Soft Computing (2020) 24:10247–10255
https://doi.org/10.1007/s00500-019-04429-x

FOCUS

Quantum and quantum-like machine learning: a note on differences and similarities

Giuseppe Sergioli[1]

Published online: 24 October 2019

Abstract
In the past few decades, researchers have extensively investigated the applications of *quantum computation* and *quantum information* to machine learning with remarkable results. This, in turn, has led to the emergence of *quantum machine learning* as a separate discipline, whose main goal is to transform standard machine learning algorithms into quantum algorithms which can be implemented on quantum computers. One further research programme has involved using quantum information to create new quantum-like algorithms for classical computers (Sergioli et al. in Int J Theor Phys 56(12):3880–3888, 2017; PLoS ONE 14:e0216224, 2019. https://doi.org/10.1371/journal.pone.0216224; Int J Quantum Inf 16(8):1840011, 2018a; Soft Comput 22(3):691–705, 2018b). This brief survey summarises and compares both approaches and also outlines the main motivations behind them.

Keywords Quantum machine learning · Quantum information · Binary classification

1 Introduction

The current exponential growth of data in a wide variety of contexts has led scientists and researchers to deal with very large *datasets* from which to extract all potentially useful information. But in what cases, and how, can one confidently assert that a specific dataset containing a certain (large) amount of data provides more useful information than another one containing fewer data? In general, one cannot provide answers to these questions, as, unfortunately, the size of a dataset does not generally match the size of the (useful) information which can be extracted from it. What one may only guess is that a big dataset 'potentially' contains much information, but the process of extracting it is not easy, anyway.

Generally speaking, the process of extraction comes in two steps: the first step consists in choosing the appropriate data which are deemed to be useful for retrieving the information one wants to extract, whereas the second step consists in analysing such data as accurately as possible. It is not difficult to see that executing both operations 'manually' would require a huge amount of time. (It is absolutely not manipulable in the case of a consistent number of data.) Therefore, creating languages to train machines to extract information, initially from a modest amount of data, and then from datasets of any size, is seen as indispensable. This is precisely what has motivated the rise of *machine learning*, which, since its inception around the 1950s, has been implemented using different methods and strategies (Duda et al. 2000).

In the 1970s, Feynamn (1982) showed that a Turing machine capable of simulating some particular physical process cannot exist without incurring an exponential slowdown of its performances. Finally, in 1985, Deutsch (1985) created the first formal model of a quantum Turing machine. The theory of quantum computing (Nielsen and Chuang 2010) has since become an autonomous discipline, which has, over the years, received increasing attention from the scientific community.

The advantages of quantum computation over classical computation are well known and widely discussed (Nielsen and Chuang 2010), but the difficulties that scientists have encountered during the physical realisation of quantum computers (mostly related to the problem of decoherence) are also well known. However, in recent years, huge progress has been made in the direction of producing quantum computers (Castelvecchi 2017), and the prospect of using quantum

Communicated by F. Holik.

Giuseppe Sergioli
giuseppe.sergioli@gmail.com

1 University of Cagliari, Via Is Mirrions 1, 09123 Cagliari, Italy

computers in the near future for running everyday calculations is not far fetched. Arguably, the most evident advantage of using a quantum computer instead of a classical one is the resulting speed up of computations. As is well known, many algorithms, which require exponential or high-degree polynomial time to be run (or are unsolvable) on classical computers, can be solved on quantum computers with a drastic reduction in the time complexity.

But let us go back to machine learning, for a moment. Over the past decades, the sizes of the datasets which scientists have had to deal with have progressively increased, while the techniques used for extracting information have been underperforming, mostly because of the large amount of time required to manipulate huge quantities of data. It is precisely then that 'big data science' has emerged as a response to this issue (Hilbert and Lopez 2011), at a time when, moreover, the theory of quantum computation was already showing that the performances of quantum computers can be much faster those of classical ones. Therefore, in the context of machine learning, the merging of big data science and quantum computation really stands out as the most suitable and natural approach to the problems posed by the manipulation of huge quantities of data.

In turn, such a merging might be carried out, fundamentally, in two ways:

(1) by running standard machine learning algorithms on quantum computers. In order to do this, it is necessary to translate classical algorithms into the language of quantum computation (quantum circuits) and afterwards physically implement the quantum circuits on a real quantum computer. This procedure is precisely what is usually called *quantum machine learning* (QML, Qiu 2007; Lloyd et al. 2013; Schuld and Petruccione 2018; Schuld et al. 2014a; Wittek 2014).

(2) by creating new algorithms which are not quantum-computational translations of classical algorithms, but are, rather, inspired by the very principles of quantum computation and quantum information (e.g. parallelism, entanglement, entropy, etc.). It should be noted that, although inspired by quantum theory, these algorithms are still coded in the 'classical' language and could also be run on classical computers. We refer to this second strategy as *quantum-like machine learning* or, sometimes, *quantum-inspired machine learning* (QiML) (Manju and Nigam 2014; Santucci 2017; Santucci and Sergioli 2018; Sergioli et al. 2017, 2018b).

The purpose of this paper is to briefly discuss and compare QML and QiML, by pointing to fundamental similarities and differences between the two.

2 Machine learning

Machine learning can be defined as that branch of artificial intelligence which deals with methods for isolating specific properties of a given dataset. The idea of creating algorithms capable of learning by experience was first conceived of by several scientists (such as Turing, Samuel, Rosenblatt, and Minsky) at the same time in the 1950s (Wittek 2014) and, quickly, several approaches and strategies emerged, which cannot be exhaustively summarised here. However, it is possible to identify two main different approaches to machine learning:

- *Supervised Machine Learning*
  In this approach, one deals with sets of 'labelled objects', which means that particular properties of each object (such as *to be a cat* or *to be a red point*) in a dataset are assigned *labels*. Moreover, objects are uniquely defined by (some of their) features (such as *the length of the tail* or *the position in a plane*). Very often, such features are of numerical nature and, in this case, each object is simply represented by a labelled vector. The aim of the supervised approach is to process features of objects and provide efficient algorithms able to identify all and only those objects that possess a given—pre-established—property.
- *Unsupervised Machine Learning*
  In the unsupervised approach, objects in the dataset are not labelled. The aim of the unsupervised approach is to identify the common 'structure' shared by all objects.

As is clear, the two approaches, respectively, generate two different kinds of outputs: while the supervised approach aims to isolate *algorithms*, the unsupervised approach aims to isolate *structures*. Frequently, though, the two approaches are used in combination to extract more information from specific datasets. This counts as a third approach, which is commonly referred to as *semi-supervised machine learning*, and one of the goal of this last approach is precisely that of searching for the best strategy to merge the supervised and the unsupervised approach to extract useful information.

Several techniques for the supervised approach (*Artificial Neural Network*, *Bayesian Methods*, *Kernel Method*, *K-Nearest Neighbours*, *Least-Squares Formulation*, *Instances Based*, *Regression Based*, *Tree Based Method*, etc.) and for the unsupervised approach (*Dimensionality Reduction*, *K-Means Clustering*, *Hierarchical Clustering*, etc.) have been developed, and a detailed description of these (which may be found in Duda et al. 2000) is beside the scope of this work. As we will see in the next section, the specific aim of the quantum-theoretic approach to machine learning is to find quantum algorithms for the *supervised* approach; therefore, from now on, we will mostly focus on the supervised approach.

2.1 Classification

As an example of a specific application of the supervised approach, we illustrate the following classification process (Duda et al. 2000).

Suppose we have different sets (sometimes called *classes*) of objects which share some property (for instance, '*to be a dog*'); the aim of a supervised classification process is to obtain an optimal algorithm able to analyse the features of each object in the dataset so as to be able to establish what set the object belongs to (in other terms, so as to be able to say for instance that '*object x is a dog*').

For the sake of simplicity (and without loss of generality), let us consider two sets of objects S_1 and S_2 (e.g. 'cats' and 'dogs', respectively). Each object is described by picking some of its pre-established features which, in this case, we take to be of numerical nature. For instance, each object (*pattern*) can be described as a labelled vector, where the label represents the set (the class) the object belongs to, and each component of the vector is expressed by the numerical value of some feature of the object. Formally, a pattern X can be denoted as: $X = (\mathbf{x}, l)$, where $\mathbf{x}$ is the vector $\mathbf{x} = (x_1, \dots, x_n)$ (x_i denotes the ith feature of $\mathbf{x}$) and l is the label that denotes the class the objects belongs to. As an example, take the very simple case of a binary classification (i.e. the case when the number of different classes is equal to two): the dataset is a set of patterns $D = \{X_i\}$ such that the label l can just take two values, say $l \in \{+, -\}$. We can then denote the classes of patterns with a positive or negative label as C_+ and C_-, respectively. Obviously, C_+ and C_- are different partitions of D. Following the mentioned strategy, let us consider the set D randomly partitioned into two subsets: D_{Tr} and D_{Ts}; the first subset is called *training dataset* and the second *test dataset*. (Generally, the cardinality of D_{Tr} amounts to the 80% of the cardinality of the original dataset D.) The D_{Tr} dataset is, then, used to define what is called a classifier, i.e. a function which assigns, with the highest possible level of accuracy, a unique label to each vector $\mathbf{x}$. Once the classifier has been defined, then it is possible to proceed to apply it to each vector of the test dataset in order to measure its level of accuracy. The ratio between the number of vectors of D_{Ts} which have been correctly classified and the cardinality of D_{Ts} is a measure of the *accuracy* of the classifier. Obviously, the successfulness of this process is strongly dependent on the lucky/unlucky choice of the partition of D into the training and the test set, and this motivates the additional use of a *statistical method*. In simple terms, this consists in repeating the same procedure described above several times, each time making a different random choice of the training and the test datasets. At the end of the process, it is possible to obtain an average value of the accuracy of the classifier defined over all outcomes of all different runnings of the classifier itself.

However, accuracy is not the only statistical parameter useful to evaluate the performance of a classifier. All other relevant parameters (such as *true/false positive/negative rates*, *balanced accuracy*, *sensitivity*, *specificity*, and *F-measure*) are summarised in the well-known confusion matrix (Duda et al. 2000) and should all be seen as integral parts of the standard analysis of a classification context. Therefore, the bulk of the strategy of the supervised approach with regard to classification problems consists in finding the best-performing classifier, that is, a classifier which makes the fewest possible mistakes.

It should be noted that machine learning is, due to its own nature, a very empirical kind of study. In particular, as shown by the celebrated 'no free lunch theorem' (Duda et al. 2000), we should be aware of the fact that there is no classifier which outperforms all other classifiers in all possible cases, that is, independently of the nature of the datasets it is applied to and, in fact, the only way to measure the efficiency of a newly found classifier, and compare it to that of other classifiers, involves setting up a large-scale experiment, wherein one may compare the performance of the new classifier to that of many other classifiers by applying all of them to as many as possible different datasets.

For instance, a very intuitive (arguably, the simplest) classifier within machine learning is the *Nearest Mean Classifier* (NMC), which can be described as follows.

Let us consider the case of a binary classification, and let us choose a partition of D_{Tr} into two subsets, $D_{\mathrm{Tr}_+} = D_{\mathrm{Tr}} \cap C_+$ and $D_{\mathrm{Tr}_-} = D_{\mathrm{Tr}} \cap C_-$ (i.e. the patterns of the training set with positive and negative labels, respectively). Now, implementing NMC consists in finding the M_+ and M_- centroids of the vectors (expressed by n-coordinates points) belonging to, respectively, D_{Tr_+} and D_{Tr_-}. Formally, the centroids are obtained by using the standard n-dimensional Euclidean distance. The line between these two points is the geometrical representation of the classifier, and all vectors closer to M_+ (i.e. all the points on one side of the line) will be labelled by $l = +$, whereas all vectors closer to M_- (i.e. all the points located on the opposite side of the line) will be labelled by $l = -$. Now, when this classifier is applied to D_{Ts}, obviously some vectors (points) will come out as correctly classified, and some others as incorrectly classified (see Fig. 1).

This procedure ultimately allows us to obtain all the quantities included in the confusion matrix, and after repeating the same experiment by making alternative, random choices of D_{Tr} and D_{Ts}, we will also be able to carry out the required statistical analysis.

The NMC is very simple and intuitive, but many more (and, in general, even better performing) classifiers, such as *Linear Discriminant Analysis*, *Quadratic Discriminant Anal-*

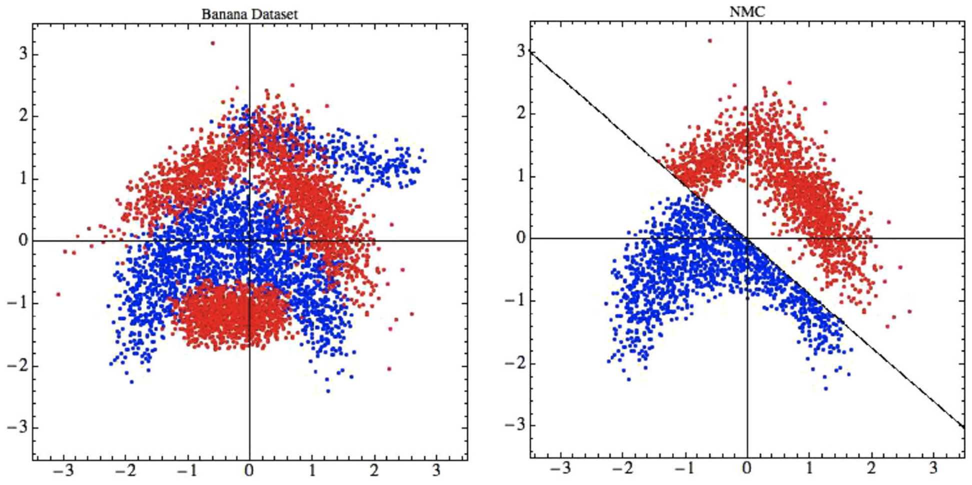


Fig. 1 The original dataset (left) and the dataset containing only the objects correctly classified by using the NMC (right)

ysis, *Gradient Boosting Classifier*, *K-Neighbours Classifier*, *Random Forest Classifier*, and *Logistic Regression*, are also available. In fact, on the Web, one can easily find large repositories of datasets and (already implemented) classifiers (the *GitHub* and *Weka* applicatives being two relevant examples).

At this stage, and in view of the purposes in the next section, it is important to stress that most standard classifiers are distance-based classifiers, that is, classifiers which involve using a notion of *distance*. For instance, the NMC uses the Euclidean distance, but other definitions of distance, such as the *squared distance*, the *cosine distance*, and the *Hamming distance*, may also be used.

As a final remark, it should also be noted that, already at the level of binary classification, several, alternative methods for addressing multi-class classification are available. For a state-of-the-art survey of all these methods, see Duda et al. (2000).

3 Quantum machine learning

Searching for meaningful interactions among different disciplines is always a fascinating, but also potentially risky, undertaking not always crowned by success. However, several external applications of quantum theory are viewed today as being very useful, and we watch 'quantum approaches' making their way through the most various contexts and disciplines, such as psychology, economics, and music (Dalla Chiara et al. 2015; Freytes and Sergioli 2014; Melkikh et al. 2019). Now, trying to merge quantum information theory and machine learning seems to be an especially safe, and, in addition, promising, task for a very simple reason: both disciplines deal with information.

As briefly argued in the Introduction, quantum information and machine learning were independently developed during the second part of the last century. The continuous advance in technology has given rise to the practical necessity of handling an ever-increasing amount of data, which include, just to make some down-to-earth examples, the photographs we keep in our PC, the messages we store in our smartphone, etc. Extracting useful information from such a huge collection of data has really become a daunting task, which, as is clear, requires a huge amount of time to be processed.

Now, the natural question arises: may quantum computers help speed up the processing of all such data by exploiting the well-known principles of quantum computation and quantum information theory (such as parallel calculations and others)? As argued before, the need to answer this very natural question has recently led to the emergence of quantum machine learning as a new, autonomous discipline. The term 'quantum machine learning' (QML) was first used by Lloyd et al. (2013); since then, the interest of the scientific community in this area of research has enormously increased, as shown by the rapid growth in the number of scientific publications devoted to it (Lloyd et al. 2014; Lu and Braunstein 2014; Manju and Nigam 2014; Schuld et al. 2014a, b; Sergioli et al. 2017; Wiebe et al. 2015).

Researchers' interest in this discipline is not only motivated by the fact that QML is able to successfully combine two areas which seem to be far apart; in recent years, also advances in the creation of the first quantum computer (Castelvecchi 2017) have increased the chances of implementing quantum machine learning processes on 'real' quantum computers. [Some toy experiment going in this direction has already been carried out (Schuld and Petruccione 2018)].

Now, what has emerged is that there is not a unique correct way to merge quantum computing and machine learning: Aimeur, Brassard, and Gambs at first (Aïmeur et al. 2006), and Schuld and Petruccione (2018) later, have described four different approaches, whose features we now proceed to review.

The first one is the Classical–Classical approach, whereby quantum-like classical algorithms are implemented on classical computers. This approach is interesting not only as a theory, but has also been shown to impact on actual *computational* processes. More on this approach will be discussed in the next section.

A second approach is the Quantum–Classical one. It consists in employing standard machine learning methods within a quantum-computational framework. In simple terms, this approach aims to find optimal procedures to allow quantum computers to learn from data (Aaronson 2007; Bisio et al. 2010; Carrasquilla and Melko 2017; Sasaki and Carlini 2002).

Probably, the most explored approach, so far, has been the Classical–Quantum approach, which is the one rightfully referred to as QML. This approach features the *translation* of classical machine learning processes into the language of quantum computation. In other words, QML is designed to replace classical machine learning algorithms with quantum ones (to be run on quantum computers). As argued in Schuld and Petruccione (2018), two alternative strategies have been pursued by QML developers: (i) translating standard machine learning algorithms into quantum-computational algorithms (Lloyd et al. 2013, 2014); (ii) using existing quantum-theoretic algorithms to solve problems related to machine learning (Schuld et al. 2014a, b).

Now, the main rationale behind the use and implementation of QML is to have quantum computers manipulate classical data. For this, the crucial preliminary steps are: (i) data pre-processing and (ii) encoding of classical data into quantum data. This second step, in particular, needs to be addressed in more detail.

Several ways to encode classical data (e.g. *basis encoding*, *amplitude encoding*, *Qsample encoding*, *dynamic encoding*, *Hamiltonian encoding*, see Schuld and Petruccione 2018) to quantum states are discussed in the literature, and each of these has been used in a specific applicative context. Again, it is not possible, in general, to establish which of these encodings performs best, as performances are strongly dependent on the chosen dataset, on the process involved, and other factors.

For example, in Schuld and Petruccione (2018), the authors show how to implement a distance classifier in a supervised scenario by using a quantum computer. The procedure involves normalising all data through a preliminary pre-processing and, afterwards, encoding classical data to qubits. In this specific case, the authors use the so-called *amplitude encoding*. (For a detailed description of this, see Schuld et al. 2017.) In order to translate the square distance between two vectors in a quantum-computational setting, the authors use a Hadamard gate and, then, proceed to perform the required measurement.

This is just one example, which, however, already shows how to translate a standard machine learning process into the language of quantum computing. In recent years, several sophisticated techniques have been developed which allow quantum computers to solve lots of different ‘classical’ machine learning processes (Gambs 2008; Trugenberg 2002; Wiebe et al. 2015; Wittek 2014). The advantages of using this methodology have already come to the fore, but the most remarkable results in this area are probably yet to come.

As for the fourth, and last, approach, the Quantum–Quantum approach, this features the use of quantum algorithms to manipulate quantum, not classical, data. Therefore, in this approach, data need not be encoded, as the goal of the process is precisely to address purely quantum-mechanical phenomena. Only very tentative results have been attained so far (Audenaert et al. 2017; Bergou et al. 2004; Chefles 2000; Guta and Kotlowski 2010; Hayashi et al. 2005; Qiu 2007), but also this approach, overall, seems to be very promising.

4 Quantum-like machine learning

In the previous section, we have briefly mentioned the Classical–Classical approach as one of the four main quantum-theoretic approaches to machine learning. In this approach, one deals with classical objects and also uses classical computers. So, precisely what is the role of quantum mechanics in it? In what sense does quantum mechanics play an ‘inspirational’ role for it? Finally, what does one gain by following this approach?

These questions have recently been addressed by different authors, who have formulated several ‘quantum-like’ (or ‘quantum-inspired’) methods for computational intelligence. (For an exhaustive survey of these methods, see Manju and Nigam 2014.) These methods are generally motivated by the need to deal with problems which have turned out to be intractable in standard machine learning (Santucci 2017; Santucci and Sergioli 2018; Sergioli et al. 2017, 2019, 2018b).

In very general terms, the Classical–Classical approach (henceforth, just QiML) features the following four steps: (i) data pre-processing; (ii) encoding; (iii) creation of quantum-inspired algorithms; and (iv) decoding. In Sergioli et al. (2019), the authors have proposed a new approach to QiML (referred to as NQiML), which is meant to apply to a very general kind of binary classification problems, and in what follows we will just be focussing on NQiML.[1]

[1] It should be stressed that quantum-like algorithms implemented by the NQiML approach are not a mere translation of classical algorithms. For instance, in Sergioli et al. (2018b, Section 2.1), the author introduces a quantum-like version of the NMC, which is very different from a mere quantum-theoretic translation of the NMC.

Let us now describe it in a little more detail. Later, in the second part of this section, we will compare the QML with the NQiML approach.

As far as step (i) above is concerned, let us consider the two sets of patterns $X_+ \in D_{\mathrm{Tr}_+}$ and $X_- \in D_{\mathrm{Tr}_-}$, where $X_i = (x_i, l_i)$. After carrying out a suitable pre-processing, which is strongly dependent on the cardinality of the dataset,[2] in NQiML, one proceeds to encode real vectors to quantum states [step (ii)]. Generally, in NQiML, the encoding is a map $e\colon \mathbb{R}^n \mapsto \otimes^{(n+1)}\mathcal{D}$. As is clear, e maps each n-dimensional vector v into an $(n+1)$-dimensional pure density operator ρ_v (which is called *density pattern*). Let $\mathcal{D}_+$ and $\mathcal{D}_-$ be the sets of pure density operators originating from, respectively, X_+ and X_-. For each class ($+$ or $-$), a *quantum centroid* is defined as $\tilde{\rho}_+ = \frac{n}{n_+}\sum_{i=1}^{n_+}(\rho_+)_i$ and $\tilde{\rho}_- = \frac{n}{n_-}\sum_{i=1}^{n_-}(\rho_-)_i$, where $(\rho_+)_i \in \mathcal{D}_+$ and $(\rho_-)_i \in \mathcal{D}_-$.

We now proceed to describe step (iii). For this, several 'quantum-inspired' algorithms may be formulated. These, in turn, as already seen in the cases described in the previous section, will employ the notion of a distance between the *quantum centroids* and the *density patterns* belonging to the test set. The choice of a notion of distance is made using quantum information: for example, the trace distance and Helstrom's metric are frequently used to this end (Helstrom 1976).

Finally, after applying the classification algorithm, all the results are decoded (step (iv)) so as to be rendered as real patterns. The experiment is, then, repeated several times, by picking up random training and test datasets, and by following the standard procedure as described in Sect. 2. Now, the main reason why the resulting algorithm is a real *quantum-like* algorithm is the fact that the higher the *degree* of distinguishability among the quantum centroids, the more accurate the chosen classifier.

In some cases, the NQiML approach also employs two additional pre-processings of the data: *rescaling* and *copy*. Rescaling consists in multiplying all the components (features) of each vector by a constant (real) factor, which results in a complete modification of the formulation of the quantum centroids (notice that the quantum centroid of the quantum density patterns obtained by rescaling the original dataset by a constant parameter does not correspond to the quantum pattern obtained by rescaling the centroid of the original dataset) which considerably (and positively) affects the whole classification process.

Copy works as follows: after one has carried out the process of encoding vectors to density operators, and before applying the classification process, one can produce one or more *tensor copies* of each quantum pattern. In other words, through using copy, each density pattern ρ obtained by the encoding is replaced by $\rho \otimes \rho \otimes \cdots \otimes \rho$. A suitable use of both pre-processing procedures can further enhance the level of accuracy of the classification process.

In conclusion, it should be noted that, unlike the standard Classical–Classical approach, and because of its very natural and flexible internal structure, the NQiML approach is, in principle, applicable to any kind of classification problem.

4.1 A comparison

As discussed in Sect. 3, the main goal of QML is to replace classical machine learning processes with quantum-computational ones and implement the latter on quantum computers, thus obtaining a considerable speed up of the computation process.

QiML and, in particular, NQiML seem to adopt a classical approach but, in fact, are also very quantum-like in their essence, even though they are run on classical computers. We now proceed to compare features of QML with those of NQiML in more detail.

4.1.1 The quantum centroid

A few lines above, we claimed that, both in the QML and in the NQiML approaches, a fundamental initial step consists in encoding *classical* (formal) objects (i.e. patterns or, simply, vectors) into *quantum* (formal) objects (vectors in the Hilbert space or density operators). In NQiML, each vector is encoded to a density operator (the aforementioned density pattern) which is always, and by construction, a pure quantum state. By the standard formulation of quantum mechanics, pure states are those states which convey maximal information; for this reason, when we have to encode a real object all of whose components (features) are known, the choice of encoding it through a pure state turns out to be quite natural. However, if one looks at the mathematical formulation of the quantum centroid provided above, it is easy to see that, in general, the latter is no longer a pure state. Therefore, the quantum centroid is not the counterpart of any real object, which means that it is not the quantum-mechanical translation of any (originally) classical object, and that its positing makes sense only within a pure quantum-theoretic scenario. Proof of this is the fact that the density pattern obtained by encoding the classical centroid is totally different from the quantum centroid.

4.1.2 The notion of distance

We have seen that several machine learning methodologies adopt a suitable notion of distance between vectors. The standard QML approach is based on the use of quantum gates which simulate the behaviour of 'classical' distances. Now, NQiML differs from QML also with regard to this. In

[2] For instance, a very usual pre-processing consists in the normalisation of all the vectors of the dataset.

NQiML, one uses the trace distance and Helstrom's metric, which are standard quantum information metrics which may not be viewed as quantum translations of any classical metric; indeed, they make sense only within a quantum scenario (Helstrom 1976).

4.1.3 The advantages of, respectively, QML and NQiML over classical machine learning

We have already made it clear that QML runs, in principle, on quantum computers, which has the natural advantage of reducing the time complexity of a computation process. On the other hand, NQiML runs on classical computers. Hence, the crucial question is: what sort of advantages does the use of each of the two, separately, have? The answer to this question highlights another remarkable difference between QML and NQiML: QML aims to translate standard algorithms into quantum-computational algorithms, so it is more advantageous in terms of computational complexity, not in terms of accuracy.

NQiML behaves very differently: as shown and discussed in some recent articles (Holik et al. 2017; Santucci 2017; Santucci and Sergioli 2018; Sergioli et al. 2016, 2017), the expressive power of quantum formalism applied to machine learning positively impacts on the accuracy of the classification process. In particular, in Sergioli et al. (2019), as a result of a large-scale experiment, we have shown that a quantum-inspired classification algorithm based on a Helstrom's metric is able to outperform, on average, all the most used standard classifiers in terms of accuracy. It must also be stressed that, unlike QML, NQiML is not a mere translation of classical machine learning algorithms, and this is precisely why it can be much more accurate than QML. We think it is important to emphasise this fact, mostly in view of the potential applications of quantum machine learning, since, in some cases, scientists will prefer a higher accuracy to a lower time complexity. This is, for instance, the case of biomedical research, wherein a variant of NQiML has been already applied with promising results (Sergioli et al. 2018a).

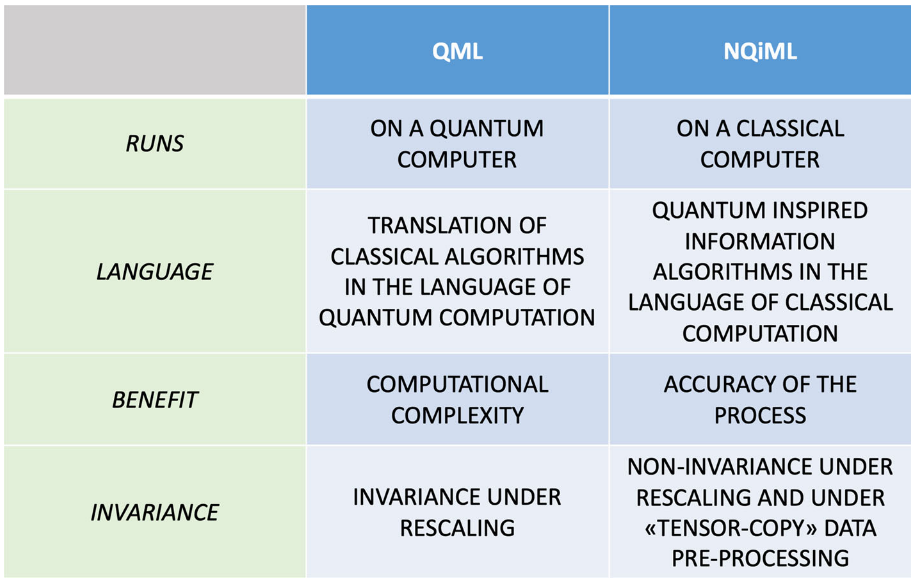
On the other hand, insofar as it runs on classical computers, in general, NQiML does not bring any benefit in terms of time complexity and, on the contrary, in some cases, depending on the complexity of quantum formalism (which also involves complex numbers, tensor products, etc.), time complexity is even increased by its use.

4.1.4 Invariance under rescaling

In all most usual scenarios for machine learning, a crucial role is played by the data pre-processing stage. The 'non-invariance under rescaling' feature discussed above further exemplifies the way QML and NQiML differ from each other. As is clear, in QML, rescaling does not affect the accuracy or the time complexity of the whole classification process. On the contrary, in NQiML, the mathematical formulation of the quantum centroid is not invariant under rescaling, and this clearly affects the classification process. This fact is used as an asset in the NQiML context: an empirical investigation of the optimal rescaling factor can help further improve the accuracy of the classification process. In Sergioli et al. (2017), we showed that a different choice of rescaling factors can further increase the accuracy of the classifier.

4.1.5 Copy

We have seen above that copy involves making one or more tensor copies of each quantum pattern before carrying out the classification process. As shown in Sergioli et al. (2019), this



	QML	NQiML
RUNS	ON A QUANTUM COMPUTER	ON A CLASSICAL COMPUTER
LANGUAGE	TRANSLATION OF CLASSICAL ALGORITHMS IN THE LANGUAGE OF QUANTUM COMPUTATION	QUANTUM INSPIRED INFORMATION ALGORITHMS IN THE LANGUAGE OF CLASSICAL COMPUTATION
BENEFIT	COMPUTATIONAL COMPLEXITY	ACCURACY OF THE PROCESS
INVARIANCE	INVARIANCE UNDER RESCALING	NON-INVARIANCE UNDER RESCALING AND UNDER «TENSOR-COPY» DATA PRE-PROCESSING

Fig. 2 Comparison between QML and NQiML

procedure can also bring substantial benefits to the accuracy of the classification process. Moreover, this is a unique feature of quantum machine learning, as, in standard machine learning, making copies of a state would not provide us with additional information.

We summarise the main differences between QML and NQiML in Fig. 2.

5 Conclusions

In this paper, we have briefly surveyed the rationale behind the merging of two different disciplines, that is, *quantum computation and information*, on the one hand, and *machine learning*, on the other, and the different ways in which such a merging may be carried out. In particular, we have discussed and compared the QML approach with the QiML approach (in fact, a more recent version of QiML, called NQiML).

We have shown that the QML approach mostly consists in translating standard algorithms in the language of quantum computation. QML is less accurate than NQiML, but is preferable in terms of computational complexity. On the other hand, the main advantage of working with NQiML is that the latter can be very easily implemented on classical computers. Moreover, in some cases, NQiML may also be able to bear significantly on the accuracy of a classification process.

As a promising further development, one could think of merging the QML and NQiML approaches: for one thing, such a merging would allow us to translate what is just a 'quantum-like' classification process to an actual quantum-computational process, and, in addition, it would also provide us with all advantages that the two approaches have over classical machine learning, that is, a relevant increase in both the time complexity and the accuracy during the same computational process.

Acknowledgements I warmly thank Claudio Ternullo for the careful linguistic revision of the last version of the manuscript.

Funding This work has been partially supported by the Project "Strategies and Technologies for Scientific Education and Dissemination" (CUP No. F71I17000330002) founded by Fondazione di Sardegna.

Compliance with ethical standards

Conflict of interest The author does not have any conflicts of interest.

Ethical standard This article does not contain any studies with human participants performed by any of the authors.

References

Aaronson S (2007) The learnability of quantum states. Proc R Soc Lond A Math Phys Eng Sci 463:3089–3114

Aïmeur E, Brassard G, Gambs S (2006) Machine learning in a quantum world. In: Conference of the Canadian Society for Computational Studies of Intelligence. Springer, Berlin

Audenaert KMR, Calsamiglia J, Munoz-Tapia R, Bagan E, Masanes LI, Acin A, Verstraete F (2017) Discriminating states: the quantum Chernof bound. Phys Rev Lett 98:160501

Bergou J, Herzog U, Hillery M (2004) Discrimination of quantum states. In: Lectures notes in Physics, vol 649. Springer, Berlin, pp 417–465

Bisio A, Chiribella G, Mauro G, Ariano D, Facchini S, Perinotti P (2010) Optimal quantum learning of unitary transformation. Phys Rev A 82(3):032324

Carrasquilla J, Melko RG (2017) Machine learning phases of matter. Nat Phys 13:431–434

Castelvecchi D (2017) IBM's quantum cloud computer goes commercial. Nature 543(7664):159

Chefles A (2000) Quantum state discriminator. Contemp Phys 41(6):401–424

Dalla Chiara ML, Giuntini R, Leporini R, Negri E, Sergioli G (2015) Quantum information, cognition and music. Front Psychol 6:1583

Deutsch D (1985) Quantum theory, the Church–Turing principle and the universal quantum computer. Proc R Soc Lond A 400:97–117

Duda RO, Hart PE, Stork DG (2000) Pattern classification, 2nd edn. Wiley Interscience, New York

Feynamn R (1982) Simulating physics with computers. Int J Theor Phys 21(6/7):467–488

Freytes H, Sergioli G (2014) Fuzzy approach for Toffoli gate in quantum computation with mixed states. Rep Math Phys 74(2):159–180

Gambs S (2008) Quantum classification. arXiv:0809.0444v2

Guta M, Kotlowski W (2010) Quantum learning: asymptotically optimal classification of qubit states. New J Phys 12:123032

Hayashi A, Horibe M, Hashimoto T (2005) Quantum pure-state identification. Phys Rev A 72(5):052306

Helstrom CW (1976) Quantum detection and estimation theory. Academic Press, New York

Hilbert M, Lopez P (2011) The World's technological capacity to store, communicate, and compute information. Science 332:60

Holik F, Sergioli G, Freytes H, Plastino A (2017) Pattern recognition in non-Kolmogorovian structures. Found Sci 23(1):119–132

Lloyd S, Mohseni M, Rebentrost P (2013) Quantum algorithms for supervised and unsupervised machine learning. arXiv:1307.0411

Lloyd S, Mohseni M, Rebentrost P (2014) Quantum principal component analysis. Nat Phys 10(9):631–633

Lu S, Braunstein SL (2014) Quantum decision tree classifier. Quantum Inf Process 13(3):757–770

Manju A, Nigam MJ (2014) Applications of quantum inspired computational intelligence: a survey. Artif Intell Rev 42(1):79–156

Melkikh AV, Khrennikov A, Yampolskiy RV (2019) Quantum metalanguage and new cognitive synthesis. NeuroQuantology 17:72–96

Nielsen MA, Chuang IL (2010) Quantum computation and quantum information, 10th Anniversary edn. Cambridge University Press, Cambridge

Qiu D (2007) Minimum-error discrimination between mixed states. arXiv:0707.3970 [quant-phis]

Santucci E (2017) Quantum minimum distance classifier. Entropy 19(12):659

Santucci E, Sergioli G (2018) Classification problem in a quantum framework. In: Khrennikov A, Bourama T (eds) Quantum foundations, probability and information, proceedings of the quantum and beyond conference, Vaxjo, Sweden, 13–16 June 2016. Springer, Berlin, Germany, in press
Sasaki M, Carlini A (2002) Quantum learning and universal quantum matching machine. Phys Rev A 66(2):022303
Schuld M, Petruccione F (2018) Supervised learning with quantum computers. In: Quantum science and technology. Springer, Berlin
Schuld M, Sinayskiy I, Petruccione F (2014a) An introduction to quantum machine learning. Contemp Phys 56(2):172–185
Schuld M, Sinayskiy I, Petruccione F (2014b) The quest for a quantum neural network. Quantum Inf Process 13(11):2567–2586
Schuld M, Fingerhuth M, Petruccione F (2017) Implementing distance-based classifier with a quantum interference circuit. Europhys Lett 119(6):60002
Sergioli G, Santucci E, Didaci L, Miszczak J, Giuntini R (2016) A quantum-inspired version of the nearest mean classifier. Soft Comput 22(3):691–705
Sergioli G, Bosyk GM, Santucci E, Giuntini R (2017) A quantum-inspired version of the classification problem. Int J Theor Phys 56(12):3880–3888
Sergioli G, Santucci E, Didaci L, Miszczak JA, Giuntini R (2018a) A quantum inspired version of the NMC classifier. Soft Comput 22(3):691–705
Sergioli G, Russo G, Santucci E, Stefano A, Torrisi SE, Palmucci S, Vancheri C, Giuntini R (2018b) Quantum-inspired minimum distance classification in biomedical context. Int J Quantum Inf 16(8):1840011
Sergioli G, Giuntini R, Freytes H (2019) A new quantum approach to binary classification. PLoS ONE 14:e0216224. https://doi.org/10.1371/journal.pone.0216224
Trugenberg CA (2002) Quantum pattern recognition. Quantum Inf Process 1(6):471–493
Wiebe N, Kapoor A, Svore KM (2015) Quantum nearest-neighbor algorithms for machine learning. Quantum Inf Comput 15(34):318–358
Wittek P (2014) Quantum machine learning: what quantum computing means to data mining. Academic Press, New York