LASSO Logic Engine: harnessing the logic parsing capabilities of the LASSO algorithm for longitudinal feature learning

Jason Orender
Department of Computer Science
Old Dominion University
Norfolk, VA
joren001@odu.edu

Mohommad Zubair
Department of Computer Science
Old Dominion University
Norfolk, VA
zubair@cs.odu.edu

Sun, Jiangwen
Department of Computer Science
Old Dominion University
Norfolk, VA
jsun@cs.odu.edu

***Abstract*—Longitudinal data, which is widely used in many disciplines to study cause and effect, poses significant computational challenges to both modeling and analysis. Longitudinal data is composed of readings on the same variable collected over time and is often high-dimensional with correlated features. The combinatorial search approach for identifying the optimal features is unrealistic for most applications. The alternative approaches, such as heuristics, greedy searches, and regularization techniques, including LASSO, can result in models that suffer from both low accuracy and unclear feature attribution. In this paper, we propose a binary transformation on the data before applying LASSO for feature learning. As demonstrated in the paper, the binary transformation enhances signal in the data, resulting in highly accurate feature attribution, including associated time lags. It avoids the typical shortcomings of the LASSO algorithm, including saturation of the feature space and arbitrary or inconsistent sparse feature selection. Both synthetic data and real-world data sets were used to demonstrate the value of the proposed transformation and in every case substantial improvements in feature learning were seen. In addition, the scalability of the solution is superior to that of the standard LASSO since an integrated high performance computing (HPC) solution would utilize integer math for most of the computational effort.**

Keywords—LASSO, logistic regression, regularization, logic, feature selection, longitudinal, classification, algorithm

I. INTRODUCTION

Longitudinal data refers to the observation of several independent variables (features) along with a specific outcome of interest for a period of time. The need for analysis of longitudinal data occurs in many disciplines, such as medical research [1, 2, 3], business [4, 5, 6], climate science [7, 8, 9], and experimental scientific disciplines of many kinds [10, 11, 12]. Analysis of such data is critical in developing drugs, effective treatments, and uncovering new cause and effect relationships in the physical sciences as well as psychological inquiry. A way is needed for the researcher to consistently and accurately identify not only the variables of interest in the data, which rules out many "black box" type solutions, but any time lag interposed between cause and effect.

This sort of data can pose a significant challenge in that it often encompasses a large set of features with the added complexity that many of them may be highly correlated. A large number of features (high dimensionality of the data) makes it computationally difficult to analyze the data set and build models. In addition, the potential correlation between many features poses a problem for feature selection (learning) techniques in that it is difficult for one feature to stand out among many that are somewhat similar. Since naïve combinatorial strategies are unrealistic for any but the smallest data sets, this data will typically be analyzed via heuristics [13, 14, 15, 16], greedy searches [17, 18, 19], or regularization techniques [20, 21, 22, 23]. However, these methods cannot guarantee optimality [24] and will often select a specific feature from a set which demonstrates multicollinearity arbitrarily and inconsistently [25]. Frequently, the researcher will also attempt to solve the multicollinearity problem by pruning the feature set using some expert knowledge specific to a particular information domain.

Related works. Using LASSO for feature selection with categorical variables is not a new concept. The "group LASSO" as originally conceived by Yuan and Lin [26], implemented by Meier, van de Geer and Buhlmann [27] as well as Yang and Zou [28] who based their application on the work of Kim, et. al. [29] was created specifically for this type of problem. This strategy uses an additional penalty that is assigned to groups of categorical variables which in typical application are intended to represent different states of a single variable. Those papers essentially improve upon the LASSO algorithm in a specific circumstance. In contrast, this work proposes a novel application of the algorithm somewhat abstracted from any particular implementation. In any case, group LASSO is known to suffer from estimation inefficiency and selection inconsistency [30].

The work of Katrutsa and Strijov [24] is similar in concept to this one. They propose representing a feature presence using a binary vector, but they propose using a system based on quadratic programming in which the quadratic term is a pairwise feature similarity and the linear term is a measure of relevance. Their objective was to "minimize the number similar features and maximize the number of relevant features." As stated, the

selection criteria in that work are evaluated pairwise, so by utilizing information on relevance gleaned from a series of hypothesis tests (with their associated *p*-values) and judging the similarity based on either the Pearson correlation coefficient or estimating the probability distribution using the concept of shared mutual information, the most relevant features that were also judged to be sufficiently dissimilar were retained. Since all features are never evaluated simultaneously, however, any confounding factors will potentially be missed, and this method is also comparatively computationally intensive and may not scale well.

Motivation. This methodology was developed specifically for situations in which a confluence of events causes another event, for the simple reason that most of the methods available have shortcomings when applied to this specific problem type. The case studies will show that using a traditional regression analysis can produce comparatively substandard results in situations like this, especially when specific causal attribution is desired. This confluence is considered as a logical event in which a set of binary conditions 'A', 'B', 'C', etc. are met in order to produce event 'E'. While the input conditions are binary in the sense of whether or not a condition has occurred, the binary element may be couched in terms of a continuous variable threshold, for example whether a pressure or temperature reading has exceeded or dipped below a threshold. An excellent example of this type of thresholding behavior in a physical application is demonstrated with the brittle fracture of materials. If a material is below a certain temperature and the stress increases above a specific amount, the material breaks catastrophically in brittle fashion. The problem being examined here comes about when such a phenomenon presents itself and the reasons are unknown, but massive amounts of data are available. This is perhaps sensor data on an engine, or possibly the results of a survey intended to identify risk factors. Sifting that data in an efficient manner to weed out irrelevancies has become an extremely important task in the age of big data. The case studies and practical analyses presented later in this paper will show some examples of these problem types as well as proposed solution methods.

TABLE I. TEST SET RESULTS SUMMARY

Case Study	True Positive / False Positive Results on Test Set		
	Type	***Transformed***	***Un-Transformed***
#1	Synthetic Data	97.8% / 0.0%	85.3% / 2.1%
#2	Synthetic Data	94.8% / 2.3%	81.0% / 2.3%
#3	Synthetic Data	96.2% / 0.8%	71.4% / 6.5%
GB	Real Data Set	97.3% / 11.9%	93.3% / 23.8%

The effectiveness of the method described here is demonstrated by case studies shown in section 4 using synthetic data, and section 5 using real-world data (see Table I). They show that not only are the solutions consistent with known results and that they reliably outperform the same calculation on un-transformed data, but that they are also both repeatable and sparse. Beyond the metrics shown in the table below, however, the precision with which it is possible to identify contributing features and associated time lags using this method is a valuable asset in itself.

In this paper, we propose a technique based on a targeted reduction in the input data's information content by creating a boolean version of the data that, when subjected to the LASSO algorithm, significantly outperforms a similar analysis on un-transformed data. This technique minimizes the aforementioned weaknesses, and we can also show via experimentation that optimality is achievable.

The main contributions of this paper are:

- Consistent and optimal feature selection.
- Computational efficiency at scale.
- Addressing the collinearity problem via optimal feature selection.
- A novel approach to modeling multiple independent causes for the same event $\big(P(Event) = P(A \cup B)\big)$.
- A demonstration of these advantages by comparison with a LASSO model using the same (un-transformed) data.

Organization. The paper is organized into six sections. After the Introduction, a short background section is meant to give a quick overview regarding what LASSO is and why it works. The next section presents the proposed LASSO logic engine, transforming the continuous data set into a matrix of logical vectors followed by applying LASSO for feature learning. The next section both introduces the approach for generating the synthetic data and discusses the case studies that utilize that data. The next section covers application to a real-world data set. Finally, the conclusion summarizes the main points and discusses the potential for application.

All case studies, synthetic data, real data sets, and implementation code are available in the GitHub repository for this paper (https://github.com/jworender/lle), and readers are invited to fork the repository and modify the parameters for the purpose of self-demonstration.

II. BACKGROUND

The LASSO algorithm, as conceived by Tibshirani in his seminal 1996 paper, minimizes the following expression which includes an error term and a regularizer [31]:

$$\sum_{i=1}^{N}\left(y_i - \sum_j \beta_j x_{ij}\right)^2 + \lambda \sum |\beta_j| \qquad (1)$$

Where:

- y_i = the ith observation in the data
- x_{ij} = the jth feature of the ith obs.
- β_j = the jth solution coefficient
- N = the number of rows / observations

- λ = a hyperparameter
- The first summation will be referred to as the "quadratic term" and the second will be referred to as the "regularization term."

Arguably, the most desirable quality of the LASSO, and why it would likely be selected over other regularized methods is its ability to produce a sparse solution. This is due to the way that the regularizer is expressed. In the LASSO, the regularizer (also referred to as the L1 norm of the coefficients) has the ability to shrink coefficient magnitudes to zero, effectively eliminating respective features from the model. This behavior can also generate criticism, which stems from the perception that the features eliminated are often arbitrary and inconsistent, and the feature space can become saturated, eliminating the sparseness advantage altogether [25]. As will be shown, however, when performed on a boolean version of the data, these criticisms are no longer applicable.

III. LASSO Logic Engine

To develop this methodology for continuous inputs, a trivial case will be initially discussed with successive complications added.

If it is known that a particular feature has a causative relationship with an event, we can simply watch the feature to determine the behavior when the event occurs. In Fig. 1, just such a circumstance is illustrated. When the feature enters a particular range, an event occurs. This could be the concentration of a reagent, the temperature of an object, a sensor measurement of some kind, etc. The more time that the feature is observed, the more certain the conclusions will be which can be drawn concerning the range in which the event occurs. If there are two features in the data set, only one of which can be the causative agent, it becomes slightly more complicated, but it would still be a relatively simple task to figure out which one could be causing the event by defining the range at which events actually occurred and then comparing them with the response vector and the other features.

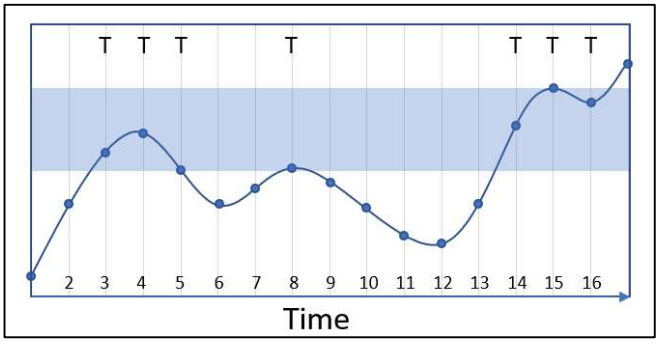

Fig. 1. Measurement of a feature over time. An event is signaled by the "T" over the curve, and the shaded area is derived from the highest and lowest values at which the feature is measured during an event. If this was observed, we could very quickly draw a conclusion about the behavior. This would become even more apparent as more data was gathered. It is clear that when the feature enters the shaded area an event occurs, and when it exits that region there are no events. The event can therefore be completely described by this feature and the critical range (the shaded area).

In Fig. 2, it is clear that the feature defined by the orange curve is not the sole causative agent, because if it were, there would be many more events. It could *not logically be* the event trigger on its own. The same procedure could be conducted for as many feature measurements as desired, but only the one that can completely define all events without including any non-event examples could possibly be the sole causative agent.

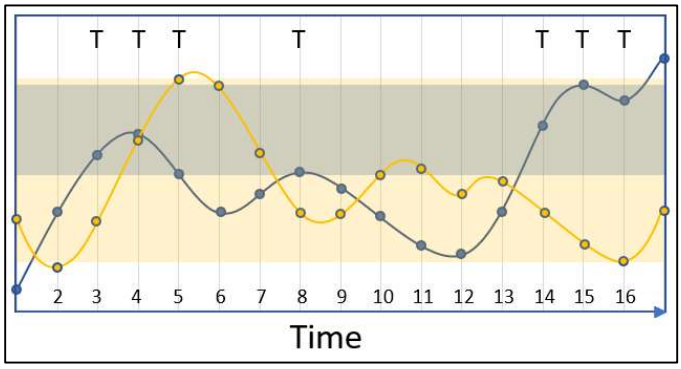

Fig. 2. When the same procedure (see Fig. 1) is tried for the orange curve as was done for the blue curve, it is apparent that when the shaded area is drawn between the maximum and minimum values at which an event occurs, it covers nearly the entire range of the curve. If this feature was actually the sole causitive agent, there would be events at time steps 1, 6, 7, 9, 10, 11, 12, 13 and 17, in addition to the events shown. Since this is not the case, it can be concluded that the orange curve is not the causitive agent, at least not by itself.

So, we will define the **critical range** as the range within which an event is *possible*, given the previous behavior and is defined by the range from the minimum value observed when an event occurs to the maximum value at which an event observation occurs.

A. Expanding the solution to include multiple triggers

The previous situation is a simplistic example, and it is unlikely that any important relationship requiring analysis would be so obvious. It turns out, however, that expanding this procedure to include situations in which *n*-features are required to simultaneously be within their own critical ranges for an event to occur does not require much more work.

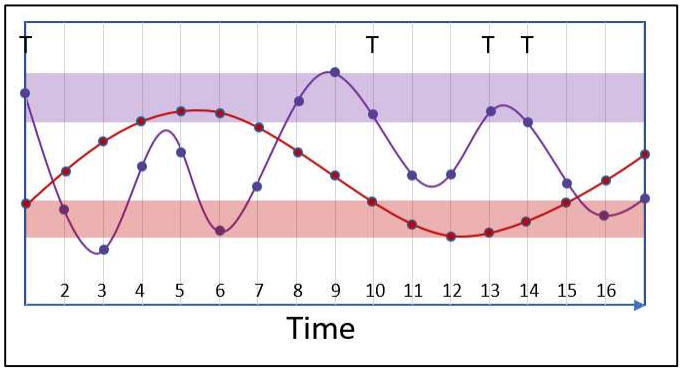

Fig. 3. Two features are shown, both of which are required to be within their critical ranges for an event to occur. During time steps in which the purple curve is within the purple range (top), while the red curve is simultaneously within the red range (bottom), an event occurs.

The critical range is still defined in the same way: the range maximum is the maximum feature value at which an event

occurs, and the range minimum is the minimum feature value in which an event occurs. In Fig. 3, a simple example is shown in which there are two features, both of which are required to be within their critical ranges for an event to occur. Each feature is independent of the other and they are both required for an event to occur.

In this problem statement, all that is known by the observer is the state of the response vector showing when the events occur and the measurements of each feature at the time steps given. The critical ranges are subsequently defined by when an event is *possible* given previous behavior. So again, the top of the red range is defined by the maximum value of the red curve at which an event occurs (time step 10), and the bottom of the critical range is defined by the minimum value of the red curve at which an event occurs (time step 12). The purple curve is similarly defined. This yields the following critical range truth table data if the examples that fall within the critical range are defined as "True", or a 1, and the examples that fall outside the critical range are defined as "False", or a -1. The first row of the transposed *X* matrix below represents the purple curve, and the second row represents the red curve:

$$X^T = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 \dots \\ 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

$$\begin{matrix} 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 \\ \dots 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 \end{bmatrix}$$

$$y^T = [T \quad F \quad F \quad F \quad F \quad F \quad F \quad F \quad F \dots$$

$$\dots T \quad F \quad F \quad T \quad T \quad F \quad F \quad F]$$

Employing the inherent logic parsing capabilities of the LASSO algorithm, a solution can be obtained which will clearly identify these columns even when obfuscated by mingling with multiple non-relevant features, and even when there are non-relevant features that are nearly the same (i.e., highly correlated). Since it is unknown at the outset which features may be relevant, among possibly hundreds or thousands of features, this provides a significant advantage to the computationally efficient LASSO method. Simulated data has been created which demonstrates this property with an added twist – a lag is introduced between the feature entering the critical range and the event occurrence, creating a lagged longitudinal data problem; this precipitates the complicating factor of having a high degree of multicorrelation among features from the same variable but measured at different time steps. See the next section for the simulated data demonstration. The simulated data creation process is also discussed in detail at the beginning of that section.

This procedure will work for a large number of features with multiple logical AND relationships and depends only upon the critical range learned from the data itself, the binary classification of the continuous measurement according to where it lies with respect to the critical range, and of course the given response vector.

B. Adding a logical OR to the solution set

If there are only logical AND relationships between the relevant features, the previous procedure will yield a complete solution, and the performance characteristics of the resultant model will reflect this. However, if there are multiple independent causative factors for the same event that constitute a logical OR relationship, the solution will likely fail on validation.

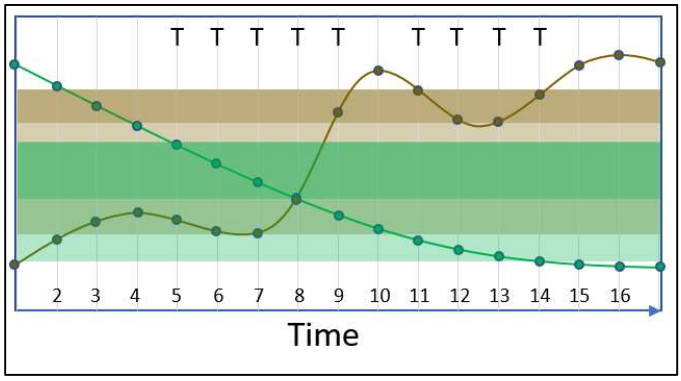

Fig. 4. Two features are shown, either of which can independently cause an event to occur. As a consequence, the critical range extends over both shaded areas for both features. The dark shaded areas are the ground truth critical ranges, and the light shaded areas (inclusive of the dark shaded areas) are the apparent critical ranges. The events at time steps 5, 6, 7, and 8 are caused by the green curve being within its critical range, and the event at time 7 also corresponds to the lowest extent on the brown curve apparent critical range since that is the minimum value for an example on the brown curve that also corresponds to an event.

The procedure described here offers a potential solution, however. The solution becomes apparent after one realizes that the critical range tabulated in the truth table is just an *apparent* critical range and may not represent the ground truth. The example in Fig. 4 shows two features over time and an event that can be caused by either feature passing within its respective "ground truth" critical range. The apparent critical range, however, will be much wider if there are multiple causative factors which can each independently trigger an event. This can be discerned in the example by noting that the minimum value of the apparent critical range of the brown curve, for instance, is due to an event caused by the green curve (time step 7), and the minimum value of the apparent critical range for the green curve is due to an event caused by the brown curve (time step 14). Since the causation of the events is presumably unknown at the time of the analysis, this situation extends both apparent critical ranges downward significantly.

The critical range truth table looks like the following, (the first row of the transposed matrix represents the brown curve, and the second is for the green):

$$X^T = \begin{bmatrix} -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \dots \\ -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\begin{matrix} -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 \\ \dots 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 \end{bmatrix}$$

$$y^T = [F \quad F \quad F \quad F \quad T \quad T \quad T \quad T \quad T \dots$$

$$\dots F \quad T \quad T \quad T \quad T \quad F \quad F \quad F]$$

Note that the method of determining the apparent critical ranges ensures that when the underlying features have a logical OR relationship which produces the response vector, the *apparent* critical ranges will still appear to have a logical AND relationship to produce the same response vector. The following logical proof applies:

e = event ‘E’ occurs

a = event ‘E’ is caused by feature ‘A’

b = event ‘E’ is caused by feature ‘B’

p_a = feature ‘A’ is within its apparent critical range

p_b = feature ‘B’ is within its apparent critical range

Given:

- (G1) Features ‘A’ and ‘B’ have an ‘OR’ causative relationship with ‘E’, meaning that either feature ‘A’ or feature ‘B’ can cause ‘E’ independent of the other.
- (G2) All events are caused by either feature ‘A’ or feature ‘B’.
- (G3) The apparent critical range must include all events since the critical range is defined by the range of values in which events occur.

1. $e \leftrightarrow a \vee b$ $(G1, G2)$
2. $e \rightarrow p_a \wedge p_b$ $(G3)$
3. $a \vee b \rightarrow p_a \wedge p_b$
   (Hypothetical Syllogism using 1 and 2)

So, the result is that whenever an event caused by either feature ‘A’ or feature ‘B’ occurs, both feature ‘A’ and feature ‘B’ must be within their *apparent* critical ranges, causing this to appear as if both feature ‘A’ and feature ‘B’ are required for the event ‘E’ to occur. It follows that given only this information, a logical AND and a logical OR relationship will appear indistinguishable. However, when applied to a test set the resultant model will likely do poorly if modeled as a logical AND relationship since these are two very different relationships in practice. When performing binary transformation, the apparent critical ranges may well cover the entire observed ranges. With enough data and an effectively random association between the variables, this will almost certainly be the trend. Even with the small amount of data shown in Fig. 4, this is nearly the case. There is a solution to this problem, however, and it relates to the excellent parsing capabilities of the LASSO algorithm over a great many features.

To develop what the ‘OR’ version of the feature might look like, it is useful to take a closer look at how the critical ranges might change if the supposition that the relationship is an ‘OR’ relationship is assumed. Comparing the apparent critical ranges with the complement to the response vector in the previous example, it becomes apparent that the True (1) values indicated by their respective critical ranges for time steps 3 and 4 for the brown curve and time step 10 for the green curve would be impossible if an OR relationship existed. This allows moving the minimum of the critical range up to the next example above that range. Similarly, the green curve can be limited to the next example above time step 10, almost contracting it to its ground truth size. With enough data, the ground truth critical ranges, or very close, should be able to be determined if a logical OR relationship is known (or assumed) to exist.

As stated, these relationships hold only for the ground truth critical ranges, and only if we know (or assume) that a logical OR relationship exists. However, by calculating a *version* of a ground truth critical range that *might* exist and applying that to the feature data, then judging them all simultaneously as if they were separate features, the LASSO algorithm can pick out the best possible combination of these versions; they can then be used to construct a prospective model.

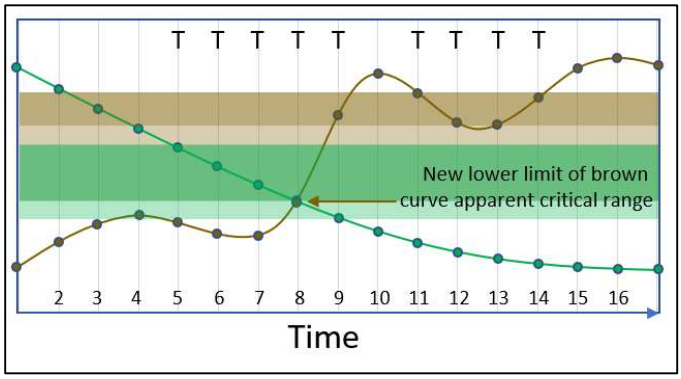

Fig. 5. The new implied size of the critical ranges if an OR relationship is assumed instead of an AND relationship. The green curve is almost contracted to its actual size, and the brown curve critical range is considerably smaller.

These different versions are calculated by narrowing the critical range in small increments to eliminate examples that stray outside the positive examples of the response vector. This, of course, also potentially eliminates examples within the set of examples corresponding to the positive examples in the response vector, presumably leaving only those with a complementary set from another feature undergoing the same process.

With a large data set, calculating every possible version of the critical ranges that might exist is a computationally intensive task, but getting close enough to produce a viable model is reasonably easy (see the simulated data experiments for an example). When there are examples that fall within the apparent critical ranges, but do not coincide with an event, contracting the critical range inside the value that this point represents produces a version of the critical range that *might* exist, and this is judged as a separate feature. The next version of the critical range will be generated by moving the range in further past the next point that falls within the critical range but is not coincident with a positive example. The process can be continued until there are no more examples that meet the required criteria.

An alternate way to generate the critical range versions if the data set is very large is to contract the critical range in n equally sized steps starting from the edge of the apparent critical range, inward to the point at which there are no more examples that are both within the critical range but do not coincide with a positive example. This method allows usage of as little or as many computational resources as desired. This will not be an exact solution, but as the simulated data shows, it does produce a potentially highly accurate and workable model, which can be a starting point for disentangling the results to produce separate models for each of the ‘OR’ clauses that cause the same event.

When the versions of the features with variable critical ranges are generated, some will show an optimal range equal to their apparent critical range. These will be assumed to be part of a logical AND relationship, which is by far the highest probability choice given that circumstance.

C. *Creating a functional model*

Once the feature versions are created, the critical range truth table that is produced can be directly fitted with the LASSO to produce a set of coefficients that will generate an outcome.

The information required to compose a model, then, is a set of hyperparameters defining the critical ranges of each non-zero feature in the sparse solution, and the coefficients of the LASSO fit. The critical ranges are used to transform an arbitrary set of feature inputs into the binary coded version which will then be used with the coefficients to determine a predicted outcome.

IV. Simulated Data Experiments

To illustrate this technique in practice, a set of simulated data experiments was conducted, which introduced a variety of complications.

A. *Generation of the simulated data*

A set of time-varying, range-bound, and suitably complex simulated data was generated using a **sine function** as the basic building block for the simulated data and then using superposition to generate complexity.

The procedure:

1. Generate a large number of sine waves with five possible pseudorandomly chosen frequencies per cycle. Something like the curve in Fig. 6 was generated.

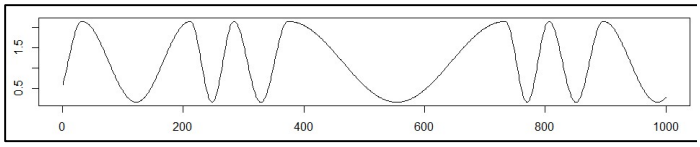

Fig. 6. The type of curve generated in step 1 of the simulated data generation procedure. Many of these curves were generated (~50-100 for most experiments).

2. Superpose several of the curves generated in step 1. Something like the curve in Fig. 7 was generated by this step.

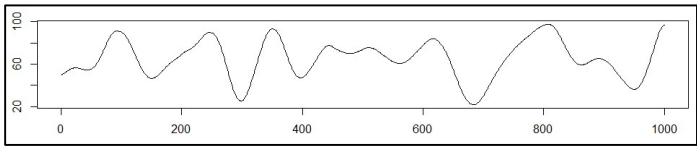

Fig. 7. The type of curve generated in step 2 of the simulated data generation procedure after several of the curves generated from step 1 were superposed.

3. To create a diversity of values a new range was selected using a pseudorandom normal distribution.

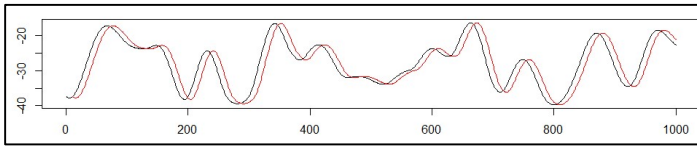

Fig. 8. A lagged version of the curve (shown in red) was created in order to test the capabilities of the method for interpreting lagged longitudinal data. Lags can have a dramatic effect on the critical range, and not taking them into account with real data can lead to error-prone models.

4. A version of the waves was shifted by a pre-determined amount to simulate lag. Fig. 8 shows an example of this.
5. Enough curves were assembled to have a large selection of potentially "relevant" and non-relevant curves. Note that the shifted curves determine when the event occurs, while the unshifted curves are matched with the time step in which they occurred to compose the data.
6. Each time step is interpreted as a separate example, and if all relevant curves meet the selection criteria during that time step, it is labeled as a "True" example, and otherwise is labeled as "False".
7. The data is next flattened such that a row contains the data from the current time step as well as a fixed number of time steps previous. The data from time steps zero (the current time step) through nine (the data nine time steps in the past) on one row, for instance, will be the data for time steps one through ten on the next chronological example.

B. *Case I*

The first case is a longitudinal data problem in which the following specific characteristics apply:

1. 7 relevant variables are intermixed with 33 non-relevant variables, all 7 of which must be within their relevant ranges to trigger an event.
2. Each variable is tracked over 10 previous time-steps as well as the current time step (time step zero) and as a consequence generates 11 features per variable, for a total of 440 features per example in the flattened data.
3. There is a single time-step/feature relevant for each variable. As a result, the ratio of non-relevant to relevant features is approximately 63:1. The time step features are meant to describe the delayed effect once the variable enters its critical range.
4. Fig. 9, which shows the results of the model fit plotted against the test set, tells a very straightforward story. The resultant model describes the data extremely well when the data is transformed into a binary format using the procedure described in section 3. If the transformation is not used, Fig. 10 shows the resultant LASSO fit on the un-transformed data.

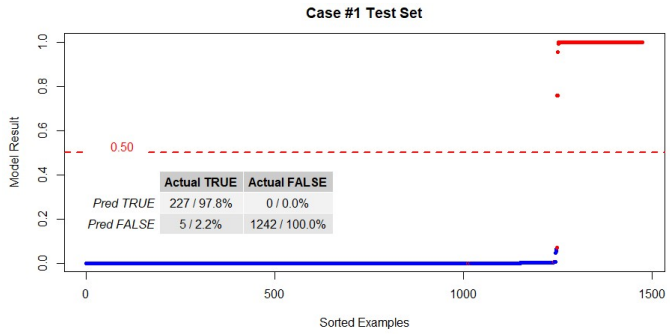

Fig. 9. The plotted results for case #1 test data. The red markers are positive examples and the blue markers are negative examples. 97.8% of the positive examples were classified correctly, and all of the negative examples were classified correctly.

Without the transformation, the LASSO fit severely underperforms with respect to the fit performed with the transformed data. In addition, the computational power required to calculate the fit on the un-transformed data is significantly greater. The LASSO fit on the un-transformed data took nearly double (1.83x) the amount of time required to both transform the data and then perform the fit in the transformed data case (single-threaded). In addition, once the data is transformed, the remainder of the calculations can nearly all be handled via integer math until the final coefficients are calculated.

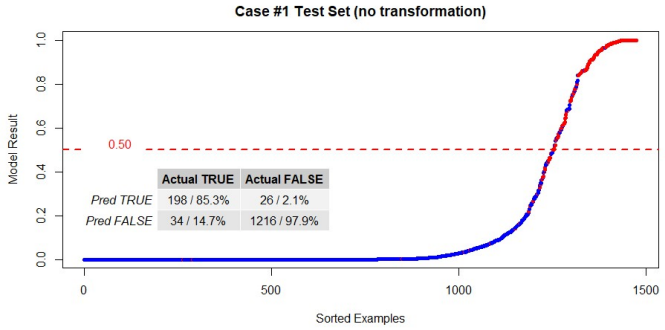

Fig. 10. The plotted results for case #1 test data using a LASSO on un-transformed data. The red markers are positive examples and the blue markers are negative examples. 85.3% of the positive examples were classified correctly, and 97.9% of the negative examples were classified correctly. The un-transformed data showed a significant performance defecit when compared to the transformed data.

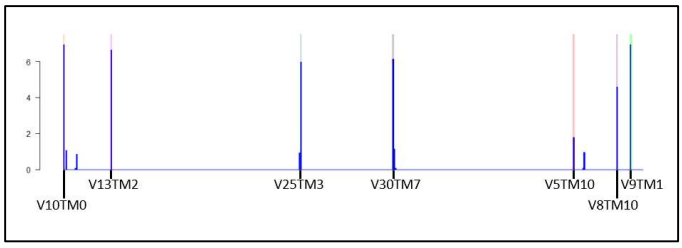

Fig. 11. A bar plot of the coefficients returned by the LASSO algorithm on transformed data. The colored background lines are the precise locations of the correct variables and time lags, while the blue bars show the locations and magnitudes of every coefficient returned. In every case except for variable #5 at time step 10, the indicators were unambiguous. Variable #5 still had almost double the coefficient value of any of the spurious returns.

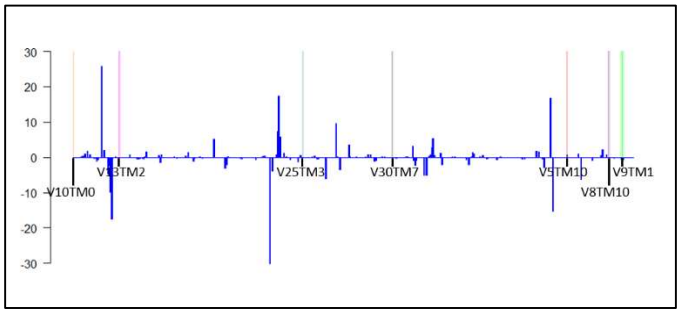

Fig. 12. A bar plot of the coefficients returned by the LASSO algorithm on un-transformed data. The colored background lines are the precise locations of the correct variables and time lags, while the blue bars show the locations and magnitudes of every coefficient returned. There is not a single case of a coefficient appearing in the correct location and there are numerous spurious returns on non-relevant variables.

The advantages of the transformation procedure go beyond the accuracy of the results and the reduced computational power. The precision with which it can pinpoint the precise features required for the sparse solution means that in a longitudinal data scenario such as this, both the variable *and the specific lag for each variable* can potentially be derived. Fig 11 shows a bar plot of the coefficients calculated for the LASSO applied to the transformed data. The sparseness of the solution is almost ideal; there are only five spurious returns and three of those were directly adjacent to the correct return.

This can be contrasted with the LASSO solution on the un-transformed data in Fig 12. The scattershot arrangement of the coefficient magnitudes clearly reveals why the solution shown in Fig. 12 so greatly underperforms.

C. *Case II*

The second case is a variation on the first case but shows the degradation of answer fidelity when not all relevant variables are present. Using the same data set as the previous case, but with relevant variable #5 deleted from the data set before training yields a coefficient bar plot nearly identical to Fig. 11. There is very little difference between the coefficients with the exception of the absent variable #5. The low magnitude of the coefficient was a clue that variable #5, while relevant, was not of great importance to the final solution. This is because all but a handful of examples can be fully explained by the other relevant variables. However, if a more important relevant variable is not in the training set, there is a considerably greater effect. With both variable #5 and variable #10 missing, Fig. 13 shows that the coefficients are far less clear about which variables and time steps are the most relevant.

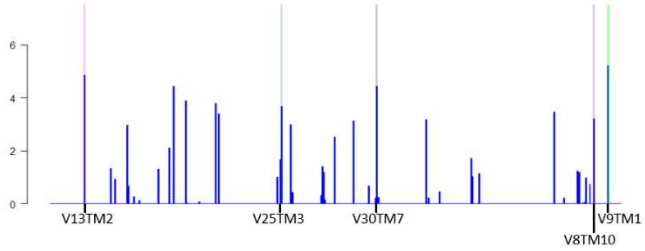

Fig. 13. A bar plot of the coefficients returned by the LASSO algorithm on transformed data, but with the relevant data of variables #5 and #10 missing from the training set. The colored background lines are the precise locations of the correct variables and time lags, while the blue bars show the locations and magnitudes of every coefficient returned. Losing both variables #5 and #10 made a big difference in the interpretability of the coefficients.

The coefficients when there is data missing are far less sparse and not nearly as interpretable as Fig. 13 and Fig. 16, but it does show that the correct variables and time steps were still captured, even if they did not stand out from all the other spurious returns on non-relevant variables by virtue of coefficient size. Because of this, the model which used the transformed data (Fig. 14) still outperforms the one that was fitted with the original continuous data (Fig. 15) by a large margin.

The more missing data there is, the greater the level of noise in the final result. While this is inconvenient, it is also a reliable indication that there might be an under- or un-represented

causative factor in the data, perhaps leading an investigator to continue searching for new features to achieve a sparse return similar to Fig. 11.

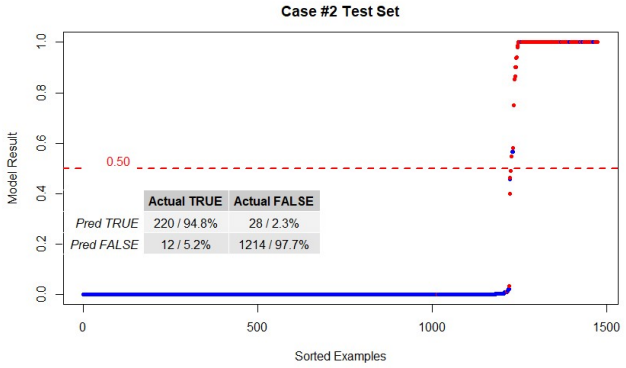

Fig. 14. The plotted results for case #2 test data. The red markers are positive examples and the blue markers are negative examples. 94.8% of the positive examples were classified correctly, and 97.7% of the negative examples were classified correctly. Even with the extra noise created by the absence of an important relevant variable, the model made from the transformed data still performed very well.

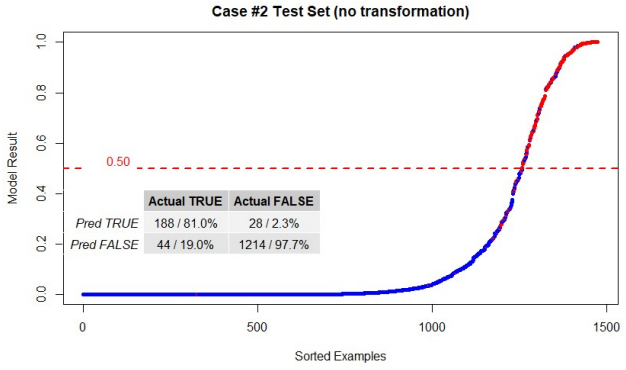

Fig. 15. The plotted results for case #2 test data with no data transformation. The red markers are positive examples and the blue markers are negative examples. 81% of the positive examples were classified correctly, and 97.7% of the negative examples were classified correctly. This represents a significant underperformance compared to the model created using the transformed data.

D. Case III

The third case introduces a logical OR relationship between four variables of the form:

$$AB + CD = Z$$

There are two variables in a logical AND relationship on either side of a logical OR operator. For the reasons explained previously, this situation poses a challenge to this method because the way that the apparent critical ranges are determined ensures that a logical AND and a logical OR relationship appear identical. If a logical OR relationship exists, the training set exhibits the distinctive pattern shown in Fig. 16.

The perfect classification of the positive examples and poor classification performance of the negative examples is a distinctive relationship that occurs primarily when there is a logical OR relationship present. Once this is known, steps can be taken to make the additional computational investment of creating the different versions of the features that might exist with different critical ranges and then fitting the new (much larger) data set. The computational advantage is decisively reversed when this is done because of the extreme size of the new data set being fitted to the model. However, the increase in performance in terms of the accuracy of the model when compared to the non-transformed data is arguably worth expending the extra computational energy to produce the much better results. Fig. 17 shows the success of the fit against the test set with the extra versions calculated.

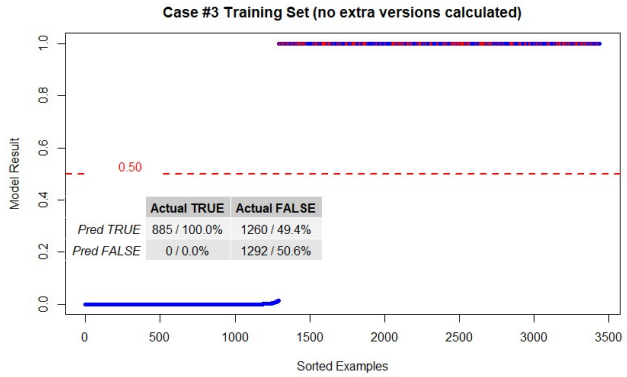

Fig. 16. The plotted results for case #3 training data. The red markers are positive examples and the blue markers are negative examples. 100% of the positive examples were classified correctly, and 72.9% of the negative examples were classified correctly. This distinctive pattern in the training set, where all of the positive examples were classified correctly, but there were many false positives is indicative that a logical OR relationship exists within this data set.

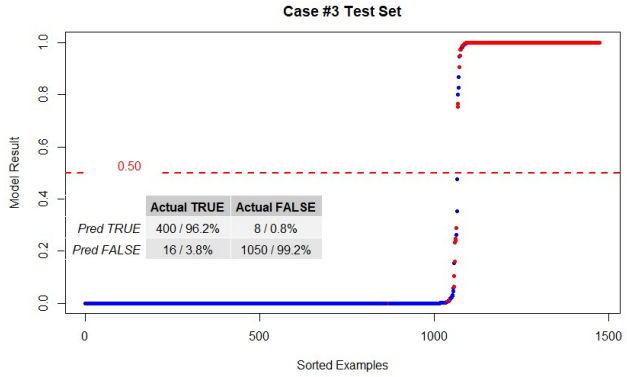

Fig. 17. The plotted results for case #3 test data with the additional feature versions calculated. The red markers are positive examples and the blue markers are negative examples. 96.2% of the positive examples and 99.2% of the negative examples were classified correctly. There is a heavy advantage demonstrated here to encoding the extra versions despite the computational investment required.

It is clear from the performance of the new training set that there is a significant performance boost when the new versions of each feature with slightly different critical ranges are created. The level of additional computational power is very significant, however. The original data set for Case III was 3,437 examples with 56 features each, but when the new versions are calculated to a modest resolution the data set expands to 3,437 examples with 526 features each. That is approximately one-sixhundredth of the maximum possible size of the expanded version set.

Remarkably, however, it appears to be enough to create a reasonably accurate model. Fig. 17 shows the test set when applied to the model created with the limited number of extra versions, and the performance advantage over the LASSO applied to the original continuous data is markedly large.

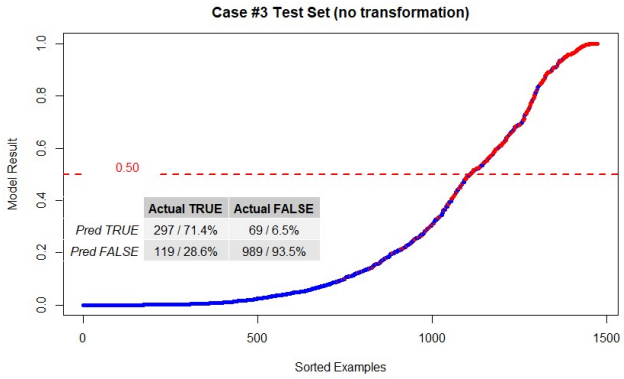

Fig. 18. The plotted results for case #3 test data with no transformation. The red markers are positive examples and the blue markers are negative examples. 71.4% of the positive examples and 93.5% of the negative examples were classified correctly. The difficulties in modeling a data set with multiple independent triggers for the same event are clearly demonstrated here.

The LASSO results with the unmodified continuous data are in Fig. 18, and show that while there is definitely a relationship, the difficulty in creating a model in which multiple features within the same data set can independently cause the same result is evident.

While limiting the size of the expanded data set with the multiple versions produces a highly accurate model (Fig. 17), the cost is that the interpretability of the coefficients suffers. In order to regain that interpretability, the features must be disentangled such that the logical OR relationships are removed, and each set of events modeled separately.

V. Real-World Data Set Application

Beyond the laboratory confines of the simulated data, this technique has also been used to analyze real-world data sets. The following is one of those sets. Additional supplementary data sets will be posted in the repository for this paper.

One thoroughly analyzed data set that is often used as an exemplar is the ionosphere data collected by a system in Goose Bay, Labrador, downloaded from the UCI data repository [32]. The original study was published in 1989 [33] and has been subsequently used in at least 55 additional papers as a test set for machine learning algorithms.

The LASSO with transformed data (Fig. 19) again exceeded the performance the LASSO with the original un-transformed data (Fig. 20) and nearly matched the neural network that was specifically tuned and used to analyze the data in the original paper. Additionally, a sparse coefficient vector was returned (Fig. 21), which clearly indicated which signals were important, along with the critical ranges of the important signals, which are arguably nearly as interesting as the model results alone.

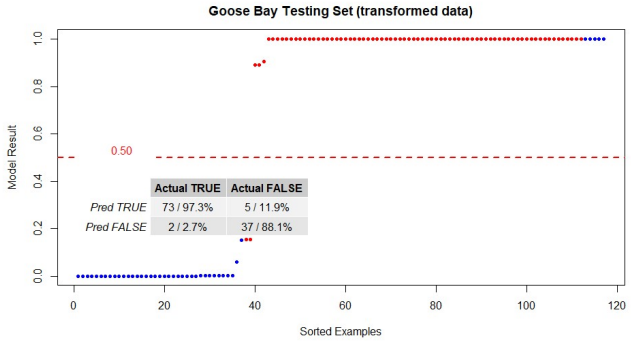

Fig. 19. The plotted results for ionosphere test data using transformed data. The red markers are positive examples and the blue markers are negative examples. 97.3% of the positive examples and 88.1% of the negative examples were classified correctly. The neural network used to analyze the data in the original paper achieved a 100% true positive rate at the same false positive rate.

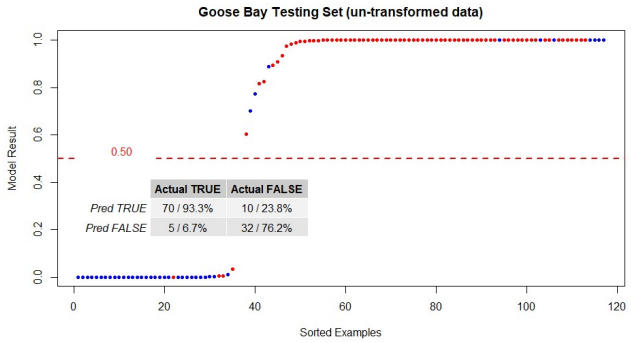

Fig. 20. The plotted results for ionosphere test data using transformed data. The red markers are positive examples and the blue markers are negative examples. 93.3% of the positive examples and 76.2% of the negative examples were classified correctly. The transformed data again out-performed the LASSO with un-transformed data in this application by a wide margin.

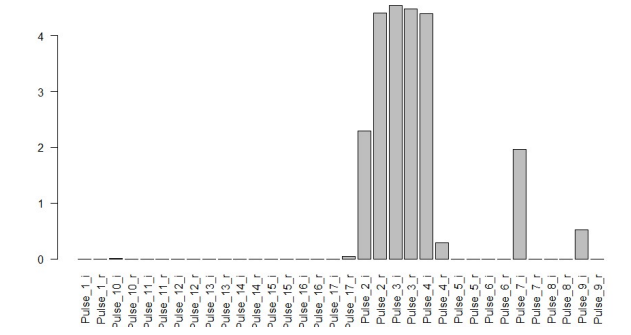

Fig. 21. A bar graph representing the coefficient vector for the transformed data solution of the Goose Bay data. The sparsness of the returned coefficient vector would enable selective monitoring of specific signals, which is information not available with the neural network solution.

VI. Conclusions

The method of shifting the frame of reference to one based in boolean logic relationships described in this paper provides an alternative to the traditional additive model generation techniques typical of regression type models by leveraging the ability of the LASSO algorithm to yield the best set of logical combinations to produce a given response vector. In addition,

as a byproduct of the fitting process for continuous variables, a set of the most important features along with their critical ranges are produced that may give insight into the causative factors behind the event being modeled which might in turn provide information regarding how the event being studied might be inhibited or enabled, as the particular situation demands.

The scalability of this method also promises to allow examination of some very large data sets. Much effort has been spent on parallelizing the LASSO algorithm and as such, the methodology described in this paper is potentially advantageous when applied on an HPC platform, especially when the critical range conversion process is also encapsulated within the parallelization effort. There is also the added benefit that many of the operations on the data in pursuit of the LASSO solution would be suitable for optimization using integer math for all operations after the critical range transformation has been applied, and up to the final calculation of coefficients. This has the potential to greatly speed up the analysis when using HPC resources. Even without any specific accommodations for integer math, there is a noticeable speed advantage over a LASSO solution using exactly the same algorithm on the same (non-encoded) data.

REFERENCES

[1] Kawas, C., Gray, S., Brookmeyer, R., Fozard, J., & Zonderman, A. (2000). Age-specific incidence rates of Alzheimer's disease: the Baltimore Longitudinal Study of Aging. Neurology, 54(11), 2072-2077.

[2] Carlson, E. A., Alan Sroufe, L., & Egeland, B. (2004). The construction of experience: A longitudinal study of representation and behavior. Child development, 75(1), 66-83.

[3] Hiploylee, C., Dufort, P. A., Davis, H. S., Wennberg, R. A., Tartaglia, M. C., Mikulis, D., ... & Tator, C. H. (2017). Longitudinal study of postconcussion syndrome: not everyone recovers. Journal of neurotrauma, 34(8), 1511-1523.

[4] Bansal, P. (2005). Evolving sustainably: A longitudinal study of corporate sustainable development. Strategic management journal, 26(3), 197-218.

[5] McClelland, D. C. (1965). N achievement and entrepreneurship: A longitudinal study. Journal of personality and Social Psychology, 1(4), 389.

[6] Henry, C., Hill, F. M., & Leitch, C. M. (2004). The effectiveness of training for new business creation: a longitudinal study. International Small Business Journal, 22(3), 249-271.

[7] Jackofsky, E. F., & Slocum Jr, J. W. (1988). A longitudinal study of climates. Journal of organizational behavior, 9(4), 319-334.

[8] González‐Romá, V., Fortes‐Ferreira, L., & Peiró, J. M. (2009). Team climate, climate strength and team performance. A longitudinal study. Journal of occupational and organizational psychology, 82(3), 511-536.

[9] Cheng, V., Ng, E., Chan, C., & Givoni, B. (2012). Outdoor thermal comfort study in a sub-tropical climate: a longitudinal study based in Hong Kong. International journal of biometeorology, 56(1), 43-56.

[10] Bodein, A., Chapleur, O., Droit, A., & Lê Cao, K. A. (2019). A generic multivariate framework for the integration of microbiome longitudinal studies with other data types. Frontiers in genetics, 10, 963.

[11] Bodein, A., Scott-Boyer, M. P., Perin, O., Lê Cao, K. A., & Droit, A. (2022). Interpretation of network-based integration from multi-omics longitudinal data. Nucleic acids research, 50(5), e27-e27.

[12] Bosch, L., Tor, M., Reixach, J., & Estany, J. J. M. S. (2012). Age-related changes in intramuscular and subcutaneous fat content and fatty acid composition in growing pigs using longitudinal data. Meat Science, 91(3), 358-363.

[13] Leardi, R. (2001). Genetic algorithms in chemometrics and chemistry: a review. Journal of Chemometrics: A Journal of the Chemometrics Society, 15(7), 559-569.

[14] Babatunde, O. H., Armstrong, L., Leng, J., & Diepeveen, D. (2014). A genetic algorithm-based feature selection.

[15] Gascón-Moreno, J., Salcedo-Sanz, S., Saavedra-Moreno, B., Carro-Calvo, L., & Portilla-Figueras, A. (2013). An evolutionary-based hyper-heuristic approach for optimal construction of group method of data handling networks. Information Sciences, 247, 94-108.

[16] Roozbeh, M., Babaie-Kafaki, S., & Sadigh, A. N. (2018). A heuristic approach to combat multicollinearity in least trimmed squares regression analysis. Applied Mathematical Modelling, 57, 105-120.

[17] Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.

[18] Ladha, L., & Deepa, T. (2011). Feature selection methods and algorithms. International journal on computer science and engineering, 3(5), 1787-1797.

[19] Takeda, A., Niranjan, M., Gotoh, J. Y., & Kawahara, Y. (2013). Simultaneous pursuit of out-of-sample performance and sparsity in index tracking portfolios. Computational Management Science, 10(1), 21-49.

[20] Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. Journal of the royal statistical society: series B (statistical methodology), 67(2), 301-320.

[21] Duzan, H., & Shariff, N. S. B. M. (2015). Ridge regression for solving the multicollinearity problem: review of methods and models. Journal of Applied Science.

[22] Bager, A., Roman, M., Algedih, M., & Mohammed, B. (2017). Addressing multicollinearity in regression models: a ridge regression application.

[23] Kidwell, J. S., & Brown, L. H. (1982). Ridge regression as a technique for analyzing models with multicollinearity. Journal of Marriage and the Family, 287-299.

[24] Katrutsa, A., & Strijov, V. (2017). Comprehensive study of feature selection methods to solve multicollinearity problem according to evaluation criteria. Expert Systems with Applications, 76, 1-11.

[25] Freijeiro‐González, L., Febrero‐Bande, M., & González‐Manteiga, W. (2022). A critical review of LASSO and its derivatives for variable selection under dependence among covariates. International Statistical Review, 90(1), 118-145.

[26] Yuan, M., & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 68(1), 49-67.

[27] Meier, L., Van De Geer, S., & Bühlmann, P. (2008). The group lasso for logistic regression. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 70(1), 53-71.

[28] Yang, Y., & Zou, H. (2015). A fast unified algorithm for solving group-lasso penalize learning problems. Statistics and Computing, 25(6), 1129-1141.

[29] Kim, Y., Kim, J., & Kim, Y. (2006). Blockwise sparse regression. Statistica Sinica, 375-390.

[30] Wang, H., & Leng, C. (2008). A note on adaptive group lasso. Computational statistics & data analysis, 52(12), 5277-5286.

[31] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), 58(1), 267-288.

[32] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[33] Sigillito, V. G., Wing, S. P., Hutton, L. V., \& Baker, K. B. (1989). Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Technical Digest, 10, 262-266.