MRM Individual Project Report

Model Risk Management for Machine Learning Models in Consumer Lending

Cheng-Yun Tsai, ct2854

Introduction

Consumer spending/ lending is one of the most important systemic risk and drivers of macroeconomic conditions (Khandani et al., 2010), and the lending decision informed by credit underwriting models could affect the lives of hundreds of millions of Americans. Upon large amounts of potential data, banks are increasingly relying on machine learning models to make better lending decisions in recent years (Aziz & Dowling, 2019). However, the black-box essential of machine learning models makes the transparency and explainability critical problems for fair and responsible use.

Transparency and explainability of machine learning models is a popular area of research, especially in the finance industry which naturally has high noise of data and high demand on model explainability. Also, lots of research has contributed to the area of applying machine learning models in consumer lending. Khandani et al. (2010) used a unique dataset consisting of transaction level, credit bureau, and account-balance data for individual consumers for constructing forecasting models of consumer credit risk. In the research, the generalized classification and regression trees (CART) model was applied to predict consumer credit default events. Moreover, using data from UniCredit Bank, Figini et al. (2017) proposed models based on Multivariate outlier detection techniques based on Local Outlier Factor to predict default for small and medium enterprises (SMEs).

Furthermore, Spiess (2022) used fidelity, consistency, and usability for evaluating machine learning models in consumer lending. The research implemented various models including Logistic regression, various versions of NN, XGBoost, as well as 6 classification models proposed by cooperated companies for default prediction analysis. Open-source tools, including LIME, SHAP, and Permutation Importance, are applied for extracting feature importancy.

In this project, follow the Spiess (2022)'s method, models including Logistic Regression, Random Forest, XGBoost, simple NN, and two CNN models are developed to forecast the lending default events. The models are trained and tested under various matrices, and the open-source tool, including LIME and SHAP, are utilized to find important features. To assess the explainability of the models nearest neighbor test and perturbation test are conducted to test model fidelity, and tool consistency and model consistency are also validated for model consistency.

Section two describes the methodology of the research, including the data used, the models implemented, and the open-source tools applied. Moreover, section three discusses the training and testing results of our machine learning models and the results from the fidelity tests and the consistency tests. Finally, section four concludes the project with the expected future research/

improvements and the conclusion of the project. Codes can be accessed at: https://github.com/mufeng703/mrm-ml-cl

Methodology

1. Data and Feature Engineering

In the project, I use the credit default data from Home Credit Group. The dataset contains various features of the loan and customer information, including both numerical and categorical features. Numerical features include the credit amount of the loan, the client's age, number of family member the client have, etc. Categorical features contain the contract type (cash loans or revolving loans), client's gender, whether the client have car, etc.

As shown in Table 1, the dataset contains 307,511 observations with 24,825 default cases. For training and testing the machine learning models, I separated the original dataset by 70% training data and 30% testing data. The original default rate of both training data and testing data are approximately 12%. Data can be retrieved from: https://www.kaggle.com/competitions/home-credit-default-risk/data?select=sample_submission.csv.

However, with the original dataset, models can easily get 87% accuracy if they predict all cases to be not default since the small number of total default cases. Hence, I use the Synthetic Minority Oversampling Technique (SMOTE) to deal with the data imbalance problem. SMOTE use features of K nearest neighbor (KNN) to generate synthetic data points for the original dataset. After using SMOTE, our new dataset has 339,223 observations with 16.6% default rate for both training and testing data.

	# Observation	# Default	Default Rate
Training Data	215257	17412	12.3%
Testing Data	92254	7413	12.4%

Table 1. Summary Statistics: Original Data

	# Observation	# Default	Default Rate
Training Data	237456	39568	16.6%
Testing Data	101767	16969	16.6%

Table 2. Summary Statistics: Data after SMOTE

2. Machine Learning Models

As we discussed above, this research implements 6 machine learning models for lending default prediction. The details of the models are described below.

2-1 Logistic Regression

Logistic regression classifier is a common statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set (Lawton). In the research I use the logistic regression classifier in scikit-learn package. The solver I use is "liblinear" which assign the solver to be LASSO regression.

2-2 Random Forest

Random forest is a Supervised Machine Learning Algorithm that is used widely in classification and regression problems (Sruthi, 2021). I also use the random forest classifier in the scikit-learn package. I adopted the default model, that is, no hyperparameters being changed.

2-3 XGBoost

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems (Nvidia). I use the vanilla version of XGBClassifier in the project, which has the binary logistic as it's objective.

2-4 Simple Neural Network

I use tensorflow2 (Keras) to implement our first NN model. The proposed NN model is simple, which only contains two dense layers and an output layer. The first dense layer has 8 neurons, and the second dense layer has 24 neurons. The simple NN model is used to compare the performance with more complex CNN models.

2-5 Convolutional Neural Network 1

Convolutional neural network (CNN) is a network architecture for deep learning which applied convolution technique and especially useful in the area of computer vision. Our first CNN model contains two convolution layers, one pooling layer, and an output layer. The CNN model is also implemented by Keras.

2-6 Convolutional Neural Network 2

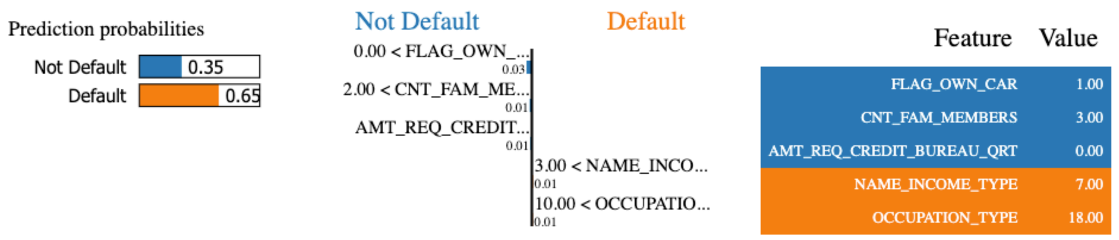
The second CNN model is more complicated. The CNN2 contains 4 convolution layers, and one pooling layer is followed after every 2 convolution layer. And finally, a flatten layer and an output layer in the end of the model structure.

3. Open-source Tools

Open-source tools are the diagnostic tools for the project to extract the features that have great contributions to the client's default (the default drivers). We use LIME and SHAP to extract default drivers in this project.

3-1. LIME

LIME stands for Local Interpretable Model-Agnostic Explanations. LIME is a technique that approximates any black box machine learning model with a local, interpretable model to explain each individual prediction (C3.ai). Figure 1 shows a LIME example that input one row of data and a trained XGBoost classifier. The plot in the left shows the prediction probability of the case, the middle shows how the features contribute to the prediction result with a LIME score, and the right table shows the most important features and its value in the dataset.



Feature	Value
FLAG_OWN_CAR	1.00
CNT_FAM_MEMBERS	3.00
AMT_REQ_CREDIT_BUREAU_QRT	0.00
NAME_INCOME_TYPE	7.00
OCCUPATION_TYPE	18.00

Figure 1. LIME example with XGBoost

3-2. SHAP

SHAP stands for Shapley Additive Explanations. SHAP is a game theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions (SHAP). Figure 2 shows the global SHAP value of our trained XGBoost classifier. EXT_SOURCE_3 on the top of the graph is the feature with the largest importance, the higher value of the feature has a smaller impact on the model output. Hence, EXT_SOURCE_3 is negatively correlated to default (EXT_SOURCE_3 with value 1 tends to be predicted for "not default").

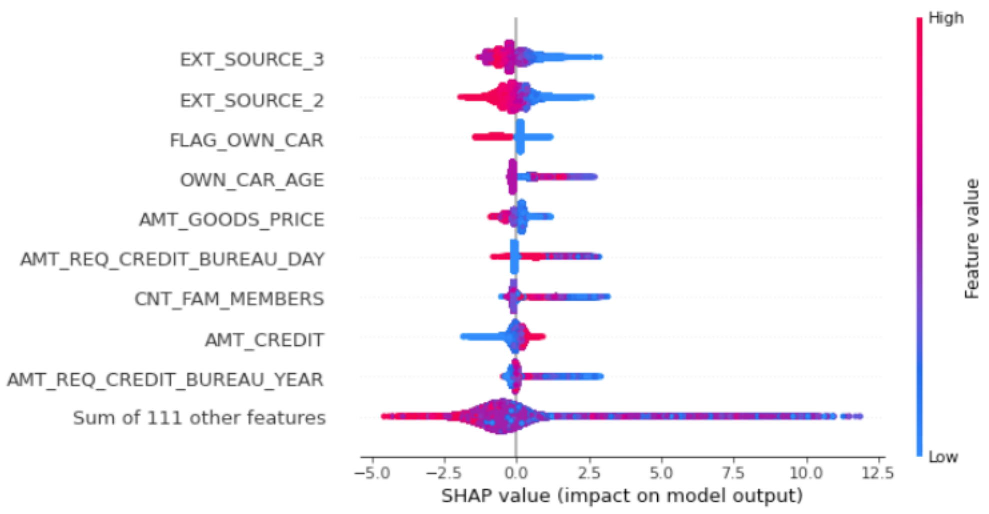


Figure 2. SHAP example with XGBoost

Model Validation and Test Results

1. Machine Learning Models

As discussed above, I implemented 6 machine learning models with scikit-learn, XGBoost, and TensorFlow. When back testing the models, I calculated six matrices, including accuracy, area under receiver operating characteristic curve (ROC AUC), mean square error (MSE) and Log Loss, for model validation. ROC curve is a common way of quantifying the magnitude of classification errors, the curve plots the true positive rate against the false positive rate. To summarize the ROC curve, AUC is a standard measure that calculates the area under the ROC curve. Models with 1.0 AUC indicates the models perform perfect classification, and 0.5 means the models with no skill. MSE computes the difference between the model's predictions and the true outcome. The smaller the MSE is, the more accurate the model is. Log Loss is a common evaluation metric that determines how close the predicted probabilities are to the true outcomes. Smaller log loss values correspond to more predictive accuracy.

The results of the matrices are concluded in the Table 3. Overall, the XGBoost classifier outperforms all other models in terms of all the matrices with 0.925 accuracy, 0.889 ROC AUC, 0.075 MSE, and 0.232 log loss. Moreover, the random forest classifier takes the second place in all matrices. Among NN models our CNN2 has the best accuracy (0.904), ROC AUC (0.855), MSE (0.079), and log loss (0.275), while all NN based models underperform the random forest classifier and the XGBoost classifier.

In the result table, we can find that the ROC AUC value of the simple NN model is 0.5, which means the model does not have any ability to describe our target value. Table 4 shows that the simple NN model predicted all test sample as negative (not default) to decrease the binary entropy loss (log loss). The model does not provide any information or make any decision in prediction. Furthermore, the default probability predicted by the model are the same while using all the sample data we have. Hence, the model is not included in the further analysis.

Model	Accuracy	ROC AUC	MSE	Log Loss
LR	0.833	0.630	0.167	0.437
RF	0.918	0.881	0.083	0.263
XGBoost	**0.925**	**0.889**	**0.075**	**0.232**
Simple NN	0.833	**0.500**	0.139	0.451
CNN1	0.890	0.842	0.088	0.301
CNN2	0.904	0.855	0.079	0.275

Table 3. Model Performance

	Predicted Negative (0)	Predicted Positive (1)
Actually Negative (0)	84798	0
Actually Positive (1)	16969	0

Table 4. Confusion Matrix for Simple NN

2. Fidelity Test

Spiess (2022) define fidelity as the ability to reliably identify features that are relevant to a model's prediction and asks how well an explanation approximates the prediction of the black box model. To assess the fidelity of the models, I conducted two tests in the project: Nearest Neighbor Test and Perturbation Test.

2.1. Nearest Neighbor Test

The aim of nearest neighbor test is to verify whether the samples have similar features with the rejected sample are also being rejected. Suppose that a customer being rejected, if he/she has any problem of this decision he/she may want to compare the result of rejection to other people with similar values of rejection drivers. If other customers have similar value in the rejection drivers (of the rejected customer), but they are not rejected the customer may not be satisfied with the decision of rejection.

In the nearest neighbor test, I first select a random sample from the whole dataset which is rejected (that is, the sample that default probability predicted by the model is greater than 0.2). After getting the rejected sample, the importance of every feature will be calculated using both LIME and SHAP. I choose five features that contribute the greatest impact to default as the default drivers of the sample (drivers are different across samples and open-source tools).

Neighbors of the rejected sample are chosen by filtering all samples have similar values in the drivers (columns). If a sample has all values in the selected five drivers that less or equal to 1.1 times the values in the original rejected sample, **and** greater or equal to 0.9 times the values in the original rejected sample, I consider it as a neighbor of the original sample. The default probability of the neighbors will be predicted using the model, and the proportion of rejection number of the neighbors will be calculated. Larger rejection proportion means samples with similar values in rejection drivers are also being rejected. The greater the value is, the higher the fidelity of the model is. I iterated the previous steps 100 times for every model and every open-source tool. The average of the proportions is the fidelity of each model with respect to the open-source tool.

Table 5 shows the result of nearest neighbor test. I use the Total Reject, which is the proportion of the rejection (default probability greater than 0.2) of the whole dataset, as the benchmark. XGBoost has the highest fidelity of 0.934 from SHAP and CNN2 has the highest fidelity of 0.561 from LIME. We can observe that the fidelity of every model/open-source tool is higher than the value of Total Reject. The result means the models are at least outperform the average value of randomly choose drivers, that is, the rejection drivers chosen by LIME and SHAP for each model has the ability to explain the rejection decisions. Also, the fidelity from SHAP for every model are significantly higher than from LIME, which means the default drivers extracted by SHAP has more ability to explain the rejection decision in the aspect of nearest neighbor than drivers from LIME. Note that the logistic regression has the lowest fidelity in both LIME and SHAP.

	LR	RF	XGBoost	CNN1	CNN2
LIME	0.351	0.368	0.403	0.480	**0.561**
SHAP	0.843	0.846	**0.934**	0.857	0.884
Total Reject	0.294	0.178	0.180	0.201	0.203

Table 5. Nearest Neighbor Test Result

2.2. Perturbation Test

Perturbation tests assess whether changes in the value of five selected default drivers actually decrease the probability of default and change the rejected cases to accepted. Consider again a customer being rejected, it will be helpful if we can tell the customer the reasons (drivers) for the rejection, and, the improvements in the reasons (drivers) can truly change the result to acceptance.

In the test, I used the five default drivers, which are extracted from LIME and SHAP (like discussed above), of rejected samples as the drivers to be improved. I perturb the drivers of the sample by the following rules: (1) if the value is 0 then change it to 1 (2) else if the value is 1 then change it to 0 (3) else if the value is larger than zero change it to 0.5 times the value (4) else if the value is smaller than zero change it to 2 times the value. After perturbing the selected sample, the new probability of default is calculated by the same model. Change in PD are calculated by deducting the original probability of default by the new probability of default. The larger the Change in PD is, the better the perturbation works, that is, the higher fidelity of the model. I iterate the procedure 100 times for each model and open-source tool, and the final Change in PD values are the average value of the Change in PD value of each iteration. The Change to Accept are derived by calculating the proportion of the sample change to be accepted (the probability of default is less than 0.2) after the perturbation.

Table 6 shows the result of perturbation test for each model and open-source tool. Random forest classifier has the highest Change in PD using both LIME (0.108) and SHAP (0.117), which means the model decrease the probability of default most after the perturbation while using the perturbation method. CNN1 has the highest Change to Accept with LIME (0.37) and logistic regression has the highest Change to Accept with SHAP (0.41). The fidelity scores from the perturbation tests show that the models are not sensitive with the perturbations. We can observe that there are three values of Change in PD that are less than zero. It demonstrates the perturbation in the rejected samples does not decrease the probability of default, instead, it increases the probability of default. Moreover, since the maximum value of Change to Accept is only 0.41 and half of the Change to Accept values are below or equal to 0.1, we can conclude that the models do not perform well in the perturbation test.

		LR	RF	XGBoost	CNN1	CNN2
LIME	Change in PD	0.003	**0.108**	0.009	-0.137	0.082
	Change to Accept	0.13	0.02	0.1	**0.37**	0.06
SHAP	Change in PD	0.033	**0.117**	-0.029	-0.018	0.075
	Change to Accept	**0.41**	0.07	0.08	0.16	0.29

Table 6. Perturbation Test Result

3. Consistency Test

Also, follow Spiess (2022), I test two notions of consistency. Consistency across models asks how often a given model diagnostic tool identifies the same features as drivers of rejection decisions across different underwriting models. Consistency across tools asks how often two participating companies – or open-source tools – identify the same features as drivers of rejection decisions.

3.1 Model Consistency

The objective of the model consistency test is to verify if the default drivers of two models extracted by open-source tools are similar. It is preferable that if the fidelity scores of two models are high, the default drivers of these two models should not vary wildly.

To conduct the model consistency test, similar to previous analysis, I draw a rejected sample from the whole dataset and extract five default drivers with LIME and SHAP. After extracting the five drivers from each model, the intersections of the features between models will be extracted. For each model pair, the score calculated by the intersection divided by 5 will be the proportion of drivers that overlap each driver list for the sample drawn. 100 iterations will be conducted, and the average of the proportion for each model pair will be the model consistency score of the pair. The higher the score is, the better the model consistency across the pair of models.

Table 7 and Table 8 shows the result of model consistency test with respect to LIME and SHAP. We can see that in both tests with LIME and SHAP, CNN1 and CNN2 have the highest model (0.246 and 0.576) consistency. The result is intuitive and reasonable, because CNN1 and CNN2 have the most similar structure among all model pairs. Moreover, logistic regression and random forest have worse model consistency when comparing with NN based model than others with LIME, while only logistic regression have worse model consistency when comparing with NN based model than others with SHAP.

And Similar to the result from fidelity test, model consistency with SHAP are significantly higher than model consistency with LIME except the logistic regression and NN1 pair.

	LR	RF	XGBoost	CNN1	CNN2
LR	1.0				
RF	0.052	1.0			
XGBoost	0.062	0.19	1.0		
CNN1	0.004	0.0	0.068	1.0	
CNN2	0.012	0.008	0.038	**0.246**	1.0

Table 7. Model Consistency with LIME

	LR	RF	XGBoost	CNN1	CNN2
LR	1.0				
RF	0.127	1.0			
XGBoost	0.347	0.474	1.0		
CNN1	0.003	0.360	0.256	1.0	
CNN2	0.015	0.414	0.338	**0.576**	1.0

Table 8. Model Consistency with SHAP

3.2 Tool Consistency

Tool consistency test the proportion of drivers overlap within open-source tools. While the performance of models with SHAP are generally better than models with LIME in the previous fidelity and consistency tests, I do not expect the high tool consistency among models.

Like previous tests, I use the drivers of a rejected sample generated by LIME and SHAP and get the intersection of two driver list, from LIME and SHAP, respectively. The proportion of the number of overlapped features over 5 are calculated. Repeat the procedure 100 times and average all proportion, the result is the final tool consistency score as regards to the model.

As shown in Table 9, CNN2 has the best tool consistency of 0.32. It is surprisingly that the more complex model tends to have better tool consistency.

	LR	RF	XGBoost	CNN1	CNN2
Consistency	0.09	0.124	0.106	0.26	**0.32**

Table 9. Tool Consistency Test Result

Conclusion

1. Future Improvement

First of all, in our fidelity test, models do not perform well in the perturbation test. One reason for the bad performances of the perturbation test may cause by the perturbation method I used. Besides change the value of binary categorical data, I directly double the negative value and half the positive value if the feature is numerical. Some of the results shows the alternation even deteriorate the probability of default. Hence, there may be some better perturbation method for changing the value default drivers.

Moreover, in the perturbation test, we can observe that although some models have relatively low Change in PD, their Change to Accept are relatively high and vice versa. A reason for the consequence may because I only draw and repeat the procedures 100 times in all fidelity and consistency tests, and the samples drawn for the tests are different across models. Therefore, increase the iteration number will provide more precise test results.

Lastly, besides the fidelity tests and perturbation tests, Spiess (2022) also conducted three tests to assess the model usability, that is, the ability to identify drivers that provide a rejected applicant with a feasible path to acceptance within limited time. Hence, to provide more details of the usability of the machine learning models in consumer lending, the results from the usability test will be useful indicators.

2. Project Conclusion

In the project, I use the data provided by Home Credit Group and train 6 machine learning models including logistic regression, random forest, XGBoost, and two CNN models. Several statistical metrices, including accuracy, ROC AUC, MSE, and log loss, are applied for model validation. To test the model fidelity and consistency, nearest neighbor test, perturbation test, model consistency test, and tool consistency test are conducted.

To summarize the result, XGBoost has the best performance in every performance metrices and in the nearest neighbor test with SHAP but has really bad performance in perturbation test. Random forest takes the second place in in every performance metrices and has relatively good performance in perturbation test. CNN2 has the best performance in the NN models and in the nearest neighbor test with LIME. Finally, for evaluation tools, drivers extracted by SHAP has better ability to explain the model and results than LIME.

References

Aziz, S., & Dowling, M. (2019). Machine learning and AI for risk management. In *Disrupting finance* (pp. 33-50). Palgrave Pivot, Cham

Figini, S., Bonelli, F., & Giovannini, E. (2017). Solvency prediction for small and medium enterprises in banking. *Decision Support Systems*, *102*, 91-97.

Khandani, A. E., Kim, A. J., & Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, *34*(11), 2767-2787.

Spiess, J. (2022). Machine Learning Explainability & Fairness: Insights from Consumer Lending.