

# **Aplicações de Inteligência Artificial**

## **Parte III**

**Hitoshi Nagano, Ph.D.**

# roteiro

- métricas de performance
  - como sabemos que um modelo é melhor do que outro?
- dividindo os dados
  - como aumentar a chance de modelo ser bom para dados não conhecidos, ...
  - ao invés de dar resultados bons somente sobre os dados conhecidos.
- treinamento e predição com scikit-learn

# MÉTRICAS DE PERFORMANCE



**para regressão**

	<b>Gold</b>	<b>Pred</b>	<b>Erro ao quad.</b>
<b>cliente0</b>	<b>3</b>	<b>3,2</b>	<b>0,04</b>
<b>cliente1</b>	<b>1</b>	<b>0,1</b>	<b>0,81</b>
<b>cliente2</b>	<b>-2</b>	<b>-1,9</b>	<b>0,01</b>
<b>cliente3</b>	<b>0,3</b>	<b>-0,1</b>	<b>0,16</b>
<b>cliente4</b>	<b>0,6</b>	<b>0,7</b>	<b>0,01</b>
<b>MSE</b>			<b>0,21</b>

$$MSE = \frac{1}{n} \sum_{i=0}^{n-1} (\hat{y}^{(i)} - y^{(i)})^2$$

	<b>Gold</b>	<b>Pred</b>	<b>Erro ao quad.</b>
<b>cliente0</b>	<b>3</b>	<b>3,2</b>	<b>0,04</b>
<b>cliente1</b>	<b>1</b>	<b>0,1</b>	<b>0,81</b>
<b>cliente2</b>	<b>-2</b>	<b>-1,9</b>	<b>0,01</b>
<b>cliente3</b>	<b>0,3</b>	<b>-0,1</b>	<b>0,16</b>
<b>cliente4</b>	<b>0,6</b>	<b>0,7</b>	<b>0,01</b>
<b>MSE</b>			<b>0,21</b>

$$MSE = \frac{1}{n} \sum_{i=0}^{n-1} (\hat{y}^{(i)} - y^{(i)})^2$$

# Erro quadrático médio e sua raiz

***mean square error***

$$MSE = \frac{1}{n} \sum_{i=0}^{n-1} (\hat{y}^{(i)} - y^{(i)})^2$$

***root mean square error***

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (\hat{y}^{(i)} - y^{(i)})^2}$$

# Coeficiente de determinação

$$\bar{y} = \frac{1}{n} \sum_{i=0}^{n-1} y^{(i)}$$



$$SS_{res} = \sum_{i=0}^{n-1} (\hat{y}^{(i)} - y^{(i)})^2$$

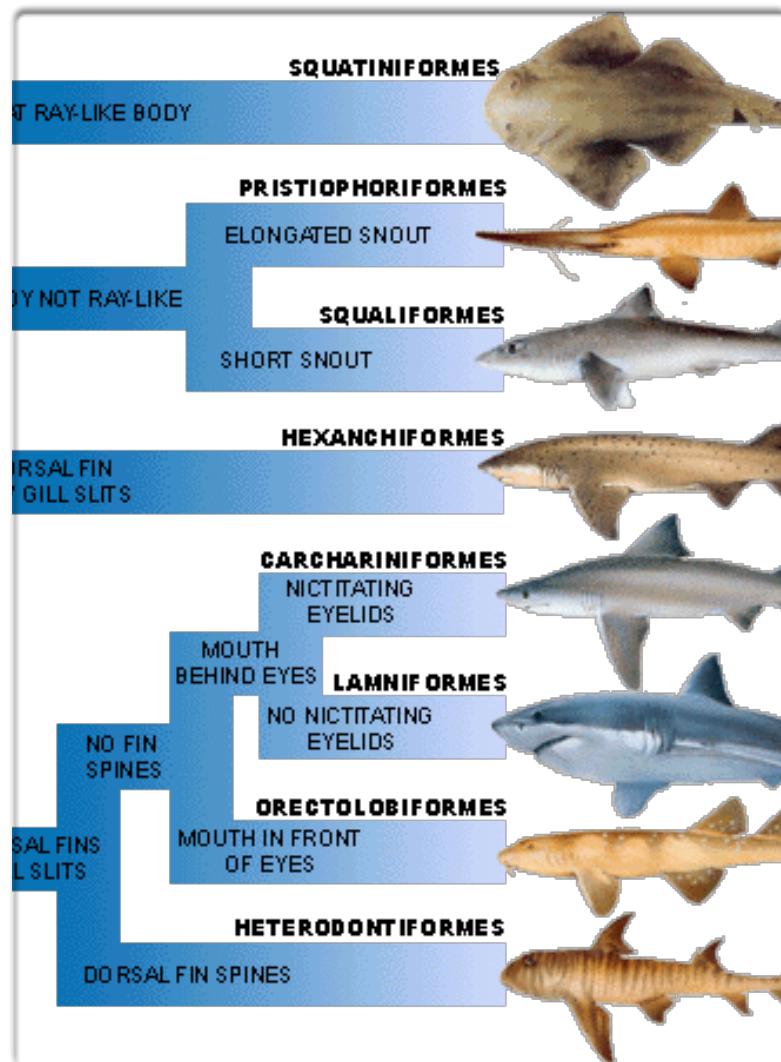
$$SS_{tot} = \sum_{i=0}^{n-1} (y^{(i)} - \bar{y})^2$$



**coefficient of  
determination**

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

# métricas



para classificação

## Predições

	<b>Gold</b>	<b>Pred</b>
<b>cliente0</b>	<b>0</b>	<b>0</b>
<b>cliente1</b>	<b>1</b>	<b>1</b>
<b>cliente2</b>	<b>0</b>	<b>0</b>
<b>cliente3</b>	<b>1</b>	<b>1</b>
<b>cliente4</b>	<b>1</b>	<b>0</b>
<b>cliente5</b>	<b>0</b>	<b>0</b>
<b>cliente6</b>	<b>0</b>	<b>0</b>
<b>cliente7</b>	<b>1</b>	<b>0</b>
<b>cliente8</b>	<b>1</b>	<b>1</b>
<b>cliente9</b>	<b>0</b>	<b>1</b>

Matriz de Confusão

		<b>Pred</b>	
		<b>0</b>	<b>1</b>
<b>Gold</b>	<b>0</b>	4	1
	<b>1</b>	2	3

## Predições

	<b>Gold</b>	<b>Pred</b>
<b>cliente0</b>	<b>0</b>	<b>0</b>
<b>cliente1</b>	<b>1</b>	<b>1</b>
<b>cliente2</b>	<b>0</b>	<b>0</b>
<b>cliente3</b>	<b>1</b>	<b>1</b>
<b>cliente4</b>	<b>1</b>	<b>0</b>
<b>cliente5</b>	<b>0</b>	<b>0</b>
<b>cliente6</b>	<b>0</b>	<b>0</b>
<b>cliente7</b>	<b>1</b>	<b>0</b>
<b>cliente8</b>	<b>1</b>	<b>1</b>
<b>cliente9</b>	<b>0</b>	<b>1</b>

Matriz de Confusão

		<b>Pred</b>	
		<b>0</b>	<b>1</b>
<b>Gold</b>	<b>0</b>	verdadeiro negativo	falso positivo
	<b>1</b>	falso negativo	verdadeiro positivo

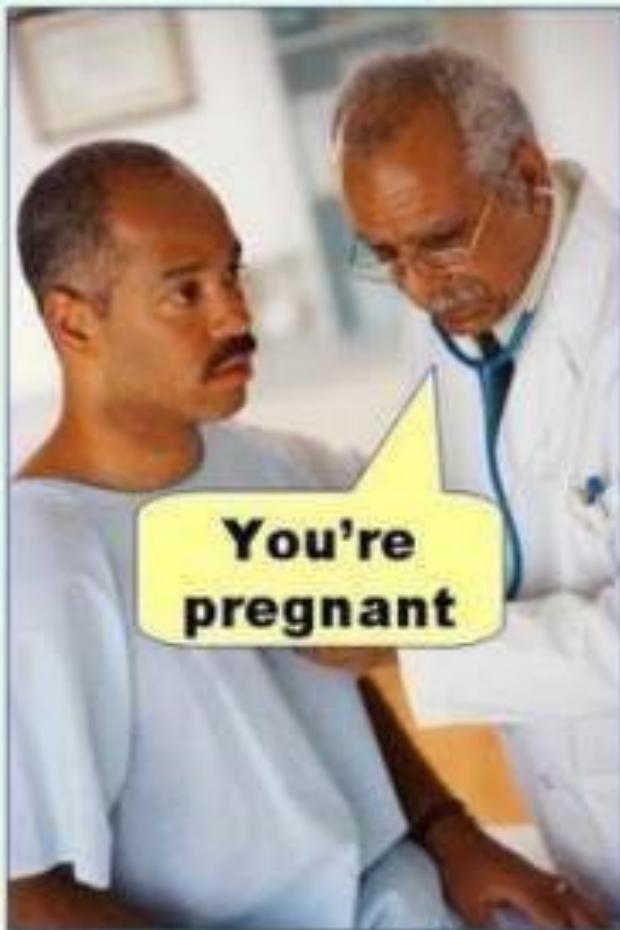
## Predições

	Gold	Pred
cliente0	0	0
cliente1	1	1
cliente2	0	0
cliente3	1	1
cliente4	1	0
cliente5	0	0
cliente6	0	0
cliente7	1	0
cliente8	1	1
cliente9	0	1

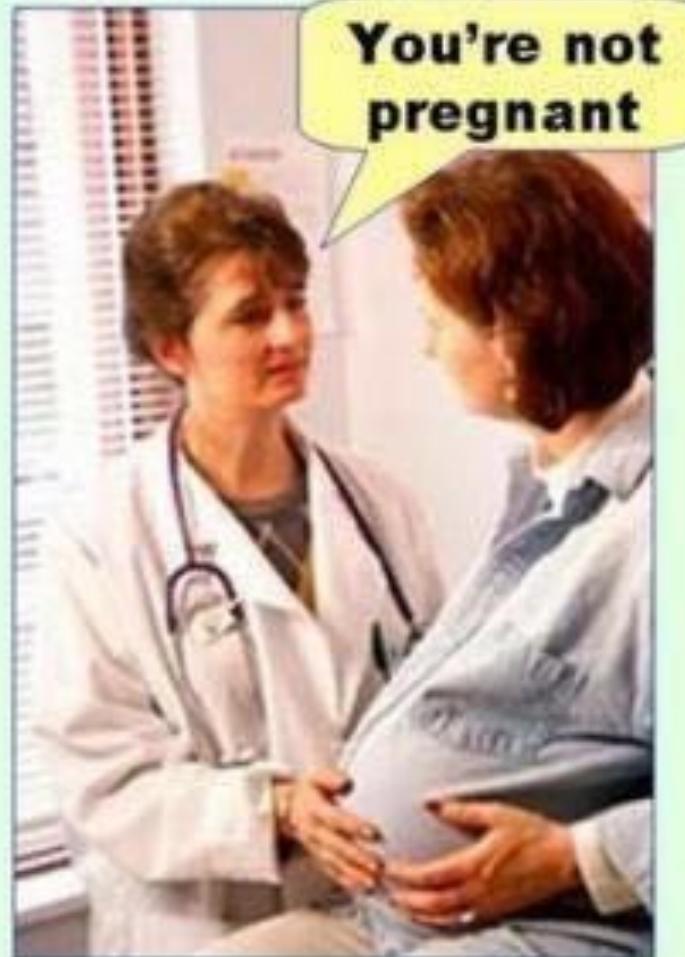
Matriz de Confusão

		Pred	
		0	1
Gold	0	4	1
	1	2	3

**Type I error**  
(false positive)



**Type II error**  
(false negative)



	<b>Gold</b>	<b>Pred</b>
<b>cliente0</b>	<b>0</b>	<b>0</b>
<b>cliente1</b>	<b>1</b>	<b>1</b>
<b>cliente2</b>	<b>0</b>	<b>0</b>
<b>cliente3</b>	<b>1</b>	<b>1</b>
<b>cliente4</b>	<b>1</b>	<b>0</b>
<b>cliente5</b>	<b>0</b>	<b>0</b>
<b>cliente6</b>	<b>0</b>	<b>0</b>
<b>cliente7</b>	<b>1</b>	<b>0</b>
<b>cliente8</b>	<b>1</b>	<b>1</b>
<b>cliente9</b>	<b>0</b>	<b>1</b>

$$\text{acurácia} = \frac{\text{labels corretos}}{\text{todas obs}}$$

$$\text{precisão} = \frac{\text{Verdadeiros positivos}}{\text{Verdadeiros positivos} + \text{Falsos Positivos}}$$

$$\text{recall} = \frac{\text{Verdadeiros positivos}}{\text{Verdadeiros positivos} + \text{Falsos Negativos}}$$

$$\text{f1-score} = \frac{2}{\frac{1}{\text{precisão}} + \frac{1}{\text{recall}}}$$

Esse accuracy é alto, vários '0' foram previstos como '0'. Mas a precisão é zero. De fato, não há verdadeiro positivos, ou seja, não há nenhum que '1' que foi corretamente previsto.

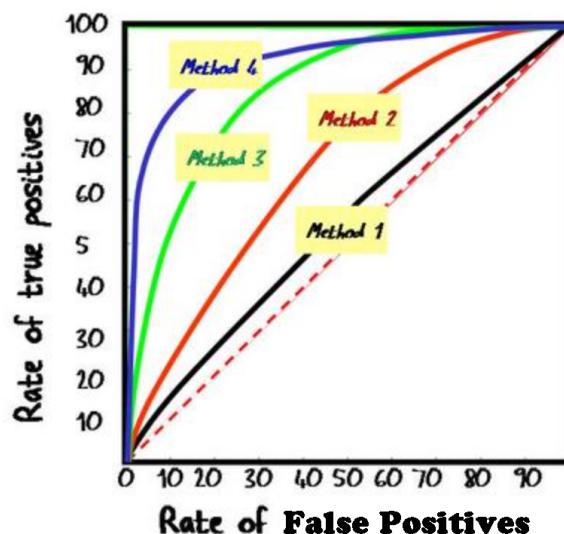
Isso acontece porque a quantidade de '0' é muito alta, maior que a de '1'. Assim, existe uma boa chance dos acertos de '0' terem sido corretos somente porque a sua quantidade no label dos dados de teste é alta, e não porque o algoritmo é bom.

Dizemos que a distribuição é desbalanceada (muitos 0 e poucos 1). E nesses casos, accuracy não é uma boa métrica.

Importante: Muitas vezes a meta é identificar 1's.  
Identificar 0's é um objetivo menor.

- precision, recall e F1 dependem de um limiar de decisão...
- ... então, qual limiar escolher?
- Existe uma métrica que independeria de um limiar?

## ROC CURVE EXAMPLES



- The best classification has the largest area under the curve.
- Too sensitive to errors in the "gold standard" classification.

$$FPR = \frac{\text{Falsos positivos}}{\text{Verdadeiros negativos} + \text{Falsos positivos}}$$

↓

**condição negativa  
gold = 0**

$$TPR = \frac{\text{Verdadeiros positivos}}{\text{Verdadeiros positivos} + \text{Falsos negativos}}$$

↓

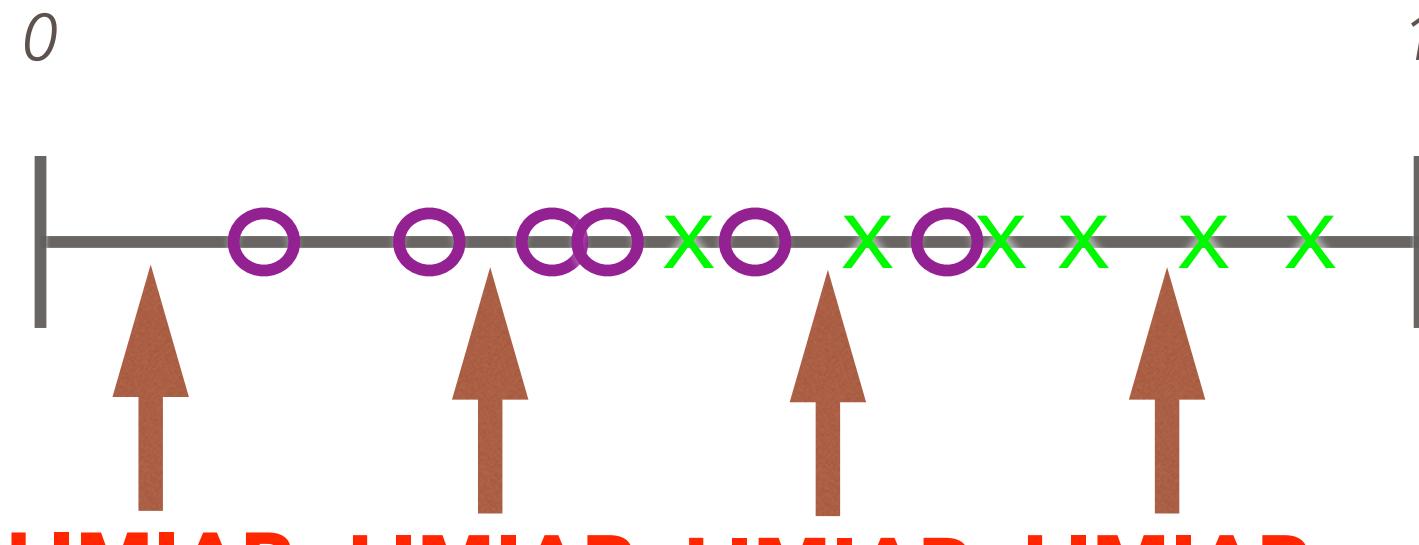
**condição positiva  
gold = 1**

$$\mathbf{FPR} = \frac{\text{Falsos positivos}}{\text{Verdadeiros negativos} + \text{Falsos positivos}}$$

○ condição negativa

$$\mathbf{TPR} = \frac{\text{Verdadeiros positivos}}{\text{Verdadeiros positivos} + \text{Falsos negativos}}$$

X condição positiva



**LIMIAR LIMIAR LIMIAR LIMIAR**

$$\mathbf{TPR} = 6/6$$

$$\mathbf{FPR} = 6/6$$

$$\mathbf{TPR} = 6/6$$

$$\mathbf{FPR} = 4/6$$

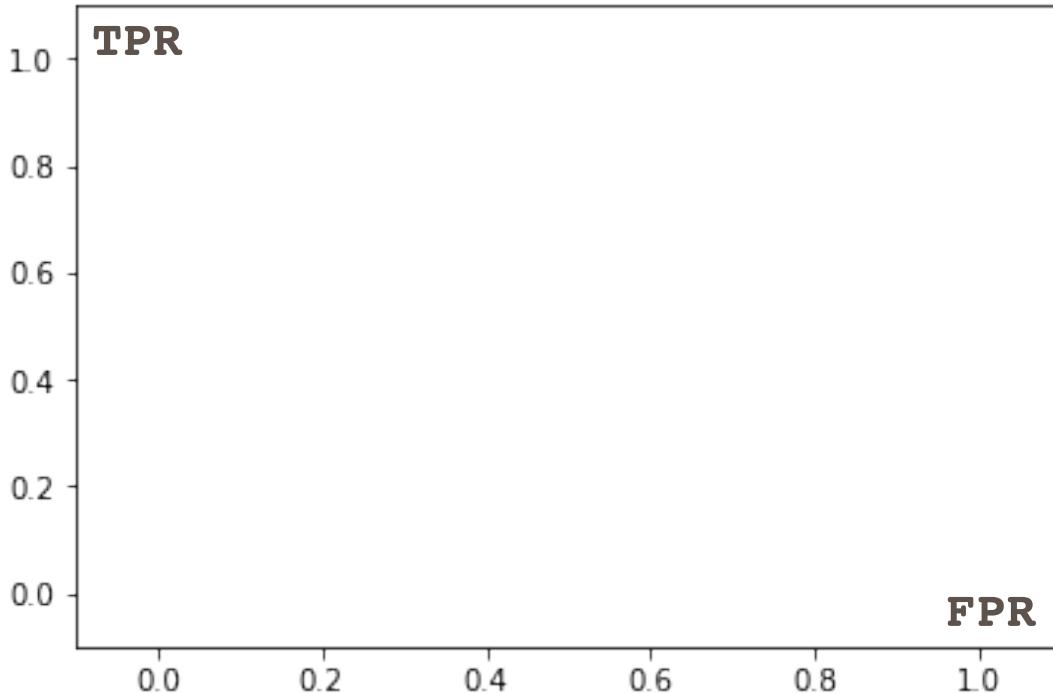
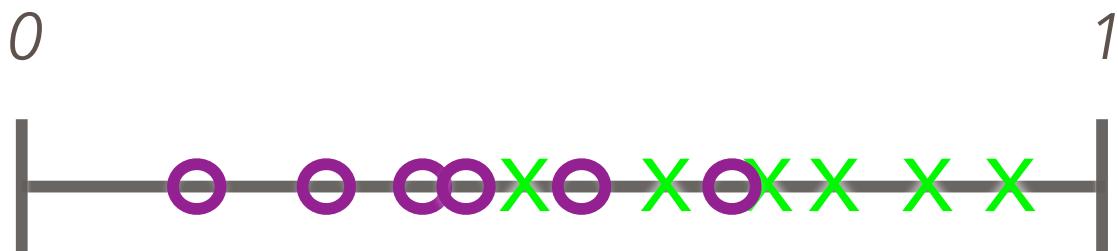
$$\mathbf{TPR} = 5/6$$

$$\mathbf{FPR} = 1/6$$

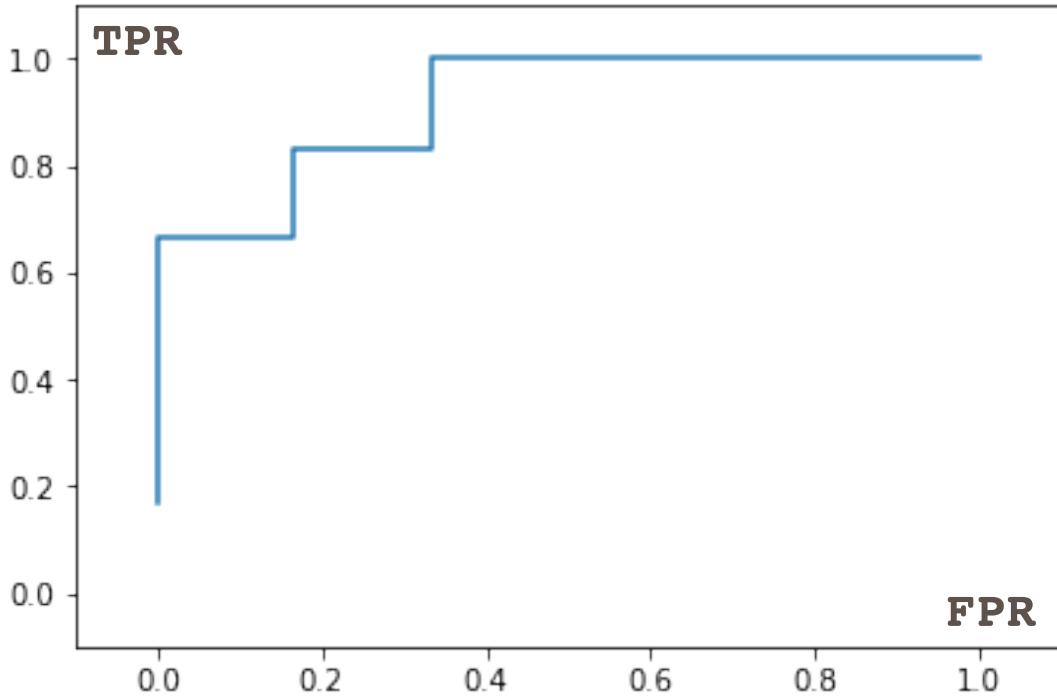
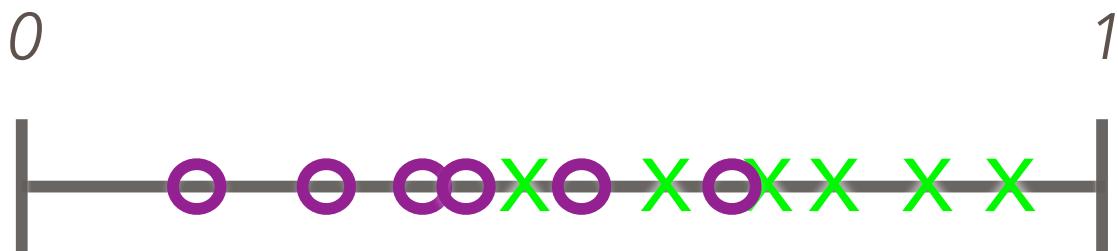
$$\mathbf{TPR} = 2/6$$

$$\mathbf{FPR} = 0/6$$

	Prob[label = 1]	label
1	0.90	1.0
2	0.80	1.0
3	0.75	1.0
4	0.70	1.0
5	0.69	0.0
6	0.60	1.0
7	0.50	0.0
8	0.45	1.0
9	0.40	0.0
10	0.38	0.0
11	0.28	0.0
12	0.20	0.0



	Prob[label = 1]	label
1	0.90	1.0
2	0.80	1.0
3	0.75	1.0
4	0.70	1.0
5	0.69	0.0
6	0.60	1.0
7	0.50	0.0
8	0.45	1.0
9	0.40	0.0
10	0.38	0.0
11	0.28	0.0
12	0.20	0.0



- Qual é o AUC?



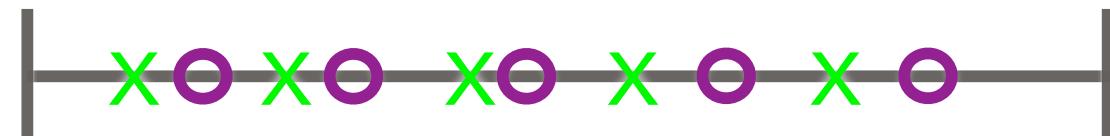
AUC =



AUC =



AUC =



AUC =

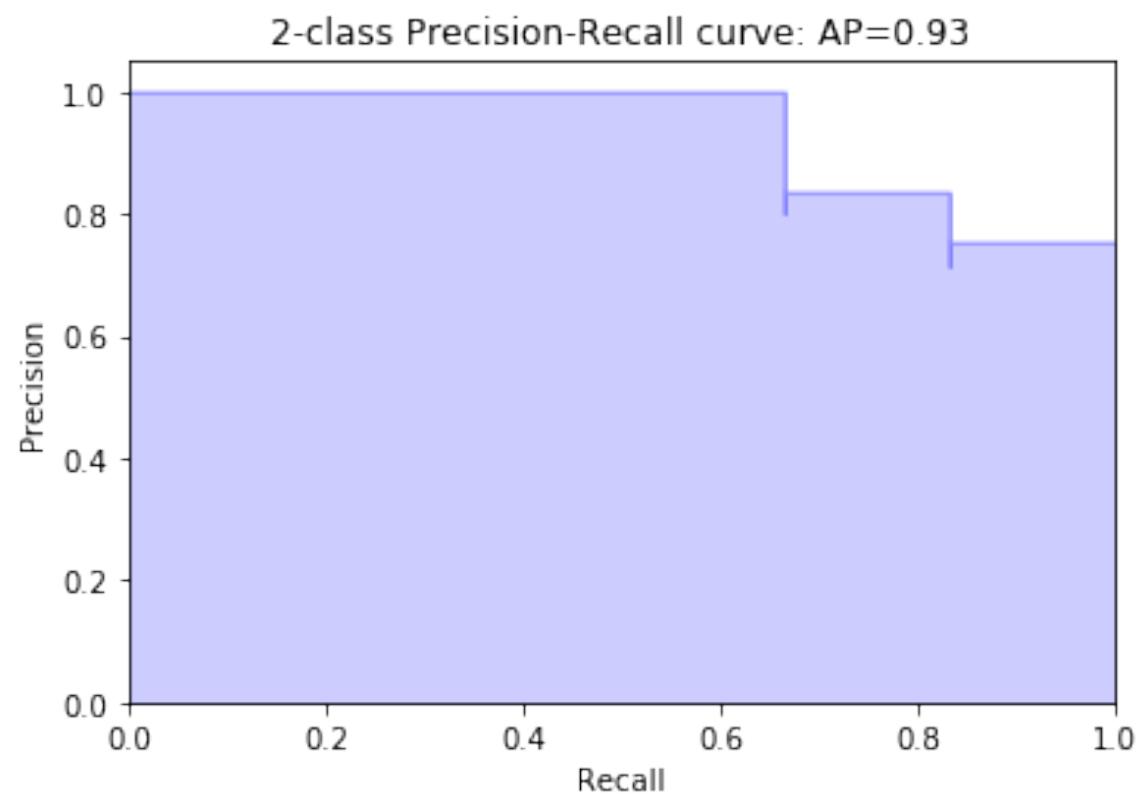


AUC =

○ condição negativa  
X condição positiva

	Prob[label = 1]	label
1	0.90	1.0
2	0.80	1.0
3	0.75	1.0
4	0.70	1.0
5	0.69	0.0
6	0.60	1.0
7	0.50	0.0
8	0.45	1.0
9	0.40	0.0
10	0.38	0.0
11	0.28	0.0
12	0.20	0.0

- precision no eixo y
- recall no eixo x
- para diferentes limiares



# terminologia

- loss function
- cost function
- objective function
- Qual a diferença?
  - Loss – uma única observação
  - Cost – todos as observações
  - Objective – uma função a otimizar durante o treinamento
- ref  
<https://stats.stackexchange.com/questions/179026/objective-function-cost-function-loss-function-are-they-the-same-thing>

- 2 médicos, Paulo e Georgia prestando o exame do conselho de medicina.
- 2 questões, Q1 e Q2. Cada questão com uma descrição de sintomas com três alternativas de doença: doença A, doença B e doença C.
- O gabarito é  
 Q1: doença C  
 Q2: doença B
- Paulo lê o enunciado e pensa nas seguintes probabilidades:
 

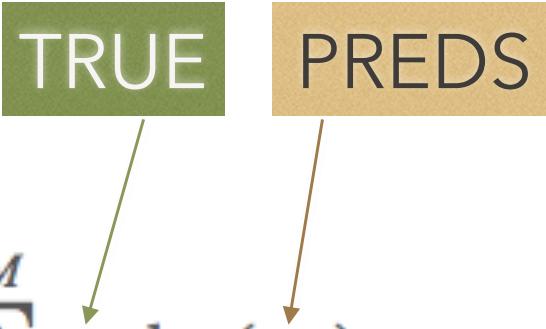
	Q1	Q2
doença A	20%	30%
doença B	30%	40%
doença C	50%	30%

 =====> Paulo acerta 2/2
- Georgia lê o enunciado e pensa nas seguintes probabilidades:
 

	Q1	Q2
doença A	0%	5%
doença B	10%	95%
doença C	90%	0%

 =====> Georgia acerta 2/2
- Ambos passam no exame, mas... QUEM É O “MELHOR” MÉDICO?

## **cross-entropy ou log-loss**

$$\text{logloss} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{i,j} \log(p_{i,j})$$


The diagram illustrates the components of the logloss formula. At the top, there are two boxes: 'TRUE' in a green box and 'PREDS' in an orange box. Two arrows point downwards from these boxes to the corresponding terms in the equation below. The first arrow points to the  $y_{i,j}$  term, and the second arrow points to the  $p_{i,j}$  term.

*<https://www.kaggle.com/wiki/LogLoss>*

# cross-entropy ou log-loss

$$\text{logloss} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{i,j} \log(p_{i,j})$$

PREDS

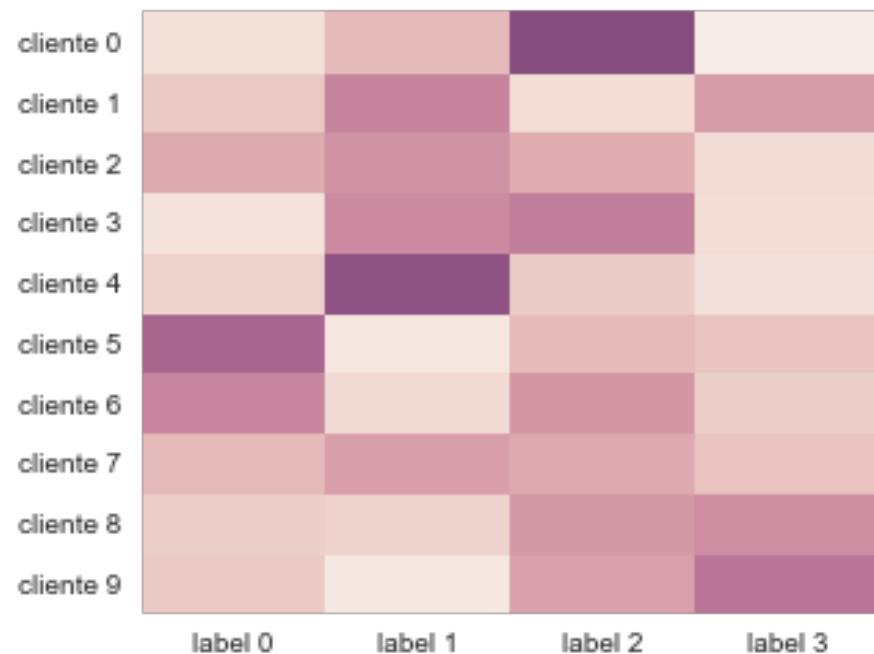
	label 0	label 1	label 2	label 3
<b>cliente 0</b>	0,069	0,213	0,698	0,020
<b>cliente 1</b>	0,164	0,426	0,081	0,329
<b>cliente 2</b>	0,278	0,370	0,269	0,083
<b>cliente 3</b>	0,055	0,405	0,460	0,080
<b>cliente 4</b>	0,124	0,666	0,145	0,066
<b>cliente 5</b>	0,569	0,045	0,214	0,172
<b>cliente 6</b>	0,421	0,091	0,353	0,135
<b>cliente 7</b>	0,215	0,323	0,279	0,182
<b>cliente 8</b>	0,139	0,123	0,347	0,390
<b>cliente 9</b>	0,155	0,036	0,311	0,498

TRUE

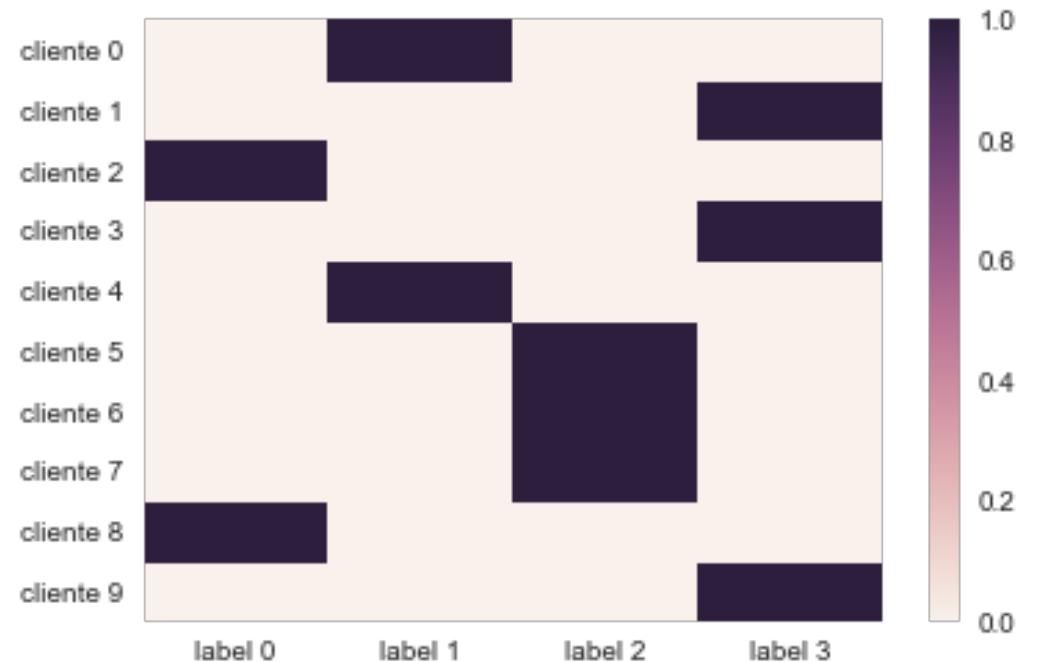
	label 0	label 1	label 2	label 3
<b>cliente 0</b>	0	1	0	0
<b>cliente 1</b>	0	0	0	1
<b>cliente 2</b>	1	0	0	0
<b>cliente 3</b>	0	0	0	1
<b>cliente 4</b>	0	1	0	0
<b>cliente 5</b>	0	0	1	0
<b>cliente 6</b>	0	0	1	0
<b>cliente 7</b>	0	0	1	0
<b>cliente 8</b>	1	0	0	0
<b>cliente 9</b>	0	0	0	1

# cross-entropy ou log-loss

PREDS

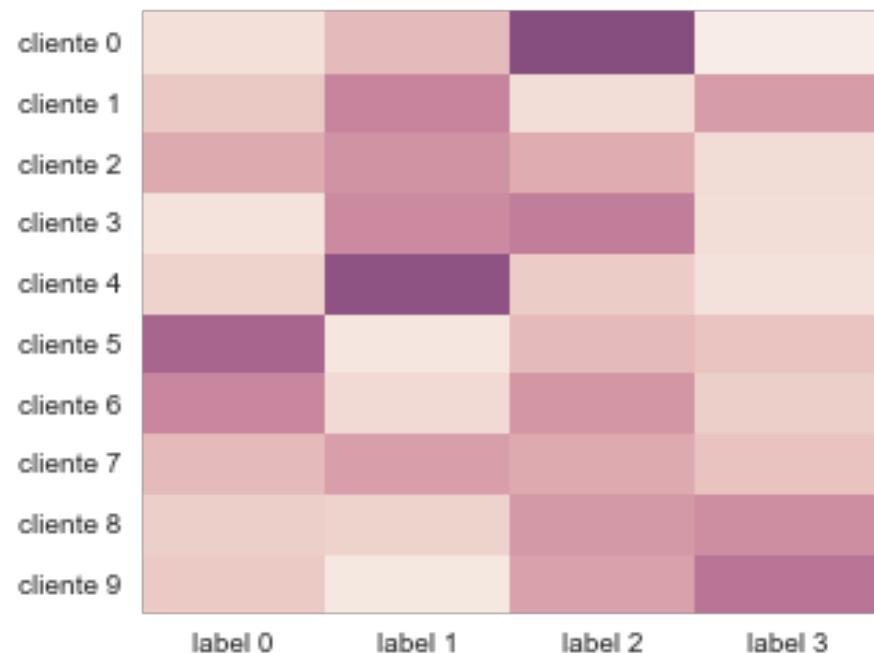


TRUE

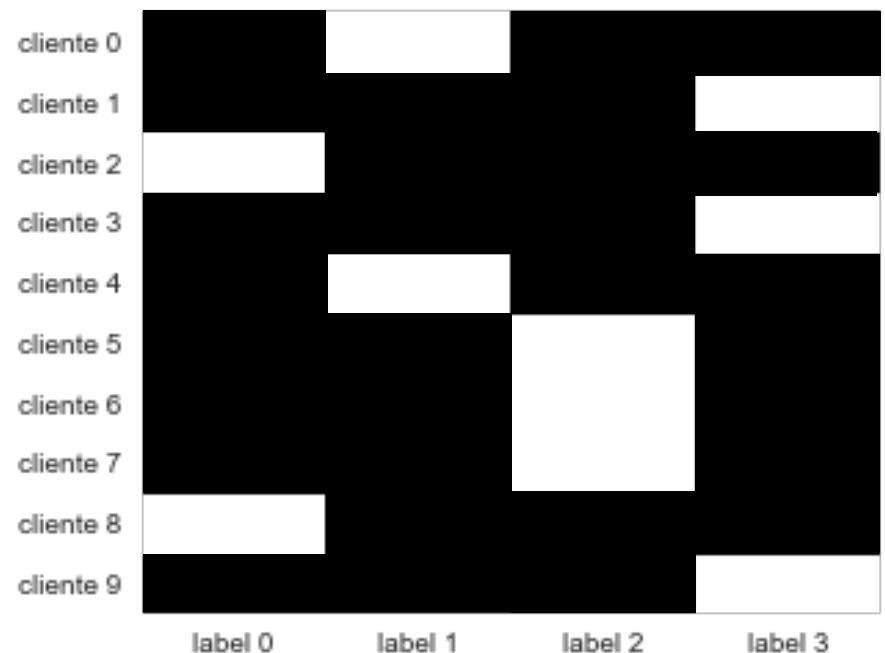


# cross-entropy ou log-loss

PREDS



TRUE



$$\text{logloss} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{i,j} \log(p_{i,j})$$

# cross-entropy ou log-loss



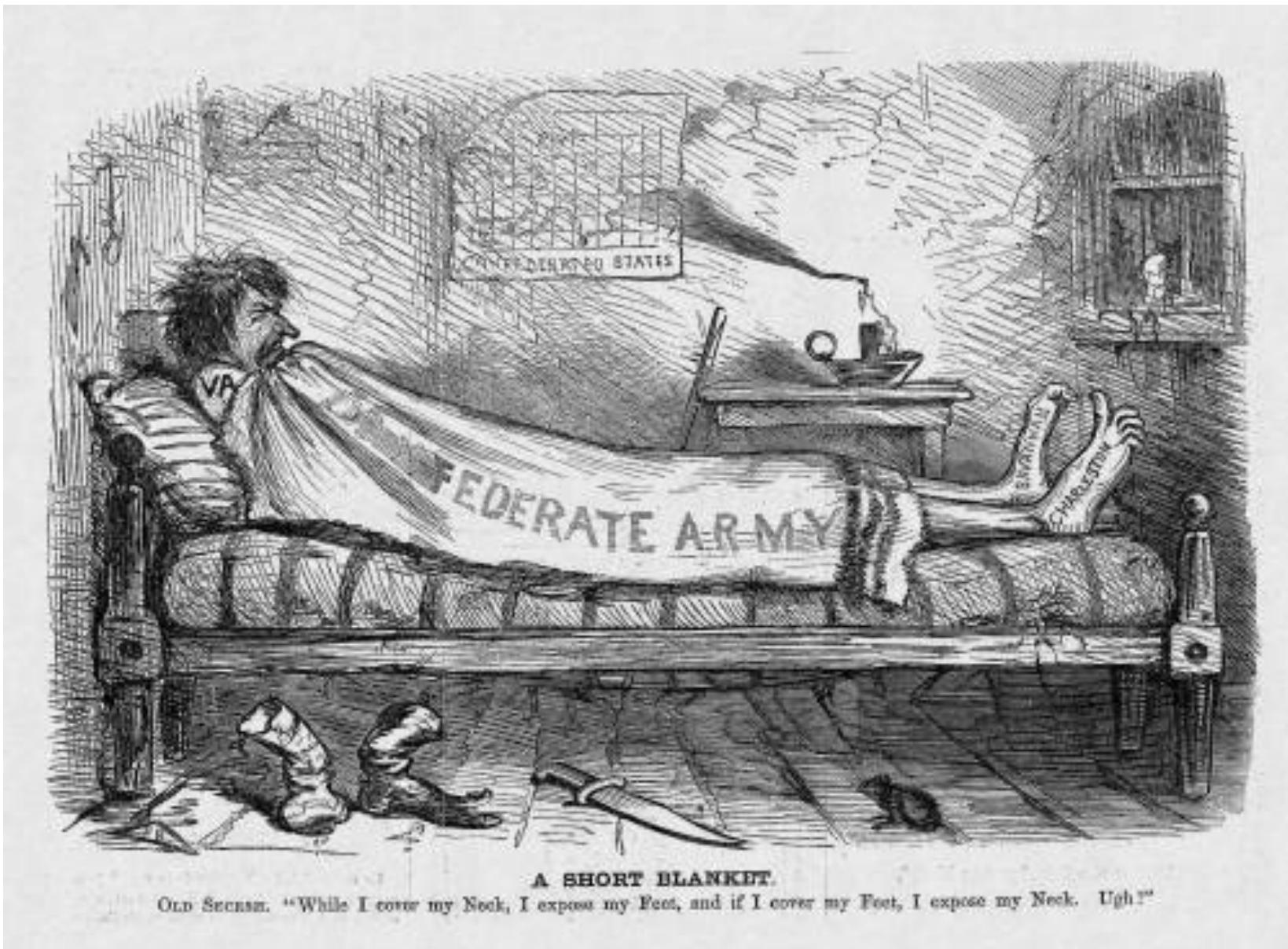
$$\text{logloss} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{ij} \log(p_{ij})$$

# DIVIDINDO OS DADOS



# dilema bias variance

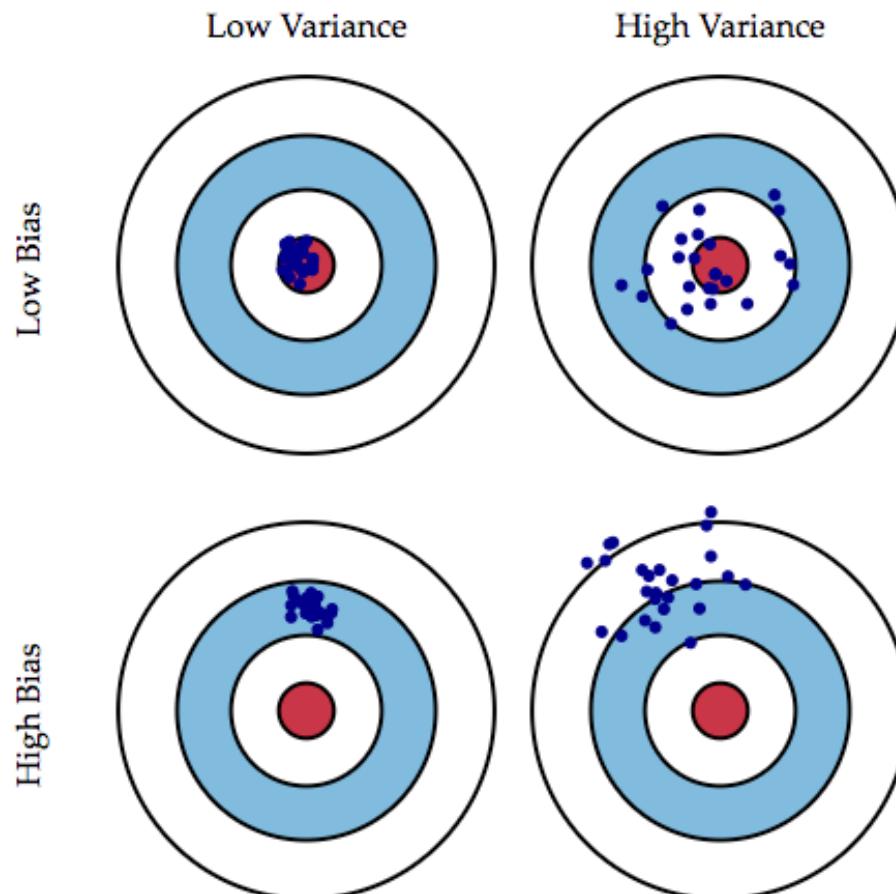
## A Short Blanket



A SHORT BLANKET.

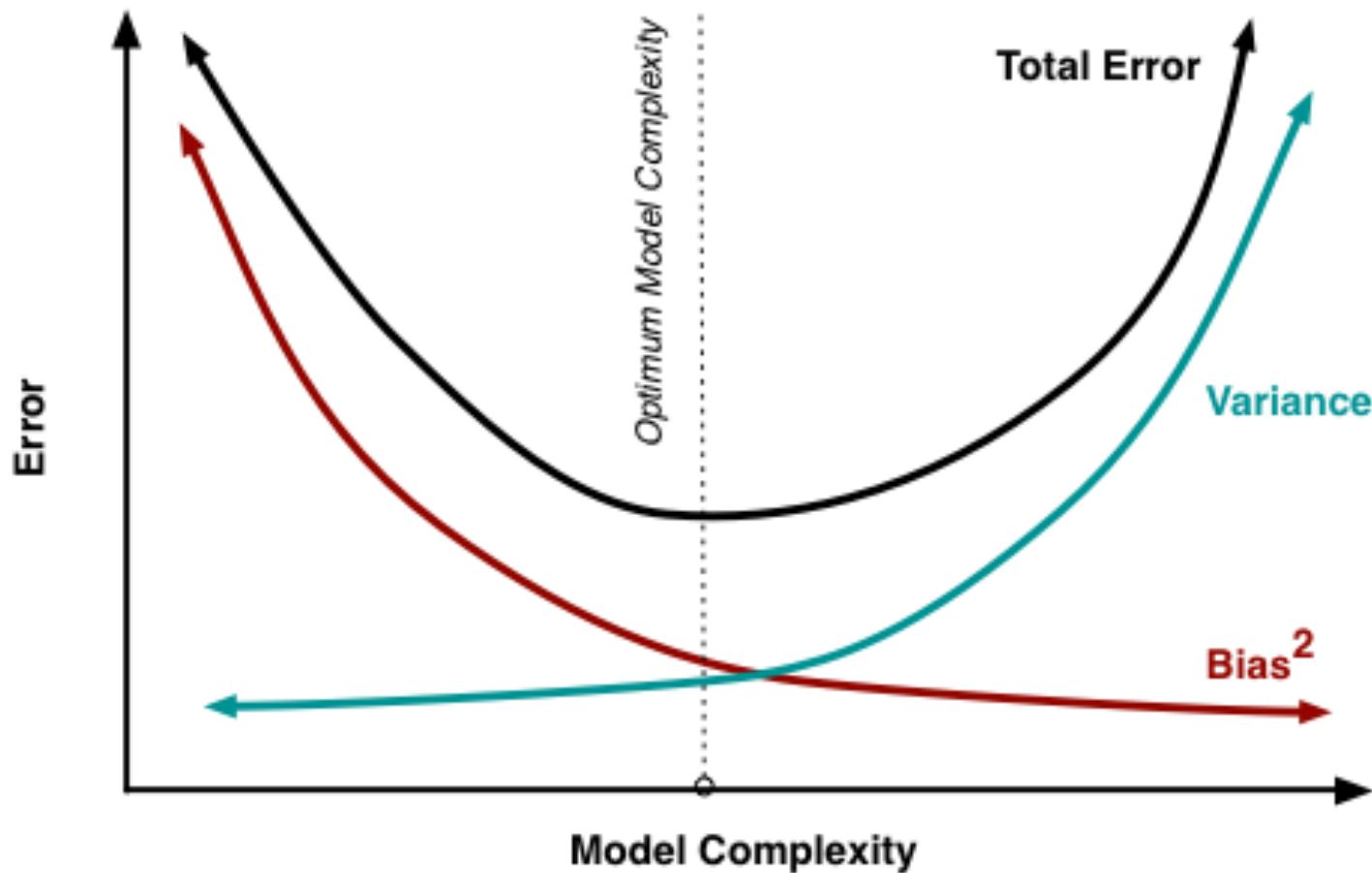
OLD SECESH. "While I cover my Neck, I expose my Feet, and if I cover my Feet, I expose my Neck. Ugh!"

# BIAS VARIANCE DILEMMA



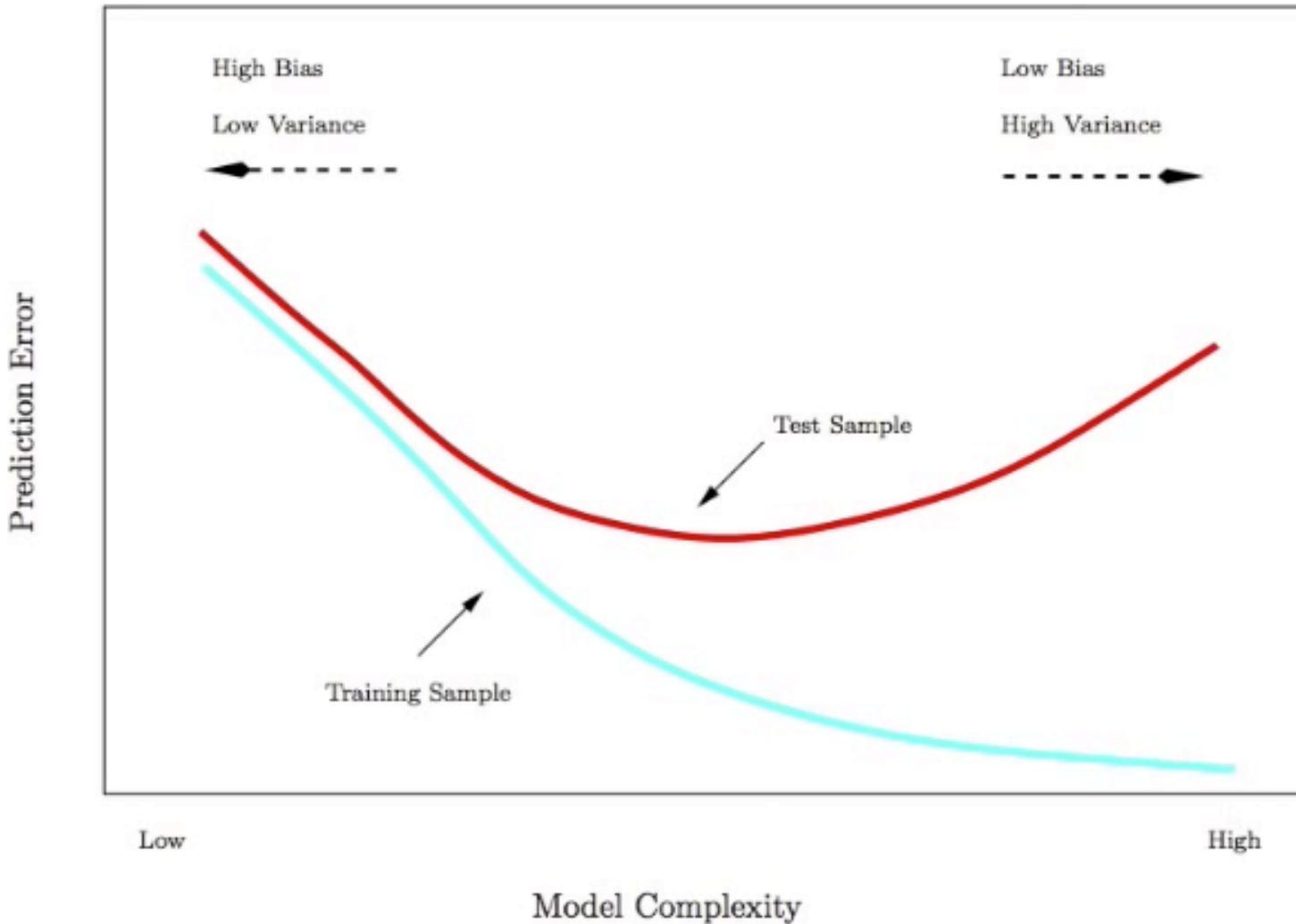
<http://scott.fortmann-roe.com/docs/BiasVariance.html>

# BIAS VARIANCE DILEMMA



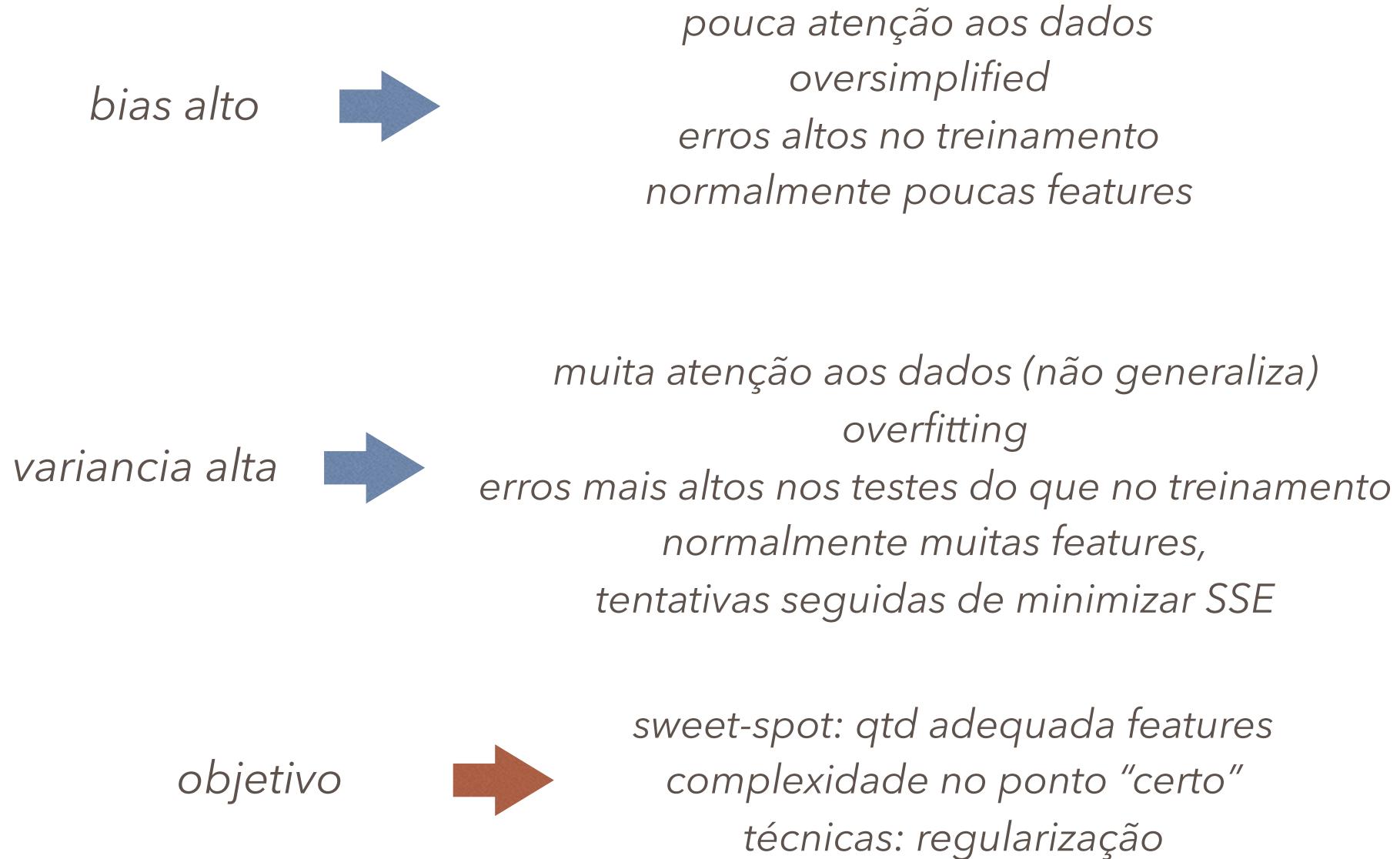
<http://scott.fortmann-roe.com/docs/BiasVariance.html>

# BIAS VARIANCE DILEMMA



<https://lagunita.stanford.edu/courses/HumanitiesandScience/StatLearning/Winter2015/info>

# BIAS VARIANCE DILEMMA



# Validação cruzada

- Holdout ou train-test-split
- K-fold
- Stratified K-fold
- mais opções em:

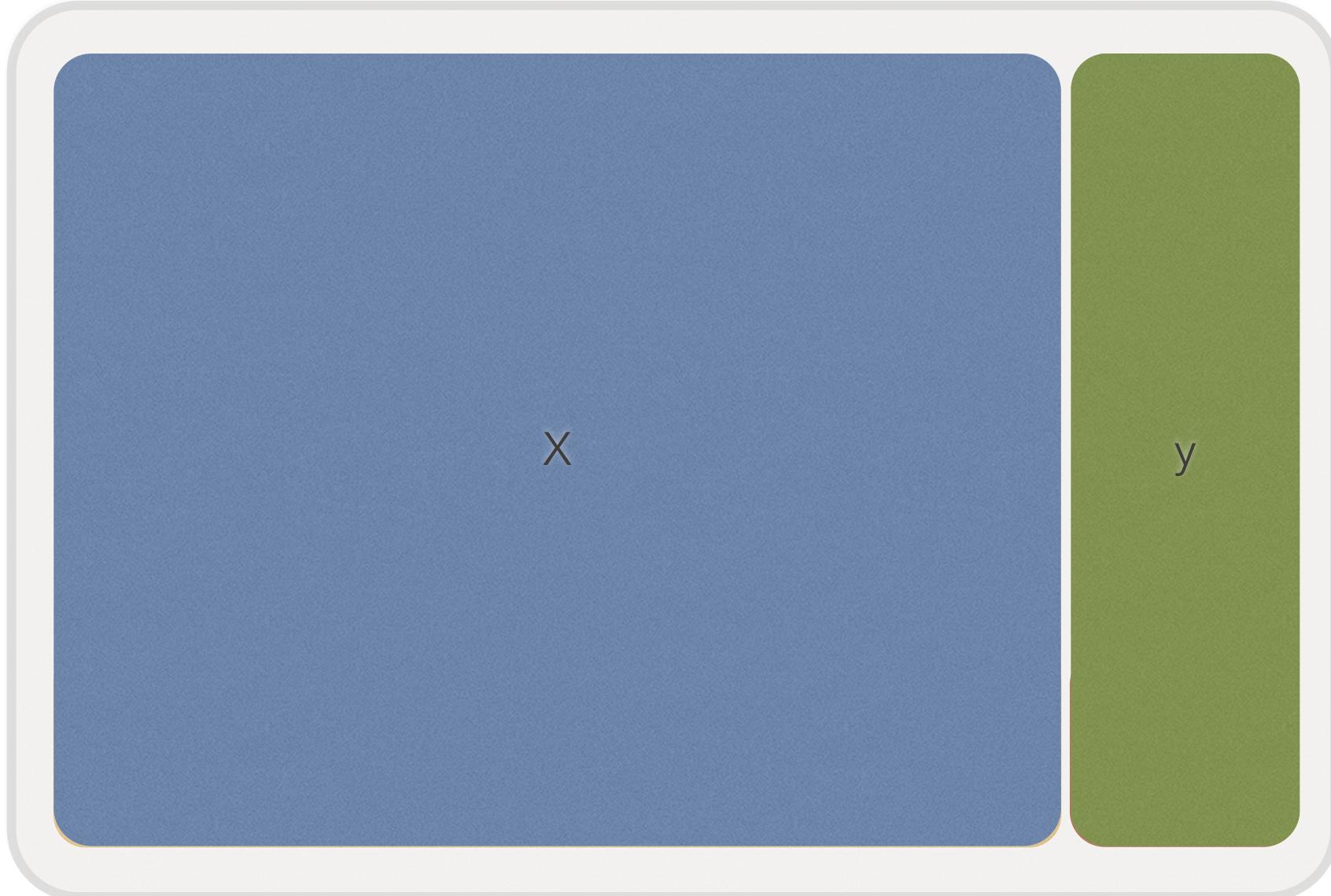
[http://scikit-learn.org/stable/modules/cross\\_validation.html#cross-validation](http://scikit-learn.org/stable/modules/cross_validation.html#cross-validation)

# Uma massa unica

X

y

# Uma massa unica



# SEPARANDO OS DADOS

X\_train

y\_train

X\_valid

y\_valid

# SEPARANDO OS DADOS

X\_train

y\_train

X\_valid

y\_valid

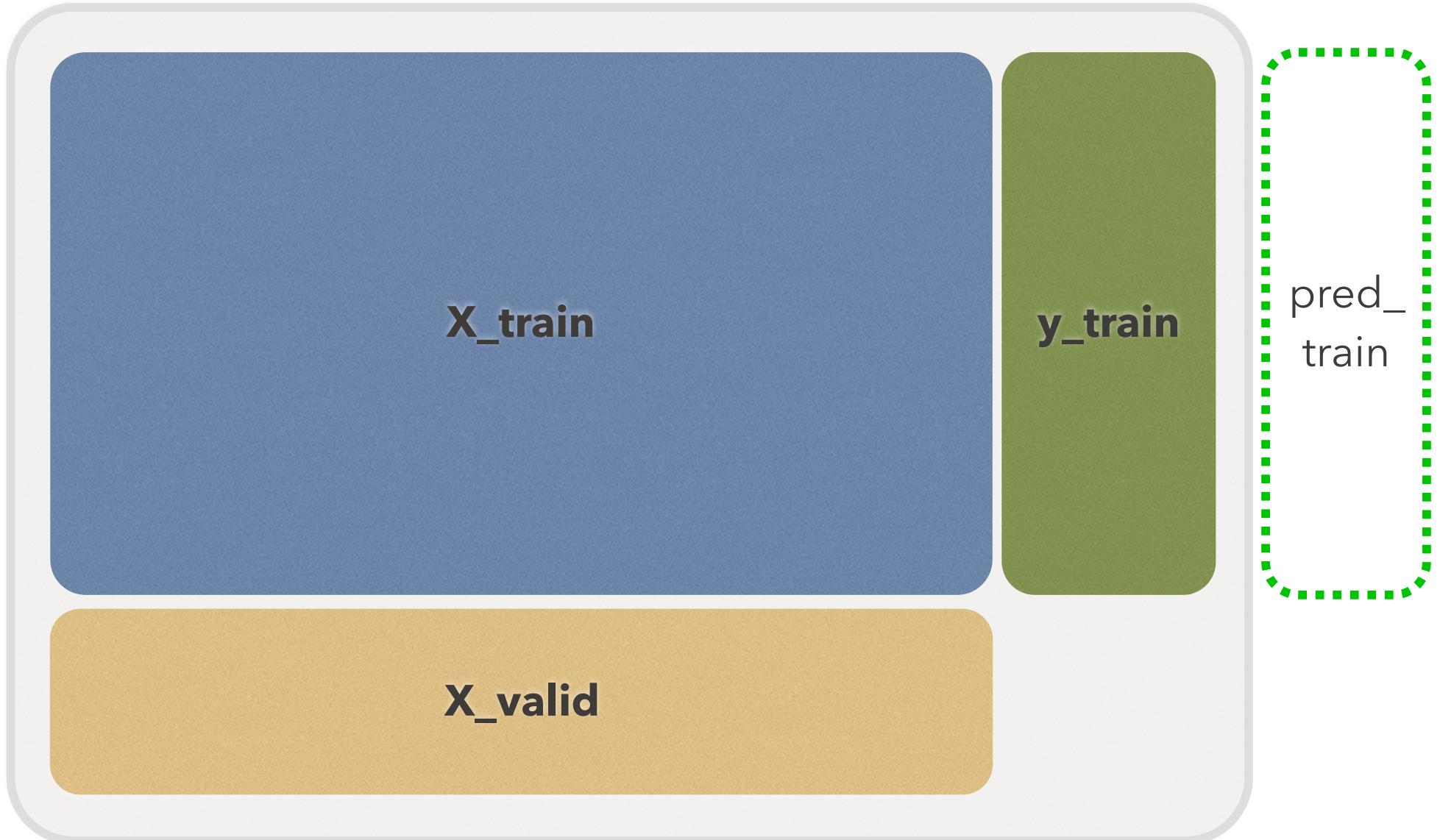
# SEPARANDO OS DADOS

X\_train

y\_train

X\_valid

# SEPARANDO OS DADOS



# SEPARANDO OS DADOS

X\_train

y\_train

X\_valid



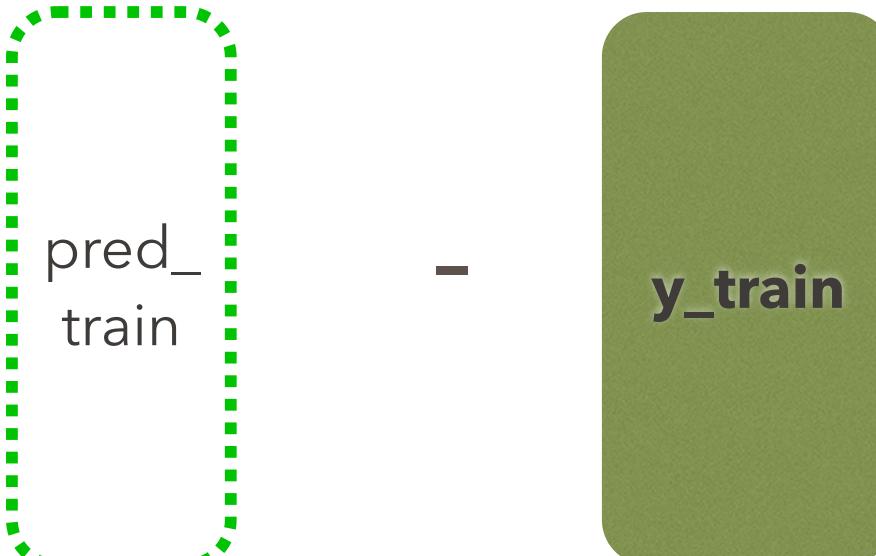
## validando o modelo...

*Objetivo: quanto menor o valor, melhor o modelo*

$$\text{mean}(\text{pred} - \text{y_valid})^2$$

# bias variance

**HIGH BIAS:** erro **alto** aqui....

$$\text{mean}(\text{pred\_train} - \text{y\_train})^2$$


# bias variance

**LOW BIAS:** erro **baixo** aqui....

$$\text{mean}(\text{pred\_train} - \text{y\_train})^2$$

**HIGH VARIANCE:** erro **alto** aqui....

$$\text{mean}(\text{pred} - \text{y\_valid})^2$$

## **Como dividir?**

- Sequencial
- Aleatorio
- Estratificado
- Leave one out / p out
- Bootstrap (amostragem com reposição)

# K Fold



# EM VEZ DE...

X\_train

y\_train

X\_valid

y\_valid

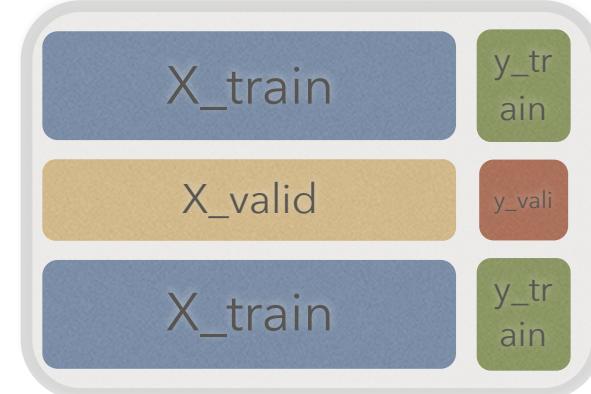
# KFOLD: k splits

para  $k = 3$

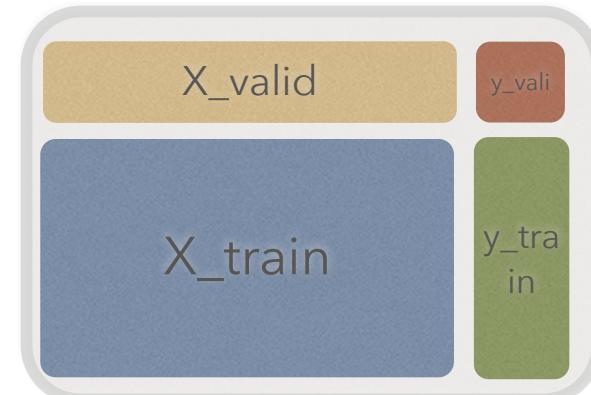
*Fold # 1*



*Fold # 2*



*Fold # 3*



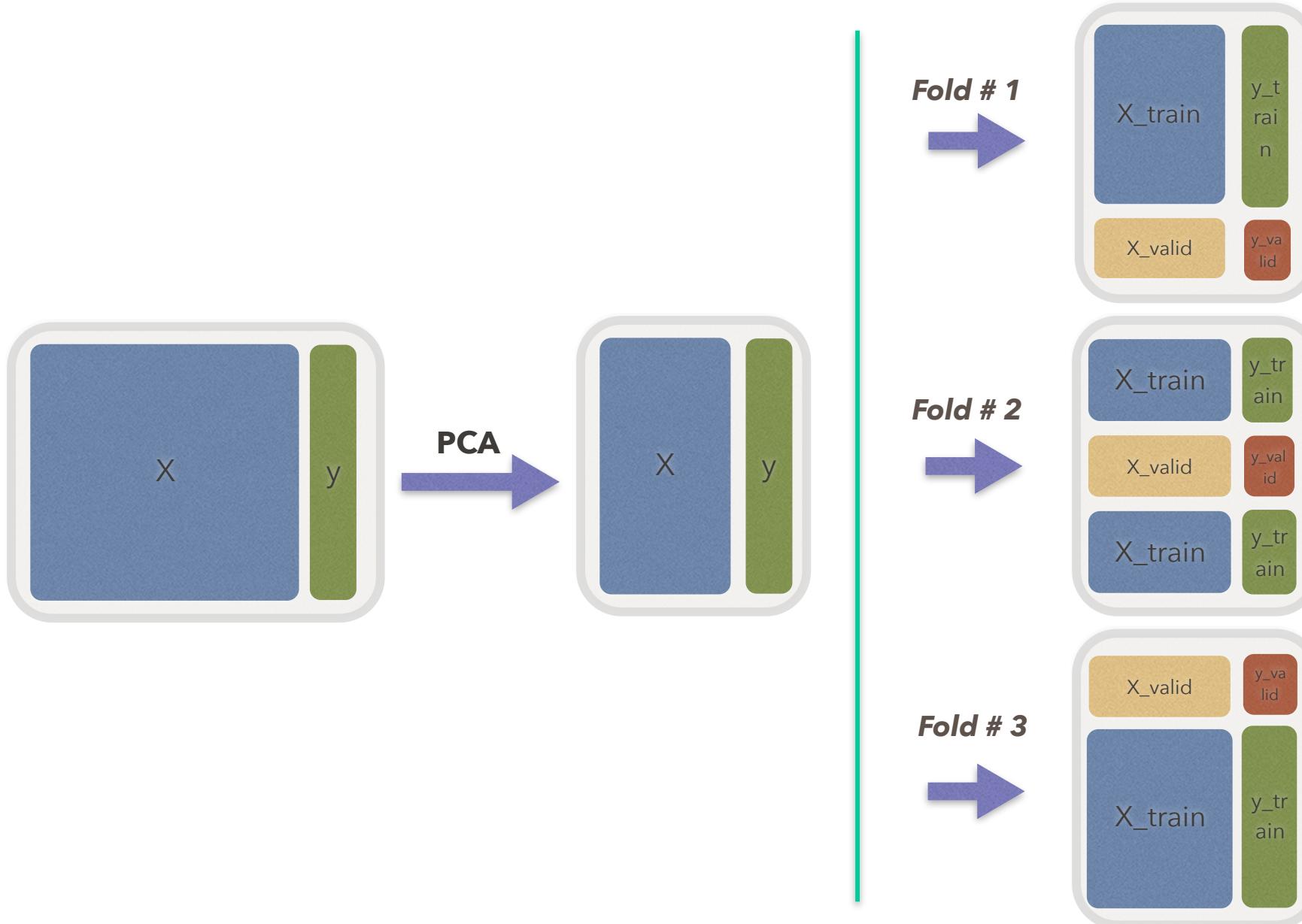
# teste e validação

- os termos teste e validação as vezes utilizados de maneira intercambiável.
  - validação: partes da massa de treinamento, com labels conhecidos e utilizados para tuning do algoritmo
  - teste: massa apartada para a prova final de desempenho
- validação  $\approx$  simulados  
teste  $\approx$  vestibular

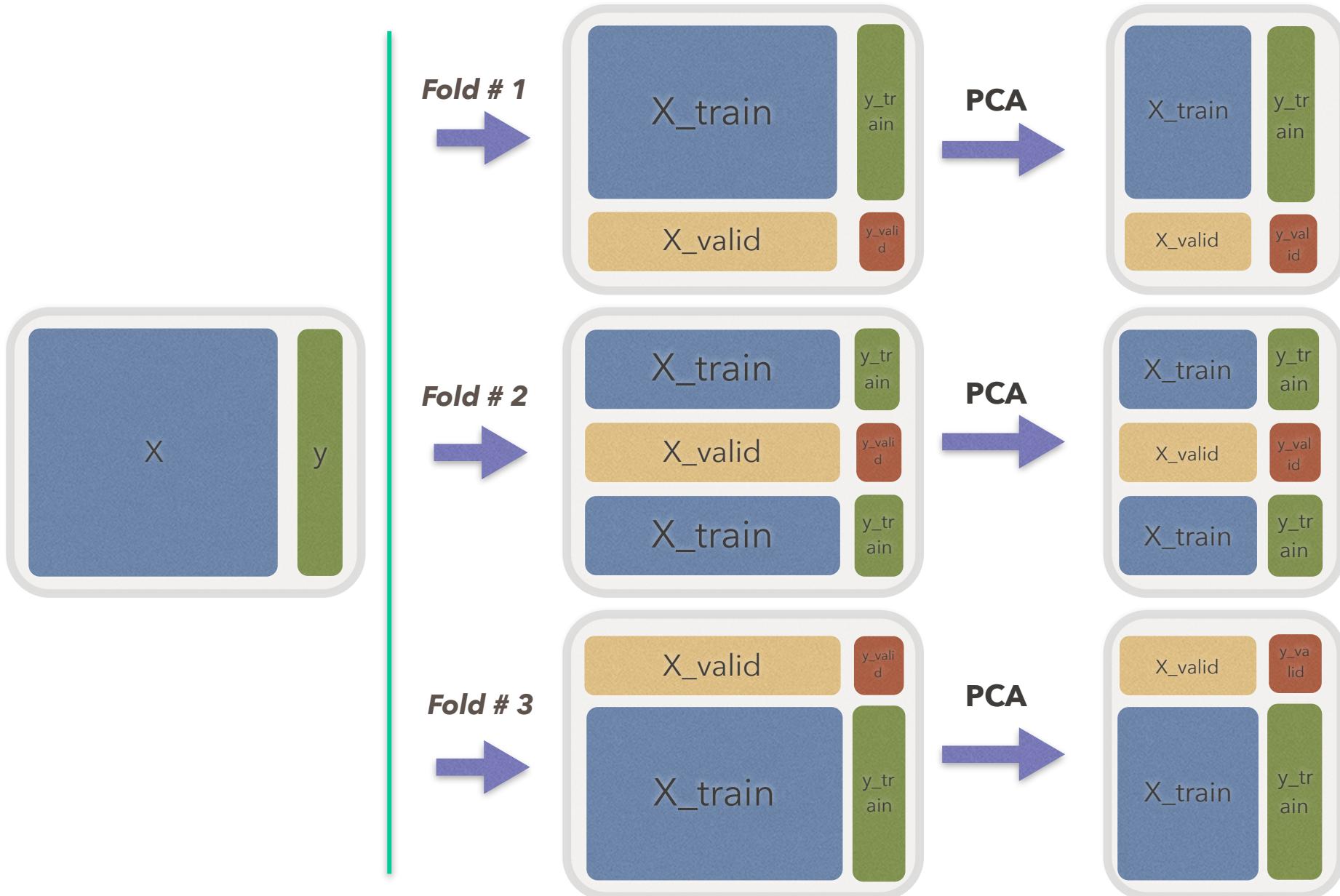
## Uma questão sutil

- As transformações nos folds de treinamento:
  - NÃO podem observar os labels da massa de validação
  - NÃO podem observar as features da massa de validação
- A exceção das transformações que independem dos valores.  
Ex: imputação com valor fixo,  $\log(x+1)$
- As demais transformações só podem aprender com o conteúdo dos folds de treinamento, como se o fold de validação não existisse.
- No sklearn, GridSearchCV ou RandomizedSearchCV com pipelines implementam essas precauções de maneira transparente.

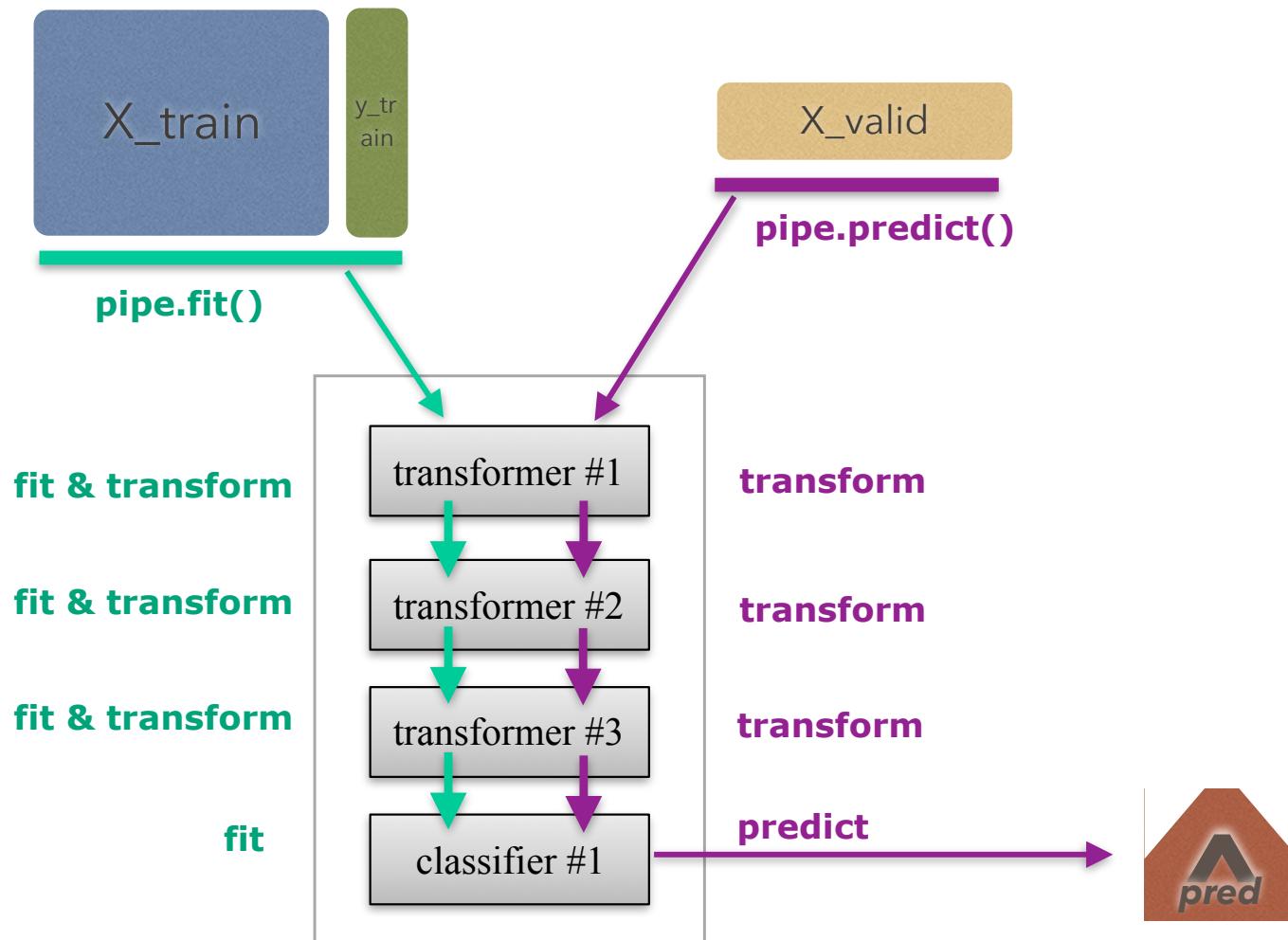
# PCA no k-fold: onde está o erro?



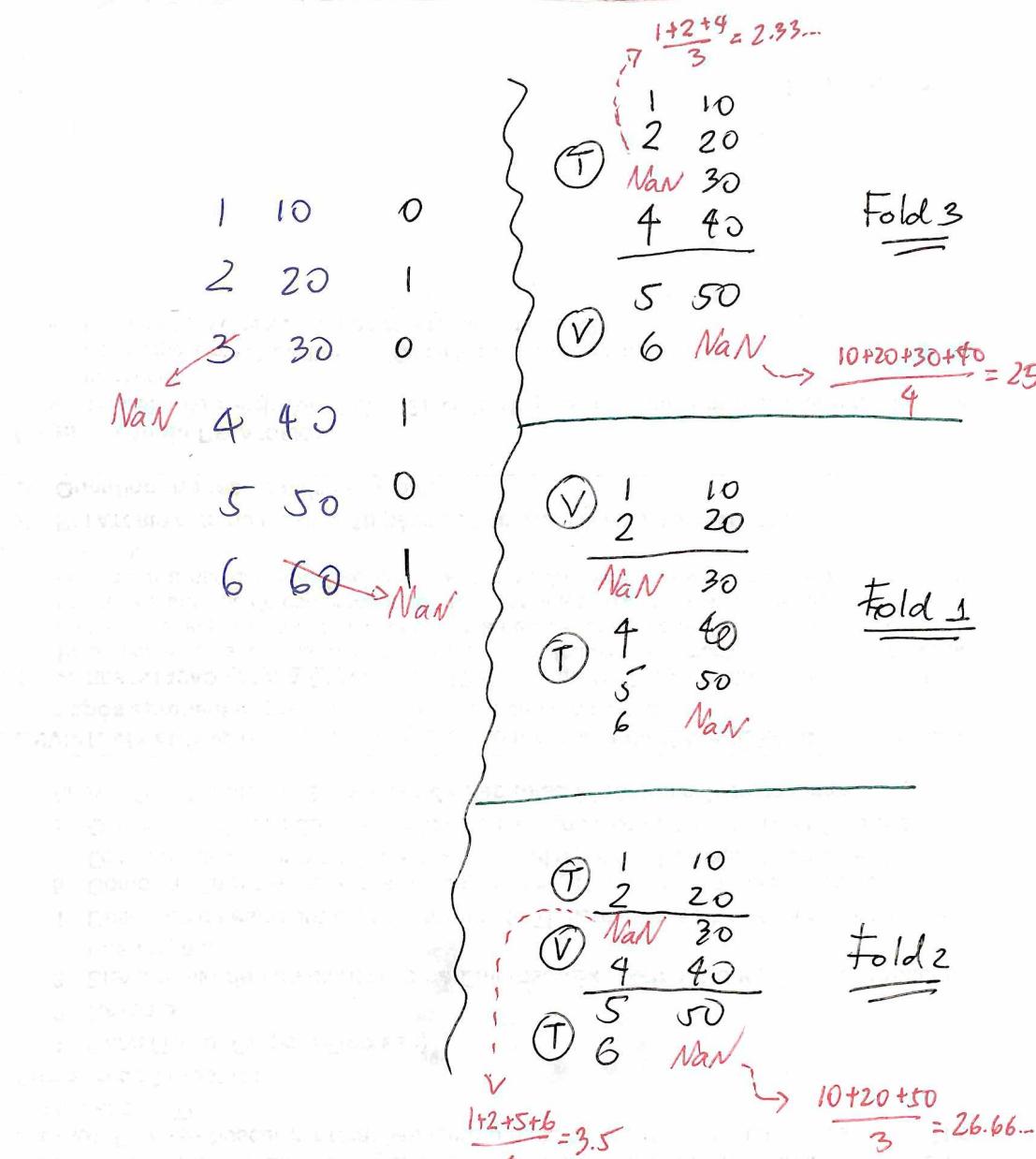
# PCA no k-fold



# Fit e predict no pipeline



# Exemplo com Imputer



# GridSearchCV & RandomizedSearchCV

- Problema:
  - tenho vários hiperparametros para *tuning*...
  - ... e cada hiperparametro pode assumir multiplos valores
  - para configuração de hiperparametros tenho que rodar uma validação cruzada
- Solução:
  - **GridSearchCV**:  
uma busca exaustiva sobre todas as possibilidades no grid (\*)
  - **RandomizedSearchCV**:  
busca aleatoria entre todas as possibilidades no grid (\*)

(\*) grid: malha resultante do produto cartesiano dos hiperparametros

# GridSearchCV & RandomizedSearchCV

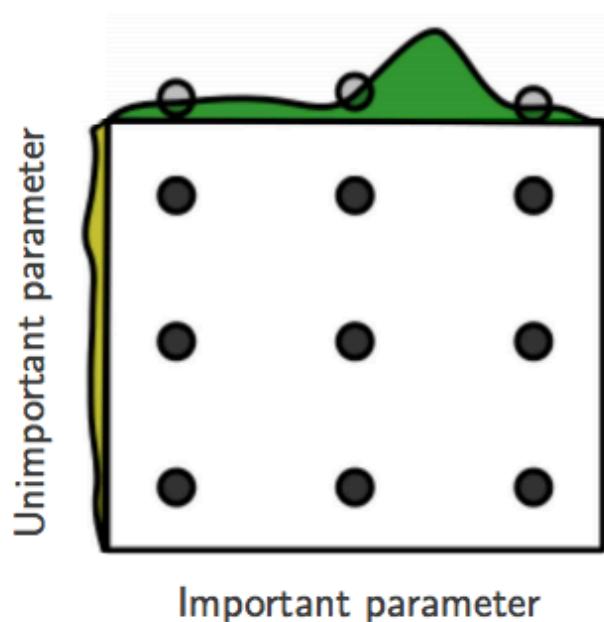
- Exemplo

```
pipeline = Pipeline([
    ('vect', CountVectorizer()),
    ('clf', SGDClassifier()),
])
parameters = [
    {
        'vect_max_df': (0.5, 0.75, 1.0),
        'clf': (SGDClassifier(),),
        'clf_alpha': (0.00001, 0.000001),
        'clf_penalty': ('l2', 'elasticnet'),
        'clf_n_iter': (10, 50, 80),
    }, {
        'vect_max_df': (0.5, 0.75, 1.0),
        'clf': (LinearSVC(),),
        'clf_C': (0.01, 0.5, 1.0)
    }
]
grid_search = GridSearchCV(pipeline, param_grid = parameters)

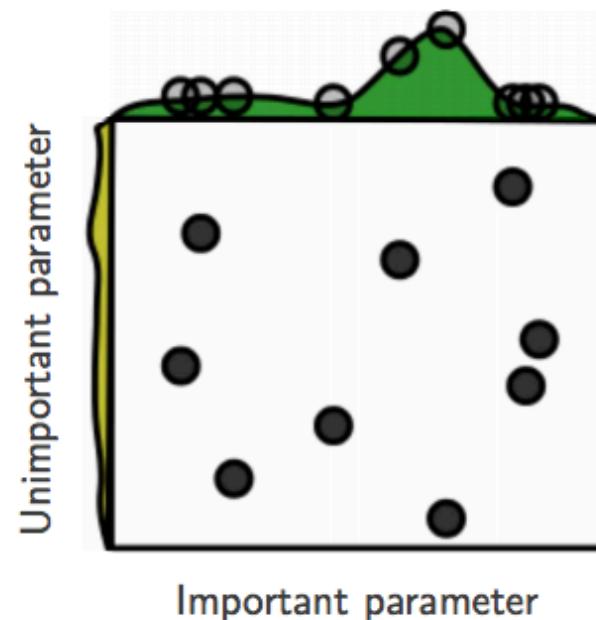
GridSearchCV.fit(X_train, y_train)
```

# GridSearchCV & RandomizedSearchCV

Grid Layout



Random Layout



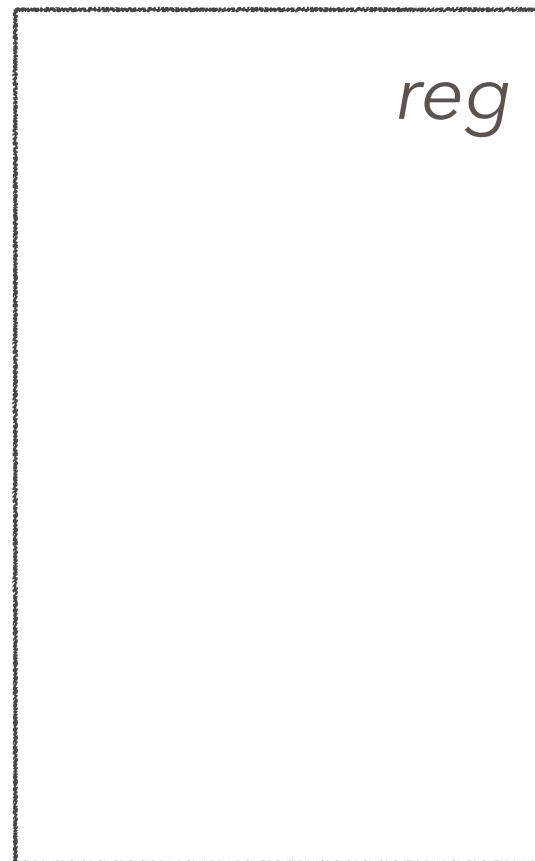
Fonte:

Random Search for Hyper-Parameter Optimization  
James Bergstra, Yoshua Bengio

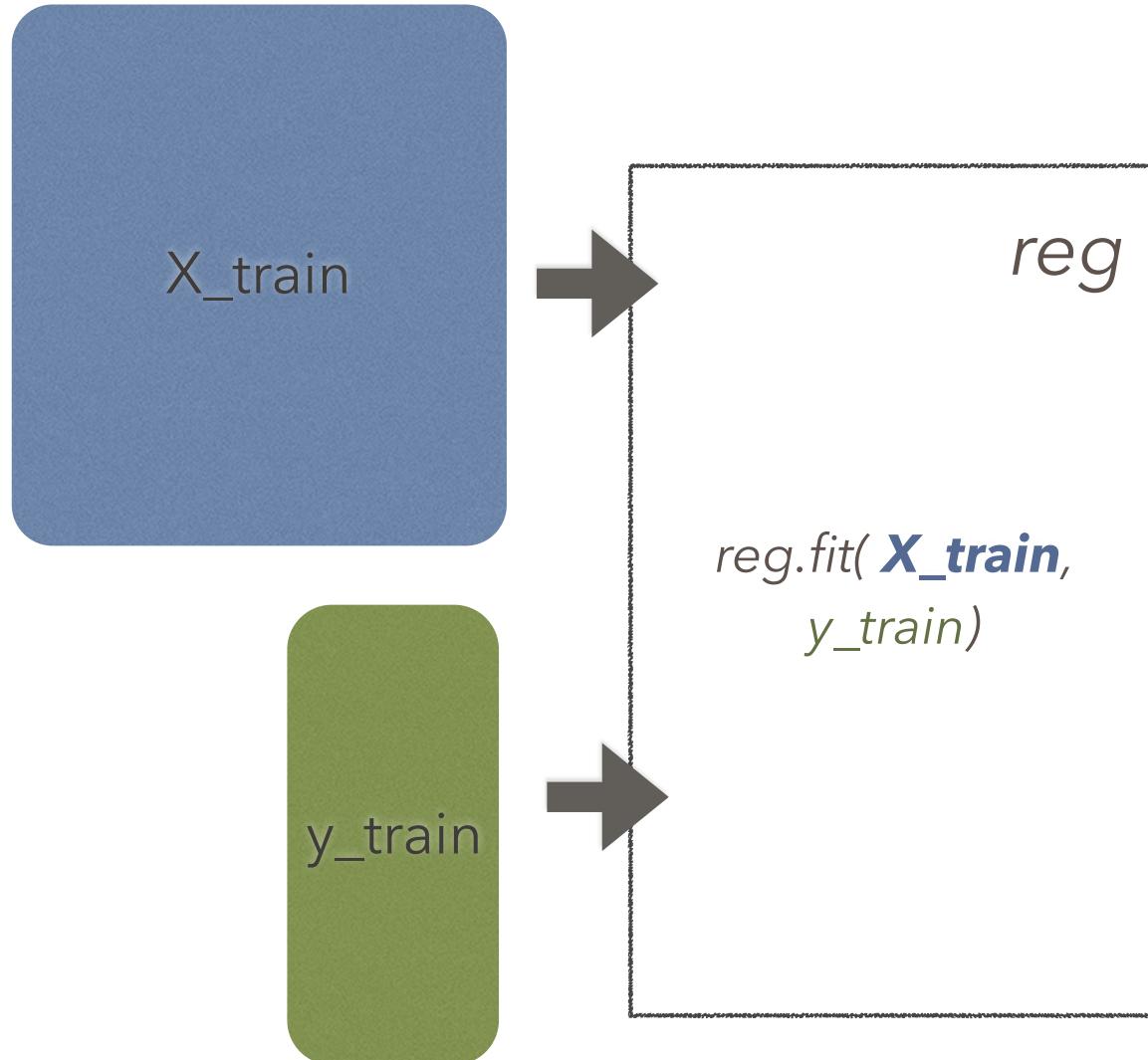
# TREINAMENTO E PREDIÇÃO COM SCIKIT-LEARN



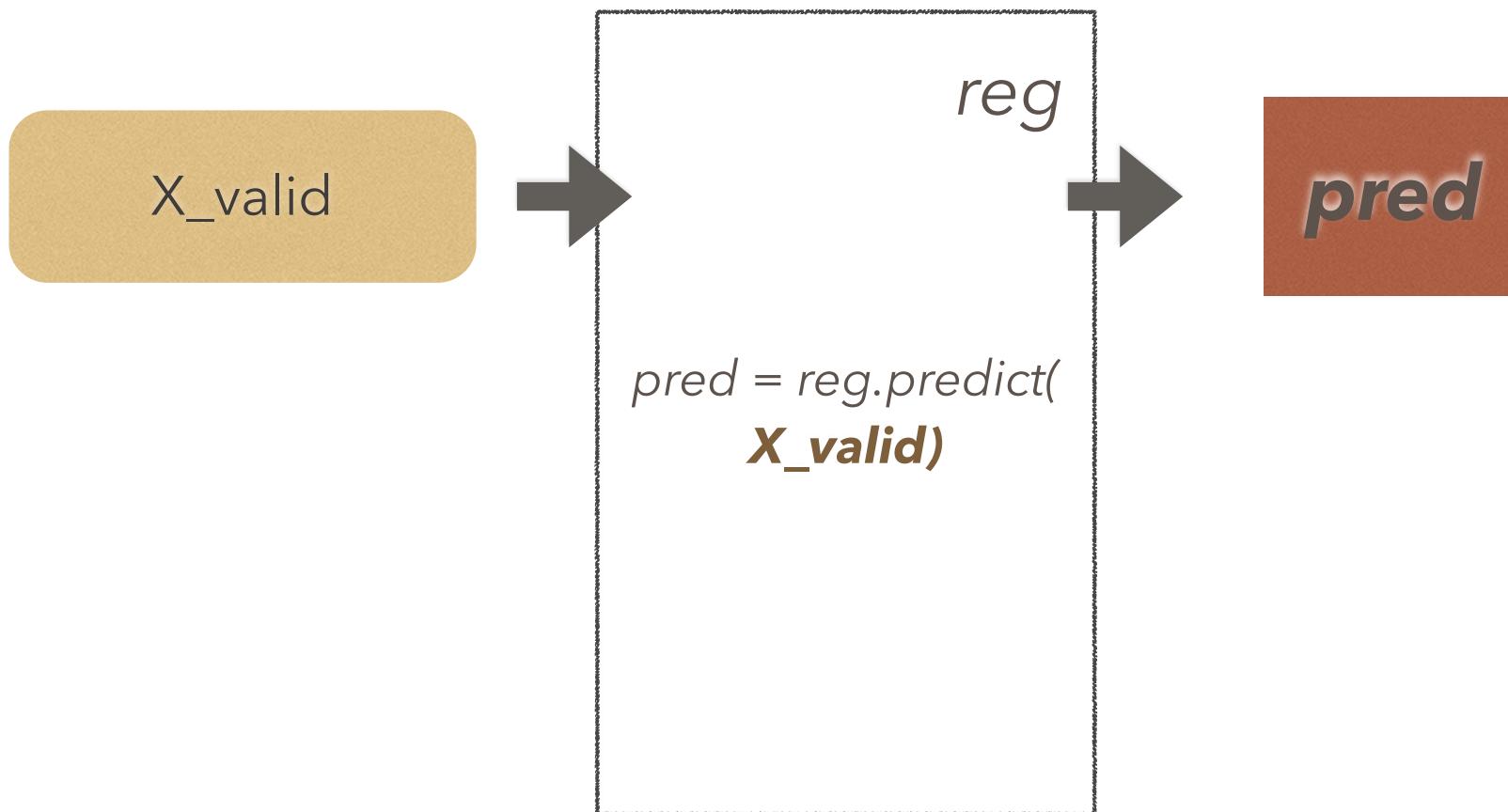
```
reg = LinearRegression()
```



## 2) treinar (fit)



### 3) predizer (predict)



*Objetivo: quanto menor o valor, melhor o modelo*

$$\text{mean}(\text{pred} - \text{y_valid})^2$$

