The RMS Titanic Drama

Philippe J.S. De Brouwer

03 January, 2022

Abstract

We use the data of the passenger on the RMS Titanic build a model that predicts survival.

Contents

This analysis will be run with R and we will use the following libraries:

```
# Libraries:
library(tidyverse)  # tibble, readr, etc.
```

```
library(forcats)    # working with factors in the tidyverse
library(gridExtra)  # put multiple ggplot graphs alongside in one box
library(missForest) # impute missing values with estimates
library(InformationValue) # WOE and IV
library(ROCR)       # AUC
library(rpart)      # fit the decision tree
library(knitr)      # include beautiful tables
```

Chapter 1

Introduction

This work-flow is inspired by: (De Brouwer 2020) and is prepared by Philippe De Brouwer to illustrate the concepts explained in the book.

Chapter 2

The Titanic

RMS Titanic was a British passenger liner, operated by the White Star Line, which sank in the North Atlantic Ocean on 15 April 1912 after striking an iceberg during her maiden voyage from Southampton, UK, to New York City. Of the estimated 2,224 passengers and crew aboard, more than 1,500 died, which made the sinking possibly one of the deadliest for a single ship up to that time. It remains to this day the deadliest peacetime sinking of a superliner or cruise ship The disaster drew much public attention, provided foundational material for the disaster film genre and has inspired many artistic works.

RMS Titanic was the largest ship afloat at the time she entered service and the second of three Olympic-class ocean liners operated by the White Star Line. She was built by the Harland and Wolff shipyard in Belfast. Thomas Andrews, who was the chief naval architect of the shipyard at that time, died in the disaster.

Titanic was under the command of Captain Edward Smith, who went down with the ship. The ocean liner carried some of the wealthiest people in the world, as well as hundreds of emigrants from Great Britain and Ireland, Scandinavia and elsewhere throughout Europe, who were seeking a new life in the United States. The first-class accommodation was designed to be the pinnacle of comfort and luxury, with a gymnasium, swimming pool, libraries, high-class restaurants, and opulent cabins. A high-powered radiotelegraph transmitter was available for sending passenger "marconigrams" and for the ship's operational use. The Titanic had advanced safety features, such as watertight compartments and remotely activated watertight doors. The ship was equipped with 16 lifeboat davits, each of which were capable of lowering three lifeboats, for a total of 48 boats. And yet the Titanic carried only 20 lifeboats, four of which were collapsible and proved hard to launch while the ship was sinking. Together, the 20 lifeboats were capable of holding 1,178 people—which was only about half the number of passengers on board, and only one-third of the number of passengers that the ship could have carried at full capacity. (This was consistent with the

maritime safety regulations in those days.) In addition, when the ship sank, the lifeboats that had been lowered were only about half full.

Titanic had departed from Southampton on 10 April 1912, then stopped at Cherbourg, France, and Queenstown (now Cobh), Ireland, before heading west towards New York.[8] On 14 April, four days into the crossing and about 375 miles (600 km) south of Newfoundland, she hit an iceberg at 11:40 p.m. ship's time. The collision caused the hull plates to buckle inwards along her starboard (right) side and laid five of her sixteen watertight compartments open to the sea; she had been designed to survive the flooding of up to four compartments. Some passengers and crew members were evacuated in lifeboats, many of which were launched only partially loaded. A disproportionate number of men were left aboard because of a "women and children first" protocol for loading lifeboats.[9] At 2:20 am, the ship broke apart and foundered, with well over one thousand people still aboard. Just under two hours after Titanic sank, the Cunard liner RMS Carpathia arrived on the scene, and took on board an estimated 710 survivors.

The disaster was met with worldwide shock and outrage, both at the huge loss of life, and at the regulatory and procedural failures that had led to it. Public inquiries in Britain and the United States led to major improvements in maritime safety. One of the most important results of the inquiries was the establishment in 1914 of the International Convention for the Safety of Life at Sea (SOLAS), which still governs maritime safety today. In addition, there was an effort to learn from the many missteps in wireless communications that had increased the number of fatalities, and as a result several new wireless regulations were put in place around the world.

Chapter 3

The Data

```
# Data input:
fl <- "http://www.de-brouwer.com/assets/tmp/titanic.csv"
d0 <- read_csv(fl)
```

The database of passengers of the Titanic has data about 1309 passengers, and has the following columns pclass, survived, name, sex, age, sibsp, parch, ticket, fare, cabin, embarked, boat, body, home.dest.

```
summary(d0)
```

```
##      pclass          survived         name                sex
##  Min.   :1.000   Min.   :0.000   Length:1309        Length:1309
##  1st Qu.:2.000   1st Qu.:0.000   Class :character   Class :character
##  Median :3.000   Median :0.000   Mode  :character   Mode  :character
##  Mean   :2.295   Mean   :0.382
##  3rd Qu.:3.000   3rd Qu.:1.000
##  Max.   :3.000   Max.   :1.000
##
##       age              sibsp            parch          ticket
##  Min.   : 0.1667   Min.   :0.0000   Min.   :0.000   Length:1309
##  1st Qu.:21.0000   1st Qu.:0.0000   1st Qu.:0.000   Class :character
##  Median :28.0000   Median :0.0000   Median :0.000   Mode  :character
##  Mean   :29.8811   Mean   :0.4989   Mean   :0.385
##  3rd Qu.:39.0000   3rd Qu.:1.0000   3rd Qu.:0.000
##  Max.   :80.0000   Max.   :8.0000   Max.   :9.000
##  NA's   :263
##       fare            cabin             embarked            boat
##  Min.   :  0.000   Length:1309        Length:1309        Length:1309
##  1st Qu.:  7.896   Class :character   Class :character   Class :character
##  Median : 14.454   Mode  :character   Mode  :character   Mode  :character
```

```
##  Mean   : 33.295
##  3rd Qu.: 31.275
##  Max.   :512.329
##  NA's   :1
##       body        home.dest
##  Min.   :  1.0   Length:1309
##  1st Qu.: 72.0   Class :character
##  Median :155.0   Mode  :character
##  Mean   :160.8
##  3rd Qu.:256.0
##  Max.   :328.0
##  NA's   :1188
```

In this example we will predict the survival chances of a passenger in function of the other variables. Since this prediction should be based on past data we cannot use variables that are only relevant after the facts. This concerns the following columns:

- **body**: the number of the body recovered, only given when the dead body was recovered
- **boat**: the number of the lifeboat on which the passenger survived

Further we notice that the

- name of each person will be individual and not helpful as such,
- the column `ticket` has 929 different values with 0 missing values and no appearant pattern (shuch as class A or B tickets for example),
- home destination (the column `home.dest`) has 370 different destinations with 564 missing values, and
- cabin number (the column `cabin`) has 187 different destinations with 1014 missing values.

```
library(naniar)    # visualization of missing values

naniar::gg_miss_upset(d0)
```

This is illustrated in Figure 3.1 at page 7.

Therefore, we will remove those columns before moving forward.

```
d1 <- d0 %>% select(-c(name, body, boat, home.dest, cabin, ticket))
```

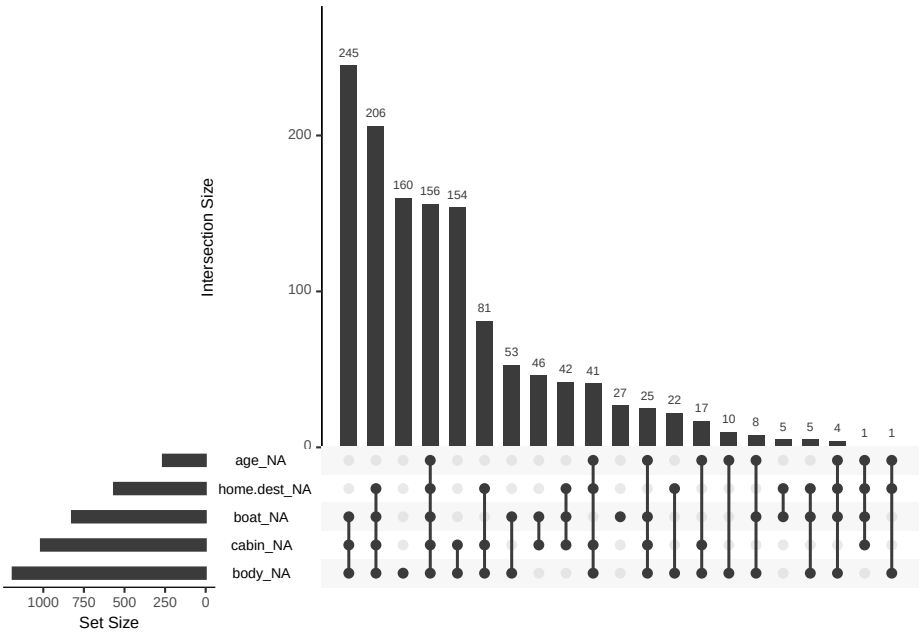


Figure 3.1: The intersections of missing values in the Titanic datasets visualised per class.

Chapter 4

Data Wrangling

4.1 Normalising data

Some algorithms for models – such as `glm()` to fit a logistic regression will create categorical variables when necessary (e.g. for the variable `embarked`). However, in some cases – such as for the variable `pclass` they might be treated as numerical while it is actually a categorical variable. Therefore we decide to prepare a normalized, numerical data-set.

```
#define the MinMax normalization function
min_max_norm <- function(x) {
    (x - min(x, na.rm = T)) / (max(x, na.rm = T) - min(x, na.rm = T))
  }

d1_n <- tibble(
  survived = d1$survived,
  class1 = if_else(d1$pclass == 1, 1, 0),
  class2 = if_else(d1$pclass == 2, 1, 0),
  class3 = if_else(d1$pclass == 3, 1, 0),
  male   = if_else(d1$sex == "male", 1, 0),
  female = if_else(d1$sex == "female", 1, 0),
  age    = min_max_norm(d1$age),
  sibl   = min_max_norm(d1$sibsp),
  par    = min_max_norm(d1$parch),
  fare   = min_max_norm(d1$fare),
  embC   = if_else(d1$embarked == "C", 1, 0),
  embS   = if_else(d1$embarked == "S", 1, 0),
  embQ   = if_else(d1$embarked == "Q", 1, 0)
)
```

```
#also remember min and max for later use:
age_r  <- range(d1$age,   na.rm = TRUE)
sib_r  <- range(d1$sibsp, na.rm = TRUE)
par_r  <- range(d1$parch, na.rm = TRUE)
far_r  <- range(d1$fare,  na.rm = TRUE)
```

We also consider a data-frame that has no obviously correlated columns:

```
d2_n <- d1_n %>% select(-c(class3, female, embS))
```

4.2 Missing values

Figure 4.1 at page 9 visualizes the missing values in the data-set for the retained columns. We notice that the missing values mainly appear in the variable `age` and are correlated with the lower class.

```
naniar::gg_miss_fct(x = d1, fct = pclass)
naniar::gg_miss_upset(d2_n)
#grid.arrange(arrangeGrob(p1, p2, ncol = 2))
```

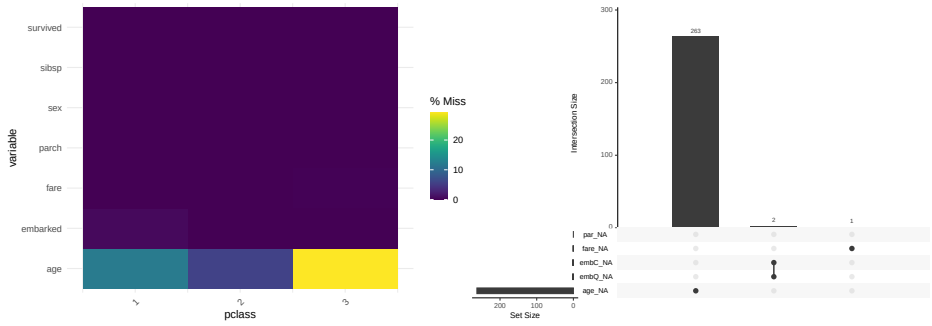


Figure 4.1: The missing values in the Titanic datasets visualised per class (left) and and missing patterns (right). Most missing values are in age.

We leave out the rows that have missing values on embarked and fare.

```
d3 <- d2_n %>% filter(!is.na(embC) & !is.na(fare))
```

The number of missing values in the variable age is 263. So, 20.14% of the age is missing, and hence we will try to fit in some values rather than leaving out one fifth of the data.

Figure 4.1 at page 9 indicates also that the data is not missing at random (MAR)[1]. Therefore we will use a random forest imputing method supplied by the `missForest` library.

[1] Missing values correlate strongly to the class in which passengers traveled and hence are not missing at random

```
# Input missing values via random forest estimates:
# ---
library(missForest)# input missing values

# Note: this fails:
# d1_imp <- missForest(d0, )$ximp
# While everything that worked for a data-frame will also work
# for a tibble, this is an exception, so we have 2 solutions:

# 1. coerce to data-frame
d_imp <- missForest(as.data.frame(d3), )$ximp
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
```

```
# 2. Manual fit the randomforest and impute the missing data:
x_cc <- d3 %>% filter(!is.na(d3$age))
x_nc <- d3 %>% filter(is.na(d3$age))
rf <- randomForest::randomForest(age ~ ., data = x_cc)
x_nc$age <- predict(rf, x_nc)
x <- rbind(x_cc, x_nc)
```

4.3 The modelling data-set

We decided to retain the imputation of missing values before the cross-validation, as suggested by (Jaeger, Tierney, and Simon 2020).

We retain the imputed data from last step.

```
d <- d_imp
```

We prepare also a data-set with the multi-level factorial variables and unscaled numerical variables. This data set will be of use for certain models such as the decision tree and will make the results more clear.

```
# Factorial levels dataset:
d_fact <- tibble(
  survived = d$survived,
  class    = factor(if_else(d$class1 == 1, 1, if_else(d$class2 == 1, 2, 3))),
  sex      = factor(if_else(d$male == 1, "male", "female")),
  age      = round(d$age  * age_r[2] + age_r[1], 1),
  sibl     = round(d$sibl * sib_r[2] + sib_r[1], 0),
  par      = round(d$par  * par_r[2] + par_r[1], 0),
  fare     = round(d$fare * far_r[2] + far_r[1], 4),
  embarked = factor(if_else(d$embC == 1, "C", if_else(d$embQ == 1, "Q", "S")))
)
```

4.4 Data binning

4.4.1 Categorical variables

```
# Information Value
library (InformationValue)
```

```
## Categorical variables
WOE_tbl <- data.frame(varName = character(0), IV = numeric(0))
for (col in colnames(d_fact)) {
    if (!(col == 'survived') & (is.factor(d_fact[[col]]))) {
     wt <- WOETable(d_fact[[col]], d$survived, valueOfGood = 1)
     tmp  <- data.frame(varName = col, IV = sum(wt$IV))
     WOE_tbl <- rbind(WOE_tbl, tmp)
     }
 }

 # show the best IVs:
 knitr::kable(WOE_tbl[order(WOE_tbl$IV, decreasing = TRUE),],
              caption = 'The table of all information values
              for each categorical variable ordered in
              decreasing order. We will work with the ones
              that have an information value above $0.1$.')
```

Table 4.1: The table of all information values for each categorical variable ordered in decreasing order. We will work with the ones that have an information value above 0.1.

	varName	IV
2	sex	1.2562481
1	class	0.4131207
3	embarked	0.1391277

```
for (col in colnames(d_fact)) {
    if (!(col == 'survived') & (is.factor(d_fact[[col]]))) {
     WOETable(X = d_fact[[col]], Y = d_fact$survived) %>%
        knitr::kable() %>% print()
       cat("\n")
    }
 }
```

```
##
##
## |CAT | GOODS| BADS| TOTAL|     PCT_G|     PCT_B|        WOE|        IV|
## |:---|-----:|----:|-----:|---------:|---------:|----------:|---------:|
```

```
## |1   |   198|  123|   321| 0.3975904| 0.1522277|  0.9600447| 0.2355591|
## |2   |   119|  158|   277| 0.2389558| 0.1955446|  0.2004904| 0.0087035|
## |3   |   181|  527|   708| 0.3634538| 0.6522277| -0.5847415| 0.1688581|
##
##
##
## |CAT    | GOODS| BADS| TOTAL|     PCT_G|     PCT_B|       WOE|        IV|
## |:------|-----:|----:|-----:|---------:|---------:|---------:|---------:|
## |female |   337|  127|   464| 0.6767068| 0.1571782|  1.459858| 0.7584379|
## |male   |   161|  681|   842| 0.3232932| 0.8428218| -0.958196| 0.4978102|
##
##
##
## |CAT | GOODS| BADS| TOTAL|     PCT_G|     PCT_B|        WOE|        IV|
## |:---|-----:|----:|-----:|---------:|---------:|----------:|---------:|
## |C   |   150|  120|   270| 0.3012048| 0.1485149|  0.7071055| 0.1079679|
## |Q   |    44|   79|   123| 0.0883534| 0.0977723| -0.1012962| 0.0009541|
## |S   |   304|  609|   913| 0.6104418| 0.7537129| -0.2108286| 0.0302056|
```

So, we shouldn't really use the port of embarking, but all the other categories seem to be fine as such.

Still try matrix variable of sex and class:

```
WOETable(X = factor(paste0(d_fact$class, d_fact$sex)),
         Y = d_fact$survived) %>%
       knitr::kable()
```

CAT	GOODS	BADS	TOTAL	PCT_G	PCT_B	WOE	IV
1female	137	5	142	0.2751004	0.0061881	3.7945050	1.0203890
1male	61	118	179	0.1224900	0.1460396	-0.1758488	0.0041412
2female	94	12	106	0.1887550	0.0148515	2.5423501	0.4421237
2male	25	146	171	0.0502008	0.1806931	-1.2807688	0.1671304
3female	106	110	216	0.2128514	0.1361386	0.4469207	0.0342845
3male	75	417	492	0.1506024	0.5160891	-1.2316361	0.4501466

4.4.2 Continous variables

```
# Continuous variables

## Visualizing the dependency
my_plot <- function(colname, title) {
  qplot(data = d_fact, get(colname), survived, xlab = "",
        ylab = "", alpha = 0.001) +
     geom_smooth(method = "loess", size = 1.5, span = 0.85) +
```

```
     ggtitle(title)  +  ylim(c(0,1)) +
     theme_bw() + theme(legend.position = "none")
}
p1 <- my_plot("age", "age")
p2 <- my_plot("sibl", "number of siblings")
p3 <- my_plot("par", "number of parents or children")
p4 <- my_plot("fare", "fare")
grid.arrange(p1, p2, p3, p4, ncol=2)
```

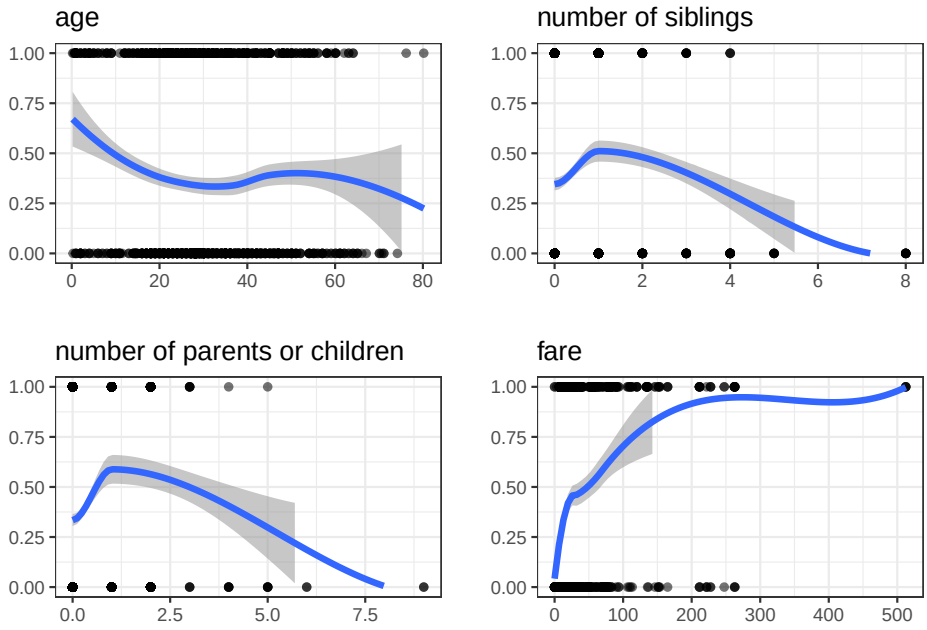
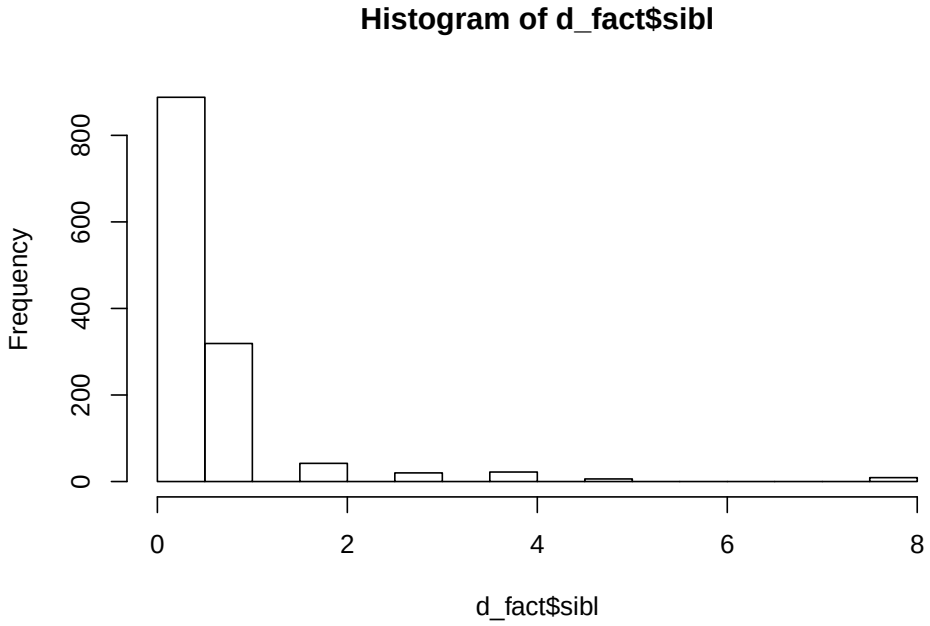


Figure 4.2: The dependency structure for the continous variables.

This suggest the following cuts:

```
## Bin age
hist(d_fact$sibl)
```




```
age_c <- cut(d_fact$age, breaks = c(0, 20, 30, 40, 100))
table(age_c)
```

```
## age_c
##   (0,20]  (20,30]  (30,40] (40,100]
##      270      477      293      266
```

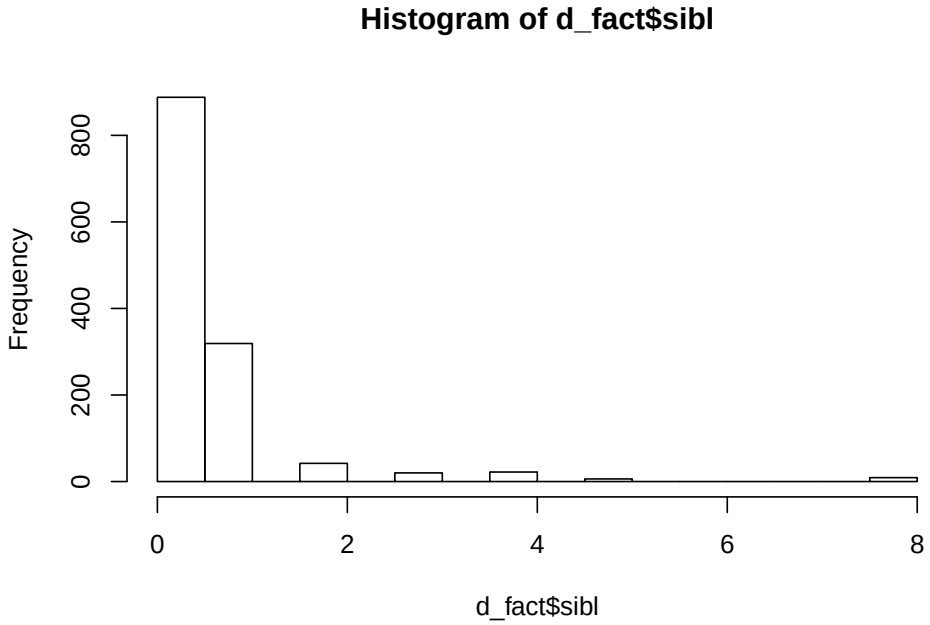
```
WOETable(X = age_c, Y = d_fact$survived)
```

```
##        CAT GOODS BADS TOTAL     PCT_G     PCT_B         WOE           IV
## 1   (0,20]   120  150   270 0.2409639 0.1856436  0.26081843 1.442855e-02
## 2  (20,30]   165  312   477 0.3313253 0.3861386 -0.15309573 8.391684e-03
## 3  (30,40]   113  180   293 0.2269076 0.2227723  0.01839295 7.606134e-05
## 4 (40,100]   100  166   266 0.2008032 0.2054455 -0.02285562 1.061034e-04
```

```
IV_age <- sum(WOETable(X = age_c, Y = d_fact$survived)$IV)
```

The IV of age is only 0.02 and hence we will not use it.

```
# Bin Sibl
hist(d_fact$sibl)
```




```
sibl_c <- cut(d_fact$sibl, breaks = c(0, 1, 10),
              include.lowest = TRUE, right = FALSE)
table(sibl_c)
```

```
## sibl_c
##  [0,1) [1,10]
##    888    418
```

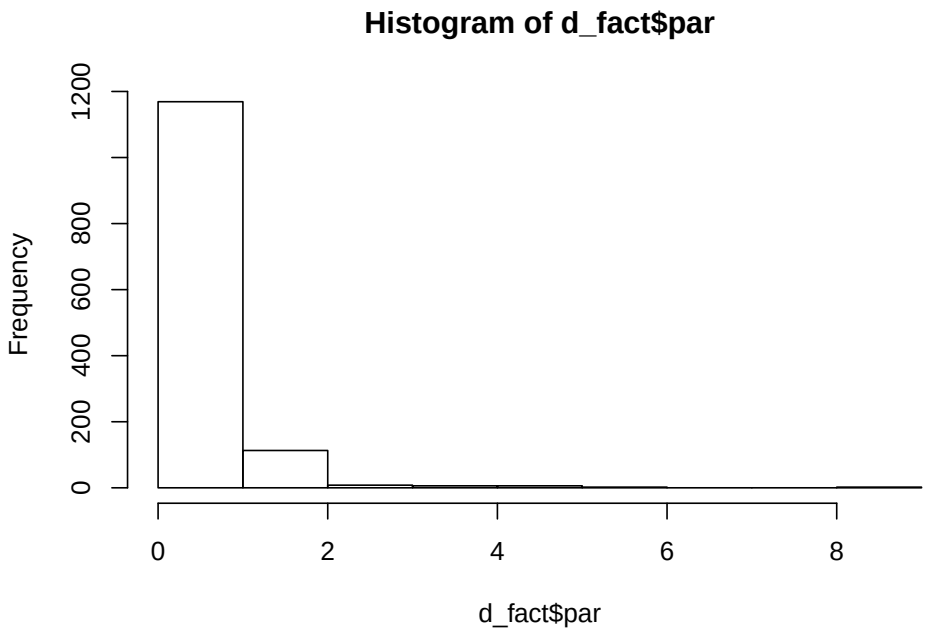
```
WOETable(X = sibl_c, Y = d_fact$survived)
```

```
##      CAT GOODS BADS TOTAL     PCT_G     PCT_B        WOE         IV
## 1  [0,1)   307  581   888 0.6164659 0.7190594 -0.1539410 0.01579336
## 2 [1,10]   191  227   418 0.3835341 0.2809406  0.3112854 0.03193587
```

```
IV_sib <- sum(WOETable(X = sibl_c, Y = d_fact$survived)$IV)
```

The IV of siblings is only 0.05 and hence we will not use it.

```
# Bin par
hist(d_fact$par)
```




```
par_c <- cut(d_fact$par, breaks = c(0, 1, 15),
             include.lowest = TRUE, right = FALSE)
table(par_c)
```

```
## par_c
##  [0,1) [1,15]
##    999    307
```

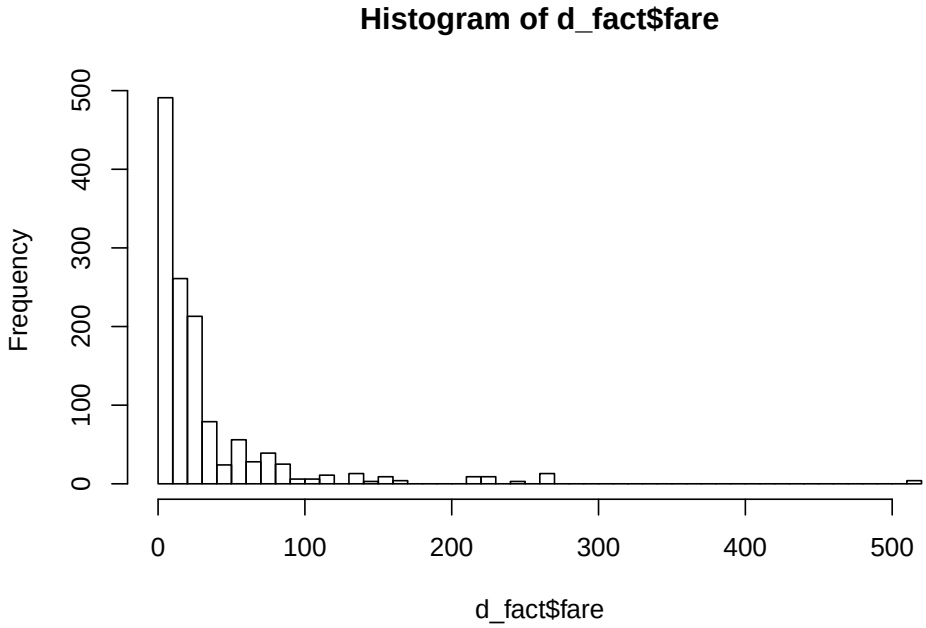
```
WOETable(X = par_c, Y = d_fact$survived)
```

```
##      CAT GOODS BADS TOTAL     PCT_G     PCT_B        WOE         IV
## 1  [0,1)   334  665   999 0.6706827 0.8230198 -0.2046841 0.03118097
## 2 [1,15]   164  143   307 0.3293173 0.1769802  0.6209838 0.09459885
```

```
IV_par <- sum(WOETable(X = par_c, Y = d_fact$survived)$IV)
```

The IV of par is 0.13 and hence we will use it.

```
# Bin fare
hist(d_fact$fare, breaks = 50)
```




```
fare_c <- cut(d_fact$fare, breaks = c(0, 10, 50, 100, 550), right = FALSE)
table(fare_c)
```

```
## fare_c
##    [0,10)   [10,50)  [50,100) [100,550)
##       491       575       156        84
```

```
WOETable(X = fare_c, Y = d_fact$survived)
```

```
##          CAT GOODS BADS TOTAL     PCT_G      PCT_B          WOE          IV
## 1    [0,10)   110  381   491 0.2208835 0.47153465 -0.75835703 0.190083038
## 2   [10,50)   232  343   575 0.4658635 0.42450495  0.09296891 0.003845055
## 3  [50,100)    96   60   156 0.1927711 0.07425743  0.95396561 0.113057955
## 4 [100,550)    60   24    84 0.1204819 0.02970297  1.40025271 0.127113481
```

```
IV_fare <- sum(WOETable(X = fare_c, Y = d_fact$survived)$IV)
```

The IV of siblings is 0.43 and hence it is a good parameter to use.

4.4.3 The binned data

```
d_bin <- tibble(
             survived = d$survived,
             class1 = d$class1,
             class2 = d$class2,
             male   = d$male,
             par0   = if_else(d$par == 0, 1, 0), # zero parent/children
```

```
                fare_010_050 = if_else(fare_c == '[10,50)',   1, 0 ),
                fare_050_100 = if_else(fare_c == '[50,100)',  1, 0 ),
                fare_100_550 = if_else(fare_c == '[100,550)', 1, 0 ),
                embC    = d$embC,
                embQ    = d$embQ
)
```

4.5 Correlation

```
# Use the library GGaly to explore the correlation structure
# ---
library(GGally)

d1$pclass <- factor(d1$pclass)
d1$survived <- factor(d1$survived)
GGally::ggpairs(d1, ggplot2::aes(colour=sex))
```

Figure 4.3: The correlation structure of the data

```
GGally::ggcorr(d_bin,  method = c("everything", "pearson"),
               label = TRUE, label_round = 2, label_size = 3)
```

we notice that being in first class correlates with paying a higher fare and being older. Other correlations are lower.

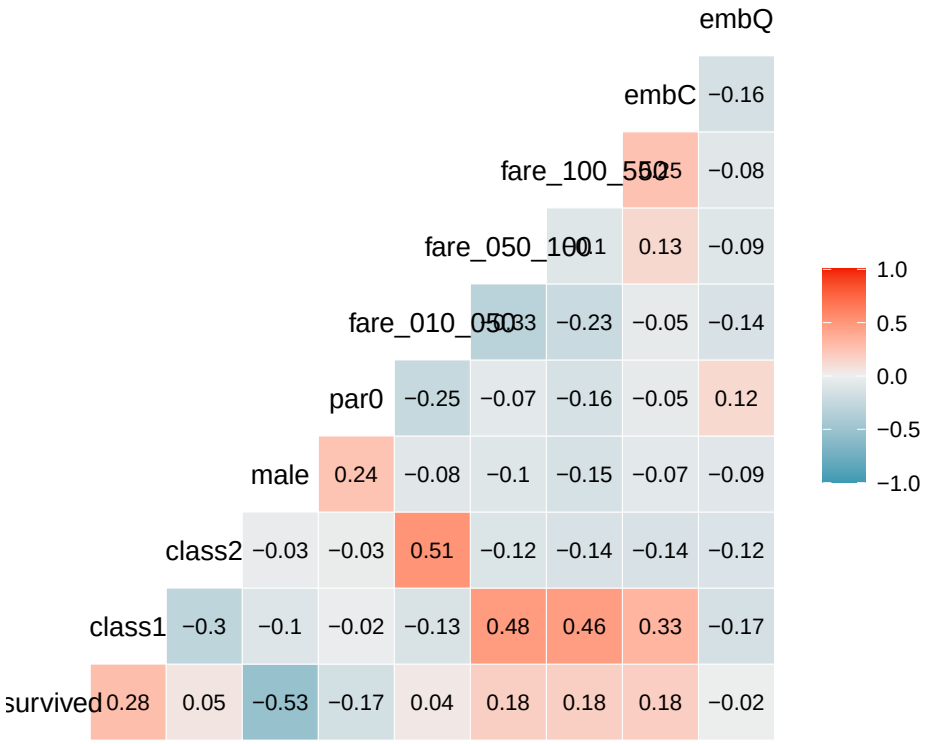


Figure 4.4: The correlations of the normalised data

Chapter 5

Models

5.1 Cross Validation

To ensure model quality we set 20% of the data aside for cross valiation.

```
library(modelr)
set.seed(18901229)
d_bin_rs  <- resample_partition(d_bin,  c(train = 0.85, test = 0.15))
d_fact_rs <- resample_partition(d_fact, c(train = 0.85, test = 0.15))

# create the datasets to be used for the modelling
d_bin_train <- as.tibble(d_bin_rs$train)
d_fact_train <- as.tibble(d_fact_rs$train)

d_bin_test <- as.tibble(d_bin_rs$test)
d_fact_test <- as.tibble(d_fact_rs$test)
```

5.2 The base model

```
# Fitting the Logistic Regression:
frm_logreg1 = survived ~ class1 * male + class2 * male + par0 + embC + embQ
m_logreg1 <- glm(formula = frm_logreg1, data = d_bin_train)
summary(m_logreg1)
```

```
##
## Call:
## glm(formula = frm_logreg1, data = d_bin_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -0.9888  -0.2032  -0.1160   0.1064   0.8840
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.51825    0.03494  14.834  < 2e-16 ***
## class1       0.43955    0.04816   9.126  < 2e-16 ***
## male        -0.33810    0.03518  -9.611  < 2e-16 ***
## class2       0.38335    0.05084   7.541 9.73e-14 ***
## par0        -0.06417    0.02844  -2.256  0.02424 *
## embC         0.09517    0.03051   3.119  0.00186 **
## embQ         0.04761    0.04270   1.115  0.26517
## class1:male -0.27515    0.05899  -4.664 3.47e-06 ***
## male:class2 -0.36032    0.06242  -5.772 1.02e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.1488883)
##
##     Null deviance: 260.09  on 1109  degrees of freedom
## Residual deviance: 163.93  on 1101  degrees of freedom
## AIC: 1046.9
##
## Number of Fisher Scoring iterations: 2
```

We notice that according to this model

- first class passengers have a higher survival chance than class 2, who in their turn have a better survival chance than class 3
- surviving chances are lower for males
- in the male group, especially men in class2 do worse, followed by class1
- having siblings is a bad thing in general, but males who have siblings do better than their peers

5.2.1 The AUC

```
library(ROCR)

# function to get AUC
fAUC <- function(model, data, ...) {
  y     <- all.vars(formula(model))[1]
  pred1 <- predict(model, newdata = data, ...)
  pred  <- ROCR::prediction(pred1, data[[y]])
  perf  <- performance(pred, "auc")
  AUC   <- attr(perf, "y.values")[[1]]
  AUC
}
```

```
AUC_logreg1_train <- fAUC(m_logreg1, data = d_bin_train, type = "response")
AUC_logreg1_test  <- fAUC(m_logreg1, data = d_bin_test, type = "response")
```

The AUC on the training data is 0.83 and on the testing data 0.85. The difference is only -0.02. This model is a good contender.

5.3 The Challenger models

5.3.1 Decision tree

```
# Fitting the Decision Tree:
library(rpart)

t1 <- rpart(survived ~ ., d_fact_train,
                  method = "class",
                  control = rpart.control(minsplit = 50,
                                          minbucket = 20,
                                          cp = 0.0001))

plotcp(t1)
```

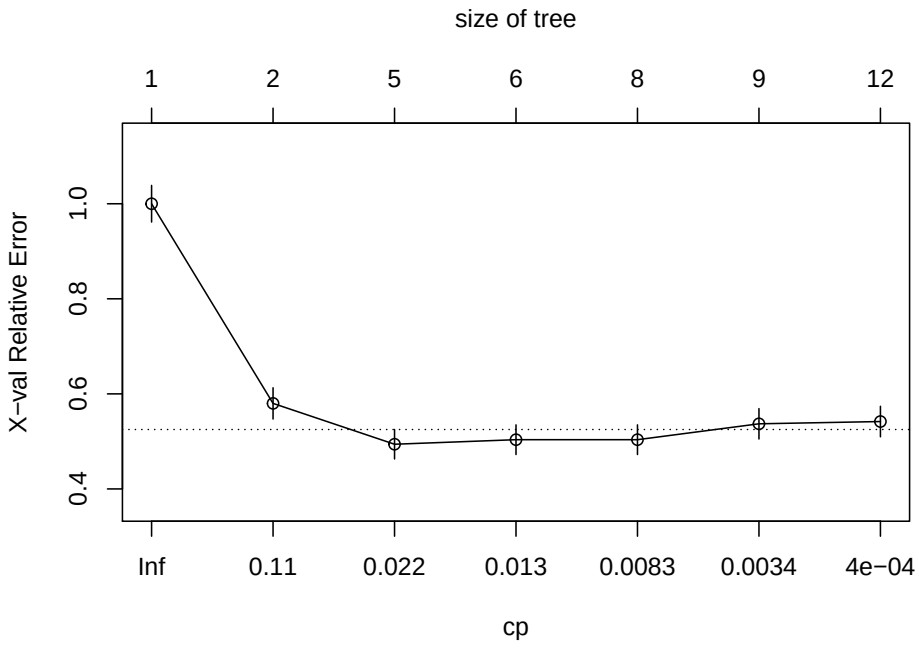


Figure 5.1: The decision tree for the titanic data.

```
t1 <- prune(t1, cp = 0.022)
library(rpart.plot)
rpart.plot::prp(t1, type = 5, extra = 8, box.palette = "auto",
    yesno = 1, yes.text="survived",no.text="dead"
    )
```

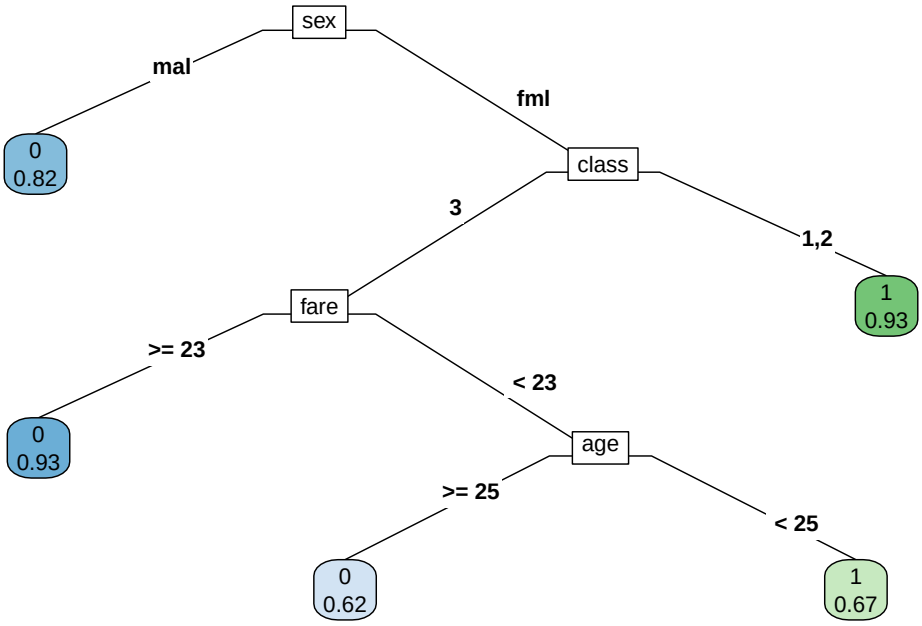


Figure 5.2: The decision tree for the titanic data.

```
AUC_t1_train <- fAUC(t1, data = d_fact_train, type = "vector")
AUC_t1_test  <- fAUC(t1, data = d_fact_test, type = "vector")
```

Chapter 6

Cross Validation

6.1 For the logistic regression

```
pctTrain <- 0.7
set.seed(18901229)
nRuns = 200

#fvAUC <- Vectorize(fAUC)
d <- d_bin_train

cv_mc <- crossv_mc(d, n = nRuns, test = 1 - pctTrain)
mods  <- map(cv_mc$train, ~ glm(frm_logreg1, data = ., family = "binomial"))

#AUCs  <- map2_dbl(mods, cv_mc$test, fAUC)
RMSE  <- map2_dbl(mods, cv_mc$test, rmse)
AUCs_train <- numeric(0)
AUCs_test  <- numeric(0)
for(k in 1:nRuns) {
  AUCs_test[k]  <- fAUC(mods[[k]], as.data.frame(cv_mc$test[[k]]))
  AUCs_train[k] <- fAUC(mods[[k]], as.data.frame(cv_mc$train[[k]]))
  }

allAUCs <- rbind(tibble(model = "train data", AUC = AUCs_train),
                 tibble(model = "test data", AUC = AUCs_test))
p1 <- ggplot(allAUCs, aes(AUC, fill = model, colour = model)) + geom_density(alpha=0.5)
p2 <- ggplot(allAUCs, aes(AUC, fill = model, colour = model)) + stat_ecdf()
  grid.arrange(p1, p2, ncol = 1)
```

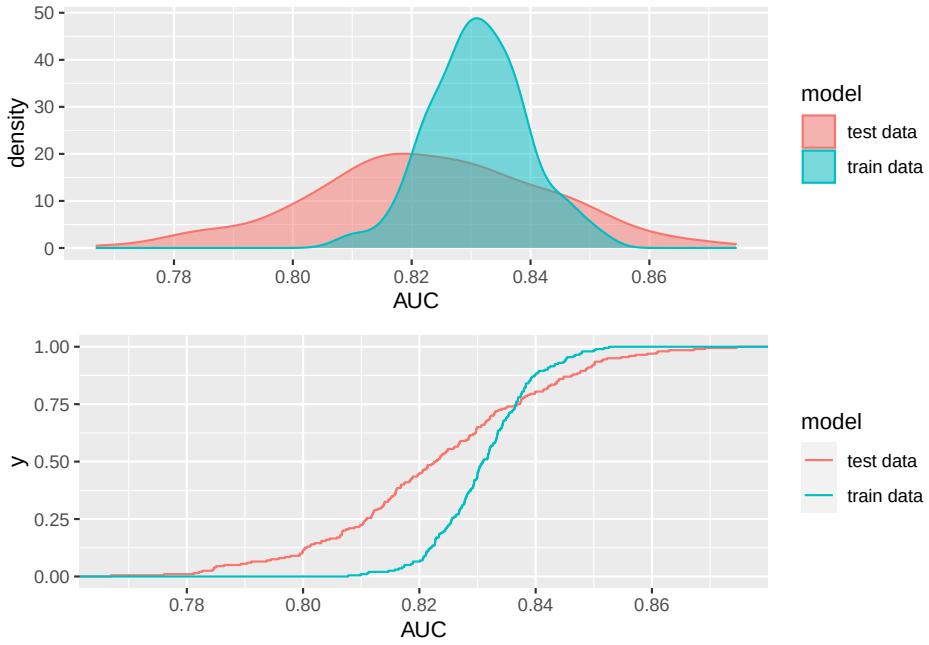


Figure 6.1: The cross validation for the logistic regression.

6.2 For the decision tree

```
pctTrain <- 0.7
set.seed(18901229)
nRuns = 200

#fvAUC <- Vectorize(fAUC)
d <- d_bin_train

cv_mc <- crossv_mc(d, n = nRuns, test = 1 - pctTrain)
# Notet that the following is an over-simplification -- instead we should
# build the trees to a small cp and then prune. This can be solved by building a function th
mods  <- map(cv_mc$train, ~ rpart(survived ~ . , data = ., cp = 0.002))

#AUCs  <- map2_dbl(mods, cv_mc$test, fAUC)
RMSE  <- map2_dbl(mods, cv_mc$test, rmse)
AUCs_train <- numeric(0)
AUCs_test  <- numeric(0)
for(k in 1:nRuns) {
  AUCs_test[k]  <- fAUC(mods[[k]], as.data.frame(cv_mc$test[[k]]))
  AUCs_train[k] <- fAUC(mods[[k]], as.data.frame(cv_mc$train[[k]]))
  }
```

```
allAUCs <- rbind(tibble(model = "train data", AUC = AUCs_train),
                 tibble(model = "test data", AUC = AUCs_test))
p1 <- ggplot(allAUCs, aes(AUC, fill = model, colour = model)) + geom_density(alpha=0.5)
p2 <- ggplot(allAUCs, aes(AUC, fill = model, colour = model)) + stat_ecdf()
  grid.arrange(p1, p2, ncol = 1)
```

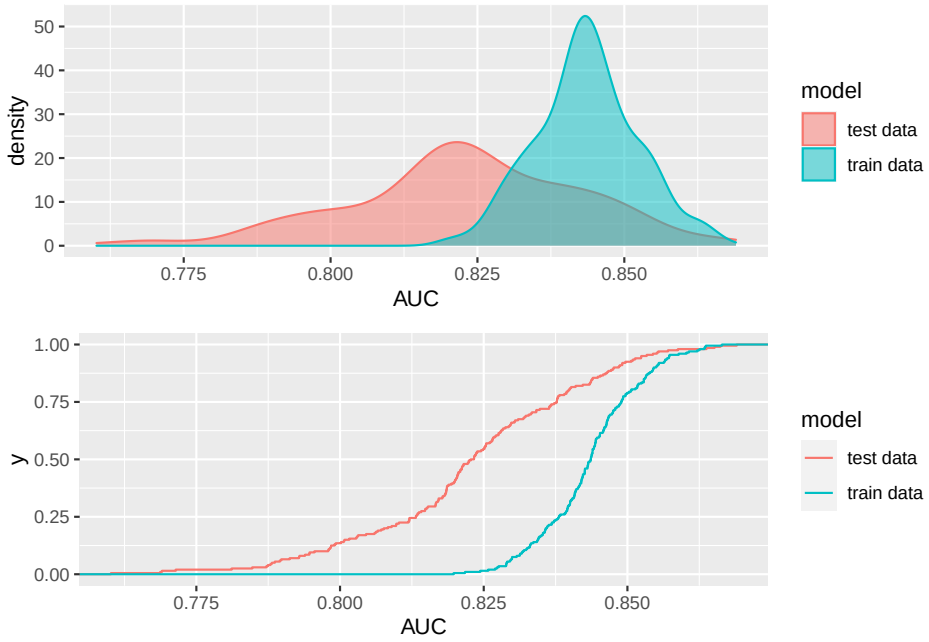


Figure 6.2: The cross validation for the decision tree.

Chapter 7

Conclusions

References

De Brouwer, Philippe J. S. 2020. *The Big r-Book: From Data Science to Learning Machines and Big Data*. New York: John Wiley & Sons, Ltd.

Jaeger, Byron C, Nicholas J Tierney, and Noah R Simon. 2020. "When to Impute? Imputation Before and During Cross-Validation." *arXiv Preprint arXiv:2010.00718*.