

Aula 2 - Gerenciamento de Dados

Introdução ao aprendizado de máquina - UEMA 2025

Thiago S. F .Silva

2025-12-10

Aquecimento: dado vs informação?

- Usabilidade
- Reproducibilidade
- Transparência
- Praticidade

Quem é o seu principal colaborador?

- Usabilidade
- Reproducibilidade
- Transparência
- Praticidade

Quem é o seu principal colaborador?

- Você mesmo daqui a 3 meses.

Dados estruturados vs. não-estruturados

- **Dados Estruturados:** planilhas, bancos de dados - dados organizados de maneira tabular.
- **Dados Não-Estruturados:** fotos, gravações, transcrições de entrevistas, paginas da web, etc.

Metadado: O dado sobre o dado. Informação adicional / documentação explicativa que permite o uso correto e reproduzível dos dados.

- Especialmente importante para dados não estruturados.
- Ex: Data, hora, local, coletor, instrumento, condições ambientais, unidades, nomes de variáveis, nomenclatura dos arquivos, organização das pastas. etc.

Exemplo: exemplo_metadado.txt

Exemplo: <https://zenodo.org/records/13887881>

O gerenciamento de dados começa no campo/laboratório

- Protocolo de amostragem
- Caderno de laboratório
- Planilhas de campo

Boas práticas

1. Faça um **estudo piloto**
2. Prepare suas planilhas / caderno **antes** de começar.
3. Anote observações sobre **tudo** o que desviar do protocolo.
Não, você não vai lembrar depois.
4. **Backup, backup, backup:**

Boas práticas

1. Faça um **estudo piloto**
2. Prepare suas planilhas / caderno **antes** de começar.
3. Anote observações sobre **tudo** o que desviar do protocolo.
Não, você não vai lembrar depois.
4. **Backup, backup, backup:**
 - Dados analógicos:
 - Ao final de cada dia/sessão, **fotografe** seu caderno / planilha de campo.
 - Se houver acesso à internet, já sincronize com a nuvem.

Boas práticas

1. Faça um **estudo piloto**
2. Prepare suas planilhas / caderno **antes** de começar.
3. Anote observações sobre **tudo** o que desviar do protocolo.
Não, você não vai lembrar depois.
4. **Backup, backup, backup:**
 - Dados digitais:
 - Baixe com a maior frequência possível.
 - Mantenha pelo menos duas cópias, em dois dispositivos diferentes.
 - Se houver acesso à internet, já sincronize com a nuvem.

5. Tente digitar os dados o mais rápido possível - enquanto os detalhes do trabalho ainda estão frescos na sua cabeça.

Caderno/planilha vs. coletar dados direto no PC/tablet/telefone?

Qual software você usa pra digitar seus dados?

- Excel: cuidado com a formatação dos dados

Qual software você usa pra digitar seus dados?

- Excel: cuidado com a formatação dos dados
- Datas e horas: [formato ISO8601](#)

Qual software você usa pra digitar seus dados?

- Excel: cuidado com a formatação dos dados
- Datas e horas: [formato ISO8601](#)
- Separadores decimais e de milhar: vírgula ou ponto?

Qual software você usa pra digitar seus dados?

- Excel: cuidado com a formatação dos dados
- Datas e horas: [formato ISO8601](#)
- Separadores decimais e de milhar: vírgula ou ponto?
- Indicadores de dado faltante (*missing data*)

Exemplo 1 - Preparando uma planilha para receber dados

Exemplo 2 - Identifique e corrija os problemas

Boa parte do trabalho de análise de dados consiste na organização / limpeza de dados.

Tidy Data: <https://www.jstatsoft.org/article/view/v059i10>

- Cada variável é uma coluna
- Cada observação é uma linha

Table 1: Tabela 1. Número de fatalidades em relação ao número de animais infectados, para três diferentes fazendas entre 2023-2025.

Local	2023	2024	2025
Fazenda A	12/50	13/45	9/52
Fazenda B	8/35	10/40	7/38
Fazenda C	8/42	5/37	3/51

Quantas variáveis tem essa tabela?

- Local/Fazenda

Exemplo

- Local/Fazenda
- Ano

Exemplo

- Local/Fazenda
- Ano
- Número de animais infectados

Exemplo

- Local/Fazenda
- Ano
- Número de animais infectados
- Número de fatalidades

Exemplo

- Local/Fazenda
- Ano
- Número de animais infectados
- Número de fatalidades

Exemplo

- Local/Fazenda
- Ano
- Número de animais infectados
- Número de fatalidades

Como organizar essa tabela de maneira *tidy*?

Exemplo

Local	ano	infectado	fatalidade
Fazenda A	2023	50	12
Fazenda A	2024	45	13
Fazenda A	2025	52	9
Fazenda B	2023	35	8
Fazenda B	2024	40	10
Fazenda B	2025	38	7
Fazenda C	2023	42	8
Fazenda C	2024	37	5
Fazenda C	2025	51	3

- Que dados você está coletado / planeja coletar?
- Como esses dados estão sendo organizados e armazenados?
- Como você acha que pode melhorar essa organização?
- Se o seu computador desaparecer agora, o que aconteceria com o seu trabalho?

Prática: Configurando o RStudio para trabalhar

Uma parte **essencial** da análise de dados é a estruturação de um *projeto*:

- Conjunto de pastas que organiza o trabalho
- Ajuda na auto-documentação do projeto
- Diferentes análises - diferentes projetos

Organização mínima de um projeto:

```
meu_projeto
|____ src
|____ data
|      |____ raw
|      |____ processed
|____ outputs
|____ docs
```

Organização mínima de um projeto:

```
meu_projeto
|____ scripts
|____ dados
|          |____ brutos
|          |____ processados
|____ resultados
|____ notas
```

Boas práticas de nomenclatura de pastas/arquivos

- sem letras maiúsculas
- sem acentos
- sem espaços
- usar somente - e _ como caracteres extra

Estilos de nomenclatura de pastas/arquivos/variáveis

Snake case: `meu_lindo_arquivo.txt`

Kebab case: `meu-lindo-arquivo.txt`

Camel case: `meuLindoArquivo.txt`

Pascal case: `MeuLindoArquivo.txt`

Escolha sua favorita - mas seja **consistente!**

Exemplo: criando um projeto no RStudio

Qual o melhor formato para armazenamento de dados estruturados?

- Excel (xls, xlsx): incluem informações sobre o tipo de dado, mas não é um formato universal.

Qual o melhor formato para armazenamento de dados estruturados?

- Excel (xls, xlsx): incluem informações sobre o tipo de dado, mas não é um formato universal.
- Texto (csv, tsv): extremamente simples, universalmente acessíveis, mas não guardam informação sobre o tipo de dado.

Qual o melhor formato para armazenamento de dados estruturados?

- Excel (xls, xlsx): incluem informações sobre o tipo de dado, mas não é um formato universal.
- Texto (csv, tsv): extremamente simples, universalmente acessíveis, mas não guardam informação sobre o tipo de dado.
- **Parquet**: formato aberto desenvolvido especialmente para armazenamento eficiente de dados. Recentemente se tornou padrão em data science - mas já está sendo substituído.

Arquivos RDS e RData

- Formato interno do R.
- RData pode armazenar múltiplos objetos, RDS somente um objeto.
- Formatos que só são entendidos pelo R.
- OK para salvar dados intermediários durante o processamento.

E dados não-estruturados?

- Priorize formatos abertos

E dados não-estruturados?

- Priorize formatos abertos
- Use texto simples para os metadados

E dados não-estruturados?

- Priorize formatos abertos
- Use texto simples para os metadados
- Dados de texto não estruturados: JSON

Exemplo prático: lendo diferentes tipos de arquivos para o R.

Passo 1: preparando o arquivo `plot_florestal.xlsx`

Pacote: `readxl()` (parte do tidyverse)

Funções: `read_xls()` e `read_xlsx()`

Help: https://readxl.tidyverse.org/reference/read_excel.html

```
library(readxl)
plot_florestal <- read_xlsx('../data/plot_florestal.xlsx')
head(plot_florestal)
str(plot_florestal)
glimpse(plot_florestal)
```

Resultado

A tibble: 6 x 16

	Espécie	Plot	Tag	Ramo	Data	`Ano Recrutamento`	V
	<chr>	<chr>	<dbl>	<chr>	<dtm>	<chr>	<
1	Eschweile~	2b	247	<NA>	2022-10-01 00:00:00	NA	m
2	Pouteria ~	2b	248	<NA>	2022-10-02 00:00:00	NA	m
3	Neea	2b	249	<NA>	2022-10-03 00:00:00	NA	v
4	Leonia gl~	2b	250	<NA>	2022-10-04 00:00:00	NA	c
5	Pouteria ~	2b	251	<NA>	2022-10-05 00:00:00	NA	m
6	Iryanther~	2b	252	<NA>	2022-10-06 00:00:00	NA	u

i 9 more variables: Família <chr>, Gênero <chr>, Mortalidade_2022 <dbl>
DAP_2017 <chr>, DAP_2022 <chr>, ALTURA_2017 <chr>, ALTURA_2022 <dbl>
pom_2017 <chr>, pom_2022 <chr>

Rows: 120

Columns: 16

\$ Espécie	<chr>	"Eschweilera albiflora", "Pouteria elegans",
\$ Plot	<chr>	"2b", "2b", "2b", "2b", "2b", "2b", "2b", "2
\$ Tag	<dbl>	247, 248, 249, 250, 251, 252, 253, 254, 255,
\$ Ramo	<chr>	NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,

**Desafio: como ler a segunda planilha
do arquivo excel?**

Texto Simples (CSV, TSV, etc)

R básico: `read.csv()`,
`read.csv2()`, `read.delim()`, `read.table()`

Tidyverse (pacote `readr`): `read_csv()`, `read_csv2()`,
`read_tsv()`, `read_delim()`

```
library(readr)
turb_datos <- read_csv('../data/turbidez.csv')
head(turb_datos)
glimpse(turb_datos)
```


Resultado

```
# A tibble: 6 x 6
```

	id_amostra	local	data	temp_c	ph	turbidez_ntu
	<dbl>	<chr>	<chr>	<chr>	<chr>	<chr>
1	1063	A02	2024-02-10	19.41	7	11.1
2	1083	B01	17/02/2024	24.59	7	5
3	1068	B01	15/02/2024	20.26	6.71	15.4
4	1065	A01	2024-01-21	29.09	7.19	4
5	1068	B01	23/01/2024	21.93	-999	11.1
6	1011	A02	25/01/2024	25.41	6.97	10.7

```
Rows: 120
```

```
Columns: 6
```

```
$ id_amostra    <dbl> 1063, 1083, 1068, 1065, 1068, 1011, 10
$ local        <chr> "A02", "B01", "B01", "A01", "B01", "A0
$ data         <chr> "2024-02-10", "17/02/2024", "15/02/2024"
$ temp_c       <chr> "19.41", "24.59", "20.26", "29.09", "21.93", "25.41"
```

Vantagens e desvantagens

Excel: - Inclui informação sobre o *data type* de cada coluna -
Tamanho de arquivo é maior - Não é legível sem ter o Excel
instalado - Quem garante que daqui a 10 anos esse formato ainda
vai existir?

Texto: - Não inclui informação sobre os *data types* - Arquivos
menores - Qualquer software é capaz de abrir/ler

Qualquer arquivo digital é armazenado internamente como números (bits).

Como representar texto como números?

Qualquer arquivo digital é armazenado internamente como números (bits).

Como representar texto como números?

Encoding: padrão que define como cada caractere é representado por um número

Os primeiros computadores tinham muito pouca memória/disco.

ASCII: encoding mais básico, só inclui 128 caracteres (7-bit encoding)

Dec	Hex	Oct	Chr	Dec	Hex	Oct	Chr	Dec	Hex	Oct	Chr	Dec	Hex	Oct	Chr	Dec	Hex	Oct	Chr
0	00		NULL	32	20	040	6#032: Space	64	40	100	6#064: @	96	60	140	6#096: `				
1	01		Start of Header	33	21	041	6#033: !	65	41	101	6#065: A	97	61	141	6#097: a				
2	02		Start of Text	34	22	042	6#034: "	66	42	102	6#066: B	98	62	142	6#098: b				
3	03		End of Text	35	23	043	6#035: #	67	43	103	6#067: C	99	63	143	6#099: c				
4	04		End of Transmission	36	24	044	6#036: \$	68	44	104	6#068: D	100	64	144	6#100: d				
5	05		Enquiry	37	25	045	6#037: %	69	45	105	6#069: E	101	65	145	6#101: e				
6	06		Acknowledgment	38	26	046	6#038: &	70	46	106	6#070: F	102	66	146	6#102: f				
7	07		Bell	39	27	047	6#039: '	71	47	107	6#071: G	103	67	147	6#103: g				
8	08		Backspace	40	28	050	6#040: (72	48	110	6#072: H	104	68	150	6#104: h				
9	09		Horizontal Tab	41	29	051	6#041:)	73	49	111	6#073: I	105	69	151	6#105: i				
10	A	012	Line feed	42	2A	052	6#042: *	74	4A	112	6#074: J	106	6A	152	6#106: j				
11	B	013	Vertical Tab	43	2B	053	6#043: +	75	4B	113	6#075: K	107	6B	153	6#107: k				
12	C	014	Form feed	44	2C	054	6#044: ,	76	4C	114	6#076: L	108	6C	154	6#108: l				
13	D	015	Carriage return	45	2D	055	6#045: -	77	4D	115	6#077: M	109	6D	155	6#109: m				
14	E	016	Shift Out	46	2E	056	6#046: .	78	4E	116	6#078: N	110	6E	156	6#110: n				
15	F	017	Shift In	47	2F	057	6#047: /	79	4F	117	6#079: O	111	6F	157	6#111: o				
16	10	020	Data Link Escape	48	30	060	6#048: 0	80	50	120	6#080: P	112	70	160	6#112: p				
17	11	021	Device Control 1	49	31	061	6#049: 1	81	51	121	6#081: Q	113	71	161	6#113: q				
18	12	022	Device Control 2	50	32	062	6#050: 2	82	52	122	6#082: R	114	72	162	6#114: r				
19	13	023	Device Control 3	51	33	063	6#051: 3	83	53	123	6#083: S	115	73	163	6#115: s				
20	14	024	Device Control 4	52	34	064	6#052: 4	84	54	124	6#084: T	116	74	164	6#116: t				
21	15	025	Negative Ack.	53	35	065	6#053: 5	85	55	125	6#085: U	117	75	165	6#117: u				
22	16	026	Synchronous idle	54	36	066	6#054: 6	86	56	126	6#086: V	118	76	166	6#118: v				
23	17	027	End of Trans. Block	55	37	067	6#055: 7	87	57	127	6#087: W	119	77	167	6#119: w				
24	18	030	Cancel	56	38	070	6#056: 8	88	58	130	6#088: X	120	78	170	6#120: x				
25	19	031	End of Medium	57	39	071	6#057: 9	89	59	131	6#089: Y	121	79	171	6#121: y				
26	1A	032	Substitute	58	3A	072	6#058: :	90	5A	132	6#090: Z	122	7A	172	6#122: z				
27	1B	033	Escape	59	3B	073	6#059: ;	91	5B	133	6#091: [123	7B	173	6#123: {				
28	1C	034	File Separator	60	3C	074	6#060: <	92	5C	134	6#092: \	124	7C	174	6#124: }				
29	1D	035	Group Separator	61	3D	075	6#061: =	93	5D	135	6#093:]	125	7D	175	6#125: }				
30	1E	036	Record Separator	62	3E	076	6#062: >	94	5E	136	6#094: ^	126	7E	176	6#126: ~				
31	1F	037	Unit Separator	63	3F	077	6#063: ?	95	5F	137	6#095: _	127	7F	177	6#127: Del				

asciichars.com

UTF-8: usa um método diferente (variable length encoding) e por isso é capaz de armazenar até 1.1 milhão de caracteres. - Múltiplas línguas - Símbolos - Emojis - Etc...

Existem muitos outros...

ASCII vs UTF-8 vs outros

Se você tenta ler um arquivo criado com um certo encoding, especificado um encoding diferente, o resultado pode ser estranho:

```
# Gravando um arquivo em UTF-8
con <- file("exemplo_utf8.txt", open = "w", encoding = "UTF-8")
writeLines("Maçã, café, romã, pêra, ", con)
close(con)
```


ASCII vs UTF-8 vs outros

```
# Lendo como UTF-8
con <- file("exemplo_utf8.txt", open = "r", encoding = "UTF-8")
content <- readLines(con, warn = FALSE)
close(con)
content
```

```
[1] "Maçã, café, romã, pêra, "
```

ASCII vs UTF-8 vs outros

```
# Lendo como ASCII
con <- file("exemplo_utf8.txt", open = "r", encoding = "ASCII")
content <- readLines(con, warn = FALSE)
close(con)
content
```

```
[1] "MaC'C#, cafC), romC#, pC*ra, p\037\030"
[2] ""
```