Machine Learning

Regressão Linear

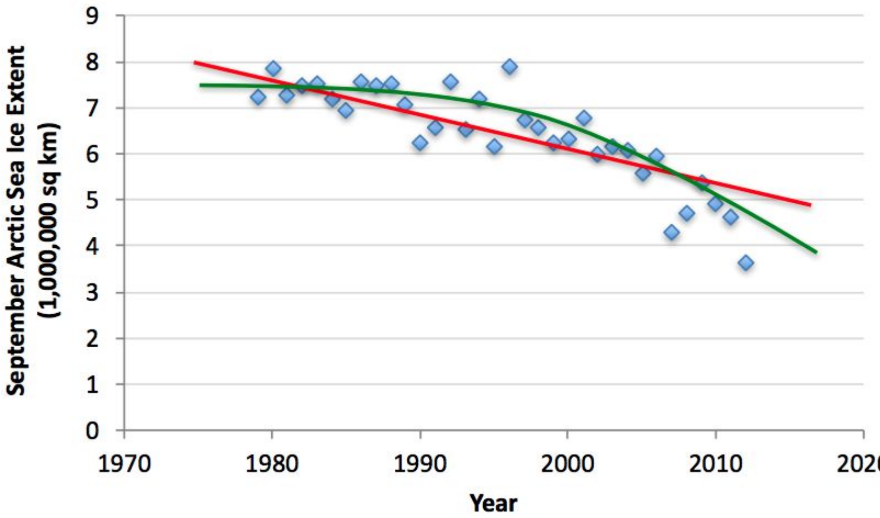
JP Magalhaes
jp@cesar.school

Aprendizado Supervisionado

- O objetivo é aprender uma **função de mapeamento das entradas *X* para as saídas *y***, dado um conjunto anotado de pares entrada-saída
- Este conjunto é conhecido como **conjunto de treinamento**
  - A entrada ***X*** é um vetor D-dimensional de características
    - ex: altura, peso e sexo de uma pessoa
  - A saída ***y*** pode ser:
    - Uma informação categórica → Classificação
      Fraude - Não-Fraude, quando avaliando crédito bancário
    - Um valor real → **Regressão**
      Qual o limite de empréstimo quando Não-Fraude

Regressão

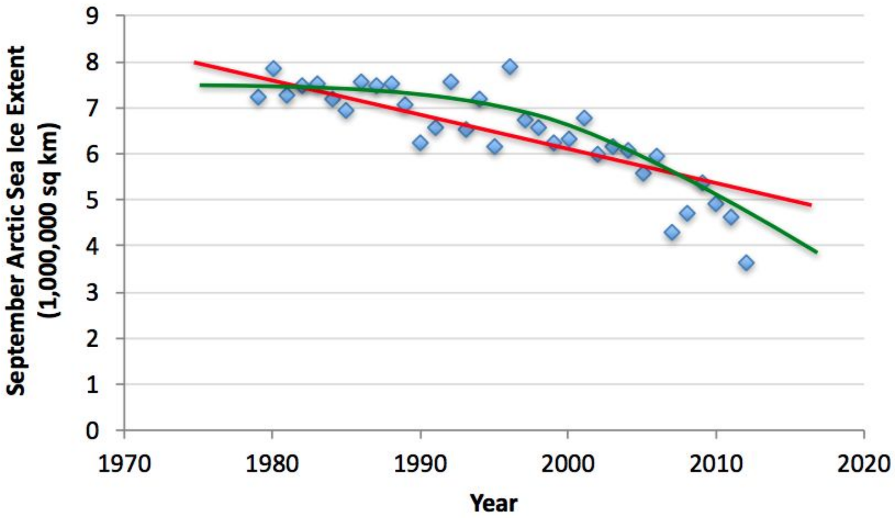
- Dados $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$
- Aprende uma função $f(x)$ capaz de predizer y dado X
  - ***y* é um valor real → regressão**



Regressão

- Dados $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$
- Aprende uma função $f(x)$ capaz de predizer y dado X
  - ***y* é um valor real → regressão**

O objetivo é determinar **como o valor de uma variável dependente *y* muda quando as variáveis independentes *X* são alteradas.**



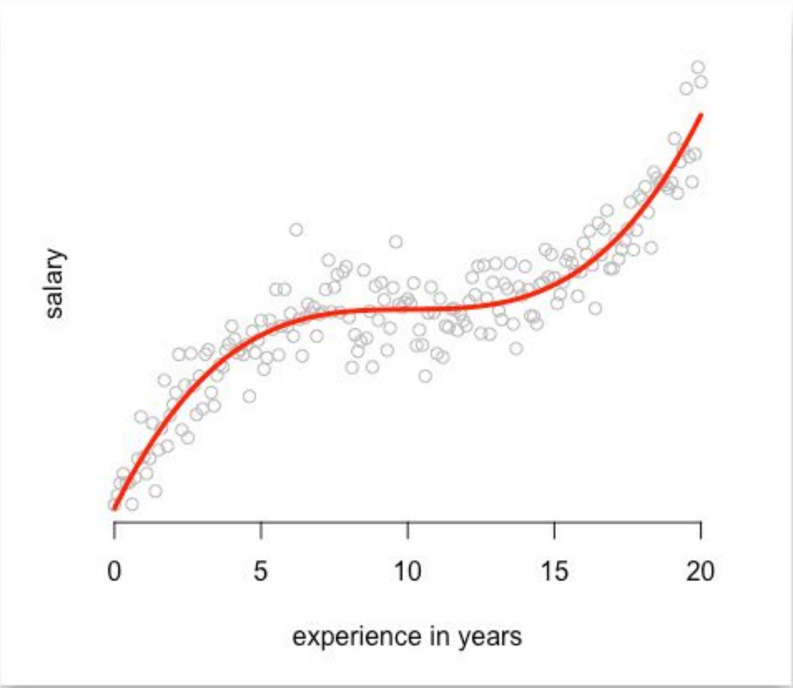
Regressão

- Exemplos:
  - Qual o valor futuro de uma ação?
  - Quanto um usuário está disposto a gastar **($)** em seu site?
  - Qual a localização (em graus) do volante de um carro autônomo baseado em sensores e mapas?
- Pode ser feita de inúmeras formas:
  - Regressão linear simples e polinomial
  - k-NN, Árvores, SVM
  - Entre muitas outras!

Regressão Linear

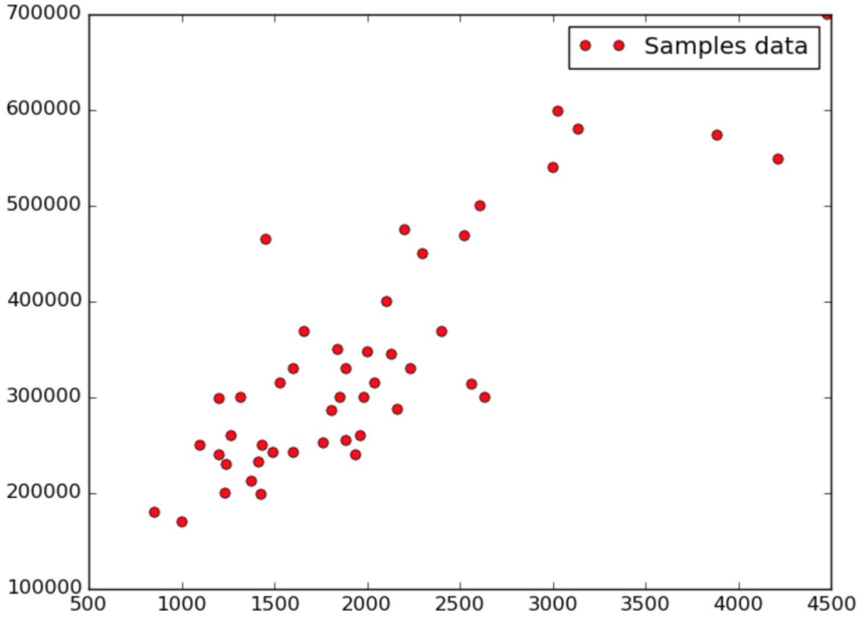
Regressão Linear

- **Regressão Linear** modela a relação de y em relação a X de forma linear
- Pode ser **simples**, quando envolve apenas duas variáveis, ou **múltipla** e ainda **polinomial**
- Os valores y são quantitativos e normalmente contínuos

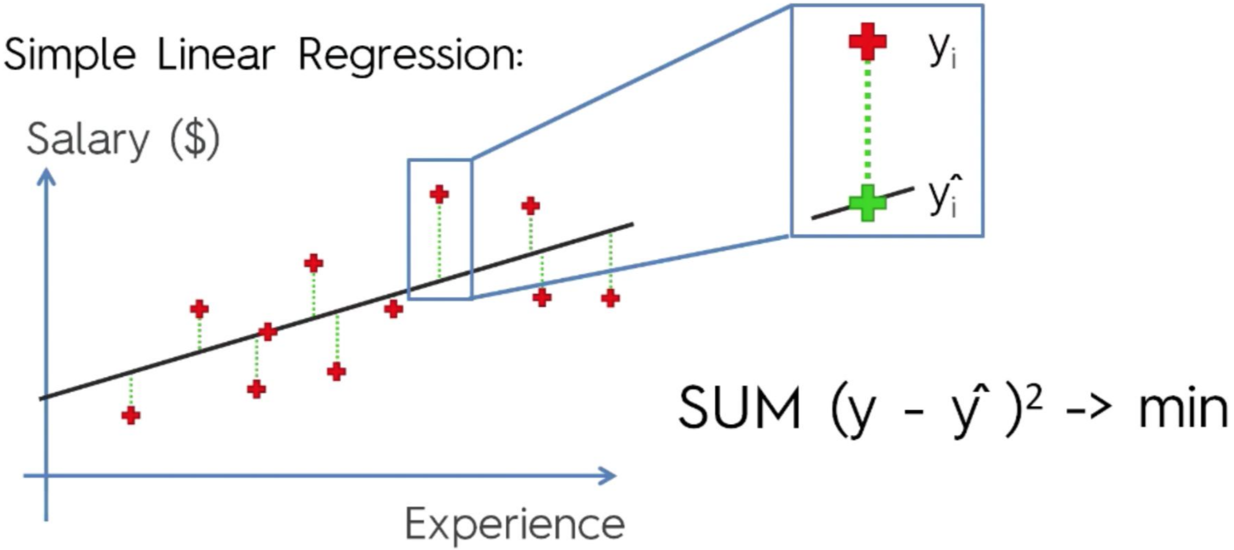


Regressão Linear

- Um dos métodos mais antigos de **aprendizado supervisionado,** fortemente utilizado na estatística
- Dados reais **nunca** são perfeitamente lineares
  - Partimos do princípio que haverá um erro para então minimizá-lo

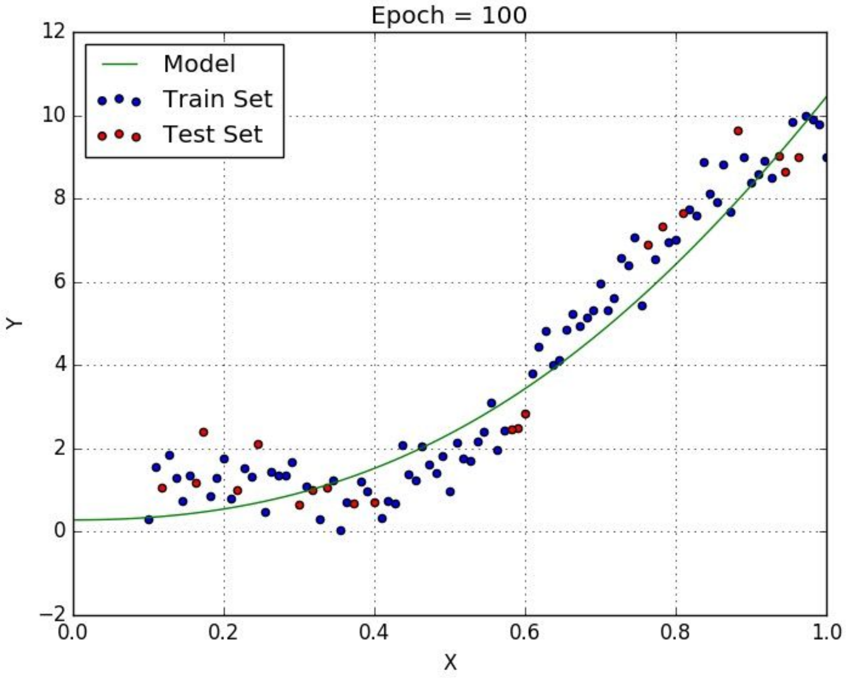


Regressão Linear



O objetivo da regressão linear é buscar a equação de uma linha que **minimize a soma dos erros** entre o valores observados de Y e os valores previstos.

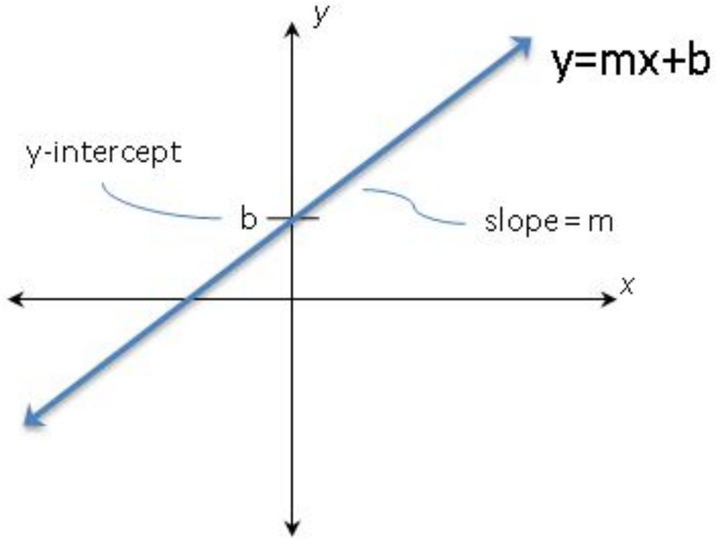
Regressão Linear



O objetivo da regressão linear é buscar a equação de uma linha que **minimize a soma dos erros** entre o valores observados de Y e os valores previstos.

Regressão Linear Simples

Quando temos apenas uma variável e a saída se comporta como uma reta, podemos considerar:

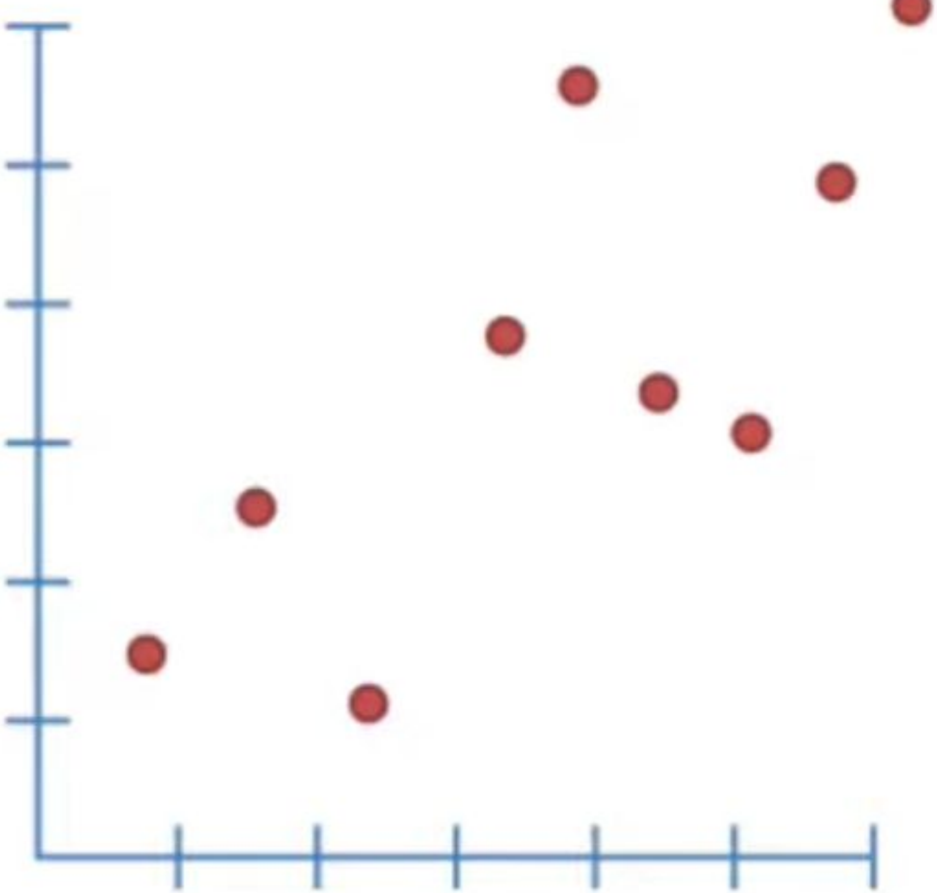


$$Y = aX + b + \varepsilon$$

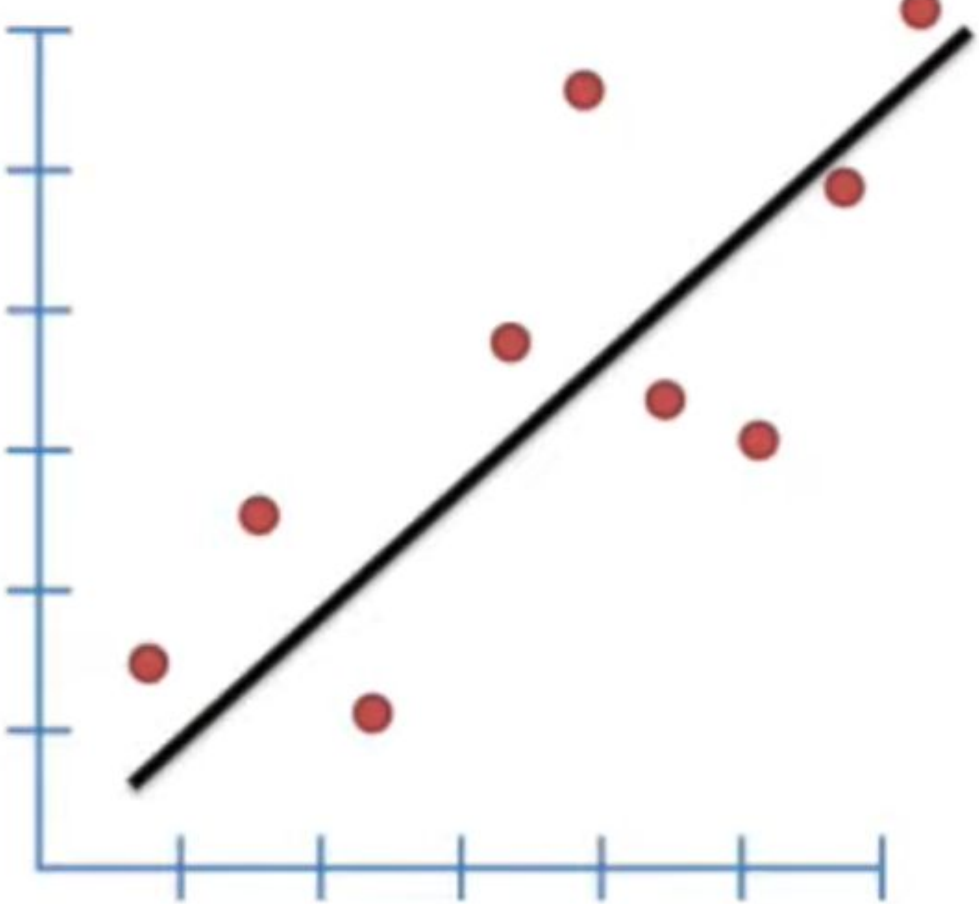
- **a** e **b** são duas constantes que representam a inclinação e o ponto de intersecção, respectivamente
- Ɛ é o termo para o erro

- Conseguindo estimar **a** e **b**, conseguiremos prever os valores futuros de Y
  - Como estimar **a** e **b**?

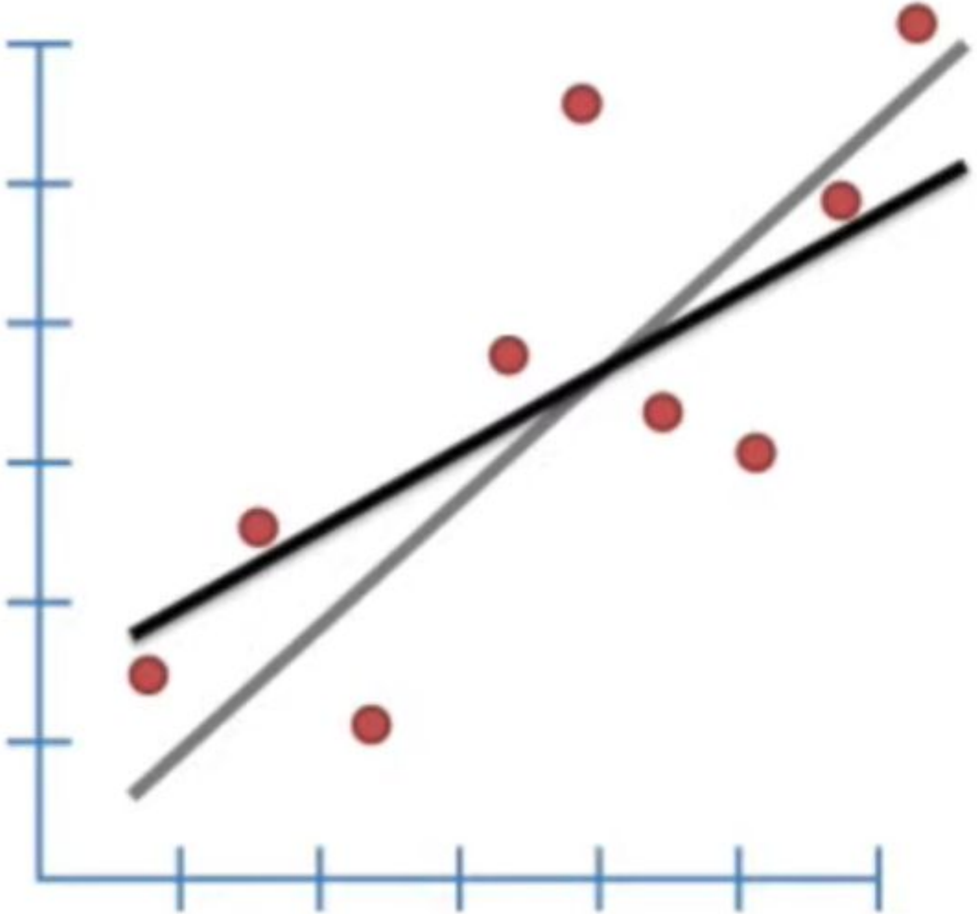
Regressão Linear



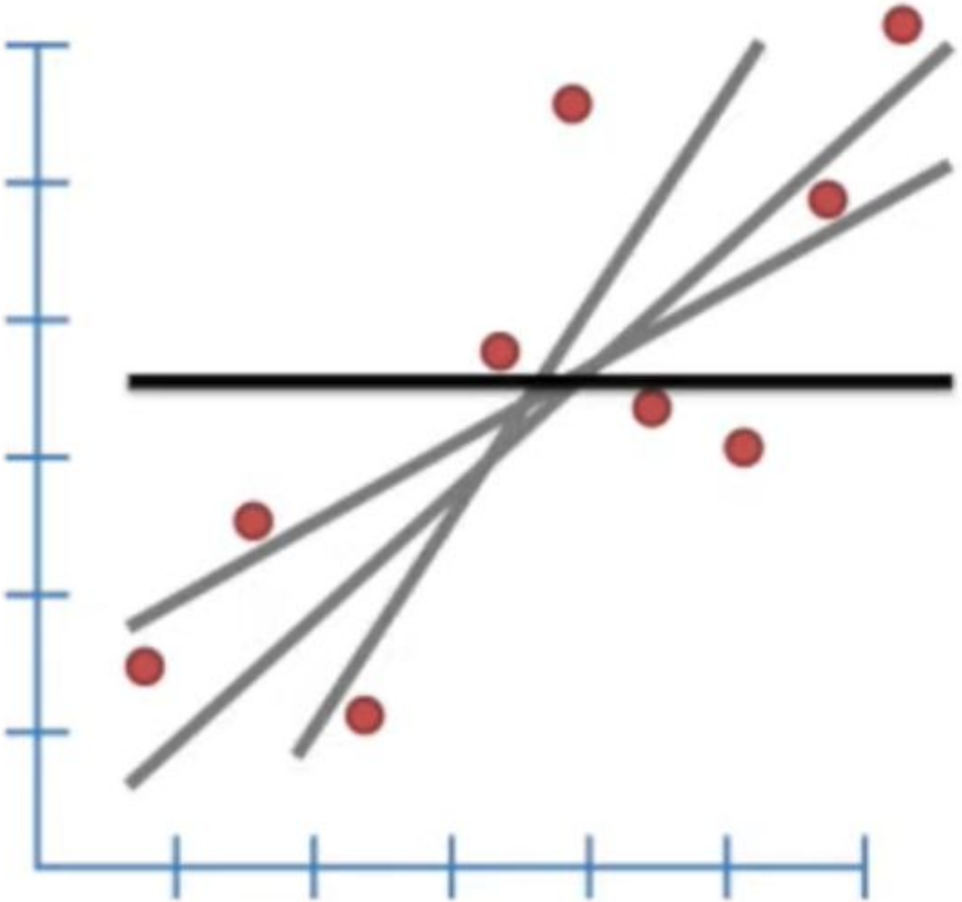
Regressão Linear



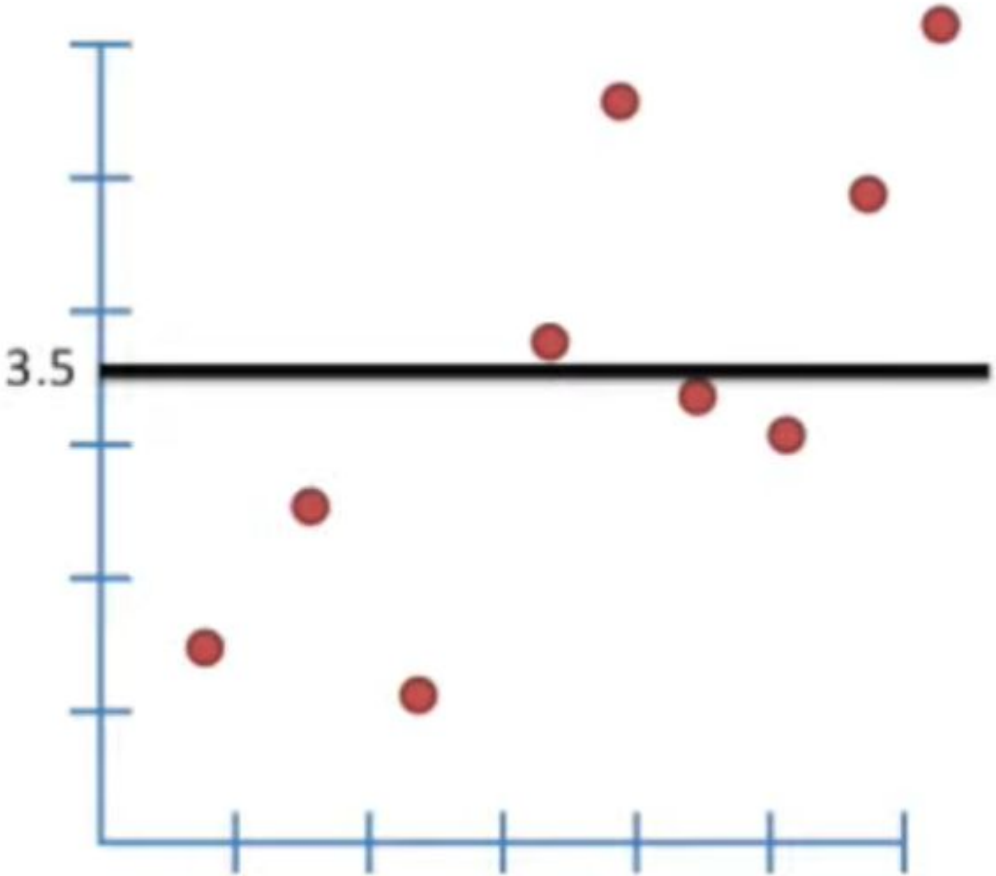
Regressão Linear



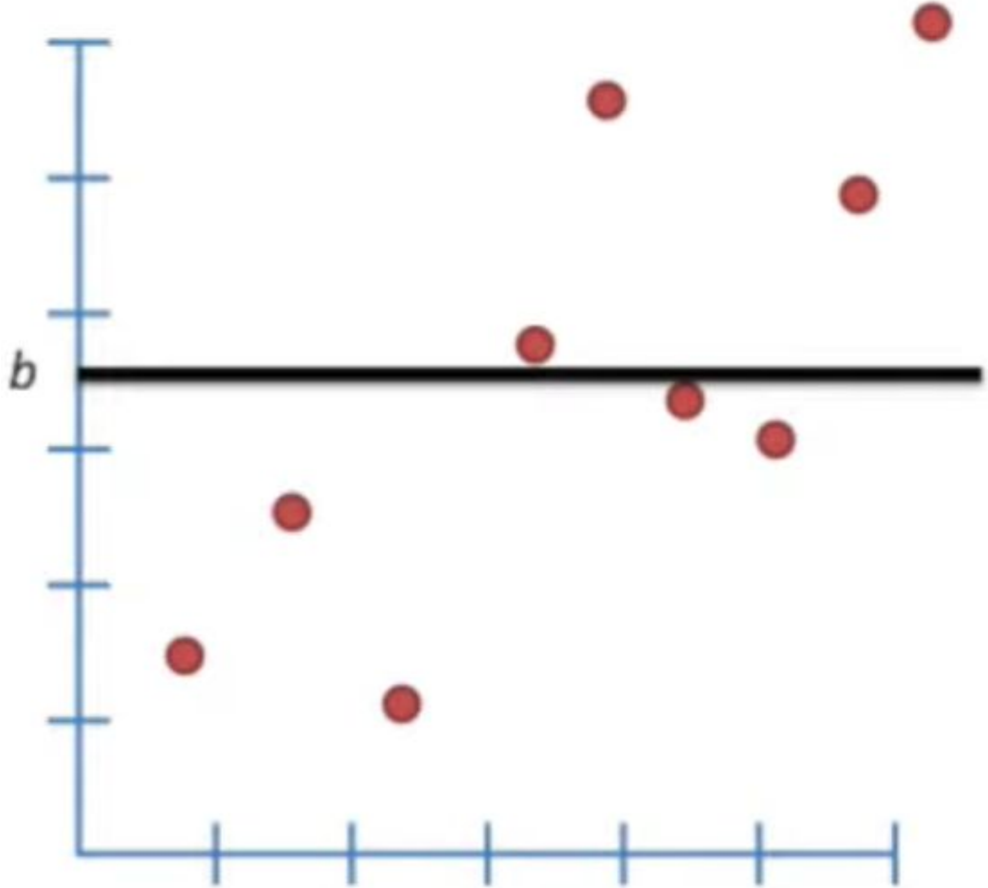
Regressão Linear



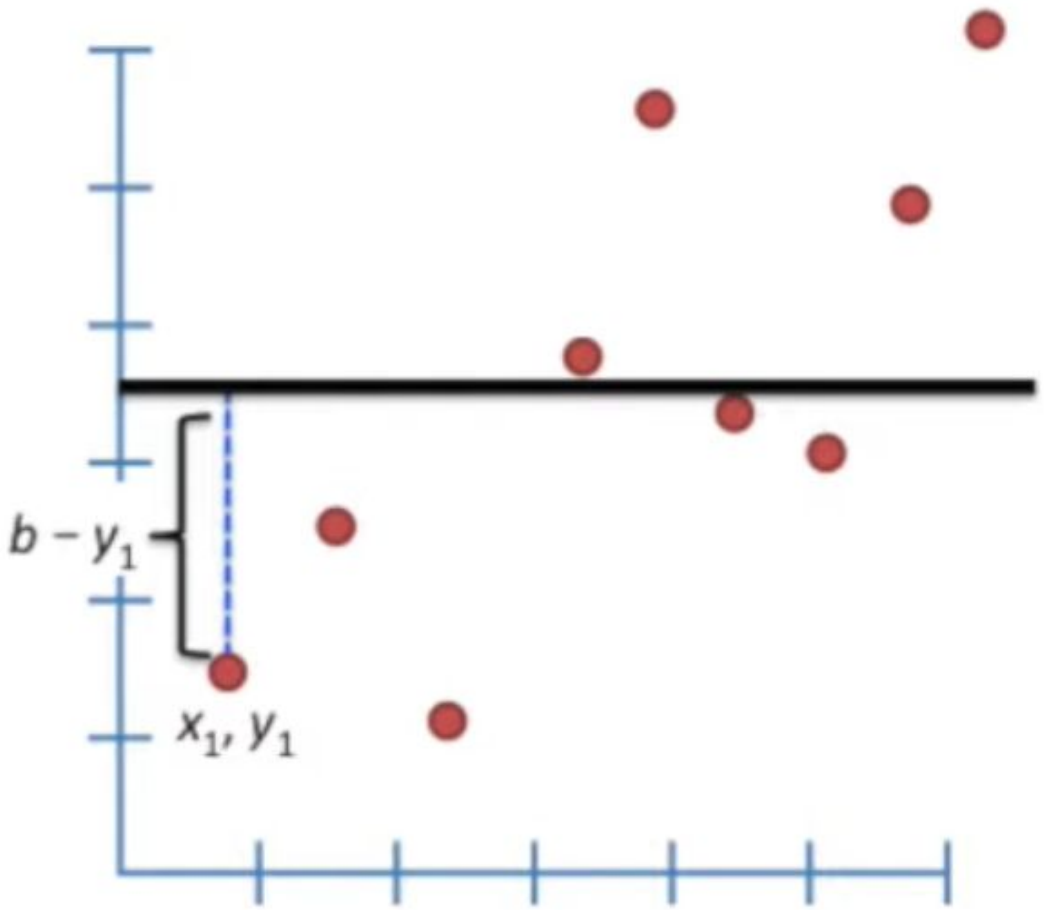
Regressão Linear



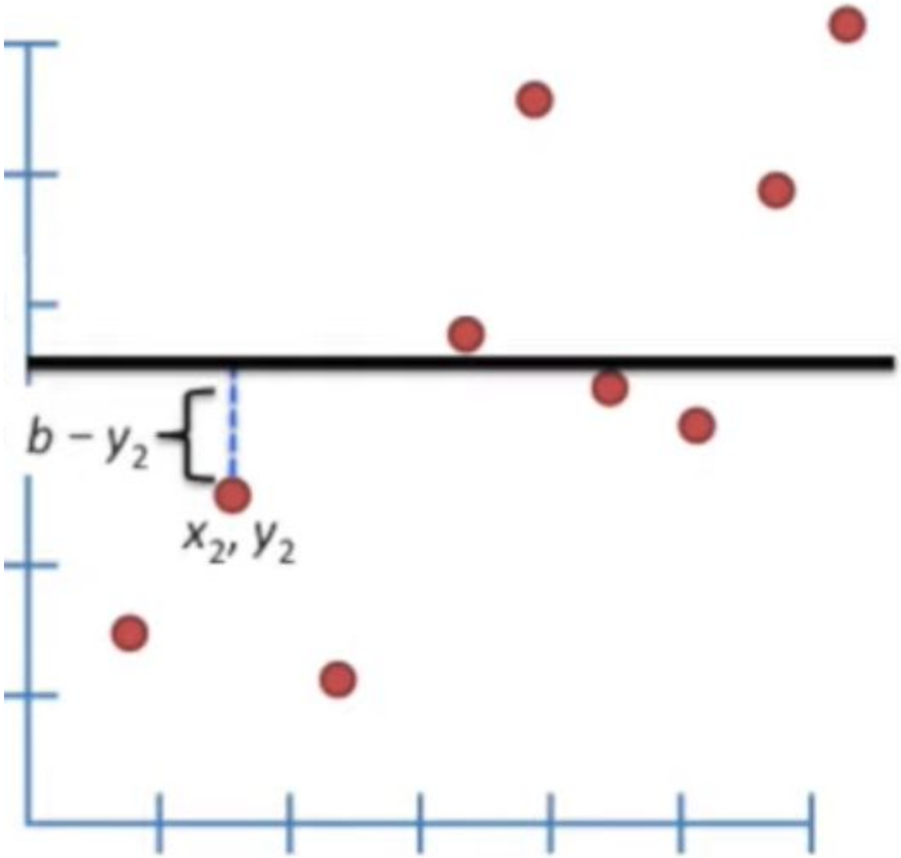
Regressão Linear



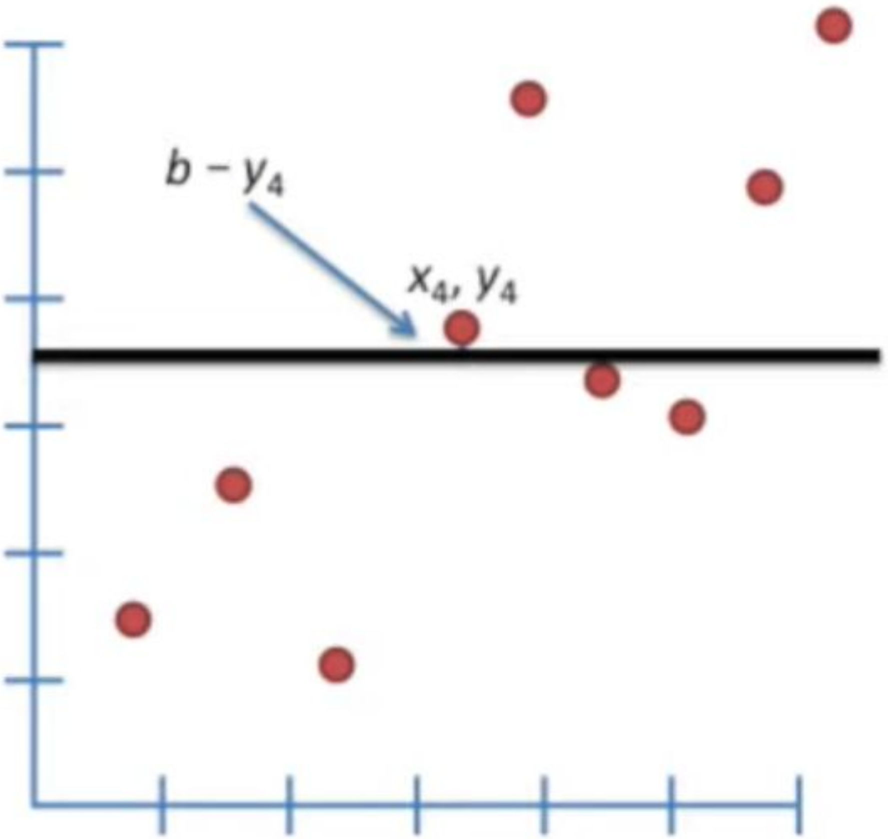
Regressão Linear



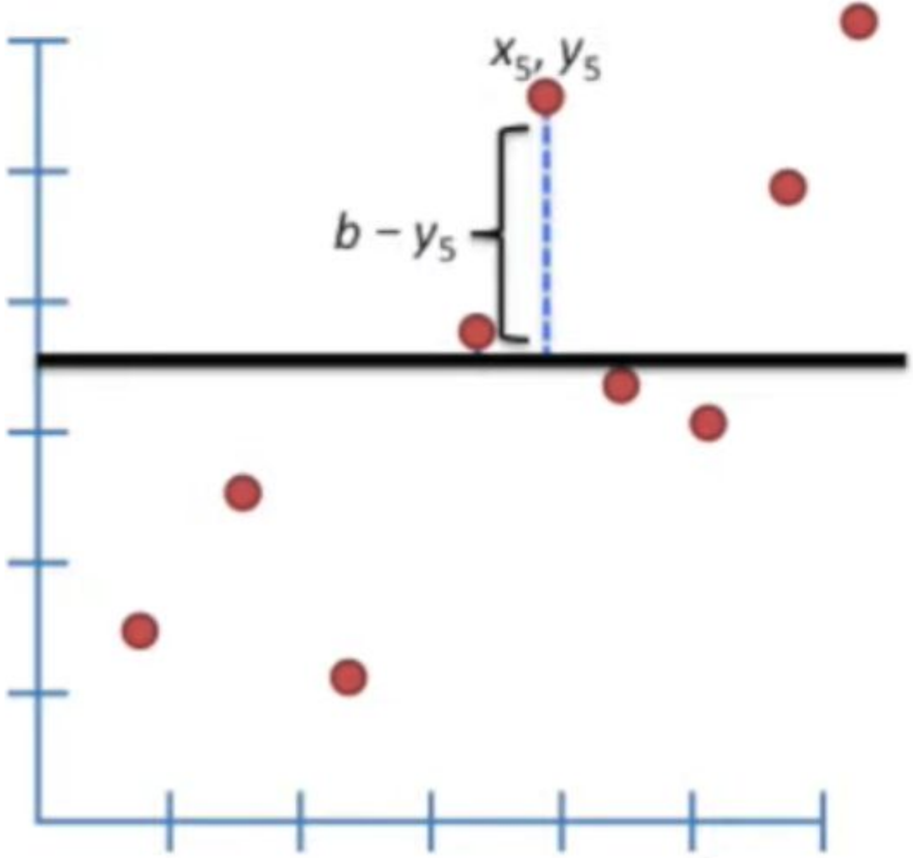
Regressão Linear



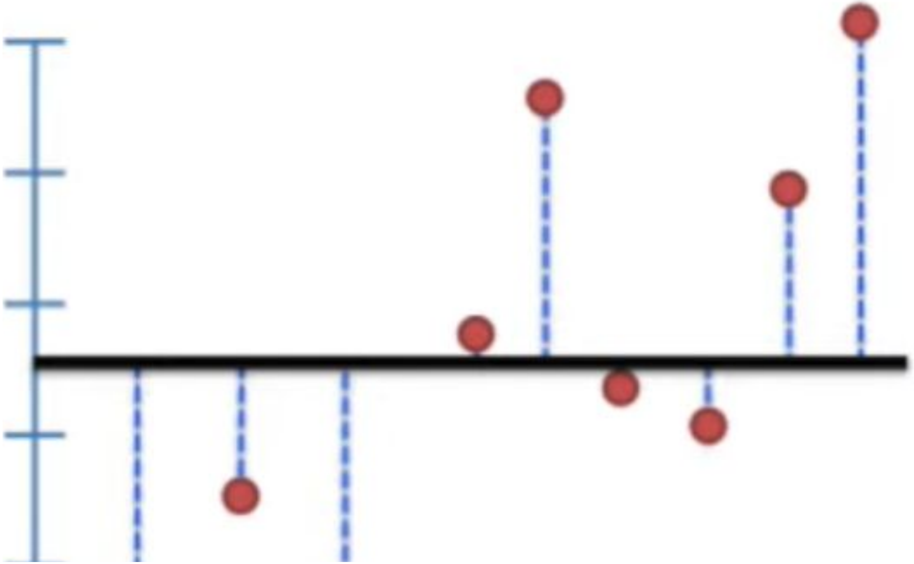
Regressão Linear



Regressão Linear

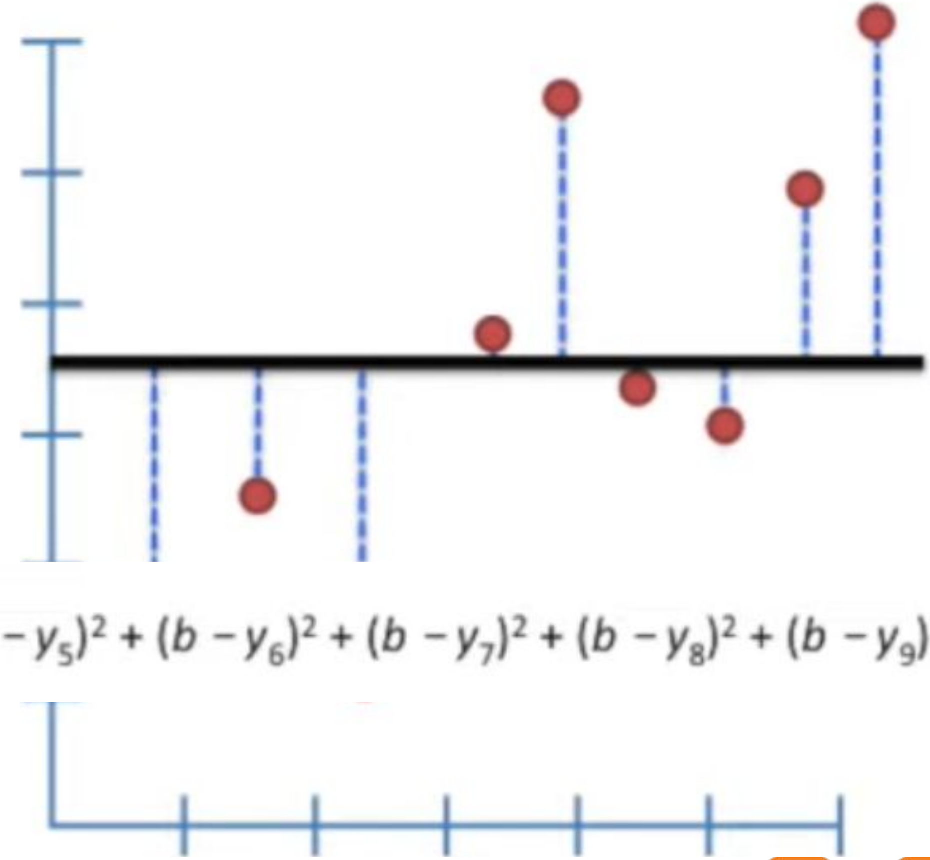


Regressão Linear



$$(b-y_1)^2+(b-y_2)^2+(b-y_3)^2+(b-y_4)^2+(b-y_5)^2+(b-y_6)^2+(b-y_7)^2+(b-y_8)^2+(b-y_9)^2$$

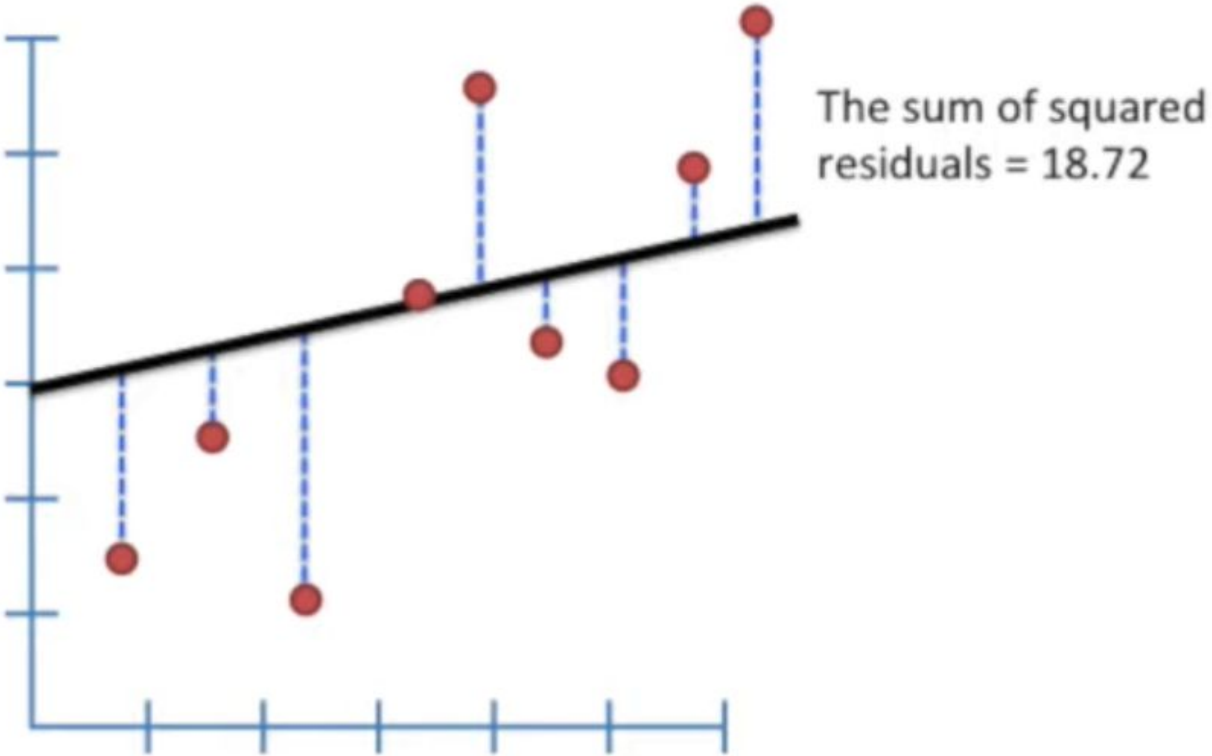
Regressão Linear



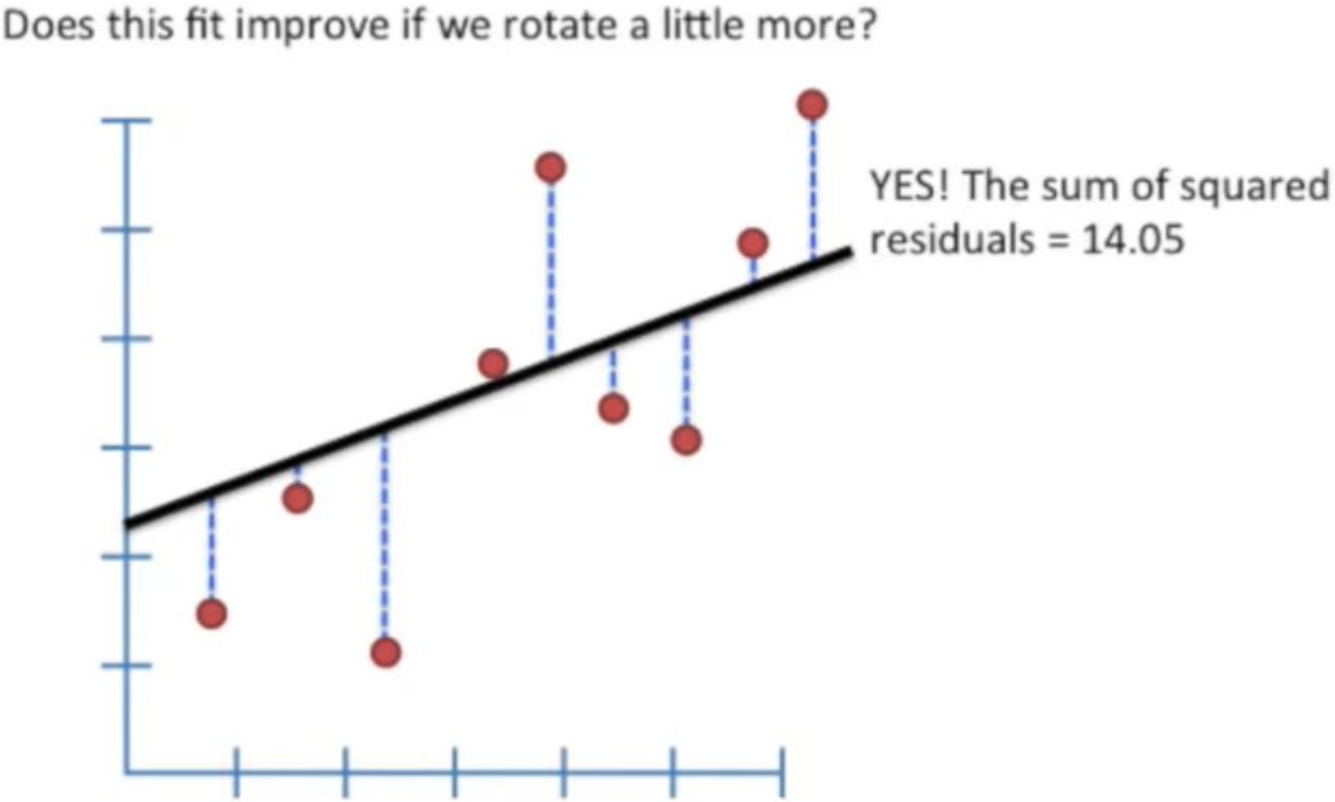
$$(b-y_1)^2+(b-y_2)^2+(b-y_3)^2+(b-y_4)^2+(b-y_5)^2+(b-y_6)^2+(b-y_7)^2+(b-y_8)^2+(b-y_9)^2$$

$$= 24.62$$

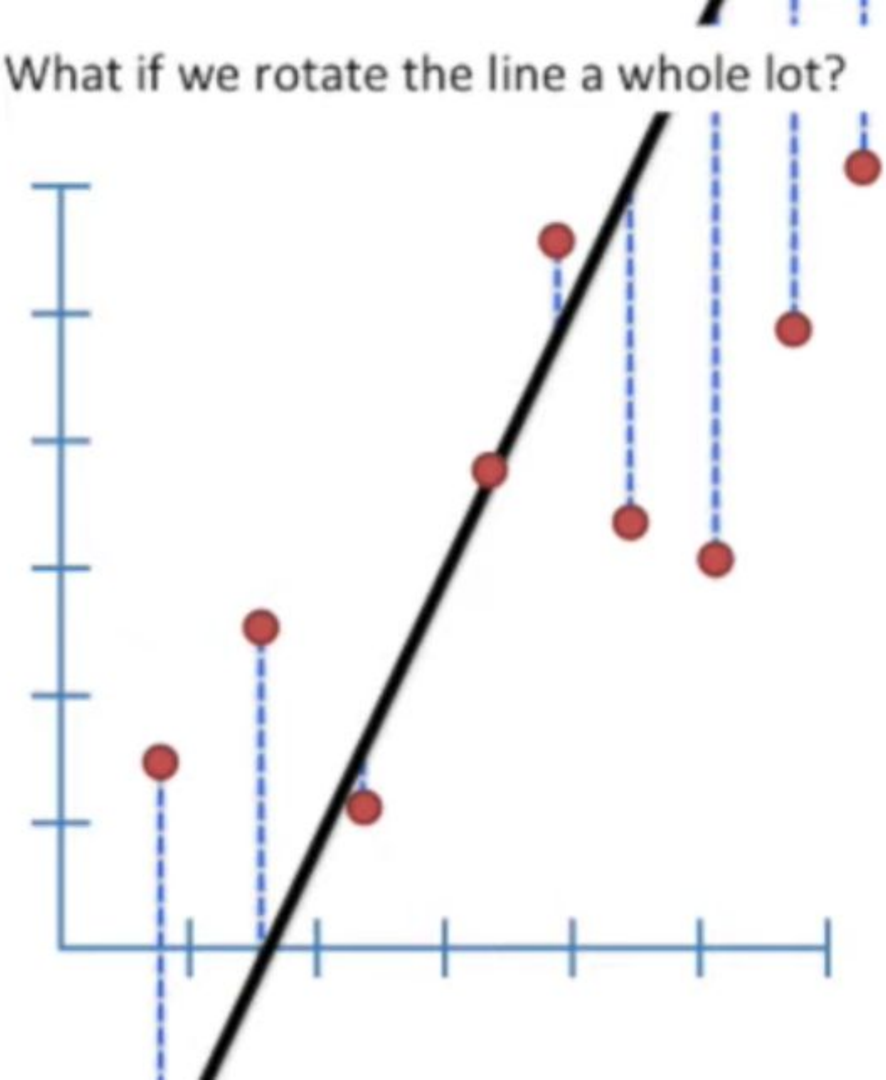
Regressão Linear



Regressão Linear

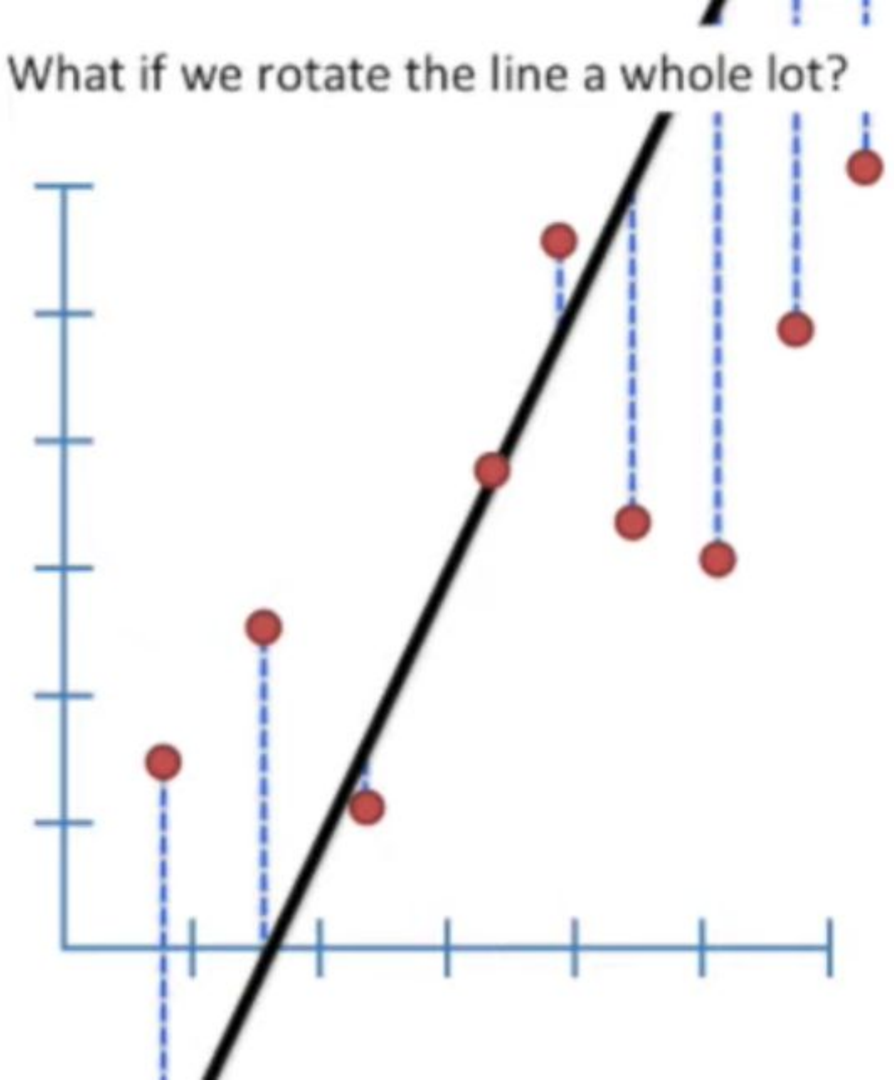


Regressão Linear



Regressão Linear

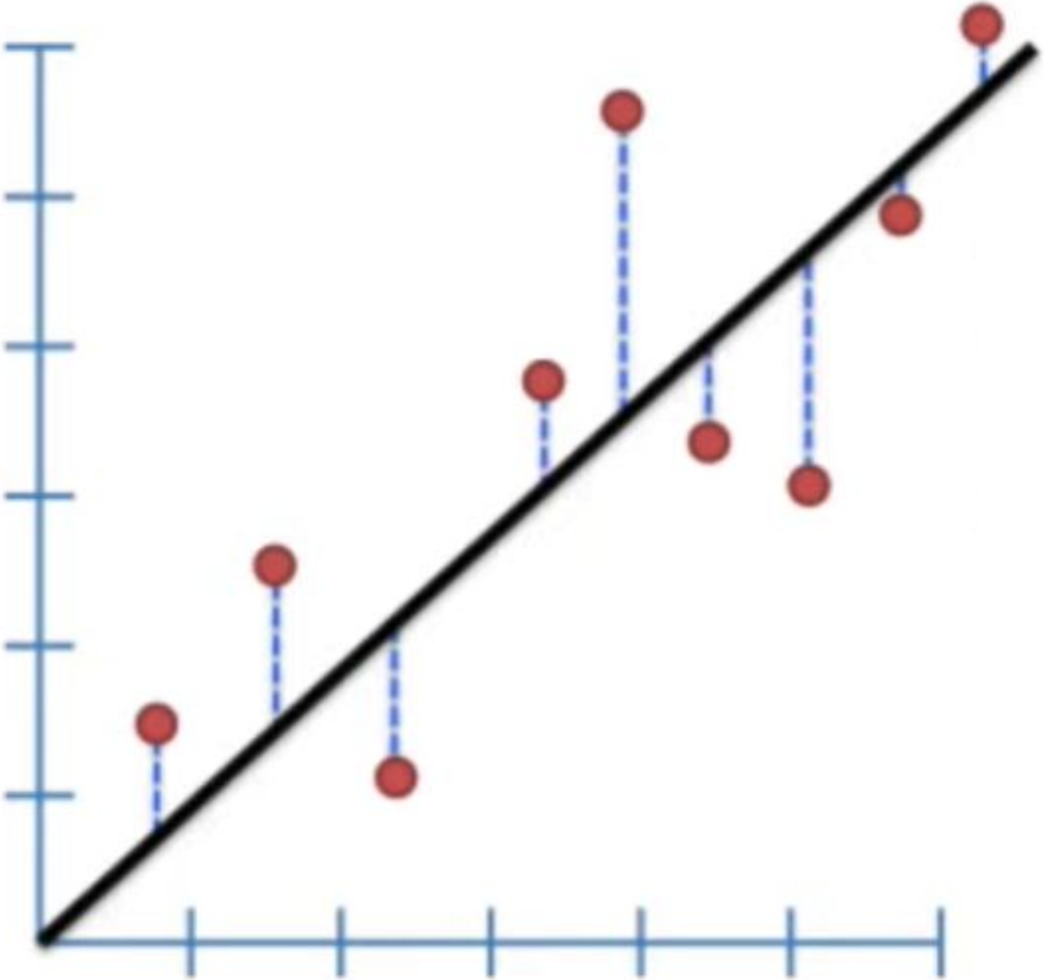
The fit gets worse. In this case the sum of squared residuals = 31.71



Regressão Linear

The generic line equation is:

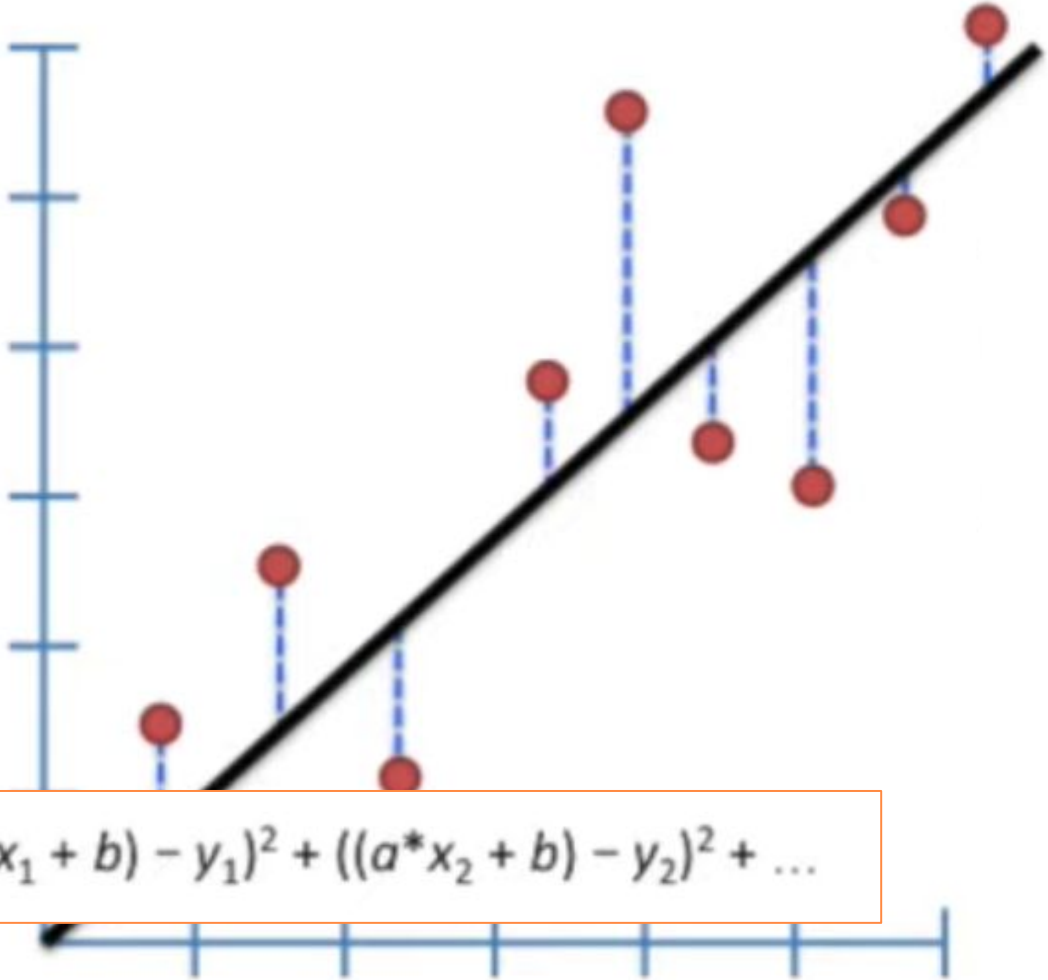
$y = a^*x + b$



Regressão Linear

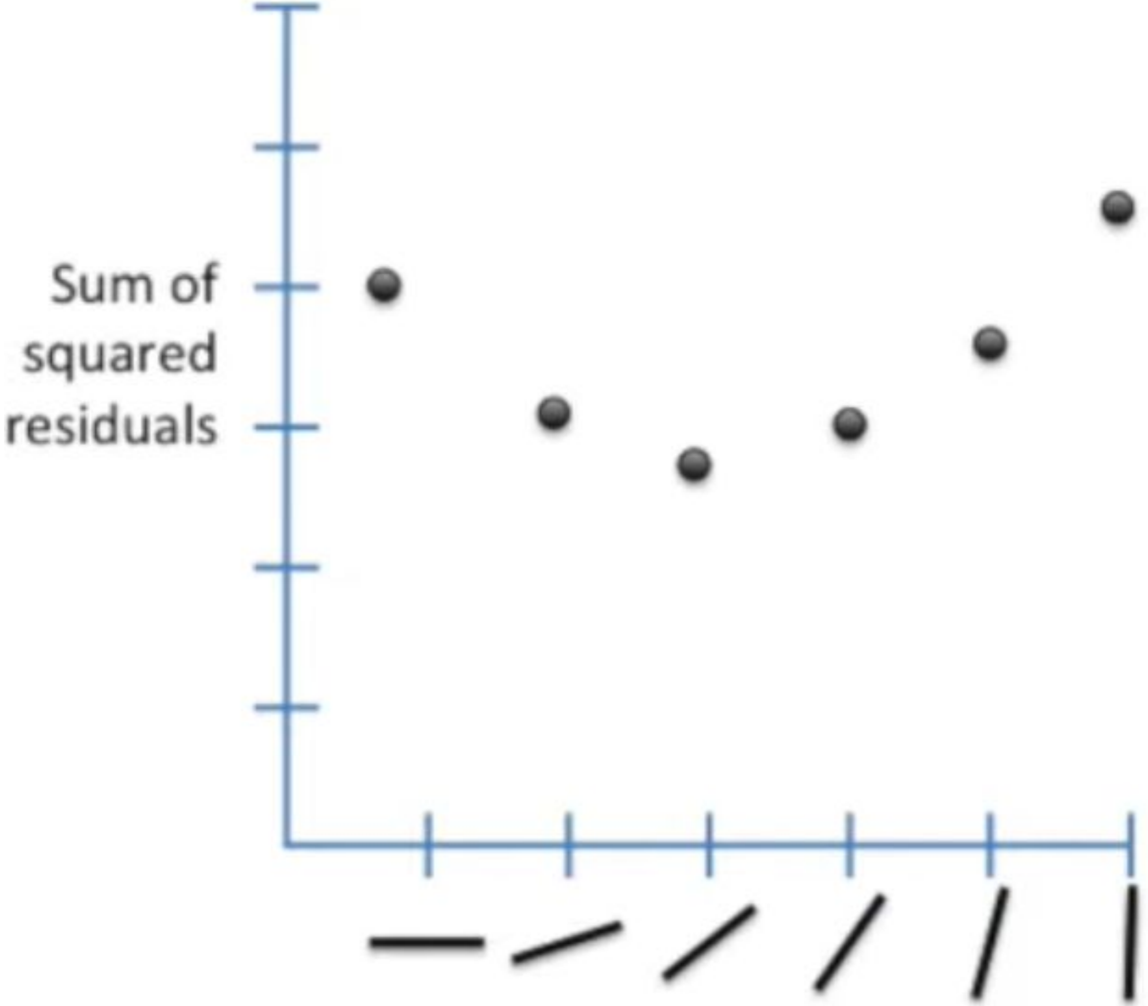
The generic line equation is:

$y = a^*x + b$

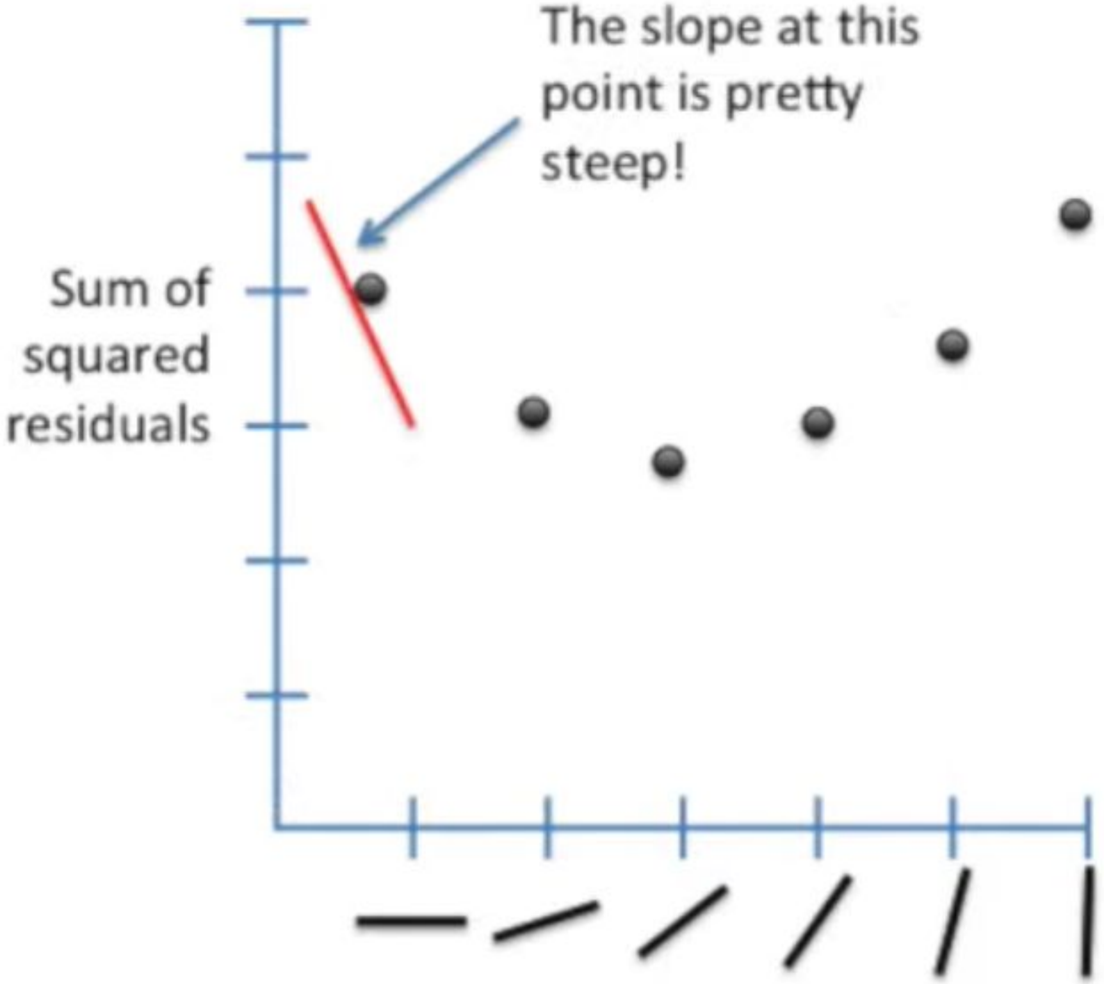


Sum of squared residuals = $((a^*x_1 + b) - y_1)^2 + ((a^*x_2 + b) - y_2)^2 + \ldots$

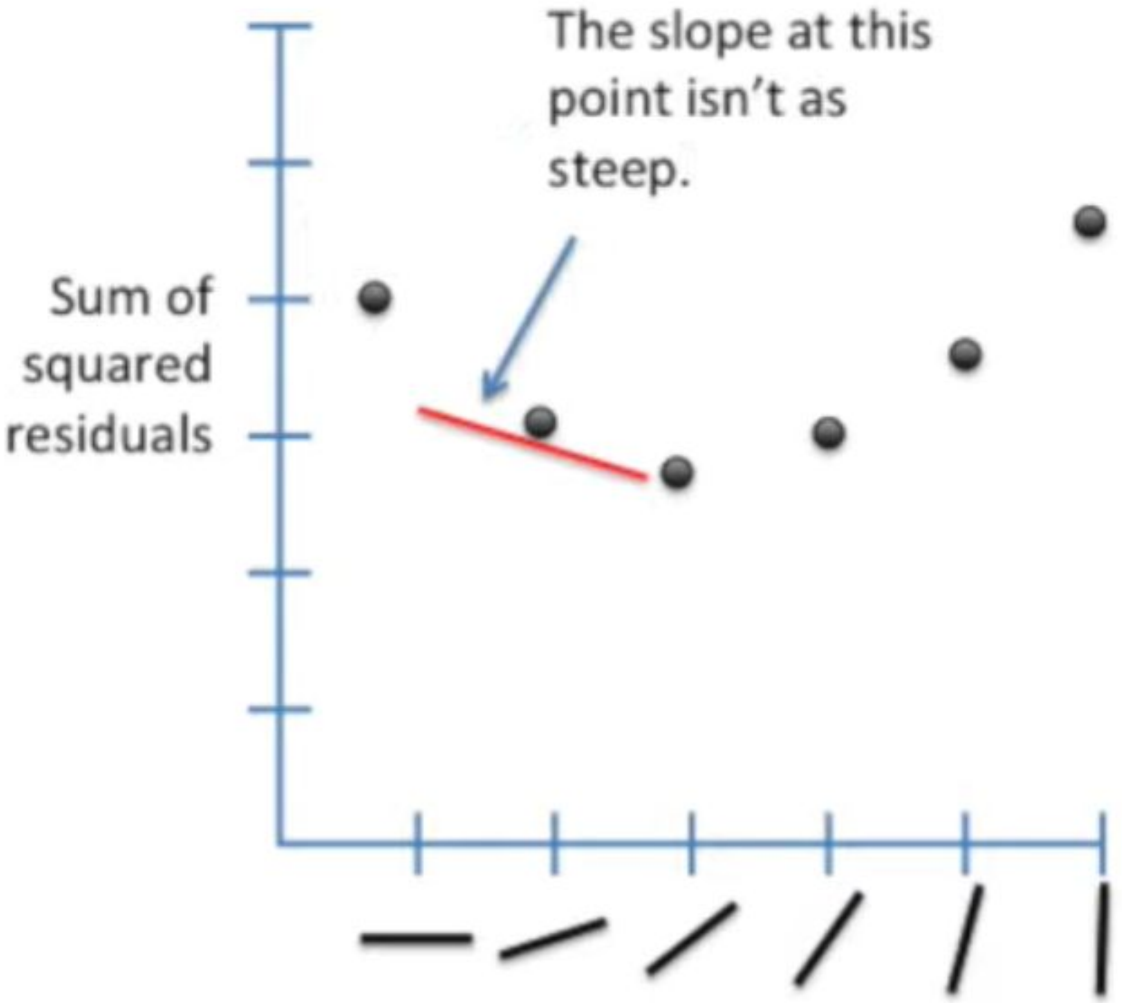
Regressão Linear



Regressão Linear

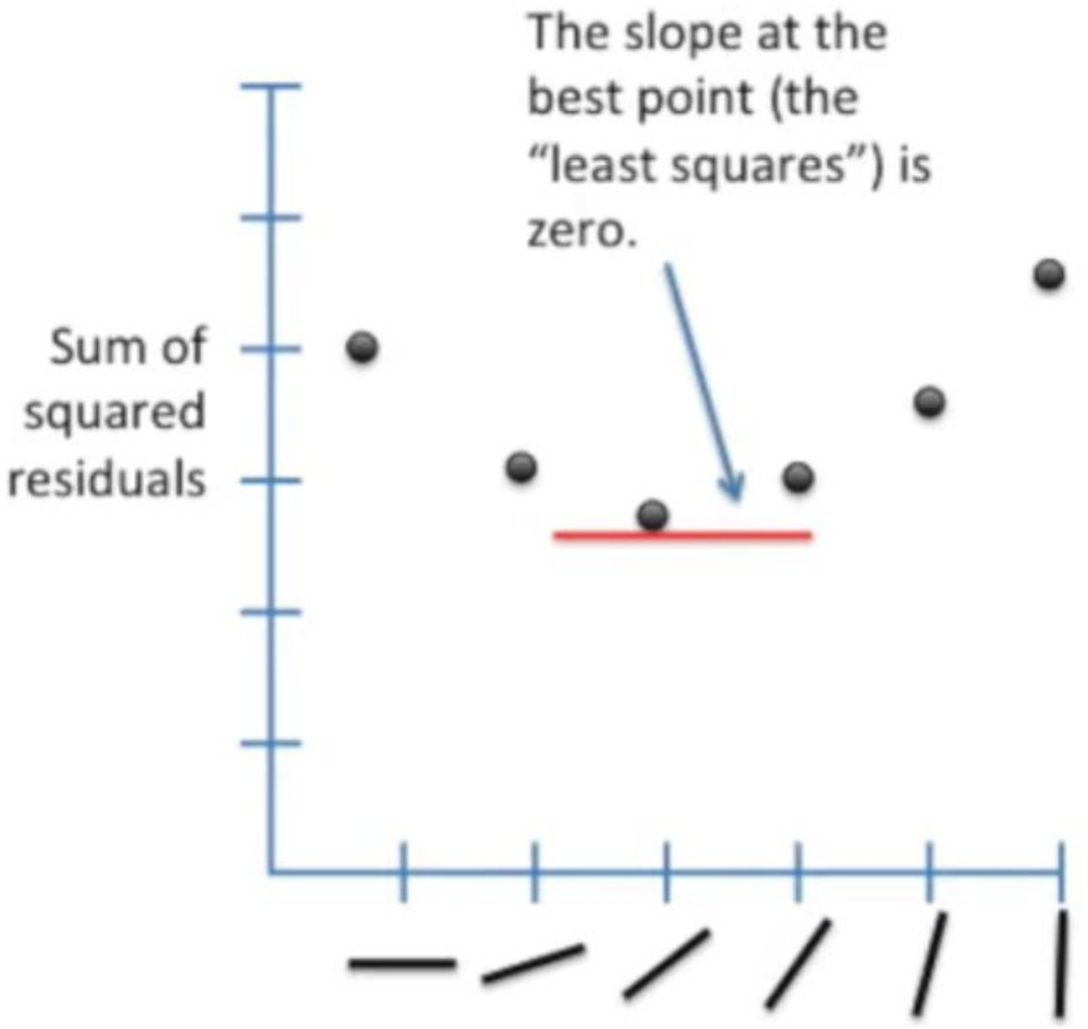


Regressão Linear



Regressão Linear

O ponto onde a inclinação da função de erro (derivada) é zero minimiza o erro (soma de erros ao quadrado)



Regressão Linear

Com isto, conseguimos calcular os parâmetros da equação linear:

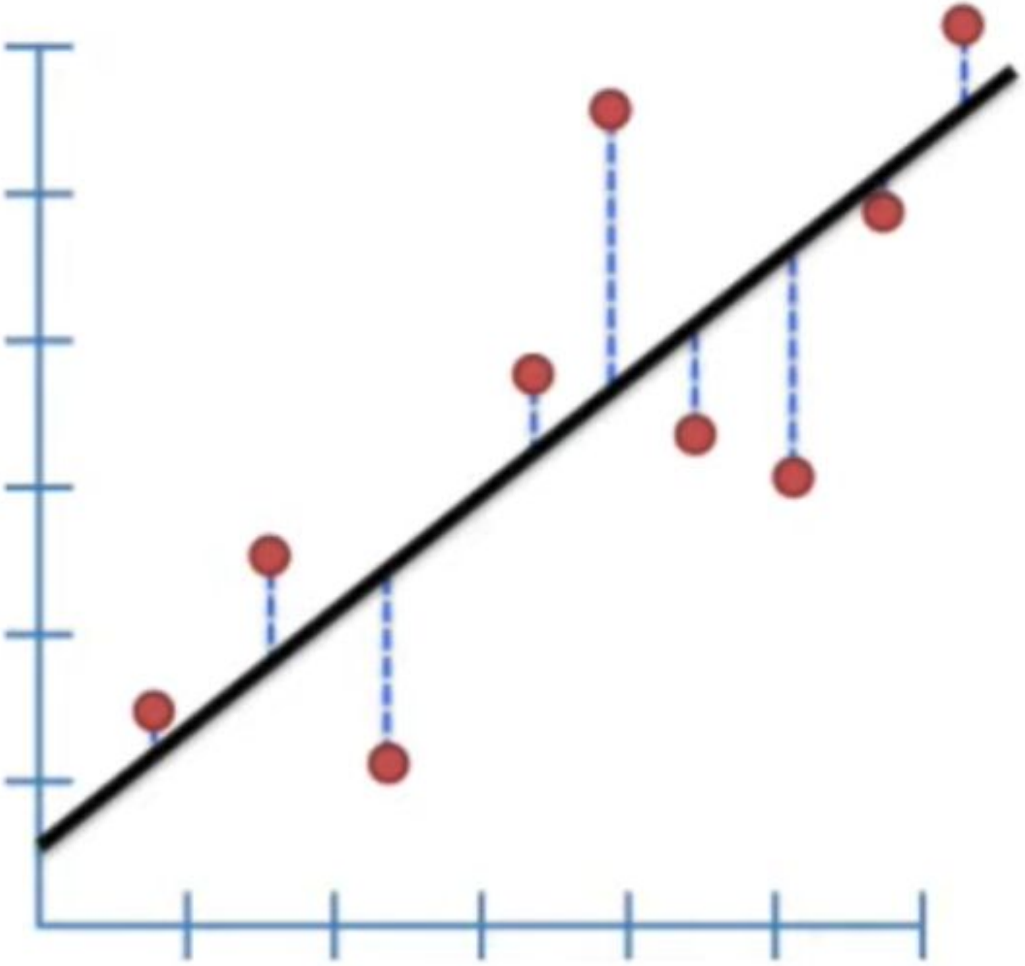
Y = **a**X + **b**

Onde:

- **a** (inclinação)
- **b** (ponto de interceptação)

No problema exemplificado:

Y = **0,77**X + **0,66**



Regressão Linear

Principais conceitos:

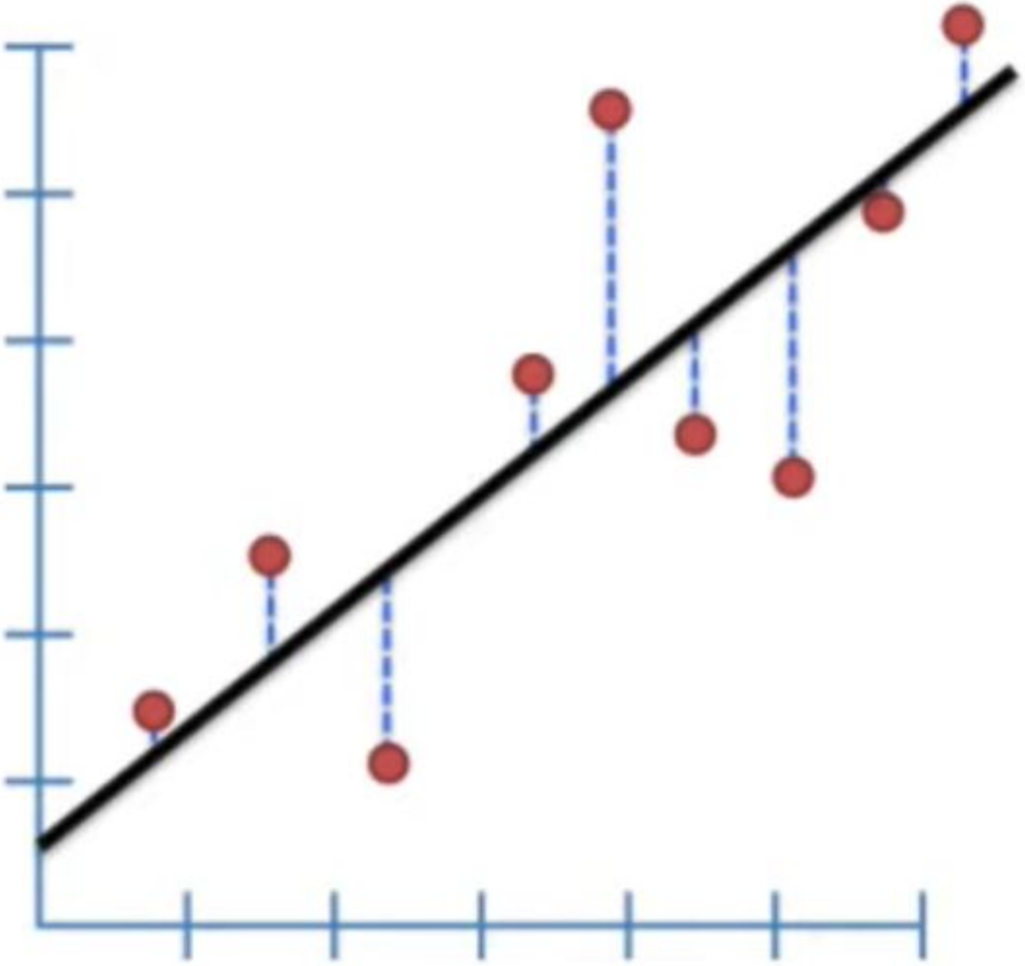
- O propósito é encontrar os parâmetros da função linear:

  $$Y = aX + b$$

- Para isto, precisamos minimizar o erro de **Y**

  Isto é feito por meio da minimização da soma do quadrado das distâncias entre o valor real (pontos vermelhos) e o valor estimado (a linha)

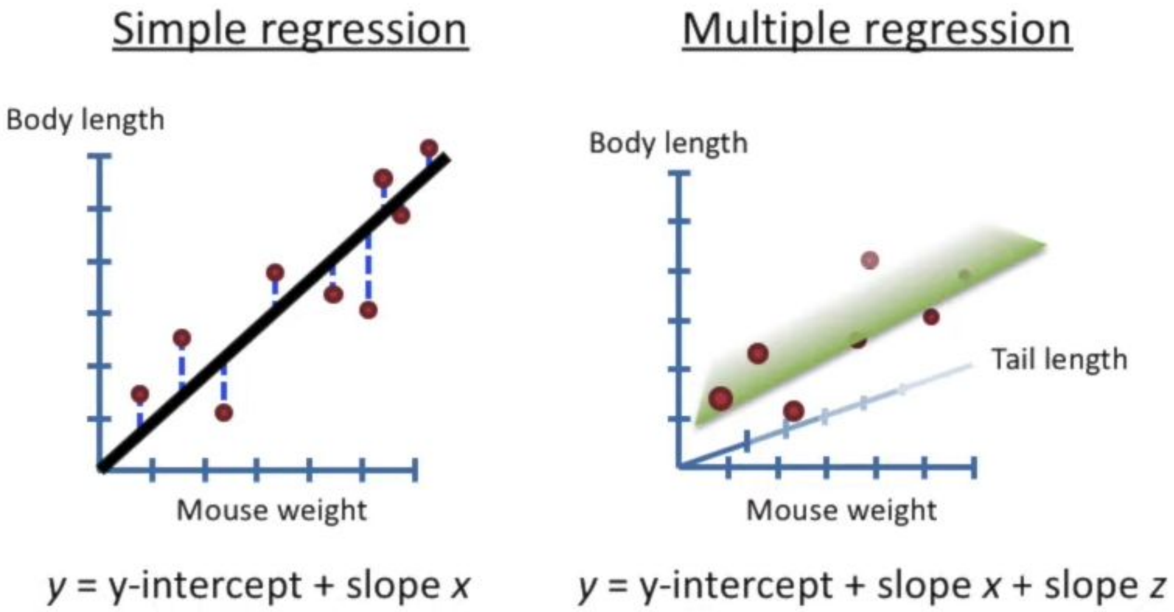
- Isto é possível calculando quando a derivada da soma dos quadrados é igual a zero



Regressão Linear Múltipla e Polinomial

Regressão Linear Múltipla

Na maioria dos casos, nós teremos mais de uma variável independente



Regressão Linear Múltipla

Na maioria dos casos, nós teremos mais de uma variável independente

- A equação vai ser equivalente à equação simples, porém com mais variáveis

$$Y = b_0 + b_1X_1 + b_2X_2 + \dots + b_nX_n + \mathcal{E}$$

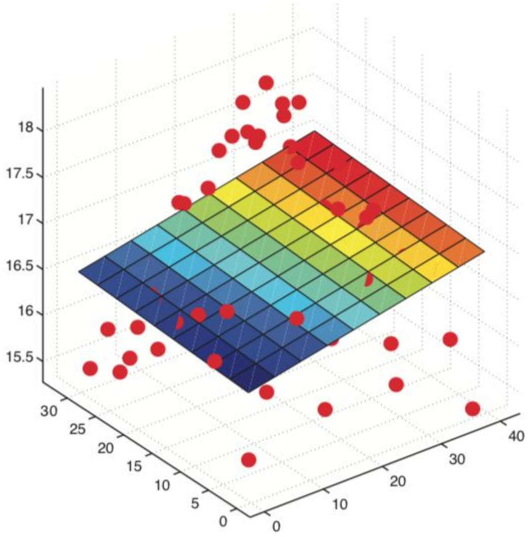
Regressão Linear Múltipla

Na maioria dos casos, nós teremos mais de uma variável independente

- A equação vai ser equivalente à equação simples, porém com mais variáveis

$$Y = b_0 + b_1X_1 + b_2X_2 + \ldots + b_nX_n + \mathcal{E}$$

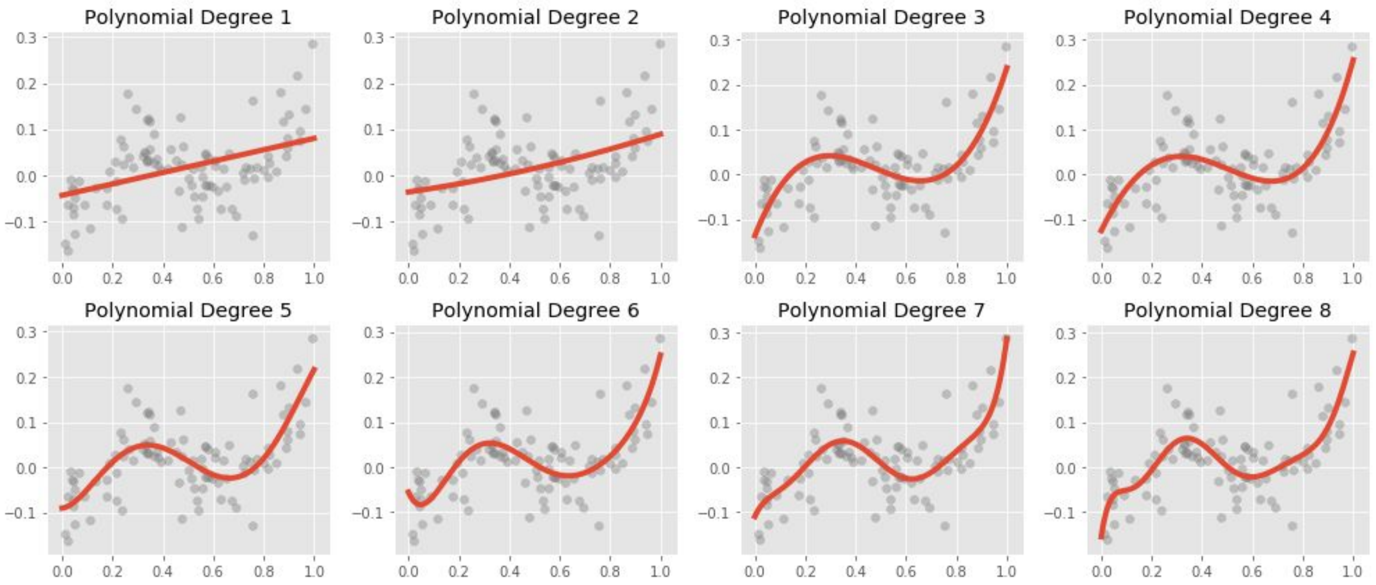
- Neste caso, a linha se torna
  - um plano - 2 variáveis independentes
  - um hiperplano - para mais variáveis independentes

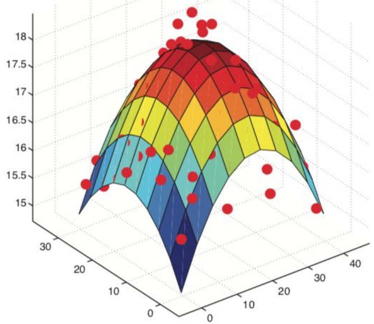


Expansão da Função Base

Regressão Linear pode modelar relações **não-lineares**

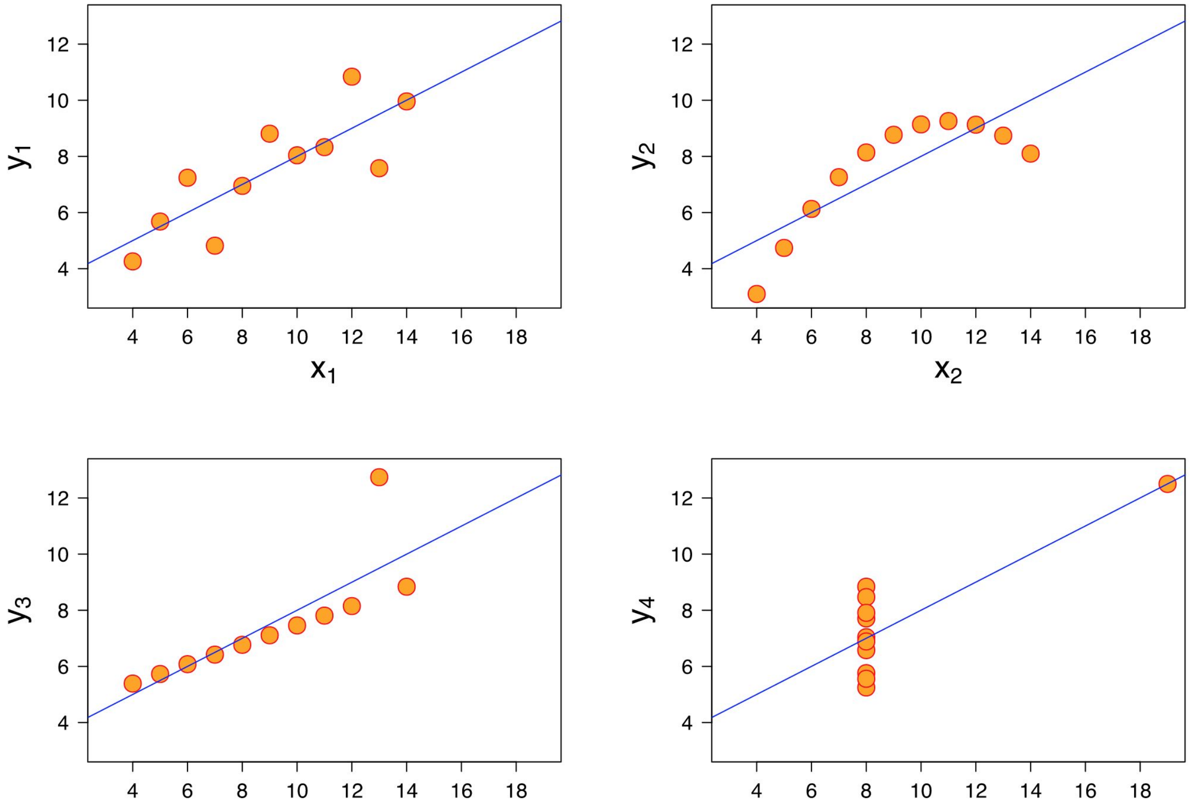
- Substituindo X por alguma função não-linear
  - Ex.: Polinomial





(b)

[Warning] Quarteto de Anscombe



[Warning] Quarteto de Anscombe

Estatística descritiva pode nos enganar

Visualize os dados!

Propriedade	Valor
Média de *x* em cada caso	9 (exato)
Variância de *x* em cada caso	11 (exato)
Média de *y* em cada caso	7,50 (em até duas casas decimais)
Variância de *y* em cada caso	4,122 ou 4,127 (em até 3 casas decimais)
Correlação entre *x* e *y* em cada caso	0,816 (em até 3 casas decimais)
Linha de regressão linear em cada caso	$y = 3,00 + 0,500x$ (em até 2 e 3 casas decimais, respectivamente)

Exemplo
Regressão Linear em Python

Aprendizado Supervisionado em Python

A Scikit Learn oferece muitos modelos para Aprendizado Supervisionado
Todos seguem a mesma API:

```
from sklearn[.desired_model] import [Estimator]

model = Estimator(params)                # Cria um modelo
model.fit(X_train,Y_train)               # 'Treina' o modelo com a base de treinamento

predictions = model.predict(X_test)      # Utiliza o modelo treinado para fazer predições
                                         # sobre a base de testes
```

Regressão Linear em Python

Classes e métodos para Regressão Linear:

```
from sklearn.linear_model import LinearRegression # importa o modelo de Regressão

regressor = LinearRegression()                # Cria um modelo
regressor.fit(X_train,Y_train)                # Treina o modelo com a base de treinamento

predictions = regressor.predict(X_test)       # Utiliza o modelo treinado para fazer predições
                                              # sobre a base de testes
```

Regressão Linear em Python

```
from sklearn.linear_model import LinearRegression

regressor = LinearRegression(normalize=False, copy_X=True)
```

- **normalize -** If True, the regressors X will be normalized before regression
- **copy_X -** If True, X will be copied; else, it may be overwritten.
- **fit_intercept -** If False, no intercept will be used in calculations

regressor.**coef_** - Estimated coefficients for the linear regression problem

regressor.**intercept_** - Independent term in the linear model.

Métodos:

`fit`(self, X, y[, sample_weight])	Fit linear model.
`get_params`(self[, deep])	Get parameters for this estimator.
`predict`(self, X)	Predict using the linear model.
`score`(self, X, y[, sample_weight])	Return the coefficient of determination R^2 of the prediction.

Exemplo - Diabetes

Modela o relacionamento entre o índice de massa corporal (BMI) e uma medida quantitativa do diabetes

Vamos tentar inferir esta medida quantitativa

- Importando as bibliotecas necessárias

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
from sklearn import datasets
from sklearn.model_selection import train_test_split
```

Exemplo - Diabetes

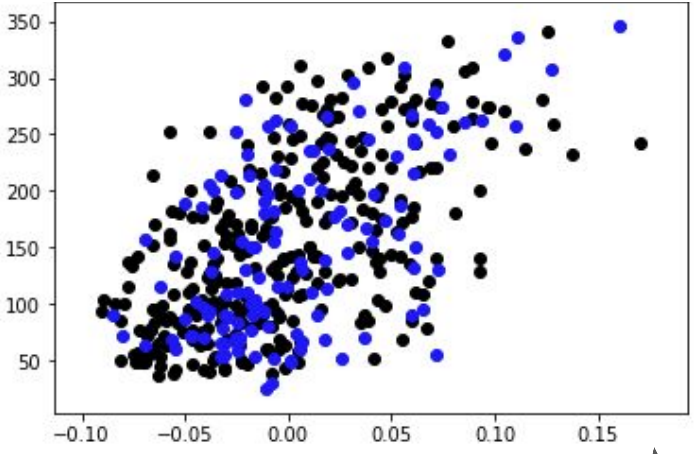
- Carrega a base e divide em subconjuntos:
  - Treinamento (70%) e testes (30%)

```python
diabetes = datasets.load_diabetes()

X = diabetes.data[:, np.newaxis, 2] # Body mass index
Y = diabetes.target

# Split the data into training/testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size=.7)
```

- Exibe a base

```python
plt.scatter(X_train, Y_train,  color='black')
plt.scatter(X_test, Y_test,  color='blue')

plt.show()
```

Exemplo - Diabetes



- Carrega a base e divide em subconjuntos:
  - Treinamento (70%) e testes (30%)

```
diabetes = datasets.load_diabetes()

X = diabetes.data[:, np.newaxis, 2] # Body mass index
Y = diabetes.target

# Split the data into training/testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size=.7)
```

- Exibe a base

```
plt.scatter(X_train, Y_train,  color='black')
plt.scatter(X_test, Y_test,  color='blue')

plt.show()
```

Exemplo - Diabetes

- Cria o classificador e treina-o

```
from sklearn import linear_model
# Create linear regression object
model = linear_model.LinearRegression()

# Train the model using the training sets
model.fit(X_train, Y_train)
```

Exemplo - Diabetes

- Cria o classificador e treina-o

```python
from sklearn import linear_model
# Create linear regression object
model = linear_model.LinearRegression()

# Train the model using the training sets
model.fit(X_train, Y_train)
```

- Usa-o para predizer os valores desejados

```python
from sklearn.metrics import mean_squared_error, r2_score
# Make predictions using the testing set
Y_pred = model.predict(X_test)

# The coefficients
print('Coefficients:', model.coef_)

# The mean squared error
print("Mean squared error: %.2f"
      % mean_squared_error(Y_test, Y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(Y_test, Y_pred))
```

Exemplo - Diabetes

- Cria o classificador e treina-o

```
from sklearn import linear_model
# Create linear regression object
model = linear_model.LinearRegression()

# Train the model using the training sets
model.fit(X_train, Y_train)
```

- Usa-o para predizer os valores desejados

```
from sklearn.metrics import mean_squared_error, r2_score
# Make predictions using the testing set
Y_pred = model.predict(X_test)

# The coefficients
print('Coefficients:', model.coef_)

# The mean squared error
print("Mean squared error: %.2f"
      % mean_squared_error(Y_test, Y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(Y_test, Y_pred))
```

```
('Coefficients:', array([969.08907833]))
('Intercept:', 151.8305010617645 5)
Mean squared error: 3908.65
Variance score: 0.33
```

$y = 151{,}83 + 969{,}09 * x$

Exemplo - Diabetes

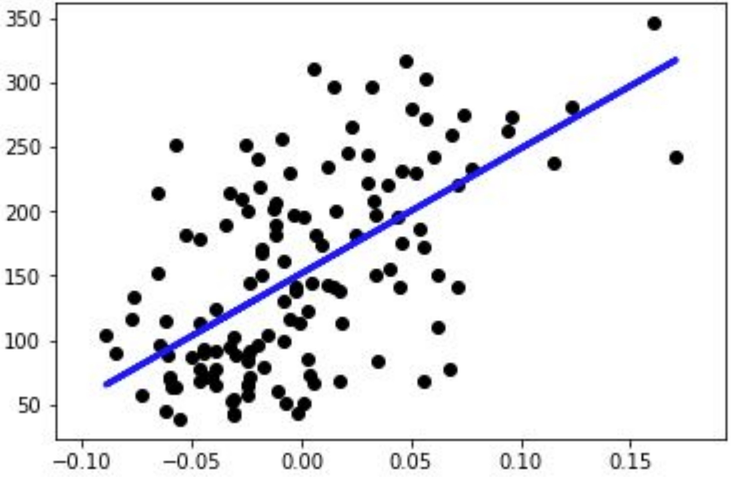
- Exibe os resultados

```
# Plot outputs
plt.scatter(X_test, Y_test,  color='black')
plt.plot(X_test, Y_pred, color='blue', linewidth=3)

plt.show()
```

Exemplo - Diabetes

- Exibe os resultados

```
# Plot outputs
plt.scatter(X_test, Y_test,  color='black')
plt.plot(X_test, Y_pred, color='blue', linewidth=3)

plt.show()
```



Hands on

Regressão Linear Múltipla

Hands on

Regressão Linear

Hands on - USA Housing

Preço de imóveis nos EUA como uma **Regressão Linear Múltipla**

- Carregue a base de dados [USA Housing]
- Defina os vetores X e y a partir de 'data', contendo:
    - X - 'Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms', 'Avg. Area Number of Bedrooms', 'Area Population'
    - y - 'Price'
- Verifique se a base precisa de algum tratamento de dados
- Divida a base de dados em treinamento e testes
- Crie o modelo, treine-o e avalie sua performance na base de treinamento

Hands on - USA Housing

- Treine-o novamente, agora com validação cruzada e compare com o resultado anterior
- Execute o modelo na base de testes
- Avalie o erro médio quadrático e o R^2 e compare com os resultados da validação cruzada
  - ***mean_squared_error***
  - ***r2_score***

```python
from sklearn.metrics import mean_squared_error, r2_score

y_pred = lm.predict(X_test)
plt.scatter(y_test, y_pred)

# The mean squared error
print("Mean squared error: %.2f"
      % mean_squared_error(y_test, y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(y_test, y_pred))
```

```
Mean squared error: 10460958907.21
Variance score: 0.92
```

Ver USA_Housing Linear_Regression

Hands on - USA Housing

- Exiba alguns resultados - Sugestões:
    - Dispersão de y_test vs y_pred
      ```
      plt.scatter(y_test, y_pred)
      ```
    - Dispersão de pred vs. y
      ```
      plt.figure(figsize=(20, 7))
      idx = np.argsort(y)
      preds = model.predict(X)

      plt.scatter(range(len(y)), preds[idx] , label= "Predictions")
      plt.scatter(range(len(y)), y[idx], label="Orignal values")
      plt.legend()
      ```
    - Distribuição do erro
      ```
      sns.distplot((y_test-y_pred), bins=50)
      ```

Para continuar...

- Kevin P. Murphy. **Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)**. 1a Edição: The MIT Press, 2012.
  - Capítulo 7 - Linear regression - 7.1 a 7.3

- Joel Grus. **Data Science do Zero: Primeiras regras com o Python.** 1ª Edição:Alta Books, 2016.
  - Capítulo 14 - Regressão Linear Simples

Outros:

- StatQuest: Fitting a line to data, aka least squares, aka linear regression
- How to calculate linear regression using least square method
- StatQuest: Linear Models Pt.1 - Linear Regression
- CNUM-019 Método dos Mínimos Quadrados Linear [MMQ]
- Evaluating a Linear Regression Model

C.E.S.A.R
school