Machine Learning (2022) 111:1551–1595
https://doi.org/10.1007/s10994-021-06109-0

A stochastic approach to handle resource constraints as knapsack problems in ensemble pruning

András Hajdu[1] · **György Terdik**[1] · **Attila Tiba**[1] · **Henrietta Tomán**[1]

Received: 22 November 2019 / Revised: 18 September 2021 / Accepted: 20 October 2021 /
Published online: 18 November 2021

Abstract
Ensemble-based methods are highly popular approaches that increase the accuracy of a decision by aggregating the opinions of individual voters. The common point is to maximize accuracy; however, a natural limitation occurs if incremental costs are also assigned to the individual voters. Consequently, we investigate creating ensembles under an additional constraint on the total cost of the members. This task can be formulated as a knapsack problem, where the energy is the ensemble accuracy formed by some aggregation rules. However, the generally applied aggregation rules lead to a nonseparable energy function, which takes the common solution tools—such as dynamic programming—out of action. We introduce a novel stochastic approach that considers the energy as the joint probability function of the member accuracies. This type of knowledge can be efficiently incorporated in a stochastic search process as a stopping rule, since we have the information on the expected accuracy or, alternatively, the probability of finding more accurate ensembles. Experimental analyses of the created ensembles of pattern classifiers and object detectors confirm the efficiency of our approach over other pruning ones. Moreover, we propose a novel stochastic search method that better fits the energy, which can be incorporated in other stochastic strategies as well.

Keywords Ensemble creation · Majority voting · Knapsack problems · Stochastic selection

Editor: Joao Gama.

✉ Attila Tiba
tiba.attila@inf.unideb.hu

András Hajdu
hajdu.andras@inf.unideb.hu

György Terdik
terdik.gyorgy@inf.unideb.hu

Henrietta Tomán
toman.henrietta@inf.unideb.hu

[1] Faculty of Informatics, University of Debrecen, POB 400, Debrecen 4002, Hungary

1 Introduction

Ensemble-based systems are rather popular in several application fields and are employed to increase the decision accuracy of individual approaches. We also encounter such approaches for pattern recognition purposes (Lam and Suen 1997), using models based on, e.g., neural networks (Cho and Kim 1995; Hansen and Salamon 1990), decision trees (Kong and Dietterich 1995) or other principles (Ho et al. 1994; Huang and Suen 1995; Xu et al. 1992). In the most recent results, we can recognize this approach in the design of state-of-the-art convolutional neural networks (such as GoogLeNet, incorporating the Inception module Szegedy et al. 2015) or the direct combination of them (Harangi et al. 2018). In our practice, we also routinely consider ensemble-based approaches to aggregate the outputs of pattern classifiers (Antal and Hajdu 2014) or detector algorithms (Antal and Hajdu 2012), usually by some majority voting-based rule. During these efforts, we have also faced perhaps the most negative property of creating ensembles, that is, the increasing demand on resources. This type of cost may occur as the execution/training time and the working hours needed to create the ensemble components, etc., according to the characteristics of the given problem. Thus, in addition to the primary aim of the composition of the most accurate ensemble, a natural constraint emerges as a cost limitation for that endeavor.

The current literature mostly refers to the selection of an efficient ensemble from a pool of possible members as ensemble pruning (Zhou 2012). Even if no resource constraints are applied, a subset of possible ensemble members may lead to better performance than selecting all the members. Moreover, the best strategy is to compose an ensemble having such good performing members which also have diverse behavior. To realize this approach, ensemble pruning methods can be categorized in the following three main groups: ordering-, clustering-, and optimization-based pruning. Ordering-based pruning ranks the individual members according to some criterion, and the most highly ranked ones are put into the final ensemble. Clustering-based pruning aims to identify representative prototype individual members to compose the ensemble, while the optimization-based approach sets up an objective function and forms a subset of members by minimizing or maximizing it. Beyond the clustering-based approaches, but still as a stochastic one we can mention (Hernández-Lobato et al. 2009), as an instance-based pruning method, where the members are selected for each instance separately. Double-pruning can also be executed based on Hernández-Lobato et al. (2009) as proposed in Soto et al. (2010), which approach has similar motivation to ensemble distilling (a.k.a. compression) to reduce the size of the ensemble/learning model (Bucilu et al. 2006; Hinton et al. 2015). There are efforts to complement the basic ensemble pruning models to consider possible resource constraints like training/test execution times or memory/storage space (Bucilu et al. 2006; Hinton et al. 2015) as well. To reach this aim a popular approach is to apply multi-objective evolutionary algorithms, like NSGA-II (Deb et al. 2002). NSGA-II is an elitist algorithm that provides fast nondominated sorting and considers density estimation and crowded-comparison to maintain diversity. These positive properties can be exploited in ensemble pruning like in Mousavi and Eftekhari (2015). As the best fits to our single-objective setup we have implemented a general purpose genetic algorithm from Goldberg (1989) (chapters 2–3), and a boosting-based pruning one from Martinez-Munoz and Suarez (2007). In our comparative analyses we will refer to them as Genetic and Pruning, respectively.

In this work, we analyze a single-constraint task on the resources to compose the most accurate ensemble regarding the energy formed by majority voting as the aggregation rule like in Hernández-Lobato et al. (2009). The constraint we consider corresponds to the

training time; however, any other type of resources could be considered. We introduce a novel, theoretically well-founded stochastic approach that considers the energy as the joint probability function of the member accuracies. As our main contribution, we show that this type of knowledge can be efficiently incorporated in any stochastic search process as a stopping rule, since we have the information on the expected accuracy or, alternatively, the probability of finding more accurate ensembles. Our empirical analyses also show that including the stochastic estimation as a stopping rule saves a large amount of search time to build accurate ensembles.

We formulate the resource constraint as a knapsack problem, which provides the opportunity of a precise constraint prescription instead of a simple good price/value expectation considered e.g. in Bucilu et al. (2006), Hernández-Lobato et al. (2009), Hinton et al. (2015) to have small, but relatively accurate ensembles. Basically, we follow an ordering-based approach combined with stochastic sampling to compose the ensembles; however, additionally as a novel contribution we suggest a new heuristics for that. Namely, besides its individual accuracy and cost, we calculate such a usefulness value for each possible member during the selection process that reflects its direct behavior according to the objective function, which is based on the majority voting rule in our case. As we will see, our novel stochastic search method is proven to be very competitive with simulated annealing (SA) (Du and Swamy 2016), and the pruning methods (Goldberg 1989; Martinez-Munoz and Suarez 2007). Also, the proposed heuristic can be successfully inserted into these general stochastic search strategies. Our approach was first proposed in our former work (Hajdu et al. 2016) with limited empirical evaluations; however, only heuristic results were achieved there without being able to take advantage of the theoretical model presented here.

The rest of the paper is organized as follows. In Sect. 2, we introduce the basic concepts and notation to formulate our constrained enesemble pruning task as a knapsack problem, where majority voting is applied for the aggregation of member opinions. Section 3 analyzes the maximum accuracy of the common deterministic ensemble creation strategies in the case of limited total cost. Though the cost may relate to any resource demand, in our scenarios it describes the training times of the classifiers that are considered as possible ensemble members. Existing stochastic approaches are described in Sect. 4 with some preliminary simulation results. Moreover, we introduce a novel stochastic search algorithm that determines the expected usefulness of possible members in a way that adapts to the characteristics of the energy function better than other stochastic search methods, e.g., SA. The stochastic estimation of the ensemble energy from the individual accuracies of the components is presented in Sect. 5.[1] Our experimental analyses are enclosed in Sect. 6, including the investigation of the possible creation of ensembles from participating methods of Kaggle challenges and binary classification problems in UCI databases.[2] We also present how the proposed model is expected to be generalized to multiclass classification tasks with a demonstrative example on our original motivating object detection problem. Finally, in Sect. 7, we discuss generalization possibilities and several issues regarding our approach that can be tuned towards special application fields.

[1] Our code is available at https://codeocean.com/capsule/3944336.

[2] Our data is available at https://ieee-dataport.org/documents/binary-classifiers-outputs-ensemble-creation.

2 Basic concepts and notation

Let us consider a pool $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_n\}$ containing possible ensemble members, where each member $\mathcal{D}_i$ $(i = 1, \ldots, n)$ is characterized by a pair (p_i, t_i) describing its individual accuracy $p_i \in [0, 1]$ and cost $t_i \in \mathbb{R}_{>0}$. The individual accuracies are supposed to be known, e.g., by determining them on some test data and by an appropriate performance metric. In this work, we will focus on the majority voting-based aggregation principle, where the possible ensemble members $\mathcal{D}_i$ $(i = 1, \ldots, n)$ are classifiers (see Kuncheva 2004). In Hajdu et al. (2013), we have dealt with the classic case in which the individual classifiers make true/false (binary) decisions. In this model, a classifier D_i with accuracy p_i is considered as a Bernoulli distributed random variable η_i, that is, $P(\eta_i = 1) = p_i$, $P(\eta_i = 0) = 1 - p_i$ $(i = 1, \ldots, n)$, where $\eta_i = 1$ means the correct classification by D_i. In this case, we obtain that the accuracy of an ensemble $\mathcal{D}' = \{\mathcal{D}_{i_1}, \ldots, \mathcal{D}_{i_\ell}\} \subseteq \mathcal{D}$ of $|\mathcal{D}'| = \ell$ members can be calculated as

$$q_\ell(\mathcal{L}) = \sum_{k=\left\lfloor \frac{\ell}{2} \right\rfloor + 1}^{\ell} \left(\sum_{\substack{\mathcal{I} \subseteq \mathcal{L} \\ |\mathcal{I}| = k}} \prod_{i \in \mathcal{I}} p_i \prod_{j \in \mathcal{L} \setminus \mathcal{I}} (1 - p_j) \right), \tag{1}$$

where $\mathcal{L} = \{i_1, \ldots, i_\ell\} \subseteq \mathcal{N} = \{1, \ldots, n\}$ is the index set of $\mathcal{D}'$. As an important practical issue, notice that (1) is valid only for independent members to calculate the ensemble accuracy. The dependency of the members can be discovered further by, e.g., using different kinds of diversity measures (Hajdu et al. 2013).

Regarding ensemble-based systems, the standard task is to devise the most accurate ensemble from $\mathcal{D}$ for the given energy function. In this paper, we add a natural constraint of a bounded total cost to this optimization problem. That is, we have to maximize (1) under the cost constraint

$$\sum_{i \in \mathcal{L}} t_i \leq T, \tag{2}$$

where the total allowed cost $T \in \mathbb{R}_{>0}$ is a predefined constant. Consequently, we must focus on those subsets $\mathcal{L} \subseteq \mathcal{N}$ with cardinalities $|\mathcal{L}| = \ell \in \{1, \ldots, n\}$ for which (2) is fulfilled. Let $\mathcal{L}_0$ denote that index set of cardinality $|\mathcal{L}_0| = \ell_0$, where the global maximum ensemble accuracy is reached. The following lemma states that one can reach $\mathcal{L}_0$ calculating $q_\ell(\mathcal{L})$ for odd values of ℓ only, which results in more efficient computation, since not all the possible subsets should be checked.

Lemma 1 *If*

$$\max\left(q_\ell(\mathcal{L}) \mid \mathcal{L} \subseteq \mathcal{N}\right) = q_{\ell_0}(\mathcal{L}_0), \tag{3}$$

then ℓ_0 is odd.

Proof See Appendix 1 for the proof. □

The optimization task defined by the energy function (1) and the constraint (2) can be interpreted as a typical knapsack problem (Martello and Toth 1990). Such problems are known to be NP-hard; however, if the energy function is linear and/or separable for the p_i

-s, then a very efficient algorithmic solution can be given based on dynamic programming. However, if the energy lacks these properties, the currently available theoretical foundation is rather poor. As some specific examples, we were able to locate investigations of an exponential-type energy function (Klastorin 1990), and a remarkably restricted family of nonlinear and nonseparable ones (Sharkey et al. 2011). In Klastorin (1990), an approach based on calculus was made by representing the energy function by its Taylor series. Unfortunately, it has been revealed that dynamic programming can be applied efficiently only to at most the quadratic member of the series; thus, the remaining higher-order members had to be omitted. This compulsion suggests a large error term if this technique is attempted to be considered generally. Thus, to the best of our knowledge, there is a lack of theoretical instructions/general recommendations to solve knapsack problems in the case of complex energy functions. As our energy (1) is also nonlinear and nonseparable, we were highly motivated to develop a well-founded framework for efficient ensemble creation.

As our main contribution, in this paper, we propose a novel stochastic technique to solve knapsack problems with complex energy functions. Though the model is worked out in detail for (1) settled on the majority voting rule, it can be applied also to other energy functions. Our approach is based on the stochastic properties of the energy q_ℓ in (1), providing that we have some preliminary knowledge on which distribution its parameters p_i $(i = 1, \ldots, n)$ are coming from. We put a special focus on *beta* distributions that fit practical problems very well. In other words, we estimate the distribution of q_ℓ in terms of its mean and variance. This information can be efficiently incorporated as a stopping rule in stochastic search algorithms, as we demonstrate it e.g. for SA. The main idea here is to be able to stop building ensembles when we can expect that better ones can be found by low probability only.

As a common empirical approach to find the optimal ensemble, the usefulness p_i/t_i $(i = 1, \ldots, n)$ of the possible members are calculated. Then, as deterministic greedy methods, the ensemble is composed of forward/backward selection strategies (see, e.g., Kurz et al. 2013). Since the deterministic methods are less efficient—e.g., the greedy one is proven to have 50% accuracy for the simplest knapsack energy $\sum_{i=1}^{n} p_i$—popular stochastic search algorithms are considered instead, such as SA. As a further contribution, we introduce a novel stochastic search strategy, where the usefulness of the components is defined in a slightly more complex way to better fit the investigated energy; the stopping rule can be successfully applied in this approach as well. For the sake of completeness, we will start our theoretical investigation regarding the accuracy of the existing deterministic methods when a cost limitation is also applied.

3 Deterministic selection strategies

In this section, we address deterministic selection strategies to build an ensemble that has maximal system accuracy $q_{\ell_0}(\mathcal{L}_0)$, applying the cost limitation. However, since we have 2^n different subsets of elements of a pool of cardinality n, this selection task is known to be NP-hard. To overcome this issue, several selection approaches have been proposed. The common point of these strategies is that in general, they do not assume any knowledge on the proper determination of the classification performance $q_\ell(\mathcal{L})$; rather, they require only the ability to evaluate it. Moreover, to the best of our knowledge, strategies that consider the capability of individual feature accuracies to be modeled by drawing them from a probability distribution, as in our approach, have not yet been proposed.

Based on the above discussion, it seems to be natural to ascertain how the widely applied selection strategies work in our setup. The main difference in our case, in contrast to the general recommendations, is that now we can properly formulate the performance evaluation using the exact functional knowledge of q_ℓ. That is, we can characterize the behavior of the strategy with a strict analysis instead of the empirical tests generally applied.

We start our investigation with greedy selection approaches by discussing them via the forward selection strategy. Here, the most accurate item is selected and put in a subset S first. Then, from the remaining $n-1$ items, the component that maximizes the classification accuracy of the extended ensemble is moved to S. This procedure is then iteratively repeated; however, if the performance cannot be increased by adding a new component, then S is not extended and the selection stops. The first issue we address is to determine the largest possible error this strategy can lead to in our scenario.

Proposition 1 *The simple greedy forward selection strategy to build an ensemble that applies the majority voting-based rule has a maximum error rate* 1/2.

Proof For the proof, see Appendix 2. □

As seen from the proof, the error rate of 1/2 holds for the forward strategy independent of the time constraint. As a quantitative example, let $p_1 = 0.510$ and $p_2 = p_3 = p_4 = p_5 = 0.505$. With this setup, where $\mathcal{I}_k = \{1, \dots, k\}$, we have $q_1(\mathcal{I}_1) = p_1 = 0.5100$, $q_3(\mathcal{I}_3) = 0.5100$, and $q_5(\mathcal{I}_5) = 0.5112$, which shows that the greedy forward selection strategy is stuck at the single element ensemble, though a more accurate larger one could be found.

In addition to forward selection, its inverse variant, the backward selection strategy, is also popular. It puts all the components into an ensemble first, and in every selection step, leaves the worst one out to gain maximum ensemble accuracy. As a major difference from the forward strategy, backward selection is efficient in our case if the time constraint is irrelevant. Namely, either the removal of the worst items will lead to an increase in q_ℓ defined in (1), or the selection can be stopped without the risk of missing a more accurate ensemble.

However, if the time constraint applies, the same maximum error rate can be proved.

Proposition 2 *The simple greedy backward selection strategy considering the individual accuracy values to build an ensemble that applies the majority voting-based rule has a maximum error rate of* 1/2.

Proof Proposition 2 is proved in Appendix 3. □

Propositions 1 and 2 have shown the worst-case scenarios for the forward and backward selection strategies. However, the greedy approach was applied only regarding the accuracy values of the members, and their execution times were omitted. To consider both the accuracies and execution times of the algorithms in the ensemble pool $\mathcal{D} = \{D_1 = (p_1, t_1), D_2 = (p_2, t_2), \dots, D_n = (p_n, t_n)\}$, we consider their usefulness in the selection strategies, defined as

$$u_i = p_i / t_i, \quad i = 1, \dots, n, \tag{4}$$

which is a generally used definition to show the price-to-value ratio of an object. The composition of ensembles based on similar usefulness measures has also been efficient, e.g., in sensor networks (Kurz et al. 2013).

After the introduction of the usefulness (4), the first natural question to clarify is to investigate whether the validity of the error rate of the deterministic greedy forward and backward selection strategies operating with the usefulness measure holds. Through the following two statements, we will see that the 1/2 error rates remain valid for both greedy selection approaches.

Corollary 1 *Proposition 1 remains valid when the forward feature selection strategy operates on the usefulness. Namely, as a worst-case scenario, let* $t_1 = t_2 = \cdots = t_n = T/n$ *be the execution times in the example of the proof of Proposition 1 while keeping the same* $p_1, p_2, \ldots, p_n$ *values. Then, the selection strategy operates completely in the same way on the* $u_i = p_i/t_i$ *values* $(i = 1, \ldots, n)$ *as on the* p_i *ones, since the* t_i *values are equal. That is, the error rate is* 1/2 *in the same way.*

Proposition 3 *The simple greedy backward selection strategy considering the individual usefulness (4) to build an ensemble that applies the majority voting-based rule has a maximum error rate of* 1/2.

Proof The proof is provided in Appendix 4. □

We note the analogy between forward and backward selection strategies and approximation algorithms, which find approximate solutions to optimization problems, in particular NP-hard ones. With this respect the most common expectation is to have provable guarantees on the distance of the returned solution to the optimal one (Williamson and Shmoys 2011). With Propositions 1, 2, 3 and Corollary 1 we exactly address this expectation by giving the worst-case scenarios for these selection strategies. A wider theoretical characterization (regarding e.g. the expected error) would need exhaustive knowledge about the specific member accuracies and cost values. However, in the later chapters we will present many empirical results for these greedy approaches in several scenarios. These results also suggest that a deeper error analysis is expected to be interesting from mainly a theoretical point of view, because other existing and the proposed approaches show remarkably better performance.

The main problem with the above deterministic procedures is that they leave no opportunity to find better performing ensembles. Thus, we move on now to the more dynamic stochastic strategies. Keep in mind that since in our model the distribution of q will be estimated, in any of the selection strategies we can exploit this knowledge as a stopping rule. Namely, even for the deterministic approaches, we can check whether the accuracy of the extracted ensemble is already attractive or whether we should continue and search for a better one.

4 Stochastic search algorithms

As the deterministic selection strategies may have poor performance, we investigate stochastic algorithms to address our optimization problem. Such randomized algorithms, where randomization only affects the order of the internal executions, produce the same result on a given input, which can cause the same problem we have found for the deterministic ones. In case of Monte Carlo (MC) algorithms (Tempo and Ishii 2007), the result of the simulations might change, but they produce the correct result with a certain probability.

The accuracy of the MC approach depends on the number of simulations N; the larger N is, the more accurate the algorithm will be. It is important to know how many simulations are required to achieve the desired accuracy. The error of the estimate of the probability failure is found to be $u_{1-\alpha/2}\sqrt{(1-\mathrm{P}_f)/N\mathrm{P}_f}$, where $u_{1-\alpha/2}$ is the $1-\alpha/2$ quantile of the standard normal distribution, and P_f is the true value of the probability of failure.

Simulated annealing (SA), as a variant of the Metropolis algorithm, is composed of two main stochastic processes: generation and acceptance of solutions. SA is a general-purpose, serial search algorithm, whose solutions are close to the global extremum for an energy function within a polynomial upper bound for the computational time and are independent of the initial conditions.

To compare the MC method with SA for solving a knapsack problem, we applied simulations for that scenario in which the deterministic approaches failed to find the most accurate ensemble, that is, when $D_1 = (1-\beta, T)$, and $D_2 = D_3 = \ldots = D_n = (1/2+\varepsilon, T/n)$ with $0 < \beta < 1/2, 0 < \varepsilon < 1/2$. For this setup, we obtained that the precision of the MC method was only 11%, while SA found the most accurate ensemble in 96% of the simulations. Beyond SA, other pruning methods cited in the introduction are naturally based on stochastic methods.

Now, we introduce a novel search strategy that takes better advantage of our stochastic approach than, e.g., SA. This strategy builds ensembles using a randomized search technique and introduces a concept of usefulness for member selection, which better adapts to the ensemble energy than the classic one (4). Namely, in our proposed approach, the selection of the items for the ensemble is based on the efficiency of the members determined in the following way: for the i-th item with accuracy p_i and execution time t_i, the system accuracy $q(p_i, t_i)$ of the ensemble containing the maximal number of i-th items

$$q(p_i, t_i) = \sum_{k=\lfloor T/t_i \rfloor/2+1}^{\lfloor T/t_i \rfloor} \binom{\lfloor T/t_i \rfloor}{k} {p_i}^k (1-p_i)^{\lfloor T/t_i \rfloor - k} \tag{5}$$

characterizes the efficiency (usefulness) of the i-th item, instead of (4).

A greedy algorithm for an optimization problem always chooses the item that seems to be the most useful at that moment. In our selection method, a discrete random variable depending on the efficiency values of the remaining items is applied in each step to determine the probability of choosing an item from the remaining set to add to the ensemble. Namely, in the k-th selection step, if the items $i_1, \ldots, i_{k-1}$ are already in the ensemble, then the efficiency values $q^{(k-1)}(p_i, t_i)$ of the remaining items are updated to the maximum time of $T_k = T - \sum_{j=1}^{k-1} t_{i_j}$, where $q^{(0)}(p_i, t_i) = q(p_i, t_i)$ and $T_0 = T$.

The i-th item is selected as the next member of the ensemble with the following probability:

$$(\mathrm{P}_{ens})_i^{(k)} = \frac{q^{(k-1)}(p_i, t_i)}{\sum_j q^{(k-1)}(p_j, t_j)}, \tag{6}$$

where $i, j \in \mathcal{N} \backslash \{i_1, \ldots, i_{k-1}\}$. This discrete random variable reflects that the more efficient the item is, the more probable it is to be selected for the ensemble in the next step.

As a new contribution, we have incorporated these probabilities based on the newly introduced member efficiencies first to SA characterizing the acceptance probabilities by them. Same considerations apply to any other stochastic search methods. Besides SA and the genetic

algorithm (Goldberg 1989), these novel stochastic methods (denoted by SA+ and Genetic+) will be compared with our proposed method in the experimental analyses.

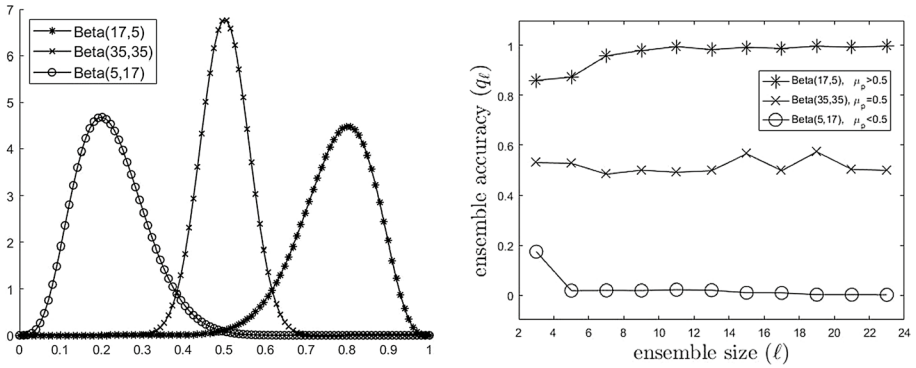
If $t_i > T_k$ for all $i \in \mathcal{N} \backslash \{i_1, \ldots, i_{k-1}\}$, then our stochastic process ends for the given search step since there is not enough remaining time for any items. Then, we restart the process to extract another ensemble in the next search step. As a formal description of our proposed stochastic search method SHErLoCk, see Algorithm 1; notice that we evaluate the accuracy of ensembles with odd cardinalities only as in Lemma 1. A very important issue regarding both our approach and other search methods (e.g. SA) is the exact definition of the number of search steps, that is, a meaningful STOP parameter—and also an escaping MAXSTEP one—for Algorithm 1. In our preliminary work (Hajdu et al. 2016), we have already tested the efficiency of our approach; however, we tested it empirically with an ad hoc stopping rule. Now, in the forthcoming sections, we present how the proper derivation of the stopping parameters (STOP and MAXSTEP) can be derived.

Algorithm 1 Proposed Stochastic searcH for EnsembLe Creation (SHErLoCk).

Input: Pool $\mathcal{D} = \{(p_i, t_i)\,,\ i = 1, \ldots, n\}$,
Total allowed time T,
Stochastic stopping value STOP,
Maximum search steps MAXSTEP.

Output: An ensemble MAXENS $\subseteq \mathcal{D}$ to maximize system accuracy (1) within time T as in (2).

STEP $\leftarrow 0$, MAXENS $\leftarrow \emptyset$, $q_{\ell_0} \leftarrow 0$
while STEP<MAXSTEP **do**
  $H \leftarrow \mathcal{D}$, ENS $\leftarrow \emptyset$, $T' \leftarrow T$, SP $\leftarrow \emptyset$
  while $\exists (p_j, t_j) \in H : t_j \leq T - \sum_{(p_k, t_k) \in \text{ENS}} t_k$ **do**
    $\forall (p_i, t_i) \in H$ calculate $q(p_i, t_i)$ by (5)
    $\forall (p_i, t_i) \in H$ calculate P_{ens_i} by (6) and $SP \leftarrow SP \cup \{\text{P}_{ens_i}\}$
    Select a (p_j, t_j) randomly from H by distribution SP
    if $t_j < T'$ **then**
      $ENS \leftarrow ENS \cup \{(p_j, t_j)\}$
      $H \leftarrow H \setminus \{(p_j, t_j)\}$
      $T' \leftarrow T' - t_j$
      if $mod(size(\text{ENS}), 2) = 1$ **then**
        Calculate $q_\ell(\text{ENS})$ by (1)
      end if
      if $q_{\ell_0} < q_\ell$ **then**
        $q_{\ell_0} \leftarrow q_\ell$, MAXENS $\leftarrow$ ENS
      end if
      if $q_{\ell_0} > STOP$ **then**
        return MAXENS
      end if
    end if
  end while
  STEP $\leftarrow$ STEP $+ 1$
end while
return MAXENS



(a) **(b)**

Fig. 1 Different sample $Beta(\alpha_p, \beta_p)$ distributions (**a**) and the convergence of ensemble accuracies for member accuracies coming from them (**b**)

5 Stochastic estimation of ensemble energy

We need to examine and characterize the behavior of q_ℓ in (1) to exploit these results to find and apply the proper stopping criteria in stochastic search methods.

Let $p \in [0, 1]$ be a random variable with mean μ_p and variance σ_p^2, where p_i $(i = 1, 2, \ldots, n)$ are independent and identically distributed according to p, i.e., a sample. Furthermore, let μ_{q_ℓ} and $\sigma_{q_\ell}^2$ denote the mean and variance of the ensemble accuracy q_ℓ, respectively. In this case, it is seen that $\mu_p \leq 1$ and a simple calculation shows that

$$\mu_{q_\ell} = \sum_{k=\lfloor \frac{\ell}{2} \rfloor + 1}^{\ell} \binom{\ell}{k} \mu_p^k \left(1 - \mu_p\right)^{\ell - k}. \tag{7}$$

The mean μ_{q_ℓ} is monotonic in ℓ, except for the case $\mu_p = 1/2$. Moreover, we get 0, 1/2 or 1 for the limit of μ_{q_ℓ} if $\ell \to \infty$ in case $\mu_p < 1/2$, $\mu_p = 1/2$, $\mu_p > 1/2$, respectively. As a demonstrative example, see Fig. 1 regarding the three possible accuracy limits (0, 1/2 or 1) described in (43) with respective $Beta(\alpha_p, \beta_p)$ distributions for p. The parameters of the beta distribution (α_p, β_p) can be chosen arbitrarily to fulfil the condition $\mu_p < 1/2$, $\mu_p = 1/2$, $\mu_p > 1/2$, respectively. Furthermore, the variance $\sigma_{q_\ell}^2$ can be expressed by the mean μ_p and variance σ_p^2, where its limit $\lim_{\ell \to \infty} \sigma_{q_\ell}^2 = 0$, if $\mu_p \neq 1/2$ and $s_T = \sigma_p^2 + \mu_p^2 \neq 1/2$ and 1, otherwise. For the exact formula for $\sigma_{q_\ell}^2$ and for the proof of these statements, see Lemma 2 in Appendix 5. Notice that the condition $\ell \to \infty$ naturally assumes the same for the pool size with $n \to \infty$.

Now, to devise a stochastic model, we start with checking the possible distributions of the member accuracy values p_i to estimate the ensemble accuracy. Then, we extend our model regarding this estimation by incorporating time information as well. Notice that the estimation of the ensemble accuracy will be exploited to derive a stopping rule for the ensemble selection process.

5.1 Estimation of the distribution of member accuracies

Among the various possibilities, we have found that the *beta* distribution is a very good choice to analyze the distribution of member accuracies. The main reason is that *beta* concentrates on the interval [0, 1], that is, it can exactly capture the domain for the smallest/largest accuracy. Moreover, the *beta* distribution is able to provide density functions of various shapes that often appear in practice. Thus, to start the formal description, let the variate p be distributed as $Beta(\alpha_p, \beta_p)$ with density

$$b(x;\alpha_p, \beta_p) = \frac{x^{\alpha_p - 1}(1-x)^{\beta_p - 1}}{B(\alpha_p, \beta_p)}, \tag{8}$$

where $B(\alpha_p, \beta_p) = \Gamma(\alpha_p)\Gamma(\beta_p)/\Gamma(\alpha_p + \beta_p)$. In this case,

$$\mu_p = \alpha_p/(\alpha_p + \beta_p), \tag{9}$$

and $\mu_p \in (1/2, 1)$ if and only if $\alpha_p > \beta_p$. If $\alpha_p = \beta_p$, then $\mu_p = 1/2$. In the case of $\alpha_p > \beta_p$, the mode is also greater than 1/2.

The mode is infinite if $\beta_p < 1$; therefore, we exclude this situation and we assume from now on that

$$1 < \beta_p < \alpha_p. \tag{10}$$

The variance of p is

$$\sigma_p^2 = \frac{\alpha_p \beta_p}{(\alpha_p + \beta_p)^2(\alpha_p + \beta_p + 1)}. \tag{11}$$

Since μ_{q_ℓ}, and $\sigma_{q_\ell}^2$ depend on μ_p, and σ_p^2 according to (7) and (44) respectively, one can calculate both of them explicitly. The convergence of μ_{q_ℓ} to 1 is fast if μ_p is close to 1, i.e., $\beta_p \ll \alpha_p$; for instance, if $\alpha_p = 17$, $\beta_p = 5$. Simulations show that the speed of the convergence of $\sigma_{q_\ell}^2$ is exponential; hence, the usual square-root law does not provide the Central Limit Theorem for q_ℓ.

In practice, we perform a *beta* fit on the p_i's ($i = 1, \ldots, n$). If a fit is found at least at the confidence level 0.95, we take the parameters α_p, β_p provided by the fit and calculate μ_p, σ_p^2 by (9) and (11), respectively. If the *beta* fit is rejected, then μ_p and σ_p^2 are estimated from the p_i's as the empirical mean and variance:

$$\mu_p = \frac{1}{n}\sum_{i=1}^{n} p_i, \quad \sigma_p^2 = \frac{1}{n-1}\sum_{i=1}^{n}(p_i - \mu_p)^2. \tag{12}$$

To simplify our further notation we do not indicate whether the mean and variance have been estimated from the fitted distribution or empirically.

5.2 Adding time constraints to the model

Now, we turn to the case when together with the item accuracy p_i, we consider its running time t_i as well. The common distribution of a random time is exponential, so let τ be an

exponential distribution with density $\lambda \exp(-\lambda t)$. If p is distributed as $Beta(\alpha_p, \beta_p)$, then with setting $\lambda = 1 - p$ for a given p, the distribution of λ becomes $Beta(\beta_p, \alpha_p)$.

This is a reasonable behavior of time because it is quite natural to assume that more accurate components require more resources such as a larger amount of computation times. On the other hand, the selection procedure becomes trivial, if, e.g., the time and accuracy are inversely proportional, since then the most accurate member is also the fastest one; therefore, it should be selected first by following this strategy for the remaining members until reaching the time limit. For some other possible simple accuracy–time relations, see our preliminary work (Hajdu et al. 2016).

For a given time constraint T, consider the random number ℓ_T such that

$$\sum_{j=0}^{\ell_T} \tau_j \leq T. \tag{13}$$

We provide an estimation for the expected size of the composed ensemble $\widehat{\ell_T} = \left\lceil T \frac{\beta_p - 1}{\alpha_p + \beta_p - 1} \right\rceil$ in Lemma 3 (see Appendix 6) and we incorporate this information into our stochastic characterization of q_ℓ. Till this point, we have assumed that p is distributed as *beta* to calculate $\widehat{\ell_T}$ by Lemma 3. If this is not the case, we consider the following simple and obvious calculation for the approximate number of ℓ under the time constraint T:

$$\widehat{\ell_T} = \left\lceil nT \Big/ \sum_{i=1}^{n} t_i \right\rceil = \lceil T/\bar{t} \rceil; \tag{14}$$

another alternative to derive $\widehat{\ell_T}$ in this case is discussed in Sect. 7. In either way it is derived, the value $\widehat{\ell_T}$ will be used in the stopping rule in our ensemble selection procedure; the proper details will be given next.

5.3 Stopping rule for ensemble selection

The procedure of finding $(\mathcal{L}_0, \ell_0)$ is a selection task that is NP-hard. We propose an algorithm such that we stop the selection when the value of $q_\ell(\mathcal{L})$ is sufficiently close to the possible maximum, which is not known. To be able to do so, we must give a proper stochastic characterization of q_ℓ by also settling on the calculation of μ_{q_ℓ} and $\sigma^2_{q_\ell}$ via Lemma 2. First, notice that the values of $q_\ell$are in (0, 1); indeed, it is positive and

$$q_\ell = \sum_{k=\lfloor \frac{\ell}{2} \rfloor + 1}^{\ell} \sum_{\substack{I \subseteq \mathcal{N} \\ |I| = k}} \prod_{i \in I} p_i \prod_{j \in \mathcal{N} \setminus I} (1 - p_j) < \prod_j (p_j + (1 - p_j)) = 1. \tag{15}$$

For the case when p_i's are *beta* distributed, the product of independent *beta* variates can be close to *beta* again (see Tang and Gupta 1984). We have also performed MC simulation and found that *beta* distributions fit q_ℓ particularly well, compared to, e.g., the gamma, normal, Weibull, and extreme-valued distributions. Specifically, though the *beta* behavior of q_ℓ was naturally more stable for *beta* distributed p_i's, the usual behavior of q_ℓ was also the same for non-*beta* p_i's.

Table 1 Probability values $\varrho_{q_{\hat{\ell}}}$ for stopping thresholds for different skewness coefficients γ

γ	$\varrho_{q_{\hat{\ell}}}$
$\gamma \leq 1$	0.6
$1 < \gamma \leq 2.5$	0.8
$2.5 < \gamma \leq 3.5$	0.9
$3.5 < \gamma$	0.95

Thus, to provide a description of the stochastic behavior of q, we consider the following strategy. With a primary assumption on the $Beta(\alpha_q, \beta_q)$ distribution of q_ℓ, we calculate α_q and β_q as

$$\alpha_q = \left(\frac{1-\mu_q}{\sigma_q^2} - \frac{1}{\mu_q} \right) \mu_q^2, \quad \beta_q = \alpha_q \left(\frac{1}{\mu_q} - 1 \right). \tag{16}$$

If time information is provided for the pool items, we calculate $\widehat{\ell_T}$ by Lemma 3, and as a simpler notation, we will write $\hat{\ell}$ from now on. If time information is not available, we will set $\hat{\ell} = n$.

Next, we decide whether q_ℓ should be considered as *beta* with requiring $1 < \beta_q < \alpha_q$ to be fulfilled to have a mode that is larger than 1/2 and finite. If this condition does not hold, we reject the *beta* behavior of q_ℓ, and based on simulations, we characterize it as a normal distribution and stop the search if

$$q_\ell \geq \kappa_{0.9} \sigma_{q_{\hat{\ell}}} / \sqrt{\hat{\ell}} + \mu_{q_{\hat{\ell}}} = \text{ STOP}\,, \tag{17}$$

where $\kappa_{0.9}$ is the 0.9 quantile of the standard normal distribution. Otherwise, when q_ℓ is considered *beta*, we calculate the mode ν of $Beta(\alpha_q, \beta_q)$ for q_ℓ as

$$\nu = \frac{\alpha_q - 1}{\alpha_q + \beta_q - 2}, \tag{18}$$

and the Pearson's first skewness coefficient as

$$\gamma = \frac{1-\nu}{\sigma_{q_{\hat{\ell}}}}. \tag{19}$$

Then, we use Table 1 to select the appropriate probability value $\varrho_{q_{\hat{\ell}}}$; the entries are determined by simulation in the case of $2 \leq \beta_q < \alpha_q$.

We stop the selection when the ensemble accuracy reaches the value of the inverse cumulative distribution $F^{-1}_{\alpha_q, \beta_q}(\varrho_{q_{\hat{\ell}}})$ of $Beta(\alpha_q, \beta_q)$ in the given probability, that is, when

$$q_\ell \geq F^{-1}_{\alpha_q, \beta_q}(\varrho_{q_{\hat{\ell}}}) = \text{ STOP}\,. \tag{20}$$

In either via (17) or (20), an estimation for the ensemble accuracy is gained; we obtain a STOP value to stop the stochastic search. However, there is some chance that STOP is not exceeded, though in our experiments it has never occurred. Thus, to avoid an infinite loop, we consider a maximum allowed step number MAXSTEP as an escaping stopping rule. Namely, to obtain MAXSTEP, we apply Stirling's approximation

$$\text{MAXSTEP} = \binom{n}{\widehat{\ell}} \sim n^{\widehat{\ell}}/\widehat{\ell}!, \tag{21}$$

assuming that $\widehat{\ell}/n \to 0$. This is a reasonable approach since $\widehat{\ell}$ is calculated according to Lemma 3 or (14). The formal description of our proposed ensemble selection method is enclosed in Algorithm 2.

Algorithm 2 Proposed Ensemble Creation Method.

Input: [NO-TIME]: Pool $\mathcal{D} = \{p_i\}_{i=1}^n$.

Input: [TIME]: Pool $\mathcal{D} = \{(p_i, t_i)\}_{i=1}^n$,
Total allowed time T.

Output: An ensemble MAXENS $\subseteq \mathcal{D}$ to maximize system accuracy (1) within time T as in (2).

1: Calculate the mean μ_p and std σ_p for $\{p_i\}_{i=1}^n$
by (9) and (11) (if a *beta* fits to p) or
empirically (if p is not *beta*) by (12)
2: **switch Input do**
3: **case** NO-TIME
4: $\widehat{\ell} \leftarrow n$
5: **case** TIME
6: Estimate # of members $\widehat{\ell}$ for T by
Lemma 3 if a *beta* fits to p, or by (14)
if p is not *beta*
7: Calculate $\mu_{q_{\widehat{\ell}}}$ by (7) and $\sigma^2_{q_{\widehat{\ell}}}$ by (44)
8: Calculate α_q, β_q by (16)
9: **if** $1 < \beta_q < \alpha_q$ **then**
10: Calculate cdf. F_{α_q,β_q}, ν, γ, $\varrho_{q_{\widehat{\ell}}}$ by (18), (19)
and Table 1, and adjust STOP with (20)
11: **else**
12: Adjust STOP with (17)
13: **end if**
14: Calculate MAXSTEP by (21)
15: **switch** Input **do**
16: **case** NO-TIME
17: Compose ensemble by SA/Genetic/Pruning
using STOP for the stopping rule
18: **case** TIME
19: Compose ensemble either by Algorithm 1
or SA/SA+/Genetic/Genetic+/Pruning using
STOP for the stopping rule

Notice that neither Algorithm 1 nor Algorithm 2 considers freely adjustable parameters beyond the input (p_i, t_i) pairs and the total allowed time T. The derivation of all estimated distribution parameters and the stopping related ones are properly referred to in the bodies of the algorithms. If no time condition is provided then any preferred unconstrained algorithm (e.g. the SA, Genetic Goldberg 1989 or Pruning Martinez-Munoz and Suarez 2007 one) can be used as handled by line 17 in Algorithm 2. Similarly, the output (line 19) of Algorithm 2 is the composed ensemble found by either our proposed method or any other preferred one again. Either time condition is provided or not, all the ensemble creator approaches can take advantage of the calculated stopping parameter STOP.

Before providing our detailed empirical results in Sect. 6, in Table 2 we summarize our findings for Algorithm 2 on simulations. Namely, in two respective demonstrative tests with $i = 1, \ldots, 30$ and $i = 1, \ldots, 100$, we have generated the p_i's to come from the same

Table 2 Result of Algorithm 2 on simulations

Search method	Ensemble accuracy		Comp. time (secs)	
	MAXSTEP	STOP	MAXSTEP	STOP
SHErLoCk (n=30)	99.56%	99.39%	60.03	0.08
SA (n=30)	98.97%	98.91%	87.40	0.30
SHErLoCk (n=100)	99.66%	99.61%	294.58	1.54
SA (n=100)	99.38%	99.37%	638.39	1.58

example $Beta(17, 5)$ as before and the execution times t_i from conditional exponential distributions with parameters $\lambda = 1 - p_i$. The time constraint T was set in seconds to 30% of the total time $\sum_{i=1}^{30} t_i$ for the first, and 20% of $\sum_{i=1}^{100} t_i$ for the second test. Both tests were repeated 100 times, and we have taken the averages of the obtained precisions. The parameters of the beta distribution for the simulations can be chosen arbitrarily to fulfil the condition $1 < \beta_p < \alpha_p$ derived in Sect. 5.1. As our primary aim, we have checked whether the stopping rule of the stochastic search indeed led to a reasonable computational gain. For the sake of completeness, in Table 2 we have also shown the results for these simulated pairs (p_i, t_i) regarding letting the search continue in the long run (stopped by MAXSTEP), though in each of our tests, the STOP value has been exceeded much earlier. Secondarily, we have compared SA with our selection method SHErLoCk given in Algorithm 1.

For Table 2, we can conclude that applying our stopping rule by using STOP saved considerable computational time compared with the exhaustive search that culminated by stopping it with MAXSTEP with a negligible drop in accuracy. Moreover, our approach has found efficient ensembles quicker than SA. These impressions have also been confirmed by the empirical evaluations on real data described in the next section.

6 Empirical analysis

In this section, we demonstrate the efficiency of our models through an exhaustive experimental test on publicly available data. Our first experiment considers the possibility of organizing competing approaches with different accuracies into an ensemble. In this scenario, accuracy values correspond to final scores of participants of Kaggle[3] challenges without cost/time information provided. Our second setup for ensemble creation considers machine learning-based binary classifiers as possible members; the performance evaluation is performed on several UCI Machine Learning Repository (Dheeru and Karra Taniskidou 2017) datasets with the training times considered as costs.

6.1 Kaggle challenges

Kaggle is an open online platform for predictive modeling and analytics competitions with the aim of solving real-world machine learning problems provided by companies or users. The main idea behind this crowd-sourcing approach is that a countless number of different strategies might exist to solve a specific task, and it is not possible to know beforehand

[3] www.kaggle.com.

Table 3 Ensemble accuracies on the Kaggle datasets found by simulated annealing (SA)

Dataset	Ensemble accuracy		Computational time (secs)	
Name	MAXSTEP	STOP	MAXSTEP	STOP
Diabetic Retinopathy Detection	94.34%	93.19%	194.12	1.31
DonorsChoose.org Application Screening	94.78%	91.96%	206.89	1.67
Statoil/C-CORE Iceberg Classifier Challenge	88.42%	87.76%	191.91	2.23
WSDM - KKBox's Churn Prediction Challenge	96.96%	96.32%	203.88	1.45
Porto Seguro's Safe Driver Prediction	92.99%	89.98%	214.28	1.95

which one is the most effective. Though primarily only the scores of the participating algorithms can be gathered from the Kaggle site, as a possible future direction, we are curious regarding whether creating ensembles from the various strategies could lead to an improvement regarding the desired task.

Not all the Kaggle competitions are suitable to test our models since in the current content, we focus on majority voting-based ensemble creation. Consequently, we have collected only such competitions and corresponding scores where majority voting-based aggregation could take place. More precisely, we have restricted our focus only to such competition metrics based on which majority voting can be realized. Such metrics include quadratic weighted kappa, area under the ROC curve (AUC), log loss, normalized Gini coefficient. For concrete competitions where these metrics were applied, we analyze the following ones: Diabetic Retinopathy Detection,[4] DonorsChoose.org Application Screening,[5] Statoil/C-CORE Iceberg Classifier Challenge,[6] WSDM - KKBox's Churn Prediction Challenge,[7] and Porto Seguro's Safe Driver Prediction.[8]

For our analytics, on the one hand it is interesting to observe the distribution of the final score of the competitors, which is often affected by the volume of the prize money offered to the winner. Moreover, accuracy measurement is usually scaled to the interval [0, 1], with 0 for the worst and 1 for the perfect performance, which allows us to test our results regarding the *beta* distributions. As a drawback of Kaggle data, access to the resource constraints corresponding to the competing algorithms (e.g., training/execution times) is rather limited; such data are provided for only a few competitions, primarily in terms of execution time interval.

Thus, to summarize our experimental setup, we interpret the competing solutions of a Kaggle challenge as the pool $\{D_1, D_2, \ldots, D_n\}$, where the score of D_i is used for the accuracy term $p_i \in [0, 1]$ in our model. Then, we apply a *beta* fit for each investigated challenge to determine whether a *beta* distribution fits the corresponding scores or not. If the test is rejected, we can still use the estimation for the joint behavior q using (12) and (14). If the *beta* test is accepted, we can also apply our corresponding results using (9), (11), and Lemma 3. Notice that reliably fitting a model for the scores of the competitors might lead

[4] www.kaggle.com/c/diabetic-retinopathy-detection.
[5] www.kaggle.com/c/donorschoose-application-screening.
[6] www.kaggle.com/c/statoil-iceberg-classifier-challenge.
[7] www.kaggle.com/c/kkbox-churn-prediction-challenge.
[8] www.kaggle.com/c/porto-seguro-safe-driver-prediction/data.

to a better insight of the true behavior of the data of the given field, also for the established expectations there.

As observed from Table 3, SA was able to stop much earlier with a slight loss in accuracy using the suggested stopping rule (STOP) in finding the optimal ensemble. Our approach SHErLoCk given in Algorithm 1 has been excluded from this analysis since no cost information was available. Though for the lack of cost information our stochastics-based results can be applied only partially to Kaggle challenges with distribution fitting and suggesting stopping criterion accordingly, we were highly motivated to include this platform as well. Kaggle collects a huge number of different approaches—sometimes also with available implementations—for the same task, so is an excellent platform to create ensembles. Moreover, several accuracy measures considered in these challenges are just completely suitable for stochastic analysis, just like in our approach. If resource information is also provided in the future for the submitted solutions, then our corresponding results also become applicable.

6.2 Binary classification problems

The UCI Machine Learning Repository (Dheeru and Karra Taniskidou 2017) is a popular platform to test the performances of machine learning-based approaches, primarily for classification purposes. A large number of datasets are made publicly available here among which our models can be tested on binary classification ones. That is, in this experiment, the members $D_1, D_2, \ldots, D_n$ of a pool for ensemble creation are interpreted as binary classifiers, whose outputs can be aggregated by the majority voting rule. Using the ground truth supplied with the datasets, the accuracy term $p_i \in [0, 1]$ stands for the individual classification accuracy of D_i.

The number of commonly applied classifiers is relatively low; therefore to increase the cardinality of the pool, we have also considered a synthetic approach in a similar way to Cavalcanti et al. (2016). Namely, we have trained the same base classifier on different training datasets, by which we can synthesize several "different" classifiers. Naturally, this method is able to provide more independent classifiers only if the base classifier is unstable, i.e., minor changes in the training set can lead to major changes in the classifier output; such an unstable classifier is, for example, the perceptron one.

To summarize our experimental setup for UCI binary classification problems, we have considered base classifiers perceptron (Freund and Schapire 1999), decision tree (Quinlan 1986), Levenberg–Marquardt feedforward neural network (Suratgar et al. 2005), random neural network (Timotheou 2010), and discriminative restricted Boltzmann machine classifier (Larochelle and Bengio 2008) for the UCI datasets MAGIC Gamma Telescope, HIGGS, EEG Eye State, Musk (Version 2), Spambase, Breast Cancer Wisconsin, Mushroom, Gisette and Adult; datasets of large cardinalities were selected to be able to train synthetic variants of base classifiers on different subsets. To check our models for different numbers of possible ensemble members, the respective pool sizes were set to $n = 30$ and $n = 100$; the necessary number of classifiers has been reached via synthesizing the base classifiers with training them on different subsets of the training part of the given datasets. In contrast to the Kaggle challenges, in these experiments we were able to retrieve meaningful cost information to devise a knapsack scenario. Namely, for a classifier D_i, its training time was adjusted as its cost t_i in our model. Notice that for even the same classifier, it was possible to obtain different t_i values with training its synthetic variants on datasets of different sizes. Using this time information, for the estimated size $\hat{\ell}$ of the optimal

ensemble, we could use Lemma 3 for $n = 30$, while (14) for the case $n = 100$. This is one of the reasons why we set the different pool sizes to $n = 30$ and $n = 100$. Another point is to get sufficiently large search spaces to show the efficiency of our proposed method. To choose a pool size greater than 100 is rather unrealistic and results in a very time-demanding problem.

We compare the performance of the proposed search strategy (SHErLoCk) with SA, SA+, Genetic (Goldberg 1989), Genetic+, Pruning (Martinez-Munoz and Suarez 2007) on binary classification problems of UCI datasets using an ensemble pool of $n = 30$ and $n = 100$ classifiers, respectively. As clearly visible from Tables 4 and 5, our stochastic search strategy SHErLoCk described in Algorithm 1 was reasonably faster than the other ones and slightly dominant in ensemble accuracy found by the different approaches presented in the tables, as well. Moreover, it can be observed again that applying the stopping rule with the threshold STOP led to an enormous computational advantage for either search strategies with only a small drop in accuracy. For the sake of completeness, we have also included the forward and backward selection techniques. As it can be observed from the tables, these deterministic techniques are naturally quicker than the others; however, their accuracies are reasonably low as well. Notice that, as deterministic techniques it is meaningless to limit the search time for them with either MAXSTEP or STOP. Moreover, as for their 50% worst-case accuracy proved in Sect. 2 we can see a better performance in our experiments.

As for evaluating the accuracy of the ensembles of the trained classifiers, we have followed the following protocol. The individual classifiers have been trained according to the common guidelines with splitting all the datasets to training, cross-validation, and testing parts. Then the individual accuracies p_i have been determined as their performances on the test set (30% randomly selected parts of the datasets) since these figures are available only for the test sets regarding the Kaggle competitors. The ensemble accuracies gained by all the methods are calculated by aggregating the outputs of the trained classifiers on the test part of the publicly available datasets by applying the classic majority voting rule without any further training processes. This is the reason why we have presented the experimental results (ensemble accuracies and computational times) for the test parts for Kaggle and UCI, as well. The official splitting is available only for the Kaggle datasets, but not for the UCI ones. Thus, for possible future comparability, we give the results for the whole UCI datasets in Tables 7 and 8 in Appendix 7, as well. We have found a small drop or a small rise regarding each ensemble creator method's performance on the whole dataset, so the trends shown in Tables 4 and 5 are not affected. The computational times are naturally the same for both evaluation protocols. For further comparison of the performance of the proposed search strategy (SHErLoCk) with SA, SA+, Genetic, Genetic+, Pruning, Bayesian signed-rank tests (Benavoli et al. 2017) were applied. The results of the tests for training time as cost on UCI test sets using an ensemble pool of $n = 30$ and $n = 100$ classifiers are enclosed in Tables 9 and 10 in Appendix 7, respectively. The signed-rank test returns three probability values from which we consider $P(\textit{practical equivalence})$ and $P(\textit{Algorithm}1 > \textit{Algorithm}2)$ describing the equivalence of the two compared algorithms and the dominance of Algorithm1 over Algorithm2, respectively. The results show that the accuracy of the two compared classifiers are practically different in each case (the largest value for equivalence was 0.0132) and the proposed algorithm SHErLoCk outperformed the other search strategies with high probability in most of the cases.

We have also checked how the ensemble accuracies found by the different approaches increased regarding the elapsed time during the search. Notice that, the accuracies

Table 4 Comparing the proposed search strategy (SHErLoCk) with other selection methods regarding ensemble accuracy and computational time on binary classification problems of the UCI test subsets using an ensemble pool of $n = 30$ classifiers and training time as cost

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
		Method									
Ensemble accuracy	MAXSTEP	SHErLoCk	93.11%	96.81%	75.33%	96.02%	98.19%	98.37%	99.52%	93.22%	88.03%
		SA+	93.56%	95.98%	75.03%	96.02%	98.23%	96.91%	99.17%	92.72%	86.19%
		SA	93.08%	96.09%	74.33%	95.99%	98.23%	96.90%	98.36%	93.22%	86.92%
		Genetic+	92.97%	94.23%	75.39%	96.12%	97.19%	96.63%	99.03%	92.01%	87.51%
		Genetic	93.08%	95.88%	75.77%	96.35%	98.48%	95.73%	99.09%	92.49%	87.18%
		Pruning	92.03%	96.21%	74.68%	95.01%	98.26%	96.08%	98.79%	91.06%	86.72%
	STOP	SHErLoCk	92.47%	95.04%	75.41%	94.04%	99.33%	96.81%	98.47%	92.24%	86.88%
		SA+	92.43%	95.51%	75.14%	94.43%	98.59%	96.45%	99.18%	92.24%	86.88%
		SA	92.35%	94.82%	74.39%	93.78%	99.04%	96.84%	98.00%	92.17%	86.64%
		Genetic+	92.41%	94.76%	74.89%	95.16%	98.98%	95.74%	97.71%	92.07%	86.81%
		Genetic	92.84%	95.22%	73.70%	94.53%	99.03%	96.43%	97.99%	91.95%	85.93%
		Pruning	91.88%	94.76%	72.91%	95.39%	97.56%	96.08%	97.51%	92.17%	86.42%
	DET	Forward	88.11%	94.78%	72.22%	94.17%	97.08%	95.12%	98.54%	90.21%	82.27%
		Backward	88.87%	94.78%	71.52%	94.17%	97.08%	95.07%	98.54%	90.21%	81.49%

Table 4 (continued)

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
Computational time (secs)	MAXSTEP	SHErLoCk	46.47	37.90	49.46	30.20	48.89	32.19	37.06	51.45	34.73
		SA+	121.9	100.98	128.90	145.57	108.21	101.73	106.13	132.43	96.51
		SA	79.06	80.12	93.47	71.39	67.26	77.25	60.19	95.98	72.28
		Genetic+	63.42	68.03	59.57	61.07	74.02	69.09	89.67	67.03	69.08
		Genetic	46.08	48.92	53.22	58.16	51.23	48.56	57.99	64.42	49.18
		Pruning	125.84	107.08	115.42	129.45	93.06	90.98	103.54	137.08	151.08
	STOP	SHErLoCk	0.98	0.39	0.93	0.38	0.92	0.74	0.39	0.43	0.28
		SA+	8.41	4.67	6.73	8.47	7.33	5.09	8.79	5.88	6.11
		SA	8.78	6.92	4.56	7.93	8.59	9.48	6.95	5.89	7.99
		Genetic+	5.53	5.96	6.45	9.97	5.15	7.56	6.93	7.91	5.69
		Genetic	6.78	6.23	6.49	10.98	8.52	10.43	6.02	5.62	8.43
		Pruning	13.61	12.21	14.91	18.85	11.07	12.98	14.03	13.57	12.09
	DET	Forward	0.28	0.19	0.41	0.29	0.28	0.38	0.46	0.49	0.73
		Backward	0.21	0.37	0.43	0.42	0.31	0.84	0.35	0.25	0.23

Table 5 Comparing the proposed search strategy (SHErLoCk) with other selection methods regarding ensemble accuracy and computational time on binary classification problems of UCI test subsets using an ensemble pool of $n = 100$ classifiers and training time as cost

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
		Method									
Ensemble accuracy	MAXSTEP	SHErLoCk	95.06%	97.18%	76.66%	95.85%	99.00%	99.06%	99.98%	93.86%	87.94%
		SA+	94.93%	96.45%	76.08%	96.01%	99.08%	98.13%	99.96%	93.22%	87.29%
		SA	95.09%	96.89%	76.13%	95.91%	98.83%	98.81%	99.64%	92.88%	87.04%
		Genetic+	95.09%	96.49%	76.48%	96.15%	99.07%	98.87%	99.80%	93.28%	88.29%
		Genetic	95.91%	97.02%	76.27%	96.39%	99.16%	98.51%	99.64%	92.97%	87.93%
		Pruning	94.76%	95.99%	75.39%	95.34%	98.95%	98.09%	99.27%	92.66%	86.95%
	STOP	SHErLoCk	94.27%	95.71%	75.89%	94.79%	99.42%	98.25%	99.63%	93.02%	87.18%
		SA+	93.91%	95.74%	76.27%	95.19%	99.58%	97.19%	99.17%	93.21%	86.61%
		SA	94.11%	95.89%	76.27%	95.23%	99.60%	97.28%	99.52%	92.49%	86.77%
		Genetic+	94.27%	95.59%	75.69%	95.08%	99.34%	97.93%	99.71%	93.05%	86.63%
		Genetic	93.91%	95.99%	76.04%	95.46%	99.26%	98.08%	99.63%	92.93%	86.90%
		Pruning	93.72%	95.80%	75.63%	94.07%	99.09%	97.82%	99.04%	92.71%	86.59%
	DET	Forward	90.68%	95.02%	74.13%	93.01%	98.42%	95.61%	98.92%	90.92%	85.36%
		Backward	90.82%	95.02%	74.19%	93.01%	98.42%	95.08%	98.91%	90.92%	85.83%

Table 5 (continued)

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
Computational time (secs)	MAXSTEP	SHErLoCk	194.12	206.89	191.91	203.88	214.28	201.03	186.67	159.32	178.01
		SA+	349.62	290.89	390.82	278.56	253.59	375.71	313.56	301.25	311.87
		SA	251.02	291.13	269.22	278.39	269.59	228.44	286.23	259.92	258.62
		Genetic+	305.26	298.34	289.37	301.26	324.12	256.67	289.44	338.98	381.55
		Genetic	226.05	301.36	197.57	239.79	223.19	267.24	210.63	231.92	290.67
		Pruning	354.23	301.21	409.22	354.59	356.18	321.82	402.34	441.23	455.63
	STOP	SHErLoCk	2.32	3.41	2.11	2.08	1.56	2.33	2.49	1.73	2.45
		SA+	13.31	14.67	12.23	12.45	14.95	13.97	12.88	10.12	14.81
		SA	13.39	12.58	16.14	15.55	12.04	16.94	11.76	12.35	13.55
		Genetic+	12.56	12.61	15.95	13.37	13.49	16.68	14.88	13.14	12.59
		Genetic	13.53	13.58	12.88	15.47	11.51	13.67	12.02	11.27	12.28
		Pruning	19.41	11.78	17.32	36.36	21.69	31.55	22.92	15.34	19.53
	DET	Forward	0.59	0.72	0.81	0.92	0.69	0.89	0.93	0.96	0.94
		Backward	0.71	0.78	0.99	0.85	0.76	0.88	0.95	0.81	0.99

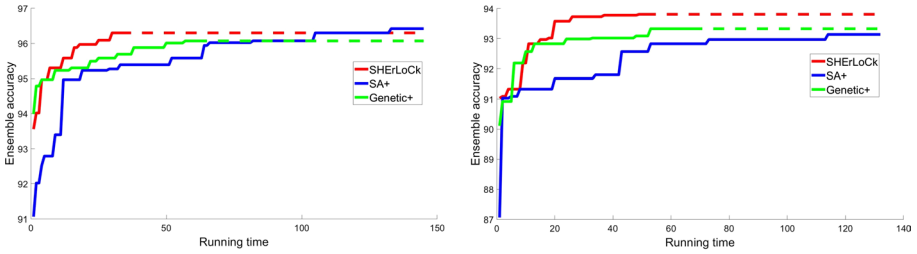


Fig. 2 The change of ensemble accuracy regarding the elapsed search time of the best performing approaches on the datasets EEG (**a**), and Gisette (**b**)

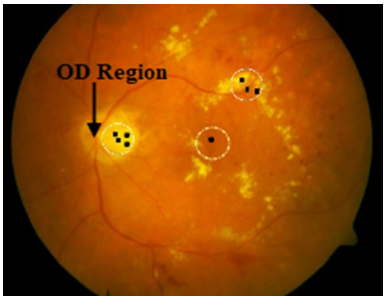


Fig. 3 Successful OD detection with the same number of correct/false ensemble member responses

indeed have a monotonously increasing trend, since at each search step we store the best performing ensemble for all the approaches till the stopping condition is met. Our findings are depicted in Fig. 2 (for $n = 30$) according to SHErLoCk, SA+, Genetic+ as best performing algorithms with indicating also the time points with dashed lines when the respective methods were stopped. For this demonstrative analysis we have selected the two datasets EEG/Gisette requiring the largest/smallest computational times. For each of the analyzed approaches we can see sudden jumps coming from their stochastic behaviours.

6.3 Optic disc detection

The majority voting rule can be applied in a problem to aggregate the outputs of single object detectors in the spatial domain (Hajdu et al. 2013); the votes of the members are given in terms of single pixels as candidates for the centroid of the desired object. In this extension, the shape of the desired object defines a geometric constraint, which should be met by the votes that can be aggregated. In Hajdu et al. (2013), our practical example relates to the detection of a disc-like anatomical component, namely the optic disc (OD) in retinal images. Here, the votes are required to fall inside a disc of diameter d_{OD} to vote together. As more false regions are possible to be formed, the correct decision can be made even if the true votes are not in the majority, as in Fig. 3. The geometric restriction transforms (1) to the following form:

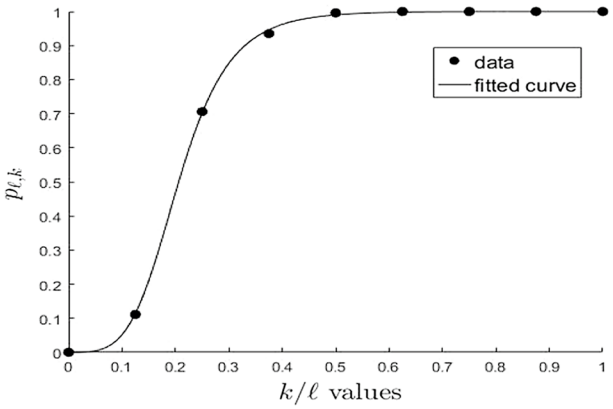


Fig. 4 Determining the constrained majority voting probabilities $p_{\ell,k}$ for our OD detector ensemble

Table 6 Comparing SA with the proposed search strategy SHErLoCk on the OD detection problem

Search method	Ensemble accuracy STOP	Comp. time (secs) STOP
SHErLoCk	99.45%	0.07
SA	99.43%	0.16

$$q_\ell(\mathcal{L}) = \sum_{k=0}^{\ell} p_{\ell,k} \left(\sum_{\substack{\mathcal{I} \subseteq \mathcal{L} \\ |\mathcal{I}|=k}} \prod_{i \in \mathcal{I}} p_i \prod_{j \in \mathcal{L} \setminus \mathcal{I}} (1 - p_j) \right). \tag{22}$$

In (22), the terms $p_{\ell,k}$ describe the probability that a correct decision is made by supposing that we have k correct votes out of ℓ. For the terms $p_{\ell,k}\,(k = 0, 1, \ldots, \ell)$, in general, we have that $0 \leq p_{\ell,0} \leq p_{\ell,1} \leq \cdots \leq p_{\ell,\ell} \leq 1$.

In our experiments, the pool consists of eight OD detector algorithms with the following accuracy and running time values: $\{(p_i, t_i)\}_{i=1}^{8} = \{(0.220, 31), (0.304, 38), (0.319, 34), \}$ $\{(0.643, 69), (0.754, 11), (0.765, 7), (0.958, 21), (0.976, 90)\}$ with $\sum_{i=1}^{8} t_i = 301$ secs. As we have mentioned earlier we can consider any resource type regarding the parameters t_i. For instance, we have considered the running times of the member algorithms here since some of them are not machine learning based ones. We can apply our theoretical foundation with some slight modifications to solve the same kind of knapsack problem for the variant (22), transforming the model to reflect the multiplication with the terms $p_{\ell,k}$.

We have empirically derived the values $p_{8,k} = \{0, 0.11, 0.70, 0.93, 0.99, 1.00, 1.00, 1.00, 1.00\}$ for (22) in our task. To adopt our approach by following the logic of Algorithm 2, we need to determine a STOP value for the search based on μ_p and σ_p (calculated by (12)), and $\widehat{\ell}$ (calculated by (14)). However, since now the energy function is transformed by the terms $p_{\ell,k}$ in (22), we must borrow the corresponding theoretical results from Tiba et al. (2019) to derive the mean $\mu_{q_{\widehat{\ell}}}$ instead of (7) proposed in Algorithm 2. Accordingly, we had to find a continuous function $\mathcal{F}$ that fit to the values $p_{\ell,k}$, which was evaluated by regression and resulted in $\mathcal{F}(x) = b/(b + x^a/(1-x)^a)$ with $a = -3.43$ and

$b = 101.7$, as also plotted in Fig. 4. Now, by using Theorem 1 from Tiba et al. (2019), we have gained $\mu_{q_{\hat{\ell}}} = \mathcal{F}(\mu_p)$.

For our experiment to search for the best ensemble, we have set the time constraint to be 80% of the total running time, with $T = 4(\sum_{i=1}^{8} t_i)/5$. For this setup, we could estimate $\widehat{\ell} = 7$ and $\mu_{q_{\hat{\ell}}} = 0.969$ for the expected ensemble size and mean accuracy, respectively. Then, these values have been considered for Algorithm 2 to demonstrate the performance of our stochastic search method SHErLoCk with SA. As shown in Table 6, our search strategy outperformed SA also for the object detection problem both in accuracy and computational time. Because of the smallness of this setup, we have omitted a full experimental evaluation.

7 Discussion

For the approximate number $\widehat{\ell_T}$ of the ensemble size, we have considered (14) when the member accuracy p is not a *beta* distribution. As an alternative notice that it is known that for a given λ, T and an independent exponential distributed τ_j, ℓ_T is distributed as Poisson with parameter λT. We can use Lemma 3 and conclude that for a starting size of the ensemble, one may choose $\widehat{\ell_T}$ such that the remaining possible values are beyond the 5% error. It follows that we apply formula either

$$\mathrm{P}\left(\ell_T > m_{0.05}\right) = 0.05, \tag{23}$$

where $m_{0.05}$ is the upper quantile of the Poisson distribution with parameter $T/\sum_{i=1}^{n} p_i$, or use the normal approximation to the Poisson distribution

$$\frac{\ell_T - T/\sum_{i=1}^{n} p_i - 0.5}{\sqrt{T/\sum_{i=1}^{n} t_i}} > 1.64, \tag{24}$$

which provides us the inequality

$$\ell_T > \frac{T}{\sum_{i=1}^{n} t_i} + 0.5 + 1.64\sqrt{\frac{T}{\sum_{i=1}^{n} t_i}} = \widehat{\ell_T}. \tag{25}$$

In our experiments we have used (14) instead of (25) to obtain $\widehat{\ell_T}$, since the latter provided slightly too large estimated size values. However, for other scenarios, it might be worthwhile to try (25) as well.

As some additional arguments, we call attention to the following issues regarding those elements of our approach that might need special care or can be adjusted differently in other scenarios:

- We have assumed independent member accuracy behavior, providing solid estimation power in our tests. However, in the case of strong member dependencies, deeper discovery of the joint behavior might be needed.
- Stirling's approximation considered in (21) may provide values that are too small for the parameter MAXSTEP in the case of small pools. Since this is an escape parameter, a sufficiently large value should be selected in such cases instead. Note that on the datasets we

examined, we found that it is not worth going above 15 000 iterations, because the ensembles' accuracies improved only slightly with higher step counts.
– The constraint we consider first in the knapsack problem corresponds to the training time; however, any other type of resources could be studied. For instance, we have examined the testing time as cost on the UCI dataset, as well. Tables 11 and 12 show the results (ensemble accuracy and computational time) of the proposed search strategy (SHErLoCk) compared with other selection methods using an ensemble pool of $n = 30$ and $n = 100$ classifiers, respectively. Similar conclusions can be drawn regarding the comparison of the performances of the methods that we have found for the training time.
– The time profile $\lambda = 1 - p$ in Sect. 5.2 is suited to our data; however, any other relationship between the member accuracy and time can be considered. In case other non-time-based resources are examined in the constraint, the exponential distribution can be changed accordingly. Nevertheless, the similar derivation of the estimation of the ensemble accuracy might be slightly more laborious depending on the selected distribution.
– In (17), we have used a one-tailed (left-side) hypothesis since q_ℓ was close to 1. However, if it is not that close to 1, a two-tailed hypothesis can be meaningful as well. Furthermore, if μ_q is even smaller (say 0.7), then we can search above this mean by considering a right-side hypothesis.

Natural generalizations of the presented results regard multiple objectives and multiclass classification. Within our framework, multiple constraints lead to a multiply-constrained knapsack problem, which can be handled e.g. with merging the constraints into a single one (Pisinger 1995) to make our approach directly applicable. As for multiclass classification, notice that our motivating example given in Sect. 6.3 has already some similarities regarding multiclass problems, since the geometric constraint intuitively describes such a type of task. To realize the classic setup, we have to consider multinomial distribution instead of the binomial one to handle majority voting in the corresponding classification problem. These changes are quite obvious; however, since they raise serious theoretical challenges over the techniques we used in this work, they need a dedicated future research beyond the current scope. Similarly, a complexity analysis of our approach is a natural issue; however, its stochastic nature makes this derivation really hard like in Dzahini (2020).

Appendix 1: Proof of Lemma 1

Proof Consider a subset $\mathcal{K}$ when $\mathcal{K} = \{1, 2, \ldots, 2\ell\}$ (otherwise we can renumerate p_i). We have

$$\begin{aligned} q_{2\ell}(\mathcal{K}) &= \sum_{k=\ell+1}^{2\ell} \sum_{\substack{I \subseteq \mathcal{K} \\ |I| = k}} \prod_{i \in I} p_i \prod_{j \in \mathcal{K} \setminus I} (1 - p_j) \\ &= \sum_{k=\ell+1}^{2\ell} \sum_{\substack{I \subseteq \mathcal{K} \\ |I| = k}} Q_{2\ell,k}(\mathcal{K}, I), \end{aligned} \tag{26}$$

where

$$Q_{2\ell,k}(\mathcal{K},I)=\prod_{i\in I}p_i\prod_{j\in\mathcal{K}\setminus I}\big(1-p_j\big), \tag{27}$$

that is, we consider a subset $\mathcal{K}\subseteq\mathcal{N}$ with $|\mathcal{K}|=2\ell$, and $Q_{2\ell,k}(\mathcal{K},I)$ is calculated for an index set $I\subseteq\mathcal{K}$ with $|I|=k$. Now, choose an index a from the set $\mathcal{N}\backslash\mathcal{K}$, i.e., $a>2\ell$, and obtain

$$Q_{2\ell,k}(\mathcal{K},I)=Q_{2\ell,k}(\mathcal{K},I)p_a+Q_{2\ell,k}(\mathcal{K},I)\big(1-p_a\big). \tag{28}$$

The term $Q_{2\ell,k}(\mathcal{K},I)p_a=Q_{2\ell+1,k+1}(\{\mathcal{K},a\},\{I,a\})$ and $Q_{2\ell,k}(\mathcal{K},I)\big(1-p_a\big)=Q_{2\ell+1,k}(\{\mathcal{K},a\},I)$; therefore,

$$\begin{aligned}q_{2\ell}(\mathcal{K})&=\sum_{k=\ell+1}^{2\ell}\sum_{\substack{I\subseteq\mathcal{K}\\|I|=k}}Q_{2\ell,k}(\mathcal{K},I)=\sum_{k=\ell+1}^{2\ell}\left(\sum_{\substack{I\subseteq\mathcal{K}\\|I|=k}}Q_{2\ell+1,k+1}(\{\mathcal{K},a\},\{I,a\})+\sum_{\substack{I\subseteq\mathcal{K}\\|I|=k}}Q_{2\ell+1,k}(\{\mathcal{K},a\},I)\right)\\&<\sum_{k=\ell+1}^{2\ell}\sum_{\substack{I\subseteq\mathcal{K}\\|I|=k}}Q_{2\ell+1,k}(\mathcal{K},I)=q_{2\ell+1}(\mathcal{K},a),\end{aligned} \tag{29}$$

since $q_{2\ell+1}(\mathcal{K})$ includes some extra additional terms, say $Q_{2\ell+1,k}(\{\mathcal{K},a\},I)$, where I contains a. Regarding that n is odd in the series of $q_\ell(\mathcal{K})$, there will be an element with odd ℓ following an element of even ℓ and the lemma is proved for odd n. For the case when n is even, we consider the $q_n(\mathcal{N})$ and $q_{n-1}(\mathcal{L})$, where $\mathcal{L}=\{1,2,\ldots,2\ell-1\}$. Set $n=2\ell$; then,

$$q_{2\ell}(\mathcal{N})=\sum_{k=\ell+1}^{2\ell}\sum_{\substack{I\subseteq\mathcal{N}\\|I|=k}}Q_{2\ell,k}(\mathcal{N},I) \tag{30}$$

with $\mathcal{N}=\{1,2,\ldots,2\ell\}$ and put

$$q_{2\ell-1}(\mathcal{L})=\sum_{k=\ell}^{2\ell-1}\sum_{\substack{I\subseteq\mathcal{L}\\|I|=k}}Q_{2\ell-1,k}(\mathcal{L},I), \tag{31}$$

notice that the number of terms is equal in both sums. If $k=2\ell$, then

$$Q_{2\ell,2\ell}(\mathcal{N},\mathcal{N})=p_{2\ell}Q_{2\ell-1,2\ell-1}(\mathcal{L},\mathcal{L}), \tag{32}$$

otherwise,

$$Q_{2\ell,k}(\mathcal{N},I)=\begin{cases}p_{2\ell}Q_{2\ell-1,k-1}(\mathcal{L},I\backslash 2\ell) & \text{if } 2\ell\in I\\ \big(1-p_{2\ell}\big)Q_{2\ell-1,k}(\mathcal{L},I) & \text{if } 2\ell\notin I\end{cases}, \tag{33}$$

hence for $k<2\ell$,

$$\sum_{\substack{I \subseteq \mathcal{N} \\ |I| = k}} Q_{2\ell,k}(\mathcal{N}, I) = p_{2\ell} \sum_{\substack{I \subseteq \mathcal{L} \\ |I| = k-1}} Q_{2\ell-1,k-1}(\mathcal{L}, I) + \left(1 - p_{2\ell}\right) \sum_{\substack{I \subseteq \mathcal{L} \\ |I| = k}} Q_{2\ell-1,k}(\mathcal{L}, I). \tag{34}$$

We start summing up $q(\mathcal{N}, 2\ell)$ from 2ℓ; then, using (32) and (34), we obtain for the first two terms

$$\begin{aligned} \sum_{k=2\ell-1}^{2\ell} \sum_{\substack{I \subseteq \mathcal{N} \\ |I| = 2\ell-1}} Q_{2\ell,k}(\mathcal{N}, I) &= p_{2\ell} Q_{2\ell-1,2\ell-1}(\mathcal{L}, \mathcal{L}) \\ &+ p_{2\ell} \sum_{\substack{I \subseteq \mathcal{L} \\ |I| = 2\ell-2}} Q_{2\ell-1,2\ell-2}(\mathcal{L}, I) + \left(1 - p_{2\ell}\right) Q_{2\ell-1,2\ell-1}(\mathcal{L}, \mathcal{L}) \\ &= Q_{2\ell-1,2\ell-1}(\mathcal{L}, \mathcal{L}) + p_{2\ell} \sum_{\substack{I \subseteq \mathcal{L} \\ |I| = 2\ell-2}} Q_{2\ell-1,2\ell-2}(\mathcal{L}, I). \end{aligned} \tag{35}$$

If we continue summing up one by one, then induction leads to

$$\begin{aligned} q_{2\ell}(\mathcal{N}) = \sum_{k=\ell+1}^{2\ell-1} \sum_{\substack{I \subseteq \mathcal{L} \\ |I| = k}} Q_{2\ell-1,k-1}(\mathcal{L}, I) \\ + p_{2\ell} \sum_{\substack{I \subseteq \mathcal{L} \\ |I| = \ell}} Q_{2\ell-1,\ell}(\mathcal{L}, I) < q_{2\ell-1}(\mathcal{L}), \end{aligned} \tag{36}$$

since $p_{2\ell} < 1$. □

Appendix 2: Proof of Proposition 1

Proof We prove the statement with an example describing the worst-case scenario for the greedy selection strategy. Let $\mathcal{D} = \{D_1 = (p_1, t_1), D_2 = (p_2, t_2), \ldots, D_n = (p_n, t_n)\}_n$ be the pool, where the index set is denoted by $\mathcal{I}_n = \{1, 2, \ldots, n\}$. Let us suppose that $\sum_{i=1} t_i \leq T$, that is, the time constraint should not be of concern. Let $p_1 = 1/2 + \varepsilon$, where $0 < \varepsilon \leq 1/2$, and $p_2 = p_3 = \cdots = p_n = 1/2 + \alpha$ with $0 < \alpha < \varepsilon$, where the proper selection of α will be given below.

The greedy strategy will move D_1 to S as the most accurate item in its first step. Next, we try to extend S by adding more members. Since we require odd members, we try to add 2 items in every selection step. Since all the remaining $n - 1$ features have the same behavior, we can check whether S should be extended via comparing the performance of $S_1 = \{D_1\}$ and $S_3 = \{D_1, D_2, D_3\}$. For the performance of the ensemble S_1, we trivially

have $q_1(\mathcal{I}_1) = p_1 = 1/2 + \varepsilon$, where $\mathcal{I}_1 = \{1\}$, while for S_3 we can apply (1) for the 3-member ensemble, with $\mathcal{I}_3 = \{1, 2, 3\}$ to calculate $q_3(\mathcal{I}_3)$:

$$q_3(\mathcal{I}_3) = p_1p_2(1-p_3) + p_2p_3(1-p_1) + p_1p_3(1-p_2) + p_1p_2p_3 = \frac{1}{2} + \frac{\varepsilon}{2} + \alpha - 2\alpha^2\varepsilon \tag{37}$$

after the appropriate substitutions and simplifications. Now, if we adjust α to have $q_1(\mathcal{I}_1) = q_3(\mathcal{I}_3)$, then via solving the equation

$$\frac{1}{2} + \varepsilon = \frac{1}{2} + \frac{\varepsilon}{2} + \alpha - 2\alpha^2\varepsilon \tag{38}$$

we obtain

$$\alpha = \frac{1 - \sqrt{1 - 4\varepsilon^2}}{4\varepsilon}. \tag{39}$$

That is, with a selection of α given in (39), the ensemble $S_1 = \{D_1\}$ is not going to be extended since it does not lead to improvement. Thus, the strategy stops after the first step with an ensemble accuracy $1/2 + \varepsilon$.

On the other hand, with a sufficiently large n, a very accurate ensemble could be achieved. More precisely, it can be easily seen that $q_n(\mathcal{I}_n)$ is strictly monotonically increasing with

$$\lim_{n\to\infty} q_n(\mathcal{I}_n) = 1. \tag{40}$$

Now, by letting $\varepsilon \to 0$, we can see that for the ensemble accuracy found with this strategy

$$\lim_{\varepsilon\to 0} q_1(S_1) = 1/2, \tag{41}$$

while an ensemble of $\lim_{n\to\infty} q_n(\mathcal{I}_n) = 1$ could also be found. Hence, the proposition follows. □

Appendix 3: Proof of Proposition 2

Proof We prove the statement with a similar example to that given in the proof of Proposition 1 in Appendix 1 to describe the worst-case scenario. Let $\mathcal{D} = \{D_1 = (p_1, t_1), D_2 = (p_2, t_2), \ldots, D_n = (p_n, t_n)\}$ be the pool and T be the time constraint. Put $p_1 = 1/2 + \varepsilon$, where $0 < \varepsilon \leq 1/2$, $t_1 = T$, and $p_2 = p_3 = \cdots = p_m = 1/2 + \alpha$, $t_2 = t_3 = \cdots = t_n = T/(n-1)$ with $0 < \alpha < \varepsilon$. If α is properly selected, then $q_1(\mathcal{I}_1) = p_1 < q_{n-1}(\mathcal{I}_n \backslash \mathcal{I}_1)$. However, because of the time constraint, we must remove elements during the selection procedure, since initially $\sum_{i=1}^{n} t_i = 2T > T$. For this requirement, the greedy approach in the first step will remove any two elements from $D_2, \ldots, D_n$ by decreasing the time with $2T/(n-1)$. This selection will go on until only D_1 remains in the ensemble. With a proper selection of α, we have $\lim_{n\to\infty} q_{n-1}(\mathcal{I}_n \backslash \mathcal{I}_1) = 1$ and by letting $\varepsilon \to 0$, the proposition follows. □

Appendix 4: Proof of Proposition 3

Proof Similar to the proof of Proposition 2, we provide an example for the worst-case scenario. Let $D_1 = (1, T)$, and $D_2 = D_3 = D_4 = (1/2 + \varepsilon, T/3)$ with $0 < \varepsilon < 1/2$. Now, since

$$u_1 = \frac{1}{T} < \frac{3/2 + 3\varepsilon}{T} = u_2 = u_3 = u_4, \tag{42}$$

the backward strategy will remove the less useful component D_1 first to maintain the time constraint and will keep the remaining ensemble $\{D_2, D_3, D_4\}$ as the most accurate one, which also fits the time constraint with $\sum_{i=2}^{4} t_i = T$. By letting $\varepsilon \to 0$, we have $\lim_{\varepsilon \to 0} q_3(\mathcal{I}_4 \backslash \mathcal{I}_1) = 1/2$. Moreover, notice that the most accurate ensemble would have been $\{D_1\}$ with $q_1(\mathcal{I}_1) = 1$ by meeting the time constraint as well. Thus, the statement follows. □

Appendix 5: Lemma 2

Lemma 2 *Let $p \in [0, 1]$ be a random variable with mean μ_p and variance σ_p^2. Consider the accuracy (1), where $p_i, i = 1, 2, \ldots, n$ are i.i.d. random variables distributed as p. Then,*

1.
$$\lim_{\ell \to \infty} \mu_{q_\ell} = \begin{cases} 0, & \text{if } \mu_p \notin [1/2, 1), \\ 1/2, & \text{if } \mu_p = 1/2, \\ 1, & \text{if } \mu_p \in (1/2, 1). \end{cases} \tag{43}$$

Moreover, for odd ℓ: if $\mu_p \in (1/2, 1)$, then μ_{q_ℓ} is increasing, and if $\mu_p \in (0, 1/2)$, then μ_{q_ℓ} is decreasing.

2. *The variance of q_ℓ is expressed by*

$$\sigma_{q_\ell}^2 = \sum_{k=k_\ell}^{\ell} \sum_{m=1}^{k} \sum_{h=k_\ell - m}^{\ell - k} \delta(\ell, m, k) \binom{\ell}{k} \binom{k}{m} \binom{\ell - k}{h} s_T^m s_{TF}^{k-m+h} s_F^{\ell-k-h} - \left(\mu_{q_\ell}\right)^2, \tag{44}$$

where $\delta(\ell, m, k) = \delta_{k_\ell - m \le \ell - k}$, $s_T = \sigma_p^2 + \mu_p^2$, $s_F = \sigma_p^2 + (1 - \mu_p)^2$, $s_{TF} = \mu_p(1 - \mu_p) - \sigma_p^2$, and $k_\ell = \left\lfloor \frac{\ell}{2} \right\rfloor + 1$.

3. *If $\mu_p \neq 1/2$, $\mu_p(1 - \mu_p) - \sigma_p^2 > 0$, and $s_T \neq 1/2$, then*

$$\lim_{\ell \to \infty} \sigma_{q_\ell}^2 = 0. \tag{45}$$

If $s_T = 1/2$, then the limit (45) is 1.

Proof The first part of the lemma corresponds to Theorem 1 in Lam and Suen (1997). For the rest, let us denote the product of probabilities by

$$\Pi(I) = \prod_{i \in I} p_i \prod_{j \in \mathcal{N} \setminus I} (1 - p_j), \tag{46}$$

for simplifying the treatment below. The formula (44) follows from expressing the variance in terms of covariance

$$*Var(q_\ell) = \sum_{k,j=k_\ell}^{\ell} \sum_{\substack{I,J \subseteq \mathcal{N} \\ |I|=k,|J|=j}} *Cov(\Pi(I), \Pi(J)). \tag{47}$$

Now, we rewrite this expression into a more appropriate form. First, the notation is introduced, where $I_T^k$and I_F^k for a partition of indices $\mathcal{N} = \{1, \ldots, \ell\}$, such that $\mathcal{N} = I_T^k \cup I_F^k$ where I_T^k denotes indices of those members voting true with accuracy p. Similarly, I_F^k contains indices of false votes. Observe $I_F^k = \mathcal{N} \setminus I_T^k$. We have $\left|I_T^k\right| = k$ and $\left|I_F^k\right| = \ell - k$. In the case of two partitions $I_T^k \cup I_F^k$ and $J_T^j \cup J_F^j$, let the number of the common elements of I_T^k and J_T^j be $\left|I_T^k \cap J_T^j\right| = n_{k,j}$; similarly, $\left|I_F^k \cap J_F^j\right| = m_{k,j}$. According to this setup

$$*Var(q_\ell) = \sum_{k,j=k_\ell}^{\ell} \sum_{I_T^k, J_T^j} *Cov\left(\Pi\left(I_T^k\right), \Pi\left(J_T^k\right)\right). \tag{48}$$

Observe $I_F^k = \mathcal{N} \setminus I_T^k$ when we apply the notation for the product. Now, we consider the covariance

$$\begin{aligned} *Cov\left(\Pi\left(I_T^k\right), \Pi\left(J_T^k\right)\right) &= *E\Pi\left(I_T^k\right)\Pi\left(J_T^k\right) - *E\Pi\left(I_T^k\right) *E\Pi\left(J_T^k\right) \\ &= *E\Pi\left(I_T^k\right)\Pi\left(J_T^k\right) - \mu^{k+j}(1-\mu)^{2\ell-k-j}. \end{aligned} \tag{49}$$

The first term contains three types of products:

$$*Ep^2 = s_T = \sigma_p^2 + \mu_p^2, \tag{50}$$

$$*E(1-p)^2 = s_F = \sigma_p^2 + \left(1 - \mu_p\right)^2, \tag{51}$$

$$*Ep(1-p) = s_{TF} = \mu_p\left(1 - \mu_p\right) - \sigma_p^2. \tag{52}$$

The pool constitutes independent variables; therefore,

$$\begin{aligned} *Var(q_\ell) = \sum_{k,j=k_\ell}^{\ell} \sum_{I^k, J^j} &\left(\sigma_p^2 + \mu_p^2\right)^{n_{k,j}} \left(\sigma_p^2 + \left(1 - \mu_p\right)^2\right)^{m_{k,j}} \\ &\left(\mu_p\left(1 - \mu_p\right) - \sigma_p^2\right)^{\ell - n_{k,j} - m_{k,j}} - \left(Eq_\ell\right)^2. \end{aligned} \tag{53}$$

since the sum of the second term gives the $\left(Eq_\ell\right)^2$, indeed

$$\sum_{k,j=k_\ell}^{\ell} \binom{\ell}{k} \binom{\ell}{j} \mu_p^{k+j} \left(1 - \mu_p\right)^{2\ell-k-j} = \left(\sum_{k=k_\ell}^{\ell} \binom{\ell}{k} \mu_p^k \left(1 - \mu_p\right)^{\ell-k}\right)^2 = \left(Eq_\ell\right)^2. \tag{54}$$

We simplify (53), collecting similar terms and obtain (44). Before we prove the limit (45), let us observe

$$s_T + s_{TF} = \mu_p, \tag{55}$$

$$s_F + s_{TF} = 1 - \mu_p, \tag{56}$$

$$s_T + s_F + 2s_{TF} = 1. \tag{57}$$

i.e., the set $\{s_T, s_{TF}, s_F, s_{TF}\}$ constitutes a probability distribution for $s_{TF} > 0$; in other words, $\mu_p^2 + \sigma_p^2 < \mu_p$. If it is so, we rewrite (44) in the form of a multinomial distribution. The coefficients in (44) are actually multinomial coefficients. The rest of the proof is based on the approximation of the binomial distribution by the normal distribution. It is not complicated but slightly lengthy; we make it available to the interested readers on request.

□

Appendix 6: Lemma 3

Lemma 3 *Let τ be an exponential distribution with density $\lambda \exp(-\lambda t)$ under the condition that the parameter λ is distributed as Beta(β_p, α_p), where $2 < \beta_p < \alpha_p$.*

1. *Then, the expected time for the sum of n variables is*

$$\sum_{k=0}^{n} E\tau_k = n\left(1 + \frac{\alpha_p}{\beta_p - 1}\right) \tag{58}$$

with variance

$$Var\left(\sum_{k=0}^{n} \tau_k\right) = n\left(1 + \frac{\alpha_p}{\beta_p - 2}\right). \tag{59}$$

This implies that the estimated number of ensemble members up to time T is $\widehat{\ell_T} = \left\lceil T \frac{\beta_p - 1}{\alpha_p + \beta_p - 1} \right\rceil$.

2. *If the interarrival times τ_j correspond to a given T and λ generated from Beta(β_p, α_p), then*

$$E(\ell_T) = \frac{\beta_p}{\alpha_p + \beta_p} T. \tag{60}$$

3. *If each component of pair (λ_j, τ_j) are independent copies of λ and τ_j corresponds to λ_j, then*

$$E(\ell_T) = T\frac{\beta_p - 1}{\alpha_p + \beta_p - 1}. \tag{61}$$

In both cases 2) and 3), ℓ_T is distributed as Poisson with parameter $T/E\tau_1$, which implies that $Var(\ell_T) = E(\ell_T)$ and the estimation of ℓ_T is

$$\widehat{\ell_T} = T/\overline{\tau}. \tag{62}$$

Proof We show only the first statement; the rest of the lemma is well known. If $\lambda \in (0,1)$ is distributed as $Beta(\alpha_p, \beta_p)$, then $1-\lambda$ is distributed as $beta(\beta_p, \alpha_p)$. The expected value of time is calculated in two steps; first, we take the conditional expectation, namely,

$$E\tau = EE(\tau|\lambda) = \int_0^1 \int_0^\infty t\lambda \exp(-\lambda t) dt b(\lambda;\beta_p,\alpha_p) d\lambda = \frac{\Gamma(\beta_p - 1)}{\Gamma(\alpha_p + \beta_p - 1)} \frac{\Gamma(\alpha_p + \beta_p)}{\Gamma(\beta_p)} = 1 + \frac{\alpha_p}{\beta_p - 1}, \tag{63}$$

where we assumed that $1 < \beta_p < \alpha_p$. Suppose $2 < \beta_p < \alpha_p$ to calculate the variance in a similar manner

$$Var(\tau) = EE\Big((\tau - E(\tau|\lambda))^2 \Big| \lambda \Big) = \int_0^1 \frac{1}{\lambda^2} b(\lambda;\beta_p,\alpha_p) d\lambda = 1 + \frac{\alpha_p}{\beta_p - 2}. \tag{64}$$

□

Appendix 7: Comparing search strategies on the whole UCI datasets

See Tables 7, 8, 9 and 10.

Table 7 Comparing the proposed search strategy (SHErLoCk) with other selection methods regarding ensemble accuracy and computational time on binary classification problems of UCI datasets using an ensemble pool of $n = 30$ classifiers and training time as cost

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
		Method									
Ensemble accuracy	MAXSTEP	SHErLoCk	93.87%	97.26%	76.19%	96.30%	99.29%	98.08%	99.90%	93.89%	86.92%
		SA+	93.34%	96.68%	76.23%	96.61%	99.25%	97.01%	99.53%	93.02%	86.10%
		SA	93.16%	96.68%	76.26%	96.09%	98.04%	97.39%	98.58%	93.05%	85.57%
		Genetic+	93.86%	97.12%	75.39%	96.07%	98.99%	97.74%	99.43%	93.14%	87.05%
		Genetic	93.87%	96.68%	76.12%	96.19%	99.27%	97.23%	99.53%	93.39%	86.91%
		Pruning	92.83%	96.06%	75.51%	95.33%	98.06%	97.58%	98.24%	92.37%	85.37%
	STOP	SHErLoCk	92.29%	95.47%	74.63%	95.43%	99.02%	97.53%	98.58%	93.03%	85.67%
		SA+	92.67%	95.95%	74.09%	95.02%	98.06%	96.75%	98.68%	92.48%	85.03%
		SA	92.13%	95.98%	74.23%	95.31%	98.01%	96.98%	98.44%	93.01%	85.06%
		Genetic+	92.04%	95.31%	74.07%	95.16%	98.84%	96.22%	98.35%	92.96%	85.38%
		Genetic	92.04%	95.11%	74.58%	95.11%	98.51%	96.20%	98.79%	93.05%	85.67%
		Pruning	92.07%	95.12%	73.85%	95.01%	98.03%	96.57%	98.39%	90.13%	84.84%
	DET	Forward	90.41%	93.82%	70.42%	94.96%	95.14%	96.22%	98.08%	91.38%	83.49%
		Backward	90.19%	93.82%	68.76%	94.96%	95.14%	95.98%	98.08%	91.38%	82.98%

Table 7 (continued)

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
Computational time (secs)	MAXSTEP	SHErLoCk	46.47	37.90	49.46	30.20	48.89	32.19	37.06	51.45	34.73
		SA+	121.9	100.98	128.90	145.57	108.21	101.73	106.13	132.43	96.51
		SA	79.06	80.12	93.47	71.39	67.26	77.25	60.19	95.98	72.28
		Genetic+	63.42	68.03	59.57	61.07	74.02	69.09	89.67	67.03	69.08
		Genetic	46.08	48.92	53.22	58.16	51.23	48.56	57.99	64.42	49.18
		Pruning	125.84	107.08	115.42	129.45	93.06	90.98	103.54	137.08	151.08
	STOP	SHErLoCk	0.98	0.39	0.93	0.38	0.92	0.74	0.39	0.43	0.28
		SA+	8.41	4.67	6.73	8.47	7.33	5.09	8.79	5.88	6.11
		SA	8.78	6.92	4.56	7.93	8.59	9.48	6.95	5.89	7.99
		Genetic+	5.53	5.96	6.45	9.97	5.15	7.56	6.93	7.91	5.69
		Genetic	6.78	6.23	6.49	10.98	8.52	10.43	6.02	5.62	8.43
		Pruning	13.61	12.21	14.91	18.85	11.07	12.98	14.03	13.57	12.09
	DET	Forward	0.28	0.19	0.41	0.29	0.28	0.38	0.46	0.49	0.73
		Backward	0.21	0.37	0.43	0.42	0.31	0.84	0.35	0.25	0.23

Table 8 Comparing the proposed search strategy (SHErLoCk) with other selection methods regarding ensemble accuracy and computational time on binary classification problems of UCI datasets using an ensemble pool of $n = 100$ classifiers and training time as cost

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
		Method									
Ensemble accuracy	MAXSTEP	SHErLoCk	94.66%	97.88%	77.16%	96.98%	99.49%	98.97%	99.99%	94.16%	87.88%
		SA+	94.57%	97.79%	76.88%	96.98%	99.38%	98.95%	99.99%	94.03%	86.92%
		SA	94.63%	97.75%	76.64%	96.11%	99.43%	97.98%	99.93%	93.94%	86.23%
		Genetic+	94.95%	97.39%	76.41%	96.09%	99.43%	98.56%	99.89%	93.72%	88.15%
		Genetic	94.79%	97.65%	76.84%	96.22%	99.08%	98.57%	99.91%	93.74%	87.19%
		Pruning	93.88%	97.19%	76.63%	95.23%	98.71%	98.66%	98.94%	93.03%	85.79%
	STOP	SHErLoCk	94.04%	96.88%	76.73%	95.64%	99.28%	97.61%	99.29%	93.19%	86.99%
		SA+	94.19%	96.96%	76.02%	95.32%	99.18%	96.91%	98.97%	93.19%	86.01%
		SA	94.17%	96.99%	76.02%	96.09%	99.21%	97.31%	99.03%	93.44%	86.02%
		Genetic+	93.91%	96.91%	76.29%	95.96%	98.97%	98.11%	99.07%	93.41%	86.49%
		Genetic	94.01%	96.46%	76.52%	95.88%	98.88%	97.77%	99.90%	93.23%	86.09%
		Pruning	93.53%	96.38%	75.56%	95.19%	98.24%	97.12%	98.29%	93.01%	85.75%
	DET	Forward	92.11%	94.92%	73.29%	95.14%	97.92%	96.69%	99.31%	92.98%	85.08%
		Backward	93.92%	94.92%	75.11%	95.23%	97.92%	96.78%	99.31%	92.95%	84.83%

Table 8 (continued)

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
Computational time (secs)	MAXSTEP	SHErLoCk	194.12	206.89	191.91	203.88	214.28	201.03	186.67	159.32	178.01
		SA+	349.62	290.89	390.82	278.56	253.59	375.71	313.56	301.25	311.87
		SA	251.02	291.13	269.22	278.39	269.59	228.44	286.23	259.92	258.62
		Genetic+	305.26	298.34	289.37	301.26	324.12	256.67	289.44	338.98	381.55
		Genetic	226.05	301.36	197.57	239.79	223.19	267.24	210.63	231.92	290.67
		Pruning	354.23	301.21	409.22	354.59	356.18	321.82	402.34	441.23	455.63
	STOP	SHErLoCk	2.32	3.41	2.11	2.08	1.56	2.33	2.49	1.73	2.45
		SA+	13.31	14.67	12.23	12.45	14.95	13.97	12.88	10.12	14.81
		SA	13.39	12.58	16.14	15.55	12.04	16.94	11.76	12.35	13.55
		Genetic+	12.56	12.61	15.95	13.37	13.49	16.68	14.88	13.14	12.59
		Genetic	13.53	13.58	12.88	15.47	11.51	13.67	12.02	11.27	12.28
		Pruning	19.41	11.78	17.32	36.36	21.69	31.55	22.92	15.34	19.53
	DET	Forward	0.59	0.72	0.81	0.92	0.69	0.89	0.93	0.96	0.94
		Backward	0.71	0.78	0.99	0.85	0.76	0.88	0.95	0.81	0.99

Table 9 Bayesian signed-rank test on the UCI test sets ($n = 30$) for training and testing times as cost

Algorithm	$P(SHErLoCk > Algorithm)$			
	MAXSTEP		*STOP*	
	Training time	Testing time	Training time	Testing time
SA+	0.9800	0.3408	0.5113	0.6287
SA	0.9983	0.9929	0.8748	0.8727
GA+	0.9982	0.6908	0.9485	0.4987
GA	0.9553	0.8814	0.9112	0.9418
Pruning	0.9999	0.9346	0.9779	0.4005

Table 10 Bayesian signed-rank test on the UCI test sets ($n = 100$) for training and testing times as cost

Algorithm	$P(SHErLoCk > Algorithm)$			
	MAXSTEP		*STOP*	
	Training time	Testing time	Training time	Testing time
SA+	0.9898	0.5854	0.7378	0.7592
SA	0.9983	0.9647	0.6888	0.3361
GA+	0.7826	0.8720	0.7246	0.5074
GA	0.7355	0.8271	0.4535	0.6399
Pruning	1.0000	0.9657	0.9998	0.8097

Appendix 8: Comparing search strategies on the UCI test dataset using testing time

See Tables 11 and 12.

Table 11 Comparing the proposed search strategy (SHErLoCk) with other selection methods regarding ensemble accuracy and computational time on binary classification problems of the UCI test subsets using an ensemble pool of $n = 30$ classifiers and testing time as cost

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
		Method									
Ensemble accuracy	MAXSTEP	SHErLoCk	93.72%	96.88%	75.28%	96.33%	97.77%	98.49%	99.54%	94.13%	88.20%
		SA+	93.99%	96.80%	75.46%	96.55%	98.38%	98.31%	99.21%	94.04%	88.11%
		SA	93.51%	96.19%	75.03%	96.07%	98.33%	97.80%	98.56%	93.99%	87.02%
		Genetic+	93.57%	96.35%	75.59%	96.19%	97.91%	98.59%	99.44%	92.51%	88.39%
		Genetic	93.88%	96.89%	75.07%	96.05%	98.16%	98.57%	99.25%	92.33%	87.04%
		Pruning	93.51%	95.48%	75.17%	96.91%	98.55%	98.42%	99.03%	93.11%	87.32%
	STOP	SHErLoCk	92.33%	94.54%	75.21%	94.54%	99.01%	96.53%	97.92%	91.84%	85.08%
		SA+	92.30%	95.10%	74.96%	94.01%	98.99%	96.32%	98.18%	91.48%	85.51%
		SA	91.78%	94.23%	74.99%	93.80%	98.42%	96.82%	98.11%	92.02%	85.42%
		Genetic+	91.91%	94.17%	75.29%	94.39%	99.18%	96.04%	97.91%	92.57%	85.99%
		Genetic	92.09%	94.02%	75.89%	93.91%	98.09%	96.33%	98.09%	91.91%	84.36%
		Pruning	92.88%	94.16%	75.66%	94.33%	98.77%	96.22%	98.01%	92.23%	84.99%
	DET	Forward	88.51%	95.01%	73.33%	94.77%	98.11%	96.00%	98.65%	91.29%	82.59%
		Backward	89.39%	94.92%	73.45%	94.68%	98.11%	96.00%	98.49%	90.98%	82.59%

Table 11 (continued)

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
Computational time (secs)	MAXSTEP	SHErLoCk	36.47	32.45	38.46	33.89	34.21	30.59	34.89	39.03	36.79
		SA+	89.9	69.56	88.35	82.13	89.46	71.93	66.09	89.31	69.39
		SA	82.78	76.26	73.89	78.33	69.28	75.95	69.88	75.08	75.74
		Genetic+	67.92	55.98	50.52	60.98	61.97	60.00	58.31	61.93	57.99
		Genetic	49.97	44.48	50.95	58.11	53.88	53.62	49.46	54.02	49.93
		Pruning	120.12	121.09	101.02	96.28	96.86	93.67	113.10	96.98	94.638
	STOP	SHErLoCk	0.38	0.29	0.56	0.36	0.72	0.52	0.29	0.36	0.418
		SA+	4.55	4.11	3.83	4.19	4.03	3.89	3.56	3.08	3.97
		SA	3.81	2.47	3.69	3.13	2.79	3.01	4.51	3.03	2.38
		Genetic+	2.38	2.34	3.13	2.99	1.55	2.16	3.16	2.11	3.10
		Genetic	3.55	2.46	3.77	4.08	2.81	4.93	3.91	2.98	3.27
		Pruning	13.39	11.33	14.39	14.23	10.33	14.15	13.39	13.29	10.36
	DET	Forward	0.23	0.24	0.32	0.24	0.32	0.34	0.36	0.31	0.33
		Backward	0.37	0.33	0.34	0.52	0.29	0.42	0.34	0.29	0.21

Table 12 Comparing the proposed search strategy (SHErLoCk) with other selection methods regarding ensemble accuracy and computational time on binary classification problems of UCI test subsets using an ensemble pool of $n = 100$ classifiers and testing time as cost

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
		Method									
Ensemble accuracy	MAXSTEP	SHErLoCk	95.98%	97.68%	76.58%	94.91%	99.17%	99.05%	99.99%	93.72%	88.49%
		SA+	95.45%	97.85%	76.98%	95.88%	99.28%	98.63%	99.99%	93.54%	87.81%
		SA	95.37%	97.44%	76.08%	95.77%	98.88%	98.51%	99.90%	93.06%	87.36%
		Genetic+	95.99%	96.87%	76.46%	95.68%	99.25%	98.63%	99.99%	93.88%	88.13 %
		Genetic	95.83%	96.75%	76.69 %	96.44%	99.31%	99.02%	99.93%	93.72%	88.51%
		Pruning	95.81%	97.00 %	76.09 %	95.53 %	99.33 %	98.41 %	99.79%	93.61%	87.32 %
	STOP	SHErLoCk	93.88 %	95.49 %	75.93%	94.66%	98.49%	98.47%	99.01%	93.74%	86.55%
		SA+	93.31 %	95.66 %	76.19%	95.13%	98.22%	97.86%	99.03%	93.55%	86.41%
		SA	93.72%	95.91%	76.03%	95.29%	98.44%	97.90 %	99.03 %	93.43%	86.87%
		Genetic+	93.84%	95.93%	75.99%	95.18%	98.02%	97.98 %	99.41 %	93.92%	86.01%
		Genetic	93.22%	95.69%	76.25 %	95.48%	98.00%	98.11%	99.42%	92.74%	86.88%
		Pruning	93.31%	95.70%	75.74%	94.79%	99.12%	97.89%	99.03%	92.76%	86.36%
	DET	Forward	93.12%	95.17%	75.99%	93.67%	98.09%	96.31%	99.83%	92.49%	87.07%
		Backward	93.12%	95.98%	75.99%	93.33%	98.81%	96.31%	99.70%	93.32%	87.07%

Table 12 (continued)

Dataset (size)			MAGIC (19,020)	Spambase (4601)	HIGGS (20,000)	EEG (14,980)	Musk (6598)	Breast (699)	Mushroom (8124)	Gisette (13,500)	Adult (48,842)
Computational time (secs)	MAXSTEP	SHErLoCk	291.01	211.22	291.32	279.22	202.25	189.33	187.34	242.67	164.78
		SA+	299.01	340.23	310.82	298.56	293.23	319.34	283.00	241.22	305.11
		SA	294.23	271.45	289.49	291.02	239.87	282.42	299.93	306.39	274.73
		Genetic+	275.21	243.89	232.51	240.89	231.92	281.55	312.19	239.49	271.97
		Genetic	261.33	296.34	248.33	263.71	228.37	284.83	273.57	299.72	254.75
		Pruning	334.13	323.89	364.79	328.53	366.92	342.22	317.04	249.23	311.18
	STOP	SHErLoCk	3.21	3.34	3.09	3.13	2.64	2.92	3.09	2.39	2.75
		SA+	17.07	15.32	16.49	14.53	14.62	15.01	14.90	16.36	17.42
		SA	16.76	14.53	16.66	13.94	13.00	13.04	14.41	14.25	12.49
		Genetic+	16.21	15.34	15.14	15.70	14.34	15.42	14.36	14.95	16.01
		Genetic	10.21	14.09	13.28	13.12	10.87	13.38	11.79	13.47	12.52
		Pruning	18.01	16.21	16.89	22.31	20.10	19.63	22.91	19.32	16.41
	DET	Forward	1.01	1.45	1.33	1.28	0.99	1.32	1.66	2.09	2.33
		Backward	1.42	1.41	1.05	1.58	1.06	1.11	1.32	0.94	1.61

Acknowledgements This work was supported by the project EFOP-3.6.2-16-2017-00015 supported by the European Union, co-financed by the European Social Fund.

The research was also supported by the ÚNKP-20-4-I New National Excellence Program of the Ministry for Innovation and Technology from the Source of the National Research, Development and Innovation Fund.

Funding Open access funding provided by University of Debrecen.

References

Antal, B., & Hajdu, A. (2012). An ensemble-based system for microaneurysm detection and diabetic retinopathy grading. *IEEE Transactions on Biomedical Engineering, 59*(6), 1720–1726. https://doi.org/10.1109/TBME.2012.2193126.

Antal, B., & Hajdu, A. (2014). An ensemble-based system for automatic screening of diabetic retinopathy. *Knowledge-Based Systems, 60,* 20–27. https://doi.org/10.1016/j.knosys.2013.12.023.

Benavoli, A., et al. (2017). Time for a change: A tutorial for comparing multiple classifiers through Bayesian analysis. *The Journal of Machine Learning Research, 18*(1), 2653–2688.

Bucilu, C., Caruana, R., & Niculescu-Mizil, A. (2006). Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'06* (pp. 535–541). Association for Computing Machinery, New York, NY, USA https://doi.org/10.1145/1150402.1150464

Cavalcanti, G. D., Oliveira, L. S., Moura, T. J., & Carvalho, G. V. (2016). Combining diversity measures for ensemble pruning. *Pattern Recognition Letters, 74,* 38–45. https://doi.org/10.1016/j.patrec.2016.01.029.

Cho, S. B., & Kim, J. H. (1995). Combining multiple neural networks by fuzzy integral for robust classification. *IEEE Transactions on Systems, Man, and Cybernetics, 25*(2), 380–384. https://doi.org/10.1109/21.364825.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation, 6*(2), 182–197.

Dheeru, D., & Karra Taniskidou, E. (2017) UCI machine learning repository.

Du, K., & Swamy, M. (2016). Search and optimization by metaheuristics: Techniques and algorithms inspired by nature. Springer

Dzahini, K. (2020). Expected complexity analysis of stochastic direct-search. Les Cahiers du GERAD. GERAD HEC Montréal. https://books.google.hu/books?id=PKuvzQEACAAJ

Freund, Y., & Schapire, R. E. (1999). Large margin classification using the perceptron algorithm. *Machine Learning, 37,* 277–296.

Goldberg, D. (1989). *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley.

Hajdu, A., Hajdu, L., Jónás, A., Kovács, L., & Tomán, H. (2013). Generalizing the majority voting scheme to spatially constrained voting. *IEEE Transactions on Image Processing, 22*(11), 4182–4194. https://doi.org/10.1109/TIP.2013.2271116.

Hajdu, A., Hajdu, L., Kovács, L., & Tomán, H. (2013). Diversity measures for majority voting in the spatial domain. In J.S. Pan, M.M. Polycarpou, M. Woźniak, A.C.P.L.F. de Carvalho, H. Quintián, E. Corchado (Eds.), *Hybrid artificial intelligent systems* (pp. 314–323). Springer Berlin Heidelberg.

Hajdu, A., Tomán, H., Kovács, L., & Hajdu, L. (2016). Composing ensembles by a stochastic approach under execution time constraint. In *2016 23rd international conference on pattern recognition (ICPR)* (pp. 222–227). https://doi.org/10.1109/ICPR.2016.7899637

Hansen, L. K., & Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 12*(10), 993–1001. https://doi.org/10.1109/34.58871.

Harangi, B., Baran, A., & Hajdu, A. (2018). Classification of skin lesions using an ensemble of deep neural networks. In *40th annual international conference of the IEEE engineering in medicine and biology society, EMBC* 2018, Honolulu, HI, USA, July 18–21, 2018 (pp. 2575–2578).

Hernández-Lobato, D., Martinez-Munoz, G., & Suarez, A. (2009). Statistical instance-based pruning in ensembles of independent classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 31*(2), 364–369.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network.

Ho, T. K., Hull, J. J., & Srihari, S. N. (1994). Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 16*(1), 66–75. https://doi.org/10.1109/34.273716.

Huang, Y. S., & Suen, C. Y. (1995). A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 17*(1), 90–94. https://doi.org/10.1109/34.368145.

Klastorin, T. (1990). On a discrete nonlinear and nonseparable knapsack problem. *Operations Research Letters, 9*(4), 233–237. https://doi.org/10.1016/0167-6377(90)90067-F.

Kong, E. B., & Dietterich, T. G. (1995). Error-correcting output coding corrects bias and variance. In A. Prieditis & S. Russell (Eds.), *Machine learning proceedings 1995* (pp. 313–321). Morgan Kaufmann. https://doi.org/10.1016/B978-1-55860-377-6.50046-3

Kuncheva, L. I. (2004). *Combining pattern classifiers: Methods and algorithms*. Wiley-Interscience.

Kurz, M., Hölzl, G., & Ferscha, A. (2013). Enabling dynamic sensor configuration and cooperation in opportunistic activity recognition systems. *International Journal of Distributed Sensor Networks, 9*(6), 652385. https://doi.org/10.1155/2013/652385.

Lam, L., & Suen, S. Y. (1997). Application of majority voting to pattern recognition: An analysis of its behavior and performance. *IEEE Transactions on Systems, Man, and Cybernetics, 27*(5), 553–568. https://doi.org/10.1109/3468.618255.

Larochelle, H., & Bengio, Y. (2008). Classification using discriminative restricted Boltzmann machines. In *Proceedings of the 25th international conference on machine learning (ICML)* (pp. 536–543).

Martello, S., & Toth, P. (1990). *Knapsack problems: Algorithms and computer implementations*. Wiley.

Martinez-Munoz, G., & Suarez, A. (2007). Using boosting to prune bagging ensembles. *Pattern Recognition Letters, 28,* 156–165. https://doi.org/10.1016/j.patrec.2006.06.018.

Mousavi, R., & Eftekhari, M. (2015). A new ensemble learning methodology based on hybridization of classifier ensemble selection approaches. *Applied Soft Computing, 37,* 652–666. https://doi.org/10.1016/j.asoc.2015.09.009.

Pisinger, D. (1995). Algorithms for knapsack problems.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1,* 81–106.

Sharkey, T. C., Romeijn, H. E., & Geunes, J. (2011). A class of nonlinear nonseparable continuous knapsack and multiple-choice knapsack problems. *Mathematical Programming, 126*(1), 69–96. https://doi.org/10.1007/s10107-009-0274-9.

Soto, V., Martínez-Muñoz, G., Hernández-Lobato, D., & Suárez, A. (2010). A double pruning algorithm for classification ensembles. In N. El Gayar, J. Kittler, & F. Roli (Eds.), *Multiple classifier systems* (pp. 104–113). Springer Berlin Heidelberg.

Suratgar, A. A., Tavakoli, M. B., & Hoseinabadi, A. (2005). *Modified Levenberg–Marquardt method for neural networks training* (pp. 24–48). World Academy of Science, Engineering and Technology.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9).

Tang, J., & Gupta, A. K. (1984). On the distribution of the product of independent beta random variables. *Statistics & Probability Letters, 2*(3), 165–168.

Tempo, R., & Ishii, H. (2007). Monte Carlo and Las Vegas randomized algorithms for systems and control*: An introduction. *European Journal of Control, 13*(2), 189–203. https://doi.org/10.3166/ejc.13.189-203.

Tiba, A., Hajdu, A., Terdik, G., & Toman, H. (2019). Optimizing majority voting based systems under a resource constraint for multiclass problems. eprint arXiv:1904.04360

Timotheou, S. (2010). The random neural network: A survey. *The Computer Journal, 53,* 251–267.

Williamson, D. P., & Shmoys, D. B. (2011). *The design of approximation algorithms* (1st ed.). Cambridge University Press.

Xu, L., Krzyzak, A., & Suen, C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics, 22*(3), 418–435. https://doi.org/10.1109/21.155943.

Zhou, Z. H. (2012). *Ensemble methods: Foundations and algorithms* (1st ed.). Chapman & Hall/CRC.