Regressão Lasso

Guilherme Seidyo Imai Aldeia

Universidade Federal do ABC
Programa de Pós Graduação em Ciência da Computação
Fundamentos de Matemática para Computação
Prof. Dr. Saul de Castro Leite

Santo André/SP
2020

Índice

1. Introdução
2. Regressão Linear
3. Regularização
4. Implementação
5. Teste

Artigo de referência

- Este projeto foi baseado no artigo **Regression Shrinkage and Selection via the Lasso** ↗ , publicado em 1996 por *Robert Tibshirani*.
- As referências secundárias (sites com discussões interessantes), que ajudaram a entender melhor o artigo principal, serão apresentadas como leitura complementar.

Índice

1. Introdução
2. Regressão Linear
3. Regularização
4. Implementação
5. Teste

Revisando a regressão linear I

Técincas de regressão

Suponha que exista uma função $f(\mathbf{x}) = y$ desconhecida, que descreve a resposta - $y \in \mathbb{R}$, variável dependente (variável alvo) em função de ou mais variáveis independentes (variáveis explanatórias) $\mathbf{x} \in \mathbb{R}^n$ — a regressão busca ajustar uma função $\hat{f}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, minimizando a distância (ou o erro) entre essa função e os dados observados.

Geralmente, os dados são obtidos de experimentos e observações, e deseja-se modelar uma função que descreva o comportamento observado.

Revisando a regressão linear II

A regressão linear parte de uma função linear, e o ajuste envolve encontrar valores para os coeficientes livres (β_i, com $i = 1, 2, \ldots, n$), associados às n variáveis explanatórias de mesmo índice.
Chamaremos de $\widehat{f}$ a função estimada, que é definida por:

$$\widehat{f}(\mathbf{x}) = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n = \widehat{y},$$

ou, em forma de vetores:

$$\widehat{y} = \mathbf{x}^T \beta,$$

Revisando a regressão linear III

Geralmente, o erro é medido pela Soma dos Resíduos ao Quadrado (RSS, da sigla em inglês):

$$\text{RSS} = \sum_{i=1}^{n} (\widehat{y}_i - y_i)^2 = \sum_{i=1}^{n} (\mathbf{x}_i^T \beta - y_i)^2$$

onde $\widehat{y}_i - y_i$ é o resíduo ϵ. Para ajustar $\widehat{f}$, queremos encontrar um vetor β ótimo ($\widehat{\beta}$) que minimize o RSS. Queremos então:

$$\widehat{\beta} = min \; \text{RSS}(\beta)$$

Revisando a regressão linear IV

Ou seja, precisamos derivar RSS(β) e igualar a zero:

$$\frac{d}{d\beta}\text{RSS}(\beta) = 0,$$

Existe uma forma de derivação desse problema, chamada de **Método dos Mínimos Quadrados** (OLS, da sigla em inglês). É dessa forma que obtemos a solução na forma fechada apresentada à seguir .

Revisando a regressão linear V

O método dos mínimos quadrados nos dá:

$$\widehat{\beta} = (X^T X)^{-1} X^T y.$$

onde X e y são o conjunto de todos os pontos observados:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,n} \\ x_{2,1} & x_{2,2} & \dots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{d,1} & x_{d,2} & \dots & x_{d,n} \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_d \end{bmatrix}.$$

Revisando a regressão linear VI

A regressão linear apresenta bons resultados quando as variáveis explanatórias e a variável alvo interagem de forma linear. Porém, em alguns casos, pode ser de interesse evitar alguns problemas como o *overfit* nos dados.

Índice

1. Introdução
2. Regressão Linear
3. Regularização
4. Implementação
5. Teste

Regularização I

A regularização é o processo de se adicionar informações para resolver um problema de regressão, buscando prevenir o *overfit*.

Regularização II

Até agora, estamos encontrando a solução dos coeficientes da regressão linear resolvendo o seguinte problema:

$$\widehat{\beta} = min(\sum_{i=1}^{n}(\widehat{y}_i - y_i)^2).$$

Vamos adicionar um termo regularizador na nossa função, chamado de $R(f)$:

$$\widehat{\beta} = min(RSS + R(f)).$$

Agora, precisamos escolher o termo regularizador $R(f)$.

Norma

Antes de continuar, vamos recordar o conceito de **norma**. Toda função $N : \mathbb{R}^n \to \mathbb{R}$, que satisfaz as condições:

1. $\|\alpha x\| = |\alpha| \, \|x\|$;
2. $\|x\| \geq 0$, para todo $x \in \mathbb{R}^n$, e $\|x\| = 0 \Leftrightarrow x = 0$;
3. $\|x + y\| \leq \|x\| + \|y\|$ (inequação do triângulo);

é uma norma. De forma geral, temos uma norma-p definida como:

$$\|x\|_p = (\sum_{i=1}^{n} |x_i|^p)^{\frac{1}{p}},$$

onde $p \in [1, \infty)$.

Regularizando a regressão linear I

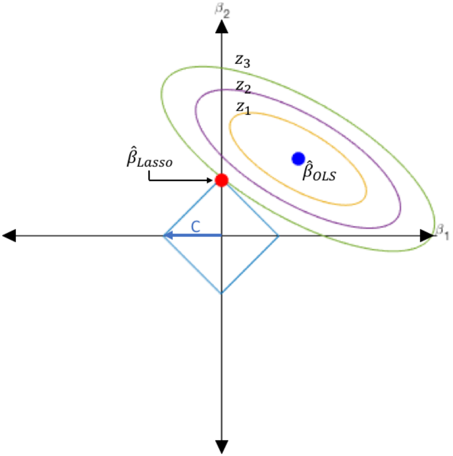
Seja a penalidade sobre β dada por $\lambda \|\beta\|_1$, com λ sendo o coeficiente de penalidade (ou parâmetro de regularização), temos agora que nosso $\hat{\beta}$ será calculado por:

$$\widehat{\beta} = min(\text{RSS} + \lambda \|\beta\|_1) = min(\sum_{i=1}^{n} (y_i - \beta \mathbf{x}_i^T)^2 + \lambda \sum_{j=1}^{n} |\beta_j|).$$

Na literatura, utilizar como penalidade a norma l_1 para a tarefa de regressão recebe um nome especial: Regressão Lasso (!)

Justificativa do uso da norma l_1 I

A adição do termo de penalidade gera uma restrição no espaço de busca, com forma de "diamante".



Justificativa do uso da norma l_1 II

Dado que λ seja adequado, $\widehat{\beta}$ ocorre na intersecção entre as duas regiões.
O ponto ótimo cairá em um dos vértices do "diamante", resultando em uma ou mais variáveis explanatórias com coeficientes iguais a zero, enquanto o OLS pode fazer com que os coeficientes assumam valores muito próximos de - mas não necessariamente - zero.

Implicações do termo regularizador

A restrição imposta ainda compartilha resultados bons com o OLS, mas limita os resultados à região onde soluções mais simples existem.
O Lasso produz soluções esparsas, onde vários parâmetros de $\widehat{\beta}$ são iguais a zero.
Com isso, **o Lasso pode resultar em menos variáveis utilizadas na equação, aliviar o overfit, ou melhorar a qualidade e interpretabilidade do resultado**.

Limitações

1. Quando $d \gg n$, o lasso selecionará até n variáveis antes de saturar, por conta do problema convexo de otimização;
2. O lasso assume independência entre variáveis. No caso onde há variáveis correlacionadas, ele selecionará apenas 1 desse grupo e descarta os outros.

Leituras complementares

- Definição formal e propriedades do Lasso. Unicidade de solução ↗ ;
- Resumo de passos importantes. Convergência ↗ ;
- Lasso para seleção de variáveis ↗ ;
- Intuições sobre o Lasso ↗ ;
- Desvantagens do Lasso ↗ .

Índice

1. Introdução
2. Regressão Linear
3. Regularização
4. Implementação
5. Teste

Nova função de custo

Vamos definir L_1 como a função de custo da regressão linear com regularização l_1:

$$L_1 = \sum_{i=1}^{n}(y_i - \beta \mathbf{x}_i^T)^2 + \lambda \sum_{j=1}^{n} |\beta_j| \, .$$

Note que, neste caso, L_1 não é diferenciável, pois $|\beta|$ não é contínua em 0.

Convexidade da nova função de custo

O termo da regularização não é estritamente convexo, podendo existir vários valores de $\widehat{\beta}$ que minimizem a função de erro. Por isso, costumam ser utilizados métodos de otimização numérica ↗ para resolver o problema de regressão com penalidade l_1.

Implementação - visão geral

Por ser uma função de custo convexa, a otimização parâmetro-por-parâmetro converge para o ótimo.
Podemos iterar sobre β, um a um, fazendo uma atualização a cada passo (como se trabalhássemos no caso unidimensional), até chegar no ponto de convergência. O *coordinate descent* funciona assim, e tem como vantagens:

1. Não envolve fatorações da matriz ou operações complicadas, apenas produtos internos;
2. Escala de forma linear para n e d.

Algoritmo de *coordinate descent* I

Algoritmo 1: Algoritmo geral de *coordinate descent* para ajuste do $\widehat{\beta}$

```
β = [1, 1, ..., 1] ;
// para t iterações
for t=0, 1, 2, ... do
    // para cada β_j
    for j=1, 2, ..., n do
        β^{t+1}[j] = min F(β) à respeito de β_j ;
        β^{t+1}[i] = β[i], para todo i ≠ j ;
        β = β^{t+1} ;
    end
end
return β ;
```

Algoritmo de *coordinate descent* II

Note que, para implementar o algoritmo anterior, precisamos saber quem é o mínimo da função de custo F. Como no nosso problema a função de custo é:

$$F(\beta) = L_1 = \sum_{i=1}^{n}(y_i - \beta x_i^T)^2 + \lambda \sum_{j=1}^{n} |\beta_j| \, ,$$

com $\beta \in \mathbb{R}^n$, a iteração $t+1$ define β^{t+1} à partir de β^t, ajustando um parâmetro por vez, minimizando a função de custo em relação a β_j.

Algoritmo de *coordinate descent* III

Note que $L_1 = OLS + \lambda \|\beta\|$. Pelo termo OLS, derivando parcialmente para um β_j:

$$\frac{d}{d\beta_j} OLS(\beta) = \sum_{i=1}^{d} x_j^{(i)} \left[y^{(i)} - \sum_{k=1}^{n} \beta_k x_k^{(i)} \right].$$

Podemos tirar o caso $k = j$ de dentro do último somatório:

$$\frac{d}{d\beta_j} OLS(\beta) = \sum_{i=1}^{d} x_j^{(i)} \left[y^{(i)} - \sum_{k=1, k \neq j}^{n} \beta_k x_k^{(i)} - \beta_j x_j^{(i)} \right].$$

Algoritmo de *coordinate descent* IV

Sabemos que não há derivada da penalidade Lasso, onde $\beta_j = 0$, pois:

$$\lambda \|\beta\| = \lambda |\beta| = \lambda |\beta_j| + \lambda \sum_{k=1, k \neq j}^{n} |\beta_k| .$$

Para derivar essa parte da equação, vamos utilizar os conceitos de subderivadas, uma forma de generalizar a derivada de funções convexas que não são necessariamente diferenciáveis.

Algoritmo de *coordinate descent* V

Temos 3 propriedades importantes de subderivadas:

1. Uma função convexa em x_0 só é diferenciável nesse ponto se o conjunto de subderivada só tem um ponto, que será a derivada em x_0;
2. (Teorema Morea-Rockafellar) Sejam f e g convexas com subderivadas, então a subderivada de $f + g$ é $\partial(f + g) = \partial f + \partial g$
3. (condição estacionária) Um ponto x_i é o mínimo global de f se e somente se o 0 está contido no conjunto de subderivadas.

Algoritmo de *coordinate descent* VI

Com essas propriedades, podemos calcular a subderivada para o termo Lasso, como:

$$\frac{\partial}{\partial \beta_j}(\lambda|\beta_j|) = \begin{cases} -\lambda, \text{ se } \beta_j < 0 \\ [-\lambda, \lambda], \text{ se } \beta_j = 0 \\ \lambda, \text{ se } \beta_j > 0 \end{cases},$$

Note que o segundo caso irá conter, dentro do intervalo, o valor 0.

Algoritmo de *coordinate descent* VII

Dessa forma, sabemos agora calcular a subderivada de L_1 para um dado β_j:

$$\frac{\partial}{\partial \beta_j} L_1(\beta) = \frac{\partial}{\partial \beta_j} OLS(\beta) + \frac{\partial}{\partial \beta_j} \lambda \left\| \beta \right\|_1 .$$

Como queremos o mínimo global da função de custo:

$$\sum_{i=1}^{d} x_j^{(i)} \left[y^{(i)} - \sum_{k=1, k \neq j}^{n} \beta_k x_k^{(i)} - \beta_j x_j^{(i)} \right] + \frac{\partial}{\partial \beta_j} \lambda \left\| \beta \right\|_1 = 0.$$

Algoritmo de *coordinate descent* VIII

Sejam ρ_j a derivada da função de custo do OLS, e Z_j o fator de normalização dos dados, definidos por:

$$\rho_j = \sum_{i=1}^{d} x_j^{(i)} \left[y^{(i)} - \sum_{k=0, k\neq j}^{n} \beta_k x_k^{(i)} - \beta_j x_j^{(i)} \right],$$

$$Z_j = \sum_{i=1}^{d} (x_j^{(i)})^2,$$

Dessa forma, utilizando as propriedades de subderivadas, e combinando as derivadas do OLS e do termo Lasso, temos:

$$\frac{\partial}{\partial \beta_j} L_1(\beta) = \begin{cases} \rho_j - \lambda, \text{ se } \beta_j < 0 \\ 0, \text{ se } \beta_j = 0 \\ \rho_j + \lambda, \text{ se } \beta_j > 0 \end{cases},$$

Algoritmo de *coordinate descent* IX

Lembrando da função de custo:

$$\frac{d}{d\beta_j} L_1(\beta) = \sum_{i=1}^{d} x_j^{(i)} \left[y^{(i)} - \sum_{k=1, k \neq j}^{n} \beta_k x_k^{(i)} - \beta_j x_j^{(i)} \right] + \frac{\partial}{\partial \beta_j} \lambda \left\| \beta \right\|_1 = 0,$$

podemos isolar o termo com β_j:

$$\sum_{i=1}^{d} x_j^{(i)} \beta_j x_j^{(i)} = \sum_{i=1}^{d} x_j^{(i)} \left[y^{(i)} - \sum_{k=1, k \neq j}^{n} \beta_k x_k^{(i)} \right] + \frac{\partial}{\partial \beta_j} \lambda \left\| \beta \right\|_1 .$$

Dividindo os dois lados pelo termo que multiplica β_j:

$$\beta_j = \frac{\sum_{i=1}^{d} x_j^{(i)} \left[y^{(i)} - \sum_{k=1, k \neq j}^{n} \beta_k x_k^{(i)} \right] + \frac{\partial}{\partial \beta_j} \lambda \left\| \beta \right\|_1}{\sum_{i=1}^{d} x_j^{(i)} x_j^{(i)}} = \frac{\rho_j + \lambda}{Z_j}.$$

Algoritmo de *coordinate descent* X

Agora que sabemos calcular um valor para β_j, podemos ver quando ele cairá em cada um dos casos. Note que isso depende então de ρ_j, Z_j, e λ. Podemos então definir a atualização de β_j por:

$$\begin{cases} \beta_j = \frac{\rho_j + \lambda}{Z_j}, \text{ se } \rho_j < -\lambda \\ \beta_j = 0, \text{ se } -\lambda < \rho_j < \lambda \\ \beta_j = \frac{\rho_j - \lambda}{Z_j}, \text{ se } \rho_j > \lambda \end{cases}.$$

Sendo assim podemos definir uma função $S(\rho_j, \lambda, Z_j)$, chamada de *Soft Threshold*, que faz a computação acima.

Algoritmo de *coordinate descent* XI

Algoritmo 2: Algoritmo final de *coordinate descent* para ajuste do $\widehat{\beta}$

$\beta = [1, 1, ..., 1]$;
for *t=0, 1, 2, ...* **do**
  for *j=1, 2, ..., n* **do**
    calcule ρ_j utilizando β ;
    calcule Z_j utilizando β ;
    $\beta[j] = S(\rho_j, \lambda, Z_j)$;
  end
end

Leituras complementares

- Como implementar o Lasso com *coordinate descent* ;
- Discussão sobre convergência do método ;
- Como os coeficientes respondem de acordo com o parâmetro de regularização ;
- **Regularization Path For Generalized linear Models by Coordinate Descent**, *Friedman, Hastie & Tibshirani, J Stat Softw*, 2010;
- **An Interior-Point Method for Large-Scale L1-Regularized Least Squares**. **S. J. Kim, K. Koh, M. Lustig, S. Boyd and D. Gorinevsky**, in IEEE Journal of Selected Topics in Signal Processing, 2007.

Índice

1. Introdução
2. Regressão Linear
3. Regularização
4. Implementação
5. Teste

Executando um pequeno teste

Conclusões

O Lasso pode ajudar a diminuir o *overfit* e melhorar os resultados por restringir o espaço de busca a soluções esparsas. Isso também pode gerar um aumento na interpretabilidade do resultado obtido, por diminuir a quantidade de termos utilizados.