IEEE Transactions on Quantum Engineering

Received 15 August 2022; revised 5 October 2022; accepted 7 October 2022; date of publication 10 October 2022; date of current version 16 November 2022.

Digital Object Identifier 10.1109/TQE.2022.3213474

Mixed Quantum–Classical Method for Fraud Detection With Quantum Feature Selection

MICHELE GROSSI[1], NOELLE IBRAHIM[2], VOICA RADESCU[3], ROBERT LOREDO[4], KIRSTEN VOIGT[5], CONSTANTIN VON ALTROCK[5], AND ANDREAS RUDNIK[5]

[1]European Organization for Nuclear Research (CERN), 1211 Geneva, Switzerland
[2]IBM Quantum, IBM, Markham, ON L3R 9Z7, Canada
[3]IBM Quantum, IBM Deutschland Research & Development GmbH, 71032 Böblingen, Germany
[4]IBM Quantum, IBM Corp, Coral Gables, FL 33134 USA
[5]IRIS Analytics GmbH, 56182 Urbar, Germany

The authors Michele Grossi, Noelle Ibrahim, Voica Radescu, Kirsten Voigt, and Constantin von Altrock declare that they are authors of patent pending entitled: "Mixed quantum-classical Method for Fraud Detection With Quantum Feature Selection" Nr. P202105918US01 filed on 12/10/2021. The authors declare that there are no competing interests. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team. (M. Grossi, N. Ibrahim, and V. Radescu are primary contributors.)
Corresponding author: Voica Radescu (e-mail: voica.radescu@ibm.com).

The work of Michele Grossi was supported by CERN Quantum Technology Initiative.

ABSTRACT This article presents a first end-to-end application of a quantum support vector machine (QSVM) algorithm for a classification problem in the financial payment industry using the IBM Safer Payments and IBM Quantum Computers via the Qiskit software stack. Based on real card payment data, a thorough comparison is performed to assess the complementary impact brought in by the current state-of-the-art quantum machine-learning algorithms with respect to the classical approach. A new method to search for best features is explored using the QSVM's feature map characteristics. The results are compared using fraud-specific key performance indicators, i.e., accuracy, recall, and false positive rate, extracted from analyses based on human expertise (such as rule decisions), classical machine-learning algorithms (such as random forest and XGBoost), and quantum-based machine-learning algorithms using QSVM. In addition, a hybrid classical–quantum approach is explored by using an ensemble model that combines classical and quantum algorithms to better improve the fraud prevention decision. We found, as expected, that the results highly depend on feature selections and algorithms that are used to select them. The QSVM provides a complementary exploration of the feature space that led to an improved accuracy of the mixed quantum-classical method for fraud detection, on a drastically reduced dataset to fit current state of quantum hardware.

INDEX TERMS Feature selection, fraud detection, quantum, quantum kernel alignment, quantum support vector machine (QSVM).

I. INTRODUCTION

Over the past few years, the financial industry has seen a substantial growth in innovation, particularly in the field of artificial intelligence/machine learning (AI/ML) with respect to the payment industry in an effort to keep fraud losses contained [1]. The current challenges are those of finding the balance between the false positives where, if too common, could serve as a negative impact to a client's experience [2] and minimizing the monetary loss incurring by fraudulent transactions. Yet criminals are also constantly increasing their capabilities to deploy ever more complex fraud schemes at a rate difficult to keep up. Many have started using AI/ML to augment the efficacy of their attacks [3]. The payment industry defends itself in multiple ways, including more data from more sources are used, more behavioral features are extracted as inputs to the AI/ML models, and better machine learning models. This is an area where quantum computing could provide a disruptive improvement, in

particular by identifying features that lead to more accurate classification.

Quantum machine learning (QML) is an active field of research that seeks to take advantage of the capabilities of both quantum computers and machine learning techniques, adapting the latter to the strengths of the current state of the art in quantum computing. There are many examples that illustrate how quantum computing can be used for anomaly detection [4], [5], to train models [6], [7], and possibly enhance machine learning models, such as quantum support vector machines (QSVMs) [8], [9], quantum classifiers (QCs) [10], and quantum neural networks [11]. Much work has been conducted on synthetic and publicly available datasets from various domains, such as drug discovery [12], image classification [13], and computational sciences [14]. Comparisons have been made to the classical counterparts of the available QML algorithms [15]. In addition, when synthetic data are used for machine learning experiments, there have been provable advantages shown involving synthetic datasets when there is a lack of necessary data [16], [17].

In this work, we investigate the impact of quantum feature selection techniques versus classical feature selection techniques on the performance of the QML classifier. We consider that prefacing the classical feature selection to the application of a QML may eliminate some or all of the complex nuances in the relationships between features and outcomes that QML methods are thought to be able to detect. Finally, we compare the performance of QSVMs to state-of-the-art methods in fraud detection, such as random forest and XGBoost, using a "real-world" dataset of card payment transactions with real fraud marks. We also introduce the concept of mixed quantum/classical machine learning ensembles, and test these against the model performance of the purely classical and purely quantum approaches.

A. METHODOLOGY

The three industry methods being analyzed in this article using same initial dataset are as follows.

1) Domain expert created decision rules-based model (no machine learning).
2) State-of-the-art type AI/ML using boosted trees (i.e., random forest and XGBoost).
3) A QSVM-type model.

As experimentation and potentially later real-world deployment platform, we are using IBM Safer Payments software product. IBM Safer Payments is unique in providing real-time and offline monitoring of payment transactions with internally and externally AI machine learning models. We have first loaded the transaction data and computed the behavioral features. We then created the domain expert-based additional features within IBM Safer Payments. We exported the training data for the methods in 2) and 3) directly from IBM Safer Payments to assure compatibility of the model with input data. This is an important aspect when discussing integration. If the QSVM model is to be used with payment processor's production system, the integration with the IBM Safer Payments product is feasible due to the external model import capabilities already built in the product. However, additional considerations related to latency requirements should be accounted when discussing integration. This is not within the scope of this article.

B. PAYMENT FRAUD PREVENTION KEY PERFORMANCE INDICATOR (KPIS)

Payment fraud prevention relies on two specific KPIs: 1) (monetary) *hit rate*; and 2) *false alarm ratio*. The hit rate, typically reported as a percentage, is defined as the number (or value amount) of correctly flagged fraudulent transactions divided by the total number (or value amount) of fraudulent transactions.

The false alarm ratio is typically given as the ratio of false alerts to true alerts. Thus, if the model created ten false alerts for every true alert, it has a false alarm ratio of 10:1. Each false alert causes disruption of a customer's payment and triggers potential manual interaction with the customer both of which are mostly independent of the amount of the transaction.

When invoking machine learning classifiers for payment fraud prevention, these are for a binary classification (fraud versus nonfraud). A statistical measure of a model is given by *accuracy*, which is the number of classifications a model correctly predicts divided by the total number of predictions made. The accuracy key performance indicator (KPI) is meaningful only for a balanced class dataset. A better diagnostic of a binary classifier performance is through a receiver operating characteristic (ROC) curve. *Area under the ROC curve (AUC)* is one of the most important evaluation metrics for checking any classification model's performance. AUC represents the degree or measure of separability. We have adapted it to align better with the commonly used financial KPIs mentioned above so that the x-axis is the false alarm ratio (instead of standard false positive rate); therefore, throughout the text, we refer to this curve as modified ROC curve or ROC*.

Machine learning classifiers are used for generating a score. This score could be on an ordinal scale or it could be representing the predicted probability of the current transaction turning out to be fraudulent later. Since the real-time decision can only be to decline or not to decline a transaction, usually a threshold is applied to the score to make this decision.

II. INPUT DATASET

We are using a dataset of real-world payment transactions that comes from the European cross-border processing portfolio and consists of about 80% debit and 20% credit card transactions. This dataset contains a total of 2.4 million payment transactions. Each transaction is flagged as fraud or nonfraud, with a total of approximately 3000 transactions marked as fraudulent in the dataset. Importantly, the transaction data can be enriched with customer reference data and

with features built on the fly, as described in the following sections.

A. TRANSACTION AND CUSTOMER DATA

The dataset we work with has only 12 input attributes from transaction data, with additional two attributes from demographic data, the remaining ones are engineered through discovery techniques, as described next. It is usually possible to enrich the transaction information with demographic data available within the financial institution for the card holder that initiated the payment, usually referred to as "customer reference data" or "masterdata". Examples of reference data include customer data linked to an account or card number, additional information related to merchants, supporting technical data, such as the countries that correspond to card number ranges bank identification number/issuer identification number (BIN/IIN), IP addresses, etc.

B. ENGINEERED FEATURES

An important ingredient to a model is feature discovery. Engineered features, such as behavioral profiles, formed from the transaction inputs encapsulate meaningful information for classification problems. Profiles provide aggregated counts of totals and transaction frequencies over calendar periods or predefined time windows for every customer or card number that is indexed in the database. Since they encapsulate a history of a transaction fulfilling the counting conditions and certain patterns, these features provide a strong discriminating power between a fraudulent and nonfraudulent transaction.

In total, before invoking any preprocessing of data, we have 48 attributes. These attributes are of various data types, including categorical, string, integers, etc. Handling categorical data type is posing some challenges for machine learning classifiers, and it will require treatment as one-hot encoding techniques and/or clustering of the relevant values.

C. DATASETS FOR USE CASES

The aim is to compare the impact of different methodologies by analyzing the same input data. However, different methodologies may have different limitations. For example, human expertise and rule generator do not necessary require a balanced dataset in terms of fraudulent versus genuine records, whereas machine learning methods (classical or quantum) require balancing the set via undersampling methods. Moreover, when using QML, quantum hardware is limited in number of qubits and error rates, one needs to reduce data dimensionality considerably while maximally preserving the accuracy of the model.

Therefore, we conducted this study using the following three distinct data references that require different levels of preprocessing for each of the analyzed use cases.

1) *Full Dimensionality of a Dataset:* Only cleaned from redundant data and split into train and test. This dataset can be used by rule generator assisted by a fraud subject matter expert, even if it is highly imbalanced by the number of genuine records relative to the fraudulent ones.
2) Balanced dataset, with genuine transactions randomly undersampled (there are various methods to achieve this), together with treatment to handle the categorical data type. This dataset then can be optimally used by classical machine learning.
3) Drastically reduced data samples that will require multiple trials to avoid bias due to heavy undersampling. In addition, further normalization of all input data is applied to ease the translation to quantum feature mapping. This dataset is then used as for a direct comparison of the classical and quantum methods.

III. CLASSICAL FRAUD DETECTION MODEL

The process of creating decision models in fraud prevention systems, such as IBM Safer Payments, is invoked via a conventional rule-based approach or using machine learning techniques.

A. HUMAN AND INTERACTIVE EXPERT METHOD

For the assisted and automatic model generation, a hybrid logic rule generating algorithm is used. The algorithm creates rule sets condition by condition and rule by rule. The assisted model generation proposes the next generation step where the expert can then either accept the suggestion, modify the parameter of the proposed condition, or not follow the proposal at all and select an own condition. The automatic model generation assumes an acceptance of each proposal and stops only if a defined stop criterion is reached. A model generation uses statistical analysis to discover fraud patterns that can be aided by a human fraud expert.

The rule generator is based on a deterministic algorithm to search the dataset for specific fraud patterns. The rule generator can account for categorical attributes by default, and all attributes can be included without treatment for categorical attributes. No undersampling is necessarily needed, because the human method and rule generator are defining behavioral patterns with the focus to catch a fraudulent transaction. The rule generator model is trained on a training set and validated on a test set.

There are various ways to split data. However, in order to mimic the real-life situation where training is done on past data and prediction applied on new incoming transaction, data have been split chronologically as follows. The first 1.5 million records are used for training and the remaining records for testing. The count of records in fraud and genuine is given in Table 1 (1:1000 imbalanced data).

The results obtained in terms of fraud prevention KPI are hit rate and false alarm ratio and shown in Fig. 1, often referred to as a modified ROC curve. The modified ROC* curve is the primary metric to understand the performance of a model.

TABLE 1 Original Dataset Count, Balanced Dataset Count, and Drastically Reduced Dataset Count. Overall and Split Into Test and Train

Set	Label	Count	Train	Test
Original	0	2396689	1897850	498839
	1	3216	2150	1066
Balanced	0	1505	984	521
	1	993	515	478
Reduced	0	366	262	104
	1	232	137	95

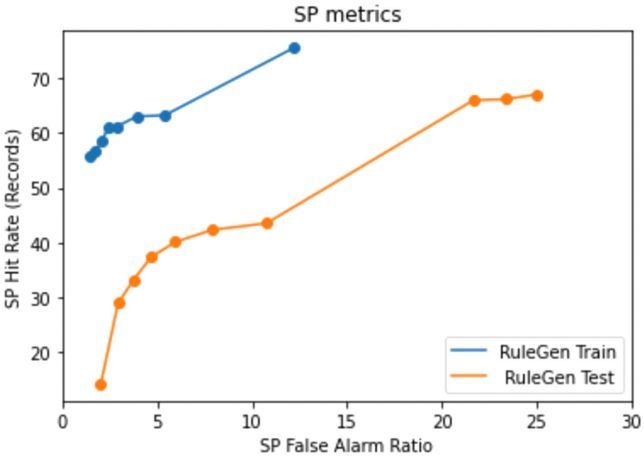

FIGURE 1. Results based on rule generator using a complete dataset that is split into training (blue) and verification (orange). The metrics are hit rate and false alarm ratio. This representation of the KPIs is often referred to as a modified ROC* curve, with the false alarm ratio on the *x*-axis (ratio false positives/true positives).

The KPIs of the model can be as good as 65% hit rate, but with the penalty of 25 nonfraudulent payments intercepted for each one fraudulent payment intercepted, or with 30% hit rate with only five nonfraudulent payments intercepted for each one fraudulent payment intercepted. This is a tradeoff decision to be taken by a processor or a bank. The train curve sits considerably above the test curve, which is an indication that the rule generator method is prone to overfit.

B. CLASSICAL MACHINE LEARNING CLASSIFIERS

Payment fraud detection is commonly using supervised machine learning classifiers where historical data have fraud marks either detected by a case investigator or reported by an affected customer and have ideally been caught by the fraud prevention models before it happened. Examples of supervised learning include regression, decision tree, random forest, support vector machine (SVM), logistic regression, etc. For this analysis, we have explored decision trees-based models, such as XGBoost and random forest, as they are known to outperform other supervised learning classifiers.

1) DATA PREPARATION FOR CLASSICAL CLASSIFIERS

There are mainly two drivers for data preparation for a classical classifier: a) balance the dataset; and 2) convert data types to numeric values. The following preprocessing steps have been applied to data before passing it to a classical classifier, apart from the undersampling.

a) Removal of highly correlated features (duplication of information).
b) *Treatment for Categorical Data Types:* Classify top categories where most fraud occurred in historical data and use them as separate features. This step has increased the number of features from 48 to 69.
c) Split of the data into "training" and "test" sets for use when training the models.
d) Treatment for imbalanced data where there is much more genuine than fraudulent transactions; achieved by undersample genuine by larger fraction than fraud with the aim to preserve all fraud marks. Finding the right balance is an art, we ran five random trials.

We first started with a complete dataset that has been split into test and training, as given in Table 1, and then on reduced dataset that has been used for the QML part as well.

TABLE 2 Accuracy and AUC Results for XGBoost and Random Forest Using Original Data, Without Undersampling

KPI	XGBoost	Random Forest
Accuracy (Train)	0.998	0.999
AUC (Train)	0.813	0.999
Accuracy (Test)	0.998	0.998
AUC (Test)	0.824	0.818

TABLE 3 Ordered Feature Importance for XGBoost and Random Forest

XGBoost	Random Forest
F_42	F_16
F_4	F_0
F_52	F_20
F_54	F_15
F_15	F_21
F_31	F_14
F_10	F_18

2) CLASSICAL MACHINE LEARNING WITH ORIGINAL DATASETS

We used *XGBoost CV* package for tuning the model's hyperparameters to find the best number of estimators, max_depth, and min_child, in a nonexhaustive iterative approach to find optimal values of these input parameters.

Similarly, for tuning random forest parameters, we used *RandomizedSearchCV* package where we identified the optimal number of trees needed in random forest, number of features to consider at every split, maximum number of levels in a tree, minimum number of samples required to split a node, and minimum number of samples required at each leaf node. We used random search of parameters, using three-fold cross-validation and searched across ten different combinations. The results of these two fits in terms of accuracy and AUC performance parameters are given in Table 2.

TABLE 4 Accuracy and AUC Results for XGBoost and Random Forest Using Balanced Dataset, With Undersampling. Therefore, We Took Five Trials

KPI	XGB1	XGB2	XGB3	XGB4	XGB5	Average
Accuracy (Test)	0.785	0.774	0.767	0.783	0.796	0.781 ± 0.010
AUC(Test)	0.832	0.837	0.823	0.852	0.845	0.834 ± 0.010

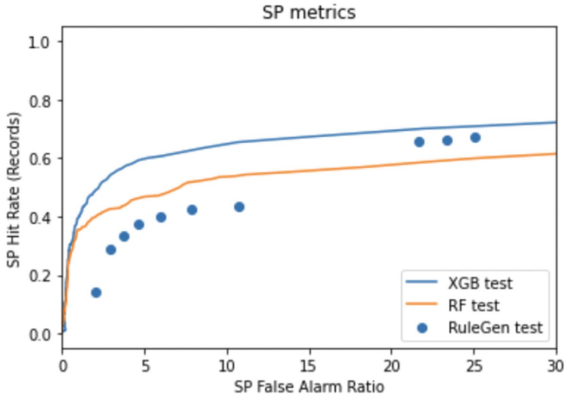

FIGURE 2. Results based on XGBoost, random forest, and rule generator using a complete dataset without categorical attributes that is split into training and verification. The metrics are hit rate and false alarm ratio.

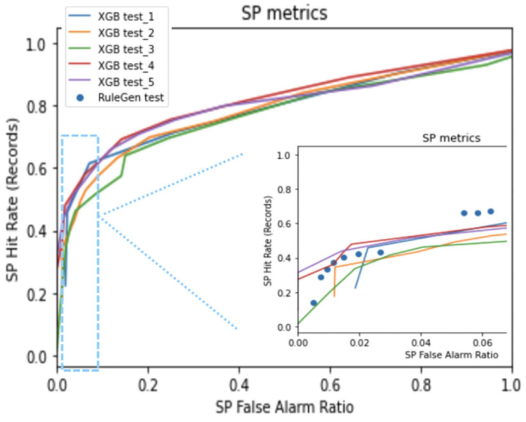

FIGURE 3. Results based on XGBoost multiple trials. The *X*-axis is false alarms; however, the range is equivalent to a factor of 500 more due to the undersampling. The snippet is a zoomed region of the plot that would be equivalent to real fraud alarm range of (0,100) and it is directly compared with results from the rule generator.

Interesting to observe is the list of feature importance ordered by the classical classifiers in Table 3. Each classifier has a different preference for the order of importance (impact) of the features. This is why choosing a different methodology for machine learning provides a different view of the feature space. This observation also prompted us to study the feature selection using QML where QML can complement classical methodology to improve the fraud KPIs.

To compare them with the rule generator, which uses a different KPI metric, we have also looked at the hit rate versus false alarms, as defined in the Safer Payments product. Fig. 2 overlays the modified ROC* curves from different machine learning models versus rule generator.

Fig. 2 shows the relation between hit rate (1 means 100% and 0 is 0% correctly identified fraudulent payments) on the *y*-axis, while *x*-axis represents the false alarm rate, meaning disturbed clients for each true fraud. The ideal case is a hit rate of 100% and a false alarm rate of 0. As we can see in this figure, reaching a hit rate of 50% of intercepted fraud would cause the false alarm rate to rise to more than $1:50$. This result, in fact, is better than using rule generator, capturing more fraudulent records than a model trained via rule generator. Similarly, XGBoost seems to outperform the random forest model for this dataset. However, the hit rate from XGBoost and random forest is on a record count basis, while from the rule generator is per amount value basis.

3) CLASSICAL MACHINE LEARNING WITH BALANCED DATASETS

For this part, to balance the set, we massively undersampled data by $1:1000$ for the genuine transactions, preserving a third of the fraudulent records. The train set has 1500 records, while the test has 1000, as given in Table 1. To minimize biasing effects from the strong undersampling, we have run the split in five separate trials, preserving the same fraudulent to genuine transaction ratio. The results are captured in Table 4, and an average is computed for reporting KPIs. We observe only small deviations across the trial runs for the KPIs, i.e., accuracy, AUC, and the dependency between hit rate versus false alarm rates for the test sample, as can be seen in the modified ROC* curves Fig 3. Notice that values on the *x*-axis are affected by the undersampling and should be scaled by the undersampling factor of 500 to be comparable with the ROC* curves from the original dataset. The zoom in the plot is aimed to help guide the eye to run that comparison; the "real" false alarm ratios in the range of 0–100 correspond to a range of 20%–60% hit rate for catching fraudulent records. Therefore, this is comparable with the previous results that did not use sampling, as it is shown in the zoomed plot, where dotted lines correspond to the results from rule generator using test data. To be noted that rule generator's hit rate has monetary value, while the hit rate from classical machine learning models is based on record count. This is an important validation of an undersampling procedure, having in view that we can only use significantly reduced datasets with the current state of quantum hardware.

Since the goal is to inquire as to whether or not a quantum advantage can be obtained over the best classical models when they are not restricted in terms of the number of features that can be accessed, we also invoked a recursive feature elimination method to extract a best XGBoost model. This is one of the standard methods used in industry to optimize

TABLE 5 Accuracy and AUC Results for XGBoost Extracted Using Best 37 Features on a Balanced Dataset, With Undersampling. Therefore, With Ten Trials

Model	Acc.	CI	Std	N_trials	N_features
XGBoost-7.	78.8%	0.5%	1.0 %	5	7
XGBoost-37	80.0%	0.5%	0.9 %	10	37

the number of features that add significance to performance of the model in terms of statistic KPIs. The recursive feature elimination method yields the best performing classical model on this dataset, finding 37 significant features out of total of 69. As this dataset is undersampled, to reduce the standard deviation of the averaged KPIs, we have used ten trials. The results are given in Table 5, which compares the XGBoost with seven features versus 37 features and more trials.

IV. QML FOR FRAUD DETECTION

There are various QML approaches for classification problems [8], [15], [18], [19]. In this article, we are focused primarily on the QSVM approach. The search for an increasingly high-performance model is the basis of every research project, and exploring the usage of quantum algorithms is a promising approach. The ultimate goal is to find a quantum kernel that provides an advantage in the classification of real-world data by improving a metrics as the classification accuracy. A general recipe for building these kernels is not yet available, except in specific cases, such as the definition of class of quantum kernels related to covariant quantum measurements, as the one introduced in [20], applicable to group-structured data. Those kernels can be optimized using a technique called *kernel alignment*.

The motivation for this work is to leverage the QSVM approach in the following two parts in order to optimize the fraud detection system.

1) The first is to determine which of the many features available should be selected to reduce the dimensionality of the dataset for running the experiment on a quantum system.
2) The second is to derive the fraud KPIs from the QML model. We used Qiskit [21] quantum software package for this work.

A. QUANTUM FEATURE IMPORTANCE SELECTION ALGORITHM

The main challenge in the near-term quantum devices is the limited number of qubits. According to the data encoding procedure adopted, which in this case is the one introduced in the QSVM paper [8] where each qubit is associated with a feature, we need to reduce the feature dimensionality of the original dataset to be managed on a real quantum device.

Not only the number of qubits, hence the number of features selected is important, but also is the number of records used for the training sample. Therefore, using undersampling techniques to scale down data is an important pr-requisite.

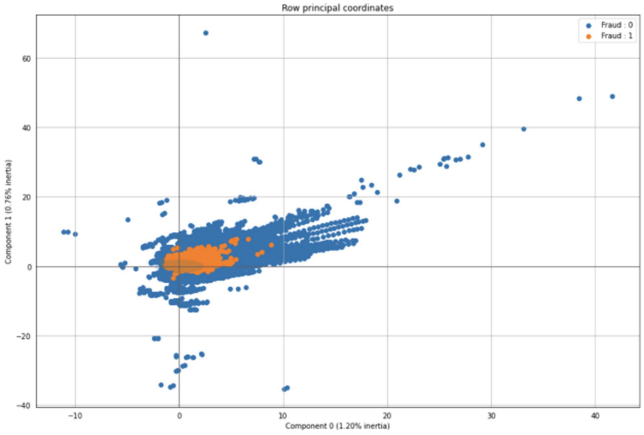

FIGURE 4. Relation between the components from the FAMD method. A total overlap is observed that limits the FAMD approach in using it for data feature reduction.

All data values are also normalized to the interval $[-1, 1]$ using MinMaxScaler package as a more convenient choice for quantum processing of those data mapped as angle rotations $[0, 2\pi]$

For the feature selection, we started by evaluating several classic methods to reduce data dimensionality for the number of features, from the classical principal component analysis (PCA) method types to the feature importance extraction from XGBoost or random forest on full dataset of 2.4 million records.

The data used for payment fraud prevention are mostly composed of binary or categorical data types, while the PCA method is designed for continuous variables and hence we could not use it. We experimented with the factorial analysis of mixed data (FAMD) method, which works for a mix of categorical and numerical variables. However, for this dataset, the method did not show any discrimination power between its reduced variables, displaying a total overlap, as shown in Fig. 4, for the first-two components.

As observed in Table 3, different features are preferred by different machine learning classifiers.

The performance of a classifier on a reduced dataset is driven by the choice of selected features. Different classifiers will favor different feature selection. Therefore, when comparing the performance of different classifiers using a subset of features, the optimal selection of features needs to correspond to the classifier's feature importance. At the time of writing, there is no inbuilt feature importance method for a quantum machine classifier, which may undermine the full performance of a QML model, in this case QSVM.

Therefore, instead of approaching variable reduction through purely classical techniques (which is best adapted to the classical machine learning), we developed a quantum algorithm that would allow use of quantum feature map and quantum kernels to determine best features.

The quantum feature map $\rho(\cdot) := |\psi(\cdot)\rangle\langle\psi(\cdot)|$ embeds a data point to a quantum state so that we can build the classification model, in particular the kernel function that measures

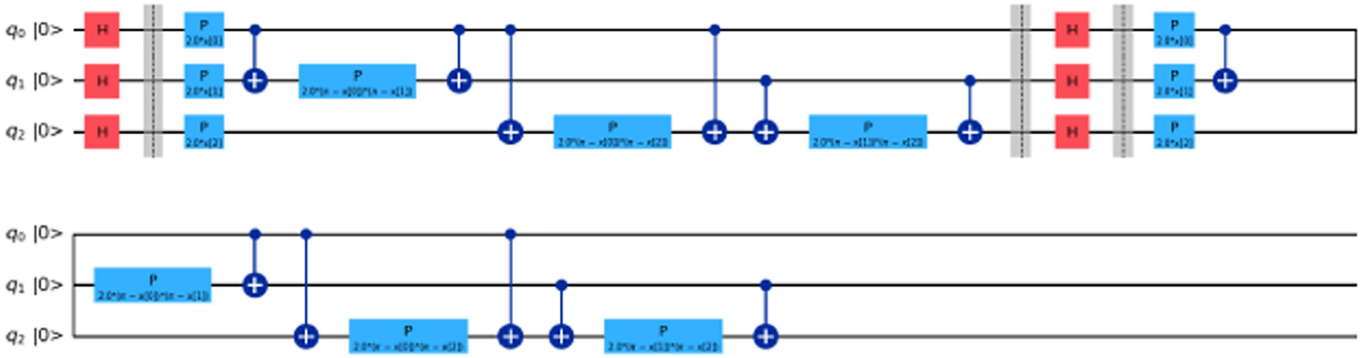

FIGURE 5. Example of a QSVM transpiled job circuit for three features.

the similarity between two data points $x,\ y \in \mathcal{I}$ in the Hilbert space with respect to the Hilbert–Schmidt inner product as

$$k(x_i, x) := \phi(x_i)^\dagger \cdot \phi(x) = \mathrm{tr}[\rho(x)\rho(y)] \equiv |\langle\psi(x)|\psi(y)\rangle|^2 \equiv \left|\langle 0|U(x)^\dagger U(y)|0\rangle\right|^2$$

where the quantum feature map is precisely the density matrix $\rho(\cdot)$, $U(\cdot)$ corresponds to a data encoding quantum circuit that represents the quantum feature map, and $|0\rangle := |0\rangle^{\otimes n}$. In our case, the *ZZ* quantum feature map is defined as $U_{\phi(x)} = \exp(ix_0Z_0 + ix_1Z_1 + i(\pi - x_0)(\pi - x_1)Z_0Z_1)$, where 0 and 1 are the qubit indexes. In terms of circuit representation, it is given by Hadamard gates at the beginning and in the middle of the circuit to create quantum interference, followed by a single qubit rotation around the *Z*-axis to encode each feature, and eventually a second-order expansion to account for interactions in the data, given by another single qubit rotations of generally the product of two features sandwiched between two controlled 2 qubit gate. As an illustration, a quantum feature map using three qubits is represented in Fig. 5.

The minimization of the objective function is realized on a classical device, while the kernel values are sampled from a quantum computer. With our training and testing datasets prepared, we proceeded by setting up the *QuantumKernel* class to calculate a kernel matrix using the *ZZFeatureMap*.

First, the application of the QSVM method has been tested and ran on a quantum simulator, under ideal condition, namely with a state_vector simulator, and then in a more realistic scenario with a noisy simulator, and eventually on the real device together with error mitigation. This is particular convenient because the first iterations are quite expensive in terms of calculation since the total number of permutations range from roughly half a million to thousands.

We have been inspired by the classic feedforward feature selection based on AUC or Accuracy as statistical metrics. In this way, we can iteratively select an increasing number of features in the problem, i.e., starting from 3 out of total of 69. This quantum approach is integrated and is part of the overall framework defined in Fig. 6 to approach the problem of fraud detection.

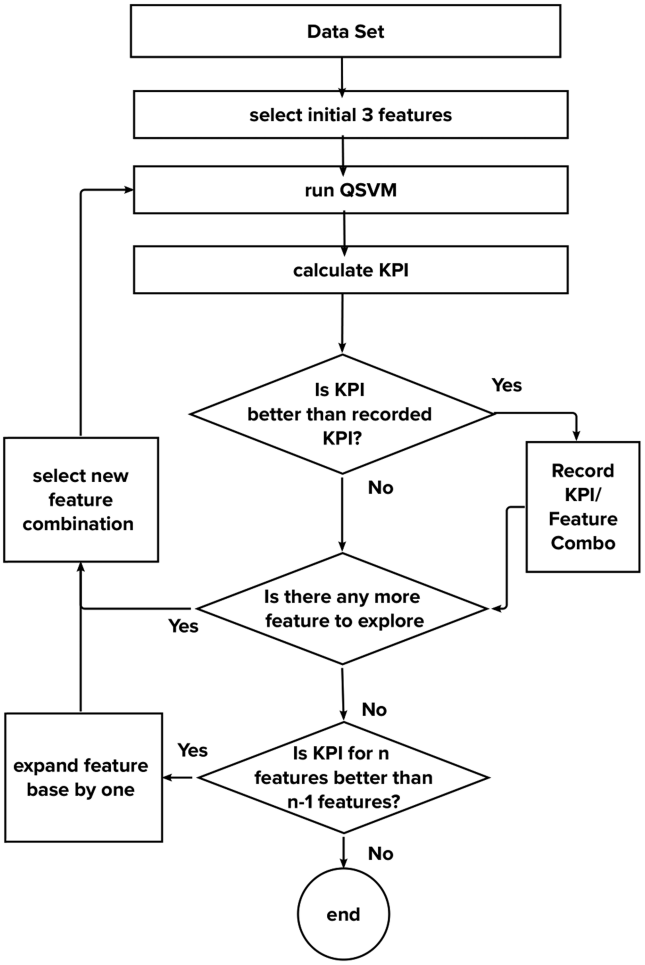

FIGURE 6. Schematic representation of the quantum algorithm feature selection to exploit the usage of quantum computing in the context of fraud detection.

Given a dataset $\mathcal{X}$ of dimension $n \times m$, where n is the total number of sample (transaction) and m is the total number of features, the algorithm does a permutation over all the possible combination of p (starting from 3) features over m. For each combination, a QC is defined, trained, and tested, and the accuracy and the AUC are stored. At the end of

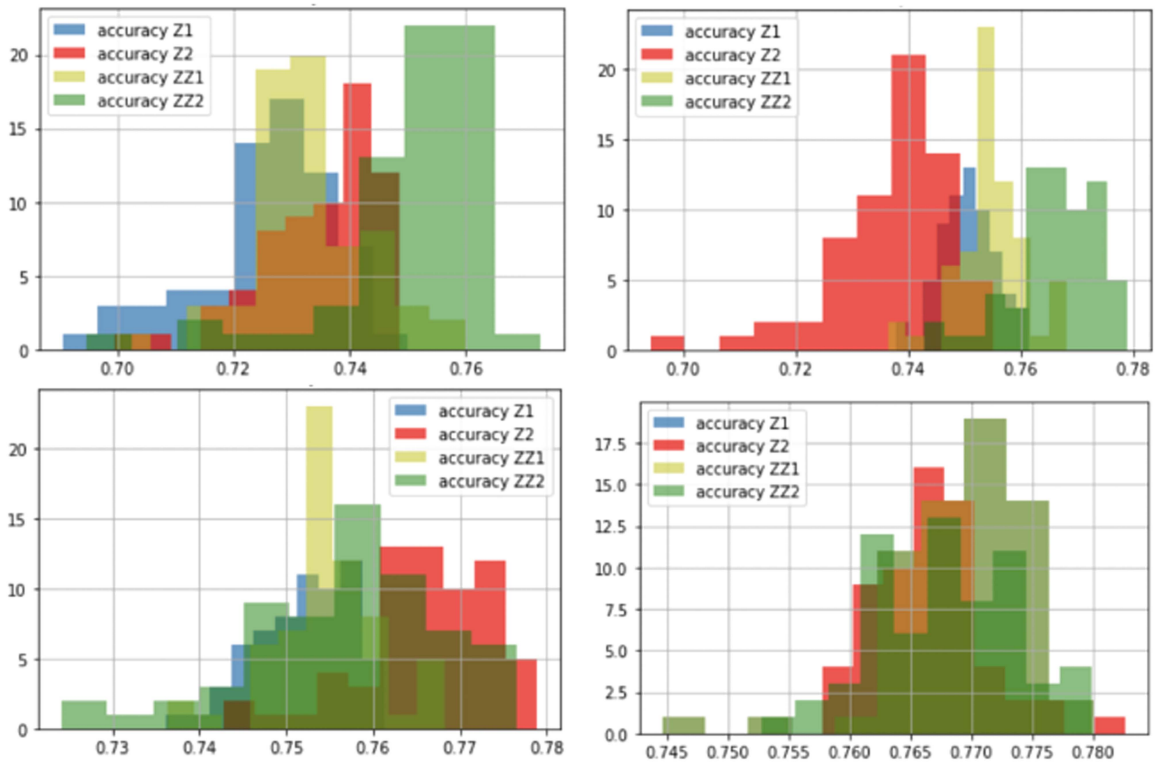

FIGURE 7. Spread in the accuracy values when each feature combination is explored for best three features, best four features (top row), best five features, and best six features (bottom row), respectively. This is done for each of the quantum feature map choices of *Z* map depths 1 and 2, as well as *ZZ* map depths 1 and 2.

the procedure, the best model based on the accuracy is key performance indicator (for future work, one could consider different KPI as the discriminator), and therefore the best three features out of 69 are chosen as a baseline for the next model iteration. This leads to exploration of few thousands of combinations (where repetition is not allowed). At this point, the fourth feature is chosen after a permutation over all the remaining features together with the previously selected (only 66 features are explored). The process can be iterated adding one feature for each permutation cycle up to the desired number of featured, preferably when the improvement saturates. This number can be chosen as a tradeoff between the maximum number of available qubit and the total accuracy obtained in the iteration.

Fig. 7 shows the spread in accuracy values at each of these feature selection stages where the best feature is selected at the maximal accuracy value for each of the quantum feature map.

Once the best features were identified, we have run the final iteration and double-checked performance and repeatability under noisy condition, targeting the execution of the algorithm on real hardware. Due to the abundance of replications and need of multiple trials, we scripted the flow to allow for running the quantum instance that controls the transpilation and execution of a circuit via many different parameters, such as the backend, for simulation the noise model, basis gates, coupling map, etc., and is quite useful when wanting to run under different data input conditions, including sample size, choice of data features. "training size": 1500, "feature size": 7, "test size": 1000, "order_of_expansion": "ZZ," "depth": 2, "entanglement": "full," "alpha": 2.0, and "n_shots": 8192.

B. QSVM RESULTS

The search of best features is performed using *Z* and *ZZ* quantum feature maps with depths 1 and 2. Fig. 8 provides an overview of the improvement observed in the accuracy when more features are added, as well as the preference towards the *ZZ* feature map with depth 2. Feature maps with more entanglement perform better. The more entanglement a feature map uses, the more difficult it is to simulate on classical hardware. As quantum hardware increases capacity in terms of qubits, the number if features that can be used will increase, which may lead to even further improved performance.

Another observation is that best features selected by this algorithm (as given in Tables 6 and 7) are different from the best features selected by the classical algorithms using same datasets, emphasizing a crucial role that feature exploration with QSVM plays to complement the feature space scanning.

Interesting observation is that the new algorithm has indeed identified features have least level of overlap, as shown in the correlation matrix in Fig. 9. This demonstrates that the choices are indeed viable. A reminder that most of the features have been engineered from same initial set of raw

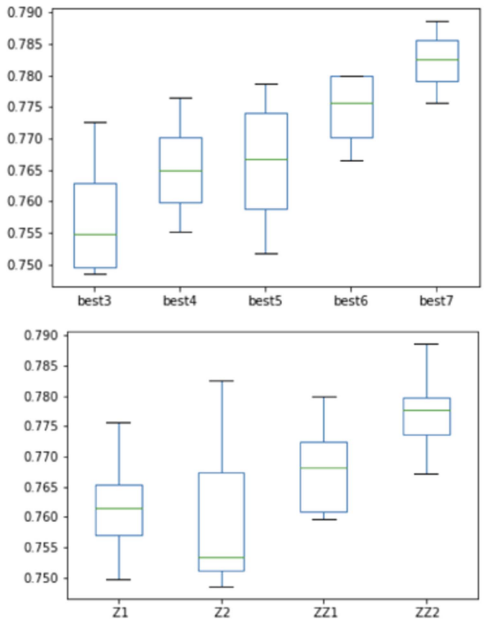

FIGURE 8. (a) Improvement of the accuracy as we add more features. (b) Preference towards the entangled feature map for better accuracy.

TABLE 6 Seven Best Features Selected With QSVM Feature Selection Algorithm Under Various *ZZ* Map and Depth Selections. Each Row Cell Corresponds to the First Set of Best Selected Features in Increasing Order of Added Features. The Defining KPI is Accuracy. The Best KPIs Have Been Found for the *ZZ* Depth 2

QSVM Best Features			
ZZ depth 2	ZZ depth 1	Z depth 1	Z depth 2
F_15 F_42 F_65	F_15 F_57 F_42	F_15 F_42 F_10	F_15 F_45 F_42
Acc= 0.772	Acc=0.759	Acc =0.749	Acc=0.748
+F_2	+F_55	+F_31	+F_3
Acc= 0.776	Acc=0.768	Acc =0.761	Acc=0.755
+F_38	+F_0	+F_7	+F_48
Acc= 0.778	Acc=0.772	Acc=0.761	Acc=0.751
+F_8	+F_3	+F_0	+F_19
Acc= 0.779	Acc=0.779	Acc=0.766	Acc=0.771
+F_64	+F_2	+F_0	+F_13
Acc= 0.788	Acc=0.786	Acc= 0.782	Acc= 0.775

inputs, so identifying independent meaningful features is not a trivial find.

We observe that using this new feature selection by QSVM improves the outcome of the model when compared to use of QSVM with best classical features from XGBOOST and random forest from Table 3. This comparison is performed using the exact same datasets with 1500 records used for the training and 1000 for testing. Due to data undersampling, we have used five random trials to minimize bias. The average KPIs for accuracy and AUC are reported in the Table 8.

Other KPIs, such as hit rate and false alarm ratio, have been compared with XGBoost and SVM. Even with a drastically reduced dataset, the results are compatible with the rule generator, which was using the full dataset, as shown in

TABLE 7 Ordered Feature Importance for XGBoost and Random Forest as in Table 3 With Additional Column Added for the QSVM Best Features for the *ZZ* Map With Depth 2

XGBoost	Random Forest	QSVM (ZZ depth 2)
F_42	F_16	F_15
F_4	F_0	F_42
F_52	F_20	F_65
F_54	F_15	F_2
F_15	F_21	F_38
F_31	F_14	F_8
F_10	F_18	F_64

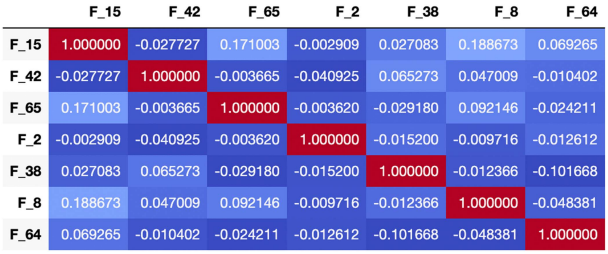

FIGURE 9. Correlation among best features selected by the QSVM method.

TABLE 8 Accuracy and AUC Results for QSVM Using XGBoost and Random Forest Best Feature Selection Versus QSVM Best Feature Selections, Based on the Balanced Dataset and Using Six Trials

KPI QSVM	w/ XGB bf	w/ RF bf.	w/ QSVM bf
Accuracy (Test)	0.76 ± 0.01	0.76 ± 0.01	0.78 ± 0.01
AUC(Test)	0.81 ± 0.01	0.81 ± 0.01	0.81 ± 0.01

TABLE 9 KPIs for Test Samples When Running QSVM on Different Backends: State Vector Simulator, Qasm Simulator With and Without Noise, and IBM Quantum Systems With and Without M.E.M. Enabled

backend	Accuracy	AUC
statevector sim.	0.78 ± 0.01	0.81 ± 0.01
qasm sim. w/o noise	0.77 ± 0.03	0.79 ± 0.05
qasm sim. w/ noise	0.55 ± 0.10	0.74 ± 0.14

Fig. 10. This validates data reduction method used for this analysis.

We ran the model using different backends available on the IBM Quantum platform. As explained, the workflow was to start with the ideal simulator (state_vector). We used this simulator for running the algorithm to determine best features. Once best features were found, we repeated the run on qasm-simulator. The standard deviation is estimated from repeating the run on six trials. The KPI that we recorded for the case without accounting for noise is close to the one we encountered with the state vector simulator, as shown in Table 9. While many opportunities exist to use quantum computing systems, there are many facets that go along with it, such as software, cloud access, benchmarking, and error correction and mitigation [22], [23]. Since the current state of the art for these systems is still considered noisy, we wanted to run our simulation tests with noise models, which are based off real quantum systems [24]. We have used the noise source analyzed for the ibm_cairo backend

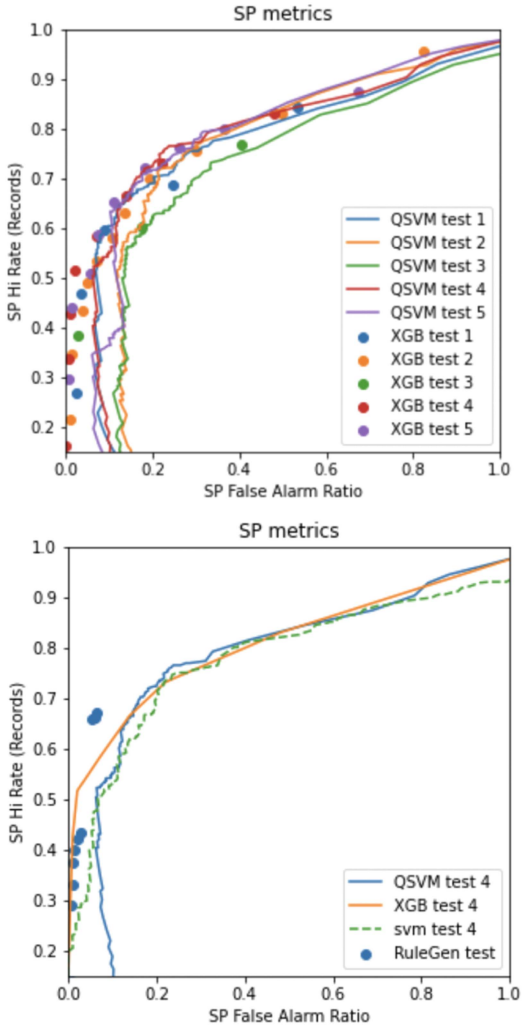

FIGURE 10. Modified ROC* curve for QSVM using state vector simulator.

and rerun the QSVM and enabled the readout measurement error mitigation flag. When using noisy simulation, the optimal circuit transpilation pipeline in Qiskit is provided with the parameter optimization_level set to 3, which selects a candidate initial_layout and SWAP mapping using the Sabre layout and routing method [25], and performs the most 1Q and 2Q gate optimizations.

The circuit depth produced by the seven best features is around 70, an illustration of a circuit depth diagram produced by three features is shown in Fig. 5. Reducing the depth of the circuit and optimize it for use on the hardware is a study for future work that is in plans.

V. MIXED QUANTUM-CLASSICAL METHOD FOR FRAUD DETECTION

Although quantum computing has been proven to speed up some types of problems [26], the existent technology allows only a limited number of qubits and gate operations. Therefore, we employ a hybrid classical/quantum solution where a classical and a quantum algorithm are stacked together in a heterogeneous ensemble. This kind of hybrid

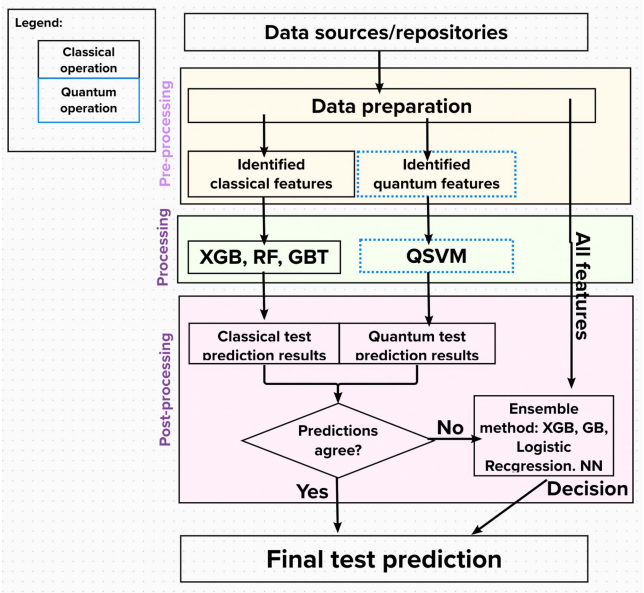

FIGURE 11. Flow chart of how to combine the classical and quantum approaches into a single decision. When the quantum and classical algorithms disagree, a Metaclassifer predicts which one is correct.

quantum/classical ensemble approaches are explored in the optimization problems, with some examples provided in [27] and [28]. This is the first time that such a method has been explored for a QML classification problem. We note that the method is relatively easy to implement and simply uses out of the box classical and QCs, such as would be used by data scientists practicing in industry today.

A. APPROACH

To exploit the complementarity of the quantum and classical machine learning models for improving the classification performance, we employ a metaclassifier to discern the classification of those transactions for which the classifiers disagree. To accomplish this, we trained both the quantum and classical algorithms on the balanced dataset, as given in Table 1. When the two classifiers disagreed on the label of a given transaction in the training set, the transaction was noted. These transactions, a subset of the training data of the balanced dataset, formed an additional dataset on which a metaclassifier was subsequently trained. The metaclassifier may take as features any of the features from the dataset. In practice, because of the size of dataset, the number of training datapoints on which the classifiers disagreed was limited, so a simple metaclassifier performed best; for the XGBoost model with 37 features, a classical SVM was used as the metaclassifier. The flowchart of the approach is shown in Fig. 11.

For this work, the list of metaclassifiers was not exhaustive, and there is opportunity for potentially obtaining better performance uplift for the ensemble by expanding this list of metaclassifier candidates.

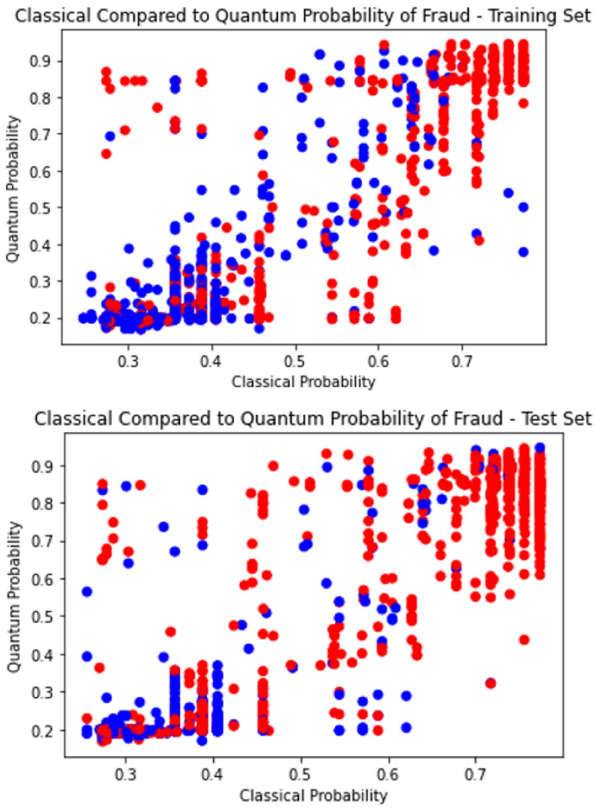

FIGURE 12. **Complementarity of the classical and quantum decisions. Red dots are fraudulent transactions and blue dots are genuine transactions. The position on the *x*-axis represents the probability of fraud predicted by the classical algorithm. The position on the *y*-axis represents the probability of fraud predicted by the quantum algorithm. When classifier agrees, the dots align on diagonal. The further is the distance from diagonal, the bigger the disagreement.**

TABLE 10 **Comparison of Model Accuracy on the Balanced Dataset: Mixed Models**

Model	Acc.	CI	Std	N_trials
XGBoost-37.	80.0%	0.5%	0.9%	10
QSVM - ZZ	78.8%	0.4%	0.7%	10
QSVM + XGBoost-37	81.0	0.3%	0.5%	10

B. RESULTS AND COMPARISONS

While the performances of the classical and quantum algorithms were similar, the actual predictions can vary for specific data points, yielding complementary results (see Fig. 12). Classifications disagree on 5.2% of training data and 5.5% of test data, with classification threshold of 0.5. On the diagonal, the quantum and classical models agree, and on the off-diagonal, they disagree. This shows that different relationships are detected by the quantum and classical models. We exploit this complementarity to increase performance using a metaclassifier, which determines which algorithm to "believe", given the surrounding circumstances as expressed by the features of a given transaction.

The results of both these classical classifiers and ensemble methods employing are presented in Table 10. It can be seen that even though the quantum model was restricted to use only seven features, while the classical models had access to all significant features, the quantum model could still add performance value in the mixed ensemble. The effect size was a small 1% (or 5% reduction in error rate or fraud loss rate), however, it was statistically significant.

Even if the QSVM approach was not optimized over all possible hyperparameters, such as exploration of a large variety of possible feature maps, we note that it may increase performance of a fully optimized XGBoost model with access to an optimal number of features, showing the potential towards quantum advantage over the best classical methods. It should be noted, however, that this, of course, is an observation on a particular real-world dataset and not a mathematical proof. Future work could explore pushing the boundaries on the number of features used by both QSVM and classical models on a dataset with a larger number of features. The trend shown in Fig. 8 suggests that performance improvements will continue to occur as more features are added to the quantum method.

VI. CONCLUSION

Classical machine learning algorithms are currently state of the art for predicting fraud in transactions. QML can provide a complementary support on this, exploiting enhanced feature space to encode historical data. In this work, we proposed a novel approach to maximize a QC performance in terms of accuracy of prediction, but other KPI can be explored as well. The method is called the quantum feature importance selection algorithm; using quantum-enhanced support vector machine, we were able to select most relevant features for the QC for an increasing number of selected features. In this case, we also noted that quantum feature map that makes use of more entanglement provides systematically better KPIs. The whole workflow requires quite an intensive care for data preprocessing from data type considerations to undersampling techniques before moving to the quantum part. We found that QC can identify different types of patterns in the data that are difficult for classical machine learning algorithms to detect while being complimentary to classical machine learning algorithms. We also defined a mixed quantum-classical ensemble method that can help businesses strike a better balance between false positives and false negatives and improve the KPI of the final model. The results presented are obtained on a simulated quantum computer, and the extension of this work to real hardware implementation will be collected in a different manuscript.

ACKNOWLEDGMENT

The authors would like to acknowledge the use of IBM Quantum cloud platform for this work.

REFERENCES

[1] L. Ryll et al., "Transforming paradigms: A global AI in financial services survey," Cambridge Centre Altern. Finance, Univ. Cambridge, Cambridge, U.K., Rep., 2020, doi: 10.2139/ssrn.3532038.

[2] I. Vorobyev and A. Krivitskaya, "Reducing false positives in bank anti-fraud systems based on rule induction in distributed tree-based models," *Comput. Secur.*, vol. 120, 2022, Art. no. 102786, doi: 10.1016/j.cose.2022.102786.

[3] P. Yeoh, "Artificial intelligence: Accelerator or panacea for financial crime?," *J. Financial Crime*, vol. 26, pp. 634–646, 2019, doi: 10.1108/JFC-08-2018-0077.

[4] N. Liu and P. Rebentrost, "Quantum machine learning for quantum anomaly detection," *Phys. Rev. A*, vol. 97, 2018, Art. no. 042315, doi: 10.1103/PhysRevA.97.042315.

[5] D. Herr, B. Obert, and M. Rosenkranz, "Anomaly detection with variational quantum generative adversarial networks," *Quantum Sci. Technol.*, vol. 6, no. 4, Jul. 2021, Art. no. 045004, doi: 10.1088/2058-9565/ac0d4d.

[6] N. Abdelgaber and C. Nikolopoulos, "Overview on quantum computing and its applications in artificial intelligence," in *Proc. IEEE 3rd Int. Conf. Artif. Intell. Knowl. Eng.*, 2020, pp. 198–199, doi: 10.1109/AIKE48582.2020.00038.

[7] M. Schuld and F. Petruccione, *Supervised Learning With Quantum Computers*, vol. 17. Berlin, Germany: Springer, 2018, doi: 10.1007/978-3-319-96424-9.

[8] V. Havlíček et al., "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, no. 7747, pp. 209–212, 2019, doi: 10.1038/s41586-019-0980-2.

[9] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for Big Data classification," *Phys. Rev. Lett.*, vol. 113, no. 13, 2014, Art. no. 130503, doi: 10.1103/PhysRevLett.113.130503.

[10] H. Yano, Y. Suzuki, R. Raymond, and N. Yamamoto, "Efficient discrete feature encoding for variational quantum classifier," in *Proc. IEEE Int. Conf. Quantum Comput. Eng.*, 2020, pp. 11–21, doi: 10.1109/QCE49297.2020.00012.

[11] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, "The power of quantum neural networks," *Nat. Comput. Sci.*, vol. 1, no. 6, pp. 403–409, 2021, doi: 10.1038/s43588-021-00084-1.

[12] K. Batra et al., "Quantum machine learning algorithms for drug discovery applications," *J. Chem. Inf. Model.*, vol. 61, no. 6, pp. 2641–2647, 2021, doi: 10.1021/acs.jcim.1c00166.

[13] I. Kerenidis and A. Luongo, "Classification of the MNIST data set with quantum slow feature analysis," *Phys. Rev. A*, vol. 101, no. 6, 2020, Art. no. 062327, doi: 10.1103/PhysRevA.101.062327.

[14] C. Ciliberto et al., "Quantum machine learning: A classical perspective," *Proc. Roy. Soc. A: Math., Phys. Eng. Sci.*, vol. 474, no. 2209, 2018, Art. no. 20170551, doi: 10.1098/rspa.2017.0551.

[15] S. A. Stein et al., "A hybrid system for learning classical data in quantum states," in *Proc. IEEE Int. Perform., Comput., Commun. Conf.*, 2021, pp. 1–7, doi: 10.1109/IPCCC51483.2021.9679430.

[16] E. A. Lopez-Rojas and S. Axelsson, "Money laundering detection using synthetic data," in *Proc. Annu. Workshop Swedish Artif. Intell. Soc.*, 2012, pp. 33–40.

[17] M. Abufadda and K. Mansour, "A survey of synthetic data generation for machine learning," in *Proc. 22nd Int. Arab Conf. Inf. Technol.*, 2021, pp. 1–7, doi: 10.1109/ACIT53391.2021.9677302.

[18] R. Orús, S. Mugel, and E. Lizaso, "Quantum computing for finance: Overview and prospects," *Rev. Phys.*, vol. 4, 2019, Art. no. 100028, doi: 10.1016/j.revip.2019.100028.

[19] A. El Bouchti, Y. Tribis, T. Nahhal, and C. Okar, "Forecasting financial risk using quantum neural networks," in *Proc. 13th Int. Conf. Digit. Inf. Manage.*, 2018, pp. 386–390, doi: 10.1109/ICDIM.2018.8847063.

[20] J. R. Glick et al., "Covariant quantum kernels for data with group structure," 2021, *arXiv:2105.03406*, doi: 10.48550/arXiv.2105.03406.

[21] Qiskit: An open-source framework for quantum computing, 2021.

[22] A. D. Córcoles et al., "Challenges and opportunities of near-term quantum computing systems," *Proc. IEEE*, vol. 108, no. 8, pp. 1338–1352, Aug. 2020, doi: 10.1109/JPROC.2019.2954005.

[23] C. J. Wood, "Special session: Noise characterization and error mitigation in near-term quantum computers," in *Proc. IEEE 38th Int. Conf. Comput. Des.*, 2020, pp. 13–16, doi: 10.1109/ICCD50377.2020.00016.

[24] K. Georgopoulos, C. Emary, and P. Zuliani, "Modeling and simulating the noisy behavior of near-term quantum computers," *Phys. Rev. A*, vol. 104, no. 6, 2021, Art. no. 062432, doi: 10.1103/PhysRevA.104.062432.

[25] G. Li, Y. Ding, and Y. Xie, "Tackling the Qubit mapping problem for NISQ-era quantum devices," in *Proc. Int. Conf. Architectural Support Program. Lang. Operating Syst.*, 2019, pp. 1001–1014, doi: 10.1145/3297858.3304023.

[26] Y. Liu, S. Arunachalam, and K. Temme, "A rigorous and robust quantum speed-up in supervised machine learning," *Nat. Phys.*, vol. 17, pp. 1013–1017, 2021, doi: 10.1038/s41567-021-01287-z.

[27] M. Schuld and F. Petruccione, "Quantum ensembles of quantum classifiers," *Sci. Rep.*, vol. 8, 2018, Art. no. 2772, doi: 10.1038/s41598-018-20403-3.

[28] A. Macaluso, L. Clissa, S. Lodi, and C. Sartori, "Quantum ensemble for classification," 2020, *arXiv:2007.01028*, doi: 10.48550/arXiv.2007.01028.