2389787

slido

Join at slido.com #2389787

ⓘ Click **Present with Slido** or install our Chrome extension to display joining instructions for participants while presenting.

⚠️ Reminder to start the Zoom recording!

2389787

💻 Lots of demo code today. Get ready to type!

LECTURE 5

Data Cleaning and EDA

Exploratory Data Analysis and its role in the data science lifecycle.

Data 100, Summer 2025 @ UC Berkeley

Josh Grossman and Michael Xiao

📣 Announcements

2389787

Lab 2A due tonight!

Homework 2A due Wednesday!

If you completed the Pre-Semester Survey on time, and requested the Graded Discussion Scheme, then **you have been assigned a discussion to attend – starting today**!

- If you didn’t complete the Pre-Semester Survey, or opted for Non-Graded Discussion, you won’t have a discussion section.
- Check the Sections Tool for more info (linked on Ed)

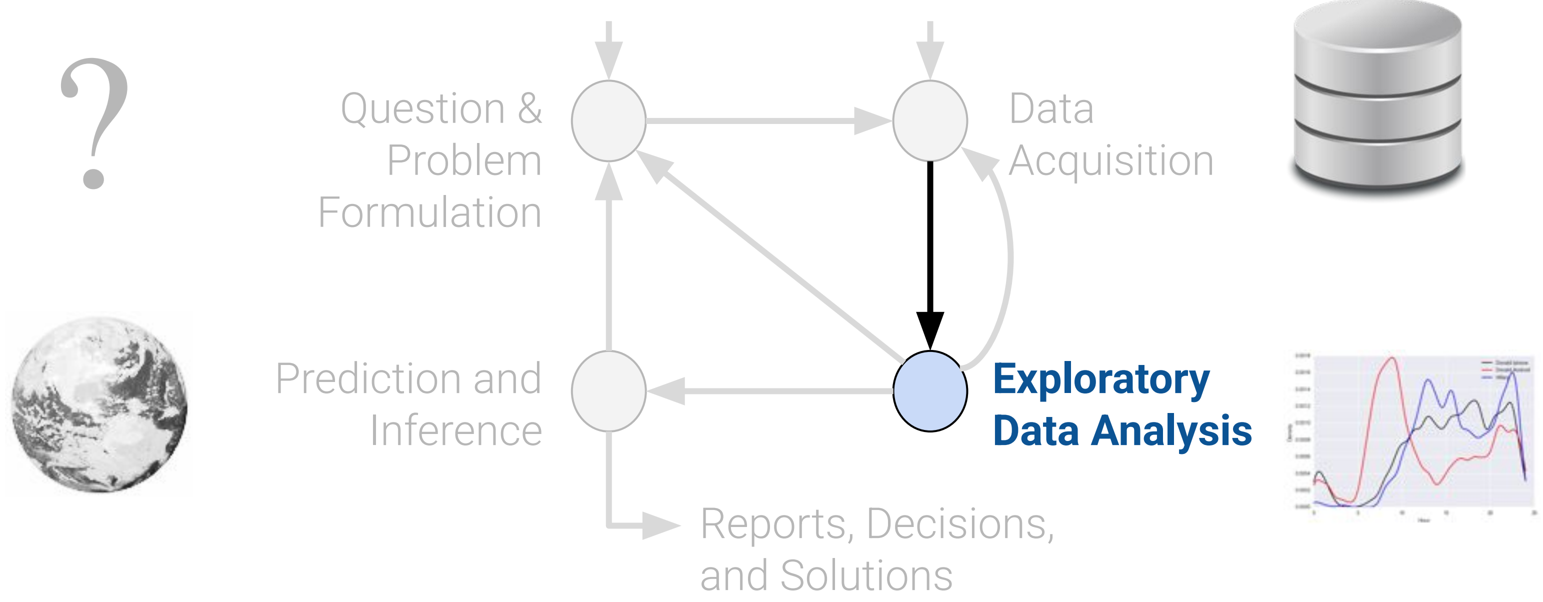
OH continues as usual! Check the course calendar for the specific hours.

Reminder to make sure your **DSP accommodations are submitted ASAP**

- **By Sunday, July 6th** at the latest
- Very important if you have exam accommodations

Plan for Next Few Weeks

2389787



Question & Problem Formulation

Data Acquisition

Exploratory Data Analysis

Prediction and Inference

Reports, Decisions, and Solutions

(Week 1)

Exploring and Cleaning Tabular Data
From `datascience` to `pandas`

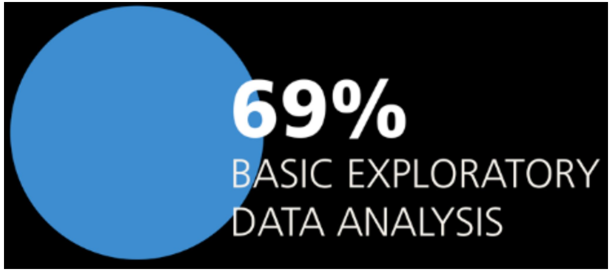
(Week 2)

Data Science in Practice
EDA, Data Cleaning, Text processing (regular expressions), Visualization

2389787

Exploratory Data Analysis (EDA)

EDA is unboxing for data!

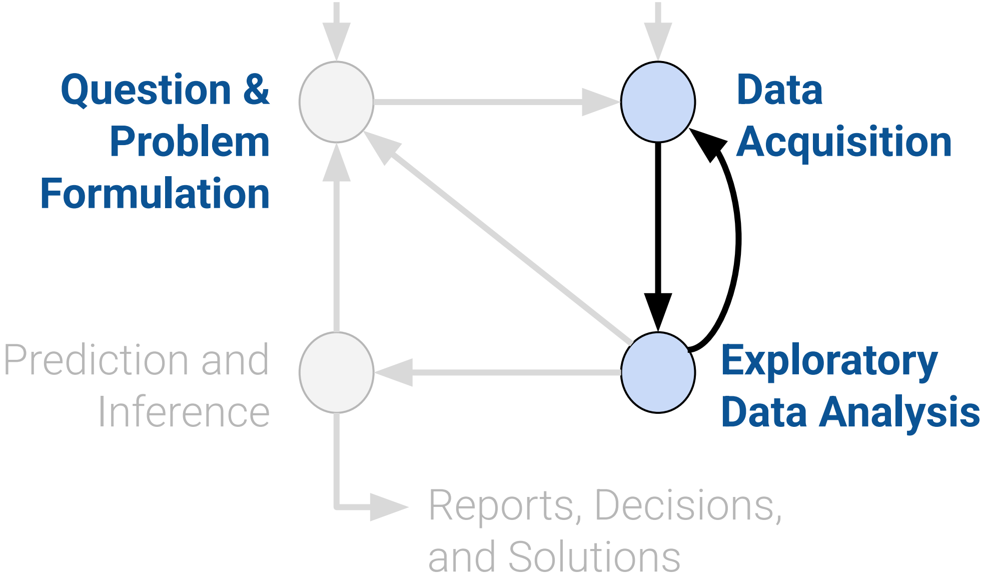


From Lecture 1

The Data Science Lifecycle is a Cycle

2389787

In practice, EDA informs whether you need more data to address your research question.



2389787

Key Data Properties to Consider in EDA

Structure -- the "shape" of a data file

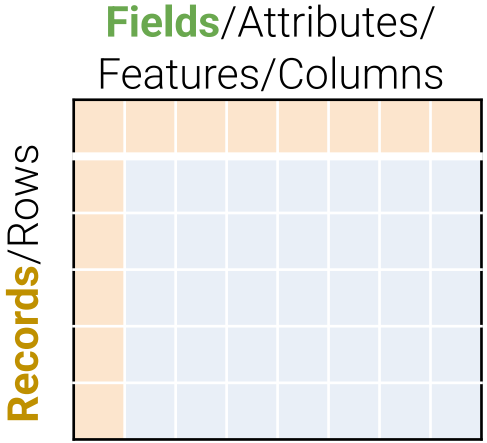
Granularity -- how fine/coarse is each datum

Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

Rectangular Data

2389787



We often prefer **rectangular data** for data analysis

- Easy to manipulate and analyze
- Big part of **data cleaning**: Reshape to be more rectangular
- Example: dataset of spam emails → table of word counts

Two kinds of rectangular data: **Tables** and **Matrices**.

Tables (`DataFrames` in R/Python)

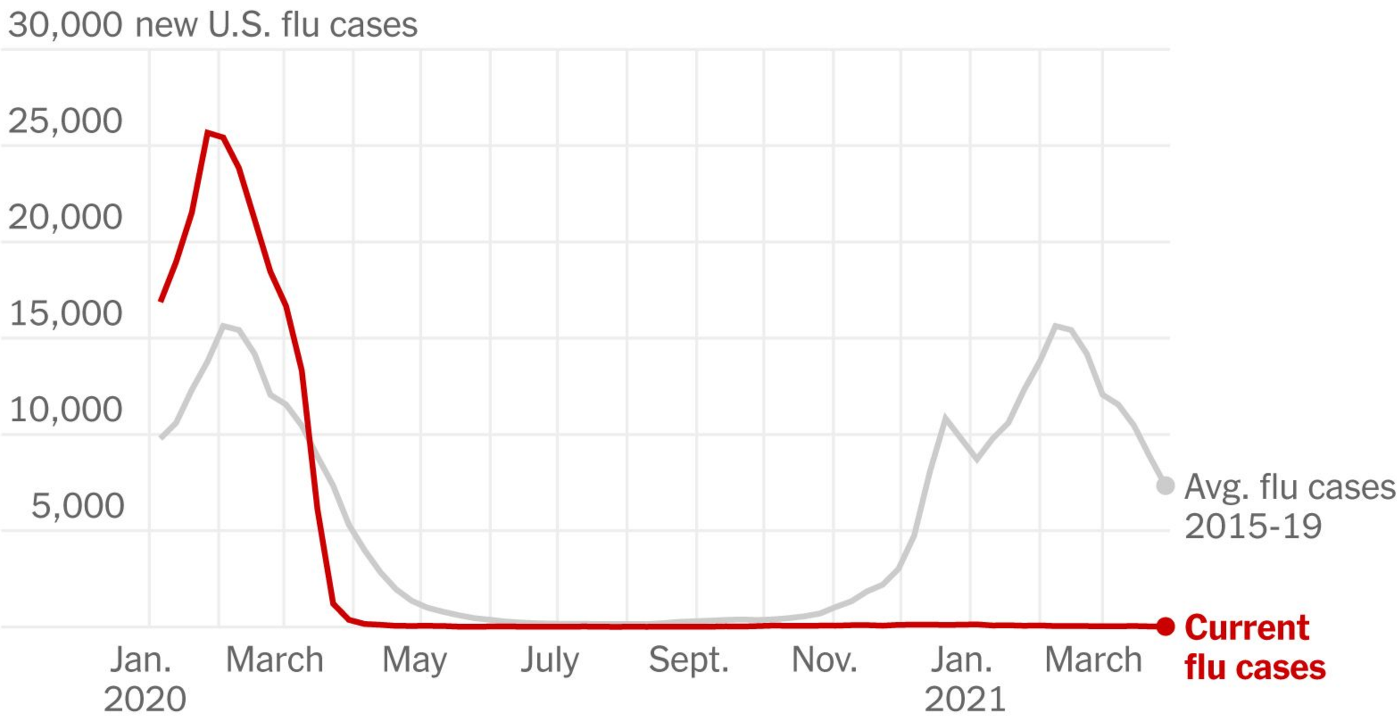
- Named columns with **different** types
- Manipulated w/ data transformation functions (group by, join, filter …)

Matrices

- **Numeric** data of the **same** type (float, int, etc.)
- Manipulated w/ linear algebra
- Faster computation, but less flexible

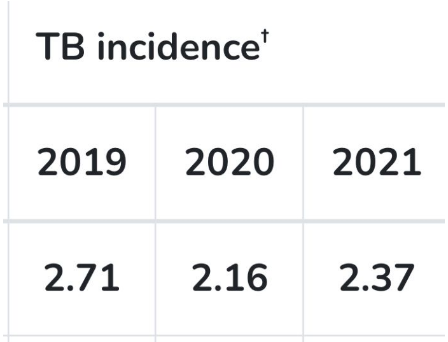
Other Illnesses During the COVID-19 Pandemic

2389787



Source: New York Times

Other Illnesses During the COVID-19 Pandemic



TB incidence[†]

2019	2020	2021
2.71	2.16	2.37

TB: Tuberculosis
Incidence: # cases per 100,000 people

Source: CDC (Centers for Disease Control and Prevention)

You're an analyst at the CDC.

How do you calculate these values?

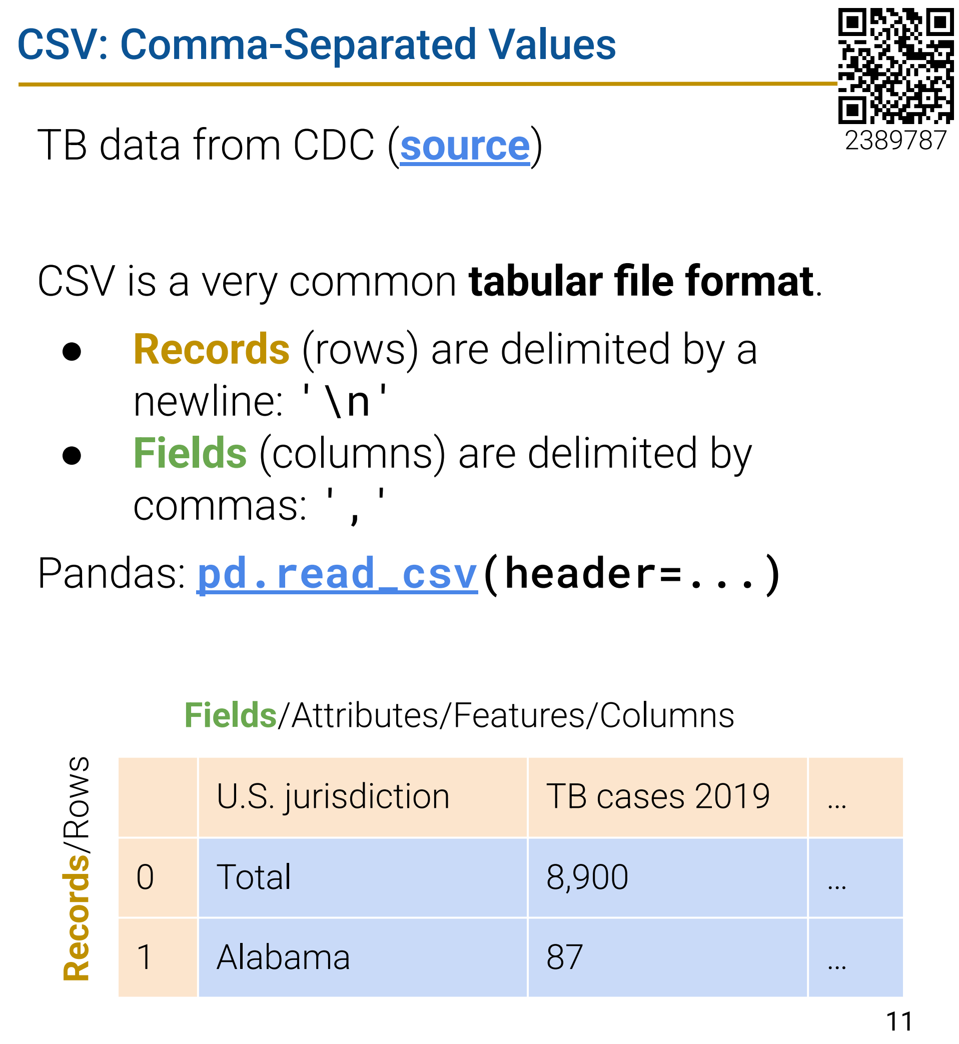
U.S. TB incidence → Need U.S. TB case counts and U.S. population

U.S. TB case counts → **State-level TB case counts**

State-level TB case counts → Hospital-level TB case counts

Demo Slides

lec05-part-1-eda-tuberculosis.ipynb



CSV: Comma-Separated Values

2389787

TB data from CDC (source)

CSV is a very common **tabular file format**.

- **Records** (rows) are delimited by a newline: `'\n'`
- **Fields** (columns) are delimited by commas: `','`

Pandas: `pd.read_csv(header=...)`

Fields/Attributes/Features/Columns

Records/Rows

	U.S. jurisdiction	TB cases 2019	...
0	Total	8,900	...
1	Alabama	87	...

11

Other Data Formats

2389787

- **Image:** medical diagnosis
- **Audio:** speech recognition, sentiment analysis
- **Video:** object tracking, facial recognition
- **Text:** LLMs, legal document review
- **…**

All formats above can be represented in tabular/matrix form.

Key Data Properties to Consider in EDA

(we'll come back to this!)

2389787

Structure -- the "shape" of a data file

Granularity -- how fine/coarse is each datum → a single "piece" of data

Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

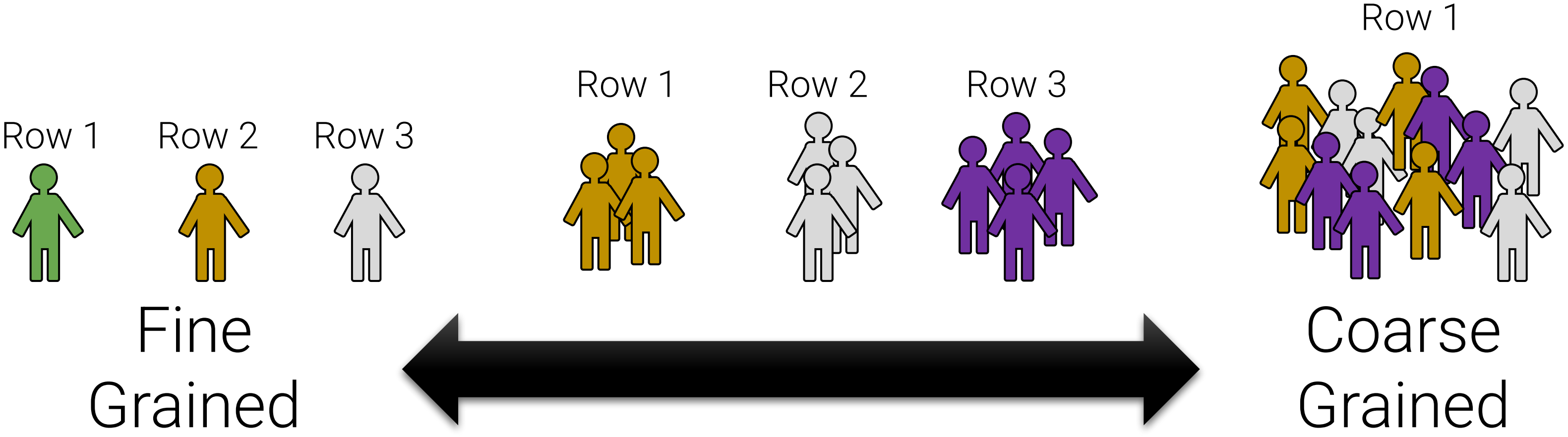
An Important Conundrum in Data Science

2389787

Singular "data"	"The data show**s** …"
Plural "data" (~~datums~~)	"The data show …"

Either is fine 🙂

Granularity: How Fine/Coarse Is Each Datum?



Row 1 Row 2 Row 3

Fine Grained

Row 1 Row 2 Row 3

Row 1

Coarse Grained

What does each **record** (row) represent?

- Examples: a single purchase, a single person, a group of users
- Some data will include summaries (aka **rollups**) as records.

If the data are **coarse**, how were the records aggregated?

- Summing, averaging, or something else?

Demo Slides

lec05-part-1-eda-tuberculosis.ipynb

Granularity of TB data

What does each row of the TB data represent?

Do all rows have the same granularity?

Image source: NPR

Multiple Files

File Format

Variable Type

Key Data Properties to Consider in EDA

Structure -- the "shape" of a data file

Granularity -- how fine/coarse is each datum

Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

2389787

Demo Slides

lec05-part-1-eda-tuberculosis.ipynb

Joining Multiple Files

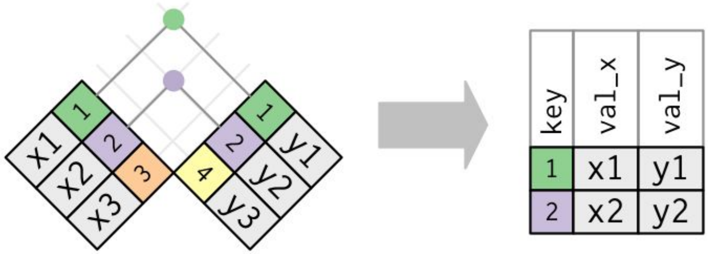
2389787

Incidence = Case Count / Population

TB case counts → CDC data

U.S. population → Census data

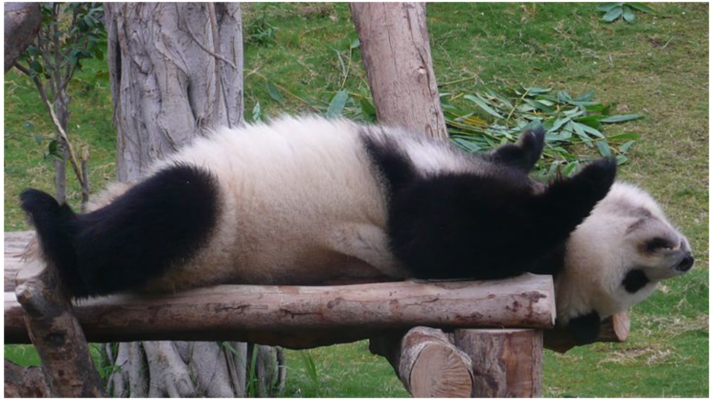
It's time to merge!



key	val_x	val_y
1	x1	y1
2	x2	y2

Image source: R4DS

2-minute stretch break!



Key Data Properties to Consider in EDA

Multiple Files

File Format

Variable Type

Structure -- the "shape" of a data file

Granularity -- how fine/coarse is each datum

Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

2389787

Demo Slides

lec05-part-2-eda-structure.ipynb

TSV: Tab Separated Values

Another common table file format.

- **Fields** are delimited by `'\t'` (tab)
- Like a CSV with tabs instead of commas

`pd.read_csv`: Need to specify

```
delimiter='\t'
```

TaB soda: Precursor to Diet Coke

Demo Slides

lec05-part-2-eda-structure.ipynb

JSON: JavaScript Object Notation

2389787

CA Senators+Reps data (congress.gov API)

Very similar to Python dictionaries

- **Self-documenting**: Metadata (data about the data) + records in the same file

`pd.read_json()`

`pd.DataFrame(json_dict)`

JSON is **non-rectangular**, so good to inspect the file before importing.

- Nested tables
- Inconsistent fields across records

2389787

Multiple Files

File Format

Variable Type

Structure -- the "shape" of a data file

Granularity -- how fine/coarse is each datum

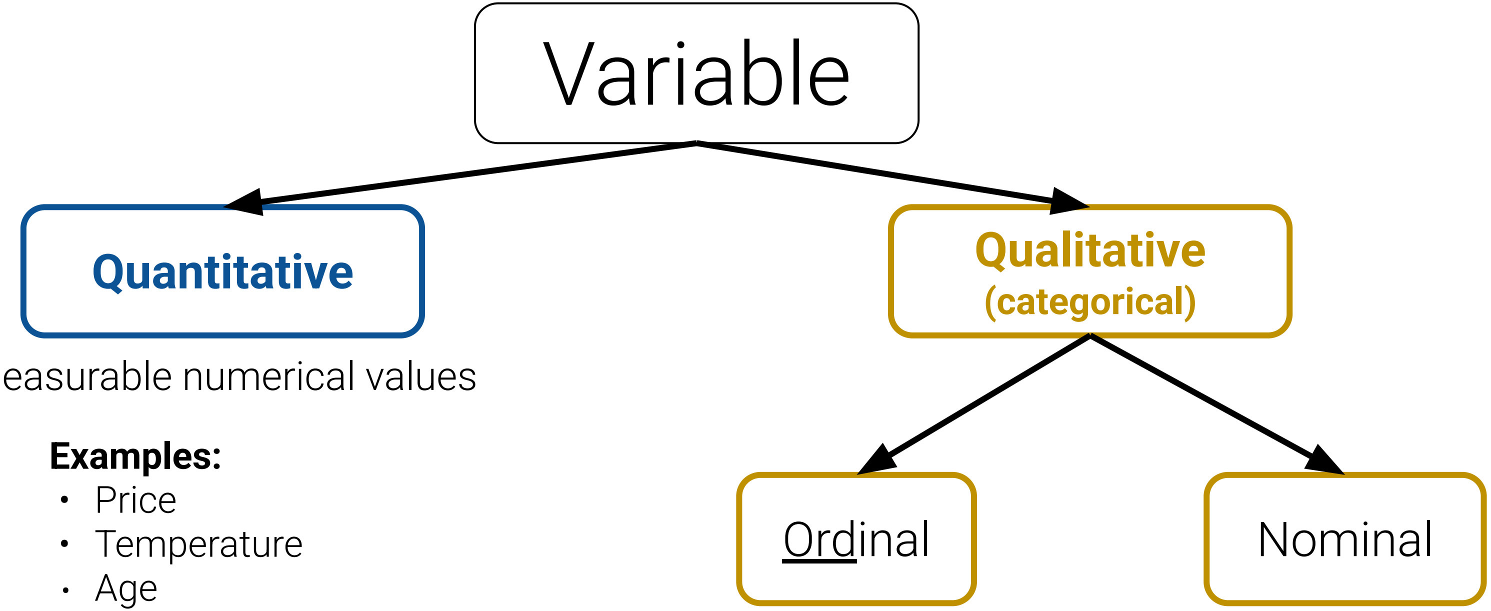
Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

Key Data Properties to Consider in EDA

Variable Feature Types

2389787



Variable

Quantitative

Qualitative (categorical)

Ordinal

Nominal

Measurable numerical values

Examples:

- Price
- Temperature
- Age

Ordered categories

Examples:

- Grade level
- Age group

Unordered categories

Examples:

- Phone brand
- Cal ID number
  (assigned, not measured!)

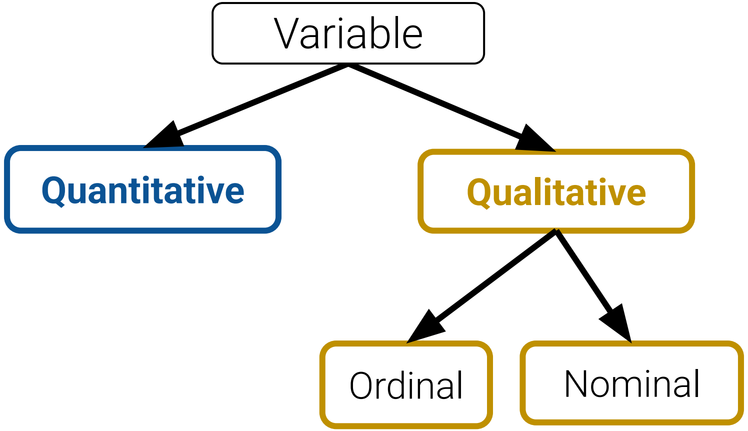
A safe default: Store qualitative data as strings!

Variable Types

2389787

What is the feature type of each variable?

Q	Variable	Feature Type
1	CO_2 level (ppm)	Quantitative
2	Income bracket (low, med, high)	Qualitative Ordinal
3	Race/Ethnicity	Qualitative Nominal
4	Political party	Qualitative Ordinal / Nominal
5	Year	Quantitative / Qualitative Ordinal
6	GPA	Quantitative / Qualitative Ordinal
7	Date and time	**Slido!**



The distinction between categories is sometimes murky. Context matters!

2389787

slido

What type of variable is a datetime (e.g., 01/01/2025 3:30pm)?

ⓘ Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.

2389787

Key Data Properties to Consider in EDA

Structure -- the "shape" of a data file

Granularity -- how fine/coarse is each datum

Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

Efficiently Storing Datetimes

As humans, we write datetimes as strings: **01/01/2025 3:30pm**

There are 13 characters in the string **010120250330p**

Datetime column with 1 billion entries → ~13 billion characters → 13 GB column 😱

What if we stored datetimes as **integers**?

1 billion integers → ~4 billion bytes → 4 GB column 😎

Temporality: Unix / POSIX Time

Datetimes measured in **seconds** since **January 1st 1970 UTC** (Coordinated Universal Time)

Jun 30, 2025 11:00am PDT → **1751306400** (1,751,306,400 seconds)

Jun 30, 1950 11:00am PDT → **-615535200** (-615,535,200 seconds)

Another bonus of numeric representation: We can do math!

For example, we can calculate # days between dates using subtraction and division.

Demo Slides

lec05-part-2-eda-structure.ipynb

Unix / POSIX Time

2389787

Berkeley PD calls for service data

`pd.to_datetime()`

`pd.series.dt.date()`

`pd.series.dt.dayofweek()`

`pd.series.dt.hour()`

...

2389787

Key Data Properties to Consider in EDA

Structure -- the "shape" of a data file

Granularity -- how fine/coarse is each datum

Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

2389787

What are some potential issues with this dataset?

ⓘ Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.

What are Some Potential Issues with this Dataset?

2389787

ID	Category	State	Location	Device	Purchased	...
0	Shoes	CA	CA	1	1	...
1	Socks	NM	NM	1	0	...
2	Socks	XY	XY	1	0	...
3	Shirts	NY	NY	1	NA	...
4	Shoes	FL	FL	1	0	...
4	Shoes	FL	FL	1	0	...
5	Shirts	CA	CA	1	0	...
6	Pnts	TX	TX	1	1	...
7	Hats	CA	CA	1	-1	...
...	...	...	...	...	...	...

Faithfulness: Do I trust this data?

2389787

Fully Duplicated Records or Fields

Identify and ignore/drop.

Labeling or Spelling Errors

Apply corrections. Only ignore if you have to.

Missing data

Need to think carefully about **why** the data is missing.

Examples

```
" "            1970, 2000
0, -1          NaN
999, 12345     Null
```

NaN: "Not a Number"

Real zero or NaN placeholder? Sometimes both!

See footnote 12 in onlinelibrary.wiley.com/doi/abs/10.1111/jels.12343

Missing Data: Approaches

2389787

A. Keep as NaN

- A good default.
- If qualitative/categorical → Create a "Missing" category.

B. Drop records with missing values

- Typically a bad default!
- Temperature probe went offline for a minute → Likely **missing at random** → OK to drop
- Police officer never records outcomes of vehicle stops → Likely not missing at random

C. Imputation/Interpolation: Infer missing values (with caution!)

- **Mean/median imputation:** replace NaN with mean/median
- **Hot deck imputation:** use a random non-NaN value
- **Regression imputation:** use a model to predict value
- **Multiple imputation:** multiple random values + check sensitivity

(beyond this course)

Demo Slides

lec05-part-2-eda-structure.ipynb

Missing Values

2389787

Berkeley PD calls for service data

Approaches:

- Keep missing values as NaN
- Drop missing values
- Impute

[pd.series.isna()]

[pd.DataFrame.info()]

We did it!

Structure -- the "shape" of a data file

Granularity -- how fine/coarse is each datum

Temporality -- how is the data situated in time

Faithfulness -- how well does the data capture "reality"

2389787

LECTURE 5

Data Cleaning and EDA

Content credit: Acknowledgments