Comparing and Intereting Machine Learning Algorithms Estimating Technical Prices

Joakim Bilyk, Sebastian Cramer and Teis Blem *University of Copenhagen*

This document provides a practical example of applications of machine learning algorithms to car insurance data using the `mlr3` package. A popular model used in pricing of non-life insurance policies is the frequency-severity model, where the price is decomposed into the product of the probability of a claim arrises and the expected claim size given a claim occurs. This model is fitted using machine learning algorithms such as penalized linear regression and random forests. This paper argues that the ranger model gives a particular well fit to both the frequency and severity model.

Keywords: mlr3, machine learning, regression, non-life insurance, estimating technical price, XGBoost, Ranger, Bart, Elastic net regression, Generalized Additive Models

Contents

Getting familiar with the data

The data we use in this project is on the form of a table where out main objective is model the claim size Y_i given the explanatory variables X_i in the table. In particular, we want to construct an estimator that predicts an expecected claim size given the information available i.e. the quantity $\mathbb{E}[Y \mid X]$.

Missing values. As with any statistical modelling we start by doing some exploratory analysis of the data `freMPL1`. As it was seen in the previous section, the number of variables (columns) missing datapoints were only one. The one in question is `RecordEnd` which has 14143 out of 30595 missing values. This does not bather us since we have the variable `Exposure` giving the time difference between `RecordBeg` and `RecordEnd` in calendar years. Furthermore we have that `RecordBeg` ranges from 2004-01-01 to 2004-12-31 and `RecordBeg + 365.25*Exposure` being at most 2004-12-31 meaning that all contracts span within the year 2004. Assuming no seasonality trends this would incentivice us to remove the two variables `RecordBeg` and `RecordEnd`.

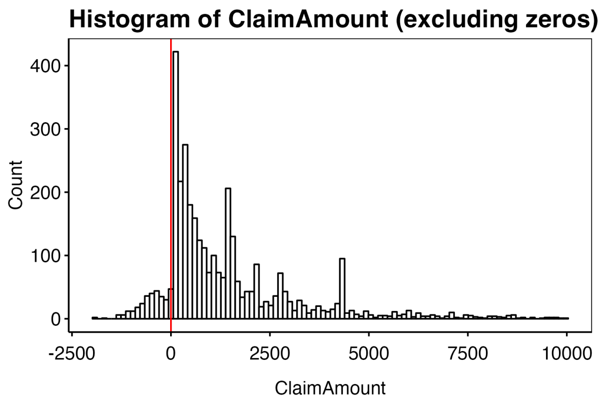
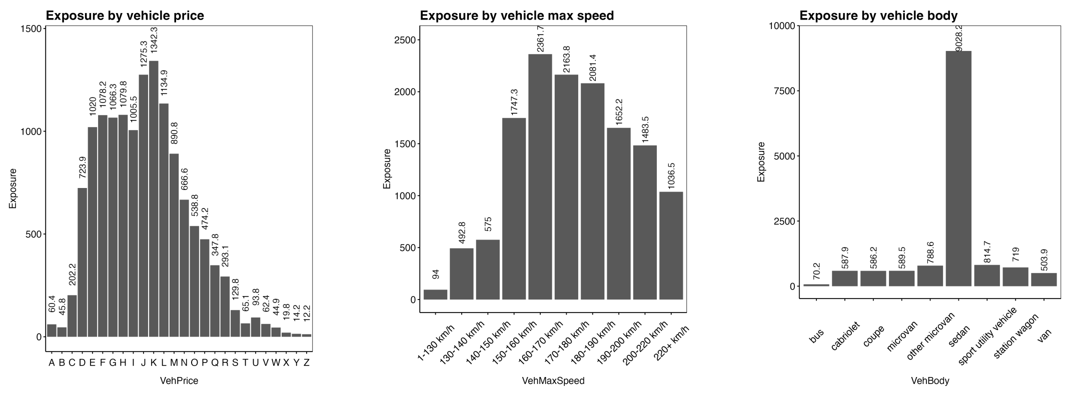


Figure 1: Histogram of the variable 'ClaimAmount'.

ClaimAmount and ClaimInd. The variable of interest, `ClaimAmount`, excibits a strange behaviour as it contains 285 strictly negative values. This is seen in figure 1. As this does not intuitively makes sense we will set these values as zero and ensure that the `ClaimInd` reflects this change.

VehEngine and VehEnergy. *(Vehicle specific, 1)* Regarding the categorical variables we notice that some levels is has sparse data. The variables `VehEngine` and `VehEnergy` has both the levels `GPL` and `electric`. We do however not have any substantial datapoints as in total these levels contain 8 observations. As the total dataset has 30595 observations in total we choose to remove these observations.



(a) (b) (c)

Figure 2: Exposure by respectively vehicle price (a), max speed (b) and body (c).

VehPrice, VehMaxSpeed and VehBody. *(Vehicle specific, 2)* In figure 2 we see that `VehPrice` is well represented in most price categories. We do however har a shorter supply of data in the tails. We will therefore combine the lowest three categories A through C, the levels R through T and lastly U through Z. Regarding to the vehicle max speed we see that a very few number of observations are in the lowest category `1-130 kmh`. We therefore combine the lowest level with the level `130-140 kmh`. The only vehicle body with very few observations is `bus` with only 159 observations. We do however see that `bus` act much like the category `sedan` with respect to frequency and severity and so these are combined under `sedan`.

VehAge, VehUsage and VehClass. *(Vehicle specific, 3)* The remaining three vehicle specific variable is well represented throughout all levels. The only scarce observation is `VehUsage` being `Professional run`. We therefore combine `Proffesional` and `Professional run` under `Professional`. We leave the remaining levels as is.

SocioCateg and Gender. *(Socails)* When constructing a statistical model, one does not simply have to consider which variables have the most explanatory value but one also have to take into consideration the lawfullness of discriminating customors based on covariates as gender, race and so forth. It is common knowledge that insurance companies cannot discriminate based on gender and so we will not use the variable `gender` as explanatory variable even though it might improve the model predictions. The variable `SocioCateg` representing the socioeconomic status of the insured ranges between category 1 and 99. It is not at the moment clear whether this is a covariate that may be used in pricing, so we will prefer not using it. However, if it does indeed improve the fit without overfitting, we may get some additional information from this variable. The data is indicate that the custumors in general are in the category 50 with a few other levels having significant more observations than others. For this reason we combine the catagories from 1 to 49 into `A` and the catagories 51 to 99 into `C` and keep the category `B` as is.

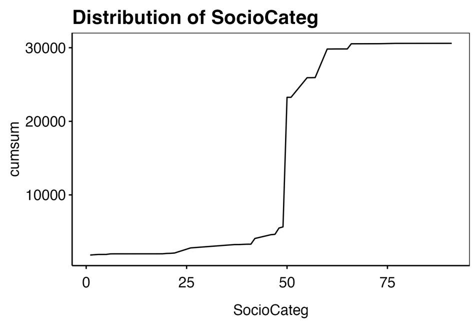


Figure 3: Cummulative distribution of the variable ‘SocioCateg‘.

LicAge, DrivAge and MariStat. *(Customer specific)* In general in France, one can acquire a drivers license at the age of 18 and so we have the obvious restriction `LicAge <= DrivAge - 18` and we will in general have that the license age will be approximately 18 years less than the drivers license. It is therefore reasonable to discuss whether to include both variables. We would however assume that for an older person the license age would be more important than for youngsters. Furthermore, we would assume that the `ClaimInd` and the `LicAge` are negative correlated. We will therefore include both. Both `Alone` and `Other` is well represented in `MariStat`.

HasKmLimit, RiskVar, Garage and BonusMalus *(Policy related and others)* The variables remaining HasKmLimit, RiskVar, Garage and BonusMalus are well represented throughout all levels and so no action is taken here.

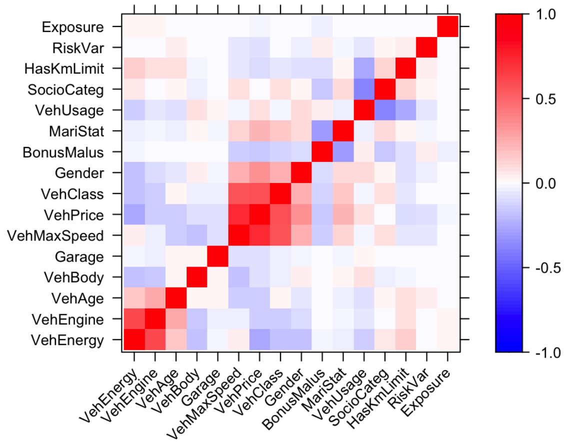


Figure 4: Spearman correlation matrix.

Correlations. As discussed previously the age of the driver and the license age i very correlated and so one may consider whether or not both variables are needed. In practice we do however see that age and experience are used when modelling the premium. Secondly, we see that the covariates vehicle engine and energy are very correlated and also the three variables vehicle max speed, price and class are well correlated. Thirdly, we see that vehicle usage and socio category are correlated. This is likely because wealthier people more often have a car for professional use.

Modelling the technical premium

The technical premium is based upon a frequency/severity model. We have the base model assumptions as: Let $Y := Y_{t+\Delta t}$ be the claim risen by a policy during the interval $[t, t+\Delta t)$ with Δt representing the exposure. Notice that in principle Δt is a stopping time defined as

$$\Delta t := \min\left(1, \inf\{s \geq t : Y_s > 0\}\right),$$

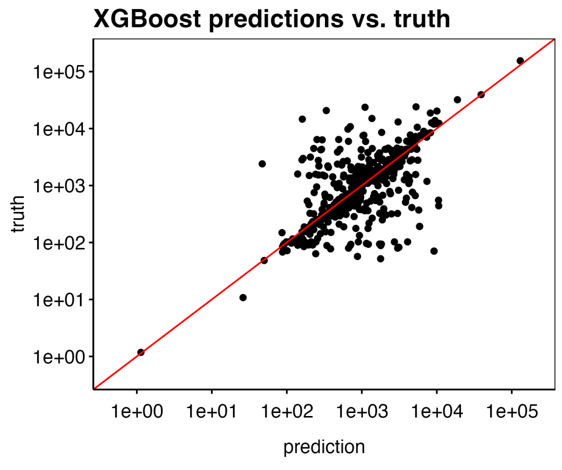
meaning the policy is terminated at the time $t + \Delta t$ with $t + \Delta t = t + 1$ if $Y = 0$. Notice that the exposure is censored if $\Delta t_i < 1$ and $Y_i = 0$. We furthermore have the covariates $X \in \mathcal{X}$ with $\mathcal{X}$ being a p-dimensional space. We are interested in the object $\mathbb{E}[Y \mid X]$ being the expected claim risen given the covariates X. Our main assumption is that this expectation is decomposed into

$$\mathbb{E}[Y \mid X] = \mathbb{E}[Y 1_{Y>0} \mid X] = \mathbb{E}[Y \mid X, 1_{Y>0}] \cdot \mathbb{E}[1_{Y>0} \mid X] = \mu_X \cdot p_X,$$

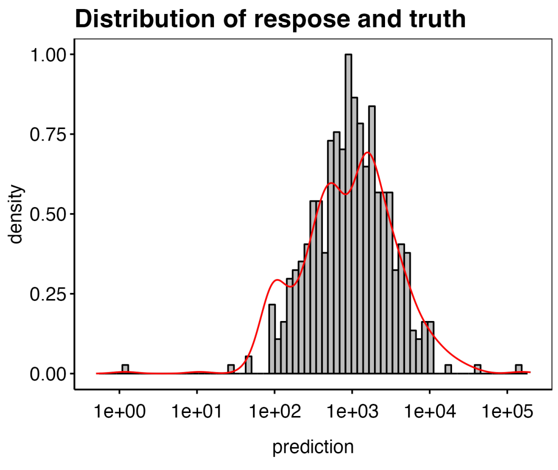
where μ_X is the expected claim in the event, that a claim arises i.e. the severity and p_X is the probability that a claim arises i.e. the frequency. However since the data is censored with exposure $\tilde{\Delta t} \leq \Delta t$ we need to consider how we may translate a predictive model into a price function for new policies. In practice, we include exposure as a covariate.

We will search for the most efficient estimators for both the severity and frequency. We will denote these estimators by $m_\mu(X)$ and $m_p(X)$ where we will denote the model by a superscript $m_j^{(*)}$ with $*$ denoting the model for $j = \mu, p$.

Severity



(a)



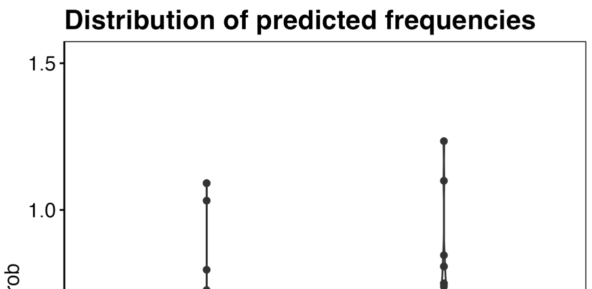
(b)

Figure 5: (a): Predictions from the model with y being the actual value and x being the estimate. The red line represent the mapping $y = x$. (b): Distribution of the response and actual values. The red density function represents the empirical density of the test data set.

Frequency

Initial estimate of the technical premium

The technical price is given by the product



$$\pi = \mu_X \cdot p_X.$$

Notice that the length of the contract i.e. the exposure is used as a covariate in the frequency/severity model, and so the effect of longer contracts is already priced in. By default new contracts are sold with a yearly length, however in the case of estimating on the dataset we use the given `Exposure`. Since the risk is the less with the ranger algorithm we estimate π by

$$\hat{\pi} = \hat{m}_{\mu}^{(1)}(X) \cdot \hat{m}_{p}^{(1)}(X) \cdot e.$$

We can apply the technical price onto the dataset `df` by running the following code:

```
# Estimating mu
sev_covariates <- df[,sev_columns] %>%
  select(-ClaimInd,-ID,-ClaimAmount,-Exposure)
sev_covariates <-  as.matrix(sev_covariates)
m_mu_ranger <- exp(predict(xgb_regr, newdata = sev_covariates))
# Estimating p
freq_covariates <- df[,freq_columns] %>%
  select(-ClaimInd,-ID,-ClaimAmount)
freq_covariates <- as.matrix(freq_covariates)
m_p_ranger <- predict(xgb_freq_train, newdata = freq_covariates)
freq_covariates[,"Exposure"] <- 1
m_p_ranger_2 <- predict(xgb_freq_train, newdata = freq_covariates)
# Inserting estimates
df[,"pi_hat"] <- m_mu_ranger * m_p_ranger
df[,"mu_hat"] <- m_mu_ranger
df[,"p_hat"] <- m_p_ranger
df[,"pi_hat_new"] <- m_mu_ranger * m_p_ranger_2
```

We can summarize the models predictions on the policies below.

Policy	Exposure	$\hat{\mu}_X$	$\hat{p}_X$	$\hat{\pi}$
6257	0.08	884.87	0.00804	7.11
24131	0.2	3099.74	0.60195	1865.89
25018	0.1	1205.1	0.0198	23.86
25196	0.08	427.6	0.01551	6.63
30503	0.05	6184.3	0.00014	0.86
30563	0.41	1317.45	0.00398	5.24

If we set the exposure to one we get new estimates and the technical price the insurance firm will give to new contracts.

Policy	Exposure	$\hat{\mu}_X$	$\hat{p}_X$	$\hat{\pi}$
6257	1	884.87	0.24141	213.62

Policy	Exposure	$\hat{\mu}_X$	$\hat{p}_X$	$\hat{\pi}$
24131	1	3099.74	0.15212	471.54
25018	1	1205.1	0.22273	268.41
25196	1	427.6	0.1326	56.7
30503	1	6184.3	0.06735	416.5
30563	1	1317.45	0.00131	1.73

In the table above we have altered the dataset with exposure set to 1 for all rows, to predict the price of a new contract i.e. with exposure of one year. One sees that the prices for the new contracts are larger than the technical price for the historical as the exposure is larger. Furthermore as expected we see that the estimate $\hat{\mu}_X$ does not depend on the exposure.

Decomposing the estimates

Diskrimination and debaissing of the model

Repreducing statement

Packages

```
library(CASdatasets)
library(lattice)
library(evmix)
library(ggplot2)
library(mlr3)
library(mlr3learners)
library(mlr3extralearners)
library(dbarts)
library(mlr3mbo)
library(mlr3measures)
library(mlr3tuning)
library(ranger)
library(mlr3viz)
library(fastDummies)
library(dplyr)
library(patchwork)
library(xgboost)
library(glex)
library(treeshap)
```

Statistical data

The data we will be working with is the `freMPL1` (*French Motor Personal Line datasets*) dataset from the package `CASdatasets` (*Computational Actuarial Science datasets*). The author write the following regarding the dataset

This collection of ten datasets comes from a private motor French insurer. Each dataset includes risk features, claim amount and claim history of around 30,000 policies for year 2004.

A detailed description of the variables may be read in the below table.

Variable	Description
Exposure	The exposure, in years.
LicAge	The driving licence age, in months.
RecordBeg	Beginning date of record.
RecordEnd	End date of record.
VehAge	The vehicle age, in years.
Gender	The gender, either “Male” or “Female”.
MariStat	The marital status, either “Alone” or “Other”.

Variable	Description
SocioCateg	The social category known as CSP in France, between “CSP1” and “CSP99”.
VehUsage	The vehicle usage among “Private”, “Private+trip to office” “Professional”, “Professional run”.
DrivAge	The driver age, in years (in France, people can drive a car at 18).
HasKmLimit	A numeric, 1 if there is a km limit for the policy, 0 otherwise.
BonusMalus	A numeric for the bonus/malus, between 50 and 350: <100 means bonus, >100 means malus in France.
VehBody	The vehicle body, among “bus”,“cabriolet”,“coupe”,“microvan”,“othermicrovan”,
VehPrice	The category of the vehicle price from “A” (cheapest) to “Z” (most expensive).
VehEngine	The vehicle engine, among “carburation”,“directinjectionoverpowered”,“electric”, “GPL”, “injection”, “injection overpowered”.
VehEnergy	The vehicle energy, among “diesel”, “eletric”, “GPL”, “regular”.
VehMaxSpeed	The VehMaxSpeed, among “1-130km/h”,“130-140km/h”,“140-150km/h”,“150-160 km/h”, “160-170 km/h”, “170-180 km/h”, “180-190 km/h”, “190-200 km/h”, “200-220 km/h”, “220+ km/h”.
VehClass	The vehicle class (unknown categories), among “0”, “A”, “B”, “H”, “M1”, “M2”.
ClaimAmount	Total claim amount of the guarantee.
RiskVar	Unkonw risk variable between 1 and 20, possibly ordered.
Garage	The garage, if any, among “Collective garage”, “None”, “Private garage”.
ClaimInd	Claim indicator of the guarantee. (this is not the claim number)

Feature formatting

The following changes has been made to the data `freMPL1`:

1. The columns `RecordBeg` and `RecordEnd` are discarded.
2. The column `ID` is added to remember policynumbers.
3. The negative `ClaimAmount` entries is overwritten with zeroes and the `ClaimIndicator` is changed accordingly.
4. The observations `VehEngine` being equal to either `electric` or `GPL` is removed.
5. The lowest two `VehMaxSpeed` are combined.
6. The category `bus` is layed under `sedan` in the column `VehBody`.

These changes may be read in the script below.

```r
## Feature selection and formatting
data("freMPL1",package = "CASdatasets")
```

```
#1: RecordBeg and RecordEnd discarded
freMPL1 <- freMPL1 %>%
  select(-RecordBeg,-RecordEnd)

#2: Add id's
freMPL1[,"ID"] <- 1:dim(freMPL1)[1]

#3: Claim amount and indicator
freMPL1$ClaimAmount[freMPL1$ClaimAmount<0] <- 0
freMPL1$ClaimInd <- ifelse(freMPL1$ClaimAmount>0,1,0)

#4: Electric or GPL vehicals
freMPL1 <- freMPL1 %>%
  filter(!(VehEngine %in% c("electric","GPL")))

#5: Combining max speed levels
levels(freMPL1$VehMaxSpeed)[1:2] <- "1-140 kmh"

#6: Bus set to sedan
levels(freMPL1$VehBody)[levels(freMPL1$VehBody) == "bus"] <- "sedan"
```

Preparing the dataset

We lastly format the data by creating a data frame called `data` which we will henceforth be referring to. The data frame is created with splitting the catagorical variables into sorted numerical variables by ordering on both severity and frequency. For instance the `RiskVar` column are translated via the table:

Table 4: RiskVar transformation

Before	Metrics		After	
RiskVar	Frequency	Severity	RiskVar_Freq	RiskVar_Sev
1	0.259	1751	16	4
2	0.231	1597	6	2
3	0.203	1811	2	5
4	0.283	3259	17	19
5	0.243	1820	9	6
6	0.228	1589	4	1
7	0.244	2170	10	11
8	0.237	2860	7	16
9	0.294	1730	19	3
10	0.257	1854	14	7
11	0.249	2107	12	8
12	0.194	1438	1	0
13	0.230	2560	5	13
14	0.185	2158	0	9
15	0.204	2611	3	14
16	0.258	2480	15	12
17	0.248	2918	11	17

Table 4: RiskVar transformation *(continued)*

Before	Metrics		After	
RiskVar	Frequency	Severity	RiskVar_Freq	RiskVar_Sev
18	0.252	2165	13	10
19	0.241	2619	8	15
20	0.290	3109	18	18

The code is implemented below.

```r
#Create df
df <- freMPL1
#Catagorical variables
cat_variables <- c("Gender","VehAge","MariStat","SocioCateg",
                   "VehUsage","HasKmLimit","VehBody","VehPrice","VehEngine",
                   "VehEnergy","VehMaxSpeed","VehClass","RiskVar","Garage")
for (col in cat_variables) {
  #Get ordering
  ord <- freMPL1 %>%
    #Group by distinct values
    group_by(!!as.name(col)) %>%
    #Calculate frequency and mean claim
    summarise(Frequency = sum(ClaimInd)/sum(Exposure),
              Severity = sum(ClaimAmount)/sum(ClaimAmount > 0)) %>%
    ungroup() %>%
    #Order Frequency
    arrange(Frequency) %>%
    #Insert 0,...,n
    mutate(Frequency = 0:(length(unique(!!as.name(col)))-1)) %>%
    #Order Severity
    arrange(Severity) %>%
    #Insert 0,...,n
    mutate(Severity = 0:(length(unique(!!as.name(col)))-1))
  #Prepare ord for merging
  colnames(ord)[2:3] <- paste0(col, c("_Freq","_Sev"))

  #Merge new columns into df
  df <- df %>%
    merge(., ord, all.x = TRUE)
}
```

Modelling the technical premium

Severity model

We work with the data `df_sev` which is a subset of `df` with the filter `ClaimAmount > 0`.

```r
#Get relevant columns
sev_columns <- colnames(df)[!grepl("_Freq",colnames(df)) &
                              !(colnames(df) %in% cat_variables)]
df_sev <- df[,sev_columns] %>%
  filter(ClaimInd == 1) %>%
  select(-ClaimInd,-Exposure)
#Save ID in row names
row.names(df_sev) <- df_sev$ID
df_sev <- df_sev %>%
  select(-ID)%>%
  mutate_if(is.integer, as.numeric)
```

```r
#Start a task
task_sev <- df_sev %>%
  mutate(ClaimAmount = log(ClaimAmount)) %>%
  #Start tast with target ClaimAmound
  as_task_regr(.,
               target = "ClaimAmount",
               id= "Severity")

### XGBoost
sev_xgb_learner <- lrn("regr.xgboost",
                   eta = to_tune(0, 0.5),
                   nrounds = to_tune(75, 5000),
                   max_depth = to_tune(1, 3))

### XGBoost
set.seed(20230328) #We choose a seed
# 1: Estimating hyperparameters with 5-fold cross validation
sev_xgb_learner_instance = tune(
  #method = tnr("random_search"), ### tuning method
  method = mlr3tuning::tnr("mbo"), ### tuning method
  task = task_sev,
  learner = sev_xgb_learner,
  resampling = rsmp("cv", folds = 5), #### resampling method: 5-fold cross validation
  measures = msr("regr.rmse"), #### Mean Squared Log Error
  terminator = trm("evals", n_evals = 200) #### terminator
)
#Save the instance
saveRDS(sev_xgb_learner_instance, file = "rds/sev_xgb_learner_instance.rds")
```

Estimating hyperparameters

Hyperparamater	Search space	Estimate
XGBoost		
`eta`	$[0, 0.5]$	0.089492
`nrounds`	$[75, 5000]$	4925

Hyperparamater	Search space	Estimate
max_depth	$[1, 3]$	3

```r
#Partition for training
indicies <- function(size,ratio){
  train <- sort(sample(1:size,ceiling(size*ratio),replace = FALSE))
  test <- (1:size)[!(1:size %in% train)]
  return(list(train = train, test = test))
}
set.seed(20230328) #We choose a seed
splits <- indicies(dim(df_sev)[1],0.85)
#Split covariates and truth
regr_data_train <- list(
  covariates = as.matrix(df_sev[splits$train,] %>% select(-ClaimAmount)),
  truth = log(as.numeric(df_sev$ClaimAmount[splits$train]))
)
regr_data_test <- list(
  covariates = as.matrix(df_sev[splits$test,] %>% select(-ClaimAmount)),
  truth = log(as.numeric(df_sev$ClaimAmount[splits$test]))
)
#Fit model using hyperparameters
xgb_regr_train <- xgboost(
  data = regr_data_train$covariates,
  label = regr_data_train$truth,
  eta = sev_xgb_learner_instance$result_learner_param_vals$eta,
  nrounds = sev_xgb_learner_instance$result_learner_param_vals$nrounds,
  max_depth = sev_xgb_learner_instance$result_learner_param_vals$max_depth
)
#Save model
saveRDS(xgb_regr_train, file = "rds/xgb_regr_train.rds")
```

```r
#Predictions vs. truth
pred <- predict(xgb_regr_train, newdata = regr_data_test$covariates)
truth <- regr_data_test$truth
mean(exp(pred))
```

Fitting the model

```
## [1] 2015.188
```

```r
mean(exp(truth))
```

```
## [1] 2438.129
```

```r
#Mean Squared Log Error
mean((pred - truth)^2)
```

```
## [1] 1.362892
```

```r
#Root MSE
sqrt(mean((exp(pred) - exp(truth))^2))
```

```
## [1] 3065.317
```

```r
#Split covariates and truth
regr_data <- list(
  covariates = as.matrix(df_sev %>% select(-ClaimAmount)),
  truth = log(as.numeric(df_sev$ClaimAmount))
)
#Fit model using hyperparameters
xgb_regr <- xgboost(
  data = regr_data$covariates,
  label = regr_data$truth,
  eta = sev_xgb_learner_instance$result_learner_param_vals$eta,
  nrounds = sev_xgb_learner_instance$result_learner_param_vals$nrounds,
  max_depth = sev_xgb_learner_instance$result_learner_param_vals$max_depth
)
#Save model
saveRDS(xgb_regr, file = "rds/xgb_regr.rds")
```

Frequency model

We work with the data `df_freq` which is a copy of
`df` with the transformed columns `._Freq`.

```r
#Get relevant columns
freq_columns <- colnames(df)[!grepl("_Sev",colnames(df)) &
                              !(colnames(df) %in% cat_variables)]
df_freq <- df[,freq_columns] %>%
  select(-ClaimAmount) %>%
  mutate(ClaimInd = factor(ClaimInd, levels = c(0,1)))
#Save ID in row names
row.names(df_freq) <- df_freq$ID
df_freq <- df_freq %>%
  select(-ID) %>%
  mutate_if(is.integer, as.numeric)
```

```r
#Start a task
task_freq <- df_freq %>%
  #Start tast with target ClaimInd
  as_task_classif(.,
                target = "ClaimInd",
                id= "Frequency")
#XGB
freq_xgb_learner <- lrn("classif.xgboost",
                        eta = to_tune(0, 0.5),
                        nrounds = to_tune(75, 5000),
                        max_depth = to_tune(1, 3),
                        predict_type = "prob")
freq_xgb_learner_instance = tune(
    #method = tnr("random_search"), ### tuning method
    method = mlr3tuning::tnr("mbo"), ### tuning method
    task = task_freq,
    learner = freq_xgb_learner,
    resampling = rsmp("cv", folds = 5), #### resampling method: 5-fold cross validation
    measures = msr("classif.logloss"), #### root mean squared error
    terminator = trm("evals", n_evals = 200) #### terminator
  )
saveRDS(freq_xgb_learner_instance, file = "rds/freq_xgb_learner_instance.rds")
```

Estimating hyperparameters

Hyperparamater	Search space	Estimate
XGBoost		
eta	$[0, 0.5]$	0.172823
nrounds	$[75, 5000]$	2085
max_depth	$[1, 3]$	3

```r
#Partition for training
set.seed(20230328) #We choose a seed
splits <- indicies(dim(df_freq)[1],0.85)
#Split covariates and truth
```

```r
freq_data_train <- list(
  covariates = as.matrix(df_freq[splits$train,] %>% select(-ClaimInd)),
  truth = as.numeric(as.character(df_freq$ClaimInd[splits$train]))
)
freq_data_test <- list(
  covariates = as.matrix(df_freq[splits$test,] %>% select(-ClaimInd)),
  truth = as.numeric(as.character(df_freq$ClaimInd[splits$test]))
)
#Fit model using hyperparameters
xgb_freq_train <- xgboost(
  data = freq_data_train$covariates,
  label = freq_data_train$truth,
  eta = freq_xgb_learner_instance$result_learner_param_vals$eta,
  nrounds = freq_xgb_learner_instance$result_learner_param_vals$nrounds,
  max_depth = freq_xgb_learner_instance$result_learner_param_vals$max_depth,
  objective="count:poisson"
)
#Save model
saveRDS(xgb_freq_train, file = "rds/xgb_freq_train.rds")
```

```r

#Predictions vs. truth
pred <- predict(xgb_freq_train, newdata = freq_data_test$covariates)
truth <- freq_data_test$truth
mean(pred)
```

Fitting the model

```
## [1] 0.1054211
```

```r
mean(truth)
```

```
## [1] 0.1017873
```

```r
#log loss
ll_loss <- function(y,p) {
  p <- ifelse(p == 1, 1-0.0000001,p)
  p <- ifelse(p==0, 0.0000001, p)
  ll <- -( y * log(p) + (1 - y)*log(1-p))
  ll <- ifelse(is.na(ll),0,ll)
  return(ll)
}
mean(ll_loss(truth,pred))
```

```
## [1] 0.2779527
```

```r
#Split covariates and truth
freq_data <- list(
  covariates = as.matrix(df_freq %>% select(-ClaimInd)),
  truth = as.numeric(as.character(df_freq$ClaimInd))
)
#Fit model using hyperparameters
xgb_freq <- xgboost(
  data = freq_data$covariates,
  label = freq_data$truth,
  eta = freq_xgb_learner_instance$result_learner_param_vals$eta,
  nrounds = freq_xgb_learner_instance$result_learner_param_vals$nrounds,
  max_depth = freq_xgb_learner_instance$result_learner_param_vals$max_depth,
  objective="count:poisson"
)
#Save model
saveRDS(xgb_freq, file = "rds/xgb_freq.rds")
```

Estimating the biased technical premium

```r
# Estimating mu
sev_covariates <- df[,sev_columns] %>%
  select(-ClaimInd,-ID,-ClaimAmount,-Exposure)
sev_covariates <-  as.matrix(sev_covariates)
m_mu_ranger <- exp(predict(xgb_regr, newdata = sev_covariates))
# Estimating p
freq_covariates <- df[,freq_columns] %>%
  select(-ClaimInd,-ID,-ClaimAmount)
freq_covariates <- as.matrix(freq_covariates)
m_p_ranger <- predict(xgb_freq_train, newdata = freq_covariates)
freq_covariates[,"Exposure"] <- 1
m_p_ranger_2 <- predict(xgb_freq_train, newdata = freq_covariates)
# Inserting estimates
df[,"pi_hat"] <- m_mu_ranger * m_p_ranger
df[,"mu_hat"] <- m_mu_ranger
df[,"p_hat"] <- m_p_ranger
df[,"pi_hat_new"] <- m_mu_ranger * m_p_ranger_2
```