Medidas de Tendência Central (Média, Mediana e Moda)

Média: Valor central que representa o "ponto de equilíbrio" dos dados. Calculada somando todos os valores e dividindo pelo número de observações. Sensível a valores extremos (outliers).

Mediana: Valor que divide o conjunto de dados ao meio quando ordenado. 50% dos valores estão abaixo e 50% acima. Mais robusta a outliers que a média.

Moda: Valor mais frequente no conjunto de dados. Útil para identificar o valor típico ou mais comum. Pode haver múltiplas modas ou nenhuma.

Outliers são valores atípicos que se desviam significativamente do padrão geral dos dados. Exploraremos este conceito em detalhes mais adiante.

Medidas de Dispersão (Amplitude, Variância, Desvio Padrão e Coeficiente de Variação)

Amplitude: Diferença entre o maior e menor valor. Medida simples que indica o alcance total dos dados, mas muito sensível a outliers.

Variância: Média dos quadrados dos desvios em relação à média. Mede o quão espalhados estão os dados, mas em unidades quadradas.

Desvio Padrão: Raiz quadrada da variância. Indica o quanto os dados se desviam da média, na mesma unidade dos dados originais. Regra empírica: ~68% dos dados estão dentro de ±1 desvio padrão.

Coeficiente de Variação: Razão entre desvio padrão e média (em %). Permite comparar variabilidade entre variáveis com diferentes escalas. CV < 10% indica baixa variabilidade.

Coeficiente de Variação por Espécie

Medida relativa de dispersão
Análise por grupo: Permite comparar a variabilidade relativa entre diferentes grupos (espécies). Útil para identificar qual grupo tem dados mais homogêneos ou heterogêneos, independente da escala das medidas.

Medidas de Posição (Quartis e Percentis)

Quartis (Q1, Q2, Q3): Dividem os dados ordenados em 4 partes iguais. Q1: 25% dos dados abaixo; Q2 (mediana): 50% abaixo; Q3: 75% abaixo. Úteis para entender a distribuição dos dados.

IQR (Intervalo Interquartil): Diferença entre Q3 e Q1. Contém 50% central dos dados. É uma medida robusta de dispersão, não afetada por valores extremos.

Detecção de Outliers com IQR: O método IQR é amplamente usado para identificar valores atípicos:

- Limite inferior: Q1 - 1.5 × IQR
- Limite superior: Q3 + 1.5 × IQR
- Valores fora desses limites são considerados outliers
- Exemplo: Se Q1=25, Q3=75, então IQR=50. Outliers seriam valores < -50 ou > 150
- Este método é robusto pois usa a mediana (Q2) e quartis, não sendo afetado pelos próprios outliers

Percentis: Generalização dos quartis. P10 significa que 10% dos dados estão abaixo desse valor. P90-P10 indica o alcance de 80% central dos dados.

Correlação entre Variáveis (Matriz de Correlação de Pearson)

Correlação de Pearson: Mede a força e direção da relação linear entre duas variáveis. Varia de -1 a +1:

r ≈ 0: sem correlação linear
r > 0: correlação positiva (variam juntas)
r < 0: correlação negativa (variam em direções opostas)
|r| > 0.7: correlação forte

Teste de Normalidade (Shapiro-Wilk Test)

Teste de Shapiro-Wilk: Testa se os dados seguem uma distribuição normal (gaussiana). Importante para valid0r pressupostos de muitos testes estatísticos:

p-valor > 0.05: dados parecem normais
p-valor < 0.05: evidência contra normalidade
Útil para amostras pequenas (n < 50)

ANOVA (Análise de Variância)

ANOVA (Análise de Variância): É um teste estatístico usado para verificar se três ou mais grupos independentes têm médias significativamente diferentes:

- H0: todas as médias são iguais
- H1: pelo menos uma média é diferente
- p-valor < 0.05: existe diferença significativa entre grupos

O ANOVA justamente compara:

- Variabilidade entre os grupos (diferenças nas médias)
- Variabilidade dentro dos grupos (ruído interno)
- *Requer homogeneidade de variâncias e normalidade*

Por que olhar a variabilidade entre as médias dos grupos?
Imagine que a diferença entre as médias seja de apenas 2 pontos:

- Se dentro de cada grupo os valores variam muito (alta dispersão), essa diferença de 2 pode ser apenas ruído aleatório.
- Mas se dentro de cada grupo os valores são bem concentrados (baixa dispersão), então uma diferença de 2 já é um sinal consistente de efeito real.

Insight: só vale a pena dizer que os grupos são diferentes quando a diferença entre médias é maior que a variação natural dentro de cada grupo.

Intervalos de Confiança

IC 95% para as médias por espécie
Por que Intervalos de Confiança são importantes?

Imagine que você mediu o comprimento médio de pétalas de 50 flores setosa e encontrou 1.46 cm. Mas e se tivesse medido outras 50 flores? Provavelmente encontraria um valor ligeiramente diferente.

O Intervalo de Confiança quantifica essa ***incerteza***:

- IC 95% = [1.41, 1.51] significa que se repetíssemos o experimento muitas vezes, 95% dos intervalos calculados conteriam a verdadeira média populacional
- Não é "95% de chance da média estar neste intervalo" (interpretação comum mas incorreta)
- É "este método produz intervalos corretos em 95% das vezes"

Usos práticos:

- Comparar grupos: ICs que não se sobrepõem sugerem diferença significativa
- Avaliar precisão: IC estreito = estimativa precisa, IC largo = muita incerteza
- Planejar amostras: ICs muito largos indicam necessidade de mais dados

Erro Padrão (SEM): Estimativa do desvio padrão da distribuição amostral da média.

- SEM = desvio padrão / √n
- Usado para calcular o IC: IC = média ± (valor crítico × SEM)
- Menor SEM => IC mais estreito => estimativa mais precisa

O que são Outliers?

Definição e Importância:

Outliers (Valores Atípicos): São observações que se desviam significativamente do padrão geral dos dados. Podem indicar:

- Erros de medição ou digitação
- Eventos raros mas genuínos
- Populações diferentes misturadas
- Descobertas importantes

Por que detectar outliers?
- Podem distorcer análises estatísticas (média, desvio padrão)
- Afetam modelos de machine learning
- Podem revelar insights valiosos sobre o processo estudado
- Ajudam a identificar problemas na coleta de dados

Métodos principais de detecção:
- Método IQR: Baseado em quartis (robusto, não-paramétrico)
- Z-Score: Baseado em desvios padrão (assume normalidade)
- Visualização: Boxplots, scatter plots (identificação visual)

💡 ***Importante: Nem todo outlier deve ser removido! Primeiro entenda sua ori***gem.

Z-Score e Detecção de Outliers (Padronização e identificação de outliers)

Z-Score: Número de desvios padrão que um valor está distante da média:

- Z = (x - μ) / σ
- |Z| > 3: possível outlier (regra empírica)
- |Z| > 2: valor incomum (95% dos dados estão dentro de ±2σ)

Útil para comparar valores de **diferentes escalas**

POR QUE A QUALIDADE DOS DADOS É CRUCIAL?

1. **Decisões baseadas em dados imprecisos levam a resultados incorretos:**
    - Análises enviesadas
    - Modelos de baixo desempenho
    - Conclusões errôneas
2. **Estatísticas sobre problemas de qualidade:**
    - Estima-se que cientistas de dados gastam 60-80% do tempo em limpeza de dados
    - Erros de dados custam às empresas 15-25% da receita
    - 80% dos projetos de análise falham devido à má qualidade dos dados
3. **Impactos dos valores ausentes:**
    - Redução do poder estatístico
    - Introdução de viés na análise
    - Muitos algoritmos não conseguem processar valores ausentes diretamente
    - Interpretação incorreta de relações entre variáveis
4. **Garbage In, Garbage Out (GIGO):**
    - Independente da sofisticação do modelo, dados ruins produzem resultados ruins
    - A qualidade da saída é diretamente proporcional à qualidade da entrada

PRINCIPAIS CATEGORIAS DE VALORES AUSENTES:

MCAR (Missing Completely At Random)

- Ausência não depende de nenhuma variável observada ou não observada
- Como um sorteio aleatório: cada valor tem a mesma probabilidade de estar ausente
- Exemplo: Falha em equipamento de coleta, perda acidental de dados
- Impacto: Reduz o poder estatístico, mas não introduz viés sistemático

MAR (Missing At Random)

- Ausência relacionada a outras variáveis observadas
- A probabilidade de um valor estar ausente depende de dados que temos
- Exemplo: Homens tendem a não responder questões sobre saúde mental
- Impacto: Pode introduzir viés se não considerado nas análises

MNAR (Missing Not At Random)

- Ausência relacionada ao próprio valor ausente
- O valor em si influencia sua probabilidade de estar ausente
- Exemplo: Pessoas com renda alta tendem a não informar renda
- Impacto: Introduz viés significativo, difícil de corrigir

Ausência Estrutural : Valores ausentes por design ou lógica do problema

- Exemplo: Tempo de entrega ausente para compras não finalizadas

- Impacto: Deve ser interpretado diferentemente de outros tipos de ausência

ESTRATÉGIAS PARA LIDAR COM VALORES AUSENTES:

Remoção de Registros ou Variáveis/Colunas

- Eliminação listwise (remover linhas com qualquer valor ausente)
- Eliminação pairwise (usar todos os dados disponíveis para cada cálculo)
- Remoção de variáveis/colunas com muitos dados ausentes

Métodos de Imputação Simples

- Média, mediana, moda
- Valor constante (zero, valor mínimo, etc.)
- Forward/Backward fill (para dados temporais)

Métodos de Imputação Avançados:

- KNN (K-Nearest Neighbors)
- Métodos baseados em regressão
- Imputação múltipla
- Algoritmos específicos (MICE, MissForest)

Indicadores de Ausência

- Criação de variáveis dummy para indicar ausência
- Combinação com métodos de imputação

Modelos que Lidam com Valores Ausentes

- Árvores de decisão e métodos baseados em árvores
- Extensões específicas de algoritmos

Conceitos de Normalização e Padronização

O termo **escalonar** (**scaling**) é usado como conceito **geral** para **ajustar a escala das variáveis** de um dataset. A **normalização** é **um tipo específico** de escalonamento.

- **Escalonar** → conceito amplo → ajustar a escala.

- **Normalizar** → caso específico → reescala para um **intervalo definido** (normalmente [0,1].

Z-Score é um tipo de padronização, que utiliza da seguinte fórmula:

$$\underline{Z=(X-\mu)/\sigma}$$

Onde as variáveis representam, respectivamente:

- X = valor original
- μ = média dos dados
- σ = desvio padrão dos dados

Interpretação

- Um valor Z=0 → está exatamente na **média**.
- Um valor Z=+1 → está **1 desvio padrão acima** da média.
- Um valor Z=−1 → está **1 desvio padrão abaixo** da média.
- Mantém a **forma da distribuição original**, apenas muda a escala.

Min-Maxing e Standardization (Z-Score)

Min-Max Scaling é um método de **normalização** que reescala os valores para um **intervalo fixo**, geralmente **[0,1]**. Ele preserva a forma da distribuição original, mas é **muito sensível a outliers**, já que utiliza os valores mínimo e máximo para o cálculo. É indicado quando os algoritmos exigem **valores limitados**, como em redes neurais e processamento de imagens, onde os dados precisam estar em faixas controladas.

Já a **Standardization** usa o **Z-score** para **padronizar** os dados, transformando-os para que tenham **média 0** e **desvio padrão 1**. Essa técnica **não limita** os valores a um intervalo específico, mas facilita a comparação entre variáveis com escalas diferentes. Além disso, é **menos sensível a outliers** e é ideal para algoritmos baseados em distância, como **KNN, SVM, PCA**, além de modelos que dependem de **gradientes**.

Em resumo, o **Min-Max** é mais usado quando precisamos **normalizar valores dentro de um intervalo definido**, enquanto a **padronização com Z-score** é preferida quando queremos **comparar variáveis em diferentes escalas** e manter uma base estatisticamente consistente.

VARIÁVEIS CATEGÓRICAS

Modelos “clássicos” de ML trabalham com **números**. Categorias (texto) precisam virar números **sem distorcer relações**.

Há três dimensões sempre em jogo: ***(i) preservar informação/ordem, (ii) dimensionalidade/memória, (iii) risco de overfitting/leakage***.

Tipos de variáveis categóricas

- **Nominal** (sem ordem): ex. cor, cidade, produto.

- **Ordinal** (com ordem): ex. escolaridade, tamanhos (P/M/G).
- **Binária**: sim/não.

TÉCNICAS BÁSICAS DE CODIFICAÇÃO DE VARIÁVEIS CATEGÓRICAS

Técnicas de codificação de variáveis categóricas se referem ao jeitos “padrão” de transformar categorias (texto/labels) em **números** para que modelos de ML consigam usar essas variáveis.

Conceito (o que é “codificar categorias”)

- **Problema**: modelos trabalham com números; categorias como “cidade = Rio” não são computáveis direto.
- **Objetivo da codificação**: representar cada categoria como número **sem inventar relações erradas** (ex.: sem sugerir que “Rio < São Paulo”) e **controlando a dimensionalidade** (quantas colunas novas surgem).

Técnicas fundamentais

1) Label Encoding

- **Como é**: mapeia cada categoria para um inteiro (A→0, B→1…).
- **Prós**: 1 coluna, simples, barato.
- **Contras**: cria **ordem artificial** (ruim p/ nominais em modelos lineares/baseados em distância).
- **Use quando**: variável **ordinal** *ou* em **árvores** (RF/XGBoost toleram bem).

2) One-Hot Encoding

- **Como é**: 1 coluna binária por categoria (com `drop='first'` para evitar multicolinearidade).
- **Prós**: não cria ordem falsa; representação fiel de nominais.

- **Contras**: **explosão de dimensionalidade** (especialmente com alta cardinalidade).
- **Use quando**: **nominais** com **poucas categorias** (regra prática: < ~10).

3) Binary Encoding

- **Como é**: Label → número → **binário** (cria **~$\log_2(n)$** colunas).
- **Prós**: reduz dimensão vs. One-Hot; ótimo para **média/alta cardinalidade**.
- **Contras**: **menos interpretável**.
- **Use quando**: muitas categorias, mas você ainda quer algo **compacto** sem ir para hashing.

4) Ordinal Encoding

- **Como é**: define **ordem correta** e codifica 0,1,2…
- **Prós**: 1 coluna, preserva **hierarquia semântica**.
- **Contras**: assume “distâncias” iguais entre níveis.
- **Use quando**: variável **ordinal** (escolaridade, faixa de renda, satisfação).

5) Target Encoding (Mean Encoding)

- **Como é**: substitui categoria pela **média** da **variável-alvo** naquela categoria.
- **Prós**: capta relação direta com o alvo; **ótimo em alta cardinalidade**.
- **Riscos**:
    - **Overfitting** (categorias raras viram 0/1 “perfeitos”).
    - **Data leakage** (NUNCA usar o conjunto inteiro; só treino).
- **Mitigações**:
    - **Smoothing**: enc=n⋅μcat+m⋅μglobaln+m\text{enc} = \frac{n\cdot \mu_{\text{cat}} + m\cdot \mu_{\text{global}}}{n+m}enc=n+mn⋅μcat+m⋅μglobal (n=contagem, m=força da suavização).

    - **CV encoding** ou **Leave-One-Out**.
    - **Agrupar raras** antes.
- **Use quando**: **supervisionado**, **alta cardinalidade**, e você controla **vazamento**.

6) Hash Encoding (Feature Hashing)

- **Como é**: função hash mapeia categorias para **n_components** colunas fixas (colisões possíveis).
- **Prós**: **dimensão controlada**, **suporta categorias novas**, eficiente.
- **Contras**: colisões ⇒ perda de informação; **pouca interpretabilidade**.
- **Use quando**: **muitíssima cardinalidade** (milhares/milhões), **streaming/produção** com categorias novas.

7) Frequency Encoding

- **Como é**: troca categoria por **frequência** (contagem ou % no treino).
- **Prós**: 1 coluna, simples, captura “popularidade/raridade”.
- **Contras**: categorias diferentes podem ter mesma frequência; sensível ao treino.
- **Use quando**: a **popularidade importa** (e.g., canal de aquisição comum versus raro) e você quer **dimensão mínima**.

Alta cardinalidade: estratégias práticas

1. **Agrupar raras** (“Outros” ou manter top-N): reduz overfitting e dimensão; perde detalhe fino.
2. **Agrupar por similaridade semântica** (prefixos/sufixos, domínios de e-mail, CEP por região): reduz cardinalidade **preservando sentido**.
3. **Híbridas**:
    - **Hash** (capturar detalhes) **+ Target** com smoothing (ganho de performance)
    - **Top-N One-Hot** (interpretável) **+** “Outros” (ou Hash/Frequency) para o rabo longo.

Compatibilidade com algoritmos (regra de bolso)

- **Árvores (RF/GBM/XGBoost)**: toleram **Label**, funcionam bem com **Target/Frequency**.
- **Lineares/SVM/kNN**: **One-Hot** para nominais; **Ordinal** para ordinais; cuidado com Label falso.
- **Redes neurais**: para altíssima cardinalidade, **embeddings**

OBS: (**Embeddings** são representações **vetoriais** de dados categóricos ou textuais usadas em **Machine Learning** e **Deep Learning**.
A ideia é transformar categorias (ou palavras, produtos, usuários, etc.) em **vetores de números** em um **espaço contínuo** que capture **relações semânticas** entre elas.)

Boas práticas de pipeline

- **Sempre** fazer **train/test split antes** de ajustar encoders.
- **Handle unknowns**: handle_unknown='ignore' no One-Hot; **Hash** lida naturalmente; no **Frequency/Target**, mapear com valores do treino e **fallback** coerente (0 ou média global).
- Documente **mapeamentos**, **tratamento de raras**, e **parâmetros de smoothing**.
- Use **validação cruzada** para comparar encoders (olhe média e **desvio-padrão**).
- Evite **data leakage** com Target/Frequency: calcular **só no treino**.

Guia rápido de escolha (cheat-sheet)

- **Nominal com poucas categorias (< ~10)** → **One-Hot**.
- **Ordinal** → **Ordinal Encoding** (defina a ordem certa).
- **Nominal com 10–50** → **Binary** ou **One-Hot** parcialmente (top-N).
- **Alta cardinalidade (50–1000+)** → **Target (com smoothing + CV)** ou **Hash**; opcional **Frequency**.
- **Produção com categorias novas** → **Hash** (ou One-Hot com handle_unknown + “Outros”).

Erros comuns (evitar!)

- **Label** em **nominais** para modelos lineares → ordem falsa.
- **One-Hot** sem drop_first (ou sem regularização) → multicolinearidade.
- **Target** sem **smoothing/CV** → overfitting violento.
- **Hash** com **poucos componentes** → muitas colisões.
- **Frequency/Target** calculados no dataset inteiro → **leakage**.

Aplicações no mundo real

- **E-commerce/Marketing**:
    - **Canal de aquisição / meio de pagamento (nominais)**: One-Hot se poucos; Frequency/Target se muitos; Hash para produção com canais novos.
    - **Cidade (média/alta cardinalidade)**: Binary/Frequency; Hash se muitas cidades e cenário dinâmico.
    - **Escolaridade/faixa de renda (ordinais)**: **Ordinal** (ordem correta melhora relação com gastos).
- **IDs de produto**:
    - **Prefixo de categoria** no ID (ELET-12345) ⇒ **extrair prefixo** (semântica) + One-Hot/Ordinal.
    - “Rabo longo” de produtos raros ⇒ **Top-N + “Outros”**, ou **Hash + Target** com smoothing.

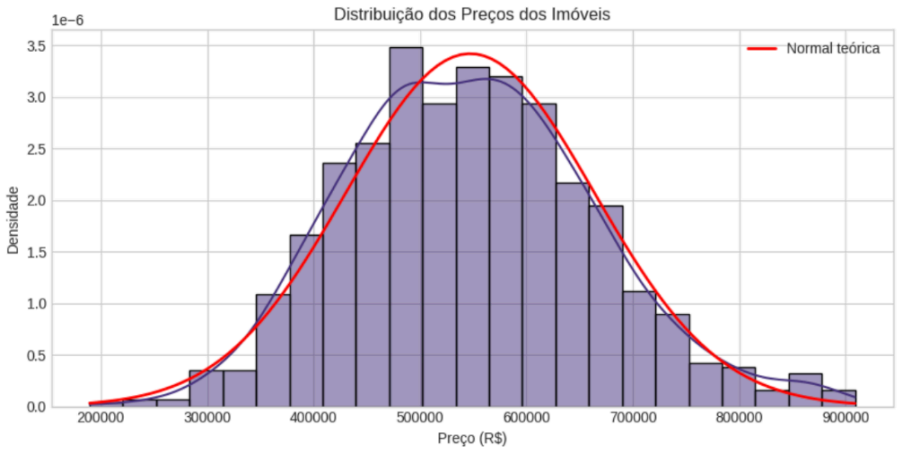
Engenharia de dados - Conceitos

Conceito e Importância

- **Engenharia de Features** é o processo de transformar dados brutos em representações mais adequadas para modelos de machine learning.
- Impactos principais:
    - Melhora o desempenho dos modelos.
    - Reduz a necessidade de algoritmos complexos.
    - Incorpora conhecimento de domínio.
    - Trata ruídos, dados ausentes e relações não lineares.
- Frase-chave: *"Feature engineering is the art part of data science"* (Andrew Ng).

Análise Exploratória (EDA)

- **Objetivo**: identificar padrões, relações e oportunidades de criação de features.
- Ferramentas: correlação, matrizes de calor, boxplots, scatterplots, análise de variáveis binárias e categóricas.
- Exemplos observados em dados de imóveis:
    - Relação não-linear entre **idade do imóvel** e preço.
    - Forte influência do **tipo de imóvel** e **localização**.
    - Interações importantes, como: andar × tipo de imóvel, casa × piscina.



Transformações Matemáticas Básicas (Parte 1)

- **Tipos principais**:
    - **Logarítmica** → lineariza relações exponenciais, reduz assimetria.
    - **Raiz quadrada** → suaviza valores altos.
    - **Quadrática** → amplifica valores altos.
    - **Inversa (1/x)** → transforma distâncias em “proximidades”.
- **Métrica-chave**: R^2 (coeficiente de determinação) → mede quanto da variabilidade do target é explicada pela variável.
- Resultados: até pequenas melhorias no R^2 (ex: 0.225 → 0.243) são relevantes em contextos multivariados.

Features de Interação (Parte 2)

- Criadas para capturar relações conjuntas. Exemplos:
    - **Índices compostos**: qualidade média, espaço, comodidades, localização.
    - **Proporções**: banheiros/quartos, área/quarto.
    - **Interações de domínio**:
        - tipo_verticalizado × andar.
        - casa × piscina.
        - impacto da renovação em imóveis antigos.
- Ganham forte correlação com o preço e revelam padrões não visíveis isoladamente.

Features Polinomiais

- Geradas automaticamente com o **PolynomialFeatures.**
- Capturam combinações de variáveis até certo grau.
- **Vantagens**: descobrem interações não óbvias.
- **Desvantagens**: explosão dimensional, multicolinearidade e risco de overfitting.

Variáveis Categóricas

- Estratégias avançadas:
    - **Estatísticas agregadas por grupo** (médias, medianas, desvio-padrão).
    - **Target encoding com regularização** → incorpora relação com o target evitando overfitting.
    - **Agrupamentos de categorias** → reduz cardinalidade e melhora generalização.
- Exemplo: agrupar tipos de imóveis em *Alto Padrão, Médio-Alto, Médio, Compacto*.

Features Temporais

- Na aula foi utilizado o mesmo dataset de preço de imóveis
- Extração de componentes: ano, mês, dia, dia da semana, trimestre
- Medidas derivadas: idade em dias, tempo desde renovação.
- **Transformações cíclicas** (seno/cosseno): preservam a natureza circular de meses, dias da semana, etc.
    - Exemplo: Janeiro e Dezembro ficam próximos no círculo, evitando inconsistências numéricas.
- Resultados: modelos com **transformações cíclicas** apresentam **ganhos expressivos de performance** (R^2 passou de 0.59 para 0.95 no exemplo).

Features Geoespaciais

- Latitude e longitude permitem extrair:
    - Distância Haversine para pontos de interesse (centros da cidade).
    - Distância mínima ao centro mais próximo.
    - Distância inversa ponderada (dando pesos diferentes a centros).
    - **Densidade de vizinhança** (nº de imóveis próximos em raio definido).
- Relação direta: proximidade a centros/amenidades e alta densidade afetam fortemente o preço.

Conclusões Gerais

- A engenharia de features é tão ou mais importante que o algoritmo de ML escolhido.
- **Transformações básicas** ajudam a linearizar relações e reduzir distorções.
- **Interações e polinômios** ampliam a capacidade de capturar padrões complexos.
- **Variáveis categóricas, temporais e/ou geoespaciais** trazem grande valor quando bem trabalhadas.
- O processo combina: conhecimento de domínio + análise exploratória + técnicas matemáticas/estatísticas.