# 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

# Introdução

- Durante o curso, aprendemos sobre a regressão linear, e vimos diferentes formas fechadas de calcular sua solução.
- Nesta apresentação, iremos estudar uma variação: **a Regressão Lasso**.
- Veremos a fundamentação matemática e uma implementação do método.

# 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

# Revisando a regressão linear I

## Técincas de regressão

Ajustam os coeficientes de uma função matemática,mapeando uma ou mais variáveis $x$, com $x \in \mathbb{R}$ ou $\mathbf{x} \in \mathbb{R}^n$ (chamadas de variáveis explanatórias ou variáveis independentes) a uma variável $y$ (variável alvo, ou variável dependente), $y \in \mathbb{R}$.

# Revisando a regressão linear II

Podemos ter o caso onde há apenas uma variável explanatória ($x \in \mathbb{R}$); ou o caso com $n$ variáveis explanatórias (onde $\mathbb{R}^n$, $\mathbf{x}$ é um vetor de tamanho $n$, onde cada elemento $x_i$ corresponde a uma variável explanatória).

## Definindo uma convenção

No restante das explicações que seguem, será tratado sempre o caso em que $\mathbf{x} \in \mathbb{R}^n$. Utilizaremos esta análise pois as conclusões para $\mathbb{R}^n$ também são válidas para o caso onde $n = 1$, isto é, $\mathbb{R}$, sem perda de generalidade.

# Revisando a regressão linear III

Suponha que exista uma função $f(\mathbf{x}) = y$ desconhecida — a tarefa de regressão é ajustar uma função $\hat{f}(\mathbf{x})$, com $\hat{f}(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$, através da escolha de coeficientes internos de $\hat{f}$ de forma a minimizar uma função de custo entre os valores reais $y$ observados com os valores preditos $\hat{y} = \hat{f}(\mathbf{x})$ para cada par $(\mathbf{x}, y)$ de pontos dados.
Geralmente, esses pares de pontos são obtidos de experimentos e observações, e deseja-se modelar uma função que descreva o comportamento observado.

# Revisando a regressão linear IV

Existem, 3 tipos de regressões:

- **Paramétricas:** o caso que $\hat{f}$ é definida previamente, e ajustamos apenas coeficientes livres dessa função (i.e. regressão linear);
- **Não paramétricas:** quando tanto os coeficientes livres quanto a própria função são ajustados durante o processo (i.e. regressão simbólica);
- **Semi-paramétricas:** quando a função apresenta uma parte fixa (pré definida) e uma parte ajustável nos mesmos moldes que regressões não paramétricas (i.e. modelo geoaditivo).

# Revisando a regressão linear V

A regressão linear é paramétrica, e sua solução vem do ajuste de coeficientes livres ($\beta_i$, com $i = 1, 2, \ldots, n$) da combinação linear das $n$ variáveis explanatórias do problema, e costuma incluir também um coeficiente não associado às variáveis, representando o intercepto ($\beta_0$).
Sendo assim, queremos estimar valores para os coeficientes livres de forma a minimizar o erro.

# Revisando a regressão linear VI

Seja o conjunto de $d$ pontos observados $\{\mathbf{x_i}, y_i\}_{i=1,2,\ldots,d}$. Partimos da suposição de que a função $f$ realiza a combinação linear das $x_i$ variáveis dependentes, com $i = 1, 2, \ldots, n$ sendo que cada variável $x_i$ tem um peso associado. Chamaremos de $\widehat{f}$ a função estimada, que é definida por:

$$\widehat{y} = \widehat{f}(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n,$$

ou, em forma de vetores:

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

onde $\beta_0$ é o intercepto.

# Revisando a regressão linear VII

Note que isso é a definição de um hiper-plano, uma generalização de uma reta para o espaço $\mathbb{R}^n$. Nosso objetivo é minimizar o erro entre esse hiper-plano definido por $\widehat{f}$ e os pontos observados $\{\mathbf{x_i}, y_i\}$ — dessa forma, a regressão linear busca ajustar os coeficientes de um hiperplano no espaço das variáveis explanatórias.
Geralmente, o erro é medido pela Soma dos Resíduos ao Quadrado (RSS, da sigla em inglês):

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

onde $\widehat{y}_i - y_i$ é o resíduo $\epsilon$.

# Revisando a regressão linear VIII

Para ajustar $\widehat{f}$, queremos encontrar um vetor $\beta$ ótimo ($\widehat{\beta}$) que minimize o RSS. Supondo que $y$ é a solução do sistema $y = \mathbf{x}^T\widehat{\beta}$, queremos então:

$$\widehat{\beta} = \mathit{min}\ \mathrm{RSS}(\beta),$$

Onde o RSS para um dado $\beta$ é calculado por:

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

# Revisando a regressão linear IX

Esse problema tem solução única desde que as $n$ colunas sejam L.I.. Esse sistema pode ser escrito e resolvido de forma matricial. Existe uma forma de derivação desse problema, chamado de **Método dos Mínimos Quadrados** (OLS, da sigla em inglês), que dá o seguinte resultado:

$$(X^T X)\widehat{\beta} = X^T y,$$

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 X

Dessa forma, isolando $\widehat{\beta}$:

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

A ideia por trás do Método dos Mínimos Quadrados é que, ao minimizar a soma dos quadrados dos resíduos, estamos estimando $\widehat{y}$. De uma forma geral:

$$\widehat{y} = \beta \mathbf{x}_i^T + \epsilon_i.$$

# Revisando a regressão linear XI

Dessa forma, se derivarmos RSS($\beta$) e igualarmos a zero:

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

podemos encontrar a solução ótima — é dessa forma que obtemos a solução ótima na forma fechada pelos mínimos quadrados .

# Revisando a regressão linear XII

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. A próxima sessão irá apresentar uma estratégia para contornar o problema.

# Índice

# 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 $R(f)$. Denotando o RSS por $V$, temos:

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

Agora, precisamos escolher o termo regularizador $R(f)$ de forma que imponha uma penalidade de complexidade $f$, restringindo o espaço de soluções em uma região que contenha candidatos desejáveis.

# 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:

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

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

# Regularizando a regressão linear I

Relembrando, o RSS é definido como:

$$RSS = \sum_{i=1}^{n} (\widehat{y}_i - y_i)^2,$$

onde **y** é uma matriz coluna com os valores de $y_i$, e **X** é a matriz de variáveis explanatórias descrita anteriormente.
Se incluirmos nessa equação uma penalidade sobre $\beta$, dada por $\lambda \|\beta\|_1$, com $\lambda$ sendo o coeficiente de penalidade (ou parâmetro de regularização), temos agora que nosso $\hat{\beta}$ será calculado por:

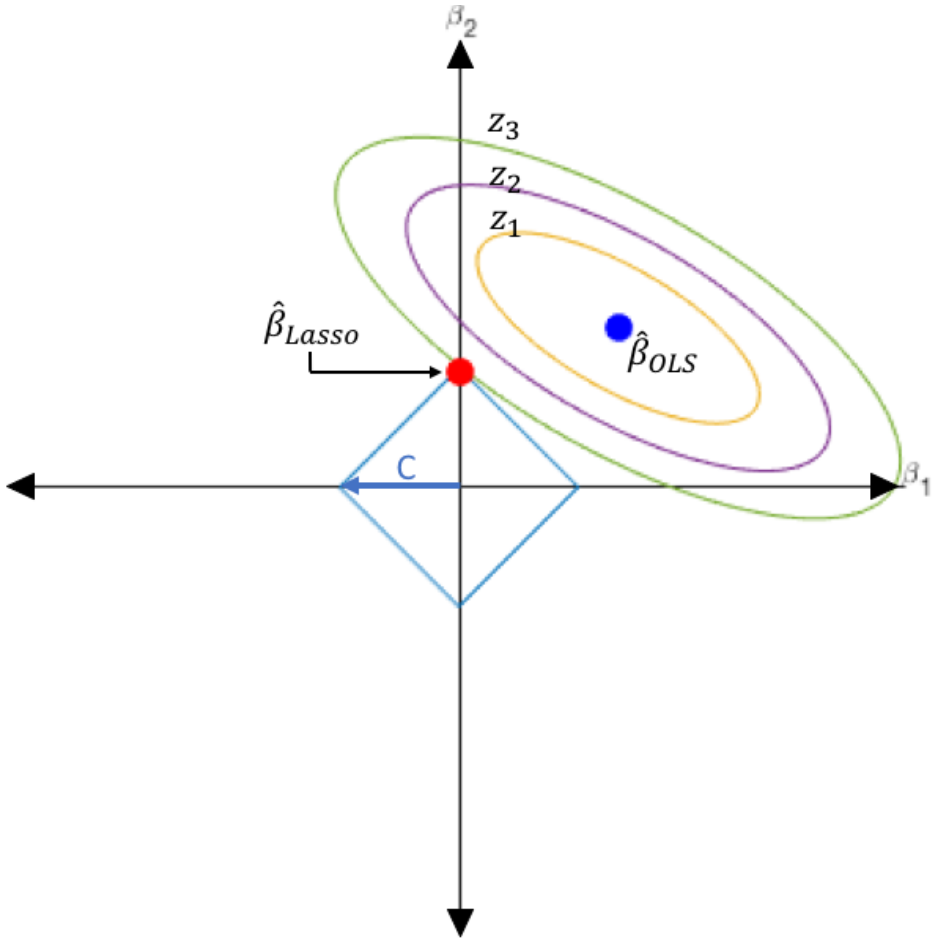
$$\widehat{\beta} = min(\text{RSS} + \lambda \|\beta\|_1) = min(\sum_{i=1}^{n} (y_i - \beta \mathbf{x}_i^T)^2 + \lambda \sum_{j=1}^{n} |\beta_j|).$$

# Regularização da regressão linear II

O primeiro termo dessa equação $(\sum_{i=1}^{n}(y_i - \beta \mathbf{x}_i^T)^2)$ é a já conhecida soma dos erros.
O segundo termo $(\lambda \sum_{j=1}^{n} |\beta_j|)$ é a penalidade adicionada. Na literatura, utilizar como penalidade a norma $l_1$ para a tarefa de regressão recebe um nome especial: Regressão Lasso, Regressão com normalização $L_1$, ou apenas Lasso.

# Justificativa do uso da norma $L_1$

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



# Implicações do termo regularizador I

Com isso, dado que $\lambda$ seja adequado, $\widehat{\beta}$ será encontrado em um ponto ótimo tal que, seja a região de diamante a região da função objetivo Lasso, e as elipses o gradiente da função MSE minimizada pelo OLS, o ponto ótimo (que seja mínimo dentro da região permitida) ocorre na intersecção entre as duas regiões.

# Implicações do termo regularizador II

O ponto ótimo cairá dentro de 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 zero, mas não necessariamente nulos ($= 0$).
Com isso, a penalidade restringe o espaço de busca dentro de uma região propício à anulação de um ou mais coeficientes, **podendo resultar em menos variáveis utilizadas na equação, podendo aliviar o overfit ou melhorar a qualidade e interpretabilidade do resultado**.

# Implicações do termo regularizador III

Além disso, por conta da anulação de algumas variáveis, a regressão Lasso costuma ser empregada no campo de Aprendizado de Máquina (ML da sigla em inglês) para a tarefa de seleção de atributos (*feature selection*, do inglês), diminuindo a dimensionalidade dos dados, e melhorando a interpretabilidade dos resultados e desempenho computacional.

# Implicações do termo regularizador IV

Então, temos que a restrição imposta ainda compartilha resultados bons com o OLS, mas se restringe à região onde soluções mais simples existem. Em um vértice qualquer do diamante, ao menos um coeficiente $\beta_i$ será nulo.
Dessa forma, o lasso tem poder de produzir soluções esparsas, onde vários parâmetros de $\widehat{\beta}$ são iguais a zero. Todo modelo estimado pela regressão Lasso terá cardinalidade $min(d, n)$, onde $n$ é o número ve variáveis e $d$ o número de pontos.

# Limitações

Porém, o lasso tem algumas limitações:

1. Quando $d \gg n$, o lasso selecionará até $n$ variáveis antes de saturar, por conta do problema convexo de otimização (como alternativa, temos técnicas como o *ElasticNet*, que contorna esse problema). Dessa forma, ele só selecionará $n$ variáveis;
2. O lasso assume independência entre variáveis. No caso onde há variáveis correlacionadas (que é o que ocorre normalmente em problemas do mundo real), 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

# Nova função de custo

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

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

Então que $\widehat{\beta} = min(L_1)$. Para a solução da regressão linear, a solução de forma fechada vem da resolução da derivada do sistema (pois, se queremos minimizar, a derivada resultará em um extremo da função, que corresponde ao mínimo global, pois a função de custo da regressão linear é convexa) — mas note que, neste caso, $L_1$ não é diferenciável, pois $|\beta|$ não é contínua em 0.

# Convexidade da nova função de custo

Ambos os termos da equação Lasso são convexos, de forma que existe um mínimo global, mas o termo da penalidade não é estritamente convexa, podendo existir vários valores de $\widehat{\beta}$ que minimizem a função de erro. Isso faz com que não exista uma solução explícita (fechada) para o problema, que encontre um valor ótimo, por não ser único.
Por isso, costumam ser utilizados métodos de otimização numérica — técnicas de otimização e análise convexa para resolver o problema de regressão com penalidade $l_1$.

# Implementações antigas

Implementações mais antigas da regressão Lasso utilizavam o *Least Angle Regression* (LARS) para encontrar a solução (em *python*, essa implementação é feita no *scikit-learn* sob o nome **Lasso LARS** , mas técnicas atuais conseguem realizar a tarefa com um menor custo computacional.
Vamos ver à seguir uma implementação por descida de coordenadas (*coordinate descent*, a técnica que a principal implementação do Lasso no *scikit-learn* se baseia ).

# Implementações mais modernas I

Por ser uma função de custo convexa, a otimização parâmetro-por-parâmetro converge para o ótimo, de forma que podemos iterar entre os parâmetros, um a um, fazendo uma atualização a cada passo, até chegar no ponto de convergência. O uso do *coordinate descent* é bom pois:

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

# Implementações mais modernas II

Seja o problema de minimizar $L_1$. A cada iteração, vamos percorrer o vetor $\beta$, um a um, sendo que quando estamos em um passo no $\beta_j$, todos os outros $\beta_{-j}$ são fixados — como fazemos a otimização coordenada a coordenada, é como se trabalhássemos no caso unidimensional. Seja o resíduo (desconsiderando o $\beta_j$) dado por:

$$r_{i,j} = y_i - \sum_{k \neq j} x_{i,k} \beta_k,$$

e seja o estimador OLS baseado em $\{r_{i,j}, x_{i,j}\}_{i=1}^{d}$:

$$Z_j = \sum_{i=1}^{d} x_{i,j} r_{i,j},$$

# Algoritmo de *coordinate descent* I

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

$\beta = [1, 1, ..., 1]$ ;
// para $t$ iterações
**for** *t=0, 1, 2, ...* **do**
    // para cada $\beta_j$
    **for** *j=1, 2, ..., n* **do**
        $\beta^{t+1}[j] = min\ F(\beta)$ à respeito de $\beta_j$ ;
        $\beta^{t+1}[i] = \beta[i]$, para todo $i \neq j$ ;
        $\beta = \beta^{t+1}$ ;
    **end**
**end**
**return** $\beta$ ;

# 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 $\beta^{t+1}$ à partir de $\beta^t$, ajustando um parâmetro por vez, minimizando a função de custo em relação a $\beta_j$. Queremos minimizar a função da mesma forma que fazemos para obter a solução da regressão pelo OLS — queremos derivar $L_1$ para obter essa expressão.

# Algoritmo de *coordinate descent* III

Note que $L_1 = OLS + \lambda \|\beta\|$, e sabemos que $\frac{d}{dx}(f+g) = \frac{d}{dx}f + \frac{d}{dx}g$. Pelo termo OLS, derivando parcialmente para um $\beta_j$:

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

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 $\beta_j = 0$. Essa observação feita é importante quando tentamos derivar o termo de penalidade Lasso:

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

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. A função de custo $L_1$ é convexa pois os termos OLS e Lasso são convexos.

# 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 $\beta_j$:

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

Como queremos o mínimo global da função de custo:

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

# Algoritmo de *coordinate descent* VIII

Sejam $\rho_j$ 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:

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

podemos isolar o termo com $\beta_j$:

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

Dividindo os dois lados pelo termo que multiplica $\beta_j$:

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

# Algoritmo de *coordinate descent* X

Agora que sabemos calcular um valor para $\beta_j$, podemos ver quando ele cairá em cada um dos casos. Note que isso depende então de $\rho_j$, $Z_j$, e $\lambda$:

$$\begin{cases} \beta_j < 0, \text{ se } \rho_j < -\lambda \\ \beta_j \in [-\lambda, \lambda], \text{ se } -\lambda < \rho_j < \lambda \\ \beta_j > 0 \text{ se } \rho_j > \lambda \end{cases}.$$

Podemos então definir a atualização de $\beta_j$ 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 $\rho_j$ utilizando $\beta$ ;
        calcule $Z_j$ utilizando $\beta$ ;
        $\beta[j] = S(\rho_j, \lambda, Z_j)$ ;
    **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

# Executando um pequeno teste

Criando um caso simples, sem ruído e onde os dados são gerados à partir da combinação linear de apenas 2 das 5 variáveis, o seguinte resultado foi obtido:

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| Coeficientes Originais | 0.000 | 0.000 | 0.000 | 97.996 | 45.705 |
| Coeficientes pelo OLS *scikit* | -4.279e-14 | -1.421e-14 | 2.664e-15 | 97.996 | 45.705 |
| Coeficientes pelo Lasso *scikit* | 0.000 | 0.000 | 0.000 | 96.846 | 44.665 |
| Coeficientes implementação | 0.000 | 0.000 | 0.000 | 97.996 | 45.705 |

# Convergência dos coeficientes

# Índice

# Conclusões

Vemos que 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, e inclusive por esse motivo o Lasso é utilizado no campo de ML como um pré processamento para diminuir a dimensionalidade dos dados.
Pelo gráfico de convergência, vemos que no momento em que o lambda se torna maior que a menor importância de variável, o algoritmo ”quebra”e passa a dar resultados incorretos. Isso ocorre pois o tamanho do passo é maior que o mínimo necessário para ajustar corretamente a importância daquela variável.