Senac

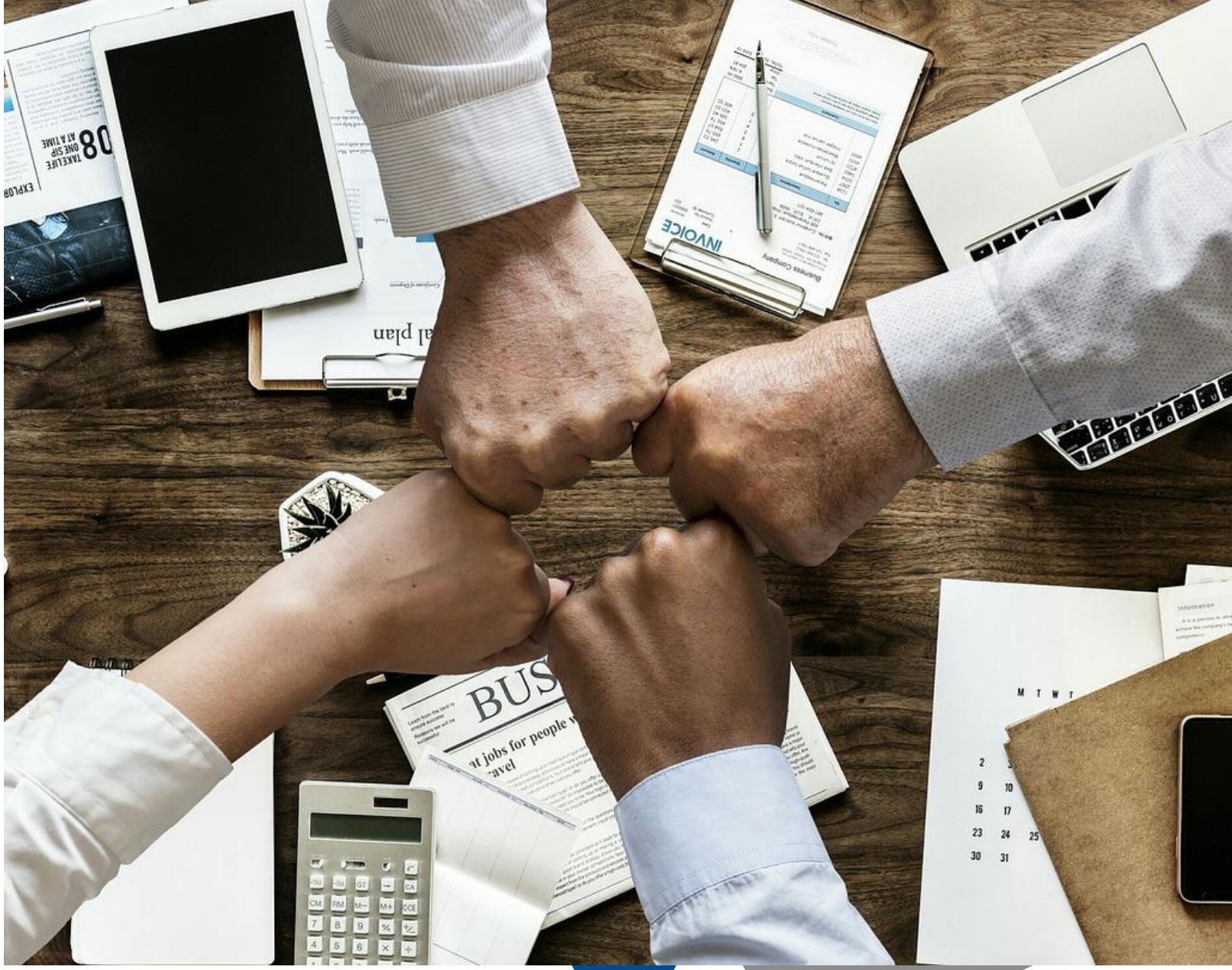
Fecomércio Sesc

# Big Data

Prof. Marco Mialaret

Março

2024



# Big Data

Onde me encontrar:

https://www.linkedin.com/in/marco-mialaret-junior/

e

https://github.com/MatmJr

# Limpeza de Dados (Relembrando)

# Big Data

Na aula passada foi dito que, em geral, eles envolvem as seguintes etapas:

1. **Entendimento do negócio:** aqui são definidas as perguntas, o objetivo da análise de dados e o plano a ser seguido;

1. **Compreensão dos dados:** etapa utilizada para coletar e explorar os dados, aumentando a compreensão sobre sua estrutura, atributos e contexto;

# Big Data

3. **Preparação dos dados:** após a análise exploratória, inicia-se o processo de limpeza, filtragem, estruturação, redução e integração dos dados;

4. **Modelagem dos dados:** envolve as tarefas de seleção dos dados, definição e construção do modelo;

# Big Data

5. **Validação do modelo:** os resultados gerados pelo modelo são avaliados, para verificar se a precisão obtida está satisfatória e coesa;

6. **Utilização do modelo:** após serem validados, os resultados dos modelos são utilizados e monitorados.

# Big Data

**Também vimos que: Os dados utilizados estão normalmente “sujos”**

Muitas vezes, as bases contêm dados incompletos, inconsistentes, corrompidos, duplicados ou em formatos inadequados, entre outros problemas. Portanto, é essencial a intervenção de um profissional capacitado para tratar esses dados antes de iniciar a análise propriamente dita.

# Big Data

Na era do big data, a crescente geração de dados amplifica a necessidade de limpeza de dados eficiente, visto que a má qualidade dos dados implica custos significativos para as empresas, chegando a US$ 12,9 milhões anualmente, conforme indicações da Gartner.

# Big Data

Além do impacto financeiro, dados imprecisos demandam dos cientistas de dados até 60% do seu tempo em tarefas de limpeza e organização, ressaltando a importância de adotar ferramentas e estratégias proativas para assegurar a qualidade dos dados desde sua coleta.

# Big Data

**Mas como resolvemos esse problema?**

Com o processo de limpeza de dados!

# Big Data

## O que é Limpeza de dados?

A limpeza de dados, essencial no gerenciamento de qualidade, envolve identificar e corrigir erros ou inconsistências nos dados para garantir sua precisão, consistência e utilidade.

# Big Data

Esse processo é crucial pois o uso de dados brutos, sem tratamento, pode levar a decisões baseadas em informações imprecisas, prejudicando a eficácia operacional e estratégica das organizações.

# Big Data

Mas por que não podemos usar dados brutos em vez de gastar tanto tempo na limpeza de dados?

# Big Data

- **Entradas com erros ortográficos:** Erros de digitação e ortografia podem levar a erros de categorização.
- **Formatos inconsistentes:** Datas, números ou categorias podem ser representados de forma diferente no mesmo conjunto de dados.
- **Valores discrepantes e erros:** Entradas incomuns ou erradas podem levar a análises imprecisas.

# Big Data

- **Registros duplicados:** Dados redundantes podem levar a estatísticas e conclusões imprecisas.
- **Valores nulos ou ausentes:** Dados incompletos podem levar a lacunas na análise e a insights imprecisos e/ou limitados.
- **Dados imprecisos:** Informações incorretas ou desatualizadas podem levar a decisões imprecisas.

# Big Data

- **Unidades não padronizadas:** Diferentes unidades de medida podem criar problemas de inconsistência de dados, especialmente ao comparar ou agregar dados.
- **Dados incompatíveis:** Dados conflitantes de fontes diferentes podem causar discrepâncias integração de dados e análise.

# Big Data

## Principais técnicas de limpeza de dados

### Removendo duplicatas:

Como fazer: Utilize algoritmos para identificar e remover linhas duplicadas com base em atributos vitais selecionados.

# Big Data

## Tratamento de dados ausentes:

Como fazer: As opções incluem imputação, exclusão ou uso de algoritmos que podem lidar com valores ausentes. A imputação pode usar estratégias baseadas em média, mediana ou modelo, como k-NN.

# Big Data

## Corrigindo dados incorretos:

Como fazer: verificações de consistência e revisão manual, se necessário. Ferramentas de preparação de dados podem ajudar na correspondência de padrões e correções.

# Big Data

## Tratamento de valores discrepantes:

Como fazer: identifique valores discrepantes por meio de métodos estatísticos, como pontuação Z ou IQR, e decida se deseja limitá-los, transformá-los ou removê-los.

# Big Data

## Normalizando Dados:

Como fazer: aplique técnicas como dimensionamento mínimo-máximo, normalização de pontuação Z ou transformações de log.

# Big Data

## Validando a consistência dos dados:

Como fazer: Crie regras de validação para verificar relacionamentos e consistência entre atributos.

# Big Data

## Transformando Dados:

Como fazer: usar transformações de dados como codificação de dados categóricos ou criação de termos de interação com base em necessidades analíticas.

# Big Data

## Mais detalhes:

https://www.astera.com/pt/type/blog/data-cleansing/

# Estudo de caso

# Big Data

Uma empresa de e-commerce especializada em produtos eletrônicos busca extrair insights de seu banco de dados de clientes. Os produtos, armazenados na sede, são entregues globalmente via navio, avião ou caminhão, com descontos baseados no peso do produto. Os clientes têm acesso ao suporte para dúvidas ou problemas e podem avaliar a experiência de compra. A única informação pessoal disponível é o gênero do cliente.

# Big Data

O objetivo do estudo de caso é explorar e organizar os dados, identificar problemas e analisá-los sob a perspectiva da entrega do produto.

# Dicionário de dados:

| Nome da Coluna | Tipo de Dado | Descrição |
|---|---|---|
| **ID** | Int | ID exclusivo de cada transação. |
| **corredor_armazem** | Character(1) | Letra do corredor do armazém onde o produto está armazenado. |
| **modo_envio** | String | Modo de envio do produto. |
| **numero_chamadas_cliente** | Int | Número de vezes que o cliente acionou o suporte da empresa. |
| **avaliacao_cliente** | Int | Avaliação do cliente sobre a experiência de compra. |
| **custo_produto** | Int | Custo do produto. |
| **compras_anteriores** | Int | Número de vezes que o cliente fez uma compra na empresa. |
| **prioridade_produto** | String | Prioridade de entrega do produto. |
| **genero** | Character(1) | Gênero do cliente (F ou M). |
| **desconto** | Int | Desconto concedido na compra do produto. |
| **peso_gramas** | Int | Peso do produto. |
| **entregue_no_prazo** | Character(1) | Se o produto foi entregue no prazo, sendo 0 (não foi entregue no prazo) ou 1 (foi entregue no prazo). |

# Dicionário de dados:

```python
import pandas as pd
import requests
from io import StringIO

# Criação do dataFrame dos alunos
# ID do arquivo no Google Drive
file_id = '1inLJ0QY0vOhD_2CKujdLEUvF4MnPbAi5'

# URL modificada para forçar o download do arquivo
url = f"https://drive.google.com/uc?id={file_id}"

# Tentando obter o arquivo com requests
try:
    response = requests.get(url)
    response.raise_for_status()  # Lança um erro para respostas não-sucedidas
    # Usando StringIO para converter o texto em um arquivo em memória e, então, lendo com o Pandas
    csv_raw = StringIO(response.text)
    ecom = pd.read_csv(csv_raw)
except requests.RequestException as e:
    print(f"Erro ao acessar o arquivo: {e}")
```

# Dicionário de dados:

ecom

✓ 0.0s

| | ID | corredor_armazem | modo_envio | numero_chamadas_cliente | avaliacao_cliente | custo_produto | compras_anteriores |
|---|---|---|---|---|---|---|---|
| 0 | 1 | D | Aviao | 4 | 2 | 177 | 3 |
| 1 | 2 | F | Aviao | 4 | 5 | 216 | 2 |
| 2 | 3 | A | Aviao | 2 | 2 | 183 | 4 |
| 3 | 4 | B | Aviao | 3 | 3 | 176 | 4 |
| 4 | 5 | C | Aviao | 2 | 2 | 184 | 3 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 10995 | 10996 | B | Navio | 4 | 1 | 232 | 5 |
| 10996 | 10997 | C | Navio | 5 | 4 | 242 | 5 |
| 10997 | 10998 | F | Navio | 5 | 2 | 223 | 6 |
| 10998 | 10999 | D | Navio | 2 | 5 | 155 | 5 |
| 10999 | 11000 | C | Navio | 4 | 2 | 255 | 5 |

11000 rows × 12 columns

# Dicionário de dados:

Verificar as informações do conjunto de dados

```
ecom.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11000 entries, 0 to 10999
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   ID                    11000 non-null  int64
 1   corredor_armazem      11000 non-null  object
 2   modo_envio            11000 non-null  object
 3   numero_chamadas_cliente  11000 non-null  int64
 4   avaliacao_cliente     11000 non-null  int64
 5   custo_produto         11000 non-null  int64
 6   compras_anteriores    11000 non-null  int64
 7   prioridade_produto    11000 non-null  object
 8   genero                11000 non-null  object
 9   desconto              11000 non-null  int64
 10  peso_gramas           11000 non-null  int64
 11  entregue_no_prazo     11000 non-null  int64
dtypes: int64(8), object(4)
memory usage: 1.0+ MB
```

# Dicionário de dados:

Separação de Variáveis numéricas das variáveis categóricas:

```
nums = ['numero_chamadas_cliente', 'custo_produto', 'compras_anteriores', 'desconto', 'peso_gramas']
✓ 0.0s

categ = ['corredor_armazem', 'modo_envio', 'avaliacao_cliente', 'prioridade_produto', 'genero']
✓ 0.0s

target = ['entregue_no_prazo']
✓ 0.0s
```

# Dicionário de dados:

Para as variáveis numéricas podemos usar o método describe para ver um resumo dos dados.

```
ecom[nums].describe()
```

| | numero_chamadas_cliente | custo_produto | compras_anteriores | desconto | peso_gramas | entregue_no_prazo |
|---|---|---|---|---|---|---|
| count | 11000.000000 | 11000.000000 | 11000.000000 | 11000.000000 | 11000.000000 | 11000.000000 |
| mean | 4.054455 | 210.200909 | 3.567727 | 13.372545 | 3633.844455 | 0.596636 |
| std | 1.141438 | 48.062985 | 1.522852 | 16.204943 | 1635.402722 | 0.490595 |
| min | 2.000000 | 96.000000 | 2.000000 | 1.000000 | 1001.000000 | 0.000000 |
| 25% | 3.000000 | 169.000000 | 3.000000 | 4.000000 | 1839.000000 | 0.000000 |
| 50% | 4.000000 | 214.000000 | 3.000000 | 7.000000 | 4149.000000 | 1.000000 |
| 75% | 5.000000 | 251.000000 | 4.000000 | 10.000000 | 5050.000000 | 1.000000 |
| max | 7.000000 | 310.000000 | 10.000000 | 65.000000 | 7846.000000 | 1.000000 |

# Dicionário de dados:

Para as variáveis categóricas isso não faz muito sentido, então devemos procurar outras formas de descrever o conjunto de dados.

```
for coluna in categ:
    print(f"Contagem de valores para {coluna}:")
    print(ecom[coluna].value_counts())
    print("\n")
```

# Dicionário de dados:

Obtendo informações do nosso target

```
ecom[target].value_counts()
```

# Dicionário de dados:

Vamos começar checando os Null e NaN

```
ecom.isnull().sum()
```

```
ecom.isna().sum()
```

# Dicionário de dados:

Não existem valores ausentes e 'Not a Number' no conjunto de dados, mas será que nossos problemas estão resolvidos ?

Também podemos checar registros duplicados.

```
ecom.duplicated().sum()
```

# Dicionário de dados:

Outliers são pontos de dados que se distanciam significativamente dos demais em um conjunto, representando valores incomuns. Esses valores atípicos podem afetar análises estatísticas, possivelmente ocultando descobertas importantes ou distorcendo resultados. A identificação de outliers não segue regras estatísticas rígidas, dependendo mais do conhecimento específico da área e da análise do processo de coleta de dados.

# Dicionário de dados:

Embora não haja uma definição matemática precisa, métodos e testes estatísticos ajudam a detectar esses pontos discrepantes, que são cruciais por seu potencial de impactar análises estatísticas adversamente.

Vamos começar a análise olhando outra vez para as variáveis numéricas

# Dicionário de dados:

```
ecom[nums].describe()
```

| | numero_chamadas_cliente | custo_produto | compras_anteriores | desconto | peso_gramas | entregue_no_prazo |
|---|---|---|---|---|---|---|
| count | 11000.000000 | 11000.000000 | 11000.000000 | 11000.000000 | 11000.000000 | 11000.000000 |
| mean | 4.054455 | 210.200909 | 3.567727 | 13.372545 | 3633.844455 | 0.596636 |
| std | 1.141438 | 48.062985 | 1.522852 | 16.204943 | 1635.402722 | 0.490595 |
| min | 2.000000 | 96.000000 | 2.000000 | 1.000000 | 1001.000000 | 0.000000 |
| 25% | 3.000000 | 169.000000 | 3.000000 | 4.000000 | 1839.000000 | 0.000000 |
| 50% | 4.000000 | 214.000000 | 3.000000 | 7.000000 | 4149.000000 | 1.000000 |
| 75% | 5.000000 | 251.000000 | 4.000000 | 10.000000 | 5050.000000 | 1.000000 |
| max | 7.000000 | 310.000000 | 10.000000 | 65.000000 | 7846.000000 | 1.000000 |

# Dicionário de dados:

O atributo desconto está com características interessantes:

- a média(mean): 13.372545
- o desvio-padrão(std): 16.204943
- o menor desconto (min): 1.000000
- o maior desconto (max): 65.00000

# Dicionário de dados:

Além disso 75% dos produtos estão com desconto menor do que 10%, ou seja, a maioria dos produtos está com desconto de até 10%, assim podemos concluir que descontos muito maiores que 10% podem sem outliers.

```
ecom.desconto.describe()
```

```
count    11000.000000
mean        13.372545
std         16.204943
min          1.000000
25%          4.000000
50%          7.000000
75%         10.000000
max         65.000000
Name: desconto, dtype: float64
```

# Dicionário de dados:

Para aprimorar a análise de outliers, uma técnica é calcular o limite em que observações podem ser consideradas atípicas usando a fórmula:

Limite = média ± 3*desvio padrão.

# Dicionário de dados:

Esse método baseia-se na regra empírica, **presumindo que os dados seguem uma distribuição normal**, onde aproximadamente 99,7% dos dados encontram-se dentro de três desvios padrão da média. Observações fora desse limite são potencialmente outliers, sugerindo variações atípicas que podem necessitar de investigação adicional para determinar suas causas ou validade.

# Dicionário de dados:

```
limiteSuperior = ecom.desconto.mean() + 3*ecom.desconto.std()
limiteSuperior = ecom.desconto.mean() - 3*ecom.desconto.std()
```

✓ 0.0s

# Dicionário de dados:

Determinando os Outliers:

```
outliers = ecom[(ecom.desconto <= limiteInferior) | (ecom.desconto >= limiteSuperior)]
outliers.head()
```

✓ 0.0s

| | ID | corredor_armazem | modo_envio | numero_chamadas_cliente | avaliacao_cliente | custo_produto |
|---|---|---|---|---|---|---|
| 36 | 37 | D | Navio | 3 | 5 | 137 |
| 60 | 61 | D | Navio | 3 | 1 | 221 |
| 62 | 63 | A | Navio | 5 | 1 | 105 |
| 111 | 112 | B | Caminhao | 4 | 2 | 239 |
| 122 | 123 | A | Caminhao | 4 | 2 | 160 |

# Dicionário de dados:

```
print("O tamanho do dataset original: ", len(ecom))
print("Os outliers do atributo desconto: ", len(outliers))
```

✓ 0.0s

```
O tamanho do dataset original:  11000
Os outliers do atributo desconto:  181
```

# Dicionário de dados:

Podemos determinar o conjunto dos valores sem outliers com a negação da operação acima:

```
dfNormDesc = ecom[(ecom.desconto > limiteInferior) & (ecom.desconto < limiteSuperior)]
dfNormDesc.head()
```

✓ 0.0s

| | ID | corredor_armazem | modo_envio | numero_chamadas_cliente | avaliacao_cliente | custo_produto |
|---|---|---|---|---|---|---|
| 0 | 1 | D | Aviao | 4 | 2 | 177 |
| 1 | 2 | F | Aviao | 4 | 5 | 216 |
| 2 | 3 | A | Aviao | 2 | 2 | 183 |
| 3 | 4 | B | Aviao | 3 | 3 | 176 |
| 4 | 5 | C | Aviao | 2 | 2 | 184 |

# Dicionário de dados:

```
print("O tamanho do dataset sem outliers no atributo desconto: ", len(dfNormDesc))
```

✓ 0.0s

```
O tamanho do dataset sem outliers no atributo desconto:  10819
```

# Dicionário de dados:

Agora podemos olhar para o describe da variável desconto

```
dfNormDesc.desconto.describe()
✓ 0.0s
count    10819.000000
mean        12.536186
std         14.981972
min          1.000000
25%          4.000000
50%          7.000000
75%         10.000000
max         61.000000
Name: desconto, dtype: float64
```

# Dicionário de dados:

Exercício: Usando outras ferramentas para resolver o mesmo problema:

Vamos usar a biblioteca SciPy para resolver esse problema

# Dicionário de dados:

```
from scipy.stats import zscore

zscore_df = dfNormDesc[nums].apply(zscore)

semOutliers_df = dfNormDesc[(zscore_df.abs()<3).all(axis=1)]
```

✓ 0.0s

# Dicionário de dados:

```
semOutliers_df.desconto.describe()
✓ 0.0s
count    10643.000000
mean        12.528047
std         14.991968
min          1.000000
25%          4.000000
50%          7.000000
75%         10.000000
max         61.000000
Name: desconto, dtype: float64
```

# Dúvidas?

## Marco Mialaret, MSc

**Telefone:**
81 98160 7018

**E-mail:**
marcomialaret@gmail.com