Aprendizagem Automática em Sistemas Empresariais

PEDRO PEREIRA

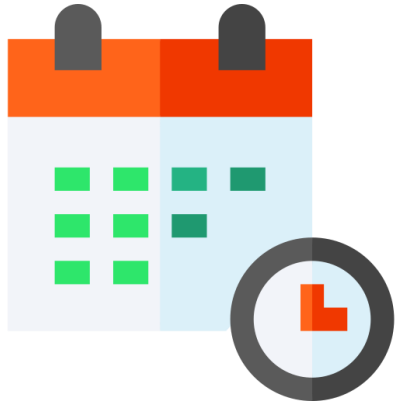
AULA 5

Agenda

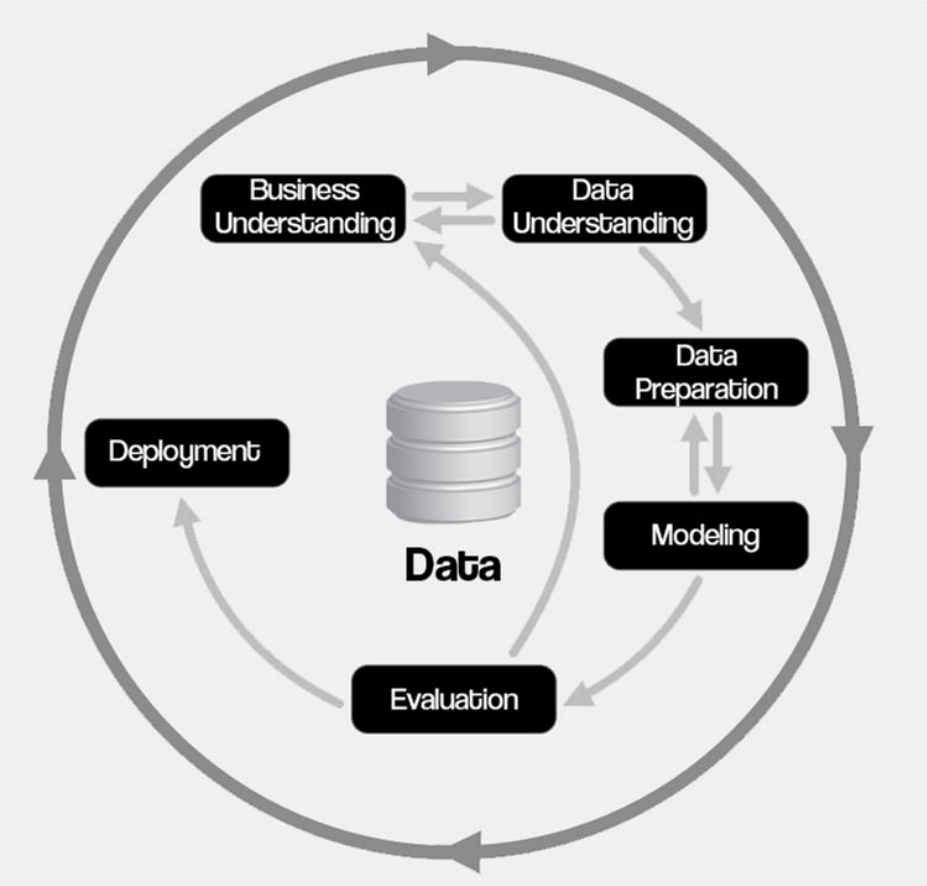
CRISP-DM: Modelação e Avaliação – Parte 1

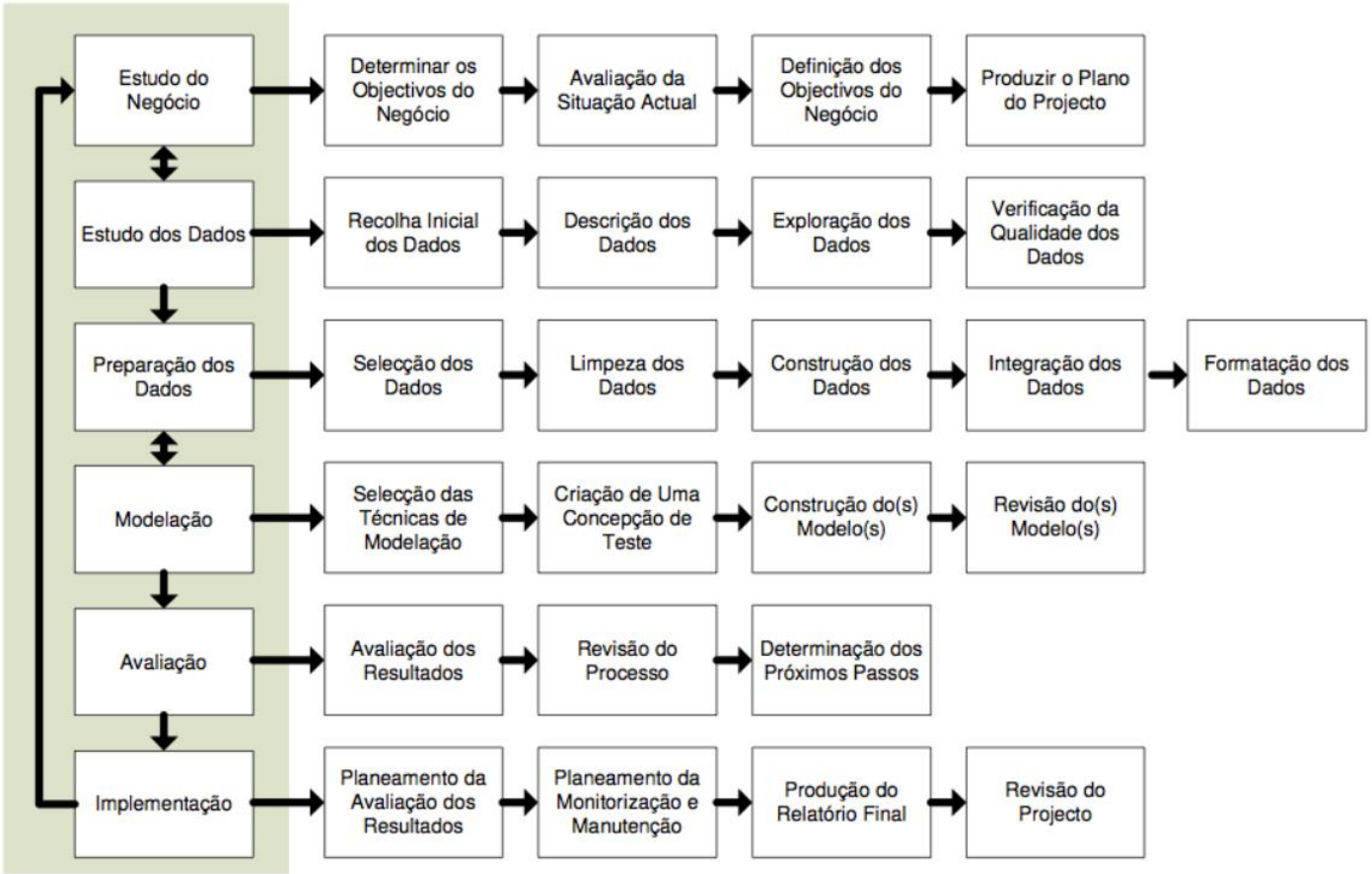
- Classificação
- Algoritmos de ML para classificação
- Métricas de classificação

Acompanhamento ao projeto

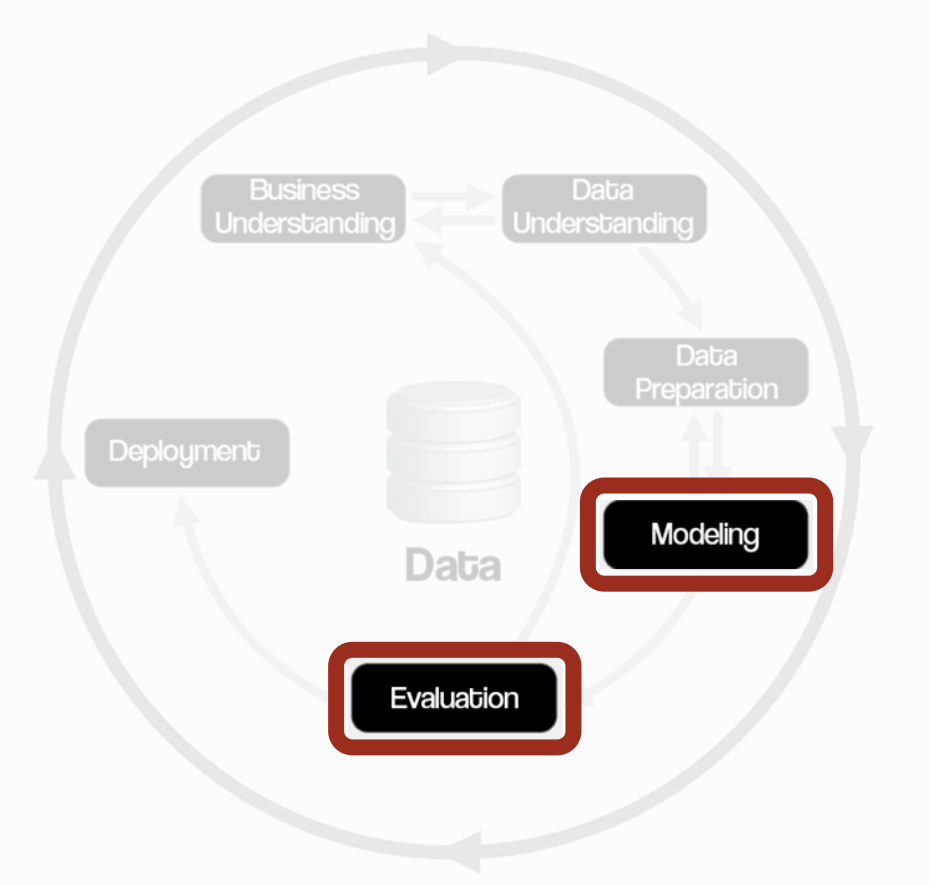


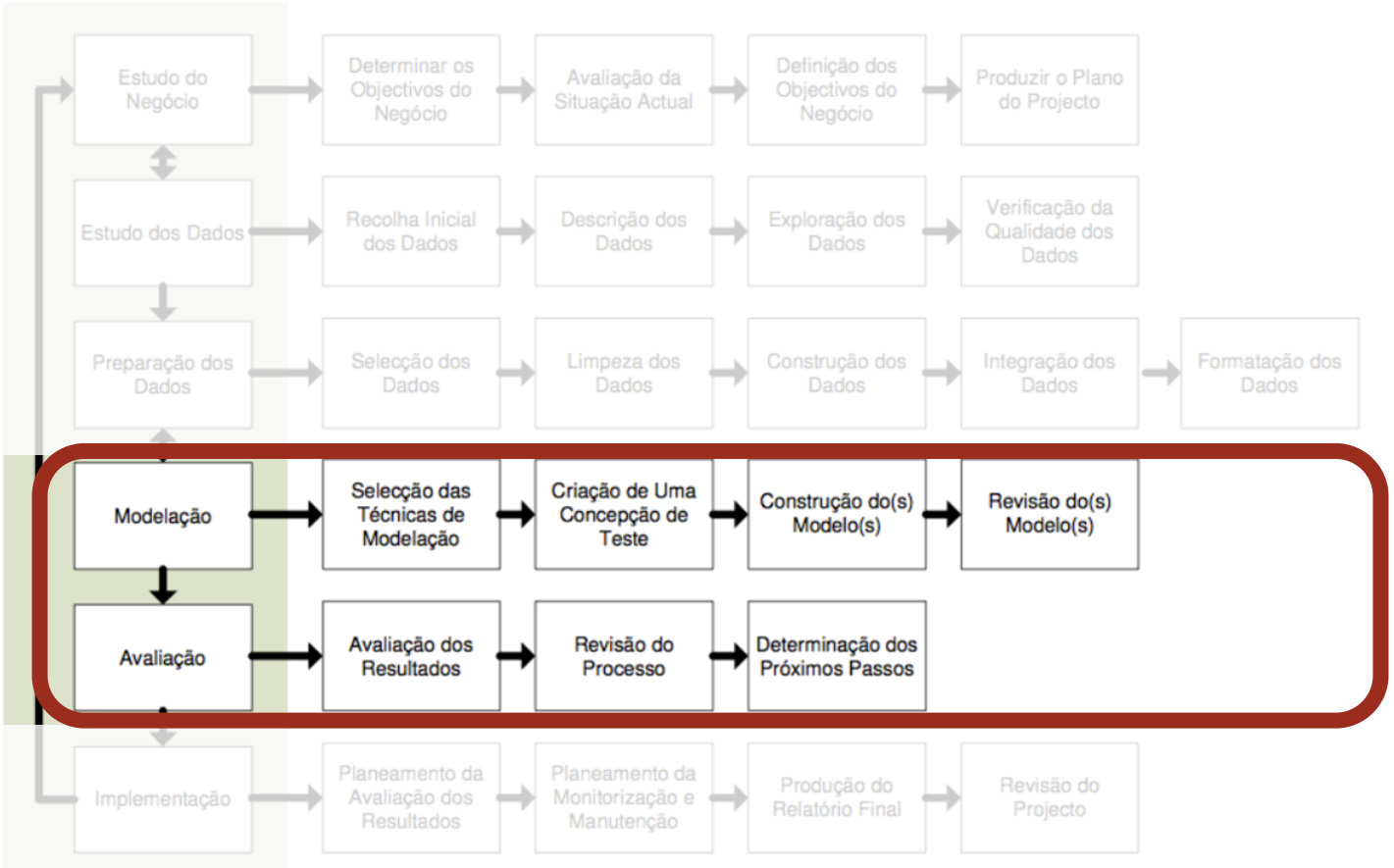
Cross Industry Process for Data Mining (CRISP-DM)



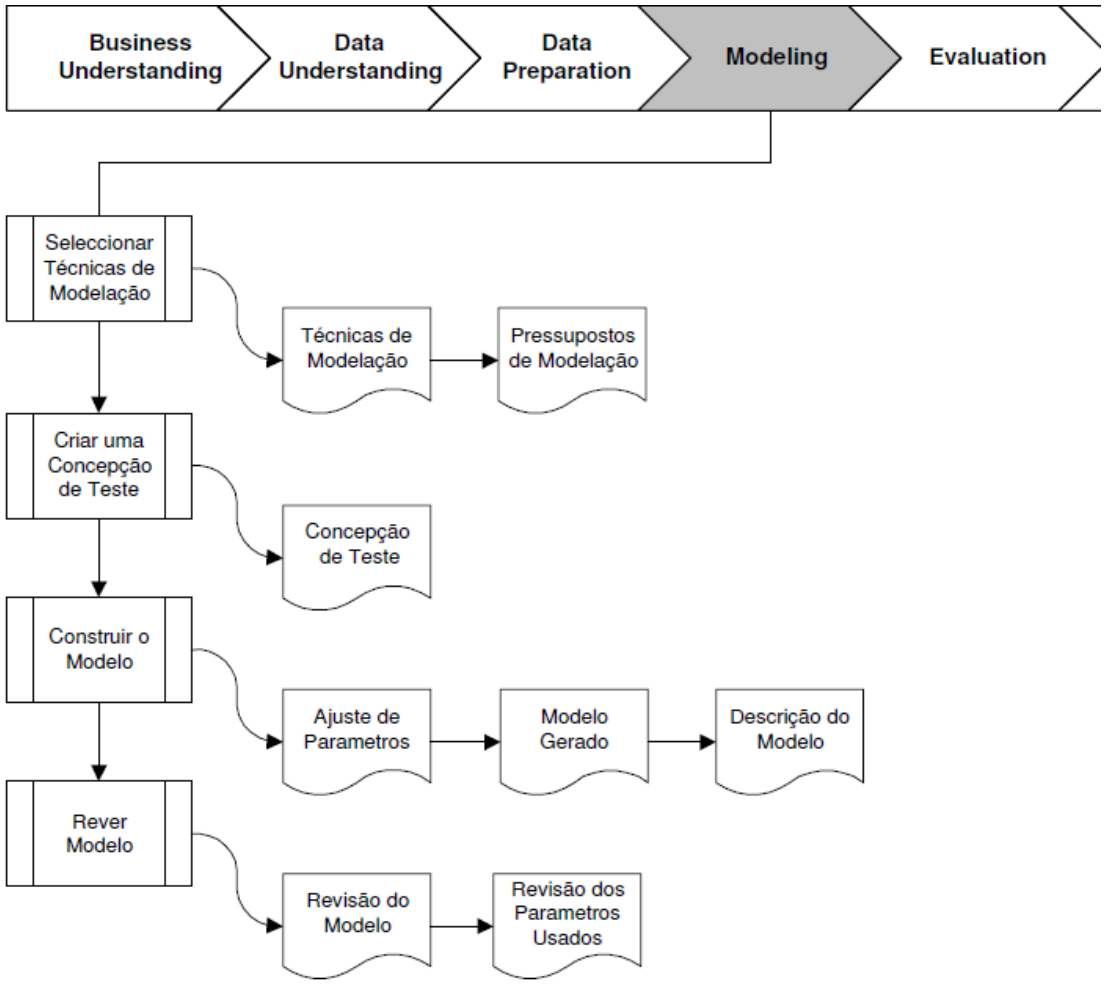


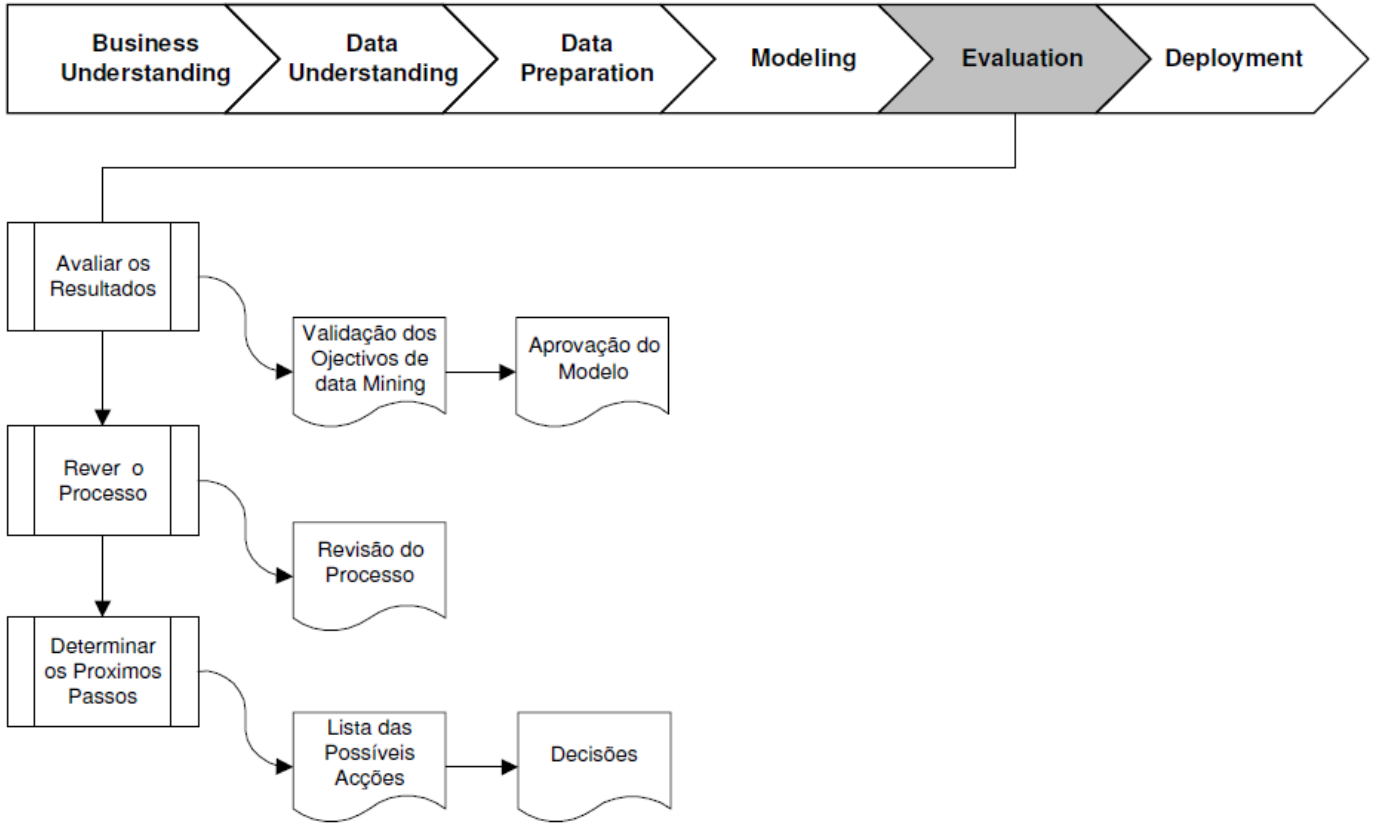
CRISP-DM – Modelação e Avaliação – Parte 1





CRISP-DM – Atividades da Modelação e Avaliação





CRISP-DM – Modelação (classificação)

Classificação – prever o valor de uma variável categórica (classe) a partir de diferente variáveis independentes.

Diferentes variantes na classificação:

- ***Output*** pode ser uma **classe** ({A, B, C}) ou uma **probabilidade** ({0.6, 0.3, 0.1}).
- **Número de classes** determina se é uma tarefa de classificação **binária** (duas classes: {aprovado, reprovado}) ou ***multiclass*** (3 ou mais: {vitória, empate, derrota}).
- **Quantidade de outputs** determina se é uma tarefa de **classificação** (1 *output*) ou **classificação *multi-label*** (2 ou mais *outputs*).

Modelação – Algoritmos de classificação

Regras de Classificação;

Linear Discriminant Analysis (LDA);

K-Nearest Neighbors (KNN);

Regressão Logística;

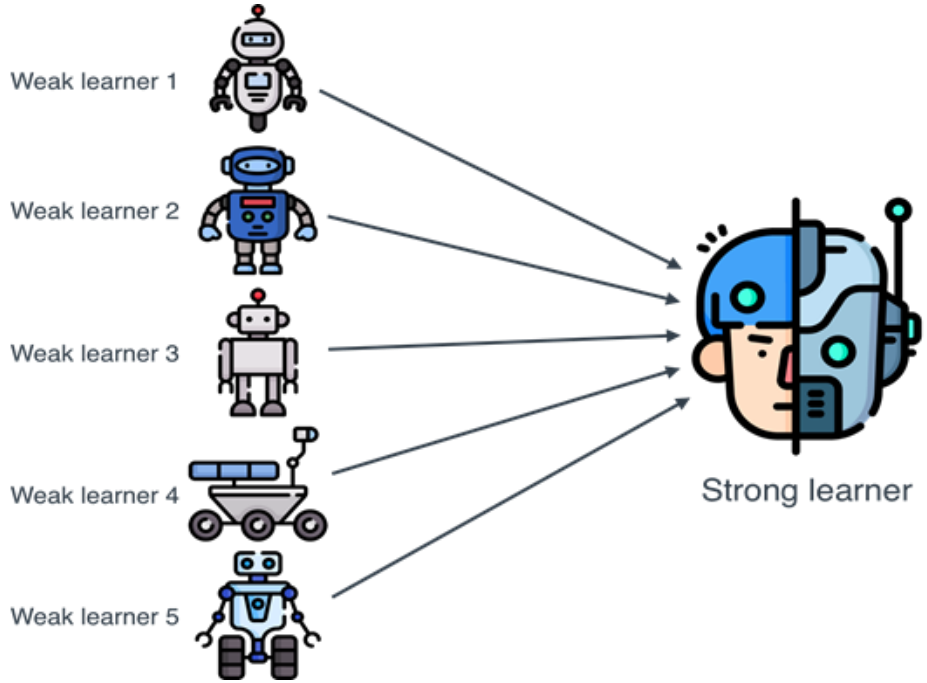
Árvores de Decisão (DT);

Máquinas de Vetor de Suporte (SVM);

Redes Neuronais Artificiais (ANN);

Ensembles (XGBoost, AdaBoost, *Random Forest*, …);

***Automated Machine Learning* (AutoML)**.



Árvores de Decisão

Dos modelos mais utilizados em *Data Mining*, nomeadamente em tarefas de classificação.

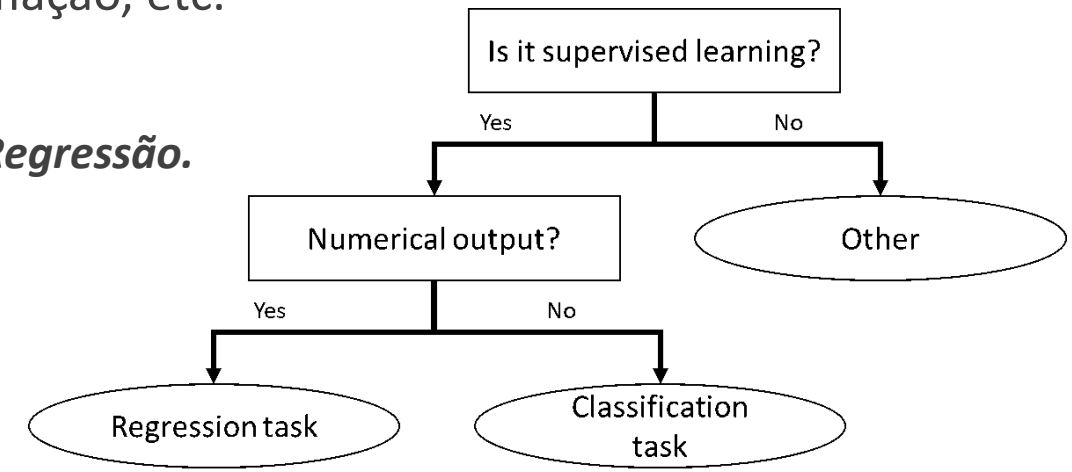
Por norma, facilmente interpretáveis pelo ser humano *(White-box models)*.

Constituídas por um conjunto de **nodos** (divisões de dados), **ramos** (decisões) e **folhas** *(output)*.

Existem vários algoritmos para criação de árvores de classificação (ex.: CART, ID3, C4.5) com base em critérios como a entropia, índice de GINI, ganho de informação, etc.

É possível extrair regras:

- Se ***Supervised_learning*** e ***numerical_output*** então ***Tarefa=Regressão.***



Ferramentas:

- Python: https://scikit-learn.org/stable/modules/tree.html.
- R: https://www.r-bloggers.com/2021/04/decision-trees-in-r/.
- Rapidminer: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/trees/parallel_decision_tree.html.

Redes Neuronais Artificiais

Modelos matemáticos que procuram **replicar o sistema de processamento do cérebro humano**.

Capazes de “memorizar” e aprender com a experiência.

Compostas por uma camada de **entrada** *(input),* uma de **saída** *(output)* e uma ou mais camadas **ocultas** *(hidden)*.

Cada camada tem um conjunto de neurónios conectados entre si através de ligações que carregam pesos (valores numéricos).

O treino de uma rede neuronal passa pelo ajuste destes valores numéricos com base numa função de erro (ex.: *MSE*, *cross entropy*) usando um algoritmo optimizador (ex.: *backpropagation, Adam*).

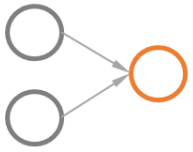
As redes neuronais podem ser aplicadas a diferentes tipos de *Machine Learning (supervised,* semi-supervised, unsupervised ou *reinforcement*).

Redes Neuronais Artificiais

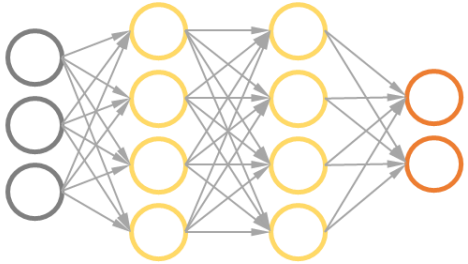
As redes neuronais artificiais podem ter diversas arquiteturas/topologias, tipicamente definidas pelo utilizador com base na sua experiência e conhecimento.

Atualmente, a rede neuronal mais popular é o *Deep Learning*, que pode conter milhares de camadas e de neurónios!

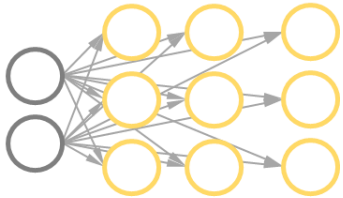
Single Layer Perceptron



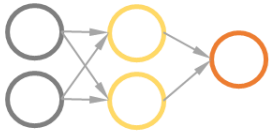
Multi Layer Perceptron



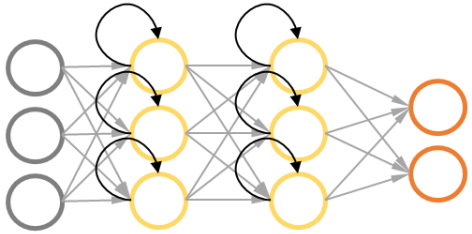
Self-Organized Map



Radial Basis Function

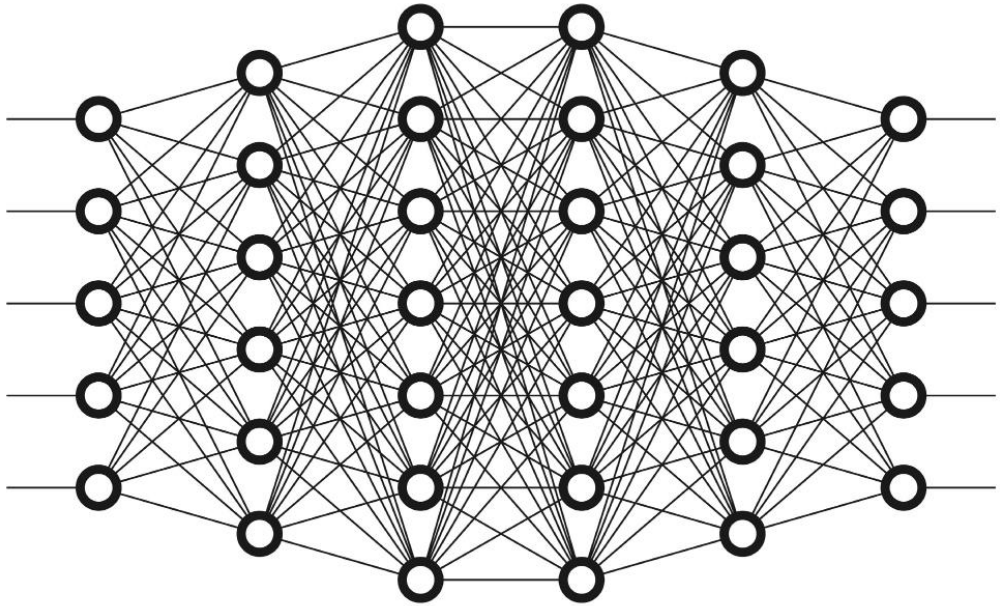


Recurrent NN



Boltzmann Machine

Deep Learning



Redes Neuronais Artificiais

Ferramentas:

- Python: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html
- R: https://rdrr.io/cran/rminer/man/fit.html
- Rapidminer: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/neural_nets/neural_net.html
- Tensorflow (Python e R): https://www.tensorflow.org/learn
- Keras (Python e R): https://keras.io/

Ensembles

Conjuntos de modelos individuais que são agregados de modo a obter-se uma resposta única.

Pode utilizar-se a média ou moda dos *outputs* dos modelos individuais para se obter um *output* único.

***Random Forest*:** proposto em 2001; agrega **centenas de árvores de decisão;** muito popular.

***Extreme Gradient Boosting (XGBoost)*:** semelhante ao *random forest*, mas com um processo de treino diferente; por norma agrega um número superior de árvores de decisão (milhares!).

AdaBoost, Extremely Randomized Trees, …..

Ferramentas:

- Python: https://scikit-learn.org/stable/modules/ensemble.html; https://xgboost.readthedocs.io/.
- R: https://daviddalpiaz.github.io/r4sl/ensemble-methods.html#classification-2.
- Rapidminer: https://towardsdatascience.com/how-to-create-ensemble-models-using-rapid-miner-72a12160fa51.
- Weka: https://machinelearningmastery.com/use-ensemble-machine-learning-algorithms-weka/.

Hiper-parâmetros

Os modelos de *Machine Learning* têm um conjunto de parâmetros que podem influenciar a sua aprendizagem. Exemplos:

- **Árvores de Decisão:** profundidade máxima (*max_depth),* influencia o seu crescimento.
- ***Random Forest:*** número de árvores *(n_estimators)*, por quantas árvores é composto.
- **Redes Neuronais:** número de camadas (arquitetura), otimizador (algoritmo de ajuste dos pesos),…

Como escolher os melhores valores?

- Usar os **valores padrão** (*default*) das implementações que usamos;
- Afinação destes valores por **tentativa-erro** (convém saber o que estamos a fazer!);
- Usar um **método de procura** para afinar estes valores por nós (ex.: *grid-search, automated search*, algoritmos genéticos,…).

Importante: não usar os dados de teste para tomar decisões!!!

Como automatizar a escolha do **melhor modelo** e dos **melhores hiper-parâmetros**??

AutoML – Automated Machine Learning

Não existe uma configuração única modelo ML + hiper-parâmetros que resulte para todos os problemas!

Dependendo da tarefa ML, dos atributos, da complexidade do problema, etc., há modelos e hiper-parâmetros que podem resultar melhor que outros.

AutoML automatiza o processo de **escolha e afinação de modelos**, testando diferentes configurações e retornando a melhor, com base em determinada métrica.

Em alguns casos, o AutoML também é usado para escolher o melhor pré-processamento de dados.

Este processo é mais **custoso** computacionalmente, visto que são testadas várias combinações.

Ferramentas: H2O, AutoGluon, Auto-Keras (apenas redes neuronais), Auto-Weka, rminer (em R), auto-sklearn, Rapidminer Auto-model, TransmogrifAI, …

CRISP-DM – Avaliação (classificação)

Avaliação de modelos deve ser feita de forma **objetiva** → uso de métricas para medir a qualidade das previsões.

2 tipos de avaliação:

- **Interna –** medida nos dados de **treino**.
- **Externa –** medida nos dados de **teste** (não utilizados no treino do modelo); serve para medir a capacidade de generalização dos modelos.

Métricas de classificação:

- *Accuracy;*
- Sensibilidade *(recall);*
- Especificidade *(precision);*
- AUC (*Area Under the* ROC *Curve*);
- F1-score, log *loss*, …

Accuracy:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

Recall:

$$Recall = \frac{TP}{TP + FN}$$

Precision:

$$Precision = \frac{TP}{TP + FP}$$

F_1 score:

$$F_1 = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

Predicted Value	Actual Value: Positive	Actual Value: Negative
Positive	TP (True Positive)	FP (False Positive)
Negative	FN (False Negative)	TN (True Negative)

CRISP-DM – Avaliação (classificação)

Curva ROC: é útil quando temos uma probabilidade ***p*** associada a cada classe.

Definindo um *threshold* ***th*** (entre 0 e 1), definimos que a classe positiva é prevista pelo modelo quando ***p > th***.

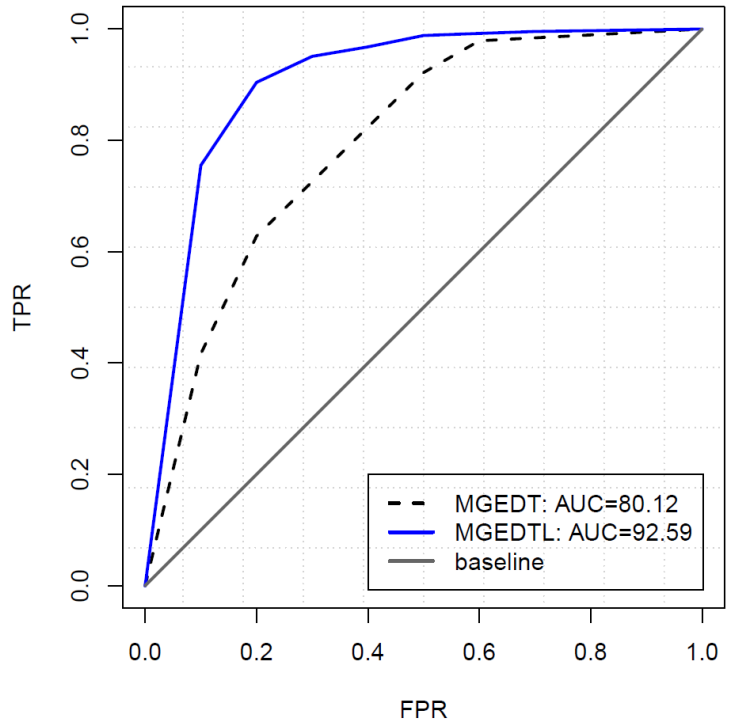
A curva ROC mostra o desempenho do modelo para todos os valores de ***th***.

A curva denota 1−Especificidade (FPR, eixo dos x) vs. a Sensibilidade (TPR);

Ajustando o valor de ***th*** podemos escolher ter um modelo mais **sensível (*th* menor)** ou um modelo mais **específico (*th* maior).**

O desempenho global é dado pelo **AUC (*Area Under the Curve*).**

Um classificador **aleatório** tem AUC de 0.5 (50%), enquanto um classificador **perfeito** tem AUC de 1 (100%).



CRISP-DM – Avaliação (classificação)

Ferramentas:

- Python: https://scikit-learn.org/stable/modules/model_evaluation.html
- R: https://www.rdocumentation.org/packages/rminer/versions/1.4.6/topics/mmetric
- Rapidminer: https://docs.rapidminer.com/latest/studio/operators/validation/performance/predictive/performance_classification.html

Avaliação: validação de modelos

A validação de modelos pretende "estimar" a sua capacidade de generalização, medindo a sua qualidade/desempenho.

Como tal, as métricas **não podem ser calculadas utilizando dados que o modelo já "viu".**

Holdout: divisão dos dados em dois conjuntos exclusivos, através de uma amostragem aleatória.

- **Treino:** tipicamente **2/3 do conjunto dos dados**, usado para treinar modelos e tomar decisões (melhor modelo, melhores hiper-parâmetros, melhor pré-processamento,…). Por vezes, este conjunto é subdividido em 2 conjuntos (**treino** e **validação**) para verificar decisões internas do modelo.
- **Teste:** tipicamente **1/3 do conjunto dos dados**, é utilizado para avaliar as capacidades do modelo.

Ferramentas:

- Python: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
- R: https://rdrr.io/cran/rminer/man/holdout.html
- Rapidminer: https://docs.rapidminer.com/latest/studio/operators/blending/examples/sampling/split_data.html

Aprendizagem Automática em Sistemas Empresariais

PEDRO PEREIRA

AULA 5