Variable Selection for Binary Classification on High Dimensional Neural Planning Data

Anonymous Author(s)
Affiliation
Address
e-mail

Abstract

Variable selection is hard for high dimensional data when the number of predictors is larger than the number of observations. Methods are needed that could select the variables having a large influence on the response. This paper implements stability selection to identify the important variables. Stability selection is based on subsampling and selection algorithms. After the variables are selected, neural networks are built to fit the model. Then K-fold cross-validation is used to evaluate the classification accuracy of the model. Stability selection combined with neural networks work quite well in making predictions of classification with high dimensional neural planning data. An average of 98.7% classification accuracy out of 1000 times is obtained from eight-fold cross-validation. The selection of variables is robust when some noisy variables are added.

1 Motivation

The motivation of this project is to build a classifier for the neural planning data. The data was collected from rats which performed one of two simple tasks. The tasks were to press three levers in a row in one of two orders: 1: 'right lever, middle lever, left lever,' or 0: 'left lever, right lever, middle lever.' As long as the rat completed the task successfully, it received a reward. Things in common for the two tasks were the rat will press the middle lever after the right lever. The neural excitation patterns were measured in the time leading up to pressing the 'middle lever' after pressing the 'right lever'. If the classifier built could distinguish between the two tasks, it suggests that the rats employ planning when performing tasks 1 and 0. However, there are 170 features to describe neural excitation patterns and there are only 32 samples. It is of interest to select important features and build a classifier that could predict which task each rat performed.

2 Objective

In this practical setting, we have a supervised problem. The data consists of \((X_1, Y_1), \ldots, (X_n, Y_n)\) with univariate response variable \(Y\) and \(p\)-dimensional covariates \(X\). \(Y\) is a binary variable which takes values from 0 or 1 to label the tasks performed by each rat. \(X\) is a 170 dimensional covariates denoting 170 different neural excitation patterns. There are 32 rats so that the sample size \(n\) is 32. It is assumed that \((X_i, Y_i)\)s are independent and identically distributed meaning that the data collected from each rat does not have an influence on other rats and the distribution of \((X_i, Y_i)\) from each rat follows the same distribution. Our goal is to identify a small subset of important covariates instead of using 170 dimensional covariates to predict \(Y\). In other words, conditional on this subset of the important covariates, the response will not depend on the other unselected covariates.
3 Data Description

The neural excitation patterns were measured by Instant Firing Rates (in Hertz) averaged over 200 ms periods. Measurements were obtained from 34 cells over 1 second. Within 1 second for a single cell, there were 5 Instant Firing Rates. In total, for each rat, there were 170 neural excitation measurements. There were 32 rats in the experiment. The data was in the format of a 32 × 171 matrix with the last column specifying the task label of each single rat.

4 Methods

4.1 Variable Selection with the Lasso

The Lasso is a widely used in model selection, estimation and prediction in regression models. It represents the Least Absolute Shrinkage and Selection Operator which was first proposed by Tibshirani (1996). In order to understand how the Lasso works for variable selection, it is first introduced in a linear regression setting before proceeding to the more complicated case of the Lasso Logistic Regression Model which is used in this project.

In the Gaussian linear regression model, the data is observed as \((y_1, x_1), \ldots, (y_n, x_n)\) where each pair of data is a realization of a random variable \(Y_i\) and \(X_i = (1, X_{i1}, \ldots, X_{ip})\) is a \((p+1)\)-dimensional covariates. It is assumed that

\[
Y_i = (X\beta)_i + \epsilon_i
\]

where each \(\epsilon_i\) is random noise i.i.d. \(N(0, \sigma^2)\), \(i=1, \ldots, n\).

If the sample size \(n\) is larger than the dimension of covariates \(p\), the parameters \(\beta\) could be inferred by least square estimators where \(\hat{\beta} = (X^T X)^{-1}X^T Y\). However, if \(p >> n\) in the high-dimensional setting, \(X^T X\) is singular making the parameter \(\beta\) non-identifiable. In order to deal with the difficulty of inferring \(\beta\) when \(p >> n\), it is assumed that the true model is sparsed meaning that only a subset of the covariates will have an influence on the response \(Y\). To describe this mathematically, \(\beta\) is a sparse \(p\)-dimensional vector where only \(s < p\) components of \(\beta\) is non-zero. Denoting \(S = \{j : \beta_j \neq 0\}\), the index of covariates having an effect on the response. The objective is to infer \(S\) by the observed data.

The Lasso solves the problem by finding \(\beta\) such that

\[
\min_{\beta} (Y - X\beta)^T (Y - X\beta) \quad \text{st} \quad \sum_{j=1}^{p} |\beta_j| \leq t
\]

where \(t > 0\) is a tuning parameter or equivalently to solve the penalized likelihood problem

\[
\min_{\beta} \frac{1}{n} (Y - X\beta)^T (Y - X\beta) + \lambda \sum_{i=1}^{p} |\beta_j|
\]

where \(\lambda \in [0, \infty)\) is a regularization parameter and the observed data \((Y, X)\) has been standardized.

By using an \(L_1\) penalty \(\lambda \sum_{i=1}^{p} |\beta_j|\) on the regression coefficients, a sparse model of the estimator \(\hat{\beta}(\lambda)\) could be obtained and the overall optimization problem is convex which stimulates efficient algorithms for solving this problem when the number of covariates \(p\) is even larger than \(10^5\).

4.2 The Lasso Logistic Regression Model

In logistic regression model, the data is recorded as \((y_1, x_1), \ldots, (y_n, x_n)\) where each pair of data is a realization of a binary random variable \(Y_i\) taking values of 0 or 1 and \(x_i = (1, x_{i1}, \ldots, x_{ip})\) is a \((p+1)\)-dimensional covariates, \(1 \leq i \leq n\). Define the sigmoid function \(\pi(t) = \frac{1}{1 + \exp(-t)}\). The logistic regression model assumes that

\[
P(y_i = 1 | X_i, \beta) = \pi(x_i \beta) = \frac{1}{1 + \exp(-x_i \beta)}
\]
\[
P(y_i = 0 | X_i, \beta) = 1 - \pi(x_i \beta) = \frac{\exp(-x_i \beta)}{1 + \exp(-x_i \beta)}
\]

where \( \beta = (\beta_0, \beta_1, \ldots, \beta_p)' \) is a \((p+1)\) dimensional coefficients to be estimated. The negative log-likelihood function is

\[
l_n(\beta) = -\sum_{i=1}^{n} \{y_i \log \pi(x_i \beta) + (1 - y_i) \log(1 - \pi(x_i \beta))\}
\]

However, if \( p >> n \), similar to the situation when \( p >> n \) in linear regression models, to estimate \( \beta \), a penalty term should be added to the negative log-likelihood function.

\[
L(\beta; \lambda) = l_n(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|
\]

where only coefficients \( \beta_1, \ldots, \beta_p \) are penalized but not the intercept \( \beta_0 \). The Lasso estimator for logistic regression model is

\[
\hat{\beta}(\lambda) = \arg \min_{\beta} L(\beta; \lambda)
\]

However, by choosing different regularization parameters \( \lambda \), the index of the components of \( \beta \) which are non-zero might be different. It is of interest to study the consistency procedure of the Lasso. A variable selection procedure is defined to be consistent if

\[
P(S) \to 1
\]

where \( S = \{ j : \hat{\beta}_j \neq 0 \} \) is the index of the estimated non-zero coefficients. In other words, \( \hat{S} \) denotes the index of the important covariates that have an influence on the response. While \( S = \{ j : \beta_j \neq 0 \} \) is the true set of the index for the non-zero coefficients corresponding to the true important covariates. In Leng’s paper [1], it has been shown that when the tuning parameter is chosen to minimize the prediction error, the Lasso is not consistent variable selectors. The consistency properties of the Lasso has been discussed in Section 3.1 in detail in Meinshausen’s paper [2].

Since when certain assumptions are not satisfied, the consistency properties of the Lasso can not be guaranteed. It implies that even when the sample size \( n \) goes to infinity, the probability that the variables we select from the Lasso are the true important covariates does not have a limit of 1. In order to deal with this problem, stability selection is implemented in this project to conduct variable selection.

### 4.3 Stability Selection

#### 4.3.1 The Advantages of Stability Selection

According to Meinshausen’s paper [2], there are two advantages when using stability selection. Results are much less sensitive the choice of regularization parameters which is important and difficult for variable selection. Second, in some settings where the original variable selection methods fail while stability selection makes this variable selection methods consistent. It has also been shown in this paper that for the randomized lasso that stability selection will be variable selection consistent even if the necessary conditions for the consistency of the original lasso method are violated.

#### 4.3.2 Stability Selection Procedures

The general idea of stability selection is that given a set of regularization parameters \( \lambda \), by subsampling half of the original number of samples without replacement from the original datasets for \( N \) times and performing variable selection techniques, for each of the regularization parameter in the set, we could get a frequency of each variable being selected out of \( N \) times. The largest frequency over the set of regularization parameters for a single variable will be selected as the probability for this variable to be selected. Then by choosing a probability threshold, variables with a high selection probability will be kept and those with low probabilities will be abandoned. To state this more rigorously, the following definitions of stability paths and stable variables need to be introduced. The following two definitions are cited in Section 2.1 and Section 2.3 from Meinshausen’s paper [2].
Definition 1 (Selection Probabilities) Let $I$ be a random subsample of $\{1, \ldots, n\}$ of size $[n/2]$, drawn without replacement. For every set $K \subset \{1, \ldots, p\}$, the probability of being in the selected set $\hat{S}^\lambda(I)$ is

$$\hat{\Pi}^\lambda_K = P^*(K \subset \hat{S}^\lambda(I))$$

where the probability $P^*$ is with respect to the random subsampling and other sources of randomness if $\hat{S}^\lambda(I)$ is a randomized algorithm. The stability paths are the probability for every variable being selected when randomly subsampling from the data. The stability path varies for different choice of regularization parameters $\lambda$.

Definition 2 (Stable Variables) For a cut-off $\pi_{thr}$ with $0 < \pi_{thr} < 1$ and a set of regularization parameters $\Lambda$, the set of stable variables is defined as

$$\hat{S}_{\text{stable}} = \{k : \max_{\lambda \in \Lambda} (\hat{\Pi}^\lambda_K) \geq \pi_{thr}\}$$

The stability selection with the Lasso logistic regression procedures to perform the variable selection are described as below.

- subsample from the original dataset with size $n/2$ without replacement
- run the Lasso logistic regression selection algorithm $\hat{S}^\lambda(I)$ on $I$
- repeat these steps many times and compute the selection probability $\hat{\Pi}^\lambda_K$ for each variable
- choose a selection probability threshold and determine the stable variables as the selected variables

In this sense, by choosing a proper threshold $\pi_{thr}$ and a set of regularization parameters, the stable variables are the variables that are selected. The good thing about stability selection is that the variable selection results vary very little for sensible choices in a range of probability thresholds. It does not depend on the choice of regularization parameters strongly neither.

4.4 Neural Networks with Resilient Backpropagation

In this project, stability selection with the Lasso logistic regression model is initially used to perform variable selection. After the important variables are selected, neural networks are built with an R package neuralnet to learn the complicated relationship between the response and covariates consisting of the important selected variables.

4.4.1 Training of Neural Networks

Artificial neural networks could work to discover complicated relationship between response variables and covariates. It does not specify the type of relationship of response variables and covariates before fitting the model. The neural networks learn the parameters iteratively by starting from random variables drawn from the standard normal distribution and learn the parameters step by step to minimize some specified error function until a prespecified condition has been achieved. Moreover, the neuralnet could handle an arbitrary number of covariates, response variables and also the hidden layers theoretically however it could have an early stop due to convergence problem or the maximum iterations have been obtained.

The package neuralnet uses multi-layer perceptrons to model the functional relationship. The input layer includes all covariates in separate neurons and the output layers consist of the response variables. The synapse which is the weight of each input in each layer is to be learnt. To illustrate how multi-layer perceptrons work, examples on only one layer and two layers are introduced.

When there is only one input layer and one output neuron, the following function is calculated to represent the relationship between response variables and covariates

$$o(x) = f(\omega_0 + \sum_{i=1}^{n} \omega_i x_i)$$
where $\omega_0$ is the intercept and $\omega_i$ is the weight for $x_i$. When there is one hidden layer with $J$ hidden neurons in the model, the following function is calculated

$$o(x) = f(\omega_0 + \sum_{j=1}^{J} \omega_j f(\omega_0 + \sum_{i=1}^{n} \omega_{ij} x_i))$$

To state this in a more general way, in each layer, the output is calculated by the input neurons $z_1, \ldots, z_k$ as $f(g(\omega_0 z_0 + \sum_{i=1}^{k} \omega_k z_k))$ where $z_0 \equiv 1$ and $f$ is known as the activation function and $g$ is the integration function.

4.4.2 Resilient Backpropagation

The resilient backpropagation method is used to learn the weights in order to achieve a local minimum of the error function. In this case, if the partial derivative is negative, then the weight will increase; if the partial derivative is positive, then the weight will decrease. The difference between resilient backpropagation and traditional backpropagation is the way it updates the weight in each iteration step. In resilient backpropagation, the weights are updated as

$$\omega_k^{(t+1)} = \omega_k^{(t)} - \eta_k^{(t)} \text{sign} \left( \frac{\partial E^{(t)}}{\partial \omega_k^{(t)}} \right)$$

where $E^{(t)}$ is the error function. In traditional backpropagation, the weights are updated as

$$\omega_k^{(t+1)} = \omega_k^{(t)} - \eta \frac{\partial E^{(t)}}{\partial \omega_k^{(t)}}$$

It could be seen that the learning rate $\eta$ is the same for all weights in all the iteration steps in traditional backpropagation but it is different for different weights and different iteration steps which is more flexible. Moreover, the sign of the partial derivatives of weights is used instead of its magnitude so that an equal effect of the learning rate could be assured.

5 Results

The dataset is a $32 \times 171$ matrix with 170 covariates and the last column specifying the task label of each single rat. However, in some columns of the covariates, all elements are zero. After deletion of the zero columns, there are 158 covariates with sample size 32. Then the data are standardized to make sure that $1^T y = 0$, $1^T x_j = 0$ and $x_j^T x_j = 1$ for $j = 1, \ldots, 158$.

To implement stability selection, a set of regularization parameters $\lambda$ need to be produced. Since the selection of stable variables does not rely strongly on the choice of regularization parameters, the set $\Lambda_0$ of regularization parameters $\lambda$ are obtained by fitting the lasso logistic regression using all the 158 covariates and response variable $y$. The produced $\Lambda_0$ serves as the set in the stability paths.

Since the sample size is 32, when conducting stability selection, each time we subsample 16 samples without replacement and then use this subsample to fit the Lasso logistic regression model over the regularization parameters set $\Lambda_0$. In other words, for every subsample, the Lasso logistic regression model is fitted under each $\lambda$ from $\Lambda_0$. For each covariate, it is selected meaning that the coefficient $\beta_j$ is not zero under at least one of the $\lambda$s in each sub-sample, then we consider this variable is selected under this subsample. Later we repeat subsampling for 1000 times and then the selection frequency of each variable could be obtained out of 1000 times. The results of estimated selection probability is summarized in Table 1. For example, the selection probability for variable 13 is 0.603 meaning that out of 1000 times, this variable has been selected 603 times.
Table 1: Stability selection estimates greater than 0.3 threshold

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relative Selection Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 V13</td>
<td>0.604</td>
</tr>
<tr>
<td>2 V138</td>
<td>0.593</td>
</tr>
<tr>
<td>3 V66</td>
<td>0.494</td>
</tr>
<tr>
<td>4 V45</td>
<td>0.492</td>
</tr>
<tr>
<td>5 V141</td>
<td>0.472</td>
</tr>
<tr>
<td>6 V32</td>
<td>0.447</td>
</tr>
<tr>
<td>7 V57</td>
<td>0.386</td>
</tr>
<tr>
<td>8 V25</td>
<td>0.352</td>
</tr>
<tr>
<td>9 V97</td>
<td>0.335</td>
</tr>
<tr>
<td>10 V96</td>
<td>0.331</td>
</tr>
<tr>
<td>11 V19</td>
<td>0.301</td>
</tr>
</tbody>
</table>

By choosing different selection probability thresholds, different variables will be selected. If we choose the threshold as 0.5, then only V138 and V13 will be selected. If the threshold is chosen as 0.3, V96, V19, V97, V25, V57, V32, V45, V66, V141, V13 and V138 will be selected. In this project, we fit five models of different combinations of covariates using neural networks. Then we evaluate each model by using leave-one-out cross-validation (LOOCV) and 8-fold cross-validation to observe the prediction accuracy. By bootstrapping 1000 times, the mean and standard error for the prediction error has also been provided. The variables selected in the five models are shown below in Table 2.

Table 2: Variables selected from Model 1 to Model 5

<table>
<thead>
<tr>
<th>MODEL</th>
<th>INDEX OF SELECTED VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>138, 13</td>
</tr>
<tr>
<td>Model 2</td>
<td>138, 13, 45, 66</td>
</tr>
<tr>
<td>Model 3</td>
<td>96, 19, 97, 25, 57, 32, 45, 66, 141, 13 and 138</td>
</tr>
<tr>
<td>Model 4</td>
<td>13, 138, 45, 66, 8, 9, 10, 12, 37</td>
</tr>
<tr>
<td>Model 5</td>
<td>8, 9, 10, 12</td>
</tr>
</tbody>
</table>

In the first three models, the variables chosen are all good ones. The variables selected in Model 1 have selection probability larger than 0.5. In Model 2, the variables are selected with the first four highest selection probabilities. In Model 3, variables involved are the ones having the highest 11 selection probabilities. However, in order to discover the effect on the prediction accuracy of bad variables with selection probability zero, V8, V9, V10, V12, V37 are included in Model 4. We are also curious about the prediction accuracy (PR) if we only include bad variables which is in Model 5. The results are summarized in Table 3.

Table 3: Prediction accuracy for from Model 1 to Model 5

<table>
<thead>
<tr>
<th>MODEL</th>
<th>PR FOR LOOCV</th>
<th>PR FOR 8-FOLD</th>
<th>SE OF PR FOR 8-FOLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>0.813</td>
<td>0.826</td>
<td>0.064</td>
</tr>
<tr>
<td>Model 2</td>
<td>0.903</td>
<td>0.891</td>
<td>0.053</td>
</tr>
<tr>
<td>Model 3</td>
<td>1.000</td>
<td>0.987</td>
<td>0.021</td>
</tr>
<tr>
<td>Model 4</td>
<td>0.903</td>
<td>0.869</td>
<td>0.059</td>
</tr>
<tr>
<td>Model 5</td>
<td>0.625</td>
<td>0.562</td>
<td>0.084</td>
</tr>
</tbody>
</table>

In Table 3, PR stands for prediction accuracy. LOOCV represents leave-one-out cross-validation. SE means standard error. In the sense of prediction accuracy, Model 3 with 11 selected variables is
preferred. In this model, the parameters from fitting the neural networks are shown in Figure 1. The numbers on the lines are the weights for each input. In this case, there is one hidden layer. Since \( y \) is a binary response, the activation function \( f \) is a sigmoid function.

![Figure 1: Estimated Parameters](image)

6 Discussions

From Table 3, it could be concluded that we would like to select Model 5 including V96, V19, V97, V25, V57, V32, V45, V66, V141, V13 and V138 to help predict the label of tasks each rat performed. By using stability selection and the Lasso logistic regression to select the important variables first and then using neural networks to build the relationship between the response \( y \) and eleven selected covariates, an average of 98.7% prediction accuracy out of 1000 bootstrapping samples could be obtained from 8-fold cross-validation and 100% prediction accuracy with leave-one-out cross-validation has been achieved. Considering the high prediction accuracy, it could be concluded that the covariates selected predict the response well. When some bad variables are added in Model 4, it shows that the prediction accuracy will decrease but not dramatically. It could be inferred that when some important covariates are selected having a great influence on the response, the prediction accuracy will slightly decrease because of the noisy variables.

In order to check the robustness of Model 3, four noisy random variables drawn from Normal, Uniform, Gamma and the Student distribution separately are combined with the eleven selected covariates. The response \( y \) is simulated from using the parameters in the neural networks shown in Figure 1 and the input is the eleven variables. It is of interest to check whether the true covariates could be selected among all the covariates including the noisy variables. By conducting stability selection and the Lasso logistic regression, the selected probability of each variable is summarized as below. It could be seen that the selection frequencies of the true covariates are much higher than the noisy variables. But the noisy variable from the Student distribution has a frequency larger than 0.3 which is not desired.
Table 4: Stability selection estimates greater than 0.3 threshold with noisy variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Relative Selection Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 V13</td>
<td>0.782</td>
</tr>
<tr>
<td>2 V45</td>
<td>0.725</td>
</tr>
<tr>
<td>3 V141</td>
<td>0.699</td>
</tr>
<tr>
<td>4 V97</td>
<td>0.690</td>
</tr>
<tr>
<td>5 V138</td>
<td>0.681</td>
</tr>
<tr>
<td>6 V66</td>
<td>0.679</td>
</tr>
<tr>
<td>7 V25</td>
<td>0.615</td>
</tr>
<tr>
<td>8 V96</td>
<td>0.577</td>
</tr>
<tr>
<td>9 V57</td>
<td>0.562</td>
</tr>
<tr>
<td>10 V32</td>
<td>0.526</td>
</tr>
<tr>
<td>11 V97</td>
<td>0.472</td>
</tr>
<tr>
<td>12 Vi</td>
<td>0.334</td>
</tr>
<tr>
<td>13 Vunif</td>
<td>0.079</td>
</tr>
<tr>
<td>14 Vgamma</td>
<td>0.061</td>
</tr>
<tr>
<td>15 Vnorm</td>
<td>0.016</td>
</tr>
</tbody>
</table>

In general, high prediction accuracy could be achieved by using stability selection to select the variables first and then determining the relationship between response variable and the selected covariates by neural networks via resilient backpropagation using the high-dimensional neural planning data. Improvements of this project could be to explore how to select the threshold of selection probability and how to find the false rate of a selected variable. The false rate of selected variables refers to the percentage of variables that are selected in the model but they are noisy variables actually. It is also promising to try stability selection with the randomized Lasso in the future.

References


