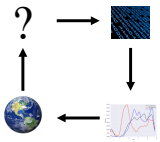
Data Science 100
Lecture 4: Data Wrangling

Slides by:
Joe Hellerstein
hellerstein@berkeley.edu



So you want to be a data scientist…

How will you spend your time?

Enterprise Data Analysis and Visualization: An Interview Study

Interview study of 35 analysts

25 companies
Healthcare
Retail, Marketing
Social networking
Media
Finance, Insurance

Various titles
Data analyst
Data scientist
Software engineer
Consultant
Chief technical officer

Kandel et al. "Enterprise Data Analysis and Visualization: An Interview Study. IEEE Visual Analytics Science & Technology (VAST), 2012
http://db.cs.berkeley.edu/papers/vast12-interview.pdf

"I spend more than half of my time integrating, cleansing and transforming data without doing any actual analysis. Most of the time I'm lucky if I get to do any 'analysis' at all…

… Most of the time once you transform the data … the insights can be scarily obvious."

"Once you play with the data you realize you made an assumption that is completely wrong. It's really useful, it's not just a waste of time, even though you may be banging your head."

"In practice it tends not to be just data prep, you are learning about the data at the same time, you are learning about what assumptions you can make."

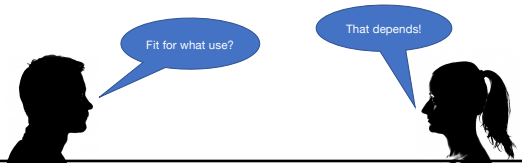
Big Data Borat
@BigDataBorat

Following

In Data Science, 80% of time spent prepare data, 20% of time spent complain about need for prepare data.

Data Wrangling

Aka Data Prep, Data Munging, Data Transformation
*Assessing and transforming raw data to make it **fit for use***



Data Wrangling

Aka Data Prep, Data Munging, Data Transformation
*Assessing and transforming raw data to make it **fit for use***

This is how you "get your head in the game"
- Understand what you have
- Assess strengths and weaknesses of your data
- Hypothesize about what to do with your data
- Get it ready

Nobody will know your data as well as you do while wrangling
- Not even the "you" of a few days later

Discussion

When is data "dirty"?

How does that happen?

Today: Data Unboxing and Wrangling

Basic tools:
- UNIX command line
- SublimeText editor

Trifacta: a free visual data wrangling tool
- Born at Berkeley/Stanford
- Codifies some good practices you can also follow "by hand"

Later: Python's Pandas library

You may choose to use other tools too
- Good data scientists maintain a well-stocked toolbox

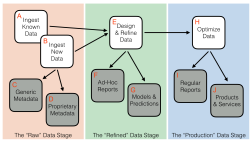
A bit of background before we jump in

Stages of Wrangling

Raw: Data ingestion & discovery ("unboxing")
- What: Exploratory *ad hoc* analysis
- Who: The individual wrangler

Refined: Curating data for reuse
- What: Data warehousing, canonical models
- Who: Data curators, IT engineers, actuaries, etc.

Production: Ensuring feeds and workflows
- What: Recurrent, automated use cases:
  - Traditional (e.g. reporting) + New (e.g. recommenders)
- Who: Often involves SW engineers and IT/ops folks



Rattenbury, et al. "Data Wrangling: Techniques and Concepts for Agile Analytics". To appear, O'Reilly Media, 2017.

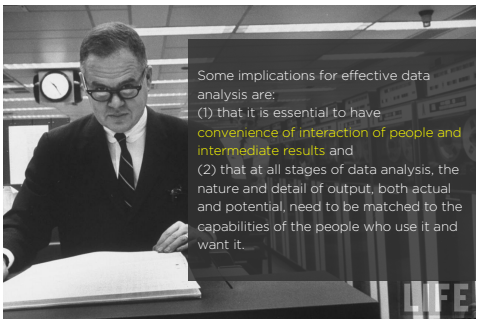
Today

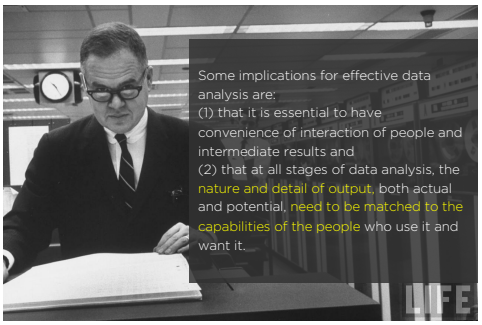
We will focus on the "Raw→Refined" stage
- Unboxing
- Transformation to analytics-ready structure
- Assessment/mitigation of quality issues

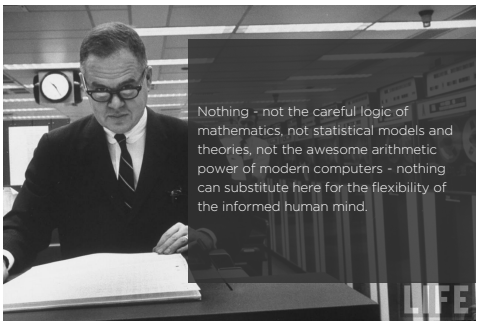
This is just a taste
- Soon you will learn to do this in Python & Pandas

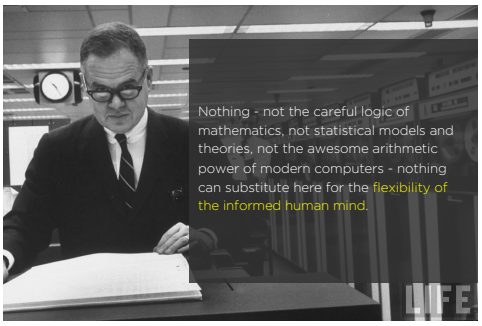
More on this in future lectures!


Data Analysis & Statistics, Tukey 1965









Unboxing Data

- What do I have here?
- What do I want to do with it?

These questions rarely have pat answers.
- Typically *contextual* and *user-driven*
- Typically subject to *iterative* cycles of wrangling and analysis

Rough Guide to Wrangling Issues
an outline for today's lecture

- Structure: the "shape" of a data file
- Granularity: how fine/coarse is each datum
- Faithfulness: how well does the data capture "reality"
- Temporality: how is the data situated in time
- Scope: how (in)complete is the data

Many of these are ***subjective*** qualities! Depend on *context*.

(Data) Science is a human process.

Structure

Structure: Rectangular Data

Natural for data analysis: easy to access, filter, tabulate

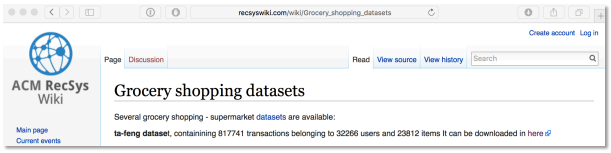
Two main variants

1. Relations (a.k.a. tables, data-frames)
   - Manipulate with Relational Algebra
2. Matrices
   - Manipulate with Linear Algebra

There are non-rectangular structures as well, e.g. JSON, XML.
For analysis, you will typically need to convert these to rectangular structures.

Structure | Granularity | Faithfulness | Temporality | Scope

Without further ado, some data

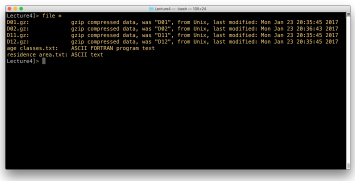


Gently modified for pedagogical purposes here.

Original dataset here.

Unboxing with UNIX Command Line

- File metadata
  - `ls -lh`
  - `file`
  - `wc`
- File (de)compression
  - `gunzip`, `zip`, `bzip`, etc.
- `stdout` and the pipe
- File content:
  - `cat`
  - `head`
  - `tail`
  - `less`
  - <ctrl>-C

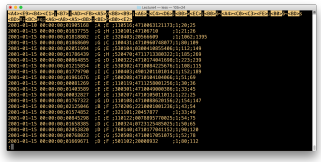


Structure | Granularity | Faithfulness | Temporality | Scope

Structure Questions: A Checklist

- Coarse structure
  - Is the data structured as a collection of *records*?
  - How are the individual records delimited in the dataset?
  - How are the record *fields* delimited from one another?
  - Do all records in the dataset contain the same fields?
  - How to access the same fields across records? By position? Name?
- Are the records nested?
  - No: one atomic (singular) value per field
  - Yes: collection of 0-to-many values in a field
- Encoding
  - How are values encoded? Strings? Codes? Binary?
  - How complex are the individual values?
    - Primitive: numbers and short strings?
    - Unstructured data: natural language text, audio, video
  - Can you decode?

```
Lecture4]> gunzip -c D01.gz | less
```

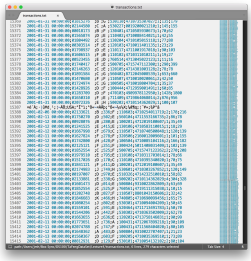


Structure | Granularity | Faithfulness | Temporality | Scope

Modern Text Editor: SublimeText

- Assessing
  - Coloring
  - Minimap
- Transformation
  - Do not destroy!
  - Names/Versions?

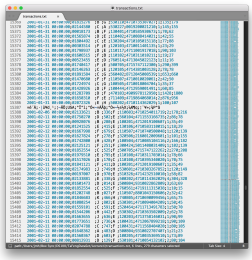
More on this soon!



Structure | Granularity | Faithfulness | Temporality | Scope

Structure Questions: A Checklist

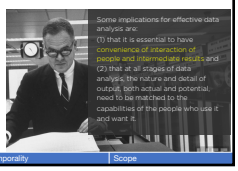
- Coarse structure
  - Is the data structured as a collection of records?
  - How are the individual records delimited in the dataset?
  - How are the record *fields* delimited from one another?
  - Do all records in the dataset contain the same fields?
  - How to access the same fields across records? By position? Name?
- Is the data nested?
  - No: one atomic (singular) value per field
  - Yes: collection of 0-to-many values in a field
- Encoding
  - How are values encoded? Strings? Codes? Binary?
  - How complex are the individual values?
    - Primitive: numbers and short strings?
    - Unstructured data: natural language text, audio, video
  - Can you decode?



Structure | Granularity | Faithfulness | Temporality | Scope

Unboxing with Trifacta

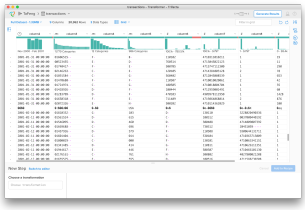
- Visual Profiling + Transformation
  - Iterative and ongoing
- Predictive Interaction
- Transformation language: Wrangle
  - Superset of relational algebra
  - Wrangle (and Pandas) both related to SQL
  - Generates a "recipe" (script)
    - You can run it on data to generate new data
- Scale & Interoperability
  - Paid version works with "big data"
    - Spark/Hadoop



Structure | Granularity | Faithfulness | Temporality | Scope

Structure Questions: A Checklist

- Coarse structure
  - Is the data structured as a collection of records?
  - How are the individual records delimited in the dataset?
  - How are the record *fields* delimited from one another?
  - Do all records in the dataset contain the same fields?
  - How to access the same fields across records? By position? Name?
- Is the data nested?
  - No: one atomic (singular) value per field
  - Yes: collection of 0-to-many values in a field
- Encoding
  - How are values encoded? Strings? Codes? Binary?
  - How complex are the individual values?
    - Primitive: numbers and short strings?
    - Unstructured data: natural language text, audio, video
  - Can you decode?



Structure | Granularity | Faithfulness | Temporality | Scope

So Far: Assessing Structure

- Basic tools
  - UNIX commands
  - Text editors
- Simple data transformation
  - Edit-in-place, character-by-character or Find/Replace
    - Save versions of files
  - Scripted transformations
    - Save scripts, inputs, outputs

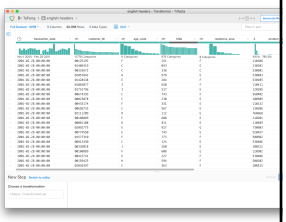
Structure | Granularity | Faithfulness | Temporality | Scope

Value Encodings: Primitive Types

Numbers and Strings
We'll spend time here today

Common special-case "primitives"
- Date/Time
- Geolocations

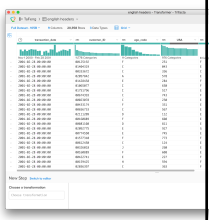
Space and time can be surprisingly subtle/messy!



Structure | Granularity | Faithfulness | Temporality | Scope

Primitive Type Semantics

- Categorical #1: Nominal
  - E.g. political party affiliation
  - Note: Sometimes even numeric data is nominal
- Categorical #2: Ordinal
  - E.g. education level (none, HS, College, Post-College)
  - Ordered, but not particularly *quantitative*
- Quantitative: amounts, measures
  - Negative or positive
  - Arithmetic "makes sense" (e.g. difference, ratio)
  - Integers vs. Rational numbers



Who cares?
- Affects how you interpret the data
- E.g. In visualization or summarization

Structure | Granularity | Faithfulness | Temporality | Scope

In Sum: Tools and Structure

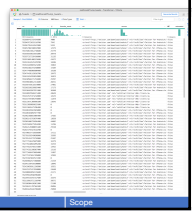
Coarse structure
- Identifying data "shape", records/fields and delimiters
- Value encodings
- Categorical, Ordinal, Quantitative

Basic tools
- UNIX commands
- Text editors

Simple data transformation
- Edit-in-place, character-by-character or Find/Replace
- Scripted transformations

Structure | Granularity | Faithfulness | Temporality | Scope

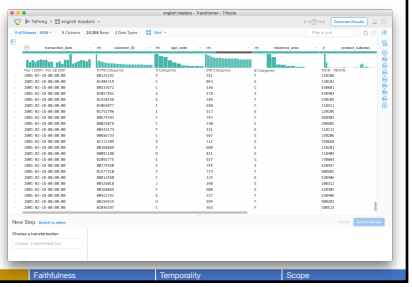
Assessing Granularity



Structure | Granularity | Faithfulness | Temporality | Scope

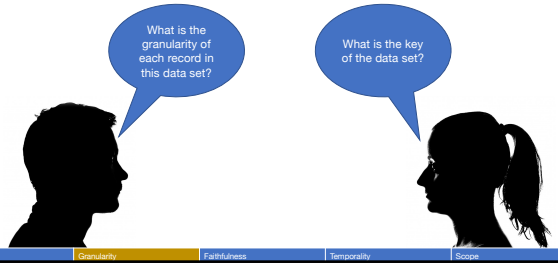
What does each TaFeng record represent?

Theories?
Justification?



Structure | Granularity | Faithfulness | Temporality | Scope

Assessing Granularity



Structure | Granularity | Faithfulness | Temporality | Scope

Key ("primary key")

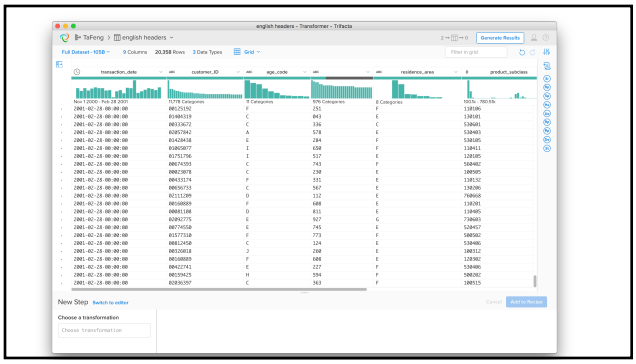
Given a relation, a key attribute uniquely determines the values in each record
- E.g. an *identifier* like `transaction_id`
- Can be *composite*: e.g. `(City, State)`

Each value occurs **at most once** in the key column(s).

Hence the semantics (meaning) of the key determines granularity
- `customer_id?` `transaction_time?`
- `(age_code, day_of_week)`

Be careful!
- Goofy key choice? Goofy data.
- Real-world data may have "noisy", duplicated keys to clean up

Structure | Granularity | Faithfulness | Temporality | Scope



Joining tables with keys

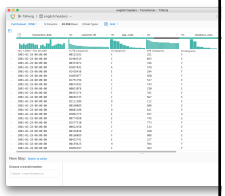
- Keys in our data
  - `age classes.txt`: `code`
  - `residence area.txt`: `code`
  - `transactions.txt`: `transaction_id`
- Transactions.txt also has *foreign keys*
  - i.e. attributes that reference the key of another table
  - `age_code` is a foreign key to `age classes.txt`
  - `residence_area` is a foreign key to `residence area.txt`
- Joining on a foreign key is a "lookup"
  - At most one match for each row of `TaFengTransactions.txt`

More on this in future lectures!

Structure | Granularity | Faithfulness | Temporality | Scope

Granularity Questions: a Checklist

- What kind of thing does each record represent?
- Do all records capture granularity at the same level?
- What alternative interpretations of the records are there?
  - E.g. Is this a list of transactions? Or a list of customers?
- What kinds of aggregation is possible/desirable?
  - From individual people to demographic groups?
  - From individual events to totals across time or regions?
  - Hierarchies (city/county/state, second/minute/hour/day)



Structure | Granularity | Faithfulness | Temporality | Scope

So Far: Assessing Granularity

Identifying (primary) keys can help assess Granularity
- Each record is the information *per key*

Foreign Keys are pointers to individual rows in other data sets
- To lookup a row in another table: join the foreign key to its primary key

Structure | Granularity | Faithfulness | Temporality | Scope

Assessing Faithfulness

Theme: the Faithfulness of a record can only be evaluated in context

- Application context
- Context in your data set
  - Across records

Students

id: integer	DOB: date	GPA: float	Rank: float
123457	01/16/1997	3.2	465
123458	01/24/2017	2.7	28
123457	01/16/2002	5.0	27
123459	03/22/1996	3.6	31
123460	06/13/1997	2.2	43

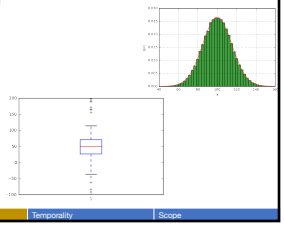
Structure | Granularity | Faithfulness | Temporality | Scope

Faithfulness Across Records: Outliers

What is an "outlier"?
- A value that is "far" from the "center"

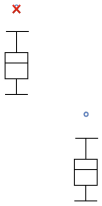
More on this in future lectures!

Distribution-based definition
- Center (e.g. average, median)
- Spread (e.g. standard deviation, IQR)



Structure | Granularity | Faithfulness | Temporality | Scope

What to do with Outliers?

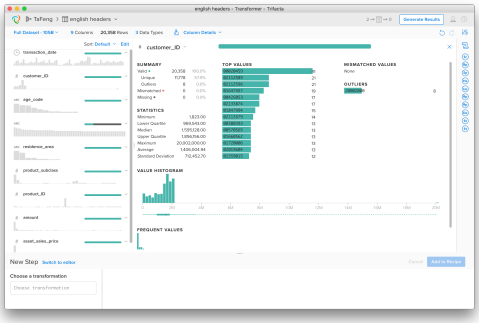


Delete. ("trimming")

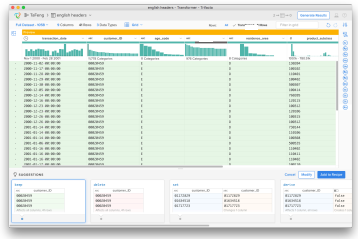
Set to a default
- E.g. the nearest non-outlier ("Winsorizing")

Good Hygiene:
- Leave the original column
- Derive an indicator column to flag presence of outlier
- Derive a clean column for your use

Structure | Granularity | Faithfulness | Temporality | Scope



Assessing Faithfulness Within Records: Dependencies and Correlations



Structure | Granularity | Faithfulness | Temporality | Scope

Functional Dependencies (FDs)

More on this in future!

- Generalization of Keys
- Attribute A *determines* Attribute B
  - `customer_id -> age_code`
  - i.e. age_code = f(customer_id)
    - Function represented as lookup table
- More generally, a set of columns determines another set of columns
  - `{transaction_time,customer_id} -> {age_code, residence_area}`
- Primary Keys are special FDs
  - Right-hand-size is the set of *all* attributes in the relation

Structure | Granularity | Faithfulness | Temporality | Scope

Correlations

More on this in future!

Dependence (i.e. lack of independence!) between 2 random variables

Think of the attributes in a relational schema
- An *instance* of that relation was generated from some real-world process
- Each column of that relation is a "random variable" generated by the process

A Functional Dependency is a "deterministic" correlation

Correlations are more general: statistical relationships
- `amount` and `sales_price` are correlated

Structure | Granularity | Faithfulness | Temporality | Scope

What to do about bad FDs/Correlations

- Cleaning Noisy FDs: `customer_id -> age_code` (kinda)
  - For a few customer IDs, there are multiple values of age_code
  - Set offending right-hand-side values to all match
  - Set offending left-hand-side values to NULL
- Cleaning Correlations: `height` correlated with `weight`
  - Some rows don't seem to follow the correlation (how do we decide?)
  - Can *impute* a likely value for one side or the other
  - Can set one side or the other to NULL
- Don't forget Good Hygiene!
  - Leave the original column
  - Derive new columns (indicators and/or cleaned data)

Careful! More on this in future.

Structure | Granularity | **Faithfulness** | Temporality | Scope

Faithfulness Questions: A Checklist

- Type-specific Faithfulness checks
  - Are dates and addresses legal/reasonable?
  - Numeric codes legal?: E.g. phone numbers, credit cards, social-security numbers etc.
  - Can you validate network endpoints? Email addresses, IP addresses, social network names
  - Need to deduplicate named entities: misspellings, acronyms (UCB vs Berkeley)
- Checks for data entry problems
  - Frequency outliers for common default entry values (00000, 1234567)
  - Misspellings (compare to dictionaries)
  - Sensor drift – often timeseries-based
  - "Curbstoning" in surveys
- Quantitative dirty data
  - Outliers, FDs, Correlations
- Check distribution of inaccuracies
  - Do inaccuracies seem to affect a large fraction of records?
  - Are they concentrated in a particular subset of records?

Tricky!

Structure | Granularity | **Faithfulness** | Temporality | Scope

Summing Up: Faithfulness

- Outliers
- Functional Dependencies & Correlations
- Good Data Cleaning Hygiene
  - Don't overwrite: use indicators and derived columns

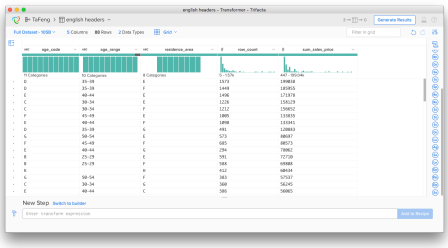
Structure | Granularity | **Faithfulness** | Temporality | Scope

Granularity *Transformations*

- We can coarsen the granularity by picking new keys and "rolling up"
  - GroupBy and Aggregation
- GroupBy
  - Choose a new primary key (the group-by columns)
  - Result will have one row per distinct values of this primary key
- Aggregation
  - Summary (rollup) results per group
  - E.g. count(), or aggregation functions on attributes (sum(x), average(x), stdev(x) etc.)

Structure | **Granularity** | Faithfulness | Temporality | Scope

Finalizing Ta Feng



What About Non-Rectangular Data

- Nested Data
- Free Text (i.e. for humans)
- Example that has both: Twitter feed

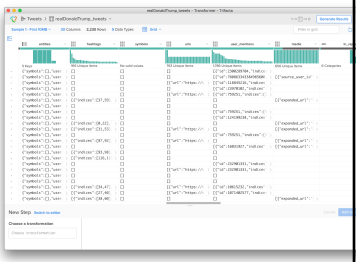
Structure | Granularity | Faithfulness | Temporality | Scope

Unnesting Nested Data Types

More on this in future lectures!

Maps
- A.k.a. dictionaries, hashes
- A set of key:val pairs

Arrays
- Or lists

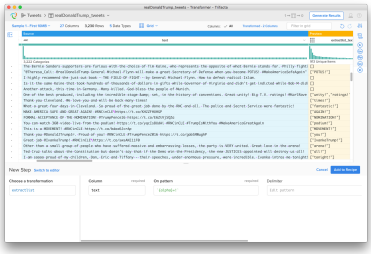


Structure | Granularity | Faithfulness | Temporality | Scope

Playing with Text I: String Manipulation

More on this in future lectures!

Regular expression (regex) extraction



Structure | Granularity | Faithfulness | Temporality | Scope

Playing with Text II: Natural Language

Natural Language Processing: a slippery slope
- Entity Resolution
  - IBM vs. International Business machines
- Named Entity Recognition
  - Steve Jobs munched on an apple as he announced new jobs at Apple.
- Sentiment
  - My windows box crashed yet again.
- Etc. Etc.
- Hot topic today: Dialog systems:
  - Alexa, where did I leave my keys?
    - I don't know. It's a pity you don't have a tracker on your key fob.
  - A what?
    - A bluetooth tracker – there are variety of them for sale
  - Stop trying to get me to buy stuff!
    - Sorry about that; I know you've been trying to save this month…
    - …but if you had one I could locate your keys.

Structure | Granularity | Faithfulness | Temporality | Scope

Playing with Text II: Natural Language

- Some simple things
  - Term frequencies
  - Simple Sentiment analysis

Structure | Granularity | Faithfulness | Temporality | Scope

Quick Note: Assessing Temporality

- Often two kinds of time in data
  - Time of data entry
  - Time of a recorded phenomenon being "true"
    - E.g. A physical time of an event happening
    - E.g. An "effective" time, e.g. date that a subscription will start
- Often more
- Time is tricky!
  - Periodicities (recurring patterns in Days of the week, or Months of the year)
  - Non-uniform hierarchy of units (# days in a month, # of days in a year, etc.)
  - Time zones are complex: especially daylight savings (summer) time
  - Clocks can be skewed
  - Relativity: true perception of event may vary (yep!)
- Assess timestamps in data carefully!!

Structure | Granularity | Faithfulness | Temporality | Scope

Quick Note: Assessing Scope

- Do you have all the data you need
  - Missing columns
    - Join in external data
  - Missing rows/values?
    - Look for sequential patterns with breaks
    - Is this a good sample?
      - Granularity
      - Extent
    - How to *impute* reasonable stand-in values

More on this in future lectures!

- Can be quite application specific/subjective!

Structure | Granularity | Faithfulness | Temporality | Scope

Looking Back: Data Transformations

Modifying structure
- Splitting rows and columns ("splitrows", "split")
- Unnesting ("flatten")

Hiding Things
- Rows ("keep", "delete" a.k.a. "select")
- Columns ("drop", "aggregate")

Adding Things
- "derive" a.k.a. "map", "apply"

Changing Things
- "replace"/"set"

Text manipulation
- "extract", "replace", "set"

GroupBy/Aggregate arithmetic
- "aggregate", a.k.a. "group by", "reduce"

Multi-table Operations
- "join", "lookup"
- "union"

Looking Back: Bigger Picture

*What **decisions** did we make along the way?*
- Data that got hidden
  - Choice of columns to drop
  - Values we cleaned: indicators, cleaned columns
  - Filtering of rows (vs. indicators)
- Data that got added
  - Other derived columns: calculations, splits, extractions
  - Joins, Unions
- Changes in granularity:
  - Coarser: grouping keys and aggregates
  - Finer: unnesting

*How might the data **wrangling** influence the **analyses** we can do?*
- Sometimes we won't figure this out until analysis begins
- And we loop back to wrangling!

NSFW

Never Stop Fiercely Wrangling

Your analysis depends on it.

More fun:
https://github.com/Quartz/bad-data-guide