REDES NEURAIS E DEEP LEARNING

INTRODUÇÃO

DIEGO RODRIGUES DSC

INFNET

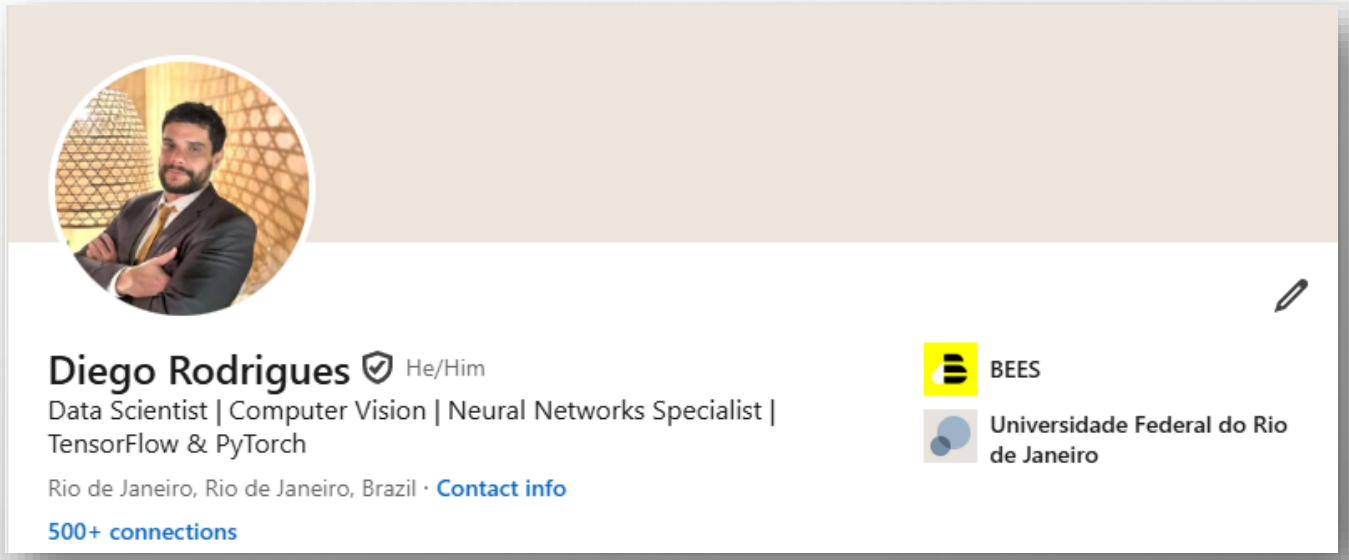
Bloco	Matéria	Calendário	Avaliação
Treinamento Clássico	Introdução	06/10	
	Classificação	08, *13	
	Regressão	20, *22	
	Agrupamento	27, *29	
	Séries Temporais	03/11, *05	<Modelo Clássico>
Redes Profundas	Deep Feed Forward	10, *12	
	Visão Computacional	17, *19	
	Autoencoders	24, *26	<Modelo Profundo>
Treinamento Moderno	Transfer Learning	01/12, *03	
	Sequências	08, *10	
	Modelos Generativos	15, *17	<Modelo Avançado>

INTRODUÇÃO

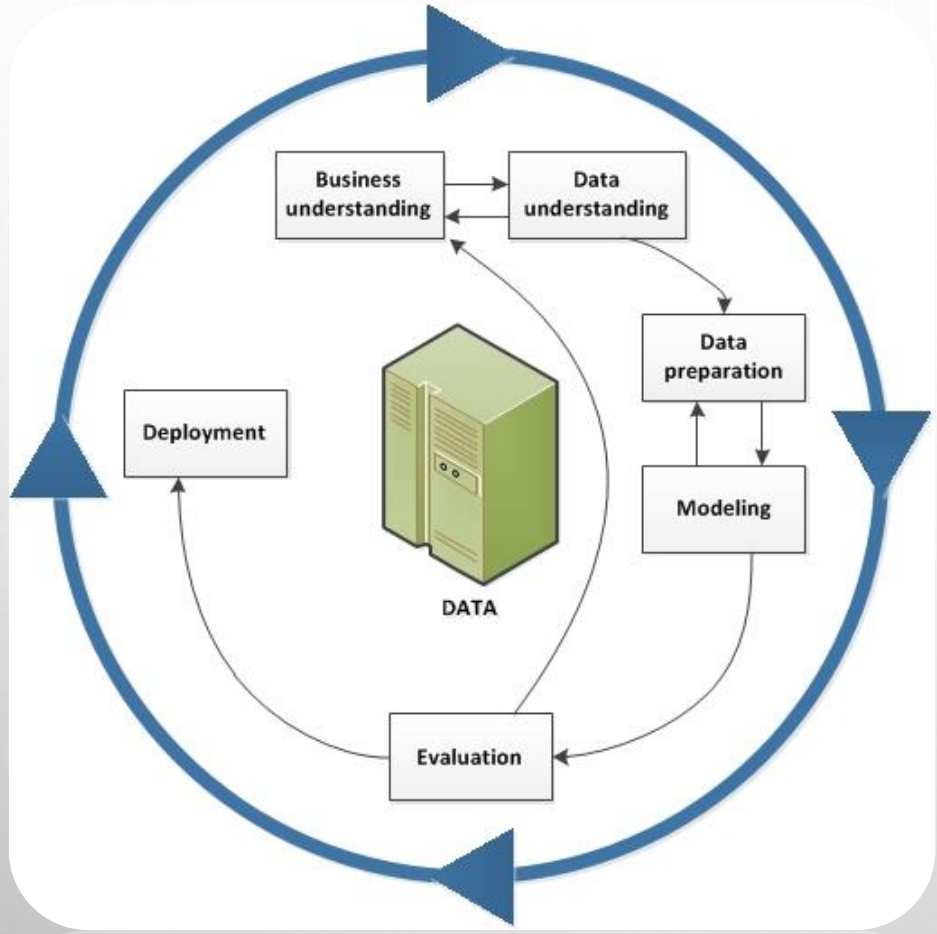
- APRESENTAÇÃO
- PARTE 1 : TEORIA
  - CRISP
    - BUSINESS UNDERSTANDING
    - DATA UNDERSTANDING & PREPARATION
    - MODELING
      - REDES NEURAIS
    - EVALUATION
- PARTE 2 : PRÁTICA
  - AMBIENTE DE DESENVOLVIMENTO
- PARTE 3 : TRABALHOS

PROFESSOR DIEGO

- Doutor em Inteligência Computacional COPPE / UFRJ
- 17 anos de experiência com ciência de dados aplicada à engenharia.
- Cientista de Dados Sênior na Bees
- Professor da INFNET na Graduação, Pós-Graduação, Trilhas e Bootcamp.
- Conselheiro na Interagente.



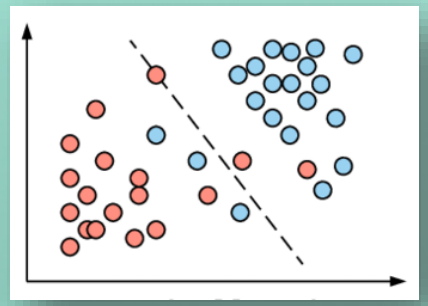
PARTE 1 : TEORIA



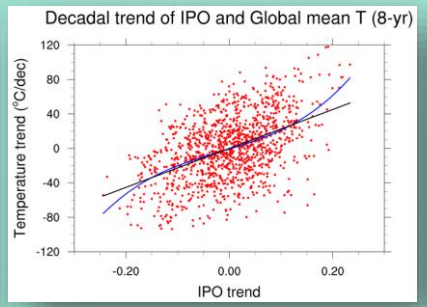
CROSS INDUSTRY PROCESS FOR DATA MINING (CRISP-DM)

BUSINESS UNDERSTANDING

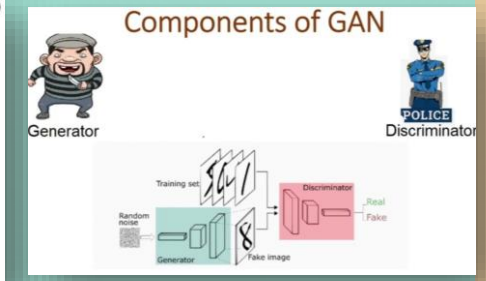
APRENDIZADO SUPERVISIONADO



CLASSIFICAÇÃO

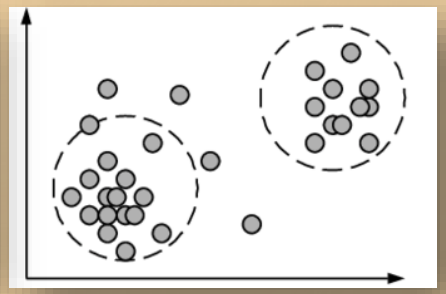


REGRESSÃO



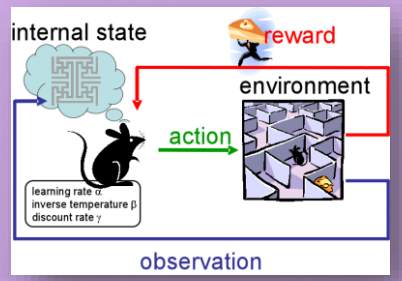
GENERATIVO

APRENDIZADO NÃO-SUPERVISIONADO



AGRUPAMENTO

APRENDIZADO POR REFORÇO



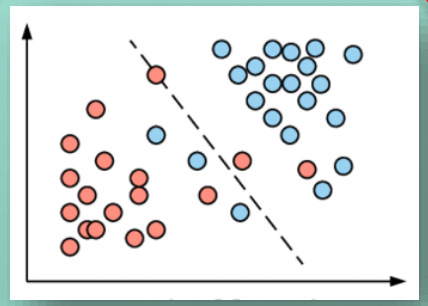
REFORÇO

APRENDIZADO SUPERVISIONADO

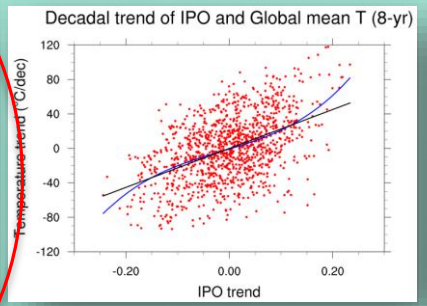
Tarefas de **classificação** e **regressão** pertencem a esta categoria.

O treinamento consiste em **encontrar parâmetros** para o modelo que **minimiza uma função de risco/erro** para uma amostra de treinamento, baseado na diferença entre os **valores previstos e reais,** para cada observação.

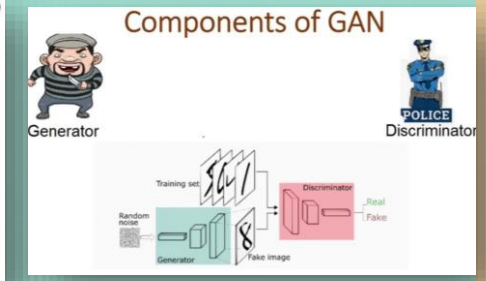
APRENDIZADO SUPERVISIONADO



CLASSIFICAÇÃO

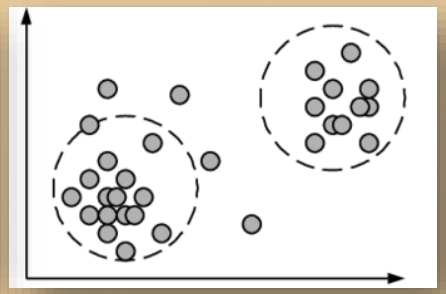


REGRESSÃO



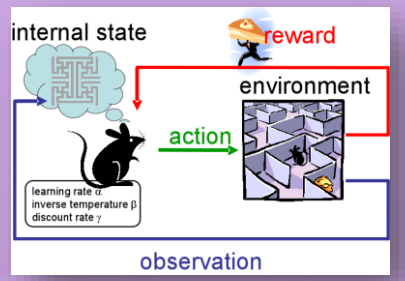
GENERATIVO

APRENDIZADO NÃO-SUPERVISIONADO



AGRUPAMENTO

APRENDIZADO POR REFORÇO



REFORÇO

CLASSIFICAÇÃO

Um bebê consegue separar e ordenar blocos com diferentes tamanhos, formas e cores. Ele também consegue **identificar os tipos diferentes de objetos.**

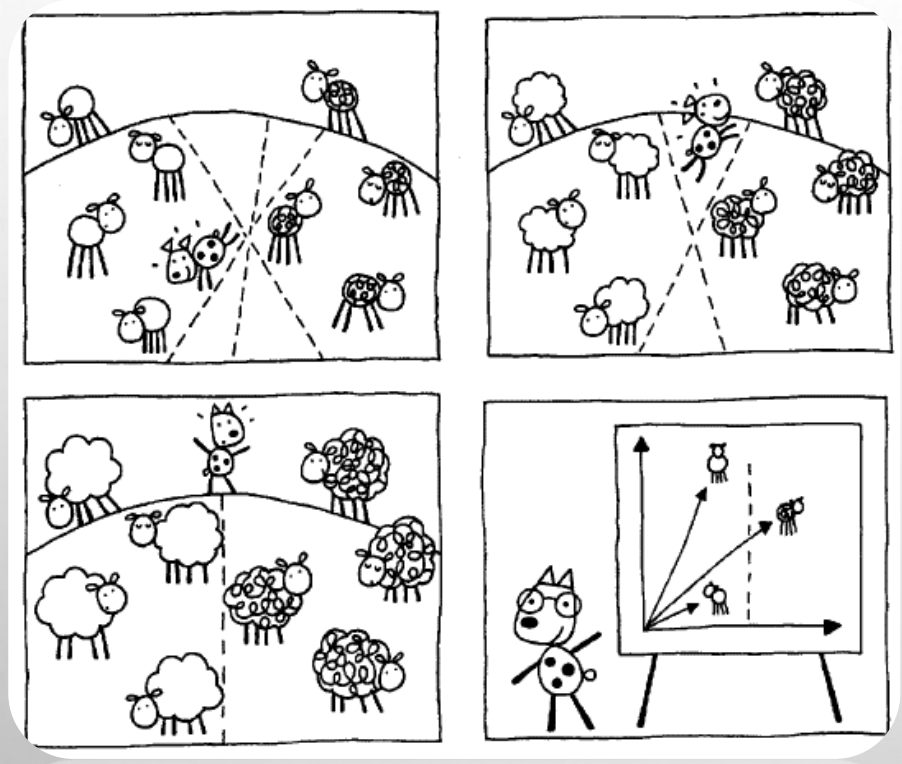
Data Scientist da Nova Geração

Os diferentes tipos de objetos são chamados de **classes**. As características dos objetos são chamadas de **variáveis** ou **atributos**.

Então, um classificador é um modelo **treinado para discriminar objetos** pertencentes a duas ou mais classes, baseado em seus atributos.

DATA UNDERSTANDING & PREPARATION

REPRESENTAÇÃO



Ideia: como quantificar um objeto no mundo físico no mundo digital?

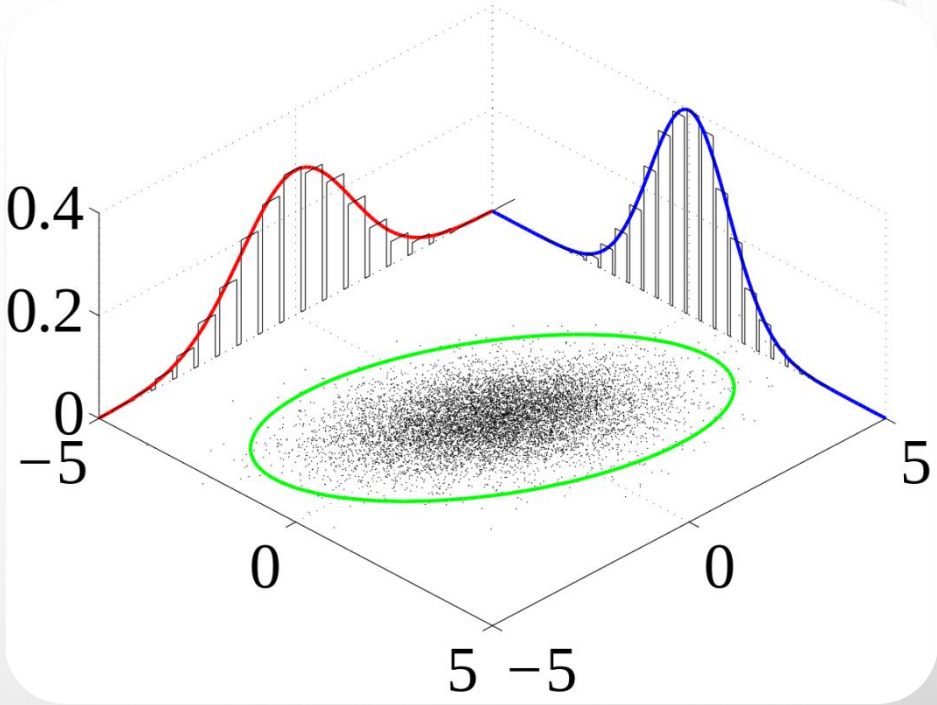
Exercício: qual seria uma boa representação para diferenciar ratos e elefantes?

MODELING

ALGORITMOS BASEADOS EM DENSIDADE

Algoritmos que dependem da **função densidade de probabilidade** dos dados, ou aproximações locais, para determinar a classe de observações fora da amostra de treino.

1) Classificador Bayesiano
2) Classificador Bayesiano "Naïve"
3) K-Vizinhos mais próximos

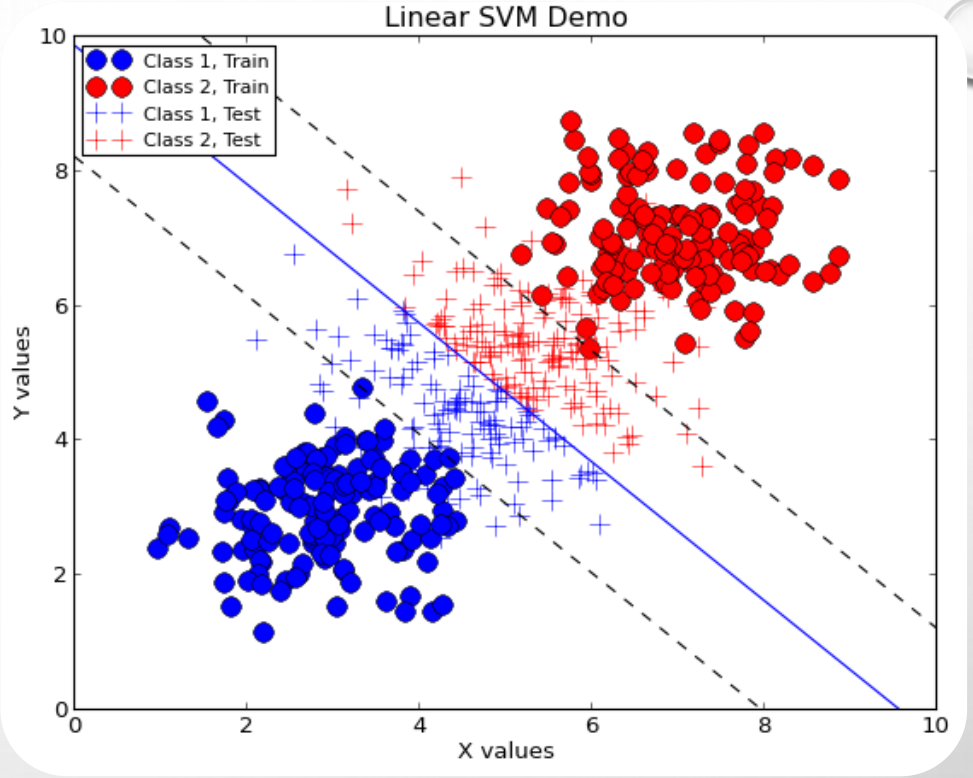


Algoritmos baseados em densidade dependem da **DENSIDADE (!!)**. Consequentemente, se beneficiam de um **conjunto grande de observações e de baixa esparsidade do espaço de atributos**. O Classificador Bayesiano é considerado o classificador "ótimo", mas é raramente utilizado, dada a dificuldade de estimar a função densidade de probabilidade dos dados. É normalmente utilizado como benchmark para comparação teórica entre os algoritmos de classificação.

MODELOS FUNCIONAIS

Algoritmos que dependem da **estimação dos parâmetros de uma função** que é utilizada como **superfície de separação** entre as classes.

1) Funções Polinomiais
2) Regressão Logística
3) Máquina de Vetores Suporte
4) **Neurônio Sigmoide / Tangente Hiperbólica**
5) Árvores de Decisão

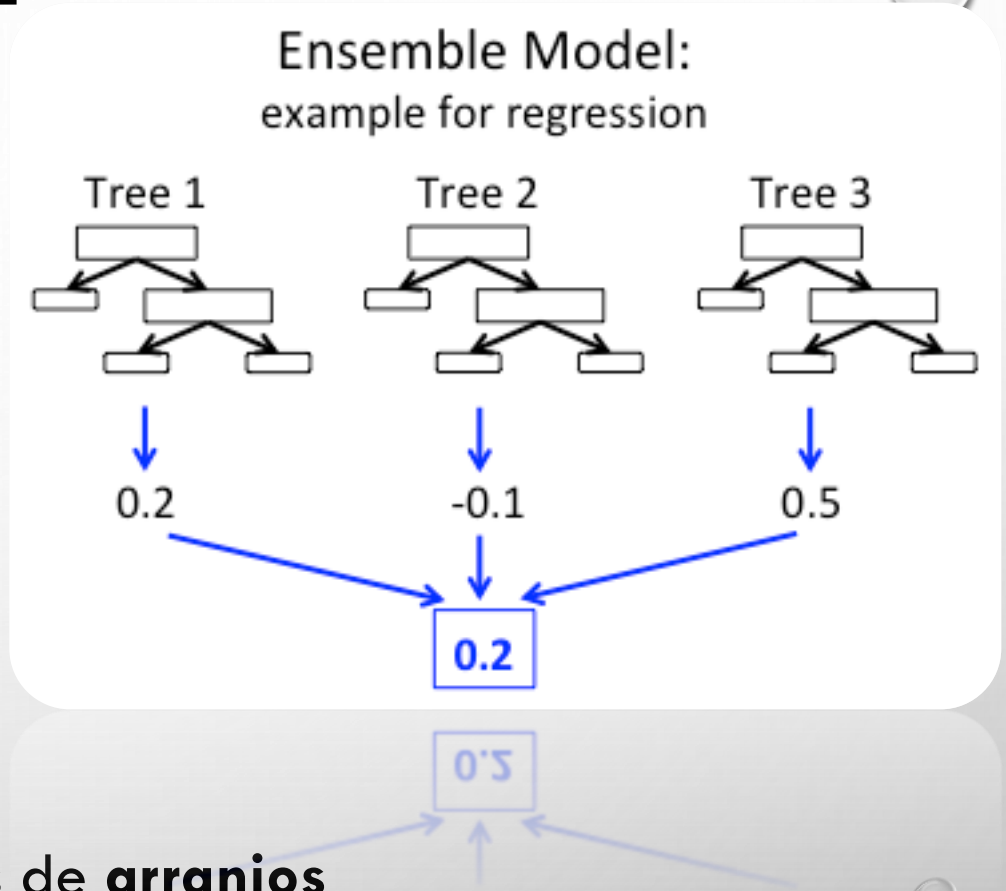


Algoritmos baseados em funções são **mais simples,** usualmente tem um **número menor de parâmetros** e não dependem em armazenar muitos dados para manter uma "memória", como por exemplo K-vizinhos mais próximos.

ALGORITMOS BASEADOS EM ENSEMBLE

Algoritmos que **combinam modelos simples,** usualmente através de **votação ou ponderação,** para atingir maiores taxas de classificação.

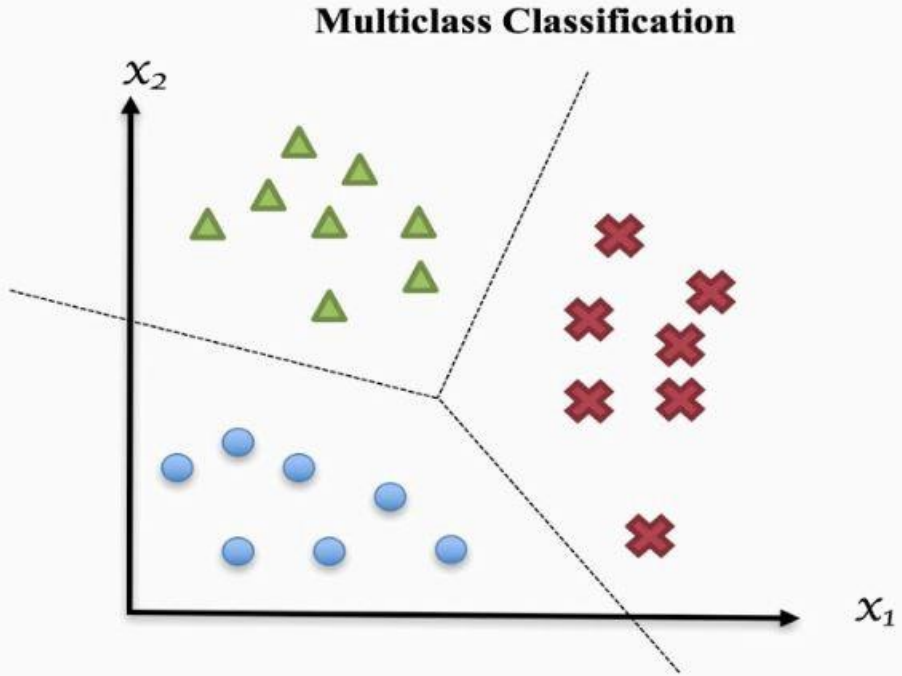
1) **Random Forest**
2) Boosting



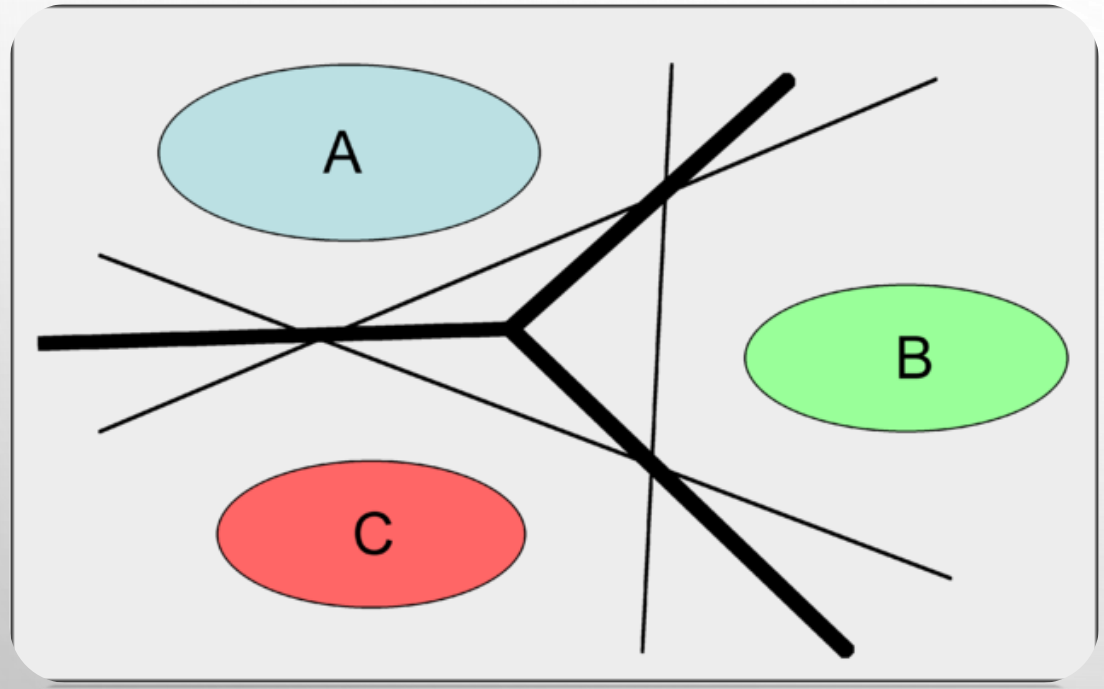
Boa **capacidade de generalização** gerado através de **arranjos complexos** de múltiplos modelos simples de machine learning.

ALGORITMOS BASEADOS EM ENSEMBLE

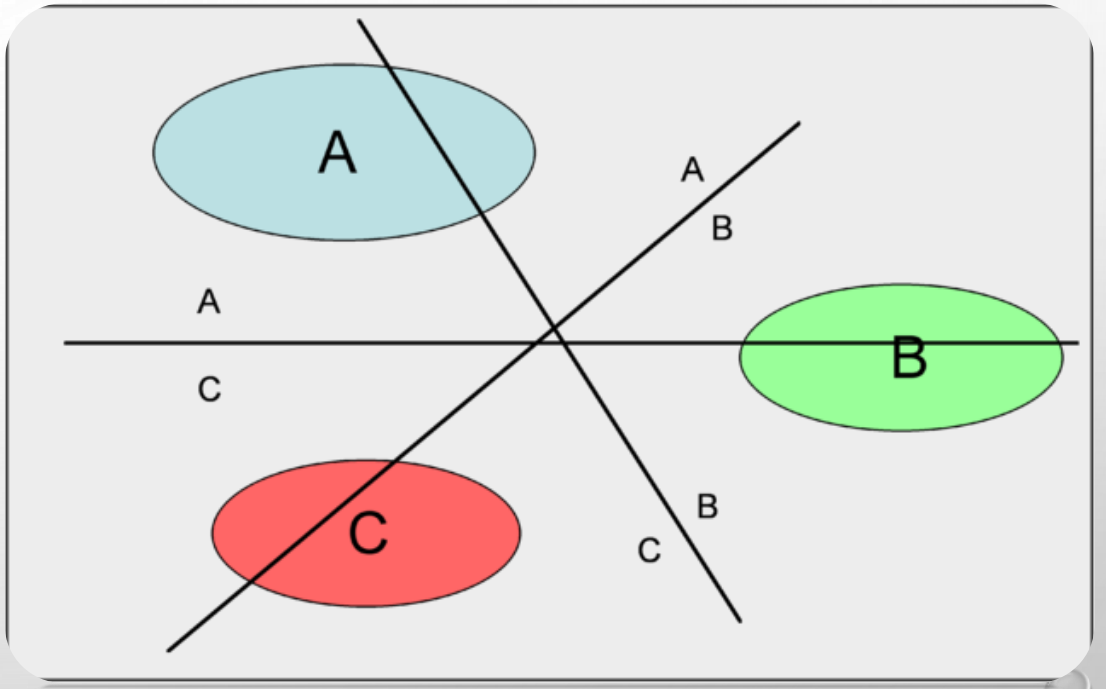
- **Modelos Multiclasse**
  - Discriminar múltiplos objetos em paralelo.
  - Ensembles podem ser utilizados para especializar modelos.
  - Alguns modelos são naturalmente multiclasse, como redes neurais.



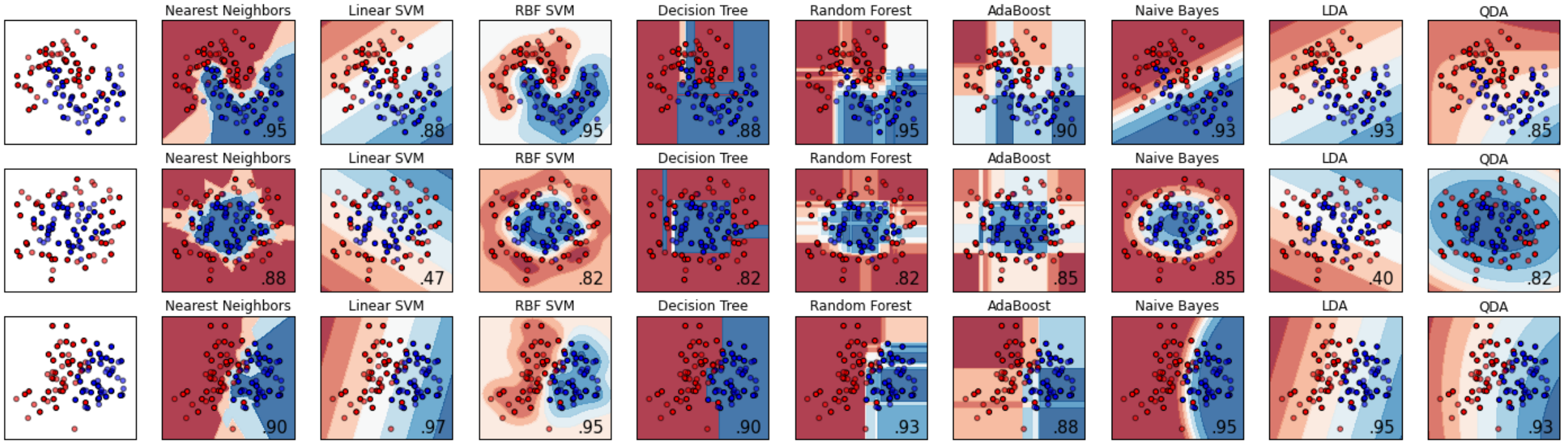
ENSEMBLES BÁSICOS



ONE AGAINST ALL



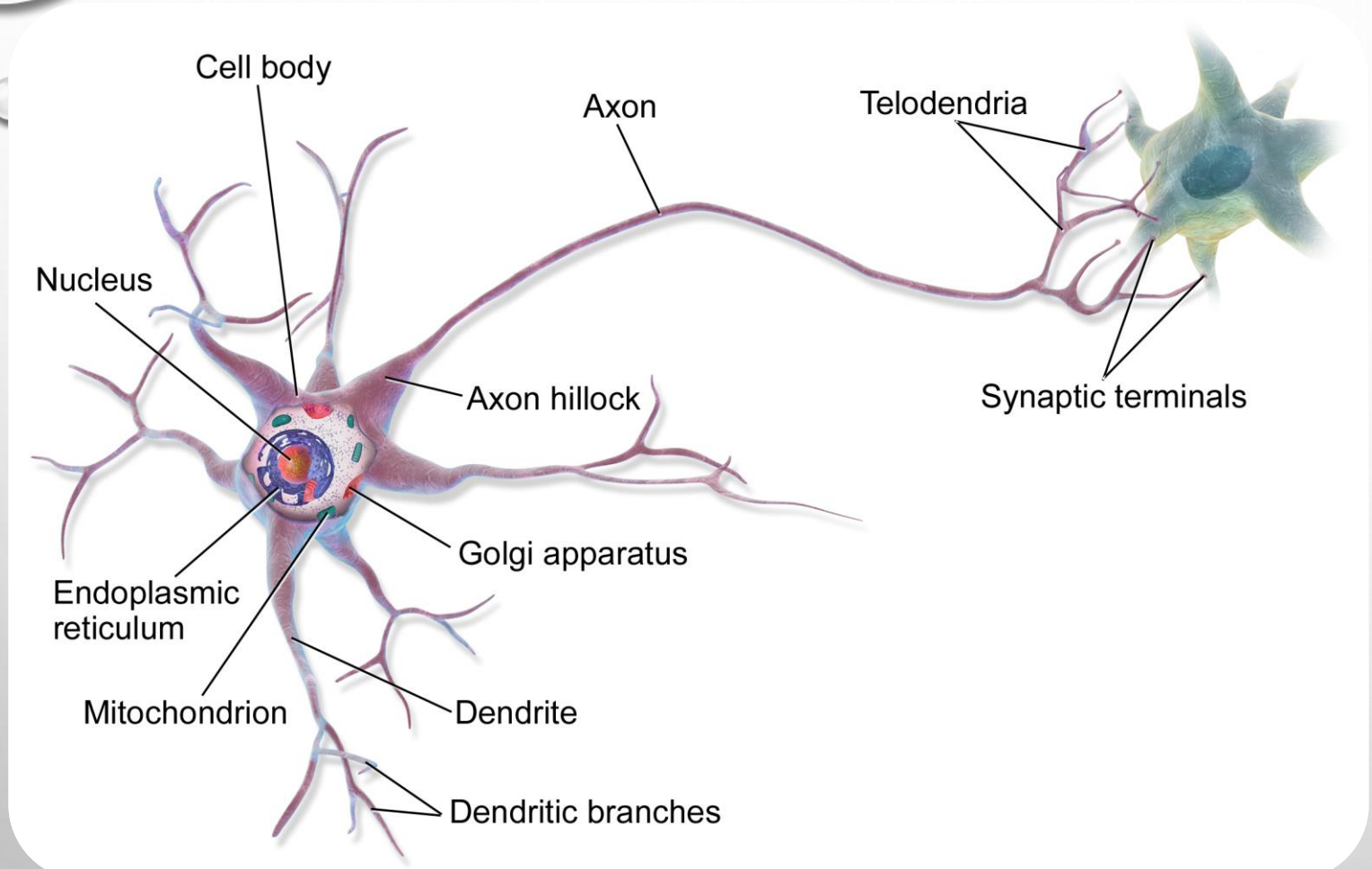
ONE AGAINST ONE


Nearest Neighbors
Linear SVM
RBF SVM
Decision Tree
Random Forest
AdaBoost
Naive Bayes
LDA
QDA
.95
.88
.95
.88
.95
.90
.93
.93
.85
Nearest Neighbors
Linear SVM
RBF SVM
Decision Tree
Random Forest
AdaBoost
Naive Bayes
LDA
QDA
.88
.47
.82
.82
.82
.85
.85
.40
.82
Nearest Neighbors
Linear SVM
RBF SVM
Decision Tree
Random Forest
AdaBoost
Naive Bayes
LDA
QDA
.90
.97
.95
.90
.93
.88
.95
.95
.93

REDES NEURAIS

HISTÓRIA DAS REDES NEURAIS

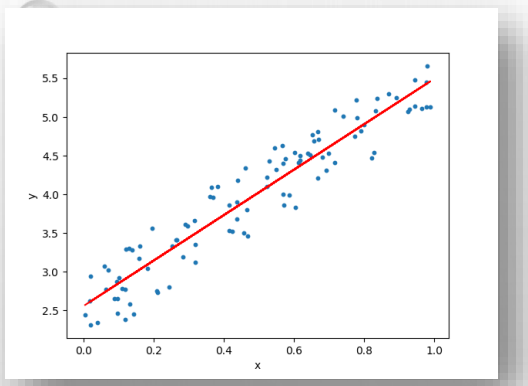
- Perceptron, Rosenblatt 1954
- Neocognitron, Fukushima 1979
- Backpropagation Rumelhart, Hinton & Williams 1986
- LSTM, Hochreiter e Schmidhuber1995
- Keras & Tensorflow, Google 2015



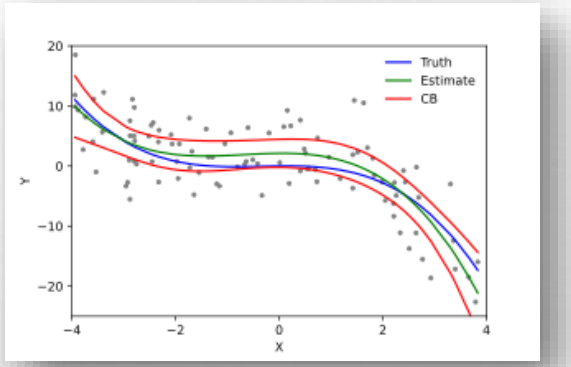
INSPIRAÇÃO BIOLÓGICA

O APROXIMADOR UNIVERSAL

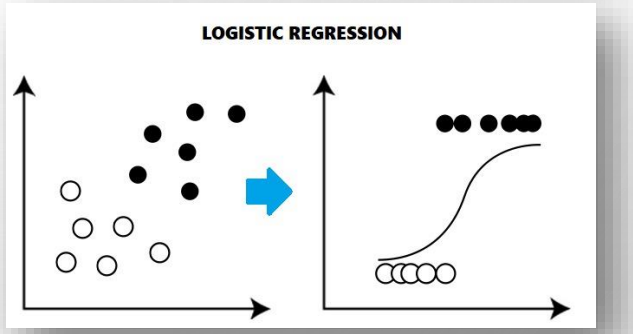
$$Y = \underbrace{F(X)}_{\text{Parte Determinística}} + \underbrace{\varepsilon}_{\text{Parte Estocástica}}$$



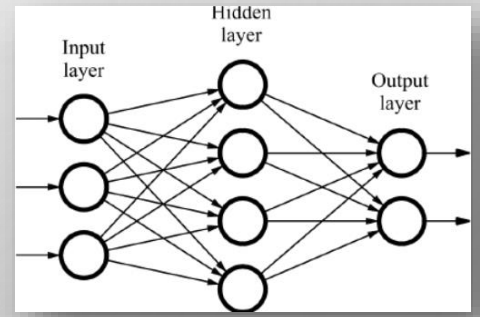
$$Y = \alpha^T x + \varepsilon$$



$$Y = X\alpha + \varepsilon$$

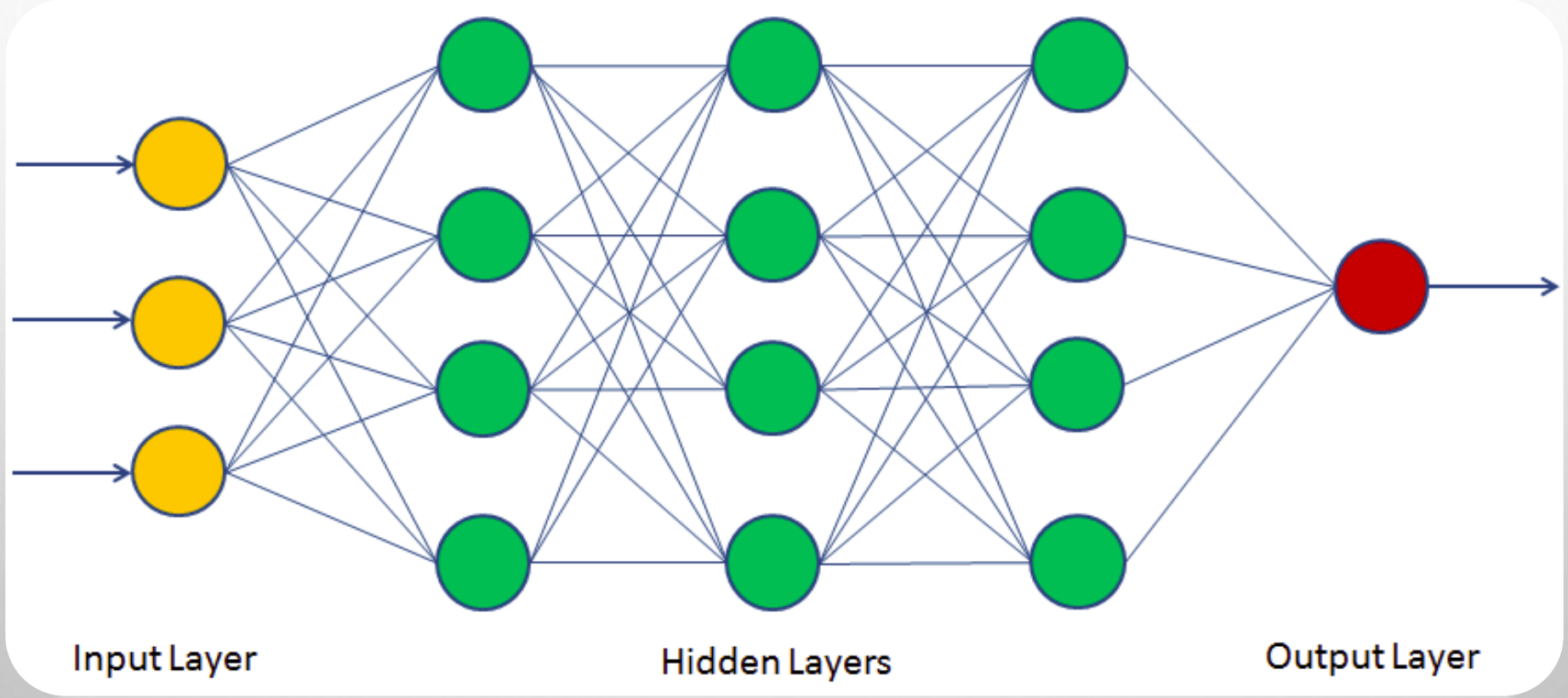


$$Y = \frac{1}{1 + e^{\alpha^t x + \varepsilon}}$$

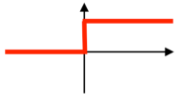
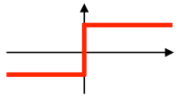
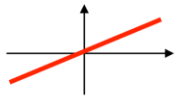
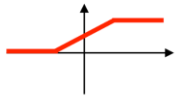
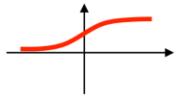
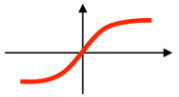
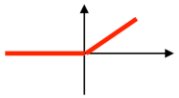
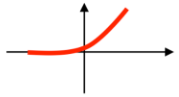


$$Y = \varphi(x) + \varepsilon$$

O APROXIMADOR UNIVERSAL



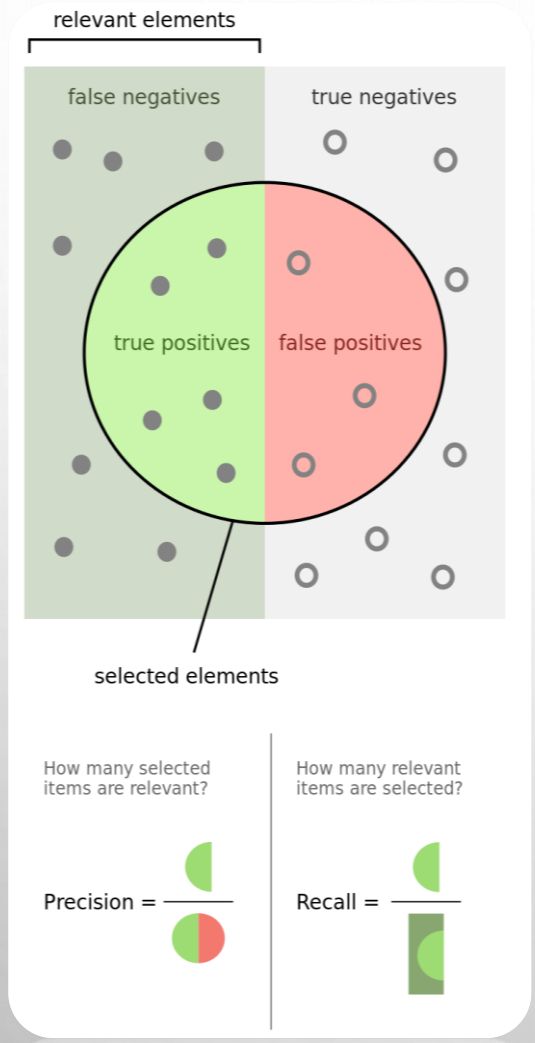
FUNÇÕES DE ATIVAÇÃO

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^{z})$	Multi-layer Neural Networks	

GOOGLE TENSORFLOW PLAYGROUND

EVALUATION

FIGURAS DE MÉRITO CLASSIFICAÇÃO



Acurácia

- (TP+TN)/(P+N)

Taxa de Erro

- 1-Acurácia

Sensibilidade (Recall)

- TP/(TP+FN)

Especificidade

- TN/(TN+FP)

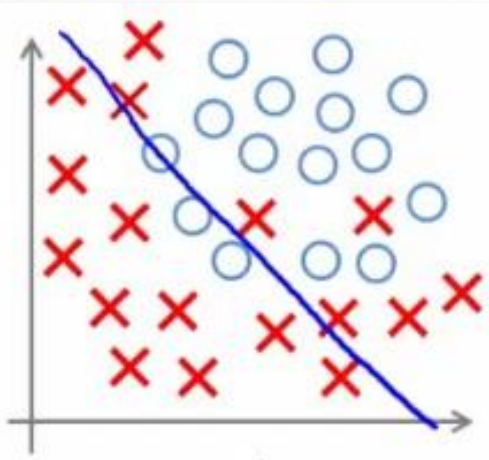
Precisão

- TP/(TP+FP)

Produto Sp

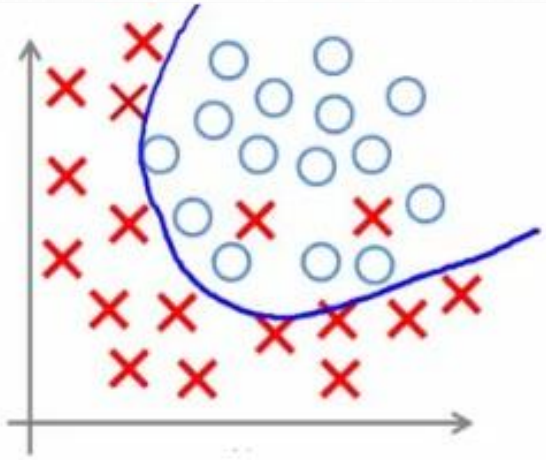
- SQRT[SQRT(R1*R2) * (R1 + R2)/2]

CAPACIDADE E GENERALIZAÇÃO

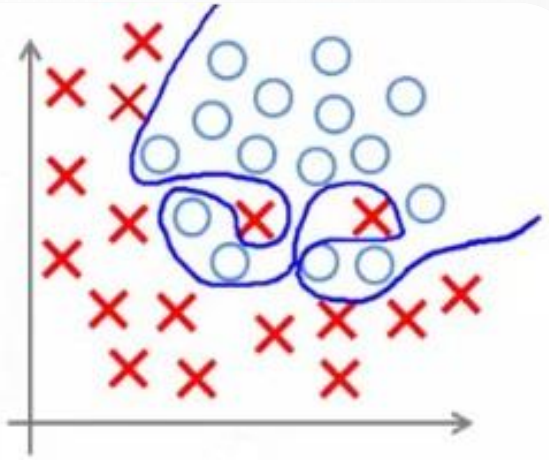


Under-fitting

(too simple to explain the variance)



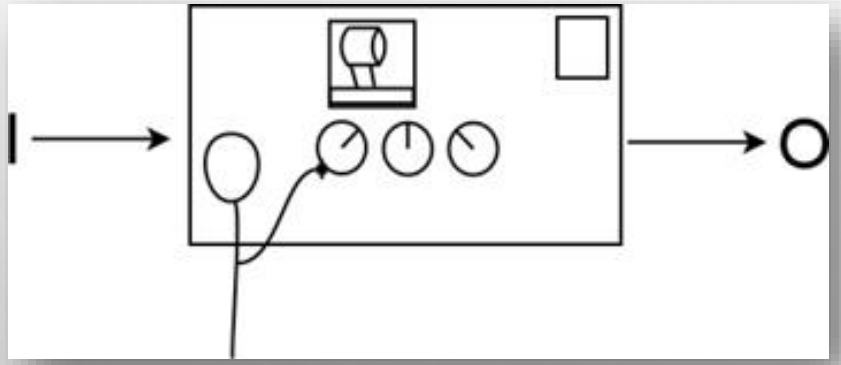
Appropriate-fitting

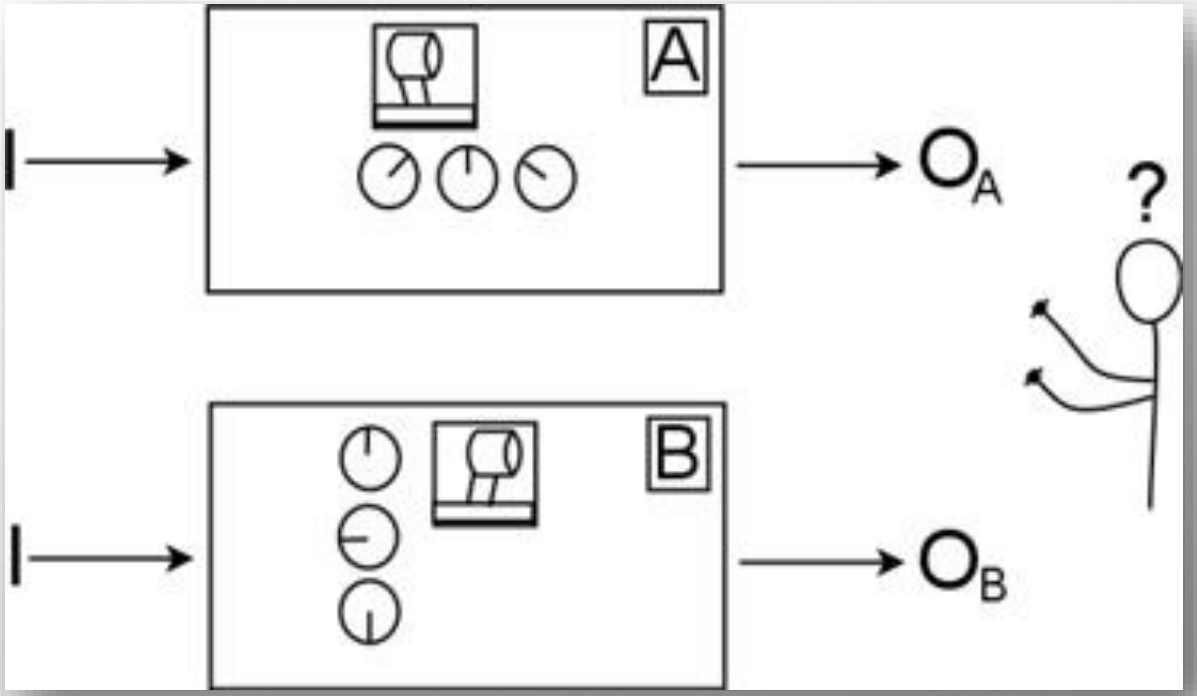


Over-fitting

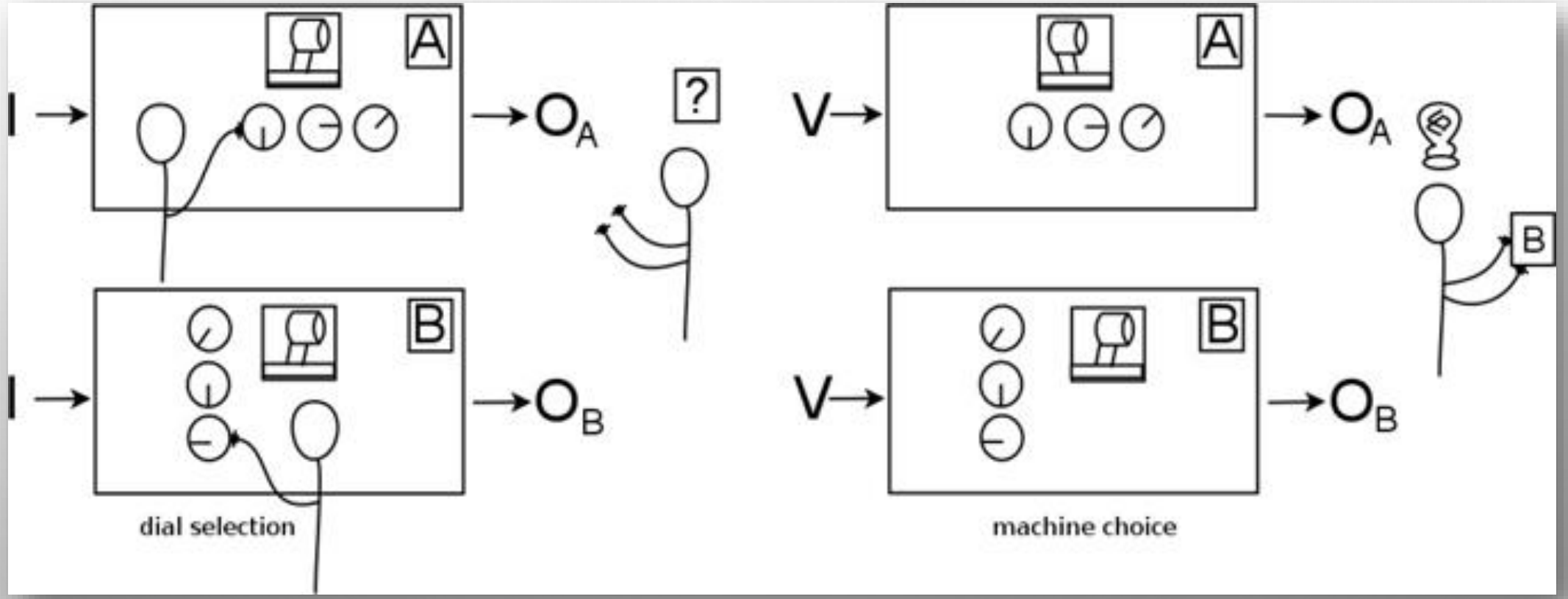
(forcefitting -- too good to be true)

PARÂMETROS E HIPERPARÂMETROS





PARÂMETROS E HIPERPARÂMETROS



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.

SINGLE SPLIT (GRUPO DE CONTROLE)

- Amostra é dividida entre treino e teste, mantendo um percentual das observações como grupo de teste externo ao treinamento.

K FOLDS

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

SHUFFLESPLIT

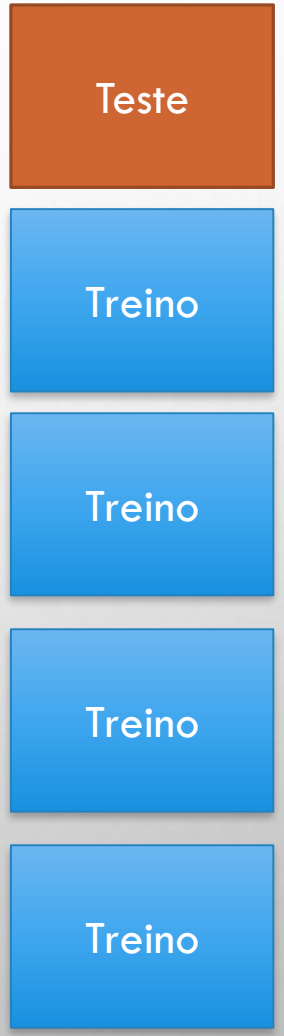
- Amostra é dividida em M conjuntos / treino e teste obedecendo uma proporção.

BOOTSTRAPPING

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

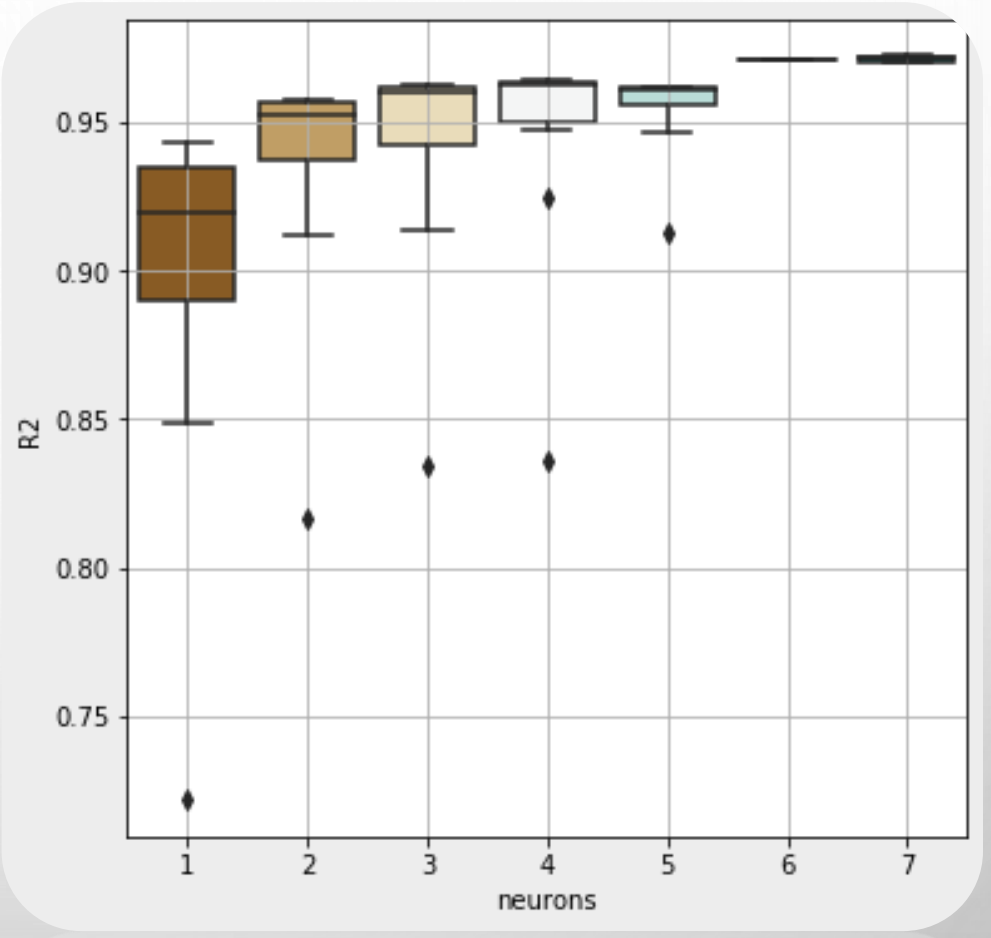
TREINAMENTO K FOLDS

- TREINAMENTO UTILIZANDO K PARTIÇÕES, COM DADOS DAS CLASSES BALANCEADOS.
- CADA TREINAMENTO É REALIZADO PARA EXPLORAR UMA CONFIGURAÇÃO DE HIPERPARÂMETROS DO MODELO.
- 4 PARTIÇÕES SÃO USADAS PARA TREINAR O MODELO, 1 PARTIÇÃO É UTILIZADA PARA MENSURAR O DESEMPENHO FORA DA AMOSTRA (GENERALIZAÇÃO).
- UMA ESTATÍSTICA DA FIGURA DE MÉRITO É SELECIONADA PARA MEDIR A QUALIDADE DE CADA CONFIGURAÇÃO DE HIPERPARÂMETRO.



K-FOLDS - EXEMPLO

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



PARTE 2 : PRÁTICA

AMBIENTE DE DESENVOLVIMENTO

Miniconda

Miniconda is a free minimal installer for conda. It is a small bootstrap version of Anaconda that includes only conda, Python, the packages they both depend on, and a small number of other useful packages (like pip, zlib, and a few others). If you need more packages, use the `conda install` command to install from thousands of packages available by default in Anaconda's public repo, or from other channels, like conda-forge or bioconda.

Is Miniconda the right conda install for you? The Anaconda or Miniconda page lists some reasons why you might want one installation over the other.

System requirements

Latest Miniconda installer links by Python version

Installing Miniconda

Miniconda release notes

Other resources

Miniconda hash information

Simple Python Version Management: pyenv

gitter join chat

pyenv lets you easily switch between multiple versions of Python. It's simple, unobtrusive, and follows the UNIX tradition of single-purpose tools that do one thing well.

This project was forked from rbenv and ruby-build, and modified for Python.

```
$ pyenv versions
  2.7.10
* 3.5.0 (set by /Users/yuu/.pyenv/version)
  miniconda3-3.16.0
  pypy-2.6.0

$ python --version
Python 3.5.0

$ pyenv global pypy-2.6.0

$ python --version
Python 2.7.9 (295ee98b69288471b0fcf2e0ede82ce5209eb90b, Jun 01 2015, 17:30:13)
[PyPy 2.6.0 with GCC 4.9.2]

$ cd /Volumes/treasuredata/jupyter

$ pyenv version
miniconda3-3.16.0 (set by /Volumes/treasuredata/.python-version)

$ python --version
Python 3.4.3 :: Continuum Analytics, Inc.
```

Boa Prática: 1 Ambiente por projeto!

CRIANDO AMBIENTE COM CONDA

Managing environments

With conda, you can create, export, list, remove, and update environments that have different versions of Python and/or packages installed in them. Switching or moving between environments is called activating the environment. You can also share an environment file.

There are many options available for the commands described on this page. For a detailed reference on all available commands, see commands.

Creating an environment with commands

Use the terminal for the following steps:

1. To create an environment:

```
conda create --name <my-env>
```

   Replace `<my-env>` with the name of your environment.

2. When conda asks you to proceed, type `y`:

```
proceed ([y]/n)?
```

   This creates the myenv environment in `/envs/`. No packages will be installed in this environment.

3. To create an environment with a specific version of Python:

```
conda create -n myenv python=3.9
```

Activating an environment

Activating environments is essential to making the software in the environments work well. Activation entails two primary functions: adding entries to PATH for the environment and running any activation scripts that the environment may contain. These activation scripts are how packages can set arbitrary environment variables that may be necessary for their operation. You can also use the config API to set environment variables.

Activation prepends to PATH. This only takes effect when you have the environment active so it is local to a terminal session, not global.

Note

When installing Anaconda, you have the option to "Add Anaconda to my PATH environment variable." *This is not recommended* because it *appends* Anaconda to PATH. When the installer appends to PATH, it does not call the activation scripts.

Note

On Windows, PATH is composed of two parts, the *system* PATH and the *user* PATH. The system PATH always comes first. When you install Anaconda for "Just Me", we add it to the *user* PATH. When you install for "All Users", we add it to the *system* PATH. In the former case, you can end up with system PATH values taking precedence over your entries. In the latter case, you do not. *We do not recommend* multi-user installs.

To activate an environment: `conda activate myenv`

https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

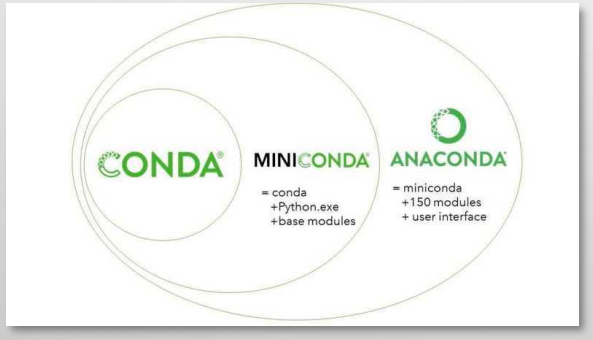
AMBIENTE PYTHON

5. Visualização

6. Machine Learning

4. Variáveis Aleatórias

1. Editor de Código



2. Gestor de Ambiente

3. Ambiente Python do Projeto

3. Notebook Dinâmico

PRÓXIMA AULA: CLASSIFICAÇÃO