# Assignment 1

Jakob og Oskar

2025

## Assignment 1 in interpretable machine learning

We are tasked with building a predictive regression model, with the best possible prediction. This submission will be split up into several parts:

- Initial data preprocessing and overview
- Introduction into the mathematics
- Modelling and justification
- Comparative discussion

### Initial data preprocessing and overview

The given data has the form:

```
##  [1] "ID"                  "Date_start_contract"  "Date_last_renewal"
##  [4] "Date_next_renewal"   "Date_birth"           "Date_driving_licence"
##  [7] "Distribution_channel" "Seniority"           "Policies_in_force"
## [10] "Max_policies"        "Max_products"         "Lapse"
## [13] "Date_lapse"          "Payment"              "Premium"
## [16] "Cost_claims_year"    "N_claims_year"        "N_claims_history"
## [19] "R_Claims_history"    "Type_risk"            "Area"
## [22] "Second_driver"       "Year_matriculation"   "Power"
## [25] "Cylinder_capacity"   "Value_vehicle"        "N_doors"
## [28] "Type_fuel"           "Length"               "Weight"
```

We are asked to predict the **Cost_claims_year** given the rest of the covariate-vector. Initially it is important to note that out data is a classical insurance dataset, where we are given rows corresponding to insurance periode for a given contract. There are several issues with this, since some contracts might overlap into multiple contracts, which can be identified by the ID. It is however very difficult to locate these policies, and we overlook this issue.

There are numerous char. vectors in the data, which can be seen here:

```
## Classes 'data.table' and 'data.frame':   105555 obs. of  30 variables:
##  $ ID                  : int  1 1 1 1 2 2 3 3 3 3 ...
##  $ Date_start_contract : chr  "05/11/2015" "05/11/2015" "05/11/2015" "05/11/2015" ...
##  $ Date_last_renewal   : chr  "05/11/2015" "05/11/2016" "05/11/2017" "05/11/2018" ...
##  $ Date_next_renewal   : chr  "05/11/2016" "05/11/2017" "05/11/2018" "05/11/2019" ...
##  $ Date_birth          : chr  "15/04/1956" "15/04/1956" "15/04/1956" "15/04/1956" ...
##  $ Date_driving_licence: chr  "20/03/1976" "20/03/1976" "20/03/1976" "20/03/1976" ...
```

```
##  $ Distribution_channel: chr  "0" "0" "0" "0" ...
##  $ Seniority           : int  4 4 4 4 4 4 15 15 15 15 ...
##  $ Policies_in_force   : int  1 1 2 2 2 2 1 1 1 1 ...
##  $ Max_policies        : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Max_products        : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Lapse               : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Date_lapse          : chr  "" "" "" "" ...
##  $ Payment             : int  0 0 0 0 1 1 0 0 0 0 ...
##  $ Premium             : num  223 214 215 217 214 ...
##  $ Cost_claims_year    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ N_claims_year       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ N_claims_history    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ R_Claims_history    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Type_risk           : int  1 1 1 1 1 1 3 3 3 3 ...
##  $ Area                : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Second_driver       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Year_matriculation  : int  2004 2004 2004 2004 2004 2004 2013 2013 2013 2013 ...
##  $ Power               : int  80 80 80 80 80 80 85 85 85 85 ...
##  $ Cylinder_capacity   : int  599 599 599 599 599 599 1229 1229 1229 1229 ...
##  $ Value_vehicle       : num  7068 7068 7068 7068 7068 ...
##  $ N_doors             : int  0 0 0 0 0 0 5 5 5 5 ...
##  $ Type_fuel           : chr  "P" "P" "P" "P" ...
##  $ Length              : num  NA NA NA NA NA ...
##  $ Weight              : int  190 190 190 190 190 190 1105 1105 1105 1105 ...
##  - attr(*, ".internal.selfref")=<externalptr>
## NULL
```

One could, model some of the char. vectors like proposed in the lectures, by

$$\begin{aligned} X_i &= E(Y \mid X_i) \\ &= \int_Y y \; \mu(x, dy) \end{aligned}$$

where $\mu$ is the appropriate probability kernel, and we let $y$ be our response. However, we choose to take the our char. vectors and round them to yearly values, which then has a ordinal ordering and can thus be used as features. Further we take and one-hot encode *Distribution_channel* by creating three new features which are either 1 or 0. Same for the type of fuel.

Next there are some missing values.

It becomes apparent that there are missing values in the length and type_fuel variable. We can see that the missingness is overlapping in 1764 rows. However the length variable suffers way heavier from missingnes compared to type_fuel. In order to impute values, we assume the missingnes it completely at random for both features. We consider correlated features to do imputation. We see from the correlation plot (fig. 3 ) that type_fuel is mostly correlated with Cylinder_capacity, Value_vehicle and Weight. The same goes for the feature Length. Therefore, we fit a multivariate linear regression model with these 3 covariates to predict both type_fuel and Length (using cbind in the response formula). Finally, we predict the missing values using this trained model.

Our respone variable **Cost_claims_year** suffers from a few extreme values. We decide to remove these values to later on acheive a better model fit. In fig. 2, **N_claims_year** is plotted against **Cost_claims_year**, where at least 5-10 extreme values of **Cost_claims_year* are spotted.

Finally we look at the correlation between the covariates.

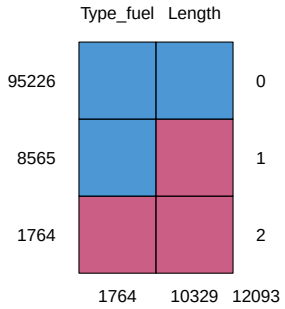


Figure 1: Missing data plot, right axis shows numer of missing columns in that row, and the left axis show how many rows have this missingness pattern

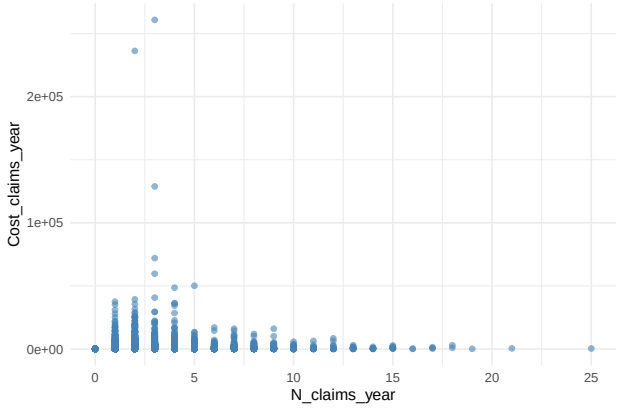


Figure 2: Claim costs over number of claims

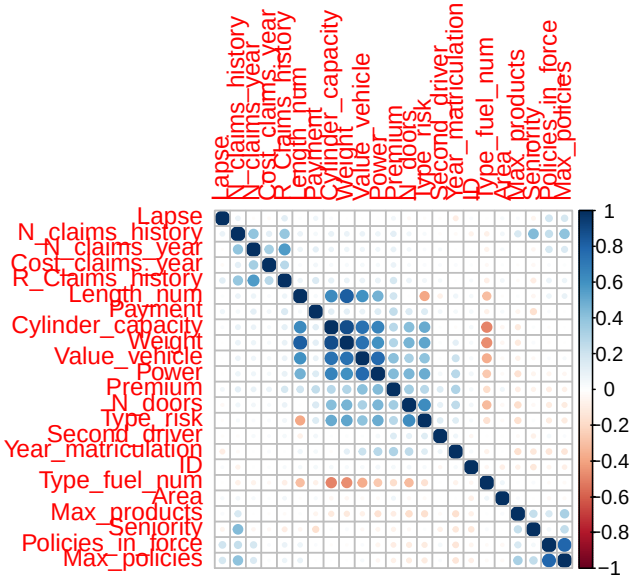


Figure 3: Correlation plot between the continious covariates

We notice some clusters, most meaningfull between *Cylinder_capacity*, *Value_vehicle* and *Weight* which is expected. We deem these to have significant predictive ability, and thus we choose to not remove these. The top left cluster, will be ignored for now, since we will later introduce a data-transformation which affects this cluster in a high degree.

**Data Aggregation** For our data aggregation we want to have the ability to assume independence, which we have if we aggregate the rows into being one policy. There are several problems to tacle to achive data in the desired format. We have to both choose an appropriate aggreations function, and we have to consider events where the insured object change. We start off by identifying some unique-identifiers. If theme columns change, we deem the contract to insure a new car, or at least insure something that is independent of what was previously insured. Amongst these unique identifiers are Length, Weight, Power etc. so things which change, probably mean that the thing that is insured has changes. \ Next, we look at how we have aggregated data. Firstly we sum the exposure for each contract, since we will use this to weight in the models later on. Next we use the max function on the number of policies, lapse, date of birth, date of driving licence, start of contract, products, payments, ratio of claims history, type of risk,. For The mean premium we use the mean of the premium without the forward premium, that is the premium for the time we want to predict for. For the value of the veichle, and number of doors we use mean. For the length we use sum since this is just a scaling. For the variable *cost_claims_this_year* we use the cost of the claims in $\mathcal{F}_t$ where we are to predict data on $\mathcal{F}_{t+1}$.

So we have $Z_i(\omega_i) := (X_i(\omega_i), Y_i(\omega_i)) : (\Omega_i, \mathcal{F}_i) \to (\mathcal{X} \times \mathcal{Y}, \mathcal{B}(\mathcal{X} \times \mathcal{Y}))$, where by aggregating data row-wise we obtain independence on every set in the Borel-set induced. However this is an empirical assumption which can be violated by catastrophe events.

## Introduction into the mathematical framework

Note the classical derivation of how to decompose a total claim sum.

$$\begin{aligned} E(S(t) \mid Z = z) &= E\left(\sum_{i=1}^{N(t)} X_i \mid Z = z\right) \\ &= E\left(\left(\sum_{i=1}^{N(t)} X_i \mid Z = z \cap \mathcal{F}(t)\right)\right) \\ &= E\left(NE\left(X_i \mid Z = z\right) \mid Z = z\right) \\ &= E(N \mid Z = z)E(X_1 \mid Z = z) \end{aligned}$$

We would however like to modify this slightly, later on. Further the general definition of the Tweedie law is

$$P_{\theta,\sigma^2}(Y \in A) = \int_A \exp\left\{\frac{\theta z - \kappa_p(\theta)}{\sigma^2}\right\} \nu_y(dz)$$

Where $A \subset \mathbb{R}$ and measurable. $\theta$ is the canonical parameter, and $\kappa_p(\theta)$ is the cumulant function dependent on $p$ the so called tweedie power parameter. $\sigma^2$ is the dispersion parameters, and $\nu_\lambda$ is some sigma-finite base measure depending on the Lebesgue measure.

since it can accommodate a zero-inflated distribution very well, we feel the need to introduce this model here. Lastly we introduce the so-called *bbrier* measure, for classification models.

$$\frac{1}{n}\sum_{i=1}^{n} w_i(1\{X_i = 1\} - P(X_i = 1))^2$$

Which is the mean weighted square euclidean distance between the probability and the true value.

**Parameter search space**

| Model | tweedie_power | learning_rate_eta | leaves | iterations | s | alpha |
|---|---|---|---|---|---|---|
| Xgboost | 1-1.9 | 1e-3 - 0.2 | 16 - 32 | 200 - 1000 | | |
| lgbm | 1-1.9 | 1e-3 - 0.2 | 3 - 9 | 200 - 1000 | | |
| glmnet | | | | | 1e-4 - 1 | 0 - 1 |

## Modelling and justification

In this section, we train and evauluate different models. For all the proposed models, we have caried out nested crossvalidition. We do stratified resampling in both the inner and outer cv loops, to ensure both zero and non-zero claims over the training and test set. I final note, that goes for all our trained models, is that we introduce offset on the exposure. Meaning: Give more importance to observations with more exposure since they represent more information.

**Tweedie** Initily we want to fit a Tweedie model on the entire data, since the Tweedie model is known for handeling zero-inflated distributions well.

We apply both the Xgboost, Ligth-gbm and glmnet. For each of these we supply search-space information in the **Parameter search space** Table.

We define the workflow as shown in 4.

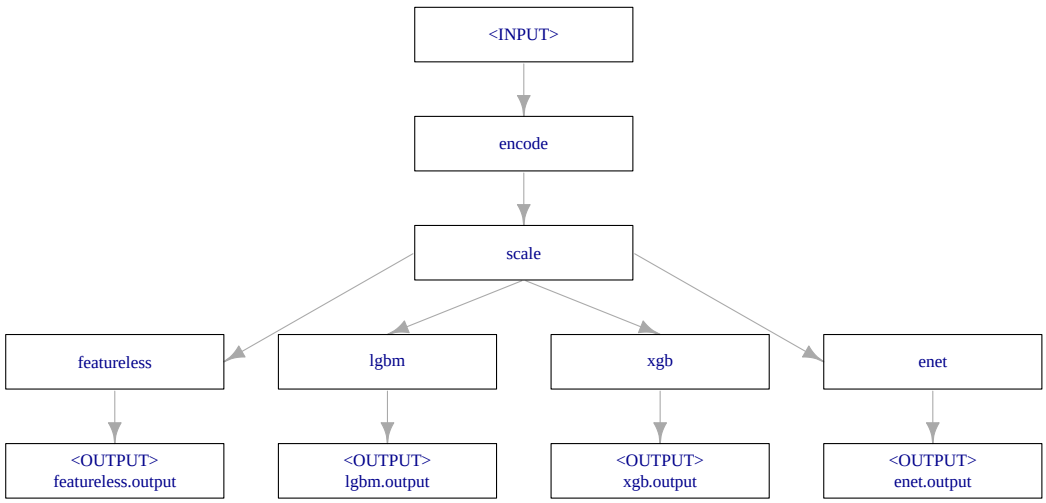


Figure 4: Input and model flow

The results of the nested cross-validation are shown in 5.

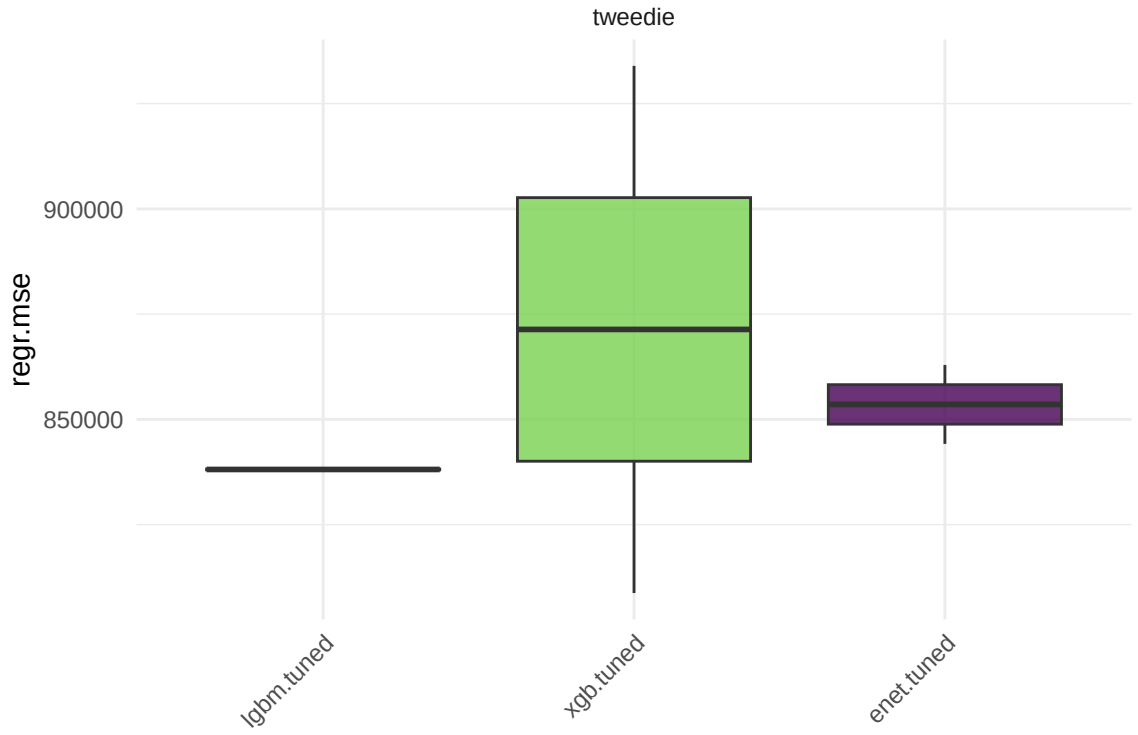


Figure 5: MSE boxplot

And finally the chosen hyperparameters are presented in Table **Best parameters tweedie**.

vi skal skrive noget mere til ovenstående, når resultater er på plads!

**Split model** Next we can look at the split model proposed in the mathematical frame, where we aim to create two models, and then predict in the following manner:

$$P(N > 0 \mid Z) \cdot E(X \mid Z)$$

Which is unlike the traditional and proposed $E(N)E(X)$, but since we are asked to predict the next year, we feel like we can provide a suitable approximation with the above prediction. This entails fitting both a frequency model and a severity model. Hence we begin with the frequency regression model.

We have fitted the model with nested cross-validation since it provides a more stable estimate of the generalisation error. We use $K = 2$ folds in both the inner and outer loops. Next we look at some model diagnostics. We can look at the combined *bbrier* score for the aggregated model:

```
## classif.bbrier
##      0.1108596
```

which seems quite small. Further we have some confusion matrices for each of the folds.

```
## Fold:  1        truth
## response     1     0
##        1  5392  1069
##        0  2429 15369
```

**Best parameters tweedie**

| parameter | value |
|---|---|
| encode.method | one-hot |
| scale.robust | FALSE |
| regr.lightgbm.objective | tweedie |
| regr.lightgbm.verbose | -1 |
| regr.lightgbm.learning_rate | list(lower = 0.001, upper = 0.2, logscale = TRUE), to_tune(0.0 |
| regr.lightgbm.num_leaves | list(lower = 16, upper = 32, logscale = FALSE), to_ |
| regr.lightgbm.num_threads | 1 |
| regr.lightgbm.tweedie_variance_power | list(lower = 1, upper = 1.9, logscale = FALSE), to |
| regr.lightgbm.num_iterations | list(lower = 200, upper = 1000, logscale = FALSE), to_ |

```
## acc :  0.8558; ce  :  0.1442; dor :  31.9147; f1  :  0.7551
## fdr :  0.1655; fnr :  0.3106; fomr:  0.1365; fpr :  0.0650
## mcc :  0.6602; npv :  0.8635; ppv :  0.8345; tnr :  0.9350
## tpr :  0.6894
## Fold:  2        truth
## response     1     0
##        1  5366  1143
##        0  2454 15294
## acc :  0.8517; ce  :  0.1483; dor :  29.2584; f1  :  0.7490
## fdr :  0.1756; fnr :  0.3138; fomr:  0.1383; fpr :  0.0695
## mcc :  0.6505; npv :  0.8617; ppv :  0.8244; tnr :  0.9305
## tpr :  0.6862
```