Statistics and Its Interface Volume 11 (2018) 503–513

Interpretable selection and visualization of features and interactions using Bayesian forests

Viktoriya Krakovna, Chenguang Dai, and Jun S. Liu*

In analysis of scientific data, it is often of interest to learn which features and feature interactions are relevant to the prediction task. We present here Selective Bayesian Forest Classifier, which strikes a balance between predictive power and interpretability by simultaneously performing classification, feature selection, feature interaction detection and visualization. It builds parsimonious yet flexible models using tree-structured Bayesian networks, and samples an ensemble of such models using Markov chain Monte Carlo. We build in its feature selection capability by dividing the trees into two groups according to their relevance to the outcome of interest. Our method performed competitively compared to top classification algorithms on both simulated data sets and real data sets in terms of classification accuracy, and often outperformed these methods in terms of feature selections and interaction visualizations.

Keywords and phrases: Feature selection, Interaction visualization, Bayesian forest.

1. INTRODUCTION

Feature selection and classification are key objectives in machine learning. Many approaches have been developed for these two problems, usually tackling them separately. Popular methods include Lasso [1], Random Forest [2], Naïve Bayes classifier [3], SVM [4], and neural networks. However, aiming only at predictions tends to produce black box solutions that are difficult to interpret, while performing feature selection alone can be difficult to justify without being validated by prediction. In addition to selecting for relevant features, it is also useful to detect interactions between them, and this problem becomes especially difficult in high dimensions. In many decision support systems, e.g. in medical diagnostics, the users care about which features and feature interactions contribute to a particular decision.

Many methods focus either on feature selection [1] [5] or on identifying feature interactions [2] [6] [7], or deal with these tasks in two independent steps. However, feature selection and feature interaction detection are often closely related. Without identifying feature interactions, feature selection can omit features that have weak marginal influence on the class label individually but have a large influence on it jointly. Selective Bayesian Forest Classifier (SBFC) combines predictive power and interpretability, by performing classification, feature selection, and feature interaction detection at the same time. Our method also provides a visual representation of the features and feature interactions that are relevant to the outcome of interest.

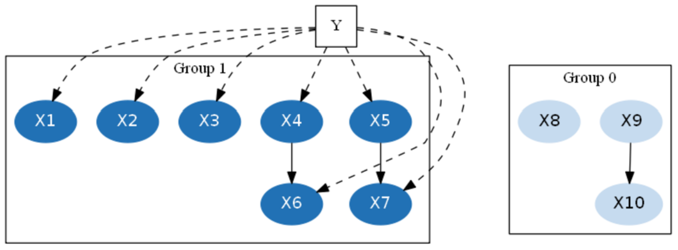


Figure 1. Example of a SBFC graph.

The main idea of SBFC is to construct an ensemble of Bayesian network models [8], each constrained to be a forest of trees divided into signal and noise groups based on their relationship with the class label Y (see Figure 1 for an example). The nodes and edges in Group 1 represent relevant features and interactions. SBFC is inspired by Naïve Bayes, an exceedingly simple yet surprisingly effective classifier, which assumes independence between the features conditional on the class label. Starting from the Naïve Bayes framework, we build dependence structures on the features. The features are partitioned into two groups based on their relationships with the class label, and the groups are further divided into independent subgroups, with each subgroup modeled by a tree structure. Such models are easy to sample using Markov chain Monte Carlo (MCMC). We combine their predictions using Bayesian model averaging, and aggregate their feature and interaction selections.

We show that SBFC outperforms state-of-the-art methods in terms of classification accuracy on both simulated data sets and some real data sets (Tables 2, 3 and 4). By adding noise features to the simulated data sets, we show that SBFC can reliably perform feature selection and interaction detection in both low and high dimensions (Figure 3 and 4). We use a high-dimensional data set from the NIPS 2003 feature selection challenge to demonstrate SBFC's superior performance on a difficult feature selection task (Figure 6). An R package `sbfc` is available on CRAN.

arXiv: 1506.02371
*Corresponding author.

2. RELATED WORK

2.1 Bayesian network model

The restrictive conditional independence assumption of NB often harms its performance when features are correlated. At the opposite extreme, we have the unrestricted Bayesian network model [8]. A Bayesian network (BN) is a directed acyclic graph (DAG) that encodes a joint probability distribution over $\boldsymbol{X}$. It contains two components: G and Θ, where G represents the DAG, and Θ stands for the parameters needed to describe the joint probability distribution. Each node represents a feature, and a directed edge corresponds to a "parent $\rightarrow$ child" dependence relationship between the features, so that each feature X_j is independent of its non-descendants given its parents Λ_j. The probability distribution over $\boldsymbol{X}$ can be written as

$$P(\boldsymbol{X}) = \prod_{j=1}^{d} P(X_j|\Lambda_j).$$

Bayesian network models present a tremendous computational challenge. Structure learning is NP-hard in the general case [9], as is exact inference [10]. The flexibility of the BN model is also its curse when the number of features is large, and the network structure can be difficult to interpret.

2.2 Tree-structured Bayesian methods

Tree structures are frequently used in computer science and statistics because they provide adequate flexibility to model complex structures, yet are constrained enough to facilitate computation. SBFC was inspired by tree-based methods such as Tree-Augmented Naïve Bayes (TAN) [11], Averaged One-Dependence Estimators (AODE) [12], and Hidden Naïve Bayes (HNB) [13], which relax the conditional independence assumption of NB to allow tree structures on the features.

TAN finds the optimal tree on all the features using the minimum spanning tree algorithm, with the class label Y as a second parent for all the features. The class label Y is the first parent of all the features, and each feature has another feature as its second parent, except for the root feature, which has only one parent Y. While the search for the best unrestricted Bayesian network is usually intractable [9], the computational complexity of TAN is only $O(d^2n)$, where d is the number of features and n is the sample size [14].

AODE constrains the model structure to a star-tree where all the features are children of the root feature, with Y as a second parent, and averages over models with all possible root features.

$$P(Y, \boldsymbol{X}) = P(Y, X_k) \prod_{j \neq k} P(X_j \mid Y, X_k).$$

HNB, an extension of AODE, designates a hidden parent X_{p_j} for each feature X_j, and assumes that the impact of this hidden parent on X_j is a weighted average of the impact of all the other features on X_j:

$$P(X_j \mid X_{p_j}, Y) = \sum_{k \neq j} w_{jk} P(X_j|X_k, Y), \ \sum_{k \neq j} w_{jk} = 1.$$

These methods put all the features into a single tree, which can be too restrictive and difficult to interpret, especially for high-dimensional data sets. We extend these methods by building forests instead of single-tree graphs, and introducing a selection of relevant features and interactions.

2.3 Feature selection for classification

While the above approaches focus on building a dependence structure, the following methods augment Naïve Bayes with feature selection. Selective Bayesian Classifier (SBC) [15] applies a forward greedy search method to select a subset of features to construct a Naïve Bayes model, while Evolutional Naïve Bayes (ENB) [16] uses a genetic algorithm with the classification accuracy as its fitness function.

More generally, one can use feature selection as a preprocessing step for any classification algorithm. Wrapper methods [17] select a subset of features tailored for a specific classifier, treating it as a black box. Feature Selection for Clustering and Classification (VSCC) [18] searches for a feature subset that simultaneously minimizes the within-class variance and maximizes the between-class variance, and remains efficient in high dimensions. Categorical Adaptive Tube Covariate Hunting (CATCH) [19] selects features based on a nonparametric measure of the relational strength between the feature and the class label.

Our approach, however, is to integrate feature selection into the classification algorithm itself, allowing it to influence the models built for classification. A classical example is Lasso [1], which performs feature selection using L_1 regularization. Some decision tree classifiers, like Random Forest [2] and BART [20], provide importance measures for features and the option to drop the least significant features. In many applications, it is also key to identify relevant feature interactions, such as epistatic effects in genetics. Interaction detection methods for gene association models include Graphical Gaussian models [21] and Bayesian Epistasis Association Mapping (BEAM) [22]. BEAM introduces a latent indicator that partitions the features into several groups based on their relationship with the class label. One of the groups in BEAM is designed to capture relevant feature interactions, but is only able to tractably model a small number of them. SBFC extends this framework, using tree structures to represent an unlimited number of relevant feature interactions.

Our work is similar to the Extended TAN algorithm (ETAN) [23], an extension of TAN that allows it to have forest structure and features disconnected from the class label. While ETAN uses a variation on the Edmonds algorithm for finding the minimum spanning forest, SBFC learns about forest structures and estimation/prediction uncertainties using MCMC.

3. SELECTIVE BAYESIAN FOREST CLASSIFIER (SBFC)

To combine feature selection and structure building, SBFC partitions the features based on their relation to the class label and builds tree structures within the partitions. It uses MCMC to sample from the space of these graph structures, and performs classification based on multiple sampled graphs via Bayesian model averaging.

3.1 Model

Given n observations with class label Y and d discrete features X_j, $j = 1, \ldots, d$, we divide the features into two groups (see Figure 1 for an example):

Group 0 (noise): features that are unrelated to Y
Group 1 (signal): features that are related to Y

We further partition each group into non-overlapping subgroups mutually independent of each other conditional on Y. For each subgroup, we infer a tree structure describing the dependence relationships among the features (many subgroups have only one feature and thus a trivial structure). The number of subgroups in each group is unknown *a priori* and will be inferred together with the tree structures via MCMC. Note that we impose such forest structures for both signal and the noise groups. Some previous work [22] assumes independence among the noise features, which is problematic when correlated noise features are present. Since the class label Y is a parent of every feature in Group 1, edges from Y are omitted in subsequent figures. We will refer to the combination of a group partition and the responding forest structure as a graph.

The prior distribution of the graph penalizes the number of edges between features in each group and the number of signal nodes (i.e., edges between features and Y) as follows:

$$P(G) \propto d^{-4(E_0(G)+E_1(G)/v)-D_1(G)/v},$$

where $D_i(G)$ is the number of nodes and $E_i(G)$ is the number of edges in Group i of graph G, while v is a constant equal to the number of classes.

The prior scales with the number of features d to penalize very large, hard-to-interpret, trees in high dimensional cases. The terms corresponding to the signal group are divided by the number of possible classes v, to avoid penalizing large trees in the signal group more than in the noise group by default. The coefficients in the prior were determined empirically to provide good classification and feature selection performance (there is a relatively wide range of coefficients that produce similar results).

Given the training data $\boldsymbol{X}_{(n\times d)}$ (with columns $\boldsymbol{X}_{\cdot j}$, $j = 1, \ldots, d$) and $\boldsymbol{y}_{(n\times 1)}$, we break down the graph likelihood according to the forest structure:

$$P(\boldsymbol{X}, \boldsymbol{y} \mid G) = P(\boldsymbol{y} \mid G) P(\boldsymbol{X} \mid \boldsymbol{y}, G)$$

Table 1. Parent sets for each feature type

Type of feature X_j	Parent set Λ_j
Group 0 root	$\emptyset$
Group 0 non-root	$\{X_{p_j}\}$
Group 1 root	$\{Y\}$
Group 1 non-root	$\{Y, X_{p_j}\}$

$$= P(\boldsymbol{y}) \prod_{j=1}^{d} P(\boldsymbol{X}_{\cdot j} \mid \boldsymbol{\Lambda}_j)$$

Here, Λ_j is the set of parents of X_j in graph G. This set includes the parent X_{p_j} of X_j unless X_j is a root, and Y if X_j is in Group 1, as shown in Table 1. We assume that the distributions of the class label Y and the graph structure G are independent *a priori*.

Let v_j and w_j be the number of possible values for X_j and Λ_j respectively. Then our hierarchical model for X_j is

$$[X_j \mid \Lambda_j = \lambda, \boldsymbol{\theta}_{j|\lambda}] \sim \text{Mult}(\boldsymbol{\theta}_{j|\lambda}),\ \lambda = 1, \ldots, w_j$$
$$\boldsymbol{\theta}_{j|\lambda} \sim \text{Dirichlet}\left(\frac{\alpha}{w_j v_j} \mathbf{1}_{v_j}\right)$$

Each conditional Multinomial model has a different parameter vector $\boldsymbol{\theta}_{j|\lambda}$. We consider the Dirichlet hyperparameters to represent "pseudo-counts" in each conditional model [11]. Let $n_{jk\lambda}$ be the number of observations in the training data with $X_j = k$ (here we denote the v_j values X_j can take as $1, \cdots, v_j$) and $\Lambda_j = \lambda$, and $n_{j\cdot\lambda} = \sum_{k=1}^{v_j} n_{jk\lambda}$. Then

$$P(\boldsymbol{X}_{\cdot j} | \boldsymbol{\Lambda}_j, \boldsymbol{\theta}_{j|1}, \ldots, \boldsymbol{\theta}_{j|w_j}) = \prod_{\lambda=1}^{w_j} \prod_{k=1}^{v_j} \theta_{jk|\lambda}^{n_{jk\lambda}}$$

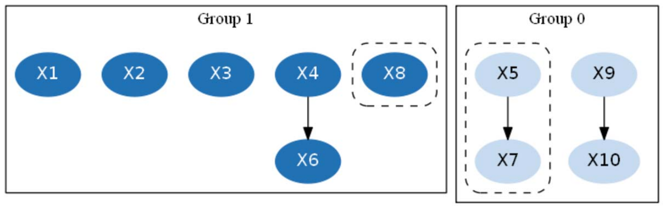
We then integrate out the nuisance parameters $\boldsymbol{\theta}_{j|\lambda}$, $\lambda = 1, \ldots, w_j$. The resulting likelihood depends only on the hyperparameter α and the counts of observations for each combination of values of X_j and Λ_j.

$$P(\boldsymbol{X}_j | \boldsymbol{\Lambda}_j) = \prod_{l=1}^{w_j} \frac{\Gamma\left(\frac{\alpha}{w_j}\right)}{\Gamma\left(\frac{\alpha}{w_j} + n_{j\cdot l}\right)} \prod_{k=1}^{v_j} \frac{\Gamma\left(\frac{\alpha}{w_j v_j} + n_{jkl}\right)}{\Gamma\left(\frac{\alpha}{w_j v_j}\right)}$$

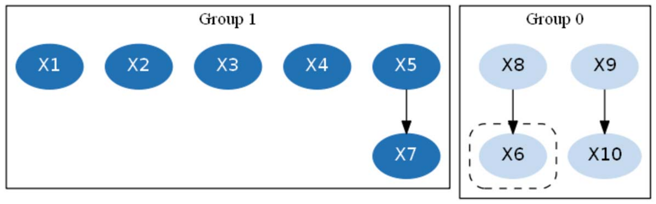
This is the Bayesian Dirichlet score, which satisfies the likelihood equivalence [9]. Namely, reparametrizations of the model that do not affect the conditional independence relationships between the features, such as pivoting a tree to a different root, do not change the likelihood.

3.2 MCMC updates

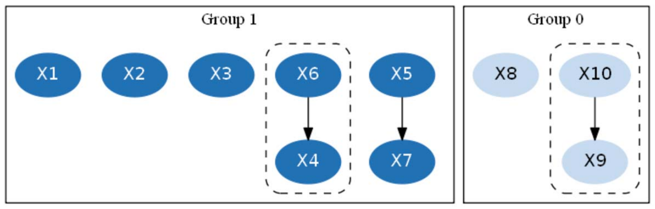
Switch Trees: Randomly choose trees $T_1, \ldots, T_k$ without replacement (we use $k = 10$, and propose switching each tree to the opposite group one by one (see Figure 2a). This is a repeated Metropolis update.



(a) Switch Trees: switch tree $\{X_5, X_7\}$ to Group 0, switch tree $\{X_8\}$ to Group 1]



(b) Reassign Subtree: reassign node X_6 to be a child of node X_8



(c) Pivot Trees: nodes X_6 and X_{10} become tree roots

Figure 2. Example MCMC updates applied to the graph in Figure 1.

Reassign Subtree: Randomly choose a node X_j, detach the subtree rooted at this node and choose a different parent node for this subtree (see Figure 2b). This is a Gibbs update, so it is always accepted.
We consider the set of nodes $X_{j'}$ that are not descendants of X_j as candidate parent nodes (to avoid creating a cycle), with corresponding graphs $G_{j'}$. We also consider a "null parent" option for each group, where X_j becomes a root in that group, with corresponding graph $\tilde{G}_i$ for group i. Choose a graph G^* from this set according to the conditional posterior distribution $\pi(G^*)$ (conditioning on the parents of all the nodes except X_j, and on the group membership of all the nodes outside the subtree). The subtree joins the group of its new parent.
As a special case, this results in a tree merge if X_j was a root node, or a tree split if X_j becomes a root (i.e. the new parent is null). Note that the new parent can be the original parent, in which case the graph does not change.

Pivot Trees: Pivot all the trees by randomly choosing a new root for each tree (see Figure 2c). By likelihood equivalence, this update is always accepted.

For computational efficiency, in practice we do not pivot all the trees at each iteration. Instead, we just pivot the tree containing the chosen node X_j within each Reassign Subtree move, since this is the only time the parametrization of a tree matters. This implementation produces an equivalent sampling mechanism.

3.3 Detailed balance for MCMC updates

3.3.1 Switch trees

Let $\pi(G) = P(G|X, y)$ be the target distribution, and let p be the probability that tree T is among the k trees proposed for switching - this only depends on k and the number of trees in the graph, which is constant during the Switch update. Let G be the starting graph when proposing to switch T, and G^* be the graph obtained from G by switching T. Then,

$$\begin{aligned}
&\pi(G)P(G \to G^*)\\
=&\pi(G)P(G \to G^* \text{ proposed})\\
&\cdot P(G \to G^* \text{ accepted} \mid G \to G^* \text{ proposed})\\
=&\pi(G) \cdot p \cdot \min\left(1, \frac{\pi(G^*)p}{\pi(G)p}\right)\\
=&p \min\left(\pi(G), \pi(G^*)\right)
\end{aligned}$$

Since this expression is symmetric in G and G^*, we can conclude that the detailed balance condition holds:

$$\pi(G)P(G \to G^*) = \pi(G^*)P(G \to G).$$

3.3.2 Reassign subtree

Let G be the starting graph. We pick any node X, and choose a non-descendant (or null) node to be its parent. Suppose X has c descendants - then the number of possible resulting graphs G_i is $d - c$ (including G itself). The backtracking update from G^* is reassigning X again, which results in the same possible graphs G_i.

$$\begin{aligned}
\pi(G)P(G \to G^*) &= \pi(G)P(G \to G^* \text{ proposed})\\
&= \pi(G)\frac{1}{d}\frac{\pi(G^*)}{\sum_{i=1}^{d-c}\pi(G_i)}\\
&= \pi(G^*)\frac{1}{d}\frac{\pi(G)}{\sum_{i=1}^{d-c}\pi(G_i)}\\
&= \pi(G^*)P(G^* \to G)
\end{aligned}$$

3.3.3 Pivot trees

Let G be the starting graph when pivoting tree T of size $|T|$. Let G^* be the graph obtained by pivoting T. By likelihood equivalence, $\pi(G) = \pi(G^*)$, so the proposal is always accepted.

$$\pi(G)P(G \to G^*) = \pi(G)P(G \to G^* \text{ proposed})$$

$$= \pi(G)\frac{1}{|T|} = \pi(G^*)\frac{1}{|T|}$$
$$= \pi(G^*)P(G^* \to G)$$

3.4 Classification using Bayesian model averaging

Graphs are sampled from the posterior distribution using the MCMC algorithm. We apply Bayesian model averaging [24] rather than using the posterior mode for classification. Since each graph structure gives us a predictive probability for each test sample to belong to each possible class, we average these predictive probabilities over a thinned subset of the sampled graph structures, and then choose the class label with the highest average probability. Given a test sample with feature $\boldsymbol{x}^{new}$, we find

$$P(Y = y \mid \boldsymbol{X} = \boldsymbol{x}^{new}, X, \boldsymbol{y})$$
$$\propto \sum_{i=1}^{S} P(Y = y \mid \boldsymbol{X} = \boldsymbol{x}^{new}, G_i)P(G_i \mid X, \boldsymbol{y})$$

where S is the number of graphs sampled by MCMC (after thinning by a factor of 50).

Given a graph G, we let R_1 denote the set of root nodes in Group 1, and let E_1 be the set of non-root ("edge") nodes in Group 1. Let p_i be the parent of feature x_i. For the test sample $\boldsymbol{x}^{new}$, we need to approximate the posterior predictive probability for $Y^{new} = c$ for each class label c. Let the training data be $(X, \boldsymbol{y})$, with sample size N. Then we have

$$\begin{aligned}
&P(Y^{new} = c \mid \boldsymbol{x} = \boldsymbol{x}^{new}, G, X, \boldsymbol{y}) \\
\propto &P(Y^{new} = c)P(\boldsymbol{x} = \boldsymbol{x}^{new} \mid Y^{new} = c, G, X, \boldsymbol{y}) \\
\approx &\frac{\#(y_i = c)}{N} \prod_{j \in R_1} P(x_j = x_j^{new} \mid Y^{new} = c, X, \boldsymbol{y}) \\
&\times \prod_{j \in E_1} P(x_j = x_j^{new} | x_{p_j} = x_{p_j}^{new}, Y^{new} = c, X, \boldsymbol{y}),
\end{aligned}$$

where we can estimate $P(x_j = x_j^{new} \mid Y^{new} = c, X, \boldsymbol{y})$ by

$$\frac{\#(x_{ij} = x_j^{new}, y_i = c)}{\#(y_i = c)},$$

and estimate $P(x_j = x_j^{new} | x_{p_j} = x_{p_j}^{new}, Y^{new} = c, X, \boldsymbol{y})$ by

$$\frac{\#(x_{ij} = x_j^{new}, x_{ip_j} = x_{p_j}^{new}, y_i = c)}{\#(x_{ip_j} = x_{p_j}^{new}, y_i = c)}.$$

In practice, instead of using the raw counts as in the above estimation, we typically add α in the denominator and $\alpha/(w_j v_j)$ to the numerator, according to our Dirichlet prior specification in Section 3.1. This avoids 0's for probability estimates and increases the robustness of method.

4. EXPERIMENTS

4.1 Simulation study

We illustrate the capability of SBFC for prediction, feature selection and feature interaction detection by comparing with the following 8 most popular classification algorithms:

Lasso: R package `glmnet` [25],
SVM: Support Vector Machines, R package `e1071` [4],
NB: Naïve Bayes, R package `e1071` [3]
LR: logistic regression,
RF: Random Forest, R package `ranger` [2],
CART: Classification and Regression Trees, R package `tree` [6],
BART: Bayesian Additive Regression Trees, R package `BayesTree` [20],
C5.0: R package `C50` [7].

We designed two simulation studies: binary logistic regression and multinomial logistic regression. In both cases the data do not follow our posited model, but were simulated in the standard forward regression fashion.

4.1.1 Logistic regression for binary responses

We consider the following "noisy" logistic regression model:

$$\text{logit}\left(\mathbb{P}\left(Y = 1|X\right)\right) = X_1 - X_2 \cdot X_3 + X_4 \cdot X_5 - X_6 + \epsilon$$

where we add different levels of noise $\epsilon \sim N\left(0, \sigma^2\right)$ to test the robustness of the algorithms. The data generating process of features is as follows. For the variables in Group 1 (signal), we simulated X_1, X_2, X_4 independently from $N(0,1)$; X_5 from $N\left(X_4, 1\right)$; and simulated X_3 and X_6 nonlinearly as follows:

$$\begin{aligned}
X_3 &= \mathbb{1}\left(q_{0.3}^{(2)} < X_2 < q_{0.9}^{(2)}\right) + 2 \times \mathbb{1}\left(X_2 > q_{0.9}^{(2)}\right) \\
X_6 &= \mathbb{1}\left(q_{0.3}^{(5)} < X_5 < q_{0.7}^{(5)}\right) + 2 \times \mathbb{1}\left(q_{0.7}^{(5)} < X_5 < q_{0.9}^{(5)}\right) \\
&\quad + 3 \times \mathbb{1}\left(X_5 > q_{0.9}^{(5)}\right)
\end{aligned}$$

where $q_\alpha^{(j)}$ refers to the α quantile of variable X_j. For variables in Group 0 (noise predictors), all were simulated independently from $N(0,1)$. We compare SBFC with other algorithms in two settings: 100 noise features and 500 noise features. Note that the simulated examples contain both continuous and discrete features. In a pre-processing step, SBFC uses the Minimum Description Length Partitioning [26] to discretize continuous features. We made some signal features highly correlated with other ones so as to increase the difficulties of both prediction and feature selection.

For each experiment, we simulated 100 training cases and 1,000 test cases. We independently ran the experiments for 100 times to calculate the empirical mean and standard deviation of the prediction accuracy. Table 2 shows that SBFC

Table 2. Classification accuracy of SBFC and seven other algorithms in logistic regression model.

Low dimensional case: 100 independent noise predictors					
	$\sigma = 0$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 4$
SBFC	**0.728** $\pm$(0.023)	**0.710** $\pm$(0.032)	**0.667** $\pm$(0.035)	**0.632** $\pm$(0.037)	**0.600** $\pm$(0.038)
Lasso	0.632 $\pm$(0.040)	0.619 $\pm$(0.038)	0.587 $\pm$(0.033)	0.555 $\pm$(0.034)	0.543 $\pm$(0.029)
SVM	0.620 $\pm$(0.020)	0.607 $\pm$(0.027)	0.588 $\pm$(0.022)	0.562 $\pm$(0.028)	0.549 $\pm$(0.025)
Naive Bayes	0.644 $\pm$(0.020)	0.627 $\pm$(0.022)	0.595 $\pm$(0.021)	0.568 $\pm$(0.025)	0.553 $\pm$(0.021)
Logistic regression	0.510 $\pm$(0.026)	0.505 $\pm$(0.024)	0.509 $\pm$(0.019)	0.500 $\pm$(0.022)	0.507 $\pm$(0.018)
Random Forest	0.660 $\pm$(0.033)	0.641 $\pm$(0.035)	0.603 $\pm$(0.030)	0.573 $\pm$(0.030)	0.556 $\pm$(0.024)
CART	0.613 $\pm$(0.047)	0.599 $\pm$(0.040)	0.562 $\pm$(0.037)	0.536 $\pm$(0.033)	0.525 $\pm$(0.028)
BART	0.649 $\pm$(0.032)	0.631 $\pm$(0.032)	0.602 $\pm$(0.025)	0.576 $\pm$(0.029)	0.560 $\pm$(0.022)
C5.0	0.606 $\pm$(0.040)	0.593 $\pm$(0.034)	0.568 $\pm$(0.037)	0.542 $\pm$(0.031)	0.528 $\pm$(0.028)
High dimensional case: 500 independent noise predictors					
	$\sigma = 0$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 4$
SBFC	**0.724** $\pm$(0.026)	**0.702** $\pm$(0.027)	**0.663** $\pm$(0.033)	**0.628** $\pm$(0.029)	**0.594** $\pm$(0.036)
Lasso	0.616 $\pm$(0.033)	0.606 $\pm$(0.033)	0.570 $\pm$(0.032)	0.548 $\pm$(0.030)	0.536 $\pm$(0.027)
SVM	0.610 $\pm$(0.015)	0.602 $\pm$(0.016)	0.575 $\pm$(0.029)	0.558 $\pm$(0.030)	0.545 $\pm$(0.026)
Naive Bayes	0.605 $\pm$(0.020)	0.594 $\pm$(0.021)	0.565 $\pm$(0.022)	0.552 $\pm$(0.021)	0.535 $\pm$(0.021)
Logistic regression	0.505 $\pm$(0.023)	0.505 $\pm$(0.024)	0.502 $\pm$(0.021)	0.503 $\pm$(0.022)	0.500 $\pm$(0.018)
Random Forest	0.651 $\pm$(0.034)	0.628 $\pm$(0.032)	0.583 $\pm$(0.034)	0.561 $\pm$(0.029)	0.543 $\pm$(0.024)
CART	0.580 $\pm$(0.045)	0.569 $\pm$(0.047)	0.526 $\pm$(0.032)	0.523 $\pm$(0.032)	0.514 $\pm$(0.025)
BART	0.617 $\pm$(0.020)	0.607 $\pm$(0.018)	0.579 $\pm$(0.023)	0.561 $\pm$(0.025)	0.547 $\pm$(0.023)
C5.0	0.589 $\pm$(0.039)	0.566 $\pm$(0.040)	0.544 $\pm$(0.030)	0.527 $\pm$(0.026)	0.514 $\pm$(0.021)

significantly outperformed all other tested classification algorithms under different noise levels in both low and high dimensional cases, demonstrating that SBFC was robust to the number of noise features. Figure 3 shows the average graph of the discovered feature interaction (100 noise features and $\sigma = 0$). We observe that SBFC reliably identifies the signal features as well as relevant edges: $X_2 - X_3$, $X_4 - X_5$ and $X_5 - X_6$.

4.1.2 Multinomial logistic regression

We extended the above simulation study to the "noisy" multinomial logistic regression setting as follows:

$$\begin{aligned}
\log\left(\mathbb{P}\left(Y=1|X\right)\right) &\propto \beta_{11}X_1 - \beta_{12}X_2\cdot X_3 + \beta_{13}X_4\cdot X_5 \\
&\quad - \beta_{14}X_6 \\
\log\left(\mathbb{P}\left(Y=2|X\right)\right) &\propto \beta_{21}X_1 - \beta_{22}X_2\cdot X_3 + \beta_{23}X_4\cdot X_5 \\
&\quad - \beta_{24}X_6 + \epsilon_1 \\
\log\left(\mathbb{P}\left(Y=3|X\right)\right) &\propto \beta_{31}X_1 - \beta_{32}X_2\cdot X_3 + \beta_{33}X_4\cdot X_5 \\
&\quad - \beta_{34}X_6 + \epsilon_2
\end{aligned}$$

where all the coefficients $\beta_{ij}, i = 1,2,3,\ j = 1,2,3,4$ were independently sampled from $N(0,1)$. Similarly, ϵ_1 and ϵ_2 were independently sampled from $N\left(0,\sigma^2\right)$ and features in both signal group and noise group were simulated exactly the same as in the previous simulation study.

For each experiment, we simulated 200 training cases and 1,000 test cases. We independently repeated the experiment for 200 times to stabilize the empirical prediction accuracy. In this simulation study, lasso refers to the multinomial lasso and we did not run logistic regression and BART since they cannot handle multi-class situations. Table 3 shows that SBFC still significantly outperformed all other tested algorithms. We note that multinomial lasso performed very unstably in this multi-class classification problem, while the stabilities of other methods are comparable. Figure 4 shows that SBFC correctly identified the feature interactions in the signal group with a small number of noise features being mistakenly selected.

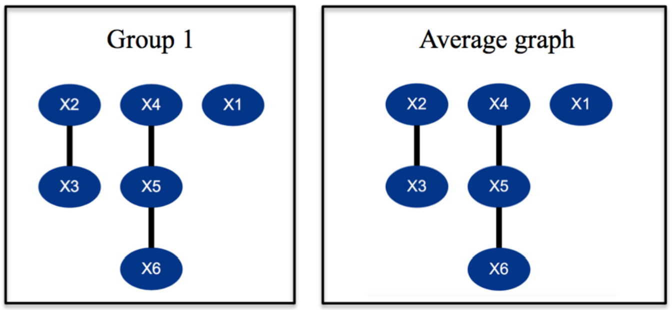


Figure 3. Left: True feature interaction in the signal group; Right: An average graph from SBFC with 100 noise features. We set $\sigma = 0$ and used 500 training cases to stabilize the selection results.

4.2 Real data sets

4.2.1 SPECTF heart data set

The SPECTF heart data set from UCI repository [27] contains the Single Proton Emission Computed Tomogra-

Table 3. Classification accuracy of SBFC and six other algorithms for multinomial logistic regression model.

Low dimensional case: 100 independent noise predictors					
	$\sigma = 0$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 4$
SBFC	**0.682** $\pm$(0.035)	**0.654** $\pm$(0.036)	**0.603** $\pm$(0.032)	**0.562** $\pm$(0.035)	**0.519** $\pm$(0.030)
Lasso	0.496 $\pm$(0.107)	0.470 $\pm$(0.100)	0.439 $\pm$(0.090)	0.394 $\pm$(0.082)	0.362 $\pm$(0.070)
SVM	0.578 $\pm$(0.035)	0.564 $\pm$(0.040)	0.527 $\pm$(0.037)	0.515 $\pm$(0.042)	0.479 $\pm$(0.032)
Naive Bayes	0.597 $\pm$(0.042)	0.573 $\pm$(0.042)	0.532 $\pm$(0.034)	0.500 $\pm$(0.032)	0.463 $\pm$(0.025)
Random Forest	0.633$\pm$(0.031)	0.606 $\pm$(0.032)	0.560 $\pm$(0.028)	0.535 $\pm$(0.027)	0.492$\pm$(0.030)
CART	0.593 $\pm$(0.045)	0.560 $\pm$(0.047)	0.503 $\pm$(0.037)	0.474 $\pm$(0.039)	0.429 $\pm$(0.026)
C5.0	0.561 $\pm$(0.042)	0.538 $\pm$(0.039)	0.487 $\pm$(0.028)	0.457 $\pm$(0.028)	0.422 $\pm$(0.022)
High dimensional case: 500 independent noise predictors					
	$\sigma = 0$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 4$
SBFC	**0.679** $\pm$(0.022)	**0.637** $\pm$(0.027)	**0.588** $\pm$(0.022)	**0.545** $\pm$(0.018)	**0.518** $\pm$(0.028)
Lasso	0.496 $\pm$(0.095)	0.483 $\pm$(0.091)	0.440 $\pm$(0.089)	0.385 $\pm$(0.076)	0.367 $\pm$(0.068)
SVM	0.550 $\pm$(0.044)	0.529 $\pm$(0.045)	0.526 $\pm$(0.042)	0.483$\pm$(0.040)	0.464$\pm$(0.033)
Naive Bayes	0.560 $\pm$(0.038)	0.541 $\pm$(0.036)	0.525 $\pm$(0.037)	0.479 $\pm$(0.038)	0.446 $\pm$(0.027)
Random Forest	0.559 $\pm$(0.022)	0.545 $\pm$(0.019)	0.533 $\pm$(0.023)	0.492 $\pm$(0.024)	0.446 $\pm$(0.018)
CART	0.585 $\pm$(0.036)	0.568 $\pm$(0.037)	0.547 $\pm$(0.030)	0.502 $\pm$(0.038)	0.470 $\pm$(0.030)
C5.0	0.534 $\pm$(0.038)	0.512 $\pm$(0.033)	0.478 $\pm$(0.034)	0.433 $\pm$(0.026)	0.405 $\pm$(0.021)

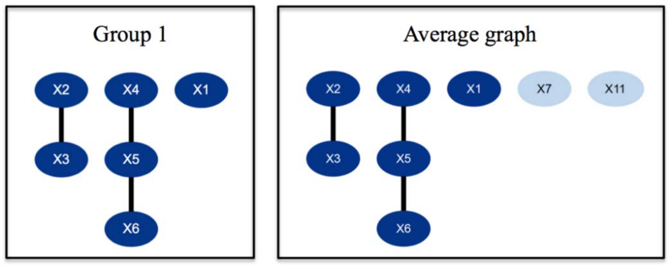


Figure 4. Left: True feature interaction in the signal group; Right: An average graph from SBFC with 100 noise features. We set $\sigma = 0$ and use 500 training cases to stabilize the selection results. We omitted some similar variables as X_7 and X_{11} with low proportions in the signal group.

phy (SPECT) image of 267 patients. For each patient, there are 44 continuous features extracted from the SPECT image and the associated binary class label is the overall myocardial perfusion diagnosis: normal and abnormal. All numerical features have integer values from 0 to 100. The SPECTF data set is originally divided into 80 instances of training data and 187 instances of testing data. We compared the prediction performances of SBFC and other classification algorithms in two ways: (1) We train all the algorithms on a common training data set and compare their prediction accuracies on the common test data; (2) We combine the training data and the test data together and run a 5-fold cross validation. Table 4 shows that SBFC outperformed other methods in terms of classification accuracy. The 78.61% classification accuracy of SBFC is also better than the CLIP3 algorithm [28], which is a standard approach to generate classification rules from the SPECT images and achieved 77.0% classification accuracy compared with cardiologists' final diagnoses.

4.2.2 corral data set

The corral data set [29] contains six binary features $\{X_1, X_2, X_3, X_4, X_5, X_6\}$, which generates the associated class label as $Y = (X_1 \wedge X_2) \vee (X_3 \wedge X_4)$. X_5 is a noise feature in Group 0 and X_6 is a signal feature in Group 1, which is correlated with the class label with 25% error. Thus in the corral data set, the relevant features are $\{X_1, X_2, X_3, X_4, X_6\}$, and the most relevant edges are $\{X_1, X_2\}$, $\{X_3, X_4\}$, while other edges between the first four features are less relevant, and any edge with X_5 or X_6 is incorrect. Figure 5 shows a sampled graph and the average graph of SBFC, illustrating that SBFC successfully recovered the true correlation structure between the features, with the most relevant edges appearing most frequently (as indicated by thickness). Table 4 compares the classification accuracy (5-fold CV) of SBFC and other standard algorithms. We see that SBFC as well as SVM have the best prediction performance for this data set.

4.2.3 madelon data set

The madelon data set has been used in the 2003 NIPS feature selection challenge. Feature selection on this data set is considered to be challenging since (1) the data set is high dimensional with 20 relevant features and 480 noise features; and (2) no feature is informative by itself and all the relevant features are correlated with each other [30]. Figure 6 shows that SBFC reliably selected the correct set of 20 relevant features [31] for this high-dimensional dataset, while other methods such as Lasso, BART and Random Forest failed this task. Regardless of the selection proportion or significance, Lasso, BART and Random Forest missed

Table 4. Classification accuracy for the SPECTF heart data set, corral data set, and madelon data set

Data	Method	SBFC	Lasso	SVM	NB	LR	RF	CART	BART	C5.0
SPECTF heart	train/test	**0.786**	0.657	0.743	0.679	0.556	0.743	0.775	0.737	0.679
	5-fold CV	**0.846**	0.809	0.793	0.738	0.763	0.824	0.793	0.812	0.801
corral	5-fold CV	**1.00**	0.867	**1.00**	0.869	0.862	0.966	0.812	0.919	0.990
madelon	5-fold CV	0.634	0.607	0.620	0.598	0.600	0.671	**0.782**	0.760	0.758

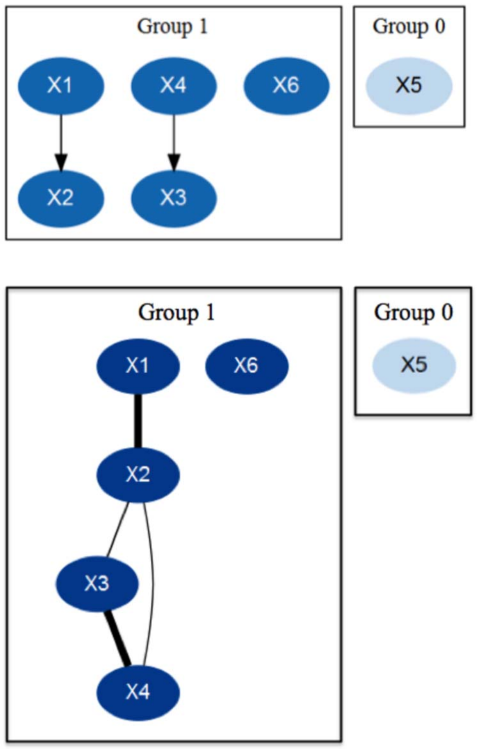


Figure 5. Up: A sampled graph for the corral data set; Down: Average graph for the corral data set.

15 features, 6 features, and 1 feature, respectively. We also note, however, that Table 4 shows that SBFC's classification accuracy was right in the middle among all the tested methods, and was at the lower end of all tree-based algorithms (e.g., Random Forest, CART, BART and C5.0). There are two possible reasons for this: (1) SBFC is designed for types of data whose features can be arranged as a set of non-overlapping trees conditional on Y. In contrast, other tree-based algorithms allow one variable to appear in multiple nodes of the single and multiple trees. For the madelon data set, since all the features are correlated with each other, it might not be sufficient to model their joint distribution as non-overlapping trees by SBFC. (2) SBFC requires pre-discretization of the data so that all the features become categorical. For computational efficiency, we use binary binning [26] for high dimensional data sets. Such a discretization procedure might lead to an information loss.

5. THE R PACKAGE

The R package is available on CRAN and at github.org/vkrakovna/sbfc. The command for running the algorithm is `sbfc(data)`. It expects a discretized data set as input, which can be produced using the `data_disc()` command.

5.1 Graph visualizations

The `sbfc_graph()` command creates visualizations of the relevant features and feature interactions identified by the MCMC algorithm, such as those in Figures 3, 4 and 5. The command `sbfc_graph(average=FALSE, iter=N)` shows the sampled graph for the N-th MCMC iteration. The Group 1 nodes are dark-shaded, and the Group 0 nodes are light-shaded.

The command `sbfc_graph(average=TRUE)` shows an average graph, aggregating information from all MCMC samples. The nodes are color-coded according to relevance - the proportion of samples where the corresponding feature appeared in Group 1 (dark-shaded nodes appear more often). Edge thickness also corresponds to relevance - the proportion of samples where the corresponding feature interaction appeared. To avoid clutter, only edges that appear in at least a certain proportion of the sampled graphs are shown, specified by the `edge_cutoff` option, which defaults to 0.1. To see more of the low-relevance edges, lower the `edge_cutoff` value.

If you have a data set with meaningful variable labels, you can add these to your graph by setting the `labels` option to a set of labels of your choice. To determine which groups of features influence the predictions the most, you can visually identify clusters of thicker edges in the graph. See Figure 5 for an example.

5.2 Example: heart disease data set

The heart disease data set from UCI repository [27] contains 13 variables related to heart disease, such as age, heart rate and blood pressure, and 270 observations. We are interested in identifying which features and feature interactions contribute to the presence of heart disease.

We begin by removing rows with missing values and discretizing the data set using Minimum Description Length partitioning. We then run SBFC on the discretized data set. Since the data set is a bit too small to divide into a training and test set, we leave the `n_train` argument for the number of training rows unspecified.

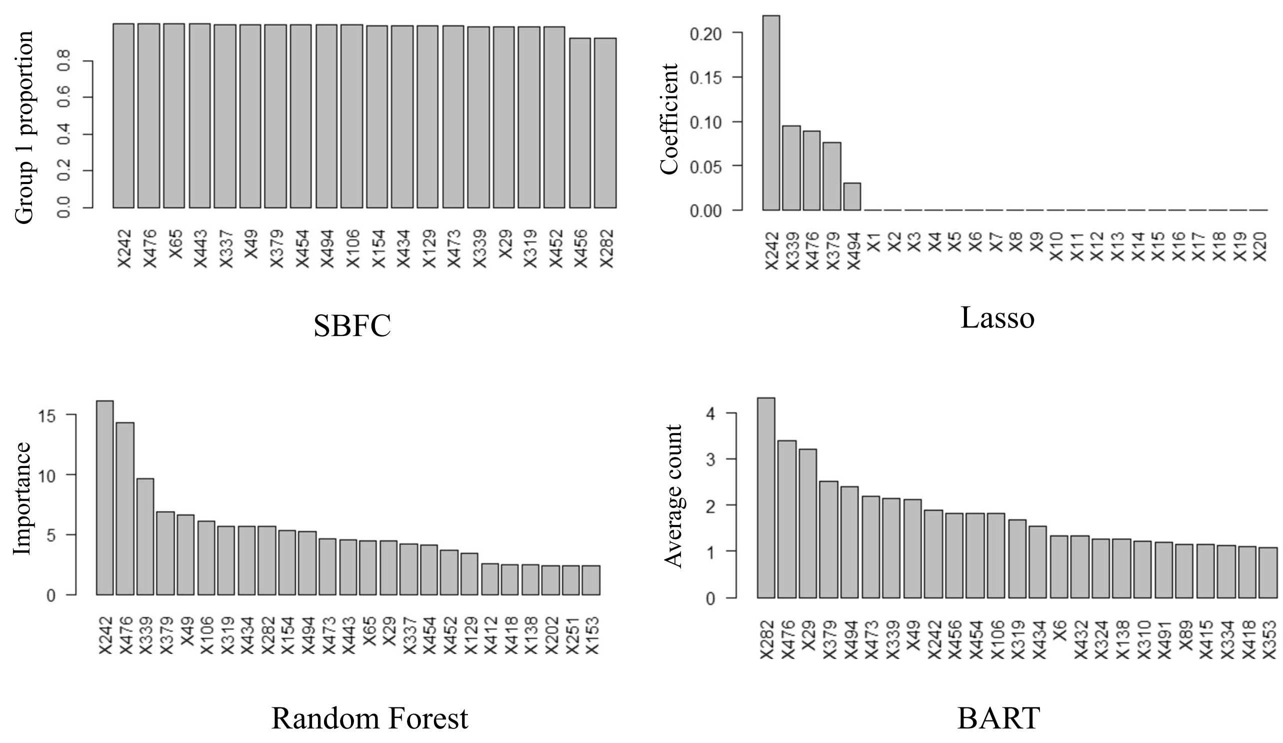


Figure 6. Feature selection results for the madelon data set. SBFC selects the correct set of 20 relevant features [31]. Other methods such as Lasso, BART and Random Forest missed 15 features, 6 features and 1 feature respectively.

```
heart = data_disc(heart_data, n_train=NULL)
heart_result = sbfc(heart)
```

By default `sbfc()` runs 5-fold cross-validation if a test set is not supplied, and `sbfc_graph()` uses the MCMC samples from the first cross-validation fold. If you are not interested in classification and just want to get MCMC samples and graphs, you can run `sbfc(heart, cv=FALSE)`. Since this data set has variables with meaningful names, we supply these as node labels for the graph.

```
heart_labels = c("Age", "Sex", "Chest Pain",
    "Rest Blood Pressure", "Cholesterol",
    "Blood Sugar", "Rest ECG", "Max Heart Rate",
    "Angina", "ST Depression", "ST Slope",
    "Fluoroscopy Colored Vessels", "Thalassemia")
sbfc_graph(heart_result, labels=heart_labels,
    width=700)
```

In order to identify those important features, run `signal_var_proportion(heart_result, nvars)`. Figure 7 shows the *signal proportion*, which is defined as the proportion of MCMC samples of a specific variable in the signal group (Group 1). The command returns the top **nvars** features in decreasing order of signal proportion. For comparison, Table 5 shows the feature rankings by Random Forest, Lasso and BART. To perform MCMC diagnostics, plot the log posterior as shown in Figure 8 using `logposterior_plot(heart_result)`.

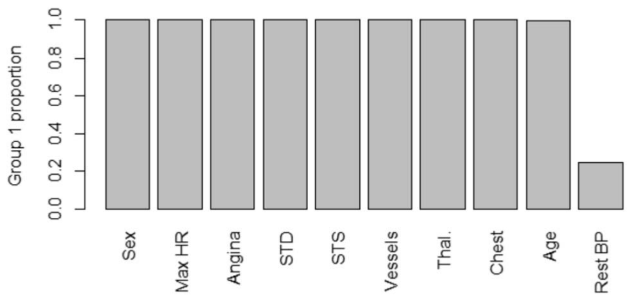


Figure 7. Top 10 selected signal features by SBFC on the heart disease data set.

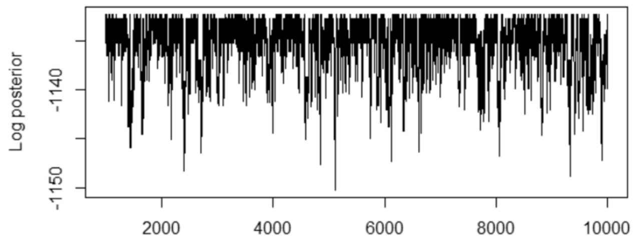


Figure 8. Log posterior diagnostic plots for the heart disease data set. Total run is 10,000 iterations, with burn-in for the first 2,000 iterations.

Table 5. Feature ranking for heart disease data set by Random Forest, Lasso, and BART

Method	1	2	3	4	5	6	7	8	9	10	11	12	13
RF	Chest	Thal.	Vessels	Angina	Max HR	STD	STS	Sex	Age	Rest BP	Chol.	Sugar	ECG
Lasso	Vessels	Chest	Thal.	STD	STS	Angina	Sex	Max HR	Age	Rest BP	Chol.	Sugar	ECG
BART	Chest	Thal.	Vessels	Max HR	Age	Sex	STS	STD	Angina	Rest BP	Chol.	Sugar	ECG

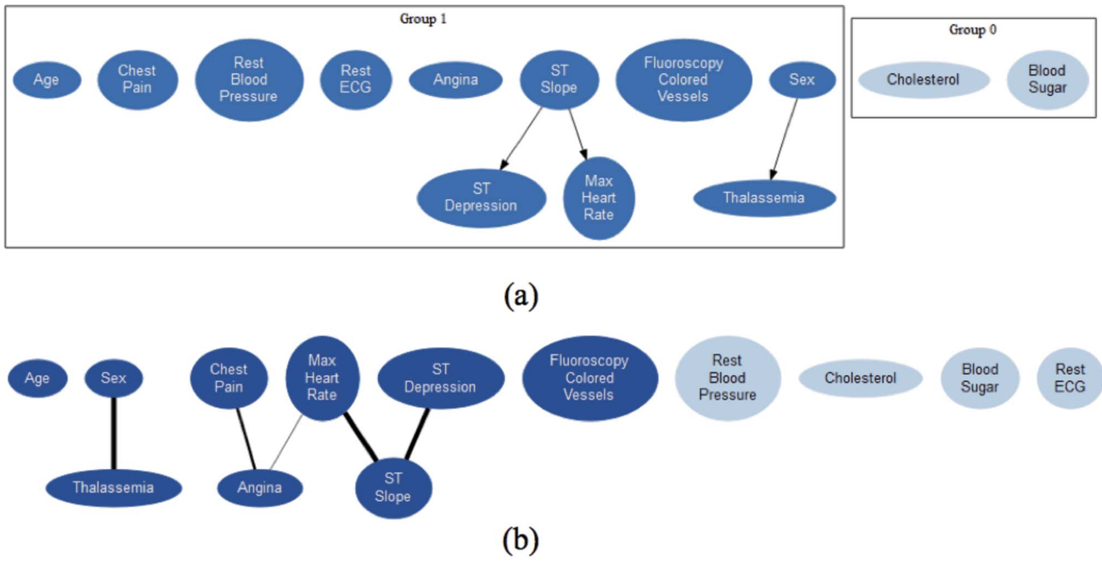


Figure 9. (a) A sampled graph for the heart disease data set by SBFC; (b) Average graph for the heart disease data set by SBFC.

Figure 9 shows the results of feature selection and feature interaction detection of SBFC on the heart disease data set. The dark-shaded features in the average graph are the most relevant for predicting heart disease. There are several groups of relevant interacting features: (Sex, Thalassemia), (Chest Pain, angina), and (Max Heart Rate, ST Slope, ST Depression). The features in each group jointly affect the presence of heart disease. Figure 9(c) compares feature rankings with other methods, showing that all the methods agree on the top 9 features, which SBFC ranked almost equally (all with posterior probability very close to 1).

6. CONCLUSION

SBFC is an integrated tool for supervised classification, feature selection, interaction detection, and visualization. It splits the features into signal and noise groups according to their relationship with the class label, and uses tree structures to model dependencies among both signal and noise features. The tree dependence structure gives SBFC adequate modeling flexibility, excellent feature and interaction selection performances, and competitive classification accuracies even when the signal to noise ratio decreases and the number of irrelevant features increases. Since our simulations showed that SBFC performed well with well-behaved features, in practice it might be helpful to first "normalize" the features (e.g., capping outliers, or conducting quantile normalizations) before feeding them into SBFC.

ACKNOWLEDGMENT

We thank Dr. Jiong Du for his efforts in obtaining preliminary results at the initial stage of this project and Drs. Yuan Yuan and Bo Jiang for helpful discussions. This research was partially supported by the NSF grant DMS-1613035 and the NIH grant R01GM122080.

Received 9 November 2017

REFERENCES

[1] R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996. MR1379242
[2] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.
[3] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. New York: John Wiley and Sons, 1973. MR1802993
[4] T. Evgeniou, M. Pontil, and T. Poggio. Regularization Networks and Support Vector Machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000. MR1759187
[5] L. Meier, S. van de Geer, and P. Buhlmann. The Group Lasso for Logistic Regression. *Journal of the Royal Statistical Society, Series B*, 70(1):53–71, 2008. MR2412631

[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *CART: Classification and Regression Trees*. Chapman and Hall, 1984. MR0726392

[7] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[8] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. MR0965765

[9] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20:197–243, 1995. MR1615024

[10] G. F. Cooper. The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks. *Artificial Intelligence*, 42(2–3):393–405, March 1990. MR1045483

[11] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29:131–163, 1997.

[12] G. I. Webb, J. Boughton, and Z. Wang. Not So Naive Bayes: Aggregating One-Dependence Estimators. *Machine Learning*, 58:5–24, 2005.

[13] H. Zhang, L. Jiang, and J. Su. Augmenting Naive Bayes for Ranking. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 1020–1027. ACM, 2005.

[14] C. K. Chow and C. N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory*, 14(11):462–467, 1968.

[15] P. Langley and S. Sage. Induction of Selective Bayesian Classifiers. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 399–406. Morgan Kaufmann, 1994.

[16] L. Jiang, H. Zhang, Z. Cai, and J. Su. Evolutional Naive Bayes. *Proceedings of the 2005 International Symposium on Intelligent Computation and its Applications, ISICA*, pages 344–350, 2005.

[17] R. Kohavi and G. H. John. Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97:273–324, 1997.

[18] J. L. Andrews and P. D. McNicholas. Variable Selection for Clustering and Classification. *Journal of Classification*, 31(2):136–153, 2014. MR3232592

[19] S. Tang, L. Chen, K. Tsui, and K. Doksum. Nonparametric Variable Selection and Classification: The CATCH Algorithm. *Computational Statistics and Data Analysis*, 72:158–175, 2014. MR3139354

[20] H. A. Chipman, E. I. George, and R. E. McCulloch. BART: Bayesian Additive Regression Trees. *Annals of Applied Statistics*, 4(1):266–298, 2010. MR2758172

[21] A. Andrei and C. Kendziorski. An efficient method for identifying statistical interactors in gene association networks. *Biostatistics*, 10(4):706–718, 2009.

[22] Y. Zhang and J. S. Liu. Bayesian Inference of Epistatic Interactions in Case-Control Studies. *Nature Genetics*, 39(9):1167–1173, 2007.

[23] Cassio P. de Campos, Marco Cuccu, Giorgio Corani, and Marco Zaffalon. *Extended Tree Augmented Naive Classifier*, pages 176–189. Springer International Publishing, 2014.

[24] J. Hoeting, D. M. Adrian, and C. T. Volinsky. Bayesian Model Averaging. In *Proceedings of the AAAI Workshop on Integrating Multiple Learned Models*, pages 77–83. AAAI Press, 1998. MR1820767

[25] J. Friedman, T. Hastie, and R. Tibshirani. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

[26] U. M. Fayyad and K. B. Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 1022–1027. Morgan Kaufmann Publishers, San Francisco, CA, 1993.

[27] M. Lichman. UCI Machine Learning Repository, 2013.

[28] Krzysztof J. Cios, Daniel K. Wedding, and Ning Liu. Clip3: Cover learning using integer programming. *Kybernetes*, 26(5):513–536, 1997.

[29] George H. John, Ron Kohavi, Karl Pfleger, et al. Irrelevant features and the subset selection problem. In *Machine learning: proceedings of the eleventh international conference*, pages 121–129, 1994.

[30] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror. Result Analysis of the NIPS 2003 Feature Selection Challenge. In *Advances in Neural Information Processing Systems 17*, pages 545–552. MIT Press, 2005.

[31] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing). Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.*

Viktoriya Krakovna
DeepMind
London
United Kingdom
E-mail address: vkrakovna@gmail.com

Chenguang Dai
Department of Statistics
Harvard University
1 Oxford St
Cambridge, MA
USA
E-mail address: chenguangdai@g.harvard.edu

Jun S. Liu
Department of Statistics
Harvard University
1 Oxford St
Cambridge, MA
USA
E-mail address: jliu@stat.harvard.edu