Instituto Federal de Santa Catarina
Campus Florianópolis

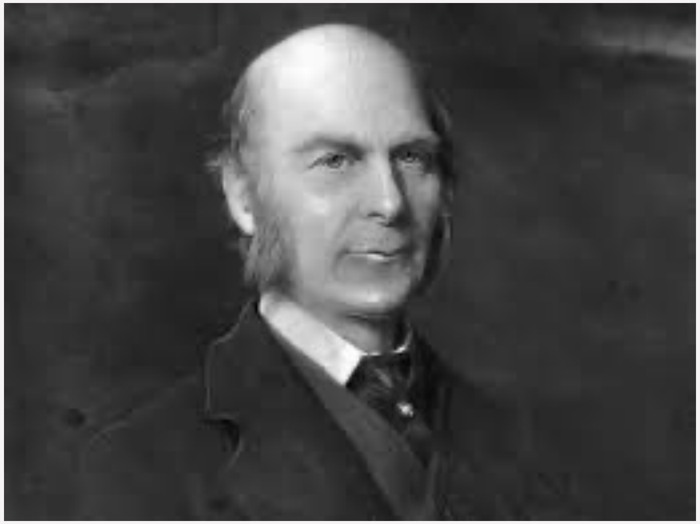
Regressão

Prof. Glauco Cardozo

glauco.cardozo@ifsc.edu.br

Regressão

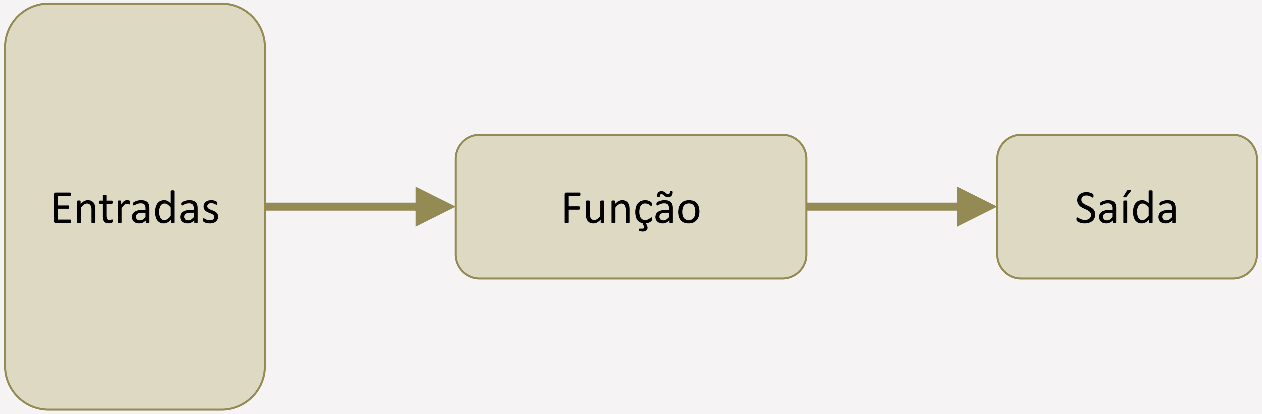
O termo “Regressão” surgiu com Francis Galton em 1885.



Galton, que era antropólogo, matemático e estatístico, estudou a relação das alturas de pais e filhos de uma população, verificando que de modo geral as alturas dos seres humanos tendem a permanecer na média.

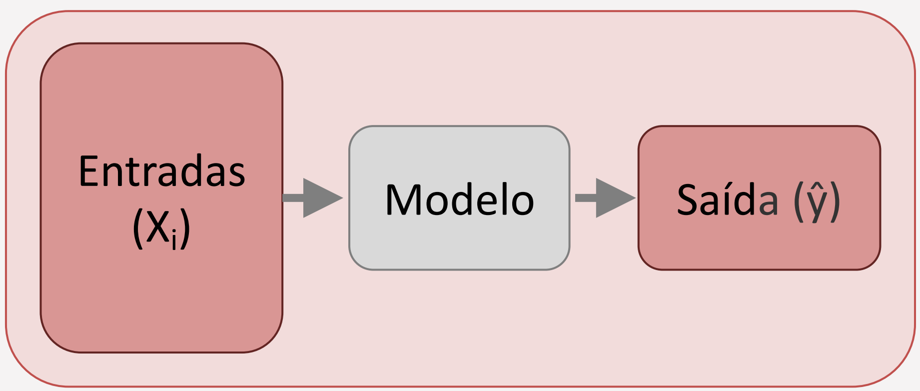
Regressão

Em estatística, regressão é uma técnica que permite quantificar e inferir a relação de uma variável dependente (Saída) com variáveis independentes (Entradas).



Regressão

Tom M. Mitchell define que **aprendizado de máquina** é quando um computador, por meio de uma experiência **E**, melhora sua habilidade em uma tarefa **T**, de acordo com alguma métrica de performance **P.**



Aprendizado Supervisionado compara $\hat{y}$ com y

Regressão

Função Custo - Para avaliar um modelo é preciso definir uma métrica de desempenho, isto é, uma função custo ou função perda L($\hat{y}$, y). Isto é. Que indique o quão “ruim” é a predição $\hat{y}$ quando o valor alvo correto é y.

$$\hat{y} = y + \epsilon$$

Regressão

Função Custo

Erro absoluto:

$$L(\hat{y}, y) = |\ \hat{y} - y|$$

Erro quadrático

$$L(\hat{y}\ , y) = (\hat{y} - y)^2$$

$$L(\hat{y}\ , y) = {}^{1}/_{2}(\hat{y} - y)^2$$

Regressão

Função Custo

Para medir o desempenho de um modelo no conjunto de treinamento, é usual calcular o erro médio (média aritmética) sobre todo o conjunto:

$$J(f) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{L}(\hat{y}, y)$$

Regressão

Função Custo

Para avaliar o poder preditivo de um modelo (generalização), deve-se medir o desempenho sobre um conjunto de teste gerado de forma independente do conjunto de treinamento

$$J(f) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{L}(\hat{y}, y)$$

Regressão

Treinamento

O erro no conjunto de treinamento é usado para determinar os parâmetros do modelo, isto é, para selecionar a hipótese $f \in H$ (dentre um espaço de hipóteses pré-definido) que melhor se ajusta aos dados de treinamento. Treinamento também é chamado de ajuste (fit)

$$\min J(f)$$

Regressão Linear

Regressão linear é uma equação para se estimar a condicional (valor esperado) de uma variável ***y***, dados os valores de algumas outras variáveis ***x.***

$$y_i = a + bx_i + \mathcal{E}_i$$

y_i - Variável explicada (dependente); representa o que o modelo tentará prever.

a - É uma constante, que representa a interceptação da reta com o eixo vertical;

b - Representa a inclinação (coeficiente angular) em relação à variável explicativa;

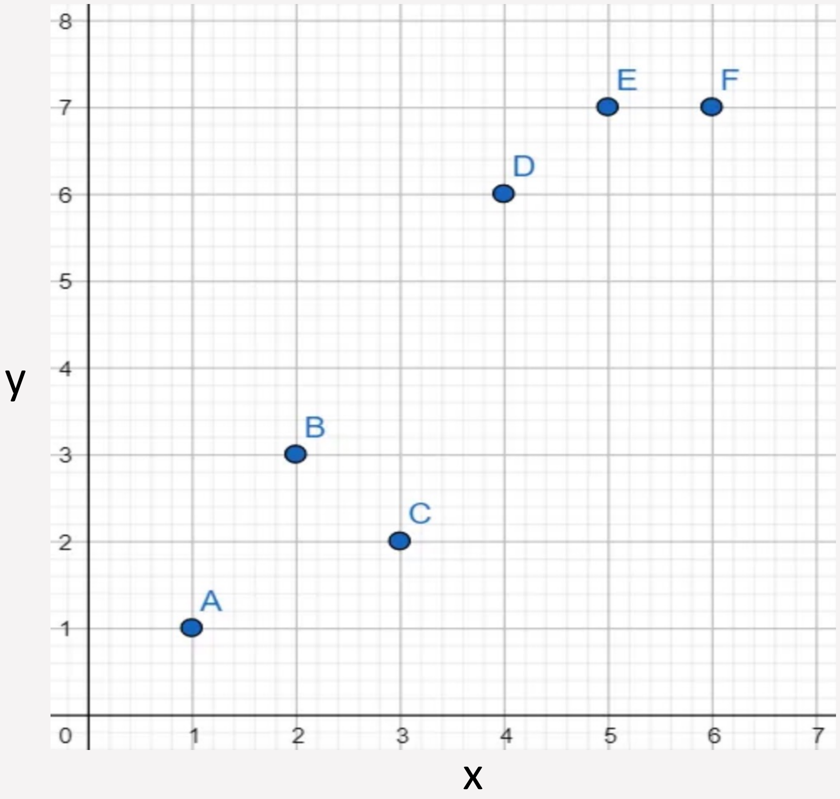
x_i - Variável explicativa (independente);

$\mathcal{E}_i$ - Representa todos os fatores residuais mais os possíveis erros de medição.

Regressão Linear

Regressão linear

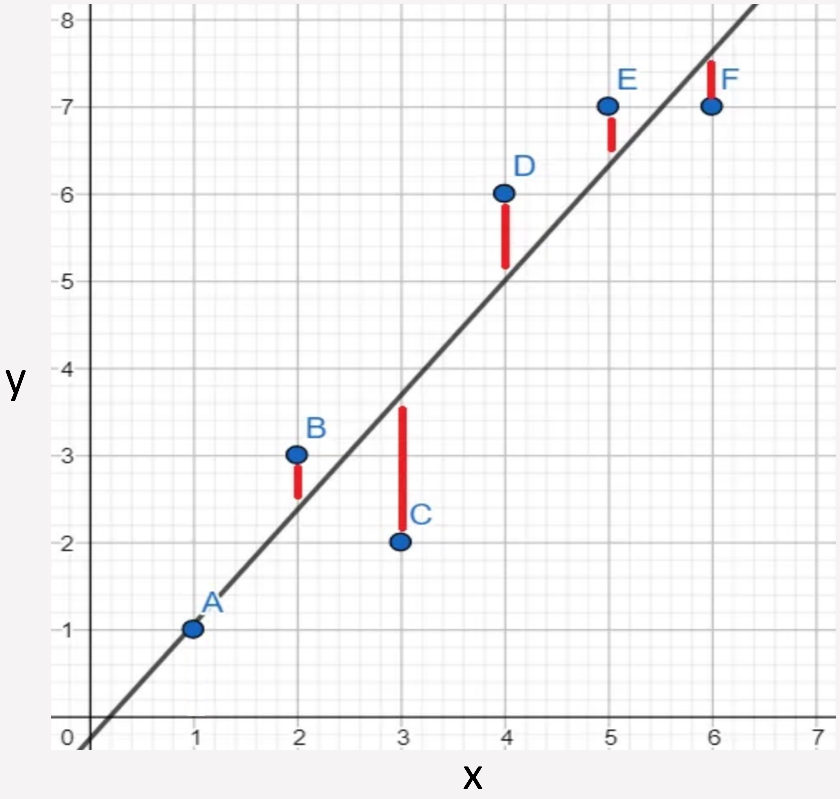
O objetivo da regressão linear é encontrar uma reta que consiga definir bem os dados e minimizar a diferença entre o valor real e a saída calculada pelo modelo



Regressão Linear

Regressão linear

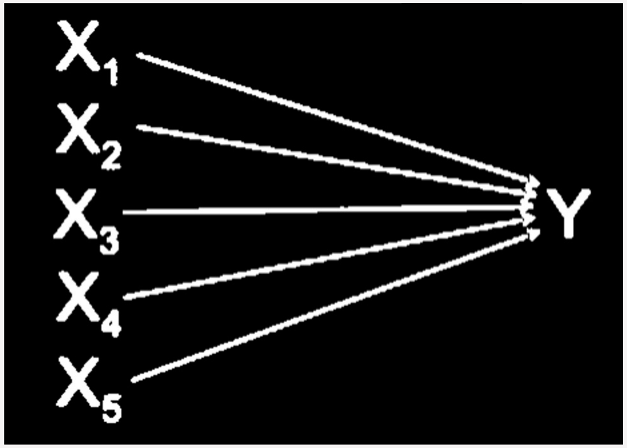
Observamos que praticamente todos os pontos (com exceção do ponto A) não estão coincidindo com a reta. A uma distância sinalizada em pelos traços em vermelho, chamamos de Erro.



Regressão Linear

Regressão linear simples: refere-se quando temos somente uma variável independente (X) para fazermos a predição.

Regressão linear múltipla: refere-se a várias variáveis independentes (X)usadas para fazer a predição.



Regressão Linear

A equação anterior pode ser reescrita em forma de matriz

$$y = bX + \mathcal{E}$$

Onde $\boldsymbol{y}$ é uma matriz de **n** x **1** observações, **X** é uma matriz de tamanho **n** x **(p+1)** (sendo a primeira coluna com valores sempre = 1, representando a constante $\boldsymbol{a}$, e **p** é a quantidade de variáveis explicativas), $\boldsymbol{b}$ é uma matriz de **(1+p)** x **1** variáveis explicativas (sendo que $\boldsymbol{b_0}$ representa a constante $\boldsymbol{a}$) e $\boldsymbol{\mathcal{E}}$ é uma matriz de **n** x **1** de resíduos.

Regressão Linear Múltipla

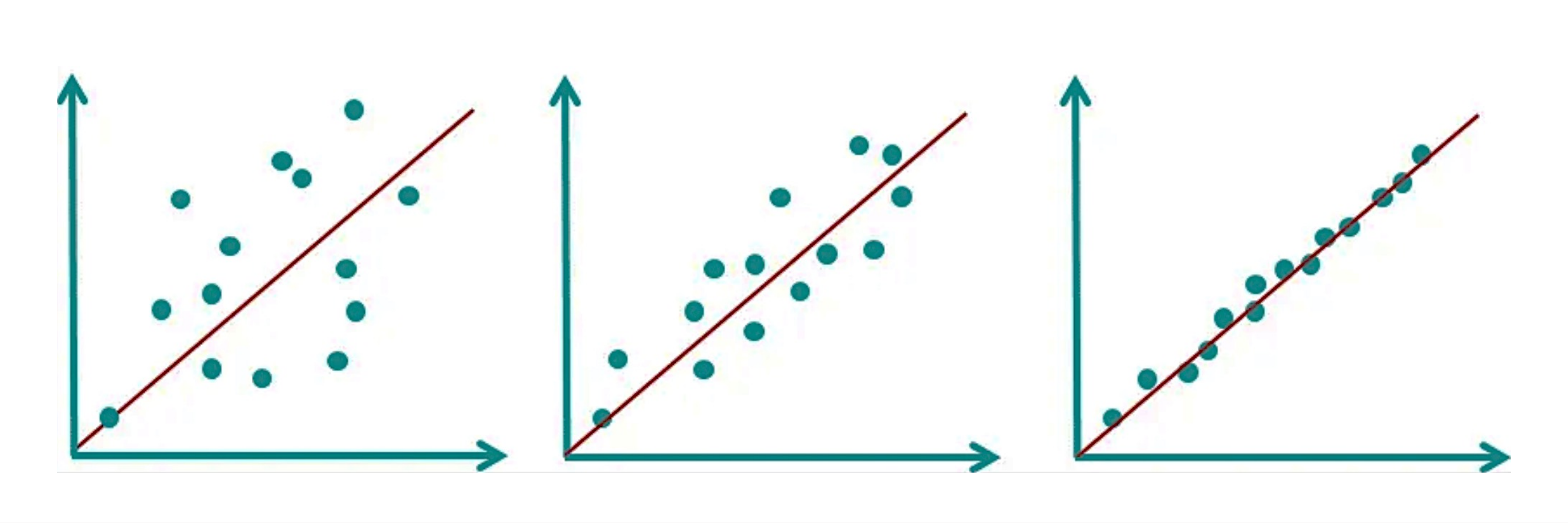
A equação anterior pode ser reescrita em forma de matriz

$$y = bX + \mathcal{E}$$

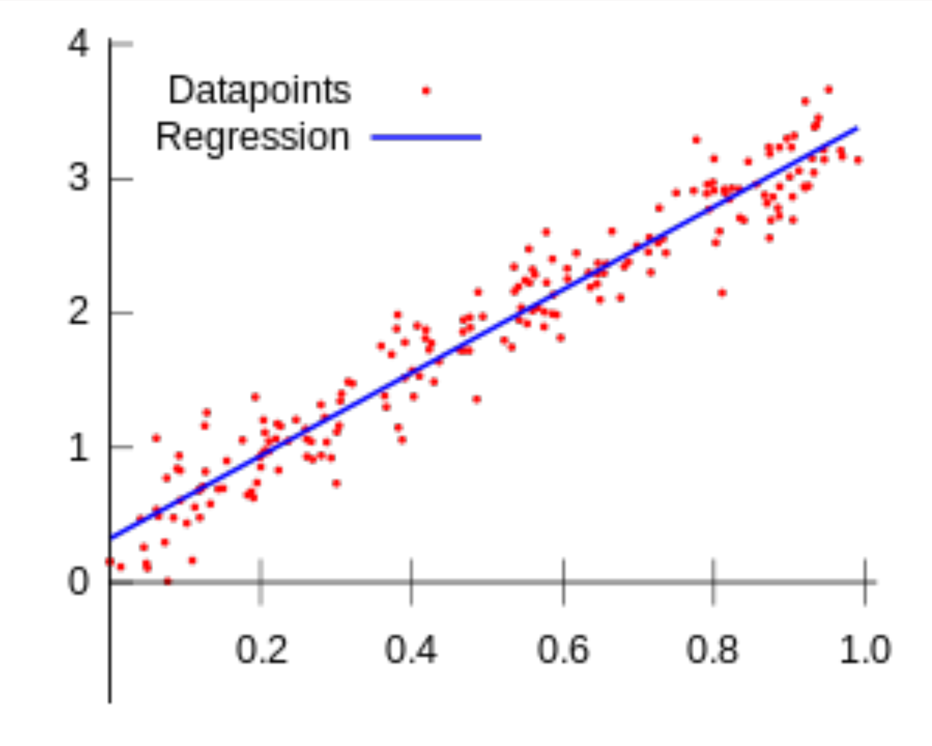
$$y = b_1 X_1 + b_2 X_2 + \cdots + b_n X_n + \mathcal{E}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & X_{11} & X_{12} & \cdots & X_{1p} \\ 1 & X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & X_{n1} & X_{n2} & \cdots & X_{np} \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}, \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Regressão Linear Múltipla



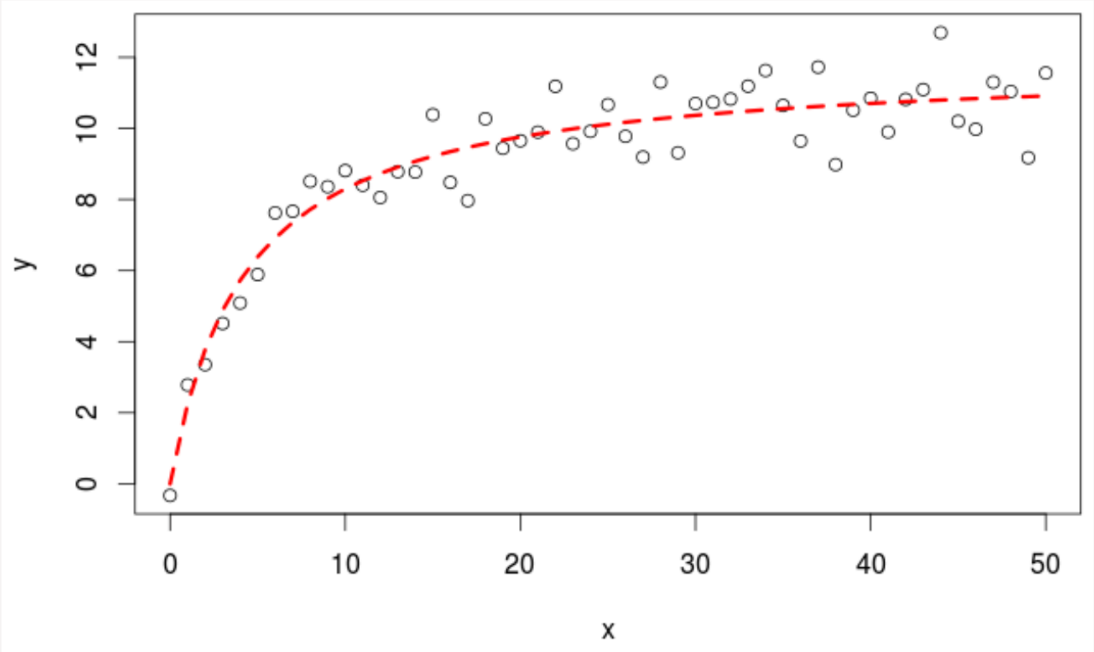
Regressão Linear



$$y = bX + \mathcal{E}$$

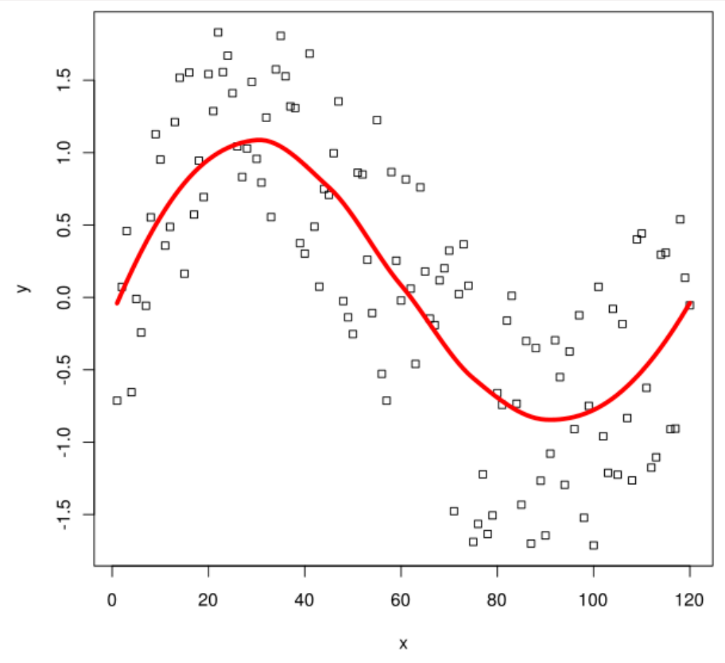
Regressão Não Linear

$$y \; = b_1X + b_2X^2 + \cdots + bnXn + \mathcal{E}$$



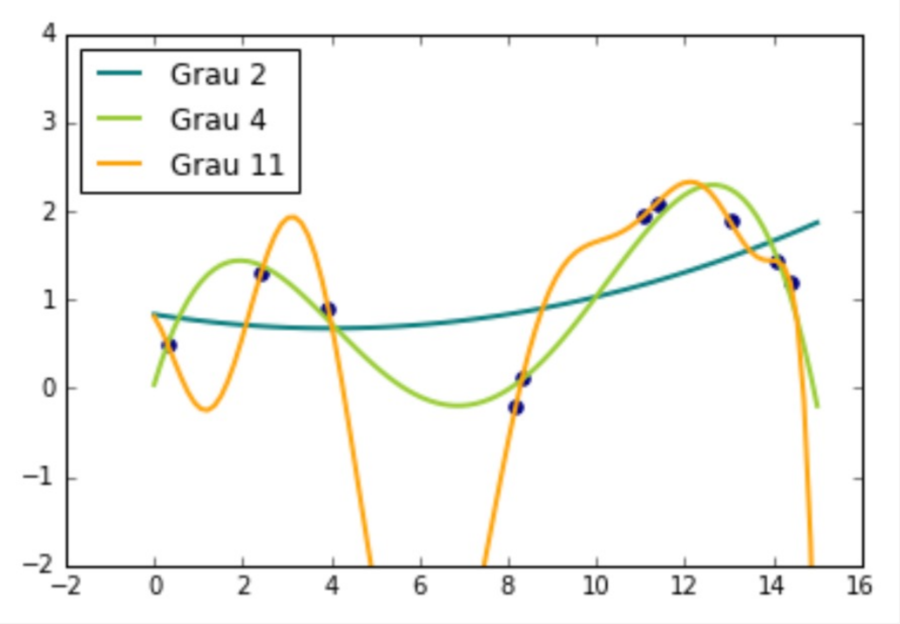
Regressão Não Linear

$$y \ = b_1X + b_2X^2 + \cdots + bnXn + \mathcal{E}$$



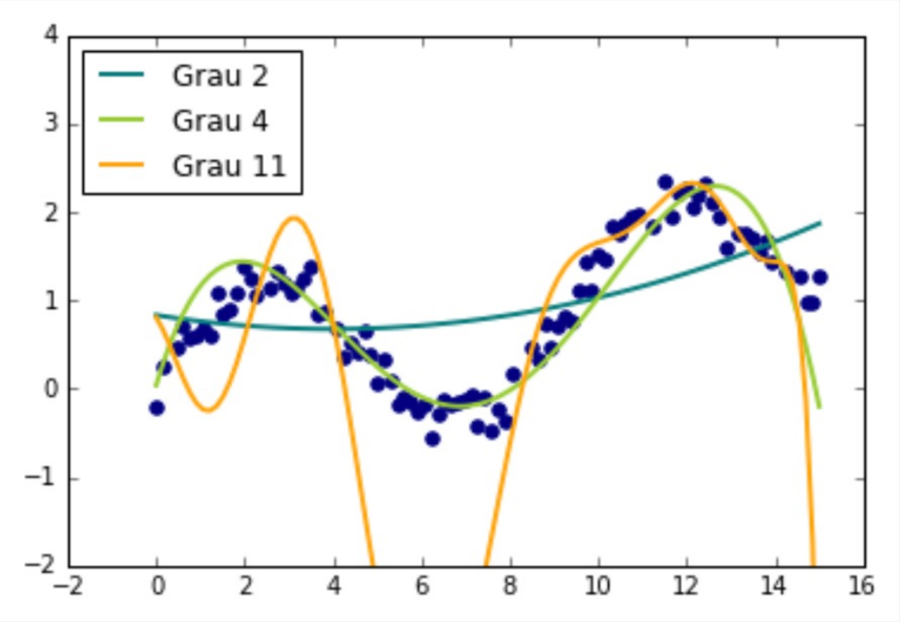
Regressão Não Linear

Teoricamente, **podemos aproximar qualquer função com um polinômio.**



Regressão Não Linear

Teoricamente, **podemos aproximar qualquer função com um polinômio.**



Regressão Linear

Função Hipótese (Modelo)

$$\hat{y} = f(x) = w_0 + w_1 x_1 + \cdots + w_n x_n$$

Parâmetros do modelo

$w_0, w_1, \cdots, w_n$

w_0 também chamado de *bias*

Regressão Linear

Em notação vetorial

$$\hat{y} = f(\mathbf{x}) = w_0 + \begin{bmatrix} w_1 & \cdots & w_n \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = b + \mathbf{w}^T \mathbf{x}$$

Matematicamente, é mais conveniente considerar $b = 0$ e incluir o atributo constante $x_0 = 1$ como parte do vetor X:

$$\hat{y} = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$$

Equação Normal

Regressão Linear

Função Custo

$$J(\mathbf{w}) = \frac{1}{2m} \sum_{i=1}^{m} (\mathbf{w}^T \mathbf{x}^{(i)} - y^{(i)})^2$$

Minimizar

$$\frac{\partial J(\mathbf{w})}{\partial w_j} = 0$$

Regressão Linear

Minimizar

$$\nabla J(\mathbf{w}) = \begin{bmatrix} \frac{\partial J(\mathbf{w})}{\partial w_0} \\ \vdots \\ \frac{\partial J(\mathbf{w})}{\partial w_n} \end{bmatrix} = \frac{1}{m}\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y})$$

onde

$$\mathbf{X} = \begin{bmatrix} (\mathbf{x}^{(1)})^T \\ \vdots \\ (\mathbf{x}^{(m)})^T \end{bmatrix} \quad \text{e} \quad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

Regressão Linear

Limitações Equação Normal

- Para que $X^T X$ seja inversível, é necessário que $m \geq n$
- Mesmo que $X^T X$ seja inversível, invertê-la pode ser computacionalmente custoso.
- Essas limitações podem ser eliminadas usando um método de otimização iterativo conhecido como método do **gradiente descendente.**

Regressão Linear - Métricas

Na regressão buscamos prever um valor numérico, como, por exemplo, as vendas de uma empresa para o próximo mês.

MAE - Mean absolute error (erro absoluto médio) é a média do valor absoluto dos erros:

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Regressão Linear - Métricas

MSE - Mean Squared Error (erro médio quadrático) é a média dos erros quadrados:

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

RMSE - Root Mean Square Error (raiz do erro quadrático médio) é a raiz quadrada da média dos erros quadrados:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

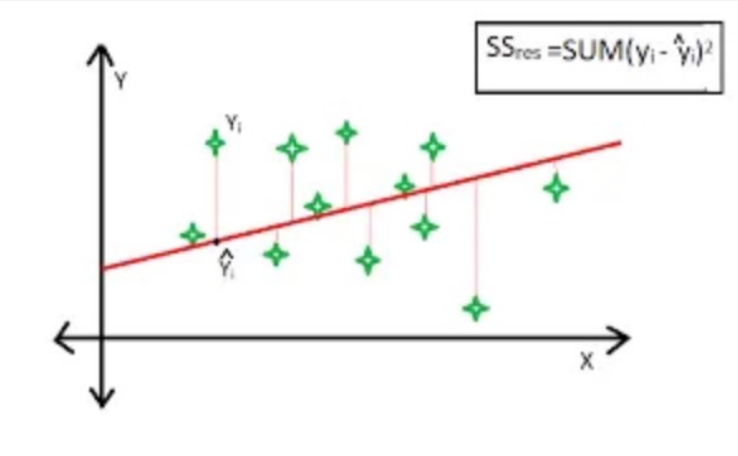
Regressão Linear - Métricas

R^2 – A pontuação R^2 (R^2 Score) é uma medida estatística que nos diz quão bem nosso modelo está fazendo todas as suas previsões em uma escala de zero a um.

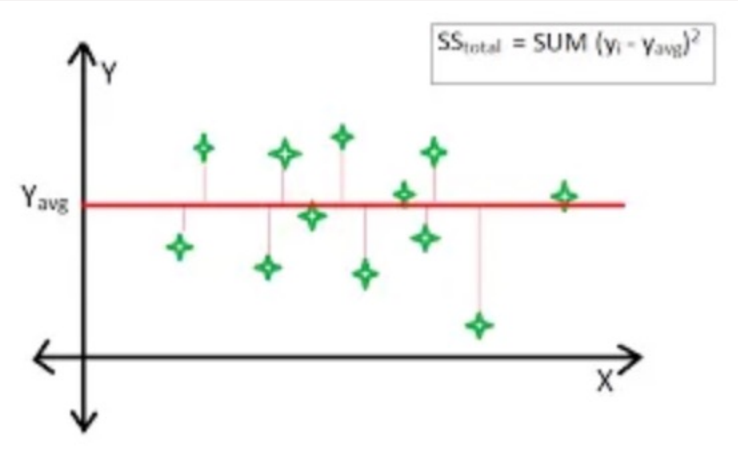
$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

Regressão Linear - Métricas

Se o valor da pontuação R^2 for 1, significa que o modelo é perfeito e se o seu valor for 0, significa que o modelo terá um desempenho ruim



Erro Quadrático Total



Variância total

Regressão Linear – Processo Machine Learning

Carregamento e ajuste dos dados

```
# Load the diabetes dataset
diabetes_X, diabetes_y = datasets.load_diabetes(return_X_y=True)

# Use only one feature
diabetes_X = diabetes_X[:, np.newaxis, 2]
```

Divisão dos Datasets

```
# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

# Split the targets into training/testing sets
diabetes_y_train = diabetes_y[:-20]
diabetes_y_test = diabetes_y[-20:]
```

Regressão Linear – Processo Machine Learning

Criação e treinamento do modelo

```
# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)

# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)
```

Métricas e teste dos modelos

```
# The coefficients
print("Coefficients: \n", regr.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(diabetes_y_test, diabetes_y_pred))
# The coefficient of determination: 1 is perfect prediction
print("Coefficient of determination: %.2f" % r2_score(diabetes_y_test, diabetes_y_pred))
```