**Use of machine learning to rapidly predict positivity to Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) using basic clinical data**

**Online-only supplement**

Authored by Thomas Langer, Martina Favarato, Riccardo Giudici, Gabriele Bassi, Roberta Garberi, Fabiana Villa, Hedwige Gay, Anna Zeduri, Sara Bragagnolo, Alberto Molteni, Andrea Beretta, Matteo Corradin, Mauro Moreno, Chiara Vismara, Carlo Federico Perno, Massimo Buscema, Enzo Grossi, Roberto Fumagalli

**Corresponding Author**

Thomas Langer, MD; Department of Anesthesia and Intensive Care Medicine, Niguarda Ca’ Granda, Milan, Italy, Department of Medicine and Surgery, University of Milan-Bicocca, Monza, Italy; thomas.langer@unimib.it

**Content of Online-only Supplement**

* Appendix 1: TWIST system
* Figure S1: Synthetic representation of the dynamic of the TWIST algorithm
* Table S1: Extended predictive results for each machine learning, employed in the study
* Table S2: Extended predictive results for each machine learning, employed in the 5 K-fold protocol cross-validation (Semeion\* and WEKA\*\*)

**Appendix 1: TWIST system**

When dealing with pattern recognition using machine learning systems, the kind of validation protocol and Inputs selection represent the most relevant problems. These problems can be expressed in this way:

1. How to generate an optimal couple of training and testing which is statistically representative of the assigned problem;
2. How to select the minimum number of input features able to maximize the accuracy of the dependent variables (target), in a blind test.

The TWIST algorithm (Training With Input Selection and Testing) is a possible scientific way to approach these two problems.

1. **Optimal distribution of records in training and testing subsets**

Any single distribution strategy of the source dataset in a training and in a testing and/or a validation set, (*i.e.* K Fold Cross Validation, Boosting, 5x2 Cross Validation, Training and Testing set splitting) is always executed with a random splitting of any single record (observation) of the source dataset.

The reason for the random criterion is of statistical nature. As the source dataset represents all the knowledge that we have of an assigned problem, we need to generate two subsets of data, which are more or less equivalent from a statistical point of view. Consequently, the training session will represent a good learning set for the machine learning, and the results of the testing session will be representative of the machine learning capability to generalize for the whole dataset. This criterion is valid, but it is not the only one.

In fact, the random criterion aims to optimize the following cost function:

(1) *f1*()*f2* ()*f0*();

where *f1*() and *f2* () = probability density function of testing and training subset, respectively; *f0*() = probability density function of the global dataset.

This means that the random criterion aims to generate two subsets with, more or less, the same probability density function, and subset should be statistically equivalent to the global dataset.

The random criterion tries to approximate the cost function defined in equation (1). However, to optimize this cost function, we should consider every possible combination of each record into the two subsets and then, for any combination, measure and compare the probability density function of each subset. There is no evidence that the random criterion can optimize this cost function.

Therefore, given a dataset *D*Γ of N records, the number of samples *dΓ*, which are comprised of K possible records, is given by



Varying K, you have:

(2)  (\*)

(\*) the search space is 2N, but the acceptable space is .

Therefore, a pair of training and testing sets represents, on the solutions space, a possible solution

, given by the vector:

(3)

To optimize the cost function presented in equation (1), we propose an evolutionary algorithm whose population expresses, after each generation, different hypotheses about the splitting of the global dataset in two subsets. Indeed, at any generation, each individual of the genetic population proposes which records of the global dataset have to be clustered into the subset A and which one into the subset B. From a technical viewpoint this is very easy: each individual of the genetic population is a vector of N Boolean values (1 or 0), where N is the number of the records of the global dataset.

The main problem, at this point, is to define a suitable fitness function, able to evaluate the validity of each hypothesis of the genetic population according to the equation (1): which hypothesis of splitting generates two subsets whose probability density functions are more similar?

To optimize these constraints we have used two independent Supervised Neural Networks (SNNs). Usually, we use a Multilayer Perceptron (Back Propagation based). The fitness evaluation of each hypothesis works in five independent steps, during which each individual of the genetic population presents its hypothesis of splitting the global dataset into two subsets, subset A and subset B:

1. The first SNN (SNN\_A) is initialized and trained using the subset A, and it is stopped when the training error (that is, for example, the Root Mean Square Error or RMSE) is minimized;
2. The SNN\_A, with the trained weights fixed, is applied in a blind way on the subset B, and its accuracy is saved;
3. The SNN\_B, which iscompletely independent from the SNN\_A, is initialized and trained using the subset B, and it is stopped when the training error (that is, for example, the RMSE) is minimized;
4. The SNN\_B, with the trained weights fixed, is applied in a blind way on the subset A, and its accuracy is saved;
5. The minimum value of the two accuracies is assigned as fitness of the hypothesis of splitting, generated by any individual of the genetic population.

The steps from 1 to 5, named “Fitness Evaluation”, are executed for each individual of the genetic population, at any generation of the evolutionary algorithm.

The flow chart of the whole algorithm is the follow:

1. Genetic Population Initialization
2. Evolutionary Loop
	1. Fitness Evaluation of the hypothesis of splitting of each individual of the genetic population at the generation (n) (From step 1 to step 5);
	2. Crossover and offspring are produced;
	3. Random mutation is applied;
	4. Setup of the new population;
	5. If the average fitness continue to grow up start from the beginning; else terminate;
3. Save the subset A and the subset B with the best fitness.

This algorithm is named Training & Testing Optimization (for short T&T).

1. **How to select the minimum number of input features able to maximize the accuracy of the dependent variable prediction**

Feature selection techniques can be developed using two different general approaches, based on whether the selection of the variables is carried out dependently or independently of the learning algorithm used to build the inductor. The filter approach attempts to select the best attribute subset by evaluating its relevance based on the data. The “wrapper” approach, instead, requires that the selection of the best attribute subset considers as relevant those attributes that allow the induction algorithm to generate a more accurate performance.

Our Input Selection algorithm (I.S.) operates as a specific evolutionary wrapper system that responds to the need to reduce the dimensionality of the data by extracting the minimum number of variables necessary to control the “peaking” phenomenon and, at the same time, conserving the most information available.

IS is an evolutionary system conceived by M. Buscema in 2000, which is based on the GenD evolutionary algorithm. It is able to weigh the different variables of the available dataset, rather than burdening the inducer to learn which of them are relevant (1). It operates as a specific evolutionary wrapper system for feature subset extraction. Unlike filter systems, that selecti features on the basis of data measures alone, IS uses the same learning algorithm for feature selection and evaluation (2).

The IS optimized sets are built considering only a subset of variables. Given a dataset  of *N* records and *M* variables, which are divided into a training set  and a testing set , there are  possible samples , and then correspondently and , of *H* extracted variables in which, varying *H*, it is also. An inducer  can be built with each sample, choosing an induction algorithm , with its configuration parameters and its initialization parameters .

It is possible to choose variables to achieve the best performance, in a transposed mode with respect to T&T, for a given pair of training and testing sets. Varying the input variables, we can consider that the goodness of the obtained results for each of the classifiers/predictors, basically depends upon whether the chosen variables, selected together, are relevant or not. We cannot know, in principle, the exact causes of the variable's relevance or irrelevance. The variables can be considered to be irrelevant by the system because they manifest redundant information, already present in other variables, or because they have no information at all, or because they manifest confusing information or noise. In any case, we have no reason to include these variables, when we can effectively eliminate them. As in T&T, the operation of analyzing the variables of the dataset before building the inducer constitutes another type of data mining, which also finalizes the extraction of the maximum amount of information present in the variables.

From a more technical perspective, IS is achieved with a GenD algorithm, considering, as space of solutions, the  possible combinations of variables and using the following two-symbol alphabet:



Where the first symbol () represents a variable's belonging to the set of relevant variables , and the second symbol () represents a variable's belonging to the set of irrelevant variables . A single possible solution  is then a pair of relevant and irrelevant variable sets and is given by the vector:



In a similar way to the T&T, a *preliminary phase* and a *computational phase* are used to evaluate the parameters of the fitness function and, respectively, to extract the most relevant training and testing set variables. A standard back propagation artificial neural network is used as inducer  for the fitness function. It is configured during the preliminary phase and used during the successive computational phase with both fixed configuration and initialization parameters (weights).

In the preliminary phase, particular attention is given to configuring the ANN parameters in order to avoid potential overfitting problems, which can arise when only a small amount of data is available with a large number of variables. (3, 4) The number  of epochs used to train the inducer is carefully evaluated so as to stop its learning phase on the training data before overfitting occurs.

GenD applied to IS usefully proposes other choices in addition to selecting the best set of variables, with a fitness less or equal to the best. These can be chosen as a practical library of input selections to be used when the best choice of variables is not accessible or is not convenient due to economic or other reasons.

As is typical for genetic wrapper systems, compared to other machine learning approaches, IS can exhibit a heavier computational time, but it also presents some useful advantages. When compared to decision tree algorithms, such as ID3, C4.5 and CART or to instance-based algorithms, like IBL, IS presents a greater performance robustness to the presence of many irrelevant features. Whereas, in comparison with naive-Bayes algorithms, a robustness to a presence of correlated features, even if relevant, can be observed. IS inherits the robustness of gradient based neural networks, like back propagation (4), and the flexibility of evolutionary algorithms to explore the space of the program solutions.

To integrate our IS algorithm with T&T algorithm in only one procedure, we have to modify the structure of each individual of the genetic population, that we have already described. This fusion will generate a complex algorithm able to look for the best distribution of the global sample into two optimal subsets with the minimum of input features, useful for optimal pattern recognition. We have named this new algorithm TWIST (Training With Input Selection and Testing).

This integration is easy: in T&T algorithm each individual of the genetic population is a vector of N component of Boolean values, where N is the number of the records of the global dataset: when the value of a generic component of the vector is 1, then that record is saved into the subset A, and when the value is 0, then the record is saved into the subset B.

In I.S. algorithm, instead, each individual of the genetic population has to be a vector of M component of Boolean values, where M is the number of all the input variables of the global dataset: in this case, when the value of a generic component of the vector is 1 the corresponding input feature is saved into the subset A and the subset B, while if the value is 0, then the corresponding input feature is removed.

Consequently, in TWIST algorithm every individual of the genetic population will be defined by two vectors of different length:

1. The first one, showing its hypothesis about which records (*N*) has to be stored into the subset A and which ones have to be stored into the subset B;
2. The second one, showing its hypothesis about which inputs (*M*) have to be used into the two subsets and which one have to be deleted.

After this modification the TWIST algorithm works as well as the T&T algorithm, already described.

At the end of its evolution, TWIST will generate two subsets of data with a very similar probability density of distribution and with the minimal number of effective variables for pattern recognition.

**Figure S1. Synthetic representation of the dynamic of the TWIST algorithm.**



**Table S1. Extended predictive results for each machine learning, employed in the study (Semeion\* and WEKA\*\*).**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Machine learning system** | **Sensitivity (%)** | **Specificity (%)** | **Overall accuracy1 (%)** | **Balanced accuracy2 (%)** | **PVV3 (%)** | **AUROC4** |
| **D\_FF\_Conic(4x12x12x12)\*** |
| A-B | 93.1 | 92.1 | 92.6 | 92.7 | 94.7 | 0.92 |
| B-A | 95.2 | 85.3 | 90.2 | 91.7 | 92.2 | 0.89 |
| **Mean** | **94.1** | **88.7** | **91.4** | **92.2** | **93.5** | **0.90** |
| **D\_FF\_Bp(24)\*** |
| A-B | 89.7 | 94.7 | 92.2 | 91.7 | 96.3 | 0.92 |
| B-A | 88.7 | 91.2 | 89.9 | 89.6 | 94.8 | 0.93 |
| **Mean** | **89.2** | **93.0** | **91.1** | **90.6** | **95.6** | **0.93** |
| **D\_FF\_Conic(6x12x12x12)\*** |
| A-B | 91.4 | 92.1 | 91.7 | 91.7 | 94.6 | 0.92 |
| B-A | 93.6 | 88.2 | 90.9 | 91.7 | 93.6 | 0.90 |
| **Mean** | **92.5** | **90.2** | **91.3** | **91.7** | **94.1** | **0.91** |
| **D\_FF\_Sn(48)\*** |
| A-B | 93.1 | 89.5 | 91.3 | 91.7 | 93.1 | 0.93 |
| B-A | 90.3 | 88.2 | 89.3 | 89.6 | 93.3 | 0.92 |
| **Mean** | **91.7** | **88.9** | **90.3** | **90.6** | **93.2** | **0.92** |
| **D\_FF\_Conic(48)\*** |
| A-B | 89.7 | 92.1 | 90.9 | 90.6 | 94.6 | 0.93 |
| B-A | 93.6 | 85.3 | 89.4 | 90.6 | 92.1 | 0.91 |
| **Mean** | **91.6** | **88.7** | **90.2** | **90.6** | **93.3** | **0.92** |
| **D\_FF\_Bp(16x16x16x16)\*** |
| A-B | 89.7 | 92.1 | 90.9 | 90.6 | 94.6 | 0.92 |
| B-A | 96.8 | 85.3 | 91.0 | 92.7 | 92.3 | 0.92 |
| **Mean** | **93.2** | **88.7** | **91.0** | **91.7** | **93.4** | **0.92** |
| **D\_FF\_GNet(64)\*** |
| A-B | 87.9 | 92.1 | 90.0 | 89.6 | 94.4 | 0.90 |
| B-A | 93.6 | 88.2 | 90.9 | 91.7 | 93.6 | 0.90 |
| **Mean** | **90.7** | **90.2** | **90.5** | **90.6** | **94.0** | **0.90** |
| **D\_FF\_Bm(48)\*** |
| A-B | 89.7 | 89.5 | 89.6 | 89.6 | 92.9 | 0.92 |
| B-A | 93.6 | 88.2 | 90.9 | 91.7 | 93.6 | 0.91 |
| **Mean** | **91.6** | **88.9** | **90.2** | **90.6** | **93.2** | **0.91** |
| **D\_FF\_Bp(0)\*** |
| A-B | 86.2 | 89.5 | 87.8 | 87.5 | 92.6 | 0.92 |
| B-A | 91.9 | 91.2 | 91.6 | 91.7 | 95.0 | 0.90 |
| **Mean** | **89.1** | **90.3** | **89.7** | **89.6** | **93.8** | **0.91** |
| **MLP\*\*** |
| A-B | 86.2 | 84.2 | 85.2 | 85.4 | 89.3 | 0.88 |
| B-A | 75.8 | 85.3 | 80.6 | 79.2 | 90.4 | 0.86 |
| **mean** | **81.0** | **84.8** | **82.9** | **82.3** | **89.8** | **0.87** |
| **RandomForest\*\*** |
| A-B | 84.5 | 65.8 | 75.1 | 77.1 | 79.0 | 0.86 |
| B-A | 88.7 | 64.7 | 76.7 | 80.2 | 82.1 | 0.87 |
| **mean** | **86.6** | **65.6** | **75.9** | **78.7** | **80.6** | **0.86** |
| **NaiveBayes\*\*** |
| A-B | 84.5 | 79.0 | 81.7 | 82.3 | 86.0 | 0.84 |
| B-A | 87.1 | 50.0 | 68.6 | 74.0 | 76.1 | 0.83 |
| **mean** | **85.8** | **64.5** | **75.1** | **78.1** | **81.0** | **0.83** |
| **RotationForest\*\*** |
| A-B | 86.2 | 65.8 | 76.0 | 78.1 | 79.4 | 0.87 |
| B-A | 90.3 | 55.9 | 73.1 | 78.1 | 78.9 | 0.82 |
| **mean** | **88.3** | **60.8** | **74.6** | **78.1** | **79.1** | **0.85** |
| **Logistic\*\*** |
| A-B | 77.6 | 71.1 | 74.3 | 75.0 | 80.4 | 0.63 |
| B-A | 83.9 | 64.7 | 74.3 | 77.1 | 81.3 | 0.63 |
| **mean** | **80.7** | **67.9** | **74.31** | **76.0** | **80.8** | **0.63** |
| **LogitBoost\*\*** |
| A-B | 84.5 | 60.5 | 72.5 | 75.0 | 76.6 | 0.83 |
| B-A | 77.4 | 61.8 | 69.6 | 71.9 | 78.7 | 0.78 |
| **mean** | **81.0** | **61.2** | **71.1** | **73.4** | **77.6** | **0.81** |
| **J48\*\*** |
| A-B | 74.1 | 68.4 | 71.3 | 71.9 | 78.2 | 0.67 |
| B-A | 80.7 | 52.9 | 66.8 | 70.8 | 75.8 | 0.47 |
| **mean** | **77.4** | **60.7** | **69.0** | **71.4** | **77.0** | **0.57** |
| **SMO\*\*** |
| A-B | 100.0 | 0.0 | 50.0 | 60.4 | 60.4 | 0.58 |
| B-A | 93.6 | 55.8 | 74.8 | 62.6 | 62.6 | 0.72 |
| **Mean** | **96.8** | **27.9** | **62.4** | **61.5** | **61.5** | **0.65** |
| **kNN\*\*** |
| A-B | 77.6 | 29.0 | 53.3 | 58.3 | 62.5 | 0.53 |
| B-A | 74.2 | 44.1 | 59.2 | 63.5 | 70.8 | 0.59 |
| **mean** | **75.9** | **36.5** | **56.2** | **60.9** | **66.6** | **0.56** |
| *This table lists the results for both testing experiment (Training Set A- Testing Set B and Training Set B- Testing Set A).***1**Aritmetic mean of sensitivity and specificity**2** Weighted mean of sensitivity and specificity**3**PPV-Positive Predictive Value.**4**AUROC- Area Under the Receiver Operator Curve. |

**Table S2. Extended predictive results for each machine learning, employed in the 5 K-fold protocol cross-validation (Semeion\* and WEKA\*\*).**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Machine learning system 5 K-fold experiments** | **Sensitivity (%)** | **Specificity(%)** | **Overall accuracy(%)1** | **Balance accuracy(%)2** | **PVV(%)3** | **AUROC4** |
| **D\_FF\_Bp(24)\*** |
| k1 | 83.3 | 85.7 | 84.5 | 84.2 | 90.9 | 0.80 |
| k2 | 91.7 | 93.3 | 92.5 | 92.3 | 95.7 | 0.93 |
| k3 | 91.7 | 73.3 | 82.5 | 84.6 | 84.6 | 0.80 |
| k4 | 100.0 | 64.3 | 82.1 | 86.8 | 82.86 | 0.77 |
| k5 | 75.0 | 71.4 | 73.2 | 73.7 | 81.8 | 0.75 |
| **Mean** | **88.3** | **77.6** | **83.0** | **84.3** | **87.3** | **0.81** |
| **D\_FF\_Conic(4x12x12)\*** |
| k1 | 87.5 | 85.7 | 86.6 | 86.8 | 91.3 | 0.90 |
| k2 | 91.7 | 93.3 | 92.5 | 92.3 | 95.7 | 0.93 |
| k3 | 79.2 | 86.7 | 82.9 | 82.1 | 90.5 | 0.80 |
| k4 | 100.0 | 64.3 | 82.1 | 86.8 | 82.8 | 0.78 |
| k5 | 79.2 | 92.9 | 86.0 | 84.2 | 95.0 | 0.86 |
| **Mean** | **87.5** | **84.6** | **86.0** | **86.5** | **91.0** | **0.84** |
| **D\_FF\_Sn(48)\*** |
| k1 | 75.0 | 92.9 | 83.9 | 81.6 | 94.7 | 0.85 |
| k2 | 91.7 | 93.3 | 92.5 | 92.3 | 95.7 | 0.93 |
| k3 | 95.8 | 80.0 | 87.9 | 89.7 | 88.5 | 0.89 |
| k4 | 100.0 | 64.3 | 82.1 | 86.8 | 82.8 | 0.82 |
| k5 | 79.2 | 85.7 | 82.4 | 81.6 | 90.5 | 0.81 |
| **Mean** | **88.3** | **83.2** | **85.8** | **86.4** | **90.4** | **0.85** |
| **D\_FF\_Conic(6x12x12)\*** |
| k1 | 87.5 | 92.9 | 90.2 | 89.5 | 95.5 | 0.92 |
| k2 | 91.7 | 93.3 | 92.5 | 92.3 | 95.7 | 0.92 |
| k3 | 91.7 | 73.3 | 82.5 | 84.6 | 84.6 | 0.83 |
| k4 | 95.8 | 71.4 | 83.6 | 86.8 | 85.2 | 0.81 |
| k5 | 79.2 | 100.0 | 89.6 | 86.8 | 100.0 | 0.85 |
| **Mean** | **89.2** | **86.2** | **87.7** | **88.0** | **92.2** | **0.86** |
| **Logistic\*\*** |
| k1 | 87.5 | 64.3 | 75.9 | 79.0 | 80.8 | 0.81 |
| k2 | 70.8 | 86.7 | 78.8 | 76.9 | 89.5 | 0.83 |
| k3 | 79.2 | 40.0 | 59.6 | 64.1 | 67.9 | 0.64 |
| k4 | 100.0 | 42.9 | 71.4 | 79.0 | 75.0 | 0.50 |
| k5 | 75.0 | 78.6 | 76.8 | 76.3 | 85.7 | 0.85 |
| **Mean** | **82.5** | **62.5** | **72.5** | **75.1** | **79.8** | **0.74** |
| **RandomForest\*\*** |
| k1 | 87.5 | 57.1 | 72.3 | 76.3 | 77.8 | 0.83 |
| k2 | 83.3 | 80.0 | 81.7 | 82.1 | 87.0 | 0.85 |
| k3 | 87.5 | 60.0 | 73.8 | 76.9 | 77.8 | 0.87 |
| k4 | 100.0 | 28.6 | 64.3 | 73.7 | 70.6 | 0.83 |
| k5 | 91.7 | 64.3 | 78.0 | 81.6 | 81.5 | 0.81 |
| **Mean** | **90.0** | **58.0** | **74.0** | **78.1** | **78.9** | **0.83** |
| *This table lists the results for each experiment. The resulting mean is the value reported in the main manuscript in Table 6.***1**Aritmetic mean of sensitivity and specificity**2** Weighted mean of sensitivity and specificity**3** PPV-Positive Predictive Value. **4** AUROC- Area Under the Receiver Operator Curve. |

**REFERENCES**

1. Rumelhart DE. Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations: MIT Press; 1986.

2. Kohavi R, John G. Wrappers for feature selection. Artificial Intelligence - AI. 1997;1.

3. Ak J, Zongker D. Feature Selection: Evaluation, Application, and Small Sample Performance. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 1997;19:153-8.

4. Rumelhart DE, Chauvin Y, Ph. D. Back propagation : theory, architectures, and applications. Hillsdale, N.J: Lawrence Erlbaum Associates; 1995.