Date of publication xxxx 00, 0000, date of current version 2022 08.

Digital Object Identifier 10.1109/TQE.2020.DOI

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[6], AND ANDREAS RUDNIK[7].

[1]European Organization for Nuclear Research (CERN), Geneva 1211, Switzerland (email: michele.grossi@cern.ch)
[2]IBM Quantum, IBM 3600 Steeles Ave East Markham, ON L3R 9Z7, CA (email: noel.ibrahim@ibm.com)
[3]IBM Quantum, IBM Deutschland Research & Development GmbH, Schönaicher Str. 220, 71032 Böblingen, Germany (email: voica.radescu@ibm.com)
[4]IBM Quantum, IBM Corp, 1 Alhambra Plaza Suite #1415 Coral Gables, FL 33134 (email: loredo@us.ibm.com)
[5]IRIS Analytics GmbH, Klostergut Besselich, 56182 Urbar, Germany (email: kirsten.voigt@iris.de)
[6]IRIS Analytics GmbH, Klostergut Besselich, 56182 Urbar, Germany (email: constantin.von.altrock@iris.de)
[7]IRIS Analytics GmbH, Klostergut Besselich, 56182 Urbar, Germany (email: andreas.rudnik@iris.de)

M. Grossi, N. Ibrahim, V. Radescu are primary contributors. List of authors N Ibrahim, V. Radescu, M. Grossi, K. Voigt, and C. 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. We declare that there are no competing interests.

ABSTRACT This paper 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 Quantum Support Vector Machine's feature map characteristics. The results are compared using fraud specific key performance indicators: Accuracy, Recall, and False Positive Rate, extracted from analyses based on human expertise (rule decisions), classical machine learning algorithms (Random Forest, 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 which led to an improved accuracy of the mixed quantum-classical method for fraud detection, on a drastically reduced data set to fit current state of Quantum Hardware.

INDEX TERMS Fraud Detection, Quantum, Feature Selection, QSVM, Quantum Kernel Alignment

I. INTRODUCTION

Over the past few years, the financial industry has seen a substantial growth in innovation, particularly in the field of 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: more data from more sources is 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 is an active field of research which 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 to train models [4], [5] and possibly enhance machine learning models such as quantum support vector machines (QSVM) [6], [7], quantum classifiers (QC) [8], and quantum neural networks (QNN) [9]. Much work has been conducted on synthetic and publicly available datasets from various domains such as drug discovery [10], image classification [11], and computational sciences [12]. Comparisons have been made to the classical counterparts of the available quantum machine learning algorithms [13]. In addition, when synthetic data is used for machine learning experiments, there have been provable advantages shown involving synthetic data sets when there is a lack of necessary data [14], [15].

In this work we investigate the impact of quantum feature selection techniques versus classical feature selection techniques on the performance of the quantum machine learning classifier. We consider that prefacing the classical feature selection to the application of a quantum machine learning may eliminate some or all of the complex nuances in the relationships between features and outcomes that quantum machine learning methods are thought to be able to detect. Finally, we compare the performance of Quantum Support Vector Machines to state of the art methods in fraud detection such as Random Forest and XGBoost, using a "real-world" data set 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 are being analyzed in this paper using same initial data set:

1) Domain expert created decision rules-based model (no machine learning)
2) State-of-the-art type AI/ML using boosted trees (Random Forest, XGBoost)
3) Quantum Support Vector Machine (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 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 paper.

B. PAYMENT FRAUD PREVENTION KPIS

Payment fraud prevention relies on two specific KPIs: (monetary) *it rate* and *false alarm ratio*. The hit rate, typically reported as a percentage, is the total amount of all transactions that were declined by the fraud prevention system that later were confirmed to have been fraudulent, divided by the total amount of transactions that were later confirmed to have been fraudulent. The false alarm ratio is typically given as the ratio of false alerts to true alerts. Thus, if the model created 10 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 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 vs non-fraud). A statistical measure of a model is given by *Accuracy*, the number of classifications a model correctly predicts divided by the total number of predictions made. The accuracy KPI is meaningful only for a balanced class data set. A better diagnostic of a binary classifier performance is through a Receiver Operating Characteristic curve, or 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 financial KPIs, so that the x axis is the false alarm ratio, 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 either decline or not to decline a transaction, usually a threshold is applied to the score to make this decision.

II. INPUT DATA SET

We are using a data set of real-world payment transactions that comes from a European cross-border processing portfolio and consists of about 80% debit and 20% credit card transactions. This data set contains a total of 2.4 million payment transactions. Each transaction is flagged as fraud or non-fraud, with a total of $\tilde{3}$k transactions marked as fraudulent in the data set. Importantly, the transaction data can be enriched with customer reference data and with features built on the fly, as described in the following subsections.

A. TRANSACTION AND CUSTOMER DATA

The data set we work with has only 12 input attributes from transaction data, with additional 2 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: 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 (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 pre-defined 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 non-fraudulent transaction.

In total, before invoking any pre-processing of data, we have 48 attributes. These attributes are of various data types: categorical, string, integers, etc. Handling categorical data type is posing some challenges for machine learning classifiers, and it will require treatment such 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 data set in terms of fraudulent vs genuine records, whereas machine learning methods (classical or quantum) require balancing the set via under-sampling methods. Moreover, when using Quantum Machine Learning (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 three distinct data references that require different levels of pre-processing for each of the analyzed use cases:

1) Full dimensionality of a data set: only cleaned from redundant data and split into train and test. This data set can be used by Rule Generator assisted by a fraud subject matter expert, even if it's highly imbalanced by the number of genuine records relative to the fraudulent ones.
2) Balanced data set, with genuine transactions randomly under-sampled (there are various methods to achieve this), together with treatment to handle the categorical data type. This data set then can be optimally used by Classical Machine Learning.
3) Drastically reduced data samples which will require multiple trials to avoid bias due to heavy under-sampling. 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 either 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 criteria 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 data set 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 under-sampling 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 has been split chronologically as follows: the first 1.5million records are used for training and the remaining records for testing. The count of records in Fraud and Genuine is shown 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 Figure 1, often referred to as a modified Receiver Operating Characteristic (ROC) curve. The modified ROC* curve is the primary metric to understand the performance of a model.

The KPIs of the model can be as good as 65% hit rate but with the penalty of 25 non-fraudulent payments intercepted for each one fraudulent payment intercepted, or with 30% hit rate with only 5 non-fraudulent payments intercepted for each one fraudulent payment intercepted. This is a trade-off decision to be taken by either a processor or a bank. The train curve sits considerably above the test curve which is an indication that Rule Generator method is prone to over fit.

B. CLASSICAL MACHINE LEARNING CLASSIFIERS

Payment Fraud Detection is commonly using supervised machine learning classifiers, where historical data has fraud

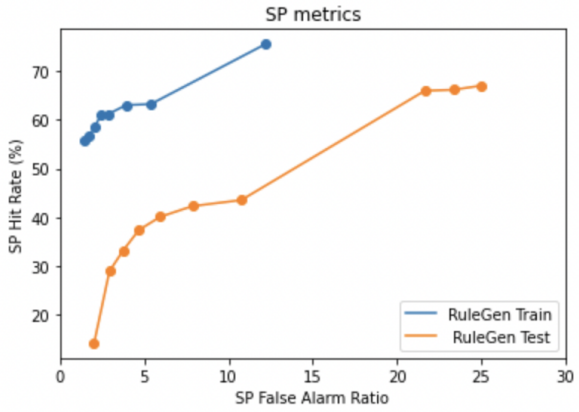

FIGURE 1: Results based on Rule generator using a complete data set that is split into training (blue) and verification (orange). The metrics are Hit Rate and False Alarm Ratio. This representation of the KPIs is often referred to as a modified ROC* curve, with the False Alarm Ratio on the x axis (ratio False Positives True Positives.

marks either detected by a case investigator or reported by an affected customer and has ideally been caught by the fraud prevention models before it happened. Examples of Supervised Learning: Regression, Decision Tree, Random Forest, SVM, Logistic Regression etc. For this analysis, we've 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: balance the data set and convert data types to numeric values. The following pre-processing steps have been applied to data before passing it to a classical classifier, apart from the under-sampling:

1) Removal of highly correlated features (duplication of information)
2) 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.
3) Split of the data into "training" and "test" sets for use when training the models.
4) Treatment for imbalanced data, where there is much more genuine than fraudulent transactions: achieved by under-sample genuine by larger fraction than fraud with the aim to preserve all fraud marks. Finding the right balance is an art, we ran 5 random trials.

We first started with a complete data set which has been split into test and training, as in Table 1 and then, on reduced data set that has been used for the Quantum Machine Learning part as well.

Set	Label	Count	Train	Test
Original	0	2396689	1498434	898255
	1	3216	1566	1650
Balanced	0	1505	984	521
	1	993	515	478
Reduced	0	366	262	104
	1	232	137	95

TABLE 1: Original Data Set Count, Balanced Data Set Count and Drastically Reduced Data set count, overall and split into test and train.

2) Classical Machine Learning with Original Data Sets

We used *XGBoost CV* package for tuning the model's hyper parameters to find the best number of estimators, max_depth, min_child, in a non-exhaustive 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 3-fold cross-validation, and searched across 10 different combinations. The results of these two fits in terms of Accuracy and AUC performance parameters are shown in Table 2.

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 2: Accuracy and AUC results for XGBoost and Random Forest using Original Data, without under sampling

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 Quantum Machine Learning 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 vs

XGBoost	Random Forest
F_10	F_16
F_16	F_0
F_15	F_15
F_0	F_14
F_7	F_10
F_3	F_19

TABLE 3: Ordered feature Importance for XGBoost and Random Forest.

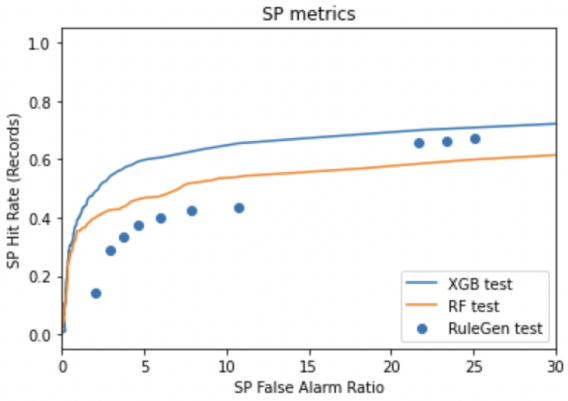

FIGURE 2: Results based on XGBoost, Random Forest and Rule Generator using a complete data set without categorical attributes that is split into training and verification. The metrics are Hit Rate and False Alarm Ratio.

False Alarms, as defined in the Safer Payments product. The figure 2 overlays the modified ROC* curves from different machine learning models versus Rule Generator.

Figure 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 you 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 data set. 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 Data Sets

For this part, to balance the set, we massively under-sampled data by $1:1000$ for the genuine transactions, preserving a third of the fraudulent records. The train set has 1500 records, while the test 1000, as shown in Table 1. To minimize biasing effects from the strong under-sampling, we have run the split in 5 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: accuracy, AUC, and the dependency between hit rate vs false alarm rates for the test sample, as seen in the modified ROC* curves Fig 3. Notice that values on the x axis are affected by the under-sampling and should be scaled by the under-sampling factor of 500 to be comparable with the ROC* curves from the original data set. 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,

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

TABLE 4: Accuracy and AUC results for XGBoost and Random Forest using Balanced Data Set, with under-sampling, therefore we took 5 trials.

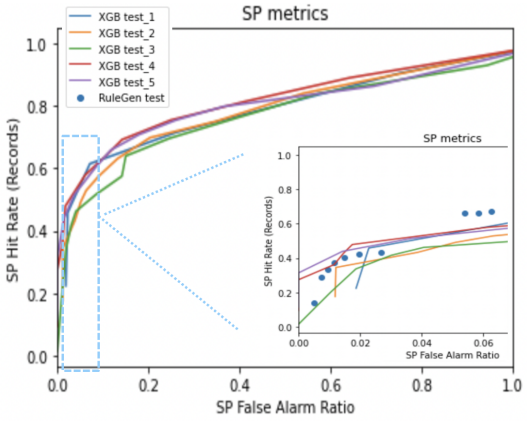

FIGURE 3: Results based on XGBoost multiple trials. The X axis is false alarms, however the range is equivalent to a factor of 500 more, due to the under-sampling. The snippet is a zoomed region of the plot that would be equivalent to real fraud alarm range of $(0, 100)$ and it is directly compared with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

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 under sampling procedure, having in view that we can only use a significantly reduced data sets with the current state of quantum hardware.

IV. QUANTUM MACHINE LEARNING FOR FRAUD DETECTION

There are various quantum machine learning approaches for classification problems [6], [13], [16], [17]. In this paper we are focused primarily on the Quantum Support Vector Machine (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 like 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 such the one introduced in [18], 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

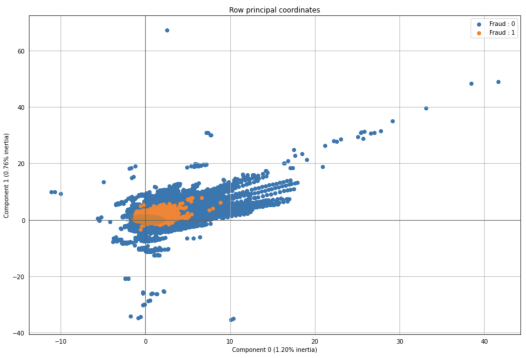

FIGURE 4: The relation between the components from the FAMD method. A total overlap is observed which limits the FAMD approach in using it for data feature reduction.

approach in two parts in order to optimize the fraud detection system. The first is to determine which of the many features available should be selected to reduce the dimensionality of the data set for running the experiment on a quantum system. The second is to derive the fraud KPIs from a quantum machine learning model. We used Qiskit [19] 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 [6] 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 prerequisite. 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 data set of 2.4 mil records.

The data used for payment fraud prevention is 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 Factorial Analysis of Mixed Data (FAMD) method, which works for a mix of categorical and numerical variables. However, for this data set 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. At the time of writing, there is no inbuilt feature importance method for QSVM, while when using features importance extracted by classical classifiers bias the outcome and 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 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 $0, 1$ are 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 3 qubits is represented in Fig.10. 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 data sets prepared, we proceeded by setting up the *QuantumKernel* class to calculate a kernel matrix using the *ZZFeatureMap*.

Firstly, the application of QSVM method has been tested and ran on a quantum simulator, under ideal condition, namely with a $state\_vector$ simulator, 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 Feed Forward Feature Selection (FFFS) 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.5 to approach the problem of fraud detection.

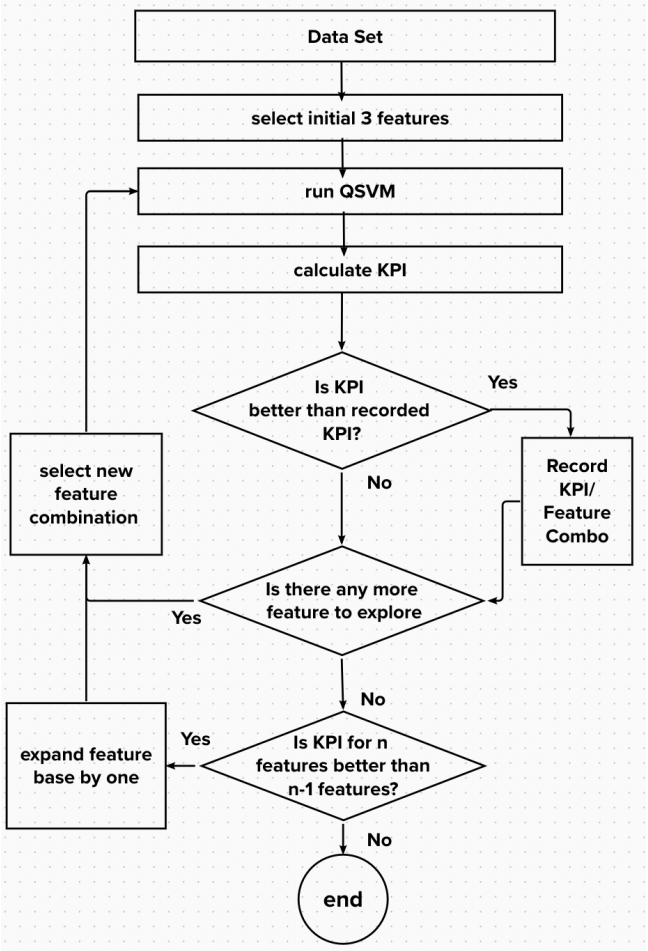

FIGURE 5: Schematic representation of the quantum algorithm feature selection to exploit the usage of quantum computing in the context of fraud detection.

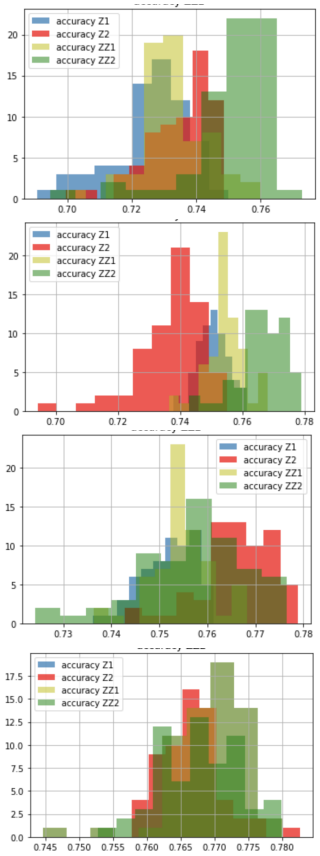

FIGURE 6: Figure shows the spread in the Accuracy Values when each feature combination is explored for best 3 features, best 4 features, best 5 features, best 6 features, respectively. This is done for each of the quantum feature map choices of Z map depth 1 and 2, as well as ZZ map depth 1 and 2.

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 quantum classifier is defined, trained, tested and the accuracy and the AUC is stored. At the end of the procedure, the best model based on the accuracy as key performance indicator (for future work, one could consider different KPI as the discriminator) and therefore the best 3 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 trade-off between the maximum number of available qubit and the total accuracy obtained in the iteration.

Figure 6 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 which 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: 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, "n_shots": 8192.

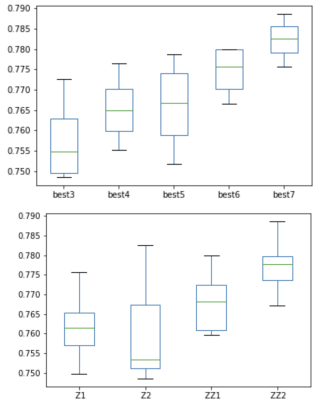

FIGURE 7: Figure a) shows the improvement of the accuracy as we add more features, while Figure b) shows the preference towards the entangled feature map for better accuracy.

B. QSVM RESULTS

The search of best features is performed using Z and ZZ Quantum Feature Maps with depth 1 and depth 2. Figure 7 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 performs 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 (Table 5) are different from the best features selected by the classical algorithms using same data sets, emphasizing a crucial role that feature exploration with QSVM plays to complement the feature space scanning.

QSVM 6 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_9	+F_55	+F_31	+F_3
Acc= 0.776	Acc=0.768	Acc =0.761	Acc=0.755
+F_36	+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

TABLE 5: The 7 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.

	F_15	F_42	F_65	F_2	F_38	F_8	F_64
F_15	1.000000	-0.027727	0.171003	-0.002909	0.027083	0.188673	0.069265
F_42	-0.027727	1.000000	-0.003665	-0.040925	0.065273	0.047009	-0.010402
F_65	0.171003	-0.003665	1.000000	-0.003620	-0.029180	0.092146	-0.024211
F_2	-0.002909	-0.040925	-0.003620	1.000000	-0.015200	-0.009716	-0.012612
F_38	0.027083	0.065273	-0.029180	-0.015200	1.000000	-0.012366	-0.101668
F_8	0.188673	0.047009	0.092146	-0.009716	-0.012366	1.000000	-0.048381
F_64	0.069265	-0.010402	-0.024211	-0.012612	-0.101668	-0.048381	1.000000

FIGURE 8: Correlation among best features selected by the QSVM method.

Interesting observation is that the new algorithm has indeed identified features with least level of overlap, as shown in the correlation matrix in Table 8. This demonstrates that the choices are indeed viable. A reminder that most of the features have been engineered from same initial set of raw 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 data sets with 1500 records used for the training and 1000 for testing. Due to data under-sampling, we have used 5 random trials to minimize bias. The average KPIs for accuracy and ACU are reported in the Table 6.

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 6: Accuracy and AUC results for QSVM using XGBoost and Random Forest best feature selection vs QSVM best feature selections, based on Balanced Data Set and using 6 trials.

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 6 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. 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 [20], [21]. Since the current state-of-the-art for these systems are still considered noisy, we wanted to run our simulation tests with noise models which are based off real quantum systems [22]. We have used the noise source analyzed for the ibm_cairo backend and re-run the QSVM and enabled the readout measurement error mitigation flag. When using noisy simulation, the optimal circuit transpilation pipeline in Qiskit

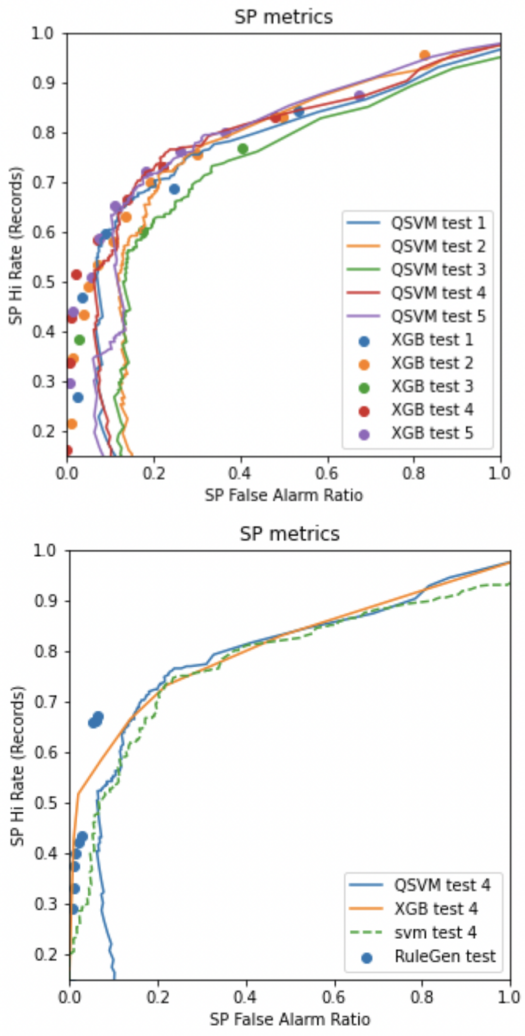

FIGURE 9: Figure shows the modified ROC* Curve for QSVM using state vector simulator.

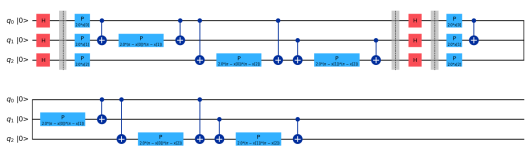

FIGURE 10: Example of a QSVM transpiled job circuit for 3 features.

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 [23], and performs the most 1Q and 2Q gate optimizations.

The circuit depth produced by the 7 best features is around 70, an illustration of a circuit depth diagram produced by 3 features is provided in Fig. 10. Reducing the depth of the circuit and optimize it for use on the hardware is a study for future work that is in plans.

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

TABLE 7: KPIs for test samples when running QSVM on different backends: state vector simulator, qasm simulator with and without noise, IBM quantum systems with and wirhout m.e.m. enabled.

V. MIXED QUANTUM-CLASSICAL METHOD FOR FRAUD DETECTION

In this paper we are exploring the complementarity to classical ML provided by QSVM.

Although quantum computing has been proven to speed up some types of problems [24], the existent technology allows only a limited number of qubits and gate operations. Therefore, we employ a hybrid classical / quantum solution where data and function learning are classical, while the classification algorithm is quantum. In this paper we use Quantum Support Vector Machine, where the quantum computer is only used once to compute the kernel matrix for the training set. The optimal separating hyperplane and the support vectors can be obtained efficiently in the training set size by solving the conventional dual optimization problem on a classical computer.

A. APPROACH

In order to exploit the complementarity of the quantum and classical algorithms to increase classification performance, we wish to discern those transactions or data points for which the two algorithms disagree in the classification. When the two algorithms disagree, we wish to predict which is correct. In order to accomplish this, we trained both the quantum and classical algorithms on the train sub-portion of the balanced dataset. When the two algorithms 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 and may take on a number of forms. In practice, because of the size of dataset, the number of training datapoints on which the classifiers disagreed was limited so a simple meta-classifier performed best. In the case of the fully optimized XGBoost a classical SVM was used as the metaclassifier. In the case of the XGBoost without optimization, a logistic regression was used.

B. RESULTS AND COMPARISONS

While the performance of the classical and quantum algorithms was similar, the actual predictions can vary for specific datapoints, yielding complementary results (see Figure 11). Classifications disagree on 5.2% of Training Data, and 5.5% of Test Data, with threshold of 0.5. On the diagonal the quantum and classical models agree. On the off-diagonal they

Model	Acc.	CI	Std	N_{trials}	$N_{best\ perf.\ features}$
XGBoost	77.7%	0.3%	0.5%	10	9
XGBoost - opt.	80.0%	0.5%	0.9 %	10	37
QSVM - ZZ	78.8%	0.4%	0.7%	10	7

TABLE 8: Comparison of Model Accuracy on Balanced Dataset - Separate Models

Model	Acc.	CI	Std	N_trials
QSVM + XGBoost	79.9%	0.4%	0.7%	10
QSVM + XGBoost- opt.	81.0	0.3%	0.5%	10

TABLE 9: Comparison of Model Accuracy on Balanced Dataset - Mixed Models

disagree. This shows that different relationships are detected by the quantum and classical models- complementarity. 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.

Using IBM Qiskit simulator, we have successfully employed Quantum Support Vector Machine method and we have measured the accuracy and AUC for different use cases.

VI. SUMMARY

Classical Machine Learning algorithms are currently state of art for predicting fraud in transactions. Quantum machine learning can provide a complementary support on this, exploiting enhanced feature space to encode historical data. In this work we propose a novel approach to maximize a quantum classifier performance in terms of accuracy of prediction, but other KPI can be explored as well. The method is called Quantum Feature Importance Selection algorithm: using Quantum enhanced Support Vector Machine we are able to select most relevant features for the quantum classifier 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 pre-processing from data type considerations to undersampling techniques before moving to the quantum part. We found that quantum classifiers 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 define 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 result 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.

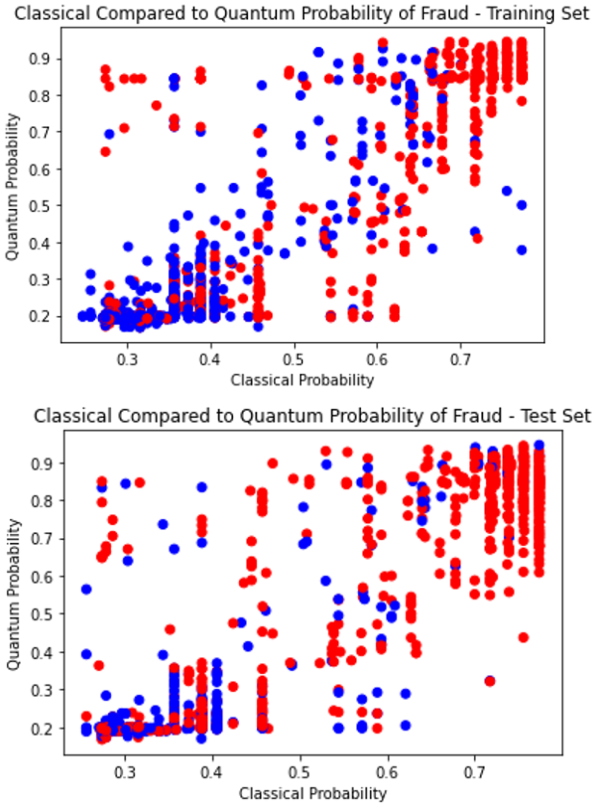

FIGURE 11: Figure shows complementarity of the Classical and Quantum decisions. Red Dots are fraudulent transactions, Blue dots are genuine transactions. Although similar Performance, different label Assignments. The position on the x axis represents the probability of fraud predicted by the classical algorithm; dots with high probability are predicted to be fraud by the classical algorithm. The position on the y axis represents the probability of fraud predicted by the quantum algorithm; dots with high probability are predicted to be fraud by the quantum algorithm.

ACKNOWLEDGMENT

We acknowledge the use of IBM Quantum cloud platform for this work. 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.G. is supported by CERN Quantum Technology Initiative.

REFERENCES

[1] L. Ryll, M. E. Barton, B. Z. Zhang, R. J. McWaters, E. Schizas, R. Hao, K. Bear, M. Preziuso, E. Seger, R. Wardrop et al., "Transforming paradigms: A global ai in financial services survey," 2020.

[2] I. Vorobyev and A. Krivitskaya, "Reducing false positives in bank anti-fraud systems based on rule induction in distributed tree-based models," Computers & Security, vol. 120, p. 102786, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016740482200181X

[3] P. Yeoh, "Artificial intelligence: accelerator or panacea for financial crime?" Journal of Financial Crime, 2019.

[4] N. Abdelgaber and C. Nikolopoulos, "Overview on quantum computing and its applications in artificial intelligence," in 2020 IEEE Third International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). IEEE, 2020, pp. 198–199.

[5] M. Schuld and F. Petruccione, Supervised learning with quantum computers. Springer, 2018, vol. 17.

[6] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, "Supervised learning with quantum-enhanced feature spaces," Nature, vol. 567, no. 7747, pp. 209–212, 2019.

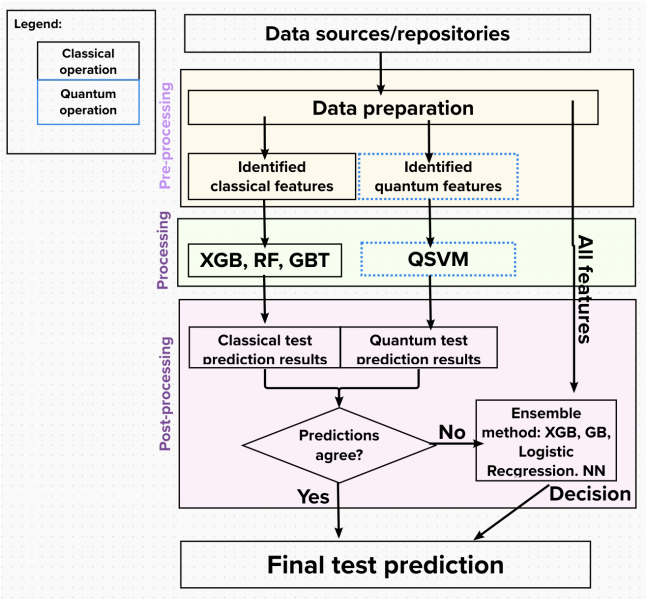

FIGURE 12: The Flow chart of how to combine the Classical and Quantum Approached into a single decision. When the quantum and classical algorithms disagree – a Metaclassifer predicts which one is correct.

[7] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," Physical review letters, vol. 113, no. 13, p. 130503, 2014.

[8] H. Yano, Y. Suzuki, R. Raymond, and N. Yamamoto, "Efficient discrete feature encoding for variational quantum classifier," in 2020 IEEE International Conference on Quantum Computing and Engineering (QCE). IEEE, 2020, pp. 11–21.

[9] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, "The power of quantum neural networks," Nature Computational Science, vol. 1, no. 6, pp. 403–409, 2021.

[10] K. Batra, K. M. Zorn, D. H. Foil, E. Minerali, V. O. Gawriljuk, T. R. Lane, and S. Ekins, "Quantum machine learning algorithms for drug discovery applications," Journal of chemical information and modeling, vol. 61, no. 6, pp. 2641–2647, 2021.

[11] I. Kerenidis and A. Luongo, "Classification of the MNIST data set with quantum slow feature analysis," Physical Review A, vol. 101, no. 6, jun 2020.

[12] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, "Quantum machine learning: a classical perspective," Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 474, no. 2209, p. 20170551, 2018.

[13] S. A. Stein, R. L'Abbate, W. Mu, Y. Liu, B. Baheri, Y. Mao, G. Qiang, A. Li, and B. Fang, "A hybrid system for learning classical data in quantum states," in 2021 IEEE International Performance, Computing, and Communications Conference (IPCCC). IEEE, 2021, pp. 1–7.

[14] E. A. Lopez-Rojas and S. Axelsson, "Money laundering detection using synthetic data," in Annual workshop of the Swedish Artificial Intelligence Society (SAIS). Linköping University Electronic Press, Linköpings universitet, 2012.

[15] M. Abufadda and K. Mansour, "A survey of synthetic data generation for machine learning," in 2021 22nd International Arab Conference on Information Technology (ACIT). IEEE, 2021, pp. 1–7.

[16] R. Orús, S. Mugel, and E. Lizaso, "Quantum computing for finance: Overview and prospects," Reviews in Physics, vol. 4, p. 100028, 2019.

[17] A. El Bouchti, Y. Tribis, T. Nahhal, and C. Okar, "Forecasting financial risk using quantum neural networks," in 2018 Thirteenth International Conference on Digital Information Management (ICDIM). IEEE, 2018, pp. 386–390.

[18] J. R. Glick, T. P. Gujarati, A. D. Corcoles, Y. Kim, A. Kandala, J. M. Gambetta, and K. Temme, "Covariant quantum kernels for data with group structure," 2021.

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

[20] A. D. Córcoles, A. Kandala, A. Javadi-Abhari, D. T. McClure, A. W. Cross, K. Temme, P. D. Nation, M. Steffen, and J. M. Gambetta, "Challenges and opportunities of near-term quantum computing systems," arXiv preprint arXiv:1910.02894, 2019.

[21] C. J. Wood, "Special session: Noise characterization and error mitigation in near-term quantum computers," in 2020 IEEE 38th International Conference on Computer Design (ICCD). IEEE, 2020, pp. 13–16.

[22] K. Georgopoulos, C. Emary, and P. Zuliani, "Modeling and simulating the noisy behavior of near-term quantum computers," Physical Review A, vol. 104, no. 6, p. 062432, 2021.

[23] G. Li, Y. Ding, and Y. Xie, "Tackling the qubit mapping problem for nisq-era quantum devices," CoRR, vol. abs/1809.02573, 2018. [Online]. Available: http://arxiv.org/abs/1809.02573

[24] K. T. Yunchao Liu, Srinivasan Arunachalam, "A rigorous and robust quantum speed-up in supervised machine learning," Nat., Phys., vol. 17, p. 1013–1017, 2021.