Bradley Voytek, Ph.D.

UC San Diego

Department of Cognitive Science
Halıcıoğlu Data Science Institute
Neurosciences Graduate Program

bvoytek@ucsd.edu
@bradleyvoytek

Administrative stuff

- **REQUIRED PROJECT LECTURE**
  - 2019 November 05 (Tuesday)

- **FINAL REQUIRED LECTURE**
  - Lukas Biewald: Co-founder *Weights & Biases*
  - 2019 November 07 (Thursday)

Administrative stuff

- **PYTHON HELP!**
  - https://www.codecademy.com/learn/learn-python-3

Administrative stuff

- **GIT HELP!**
  - http://codecademy.com/learn/learn-git

Administrative stuff

- **DATA SCIENCE HELP!**
  - https://software-carpentry.org/

COGS 108
Introduction to Data Science

Data science questions

UC San Diego

Data science questions

"If I had four hours to chop down a tree, I'd spend the first two hours sharpening the axe."

Data science questions

- Be specific
- Be answerable with data
- Specify what's being measured

What makes a good
data science
question?

Data science questions

Specifying what you're going to measure is important

Poor questions that leave wiggle room for useless answers:

- What can my data tell me about my business?
- What should I do?
- How can I increase my profits?

Data science questions

Specifying what you're going to measure is important

Good questions where the answer is impossible to avoid:

- How many Model 3s will Tesla sell in San Diego in 2020?
- How many parking spaces does UCSD need?
- How many students should UCSD admit in 2019 for a target class size of 5000?

COGS 108
Introduction to Data Science

Data sources

UC San Diego

Types of data

Structured & semi-structured

Spreadsheets (CSVs, .xlsx)
JSON & XML
relational databases (SQL)

Unstructured

everything else: video, audio, images, websites, apps, text, etc.

CSV

Each column separated by a comma

```
10/29/19 9:33,170,165,15,100000,100
10/29/19 9:34,183,160,10,10000000,90
10/29/19 9:34,182,164,16,123456,727
10/29/19 9:34,162,165,15,10000,5
10/29/19 9:34,174,165,30,200000,15
10/29/19 9:34,168,172,21,1000,200
10/29/19 9:34,159,175,50,500,100
10/29/19 9:34,170.18,173.6,10,34600000,73
10/29/19 9:34,190,160,30,6,60
10/29/19 9:34,170,170,25,30,15
```

Each row is separated by a new line

JSON

JSON: key-value pairs

nested/hierarchical data

{"Name": "Isabela"}

key

value

JSON

These are all nested within `attributes`

These are all nested within "Good For"

```
"attributes": {
  "Take-out": true,
  "Wi-Fi": "free",
  "Drive-Thru": true,
  "Good For": {
    "dessert": false,
    "latenight": false,
    "lunch": false,
    "dinner": false,
    "breakfast": false,
    "brunch": false
  },
```

XML

A **node**

```
$node
<tag>
    <tag2> more content
</tag2>
    <tag3> more content
</tag3>
</tag>
```

An *opening* **tag**

An **element**

A *closing* **tag**

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<customers>
    <customer>
        <customer_id>1</customer_id>
        <first_name>John</first_name>
        <last_name>Doe</last_name>
        <email>john.doe@example.com</email>
    </customer>
    <customer>
        <customer_id>2</customer_id>
        <first_name>Sam</first_name>
        <last_name>Smith</last_name>
        <email>sam.smith@example.com</email>
    </customer>
    <customer>
        <customer_id>3</customer_id>
        <first_name>Jane</first_name>
        <last_name>Doe</last_name>
        <email>jane.doe@example.com</email>
    </customer>
</customers>
```

Relational database

1. Efficient Data Storage
2. Avoid Ambiguity
3. Increase Data Privacy

Results | Messages

	EmployeeId	FirstName	LastName	DepartmentName
1	1	Ken	Sanchez	Executive
2	2	Terri	Duffy	Engineering
3	3	Roberto	Tamburello	Engineering
4	4	Rob	Walters	Engineering
5	5	Gail	Erickson	Engineering
6	6	Jossef	Goldberg	Engineering
7	7	Dylan	Miller	Support
8	8	Diane	Margheim	Support
9	9	Gigi	Matthew	Support
10	10	Michael	Raheem	Support

Relational database

Information is stored across tables

	unique_identifier	
	AH13JK	
	JJ29JJ	
	CI21AA	

	unique_identifier	
	AH13JK	
	JJ29JJ	
	JJ29JJ	
	XJ11AS	
	CI21AA	

	unique_identifier	
	AH13JK	
	SE92FE	
	CI21AA	

entries are *related* to one another by their unique identifier

Relational database

restaurant

name	id	address	type
Taco Stand	AH13JK	1 Main St.	Mexican
Pho Place	**JJ29JJ**	192 Street Rd.	Vietnamese
Taco Stand	XJ11AS	18 W. East St.	Fusion
Pizza Heaven	CI21AA	711 K Ave.	Italian

health inspections

id	inspection_ date	inspector	score
AH13JK	2018-08-21	Sheila	97
JJ29JJ	2018-03-12	D'eonte	98
JJ29JJ	2018-01-02	Monica	66
XJ11AS	2018-12-16	Mark	43
CI21AA	2018-08-21	Anh	99

rating

id	stars
AH13JK	4.9
JJ29JJ	4.8
XJ11AS	4.2
CI21AA	4.7

Relational database

restaurant

name	id	address	type
Taco Stand	AH13JK	1 Main St.	Mexican
Pho Place	JJ29JJ	192 Street Rd.	Vietnamese
Taco Stand	XJ11AS	18 W. East St.	Fusion
Pizza Heaven	CI21AA	711 K Ave.	Italian

Two different restaurants with the same name will have different unique identifiers

health inspections

id	inspection_date	inspector	score
AH13JK	2018-08-21	Sheila	97
JJ29JJ	2018-03-12	D'eonte	98
JJ29JJ	2018-01-02	Monica	66
XJ11AS	2018-12-16	Mark	43
CI21AA	2018-08-21	Anh	99

rating

id	stars
AH13JK	4.9
JJ29JJ	4.8
XJ11AS	4.2
CI21AA	4.7

Variable type

Quantitative data consists of numerical values, like height and weight.

Categorical data consists of labels describing the properties of the objects under investigation, like gender, hair color, and occupation.

Categorical data doesn't have an order to it.

Does it make any sense to talk about the maximum or minimum hair color? What is the interpretation of my hair color minus your hair color?

Inferences

Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

DepartmentName
Sales
Engineering
Clerical
Marketing

0- or 1-based indexing?!

Others database issues

King Filip given dozens of speeding fines

The King is accused of having broken the speed limit on numerous occasions in his official car with registration number "1".

The French police tell the king that if he doesn't pay up his car will be confiscated. However, no fines have been issued to other cars that form the royal motorcade. Furthermore, the royal BMW has only been in France once since King Filip became monarch.

Others database issues

An investigation was launched and a computer error was found to be behind the deluge of fines that had been issued to the King of the Belgians. Since the introduction of a new vehicle registration system in Belgium in 2010, most new registration numbers start with the number "1".

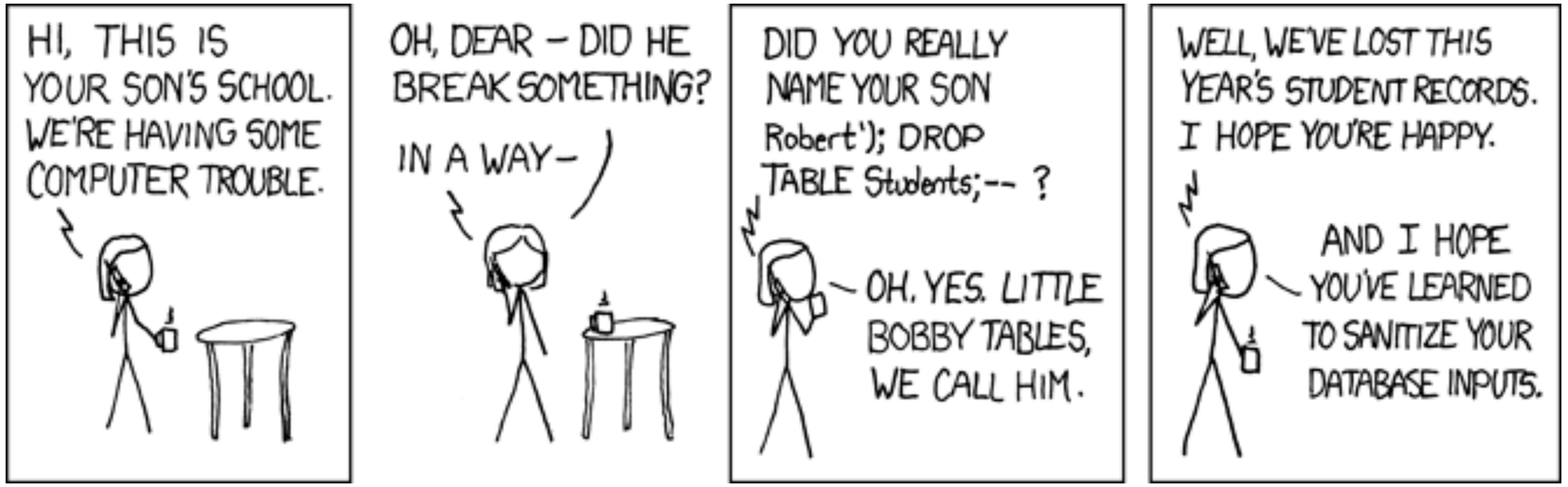
However, not all French speed cameras have not been adapted to take account of this and if there is the slightest problem reading a numberplate, the French traffic police computer only remembers the first number of the numberplate, in this case '1'. As this is also the King's numberplate, fines have been issued even though King Filip and his car weren't even in France at the time when the alleged speeding incidents took place.

Others database issues

The French authorities accept that the fines were issued in error and they have since been quashed.

Others database issues

Others database issues



Let's say the name was used in a variable, `$Name`. You then run this query:

```
INSERT INTO Students VALUES ( '$Name' )
```

What you get is:

```
INSERT INTO Students VALUES ( 'Robert' );  DROP TABLE STUDENTS; --')
```

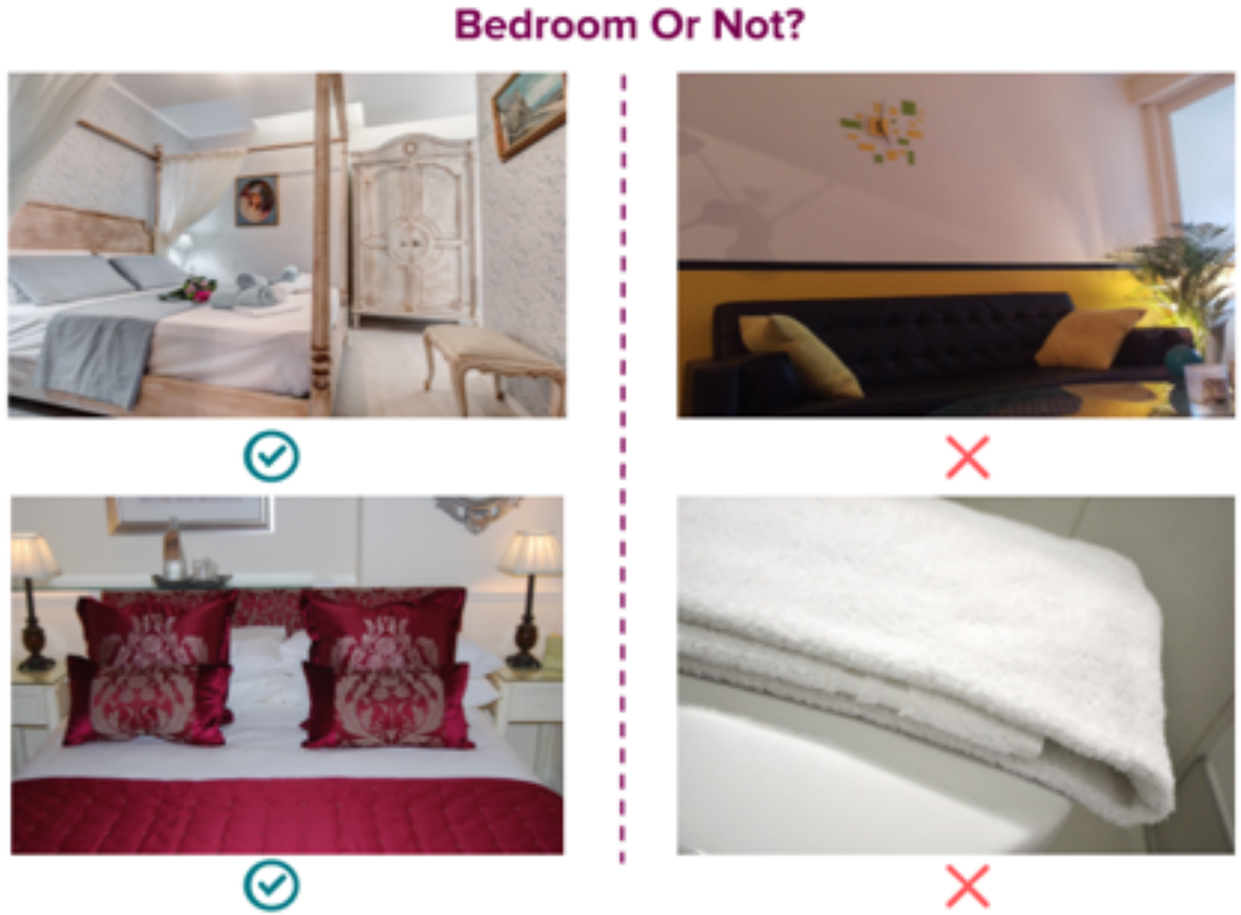
The `--` only comments the remainder of the line.

Unstructured data

Some datasets record information about the state of the world, but in a more heterogeneous way.

Perhaps it is a large text corpus with images and links like Wikipedia, or the complicated mix of notes and test results appearing in personal medical records.

Unstructured data

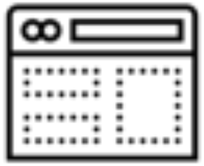


"The left two photos were correctly predicted as bedrooms; The right two photos were correctly predicted NOT as bedrooms."

Source: https://medium.com/airbnb-engineering/categorizing-listing-photos-at-airbnb-f9483f3ab7e3

Unstructured data types

Text files and documents



Websites and applications

Sensor data

Image files

Audio files

Video files

Email data

Social media data

COGS 108
Introduction to Data Science

Pandas

UC San Diego

Pandas

Preprocessing our data

Much of what data scientists do involves cleaning and preprocessing data:

- Handling missing or invalid values
- Extracting usable information from messy strings
- Transforming/normalizing variables and variable names
- Filtering redundant or bad data
- Merging with other datasets
- Etc...

Source: Tal Yarkoni

Pandas

Pandas data structures

- Provides functionality similar to data frames in R
- Two main data structures: Series and DataFrames
- A Series is a 1-dimensional numpy array with axis labels

Source: Tal Yarkoni

Pandas

```
# Initialize a Series from a numpy array and index labels
a = np.arange(3, 8)
b = pd.Series(a, index=['apple', 'banana', 'orange', 'pear', 'grapes'])

# Let's take a look...
print(b)
```

```
apple     3
banana    4
orange    5
pear      6
grapes    7
dtype: int64
```

Source: Tal Yarkoni

UC San Diego

Pandas

```
# Unlike numpy arrays, we can now refer to elements by label.
# The syntax is similar to dictionary indexing. You can also
# treat labels like attributes (e.g., b.pear), but this runs
# the risk of collisions and should be avoided.
print(b['pear'])

# We can always retrieve the underlying numpy array with .values
print(b.values)

# Many numpy operations work as expected, including slicing
print(b[2:4])

# Each column in our loaded dataset is a Series
print(data['Breed'][:5])
```

```
6
[3 4 5 6 7]
orange    5
pear      6
dtype: int64
0    Labrador Retriever Mix
1    Domestic Shorthair Mix
2    Domestic Shorthair Mix
3    Domestic Shorthair Mix
4                Bulldog Mix
Name: Breed, dtype: object
```

Source: Tal Yarkoni

UC San Diego

Pandas

The pandas DataFrame

- The workhorse of data analysis in pandas
- A container of multiple aligned Series
- Heterogeneous: a DF's Series can have different dtypes

Source: Tal Yarkoni

UC San Diego

Pandas

Indexing pandas DataFrames

- pandas DFs spport flexible indexing by labels and/or indices
  - A common gotcha: R-style indexing won't work
  - Be explicit about whether you're using integer or label indexing

Source: Tal Yarkoni

UC San Diego

Pandas

Python: 0-indexing
R: 1-indexing

Indexing pandas DataFrames

- pandas DFs spport flexible indexing by labels and/or indices
  - A common gotcha: R-style indexing won't work
  - Be explicit about whether you're using integer or label indexing

Source: Tal Yarkoni

UC San Diego

Pandas

```
# This won't work!
data[0, 'Animal Type']

# # but .ix supports mixed integer and label based access
data.ix[0, 'Animal Type']

# # Returns the entire column
data['Animal Type']

# # Position-based selection; returns all of rows 2 - 5
data.iloc[2:5]

# # Returns rows 2 - 5, columns 2 and 7
data.iloc[2:5, [2, 7]]

# # Label-based indexing; equivalent to data['Animal Type']
# # in this case
data.loc[:, 'Animal Type']
```

Source: Tal Yarkoni

UC San Diego

Pandas

Slide Type Fragment

```
data.describe()
```

	Animal ID	Name	DateTime	MonthYear	Outcome Type	Outcome Subtype	Animal Type	Sex upon Outcome	Age upon Outcome	Breed	Color
count	43870	30614	43870	43870	43861	21197	43870	43869	43836	43870	43870
unique	40612	9939	36235	36235	8	18	5	5	45	1792	433
top	A694501	Bella	08/11/2015 12:00:00 AM	08/11/2015 12:00:00 AM	Adoption	Partner	Dog	Neutered Male	1 year	Domestic Shorthair Mix	Black/White
freq	8	207	25	25	17342	11652	24964	15645	7478	13039	4602

Pandas

Importing data

- Before we do anything else, we need to get our data into a usable form
- Most commonly, data will come from a flat file
- But sometimes we need to retrieve data from other sources
- We'll do both

Source: Tal Yarkoni

UC San Diego

Not Pandas

Reading data in with the standard library

There are many ways to read in data in Python using the standard library. Here's a simple example, where we read in the data line-by-line and split each line into its own list.

Source: Tal Yarkoni

UC San Diego

Not Pandas

```
filename = '../data/Austin_Animal_Center_Outcomes.csv'
data = []  # Initialize an empty list to store the data

# Loop over rows in the file, split each one into a list
# of values, and add the result to the data list.
for line in open(filename).readlines():
    line = line.strip().split(',')
    data.append(line)

print("Found {} rows.".format(len(data)))

# Print the 1000th row to see what it looks like
data[1000]
```

```
Found 43871 rows.

['A664984',
 'Buddy',
 '10/18/2013 06:46:00 PM',
 '10/18/2013 06:46:00 PM',
 'Adoption',
 '',
 'Dog',
 'Neutered Male',
 '1 year',
 'Pit Bull Mix',
 'Blue']
```

Source: Tal Yarkoni

UC San Diego

Pandas

Slide Type Skip

The problem with approaches like the one above is that the data lack a tabular format, making it very hard to operate over rows or columns. We're much better off using the *pandas* package to hold our data in a pandas DataFrame (DF)--a data structure that wraps around numpy arrays and is expressly designed to support a range of powerful operations over data. Reading a dataset into a pandas DF is very easy with the workhorse read_csv() or read_table() methods. These methods take a large number of optional arguments that make it easy to read in almost any kind of orderly data represented in a text file.

Pandas

Slide Type Sub-Slide

Reading data, the pandas way

Slide Type Fragment

```
# Note that we're reading the file directly from GitHub.
# pandas accepts URLs in addition to local files.

# url = 'http://raw.githubusercontent.com/tyarkoni/SSI2016/master/data/Austin_Animal_Center_O
# If you're working from the cloned course GitHub repo, comment the line above and uncomment
# the line below for faster loading.
url = '../data/Austin_Animal_Center_Outcomes.csv'

# The workhorse data-reading method in pandas.
# It accepts a LOT of optional arguments--
# see http://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_csv.html
data = pd.read_csv(url)

# calling head() on a DataFrame shows the top N rows.
data.head(5)
```

Source: Tal Yarkoni

UC San Diego

Pandas

Other formats

Pandas has built-in support for reading from or to other common formats/sources:

- Generic delimited text -- read_table()
- Excel -- read_excel()
- JSON -- read_json()
- SQL -- read_sql()
- Stata -- read_stata()
- SAS (XPORT or SAS7BDAT) -- read_sas()
- etc...

Source: Tal Yarkoni