LASSO Regression

A Convex Optimization Approach for High-dimensional Problems

Wooseok Song

1 Abstract

'High dimensional' is the case when the number of unknown parameters is larger than observations. Previous statistical methods assume that the observations are larger than the number of unknown parameters. We show that LASSO(Least Absolute Shrinkage and Selection Operator) method can get attention in solving high dimensional problem

Contribution

1. Provide introduction of LASSO as a linear regression model , constrained quadratic programming problem.

2. Discuss convex optimization based approach to solve LASSO problem.

3. show applications of LASSO using a simulated and a real data examples

2 Introduction

2.1 Motivation

We consider underdetermined linear model :

$$\mathbf{Y} = \mathbf{X}\beta^0, \qquad (1)$$

where Y $\in R^n$ is a response vector, X $\in R^{n\,x\,p}$ is a design matrix and $\beta^0 \in \mathrm{R}^p$ is a vector of unknown true regression coefficients.

For p > n , But it is impossible to identify the correct solution from the infinite solution set without some additional information or constraints. So we give constraints.

L0-norm (non convex) :

$$\underset{\beta\in\mathbb{R}^p}{minimize}\ \|\beta\|_0 \text{ such that } \mathbf{Y} = \mathbf{X}\beta. \qquad (2)$$

L1-norm (convex)

$$\underset{\beta \in \mathbb{R}^p}{\textit{minimize}} \ \|\beta\|_1 \text{ such that } \mathbf{Y} = \mathbf{X}\beta. \qquad (3)$$

2.2 Lasso

I. L1 regularization approximates L0 regularization (best subset selection).

II. Shrinkage (if done properly) helps to improve prediction performance.

$$\hat{\beta} := \hat{\beta}(\lambda) = \underset{\beta \in \mathbb{R}^p}{\arg\min} \left\{ \frac{1}{n} \|\mathbf{Y} - \mathbf{X}\beta\|_2^2 + \lambda \|\beta\|_1 \right\}, \qquad (6)$$

Where ($\lambda \geq 0$) is the regularization parameter that controls the amount of shrinkage.

3 Background

3.1 Convexity

3.1.1 Affine functions

Let $A(x): R^n \to R^m$ *is affined if there exists a linear function* $L: R^n \to R^m$ *and a vector* $y \in R^m$ *such that A(x)=L(x)+y*

3.1.2 Convex sets

set C is said to be convex, if it contains the line segments between any two of its points, that is:

λx + (1 − λ)y ∈ C, for all x, y ∈ C, and for all λ ∈ [0, 1]

3.1.3 Convex functions

A function f is convex if its domain, D(f), is a convex set and the following holds: f(λx + (1 − λ)y) ≤ λ f(x) + (1 − λ)f(y), for all x, y ∈ D(f), for all λ ∈ [0, 1]. (12) If the above definition (12) holds with strict inequality for x y and for all λ ∈ (0, 1), then f is strictly convex

3.1.4 Sub-level sets

The level set of a function $f : R^n \to R$ at level c is the set of points

$S = \{x: f(x) = c\}$

For $f : R^2 \to R$, we are usually interested in S when it is a curve. For $f : R^3 \to R$,

the sets S most often considered are surfaces

(Sub-level Sets) The α-sublevel set of a function f , is the set of all points x such that f(x) ≤ α, where α ∈ R

3.1.5 First order condition for Convexity

If a function f is differentiable, then f is convex if and only if, D(f) is a convex set and for all x, y ∈ D(f) the following holds: f(y) ≥ f(x) + $\nabla f(x)^T$ (y − x)

3.1.6 Sub-gradients

A sub-gradient of a convex function f at x is any g ∈ R^p such that the following holds: f(y) ≥ f(x) + R^T (y − x), for all y.

3.1.7 Sub-differentials

Set of all sub-gradients of a convex function f at x is called the sub-differential of f at x, and it is denoted as: δ f(x) = { g ∈ R^T: g is a sub-gradient of f at x }

3.1.8 Convex Optimization Problems

A convex optimization problem is an optimization problem of the form:

minimize f(x)

subject to g_i(x) ≤ 0, i = 1, ..., m

h_i((x) = 0, i = 1, ...,r (16) where x ∈ R^P is the optimization variable, f and g_i are convex functions for all i = 1, ..., m and h_i are affine functions for all i = 1, ...,r. If there are no constraints, m = r = 0, it is called unconstrained convex optimization problem.

3.1.9 Lagrange Duality

C is primal feasible set , $f^* \; primal \; optimization$

$$f^* \geq \min_{x \in C} L(x,u,v) \geq \min_{x} L(x,u,v) := g(u,v)$$

3.1.10 Weak and Strong Duality

Weak : x is primal optimal solution and (u,v) is the a dual optimal solution .

f(x) ≥ l(u , v)

Strong : f(x) = l(u , v).

3.1.11 KKT condition

1. $\boldsymbol{\mu}^* \geq \mathbf{0}$;
2. $Df(\boldsymbol{x}^*) + \boldsymbol{\lambda}^{*T} D\boldsymbol{h}(\boldsymbol{x}^*) + \boldsymbol{\mu}^{*T} D\boldsymbol{g}(\boldsymbol{x}^*) = \mathbf{0}^T$;
3. $\boldsymbol{\mu}^{*T} \boldsymbol{g}(\boldsymbol{x}^*) = 0$;
4. $\boldsymbol{h}(\boldsymbol{x}^*) = \mathbf{0}$;
5. $\boldsymbol{g}(\boldsymbol{x}^*) \leq \mathbf{0}$.

3.2 Optimization Techniques

3.2.1 Gradient Descent Methods

$$x^{(k)} = x^{(k-1)} - t_k \nabla f(x^{(k-1)}), k = 1,2,3,\ldots$$

3.2.2 Sub-gradient Method

Unlike gradient descent, the subgradient method does not descent (why it is not called subgradient "descent"). So you can use any trial (iteration) where you can use the subgradient method.

$$f(x_{best}^{(k)}) = \min_{i=0,\ldots k} f(x^{(i)})$$

3.2.3 Coordinate Descent Method

For k=1,2,3,...,

$$\begin{aligned}
x_1^{(k)} &\in \arg\min_{x_1} f(x_1, x_2^{(k-1)}, x_3^{(k-1)}, \ldots, x_n^{(k-1)}) \\
x_2^{(k)} &\in \arg\min_{x_2} f(x_1^{(k)}, x_2, x_3^{(k-1)}, \ldots, x_n^{(k-1)}) \\
x_3^{(k)} &\in \arg\min_{x_3} f(x_1^{(k)}, x_2^{(k)}, x_3, \ldots, x_n^{(k-1)}) \\
&\ldots \\
x_n^{(k)} &\in \arg\min_{x_n} f(x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, \ldots, x_n)
\end{aligned}$$

3.3 Least Square Regression

$$\hat{\beta}_{OLS} := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{n} \|\mathbf{Y} - \mathbf{X}\beta\|_2^2 \right\}.$$

3.4 The variable Selection problem

Subset selection is an important issue particularly when p is large and it is believed that many covariates are redundant or irrelevant. In the context of linear regression, the subset selection problem is to select and fit a model of the form:

$$\mathbf{Y} = \mathbf{X}_S\beta_S + \epsilon,$$

4 LASSO

Lasso regression performs L1 regularization, which adds a penalty equal to the absolute value of the magnitude of coefficients. This type of regularization can result in sparse models with few coefficients; Some coefficients can become zero and eliminated from the model. Larger penalties result in coefficient values closer to zero, which is the ideal for producing simpler models. On the other hand, L2 regularization (e.g. Ridge regression) *doesn't* result in elimination of coefficients or sparse models. This makes the Lasso far easier to interpret than the Ridge.

Performing the Regression
Lasso solutions are quadratic programming problems, which are best solved with software.
The goal of the algorithm is to minimize:

$$\sum_{i=1}^{n}(y_i - \sum_j x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

Some of the βs are shrunk to exactly zero, resulting in a regression model that's easier to interpret.
A tuning parameter, λ controls the strength of the L1 penalty. λ is basically the amount of shrinkage:

5 Experiment

5.1 Generate data

I made ill-conditioned dataset. (unknown>observation)

Y ∈ R^{20}

$X \in R^{20*500}$

Then, $\beta\ will\ satisfies$ ($\beta \in R^{500}$)

5.2 Fit the lasso model with various lambda(0~1)

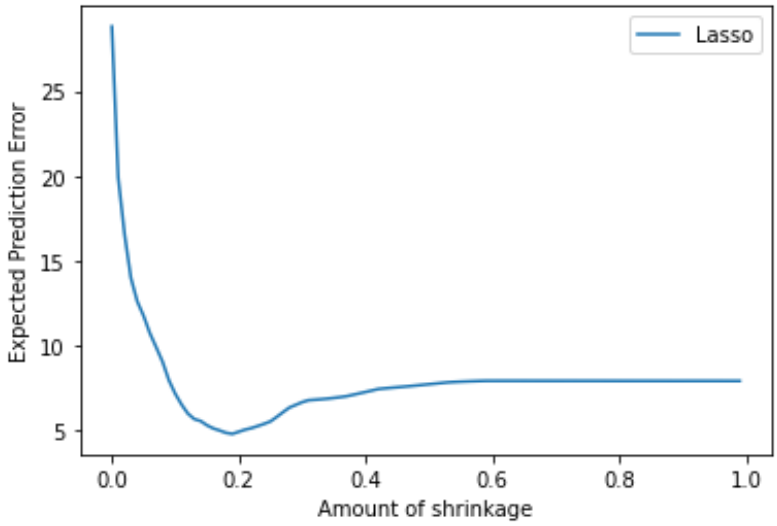
5.3 Calculate expected precision error with each lambda

5.4 Get the lambda which makes the error minimum

5.5 Get Beta of the result of the regression

6 Result

6.1 Expected precision error depending on lambda



6.2 The value of optimized lambda(amount of shrinkage) is 0.19

6.3 As the result of LASSO regression, The number of zero components of Beta matrix is 481.

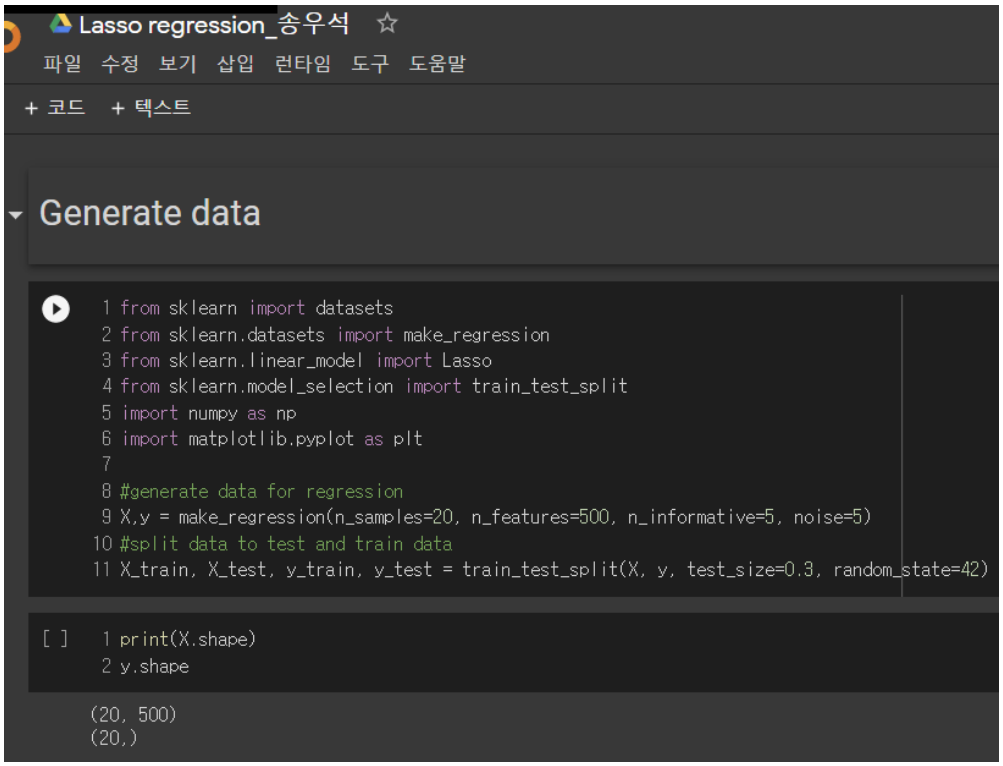
7 Conclusion

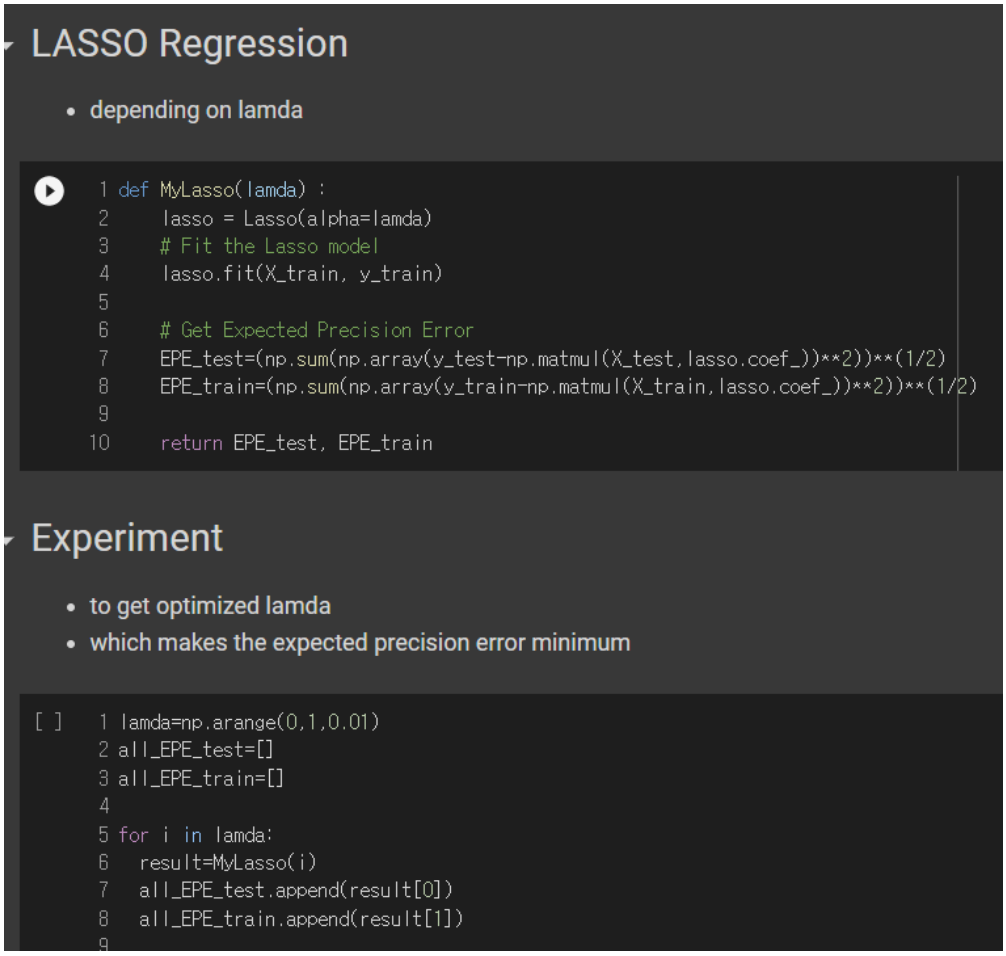
The purpose of this paper is introduction of LASSO regression and aim of the experiment was to get the optimized amount of shrinkage coefficients depends on the data. As a result, We can get the coefficients that makes Beta sparse!

8 References

[1] Introduction to the LASSO, Niharika Gauraha ,2018

9 Appendix


Lasso regression_송우석
파일 수정 보기 삽입 런타임 도구 도움말
+ 코드 + 텍스트
Generate data
1 from sklearn import datasets
2 from sklearn.datasets import make_regression
3 from sklearn.linear_model import Lasso
4 from sklearn.model_selection import train_test_split
5 import numpy as np
6 import matplotlib.pyplot as plt
7
8 #generate data for regression
9 X,y = make_regression(n_samples=20, n_features=500, n_informative=5, noise=5)
10 #split data to test and train data
11 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
[ ] 1 print(X.shape)
2 y.shape
(20, 500)
(20,)


LASSO Regression
• depending on lamda
1 def MyLasso(lamda) :
2 lasso = Lasso(alpha=lamda)
3 # Fit the Lasso model
4 lasso.fit(X_train, y_train)
5
6 # Get Expected Precision Error
7 EPE_test=(np.sum(np.array(y_test-np.matmul(X_test,lasso.coef_))**2))**(1/2)
8 EPE_train=(np.sum(np.array(y_train-np.matmul(X_train,lasso.coef_))**2))**(1/2)
9
10 return EPE_test, EPE_train
Experiment
• to get optimized lamda
• which makes the expected precision error minimum
[ ] 1 lamda=np.arange(0,1,0.01)
2 all_EPE_test=[]
3 all_EPE_train=[]
4
5 for i in lamda:
6 result=MyLasso(i)
7 all_EPE_test.append(result[0])
8 all_EPE_train.append(result[1])
9

Result

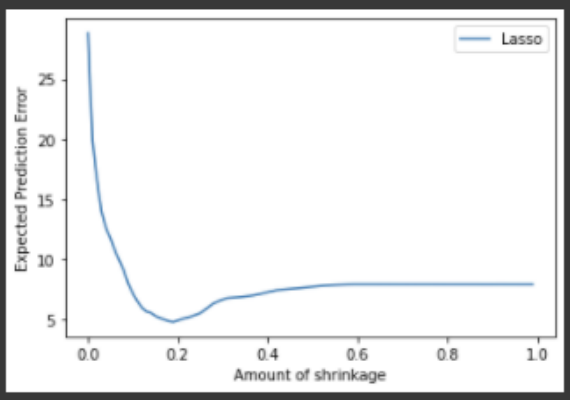
- Optimized lambda
- sparse Beta

```
plt.plot(lamda,np.array(all_EPE_test)/len(y),label='Lasso')
#plt.plot(lamda,np.array(all_EPE_train)/len(y),label='train')
plt.legend()
plt.xlabel('Amount of shrinkage')
plt.ylabel('Expected Prediction Error')
plt.show()
print("")
print("The lamda is ",lamda[np.argmin(all_EPE_test)])

lasso = Lasso(alpha=lamda[np.argmin(score)])
    #
    # Fit the Lasso model
    #
lasso.fit(X_train, y_train)
count=0
for i in range(len(lasso.coef_)):
  if lasso.coef_[i] ==0 :

    count=count+1

print("As the result of LASSO , The number of Zero components of Beta is ",count)
```




```
The lamda is  0.19
As the result of LASSO , The number of Zero components of Beta is  481
```