Aprendizagem Automática em Sistemas Empresariais

PEDRO PEREIRA

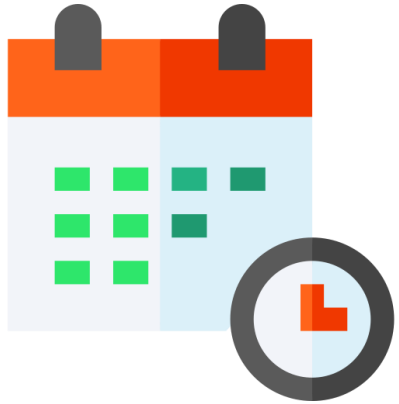
AULA 4

Agenda

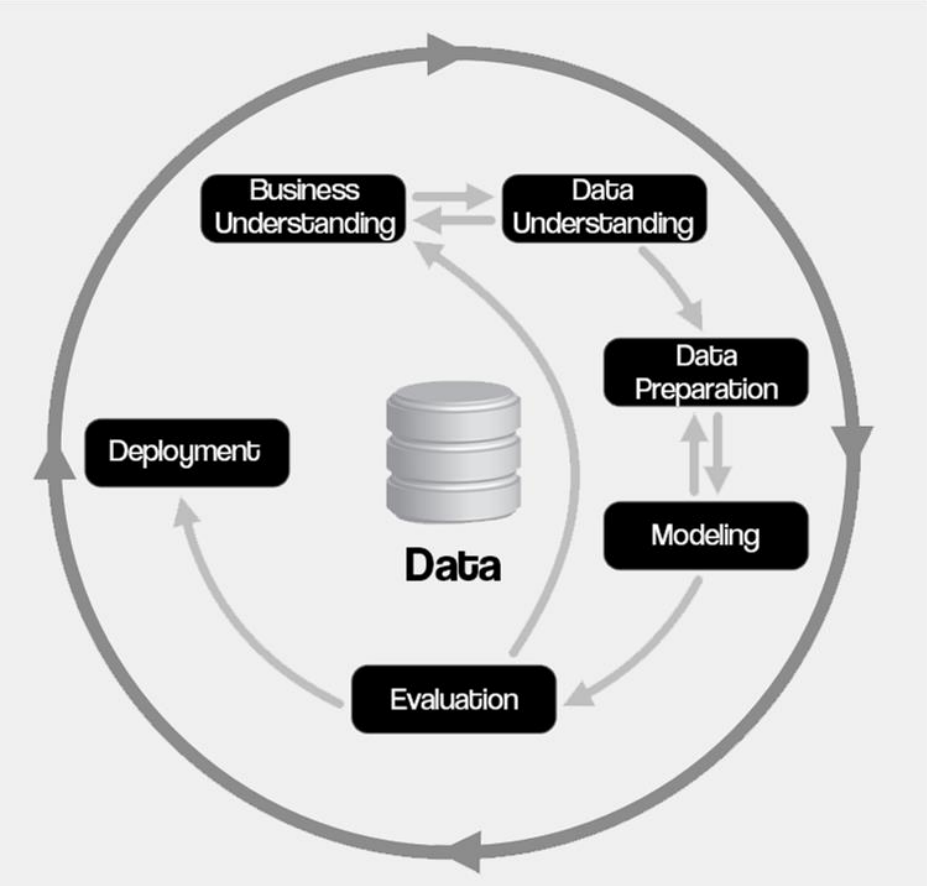
CRISP-DM: Preparação dos Dados

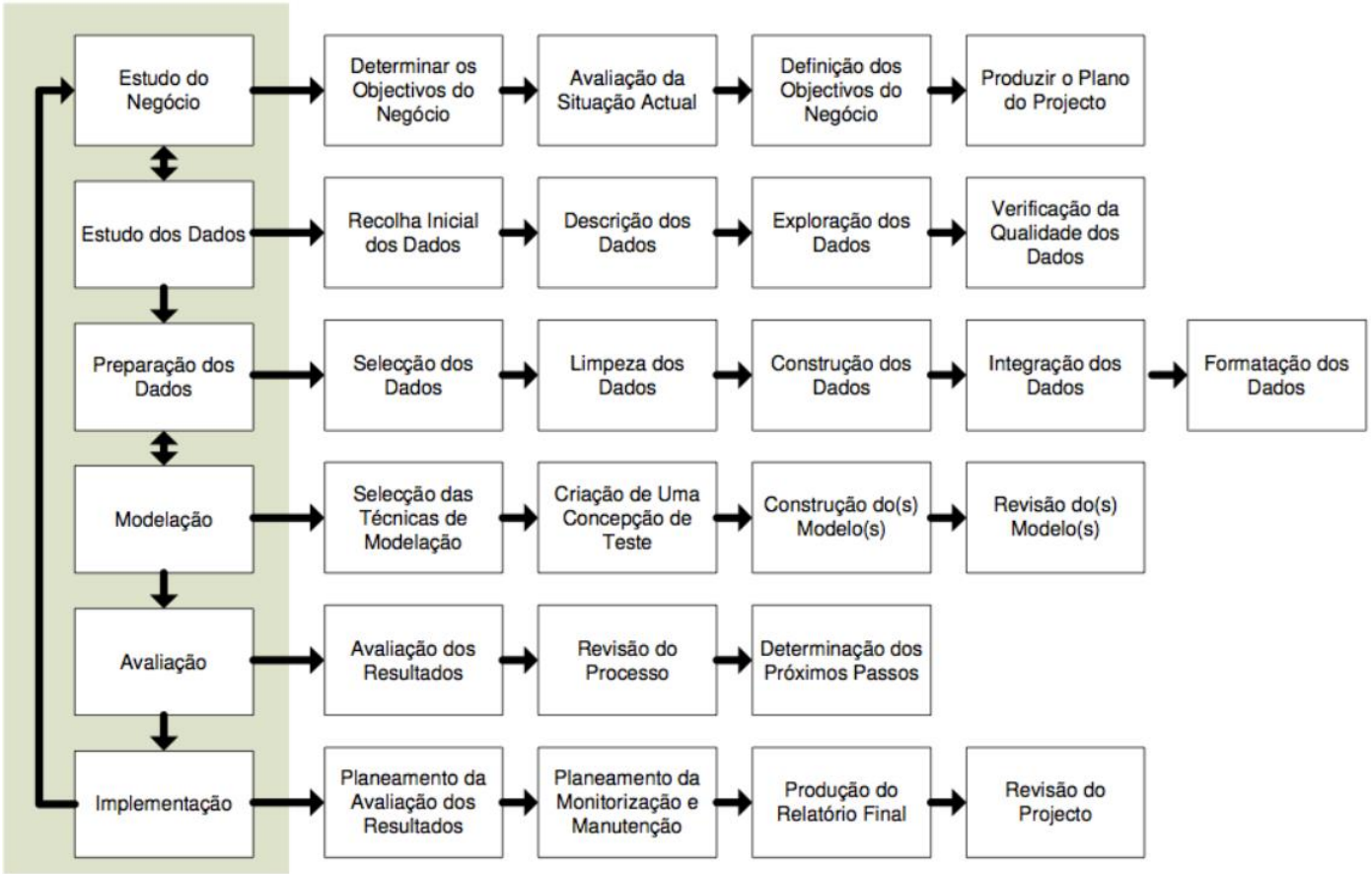
- Valores Omissos
- *Outliers*
- Transformações (numéricas e categóricas)
- Seleção de Atributos
- Dados Desbalanceados

Acompanhamento ao projeto

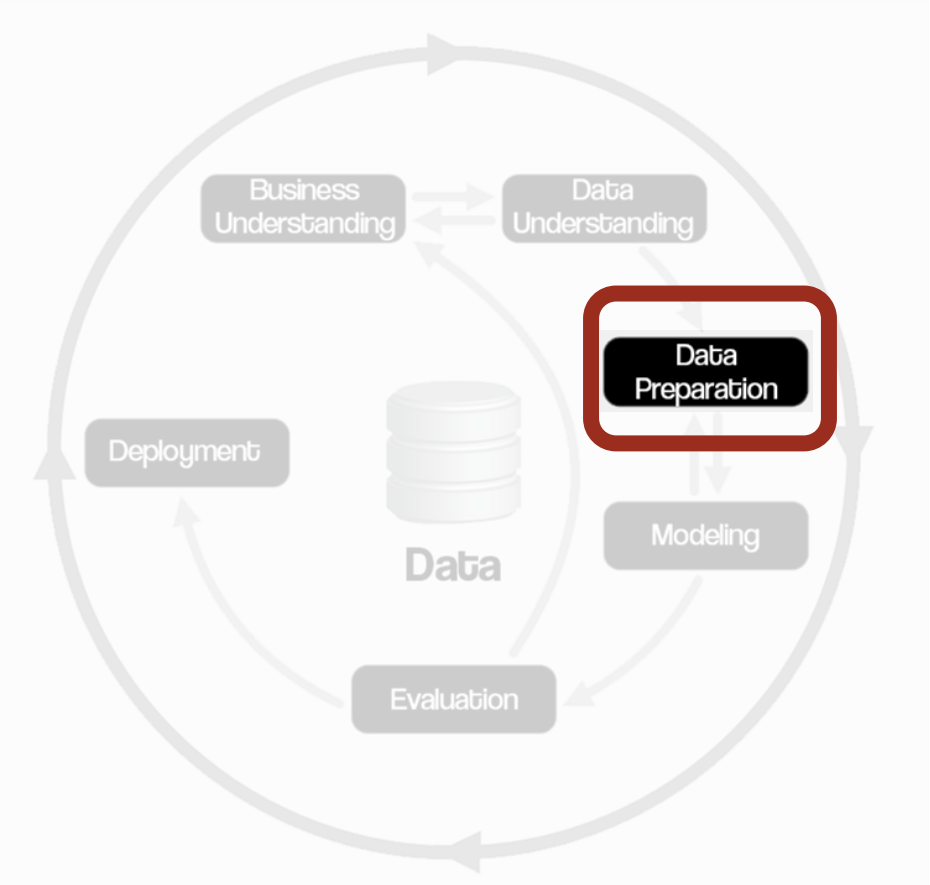


Cross Industry Process for Data Mining (CRISP-DM)

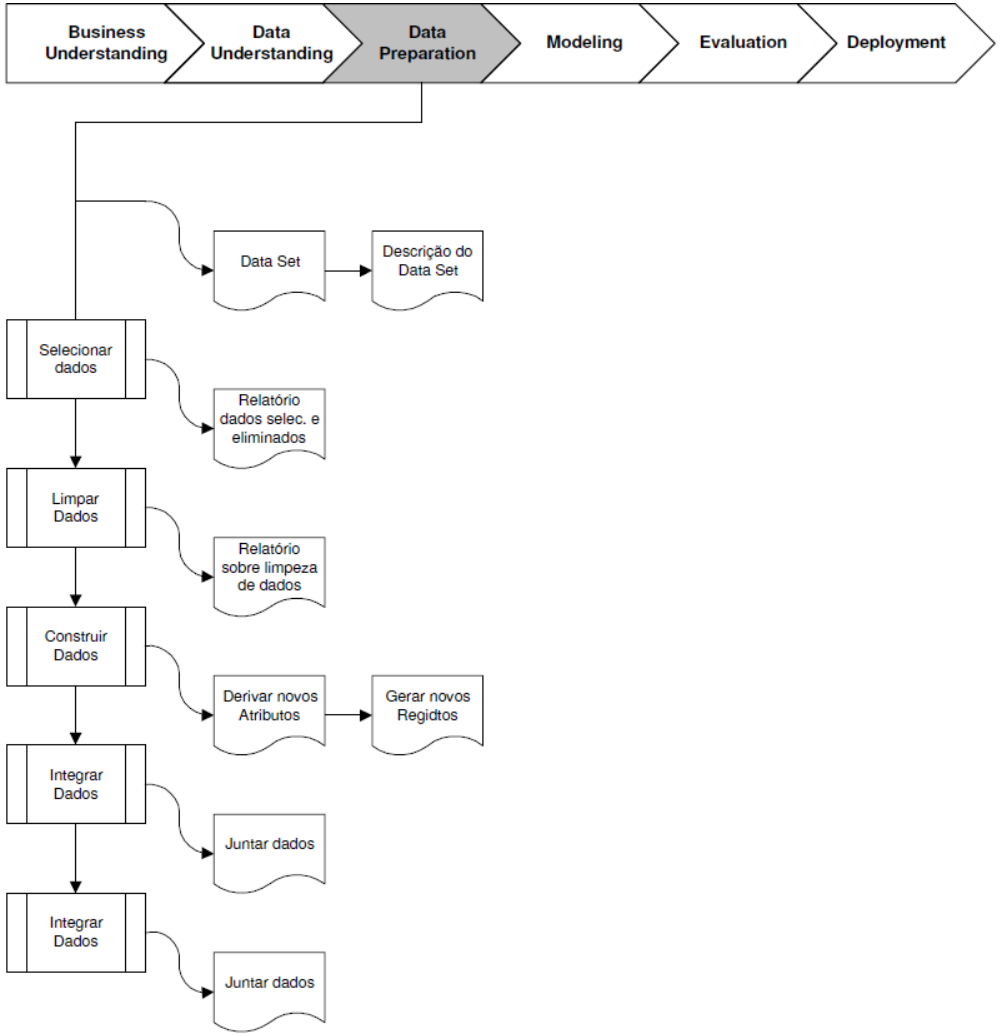




CRISP-DM – Preparação dos Dados



CRISP-DM – Atividades da Preparação dos Dados



CRISP-DM – Preparação dos Dados

Também conhecido como pré-processamento dos dados.

Fundamental para que haja sucesso em *Data Mining*!

Inclui diversas etapas, nomeadamente:

- Verificação da qualidade e integridade dos dados.
- Tratamento de valores omissos/desconhecidos (nulos).
- Deteção e tratamento de *outliers*.
- Transformações de atributos (numéricos e categóricos).
- Seleção de atributos *(a.k.a.: feature selection).*

Valores omissos/desconhecidos – soluções

Ignorar/remover os registos com valores omissos (linhas ou colunas).

Transformar todos os valores omissos numa "classe" (apenas os categóricos): **"desconhecido"**.

Substituir (*imputation*) cada valor omisso por*:*

- Valor dado por um especialista do negócio (*case substitution*);
- Valor médio, mediana ou mais comum (moda) do respetivo atributo;
- Valor retirado de outra base de dados (*cold deck*);
- Valor do exemplo mais semelhante/próximo (hot deck);
- Valor estimado por um modelo (ex.: regressão linear);
- Combinação dos métodos anteriores (*multiple combination*).

Ferramentas para imputação de valores omissos:

- Python: https://scikit-learn.org/stable/modules/impute.html;
- R: https://www.analyticsvidhya.com/blog/2016/03/tutorial-powerful-packages-imputing-missing-values/;
- Rapidminer: https://docs.rapidminer.com/latest/studio/operators/cleansing/missing/impute_missing_values.html.

Outliers

***Outliers*:** Valores atípicos provenientes de erros na coleta dos dados ou de eventos raros.

Se não estiverem relacionados com o fenómeno em estudo, **aumentam a complexidade** dos dados e **dificultam a aprendizagem** (na fase de modelação).

Solução: detetar e eliminar estes valores.

Deteção:

- **Uso de especialistas no negócio;**
- **Visualização de dados e estatísticas:** diagrama de caixa e 1.5 x desvio padrão, etc.;
- **Uso de métodos de deteção de *outliers*:** *clustering*, *Local Outlier Factor* (LOF), *Isolation Forest*, etc.

Eliminação: **ignorar/remover** registos ou **substituir** (*imputation*).

Ferramentas: Python (https://scikit-learn.org/stable/modules/outlier_detection.html), R (https://statsandr.com/blog/outliers-detection-in-r/), Rapidminer (https://docs.rapidminer.com/latest/studio/operators/cleansing/outliers/detect_outlier_distances.html), etc.

Transformações de atributos: numérico → numérico

Por vezes é útil **normalizar** ou **reescalar** atributos numéricos devido a discrepâncias de valores (dentro de um determinado atributo ou entre atributos).

O objetivo é converter os valores para uma nova escala (*normalization*/*scaling*), de modo a garantir a mesma importância para diferentes atributos.

Exemplos de transformações matemáticas (atributos numéricos):

- ***Min-max;***
- ***Z-score;***
- ***Max absolute;***
- ***Robust scaling.***

Ferramentas:

- Python: https://towardsdatascience.com/data-normalization-with-pandas-and-scikit-learn-7c1cc6ed6475;
- R: https://medium.com/swlh/data-normalisation-with-r-6ef1d1947970;
- Rapidminer: https://docs.rapidminer.com/latest/studio/operators/cleansing/normalization/normalize.html.

Transformações de atributos: numérico → categórico

Transformação dos valores numéricos em categorias, por exemplo via definição de intervalos.

Exemplo: temperatura, com definições de classes como “frio”, “morno” e “quente”.

Duas formas de o fazer:

- Definir número de classes N com **intervalos espaçados de igual cumprimento numérico;**
  Exemplo: 0 <= x <= 1, N=3; Classes: A -> [0, 0.33[; B -> [0.33, 0.66[; C -> [0.66, 1].
- Definir número de classes N, de forma a que **cada classe tenha o mesmo número de exemplos.**

Ferramentas:

- Python: https://towardsdatascience.com/data-preprocessing-with-python-pandas-part-5-binning-c5bd5fd1b950;
- R: https://r-coder.com/cut-r/;
- Rapidminer: https://docs.rapidminer.com/latest/studio/operators/cleansing/binning/discretize_by_bins.html.

Transformações de atributos: categórico → categórico

Usado quando existe um número muito elevado de classes.

Permite simplificar os modelos e facilitar a aprendizagem.

Pode ser feito pela agregação de classes numa única.

- Exemplo: Marcas de automóveis -> respetiva gama.
  - "BMW", "Mercedes", … -> "gama alta";
  - …
  - "Smart", "Citroen", … -> "gama baixa".

Pode usar-se a frequência em que a classe aparece nos dados, agregando classes menos frequentes em "Outros".

- PCP (*Percentage Categorical Pruned*).
- Implementações:
  - Python (https://pypi.org/project/cane/);
  - R (https://cran.r-project.org/web/packages/rminer/index.html).

Transformações de atributos: categórico → numérico

Alguns algoritmos de *Machine Learning* **não são capazes de lidar com valores categóricos**.

Nestes casos, é necessário converter o texto de acordo com um sistema de mapeamento.

Exemplo: cores (“vermelho”, “azul”, “amarelo”). Possibilidades:

- ***Ordinal Encoder*:** “vermelho” -> 1; “azul” -> 2; “amarelo” -> 3.
- ***One-hot Encoding* (*1-of-C*):** “vermelho” -> (1,0,0); “azul” -> (0,1,0); “amarelo” -> (0,0,1).
- ***Inverse Document Frequency* (IDF):** baseado na frequência da classe no dataset, onde valores mais próximos do zero são mais frequentes.
- **…**

Ferramentas:

- Python (https://pbpython.com/categorical-encoding.html, https://pypi.org/project/cane/);
- R (https://stats.idre.ucla.edu/r/library/r-library-contrast-coding-systems-for-categorical-variables/, https://cran.r-project.org/web/packages/rminer/index.html);
- Weka (https://machinelearningmastery.com/transform-machine-learning-data-weka/).

Transformações de atributos: categórico → numérico

Apesar da transformação categórica mais comum ser o *one-hot encoding*, casos com elevado número de classes podem ser problemáticos.

Exemplo:

- +100 países -> centenas de colunas!
- +10.000 cidades -> milhares de colunas!!!

Possíveis soluções:

- IDF: valor numérico com base na frequência no *dataset*;
- PCP: agrupar valores pouco frequentes no *dataset* em classe “outros”;
- Outras alternativas: latitude e longitude.

Seleção de Atributos

Nem todos os atributos influenciam da mesma forma o conceito a aprender (variável a prever).

Atributos irrelevantes podem aumentar o **ruído** nos dados, resultando em:

- Dificuldades na aprendizagem;
- Modelos mais complexos -> mais difíceis de explicar;
- Maior exigência computacional (ex.: mais memória, tempo de execução, …).

Como resolver?

- **Uso de especialistas do negócio:** conhecimento a *priori;*
- **Uso de filtros:** análise de correlações (ex.: gráfico de correlação de *Spearman,* coeficiente de correlação);
- **Métodos *wrapper:*** aplicados a modelos de aprendizagem.

Ferramentas: Python (https://machinelearningmastery.com/feature-selection-machine-learning-python/), R (https://www.machinelearningplus.com/machine-learning/feature-selection/), Weka (https://machinelearningmastery.com/perform-feature-selection-machine-learning-data-weka/), Rapidminer (https://docs.rapidminer.com/latest/studio/operators/modeling/optimization/feature_selection/optimize_selection.html).

Seleção de Exemplos

Não só os atributos têm influência na aprendizagem, também os exemplos devem ser representativos do fenómeno a estudar: **quantos mais registos, melhor!**

Em muitos casos do mundo real, a variável a prever é **desbalanceada**, i.e., existem muitos registos mas relativos a apenas um dos fenómenos

- Exemplo: ☺ previsão da aprovação a esta UC -> "aprovado": 98%; "reprovado": 2%.

Desbalanceamento nos dados **pode prejudicar a aprendizagem!** Como equilibrar/balancear os dados (apenas em **dados de treino**!!!) :

- ***Undersampling*,** reduzir os registos da classe maioritária "aprovado" (ex*.: random, Tomek Links);*
- ***Oversampling,*** aumentar os registos da classe minoritária "reprovado" (ex.: amostragem aleatória, geração de dados sintéticos -> *SMOTE, Gaussian Copula*).

Ferramentas:

- Python (https://imbalanced-learn.org/stable/references/index.html, https://sdv.dev/SDV/index.html);
- R (https://www.analyticsvidhya.com/blog/2016/03/practical-guide-deal-imbalanced-classification-problems/).

Exemplo: https://repositorium.sdum.uminho.pt/bitstream/1822/73976/1/paper88.pdf

Aprendizagem Automática em Sistemas Empresariais

PEDRO PEREIRA

AULA 4