Logistic Regression vs Machine Learning

By Mike Hay for Open Analytics

www.open-analytics.com.au

Introduction

The purpose of this project is to describe some practical considerations that could influence your decision to decide between machine learning and a more traditional statistical technique in your next credit risk modelling project.

The aim is not to find a winner, they both have their place, we're exploring some of the pros and cons of each.

The Task

The task is to predict the probability of default over a 12-month forward-looking window on Open Analytics CPM data, where default means 90+ days in arrears. As such, it's a binary classification model with a combination of numeric and categorical predictors.

The methods used to model the PD are binary logistic regression, where all the predictors have been binned, and LightGBM, which is a gradient boosting method similar to XGBoost.

The performance of the models is measured using the Gini coefficient. We will also discuss the ease of use and explainability of each technique.

There are multiple aspects to any modelling project, many of which are omitted here. The data preparation phase is usually where most time is spent, but we do not consider data preparation in this study. Variable selection is another crucial modelling phase that is omitted. These two phases are similar whether you choose machine learning or logistic regression.

The data for our experiment is Open Analytics' Non-Bank Asset Finance data used for Credit Portfolio Manager (CPM):

- Mixture of asset finance lenders using our hosted credit provisioning service.
- Primarily broker-originated low doc lending (non-financial assessment).
- The training set runs from 2020-03 to 2023-12.
- The out-of-time sample runs from 2024-01 to 2024-07 (with a year removed for the 12-month default window).

Model Design:

- The definition of default is a 90+ arrears trigger and a forward-looking 12-month default window. It's a behavioural scorecard.
- Model performance will be measured using the Gini coefficient.
- The machine learning model is LightGBM – a gradient boosted decision tree model developed by Microsoft. Similar to XGBoost.
- The implementation of the Logistic Regression model is the glm function in R. It is supported by Open-Analytics variable selection, binning, and diagnostics algorithms.

Fitting Logistic Regression

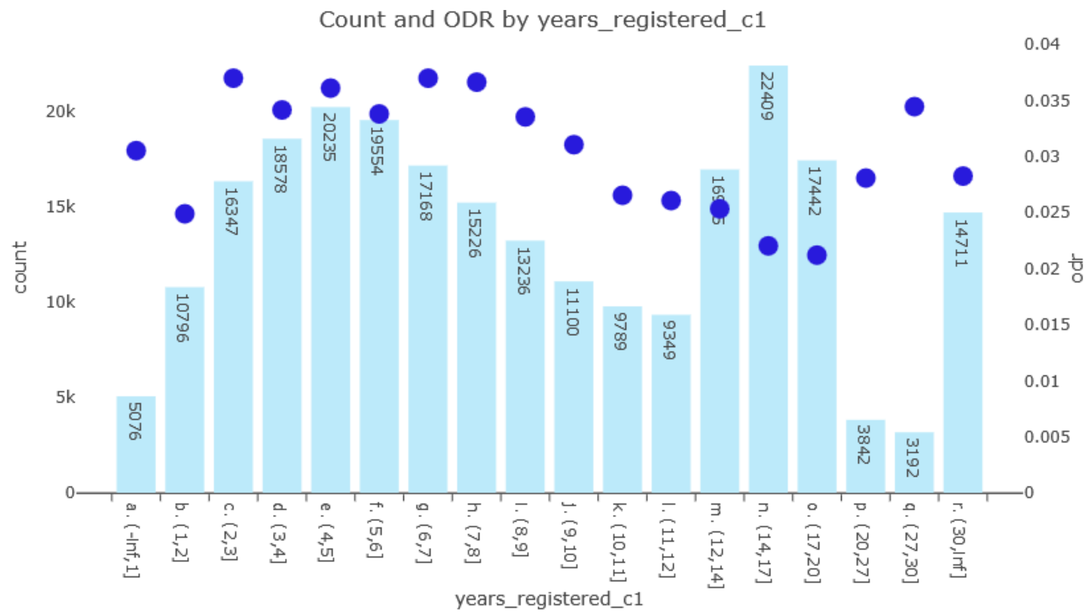
The logistic regression model is very well established and understood. There are many references explaining it and trust-worthy packages that perform Logistic regression very efficiently.

The time-consuming part with logistic regression is often the binning (after the data has been prepared and predictors selected). But this is also a major strength of Logistic regression, providing flexibility and control. There are plenty of binning algorithms that will automatically discretise numerical fields, or group together categorical fields, in an optimal way relative to some objective. However, these algorithms will usually produce results that have some undesirable characteristics from a business perspective. The modeller has total freedom over the binning, so they can choose it in a way that carefully balances the competing objectives of model performance, interpretability, and business objectives.

In practice, what often happens is the binning is first carried out algorithmically, providing the statistically optimal outcome, then the bins are modified manually to arrive at a solution that is both performant and palatable to business stakeholders, potentially including regulatory bodies.

For example, a binning algorithm might be run on the numerical field “age of business” and produce the outcome shown below. It’s a granular binning with good separation that achieves a univariate Gini of 12.6%.

However, it does have some issues. The trend is not smooth, there are reversals from 1 to 8 years and above 17 years. If the objective is to produce the highest Gini, irrespective of the trend, then this binning may suffice. Usually, business objectives dictate that the trend needs to be intuitive. It would be hard to explain that a loan can be provided if the customer has been in business for 3 to 4 or 5 to 6 years, but not 4 to 5 years.

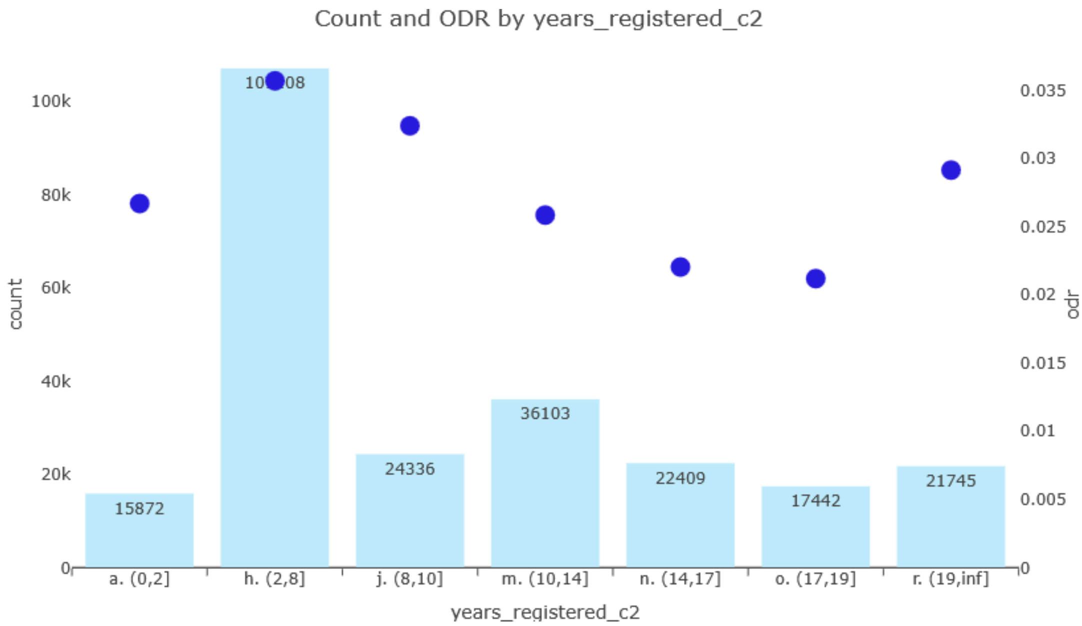


Bin boundaries can be manually changed to bring about an expected trend, as shown below, which decreases the univariate Gini from 12.6% to 12.1%, but is more explainable. The trend makes sense because between 0 and 2 years there is a lower default rate, new businesses usually begin with enough capital to survive their first few years. Between 2 and 19 years, the expected trend occurs, where a longer relationship implies lower risk.

Potentially unexpected is the uptick in defaults for greater than 19 in business. This is understood, though, because established businesses that are seeking funds tend to either be facing difficulties or undergoing expansion, circumstances associated with higher risk. This part of the trend is therefore intuitively reasonable, but it still may not be desirable from the business perspective.

With Logistic regression this is ok, many implementations of the algorithm include an offset function, so the 19+ years bucket can have whatever coefficient the modeller deems appropriate, and the algorithm will optimise the rest of the coefficients around it.

In this way the modeller has complete control over logistic regression, but must find the right balance between optimising model performance and achieving business objectives. This can be time consuming and may require multiple iterations as various stakeholders add requirements.



All variables can be binned in this way, then the logistic regression algorithm is run to find the marginal scores in each bin. Due to correlations between predictors, score reversals may still occur, but these can be dealt with by re-binning.

Fitting lightGBM

Gradient boosting methods, such as LightGBM or XGBoost, are newer than logistic regression but they are still well-established and are supported by mature and readily available packages. No binning is required. So long as the data is in the right form, these algorithms can handle pretty much anything you can throw at them.

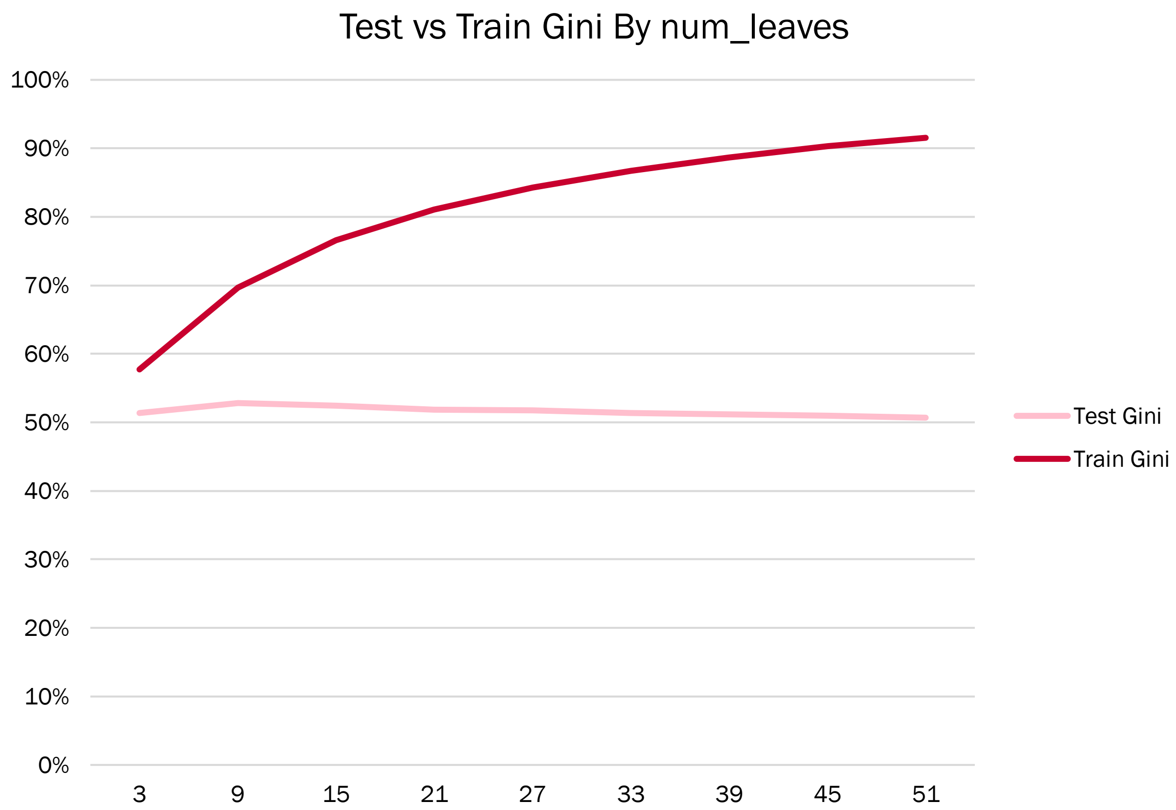
That all sounds very convenient, but care must be taken to avoid overfitting the model. The level of overfitting can be controlled using optional hyperparameters. If the modeller chooses to omit setting hyperparameter values, the LightGBM package will use its default values, which are reportedly quite good (more on this below). We recommend performing a hyperparameter search.

Hyperparameter optimisation was conducted using a grid search and cross validation. Caution was taken to capture any overfitting, cross validation was used to tune the hyperparameters on the modelling sample, with an out of time sample was kept separate for further testing.

There are many hyperparameters available with LightBGM. When varying them independently, max_bins, num_leaves, and max_hessian_in_leaf were the three that had the biggest impact, so these three are the focus of the present study.

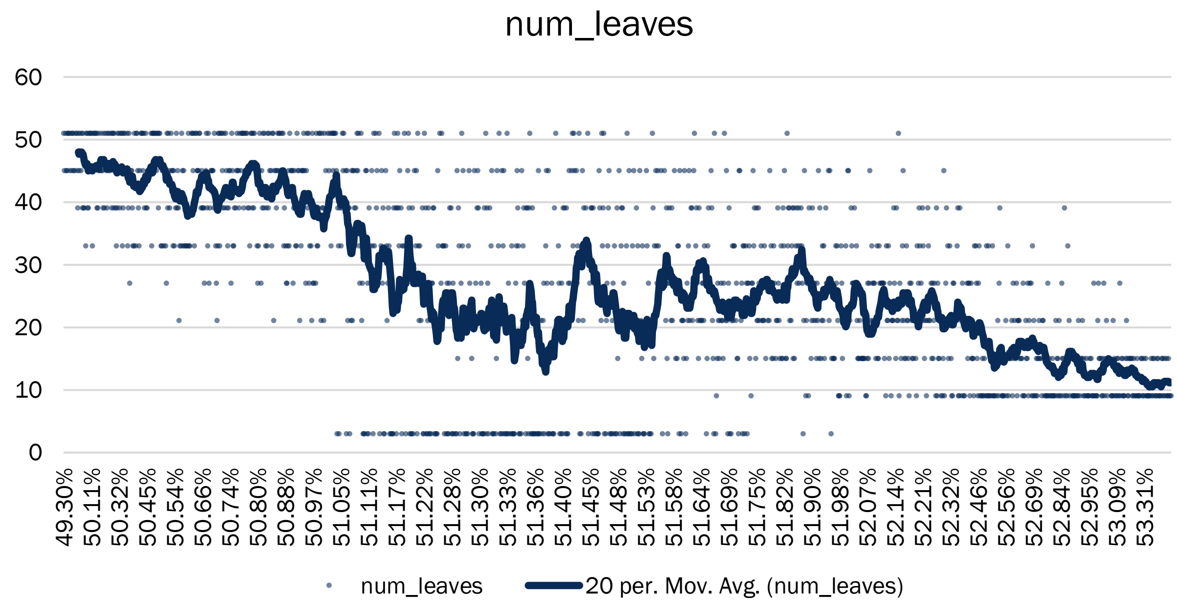
Grid search is a brute-force method that simply steps through each dimension of the hyperparameter space. There are more efficient methods, but grid search is good for demonstrating how the performance changes throughout the tested ranges of the parameters.

A wider, course grid search was conducted first, to view a wide range of parameter values. A single hyperparameter can be viewed by averaging the performance over the other two. The test and train Gini are plotted against num_leaves in the following chart. Both the test and train Gini results given in this section are from the training sample, not the out of time sample. The test Gini is the average result from the cross-validation fold that was not used in training. As num_leaves is increased, overfitting sets in and the train Gini increases while the test Gini decreases. In this case the difference is between test and train performance was substantial, and the training Gini increases to above 90%, indicating significant overfitting.

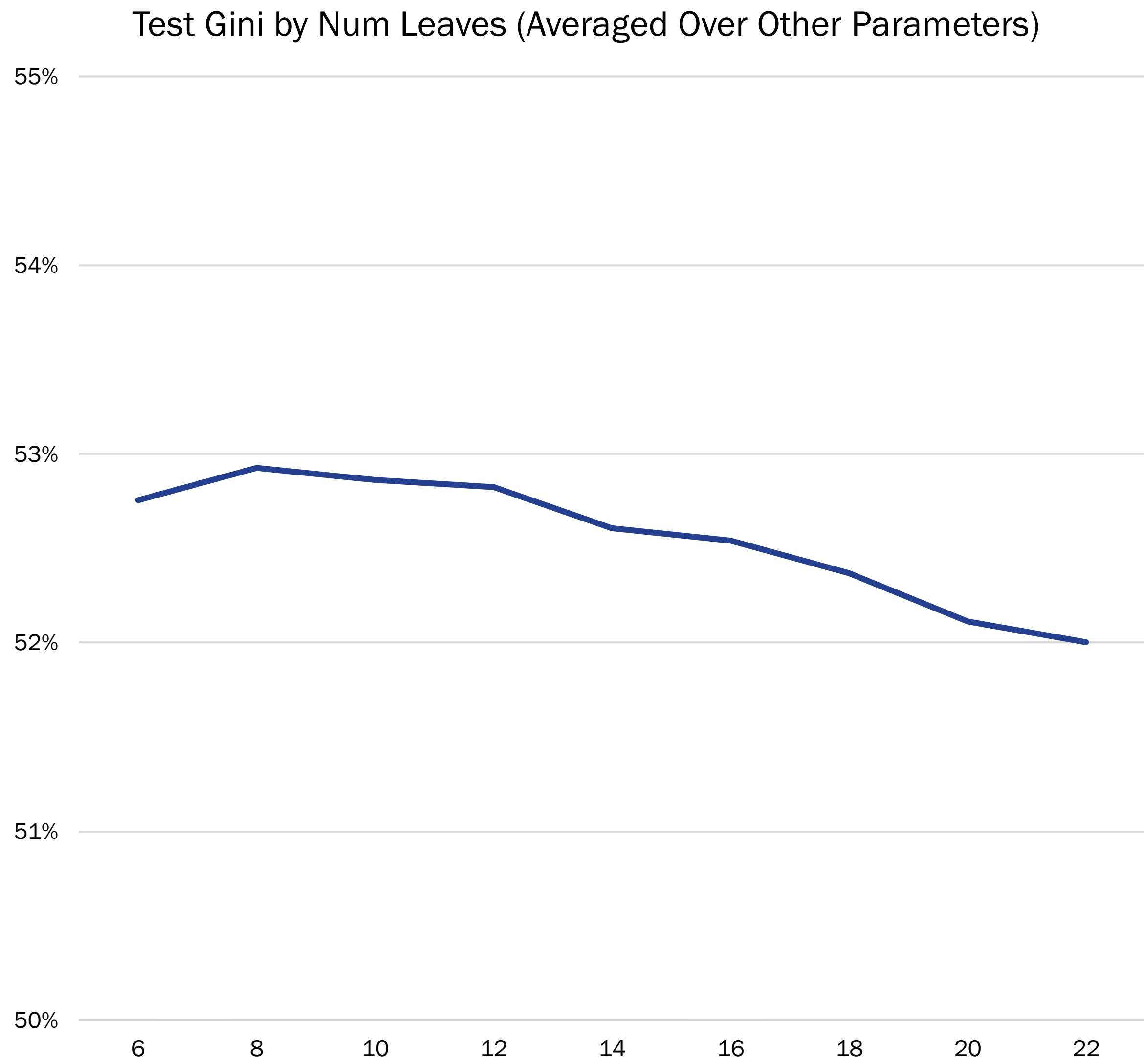


Another way to view the relationship between hyperparameters and performance is shown in the following chart, where each point represents a set of hyperparameter values (there were 1440 in this grid search), the performance has been moved to the x-axis, and the hyperparameter value is plotted against it. A moving average has been added to view the general trend in the hyperparameter value against performance.

In this chart, the best models are on the right and the less performant models on the left. The trend for num_leaves is quite strong and shows that less complex models tend to be better. However, the best 5% of models all have num_leaves either 9 or 15. The very simplest models, with num_leaves = 3, have mid-range performance. This result provides a strong indication that the refined parameter search should concentrate around 9 <= num_leaves <= 15.



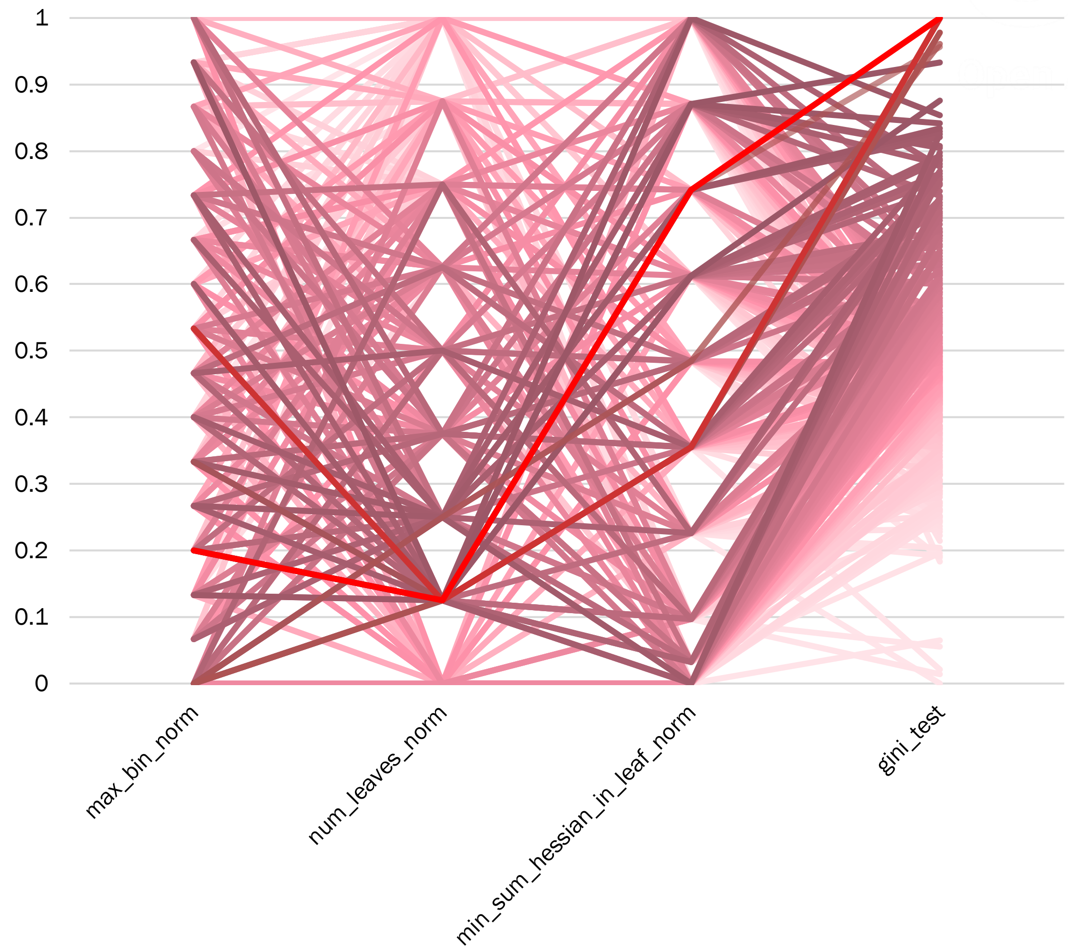
A refined search was conducted over a smaller area around the optimal parameters from the course search. The following chart showing a refined search over num_leaves suggests num_leaves = 8 as the optimal value, but it's not necessarily the case, because all the hyperparameters must be considered simultaneously.



One way to simultaneously consider all hyperparameters being tuned is to use a parallel coordinate plot. This aligns all three hyperparameter dimensions (or however many are being searched) and the performance measure. The dimensions are linearly transformed so they range between 0 and 1 and can be easily compared.

In the chart below, each line represents a set of hyperparameters. The performance measure is on the right, where better performance is at the top of the chart. The lines are coloured such that more performant models are deeper blue, but the top 5 models are coloured red so they can be easily seen. This shows that the best models tend to have mid to high values of min_sum_hessian_in_leaf, num_leaves = 9 or 15, and low to mid values of max_bin.

Parallel Coordinate Chart - Red is Best, Dark Blue is Good



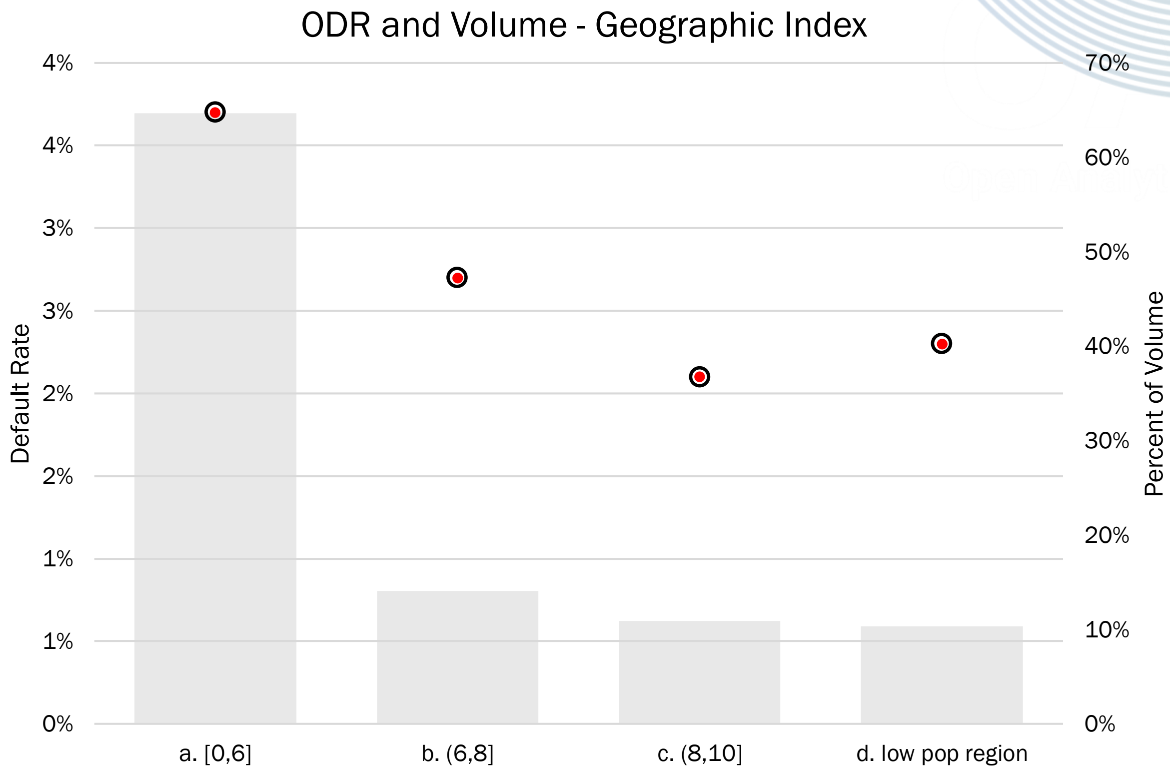
Perhaps the main take-away is that the hyperparameter search can be quite involved and requires effort. Once the hyperparameters have been chosen, training the model is simple and fast, as it is with logistic regression.

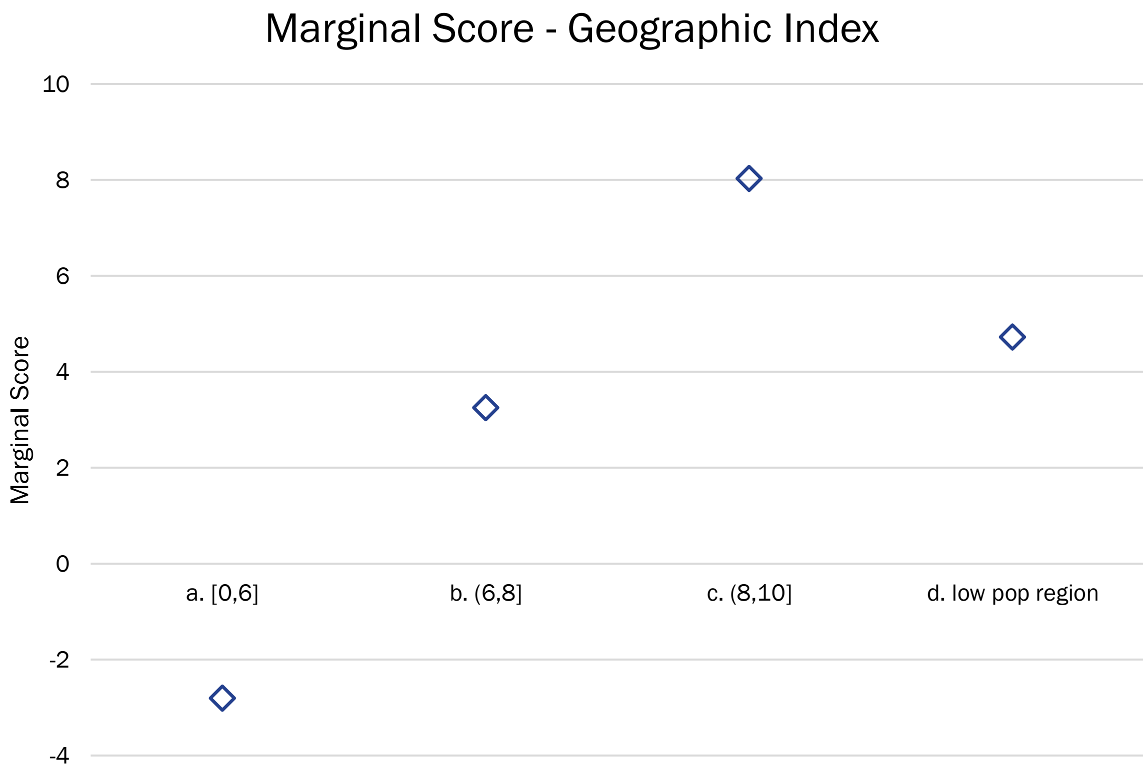
Model Transparency

Dependence on Predictors

In logistic regression, the relationship between the outcome and the predictors is clear. Every bin over every predictor gets a marginal score, these are all summed and added to an intercept to get the log-odds, which can be transformed to PD. As such, the relationship is very clean and explainable.

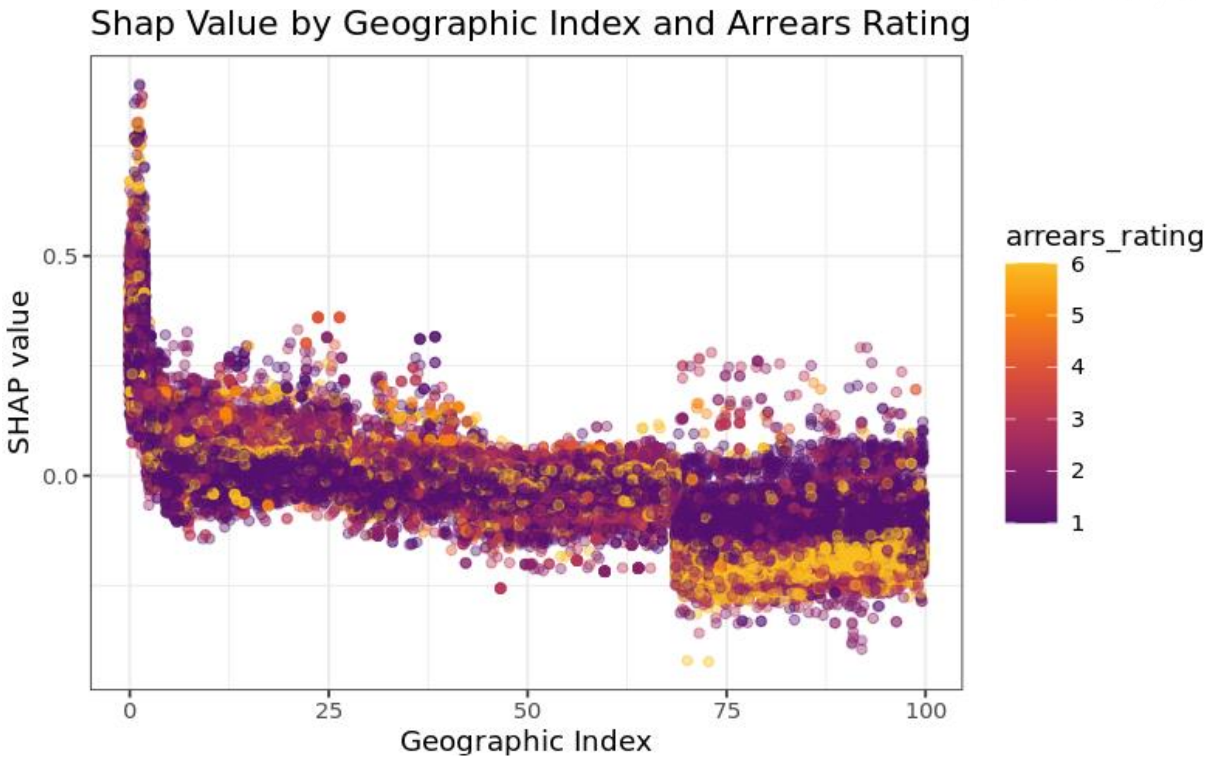
In the following chart, the bins for the geographic index are shown. The chart after that shows the corresponding marginal scores. The trend is intuitive by construction, and it's easy to explain that an entity enjoys a 8-point benefit because it is situated in a low-risk area, or otherwise.



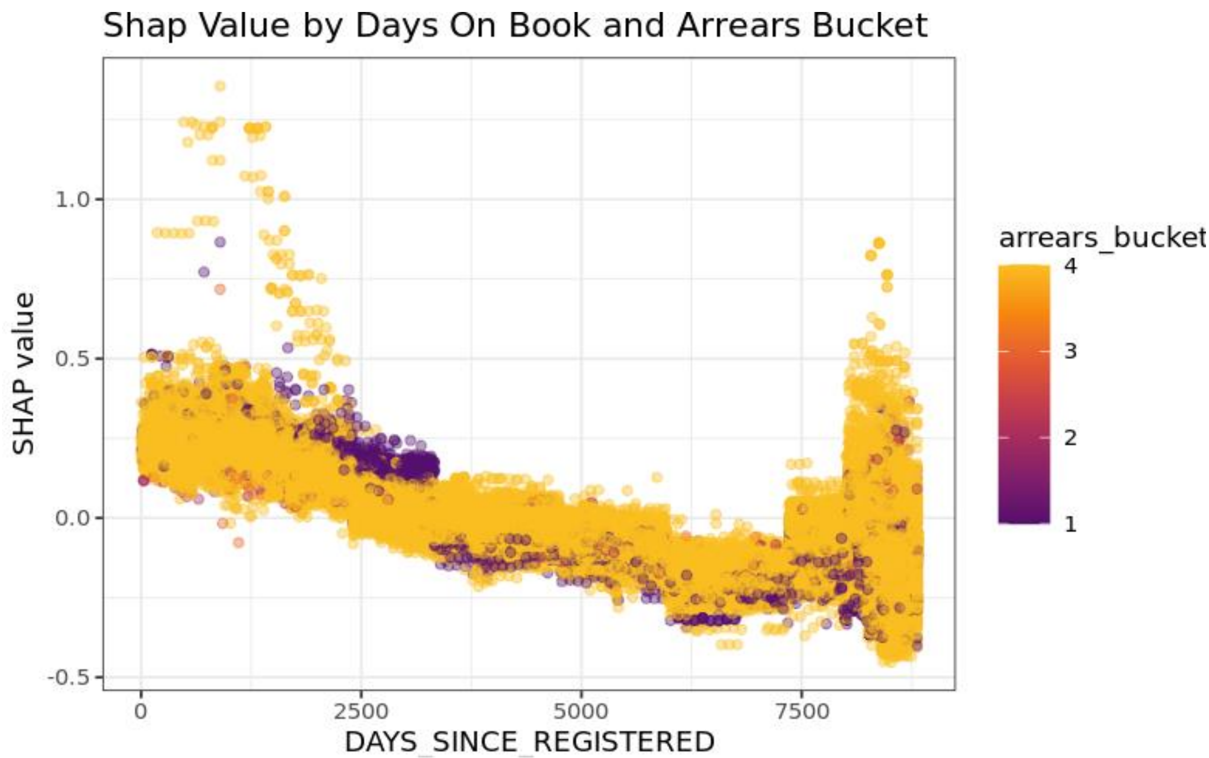


The relationship between the model outcome and the predictors in the machine learning model can be ascertained by viewing partial dependence plots, which are one aspect of the very informative Shapley analysis. The following chart shows the partial dependence plot for geographic index.

This relationship is much harder to explain. It's clear that lower risk regions do have lower predicted default rates (lower SHAP values), but the relationship is complex and involves interactions with the other predictors. Arrears rating is shown in this partial dependence plot by the colour, but that relationship is also difficult to explain. High arrears sometimes appears to increase the PD compared to the general trend, and sometimes decrease it.



The following chart shows another example with age of business in days. It has a similar level of explainability.



Model Performance

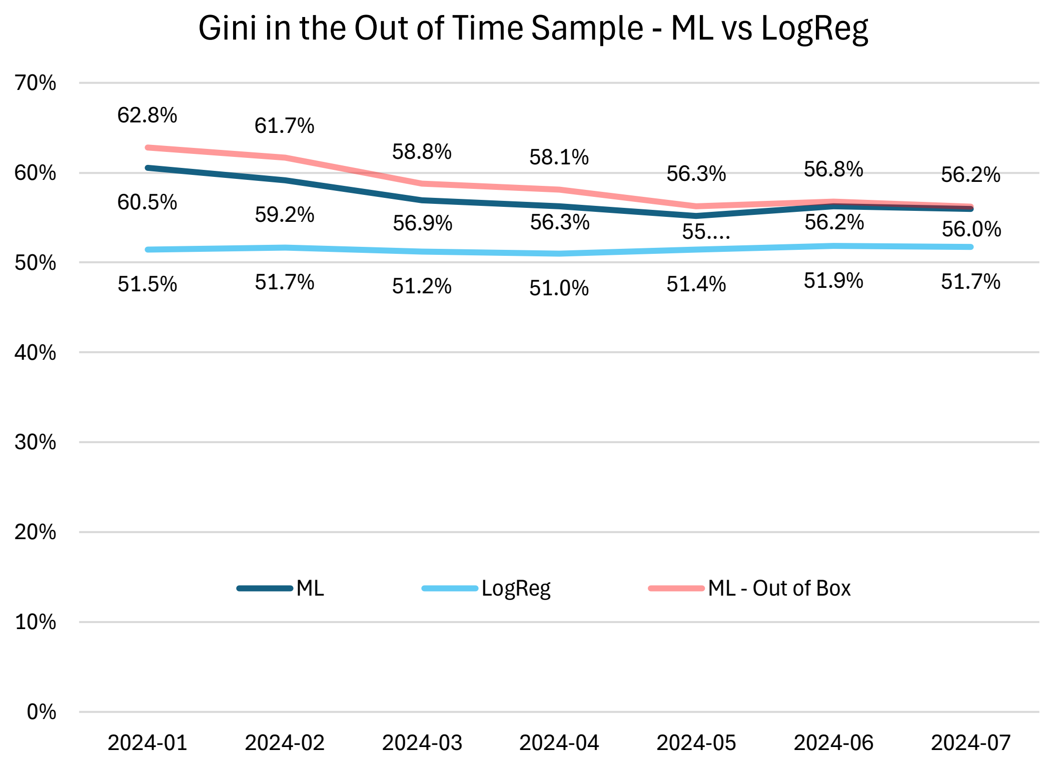
The Gini coefficient was used to measure model performance. The Gini coefficients mentioned in the hyperparameter search were the cross-validation test results. Here we present the out-of-time test results.

As expected, the gradient boosting method outperforms logistic regression by several Gini points. The chart below shows three performance time series:

1. ML: the machine learning model with hyperparameters specified,
2. LogReg: the logistic regression model,
3. ML – Out of Box: machine learning model without hyperparameters specified.

Despite a careful hyperparameter search, the machine learning Gini decreases over time in the out-of-time sample, while the logistic regression Gini is stable. With a seven month out of time sample, it is possible the decrease is just statistical fluctuation, but it does appear to be due to overfitting. At the very least, close monitoring of the model would be required, and potentially the need for a refit after a year.

The out-of-the-box model is even more overfit, with a greater deterioration in performance over the out of time sample.



The table below compares the average Gini in the out of time sample, the level of deterioration, and the Gini at the end of the sample.

Model	OOT Average	OOT Deterioration (Over 7 Months)	OOT End (7 months OOT)
Machine Learning	57.0%	-4.5%	56.0%
Logistic	52.0%	+0.2%	51.7%
Δ (ML – Logistic)	+5.0%	-4.7%	+4.3%
ML out of the box	58.7%	**-6.6%**	56.2%

Conclusions

We have looked at the model build, the level of transparency, and the performance of logistic regression and LightGBM.

Ease of construction is about a tie – it depends on the objectives and level of experience of the modeller. If the data is familiar and the modeller has a good understanding of the business objectives, the binning for logistic regression can be done relatively quickly. If multiple iterations are required, however, this part of the project can drag out.

For LightGBM, the ease of construction depends on the effort the modeller is prepared to expend on hyperparameter tuning. If the team is willing to use out-of-the-box parameters, fitting the model requires little effort. However, this can lead to over-fitting, which may lead to more work later on, in the form of monitoring and model re-fits.

Model transparency easily goes to logistic regression. The logistic regression model is perfectly understood, and the modeller has a lot of control over the effects of each predictor. In contrast, ML models provide little avenue for control and the outcomes can be difficult to interpret. Even though SHAP analysis provides tools to visualise the effects of the predictors, it doesn't answer the question of why they should have their effects.

Model performance favours machine learning by around 4 Gini points. However, great care must be taken to ensure the ML model isn't overfit. If it is overfit, the performance will quickly deteriorate. Logistic regression, on the other hand, is difficult to overfit with the number of predictor bins and rows of data typically seen in credit risk projects.

About Open Analytics

Open Analytics is an Australian financial services consulting firm dedicated to modernising credit risk management through advanced data science and risk modelling. The company specialises in predictive modelling, automated credit processes, and risk quantification to ensure clients are ahead in an evolving financial landscape.

Contact Details

For any questions regarding Open Analytics products and consulting services, please email us at:

contact@open-analytics.com.au

www.open-analytics.com.au

Open Analytics