AN $L_1$-REGULARIZED NAÏVE BAYES-INSPIRED CLASSIFIER FOR DISCARDING REDUNDANT AND IRRELEVANT PREDICTORS

DIEGO VIDAURRE
Computational Intelligence Group, Universidad Politécnica de Madrid, Spain
diego.vidaurre@fi.upm.es

CONCHA BIELZA
Computational Intelligence Group, Universidad Politécnica de Madrid, Spain
mcbielza@fi.upm.es

PEDRO LARRANAGA
Computational Intelligence Group, Universidad Politécnica de Madrid, Spain
pedro.larranaga@fi.upm.es

Received 23 July 2012
Accepted 25 March 2013
Published 19 August 2013

The naïve Bayes model is a simple but often satisfactory supervised classification method. The original naïve Bayes scheme, does, however, have a serious weakness, namely, the harmful effect of redundant predictors. In this paper, we study how to apply a regularization technique to learn a computationally efficient classifier that is inspired by naïve Bayes. The proposed formulation, combined with an $L_1$-penalty, is capable of discarding harmful, redundant predictors. A modification of the LARS algorithm is devised to solve this problem. We tackle both real-valued and discrete predictors, assuring that our method is applicable to a wide range of data. In the experimental section, we empirically study the effect of redundant and irrelevant predictors. We also test the method on a high-dimensional data set from the neuroscience field, where there are many more predictors than data cases. Finally, we run the method on a real data set that combines categorical with numeric predictors. Our approach is compared with several naïve Bayes variants and other classification algorithms (SVM and kNN), and is shown to be competitive.

Keywords: Lasso; regularization; naïve Bayes; redundancy.

1. Introduction

Bayesian network classifiers are often used for classification problems. The model parameters are usually found by maximizing the joint likelihood. The naïve Bayes model is a simple Bayesian network classifier that assumes the predictors are independent given each class value. In spite of this strong assumption, this classifier has been proven to work satisfactorily in many domains.\textsuperscript{2,3}
Some training schemes have been proposed on top of the naive Bayes idea. For example, the \textit{weighted naive Bayes} \cite{4} assigns a weight to each predictor so that some predictors have more influence than others. Unfortunately, the naive Bayes model (including weighted naive Bayes) always includes all predictors in the model and behaves poorly in the presence of redundant predictors. This issue is discussed by Langley and Sage, \cite{5} who proposed the \textit{selective naive Bayes} classifier. This classifier greedily includes predictors in a search-based algorithm. However, this is a heuristic method and is not guaranteed to find an optimal model. Without a prefiltering step, \cite{6,7} the selective naive Bayes is seldom applicable for high-dimensional settings on computational grounds. On the other hand, the so-called \textit{semi-naive Bayes} \cite{8} performs a heuristic greedy search to select predictors and find dependences between them, fusing these predictors to a single predictor. The same computational issue applies here.

Regularization techniques have occasionally been used to improve naive Bayes. \cite{9} The criterion used to fit data to a model is the data likelihood plus a penalization term. This is derived from a Bayesian approach with a prior distribution that assigns higher probabilities to networks with fewer predictors. This is embedded in a greedy search heuristic that iteratively selects predictors for inclusion in (or exclusion from) the model.

The \textit{lasso} \cite{10} is a popular regularization technique that imposes an $L_1$-penalty on the usual least-squares linear regression, with the aim of reducing the variance of the estimates, preventing overfitting, performing simultaneously variable selection and, finally, improving the model interpretability. Depending on the chosen regularization parameter, some regression coefficients are set to exactly zero, and the corresponding predictors are discarded. The lasso has a solid theoretical ground-work. \cite{11} The $L_1$-penalty has been widely used in many classification paradigms, like logistic regression. \cite{12}

With a minor modification, the \textit{LARS} algorithm \cite{13} assesses the complete lasso regularization path, that is, the whole set of regression coefficient estimates with regard to the regularization parameter. LARS is of particular interest because it solves the complete regularization path at the cost of an ordinary least squares fit. Besides least squares functions, the LARS algorithm can be used to efficiently minimize other loss functions subject to an $L_1$-penalty provided these loss functions meet certain conditions. \cite{14}

In this paper, we introduce a supervised classification method that is inspired on naive Bayes and based on convex optimization. On the one hand, this formulation allows to apply regularization techniques from linear regression that permit to discard both redundant and irrelevant predictors. Redundant predictors are known to be harmful for naive Bayes and variants, and also for our model. On the other hand, like naive Bayes, it can directly deal with both continuous and discrete predictors and can be directly used in multi-class problems. Thus, our method is applicable to a wide range of data sets.
The proposed method establishes a linear combination of the likelihood contributions of each predictor. This linear combination is chosen so that the result is maximized, assuming that the coefficients are somehow constrained. This will give priority to those variables whose likelihood contributions are higher. The applied constraint is an $L_1$-penalty, which yields a sparse vector of coefficients, dropping the likelihood contribution of some predictors and, thus, enhancing the interpretability of the model. As we will show, this method can discard both redundant and irrelevant predictors (i.e. their respective likelihood contributions).

The devised loss function also meets the requirements for applying a LARS type algorithm. This algorithm would efficiently compute the entire regularization path at one shot. This is beneficial in high dimensional settings on computational grounds. Finally, our method is applicable to a wide range of data.

The rest of the paper is organized as follows. Section 2 presents the terminology and some related methods. Section 3 introduces the proposed scheme in detail. Section 4 discusses the reasons why our method discards both redundant and irrelevant predictors. Section 5 presents an efficient LARS-based algorithm to solve the problem formulated in Section 3. Section 6 details the set of experiments used to test the algorithm. Section 7 discusses conclusions and future work.

2. Basics

2.1. Terminology

Let $\{X_1, \ldots, X_p\}$ be the set of $p$ predictors and $Y$ the class variable. Let $D = \{(x_{r1}, \ldots, x_{rp}, y_r), r = 1, \ldots, n\}$ be the labeled data set containing $n$ instances. We denote the $n \times p$ predictor data matrix as $X$ and the vector of responses as $y = (y_1, \ldots, y_n)$. We assume that the class variable, $Y$, may take values $j \in \{1, \ldots, c\}$.

The objective is to learn a classifier from $D$ so as to predict the class value for incoming data points.

Without any loss of generality, we will assume that predictors indexed by $\Upsilon = \{1, \ldots, q\}$ are discrete and predictors indexed by $\Gamma = \{q + 1, \ldots, p\}$ are continuous.

Each discrete predictor $X_i, i \in \Upsilon$, has $m_i$ possible states. Considering a naive Bayes model where the predictors are conditionally independent given the class, we denote their conditional probability table (CPT) as an $m_i \times c$ matrix $\Theta_i$. Each element $\theta_{ikj}$ of $\Theta_i, j \in \{1, \ldots, c\}, k \in \{1, \ldots, m_i\}$, is the probability of the predictor $X_i$ taking its $k$-th state given the $j$-th class variable state, i.e. $P(X_i = k|Y = j; \Theta_i)$.

We assume that continuous predictors $X_i, i \in \Gamma$, follow a Gaussian distribution within each class value. We denote as $\mu_i$ and $\sigma_i$ the vectors whose elements are, for each state of $Y$, the expectation and standard deviation of $X_i$, respectively, i.e. $X_i|Y = j \sim N(\mu_{ij}, \sigma_{ij}^2), j \in \{1, \ldots, c\}$. We denote the conditional density function for predictor $X_i$ given that $Y = j$ as $f(x_i|j; \mu_{ij}, \sigma_{ij}^2)$.

Let $\Omega = \{\Theta_1, \ldots, \Theta_q, \mu_{q+1}, \sigma_{q+1}^2, \ldots, \mu_p, \sigma_p^2\}$ be the whole set of parameters. Considering the predictors to be conditional independent given the class, the full
selective naïve Bayes and the weighted naïve Bayes. Since they will be useful for the sake of comparison, we define the above-mentioned 2.2. Naïve Bayes and variants

Since they will be useful for the sake of comparison, we define the above-mentioned selective naïve Bayes and the weighted naïve Bayes.
An $L_1$-Regularized Naïve Bayes-Inspired Classifier

Let $\hat{\Theta}_i, i \in \Upsilon$, be the maximum likelihood (ML) estimation of parameters for a discrete predictor. Let $\hat{\mu}_i$ and $\hat{\sigma}^2_i, i \in \Gamma$, be the ML estimation of parameters for a continuous predictor:

$$
\hat{\theta}_{ikj} = \frac{\#_D(X_i = k, Y = j)}{\#_D(Y = j)},
$$

$$
\hat{\mu}_{ij} = \frac{\sum_{r:y_r=j}x_r}{\#_D(Y = j)},
$$

$$
\hat{\sigma}^2_{ij} = \frac{\sum_{r:y_r=j}(x_r - \hat{\mu}_{ij})^2}{\#_D(Y = j)},
$$

where $\#_D()$ is a count function over the data set $D$.

Now, a pure naïve Bayes formulation for the probability of the class given the predictors is

$$
P(Y = j | X_1 = k_1, \ldots, X_q = k_q, X_{q+1}, \ldots, X_p = x_p, \hat{\Theta}) \propto P(Y = j) \prod_{i=1}^q P(X_i = k_i | Y = j, \hat{\Theta}_i) \prod_{i=q+1}^p f(x_i | j; \hat{\mu}_{ij}, \hat{\sigma}^2_{ij}),
$$

where probabilities and density functions are computed over parameters $\hat{\Theta}_i$, $\hat{\mu}_i$ and $\hat{\sigma}^2_i$. For an instance whose class value is to be predicted, the value $j \in \{1, \ldots, c\}$ that maximizes (7) will be chosen.

The selective naïve Bayes obeys Equation (7) but it is applied only over a subset of predictors. This subset of predictors can be found in a forward greedy search, so that predictors are included in the model as long as the prediction accuracy for a validation data set is increasing.

Instead, the weighted naive Bayes model includes all the predictors. In the paper by Ferreira et al., for example, the model is only defined for discrete predictors, devising a procedure for continuous predictor discretization. Predictors are weighted according to their relevance, computed by

$$
w_i = \sqrt{\sum_{j=1}^c \sum_{k=1}^{m_i} \left[ P(Y = j | X_i = k) - P(Y = j) \right]^2},
$$

so that the model is

$$
P(Y = j | X_1 = k_1, \ldots, X_q = k_q, \hat{\Theta}) \propto P(Y = j) \prod_{i=1}^q P(X_i = k_i | Y = j, \hat{\Theta}_i)^{w_i}.
$$

3. The Method

In this paper, we separately focus on each predictor to build a penalized linear expression whose minimization will yield a classifier that discards irrelevant and redundant predictors.
We first obtain the ML parameters \( \hat{\Theta}_i \), for \( i \in \mathcal{Y} \), and \( \hat{\mu}_i \) and \( \hat{\sigma}_{i}^2 \), for \( i \in \Gamma \), from Equation (6). Let \( \Omega_i \) be either \( \Theta_i \) or \( \{ \hat{\mu}_i, \hat{\sigma}_{i}^2 \} \). Now, we establish the linear expression:

\[
\sum_{r=1}^{n} \sum_{i=1}^{p} \beta_i P(Y = y_r | X_i = x_{ri}, \Theta_i) \quad \text{s.t.} \quad \sum_{i=1}^{p} \beta_i = 1, \quad 0 \leq \beta_i \leq 1, \forall i,
\]

(10)

where, following the Bayes’ rule and using Equations (2) and (3), we obtain for discrete and continuous predictors, respectively,

\[
P(Y = y_r | X_i = x_{ri}, \Theta_i) = \frac{P(X_i = x_{ri} | Y = y_r, \Theta_i) P(Y = y_r)}{\sum_{j=1}^{c} P(X_i = x_{ri} | Y = j, \Theta_i) P(Y = j)}
\]

(11)

and

\[
P(Y = y_r | X_i = x_{ri}, \hat{\mu}_i, \hat{\sigma}_i^2) = \frac{f(x_{ri} | y_r; \hat{\mu}_i, \hat{\sigma}_i^2) P(Y = y_r)}{\sum_{j=1}^{c} f(x_{ri} | j; \hat{\mu}_i, \hat{\sigma}_i^2) P(Y = j)}.
\]

(12)

Vector \( \hat{\beta} = (\hat{\beta}_1, \ldots, \hat{\beta}_p) \) would be chosen to maximize (10), hence giving more weight to predictors that are more relevant for the classification. The rationale of this approach is that relevant predictors will have values \( P(Y = y_r | X_i = x_{ri}, \Theta_i) \) closer to one than irrelevant predictors. Hence, when maximizing (10) across the data set, the coefficients \( \beta_i \) of the relevant predictors are promoted to be higher. Note also that, as long as \( \sum_{i=1}^{p} \beta_i = 1 \), expression

\[
\sum_{i=1}^{p} \beta_i P(Y = y_r | X_i = x_{ri}, \hat{\Theta}_i)
\]

(13)

ranges from 0 to 1, like a probability, We can use this as a basis for classifying future instances. Specifically, given \( \hat{\beta} \) and \( \hat{\Theta}_i \), we would select, for a new instance given by \( x_i \), the class value \( j \in \{1, \ldots, c\} \) that maximizes

\[
\sum_{i=1}^{p} \hat{\beta}_i P(Y = j | X_i = x_i, \hat{\Theta}_i).
\]

(14)

Note that \( \beta_i = 0 \) implies that predictor \( X_i \) is not selected. Likewise, higher values of \( \beta_i \) would attach more importance to predictor \( X_i \). Predictors that are considered to be relevant (i.e., with a high \( \beta_i \)) are expected to have a higher probability \( P(Y = j | X_i = x_i, \hat{\Theta}_i) \) for the true class, as it was in the training data set.

To obtain \( \hat{\beta} \), we could devise a linear optimization problem that maximizes (10) for the data set. However, it will not drive any \( \beta_i \) to exactly zero, and, hence, will not perform variable selection. We alternatively propose an \( L_1 \)-constrained problem to estimate \( \hat{\beta} \):

\[
\text{min}_{\beta} \sum_{r=1}^{n} \left( 1 - \sum_{i=1}^{p} \beta_i P(Y = y_r | X_i = x_{ri}, \hat{\Theta}_i) \right)^2 \quad \text{s.t.} \quad 0 \leq \beta_i \leq 1, \forall i,
\]

\[
\sum_{i=1}^{p} \beta_i \leq s
\]

(15)
Algorithm 1 \( L_1 \)-NB

**Input:** Data set \( D \) with \( p \) predictors and \( n \) labeled cases

**Input:** A set of unlabeled cases

**Output:** A vector of coefficients \( \hat{\beta} = (\beta_1, \ldots, \beta_p) \)

**Output:** The predicted classes for the unlabeled cases

Obtain ML parameters \( \hat{\Omega}_i, i = 1, \ldots, p \), from Equation (6)

Obtain matrix \( B, B_{ri} = P(Y = y_r|X_i = x_{ri}, \hat{\Omega}_i), i = 1, \ldots, p, r = 1, \ldots, n \)

Obtain solutions \( \hat{\beta}^{(l)}, l = 1, \ldots, L \), with LARS from \( B \)

\( \hat{\beta} := \arg\min_{\beta} \text{AIC}(\beta^{(l)}), l = 1, \ldots, L \)

Classify unlabeled classes using \( \hat{\beta} \) and Equation (14)

Hence, for some \( s = s_1 \) such that \( \sum_{i=1}^p \hat{\beta}_i = 1 \) is imposed, we have, as before,

\[
0 \leq \sum_{i=1}^p \beta_i P(Y = y_r|X_i = x_{ri}, \hat{\Omega}_i) \leq 1
\]  
(16)

and therefore

\[
\max_{\beta} \sum_{r=1}^n \sum_{i=1}^p \beta_i P(Y = y_r|X_i = x_{ri}, \hat{\Omega}_i)
\]

\[
= \min_{\hat{\beta}} \sum_{r=1}^n \left( 1 - \sum_{i=1}^p \beta_i P(Y = y_r|X_i = x_{ri}, \hat{\Omega}_i) \right) .
\]

(17)

Note that the above optimization problem is convex, because the objective function is quadratic on the parameters (and, thus, convex) and the inequality constraints are also convex. Hence, it is guaranteed to have a unique solution.

Thus, a vector \( \hat{\beta} \) solving (15) for \( s = s_1 \) will be an estimator of the maximizer of (10). Because of the variable selection effect of the lasso penalty, \( \hat{\beta} \) is expected to be sparse.

In this paper, instead of fixing \( s \) to \( s_1 \), we let \( s \) to traverse the whole regularization path, choosing it either to maximize the classification accuracy on a validation data set or to minimize some penalization criterion like AIC.

Equation (15) fulfills the necessary requirements\(^{14}\) to be solvable by an efficient LARS procedure. Specifically, as sufficient conditions, the loss function is a quadratic loss function and the penalty function is a lasso penalty. In Section 5, we derive a LARS-type algorithm with a couple of modifications to include the restriction \( 0 \leq \beta_i \leq 1 \). As we discuss below, this formulation allows us to discard both redundant and irrelevant predictors. The method, which we will call \( L_1 \)-NB, is summarized in Algorithm 1. In the pseudocode, AIC is used for model selection. It is defined as the number of parameters in the statistical model minus the likelihood.

Although our approach is definitely different from naïve Bayes, we rely, to some extent, on the same two principles. First, since the loss function in Equation (15) is linear on \( P(Y = y_r|X_i = x_{ri}, \hat{\Omega}_i) \), we are assuming that the values of the class
are linearly separable given the predictors. Naive Bayes establishes the same assumption. Second, both do not model any explicit relation between the predictors. Nonetheless, unlike naive Bayes, we are implicitly avoiding redundancy. This is detailed in the next section.

4. Redundant Predictors and Irrelevant Predictors

The proposed classifier can discard redundant predictors by solving (15). Let \( X_{i_1} \) and \( X_{i_2} \) be two redundant predictors, for example, a predictor that appears twice. First, if \( X_{i_1} \) and \( X_{i_2} \) are discrete and Equation (5) is satisfied, the value of \( X_{i_2} \) can be determined if \( X_{i_1} \) is known and vice versa. Hence, there is a bijection between the \( i_1 \)-th and the \( i_2 \)-th columns of matrix \( X \). Obviously, this means that \( P(X_{i_1} = x_{ri_1}, | Y = y_r, \Theta_{i_1}) \) and \( P(X_{i_2} = x_{ri_2}, | Y = y_r, \Theta_{i_2}) \) are equal, \( r = 1, \ldots, n \). Therefore, it follows from Equation (12) that \( P(Y = y_r | X_{i_1} = x_{ri_1}, \Theta_{i_1}) \) and \( P(Y = y_r | X_{i_2} = x_{ri_2}, \Theta_{i_2}) \), \( r = 1, \ldots, n \), are equal too.

Hence, if two predictors, \( X_{i_1} \) and \( X_{i_2} \), are highly correlated then vector \( P(Y = y_r | X_{i_1} = x_{ri_1}, \Theta_{i_1}), r = 1, \ldots, n \), and vector \( P(Y = y_r | X_{i_2} = x_{ri_2}, \Theta_{i_2}), r = 1, \ldots, n \), will also be highly correlated. Therefore, Equation (15), which can be solved by LARS, would drop either \( X_{i_1} \) or \( X_{i_2} \) due to the lasso constraint properties (i.e., the ability of the \( L_1 \)-penalty to discard redundant predictors).

If \( X_{i_1} \) and \( X_{i_2} \) are continuous and redundant, either \( X_{i_1} \) or \( X_{i_2} \) would also be discarded.

Proposition 4.1. If \( X_{i_1} \) and \( X_{i_2} \) are continuous and redundant, vector \( P(Y = y_r | X_{i_1} = x_{ri_1}, \Theta_{i_1}) \) and vector \( P(Y = y_r | X_{i_2} = x_{ri_2}, \Theta_{i_2}) \) \( r = 1, \ldots, n \) are equal.

Proof. If \( X_{i_1} \) and \( X_{i_2} \) are continuous and redundant, then \( X_{i_1} = g(X_{i_2}) \), \( g() \) being some deterministic linear function \( X_{i_1} = g(X_{i_2}) = b_0 + b_1 X_{i_2} \).

In this case, we have that \( \mu_{i_1} = b_1 \mu_{i_2} + b_0, \sigma_{i_1} = |b_1| \sigma_{i_2} \), and, trivially, \( f(x_{ri_1} | y_r; \mu_{i_1,y_r}, \sigma_{i_1}^2) = |b_1| f(x_{ri_2} | y_r; \mu_{i_2,y_r}, \sigma_{i_2}^2) \). By plugging this into Equation (12) we obtain

\[
P(Y = y_r | X_{i_1} = x_{ri_1}, \hat{\mu}_{i_2}, \hat{\sigma}_{i_2}^2) = \frac{|b_1|^{-1} f(x_{ri_2} | y_r; \hat{\mu}_{i_2,y_r}, \hat{\sigma}_{i_2}^2) P(Y = y_r)}{\sum_{j=1}^n |b_1|^{-1} f(x_{ri_2} | y_r; \hat{\mu}_{ij,y_r}, \hat{\sigma}_{ij}^2) P(Y = j)} = \frac{f(x_{ri_2} | y_r; \hat{\mu}_{i_2,y_r}, \hat{\sigma}_{i_2}^2) P(Y = y_r)}{\sum_{j=1}^n f(x_{ri_2} | y_r; \hat{\mu}_{ij,y_r}, \hat{\sigma}_{ij}^2) P(Y = j)} = P(Y = y_r | X_{i_2} = x_{ri_2}, \hat{\mu}_{i_2}, \hat{\sigma}_{i_2}^2).
\]

On the other hand, for all irrelevant predictors \( X_i \), following Equation (4), we have that

\[
P(Y = y_r | X_i = x_{ri}, \Theta_i) = P(Y = y_r), \quad r = 1, \ldots, n.
\]

1350019-8
Thus, irrelevant predictors give rise to equal vectors \( P(Y = y_r | X_i = x_{ri}, \hat{\Theta}_i) \), \( r = 1, \ldots, n \) (or approximately equal, when working with data sets) and will be also discarded.

### 5. An Efficient LARS-Type Algorithm

In this section, we present a LARS variant to accommodate the restriction \( 0 \leq \beta_i \leq 1 \). This restriction can be considered as two separate conditions: \( \beta_i \geq 0 \) and \( \beta_i \leq 1 \).

The LARS algorithm is an iterative procedure for multivariate regression that adds a predictor to the model at each step. LARS starts with no predictors. Firstly, it includes the predictor that is most correlated with the response into the active set of predictors \( \mathcal{A} \). The response is regressed on this predictor, so that the coefficient of this predictor is moved towards the least squares solution until a new predictor reaches the same absolute correlation with the vector of residuals as that of \( \mathcal{A} \). This new predictor is included in the active set \( \mathcal{A} \). Now, the vector of residuals is regressed on the predictors in \( \mathcal{A} \), moving their coefficients towards the joint least squares solution until a new predictor not in \( \mathcal{A} \) reaches the same absolute correlation with such vector of residuals as that of \( \mathcal{A} \). When \( n \geq p \), this procedure is repeated until all predictors are into the model. Otherwise, after \( n - 1 \) steps, the residuals are zero and the algorithm terminates.

We denote the LARS input matrix as \( B \), so that \( B_{ri} = P(Y = y_r | X_i = x_{ri}, \hat{\Theta}_i) \).

Let \( B_A \) be the columns of \( B \) indexed by \( A \), \( \hat{\beta}_A^{(l)} \) be the regression coefficients of the predictors in \( A \) at step \( l \), \( 1 \) be a column vector with \( n \) elements equal to one, and \( c = (c_1, \ldots, c_p) = B(1 - B_A \hat{\beta}_A^{(l)}) \) be the correlation with the residuals.

Hence, at each step, the coefficients in \( A \) are updated as

\[
\hat{\beta}_A^{(l+1)} = \hat{\beta}_A^{(l)} + \gamma w_A ,
\tag{18}
\]

where \( w_A \) is the joint least squares direction for the predictors in \( A \), and \( \gamma \) is “how much” \( \hat{\beta}_A^{(l)} \) must be updated at step \( l \). Then, \( \gamma \) is computed as the minimum value such that some predictor \( i \notin A \) reaches the same absolute correlation with such vector of residuals as that of \( A \). Algebraic details about the exact computation of \( \gamma \) and \( w_A \) were described by Efron et al.\(^{13}\)

The LARS modification for computing the exact regularization path of the lasso problem is based on detecting when a non-zero coefficient hits zero. Then, this predictor is dropped from \( A \) and the new least squares direction is computed. Working out \( \gamma \) in (18), for each predictor \( i \in A \), this happens when \( \gamma \) reaches

\[
\gamma_i = -\frac{\hat{\beta}_i^{(l)}}{w_i} ,
\tag{19}
\]

It will happen first at

\[
\tilde{\gamma} = \min_{\gamma_i > 0} \{ \gamma_i \} .
\tag{20}
\]
Hence, if $\tilde{\gamma} < \gamma$, $\gamma$ is corrected to be $\tilde{\gamma}$, and the new coefficients are computed by (18). The corresponding predictor is dropped from $A$ for the next iteration.

Now, to accomplish the first condition $\beta_i \geq 0$, we compute $\gamma$ as the minimum value such that some predictor $i \notin A$ reaches the same positive correlation with the vector of residuals as that of $A$. Thus, the difference is that the negative correlations with the residuals of predictors $i \notin A$ are ignored for computing $\gamma$ and deciding which predictor $i \notin A$ enters the model. This modification was presented in the paper by Efron et al.\textsuperscript{13}

Condition $\beta_i \leq 1$ is not in the literature and is slightly more complex. In this case, we need to detect when a regression coefficient hits 1. Let $M$ be the set containing all predictors that have already reached 1. Again, for each predictor $i \in A$, we work out $\gamma$ in (18):

$$\gamma_i = \frac{1 - \tilde{\beta}_i^{(l)}}{w_i},$$

so that

$$\tilde{\gamma} = \min_{\gamma_i > 0} \{\gamma_i\}.$$ (22)

If $\tilde{\gamma} < \gamma$, then we would set $\gamma = \tilde{\gamma}$, compute the new coefficients by (18) and move this predictor from $A$ to $M$. The new direction $w_A$ is computed on the current residual as usual.

However, it is well known that, when there is some dependence between the predictors, some predictors can decrease their regression coefficients at some step of the algorithm. We need to verify when it happens for predictors in $M$, because they would detach from 1 and should be included in $A$ again. Since the regularization path is piecewise linear, it can only occur when a new predictor is included into or dropped from the model.

Let us include the predictors in $M$ into the calculation of the joint least squares direction at step $l$. Let $w_{A \cup M}$ denote this direction, assuming that $M$ is not empty. Predictors from $M$ that have a positive direction $w_i \geq 0$ are definitely discarded at this step. Let $M^-$ contain all predictors in $M$ excepting predictors with direction $w_i \geq 0$. Now, we calculate a new joint least squares direction $w_{A \cup M^-}$. Again, we check if there are predictors in $M^-$ whose direction $w_i$ is positive. If this occurs, we delete them from $M^-$ and update $w_{A \cup M^-}$. In summary, at each step $l$, this procedure must be repeated until $M^-$ is empty or all its predictors have a negative direction. These predictors must be moved from $M$ to $A$ for the next step.

Notice that, unlike $\beta_i \geq 0$, the $\beta_i \leq 1$ restriction implies additional computations. Specifically, at each step, additional least squares directions must be computed if $M$ is not empty. If $p$ is high and efficiency is a main concern, a possibility is, once regression coefficients reach 1, to attach these predictors to $M$ for the rest of the algorithm. Hence, at each step, the joint least squares direction is only computed in $A$ and we do not need to check whether any predictor in $M$ has to be moved.
An $L_1$-Regularized Naïve Bayes-Inspired Classifier

to $\mathcal{A}$. Note that this can potentially produce a different regularization path. This is the approach followed in this paper because the exact calculation of the regularization path is not crucial.

Summing up, the three described modifications are trivially combined by choosing $\gamma$ as the value that first triggers any of the following events:

- A non-zero coefficient hits zero (Equations (19), (20)).
- Some predictor $i \notin \mathcal{A}$ reaches the same positive correlation with the vector of residuals as that of $\mathcal{A}$.
- A non-zero coefficient hits 1 (Equations (21), (22)).

Note that the computational cost of the LARS algorithm is dominated by the inversion of $B'_{\mathcal{A}}B_{\mathcal{A}}$ for computing the joint least squares direction at each step. The entire LARS solution path for $p < n$ variables, however, can be computed at the same cost than a least squares fit, i.e., $O(p^3 + np^2)$. This is achieved by updating the Cholesky factorization of $B'_{\mathcal{A}}B_{\mathcal{A}}$ found at the previous step. At the final step, we have computed the Cholesky factorization of $B'B$. Nevertheless, the introduced modifications can induce more than $p$ steps, and, hence, the computational cost can be slightly increased.

For example, our approach takes around 7.1 seconds for a data set with $n = 1000$, $p = 100$ and four non-spurious predictors, whereas selective naïve Bayes takes approximately 140.0 seconds and prefiltering by mutual information conditional on the class (following the approach introduced by Fleuret) takes 19.1 seconds.

6. Experiments

We present some illustrative results on two different scenarios. First, we evaluate the effect of redundant and irrelevant predictors. Second, we test the proposed method on a high dimensional data set. Finally, we run the naïve Bayes methods on a data set that combine numeric with categorical predictors.

6.1. Irrelevance and redundancy

In this section, we test the behavior of our method on one of the Soybean data sets, where all the predictors are discrete with four or five categories. We focus on the version with no missing values, called Soybean Small in the UCI repository. This data set has $n = 47$ instances, $p = 21$ predictors and four classes, whose relative proportions are $(0.21, 0.21, 0.21, 0.37)$. We have chosen Soybean because it is a well-behaved data set, suitable for testing how sensitive the algorithm is to the above issues.

Based on the original data set, we built several new data sets by adding different numbers of irrelevant and redundant predictors. We added $0$, $p$, $2p$, $3p$, $4p$ and $5p$ irrelevant (randomly generated) predictors, and the same numbers of redundant predictors. We tested all combinations of redundancy and irrelevance. Redundant
predictors are randomly generated values that are highly correlated (0.8) with an existing predictor, which is itself highly correlated to the class. We have tested a total of $6 \times 6 = 36$ data sets.

We compared the proposed method to ordinary naïve Bayes, naïve Bayes with prefiltering feature selection, weighted naïve Bayes with prefiltering feature selection and selective naïve Bayes. Prefiltering is based on mutual information to the class. We introduced three random predictors sampled from a multinomial distribution with five categories and equal probabilities for each category. Afterwards, we discarded those predictors whose mutual information is lower than one of the three random predictors. An analogous prefiltering approach was taken for example by Bi et al.\textsuperscript{21}

For each data set we performed 5-fold cross-validation, so that 80\% of the data is used for training at each fold. Model evaluation was based on the AIC statistic. Note that this is needed by both our approach (for selecting $\lambda$) and selective naïve Bayes.

Graphs in Figs. 1 and 2 show, respectively, the accuracy and the number of selected predictors. For a given number of irrelevant predictors, each graph displays the results for increasing numbers of redundant predictors. Figure 1 indicates with a horizontal thick line that the difference of the $L_1$-NB accuracy to the second best method is statistically significant with a significance level of 0.05. We do not show the number of correctly selected predictors because it is not clear which variables from the original set should really be selected. The total number of predictors in the data set is marked by the ordinary naïve Bayes line.

As expected, irrelevant predictors do not affect the performance of the evaluated classifiers much, except for selective naïve Bayes. Their accuracies do not greatly decrease as the number of irrelevant predictors grows. On the other hand, excepting our approach and selective naïve Bayes, there is an increment of selected predictors for data sets containing more irrelevant predictors.

The effect of redundant predictors is stronger. As a general rule, selective naïve Bayes exhibits lower accuracy in the presence of redundant predictors. The $L_1$-NB accuracy is the least affected by this issue, and, generally, it shows the best classification performance. Note that accuracy is very similar for ordinary naïve Bayes and weighted naïve Bayes. More impressive are the graphs considering the number of selected predictors. As expected, prefiltering does not satisfactorily handle redundancy. The more redundant predictors there are, the greater the number of selected predictors. On the other hand, the number of selected variables for $L_1$-NB and selective naïve Bayes barely fluctuates at around 3 predictors for all data sets, always selected from the original set of variables.

### 6.2. High-dimensional data: brain imaging

The discrimination of mental states from neural activity is a hot topic in cognitive neuroscience. Data are usually high-dimensional. Functional magnetic resonance
imaging (fMRI) is of particular interest. Such data often contains thousands or even millions of predictors mapping 3D voxels.

In this paper we deal with a data set that considers visual stimuli\textsuperscript{22} as provided within the MVPA MatLab Toolbox.\textsuperscript{a} A single subject is analyzed over 12 trials. At

\textsuperscript{a}http://code.google.com/p/princeton-mvpa-toolbox
each trial, the subject is shown pictures illustrating each of eight types of content (classes) for a length of time. A brain image is taken every few seconds. Each image is thus an instance, also referred to as repetition time (TR). At each trial, we have 9 TRs for each content type. Also, there are some TRs that do not match any content. We ignore these no-content TRs, so that $n = 12 \times 8 \times 9 = 864$ instances are available. The relative proportions of the classes are thus equal. There are $p = 39912$ voxels.
Table 1. Mean accuracy (and standard deviation) and mean number of selected predictors (and standard deviation) for $L_1$-NB, discretized $L_1$-NB (d$L_1$-NB), naïve Bayes (NB), discretized naïve Bayes (dNB), weighted naïve Bayes (WNB), discretized weighted naïve Bayes (dWNB), selective naïve Bayes (SNB), discretized selective naïve Bayes (dSNB), $k$-nearest neighbors (KNN) and Support Vector Machine (SVM). Excepting $L_1$-NB and d$L_1$-NB, all methods use FSS. Best results are highlighted.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (±)</th>
<th>#predictors (±)</th>
<th>Method</th>
<th>Accuracy (±)</th>
<th>#predictors (±)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1$-NB</td>
<td>0.29(±0.09)</td>
<td>511(±130)</td>
<td>WNB+FSS</td>
<td>0.33(±0.01)</td>
<td>100.0(±0.0)</td>
</tr>
<tr>
<td>d$L_1$-NB</td>
<td><strong>0.45</strong></td>
<td>509(±93)</td>
<td>dWNB+FSS</td>
<td>0.44(±0.01)</td>
<td>100.0(±0.0)</td>
</tr>
<tr>
<td>NB+FSS</td>
<td>0.37(±0.01)</td>
<td>100.0(±0.0)</td>
<td>SNB+FSS</td>
<td>0.34(±0.02)</td>
<td><strong>60.5(±6.2)</strong></td>
</tr>
<tr>
<td>dNB+FSS</td>
<td>0.40(±0.01)</td>
<td>100.0(±0.0)</td>
<td>dSNB+FSS</td>
<td>0.40(±0.01)</td>
<td>75.5(±3.3)</td>
</tr>
<tr>
<td>KNN+FSS</td>
<td>0.39(±0.1)</td>
<td>100.0(±0.0)</td>
<td>SVM+FSS</td>
<td>0.31(±0.01)</td>
<td>100.0(±0.0)</td>
</tr>
</tbody>
</table>

We have tested the proposed method, naïve Bayes, weighted naïve Bayes and selective naïve Bayes on this data set. All these classifiers are also trained over a discretized version of the data set. Discretization conformed to the MDL-based scheme described by Fayyad and Irani.\textsuperscript{23} In order to compare with other classification paradigms, we have also run a support vector machine (SVM) with a radial kernel and $k$-nearest neighbors (KNN), where the number of neighbors is chosen by cross-validation.

All except $L_1$-NB were preceded by feature subset selection (FSS). For selective naïve Bayes, this is necessary on computational grounds. In this case, for the comparison to be fair, FSS has been performed by using mutual information conditional on the class, following the procedure proposed by Fleuret.\textsuperscript{24} With this method, we can identify both irrelevant and redundant predictors.

Taking advantage of the trial structure of the data set, we performed 12-fold cross-validation for all learning procedures, leaving out one trial at each iteration for testing. At each fold, one trial was reserved for model selection and determination of the number of predictors in the prefiltering step. Table 1 presents the results.

All naïve Bayes-based methods behave better on the discretized data set. The discretized $L_1$-NB method shows the best overall accuracy, followed by weighted naïve Bayes with FSS. The differences between the discretized $L_1$-NB and the other methods (excepting discretized weighted naïve Bayes with FSS) are statistically significant with a significance level of 0.01. The performance of the $L_1$-NB method for the non-discretized data set is however poor.

With regard to variable selection, although $L_1$-NB selects a higher number of predictors than selective naïve Bayes, the number of selected predictors for $L_1$-NB is not out of proportion. Note that FSS always chooses 100 variables, being 10, 100 and 1000 the possible choices.

It is known that sparse brain areas are simultaneously activated under certain stimuli. The distributed nature of the brain is very closely related to redundancy from a pattern analysis perspective. This might explain why the discretized $L_1$-NB performs better than SNB and the approaches based on naïve Bayes and FSS. Both
SNB and conditional mutual information FSS cope with interaction between input variables in a somewhat roughly manner, either selecting or discarding completely the input variables.  

It appears that the assumptions of building a Gaussian naive Bayes model for fMRI data are too strong. This could be the cause of the lower performance of the classifiers for non-discretized data. The normality assumption for the predictors given the class is not always met. For example, if we take the voxel that is most correlated to the class and perform a Shapiro-Wilk hypothesis test to check normality within each class, we obtain 0.376, 0.276, 0.3608, 0.4819, 0.001, 0.565, 0.021 and 0.0579 p-values. For a p-value threshold of 0.05, the predictor does not follow a normal distribution within classes 5 and 7. Only 1045 out of 39912 voxels (a proportion of 0.025) fulfill the normality assumption within the eight classes. On average, voxels fulfill the normality assumption only within 2.2 classes. The superior performance of the naive Bayes classifiers when they are applied on discretized data was reported.

6.3. Flags data set

The Flags data set from the UCI repository contains information about countries and their flags. It has \( N = 194 \) instances and \( p = 30 \) (numeric and categorical) features. We have chosen the religion of the country as the response, for a total of six different values of the class, whose relative proportions are (0.16, 0.09, 0.18, 0.27, 0.20, 0.10). We have tested all the aforementioned naive Bayes classifiers without discretizing, as they deal more naturally with data sets with different types of variables. We have not included kNN and SVM in the comparison, because they do not work that straightforwardly on mixed numeric and categorical sets of features. Previous feature subset selection has been performed for the naive Bayes and weighted naive Bayes classifiers using conditional mutual information.

At each cross-validation iteration, one fourth of the training data was reserved for choosing the number of preselected variables. Table 2 shows the results over 10-fold cross-validation.

In this data set, \( L_1 \)-NB and SNB perform better (with a non-statistically significant advantage of SNB) than the (weighted) naive Bayes, which indicates a preference of this data set for wrapped feature selection over prefiltering. The number of variables selected by \( L_1 \)-NB is however lower than that of SNB.

Table 2. Mean accuracy (and standard deviation) and mean number of selected predictors (and standard deviation) for \( L_1 \)-NB, naive Bayes (NB), weighted naive Bayes (WNB) and selective naive Bayes (SNB). NB and WNB use FSS. Best results are highlighted.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (±std)</th>
<th>#predictors (±std)</th>
<th>Method</th>
<th>Accuracy (±std)</th>
<th>#predictors (±std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_1 )-NB</td>
<td>0.49 (±0.13)</td>
<td>3.8 (±0.63)</td>
<td>WNB+FSS</td>
<td>0.30 (±0.10)</td>
<td>5.5 (±0.12)</td>
</tr>
<tr>
<td>NB+FSS</td>
<td>0.21 (±0.08)</td>
<td>5.5 (±0.12)</td>
<td>SNB</td>
<td>0.50 (±0.13)</td>
<td>12.9 (±1.72)</td>
</tr>
</tbody>
</table>
Summing up, through synthetic and real data experiments, we have shown that
the proposed method is a flexible classifier. In particular, it deals with both numeric
and continuous predictors and, unlike most naive Bayes methods, behaves
reasonably well when there exists a strong correlation between predictors.

7. Conclusions and Future Work
So far, we have discussed the issue of irrelevant predictors and redundant predictors
for the naive Bayes model. We have proposed a model that, initially inspired by the
naive Bayes scheme, deals reasonably well with these spurious predictors.

This has been proved empirically on several data sets, where different numbers
of irrelevant and redundant predictors have been added. As shown, our method
works on both discrete and continuous data sets. Moreover, a high-dimensional
setting, extracted from the neuroscience domain, has been tested. We found that
the proposed method works much better on a discretized version of this data set.

Like the naive Bayes model, we have not explicitly considered dependence be-
tween predictors in this paper. However, since the $L_1$-penalty deals with redun-
dancy (as seen above, with regard to the loss function), we can discard redundant
predictors.

In the future, we plan to extend this or alternative formulations for exploring
more complex predictor relations than redundancy. Relaxations in the attribute in-
dependence assumption have been explored. We intend to pursue this line. Multi-
label classification, where dependences between the response variables come into
play, is also on the agenda. We also want to tackle the semi-supervised learning
task, where some values of Y might be missing, as well as the detection of emerging
new classes.

Acknowledgment
Research partially supported by the Spanish Ministry of Economy and Competi-
tiveness, projects TIN2010-20900-C04-04, Consolider Ingenio 2010-CSD2007-00018
and Cajal Blue Brain.

References
1. N. Friedman, D. Geiger, and M. Goldszmidt, Bayesian network classifiers Machine
2. P. Domingos and M. Pazzani, Beyond independence: Conditions for the optimality of
the simple Bayesian classifier, Machine Learning 29 (1997) 103–130.
4. J. T. A. S. Ferreira, D. G. T. Denison, and D. J. Hand, Data mining with products
of trees, in Advances in Intelligent Data Analysis, Volume 2189 of Lecture Notes in
5. P. Langley and S. Sage, Induction of selective Bayesian classifiers, in Proc. of the 10th

1350019-17
D. Vidaurre, C. Bielza & P. Larrañaga