William Yao
Mentor: Facheng Yu
Winter 2024

Sparse Linear Model in High Dimensions

Linear models are fundamental tools in statistical analysis and machine learning, providing a straightforward yet powerful way to understand relationships between variables. At their core, linear models predict an outcome variable as a linear combination of one or more predictor variables, offering clear interpretations through the coefficients that represent the effect of each predictor. This simplicity makes linear models a go-to method for many applications, from economics and finance to biology and engineering, where understanding the influence of variables is crucial. Despite their basic form, linear models can be adapted and extended to handle complex and high-dimensional data, maintaining their status as essential instruments in the analyst's toolbox for both explanatory and predictive tasks. Through methods like regression, these models illuminate patterns and trends in data, guiding decision-making and providing insights into the underlying processes that generate the data. The most general form of linear models is

$$\mathcal{F} = \{f : f(x) = x^T\theta, \theta \in \mathbb{R}^d\} \tag{1}$$

A common estimator of linear model is Least Squares Estimator which gives an estimated parameter with the smallest mean squared error.

$$MSE_f = E[(f(x) - y)^2] \tag{2}$$

$$f^* = argmin_{f\in\mathcal{F}}\ MSE_f = argmin_{f\in\mathcal{F}}\ E[(f(x) - y)^2] \tag{3}$$

By definition, it is easy to see that least squares estimator is also the smallest variance estimator. In applications, since we don't know the distribution of x and y, we instead use the empirical mean squared error and empirical least squares estimator.

$$\widehat{MSE}_f = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 \tag{4}$$

$$\hat{f} = argmin_{f\in\mathcal{F}}\ \widehat{MSE}_f = argmin_{f\in\mathcal{F}}\ \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 \tag{5}$$

We can get the least squares estimator of linear model by differentiating the empirical mean squared error, where

$$\hat{\theta} = (X^TX)^{-1}(X^TY) \tag{6}$$

By looking at the result, it is natural to ask whether X^TX is invertible. Unfortunately, when dealing datasets which dimension of it is significantly larger than the size of the datasets, it is not.

This is because when the dimension of data d is significantly larger than the size of the dataset n, X^TX will have rank at most n while having size of $\mathbb{R}^{d\times d}$.

This is very common when collecting data from the real world. Because of the complexity of the real world, the data collected are high-dimensional from its nature. When the data is hard to collect or when subject to study is rare, few data we can collect. This became a common problem when using naive linear models. To solve this problem, we introduce sparsity models.

The term "sparsity" indicates that the parameters are not "dense", which means that the majority of parameters are zero while only few non-zero parameters contributes to the result. The definition of sparsity can be derived from the concept of support. The support set of θ^* is defined as

$$S(\theta^*) := \{j \in \{1, \ldots, d\} : \theta_j^* \neq 0\}.$$

The hard sparsity requires the L1 norm of $S(\theta^*)$ substantially smaller than d. Under the sparsity assumption, we may have a unique linear solution of the least squares estimator.

To include hard sparsity in linear model, we can add an constraints on the L1 norm of the parameter.

$$\min_{\theta\in\mathbb{R}^d}\left\{\frac{1}{2n}\|y-\mathbf{X}\theta\|_2^2\right\} \quad \text{such that } \|\theta\|_1 \leq R \tag{7}$$

for some radius $R > 0$. However, optimization problems with constraints is still not easy to solve. We can further convert this minimization problem with constraints into a minimization problem with no constraints through Lagrangian method, which is the Lasso program.

$$\min_{\theta\in\mathbb{R}^d}\left\{\frac{1}{2n}\|y-\mathbf{X}\theta\|_2^2 + \lambda_n\|\theta\|_1\right\}. \tag{8}$$

The θ_n is a regularization parameter to be chosen by the user. When θ_n is large, the lasso program will tend to give sparser solution, while when θ_n is small, the program will tend to do more optimization on the squared error and result in less sparse solution.

Having the lasso program, we want know how well it estimates the true parameter. To do this, we still have to introduce another assumption, that is the Restricted Eigenvalue condition (RE condition). Intuitively, RE condition provides a guarantee that these algorithms can distinguish the truly important predictors from the irrelevant ones, even when the data involves many variables that are correlated with each other. Formally, the design matrix $\mathbf{X}$ satisfies the restricted eigenvalue (RE) condition over S with parameters (k, α) if

$$\frac{1}{n}\|\mathbf{X}\Delta\|_2^2 \geq k\|\Delta\|_2^2 \text{ for all } \Delta \in \mathbb{C}_\alpha(S). \tag{9}$$

where $\mathbb{C}_\alpha(S)$ represents the restricted null space.

With sparsity assumption and RE condition, the difference between any solution $\widehat{\theta}$ of the lasso program with regularization parameter $\lambda_n \geq 2\|\frac{\mathbf{X}^Tw}{n}\|_\infty$ and the true parameter θ^* has an upper bound of

$$\|\widehat{\theta} - \theta^*\|_2 \leq \frac{3}{k}\sqrt{s}\lambda_n. \tag{10}$$

There are lots of applications of lasso program. Since the lasso program constrains the number of non-zero parameters, one of the common application is feature selection. A widely used procedure of selecting effective and efficient features for models is through data augmentation and lasso program. There are techniques like polynomial data augmentation and other pre-processing tools that can generate over thousands of features, and lasso program are used to select few features that are actually contributing to the final prediction. By using selected features over original features, models usually performs better with higher accuracy of prediction.