Quantum-inspired algorithm for direct multi-class classification

Roberto Giuntini[1], Federico Holik[2], Daniel K. Park[3,4,*], Hector Freytes[1], Carsten Blank[5], and Giuseppe Sergioli[1]

[1]University of Cagliari, Cagliari, Italy
[2]Instituto de Fisica de La Plata - Conicet, Buenos Aires, Argentine
[3]Department of Applied Statistics, Yonsei University, South Korea
[4]Department of Statistics and Data Science, Yonsei University, South Korea
[5]Data Cybernetics, Landsberg am Lech, Germany
[*]Corresponding author; email: *dkd.park@yonsei.ac.kr*

ABSTRACT

Over the last few decades, quantum machine learning has emerged as a groundbreaking discipline. Exploiting the peculiarities of quantum computation for machine learning tasks offers promising advantages. Quantum-inspired machine learning, which is a branch of quantum machine learning, has revealed how relevant benefits for machine learning problems can be obtained using quantum-inspired features even without employing quantum computers. In the recent past, experiments have demonstrated how to design an algorithm for binary classification inspired by the method of quantum state discrimination, which exhibits high performance with respect to several standard classifiers. However, a generalization of this quantum-inspired binary classifier to a multi-class scenario proved to be a nontrivial task. Typically, a simple solution in machine learning decomposes multi-class classification into a combinatorial number of binary classifications, with a concomitant increase in computational resources. In this study, we introduce a quantum-inspired classifier that avoids this problem. Inspired by quantum state discrimination, our classifier performs multi-class classification directly without using binary classifiers. We first compared the performance of the quantum-inspired multi-class classifier with eleven standard classifiers. The comparison revealed a surprisingly good performance of the quantum-inspired classifier. Comparing these results with those obtained using the decomposition in binary classifiers shows that our method improves the accuracy and reduces the time complexity. Therefore, the quantum-inspired machine learning algorithm proposed in this work serves as an effective and efficient framework for multi-class classification. Finally, although these advantages can be attained without employing any quantum component in the hardware, we discuss how it is possible to implement the model in quantum hardware.

Introduction

Quantum technologies offer exciting opportunities for improving various data analysis solutions, such as pattern recognition and machine learning. The application of quantum computing methods to such problems has opened a novel research field, known as quantum machine learning. The results obtained thus far are promising, thereby illustrating the potential advantages in solving several problems that are challenging for computers built with purely classical components[1–10]. However, building the fault-tolerant quantum hardware required for the reliable implementation of such quantum algorithms remains a technological hurdle. Nevertheless, merely simulating quantum theory-based algorithms with classical hardware can achieve promising results, because the theory provides new perspectives for information processing tasks. We refer to the algorithms that use the mathematical formalism of quantum theory but are intended to be executed on a classical device as quantum-inspired (QI) algorithms[1]. The immediate availability of classical computers for QI algorithms has introduced several applications in solving industry-relevant problems related to chemistry, optimization, and finance [11–17].

The idea of developing an algorithm based on the mathematical structure of quantum mechanics can also be applied for solving machine-learning problems[18–27]. One of the most important applications of machine learning is supervised classification. Supervised classification trains a model from a labeled dataset so that the model can predict the labels of unseen objects in such a manner that the agreement between the new labelling and the existing pattern of the training dataset is maximized. Quantum measurement theory is an intriguing characteristic of quantum theory that can have significant impact on classification. In brief, according to theory, quantum measurement can discriminate not only orthogonal states, such as 0 and 1 in the classical case, but also non-orthogonal states with a non-zero success probability[28] through positive operator-valued measurement (POVM)[29,30].

[1] In contrast, quantum algorithms are intended to be executed on quantum devices.

This implies that quantum theory provides a richer data representation because a single qubit can represent an arbitrary number of data points, whereas a classical bit can only represent two. This is a direct consequence of the geometry of the quantum states, which is mathematically defined in Hilbert space. Alternatively, by choosing Hilbert space as the feature space of classical data, the mathematical structure of the quantum measurement theory can be exploited. The probability of successfully discriminating different data mapped to a Hilbert space can be maximized by determining the optimal measurement using the well-developed theory of quantum state discrimination (QSD)[31,32]. In the case of two quantum states (or two-element POVM), the optimal measurement can be derived analytically, and it is known as Helstrom measurement[28]. A quantum-inspired binary classifier based on the Helstrom measurement, called the Helstrom quantum classifier (HQC), was introduced in Refs. [20,33]. Previous studies compared HQC with other standard binary classifiers on a classical computer and demonstrated that HQC achieved good classification accuracy on average. Furthermore, it was also empirically shown that the accuracy can be improved by using multiple copies of the quantum states to encode classical data, although a theoretical proof was not provided. One of the main drawbacks of the HQC is that it is restricted to binary problems. Consequently, for general multi-class classification problems, the One-vs-One (OvO) or One-vs-Rest (OvR) comparison strategies must be utilized. However, these approaches increase the computational complexity of the entire classification process because they require binary classification to be repeated $l(l-1)/2$ and l times, respectively, for an l-class dataset. In contrast, classical machine learning algorithms can intrinsically perform multi-class classification. For example, this can be done by evaluating the classification score associated with each class as a probability and assigning the test data to a class with the highest probability (i.e., argmax). Consequently, the generalization of previous QI binary models to intrinsically multi-class classifiers must be addressed.

In this study, we propose a QI algorithm that directly performs l-class classification using l-element POVM that is optimized based on the QSD theory. We performed numerical experiments to compare our QI algorithm with other standard classical machine learning algorithms, both parametric and non-parametric, with 11 benchmarking datasets. The experiments confirmed our algorithm's excellent performance [2]. Furthermore, our algorithm was compared with the OvO and OvR strategies constructed with HQC regarding the classification accuracy and time complexity. The results reveal that, for all datasets, our algorithm achieved a higher classification accuracy than the HQC-based approaches. Therefore, our algorithm improves both, the time complexity and classification accuracy of the previous QI classifier. Similar to the observations regarding the HQC classifier in previous studies, we provide empirical evidence that using multiple copies of a quantum state that encodes classical information can improve the classification accuracy of our QI multi-class classifier at the cost of increased runtimes.

Our findings pave a way for the near-term application of quantum technology because they demonstrate the computational advantage of QI algorithms even in the absence of quantum computing hardware. Finally, considering the era of full-fledged quantum computing, we outline the implementation of our algorithm on a quantum circuit. The fact that our algorithm uses the formalism of quantum theory, renders it naturally implementable on actual quantum hardware. We discuss the advantages of this implementation in terms of both the computational power and memory use.

The remainder of this paper is organized as follows. In the Results section, we first introduce in detail the general concept of a quantum classifier in the standard supervised scenario, focusing on the crucial notion of quantum centroid. Following that, we introduce the concept of a multi-class quantum-inspired classifier. Subsequently, we present the experimental results of the comparison between the multi-class QI classifier with i) 11 standard classifiers and ii) a binary QI classifier, using the OvO and OvR strategies. Finally, in that section, we discuss a possible implementation of the multi-class quantum-inspired classifier on a real near-term quantum computer. The Discussion section provides a detailed analysis of the results obtained, whereas the Methods section formally describes the theoretical framework of the model. Therein, we detail the technicalities of the experimental setup, highlighting the impact of our findings, limitations of the model, and potential future developments. The paper also includes a Supplementary Information section with useful technical remarks that make the paper more self-contained and a complete set of the experimental data.

Results

We first describe the task of multi-class classification and introduce the basic notations used in the definition of our learning problem. A training dataset can be represented by a set of patterns, as follows:

$$\mathscr{S}_{\mathrm{tr}} := \{(\vec{x}_1, \lambda_1), \ldots (\vec{x}_m, \lambda_m)\}.$$

where $\vec{x}_j \in \mathbb{C}^d$ is a d-dimensional feature vector and $\lambda_j \in L := \{0, \ldots, \ell\}\ \forall j \in \{1, \ldots, m\}$ is the corresponding class label. Given a class label $i \in L$, the set $\mathscr{S}^i_{\mathrm{tr}}$ of all object vectors whose associated class label is i can be defined as follows:

$$\mathscr{S}^i_{\mathrm{tr}} = \{\vec{x}_j \in \mathscr{S}_{\mathrm{tr}} : \lambda_j = i\}.$$

[2] The comparison of our distance-based classifier with some of the standard classical machine learning algorithms cannot be termed entirely fair. It should only be compared with other distance-based classifiers. However, analyzing its global performance is interesting; therefore, we decided to include the results here.

The cardinality of $\mathscr{S}_{\text{tr}}^{i}$ is denoted as $|\mathscr{S}_{\text{tr}}^{i}|$. Consequently, $\sum_{i=1}^{\ell}|\mathscr{S}_{\text{tr}}^{i}| = m$. Given the aforementioned training dataset, the task of supervised classification is to infer a function from the training dataset, $\mathscr{S}_{\text{tr}}$, using a classifier to maximize the accuracy of the class label assignment for an unseen (i.e., unlabeled) object vector $\vec{x}$.

More formally, a multi-class classifier can be constructed as follows. First, we find a map that associates with any feature vector $\vec{x}$, a sequence of l-numbers in the unit real-interval $[0,1] \subset \mathbb{R}$

$$f : \mathbb{C}^d \to [0,1]^{\ell}.$$

The i^{th} component of $f(\vec{x})$ is denoted by $f(\vec{x})_i$, which can be considered the classification score for the given data to be assigned a label i. We further constrain the score function, $f(\vec{x})$, to a probability vector by imposing $\sum_{i=1}^{\ell} f(\vec{x})_i = 1$. Hence, $f(\vec{x})_i$ can be interpreted as the probability of feature vector $\vec{x}$ belonging to the i^{th} labeled class.

The classifier by f (or simply the f-classifier) is the map

$$Cl_f : \mathbb{C}^d \mapsto L$$

that assigns any feature vector $\vec{x} \in \mathbb{C}^d$, the class label associated with the highest value of $f(\vec{x})_i$ with $1 \leq i \leq \ell$. In other words,

$$Cl_f(\vec{x}) = \arg\max_i \{f(\vec{x})_i : 1 \leq i \leq \ell\}.$$

It should be noted that there can be multiple solutions for the maximum value of $f(\vec{x})$. Accordingly, our definition of the multi-class classifier is:

$$Cl_f(\vec{x}) := \min \left\{ i \in L : f(\vec{x})_i = \max_k \{f(\vec{x})_k, 1 \leq k \leq \ell\} \right\}. \tag{1}$$

Quantum Centroid

The QI classifier construction defined in Eq. (1) involves the evaluation of $f(\vec{x})$ based on the mathematical formalism of the quantum theory. To utilize the quantum theory, we first need to represent each feature vector $\vec{x}$ as a quantum state, which is mathematically described by the density operator, $\rho_{\vec{x}}$. Various quantum feature mapping techniques for transforming classical information contained in the original feature vector into a quantum state have been proposed in previous studies[4,7,34,35], and our current study employs the two encoding methods presented in Ref. [22] (see Supplementary Information section).

Given quantum encoding $\vec{x} \mapsto \rho_{\vec{x}}$, a quantum pattern is any pair $(\rho_{\vec{x}_j}, \lambda_j)$. A quantum training dataset is defined as the set of all quantum patterns as follows: $\mathscr{S}_{\text{Qtr}} = \left\{(\rho_{\vec{x}_1}, \lambda_1), \ldots, (\rho_{\vec{x}_m}, \lambda_m)\right\}$. For class label $i \in L$, we can define set $\mathscr{S}_{\text{Qtr}}^{i}$ as the set of all object quantum states $\rho_{\vec{x}_j}$ associated with the set $\mathscr{S}_{\text{tr}}^{i}$ of all vectors in the i^{th} class as

$$\mathscr{S}_{\text{Qtr}}^{i} = \{\rho_{\vec{x}_j} : \vec{x}_j \in \mathscr{S}_{\text{tr}}^{i}\}. \tag{2}$$

We now define the quantum centroid, which is a primary component of the classifier. Geometrically, the centroid is the arithmetic mean of all vectors under consideration. Similarly, a quantum centroid for class i is defined as the uniformly weighted convex sum of all density operators in the class, as follows:

$$\rho_{(i)} = \frac{1}{|\mathscr{S}_{\text{Qtr}}^{i}|} \sum_{\vec{x}_j \in \mathscr{S}_{\text{tr}}^{i}} \rho_{\vec{x}_j}, \tag{3}$$

where $|\mathscr{S}_{\text{Qtr}}^{i}|$ is the cardinality of $\mathscr{S}_{\text{Qtr}}^{i}$ (which is equivalent to $|\mathscr{S}_{\text{tr}}^{i}|$, i.e., the cardinality of $\mathscr{S}_{\text{tr}}^{i}$). Thus, ℓ class labels have a one-to-one correspondence with set $\{\rho_{(1)}, \ldots, \rho_{(\ell)}\}$ for all quantum centroids.

We encode the classical vector into multiple copies of a density operator as $\rho_{\vec{x}}$, that is, $\vec{x} \mapsto \rho_{\vec{x}}^{\otimes n}$, to generalize the notion of a quantum centroid as follows:

$$\rho_{(i)}^{(n)} = \frac{1}{|\mathscr{S}_{\text{Qtr}}^{i}|} \sum_{\vec{x}_j \in \mathscr{S}_{\text{tr}}^{i}} \rho_{\vec{x}_j}^{\otimes n}. \tag{4}$$

The motivation to use multiple copies (i.e., tensor product) of a density matrix is supported by kernel theory, which suggests that the classification of a non-linear dataset can be performed with a separating hyperplane in a higher-dimensional feature space[4,7,34,36]. However, the number of copies must be carefully chosen because increasing the dimension of the feature space does not always monotonically increase the classification accuracy[37]. Furthermore, introducing additional copies increases the computation time, thereby demanding that the user tune n to find the appropriate tradeoff between the runtime and classification accuracy. Hereafter, we use the term tensor copy to indicate the density operator created by using multiple copies of a density operator.

PGM Classifier

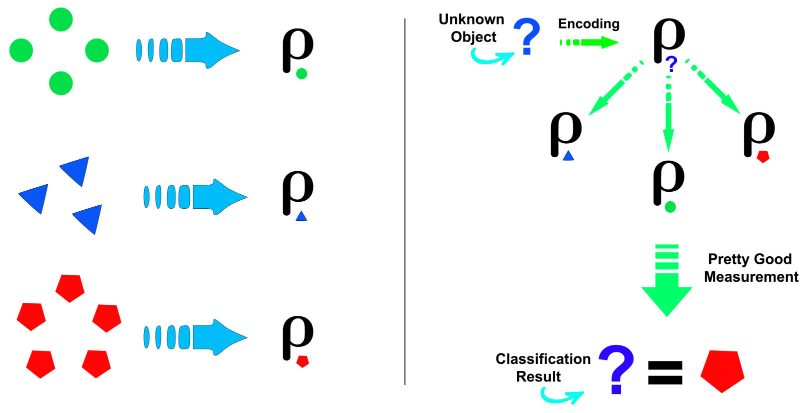
Given that the training information is encoded as quantum centroids, one can apply measurement techniques developed in the context of QSD to construct the score function, f. In brief, QSD describes the distinguishability of different states of a quantum system[32,38]; therefore, it is expected to provide the correct ingredients for differentiating quantum centroids.

For multiclass classification, we are interested in minimum-error discrimination and employ the pretty good measurement (PGM) formalism[39,40]. PGM is a protocol for systematically finding a POVM so that an arbitrary number of unknown quantum states can be discriminated with high success probabilities. Although optimality of the PGM is not guaranteed in general, it is the optimal measurement in several cases, such as with geometrically uniform states[32,41]. The success probability P_{PGM} is related to the optimal success probability, P_{OPT}, as $P_{OPT}^2 \leq P_{PGM} \leq P_{OPT}$[42,43]. The mathematical formulation of PGM is briefly summarized in the Methods section.

Given an ensemble $R = \{(p_i, \rho_{(i)}^{(n)}), \ldots, (p_l, \rho_{(l)}^{(n)})\}$, where p_i is the ratio between the cardinality of the training dataset in class i and the entire training dataset, a set of measurement operators $\{F_i\}$ can be constructed to perform PGM on this ensemble, as explained in the Methods section. Then, for target (unseen) quantum data $\rho_{\vec{x}}^{(n)}$, the PGM classifier is defined as

$$Cl_f(\vec{x}) := \min_i \{i \in L : p_i \mathrm{tr}(F_i \rho_{\vec{x}}^{(n)}) = \max_k \{p_k \mathrm{tr}(F_k \rho_{\vec{x}}^{(n)}), 1 \leq k \leq \ell\}\}. \tag{5}$$

The overall classification process for an unknown object is illustrated in Figure 1.



(a) (b)

Figure 1. Illustration of the QI classification procedure. (a) Construction of quantum centroids using the feature map. The three classes of objects (classical data) are represented as three different types of geometric figures (red balls, blue triangles, and red pentagons). Under the action of the encoding map (blue arrows), the different classes are transformed into quantum states (centroids), in the form of density operators. In principle, the number of classes and the number of elements in each class can be arbitrary. (b) An unknown object is compared with the quantum centroids. Classification is performed using the PGM classifier: an unknown object (represented by a blue question mark) is identified by a red pentagon.

Application of the QSD procedure to the classification problem reveals the following observation: the higher the probability of success of the QSD between quantum centroids, the better the classification accuracy. Consequently, for the binary case, the accuracy of the QI classifier can be expected to be improved by increasing the Helstrom bound. The numerical experiments performed in Refs. [20] empirically demonstrated that the Helstrom bound can be improved by increasing n, namely, the number of tensor copies. We derived the relationship between the Helstrom bound and the number of tensor products in Theorem 1 given in the Supplementary Information section to rigorously prove the empirical evidence. Motivated by this result, we investigated the effect of using multiple copies of a density matrix in the PGM classifier (i.e., n >1). The bound for the success probability of the PGM classifier is defined as follows:

$$\mathtt{PGM_b}(R, n) = \sum_{i=1}^{\ell} p_i \mathrm{tr}(F_i \rho_{(i)}^{(n)}). \tag{6}$$

In the following subsection, we show through numerical experiments that empirical evidence that increasing n can improve the PGM bound, and hence the classification accuracy. However, deriving the general relationship between the PGM bound ($\mathtt{PGM_b}$) and the number of tensor products (n) in the PGM classifier will be taken up as a future study.

Experimental Results

We performed numerical experiments to compare the PGM classifier with other standard multi-class classifiers, and the OvO and OvR strategies with the Helstrom classifier. The standard classifiers for comparison were chosen from the scikit-learn package with default settings. The classification tasks for the following canonical datasets were implemented in the experiments: analcatdata-dmft, balance-scale, car, cleveland-nominal, cloud, confidence, ecoli, haberman, iris, led7, and new-thyroid. The scores used were balanced accuracy, weighted f1, weighted precision, and weighted recall. The main manuscript only reports balanced accuracy as a representative metric, whereas the other scores are presented in the Supplementary Information section. The number of test data points was 0.2 times the number of training data points. The values used for tensor copies (n) were 1, 2, and 3. The hypertuning parameters are the choice of the feature map between stereographic encoding and amplitude encodings[20,21] (see Supplementary Information section) and the scaling factor of the feature values of the original data. The latter is used from a previous result in Ref[21], which demonstrated that the choice of a homogeneous scaling factor of the feature vectors can affect the accuracy of the classification. The balanced accuracy obtained for each classifier applied to each dataset is depicted as a heatmap in Fig. 2. The average balanced accuracy for each classifier for all datasets presented in Table 1 reveals that the PGM classifier outperforms all classifiers under consideration, except for the multi-layer perceptron. Among the distance-based classifiers, the PGM classifier achieved the best accuracy.

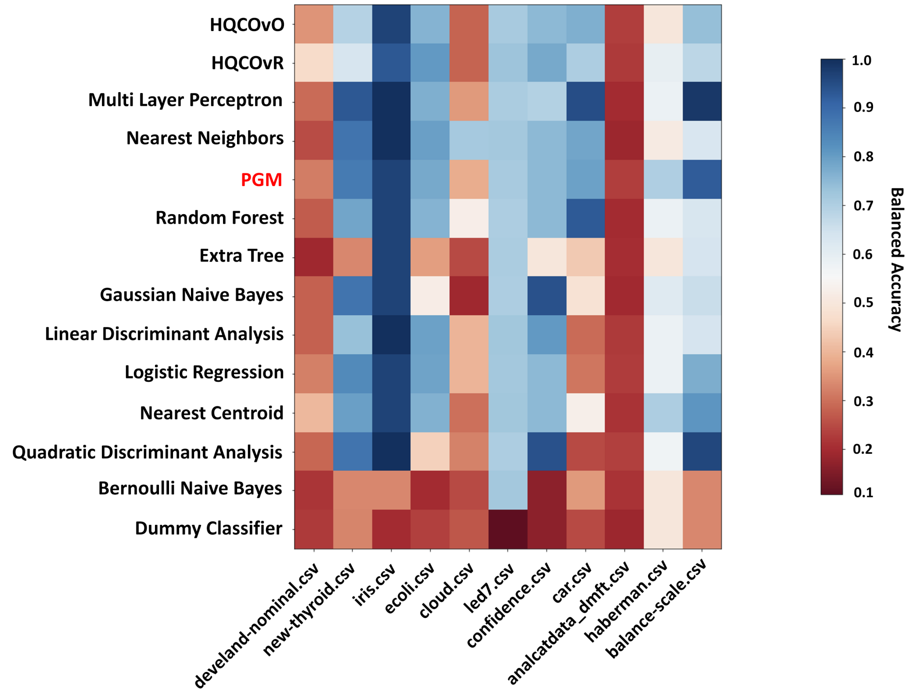


Figure 2. Heatmap of the values of each classifier applied to each dataset.

Classifier	Average Balanced Accuracy
Multi Layer Perceptron	0.679 ± 0.276
PGM	$\mathbf{0.675 \pm 0.238}$
Nearest Neighbors	0.657 ± 0.238
Random Forest	0.646 ± 0.231
Nearest Centroid	0.633 ± 0.227
HQC OvR	0.621 ± 0.208
HQC OvO	0.614 ± 0.225
Logistic Regression	0.606 ± 0.24
Quadratic Discriminant Analysis	0.6 ± 0.294
Linear Discriminant Analysis	0.589 ± 0.243
Gaussian Naive Bayes	0.586 ± 0.269
Extra Tree	0.463 ± 0.226
Bernoulli Naive Bayes	0.329 ± 0.153
Dummy Classifier	0.255 ± 0.1

Table 1. Average value of Balanced Accuracy for each classifier over all datasets.

Furthermore, the biclustering heatmap depicted in Fig. 3, which compares classifiers for each dataset shows that the average winning ratio (the ratio of the number of datasets for which one classifier outperforms the other) of the PGM classifier outperforms those of all other classifiers. The average winning ratio presented in Table 2 supports this assumption.

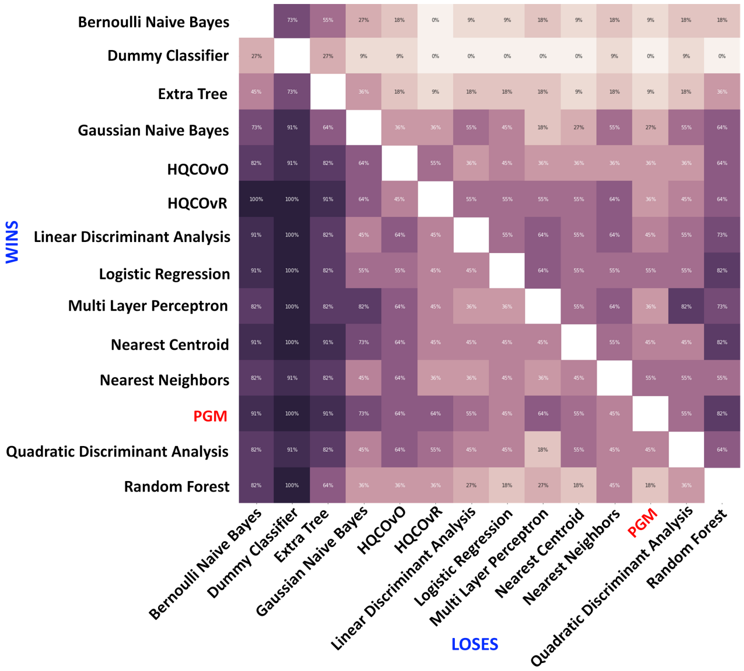


Figure 3. Percentage of datasets for which the balanced accuracy of classifier A (in the column) outperforms that of classifier B (in the row). Darker (lighter) color indicates higher (lower) percentage.

Table 3 presents a comparison of the balanced accuracy and average runtime of the PGM classifier, which is a direct multiclass classifier, with the OvO and OvR strategies based on the HQC from Refs. [20,33]. The results revealed that the PGM classifier outperformed both OvO and OvR with HQC in terms of both accuracy and runtime. The balanced accuracies of the PGM, OvO, and OvR Helstrom classifiers averaged over all datasets were 0.675 ± 0.238, 0.614 ± 0.225, and 0.612 ± 0.208, respectively.

The PGM classifier performed similarly well in terms of the other statistical metrics (i.e., weighted f1, weighted precision, and weighted recall), whose values are given in the Supplementary Information section. Finally, Fig. 4 demonstrates that the

Classifier	**Average winning ratio (%)**
PGM	**67.32**
Multi Layer Perceptron	66.01
Logistic Regression	65.36
Linear Discriminant Analysis	64.71
Nearest Centroid	64.71
HQC OvR	64.05
Quadratic Discriminant Analysis	58.82
Nearest Neighbors	58.17
HQC OvO	54.9
Gaussian Naive Bayes	49.02
Random Forest	43.79
Extra Tree	23.53
Bernoulli Naive Bayes	20.26
Dummy Classifier	7.84

Table 2. Average winning ratio of balanced accuracy for each classifier over all datasets.

	Balanced Accuracy			Mean Runtime(s)		
Dataset \ Classifier	OvR	OvO	PGM	OvR	OvO	PGM
Analcatdata_dmft	0.224 ± 0.045	0.232 ± 0.041	0.232 ± 0.038	221.2	472.3	64.4
Balance-scale	0.681 ± 0.048	0.739 ± 0.070	0.924 ± 0.051	99.2	99.5	46.2
Car	0.706 ± 0.046	0.768 ± 0.014	0.793 ± 0.011	2497.9	3477.9	756.9
Cleveland-nominal	0.466 ± 0.061	0.348 ± 0.069	0.319 ± 0.057	2391.2	4823	417.4
Cloud	0.283 ± 0.106	0.283 ± 0.076	0.383 ± 0.094	1437.1	2257	229.7
Confidence	0.778 ± 0.082	0.750 ± 0.041	0.750 ± 0.041	31.6	88.6	13.6
Ecoli	0.805 ± 0.065	0.760 ± 0.057	0.779 ± 0.045	2437.1	4955.1	460.1
Haberman	0.595 ± 0.050	0.500 ± 0.000	0.702 ± 0.033	11.2	13.2	13.8
Iris	0.933 ± 0.057	0.967 ± 0.033	0.967 ± 0.042	49.2	53.1	30
Led7	0.728 ± 0.024	0.712 ± 0.028	0.713 ± 0.024	28931	113356.1	6259.8
New-thyroid	0.635 ± 0.131	0.690 ± 0.123	0.867 ± 0.049	129.8	145	60.6

Table 3. Comparison of balanced accuracy between HQC OvO, HQC OvR, and PGM

classification performance of balanced accuracy, weighted f1, weighted precision, and weighted recall with the number of tensor copies, n. Without loss of generality, the benchmark evaluation was performed with four representative datasets: New Thyroid, Iris, Analcatdata, and Confidence. It was observed for all datasets that increasing n improves all quantifiers of the classification performance. Furthermore, it is noteworthy that the PGM bound monotonically increased with n for all datasets tested here, up to $n = 5$.

Extension to the quantum circuit model

Although the QI algorithm developed in this study can be immediately applied to real-world problems using classical computers, it is important to discuss the prospects of employing quantum computers because quantum hardware is improving progressively. Because our algorithm is based on the mathematical formalism of quantum mechanics, all steps described in the previous section can be implemented on a quantum circuit in principle. Here, we briefly outline the implementation of our algorithm on quantum hardware.

Universal quantum computation allows the preparation of an arbitrary quantum state; hence, quantum-encoding maps are feasible in principle. The quantum centroid, $\rho_{(i)}^{(n)}$, can be constructed by classically sampling tensor products of $\rho_{\vec{x}_j}$ from n. By definition, storing such quantum information requires only n quantum systems, whereas the classical memory increases exponentially with n. Thus, the implementation has an exponential memory advantage at this point. It should be noted that the quantum encoding step often requires deep quantum circuits[35,44,45]; hence, fault tolerance is required. The second step involves the implementation of QSD between an unknown encoded state regarding quantum centroids. The PGM technique is based on POVM, which can be implemented on a quantum circuit using Neumark's theorem[46,47]. In this case, the number of ancilla qubits onto which the projective measurement is performed, increases logarithmically with the number of POVM

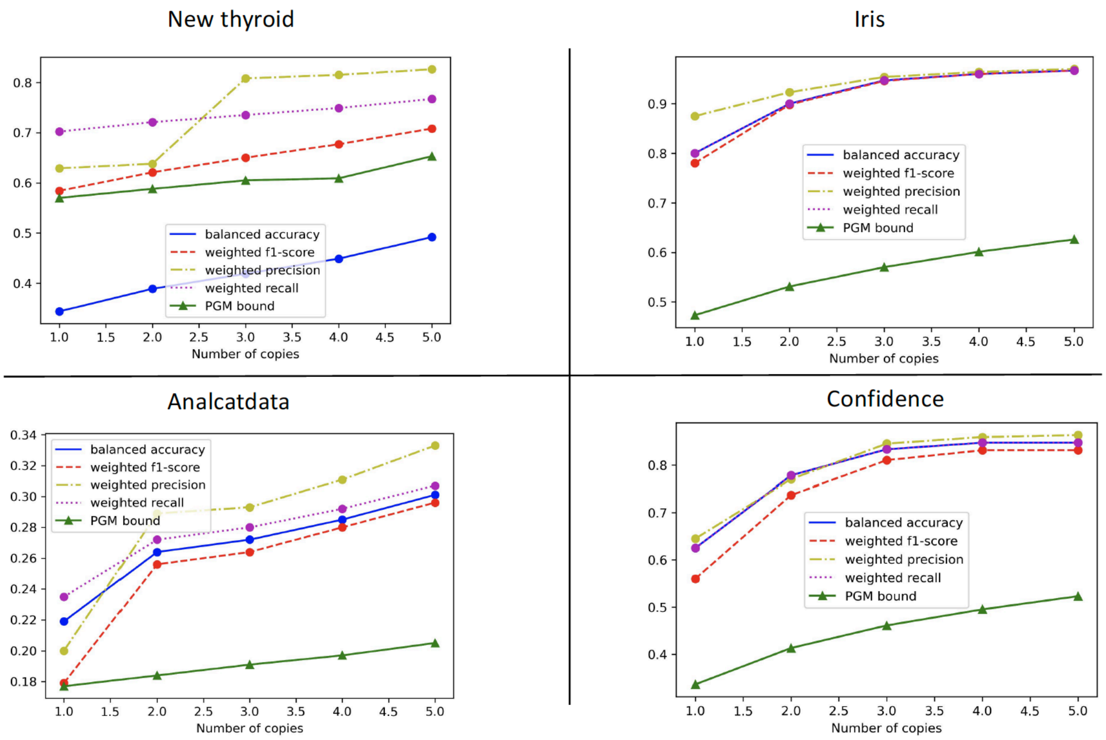


Figure 4. The plots demonstrate the improvement in metrics: balanced accuracy, weighted f1-score, weighted precision, weighted recall, and PGM bound, with an increasing number of tensor copies, n, of the original data.

elements, which equals the number of classes, ℓ. Although Ref.[47] presented a basic quantum circuit structure for implementing an arbitrary POVM, determining the exact circuit parameters for a POVM remains a challenge.

Discussion

The development of full-fledged quantum computation remains a long-term prospect; therefore, QI algorithms implemented on classical computers are a promising approach to enhance existing machine learning methods. Previous QI classifiers were able to perform only binary classification, thereby posing challenges for general multi-class datasets. Our QI algorithm for direct multi-class classification is based on the theory of QSD, which provides a systematic approach to find suboptimal measurements for discriminating multiple quantum states. In our algorithm, training data of class i are encoded as a density matrix that must be discriminated from other density matrix representations of the training data in different classes.

We conducted numerical experiments to compare our algorithm with existing standard classifiers, both classical and quantum-inspired, using 11 benchmarking datasets. The experimental results revealed an improvement in classification accuracy in most cases. Our algorithm, when compared with the OvO and OvR strategies with a QI binary classifier known as the Helstrom quantum classifier, not only provided higher classification accuracy but also reduced runtime. This is owing to the ability of our algorithm to directly perform multi-class classification. Furthermore, we showed that classification accuracy can be further improved by increasing the number of copies of the density matrix that represents a classical feature vector.

The experimental results are remarkable because they illustrate the potential advantage of using the mathematics of quantum theory while constructing a machine learning model, even if all hardware components are classical. This raises intriguing questions regarding the development of algorithms for classification and pattern recognition.

Finally, we highlight that the use of quantum computers is a natural attempt to further improve our QI algorithm. We provided empirical evidence that increasing the number of copies of a quantum system used in the construction of a quantum centroid improves classification accuracy. However, such an approach exponentially increases the computational resources such as runtime and memory. Although it is possible to exploit quantum computers to implement the PGM classifier, as briefly explained in the previous section, finding an efficient approach to identify quantum circuit parameters that implement the

POVM desired for a given dataset remains to be investigated. The time complexity of the quantum PGM classifier needs to be evaluated in future study for comparison with classical runtime. Another important future research direction is to rigorously prove the relationship between the PGM bound, which determines the classification accuracy, and the number of copies of the quantum-encoded data.

Findings, limitations, and potential future developments

As discussed in the Introduction section, QI machine learning is an extremely interesting branch of quantum machine learning because it offers relevant benefits, without the involvement of quantum computers. In particular, as shown in[20], it is possible to design an algorithm inspired by QSD that exhibits excellent performance when applied to a binary classification context. However, extending this model to multi-class classification problems is challenging. Herein, we introduce a model inspired by QSD. The model can perform multi-class classification without using a combinatorial strategy. The comparison of the model with 11 standard classifiers revealed that it performs well not only in terms of the accuracy of the classification but also regarding other relevant statistical metrics (weighted f1, weighted precision, and weighted recall; see Supplementary Information section for details). In particular, the average value of the balanced accuracy obtained by our model outperforms all the other standard classifiers (except multi layer perceptron; see Table 1), and it is the best in terms of the average winning ratio (see Table 2). Furthermore, our method can perform an accurate multi-class classification by avoiding any type of OvO or OvR procedures. Comparison of our new QI multi-class classifier with the OvO and OvR, reveals that the former outperforms the latter, not only in terms of accuracy but also time complexity. A detailed comparison is presented in Table 3. Thus, our new classifier is more accurate and efficient. This represents a promising improvement with respect to the actual state-of-the-art, thereby paving the way for several potential practical applications. A comparison of our algorithm with 11 standard classifiers yielded promising results. However, further improvements to our technique should incorporate state-of-the-art technologies. To this end, we discussed the possibility of moving from QI to real quantum machine learning by extending our model to a real quantum circuit model by employing Neumark's theorem. In future, we will present a prototypical example of an application of our multi-class classifier on a test set by using a quantum circuit, and additionally provide a simulation using the IBM-Q platform, demonstrating that the classification results are not significantly affected by different types of noise. A limitation of this approach is that, in principle, Neumark's theorem only allows the design of a quantum circuit capable of performing the test part of a QI algorithm. The training step, which is the most time-consuming, was still computed using a classical computer. Therefore, further investigation is required for full implementation on a quantum computer because the expression of the quantum algorithm that can implement the training part of the classifier is absent. In addition, the tensor copies strategy described above is particularly burdensome for classical computers, and an optimal strategy to compute it using a quantum computer for realizing the aims of the algorithms is also absent. To this end, our future efforts will include designing a quantum circuit capable of performing the training step, including tensor copies, and evaluating the cost of the process compared with other standard classifiers. Moreover, our future research will include the application of our algorithm to real datasets in different practical contexts and explore several potential high-impact applications. In particular, we will focus on bioimaging applications that have already been considered in recent studies[22,33]. Finally, another limitation of the model is that the choice of the particular encoding (feature map) used in the QI model is given by hypertuning that is manifestly dataset dependent (see Supplementary Information section). Therefore, future investigation will additionally address finding an a-priori optimal encoding that simultaneously maximizes the accuracy and efficiency of the classification.

Methods

Pretty Good Measurement

Given an ensemble of possible quantum states with their respective a-priori probabilities,

$$R = \{(\mathrm{p}_1, \rho_1), \cdots, (\mathrm{p}_\ell, \rho_\ell)\}, \tag{7}$$

no known analytical description exists for the exact optimal measurement for discriminating the states in R. However, the so-called PGM[30–32] performs well in several situations. The average state of R is given by:

$$\sigma = \sum_{i=1}^{\ell} \mathrm{p}_i \rho_i$$

and for any $i : 1 \leq i \leq \ell$, operators

$$E_i = (\sigma^{\neg})^{1/2} \mathrm{p}_i \rho_i (\sigma^{\neg})^{1/2},$$

where $\sigma^{\neg}$ is the pseudoinverse (or Moore-Penrose inverse) of σ. For any i with $1 \leq i \leq \ell$, we define operators

$$F_i = E_i + \frac{1}{\ell} P_{ker(\sigma)}, \tag{8}$$

where $P_{ker(\sigma)}$ denotes the projection associated with the subspace spanned by the kernel σ. It was found that the map

$$\mathscr{F} : \{1,2,\cdots,\ell\} \to \mathscr{B}(\mathbb{C}^n)^+$$

is a measurement because $\sum_{i=1}^{\ell} E_i = P_{im(\sigma)}$. The map $\mathscr{F}$, called the PGM[48], is sub-optimal [49] because

$$P_{OPT}^2 \leq P_{PGM} \leq P_{OPT},$$

where P_{PGM} is the success probability of PGM and P_{OPT} is the optimal success probability. Furthermore, if $\mathrm{tr}(P_{Ker(\sigma)}) = 0$, we can replace F_i with E_i in the above equation.

Statistical metrics

The statistical metrics analyzed in this study to evaluate the classification follow the standard nomenclature in machine learning. The metrics are defined as follows:

$$\begin{aligned} \text{Balanced accuracy} &= \frac{TPR+TNR}{2}, \quad \text{weighted-f1} = \frac{2TP}{2TP+FP+FN}, \\ \text{weighted-precision} &= \frac{TP}{TP+FP}, \quad \text{weighted-recall} = \frac{TP}{TP+FN}, \end{aligned}$$

where TPR is the true positive rate, TNR is the true negative rate, FPR is the false positive rate, and FNR is the false negative rate of the classification. In the main manuscript, we only report the balanced accuracy, whereas the remaining metrics are reported in the Supplementary Information section.

Experiments

All datasets used in the numerical experiments can be downloaded from the UCI machine learning repository at https://archive.ics.uci.edu/ml/index.ph or from GitHub repository at https://github.com/EpistasisLab/pmlb. Information regarding the datasets is summarized in Table 4. The amount of training data, m, was chosen to be approximately 80% of the total number of objects, M.

Name	M	d	l
Analcatdata-dmft	797	4	6
Balance-scale	625	4	3
Car	1728	6	4
Cleveland-nominal	303	7	5
Cloud	108	7	4
Confidence	72	3	6
Ecoli	327	7	5
Haberman	306	3	2
Iris	150	4	3
Led7	3200	7	10
New-thyroid	215	5	3

Table 4. Details of datasets used in this study. Here, M denotes the number of objects, d denotes the number of features, and l denotes the number of classes.

A comparison with the existing classifiers was performed using the scikit-learn package. For the QI classifiers (i.e., PGM, HQC OvO, and HQC OvR), the hypertuning parameters were the number of copies (n) ranging from 1 to 3, rescaling factor ranging from 0.1 to 3 with step of 0.1, and the encoding method was chosen from stereographic or amplitude (see Supplementary Information section). For all datasets, setting $n = 3$ yielded the highest classification accuracy. Owing to the increase in runtime, we did not increase n beyond 3. To test the classification performance with respect to n, we increased n up to 5 for the selected dataset. In this experiment, we used amplitude encoding and rescaling factor of 1, indicating that the data were not rescaled. Finally, all experiments were performed on a 3.6 GHz 10-Core i9 Intel processor and 64 GB RAM.

Acknowledgements

R.G. and G.S. acknowledge the support from the Italian Ministry of Education, PRIN project (Grant No. 20173YP4N3)and the FdS (Fondazione di Sardegna) project (cup:F73C22001360007). D.K.P. acknowledges the support from the National Research Foundation of Korea (Grant No. 2019R1I1A1A01050161 and Grant No. 2021M3H3A1038085). We thank Keng Loon Chow for his valuable suggestions during the conceptualization of this study.

Author contributions statement

R.G., F.H., D.P., and G.S. conceived the experiments; R.G. conducted the experiment (s); R.G., F.H., D.P., and G.S. analyzed the results; R.G., H.F., and G.S. conceived the mathematical framework; F.H., D.P., C.B., and G.S. conceived the physical framework. All authors reviewed the manuscript and contributed equally to this work.

Competing interests

The authors declare that they have no conflicts of interest.

References

1. Biamonte, J. *et al.* Quantum machine learning. *Nature* **549**, 195 EP – (2017).

2. Rebentrost, P., Mohseni, M. & Lloyd, S. Quantum support vector machine for big data classification. *Phys. Rev. Lett.* **113**, 130503, DOI: 10.1103/PhysRevLett.113.130503 (2014).

3. Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum principal component analysis. *Nat. Phys.* **10**, 631–633, DOI: 10.1038/nphys3029 (2014).

4. Havlícek, V. *et al.* Supervised learning with quantum-enhanced feature spaces. *Nature* **567**, 209–212, DOI: 10.1038/s41586-019-0980-2 (2019).

5. Blank, C., Park, D. K., Rhee, J.-K. K. & Petruccione, F. Quantum classifier with tailored quantum kernel. *npj Quantum Inf.* **6**, 41, DOI: 10.1038/s41534-020-0272-6 (2020).

6. Liu, N. & Rebentrost, P. Quantum machine learning for quantum anomaly detection. *Phys. Rev. A* **97**, 042315, DOI: 10.1103/PhysRevA.97.042315 (2018).

7. Liu, Y., Arunachalam, S. & Temme, K. A rigorous and robust quantum speed-up in supervised machine learning. *Nat. Phys.* **17**, 1013–1017, DOI: 10.1038/s41567-021-01287-z (2021).

8. Abbas, A. *et al.* The power of quantum neural networks. *Nat. Comput. Sci.* **1**, 403–409, DOI: 10.1038/s43588-021-00084-1 (2021).

9. Mangini, S., Tacchino, F., Gerace, D., Bajoni, D. & Macchiavello, C. Quantum computing models for artificial neural networks. *EPL (Europhysics Lett.* **134**, 10002, DOI: 10.1209/0295-5075/134/10002 (2021).

10. Hur, T., Kim, L. & Park, D. K. Quantum convolutional neural network for classical data classification (2021). 2108.00661.

11. Montiel, O., Rubio, Y., Olvera, C. & Rivera, A. Quantum-inspired acromyrmex evolutionary algorithm. *Sci. Reports* **9**, 12181, DOI: 10.1038/s41598-019-48409-5 (2019).

12. Mugel, S. *et al.* Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks (2020). 2007.00017.

13. Mugel, S., Lizaso, E. & Orus, R. Use cases of quantum optimization for finance (2020). 2010.01312.

14. Chen, J., Cheng, H.-P. & Freericks, J. A quantum-inspired algorithm for the factorized form of unitary coupled cluster theory (2020). 2008.06637.

15. Alvarez-Alvarado, M. S., Alban-Chacón, F. E., Lamilla-Rubio, E. A., Rodríguez-Gallegos, C. D. & Velásquez, W. Three novel quantum-inspired swarm optimization algorithms using different bounded potential fields. *Sci. Reports* **11**, 11655, DOI: 10.1038/s41598-021-90847-7 (2021).

16. Bauer, B., Bravyi, S., Motta, M. & Chan, G. K.-L. Quantum algorithms for quantum chemistry and quantum materials science. *Chem. Rev.* **120**, 12685–12717, DOI: 10.1021/acs.chemrev.9b00829 (2020). https://doi.org/10.1021/acs.chemrev.9b00829.

17. Hull, I., Sattath, O., Diamanti, E. & Wendin, G. Quantum technology for economists (2021). 2012.04473.

18. Tang, E. A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, 217–228, DOI: 10.1145/3313276.3316310 (Association for Computing Machinery, New York, NY, USA, 2019).

19. Tang, E. Quantum-inspired classical algorithms for principal component analysis and supervised clustering (2019). ArXiv:1811.00414, 1811.00414.

20. Sergioli, G., Giuntini, R. & Freytes, H. A new quantum approach to binary classification. *PLOS ONE* **14**, 1–14, DOI: 10.1371/journal.pone.0216224 (2019).

21. Sergioli, G., Bosyk, G. M., Santucci, E. & Giuntini, R. A quantum-inspired version of the classification problem. *Int. J. Theor. Phys.* **56**, 3880–3888, DOI: 10.1007/s10773-017-3371-1 (2017).

22. Sergioli, G. *et al.* Quantum-inspired minimum distance classification in a biomedical context. *Int. J. Quantum Inf.* **16**, 1840011, DOI: 10.1142/S0219749918400117 (2018). https://doi.org/10.1142/S0219749918400117.

23. Tiwari, P. & Melucci, M. Binary classifier inspired by quantum theory. *Proc. AAAI Conf. on Artif. Intell.* **33**, 10051–10052, DOI: 10.1609/aaai.v33i01.330110051 (2019).

24. Zhang, J. *et al.* Interactive quantum classifier inspired by quantum open system theory. In *2021 International Joint Conference on Neural Networks (IJCNN)*, 1–7, DOI: 10.1109/IJCNN523inproceedings1Zhang_2021 (2021).

25. Zhang, J. *et al.* Quantum entanglement inspired correlation learning for classification. In Gama, J. *et al.* (eds.) *Advances in Knowledge Discovery and Data Mining*, 58–70 (Springer International Publishing, Cham, 2022).

26. Sergioli, G. Quantum and quantum-like machine learning: a note on differences and similarities. *Soft Comput.* **24**, 10247–10255, DOI: 10.1007/s00500-019-04429-x (2020).

27. Villmann, T., Engelsberger, A., Ravichandran, J., Villmann, A. & Kaden, M. Quantum-inspired learning vector quantizers for prototype-based classification. *Neural Comput. Appl.* **34**, 79–88, DOI: 10.1007/s00521-020-05517-y (2022).

28. Helstrom, C. W. Quantum detection and estimation theory. *J. Stat. Phys.* **1**, 231–252, DOI: 10.1007/BF01007479 (1969).

29. Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, New York, NY, USA, 2011), 10th edn.

30. Watrous, J. *The Theory of Quantum Information* (Cambridge University Press, 2018).

31. Barnett, S. M. & Croke, S. Quantum state discrimination. *Adv. Opt. Photon.* **1**, 238–278, DOI: 10.1364/AOP.1.000238 (2009).

32. Bae, J. & Kwek, L.-C. Quantum state discrimination and its applications. *J. Phys. A: Math. Theor.* **48**, 083001, DOI: 10.1088/1751-8113/48/8/083001 (2015).

33. Sergioli, G. *et al.* A quantum-inspired classifier for clonogenic assay evaluations. *Sci. Reports* **11**, 2830, DOI: 10.1038/s41598-021-82085-8 (2021).

34. Schuld, M. & Killoran, N. Quantum machine learning in feature hilbert spaces. *Phys. Rev. Lett.* **122**, 040504, DOI: 10.1103/PhysRevLett.122.040504 (2019).

35. Araujo, I. F. *et al.* Configurable sublinear circuits for quantum state preparation (2021). 2108.10182.

36. Park, D. K., Blank, C. & Petruccione, F. The theory of the quantum kernel-based binary classifier. *Phys. Lett. A* **384**, 126422, DOI: https://doi.org/10.1016/j.physleta.2020.126422 (2020).

37. Huang, H.-Y. *et al.* Power of data in quantum machine learning. *Nat. Commun.* **12**, 2631, DOI: 10.1038/s41467-021-22539-9 (2021).

38. Fanizza, M., Mari, A. & Giovannetti, V. Optimal universal learning machines for quantum state discrimination. *IEEE Transactions on Inf. Theory* **65**, 5931–5944, DOI: 10.1109/TIT.2019.2916646 (2019).

39. Hausladen, P. & Wootters, W. K. A 'pretty good' measurement for distinguishing quantum states. *J. Mod. Opt.* **41**, 2385–2390, DOI: 10.1080/09500349414552221 (1994). https://doi.org/10.1080/09500349414552221.

40. Mochon, C. Family of generalized "pretty good" measurements and the minimal-error pure-state discrimination problems for which they are optimal. *Phys. Rev. A* **73**, 032328, DOI: 10.1103/PhysRevA.73.032328 (2006).

41. Eldar, Y., Megretski, A. & Verghese, G. Optimal detection of symmetric mixed quantum states. *IEEE Transactions on Inf. Theory* **50**, 1198–1207, DOI: 10.1109/TIT.2004.828070 (2004).

42. Barnum, H. & Knill, E. Reversing quantum dynamics with near-optimal quantum and classical fidelity. *J. Math. Phys.* **43**, 2097–2106, DOI: 10.1063/1.1459754 (2002). https://aip.scitation.org/doi/pdf/10.1063/1.1459754.

43. Montanaro, A. On the distinguishability of random quantum states. *Commun. Math. Phys.* **273**, 619–636, DOI: 10.1007/s00220-007-0221-7 (2007).

44. Araujo, I. F., Park, D. K., Petruccione, F. & da Silva, A. J. A divide-and-conquer algorithm for quantum state preparation. *Sci. Reports* **11**, 6329, DOI: 10.1038/s41598-021-85474-1 (2021).

45. de Veras, T. M. L., de Araujo, I. C. S., Park, D. K. & da Silva, A. J. Circuit-based quantum random access memory for classical data with continuous amplitudes. *IEEE Transactions on Comput.* **70**, 2125–2135, DOI: 10.1109/TC.2020.3037932 (2021).

46. Gelfand, I. M. & Neumark, M. A. On the imbedding of normed rings into the ring of operators in hilbert space. *Matematiceskij sbornik* **54**, 197–217 (1943).

47. Yordanov, Y. S. & Barnes, C. H. W. Implementation of a general single-qubit positive operator-valued measure on a circuit-based quantum computer. *Phys. Rev. A* **100**, 062317, DOI: 10.1103/PhysRevA.100.062317 (2019).

48. Hausladen, P. & Wootters, W. K. A pretty good measurement for distinguishing quantum states. *J. Mod. Opt.* **41**, 2385–2390, DOI: 10.1080/09500349414552221 (1994). https://doi.org/10.1080/09500349414552221.

49. Barnum, H. & Knill, E. Reversing quantum dynamics with near-optimal quantum and classical fidelity. *J. Math. Phys.* **43**, 2097–2106, DOI: 10.1063/1.1459754 (2002). https://aip.scitation.org/doi/pdf/10.1063/1.1459754.

50. Helstrom, C. W. *Quantum detection and estimation theory* (Academic Press New York, 1976).

51. Ruskai, M. B. *Beyond Strong Subadditivity? Improved Bounds on the Contraction of Generalized Relative Entropy*, 350–366 (World Scientific, 1994). https://www.worldscientific.com/doi/pdf/10.1142/9789812798251_0014.

Supplementary Information

0.1 Quantum feature maps

As discussed above, an essential process of our algorithm is to encode the features of objects to be classified into quantum states. There are infinite options to map an object vector to a density matrix. Different encodings have have been considered in previous studies[22]. In particular, the data in this study were obtained by applying hypertuning between the following two feature maps (i.e., encodings).

Amplitude encoding:

For each vector $\vec{x} = \left(x^1, \dots, x^d\right) \in \mathbb{R}^d$, let $\vec{x'} = (\frac{x^1}{\sqrt{\sum_{i=1}^d (x^i)^2 + 1}}, \dots, \frac{x^d}{\sqrt{\sum_{i=1}^d (x^i)^2 + 1}}, \frac{1}{\sqrt{\sum_{i=1}^d (x^i)^2 + 1}})$. The amplitude encoding is a map, $\mathbb{R}^d \ni \vec{x} \mapsto \rho_{\vec{x}} = \tilde{x}^\dagger \tilde{x}$.

Stereographic encoding:

For each vector $\vec{x} = \left(x^1, \dots, x^d\right) \in \mathbb{R}^d$, consider the vector $\vec{x'} \in \mathbb{R}^{d+1}$ given by $\vec{x'} = \alpha(2x^1, \dots, 2x^d, \sum_{i=1}^d (x^i)^2 - 1)$, with $\alpha = \frac{1}{\sum_{i=1}^d (x^i)^2 + 1}$. The stereographic encoding is the map $\mathbb{R}^d \ni \vec{x} \mapsto \rho_{\vec{x}} = \vec{x'}^\dagger \vec{x'}$.

In both cases, $\rho_{\vec{x}}$ is $(d+1)$-dimensional pure state. Here, $\rho_{\vec{x}}$ is the object quantum state associated with the object vector, $\vec{x}$.

It should be noted that the protocols for the application of the QI algorithms discussed here (HQC and PGM classifiers) also include hypertuning over a rescaling factor. Formally, each component x^i of a given vector $\vec{x}$ is replaced with $x^i + \delta$, where δ is a real number called the *rescaling factor*. In[21] we showed that rescaling the features of all the objects of a given dataset before encoding can improve the accuracy of the classification.

0.2 Helstrom bound and tensor copies

In this Appendix, we provide a proof of Theorem 1. In particular, we demonstrate that the Helstrom bound calculated for two quantum centroids increases whenever the centroids are calculated over datasets in which each element is obtained by a tensor copy of itself.

Theorem 1 *Consider a dataset partitioned into two classes, which are labeled by numbers* 1 *and* 2. *Consider extracting a training set for each class and providing a given quantum encoding to each vector of each class. Let us denote by S^i_{Qtr} the set of the encoded vectors $\rho_{\vec{x}_j}$ belonging to the i^{th} class of the training dataset (with $i \in \{0, 1\}$).*

Following Eq. (3), the quantum centroids ρ_1 and ρ_2 for the two classes of the training set are calculated, where n is computed for ρ_1 as the cardinality of class 1 *of the training set, and similarly for ρ_2. Formally, $\rho_1 = \frac{1}{|S^1_{Qtr}|} \sum_j \rho_{\vec{x}_j}$, where $\rho_{\vec{x}_j}$ is an arbitrarily encoded vector of class* 1 *of the quantum training set. Similarly, the centroid, ρ_2, of the second class of the quantum training dataset can be calculated. Furthermore, $\rho_i^{(n)}$ the centroid of class i is obtained by making n-tensor copies of each vector of the quantum training dataset of class i, that is, $\rho_i^{(n)} = \frac{1}{|S^1_{Qtr}|} \sum_j \rho_{\vec{x}_j} \otimes \cdots \otimes \rho_{\vec{x}_j}$. Following its standard formulation[50] the Helstrom bound for ρ_1 and ρ_2 is given by*

$$\mathtt{H_b}(\rho_1, \rho_2) = \frac{1}{2}(1 + \mathtt{T}(p_1\rho_1, p_2\rho_2)), \tag{S1}$$

where T *denotes the trace distance induced by the trace norm, $\mathtt{T}(\sigma, \tau) = \frac{1}{2}\mathtt{Tr}[\sqrt{(\sigma - \tau)^\dagger(\sigma - \tau)}]$.*

We prove that, for any $k \in \mathbb{N}^+$

$$\mathtt{H_b}(\rho^{(k)}_{(1)}, \rho^{(k)}_{(2)}) \leq \mathtt{H_b}(\rho^{(k+1)}_{(1)}, \rho^{(k+1)}_{(2)}).$$

Proof:

To prove Theorem 1, it suffices to show that:

$$\mathtt{T}(\rho^{(k)}_{(1)}, \rho^{(k)}_{(2)}) \leq \mathtt{T}(\rho^{(k+1)}_{(1)}, \rho^{(k+1)}_{(2)}).$$

Consider the Hilbert space $\mathscr{H} = \mathscr{H}_1 \otimes \mathscr{H}_2$, where: $\mathscr{H}_1 = \mathbb{C}^d$ and $\mathscr{H}_2 = \otimes^k \mathbb{C}^d$. Let $\mathtt{Tr}_1$ be a partial trace of the first component of $\mathscr{H}$, that is, $\mathscr{H}_1$.

By using the definition of the quantum centroid (Eq. (3)) and the linearity of the partial trace operator,

$$\begin{aligned}
\mathtt{Tr_1}(\rho_1^{(k+1)}) &= \mathtt{Tr_1}\left(\frac{1}{|\mathscr{S}^1_{\mathrm{Qtr}}|}\sum_{\vec{x}_j\in\mathscr{S}^1_{\mathrm{tr}}}\otimes^{(k+1)}\rho_{\vec{x}_j}\right)\\
&= \frac{1}{|\mathscr{S}^1_{\mathrm{Qtr}}|}\sum_{\vec{x}_j\in\mathscr{S}^1_{\mathrm{tr}}}\mathtt{Tr_1}\left(\otimes^{(k+1)}\rho_{\vec{x}_j}\right)\\
&= \frac{1}{|\mathscr{S}^1_{\mathrm{Qtr}}|}\sum_{\vec{x}_j\in\mathscr{S}^1_{\mathrm{tr}}}\otimes^{(k)}\rho_{\vec{x}_j}\\
&= \rho_1^{(k)}
\end{aligned} \tag{S2}$$

The above list of equations can be physically interpreted as follows. Given a product state of $k+1$ components, tracing out the first system will not alter the remaining k states. Similarly,

$$\mathtt{Tr_1}(\rho_2^{(k+1)}) = \frac{1}{|\mathscr{S}^2_{\mathrm{Qtr}}|}\sum_{\vec{x}_j\in\mathscr{S}^2_{\mathrm{tr}}}\otimes^{(k)}\rho_{\vec{x}_j} = \rho_2^{(k)}. \tag{S3}$$

The trace distance satisfies the contractivity property under the action of complete trace preservation of positive maps (i.e., trace-preserving quantum operators, see[51]*). Because* $\mathtt{Tr_1}$ *is a trace-preserving quantum operation,*

$$\mathtt{T}\left(\mathtt{Tr_1}\left(\rho_1^{(k+1)}\right),\mathtt{Tr_1}\left(\rho_2^{(k+1)}\right)\right) \leq \mathtt{T}\left(\rho_1^{(k+1)},\rho_2^{(k+1)}\right). \tag{S4}$$

Thus, by Eqs. (S2), (S3), and (S4), we can conclude that

$$\mathtt{T}(\rho_1^{(k)},\rho_2^{(k)}) \leq \mathtt{T}(\rho_1^{(k+1)},\rho_2^{(k+1)}).$$

We recall that this theorem has a relevant impact regarding classification. Thus, making the tensor product of each quantum encoded vector of the initial datasets increases the probability of distinguishing the two quantum centroids of each class (with a natural benefit on the accuracy of the classification).

We empirically demonstrate that this argument can also be extended to the n-ary PGM classifier. Equation 6 represents the PGM-bound. We simulated (with Wolfram Mathematica) a random three-class dataset and applied the PGM protocol with the original dataset before and after using a new dataset created by making a tensor copy of each element with itself. Finally, we calculated the difference between the value of the PGM-bound with respect to the original dataset ($\mathtt{PGM_b}^{(1)}$) and the tensor copy dataset ($\mathtt{PGM_b}^{(2)}$). We repeated the procedure for 10^4 random datasets and observed that $\mathtt{PGM_b}^{(2)} - \mathtt{PGM_b}^{(1)} > 0$ always, as depicted in Supplementary Fig.1(a).

The same simulation has been repeated comparing $\mathtt{PGM_b}^{(3)}$ with $\mathtt{PGM_b}^{(2)}$ (Supplementary Fig.1(b)), $\mathtt{PGM_b}^{(4)}$ with $\mathtt{PGM_b}^{(3)}$ (Supplementary Fig.1(c)), and $\mathtt{PGM_b}^{(5)}$ with $\mathtt{PGM_b}^{(4)}$ (Supplementary Fig.1(d)), thereby confirming that $\mathtt{PGM_b}^{(i+1)} - \mathtt{PGM_b}^{(i)} > 0$ in all the cases.

Other statistical metrics

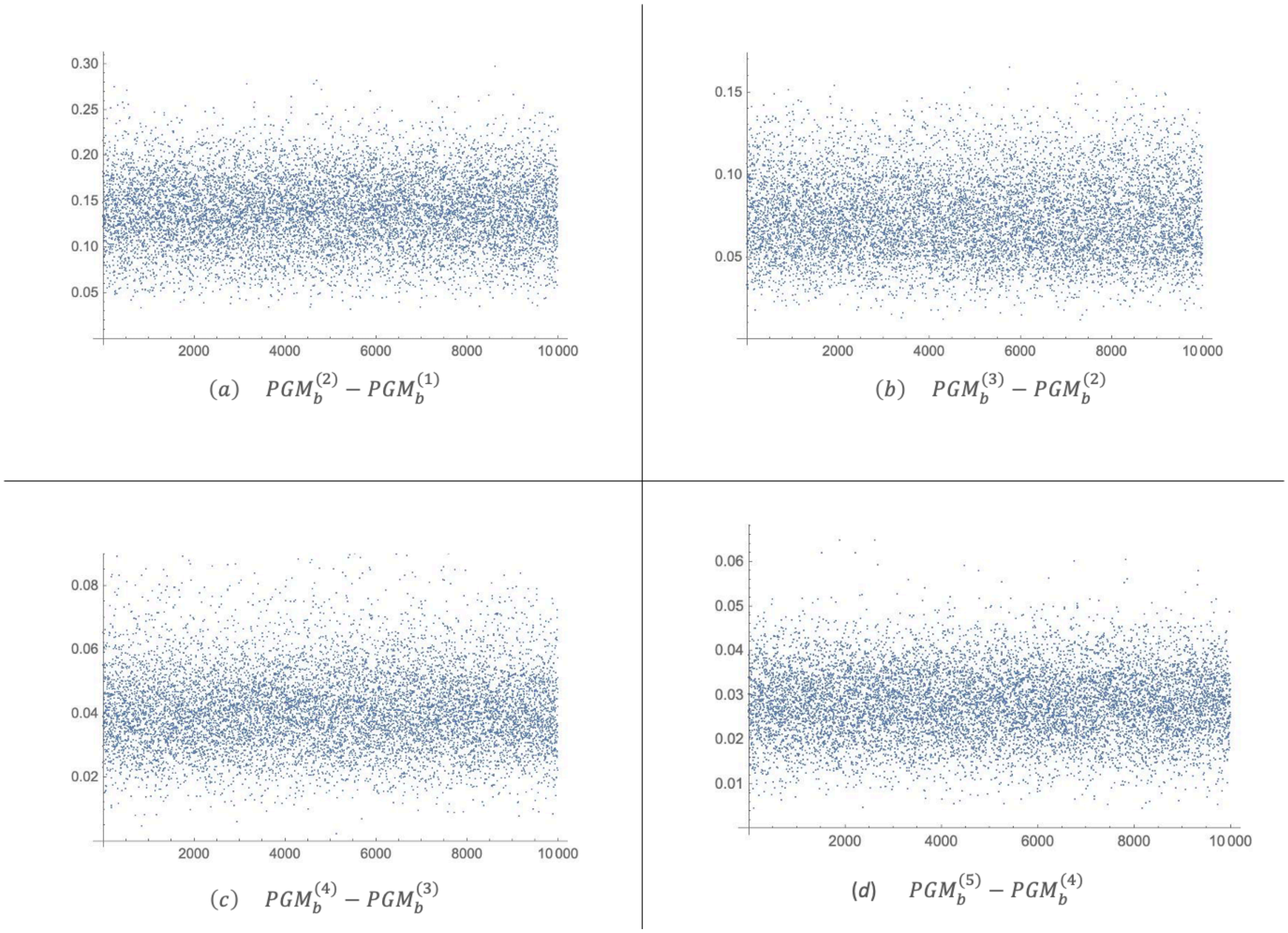
The main manuscript focuses on reporting the metric, balanced accuracy, while comparing the performances of different classifiers. In this section, other metrics, such as weighted f1, weighted precision, and weighted recall are discussed.

By referring to the standard nomenclature in machine learning, we briefly recall the expressions of the statistical metrics cited above. The metric, balanced accuracy is defined as $\frac{TPR+TNR}{2}$ (where TPR is the true positive rate of a classification, TNR is the true negative rate, FPR is the false positive rate, and FNR is the false negative rate). The metric, weighted f1 is defined as $\frac{2TP}{2TP+FP+FN}$ (where TP, FP, and FN denote the number of true-positive, false-positive, and false-negative occurrences, respectively). Furthermore, weighted precision is defined as $\frac{TP}{TP+FP}$, and weighted recall as $\frac{TP}{TP+FN}$. For each statistical metric, we present the following data.

1. A heatmap of the values of the statistical metric after the application of all the classifiers for each dataset

2. A biclustering heatmap, wherein we indicate the average winning percentage of each classifier against the other, when calculated over all the datasets

3. The rank of the performances of the classifiers obtained by the data given in item 1



$(a)\quad PGM_b^{(2)} - PGM_b^{(1)}$

$(b)\quad PGM_b^{(3)} - PGM_b^{(2)}$

$(c)\quad PGM_b^{(4)} - PGM_b^{(3)}$

$(d)\quad PGM_b^{(5)} - PGM_b^{(4)}$

SUPPLEMENTARY FIG. 1. Three random datasets under consideration. Each dataset contains two objects, where each object contains two features. We calculated the different values of $\texttt{PGM}_\texttt{b}$ with number of copies as 1 (no copies), 2, 3, 4, and 5. Finally, we calculated the difference between the values of $\texttt{PGM}_\texttt{b}$ for i and $i+1$ number of copies. We performed the same calculation for 10^4 different random datasets. The plots reveal that, for all the $4*10^4$ cases, $\texttt{PGM}_\texttt{b}^{(\texttt{i+1})} - \texttt{PGM}_\texttt{b}^{\texttt{i}} > \texttt{0}$, i.e., the value of $\texttt{PGM}_\texttt{b}$ increases by increasing the number of copies.

4. The rank of the performances of the classifiers obtained by the data given in item 3.

The following classifiers were considered: PGM classifier, HQC OvO, HQC OvR, linear discriminant analysis, quadratic discriminant analysis, dummy classifier (with strategy "prior"), nearest neighbors, nearest centroid, logistic regression, multi layer perceptron, Bernoulli naive Bayes, Gaussian naive Bayes, random forest, and extra tree.

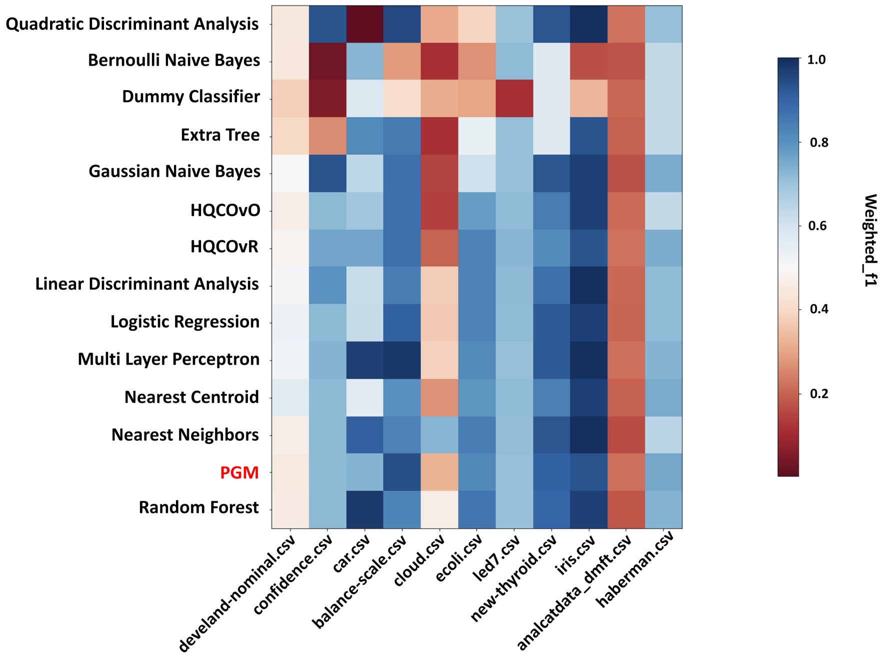
A comparison with the existing classifiers was performed using the scikit-learn package. For the QI classifiers (i.e., PGM, HQC OvO, and HQC OvR), the hypertuning parameters were the number of copies (n) ranging from 1 to 3, rescaling factor was from 0.1 to 3 with step 0.1, and the encoding method was either stereographic or amplitude. For all datasets, setting $n = 3$ yielded the highest classification accuracy. Owing to the increase in runtime, n was not increased beyond 3.

Note that the full data that contain the values of all statistical quantities after the application of all classifiers on all datasets and their respective run times are provided in the supplementary file.

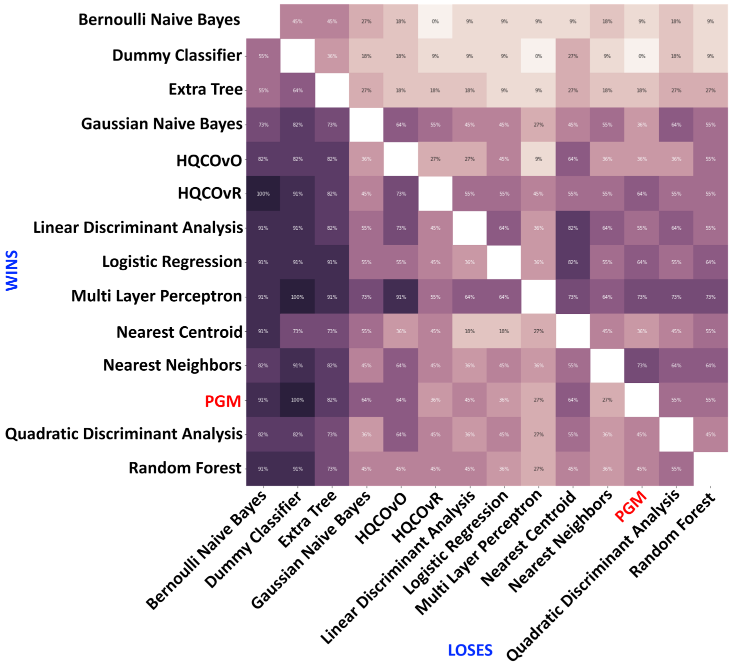
SUPPLEMENTARY TABLE 1. Weighted f1 averaged over all datasets

Rank	Classifier	Average weighted f1
1)	Multi Layer Perceptron	0.726 ± 0.245
2)	Nearest Neighbors	0.724 ± 0.226
3)	Random Forest	0.708 ± 0.236
4)	Logistic Regression	0.683 ± 0.227
5)	PGM	0.683 ± 0.234
6)	Linear Discriminant Analysis	0.681 ± 0.225
7)	HQCOvR	0.669 ± 0.24
8)	Gaussian Naive Bayes	0.656 ± 0.273
9)	Nearest Centroid	0.652 ± 0.227
10)	HQCOvO	0.641 ± 0.252
11)	Quadratic Discriminant Analysis	0.599 ± 0.327
12)	Extra Tree	0.548 ± 0.262
13)	Bernoulli Naive Bayes	0.377 ± 0.241
14)	Dummy Classifier	0.352 ± 0.18

Weighted f1



SUPPLEMENTARY FIG. 2. Weighted f1 for all datasets



SUPPLEMENTARY FIG. 3. One-to-one winning ratio for weighted f1 among all classifiers

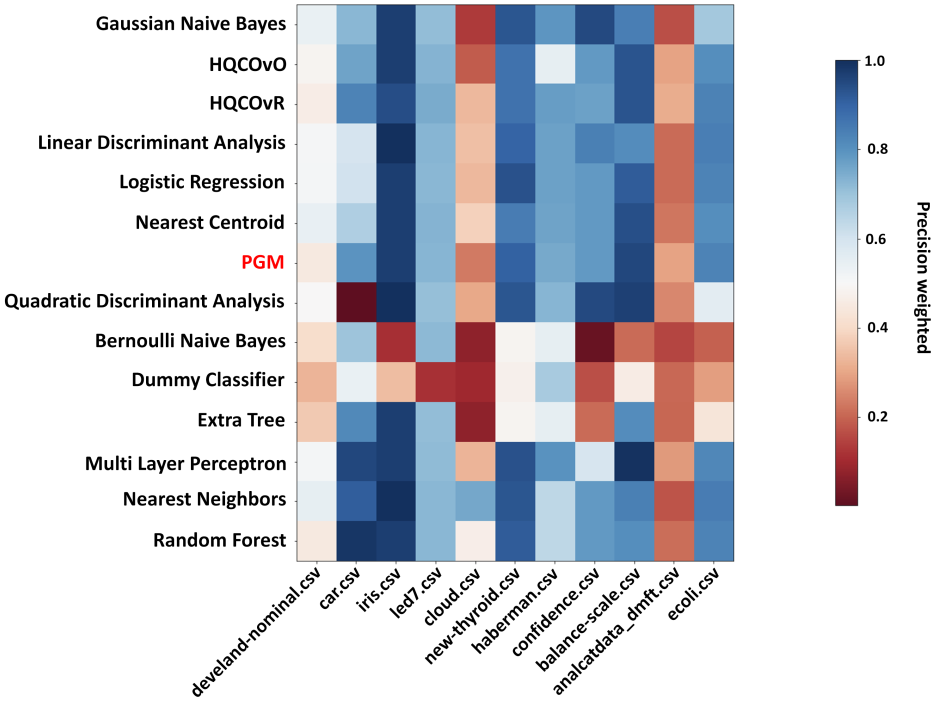
SUPPLEMENTARY TABLE 2. Average winning ratio for weighted f1 (%)

Classifier	**Average winning ratio** (%)
Multi Layer Perceptron	75.16
Linear Discriminant Analysis	65.36
HQCOvR	63.4
Logistic Regression	62.75
Nearest Neighbors	60.13
PGM	58.17
Gaussian Naive Bayes	53.59
Quadratic Discriminant Analysis	52.29
Random Forest	52.29
Nearest Centroid	49.67
HQCOvO	48.37
Extra Tree	24.84
Bernoulli Naive Bayes	16.34
Dummy Classifier	16.34

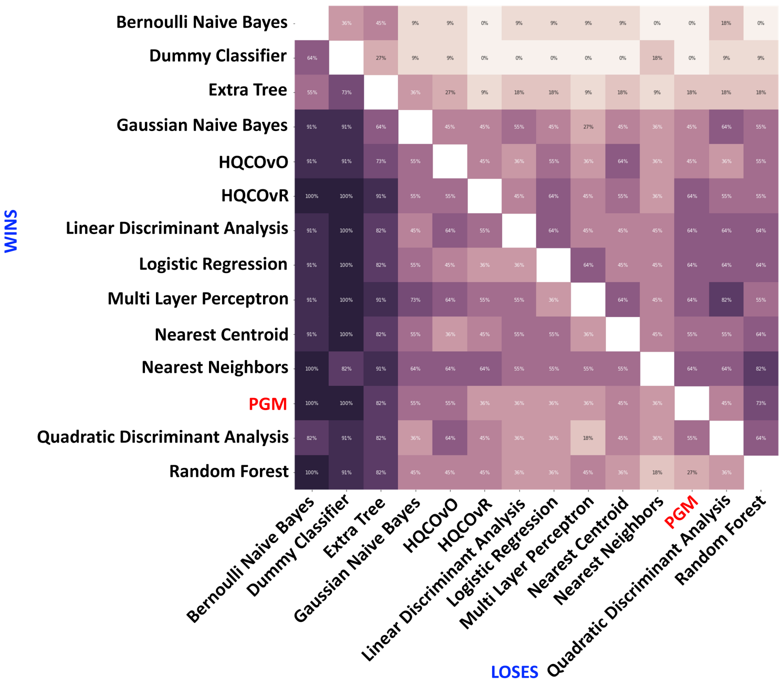
SUPPLEMENTARY TABLE 3. Weighted precision averaged over all datasets

Rank	Classifier	Average weighted precision
1)	Nearest Neighbors	0.741 ± 0.218
2)	Multi Layer Perceptron	0.719 ± 0.246
3)	HQCOvR	0.71 ± 0.219
4)	Random Forest	0.708 ± 0.232
5)	PGM	0.7 ± 0.245
6)	Nearest Centroid	0.696 ± 0.218
7)	Logistic Regression	0.69 ± 0.238
8)	Linear Discriminant Analysis	0.686 ± 0.233
9)	Gaussian Naive Bayes	0.676 ± 0.277
10)	HQCOvO	0.67 ± 0.246
11)	Quadratic Discriminant Analysis	0.627 ± 0.32
12)	Extra Tree	0.513 ± 0.277
13)	Dummy Classifier	0.336 ± 0.177
14)	Bernoulli Naive Bayes	0.329 ± 0.242

Weighted precision



SUPPLEMENTARY FIG. 4. Weighted precision for all datasets



SUPPLEMENTARY FIG. 5. One-to-one winning ratio for weighted precision among all classifiers

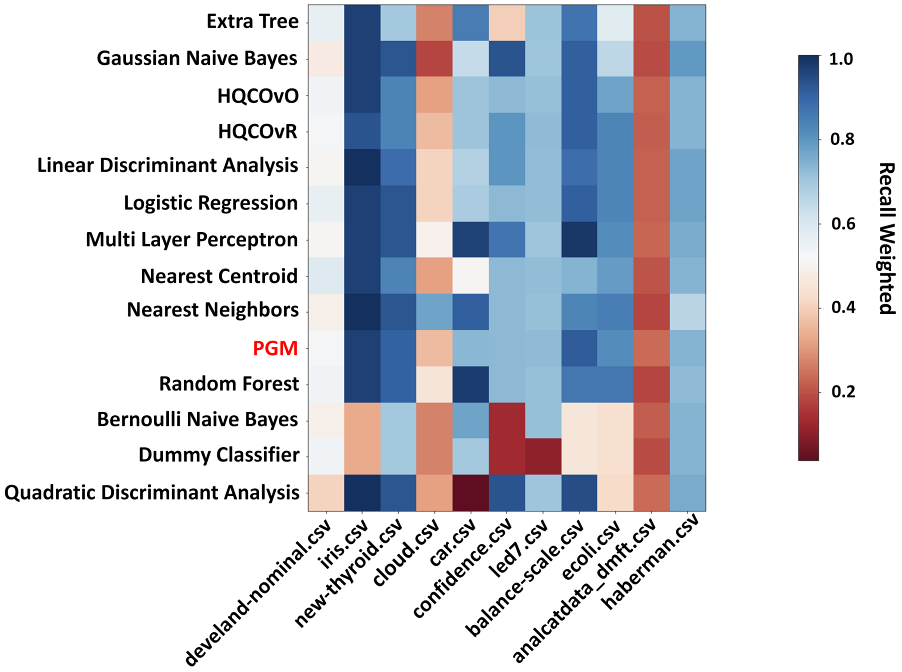
SUPPLEMENTARY TABLE 4. Average winning ratio for weighted precision (%)

Classifier	Average winning ratio (%)
Nearest Neighbors	68.63
Multi Layer Perceptron	67.32
Linear Discriminant Analysis	63.4
HQCOvR	62.75
Logistic Regression	61.44
Nearest Centroid	59.48
PGM	58.82
HQCOvO	55.56
Quadratic Discriminant Analysis	54.9
Gaussian Naive Bayes	52.29
Random Forest	50.98
Extra Tree	24.18
Bernoulli Naive Bayes	11.11
Dummy Classifier	10.46

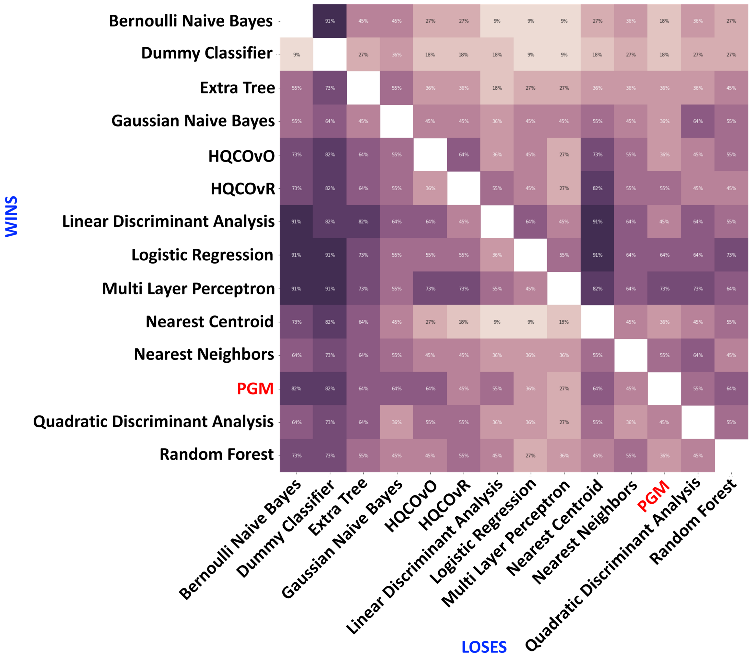
Weighted recall

SUPPLEMENTARY TABLE 5. Weighted recall averaged over all datasets

Rank	**Classifier**	**Average weighted precision**
1)	Multi Layer Perceptron	0.749 ± 0.232
2)	Nearest Neighbors	0.735 ± 0.221
3)	Random Forest	0.721 ± 0.233
4)	Logistic Regression	0.705 ± 0.218
5)	Linear Discriminant Analysis	0.701 ± 0.221
6)	PGM	0.698 ± 0.221
7)	HQCOvR	0.691 ± 0.218
8)	HQCOvO	0.68 ± 0.221
9)	Gaussian Naive Bayes	0.671 ± 0.27
10)	Nearest Centroid	0.651 ± 0.216
11)	Extra Tree	0.622 ± 0.237
12)	Quadratic Discriminant Analysis	0.609 ± 0.32
13)	Bernoulli Naive Bayes	0.48 ± 0.216
14)	Dummy Classifier	0.419 ± 0.221



SUPPLEMENTARY FIG. 6. Weighted recall for all datasets



SUPPLEMENTARY FIG. 7. One-to-one winning ratio for the weighted recall among all classifiers

SUPPLEMENTARY TABLE 6. Average winning ratio for weighted recall (%)

Classifier	Average winning ratio
Multi Layer Perceptron	70.59
Logistic Regression	66.01
Linear Discriminant Analysis	64.71
PGM	56.86
HQCOvR	55.56
HQCOvO	54.9
Nearest Neighbors	52.29
Quadratic Discriminant Analysis	50.33
Random Forest	49.67
Gaussian Naive Bayes	49.02
Nearest Centroid	42.48
Extra Tree	39.22
Bernoulli Naive Bayes	29.41
Dummy Classifier	20.92