Journal of Machine Learning Research 10 (2009) 1341-1366 Submitted 5/08; Revised 3/09; Published 7/09

Feature Selection with Ensembles, Artificial Variables, and Redundancy Elimination

Eugene Tuv EUGENE.TUV@INTEL.COM
Intel, Logic Technology Development
5000 W Chandler Blvd, CH5295
Chandler, AZ, 85226, USA

Alexander Borisov ALEXANDER.BORISOV@INTEL.COM
Intel, Logic Technology Development
Lopatina st. 3-199
N.Novgorod
Russia, 603163

George Runger RUNGER@ASU.EDU
Industrial Engineering Department
Arizona State University
Tempe, AZ, 85287, USA

Kari Torkkola KARITO@AMAZON.COM
Amazon.com
701 5th Avenue, Suite 1800
Seattle, WA, 98104, USA

Editor: Isabelle Guyon and Amir Reza Saffari

Abstract

Predictive models benefit from a compact, non-redundant subset of features that improves interpretability and generalization. Modern data sets are wide, dirty, mixed with both numerical and categorical predictors, and may contain interactive effects that require complex models. This is a challenge for filters, wrappers, and embedded feature selection methods. We describe details of an algorithm using tree-based ensembles to generate a compact subset of non-redundant features. Parallel and serial ensembles of trees are combined into a mixed method that can uncover masking and detect features of secondary effect. Simulated and actual examples illustrate the effectiveness of the approach.

Keywords: trees, resampling, importance, masking, residuals

1. Introduction

Large data sets are becoming the norm and traditional methods designed for data sets with a modest number of features will struggle in the new environment. This problem area was described by Guyon and Elisseeff (2003) along with other publications in the same issue, and it has increased in importance since then. Additional comments and examples have been provided by Liu and Yu (2005) in a recent survey article.

1.1 Feature Selection

There are three major categories of feature selection methods. Filter methods score variables, typically individually, and eliminate some before a model is constructed. The filter needs to be generated carefully to relate well to the requirements of the modeling task. In particular, the filter may not consider the value of one variable in the presence of others. For example, the widely-used value difference metric (VDM) (Stanfill and Waltz, 1986) and its modified version (MVDM) (Cost and Salzberg, 1993) consider the conditional probability distribution of the response at a predictor value. Such a measure is not sensitive to the effects of some predictors in a model with others present even though interactions among predictors might be critical for an effective subset. A sequential, subset search is sometimes implemented to improve the performance when interactions are important, although a greedy search also has disadvantages in the presence of interactions. Several common filter methods such as ReliefF (Robnik-Sikonja and Kononenko, 2003), CFS (Hall, 2000), and FOCUS (Almuallin and Dietterich, 1994) were modified with sequential search and compared by Yu and Liu (2004).

Wrapper methods form a second group of feature selection methods. The prediction accuracy (or the change in accuracy) of a model directly measures the value of a feature set. Although effective, the exponential number of possible subsets places computational limits for the wide data sets that are the focus of this work.

Embedded methods form a third group for feature selection. These methods use all the variables to generate a model and then analyze the model to infer the importance of the variables. Consequently, they directly link variable importance to the learner used to model the relationship.

1.2 Subset Feature Selection

Fundamentally, the goal of feature selection is to model a target response (or output) variable y, with a subset of the (important) predictor variables (inputs). This is a general goal and several more specific objectives can be identified. Each can lead to different strategies and algorithms. In *filtering* the interest is to remove irrelevant variables. Another objective is *variable ranking* where the interest is in obtaining relative relevance for all input variables with respect to the target. Finally, we might be interested in a compact, yet effective model, where the goal is to identify the smallest subset of independent features with the most predictive power, although a few alternative groups might be reasonable. An important concept here is *the masking relationships* among the predictor variables. Masking occurs when one variable can effectively represent others in a model. Along with the related issue of masking, this paper focuses on the subset selection.

1.3 Contributions of this Paper

Existing tree ensembles such as random forest (Breiman, 2001) or gradient boosting trees (Friedman, 1999) were developed primarily for predictive modeling. In addition, they can provide an importance ranking of the features, but this information has been considered an ad hoc benefit. Random forest (RF) is a random subspace method, and is capable of efficiently ranking features for large data sets. We exploit this property of RF, augment the original data with *artificial contrast variables* constructed independently from the target, and use their ranking for removal of irrelevant variables from the original set. The tree construction method is also modified to produce a more reliable variable ranking in the presence of high cardinality variables. A variable masking measure

is then introduced that incorporates surrogate variable scores from ensembles of trees. This forms the basis for *redundancy elimination*. Residual effects are calculated to enable the method to detect variables of secondary importance. These elements are integrated into an efficient algorithm for subset selection called ACE (artificial contrasts with ensembles).

The structure of this paper is as follows. In Section 2 we describe previous work and outline directions taken in this paper. Section 3 describes variable importance measures defined through tree ensembles and explains how they could be used to remove irrelevant features using random, artificial features. Next, we introduce a masking measure and use it for redundancy elimination. Section 4 describes the details of the ACE algorithm to generate the selected subset, and compares ACE with its closest competitors in detail. Section 5 provides results from experiments. Section 6 provides conclusions.

2. Background

This section defines the problem of finding the best susbset of features, discusses previous approaches, and outlines our solution.

2.1 Markov Boundaries

Let F be a full set of features. A feature selection solution can be described in terms of a Markov blanket (Koller and Sahami, 1996). Given a target feature Y, let $M \subset F$ and $Y \notin M$. M is said to be a Markov blanket for Y if $Y \perp (F - M)|M$. That is, Y is conditionally independent of other features given M. A minimal Markov blanket is referred to as Markov boundary (MB) and such a subset might be considered a feature selection solution. However, an important issue is that a MB need not be unique. Redundant features can replace others in a feature subset. Usually feature redundancy is defined in terms of feature correlation (Hall, 2000). For example, two features are redundant to each other if their values are completely correlated. In reality, it is not so straightforward to determine feature redundancy if a feature is partially correlated to a set of features.

Our goal is to focus on the important case with redundant features and obtain at least one MB. In most real-life problems exactly determining the MB or measuring feature relevance is very difficult because of a limited sample size, high time complexity, and noise in the data. Furthermore, evaluation of the distribution of the input variables and the response always relies on some model (linear, support vector machine, frequency tables, trees, etc.). In practice, most algorithms just try to remove irrelevant features and then apply some heuristics that remove "possibly" redundant variables.

2.2 Existing Approaches in Feature Selection

The nature of real life data sets provides strong restrictions for model fitting and feature selection methods. First, data sets may be very large both in terms of the number of predictors and in the number of samples (tens of thousands $\times$ tens of millions). Second, the predictors and the response can be of mixed type (both numeric and categoric), and can contain missing values. Lastly and also very importantly, dependency of the response on predictors can be highly non-linear, noisy and multivariate.

This leaves most existing methods out of scope for such problems. For example, wrapper methods (forward selection or backward elimination) are simply computationally unfeasible when dealing with thousands of predictors. Filter methods are also useless for the minimal subset selection

problem, as they do not deal with the notion of redundancy and most of them are inherently univariate. However, there are filters that use a"local" feature importance measure (like RELIEF) that can be considered multivariate (Kira and Rendell, 1992), but still they do not deal with redundancy giving just a ranked list of features instead of a selected minimal subset.

Subset evaluation filter methods such as CFS (Hall, 2000) are neither suitable because they do not deal explicitly with redundancy, and have to iterate over many feature subsets incurring a high time complexity. For example, the time complexity of the CFS is at least quadratic in the number of features and linear in number of samples. Also CFS is highly sensitive to outliers as it uses correlations between features.

Many embedded methods that use a built-in feature relevance measurement, such as SVM-RFE (Guyon et al., 2002) and linear regression with backward feature elimination are heavily dependent on the model (linear or SVM), that can fail to fit the data well. These methods have at least quadratic complexity in the number of samples for fitting an SVM and at least cubic complexity in the number of features ($O(nm^2+m^3)$, where m is the number of features, and n is number of samples) for fitting a regression model. Data sets with tens of thousands of features or samples become very time consuming and impractical to handle. For example, SVM-RFE involves retraining the SVM after features with smallest relevance are removed, thus incurring at least cubic complexity in number of samples ($O(max(m,n)n^2)$).

An issue that discourages using regression methods and methods that rely on some kind of distance measure between observations (linear regression, SVM, Kernel-based methods, RELIEF) is the difficulty of dealing with outliers in the input (predictor) space. Also, selection of important model parameters (kernel width and type, feature relevance thresholds, etc) is non-obvious, and the results of feature selection depend heavily on them.

Most methods return just a ranked list of features instead of an optimal subset. These methods include RELIEF, Koller's Markov blanket based backward elimination (referred to here as MBBE) (Koller and Sahami, 1996), and SVM-RFE. Some methods such as FCBS use a relevance threshold that is not clear how to adjust (Yu and Liu, 2004). In reality, the user also obtains a number of feature subsets corresponding to different values of parameters without a hint of how to choose the best subset.

Many methods work with frequency tables. They can thus deal well with categorical inputs only. For numerical inputs, they require discretization. Such methods are not always able to deal with interacting variables and have great difficulties with multivariate dependencies on numerical inputs. Examples of such methods are FCBS and MBBE. These two algorithms need discretization because they use an entropy measure computed on frequency tables. If the number of categories is large, or if we use frequency tables with more than two inputs, the tables can be sparse and may not represent the data distribution well. Another issue for MBBE is computational complexity. Considering all feature pairs incurs a quadratic complexity on the number of features.

Hence we see that most methods at hand are either not applicable at all to the best subset selection problem, or have some major problems. The most useful methods in such a setting (that appeared to be applicable to the examples of large "real-life" data in the challenge data sets discussed in Sec. 5.3) are methods based on backward feature elimination using an approximate Markov blanket concept (Koller and Sahami, 1996; Yu and Liu, 2004). Our method approximates the optimal Markov blanket redundancy elimination procedure, but without most of the drawbacks of previous methods.

2.3 Towards Efficient and Approximately Optimal Feature Selection

We propose a method that uses an idea similar to those proposed by Koller and Sahami (1996) and Yu and Liu (2004) that tries to overcome their limitations. It does not have quadratic time complexity in the number of features, can deal with thousands of predictors, uses a model (ensembles of trees) that can be applied to mixed variable types, thus eliminating need for discretization of numeric inputs, does not require imputation of missing values, captures local information (like RELIEF), is invariant to a monotone transformation of inputs, thus not very sensitive to noise and outliers, and deals well with multivariate dependencies.

It is well known that trees and especially ensembles of trees can provide robust and accurate models in "real-life" data settings. They handle mixed and noisy data, and are scale insensitive. Ensembles of trees have high predictive power and are resistant to over-fitting (Breiman, 2001). Our approach relies heavily on ensembles of trees.

First, we find irrelevant features that are conditionally independent of the response given the rest of the features. It is accomplished by comparing the relevance of the original variables with the relevance of random, artificial features (appended to the original data) constructed from the same distribution, but independently from the response. These features are referred to as artificial contrasts. We measure feature relevance as variable importance in random forests with a modified robust splitting criteria. We assume that if an original variable had a relevance score not statistically higher than that of an artificial probe (independent from the target by construction) then it is also independent from the target, irrelevant, and should be removed. Note that we try to remove irrelevant features by directly assessing conditional independence without searching for a MB, the existence of which is a much stronger requirement. Although the idea of artificial contrasts was already used by other researchers in simple filter methods with success (Stoppiglia et al., 2003), its application to tree ensembles is novel and promising. Actually, our approach can be considered as non-parametric because all parameters in our algorithm can be assigned reasonable default values that work well for wide range of problems.

Then the redundant feature elimination step is performed. Redundancy between features is measured using surrogate scores. The variable with the largest impurity reduction score at a node is the primary splitter. If surrogate variables (ones that partition the node in same way as the primary variable) are present, these surrogate variables are considered as "masked". Masking scores between all pairs of important variables are computed and evaluated using a statistical test, and variables masked by more important variables ("approximately redundant") are removed iteratively.

Finally, after a set of non-redundant relevant features has been found, our method removes the influence of the found subset with an ensemble and proceeds. Because redundancy elimination is approximate in nature this iterative approach is another advantage of our method. It allows one to recover variables with small importance and to reduce the chance to lose important variables during redundancy elimination.

3. Tree Ensembles for Feature Selection

For our embedded method, we focus on ensembles of decision trees for the following reasons. Trees can be applied in ubiquitous scenarios so that they provide a good entry point for feature selection for interdisciplinary, wide data sets. They apply to either a numerical or a categorical response. They are nonlinear, simple and fast learners that handle also both numerical and categorical predictors well. They are scale invariant and robust to missing values. A simple decision tree also provides an

embedded measure of variable importance that can be obtained from the number and the quality of splits that are generated from a predictor variable. However, a single tree is produced by a greedy algorithm that generates an unstable model. A small change to the data can result in a very different model. Consequently, ensemble methods have been used to counteract the instability of a single tree.

Supervised ensemble methods construct a set of simple models, called base learners, and use their weighted outcome (or vote) to predict new data. That is, ensemble methods combine outputs from multiple base learners to form a committee with improved performance. Numerous empirical studies confirm that ensemble methods often outperform any single base learner (Freund and Schapire, 1996; Bauer and Kohavi, 1999; Dietterich, 2000a). The improvement can be dramatic when a base algorithm is unstable. More recently, a series of theoretical developments (Bousquet and Elisseeff, 2001; Poggio et al., 2002; Mukherjee et al., 2006; Poggio et al., 2004) also confirmed the fundamental role of stability for the generalization of a learning algorithm. More comprehensive overviews of ensemble methods were presented by Dietterich (2000b) and Valentini and Masulli (2002). There are two primary approaches to ensemble construction: parallel and serial.

A parallel ensemble combines independently constructed and diverse base learners. That is, different base learners should make different errors on new data. An ensemble of such base learners can outperform any single one of its components since diverse errors cancel out (Hansen and Salamon, 1990; Amit and Geman, 1997). Parallel ensembles are variance-reduction techniques, and in most cases, they are applied to unstable, high-variance algorithms (such as trees). Also, Valentini and Dietterich (2003) showed that ensembles of low-bias support vector machines (SVMs) often outperformed a single, best-tuned, canonical SVM (Boser et al., 1992).

Random forest (RF) is an exemplar for parallel ensembles (Breiman, 2001). It is an improved bagging method (Breiman, 1996) that extends the "random subspace" method (Ho, 1998). It grows a forest of random decision trees on bagged samples showing excellent results comparable with the best known classifiers. A RF can be summarized as follows: (1) Grow each tree on a bootstrap sample of the training set to maximum depth, (2) Given M predictors, select at random $m < M$ variables at each node, and (3) Use the best split selected from the possible splits on these m variables. Note that for every tree grown in RF, about one-third of the cases are out-of-bag (out of the bootstrap sample). The out-of-bag (OOB) samples can serve as a test set for the tree grown on the non-OOB data. We discuss later how OOB samples can be used for feature selection.

In serial ensembles, every new learner relies on previously built learners so that the weighted combination forms an accurate model. A serial ensemble algorithm is often more complex. It is targeted to reduce both bias and variance. A serial ensemble results in an additive model built by a forward-stagewise algorithm. The *adaboost* algorithm was introduced by Freund and Schapire (1996). At every step of ensemble construction the boosting scheme adds a new base learner that is forced (by iteratively reweighting the training data) to concentrate on the training observations that are misclassified by the previous sequence. Boosting showed dramatic improvement in accuracy even with very weak base learners (like decision stumps, single split trees). Breiman (1998) and Friedman et al. (2000) showed that the adaboost algorithm is a form of gradient optimization in functional space, and is equivalent to a forward-stagewise, additive algorithm with the exponential loss function $\Psi(y,F(\mathbf{x})) = \exp(-yF(\mathbf{x}))$ referred to as a gradient boosted tree (GBT).

3.1 Relative Variable Importance Metrics

A single decision tree partitions the input space into a set of disjoint regions, and assigns a response value to each corresponding region. It uses a greedy, top-down recursive partitioning strategy. At every step an exhaustive search is used to test all variables and split points to achieve the maximum reduction in impurity. Therefore, the tree constructing process itself can be considered as a type of variable selection (a kind of forward selection, embedded algorithm), and the impurity reduction due to a split on a specific variable indicates the relative importance of that variable to the tree model (Breiman et al., 1984). For ensembles, the metric is averaged over the collection of base learners. Note, that this relative importance automatically incorporates variable interaction effects thus being very different from the relevance measured by a univariate filter method.

For a single decision tree the measure of variable importance is

$$VI(X_i,T) = \sum_{t \in T} \Delta I(X_i,t), \tag{1}$$

where $\Delta I(X_i,t)$ is the decrease in impurity due to an actual (or potential) split on variable X_i at a node t of the optimally pruned tree T (Breiman et al., 1984). Node impurity $I(t)$ for regression is defined as $\sum_{i \in t}(y_i - \bar{y})^2/N(t)$ where the sum and mean are taken over all observations i in node t, and $N(t)$ is the number of observations in node t. For classification $I(t) = Gini(t)$ where $Gini(t)$ is the Gini index of node t defined as

$$Gini(t) = \sum_{i \neq j} p_i^t p_j^t,$$

and p_i^t is the proportion of observations in t whose response label equals i ($y = i$) and i, j run through all response class numbers. The Gini index is in the same family of functions as *cross-entropy* $= -\sum_i p_i^t log(p_i^t)$, and measures node impurity. It is zero when t has observations only from one class, and is maximum when classes are perfectly mixed. The decrease $\Delta I(X_i,t)$ computes the impurity at the node t and the weighted average of impurities at each child node of t. The weights are proportional to the number of observations that are assigned to each child from the split at node t so that $\Delta I(X_i,t) = I(t) - p_L I(t_L) - p_R I(t_R)$.

For an ensemble of M trees this importance measure is easily generalized. It is simply averaged over the trees

$$E(X_i) = \frac{1}{M} \sum_{m=1}^{M} VI(X_i, T_m). \tag{2}$$

The averaging makes this measure more reliable.

This split weight measure $\Delta I(X_i,t)$ in Equation (1) can be improved if OOB samples are used. The split value for a variable is calculated using the training data as usual. However, the variable selected as the primary splitter uses only the OOB samples. Also, the variable importance measure is calculated from only the OOB samples. This provides a more accurate and unbiased estimate of variable importance in each tree and improves the filtering of noise variables.

Breiman (2001) also proposed a *sensitivity* based measure of variable relevance evaluated by a RF. For a classification problem it is summarized as follows: (1) Classify the OOB cases and count the number of votes cast for the correct class in every tree grown in the forest, (2) randomly permute the values of variable m in the OOB cases and classify these cases down the tree, (3) Subtract the number of votes for the correct class in the variable-m-permuted OOB data from the original OOB data, and (4) Average this number over all trees in the forest to obtain the raw importance score for variable m. Similar ideas were presented by Parmanto et al. (1996) and a similar resampling strategy

was successfully used in a more traditional model by Wisnowski et al. (2003). The sensitivity measure is computationally expensive. Furthermore, it does not account for masking, nor does it consider an iterative process with residuals (that we describe in Sec. 4.2). Experiments by Tuv (2006) demonstrated that weaker but independent predictors can rank higher than stronger, but related predictors. Also, related predictors can all be identified as important. Neither of these results are desirable for a best subset model and a more effective algorithm is described in Sec. 4.

With the importance measure (2) we can thus merely rank the variables. The following two subsections discuss how to amend the ranking so that irrelevant variables can be reliably detected, and how the redundancies among the remaining relevant variables can then be handled.

3.2 Removing Irrelevant Features by Artificial Contrasts

Although an ensemble can be used to calculate a relative feature ranking from the variable importance score in (2) the metric does not separate relevant features from irrelevant. Only a list of importance values is produced without a clear indication which variables to include, and which to discard. Also, trees tend to split on variables with more distinct values. This effect is more pronounced for categorical predictors with many levels. It often makes a less relevant (or completely irrelevant) input variable more "attractive" for a split only because it has high cardinality.

The variable importance score in (2) is based on the relevance of an input variable to the target. Consequently, any stable feature ranking method should favor a relevant input X_i over an artificially generated variable with the same distribution as X_i but generated to be irrelevant to the target. That is, a higher variable importance score is expected from a true relevant variable than from an artificially generated contrast variable. With sufficient replicates in an analysis one can select important variables from those that have statistically significantly higher variable importance scores than the contrast variables (Tuv et al., 2006). Here, these contrast variables are integrated into a subset algorithm. We discuss this in detail in Section 4.

Also, artificial contrasts can be applied to masking discussed in the next subsection. Given a selected subset of relevant variables, one computes the masking scores of all variables by elements of this subset, and the masking of contrast variables by this subset. Masking scores statistically higher than the contrast variables are considered to be real masking. Variables that are masked are dropped from the relevant subset list over a sequence of iterations of the algorithm.

3.3 Masking Measures

An important issue for variable importance in tree-based models is how to evaluate or rank variables that were masked by others with slightly higher splitting scores, but could provide as accurate a model if used instead. One early approach in the CART methodology used surrogate splits (Breiman et al., 1984). The predictive association of a surrogate variable X^s for the best splitter X^* at a tree node T is defined through the probability that X^s predicts the action of X^* correctly and this is estimated as

$$p(X^s, X^*) = p_L(X^s, X^*) + p_R(X^s, X^*),$$

where $p_L(X^s, X^*)$ and $p_R(X^s, X^*)$ define the estimated probabilities that both X^s and X^* send a case in T left (right). The predictive measure of association $\lambda(X^*|X^s)$ between split X^s and primary split X^* is defined as

$$\lambda(X^*|X^s) = \frac{\min(\pi_L, \pi_R) - [1 - p(X^s, X^*)]}{\min(\pi_L, \pi_R)},$$

where π_L, π_R are the proportions of cases sent to the left(or right) by X^*. It measures the relative reduction in error $(1 - p(X^s, X^*))$ due to using X^s to predict X^* as compared with the "naive" rule that matches the action with $\max(\pi_L, \pi_R)$ (with error $\min(\pi_L, \pi_R)$). If $\lambda(X^*|X^s) < 0$ then X^s is disregarded as a surrogate for X^*. Sometimes a small, nonnegative threshold is used instead. The variable importance sum in Equation (1) is taken over all internal tree nodes where X_i is a primary splitter or a surrogate variable ($\lambda(X^*|X_i) > 0$ for a primary splitter X^*). Often a variable that does not appear as a primary splitter in a tree is still ranked high on the variable importance list constructed using surrogate variables.

We extend the surrogate concept to define a masking score as follows. Variable i is said to mask variable j in a tree, if there is a split in variable i in a tree with a surrogate on variable j. We define the masking measure for a pair of variables i, j in tree T as

$$M_{ij}(T) = \sum_{\{t \in T | \text{split on } X_i\}} w(X_i, t)\lambda(X_i|X_j),$$

where $w(X_i, t) = \Delta I(X_i, t)$ is the decrease in impurity from the primary split on variable X_i, and summation is done over the nodes where primary split was made on variable X_i. Here we take into account both the similarity between variables X_i, X_j at the node, and the contribution of the actual split of variable X_i to the model. For an ensemble the masking measure is simply averaged over the trees. Note that in general the measure is not symmetric in the variables. One variable may mask several others, but for a single selected masked variable the reverse may not be true.

4. Algorithm: Ensemble-Based Feature Selection with Artificial Variables and Redundancy Elimination

We now integrate the previously described concepts and metrics into a subset selection algorithm. The fundamental steps outlined in Section 2.3 consist of using the advantages of a parallel ensemble to detect important variables among potentially a very large feature set, using the advantages of a serial ensemble to de-mask the important variables, and calculating residuals and repeating in order to recover variables of secondary importance.

Within the algorithm, artificial contrast variables are re-generated a number of times. Then the significance from a paired t-test over the replicates is used to identify important variables and masked variables. Essentially the t-test is used to define thresholds for selection and masking. These thresholds could also be set as tunable parameters. An advantage of the statistical test is that the significance of selected variables relative to noise can be quantified.

4.1 Algorithm Details

1. **Identify Important Variables:** Artificially generated noise variables are used to determine a threshold to test for statistically significant variable importance scores. The test is used to remove irrelevant variables. Details are presented in the displayed algorithms and further described as follows.

   In each replicate r, $r = 1, 2, \ldots, R$ artificial variables are constructed as follows. For every real variable X_j $j = 1, 2, \ldots, M$ a corresponding artificial variable Z_j is generated from a random permutation. Then in each replicate a small RF is trained and variable importance scores are computed for real and artificial variables. The scores from each replicate r are compiled

into the rth row of a matrix $\mathbf{V}$ $R \times 2M$. Furthermore, the $1-\alpha$ percentile of the importance scores in replicate r is calculated from only the artificial variables. This is denoted as v_r and the vector of percentiles over the R replicates is $\mathbf{v}$ $R \times 1$. For each real variable X_j a paired t-test compares importance scores for X_j (obtained from the jth column of $\mathbf{V}$) to the vector of scores $\mathbf{v}$. A test that results in statistical significance identifies an important variable.

Significance is evaluated through a suitably small p-value. The use of a p-value requires a feature to consistently score higher than the artificial variables over multiple replicates. Furthermore, this statistical testing framework also allows any method to control false selections to be applied. We routinely use the Bonferroni adjustment, but a false discovery rate approach is also reasonable. Each replicate uses a RF with $L =$ 20-50 trees to score the importance of the original and artificial noise variables. Also, the split weight calculation for variable importance in (2) only uses OOB samples as described previously.

2. **Calculate Masking Scores:** A masking matrix is computed from independent replicates in order to evaluate the statistical significance of masking results. Suppose there are m important variables from step 1. For similar reasons as in the previous step, replicates and noise variables are used to detect masking among the relevant variables. These are currently the same replicates that are used for variable importance. A set of R independent GBT models are generated each with $L =$ 10-50 trees. Note that all variables are tested in each node in each tree in a serial ensemble. Therefore, richer, more effective masking information is obtained from a serial ensemble than from a random subspace method like RF. In these calculations, the surrogate scores and the split weights are calculated from the OOB samples as in the previous step. Let $M^r_{i,j}$ denote the masking score for variables X_i and X_j from the ensemble in replicate r, for $r = 1,2,\ldots,R$. Also, let $M^r_{i,\alpha}$ denote the $(1-\alpha)$-percentile of the masking score in replicate r from the distribution of scores between variable X_i and the noise variables. That is, $M^r_{i,\alpha}$ denotes the $(1-\alpha)$-percentile of $M^r_{i,j}$ for $j = m+1,\ldots,2m$. Similar to the check for variable importance, a paired t-test compares the masking score between variables (X_i, X_j) with masking score $M^r_{i,\alpha}$ computed from the noise variables. There is a significant masking between variables (X_i, X_j) if the paired t-test is significant. Variable X_j is masked by variable X_i if the test is significant.

3. **Eliminate Masked Variables:** Masked variables are removed from the list of important variables as follows. Given a list of important variables upon entry to this step, the variables are sorted by the importance score calculated in step 2. The most important variable is added to an exit list, and dropped from the entry list. Assume this is variable X_i. All variables that are masked by X_i are dropped from the entry list. This is repeated until the entry list is empty. The exit list represents the unmasked important variables.

4. **Generate Residuals for Incremental Adjustment:** An iteration is used to enhance the ability of the algorithm to detect variables that are important, but possibly weaker than a primary set. Given a current subset of important variables, only this subset is used to predict the target. Residuals are calculated and form a new target. For a numerical target the residuals are simply the actual minus the predicted values. For a classification problem residuals are calculated from a multiclass logistic regression procedure (Friedman et al., 2000). We predict the log-odds of class probabilities for each class (typically GBT is used), and then take

pseudo residuals as summarized in the following multi-class logistic regression algorithm. The algorithms are described using the notation in Table 1.

The iterations are similar to those used in forward selection. See, for example, Stoppiglia et al. (2003). The Gram-Schmidt procedure first selects the variable with highest correlation with the target. To remove the information from this variable the remaining predictors and the target are othogonalized with respect to the selected variable. This provides residuals from the fit of the target to the first selected variable. In the feature selection method here we do not require orthogonal predictors, but we adjust the target for the variables already selected through residuals. We also can select more than a single variable in each iteration. The method also uses a conservative selection criterion (Bonferroni adjustment) and the residuals allow a variable to enter on another iteration. There are similar procedures used elsewhere in regression model building. Least angle regression (Efron et al., 2004) and projection pursuit methods (Friedman et al., 1981) are well known examples that use residuals in forward-stagewise modeling.

The algorithm returns to step 1 and continues until no variables with statistically significant importance scores remain. The current subset of important variables is used for the prediction model. Whenever step 1 is calculated, all variables are used to build the ensembles—not only the currently important ones. This approach allows the algorithm to recover partially masked variables that still contribute predictive power to the model. This can occur after the effect of a masking variable is completely removed, and the partial masking is eliminated. The algorithms for numerical (regression) and categorical (classification) targets are presented as algorithms 1 and 2. A separate algorithm 3 describes the variable masking calculations.

Algorithm 1: Ensemble-Based Feature Selection, Regression

1. Set $\Phi \leftarrow \{\}$; set $F \leftarrow \{X_1, \ldots, X_M\}$; set $W = 0$ $(|W| = M)$.
2. for $r = 1, \ldots, R$ do
3. $\quad \{Z_1, \ldots, Z_M\} \leftarrow$ permute$\{X_1, \ldots, X_M\}$
4. $\quad$ set $F_P \leftarrow F \cup \{Z_1, \ldots, Z_M\}$
5. $\quad r^{th}$ row of $\mathbf{V}$ = $\mathbf{V}_{r.} = g_I(F_P, Y)$;

   endfor
6. $R \times 1$ vector (element wise) $\mathbf{v} = Percentile_{1-\alpha}(\mathbf{V}[\cdot, M+1, \ldots, 2M])$
7. Set $\hat{\Phi}$ to those $\{X_j\}$ for which element wise $\mathbf{V}_{.j} > \mathbf{v}$

   with specified paired t-test significance (0.05)
8. Set $\hat{\Phi} = RemoveMasked(\hat{\Phi}, W + g_I(F_P, Y))$
9. If $\hat{\Phi}$ is empty, then quit.
10. $\Phi \leftarrow \Phi \cup \hat{\Phi}$;
11. $Y = Y - g_Y(\hat{\Phi}, Y)$
12. $W(\hat{\Phi}) = W(\hat{\Phi}) + g_I(\hat{\Phi}, Y)$
13. Go to 2.

Algorithm 2: Ensemble-Based Feature Selection, Classification

1. set $\Phi \leftarrow \{\}$; $G_k(F) = 0, W_k = 0$
2. for $k = 1, \ldots, K$ do
3. set $V = 0$.
4. for $r = 1, \ldots, R$ do
   $\{Z_1, \ldots, Z_M\} \leftarrow$ permute$\{X_1, \ldots, X_M\}$
   set $F \leftarrow X \cup \{Z_1, \ldots, Z_M\}$
   Compute class proportion $p_k(x) = exp(G_k(x))/\sum_{l=1}^{K} exp(G_l(x))$
   Compute pseudo-residuals $Y_i^k = I(Y_i = k) - p_k(x_i)$
   $\mathbf{V}_{r.} = \mathbf{V}_{r.} + g_I(F, Y^k)$;
   endfor
5. Element wise $\mathbf{v} = Percentile_{1-\alpha}(\mathbf{V}[\cdot, M+1, \ldots, 2M])$
6. Set $\hat{\Phi}_k$ to those $\{X_k\}$ for which $\mathbf{V}_{.k} > \mathbf{v}$
   with specified paired t-test significance (0.05)
7. Set $\hat{\Phi}_k = RemoveMasked(\hat{\Phi}_k, W_k + g_I(F, Y^k))$
8. $\Phi \leftarrow \Phi \cup \hat{\Phi}_k$;
   for $k = 1, ..., K$ do
9. $G_k(F) = G_k(F) + g_Y(\hat{\Phi}_k, Y^k)$
10. $W_k(\hat{\Phi}_k) = W_k(\hat{\Phi}_k) + g_I(\hat{\Phi}_k, Y^k)$
    endfor
    endfor
11. If $\hat{\Phi}_k$ for all $k = 1, \ldots, K$ is empty, then quit.
12. Go to 2.

4.2 Comparison to Previous Work

Two earlier methods are closely related to ACE, FCBS (Yu and Liu, 2004) and MBBE (Koller and Sahami, 1996). We compare our method in detail to these two methods. Because we use a multivariate model (tree) instead of frequency tables, our method fits in the category of embedded methods. This is unlike FCBS and MBBE that can be considered as correlation filters, although Koller works with frequency tables of 2-5 variables.

FCBS first sorts features by correlation with the response using a symmetric uncertainty, optionally removing the bottom of the list by a user-specified threshold, then

1. The feature most correlated to the response is selected.

2. All features that have correlation with the selected feature higher than it's correlation with response are considered redundant and removed. The feature is added to the minimal subset (and this is an approximate heuristic for Markov blanket filtering).

3. Return to 1).

FCBS is similar in structure to our method, with the following important differences.

Algorithm 3: RemoveMasked(F,W)

1 Let $m = |F|$.
2. for $r = 1,\ldots,R$ do
3. $\{Z_1,\ldots,Z_m\} \leftarrow \text{permute}\{X_1,\ldots,X_m\}$
4. set $F_P \leftarrow F \cup \{Z_1,\ldots,Z_m\}$
5. Build GBT model $G_r = GBT(F_P)$.
6. Calculate masking matrix $M^r = M(G_r)$ ($2m \times 2m$ matrix).
endfor
7. Set $M^r_{i,\alpha_m} = Percentile_{1-\alpha_m}(M^r[i, m+1,\ldots,2m]),\ r = 1,\ldots,R$
8. Set $M^*_{ij} = 1$ for those $i, j = 1\ldots m$ for which $M^r_{ij} > M^r_{i,\alpha_m},\ r = 1,\ldots,R$ with specified paired t-test significance (0.05), otherwise set $M^*_{ij} = 0$
9. Set $L = F, L^* = \{\}$.
10. Move $X_i \in L$ with $i = \text{argmax}_i W_i$ to L^*.
11. Remove all $X_j \in L$ from L, for which $M^*_{ij} = 1$.
12. Return to step 10 if $L \neq \{\}$.

1. We use tree importance instead of univariate correlation with the response. This makes ACE much more robust and accurate.

2. We use a surrogate masking measure instead of correlation. This takes the response into account, not only the correlations between inputs. No arbitrary thresholds for correlation are used.

3. We compute residuals to find smaller effects reducing the chance to drop a non-redundant feature.

Koller's MBBE works as follows:

1. For each feature X_i, find the set M_i of K features ($K = 1-4$) that are most correlated to it. (That is, which provide little information on the response when added to the selected feature in frequency table models.) Additional information is measured as KL-distance (Kullback and Liebler, 1951) $D(P(y|X_i,X_j),P(y|X_i))$. The set M_i is called the approximate Markov blanket for feature X_i. The authors state that $K = 1-2$ gives the best results.

2. For each feature compute the relevance score $\delta_i = D(P(y|M_i,X_i),P(y|M_i))$. This represents the additional information it brings when added to its approximate Markov blanket, and remove features that have the smallest relevance scores (i.e., most redundant).

3. Repeat (1,2) until all features are ranked in the order they are deleted. This method returns a ranked list of features rather than one subset.

Our ACE algorithm works more like FCBS as it uses only one feature as an approximate MB for each feature (as does the MBBE algorithm with $K = 1$). Furthermore, it filters features by relevance before computing redundancy between the features, and reports a final minimum feature subset.

K	Number of classes (if classification problem)
X	set of original variables
Y	target variable
M	Number of variables
R	Number of replicates for t-test
α	quantile used for variable importance estimation
α_m	quantile used for variable masking estimation
Z	permuted versions of X
W	cumulative variable importance vector.
W_k	cumulative variable importance vector for k-th class in classification.
F	current working set of variables
Φ	set of important variables
$\mathbf{V}$	variable importance matrix $(R \times 2M)$
$\mathbf{V}_{r.}$	rth row of variable importance matrix $\mathbf{V}$, $r = 1 \ldots R$
$\mathbf{V}_{.j}$	jth column of matrix $\mathbf{V}$
$g_I(F,Y)$	function that trains an ensemble of L trees based on variables F and target Y, and returns a row vector of importance for each variable in F
$g_Y(F,Y)$	function that trains an ensemble based on variables F and target Y, and returns a prediction of Y
$G_k(F)$	current predictions for log-odds of k-th class
$GBT(F)$	GBT model built on variable set F
$M(G)$	Masking measure matrix calculated from model G
M^k	Masking matrix for k-th GBT ensemble G_t.
M^*	Masking flags matrix

Table 1: Notation in Algorithms 1-3

However, the major difference is that our redundancy measure approximates KL-distance taking the response into account and uses local information. Thus, it can deal with multivariate dependencies. MBBE for $K > 1$ will incur three (or more) dimensional frequency tables that are hard to deal with if number of categories is large.

The learner $g(.,.)$ in the ACE algorithms is an ensemble of trees. Any classifier/regressor function can be used, from which the variable importance from all variable interactions can be derived. To our knowledge, only ensembles of trees can provide this conveniently.

The computational complexity of the algorithm is of the same order as the maximal complexity of a RF on the whole feature set and a GBT model on the selected important feature subset. A GBT model is usually more complex, because all surrogate splits at every tree node are computed. However, a smaller tree depth setting for the GBT model reduces the calculations in this part of the algorithm. The complexity is proportional to

$$(Fsel + Fimpvar) * N * logN * Ntrees * Nensembles * Niter + Niter * Fimpvar^2,$$

where the variables are defined as follows: *Niter* is the number of iterations of the ACE algorithm (for example, for the challenge discussed in Sec. 5.3 this was always less than 10 and usually 3-4); *Nensembles* is the number of replicates for t-tests (equal to 20 in the challenge); *Ntrees* is the number of trees in the RF or ensemble (equal to 20-100 in the challenge); N is the number of samples; *Fsel* is the number of selected variables per tree split in RF (equal to the square root of the total number features or less); *Fimpvar* is the number of selected important variables (for example, for the challenge data set NOVA discussed in Section 5.3 this was approximately 400-800 depending on parameters). The algorithm is very fast with approximately a minute for one feature selection iteration on the challenge NOVA data set with 16K variables with 20 replicates with 70 trees on a Windows XP-based four-processor Xeon (2 x HT) 3.4GHz workstation.

5. Experiments

In order to evaluate the goodness of feature selection algorithms, two options have been used in the literature. The first is not to evaluate the actual feature selection performance at all, but the performance of a subsequent learner in some task. This facilitates the use of any data set in the "evaluation" but does not give much useful information at all in characterizing the actual feature selection. The second option is to directly evaluate the feature selection performance without using a subsequent proxy task. The latter dictates the need to know the ground truth behind the data, which typically means that the data must be artificially generated, either completely, or by adding some redundant and/or irrelevant features to some known data.

As the topic of the paper at hand is a method for the subset feature selection, the first evaluation method is affected by the choice of the classifier. The effects of feature selection are mixed in with how well the learner is able to handle redundant or irrelevant features. The results would thus depend on the choice of learners and on the choice of data sets. Therefore we will mainly describe experiments with two types of simulated data with known ground truth.

Experiments on data with linear relationships are presented first. Then a nonlinear data generator is used to study the sensitivity to multiple variable interactions with nonlinear relations. Further results are from the 2007 International Joint Conference on Neural Networks (IJCNN), "Agnostic learning vs. prior knowledge challenge & data representation discovery workshop". The algorithm described here had the second best performance in the agnostic track. Here we demonstrate the effect of the subset to predictor performance as compared to the full set of features. Also, an actual manufacturing data set as well as a comparison to a previous analysis of the well known Hepatitis data are also presented in terms of predictive power of the resulting feature set.

5.1 Generated Data with Linear Relationships

The data in this experiment has an additive structure with one numeric response variable and 203 input variables. Inputs $x_1, \ldots, x_{100}$ are highly correlated with one another, and they are all reasonably predictive of the response (regression $R^2 \sim 0.5$). But a, b, and c are independent variables that are much weaker predictors (regression $R^2 \sim 0.1$). Further $u_1, \ldots, u_{100}$ are i.i.d. $N(0,1)$ variables that are unrelated to the target. The target variable was generated as an additive model with additional noise using $y = x_1 + a + b + c + \varepsilon$, where $\varepsilon \sim N(0,1)$. This structure is chosen because it is well known that linear (oblique) relationships are not optimal for a tree representation. However, they are ideal for correlation-based methods. Thus we have here the worst possible case for ACE and the best possible case for CFS. The methods were evaluated on 50 data sets of size 400 samples.

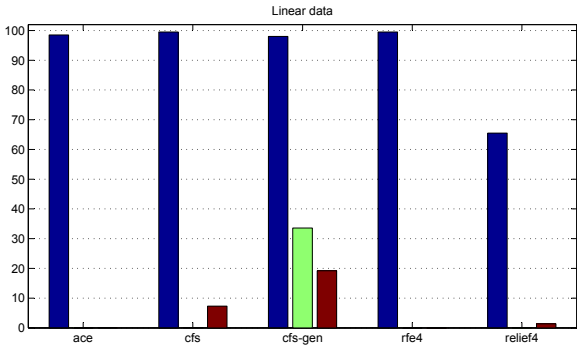

Figure 1: Artificial data with linear relationships. Subset discovery methods (ACE, CFS, CFS-gen) and methods finding a subset of predefined size four (RFE4, Relief4) are compared. The results for each method consist of three bars. The first is the percentage of relevant variables detected (out of four), the second is the percentage of redundant variables detected (out of 100), and the third is the percentage of irrelevant variables detected (out of 100). The results are averages over 50 data sets.

Figure 1 depicts the performance of ACE against methods that also discover the subsets (CFS with best-first search, CFS with genetic search), as well as against some subset ranking methods (RFE, Relief).

RFE and Relief are ranking methods. In this experiment they were given the advantage of knowing the number of relevant features beforehand, that is, their task was to "find the best possible four variable subset" (RFE4, Relief4), whereas ACE and CFS had to also find the number themselves. A further advantage was given to RFE by matching the underlying support vector regressor to the problem with a linear kernel (using the standard RBF kernel produced inferior results). This experiment demonstrates one aspect of the advantages of ACE. In a task ideal for correlation-based methods but hard for trees, we show equal performance.

5.2 Generated Nonlinear Data

Next, experiments were conducted using a well-known data generator (Friedman, 1999), which produces data sets with multiple non-linear interactions between input variables. The true model can be designed with relevant, redundant, and noise inputs. We selected 10 relevant inputs plus random, uniform (0, 1) noise. Also, 20 redundant inputs were used. Each was a random linear combination of three inputs plus random, uniform noise. Finally, 40 noise inputs were added, so that 70 features were available to the full model. The target function was generated as a weighted sum of 10 multidimensional Gaussians, each Gaussian at a time involving about four input variables randomly drawn from the relevant 10 variables. Thus all of the relevant 10 input variables are involved in the target, to a varying degree. The Gaussian functions also have a random mean vector and a random covariance matrix as described by Friedman (1999). Weights for the Gaussians were randomly drawn from $U[-1,1]$.

The data generator produces continuous-valued variables. Thus the data sets can be used as such for regression problems. Data sets of two different sizes were generated, 1000 and 4000 samples. In

order to generate classification problems, the target variable was discretized to two levels. Mixed-type data was generated by randomly discretizing half of the variables, each to a random number of levels drawn from $U[2,32]$. There are thus eight different experiments altogether. For each experiment, 50 data sets were generated with different seeds. Figure 2 presents the results for each case as average percentages of features selected in each group (relevant, redundant, or noise) over the 50 generated data sets.

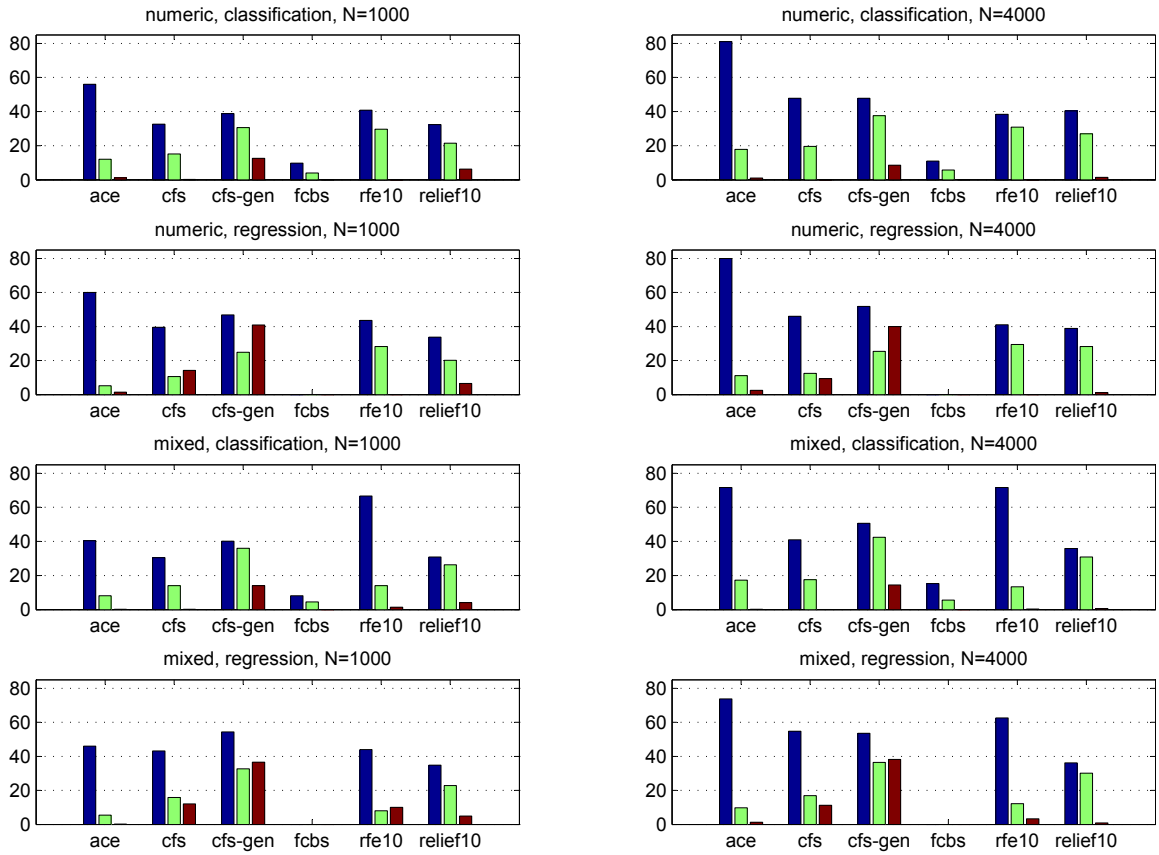

Figure 2: Artificial data with nonlinear relationships. Subset discovery methods (ACE, CFS, CFS-gen, FCBS) and methods finding a subset of predefined size 10 (RFE10, Relief10) are compared. FCBS works only in classification problems. The results for each method consist of three bars. The first is the percentage of relevant variables detected (out of 10), the second is the percentage of redundant variables detected (out of 20), and the third is the percentage of irrelevant variables detected (out of 40). The results are averages over 50 data sets.

RFE and Relief were again given the advantage of knowing the number of relevant features beforehand, that is, their task was to "find the best possible ten-variable subset", whereas ACE, CFS, and FCBS had to also find the number by themselves. A further advantage was given to RFE by matching the underlying support vector classifier to the problem with an RBF kernel. Using a linear kernel produced inferior results.

The notable failure of FCBS on this data can be explained as follows. Most numerical important variables are dropped at the discretization step of FCBS, because MDL discretization works as a filter method, and it cannot deal with the multivariate dependency from Friedmans's generator. It works well with discrete variables only when the number of categories is small and the response is categorical with a small number of categories.

This experiment demonstrates another aspect of the universality of ACE. The only case where another method (RFE10) showed a superior result was a classification problem with a smaller sample size and mixed type inputs. Again RFE10 was given the advantage of knowing the number of relevant features and an appropriate kernel beforehand.

5.3 IJCNN 2007 Agnostic Learning vs. Prior Knowledge Challenge

In this experiment we show the effect of the selected subset within various classification tasks. The ACE feature selection algorithm was applied to the data sets in the Agnostic Learning Challenge. The number of training/validation/testing instances and the number of features are shown in the following list:

- ADA, Marketing, 4147/415/41471, 48 features
- GINA, Handwriting recognition, 3153/315/31532, 970 features
- HIVA, Drug discovery, 3845/384/38449, 1617 features
- NOVA, Text, 1754/175/17537, 16969 features
- SYLVA, Ecology, 13086/1309/130857, 216 features

For feature selection with ACE, the number of trees, importance and masking quantiles were parameters that were optimized. Next GBT with embedded feature selection (to prevent overfitting) (Borisov et al., 2006) was built on the subset. The following parameters of GBT were optimized: number of trees, tree depth, shrinkage, number of selected features per tree node, and the importance adjustment rate for embedded feature selection, stratified sampling for 0/1 class proportions, and priors. The optimization strategy (manual) was to set reasonable parameter values, and then try to adjust each parameter sequentially, so that the test error decreased. The model was trained on 60% of the training data during parameter optimization. Several passes over all the GBT parameters were used, and one for the feature selection parameters. Priors were selected using cross validation. Feature selection and GBT were used on K partitions of the data and then optimal priors were selected on the remaining part.

Table 2 shows the results before and after subset selection for the five challenge data sets. The CV-error was either preserved or reduced through a good subset. The overall results were the second best in the agnostic learning challenge. Redundancy elimination was applied on ADA, HIVA, SYLVA, and feature selection without redundancy elimination was used on NOVA and GINA.

5.4 TIED Data Set

A data set with multiple Markov boundaries was generated by Statnikov and Aliferis (2009). The data was obtained from a discrete Bayesian network with 1000 variables and a target variable with four classes. A training set was constructed with 750 instances simulated from the network.

Original	Features	CV-error from all features	Best subset size	CV-error from selected subset
Ada	47	0.1909	16	0.1855
Gina	970	0.0527	75	0.0506
Hiva	1617	0.2847	221	0.2559
Nova	12993	0.0591	400	0.0518
Sylva	212	0.0133	69	0.0129

Table 2: IJCNN 2007 Agnostic Learning vs. Prior Knowledge Challenge results.

Variable	p-value	Importance Score
3	0	100.0%
2	0	98.4%
10	0	96.4%
1	1.E-10	96.4%
11	3.E-07	83.3%
12	2.E-07	83.3%
13	5.E-07	79.1%
18	3.E-09	67.5%
19	2.E-07	67.5%
15	2.E-07	41.4%
20	2.E-06	39.5%
29	2.E-06	29.8%
8	3.E-06	26.2%
14	1.E-08	11.6%
4	8.E-06	9.5%
9	6.E-06	8.3%

Table 3: Feature selection scores for the TIED data set. Variables in any Markov boundary are recovered as significant with three false alarms.

The network contained 72 Markov boundaries. Each boundary contained five variables (one from each of the following subsets):(1)$\{X_9\}$, (2) $\{X_4, X_8\}$, (3)$\{X_{11}, X_{12}, X_{13}\}$, (4) $\{X_{18}, X_{19}, X_{20}\}$, and (5) $\{X_1, X_2, X_3, X_{10}\}$.

The ACE feature selection method described here was used to remove irrelevant features. After three iterations of the residual calculations described previously the algorithm stopped with the important variables (and p-values from the artificial contrasts) shown in Table 3. The list of statistically significant variables reproduces all the variables in any of the Markov boundaries listed above, with false alarms from variables X_{14}, X_{15}, and X_{29}.

Although ACE recovered the variables in the Markov boundaries, there are limitations with the masking methods for a multi-class target. The GBT ensembles model each class (versus the rest) with a binary logistic function and averages variable masking scores over the binary models. Consequently, some attenuation of the importance scores are expected. Redundancy elimination did not effectively eliminate masking in the TIED data. However, we used the TIED network and TIED

data for binary problems with each class versus the rest. For example, for class 1 versus the rest the TIED network generates the same collection of 72 Markov boundaries. The results from ACE without redundancy elimination for the binary target are shown in Table 4. The list of statistically significant variables reproduces all the variables in any of the Markov boundaries, with no false alarms.

Variable	Importance Score
4	100.0%
8	100.0%
19	88.1%
18	88.1%
20	88.1%
9	64.8%
13	39.5%
12	39.5%
11	39.5%
10	21.9%
2	21.9%
3	21.9%
1	21.9%
6	0.0%

Table 4: Variable importance for TIED data modified for a binary target (class 1 versus the rest). All variables in the true Markov boundaries are identified with no false alarms.

As the importance scores are arranged in decreasing order in Table 4, groups of variables with similar scores become noticeable and these groups correspond to the subsets (equivalence classes) in the cross-product that defines the Markov boundaries. That is, the most important variables in Table 4 are those in the subset $\{X_4, X_8\}$ in the Markov boundaries and the last group matches the subset $\{X_1, X_2, X_3, X_{10}\}$. The equivalent groups are clear from their importance scores in this case.

The analysis with redundancy elimination generated the list of significantly significant variables in Table 5. One equivalent variable from the subset $\{X_1, X_2, X_3, X_{10}\}$ was missed in the recovery of a Markov boundary. The contribution from this subset was, however, small. The predictive performance of a tree ensemble on the recovered variables is nearly identical to a model on a true Markov boundary. In addition, the three variables $\{X_{18}, X_{20}, X_4\}$ are identified in Table 5 as important, but they are redundant in the true network. Although these comprise false alarms, the magnitudes of the importance scores indicate that the last three variables are much less important than the others. Similar results (not shown here) were obtained for the binary target class 2 (versus the rest). Results without any errors were obtained for classes 0 and 3 (each versus the rest). Specifically, for class 0 the Markov boundaries from the TIED network consist of one element from $\{X_1, X_2, X_3, X_{10}\}$. In this case the ACE analysis without redundancy elimination recovered these four variables without false alarms. The analysis with redundancy elimination correctly recovered a single variable from this set. Similarly for class 3, without redundancy elimination all variables in the Markov boundaries $\{X_{12}, X_{13}, X_{14}\}$ were recovered, and only one variable from this set was recovered with redundancy elimination.

Variable	Importance Score
8	100.0%
9	61.3%
19	43.8%
1	10.2%
18	2.6%
20	0.9%
4	0.3%

Table 5: Variable importance for TIED data modified for a binary target (class 1 versus the rest) with redundancy elimination.

5.5 Manufacturing Data

In multiple real world applications collecting unnecessary variables is a cost issue and finding a suitable subset is critical in terms of cost-efficiency. As an example we present manufacturing data from a process that contained approximately 10K rows and consisted of 35 predictors that were all numerical, continuous measurements. The target was a binary response and approximately 20% of the data belonged in the rare class. Because the data is actual manufacturing data, the specific variable names are not provided. The data was analyzed extensively with traditional regression methods (the response was coded as 0 and 1) and models obtained were complex and not accurate. A list of the results from our algorithm is shown in Table 6. It is not unusual for manufacturing data to consist of related predictors. Without redundancy elimination, 20 variables were identified as related to the target. However, after masking scores were used to remove redundant predictors the final subset model consisted of only five predictors.

The predictive accuracy for the binary target was nearly identical using a GBT model with these 5 predictors to the full set of 35 predictors. Table 6 also compares other subset selection algorithms to ACE in terms of their predictive accuracy and the size of the selected feature set.

A previous analysis of this data by Berrado and Runger (2007) used association rules applied after the predictors were discretized with simple equal-frequency discretization. Only rules with consequent equal to the rare target class were considered. A total of 25 rules were detected that met the minimum support threshold. These rules contained 14 variables and 13 out of 14 are listed in the Table 6. Although the objectives of the association analysis were different, the relatively high proportion of important variables is consistent with the results in Table 6.

5.6 Hepatitis Data

The hepatitis data available from the UC-Irvine repository has been widely analyzed. There are 155 patients and 19 predictors and the response is a binary survival result. Breiman (2001) considered this data and cited a previous analysis from the Stanford Medical School and another analysis by Diaconis and Efron (1983). The analysis from the medical school concluded that the important variables were 6, 12, 14, 19. But Breiman (2001) concluded after a set of analyses that number 12 or 17 provided predictive power nearly equivalent to the full set of variables, and that these masked each other. A notable difficulty is the small sample size in this example.

Variables	ACE without redundancy elim.	ACE with redundancy elim.	CFS	CFS-gen	FCBS
V11	100.0%	72.4%	1	1	
V4	96.1%	100.0%	1	1	
V5	49.8%	49.4%	1	1	1
V12	48.6%		1	1	
V14	46.6%		1	1	
V10	43.5%		1	1	
V2	43.3%	36.4%	1	1	
V13	38.7%	21.6%			
V8	30.3%		1		
V1	27.9%			1	
V9	23.7%				
V3	23.6%			1	
V19	21.8%				
V7	21.5%				
V20	20.4%				
V26				1	
V27				1	
Errors		0.145	0.144	0.145	0.190

Table 6: Manufacturing data with a binary target with redundancy elimination excludes many variables. Only a smaller subset of the relevant predictors remain. We compare the extracted variables to other subset selection algorithms (selected variables are marked as '1' in the table). The error rate for the full set of variables was 0.146.

We confirmed the strong masking between variables 12 and 17 (and vice versa) from our masking matrix. We also obtained a subset model that consists of variables 6, 17, 14, 19, and 11, similar to medical school. Variable 11 was also identified in unpublished lecture notes by Breiman. The subset selected by our algorithm has the lowest cross-validation error using logistic regression.

6. Conclusions

We have presented an efficient method for feature subset selection that builds upon the known strengths of the tree ensembles and is designed explicitly to discover a non-redundant, effective subset of features in large, dirty, and complex data sets.

Our method attempts to eliminate irrelevant variables using statistical comparisons with artificial contrasts to obtain a threshold for importance estimated from the parallel ensembles of trees capable of scoring very large number of variables.

It uses serial ensembles to discover significant masking effects for redundancy elimination. Furthermore we have showed that the redundancy elimination based on feature masking approximates the Markov blanket redundancy filtering. It also uses an iterative strategy to allow for weaker predictors to be identified after stronger contributors.

Variables	ACE	CFS	CFS-gen	FCBS
malaise-6	1	1	1	
albumin-17	1			
bilirubin-14	1	1	1	
histology-19	1	1	1	
spiders-11	1	1	1	1
age-1		1	1	
sex-2		1	1	1
ascites-12		1	1	1
varices-13		1	1	
Errors	0.142	0.155	0.155	0.194

Table 7: Hepatitis data. Features selected from ACE compared to other subset selection algorithms (selected variables are marked as '1' in the table). The baseline error rate for the full set of variables was 0.148.

The superior performance of the algorithm is illustrated with a number of experiments on both artificial and real data as well as by its success in the agnostic learning challenge.

Acknowledgments

This material is partly based upon work supported by the National Science Foundation under Grant No. 0743160.

References

H. Almuallin and T. G. Dietterich. Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69(1-2):279–305, 1994.

Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–88, 1997.

E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *Machine Learning*, 36(1/2):525–536, 1999.

A. Berrado and G.C. Runger. Using metarules to organize and group discovered association rules. *Data Mining and Knowledge Discovery*, 14(3):409–431, 2007.

A. Borisov, V. Eruhimov, and E. Tuv. Tree-based ensembles with dynamic soft feature selection. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction Foundations and Applications: Studies in Fuzziness and Soft Computing*. Springer, 2006.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *5th Annual ACM Workshop on COLT, Pittsburgh, PA*, pages 144–152. ACM Press, 1992.

O. Bousquet and A. Elisseeff. Algorithmic stability and generalization performance. In *Advances in Neural Information Processing Systems*, volume 13, pages 196–202. MIT Press, 2001.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

L. Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–849, 1998.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, MA, 1984.

S. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10(1):57–78, 1993.

P. Diaconis and B. Efron. Computer intensive methods in statistics. *Scientific American*, (248): 116–131, 1983.

T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000a.

T. G. Dietterich. Ensemble methods in machine learning. In *First International Workshop on Multiple Classifier Systems 2000, Cagliari, Italy*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2000b.

B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *The 13th International Conference on Machine Learning*, pages 148–156. Morgan Kaufman, 1996.

J. Friedman. Greedy function approximation: a gradient boosting machine. *Technical report, Dept. of Statistics, Stanford University*, 1999.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28:832–844, 2000.

J. H Friedman, M. Jacobson, and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76:817–823, 1981.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, Mar 2003.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 359–366, 2000.

L. K. Hansen and P. Salamon. Neural network ensembles. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

T. K. Ho. The random subspace method for constructing decision forests. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

K. Kira and L. A. Rendell. A practical approach to feature selection. In *ML92: Proceedings of the ninth international workshop on Machine learning*, pages 249–256, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc. ISBN 1-5586-247-X.

D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of ICML-96, 13th International Conference on Machine Learning*, pages 284–292, Bari, Italy, 1996. URL `citeseer.nj.nec.com/koller96toward.html`.

S. Kullback and R.A. Liebler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:76–86, 1951.

H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. Knowledge and Data Eng.*, 17(4):491–502, 2005.

S. Mukherjee, P. Niyogi, T. Poggio, and R. Rifkin. Learning theory: Stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25:161–193, 2006.

B. Parmanto, P. Munro, and H. Doyle. Improving committee diagnosis with resampling techniques. In D. S. Touretzky, M. C. Mozer, and M. Hesselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 882–888. Cambridge, MA: MIT Press, 1996.

T. Poggio, R. Rifkin, S. Mukherjee, and A. Rakhlin. Bagging regularizes. In *CBCL Paper 214/AI Memo 2002-003*. MIT, Cambridge, MA, 2002.

T. Poggio, R. Rifkin, S. Mukherjee, and P. Niyogi. General conditions for predictivity in learning theory. *Nature*, 428:419–422, 2004.

M. Robnik-Sikonja and I. Kononenko. Theoretical and empirical analysis of relief and relieff. *Machine Learning*, 53:23–69, 2003.

C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29: 1213–1228, December 1986.

A. Statnikov and C.F. Aliferis. Tied: An artificially simulated dataset with multiple Markov boundaries. *Journal of Machine Learning Research Workshop Conference & Proceedings*, 2009. to appear.

H. Stoppiglia, G. Dreyfus, R. Dubois, and Y. Oussar. Ranking a random feature for variable and feature selection. *Journal of Machine Learning Research*, 3:1399–1414, March 2003.

E. Tuv. Ensemble learning and feature selection. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications*. Springer, 2006.

E. Tuv, A. Borisov, and K. Torkkola. Feature selection using ensemble based ranking against artificial contrasts. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2006.

G. Valentini and T. Dietterich. Low bias bagged support vector machines. In *ICML 2003*, pages 752–759, 2003.

G. Valentini and F. Masulli. Ensembles of learning machines. In M. Marinaro and R. Tagliaferri, editors, *Neural Nets WIRN Vietri-02*, Lecture Notes in Computer Science. Springer-Verlag, 2002.

J.W. Wisnowski, J.R. Simpson, D.C. Montgomery, and G.C. Runger. Resampling methods for variable selection in robust regression. *Computational Statistics and Data Analysis*, 43(3):341–355, 2003.

L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *J. of Machine Learning Research*, 5:1205–1224, 2004.