High dimensional Logistic Regression

Rohan Shinde

Indian Statistical Institute

rohanshinde998@gmail.com

May 18, 2023

Overview

Study of blue whales

- The deep blue sea is home to fascinating and mysterious creatures. Secrets lie within the depths waiting to be discovered.
- Whales, like the majestic Blue Whales, communicate through sound. Their vocalizations hold captivating mysteries yet to be fully understood.
- High-dimensional data holds the key to unraveling these mysteries. Advanced techniques allow us to delve into the complexity of whale vocalizations.
- Collecting and analyzing this data is a challenging task requiring expertise and specialized equipment. Understanding whale vocalizations aids conservation efforts, helping us protect these magnificent creatures
- Tackling the complexity of high-dimensional data leads to groundbreaking discoveries. Developing accurate models empowers marine biologists and environmentalists in their crucial work

Study of blue whales (Contd.)

- Blue Whale vocalization data:
  - 2 classes of audio files: **A-calls** and **non A-calls**
  - *A-calls*: Characterized by a low-frequency, repetitive pattern of pulses that are typically around 70-90 Hz in frequency; typically produced by adult males and can last for several minutes
  - The data has about 26,000 audio files out of which 13,000 are type A-calls and 13,000 are type - non A calls. The spectrogram of each audio file is then converted to a vector of length 2,30,400 consisting of pixel intensity values of the Mel-Spectrogram
  - The feature matrix we have thus is a 26000×230400 matrix with 230400 predictors and one label (whether the audio is of type A call or not) : 8.8 times more number of predictors than number of observations

What does High dimensional data mean?

Definition

High-dimensional data are defined as data in which the number of features (variables observed), p, are close to or larger than the number of observations (or data points), n.

Common in

- Audio and image data
- Sensor data: Data obtained from IoT devices
- Text data: Where each word or n-gram is different dimension
- Genomic data : Variables representing different genes and their expression levels

Fields of study with High Dimensional data

- **Audio data**: Examples of features that can be extracted:
  - Mel-Frequency Cepstral Coefficients (MFCCs): Spectral characteristics of audio signals, representing the shape of the power spectrum of the audio signal over time
  - Spectral features: Frequency content or patterns in an audio signal, e.g. power spectral density, spectral centroid, spectral contrast, or spectral roll-off
  - Mel-spectrogram: Visual representation of frequency content of audio signal; Applying mel-frequency scaling to the power spectrum of audio signal
  - Temporal features: Examples- zero-crossing rate, root mean square energy, or pitch
- **Image data**: Examples:
  - Pixel intensity values: Feature matrix contains the pixel intensity values for all pixels in the image
  - Texture Features: Spatial arrangement of pixel intensities e.g. mean, variance, entropy, etc.

Fields of study with High Dimensional data (Contd.)

- Color features: Examples- Color histograms, color moments, or color-based texture features
- Frequency domain features: Examples- Fourier transform coefficients, wavelet coefficients, etc.

- **Sensor data**:
  - Time-domain features: Characteristics in time domain e.g. minimum/maximum/amplitude of sensor measurements, rate of change or time duration of certain events
  - Frequency-domain features: Characteristics in the frequency domain e.g. power spectral density, spectral entropy, or dominant frequency
  - Autocorrelation features: Similarity or periodicity of sensor measurements over time, e.g. autocorrelation coefficients/energy/entropy.
  - Waveform-based features: Shape or waveform characteristics of sensor data, e.g. peak value, zero-crossing rate, or waveform slope

Fields of study with High Dimensional data (Contd.)

- **Text Data**:
  - Bag-of-words (BoW): Documents represented as vectors; Values represent word frequency in each document
  - Term Frequency-Inverse Document Frequency (TF-IDF): Considers term frequency (TF) as well as inverse document frequency (IDF) across the corpus; Measures term importance relative to its frequency in the corpus
  - N-grams:Contiguous sequences of N words in a text document; Useful for capturing local word order in text data.
- **Genomic data**: Examples:
  - Variant data: Presence or absence of specific genetic variants in a sample or population; measured using techniques such as genotyping arrays, whole genome sequencing (WGS), or targeted sequencing approaches.

Fields of study with High Dimensional data (Contd.)

- DNA sequences: Series of nucleotide bases
- Gene expression data: Activity levels of genes; can be measured using techniques such as RNA sequencing (RNA-seq) or microarray assays

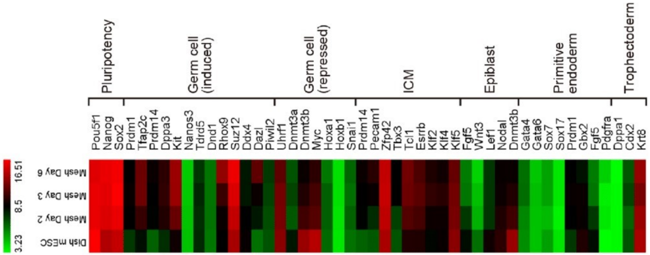

Figure: Microarray data

Why different methods for High dimensional data?

Consider the problem of linear regression:

$$\mathbf{y}_{n\times 1} = \mathbf{X}_{n\times p}\beta_{p\times 1} + \boldsymbol{\epsilon}_{n\times 1} \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}_n(\mathbf{0}_{n\times 1}, \mathrm{I_n})$$

- When $\mathbf{X}$ is random, the least squares solution to the problem is given by $\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}$
- $\hat{\beta} \sim \mathcal{N}_n(\beta, (\mathbf{X}'\mathbf{X})^{-})$
- What happens if $p > n$?
  - We have more number of variables than number of equations
  - Intuitively, we should be able to solve for β_j's certainly *but there would be infinitely many solutions* i.e. we have over-parametrized the model
  - The likelihood function may have multiple local maxima, and the optimization algorithm may converge to a sub-optimal solution
  - The validity of MLE comes into question

Example using R

R code

```
x <- matrix(rlogis(40),
         ncol = 8)
y <- rnorm(5)
model <- lm(y~x)
summary(model)
```

```
Call:
lm(formula = y ~ x)

Residuals:
ALL 5 residuals are 0: no residual degrees of freedom!

Coefficients: (4 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.3909         NA      NA       NA
x1           -0.1616         NA      NA       NA
x2           -0.3215         NA      NA       NA
x3            0.3599         NA      NA       NA
x4            0.1566         NA      NA       NA
x5                NA         NA      NA       NA
x6                NA         NA      NA       NA
x7                NA         NA      NA       NA
x8                NA         NA      NA       NA

Residual standard error: NaN on 0 degrees of freedom
Multiple R-squared:      1,     Adjusted R-squared:    NaN
F-statistic:   NaN on 4 and 0 DF,  p-value: NA
```

Figure: Output of R code

What is Logistic Regression?

Logistic Regression

In logistic regression, the conditional probability of the dependent variables (class) $y_1, y_2, \cdots, y_n \in \{0, 1\}$ are modeled as a logit-transformed multiple linear regression of the explanatory variables (input features) $\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_n} \in \mathbb{R}^p$:

$$\mathbb{P}(y_i = 1 | \mathbf{x_i}, \beta_1, \beta_2, \cdots, \beta_p) = \frac{1}{1 + \exp\left(-\mathbf{x}_i^T \boldsymbol{\beta}\right)}$$

where $\boldsymbol{\beta}' = (\beta_1 \ \ \beta_2 \ \ \cdots \ \ \beta_p)$ is the vector of parameters of the model. Assume that $y_i | \mathbf{x_i}, \beta_1, \beta_2, \cdots, \beta_p$ are independent of each other $\forall \, i \in \{1, \cdots, n\}$

Parameters are estimated using the Maximum Likelihood approach

Estimation of parameters in logistic regression

$$\begin{aligned}
\hat{\beta} &= \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmax}} \prod_{i=1}^{n} \mathbb{P}(y_i | \mathbf{x_i}, \beta) \\
&= \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmax}} \prod_{\substack{1 \le i \le n: \\ \xi_i = 1}} \mathbb{P}(y_i = \xi_i | \mathbf{x_i}, \beta) \prod_{\substack{1 \le i \le n: \\ \xi_i = 0}} \mathbb{P}(y_i = \xi_i | \mathbf{x_i}, \beta) \\
&= \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmax}} \prod_{i=1}^{n} \left(\frac{1}{1 + \exp\left(-\mathbf{x}_i^T \beta\right)} \right)^{y_i} \left(\frac{\exp\left(-\mathbf{x}_i^T \beta\right)}{1 + \exp\left(-\mathbf{x}_i^T \beta\right)} \right)^{1 - y_i}
\end{aligned}$$

- If p $\geq$ n, there exists a hyperplane in $\mathbb{R}^p$ that exactly separates the points $\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_n}$ based on their classes
- Albert et al [1] prove that in case of this separating hyperplane, the MLE estimate of β does not exist

[1] A. Albert and J. A. Anderson, On the Existence of Maximum Likelihood Estimates in Logistic Regression Models

Problems in High dimensional data

- This problem could have been tackled if we had $p < n$; So we need to reduce the number of dimensions (select variables cautiously)

Thus, we focus on the below two problems in the context of high-dimensional logistic regression:

- Variable selection: We introduce different penalties in the optimization problem to introduce *sparsity*. We discuss majorly:
  - LASSO (Least Absolute Shrinkage and Selection Operator)
  - Group LASSO: To deal with dummy variables created from categorical explanatory variables
- Statistical inference based on the variable selection method

What is regularization by penalization?

Definition[2]

Regularization methods that are derived from maximum likelihood estimates are based on the *penalized log-likelihood*:

$$\ell_p(\beta) = \sum_{i=1}^{n} \ell_i(\beta) - \lambda J(\beta)$$

where $\ell_i(\beta)$ is the usual log-likelihood contribution of the ith observation, λ is a tuning parameter, and $J(\beta)$ is a function that penalizes the size of the parameters.

Why regularization

- It is possible to increase the likelihood beyond any bound, without affecting predictive accuracy at all[3]

[2] G. Tutz, Regression for Categorical Data

[3] https://stats.stackexchange.com/a/261063

Important aspects for regression modelling by regularization

- **Existence of unique estimates**: This is where MLE's often fail
- **Prediction accuracy** is not compromised much
- **Sparseness and interpretation**

Definition[4]

A regression vector is sparse if, only some of its components are nonzero while the rest is set equal to zero, thereby inducing variable selection.

- To increase prediction accuracy in high-dimensional settings and enhance model interpretability, we prefer sparse solutions (Ballings, Van den Poel, 2015, Bertsimas, Copenhaver, 2018, Ma, Fildes, Huang, 2016, Wilms, Gelper, Croux, 2016

[4] Lea Bottmer et al, Sparse regression for large data sets with outliers

What is LASSO regression?

LASSO penalty

Originally proposed by Tibshirani (1996) for the linear model in the constrained regression version , LASSO uses the L_1 penalty:

$$J(\beta) = \sum_{j=1}^{n} |\beta_j|$$

- The log-likelihood is maximized subject to the constraint $\sum_{j=1}^{n} |\beta_j| \leq t$ for some $t \in \mathbb{R}$

Example: Consider the problem of simple linear regression $y_i = x_i\beta + \epsilon_i$ for $i \in \{1, 2, \cdots, n\}$; $y_i, x_i \in \mathbb{R} \; \forall \; i \in \{1, 2, \cdots, n\}$ where x_i's are non-random. The LASSO penalized solution to the least squares problem can be given by

$$\hat{\beta}_{\text{LASSO}} = \underset{\beta \in \mathbb{R}}{\text{argmin}} \sum_{i=1}^{n} (y_i - x_i\beta)^2 + \lambda|\beta| \quad \lambda > 0$$

Example: LASSO penalty in simple linear regression

$$\hat{\beta}_{\text{LASSO}} = \frac{\mathcal{S}_{\lambda/2}\left(\sum_{i=1}^{n} x_i y_i\right)}{\sum_{i=1}^{n} x_i^2} \text{ where}$$

$$\mathcal{S}_{\lambda}(x) = \begin{cases} x + \lambda, & \text{if } x < -\lambda \\ 0, & \text{if } |x| < \lambda \\ x - \lambda, & \text{if } x > \lambda \end{cases}$$

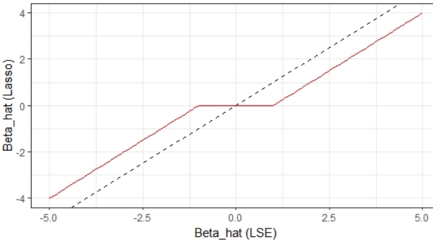

Figure: $\hat{\beta}_{\text{LASSO}}$ vs. $\hat{\beta}_{\text{LSE}}$

Regularization path for LASSO

Plot showing how the coefficients of the variables change as the regularization parameter varies

Regularization path for LASSO

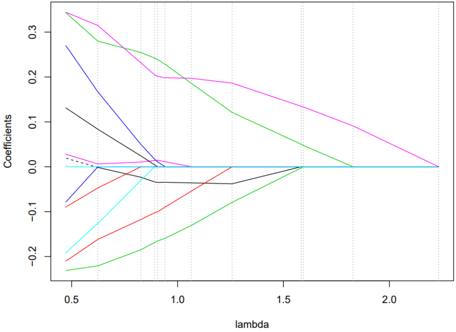

Figure: An example of the lasso regularization path (Taken from notes by Tibshirani). Each coloured line denotes a component of the lasso solution $\hat{\beta}_j(\lambda)$, $j = 1, \ldots, p$ as a function of λ

Why use LASSO in high dimensional data?

- With a large number of predictors one often wants to determine a smaller subset that contains the strongest variables : LASSO shrinks some coefficients and sets others to 0
- But if $p > n$, does LASSO even guarantee that the number of non-zero coefficient estimates is less than n? Yes it does:

Consider the more general minimization problem:

$$\hat{\beta} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\text{argmin}} \; f(\mathbf{X}\beta) + \lambda \|\boldsymbol{\beta}\|_1$$

where the loss function $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable and strictly convex.

Lemma

If $\mathbf{X} \in \mathbb{R}^{n \times p}$ has entries drawn from a continuous probability distribution on R^{np}, then for any differentiable, strictly convex function f, for any $\lambda > 0$, the minimization problem stated above has a unique solution with probability one. This solution has at most $\min\{n, p\}$ nonzero components.

Coordinate Descent for fitting LASSO penalized Logistic Regression

- The objective function for LASSO penalized negative log-likelihood of logistic model is convex and the likelihood part is differentiable, so in principle finding a solution is a standard task in convex optimization. Coordinate descent is both attractive and efficient for this problem
- The glmnet package uses a proximal-Newton iterative approach, which repeatedly approximates the negative log-likelihood by a quadratic function
- The log-likelihood of logistic regression without the lasso penalty can be given as:

$$\ell(\boldsymbol{\beta}) = \frac{1}{N}\sum_{i=1}^{N}\left[y_i(\beta_0 + \mathbf{x}_i'\boldsymbol{\beta}) - \log\left(1 + \exp\left(\beta_0 + \mathbf{x}_i'\boldsymbol{\beta}\right)\right)\right] \quad (1)$$

which corresponds to a concave function of the parameters

Detailed Coordinate Descent Algorithm

- The Newton algorithm for maximizing the (unpenalized) log-likelihood (1) amounts to iteratively reweighted least squares
- Hence, if the current estimates of the parameters are $(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}})$, we form a second-order Taylor expansion about current estimates
- In terms of the shorthand $\tilde{p}(\mathbf{x}_i) = p(\mathbf{x}_i; \tilde{\beta}_0, \tilde{\boldsymbol{\beta}})$, and $w_i = \tilde{p}(\mathbf{x}_i)(1 - \tilde{p}(\mathbf{x}_i))$, this Taylor expansion leads to the quadratic objective function:

$$\ell_Q(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}}) = -\frac{1}{2N} \sum_{i=1}^{N} w_i (z_i - \beta_0 - \mathbf{x}_i' \beta)^2 + C(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}}) \qquad (2)$$

where $z_i = \tilde{\beta}_0 + \mathbf{x}_i' \tilde{\boldsymbol{\beta}} + \dfrac{y_i - \tilde{p}(\mathbf{x}_i)}{\tilde{p}(\mathbf{x}_i)(1 - \tilde{p}(\mathbf{x}_i))}$ is the current working response.

Detailed Coordinate Descent Algorithm (Contd.)

- The Newton update is obtained by minimizing ℓ_Q, which is a simple weighted least-squares problem. In order to solve the regularized problem, one could apply coordinate descent directly to the criterion

$$\ell(\boldsymbol{\beta}) = \frac{1}{N}\sum_{i=1}^{N}[y_i(\beta_0 + \mathbf{x}_i'\boldsymbol{\beta}) - \log(1 + \exp(\beta_0 + \mathbf{x}_i'\boldsymbol{\beta}))] - \lambda P_\alpha(\boldsymbol{\beta}) \tag{3}$$

where $P_\alpha(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1$

- A disadvantage of this approach is that the optimizing values along each coordinate are not explicitly available and require a line search
- In our experience, it is better to apply coordinate descent to the quadratic approximation, resulting in a nested algorithm

Detailed Coordinate Descent Algorithm (Contd.)

- For each value of λ, we create an outer loop which computes the quadratic approximation ℓ_Q about the current parameters $(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}})$
- Then we use coordinate descent to solve the penalized weighted least-squares problem

$$\underset{(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}}) \in \mathbb{R}^{p+1}}{\text{minimize}} \{-\ell_Q(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}}) + \lambda P_\alpha(\boldsymbol{\beta})\} \tag{4}$$

- This is known as a generalized Newton algorithm, and the solution to the minimization problem (4) defines a proximal Newton map
- When $p \gg N$, one cannot run λ all the way to zero, because the saturated logistic regression fit is undefined (parameters wander off to $\pm\infty$ in order to achieve probabilities of 0 or 1)

Algorithm of Coordinate Descent

Overall the procedure consists of a sequence of nested loops:

1. OUTER LOOP: Decrement λ
2. MIDDLE LOOP: Update the quadratic approximation ℓ_Q using the current parameters $(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}})$
3. INNER LOOP: Run the coordinate descent algorithm on the penalized weighted-least-squares problem given in (4)

- The Newton algorithm is not guaranteed to converge without step-size optimization[5]. The glmnet package, which we will be using for application part in the presentation, does not implement any checks for divergence
- We have a closed form expression for the starting solutions, and each subsequent solution is warm-started from the previous close-by solution, which generally makes the quadratic approximations very accurate

[5] Lee, Lee, Abneel and Ng 2006

Shortcomings of LASSO in presence of categorical predictors?

- LASSO solution is not satisfactory as it only selects individual dummy variables instead of whole factors
- The LASSO solution depends on how the dummy variables are encoded. Choosing different contrasts for a categorical predictor will produce different solutions in general
- It is more sensible to select whole factors or continuous variables
- The group lasso proposed by Yuan and Lin (2006) can overcome these problems

Inferences for High Dimensional L1 penalized Logistic regression

- The penalized maximum likelihood estimation methods have been well developed to estimate $\beta \in \mathbb{R}^p$ in the high-dimensional logistic model (Bunea, 2008; Bach, 2010; Buhlmann and van de Geer, 2011; Meier et al., 2008; Negahban et al., 2009; Huang and Zhang, 2012)
- The penalized estimators enjoy desirable estimation accuracy properties. However, these methods do not lend themselves directly to statistical inference for the case probability mainly because the bias of the penalized estimator dominates the total uncertainty

Guo et al's method for estimating case probabilities

- Xiao Guo et al discuss a method to draw inferences by tweaking the penalized estimator to obtain optimal confidence intervals for case probabilities
- The quantity of interest is the case probability $\mathbb{P}(y_i = 1|X_i = x_*) \equiv h(x_*^T\beta)$, which is the conditional probability of $y_i = 1$ given $X_i = x_* \in \mathbb{R}^p$, where $h(z) = \dfrac{\exp(z)}{1 + \exp(z)}$
- The penalized log-likelihood estimator $\hat{\beta}$ is defined as in (3) with the tuning parameter $\lambda \asymp \sqrt{\log p/n}$

Guo et al's method for estimating case probabilities (Contd.)

- Even though $\hat{\beta}$ follows certain nice accuracy properties, the plugin estimator $h(\mathbf{x}_*^T \hat{\beta})$ cannot be directly used for confidence interval construction and hypothesis testing, because its bias can be as large as its variance [6]
- The proposed method by Guo et al. is built on the idea of correcting the bias of the plug-in estimator $x_*^T \hat{\beta}$ and then applying the h function to estimate the case probability
- We conduct the bias correction through estimating the error of the plug-in estimator $x_*^T \hat{\beta} - x_*^T \beta = x_*^T (\hat{\beta} - \beta)$

[6] Guo et al, Inference for the Case Probability in High-dimensional Logistic regression

Guo et al's method for estimating case probabilities (Contd.)

Linearization:

- A bias-corrected estimator of β_j can be constructed as

$$\hat{\beta}_j + \hat{u}^T \frac{1}{n} \sum_{i=1}^{n} [h(X_{i.}^T \hat{\beta})(1 - h(X_{i.}^T \hat{\beta}))]^{-1} X_{i.}(y_i - h(X_{i.}^T \hat{\beta})) \quad (5)$$

  where $\hat{u} \in \mathbb{R}^p$ is the projection direction used for correcting the bias of $\hat{\beta}_j$ and $X_{i.}$ is the ith row of design matrix X

- Define the error $\epsilon_i = y_i - h(X_{i.}^T \beta)$ for $1 \leq i \leq n$. Applying Taylor series expansion of h with $R_i = \int_0^1 (1-t)h''(X_{i.}^T \hat{\beta} + tX_{i.}^T(\beta - \hat{\beta}))dt \cdot (X_{i.}^T(\beta - \hat{\beta}))^2$ we get $y_i - h(X_{i.}^T \hat{\beta}) = h(X_{i.}^T \hat{\beta})(1 - h(X_{i.}^T \hat{\beta}))[X_{i.}^T(\beta - \hat{\beta}) + \Delta_i] + \epsilon_i$ with $\Delta_i = R_i / h'(X_{i.}^T \hat{\beta})$

Guo et al's method for estimating case probabilities (Contd.)

Hence the second term of eq. (5) can be decomposed as

$$\hat{u}^T \frac{1}{n}\sum_{i=1}^{n}[h(X_{i.}^T\hat{\beta})(1-h(X_{i.}^T\hat{\beta}))]^{-1}\epsilon_i X_{i.} + \hat{u}^T\hat{\Sigma}(\beta-\hat{\beta}) + \hat{u}^T\frac{1}{n}\sum_{i=1}^{n}\Delta_i X_{i.}$$

with $\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} X_{i.}X_{i.}^T$
Now for the bias correction step, the authors chose $\hat{u} \in \mathbb{R}^p$ such that $\hat{\Sigma}\hat{u} \approx e_j$ so that

$$\begin{aligned}\hat{u}^T \frac{1}{n}\sum_{i=1}^{n}[h(X_{i.}^T\hat{\beta})(1-h(X_{i.}^T\hat{\beta}))]^{-1}X_{i.}(y_i - h(X_{i.}^T\hat{\beta}))& \\ \approx \hat{u}^T\hat{\Sigma}(\beta-\hat{\beta})& \\ \approx e_j^T(\beta-\hat{\beta})& \\ = \beta_j - \hat{\beta}_j&\end{aligned}$$

Guo et al's method for estimating case probabilities (Contd.)

Variance enhancement: Uniform procedure for x_*:

- The authors correct the bias of the plug-in estimator $x_*^T\hat{\beta}$ as

$$\widehat{x_*^T\beta} = x_*^T\hat{\beta} + \hat{u}^T\frac{1}{n}\sum_{i=1}^{n}[h(X_{i.}^T\hat{\beta})(1-h(X_{i.}^T\hat{\beta}))]^{-1}X_{i.}(y_i - h(X_{i.}^T\hat{\beta}))$$

- Decompose the estimation error $\widehat{x_*^T\beta} - x_*^T\hat{\beta}$ as

$$\frac{1}{n}\sum_{i=1}^{n}[h(X_{i.}^T\hat{\beta})(1-h(X_{i.}^T\hat{\beta}))]^{-1}\epsilon_i\hat{u}^TX_{i.} + (\hat{\Sigma}\hat{u} - x_*)^T(\beta - \hat{\beta}) + \frac{1}{n}\sum_{i=1}^{n}\Delta_i\hat{u}^TX_{i.}$$

Guo et al's method for estimating case probabilities (Contd.)

Motivated by above decomposition, we construct $\hat{u} \in \mathbb{R}^p$ as the solution to the following optimization problem

$$\hat{u} = \arg\min_{u \in \mathbb{R}^p} u^T \hat{\Sigma} u \ \text{ subject to } \|\hat{\Sigma} u - x_*\|_\infty \leq \|x_*\|_2 \lambda_n \tag{6}$$

$$|x_*^T \hat{\Sigma} u - \|x_*\|_2^2| \leq \|x_*\|_2^2 \lambda_n \tag{7}$$

$$\|Xu\|_\infty \leq \|x_*\|_2 \tau_n \tag{8}$$

where $\lambda_n \asymp \sqrt{\log p/n}$ and $\tau_n \asymp \sqrt{\log n}$

Guo et al's method for estimating case probabilities (Contd.)

- Objective function in eq. (6) scaled by $1/n$, $u^T\hat{\Sigma}u$ is of the same order of magnitude as the variance of the first term in the error decomposition given at the start of variance enhancement section
- The constraints in eq. (6) and eq. (8) are introduced to control the second and third terms in the same error decomposition
- Thus objective function together with eq. (8) and eq. (8) ensure that $\widehat{x_*^T\beta} - x_*^T\hat{\beta}$ is controlled to be small
- Constraint in eq. (7) is to ensure that the first term of the decomposition is the dominant terms among the three terms in the error decomposition

Guo et al's method for estimating case probabilities (Contd.)

- In practice, instead of solving the problem in eqs. (6), (7), and (8) we solve it's dual problem

$$\hat{v} = \underset{v \in \mathbb{R}^{p+1}}{\arg\min} \ \frac{1}{4} v^T H^T \hat{\Sigma} H v + b^T H v + \lambda_n \|v\|_1$$

with $H = [b, \mathbb{I}_{p \times p}],\ b = \dfrac{x_*}{\|x_*\|_2}$

- We then solve the primal problem as $\hat{u} = -\frac{\hat{v}_{-1} + \hat{v}_1 b}{2}$
- Using the above we estimate $x_*^T \beta$ by $\widehat{x_*^T \beta}$ and subsequently we estimate the case probability $\mathbb{P}(y_i = 1 | X_{i\cdot} = x_*)$ by $\hat{\mathbb{P}}(y_i = 1 | X_{i\cdot} = x_*) = h(\widehat{x_*^T \beta})$

Inference for case probabilities

- From the above procedure, Guo et al provide an estimate for asymptotic variance of $\widehat{x_*^T\beta}$ as

$$\hat{V} = \hat{u}^T \left[\frac{1}{n^2} \sum_{i=1}^{n} [h(X_{i.}^T\hat{\beta})(1 - h(X_{i.}^T\hat{\beta}))]^{-1} X_{i.}X_{i.}^T \right] \hat{u}$$

- The authors then construct the confidence intervals for the case probability $\mathbb{P}(y_i = 1 | X_{i.} = x_*)$ as follows:

$$\mathrm{CI}_\alpha(x_*) = \left[h(\widehat{x_*^T\beta} - z_{\alpha/2}\hat{V}^{1/2}), h(\widehat{x_*^T\beta} + z_{\alpha/2}\hat{V}^{1/2}) \right]$$

  where $z_{\alpha/2}$ is the upper $\alpha/2$-quantile of the standard normal distribution

- If the goal is to test the null hypothesis $\mathrm{H}_0 : h(x_*^T\beta) < c_*$ for $c_* \in (0,1)$ we use the testing procedure $\phi_\alpha^{c_*}(x_*) = \mathbf{1}\left(\widehat{x_*^T\beta} - z_{\alpha/2}\hat{V}^{1/2} \geq h^{-1}(c_*)\right)$ which means we label the observation as case if $\widehat{x_*^T\beta} - z_{\alpha/2}\hat{V}^{1/2} \geq h^{-1}(c_*)$, as a control otherwise

Thank You

Appendix

Setup of Yuan and Lin's Group LASSO

- Let the p-dimensional predictor be structured as $\mathbf{x}_i^T = (\mathbf{x}_{i,1}^T, \cdots, \mathbf{x}_{i,G}^T)$, where $\mathbf{x}_{i,j}$ corresponds to the jth group of variables
- A group of variables may refer to the dummy variables of one factor, with df_j denoting the number of the variables in the jth group. A continuous variable that has a linear form within the predictor obviously has $\text{df}_j = 1$
- A group of variables may also refer to interactions between factors or between factors and continuous variables, where df_j is the number of individual interaction terms
- Correspondingly the parameter vector is partitioned into sub-vectors, $\beta^{\mathsf{T}} = (\beta_1^{\mathsf{T}}, ..., \beta_G^{\mathsf{T}})$

Group LASSO penalty

Penalty for Group LASSO

The group lasso uses the penalty

$$J(\beta) = \sum_{i=1}^{G} \sqrt{\beta^{\mathsf{T}} K_j \beta}$$

where K_j's are positive definite matrices. In the original paper of Yuan and Lin (2006), authors use $K_j = \mathrm{df}_j \mathrm{I}_j$ $\forall j \in \{1, \ldots, J\}$. Using these K_j's, the penalty of group LASSO is given by

$$J(\beta) = \sum_{i=1}^{G} \sqrt{\mathrm{df}_j} \|\beta_j\|_2$$

- The penalty encourages sparsity in the sense that either $\hat{\beta}_j = 0$ or $\hat{\beta}_{js} = 0$ for $s = 1, ..., \mathrm{df}_j$.

Advantages of Group LASSO over LASSO

- The group lasso can *select entire groups of variables together*, which can be useful when you have multiple variables that are related or belong to the same group and you want to either include or exclude the entire group of variables in the model
- In general, the group LASSO tends to produce *sparser models* compared to LASSO when groups of related variables are present in the data
- *Flexibility* in specifying the group structure *between groups*; groups can be predefined based on known domain knowledge
- *More interpretable models* than LASSO

Some shortcomings of Group LASSO

- Suppose we have too many categories within some categorical variable and we have encoded that variable using dummy coding
- It may well happen that only a few of those categories are actually useful for the underlying regression
- But group LASSO either includes the categorical variable or completely disregards it
- Thereby, group LASSO is not much flexible to bring sparsity within-groups

Block Coordinate Gradient Descent

The key idea of the block coordinate gradient descent method of Tseng and Yun (2006) is to combine a quadratic approximation of the log-likelihood with an additional line search. Using a second-order Taylor expansion at $\hat{\beta}^{(t)}$ and replacing the Hessian of the log-likelihood function $\ell(.)$ by a suitable matrix $H^{(t)}$ we define

$$M_{\lambda}^{(t)}(\mathbf{d}) = -\{\ell(\hat{\boldsymbol{\beta}}^{(t)}) + \mathbf{d}^T \nabla \ell(\hat{\boldsymbol{\beta}}^{(t)}) + \frac{1}{2}\mathbf{d}^T H^{(T)} \mathbf{d}\} + \lambda \sum_{g=1}^{G} \sqrt{\mathrm{df}_g} \|\hat{\boldsymbol{\beta}}_g^{(t)} + \mathbf{d}_g\|_2$$

$$\approx S_{\lambda}(\hat{\boldsymbol{\beta}}^{(t)} + \mathbf{d})$$

where $S_{\lambda}(\boldsymbol{\beta}) = -\ell(\boldsymbol{\beta}) + \lambda \sum_{g=1}^{G} \sqrt{\mathrm{df}_g} \|\boldsymbol{\beta}_g\|_2$ and $\ell(.)$ defined as in (5) and $\mathbf{d} \in \mathbb{R}^{p+1}$

Block Coordinate Gradient Descent

Now we consider the minimization of $M_\lambda^{(t)}(.)$ with respect to the gth penalized parameter group. This means that we restrict ourselves to vectors $\mathbf{d}$ with $\mathbf{d}_k = 0$ for $k \neq g$. Moreover, we assume that the corresponding $\mathrm{df}_g \times \mathrm{df}_g$ submatrix $H_{gg}^{(t)}$ is diagonal, i.e. $H_{gg}^{(t)} = h_g^{(t)} \cdot I_{\mathrm{df}_g}$ for some scalar $h_g^{(t)} \in \mathbb{R}$
If $\|\nabla\ell(\hat{\beta}^{(t)})_g - h_g^{(t)}\hat{\beta}^{(t)}\|_2 \leq \lambda\sqrt{\mathrm{df}_g}$, the minimizer of $M_\lambda^{(t)}(\mathbf{d})$ is $\mathbf{d}_g^{(t)} = -\hat{\beta}_g^{(t)}$. Otherwise

$$\mathbf{d}_g^{(t)} = -\frac{1}{h_g^{(t)}}\left(\nabla\ell(\hat{\beta}^{(t)})_g - \lambda\sqrt{\mathrm{df}_g}\frac{\nabla\ell(\hat{\beta}^{(t)})_g - h_g^{(t)}\hat{\beta}^{(t)}}{\|\nabla\ell(\hat{\beta}^{(t)})_g - h_g^{(t)}\hat{\beta}^{(t)}\|_2}\right)$$

$\mathbf{d}^{(t)} \neq \mathbf{0}$ an inexact line search using the Armijo rule has to be performed: Let $\alpha^{(t)}$ be the largest value in $\{\alpha_0 \delta^l\}_{l \geq 0}$ such that

$$S_\lambda(\hat{\beta}^{(t)} + \alpha^{(t)}\mathbf{d}) - S_\lambda(\hat{\beta}^{(t)}) \leq \alpha^{(t)} \sigma \Delta^{(t)}$$

where $0 < \delta < 1, 0 < \sigma < 1, \alpha_0 > 0$, and $\Delta^{(t)}$ is the improvement in the objective function S_λ when using a linear approximation for the log-likelihood, i.e.

$$\Delta^{(t)} = -(\mathbf{d}^{(t)})^T \nabla \ell(\hat{\beta}^{(t)}) + \lambda\sqrt{\text{df}_g}\|\hat{\beta}_g^{(t)} + \mathbf{d}_g^{(t)}\|_2 - \lambda\sqrt{\text{df}_g}\|\hat{\beta}_g^{(t)}\|$$

and we finally define $\hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} + \alpha^{(t)}\mathbf{d}^{(t)}$. The outline of the algorithm is given on the next slide.

Block Coordinate Gradient Descent for Group LASSO

Algorithm Logistic Group Lasso Algorithm (Block Coordinate Gradient Descent)

1: Let $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$ be an initial parameter vector
2: **for** $g = 0, \ldots, G$ **do**
3: $\quad H_{gg} \leftarrow h_g(\boldsymbol{\beta}) \cdot I_{\mathrm{df}_g}$
4: $\quad \mathbf{d} \leftarrow \underset{\mathbf{d} | \mathbf{d}_k = 0, k \neq g}{\text{minimize}} \, M_\lambda(\mathbf{d})$
5: $\quad$ **if** $\mathbf{d} \neq \mathbf{0}$ **then**
6: $\quad\quad \alpha \leftarrow$ Line Search
7: $\quad\quad \boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \alpha \cdot \mathbf{d}$
8: $\quad$ **end if**
9: **end for**
10: Repeat step (2)–(9) until some convergence criteria is met