Data Science Challenge

Summary

- The Origination data and Monthly data files were used to predict defaults and estimate the time to default of single family loans.
- Tree based and perceptron based models were examined for predicting defaults. LightGBM and CatBoost algorithms were investigated as contenders of GBM model algorithm.
- A GBM Decision Tree Classifier using LightGBM was proposed to predict defaults; it achieved an AUC = 0.86.
- Tree based regression techniques were investigated for estimating time to default.
- An analysis on different loss functions to be optimized was performed.
- A Random Forest Quantile Regressor was proposed to estimate the time to default of loans. It estimated the time of default of 87% of the loans within the intervals it established.

Data

- Number of loans in Origination Data file (ODF): 621,539.
- 49 loans excluded from the data set - do not appear in Monthly Performance Data file (MDF).
- Default rate = 2.3% (14,317 defaults).
- The MDF has loan history starting from Loan Age = 0.
    - However , 5.8% loans start from Loan Age = 1.
    - Triangulation using Unpaid Principal (in ODF) and Initial Unpaid Balances (in MDF) did not work.
    - Possible reasons:
        - *"Note that the monthly performance data file does not contain monthly performance information for the timeframe between loan origination and loan acquisition by Freddie Mac"* - User Guide, page 18.
        - causes of data imperfection listed on page 19 of User Guide.
    - These were retained in the data set; the minimum "loan age" was used to select appropriate rows in MDF (instead of "loan age" = 0); removing them did not affect the model.
- 50 accounts did not have a credit score (i.e. credit score set to 9999); these were retained in the data set.

FIGURE

Data

Handling of empty values

- Missing values due to the nomenclature (lack of!) in data recording:
  - missing values in Super Conforming Flag and HARP were imputed with “N”.
  - missing values in Metropolitan Statistical Area were imputed with “99999”.
  - missing values in Zero Balance UPB were imputed by “0”.
- Pre HARP Status was modified to indicate the following: A - ARM, F - FRM and N - No Pre HARP.

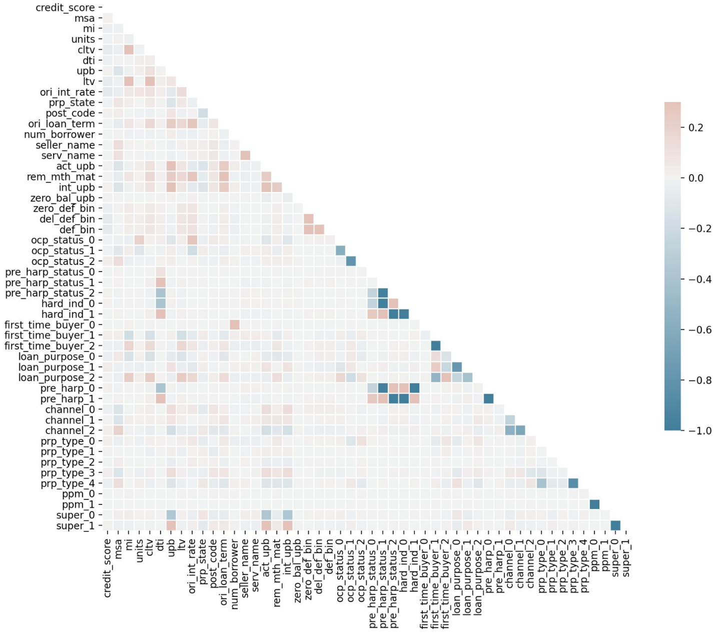
Handling of outliers

- Program- and Interest Only- Indicator and Property Valuation Method have no variability and are dropped.
- Some outliers were removed from the data set.

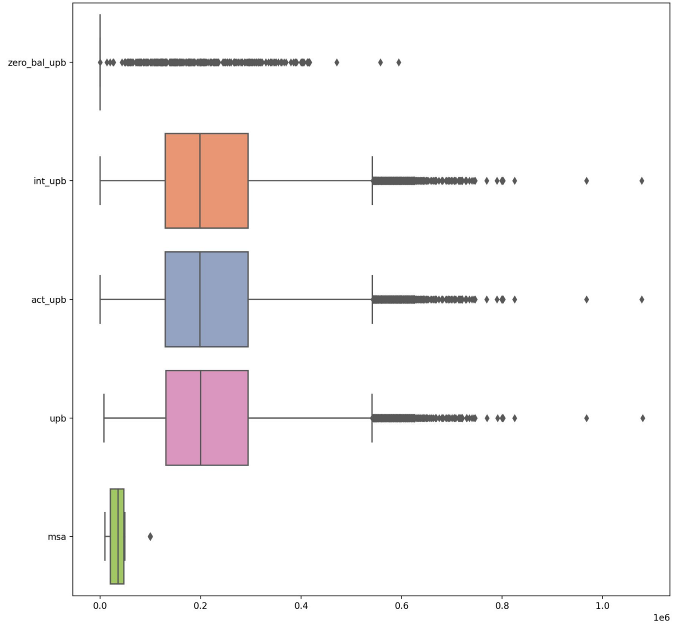
FIGURE

Data

Correlation heat map



Boxplot of some numerical variables



Data

Handling categorical variables

- Categorical variables are encoded if they are non-binary. The non-ordinal variables are one-hot-encoded where as the ordinal ones are label encoded.

  Note: The Gradient Decision and Categorical boosted tree algorithms auto-encode the variables that are labelled with categorical data types.

Handling class imbalance

- The data is unbalanced given the minority class comprises 2.3% of data (default rate). We apply majority undersampling techniques to adjust the dataset to be more balanced.

Part 1: Prediction of probability of default

Model

We investigate two schools of models to predict loan default at origination: tree based and perceptron based. This is because:

- tree based techniques often works great with categorical and numerical values.
- perceptrons can efficiently approximate complex target functions and they come with good algorithms for fitting the data

Gradient Boosted Decision Tree algorithm

A procedure that combines the output of weak classifiers to produce a powerful committee. We examine the performance of two flairs of this algorithm, LightGBM and CatBoost.

Neural Network

A generalization of the perceptron which uses a feature transform that is learned from the data.

FIGURE

Model: Performance metrics

- We examine metrics to evaluate the performance of the models:
    - false positives lead to missed business opportunities.
    - false negatives cause defaults.
- We establish the performance of the models by examining the true positive rate (sensitivity/recall) and true negative rate (specificity) using the area under the ROC curve.
- We favor this metric over precision since the loss incurred from defaults can be more costly than the loss incurred from missed business opportunities.
- Due to the imbalance nature of the dataset, we measure the F1 performance of the models.

Neural network: Simulation framework

- One-hot-encoding was applied to categorical variables that were non-ordinal. However, this did not improve the out-of-sample performance of the model.
- Label encoding was applied to categorical variables that were ordinal.
- Min-max scaling was applied. Other transformations did not yield good performance.
- The network was trained and validated on 60% and 20% of the data respectively, and tested on 20% of the data. All splits were stratified.
- The network was trained with batches of 100 and across 100 epocs.
- Undersample the majority class to 0.33 : 0.67 ratio. Ensured that the test sample was not adjusted by undersampling; only the training and validation data were adjusted. This was done to examine the performance of the model in real world scenarios.
- The performance of the network was established across 10 iterations. Each iteration took 20 minutes to complete.

FIGURE

Neural network: Architecture and Results

- We propose the following architecture for the neural network :
    - dimension: 45 x 90 x 45 x 1
    - activation function: Leaky ReLU (hidden layers), Sigmoid (output layer)
    - loss optimization function: Binary cross entropy
    - regularization: L^2 Regularization, 1e-3
    - batch normalization at every hidden layer
    - learning rate: 1e-3
    - optimizer: Adam optimizer
    - weight initialization: Xavier

Results

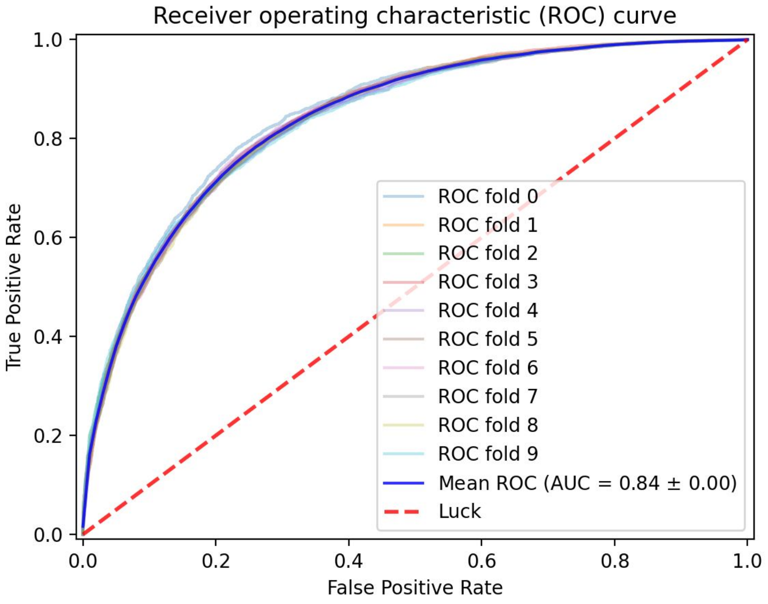
Table 1. Performance of Neural network in default classification

Recall	Specificity	AUC	F1-score
81.5%	67.0%	0.84	0.6

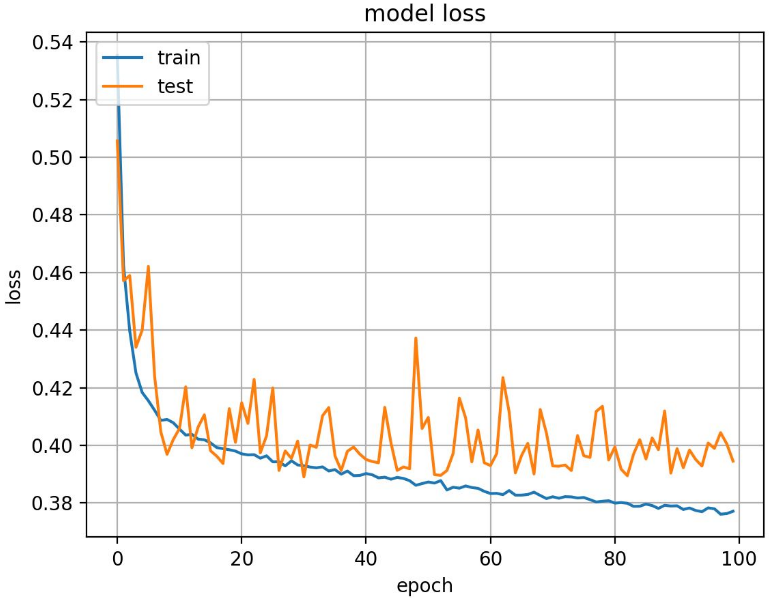
Note: All values are averages taken across 10 iterations

Neural network: Results

Aggregate performance



Epoc performance

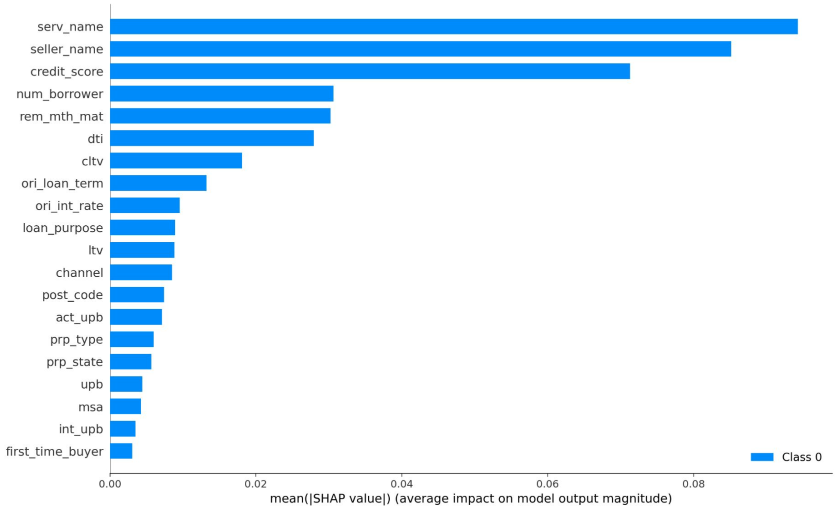


Note: Loss is Binary cross entropy

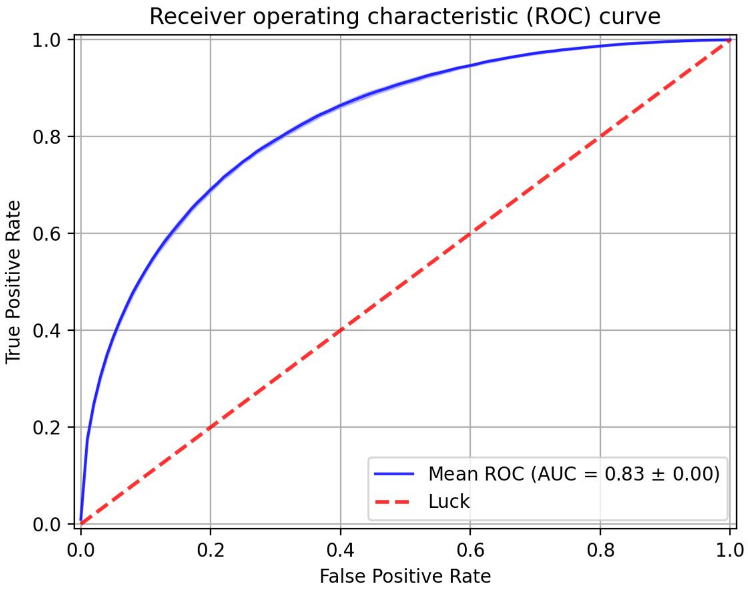
FIGURE

Neural network: Variable importance

- We examine the variables importance using SHAP values.



- In efforts to reduce model complexity we build a network with the top 6 variables and lose 1% in AUC.



FIGURE

Boosted decision tree: Simulation framework

- Undersample the majority class to 0.33 : 0.67 ratio. Ensured that the test sample was not adjusted by undersampling; only the training and validation data were adjusted. This was done to examine the performance of the model in real world scenarios.
- The network was trained and validated on 60% and 20% of the data respectively, and tested on 20% of the data. All splits were stratified.

Boosted decision tree: Architecture and Results

- We use LightGBM and CatBoost algorithm's 'default' architectures to investigate their performance.
- Results:

Table 2. Performance of LightGBM in default classification

Recall	Specificity	AUC	F1-score
82.3%	73.0%	0.86	0.62

Note: All values are averages taken across 10 iterations

Table 3. Performance of CatBoost in default classification

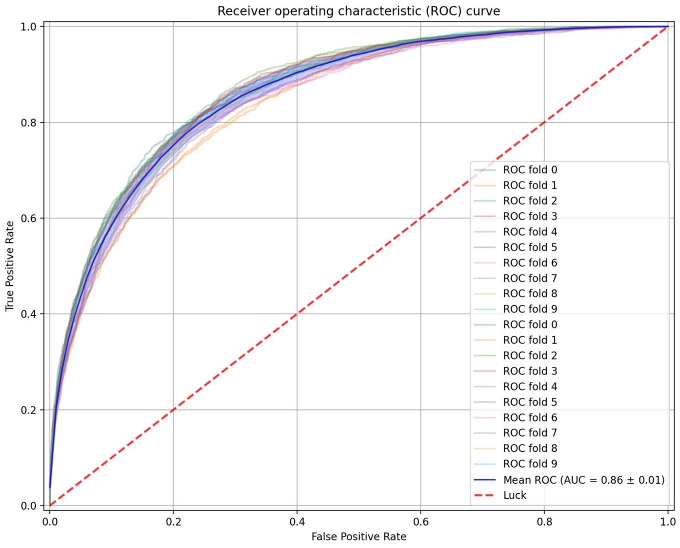
Recall	Specificity	AUC	F1-score
82.7%	73.0%	0.86	0.62

Note: All values are averages taken across 10 iterations

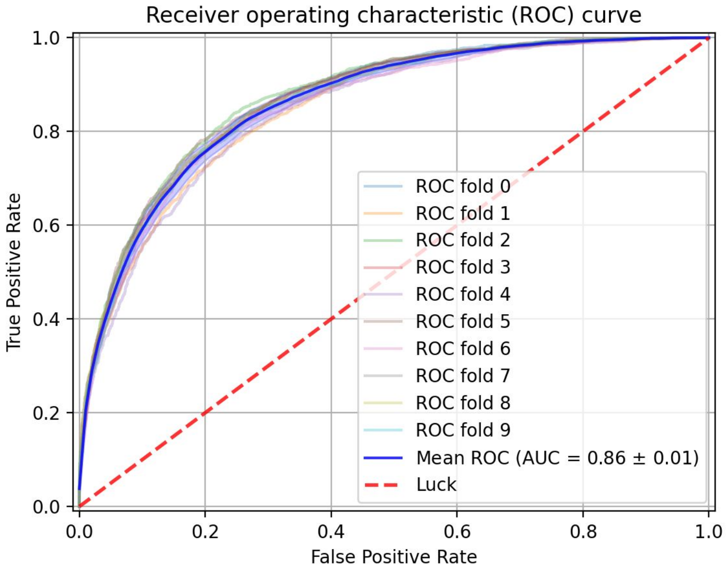
- We observe similar performance between the two algorithms.
- We also gain a 6% increase in specificity over the neural network model.

Boosted decision tree: Baseline performance

Light Gradient Boosted Machine

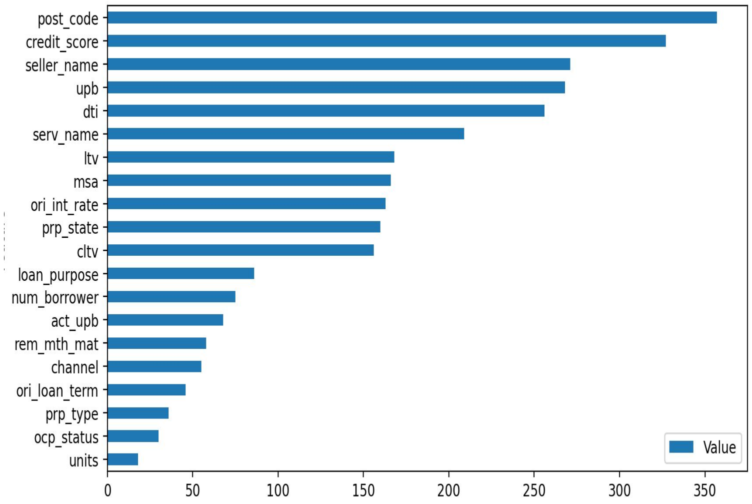


Categorical Boosted Tree

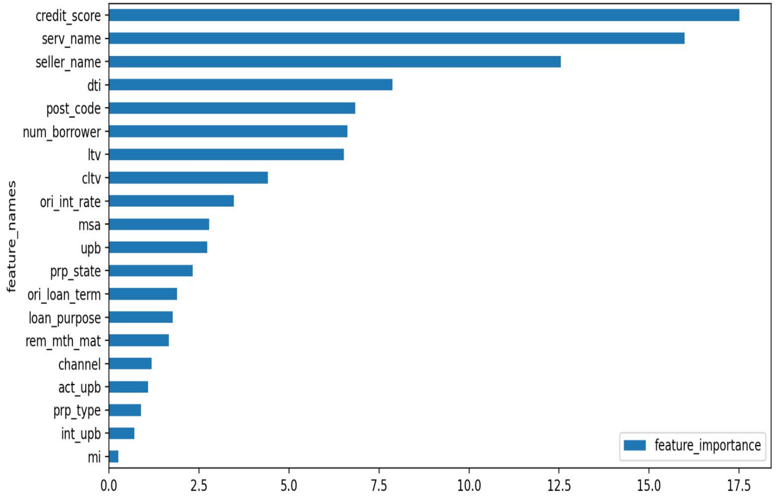


Boosted decision tree: Variable importance

Light Gradient Boosted Machine



Categorical Boosted Tree



FIGURE

Hyperparameter tuning

- In efforts to attain better results, we perform hyper-parameter tuning of the gradient boosted models.
- We apply the Bayesian optimization searching method.
- The following parameters are tuned for the LightGBM and CatBoost algorithm:
    - learning rate; number of leaves; maximum depth ;
    - regularization parameters, l1 and l2;
    - minimum number of datapoints in leaves

- Some hyperparameters overlap in their purpose. For example, tree complexity can be controlled by maximum depth, maximum number of leaves or minimum sample (count or weight) per leaf. Using a combination of these might be optimal for some problems.

- L1 regularization is applied to leaf scores rather than to features. This reduces the impact of less predictive features, but does not remove them (as in logistic regression). It made sense to use both L1 and L2: some L1 to punish the less-predictive features, and some L2 to further punish large leaf scores.

FIGURE

Hyperparameter tuning: Results

Table: 4 Hyper-parameter tuning of GBM Decision Tree

```
[LightGBM] [Warning] min_data_in_leaf is set=27, min_child_samples=65 will be ignored. Current value: min_data_in_leaf=27
[LightGBM] [Warning] verbosity is set=-1, verbose=-1 will be ignored. Current value: verbosity=-1
```

iter	target	lambda_l1	lambda_l2	learni...	max_depth	min_ch...	min_da...	num_bo...	num_le...
1	0.852	1.635	0.7974	0.1497	7.775	65.53	27.8	503.3	47.7
2	0.8574	1.971	1.032	0.1122	45.15	81.27	31.2	317.1	31.94
3	0.8571	0.7768	0.7898	0.101	41.31	79.92	29.19	321.8	37.02
4	0.8623	0.9731	1.432	0.08725	44.95	84.52	83.36	195.7	12.37
5	0.8436	1.464	0.7189	0.22	17.57	78.21	76.2	194.5	33.11
6	0.843	1.558	0.8114	0.2421	44.11	87.9	86.64	192.4	10.99
7	0.8538	1.518	1.993	0.2256	20.36	59.13	19.22	464.8	17.68
8	0.8283	1.032	1.607	0.2478	9.636	88.28	40.44	910.7	27.74
9	0.8659	1.917	1.394	0.01974	33.88	74.51	72.93	471.3	48.96
10	0.7902	1.165	1.645	0.2974	33.16	56.3	82.81	718.4	12.94
11	0.852	0.6034	0.8831	0.1492	48.89	76.63	72.41	878.7	39.83
12	0.8539	0.7943	1.204	0.003936	41.8	81.07	78.22	205.1	15.59
13	0.8534	1.042	1.108	0.1397	37.3	68.55	78.46	466.3	41.7
14	0.8337	1.644	0.5197	0.2358	37.12	85.13	70.7	472.2	43.99
15	0.8494	1.819	1.869	0.1701	42.47	68.1	81.46	335.5	42.58

Table: 5 Hyper-parameter tuning of CatBoost

iter	target	depth	l2_lea...	learni...	max_le...	num_bo...
1	0.855	1.388	3.196	0.3944	87.69	143.9
2	0.8561	2.265	2.929	0.9722	73.21	152.1
3	0.8584	2.182	3.89	0.9177	73.12	152.7
4	0.8542	3.128	2.239	0.701	66.03	138.0
5	0.8631	2.507	2.998	0.4989	72.51	152.9
6	0.859	2.066	2.456	0.8631	98.62	174.3
7	0.856	2.147	2.172	0.9963	78.01	191.2
8	0.8551	1.376	4.578	0.9407	63.1	164.8
9	0.8423	3.339	3.31	0.9961	84.77	198.2
10	0.8647	3.073	3.734	0.3182	72.04	153.8

- Hyper-parameter tuning does not yield a significant gain in performance;
  AUC = 0.8659
- Final proposed model:
  GBM Decision Tree Classifier

  learning rate = 0.01974
  max. depth = 33
  regularizer, l1 and l2 = 1.917, 1.394
  number of leaves = 48
  minimum number of datapoints in leaves = 73

FIGURE

Points to note

- Postcodes were retained in the dataset. This is because the splitting nature of tree based models could group postcodes into areas that can classify defaults. Postcodes were also included in the neural network model.

- The three models have similar performances. It is worth examining overlaps amongst the misclassified loans. Investigating their monthly performance may give insights into their defaults. For example, some of them may be delinquency due to disaster, unpredictable at loan loan origin.

- Using the top 6 variables in the neural network reduced the AUC by 0.01 only. A similar investigation could be performed to see the impact on the tree based models, as it may lead to simpler models with stronger generalization.

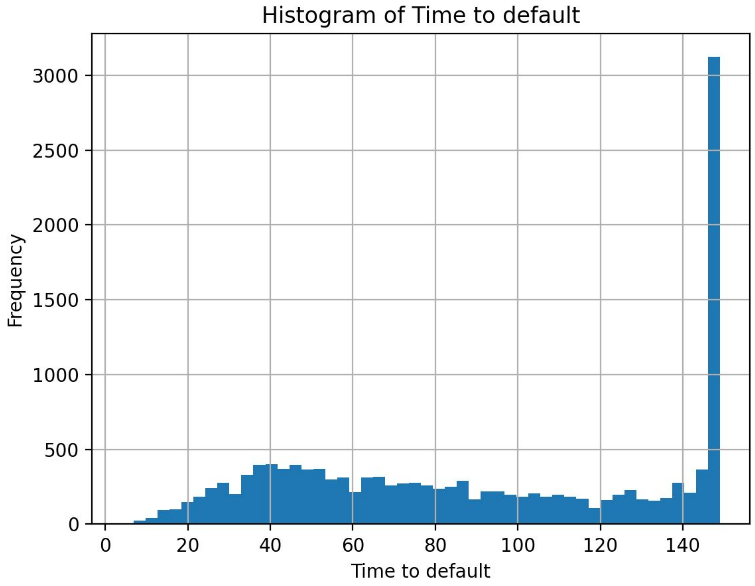
- The application of minority over-sampling techniques such as SMOTE could be investigated to improve performance.

FIGURE

Part 2: Estimation of time to default

Data

- We get the time to default of the 14,317 loans that defaulted in the data set.



- Average time to default = 91 months

- 21.8% defaults have default time between [147,149] months.

- Therefore, high variability in default time; std. dev. = 44 months

- 5 defaults have no history of credit score, these are retained in the dataset.

FIGURE

Model

We investigate the use of Gradient Boosted Decision Tree Regression model for estimating the time to default.

- We use the CatBoost implementation of this algorithm.
- We examine three loss functions: the RMSE, Huber and Quantile loss.
    - quantile loss: proposed because majority of the errors realized in RMSE were under predictions.
    - huber loss: proposed since there are large “outliers” in the data where MAE can be applied as well as points where RMSE would suit.
    - despite data being of type “count”, the use of Poisson loss was relegated since the mean and variance of defaults were very different.

Simulation framework

- We perform 5-fold nested cross validation to assess the performance of the tuned model with different loss functions.
- Hyper-parameter tuning with 10 iterations of random search and 3-fold cross validation.
- Early stopping exercised with 100 iterations of fitting.
- The following parameters are tuned (for each loss):
    - learning rate
    - l2 leaf regularization
    - depth

Result

Table 6: Perfomance of CatBoost with different loss functions

Loss	in-sample		out-of-sample	
	R2	MAE	R2	MAE
RMSE loss	0.27	31	0.17	72
Huber loss*	0.29	31	0.24	78
Quantile loss**	0.84	9	-0.14	84

* Huber metric of 0.2 maximizes in- and out-of- sample R2

** Quantile value of 0.2 maximizes in and out-of- sample R2

All values are averages across 5-fold cross validations.

- The Quantile loss has good in-sample r-square; however the model performs very poorly out-of-sample.

- The model had poor generalization for all loss functions despite applying regularization techniques.

FIGURE

Random Forest Quantile: Simulation framework

- We propose an alternative scheme, where upper and lower bounds of default time are established per loan. This done using a Random Forest Quantile Regression model.
- The model is trained and hyper-tuned on an in-sample fold and is then used to compute the upper and lower threshold for loans in the outer-fold.
- We establish the upper and lower boundaries on estimates of default time, at quantiles of 0.95 and 0.5 respectively.
- This yields a 90% prediction interval. Therefore, at the time of origination of the loan, an interval of time of default can be estimated, and the probability that the loan would default within this interval is 0.9.

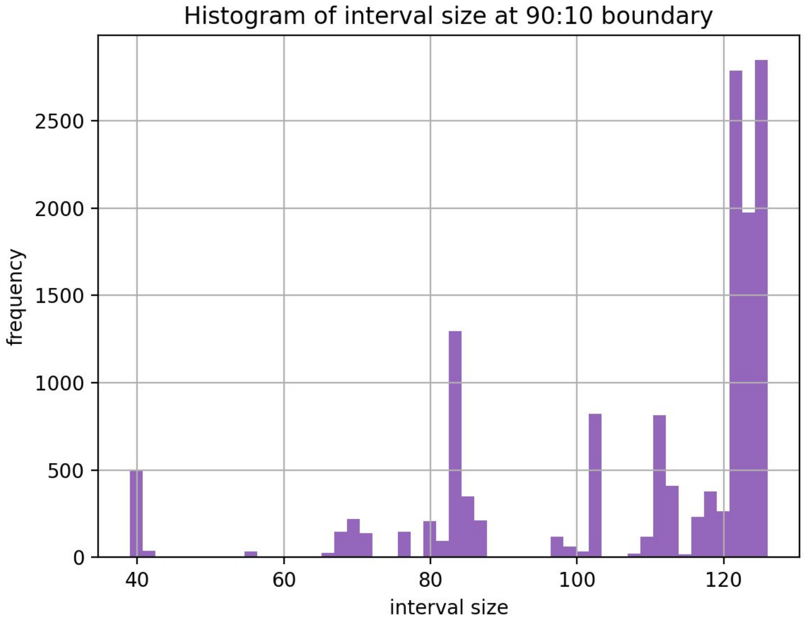
FIGURE

RF Quantile Regression: Result

- We establish the performance of the model on out-of-sample folds, put together to form the entire data set of defaults.

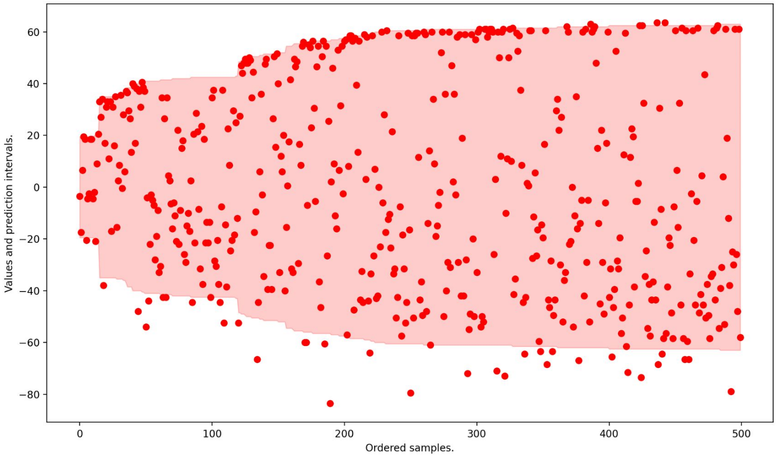
Table 7: Performance of RF Quantile Reg.

Interval	within	above	below
95:5	87.4%	1.5%	4.6%
90:10	76.2%	6.7%	9.2%
80:20	57.1%	16.2%	19.3%
70:30	37.9%	28.1%	29.4%



FIGURE

RF Quantile Regression: Result



- We plot 500 random loans along with their intervals (with all values scaled to zero-mean).

- Final proposed model:

  Random Forest Quantile Regression
  number of estimators: 50
  maximum leaf nodes: 10
  maximum depth: 6

FIGURE

Part 3: Prepayment risk

Prepayment risk

- Prepayment, much like default, is an event that generates loss of potential income.
- In the presence of prepayment risk, the model would have to be able to identify whether the loss of income would likely be due to prepayment or default.
- It would therefore need to model the likelihood of prepayment as well as default, together with estimating the time, at the time of loan origination.
- To build the model, the “zero balance code” can be used to distinguish the cause of loss of income.
- A neural network based approach with 2 or 3 outputs: modelling probability of prepayment and probability of default, and estimate of time, may prove useful.

FIGURE