ISSN/2499-5347

MARCO RAO

Department of Fusion and Technology for
Nuclear Safety and Security
Division of Technology Applications for
Security, Health and Heritage
Development of Particle Accelerators and
Medical Applications Laboratory
Frascati Research Centre, Rome

EXPERIMENTAL FINDINGS IN MACHINE LEARNING METHODS DEVELOPMENT

RT/2020/4/ENEA

ITALIAN NATIONAL AGENCY FOR NEW TECHNOLOGIES,
ENERGY AND SUSTAINABLE ECONOMIC DEVELOPMENT

MARCO RAO

Department of Fusion and Technology for
Nuclear Safety and Security
Division of Technology Applications for
Security, Health and Heritage
Development of Particle Accelerators and
Medical Applications Laboratory
Frascati Research Centre, Rome

EXPERIMENTAL FINDINGS IN MACHINE LEARNING METHODS DEVELOPMENT

RT/2020/4/ENEA

ITALIAN NATIONAL AGENCY FOR NEW TECHNOLOGIES,
ENERGY AND SUSTAINABLE ECONOMIC DEVELOPMENT

EXPERIMENTAL FINDINGS IN MACHINE LEARNING METHODS DEVELOPMENT

Marco Rao

Abstract

This work describes the results obtained by applying a machine learning method based on cluster analysis applied to a database of data that simulates an industrial production process. The topic is pattern recognition, and the method is compared to other 7 methods from literature: Classification and Regression Trees; C4.5; PART; Bagging CART; Random Forest; Boosted C5.0; Support Vector Machines.

Key words: *Machine Learning, Pattern recognition.*

Riassunto

Questo lavoro descrive i risultati ottenuti dall'applicazione di un metodo di apprendimento automatico basato sull'analisi dei cluster applicato a un database di dati che simulano un processo di produzione industriale. Il tema trattato è il riconoscimento di pattern, e il metodo è posto a confronto con altri sette metodi tratti dalla letteratura: Classification and Regression Trees; C4.5; PART; Bagging CART; Random Forest; Boosted C5.0; Support Vector Machines.

Parole chiave: Machine Learning, Pattern recognition.

INDEX

Introduction

Machine learning (ML) is a subfield of Artificial Intelligence (AI) that study the way in which the computer systems can perform a specific task without using explicit instructions, relying on patterns and inference instead. In practice, algorithms (the *learner*) build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. A complete overview of the ML areas is very hard but some introductory works could be very useful to approach this topic with an open and well-founded perspective (**Berguglia & Vaio, 2016**).

The present work is about a particular categories of ML, the supervised learning (SL, like classification algorithms and regression algorithms), in which the algorithm builds a mathematical model from a set of data that contains both the inputs and the desired outputs (**Russell & Norvig, 2010**).
SL algorithms use the training data to build mathematical model, in which each training example is represented by an array or vector (feature vector), and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with every new input. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. If the learner improves the accuracy of its outputs or predictions over time is said to have learned to perform that task (**Mitchell, 1997**).

In particular, the developed algorithm is compared, in the present report, with a Support Vector Machines and several decision tree learning:

- Decision tree learners (**DTL**), frequently used in statistics, data mining and machine learning, consists in decision tree used as a predictive model to move from observations about an item (the *branches*) to conclusions about the item's target value (the *leaves*). In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data, but the resulting classification tree can be an input for decision making.

- SVM (**Vapnik, 1995**) fall in the SL methods used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. The SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

The reason for this comparison relies in the particular type of experiment chosen, in a context of first order assessment of the potential of the developed methods in pattern recognition.

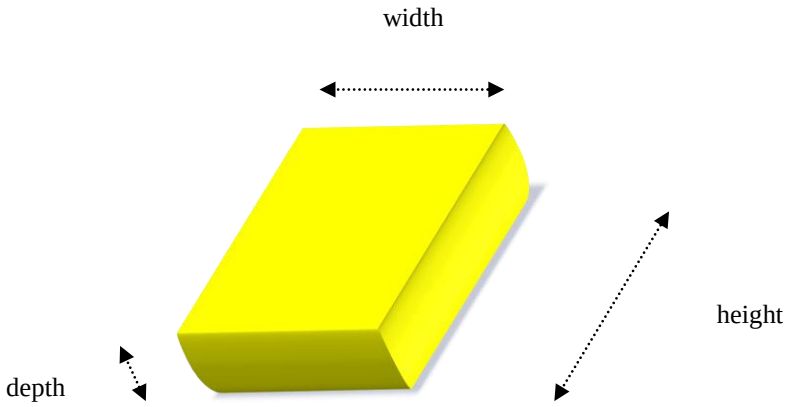
Methodology

2.1 The dataset

The database consist in list of 30 observations about 3 types of an object. The object is the simplest characterization of an industrial product out of an assembly line. About the created objects, take into account only three parameters:

1) the object height (in cm)
2) the object width (in cm)
3) the object depth (in cm)

We can imagine an assembly line that produces an object like the following:



The object was characterized by dimensions relatively similar among the three typology considered.

In order to proof the learner accuracy, we have built three types of dataset, reported in table 1.

Set	Object	width		height		depth	
		min	max	min	max	min	max
Set 1	Object1	2,0	2,2	3,0	3,3	0,5	0,6
Set 1	Object2	2,0	2,2	3,0	3,4	0,6	0,7
Set 1	Object3	2,4	2,8	3,8	3,9	0,8	0,9
Set 2	Object1	2,0	2,2	3,0	3,3	0,5	0,6
Set 2	Object2	2,3	2,5	3,5	3,7	0,7	0,9
Set 2	Object3	2,4	2,8	3,8	3,9	0,8	0,9
Set 3	Object1	2,0	2,2	3,0	3,3	0,5	0,6
Set 3	Object2	3,0	3,1	3,7	4,0	0,8	1,0
Set 3	Object3	4,0	4,2	3,9	4,2	1,5	2,0

Table 1 - Dataset used in the simulation

As shown in Table 1, the three set are characterized by an increasing level of dissimilarity among the dimension of the three types of object (for example, the absolute difference for the min and max of width is higher in the set 3 respect to the set 1).

The algorithm

The basic algorithm used performs an encoding of the main features of the simulated objects (in this case, width, height and depth of the simulated objects).

The procedure used to encode the data involves three sequentially executed sub-procedures.

As a first step, a conversion of the basic features of the object, is performed: the result is a set of real number connected to the features and arranged in a matrix X.

As a second step, a comparison between each element of the X matrix, is performed: the result is a new matrix, called D.

The third step consists in deriving a similarity index (SI) from D to perform the training step.

When a new dataset (test) is provided, the SI is used to classify the new, "un-labeled" data. The results of the performed test are shown in the next section.

We have present four version of the algorithm. The difference among the versions consists essentially in the way in which the second step is performed, being a different number of algebraic manipulation involved (in particular, the first algorithm represent the simplest case, and the subsequent version are more complex).

A complete description of the algorithm is expected in a forthcoming publication.

Results

Figure 1 shows the percentage of success for 4 versions of the algorithm, by typology of dataset and size of training dataset.

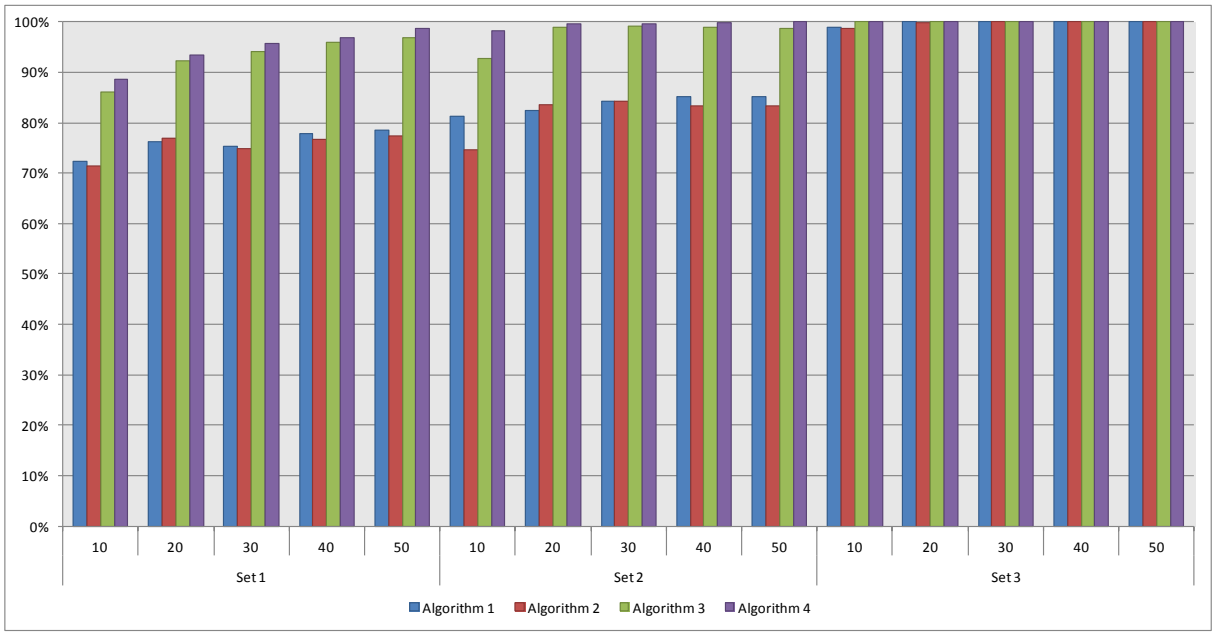


Figure 1- Comparison among four version of the developed method (percentage of success in recognition by type of dataset and by sample size)

Every version make perfect prediction for the third set, version 4 is very good also in the most difficult one, with an average percentage of success equal to 94,7%.
The size of training dataset is crucial, making also the most elementary version effective in the most difficult case, with a 76% of success.

About the size of the training dataset, figure 2 shows the average gain in percentage of success moving from 10 to 50.

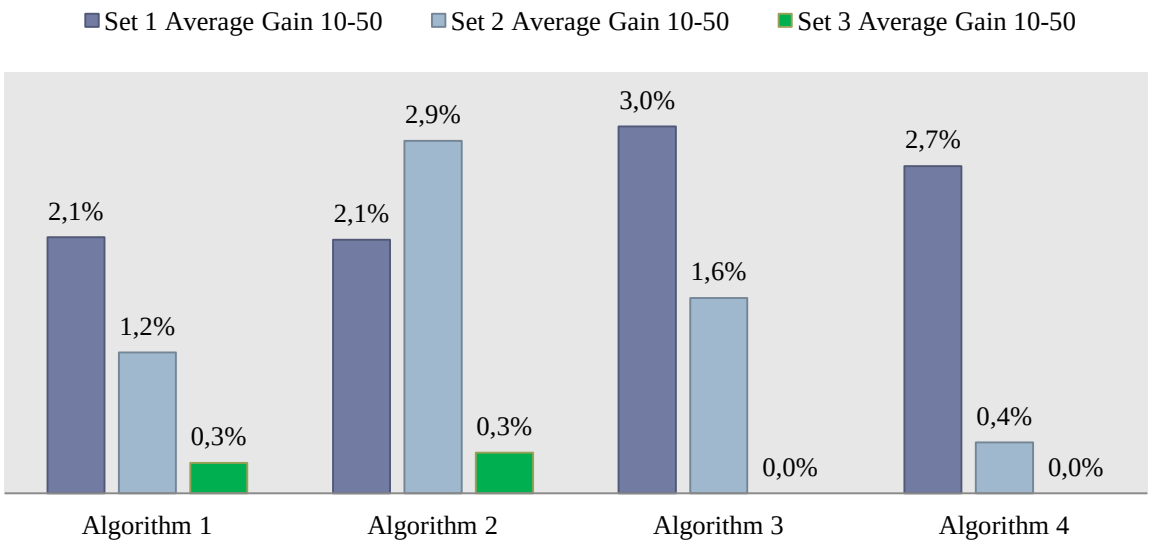


Figure 2 - Average gain in accuracy among four version of the developed algorithm, by type of dataset and by sample size

The more the Set is easier to handle for the various versions, the more the gain in percentage of success decrease (for Algorithm 4 there's no gain since the percentage is constant at 100%).

A synthetic view of the different performances of the implemented versions of the algorithm is depicted in figure 3.

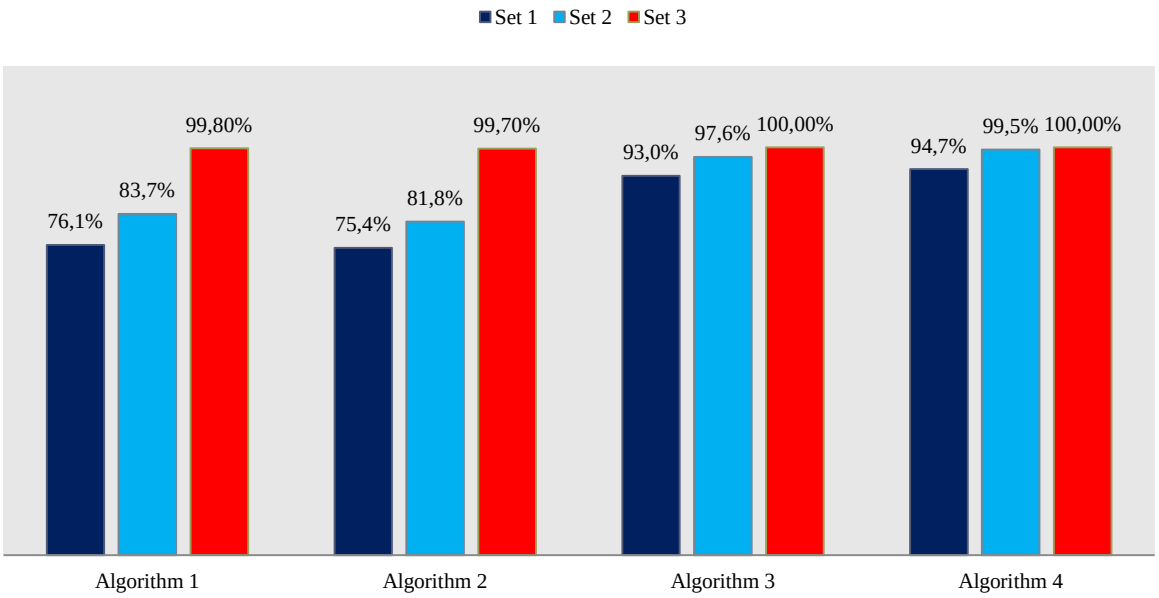


Figure 3 - Accuracy of the four version of the developed algorithm, by type of dataset (average of the accuracy by sample size, for each type of dataset).

Version 4 is clearly the best one, with a quasi-perfect recognition in all the Set provided.

The results obtained confirm a significant capacity of pattern recognition, comparable with other methods currently in use, as shown in the following section.

Comparison with other methods

The methods chosen are: Classification and Regression Trees (**Yin Loh, 2011**); C4.5 (**Quinlan, 1993**); PART (**Frank & Witten, 1998**); Bagging CART (**Breiman, 1996**); Random Forest (**Svetnik, Liaw, Tong, Culberson, Sheridan, & Feuston, 2003**); Boosted C5.0 (**Siknun & Sitanggang, 2016**); Support Vector Machines.

Table 2 reports the results of the test results, in which four different versions of the algorithm was compared to the 7 chosen predictors used. The numbers in the table reports the percentage of success in object recognition in 1000 replies. There are three set of data, characterized by different similarity among the simulated objects, and five class of size of training dataset (from 10 to 50).

This is a very partial experiment, useful to demonstrate the behavior of the new method: moving from version 1 to 4 we see an increase in success percentage. Furthermore, the success percentage rises, as similarity between object decrease while dataset training size increase.

	Set 1					Set 2					Set 3				
	10	20	30	40	50	10	20	30	40	50	10	20	30	40	50
Algorithm 1	0,72	0,76	0,75	0,78	0,79	0,81	0,83	0,84	0,85	0,85	0,99	1,00	1,00	1,00	1,00
Algorithm 2	0,71	0,77	0,75	0,77	0,77	0,75	0,84	0,84	0,83	0,83	0,99	1,00	1,00	1,00	1,00
Algorithm 3	0,86	0,92	0,94	0,96	0,97	0,93	0,99	0,99	0,99	0,99	1,00	1,00	1,00	1,00	1,00
Algorithm 4	0,89	0,94	0,96	0,97	0,99	0,99	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
SVM	0,78	0,76	0,76	0,76	0,76	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
CART	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
C4.5	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
PART	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
Bagging CART	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
Random Forest	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
C5.0	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00

Table 2 - Comparison among the four versions of the developed algorithm and the concurrent techniques, by type of dataset and by sample size.

All the algorithm versions tested significantly outperform the support vector machine, while the other predictors are better for Set 1 and Set 2. Nevertheless, version is very near to other concurrent, indicating that further refinement in the algorithm could be making it more effective.

Conclusions

The proposed algorithm was tested in simple case of pattern recognition, in which the challenge was the recognition of an industrial object, characterized by only three dimensions (width, height and depth). We've used 4 different version of the algorithm that beats the support vector machine and, for the 4th version, shows a behavior very similar to the other predictors.

Other tests are expected as soon the methodology will be consolidated.

Bibliography

Berguglia, C. S., & Vaio, F. (2016). *Complessità e modelli.* Torino: Bollati Boringhieri.

Breiman, L. (1996). Bagging predictors. *Machine Learning* , 123-140.

Frank, E., & Witten, I. H. (1998). *Generating accurate rule sets without global optimization.* Hamilton, New Zealand: University of Waikato, Department of Computer Science.

Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine . *Annals of Statistics* , 1189-1232.

Mitchell, T. (1997). *Machine Learning.* McGraw Hill.

Quinlan, J. R. (1993). *C 4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Russell, S. J., & Norvig, P. (2010). *Artificial Intelligence: A Modern Approach (Third ed.).* Prentice Hall.

Schepire, R. L. (2003). The Boosting Approach to Machine Learning: An Overview. *Nonlinear Estimation and Classification* , 149-171.

Siknun, G. P., & Sitanggang, I. S. (2016). Web-based Classification Application for Forest Fire Data Using the Shiny Framework and the C5.0 Algorithm. *Procedia Environmental Sciences* , 332-339.

Svetnik, V., Liaw, A., Tong, C., Culberson, J. C., Sheridan, R. P., & Feuston, B. P. (2003). Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. *Journal of Chemical Information and Modeling* , 1947-1958.

Vapnik, V. (1995). Support vector networks. *Machine Learning* , 273-297.

Yin Loh, W. (2011). Classification and regression trees. *WIREs Data Mining and Knowledge Discovery* , 14-23.

Yuan Deng, L., & Lin, D. K. (2000). Random number generation for the new century. *The American Statistician* , 145-150.

ENEA
Servizio Promozione e Comunicazione
www.enea.it

Stampa: Laboratorio Tecnografico ENEA - C.R. Frascati
febbraio 2020