Modelling Threshold Effects for Fraud Detection with Novel Machine Learning Techniques

Athan Liu, Mason Mackall

June 1, 2021

Abstract

Dumitrescu, et al. (2021) present a new method, named penalized logistic tree regression (PLTR) for discrimative modelling of data with threshold effects. PLTR aims to combine the accuracy of random forests with the ease of interpretability in logistic regression. Dumitrescu, et al. (2021) illustrated the competitive accuracy and interpretability of PLTR in credit scoring, a data with well-known threshold effects. This work aims to demonstrate the robustness of PLTR for modelling threshold effects by implementing PLTR for a different problem, fraud detection. Monte Carlo simulation of credit card transaction data provided evidence for the existence of threshold effects, and the potential for improved accuracy and reliable interpretability with PLTR. The simulated results were backed by experiments on real data, with PLTR outperforming random forest and logistic regression, and identifying several bivariate threshold effects with significant marginal effects on fraud detection.

1 Introduction

Logistic regression has been the industry standard for discrimative modelling (modelling $\mathbb{P}(Y \mid \mathbf{X})$) for many years. This is because logistic regression provides an easily interpretable model with sufficient accuracy. In recent years, however, econometricists have obtained much greater accuracy for some problems with machine learning methods. In particular for credit scoring, it was found that random forests of binary decision trees greatly outperformed logistic regression; see Lessmann [4].

The reason for the improved performance of random forest in credit scoring was due to the ability for random forests to model threshold effects. Threshold effects occur when an effect due to some predictive feature does not occur until that feature crosses a certain threshold. In generalized linear models, these can be modeled by indicator variables that take value 0 when the relevant feature is below (above) the threshold, and 1 when the feature is above (below) the threshold. Random forests effectively model this behavior as they generate decision rules for classification.

However, random forests have yet to become industry standard in many fields where logistic regression is used for classification and prediction. This is because random forests are not easily interpretable. Random forests tend to generate a large number of very complex decision rules with multiple predicates (conditions), and how random forests aggregate multiple decision trees can often be a black-box process. Random forests therefore are confusing and unusable to customers and industry regulators. While less accurate, the components of a logistic regression model have clear interpretability, and it is easy to see how each component of the model affects the final outcome, as it is a simple linear model transformed by a logit function. In short, there is a tradeoff between interpretability and accuracy.

With this tradeoff in mind, Dumitrescu, et al. [2] proposed the penalised logistic tree regression (PLTR). The model, in short, generates binary predictors from the decision rules of binary trees, selects the relevant features using variable selection with adaptive LASSO logistic regression, and then predicts on this fitted model. In this way, PLTR hybridizes the improved predictive power from nonlinear effects of random forests with the ease of interpretability of logistic regression. Dumitrescu, et al. [2] details a process for assessing their model's effectiveness on data with non-linear effects using Monte Carlo simulations, and then compared the results to a variety of other benchmark credit scoring methodologies on real datasets.

The goal of this paper is to see if the hybridized PLTR model maintains its proposed benefits in accuracy and interpretability against benchmark methods when there is assumed to be a nonlinear effect in the model, but the problem is different. Dumitrescu et al. [2] used consumer data that had been labeled with a binary indicator for whether or not a consumer defaulted on a loan. This paper will use credit card transaction data labeled with 'fraudulent' and 'genuine' labels in an attempt to construct a fraud detection model with PLTR. Aside from uncovering an interpretable model that will allow people to determine what characteristics of a transaction increase the risk of fraud, it will also further demonstrate the robustness of these hybridized techniques in modelling threshold effects for binary classification. The methodology for assessing model accuracy and interpretability has been slightly altered for the problem at hand, but will roughly follow the methodology outlined by Dumitrescu et al [2].

The paper is organized as follows. Section 2 outlines assumptions on a simulated data generating process (DGP), provides more background on logistic regression and random forests, and contain Monte Carlo simulations of the two models. Section 3 will explain the PLTR model and contain Monte Carlo simulations of the PLTR model. Section 4 will apply the logistic regression model, the random forest model, and the PLTR model to real-world credit card transaction data and analyze the numerical results. Section 5 summarizes the results and details areas for improvement and expansion in future work.

2 Data modelling and analysis

The code for all this work can be found at https://github.com/uc1309/pltr-credit-fraud.

2.1 The logistic regression model

Consider the data $(\mathbf{x}_i, y_i)$ where $i \in \{1, \cdots, n\}$, $\mathbf{x}_i \in \mathbb{R}^p$, and $y_i \in \{0, 1\}$ a binary indicator where 1 indicates a fraudulent transaction. The goal is to model $\mathbb{P}(y_i = 1 \,|\, \mathbf{x}_i)$. The components of $\mathbf{x}$ are suspected factors of consumer behavior or transnational data that may indicate a fraudulent transaction. The classical logistic regression model for this problem would take some form

$$\mathbb{P}(y_i = 1 \mid \mathbf{x}_i) = \frac{1}{1 + \exp(-\eta(\mathbf{x}_i; \beta))}$$
$$\eta(\mathbf{x}_i; \beta) = \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} \qquad \beta = (\beta_0, \beta_1, \cdots, \beta_p) \in \mathbb{R}^{p+1}$$

Because publicly available transaction data obscures the actual data categories for privacy reasons (commonly via principal component analysis), it is impossible to provide specific examples of what these predictors are. However, Le Borgne and Bontempi [3] advocate for the inclusion of features such as weekday-weekend indicators, day-night indicators, average consumer transaction frequency, average transaction amount over time, average transactions over time at a particular terminal, and average frauds over time at a particular terminal.

From these proposed features, it is reasonable to assume the existence of non-linear threshold effects. For example, it would be reasonable to assume that a consumer suddenly making significantly more transactions in a day relative to their week and month-long average is at a higher risk for fraud than a consumer maintaining their average transaction count. Similarly, if a transaction is being performed at a terminal with a high risk of fraud, it follows that the transaction at that terminal is more likely to be fraudulent than a transaction at a terminal with a lower risk of fraud. A simple linear logistic regression model cannot capture these nonlinear effects.

The general failure of logistic regression in the presence of threshold effects is proven in Monte Carlo simulations of the following DGP as specified in Dumitrescu et al. [2]

$$\begin{aligned} x_{i,j} &\sim \mathcal{N}(0,1) \\ \Theta &= (\beta_0, \beta_1, \cdots, \beta_p, \beta_{1,2}, \cdots, \beta_{(p-1),p}) \sim \text{Unif}(-1,1) \\ \eta(\mathbf{x}_i; \Theta) &= \beta_0 + \sum_{j=1}^{p} \beta_j \mathbb{1}(x_{i,j} \le \gamma_j) + \sum_{j=1}^{p-1} \sum_{k=j+1}^{p} \beta_{j,k} \mathbb{1}(x_{i,j} \le \delta_j) \mathbb{1}(x_{i,k} \le \delta_k) \\ y_i &= \begin{cases} 1 & \text{if } \mathbb{P}(y_i = 1 \mid \mathbf{x}_i) > \pi \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where π is the median of the generated probabilities, and the thresholds δ_j, γ_j are chosen to exclude data below the first decile of the support of the data. Dumitrescu, et al. [2] empirically verifies that as the number of parameters p increases in this model, the predictive accuracy of logistic regression fails, even when the model is fitted with quadratic interaction terms to introduce some nonlinearity.

It is of greater interest to this work to see if the same predictive failure occurs in a simulation of credit card fraud with imposed threshold effects. If the disparity between the predictive accuracy of the decision tree and the logistic model exists, then there is reason to believe that the PLTR model can be applied here to obtain better accuracy than the logistic model and better explainability than the binary decision tree.

2.2 The random forest model

A random forest classifier uses the output of multiple decision trees to classify data. The individual decision trees use a greedy algorithm of recursive partitioning, continually adding different decision rules ($x_{i,j} \leq \delta_j$ or $x_{i,j} \geq \delta_j$), as nodes to the tree. The tree is binary; each added node splits a subset of the data into two new subsets. The recursion ends when each subset contains only one type of class, or when the maximum depth is reached. When predicting new labels, the random forest runs each decision tree on the new observation and picks the most frequent label produced by the trees. For the fraud detection problem, random forests of 100 trees were constructed by random sampling with replacement of the training data. The maximum depth was determined to be 10 to prevent overfitting.

Random forests are well suited to the fraud detection classification problem for several reasons. Primarily, decision rules purely encapsulate threshold effects, and multiple decision rules of varying complexity can be generated at low cost. Additionally, they are resistant to overfitting in problems with imbalanced class labels, as is the case in fraud detection. The resistance is due to the creation through random sampling of multiple trees, so that while any particular tree may, by chance, overfit to its random sample, that particular fit becomes outweighed by the combination of trees.

2.3 Simulating credit card fraud data

Because of the lack of publicly available data sets, any research on fraud detection methods must rely on simulated data. This paper adopts the data simulation outlined by LeBorgne and Bontempi [3], with slight modifications. The data generating process and feature engineering is described in this section, as well as the reason for adopting the simulation method.

There are three core groups of features in credit card transaction data. The first group is information about the transaction itself, such as time and date. The second is information about the consumer, such as average transactions per length of time, and average transaction amount per length of time. The final information is about the terminal (the point where the transaction occurs). This includes data such as the transaction frequency and the ratio of fraudulent to non-fraudulent transactions at a given terminal.

The simulation first starts with generating customer profiles. Each customer is assigned a geographic location, with the x and y-coordinate drawn from a Unif$(0, 100)$ distribution. Each customer then makes a transaction of amount T, where

$$T \sim \mathcal{N}(\mu, \sigma = \frac{\mu}{2})$$
$$\mu \sim \text{Unif}(5, 100)$$

If $T < 0$, then it will be redrawn from a Unif$(0, 2\mu)$ distribution. A customer also makes M number of transactions per day, where

$$M \sim \text{Poisson}(\lambda)$$
$$\lambda \sim \text{Unif}(0, 4)$$

Next, terminals are generated. Terminals are simply assigned a geographic location in the same way customers are assigned a geographic location. Customers and terminals are then linked by finding all terminals within a Euclidean distance of r.

Each transaction is then generated at a terminal, chosen randomly from the list of available terminals, at a time $t \sim \mathcal{N}(43200, \sigma = 20000)$. For reference, there are 86400 seconds in a day, and empirical evidence suggests that transactions tend to follow this normal distribution, centered at noon. Unlike the transaction amount, if the time t is outside of a day (i.e outside the range [0, 86400]), the transaction is considered a failure and discarded.

Finally, fraud labels are assigned. This is where suspected threshold effects are imposed on the data, and for that reason why it is expected for logistic regression to perform poorly and PLTR and random forest to perform better. Fraud labels are assigned under the following 3 scenarios, which have been modified from the original to match the fraud rate observed in the real world dataset:

1.) A transaction whose amount is over \$220. This introduces a very obvious threshold effect in the model, based on the amount of the transaction.

2.) One terminal selected at random each day will have only fraudulent transactions for 14 days. This introduces a threshold effect based on fraudulent transaction frequency per terminal.

3.) Two customers selected at random each day will have 1/3rd of their transactions multiplied in value by 5 and marked as fraudulent for 7 days. This introduces a threshold effect based on consumer spending habits.

Some transformations on the transaction data are necessary to generate predictive features that can encapsulate these threshold effect behaviors. For the sake of simplicity this paper will adhere to the transformations recommended by Le Borgne and Bontempi [3], which include a weekend-weekday indicator, a day-night indicator, transaction count over time, transaction amount over time, transaction count over time per terminal, and fraudulent transaction count over time per terminal. There are, however, many other predictive features that could be of interest; a few this paper would like to propose for future research would be metrics calculating the deviation of transaction amount from the customer averages, and deviation of fraudulent transaction amount from the terminal averages. Again, the actual features used in real data for fraud detection are unknown, and it is to the advantage of the PLTR method to introduce as many potential features hypothesized to have a threshold effect on the outcome. The effect of these hypothesized features in simulated experiments are of interest in future work.

2.4 Monte Carlo simulation results

To evaluate the precision of these models, customer and terminal data was generated and matched 100 times. 45 days of transaction data were then generated, labeled with fraud indicators, and the last 2 days of data were then extracted for model training and validation, using a simple 80-20 train-test split.

The metric for assessing model accuracy was the average precision score (AP score), which is also the area under the precision-recall curve (PRC). AP score was selected for two reasons. First, there is heavy class imbalance in the data; in actual data, only 0.1% of the transactions are fraudulent, and parameters of the simulation were adjusted to encapsulate this behavior. Second, the cost of misclassification is not equal; a false positive (flagging a genuine transaction as fraudulent) is less harmful than a false negative (flagging a fraudulent transaction as genuine), and the AP score helps reflect this.

Model	PRC AUC	STDEV
Training scores		
Logistic regression	0.222	0.0791
Random forest	0.702	0.0851
Test scores		
Logistic regression	0.186	0.1364
Random forest	0.406	0.2015

From Monte Carlo simulation, it would appear that assumptions regarding threshold effects are correct. Logistic regression is consistently and drastically outperformed by the random forest in predictive accuracy.

There are some concerns regarding numerical stability. Namely, the standard deviation of the AP score for both the logistic regression and random forest on the test data is quite high. In addition, the mean of the AP scores themselves are quite low. The authors hypothesize that the reason for this numerical instability is that the number of samples analyzed in each simulation is too small, relative to real data. In real data, around 240,000 transactions can be observed in two days' time. However, in Monte Carlo simulation only approximately 19,000 are observed in two days' time. Unfortunately, simulating transactions at a rate of approximately 120,000 per day was too computationally intensive for both the author's computers given the timeline. As the goal of Monte Carlo simulation is to mimic the actual data as much as possible, a compromise was made to ensure the rate of fraud was the same at the cost of the number of samples generated in a day.

3 Specifying the PLTR model

3.1 The algorithm

The PLTR model as proposed by Dumitrescu, et al. [2] is rather straightforward. As the name suggests, it is broken into three parts: a tree component, a logistic regression component, and a penalization component.

To start, binary decision trees are used to determine threshold effects. Dumitrescu, et. al [2] use trees with a maximum depth of 2 for model and computational simplicity, although deeper trees can be used to generate multivariate threshold effects. For each univariate threshold effect for a particular predictive feature j, a binary indicator variable $\mathcal{V}_{i,1}^{(j)}$ is returned, where $\mathcal{V}_{i,1}^{(j)} = 1$ if the value $x_{i,j}$ is greater than the decision rule, and 0 otherwise. For each bivariate threshold effect for a pair of predictive features (j, k), a binary indicator variable $\mathcal{V}_{i,2}^{(j,k)}$ is returned, where $\mathcal{V}_{i,2}^{(j,k)} = 1$ if $x_{i,j}$ is less than the decision rule but $x_{i,k}$ is greater than the decision rule, and 0 otherwise. In the event that the predictive feature j or k is determined to be non-informative by the binary tree, then the binary variable is simply omitted. This process generates at most $p + q, q \leq \frac{p(p-1)}{2}$ threshold effects, where p is the number of predictive features present in the data.

After generating the threshold effects from short-depth decision trees, these binary variables are added to the logistic regression model as such

$$\eta(\mathcal{V}_{i,1}^{(j)}, \mathcal{V}_{i,2}^{(j,k)}; \Theta) = \beta_0 + \sum_{j=1}^{p} \alpha_j x_{i,j} + \sum_{j=1}^{p} \beta_j \mathcal{V}_{i,1}^{(j)} + \sum_{j=1}^{p-1} \sum_{k=j+1}^{p} \gamma_{j,k} \mathcal{V}_{i,2}^{(j,k)}$$

$$\mathbb{P}(y_i = 1 \mid \mathcal{V}_{i,1}^{(j)}, \mathcal{V}_{i,2}^{(j,k)}; \Theta) = \frac{1}{1 + \exp(-\eta(\mathcal{V}_{i,1}^{(j)}, \mathcal{V}_{i,2}^{(j,k)}; \Theta))}$$

where $\Theta = (\beta_0, \alpha_1, \cdots, \alpha_p, \beta_1, \cdots, \beta_p, \gamma_{1,2} \cdots, \gamma_{p-1,p})$ are the parameters to be estimated via maximum likelihood estimation of the logistic regression log-likelihood.

It becomes immediately obvious that Θ has the risk of being very large. For context, the real world credit card transaction data has $p = 28$ predictors, which means that there can be up to 378 bivariate threshold effects alone. Therefore, a regularization (penalisation) scheme is employed to select relevant parameters and reduce the risk of overfitting. The final estimate of our model thus becomes

$$\hat{\Theta}(\lambda) = \arg\min_{\Theta} -\ell(\mathcal{V}_{i,1}^{(j)}, \mathcal{V}_{i,2}^{(j,k)}; \Theta) + \lambda P(\Theta)$$

where ℓ is the negative log-likelihood of logistic regression, $P(\Theta)$ is a penalty term and λ is a parameter controlling the strength of regularization (penalisation), usually selected by cross-validation. Dumitrescu, et al. [2] select the adaptive lasso estimator proposed by Zou [5], as the probability of excluding relevant predictor and including irrelivant predictors is 0 (the oracle property). The adaptive lasso penalty function is

$$P(\Theta) = \sum_{v=1}^{V} w_v \|\theta_v\| \qquad w_v = \left\|\hat{\theta}_v^{(0)}\right\|^{-\nu}$$

where in practice $\nu = 1$ and $\hat{\theta}_j^{(0)}$ is obtained from logistic ridge regression.

In summary, the PLTR algorithm can be broken into a three-stage process: 1.) generate threshold effects from 2-deep binary decision trees, 2.) calculate initial parameter estimates from logistic ridge regression and 3.) calculate final parameter estimates using logistic adaptive lasso regression.

3.2 Simulation results

The PLTR model was subsequently trained and tested on the same simulated data in Section 2.

Model	PRC AUC	STDEV
Training scores		
Logistic regression	0.222	0.0791
Random forest	0.702	0.0851
PLTR	0.598	0.0758
Test scores		
Logistic regression	0.186	0.1364
Random forest	0.406	0.2015
PLTR	0.537	0.0814

While PLTR did not obtain as high a training score as the random forest, it did outperform the random forest considerably when predicting on test data. The AP scores of PLTR on test data, in addition, have considerably smaller standard deviation to both logistic regression and random forest.

Similar issues regarding numerical stability arose. A very notable issue was that the penalized LASSO logistic regression had issues with convergence on select k-folds and data partitions during cross validation. The authors again suspect this is a result of the combination of a low rate of fraud and a small number of simulated samples. One of the warnings generated mentioned having less than 8 observations of a particular class in logistic regression. Having a small number of samples decreases the total number of observations of the fraudulent class if the sample is proportionally split. The fix was to raise the threshold for convergence of the penalized LASSO solver. Although this fix introduces questions regarding the accuracy of the coefficient values, it allowed for the model to be fit to some extent on 97 out of 100 samples.

The main goal of Monte Carlo simulation of the PLTR model was to observe consistency in interpretability. By nature of the model's design, each Monte Carlo iteration will generate different decision rules with different thresholds, as the short-depth decision tree classifier will identify different sets of features as significant or not at different levels given different data. However, it remains of interest to see if the same predictive feature was selected in a decision rule across simulations, irrespective of the actual threshold in the decision rule. For that reason, the coefficients from simulated models were saved, sorted by absolute average coefficient value, and three sets of coefficients are included in this paper. One simulation is presented here, the other two can be found in the appendix. The rest of the coefficients can be found in the GitHub repository.

#	Decision rules	Avg. value
9	TX_AMOUNT≤218.64,TERMINAL_RISK_7DAY≤0.54	-2.286
10	TX_AMOUNT≤218.64,TERMINAL_RISK_1DAY≤0.75	-1.961
10	TX_AMOUNT>218.64	1.393
5	TERMINAL_RISK_7DAY>0.54,TERMINAL_RISK_30DAY≤0.25	0.999
9	TX_AMOUNT≤218.64,TERMINAL_RISK_30DAY≤0.16	-0.721

From the results of the Monte Carlo simulation, it is evident that PLTR has highly reliable interpretability. The dominant threshold effect, marking every transaction higher than \$220 as fraudulent, is consistently selected and evaluated as the primary threshold effect. The threshold does not deviate from the true value by more than \$3 across all samples. The magnitude of the coefficients are consistent with expected behavior. Lower transaction amount and low terminal risk corresponds with a decreased chance of fraud, while high transaction amount and high long-term terminal risk with low long-term terminal risk correspond with an increased risk of fraud.

4 Real world data

The three models outlined above were applied to real data. None of the numerical stability issues encountered in the simulation experiments were encountered during experiments on the real data set, which reinforces the hypothesis that not enough samples were generated in simulation.

4.1 Predictive performance

The real world credit card transaction data was taken from Kaggle. It includes the time of each transaction, the transaction amount, and 28 features transformed by principle component analysis to protect the privacy of the consumers. There are 284807 transactions, taken over a period of 2 days, with only 492 fraudulent transactions (a rate of 0.17%). For the PLTR model, the maximum number of threshold effects was $p+q = 29+\binom{29}{2} = 435$. Fitting the data with the short-depth decision trees yielded 28 univariate threshold effects (only the category $V26$ was excluded) and 181 bivariate threshold effects (of a maximum 406, around 41.6%), for a grand total of 209 threshold effects, or 44.5% of the maximum total threshold effects. As mentioned above, despite being well short of the maximum possible threshold effects, a considerable amount of new features are added to the data in this process, which motivates the need for penalized logistic regression.

For the random forest and logistic regression, data was partitioned with a 80-20 train-test split, and re-partitioned 100 times. For PLTR, the data was partitioned and used to train the model following the 5×2 cross-validation procedure detailed by Dietterich [1] due to computational limitations. The average AP score and standard deviation for each model was recorded and reported below. The average number of covariates selected after adaptive LASSO in PLTR was 44.5.

Model	PRC AUC	STDEV
Training scores		
Logistic regression	0.557	0.0223
Random forest	0.840	0.0082
PLTR	0.844	0.0185
Test scores		
Logistic regression	0.536	0.0468
Random forest	0.739	0.0374
PLTR	0.819	0.0096

There are a number of key takeaways from the results. The first is that logistic regression is drastically outperformed by both the random forest and PLTR. This suggests that the hypothesis assuming the existence of threshold effects on credit card transaction data are valid in the real world. Furthermore, it highlights the importance of fitting models encapsulating threshold effects to accurately detect credit card fraud. Simply by adding some threshold effects to the logistic regression model in PLTR drastically improves performance.

The second is that when predicting on test data, PLTR outperforms the random forest by a small but not insignificant amount. While Dumitrescu et al. [2] provides both theoretical and empirical justification for this behavior when fitting the two models on a large number of predictive features and corresponding threshold effects, this paper is hesitant to accept these findings as strong empirical justification that PLTR is determinedly better than random forest. The first reason is that while the authors of this paper believe AP score to be the best metric in evaluating model accuracy, it is not the only academically acceptable benchmark. It would be worth investigating each model's performance under a different metric given the large disparity in AP score. The second reason is that the disparity may simply be a result of the data itself. To robustly test these models on the problem of fraud detection would require robustness checks on more real world data. Unfortunately, as mentioned numerous times, very little credit card transaction data is publicly available due to privacy concerns, and so the robustness checks must be explored in future work.

4.2 Interpretability

A key benefit of PLTR is the ability to more easily interpret the impact of decision rules and threshold effects on scoring. The first metric of this is model complexity, as defined by Dumitrescu, et al [2]. In one particular partition, all 100 trees generated by the random forest reached the max depth of 10 with an average of 49.82 leaves (terminal nodes) per tree. This corresponds to an average of 498.2 binary variables in PLTR with a maximum of 10 predicates. In contrast, PLTR selected on average 22.4 binary variables with a maximum of 2 predicates (the number of predicates is restricted to 2 by design). The ease of interpretability of PLTR is obvious.

# times selected	Decision rules	Average coef. value
10	V16≤-4.04, V6≤-2.88	-2.365
10	V17≤-2.75, V14≤-8.10	-2.001
10	V14>-5.84, V10≤-1.85	1.567
7	V18≤-3.74, V26≤-0.22	-1.302
10	V12>-4.56, V14≤-4.42	1.165
6	V16≤-4.04, V22≤-0.20	-1.051
8	V11>3.48, V26≤-0.27	-0.967
7	V17≤-2.75, V3≤0.39	0.860
10	V14>-5.84, V4≤1.04	-0.757
10	V12>-4.56, V11≤0.38	-0.722

The above table displays the first 10 decision rules with the greatest absolute average coefficient value. They also correspond to the first 10 features with the greatest absolute coefficient value, again underscoring the importance of threshold effects in predicting fraudulent transaction. Of note is the lack of univariate threshold effects in this list of 10 most meaningful decision rules. Regardless, it is very easy to extract meaning from these values. For example, the first decision rule V16≤-4.04, V6≤-2.88 has an average coefficient value of -2.365. A regulator can easily use this information to calculate the average marginal effect for this feature and can then report the percent likelihood of fraud decreasing when feature V16 ≤ -4.04, and feature V6 ≤ -2.88,

5 Conclusion

The empirical results of this work would suggest that PLTR is a robust model for binary prediction and classification tasks on problems where threshold effects can be reasonably assumed to exist in the population. The application of Dumitrescu, et al. [2] was credit scoring, and the application of this paper was fraud detection, but there are many other applications to which this method could be applied, and it would appear that PLTR is likely to deliver reliable predictive results.

A major area to be revisited and improved in this work lies in the simulated results. As mentioned, a trade-off between the number of observations and the rate of fraud was made while generating the data. However, maintaining such a low rate of fraud over a smaller sample size led to numerical instability. The optimal solution would have been to increase the number of samples, but that was not a computationally feasible option. A better compromise would have been to increase the rate of fraud. Regardless, the authors argue that the results of the simulation still illustrate the disparity between logistic regression and threshold models to detect fraud, and do not affect the results and conclusions drawn from the experiment on real data.

Perhaps the most important takeaway is the strong and reliable interpretability provided by PLTR in the simulated experiments. PLTR managed to identify the primary fraud decision rule and appeared to make good guesses towards the other secondary rules. While the interpretation of the real-world data is obscured by the PCA transformation, the stronger fit of the PLTR model on the data would lends support to the belief that the decision rules uncovered by PLTR are equally meaningful. Therefore, PLTR successfully accomplishes the goal of being an accurate and interpretable model.

6 Appendix: Additional Simulation PLTR Coefficients

#	Decision rules	Avg. value
10	TX_AMOUNT≤220.44,TERMINAL_RISK_7DAY≤0.18	-4.828
8	TERMINAL_RISK_7DAY>0.18,TERMINAL_RISK_30DAY≤0.04	2.446
10	TX_AMOUNT≤220.44,TERMINAL_RISK_1DAY≤0.75	-1.892
9	TX_AMOUNT>220.44	0.462
10	TX_AMOUNT	0.431

#	Decision rules	Avg. value
9	TX_AMOUNT≤219.69,TERMINAL_RISK_7DAY≤0.37	-2.740
10	TX_AMOUNT≤219.69,CUSTOMER_AVG_AMOUNT_7DAY≤186.68	-2.268
6	TX_AMOUNT≤219.69,TERMINAL_RISK_1DAY≤0.75	-1.047
6	TX_AMOUNT>219.69	0.827
10	TX_AMOUNT	0.538

References

[1] Dietterich, T. G. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation 10*, 7 (10 1998), 1895–1923.

[2] Dumitrescu, E.-I., Hué, S., Hurlin, C., and Tokpavi, S. Machine learning or econometrics for credit scoring: Let's get the best of both worlds. *SSRN Electronic Journal* (01 2020).

[3] Le Borgne, Y.-A., and Bontempi, G. *Machine Learning for Credit Card Fraud Detection - Practical Handbook*. Université Libre de Bruxelles, 2021.

[4] Lessmann, S., Baesens, B., Seow, H.-V., and Thomas, L. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research (doi:10.1016/j.ejor.2015.05.030)* (05 2015).

[5] Zou, H. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association 101*, 476 (2006), 1418–1429.