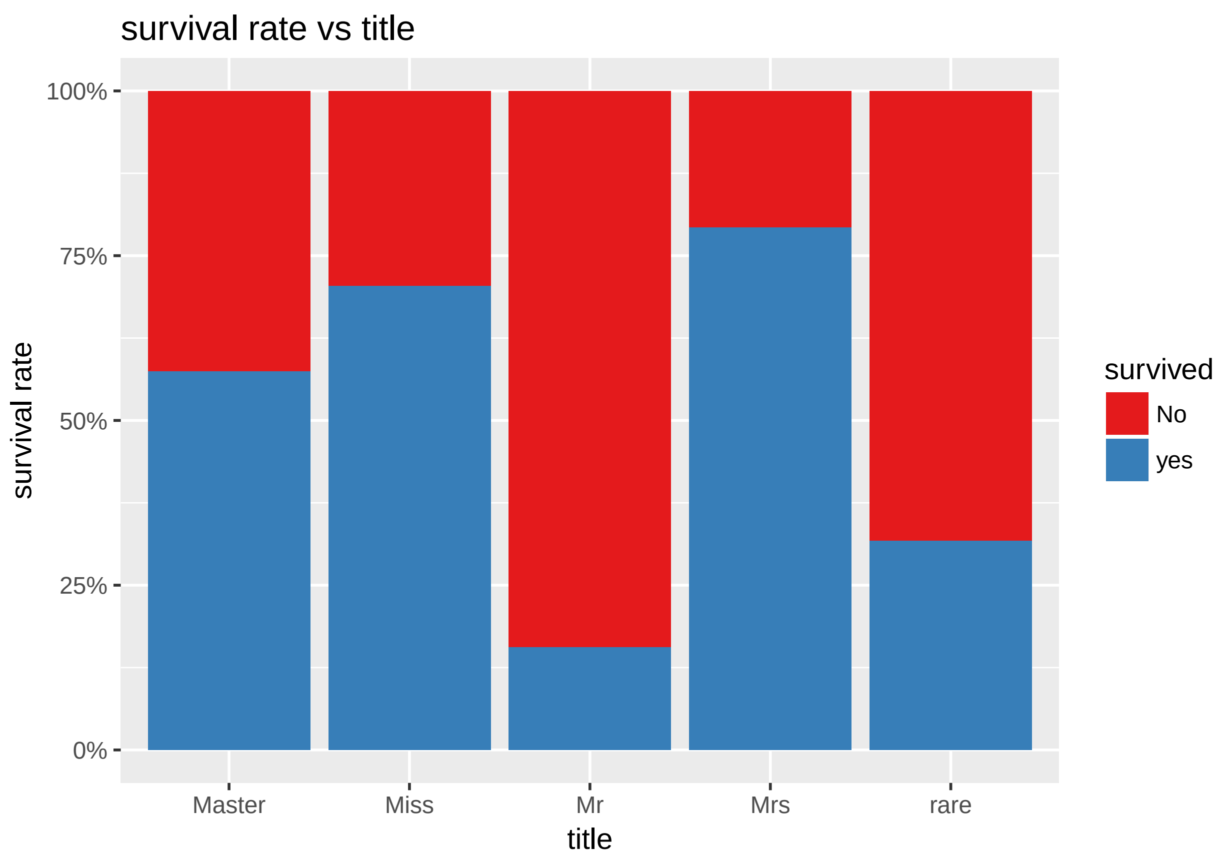


Title

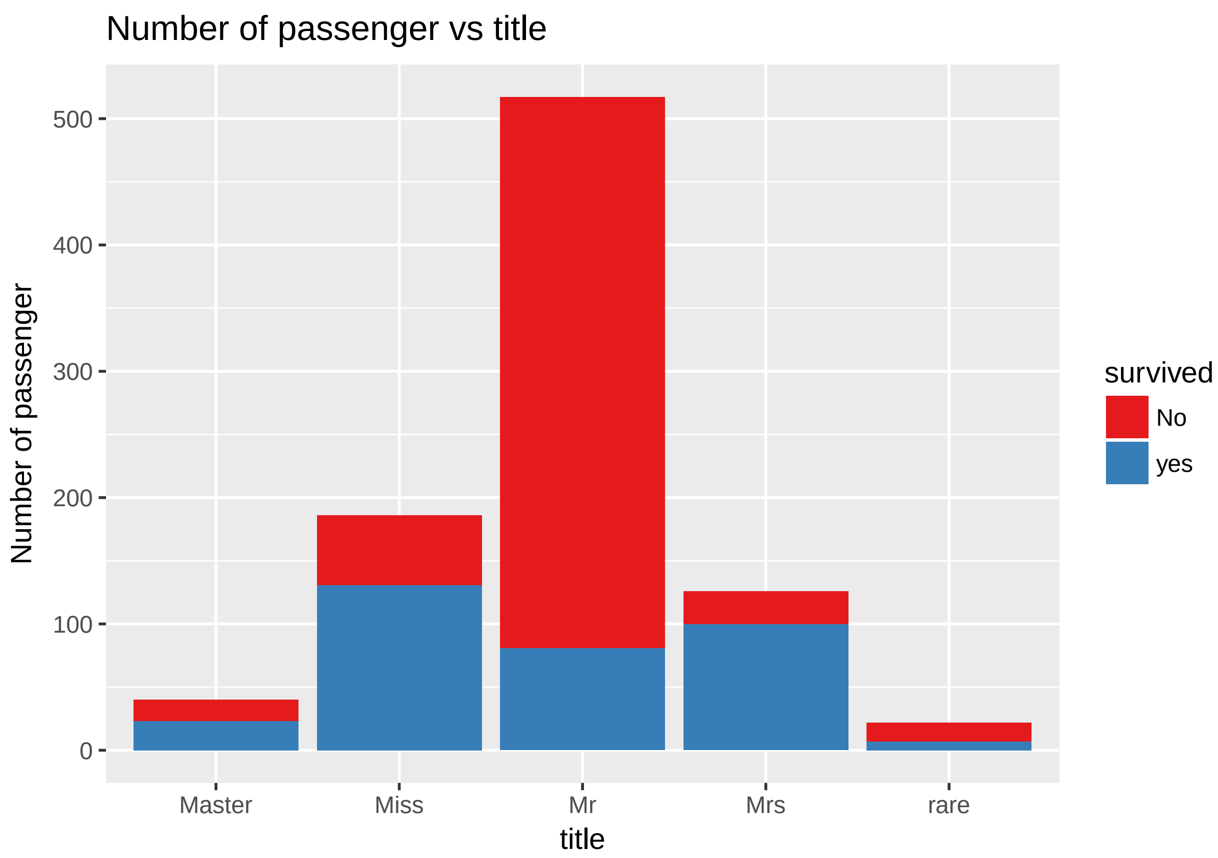
```
summary <- full %>% filter(set=='train') %>% group_by(title) %>% summarise(passenger=n(),survived=sum(a
kable(summary)
```

title	passenger	survived	survival_rate
Master	40	23	58
Miss	186	131	70
Mr	517	81	16
Mrs	126	100	79
rare	22	7	32

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tr
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

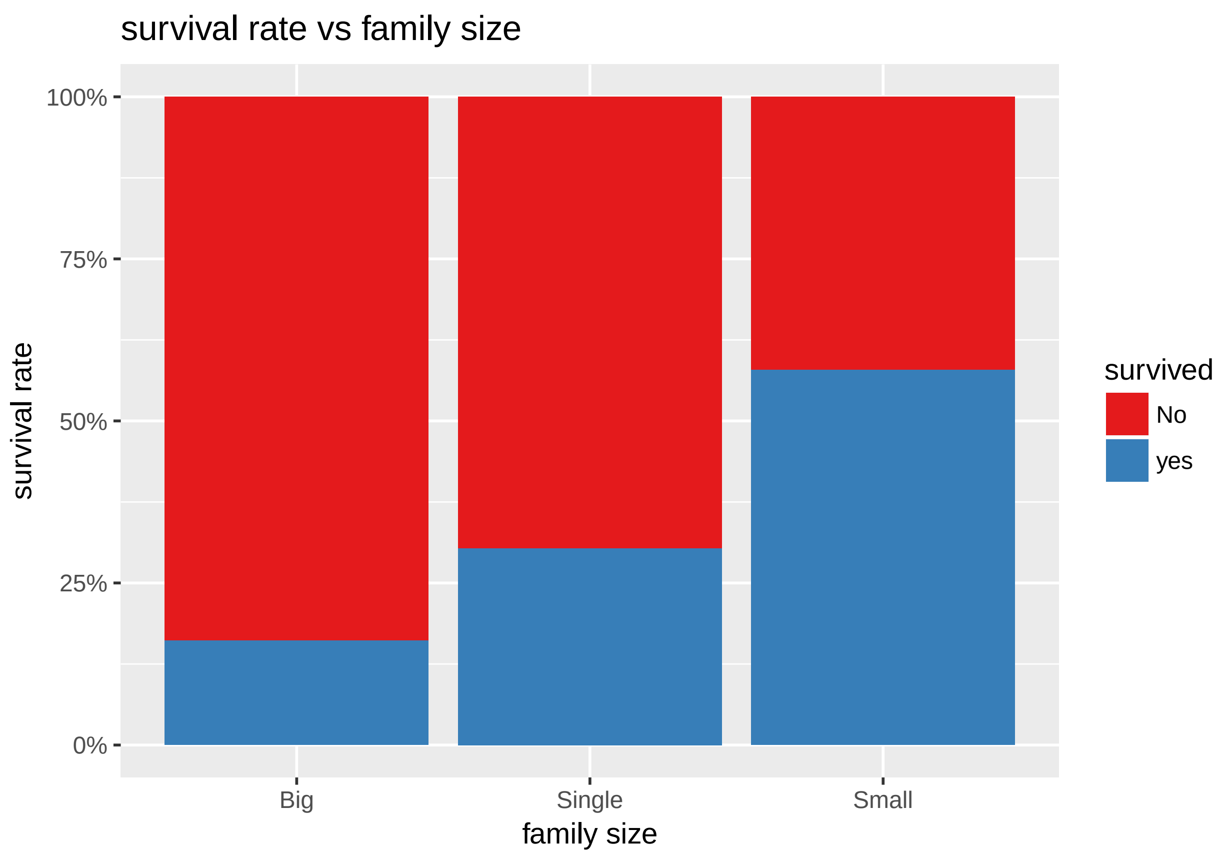


Family size

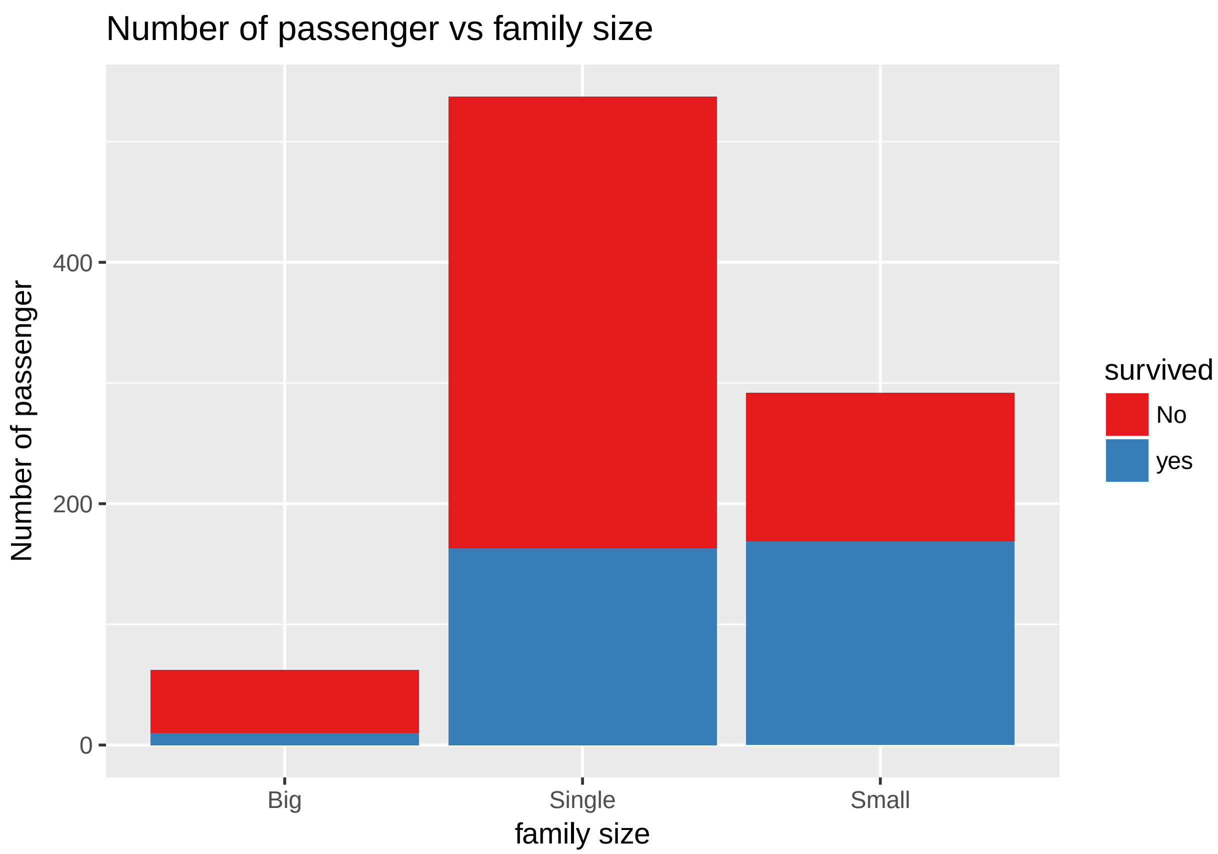
```
summary <- full %>% filter(set=='train') %>% group_by(FamilySized) %>% summarise(passenger=n(),survived=
kable(summary)
```

FamilySized	passenger	survived	survival_rate
Big	62	10	16
Single	537	163	30
Small	292	169	58

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```




```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

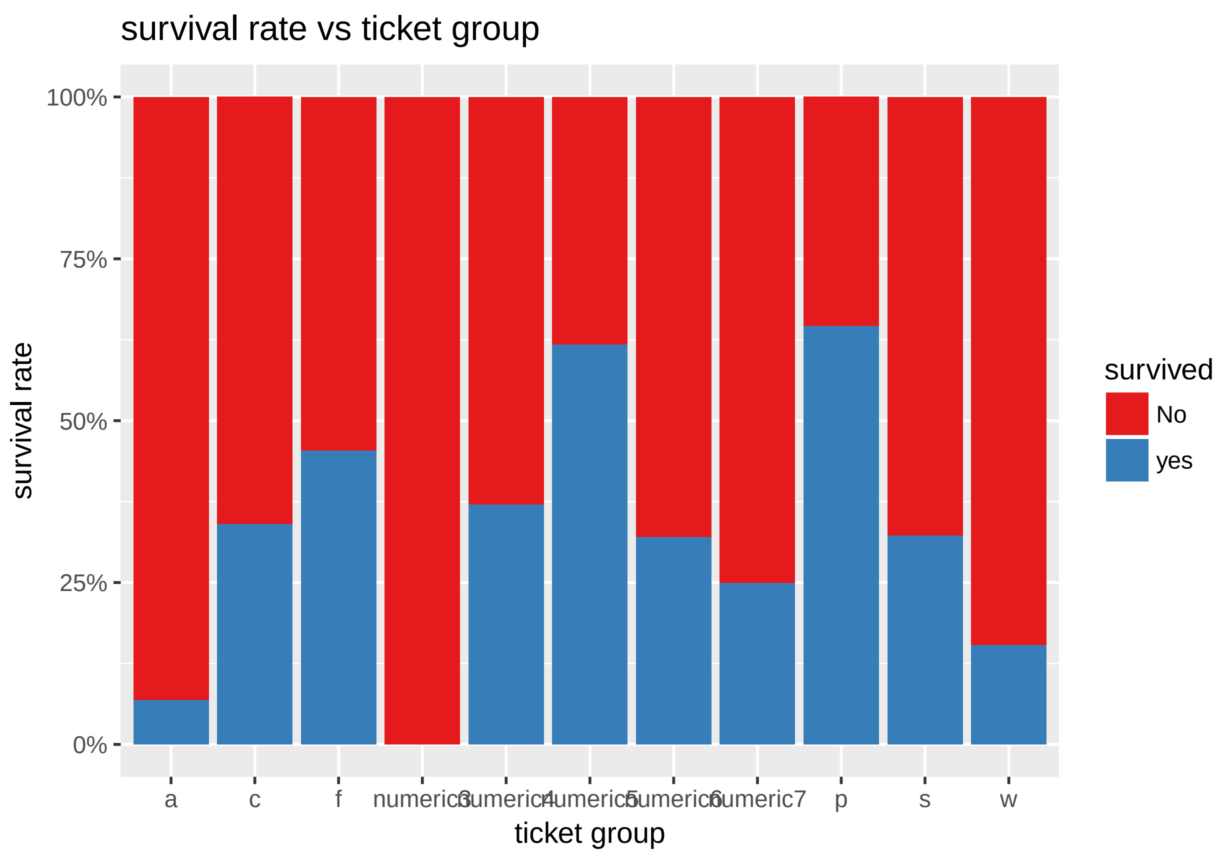


Ticket Group

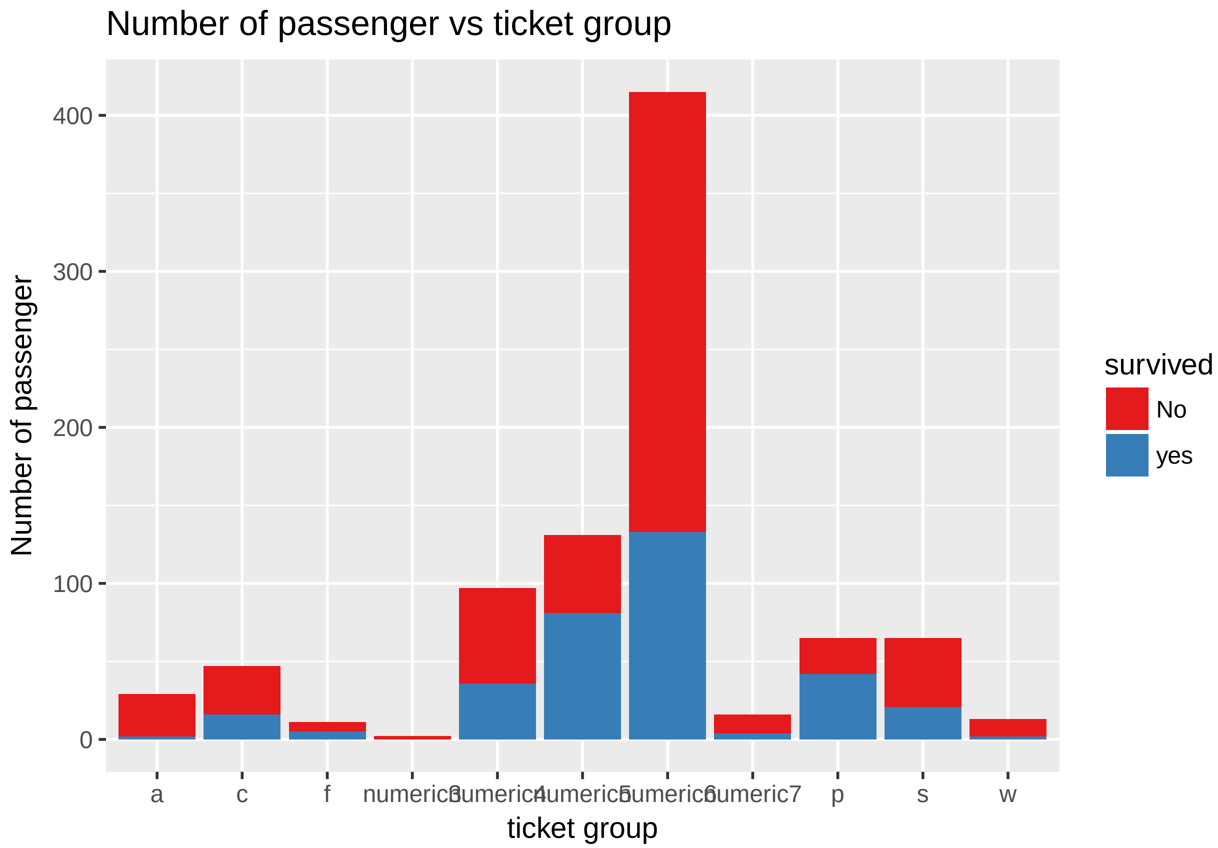
```
summary <- full %>% filter(set=='train') %>% group_by(ticketgroup) %>% summarise(passenger=n(),survived=
kable(summary)
```

ticketgroup	passenger	survived	survival_rate
a	29	2	7
c	47	16	34
f	11	5	45
numeric3	2	0	0
numeric4	97	36	37
numeric5	131	81	62
numeric6	415	133	32
numeric7	16	4	25
p	65	42	65
s	65	21	32
w	13	2	15

```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```

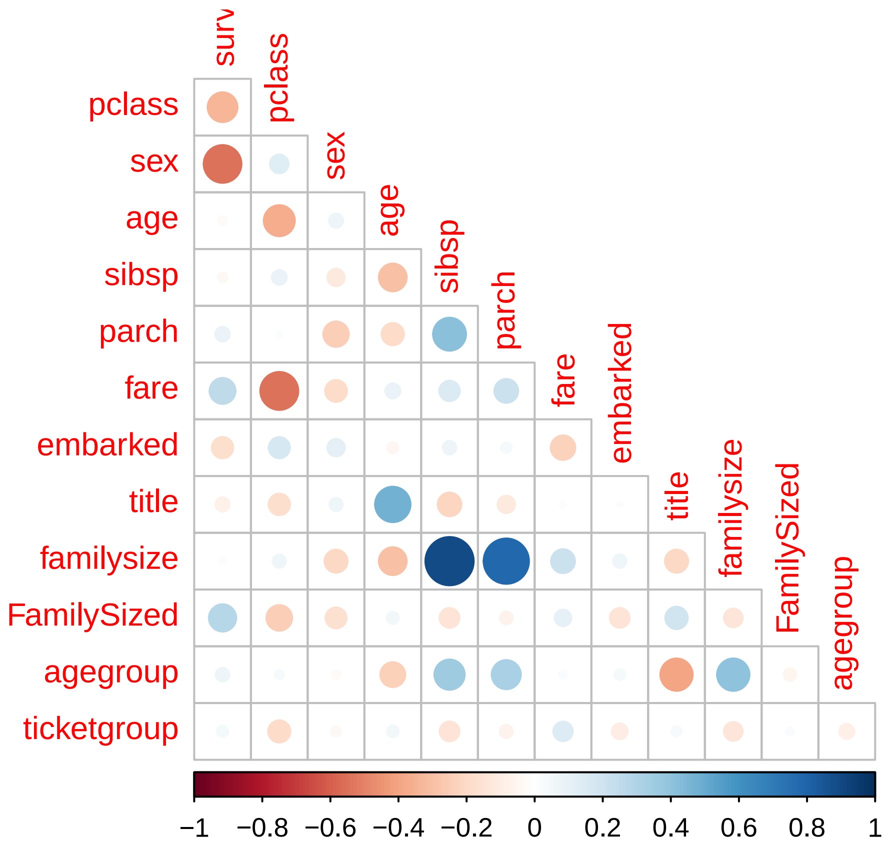



```
ggplot(full %>% mutate(survived=case_when(survived==0 ~ "No", survived==1 ~'yes'))  %>% filter(set=='tra
```



Feature Correlation

```
#feature correlation
coor_tbl <- full %>% filter(set=="train")%>%select(-passengerid,-name,-ticket,-cabin,-set) %>% mutate_al
```



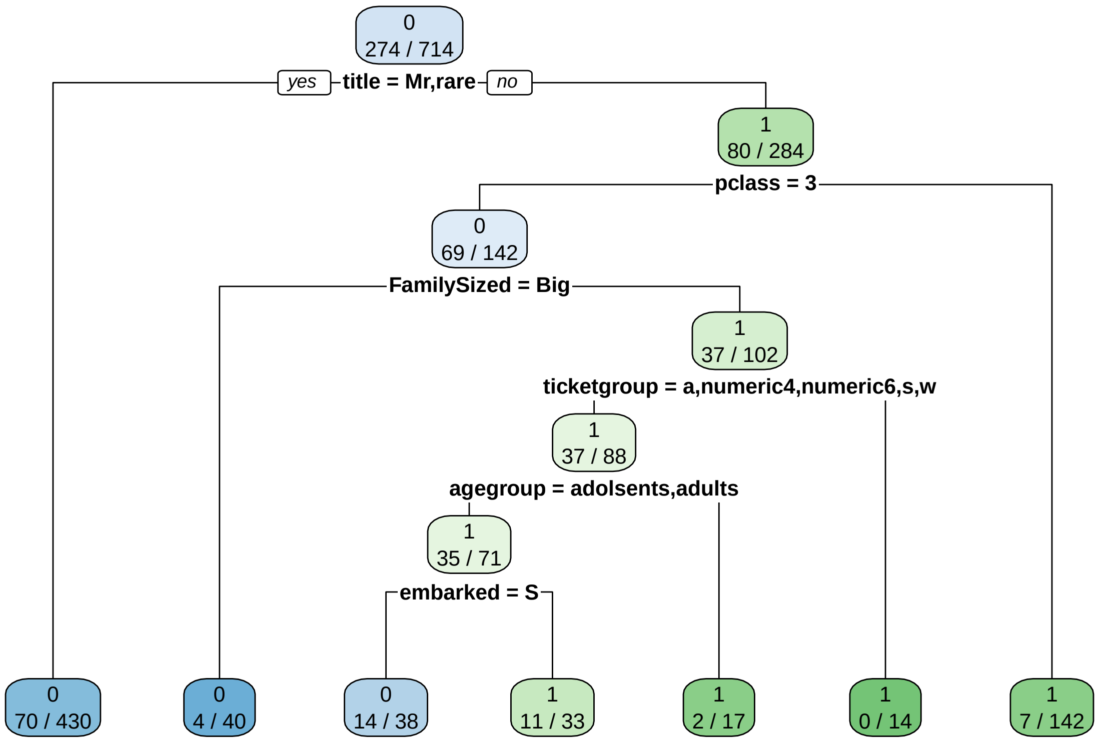
Data Preperation for prediction

```
#prep for prediction
set.seed(120)

train_dev <- full %>% filter(set=="train") %>% select(survived,pclass,sex,agegroup,ticketgroup,FamilySi
data_partition <- createDataPartition(train_dev$survived, p=0.8, list=F)

#Train
train_final <- train_dev[data_partition,]

#Development
dev_final <- train_dev[-data_partition,]

#test set for final prediction
test_final <- full %>% filter(set=="test") %>% select(survived,pclass,sex,agegroup,ticketgroup,FamilySi
```

Prediction Models

Classification Tree

```
model_dt <- rpart(survived~., data=train_final,method='class')
rpart.plot(model_dt,extra =  3)
```




```
predict_train <- predict(model_dt,data=train_final,type = "class")
confusionMatrix(predict_train,train_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 420  88
##          1  20 186
##
##                Accuracy : 0.8487
##                  95% CI : (0.8203, 0.8742)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6645
##  Mcnemar's Test P-Value : 1.14e-10
##
##             Sensitivity : 0.9545
##             Specificity : 0.6788
##          Pos Pred Value : 0.8268
##          Neg Pred Value : 0.9029
##              Prevalence : 0.6162
##          Detection Rate : 0.5882
##    Detection Prevalence : 0.7115
```

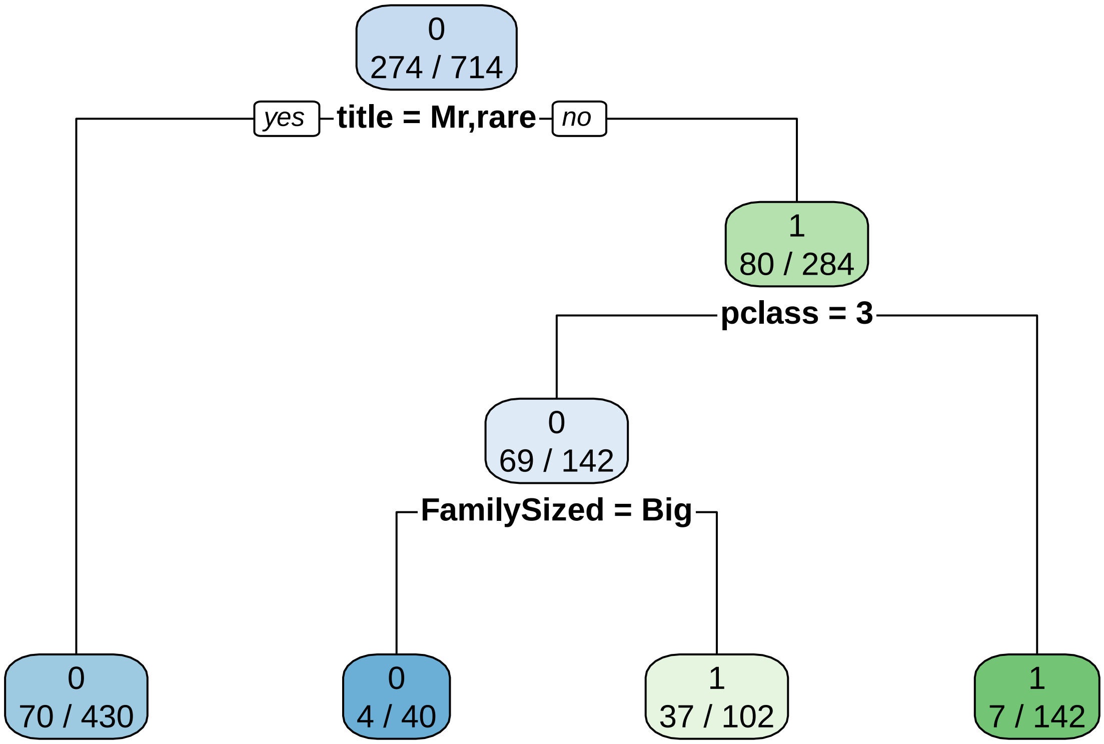
```
##       Balanced Accuracy : 0.8167
##
##        'Positive' Class : 0
##
```

```
predict_dev <- predict(model_dt, newdata = dev_final, type = "class")
confusionMatrix(predict_dev,dev_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 102  22
##          1   7  46
##
##                Accuracy : 0.8362
##                  95% CI : (0.7732, 0.8874)
##     No Information Rate : 0.6158
##     P-Value [Acc > NIR] : 1.38e-10
##
##                   Kappa : 0.6388
##  Mcnemar's Test P-Value : 0.00933
##
##             Sensitivity : 0.9358
##             Specificity : 0.6765
##          Pos Pred Value : 0.8226
##          Neg Pred Value : 0.8679
##              Prevalence : 0.6158
##          Detection Rate : 0.5763
##    Detection Prevalence : 0.7006
##       Balanced Accuracy : 0.8061
##
##        'Positive' Class : 0
##
```

Cross validated decision tree

```
set.seed(120)
cv.10 <- createMultiFolds(train_final$survived,k=10,times=10)
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 10, index=cv.10)
train_final <- as.data.frame(train_final)
set.seed(120)
model_cdt <- train(x=train_final[,-1],y=train_final[,1], method="rpart", trControl= ctrl)
rpart.plot(model_cdt$finalModel,extra =  3)
```




```
predict2_train <- predict(model_cdt$finalModel, data=train_final, type="class")
confusionMatrix(predict2_train,train_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 396  74
##          1  44 200
##
##                Accuracy : 0.8347
##                  95% CI : (0.8054, 0.8612)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6432
##  Mcnemar's Test P-Value : 0.007593
##
##             Sensitivity : 0.9000
##             Specificity : 0.7299
##          Pos Pred Value : 0.8426
##          Neg Pred Value : 0.8197
##              Prevalence : 0.6162
##          Detection Rate : 0.5546
##    Detection Prevalence : 0.6583
##       Balanced Accuracy : 0.8150
```

```
##
##        'Positive' Class : 0
##
```

```r
predict2_dev <- predict(model_cdt$finalModel, newdata=dev_final, type="class")
confusionMatrix(predict2_dev,dev_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 96 18
##          1 13 50
##
##                Accuracy : 0.8249
##                  95% CI : (0.7607, 0.8778)
##     No Information Rate : 0.6158
##     P-Value [Acc > NIR] : 1.304e-09
##
##                   Kappa : 0.6247
##  Mcnemar's Test P-Value : 0.4725
##
##             Sensitivity : 0.8807
##             Specificity : 0.7353
##          Pos Pred Value : 0.8421
##          Neg Pred Value : 0.7937
##              Prevalence : 0.6158
##          Detection Rate : 0.5424
##    Detection Prevalence : 0.6441
##       Balanced Accuracy : 0.8080
##
##        'Positive' Class : 0
##
```

Logistic Regression

```r
model_logit <- glm(survived~.,data = train_final, family = binomial)
predict_logit_train <- predict(model_logit, data=train_final, type='response')
table(train_final$survived, predict_logit_train>0.5)
```

```
##
##     FALSE TRUE
##   0   394   46
##   1    65  209
```

```r
accurcy <- (389+206)/(389+206+51+68)
accurcy
```

```
## [1] 0.8333333
```

```r
predict_logit_dev <- predict(model_logit, newdata=dev_final, type='response')
table(dev_final$survived, predict_logit_dev>0.5)
```

```
##
##     FALSE TRUE
```

```
##   0    95   14
##   1    17   51
```

```r
accurcy <- (95+51)/(95+51+17+14)
accurcy
```

```
## [1] 0.8248588
```

Random Forest

```r
model_rf <- randomForest(x=train_final[,-1],y=train_final[,1], mtry = 3, ntree = 1000, importance=T)
model_rf
```

```
##
## Call:
##  randomForest(x = train_final[, -1], y = train_final[, 1], ntree = 1000,      mtry = 3, importance =
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 17.51%
## Confusion matrix:
##     0   1 class.error
## 0 391  49   0.1113636
## 1  76 198   0.2773723
```

```r
predict_train_rf <- predict(model_rf,data=train_final,type = "class")
confusionMatrix(predict_train_rf,train_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 391  76
##          1  49 198
##
##                Accuracy : 0.8249
##                  95% CI : (0.795, 0.8521)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6228
##  Mcnemar's Test P-Value : 0.02004
##
##             Sensitivity : 0.8886
##             Specificity : 0.7226
##          Pos Pred Value : 0.8373
##          Neg Pred Value : 0.8016
##              Prevalence : 0.6162
##          Detection Rate : 0.5476
##    Detection Prevalence : 0.6541
##       Balanced Accuracy : 0.8056
##
##        'Positive' Class : 0
```

```
##
```

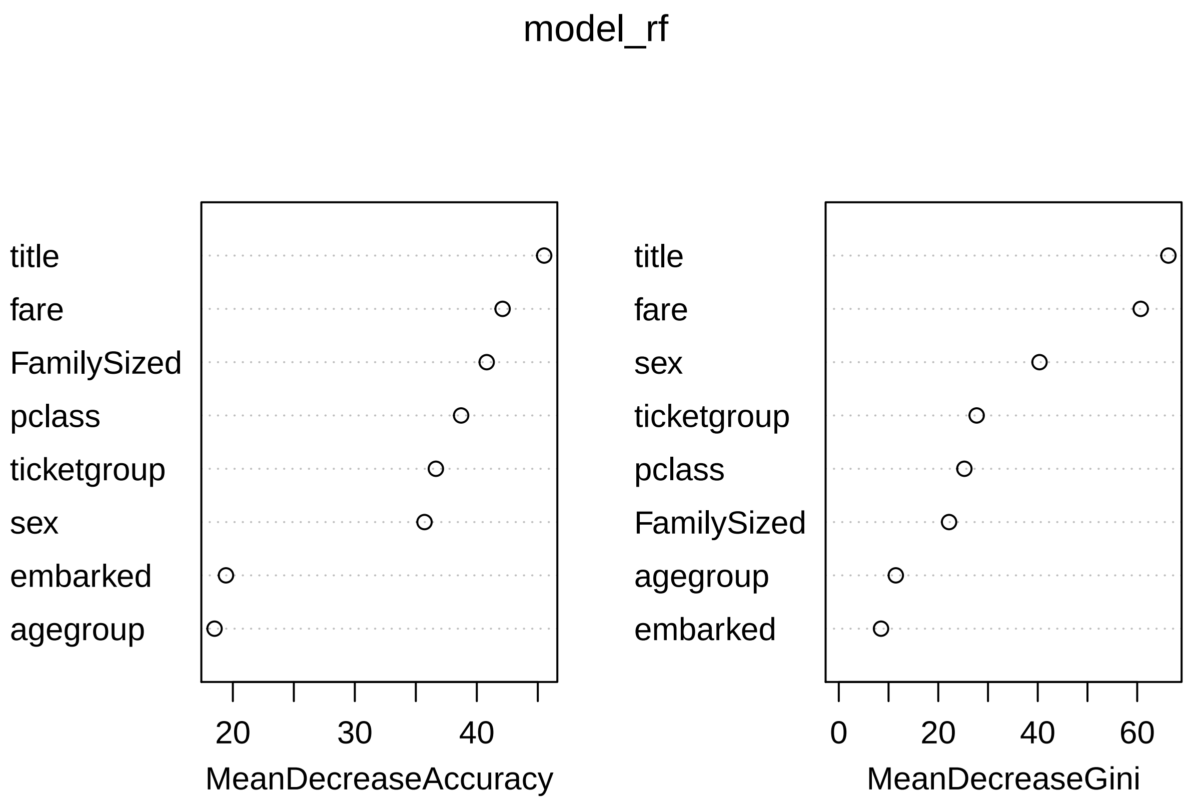
```r
predict_dev_rf <- predict(model_rf, newdata = dev_final, type = "class")
confusionMatrix(predict_dev_rf,dev_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 97 16
##          1 12 52
##
##                Accuracy : 0.8418
##                  95% CI : (0.7795, 0.8922)
##     No Information Rate : 0.6158
##     P-Value [Acc > NIR] : 4.229e-11
##
##                   Kappa : 0.6619
##  Mcnemar's Test P-Value : 0.5708
##
##             Sensitivity : 0.8899
##             Specificity : 0.7647
##          Pos Pred Value : 0.8584
##          Neg Pred Value : 0.8125
##              Prevalence : 0.6158
##          Detection Rate : 0.5480
##    Detection Prevalence : 0.6384
##       Balanced Accuracy : 0.8273
##
##        'Positive' Class : 0
##
```

```r
importance(model_rf)
```

```
##                    0         1 MeanDecreaseAccuracy MeanDecreaseGini
## pclass      26.14943 28.243133             38.71321        25.245956
## sex         34.25957 22.039769             35.71113        40.356064
## agegroup    11.80554 16.679932             18.49860        11.459044
## ticketgroup 34.37507  8.020393             36.64303        27.718834
## FamilySized 33.29631 14.895612             40.80811        22.172790
## title       39.50196 34.794106             45.51909        66.273201
## fare        26.57558 28.625637             42.11207        60.705559
## embarked    15.29511  8.283617             19.43542         8.492381
```

```r
varImpPlot(model_rf)
```



Cross Validated Random Forest

```
set.seed(120)
cv.10 <- createMultiFolds(train_final$survived,k=10,times=10)
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 10, index=cv.10)
train_final <- as.data.frame(train_final)
set.seed(120)
model_crf <- train(x=train_final[,-1],y=train_final[,1], method="rf", trControl= ctrl, ntree=1000, impo
model_crf
```

```
## Random Forest
##
## 714 samples
##   8 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 643, 642, 643, 642, 643, 643, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.8268995  0.6242872
##   5     0.8205966  0.6151005
##   8     0.8148787  0.6041650
```

```
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```r
predict_train_crf <- predict(model_crf,data=train_final)
confusionMatrix(predict_train_crf,train_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 418  55
##          1  22 219
##
##                Accuracy : 0.8922
##                  95% CI : (0.8671, 0.914)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7667
##  Mcnemar's Test P-Value : 0.0002656
##
##             Sensitivity : 0.9500
##             Specificity : 0.7993
##          Pos Pred Value : 0.8837
##          Neg Pred Value : 0.9087
##              Prevalence : 0.6162
##          Detection Rate : 0.5854
##    Detection Prevalence : 0.6625
##       Balanced Accuracy : 0.8746
##
##        'Positive' Class : 0
##
```

```r
predict_dev_crf <- predict(model_crf, newdata = dev_final)
confusionMatrix(predict_dev_crf,dev_final$survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 98 18
##          1 11 50
##
##                Accuracy : 0.8362
##                  95% CI : (0.7732, 0.8874)
##     No Information Rate : 0.6158
##     P-Value [Acc > NIR] : 1.38e-10
##
##                   Kappa : 0.6469
##  Mcnemar's Test P-Value : 0.2652
##
##             Sensitivity : 0.8991
##             Specificity : 0.7353
##          Pos Pred Value : 0.8448
```

```
##          Neg Pred Value : 0.8197
##              Prevalence : 0.6158
##          Detection Rate : 0.5537
##    Detection Prevalence : 0.6554
##       Balanced Accuracy : 0.8172
##
##        'Positive' Class : 0
##
```

```r
var_imp <- varImp(model_crf, scale=F)
var_imp
```

```
## rf variable importance
##
##             Importance
## title            35.67
## sex              30.97
## pclass           27.54
## fare             26.18
## FamilySized      23.88
## ticketgroup      18.77
## agegroup         15.60
## embarked         11.48
```

```r
plot(var_imp)
```