

Regressão Linear

1. Motivação

Exemplo: estimativa do preço de casas



\$150.000



\$385.000



????

Hipótese: existe uma relação matemática entre o valor do imóvel e a sua área, porém ela é desconhecida.

Desafio: a partir do conhecimento acerca da área e do valor de vários imóveis, queremos aproximar a relação matemática entre as grandezas envolvidas para, então, podermos estimar o valor de novos imóveis.

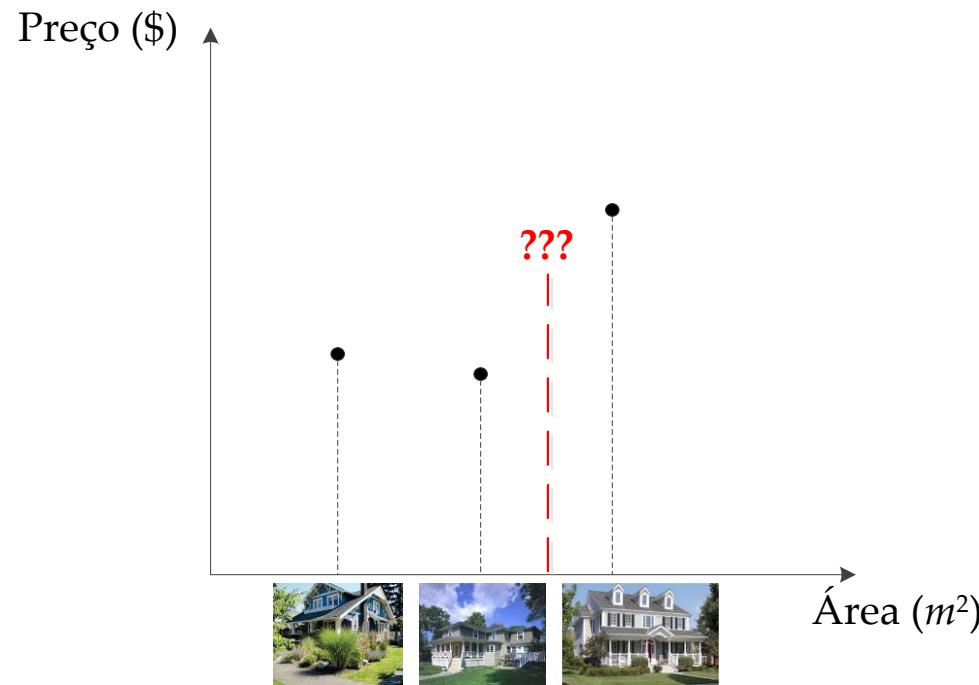


Figura 1 – Ilustração do problema de regressão. Dados inspirados em <https://www.kaggle.com/harlfoxem/housesalesprediction>.

Este tipo de desafio caracteriza o problema conhecido como **regressão**, também chamado de aproximação de funções, ou aproximação de mapeamentos.

2. Definição do problema

Visto de uma forma geral, o problema de regressão pode ser enunciado da seguinte forma.

Dados disponíveis: $\{\mathbf{x}(i); y(i)\}_{i=0}^{N-1}$

- $\mathbf{x}(i) \in \mathbb{R}^{K \times 1}$ representa o i -ésimo padrão de entrada, caracterizado por K atributos;
- $y(i)$ é o valor esperado de saída para o i -ésimo padrão de entrada.

Modelo proposto: $\hat{y}(\mathbf{x}) = g(\mathbf{x}, \boldsymbol{\theta})$, onde o vetor $\boldsymbol{\theta} \in \mathbb{R}^{P \times 1}$ contém os parâmetros que definem o mapeamento $g(\cdot)$.

Desejamos que a saída gerada pelo modelo para a entrada $\mathbf{x}(i)$, denotada por $\hat{y}(\mathbf{x}(i))$, seja a mais próxima possível do valor conhecido $y(i)$, para $i = 0, \dots, N - 1$. Para isto, precisamos determinar os valores ideais dos P parâmetros do modelo, i.e., o vetor $\boldsymbol{\theta}$.

Critério: estabelece matematicamente o objetivo que se busca atingir com o modelo. No caso do problema de regressão, queremos que o erro entre a aproximação $\hat{y}(\mathbf{x}(i))$ e o valor desejado $y(i)$ seja o menor possível.

Existem várias possibilidades para se definir a função de erro a ser minimizada. Classicamente, a medida de erro quadrático médio (MSE, do inglês *mean squared error*) é a mais utilizada neste contexto (BISHOP, 2006; HAYKIN, 2013):

$$J_e(\boldsymbol{\theta}) = E\{(y(i) - g(\mathbf{x}(i), \boldsymbol{\theta}))^2\} \quad (1)$$

onde $E\{\cdot\}$ denota o operador de esperança estatística.

Como, em geral, tem-se à disposição um conjunto finito de dados, a esperança estatística pode ser aproximada pela média amostral, levando-nos ao funcional:

$$J_e(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=0}^{N-1} (y(i) - g(\mathbf{x}(i), \boldsymbol{\theta}))^2, \quad (2)$$

o qual caracteriza o critério conhecido como quadrados mínimos (LS, do inglês *least-squares*) .

Problema: consiste em buscar os valores ótimos para os parâmetros do modelo, ou seja, aqueles que levam ao mínimo valor para o funcional de erro.

$$\min_{\boldsymbol{\theta}} J_e(\boldsymbol{\theta}) \quad (3)$$

Surge, portanto, um problema de otimização durante o ajuste, ou **treinamento**, do modelo.

2.1. Abordagem via máxima verossimilhança

Embora o critério baseado no erro quadrático médio seja bastante intuitivo, poderíamos, também, explorar o princípio da máxima verossimilhança para tentar resolver o problema de regressão.

Hipótese: $y(i) = \hat{y}(\mathbf{x}(i)) + \eta_i = g(\mathbf{x}(i), \boldsymbol{\theta}) + \eta_i$, em que $\eta_i \sim N(0, \sigma_\eta^2)$.

- Em outras palavras, presume-se que o erro entre a aproximação gerada pelo modelo e o valor desejado segue uma distribuição Gaussiana.

PDF conjunta:

$$P(\mathbf{Y}|\boldsymbol{\theta}) = \prod_{i=0}^{N-1} p_y(y(i); \boldsymbol{\theta}) = \prod_{i=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma_\eta^2}} e^{-\frac{(y(i)-g(\mathbf{x}(i), \boldsymbol{\theta}))^2}{2\sigma_\eta^2}}$$

Aplicando o logaritmo, obtemos a seguinte expressão para a *log-likelihood*:

$$\ln P(\mathbf{Y}|\boldsymbol{\theta}) = -\frac{N}{2} \ln 2\pi - \frac{N}{2} \ln \sigma_\eta^2 - \frac{1}{2\sigma_\eta^2} \sum_{i=0}^{N-1} (y(i) - g(\mathbf{x}(i), \boldsymbol{\theta}))^2$$

Os dois primeiros termos não dependem do vetor de parâmetros $\boldsymbol{\theta}$ (são constantes). Logo, maximizar $\ln P(\mathbf{Y}|\boldsymbol{\theta})$ leva à mesma solução que maximizar somente o último termo:

$$\max_{\boldsymbol{\theta}} -\frac{1}{2\sigma_\eta^2} \sum_{i=0}^{N-1} (y(i) - g(\mathbf{x}(i), \boldsymbol{\theta}))^2$$

Como $\max -f = \min f$ e $\min f = \min \alpha f$, com $\alpha > 0$ escalar, então a solução de máxima verossimilhança pode ser obtida resolvendo o seguinte problema:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^N (y(i) - g(\mathbf{x}(i), \boldsymbol{\theta}))^2,$$

que corresponde ao próprio critério de quadrados mínimos.

3. Regressão linear

Neste tópico, o modelo explorado para realizar a aproximação do mapeamento subjacente aos dados disponíveis é linear.

Modelo proposto: $\hat{y}(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_K x_K$.

Explorando uma notação vetorial, podemos escrever que:

$$\hat{y}(\mathbf{x}) = \boldsymbol{\Phi}(\mathbf{x})^T \mathbf{w}, \tag{4}$$

onde $\mathbf{w} = [w_0 \ w_1 \ \cdots \ w_K]^T$ e $\Phi(\mathbf{x}) = [1 \ x_1 \ \cdots \ x_K]^T$. Note que o vetor $\Phi(\mathbf{x})$ inclui uma entrada fixa de valor unitário que está relacionada ao coeficiente w_0 , conhecido como *bias* ou termo de polarização.

Como temos à disposição N pares de dados entrada-saída, podemos montar a matriz $\Phi \in \mathbb{R}^{N \times (K+1)}$:

$$\Phi = \begin{bmatrix} \Phi(\mathbf{x}(0))^T \\ \vdots \\ \Phi(\mathbf{x}(N-1))^T \end{bmatrix} \quad (5)$$

e o vetor de resposta do modelo

$$\hat{\mathbf{y}} = [\hat{y}(\mathbf{x}(0)) \ \cdots \ \hat{y}(\mathbf{x}(N-1))]^T, \quad (6)$$

de maneira que

$$\hat{\mathbf{y}} = \Phi \mathbf{w}. \quad (7)$$

Para simplificarmos um pouco a notação, vamos escrever que $\hat{\mathbf{y}}(i) = \hat{\mathbf{y}}(\mathbf{x}(i))$, onde $i = 0, \dots, N - 1$. Agrupamos, também, todas as saídas desejadas no vetor $\mathbf{y} = [y(0) \cdots y(N - 1)]^T$.

Idealmente, desejamos que $\hat{\mathbf{y}} = \Phi \mathbf{w}$ seja igual a \mathbf{y} :

$$\Phi \mathbf{w} = \mathbf{y} \tag{8}$$

Na prática, devemos encontrar os valores dos coeficientes em \mathbf{w} que tornem a saída do modelo a mais parecida possível com a saída desejada.

3.1. Solução de quadrados mínimos e equações normais

1º caso: $\Phi \in \mathbb{R}^{N \times K+1}$, com $K + 1 < N$ e de posto completo.

- Nesta situação, existem mais padrões de entrada $\mathbf{x}(i)$ do que parâmetros ajustáveis (i.e., mais equações do que variáveis no sistema linear em (8)).

Critério: quadrados mínimos.

$$\min_{\mathbf{w}} J_e(\mathbf{w}) = \min_{\mathbf{w}} \frac{1}{N} \sum_{i=0}^{N-1} (y(i) - \hat{y}(i))^2 \quad (9)$$

Seja $e(i) = y(i) - \hat{y}(i)$ o erro cometido na aproximação do i -ésimo padrão.

Novamente explorando a notação vetorial, podemos definir:

$$\mathbf{e} = [e(0) \ \cdots \ e(N-1)]^T \quad (10)$$

como o vetor de erro. Desta forma, o funcional de erro pode ser reescrito como:

$$J_e(\mathbf{w}) = \frac{1}{N} \sum_{i=0}^{N-1} (y(i) - \hat{y}(i))^2 = \frac{1}{N} \|\mathbf{e}\|^2. \quad (11)$$

Com isto, o problema de otimização que surge no contexto de regressão linear pode ser definido como:

$$\min_{\mathbf{w}} \|\mathbf{e}\|^2 \quad (12)$$

Ora,

$$\|\mathbf{e}\|^2 = \|\mathbf{y} - \hat{\mathbf{y}}\|^2 = \|\mathbf{y} - \Phi\mathbf{w}\|^2 \quad (13)$$

Desenvolvendo a expressão, obtemos:

$$\|\mathbf{e}\|^2 = (\mathbf{y} - \Phi\mathbf{w})^T(\mathbf{y} - \Phi\mathbf{w}) = \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\Phi\mathbf{w} - \mathbf{w}^T\Phi^T\mathbf{y} + \mathbf{w}^T\Phi^T\Phi\mathbf{w} \quad (14)$$

Derivando (14) com respeito a \mathbf{w} e igualando a zero:

$$\frac{\partial \|\mathbf{e}\|^2}{\partial \mathbf{w}} = 2\mathbf{w}^T\Phi^T\Phi - \mathbf{y}^T\Phi - (\Phi^T\mathbf{y})^T = 0 \quad (15)$$

Então,

$$\mathbf{w}^T\Phi^T\Phi - \mathbf{y}^T\Phi = 0 \quad (16)$$

Aplicando o operador $(\cdot)^T$:

$$\Phi^T\Phi\mathbf{w} = \Phi^T\mathbf{y} \quad (17)$$

As equações envolvidas em (17) são conhecidas como **equações normais**.

Como, por hipótese, Φ tem posto completo, existe $(\Phi^T\Phi)^{-1}$. Com isto:

$$\boxed{\mathbf{w} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y}} \quad (18)$$

Questão: E se a matriz Φ não tiver posto completo, i.e., se $\nexists (\Phi^T\Phi)^{-1}$?

- Isto pode ocorrer devido à presença de atributos redundantes, i.e., colunas linearmente dependentes em Φ .
- **Algumas soluções possíveis:**
 - Eliminar as colunas que apresentem dependência linear.
 - Isto pode ser feito com o auxílio de técnicas de seleção de variáveis (GUYON & ELISSEEFF, 2003), de modo que se retenha apenas um subconjunto das variáveis mais relevantes, segundo algum critério, para o problema.
 - Também pode ser feito via redução de dimensionalidade.
 - $\Phi^T\Phi$ possui autovalores nulos (daí o fato de não possuir inversa). Então, uma forma de viabilizar o cálculo de $(\Phi^T\Phi)^{-1}$ consiste em positivar a matriz através da adição de uma constante (pequena) $\lambda > 0$ à diagonal de $\Phi^T\Phi$.

Esta opção, conforme veremos mais adiante, está associada a uma técnica de regularização.

2º caso: $\Phi \in \mathbb{R}^{N \times K+1}$, com $N < K + 1$ e de posto completo.

- Agora, temos mais parâmetros livres do que padrões de entrada, ou, equivalentemente, mais variáveis a determinar do que equações no sistema linear.
Nesta situação, há infinitas soluções.
- Sendo assim, é possível definir como solução única aquela que minimiza a soma dos erros quadráticos e que possui a mínima norma do vetor de coeficientes.

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 \quad (19)$$
$$s. a. \quad \Phi \mathbf{w} = \mathbf{y}$$

Solução: método dos coeficientes de Lagrange.

Lagrangeano: $\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}) = \frac{1}{2} \|\mathbf{w}\|^2 - \boldsymbol{\lambda}^T (\Phi \mathbf{w} - \mathbf{y})$

Aplicando as condições de otimalidade:

$$\frac{\partial \mathcal{L}(\mathbf{w}, \boldsymbol{\lambda})}{\partial \mathbf{w}} = \mathbf{w}^T - \boldsymbol{\lambda}^T \Phi = 0 \Rightarrow \mathbf{w}^T = \boldsymbol{\lambda}^T \Phi \quad (20)$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} = (\Phi \mathbf{w} - \mathbf{y}) = 0 \quad (21)$$

Substituindo \mathbf{w} de (20) em (21), temos que $\Phi \Phi^T \boldsymbol{\lambda} = \mathbf{y}$.

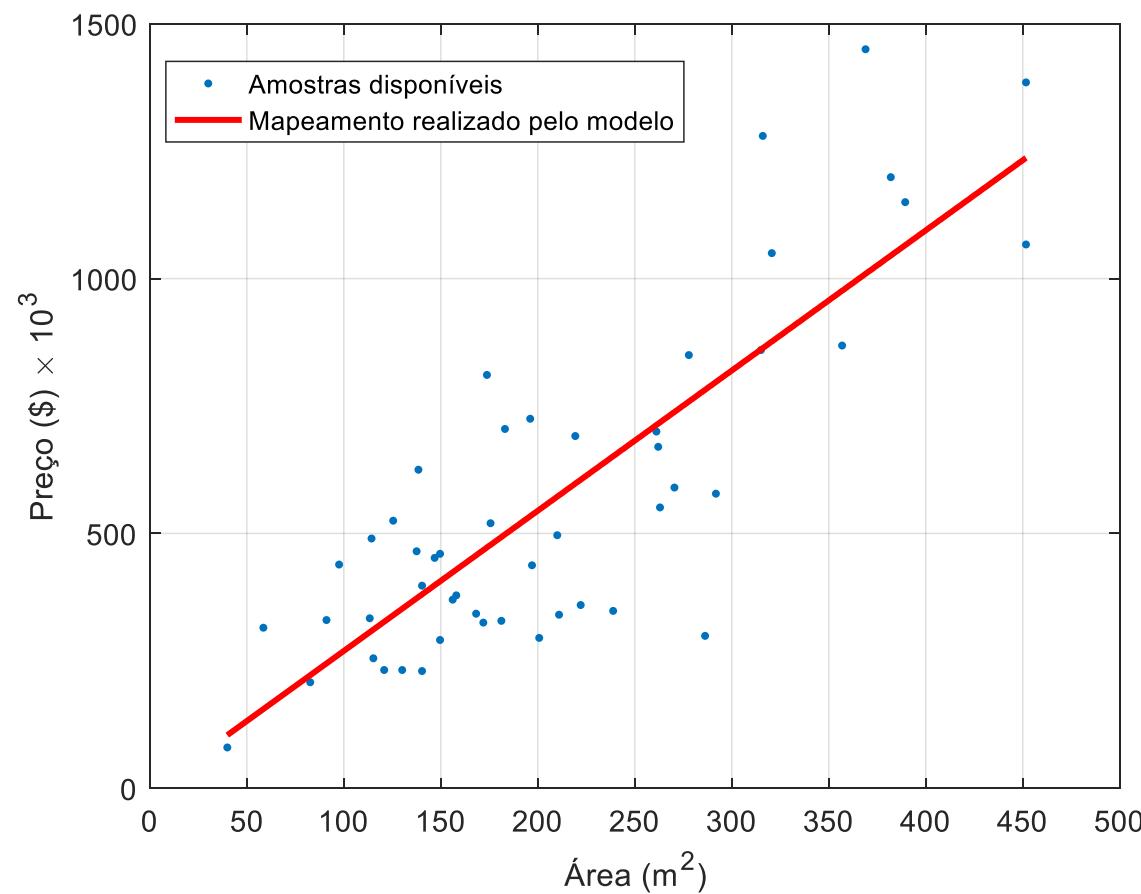
Como a matriz Φ tem posto completo, $\Phi \Phi^T$ possui inversa, de modo que:

$$\boldsymbol{\lambda} = (\Phi \Phi^T)^{-1} \mathbf{y} \quad (22)$$

Finalmente, substituindo (22) em (20) e tomando o transposto, chegamos à outra forma da solução de quadrados mínimos baseada na pseudo-inversa de Φ :

$$\boxed{\mathbf{w} = \Phi^T (\Phi \Phi^T)^{-1} \mathbf{y}} \quad (23)$$

Exemplo: predição do preço de casas



Vetor ótimo de coeficientes: $\mathbf{w} = \begin{bmatrix} -5722,9 \\ 2751,1 \end{bmatrix}$

Relação entrada-saída: $\hat{y}(\mathbf{x}) = -5722,9 + 2751,1x_1$

4. Aprendizado a partir de dados amostrados

No problema de regressão, assim como em outras tarefas de aprendizado supervisionado, em que também se adapta um modelo a partir de dados amostrados, há **três mapeamentos** envolvidos:

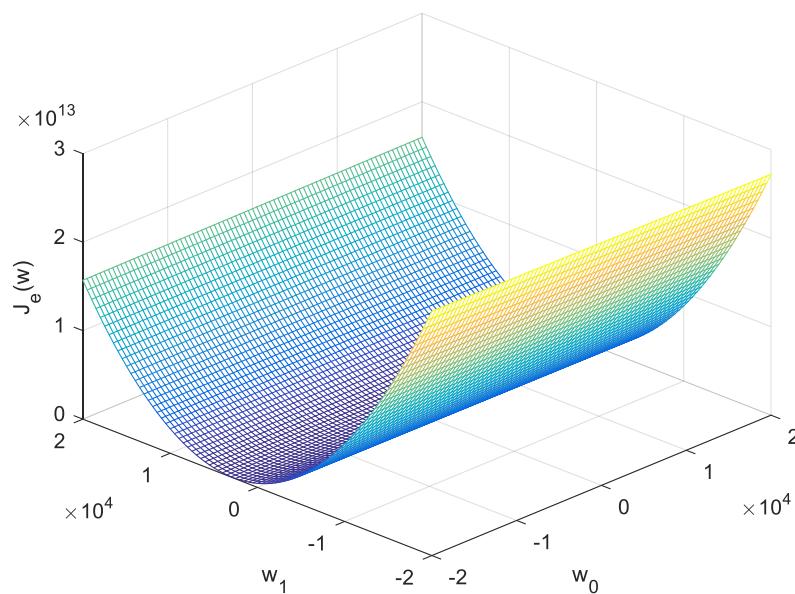
1. O mapeamento a ser aproximado, $y = f(\mathbf{x}): \mathbb{R}^K \rightarrow \mathbb{R}$, do qual se conhece apenas um conjunto finito de dados amostrados.
2. O mapeamento resultante do processo de aproximação, associado a um único vetor de parâmetros, $\hat{y} = g(\mathbf{x}; \boldsymbol{\theta}): \mathbb{R}^K \rightarrow \mathbb{R}$. Este é o mapeamento que o modelo – no caso, linear – efetivamente produz.
3. O mapeamento entre cada possível atribuição de valores aos parâmetros do modelo e o erro correspondente: $J_e(\boldsymbol{\theta}): \mathbb{R}^P \rightarrow \mathbb{R}$. Este mapeamento define a **superfície de erro** e está intimamente ligado ao critério adotado.

No problema de regressão linear, a superfície de erro para o critério de quadrados mínimos é dada pela expressão:

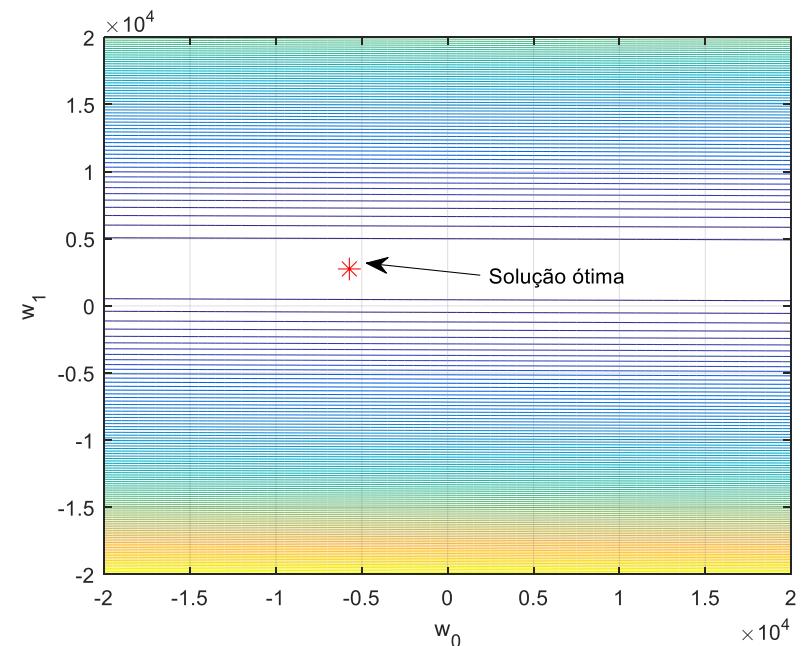
$$J_e(\mathbf{w}) = \mathbf{y}^T \mathbf{y} - \mathbf{y}^T \Phi \mathbf{w} - \mathbf{w}^T \Phi^T \mathbf{y} + \mathbf{w}^T \Phi^T \Phi \mathbf{w}$$

- $J_e(\mathbf{w})$ assume uma forma quadrática com respeito ao vetor $\mathbf{w} \in \mathbb{R}^{K+1 \times 1}$ e possui um único mínimo global, dado pela solução baseada em pseudo-inversa.

Exemplo: *house sales prediction*



(a) Superfície de erro.



(b) Curvas de nível.

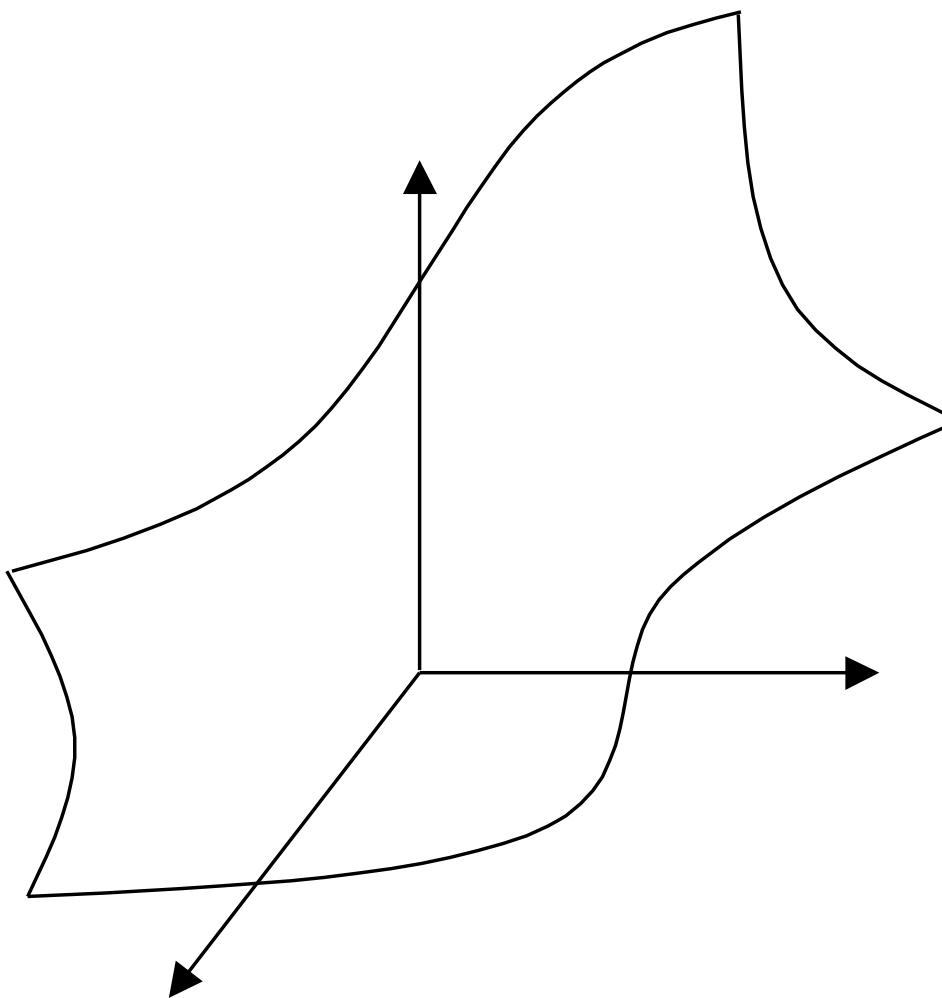


Figura 2 – Mapeamento desconhecido a ser aproximado.

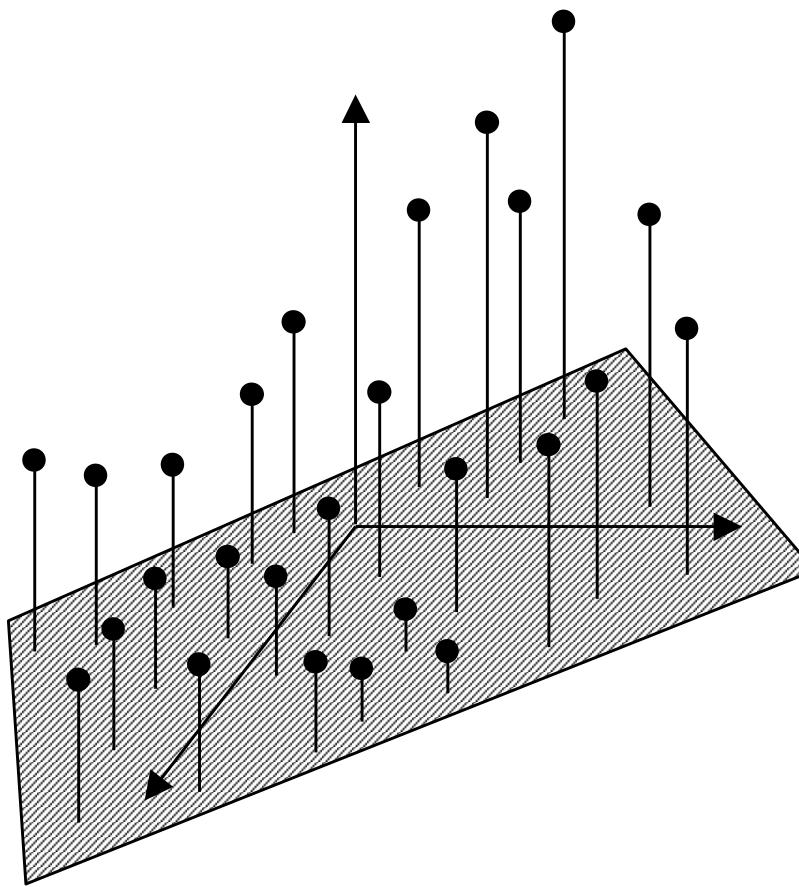


Figura 3 – Conjunto de amostras disponíveis. Com isto, passamos a conhecer o comportamento da função para pontos específicos na região de operação.

5. Métodos iterativos

A existência de uma solução ótima, no sentido de quadrados mínimos, em forma fechada para o problema de regressão linear é, sem dúvidas, algo importante, pois significa que podemos obter diretamente o ponto de mínimo da superfície de erro envolvida no problema.

Entretanto, o cálculo da pseudo-inversa pode se tornar bastante custoso quando o número de coeficientes (ou, analogamente, o número de atributos K) se torna elevado: $O(K^3)$.

Nesta situação, uma alternativa consiste em lançar mão de algoritmos iterativos de busca que façam a atualização dos parâmetros \mathbf{w} à medida que os dados forem apresentados ao modelo (LUENBERGER, 1984; BATTITI, 1992).

5.1. Método de primeira ordem: gradiente descendente

A cada iteração, a partir do ponto atual no espaço de busca, toma-se a direção oposta ao vetor gradiente. Ou seja, seguimos na direção de máximo decrescimento do valor do funcional de erro.

Regra de atualização: $\mathbf{w}_{i+1} = \mathbf{w}_i - \alpha \nabla J_e(\mathbf{w}_i)^T$.

Gradiente: $\nabla J_e(\mathbf{w}) = 2\mathbf{w}^T \Phi^T \Phi - 2\mathbf{y}^T \Phi = -2\mathbf{e}^T \Phi$.

Exemplo:

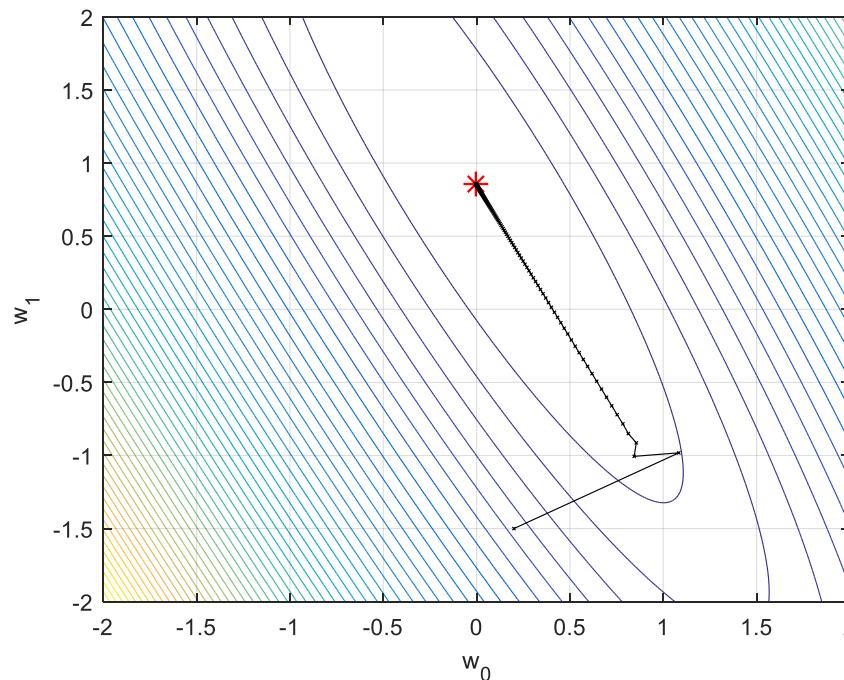


Figura. Trajetória realizada pelo algoritmo do gradiente descendente, com $\alpha = 0,01$, até a convergência.

Enxergando o processo de busca:

Iteração 1

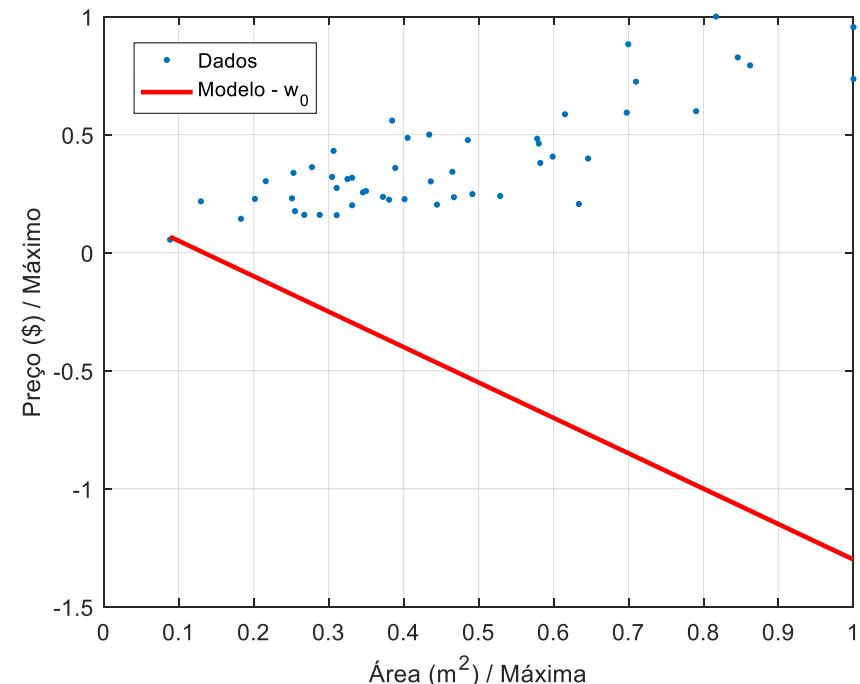
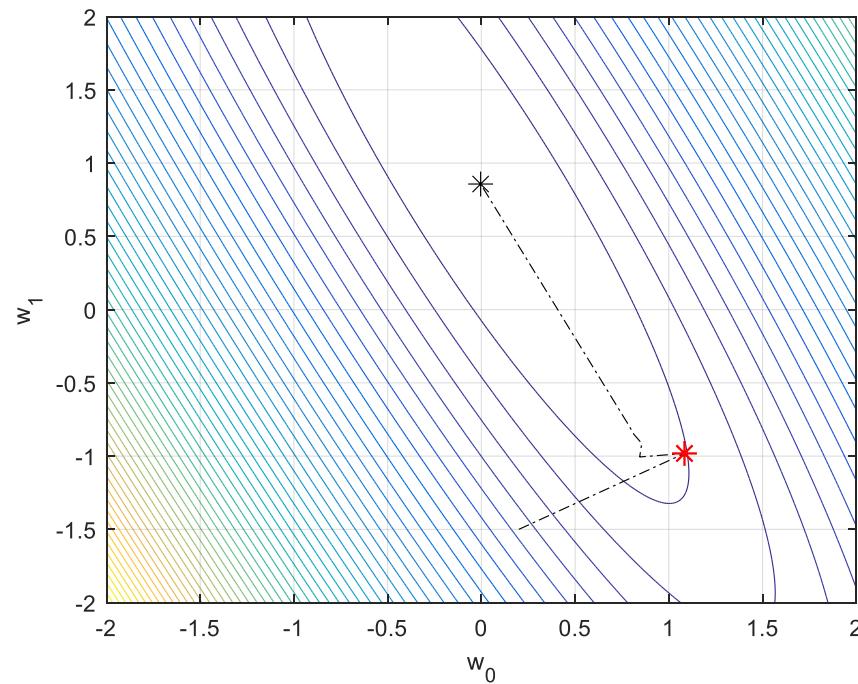


Figura 4 – Em destaque (asterisco em vermelho), o vetor de coeficientes obtido na iteração 1 do algoritmo do gradiente descendente. Ao lado, o mapeamento entrada-saída correspondente.

Iteração 22

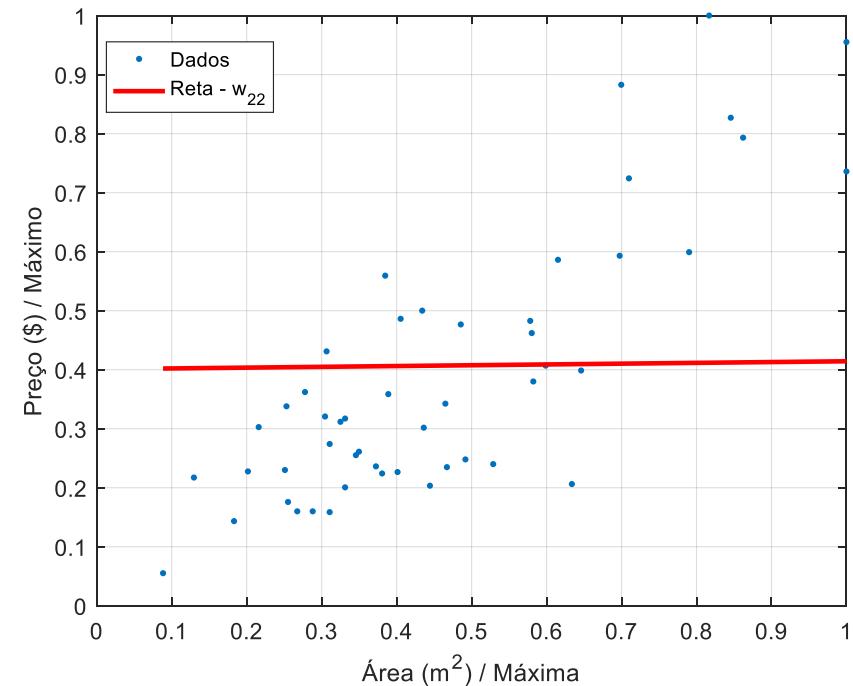
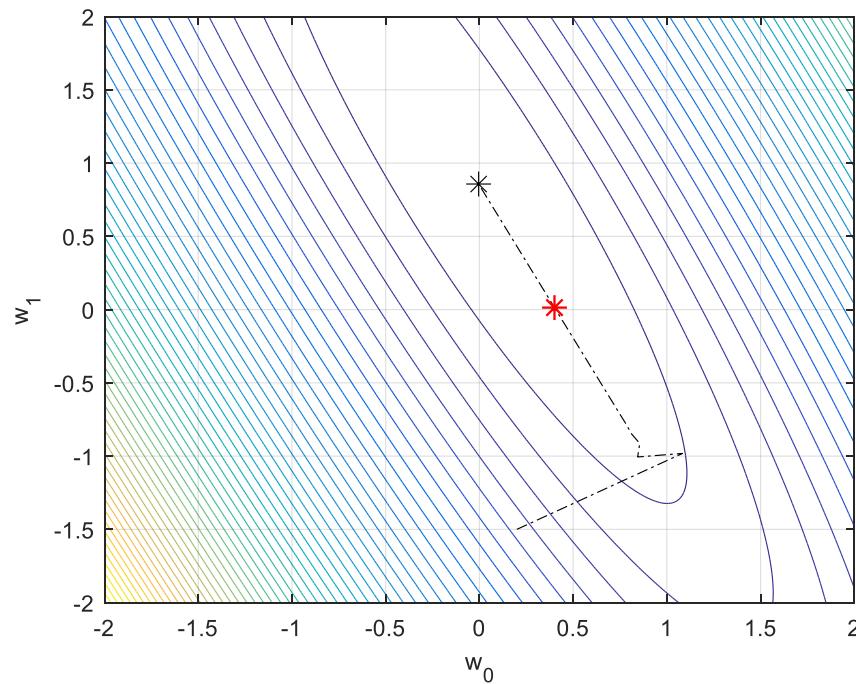


Figura 5 – Em destaque (asterisco em vermelho), o vetor de coeficientes obtido na iteração 22 do algoritmo do gradiente descendente. Ao lado, o mapeamento entrada-saída correspondente.

Iteração 50

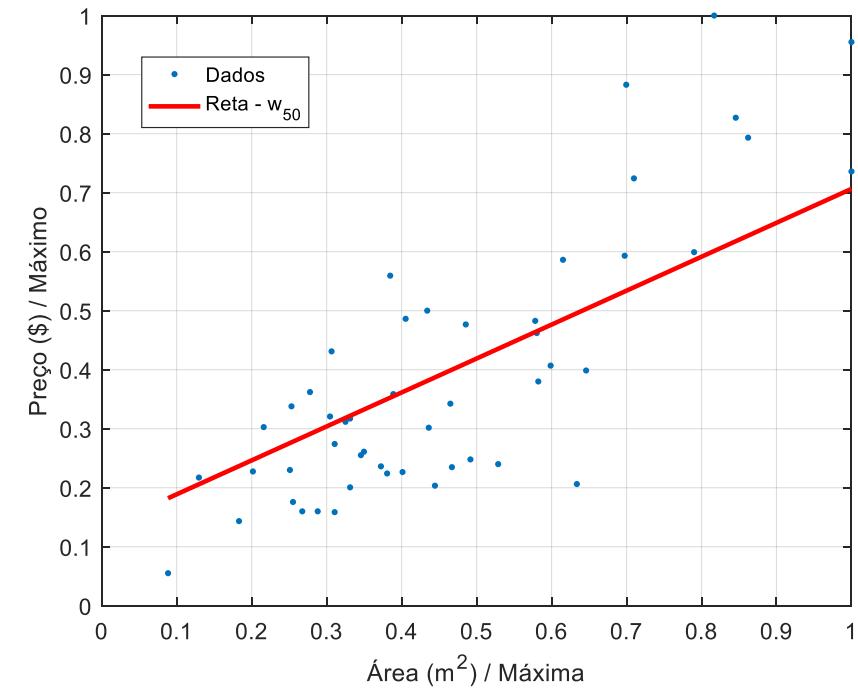
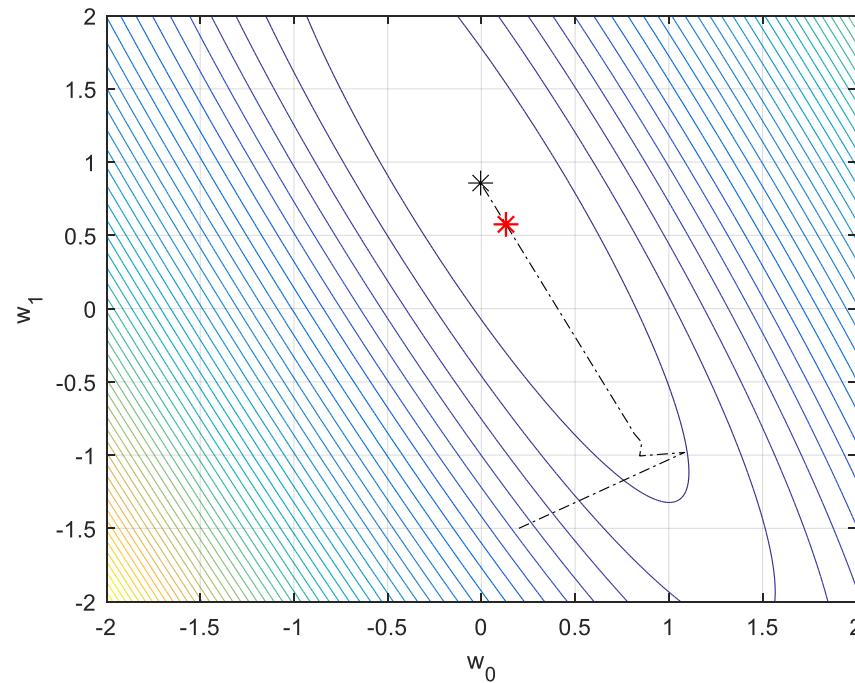


Figura 6 – Em destaque (asterisco em vermelho), o vetor de coeficientes obtido na iteração 50 do algoritmo do gradiente descendente. Ao lado, o mapeamento entrada-saída correspondente.

Versões:

- **Batelada (batch):** em cada passo do algoritmo, todos os padrões de treinamento são considerados. Esta estratégia foi explorada no exemplo anterior.
- **Padrão-a-padrão (online):** para cada padrão de entrada, faz-se o ajuste dos parâmetros conforme a direção definida pelo vetor do gradiente instantâneo.

Problema: $\min_{\mathbf{w}} (y(i) - \hat{y}(i))^2 \rightarrow$ erro instantâneo.

Vetor gradiente: $\nabla J_e(\mathbf{w}) = -2(y(i) - \hat{y}(i))\Phi(\mathbf{x}(i))^T$.

Esta versão do método de primeira ordem é conhecida como *stochastic gradient descent* (SGD).

- Cada iteração do algoritmo corresponde a uma atualização do vetor de coeficientes, para um padrão de entrada. Uma época corresponde a uma apresentação completa do conjunto de amostras de treinamento.

Exemplo: SGD

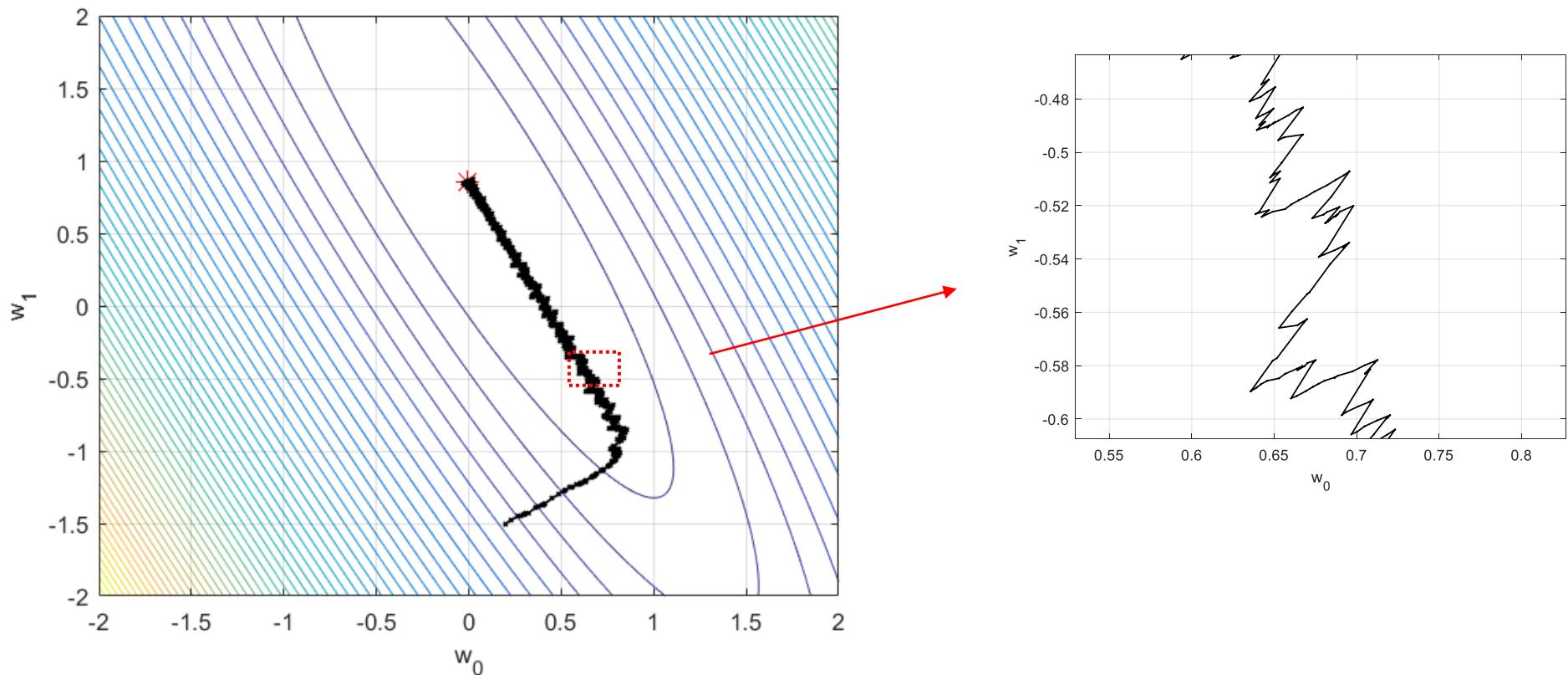


Figura 7 – Trajetória realizada pelo algoritmo do gradiente estocástico, com $\alpha = 0,001$, até atingir 1000 épocas. Em destaque, percebemos o comportamento oscilatório durante o percurso até o ótimo.

- *Mini-batch*: pode ser visto como um meio-termo entre as duas abordagens anteriores. Aqui, cada passo do gradiente descendente considera $1 < Q < N$ amostras do conjunto de treinamento.

6. Escala das variáveis de entrada

Em algumas situações, alguns atributos de entrada acabam sendo dominantes sobre os demais no sentido de exercerem grande influência sobre o erro cometido pelo modelo. Isto pode ocorrer devido à diferença de magnitude entre os atributos.

Exemplo: número de quartos no imóvel: 1 a 5.

área do imóvel: 0 a 2000 m².

Nestes casos, é pertinente realizar um pré-processamento sobre os atributos de entrada, visando deixar todas as variáveis na mesma faixa de valores.

Abordagens:

- *Feature scaling*: ajusta a escala de todos os atributos para um intervalo comum.

Exemplo: $x_i \in (0,2000) \implies x_i' \leftarrow \frac{x_i}{\max(x_i)} \in (0,1)$

$$\text{Mapeamento linear: } x_i' \leftarrow \frac{1}{(\max(x_i) - \min(x_i))} x_i - \frac{\min(x_i)}{(\max(x_i) - \min(x_i))}$$

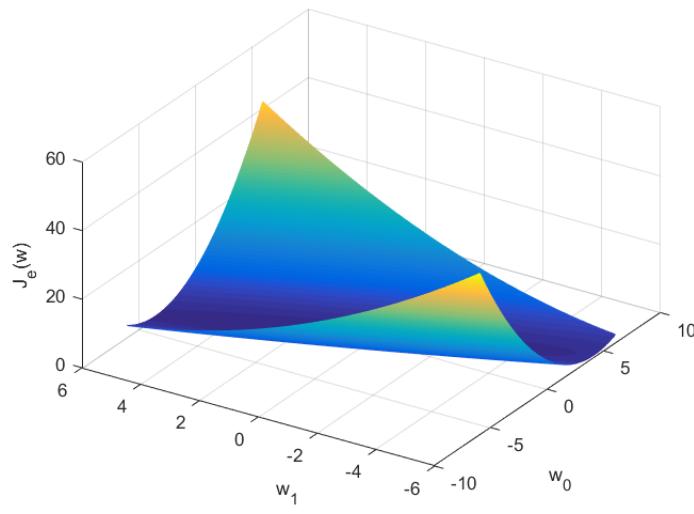
- Normalização de média: todos os atributos passam a ter média nula.

Exemplo: $x_i' \leftarrow \frac{x_i - \mu_i}{s_i}$, onde s_i denota o comprimento do intervalo de valores de x_i .

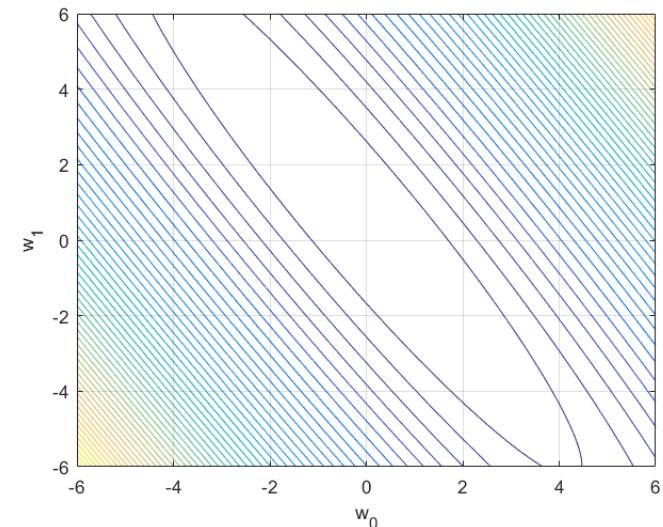
- *Data standardization*: $x_i' \leftarrow \frac{x_i - \mu_i}{\sigma_i}$, onde $\sigma_i = \text{std}(x_i)$ denota o desvio padrão da variável x_i .

Este tipo de pré-processamento favorece o progresso de algoritmos baseados no gradiente, uma vez que deixa as curvas de nível da superfície de erro mais circulares.

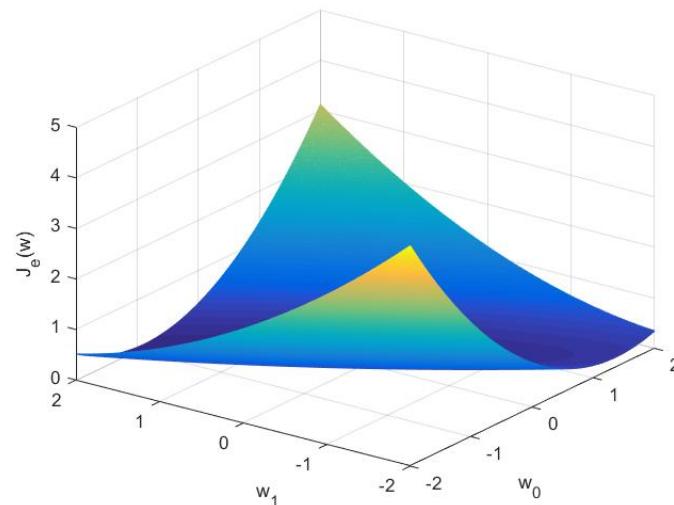
Exemplo: *house sales prediction*



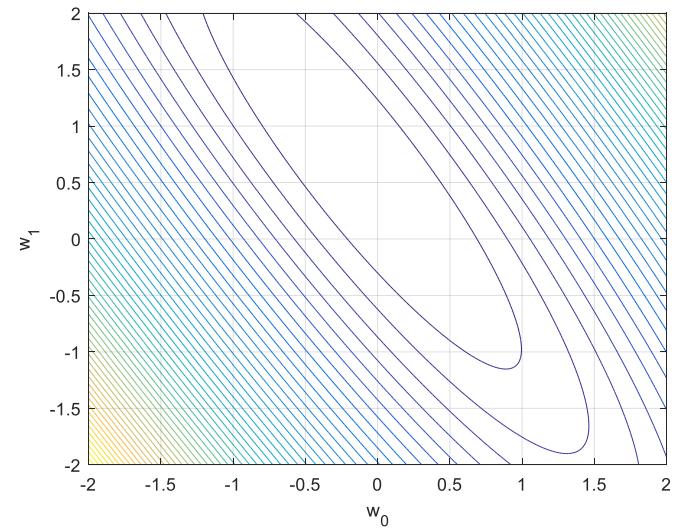
(a) Sup. de erro - *Feature scaling*



(b) Curvas de nível - *Feature scaling*



(c) Sup. de erro – *Data standardization*



(d) Curvas de nível – *Data standardization*

7. Extensão para múltiplas saídas

Agora, cada padrão de entrada $\mathbf{x}(i)$ está associado a um conjunto de L variáveis de saída, $y_k(i)$, $k = 0, \dots, L - 1$.

Modelo proposto:

$$\hat{y}_0(\mathbf{x}) = w_0^{(0)} + w_1^{(0)}x_1 + w_2^{(0)}x_2 + \cdots + w_K^{(0)}x_K$$

$$\hat{y}_1(\mathbf{x}) = w_0^{(1)} + w_1^{(1)}x_1 + w_2^{(1)}x_2 + \cdots + w_K^{(1)}x_K$$

⋮

$$\hat{y}_{L-1}(\mathbf{x}) = w_0^{(L-1)} + w_1^{(L-1)}x_1 + w_2^{(L-1)}x_2 + \cdots + w_K^{(L-1)}x_K$$

Explorando uma notação matricial, podemos escrever que:

$$\hat{\mathbf{y}}(\mathbf{x}) = \boldsymbol{\Phi}(\mathbf{x})^T \mathbf{W},$$

onde $\hat{\mathbf{y}}(\mathbf{x}) = [\hat{y}_0(\mathbf{x}) \ \cdots \ \hat{y}_{L-1}(\mathbf{x})]$ (vetor linha) e

$$\mathbf{W} = \begin{bmatrix} w_0^{(0)} & \cdots & w_0^{(L-1)} \\ \vdots & \ddots & \vdots \\ w_K^{(0)} & \cdots & w_K^{(L-1)} \end{bmatrix}.$$

Agrupando todos os padrões de entrada na matriz

$$\Phi = \begin{bmatrix} \Phi(\mathbf{x}(0))^T \\ \vdots \\ \Phi(\mathbf{x}(N-1))^T \end{bmatrix},$$

podemos obter as L saídas para os N padrões de entrada por meio da relação:

$$\hat{\mathbf{Y}} = \Phi \mathbf{W},$$

em que $\hat{\mathbf{Y}} \in \mathbb{R}^{N \times L}$.

Critério: quadrados mínimos

$$\min_{\mathbf{W}} \frac{1}{NL} \sum_{i=0}^{N-1} \sum_{k=0}^{L-1} (y_k(i) - \hat{y}_k(i))^2$$

Reescrevendo o funcional com base em uma notação matricial, desprezando as constantes multiplicativas, obtemos:

$$\min_{\mathbf{W}} \text{tr}((\mathbf{Y} - \Phi \mathbf{W})^T (\mathbf{Y} - \Phi \mathbf{W})),$$

onde $\text{tr}(\cdot)$ representa o operador de traço de uma matriz.

A solução de quadrados mínimos assume a mesma forma que em (18):

$$\mathbf{W} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{Y} \quad (24)$$

8. Extensão para modelos lineares nos parâmetros

Todo o desenvolvimento matemático feito neste tópico para o problema de regressão linear também é válido para o caso em que o modelo de aproximação é não-linear, i.e., realiza um mapeamento não-linear das entradas para a saída –, mas possui uma dependência linear com respeito aos parâmetros ajustáveis.

Os modelos que se encaixam nesta categoria constroem uma aproximação do mapeamento verdadeiro por meio de uma combinação linear de funções base não-lineares, na forma:

$$\hat{y}(\mathbf{x}) = w_0 + w_1 b_1(\mathbf{x}) + \cdots + w_M b_M(\mathbf{x}),$$

onde $b_i(\mathbf{x}): \mathbb{R}^K \rightarrow \mathbb{R}$ denota a i -ésima função base.

A especificação de cada função base pode requerer a definição de alguns parâmetros (e.g., média e desvio padrão em uma função gaussiana). Porém, estes parâmetros não participam do treinamento do modelo, de modo que, do ponto de vista do algoritmo de treinamento, as funções base já estão pré-estabelecidas e permanecem fixas.

Sendo assim, as derivações feitas na Seção 3 podem ser prontamente estendidas para o caso de modelos não-lineares, mas lineares nos parâmetros.

Para isto, basta definir o vetor $\Phi(\mathbf{x})$ como sendo:

$$\Phi(\mathbf{x}) = [1 \ b_1(\mathbf{x}) \ \cdots \ b_M(\mathbf{x})]^T,$$

de acordo com o tipo de função base explorada.

Exemplos: filtros polinomiais (Volterra) (MATHEWS, 1991), *extreme learning machines* (ELMs) (HUANG, ZHU & SIEW, 2006), redes RBF (*radial-basis function*) (HAYKIN, 2008) com centros e dispersões pré-ajustados.

8.1. Regressão polinomial

Por simplicidade, vamos considerar um modelo de regressão polinomial no contexto de um problema de aproximação de um mapeamento $y = f(\mathbf{x}) : \mathbb{R} \rightarrow \mathbb{R}$. Ou seja, o vetor de entrada \mathbf{x} contém um único atributo, x_1 .

Modelo proposto: polinômio de grau M .

$$\hat{y}(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_1^2 + \dots + w_M x_1^M$$

Vetor de funções base:

$$\Phi(\mathbf{x}) = [1 \ x_1 \ x_1^2 \ \dots \ x_1^M]^T$$

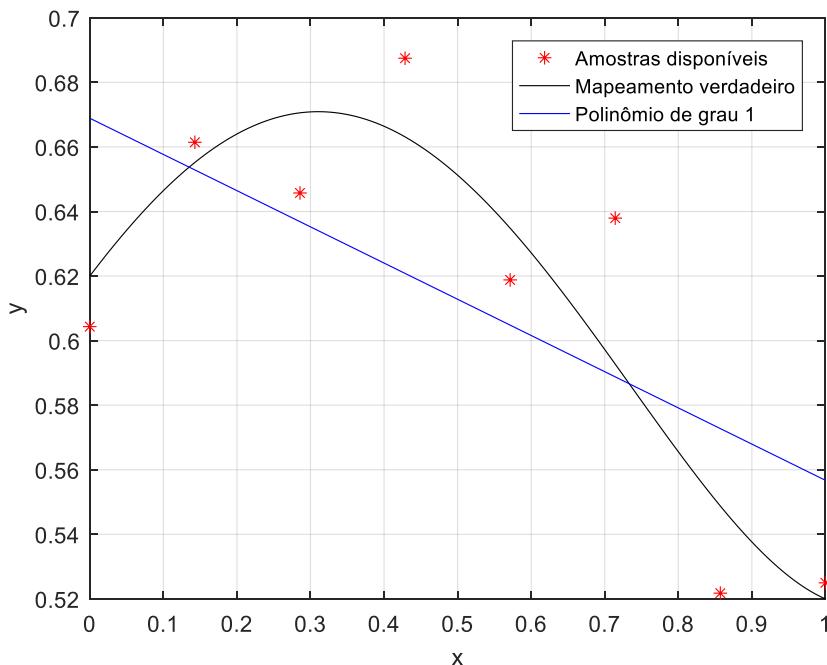
Exemplo:

$$y(i) = 0,62 + 0,3x_1(i) - 0,3x_1(i)^2 - 0,6x_1(i)^3 + 0,5x_1(i)^4 + 0,025\eta(i),$$

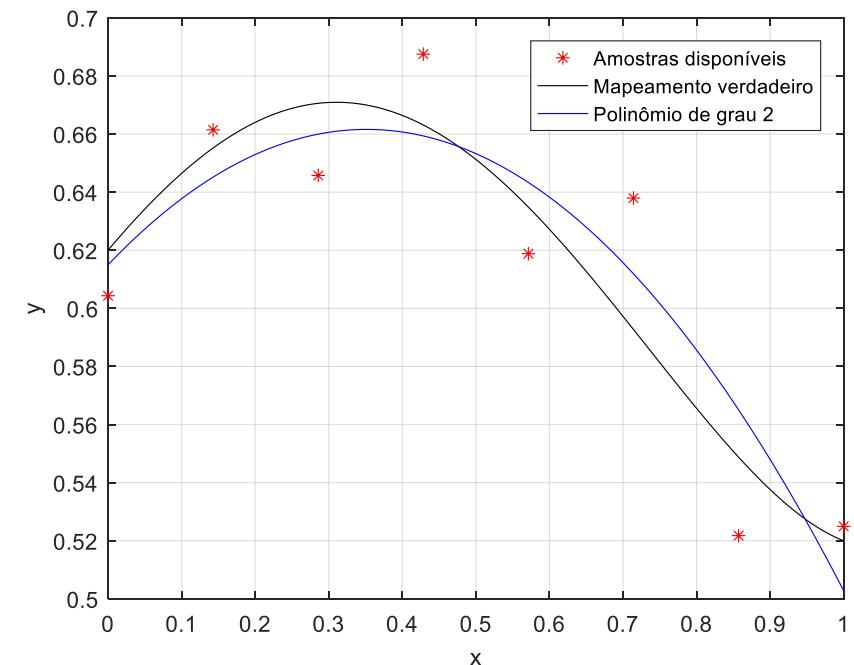
onde $\eta(i) \sim N(0,1)$.

Conjunto de treinamento: 8 amostras do par (x_1, y) .

Modelos testados: polinômios até a ordem 8.

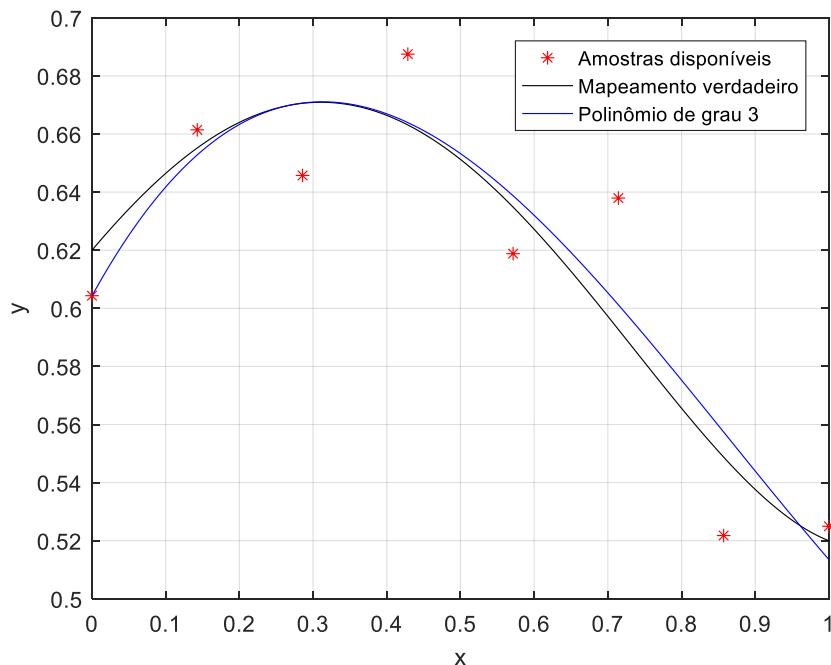


(a) Polinômio de grau 1

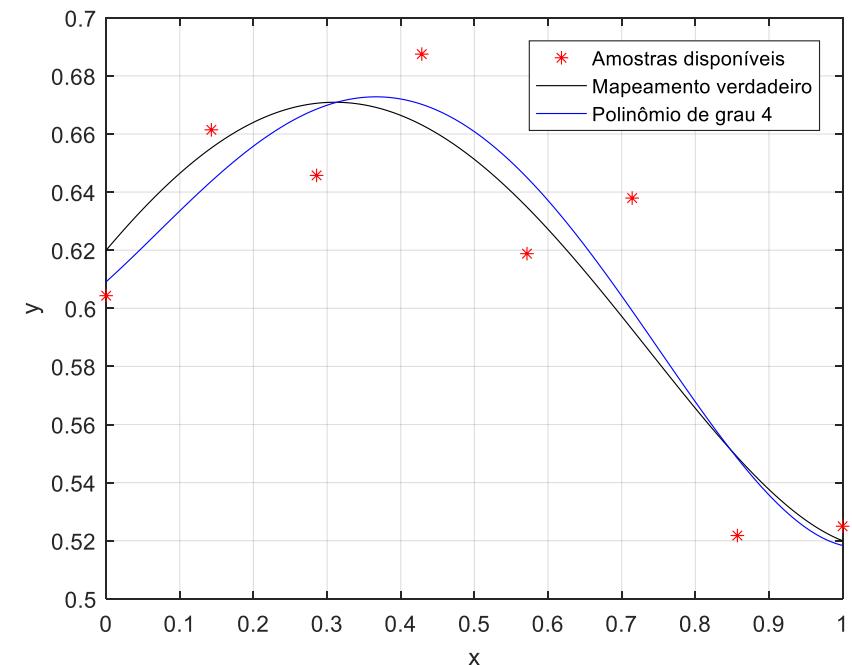


(b) Polinômio de grau 2

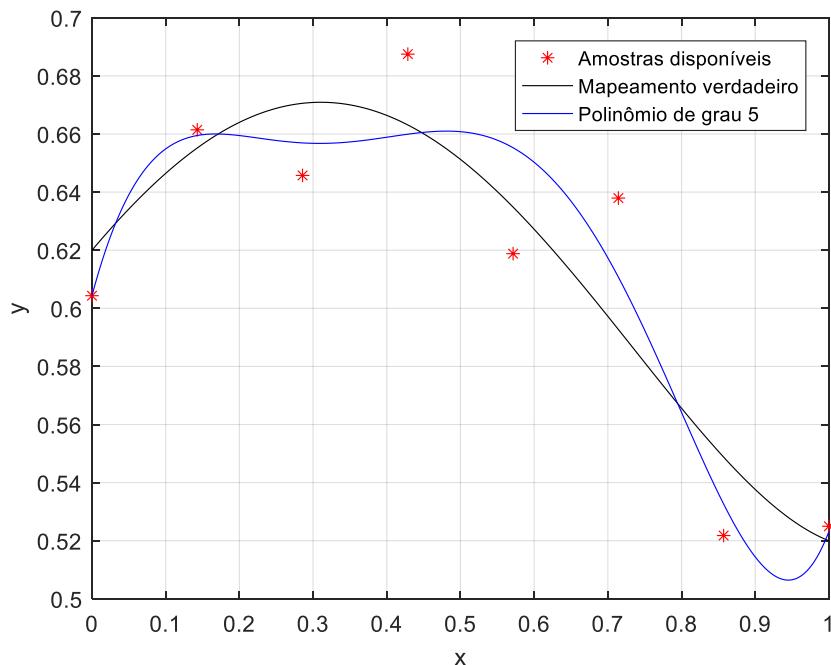
Neste caso, o modelo não possui flexibilidade suficiente para aproximar bem os dados disponíveis.



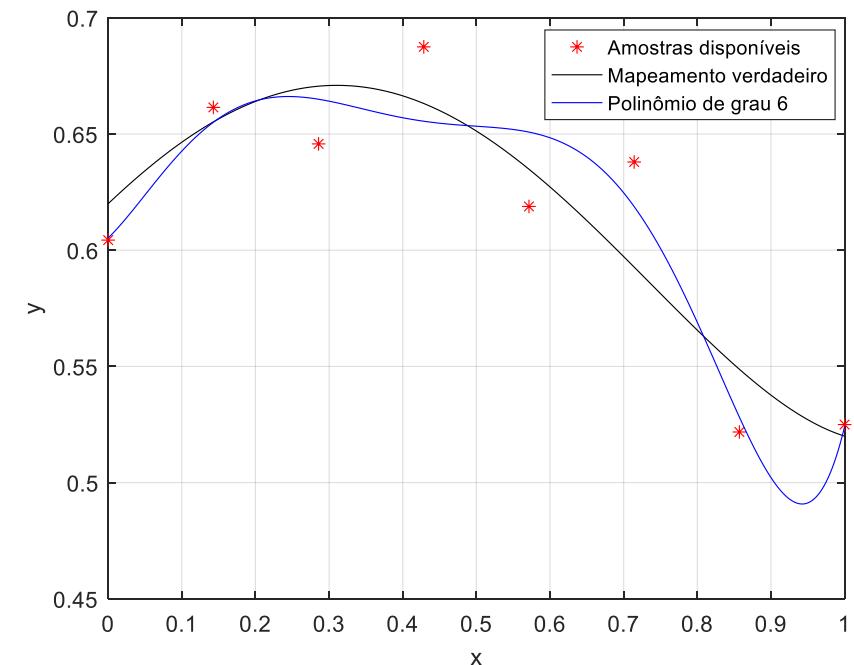
(c) Polinômio de grau 3



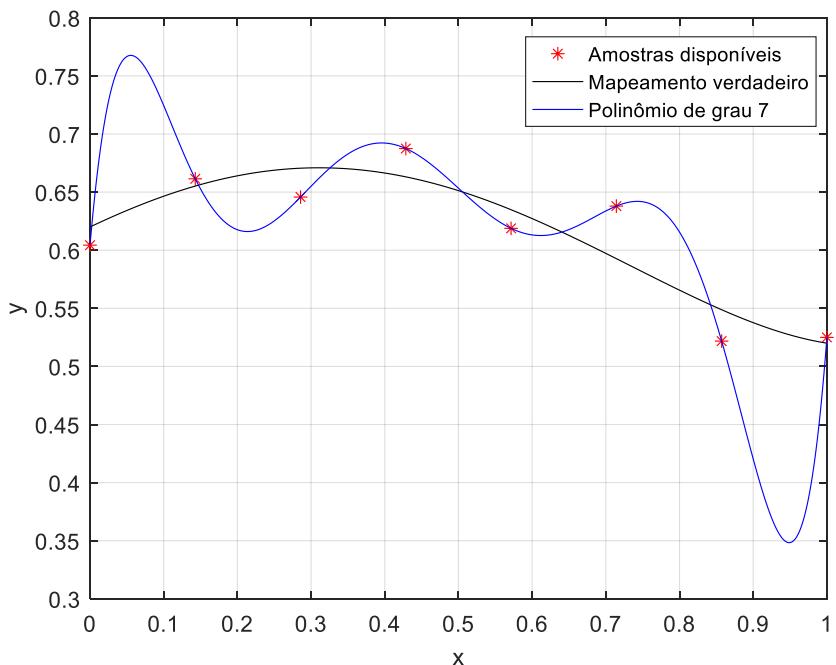
(d) Polinômio de grau 4



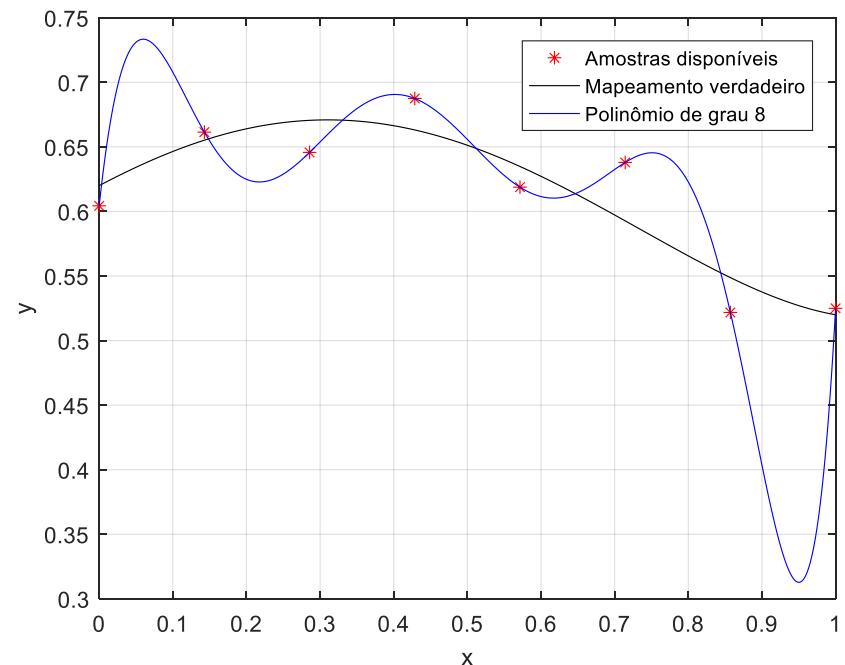
(e) Polinômio de grau 5



(f) Polinômio de grau 6



(g) Polinômio de grau 7



(h) Polinômio de grau 8

Nestes casos, a aproximação é quase perfeita nos pontos disponíveis. Contudo, nas demais regiões, o mapeamento gerado pelos polinômios se distancia bastante daquele que, de fato, está por trás das amostras coletadas. Isto significa que a resposta gerada por estes modelos apresentaria níveis de erro significativamente maiores quando eles fossem aplicados a dados novos.

Observações:

- Através deste exemplo, percebemos que, no fundo, ao construir um modelo de regressão, tem-se como real objetivo alcançar a melhor aproximação possível do mapeamento verdadeiro. Entretanto, o processo de ajuste dos parâmetros do modelo, ou, equivalentemente, o processo de treinamento dispõe somente de uma quantidade limitada de amostras.
- Além disso, notamos que é possível atingir uma excelente aproximação das amostras disponíveis e, ao mesmo tempo, produzir um mapeamento que difere significativamente daquele que realmente gerou os dados.
- No caso, ao aumentarmos a ordem do polinômio, percebemos uma progressiva redução do erro de aproximação em relação aos dados utilizados no treinamento. Porém, isso não necessariamente significa que estamos, de fato, construindo um modelo melhor.

Precisamos, portanto, de estratégias que forneçam indicativos de como o modelo se comporta ao aproximar o mapeamento verdadeiro como um todo (e não somente nas amostras de treinamento), e que também nos auxiliem a selecionar de forma confiável qual é o modelo mais adequado no problema de regressão.

9. Capacidade de generalização

Como o conjunto de dados é limitado, infinitos mapeamentos podem produzir o mesmo desempenho de aproximação nos pontos conhecidos, independente do critério de desempenho adotado.

- Esses mapeamentos alternativos vão diferir justamente nas regiões em que não há amostras disponíveis para diferenciá-los.
- Cada um destes mapeamentos, porém, alcançará um nível de aproximação diferente em relação ao mapeamento verdadeiro.

O conceito de capacidade de generalização está ligado à qualidade da aproximação produzida por um modelo quando exposto a dados de entrada não vistos durante o seu treinamento.

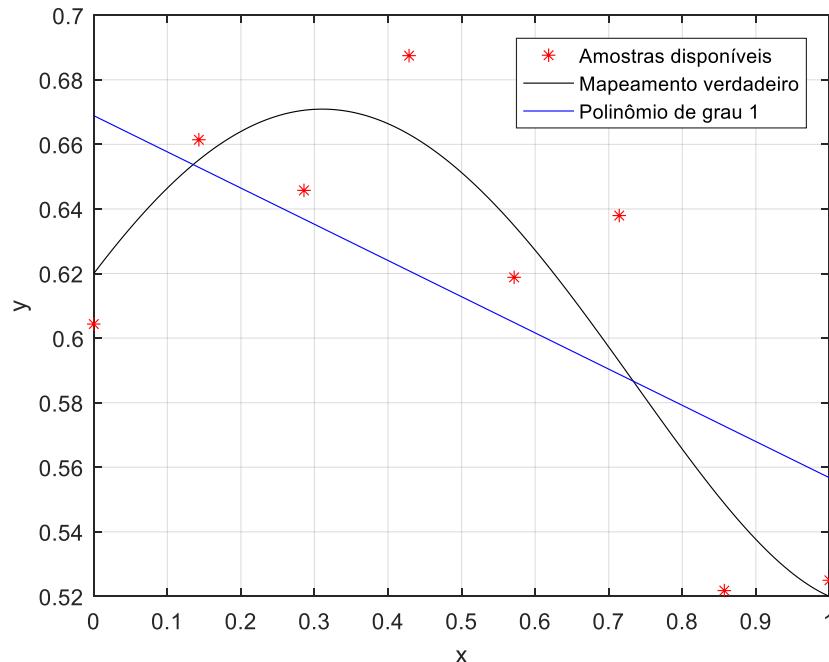
9.1. *Overfitting e underfitting*

Durante o processo de aprendizado, deseja-se evitar duas situações básicas que comprometem o uso de um modelo:

- **Sobreajuste (*overfitting*)**: a aproximação gerada pelo modelo se contorceu de forma excessiva a fim de reduzir ao máximo o erro junto aos dados de treinamento. Contudo, embora o erro de treinamento seja baixo, o modelo comete erros significativos quando recebe amostras inéditas de entrada.
- **Subajuste (*underfitting*)**: o modelo não foi capaz de aproximar adequadamente o mapeamento verdadeiro, nem mesmo nos dados utilizados no treinamento. Isto pode ocorrer pelo fato de o grau de flexibilidade do modelo ser insuficiente

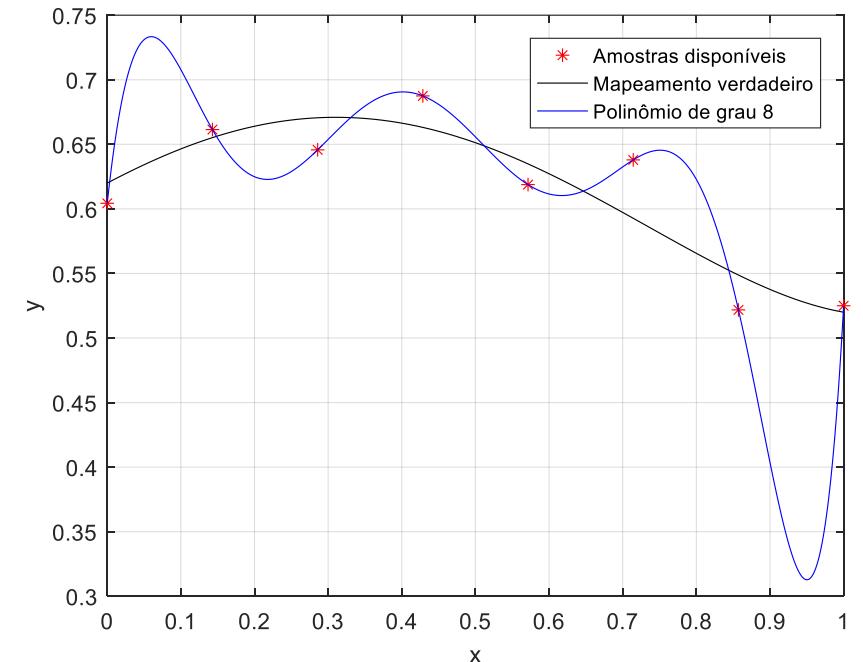
diante da complexidade do mapeamento a ser aproximado, ou, também, por problemas de convergência do processo de treinamento.

Retomando o exemplo de regressão polinomial:



(a) Polinômio de grau 1

↓
Underfitting



(b) Polinômio de grau 8

↓
Overfitting

9.2. Os três erros do processo de aproximação

Durante o processo de aproximação do mapeamento $y = f(\mathbf{x})$ por meio do modelo $\hat{y} = g(\mathbf{x}, \boldsymbol{\theta})$, devem ser considerados três tipos de erros (VAN DER SMAGT, 1994): o erro de *representação*, o erro de *generalização* e o erro de *computação*.

- *Erro de Representação*: considere o caso em que todo o conjunto amostral está disponível, $\{(\mathbf{x}(i); y(i))\}_{i=1}^{\infty}$. Considere também que, dado $\{(\mathbf{x}(i); y(i))\}_{i=1}^{\infty}$, é possível encontrar um conjunto de parâmetros ótimo $\boldsymbol{\theta}^*$. Neste caso, o erro vai depender da adequação e do nível de flexibilidade do modelo de aproximação $g(\mathbf{x}, \boldsymbol{\theta})$. Este erro é também conhecido como erro de aproximação, ou *efeito bias*.
- *Erro de Generalização*: em aplicações do mundo real, somente um número finito de amostras está disponível ou pode ser usado simultaneamente. Além disso, os dados podem conter ruído. Os valores de $f(\mathbf{x})$ para os quais nenhuma amostra está disponível devem, portanto, ser interpolados. Devido a estes fatores, pode

ocorrer um erro de generalização, também conhecido como erro de estimação, ou *variância*.

- *Erro de Computação:* Decorre do fato de nem sempre ser possível explorar devidamente o espaço de hipóteses. Também é conhecido como erro de otimização.
 - Algumas razões possíveis: mínimos locais, limitação dos recursos computacionais para a busca e uso de representação numérica de precisão finita, todas associadas ao problema de otimização vinculado.

10. Regularização

O termo *regularização* refere-se a procedimentos que visam obter um modelo de aproximação bem-comportado através da incorporação de informações adicionais ao processo de treinamento do modelo, na forma de restrições de suavidade junto

ao mapeamento, ou de penalizações proporcionais à norma do vetor de parâmetros (GIROSI, JONES & POGGIO, 1995; HASTIE, TIBSHIRANI & FRIEDMAN, 2009).

Benefício: o uso de regularização se mostra atraente na medida em que pode reduzir a possibilidade de ocorrer sobre-ajuste (*overfitting*), melhorando o processo de generalização e, em última análise, o desempenho efetivo do modelo.

Três técnicas serão aqui destacadas: *ridge regression*, LASSO e *elastic net*. Todas elas introduzem restrições ligadas a alguma norma do vetor de parâmetros, sendo, por este motivo, também conhecidas como técnicas de *shrinkage*.

10.1. Ridge regression

Em vez de minimizar apenas o erro quadrático junto ao conjunto de treinamento, como estabelecido em (9) e (12), agora há a introdução de um termo de penalização proporcional à norma Euclidiana do vetor de parâmetros (TIKHONOV, 1963):

$$\min_{\mathbf{w}} \|\mathbf{e}\|^2 + \lambda \|\mathbf{w}\|^2, \quad (25)$$

onde $\lambda \geq 0$ é o coeficiente de regularização.

- Conforme cresce o valor de λ , maior é a redução da magnitude dos coeficientes do modelo, até o limite em que $\lambda \rightarrow \infty$, quando $w_i \rightarrow 0$.
- O parâmetro λ controla, portanto, o *trade-off* entre a redução do erro de aproximação e a limitação da magnitude dos parâmetros.

É possível, também, reescrevermos o problema envolvido em *ridge regression* como um problema de otimização com restrições*:

$$\begin{aligned} & \min_{\mathbf{w}} \|\mathbf{e}\|^2 \\ & s. a. \quad \|\mathbf{w}\|^2 \leq \chi, \end{aligned} \quad (26)$$

* Frequentemente, o coeficiente w_0 não é levado em consideração no cálculo do termo de penalização ($\|\mathbf{w}\|^2$) ou na restrição de norma, para evitar que o resultado dependa da origem escolhida para o vetor desejado (HASTIE, TIBSHIRANI & FRIEDMAN, 2009). Assim, em (25) e (26), estamos considerando que $\|\mathbf{w}\|^2 = \sum_{i=1}^K w_i^2$.

onde χ restringe a magnitude dos coeficientes e é inversamente proporcional ao parâmetro λ de (25).

Interessantemente, o funcional definido em (25) continua sendo quadrático com respeito aos coeficientes do modelo. Logo, é possível obter uma solução em forma fechada muito semelhante àquela baseada em pseudo-inversa:

$$\mathbf{w} = (\Phi^T \Phi + \lambda \mathbf{I}')^{-1} \Phi^T \mathbf{y}, \quad (27)$$

onde $\mathbf{I}' = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$.

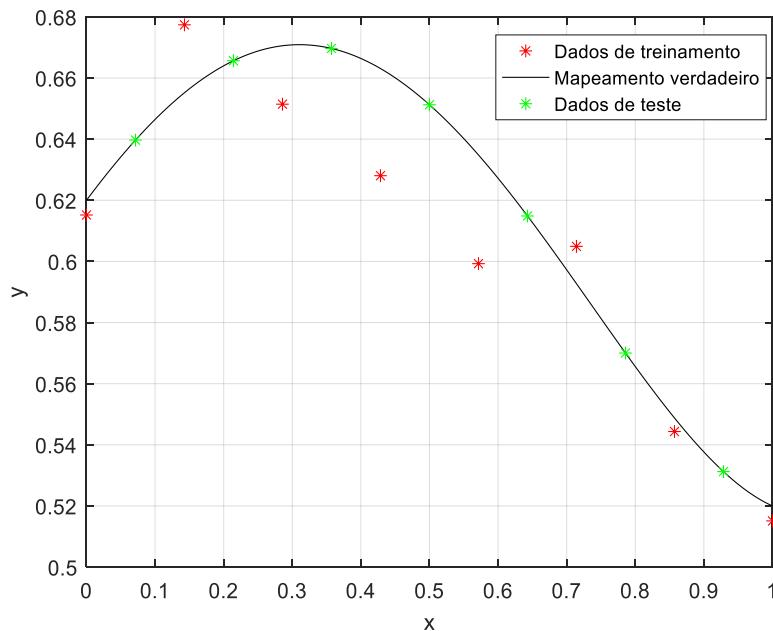
Nota: mesmo que a matriz Φ não possua posto completo, a inversa em (27) sempre existe por conta da adição do termo de regularização à diagonal principal.

Questão: Como determinar o valor ótimo de λ ?

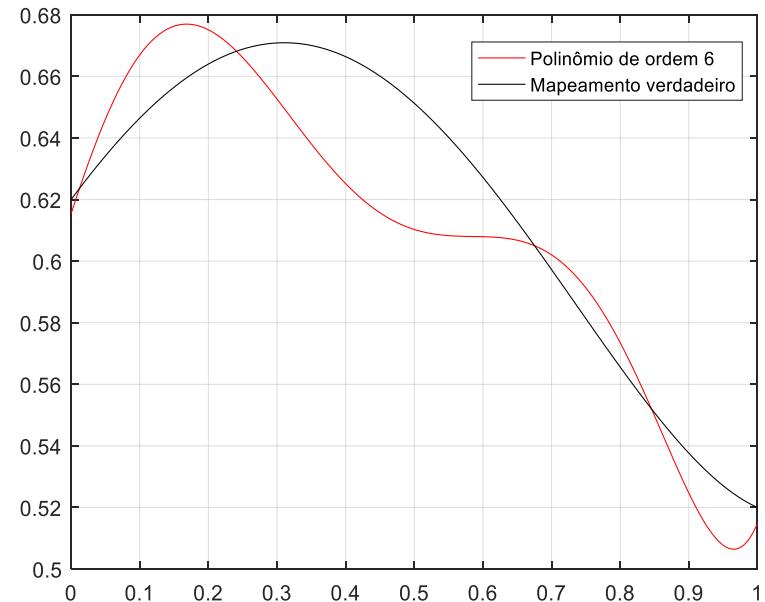
- Lembrando que se deseja maximizar a capacidade de generalização do modelo, é preciso ter uma forma apropriada de estimar o erro de generalização do modelo para cada valor atribuído a λ .
- Sugere-se aqui o uso de uma busca unidimensional (por exemplo, via seção áurea), acompanhada de uma estratégia de validação cruzada.

Exemplo: regressão polinomial

- Considere novamente o cenário descrito na Seção 8.1. Vamos analisar o impacto da regularização sobre o modelo polinomial de ordem $M = 6$.



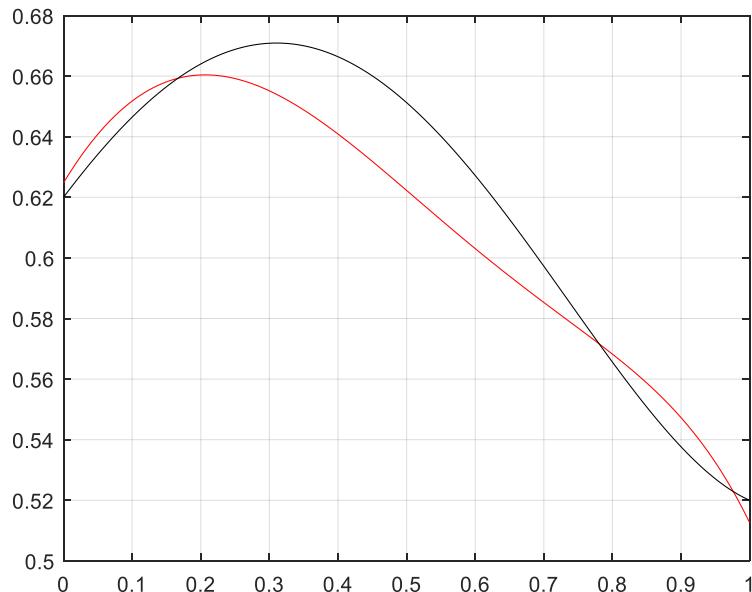
(a) Dados de treinamento e de teste



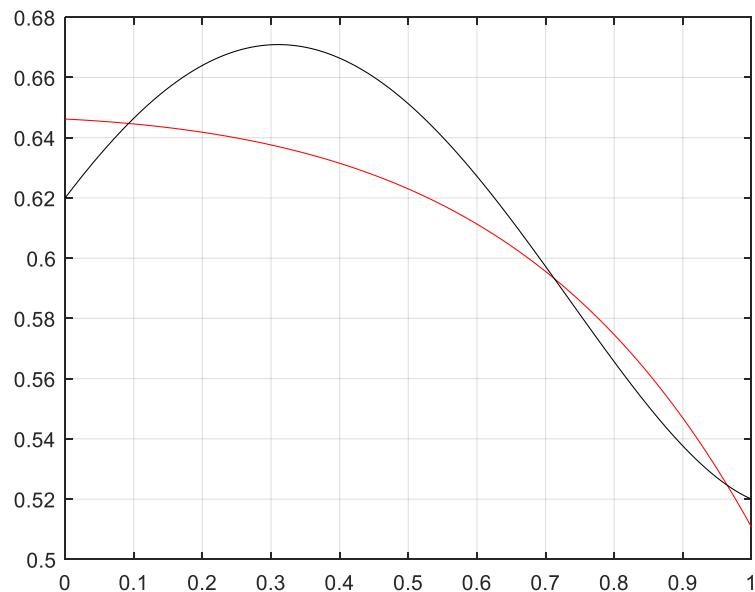
(b) Mapeamento gerado pelo polinômio ótimo



Overfitting



(c) Mapeamento gerado pelo polinômio ótimo
com $\lambda = 10^{-4}$



(d) Mapeamento gerado pelo polinômio ótimo
com $\lambda = 10^{-1}$



- A imposição de uma penalidade proporcional à norma do vetor de coeficientes deixou o modelo mais “rígido”, fazendo com que o mapeamento por ele gerado não se contorcesse tanto na tentativa de aproximar os dados de treinamento.

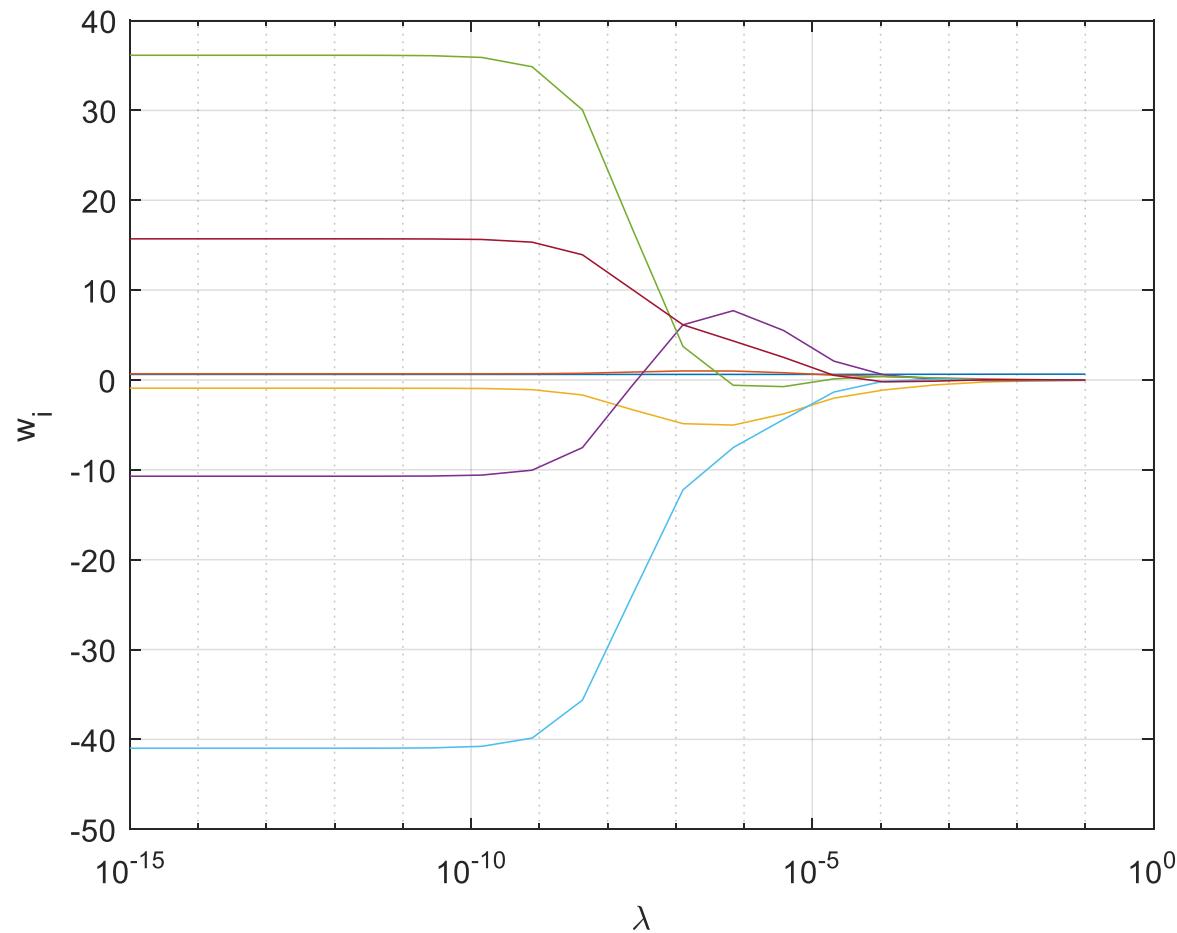


Figura 8 – Evolução dos coeficientes w_i do modelo em função do fator de regularização.

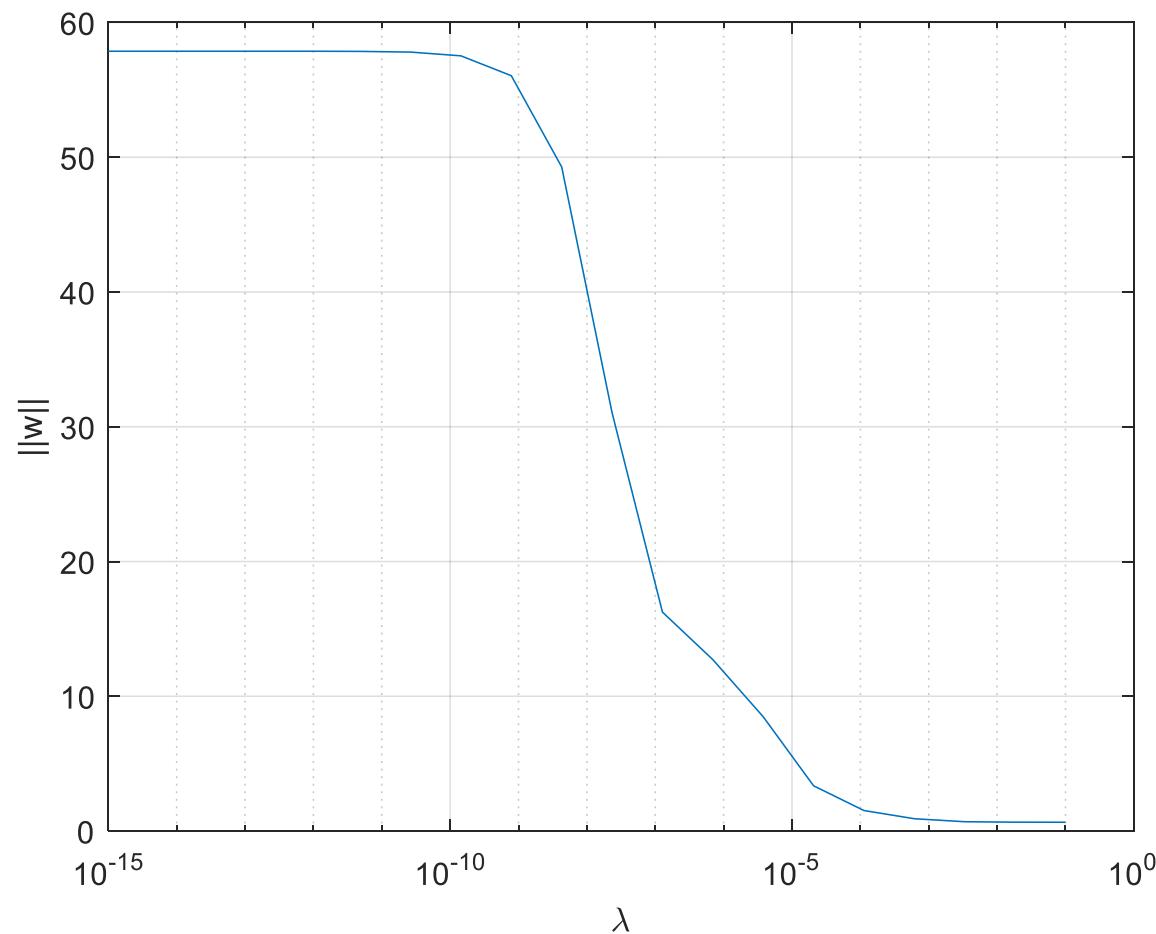


Figura 9 – Evolução da norma do vetor de coeficientes w .

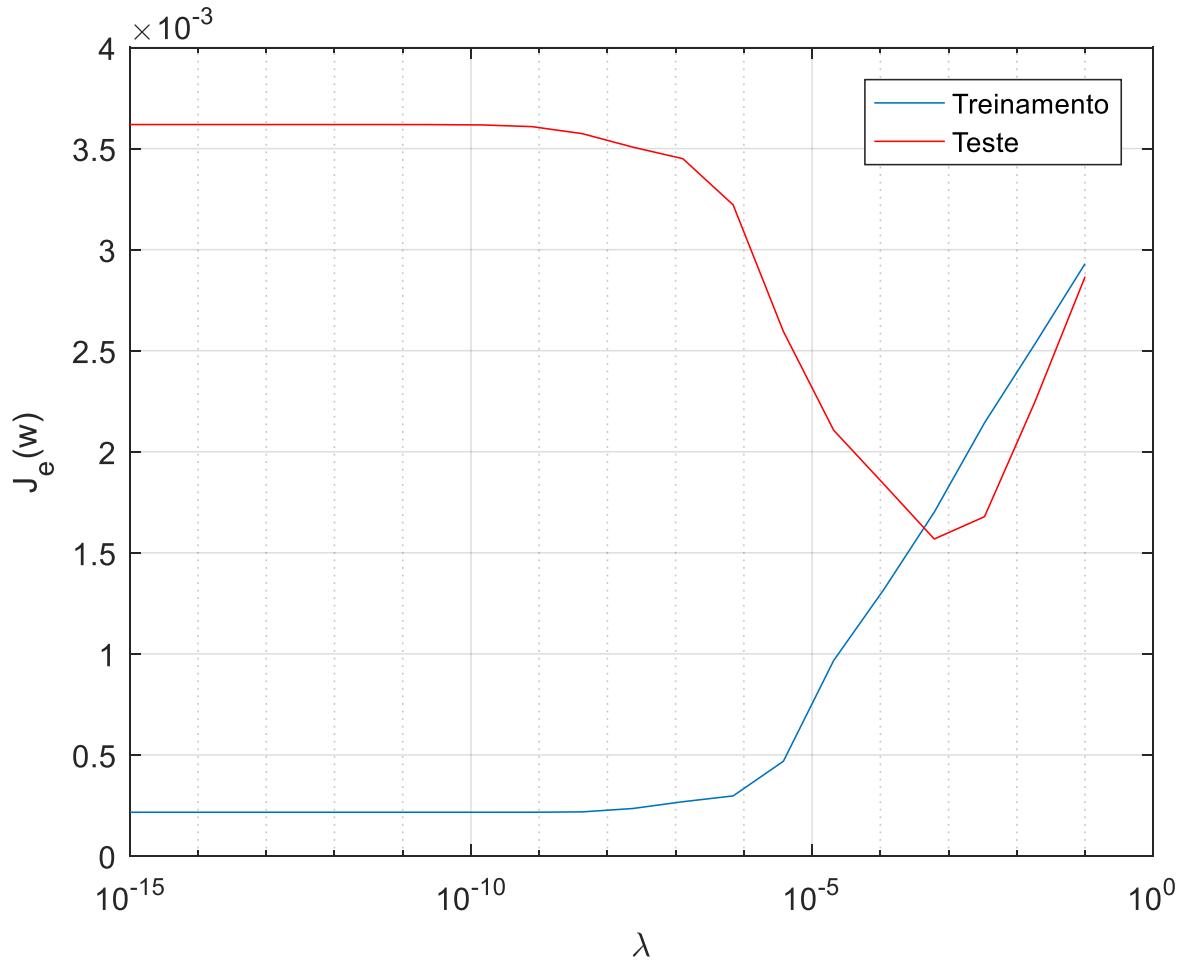


Figura 10 – Curvas de evolução do erro de treinamento e do erro junto aos dados de teste em função do fator de regularização.

10.2. LASSO

A técnica denominada LASSO (*Least Absolute Shrinkage and Selection Operator*) trabalha com um termo de penalização proporcional à norma L_1 do vetor de parâmetros (HASTIE, TIBSHIRANI & FRIEDMAN, 2009):

$$\min_{\mathbf{w}} \|\mathbf{e}\|^2 + \lambda \|\mathbf{w}\|_1, \quad (28)$$

onde $\|\mathbf{w}\|_1 = \sum_{i=1}^P |w_i|$ e $\lambda \geq 0$ ^t.

A **vantagem** do LASSO está principalmente no fato de que múltiplos elementos do vetor \mathbf{w} acabam sendo anulados, o que sugere a ocorrência implícita de um processo de seleção de variáveis, e leva a modelos de aproximação mais parcimoniosos.

Desvantagem: o problema em (28) não admite solução em forma fechada.

^t Note mais uma vez o coeficiente w_0 não participa da norma envolvida na penalização. Na realidade, a solução para w_0 é o próprio valor médio da saída desejada.

10.2.1. Interpretação geométrica

Na figura abaixo, temos as curvas de nível do funcional de erro clássico do problema de regressão linear ($\|\mathbf{e}\|^2$), bem como as regiões do espaço em que as restrições $\|\mathbf{w}\|^2 \leq \chi$ (direita) e $\|\mathbf{w}\|_1 \leq \chi$ (esquerda) são válidas, considerando o caso em que o vetor \mathbf{w} possui dois parâmetros sujeitos à regularização.

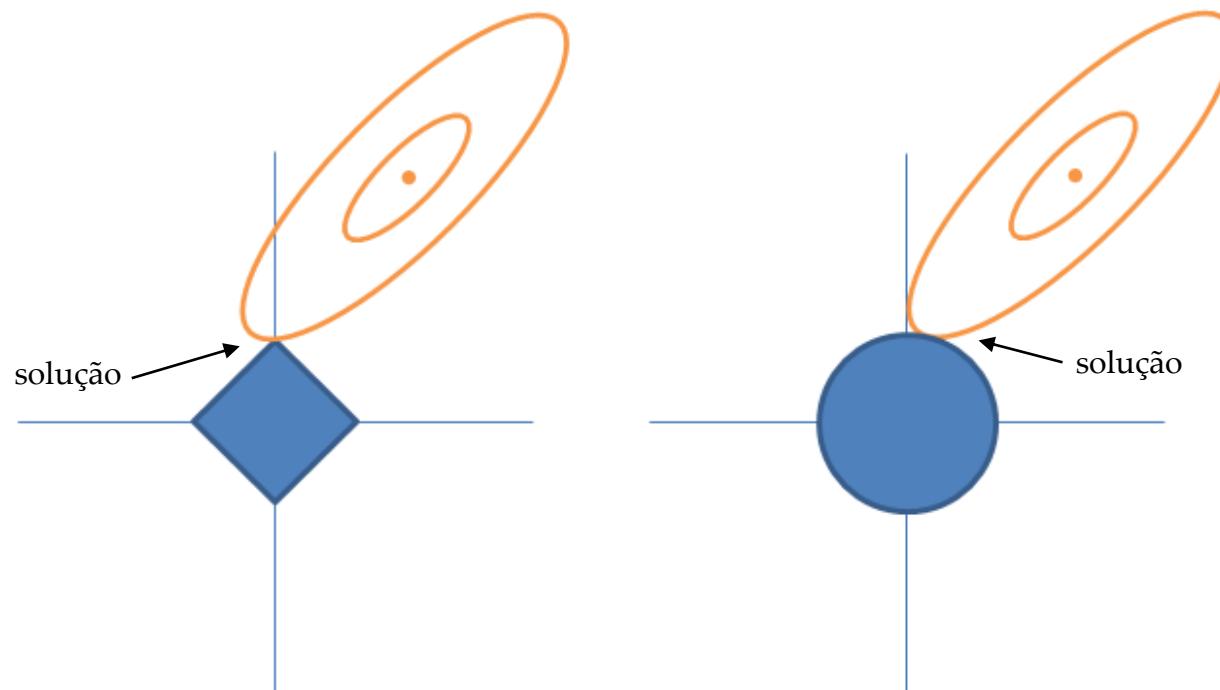


Figura 11 – Interpretação geométrica para o LASSO, inspirada em HASTIE, TIBSHIRANI & FRIEDMAN (2009).

A solução para ambos os métodos corresponde ao primeiro ponto em que as curvas de nível elípticas do funcional de erro interceptam a região de factibilidade das respectivas restrições.

- A existência de cantos nas curvas de nível de $\|\mathbf{w}\|_1$ aumenta as chances de alguns parâmetros assumirem o valor zero.

10.2.2. Interpretação estatística (Bayesiana)

As estratégias LASSO e *ridge regression* também podem ser vistas como casos particulares de estimação bayesiana dos parâmetros do modelo de regressão linear.

- LASSO: a distribuição *a priori* considerada para o vetor de parâmetros é Laplaciana.
- *Ridge regression*: a distribuição *a priori* de w_i é Gaussiana.

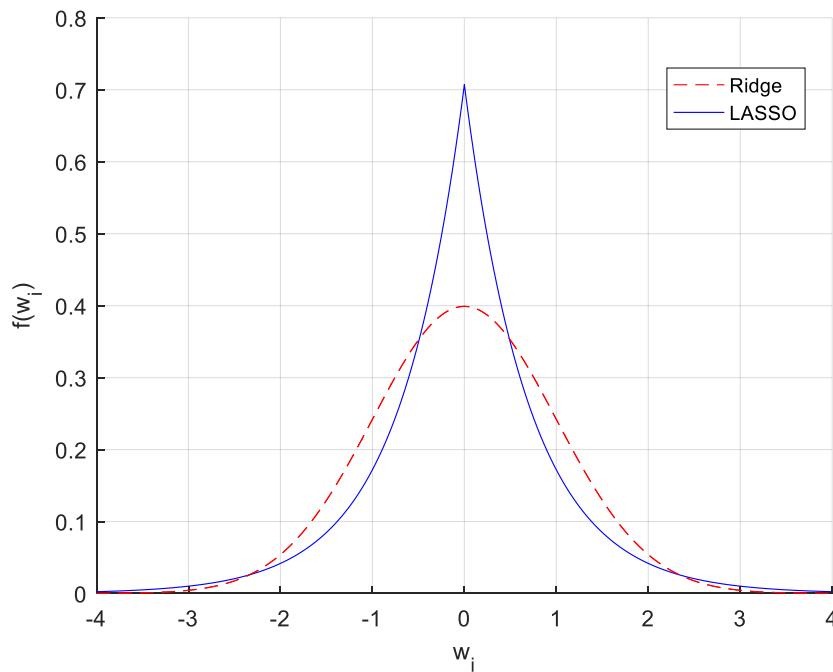


Figura 12 – Ilustração das distribuições Gaussiana e Laplaciana com média nula e variância unitária.

10.3. *Elastic net*

Uma solução intermediária entre *ridge regression* e LASSO é a *elastic net* (ZOU & HASTIE, 2005), na qual se faz uma combinação convexa entre as penalizações baseadas nas normas L_1 e L_2 do vetor de parâmetros.

$$\min_{\mathbf{w}} \|\mathbf{e}\|^2 + \lambda \{\kappa \|\mathbf{w}\|_1 + (1 - \kappa) \|\mathbf{w}\|_2^2\}, \quad (29)$$

onde $\kappa \in [0,1]$.

- A definição dos coeficientes λ e κ pode ser feita por meio de uma busca visando minimizar o erro junto a um conjunto de dados de validação.
- As propostas anteriores de solução para regressão linear podem ser obtidas através da atribuição de valores específicos para λ e κ :
 - Quadrados mínimos irrestrito: $\lambda = 0$;
 - *Ridge Regression*: $\lambda > 0$ e $\kappa = 0$;
 - LASSO: $\lambda > 0$ e $\kappa = 1$.
 - *Elastic net*: $\lambda > 0$ e $0 < \kappa < 1$.

11. Validação cruzada e *early stopping*

Outra estratégia que auxilia na obtenção de um modelo com melhor capacidade de generalização, denominada **validação cruzada** (CV, do inglês *cross-validation*), consiste em dividir o conjunto de amostras para treinamento em dois:

- O primeiro conjunto será efetivamente empregado no ajuste dos parâmetros do modelo – *conjunto de treinamento*.
- O segundo conjunto será utilizado para monitorar a capacidade de generalização do modelo – *conjunto de validação*.

Também é usual reservar uma porção de dados para uma etapa de teste, aplicada após o modelo já ter sido ajustado e validado – *conjunto de teste*.

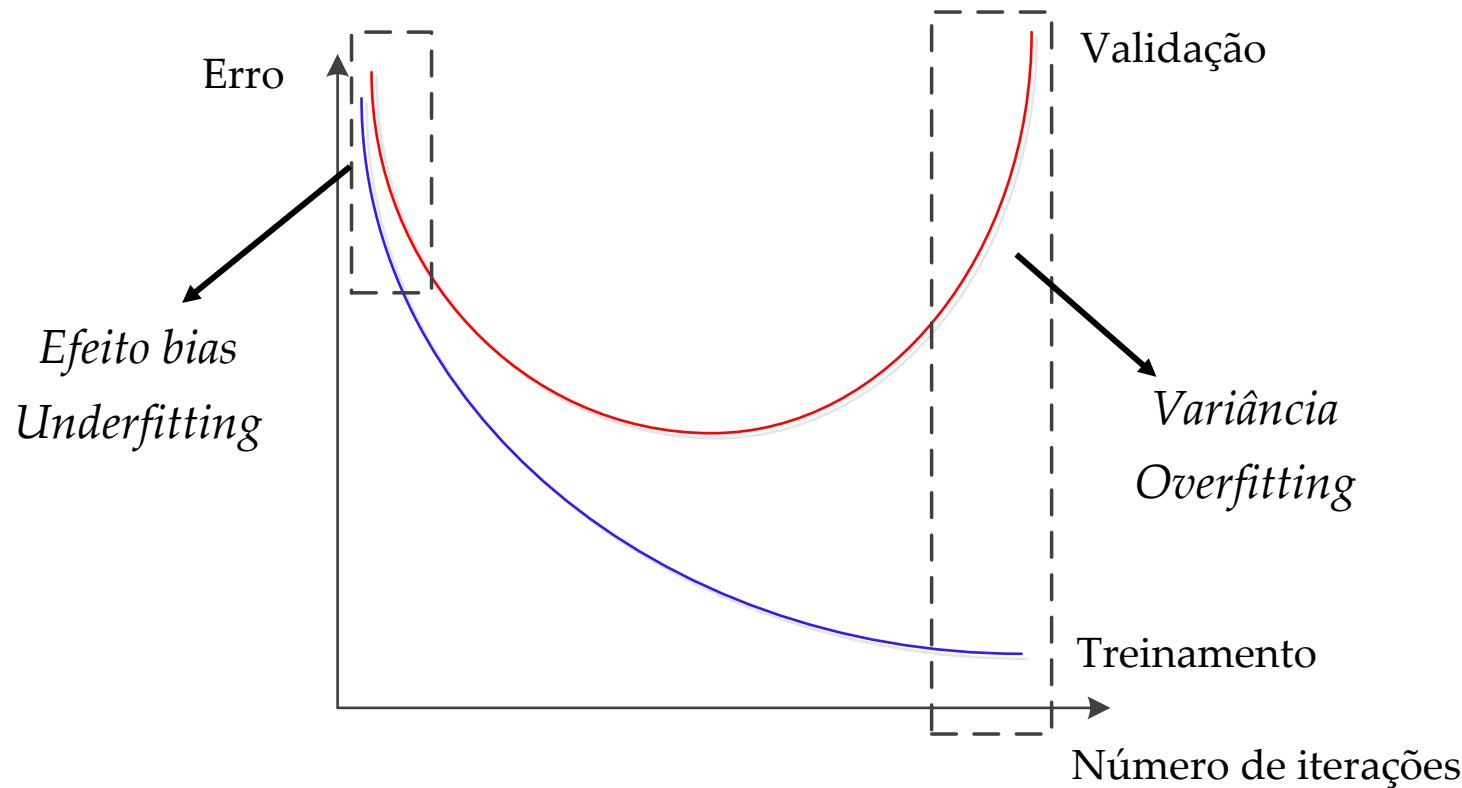
Nota: deve-se assegurar que os conjuntos sejam suficientemente representativos do mapeamento que se pretende aproximar.

Tendo à disposição um conjunto de amostras separado para validação, durante o treinamento do modelo, monitora-se também o seu desempenho junto a estes dados de validação ao longo das iterações.

Premissa: observar o desempenho do modelo junto aos dados de validação fornece um indicativo antecipado de como ele se comportará quando exposto a amostras não vistas no treinamento. Em outras palavras, o desempenho de validação é interpretado como uma estimativa do erro de generalização.

- Assim, espera-se que minimizar o erro junto ao conjunto de validação implique em aumentar a capacidade de generalização.
- Logo, espera-se que a configuração do modelo que leva ao menor erro junto ao conjunto de validação, o qual não é usado para o ajuste dos parâmetros, tenha o melhor desempenho possível junto a novas amostras.

Comportamento “típico” dos erros de treinamento e validação:



Bias: o erro de treinamento tende a ser elevado, assim como o erro de validação.

Variância: embora o erro de treinamento seja baixo, o erro de validação é elevado (GEMAN, BIENENSTOCK & DOURSAT, 1992).

11.1. *Early stopping*

É uma abordagem de regularização empregada para tentar evitar o sobreajuste de modelos adaptados com métodos iterativos, a qual recomenda que se interrompa o treinamento quando o erro de validação começar a crescer de forma sistemática.

Existem diversas propostas na literatura para definir o critério de parada a ser explorado no *early stopping* (PRECHELT, 1998; BISHOP, 2006).

- Uma estratégia simples consiste em interromper o treinamento quando o valor do erro de validação aumenta por P iterações sucessivas, sendo o parâmetro P denominado paciência.
- **Problema:** nem sempre o erro de validação apresenta um comportamento bem definido, como aquele ilustrado na seção anterior. Como a curva do erro de validação pode oscilar bastante e apresentar um comportamento pouco

previsível, nem sempre é pertinente desenvolver detectores automáticos de mínimos e encerrar o treinamento ali.

- Uma alternativa é permitir que o treinamento prossiga, mas armazenar os parâmetros associados ao mínimo erro de validação, os quais serão considerados como a solução para o modelo ao final do treinamento.

Ponto de contato: *early stopping* também controla implicitamente a norma do vetor de parâmetros.

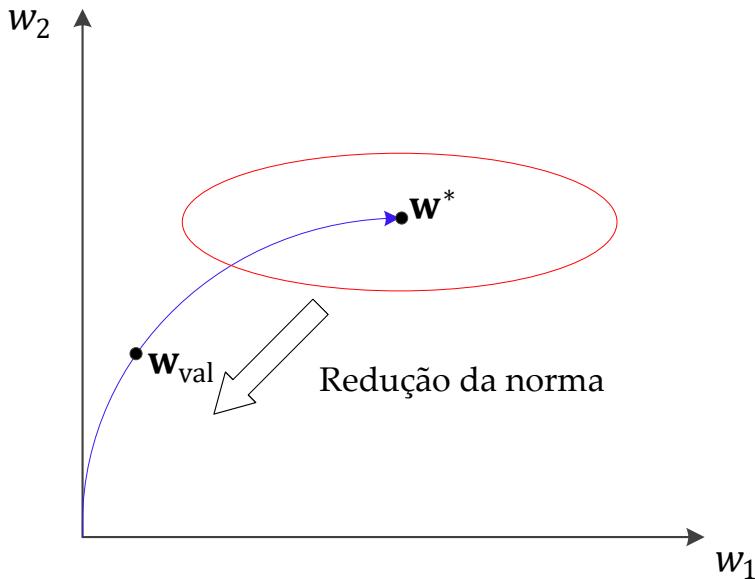


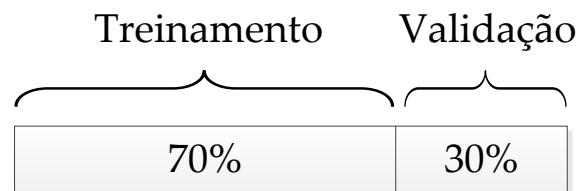
Figura 13 – Trajetória do vetor de parâmetros do modelo até convergir para o ponto \mathbf{w}^* de mínimo erro de treinamento. O ponto \mathbf{w}_{val} indica a configuração que atingiu o menor erro de validação.

11.2. Estratégias para validação cruzada

As técnicas aqui apresentadas são, também, formas de se realizar re-amostragem (*sampling*) (ALPAYDIN, 2013) a partir de um conjunto limitado de dados, a fim de se ter uma estimativa suficientemente precisa de propriedades estatísticas do modelo, como o erro de generalização.

11.2.1. *Holdout*

Dividimos o conjunto de dados em p (%) para treinamento e $(1-p)$ (%) para validação.



Desvantagens: alguns dados disponíveis sempre serão usados no ajuste dos parâmetros, enquanto outros nunca serão usados para este fim, pois compõem o

conjunto de validação. Além disso, não se tem qualquer indicativo sobre como o modelo varia com diferentes dados de treinamento.

11.2.2. Amostragem aleatória

Várias divisões do conjunto de dados no perfil *holdout* são feitas de forma independente. Considera-se, então, a média e o desvio padrão dos resultados obtidos pelo modelo nos conjuntos de validação.

11.2.3. Validação cruzada com k pastas

Uma técnica mais elaborada, denominada *k-fold cross validation*, consiste em dividir o conjunto de amostras disponíveis para treinamento em k pastas e realizar k treinamentos, cada um considerando $k-1$ pastas para o ajuste dos parâmetros e 1 pasta para a validação. Com este procedimento, toda amostra disponível vai aparecer $k-1$ vezes no conjunto de treinamento e 1 vez no conjunto de validação.

- Repare que os k conjuntos de treinamento terão composição distinta, assim como os k conjuntos de validação. O desempenho do modelo é tomado como a média dos desempenhos nas k pastas de validação.

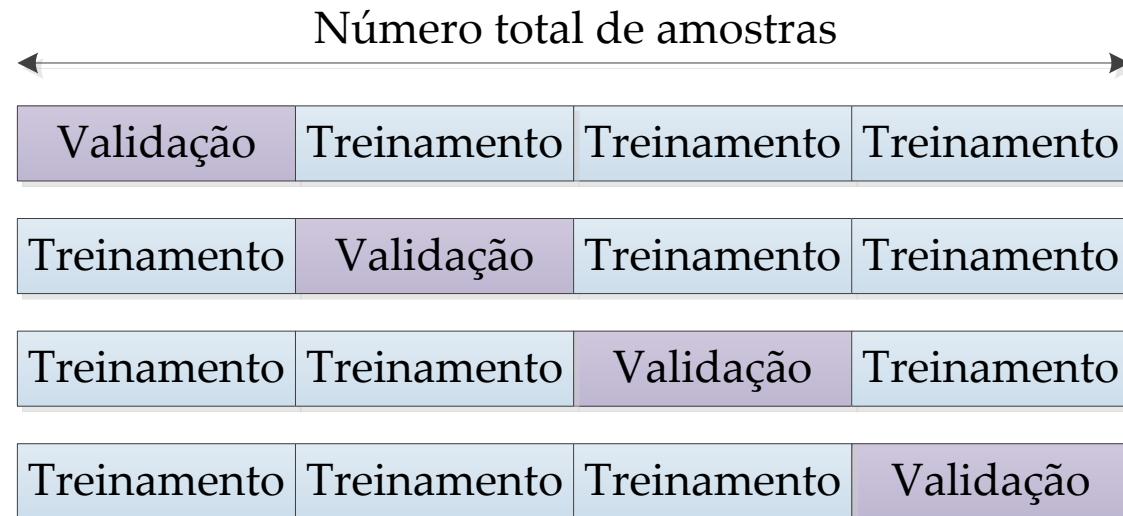


Figura 14 – Esquema de divisão das amostras disponíveis para treinamento na abordagem de validação cruzada com k -pastas (k -fold cross-validation), com $k = 4$.

Procedimento:

- Particione o conjunto de dados em k folds (sem sobreposição) de mesmo tamanho.
- Para cada $j = 1, \dots, k$:

- Ajuste os parâmetros do modelo considerando todos os dados, exceto aqueles que pertencem ao j -ésimo *fold*.
 - Compute as saídas estimadas do modelo para os dados do j -ésimo *fold*, e calcule o erro de validação.
- Ao final, o desempenho do modelo é dado pela média dos erros de validação calculados para cada *fold*.

Comentários:

- **Estratificação:** procura assegurar que as classes sejam representadas na mesma proporção que aquela exibida no conjunto completo dos dados quando os subconjuntos (*folds*) são formados.
- O caso em que $k = N$ é conhecido como *leave one out validation*.

11.2.4. *Bootstrapping*

Neste caso, criamos k conjuntos de treinamento fazendo uma amostragem com reposição dos dados disponíveis. Os dados não-amostrados em cada realização formam o respectivo conjunto de validação. O resultado do modelo é tomado com base na média e no desvio padrão dos desempenhos nos conjuntos de validação.

11.2.5. Aplicações: seleção de modelos e de hiperparâmetros

Uma vez que a validação cruzada nos proporciona uma forma de inferir a qualidade da generalização de um modelo, esta técnica também pode ser explorada durante a escolha de valores para hiperparâmetros de um modelo, ou, analogamente, para a seleção do modelo mais apropriado para o problema.

Exemplo: regressão polinomial com *holdout*

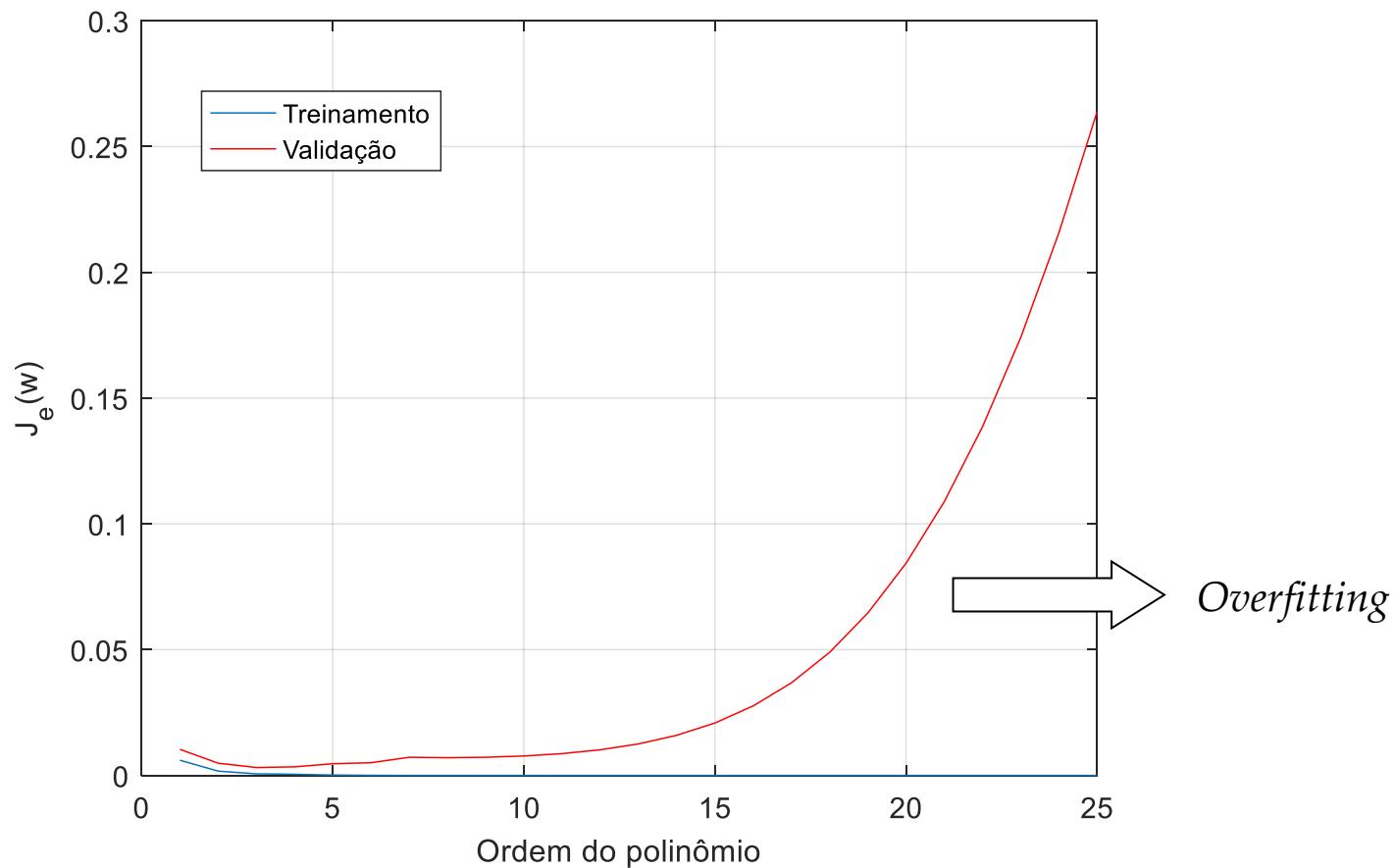


Figura 15 – Curvas de evolução do erro quadrático junto aos conjuntos de treinamento e validação em função da ordem do polinômio.

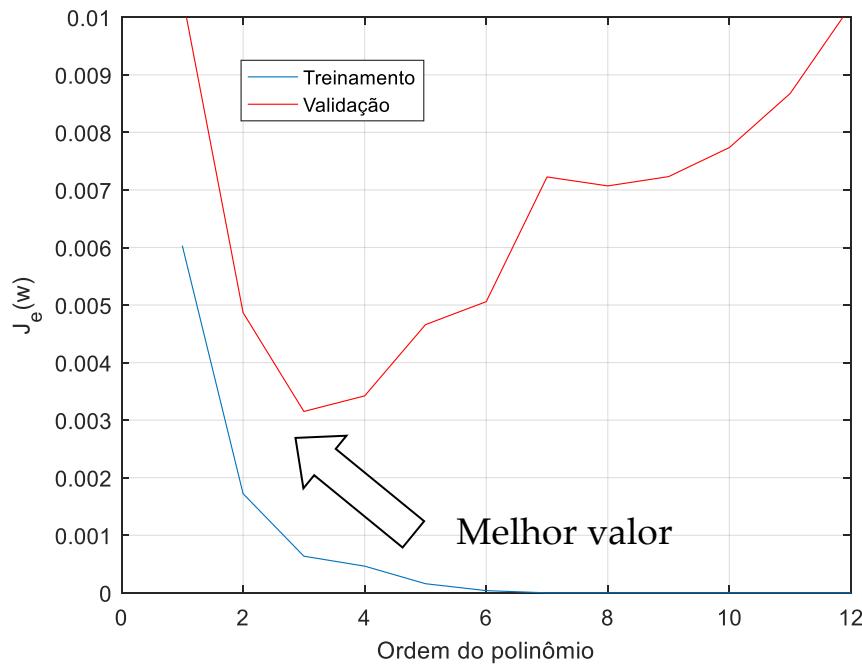


Figura 16 – Curvas de evolução do erro quadrático junto aos conjuntos de treinamento e validação em função da ordem do polinômio, com destaque para a parte inicial.

Questão: o uso de CV não nos fornece automaticamente a configuração “ótima” dos parâmetros ajustáveis do modelo (e.g., os coeficientes da combinação linear).

- No caso do *k-fold* CV, para cada conjunto de treinamento, um modelo com parâmetros diferentes usualmente é obtido.
- **Soluções:**
 - Construir um *ensemble* dos k modelos (ou de um subconjunto deles devidamente selecionado), desde que haja recursos computacionais para tanto.
 - Treinar o modelo usando todos os dados – ou seja, todas as amostras que anteriormente foram empregadas para geração dos *folds* –, explorando *early stopping* e/ou regularização para evitar *overfitting*. Daí, o modelo estará pronto para ser posto à prova com o conjunto de teste.

12. Seleção de variáveis

Definição: é o processo de selecionar um subconjunto de atributos (ou variáveis) de entrada mais relevantes ou mais informativos para uso no modelo (GUYON & ELISSEEFF, 2003).

Este tipo de estratégia parte da premissa que os dados podem conter alguns atributos que são redundantes, ou até mesmo irrelevantes, para a realização da tarefa (e.g., estimativa do preço de imóveis), os quais podem ser removidos sem que isto implique uma perda significativa de informação. Com efeito, o uso de métodos de seleção de variáveis pode favorecer a capacidade de generalização de modelos, evitando o *overfitting*.

Outros benefícios:

- Facilitar a visualização e a compreensão dos dados.

- Minimizar o efeito da maldição da dimensionalidade: menos amostras podem ser suficientes para descrever o comportamento geral do mapeamento a ser aproximado.
- Reduzir os custos envolvidos no armazenamento dos dados e no treinamento do modelo.

Veremos neste tópico duas das abordagens mais comuns de seleção de variáveis.

12.1. Filtros

Fazem a pré-seleção das variáveis de entrada sem considerar o modelo escolhido para a tarefa. Através do cálculo de alguma métrica de informação entre cada variável de entrada e a saída desejada, faz-se a seleção daquelas que se mostram as mais informativas, em número pré-determinado ou considerando o perfil dos valores observados para a métrica adotada.

Exemplos de métricas utilizadas: correlação, correlação não-linear, informação mútua.

12.2. *Wrappers*

Os *wrappers*, por sua vez, recorrem ao modelo escolhido para realizar a tarefa, utilizando o desempenho alcançado pelo modelo como indicativo da qualidade de cada combinação de variáveis de entrada.

Em outras palavras, cada subconjunto de variáveis em análise é efetivamente utilizado para treinamento do modelo, sendo que o erro (médio) observado no conjunto de validação revela a sua relevância.

Estratégias:

- Busca exaustiva: viável somente em situações em que o número de variáveis candidatas é bem reduzido.

- *Forward selection*: estratégia gulosa (*greedy*) de construção do subconjunto de variáveis. Começando do conjunto vazio, uma nova variável de entrada é acrescentada ao subconjunto ótimo em cada iteração do processo, a saber, a variável que faz com que o modelo atinja o menor erro de validação naquela iteração.

- Opção mais viável (menos custosa) quando o número de atributos candidatos é elevado.
- Pode, contudo, dar preferência a variáveis que isoladamente se mostram boas, mas que não fazem parte do subconjunto ótimo.

Exemplo: $K = 5$ variáveis candidatas

$S(i) = \{1,3\}$ → subconjunto de variáveis já escolhidas até a iteração i .

$C(i) = \{2,4,5\}$ → subconjunto de variáveis em análise.

Nesta iteração, são formados os subconjuntos candidatos $\{1,2,3\}$, $\{1,3,4\}$ e $\{1,3,5\}$ por meio da inserção de uma variável de $C(i)$ em $S(i)$. Os três subconjuntos são, então, utilizados no treinamento do modelo e aquele que levar ao menor erro de validação será definido como o conjunto $S(i + 1)$. Por exemplo, $S(i + 1) = \{1,3,4\}$, indicando que a variável 4 era a melhor opção a ser incluída.

- *Backward elimination*: também é uma estratégia gulosa, mas que parte do conjunto completo de variáveis de entrada e progressivamente escolhe uma para ser descartada.
 - Opção mais custosa que *forward selection*, porém com um potencial maior de identificar o subconjunto ótimo de variáveis pelo fato de iniciar a análise com todas as variáveis presentes.

Exemplo: $K = 5$ variáveis candidatas

$S(i) = \{1,2,3,4,5\} \rightarrow$ subconjunto de variáveis em análise.

Cada variável de entrada é retirada do conjunto de forma isolada, i.e., uma por vez, dando origem aos subconjuntos candidatos $\{2,3,4,5\}$, $\{1,3,4,5\}$, $\{1,2,4,5\}$, $\{1,2,3,5\}$ e $\{1,2,3,4\}$. Os cinco subconjuntos são, então, utilizados no treinamento do modelo e aquele que levar ao menor erro de validação será definido como o conjunto $S(i + 1)$. Por exemplo, $S(i + 1) = \{1,3,4,5\}$, indicando que a variável 2 era a melhor opção a ser retirada.

13. Referências bibliográficas

- ALPAYDIN, E. **Introduction to Machine Learning**. MIT Press. 3rd edition. 2014.
- BATTITI, R. First- and Second-Order Methods for Learning: Between Steepest Descent and Newton's Method, *Neural Computation*, vol. 4, pp. 141-166, 1992.
- BISHOP, C. M. **Pattern Recognition and Machine Learning**. Springer, 2006.
- GEMAN, S., BIENENSTOCK, E., DOURSAT, R. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, vol. 4, no. 1, pp. 1-58, 1992.
- GIROSI, F., JONES, M., POGGIO, T. Regularization Theory and Neural Networks Architectures. *Neural Computation*, vol. 7, no. 2, pp. 219-269, 1995.
- GUYON, I., ELISSEEFF, A. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, vol. 3, pp. 1157-1182, 2003.
- HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference and Prediction**. Springer. 2nd edition, 2006.
- HAYKIN, S. **Adaptive Filter Theory**. Pearson, 5th edition, 2013.
- HAYKIN, S. **Neural Networks and Learning Machines**. Prentice Hall, 3rd edition, 2008.
- HUANG, G.-B., ZHU, Q.-Y., SIEW, C.-K. Extreme Learning Machine: Theory and Applications. *Neurocomputing*, vol. 70, pp. 489-501, 2006.

HUSH, D.R. **Learning from Examples: From Theory to Practice**. Tutorial at the IEEE International Conference on Neural Networks, 1997.

LUENBERGER, D.G. **Linear and Nonlinear Programming**. 2nd edition, Addison-Wesley Publishing Company, 1984.

MATHEWS, V. J. Adaptive Polynomial Filters. *IEEE Signal Processing Magazine*, vol. 8, pp. 10-26, 1991.

PRECHELT, L. Automatic Early Stopping Using Cross Validation: Quantifying the Criteria, *Neural Networks*, vol. 11, no. 4, pp. 761-767, 1998.

TIKHONOV, A. N. Solution of Incorrectly Formulated Problems and the Regularization Method. *Soviet Math. Dokl.*, vol. 4, pp. 1035-1038, 1963.

VAN DER SMAGT, P.P. Minimization Methods for Training Feedforward Neural Networks, *Neural Networks*, vol. 7, no. 1, pp. 1-11, 1994.

ZOU, H., HASTIE, T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, vol. 67, pp. 301–320, 2005.