Por que pré-processar os dados?

Para evitar complicações nas bases de dados, que podem surgir na forma de:

- **Dados incompletos**: atributos com ausências de valores, atributos de interesse ausentes ou contendo apenas dados agregados.
- **Ruídos**: dados errados ou *outliers* (desvio com relação ao esperado).
- **Inconsistência**: discrepância com relação aos nomes ou códigos utilizados na base de dados.

A falta de qualidade nos dados implica diretamente em um resultado pobre após a mineração. Por esse motivo, é necessário um pré-processamento para tratar os dados e garantir um resultado satisfatório. Grande parte do tempo do processo de KDD é gasto justamente pré-processando os dados.

Principais tarefas do pré-processamento

OBS: Itens marcados em **negrito** correspondem a tarefas que são de suma importância para a etapa destacada e não estão segregadas apenas a uma tarefa do pré-processamento, mas também a outras.

Limpeza de dados

Essa etapa tem como principal função garantir a consistência dos dados.

- Preenchimento de valores ausentes.
- **Suavização de ruídos**.
- Identificação e remoção de valores aberrantes (*outliers*).
- Tratamento de inconsistências.

Ausência de Valores de Atributos

- Valores de atributos (dados) nem sempre estão disponíveis. É extremamente comum que um registro não tenha valores para todas as instâncias de determinado atributo.

O que leva a essa ausência?

- Mau funcionamento de equipamento.
- Inconsistência com outros dados armazenados, causando o apagamento desse dado.
- Dado não inserido devido a falta de entendimento.
- Dado considerado irrelevante no momento da coleta.

Como lidar com os valores ausentes de atributos?

- Ignorando o registro (instância): tipicamente feito quando o atributo que está faltando é o atributo classe (de classificação) ou quando a instância contém muitos valores de atributos desconhecidos.
  OBS: Essa estratégia nem sempre é viável.
- Preencher os valores ausentes manualmente.
  OBS: Essa estratégia nem sempre é viável e, além disso, pode ter um custo de energia muito grande.
- Uso de uma constante global para preencher os valores ausentes: como o uso de "*null*" (desconhecido).
  OBS: Quando estamos treinando um modelo de classificação, o uso dessa estratégia pode fazer com que a máquina entenda, erroneamente, o valor repetido "*null*" em vários dados como um padrão.
- Usar uma medida de tendência central (exemplo: média, mediana) para preencher os valores ausentes.
- Usar uma medida de tendência central para instância pertencentes à mesma classe (caso haja o atributo classe na base de dados) da instância que possui o valor ausente.
- Utilizar o valor mais provável para preencher o valor ausente: usando regressão, inferência Bayesiana, etc.
  OBS: Solução mais adequada na maioria das situações!

Ruídos e *Outliers*

O que é um ruído?

Erro aleatório ou valor aberrante (*outlier*) dentro da base de dados.

Valores errôneos podem ocorrer devido a:

- Defeito no instrumento de coleta de dados.
- Problema na transmissão de dados.
- Limitações tecnológicas.
- Inconsistências nas convenções de nomes.

Como identificar os *outliers* (que podem representar um ruído)?

- Técnicas de estatística descritiva (exemplo: *boxplots*)
- Métodos de visualização de dados.
- Métodos de agrupamento.

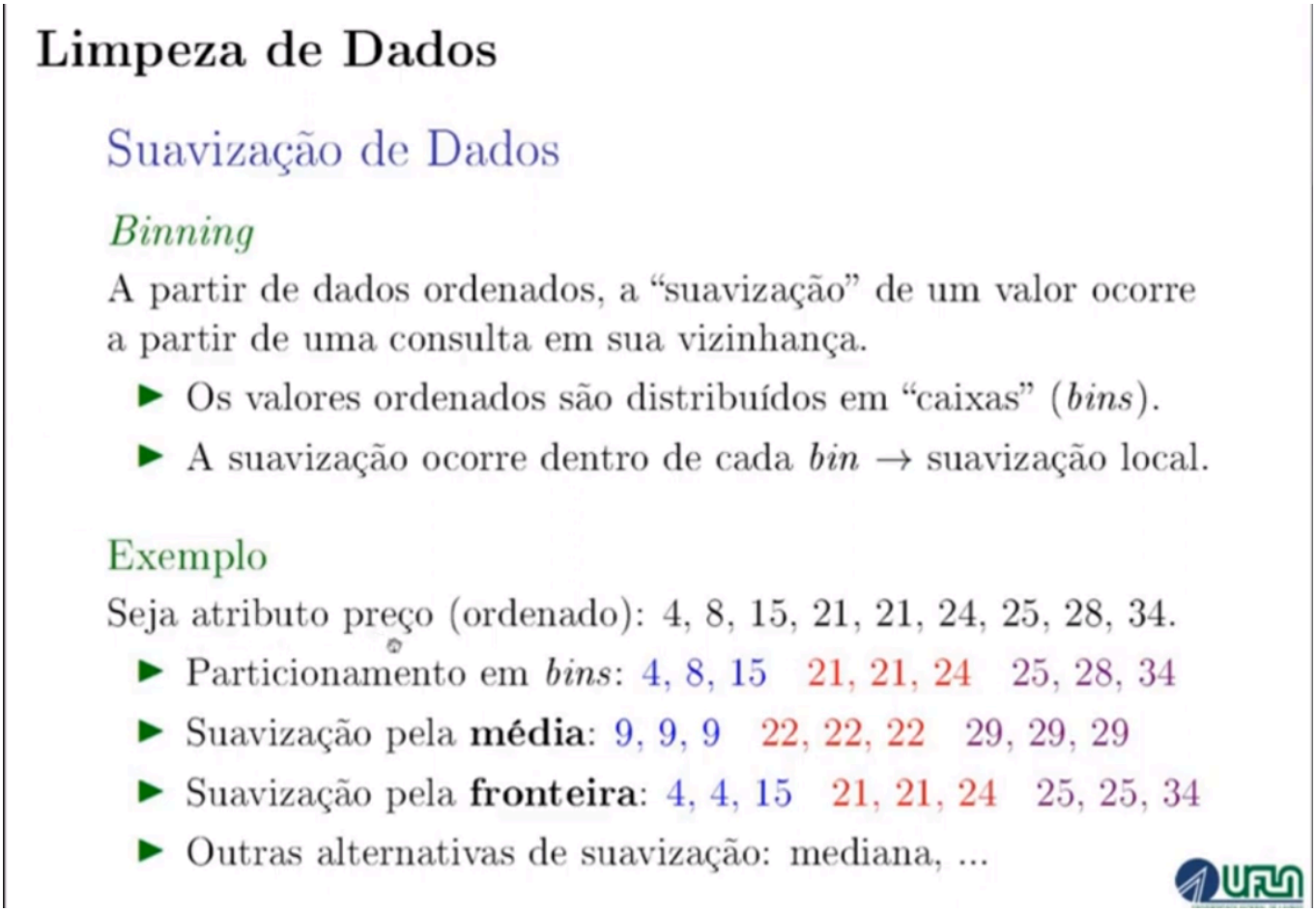
Como suavizar os dados para remover os ruídos?

-> *Binning*.
-> Regressão.

***Binning*:**

A partir de dados ordenados, suaviza o valor a partir de uma consulta em sua vizinhança.

- Valores ordenados são distribuídos em partições/"caixas" (*bins*).
- Suavização ocorre dentro de cada *bin*, sendo uma **suavização local**.

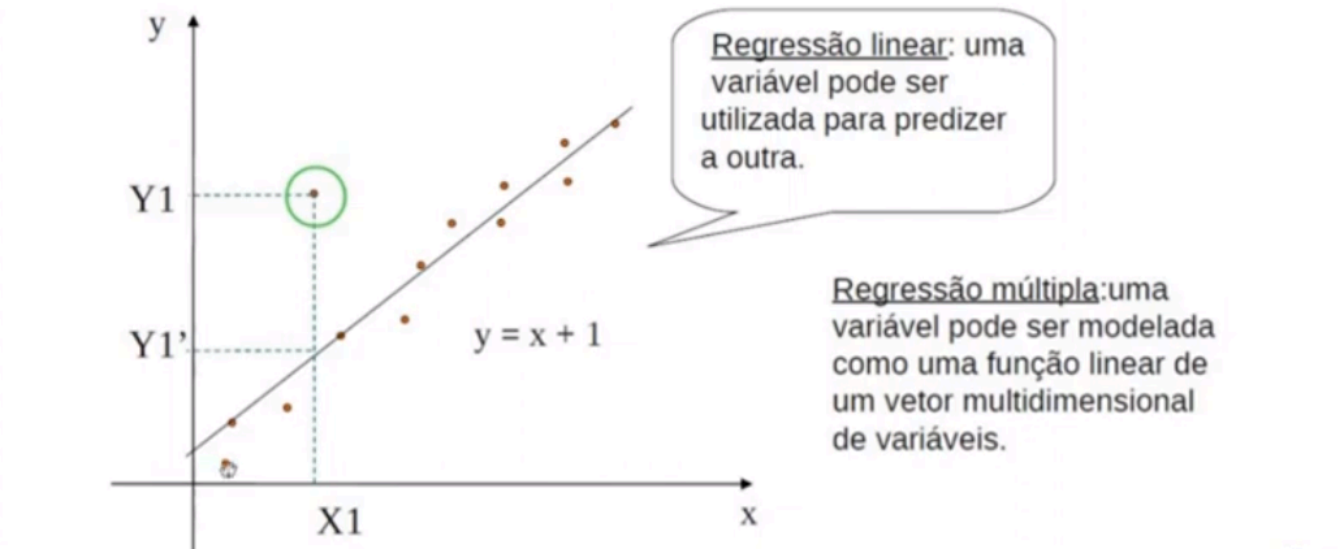


Regressão:

Dados são suavizados ajustando-os a uma função (exemplo: regressão linear).
No exemplo abaixo, Y1 é substituído por Y1' dado que valores Ys que tem um X próximo ao X1 normalmente estariam próximos daquela posição.

Regressão

Os dados são suavizados ajustando-os a uma função (p. ex.: regressão linear).



Dados Inconsistentes

Como surgem dados inconsistentes?

- Erro humano na coleta de dados.
- Erros gerados pelos equipamentos de coleta de dados.
- Erros provenientes da integração de diferentes bases de dados (mesmo atributo contendo diferentes codificações ou duplicação de instâncias).

Como encontrar dados inconsistentes?

R: A partir do conhecimento proveniente da análise prévia dos dados, buscando, por exemplo, valores que seriam inaceitáveis para determinado atributo, formatação incorreta de dados, avaliar as regras de cada tipo de atributo, etc.

Como corrigir as inconsistências?

- Escrever seu próprio código para fazer a correção ou fazer uso de ferramentas de terceiros disponíveis no mercado.
  Exemplos de ferramentas: *Potter's Wheel*, *OpenRefine*, *Trifacta Wrangler* e *Winpure*.

Integração de dados

Essa etapa tem como principal função tratar da integração de dados de diferentes fontes, que podem ser de diversas bases de dados.

Identificação de Entidade

Cuidado ao identificar entidades, para saber se ambas as entidades correspondem, ou não, à mesma informação. Para que não haja esses problemas, faz-se o uso de metadados.
Exemplo: `customer_id` pode significar o mesmo que `cust_number` em outra base de dados.

Redundância em termos de atributos

Dados redundantes podem ocorrer frequentemente quando integramos dados de fontes distintas, já que o mesmo atributo pode ter nomes diferentes a depender da base de dados e um atributo pode ter sido derivado de um outro atributo em outra tabela.

Como detectar redundâncias?

R: Análise de correlação (Teste x^2, coeficiente de correlação de Pearson, etc).

Redundância em registros: duplicação de instâncias.

Transformação de dados

Objetivo: Colocar os dados de forma apropriada para a mineração.

Vantagens: facilitar o processo de mineração e otimizar os resultados desse processo.

Abaixo estão alguns itens que compõem essa etapa:

- **Suavização de ruídos**.
  OBS: Suavização de ruídos pode estar atrelada tanto à tarefa de limpeza de dados, quanto à transformação dos dados, por isso está em negrito, como outros termos abaixo.
- **Agregação**.
  É onde ocorre as operações de sumarização dos dados, sendo algo extremamente utilizado na construção de cubos de dados.
- **Geração de hierarquia de conceitos**.
  Dados primitivos são substituídos por conceitos de ordem superior, utilizando-se de uma hierarquia de conceitos. Exemplo: atributo "Rua" tem como conceitos de ordem superior: Cidade ou País. Isso ocasiona em uma perda de detalhamento, mas facilita o reconhecimento de padrões na hora da mineração.
- Construção de atributos.
  Novos atributos são construídos e adicionados ao conjunto já existente.

- Normalização.
  Se trata do ajuste de escala.
- **Discretização dos dados**.
  Reduz a quantidade de valores de um atributo contínuo pela divisão da amplitude do atributo em intervalos (algo semelhante a classes de frequência). Esses rótulos de intervalos substituem os valores originais do atributo.

Normalização de Dados

Objetivo: Colocar os valores numa faixa pré-especificada, como, por exemplo, entre 0 e 1.

Para que serve? É importante para algoritmos que envolvem redes neurais ou cálculos de distâncias (k-NN/clusterização).
O algoritmo *backpropramation* é extremamente beneficiado da normalização de dados, já que esse processo acelera sua fase de treinamento.
Além disso, esse processo evita que algum atributo tenha mais importância (peso) que outro.

Métodos de normalização

Normalização min-máx:

$$v' = \frac{v - minA}{maxA - minA}(new\_maxA - new\_minA) + new\_minA$$

onde:

$minA$: valor mínimo do atributo A.
$maxA$: valor máximo do atributo A.
new_minA: novo valor mínimo do atributo A.
new_maxA: novo valor máximo do atributo A.
v: valor original do atributo A.

Normalização z-score:

Valores do atributo são normalizados com base na média e no desvio padrão do atributo.

$$v' = \frac{v - medA}{desv\_padA}$$

onde:

v: valor original do atributo A.
$medA$: média do atributo A.
$desv\_padA$: desvio-padrão do atributo A.

Normalização por Escala Decimal:

Normalização movendo-se o ponto decimal dos valores do atributo.

$$v' = \frac{v}{10^j}$$

onde j é o menor inteiro tal que $\text{Max}(|v'|) < 1$.

Exemplo: Atributo A contendo valores entre -986 e 917. A normalização é realizada dividindo-se os valores do atributo por 1000 (j = 3), de modo que $|-986/1000| < 1$.

Discretização

Objetivo: Substituir os valores de um atributo numérico por um classe intervalar. Dessa forma, uma instância com o valor 17 no atributo idade poderia ser substituída por 10-20 ou mesmo por um rótulo conceitual, como "adolescente" ou "jovem".

Para que serve? R: Simplificar os dados originais, tornando a mineração de dados mais eficiente; além disso, torna os padrões minerados mais fáceis de serem entendidos. Há de se levar em conta também que alguns algoritmos de mineração só trabalham com atributos discretizados (não utilizam atributos contínuos).

Redução de Dados

- **Agregação**, Agrupamento, Amostragem.
- **Geração de hierarquia de conceitos**.
- **Discretização dos dados**.
- Seleção de Atributos.