Performance Impact of Minority Class Reweighting on XGBoost-based Anomaly Detection

Altan Allawala, Anand Ramteke, and Pavan Wadhwa

***Abstract*—This paper explores the impact of reweighting the minority class of an imbalanced fraud dataset on the performance of an XGBoost binary classifier. Classifier performance is measured here in terms of true positive rate, false positive rate, precision, accuracy, AUC-ROC and AUC-PR. Our results suggest that reweighting the minority class has significant impact on these four key performance metrics when the classification threshold is held fixed and the model bias is not corrected. However, this impact becomes insignificant when (1) classification threshold is held fixed and the bias is corrected, or (2) when the target number of predicted positives is held fixed. Since fraud detection often prescribes a target number of cases for special treatment, these findings suggest that reweighting a dataset offers performance advantage only under very specific conditions for XGBoost-based classifiers. These conclusions can also generalize to problems where certain resampling techniques are used instead of reweighting since the two approaches tend to converge for sufficiently large datasets.**

***Index Terms*—XGBoost, binary classifiers, class imbalance, reweighting, resampling, bias-variance tradeoff.**

I. Introduction

A common situation that arises with binary classification problems is where Type I and Type II errors have asymmetric costs. For example, it may be more problematic for a fraud detector to incorrectly miss a fraud (false negative, or Type II error) than it is to incorrectly flag a non-fraud (false positive, or Type I error) because the business cost of the former outweighs the cost of the latter.

A standard practice amongst teams developing such classifiers is to resample or reweight one of the classes (usually to upweight the minority class) with the intent of improving some measure of classifier performance. This skews the decision boundary in favor of the minority class, making this class easier to detect. Such practice is common with logistic regression and even tree-based methods such as decision trees and random forests. However logistic regression models can suffer from high bias because they are linear in the log odds and therefore the decision boundary is linear too. On the other hand, many tree-based methods often suffer from high variance because they use simple heuristics for controlling model complexity (or bagging, in the case of random forests). This study investigates the effect of reweighting the minority class of an imbalanced training dataset on certain key performance metrics of an XGBoost binary classifier.

XGBoost, or Extreme Gradient Boosting, is a type of supervised machine learning algorithm. It is widely used due to its accuracy and speed when working with a large amount of data in a distributed or memory-limited computing environment. The library is available as an open-source Python package[1], GPU-optimized for handling large datasets [1].

XGBoost adopts a more principled approach to controlling model complexity by formally including a regularization term in its objective function. This careful control of model complexity mitigates overfitting and generalizes well to unseen data [2]. Therefore, an appropriately regularized XGBoost model with sufficient number of estimators is able to adequately capture the highly nonlinear decision boundaries of minority classes such as fraud much better than other supervised learning techniques. We find accordingly that since reweighting a minority fraud class does not add any new information to the dataset, reweighting has little impact on XGBoost model performance for fraud detection problems.

We also note that upweighting an observation is formally equivalent to upsampling via duplication, while downweighting an observation is approximately equivalent to downsampling (modulo the slight information loss from discarding observations). Both these techniques, in turn, correspond to changing the ratio of the positive to negative class count. This equivalency allows the findings of this study to generalize to cases where the class imbalance ratio is changed simply via moderate resampling (so that the classes within the downsampled data are still representative).

Our findings suggest that reweighting the fraud class of a dataset has significant impact on XGBoost classifier performance when the classification threshold is held fixed. However, reweighting a dataset introduces a bias to the output of the classifier, where the bias is a function of the amount of reweighting applied. When this bias introduced by reweighting the dataset is corrected, we see little impact on XGBoost classifier performance. In addition, we also see little impact on XGBoost classifier performance when the target number of predicted positives is held fixed (due to, for example, cost constraints around manual processing of predicted positives).

In other words, substantial performance improvements in an XGBoost classifier on imbalanced fraud datasets should not be expected when the classification threshold is held fixed unless the user is willing to 1) introduce a bias (which may be appropriate if, for example, the cost of a Type 2 error is

Manuscript received June 16, 2021; revised December 27, 2021. This work was supported by J.P.Morgan Chase & Co.

The authors are with the MRG Machine Learning Center of Excellence at J.P.Morgan Chase & Co. in Manhattan, NY 10016, USA (e-mail: altan.allawala@gmail.com, anand.k.ramteke@jpmchase.com, pavan.wadhwa@jpmorgan.com).

[1] https://github.com/dmlc/xgboost

doi: 10.18178/ijmlc.2022.12.4.1093

significantly different from the cost of a Type 1 error), or 2) somehow introduce new information into the system.

The rest of the paper is organized as follows. Section II provides a survey of some of the latest work using XGBoost-based classifiers for anomaly detection, especially fraud problems. Section III describes the key metrics used to evaluate model performance in this study, along with a description of the XGBoost model and dataset. Section IV presents the results of model performance with minority class reweighting measured as a function of two different cutoffs: probability cutoff and percentile cutoff. We also consider a de-biasing technique to correct for the reweighting. A summary and discussion around future work is presented in Section V.

II. Related Work

Owing to its scalability and performance, XGBoost has been successfully applied to a broad scope of fields such as credit risk assessment [3], cancer diagnosis [4]-[8], epilepsy diagnosis [9] and pedestrian detection [10]. In [3], classifiers based on XGBoost, logistic regression, self-organizing algorithms and support vector machine are used to assess credit risk for financial institutions, with XGBoost found to outperform the other three classifiers. In [9], an XGBoost-based model is used in an epilepsy study to classify participants as healthy patients or patients with epilepsy. This classification is made using patterns of language networks identified using fMRI scans. In [10], a genetic search algorithm is used to optimize the hyperparameters of an XGBoost-based classifier which is used to detect whether an image contains a pedestrian. The model is found to outperform the benchmark model based on support vector machines. In addition, XGBoost is also the supervised learning method used by several winning teams of Kaggle competitions (see [11] and [12]).

Recent works ([13]-[14]) in intrusion/anomaly detection have reported good performance using model formulations that comprise of XGBoost-based classifiers. In [13], an XGBoost-based classifier is trained on a dataset whose size is reduced by a hybrid PCA-firefly algorithm. This proposed model is shown to outperform benchmark models based on PCA, random forest and support vector machines. In [14], an ensemble model containing an XGBoost classifier is trained on features identified using a Crow-Search algorithm. The proposed model is shown to have a superior precision, recall and accuracy compared to state-of-the-art models.

Since XGBoost is based on gradient boosting, it can rapidly learn outliers, making it appealing for datasets with highly imbalanced classes [15]. In fact, XGBoost has been found to perform particularly well for fraud detection use cases (see [16]-[19]). In particular, [16] provides a benchmark for various XGBoost-based classifiers trained on two imbalanced credit card datasets.

III. Methodology

A. Performance Metrics

The key performance metrics considered in this study are true positive rate (TPR or recall), false positive rate (FPR or Type I error), precision and accuracy, defined as:

$$TPR = \frac{TP}{P} = \left(1 + \frac{FN}{TP}\right)^{-1} \quad (1)$$

$$FPR = \frac{FP}{N} = \left(1 + \frac{TN}{FP}\right)^{-1} \quad (2)$$

$$\text{precision} = \frac{TP}{TP+FP} = \left(1 + \frac{FP}{TP}\right)^{-1} \quad (3)$$

$$\text{accuracy} = \frac{TP+TN}{P+N} = \left(1 + \frac{FP+FN}{TP+TN}\right)^{-1}, \quad (4)$$

where TP, FP, TN and FN are the number of true/false positives and true/false negatives respectively and P and N are the total number of positive and negative classes respectively. These four performance metrics are standard primary metrics for binary classifiers as most other non-global secondary metrics can be inferred from a combination of these. For example, true negative rate (TNR) and false negative rate (FNR or Type II error) are just the complement of FPR and TPR respectively. Furthermore, if all four of these metrics can be shown to be insensitive to the effect of reweighting the minority class, then most secondary metrics are expected to be insensitive too.

This study also examines the area under the curve (AUC) of the Receiver Operating Characteristic (ROC) and Precision-Recall (PR) curves. PR-AUC gives the average precision over the recall and ROC-AUC gives the probability that the classifier will rank a randomly chosen positive instance above a randomly chosen negative instance. Larger values indicate better classifiers.

B. XGBoost

XGBoost consists of an ensemble of learners (decision trees in this case), taking the form:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F} \quad (5)$$

where K is the number of learners (in this case, trees), $f_k(x_i)$ is the output (or score) of the k^{th} learner, and $\mathcal{F}$ is the functional space of f_k.

Unlike other ensemble decision tree models such as random forests, XGBoost is based on the gradient boosting algorithm, introduced by [20]. Therefore, each tree is trained and added sequentially as a function of the dataset and the output of the previous trees. The output of the model at step t is thus given by:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i). \quad (6)$$

This leads to the following form of the objective function to be optimized at step t:

$$obj^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t)} + f_t(x_i)\right) + \Omega(f_t) + const, \quad (7)$$

where n is the number of data points, l is a customizable loss function and Ω is a regularization term. A Taylor expansion of the loss function then leads to the familiar XGBoost objective function. For a detailed discussion, see [21].

To understand the impact on model performance of reweighting the minority class of an imbalanced dataset, five

XGBoost models are trained. Each model is trained separately on a dataset with a different reweighting factor on the minority class. All five models are then scored on the same (unweighted) validation set which reflects the dataset on which the model will be used.

These five models span 100X, 10X, 1X (unweighted), 0.1X and 0.01X weightings on the minority class. Since reweighting the minority class by 10X, for instance, is equivalent to decreasing the ratio of the majority-to-minority class by the same factor[2], these five choices of weightings correspond to different majority-to-minority class ratios. Since the amount of reweighting straddles four orders of magnitude (0.01X to 100X), a comparison of these models offers insight into the impact of reweighting on model performance for a wide range of weights.

Hyperparameters are held fixed across the five models. The choice of XGBoost hyperparameters used in this study are stated in Table I. The default hyperparameter values of the Python package were used for all other hyperaparameters.

TABLE I: XGBOOST HYPERPARAMETERS FOR THE FIVE MODELS

'num_round' = 600	'max_depth' = 3
'objective'='binary:logistic'	'gamma' = 1
'colsample_bytree' = 0.8	'alpha' = 1
'subsample' = 0.8	'eta' = 0.1

C. Dataset

The dataset used in our study consists of 160 explanatory variables and a target binary variable for fraud. The training and validation sets consist of 1.8M and 0.8M observations respectively. The dataset represents a type of credit card fraud whose characteristics are left vague to maintain confidentiality. But importantly, it is a highly imbalanced dataset with a positive class ratio of around 2.7%. This corresponds to a non-fraud : fraud ratio of approximately 36.5:1 and so the four reweighting factors of 100X, 10X, 0.1X and 0.01X correspond to a non-fraud to fraud ratio of approximately 0.365:1, 3.65:1, 365:1, and 3650:1 respectively.

IV. EXPERIMENTAL RESULTS

A. Classifying Predictions by Probability Cutoff

We begin by investigating the effect of reweighting the minority class on the four metrics as a function of probability cutoff, also known as classification threshold. Classification threshold here is defined as the probability value above which an observation is classified as a member of the positive (minority) class. For example, a classification threshold of 0.6 means that only those observations with a model probability of 0.6 or higher are assigned a positive class prediction. As this classification threshold increases, a higher probability is required for any observation to be assigned a positive class and hence the number of positive class predictions declines. This is demonstrated in Fig. 1.

Fig. 1 shows that upweighting the minority class monotonically increases both TPR (recall) and FPR as a function of classification threshold. This behavior is expected because upweighting the minority class effectively introduces a positive bias in the classifier, making it more likely that both negative and positive cases are classified as positive. This increases both the true and false positive rates.

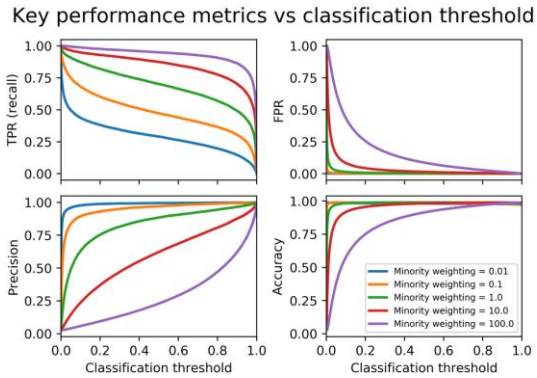

Fig. 1. Comparison of primary performance metrics of five (biased) models at varying classification thresholds (probability cutoffs).

Further, although both the true and false positive rates increase, Fig. 1 also shows that minority class upweighting at a given classification threshold causes a larger fractional increase in the false positive rate than the true positive rate. Combining (1) and (2) gives:

$$\frac{FP}{TP} = \alpha \frac{FPR}{TPR} \tag{8}$$

where $\alpha > 0$ is the class imbalance, defined as the ratio of negative to positive observations (or 36.5 in the case of 1X weighting). Therefore minority class upweighting increases the FPR/TPR ratio, and hence also the FP/TP ratio, which from (3) leads to the observed decrease in the precision of the classifier at a given classification threshold.

Finally, although introducing this positive bias to the model increases the absolute number of predicted positives, it decreases the absolute number of predicted negatives by an even larger amount because the high degree of class imbalance within the dataset means that there are far more negative than positive cases. Therefore from (4) it is evident that the decrease in $TP + TN$ leads to a drop in accuracy when the minority class is upweighted. This is demonstrated in Table II.

TABLE II: KEY PERFORMANCE METRICS AT A CLASSIFICATION THRESHOLD OF 0.6 FOR THE 0.01X, 1X AND 100X MODELS

Performance metric	0.01X model	1X model	100X model
TPR	26.3%	66.4%	94.0%
FPR	0.0%	0.2%	6.2%
Precision	99.6%	90.6%	29.1%
Accuracy	98.0%	98.9%	93.8%

Therefore at a given classification threshold, the positive bias introduced by upweighting the minority class causes a monotonic increase in TPR and FPR while generally decreasing the precision and accuracy.

However, an "apples-to-apples" comparison between the five models should first correct for the varying degrees of model bias that is artificially introduced by the different upweighting factors. A common way to correct for this bias (as in [22]) is to scale the minority class odds by the amount of reweighting applied:

[2] https://github.com/dmlc/xgboost/issues/144

$$p = \frac{p'}{p' + (1 - p') * \beta} \quad (9)$$

where p is the unbiased (corrected) probability, p' is the biased (uncorrected) probability and β is the reweighting factor applied to the minority class. Under this transformation, the corrected probabilities lead to the performance results shown in Fig. 2.

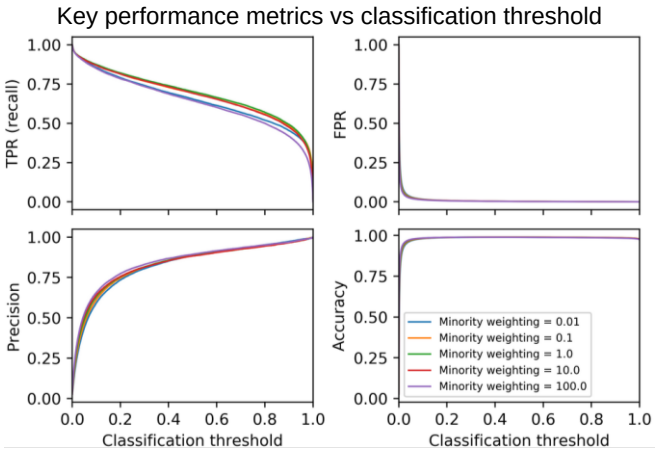

Fig. 2. Comparison of primary performance metrics of five de-biased models at varying classification thresholds (probability cutoffs).

Therefore once the bias has been corrected, neither upweighting nor downweighting the minority class has substantial impact on performance metrics when classification threshold is held fixed.

B. *Classifying Predictions by Percentile Cutoff*

Many use cases require a fixed target number of predicted positives. This is done by selecting the observations with the largest probabilities. In the case of fraud, each positive case usually needs to be manually reviewed and this target number is set by business based on available resources and funding constraints. Therefore, in such instances it is more relevant to investigate the relationship between the key performance metrics and the reweighting factor while fixing the percentile cutoff rather than the probability cutoff.

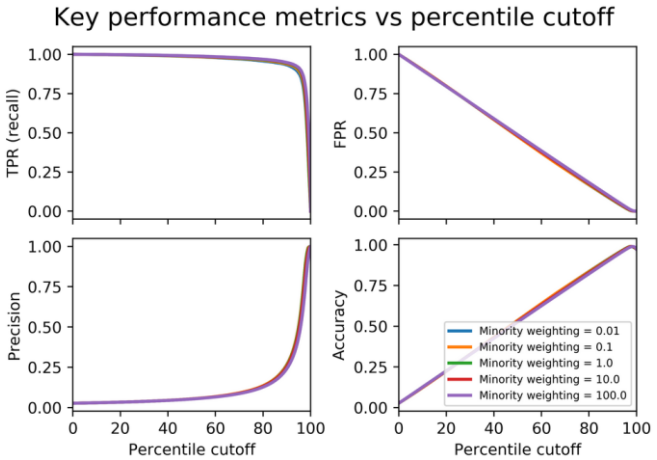

Fig. 3. At a fixed percentile cutoff, upweighting the minority class has negligible impact on TPR, FPR, precision and accuracy of the classifier.

Unlike probability cutoff**,** percentile cutoff is linearly proportional to the total number of negative predictions. For example, a percentile cutoff of 90% means that the classifier assigns a positive prediction to the top 10% highest probability cases and assigns a negative prediction to the rest. As percentile cutoff increases, fewer observations get classified as positive. Fig. 3 shows the four performance metrics as a function of the percentile cutoff.

In contrast to Fig. 1 which shows that the performance metrics change monotonically with increasing upweighting of the minority class when bias is uncorrected, Fig. 3 shows that when the percentile cutoff is fixed, the four performance metrics are largely insensitive to the minority reweighting factor. Note that this insensitivity persists even after correcting for bias because the monotonicity of the bias-correction transformation (9) maintains the same rank order of the probabilities.

It is also instructive to examine the area under the curve (AUC) of the Receiver Operating Characteristic (ROC) curves and Precision-Recall (PR) curves. Given the relative insensitivity of the four performance metrics to the minority reweighting factor, the AUCs are expected to not change substantially with changing reweighting factors. This is confirmed in Fig. 4. The difference in AUC performance between the five models is less than 2%**.**

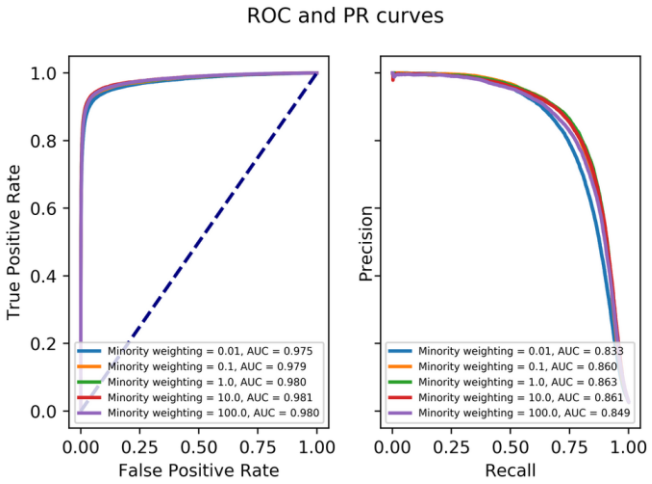

Fig. 4. ROC curve and PR curve of the five models with different minority class reweighting factors.

As a final comparison, we examine the rank order correlation between the five classifiers, based on the five separate weightings used. The rank order correlation provides a measure of the correlation between each classifier's rankings of fraud likelihood for all observations. Fig. 5 shows that, as expected, the correlation decreases monotonically as the difference in reweighting factors between two classifiers increases**.**

However the rank correlation is very high (approximately 0.9) even for models with reweighting factors that differ by four orders of magnitude (i.e. 0.01X and 100X). This high rank correlation implies that mostly the same observations are being classified as positive in the two models, which effectively leads to very similar model performance for different classifiers when the percentile cutoff is held fixed.

Since the XGBoost models have a sufficient number of estimators whilst being sufficiently well-regularized, all five models are adequately capturing the varying degrees of minority class frequency within the data. This leads to the high rank correlation between all five models.

As before, note that reweighting is not introducing any new information to the system. Also, the de-biasing formula (9) monotonically transforms the predicted probability of an observation being positive without changing the rank ordering of the observations. Therefore these results generalize to cases where the model biases introduced by reweighting have not been corrected.

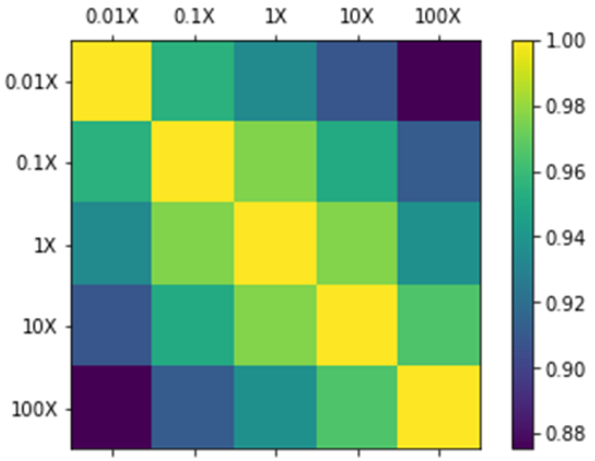

Fig. 5. Although the rank correlation between classifiers decreases monotonically as the difference in weightings between the classifiers increases, the correlations remain very high which leads to very similar model performance at a fixed percentile cutoff.

V. Conclusion

XGBoost is both flexible and well-regularized. This allows for better treatment of the bias-variance tradeoff. Therefore an XGBoost model with optimized hyperparameters to control for both bias and variance is well-suited to handling highly imbalanced datasets. As such, reweighting the minority class has little impact on model performance. Specifically, we find that:

1. At a fixed probability cutoff (classification threshold), increasing the weight of the minority class introduces a bias in the classifier which causes the TPR (recall) and FPR to generally increase and precision and accuracy to decrease. However when this bias is corrected, reweighting has little impact on key performance metrics.
2. At a fixed percentile cutoff, the effect on key performance metrics of reweighting the minority class is again negligible. This also includes ROC-AUC and PR-AUC. Given cost constraints, percentile cutoffs are more relevant than classification thresholds from a business perspective. Therefore reweighting a training set with the aim of improving classifier performance is unlikely to offer any performance benefit to a business.
3. As the difference in the minority class reweighting factor between two models increases, their rank order correlation declines monotonically, though the decline is marginal.
4. To establish the generalizability of the above conclusions, the same analysis was separately reproduced on a different fraud dataset with an even more imbalanced class ratio of 0.8% using a different (random) choice of hyperparameters.
5. Since upweighting an observation is formally equivalent to upsampling it, while downweighting an observation is approximately equivalent to downsampling it (modulo the slight information loss caused by discarding observations), the conclusions from this study may be generalized to other XGBoost models where the minority class is upsampled or the majority class is moderately downsampled.

XGBoost is one of several gradient boosting algorithms. Other such algorithms include LightGBM [23], CatBoost [24] and AdaBoost [25]. These algorithms work in a way similar to XGBoost but with key differences. For example, LightGBM uses Gradient-Based One-Sided Sampling (GOSS) to find the optimum split points; CatBoost (short for Categorical Boosting) specializes in categorical features; and Adaboost (short for Adaptive Boosting) modifies the sample distribution by weighting the data points for each iteration. Since these techniques, like XGBoost, use boosting to focus the subsequent weak learners disproportionately on misclassified observations, their performance is likely similarly insensitive to minority class reweighting. Future scope of the current work involves quantifying this performance impact by performing a similar analysis on these gradient boosting-based algorithms.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

A. Allawala, A. Ramteke and P. Wadhwa formulated the study. A. Allawala built the XGBoost classifiers and conducted the analysis. All authors wrote the paper and have approved the final version.

References

[1] M. Rory and E. Frank, "Accelerating the XGBoost algorithm using GPU computing," *PeerJ Computer Science*, vol. 3, p. e127, 2017.
[2] Bhati, B. Singh, G. Chugh, A.-T. Fadi, and N. S. Bhati, "An improved ensemble based intrusion detection technique using XGBoost," *Transactions on Emerging Telecommunications Technologies*, p. e4076, 2020.
[3] Y.-C. Chang, K.-H. Chang, and G.-J. Wu, "Application of eXtreme gradient boosting trees in the construction of credit risk assessment models for financial institutions," *Applied Soft Computing*, vol. 73, pp. 914-920, 2018.
[4] D. Komura, S. Ishikawa, S.-P. Cheng *et al*., "A benchmark for comparing precision medicine methods in thyroid cancer diagnosis using tissue microarrays," *Bioinformatics*, vol. 34, no. 10, 2018, pp. 1767-1773.
[5] Z. Xuan, T. J. Li, J. Wang, J. Li, L. Chen, and C. N. Liu, "Identification of cancer-related long non-coding RNAs using XGBoost with high accuracy," *Frontiers in Genetics*, vol. 10, p. 735, 2019.
[6] D. P. Yu, Z. D. Liu, C. Y. Su *et al*., "Copy number variation in plasma as a tool for lung cancer prediction using Extreme Gradient Boosting (XGBoost) classifier," *Thoracic Cancer*, vol. 11, no. 1, pp. 95-102, 2020.
[7] B. Siddharth, Y. Sinha, and L. Goel, "Lung cancer detection: A deep learning approach," *Soft Computing for Problem Solving*, Springer, Singapore, pp. 699-705, 2019.
[8] X. Y. Deng, Y. Luo, and C. Wang, "Analysis of risk factors for cervical cancer based on machine learning methods," in *Proc. 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS)*, 2018, pp. 631-635.
[9] L. Torlay, P.-B. Marcela, E. Thomas, and M. Baciu, "Machine learning–XGBoost analysis of language networks to classify patients with epilepsy," *Brain Informatics*, vol. 4, no. 3, pp. 159-169, 2017.
[10] J.Yu, G. X. Tong, H. N. Yin, and N. X. Xiong, "A pedestrian detection method based on genetic algorithm for optimize XGBoost training parameters," *IEEE Access*, vol. 7, pp. 118310-118321, 2019.

[11] N. Didrik, "Tree boosting with xgboost-why does xgboost win "every" machine learning competition?" Master's thesis, NTNU, 2016.
[12] A. Omar and K. Belkhayat, "XGBoost and LGBM for Porto Seguro's Kaggle challenge: A comparison," *Preprint Semester Project*, 2018.
[13] B. Sweta, P. K. R. Maddikunta, R. Kaluri, S. Singh, T. R. Gadekallu, M. Alazab, and U. Tariq, "A novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU," *Electronics*, vol. 9, no. 2, p. 219, 2020.
[14] S. Gautam, N. Deepa, B. Prabadevi, and M. P. K. Reddy, "An ensemble model for intrusion detection in the Internet of Softwarized Things," in *Adjunct Proc. the 2021 International Conference on Distributed Computing and Networking*, 2021, pp. 25-30.
[15] Z. X. Zhao, H. Peng, C. W. Lan, Y. Zheng, L. Fang, and J. Y. Li, "Imbalance learning for the prediction of N 6-Methylation sites in mRNAs," *BMC Genomics*, vol. 19, no. 1, pp. 1-10, 2018.
[16] P. C. Victoria and D. P. Prabha, "Influence of optimizing XGBoost to handle Class Imbalance in credit card fraud detection," in *Proc. 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020, pp. 1309-1315.
[17] C. Z. Meng, L. Zhou, and B. S. Liu, "A case study in credit fraud detection with SMOTE and XGBoost," *Journal of Physics: Conference Series*, vol. 1601, no. 5, p. 052016, IOP Publishing, 2020.
[18] Majhi, S. Kumar, S. Bhatachharya, R. Pradhan, and S. Biswal, "Fuzzy clustering using salp swarm algorithm for automobile insurance fraud detection," *Journal of Intelligent & Fuzzy Systems*, vol. 36, no. 3, pp. 2333-2344, 2019.
[19] S. Amit, R. K. Ranjan, and A. Tiwari, "Credit card fraud detection under extreme imbalanced data: A comparative study of data-level algorithms," *Journal of Experimental & Theoretical Artificial Intelligence*, pp. 1-28, 2021.
[20] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, pp. 1189-1232, 2001.
[21] T. Q. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proc. the 22nd ACM sigkdd international Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785-794.
[22] A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi,, "Calibrating probability with undersampling for unbalanced classification," in *Proc. 2015 IEEE Symposium Series on Computational Intelligence*, IEEE, December 2015, pp. 159-166.
[23] G. L. Ke, Q. Meng, T. Finley, T. F. Wang, W. Chen, W. D. Ma, Q. W. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, vol. 30, pp. 3146-3154, 2017.
[24] Dorogush, A. Veronika, V. Ershov, and A. Gulin, "CatBoost: Gradient boosting with categorical features support," *arXiv preprint arXiv: 1810.11363*, 2018.
[25] F. Yoav, R. Schapire, and N. Abe, "A short introduction to boosting," *Journal-Japanese Society for Artificial Intelligence*, vol. 14, pp. 771-780, 1999.

Altan Allawala holds a PhD and MSc in theoretical physics from Brown University, USA and a BSc in physics from University of Melbourne, Australia.

He is the vice president at J.P.Morgan's MRG Machine Learning Center of Excellence where he is the project leader of a proprietary financial machine learning Python-based library. He also conducts fundamental research into machine learning topics relevant to the industry and hosts firmwide machine learning training.

Anand Ramteke holds a finance MBA from the Indian Institute of Management, Bangalore and a bachelor's degree in metallurgical engineering from the Indian Institute of Technology, Chennai.

He is the executive director at J.P.Morgan's model review group and is responsible for reviewing machine learning (ML) models and related research.

Anand spent 14 years building / reviewing ML models across industries and previously led ML development teams at American Express and AIG. Anand also holds a patent for ML solution built at Wolters Kluwer.

Pavan Wadhwa holds a PhD and MBA in finance from the University of Texas at Austin and a bachelor's degree in electrical engineering from the Indian Institute of Technology, Kanpur.

He is the managing director in the model risk group at J.P.Morgan where he is responsible for reviewing models related to deposits, fee/revenue and anti-money laundering. Additionally, he has established a Center of Excellence to validate Machine Learning models.

Pavan spent 15 years on various trading desks of J.P.Morgan in NY and London generating thematic and Relative Value trade ideas for traders and clients, eventually running global rates strategy. Prior to his current role, he was head of US interest rate strategy at Blackrock.