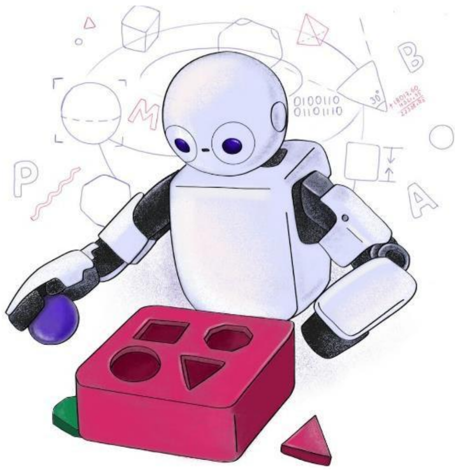
TP555 - Inteligência Artificial e Machine Learning: *Exemplos de implementação do gradiente descendente*

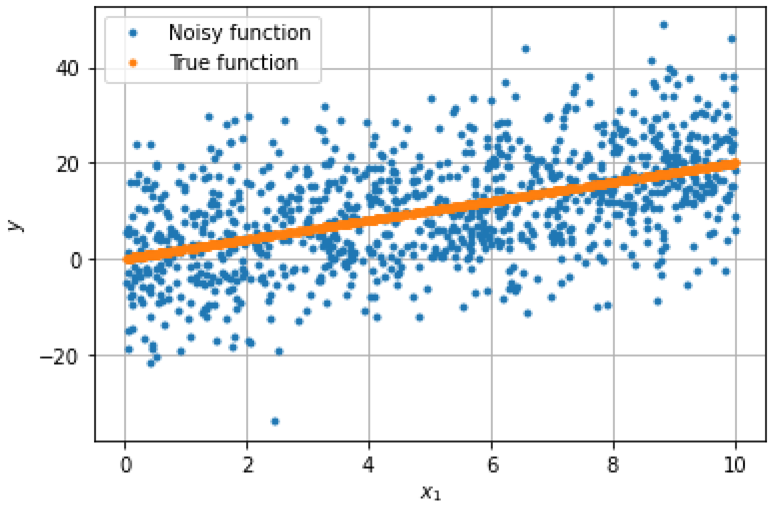


Esse material foi desenvolvido e gentilmente cedido pelo Prof. Dr. Felipe Augusto Pereira de Figueiredo, do Inatel.(felipe.figueiredo@inatel.br)

Inatel

Prof. Dr. Luiz Augusto Melo Pereira
luiz.melo@inatel.br

Exemplo #1



- De posse dos dados ruidosos mostrados na figura acima, encontrar com o ***gradiente descendente*** os pesos de uma função, $h(\boldsymbol{x})$, que aproxime a função objetivo.
- Para facilitar nossa análise, vamos simplificar um pouco e usar uma ***função hipótese*** com apenas um peso, $\hat{a}_1$:

$$\hat{y}(n) = h\big(x_1(n)\big) = \hat{a}_1 x_1(n)$$

- Qual é a regra de atualização para o peso $\hat{a}_1$?
  - **Dicas**:
    - Comecem substituindo $\hat{y}(n)$ em $J_e(\boldsymbol{a}) = \frac{1}{N}\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2 = \frac{1}{N}\sum_{n=0}^{N-1}\big(y(n) - \hat{y}(n)\big)^2$.
    - Lembrem-se que a operação da derivada parcial é distributiva.

Exemplo #1

[Exemplo 1: linear_regression_with_gradient_descent_exemplo1.ipynb]

Para facilitar nossa análise, vamos simplificar um pouco e usar uma ***função hipótese*** com apenas um peso

$$\hat{y}(n) = h\big(x_1(n)\big) = \hat{a}_1 x_1(n)$$

Com ***função de erro*** dada por

$$J_e(\hat{a}_1) = \frac{1}{N}\sum_{n=0}^{N-1}\big(y(n) - \hat{a}_1 x_1(n)\big)^2$$

Gradiente dado por

Operação da derivada parcial é distributiva.

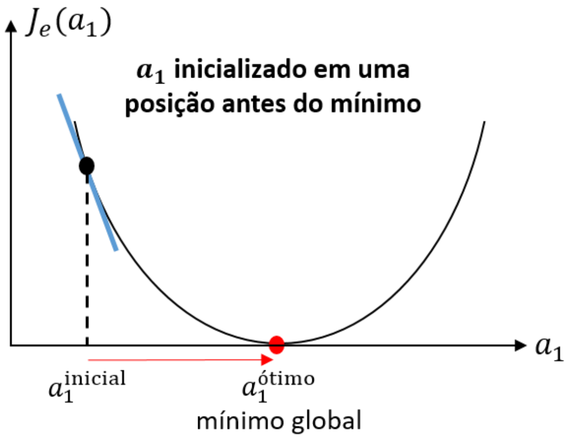
$$\frac{\partial J_e(\hat{a}_1)}{\partial\ \hat{a}_1} = \frac{1}{N}\sum_{n=0}^{N-1}\frac{\partial\big(y(n) - \hat{a}_1 x_1(n)\big)^2}{\partial\ \hat{a}_1}$$

$$= -\frac{2}{N}\sum_{n=0}^{N-1}\big(y(n) - \hat{a}_1 x_1(n)\big)x_1(n)$$

E atualização do peso $\hat{a}_1$ dada por

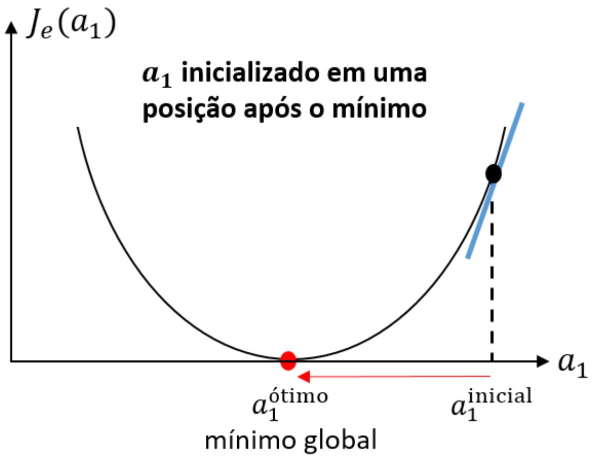
$$\hat{a}_1 = \hat{a}_1 - \alpha\frac{\partial J_e(\hat{a}_1)}{\partial\ \hat{a}_1} \therefore \hat{a}_1 = \hat{a}_1 + \alpha\sum_{n=0}^{N-1}\big(y(n) - \hat{a}_1 x_1(n)\big)x_1(n),$$

onde o termo $\frac{2}{N}$ foi absorvido pelo ***passo de aprendizagem***, α.



gradiente negativo: $a_1 = a_1^{\text{inicial}} + \alpha\nabla J_e(a_1)$

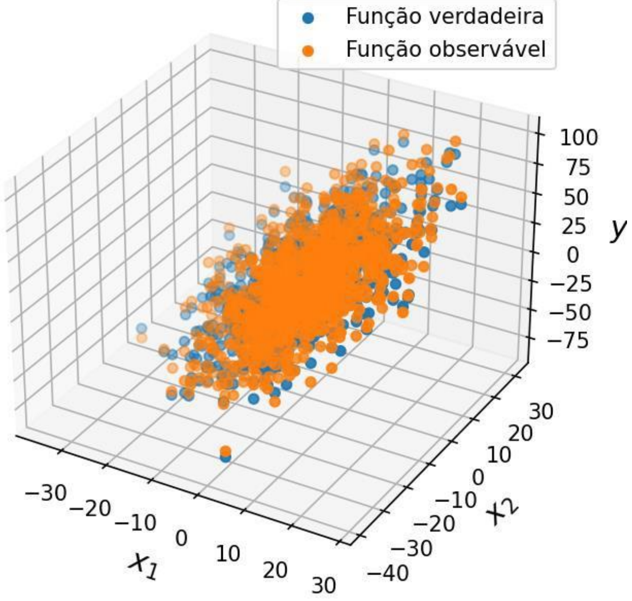
a_1 aumenta e se aproxima do mínimo



gradiente positivo: $a_1 = a_1^{\text{inicial}} - \alpha\nabla J_e(a_1)$

a_1 diminiu e se aproxima do mínimo

Exemplo #2



- De posse dos dados ruidosos mostrados na figura acima, encontrar com o ***gradiente descendente*** os pesos de uma função, $h(\boldsymbol{x})$, que aproxime a função objetivo.
- Neste exemplo, vamos usar uma ***função hipótese*** com 2 pesos, $\hat{a}_1$ e $\hat{a}_2$ onde $\hat{a}_0 = 0$

$$\hat{y}(n) = h\big(\boldsymbol{x}(n)\big) = \hat{a}_1 x_1(n) + \hat{a}_2 x_2(n)$$

- Qual é a regra de atualização para os pesos $\hat{a}_1$ e $\hat{a}_2$?
    - **Dicas**:
        - Comecem substituindo $\hat{y}(n)$ em $J_e(\boldsymbol{a}) = \frac{1}{N}\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2 = \frac{1}{N}\sum_{n=0}^{N-1}\big(y(n) - \hat{y}(n)\big)^2$.
        - Lembrem-se que a operação da derivada parcial é distributiva.

Exemplo #2

[Exemplo 2: linear_regression_with_gradient_descent_exemplo2.ipynb]

Neste exemplo, usaremos uma ***função hipótese*** com 2 pesos, $\hat{a}_1$ e $\hat{a}_2$, sendo $\hat{a}_0 = 0$

$$\hat{y}(n) = h(\boldsymbol{x}(n)) = \hat{a}_1 x_1(n) + \hat{a}_2 x_2(n)$$

A função de erro é dada por

$$J_e(\boldsymbol{a}) = \frac{1}{N}\sum_{n=0}^{N-1}\left[y(n) - \left(\hat{a}_1 x_1(n) + \hat{a}_2 x_2(n)\right)\right]^2.$$

Operação da derivada parcial é distributiva.

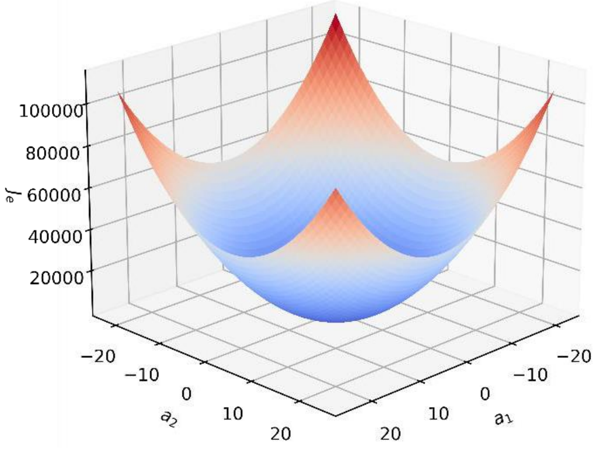
Cada elemento do vetor gradiente é dado por

$$\frac{\partial J_e(\boldsymbol{a})}{\partial\, \hat{a}_k} = \frac{1}{N}\sum_{n=0}^{N-1}\frac{\partial\left[y(n) - \left(\hat{a}_1 x_1(n) + \hat{a}_2 x_2(n)\right)\right]^2}{\partial\, \hat{a}_k} = -\frac{2}{N}\sum_{n=0}^{N-1}\left[y(n) - \left(\hat{a}_1 x_1(n) + \hat{a}_2 x_2(n)\right)\right]^2 x_k(n), \qquad k = 1{,}2$$

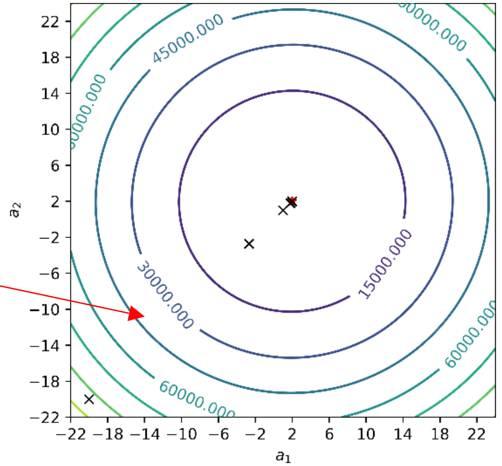
A atualização dos pesos a_k, $k = 1$ e 2 é dada por

$$\hat{a}_k = \hat{a}_k - \alpha\frac{\partial J_e(\boldsymbol{a})}{\partial\, \hat{a}_k} \therefore \hat{a}_k = \hat{a}_k + \alpha\sum_{n=0}^{N-1}\left[y(n) - \left(\hat{a}_1 x_1(n) + \hat{a}_2 x_2(n)\right)\right]x_k(n), \qquad k = 1{,}2$$

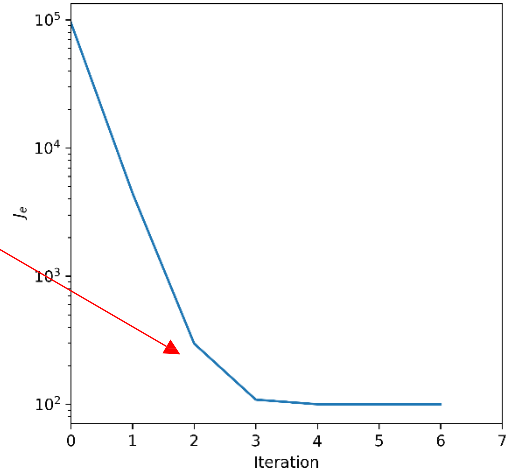
onde o termo $\frac{2}{N}$ foi absorvido pelo ***passo de aprendizagem***, α.



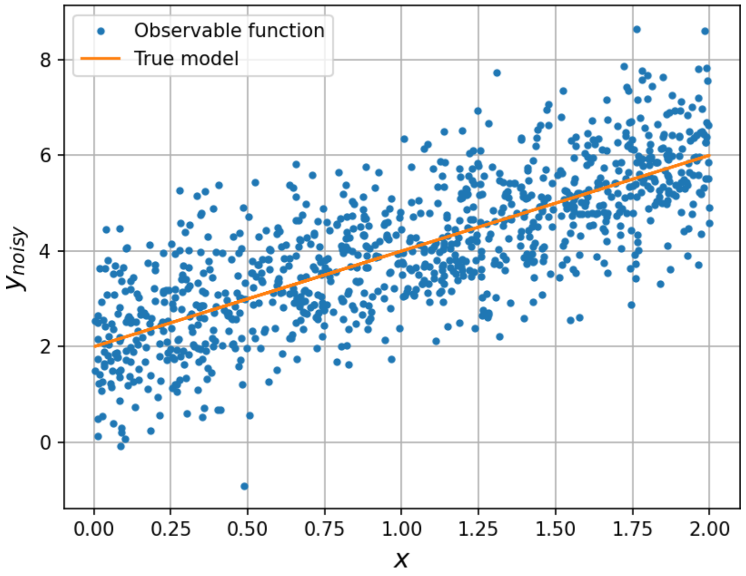
Superfície de contorno com o caminho feito pelo algoritmo até a convergência.



Curva do EQM em função do número de épocas.



Exemplo #3



- De posse dos dados ruidosos mostrados na figura acima, encontrar com o ***gradiente descendente*** os pesos de uma função, $h(\boldsymbol{x})$, que aproxime a função objetivo.
- Neste exemplo, vamos usar uma ***função hipótese*** com 2 pesos, $\hat{a}_0$ e $\hat{a}_1$

$$\hat{y}(n) = h\big(\boldsymbol{x}(n)\big) = \hat{a}_0 + \hat{a}_1 x_1(n)$$

- Qual é a regra de atualização para os pesos $\hat{a}_0$ e $\hat{a}_1$?
  - **Dicas**:
    - Comecem substituindo $\hat{y}(n)$ em $J_e(\boldsymbol{a}) = \frac{1}{N}\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2 = \frac{1}{N}\sum_{n=0}^{N-1}\big(y(n) - \hat{y}(n)\big)^2$.
    - Lembrem-se que a operação da derivada parcial é distributiva.

Exemplo #3

Exemplo 3: linear_regression_with_gradient_descent_exemplo3.ipynb

Função hipótese com 2 pesos, $\hat{a}_0$ e $\hat{a}_1$,

$$\hat{y}(n) = h\big(\boldsymbol{x}(n)\big) = \hat{a}_0 + \hat{a}_1 x_1(n)$$

A ***função de erro*** é dada por

E a atualização dos pesos $\hat{a}_k$, $k = 0$ e 1 é dada por

Operação da derivada parcial é distributiva.

$$\frac{\partial J_e(\boldsymbol{a})}{\partial\, \hat{a}_k} = \frac{1}{N}\sum_{n=0}^{N-1} \frac{\partial \left[y(n) - \left(\hat{a}_0 + \hat{a}_1 x_1(n)\right)\right]^2}{\partial\, \hat{a}_k} = -\frac{2}{N}\sum_{n=0}^{N-1}\left[y(n) - \left(\hat{a}_0 + \hat{a}_1 x_1(n)\right)\right] x_k(n)\,, \qquad k = 0{,}1$$

$$\hat{a}_k = \hat{a}_k - \alpha \frac{\partial J_e(\boldsymbol{a})}{\partial\, \hat{a}_k} \therefore \hat{a}_k = \hat{a}_k + \alpha \sum_{n=0}^{N-1} [y(n) - (\hat{a}_0 + \hat{a}_1 x_1(n))] x_k(n), \qquad k = 0{,}1$$

onde $x_0(n) = 1, \forall n$.

o termo $\frac{2}{N}$ foi absorvido pelo ***passo de aprendizagem***, α

OBS.1: Temos o termo de bias/intercept nesta função hipótese, portanto, não se esqueçam da coluna de '1's (*vetor do atributo de bias*) na implementação do código.

OBS.2: Para executar este exemplo, é necessário instalar a biblioteca ffmpeg com o comando: *conda install ffmpeg*

Exemplo #4: GDE com Scikit-Learn

Exemplo: SGD_with_scikit_learn_lib.ipynb

- A classe ***SGDRegressor***, da biblioteca ***Scikit-Learn***, implementa o ***gradiente descendente estocástico***.
- A classe possui vários ***hiperparâmetros*** que podem ser configurados (tipo de função de erro, esquema de variação do passo de aprendizagem, penalização, etc.).
  - **Hiperparâmetro**: parâmetro que controla o processo de aprendizagem.
- A ***função de erro*** pode ser configurada entre várias opções, mas por padrão, a classe usa o ***erro quadrático médio*.**
- É possível definir o ***esquema de variação do passo de aprendizagem***: constante, redução programada ou adaptativo.
- Por padrão, o esquema é o da escala inversa, "***invscaling***"

$$\alpha = \frac{\alpha_{\text{init}}}{i^{power}},$$

- onde α_{init} é o passo inicial (por padrão = 0.01), i é o número da iteração e $power$ é o expoente da escala inversa (por padrão = 0.25).
- Porém, não conseguimos implementar as versões em batelada e mini-batch.