**泰坦尼克数据训练 2**

**--完整版数据整理、分析、可视化**

# 数据背景与来源

泰坦尼克号（Titanic）是英国的一艘奥林匹克级邮轮，它是当时世界上体积最庞大、内部设施最豪华的客运轮船，号称“世界工业史上的奇迹”。1912年4月14日，在它的处女航中——英国的南安普顿出发驶往美国纽约，泰坦尼克号与一座冰山相撞，造成船体破裂进水，几小时后沉入大西洋底，船上2224名船员及乘客中，一共有超过1500人丧生。而在这2224人中，丧生并非完全随机发生的，船上有来自不同年龄、性别和社会地位的人。在我们的一般认知中，儿童和老人、妇女、社会地位高的人存活的概率越高，这些人的生存与否和他们各自的特征是息息相关的，但是我们也不能主观臆断说上面提到的三类人的存活率就一定高，因此把这些特征作为自变量，生存的结果作为因变量，对数据进行分析建模以及预测，去探索隐藏在这些数据背后的规律。本次分析所用的数据是来自Kaggle竞赛平台的入门项目Titanic:Machine Learning from Disaster，数据一共包含三个文件（train .csv、test.csv和gender\_submission.csv），分别为训练数据、测试数据和提交文件。

# 数据导入

**library**(tidyverse)**library**(ggplot2)**library**(caret)**library**(caretEnsemble)

train = read\_csv("C:/Users/babe/Desktop/Titanicdata/train.csv")

test = read\_csv("C:/Users/babe/Desktop/Titanicdata/test.csv")

sub= read\_csv("C:/Users/babe/Desktop/Titanicdata/gender\_submission.csv")

# 数据分析

## 数据结构

dim(train)

## [1] 891 12

dim(test)

## [1] 418 11

names(train)

## [1] "PassengerId" "Survived" "Pclass" "Name" "Sex"

## [6] "Age" "SibSp" "Parch" "Ticket" "Fare"

## [11] "Cabin" "Embarked"

names(test)

## [1] "PassengerId" "Pclass" "Name" "Sex" "Age"

## [6] "SibSp" "Parch" "Ticket" "Fare" "Cabin"

## [11] "Embarked"

训练数据集一共有891条记录，12个字段，测试数据集一共有418条记录，11个字段，测试数据相比训练数据缺少了Survived这个字段，也就是需要预测的值，12个字段的含义分别为： “PassengerId”：乘客编号ID（唯一标识） “Survived”：存活情况 “Pclass”：舱位等级 “Name”：乘客姓名 “Sex”：性别 “Age”：年龄 “SibSp”：同在船上的兄弟姐妹/配偶数量 “Parch”：同在船上的父母/小孩数量 “Ticket”：船票信息 “Fare”：船票价格 “Cabin”：客舱号码 “Embarked”：登船港口 为了方便查看整体情况，把它们进行行合并,由于测试没有Survived这个字段，因此只对11个字段进行行合并。

train\_test = rbind(train[c(1,3:12)],test)

dim(train\_test)

## [1] 1309 11

合并后的数据一共有1309条记录，11个字段。缺失值对数据分析有很大的影响，接下来查看一下哪些字段存在缺失值以及缺失程度。

na=data.frame(is.na(train\_test))%>%map\_int(sum)

na

## PassengerId Pclass Name Sex Age SibSp

## 0 0 0 0 263 0

## Parch Ticket Fare Cabin Embarked

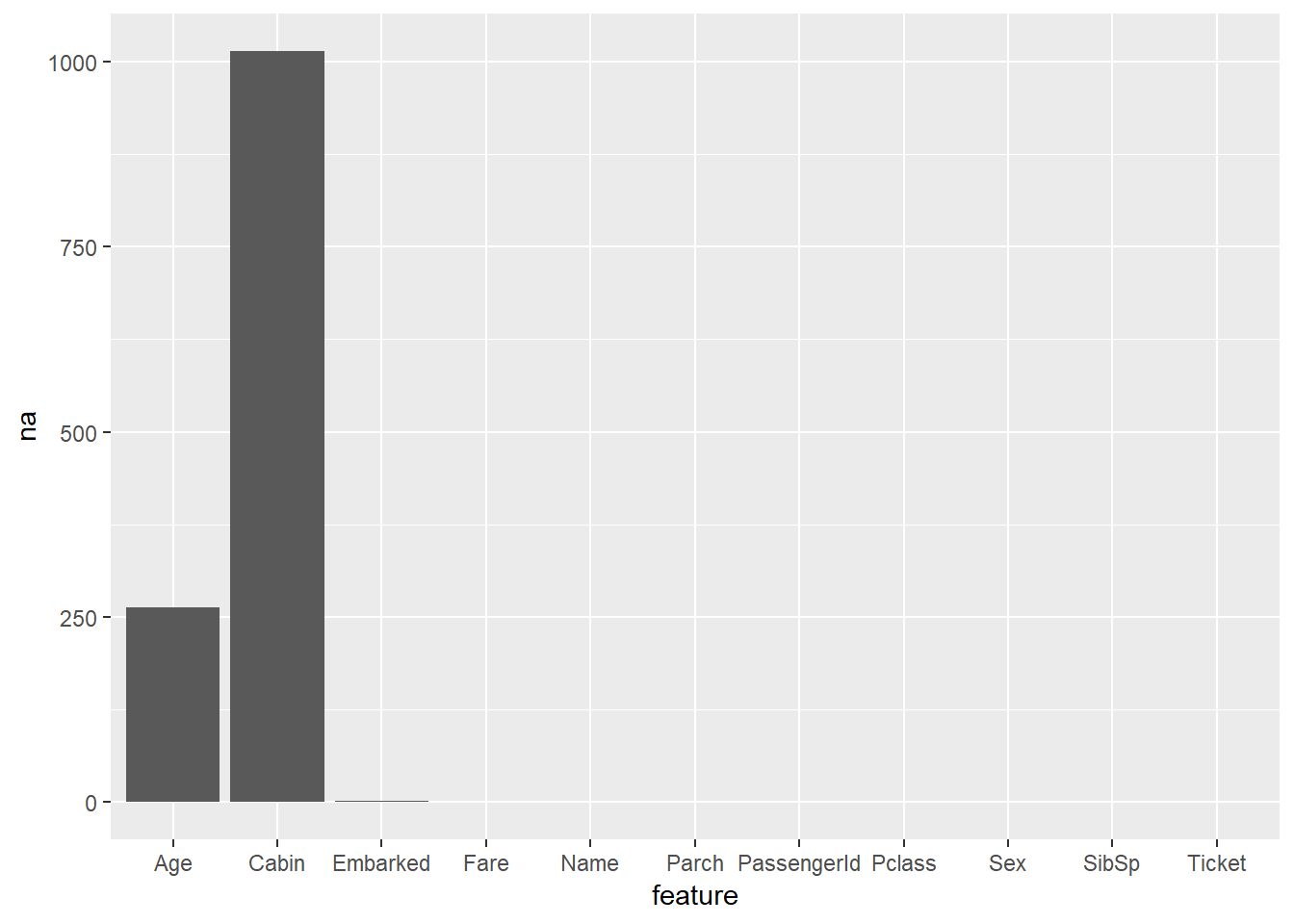
## 0 0 1 1014 2

na\_data=data.frame('feature'=row.names(data.frame(na)),'na\_count'=na)

na\_data%>%

ggplot()+

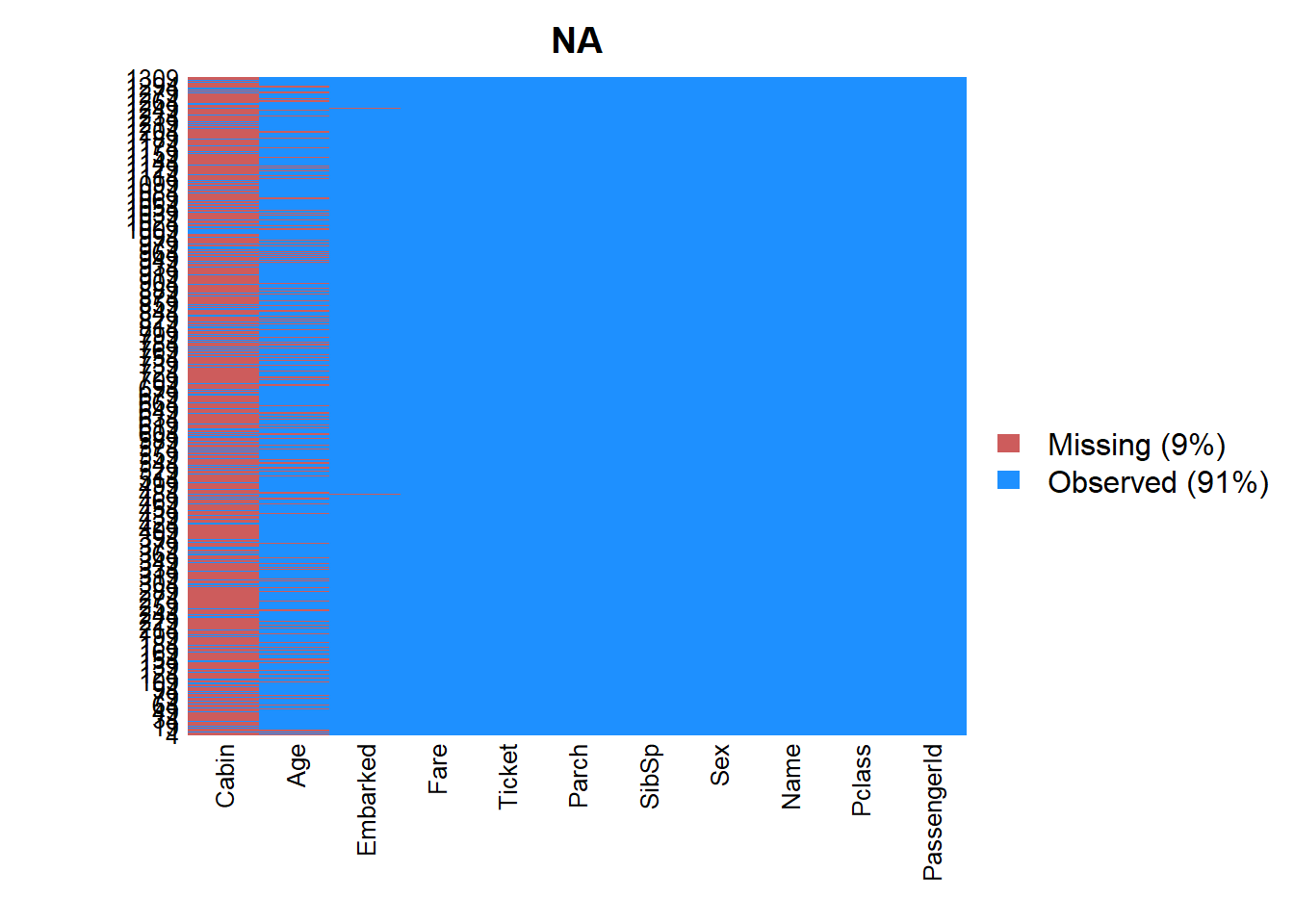
geom\_bar(aes(x=feature,y=na),stat = 'identity')



也可以直接用Amelia包对缺失值可视化

**library**(Amelia)

missmap(train\_test,main = "NA")



可以看到Age,Cabin,Embarked,Fare四个字段存在缺失值，其中Cabin字段的缺失值超过了1000个，缺失程度较为严重。

## 特征分析和缺失值处理

利用summary函数对数据进行简单的描述性统计，包括最小值、最大值、中位数、四分位数、均值，以及非数值型特征的属性和频数统计。

summary.data.frame(train\_test)

## PassengerId Pclass Name Sex

## Min. : 1 Min. :1.000 Length:1309 Length:1309

## 1st Qu.: 328 1st Qu.:2.000 Class :character Class :character

## Median : 655 Median :3.000 Mode :character Mode :character

## Mean : 655 Mean :2.295

## 3rd Qu.: 982 3rd Qu.:3.000

## Max. :1309 Max. :3.000

##

## 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

## Min. : 0.000 Length:1309 Length:1309

## 1st Qu.: 7.896 Class :character Class :character

## Median : 14.454 Mode :character Mode :character

## Mean : 33.295

## 3rd Qu.: 31.275

## Max. :512.329

## NA's :1

观察结果可以得知：Age特征中最小的值只有0.17岁，最大值有80岁，代表的分别是婴儿和老人，它的中位数和均值很接近，表明年龄分布较为均匀，大部分分布在21岁到39岁之间。SibSp特征中最小值、下四分位数、中位数均为0，而最大值为8，与它的情况类似的有Parch特征，考虑到二者的实际含义，不难理解大部分乘客都是独自乘坐，另外一部分则是拖家带口乘坐。Fare特征中，最小值为0，表明该名乘客坐的可能是甲板，而最大值为512.239，表明该名乘客坐的可能是最好的头等舱。而船票价格均值比中位数大了一倍多，表明船票价格的分布为右偏分布，有钱人和穷人之间的差距较大。 上面对数据粗略的分析并不能得出多少结论，接下来对数据的各个特征之间的关系以及特征与生存率之间的关系进行深入分析。特征中的Pclass、Sex、Embarked为定类变量且类别较少，故先对它们进行分析。由于数据中的变量类型为character型，因此先把它们转化成factor型。

train=train%>%

mutate(Survived=as.factor(Survived),Pclass=as.factor(Pclass),

Sex=as.factor(Sex),Embarked=as.factor(Embarked))

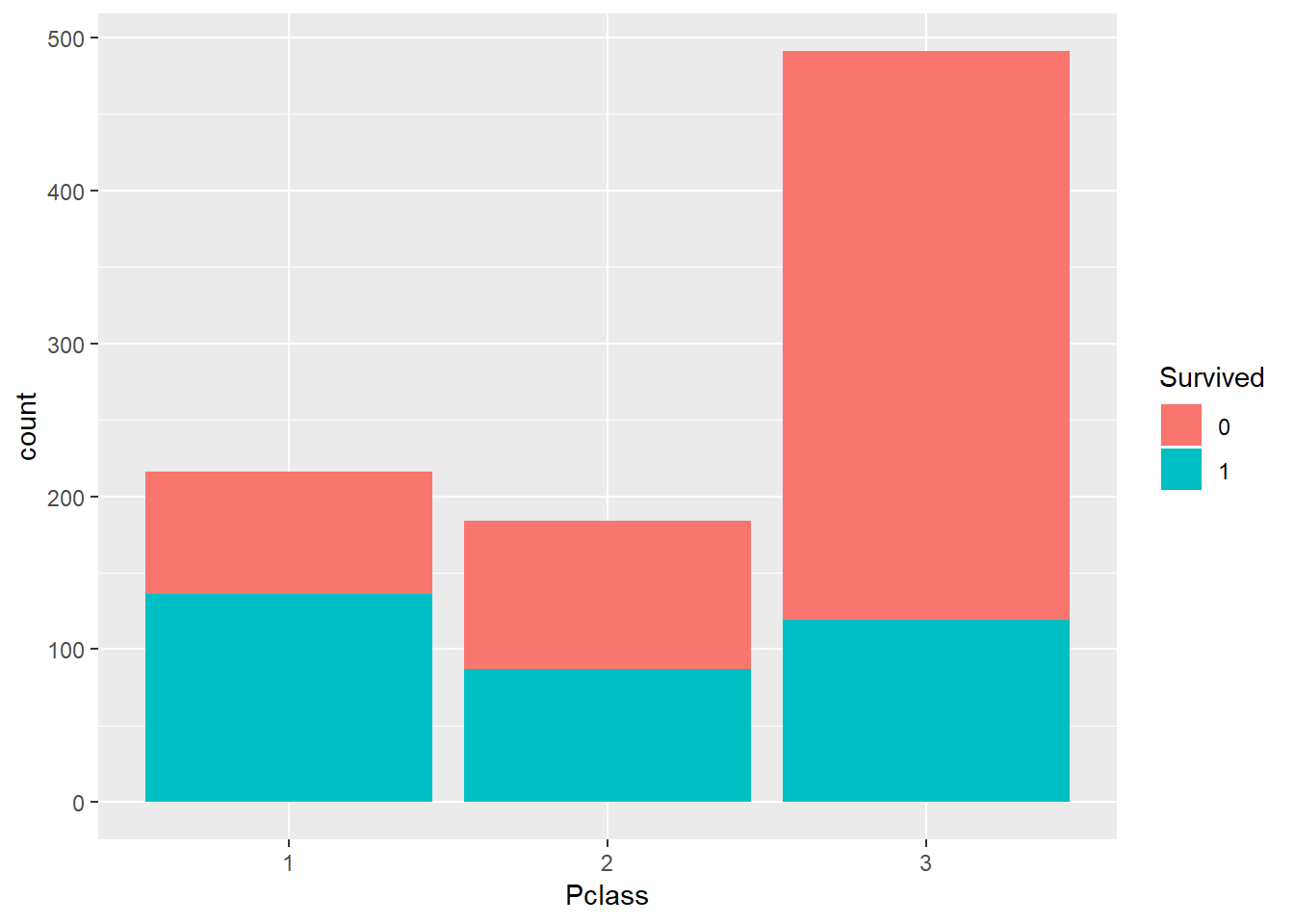
### 舱位等级和存活的关系

把舱位等级作为自变量，观察1、2、3三种等级的舱位在Survived特征上0（未存活）和1（存活）的频数和比例。

train%>%

ggplot()+

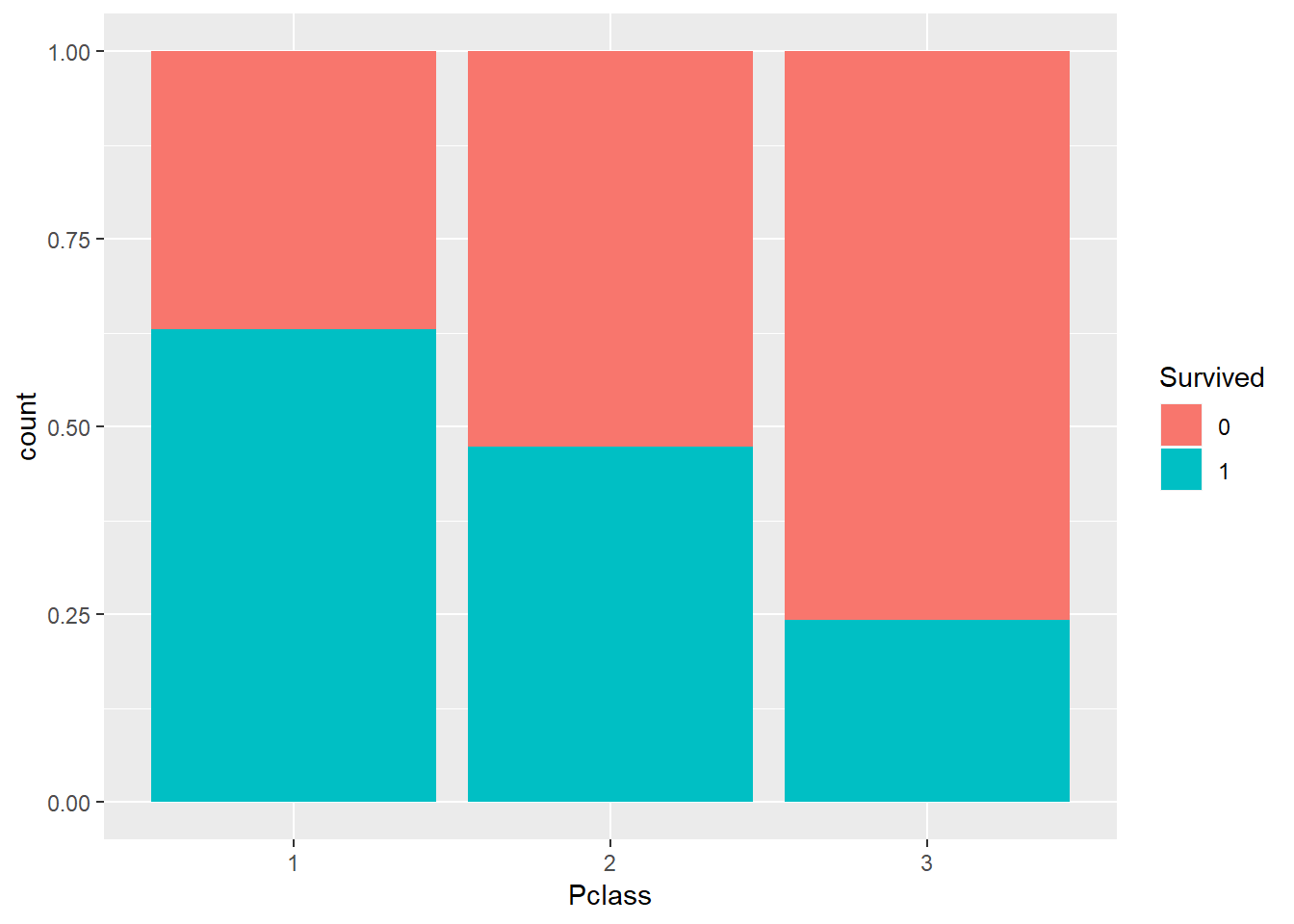
geom\_bar(aes(x=Pclass,fill = Survived))



train%>%

ggplot()+

geom\_bar(aes(x=Pclass,fill = Survived),position = 'fill')



可以看出三个舱位等级的数量和存活率有明显的区别，一等舱的人数在200左右，存活率超过60%，二等舱的人数不到200，存活率接近50%，三等舱的人数高达500人左右，存活率却只有25%左右，随着舱位等级的降低，存活率也在不断下降。舱位等级越高票价越贵，能坐得起一等舱的说明该乘客的经济条件较好，所以在救援时可能更容易获救，毕竟救援人员也不会和钱过不去嘛。

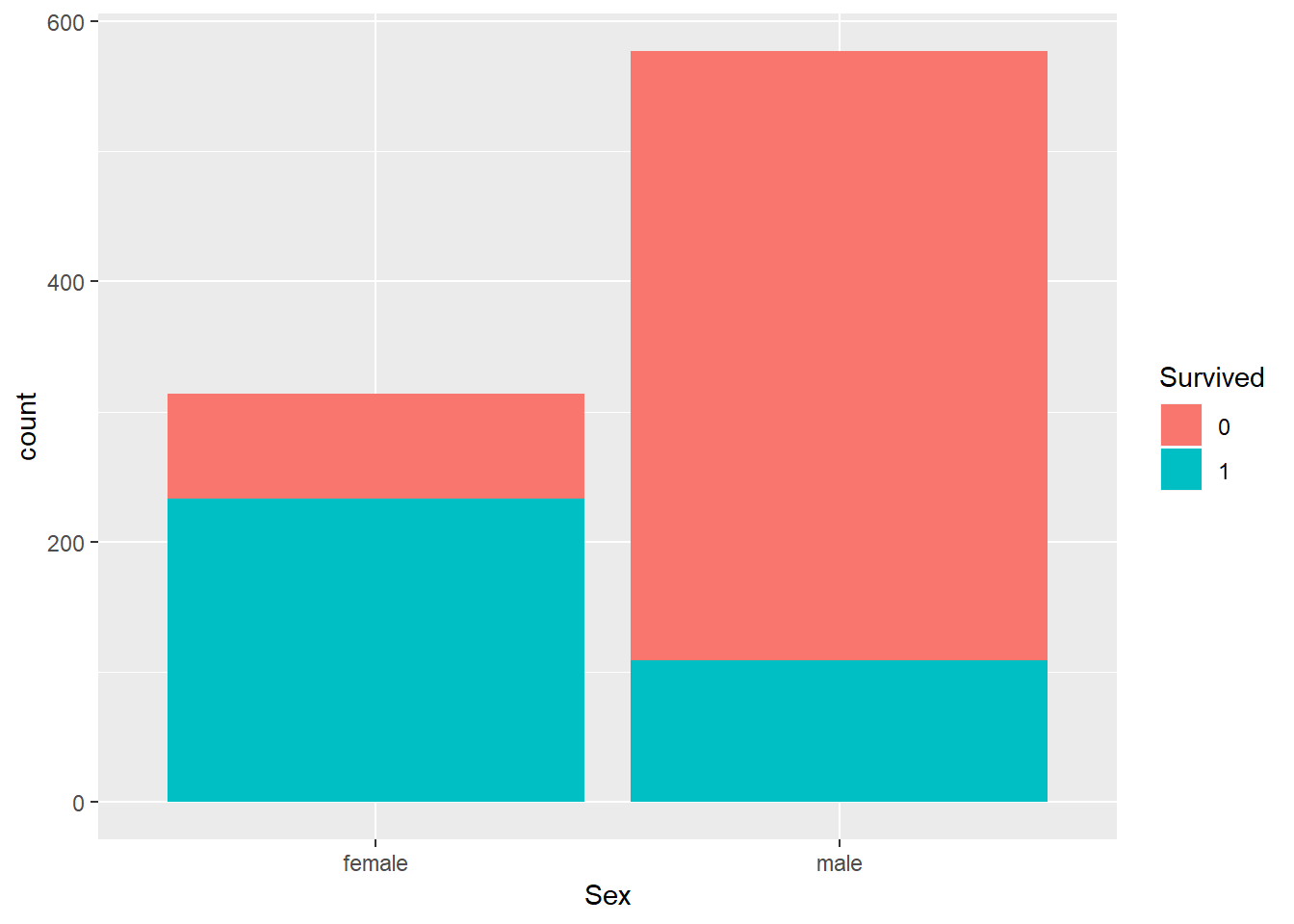
### 性别与存活的关系

把性别特征作为自变量，观察male和female两种类别在Survived特征上0（未存活）和1（存活）的频数和比例。

train%>%

ggplot()+

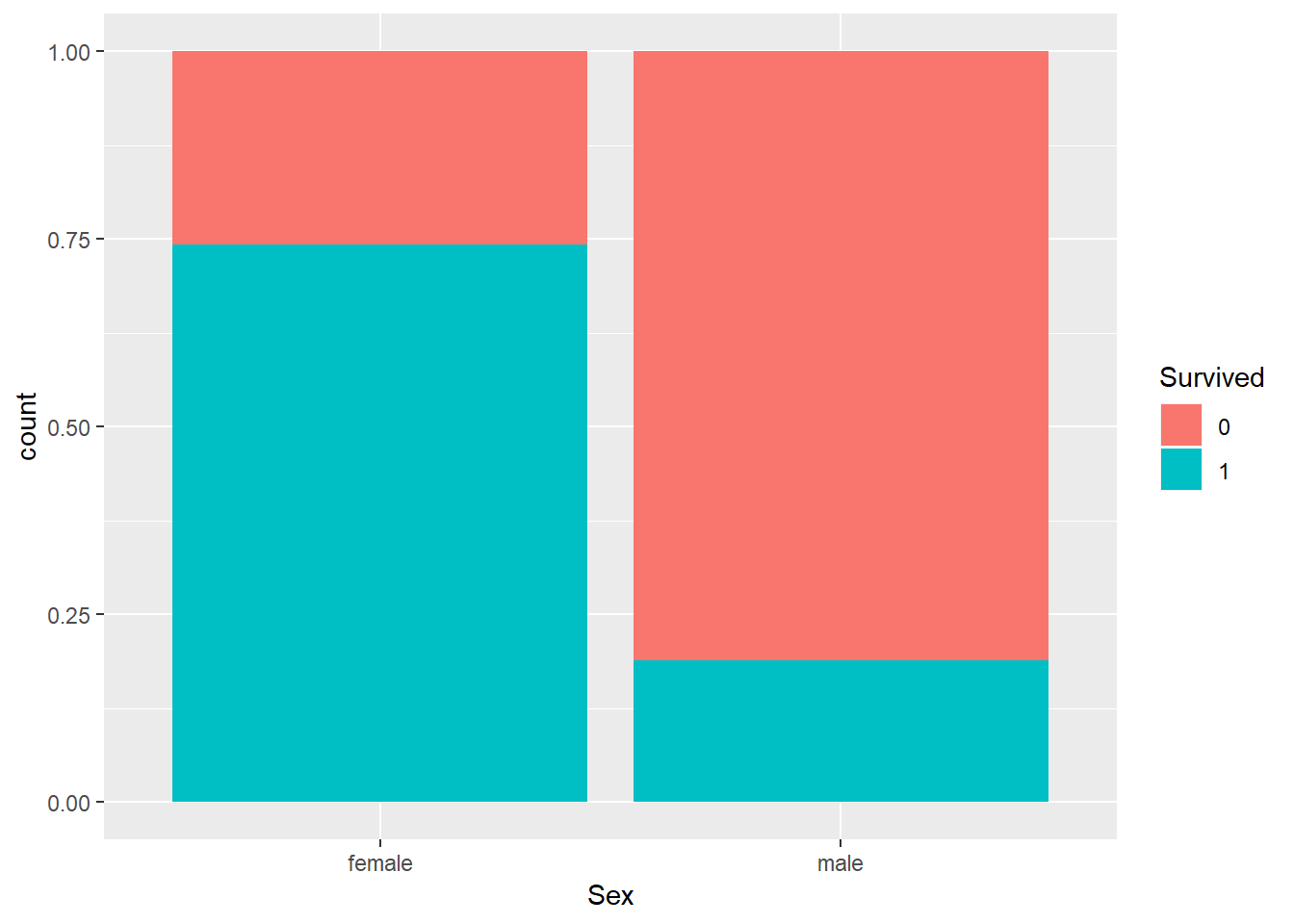
geom\_bar(aes(x=Sex,fill = Survived))



train%>%

ggplot()+

geom\_bar(aes(x=Sex,fill = Survived),position = 'fill')



可以看到女性和男性的数量和存活率差距很大，男性的数量接近600人，而女性只有300人左右。但是女性获救的人数却超过300，存活率接近75%，男性获救的人数只有100左右，存活率不超过25%。看来性别对存活与否影响很大，女性获救的概率要远远高于男性获救的概率，这也比较符合我们常识中女士优先的原则。

### 登船港口与存活的关系

把登船港口作为自变量，观察不同登船港口在Survived特征上0（未存活）和1（存活）所占的频数和比例，该特征存在两个缺失值，相对总体数据来说影响可以忽略不记，所以用众数对缺失值进行填充。

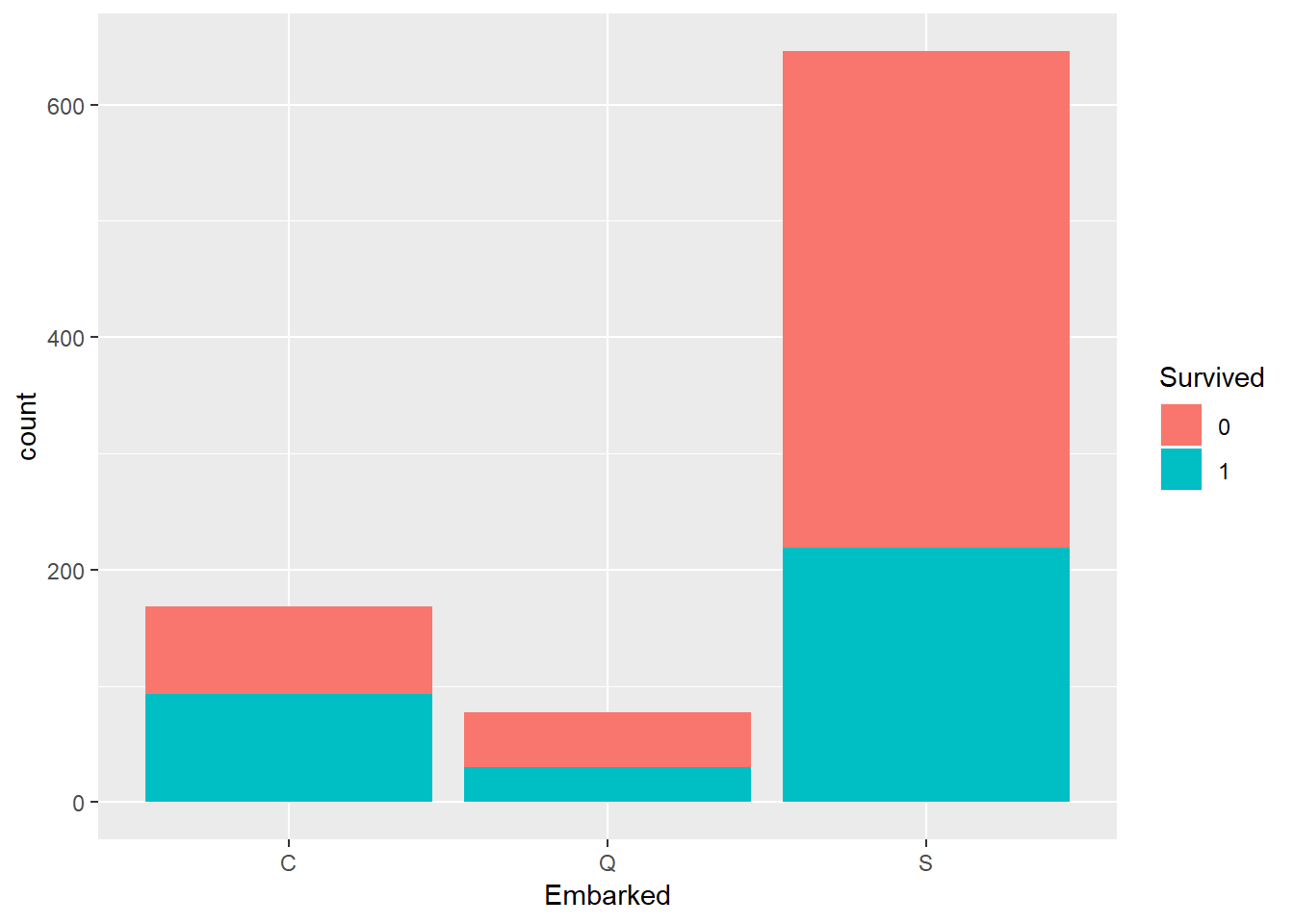
train$Embarked[is.na(train$Embarked)]=

names(table(train$Embarked)[which.max(table(train$Embarked))])

train%>%

ggplot()+

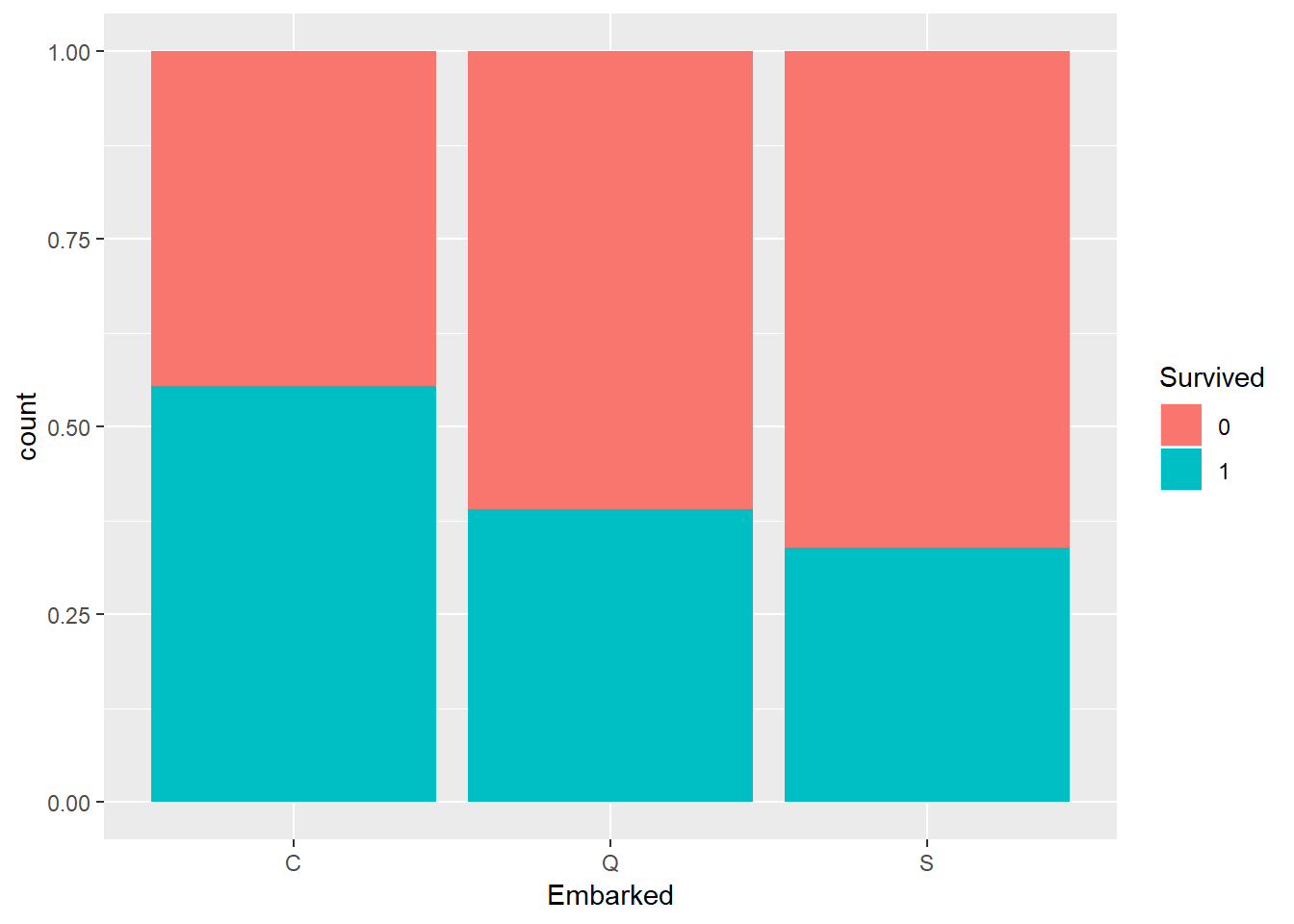
geom\_bar(aes(x=Embarked,fill = Survived))



train%>%

ggplot()+

geom\_bar(aes(x=Embarked,fill = Survived),position = 'fill')



从图中可以看到，C登船港口的人数接近200，存活率超过50%，Q登船港口人数不到100，存活率为37.5%左右，S登船港口足足超过600人，但是存活率却是最低的，只有35%左右。看来C登船港口的乘客比较幸运，能在C登船港口登船的乘客最有可能获救。 以上就是简单的三个因子变量对存活情况的影响，接下来我们分析其他特征对存活情况的影响。

### 年龄与存活的关系

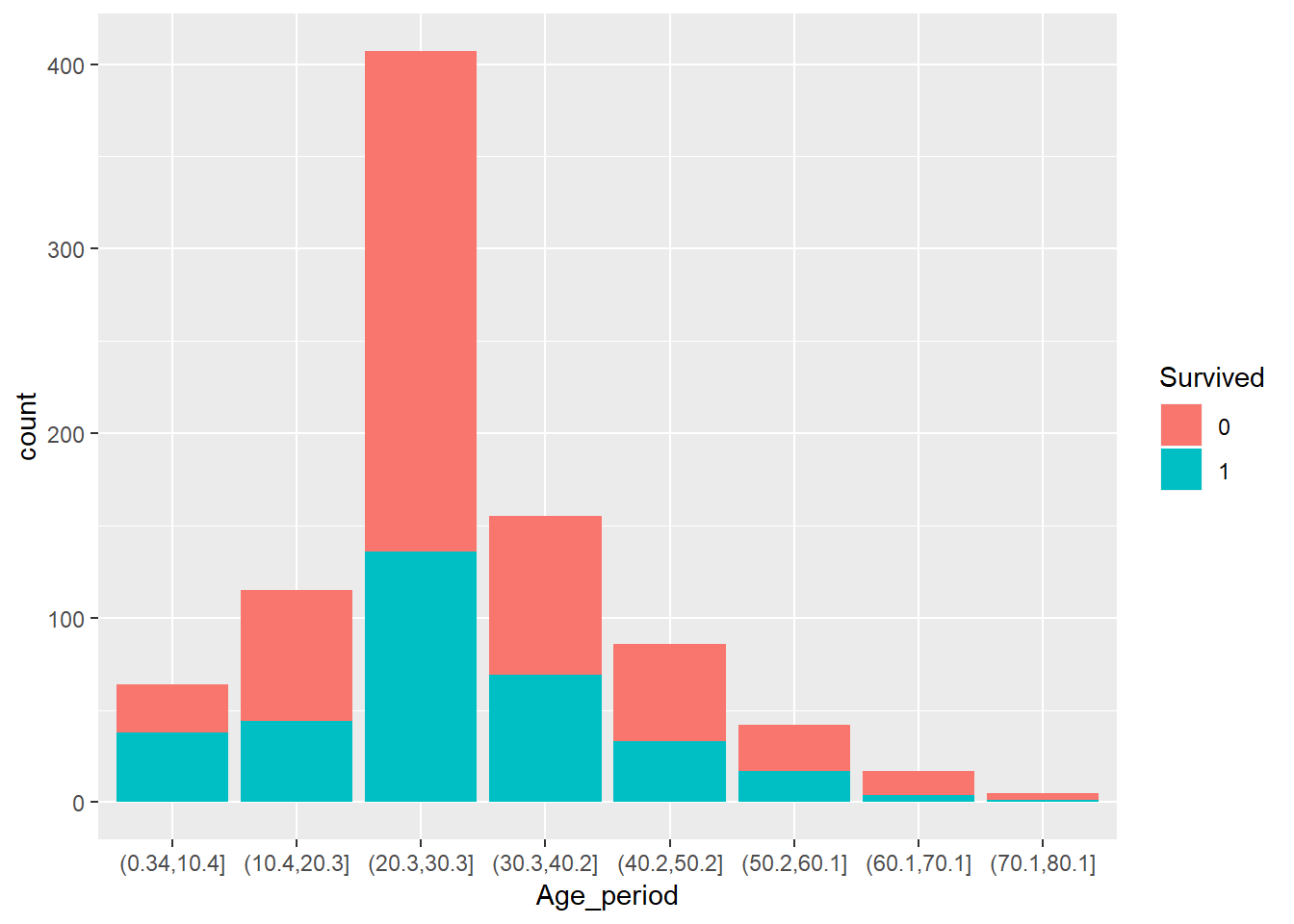
年龄特征的缺失值较多，但是前面分析可知年龄的分布较为均匀，因此采用均值对年龄特征的缺失值进行填充,又由于年龄是连续变量，所以把年龄分为8组，组距为10，比较不同年龄段下存活情况。

train$Age[is.na(train$Age)]=mean(train$Age,na.rm = T)

train%>%mutate(Age\_period=cut(train$Age,8))%>%

ggplot()+

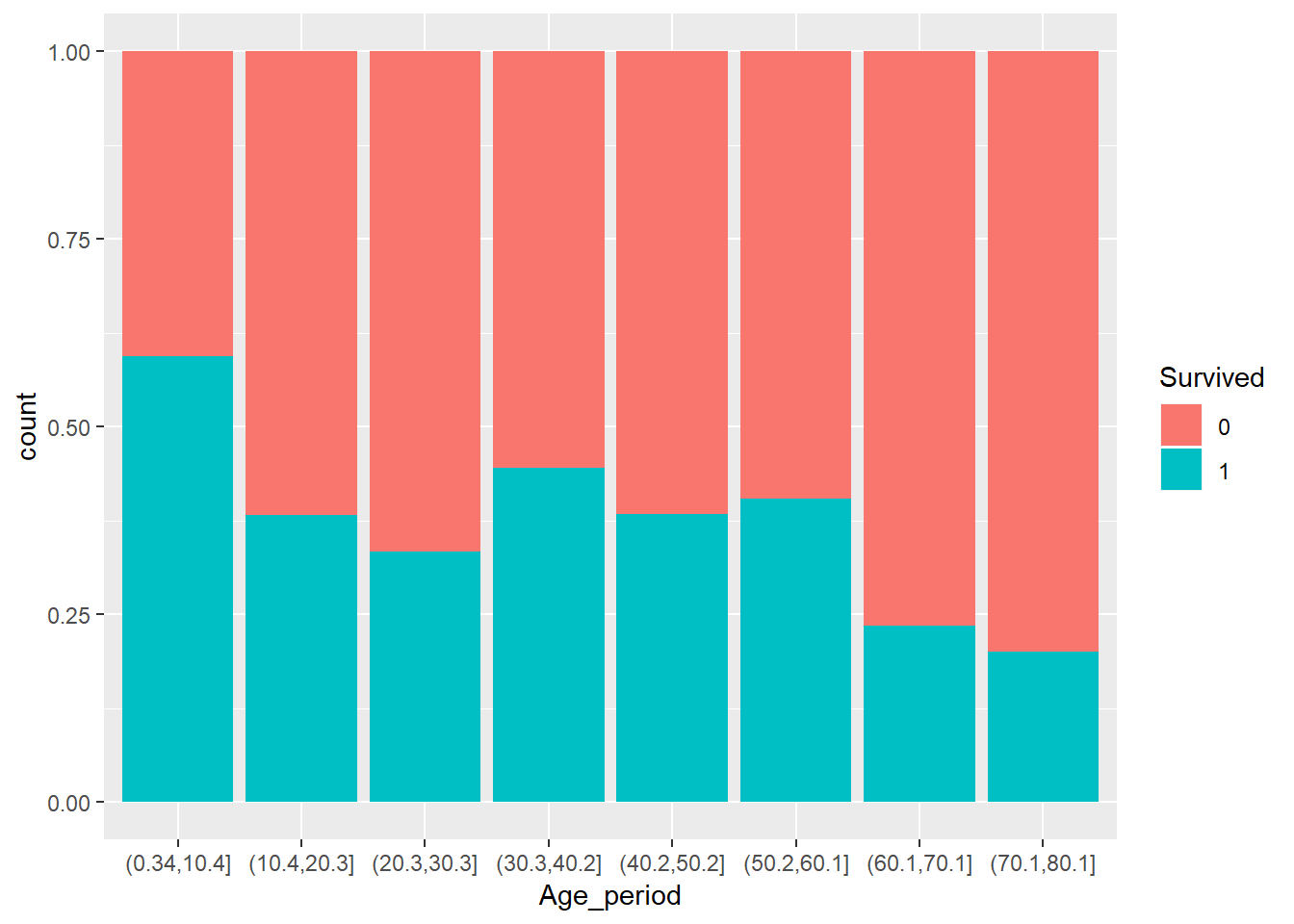
geom\_bar(aes(x=Age\_period,fill=Survived))



train%>%mutate(Age\_period=cut(train$Age,8))%>%

ggplot()+

geom\_bar(aes(x=Age\_period,fill=Survived),position = 'fill')



可以看到10岁以下的孩子存活率最高，60岁以上的老人存活率是最低的,，其他年龄段中20-30岁的存活率是最低的。

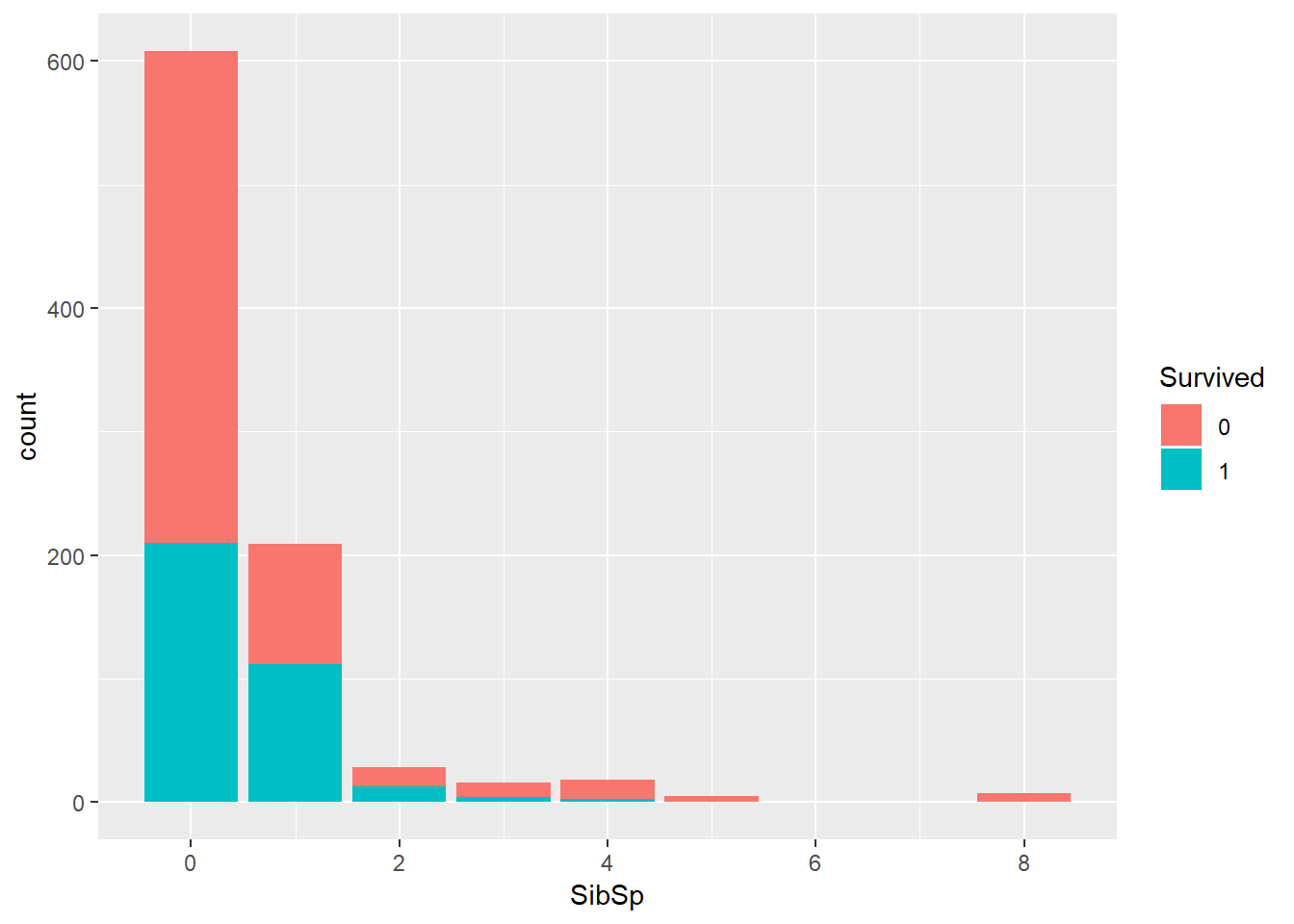
### SibSp与存活的关系

SibSp表示的是同在船上的兄弟姐妹/配偶数量，把不同数量作为因子变量比较不同数量下的存活情况。

train%>%

ggplot()+

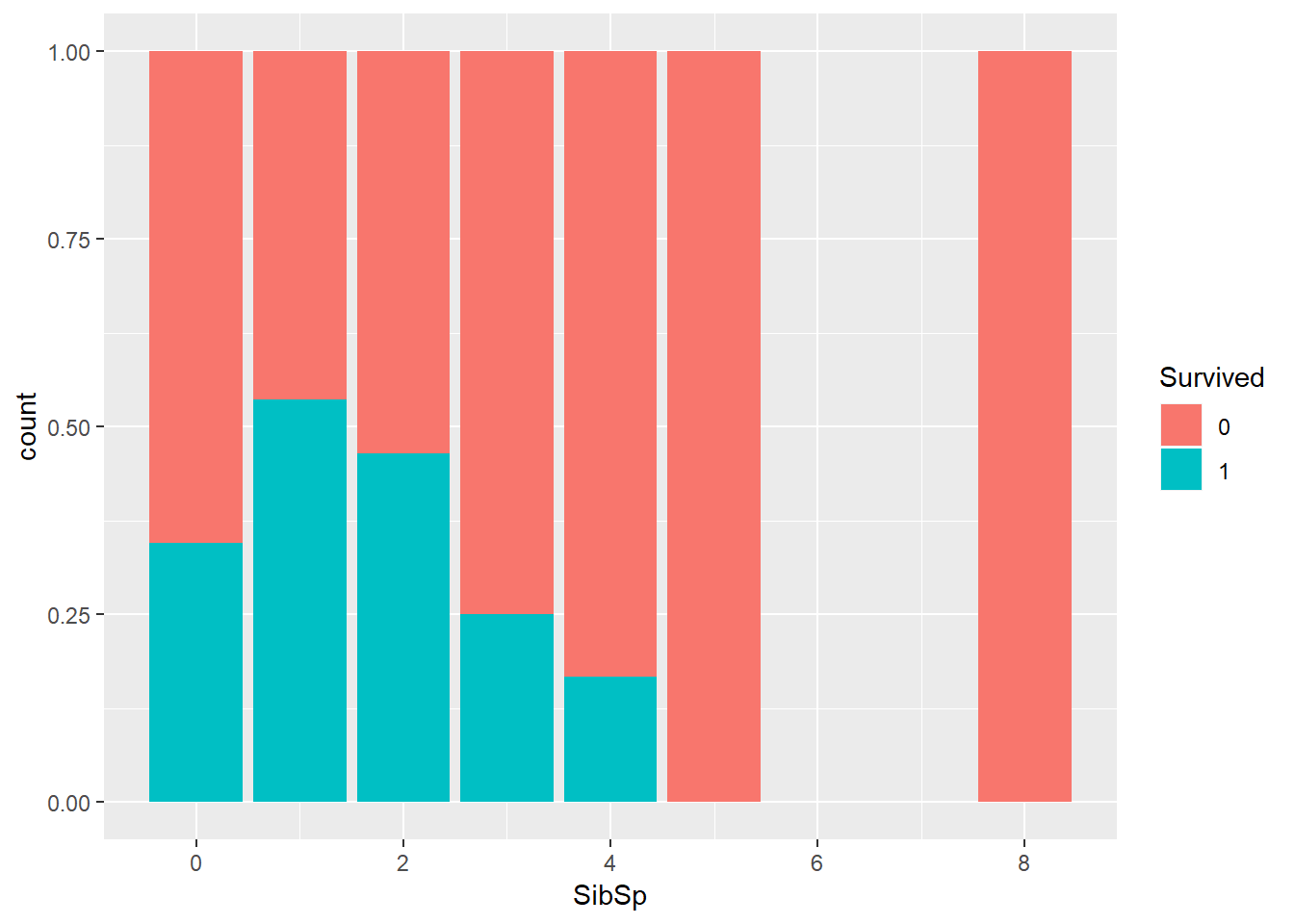
geom\_bar(aes(x=SibSp,fill = Survived))



train%>%

ggplot()+

geom\_bar(aes(x=SibSp,fill = Survived),position = 'fill')



 只有一个或两个兄弟姐妹/配偶的乘客存活率最高，接近50%，在数量大于2时，随着兄弟姐妹/配偶数量的增加，存活率不断降低，当数目达到5个和8个时，存活率居然为0。可能是一个大家庭都不想分开，所以一起坚守在船上直至沉没。

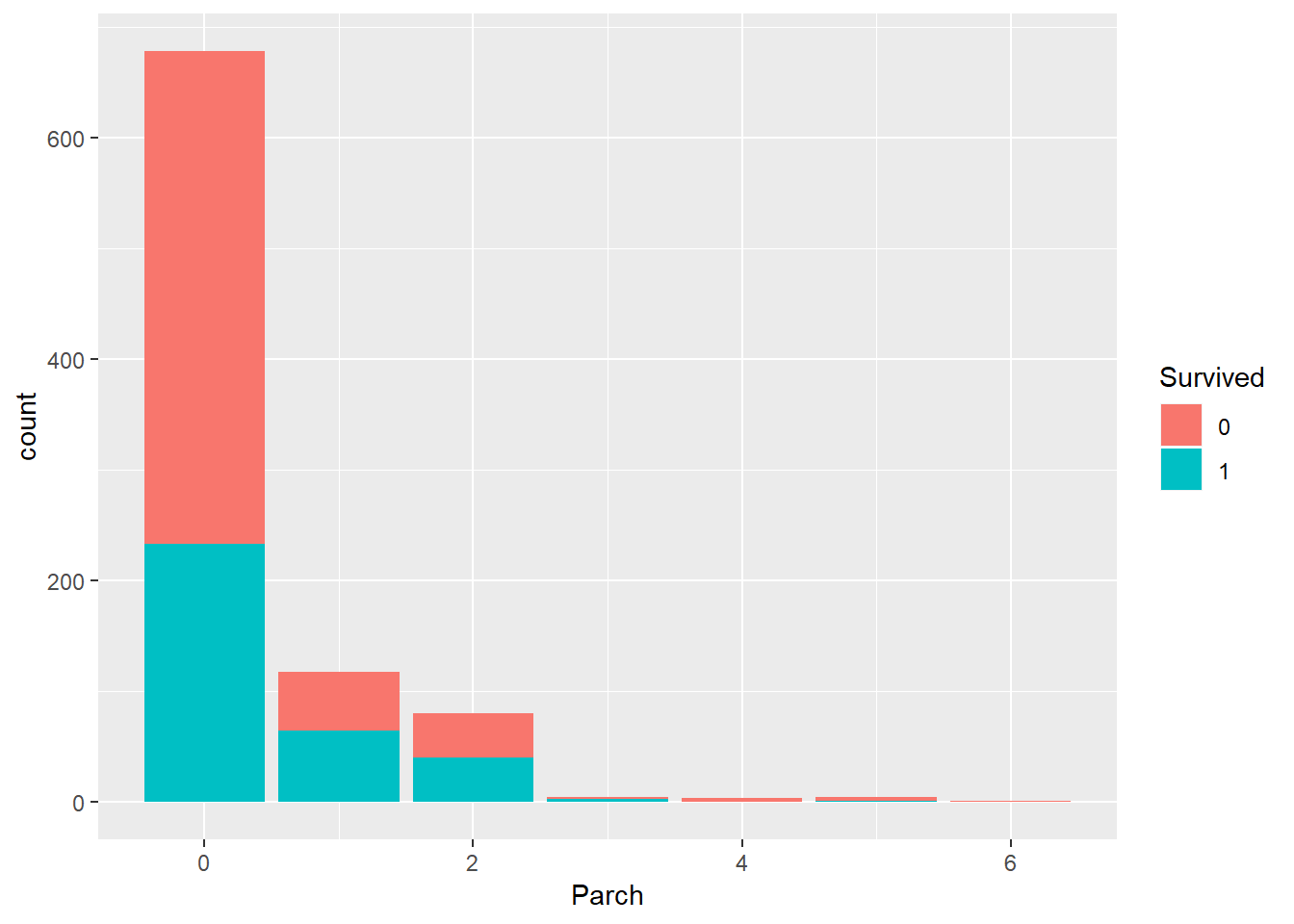
### Parch与存活的关系

Parch特征与SibSp有点相似，表示的是同在船上的父母/小孩数量，因此采取同样的方法处理SibSp。

train%>%

ggplot()+

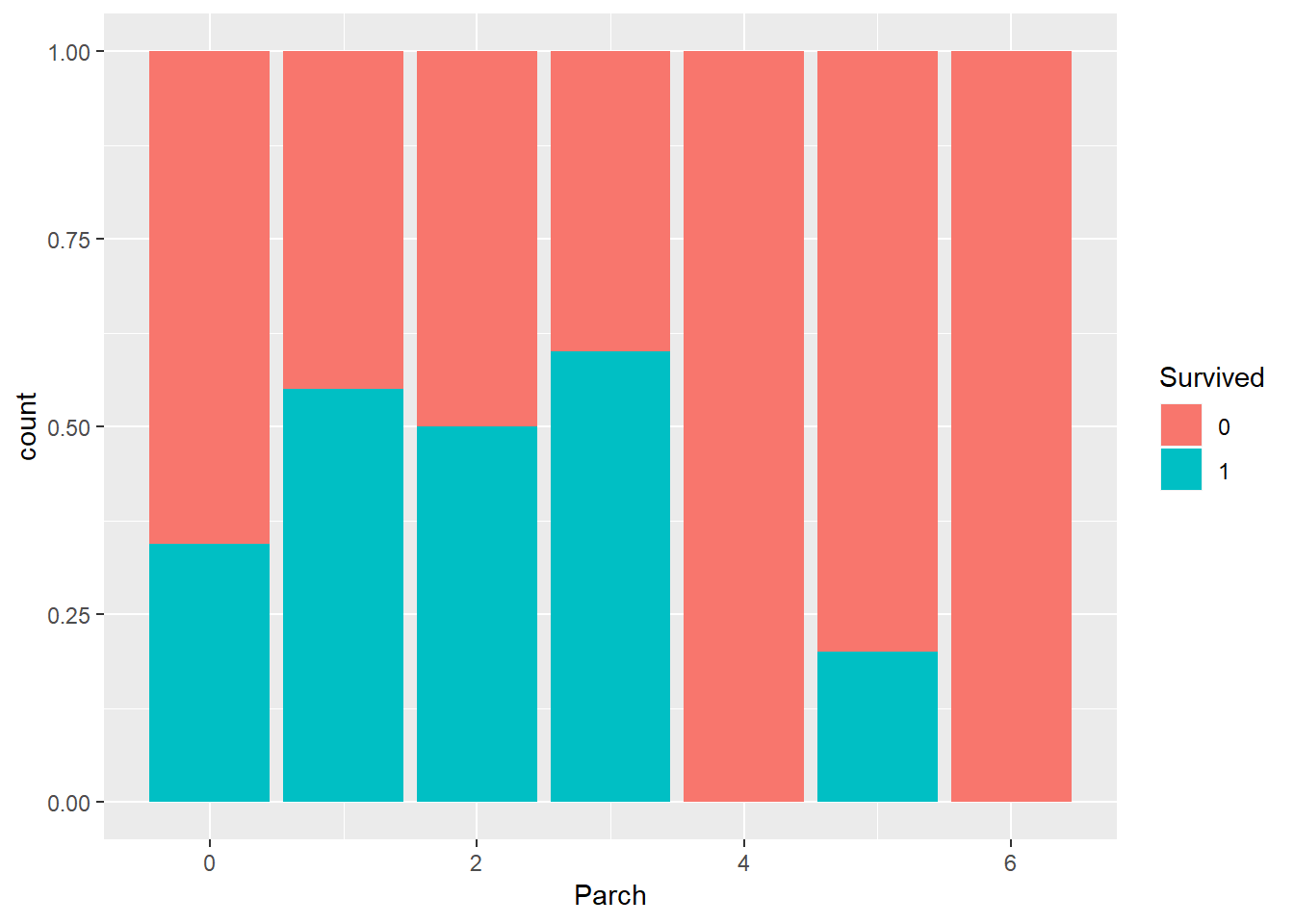
geom\_bar(aes(x=Parch,fill = Survived))



train%>%

ggplot()+

geom\_bar(aes(x=Parch,fill = Survived),position = 'fill')



尽管含义和SibSp有点近似，但是二者对存活情况的影响却不尽相同。同在船上的父母/小孩数量为5个时存活率最高，接近60%，而数量为4个或6个时存活率为0。

### 船票价格和存活的关系

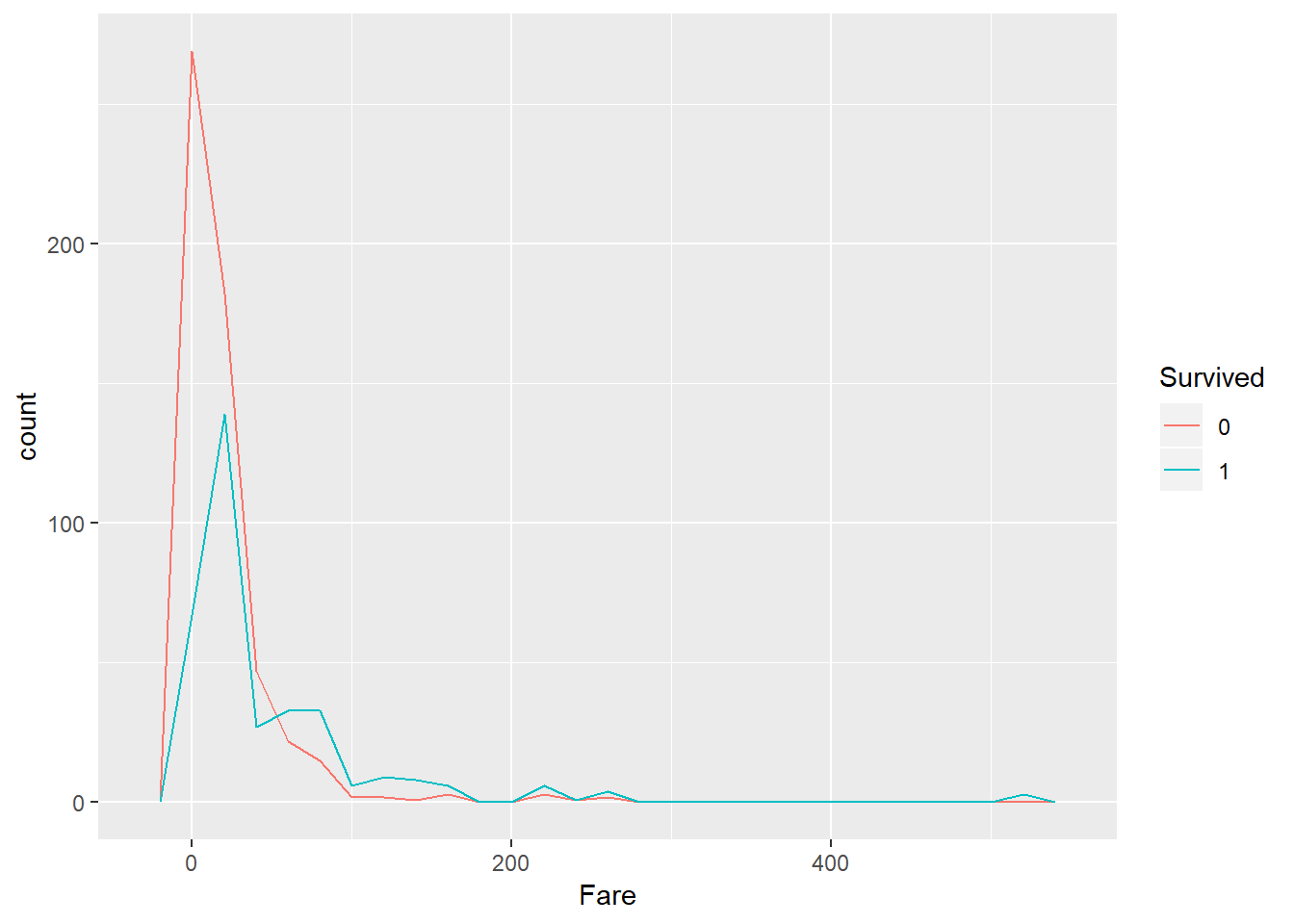
船票价格特征有一个缺失值，由于船票价格为右偏分布，因此采用中位数对缺失值进行填充。由于船票价格分布过于右偏，因此不对其分组处理，而是采用折线形式的直方图和箱型图来对其进行展示,并在箱型图上加上价格的散点图。

train$Fare=ifelse(is.na(train$Fare),median(train$Fare,na.rm = T),train$Fare)

train%>%

ggplot()+

geom\_freqpoly(aes(x=Fare,color=Survived),binwidth=20)

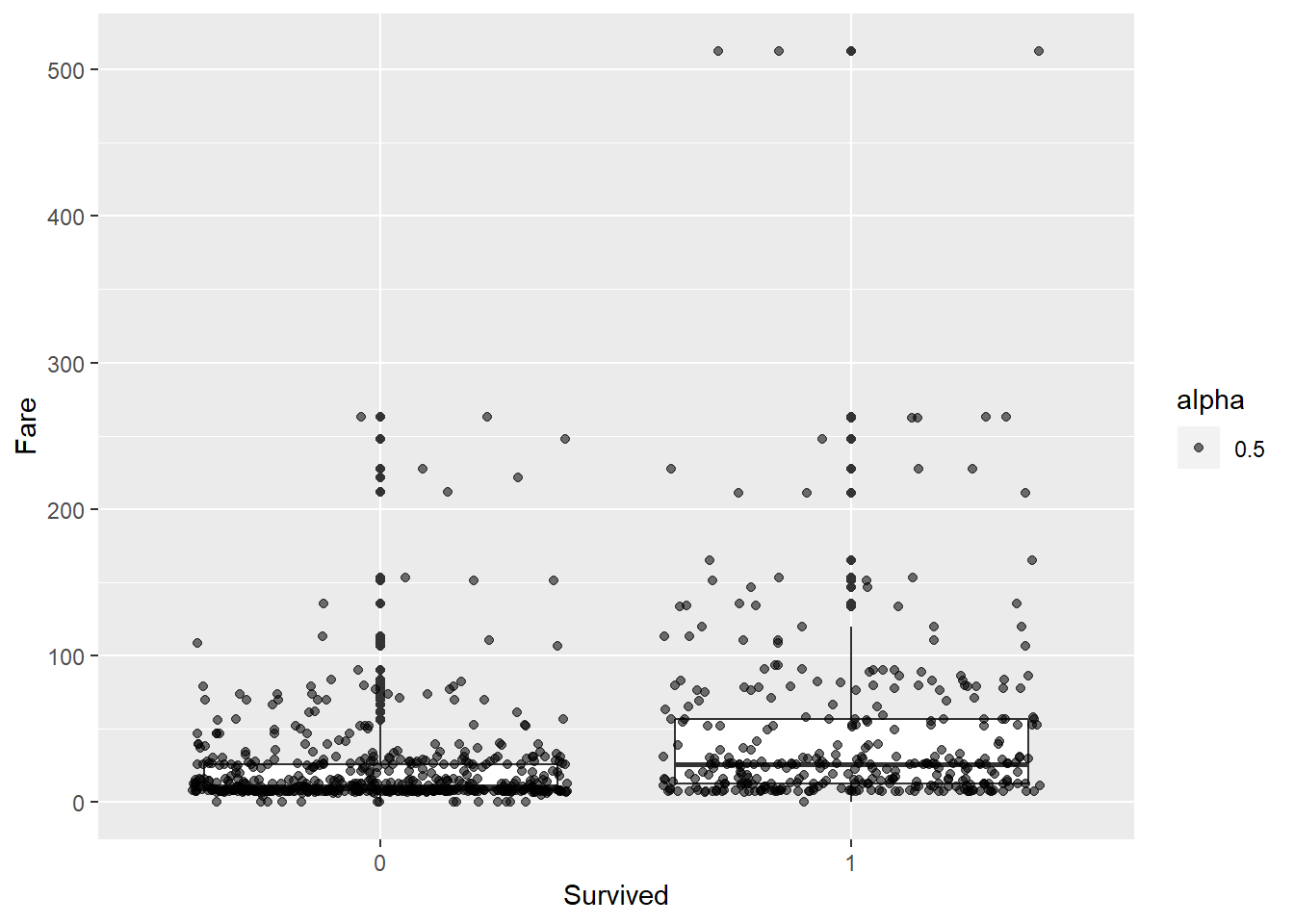


train%>%

ggplot()+

geom\_boxplot(aes(x=Survived,y=Fare))+

geom\_jitter(aes(x=Survived,y=Fare,alpha=.5))



可以看到存活曲线的船票价格集中区比未存活曲线的船票价格集中区要大，而在箱型图中，存活乘客的船票价格中位数要大于未存活乘客的船票价格的中位数,而且右边在高端价位的点明显要大于左边的，看来票价高的乘客的存活的人要更多一些。

### 客舱号码与存活的关系

客舱号码的缺失值特别多，所以如果直接删除可能对结果影响较大，考虑到没有客舱号码可能是没有坐票只能呆在甲板上的乘客，因此采用分类的办法对其进行处理，把缺失的归类为0，未缺失的归类为1，然后把该特征转换成因子变量，分析1和0两类乘客的存活情况。

train$Cabin=ifelse(is.na(train$Cabin),0,1)

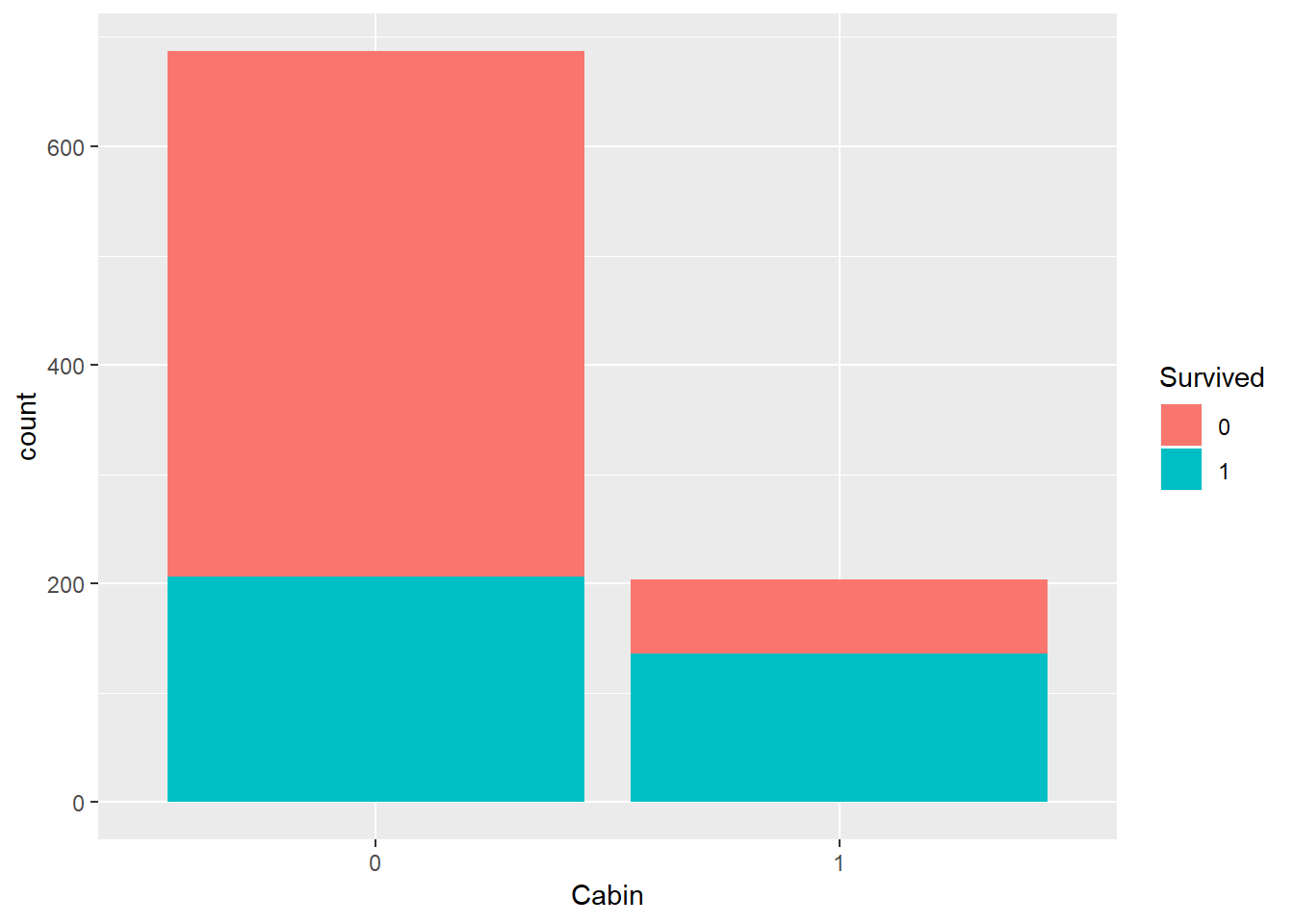
train=train%>%

mutate(Cabin=as.factor(Cabin))

train%>%

ggplot()+

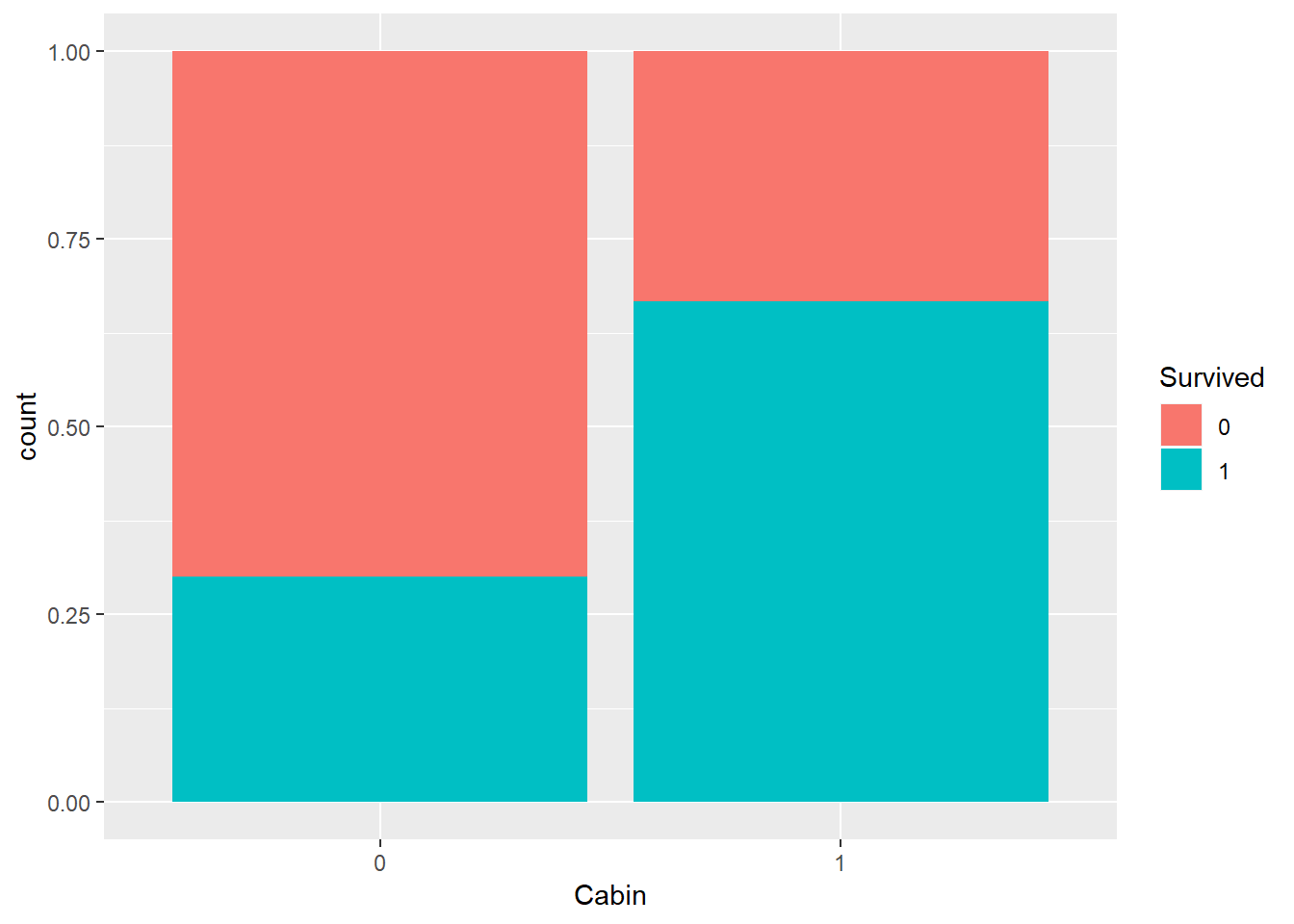
geom\_bar(aes(x=Cabin,fill=Survived))



train%>%

ggplot()+

geom\_bar(aes(x=Cabin,fill=Survived), position = 'fill')



从图中可以看到，0类乘客接近700人，但是存活率只有30%左右，而1类乘客只有200人左右，存活率却接近70%，两者的区别很大，看来该特征对存活情况的影响较大。

### 姓名与存活的关系

姓名每个人都有并且存在部分相似，对其观察发现名字的中间都有一个头衔，为了验证是否每个人都有头衔，采用正则表达式将中间的头衔提取出来，看个数是否与人数相等。

length(str\_extract(train$Name,"[A-Z][a-z]+\\."))==891

## [1] TRUE

结果为TRUE，验证了这个猜想。接下来把每个人的的头衔都提取出来然后对其进行频数统计看看分布如何。

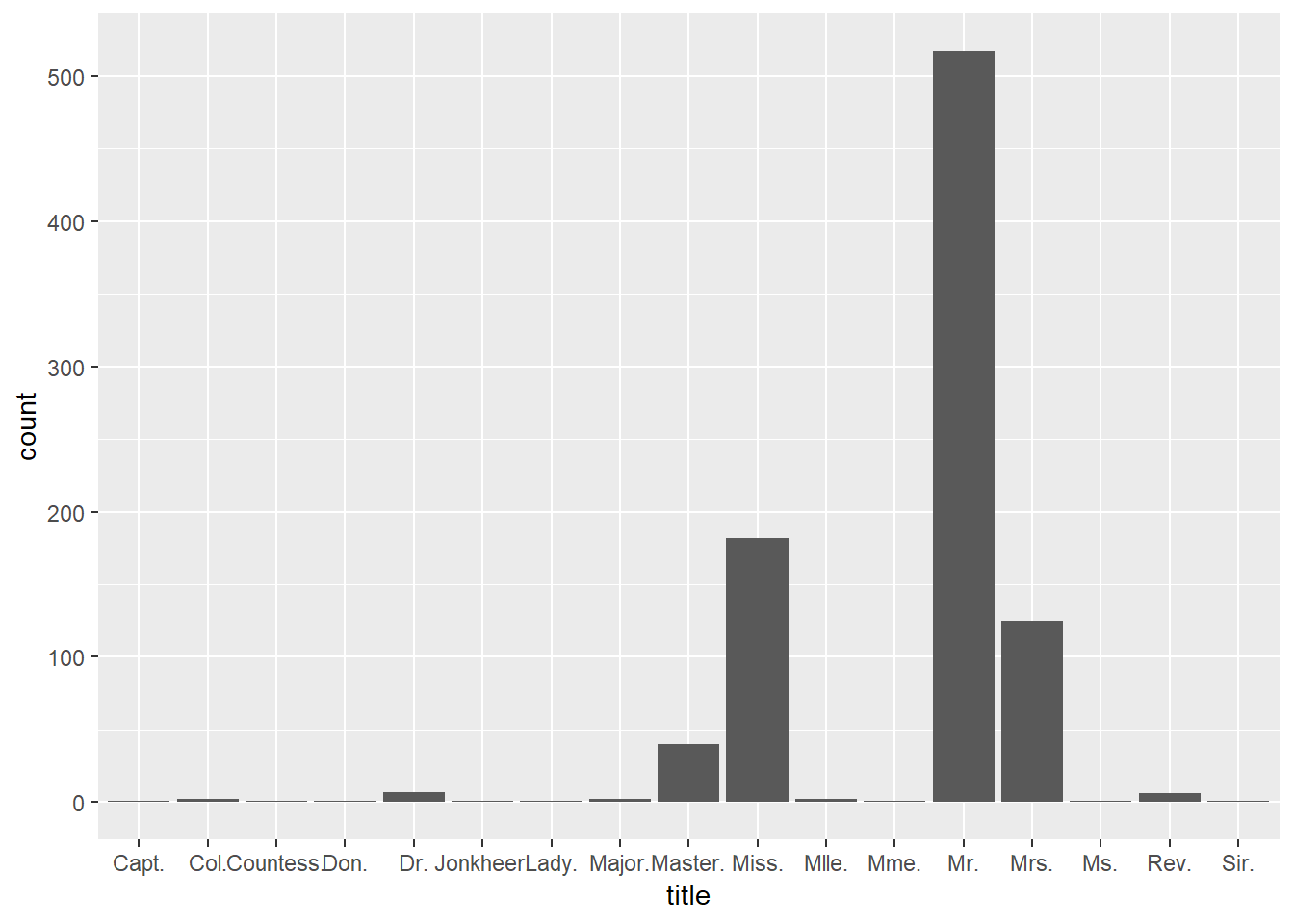
train=train%>%

mutate(title=str\_extract(Name,"[A-Z][a-z]+\\."))

train%>%

ggplot()+

geom\_bar(aes(x=title))



可以看到Master、Miss、Mr、Mrs四种头衔的个数最多，而其他头衔的个数少的可怜，因此可以把头衔归成五类，除了前面四种的其他头衔归类为Other，然后分析不同头衔的乘客的存活情况。

train$title=ifelse(train$title %**in**% c('Master.','Miss.','Mr.','Mrs.'),

train$title,'Other')

train%>%

mutate(title=as.factor(title))

## # A tibble: 891 x 13

## PassengerId Survived Pclass Name Sex Age SibSp Parch Ticket Fare

## <int> <fct> <fct> <chr> <fct> <dbl> <int> <int> <chr> <dbl>

## 1 1 0 3 Brau~ male 22 1 0 A/5 2~ 7.25

## 2 2 1 1 Cumi~ fema~ 38 1 0 PC 17~ 71.3

## 3 3 1 3 Heik~ fema~ 26 0 0 STON/~ 7.92

## 4 4 1 1 Futr~ fema~ 35 1 0 113803 53.1

## 5 5 0 3 Alle~ male 35 0 0 373450 8.05

## 6 6 0 3 Mora~ male 29.7 0 0 330877 8.46

## 7 7 0 1 McCa~ male 54 0 0 17463 51.9

## 8 8 0 3 Pals~ male 2 3 1 349909 21.1

## 9 9 1 3 John~ fema~ 27 0 2 347742 11.1

## 10 10 1 2 Nass~ fema~ 14 1 0 237736 30.1

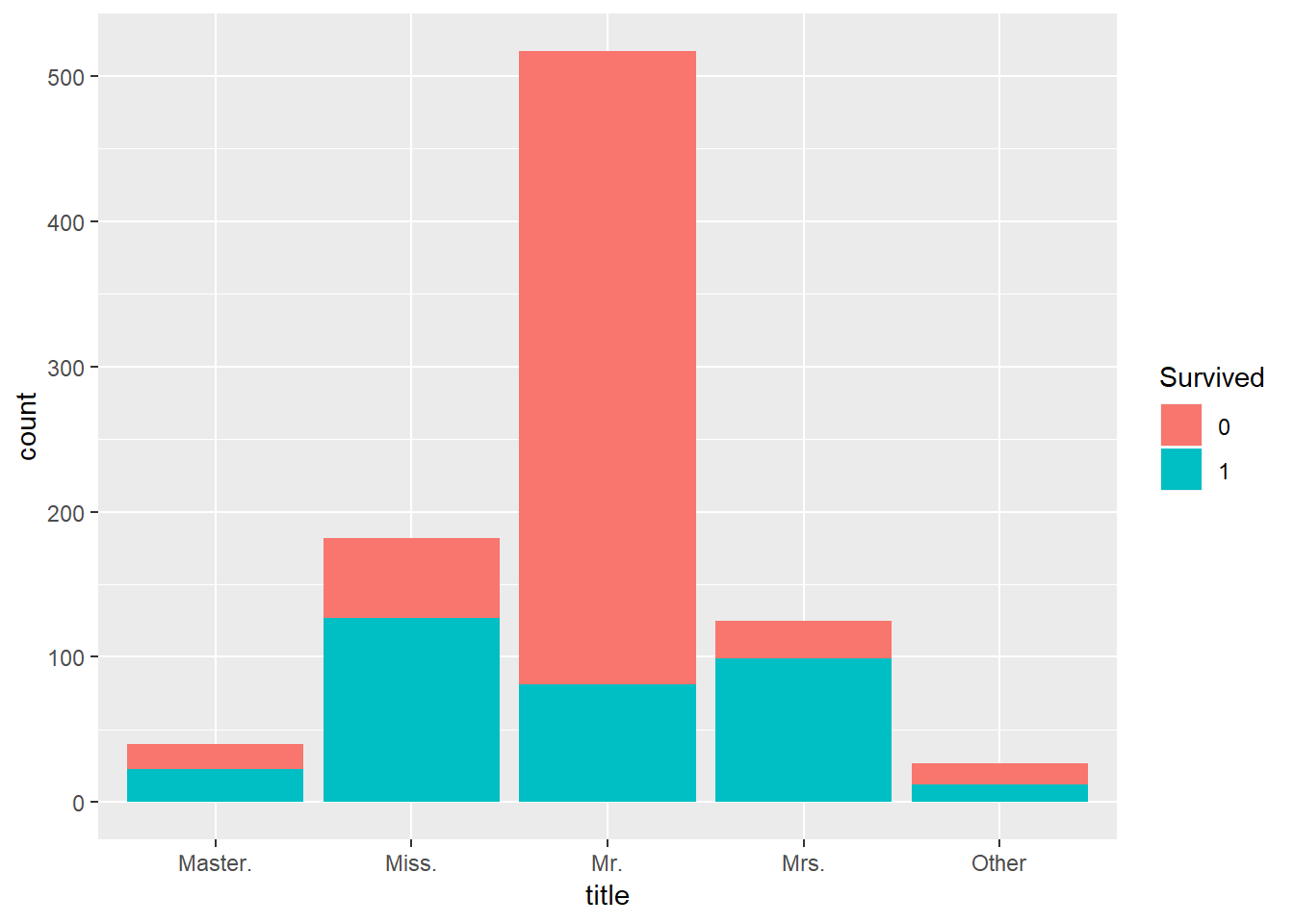
## # ... with 881 more rows, and 3 more variables: Cabin <fct>,

## # Embarked <fct>, title <fct>

train%>%

ggplot()+

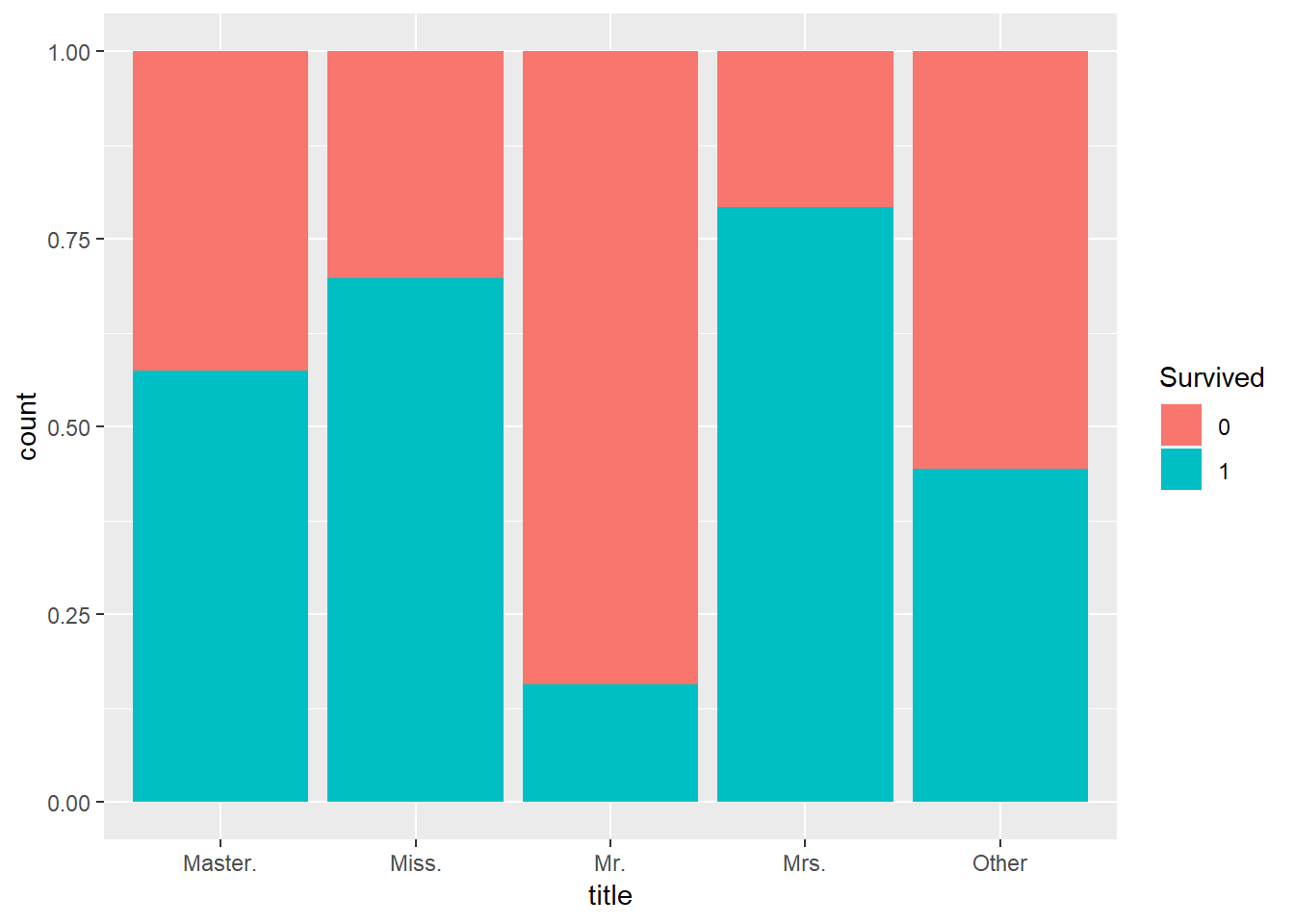
geom\_bar(aes(x=title,fill=Survived))



train%>%

ggplot()+

geom\_bar(aes(x=title,fill=Survived),position = 'fill')



含有Mr头衔的乘客最多，超过了500人，但是存活率却最低，只有13%左右。含有Mrs头衔的乘客存活率最高，超过了75%，Miss头衔的乘客存活率其次，接近70%。Mr的实际含义代表的是先生，而Mrs和Miss代表的是夫人和小姐，看来女性的存活率确实很高。 以上都是单个特征对存活情况的分析，但是实际上存活情况是受多个因素共同影响的，因此我们需要了解某个特征在另外一个或几个特征不同类别下对因变量的影响，这样才能更好的分析不同特征之间的关系。

### 舱位等级在不同性别下与存活的关系

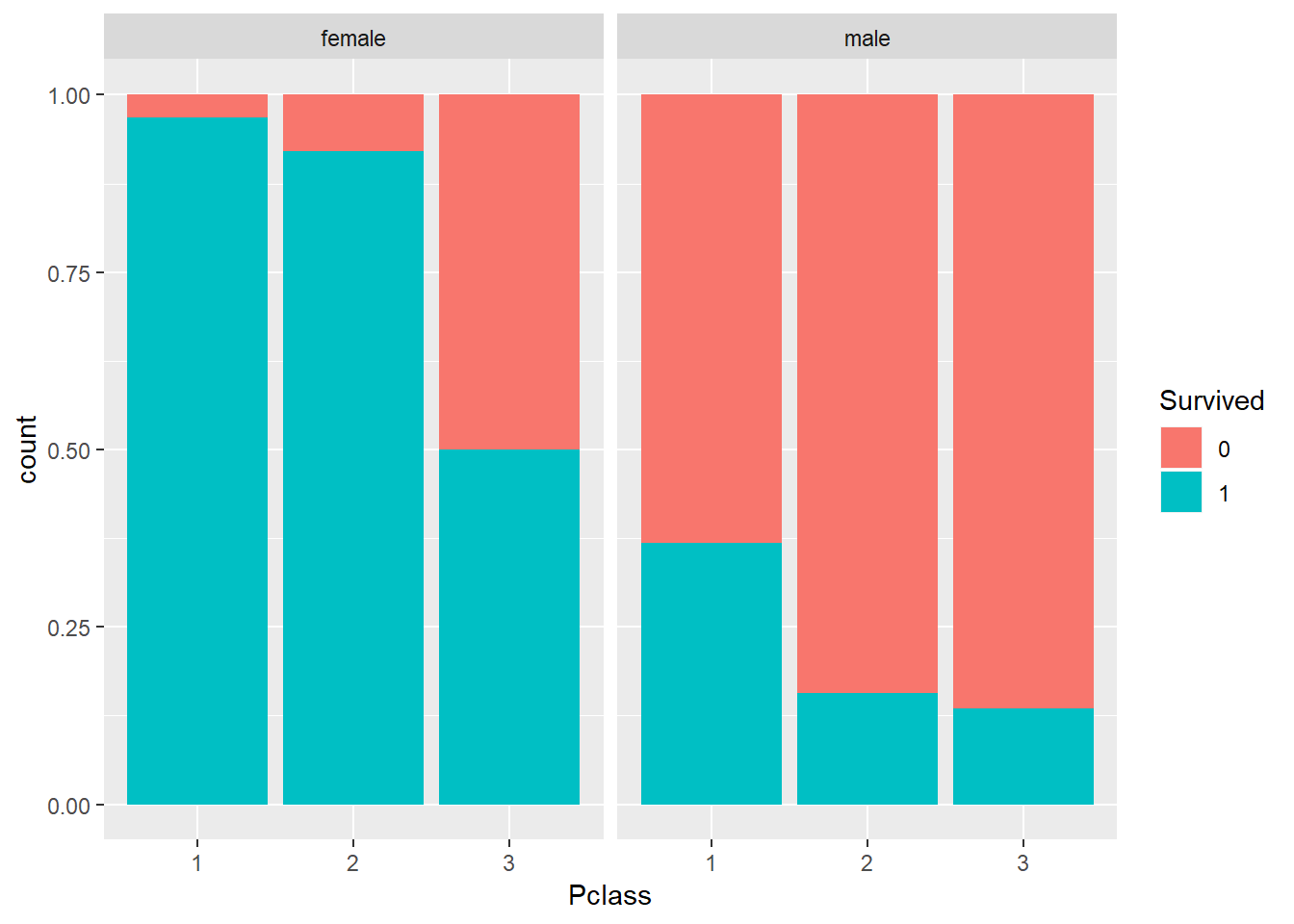
把舱位等级看成自变量，分成男性和女性两个版块观察不同船舱等级在不同性别下的存活情况。

train%>%

ggplot()+

geom\_bar(aes(x=Pclass,fill=Survived),position = 'fill')+

facet\_wrap(~Sex)

 一号舱和二号舱女性的存活率都特别高，特别是一号舱女性存活率接近100%，而三号舱女性的存活率就比较低，只有50%左右，但是相对于男性来说却很高，其中最高的一号舱的男性存活率只有37.5%，三号舱的男性最惨，平均每8个人只有1个能活下来。

### 登船港口在不同性别下与存活的关系

把登船港口看成自变量，分成男性和女性两个版块观察不同登船港口在不同性别下的存活情况。

train%>%

ggplot()+

geom\_bar(aes(x=Embarked,fill=Survived),position = 'fill')+

facet\_wrap(~Sex)

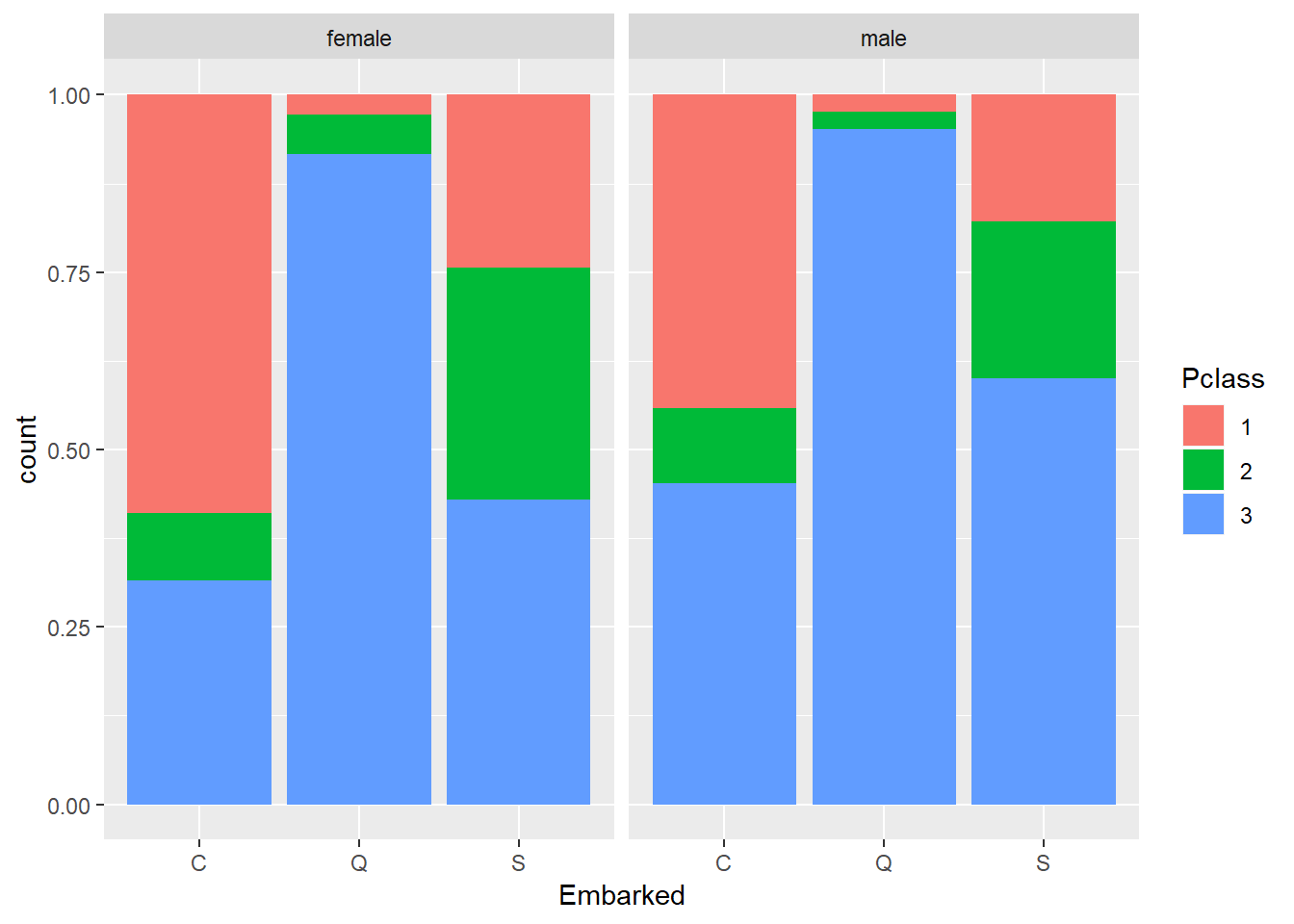
 C港口女性的存活率最高，其次是Q港口，再是S港口，一个有趣的现象是Q港口的男性存活率居然是最低的，于是便想到需要进一步分析这几个港口的乘客的船舱等级的分布，因为船舱等级代表的是乘客的经济条件。

train%>%

ggplot()+

geom\_bar(aes(x=Embarked,fill=Pclass),position = 'fill')+

facet\_wrap(~Sex)

 可以看到Q港口的乘客三等舱人数是最多的，而之前分析过三等舱的男性存活率是最低的，因此Q港口的男性存活率比较低也就不奇怪了。

### 是否有座在不同性别、不同船舱等级下与存活的关系

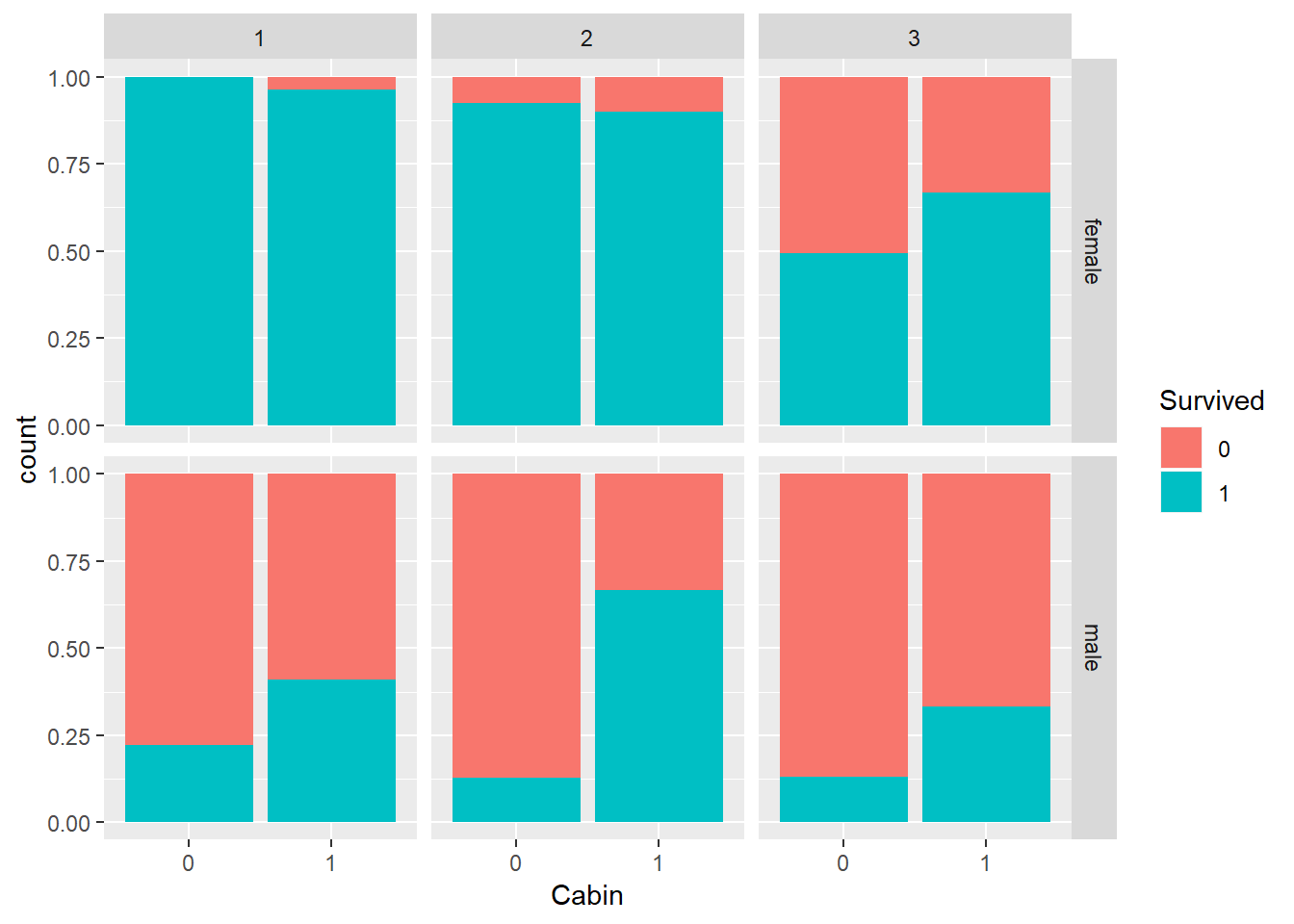
把Cabin作为自变量，观察乘客是否有座位在不同性别、不同船舱等级下的存活情况。

train%>%

ggplot(aes(x=Cabin))+

geom\_bar(aes(fill=Survived),position = 'fill')+

facet\_grid(Sex~Pclass)

 可以看到男性中，有座乘客的存活率要明显高于无座乘客存活率，而在女性中，一等舱和二等舱的无座乘客的存活率却比有座乘客的存活率搞，三等舱的规律和男性一样。

### 年龄在不同性别、不同船舱等级下与存活的关系

把年龄作为自变量，观察不同年龄段乘客在不同性别、不同船舱等级下的存活情况。

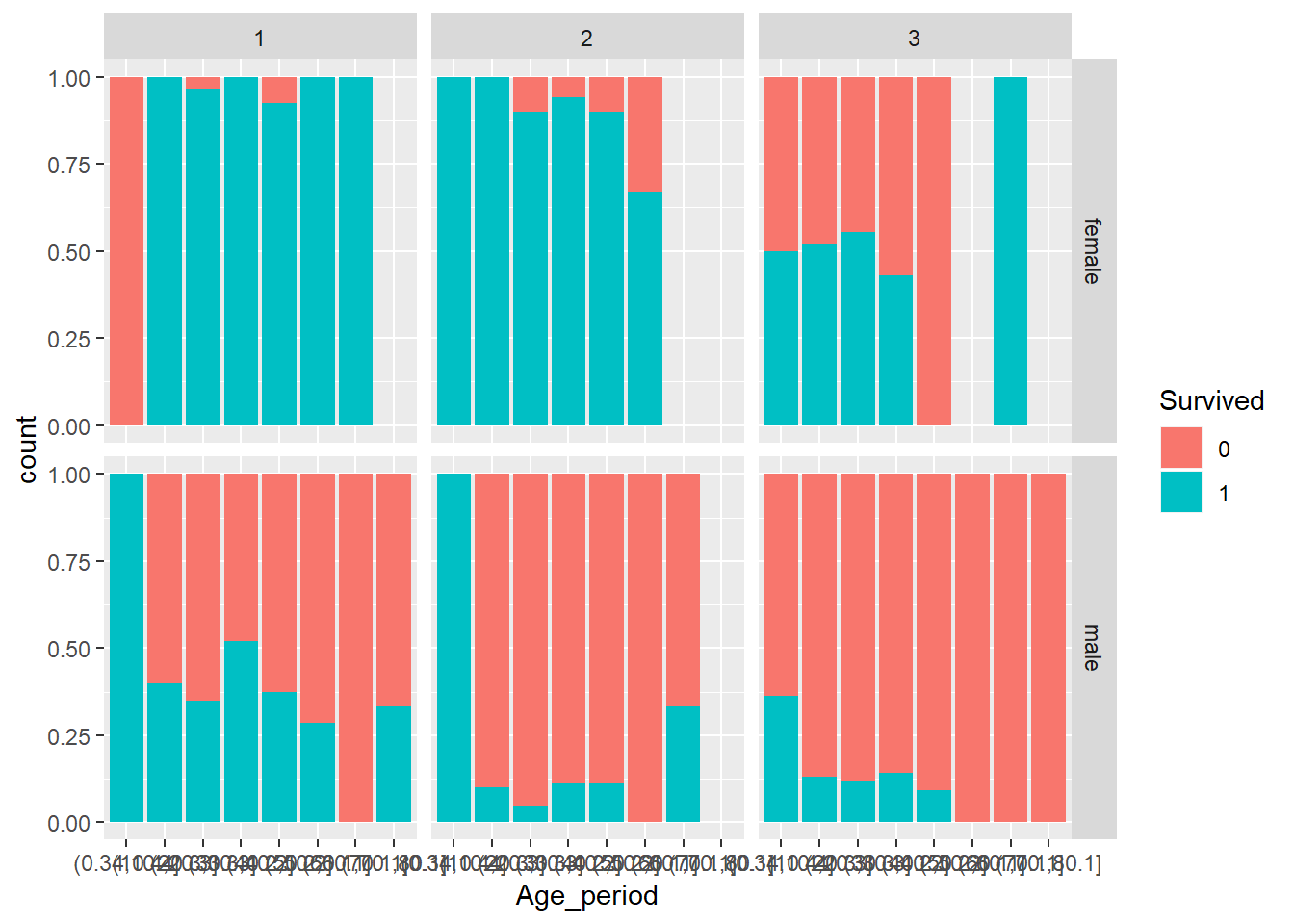
train%>%

mutate(Age\_period=cut(train$Age,8))%>%

ggplot(aes(x=Age\_period))+

geom\_bar(aes(fill=Survived),position = 'fill')+

facet\_grid(Sex~Pclass)

 可以看到10岁以下的乘客在各个等级的船舱以及不同性别下的存活比例都是最高的，除了一等舱的女性。随着船舱等级的降低，存活比例也在不断降低。

## 特征工程

数据和特征决定了机器学习的上限，而模型和算法只是逼近这个上限而已。这句话充分说明了特征工程的重要性，之前的可视化也是为了帮助理解数据，为特征工程服务。

### 缺失值处理

在数据分析中得知Age,Cabin,Embarked,Fare四个字段存在缺失值，在可视化的过程中也分别进行了处理，下面对训练数据和测试数据合并的表进行统一处理，首先对年龄特征缺失值用均值进行填充。

train\_test$Age[is.na(train\_test$Age)]=mean(train\_test$Age,na.rm = T)

对客舱号码特征进行分类处理，缺失的为0类，未缺失的为1类

train\_test$Cabin=ifelse(is.na(train\_test$Cabin),0,1)

对登船港口特征缺失值用众数进行填充

train\_test$Embarked[is.na(train\_test$Embarked)]=

names(table(train\_test$Embarked)[which.max(table(train\_test$Embarked))])

对船票价格缺失值用中位数进行填充

train\_test$Fare=ifelse(is.na(train\_test$Fare),

median(train\_test$Fare,na.rm = T),train\_test$Fare)

### 特征生成

首先对姓名进行处理，由前面的分析可知Master、Miss、Mr、Mrs四种头衔最多，其他的都很少，因此同前面一样，把所有姓名按照不同的头衔分为五类。

train\_test=train\_test%>%

mutate(Name=str\_extract(Name,"[A-Z][a-z]+\\."))

train\_test$Name=ifelse(train\_test$Name %**in**% c('Master.','Miss.','Mr.','Mrs.'),

train\_test$Name,'Other')

接下来对SibSp和Parch两个特征进行处理，因为这两个特征代表的分别是同在船上的兄弟姐妹/配偶数量和同在船上的父母/小孩数量，发现它们都和家庭有关，因此考虑将它们加起来合并成一个新特征——Familymembers，并且把原来的两个特征删掉。

train\_test=train\_test%>%

mutate(Familymembers=SibSp+Parch)%>%

select(-SibSp,-Parch)

把其中的类别特征转换成因子向量，方便模型训练。

train\_test=train\_test%>%

mutate(Name=as.factor(Name),

Sex=as.factor(Sex),

Embarked=as.factor(Embarked))

最后把一些多余的特征删除，PassengerId特征是每个乘客的编号，没有实际意义，但是需要利用该特征作为主键和Survived特征连接因此暂时保留，而Ticket特征是每张船票的信息，具体内容类似于A/5 21171、PC17599、373450等等，有些有字母有些是纯数字，不 仅杂乱无章并且不好处理，所以也选择删除。

train\_test=train\_test%>%

select(-Ticket)

以上就把全部特征都处理完了，现在来看一下数据的样子。

str(train\_test)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 1309 obs. of 9 variables:

## $ PassengerId : int 1 2 3 4 5 6 7 8 9 10 ...

## $ Pclass : int 3 1 3 1 3 3 1 3 3 2 ...

## $ Name : Factor w/ 5 levels "Master.","Miss.",..: 3 4 2 4 3 3 3 1 4 4 ...

## $ Sex : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...

## $ Age : num 22 38 26 35 35 ...

## $ Fare : num 7.25 71.28 7.92 53.1 8.05 ...

## $ Cabin : num 0 1 0 1 0 0 1 0 0 0 ...

## $ Embarked : Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...

## $ Familymembers: int 1 1 0 1 0 0 0 4 2 1 ...

**以下内容仅供对机器学习有兴趣的同学学习，不是必学内容**

# 建模和预测

本次分析的目标是预测乘客是存活还是未存活，因此属于二分类问题，二分类模型有很多，一般分为线性和非线性。线性模型解释性强但拟合效果不如非线性模型，非线性模型拟合效果好但是因此也容易出现过拟合，而且解释性也不如线性模型好，常见的二分类算法有逻辑回归、朴素贝叶斯分类器、支持向量机、K近邻、随机森林、xgboost等等。通常情况下，为了评估模型的预测性能，特别是训练好的模型在新数据上的泛化能力，需要对数据进行交叉验证，主流的交叉验证方法有留出法、k折交叉验证法，这里尝试采用k折交叉验证法。

## 模型训练

把Survived特征加入之前处理好的数据train\_test中的训练数据部分，并把PassengerId特征删掉。

train\_all=train%>%

select(PassengerId,Survived)%>%

right\_join(train\_test[1:891,],by='PassengerId')%>%

select(-PassengerId)

设置交叉验证的参数，折数为3，重复次数为5，超参数的设定为随机搜索，这样训练模型比较省时间，把分类概率输出并把设置保存到ctrl中。

ctrl= trainControl(method = "repeatedcv",number = 3,repeats=5,search="random",

summaryFunction = twoClassSummary,

classProbs = TRUE, savePredictions = "final")

把数据中的Survived特征改成字符型向量，否则会报错。

train\_all=train\_all%>%

mutate(Survived=ifelse(Survived==1,'Alive','Dead'))

把数据分别用逻辑回归、朴素贝叶斯分类器、支持向量机、K近邻、随机森林、xgboost来训练模型,把结果保存到model\_List中。

set.seed(1)

model\_list=caretList(

Survived~.,data=train\_all,

trControl=ctrl,

metric="ROC",

preProcess=c("center","scale"),

methodList=c("glm","svmRadialCost","knn","nb","rf","xgbTree")

)

把结果展示出来，然后比较每个模型的准确率，选择其中较好的几个进行模型融合，加强模型的泛化能力。尽管准确率高的模型在测试集表现不一定好，但是在某种程度上还是比较有效的。

set.seed(2)

result = resamples(model\_list)

summary(result)

##

## Call:

## summary.resamples(object = result)

##

## Models: glm, svmRadialCost, knn, nb, rf, xgbTree

## Number of resamples: 15

##

## ROC

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## glm 0.8262391 0.8556706 0.8714409 0.8697776 0.8856294 0.9044914

## svmRadialCost 0.8097977 0.8343160 0.8526028 0.8503483 0.8663719 0.8945211

## knn 0.8132969 0.8331656 0.8487681 0.8505896 0.8610632 0.8946889

## nb 0.8169159 0.8314040 0.8506615 0.8481993 0.8577557 0.8878823

## rf 0.8407152 0.8579714 0.8694996 0.8723133 0.8860847 0.9121848

## xgbTree 0.8193845 0.8481210 0.8690921 0.8640319 0.8783913 0.8918848

## NA's

## glm 0

## svmRadialCost 0

## knn 0

## nb 0

## rf 0

## xgbTree 0

##

## Sens

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## glm 0.6842105 0.7149123 0.7631579 0.7526316 0.7807018 0.8157895

## svmRadialCost 0.6842105 0.7017544 0.7543860 0.7421053 0.7675439 0.8333333

## knn 0.5789474 0.6184211 0.6491228 0.6520468 0.6798246 0.7456140

## nb 0.3771930 0.3991228 0.4210526 0.4339181 0.4605263 0.5263158

## rf 0.6403509 0.7017544 0.7192982 0.7216374 0.7456140 0.7894737

## xgbTree 0.6491228 0.7192982 0.7280702 0.7251462 0.7456140 0.7631579

## NA's

## glm 0

## svmRadialCost 0

## knn 0

## nb 0

## rf 0

## xgbTree 0

##

## Spec

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## glm 0.8469945 0.8633880 0.8743169 0.8728597 0.8852459 0.9016393

## svmRadialCost 0.8142077 0.8524590 0.8688525 0.8688525 0.8825137 0.9180328

## knn 0.8360656 0.8606557 0.8852459 0.8819672 0.9043716 0.9125683

## nb 0.9344262 0.9617486 0.9726776 0.9668488 0.9781421 0.9836066

## rf 0.8579235 0.9043716 0.9125683 0.9078324 0.9180328 0.9289617

## xgbTree 0.8251366 0.8579235 0.8907104 0.8801457 0.8989071 0.9234973

## NA's

## glm 0

## svmRadialCost 0

## knn 0

## nb 0

## rf 0

## xgbTree 0

可以看到这些模型的准确率都在0.84-0.88之间，准确率还是比较高的，表现最好的是随机森林（rf），超过了0.87，从结果中选出准确率排名前三的模型：rf，glm，xgbTree。

## 模型融合及预测

模型融合，也就是集成学习，常用的方法有三种：Bagging，Boosting，Stacking，模型融合有利于加强模型的泛化能力，下面用选出来的三个模型建立新的模型列表。

set.seed(3)

model\_list1=caretList(

Survived~.,data=train\_all,

trControl=ctrl,

metric="ROC",

preProcess=c("center","scale"),

methodList=c("glm","rf","xgbTree")

)

设定好参数然后对模型进行融合，并用得到的模型测试准确率。

crtl1=trainControl(

method="boot",

number=10,

savePredictions="final",

classProbs=TRUE,

summaryFunction=twoClassSummary

)

set.seed(4)

Ensemble <- caretStack(

model\_list1,

method="glm",

metric="ROC",

trControl=crtl1

)

Ensemble

## A glm ensemble of 2 base models: glm, rf, xgbTree

##

## Ensemble results:

## Generalized Linear Model

##

## 4455 samples

## 3 predictor

## 2 classes: 'Alive', 'Dead'

##

## No pre-processing

## Resampling: Bootstrapped (10 reps)

## Summary of sample sizes: 4455, 4455, 4455, 4455, 4455, 4455, ...

## Resampling results:

##

## ROC Sens Spec

## 0.8753216 0.734834 0.8979334

可以看到模型融合之后，准确率超过了0.87，提升效果还是很明显的，接下来用融合的模型对测试集进行预测，并把结果转换成0和1存入文件,在kaggle上提交结果。

newtest=train\_test[892:1309,]%>%select(-PassengerId)

predict=predict(Ensemble,newdata = newtest)

predict=ifelse(predict=='Alive',1,0)

sub=sub%>%mutate(Survived=predict)

write.csv(sub,'submitfile.csv',row.names = F)

最后得分只有0.7655，泛化能力还不够强，还需深入挖掘新特征！

# 总结与反思

1.本次数据分析的数据来源是已经处理好的整齐数据，因此没有缺少了整理数据的相关实践，比较遗憾。

2.在可视化的过程中，内容虽然较多但是深度还不够，没有对数据进行更深层的可视化挖掘。

3.缺失值填充较粗糙，没有分析缺失记录在其他特征的特点，显得不够严谨。

4.特征生成的比较少，因此也缺少对特征的筛选过程，应该对特征进行更加深入的挖掘。