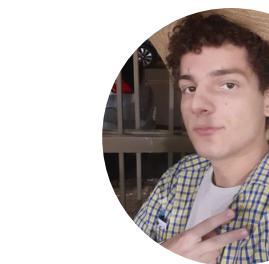
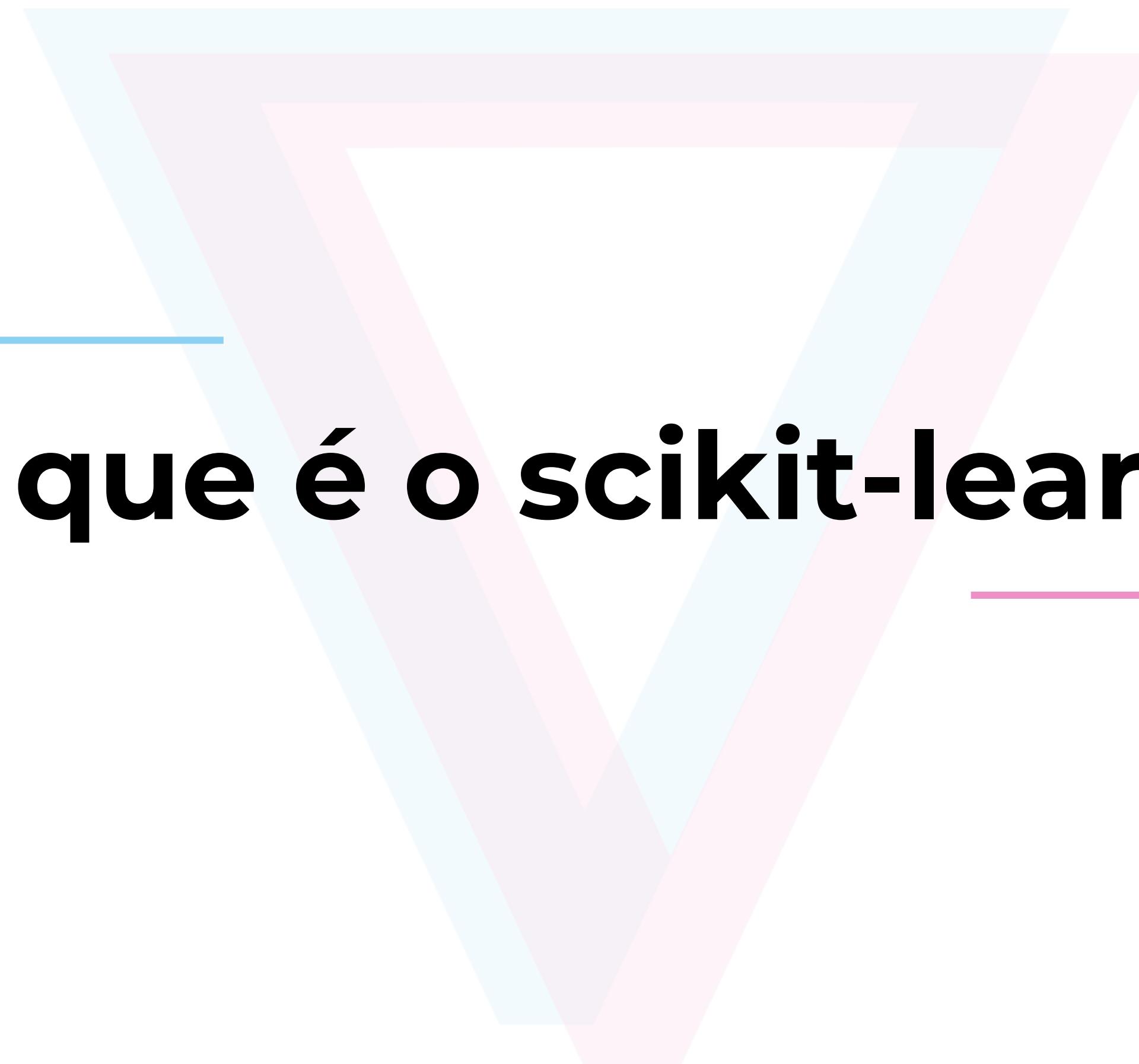


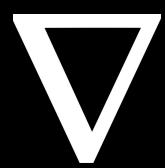
Introdução ao Scikit-learn



Murilo Leandro Garcia
@murilouco_louco



O que é o scikit-learn?



O que é o scikit-learn?

É uma biblioteca do python (sklearn) para aprendizado de máquina.

Ela é capaz de trabalhar com diversos modelos, sem que o programador se preocupe com o funcionamento interno matemático.

▽

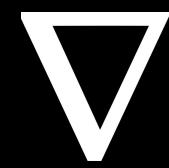


Aprendizado de Máquina?

► Aprendizado de Máquina?

Já sabemos como tratar dados (com o pandas) e obter informações estatísticas deles (como a média).

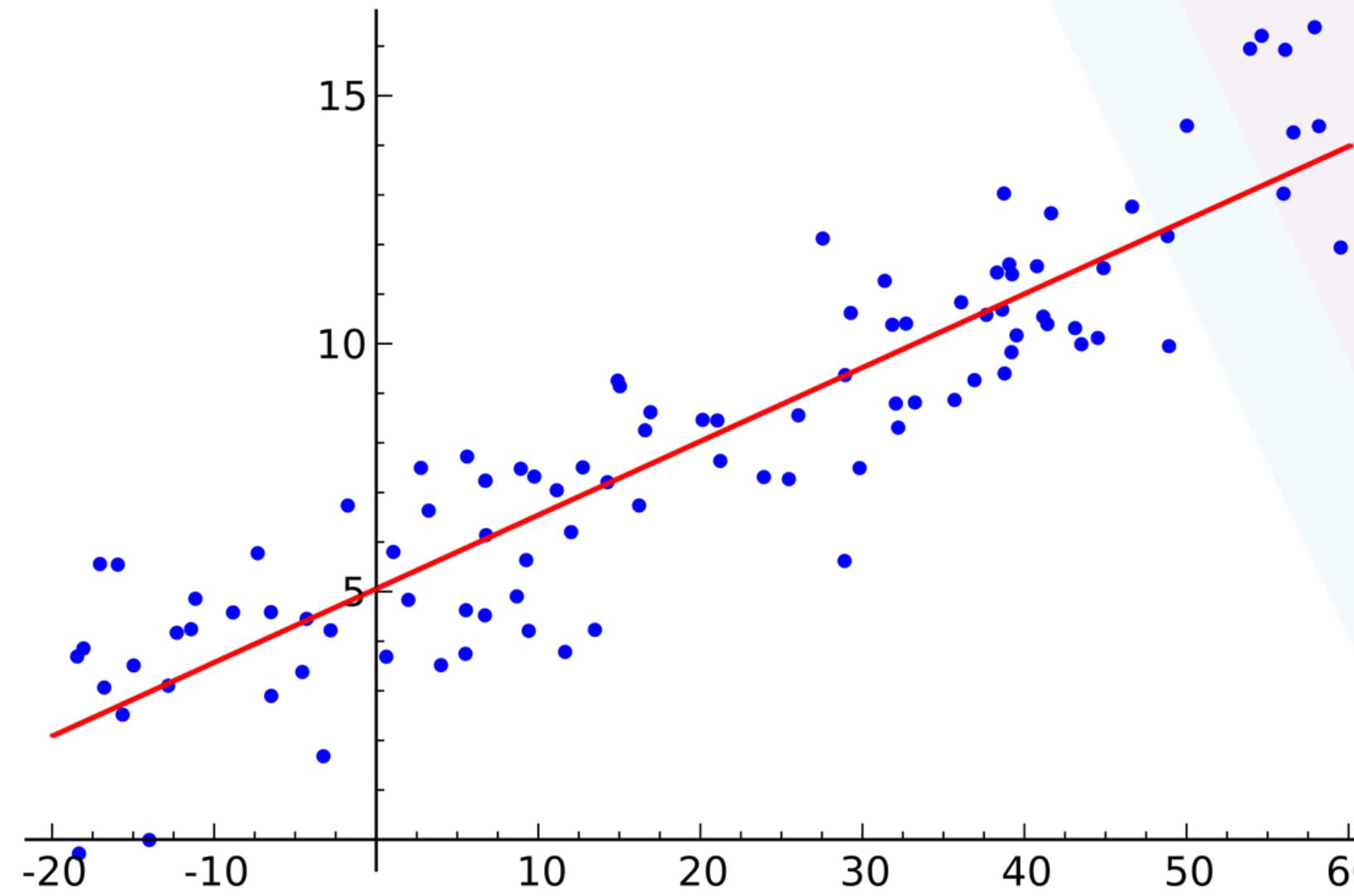
Queremos agora fazer previsões em cima desses dados.



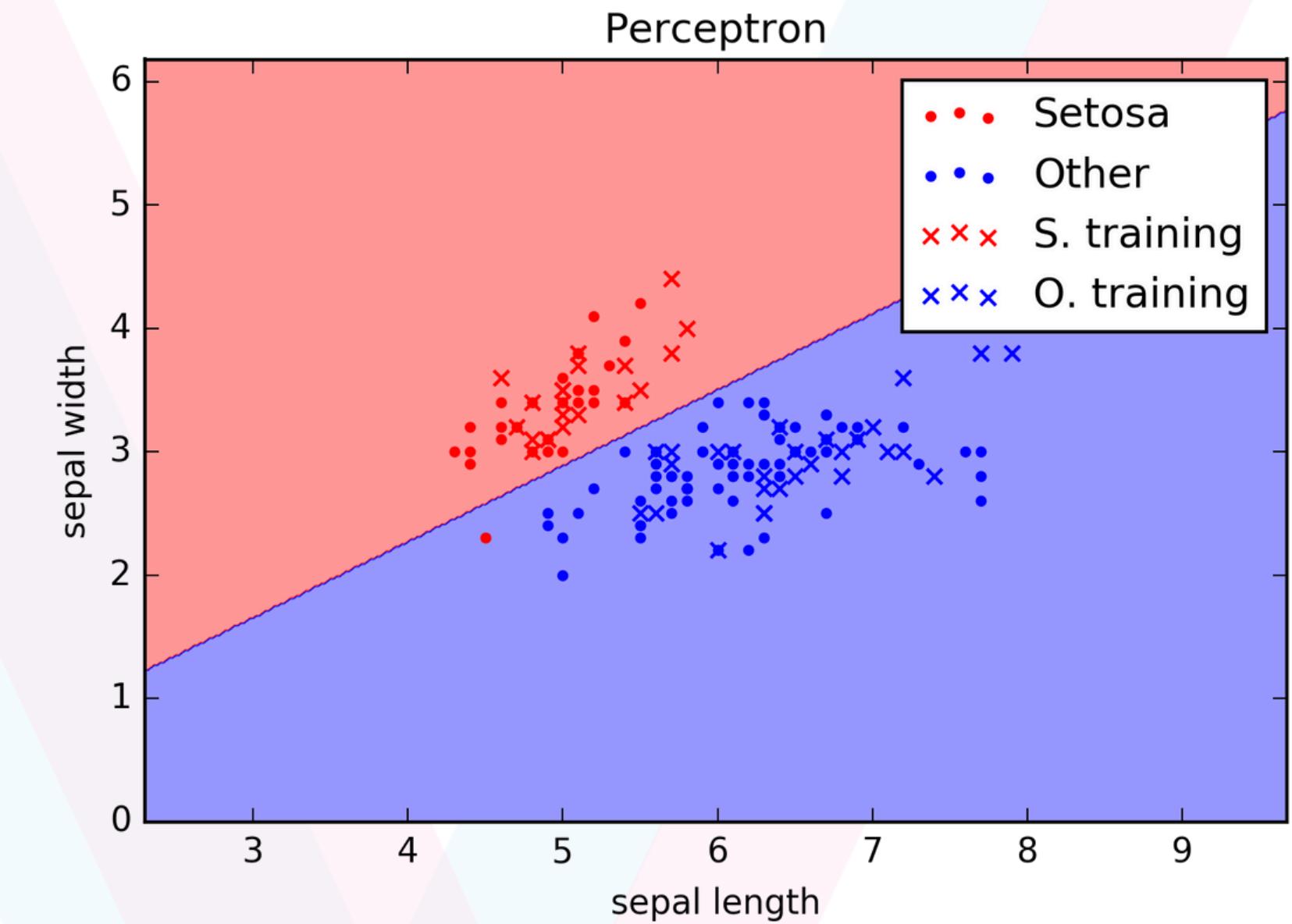
Aprendizado de Máquina?

Exemplos de modelos lineares:

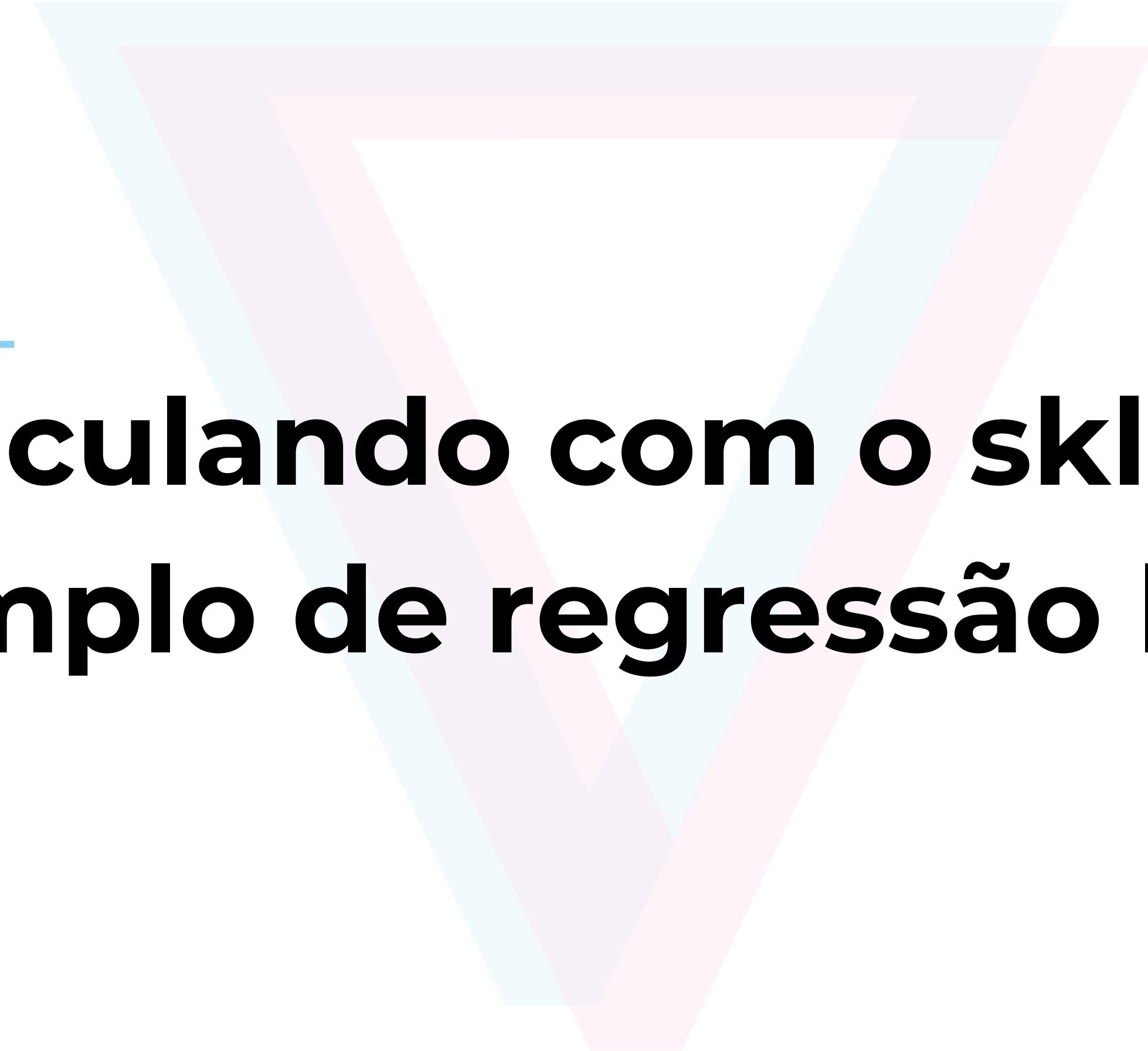
Regressão (linear)



Classificação (linear)



$$y = ax + b, \text{ queremos encontrar } 'a' \text{ e } 'b'$$



Calculando com o sklearn

(exemplo de regressão linear)

Instanciar um modelo

Importar a classe de modelos, depois instanciar um modelo dessa classe:

```
from sklearn import linear_model  
regressao_linear = linear_model.LinearRegression( )
```

Aprender

No regressão linear, a entrada sempre será um vetor, enquanto a saída é um único número.

Por exemplo, uma sorveteria quer fazer regressão linear do número de clientes (saída).

A entrada poderia ser um vetor de um único elemento: [temperatura], mas poderia ser um vetor de vários elementos: [temperatura, horário, preço].

Um dataset vai ser uma lista de vetores de entrada, e uma lista de saída.

Exemplo da sorveteria

A entrada é uma lista de vetores. Cada vetor contém somente um elemento (a temperatura).

A saída é uma lista de números, (a quantidade de clientes):

```
X = [[20], [21], [25], [29], [31], [32], [32], [34]]  
Y = [ 3, 4, 8, 12, 14, 15, 15, 17]
```



Aprender

Chamamos a função `.fit()` no nosso modelo, passando as entradas e saídas:

```
regressao_linear.fit(X, Y)  
  
print(regressao_linear.coef_) # coeficientes de X  
print(regressao_linear.intercept_) # termo independente
```

O item `'.coef_'` possui a lista de coeficientes de x, e `'.intercept_'` possui o termo independente;



Prever

Com nosso modelo treinado, agora podemos passar uma lista com novos vetores de entrada e adivinhar qual seria a saída deles:

```
X2 = [[40], [41], [17]]  
regressao_linear.predict(X2)
```

▽

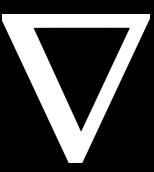


No colab





Classificação linear



Classificação

O dataset de entrada é uma lista de vetores, o dataset de saída é uma lista de categorias.

Exemplo de elementos da lista de entrada e de saída para pinguins:

$x = [\text{peso}, \text{grossura do bico}]$

$y = \text{'macho' ou 'fêmea'}$



Classificação

Instanciamento, aprendizado e predição:

```
from sklearn import svm  
classificador = svm.LinearSVC()  
  
classificador.fit(X, Y)  
classificador.predict([...])
```

Classificação

Podemos usar o método ‘.score(X, Y)’ para ver o quanto acurado é nosso modelo.

Ele vai comparar os valores de saída previstos com o .predict(X), e comparar com os valores de saída reais (Y).

Depois, vai calcular a taxa de acertos.

▽



No colab



Kernels

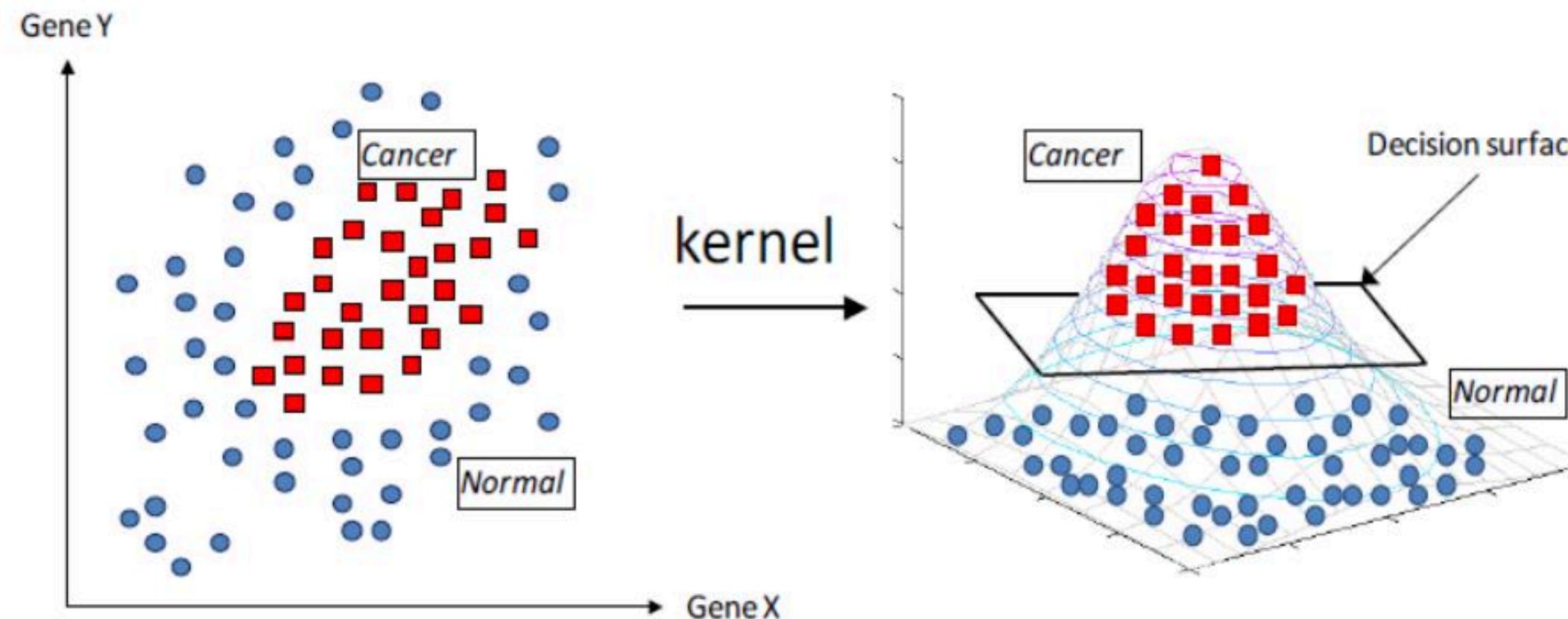
Kernels

Até agora vimos somente modelos que fazem previsões “linearmente”.

Problema: nem todo dataset pode ser modelado ou classificado por uma linha.

Kernels

Solução: mapear seu dataset em um espaço que pode ser modelado por linhas, fazer o machine learning linear nesse espaço, e depois desfazer essa transformação.



Kernels

Isso pode ser especificado no sklearn com o parâmetro “kernel=” na hora de instanciar um modelo.

▽



No colab



Clustering



Clustering

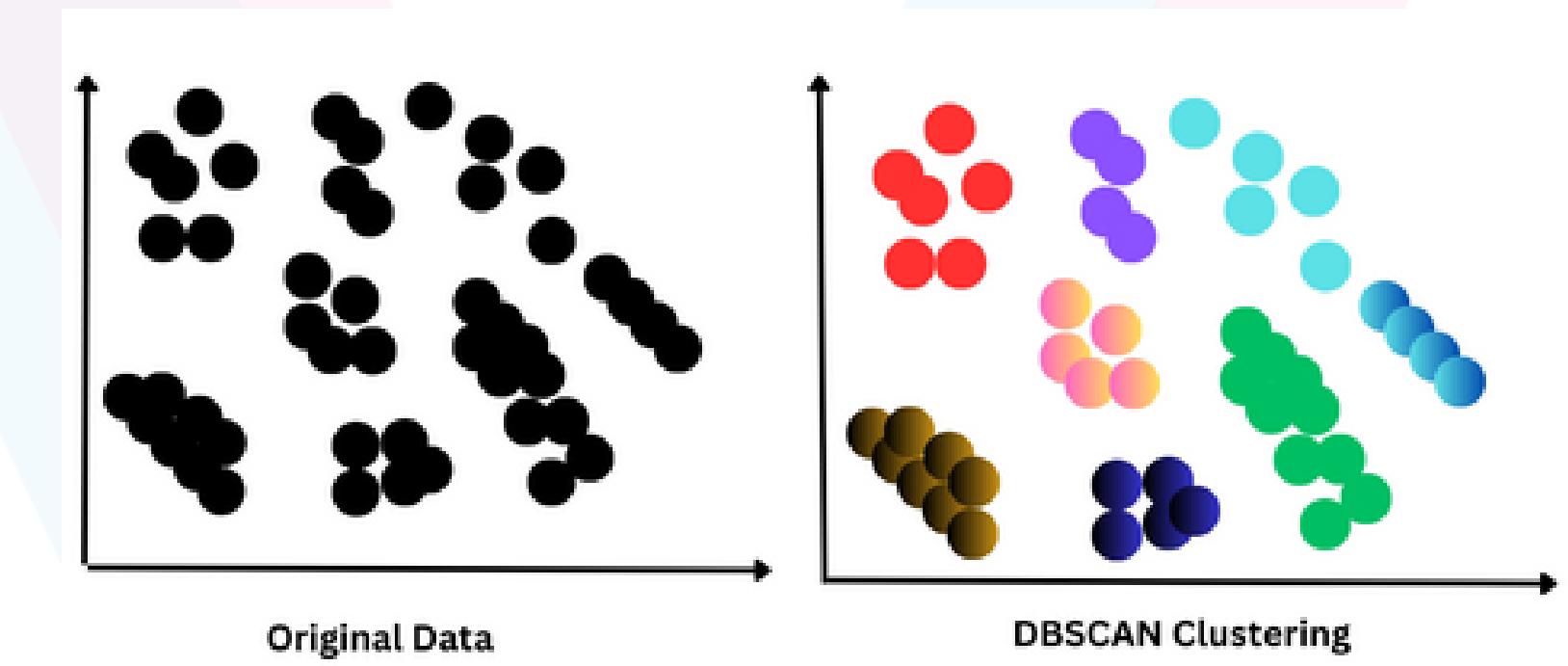
Clustering é uma forma de aprendizado de máquina, similar à classificação, porém não existem classes pré-determinadas. O próprio código vai criar as classes e classificar cada ponto.



DBSCAN

DBSCAN é um algoritmo de clustering muito famoso por ser muito eficiente. Possui um parâmetro obrigatório 'eps' que determina quão longe pontos de uma mesma classe podem estar.

```
from sklearn import cluster  
dbscan = cluster.DBSCAN(eps=2)  
dbscan.fit(data[['X', 'Y']])  
  
classes = dbscan.labels_
```



Outros

Os modelos que eu apresentei são os mais básicos e comuns, mas você pode fazer classificações, regressões e clusterings usando outros modelos:
https://scikit-learn.org/stable/user_guide.html

▽



No colab

