Data preprocessing and fire insurance estimation:

Taking it a step further with machine learning

Presentation made by
Luca Morarevic, Victoria Kledal, Nicklas B. Christensen & Emma Juul

Presentation of data

19 variables

~ 993 000 observations from 2019-2024

~ 2% missing data

Setup: Fire insurance data from 2019-2024 with roughly two categories of variables:

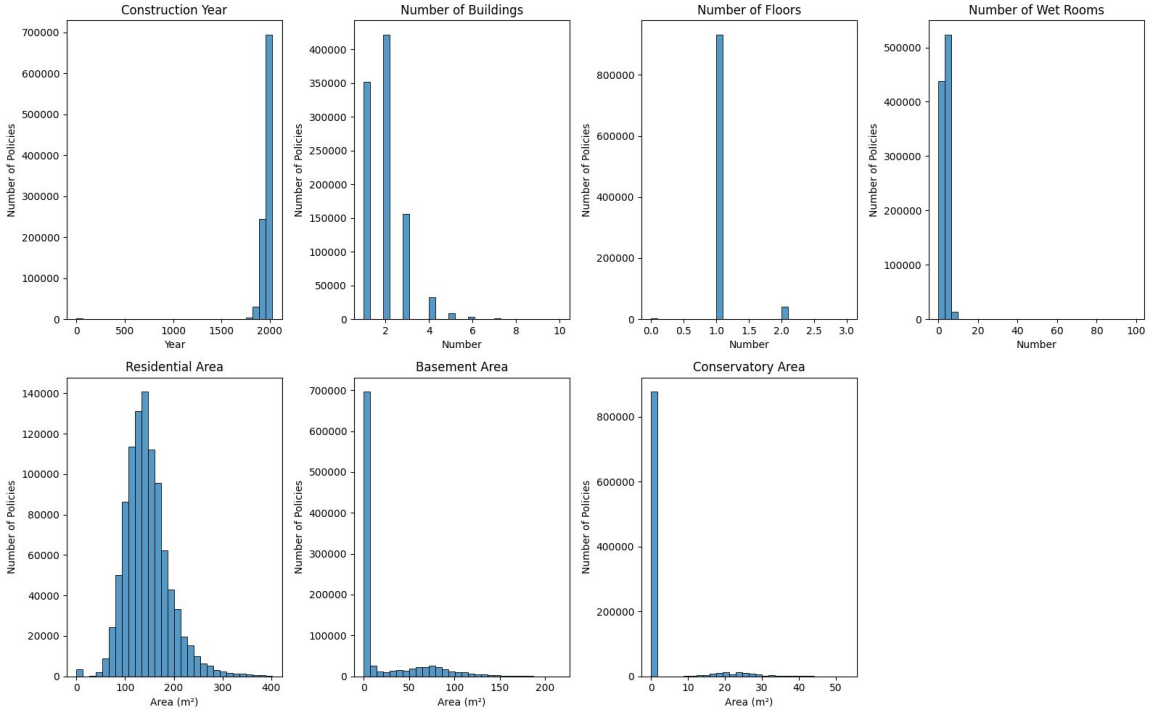
- Variables describing **the policy**: claim count, size, deductible, etc.
- Variables describing **the home**: m^2, construction year, roof type, etc.

Data preprocessing

Anomaly detection

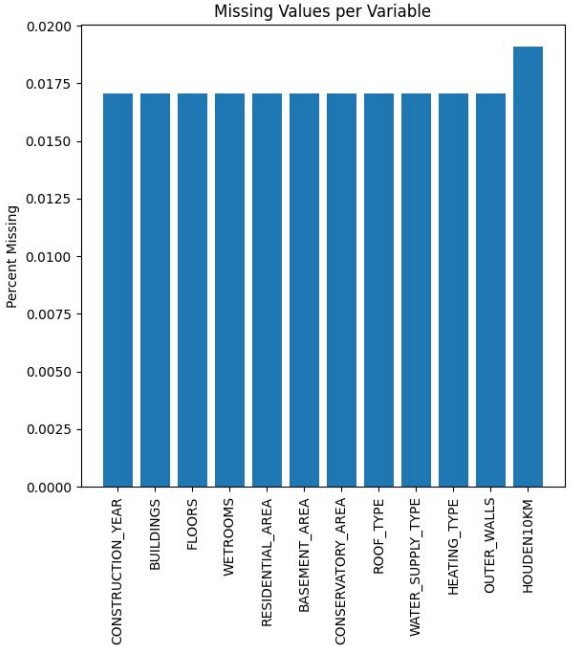
Method used:

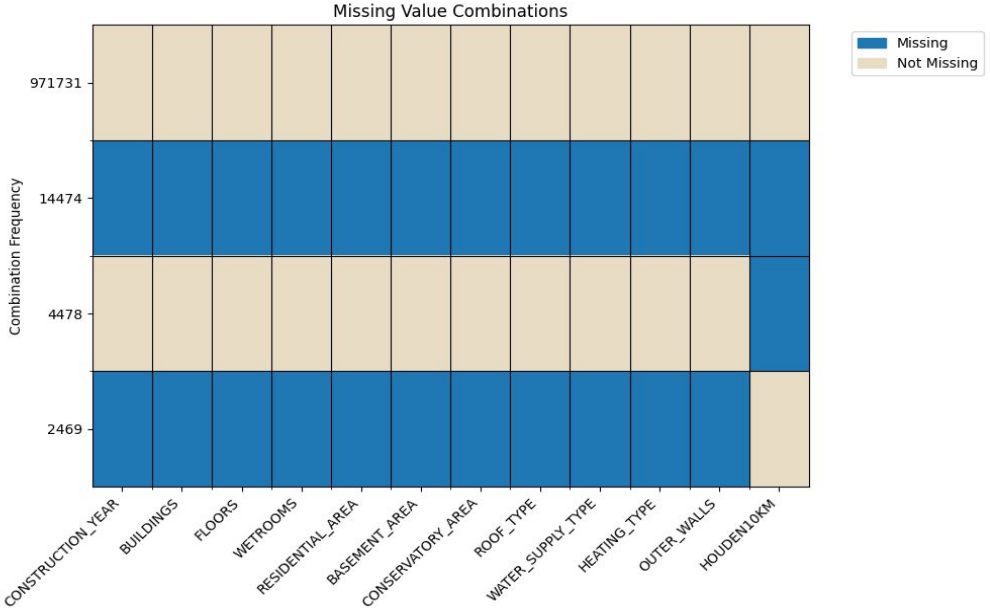
- Plotted the distribution of each variable
- Determined which plots show outliers
- Used the algorithm “Isolation forest” on each of these variables to detect outliers
    - Only sort-of worked
- Changed the values that were “appropriately” flagged by algorithm to impute later on
- Example: 99 wetrooms (Buckingham Palace has 78 in comparison)



Missing values

- Looking at variables describing **the home**
- Removed the combinations where all variables or all but HOUDEN10KM were missing since it would be “double predicting” otherwise (removes only 1.7% of data)





Imputing missing values

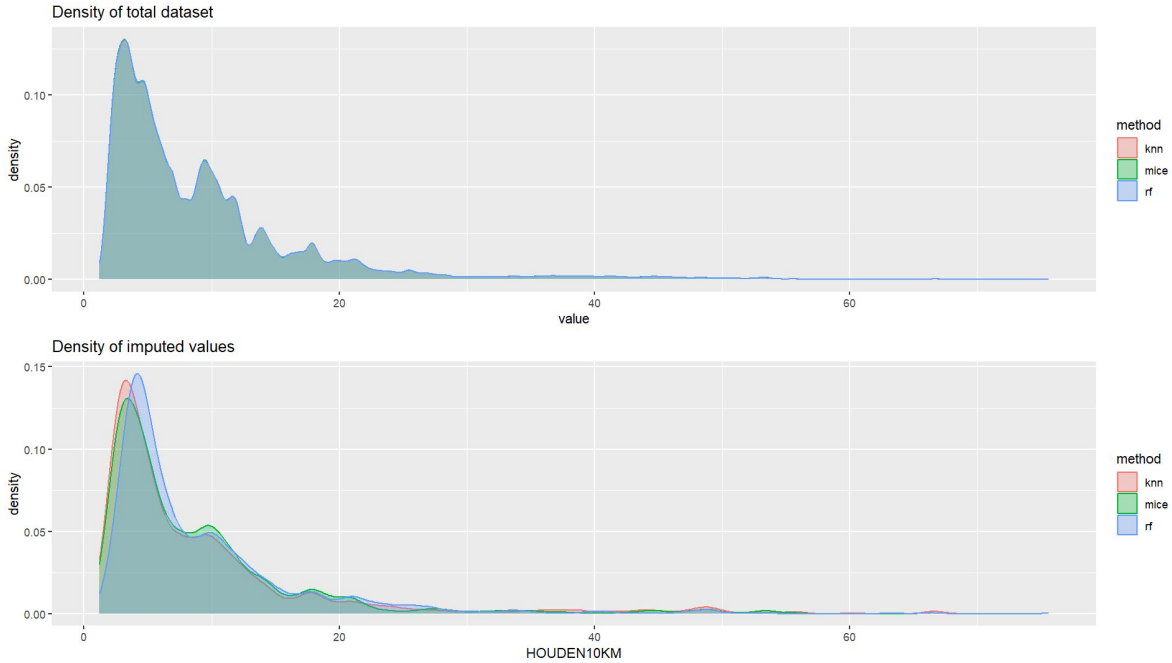
Used a total of three algorithms:

- MICE
- Random Forest
- k-Nearest Neighbors

Assessment of algorithms:

- Huge difference in computation time
  - Minutes, hours and days
- Almost same result

➔ Final algorithm: MICE



One hot encoding of categorical variables

Motivation: Counteracting the possible assumption of a hierarchy of values within a variable when there is none

Used on attribute-like variables:

- OUTER_WALLS
- HEATING_TYPE
- WATER_SUPPLY_TYPE
- ROOF_TYPE

Number of variables: 19 → 58

Roof type	Water supply type	Heating type	Outer walls type
Flat roof	Public water supply	District heating	Brick walls
Tar paper	Private water supply	Central heating own system	Lightweight concrete
Fiber cement asbestos	Water extraction plant	Stove fireplace	Fiber cement plates
Concrete tiles	Well water	Heat pump	Timber framing
Brick	Non-public water supply	Central heating two units	Wood cladding
Metal sheets	Mixed water supply	Electric heating	Concrete elements
Thatched roof	No water supply	Gas radiators	Metal plates
Fiber cement		No heating	Fiber cement
PVC		Mixed	Glass
Glass		Unknown	Other material Unknown
Other material			
Unknown			

Modeling aggregated claim size

The classical actuarial model - simplified

- ➔ Assume the aggregated claim size can be decomposed into the a **claim count** and an individual **claim size**.
- ➔ We then model the two separately and combine them after.
- ➔ Usual practice: Handle claims over a large loss threshold separately
- ➔ Exposure variable: Used in modelling claim count as a “scaler”

Predicting claim size

Feature ranking

Filtered data to only contain claims

↓

Trained models using 5 fold CV, LGBMRegressor and a grid of HPs

↓

Ranked features by aggregating score from: split, gain, SHAP and permutation importance

↓

Checked for high correlation among highest ranking features

Model selection

Selected four seemingly reasonable feature sets based primarily on ranking, correlation and variable type

↓

Tried 3 tree-based models and MLPRegressor (NN) across seeds and a coarse HP grid (raw and log-transformed target)

↓

MLP performed the best: Fine-tuned HPs using Random Search, finer grid search and Optuna

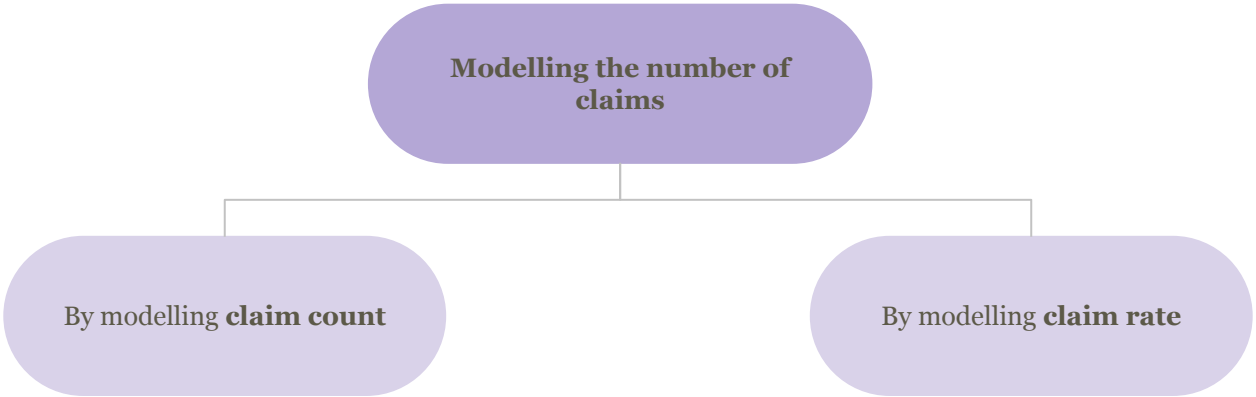
↓

Ran validation on final parameters before committing to the model

One model used a loss fct. that further penalizes bad predictions for large losses

Predicting claim rate and claim count

Our general setup



Implicit assumption that we are making: linear relationship between **exposure** and the **probability of a claim**. Lets test this!

Feature ranking and model selection was very similar to that of claim size regression

Predicting a yearly claim size

Our final models

Claim count model

CatBoostRegressor

Depth: 6

l2 leaf reg: 14

Learning rate: 0.032

Claim size model with penalty

MLPRegressor

Hidden layers: (64, 64)

Alpha: $8.9*10^{-4}$

Learning rate: $6.3*10^{-4}$

Claim rate model

MLPRegressor

Hidden layers: (128, 128)

Alpha: 0.0010

Learning rate: 0.0005

Claim size model without penalty

CatBoostRegressor

Depth: 5

l2 leaf reg: 20

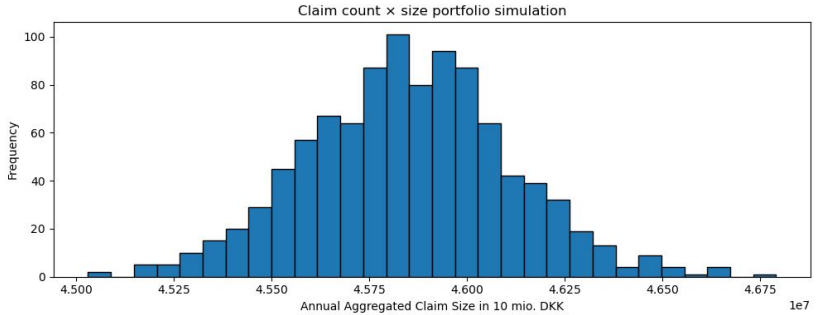
Learning rate: 0.01

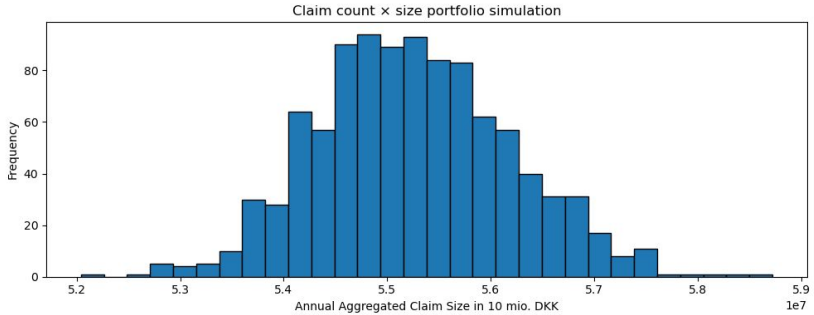
Simulating a yearly claim size

The described method was performed on both claim count and rate. Claim size was used in both cases.

Simulated “annual population” from data using bootstrap

- Predicted claim size and claim count/rate for each sampled policy and multiplied the two predictions
- Summed all prediction to get an aggregated yearly claim size prediction.
- Repeated 1.000 times
- Mean of observed yearly claim size: 45,8 mio. DKK





Joint model for risk prediction

Our general problem

Claim size data naturally have two defining characteristics.

- Lots of zeroes.
- A tendency for extreme values.

This combination restricts the choice of ‘normal’ models to choose from. Hence machine learning.

Nonetheless the above challenges still apply. We however have two ways to deal with this:

- Tinkering with loss functions.
- Transformations of data.

Loss functions and data transformation

Why not just guess zero? The trouble of Pareto's principle.

- Loss functions in $\mathcal{L}^p$ -spaces; especially relative.
- The Tweedie loss function (p in (1,2)).
- Adding a penalty term to the loss function if predicted less. actual values were negative.

$$\mathcal{L}_{rel,p} = \sum_{i=1}^{n} \left(\frac{|y_i - \hat{y}_i|}{|y_i| + \epsilon} \right)^p$$

Transformations of data gives:

$$\mathcal{L}_{Tweedie} = -\sum_{i=1}^{n} x_i \frac{\tilde{x}_i^{1-p}}{1-p} + \frac{\tilde{x}_i^{2-p}}{2-p}$$

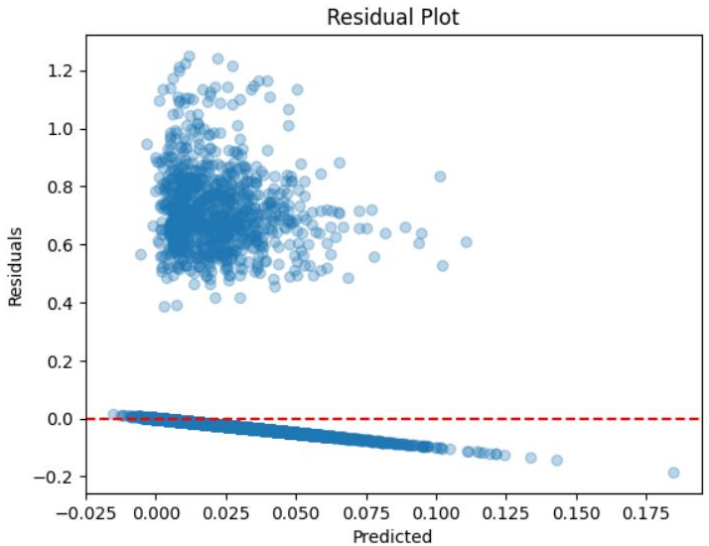
- At least somewhat 'regular' data to work with.
- Helps justifying the use of the Tweedie loss function.

Results

Bad. Genuinely very bad. Both with XGBoost and PyCharm type models for several iterations of model/loss functions/transformations of variables.

By far the best model was XGBoost with a Tweedie loss on data with with the shown transform and without penalty. This, however, will not flow in a business.

$$t(x_i) = \frac{\frac{log(x_i+1)}{sd(log(x))}}{10}$$



Appendix

Github

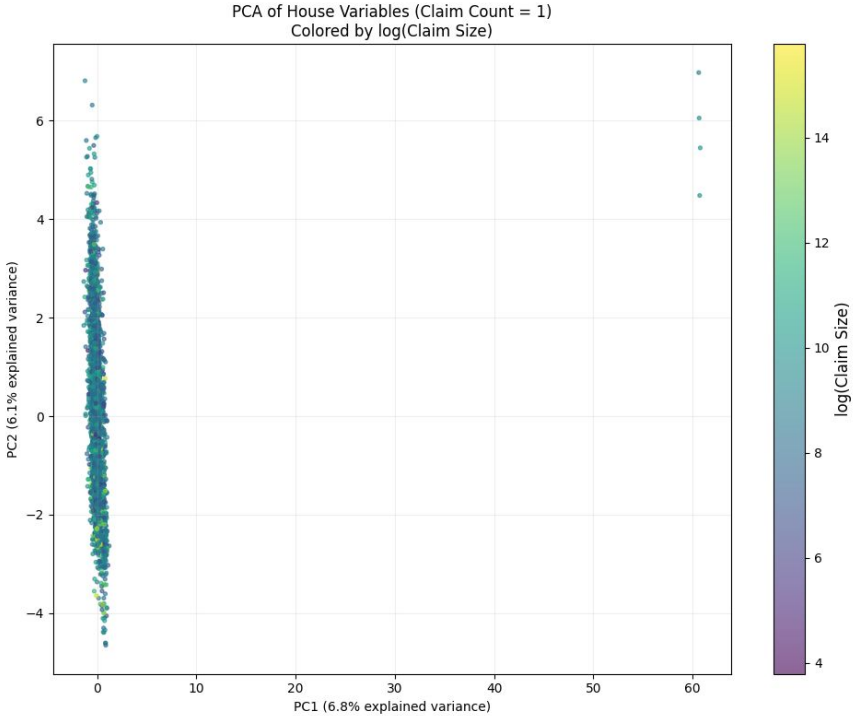
See all our code in our Github folder. Link:

https://github.com/lucmor65/AML-Final-Project-2026/tree/main/Code

Clustering with PCA colored by claim size

We tried to use clustering with PCA on the house variable colored by their respective claimsize to see if any specific combinations of the house variables had a tendency to have a higher claim size.

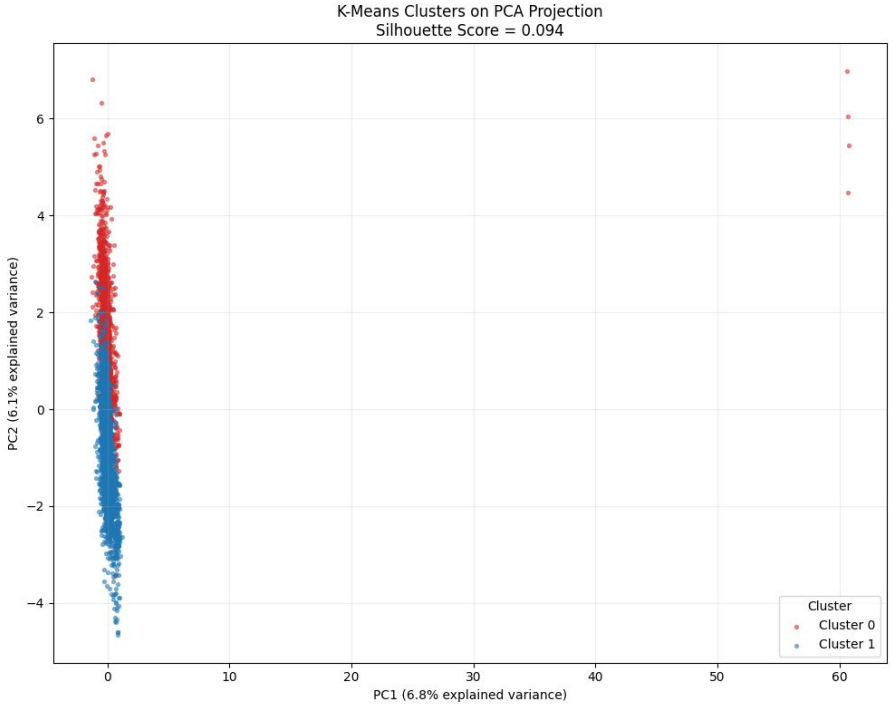
The PCA of the house-related variables showed no clear clustering structure. The first two principal components explained only 12.9% of the total variance (PC1: 6.8%, PC2: 6.1%), indicating that the underlying variation is distributed across many dimensions. Claim sizes, represented by color, were dispersed throughout the PCA space without any obvious pattern, suggesting that house characteristics alone have limited ability to distinguish between small and large claims.



Clustering with K-means and PCA

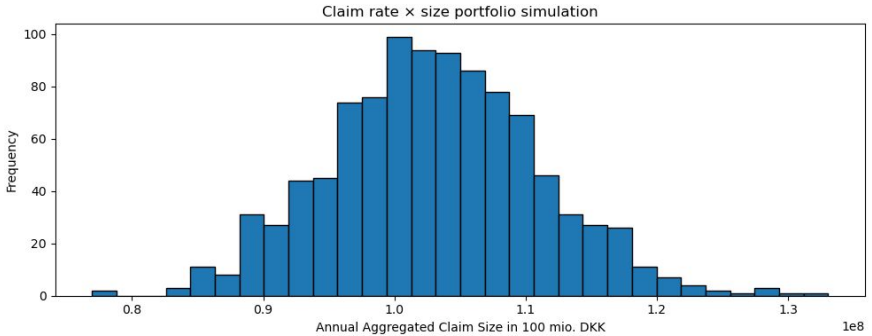
K-means clustering identified two groups (because we asked it to), but the clusters showed substantial overlap and a very low silhouette score (0.094), indicating poor cluster separation.

Together with the PCA results, this suggests that the house-related variables do not form distinct natural clusters.



Simulating a yearly claim size - a failed example

The plot on the left is a result of trying to simulate a yearly claim size using the **claim size model (with penalty) together with the claim rate model**. It seems by the large overestimation of the yearly claim size (which was observed to be 45,8 mio. DKK), that the two models do not work very well together. The purpose of the claim size model (with penalty) was to simulate well with the claim count model and not the claim rate model. By the simulation it seems that the higher penalty for wrongly predicting large losses overly contributes to the yearly claim size, when used in combination with the claim rate model, thus making the two models a bad pair.



Feature selections

Claim count model

'EXPOSURE', 'DEDUCTIBLE', 'CONSTRUCTION_YEAR', 'HEATING_TYPE_CAT_District_Heating', 'RESIDENTIAL_AREA', 'HOUDEN10KM', 'WATER_SUPPLY_TYPE_CAT_Water_Extraction_Plant', 'HEATING_TYPE_CAT_Heat_Pump', 'BUILDINGS', 'BASEMENT_AREA', 'HEATING_TYPE_CAT_Central_Heating_Two_Units', 'HEATING_TYPE_CAT_Stove_Fireplace', 'CONSERVATORY_AREA', 'WATER_SUPPLY_TYPE_CAT_NonPublic_WaterSupply', 'ROOF_TYPE_CAT_Tar_Paper', 'WETROOMS', 'ROOF_TYPE_CAT_Concrete_Tiles', 'ROOF_TYPE_CAT_Fiber_Cement_Asbestos', 'WATER_SUPPLY_TYPE_CAT_Mixed_Water_Supply', 'WATER_SUPPLY_TYPE_CAT_Well_Water'

Claim size model with penalty

'CONSTRUCTION_YEAR', 'HOUDEN10KM', 'DEDUCTIBLE', 'RESIDENTIAL_AREA', 'BUILDINGS', 'ROOF_TYPE_CAT_Thatched_Roof', 'WATER_SUPPLY_TYPE_CAT_Private_Water_Supply', 'BASEMENT_AREA', 'OUTER_WALLS_CAT_Brick_Walls', 'HEATING_TYPE_CAT_Central_Heating_Own_System', 'YEAR', 'WETROOMS'

Feature selections

Claim rate model

'CONSTRUCTION_YEAR', 'HOUDEN10KM',
'DEDUCTIBLE', 'RESIDENTIAL_AREA',
'ROOF_TYPE_CAT_Thatched_Roof',
'WATER_SUPPLY_TYPE_CAT_Private_Water_Supply',
'BASEMENT_AREA', 'BUILDINGS',
'OUTER_WALLS_CAT_Brick_Walls',
'WATER_SUPPLY_TYPE_CAT_Water_Extraction_Plant'

Claim size model without penalty

'CONSTRUCTION_YEAR', 'HOUDEN10KM',
'DEDUCTIBLE', 'RESIDENTIAL_AREA',
'ROOF_TYPE_CAT_Thatched_Roof',
'WATER_SUPPLY_TYPE_CAT_Private_Water_Supply',
'BASEMENT_AREA', 'BUILDINGS',
'OUTER_WALLS_CAT_Brick_Walls',
'WATER_SUPPLY_TYPE_CAT_Water_Extraction_Plant'
'OUTER_WALLS_CAT_Timber_Framing',
'HEATING_TYPE_CAT_Central_Heating_Own_System',
'OUTER_WALLS_CAT_Wood_Cladding',
'ROOF_TYPE_CAT_Fiber_Cement_Asbestos',
'HEATING_TYPE_CAT_District_Heating',
'OUTER_WALLS_CAT_Lightweight_Concrete',
'HEATING_TYPE_CAT_Electric_Heating',
'WETROOMS', 'HEATING_TYPE_CAT_Stove_Fireplace'