The Quantum Version of Random Forest Model for Binary Classification Problem

Kamil Khadiev [a], Liliya Safina [a]

[a] *Kazan Federal University, 18 Kremlyovskaya street, Kazan, 420008, Russia*

Abstract

We suggest the quantum version of prediction using random forest model for binary classification problem. The idea of the paper is to combine quantum amplitude amplification algorithm and the probabilistic aggregation of the results of different decision trees in the forest. Quantum amplitude amplification algorithm is used as a subroutine and helps us to quadratically speed up a prediction. In the classical case, a random forest model works in $O(N\,T)$, where N is a number of trees in a forest and $O(\,T)$ is a running time of prediction on one tree. The running time of our version is $O\left(\sqrt{N}\,T\right)$.

Keywords [1]

Quantum Random Forest, Random Forest, Quantum Machine Learning, Quantum Algorithms, Binary Classification

1. Introduction

Random forest is an ensemble of decision trees model of machine learning [4, 17] that is used for classification and regression problems. It is based on the bagging (bootstrap aggregation) method [7, 26]. To construct each decision, tree the random forest randomly chooses input data and attributes. Decision tree [9, 13, 20, 23] is a simple and intuitive method for solving classification problems, but it is also used for regression problems. Each non-leaf node of a tree is a rule for going to a particular child node. The result is in a leaf of the tree.

Let N be a number of trees in a forest. It is used for predictions of each tree and getting a result. We get N results from each tree. Random forest averages a result for regression problem. A result of a random forest model is obtained by voting of all trees for classification problems.

To construct decision trees random forest uses such famous algorithms as Id3, C4.5 [24], C5.0 [1], CART [10]. We propose to create a random forest model using our quantum version of constructing decision tree algorithm [19]. It allows us to build a tree for $O\left(h\sqrt{d}\,\log d\,K\,\log K\right)$, K is the size of a training data set, d is a number of attributes of each element, h is a tree height (a given parameter). In classical case the running time is equal to $O(h\,d\,K\,\log K)$ for binary classification problem. The variables K and d are different for each tree and selected by bagging method. In addition, in this work [5] the authors consider the problems where they note an influence of pseudorandom and quantum-random number generators also for quantum random forest. We can also use other quantum decision tree constructing algorithms [22] or use classical version of random forest [14].

Let $O(T)$ be the running time of a prediction on one tree. Thus the running time of a prediction on random forest model is $O(N\,T)$.

Quantum machine learning [2, 3, 21] is a new direction in artificial intelligence sphere. Quantum computing and the known quantum algorithms let speed up classical machine learning algorithms that use big data or to increase accuracy.

YRID-2020: International Workshop on Data Mining and Knowledge Engineering, October 15-16, 2020, Stavropol, Russia
EMAIL: kamilhadi@gmail.com (Kamil Khadiev); liliasafina94@gmail.com (Liliya Safina);
ORCID: 0000-0002-5151-9908 (Kamil Khadiev); 0000-0001-7182-3731 (Liliya Safina);


In this paper we present a quantum version of prediction a result for some input object. Our algorithm is based on quantum amplitude amplification algorithm [6] and has a running time equal to $O(\sqrt{N}\,T)$.

We can consider a random forest model as one classifier from big classification metamodel. In this case, the model should return a probability of belonging to a class. Due to large datasets this method, combination of big models, is needed high computing capabilities and becomes popular only recent. This method is named a stacked generalization [8, 27]. For example, in works [12, 18] the authors use a random forest model as a one classifier.

Quantum computing can be useful with this approach. In work [25] the authors offer the quantum algorithm to construct an ensemble of quantum classifiers also for binary classification problem. They suggest use any quantum machine learning algorithm as classifier. Results of classifiers are computed in parallel. An answer for some input object □ is evaluated from a single qubit measurement. In the algorithm the authors use some abstract classifier, therefore, the paper does not have a running time analysis of the algorithm.

The paper has the following structure. Section 2 contains brief information about random forest model. We present our probabilistic algorithm for prediction a result for binary classification problem in Section 3.

2. Random Forest

2.1. Decision Trees

A decision tree [23, 9, 20] is simple, fast and intuitive the machine learning method. It uses for classification, regression and clustering problems. Decision trees consist of inner nodes, which contain a simple rule as IF ... $THEN$ ... $ELSE$, and leaf nodes that contain an answer. The disadvantage of a decision tree is overfitting [11].

Random forest [16] uses an ensemble of decision trees. It is based on bootstrap aggregation method. A random forest model creates a new training dataset by randomly selecting data or attributes from a given dataset for each tree. It allows us avoid such problem as overfitting.

Let $N \in \mathbb{N}$ be a number of trees in the learned random forest. A number of trees is a given parameter of fitting the random forest model. It is selected by minimizing errors in the test sample.

2.2. Prediction

In prediction process an input object X step by step passes through the rules written in tree nodes. It works in $O(h)$, where h is a height of a tree. Heights of trees from forest can be different because height is a given parameter, and we can construct trees with different heights to increase accuracy. Let us take $O(T)$ as a running time of prediction on one tree.

This process is repeated on each tree from a forest. The running time of prediction of random forest model is equal to $O(N\,T)$. An answer is obtained by voting or averaging the result.

Let us present most popular formula for averaging a result.

Let $T_i(X)$ be a result of i-th tree. The general result of random forest can be presented:

$$answer = \frac{1}{N}\sum_{i=1}^{N} T_i(X)$$

for regression problem, for example.

For classification problem we use the voiting method. It can be presented by formula:

$$answer = \underset{c=1\ldots M}{\operatorname{argmax}} \sum_{i=1}^{N} f(T_i(X), c),$$

where

$$f(r, c) = \begin{cases} 1, if\ r = c \\ 0, otherwise \end{cases},$$

M is a number of classes.

Bellow we suggest another version to compute the answer for binary classification.

2.3. Probabilistic Algorithm

Let us consider the next problem. There are two classes: $Class1$ and $Class2$, a learned random forest model for binary classification problem. The model should determinate a result class for an input object X. It can return an answer ($Class1$ or $Class2$) or a probability of belonging of X for the first class.

Let each tree of a forest returns a probability of belonging to the first class. Let p_i be a probability of an input object X belongs to the first class and be obtained by i-th tree. Then, $1 - p_i$ is a probability of the second class for i-th tree.

Let us consider the following probabilistic algorithm. We randomly select any tree from the random forest. We choose a tree with a number i. All trees can be selected with equal probability. We classifier an object X by the i-th tree and get p_i, it is a probability of X belongs to the first class. Thus a probability of the i-th tree is selected and returns a probability of the first class for an input object X is equal to $p'_i = \frac{1}{N} p_i$, where N is a number of trees in the forest. Then, the general probability of the forest will return the first class is

$$p = \sum_{i=1}^{N} p'_i = \frac{1}{N} \sum_{i=1}^{N} p_i,$$

where p_i is a probability of the first class getting by i-th tree.

We propose the quantum algorithm for computing the value p.

3. The Quantum Version of Random Forest Prediction

Our quantum version of prediction a result for binary classification problem uses quantum amplitude amplification algorithm based on Grover's search quantum algorithm [15].

Let us consider the idea of quantum amplitude amplification algorithm. There is some probabilistic algorithm that can find an element a with probability p_a. Let a set of elements S be given. Let p_a be a probability to find a marked element $a \in S$. The sum of probabilities of all elements in S is $sum = \sum_{i=1}^{|S|} p_i = 1$. We repeat the finding process $O\left(\frac{1}{p_a}\right)$ times on the average before we find a.

The quantum amplitude amplification allows us to find the marked element a in S for $O\left(\frac{1}{\sqrt{p}_a}\right)$.

Each tree of a forest returns a number of class (1 or 2) with some probability. Then, we find the $Class1$ with the probability $p = \frac{1}{N}\sum_{i=1}^{N} p_i$. If we know p, we can repeat a search process of the first class $O\left(\frac{1}{\sqrt{p}}\right)$ times on the average to get the first class in set S. This is the idea of quantum amplitude amplification algorithm. Let us consider our approach to compute p.

Let $\frac{1}{p}$ be 2^j, where $j = \{1, 2, \dots\}$. The variable j changes until the amplitude amplification returns the first class.

Let the subroutine $QuantumAmplitudeAmplification(Forest, X, p)$ be the quantum amplitude amplification algorithm. It gets such parameters as:

- $Forest$ is a set of learned trees,
- X is an input object,
- p is a probability of belonging X to the first class.

The $QuantumAmplitudeAmplification$ function uses trained trees of the random forest. The quantum procedure allows us test in parallel an input object X on each tree in $O(T)$.

Lemma 1

The $QuantumAmplitudeAmplification$ works in $O\left(\frac{1}{\sqrt{p}}\, T\right)$.

Algorithm 1: Procedure of computing p

```
ComputingP(Forest, X)
Result: a real number p
ClassResult ← −1;
j ← −1;
while ClassResult ≠ Class1 and p ≥ 0.0001 do
    j ← j + 1;
    p ← 1/2^j;
    ClassResult ← QuantumAmplitudeAmplification(Forest, X, p);
end
return p;
```

Figure 1: Algorithm 1

Algorithm 1 presents how to compute p. The value p can be useful if we consider a random forest as a one classifier from a big metamodel. $QuantumAmplitudeAmplification$ is used as a quantum subroutine, the rest of the algorithm works classically. Due to quantum computing advantages (quantum parallelism) $QuantumAmplitudeAmplification$ tests an input object on all trees in parallel.

Algorithm 2: Procedure of prediction

```
Prediction(Forest, X)
Result: a class number (Class1 or Class2)
ClassResult ← −1;
j ← −1;
while ClassResult ≠ Class1 do
    j ← j + 1;
    p ← 1/2^j;
    ClassResult ← QuantumAmplitudeAmplification(Forest, X, p);
end
if p ≥ 0.5 then
    ClassResult ← Class1;
end
else
    ClassResult ← Class2;
end
return ClassResult;
```

Figure 2: Algorithm 2

Algorithm 2 uses an approach of p estimation to return only an answer ($Class\ 1$ or $Class\ 2$) for classification problem.

3.1. The Running Time

Let us consider the running time of our quantum version of prediction a result by a random forest model. The running time of Algorithm 1 and Algorithm 2, presented in Figure 1 and Figure 2 correspondingly, is the same and is showed bellow.

Theorem 1

The running time of the our quantum version of prediction a result for some input object X for binary classification problem is

$$O\left(\sqrt{N}\ T\right),$$

where N is a number of trees in forest and $O(T)$ is the running time of prediction on one tree.

Proof.

Let $\frac{1}{p} = 2^j$, where $j = \{1, 2, \dots\}$, j changes while the $QuantumAmplitudeAmplification$ returns the second class. We can say

$$2^{j-1} < \frac{1}{p} \leq 2^j.$$

The running time of the algorithm is

$$O\left(\sum_{i=1}^{j} \sqrt{2^i}\, T\right) = O\left(\sqrt{2^j}\, T\right) = O\left(\frac{1}{\sqrt{p}}\, T\right).$$

We can compute $\frac{1}{p}$ as follow. Let $p_i \geq \varepsilon$ for $\varepsilon = const$. Then,

$$p = \frac{1}{N}\sum_{i=1}^{N} p_i \geq \frac{1}{N}\sum_{i=1}^{N} \varepsilon \geq \frac{e}{N}$$

and

$$\frac{1}{p} \leq \frac{N}{e}.$$

We get

$$O\left(\frac{1}{\sqrt{p}}\, T\right) \leq O\left(\sqrt{N}\, T\right).$$

Moreover in Algorithm 2 (presented in Figure 2) we can stop the loop when j will be 2, because $\frac{1}{p} = 2^j$ follows $p = 0.25$. It means the probability of the second class is more than the probability of the first class. In this case the algorithm can return the answer $Class\ 2$.

4. Conclusion

We have considered a method for finding the probability of an input object belonging to the first class for a binary classification problem by a random forest model. The forest contains N trees constructed by any method: quantum or classical algorithms. A new input object X passes through the conditions contained in the tree nodes. To predict a result on one tree it needs $O(T)$ running time.

Our method of prediction is based on the quantum amplitude amplification algorithm and uses the idea of probabilistic solution. As an answer, we used the arithmetic mean of the probabilities of belonging to the first class of an input object X of all trees $\mathrm{p} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{p_i}$. Our algorithm can return $Class1$ or $Class2$ as an answer or the value p. It allows us quadratically speed up a prediction.

In classical case a prediction works in $O(N\,T)$. We propose the quantum version of a prediction that works in $O\left(\sqrt{N}\, T\right)$, where $O(T)$ is the running time of a prediction of one tree.

In the further we plan to implement our algorithm on quantum simulator and to test it on some machine learning problem.

5. Acknowledgements

The research is funded by the subsidy allocated to Kazan Federal University for the state assignment in the sphere of scientific activities, project No. 0671-2020-0065.

6. References

[1] C5.0: An informal tutorial (2019), URL: https://www.rulequest.com/see5-unix.html

[2] F. Ablayev, Ablayev M., Huang J., K. Khadiev, N. Salikhova, D. Wu, On quantum methods for machine learning problems part i: Quantum tools. Big Data Mining and Analytics pp. 41-55 (2019)
[3] F. Ablayev, Ablayev M., Huang J., K. Khadiev, N. Salikhova, D. Wu, On quantum methods for machine learning problems part ii: Quantum classification algorithm. Big Data Mining and Analytics pp. 56-67 (2019)
[4] E. Alpaydin, Introduction to machine learning. MIT press (2020)
[5] J. Bird, A. Ekart, D. Faria, On the effects of pseudorandom and quantum-random number generators in soft computing. Soft Comput 24 (2020)
[6] G. Brassard, P. Hoyer, M. Mosca, A. Tapp, Quantum amplitude amplification and estimation. Contemporary Mathematics pp. 53-74 (2002)
[7] L. Breiman, Bagging predictors. Mach Learn, 24 pp. 123-140 (1996)
[8] L. Breiman, Stacked regressions. Neural Networks pp. 49-64 (1996)
[9] L. Breiman, Random forests. Machine Learning, 45 pp. 5-32 (2001)
[10] L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and regression trees (1984)
[11] C. Brian, T. Griffiths, "Chapter 7: Overfitting", Algorithms To Live By: The computer science of human decisions. William Collins (2017)
[12] N. P. M. Chand, C.R. Krishna, E.S. Pill, M.C. Govil, A comparative analysis of svm and its stacking with other classification algorithm for intrusion detection. International Conference on Advances in Computing, Communication, Automation, (ICACCA)(Spring) pp. 1-6 (2016)
[13] scikit-learn developers: Decision trees, https://scikitlearn.org/stable/modules/tree.html
[14] scikit-learn developers: Random forest classifier, https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[15] L. Grover, A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pp. 212-219 (1996)
[16] T. Hastie, R. Tibshirani, J. Friedman, The elements of statistical learning: Data mining, inference, and prediction. Springer-Verlag p. 746 (2009)
[17] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Second Edition (2009)
[18] R. Hansch, O. Hellwich, Classification of polsar images by stacked random forests. ISPRS International Journal of Geo-Information p. 74 (2018)
[19] K. Khadiev, I. Mannapov, L. Safina, The quantum version of classification decision tree constructing algorithm c5. 0. //arXiv preprint arXiv:1907.06840 (2019)
[20] Kohavi, R., J. Quinlan, Data mining tasks and methods: Classification: decision tree discovery, Handbook of data mining and knowledge discovery. Oxford University Press (2002)
[21] D. Kopczyk, Quantum machine learning for data scientists. arXiv preprint arXiv:1804.10068 (2018)
[22] S. Lu, S. Braunstein, Quantum decision tree classifier. Quantum Inf Process 13, pp. 757-770 (2014)
[23] J. Quinlan, Induction of decision trees (1986)
[24] J. Quinlan, Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research, pp. 77-90 (1996)
[25] M. Schuld, F. Petruccione, Quantum ensembles of quantum classifiers. Sci Rep 8 (2018)
[26] D. Windridge, R. Nagarajan, Quantum bootstrap aggregation. Quantum Interaction. QI 2016. Lecture Notes in Computer Science, vol 10106. Springer, Cham (2017)
[27] D. Wolpert,Stacked generalization. Neural Networks, pp. 241-259 (1992)