Credit risk scoring modeling with XGBoost

PwC Business Case Competition 2019

CONTENTS

UNDERSTANDING THE OBJECTIVE AND THE DATA SET

The objective was to create a credit scoring model, which estimates the probability of default on loan of a potential customer

Process of thinking:

Input variables – 90 features in total

Application Data: 13 features

Hypothesis: The more relatively wealthier a given customer is, the lower the probability of the default

Geolocation Data: 38 features

Hypothesis: The more relatively wealthier a given location is, the lower the probability of the default

Behavioural Data: 39 features

Hypothesis: The less risky behaviour in the past, the lower the probability of the default in the future

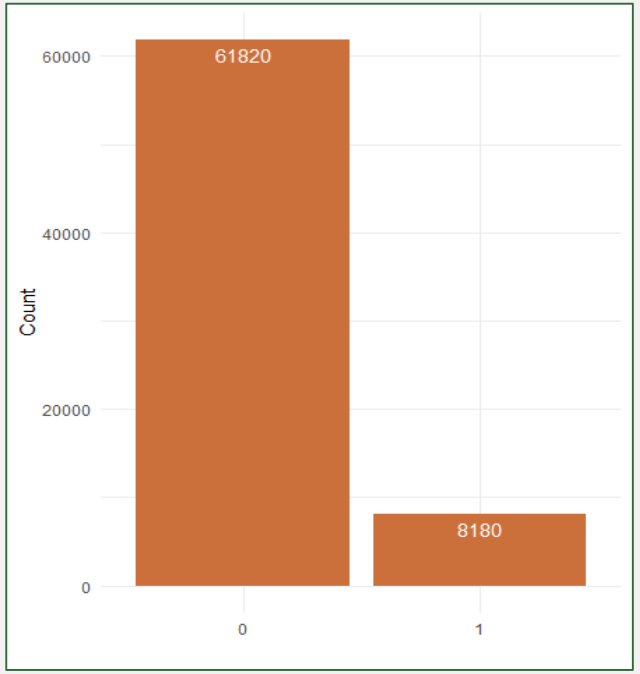
Model: binary classifier

Objective: find a function *f*, which describes the relationship between input variables and output variable

Output variable

Default Flag: binary variable, indicates whether the exposure defaulted (is due more than 90 days) during 12 month time form origination period

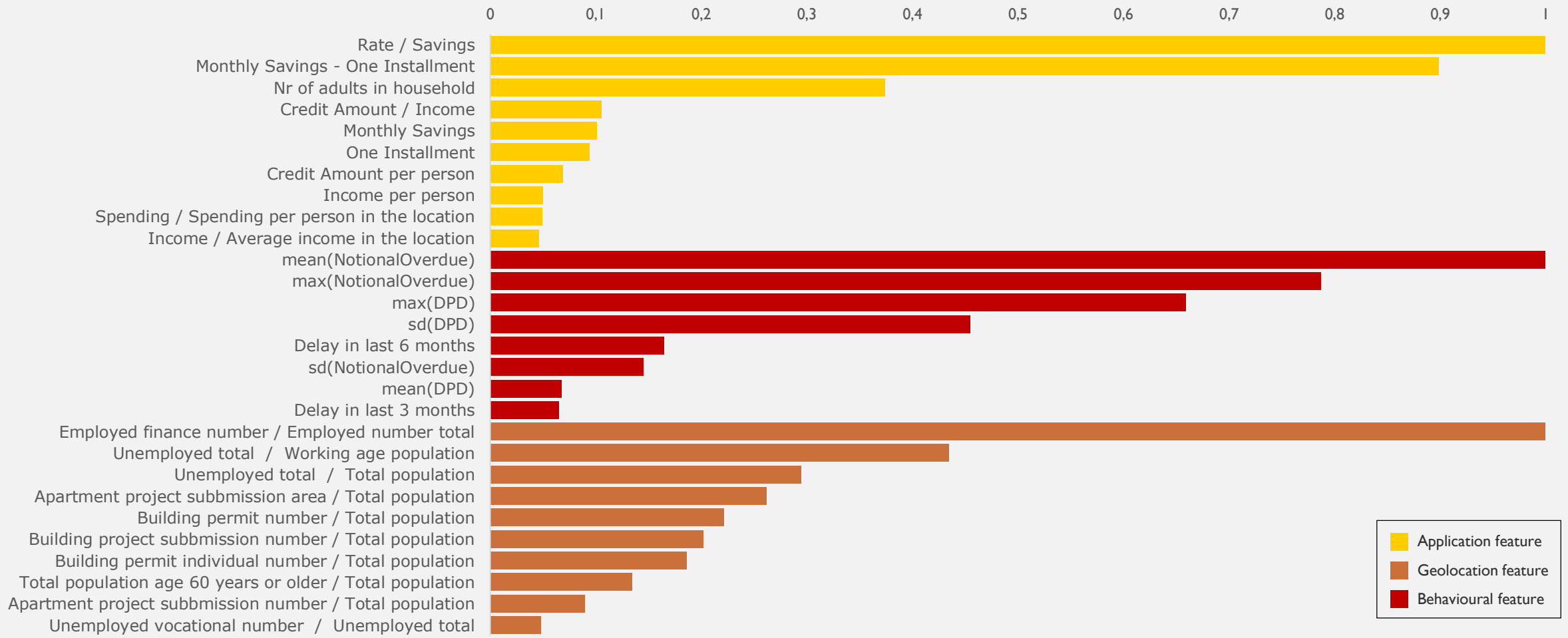
Class unbalance: 9 : 1



FEATURE ENGINEERING

To boost a model performance, 54 additional features were created based on raw data set

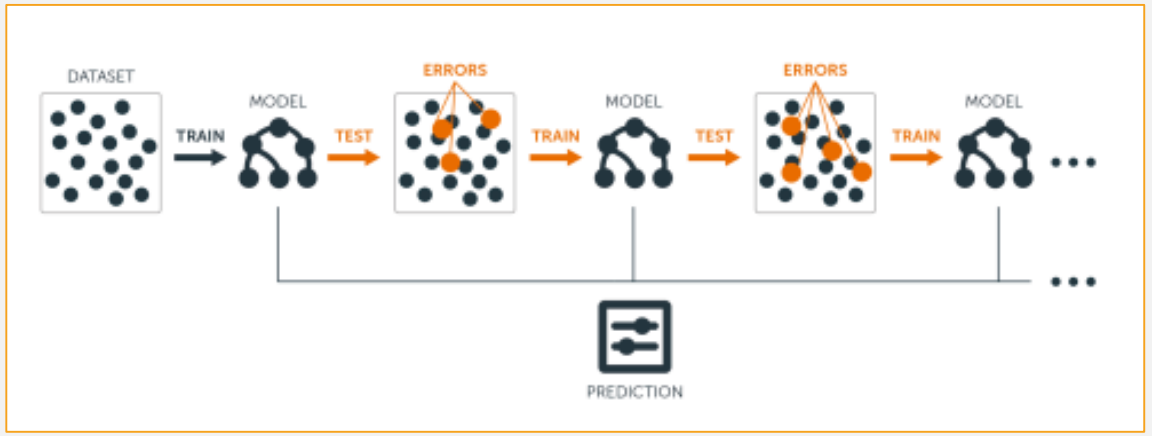
Relative gain of chosen variables created by our team



MODEL SELECTION

After the experimentation with different models, XGBoost implementation of gradient boosted decision trees was chosen as the most effective

- **GBMs** build an ensemble of shallow and weak successive trees with each tree learning and improving on the previous. At each particular iteration, a new weak, base-learner model is trained with respect to the error of the whole ensemble learnt so far. The final decision is made by calculating a weighted average of learners' estimates, where weights are their accuracy of classification.



The most important model's characteristics from the objective standpoint:

- Approximates non-linear transformations as well as subtle and strong variables interactions
- It's only approximation – strong interactions benefit from being explicitly defined (rations/sums/differences among features)
- Very prone to overfitting – the model requires hyperparameters tuning (parameters set before the learning process begins) and careful test set error estimation approach
- XGBoost implementation handles only numeric features – categorical features need to be one-hot encoded

MODEL TUNING

The hyperparameters were chosen through simple grid search and then adjusted manually. To accurately estimate the test set error, 10-Fold Cross-Validation was used

Hyperparameters values used in the final model:

- booster = "gbtree"
- eval_metric = "auc" – **Gini = 2 * AUC - 1**
- nthread = 4
- eta = 0.05 - **Learning rate**
- max_depth = 4 - **the maximum depth of a tree**
- min_child_weight = 30 – **the minimum sum of weights of all observations required in a child**
- gamma = 0

Test set error estimation approach:

- 10-Fold Cross-Validation

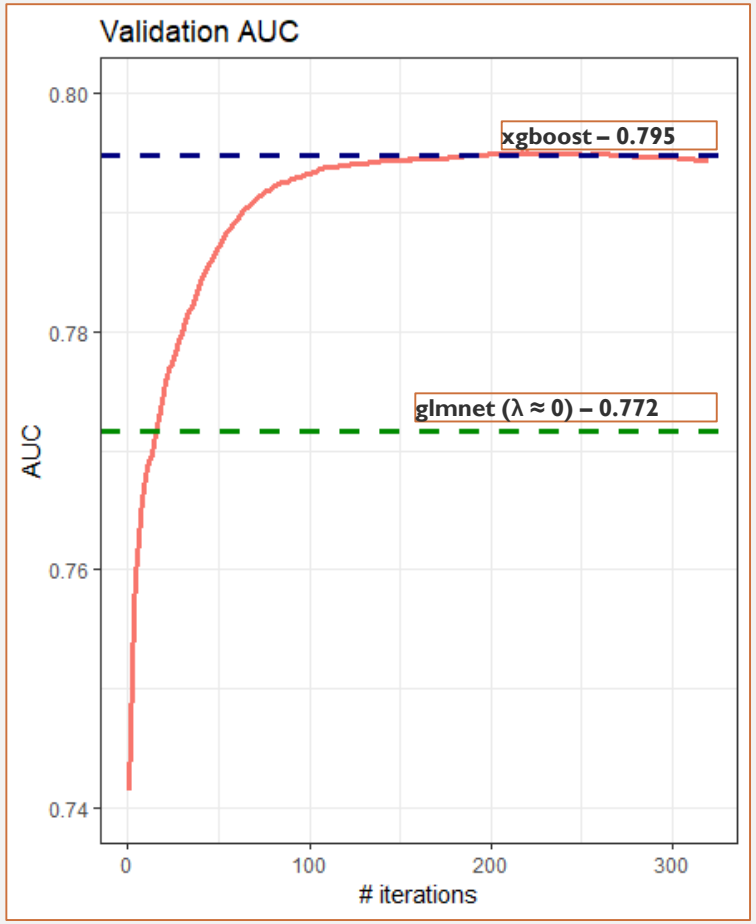
- subsample = 0.8 – **the fraction of observations to be randomly samples for each tree**
- colsample_bytree = 0.75 - **the fraction of columns to be randomly samples for each tree**
- colsample_bylevel = 0.85 - **the subsample ratio of columns for each split, in each level**
- alpha = 0
- lambda = 1 - **regularization term on weights (analogous to Ridge regression)**
- nrounds = 2000
- scale_pos_weight = 9 – **additional weight for incorrectly classified observation with positive class**

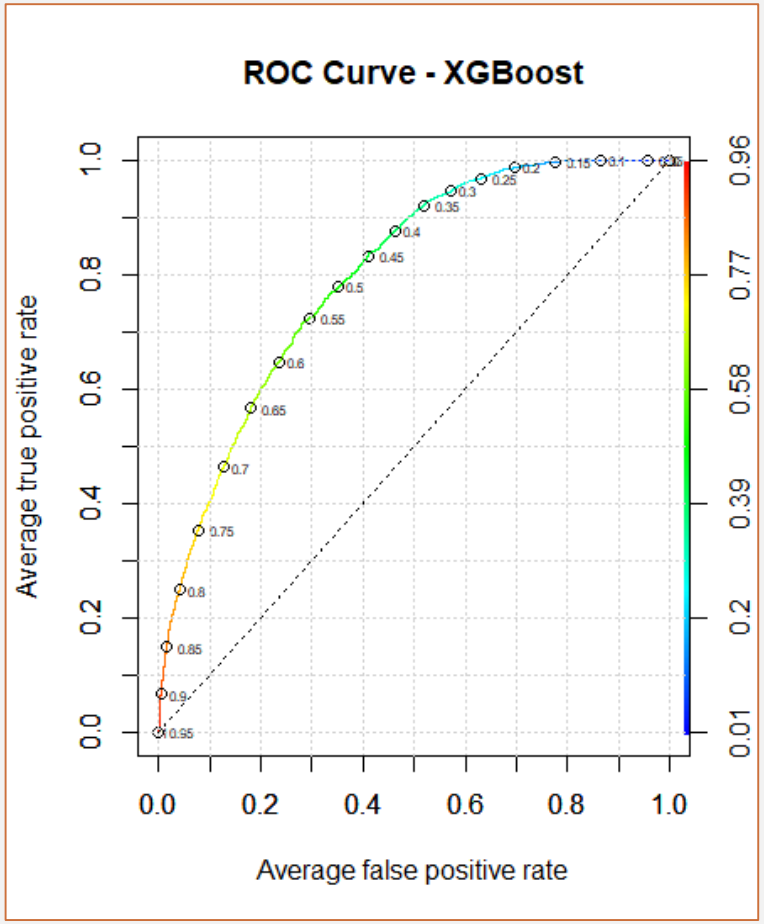
RESULTS – MODEL PERFORMANCE

The final model was able to reach Gini coefficient of 63.6% on the entire data set, while achieving 3.4% discrepancy between the estimates on training and test sets

Final model performance:

- Train:
  AUC – 0.82 | Gini - 64.63%
- Test:
  AUC – 0.80 | Gini - 61.24%
- All:
  AUC – 0.81 | Gini - 63.60%

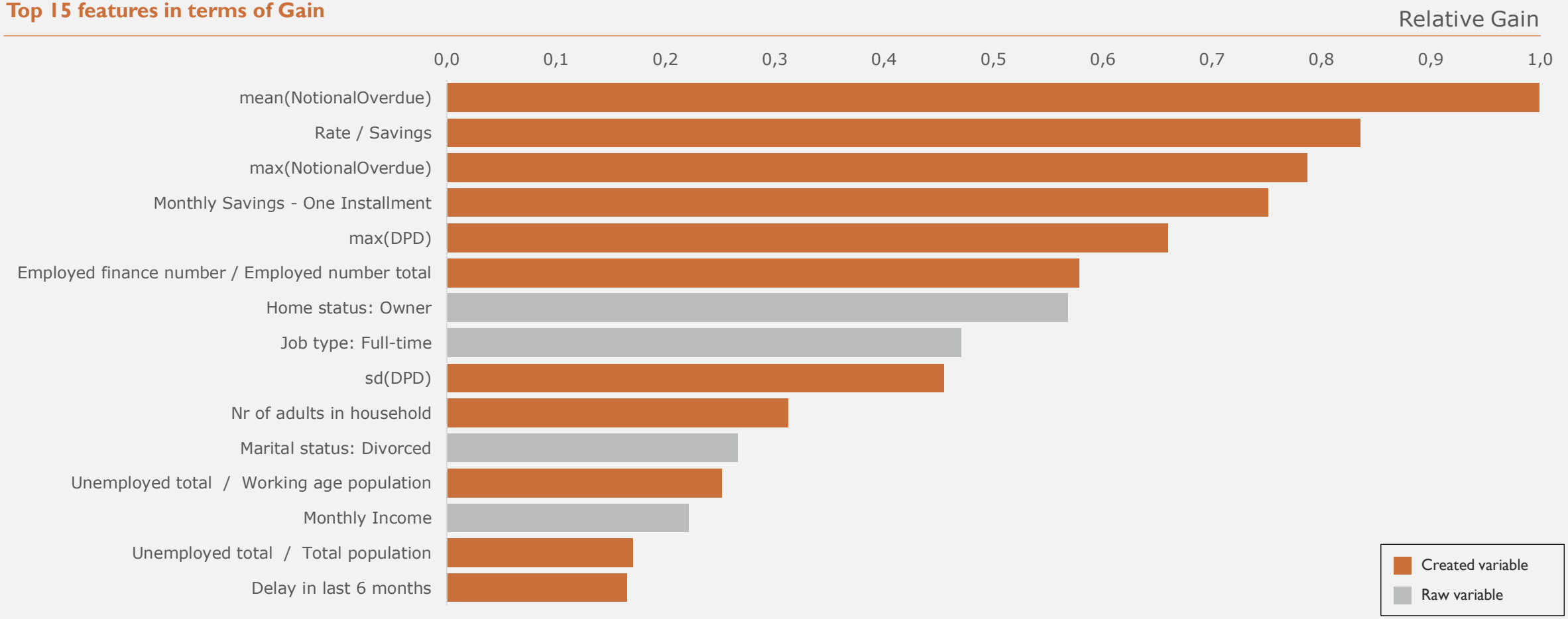




RESULTS - FEATURE IMPORTANCE

Our team created 11 out of 15 the most important variables in the final model in terms of Gain measure

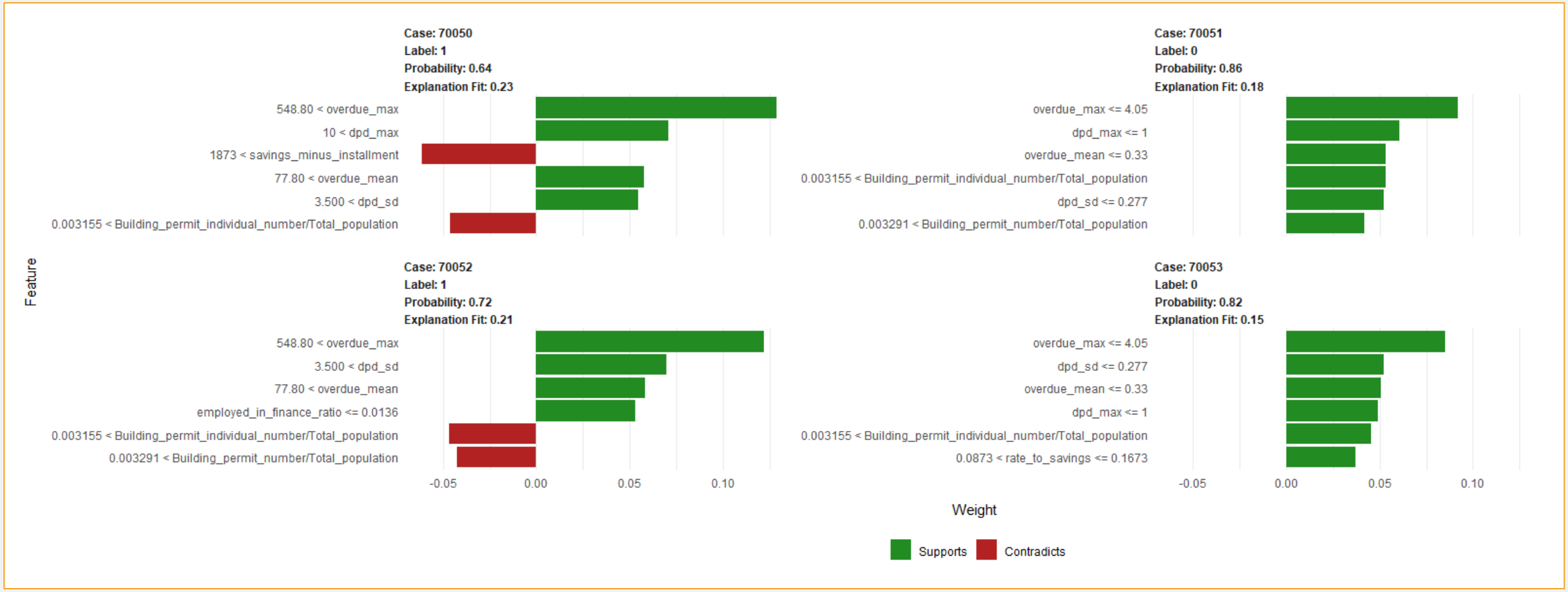
Top 15 features in terms of Gain



RESULTS - INFERENCE

GBMs flexibility and prediction accuracy come at the expense of interpretability. To address that, the end client of the model can use local interpretations to understand the predictions

Local interpretations of the final model for 4 example observations from test set using LIME package:



POSSIBLE IMPROVEMENTS

1. Using Bayesian optimisation / Random search to tune the model's hyperparameters
2. Stacking the Elasticnet and XGBoost models