CASCARO: Cascade of Classifiers for Minimizing the Cost of Prediction

Article *in* Pattern Recognition Letters · June 2021

DOI: 10.1016/j.patrec.2021.06.010

2 authors:

Blaise Hanczar
Paris Descartes, CPSC

Avner Bar-Hen
Conservatoire National des Arts et Métiers

Pattern Recognition Letters
journal homepage: www.elsevier.com

CASCARO: Cascade of Classifiers for Minimizing the Cost of Prediction

Blaise Hanczar[a,**], Avner Bar-Hen[b]

[a]*IBISC, Univ. Paris-Saclay (Univ. Evry), 23 Bd de France, 91037 Evry, France*
[b]*CEDRIC, CNAM, 2 rue Conte, 75003 Paris, France*

ABSTRACT

Although the prediction performance is crucial for a classifier, its cost of use is also an essential issue for practical application, however, this question is rarely addressed in the literature. The aim of this article is to propose a prediction method that controls not only the error rate but also the cost of the construction of the classifier. The main idea is that some examples are easier to predict than others and can be predicted using fewer variables i.e. with a lower prediction cost. Our method, called CASCARO, is based on a cascade of reject classifiers of increasing cost. The first classifier of the cascade required only one variable, if the prediction is not reliable the second classifier requiring one more variable is used. The principle is repeated until the last classifier using all variables. We solve the two main problems for the construction of this type of cascade: its architecture (the order of the classifier) and the simultaneous computation of the rejection regions of the classifiers. The experiments show that CASCARO produces significant improvements in the use cost without decreasing prediction performance.

1. Introduction

Quality of a prediction system is generally based on the accuracy and bounds on error rate are very important tools to control the risk of error when a new observation is classified. However, from a practical point of view, the cost to accurately predict a new observation is fundamental. It is also important to consider the price to acquire the variables required by the classifier to make a prediction. For example, in a medical application context, the classifier requires a set of biological variables for each patient and the cost represents the price of the different medical exams to obtain these variables. Note that the cost does not necessarily represent money, it may represent time in the online classifiers, memory in big data based classifiers, tolerance to second effects of treatment in medical application, or any other non-infinite resource. Another case is data confidentiality: if the quality of the classifier is sufficient without personal data, it is better, from ethical and legal matters, not to use it. A last example is given by radiology where X-ray gives very useful diagnostic but is dangerous for the patient. Minimizing the quantity of X-ray exposition is an important health issue.

**Corresponding author: Tel.: +33 1 64 85 34 61;
e-mail: blaise.hanczar@ibisc.univ-evry.fr (Blaise Hanczar)

The objective of this paper is to propose a classifier with reliable predictions and reduced costs. The main idea is that some examples are easier to predict than others and do not need all variables. For these examples, a reliable prediction can be done with a small subset of variables and can be therefore less expensive (or faster, or more ethical). We thus propose a new supervised classification approach using a cascade of classifiers with a rejection option, called CASCARO. This cascade is a sequential set of classifiers of increasing cost. The examples are submitted to the first classifier that makes a prediction. If this prediction is judged not reliable, the example is rejected to the next classifier of the cascade needing additional variables. The process is repeated until a reliable prediction has been done. This approach allows reducing the cost of the base classifier using all variables. In this approach, there is a trade-off between the accuracy and the cost of the predictions. The two main questions of our method are : (i) the computation of the rejection region of all classifiers of the cascade and (ii) the optimal order of the variables that forms the structure of the cascade. Section two presents the related works about the minimization of prediction cost and the cascade classifiers. Section three gives the formulation of our cascade model, and the approaches used for the rejection regions computation and variables order selection. Section four presents the results of CASCARO and compares them with the state of the art.

2. State of the art

The reduced prediction cost problem is related to the active feature acquisition problem in the cost-sensitive learning domain Saar-Tsechansky et al. (2009), the learner must decide whether to acquire new variable information about the case at hand to deliver more accurate a prediction. A well-known formalization of this problem is in terms of reinforcement learning and Markov decision processes, as illustrated by Kapoor and Horvitz (2009); Tan and yen Kan (2010); Nan et al. (2015). Respectively, Kapoor Kapoor and Horvitz (2009) formalizes active learning policies. Tan Tan and yen Kan (2010) proposes an attribute value acquisition algorithm driven by the expected cost saving of acquisition in the support vector machine setting. Nan Nan et al. (2015) presents an extension of the random forest approach, dealing with the cost of the variables. Another approach deployed to learn variable-effective decision-maker relies on designing cascades of classifiers. This approach, popularized by Viola and Jones Viola and Jones (2004) in the framework of image analysis and object detection, uses cheap variables to discard examples belonging to the negative (majority) class. This type of method is focused on imbalanced data with very few positive examples and a large number of negative examples. It must be noted however that Viola and Jones' main objective is to maximize the prediction accuracy, and the decision cost is not taken into account. Note that our problem is also related but different from the budget learning Kapoor and Greiner (2005), where the cost of feature acquisition is minimized during the learning procedure.

In the context of information retrieval, Wang extended the cascade mechanism to the ranking setting, taking into account variable costs within a greedy cascade approach Wang et al. (2011). Trapeznikov and Saligrama Trapeznikov and Saligrama (2013) propose a multi-stage multi-class system where the reject decision at each stage is addressed as a supervised binary classification problem; the associated generalization error is bounded depending on the cascade complexity using VC dimension. Another approach is built upon the boosting mechanism Benbouzid et al. (2012), where the cascade is implemented by skipping some of the classifiers in the boosting ensemble, depending on the case and the decision of the former classifiers. Raykar et al. Raykar et al. (2010) investigate a cascade of classifiers with a reject option: they design a soft cascade where each stage accepts or rejects examples according to a probability distribution induced by the previous stage. Each stage of the cascade is limited to linear classifiers, but these are learned jointly and globally take into account the variable cost. Chen Chen et al. (2012) re-weights and re-orders the weak learners obtained from boosting methods. After the initial training, a dictionary of classifiers is re-organized into a chain of cascades where each can reject an input or pass it on to the subsequent stage. The combination of rejection rule and cascade of classifiers is introduced by Ferri Ferri et al. (2004). A model based on small sequences of reject classifiers has been proposed in the context of personalized medicine context Hanczar and Bar-Hen (2016).

In the majority of the published methods, the structure of the cascade deterministically depends on upon the fixed order of the variables, or the classifiers. Moreover, these methods are generally developed and tested for small cascades. The approach that we propose in this paper, aims at overcoming these limitations, through learning to rank the variables themselves.

3. CASCARO : Cascade of classifiers with reject option

3.1. Formulation of the cascade

We consider a classification problem with two classes (positive "1" and negative "0") with D variables $\{v_1, ..., v_D\}$. Let a training set of N examples $\{(x_1, y_1), ..., (x_N, y_N)\}$ where $x_i \in \mathbb{R}^D$ is the variable vector and $y_i \in \{0, 1\}$ is the label. We denote c_i the cost for acquiring the i-th variable of an example. Let's $\Psi : \mathbb{R}^D \rightarrow \{0, 1\}$. Let's call basic classifier, a classifier constructed from a usual supervised learning procedure and making predictions in using all variables. Our objective is to construct a cascade that obtains better performances than the basic classifier.

In this context, the performance of a classifier is measured by two values: its error rate i.e. the probability that the prediction does not correspond to the true label, denoted $E = p(\Psi(x) \neq y)$ and its cost that is the total acquisition cost of all variables required by the classifier, denoted $C = \mathbb{E}_x \left[\sum_{i \in V(x)} c_i\right]$ where $V(x)$ is the index of variables used to classify the example x. These values are combined into a new value called loss, that represents the total performance of the classifier and is defined by:

$$L = C + \Lambda E \tag{1}$$

$\Lambda \geq 0$ is a parameter that represented the penalty of a misclassification. In our cascade, this parameter controls the trade-off between the cost and the error rate. For the basic classifier, the cost C is constant since we always have to pay for all variables. Our objective is to construct a cascade with a loss lower than the loss of the basic classifier.

3.2. Classifier with rejection option

The base element of our cascade system is the classifier with reject option Chow (1970). This type of classifier can reject examples if it does not enough confidence in the predictions. No class is assigned to rejected examples. Let's Ψ a classifier whose output $\omega(x)$ is a continuous value. In fixing a threshold t on this output, we define a classic classifier that assigns one of the two classes to each example. In fixing two thresholds $\{t_0, t_1\}$, we define a classifier that rejects some examples and assigns one of the two classes to the non-rejected examples.

$$\Psi(x) = \begin{cases} 0 & \text{if} \quad \omega(x) \leq t_0 \\ 1 & \text{if} \quad \omega(x) \geq t_1 \\ r & \text{if} \quad t_0 < \omega(x) < t_1 \end{cases} \tag{2}$$

with the constraint $t_0 \leq t_1$ and $t_0, t_1 \in [0, 1]$. r represents the rejection of the example x. The performance of the classifier depends on the following values: the error rate $E = p(\Psi(x) \neq y, \Psi(x) \neq r)$ (represented by the FP and FN regions), the penalty of an error λ_E, the accuracy $A = p(\Psi(x) = y)$ (represented by

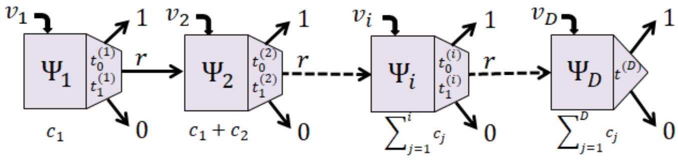


Fig. 1. Cascade of D reject classifiers.

the TP and TN regions), the penalty of a good classification λ_A, the rejection rate $R = p(\Psi(x) = r)$ (represented by the R region) and the penalty of a rejection λ_R. Note that we have $A+R+E = 1$. The performance of a reject classifier for a given example x is measured by its loss:

$$L_{\Psi}(x) = \lambda_A A + \lambda_E E + \lambda_R R \tag{3}$$

The objective is to find the thresholds t_0 and t_1 minimizing the expected loss of the classifier.

3.3. Learning of the reject options into the cascade

Our cascade system is a sequence of D classifiers with reject option $\Psi_1, ..., \Psi_D$ of increasing cost, illustrated by Figure 1. The i-th classifier Ψ_i receives all examples rejected by the classifier Ψ_{i-1}, makes predictions and sends all rejected examples to Ψ_{i+1}. The last classifier Ψ_D has no reject option and makes a prediction for all received examples. The first classifier Ψ_1 receives all examples. For the moment, we consider that the order of the variables is fixed, the classifier Ψ_i uses only the i first variables so its cost is $\sum_{j=1}^{i} c_j$. For each classifier Ψ_i, its error rate E_i, accuracy A_i and rejection rate R_i are computed as :

$$\begin{aligned}
E_i &= p(\Psi_i(x) \neq y, \Psi_i(x) \neq R | \Psi_j(x) = R\ \forall j \in [1, i-1]) \\
A_i &= p(\Psi_i(x) = y | \Psi_j(x) = R\ \forall j \in [1, i-1]) \\
R_i &= p(\Psi_i(x) = R | \Psi_j(x) = R\ \forall j \in [1, i-1])
\end{aligned} \tag{4}$$

From these formulas, we can define the loss L_i of each classifier of the cascade by a weighted combination of their error rate, accuracy and rejection rate. The weight of a good classification is the cost of the used variables, the weight of an error is the cost of the used variables plus the penalty of misclassification. When an example is rejected, it is sent to the next classifier so the weight of rejection is the loss of the next classifier L_{i+1}. The loss of an entire cascade L can be computed recursively by:

$$\begin{aligned}
L_i &= A_i \sum_{j=1}^{i} c_j + E_i(\sum_{j=1}^{i} c_j + \Lambda) + R_i L_{i+1} \\
L_D &= A_D \sum_{j=1}^{D} c_j + E_D(\sum_{j=1}^{D} c_j + \Lambda)
\end{aligned} \tag{5}$$

with $L = L_1$. The error rate and the cost of the cascade are defined by :

$$\begin{aligned}
E &= (1-R_1)E_1 + \sum_{i=2}^{D}(1-R_i)(\prod_{j=1}^{i-1} R_j)E_i \\
C &= (1-R_1)c_1 + \sum_{i=2}^{D}(1-R_i)(\prod_{j=1}^{i-1} R_j)c_i
\end{aligned} \tag{6}$$

The optimization of the cascade consist of finding the optimal rejection regions of each classifier that minimize the loss of the cascade. For the i-th classifier Ψ_i, the rejection region is defined by their two decision thresholds $(t^*_{0,(i)}, t^*_{1,(i)})$. The penalty of a good classification is $\lambda_{A,(i)} = \sum_{j=1}^{i} c_j$, the penalty of an error is $\lambda_{E,(i)} = \sum_{j=1}^{i} c_j + \Lambda$ and the penalty of a rejection is $\lambda_{R,(i)} = L_{i+1}$. In introducing these penalties in the formulas (5) we derive the optimal rejection thresholds of the classifier Ψ_i (proof in supp. mat.).

$$t^*_{0,(i)} = \frac{L_{i+1} - \sum_{j=1}^{i} c_j}{\Lambda} \qquad t^*_{1,(i)} = \frac{\sum_{j=1}^{i} c_j + \Lambda - L_{i+1}}{\Lambda} \tag{7}$$

Unfortunately, we can not simply use these formulas on each classifier to obtain the optimal cascade. The problem is that the classifiers and their performances are depending on each other. When a new rejection region of a classifier is computed, the sets of examples rejected to the next classifiers change, the performances of the next classifiers and their penalties of rejection change too. A new rejection region has, therefore, to be computed. All rejection regions, performances and penalties of all classifiers are circularly dependent.

The cascade is initialized as the basic classifier i.e. all classifiers reject all examples and all examples are sent to the last classifier using all variables. The iterative procedure contains three steps. The first one is to compute the accuracy, error rate and rejection rate of all classifiers. Then the penalties of rejection of all classifiers (excepted the last one) are computed by using the formulas (5-6). The penalty of rejection depends on the performances of the next classifier, the penalties are therefore computed from the classifier Ψ_{D-1} to the classifier Ψ_1. Finally, the two rejection thresholds are computed for each classifier from the penalties of good classification, misclassification, and rejection. This procedure is iterated until convergence.

3.4. Order of the variables

The second problem to construct the cascade is to find the optimal order of the variables. The performance of the cascade is highly depending on this order. We want the most informative and less expensive variables at the beginning of the cascade and the less informative and most expensive at the end. However the amount of information brought by a variable for prediction is not correlated to its cost, a combination of these two quantities has therefore to drive the order computation. Moreover, the amount of information of a variable is depending on the previous variables selected in the cascade. A variable correlated with the label and highly redundant with previously selected variables, is not very informative for the classification. For these reasons, the computation of the usefulness of the variables and

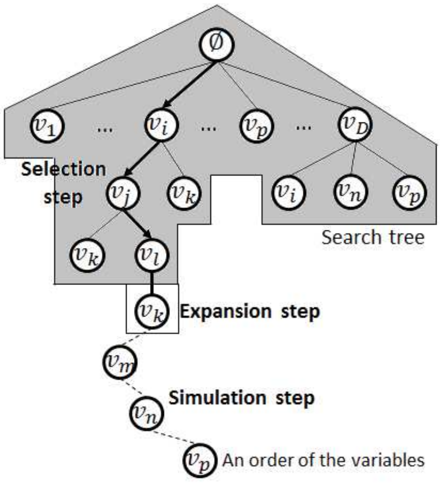


Fig. 2. Monte Carlo Search Tree procedure.

their position in the cascade is not an easy task. One solution is to test all orders and select the one that produces the cascade with the lowest loss. However there are D! different orders, this solution is tractable only for low dimension data ($D < 10$), for $D = 10$ there are already 3,628,800 orders. We propose a heuristic to find a good variable order, based on the monte carlo tree search (MCTS). The MCTS is a heuristic search algorithm for decision processes, it is particularly efficient and popular in game playing Browne et al. (2012) and has already been used to solve problems of variables selection for classification tasks Gaudel and Sebag (2010).

For the variable order problem, all orders are represented in a tree of depth $D + 1$. Each node of the tree, except the root, represents a variable. The root of the tree represents the null set, the second level represents the variable in the first position, the i-th level represents the variable in the (i-1)-th position. An order is therefore represented by a path from the root to a leaf. We call a partial order, a path from the root to a node that is not a leaf. It represents the beginning of an order to be completed. The MCTS tries to maximize a reward value representing the performance of each node in the tree. In our case, the reward depends on the performance of the cascade constructed from the partial order corresponding to a given node. The MCTS algorithm asymmetrically grows the search tree to explore the most promising order. It is an iterative algorithm containing four steps: selection, expansion, simulation and backpropagation.

Selection: Starting from the root, we select successively the best child for each node n. The best child of n is the next variable in the partial order represented by the path root to n. We select the variable v^* maximizing the following selection criterion:

$$v^* = argmax_{v \in V_n} \left\{ \hat{\mu}_{n,v} + \sqrt{\frac{k.ln(T_n)}{t_{n,v}}} \right\} \tag{8}$$

where $\hat{\mu}_{n,v}$ is the average reward for selecting the variable v from the node n, T_n is the number of times the node n has been visited, $t_{n,v}$ is the number of times v has been selected from the node n, V_n is the set of variables not selected in the partial order. This criterion is a trade-off between exploitation and exploration of the tree. $\hat{\mu}_{n,v}$ represents the exploitation and the square root term the exploration. k is a parameter controlling this trade-off.

Expansion: The selection step stops when we reach a leaf or a node whose selection criterion is higher than the selection criterion of all of its children. A child of this node is created and added to the tree by selecting randomly a variable from V_n.

Simulation: The partial order represented by the newly created node, is completed by adding randomly the rest of the variables. A reward of this order has to be computed. We should construct a cascade with this variable order, compute the rejection regions and estimate its performance on the validation set. This performance will be used as the reward of the tested order. Unfortunately, this approach is not tractable in a MCTS procedure because of the computation time of the rejection options computation. We propose an approximation of the cascade performance without computing the rejection options.

For the i-th classifier of the cascade, we associate to the rejection thresholds (t_0, t_1) a half normal distribution $N(0, \sigma_i)$ with $t_0 \leq 0$ and $t_1 \geq 0$, where σ_i is the empirical standard deviation of the output of the i-th classifier computed on the validation set. From this distribution, we can compute the probability of rejection for each example of the validation set $Pr_i(x) = 2\Phi(\frac{-|w_i(x)|}{\sigma_i})$ where Φ is the cumulative normal distribution. An estimation of the rejection rate and error rate of the i-th classifier are given by:

$$\hat{R}_i = \frac{1}{N_v} \sum_{j=1}^{N_v} Pr_i(x_j) \qquad \hat{E}_i = \frac{1}{N_v} \sum_{j=1}^{N_v} I[\Psi_i(x_j) \neq y_j] \tag{9}$$

where $I[a] = 1$ of a is true, 0 otherwise. In using these formulas in the equations (1) and (4), we obtain an estimation of the loss of the cascade for the tested order whatever the rejection regions. This loss is used to compute the reward of the order as $reward = \frac{1}{2}\Lambda - \hat{L}_{cas}$. This reward is very fast to compute, it needs only to apply all classifiers of the cascade on the validation sets and keep in memory the predictions and outputs.

Backpropagation: The average reward of each node forming the tested order, is updated with the computed reward in the simulation step.

These four steps are iterated a large number of times. Once the iterations are finished, the child of the root with the highest average reward is identified. The corresponding variable is selected as the first variable of the cascade. The selected child becomes the root of the tree and the MCTS procedure is relaunched on this sub-tree. The average reward, number of visits and standard deviation of each node are kept. At each launch of the MCTS procedure, a new variable is selected and is added to the cascade. After D MCTS launches, we obtain the complete order of the variables.

4. Experimental validation

4.1. Datasets and study design

We analyze the behavior of our method and compare its performance through a set of experiments based on both artificial and real datasets.

The artificial datasets are generated from Gaussian distributions in dimension D. The positive class follows the distribution $N(\Delta, \sigma^2 I)$ and negative class $N(\mathbf{0}, \sigma^2 I)$ where $\Delta = \{\delta_1, ..., \delta_D\}$ is a vector giving the center of the positive class. The value δ_i is also an index of the discriminative power of the i-th variable. σ^2 controls the variance of the two classes. The cost of variables is randomly generated from a uniform distribution $U[1, 10]$. Then the costs are normalized such that the sum gives 1. From this model, four artificial datasets are generated. In Artif.1 all δ_i are equal, all the variables have the same discriminatory power. In consequence, the optimal order of a cascade depends only on the cost of the variables. In Artif.2 and Artif.3 the μ_is are generated from an uniform distribution $U[0.5, 1.5]$. Artif.3 and artif.4 has a higher dimensionality than Artif.1 and artif.2 ($D = 20$) and is unbalanced. For each artificial datasets, 2000 examples are generated for the training and 10000 examples for the test.

Eight real public datasets from UCI has been used. The acquisition cost of variables is rarely provided with the dataset, for five of the datasets (WDBC, magic04, spam, sonar, and madelon) we generate artificial variable costs. The cost of variables is randomly drawn from a uniform distribution $U[1, 10]$ and normalized such that the sum gives 1. Even if these costs are not realistic for their respective classification task, they allow making a fair comparison of the algorithms. For three other datasets (lung, breast and pima) the real variables cost is available. The magic04, spam, and madelon datasets are randomly split into a training, validation, and test set. The training and validation sets are used to construct the cascade and the test set is only used to compute the performances. For the sonar, WDBC, lung, breast, and pima datasets, the performances are estimated by 10-times 10-fold cross-validation.

4.2. Sensitivity analysis

In these experiments, we investigate the impact of the parameter Λ on the performance of the cascade. Figure 3 gives respectively the error rate vs λ, the cost of the cascade vs Λ and the cost vs the error rate on an artificial dataset with the LDA classifier. The dotted line represents the basic classifier and the cross line is the cascade obtained with the CASCARO method. Λ is increasing with the cost of the cascade and decreasing with its error rate. Λ controls the trade-off between the error rate and the variable cost. For a low value of Λ, the misclassifications are more tolerated, fewer variables are therefore needed, but the error rate increases. At the extreme, $\Lambda \leq 2$ in these figures, the cascade keeps only the first variable for all examples. For a high value of Λ, the misclassifications are very penalized, the cascade needs more variables to get more information and minimize the risk of error. We see that the error rate of the cascade is never lower than the error rate of the basic classifier. That is logic since the basic classifier uses all information, i.e. all variables for all examples. The error rate of the cascade can be only higher or equal to the error of the basic classifier.

In Figure 3 left, we see that at $\Lambda = 15$ the error rate of the cascade reaches the error rate of the basic classifier. The same point at (0.399,0.237) can be observed in Figure 3 right. This point is interesting because it represents the performance of a cascade that does not increase the error rate with minimal cost. This cascade makes predictions with the same accuracy as the basic classifier for a lower cost. In this example, the cost is reduced by 60%. We call this point the SAMC (Same Accuracy Minimal Cost) point. For any classification problem, the cascade has always a SAMC point. At the extreme the cost of the SAMC point is 1, corresponding to the performance of the basic classifier. In this case, the cascade cannot improve the performance of the basic classifier.

4.3. Results on real datasets

A set of experiments has been done to estimate the performance of CASCARO and compare it with the state of the art. CASCARO is compared with the following other approaches:

- **Base classifier:** It corresponds to the basic classifier using all variables.
- **Variable selection:** A t-test score is used to rank all variables according to their discriminative power for the classification problem. Then the top d variables are selected to construct the classifier.
- **Cronus:** A cascade is constructed with the Cronus algorithm proposed by Chen et al. (2012). In Cronus paper λ corresponds to $1/\Lambda$.
- **SoftCascade:** A cascade is constructed with the soft cascade algorithm proposed by Raykar Raykar et al. (2010). In softcascade paper β corresponds to $1/\Lambda$.
- **Cheapest variable:** A cascade is constructed where the order of variables is given by their increasing cost. The rejection regions are computed by the method of section 3.3.

For a fair comparison we use the same classification algorithm for the methods "base classifier", Variable selection" and CASCARO. The two classification algorithms used in our experiments are the linear discriminant analysis (LDA) and the support vector machine (SVM) with a Gaussian kernel. For the "variable selection", the number of selected variables d varies from 1 to D. We report the results of each value of d. For the cascade methods, we use several values of $\Lambda \in [1, 30]$ to test different trade-offs between the cost and error rate.

Figure 4 shows the results of the six tested methods using the LDA classifier on three datasets. The dotted line represents the error rate of the base classifier. Note that the cost of this classifier is 1, its performances should be represented by a unique point, we use a line for a better visual comparison of the performances of the methods. The full line represents the performance of the "variables selection" method. The points represent the performances of the cascade methods: Triangles for Cronus, squares for SoftCascade, black dots for "cheapest variables" and crosses for CASCARO. For cascade methods, different points are obtained by varying the value of Λ. For

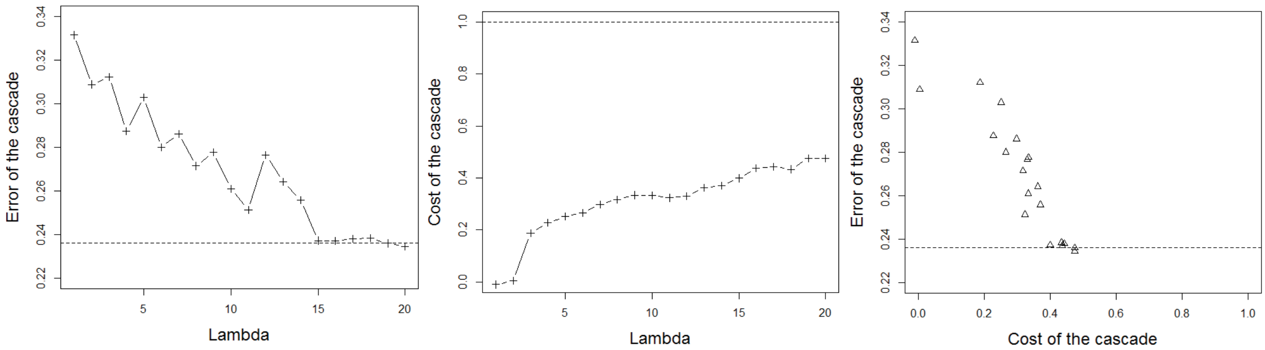
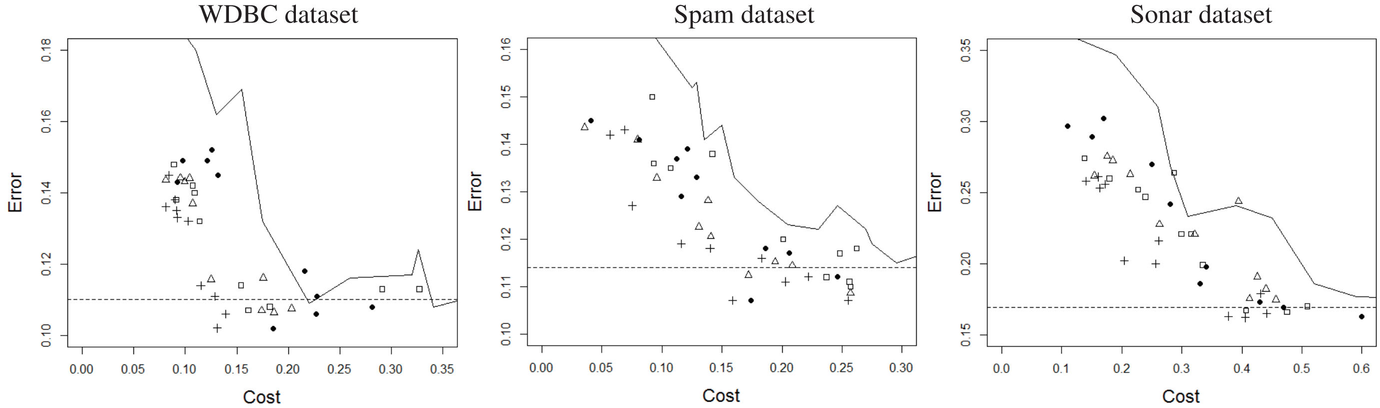


Fig. 3. Behavior of the error rate and cost of the cascade in function on its cost on artif.3 datasets.



+:CASCARO, •:Cheapest variable, □ :SoftCascade, △ :Cronus
Full line: Variable selection, Dotted line: basic classifier

Fig. 4. Results on real datasets with the LDA classifier for the six tested methods.

all methods, the error rate is naturally decreasing with the cost. We see on the four figures that CASCARO gives the best results (the crosses are closer to the bottom left corner than the other points). The performance of SoftCascade and Cronus are similar. The "cheapest variable" method is competitive with the cascade methods for the Magic and spam datasets but gives worse results on the WDBC and Sonar dataset. From these figures, we can identify the SAMC points for all cascade methods. This point represents the performance of the cascade with the minimal cost at the same accuracy as the basic classifier. Table 1 gives the cost reduction without decreasing the accuracy from these SAMC points. From this table, we conclude that the use of the cascade methods allows decreasing drastically the cost of the predictions and the best cost reduction is provide by CASCARO.

The error rates of the different methods are never significantly lower than the error rate of the "base classifier". The "base classifier" uses all variables for all test examples, it is, therefore, logical that it obtains the lower error rate. Note that in high dimension setting, the datasets may contain many redundant variables or variables non-related to the classes that can affect badly the classifier construction. In this case, the "variables selection" and cascade methods may obtain a lower error rate than the base classifier. It is not the case in our experiments because all variables are more or less relevant.

Table 2 gives the loss of the different methods on the eight real datasets with the LDA and SVM classification rules for $\Lambda = 10$. For the "variable selection", we choose the d minimizing the loss on the validation set. The "variable selection" improves the performance of the "base classifier" for real datasets in dropping weakly informative variables. The performances of "variable selection", "cheapest variables" and CASCARO are better with SVM then LDA. The "cheapest variables" improves significantly the performance of "base" and "variable selection" methods and are competitive with SoftCascade and Cronus. Cronus gives better results than SoftCascade and "cheapest variables" with LDA and has a similar performance of "cheapest variables" with SVM. CASCARO gives the lowest loss with any classification rule expected for the sonar dataset where Cronus is better than CASCARO with LDA. CASCARO outperforms all other methods.

Figure 5 gives the number of test examples classified by each classifier of CASCARO. We see that the number of examples classified by a classifier is decreasing with its position in the cascade. The first classifiers deal with the largest part of the examples that are easy to classify. Note that the number of examples in the last classifier is pretty high, this corresponds to the examples difficult to classified that is rejected by all other classifiers. A large part of the errors of the cascade comes from these difficult examples.

Methods	Artif.1	Artif.2	Artif.3	Artif.4	Magic04	WDBC	Spam	Sonar	Breast	Lung	Pima	Madelon
Cheapest variables	8%	25%	39%	59%	12%	81%	83%	57%	55%	57%	49%	51%
SoftCascade	17%	39%	52%	72%	31%	84%	76%	59%	71%	86%	78%	58%
Cronus	17%	44%	63%	81%	31%	83%	83%	58%	70%	82%	83%	56%
CASCARO	16%	41%	66%	91%	56%	89%	84%	63%	77%	91%	86%	62%

Table 1. Cost reduction without decreasing the accuracy of each cascade methods on all real datasets.

Methods	Artif.1	Artif.2	Artif.3	Artif.4	Madelon	Pima	Lung	Breast	Magic04	WDBC	Spam	Sonar
Linear Discriminant Analysis												
Base classifier	2.74	2.84	2.37	2.55	5.39	3.30	4.93	1.89	3.41	2.12	2.13	2.79
Variable selection	2.78	2.69	2.33	2.33	4.79	2.72	4.17	0.74	3.65	1.42	1.57	2.68
Cheapest variables	2.22	2.57	2.23	2.20	4.67	2.80	4.29	0.76	3.22	1.35	1.43	2.32
CASCARO	2.31	2.42	2.07	2.01	4.47	2.54	4.12	0.54	2.98	1.21	1.31	2.21
Support Vector Machine												
Base classifier	2.57	2.81	2.28	2.40	5.42	2.46	4.64	1.38	3.32	1.97	2.03	2.69
Variable selection	2.55	2.90	2.03	2.29	4.74	2.10	4.17	0.64	3.18	1.45	1.41	2.21
Cheapest variables	2.16	2.57	1.92	2.08	4.51	1.98	4.09	0.67	3.07	1.32	1.39	1.93
CASCARO	2.28	2.39	1.80	1.89	4.33	1.84	3.94	0.48	2.89	1.20	1.14	1.72
State of the art												
SoftCascade	2.33	2.41	2.19	2.19	4.59	1.89	4.03	0.51	3.41	1.30	1.47	2.35
Cronus	2.29	2.45	1.86	2.09	4.84	2.01	4.09	0.48	3.09	1.29	1.35	2.10

Table 2. Performances of the methods on the real datasets where $\Lambda = 10$.

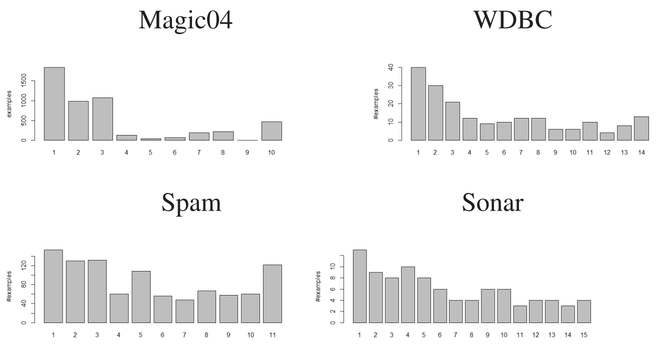


Fig. 5. Number of examples classified by each stages of CASCARO.

5. Conclusion

CASCARO is a first step to incorporate the prediction cost within classifiers. The experiments show that CASCARO produces a better trade-off cost - error rate than the other cascade methods. Note that in this paper we add only one variable to each stage of the cascade. We can easily extend this to more general cases in considering v_i as a subset of variables and c_i as the sum of the cost of the variables in this subset. Future works include optimization of the method in the function of the distribution of the costs as well as work on multi-cost problems i.e. datasets where there are several acquisition costs for each variable, for example, money and time. Selection of variables and high dimension setup are still open questions.

These cascades should be able to reduce significantly the cost of the use of predictive models in many domains. The main motivation of the use of cascades is not necessarily the economy of resources, it can also be the increase of population that will benefit from this model by redeploying the saved resources. For example, in medical diagnosis, with a fixed budget, it would be possible to test many more patients and improve the general public health policy. It is important to note some risks of the use of cascades. Some people may choose to reduce the accuracy of the model to maximize their economy by choosing a cascade whose cost is less than the SMAC point. This could have a harmful effect on some critical domains like medical applications.

References

Benbouzid, D., Busa-Fekete, R., Kégl, B., 2012. Fast classification using sparse decision dags. Proceedings of the 29th International Conference on Machine Learning (ICML-12) , 951–958.

Browne, C.B., Powley, E., Whitehouse, D., Lucas, S.M., Cowling, P.I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., Colton, S., 2012. A survey of monte carlo tree search methods. Computational Intelligence and AI in Games, IEEE Transactions on 4, 1–43.

Chen, M., Xu, Z.E., Weinberger, K.Q., Chapelle, O., Kedem, D., Saint Louis, M., 2012. Classifier cascade for minimizing feature evaluation cost., in: AISTATS, pp. 218–226.

Chow, C., 1970. On optimum recognition error and reject tradeoff. IEEE Transactions on Information Theory 16, 41–46.

Ferri, C., Flach, P., Hernández-Orallo, J., 2004. Delegating classifiers, in: Proceedings of the twenty-first international conference on Machine learning, p. 37.

Gaudel, R., Sebag, M., 2010. Feature selection as a one-player game, in: International Conference on Machine Learning, pp. 359–366.

Hanczar, B., Bar-Hen, A., 2016. Controlling the cost of prediction in using a cascade of reject classifiers for personalized medicine, in: Proceedings of the International Joint Conference on Biomedical Engineering Systems and Technologies, SCITEPRESS-Science and Technology Publications, Lda. pp. 42–50.

Kapoor, A., Greiner, R., 2005. Learning and classifying under hard budgets, in: European Conference on Machine Learning, Springer. pp. 170–181.

Kapoor, A., Horvitz, E., 2009. Breaking boundaries: Active information acquisition across learning and diagnosis. Advances in neural information processing systems .

Nan, F., Wang, J., Saligrama, V., 2015. Feature-budgeted random forest. International Conference on Machine Learning .

Raykar, V.C., Krishnapuram, B., Yu, S., 2010. Designing efficient cascaded classifiers: tradeoff between accuracy and cost, in: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM. pp. 853–860.

Saar-Tsechansky, M., Melville, P., Provost, F., 2009. Active feature-value acquisition. Management Science 55, 664–684.

Tan, Y.F., yen Kan, M., 2010. Cost-sensitive attribute value acquisition for support vector machines. Technical Report. National University of Singapore.

Trapeznikov, K., Saligrama, V., 2013. Supervised sequential classification under budget constraints, in: Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, pp. 581–589.

Viola, P., Jones, M.J., 2004. Robust real-time face detection. International journal of computer vision 57, 137–154.

Wang, L., Lin, J., Metzler, D., 2011. A cascade ranking model for efficient ranked retrieval, in: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA. pp. 105–114. URL: http://doi.acm.org/10.1145/2009916.2009934, doi:10.1145/2009916.2009934.