Full Model Docs

Steph Locke

16 February 2017

We're working with data on who survived the Titanic.

## Source

We're collecting our data from a SQLite database. The titanic3 data was originally pulled in from the PASWR package and is the third major version of that dataset. It contains more features than the basic titanic dataset available in the datasets package.

library(DBI)  
library(RSQLite)  
titanicdb<-dbConnect(SQLite(),dbname="../data-raw/titanic.sqlite")

## Data

We're using just a single table of data that has already been collated. Here is a quick overview of the data.

titanic\_all<-dbReadTable(titanicdb, "titanic")  
knitr::kable(head(titanic\_all))

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| pclass | survived | name | sex | age | sibsp | parch | ticket | fare | cabin | embarked | boat | body | home.dest |
| 1st | 1 | Allen, Miss. Elisabeth Walton | female | 29.0000 | 0 | 0 | 24160 | 211.3375 | B5 | Southampton | 2 | NA | St Louis, MO |
| 1st | 1 | Allison, Master. Hudson Trevor | male | 0.9167 | 1 | 2 | 113781 | 151.5500 | C22 C26 | Southampton | 11 | NA | Montreal, PQ / Chesterville, ON |
| 1st | 0 | Allison, Miss. Helen Loraine | female | 2.0000 | 1 | 2 | 113781 | 151.5500 | C22 C26 | Southampton |  | NA | Montreal, PQ / Chesterville, ON |
| 1st | 0 | Allison, Mr. Hudson Joshua Crei | male | 30.0000 | 1 | 2 | 113781 | 151.5500 | C22 C26 | Southampton |  | 135 | Montreal, PQ / Chesterville, ON |
| 1st | 0 | Allison, Mrs. Hudson J C (Bessi | female | 25.0000 | 1 | 2 | 113781 | 151.5500 | C22 C26 | Southampton |  | NA | Montreal, PQ / Chesterville, ON |
| 1st | 1 | Anderson, Mr. Harry | male | 48.0000 | 0 | 0 | 19952 | 26.5500 | E12 | Southampton | 3 | NA | New York, NY |

knitr::kable(summary(titanic\_all))

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | pclass | survived | name | sex | age | sibsp | parch | ticket | fare | cabin | embarked | boat | body | home.dest |
|  | Length:1309 | Min. :0.000 | Length:1309 | Length:1309 | Min. : 0.1667 | Min. :0.0000 | Min. :0.000 | Length:1309 | Min. : 0.000 | Length:1309 | Length:1309 | Length:1309 | Min. : 1.0 | Length:1309 |
|  | Class :character | 1st Qu.:0.000 | Class :character | Class :character | 1st Qu.:21.0000 | 1st Qu.:0.0000 | 1st Qu.:0.000 | Class :character | 1st Qu.: 7.896 | Class :character | Class :character | Class :character | 1st Qu.: 72.0 | Class :character |
|  | Mode :character | Median :0.000 | Mode :character | Mode :character | Median :28.0000 | Median :0.0000 | Median :0.000 | Mode :character | Median : 14.454 | Mode :character | Mode :character | Mode :character | Median :155.0 | Mode :character |
|  | NA | Mean :0.382 | NA | NA | Mean :29.8811 | Mean :0.4989 | Mean :0.385 | NA | Mean : 33.295 | NA | NA | NA | Mean :160.8 | NA |
|  | NA | 3rd Qu.:1.000 | NA | NA | 3rd Qu.:39.0000 | 3rd Qu.:1.0000 | 3rd Qu.:0.000 | NA | 3rd Qu.: 31.275 | NA | NA | NA | 3rd Qu.:256.0 | NA |
|  | NA | Max. :1.000 | NA | NA | Max. :80.0000 | Max. :8.0000 | Max. :9.000 | NA | Max. :512.329 | NA | NA | NA | Max. :328.0 | NA |
|  | NA | NA | NA | NA | NA's :263 | NA | NA | NA | NA's :1 | NA | NA | NA | NA's :1188 | NA |

## Defensive stuff

### Store a copy (one-off)

cache\_file<-"../data-raw/rawdatacache.Rdata"  
if(!file.exists(cache\_file)) {  
 titanic\_cache<-titanic\_all  
 save(titanic\_cache,file = cache\_file)  
 rm(titanic\_cache)  
 }

### Check for changes

load(cache\_file)  
if(!identical(titanic\_all, titanic\_cache)) stop("Hey, the data has changed, you should check that out!")

We could change things and rerun with:

dbWriteTable(titanicdb, "titanic", head(titanic\_all), append=TRUE)

Restore your original data with your get\_data.R file.

We want to split our data into a training and a test sample for predicting survival of the Titanic. There are two sampling strategies we'll consider here.

## Random sample

n\_titanic\_all<-nrow(titanic\_all)  
prop\_train<-0.7  
n\_train<-floor(prop\_train \* n\_titanic\_all)  
  
# Sample RowIDs  
rows\_train\_s<-sample.int(n\_titanic\_all, n\_train)  
  
titanic\_train\_s<-titanic\_all[rows\_train\_s,]  
titanic\_test\_s<-titanic\_all[-rows\_train\_s,]

### Class Distribution

pval<-t.test(titanic\_train\_s$survived, titanic\_test\_s$survived, conf.level = .95)$p.value   
pval

## [1] 0.6089804

t.test(titanic\_train\_s$survived, titanic\_test\_s$survived, conf.level = .95)$p.value >= .05

## [1] TRUE

#### How often would they be sufficiently similar?

pass\_ttest<-function(x, prop\_train = .07){  
 n\_x\_all<-length(x)  
 n\_train<-floor(prop\_train \* n\_x\_all)  
   
 # Sample RowIDs  
 rows\_train<-sample.int(n\_x\_all, n\_train)  
   
 t.test(x[rows\_train], x[-rows\_train], conf.level = .95)$p.value >= .05  
}  
  
sum(replicate(1000,pass\_ttest(titanic\_all$survived)))/1000

## [1] 0.938

#### How do we get the same sample every time?

pass\_ttest<-function(x, prop\_train = .07){  
 set.seed(6767)  
 n\_x\_all<-length(x)  
 n\_train<-floor(prop\_train \* n\_x\_all)  
   
 # Sample RowIDs  
 rows\_train<-sample.int(n\_x\_all, n\_train)  
   
 t.test(x[rows\_train], x[-rows\_train], conf.level = .95)$p.value >= .05  
}  
  
sum(replicate(1000,pass\_ttest(titanic\_all$survived)))/1000

## [1] 1

## Maintaining class ratio

Using caret

library(caret)  
rows\_train\_c<-createDataPartition(titanic\_all$survived, p = prop\_train, list=FALSE)  
titanic\_train\_c<-titanic\_all[rows\_train\_c,]  
titanic\_test\_c<-titanic\_all[-rows\_train\_c,]

### Testing the class ratio

pass\_ttest<-function(x, prop\_train = .07){  
   
 # Sample RowIDs  
 rows\_train<-caret::createDataPartition(x, p=prop\_train, list=FALSE)  
   
 t.test(x[rows\_train], x[-rows\_train], conf.level = .95)$p.value >= .05  
}  
  
sum(replicate(1000,pass\_ttest(titanic\_all$survived)))/1000

## [1] 0.953

This gives us fewer cases where the class ratio is significantly different between training and sample datasets. Why is it not substantially much higher? Only so many permutations of 1309 rows for sampling - better results over more data.

## Reproducible sampling

We saw that we can set the seed within in a function to always yield the same value from our sampling process. We need to do this to select our training data so that the results don't shift each time we run the analysis.

## Defensive sampling

### Final sample

library(caret)  
set.seed(8787)  
rows\_train\_c<-createDataPartition(titanic\_all$survived, p = prop\_train, list=FALSE)  
titanic\_train\_c<-titanic\_all[rows\_train\_c,]  
titanic\_test\_c<-titanic\_all[-rows\_train\_c,]

### Storing a copy (one-off)

cache\_file<-"../data-processed/sampleIDs.Rdata"  
if(!file.exists(cache\_file)){  
 sample\_cache<- rows\_train\_c  
 save(sample\_cache,file=cache\_file)  
 rm(sample\_cache)  
}

### Check for changes

load(cache\_file)  
if(!identical(rows\_train\_c, sample\_cache)) stop("Hey, the sample has changed, you should check that out!")

Now that we have our data split into two sections, we can start processing features.

## Our outcome variable

Some stats packages don't like booleans, others don't like factor labels with spaces. Let's process our outcome variable and put it at the end of the data.frame, as is also expected.

convertsurvived<-function(x) {  
 factor(x,levels = c("0","1")  
 ,labels=c("Died","Survived")  
 )}  
   
survived\_tr <- convertsurvived(titanic\_train\_c$survived)  
titanic\_train\_c$survived<-NULL  
titanic\_train\_c$survived<-survived\_tr

## Factors

In some cases with strings, if you want to make these a categorical variable, you need to make sure to incorporate levels from both sets of data.

stringcols<- sapply(titanic\_train\_c, is.character)  
lapply(titanic\_train\_c[,stringcols],function(x) head(unique(x)))

## $pclass  
## [1] "1st" "2nd" "3rd"  
##   
## $name  
## [1] "Allison, Master. Hudson Trevor" "Allison, Mrs. Hudson J C (Bessi"  
## [3] "Anderson, Mr. Harry" "Andrews, Miss. Kornelia Theodos"  
## [5] "Andrews, Mr. Thomas Jr" "Appleton, Mrs. Edward Dale (Cha"  
##   
## $sex  
## [1] "male" "female"  
##   
## $ticket  
## [1] "113781" "19952" "13502" "112050" "11769" "PC 17609"  
##   
## $cabin  
## [1] "C22 C26" "E12" "D7" "A36" "C101" ""   
##   
## $embarked  
## [1] "Southampton" "Cherbourg" "Queenstown"   
##   
## $boat  
## [1] "11" "" "3" "10" "D" "4"   
##   
## $home.dest  
## [1] "Montreal, PQ / Chesterville, ON" "New York, NY"   
## [3] "Hudson, NY" "Belfast, NI"   
## [5] "Bayside, Queens, NY" "Montevideo, Uruguay"

convertclass<-function(x){ ordered(x , labels=c("3rd","2nd","1st"), levels=c("3rd","2nd","1st"))}  
  
titanic\_train\_c$pclass <- convertclass(titanic\_train\_c$pclass )  
  
prop.table(table(titanic\_train\_c$pclass, titanic\_train\_c$survived),margin = 1)

##   
## Died Survived  
## 3rd 0.7449799 0.2550201  
## 2nd 0.5550000 0.4450000  
## 1st 0.3926941 0.6073059

convertsex<-function(x) factor(x, labels=c("male","female"))  
titanic\_train\_c$sex <- convertsex(titanic\_train\_c$sex)  
prop.table(table(titanic\_train\_c$sex, titanic\_train\_c$survived),1)

##   
## Died Survived  
## male 0.2647975 0.7352025  
## female 0.8104027 0.1895973

convertcabin<-function(x){  
 f<-factor(substr(x,1,1),levels=c("",LETTERS[1:7],"T"))   
 levels(f)<-c("",LETTERS[1:6],"","")  
 return(f)}  
titanic\_train\_c$cabin<-convertcabin(titanic\_train\_c$cabin)  
prop.table(table(titanic\_train\_c$cabin, titanic\_train\_c$survived),1)

##   
## Died Survived  
## 0.6907074 0.3092926  
## A 0.5263158 0.4736842  
## B 0.3250000 0.6750000  
## C 0.4237288 0.5762712  
## D 0.3030303 0.6969697  
## E 0.2068966 0.7931034  
## F 0.3750000 0.6250000

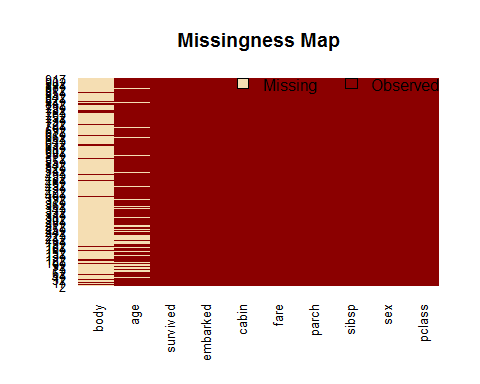
convertembarked<-function(x){factor(x,levels = c("Southampton","Cherbourg" , "Queenstown" ))}  
titanic\_train\_c$embarked<-convertembarked(titanic\_train\_c$embarked)  
prop.table(table(titanic\_train\_c$embarked, titanic\_train\_c$survived),1)

##   
## Died Survived  
## Southampton 0.6600000 0.3400000  
## Cherbourg 0.4594595 0.5405405  
## Queenstown 0.6585366 0.3414634

titanic\_train\_c$name <- NULL  
titanic\_train\_c$ticket <- NULL  
  
# Note this could be different if we wanted to geocode and then further categorise this variable  
titanic\_train\_c$home.dest<-NULL  
  
# Boat appears to have too many values to be useful  
titanic\_train\_c$boat<-NULL

## Missings

library(Amelia)  
missmap(titanic\_train\_c)



Looking at the data dictionary (?titanic3), the column body represents a number associated with a retrieved dead body. That would of course be correlated with not surviving so let's remove it!

titanic\_train\_c$body<-NULL

Age has a fairly high degree of missingness (19.8).

The sample size is small and removing these records would further decrease our data, an undesirable course of action.

The small sample size also means that imputing values by using a subset of data to predict missings would require using the same data, causing overfitting to creep in.

Utilising the average value, is also not a particularly good way to go.

Removing the column would substantially reduce our number of features also.

We should see if converting the converting the continuous variable to a binned variable, and the addition of an "NA" category will yield substantial information. This will lose precision on ages where provided but increases the amount of observations and features overall.

library(smbinning)  
alt\_titanic\_train\_c<-titanic\_train\_c[,c("survived","age")]  
alt\_titanic\_train\_c$survivedbool<-unclass(alt\_titanic\_train\_c$survived)-1  
binnings<-smbinning::smbinning(alt\_titanic\_train\_c,"survivedbool","age")  
binnings$ivtable

## Cutpoint CntRec CntGood CntBad CntCumRec CntCumGood CntCumBad PctRec  
## 1 <= 8 51 35 16 51 35 16 0.0556  
## 2 > 8 684 267 417 735 302 433 0.7459  
## 3 Missing 182 47 135 917 349 568 0.1985  
## 4 Total 917 349 568 NA NA NA 1.0000  
## GoodRate BadRate Odds LnOdds WoE IV  
## 1 0.6863 0.3137 2.1875 0.7828 1.2698 0.0916  
## 2 0.3904 0.6096 0.6403 -0.4458 0.0412 0.0013  
## 3 0.2582 0.7418 0.3481 -1.0551 -0.5681 0.0585  
## 4 0.3806 0.6194 0.6144 -0.4870 0.0000 0.1514

titanic\_train\_c$age<-smbinning.gen(alt\_titanic\_train\_c,binnings,"Age2")$Age2

## Processing numbers

We should scale numbers so that values are on similar scales to prevent individual variables overpowering the model.

procVar<-caret::preProcess(titanic\_train\_c)  
titanic\_train\_c<-predict(procVar, titanic\_train\_c)

## Checking for highly correlated variables

colsToRemoveDueToCorr<-findCorrelation(cor(sapply(titanic\_train\_c[,-ncol(titanic\_train\_c)],unclass)))  
if(length(colsToRemoveDueToCorr)>0)  
titanic\_train\_c<-titanic\_train\_c[,-colsToRemoveDueToCorr]

## Checking for low variance columns

colsToRemoveDueToLowVar<-nearZeroVar(titanic\_train\_c)  
if(length(colsToRemoveDueToLowVar)>0)  
titanic\_train\_c<-titanic\_train\_c[,-colsToRemoveDueToLowVar]

## Applying to test samples

Once processed for our training, we should apply changes to our test data. Consolidating transformations like this allows us to think about our production requirements in relation to the data.

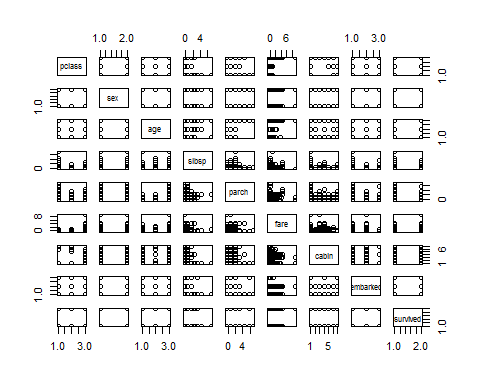
survived\_t <- convertsurvived(titanic\_test\_c$survived)  
titanic\_test\_c$survived<-NULL  
titanic\_test\_c$survived<-survived\_t  
  
titanic\_test\_c$pclass <- convertclass(titanic\_test\_c$pclass )  
titanic\_test\_c$sex <- convertsex(titanic\_test\_c$sex)  
titanic\_test\_c$cabin<-convertcabin(titanic\_test\_c$cabin)  
titanic\_test\_c$embarked<-convertembarked(titanic\_test\_c$embarked)  
titanic\_test\_c$name <- NULL  
titanic\_test\_c$home.dest <- NULL  
titanic\_test\_c$boat <- NULL  
titanic\_test\_c$ticket <- NULL  
titanic\_test\_c$body<-NULL  
  
alt\_titanic\_test\_c<-titanic\_test\_c[,c("survived","age")]  
alt\_titanic\_test\_c$survivedbool<-unclass(alt\_titanic\_test\_c$survived)-1  
titanic\_test\_c$age<-smbinning.gen(alt\_titanic\_test\_c,binnings,"Age2")$Age2  
  
titanic\_test\_c<-predict(procVar, titanic\_test\_c)  
  
if(length(colsToRemoveDueToCorr)>0)  
titanic\_test\_c<-titanic\_test\_c[,-colsToRemoveDueToCorr]  
if(length(colsToRemoveDueToLowVar)>0)  
titanic\_test\_c<-titanic\_test\_c[,-colsToRemoveDueToLowVar]

## Our data now

summary(titanic\_train\_c)

## pclass sex age sibsp parch   
## 3rd:498 male :321 00 Miss:182 Min. :-0.4582 Min. :-0.4755   
## 2nd:200 female:596 01 <= 8: 51 1st Qu.:-0.4582 1st Qu.:-0.4755   
## 1st:219 02 > 8 :684 Median :-0.4582 Median :-0.4755   
## Mean : 0.0000 Mean : 0.0000   
## 3rd Qu.: 0.5033 3rd Qu.:-0.4755   
## Max. : 7.2341 Max. : 6.9573   
##   
## fare cabin embarked survived   
## Min. :-0.61237 :721 Southampton:650 Died :568   
## 1st Qu.:-0.46023 A: 19 Cherbourg :185 Survived:349   
## Median :-0.34052 B: 40 Queenstown : 82   
## Mean : 0.00000 C: 59   
## 3rd Qu.:-0.04009 D: 33   
## Max. : 9.25961 E: 29   
## F: 16

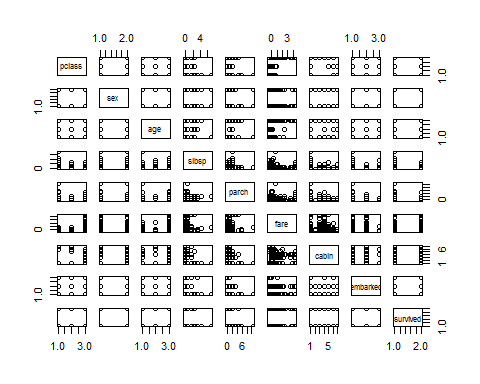
pairs(titanic\_train\_c)



summary(titanic\_test\_c)

## pclass sex age sibsp   
## 3rd:211 male :145 00 Miss: 81 Min. :-0.4582   
## 2nd: 77 female:247 01 <= 8: 21 1st Qu.:-0.4582   
## 1st:104 02 > 8 :290 Median :-0.4582   
## Mean : 0.0716   
## 3rd Qu.: 0.5033   
## Max. : 7.2341   
##   
## parch fare cabin embarked   
## Min. :-0.475529 Min. :-0.61237 :299 Southampton:264   
## 1st Qu.:-0.475529 1st Qu.:-0.45966 A: 3 Cherbourg : 85   
## Median :-0.475529 Median :-0.31370 B: 25 Queenstown : 41   
## Mean : 0.004825 Mean : 0.09766 C: 34 NA's : 2   
## 3rd Qu.:-0.475529 3rd Qu.: 0.15236 D: 14   
## Max. :10.673738 Max. : 4.45533 E: 12   
## NA's :1 F: 5   
## survived   
## Died :241   
## Survived:151   
##   
##   
##   
##   
##

pairs(titanic\_test\_c)



Our fully prepared data can now be used to generate models.

## All-in

allin<-caret::train(survived~., data=titanic\_train\_c,  
 method="glm")  
  
summary(allin)

##   
## Call:  
## NULL  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.2792 -0.6153 -0.4380 0.6544 2.5709   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.81234 0.28995 2.802 0.00508 \*\*   
## pclass.L 0.62747 0.28115 2.232 0.02563 \*   
## pclass.Q -0.17882 0.21645 -0.826 0.40872   
## sexfemale -2.74971 0.19968 -13.771 < 0.0000000000000002 \*\*\*  
## `age01 <= 8` 2.21237 0.47895 4.619 0.00000385 \*\*\*  
## `age02 > 8` 0.11392 0.25211 0.452 0.65135   
## sibsp -0.34156 0.12084 -2.827 0.00471 \*\*   
## parch -0.07831 0.09978 -0.785 0.43259   
## fare 0.20650 0.13412 1.540 0.12365   
## cabinA 0.32198 0.63369 0.508 0.61138   
## cabinB 0.77015 0.54835 1.405 0.16017   
## cabinC 0.31924 0.47901 0.666 0.50511   
## cabinD 0.81076 0.53176 1.525 0.12734   
## cabinE 1.88450 0.60746 3.102 0.00192 \*\*   
## cabinF 1.16439 0.66086 1.762 0.07808 .   
## embarkedCherbourg 0.56587 0.23082 2.452 0.01422 \*   
## embarkedQueenstown -0.05388 0.33670 -0.160 0.87286   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1218.42 on 916 degrees of freedom  
## Residual deviance: 813.82 on 900 degrees of freedom  
## AIC: 847.82  
##   
## Number of Fisher Scoring iterations: 5

caret::varImp(allin)

## glm variable importance  
##   
## Overall  
## sexfemale 100.000  
## `age01 <= 8` 32.763  
## cabinE 21.617  
## sibsp 19.592  
## embarkedCherbourg 16.836  
## pclass.L 15.221  
## cabinF 11.769  
## fare 10.136  
## cabinD 10.026  
## cabinB 9.144  
## pclass.Q 4.894  
## parch 4.590  
## cabinC 3.721  
## cabinA 2.557  
## `age02 > 8` 2.144  
## embarkedQueenstown 0.000

## Simple

simple<-caret::train(survived~sex+pclass, data=titanic\_train\_c,  
 method="glm")  
  
summary(simple)

##   
## Call:  
## NULL  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1415 -0.6939 -0.4690 0.6969 2.1269   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.30622 0.14119 9.251 <0.0000000000000002 \*\*\*  
## sexfemale -2.59252 0.17782 -14.580 <0.0000000000000002 \*\*\*  
## pclass.L 1.23464 0.14727 8.383 <0.0000000000000002 \*\*\*  
## pclass.Q 0.01809 0.16289 0.111 0.912   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1218.42 on 916 degrees of freedom  
## Residual deviance: 872.55 on 913 degrees of freedom  
## AIC: 880.55  
##   
## Number of Fisher Scoring iterations: 4

caret::varImp(simple)

## glm variable importance  
##   
## Overall  
## sexfemale 100.00  
## pclass.L 57.17  
## pclass.Q 0.00

## GLMNET

net<-caret::train(survived~., data=titanic\_train\_c,  
 method="glmnet")  
  
coef(net$finalModel,s=net$bestTune$lambda)

## 17 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 0.92543294  
## pclass.L 0.82850131  
## pclass.Q -0.02215632  
## sexfemale -2.58419554  
## age01 <= 8 1.78656267  
## age02 > 8 0.01353272  
## sibsp -0.26566590  
## parch -0.00682437  
## fare 0.14187680  
## cabinA .   
## cabinB 0.36521581  
## cabinC .   
## cabinD 0.43466212  
## cabinE 1.38194229  
## cabinF 0.89733862  
## embarkedCherbourg 0.48609230  
## embarkedQueenstown .

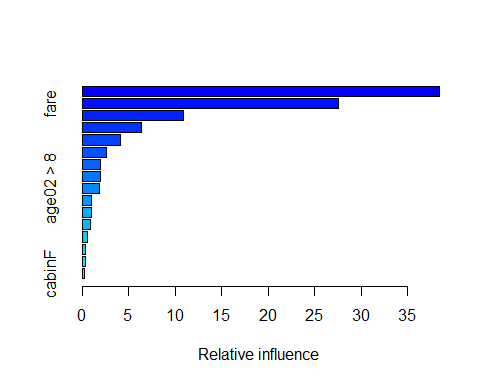
caret::varImp(net)

## glmnet variable importance  
##   
## Overall  
## sexfemale 100.0000  
## age01 <= 8 69.1342  
## cabinE 53.4767  
## cabinF 34.7241  
## pclass.L 32.0603  
## embarkedCherbourg 18.8102  
## cabinD 16.8200  
## cabinB 14.1327  
## sibsp 10.2804  
## fare 5.4902  
## pclass.Q 0.8574  
## age02 > 8 0.5237  
## parch 0.2641  
## cabinA 0.0000  
## embarkedQueenstown 0.0000  
## cabinC 0.0000

## gbm

boosted<-caret::train(survived~., data=titanic\_train\_c,  
 method="gbm"  
 )

summary(boosted)



## var rel.inf  
## sexfemale sexfemale 38.4587292  
## fare fare 27.5292796  
## pclass.L pclass.L 10.9334329  
## age01 <= 8 age01 <= 8 6.3382095  
## sibsp sibsp 4.0737810  
## embarkedCherbourg embarkedCherbourg 2.6522872  
## parch parch 1.9625352  
## cabinE cabinE 1.9150234  
## age02 > 8 age02 > 8 1.8837890  
## embarkedQueenstown embarkedQueenstown 1.0302212  
## cabinB cabinB 0.9770697  
## pclass.Q pclass.Q 0.8537863  
## cabinC cabinC 0.5173965  
## cabinA cabinA 0.3312845  
## cabinD cabinD 0.3007910  
## cabinF cabinF 0.2423838

caret::varImp(boosted)

## gbm variable importance  
##   
## Overall  
## sexfemale 100.0000  
## fare 71.4011  
## pclass.L 27.9751  
## age01 <= 8 15.9508  
## sibsp 10.0255  
## embarkedCherbourg 6.3059  
## parch 4.5011  
## cabinE 4.3768  
## age02 > 8 4.2950  
## embarkedQueenstown 2.0615  
## cabinB 1.9224  
## pclass.Q 1.5998  
## cabinC 0.7196  
## cabinA 0.2326  
## cabinD 0.1528  
## cabinF 0.0000

We can gather up all our predicted values.

trainres<-data.frame(  
 actual=titanic\_train\_c$survived,  
 allin=predict(allin,titanic\_train\_c),  
 simple=predict(simple,titanic\_train\_c),  
 net=predict(net,titanic\_train\_c),  
 boosted=predict(boosted,titanic\_train\_c),  
 allinval=predict(allin,titanic\_train\_c,type = "prob"),  
 simpleval=predict(simple,titanic\_train\_c,type = "prob"),  
 netval=predict(net,titanic\_train\_c,type = "prob"),  
 boostedval=predict(boosted,titanic\_train\_c,type = "prob")  
)  
  
titanic\_test\_c\_na<-na.omit(titanic\_test\_c)  
testres<-data.frame(  
 actual=titanic\_test\_c\_na$survived,  
 allin=predict(allin,titanic\_test\_c\_na),  
 simple=predict(simple,titanic\_test\_c\_na),  
 net=predict(net,titanic\_test\_c\_na),  
 boosted=predict(boosted,titanic\_test\_c\_na),  
 allinval=predict(allin,titanic\_test\_c\_na,type = "prob"),  
 simpleval=predict(simple,titanic\_test\_c\_na,type = "prob"),  
 netval=predict(net,titanic\_test\_c\_na,type = "prob"),  
 boostedval=predict(boosted,titanic\_test\_c\_na,type = "prob")  
)

## Confusion Matrices

### allin

confusionMatrix(trainres$actual, trainres$allin)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 481 87  
## Survived 91 258  
##   
## Accuracy : 0.8059   
## 95% CI : (0.7788, 0.831)   
## No Information Rate : 0.6238   
## P-Value [Acc > NIR] : <0.0000000000000002  
##   
## Kappa : 0.5874   
## Mcnemar's Test P-Value : 0.8221   
##   
## Sensitivity : 0.8409   
## Specificity : 0.7478   
## Pos Pred Value : 0.8468   
## Neg Pred Value : 0.7393   
## Prevalence : 0.6238   
## Detection Rate : 0.5245   
## Detection Prevalence : 0.6194   
## Balanced Accuracy : 0.7944   
##   
## 'Positive' Class : Died   
##

confusionMatrix(testres$actual, testres$allin)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 200 40  
## Survived 43 106  
##   
## Accuracy : 0.7866   
## 95% CI : (0.7425, 0.8263)   
## No Information Rate : 0.6247   
## P-Value [Acc > NIR] : 0.000000000004688  
##   
## Kappa : 0.5468   
## Mcnemar's Test P-Value : 0.8262   
##   
## Sensitivity : 0.8230   
## Specificity : 0.7260   
## Pos Pred Value : 0.8333   
## Neg Pred Value : 0.7114   
## Prevalence : 0.6247   
## Detection Rate : 0.5141   
## Detection Prevalence : 0.6170   
## Balanced Accuracy : 0.7745   
##   
## 'Positive' Class : Died   
##

### simple

confusionMatrix(trainres$actual, trainres$simple)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 483 85  
## Survived 113 236  
##   
## Accuracy : 0.7841   
## 95% CI : (0.756, 0.8103)   
## No Information Rate : 0.6499   
## P-Value [Acc > NIR] : < 0.0000000000000002  
##   
## Kappa : 0.5348   
## Mcnemar's Test P-Value : 0.05501   
##   
## Sensitivity : 0.8104   
## Specificity : 0.7352   
## Pos Pred Value : 0.8504   
## Neg Pred Value : 0.6762   
## Prevalence : 0.6499   
## Detection Rate : 0.5267   
## Detection Prevalence : 0.6194   
## Balanced Accuracy : 0.7728   
##   
## 'Positive' Class : Died   
##

confusionMatrix(testres$actual, testres$simple)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 198 42  
## Survived 48 101  
##   
## Accuracy : 0.7686   
## 95% CI : (0.7235, 0.8096)  
## No Information Rate : 0.6324   
## P-Value [Acc > NIR] : 0.000000005642   
##   
## Kappa : 0.5067   
## Mcnemar's Test P-Value : 0.5982   
##   
## Sensitivity : 0.8049   
## Specificity : 0.7063   
## Pos Pred Value : 0.8250   
## Neg Pred Value : 0.6779   
## Prevalence : 0.6324   
## Detection Rate : 0.5090   
## Detection Prevalence : 0.6170   
## Balanced Accuracy : 0.7556   
##   
## 'Positive' Class : Died   
##

### glmnet

confusionMatrix(trainres$actual, trainres$net)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 483 85  
## Survived 94 255  
##   
## Accuracy : 0.8048   
## 95% CI : (0.7776, 0.83)   
## No Information Rate : 0.6292   
## P-Value [Acc > NIR] : <0.0000000000000002  
##   
## Kappa : 0.5839   
## Mcnemar's Test P-Value : 0.5499   
##   
## Sensitivity : 0.8371   
## Specificity : 0.7500   
## Pos Pred Value : 0.8504   
## Neg Pred Value : 0.7307   
## Prevalence : 0.6292   
## Detection Rate : 0.5267   
## Detection Prevalence : 0.6194   
## Balanced Accuracy : 0.7935   
##   
## 'Positive' Class : Died   
##

confusionMatrix(testres$actual, testres$net)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 197 43  
## Survived 45 104  
##   
## Accuracy : 0.7738   
## 95% CI : (0.7289, 0.8144)  
## No Information Rate : 0.6221   
## P-Value [Acc > NIR] : 0.0000000001127   
##   
## Kappa : 0.5201   
## Mcnemar's Test P-Value : 0.9151   
##   
## Sensitivity : 0.8140   
## Specificity : 0.7075   
## Pos Pred Value : 0.8208   
## Neg Pred Value : 0.6980   
## Prevalence : 0.6221   
## Detection Rate : 0.5064   
## Detection Prevalence : 0.6170   
## Balanced Accuracy : 0.7608   
##   
## 'Positive' Class : Died   
##

### gbm

confusionMatrix(trainres$actual, trainres$boosted)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 516 52  
## Survived 89 260  
##   
## Accuracy : 0.8462   
## 95% CI : (0.8212, 0.869)   
## No Information Rate : 0.6598   
## P-Value [Acc > NIR] : < 0.00000000000000022  
##   
## Kappa : 0.6671   
## Mcnemar's Test P-Value : 0.002431   
##   
## Sensitivity : 0.8529   
## Specificity : 0.8333   
## Pos Pred Value : 0.9085   
## Neg Pred Value : 0.7450   
## Prevalence : 0.6598   
## Detection Rate : 0.5627   
## Detection Prevalence : 0.6194   
## Balanced Accuracy : 0.8431   
##   
## 'Positive' Class : Died   
##

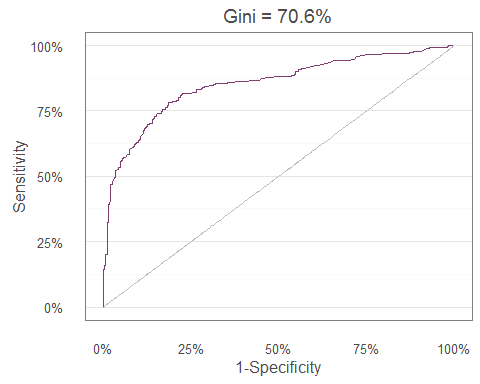
confusionMatrix(testres$actual, testres$boosted)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Died Survived  
## Died 215 25  
## Survived 45 104  
##   
## Accuracy : 0.8201   
## 95% CI : (0.7782, 0.857)   
## No Information Rate : 0.6684   
## P-Value [Acc > NIR] : 0.00000000001662  
##   
## Kappa : 0.6093   
## Mcnemar's Test P-Value : 0.02315   
##   
## Sensitivity : 0.8269   
## Specificity : 0.8062   
## Pos Pred Value : 0.8958   
## Neg Pred Value : 0.6980   
## Prevalence : 0.6684   
## Detection Rate : 0.5527   
## Detection Prevalence : 0.6170   
## Balanced Accuracy : 0.8166   
##   
## 'Positive' Class : Died   
##

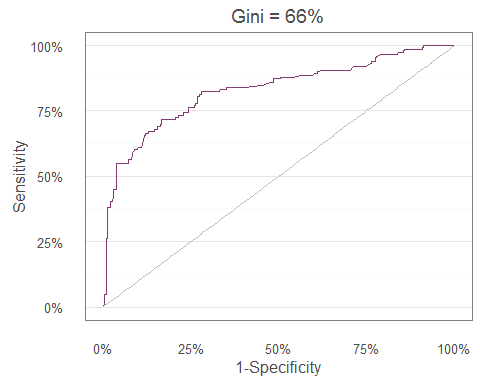
## Gini and AUC

### allin

library(optiRum)  
 giniChart(trainres$allinval.Survived,trainres$actual)

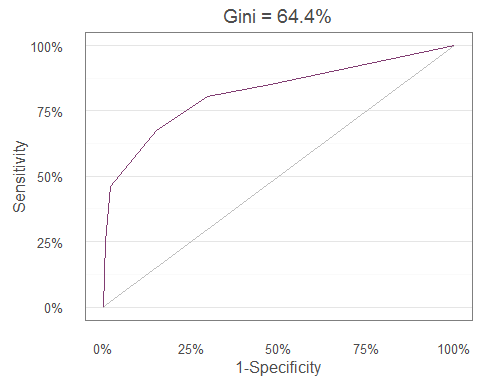


giniChart(testres$allinval.Survived,testres$actual)

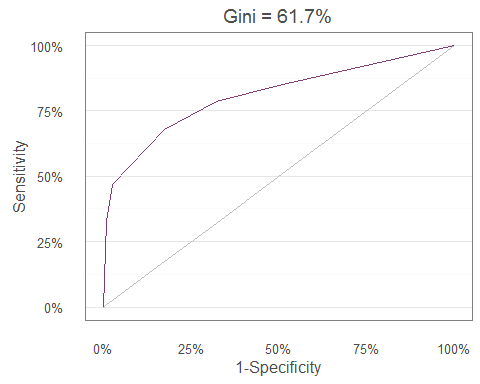


### simple

giniChart(trainres$simpleval.Survived,trainres$actual)

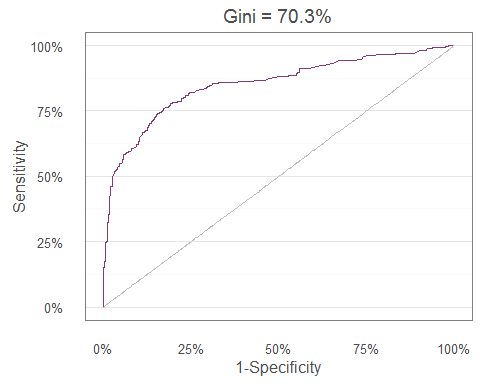


giniChart(testres$simpleval.Survived,testres$actual)

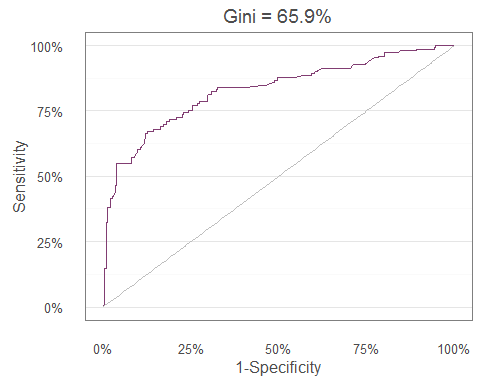


### glmnet

giniChart(trainres$netval.Survived,trainres$actual)

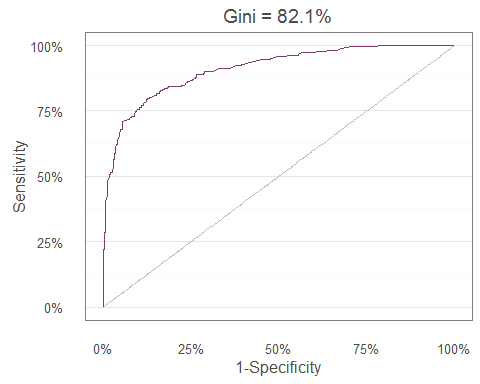


giniChart(testres$netval.Survived,testres$actual)

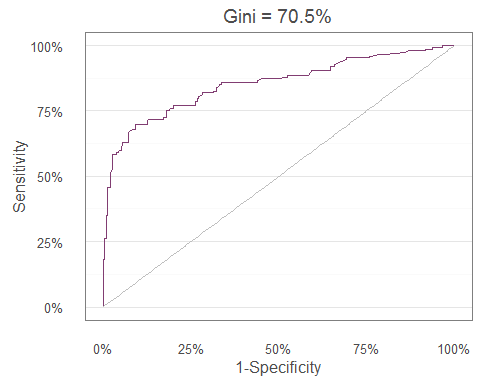


### gbm

giniChart(trainres$boostedval.Survived,trainres$actual)



giniChart(testres$boostedval.Survived,testres$actual)



## Selecting a model

We can discard the simple model - it doesn't give sufficient discrimination compared to the other models.

The full GLM and the regularized version, look like they have similar levels of discrimination. We could do a tuning exercise on the regularized model to see if we could improve it. Given time constraints, we will use the full GLM.

The gradient boosted model appears to be performing best.

## Future work

We could further tune models and remove unimportant variables.

In other situations, we could compare against another out of time sample.