praelexis credit

WHITEPAPER

Features of the Praelexis Credit Toolkit

Features of the Praelexis Credit Toolkit

Internationally, banks are regulated by the three BASEL accords with the latest, BASEL III, dating from November 2010. The main take-away for our purposes is that it requires banks to have a minimum amount of common equity and minimum liquidity ratio based on the risk exposure of the bank. In practice this means, among other things, that the requirements can be relaxed if a bank can more accurately estimate its risk exposure. This encourages the development of more accurate credit granting and/or risk assessment models.
During the typical credit modelling process a number of disparate systems and tools are typically used by members of the technical, as well as by the management and business teams. In order to navigate these tools, a substantial amount of specialised knowledge is expected from the user. Disjoint tools, each performing a very specific task during the credit model development add to the complexity of the process. If the integration of the different steps is not seamless, the modelling process becomes tedious, time-consuming and error-prone.
The flexibility of Praelexis Credit allows state-of-the-art machine learning techniques to be combined with the tried and tested credit modelling procedures. Superior credit models are created seamlessly and reliably by following best-practice, model life-cycle management. Reliable software development and credit modelling practices are incorporated into the platform which allows, for example, the effortless generation of auditing documentation.

The Praelexis Credit Toolkit allows you to:

- Enhance bureau scores by quickly and cost-effectively incorporating your own data
- Build custom credit scorecards interactively
- Incorporate non-standard data such as transactions, text, or data that only your organisation has access to
- Securely deploy your credit model for business use at scale

Integrated, Flexible, Powerful

The credit modelling solution offered by Praelexis Credit includes a Jupyter Lab notebook utilising an underlying core credit library. The notebook is developed with the following main design principle in mind:

An integrated environment providing maximum flexibility as well as ease-of-use for the credit modeller and manager.

This is achieved by utilising the power of the Python ecosystem exposed through a Jupyter Lab notebook. Since development takes place in the Python environment, the modeller is not restricted to any prescribed modelling solution. For example, should the modeller want to develop a model based on a deep neural network, he/she can build it on top of TensorFlow, Keras, PyTorch, or any other framework by simply installing and importing it into the notebook.
At the same time, the Praelexis Credit solution offers a complete toolset for developing a credit model, based on Weight-Of-Evidence (WOE) and logistic regression. This approach should be familiar to most, if not all, credit modellers.

The notebook consists of the following sections:

- Data Loading and Preprocessing
- Target Definition and Roll Rates
- Initial Variable Selection
- Manual Binning
- Variable Reduction
- Scorecard Development
- Model Performance and Stability Checks
- Model and Client Summaries
- Deployment on a Kubernetes cluster

> *In the rest of this Whitepaper we discuss the functionality of the Praelexis Credit Toolkit according to these modules.*

Data Loading and Preprocessing

Data Loading

The flexibility of the notebook solution allows the user to connect to any data source.

The data sources differ between various organisations. Apart from an initial setup, this can be fully automated. In the notebook, an example is given how to load an example dataset from Azure Blobstore.

Preprocessing

Preprocessing of data is a standard requirement for any machine learning project.

Fortunately there are excellent libraries available to aid the modeller. Sklearn, for example, offers the following modules:

- Standardisation, or mean removal and variance scaling
- Non-linear transformation
- Normalisation
- Encoding categorical features
- Discretisation
- Imputation of missing values
- Generating polynomial features
- Custom transformers

Roll Rate Analysis and Target Definition

The credit modelling process utilises information about existing clients of the institution and the first important objective of the credit modeller is to determine the distinction between a 'good' and 'bad' client. This decision typically involves a number of roll-players including the credit modeller and management, as well as considerations such as compliance standards.

Roll rate analysis assists in this decision making process.

The basic idea is the following:

- Accumulate the data of the client/loan base, from a specified **starting date** for a specific number of months on the books (12 months for example).,
- Each loan is put in a bucket based on the days-past-due (the number of days behind payment). The buckets can typically be '0 days', '1-30 days', '31-60 days', '61-90 days' and '90+ days'. Incidentally, these are the default buckets used in the code.
- After an additional period, for example another twelve months, these loans are examined again. At this time the loans in the different buckets are explored to investigate to which extent they migrated to other buckets over the specified period. This allows the modeller to determine the probability of a loan, for example, in the '61-90 day' bucket, to recover (move to '0 days').
- Loans that have a low probability of repayment will naturally be classified as 'bad'. The main objective of Roll Rate analysis is to determine an appropriate definition of 'bad'. Based on the analysis this decision typically involves the modeller as well as other stakeholders.

This information is summarised in Figure 1

Initial Variable Selection

Modellers are usually provided with many unsuitable variables. An initial automatic Weight-Of-Evidence (WOE) transformation therefore discards variables that do not contribute based on the Information Value and the Gini index.

In this section, the initial automatic WOE transformation is performed and individual variables that do not contribute to the model, are discarded. The decision of whether or not to discard a variable is based on the Information Value (IV) and the Gini

Figure 1: Roll Rates

index of the variable. Before calculating these variable statistics, one has to first ensure that the variables are transformed in the same way as they will be for the final model. To do this, the following steps are performed:

- Perform auto-binning
- Estimate the WOE transformation
- Apply the transformation to the dataset
- Calculate the Information Value (IV) for each variable
- Select an initial set of variables according to their IV's.
- Calculate the Gini indices for the selected variables

Calculate the WOE Transformation

The weight-of-evidence transformation is a nonlinear transformation of the original variables. In this section a brief summary of the WOE is given.

Purpose of the WOE transformation

- To increase the discriminative power of the variables and identify variables with the highest discriminative power (i.e. those variables that are best suited to distinguish between good and bad loans).
- To provide stability for the model.

Summary of WOE

In the case of a continuous variable, it is first converted to a categorical variable through binning. A term that is often encountered is 'buckets', which has the same meaning as 'bins'. Each bin or bucket is then assigned a WOE value that is calculated from the client database. Note that these WOE values for each variable are fixed. Any change in these values require a retraining of the model.

Specifying the bin boundaries is important and below (see Figure 4) a tool is described to help with the fine adjustment. In order to get the process going an auto-binning procedure is used.

Once the buckets are fixed, the continuous variable has essentially been converted to a categorical variable with the number of 'categories' equal to the number of bins. From hereon there is no distinction between categorical and continuous variables.

Details of the calculation

The roll rate analysis above leads to a definition of 'good' or 'bad'. Each client in the database is therefore assigned a good (0), or bad (1), label. The WOE for each variable is calculated according to,

$$\text{WOE} = \ln\left(\frac{\text{distribution of good}}{\text{distribution of bad}}\right).$$

Example

As an example consider the following variable for which the buckets are already known. 'Count good' and 'Count bad' are obtained from the database of clients. Since a good/bad label is available for each client, the buckets for the specific variable entry for each client can be determined. These are simply added for each bucket, giving the good/bad count, as illustrated in Table 1.

	Bucket	Boundaries	Count good	Fraction good	Count bad	Fraction bad	WOE
	1	$(-\infty, 1]$	1760	0.0973	798	0.2033	-0.37
	2	(1, 2]	5238	0.2896	1 223	0.3116	-0.07
	3	(2, 3]	7881	0.4357	1 034	0.2634	0.50
	4	$(3, \infty)$	3210	0.1775	870	0.2217	-0.22
Total			18 089	1.0	3 925	1.0	

Table 1: Calculating the WOE

The Information Value (IV)

The IV of a variable is an indication of that variable's ability to separate between good/bad loans.

IV range	**Interpretation**
< 0.02	Not predictive
$[0.02, 0.1)$	Weak predictive
$[0.1, 0.3)$	Medium predictive
$[0.3, 0.5)$	Strong predictive
> 0.5	Suspicious

Table 2: IV Interpretation

Mathematically the IV is defined by,

$$\text{IV} = 100 * \sum_{i=1}^{L} (\text{fraction good} - \text{fraction bad}) \times \ln\left(\frac{\text{fraction good}}{\text{fraction bad}}\right),$$

where the sum is taken over all bins L.

The interpretation of the IV is shown in Table 2.

Figure 2 shows the IV's for a number of typical variables, ordered according to decreasing IV.

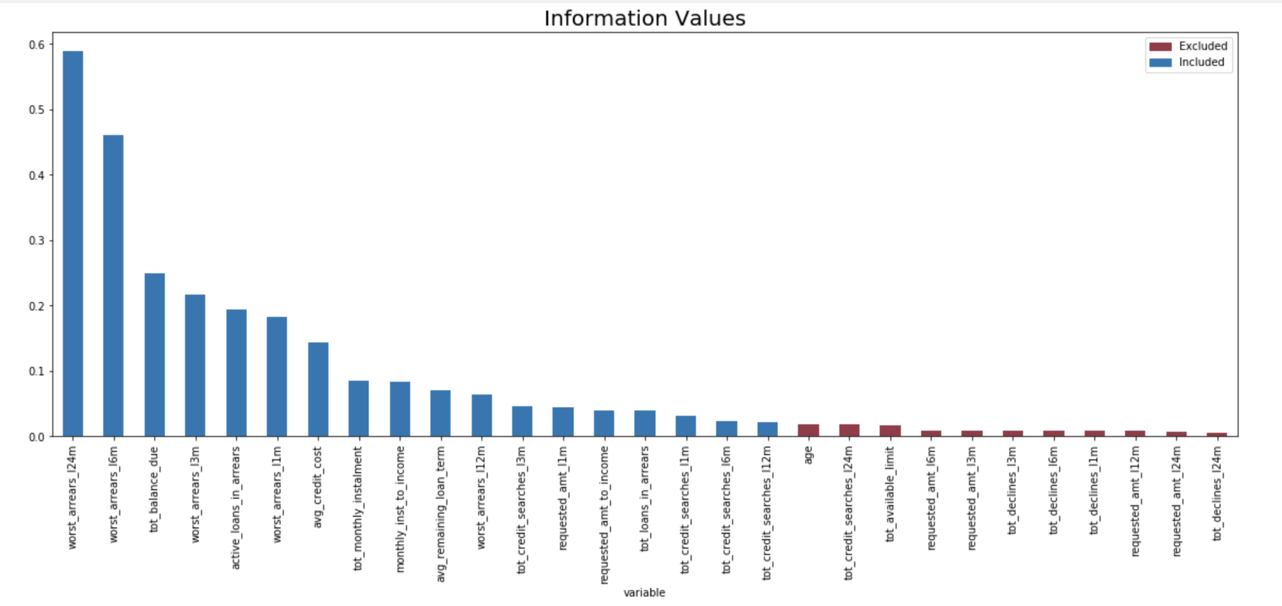

Figure 2: Information Value

From this graph the modeller can make an initial variable selection by choosing variables with information values above a chosen threshold.

For each of those selected variables, the Gini coefficient can be calculated. Although the Gini coefficient is perhaps better known as a measure of the economic inequalities in a country, in general it is a measure of how well the model distinguishes between two classes. In this application, between 'good' and 'bad' clients. It is therefore a measure of the efficacy of an *individual* variable for prediction.

Figure 3 shows the Gini values for a number of variables, selected according to their IV values from Figure 2.

Manual Binning

WOE and, by implication, the IV and Gini depend on the binning. It is therefore standard in credit modelling to manually adjust bin boundaries and a convenient tool is provided in the core library.

Note that, as the name indicates, this is a manual process. Something that is not feasible to do for a large number of variables. The reason behind the initial variable selection is to reduce the number of variables to something that is manageable. It should also be noted that the modeller keeps control and there may be reasons, perhaps based on the experience of the modeller, that would cause the modeller to discard or include specific variables.

The tool allows the user to:

- Manually adjust (by using the widgets in this module) the binning by either splitting or merging existing bins, and adjusting the bin boundaries.
- Watch how different statistics change throughout the process.

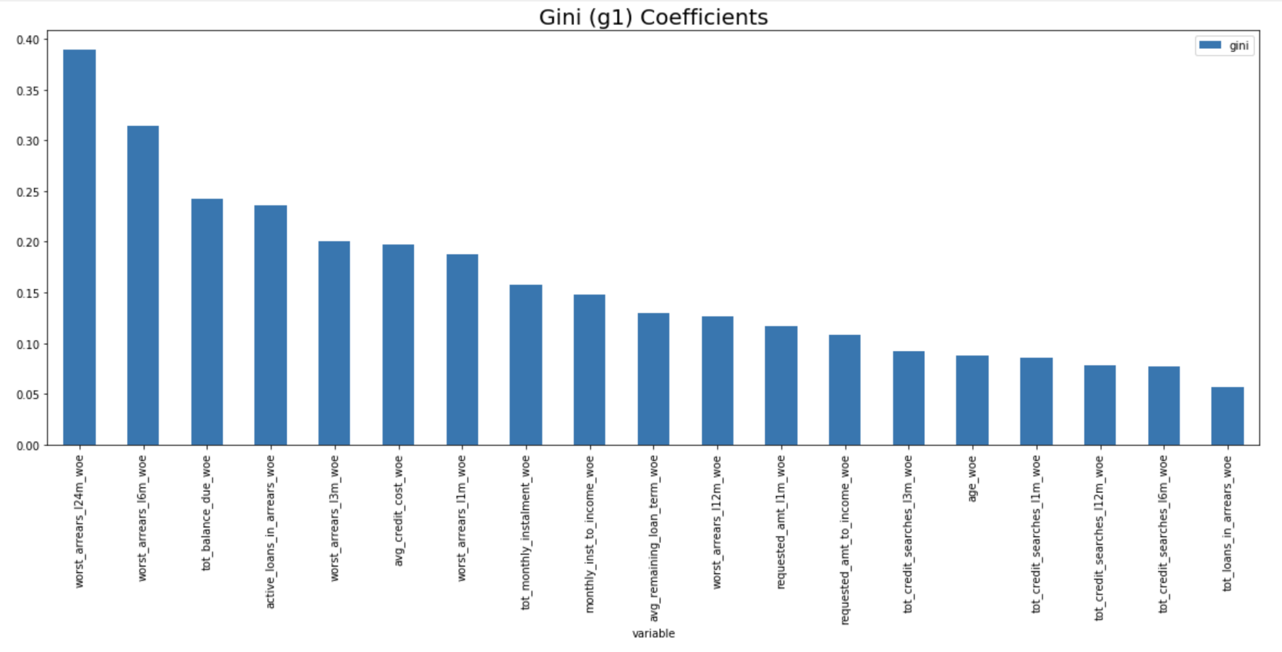

Figure 3: Gini coefficients

Figure 4 provides a view of the manual binning tool.

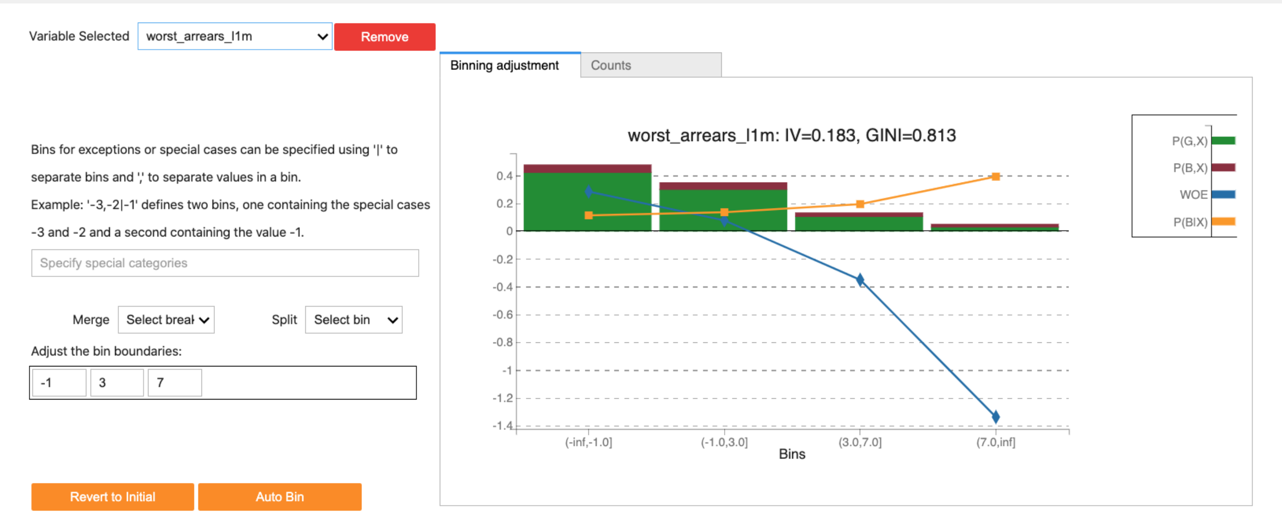

Figure 4: Manual binning

In order to determine the stability of a variable one can compare the WOE for the train and validation sets as shown in Figure 5.

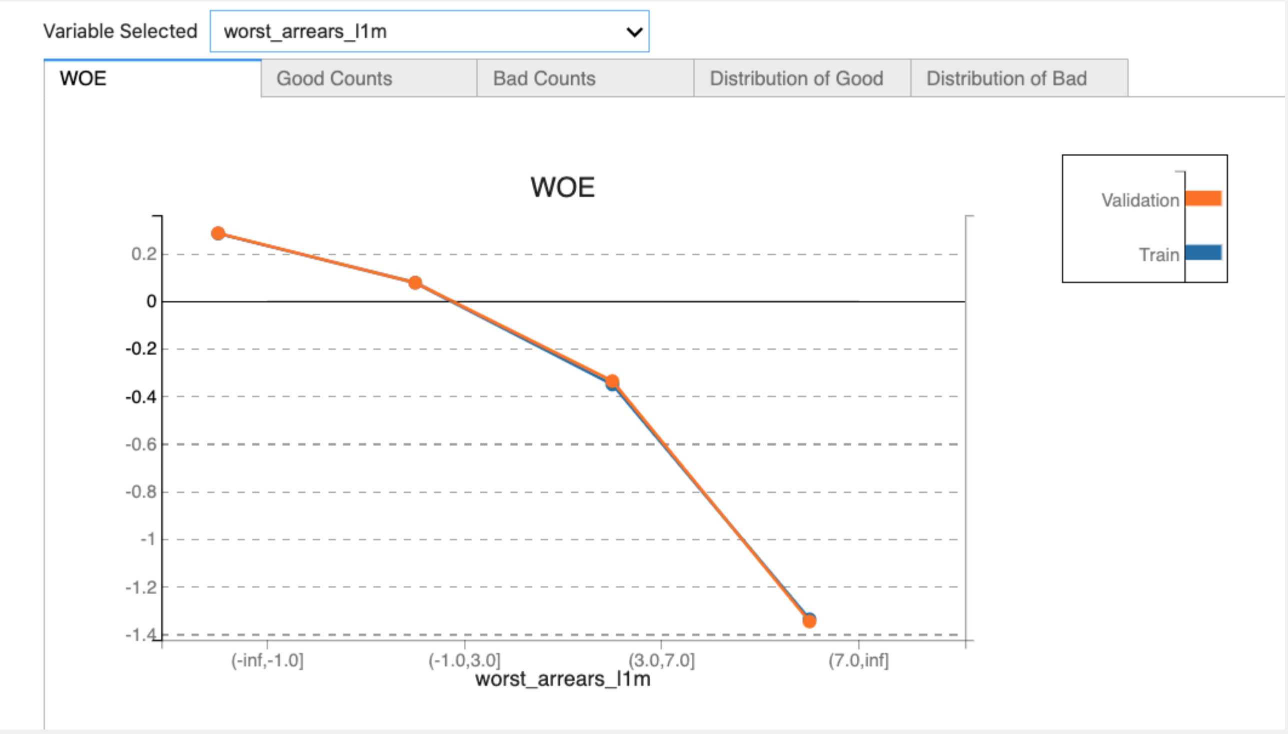

Figure 5: Comparison between training and validation sets

For this particular variable there is a negligible difference between the train- and validation sets.

Variable Reduction

The initial variable selection was based on the the ability of individual variables to distinguish between the two classes of loan, 'good' or 'bad'. In practice, however, a model is built using a combined set of variables. Some models, including logistic regression, are prone to instabilities if *highly correlated* variables are included.

During variable selection the set of variables is further reduced but now based on the correlation between the variables.

Correlation

The correlation between variables is calculated and the variables are clustered according to their correlations – correlated variables are are grouped into the same cluster. The most useful variables can then be selected from each cluster and the reduced set will be used to train a logistic regression model.

The clusters of correlated variables are displayed in the Figure 6 and more explicitly described in Table 3.

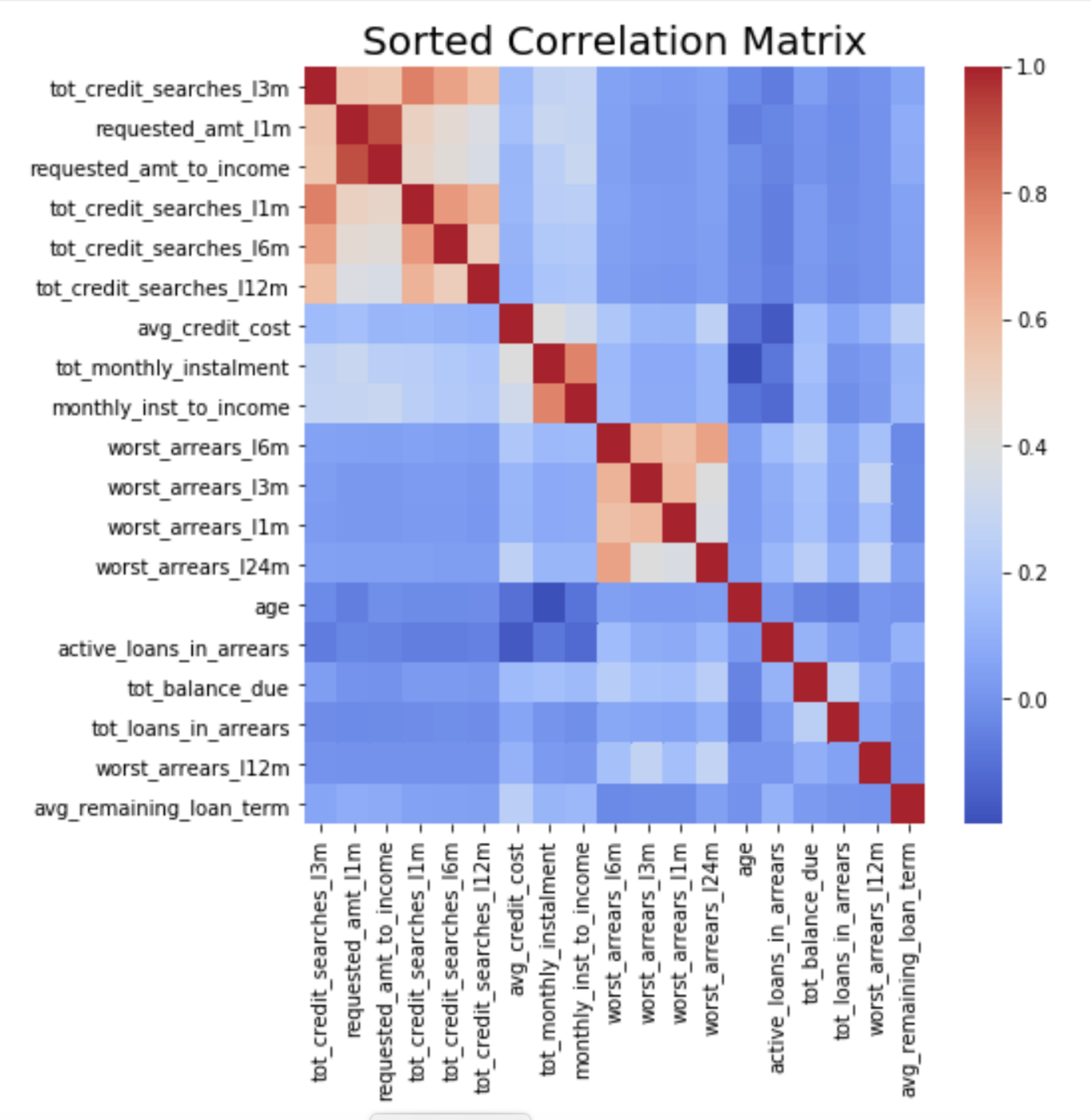

Figure 6: Clusters of correlated variables

Selection

The 'best' variable from each cluster should now be selected as representative of that cluster. This choice can be based on the Gini index or the Variable Inflation Factor (VIF), shown in the Figure 7. This 'final set' of variables is used to train the logistic regression model.

Note:

- Although the term 'final set' is used above, it should be emphasised that this is an iterative procedure where the credit modeller may want to review the selected variables and make adjustments as is deemed appropriate.
- Of course, the modeller also has the choice to stay with the automated procedures of the notebook!

The correlation values of the variables with their VIF's are shown in Figure 7. The correlation values within a user-determined range are highlighted.

Clusters

Cluster index	Variables	Cluster index	Variables
1	requested_amt_l1m	3	worst_arrears_l1m
	requested_amt_to_income		worst_arrears_l3m
	tot_credit_searches_l1m		worst_arrears_l6m
	tot_credit_searches_l3m		worst_arrears_l24m
	tot_credit_searches_l6m	4	age
	tot_credit_searches_l12m	5	active_loans_in_arrears
2	avg_credit_cost	6	tot_balance_due
	tot_monthly_instalment	7	tot_loans_in_arrears
	monthly_inst_to_income	8	worst_arrears_l12m
		9	avg_remaining_loan_term

Table 3: Correlation clusters

	VIF	requested_amt_l1m_woe	avg_credit_cost_woe	worst_arrears_l24m_woe	tot_balance_due_woe	tot_loans_in_arrears_woe	worst_arrears_l12m_woe	avg_remaining_loan_term_woe	active_loans_in_arrears_woe
requested_amt_l1m_woe	1.032986	1.000000	0.165260	0.044057	0.003955	-0.028091	-0.003857	0.086648	-0.038696
avg_credit_cost_woe	1.251608	0.165260	1.000000	0.259469	0.148003	0.060896	0.107260	0.244644	-0.166929
worst_arrears_l24m_woe	1.227790	0.044057	0.259469	1.000000	0.240078	0.097832	0.276180	0.043889	0.127159
tot_balance_due_woe	1.143892	0.003955	0.148003	0.240078	1.000000	0.248150	0.096284	0.025040	0.114166
tot_loans_in_arrears_woe	1.069505	-0.028091	0.060896	0.097832	0.248150	1.000000	0.054660	0.008257	0.036306
worst_arrears_l12m_woe	1.086942	-0.003857	0.107260	0.276180	0.096284	0.054660	1.000000	-0.000615	0.011328
avg_remaining_loan_term_woe	1.097235	0.086648	0.244644	0.043889	0.025040	0.008257	-0.000615	1.000000	0.109248
active_loans_in_arrears_woe	1.106388	-0.038696	-0.166929	0.127159	0.114166	0.036306	0.011328	0.109248	1.000000

Figure 7: Variable Inflation Factor

Scorecard Development

Having selected the variables to be used in the model, it is straightforward to build a logistic regression model, as described below.

The model outputs a probability, i.e. a number between 0 and 1. These numbers are not calibrated and therefore not directly interpretable. For example, how does a value of 0.7 compare with 0.6? For this reason, it is customary to transform the probability to a scorecard value. In order to do this, the modeller specifies:

- A scorecard value that corresponds to a specific bad/good odds. For example, the modeller may decide that a bad/good odds of 1.0/10.0 should correspond to a scorecard value of 500.
- How much the scorecard value should increase to double the odds. This allows the user to more effectively interpret how much better a scorecard value of, say, 600 is than 500. Typically, doubling the odds (good/bad) corresponds to an increase of 50 in the scorecard.

Logistic Regression Model

Logistic regression is a linear model of features which themselves can be a nonlinear transformation of the variables. For the credit model, the features are the WOE, a nonlinear transform of the original variables.

Each feature is divided into different buckets with a WOE value assigned to each bucket. Let's indicate the WOE values by x_i. The logistic regression model for predicting the probability of default p is then given by,

$$p = \sigma\left(\sum_{i=1}^{n} w_i x_i + w_0\right),$$

where the sigmoid function $\sigma(z)$ is given by,

$$\sigma(z) = \frac{1}{1+\exp(-z)}.$$

The weights, w_i, are obtained during the training process, and n is the number of features.

Odds

The odds are defined as,

$$\text{odds} = \frac{p}{1-p},$$

where p is given above. An easy calculation shows that the log-odds is given by

$$z := \log(\text{odds}) = \sum_{i=1}^{n} w_i x_i + w_0.$$

Score

We transform the log-odds to a score by,

$$\text{score} = f * z + \gamma,$$

where the scaling f and the offset γ need to be determined. To calculate these parameters, two equations are needed. For the first equation, the user specifies the odds that is mapped to a given base score. This gives,

$$\text{score} = \log(\text{odds}) * f + \gamma.$$

For the other equation, one specifies what an increase in the score means. Typically, the increase is the number of points that doubles the odds (pdo). This gives us,

$$\text{score} - \text{pdo} = \log(2 * \text{odds}) * f + \gamma.$$

Here it is important to note the sign of pdo. Since the odds are defined as bad/good, a doubling of the odds should *decrease* the score. Therefore,

$$f = -\text{pdo}/\log(2),$$

and

$$\gamma = \text{score} - \log(\text{odds}) * f.$$

These equations allow one to convert the probability output of the logistic regression model to a score that is easy to interpret.

Model Performance and Stability Checks

Evaluating, testing and monitoring the performance of a credit model are crucial for the institution using the model. One simply cannot afford (literally) that any problems with model go undetected. This section describes some of the tools that can help detecting any issues that might arise. More specifically, the following will be briefly discussed:

- Scaled Stability Index (SSI), calculated over different data sets. The most important of these is out-of-time data sets. These contain samples of loan performance from a time more recent than that used for training.
- Receiver Operating Characteristic (ROC) curve.
- Scorecard distributions.

SSI	Interpretation
< 10	Slight Change
$10 - 20$	Minor Change
> 20	Significant Change

Table 4: Interpretation of the Scaled Stability Index

Scaled Stability Index (SSI)

The stability of the model is assessed by determining whether or not the current population has changed from the model development population. One measure that can be used to determine if a shift has occurred is the Scaled Stability Index (also known as the Population Stability Index). This measures the magnitude of the population shift between the current and development populations. If n is the number of variables, the SSI is calculated for each bucket using,

$$\text{SSI} = 100 \times \sum_{i=1}^{n} (\text{Fraction Expected} - \text{Fraction Actual}) \times \ln\left(\frac{\text{Fraction Expected}}{\text{Fraction Actual}}\right),$$

where Fraction Expected is the fraction of the population that the model predicts to fall in the specific bucket and Fraction Actual is the fraction that is observed to fall in that bucket.

The overall SSI score is the sum of all the values in the different buckets.

The SSI is typically interpreted as shown in Table 4.

Figure 8 shows the output of the calculation comparing two data sets, ‘development’ and ‘recent’.

	development	recent	stability_index	scaled_stability_index
(-inf, 450.0]	0.029323	0.030276	0.000030	0.030452
(450.0, 480.0]	0.063727	0.063936	0.000001	0.000686
(480.0, 510.0]	0.110598	0.105432	0.000247	0.247133
(510.0, 540.0]	0.135996	0.143900	0.000447	0.446517
(540.0, 570.0]	0.141307	0.146215	0.000168	0.167619
(570.0, 600.0]	0.142404	0.144435	0.000029	0.028761
(600.0, 630.0]	0.142057	0.137667	0.000138	0.137824
(630.0, 660.0]	0.126068	0.119323	0.000371	0.370850
(660.0, 690.0]	0.084622	0.082458	0.000056	0.056100
(720.0, inf]	0.002193	0.003206	0.000384	0.384070
total	0.978296	0.976848	0.001870	1.870009

Stability index range	Shift indicator	Action
<10	No shift	Scorecard looks stable, only perform a few checks
10 - 20	Slight shift	Take caution as there may be underlying issuse, more in-depth checks
>20	Possible serious shift	Further investigation

Figure 8: Model performance

Receiver Operating Characteristic (ROC) curves

The standard ROC curves are compared for the training-, validation- and test sets in Figure 9. The Gini values for these sets are also shown for comparison.

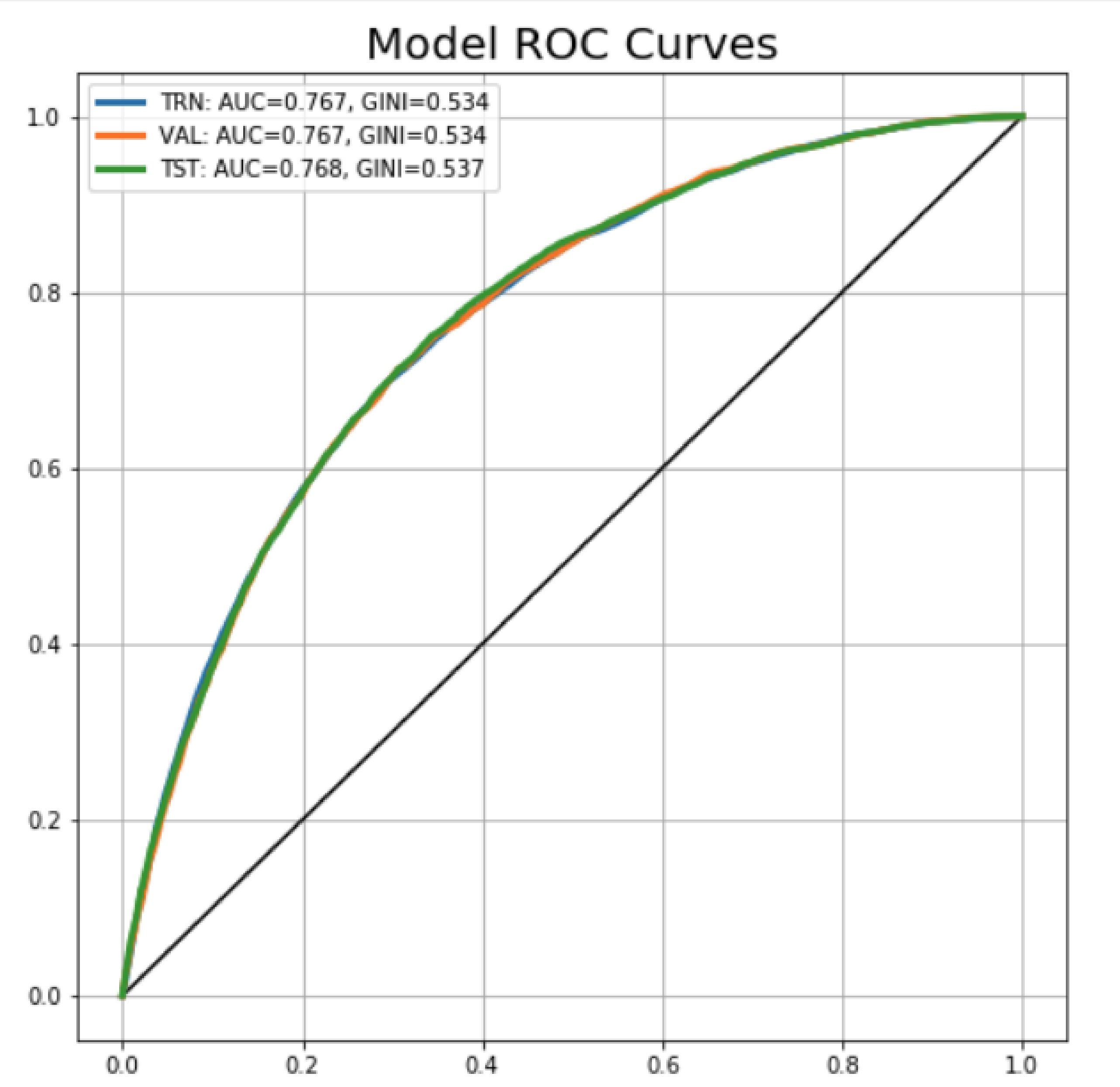

Figure 9: Model ROC curves

Were the test set from an out-of-time sample, a ROC curve like this one would have been very satisfactory as it indicates negligible difference between performance on the different data sets.

Scorecard distributions

Monitoring the distribution of the scorecard values for the development data set and the most recently available data is important to verify that there is no significant change in the population. Figure 10 compares the distributions of two different data sets. No significant change in the distributions is detected for these data sets.

Figure 10: Scorecard distribution

Model Characteristics Report

Clients are awarded a total score by the model, where the total score is the sum of the scores for each attribute in the model.

Since all the relevant information is available, it is possible to summarise the model in terms of the contributions of all the variables used by the model. This allows the model to be interpreted easily and can also, for example, assist in detecting any anomalies in the variables.

It is also possible to do the same thing for an individual client. In this case the contributions of the different variables to the total client score is directly calculated.

Model Summary

The model is summarised in Table 5.

In order to understand Table 5 it is necessary to recall how the score is calculated. In the earlier, Scorecard Development section, the offset f and scale factor γ are calculated. With these values available, a summary of the model is obtained by rewriting the equation for the score as,

$$\begin{aligned} \text{score} &= f * \left(\sum_{i=1}^{n} w_i x_i + w_0 \right) + \gamma \\ &= \sum_{i=1}^{n} (f w_i) * x_i + (f w_0 + \gamma). \end{aligned}$$

Two things to keep in mind:

- For each attribute (bucket) of each variable we have a pre-determined value for the WOE, as learned from the data. This means that for the equation above, we have a value for x_i for each of the attributes of variable i.
- From training the model, we have values for all the weights $w_i, i = 0, \ldots, n$. These weights are scaled by the scale factor and the *scaled_weights* shown in Table 5 are therefore given by $\widehat{w}_i = f w_i$.

This means that we can determine in advance how many points each attribute will contribute to the total score, as shown in Table 5.

GINI Summary

gini: TRN	gini: VAL	gini: TST
0.543680	0.544066	0.544248

variable	buckets	woe_values	scaled_weights $\widehat{w}_i$	base_points	relative_points	points	bad_rate
age	(-inf,24.0]	-0.28	95.5	88	-26	62	0.18
age	(24.0,26.0]	-0.07	95.5	88	-6	82	0.15
age	(26.0,30.0]	0.01	95.5	88	1	89	0.14
age	(30.0,50.0]	0.12	95.5	88	11	99	0.13
age	(50.0,58.0]	-0.01	95.5	88	0	88	0.15
age	(58.0,inf]	-0.18	95.5	88	-17	71	0.17
requested_amt_l1m	(-inf,500]	0.18	69.1	88	12	100	0.12
requested_amt_l1m	(500,2500]	0.03	69.1	88	2	90	0.14
requested_amt_l1m	(2500,4000]	-0.08	69.1	88	-5	83	0.16
requested_amt_l1m	(4000,11000]	-0.16	69.1	88	-11	77	0.17
requested_amt_l1m	(11000,inf]	-0.41	69.1	88	-28	60	0.20
avg_credit_cost	(-inf,100]	0.35	40.0	88	13	101	0.11
avg_credit_cost	(100,300]	-0.33	40.0	88	-13	75	0.19
avg_credit_cost	(300,inf]	-0.74	40.0	88	-29	59	0.26
avg_remaining_loan_term	(-inf,5]	-0.22	45.7	88	-9	79	0.17
avg_remaining_loan_term	(5,20]	0.52	45.7	88	23	111	0.09
avg_remaining_loan_term	(20,30]	0.13	45.7	88	5	93	0.13
avg_remaining_loan_term	(30,inf]	-0.15	45.7	88	-6	82	0.16
tot_balance_due	(-inf,10]	0.37	43.4	88	16	104	0.10
tot_balance_due	(10,110]	-0.01	43.4	88	0	88	0.15
tot_balance_due	(110,inf]	-0.75	43.4	88	-32	56	0.26
active_loans_in_arrears	(-inf,1.0]	-0.46	65.5	88	-30	58	0.21
active_loans_in_arrears	(1.0,2.0]	0.10	65.5	88	6	94	0.13
active_loans_in_arrears	(2.0,5.0]	0.32	65.5	88	20	108	0.11
active_loans_in_arrears	(5.0,7.0]	0.54	65.5	88	35	123	0.09
active_loans_in_arrears	(7.0,inf]	0.74	65.5	88	48	136	0.07
tot_loans_in_arrears	(-inf,1.00]	0.06	26.4	88	1	89	0.14
tot_loans_in_arrears	(1.00,2.00]	-0.62	26.4	88	-16	72	0.24
tot_loans_in_arrears	(2.00,inf]	-0.71	26.4	88	-18	70	0.26
worst_arrears_l12m	(-inf,-5]	0.22	5.6	88	1	89	0.12
worst_arrears_l12m	(-5,5]	0.12	5.6	88	0	88	0.13
worst_arrears_l12m	(5,25]	-0.05	5.6	88	0	88	0.15
worst_arrears_l12m	(25,inf]	-0.61	5.6	88	-3	85	0.24
worst_arrears_l24m	(-inf,-8.0]	0.23	55.8	88	12	100	0.12
worst_arrears_l24m	(-8.0,1.0]	0.78	55.8	88	43	131	0.07
worst_arrears_l24m	(1.0,5.0]	-0.05	55.8	88	-2	86	0.15
worst_arrears_l24m	(5.0,inf]	-1.06	55.8	88	-59	29	0.33

Table 5: Model summary

Client Score

905

Client Summary

variable	feature_values	buckets	woe_values	base_points	relative_points	points	bad_rate
age	35.0	(30.0,50.0]	0.12	88	11	99	0.13
requested_amt_l1m	0.00	(-inf,500]	0.18	88	12	100	0.12
avg_credit_cost	106.59	(100,300]	-0.33	88	-13	75	0.19
avg_remaining_loan_term	45.00	(30,inf]	-0.15	88	-6	82	0.16
tot_balance_due	0.00	(-inf,10]	0.37	88	16	104	0.10
active_loans_in_arrears	9.00	(7.0,inf]	0.74	88	48	136	0.07
tot_loans_in_arrears	0.00	(-inf,1.00]	0.06	88	1	89	0.14
worst_arrears_l12m	-11.00	(-inf,-5]	0.22	88	1	89	0.12
worst_arrears_l24m	1.00	(-8.0,1.0]	0.78	88	43	131	0.07

Table 6: The client sunmmary

The columns in Table 5 have the following meaning:

- variable: The names of the variables used in the model.
- buckets: The different buckets (attributes) for the variable.
- woe_values: The weight-of-evidence for the different buckets.
- scaled_weights: The model weights w_i multiplied by the scaling factor f. Note that there is one value per variable.
- base_points: These are $(f * w_0 + \gamma)/n$
- relative_points: The product of woe_values and scaled_weights $((f\, w_i) * x_i)$. These are the relative contributions of each bucket to the overall score.
- points: The sum of the relative contribution and the base_points. Each account that needs to be scored will be assigned a point value for each variable, depending on the bucket specific to that account. The sum of the point values from each variable determines the total score of the account.

Client Summary

Table 6 shows how the overall score of 435 for this client is broken down into contributions from the different variables.

Deployment

For a model to be useful it has to be put into production, the model preformance needs to be monitored and the relevant information has to be readily accessible.

The discussion above focused on aspects of the core functionality of the Praelexis Credit Toolkit with emphasis on the way the core library is accessed from a Jupyter notebook.

These notebooks are hosted on secure cloud infrastructure and accessible over the internet with no setup required.

Once a model has been developed and signed off by the responsible entity, the model has to be put into production. For the Praelexis Credit Toolkit this implies three things:

- Deployment of the model on a Kubernetes cluster where the model is made available through an API. There is no reason for the credit modeller to have any interest in this process and the Praelexis Credit Toolkit automates the process.

- Monitoring. The importance of early detection of any shifts in the population distribution (Covid 19 is an extreme example) has already been emphasised. This is an ongoing process and the Praelexis Credit Toolkit provides automated procedures for monitoring the behaviour of the model.
- Reporting. The credit model generates a wealth of information that should be made available to all stakeholders. The Praelexis Credit Toolkit provides a dashboard where all the relevant information is readily accessible. This functionality provides crucial information that, among other things, can be used for decision making on a management level.

The Future of Credit Granting

By using Praelexis Credit's unified platform, credit modellers will be joining the AI and machine learning revolution. Praelexis Credit simplifies and streamlines the deployment process, decreasing the time to delivery on credit models. Current credit modelling solutions are often less flexible for developer-minded practitioners, less cost effective and do not always support good software development practices. Praelexis Credit encompasses a range of modelling techniques. It encourages fast model development, supports swift and stable deployment, and provides an end-to-end solution to credit management. Model development is intuitive with user-friendly notebooks and widgets, assisting both technical and non-technical users. To discuss your credit modelling needs, or for a live demo, contact us at credit@praelexis.com.