Models interpretation

Name: Lúcia Moreira

Date: 02/05/2020

Summary

This project deals with the exploratory use of the interpretML library, recently made publicly available for models interpretation. A binary classification problem is addressed for predicting loan repayment. Logistic Regression (LR), Explainable Boosting Classifier (EBM, from the interpretML), Random Forest (RF), Naïve Bayes (NB) and Neural Networks (NN) were considered and interpreted using the interpretML library. LR and EBM were elected for the glassbox tool while RF, NB and NN for the blackbox tool. Despite Naïve Bayes is a linear model, it is not available in the glassbox tool. However, as any model can be used in the blackbox, Morris sensitivity analysis for LR and NB was also considered for these linear models. It was not possible to use NN with any of the interpretML library available tools. Results show that using the AUC criterion Logistic Regression, EBM, Naïve Bayes and NN are the best predicting models with NN slightly better. But taking into account the analysis performed with interpretML, the Morris Sensitivity shows that a better convergence between the LR and NB methods is obtained for Logistic Regression. Based on the overall findings and explainability, LR was the selected model from the interpretML tool. Moreover, the new available tool is quite user friendly and a helpful tool for model interpretation. Despite disappointing when using neural networks, where could indeed bring a huge advantage for interpretation of such models. However, maybe possible lack of skills from the author may also be related with the misuse of intrepretML with NN.

Dataset

The proposed dataset concerns predicting loan repayment. It is a public dataset made available to us by our tutor and accordingly retrieved from LendingClub.com, a website that connects borrowers and investors over the Internet. The available dataset represents 9 578 3-year loans that were funded through the LendingClub.com platform between May 2007 and February 2010.

In the lending market, investors provide loans to borrowers in exchange for the promise of repayment with interest. If the borrower repays the loan, then the lender profits from the interest. However, if the borrower is unable to repay the loan, then the lender loses money. Therefore, lenders face the problem of predicting the risk of a borrower being unable to repay a loan.

The binary dependent variable notFullyPaid indicates that the loan was not paid back in full (the borrower either defaulted or the loan was "charged off," meaning the borrower was deemed unlikely to ever pay it back). To predict the dependent variable, the following 13 independent variables available to the investor when deciding whether to fund a loan will be used:

1) creditPolicy: 1 if the customer meets the credit underwriting criteria of LendingClub.com, and 0 otherwise.
2) purpose: The purpose of the loan (takes values "credit_card", "debt_consolidation", "educational", "major_purchase", "small_business", and "all_other").
3) intRate: The interest rate of the loan, as a proportion (a rate of 11 % would be stored as 0.11). As judged by LendingClub.com, riskier borrowers are assigned higher interest rates.
4) installment: The monthly instalments ($) owed by the borrower if the loan is funded.
5) logAnnualInc: The natural log of the self-reported annual income of the borrower.
6) dti: The debt-to-income ratio of the borrower (amount of debt divided by annual income).
7) fico: The FICO credit score of the borrower.

8) daysWithCrLine: The number of days the borrower has had a credit line.
9) revoBal: The borrower's revolving balance (amount unpaid at the end of the credit card billing cycle).
10) revolUtil: The borrower's revolving line utilization rate (the amount of the credit line used relative to total credit available).
11) inqLast6mths: The borrower's number of inquiries by creditors in the last 6 months.
12) delinq2yrs: The number of times the borrower had been 30+ days past due on a payment in the past 2 years.
13) pubRec: The borrower's number of derogatory public records (bankruptcy filings, tax liens, or judgments).

Despite missing values were small (0.15 %), an imputation process for the missing values was performed when there was a missing value. One-hot encoding was performed to the categorical variable 'purpose' by creating a 1-0 array with one column per categorical value, with 1's indicating rows belonging to that category. After one-hot encoding we have 19 final features.

Analysis of the data set indicated that an unbalanced dataset is observed with 16 % of the instances classified as 1, i.e, loan was not fully paid. Dataset was divided into training (80 %) and test (20 %). The option 'stratify' in 'train_test_split' was used for keeping the same proportion of elements of each target class in the training and test sets. Numerical features were divided into continuous and integer; in order to allow better scaling and further imputing process.

Classification problem modelling

Logistic Regression - GlassBox

Logistic regression provided a test accuracy of 83 % i.e. provided the same as random guess. So accuracy is not a good performance measure for this problem. So it was used instead the area under the ROC curve. Fig. 1 shows the ROC curve and AUC for the logistic regression fitting obtained by the interpretML library (and the same area as the standard ROC curve calculation from the scikitlearn library - not shown). An area under the curve of 0.66(9) can be observed (l2 or l1 penalization gave nearly the same results).

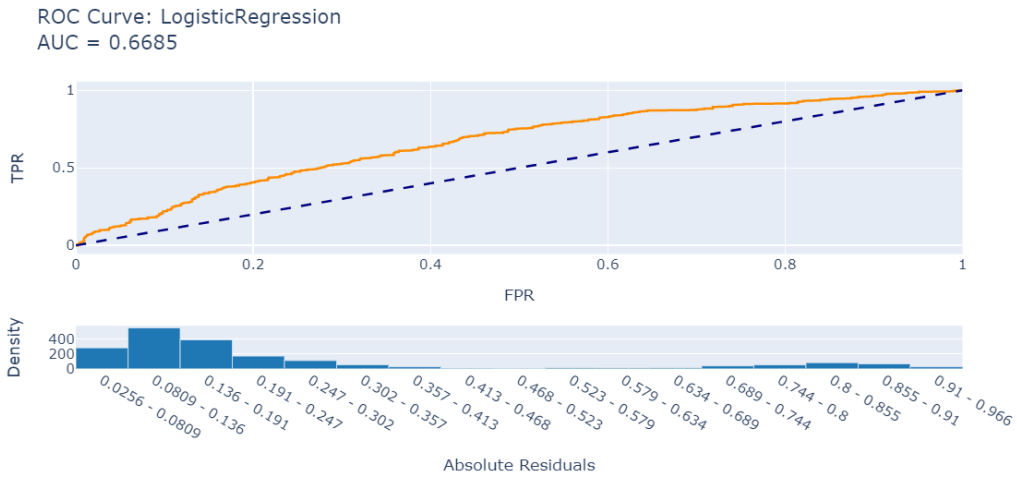

Fig. 1- ROC plot for the Logistic Regression algorithm.

From the interpretML, the following overall normalized magnitude of the coefficients for attributing each of the two classes (Fig. 2) is considered. From the analysis of the plot, one can see that the Logistic Regression Model overall classifieds as notFullyPaid=0 (negative score, blue bars), i.e. the borrower would most probably indeed pay back the full loan, mostly when the purpose of the loan is either for credit card payment or for debt consolidation or if the borrower meets the credit policy. On the other hand, the higher magnitude

coefficients for classifying notFullyPaid=1 (positive score, orange bars) are if the credit is meant to be applied in small businesses, if high instalment values are present, or if the borrower's number of derogatory public records (bankruptcy filings, tax liens, or judgments) is high.

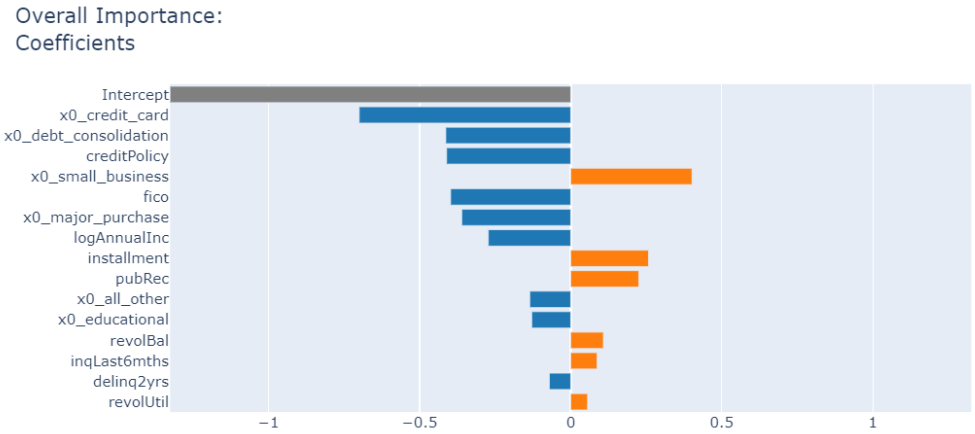

Fig. 2 – Overall normalized coefficients magnitude for the Logistic Regression model from the interpretML library.

Figures 3 and 4 show a few interpretations for some of the test set instances. Four cases are considered, Fig. 3 (left) shows for the notFullyPaid=1 class, in the case where prediction was correct. This classification was majorly based on the borrower's high revolving balance value (amount unpaid at the end of the credit card billing cycle). One can see that this feature was also one of the relevant ones for the 1-type classification, as seen in Fig. 2. In its turn, Fig. 3 (right) shows a correctly predicted 0-type class. The 0-type classification was based on the high value of the self-reported log annual income of the borrower and the high fico value. Fig. 4 shows miss classified instances both for the positive and negative classes. Fig. 4 (left) shows a 0-type misclassified instance (model predicted type 1). The model attributed the prediction based on the quite high borrower's high revolving balance (revolBal) value and other features that the model usually scores the most probably defaulters such as for instance the low fico values and the purpose of the credit for a small business. But indeed the borrower had a quite high annual income, moreover the credit was probably able to improve his/her business and turned out to be able to pay back the loan. Finally, Fig. 4 (right) shows a 1-type misclassification. Model based its prediction on the high fico value and the purpose being for debt consolidation however the annual income of the borrower was very low so probably because of this was not able to pay back the loan.

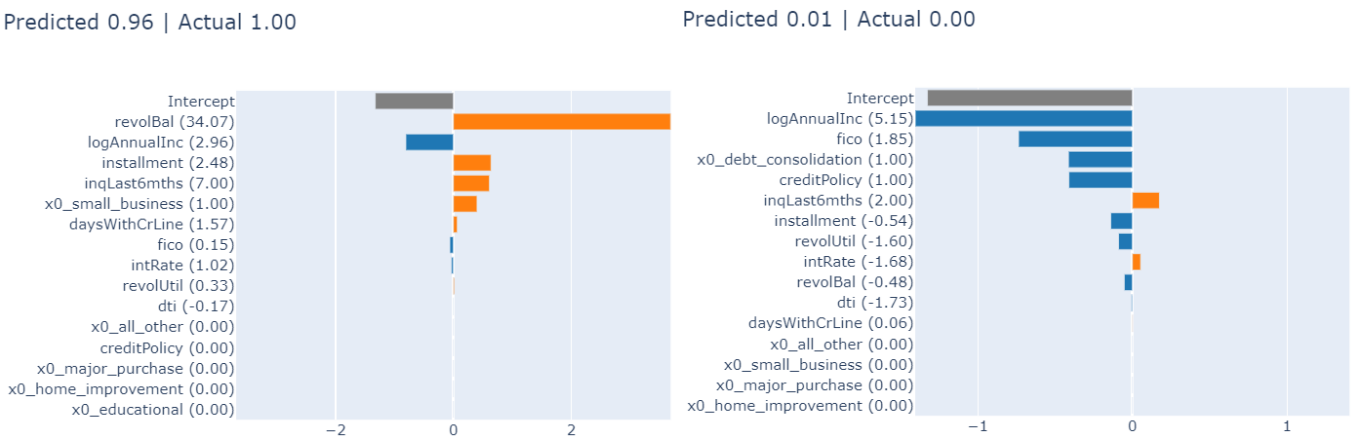

Fig. 3 – Correctly predicted type-1 class from the Logistic Regression model from the interpretML library (left). Correctly predicted type-0 class from the Logistic Regression model from the interpretML library (right).

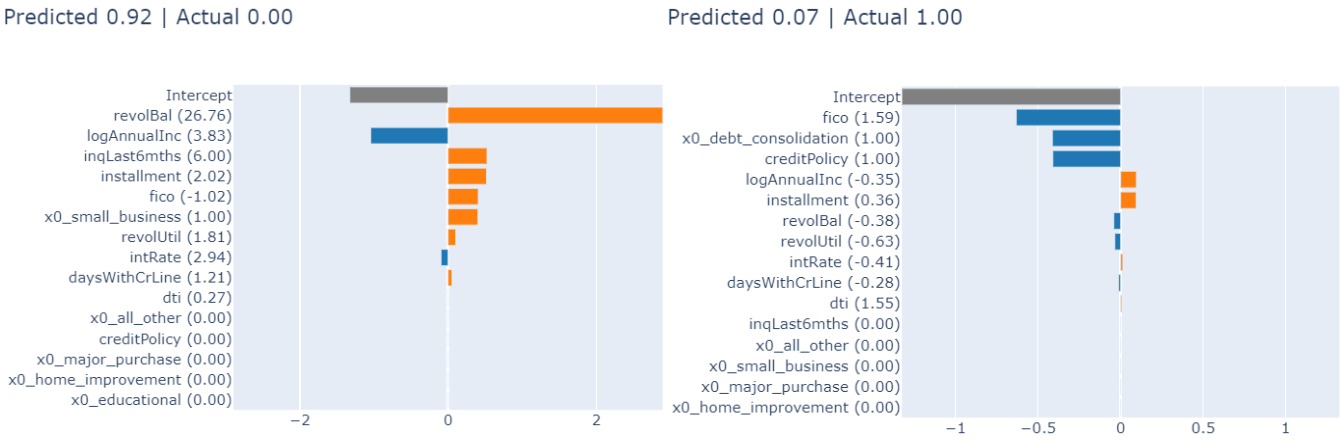

Fig. 4 – Incorrectly predicted type-0 class from the Logistic Regression model from the interpretML library (left). Incorrectly predicted type-1 class from the Logistic Regression model from the interpretML library (right).

Fig. 5 shows the results from the Morris sensitivity from the blackbox tool applied for this algorithm. One can see that the most relevant features are the borrower's number of inquiries by creditors in the last 6 months followed by the annual income and in third the fico value.

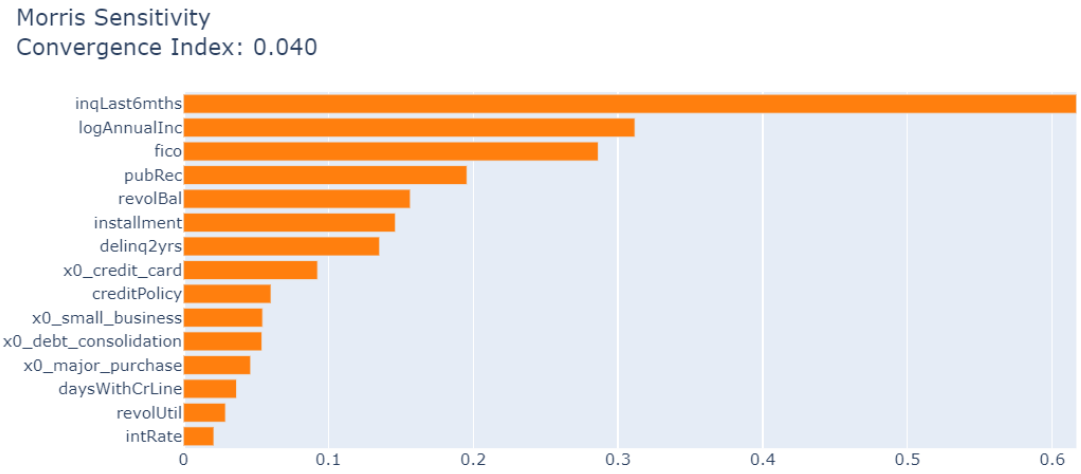

Fig. 5- Morris sensitivity from the interpretML library.

Explainable Boosting Machine - Glassbox

Fig. 6 shows the ROC curve for the EBM fitting available from the interpretML library. EBM is a generalized additive model and is also a glassbox model. It was designed to have accuracy comparable to state-of-the-art machine learning methods like Random Forest and Boosted Trees [https://arxiv.org/pdf/1909.09223.pdf], while being highly intelligible and explainable. An area under the curve of 0.66(9) can be observed, very similar to Logistic Regression.

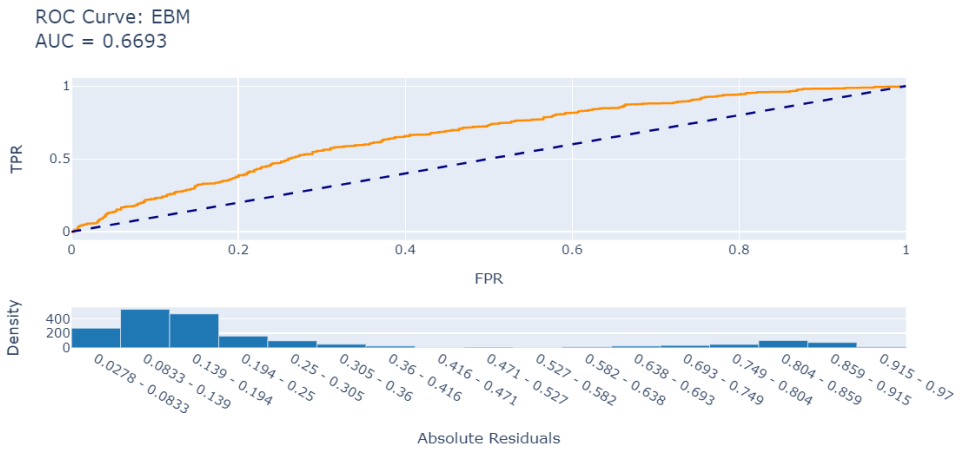

Fig. 6- ROC plot for the EBM algorithm.

From the interpretML, the following overall importance of the features is obtained for the classification (Fig. 7). In EBM, the most relevant features for classification were: the interest rate, first, the borrower's number of inquiries by creditors in the last 6 months, second, followed by the compliance with the credit policy. Comparing with Logistic Regression, one can see for instance that interest rate is not at all an important feature in this classification problem for the Logistic Regression model (Fig. 5) while the second most relevant feature here (*inqLast6mths*- the borrower's number of inquiries by creditors in the last 6 months) appears as first place in the Logistic Regression feature importance (Fig. 5).

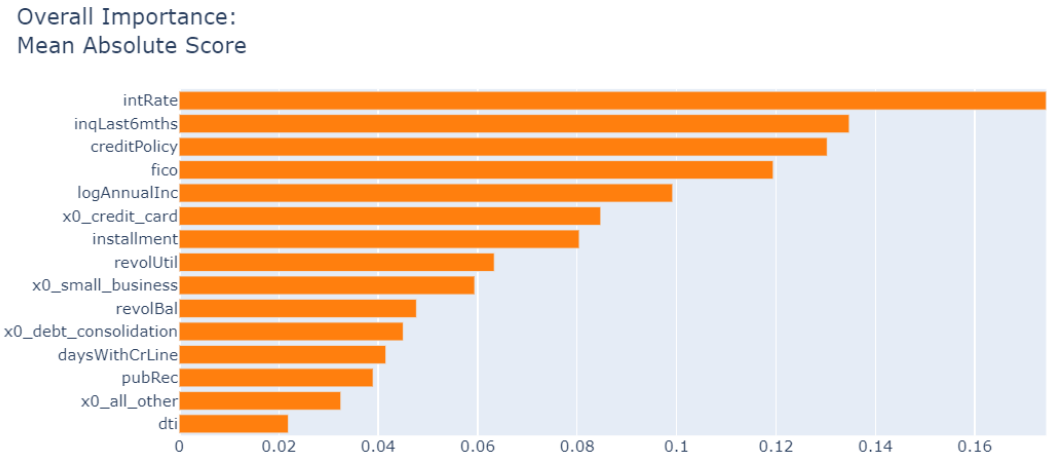

Fig. 7 – Overall variable importance for the EBM model from the interpretML library.

Fig. 8 show how the classification is attributed based on the distribution of the interest rate values and the *inqLast6mths* values. Regarding the interest rate, normalized interest rate lower than -1 are usually classified as 0-type borrower and normalized interest rates higher than 3 are classified as type 1. However, most of the instances lay in the middle range where the discriminatory power of the feature seems to be quite poor. In its turn, the variable related to the borrower's number of inquiries by creditors in the last 6 months (*inqLast6mths*) shows that the model classifies the instance as a type-1 if the *inqLast6mths* normalized value is higher than 3, however the great majority of the instances are below that threshold, so once again this variable may not be that relevant on this classification problem.

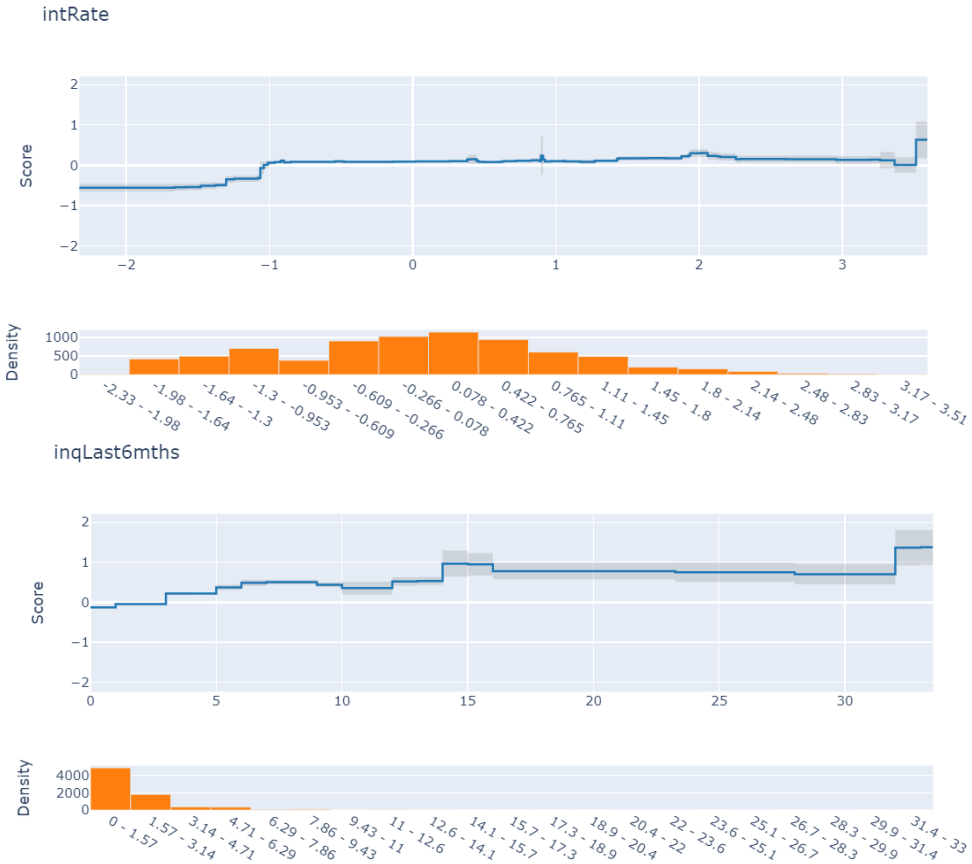

Fig. 8 – *intRate* and *inqLast6mths* variables discrimination power on the EBM model from the interpretML library.

Fig. 9 and 10 show a few interpretations for some of the test set instances. Three cases are considered, Fig. 9 (left) shows for the notFullyPaid=1 class, in the case where prediction can be considered correct. This classification was majorly based on the purpose of the loan being for small business, and the high instalment and interests rate values. In its turn, Fig. 9 (right) shows a correctly predicted 0-type class. The 0-type classification was based on the high fico value, the low interest rate and the purpose of the loan for credit card payment. Fig. 10 shows a miss classified instance for the positive class; this 1-type misclassified instance (model predicted type 0) was based on the purpose of the loan being for credit card and the positive fico value, however probably the class 1 was due to quite low annual income and the model failed to predict that.

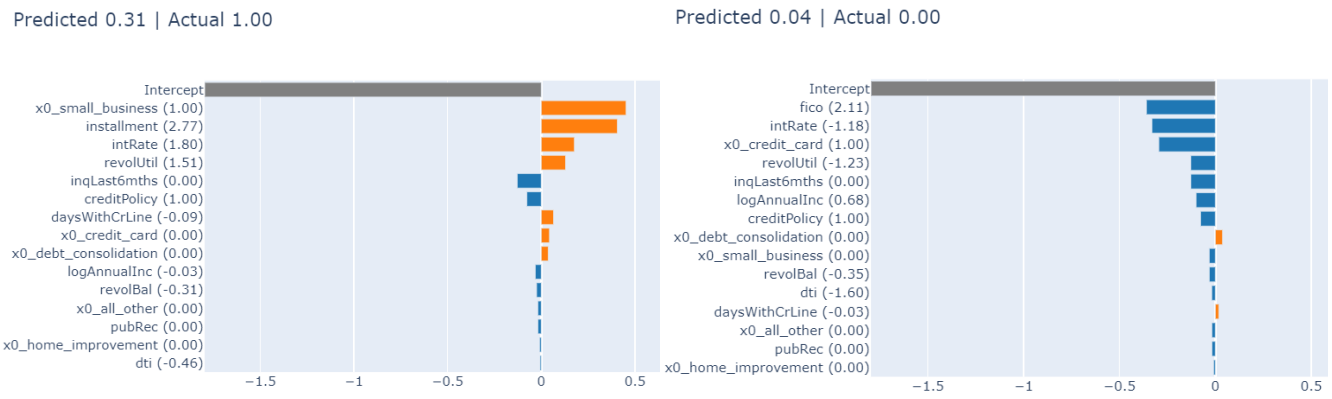

Fig. 9 – Correctly predicted type-1 class from the EBM model from the interpretML library (left). Correctly predicted type-0 class from the EBM model from the interpretML library (right).

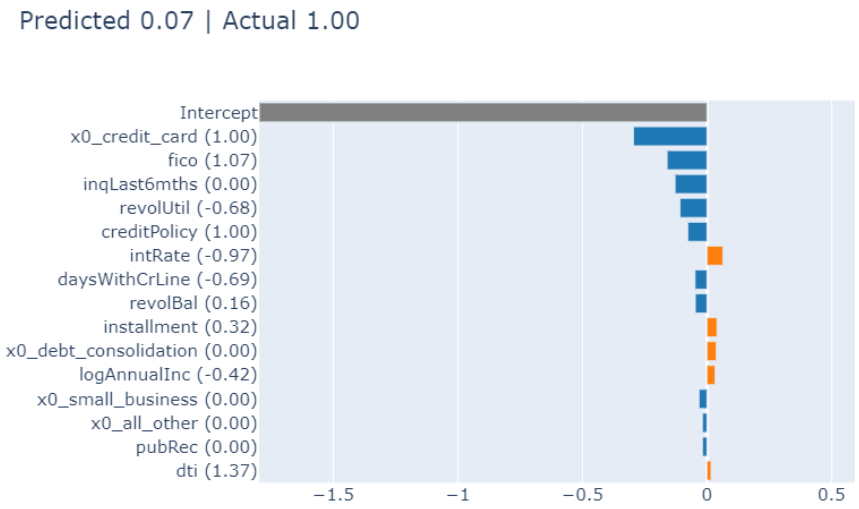

Fig. 10 – Incorrectly predicted type-1 class from the EBM model from the interpretML library.

Moreover, Decision Trees was also considered for the glassbox analysis but the model gave very poor AUC (AUC= 0.55, plot not shown) so further analysis with interpretML was not taken into account.

Following are considered two blackbox models: Random Forest and Naïve Bayes. Despite Naïve Bayes being an explainable model it seems that is not available in glassbox tool from interpretML. Because of that, it was considered a blackbox model because any model can be used with the blackbox tool.

Random Forest - Blackbox

Fig. 11 shows the ROC curve for the RF fitting available in the interpretML library. An area under the curve of 0.65(0) can be observed (equal area is achieved by the scikit-learn library). This AUC value is lower than that for the EBM method and the Logistic Regression method.

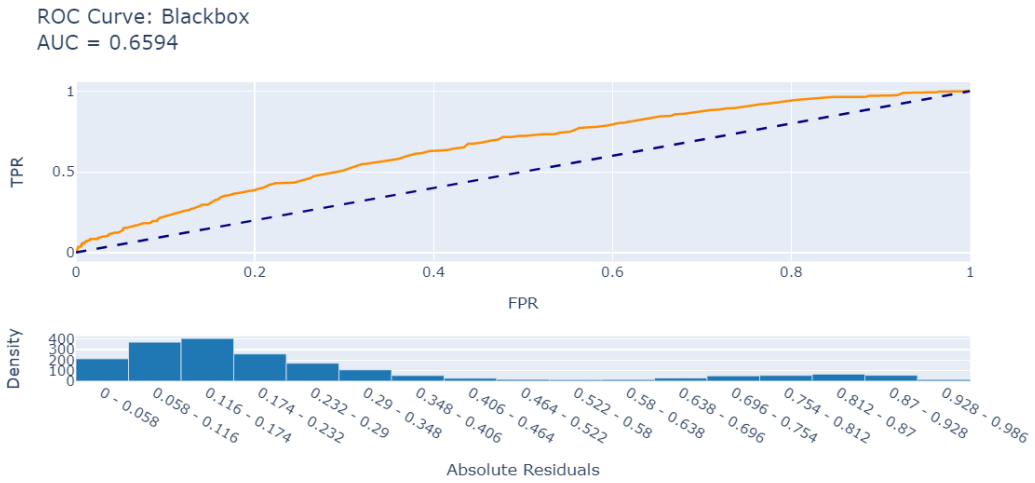

Fig. 11 - ROC plot for the Random Forest algorithm.

Fig. 12 shows the feature importance obtained from the scikitlearn library and the Morris Sensitivity obtained from the interpretML library. In applied statistics, the Morris method for global sensitivity analysis is a so-

called one-step-at-a-time method, meaning that in each run only one input parameter is given a new value. It facilitates a global sensitivity analysis by making local changes at different points of the possible range of input values [https://en.wikipedia.org/wiki/Morris_method]. In its turn, feature importance from scikit-learn is calculated as the decrease in node impurity weighted by the probability of reaching that node. The node probability can be calculated by the number of samples that reach the node, divided by the total number of samples. The higher the value the more important the feature [https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3].

One can see that feature importance in both methods are different. In scikit-learn the four most relevant features for the binary classification are instalment value, the number of days the borrower has had a credit line, the amount unpaid at the end of the credit card billing cycle and the amount of the credit line used relative to total credit available, respectively. While from the interpretML, the Morris method shows that the four most relevant features are the borrower's number of inquiries by creditors in the last 6 months, the fico value, the annual income and the instalment value, respectively.

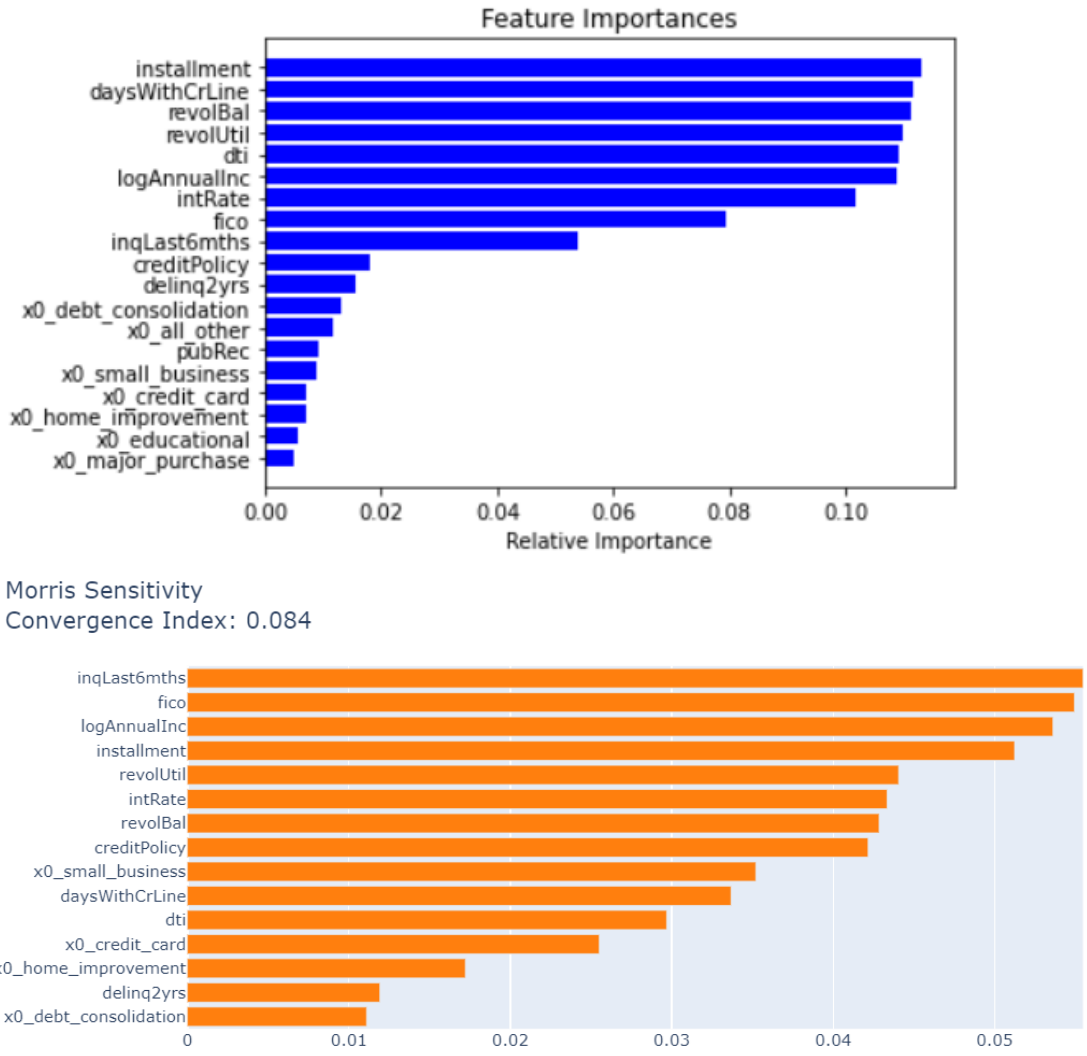

Fig. 12 – Overall variable importance for the Random Forest model from scikit-learn (top) and Morris sensitivity from the interpretML library (bottom).

Naïve Bayes - Blackbox

As mentioned before despite Naïve Bayes is a linear classifier apparently is not available in the glassbox tools from the interpretML library. Because of that the blackbox tool was used to interpret the Naive Bayes classifier. Compared to logistic regression, Naive Bayes classifier is generative model while logistic regression is a discriminative model. The Naive Bayes is a linear classifier using Bayes Theorem and strong independence condition among features; Naive Bayes assumes that the features are conditionally independent [https://medium.com/@sangha_deb/naive-bayes-vs-logistic-regression-a319b07a5d4c]. This, in the finance setting, maybe a very strongly assumption once finance features are usually strongly correlated. Fig. 13 shows the ROC curve for the NB fitting from the interpretML library. An area under the curve of 0.65(9) can be observed (equal area is achieved by the scikit-learn library). This AUC value is also very similar to the one obtained by the Logistic Regression method and EBM.

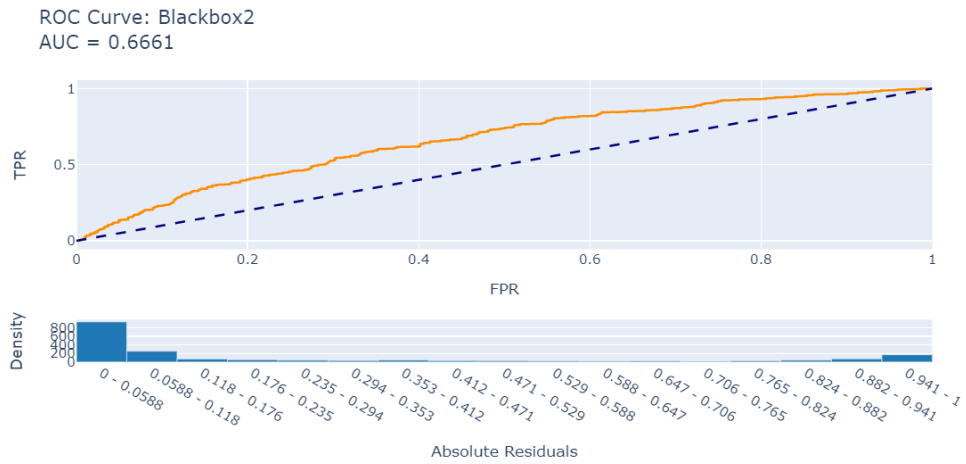

Fig. 13- ROC plot for the Naïve Bayes algorithm.

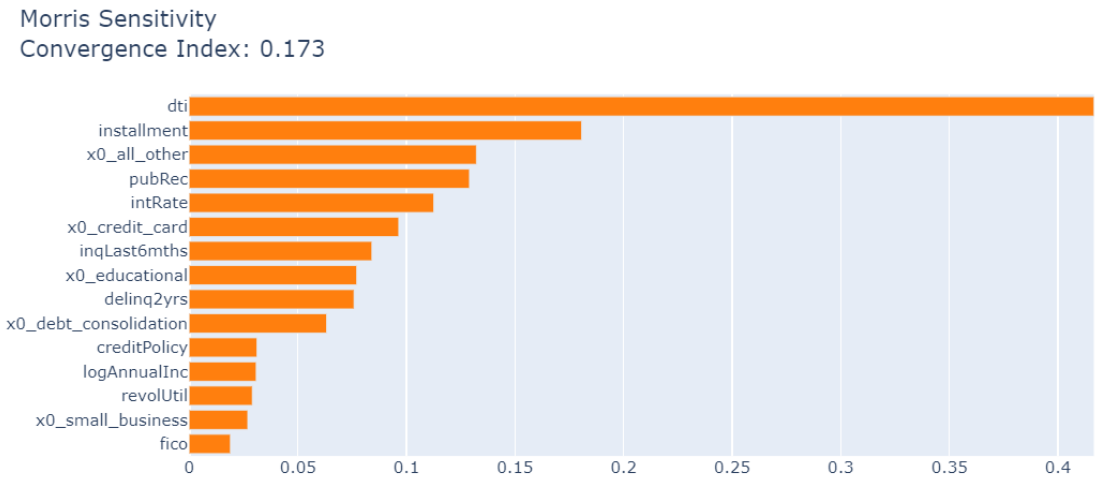

Fig. 14- Morris Sensitivity for Naïve Bayes algorithm.

Fig. 14 shows the results from the Morris method applied for this algorithm. One can see that the most relevant features are the amount of debt divided by annual income, followed by instalment value and the

purpose of the loan is for not any of the other available options. Comparing with the Morris method for Logistic Regression algorithm, one can see that despite AUC in very similar the classification criterion is based on different variables. For instance, in Logistic Regression, the borrower's number of inquiries by creditors in the last 6 months is a majorly important feature followed by the annual income and in third the fico value. Comparing based on the most important feature, for Naïve Bayes the amount of debt divided by the annual income is the most relevant while for Logistic Regression is if the borrower had failed credit payment in the last 6 months. So, apparently, Logistic Regression gives stronger importance to the past finance life of the borrower (including the fico value, bankruptcy filings, tax liens, or judgments) while Naive Bayes seems to focus more on his/her present finance situation. Moreover, the convergence criteria for Logistic Regression seem to be better achieved, so making its importance feature analysis maybe more reliable.

According to these findings and taking into account the AUC and the convergence index the Logistic Regression seems to be the better model be used in this binary classification setting.

Finally; SHAP (SHapley Additive exPlanations) was used to explain individual predictions from some instances from the Naïve Bayes algorithm. Three cases are considered, Fig. 15 (left) shows for the notFullyPaid=1 class, in the case where prediction can be considered correct. This classification was majorly based on the purpose of the loan being for small business, the positive number of inquiries performed by creditors in the last 6 months and the high interests rate values. In its turn, Fig. 15 (right) shows a correctly predicted 0-type class. The 0-type classification was based on compliance with credit policy, that the purpose of the credit was not for small business and the nearly zero inquiries performed by creditors in the last 6 months. In its turn, Fig. 16 shows a miss classified instance for the negative class; this 0-type misclassified instance (model predicted type 1) was based on the positive number of inquiries performed by creditors in the last 6 months, also the failure with credit policy and the high interest rate have contributed for that prediction.

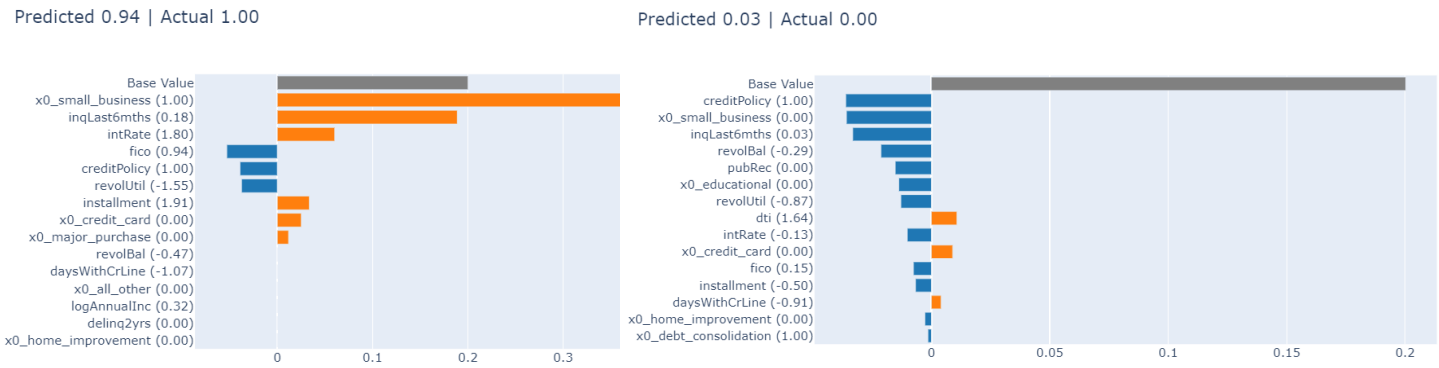

Fig. 15- – Correctly predicted type-1 class from the Naïve Bayes model from the interpretML library (left). Correctly predicted type-0 class from the Naïve Bayes model from the interpretML library (right).

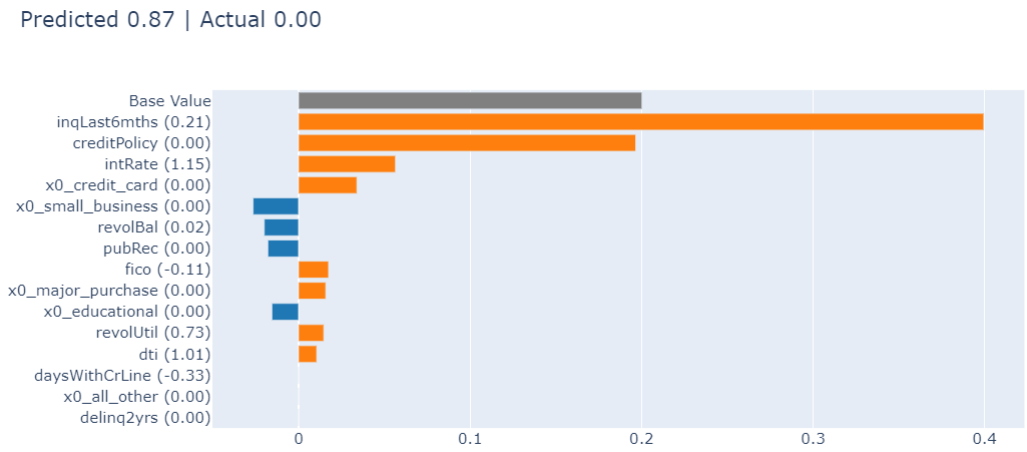

Fig. 16 – Incorrectly predicted type-1 class from the Naïve Bayes model from the interpretML library.

Neural Networks

NN were also considered for this binary classification problem. Full model considered 3 hidden layers and dropout as a regularization approach. Hyperopt output helped on the choice of the relevant parameters. The optimized parameters included the number of units in each hidden layer, batch size, number of epochs and the dropout regularization in each layer by using 100 hyperot evaluations. Indeed, the best result was obtained for 64 units in the first layer with a 0.9 rate regularization, 128 units in the second layer and also a 0.9 rate in the dropout regularization, a batch size of 16 and 100 epochs. A single hidden layer or 3 hidden layers provided slightly worse results. A binary_crossentropy loss function was considered, using the 'adam' optimizer and a sigmoid activation output function. PReLU was used as activation layer in each hidden layer and initialization was 'glorot_normal'. Fig. 17 shows that the NN analysis provided slightly better AUC value than Logistic Regression, RF, EBM or Naïve Bayes. Despite of that, unfortunately it was not possible to analyse NN with the interpretML.

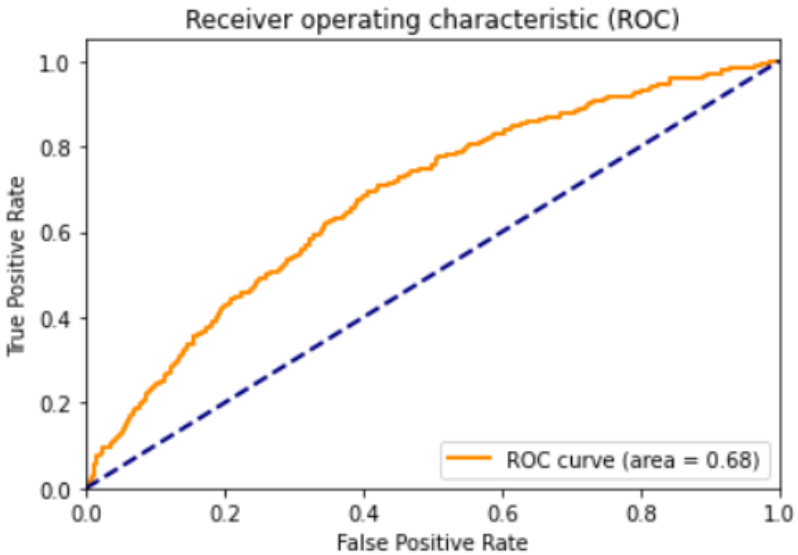

Fig. 17 - ROC curve for the Neural network algorithm.