

Fecomércio Sesc

Big Data

Prof. Marco Mialaret

Março

2024



Onde me encontrar:

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

e

https://github.com/MatmJr



Na aula passada...

Eita, já esqueci ...

- Conversamos alguns conceitos importantes da análise de dados.
- Começamos a resolver um problema prático.





Continuando o exercício (Abra o Colab da aula passada)



O engenheiro de dados da sua empresa forneceu acesso a dois conjuntos de dados pré-processados. Agora, cabe a você analisá-los cuidadosamente para identificar quais escolas utilizaram seus recursos de forma mais eficiente na preparação de estudantes que se destacaram nas Olimpíadas de Redação e de Matemática.



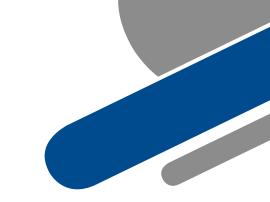
Acessando o Arquivo no drive

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

# Criação do dataFrame dos alunos
# ID do arquivo no Google Drive
file_id = '15aOJIGAyLMSY1gecjiCgu2ko_riIcKQy'

# 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)
    estudantes = pd.read_csv(csv_raw)
except requests.RequestException as e:
    print(f"Erro ao acessar o arquivo; {e}")
```



```
#Criação do dataFrame das escolas
# ID do arquivo no Google Drive
file id = '1Jgto7psHaMRTAVzcFt7D6SgJiHMB7uGT'
url = f"https://drive.google.com/uc?id={file id}"
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)
    escolas = pd.read csv(csv raw)
except requests.RequestException as e:
    print(f"Erro ao acessar o arquivo: {e}")
```









estudantes							
	ID_Estudante	Nome_Estudante	Genero	Serie	Nome_Escola	Nota_Redacao	Nota_Matematica
0	0	Kevin Bradley	М	6	Escola A	66	79
1	1	Paul Smith	М	9	Escola A	94	61
2	2	John Rodriguez	М	9	Escola A	90	60
3	3	Oliver Scott	М	9	Escola A	67	58
4	4	William Ray	F	6	Escola A	97	84



Perguntas:

Existem dados faltantes nas tabelas?

Existe algo em comum nas duas tabelas?



Verificação de dados faltantes:

```
escolas.isnull().sum()
```

```
escolas.isna().sum()
```

Repetir o processo para escolas.



Combinando os datasets:



Entendendo as variáveis categóricas:

```
data["Genero"].unique()

data["Serie"].unique()

data["Tipo_Escola"].unique()
```



Algumas perguntas:

- Qual o orçamento total das escolas?

- Qual a nota média dos alunos nas disciplinas analisadas?



Algumas perguntas:

- Qual o orçamento total das escolas?

```
escolas["Orcamento_Anual"].sum()
```

- Qual a nota média dos alunos nas disciplinas analisadas?

```
mediaRedacao = data["Nota_Redacao"].mean()
mediaMatematica = data["Nota_Matematica"].mean()
```



Personalizando os resultados obtidos:

```
# Criando um novo DataFrame com os resultados
resultados = pd.DataFrame({
    "Operação": ["Média Redação", "Média Matemática"],
    "Resultado": [mediaRedacao, mediaMatematica]
})

# Exibindo a tabela
display(resultados)
```



```
Quantos alunos ficaram com nota superior a 90 em redação?
   highRed = data[data["Nota Redacao"]>90]
   len(highRed)
E qual o percentual?
   len(highRed)/len(estudantes) * 100
```



Quantos alunos ficaram com nota superior a 90 em matemática?

```
highMat = data[data["Nota_Matematica"]>90]
len(highMat)
```

Qual o percentual?

len(highMat)/len(estudantes) * 100



Paramos aqui na aula passada!





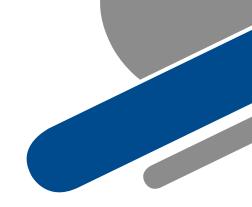
```
Quantos Alunos tiraram nota maior do que 90 nas duas disciplinas?
```

```
highBoth = data[(data["Nota_Redacao"]>90) & (data["Nota_Matematica"]>90)]
len(highBoth)
```

Qual o percentual?

```
len(data[(data["Nota_Redacao"]>90) & (data["Nota_Matematica"]>90)])/len(estudantes) * 100
```





Quantos alunos que obtiveram alto desempenho em ambas as disciplinas são de escolas públicas?

```
highBoth["Tipo_Escola"].value_counts()

v 0.0s

Tipo_Escola
```

Publica 1321

Particular 1002

Name: count, dtype: int64



Quantas alunas obtiveram alto desempenho em ambas as disciplinas?

highBoth["Genero"].value_counts()

Genero

M 1167

F 1156
Name: count, dtype: int64



```
Como ficou a distribuição dos alunos de alto desempenho pelas série?

highBoth["Serie"].value_counts()

volume

Serie
6 663
8 604
7 599
9 457
Name: count, dtype: int64
```



```
Qual o total por escolas?

highBoth["Nome_Escola"].value_counts()
```





Qual o total por escolas?

```
# Agrupando por 'Nome_Escola' e 'Tipo_Escola', e contando o número de registros em cada grupo
sumario = highBoth.groupby(['Nome_Escola', 'Tipo_Escola']).size().sort_values(ascending=False)
# Exibindo o sumário
print(sumario)
```



Qual o orçamento per capita de cada escola?

```
perCapita = escolas["Orcamento_Anual"]/escolas["Numero_Alunos"]
escolas["Per_Capita"] = perCapita
```

√ 0.0s



Quais foram as escolas mais eficientes?

Para tentar responder essa pergunta vamos adicionar o percentual de alunos de alto desempenho ao nosso dataset.

```
highBoth_num = highBoth.groupby("Nome_Escola").size()
```

numStudents = data.groupby("Nome_Escola").size()



highBoth_perc = highBoth_num/numStudents highBoth_perc



Tentando adicionar direto ao DataFrame ocorre um erro.





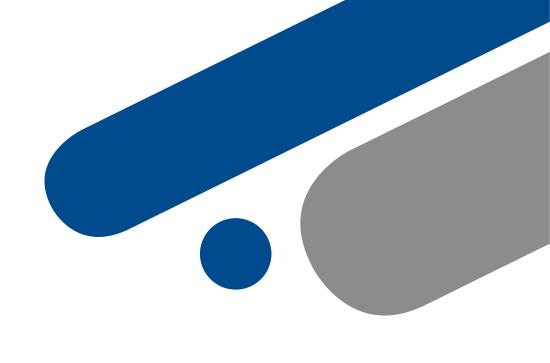
```
escolas["HighBoth_perc"] = escolas["Nome_Escola"].map(highBoth_perc)
```





```
escolas.sort_values(by=['Per_Capita', 'High_Performance_perc'], ascending=[True, False])
```









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

- Entendimento do negócio: aqui são definidas as perguntas, o objetivo da análise de dados e o plano a ser seguido;
- 2. Compreensão dos dados: etapa utilizada para coletar e explorar os dados, aumentando a compreensão sobre sua estrutura, atributos e contexto;



- 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;



- 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.



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.



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 <u>Gartner</u>.



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.



Mas como resolvemos esse problema?

Com o processo de limpeza de dados!



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.



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.



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



- 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.

- 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.



- 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.



Principais técnicas de limpeza de dados

Removendo duplicatas:

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



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.



Corrigindo dados incorretos:

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



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.



Normalizando Dados:

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



Validando a consistência dos dados:

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



Transformando Dados:

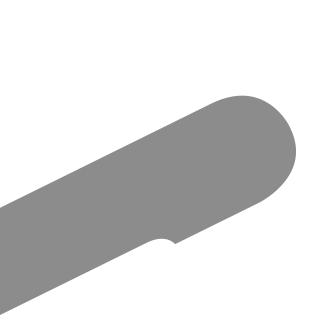
Como fazer: usar <u>transformações de dados</u> como codificação de dados categóricos ou criação de termos de interação com base em necessidades analíticas.



Mais detalhes:

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





Estudo de caso



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.



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:

```
import pandas as pd
import requests
from io import StringIO
# Criação do dataFrame dos alunos
# ID do arquivo no Google Drive
file id = 'linLJ0QY0vOhD 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}")
```

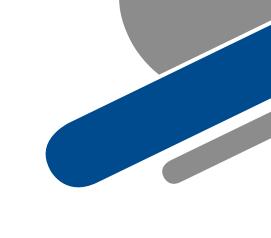


Dicionário de dados:

```
ecom.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11000 entries, 0 to 10999
Data columns (total 12 columns):
                            Non-Null Count Dtype
   Column
    TD
                            11000 non-null int64
                            11000 non-null object
    corredor armazem
    modo envio
                            11000 non-null object
    numero chamadas cliente 11000 non-null int64
    avaliacao cliente
                            11000 non-null int64
    custo produto
                            11000 non-null int64
                            11000 non-null int64
    compras anteriores
    prioridade produto
                            11000 non-null object
                            11000 non-null object
    genero
    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
```



Dúvidas?







Marco Mialaret, MSc

Telefone:

81 98160 7018

E-mail:

marcomialaret@gmail.com

