Tight Combinatorial Generalization Bounds for Threshold Conjunction Rules[⋆]

Konstantin Vorontsov, voron@forecsys.ru
and Andrey Ivahnenko, ivahnenko@forecsys.ru

Dorodnycin Computing Center RAS, Moscow, Russia
Moscow Institute of Physics and Technology, Moscow, Russia

Abstract. We propose a combinatorial technique for obtaining tight data dependent generalization bounds based on a splitting and connectivity graph (SC-graph) of the set of classifiers. We apply this approach to a parametric set of conjunctive rules and propose an algorithm for effective SC-bound computation. Experiments on 6 data sets from the UCI ML Repository show that SC-bound helps to learn more reliable rule-based classifiers as compositions of less overfitted rules.

Keywords: computational learning theory, generalization bounds, permutational probability, splitting-connectivity bounds, rule induction

1 Introduction

Obtaining exact generalization bounds remains an open problem in Computational Learning Theory [1]. Experiments [8, 9] have shown that two fine effects should be taken into account simultaneously to obtain exact bounds: the *splitting* of the set into error levels and the similarity of classifiers. Many practically used sets contain a lot of similar classifiers that differ on one object, they are called *connected*. In this work we develop a combinatorial approach that deals with splitting and connectivity together and gives tight or even exact bounds on probability of overfitting. We apply a new SC (splitting and connectivity) combinatorial bound to the set of conjunctive rules and propose the overfitting reduction method compatible with most usual rule induction engines.

2 Definitions and notation

Let $\mathbb{X} = \{x_1, \dots, x_L\}$ be a set of objects and A be a set of classifiers. A binary loss function $I\colon A \times X \to \{0, 1\}$ exists such that $I(a, x) = 1$ if a classifier a produces an error on an object x. A binary vector $(a_i) \equiv \bigl(I(a, x_i)\bigr)_{i=1}^{L}$ of size L is called an *error vector* of the classifier a. Assume that all classifiers from A have pairwise different error vectors.

[⋆] Supported by Russian Foundation for Basic Research grant 11-07-00480 and the program "Algebraical and combinatorial methods of cybernetics and new generation information systems" of Russian Academy of Sciences, Branch of Mathematics.

The number of errors of a classifier a on a sample $X \subseteq \mathbb{X}$ is defined as $n(a,X) = \sum_{x \in X} I(a,X)$. For shorter notation denote $n(a) = n(a,\mathbb{X})$.

The error rate is defined as $\nu(a,X) = \frac{1}{|X|} n(a,X)$.

The *learning algorithm* is a mapping $\mu\colon 2^{\mathbb{X}} \to A$ that takes a training sample $X \subseteq \mathbb{X}$ and gives a classifier $\mu X \in A$.

Transductive (permutational) probability. Denote $[\mathbb{X}]^\ell$ the set of all $\binom{L}{\ell} = \frac{L!}{\ell!(L-\ell)!}$ samples $X \subset \mathbb{X}$ of size ℓ. Assume that all partitions of the general set $\mathbb{X}$ into an observed training sample X of size ℓ and a hidden test sample $\bar{X} = \mathbb{X} \setminus X$ of size $k = L - \ell$ can occur with equal probability The classifier $a = \mu X$ is said to be *overfitted* if the *discrepancy* $\delta(a,X) = \nu(a,\bar{X}) - \nu(a,X)$ is greater than a given nonnegative threshold ε. Define the *probability of overfitting* as

$$Q_\varepsilon(\mu,\mathbb{X}) = \mathsf{P}\bigl[\delta(\mu X,X) \geq \varepsilon\bigr] = \frac{1}{\binom{L}{\ell}} \sum_{X \in [\mathbb{X}]^\ell} \bigl[\delta(\mu,X) \geq \varepsilon\bigr].$$

where brackets transform a logical value into numerical one: $[\mathit{true}] = 1$, $[\mathit{false}] = 0$. The *inversion* of a bound $Q_\varepsilon \leq \eta(\varepsilon)$ is an inequality $\nu(\mu X,\bar{X}) \leq \nu(\mu X,X) + \varepsilon(\eta)$, which holds with probability at least $1-\eta$, where $\varepsilon(\eta)$ is the inverse for $\eta(\varepsilon)$.

Empirical risk minimization (ERM) is a classical and perhaps most natural example of the learning algorithm:

$$\mu X = \arg\min_{a \in A} n(a,X).$$

Hypergeometric distribution. For a classifier a such that $m = n(a,\mathbb{X})$ the probability to make s errors on a sample X is given by a hypergeometric function: $\mathsf{P}\bigl[n(a,X) = s\bigr] = h_L^{\ell,\,m}(s)$, where $h_L^{\ell,\,m}(s) = \binom{m}{s}\binom{L-m}{\ell-s}/\binom{L}{\ell}$, argument s runs from $s_0 = \max\{0, m-k\}$ to $s_1 = \min\{m,\ell\}$, parameter m takes values $0,\dots,L$. It is assumed that $\binom{m}{s} = h_L^{\ell,\,m}(s) = 0$ for all other integers m,s. Define the cumulative distribution function (left tail) of the hypergeometric distribution

$$H_L^{\ell,\,m}(z) = \sum_{s=s_0}^{\lfloor z \rfloor} h_L^{\ell,\,m}(s).$$

Consider a set $A = \{a\}$ containing one *fixed classifier*, so that $\mu X = a$ for any X. Then the probability of overfitting Q_ε transforms into the probability of large deviation between error rates in two samples $X, \bar{X}$.

Theorem 1 (FC-bound). *For a fixed classifier a such that $m = n(a)$, for any set $\mathbb{X}$ and any $\varepsilon \in [0,1]$ the probability of overfitting is given by the left tail of hypergeometric distribution:*

$$Q_\varepsilon(a,\mathbb{X}) = H_L^{\ell,\,m}\left(\tfrac{\ell}{L}(m - \varepsilon k)\right). \tag{1}$$

The exact FC-bound plays a role of the Law of Large Numbers in permutational probabilistic framework because it predicts the error rate on the hidden sample from the observed one, whereas the "probability of error" is undefined here. The hypergeometric distribution is fundamental for all further bounds.

Theorem 2 (VC-bound). *For any set $\mathbb{X}$, any learning algorithm μ, and any $\varepsilon \in [0,1]$ the probability of overfitting is bounded by the sum of FC-bounds over A:*

$$Q_\varepsilon(\mu, \mathbb{X}) \leq \sum_{a \in A} H_L^{\ell,\, m}\left(\tfrac{\ell}{L}(m - \varepsilon k)\right), \quad m = n(a). \tag{2}$$

Further weakening this bound gives a well known form of the VC-bound: $Q_\varepsilon(\mu, \mathbb{X}) \leq |A| \max\limits_m H_L^{\ell,\, m}\left(\tfrac{\ell}{L}(m - \varepsilon k)\right) \leq |A| \cdot \tfrac{3}{2} e^{-\varepsilon^2 \ell}$ for a case $\ell = k$.

VC-bound is highly overestimated because all classifiers make approximately equal contributions to the VC-bound. However, the set of classifiers is usually split into error rates in quite nonuniform manner. Most classifiers are unsuitable, have vanishing probability to be obtained as a result of learning and make a negligible contribution to the probability of overfitting. On the other hand, similar classifiers share their contribution, thus each of them contributes poorly again. VC theory totally ignores both advantageous effects.

3 Splitting and connectivity bounds

Define the order relation on classifiers $a \leq b$ as a natural order on their error vectors: $a_i \leq b_i$ for all $i = 1, \dots, L$. Introduce Hamming distance between error vectors: $\rho(a,b) = \sum_{i=1}^L |a_i - b_i|$. Classifiers a and b are called *connected* if $\rho(a,b) = 1$. Define the precedence relation on classifiers $a \prec b$ as $\bigl(a \leq b\bigr) \wedge \bigl(\rho(a,b) = 1\bigr)$.

The set of classifiers A can be represented by a multipartite directed graph that we call the *splitting and connectivity graph* (SC-graph) in which vertices are classifiers, and edges (a,b) are pairs of classifiers such that $a \prec b$. Partite subsets $A_m = \{a \in A \colon n(a) = m\}$ are called *error layers*, $m = 0, \dots, L$. Each edge of the SC-graph (a,b) can be uniquely labeled by an object $x_{ab} \in \mathbb{X}$ such that $I(a, x_{ab}) = 0$ and $I(b, x_{ab}) = 1$.

Upper connectivity $q(a) = \#\bigl\{x_{ab} \in \mathbb{X} \bigm| a \prec b\bigr\}$ of a classifier a is the number of edges leaving the vertex a.

Inferiority $r(a) = \#\bigl\{x_{bc} \in \mathbb{X} \bigm| b \prec c \leq a\bigr\}$ of a classifier a is the number of different objects assigned to edges below the vertex a in the SC-graph. If a correct classifier $a_0 \in A$ exists such that $n(a_0) = 0$ then inferiority is equal to the number of errors, $r(a) = n(a)$. In general case $r(a) \leq n(a)$.

Theorem 3 (SC-bound). *If μ is ERM then for any $\varepsilon \in [0,1]$ the probability of overfitting is bounded by the weighted sum of FC-bounds over the set A:*

$$Q_\varepsilon(\mu, \mathbb{X}) \leq \sum_{a \in A} P_a H_{L-q-r}^{\ell-q,\, m-r}\left(\tfrac{\ell}{L}(m - \varepsilon k)\right), \tag{3}$$

where $q = q(a)$ is upper connectivity, $r = r(a)$ is inferiority, $m = n(a)$ is the number of errors of classifier a, $P_a = \binom{L-q-r}{\ell-q} / \binom{L}{\ell}$ is an upper bound on the probability to learn the classifier a, $\mathsf{P}[\mu X = a] \leq P_a$.

The weight P_a decreases exponentially as connectivity $q(a)$ and inferiority $r(a)$ increase. This fact has two important consequences.

First, connected sets of classifiers are less subjected to overfitting. Not only the fact of connectedness but better the number of connections is important.

Second, only lower layers contribute significantly to the probability of overfitting. This fact encourages effective procedures for SC-bound computation.

SC-bound (3) is much more tight than the VC-bound (2). It transforms to the VC-bound by substituting $q = r = 0$, i. e. by disregarding the SC-graph.

4 SC-bound for threshold conjunctive rules

Consider a classification problem with labels $y_i \in Y$, $i = 1, \ldots, L$ assigned to each object $x_i \in \mathbb{X}$ respectively. Consider a parametric set R of *conjunctive rules*

$$r(x;\theta) = \prod_{j\in J} \bigl[x^j \le \theta^j\bigr],$$

where $x = (x^1, \ldots, x^n)$ is a vector of numerical features of object x, $J \subseteq \{1, \ldots, n\}$ is a subset of features, $\theta^j \in \mathbb{R}$ is a threshold parameter for j-th feature. An object x is said to be *covered* by the rule $r(x)$ if $r(x) = 1$.

A rule induction system learns a rule set R_y for each class $y \in Y$ from a training set X. Two criteria are optimized simultaneously to select useful rules — the number of positive and negative examples covered by r, respectively:

$$p(r, X) = \#\bigl\{x_i \in X \bigm| r(x_i) = 1,\ y_i = y\bigr\} \to \max;$$
$$n(r, X) = \#\bigl\{x_i \in X \bigm| r(x_i) = 1,\ y_i \ne y\bigr\} \to \min\,.$$

In practice the two-criteria optimization task is reduced to one-criterion task by means of heuristic function $H(p, n)$. Examples of H are Fisher's exact test [5], entropy, Gini index, χ^2- and ω^2-tests, and many others [4].

Rule based classifier. After learning the rule sets R_y for all $y \in Y$ the classifier can be buildup as a composition of rules, e.g. as a weighted voting:

$$a(x) = \arg\max_{y\in Y} \sum_{r\in R_y} w_r r(x),$$

where weights $w_r \ge 0$ are learned from the training set X. So, there are three things to learn: (1) thresholds θ^j, $j \in J$ for each subset J; (2) feature subset J for each rule r; (3) weight w_r for each rule r. Respectively, there are three reasons for overfitting. In this work we use SC-bound to estimate overfitting resulting from thresholds learning and build a criterion for feature subset selection, with motivation that a good classifier can be hardly build up from overfitted rules.

The idea of heuristic modification is to obtain the SC-bound on p and n for a fixed J; then to get inverted estimates that hold with probability at least $1 - \eta$:

$$\tfrac{1}{k} p(r, \bar{X}) \ge \tfrac{1}{\ell} p(r, X) - \varepsilon_p(\eta), \quad \tfrac{1}{k} n(r, \bar{X}) \le \tfrac{1}{\ell} n(r, X) + \varepsilon_n(\eta),$$

and to use them instead of p, n in a heuristic $H'(p, n) = H\bigl(p - \ell\varepsilon_p(\eta), n + \ell\varepsilon_n(\eta)\bigr)$. The modified heuristic H' gives a more accurate features selection criterion that takes into account the overfitting resulting from thresholds learning.

Specialization of SC-bound for conjunctive rules. Define the binary loss function as $I(r, x_i) = \bigl[r(x_i) \neq [y_i = y]\bigr]$, $i = 1, \dots, L$, for any rule r of class y. For the sake of simplicity suppose that all values x_i^j are pairwise different for each feature j, features take integers $1, \dots, L$ and the thresholds take integers $0, \dots, L$.

For any pair of vectors $u = (u^j)_{j \in J}$, $v = (v^j)_{j \in J}$ define an order relation $(u \leq v) \leftrightarrow \forall j \in J\ (u^j \leq v^j)$. Define $(u < v) \leftrightarrow (u \leq v \text{ and } u \neq v)$.

A *boundary point* of the subset $S \subseteq \mathbb{X}$ is a vector θ_S: $\theta_S^j = \max\limits_{x \in S} x^j$, $j \in J$. Note that $r(x, \theta_S) = 1$ for any $x \in S$.

A *boundary object* of the subset $S \subseteq \mathbb{X}$ is any object $x \in S$: $\exists j \in J$: $x^j = \theta_S^j$.

A *boundary subset* is a subset $S \subseteq \mathbb{X}$ such that all objects $x \in S$ are its boundary objects. Each boundary subset is unambiguously defined by its boundary point. Empty set is a boundary subset with boundary point $\theta_\varnothing^j = 0$, $j \in J$.

The following theorem states that the sum over all rules in SC-bound (3) can be calculated by running over all boundary subsets.

Theorem 4. *The set of rules $r(x; \theta_S)$ where S runs over all boundary subsets coincide with the set of all rules $r(x; \theta)$ having pairwise different error vectors.*

Then our idea is to iterate all rules (really, boundary subsets) from bottom to upper levels of SC-graph and use early stopping to bypass rules from higher levels that make no significant contribution to the SC-bound. To do this effectively we first consider an algorithm for the neighbor search of boundary subsets.

Searching the set of neighbor rules. Consider a rule $r(x; \theta)$ defined by a threshold vector $\theta = (\theta^j)_{j \in J}$. Its neighborhood V_θ is defined as a set of all rules $r(x, \theta')$ that differ from $r(x; \theta)$ on single object. Algorithm 4.1 iterates all neighbor rules $r(x, \theta')$ such that θ' is a boundary point of a boundary subset.

At first stage (steps 1–5) neighbor rules $r(x; \theta')$ are produced from θ by decreasing thresholds $\theta' \leq \theta$. For each boundary object thresholds θ decrease until another object becomes boundary.

At second stage (steps 6–11) neighbor rules $r(x; \theta')$ are produced from θ by increasing thresholds $\theta' \geq \theta$. This is a more involved case that requires a recursive procedure. The preliminary work at steps $6, 7$ helps to reduce further search by determining the maximal boundary $\overline{\theta}$ that neighbor rules may fall in.

Each object $x \in \mathbb{X}$ can have one of three states: $x.\mathsf{checked} \in \{\mathsf{false}, \mathsf{bad}, \mathsf{good}\}$, what enables to avoid the repeated processing of objects at second stage.

Initially all objects are not checked. Then for each object $x \in \mathbb{X}$ covered by the rule $r(x; \overline{\theta})$ but not covered by the rule $r(x; \theta)$ the recursive procedure $\mathsf{Check}(x)$ is invoked. If the rule $r(x; \theta')$ covers only one object x in addition to objects covered by $r(x; \theta)$ then x is good. Otherwise the rule $r(x; \theta')$ covers two objects $x, \tilde{x}$ not covered by the rule $r(x; \theta)$; in such case x is bad and the procedure $\mathsf{Check}(\tilde{x})$ is invoked recursively with the object $\tilde{x}$. Each good object x induces the neighbor rule $\theta' = \max\{\theta, x\}$, which is added to the set V_θ.

Algorithm 4.1 guarantee that all neighbor rules will be found. Besides it calculates all characteristics of the rule needed to calculate SC-bound: $q(\theta)$,

Algorithm 4.1 Seek the neighborhood V_θ of the rule $r(x;\theta)$.

Require: features subset J, thresholds $\theta = (\theta^j)_{j \in J}$, class label $y \in Y$, general set $\mathbb{X}$.
Ensure: V_θ, X_θ, X'_θ, $q(\theta)$, $r(\theta)$, $n(\theta)$.

```
 1: $V_\theta := \varnothing$;
 2: for all $x \in \mathbb{X}$ such that $r(x;\theta) = 1$ and $\theta^j = x^j$ for some $j \in J$ do
 3:     for all $j \in J$ such that $x^j = \theta^j$ do
 4:         $\theta'^j := \max\{x_i^j \mid x_i \in \mathbb{X},\ x_i < \theta\}$;
 5:     AddNeighbor$(\theta, \theta', x)$;
 6: for all $j \in J$ do
 7:     $\bar{\theta}^j := \max\{L, x^j \mid x \in \mathbb{X},\ x^j > \theta^j,\ x^t \le \theta^t,\ t \ne j\}$;
 8: for all $j \in J$ do
 9:     for all $x$ such that $\theta^j < x^j \le \bar{\theta}^j$ do
10:         if $x < \bar{\theta}$ and $x$.checked = false then
11:             Check$(x)$;

12: Procedure Check$(x)$
13: for all $j \in J$ such that $\theta^j < x^j$ do
14:     for all $\tilde{x}$ such that $\theta^j < \tilde{x}^j < x^j$ do
15:         if $\theta < \tilde{x} < x$ then
16:             $x$.checked := bad;
17:             if $\tilde{x}$.checked = false then Check$(\tilde{x})$;
18:             exit;
19: $x$.checked := good;
20: $\theta'^j := \max\{\theta^j, x^j\}$, for all $j \in J$;
21: AddNeighbor$(\theta, \theta', x)$;

22: Procedure AddNeighbor $(\theta, \theta', x_i)$
23: add the rule $\theta'$ into the set of neighbors $V_\theta$;
24: if $r(x_i;\theta) = [y_i{=}y]$ then
25:     $X_\theta := X_\theta \cup \{x_i\}$;    $q(\theta) := |X_\theta|$;
26: else
27:     $X'_\theta := X'_\theta \cup X'_{\theta'} \cup \{x_i\}$;    $r(\theta) := |X'_\theta|$;    $n(\theta) := n(\theta') + 1$;
```

$r(\theta)$, and $n(\theta)$. To avoid the exhaustive search of all objects at steps 4, 7, 9, 14 the sorting index is to be built in advance for each coordinate $j \in J$.

Level-wise bottom-up calculation of SC-bound. Algorithm 4.2 starts from a lowest layer of classifiers. At each step it process a layer Θ consisting of all rules θ that makes $m = n(\theta)$ errors on general set. For each rule θ Algorithm 4.2 calculates its contribution to the SC-bound, builds its neighborhood, and forms the $(m{+}1)$-th layer Θ' joining upper parts of all neighborhoods. The steps are repeated until layer contribution becomes sufficiently small. Note that early stopping gives a lower estimate for (3), witch is upper bound on probability of overfitting.

Experiment. We used following state-of-the art algorithms as baseline rule learners: C4.5 [7], C5.0 [6], RIPPER [2], and SLIPPER [3]. Our rule learning engine was based on breadth-first search as feature selection strategy and Fisher's exact test (FET) as heuristic H. To build compositions of rules three algorithms

Algorithm 4.2 SC-bound calculation for the set of conjunction rules.

Require: features subset J, class label $y \in Y$, set of objects $\mathbb{X}$.
Ensure: Q_ε — SC-bound on probability of overfitting (3).

```
$\varTheta := \operatorname{Arg}\min_{\theta} n(\theta)$;   $Q_\varepsilon := 0$;
repeat
    $Q_{\varepsilon,m} := 0$;   $\varTheta' := \varnothing$;
    for all $\theta \in \varTheta$ do
        call Algorithm 4.1 to build the neighborhood $V_\theta$;
        $Q_{\varepsilon,m} := Q_{\varepsilon,m} + \frac{1}{C_L^\ell} C_{L-q(\theta)-r(\theta)}^{\ell-q(\theta)} H_{L-q(\theta)-r(\theta)}^{\ell-q(\theta),\, n(\theta)-r(\theta)} \bigl(\frac{\ell}{L}(n(\theta) - \varepsilon k)\bigr)$;
        $\varTheta' := \varTheta' \cup \bigl\{\theta' \in V_\theta \colon n(\theta') = n(\theta) + 1\bigr\}$;
        $Q_\varepsilon := Q_\varepsilon + Q_{\varepsilon,m}$;   $\varTheta := \varTheta'$;
until the contribution of the $m$-th layer $Q_{\varepsilon,m}$ becomes small.
```

has been implemented. Logistic Regression (LR) is a linear classifier that aggregates rules learned independently. Weighted Voting (WV) is a boosting-like ensemble of rules similar to SLIPPER which learns each subsequent rule from reweighted training set. Decision List (DL) is a greedy algorithm which learns each subsequent rule from training objects not covered by all previous rules.

We implemented two modifications of heuristic $H'(p, n)$. The SC modification uses SC-bound on the probability of overfitting Q_ε as described above. The MC modification uses the Monte-Carlo estimation of Q_ε via 100 random partitions $\mathbb{X} = X \sqcup \bar{X}$. For both modifications we set $\ell = k$.

Results are presented in Table 1. Initial unmodified versions of our algorithms WV and DL with FET heuristic have a performance comparable to the baseline. WV outperforms DL, what corresponds to the results of other authors. Both SC and MC modifications of the FET heuristic reduce overfitting of rules and of classifier as a whole. Both modified classifiers outperform their respective initial versions for all 6 tasks. It must be emphasized that the only modification is the rule evaluation heuristic, all other things being equal. Thus, all difference in performance is due to generalization bound used in modified FET heuristic. The difference between SC and MC results is not significant, but MC estimation is much more time consuming. A moderate looseness of the SC-bound really takes place but does not reduce its practical usefulness as a rule selection criterion.

5 Conclusion

This work gives a new SC (splitting and connectivity) combinatorial bound on probability of overfitting. It takes into account a fine internal structure of the set of classifiers formalized in terms of the SC-graph. For a set of threshold conjunctive rules a level-wise bottom-up algorithm with early stopping is proposed for effective SC-bound computation. The inverted SC-bound is used to modify standard rule evaluation heuristic. This modification can be build in most known rule learners. It enables to take into account an amount of overfitting resulting from thresholds learning, and then to select features subset for each

Table 1. Experimental results on 6 real data sets from UCI Machine Learning Repository. For each pair ⟨task, algorithm⟩ an average testing error obtained from 10-fold cross validation is given, in percents. For each task three best results are bold-emphasized. Algorithms 1–7 are baseline rule learners. Our algorithms: WV — Weighted Voting, DL — Decision List, SC — using heuristic modified by SC-bound, MC — using heuristic modified by Monte-Carlo estimation of the probability of overfitting.

	algorithms	tasks					
		australian	echo-card	heart dis.	hepatitis	labor	liver
1	RIPPER−opt	15.5	**2.9**	19.7	20.7	18.0	32.7
2	RIPPER+opt	15.2	5.5	20.1	23.2	18.0	**31.3**
3	C4.5 (Tree)	14.2	5.5	20.8	18.8	14.7	37.7
4	C4.5 (Rules)	15.5	6.8	20.0	18.8	14.7	37.5
5	C5.0	**14.0**	4.3	21.8	20.1	18.4	31.9
6	SLIPPER	15.7	4.3	**19.4**	**17.4**	**12.3**	32.2
7	LR	14.8	4.3	19.9	18.8	14.2	32.0
8	WV	14.9	4.3	20.1	19.0	14.0	32.3
9	DL	15.1	4.5	20.5	19.5	14.7	35.8
10	**WV+MC**	**13.9**	**3.0**	**19.5**	**18.3**	**13.2**	**30.7**
11	**DL+MC**	14.5	3.5	19.8	18.7	13.8	32.8
12	**WV+SC**	**14.1**	**3.2**	**19.3**	**18.1**	**13.4**	**30.2**
13	**DL+SC**	14.4	3.6	19.5	18.6	13.6	32.3

rule more accurately. Experiments on six real data sets show that the proposed modification reduces overfitting significantly.

References

1. Boucheron, S., Bousquet, O., Lugosi, G.: Theory of classification: A survey of some recent advances. ESAIM: Probability and Statistics (9), 323–375 (2005)
2. Cohen, W.W.: Fast effective rule induction. In: Proc. of the 12th International Conference on Machine Learning, Tahoe City, CA. pp. 115–123. Morgan Kaufmann (1995)
3. Cohen, W.W., Singer, Y.: A simple, fast and effective rule learner. In: Proc. of the 16 National Conference on Artificial Intelligence. pp. 335–342 (1999)
4. Fürnkranz, J., Flach, P.A.: Roc 'n' rule learning-towards a better understanding of covering algorithms. Machine Learning 58(1), 39–77 (2005)
5. Martin, J.K.: An exact probability metric for decision tree splitting and stopping. Machine Learning 28(2-3), 257–291 (1997)
6. Quinlan, J.R.: C4.5: Programs for machine learning. Morgan Kaufmann, San Francisco, CA (1993)
7. Quinlan, J.R.: Bagging, boosting, and C4.5. In: AAAI/IAAI, Vol. 1. pp. 725–730 (1996)
8. Vorontsov, K.V.: Combinatorial probability and the tightness of generalization bounds. Pattern Recognition and Image Analysis 18(2), 243–259 (2008)
9. Vorontsov, K.V.: Splitting and similarity phenomena in the sets of classifiers and their effect on the probability of overfitting. Pattern Recognition and Image Analysis 19(3), 412–420 (2009)