
MODEL LIFECYCLE

# CRIANDO MODELOS SIMPLES DE MACHINE LEARNING II

DIEGO RODRIGUES DSC

INFNET

# MODEL LIFECYCLE : CRIANDO MODELOS SIMPLES DE MACHINE LEARNING II

## PARTE 1 : TEORIA

- DATA PREPARATION
  - AUMENTO DE DADOS
  - NORMALIZAÇÃO
- MODELING
  - BIAS VS VARIANCE
- EVALUATION
  - VALIDAÇÃO CRUZADA
  - PONTO DE OPERAÇÃO
  - MATRIZ DE CONFUSÃO

## PARTE 2 : PRÁTICA

- NOTEBOOK CLASSIFICAÇÃO IRIS KNN KFOLDS

Produzir Ação

# CICLO DE VIDA DO MODELO

Baseado em Dados

# AMBIENTE PYTHON

5. Visualização

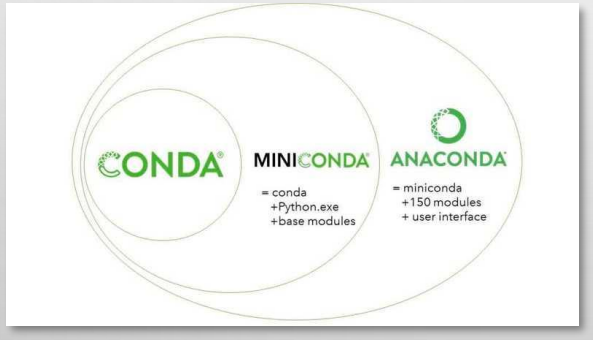
6. Estimação e Inferência

7. Machine Learning


4. Variáveis Aleatórias




1. Editor de Código




2. Gestor de Ambiente

3. Ambiente Python do Projeto


3. Notebook Dinâmico

Cross Industry Standard Process for Data Mining - IBM


**1) Requerimentos e Análise de Negócio**

Entendimento do problema decisório, dados relacionados & revisão bibliográfica.

**2) Preparação dos Dados**

Entendimento das fontes de dados, dos tipos e elaboração da representação.

**3) Modelagem**

Análise Exploratória, Seleção de atributos e treinamento.

**4) Avaliação**

Seleção do melhor modelo.

**5) Liberação**

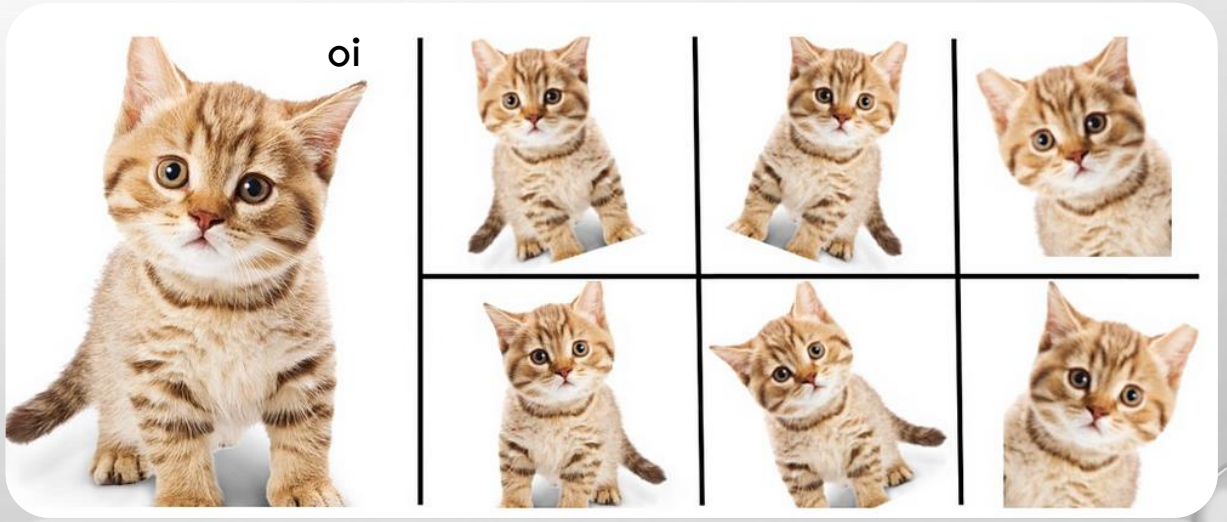
Liberação do modelo no ambiente de produção.

# DATA PREPARATION

# AUMENTO DE DADOS

**Aumento de Dados (Data Augmentation)** é uma técnica poderosa utilizada para aumentar a **quantidade e diversidade** do conjunto de dados de treinamento**, sem a necessidade de coletar novos dados.**

Ela é aplicada em vários domínios, como **imagens, texto e áudio,** e envolve a **criação de novas observações** a partir de **re-amostragem ou transformações** nos dados existentes.

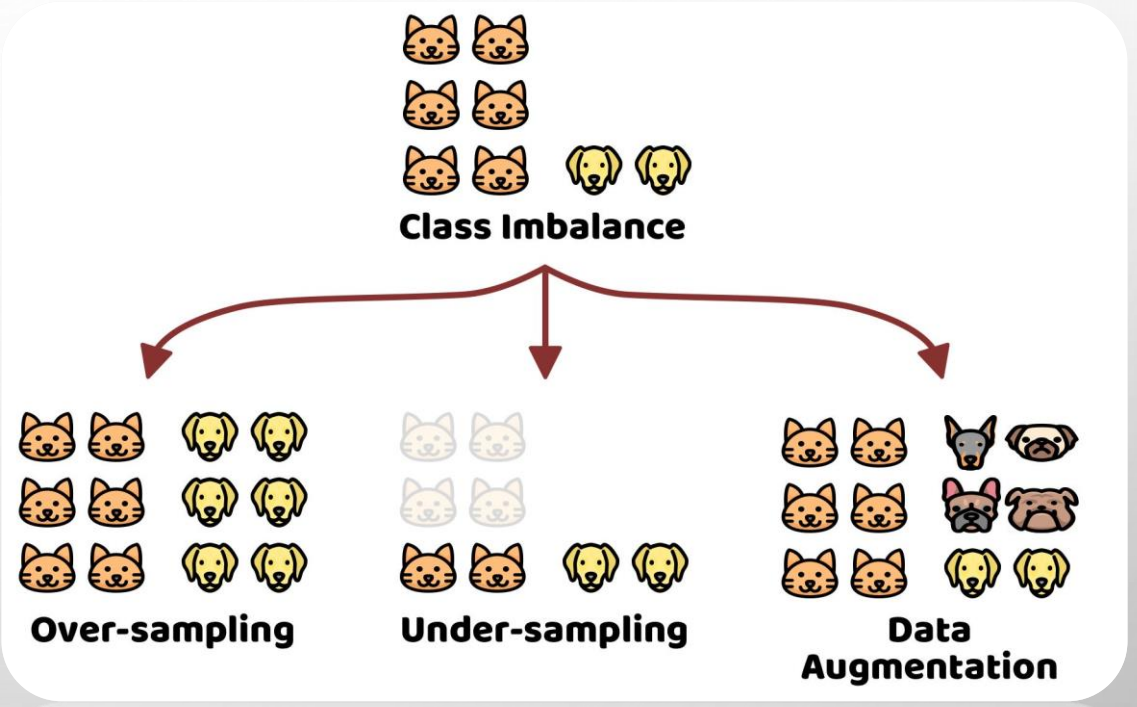


# AUMENTO DE DADOS: BALANCEAMENTO DE CLASSES

**Aumento de Dados** pode ser utilizado para diferentes **estratégias de balanceamento das classes**!

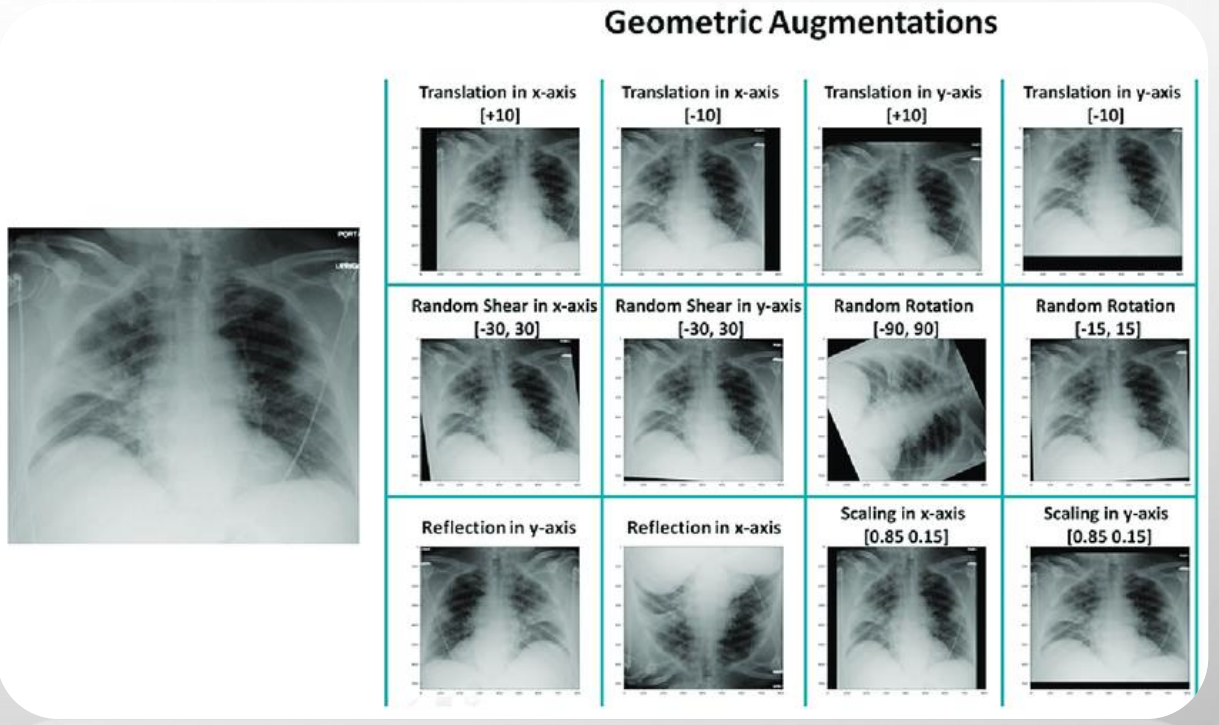
**BENEFÍCIOS ADICIONAIS**

- **Robustez a Ruído:** Aumentar a capacidade do modelo de lidar com dados ruidosos ou imperfeitos, introduzindo variações que o ajudam a generalizar melhor.
- **Melhoria da Generalização:** Evitar o overfitting ao modelo de treinamento, fornecendo exemplos diversificados que permitem ao modelo aprender características mais robustas.



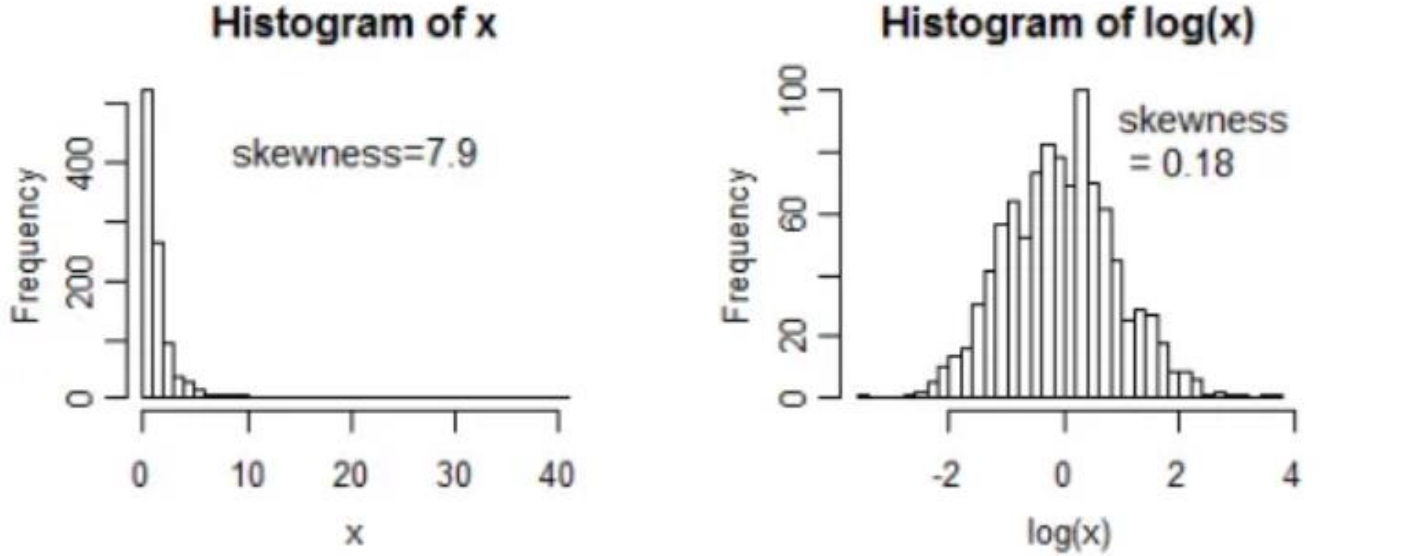
# AUMENTO DE DADOS: TIPOS DE TRANSFORMAÇÕES

- **Transformações Geométricas:** Como rotações, translações, e alterações de escala, aplicadas a diferentes tipos de dados.
- **Alterações em Valores:** Como mudança na intensidade de pixels em imagens, **adição de ruído** ou substituição de palavras em textos.
- **Composição e Mistura:** Combinação de diferentes amostras de dados para criar novos exemplos compostos.



# NORMALIZAÇÃO

# NORMALIZAÇÃO




Example distribution before (left) and after (right) log transformation

https://medium.com/@isalindgren313/transformations-scaling-and-normalization-420b2be12300

Transformar as variáveis originais por funções, facilitando o problema numérico de otimização e ao mesmo tempo inserindo "não-linearidades" para resolver um problema não-linear de forma linear.

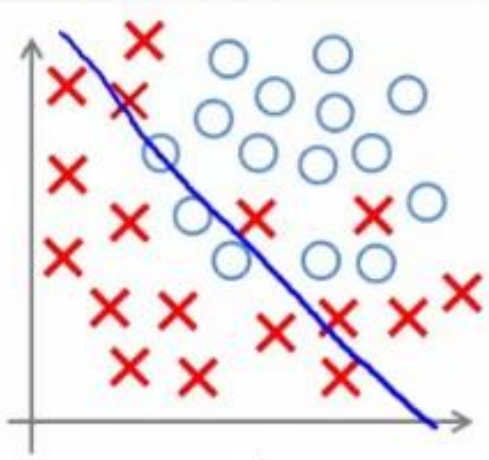
# NORMALIZAÇÃO

| TRANSFORMATION | USE IF | LIMITATIONS | SPSS EXAMPLES |
|---|---|---|---|
| Square/Cube Root | Variable shows positive skewness<br>Residuals show positive heteroscedasticity<br>Variable contains frequency counts | Square root only applies to positive values | compute newvar = sqrt(oldvar).<br>compute newvar = oldvar**(1/3). |
| Logarithmic | Distribution is positively skewed | Ln and log10 only apply to positive values | compute newvar = ln(oldvar).<br>compute newvar = lg10(oldvar). |
| Power | Distribution is negatively skewed | (None) | compute newvar = oldvar**3. |
| Inverse | Variable has platykurtic distribution | Can't handle zeroes | compute newvar = 1 / oldvar. |
| Hyperbolic Arcsine | Distribution is positively skewed | (None) | compute newvar = ln(oldvar + sqrt(oldvar**2 + 1)). |
| Arcsine | Variable contains proportions | Can't handle absolute values > 1 | compute newvar = arsin(oldvar). |

https://www.spss-tutorials.com/normalizing-variable-transformations/

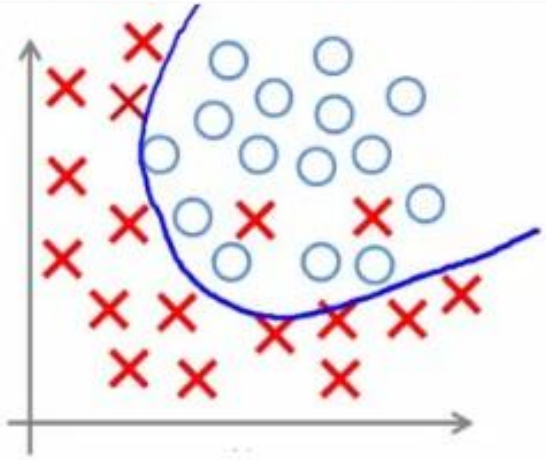
# MODELING

# CAPACIDADE E GENERALIZAÇÃO

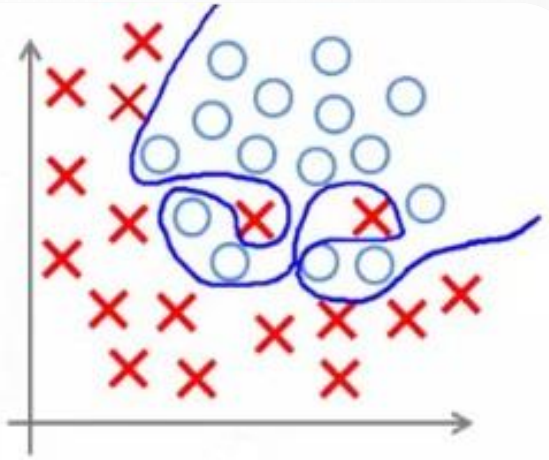


**Under-fitting**

(too simple to explain the variance)



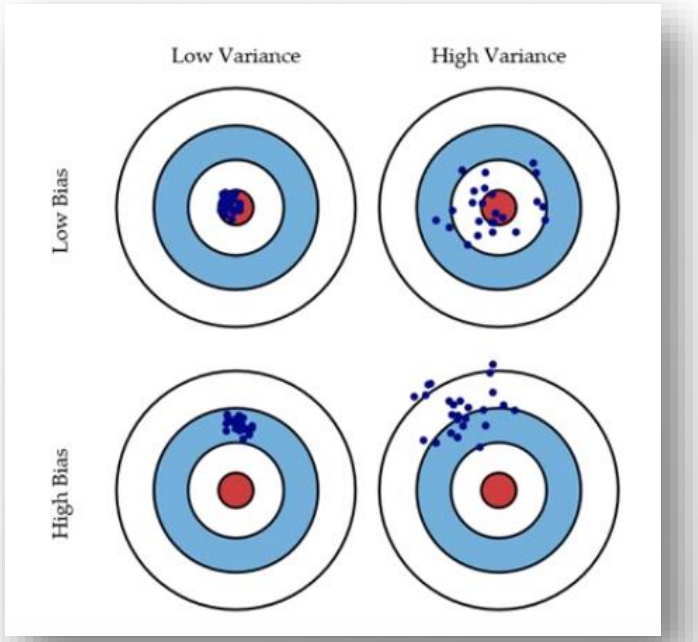
**Appropriate-fitting**

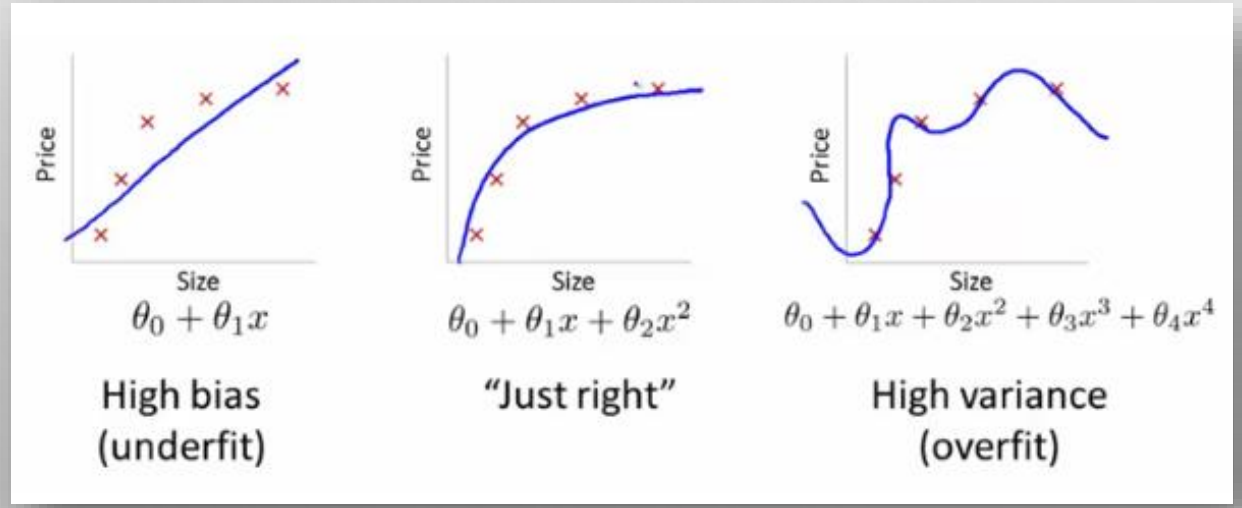


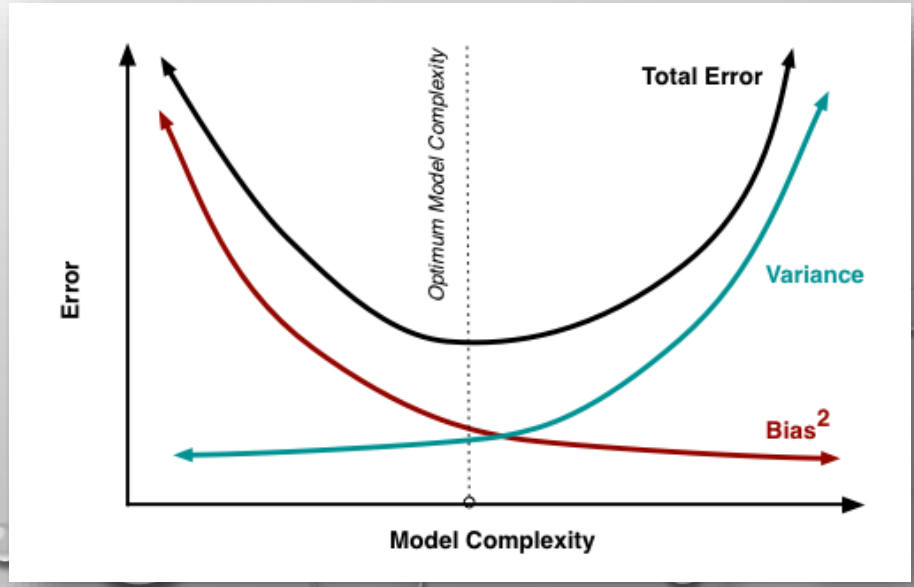
**Over-fitting**

(forcefitting -- too good to be true)

# BIAS x VARIANCE

# REGULARIZAÇÃO

In mathematics, statistics, finance,[1] and computer science, particularly in machine learning and inverse problems, **regularization** is a process that changes the result answer to be "simpler". It is often used to obtain results for ill-posed problems or to prevent overfitting.[2]

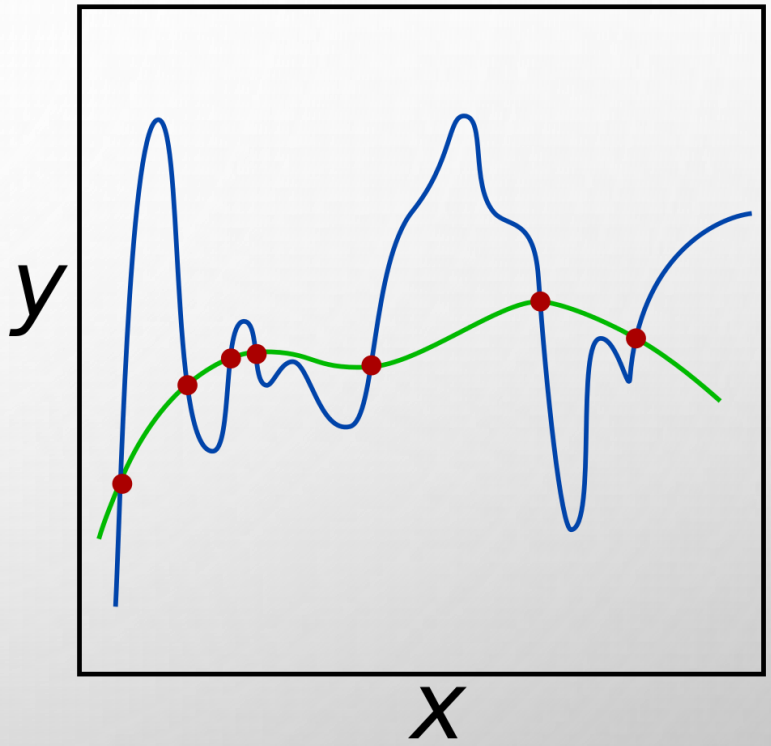
Although regularization procedures can be divided in many ways, the following delineation is particularly helpful:

- **Explicit regularization** is regularization whenever one explicitly adds a term to the optimization problem. These terms could be priors, penalties, or constraints. Explicit regularization is commonly employed with ill-posed optimization problems. The regularization term, or penalty, imposes a cost on the optimization function to make the optimal solution unique.
- **Implicit regularization** is all other forms of regularization. This includes, for example, early stopping, using a robust loss function, and discarding outliers. Implicit regularization is essentially ubiquitous in modern machine learning approaches, including stochastic gradient descent for training deep neural networks, and ensemble methods (such as random forests and gradient boosted trees).

In explicit regularization, independent of the problem or model, there is always a data term, that corresponds to a likelihood of the measurement and a regularization term that corresponds to a prior. By combining both using Bayesian statistics, one can compute a posterior, that includes both information sources and therefore stabilizes the estimation process. By trading off both objectives, one chooses to be more addictive to the data or to enforce generalization (to prevent overfitting). There is a whole research branch dealing with all possible regularizations. In practice, one usually tries a specific regularization and then figures out the probability density that corresponds to that regularization to justify the choice. It can also be physically motivated by common sense or intuition.

In machine learning, the data term corresponds to the training data and the regularization is either the choice of the model or modifications to the algorithm. It is always intended to reduce the generalization error, i.e. the error score with the trained model on the evaluation set and not the training data.[3]

One of the earliest uses of regularization is Tikhonov regularization (ridge regression), related to the method of least squares.



# EVALUATION

# GENERALIZAÇÃO: IDENTIFICANDO OS HIPER-PARÂMETROS ÓTIMOS

**LEAVE ONE OUT**

- Uma única observação é deixada de fora a cada treinamento. N treinamentos são realizados para calcular a estatística de erro.

**K FOLDS**

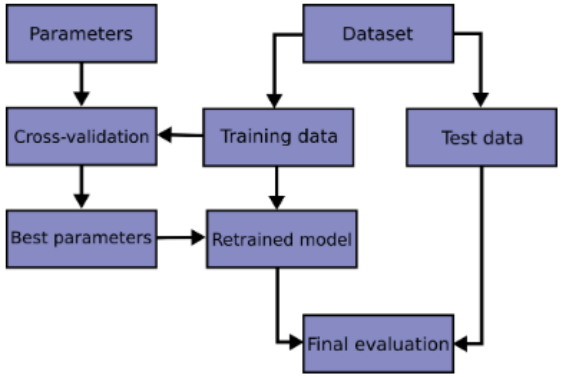
- Amostra é dividida em K conjuntos. K treinamentos são realizados, mantendo um conjunto como fora-da-amostra.

**BOOTSTRAPPING**

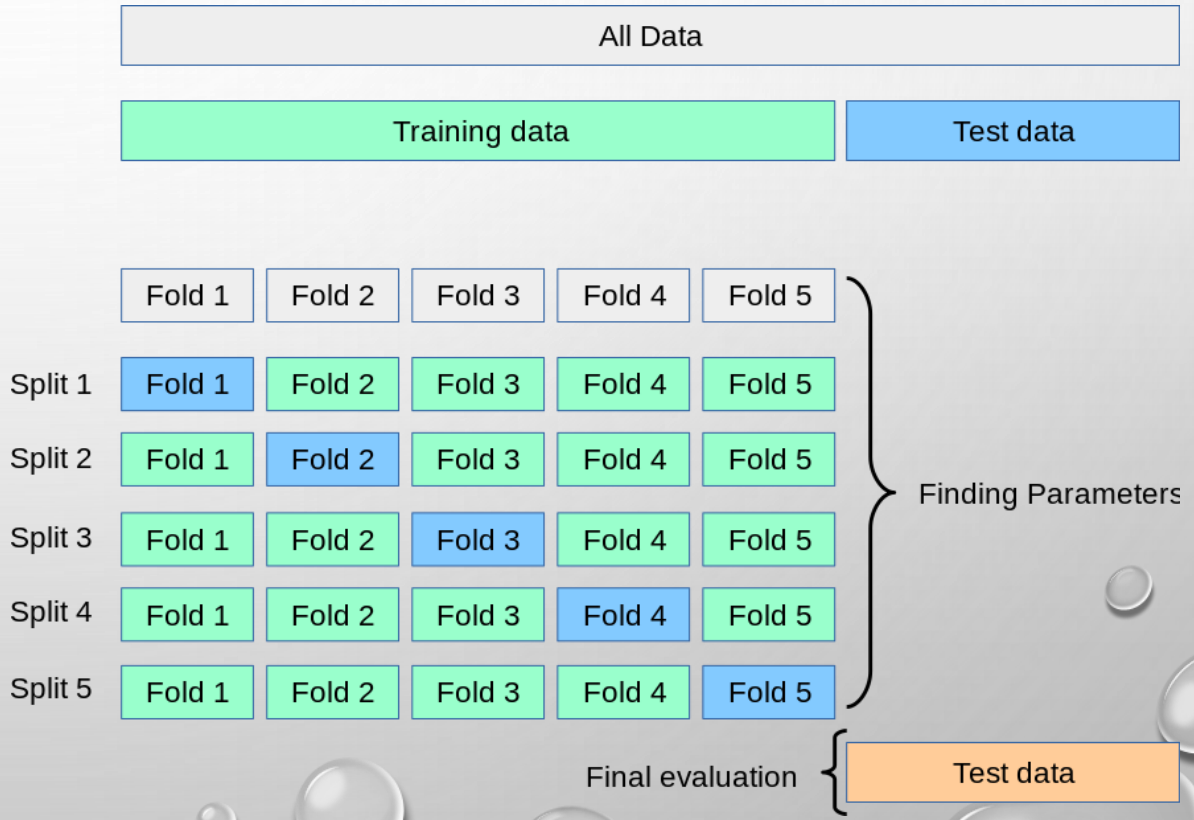
- O algoritmo itera, amostrando aleatoriamente M observações quantidade Q desejada de treinamentos.

# 3.1. Cross-validation: evaluating estimator performance

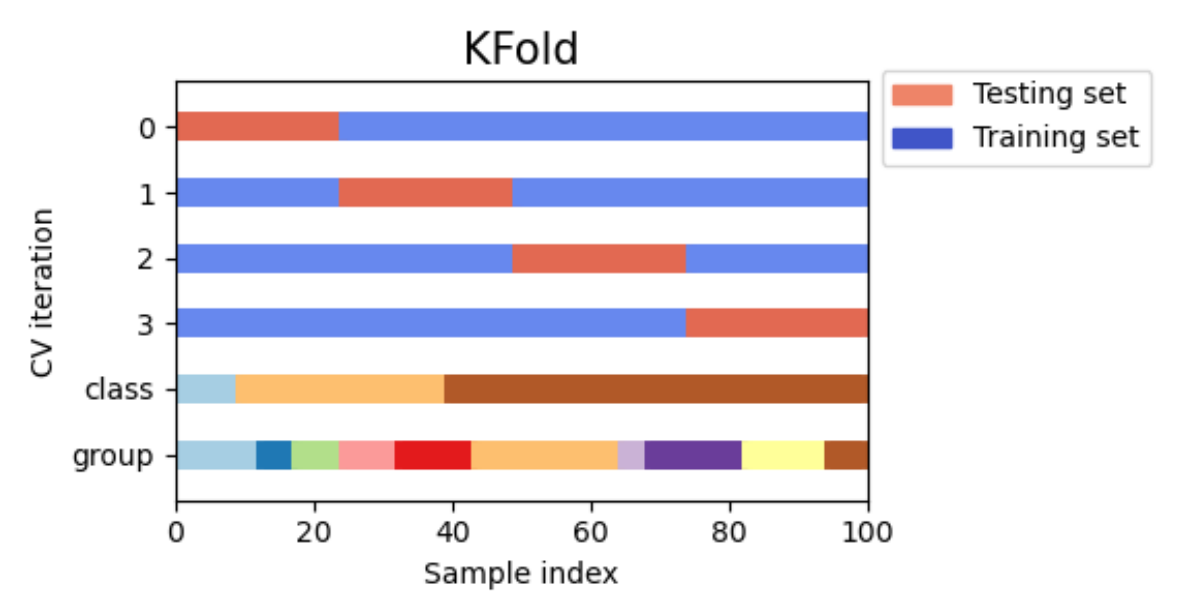
Learning the parameters of a prediction function and testing it on the same data is a methodological mistake: a model that would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on yet-unseen data. This situation is called **overfitting**. To avoid it, it is common practice when performing a (supervised) machine learning experiment to hold out part of the available data as a **test set** `X_test, y_test`. Note that the word "experiment" is not intended to denote academic use only, because even in commercial settings machine learning usually starts out experimentally. Here is a flowchart of typical cross validation workflow in model training. The best parameters can be determined by grid search techniques.

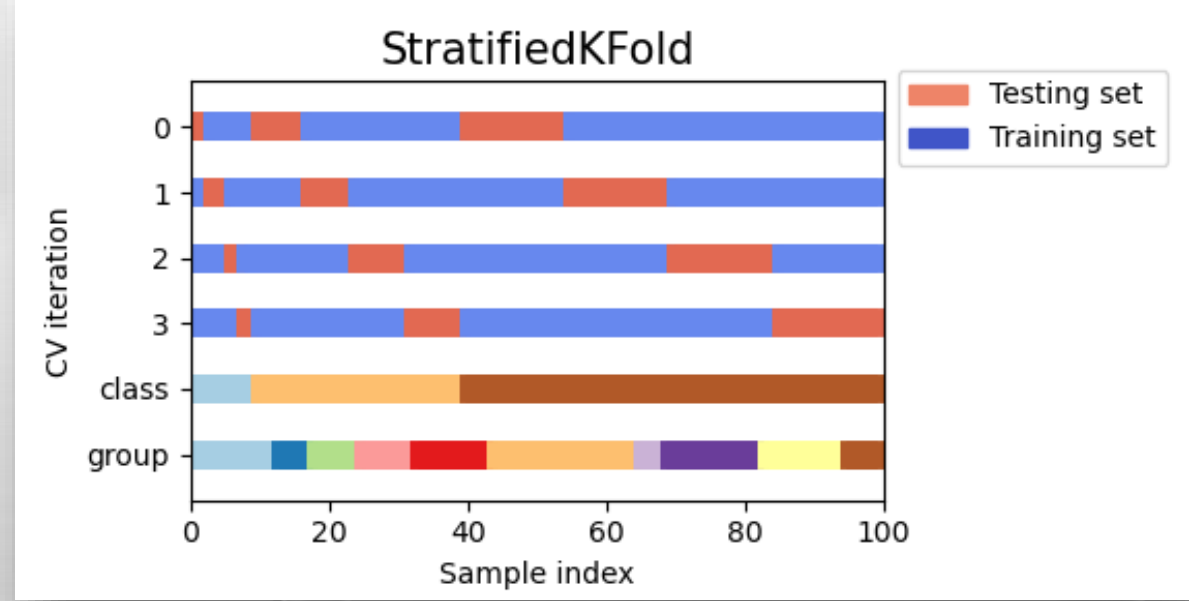



# VALIDAÇÃO CRUZADA



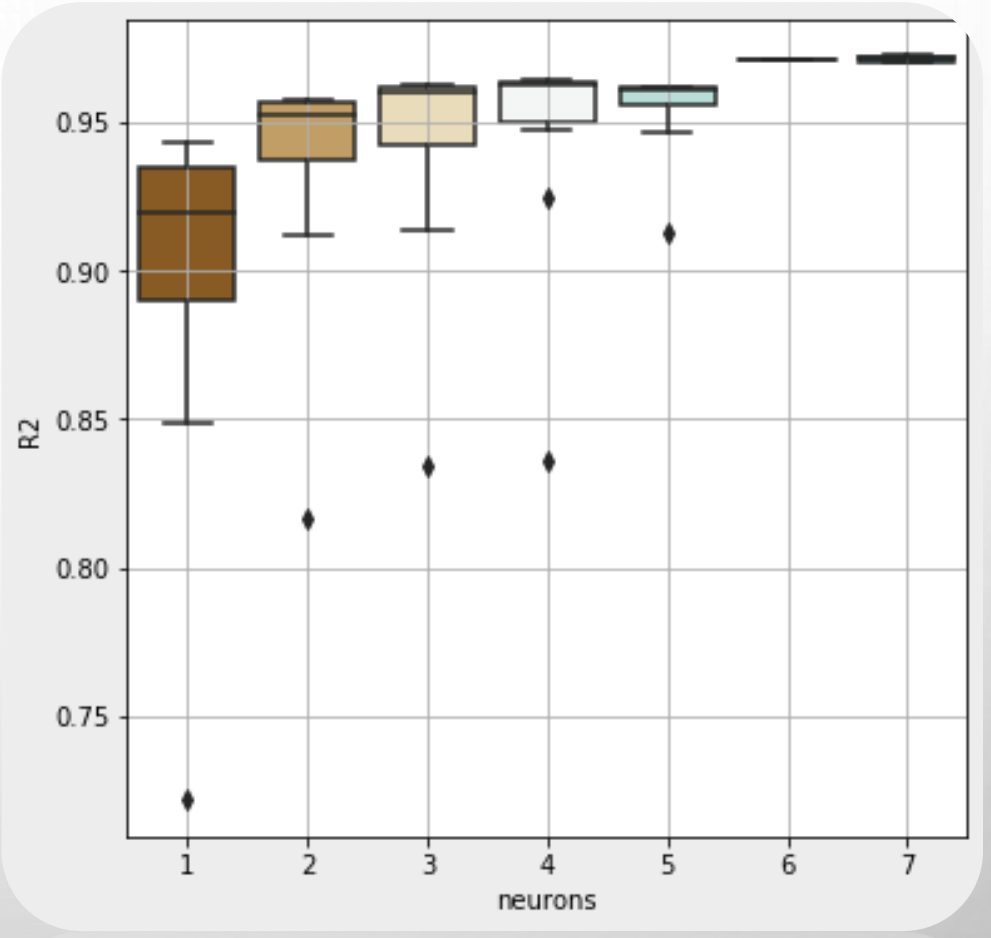
# K-FOLDS & K-FOLDS ESTRATIFICADO



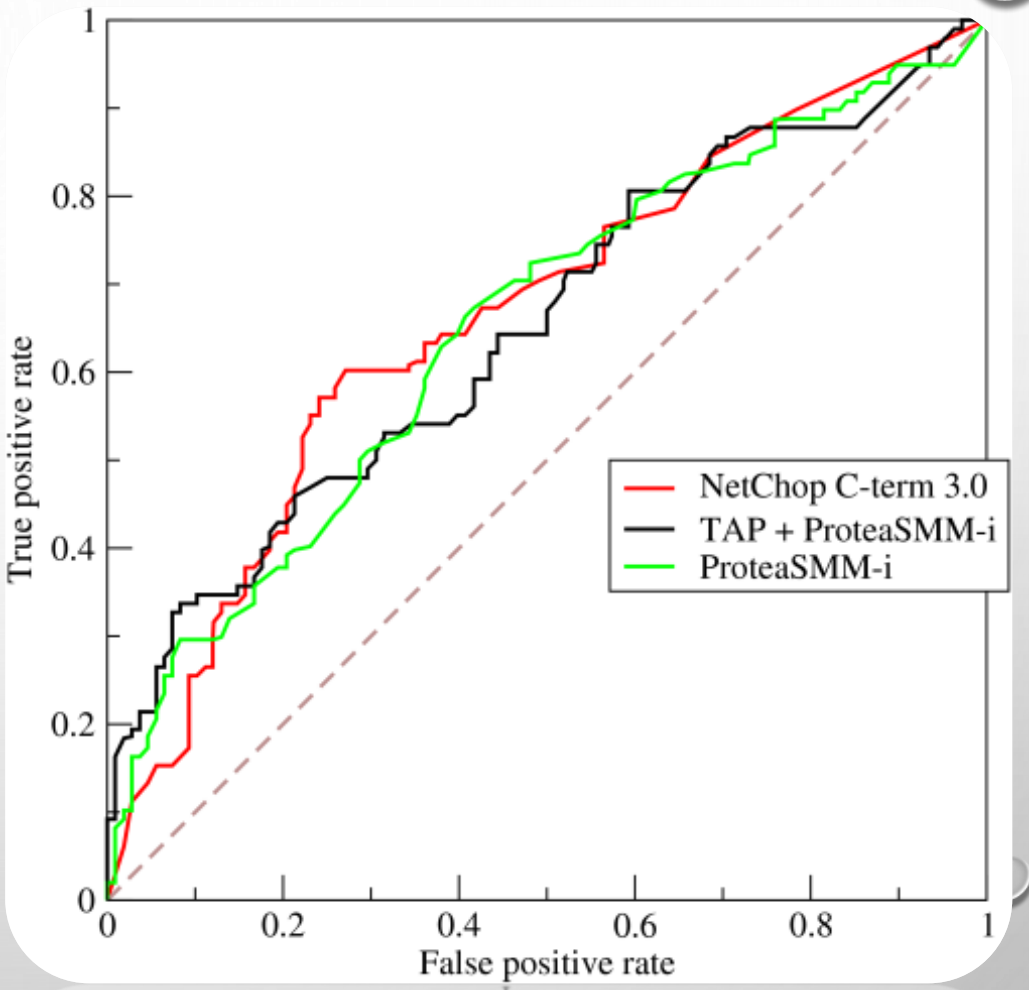
# K-FOLDS - EXEMPLO

- **Iteração dos hiperparâmetros**
- **Seleção da Figura de Mérito**
- **Seleção da Estatística de Ganho**



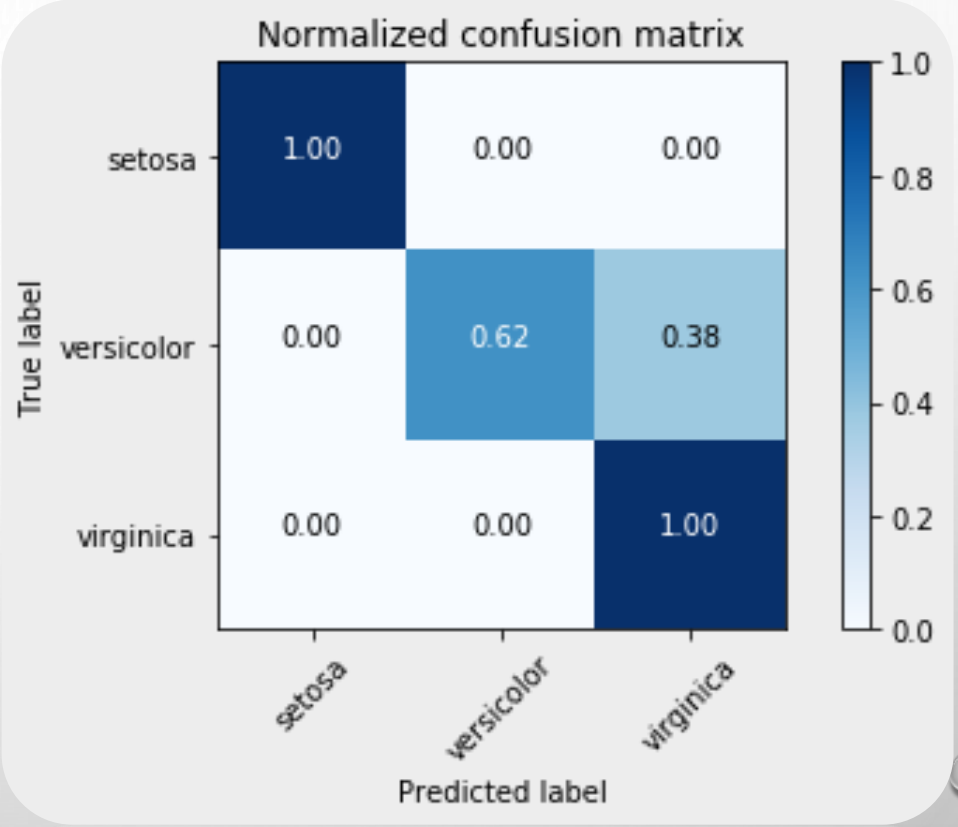
# PONTO DE OPERAÇÃO

- **Curva ROC**
  - Calibra a saída do modelo, ajudando a configurar o ponto de operação entre Precisão / Recall / Acurácia.

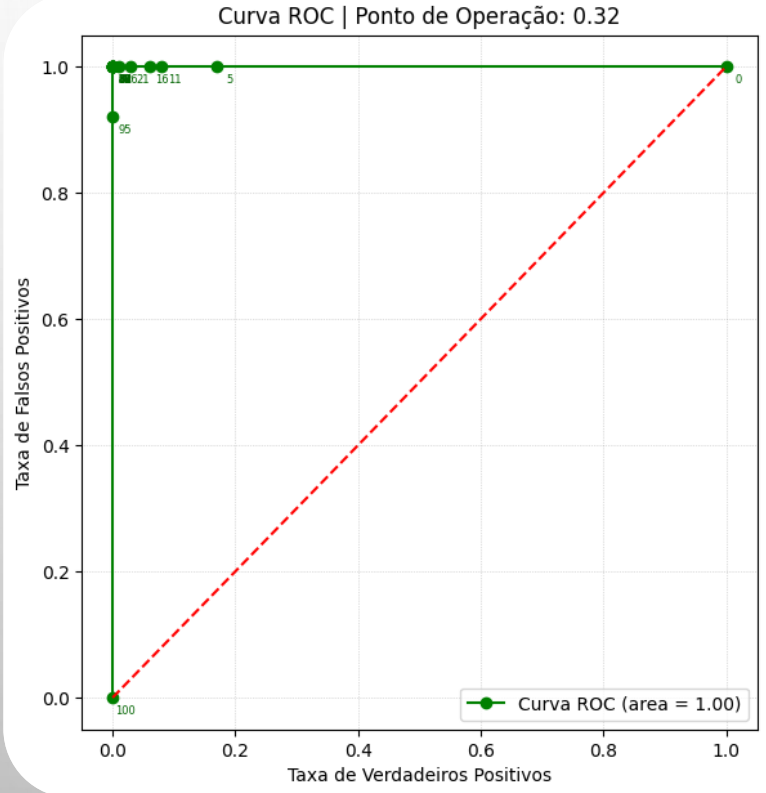


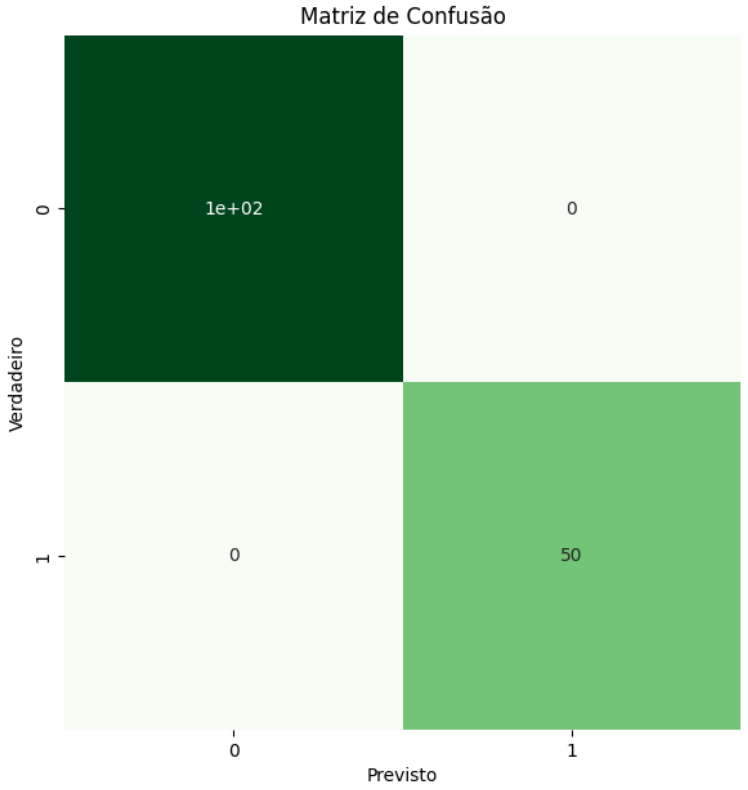
# MATRIZ DE CONFUSÃO

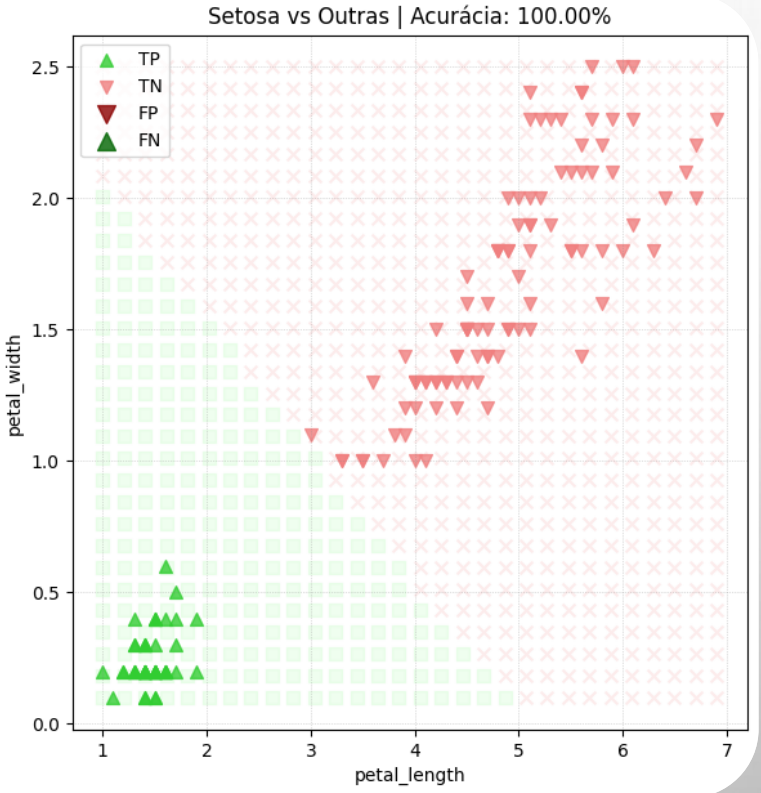
Comparação entre o resultado do classificador para as diferentes classes.



# PONTO DE OPERAÇÃO



# CRIANDO MODELOS SIMPLES DE MACHINE LEARNING III