Estudo SVR - Da Classificação para a Regressão

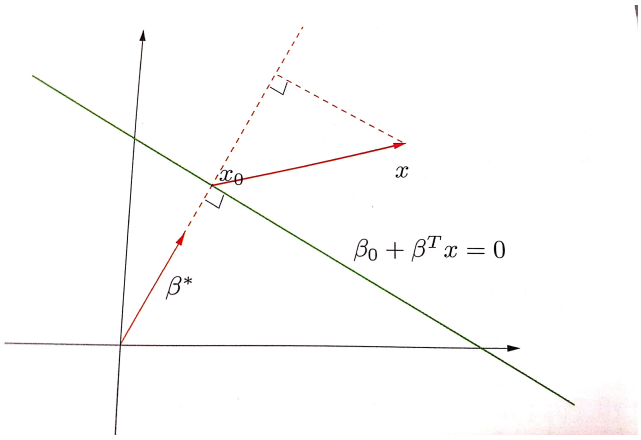
pedro.bloss.braga

April 2020

Objetivos

Pretendo iniciar com as definições do problema para classificação, e utilizar estas noções para intuir o problema de regressão, que, por definição, herda propriedades do primeiro.

Separando Hiperplanos



Na imagem é ilustrado um hiperplano (ou espaço afim L) de equação $f(x) = \beta_0 + \beta^T x = 0$, formando uma reta em $\mathbb{R}^2$ Temos: 1. Para quaisquer dois pontos $x_1, x_2 \in L$, vale que

$$\beta^T(x_1 - x_2) = 0 \Rightarrow \beta^* = \frac{\beta}{||\beta||}$$

β^* é o vetor normal à superfície L.

2. Para qualquer ponto $x_0 \in L$:

$$\beta^T x_0 = -\beta_0$$

3. A distância de qualquer ponto x a L é dada por

$$\beta^{*T}(x - x_0) = \frac{1}{||\beta||}(\beta^T x + \beta_0) = \frac{1}{||f'(x)||} f(x)$$

(4.40)

Então, $f(x)$ é proporcional à distância de x ao hiperplano definido por $f(x) = 0$.

Otimização em Separação de Hiperplanos

Desejamos maximizar a margem M entre duas classes, melhorando a performance de classificação.

$$\begin{aligned} &\underset{\beta,\beta_0,||\beta||=1}{max} && M \\ &\text{sujeito a} && y_i(x_i^T\beta + \beta_0) \geq M, i = 1, ..., N \end{aligned}$$

(4.45)

As restrições garantem que todos os pontos estão ao mínimo a uma distância $|M|$ da região de decisão definida por β e β_0. Livramo-nos da restrição $||\beta|| = 1$ fazendo

$$\frac{1}{||\beta||} y_i(x_i^T\beta + \beta_0) \geq M$$

(4.45)

e tomando $M = \frac{1}{||\beta||}$, pela lei do cancelamento:

$$\begin{aligned} &\underset{\beta,\beta_0}{min} && \frac{1}{2}||\beta||^2 \\ &\text{sujeito a} && y_i(x_i^T\beta + \beta_0) \geq 1, i = 1, ..., N \end{aligned}$$

(4.48)

O Lagrangiano (primário) funcional, a ser minimizado com respeito a β e β_0 é

$$L_P = \frac{1}{2}||\beta||^2 - \sum_{i=1}^{N} \alpha_i[y_i(x_i^T\beta + \beta_0) - 1]$$

(4.49) Tomando as derivadas igualadas a zero: $\partial_\beta L_P = 0$, $\partial_{\beta_0} L_P = 0$

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \qquad\qquad 0 = \sum_{i=1}^{N} \alpha_i y_i$$

(4.50) (4.51)

E substituindo estas na (4.49):

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

sujeito a

$$\alpha_i \geq 0 \qquad \text{e} \qquad \sum_{i=1}^{N} \alpha_i y_i = 0$$

(4.52)

A solução é obtida por meio da maximização de L_D , um problema simples de otimização convexa, sobre o qual pode-se usar algum software.

Adicionalmente, a solução deve satisfazer as condições de Karush-Kuhn-TUcker (KKT), que incluem (4.50), (4.51), (4.52) e

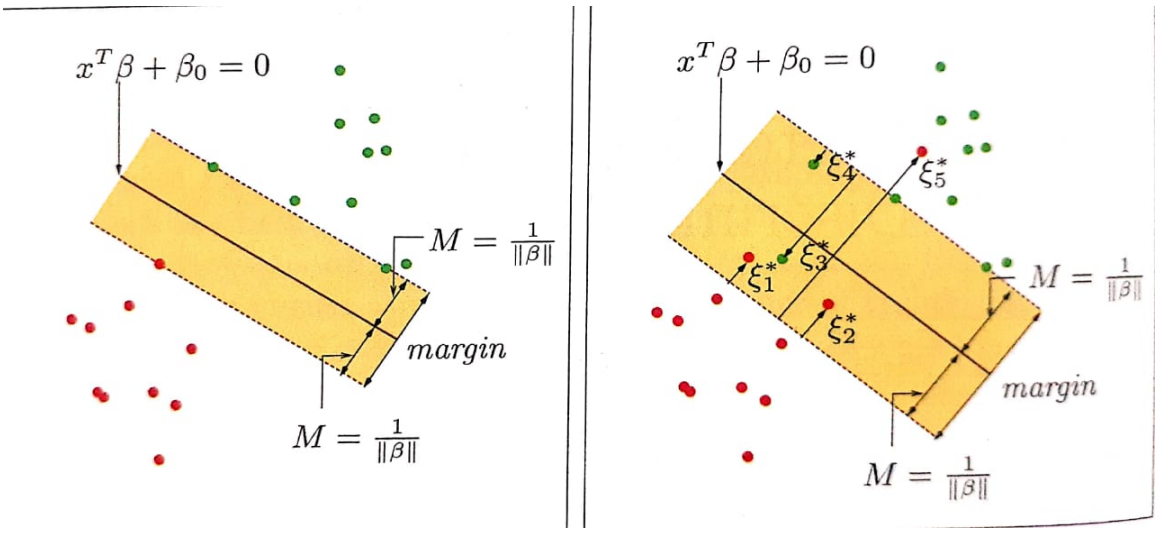
$$\alpha_i[y_i(x_{i=1}^T\beta + \beta_0) - 1] = 0$$

, $\forall i$.

A partir destas, podemos ver que:

- se $\alpha_i > 0 \Rightarrow y_i(x_i^T\beta + \beta_0)$
- se $y_i(x_i^T\beta + \beta_0) > 1 \Rightarrow x_i$ não está no limite da margem, e $\alpha_i = 0$.

Support Vector Classifier



Com os dados de treinamento $(x_1, y_1), ..., (x_N, y_N)$ com $x_i \in \mathbb{R}$ e $y_i n{-}1, 1$ (Classificador dicotômico), definimos o hiper-plano

$$\{x : f(x) = x^T\beta + \beta_0 = 0\}$$

(12.1)

onde β é o vetor unitário $||\beta|| = 1$.

A regra de classificação induzida por f é

$$G(x) = sign[x^T\beta + \beta_0]$$

(12.2)

Como as classes são separáveis, podemos achar uma função $f(x) = x^T\beta + \beta_0$ com $f(x_i) \geq 0 \forall i$.

Queremos achar o hiperplano com a maior margem entre pontos de treinamento para classe -1 e 1. O problema de otimização

$$\begin{aligned} \max_{\beta,\beta_0,||\beta||=1} \quad & M \\ \text{sujeito a} \quad & y_i(x_i^T\beta + \beta_0) \geq M, i = 1, ..., N \end{aligned}$$

(12.3) faz jus a este conceito.

Podemos escrever o mesmo problema na seguinte forma:

$$\begin{aligned} \min_{\beta,\beta_0} \quad & ||\beta|| \\ \text{sujeito a} \quad & y_i(x_i^T\beta + \beta_0) \geq 1, i = 1, ..., N \end{aligned}$$

(12.4)

(Uma maneira mais conveniente). Esta expressão é a maneira usual de definir o critério para o suporte vetorial de dados separados.

Note que sumimos com a restrição da norma de β pois $M = \frac{1}{||\beta||}$.

Este problema é de otimização convexa (critério quadrático, com restrições de desigualdades lineares), e, por definição, temos a unicidade do minimizador $x*$ global de uma f em um conjunto convexo C, tal que, $f(x^*) \leq f(x), \forall x \in C$.

Suponha agora que as classes se sobrepõe no espaço de features. Uma maneira de lidar com esta sobreposição é ainda maximizar M, mas permitindo que alguns pontos fiquem no "lado errado" da margem. Definimos as variáveis auxiliares $\xi = (\xi_1, ..., \xi_N)$. Existem duas maneiras "naturais" para modificar a restrição em (12.3):

$$y_i(x_i^T\beta + \beta_0) \geq M - \xi$$

(12.5)

$$y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi)$$

(12.6)

$\forall i, \xi_i \geq 0, \sum_i^N \xi_i \leq cte.$

As escolhas levam a diferentes soluções. A primeira parece mais natural, já que mede a sobreposição na real distância da margem; a segunda mede a sobreposição em distância relativa, que é modificada com a largura da margem M. No entanto, a primeira resulta num problema de otimização não convexa, enquanto a segunda é convexa; então, leva ao classificador SV "regular", que usaremos a partir daqui.

Aqui está a ideia da formulação:

O valor ξ_i na restrição $y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi)$ é proporcional à quantidade pela qual a previsão $f(x_i) = x_i^T\beta + \beta_0$ está no lado errado da margem. Então, por restringir a soma $\sum_i \xi_i$, restringimos a quantidade total de proporção pela qual previsões caem do lado errado da margem. Classificações equivocadas ocorrem quando $\xi_i > 1$, então restringindo $\sum_i \xi_i$ no valor K, restringe o número total de classificações erradas em K.

Utilizamos a seção de "Otimização em Separação de Hiperplanos" problema de otimização concretamente, e então sumimos com a restrição da norma de β, definindo $M = \frac{1}{||\beta||}$, e escrevemos (12.4) na maneira equivalente:

$$\begin{aligned} \min \quad & ||\beta|| \\ \text{sujeito a} \quad & \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \forall i \\ \xi_i \geq 0, \sum_i \xi_i \leq constante. \end{cases} \end{aligned}$$

(12.7)

Esta é a maneira usual de definir o classificador SV para casos não-separáveis.

Computando o Classificador SV

Support Vector Machines para regressão

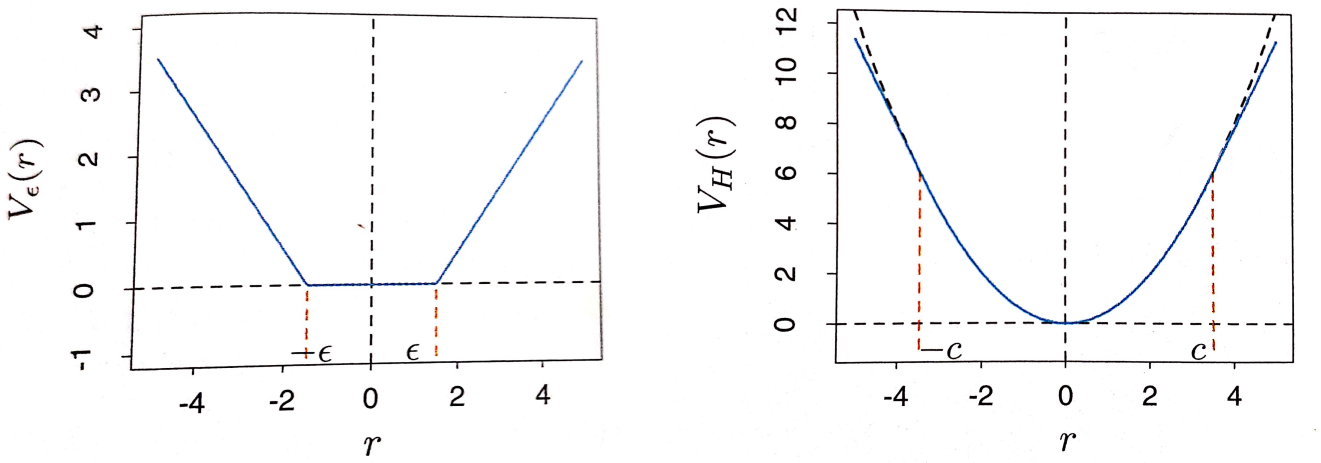
Pode-se adaptar o classificador SVM para regressão, com uma resposta quantitativa (ao invés de qualitativa), herdando algumas propriedades do clasificador SVM.

Por simplicidade, iniciamos pensando numa regressão linear $f(x) = x^T\beta + \beta_0$ e adaptamos para situações não-lineares.

Para estimar β, consideramos a minimização de

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2}||\beta||^2$$

sendo essa V uma função de erro ϵ-sensitiva, como na imagem do gráfico à esquerda:



De maneira que

$V_\epsilon = 0$ se $|r| < \epsilon$, e $V_\epsilon = |r| - \epsilon$, caso contrário.

$V_H = \frac{r^2}{2}$ se $|r| \leq c$, e $V_H = c|r| - \frac{c^2}{2}$, caso contrário.

A figura à direita expõe uma função de erro usada na Huber's Robust Regression, em que no intervalo $[-c, c]$ a função é quadrática, e em $[c, \infty+)$ e $(-\infty, c]$ é linear.

Esta função reduz de quadráticas a lineares as contribuições de observações com valor absoluto maior que um limítrofe previamente escolhido c. Isto faz com que o "fitting" seja menos sensitivo a outliers.

Se $\hat{\beta}$ e $\hat{\beta}_0$ são minimizadores de H, então a solução tem forma:

$$\hat{\beta} = \sum_{i=1}^{N} (\hat{\alpha}_i^* - \hat{\alpha}_i) x_i$$

(12.39)

$$\hat{f}(x) = \sum_{i=1}^{N} (\hat{\alpha}_i^* - \hat{\alpha}_i) x, x_i\rangle + \beta_0$$

(12.40)

onde $\hat{\alpha}_i^*$ e $\hat{\alpha}_i$ são positivos e solucionam o problema de programação quadrática:

$$\min_{\alpha_i^*, \alpha_i} \epsilon \sum_{i=1}^{N} (\alpha_i^* + \alpha_i) - \sum_{i=1}^{N} y_i(\alpha_i^* - \alpha_i) + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)\langle x_i, x_j\rangle$$

sujeito a

$$0 \leq \alpha_i \quad , \quad \alpha_i^* \leq \frac{1}{\lambda},$$

$$\sum_{i=1}^{N}(\alpha_i^* - \alpha_i) = 0,$$

$$\alpha_i^* \alpha_i = 0$$

(12.41)

Devido à natureza destas restrições, tipicamente apenas um subconjunto dos valores de solução $(\hat{\alpha}_i^* - \hat{\alpha}_i)$ são não-nulos, e os valores de dados associados são denominados **Support Vectors** (vetores de suporte).

Como no caso da classificação, a solução depende dos valores de input apenas sobre os produtos internos $\langle x_i, x_j \rangle$. Desta maneira, podemos generalizar os métodos para espaços de dimensões maiores (mais ricos), definindo apropriadamente o produto interno $\langle , \rangle$.

Faz-se a transformação de espaço por meio da função denominada **Kernel** (núcleo).

Alguns exemplos de Kernels:

Polinomial de ordem d: $K(x, x') = (1 + \langle x, x' \rangle)^d$

Radial basis (RBF): $K(x, x') = exp(-\gamma ||x - x'||^2$

Neural Network: $K(x, x') = tanh(k1\langle x, x' \rangle + k2)$

(12.22)

Note que há parâmetros ϵ e λ associados com a expressão

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2}||\beta||^2$$

ϵ é um parâmetro de perda da função V_ϵ, assim como c é para V_c.

Note que ambas V_ϵ e V_H dependem da escala de y, e portanto r. Se "escalarmos" a resposta (então usamos $V_H(r/\sigma)$ e $V_\epsilon(r/\sigma)$ ao invés), então podemos considerar usar valores escolhidos de c e ϵ. A quantidade λ é um parâmetros de regularização tradicional, estimado por meio de Cross-Validation.

Na escolha do Kernel, há certas condições, para que seja um kernel admissível.

Teorema de Mercer [1909]: Suponha $k \in L_\infty(\chi^2)$ tal que o operador de integral $T_k : \quad L_2(\chi) \to L_2(\chi)$, $T_k \quad f(.) := \int_\chi k(., x) f(x) d\mu(x)$ é positivo

Algumas noções importantes

Função convexa

Dado um conjunto C, Uma função $f : C \to \mathbb{R}$ é convexa, se e somente se,

$$f((1 - \lambda)x + \lambda y) \leq (1 - \lambda)f(x) + \lambda f(y)$$

$\forall x, y \in C$, $\forall \lambda \in [0, 1]$.

Conjunto Convexo

Um conjunto C é convexo quando

$$(1 - \lambda)x + \lambda y \in C, \forall x, y \in C, \forall \lambda \in [0, 1]$$