Chapter 2 Data Sets

We illustrate the techniques presented in this book by using three datasets:

- *Sinking of the RMS Titanic*
- *Apartment prices*
- *Hire or Fire*

The first dataset will be used to illustrate the application of the techniques in the case of a predictive model for a binary dependent variable. The second one will provide an example for models for a continuous variable. Finally, the third dataset will be used for illustration of models for a categorical dependent variable.

In this chapter, we provide a short description of each of the datasets, together with results of exploratory analyses. We also introduce models that will be used for illustration purposes in subsequent chapters.

2.1 Sinking of the RMS Titanic

Titanic sinking by Willy Stöwer

Sinking of the RMS Titanic is one of the deadliest maritime disasters in history (during peacetime). Over 1500 people died as a consequence of collision with an iceberg. Projects like *Encyclopedia titanica* `https://www.encyclopedia-titanica.org/` are a source of rich and precise data about Titanic's passengers. The data are available in a dataset included in the `stablelearner` package. The dataset, including some variable transformations, is also avaliable in the `DALEX` package. In particular, the `titanic' data frame contains 2207 observations (for 1317 passengers and 890 crew members) and nine variables:

- *gender*, person's (passenger's or crew member's) gender, a factor (categorical variable) with two levels (categories)
- *age*, person's age in years, a numerical variable; for adults, the age is given in (integer) years; for children younger than one year, the age is given as $x/12$, where x is the number of months of child's age
- *class*, the class in which the passenger travelled, or the duty class of a crew member; a factor with seven levels
- *embarked*, the harbor in which the person embarked on the ship, a factor with four levels
- *country*, person's home country, a factor with 48 levels
- *fare*, the price of the ticket (only available for passengers; 0 for crew members), a numerical variable
- *sibsp*, the number of siblings/spouses aboard the ship, a numerical variable
- *parch*, the number of parents/children aboard the ship, a numerical variable
- *survived*, a factor with two levels indicating whether the person survived or not

Models considered for this dataset will use *survived* as the (binary) dependent variable.

```
library("DALEX")
head(titanic, 2)
```

```
##   gender age class    embarked       country  fare sibsp parch survived
## 1   male  42   3rd Southampton United States  7.11     0     0       no
## 2   male  13   3rd Southampton United States 20.05     0     2       no
```

```
str(titanic)
```

```
## 'data.frame':    2207 obs. of  9 variables:
##  $ gender  : Factor w/ 2 levels "female","male": 2 2 2 1 1 2 2 1 2 2 ...
##  $ age     : num  42 13 16 39 16 25 30 28 27 20 ...
##  $ class   : Factor w/ 7 levels "1st","2nd","3rd",..: 3 3 3 3 3 3 2 2 3 3 ...
##  $ embarked: Factor w/ 4 levels "Belfast","Cherbourg",..: 4 4 4 4 4 4 2 2 2 4 ...
##  $ country : Factor w/ 48 levels "Argentina","Australia",..: 44 44 44 15 30 44 17 17
##  $ fare    : num  7.11 20.05 20.05 20.05 7.13 ...
##  $ sibsp   : num  0 0 1 1 0 0 1 1 0 0 ...
##  $ parch   : num  0 2 1 1 0 0 0 0 0 0 ...
##  $ survived: Factor w/ 2 levels "no","yes": 1 1 1 2 2 2 1 2 2 2 ...
```

```
levels(titanic$class)
```

```
## [1] "1st"              "2nd"              "3rd"
## [4] "deck crew"        "engineering crew" "restaurant staff"
## [7] "victualling crew"
```

```
levels(titanic$embarked)
```

```
## [1] "Belfast"     "Cherbourg"   "Queenstown"  "Southampton"
```

2.1.1 Data exploration

It is always advisable to explore data before modelling. However, as this book is focused on model exploration, we will limit the data exploration part.

Before exploring the data, we first do some pre-processing. In particular, the value of variables *age*, *country*, *sibsp*, *parch*, and *fare* is missing for a limited number of observations (2, 81, 10, 10, and 26, respectively). Analyzing data with missing values is a topic on its own (Little and Rubin 1987; Schafer 1997; Molenberghs and Kenward 2007). An often-used approach is to impute the missing values. Toward this end, multiple imputation should be considered (Schafer 1997; Molenberghs and Kenward 2007; van Buuren 2012). However, given the limited number of missing values and the intended illustrative use of the dataset, we will limit ourselves to, admittedly inferior, single imputation. In particular, we replace the missing *age*

values by the mean of the observed ones, i.e., 30. Missing *country* will be coded by “X”. For *sibsp* and *parch*, we replace the missing values by the most frequently observed value, i.e., 0. Finally, for *fare*, we use the value of 0.

[TOMASZ: FOR FARE, ONE COULD USE THE MEAN OF OBSERVED VALUES, AS FOR AGE. TAKING 0 CORRESPONDS TO “crew”.]

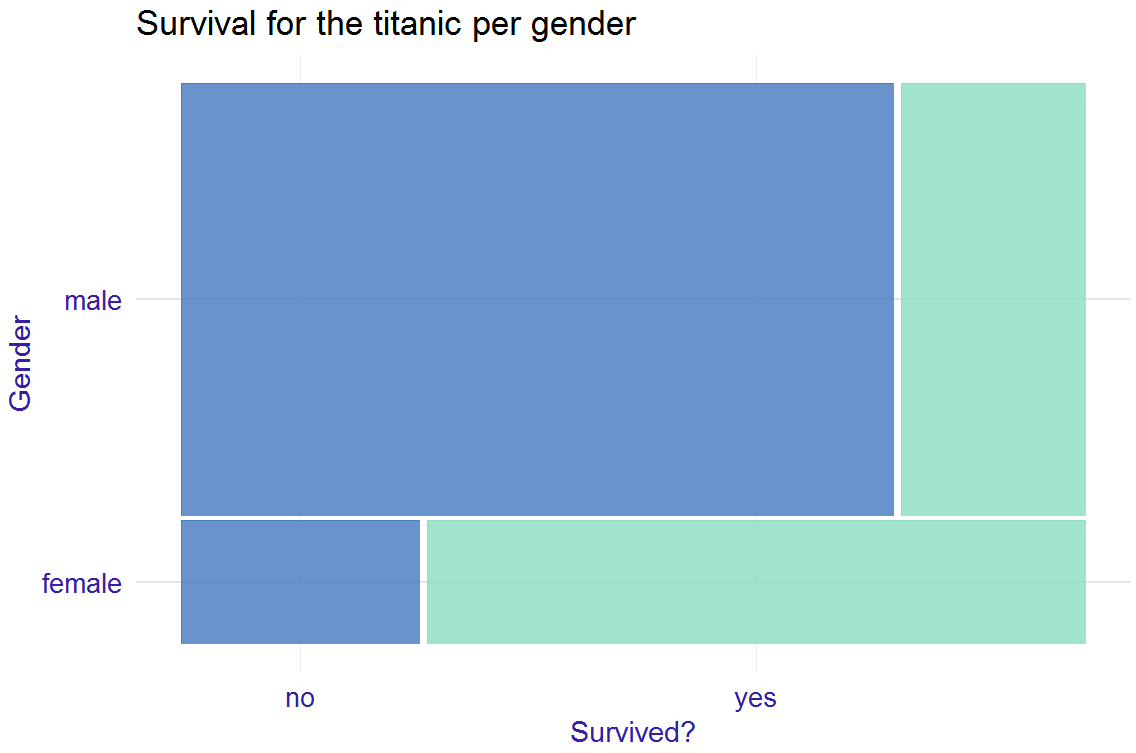
```
# missing country is replaced by "X"
titanic$country[is.na(titanic$country)] = "X"
# missing age is replaced by average (30)
titanic$age[is.na(titanic$age)] = 30
# missing fare, sibsp, parch are replaced by 0
titanic$fare[is.na(titanic$fare)] = 0
titanic$sibsp[is.na(titanic$sibsp)] = 0
titanic$parch[is.na(titanic$parch)] = 0
```

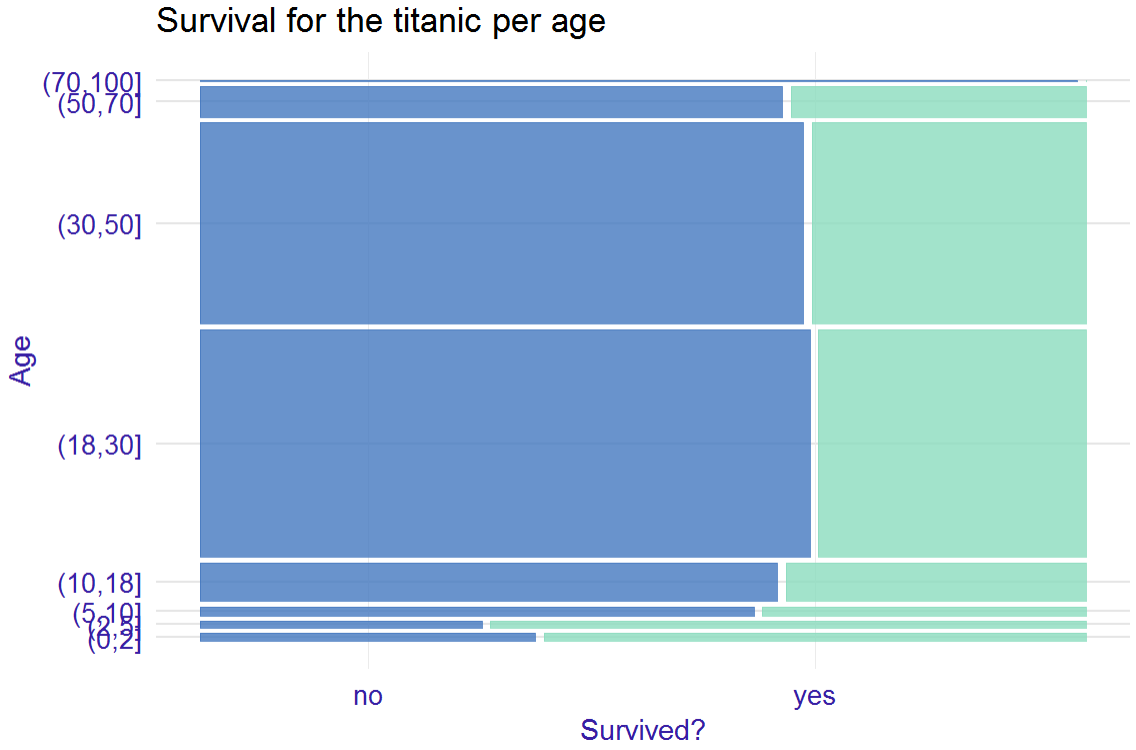
After imputing the missing values, we investigate the association between survival status and the other variables. Figures XXX-XXX present graphically the proportion non- and survivors for different levels of the other variables. The height of the bars (on the y-axis) reflects the marginal distribution (proportions) of the observed levels of the variable. On the other hand, the width of the bars (on the x-axis) provides the information about the proportion of non- and survivors. Note that, to construct the graphs for *age* and *fare*, we categorized the range of the observed values.

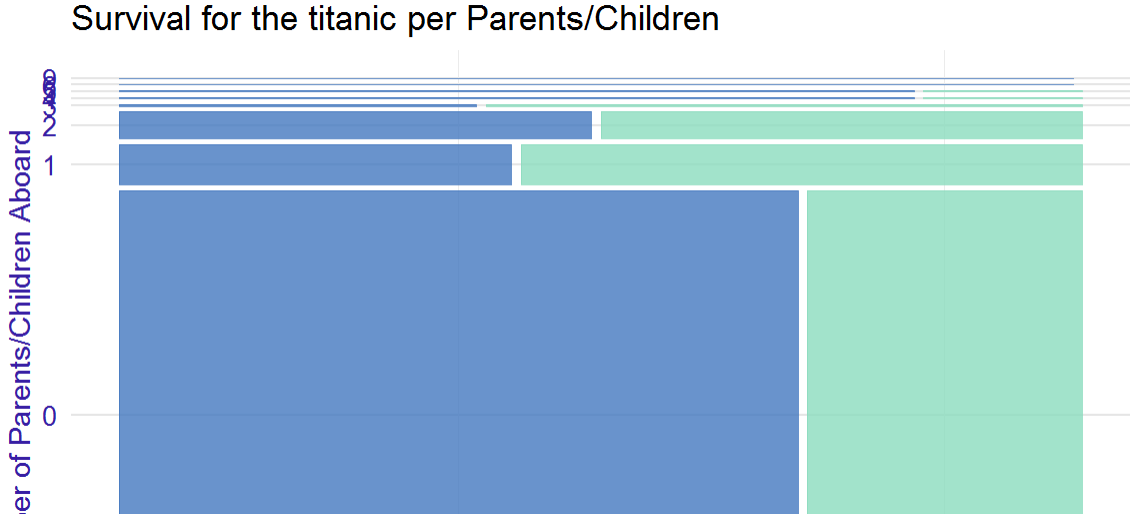
[TOMASZ: MAKE SEPARATE FIGURES SO THAT WE CAN REFER TO THEM. CHANGE THE ORDER, AS SUGGESTED IN THE R CODE. LABEL THE HORIZONTAL “SURVIVAL” AXIS WITH PROPORTIONS. IMPROVE LABELING OF THE Y-AXIS. CATEGORIZE FARE IN A SIMILAR WAY AS FOR AGE.]

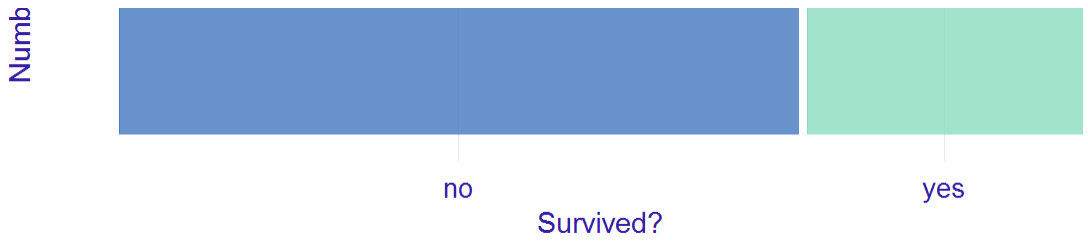
Figures XXX and XXX indicate that the proportion of survivors was larger for females and children below 5 years of age. This is most likely the result of the "women and children first principle that is often evoked in situations that require evacuation of persons whose life is in danger. The principle can, perhaps, partially explain the trend seen in Figures XXX and XXX, i.e., a higher proportion of survivors among those with 1-3 parents/children and 1-2 siblings/spouses aboard. Figure XXX indicates that passengers travelling in the first and second class had a higher chance of survival, perhaps due to the proximity of the location of their cabins to the deck. Interestingly, the proportion of survivors among crew deck was similar to the proportion of the first-class passengers. Figure XXX shows that the proportion of

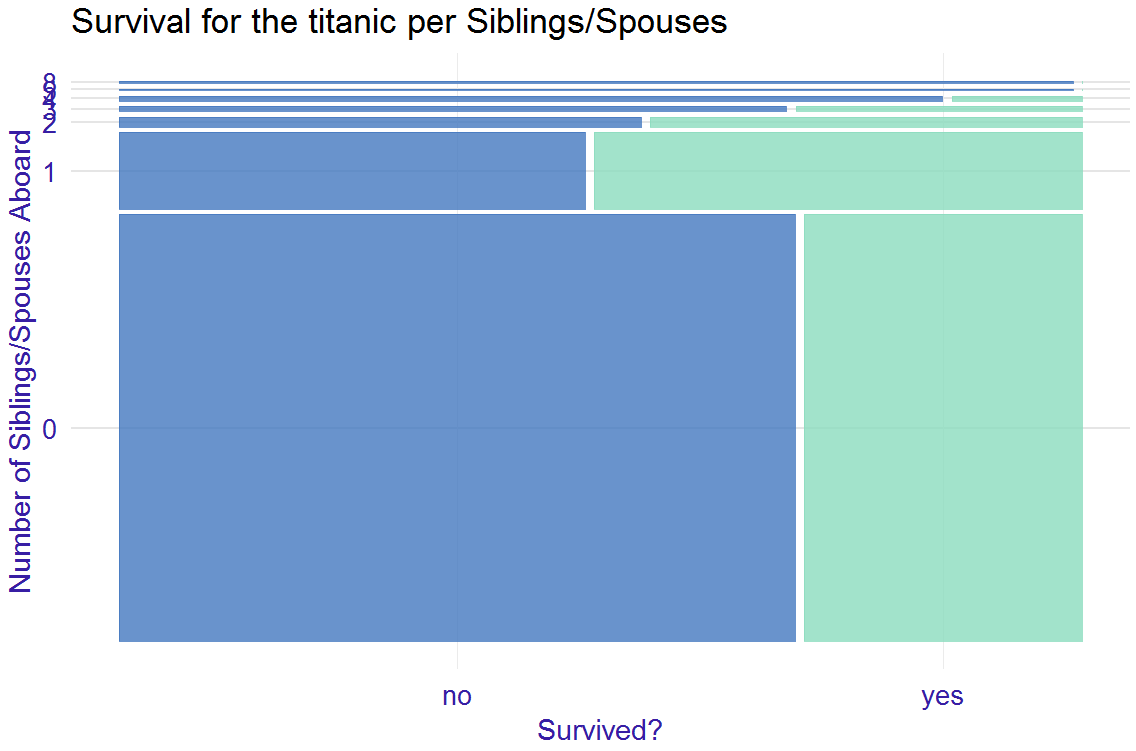
survivors increased with the fare, which is consistent with the fact that the proportion was higher for passengers travelling in the first and second class. Finally, Figures XXX and XXX do not suggest any noteworthy trends.

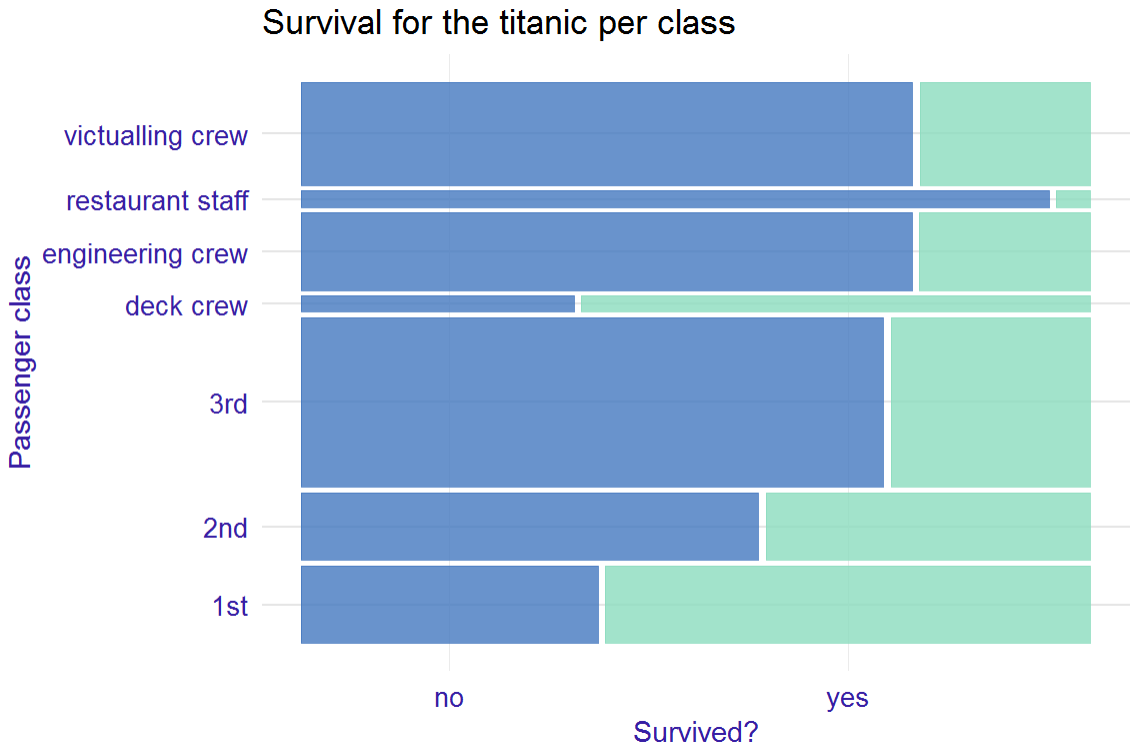

Survival for the titanic per gender
Gender
male
female
no
yes
Survived?

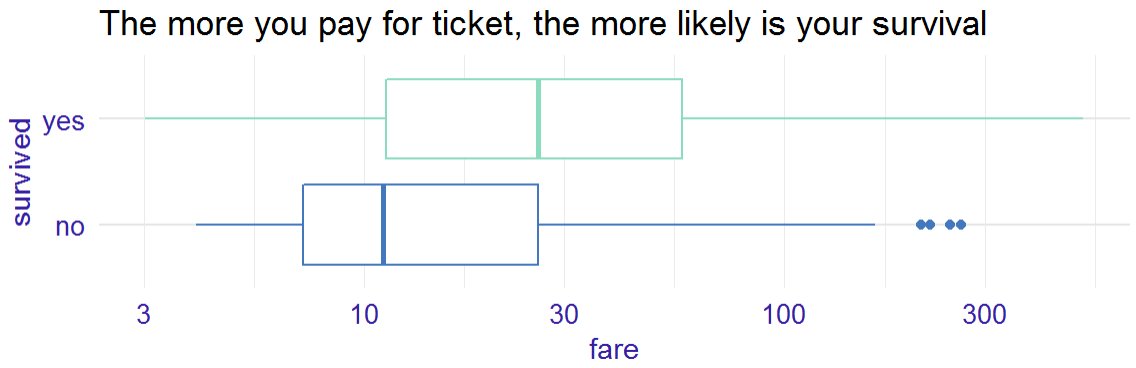

Survival for the titanic per age
Age
(70,100]
(50,70]
(30,50]
(18,30]
(10,18]
(5,10]
(2,5]
(0,2]
no
yes
Survived?

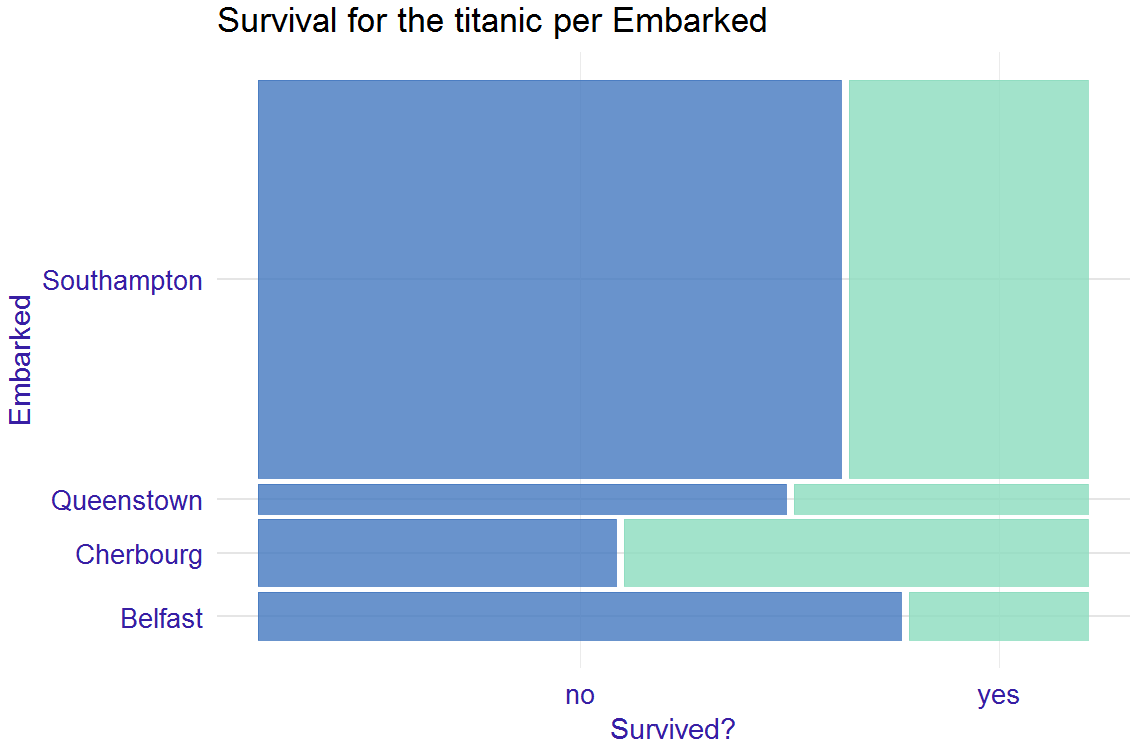

Survival for the titanic per Parents/Children
2
1
0

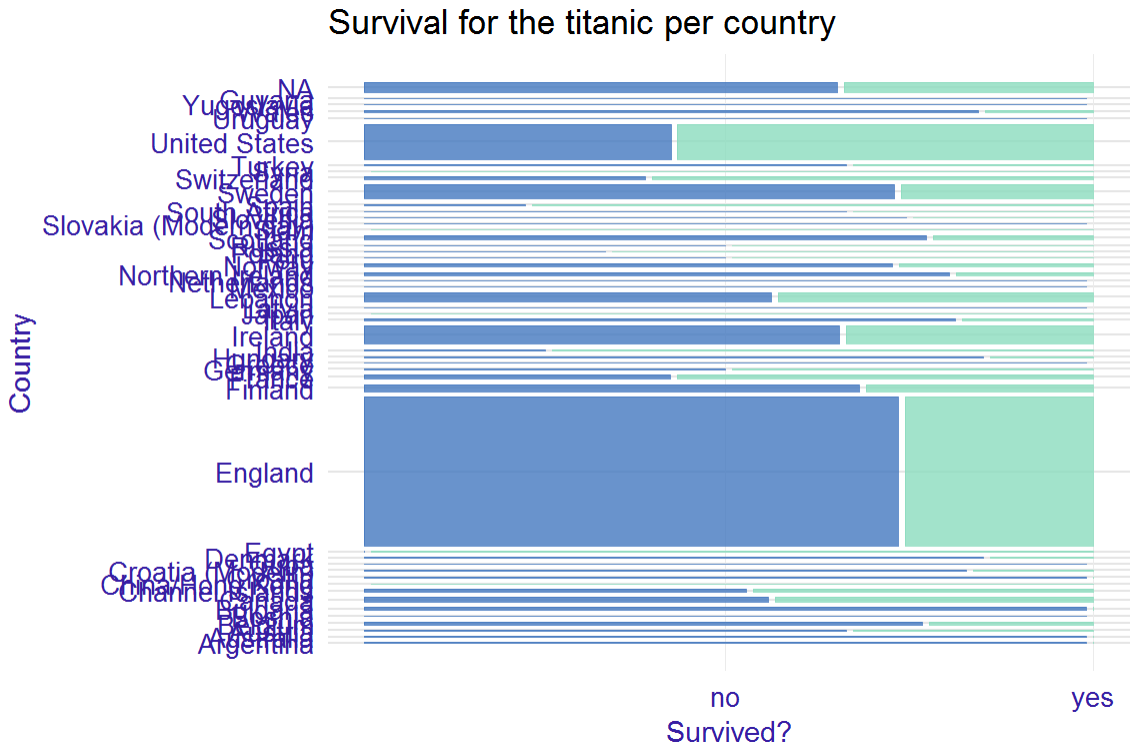

Numb
no
yes
Survived?


Survival for the titanic per Siblings/Spouses
Number of Siblings/Spouses Aboard
1
0
no
yes
Survived?


Survival for the titanic per class
Passenger class
victualling crew
restaurant staff
engineering crew
deck crew
3rd
2nd
1st
no
yes
Survived?


The more you pay for ticket, the more likely is your survival
survived
yes
no
3
10
30
100
300
fare


Survival for the titanic per Embarked
Embarked
Southampton
Queenstown
Cherbourg
Belfast
no
yes
Survived?


Survival for the titanic per country
Country
NA
United States
England
Finland
Ireland
no
yes
Survived?

2.1.2 Logistic regression

The dependent variable of interest, *survival*, is binary. Thus, a natural choice to build a predictive model is logistic regression. We do not consider country as an explanatory variable. As there is no reason to expect a linear relationship between age and odds of survival, we use linear tail-restricted cubic splines, available in the `rcs()` function of the `rms` package (Harrell Jr 2018), to model the effect of age. [TOMASZ: IS THERE A REASON TO ASSUME A LINEAR EFFECT OF FARE?] The results of the model are stored in model-object `titanic_lmr_v6`, which will be used in subsequent chapters.

```
library("rms")
titanic_lmr_v6 <- lrm(survived == "yes" ~ gender + rcs(age) + class + sibsp +
                    parch + fare + embarked, titanic)
titanic_lmr_v6
```

```
## Logistic Regression Model
##
##  lrm(formula = survived == "yes" ~ gender + rcs(age) + class +
##      sibsp + parch + fare + embarked, data = titanic)
##
##                        Model Likelihood     Discrimination    Rank Discrim.
##                           Ratio Test           Indexes           Indexes
##  Obs           2207    LR chi2     752.73    R2       0.404    C       0.817
##   FALSE        1496    d.f.            17    g        1.648    Dxy     0.635
##   TRUE          711    Pr(> chi2) <0.0001    gr       5.195    gamma   0.636
##  max |deriv| 0.0001                          gp       0.282    tau-a   0.277
##                                              Brier    0.146
##
##                          Coef    S.E.   Wald Z Pr(>|Z|)
##  Intercept                4.5458 0.5460   8.33 <0.0001
##  gender=male             -2.7658 0.1587 -17.43 <0.0001
##  age                     -0.1186 0.0221  -5.37 <0.0001
##  age'                     0.6339 0.1629   3.89 <0.0001
##  age''                   -2.6621 0.7841  -3.39 0.0007
##  age'''                   2.8936 1.0131   2.86 0.0043
##  class=2nd               -1.1029 0.2486  -4.44 <0.0001
##  class=3rd               -2.0185 0.2466  -8.18 <0.0001
##  class=deck crew          1.1067 0.3468   3.19 0.0014
##  class=engineering crew  -0.9287 0.2615  -3.55 0.0004
##  class=restaurant staff  -3.1278 0.6571  -4.76 <0.0001
##  class=victualling crew  -1.0473 0.2558  -4.09 <0.0001
##  sibsp                   -0.4574 0.1012  -4.52 <0.0001
##  parch                   -0.0970 0.0992  -0.98 0.3282
##  fare                     0.0021 0.0020   1.05 0.2928
##  embarked=Cherbourg       0.7725 0.2844   2.72 0.0066
##  embarked=Queenstown      0.2653 0.3411   0.78 0.4368
##  embarked=Southampton     0.2287 0.2119   1.08 0.2805
##
```

2.1.3 Random forest

As an alternative to a logistic regression model, we consider a random forest model. Random forest is known for good predictive performance, is able to grasp low-level variable interactions, and is quite stable [TOMASZ: REFERENCE?]. To fit the model, we apply the `randomForest()` function, with default settings, from the package with the same name (Liaw and Wiener 2002).

In the first instance, we fit a model with the same set of explanatory variables as the logistic regression model. The results of the model are stored in model-object `titanic_rf_v6`.

```
library("randomForest")
titanic_rf_v6 <- randomForest(survived ~ class + gender + age + sibsp + parch + fare + e
                              data = titanic)
titanic_rf_v6
```

```
##
## Call:
##  randomForest(formula = survived ~ class + gender + age + sibsp +      parch + fare +
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 18.71%
## Confusion matrix:
##       no yes class.error
## no  1393 103  0.06885027
## yes  310 401  0.43600563
```

For comparison purposes, we also consider a model with only three explanatory variables: *class*, *gender*, and *age*. The results of the model are stored in model-object `titanic_rf_v3`.

```
titanic_rf_v3 <- randomForest(survived ~ class + gender + age, data = titanic)
titanic_rf_v3
```

```
##
## Call:
##  randomForest(formula = survived ~ class + gender + age, data = titanic)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 20.93%
## Confusion matrix:
##       no yes class.error
## no  1349 147  0.09826203
## yes  315 396  0.44303797
```

2.1.4 Gradient boosting

Finally, we consider the gradient-boosting model. The model is known for being able to grasp deep interactions between variables. [TOMASZ: WHAT ARE “DEEP INTERACTIONS”? REFERENCE?] We use the same set of explanatory variables as for the logistic regression model. To fit the gradient-boosting model, we use the function `gbm()` from the `gbm` package (Ridgeway 2017). The results of the model are stored in model-object `titanic_gbm_v6` .

```
library("gbm")
titanic_gbm_v6 <- gbm(survived == "yes" ~ class + gender + age + sibsp + parch + fare +
                      data = titanic, n.trees = 15000)
```

```
## Distribution not specified, assuming bernoulli ...
```

```
titanic_gbm_v6
```

```
## gbm(formula = survived == "yes" ~ class + gender + age + sibsp +
##     parch + fare + embarked, data = titanic, n.trees = 15000)
## A gradient boosted model with bernoulli loss function.
## 15000 iterations were performed.
## There were 7 predictors of which 7 had non-zero influence.
```

2.1.5 Model predictions

Let us now compare predictions that are obtained from the three different models. In particular, we will compute the predicted probability of survival for an 8-year-old boy who embarked in Belfast and travelled in the 2nd class with no parents nor siblings with a ticket costing 72 pounds. First, we create a data frame `henry` that contains the data describing the passenger.

```
henry <- data.frame(
            class = factor("2nd", levels = c("1st", "2nd", "3rd", "deck crew", "engineer
            gender = factor("male", levels = c("female", "male")),
            age = 8,
            sibsp = 0,
            parch = 0,
            fare = 72,
            embarked = factor("Belfast", levels = c("Belfast","Cherbourg","Queenstown",'
)
```

Subsequently, we use the generic function `predict()` to get the predicted probability of survival for the logistic regression model.

```
(pred_lmr = predict(titanic_lmr_v6, henry, type = "fitted"))
```

```
##         1
## 0.4702145
```

The predicted probability is equal to 0.47.

We do the same for the random forest and gradient boosting models.

```
(pred_rf = predict(titanic_rf_v6, henry, type = "prob"))
```

```
##      no   yes
## 1 0.646 0.354
## attr(,"class")
## [1] "matrix" "votes"
```

```
(pred_gbm = predict(titanic_gbm_v6, henry, type = "response", n.trees = 15000))
```

```
## [1] 0.3597424
```

As a result, we obtain the predicted probabilities of 0.35 and 0.36, respectively.

The models lead to different probabilities. Thus, it might be of interest to understand the reason for the differences, as it could help us to decide which of the predictions we might want to trust.

[TOMASZ: GRADIENT-BOOSTING LEADS TO DIFFERENT PREDICTIONS AT EACH RUN. POSSIBLE TO “FREEZE” THE MODEL/PREDICTIONS? PERHAPS AT A RUN GIVING A DISTINCT PREDICTION? AT ONE POINT I GOT 0.6 FOR GBM, WHICH WAS MARKEDLY - AND INTERESTINGLY - DIFFERENT FROM LR AND RF.]

2.2 Apartment prices

Warsaw skyscrapers by Artur Malinowski Flicker

Predicting house prices is a common exercise used in machine-learning courses. Various datasets for house prices are available at websites like Kaggle (https://www.kaggle.com) or UCI Machine Learning Repository (https://archive.ics.uci.edu).

In this book we will work with an interesting version of this problem. The `apartments` dataset is an artificial dataset created to match key characteristics of real apartments in Warszawa, the capital of Poland. However, the dataset is created in a way that two very different models, namely linear regression and random forest, have almost exactly the same accuracy. The natural question is which model should we choose? We will show that the model-explanation tools provide important insight into the key model characteristics and are helpful in model selection.

The dataset is available in the `DALEX` package (Biecek 2018). It contains 1000 observations (apartments) and six variables:

- *m2.price*, apatments price per meter-squared (in EUR), a numerical variable
- *construction.year*, the year of construction of the block of flats in which the apartment is located, a numerical variable
- *surface*, apartment's total surface in squared meters, a numerical variable
- *floor*, the floor at which the apartment is located (ground floor taken to be the first floor), a numerical integer variable with values from 1 to 10

- *no.rooms*, the total number of rooms, a numerical variable with values from 1 to 6
- *distric*, a factor with 10 levels indicating tha distric of Warszawa where the apartment is located

Models considered for this dataset will use *m2.price* as the (continuous) dependent variable.

```
library("DALEX")
head(apartments, 2)
```

```
##   m2.price construction.year surface floor no.rooms    district
## 1     5897              1953      25     3        1 Srodmiescie
## 2     1818              1992     143     9        5     Bielany
```

```
str(apartments)
```

```
## 'data.frame':    1000 obs. of  6 variables:
##  $ m2.price         : num  5897 1818 3643 3517 3013 ...
##  $ construction.year: num  1953 1992 1937 1995 1992 ...
##  $ surface          : num  25 143 56 93 144 61 127 105 145 112 ...
##  $ floor            : int  3 9 1 7 6 6 8 8 6 9 ...
##  $ no.rooms         : num  1 5 2 3 5 2 5 4 6 4 ...
##  $ district         : Factor w/ 10 levels "Bemowo","Bielany",..: 6 2 5 4 3 6 3 7 6 6
```

```
table(apartments$floor)
```

```
##
##   1   2   3   4   5   6   7   8   9  10
##  90 116  87  86  95 104 103 103 108 108
```

```
table(apartments$no.rooms)
```

```
##
##   1   2   3   4   5   6
##  99 202 231 223 198  47
```

```
levels(apartments$district)
```

```
##  [1] "Bemowo"      "Bielany"     "Mokotow"     "Ochota"      "Praga"
##  [6] "Srodmiescie" "Ursus"       "Ursynow"     "Wola"        "Zoliborz"
```

Model predictions will be obtained for a set of six apartments included in data frame `apartments_test`, also included in the `DALEX` package.

```
head(apartments_test)
```

```
##      m2.price construction.year surface floor no.rooms    district
## 1001     4644              1976     131     3        5 Srodmiescie
## 1002     3082              1978     112     9        4     Mokotow
## 1003     2498              1958     100     7        4     Bielany
## 1004     2735              1951     112     3        5        Wola
## 1005     2781              1978     102     4        4      Bemowo
## 1006     2936              2001     116     7        4      Bemowo
```

2.2.1 Data exploration

[TOMASZ: TO ADD, EVEN IF SHORT]

2.2.2 Linear regression

The dependent variable of interest, *m2.price*, is continuous. Thus, a natural choice to build a predictive model is linear regression. We treat all the other variables in the `apartments` data frame as explanatory and include them in the model. The results of the model are stored in model-object `apartments_lm_v5`.

```
apartments_lm_v5 <- lm(m2.price ~ ., data = apartments)
apartments_lm_v5
```

```
##
## Call:
## lm(formula = m2.price ~ ., data = apartments)
##
## Coefficients:
##         (Intercept)    construction.year              surface
##            5020.139              -0.229             -10.238
##               floor             no.rooms     districtBielany
##             -99.482             -37.730              17.214
##     districtMokotow       districtOchota       districtPraga
##             918.380             926.254             -37.105
## districtSrodmiescie        districtUrsus     districtUrsynow
##            2080.611              29.942             -18.865
##        districtWola     districtZoliborz
##             -16.891             889.973
```

2.2.3 Random forest

As an alternative to linear regression, we consider a random forest model. To fit the model, we apply the `randomForest()` function, with default settings, from the package with the same name (Liaw and Wiener 2002).
The results of the model are stored in model-object `apartments_rf_v5` .

```
library("randomForest")
set.seed(72)
```

```
apartments_rf_v5 <- randomForest(m2.price ~ ., data = apartments)
apartments_rf_v5
```

```
##
## Call:
##  randomForest(formula = m2.price ~ ., data = apartments)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##           Mean of squared residuals: 79789.39
##                     % Var explained: 90.28
```

2.2.4 Model predictions

By aplying the `predict()` function to model-object `apartments_lm_v5` with `apartments_test` as the data frame for which predictions are to be computed, we obtain the predicted prices for the testing set of six apartments for the linear regression model. Subsequently, we compute the mean squared difference between the predicted and actual prices for the test apartments. We repeat the same steps for the random forest model.

```
predicted_apartments_lm <- predict(apartments_lm_v5, apartments_test)
rmsd_lm <- sqrt(mean((predicted_apartments_lm - apartments_test$m2.price)^2))
rmsd_lm
```

```
## [1] 283.0865
```

```
library("randomForest")
predicted_apartments_rf <- predict(apartments_rf_v5, apartments_test)
rmsd_rf <- sqrt(mean((predicted_apartments_rf - apartments_test$m2.price)^2))
rmsd_rf
```

```
## [1] 282.9519
```

For the random forest model, the square-root of the mean squared difference is equal to 283. It is only minimally smaller than the value of 283.1, obtained for the linear regression model. Thus, the question we may face is: should we choose the model complex, but flexible random-

forest model, or the simpler and easier to interpret linear model? In the subsequent chapters we will try to provide an answer to this question.

2.3 Hire or fire

Predictive models can be used to support decisions. For instance, they could be used in a human-resources department to decide whether, for instance, promote an employee. An advantage of using a model for this purpose would be the objectivity of the decision, which would not be subject to personal preferences of a manager. However, in such a situation, one would most likely want to understand what influences the model's prediction.

To illustrate such a situation, we will use the `HR` dataset that is available in the `DALEX` package (Biecek 2018). It is an artificial set of data from a human-resources department of a call center. It contains 7847 observations (employees of the call center) and six variables:

- *gender*, person's gender, a factor with two levels
- *age*, person's age in years, a numerical variable
- *hours*, average number of working hours per week, a numerical variable
- *evaluation*, the last evaluation score, a numerical variable with values 2 (fail), 3 (satisfactory), 4 (good), and 5 (veru good) [TOMASZ: CORRECT?]
- *salary*, the salary level, a numerical variable with values from 0 to 5 [TOMASZ: WHAT DOES 0 MEAN?]
- *status*, a factor with three indicating whether the employee was fired, retained, or promoted

Models considered for this dataset will use *status* as the (categorical) dependent variable.

```
library("DALEX")
head(HR, 4)
```

```
##   gender      age    hours evaluation salary status
## 1   male 32.58267 41.88626          3      1  fired
## 2 female 41.21104 36.34339          2      5  fired
## 3   male 37.70516 36.81718          3      0  fired
## 4 female 30.06051 38.96032          3      2  fired
```

```
str(HR)
```

```
## 'data.frame':    7847 obs. of  6 variables:
##  $ gender    : Factor w/ 2 levels "female","male": 2 1 2 1 2 2 1 2 1 1 ...

##  $ age       : num  32.6 41.2 37.7 30.1 21.1 ...
##  $ hours     : num  41.9 36.3 36.8 39 62.2 ...
##  $ evaluation: num  3 2 3 3 5 2 4 2 2 4 ...
##  $ salary    : num  1 5 0 2 3 0 0 4 4 4 ...
##  $ status    : Factor w/ 3 levels "fired","ok","promoted": 1 1 1 1 3 1 3 2 1 3 ...
```

```
table(HR$evaluation)
```

```
##
##    2    3    4    5
## 2371 2272 1661 1543
```

```
table(HR$salary)
```

```
##
##    0    1    2    3    4    5
## 1105 1417 1461 1508 1316 1040
```

2.3.1 Multinomial logistic regression

The dependent variable of interest, *status*, is categorical with three categories. Thus, a simple choice is to consider a multinomial logistic regression model [TOMASZ: REFERENCE]. We fit the model with the help of function `multinom` from package `nnet`. The function fits multinomial log-linear models by using the neural-networks approach. [TOMASZ: WHY THIS APPROACH?] We treat all variables other than `status` in the `HR` data frame as explanatory and include them in the model. The results of the model are stored in model-object `HR_glm_v5`.

```r
library("nnet")
HR_glm_v5 <- multinom(status ~ gender + age + hours + evaluation + salary, data = HR)
```

```
## # weights:  21 (12 variable)
## initial  value 8620.810629
## iter  10 value 7002.127738
## iter  20 value 6239.478146
## iter  20 value 6239.478126
## iter  20 value 6239.478124
## final  value 6239.478124
## converged
```

```r
HR_glm_v5
```

```
## Call:
## multinom(formula = status ~ gender + age + hours + evaluation +
##     salary, data = HR)
##
## Coefficients:
##          (Intercept) gendermale         age      hours  evaluation
## ok         -3.199741 0.05185293 0.001003521 0.06628055 -0.03734345
## promoted  -12.677639 0.11838037 0.003436872 0.16253343  1.26109093
##               salary
## ok         0.01680039
## promoted 0.01507927
##
## Residual Deviance: 12478.96
## AIC: 12502.96
```

2.3.2 Random forest

As an alternative to multinomial logisitc regression, we consider a random forest model. To fit the model, we apply the `randomForest()` function, with default settings, from the package with the same name (Liaw and Wiener 2002). The results of the model are stored in model-object `HR_rf_v5` .

```
set.seed(59)
library("randomForest")
HR_rf_v5 <- randomForest(status ~ gender + age + hours + evaluation + salary, data = HR)
HR_rf_v5
```

```
##
## Call:
##  randomForest(formula = status ~ gender + age + hours + evaluation +      salary, dat
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 27.31%
## Confusion matrix:
##          fired   ok promoted class.error
## fired     2274  384      197   0.2035026
## ok         525 1255      441   0.4349392
## promoted   201  395     2175   0.2150848
```

2.3.3 Model predictions

[TOMASZ: TO ADD]

2.4 List of models

In the previous sections we have built several predictive models for differnt datasets. The models will be used in the rest of the book to illustrate the mdoel explanation methods and tools. For the ease of reference, we summarize the models in the table beloW. The binary model-objects can be donwloaded by using the attached `archivist` hooks (Biecek and Kosinski 2017). [TOMASZ: THIS IS USEFUL FOR E-BOOK. HOW ABOUT THE PRINTED VERSION?]

Model name	Model generator	Dataset	Variables	Link to the mo
titanic_lmr_v6	rms:: lmr	DALEX:: titanic	gender, age, class, sibsp, parch, fare, embarked	archivist:: aread("pbiecek/models
titanic_rf_v6	randomForest:: randomForest	DALEX:: titanic	gender, age, class, sibsp, parch, fare, embarked	archivist:: aread("pbiecek/models
titanic_rf_v3	randomForest:: randomForest	DALEX:: titanic	gender, age, class	archivist:: aread("pbiecek/models
titanic_gbm_v6	gbm:: gbm	DALEX:: titanic	gender, age, class, sibsp, parch, fare, embarked	archivist:: aread("pbiecek/models
apartments_lm_v5	stats:: lm	DALEX:: apartments	construction .year, surface, floor, no.rooms, district	archivist:: aread("pbiecek/models
apartments_rf_v5	randomForest:: randomForest	DALEX:: apartments	construction .year, surface, floor, no.rooms, district	archivist:: aread("pbiecek/models

Model name	Model generator	Dataset	Variables	Link to the mo
HR_rf_v5	randomForest:: randomForest	DALEX:: HR	gender, age, hours, evaluation, salary	archivist:: aread("pbiecek/models
HR_glm_v5	stats:: glm	DALEX:: HR	gender, age, hours, evaluation, salary	archivist:: aread("pbiecek/models

References

Harrell Jr, Frank E. 2018. *Rms: Regression Modeling Strategies*. https://CRAN.R-project.org/package=rms.

Liaw, Andy, and Matthew Wiener. 2002. “Classification and Regression by randomForest.” *R News* 2 (3): 18–22. https://CRAN.R-project.org/doc/Rnews/.

Ridgeway, Greg. 2017. *Gbm: Generalized Boosted Regression Models*. https://CRAN.R-project.org/package=gbm.

Biecek, Przemyslaw. 2018. *DALEX: Descriptive mAchine Learning Explanations*. https://pbiecek.github.io/DALEX/.

Biecek, Przemyslaw, and Marcin Kosinski. 2017. “archivist: An R Package for Managing, Recording and Restoring Data Analysis Results.” *Journal of Statistical Software* 82 (11): 1–28. https://doi.org/10.18637/jss.v082.i11.