IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 25, 2021, accepted July 15, 2021, date of publication July 26, 2021, date of current version August 11, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3099473

Sparse Nonnegative Interaction Models

MIRAI TAKAYANAGI[1], YASUO TABEI[2], EINOSHIN SUZUKI[3], AND HIROTO SAIGO[3]

[1] Department of Information Science and Technology, Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka 819-0395, Japan
[2] Riken Center for Advanced Intelligence Project, Tokyo 103-0027, Japan
[3] Department of Information Science and Technology, Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka 819-0395, Japan

Corresponding author: Hiroto Saigo (hiroto.saigo@gmail.com)

The work of Hiroto Saigo was supported by KAKENHI, Grant-in-Aid for Scientific Research (B), Japan Society for the Promotion of Science (JSPS), under Project 19H04176 and Project 21H01684.

ABSTRACT Non-negative least square regression (NLS) is a constrained least squares problem where the coefficients are restricted to be non-negative. It is useful for modeling non-negative responses such as time measurements, count data, histograms and so on. Existing NLS solvers are designed for cases where the predictor variables and response variables have linear relationships, and do not consider interactions among predictor variables. In this paper, we solve NLS in the complete space of power sets of variables. Such an extension is particularly useful in biology, for modeling genetic associations. Our new algorithms solve NLS problems exactly while decreasing computational burden by using an active set method. The algorithm proceeds in an iterative fashion, such that an optimal interaction term is searched by a branch-and-bound subroutine, and added to the solution set one another. The resulting large search space is efficiently restricted by novel pruning conditions and two kinds of sparsity promoting regularization; l_1 norm and non-negativity constraints. In computational experiments using HIV-1 datasets, 99% of the search space was safely pruned without losing the optimal variables. In mutagenicity datasets, the proposed method could identify long and accurate patterns compared to the original NLS. Codes are available from https://github.com/afiveithree/inlars.

INDEX TERMS LASSO/LARS, itemset mining, nonnegative least squares, variable interaction, interpretable machine learning.

I. INTRODUCTION

Non-negative least square regression (NLS) is a constrained least squares problem where the coefficients are restricted to be non-negative. It is useful for modeling non-negative data such as time measurements, count data, or price. It has been introduced in [14], and since then many algorithms have been developed. NLS is not only famous for its use as a subroutine for solving *non-negative matrix factorization* [11], but also has many other applications. For example in computational chemistry, it is used for estimating concentrations from spectra data, in which the percentage of a composition does not take negative values [2]. Another example in computational biology is mass spectrometry analysis, in which observed spectrum is to be recovered by fitting templates isotope patterns [23]. A review by [3] contains other applications such as text mining and speech recognition, as well as algorithmic solutions.

However, existing methods for NLS assume linear relationships among predictor variables and response variables, and cannot automatically consider high order interactions. As we show later in this paper, there are promising applications in biology and chemistry in which NLS with variable interactions are useful. For example in HIV research, drug resistance of a virus is modeled by weighted sum of known mutations, where weights are sought to be nonnegative, and the accumulation of mutations often triggers severe increase in the drug resistance. The reason that NLS with variable interaction has not been considered before lies in the necessity of enumerating all the variable interactions a-priori, an NP-hard problem.

Frequent pattern discovery, or itemset mining methods such as Apriori [1], FP-growth [8], or Eclat [31], are typically used for enumerating patterns, and enumeration of all the itemsets is equivalent to enumeration of all the combinations of variables in the given dataset. However, using all the resulting patterns as features for machine learning algorithm requires some engineering. For example, [22] employed LASSO regression [27] as a machine learning algorithm and LCM [23] as an enumeration algorithm to consider all the interactions among mutations for predicting the drug resistance of anti-HIV agents. Reference [18] extended this work with a GAP safe screening, and

The associate editor coordinating the review of this manuscript and approving it for publication was Larbi Boubchir.

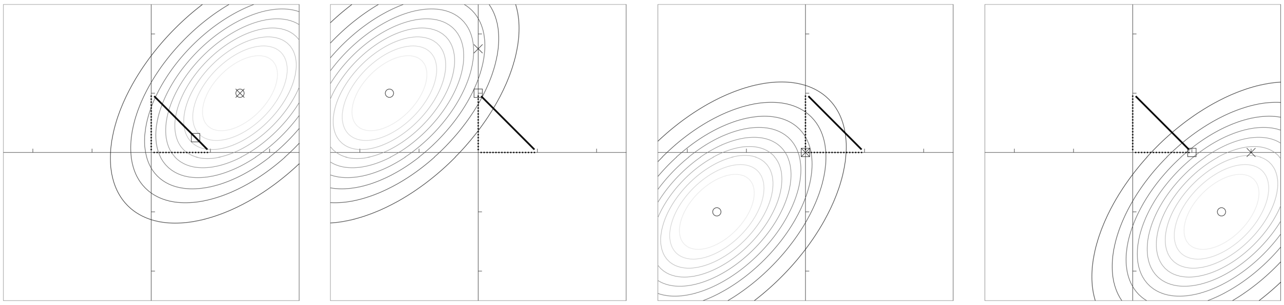


FIGURE 1. Solutions to a 2-dimensional problem by OLS (○), NIM (×) and NIL1 (□). Ellipses are the contours of the least squares error. Four possible cases are displayed depending on the position of the OLS solution. Sparsity is induced if the OLS solution occurs any orthants other than the 1st orthant.

demonstrated that further efficiency can be brought to pattern search. SHIMR [4] extended LASSO to classification by employing double hinge loss, and demonstrated its ability to reject uncertain samples for improving the test set accuracy. In all of the above works, ℓ_1 constraints played an important role in suppressing the number of variables in building prediction models. This paper is positioned in the line of these works, however, this work employs, for the first time, non-negativity constraints for inducing sparsity, whose property in terms of support recovery is studied recently [17] and [23]. Similarly to the previous works, we employ a modified itemset mining algorithm as a subroutine to find variable interactions necessary for explaining the response. So the solution obtained is not different from the one obtained by a two-step approach that enumerates all the variable interactions first, then run NLS solver on it. In feature selection literature [7], the above two-step approach is known as a filter method, and our approach is known as an embedded method. An obvious disadvantage of a filter approach is its necessity for enumerating all the interacting variables that satisfy a certain predefined threshold. In data mining literature, this threshold is known as a *minimum support*, and pattern enumeration with low minimum support is known to be often intractable due to the exponential growth in the number of the resulting output. Nevertheless, using high minimum support is harmful for building an accurate classifier or regressor, since infrequent ones may be crucially important for a given problem. Thus in our solution, we employ an iterative approach similar to boosting, and collect a handful of salient variable interactions one another. Compared to the filter methods, this embedded approach has an advantage in terms of storage, since it does not need to store an intractably large number of variables into memory. For the sake of the search efficiency, we define a tree-based search space, and develop pruning conditions such that ue unnecess parts of teary search space can be pruned. Our pruning conditions make use of the response label attached to each sample, so it is a kind of a supervised approach. Therefore it is much more efficient than the unsupervised filter approach based solely on minimum support. Another important difference of this work from [22] is the efficiency in searching the solution set, which is obtained by additional non-negativity constraints. We demonstrate this fact using simulated data in Section IV.

This paper is based on the ICDM'18 paper [25] which included a solution to the ℓ_1 regularized NLS problem considering variable interactions. However, recent studies revealed that non-negativity constraints alone work as a weak regularization that helps promoting sparsity, so we pursue this direction in this extended version. In this paper, we introduce two kinds of nonnegative least square models considering interactions;

1) NIM (Nonnegative Interaction Model), that makes use of only non-negative constraints, whose solution is found by a cutting plane method [16]. If the size of the interaction is limited to 1, the solution is identical to that of NLS.
2) NIL1 (Nonnegative Interaction model with $\ell 1$ regularization), that makes use of both ℓ_1 and non-negative constraints. For efficiently searching the best regularization parameter, least angle regression [6] is employed. Note that NIL1 itself is already described in [25], but we demonstrate additional experimental results.

In order to understand the differences among OLS, NIM and NIL1, we illustrate their solutions in a two dimensional toy problem (Figure 1). Depending on the location of the OLS solution, four different scenarios are possible, and displayed from left to right. They are categorized into one of the following three cases.

1) OLS solution occurs in the 1st orthant. NIM solution coincides with it, but NIL1 solution may shrink more to the origin depending on the size of the triangle. Sparsity is not induced in this case. (The leftmost plot in Figure 1)
2) OLS solution occurs in the 2nd or 4th orthant, then sparsity is induced since NIM solution typically occurs on either x or y axis where the ellipsoid hits. NIM solution and NIL1 solution do not coincide, but NIL1 solution may shrink more towards zero depending on the size of the triangle, or the strength of the regularization. (The 2nd leftmost plot and the rightmost plot in Figure 1, respectively)

3) OLS solution occurs in the 3rd orthant. NIM solution and NIL1 solution coincide, and the maximum sparsity can be obtained. (The 3rd leftmost plot in Figure 1)

The solutions of NIM and NIL1 depend not only on the position of the OLS solution, but also on the shape of the error contour around the OLS solution .[1] In practical situation, in which we have not only two but many more variables, cases 1 and 3 rarely occur and the case 2 dominates, since it is unlikely that all the OLS coefficients take the same signs.

In practice, the benefit of NIM resides in its simplicity. Since it does not need any regularization parameter to tune, it is easy to use for practitioners [23]. On the other hand, the benefit of NIL1 is its ability to carefully tune the regularization parameter. We demonstrate in the experiments that NIM finds smaller solution sets, but NIL1 performs better due to its ability to control the size of the solution sets.

This paper is organized as follows. In Section II, we introduce notations and settings used throughout the paper. Section III describes all the algorithms for the proposed methods. Section IV describes experimental settings and results based on simulation datasets. We demonstrate the efficiency obtained by pruning conditions, and highlight the difference made by ℓ_1 regularization. In Section V, we demonstrate the usefulness of the proposed methods in HIV-1 datasets and a mutagenicity dataset. Section VII concludes the paper with discussion.

II. PRELIMINARIES

We work on training data with n samples, each sample consisting of up to D items and the corresponding label, which is represented formally as $\{(z_1, y_1), (z_2, y_2), \dots (z_n, y_n)\}$, where $y \in \mathbb{R}^+$, and $z \in \{0, 1\}^D$. We also use compactly represented design matrix $\boldsymbol{Z} = \{z_1, z_2, \dots, z_n\}^\top$, as well as full design matrix $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_n\}^\top$, in which presence or absence of an itemset is represented as;

$$x_{i,t} = I(t \subseteq z_i), \quad \forall t \in \mathcal{T}, \tag{1}$$

where $I(.)$ is an indicator function that returns 1 if an itemset (a set of items) t is included in sample $\boldsymbol{x}$, otherwise 0, and $\mathcal{T}$ be the set of all the frequent items appearing in at least one of given samples. Let $p = |\mathcal{T}| = 2^D - 1$ be the size of the combinatorial feature space, then $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ is a binary matrix that contains all the interaction terms. Due to its size, preparation and storage of $\boldsymbol{X}$ is not only inefficient, but often intractable in practice. So in our proposed algorithm, we work with $\boldsymbol{Z}$, and only necessary part of $\boldsymbol{X}$ is dynamically accessed and stored into memory.

A. SPARSE NON-NEGATIVE LEAST SQUARES

Our model is a linear regression model learnable from $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots (\boldsymbol{x}_n, y_n)\}$, and can be represented as;

$$f(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{\beta},$$

where $\boldsymbol{\beta}$ are regression coefficients. Below we incorporate non-negative constraints to coefficient vector $\boldsymbol{\beta}$, and attempt to predict response vector $\boldsymbol{y}$ in a least squares sense. The resulting non-negative least squares (NLS) problem is;

$$min_{\boldsymbol{\beta} \geq 0} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2, \tag{2}$$

where $\boldsymbol{\beta} \succeq 0$ stands for non-negativity constraints for a vector $\boldsymbol{\beta}$. Introduction of ℓ_2 penalty results in non-negative ridge regression, and that of ℓ_1 penalty results in ℓ_1 regularized non-negative least squares (ℓ_1NLS) or non-negative LASSO;

$$min_{\boldsymbol{\beta} \geq 0} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1, \tag{3}$$

where λ is a regularization parameter, $\boldsymbol{\beta} \in \mathbb{R}^p$ is a sparse coefficient vector whose entries are mostly zeros due to ℓ_1 penalty [9]. The λ that performs best in a given dataset depends on the data, and one has to use an adaptive method such as cross-validation with grid search [9]. However, in our case, we have to run expensive enumeration algorithm as many as the candidate λs. This task is not only inefficient, but also may become intractable for large number of λs. Thus we make advantage of regularization path tracking algorithm, and attempt to reduce computational burden in the enumeration algorithm as much as possible. It can be realized by jumping from the current λ to the next λ, and reusing the already found features in identifying the next feature [6].

B. PATH TRACKING ALGORITHM

Reference [21] has shown that ℓ_1 regularized regression with quadratic loss function has piecewise linear solution paths, and can be efficiently computed with the same time complexity as that of ordinal least squares. Let us decouple loss and penalty in equation (3), and define $L(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{\beta}) = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$ and $J(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1$. Our goal is to show coefficient profile

$$\hat{\boldsymbol{\beta}}(\lambda) = \underset{\boldsymbol{\beta} \geq 0}{arg\ min}\, L(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{\beta}) + \lambda J(\boldsymbol{\beta}), \tag{4}$$

along the λs in sequence.

This problem can be solved efficiently only when $\hat{\boldsymbol{\beta}}(\lambda)$ has piecewise linearity [21], namely that for a finite set of ordered λs such that $\lambda_0 < \lambda_k \dots < \lambda_\infty$, $\hat{\boldsymbol{\beta}}(\lambda) = \hat{\boldsymbol{\beta}}(\lambda_k) + (\lambda - \lambda_k)\gamma_k$ holds for $\lambda_k \leq \lambda \leq \lambda_{k+1}$, where $\gamma_k \in \mathbb{R}^p$ is a direction of the path at λ_k, and is known as an *equiangular vector*. Proposition 1 in [21] shows that the gradient of equation (4) is given as

$$\frac{\partial \hat{\boldsymbol{\beta}}(\lambda)}{\partial \lambda} = -\left[\nabla^2 L(\hat{\boldsymbol{\beta}}(\lambda)) + \lambda \nabla^2 J(\hat{\boldsymbol{\beta}}(\lambda))\right]^{-1} \nabla J(\hat{\boldsymbol{\beta}}(\lambda)), \tag{5}$$

by using Taylor's expansion. As we will see in this section, nonnegative LASSO (3) satisfies the sufficient conditions for the coefficient solution path to be piecewise linear, so it is possible to keep track of it.

More formally, the Lagrange primal function of nonnegative LASSO (3) can be written as

$$\sum_i L(\beta) + \lambda \sum_j \beta_j - \sum_j \delta_j \beta_j, \tag{6}$$

[1] For example in the 3rd orthant, if the contour is inclined more towards $y > 0$, then the NIM solution can change from $(0, 0)$ to $(c, 0)$, where $c > 0$.

where $\delta_j \geq 0$ is a Lagrange multiplier, and the dependence of L on $\boldsymbol{X}$ and $\boldsymbol{y}$ are omitted for simplicity. From the Karush-Kuhn-Tucker (KKT) condition, we have

$$\nabla L(\beta)_j + \lambda - \delta_j = 0 \tag{7}$$

$$\delta_j \beta_j = 0 \tag{8}$$

Since both β and λ take either zero or positive values, we can derive the following conditions;

$$\begin{cases} \beta_j = 0, \lambda = 0 & \to (\nabla L(\beta))_j = 0, \\ \beta_j > 0, \lambda = 0 & \to (\nabla L(\beta))_j = 0, \\ \beta_j = 0, \lambda > 0 & \to \delta_j = (\nabla L(\beta))_j + \lambda > 0, \\ \beta_j > 0, \lambda > 0 & \to (\nabla L(\beta))_j = -\lambda < 0. \end{cases} \tag{9}$$

If $\lambda = 0$, it corresponds to an unregularized case, and a solution can be obtained by a standard active set method [14]. If $\lambda > 0$, active variables and inactive variables have different conditions. Let a set of active variables be $\mathcal{A} = \{j : \hat{\beta}_j(\lambda) \neq 0\}$, then the direction of regularization path at some $\lambda(> 0)$ is calculated as

$$\frac{\partial \hat{\boldsymbol{\beta}}(\lambda)_{\mathcal{A}}}{\partial \lambda} = -(\boldsymbol{X}_{\mathcal{A}}^{\top} \boldsymbol{X}_{\mathcal{A}})^{-1} \mathbf{1}_{\mathcal{A}} \left(= \boldsymbol{\gamma}_{\mathcal{A}}\right), \tag{10}$$

where $\boldsymbol{X}_{\mathcal{A}}$ and $\mathbf{1}_{\mathcal{A}}$ denotes a design matrix and a vector of ones, restricted to an active set $\mathcal{A}$, respectively. Our update rule is represented as an affine combination of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, such that

$$\boldsymbol{\beta}_j^* \leftarrow (1 - \tau)\boldsymbol{\beta}_j + \tau \boldsymbol{\gamma}_j, \tag{11}$$

where $0 < \tau < 1$ is a step length. Note that setting $\tau = 0$ corresponds to staying at the current active set, so $\nabla L(\boldsymbol{\beta}) = -\lambda$ holds. On the other hand, setting $\tau = 1$ corresponds to taking a full step towards the least squares solution so that $\nabla L(\boldsymbol{\gamma}) = 0$ holds. Since equation(11) is a linear function in its feasible region, we can track it. Along the path, one of the following two event can occur;

- Deleting a variable: In this case, an active variable becomes inactive, thereby one of $\boldsymbol{\beta}_j^*$ reaches zero. The step length until this event is obtained by solving $((1 - \tau)\boldsymbol{\beta} + \tau \boldsymbol{\gamma})_j = 0$ for τ;
$$\tau = \left(\frac{\beta_j}{\beta_j - \gamma_j} \right). \tag{12}$$
- Adding a variable: In this case, an inactive variable becomes active. Since $\nabla L(\boldsymbol{\beta}^*)$ is given as an affine combination of $\nabla L(\boldsymbol{\beta})$ and $\nabla L(\boldsymbol{\gamma})$, it satisfies
$$\nabla L(\boldsymbol{\beta}^*) = (1 - \tau)\nabla L(\boldsymbol{\beta}) + \tau \nabla L(\boldsymbol{\gamma}).$$
By plugging $\nabla L(\boldsymbol{\beta}) = -\lambda$ and $\nabla L(\boldsymbol{\gamma}) = 0$ into the above equation, the step length until this event is obtained as
$$\tau = \frac{\lambda + (\nabla L(\boldsymbol{\beta}))_j}{\lambda + (\nabla L(\boldsymbol{\beta}))_j - (\nabla L(\boldsymbol{\gamma}))_j}. \tag{13}$$

Step lengths of both events are recorded, and a smaller one is chosen. After the occurrence of an event, the direction of the soln path (10) needs to be recalculated.

Algorithm 1 Entire Regularization Path for Nonnegative Interaction Model (NIL1)

```
1:  $\boldsymbol{\beta} = \mathbf{0}, \mathcal{A} = \underset{i}{arg\ max}\, |\nabla L(\boldsymbol{\beta})|_j, \quad j \notin \mathcal{A}$    ▷ Search Problem 2
2:  $\lambda = |\nabla L(\boldsymbol{\beta})|_{\mathcal{A}}, \boldsymbol{\gamma}_{\mathcal{A}} = \mathbf{1}, \boldsymbol{\gamma}_{j \notin \mathcal{A}} = 0$
3:  while $\lambda > 0$ do
4:      $\tau_1 = \left( \frac{\beta_j}{\beta_j - \gamma_j} \right), \quad j \in \mathcal{A}$
5:      $\tau_2 = \frac{\lambda + (\nabla L(\beta))_j}{\lambda + (\nabla L(\beta))_j - (\nabla L(\gamma))_j}, \quad j \notin \mathcal{A}$    ▷ Search Problem 1
6:      step length $\tau = min\{\tau_1, \tau_2\}$
7:      if $\tau = \tau_1$ then remove the variable from active set $\mathcal{A}$.
8:      end if
9:      if $\tau = \tau_2$ then add the variable to active set $\mathcal{A}$.
10:     end if
11:     $\boldsymbol{\beta} \leftarrow (1 - \tau)\boldsymbol{\beta} + \tau \boldsymbol{\gamma}$
12:     $\nabla L(\boldsymbol{\beta}) \leftarrow (1 - \tau)\nabla L(\boldsymbol{\beta}) + \tau \nabla L(\boldsymbol{\gamma})$
13:     $\lambda \leftarrow (1 - \tau)\lambda$
14:     Recalculate new direction $\boldsymbol{\gamma}_{\mathcal{A}} = -(\boldsymbol{X}_{\mathcal{A}}^{\top} \boldsymbol{X}_{\mathcal{A}})^{-1} \mathbf{1}_{\mathcal{A}}$
15:         $\boldsymbol{\gamma}_{j \notin \mathcal{A}} = 0$
16: end while
```

An interesting extreme case is when λ is largest. In such a case, $\boldsymbol{\beta} = \mathbf{0}$ becomes the solution to equation (3), and from KKT condition, such λ is found as

$$\lambda_{max} = min - (\nabla L(\mathbf{0}))_j = max(\boldsymbol{X}^{\top} \boldsymbol{y})_j. \tag{14}$$

So we start by searching the largest λ, then gradually decrease it along the regularization path. The whole algorithm is shown in Algorithm 1.

III. METHODS

In applying Algorithm 1 to full design matrix $\boldsymbol{X}$, an obvious but an inefficient way is to extract $\boldsymbol{X}$ from training dataset $\boldsymbol{Z}$ in the first place, then run a path-tracking algorithm on it. However, with respect to the increase in the number of items D or the number of samples n, it quickly becomes intractable. So we devise to extract the necessary part of $\boldsymbol{X}$ dynamically from $\mathbf{Z}$. An intuition behind this idea is that, due to the sparsity induced by ℓ_1 norm, the number of active variables should be much smaller than that of non-active variables, which enables us to keep only active variables. Therefore, we focus on lines 2 and 6 of Algorithm 1, where a switch of variable from inactive set to active set occurs. Below, we call them Search Problem 1 and 2, and propose efficient branch and bound algorithms for both of them.

As a canonical search space, we employ the one defined by closed itemset mining (CIM) algorithm, LCM [29]. Given transactions consisting of itemsets, CIM can enumerate frequently appearing itemsets in the database while ignoring their subsets with the same frequency. This idea comprises *closed itemset* rather than frequent itemset, and helps reducing the size of the output as well as improving the interpretability. An exemplar canonical search space of CIM is illustrated in Figure 2. In a canonical search space spanned

Algorithm 2 Search Problem 1

```
procedure
    $\tau = \infty$
    for all $t$ consisting of single item do project(t)
    end for
    return $\tau$
end procedure
function Project(t)
    if pruning condition (15) holds then
        return
    end if
    Calculate $\tau_{cur} = \frac{\lambda+(\nabla L(\beta))_j}{\lambda+(\nabla L(\beta))_j-(\nabla L(\ \gamma))_j}$
    if $\tau < \tau_{cur}$ and $\tau_{cur} \neq 0$ then
        $\tau = \tau_{cur}$
    end if
    for all child $t' \in t$ do
        project($t'$)
    end for
end function
```

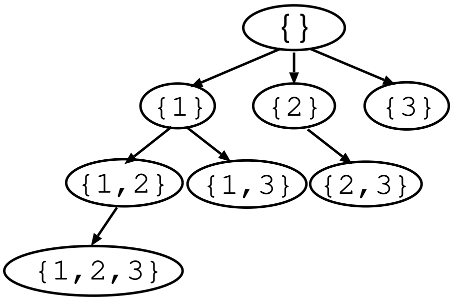


FIGURE 2. Exemplar search space for items {1,2,3}. It starts with the empty set, then traverses all the combination of items without duplication.

by CIM algorithm, one can reach a node corresponding to combination of items (itemset) without visiting the same node again. However, the number of nodes grows exponentially to the increase in the number of items, therefore tree pruning is of crucially importance. Below, we propose efficient bounding conditions using target label information attached to each sample.

A. SEARCH PROBLEM 1

We show the pseudocode for solving Search Problem 1 in Algorithm 2, which searches for the minimum step length τ. Suppose that we have reached a node t in a search tree, and the minimum step length found so far be τ^*. If we can guarantee that there exists no smaller step length in the downstream nodes of t, then we can safely prune the rest of the tree without losing the optimal pattern. We introduce two such pruning conditions for the Search Problem 1 below.

Theorem 1: Let $(\nabla L(\boldsymbol{\beta}))_t = \sum_{i=1}^{n} X_{ti}C_{it}$ *where* $C_{it} = X_{it}\beta_t - y_i$. *If the following condition is satisfied,*

$$\sum_{\{i|C_{it}\geq 0\}} X_{ti}C_{it} + \lambda < 0, \qquad (15)$$

then there exists no solution in the downstream of the node t, so we can safely prune the rest of the tree without losing the optimal pattern.

Proof: Let t' be a child node of t such that $t \subseteq t'$ holds, then

$$\begin{aligned}(\nabla L(\boldsymbol{\beta}))_{t'} &= \sum_{\{i|C_{it'}\geq 0\}} X_{t'i}C_{it'} + \sum_{\{i|C_{it'}\leq 0\}} X_{t'i}C_{it'} \\ &\leq \sum_{\{i|C_{it'}\geq 0\}} X_{t'i}C_{it'} \\ &\leq \sum_{\{i|C_{it}\geq 0\}} X_{ti}C_{it}\end{aligned}$$

The second inequality holds due to the fact that $\{i \mid C_{it'} \geq 0\} \subseteq \{i \mid C_{it} \geq 0\}$. On the other hand, from KKT condition, $(\nabla L(\boldsymbol{\beta}))_t + \lambda > 0$ must hold for any $(\nabla L(\boldsymbol{\beta}))_t$. Therefore if $\sum_{\{i|C_{it}\geq 0\}} X_{ti}C_{it} + \lambda < 0$ is guaranteed at the current node t, then there is no need to search for downstream nodes that do not satisfy KKT condition. □

The computational complexity for computing the above pruning condition is $O(nk)$, where k is the size of the active patterns, which is generally very small thanks to the sparsity induced by ℓ_1 norm. Similarly to the first pruning condition, we propose the second pruning condition.

Theorem 2: If the following condition is satisfied,

$$\sum_{\{i|C_{it}\leq 0\}} X_{ti}C_{it} = 0,$$

then there exists no solution in the downstream of the node t, so we can safely prune the rest of the tree without losing the optimal pattern.

Proof: Let t' be a child node of t such that $t \subseteq t'$ holds, then

$$\begin{aligned}(\nabla L(\boldsymbol{\gamma}))_{t'} &= \sum_{\{i|C_{it'}\geq 0\}} X_{t'i}C_{it'} + \sum_{\{i|C_{it'}\leq 0\}} X_{t'i}C_{it'} \\ &\geq \sum_{\{i|C_{it'}\geq 0\}} X_{t'i}C_{it'} + \sum_{\{i|C_{it}\leq 0\}} X_{ti}C_{it}\end{aligned}$$

The inequality holds due to the fact that $\{i \mid C_{it'} \leq 0\} \subseteq \{i \mid C_{it} \leq 0\}$. If $\sum_{\{i|C_{it}\leq 0\}} X_{ti}C_{it} = 0$ holds, then $(\nabla L(\boldsymbol{\gamma}))_{t'} \geq 0$, which violates the non-positiveness condition on $\nabla L(\boldsymbol{\beta})$ (equations (9)). □

B. SEARCH PROBLEM 2

It is called at line 2 in Algorithm 1. It is a maximization problem, but its procedure is similar to Algorithm 2, so we omit it. What we search for is a node that maximizes the following criterion.

$$max\left(X^{\top}\boldsymbol{y}\right)_j. \qquad (16)$$

Suppose that we have reached a node t, the corresponding objective value be $(\nabla L(\boldsymbol{\beta}))_t = \sum_{i=1}^{n} X_{it}y_i$, and the maximum value found so far be $(\nabla L(\boldsymbol{\beta}))_t^*$. If we can guarantee that there exists no larger objective value in the downstream of nodes of t, then we can safely prune the rest of the tree without losing the optimal pattern.

Theorem 3: [12] If the following condition is satisfied,

$$(\nabla L(\boldsymbol{\beta}))_t^* \geq \sum_{i=1}^{n} X_{it} y_i \tag{17}$$

then there exists no greater $(\nabla L(\boldsymbol{\beta}))_t$ in the downstream of the node t, so we can safely prune the rest of the tree without losing the optimal pattern.

For the proof, please refer to [12].

C. NON-NEGATIVE INTERACTION MODEL WITHOUT ℓ_1 CONSTRAINTS

For solving NLS problem with variable interactions, we revisit equation (2). For efficiently solving the problem, we introduce a cutting plane approach below. The problem can be rewritten as the following optimization problem;

$$\min_{\boldsymbol{\beta},\boldsymbol{\xi}} \frac{1}{2}\sum_{i=1}^{n} \xi_i^2 \tag{18}$$

$$\text{s.t.} \sum_t X_{it}\beta_t - y_i \leq \xi_i, \quad i = 1, \ldots, n \tag{19}$$

$$\sum_t y_i - X_{it}\beta_t - y_i \leq \xi_i, \quad i = 1, \ldots, n$$

$$\boldsymbol{\beta} \geq \mathbf{0}, \quad \boldsymbol{\xi} \geq \mathbf{0}, \tag{20}$$

where $\boldsymbol{\xi}$ are slack variables. The above program has p variables and n constraints. Since p can be extremely large due to the potential number of interactions, directly solving the primal problem is hard. So we derive the dual problem. From the KKT condition, we have

$$\frac{\partial L}{\partial \beta_t} = -\sum_t (w_i^+ - w_i^-)X_{it} - \gamma_t = 0, \tag{21}$$

$$\frac{\partial L}{\partial \xi_i} = \xi_i - w_i^+ - w_i^- - \delta_t = 0, \tag{22}$$

where δ is a slack variable. They can be rewritten as

$$-\sum_i (w_i^+ - w_i^-)X_{it} \geq 0, \quad \forall t \in \mathcal{T} \tag{23}$$

$$\xi_i \geq w_i^+ - w_i^- \tag{24}$$

Substituting them back to the primal problem in its Lagrange form, the dual problem is obtained as;

$$\min_{\boldsymbol{w}} \frac{1}{2}\sum_{i=1}^{n}(z_i^+ + z_i^-)^2 - \sum_{i=1}^{n} y_i(z_i^+ - z_i^-) \tag{25}$$

$$\text{s.t.} \sum_{i=1}^{n}(z_i^+ - z_i^-)X_{it} \leq 1, \forall t \in \mathcal{T} \tag{26}$$

$$\sum_{i=1}^{n}(z_i^+ - z_i^-) = 0, \tag{27}$$

$$\boldsymbol{w}^+ \geq \mathbf{0}, \quad \boldsymbol{w}^- \geq \mathbf{0}, \tag{28}$$

where $\boldsymbol{w}^+$ and $\boldsymbol{w}^-$ are Lagrange multipliers. After solving the dual problem, primal solution vector $\boldsymbol{\beta}$ can be recovered from the Lagrange multipliers of the dual problem. The dual problem has a large number of constraints, but we employ a cutting plane method [16] to handle them. In principle, it finds the initial solution by ignoring all the constraints, and repeat adding a constraint one by one to the dual problem until the stopping criterion is satisfied. The constraint to be added is called the most violated constraint, and can be found by solving the following search problem;

Algorithm 3 Cutting Plane Method Applied to the Dual of the Non-Negative Least Squares

```
$\mathcal{A} = \emptyset$
loop
    $j = arg\ max\ |\nabla L(\boldsymbol{\beta})|_j, \quad j \notin \mathcal{A}$ ▷ search subproblem
    if $\left|\sum_{i=1}^{n} X_{ij} w_i\right| < 1$ then break;
    end if
    Add $j$ to the active set $\mathcal{A}$
    $\boldsymbol{w}$ ← solve the dual problem (equations (25),(26),(27) and (28))
end loop
```

$$\max_t \left|\sum_{i=1}^{n} X_{it} w_i\right|. \tag{29}$$

In our case X is a binary matrix, and this problem involves *integer programming*. However, it is equivalent to equation (16), thus we can receive the benefit from Theorem 3; an efficient tree pruning condition in the space of power set of variables. To summarize, we iteratively solve the quadratic problem (equations (25),(26),(27) and (28)) with a limited number of constraints, and the solution vector $\boldsymbol{w}$ is used in a subproblem (29) to search for the next constraint. The resulting algorithm is shown in Algorithm 3.

Since the primal problem is convex and so is its dual, we can obtain the global solution by repeating the loop. Moreover, the relative distance to the optimal solution can be measured using the *duality gap*.

IV. SIMULATION STUDY

In this section, we demonstrate the effectiveness of the proposed pruning conditions on simulated datasets. Simulated datasets are generated as follows. First we draw a random variable from Bernoulli distribution ($q = 0.6$), then generated design matrices $X \in \{0, 1\}^{n \times p}$ of varying sizes. Then we randomly selected 5 columns in the design matrix, and further selected 5 features out of their 2^5 combinations. The true coefficients $\boldsymbol{w}$ are drawn from uniform distribution $\mathcal{U}_{[0,1]}$, and the response vector is built as $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w}$. All the computation time are measured on a 64-bit machine with Intel(R) Xeon(R) E5-2697 Processor 2.70GHz.

In Figure 3, we compare the computation time with/without the proposed pruning conditions. For this experiment, we fixed the number of items to 50, and varied the sample size $n \in \{50, 100, 200, 500, 1000\}$. Note that setting the number of items D to 50 corresponds to set-

TABLE 1. Computation time of NIL1 in seconds in large datasets. The digit in parentheses shows the number of traversed nodes.

# of samples \ # of Items	10	100	1,000	10,000	100,000	1,000,000
10	0.00412 (100)	0.0826 (335)	4.27 (862)	900 (7013)	15,920 (9923)	21,100 (1155)
100	0.224 (189)	7.22 (9120)	8,320 (469,016)	> 24 hours	> 24 hours	> 24 hours
1,000	0.254 (432)	76.5 (11287)	> 24 hours	> 24 hours	> 24 hours	> 24 hours
10,000	3.54 (624)	71.7 (7772)	> 24 hours	> 24 hours	> 24 hours	> 24 hours
100,000	31.2 (860)	1974 (7721)	> 24 hours	> 24 hours	> 24 hours	> 24 hours
1,000,000	762 (1592)	25,700 (12914)	> 24 hours	> 24 hours	> 24 hours	> 24 hours

ting the maximum size of the search space to 2^{50}. Therefore, the size of the search tree and the time for traversing the entire space grows exponentially. Indeed, in Figure 3, a naïve approach did not finish within 24 hours when n is set to more than 200. In contrast, with our pruning conditions, the proposed search method could successfully control the increase in computation time low as long as n is set to 1000.

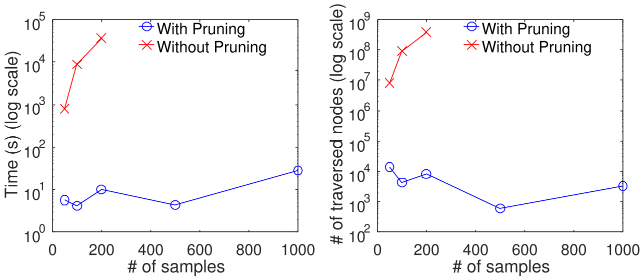


FIGURE 3. Efficiency obtained by tree pruning. (left) Computation time in seconds are shown in terms of the sample size while fixing the number of items to 50. (right) Similarly, the number of traversed nodes is shown in terms of the sample size.

A similar observation is obtained when we fixed the number of samples to 50, and chose the number of items D from $\{20, 50, 100\}$, corresponding to $p = \{2^{20}, 2^{50}, 2^{100}\}$. In Figure 4, we can see that the size of the search space of a naïve approach increases rapidly, so is the time to traverse the space. On the other hand, with the proposed pruning condition, the size of the search space is restricted to 0.01 % of the original one, and the corresponding search time turns into more than a thousand times faster, when we set the number of items to 100. For investigating the scalability of the proposed approach, we have run NIL1 in simulated datasets with lager D and n. The result appears in Table 1. It turns out that NIL1 scales to even $n = 1,000,000$ or $D = 1,000,100$ when either one of them is kept small, but does not scale when both n and D are larger than 1000. If we consider a case when n is small and D is large, it is likely that we have not observed all the important features, so this case is hard to solve in reality. On the other hand, when n is large and D is small, it is likely that we have observed all the necessary features. NIL1 could successfully handle $n = 1,000,000$ samples when the number of items is 100.

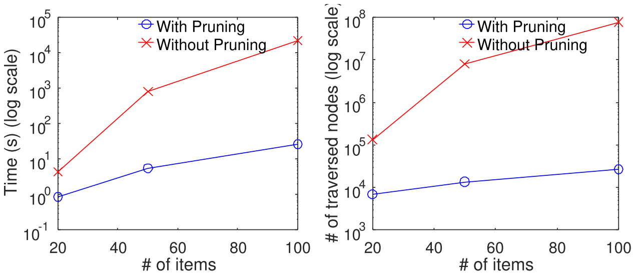


FIGURE 4. Efficiency obtained by tree pruning. (left) Computation time in seconds is shown in terms of the sample size while fixing the sample size to 50. (right) Similarly, the number of traversed nodes is shown in terms of the number of items.

In order to further understand the nature of the proposed method, we compare the proposed method with itemset LAR/LASSO. For this purpose, we implemented itemset LAR/LASSO using exactly the same procedure as *gLARS* [28], except that the pattern mining subroutine is replaced from graph mining to itemset mining. First, we generated a dataset of 200 samples and 100 items, and compared the number of iterations until reaching the peak of the learning curve in terms of the validation set accuracy. NIL1 took 23 seconds until reaching its best (10-th iteration), whereas it took 100,000 seconds for itemset LAR/LASSO to reach its best (95-th iteration). It suggests that the size of the search space is smaller for the proposed method than that of itemset LAR/LASSO. In order to verify this tendency, we have measured the number of traversed nodes in the search tree until showing the full regularization path, and corresponding computational time while varying the maximum pattern size. Figure 5 compares the number of nodes visited for the proposed method and that for itemset LAR/LASSO. In the figure, we can observe that the size of the search space and the resulting time consumption were more than a thousand times smaller for the proposed method compared to itemset LAR/LASSO. This efficiency resorts to non-negative constraints.

V. REAL-WORLD DATA EXPERIMENTS

In this section, we demonstrate the performance of the proposed methods in two applications. The first one is HIV-1 drug resistance prediction problem, and the other is mutagenicity prediction problem.

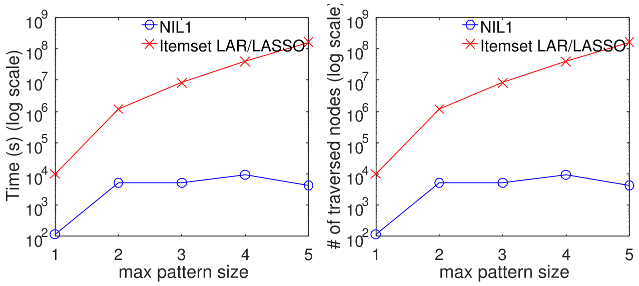


FIGURE 5. Efficiency comparison of NIL1 with itemset LAR/LASSO. (left) Computation time in seconds is shown in terms of the maximum pattern size while fixing the sample size to 200 and the number of items to 100. (right) Similarly, the number of traversed nodes is shown in terms of the maximum pattern size.

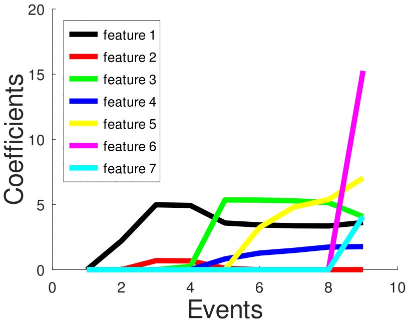


FIGURE 6. Coefficient profile in HIV-1 AZT dataset, as a function of the number of learning events. A non-negative weight is given to each feature, and each feature is represented by a curve with an unique color.

TABLE 2. Comparison of computational efficiency with or without tree pruning.

	With Pruning		Without Pruning	
max. pat. size	pruning rate	time (s)	pruning rate	time (s)
1	36.6%	6.57	0%	11.37
2	96.8%	14.97	0%	90.69
3	98.2%	23.06	0%	726.81
4	98.7%	29.47	0%	422.89
5	98.7%	27.98	0%	340.56
6	99.0%	24.26	0%	383.66
7	99.0%	26.67	0%	409.56
8	99.2%	23.12	0%	424.05

A. HIV-1 DRUG RESISTANCE PREDICTION

In HIV-1, treatments to the patients resort to subscribing drugs for decreasing the number of viral copies in blood. However, some strains of the virus copies survive by obtaining drug-resistant mutations in its genome. These drug-resistant strains then dominate the population while non-resistant strains are eliminated by drugs. Clinicians and statisticians then desire to decompose experimentally measured drug resistance into a weighted sum of resistance by each mutation, which is traditionally modeled by linear regression [20]. Considering the biological selection mechanism, however, these weights never take negative values. Since mutations with negative weights suggest increased susceptibility to drugs, then they do not survive the biological selection. In previous works [18], [22], negative regression coefficients are observed, but they could be artifacts due to this reason. The NLS approach to this dataset is therefore considered to be more relevant.

The dataset we use is a collection of results of *in vitro* susceptibility tests to available drugs in market, in which the level of resistance is recorded in fold change compared to that of wild type [20]. This record comprises our target response vector $\boldsymbol{y}$, and the genotypes are recorded as the difference from the wild type sequence. For example, if an isolate x has two mutations at the first position and the sixth position in the reference sequence that turn original amino acids into Arginin and Cystein, respectively, then the genotype is recorded in a set representation as $\{1A, 6C\}$.

Figure 6 shows the transition of coefficient $\boldsymbol{\beta}$ as the iteration proceeds. We can observe that seven different variables once moved into the active set, but one variable is removed at the fifth event. We can also observe that all the regression coefficients are non-negative. Figure 7 shows the learning curves by NIL1 and NIFS in three HIV datasets. Errors in terms of the residual sum of squares $\left(RSS : \sum_{i=1}^{n}(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}).^2\right)$ are displayed for both the training set and the validation set. We can observe that the error curves for the training set keeps decreasing as a function of the numbers of steps by NIL1. On the other hand, the error curve for the validation set stops decreasing, and starts increasing at some point. The error curve of NIM in the training set and the validation set are displayed as the horizontal flat lines, since they have no parameter to control regularization. The error curve of the validation set by NIL1 almost always lies lower than that of NIM, suggesting its generally better performance in HIV-1 datasets. Figure 8 displays the distributions of the found patterns by NIL1 and NIM. It is observed that NIL1 resulted in much larger solution sets than that by NIM. The same tendency was observed in five other datasets as well.

In Table 2, we demonstrate the efficiency obtained by tree prunings. We compare the computation time of the proposed method with that of the naïve counterpart without pruning. In addition to the pruning introduced in Section III, we employ the maximum pattern size (the number of items in an itemset) as an additional pruning condition for illustration purpose. We can observe in Table 2 that the proposed method has successfully pruned more than 90% of the search space spanned by the naïve method. From Table 2, we can also observe that roughly 90% of the computation time is saved thanks to the tree pruning.

Lastly, we demonstrate the validity of non-negativity constraints in HIV-1 dataset. As it has already been discussed in Section II, the difference between NIL1 and itemset LAR/LASSO lies just in the fact that whether non-negative constraints are enforced. Too see this, we ran both methods in HIV-1 AZT dataset, and compared the features obtained until the 10-th iteration. Table 3 shows the features obtained by NIL1 and that by itemset LAR/LASSO

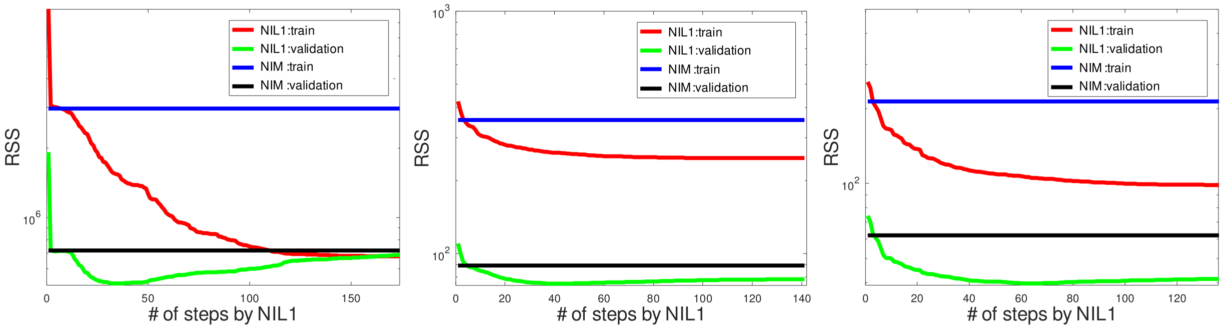


FIGURE 7. Learning curves in HIV-1 AZT (left), ABC (center), D4T (right) datasets. The error of NIL1 in the training set decreases monotonically as the number of steps increases, while the error in the validation set stops decreasing, and start increasing at some point. The error of NIFS in the training set and the validation set are shown as horizontal flat lines, since they have no parameter to control regularization.

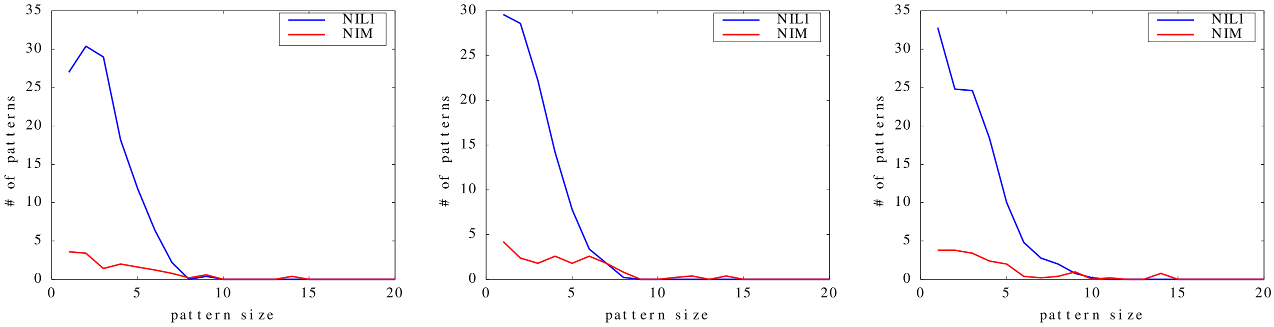


FIGURE 8. Pattern size distribution in HIV-1 AZT (left), ABC (middle) and D4T (right) datasets.

TABLE 3. Features selected by NIL1 and itemset LAR/LASSO until the tenth iteration. In itemset LAR/LASSO, features "139R" and "139R, 173E" receives negative coefficients, but they are suspected to be artifacts since mutants with increased susceptibility to drugs do not survive the biological selection.

	NIL1		itemset LAR/LASSO	
Absolute rank	Coefficient	Itemset	Coefficient	Itemset
1	108	60I	127	170L, 173E
2	101	170L, 173E	-85.1	139R
3	40.2	54I	71.2	60I, 173E
4	26.6	173E	45.7	60I
5	23.7	170L	-32.9	139R, 173E
6	15.5	101Q	22.9	39K, 54I
7	11.2	39N, 170P	5.49	39N, 170P
8	9.50	39K	3.34	54I
9	7.93	39K, 60I	3.29	39K
10	0.15	60I, 173E	0.61	39K, 60I

as well as their coefficients and the ranking in terms of absolute values. Both of the methods identify combinatorial feature $\{170L, 173E\}$ as one of the most influential feature with large positive weights. However, itemset LAR/LASSO also assigns large negative coefficient to $\{139R\}$ as well. However, mutations with large negative weights do not survive biological selection, so this feature is considered as an artifact.

B. MUTAGENICITY PREDICTION

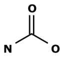
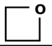
In computational chemistry, small molecules are often represented as *descriptors* and stored in databases together with bioactivity or chemical reactivities. These records can be used for building a statistical model for predicting activity or reactivity of unseen compounds. We use the CPDB mutagenicity dataset [10], which provides mutagenicity classifications (341 mutagens and 343 nonmutagens) as determined by the Salmonella/microsome assay (Ames test). We ran regression methods by making target labels to either mutagen (1.0) or nonmutagen (0.0). In this experiment, we aim at detecting a set of descriptors that trigger mutagenicity as well as building a classifier, so can assume that the regression coefficients take positive values [2] As descriptors, we employed 166 MACCS keys [5] available through Open Babel [19]. We randomly split the dataset into 80% training and 20% testing, and measured the performance in the test set. We repeated this procedure 10 times, and report the averaged statistics in Table 4. We compare NIL1 with NIM. We also ran LARS and NLS as baselines, which do not consider variable interactions. In NIL1, 20% of the training set is reserved as the validation set, and used to determine the regularization parameter. In each experiment, we counted the number of features and the size of each feature, which represents the number of items in it. Rule coverage stands for the ratio of samples represented by at least one feature.

[2] If one is interested in analysis in nonmutagens, then one can swap target values.

TABLE 4. Classification performance in CPDB dataset based on binary fingerprints. The best value in each measure is highlighted in bold font, and the second best one in italic font.

Method	Accuracy	Precision	Recall	F-measure	AUC	# of features	feature size	Rule coverage
LARS	0.671 ± 0.0427	0.687 ± 0.0818	**0.643 ± 0.0847**	0.345 ± 0.674	0.729 ± 0.0320	16.4	1	0.994
NLS	*0.674 ± 0.0427*	*0.785 ± 0.0818*	0.495 ± 0.0847	*0.602 ± 0.0674*	**0.761 ± 0.0376**	12.3	1	0.744
NIM	0.642 ± 0.0254	0.767 ± 0.0754	0.440 ± 0.113	0.544 ± 0.0837	0.612 ± 0.0387	9.94	4.6	0.353
NIL1	**0.691 ± 0.0386**	**0.788 ± 0.0557**	*0.532 ± 0.0802*	**0.631 ± 0.0666**	*0.737 ± 0.0530*	9.32	19.1	0.473

TABLE 5. Representative features found by NLS. In the table, A stands for any atom, Q stands for any atom other than H or C.: stands for an aromatic bond, @ stands for any ring bond, and ! stands for negation.

MACCS ID	Structure	Frequency	Coverage(%)	Accuracy(%)
16	3 molecules ring QAA, such as or	10	2.92	80.0
41	N≡C	10	0.877	83.3
46	Br	10	2.34	81.3
70	$-N-$	10	19.2	90.1
84	NH_2	10	25.1	66.7
133	$N(@)A(!@)A$	10	36.7	72.5
23		8	1.02	71.4
52	NN	6	26.7	77.2
135	$N(:)A(!:)A$	5	33.7	71.2
38	$NC(C)N$	5	43.9	70.0
8	4 molecules ring $QAAA$, such as 	3	0.731	40.0

If we compare the proposed methods with the baselines, we can have an interesting observation in its rule coverage. LARS and NLS showed rule coverage as high as 0.994 and 0.744, respectively. On the other hand, the NIM and NIL1 showed lower coverage such as 0.353 and 0.473, respectively. As we discuss more details below, this is due to the nature of the proposed methods which attempt to search for complex variable interactions not covered by a large number of samples, but have high classification accuracy.

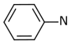
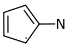
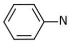
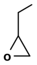
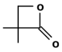
Among the non-negative models, NIL1 performed best in terms of the test set accuracy , which would be due to the consideration of variable interactions unlike LARS and NLS. If we focus on the feature size, NIL1 discovered much larger ones than by NIM. On the other hand, the slightly inferior performance of NIM would be due to the lack of ability to avoid overfitting. As we explain in the next paragraph, large features are important, since they can lead to improved interpretability.

Tables 5 and 6 show representative features selected by NLS and NIL1, respectively. MACCS ID 133 found by NLS has a ring structure with Nitrogen attached to it, and has a large coverage (36.7%) with a moderate accuracy (72.5%). The same structure is captured by NIL1, but with additional structures (Indices 1 and 4 in Table 6.) It is observed that by adding additional structures, the accuracy of this feature boosts to either 77.8% or 78.3% at the cost of smaller coverage. The same observation can be made for the MAACS ID 16 found by NLS. This three molecules hetero cycle structure is detected by NIL1 as well (Index 6 in Table 6, but with an additional carbon structure. Its accuracy is boosted from 80% to 88.2%, at the cost of a smaller coverage. In general, an easy problem can be solved with features with high accuracy and high coverage, but a harder problem requires many rules with low coverage. In this experiment, NIL1 has successfully discovered such rules unlike the baselines. Another interesting observation can be made for MACCS ID 70 in Table 5. The corresponding structure in NIL1 is Index 2 in Table 6. Although the coverage and accuracy do not differ between the two features, the one by NIL1 suggests that Nitrogen is used in the form of NO_2 or $NOOH$. This property stems from closed itemset mining, which returns the longest itemset among the itemsets with the same frequency. This additional information is useful for practitioners to understand the classification mechanism, and designing new molecules.

VI. DISCUSSION

LASSO is known to be an interpretable method due to its ability to induce sparseness to regression coefficients. In this paper, we incorporated two additional factors which could improve interpretability. The first one is itemset-based features. Machine learning algorithms which employ itemset-based features are known to be interpretable [13], [30]. The second one is non-negativity on regression coefficients. Non-negative matrix factorization is

TABLE 6. Representative features found by NIL1. In the table, A stands for any atom, Q stands for any atom other than H or C.

Index	MACCS ID	Structure	Frequency	Coverage(%)	Accuracy(%)
1	84 133 135 151 156 158 161 162 165	( or ) $\cap NH_2 \cap CN$	7	16.8	78.3
2	70 94 124 130 161	$QNQ \cap QQ$, such as, NO_2, $NOOH$ or NNO.	6	19.2	90.1
3	46 134	Br	5	2.34	81.3
4	133 135 145 151 156 158 161 162 163 165	 $\cap NH_2$.	4	11.8	77.8
5	70 94 124 130 156 161	$QNQ \cap QQ$, such as NO_2, $NOOH$ or NNO.	3	18.1	91.1
6	16 22 137 153 165	3 molecules ring $QAA \cap QCH_2A$, such as 	3	2.49	88.2
7	16 22 137 165	3 molecules ring QAA, such as or	3	2.92	80.0
8	34 99 103 134	Double bond $\cap$ Halogen	3	0.731	83.3
9	52 70 94 124 130 142 161	$QNQ \cap NN$ such as NNO	3	0.0994	91.2
10	8 11 57 89 109 123 127 137 143 153 154 157 159 164 165	4 molecules ring $QAAA$ $\cap OAAAO \cap OCO$ $\cap QCH_2O \cap C = O$ such as 	3	0.00292	100
11	84 133 151 156 158 161 165	Ring-N $\cap NH_2 \cap CN$	3	0.0526	80.6
12	94 122 124 148 156 158 161	$NA(A)A \cap QA(A)A \cap QQ \cap AN(A)A \cap QN$ such as $NOOH, NNO$	3	0.208	85.2

often used as an analytical and interpretable tool due to its ability to automatically extract sparse factors [15]. Its interpretability is deemed to originate from the addition of positive factors, which is inherited to non-negative least squares in our case.

On the other hand, our model has limitations in its interpretability. For example if we consider an itemset $\{A, B\}$, then its occurrence is highly correlated with the occurrence of either item A or item B. Pattern-based features are often overlapping each other, then their interpretations are not straightforward. These are common problems when we employ patterns as features [26], and their solutions are left remained as a future work.

VII. CONCLUSION

In this paper, we have proposed to incorporate interaction terms to the non-negative least squares problem. For suppressing the resulting intractable number of variables, we took advantage of ℓ_1 regularization and non-negativity constraints. For efficiently finding interaction terms, novel bounding conditions are introduced to the tree based search space. In terms of the prediction accuracy, enforcing not only non-negativity constraints, but also ℓ_1 constraints together turned out to be useful. It can be attributed to the ability of ℓ_1 constraints which can avoid overfitting. In application to HIV-1 drug resistance prediction, we discussed the necessity of introducing both non-negativity constraints and variable interactions from the biological viewpoint. In the application to mutagenicity prediction problem, we demonsrtrated that the proposed method is relatively easy to interpret and useful for predictive as well as for explanatory purposes.

REFERENCES

[1] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Databases*, 1994, pp. 487–499.

[2] R. Bro and S. D. Jong, "A fast non-negativity-constrained least squares algorithm," *J. Chemometrics*, vol. 11, no. 5, pp. 393–401, 1997.

[3] D. Chen and R. Plemmons, "Nonnegativity constraints in numerical analysis," in *Proc. Symp. Birth Numer. Anal.*, 2007, pp. 1–32.

[4] D. Das, J. Ito, T. Kadowaki, and K. Tsuda, "An interpretable machine learning model for diagnosis of Alzheimer's disease," *PeerJ*, vol. 7, p. e6543, Mar. 2019.

[5] J. L. Durant, B. A. Leland, D. R. Henry, and J. G. Nourse, "Reoptimization of MDL keys for use in drug discovery," *J. Chem. Inf. Comput. Sci.*, vol. 42, no. 6, pp. 1273–1280, 2002.

[6] B. Efron, T. Hastie, I. M. Johnstone, and R. Tibshirani, "Least angle regression (with discussion)," *Ann. Statist.*, vol. 32, no. 2, pp. 407–499, 2004.

[7] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Jan. 2003.

[8] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2000, pp. 1–12.

[9] T. Hastie, R. Tibshirani, and M. Wainright, *Statistical Learning With Sparsity: The Lasso and Generalizations*. Boca Raton, FL, USA: CRC Press, 2015.

[10] C. Helma, T. Cramer, S. Kramer, and L. De Raedt, "Data mining and machine learning techniques for the identification of mutagenicity inducing substructures and structure activity relationships of noncongeneric compounds," *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 4, pp. 1402–1411, 2004.

[11] H. Kim and H. Park, "Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis," *Bioinformatics*, vol. 23, no. 12, pp. 1495–1502, 2007.

[12] T. Kudo, E. Maeda, and Y. Matsumoto, "An application of boosting to graph classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 729–736.

[13] H. Lakkaraju, S. H. Bach, and J. Leskovec, "Interpretable decision sets: A joint framework for description and prediction," in *Proc. Int. Conf. Knowl. Discovery Data Mining (KDD)*, San Francisco, CA, USA, 2016, pp. 1675–1684.

[14] R. Lawson and C. Hanson, *Solving Least Squares Problems*. Philadelphia, PA, USA: SIAM, 1995.

[15] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999.

[16] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey, "Cutting planes in integer and mixed integer programming," *Discrete Appl. Math.*, vol. 123, nos. 1–3, pp. 387–446, 2002.

[17] N. Meinshausen, "Sign-constrained least squares estimation for high-dimensional regression," *Electron. J. Statist.*, vol. 7, pp. 1607–1631, Jan. 2013.

[18] K. Nakagawa, S. Suzumura, M. Karasuyama, K. Tsuda, and I. Takeuchi, "Safe pattern pruning: An efficient approach for predictive pattern mining," in *Proc. KDD*, 2016, pp. 1785–1794.

[19] N. M. O'Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch, and G. R. Hutchison, "Open babel: An open chemical toolbox," *J. Cheminformatics*, vol. 3, no. 1, pp. 1–14, Dec. 2011.

[20] S. Y. Rhee, J. G. Matthew, R. Kantor, B. J. Betts, J. Ravela, and R. W. Shafer, "Human immunodeficiency virus reverse transcriptase and protease sequence database," *Nucleic Acids Res.*, vol. 31, no. 1, pp. 298–303, Jan. 2003.

[21] S. Rosset and J. Zhu, "Piecewise linear regularized solution paths," Stanford Univ., Stanford, CA, USA, Tech. Rep. HAL:ccsd-00020066, 2003.

[22] H. Saigo, T. Uno, and K. Tsuda, "Mining complex genotypic features for predicting HIV-1 drug resistance," *Bioinformatics*, vol. 23, no. 18, pp. 2455–2462, Sep. 2007.

[23] M. Slawski, R. Hussong, A. Tholey, T. Jakoby, B. Gregorius, A. Hildebrandt, and M. Hein, "Isotope pattern deconvolution for peptide mass spectrometry by non-negative least squares/least absolute deviation template matching," *BMC Bioinf.*, vol. 13, no. 1, pp. 1–18, Dec. 2012.

[24] M. Slawski and M. Hein, "Non-negative least squares for high-dimensional linear models: Consistency and sparse recovery without regularization," *Electron. J. Statist.*, vol. 7, pp. 3004–3056, Jan. 2013.

[25] M. Takayanagi, Y. Tabei, and H. Saigo, "Entire regularization path for sparse nonnegative interaction model," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Singapore, Nov. 2018, pp. 1254–1259.

[26] I. Takigawa and H. Mamitsuka, "Generalized sparse learning of linear models over the complete subgraph feature set," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 3, pp. 617–624, Mar. 2017.

[27] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc., B, Methodol.*, vol. 58, no. 1, pp. 267–288, 1996.

[28] K. Tsuda, "Entire regularization paths for graph data," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2007, pp. 919–926.

[29] T. Uno, T. Asai, Y. Uchiyama, and H. Arimura, "LCM: An efficient algorithm for enumerating frequent closed item sets," in *Proc. CEUR Workshop*, vol. 90, B. Goethals and M. J. Zaki, Eds., New York, NY, USA, 2003, pp. 1–10.

[30] S. Vojíř, V. Zeman, J. Kuchař, and T. Kliegr, "EasyMiner.Eu: Web framework for interpretable machine learning based on rules and frequent itemsets," *Knowl.-Based Syst.*, vol. 150, pp. 111–115, Jun. 2018.

[31] M. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New algorithms for fast discovery of association rules," in *Proc. 3rd Int. Conf. Knowl. Discovery Data Mining (KDD)*, Newport Beach, CA, USA, 1997, pp. 283–286.

MIRAI TAKAYANAGI received the B.E. degree in electrical engineering and computer science from Kyushu University, in 2019. He is currently enrolled with the School of Information Science and Electrical Engineering, Kyushu University.

YASUO TABEI received the M.Sci. and D.Sci. degrees from The University of Tokyo, Japan, in 2007 and 2010, respectively. From 2010 to 2013, he was a Postdoctoral Researcher with the Minato Discrete Structure Manipulation System Project, JST ERATO. From 2013 to 2016, he was a Research Scientist with the Advanced Core Technologies for Big Data Integration, JST PRESTO. He started the current position, in 2017. He is currently a Unit Leader of the Succinct Information Processing Unit, RIKEN Center for Advanced Intelligence Project. Many of his research papers have been published in top conferences of data mining, machine learning, and bioinformatics, such as KDD, ICDM, IJCAI, ISMB, and ECCB. His research interests include data mining, machine learning, data compression, and their application to chemoinformatics and bioinformatics.

EINOSHIN SUZUKI received the B.E., M.E., and D.E. degrees from The University of Tokyo, in 1988, 1990, and 1993, respectively. From 1993 to 1996, he was a Faculty Member with Tokyo Institute of Technology. From 1996 to 2006, he was a Faculty Member with Yokohama National University. Since 2006, he has been a Professor with Kyushu University. His research interests include data mining, machine learning, and autonomous mobile robots. He is on the editorial board of *JIIS* and has served as a PC Member for KDD 12 times.

HIROTO SAIGO received the master's and Ph.D. degrees in informatics from Kyoto University, in 2003 and 2006, respectively. He was also a Visiting Student with the University of California, Irvine (UCI). He worked as a Research Scientist with the Max Planck Institute (MPI) for Biological Cybernetics, from 2008 to 2010, and MPI for Informatics, from 2008 to 2010. From 2010 to 2015, he was an Associate Professor with Kyushu Institute of Technology. Since 2016, he has been an Associate Professor with Kyushu University. His research interest includes development of statistical machine learning methods tailored for problems in bioinformatics and cheminformatics. He serves as a reviewer/PC member for numerous top-level conferences and journals.