Responsible Data Science

The data science lifecycle

February 27 & March 6, 2023

Prof. Julia Stoyanovich

Center for Data Science &
Computer Science and Engineering
New York University

NYU TANDON SCHOOL OF ENGINEERING

NYU Center for Data Science

r/ai center for responsible ai

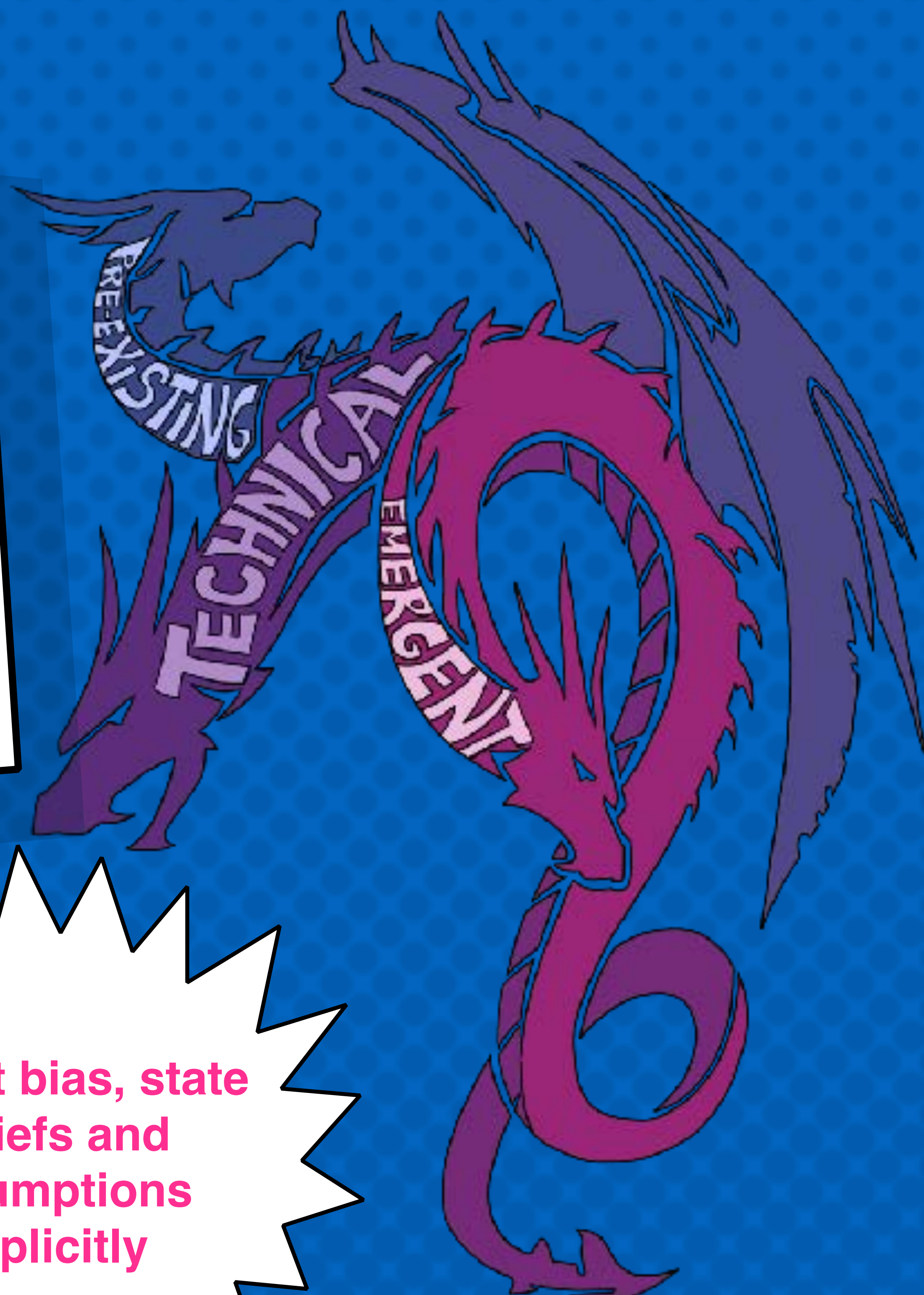
This week's reading

Recall: Bias in computer systems

Pre-existing is independent of an algorithm and has origins in society

Technical is introduced or exacerbated by the technical properties of an ADS

Emergent arises due to context of use

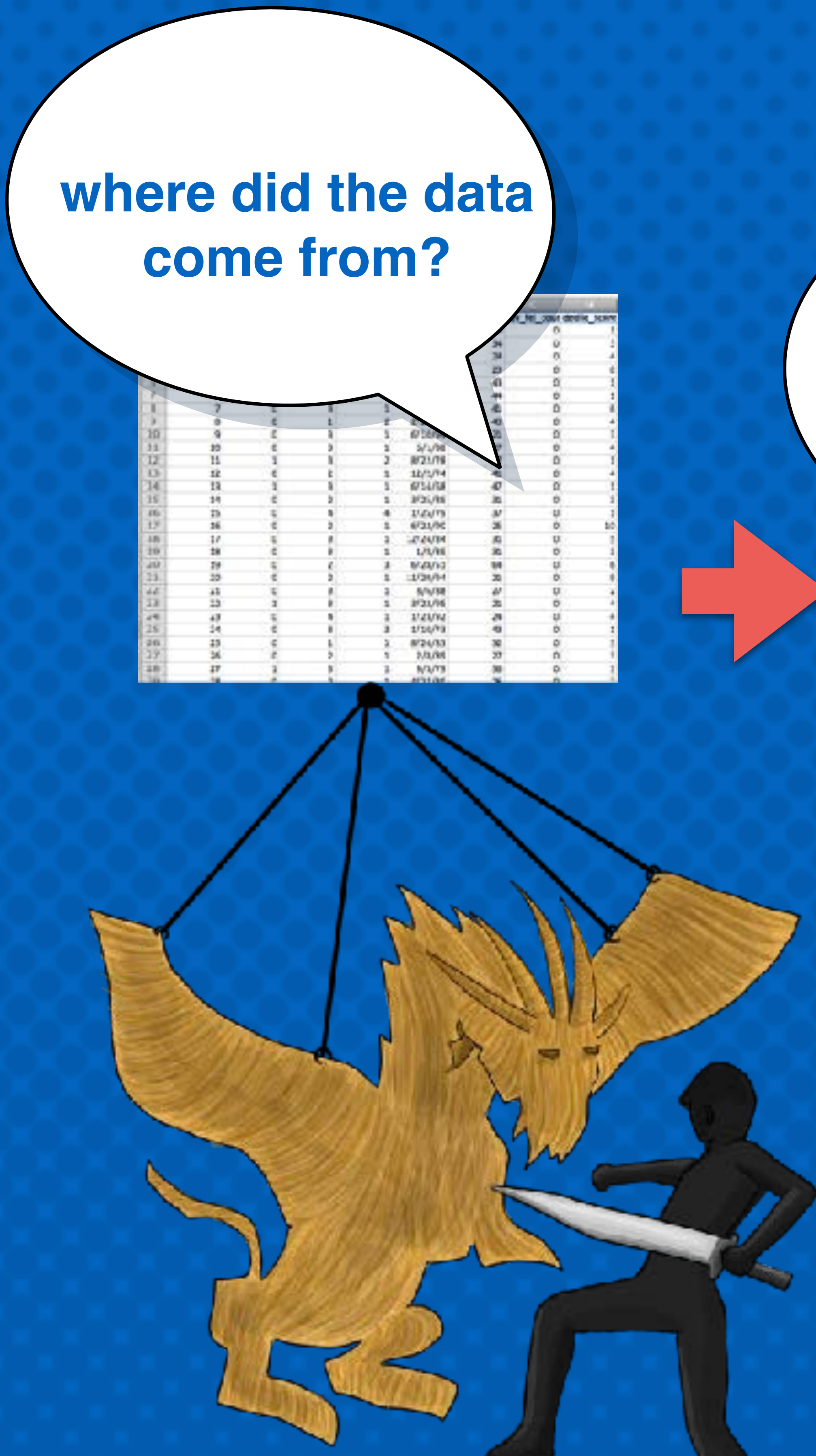


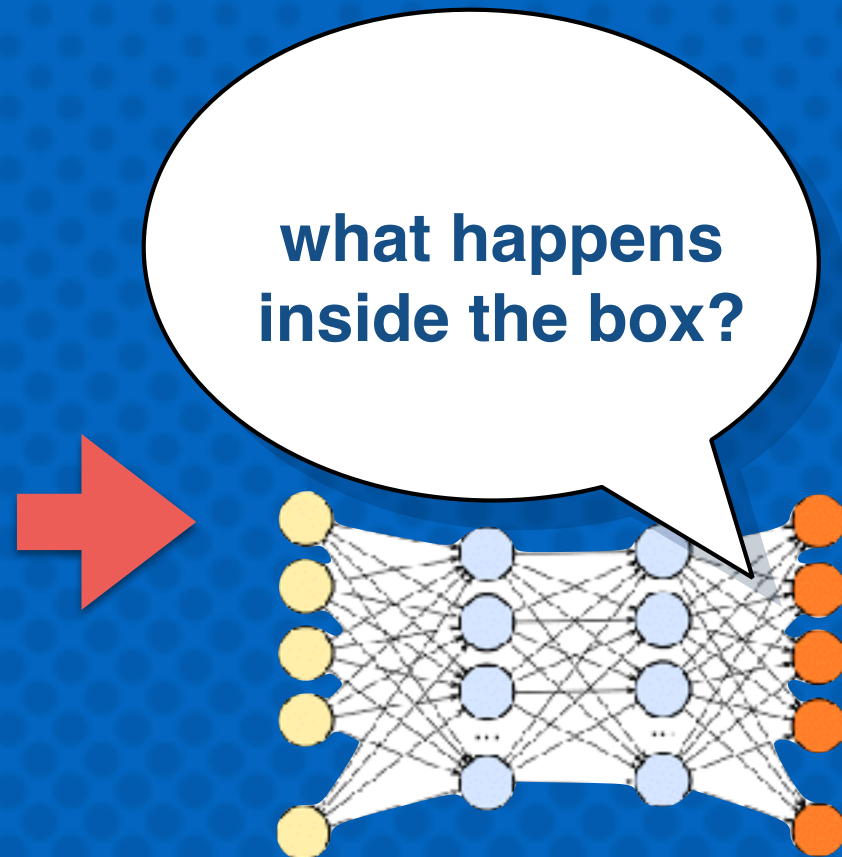
to fight bias, state beliefs and assumptions explicitly

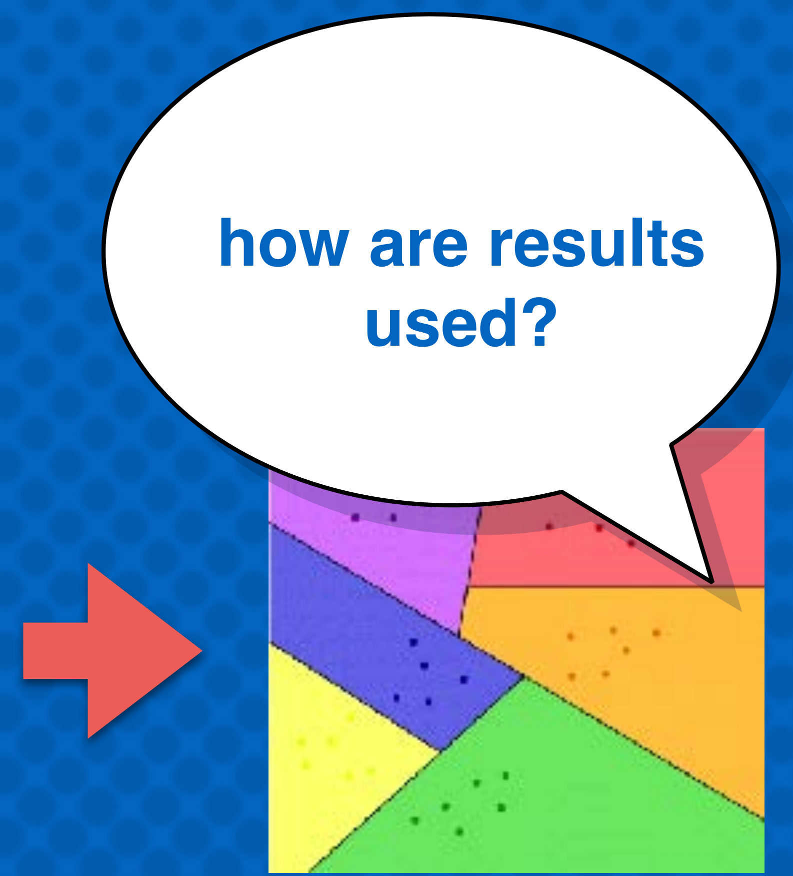
[Friedman & Nissenbaum (1996)]

r/ai center for responsible ai

The “last-mile” view of responsible AI

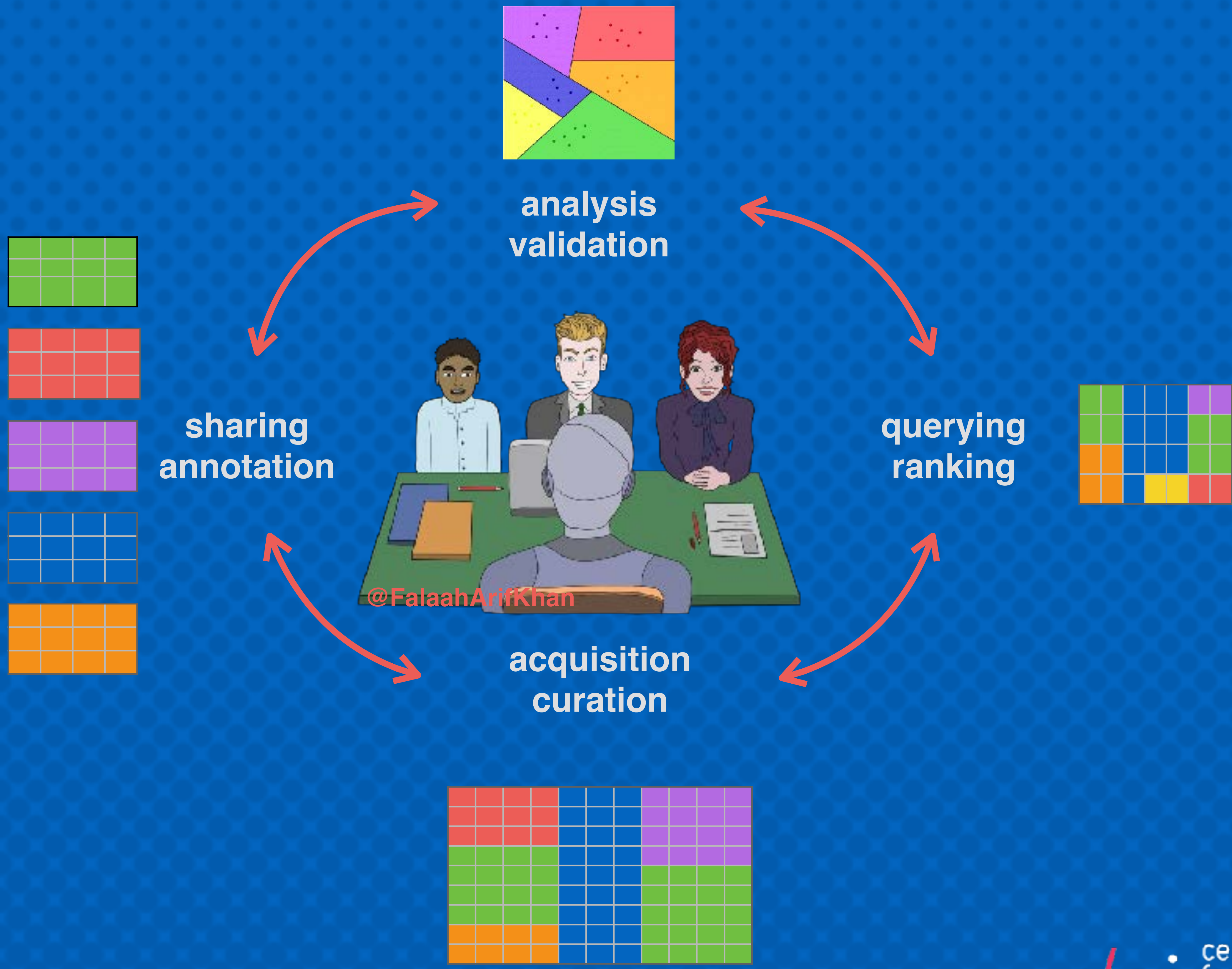






r/ai center for responsible ai

Data lifecycle of an ADS



r/ai center for responsible ai

Understand your data!

"Given the heterogeneity of the flood of data, it is **not enough merely to record it and throw it into a repository**. Consider, for example, data from a range of scientific experiments. If we just have a bunch of data sets in a repository, it is **unlikely anyone will ever be able to find, let alone reuse**, any of this data. With adequate **metadata**, there is some hope, but even so, challenges will remain due to differences in experimental details and in data record structure."

https://cra.org/ccc/wp-content/uploads/sites/2/2015/05/bigdatawhitepaper.pdf

Understand your data!

2.2 Big data

In the analog age, most of the data that were used for social research was created for the purpose of doing research. In the digital age, however, a huge amount of **data is being created by companies and governments for purposes other than research**, such as providing services, generating profit, and administering laws. Creative people, however, have realized that you can **repurpose** this corporate and government data for research.

https://www.bitbybitbook.com/en/1st-ed/observing-behavior/data/

center for responsible ai

Understand your data!

2.2 Big data

… from the perspective of researchers, big data sources are "found," they don't just fall from the sky. Instead, data sources that are "found" by researchers are **designed by someone for some purpose**. Because "found" data are designed by someone, I always recommend that you **try to understand as much as possible about the people and processes that created your data**.

https://www.bitbybitbook.com/en/1st-ed/observing-behavior/data/

Understand your data!

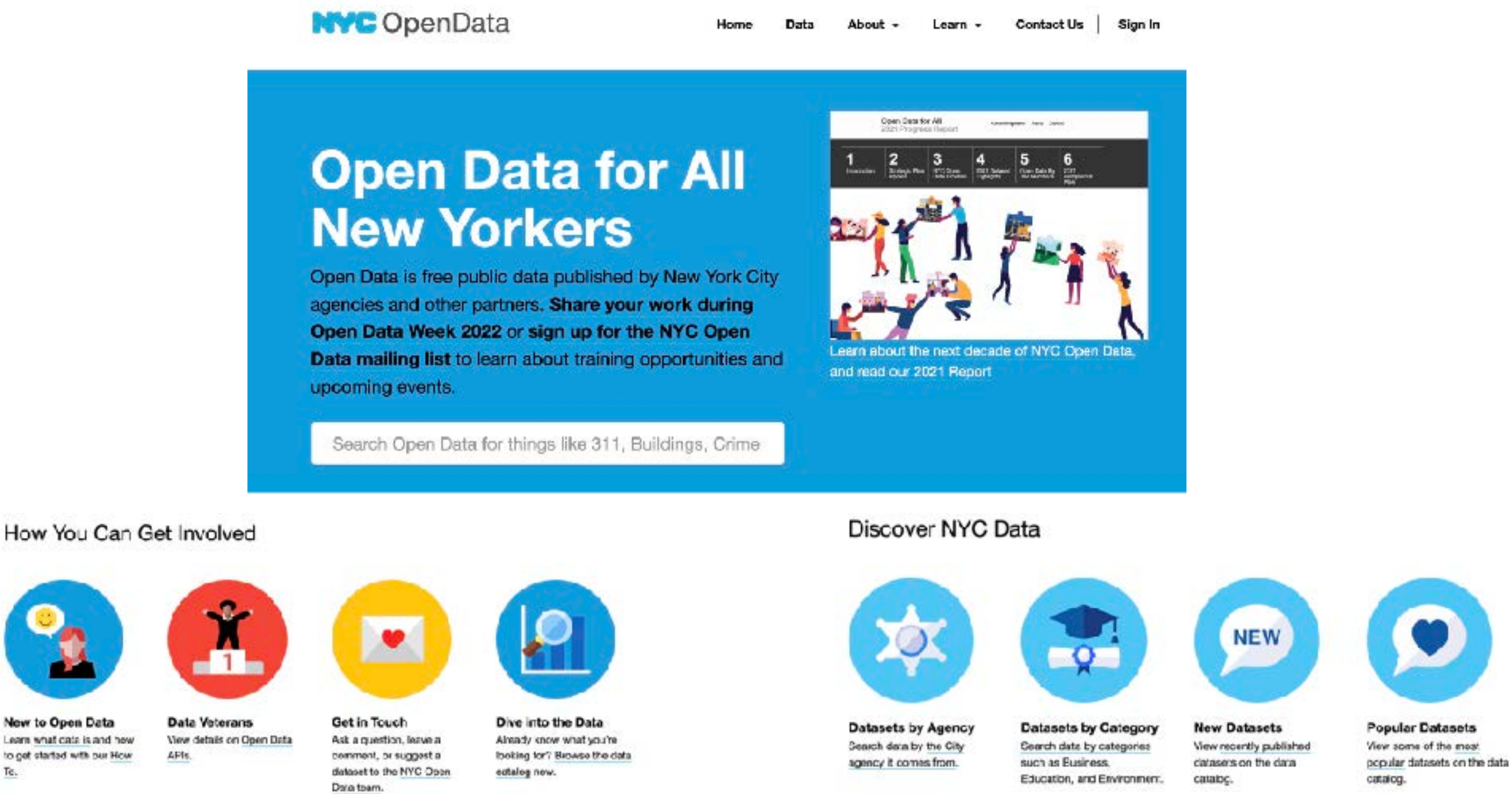
Need **metadata** to:

- enable data **re-use** (have to be able to find it!)
- determine **fitness for use** of a dataset in a task
- help establish **trust** in the data analysis process and its outcomes

Data is considered to be of high quality if it's "**fit for intended uses** in operations, decision making and planning"

[Thomas C. Redman, "Data Driven: Profiting from Your Most Important Business Asset." 2013]

NYC Open Data



https://opendata.cityofnewyork.us/

center for responsible ai

NYC Open Data

SAT (College Board) 2010 School Level Results

Education

Dataset

freshness

New York City school level College Board SAT results for the graduating seniors of 2010. Records contain 2010 College-bound seniors mean SAT scores.

summary

Records with 5 or fewer students are suppressed (marked 's').

privacy

Updated
April 25, 2019

Views
28,463

popularity

College-bound seniors are those students that complete the SAT Questionnaire when they register for the SAT and identify that they will graduate from high school in a specific year. For example, the 2010 college-bound seniors are those students that self-reported they would graduate in 2010. Students are not required to complete the SAT Questionnaire in order to register for the SAT. Students who do not indicate which year they will graduate from high school will not be included in any college-bound senior report.

Students are linked to schools by identifying which school they attend when registering for a College Board exam. A student is only included in a school's report if he/she self-reports being enrolled at that school.

Data collected and processed by the College Board.

source

Less

Tags *No tags assigned*

API Docs

NYC Open Data

About this Dataset

Updated

April 25, 2019

Data Last Updated
February 29, 2012

Metadata Last Updated
April 25, 2019

Date Created
October 6, 2011

Views
28.5K

Downloads
48.4K

Data Provided by
Department of Education (DOE)

Dataset Owner
NYC OpenData

Update

Update Frequency	Historical Data
Automation	No
Date Made Public	10/11/2011

Dataset Information

Agency	Department of Education (DOE)

Attachments

SAT Data Dictionary.xlsx

Topics

Category	Education
Tags	*This dataset does not have any tags*

https://opendata.cityofnewyork.us/

center for responsible ai

NYC Open Data

What's in this Dataset?

Rows	Columns
460	6

Columns in this Dataset

Column Name	Description	Type
DBN		Plain Text T
School Name		Plain Text T
Number of Test Takers		Number #
Critical Reading Mean		Number #
Mathematics Mean		Number #
Writing Mean		Number #

https://opendata.cityofnewyork.us/

NYC Open Data

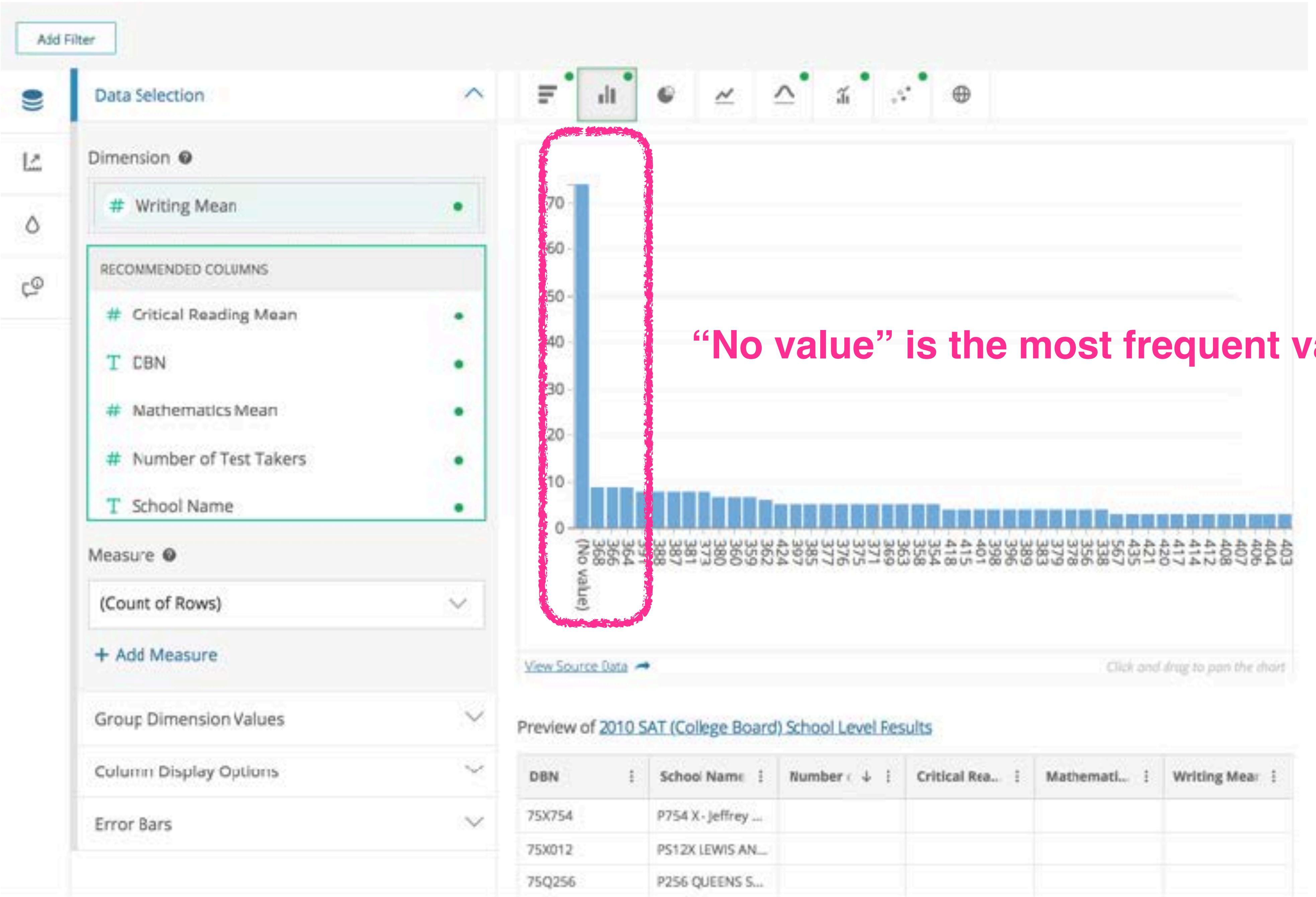
What's in this Dataset?

Rows	Columns
460	6

Columns in this Dataset

Column Name	Description	Type
DBN		Plain Text T
School Name		Plain Text T
Number of Test Takers		Number #
Critical Reading Mean		Number #
Mathematics Mean		Number #
Writing Mean		Number #

NYC Open Data

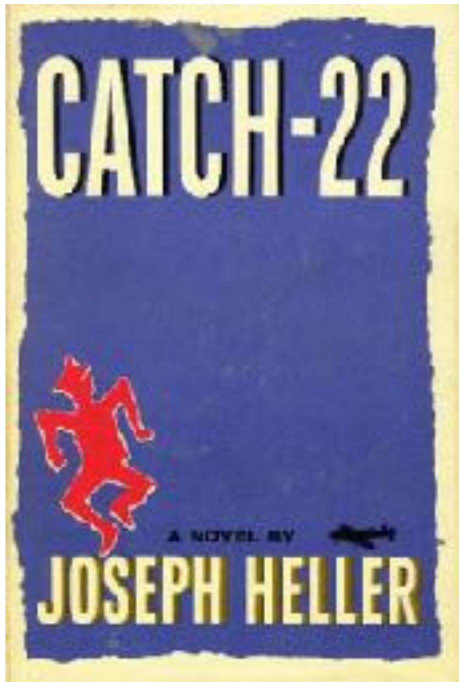


“No value” is the most frequent value

https://opendata.cityofnewyork.us/

Data profiling

- **Data profiling** refers to the activity of creating **small** but **informative** summaries of a database
- What is informative depends on the task, or set of tasks, we have in mind



should profiling be task-agnostic or task-specific?

A related activity is **data cleaning**

Data cleaning

Data cleansing or **data cleaning** is the process of detecting and repairing corrupt or inaccurate records from a data set in order to improve the **quality of data**.

Erhard Rahm, Hong Hai Do: Data Cleaning: Problems and Current Approaches, IEEE Data Engineering Bulletin, 2000.

… **data** is generally considered high **quality** if it is "**fit for [its] intended uses** in operations, decision making and planning“

Thomas C. Redman, Data Driven: Profiting from Your Most Important Business Asset. 2013

Even though quality cannot be defined, you know what it is.

Robert M. Prisig, Zen and the Art of Motorcycle Maintenance, 1975

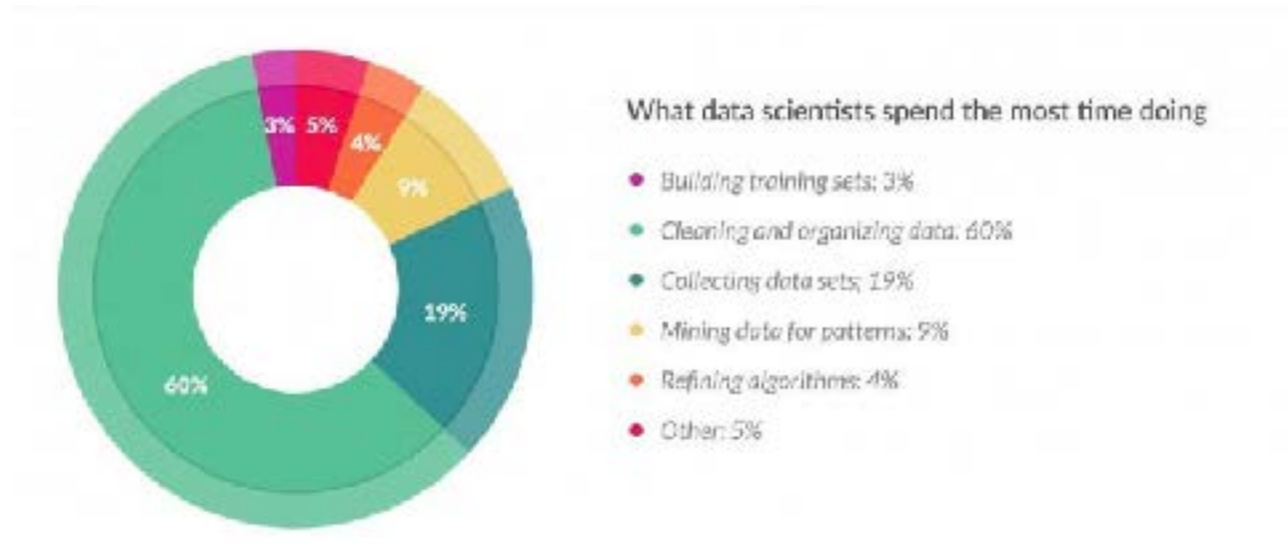
Data cleaning


Forbes

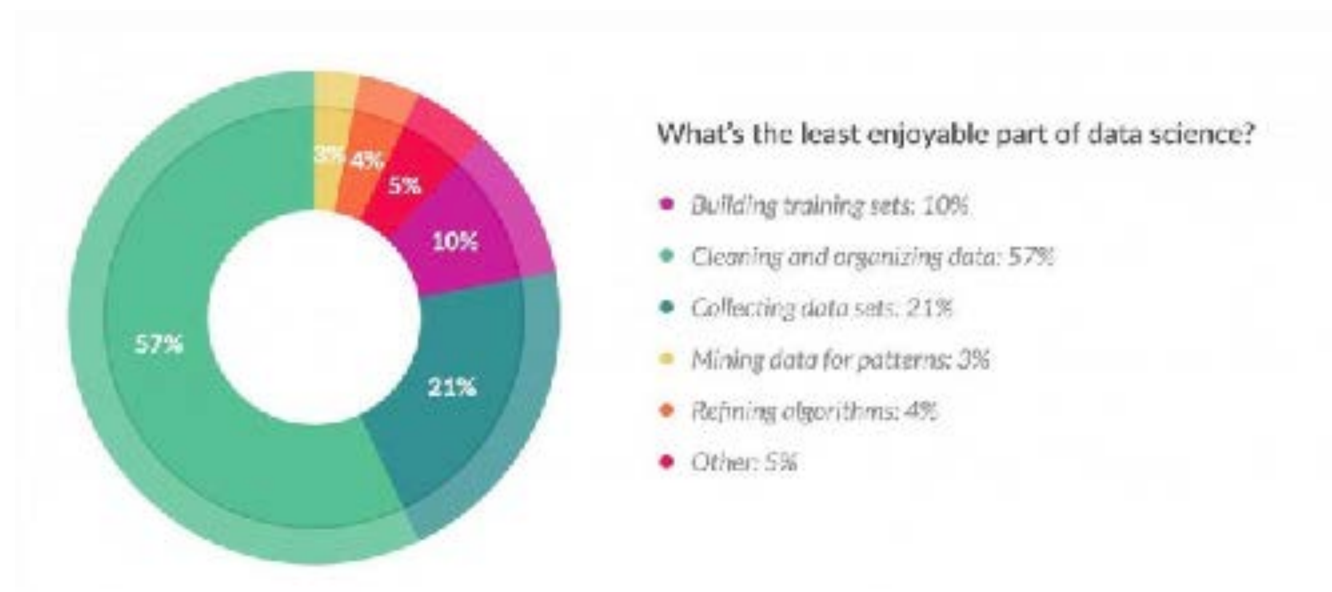
Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says

Gil Press Contributor

I write about technology, entrepreneurs and innovation.



Spend most time doing
Collecting data (19%)
Cleaning and organizing data (60%)



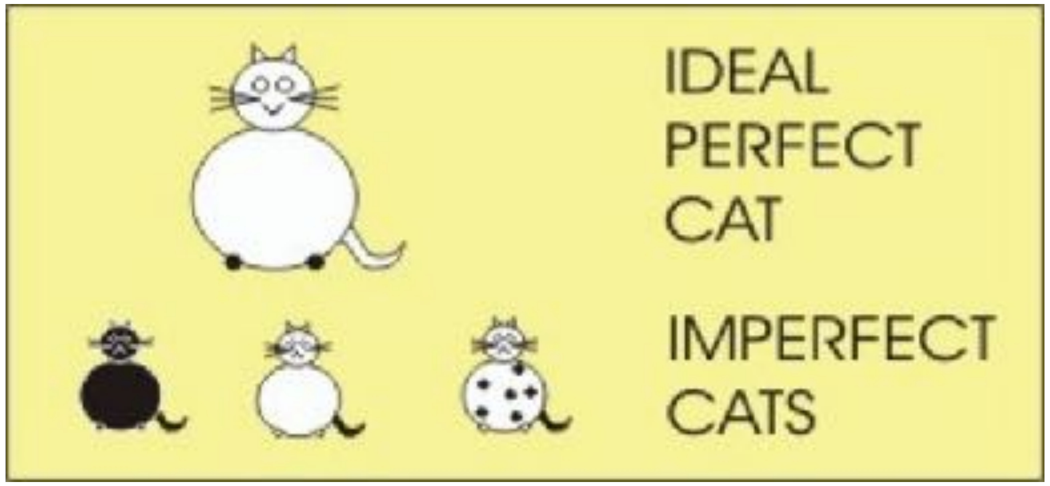
Find least enjoyable
Collecting data (21%)
Cleaning and organizing data (57%)

center for responsible ai

data profiling

r/ai center for responsible ai

DB (databases) vs DS (data science)



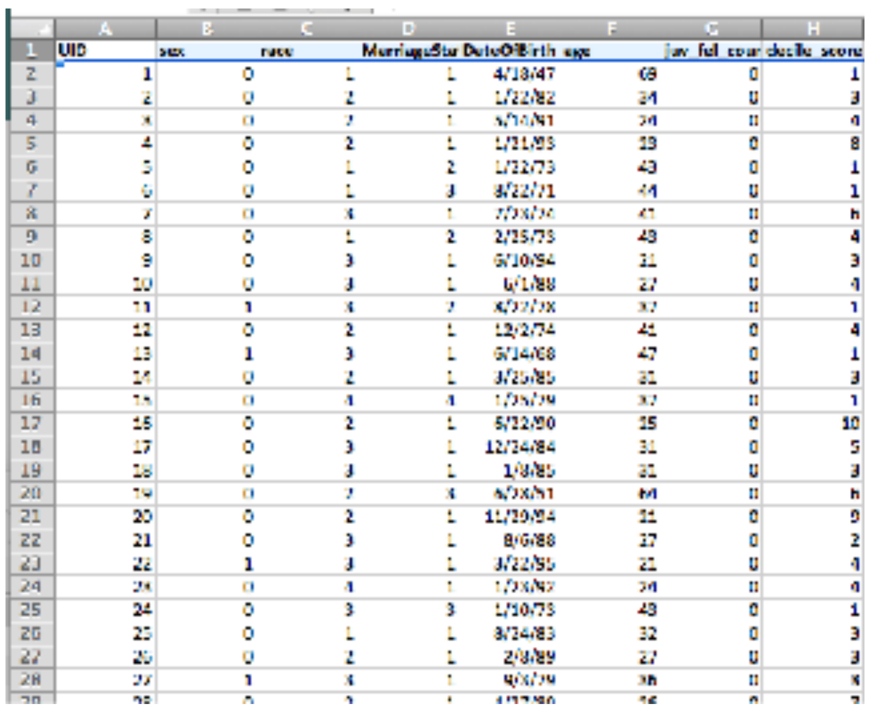
https://midnightmediamusings.wordpress.com/2014/07/01/plato-and-the-theory-of-forms/

- **DB**: start with the schema, admit only data that fits; iterative refinement is possible, and common, but we are still schema-first
- **DS**: start with the data, figure out what schema it fits, or almost fits - reasons of usability, repurposing, low start-up cost

the "right" approach is somewhere between these two, **data profiling aims to bridge** between the two world views / methodologies

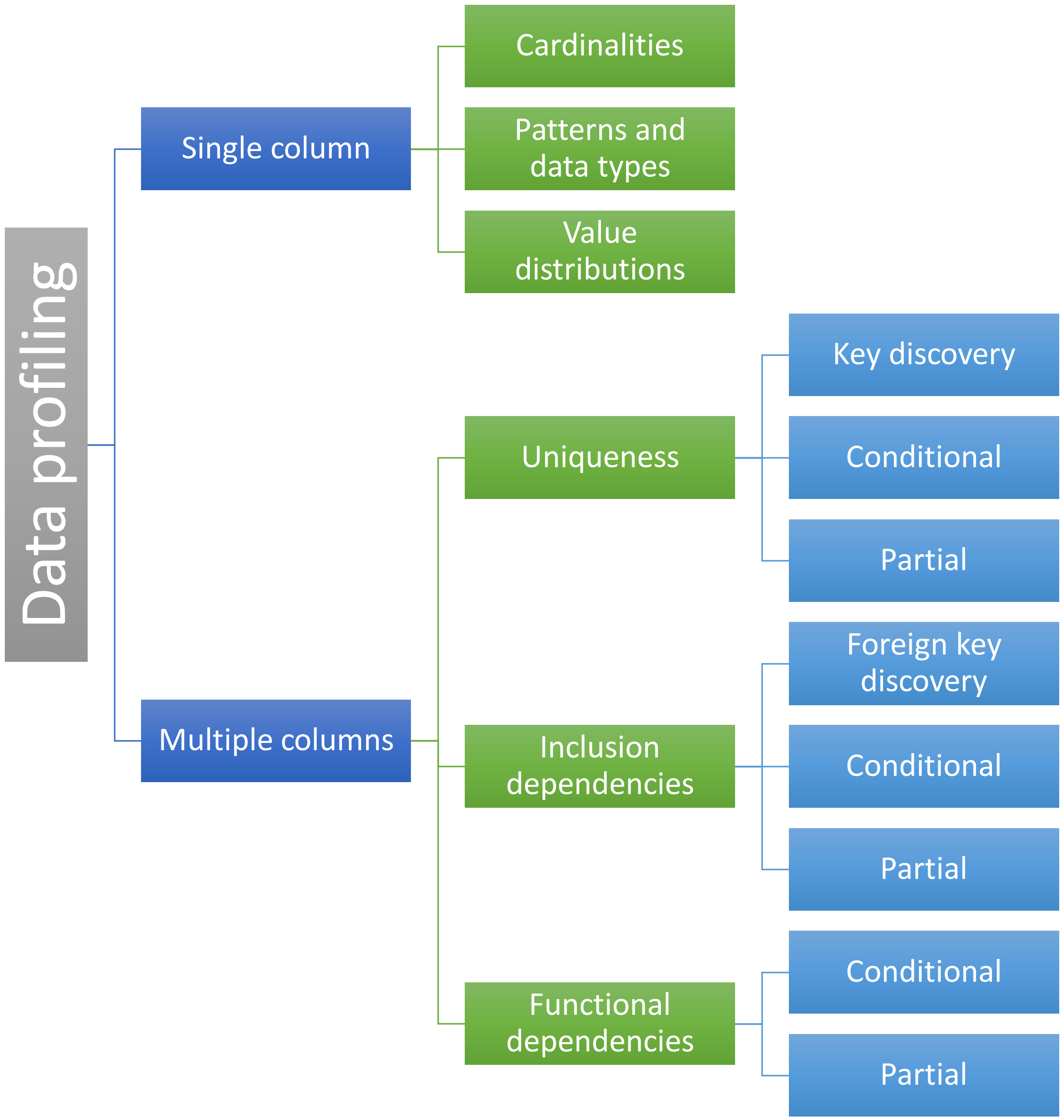
[Abedjan, Golab & Naumann (2017)]

Data profiling



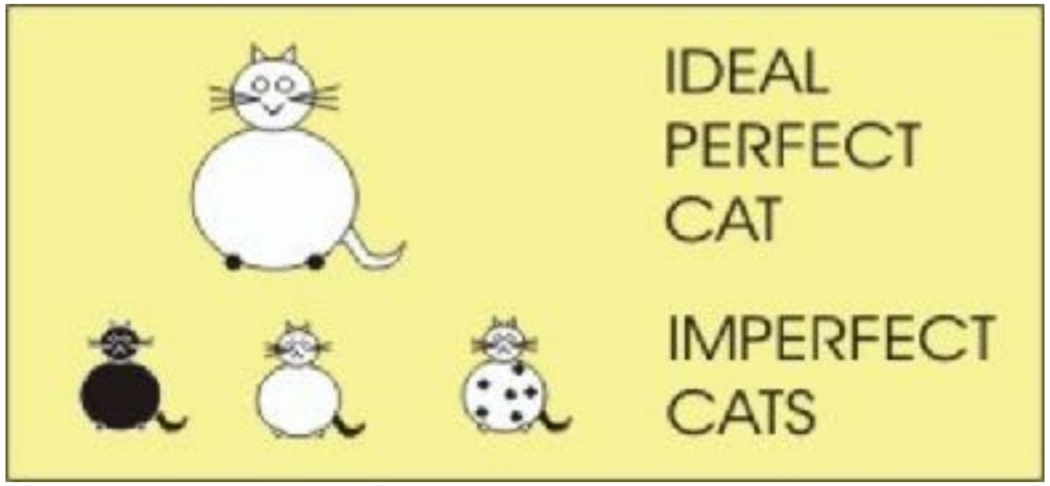
	UID	sex	race	MarriageSta	DateOfBirth	age	juv_fel_cour	decile_score
2	1	0	1	1	4/18/47	69	0	1
3	2	0	2	1	1/22/82	34	0	3
4	3	0	[illegible]	1	[illegible]	34	0	4
5	4	0	2	1	1/21/93	23	0	8
6	5	0	1	2	1/22/73	43	0	1
7	6	0	1	3	8/22/71	44	0	1
8	7	0	[illegible]	1	[illegible]	[illegible]	0	[illegible]
9	8	0	1	2	2/25/73	43	0	4
10	9	0	3	1	6/10/94	21	0	3
11	10	0	3	1	6/1/88	27	0	4
12	11	1	[illegible]	[illegible]	[illegible]	[illegible]	0	1
13	12	0	2	1	12/2/74	41	0	4
14	13	1	3	1	6/14/68	47	0	1
15	14	0	2	1	3/25/85	31	0	3
16	15	0	4	4	[illegible]	[illegible]	0	1
17	16	0	2	1	5/22/90	25	0	10
18	17	0	3	1	12/24/84	31	0	5
19	18	0	3	1	1/8/85	31	0	3
20	19	0	[illegible]	[illegible]	[illegible]	[illegible]	0	[illegible]
21	20	0	2	1	11/20/94	21	0	9
22	21	0	3	1	8/6/88	27	0	2
23	22	1	3	1	3/22/95	21	0	4
24	23	0	4	1	[illegible]	[illegible]	0	4
25	24	0	3	3	1/10/73	43	0	1
26	25	0	1	1	3/24/83	32	0	3
27	26	0	2	1	2/8/89	27	0	3
28	27	1	[illegible]	1	[illegible]	[illegible]	0	[illegible]

relational data (here: just one table)



[Abedjan, Golab & Naumann (2017)]

An alternative classification



- To help understand the **statistics**, we look at value ranges, data types, value distributions per column or across columns, etc

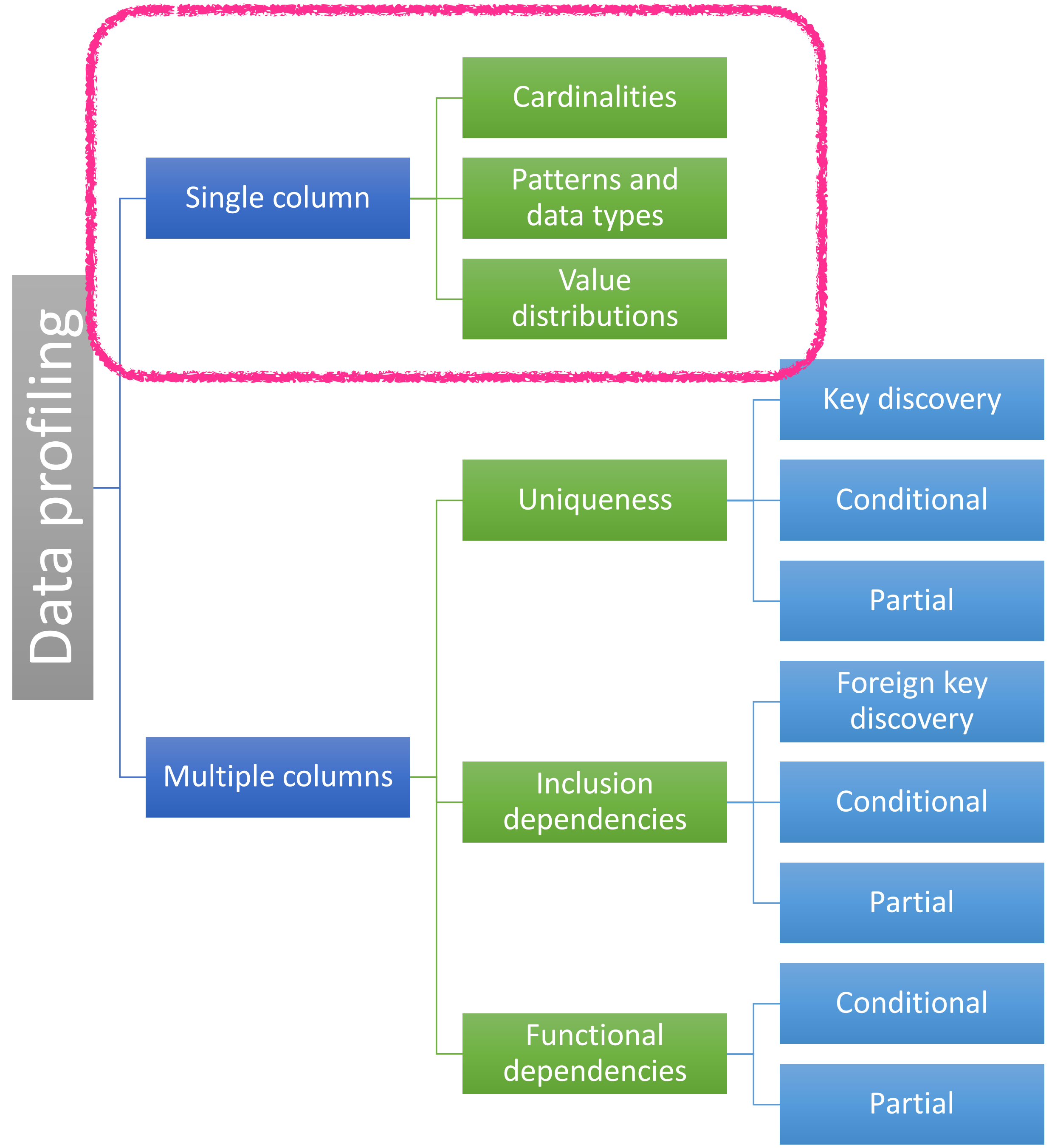
- To help understand the **structure** - the (business) rules that generated the data - we look at unique columns / column combinations, dependencies between columns, etc - **reverse-engineer the relational schema** of the data we have

- We need both statistics and structure, they are mutually-reinforcing, and help us understand the **semantics** of the data - it's meaning

Data profiling

	A	B	C	D	E	F	G	H
1	UID	sex	race	MarriageStatus	DateOfBirth	age	juv_fel_count	decile_score
2	1	0	1	1	4/18/47	69	0	1
3	2	0	2	1	1/22/82	34	0	3
4	3	0	2	1	[illegible]	24	0	4
5	4	0	2	1	1/21/93	23	0	8
6	5	0	1	2	1/22/73	43	0	1
7	6	0	1	3	8/22/71	44	0	1
8	7	0	3	1	[illegible]	47	0	6
9	8	0	1	2	2/25/73	43	0	4
10	9	0	3	1	6/10/94	21	0	3
11	10	0	3	1	6/1/88	27	0	4
12	11	1	3	2	[illegible]	37	0	1
13	12	0	2	1	12/2/74	41	0	4
14	13	1	3	1	6/14/68	47	0	1
15	14	0	2	1	3/25/85	31	0	3
16	15	0	4	4	[illegible]	37	0	1
17	16	0	2	1	5/22/90	25	0	10
18	17	0	3	1	12/24/84	31	0	5
19	18	0	3	1	1/8/85	31	0	3
20	19	0	2	3	[illegible]	64	0	6
21	20	0	2	1	11/29/94	21	0	9
22	21	0	3	1	8/6/88	27	0	2
23	22	1	3	1	3/22/95	21	0	4
24	23	0	4	1	[illegible]	24	0	4
25	24	0	3	3	1/10/73	43	0	1
26	25	0	1	1	8/24/83	32	0	3
27	26	0	2	1	2/8/89	27	0	3
28	27	1	3	1	[illegible]	36	0	8

relational data (here: just one table)



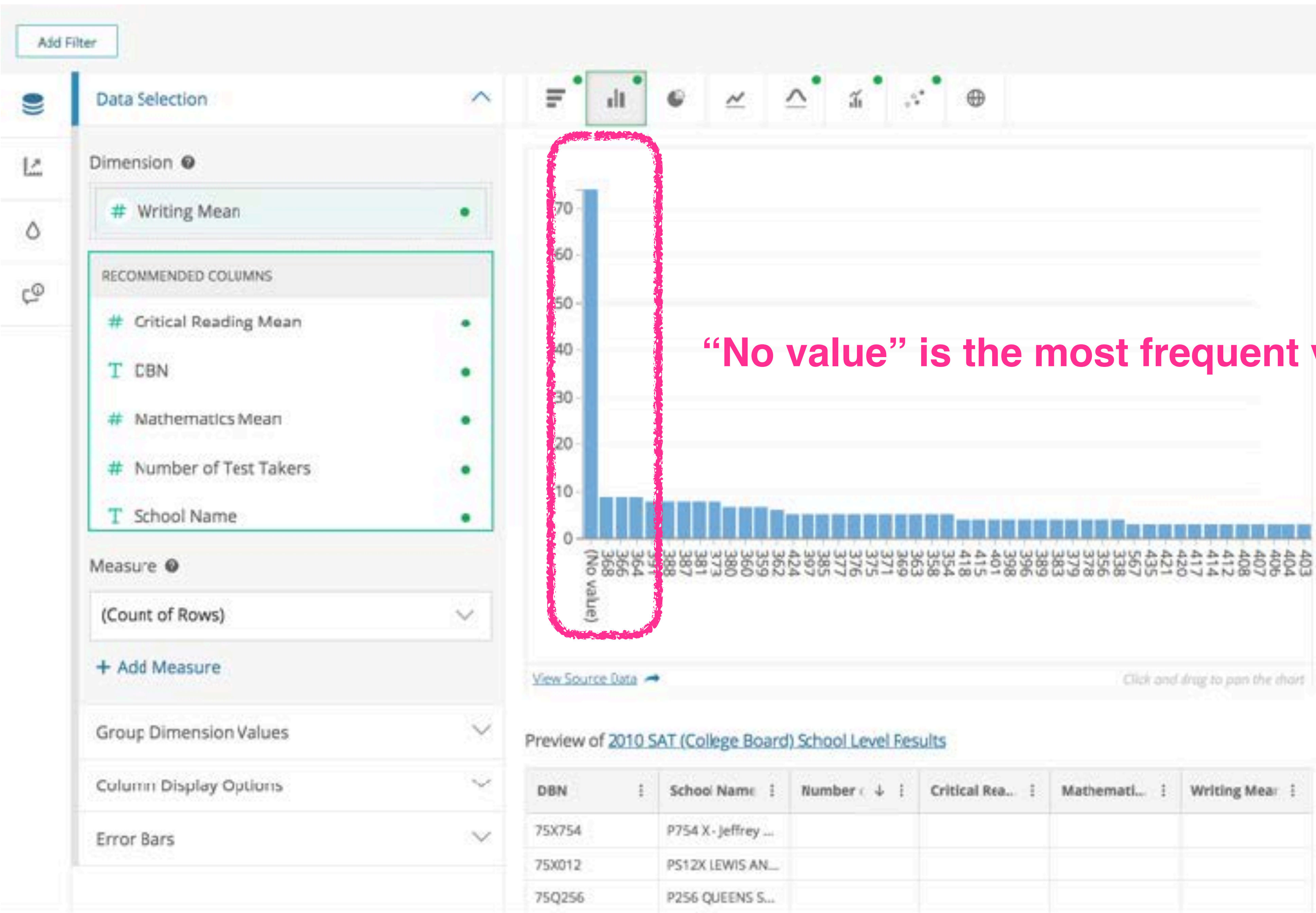
[Abedjan, Golab & Naumann (2017)]

Single column: cardinalities, data types

- cardinality of relation **R** - number of rows
- domain cardinality of a column **R.a** - number of **distinct** values
- attribute value **length**: min, max, average, median
- **basic data type**: string, numeric, date, time, ….
- number of percentage of **null** values of a given attribute
- regular expressions
- semantic domain: SSN, phone number
- ….

[Abedjan, Golab & Naumann (2017)]

NYC Open Data



“No value” is the most frequent value

https://opendata.cityofnewyork.us/

The trouble with *null* values

A CRITIQUE OF

THE SQL DATABASE LANGUAGE

C.J.Date

PO Box 2647, Saratoga
California 95070, USA

December 1983

* Null values

I have argued against null values at length elsewhere [6], and I will not repeat those arguments here. In my opinion the null value concept is far more trouble than it is worth. Certainly it has never been properly thought through in the existing SQL implementations (see the discussion under "Lack of Orthogonality: Miscellaneous Items", earlier). For example, the fact that functions such as AVG simply ignore null values in their argument violates what should surely be a fundamental principle, viz: The system should never produce a (spuriously) precise answer to a query when the data involved in that query is itself imprecise. At least the system should offer the user the explicit option either to ignore nulls or to treat their presence as an exception.

50 shades of *null*

- **Unknown** - some value definitely belongs here, but I don't know what it is (e.g., unknown birthdate)
- **Inapplicable** - no value makes sense here (e.g., if marital status = single then spouse name should not have a value)
- **Unintentionally omitted** - values is left unspecified unintentionally, by mistake
- **Optional** - a value may legitimately be left unspecified (e.g., middle name)
- **Intentionally withheld** (e.g., an unlisted phone number)
- …..

(this selection is mine, see reference below for a slightly different list)

https://www.vertabelo.com/blog/technical-articles/50-shades-of-null-or-how-a-billion-dollar-mistake-has-been-stalking-a-whole-industry-for-decades

50 shades of *null*… and it gets worse

- **Hidden missing values** -
  - 99999 for zip code, Alabama for state
  - need data cleaning….
- lots of houses in Philadelphia, PA were built in 1934 (or 1936?) - not really!

how do we detect hidden missing values?

Single column: cardinalities, data types

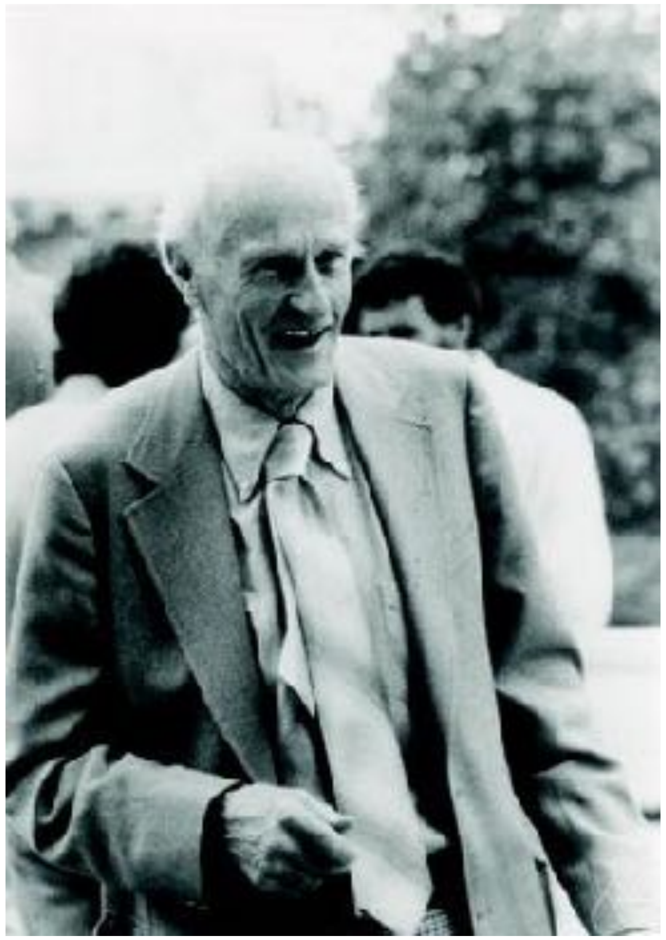
- cardinality of relation **R** - number of rows
- domain cardinality of a column **R.a** - number of **distinct** values
- attribute value **length**: min, max, average, median
- **basic data type**: string, numeric, date, time, ….
- number of percentage of **null** values of a given attribute
- **regular expressions**
- semantic domain: SSN, phone number
- ….

[Abedjan, Golab & Naumann (2015)]

Regular expressions

- some attributes will have values that follow a regular format, e.g, telephone numbers: 212-864-0355 or (212) 864-0355 or 1.212.864-0355
- we may want to identify a small set of **regular expressions** that match all (or most) values in a column
- challenging - **very many possibilities!**

A **regular expression**, **regex** or **regexp** … is a sequence of characters that define a search pattern. Usually this pattern is used by string searching algorithms for “find” or “find and replace” operations on strings, or for input validation. It is a technique that developed in theoretical computer science and formal language theory.



Stephen Kleene

https://en.wikipedia.org/wiki/Regular_expression

r/ai center for responsible ai

Inferring regular expressions

- we may want to identify a small set of **regular expressions** that match all (or most) values in a column
- challenging - **very many possibilities!**

telephone
(201) 368-1000
(201)373-9599
(718) 206-1088
(718) 206-1121
(718) 206-1420
(718) 206-4420
(718) 206-4481
(718) 262-9072
(718) 868-2300
(718) 206-0545
(814) 681-6200
(888) 8NYC-TRS
800-624-4143

Example Regular Expression Language

.	Matches any character
abc	Sequence of characters
[abc]	Matches any of the characters inside **[ ]**
*	Previous character matched zero or more times
?	Previous character matched zero or one time
{m}	Exactly **m** repetitions of previous character
^	Matches beginning of a line
$	Matches end of a line
\d	Matches any decimal digit
\s	Matches any whitespace character
\w	Matches any alphanumeric character

Oakham's razor

Lex parsimoniae

If multiple hypotheses explain an observation, the simplest one should be preferred.

Ockham's motivation: can one prove the existence of God?

Used as a heuristic to help identify a promising hypothesis to test

Many applications today: biology, probability theory, ethics - also good for inferring regular expressions :)

William of Ockham
(1285-1347)

Inferring regular expressions

telephone
800-624-4143
(201)373-9599
(201) 368-1000
(718) 206-1088
(718) 206-1121
(718) 206-1420
(718) 206-4420
(718) 206-4481
(718) 262-9072
(718) 868-2300
(718) 206-0545
(814) 681-6200
(888) 8NYC-TRS

Simple Algorithm

(1) Group values by length

(2) Find pattern for each group

- Ignore small groups
- Find **most specific character** at each position

(	2	0	1	)		3	6	8	-	1	0	0	0
(	2	0	1	)		2	0	6	-	1	0	8	8
(	7	1	8	)		2	0	6	-	1	1	2	1
(	7	1	8	)		2	0	6	-	1	4	2	0
(	7	1	8	)		2	0	6	-	4	4	2	0
(	7	1	8	)		2	0	6	-	4	4	8	1
(	7	1	8	)		2	6	2	-	9	0	7	2
(	7	1	8	)		8	6	8	-	2	3	0	0
(	7	1	8	)		2	0	6	-	0	5	4	5
(	8	1	4	)		6	8	1	-	6	2	0	0
(	8	8	8	)		8	N	Y	C	-	T	R	S

(	\d	\d	\d	)		\d	\w	\w	.	.	\w	\w	\w

Inferring regular expressions

telephone
800-624-4143
(201)373-9599
(201) 368-1000
(718) 206-1088
(718) 206-1121
(718) 206-1420
(718) 206-4420
(718) 206-4481
(718) 262-9072
(718) 868-2300
(718) 206-0545
(814) 681-6200
(888) 8NYC-TRS

Simple Algorithm

(1) Group values by length

(2) Find pattern for each group

- Ignore small groups
- Find **most specific character** at each position

ignoring small groups: alternatives?

(	\d	\d	\d	)		\d	\w	\w	.	.	\w	\w	\w

```
(\d{3}) \d\w{2}.{2}\w{3}
```

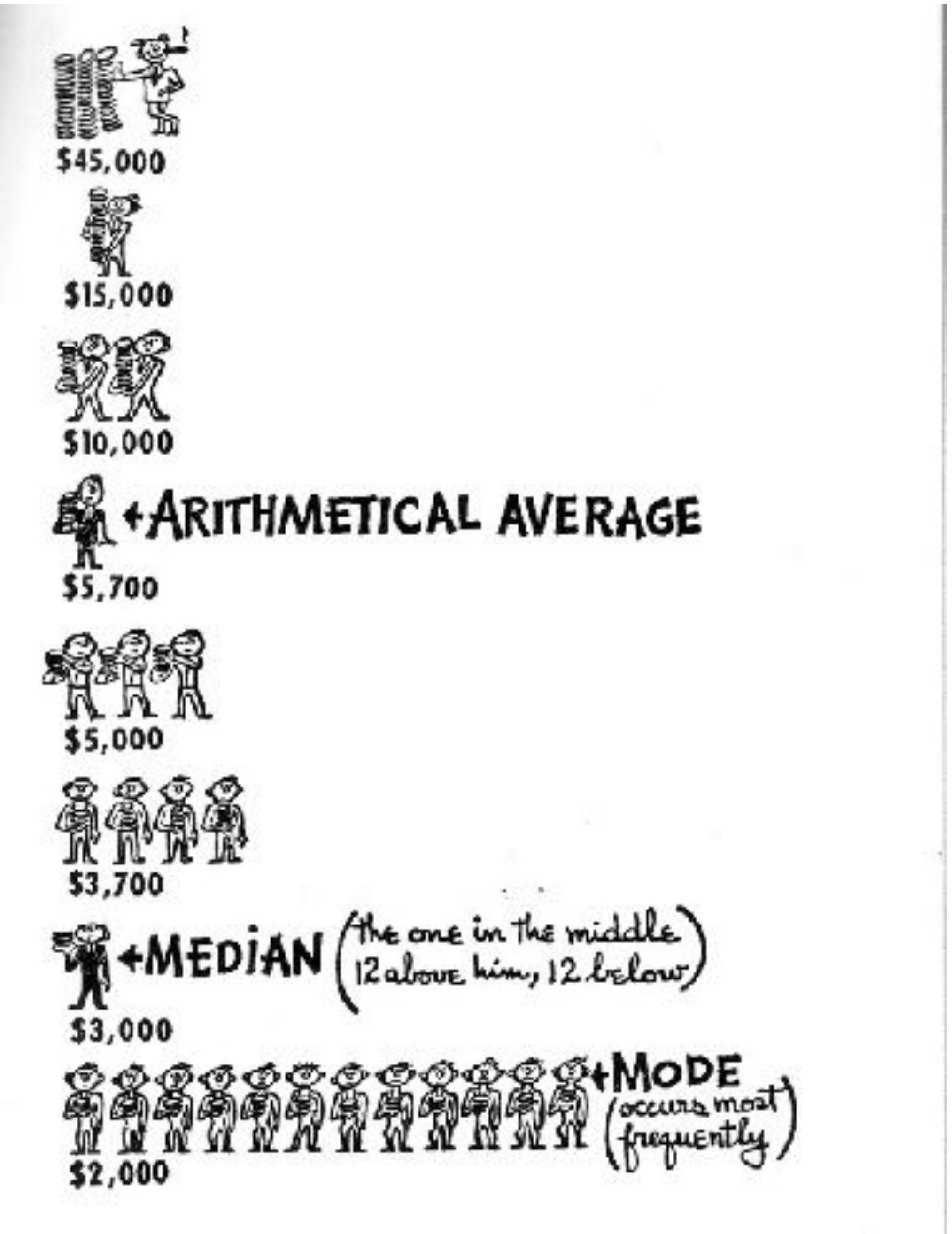
based on a slide by Heiko Mueller

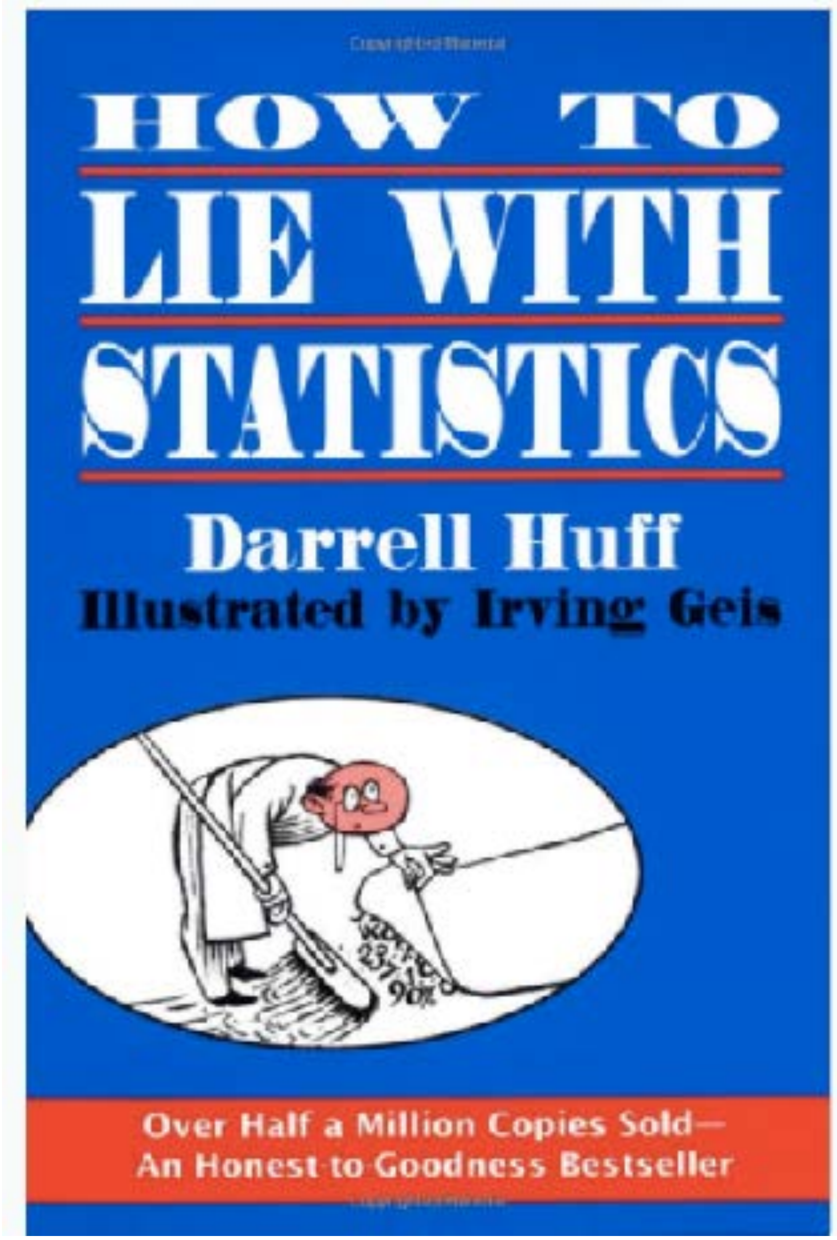
Single column: basic stats, distributions

- min, max, **average**, median value of **R.a**
- **histogram**
    - equi-width - (approximately) the same number of distinct values in each bucket (e.g., age broken down into 5-year windows)
    - equi-depth (approximately) the same number of tuples in each bucket
    - biased histograms use different granularities for different parts of the value range to provide better accuracy
- quartiles - three points that divide the numeric values into four equal groups - a kind of an equi-depth histogram
- **first digit** - distribution of first digit in numeric values, to check Benford law
- …

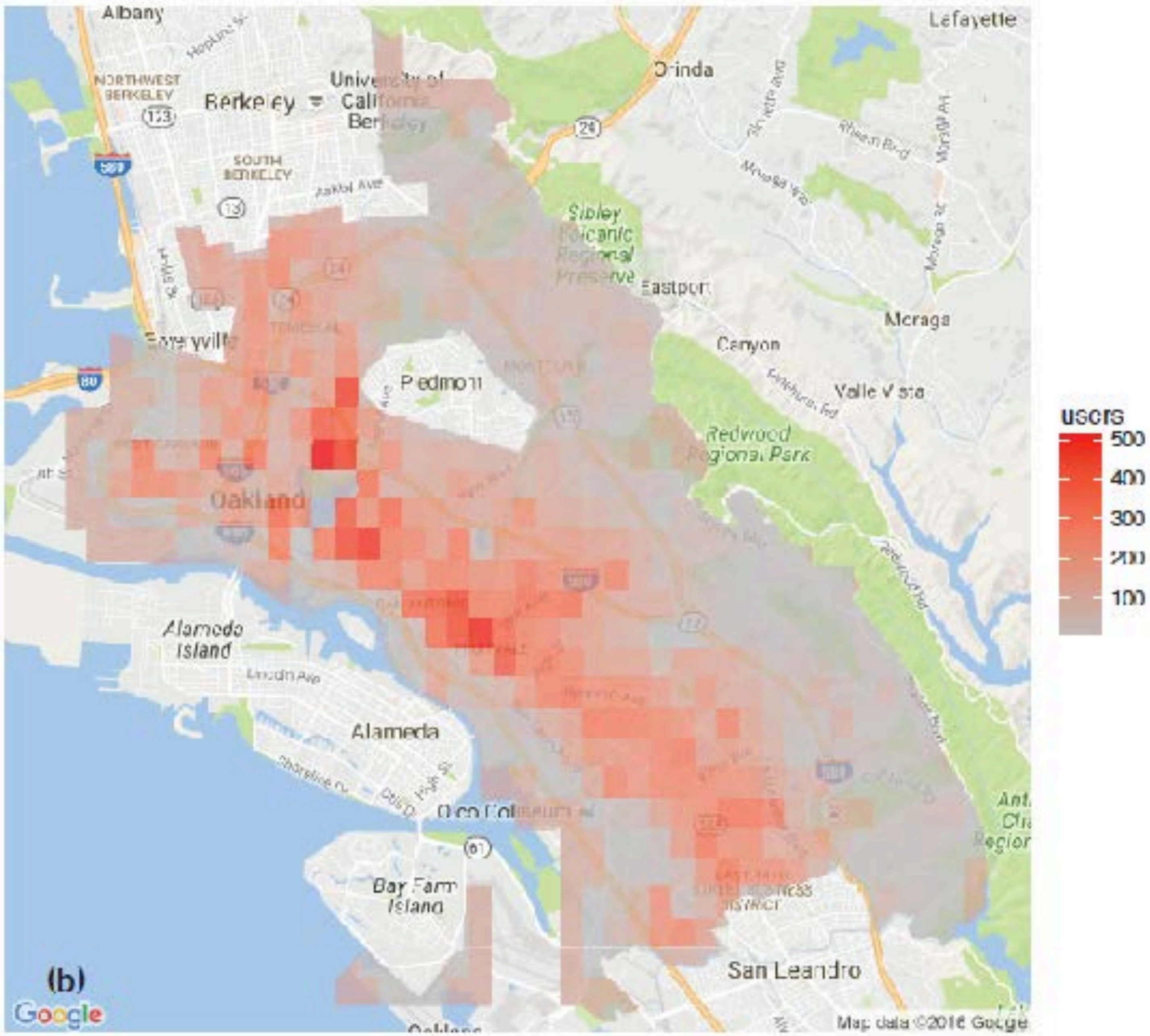
[Abedjan, Golab & Naumann (2015)]

The well-chosen average

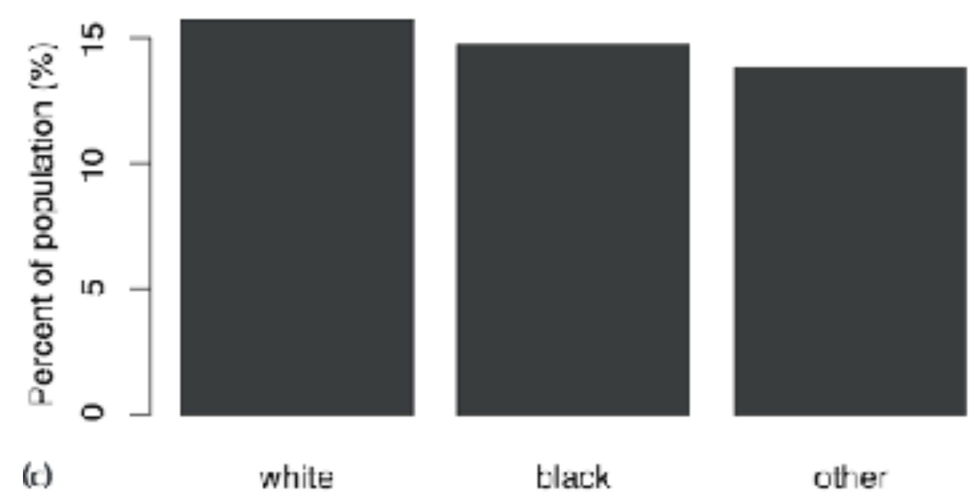




Is my data biased? (histograms + geo)



Estimated number of drug users, based on 2011 National Survey on Drug Use and Health, in Oakland, CA

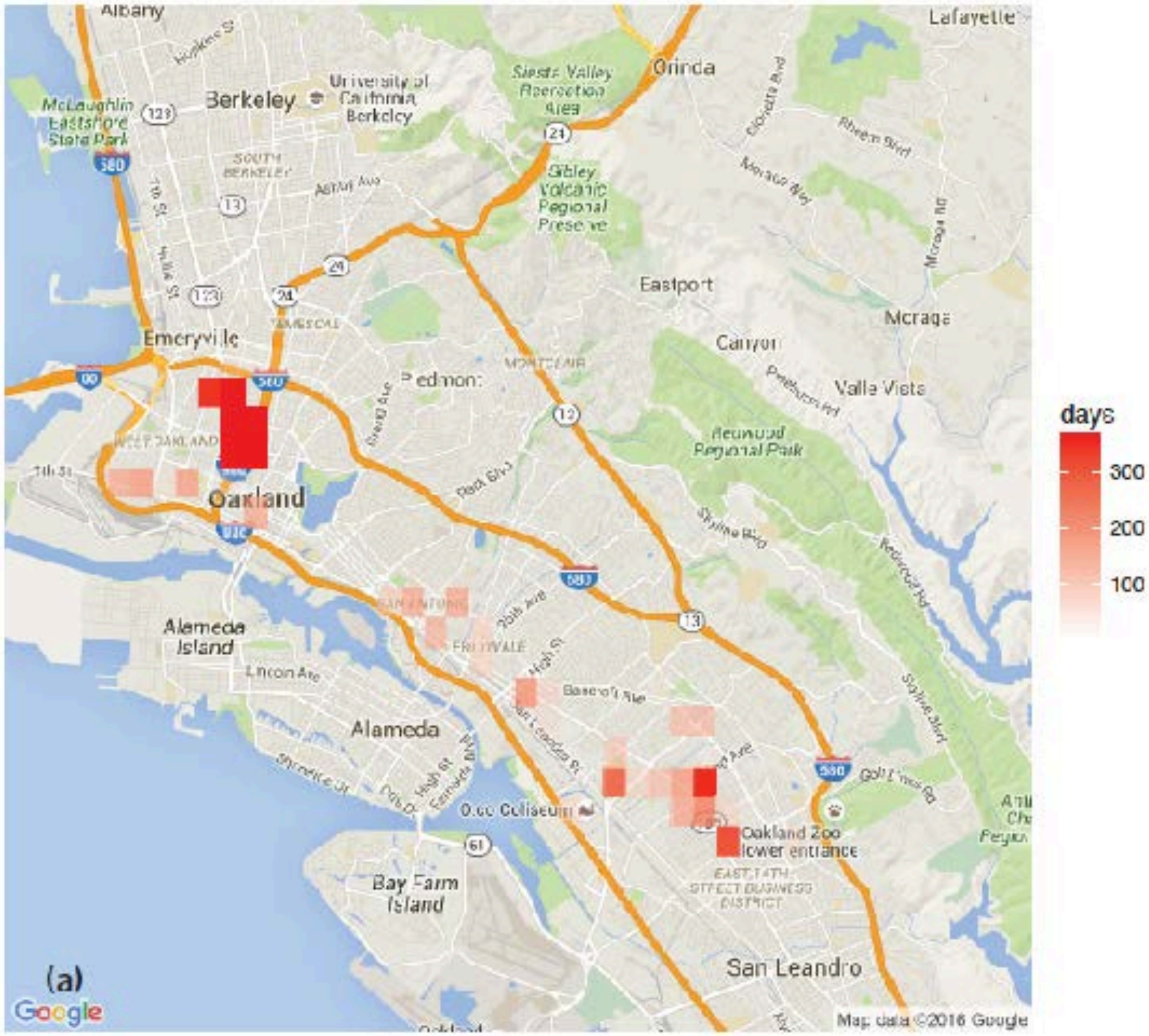


Estimated drug use by race

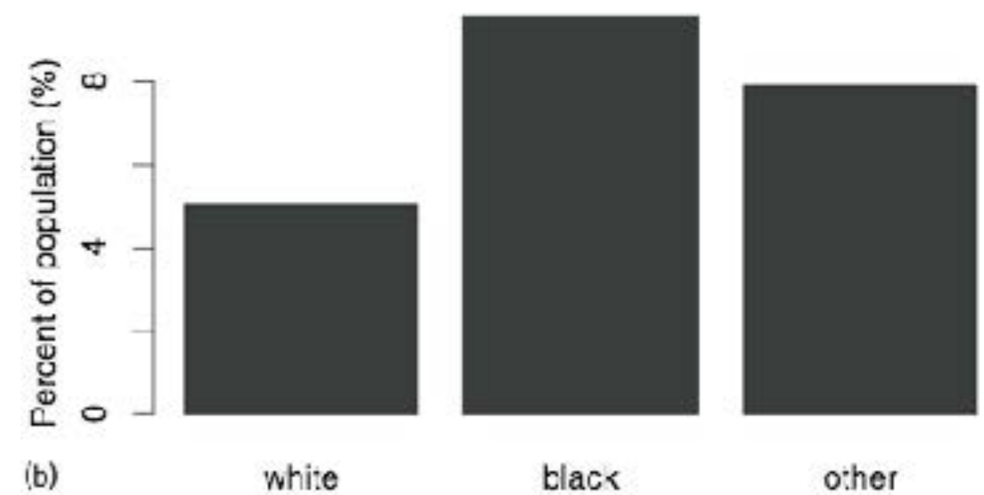
[Lum & Isaac (2016)]

r/ai center for responsible ai

Is my data biased? (histograms + geo)

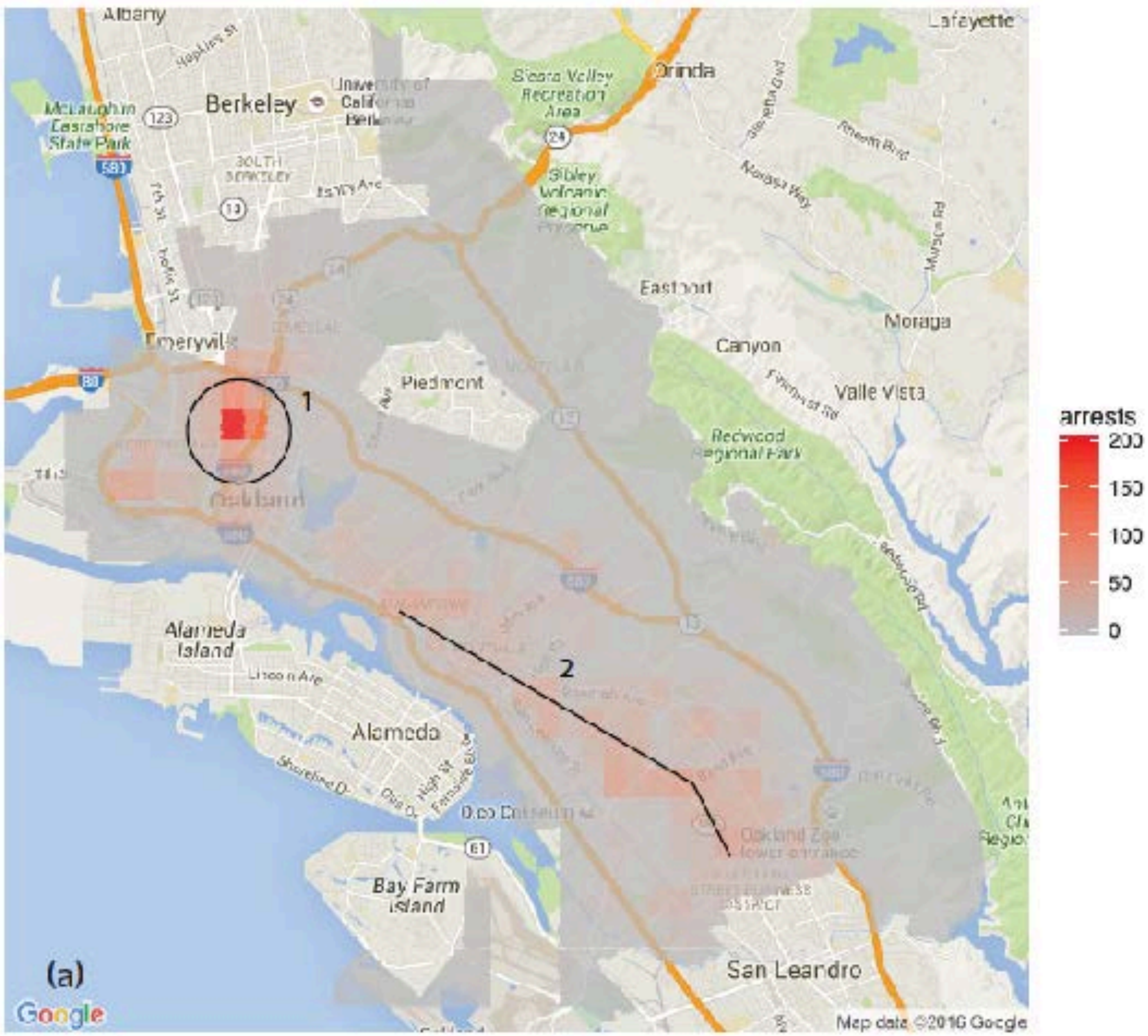


Number of days with targeted policing for drug crimes in areas flagged by PredPol analysis of Oakland, CA, police data for 2011

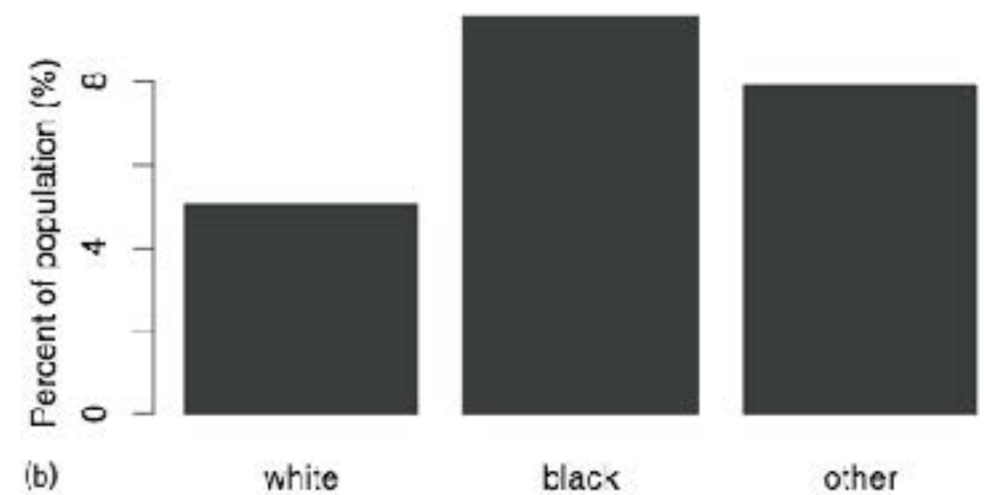


Targeted policing for drug crimes by race

Is my data biased? (histograms + geo)



Number of drug arrests made by the Oakland, CA, police department in 2010

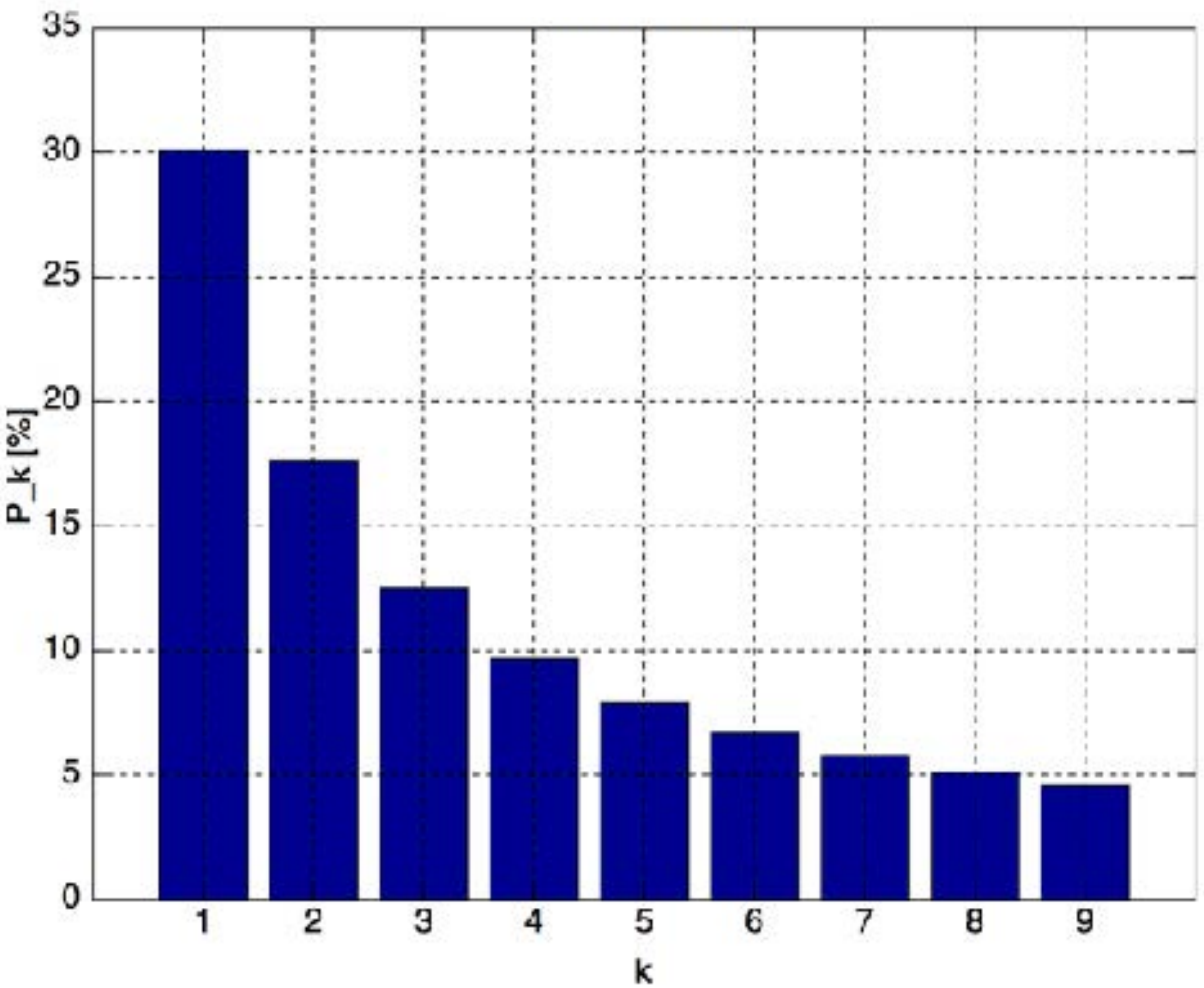


Targeted policing for drug crimes by race

Benford Law

The distribution of **the first digit d** of a number, in many naturally occurring domains, approximately follows

$$P(d) = \log_{10}\left(1 + \frac{1}{d}\right)$$



https://en.wikipedia.org/wiki/Benford%27s_law

1 is the most frequent leading digit, followed by 2, etc.

[Benford: “The law of anomalous numbers” *Proc. Am. Philos. Soc.*, 1938]

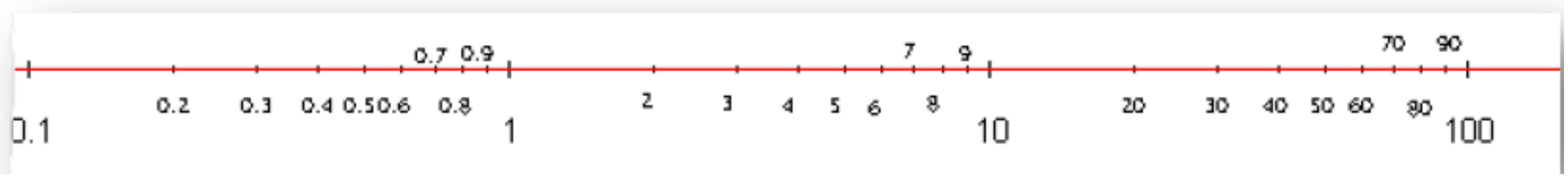
r/ai center for responsible ai

Benford Law

The distribution of **the first digit d** of a number, in many naturally occurring domains, approximately follows

$$P(d) = \log_{10}\left(1 + \frac{1}{d}\right)$$

Holds if **log(x) is uniformly distributed**. **Most accurate** when values are distributed across multiple orders of magnitude, especially **if the process generating the numbers is described by a power law** (common in nature)



A logarithmic scale bar. Picking a random x position uniformly on this number line, roughly 30% of the time the first digit of the number will be 1.

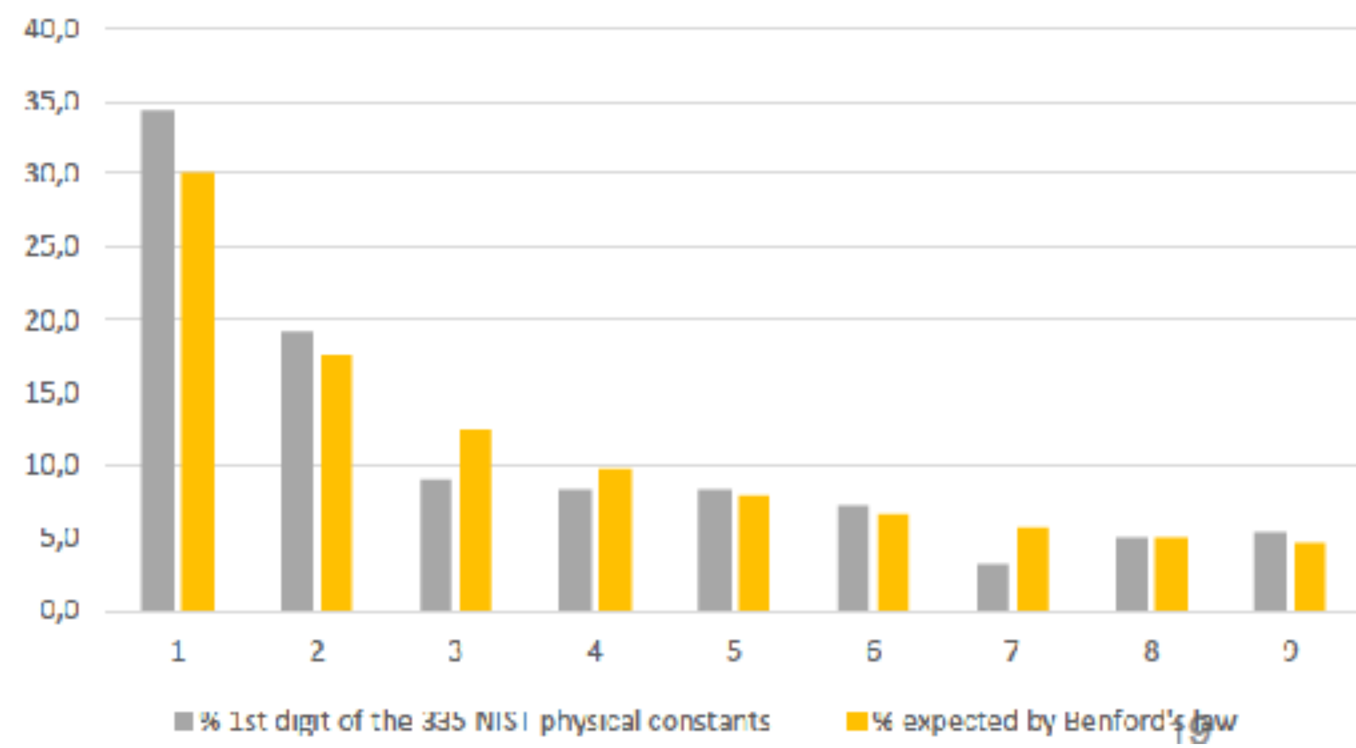
https://en.wikipedia.org/wiki/Benford%27s_law

[Benford: "The law of anomalous numbers" *Proc. Am. Philos. Soc.*, 1938]

Examples of Benford Law

- surface area of 355 rivers
- sizes of 3,259 US populations
- 104 physical constants
- 1,800 molecular weights
- 308 numbers contained in an issue of Reader's Digest
- Street addresses of the first 342 persons listed in American Men of Science
- ….

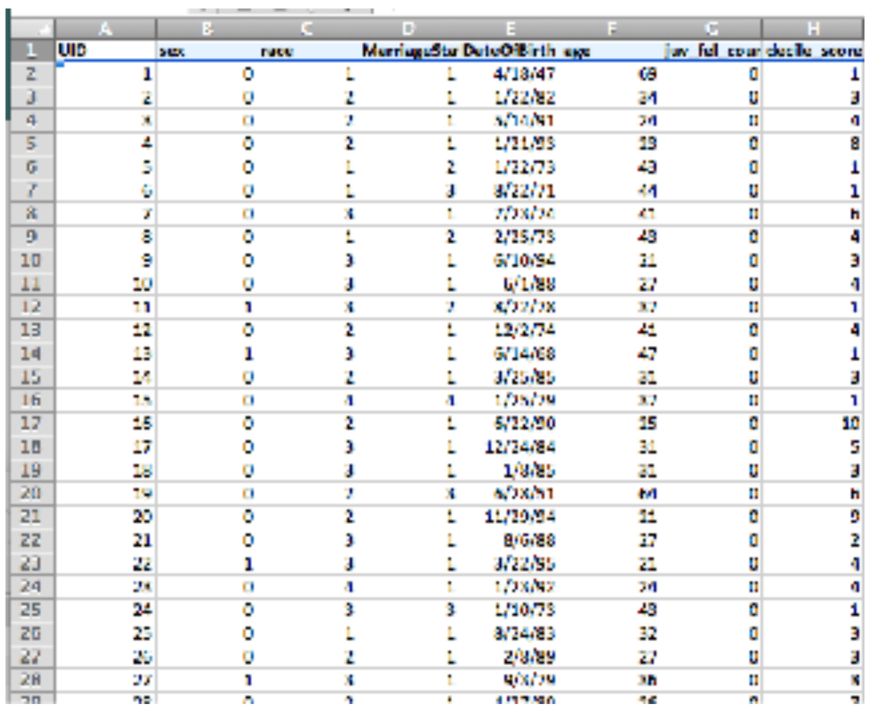
used in fraud detection!



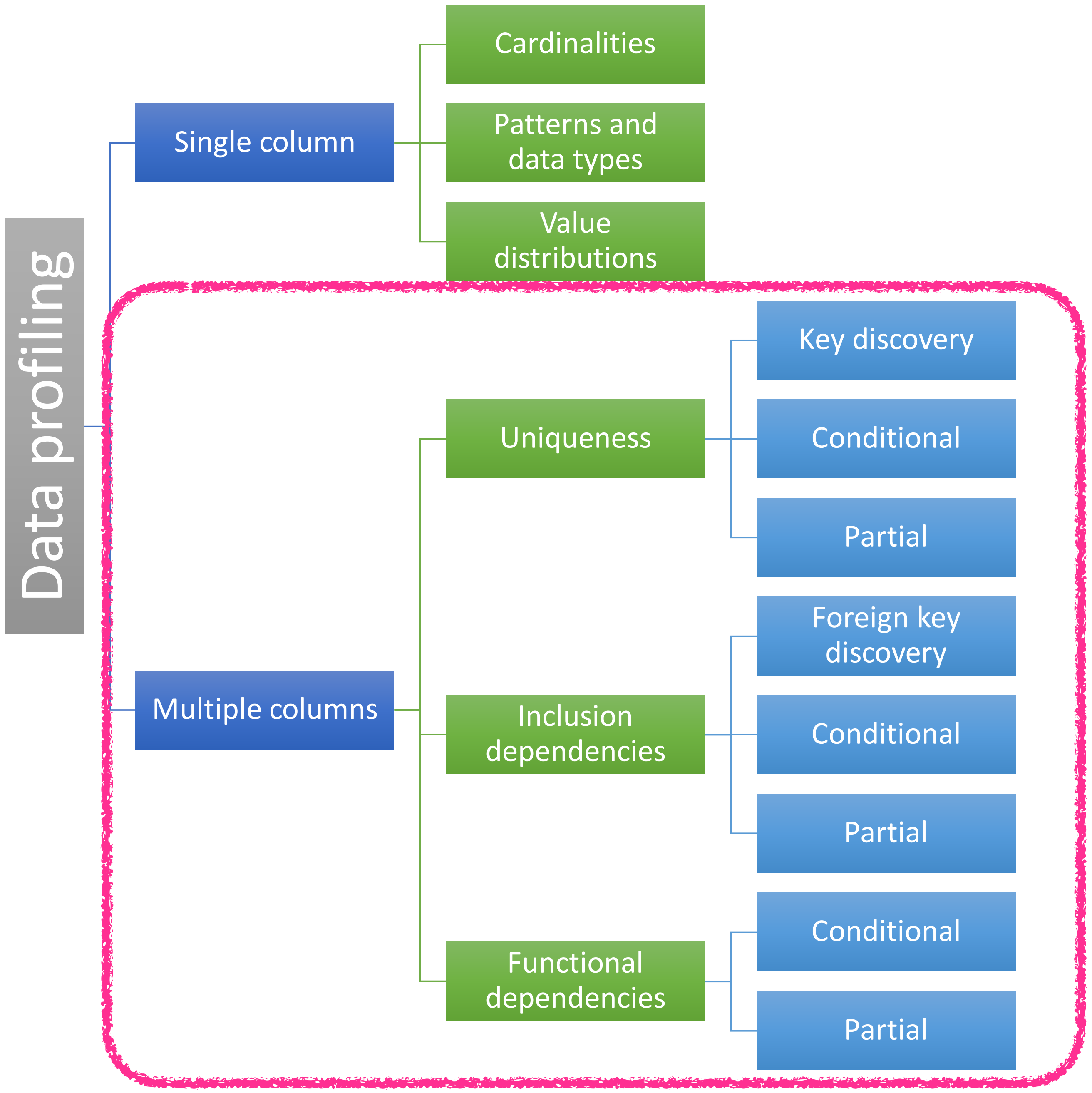
physical constants

center for responsible ai

Data profiling

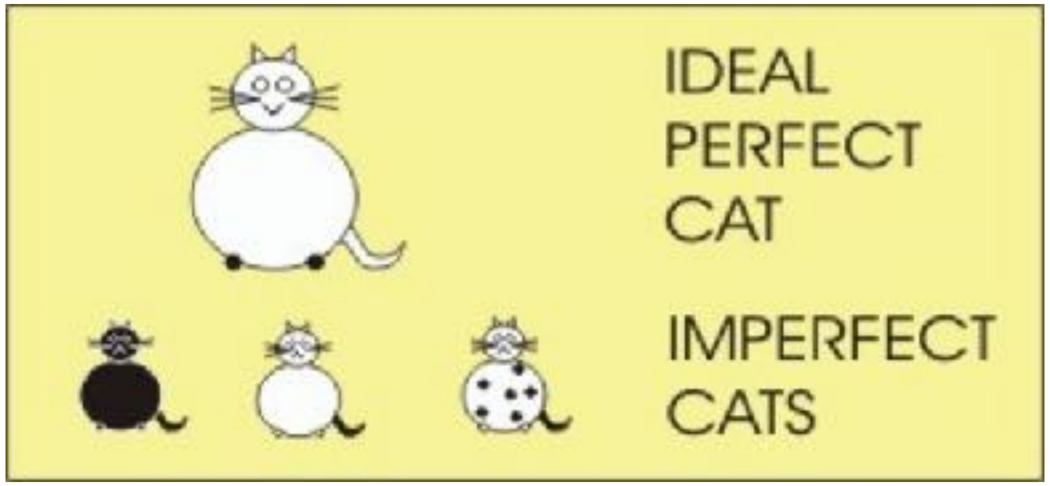


relational data (here: just one table)



[Abedjan, Golab & Naumann (2015)]

An alternative classification



- To help understand the **statistics**, we look at value ranges, data types, value distributions per column or across columns, etc
- To help understand the **structure** - the (business) rules that generated the data - we look at unique columns / column combinations, dependencies between columns, etc - **reverse-engineer the relational schema** of the data we have
- We need both statistics and structure, they are mutually-reinforcing, and help us understand the **semantics** of the data - it's meaning

discovering uniques

center for responsible ai

Discovering uniques

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a **unique column combination** (or a **"unique"** for short) is a set of attributes ***X*** whose **projection** contains no duplicates in ***r***

$Episodes(season, num, title, viewers)$

season	num	title	viewers
1	1	Winter is Coming	2.2 M
1	2	The Kingsroad	2.2 M
2	1	The North Remembers	3.9 M

Projection is a relational algebra operation that takes as input relation ***R*** and returns a new relation ***R'*** with a subset of the columns of ***R***.

$\pi_{season}(Episodes)$

season
1
1
2

non-unique

$\pi_{season,num}(Episodes)$

season	num
1	1
1	2
2	1

unique

$\pi_{title}(Episodes)$

title
Winter is Coming
The Kingsroad
The North Remembers

unique

Discovering uniques

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a **unique column combination** (or a **"unique"** for short) is a set of attributes ***X*** whose **projection** contains no duplicates in ***r***

Episodes(season,num,title,viewers)

season	num	title	viewers
1	1	Winter is Coming	2.2 M
1	2	The Kingsroad	2.2 M
2	1	The North Remembers	3.9 M

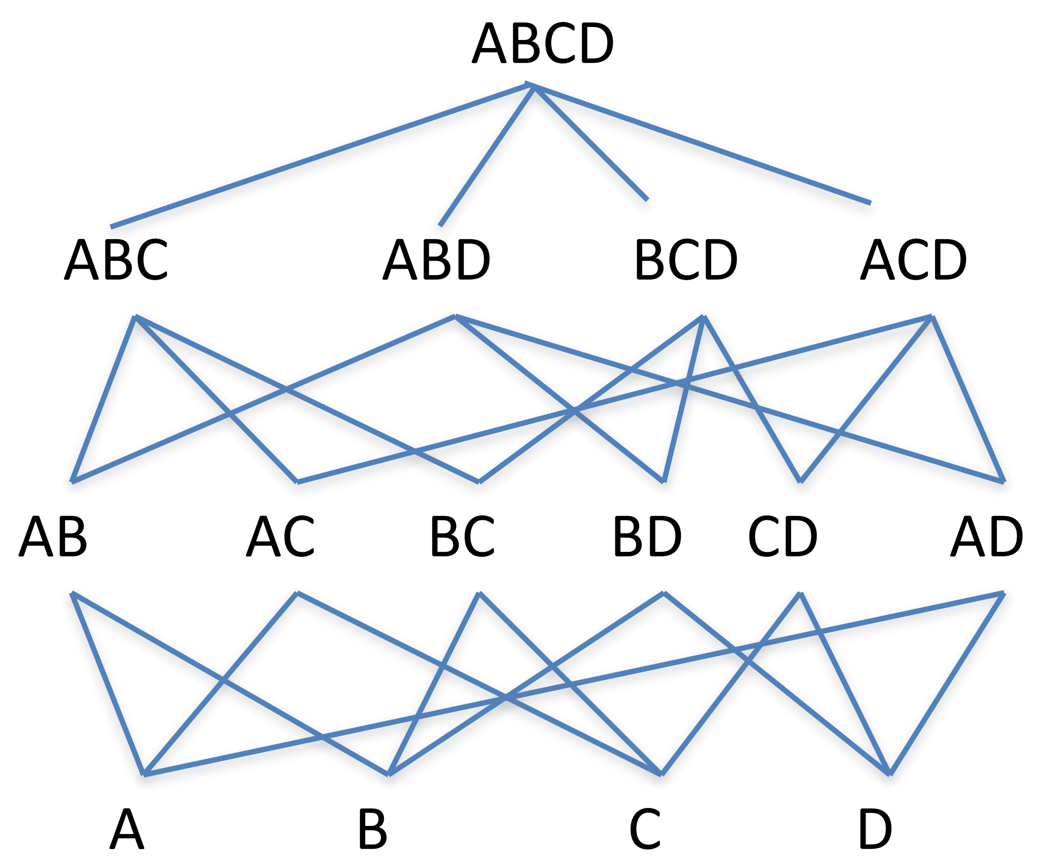
Projection is a relational algebra operation that takes as input relation ***R*** and returns a new relation ***R'*** with a subset of the columns of ***R***.

- Recall that more than one set of attributes **X** may be unique
- It may be the case that **X** and **Y** are both unique, and that they are not disjoint. When is this interesting?

Discovering uniques

R (A, B, C, D)

attribute lattice of ***R***



$$\binom{4}{4} = 1$$

$$\binom{4}{3} = 4$$

$$\binom{4}{2} = 6$$

$$\binom{4}{1} = 4$$

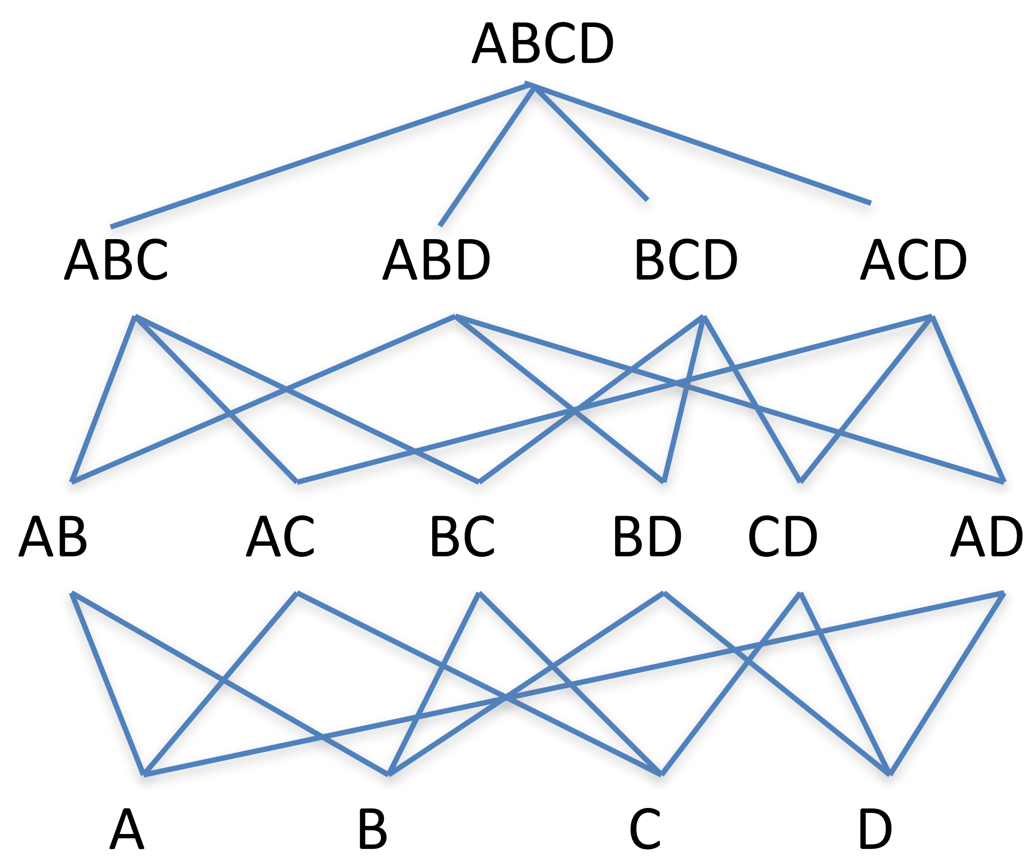
What's the size of the attribute lattice of ***R***?

Look at all attribute combinations?

Discovering uniques

R (A, B, C, D)

attribute lattice of R



- If **X** is unique, then what can we say about its **superset Y**?
- If **X** is non-unique, then what can we say about its **subset Z**?

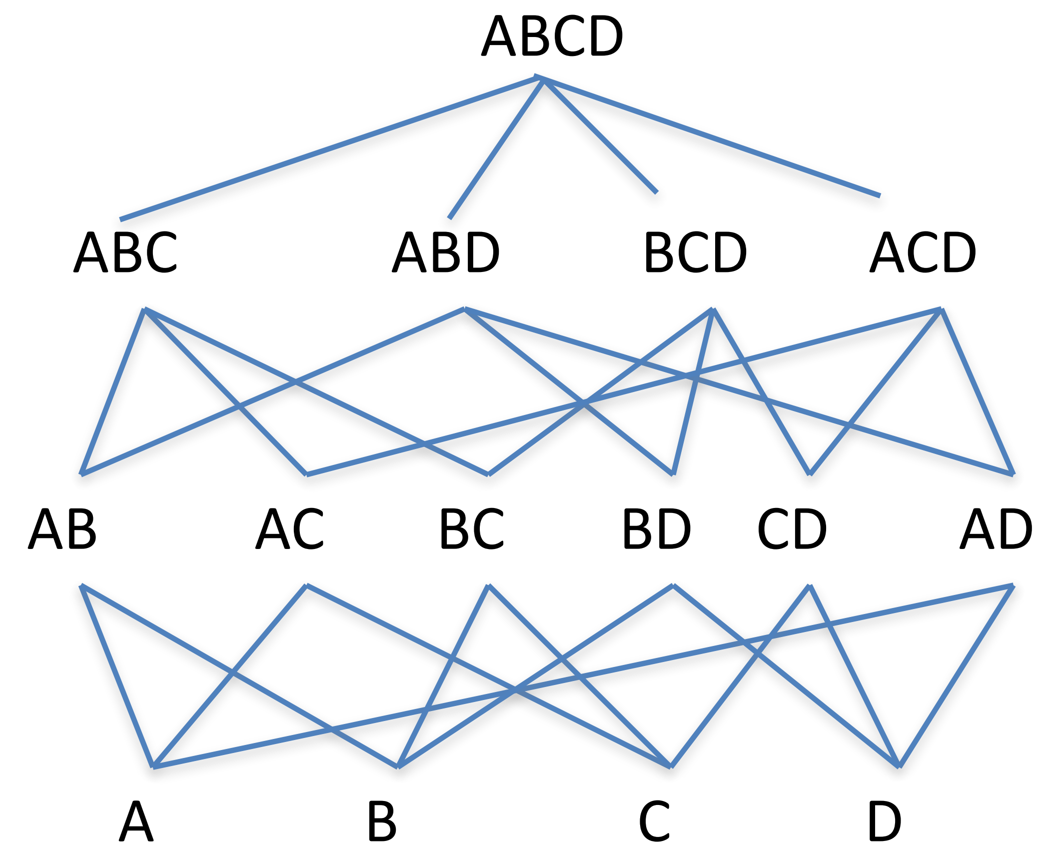
Discovering uniques

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a **unique column combination** (or a **"unique"** for short) is a set of attributes ***X*** whose **projection** contains no duplicates in ***r***

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a set of attributes ***Y*** is **non-unique** if its projection contains duplicates in ***r***

X is **minimal unique** if every subset ***Y*** of ***X*** is non-unique

Y is maximal non-unique if every superset ***X*** of ***Y*** is unique



From uniques to candidate keys

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a **unique column combination** is a set of attributes ***X*** whose **projection** contains no duplicates in ***r***

$$Episodes(season, num, title, viewers)$$

season	num	title	viewers
1	1	Winter is Coming	2.2 M
1	2	The Kingsroad	2.2 M
2	1	The North Remembers	3.9 M

A set of attributes is a **candidate key** for a relation if:

(1) no two distinct tuples can have the same values for all key attributes (candidate key **uniquely identifies** a tuple), *and*

(2) this is not true for any subset of the key attributes (candidate key **is minimal**)

A minimal unique of a relation instance is a (possible) candidate key of the relation schema. To find all possible candidate keys, find all minimal uniques in a relation instance.

association rule mining

r/ai center for responsible ai

The early days of data mining

- Problem formulation due to Agrawal, Imielinski, Swami, SIGMOD 1993
- Solution: the **Apriori** algorithm by Agrawal & Srikant, VLDB 1994
- Initially for **market-basket data** analysis, has many other applications, we'll see one today
- We wish to answer two related questions:
  - **Frequent itemsets:** Which items are often purchased together, e.g., milk and cookies are often bought together
  - **Association rules:** Which items will likely be purchased, based on other purchased items, e.g., if diapers are bought in a transaction, beer is also likely bought in the same transaction

Market-basket data

- $I = \{i_1, i_2, \ldots, i_m\}$ is the set of available items, e.g., a product catalog of a store
- $X \subseteq I$ is an **itemset**, e.g., {milk, bread, cereal}
- **Transaction** t is a set of items purchased together, $t \subseteq I$, has a transaction id (TID)

  t_1: {bread, cheese, milk}

  t_2: {apple, eggs, salt, yogurt}

  t_3: {biscuit, cheese, eggs, milk}
- Database T is a set of transactions $\{t_1, t_2, \ldots, t_n\}$
- A transaction t **supports** an itemset X if $X \subseteq t$
- Itemsets supported by at least ***minSupp*** transactions are called **frequent itemsets**

minSupp, which can be a number or a percentage, is specified by the user

Itemsets

TID	Items
1	A
2	A C
3	A B D
4	A C
5	A B C
6	A B C

minSupp = 2 transactions

How many possible itemsets are there (excluding the empty itemset)?

$2^4 - 1 = 15$

	itemset	support
★	A	6
★	B	3
★	C	4
	D	1
★	A B	3
★	A C	4
	A D	1
★	B C	2
	B D	1
	C D	0
★	A B C	2
	A B D	1
	B C D	0
	A C D	0
	A B C D	0

Association rules

An **association rule** is an implication $X \rightarrow Y$, where $X, Y \subset I$, *and* $X \cap Y = \varnothing$

example: {milk, bread} → {cereal}

"A customer who purchased X is also likely to have purchased Y in the same transaction"

we are interested in rules with a **single item** in Y

can we represent {milk, bread} → {cereal, cheese}?

Rule $X \rightarrow Y$ holds with **support** *supp* in T if *supp* of transactions contain $X \cup Y$

Rule $X \rightarrow Y$ holds with confidence *conf* in T if *conf* % of transactions that contain X also contain Y

$conf \approx \Pr(Y \mid X)$

$conf\ (X \rightarrow Y) = supp\ (X \cup Y)\ /\ supp\ (X)$

Association rules

minSupp = 2 transactions
minConf = 0.75

	supp = 3
$A \rightarrow B$	conf = 3 / 6 = 0.5
$B \rightarrow A$	conf = 3 / 3 = 1.0 ★
	supp = 2
$B \rightarrow C$	conf = 2 / 3 = 0.67
$C \rightarrow B$	conf = 2 / 4 = 0.5
	supp = 4
$A \rightarrow C$	conf = 4 / 6 = 0.67
$C \rightarrow A$	conf = 4 / 4 = 1.0 ★
	supp = 2
$AB \rightarrow C$	conf = 2 / 3 = 0.67
$AC \rightarrow B$	conf = 2 / 4 = 0.5
$BC \rightarrow A$	conf = 2 / 2 = 1.0 ★

	itemset	support
★	A	6
★	B	3
★	C	4
	D	1
★	A B	3
★	A C	4
	A D	1
★	B C	2
	B D	1
	C D	0
★	A B C	2
	A B D	1
	B C D	0
	A C D	0
	A B C D	0

$$conf\ (X \rightarrow Y) = supp\ (X \cup Y)\ /\ supp\ (X)$$

Association rule mining

- Goal: find all association rules that satisfy the user-specified minimum support and minimum confidence

- Algorithm outline
  - Step 1: find all frequent itemsets
  - Step 2: find association rules

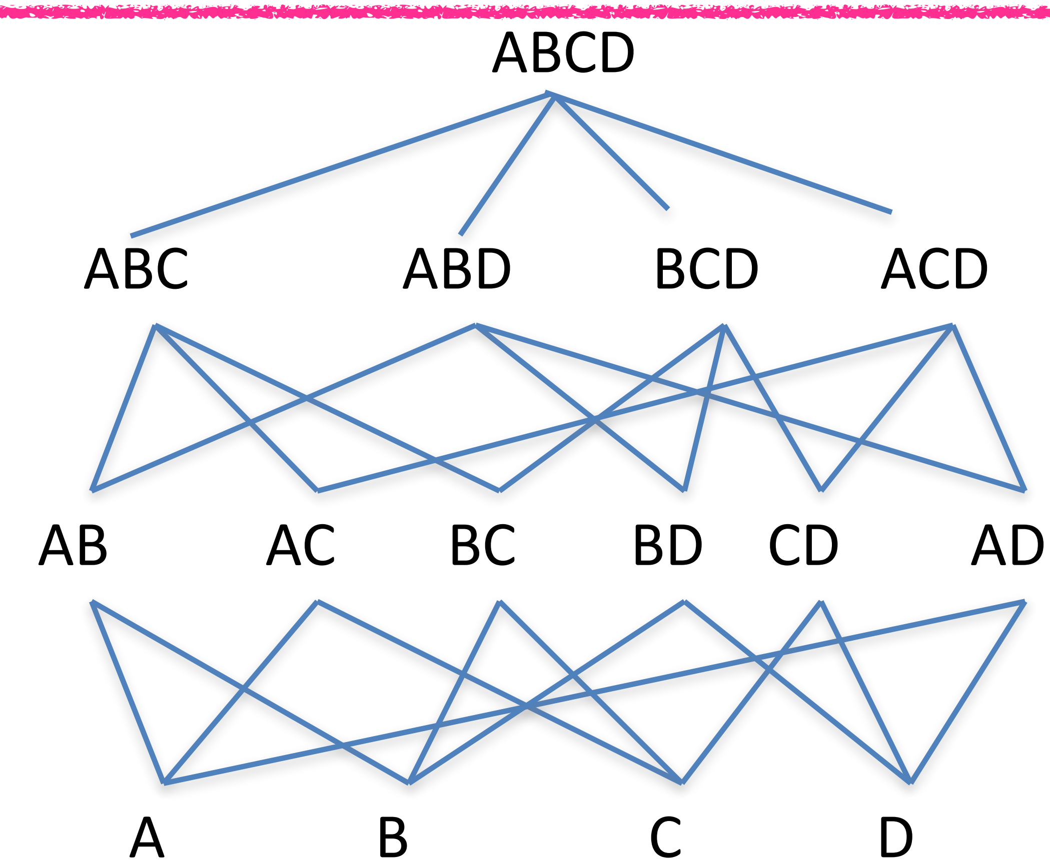
- Take 1: naïve algorithm for frequent itemset mining
  - Enumerate all subsets of ***I***, check their support in ***T***
  - **What is the complexity?**

Key idea: downward closure

itemset	support
★ A	6
★ B	3
★ C	4
D	1
★ A B	3
★ A C	4
A D	1
★ B C	2
B D	1
C D	0
★ A B C	2
A B D	1
B C D	0
A C D	0
A B C D	0

All subsets of a frequent itemset ***X*** are themselves frequent

So, if some subset of X is infrequent, then X cannot be frequent, we know this **apriori**



The converse is not true! If all subsets of ***X*** are frequent, ***X*** is not guaranteed to be frequent

The *Apriori* algorithm

Algorithm Apriori(*T*, *minSupp*)

F_1 = {*frequent 1-itemsets*};

for ($k = 2$; $F_{k-1} \neq \emptyset$; k++) **do**

$C_k \leftarrow$ **candidate-gen**(F_{k-1});

for each transaction $t \in T$ **do**

for each candidate $c \in C_k$ **do**

if c is contained in t **then**

c.count++;

end

end

$F_k \leftarrow \{c \in C_k \mid c.count \geq minSupp\}$

end

return $F \leftarrow \bigcup_k F_k$;

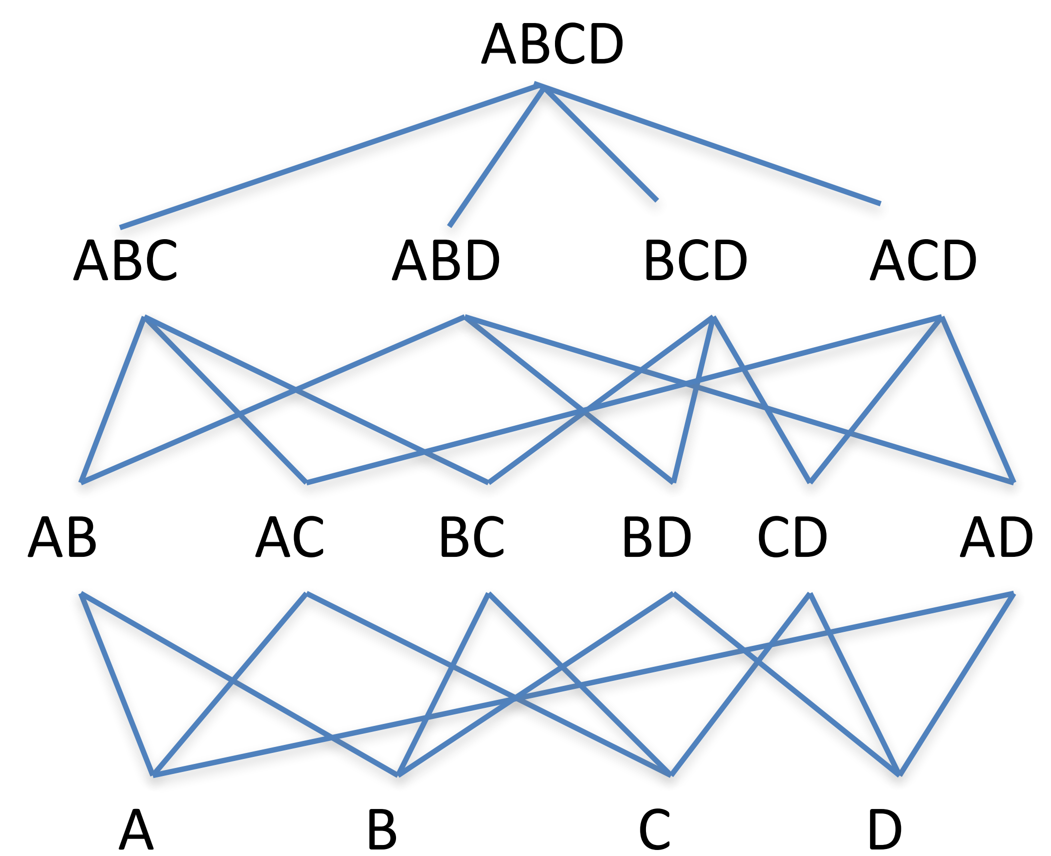
itemset	support
A	6
B	3
C	4
D	1
A B	3
A C	4
A D	1
B C	2
B D	1
C D	0
A B C	2
A B D	1
B C D	0
A C D	0
A B C D	0

Candidate generation

The **candidate-gen** function takes F_{k-1} and returns a superset (called the candidates) of the set of all frequent k-itemsets. It has two steps:

Join: generate all possible candidate itemsets C_k of length k

Prune: optionally remove those candidates in C_k that have infrequent subsets



Candidate generation: join

```
Insert into C_k (
 select   p.item_1, p.item_2, …, p.item_{k-1}, q.item_{k-1}
 from     F_{k-1} p, F_{k-1} q
 where    p.item_1 = q.item_1
   and       p.item_2 = q.item_2
  and     …
  and     p.item_{k-1} < q.item_{k-1})
```

F_1 as p

A
B
C

F_1 as q

A
B
C

C_2

A	**B**
A	**C**
B	**C**

	itemset	support
★	A	6
★	B	3
★	C	4
	D	1
★	A B	3
★	A C	4
	A D	1
★	B C	2
	B D	1
	C D	0
★	A B C	2
	A B D	1
	B C D	0
	A C D	0
	A B C D	0

Candidate generation: join

```
Insert into Ck (
 select   p.item1, p.item2, …, p.itemk-1, q.itemk-1
 from     Fk-1 p, Fk-1 q
 where    p.item1 = q.item1
   and       p.item2 = q.item2
  and     …
  and     p.itemk-1 < q.itemk-1)
```

itemset	support
★ A	6
★ B	3
★ C	4
D	1
★ A B	3
★ A C	4
A D	1
★ B C	2
B D	1
C D	0
★ A B C	2
A B D	1
B C D	0
A C D	0
A B C D	0

F_2 as p

A	**B**
A	C
B	C

F_2 as q

A	B
A	**C**
B	C

C_3

A	**B**	**C**

Candidate generation

Assume a lexicographic ordering of the items

Join

```
Insert into C_k (
  select   p.item_1, p.item_2, …, p.item_{k-1}, q.item_{k-1}
  from     F_{k-1} p, F_{k-1} q
  where    p.item_1 = q.item_1
  and      p.item_2 = q.item_2
  and      …
  and      p.item_{k-1} < q.item_{k-1} )
```

why not $p.item_{k-1} \neq q.item_{k-1}$?

Prune

for each c in C_k **do**
  for each (k-1) subset s of c **do**
    if (s not in F_{k-1}) **then**
      delete c from C_k

Generating association rules

Rules = ∅

for each frequent *k-itemset* X **do**

  for each 1-itemset A ⊂ X **do**

   compute conf (X / A → A) = supp(X) / sup (X / A)

   if conf (X / A → A) $\geq$ minConf **then**

    Rules ← “X / A → A"

  end

 end

end

return Rules

Performance of *Apriori*

- The possible number of frequent itemsets is exponential, O($\boldsymbol{2^m}$), where $\boldsymbol{m}$ is the number of items
- Apriori exploits sparseness and locality of data
  - Still, it may produce a large number of rules: thousands, tens of thousands, ….
  - So, thresholds should be set carefully. What are some good heuristics?

back to data profiling

center for responsible ai

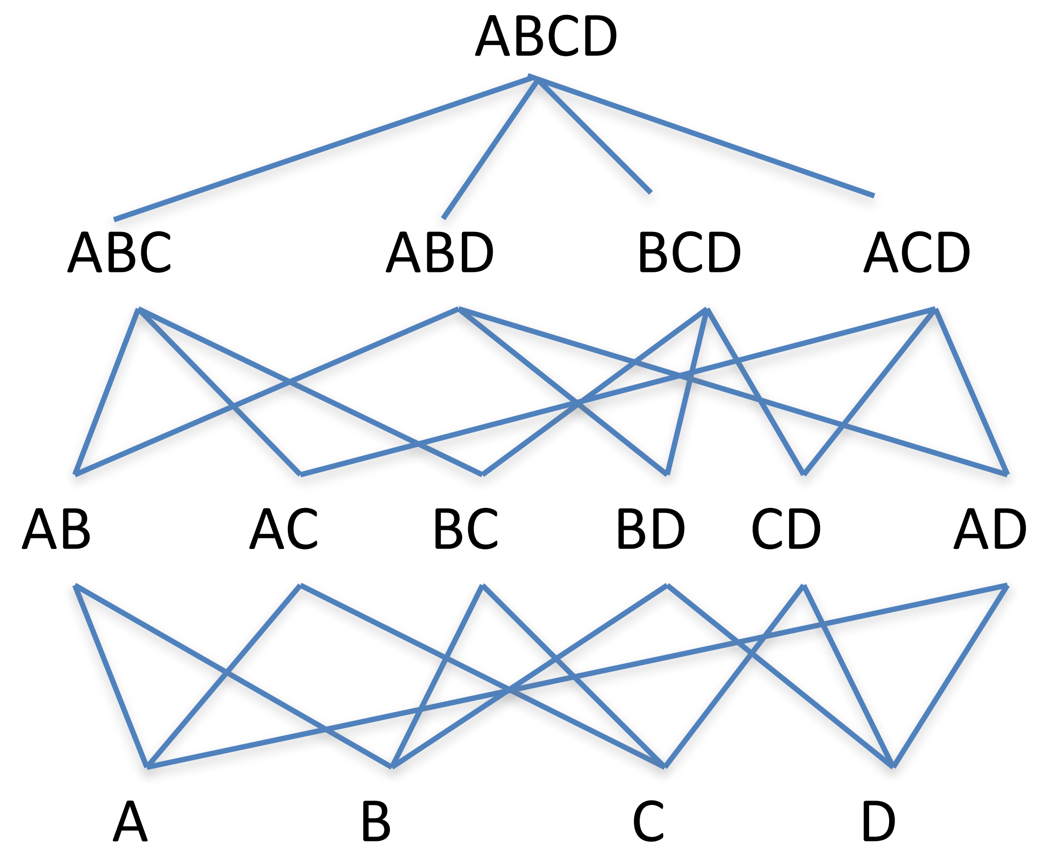
Discovering uniques

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a **unique column combination** (or a **"unique"** for short) is a set of attributes ***X*** whose **projection** contains no duplicates in ***r***

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a set of attributes ***Y*** is **non-unique** if its projection contains duplicates in ***r***

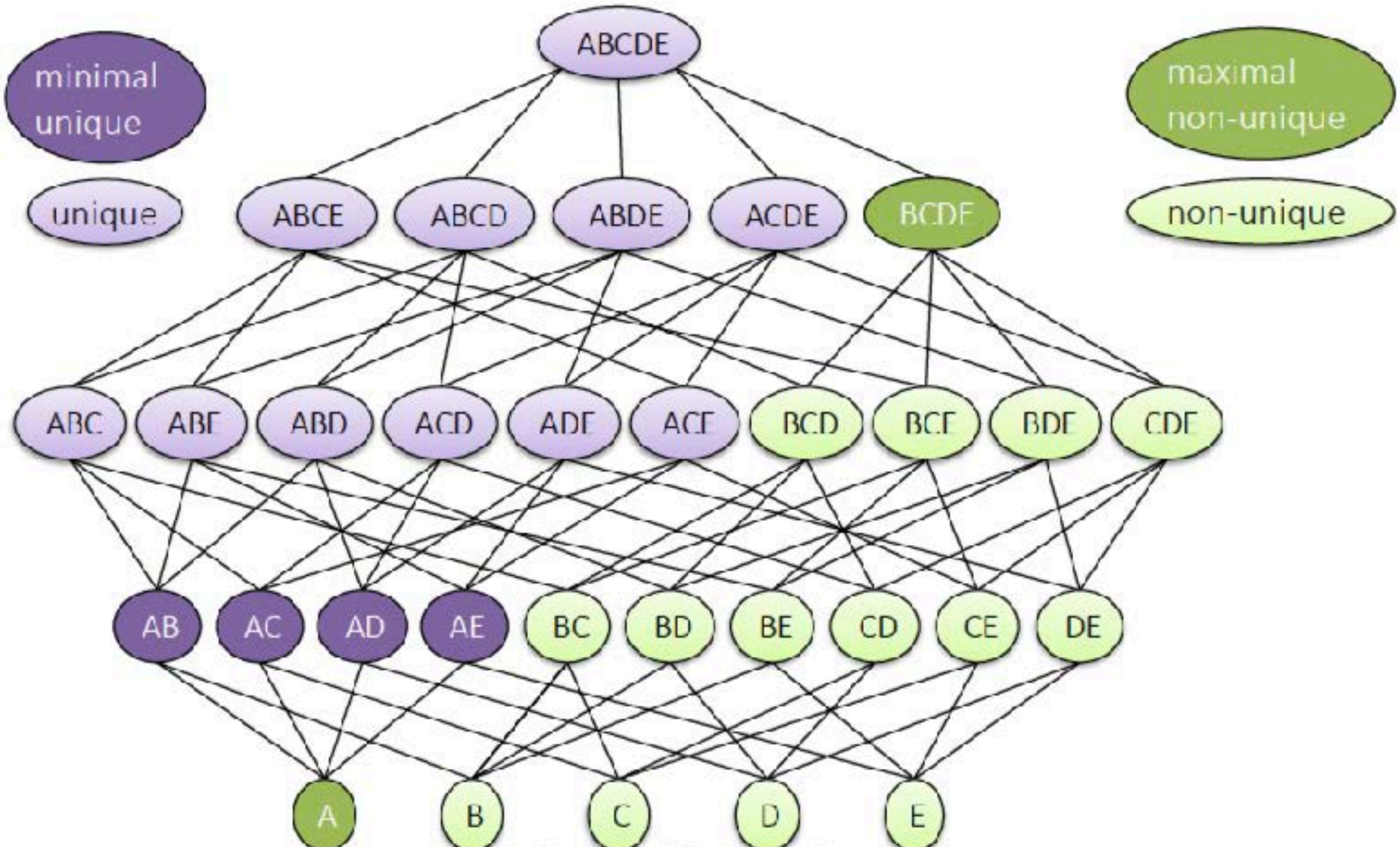
X is **minimal unique** if every subset ***Y*** of ***X*** is non-unique

Y is maximal non-unique if every superset ***X*** of ***Y*** is unique



[Abedjan, Golab, Naumann; *SIGMOD 2017*]

Output



From uniques to candidate keys

Given a relation schema ***R*** *(A, B, C, D)* and a relation instance ***r***, a **unique column combination** is a set of attributes ***X*** whose **projection** contains no duplicates in ***r***

$$Episodes(season, num, title, viewers)$$

season	num	title	viewers
1	1	Winter is Coming	2.2 M
1	2	The Kingsroad	2.2 M
2	1	The North Remembers	3.9 M

A set of attributes is a **candidate key** for a relation if:

(1) no two distinct tuples can have the value values for all key attributes (candidate key **uniquely identifies** a tuple), *and*

(2) this is not true for any subset of the key attributes (candidate key **is minimal**)

A minimal unique of a relation instance is a (possible) candidate key of the relation schema. To find such possible candidate keys, find all minimal uniques in a given relation instance.

center for responsible ai

Apriori-style uniques discovery

[Abedjan, Golab, Naumann; *SIGMOD 2017*]

A minimal unique of a relation instance is a **(possible) candidate key** of the relation schema.

Algorithm Uniques **// sketch, similar to HCA**

U_1 = {1-uniques} N_1 = {1-non-uniques}

for (k = 2; $N_{k-1} \neq \varnothing$; k++) **do**

$C_k \leftarrow$ **candidate-gen**(N_{k-1})

$U_k \leftarrow$ **prune-then-check** (C_k)

// prune candidates with unique sub-sets, and with **value distributions that cannot be unique**

// check each candidate in pruned set for uniqueness

$N_k \leftarrow C_k \setminus U_k$

end

return $U \leftarrow \bigcup_k U_k$;

breadth-first bottom-up strategy for attribute lattice traversal

taming technical bias

center for responsible ai

This week's reading

Taming Technical Bias in Machine Learning Pipelines *

Sebastian Schelter
University of Amsterdam & Ahold Delhaize
Amsterdam, The Netherlands
s.schelter@uva.nl

Julia Stoyanovich
New York University
New York, NY, USA
stoyanovich@nyu.edu

Abstract

Machine Learning (ML) is commonly used to automate decisions in domains as varied as credit and lending, medical diagnosis, and hiring. These decisions are consequential, imploring us to carefully balance the benefits of efficiency with the potential risks. Much of the conversation about the risks centers around bias — a term that is used by the technical community ever more frequently but that is still poorly understood. In this paper we focus on technical bias — a type of bias that has so far received limited attention and that the data engineering community is well-equipped to address. We discuss dimensions of technical bias that can arise through the ML lifecycle, particularly when it's due to preprocessing decisions or post-deployment issues. We present results of our recent work, and discuss future research directions. Our over-all goal is to support the development of systems that expose the knobs of responsibility to data scientists, allowing them to detect instances of technical bias and to mitigate it when possible.

1 Introduction

Machine Learning (ML) is increasingly used to automate decisions that impact people's lives, in domains as varied as credit and lending, medical diagnosis, and hiring. The risks and opportunities arising from the wide-spread use of predictive analytics are garnering much attention from policy makers, scientists, and the media. Much of this conversation centers around *bias* — a term that is used by the technical community ever more frequently but that is still poorly understood.

In their seminal 1996 paper, Friedman and Nissenbaum identified three types of bias that can arise in computer systems: pre-existing, technical, and emergent [9]. We briefly discuss these in turn, see Stoyanovich et al. [33] for a more comprehensive overview.

- *Pre-existing bias* has its origins in society. In ML applications, this type of bias often exhibits itself in the input data; detecting and mitigating it is the subject of much research under the heading of algorithmic fairness [5]. Importantly, the presence or absence of pre-existing bias cannot be scientifically verified, but rather is postulated based on a belief system [8, 12]. Consequently, the effectiveness — or even the validity — of a technical attempt to mitigate pre-existing bias is predicated on that belief system.

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

*This work was supported in part by NSF Grants No. 1926250, 1934464, and 1922658, and by Ahold Delhaize. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

The VLDB Journal
https://doi.org/10.1007/s00778-021-00726-w

SPECIAL ISSUE PAPER

Data distribution debugging in machine learning pipelines

Stefan Grafberger[1] · Paul Groth[1] · Julia Stoyanovich[2] · Sebastian Schelter[1]

Received: 27 February 2021 / Revised: 9 September 2021 / Accepted: 3 December 2021

Abstract
Machine learning (ML) is increasingly used to automate impactful decisions, and the risks arising from this widespread use are garnering attention from policy makers, scientists, and the media. ML applications are often brittle with respect to their input data, which leads to concerns about their correctness, reliability, and fairness. In this paper, we describe mlinspect, a library that helps diagnose and mitigate technical bias that may arise during preprocessing steps in an ML pipeline. We refer to these problems collectively as *data distribution bugs*. The key idea is to extract a directed acyclic graph representation of the dataflow from a preprocessing pipeline and to use this representation to automatically instrument the code with predefined *inspections*. These inspections are based on a lightweight annotation propagation approach to propagate metadata such as lineage information from operator to operator. In contrast to existing work, mlinspect operates on declarative abstractions of popular data science libraries like estimator/transformer pipelines and does not require manual code instrumentation. We discuss the design and implementation of the mlinspect library and give a comprehensive end-to-end example that illustrates its functionality.

Keywords Data debugging · Machine learning pipelines · Data preparation for machine learning

1 Introduction

Machine learning (ML) is increasingly used to automate decisions that impact people's lives, in domains as varied as credit and lending, medical diagnosis, and hiring, with the potential to reduce costs, reduce errors, and make outcomes more equitable. Yet, despite their potential, the risks arising from the widespread use of ML-based tools are garnering attention from policy makers, scientists, and the media [52]. In large part this is because the correctness, reliability, and fairness of ML models critically depend on their training data. Preexisting bias, such as under- or over-representation of particular groups in the training data [12], and technical bias, such as skew introduced during data preparation [45], can heavily impact performance. In this work, we focus on helping diagnose and mitigate technical bias that arises during preprocessing steps in an ML pipeline. We refer to these problems collectively as *data distribution bugs*.

Data distribution bugs are often introduced during preprocessing. Input data for ML applications come from a variety of data sources, and it has to be preprocessed and encoded as features before it can be used. This preprocessing can introduce skew in the data, and, in particular, it can exacerbate under-representation of historically disadvantaged groups. For example, preprocessing operations that involve filters or joins can heavily change the distribution of different groups represented in the training data [58], and missing value imputation can also introduce skew [47]. Recent ML fairness research, which mostly focuses on the use of learning algorithms on static datasets [14], is therefore insufficient because it cannot address such technical bias originating from the data preparation stage. Furthermore, it is important to detect and mitigate bias as close to its source as possible [52].

Data distribution bugs are difficult to catch. In part, this is because different pipeline steps are implemented using different libraries and abstractions, and data representation often

Sebastian Schelter
s.schelter@uva.nl

Stefan Grafberger
s.grafberger@uva.nl

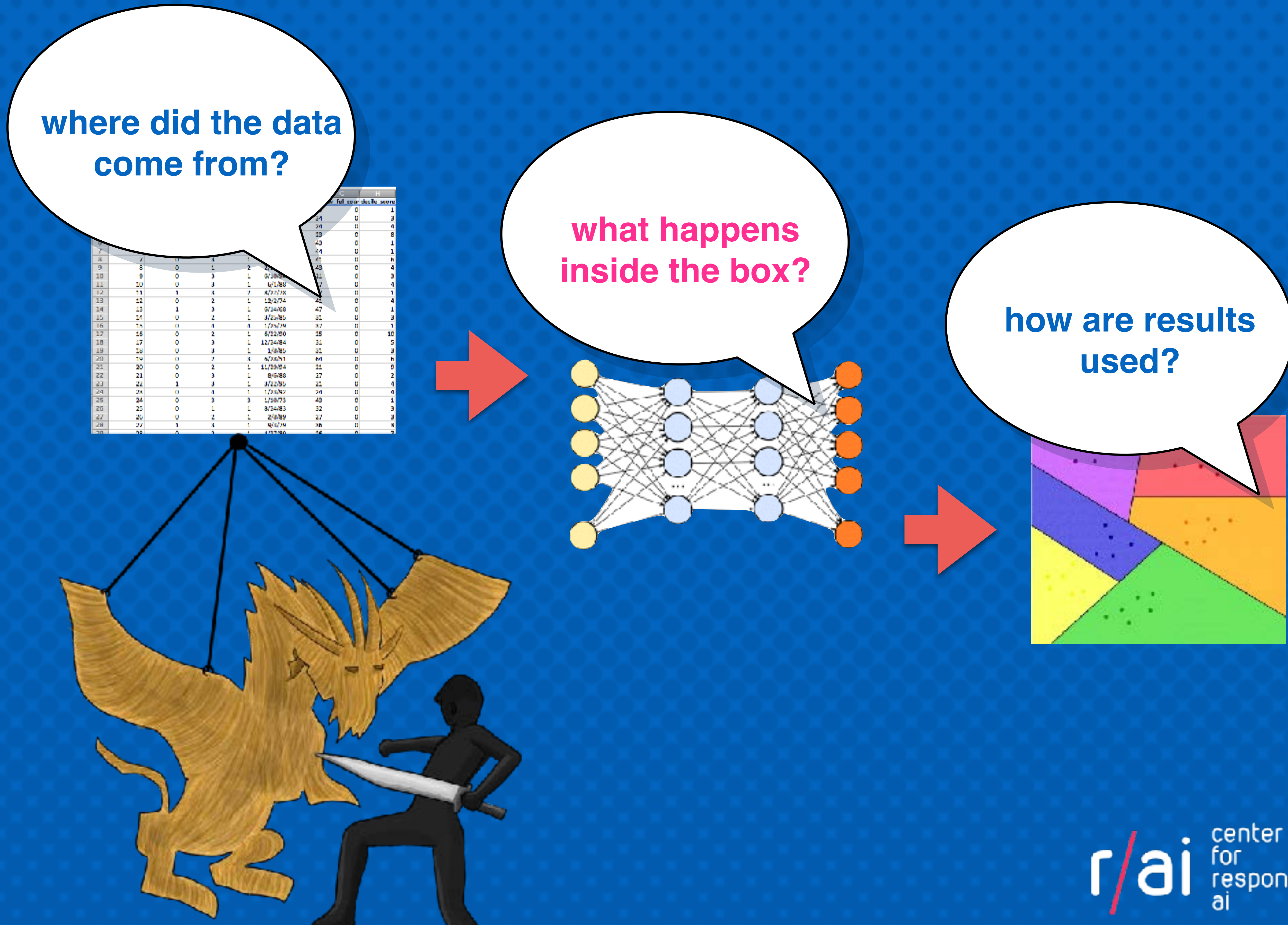
Paul Groth
p.t.groth@uva.nl

Julia Stoyanovich
stoyanovich@nyu.edu

[1] University of Amsterdam, Amsterdam, Netherlands
[2] New York University, New York, USA

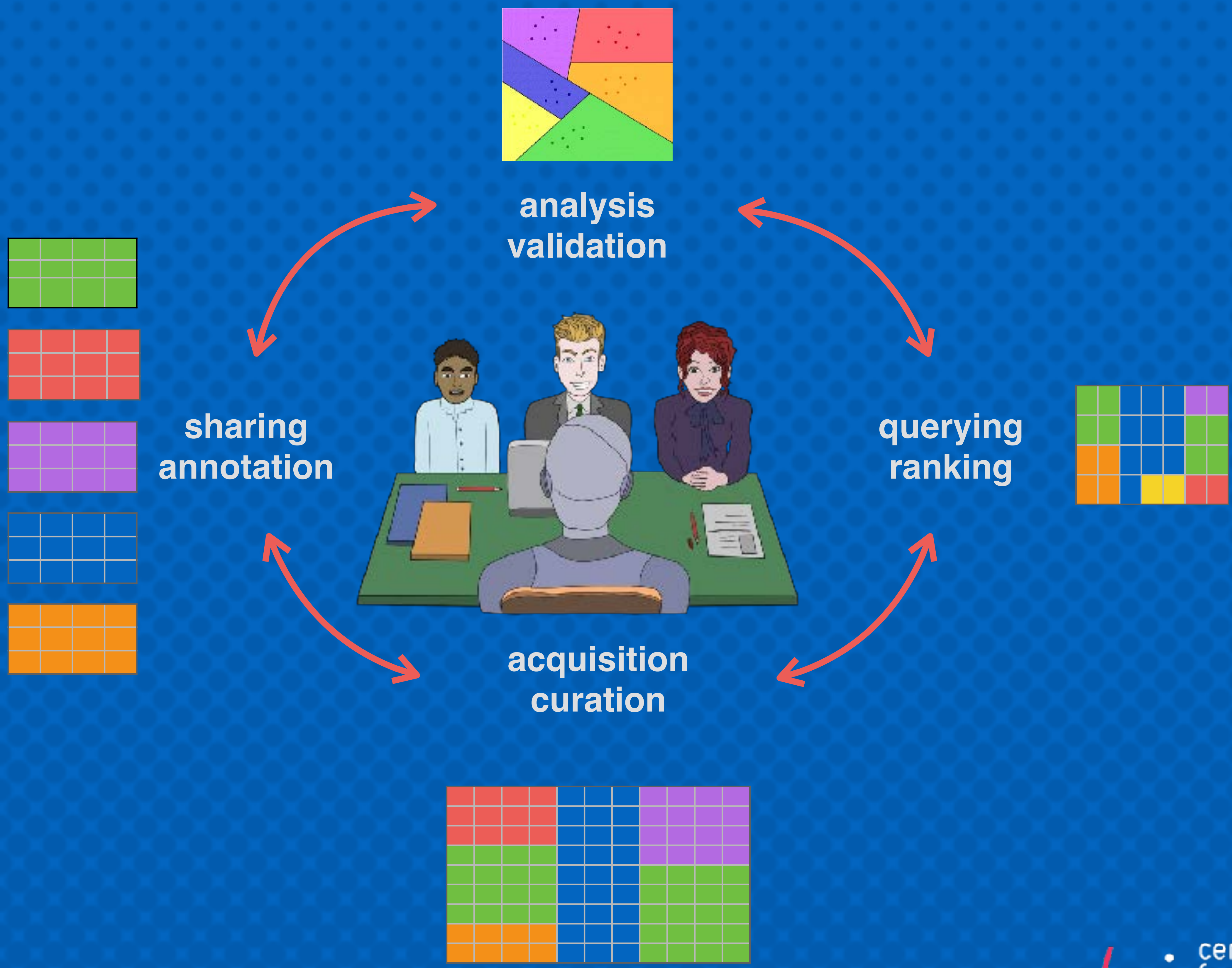
Published online: 21 January 2022

The “last-mile” view of responsible AI



center for responsible ai

Zooming out to the lifecycle view



r/ai center for responsible ai

Bias in computer systems

Pre-existing is independent of an algorithm and has origins in society

Technical is introduced or exacerbated by the technical properties of an ADS

Emergent arises due to context of use

[Friedman & Nissenbaum (1996)]

r/ai center for responsible ai

Model development lifecycle

Goal

design a model to predict an appropriate level of compensation for job applicants

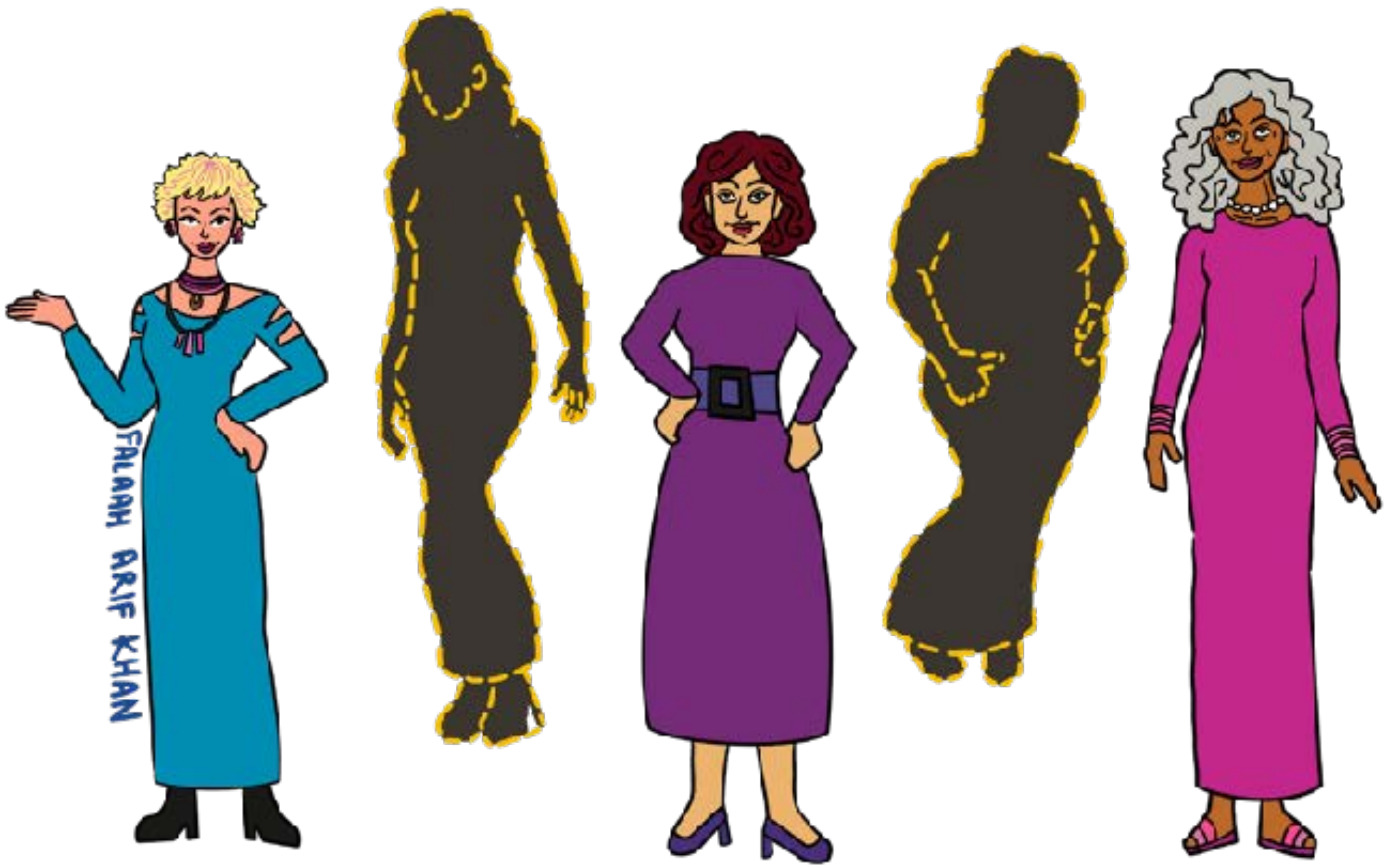
Problem

women are offered a lower salary than they would expect, potentially reinforcing the gender wage gap

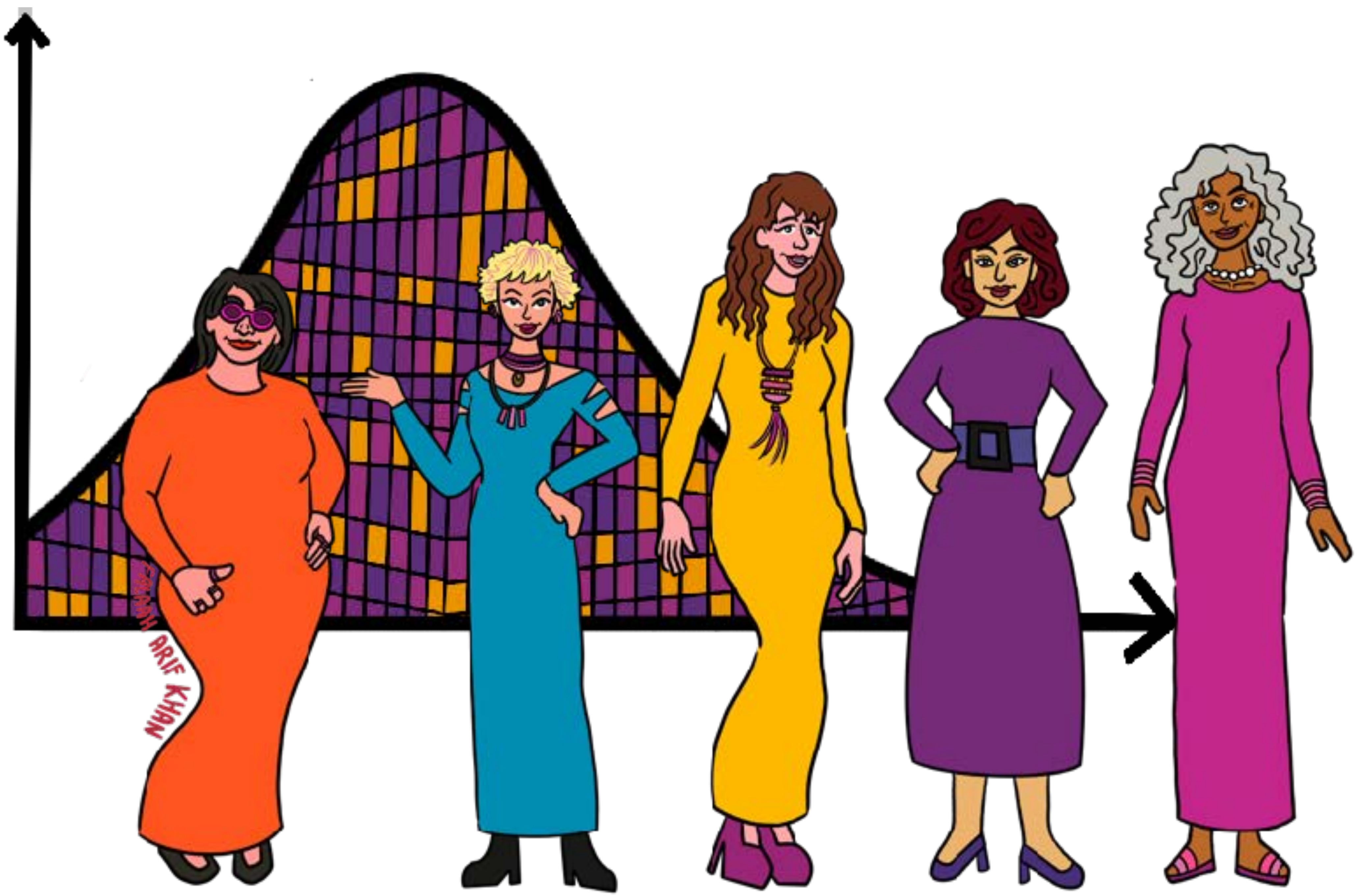
[Schelter, He, Khilnani, Stoyanovich (2020)]

r/ai center for responsible ai

Missing values: Observed data



Missing values: Imputed distribution



Missing values: True distribution

dimensions of technical bias

r/ai center for responsible ai

Recall: 50 shades of *null*

- **Unknown** - some value definitely belongs here, but I don't know what it is (e.g., unknown birthdate)
- **Inapplicable** - no value makes sense here (e.g., if marital status = single then spouse name should not have a value)
- **Unintentionally omitted** - values is left unspecified unintentionally, by mistake
- **Optional** - a value may legitimately be left unspecified (e.g., middle name)
- **Intentionally withheld** (e.g., an unlisted phone number)
- …..

should we be filling these in? if so, how?

https://www.vertabelo.com/blog/technical-articles/50-shades-of-null-or-how-a-billion-dollar-mistake-has-been-stalking-a-whole-industry-for-decades

Missing value imputation

are values **missing at random** (e.g., gender, age, disability on job applications)?

are we ever interpolating **rare categories** (e.g., Native American)

are **all categories** represented (e.g., non-binary gender)?

Data filtering

"filtering" operations (like selection and join), **can arbitrarily change demographic group proportions**

select by zip code, country, years of C++ experience, others?

age_group	county
60	CountyA
60	CountyA
20	CountyA
60	CountyB
20	CountyB
20	CountyB

50% vs 50%

→

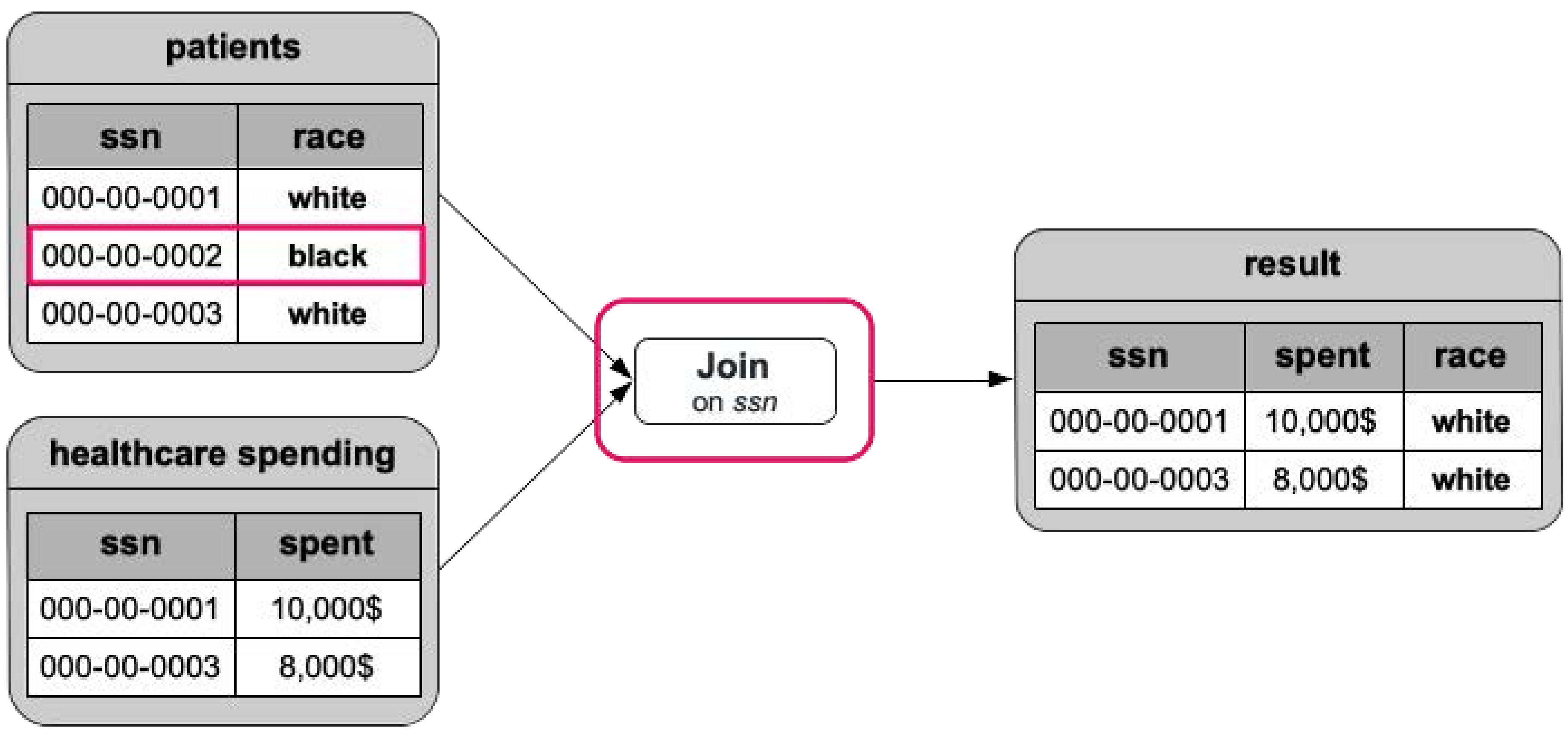
age_group	county
60	CountyA
60	CountyA
20	CountyA

66% vs 33%

Data filtering

“filtering” operations (like selection and join), **can arbitrarily change demographic group proportions**

select by zip code, country, years of C++ experience, others?



Data distribution debugging: mlinspect

Potential issues in preprocessing pipeline:

1. Join might change proportions of groups in data
2. Column ‘age_group’ projected out, but required for fairness
3. Selection might change proportions of groups in data
4. Imputation might change proportions of groups in data
5. ‘race’ as a feature might be illegal!
6. Embedding vectors may not be available for rare names!

Python script for preprocessing, written exclusively with native pandas and sklearn constructs

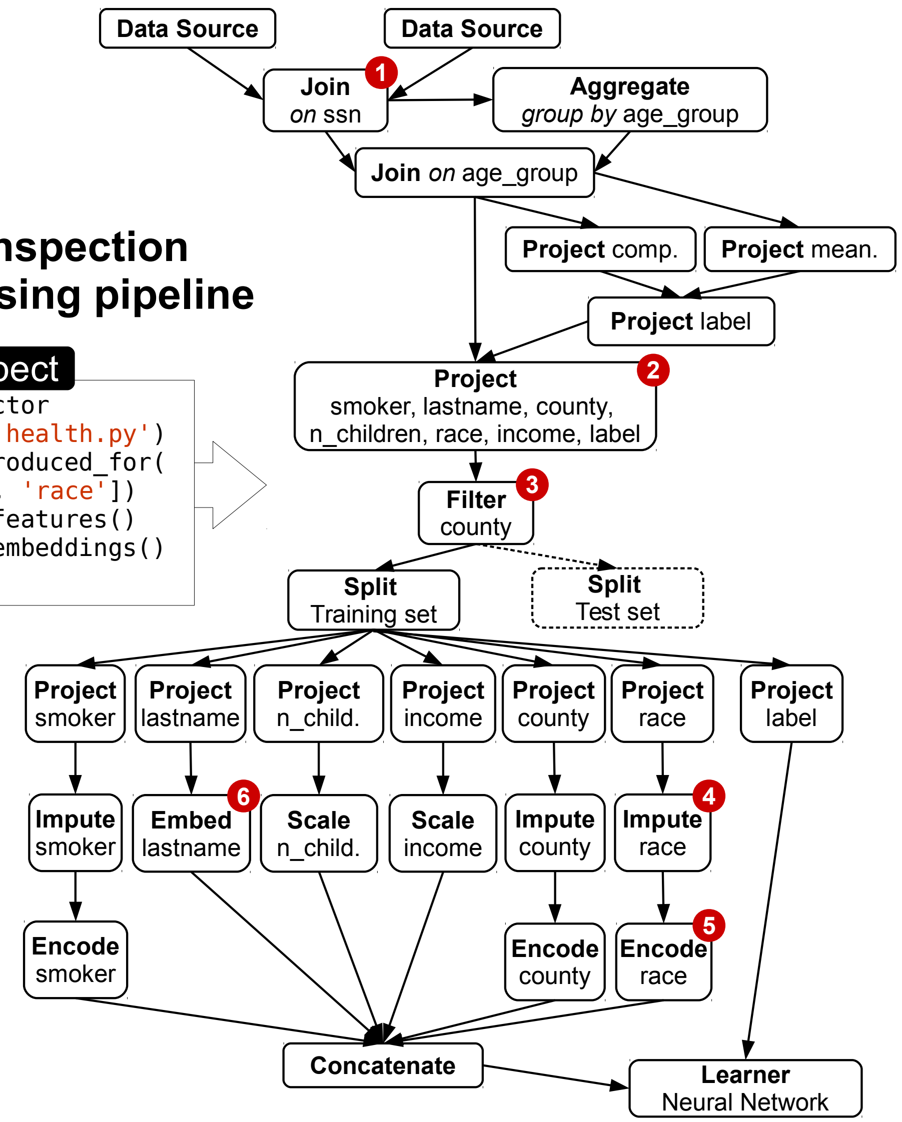
```
# load input data sources, join to single table
patients = pandas.read_csv(…)
histories = pandas.read_csv(…)
data = pandas.merge([patients, histories], on=['ssn'])

# compute mean complications per age group, append as column
complications = data.groupby('age_group')
  .agg(mean_complications=('complications','mean'))
data = data.merge(complications, on=['age_group'])

# Target variable: people with frequent complications
data['label'] = data['complications'] >
  1.2 * data['mean_complications']

# Project data to subset of attributes, filter by counties
data = data[['smoker', 'last_name', 'county',
             'num_children', 'race', 'income', 'label']]
data = data[data['county'].isin(counties_of_interest)]

# Define a nested feature encoding pipeline for the data
impute_and_encode = sklearn.Pipeline([
  (sklearn.SimpleImputer(strategy='most_frequent')),
  (sklearn.OneHotEncoder())])
featurisation = sklearn.ColumnTransformer(transformers=[
  (impute_and_encode, ['smoker', 'county', 'race']),
  (Word2VecTransformer(), 'last_name')
  (sklearn.StandardScaler(), ['num_children', 'income']])

# Define the training pipeline for the model
neural_net = sklearn.KerasClassifier(build_fn=create_model())
pipeline = sklearn.Pipeline([
  ('features', featurisation),
  ('learning_algorithm', neural_net)])

# Train-test split, model training and evaluation
train_data, test_data = train_test_split(data)
model = pipeline.fit(train_data, train_data.label)
print(model.score(test_data, test_data.label))
```

Declarative inspection of preprocessing pipeline

mlinspect

```
PipelineInspector
.on_pipeline('health.py')
 .no_bias_introduced_for(
 ['age_group', 'race'])
 .no_illegal_features()
 .no_missing_embeddings()
.verify()
```

Corresponding dataflow DAG for instrumentation, extracted by *mlinspect*



[Grafberger, Stoyanovich, Schelter (2022)]

r/ai center for responsible ai

Data debugging: mlinspect

- similar to code inspection in modern IDEs, but specifically for data
- works on existing pipeline code using libraries like pandas and scikit-learn
- negligible performance overhead

ACM SIGMOD 2021 demo (4 min)

https://surfdrive.surf.nl/files/index.php/s/ybriyzsdc6vcd2w

CIDR 2021 talk (10 min)

https://www.youtube.com/watch?v=Ic0aD6lv5h0

Sound experimentation

> "A theory or idea shouldn't be scientific unless it could, in principle, be proven false."
>
> *Karl Popper*

- software-engineering and data science best-practices
- data isolation: training / validation / test
- accounting for **variability** when observing trends
- tuning hyper-parameters: **for what objective**?

Responsible Data Science

The data science lifecycle

Thank you!

NYU TANDON SCHOOL OF ENGINEERING

NYU Center for Data Science

r/ai center for responsible ai