Biblioteca
Scikit-learn

Prof. Dr. Diego Bruno

Education Tech Lead na DIO
Doutor em Robótica e *Machine Learning* pelo ICMC-USP

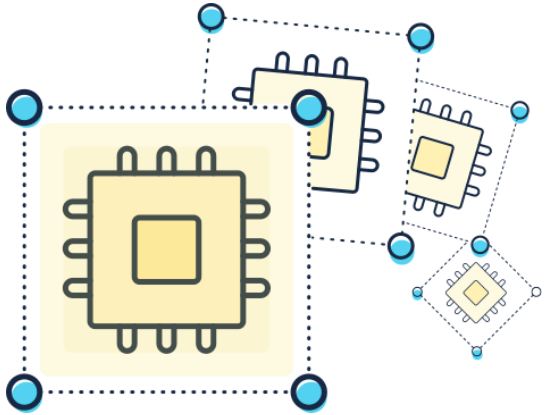
Biblioteca Scikit-learn

Prof. Dr. Diego Bruno

Machine Learning

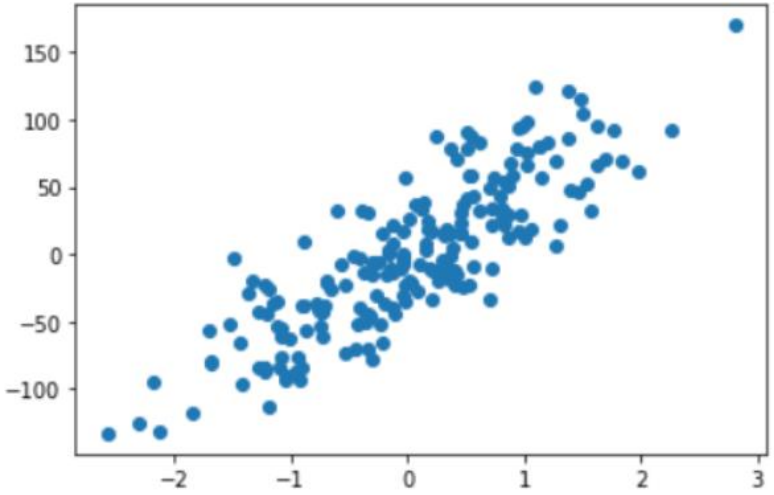
Utilizando a biblioteca

Esta biblioteca dispõe de ferramentas simples e eficientes para análise preditiva de dados, é reutilizável em diferentes situações, possui código aberto, sendo acessível a todos e foi construída sobre os pacotes **NumPy, SciPy e matplotilib**.



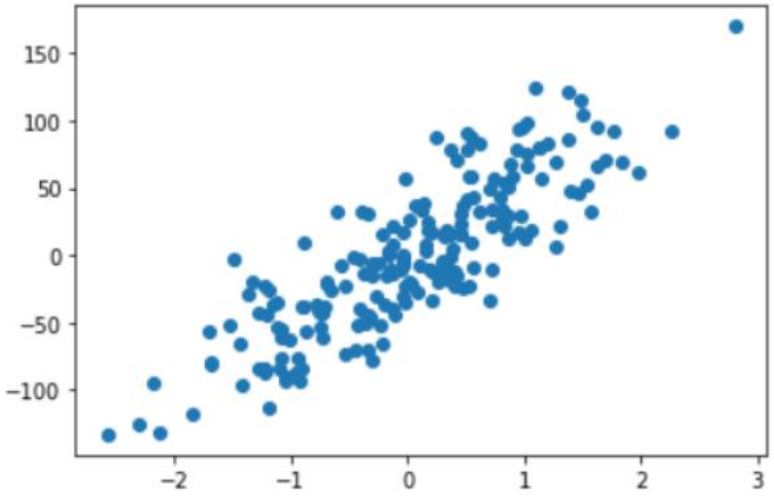
Scikit-learn

Neste exemplo iremos criar uma massa de dados com 200 observações, com apenas uma variável preditora, que será a variável *x* e a variável target, que será a *y*. Para isso indicamos os parâmetros *n_samples* = *200* e *n_features* = *1*. O parâmetro *noise* define o quão dispersos os dados estarão um dos outros.



Scikit-learn – Etapa 1

```
from sklearn.datasets import make_regression
#gerando uma massa de dados:
x, y = make_regression(n_samples=200, n_features=1, noise=30).
```

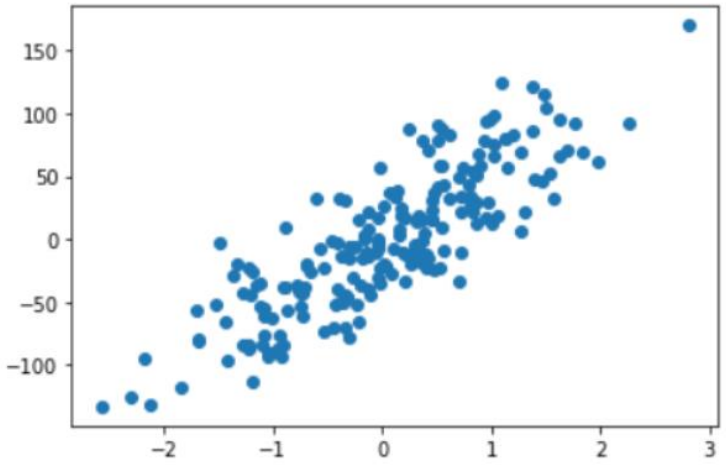


Scikit-learn

Utilizaremos o pacote *matplotlib*, com o módulo *pyplot* e a função *scatter()*, que criará o gráfico, e função *show()* que o exibirá na tela.

```
In [2]: import matplotlib.pyplot as plt

        # mostrando no gráfico:
        plt.scatter(x,y)
        plt.show()
```



Scikit-learn

Com os dados gerados, já podemos iniciar a criação de nosso modelo de machine learning. Para isso utilizaremos o módulo *linear_model*, e a função *LinearRegression()*.

```
from sklearn.linear_model import LinearRegression
# Criação do modelo
modelo = LinearRegression()
```

Scikit-learn

Com os dados gerados, já podemos iniciar a criação de nosso modelo de machine learning. Para isso utilizaremos o módulo *linear_model*, e a função *LinearRegression()*.

```
from sklearn.linear_model import LinearRegression
# Criação do modelo
modelo = LinearRegression()
```

Scikit-learn

Após esta execução, o objeto ***modelo*** que acabamos de criar está pronto para receber os dados que darão origem ao modelo. Como não indicamos nenhum parâmetro específico na função, estamos utilizando suas configurações padrão.

Agora precisamos apenas apresentar os dados ao modelo, e para isso temos o ***método fit().*** Na documentação da função podemos conferir todos os métodos que ela possui.

Scikit-learn – Etapa 2

Após esta etapa, nosso modelo de *machine learning* está pronto e **podemos utilizá-lo para prever dados desconhecidos**. Simplificando este primeiro entendimento, vamos apenas visualizar a **reta de regressão linear** que o modelo gera, com os mesmos dados que criaram o modelo. Para isso iremos utilizar o método ***predict()*,** indicando que queremos aplicar a previsão nos valores de ***x***. O resultado do método será uma previsão de *y* para cada valor de *x* apresentado.

Scikit-learn – Etapa 2

modelo.predict(x)

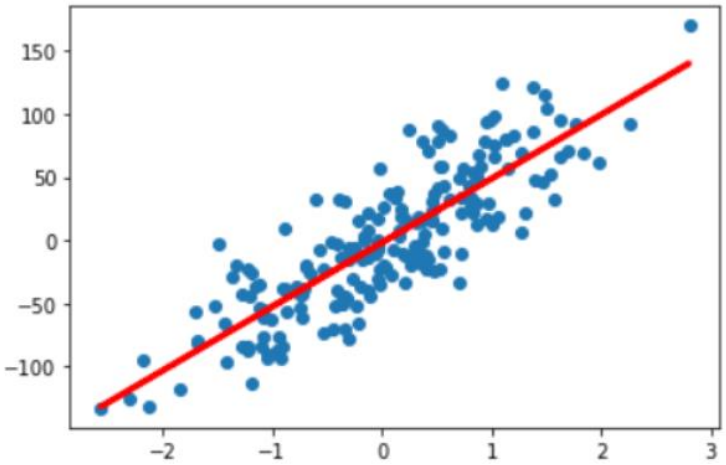
```
In [5]: modelo.predict(x)

Out[5]: array([  35.6203308 ,   -3.17423057,  -45.61500036,  -35.55761336,
               -36.64951896,   -0.26583513, -109.2048109 ,   -4.03112327,
               -74.775194  ,  -40.03149364,  -63.76819387,  -79.38003634,
                15.28045518,   76.15050169,   -9.89632462,   16.9883385 ,
               -17.40250852,   45.81916612,  -56.95678911,   -3.81705028,
                -8.97268015,    6.0599397 ,  -38.26310679,   87.85259224,
                34.68692551,   19.82857138,  -63.84119353,   -2.66023192,
                38.05629487,  -28.88635211,    8.68916995, -131.85857672,
                80.7606594 ,   29.64818836,   21.09928963,   -8.59112398,
                84.28593443,  -22.35286161,  -39.30567938,  -16.93842323,
                -1.18700474,   13.989157  ,  -17.13834937,  112.76375682,
                77.7204409 ,  -13.03318035,  -19.1297018 ,   -8.05979909,
                 8.41315509,  -58.33337327,   42.41831639,    6.5197212 ,
                72.27310044,    7.52191056,   43.53561478,   56.02630839,
```

Scikit-learn – Etapa 2

A função *plot()* do pacote *pyplot* gera uma reta com os dados apresentados. Como já temos os dados de *x* e *y*, basta indicá-los na função. Assim, primeiramente montamos novamente o gráfico de *x* e *y* original com a função *scatter()*, e somamos a ele a *reta de*

```
In [6]: plt.scatter(x,y)
        plt.plot(x, modelo.predict(x), color='red', linewidth=3)
        plt.show()
```



Obrigado!

Machine Learning

Prof. Dr. Diego Bruno