INTRODUCTION TO DATA SCIENCE

JOSÉ MANUEL CALDERÓN TRILLA

(SLIDES BY JOHN P. DICKERSON)

Lecture #2 – 01/27/2020

CMSC320
Mondays & Wednesdays
5:00pm – 6:15pm
(… or anytime on the Internet)

COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

ANNOUNCEMENTS

Register on Discord:

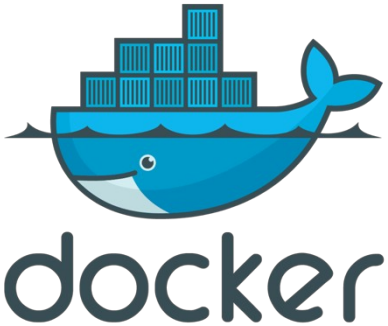
- 110 have registered already ♡
- The rest have not 💔

If you were on Discord, you'd know …

- Project 0 is out! It is "due" next Wednesday evening.
- Link: https://github.com/cmsc320/spring2021/tree/main/project0

We've also linked some reading for the week!

- First quiz will be due Monday at noon.
- Quiz will go up tomorrow (Thursday)



(A FEW) DATA SCIENCE SUCCESS STORIES & CAUTIONARY TALES

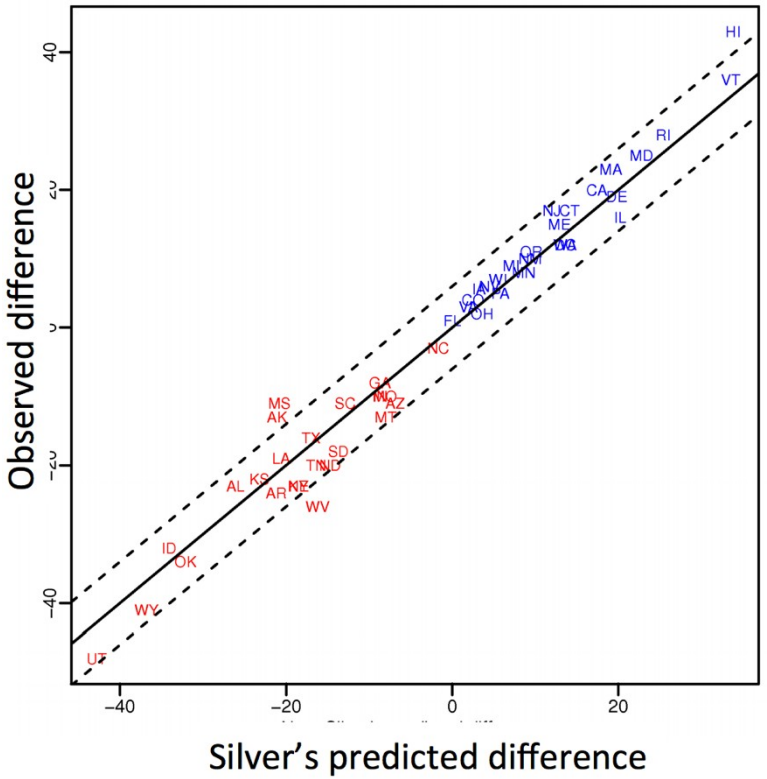
POLLING: 2008 & 2012

Nate Silver uses a simple idea – taking a principled approach to aggregating polling instead of relying on punditry – and:

- **Predicts 49/50 states in 2008**
- **Predicts 50/50 states in 2012**

- **(He is also a great case study in creating a brand.)**

https://hbr.org/2012/11/how-nate-silver-won-the-2012-p



Democrat (+) or Republican (-) in 2012

POLLING: 2016

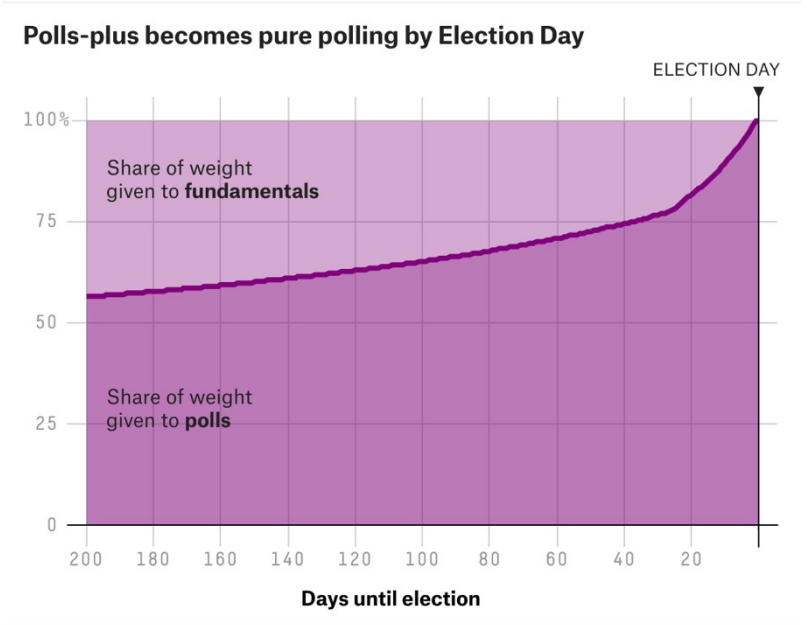
POLITICS

Nate Silver Is Unskewing Polls — All Of Them — In Trump's Direction

The vaunted 538 election forecaster is putting his thumb on the scales.

HuffPo: "He may end up being right, but he's just guessing. A "trend line adjustment" is merely political punditry dressed up as sophisticated mathematical modeling."

538: Offers quantitative reasoning for re-/under-weighting older polls, & changing as election approaches



http://www.huffingtonpost.com/entry/nate-silver-election-forecast_us_581e1c33e4b0d9ce6fbc6f7f
https://fivethirtyeight.com/features/a-users-guide-to-fivethirtyeights-2016-general-election-forecast/

AD TARGETING

Pregnancy is an expensive & habit-forming time

- **Thus, valuable to consumer-facing firms**

2012:

- **Target identifies 25 products and subsets thereof that are commonly bought in early pregnancy**
- **Uses purchase history of patrons to predict pregnancy, targets advertising for post-natal products (cribs, etc)**
- **Good: increased revenue**
- **Bad: this can expose pregnancies – as famously happened in Minneapolis to a high schooler**

http://www.businessinsider.com/the-incredible-story-of-how-target-exposed-a-teen-girls-pregnancy-2012-2

AUTOMATED DECISIONS OF CONSEQUENCE

[Sweeney 2013, Miller 2015, Byrnes 2016, Rudin 2013, Barry-Jester et al. 2015]

textio

DOXA

gild

GJ GapJumpers

zest finance

LendingClub

Kabbage

PREDPOL

NORTHPOINTE
Advancing Justice. Embracing Community.

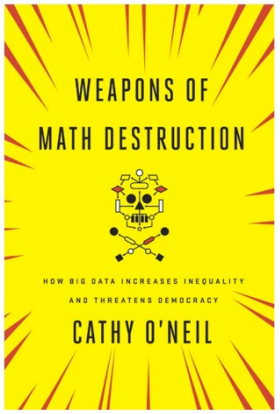
Hiring

Lending

Policing/ sentencing

Search for minority names ✉
ads for DUI/arrest records

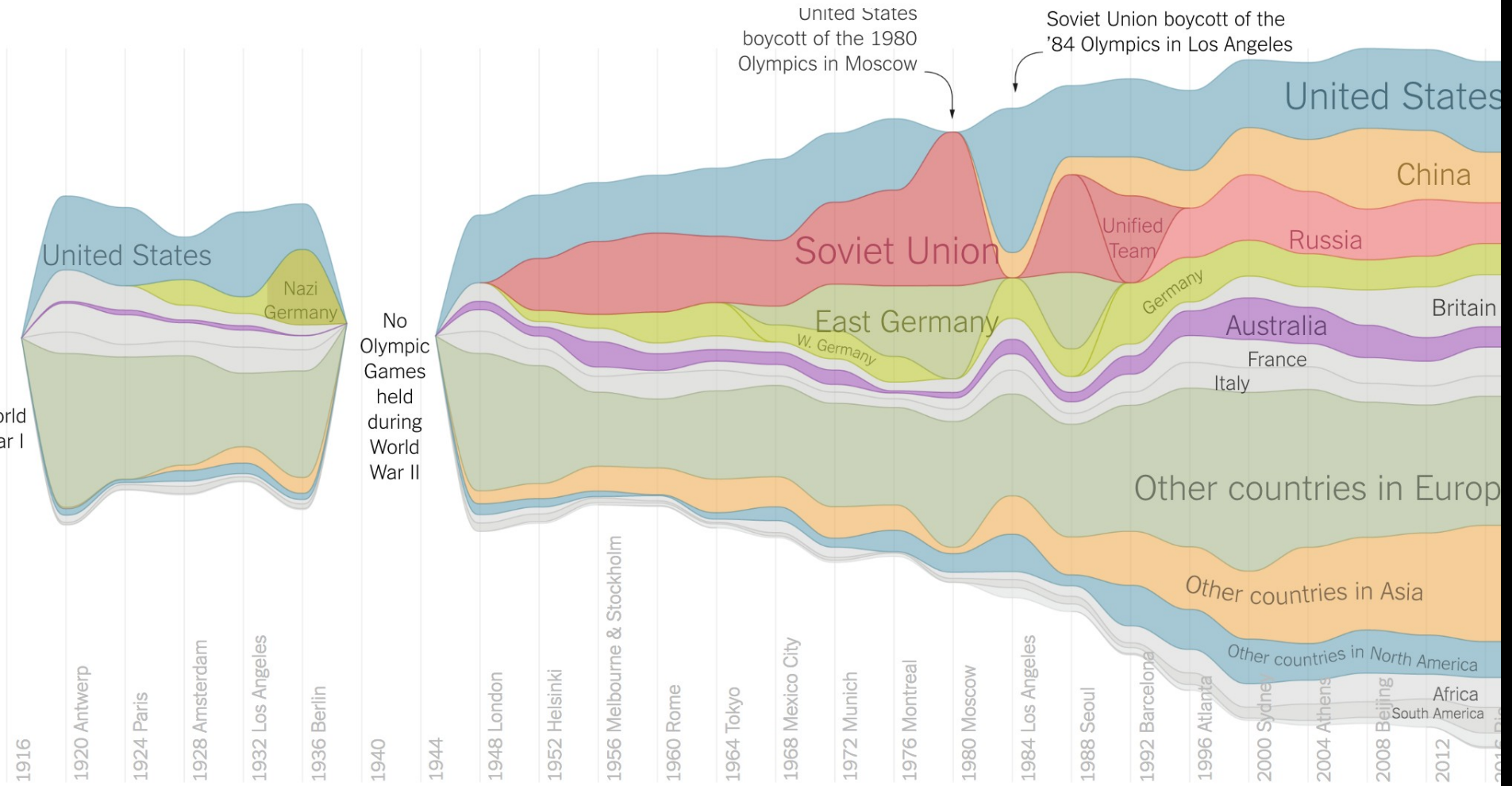
Female cookies ✉
less freq. shown professional job opening ads



"… a lot remains unknown about how big data-driven decisions may or may not use factors that are proxies for race, sex, or other traits that U.S. laws generally prohibit from being used in a wide range of commercial decisions … What can be done to make sure these products and services–and the companies that use them treat consumers fairly and ethically?"

- FTC Commissioner Julie Brill [2015]

OLYMPIC MEDALS



https://www.nytimes.com/interactive/2016/08/08/sports/olympics/history-olympic-dominance-charts.html

NETFLIX PRIZE I

Recommender systems: predict a user's rating of an item

	Twilight	Wall-E	Twilight II	Furious 7
User 1	+1	-1	+1	?
User 2	+1	-1	?	?
	-1	+1	-1	**+1**

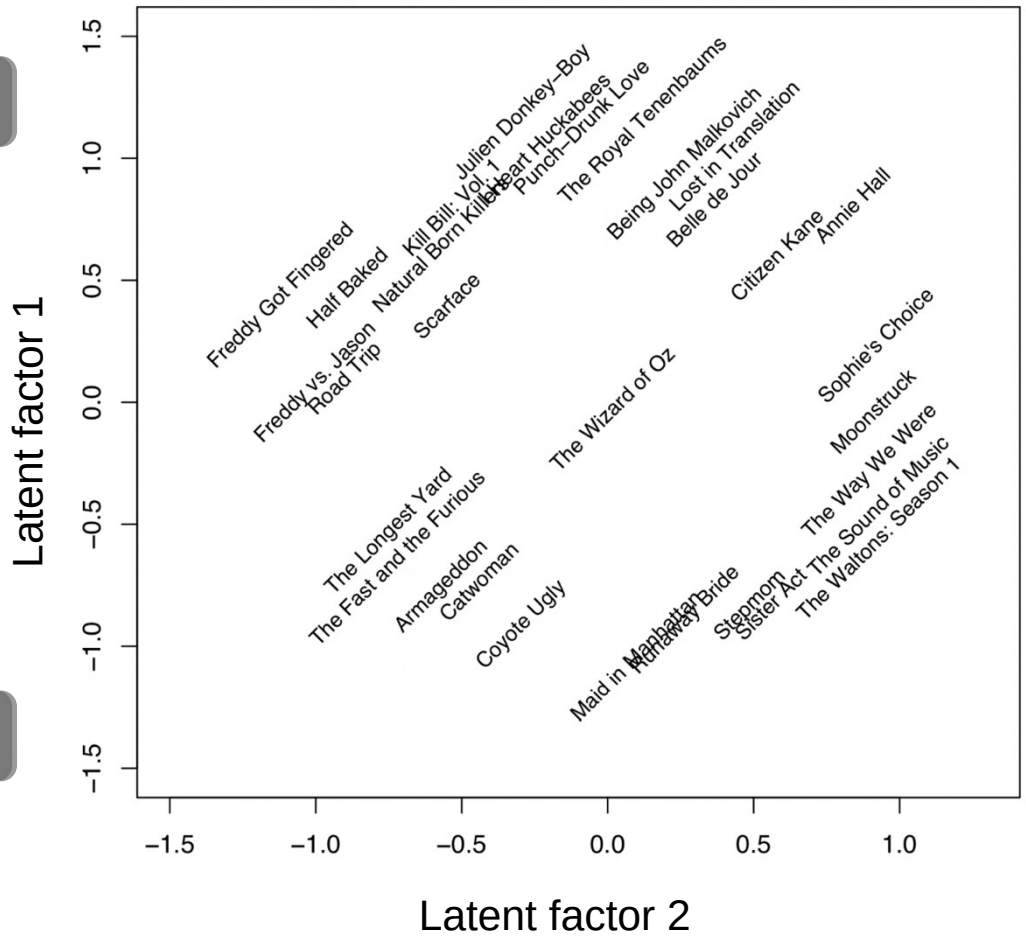
Netflix Prize: $1MM to the first team that beats our in-house engine by 10%

- **Happened after about three years**
- **Model was never used by Netflix for a variety of reasons**
  - Out of date (DVDs vs streaming)
  - Too complicated / not interpretable

NETFLIX PRIZE II

Frat/Gross-Out Comedy ⟷ Critically-Acclaimed/Strong Female Lead

Artsy ↕ BB



Latent factors model:

Identify factors with max discrimination between movies

Image courtesy of Christopher Volinsky

NETFLIX PRIZE III

Netflix initially planned a follow-up competition

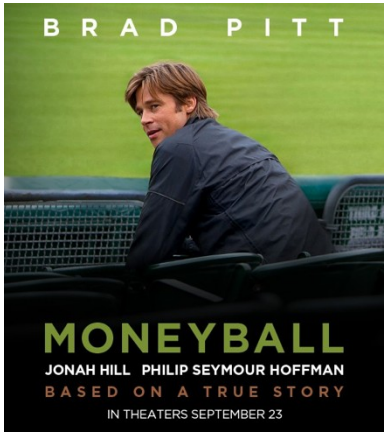
In 2007, UT Austin managed to deanonymize portions of the original released (anonymized) Netflix dataset:

- **??????????**
- **Matched rating against those made publicly on IMDb**

Why could this be bad?

2009—2010, four Netflix users filed a class-action lawsuit against Netflix over

MONEYBALL

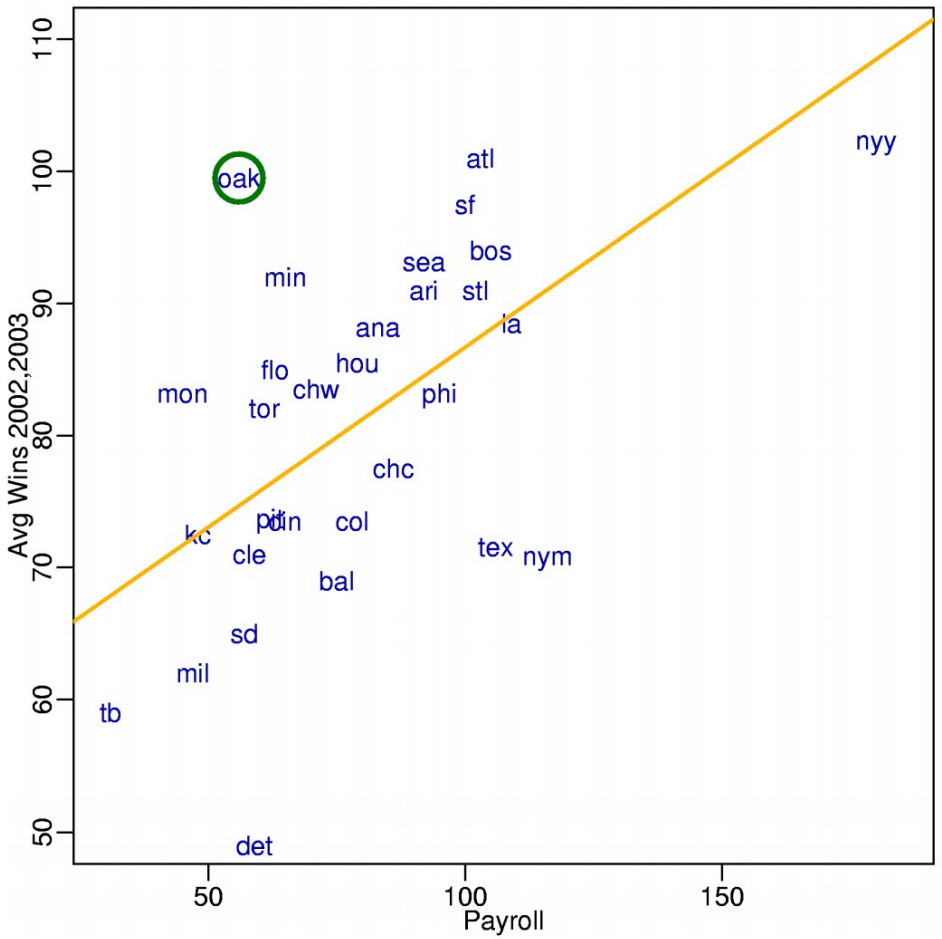


Baseball teams drafted rookie players primarily based on human scouts' opinions of their talents

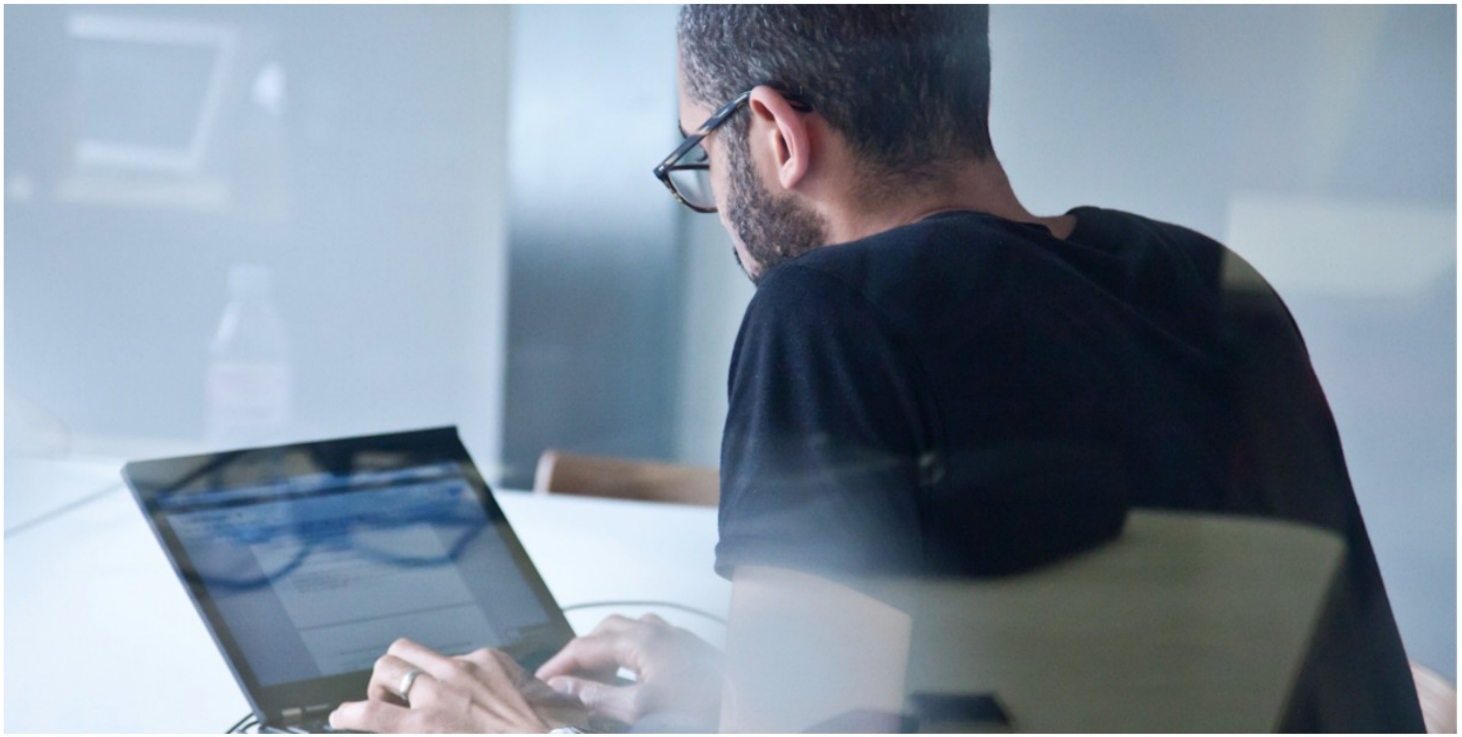
Paul DePodesta, data scientist *du jour*, convinces the {bad, poor} Oakland Athletics to use a quantitative aka sabermetric approach to hiring

(Spoiler: Red Sox offer Brand a job, he says no, they take a sabermetric approach and win the World Series.)

(Spoiler #2: DePodesta is now GM for the Browns, who are extremely bad right now. We'll see what happens!)



1. Data scientist



Shutterstock

Overall job score (out of 5.0): 4.8

Job satisfaction rating (out of 5.0): 4.4

Number of job openings: 4,184

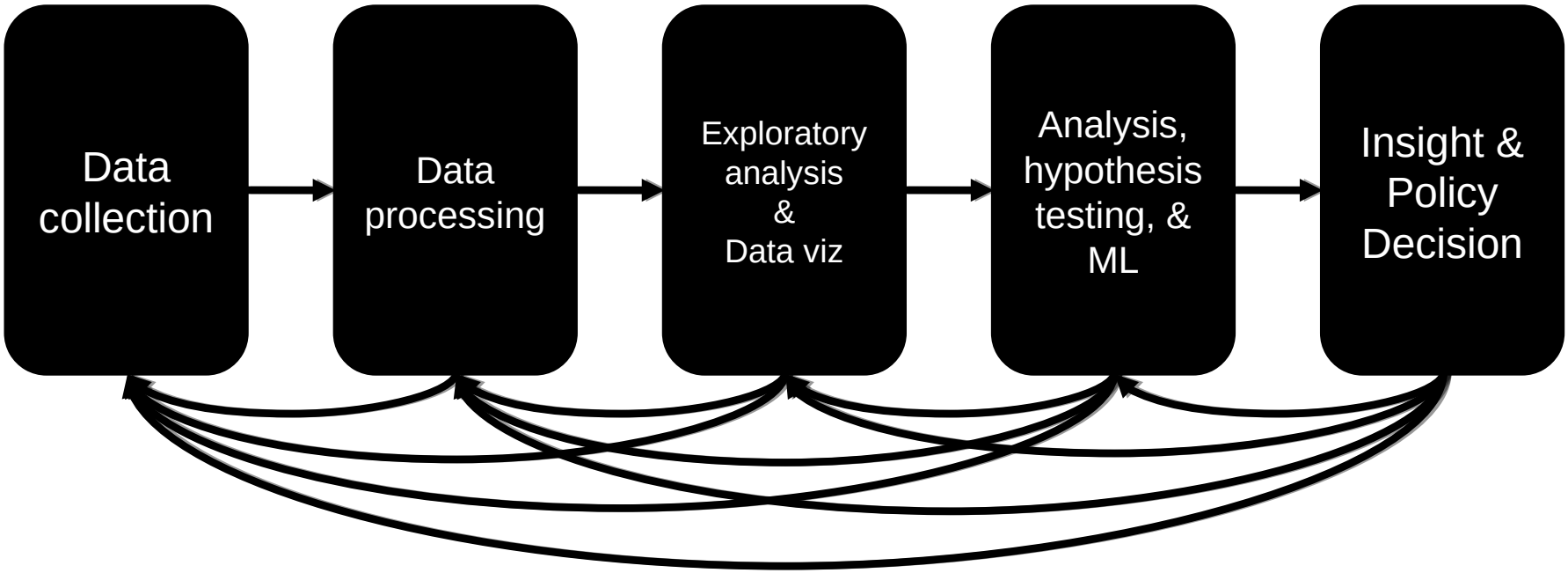
Median base pay: $110,000

http://www.businessinsider.com/best-jobs-in-america-in-2017-2017-1/

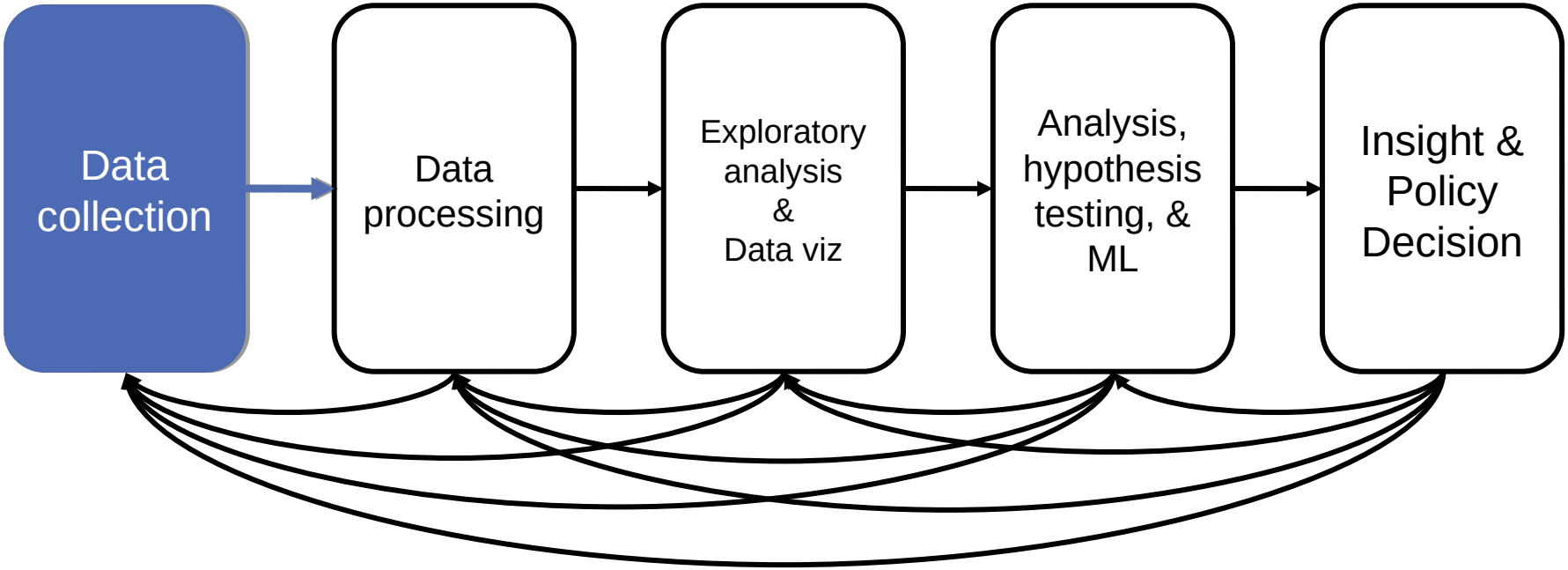
UP NEXT …

SCRAPING DATA WITH PYTHON

THE DATA LIFECYCLE



(THE REST OF) TODAY'S LECTURE



BUT FIRST, SNAKES!

Python is an interpreted, dynamically-typed, high-level, garbage-collected, object-oriented-functional-imperative, and widely used scripting language.

- **Interpreted:** instructions executed without being compiled into (virtual) machine instructions*
- **Dynamically-typed:** verifies type safety at runtime
- **High-level:** abstracted away from the raw metal and kernel
- **Garbage-collected:** memory management is automated
- **OOFI:** you can do bits of OO, F, and I programming

Not the point of this class!

- Python is fast (developer time), intuitive, and used in industry!

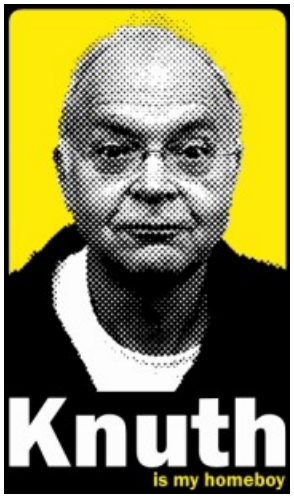
*you can compile Python source, but it's not required

THE ZEN OF PYTHON

- **Beautiful is better than ugly.**
- **Explicit is better than implicit.**
- **Simple is better than complex.**
- **Complex is better than complicated.**
- **Flat is better than nested.**
- **Sparse is better than dense.**
- **Readability counts.**
- **Special cases aren't special enough to break the rules …**
- **… although practicality beats purity.**
- **Errors should never pass silently …**
- **… unless explicitly silenced.**

Thanks: SDSMT ACM/LUG

LITERATE PROGRAMMING



Literate code contains in one document:

- the source code;
- text explanation of the code; and
- the end result of running the code.

Basic idea: present code in the order that logic and flow of human thoughts demand, not the machine-needed ordering

- Necessary for data science!
- Many choices made need textual explanation, ditto results.

Stuff you'll be using in Project 0 (and beyond)!

IP[y]: IPython Interactive Computing

jupyter

JUPYTER PROJECT

Started as iPython Notebooks, a web-based frontend to the iPython Shell

- Notebook functionality separated out a few years ago
- Now supports over 40 languages/kernels
- Notebooks can be shared easily
- Can leverage big data tools like Spark

Apache Zeppelin:

- https://www.linkedin.com/pulse/comprehensive-comparison-jupyter-vs-zeppelin-hoc-q-phan-mba-

Several others including RStudio (specific to R)

10-MINUTE PYTHON PRIMER

Define a function:

```
def my_func(x, y):
    if x > y:
        return x
    else:
        return y
```

Python is whitespace-delimited

Define a function that returns a tuple:

```
def my_func(x, y):
    return (x-1, y+2)

(a, b) = my_func(1, 2)
```

```
a = 0; b = 4
```

USEFUL BUILT-IN FUNCTIONS: COUNTING AND ITERATING

`len`: returns the number of items of an enumerable object

```
len( ['c', 'm', 's', 'c', 3, 2, 0] )
```

```
7
```

`range`: returns an iterable object

```
list( range(10) )
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

`enumerate`: returns iterable tuple (index, element) of a list

```
enumerate( ["311", "320", "330"] )
```

```
[(0, "311"), (1, "320"), (2, "330")]
```

https://docs.python.org/3/library/functions.html

USEFUL BUILT-IN FUNCTIONS: MAP AND FILTER

`map`: apply a function to a sequence or iterable

```
arr = [1, 2, 3, 4, 5]
map(lambda x: x**2, arr)
```

```
[1, 4, 9, 16, 25]
```

`filter`: returns a list* of elements for which a predicate is true

```
arr = [1, 2, 3, 4, 5, 6, 7]
filter(lambda x: x % 2 == 0, arr)
```

```
[2, 4, 6]
```

We'll go over in much greater depth with pandas/numpy.

**in Python 3, returns Iterable*

PYTHONIC PROGRAMMING

Basic iteration over an array in Java:

```
int[] arr = new int[10];
for(int idx=0; idx<arr.length; ++idx) {
    System.out.println( arr[idx] );
}
```

Direct translation into Python:

```
idx = 0
while idx < len(arr):
    print( arr[idx] ); idx += 1
```

A more “Pythonic” way of iterating:

```
for element in arr:
    print( element )
```

LIST COMPREHENSIONS

Construct sets like a mathematician!

- ***P* = { 1, 2, 4, 8, 16, …, 2^{16} }**
- ***E* = { *x* | *x* in ℕ and *x* is odd and *x* < 1000 }**

Construct lists like a mathematician who codes!

```
P = [ 2**x for x in range(17) ]
```

```
E = [ x for x in range(1000) if x % 2 != 0 ]
```

Very similar to `map`, but:

- **You'll see these way more than `map` in the wild**
- **Many people consider `map`/`filter` not "pythonic"**
- **They can perform differently (`map` is "lazier")**

EXCEPTIONS

Syntactically correct statement throws an exception:

- `tweepy` (Python Twitter API) returns “Rate limit exceeded”
- sqlite (a file-based database) returns `IntegrityError`

```
print('Python', python_version())

try:
   cause_a_NameError
except NameError as err:
   print(err, '-> some extra text')
```

PYTHON 2 VS 3

Python 3 is intentionally backwards incompatible

- (But not *that* incompatible)

Biggest changes that matter for us:

- `print "statement"` ✉ `print("function")`
- `1/2 = 0` ✉ `1/2 = 0.5` and `1//2 = 0`
- ASCII `str` default ✉ default Unicode

Namespace ambiguity fixed:

```
i = 1
[i for i in range(5)]
print(i)    # ????????
```

TO ANY CURMUDGEONS …

If you're going to use Python 2 anyway, use the `_future_` module:

- Python 3 introduces features that will throw runtime errors in Python 2 (e.g., `with` statements)
- `_future_` module incrementally brings 3 functionality into 2
- https://docs.python.org/2/library/__future__.html

```
from _future_ import division
from _future_ import print_function
from _future_ import please_just_use_python_3
```

SO, HOW DOES IMPORT WORK?

Python code is stored in module – simply put, a file full of Python code

A package is a directory (tree) full of modules that also contains a file called `__init.py__`

- Packages let you structure Python's module namespace
- E.g., `X.Y` is a submodule `Y` in a package named `X`

For one module to gain access to code in another module, it must import it

EXAMPLE

```
sound/                          Top-level package
      __init__.py               Initialize the sound package
      formats/                  Subpackage for file format conversions
              __init__.py
              wavread.py
              wavwrite.py
              aiffread.py
              aiffwrite.py
              auread.py
              auwrite.py
              ...
      effects/                  Subpackage for sound effects
              __init__.py
              echo.py
              surround.py
              reverse.py
              ...
      filters/                  Subpackage for filters
              __init__.py
              equalizer.py
              vocoder.py
              karaoke.py
              ...
```

```
# Load (sub)module sound.effects.echo
import sound.effects.echo
# Must use full name to reference echo functions
sound.effects.echo.echofilter(input, output, delay=0.7)
```

https://docs.python.org/2/tutorial/modules.html

EXAMPLE

```
# Load (sub)module sound.effects.echo
import sound.effects.echo
# Must use full name to reference echo functions
sound.effects.echo.echofilter(input, output, delay=0.7)
```

```
# Load (sub)module sound.effects.echo
from sound.effects import echo
# No longer need the package prefix for functions in echo
echo.echofilter(input, output, delay=0.7)
```

```
# Load a specific function directly
from sound.effects.echo import echofilter
# Can now use that function with no prefix
echofilter(input, output, delay=0.7)
```

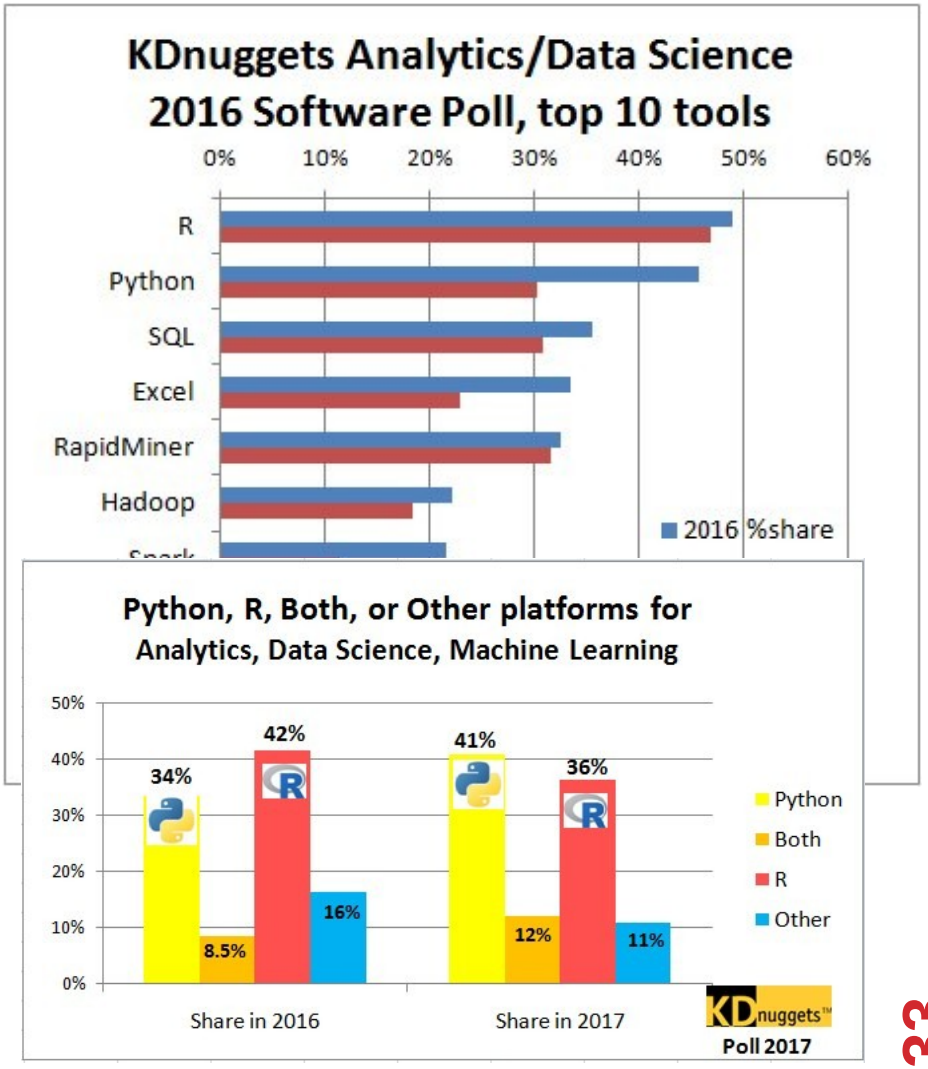
https://docs.python.org/2/tutorial/modules.html

PYTHON VS R (FOR DATA SCIENTISTS)

There is no right answer here!

- **Python is a "full" programming language – easier to integrate with systems in the field**
- **R has a more mature set of pure stats libraries …**
- **… but Python is catching up quickly …**
- **… and is already ahead specifically for ML.**

You will see Python more in the tech industry.



EXTRA RESOURCES

Plenty of tutorials on the web:

- https://www.learnpython.org/

Work through Project 0, which will take you through some baby steps with Python and the Pandas library:

- (We'll also post some more readings soon.)

Come (virtually!) hang out at office hours:

- All office hours will be on the website/Piazza by early next week.
- Will have coverage MTWThF.