Machine Learning: Data Foundations + Algorithms & Applications

Day 1

Objective

- Get to know each other
- Level set
- Establish the overview of the course

Who am I?

Who are you?

Name, Role at Salesforce, and Machine Learning Experience

"A change in perspective is worth 80 IQ points."

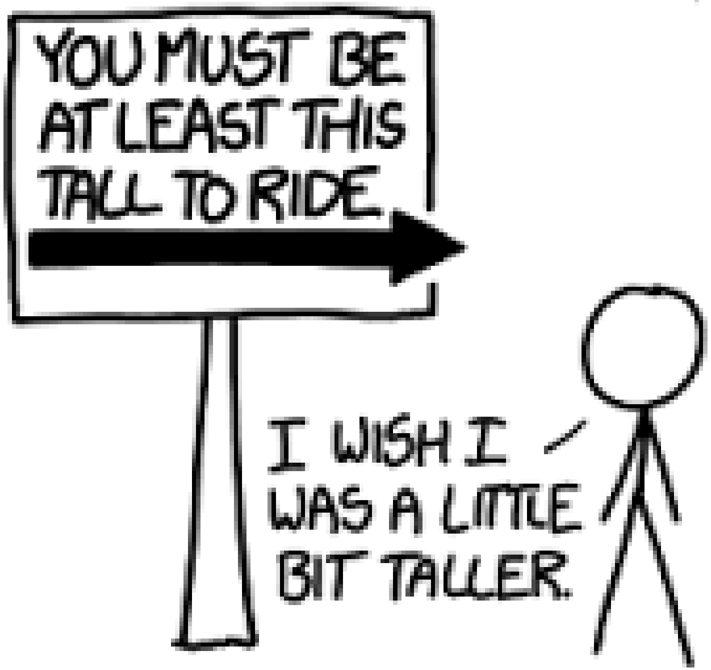
Alan Kay

What is Machine Learning?

- It's the rocketship by which we travel to Planet AI
  - By the way, what's the difference between Machine Learning and AI?
  - Oh, and what's fueling that rocketship?
- automating automation
- getting computers to program themselves
- letting the data do the work
- how is ML different from traditional software development?
  - computers produce output from what input?
  - ML is the reverse: data + output = programs
  - ML is also...

The Rockstar Job of the 21st Century...

...Requires a Lot of Sweeping!



Machine Learning will NOT succeed without…

- understanding our data
- cleaning up our data
- visualizing our data

https://what-if.xkcd.com/77/

Isn't This Someone Else's Problem?

"Poor data quality is enemy number one to the widespread, profitable use of machine learning."

"The quality demands of machine learning are steep, and bad data can rear its ugly head twice — first in the historical data used to train the predictive model and second in the new data used by that model to make future decisions."

"…today, most data fails to meet basic “data are right” standards. Reasons range from data creators not understanding what is expected, to poorly calibrated measurement gear, to overly complex processes, to human error. To compensate, data scientists cleanse the data before training the predictive model. It is time-consuming, tedious work (taking up to 80% of data scientists’ time), and it’s the problem data scientists complain about most."

https://hbr.org/2018/04/if-your-data-is-bad-your-machine-learning-tools-are-useless

A Day in the Life of salesforce

Five Elements of Trust

Security

Availability

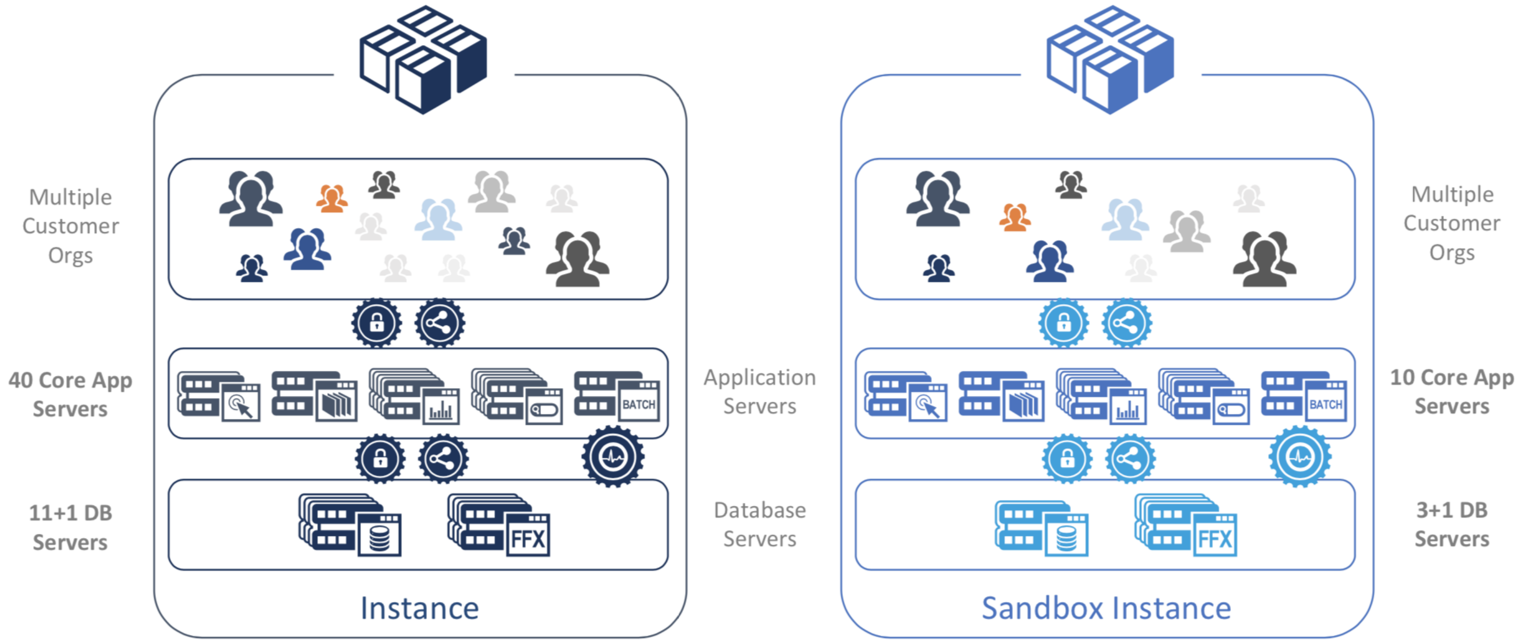
Scalability

Multi-Tenant

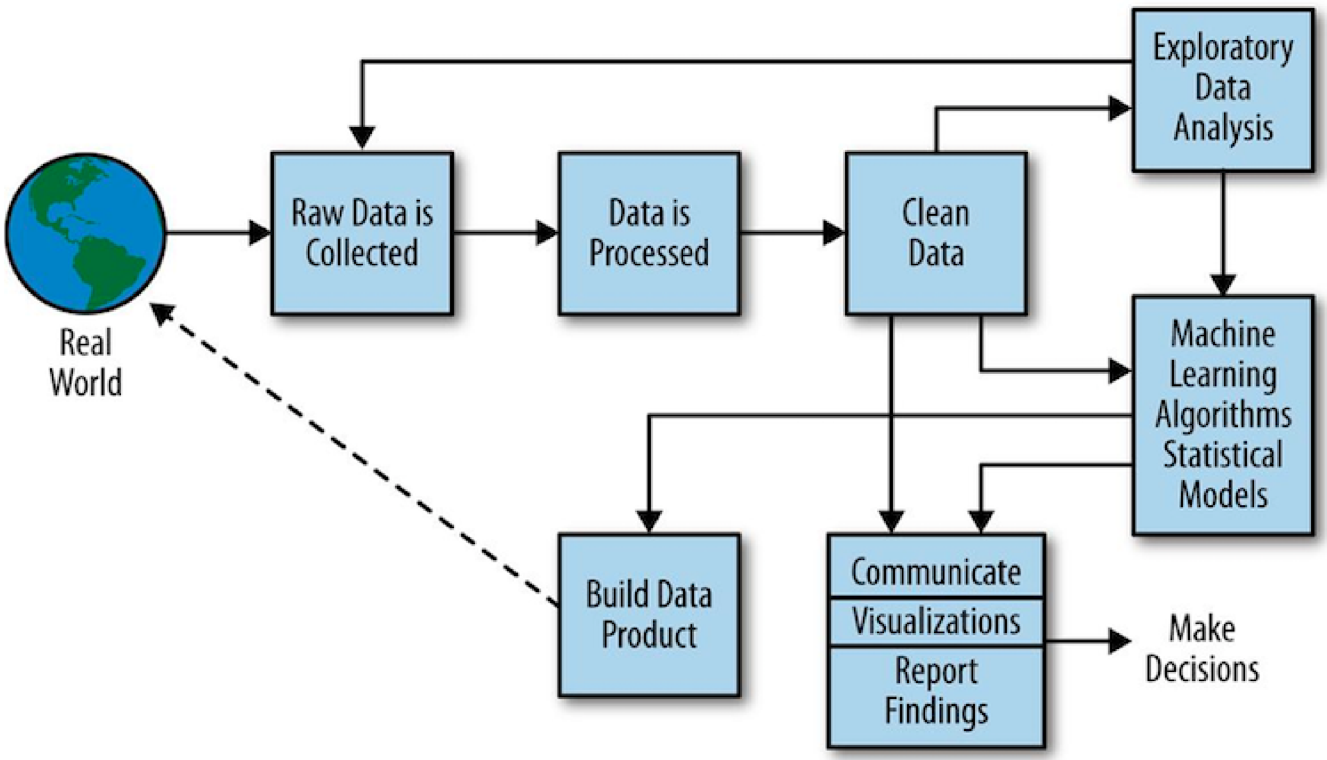
Continuous Innovation

What information from all of the data that we collect from our systems will alert us that things are about to go sideways?

Core and Sandbox Instances



Our Roadmap: Data Science Pipeline



"Doing Data Science"

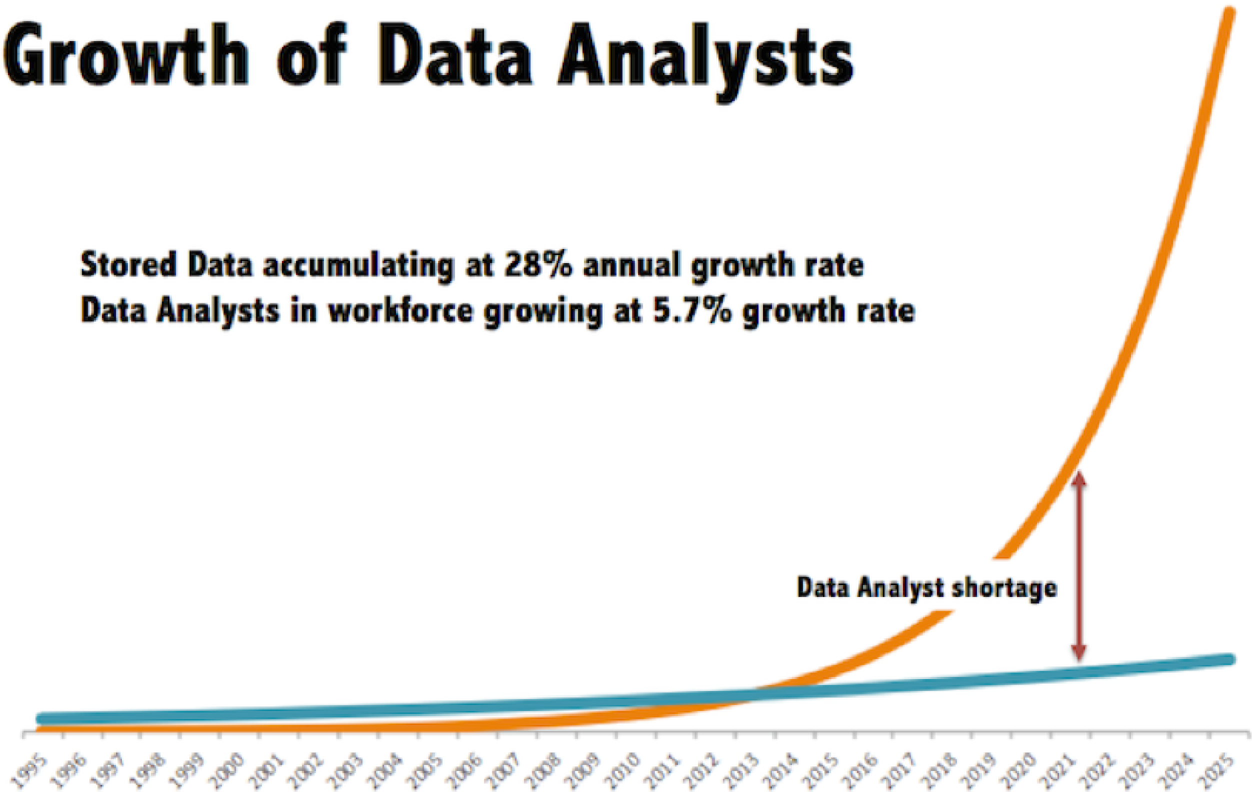
Data and Data Processing

Objective

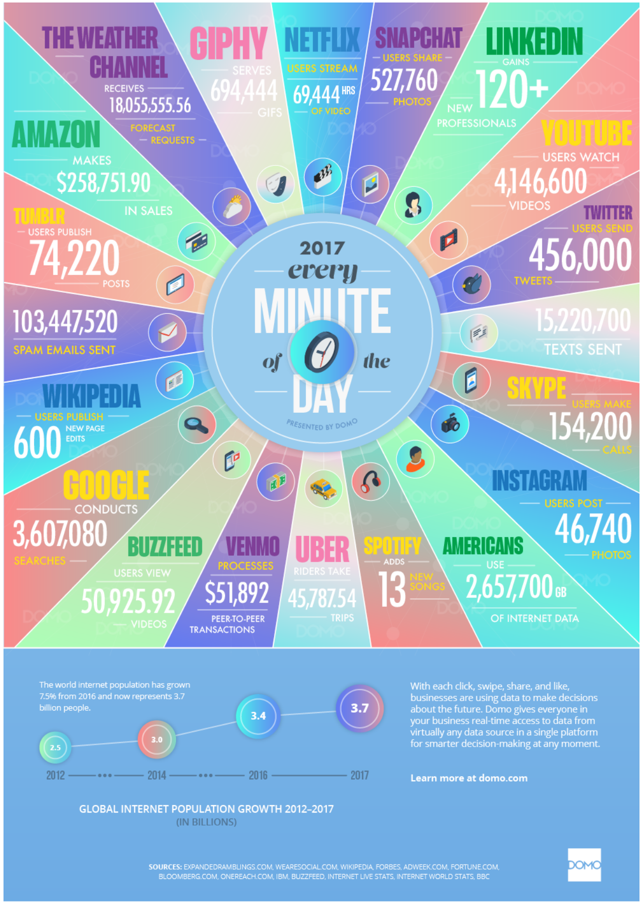
- Connect Data Science with other Information Technology (IT) activities
- Explore how we model our domains
- Understand some of the obstacles we face

Growth of Data vs.
Growth of Data Analysts

Stored Data accumulating at 28% annual growth rate
Data Analysts in workforce growing at 5.7% growth rate



Humans Generate a LOT of Data!



http://bit.ly/2fl2JCT

Demand for Data Scientists Will Soar 28% by 2020

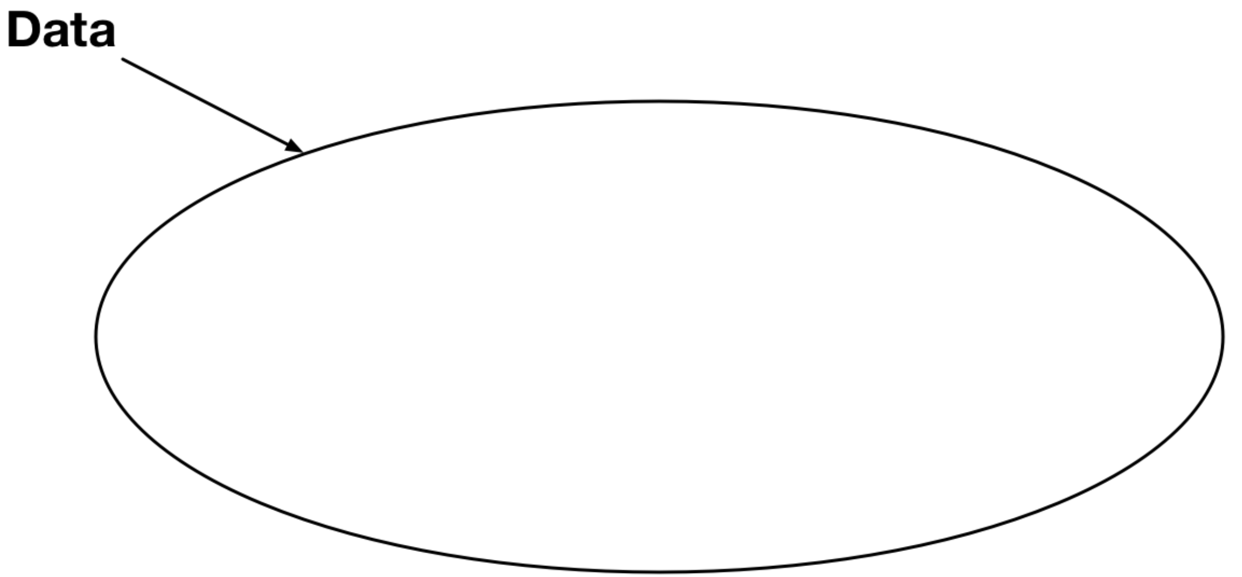
"Annual demand for the fast-growing new roles of data scientist, data developers, and data engineers will reach nearly 700,000 openings by 2020."

"By 2020, the number of jobs for all US data professionals will increase by 364,000 openings to 2,720,000 according to IBM."

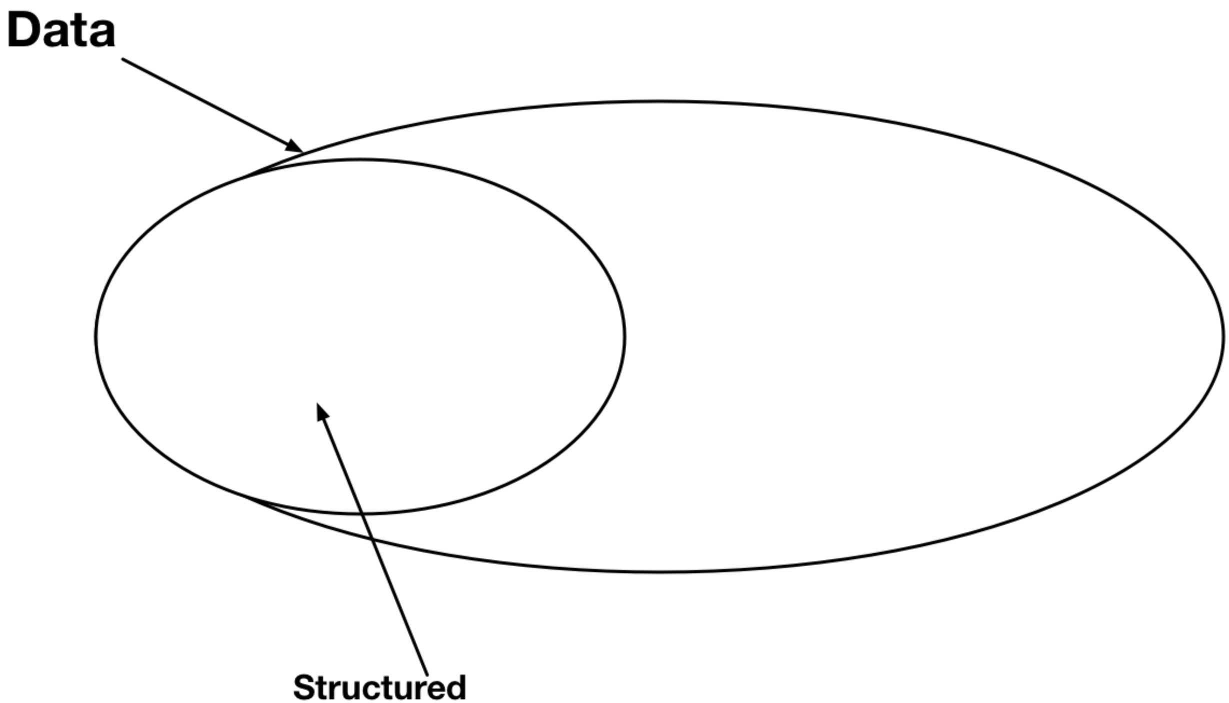
"Machine learning, big data, and data science skills are the most challenging to recruit for and potentially can create the greatest disruption to ongoing product development and go-to-market strategies if not filled."

http://bit.ly/2GOEMiy

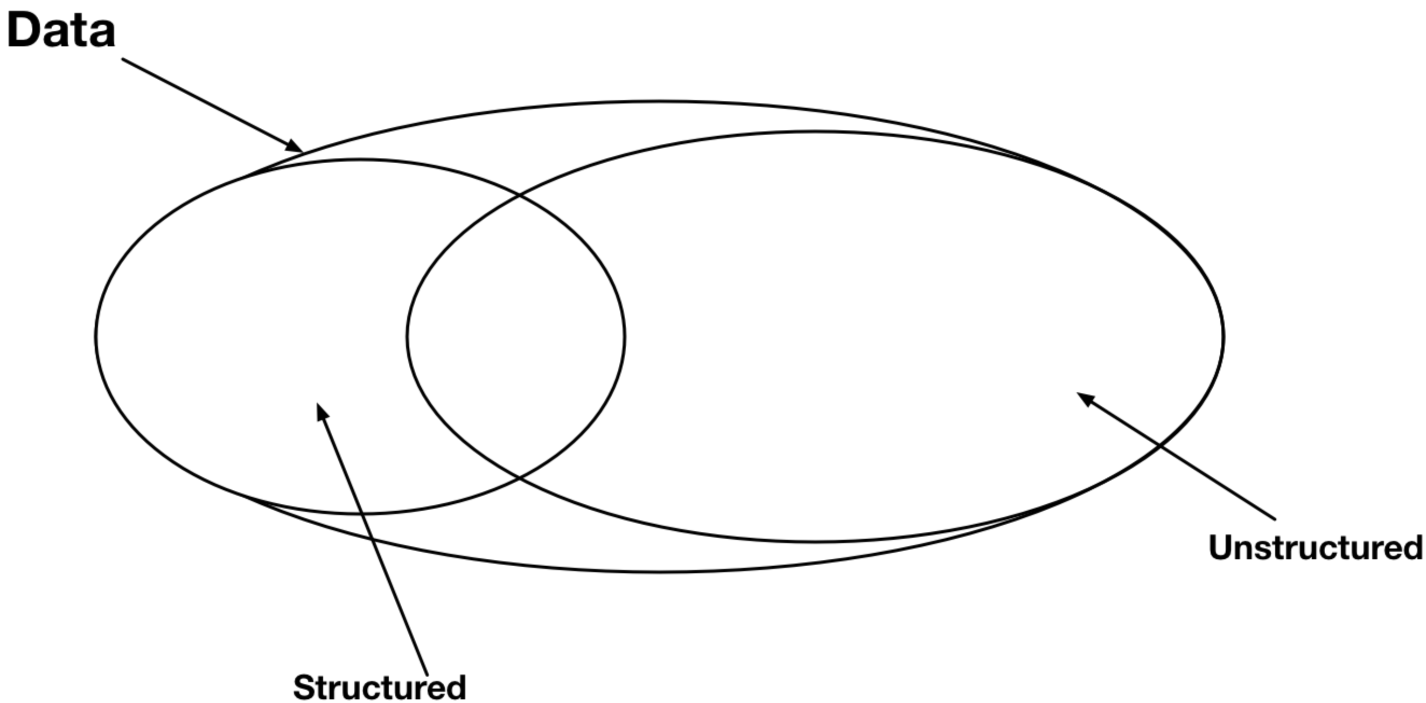
What is Data?



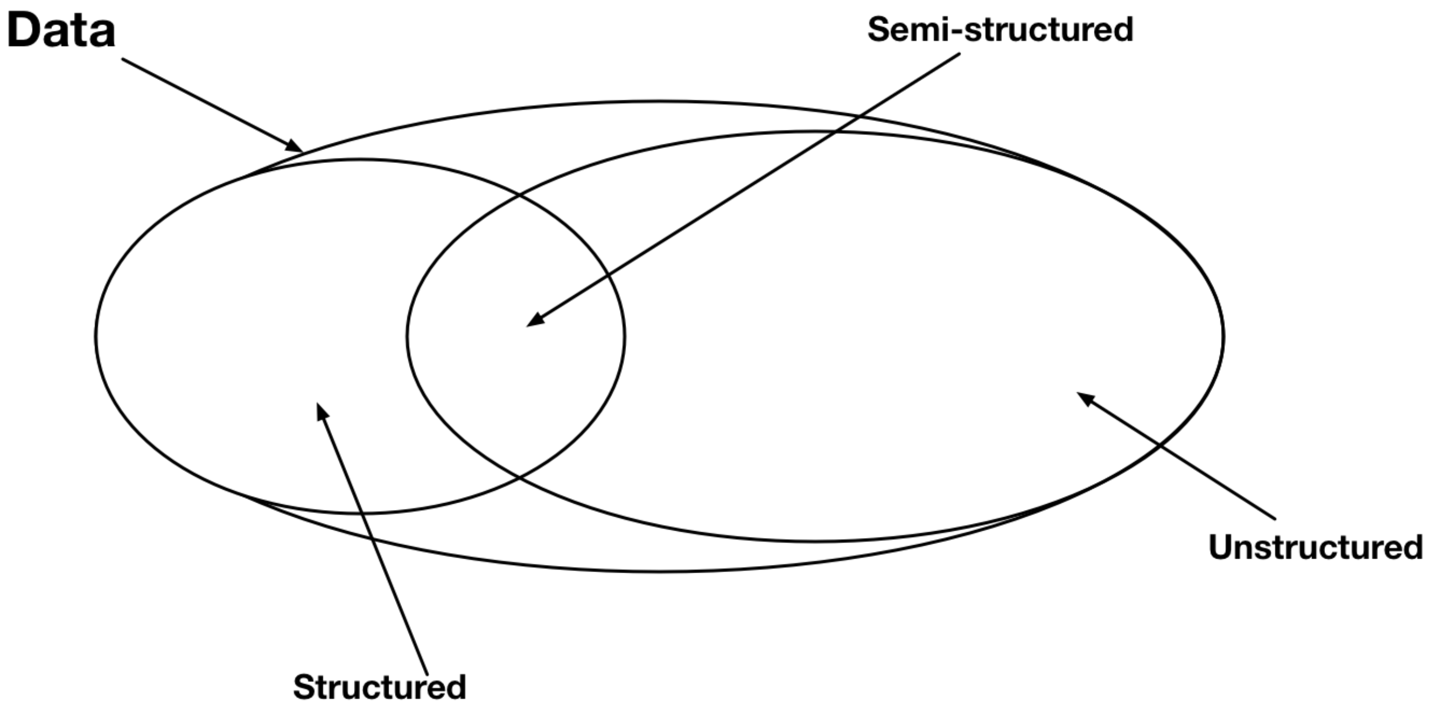
- "Burrito" of data, i.e., all of the things we can detect
  - only a small amount is of a structured nature



- What is structured data?

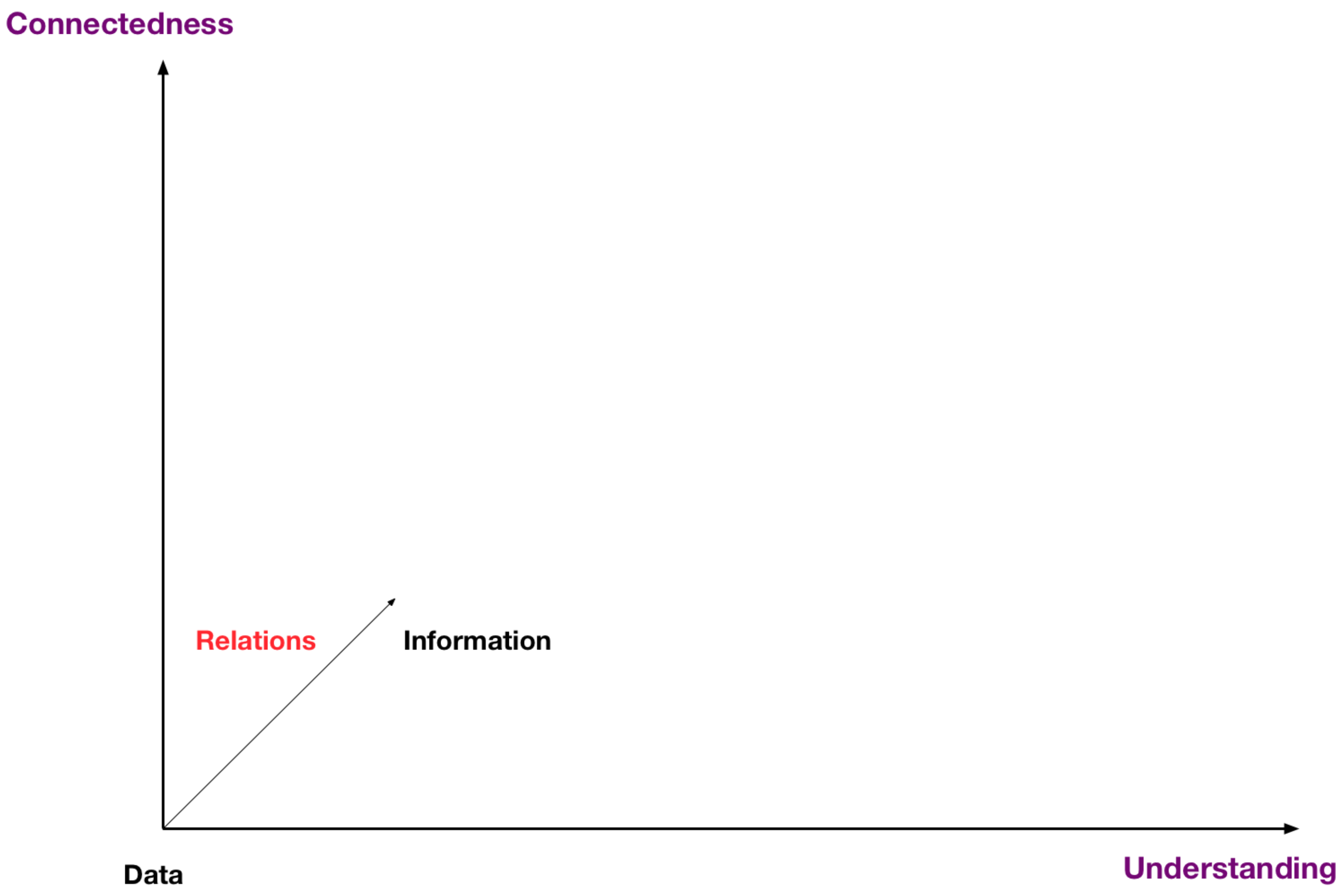


- most data is unstructured, we don't know what's in it
    - emails, videos, documents, wikis, images
- "Show me a picture of a cat on the Internet"

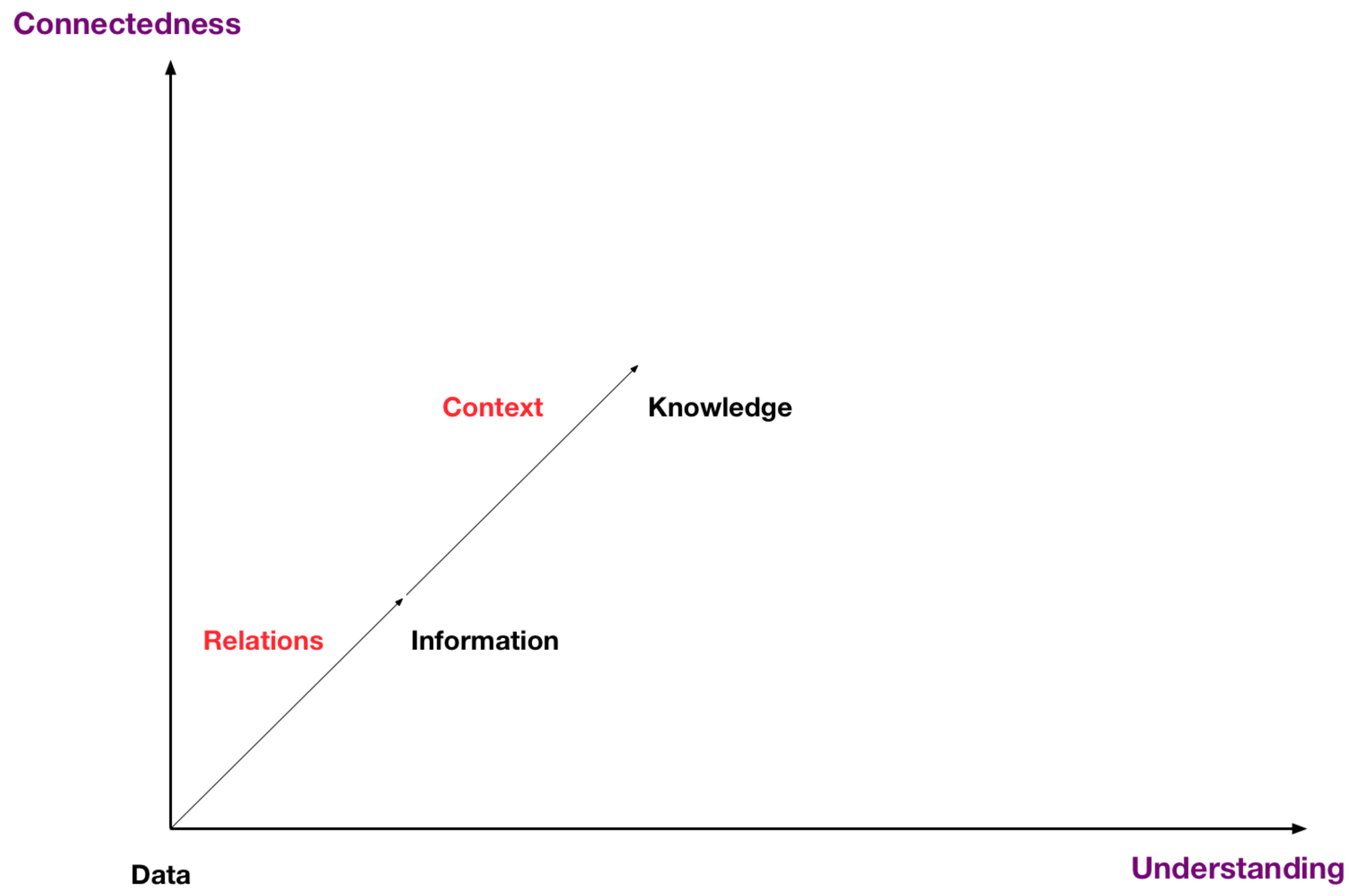


- semi-structured data: how do we put some structure around the unstructured data?
  - encourage people to tell us what it is

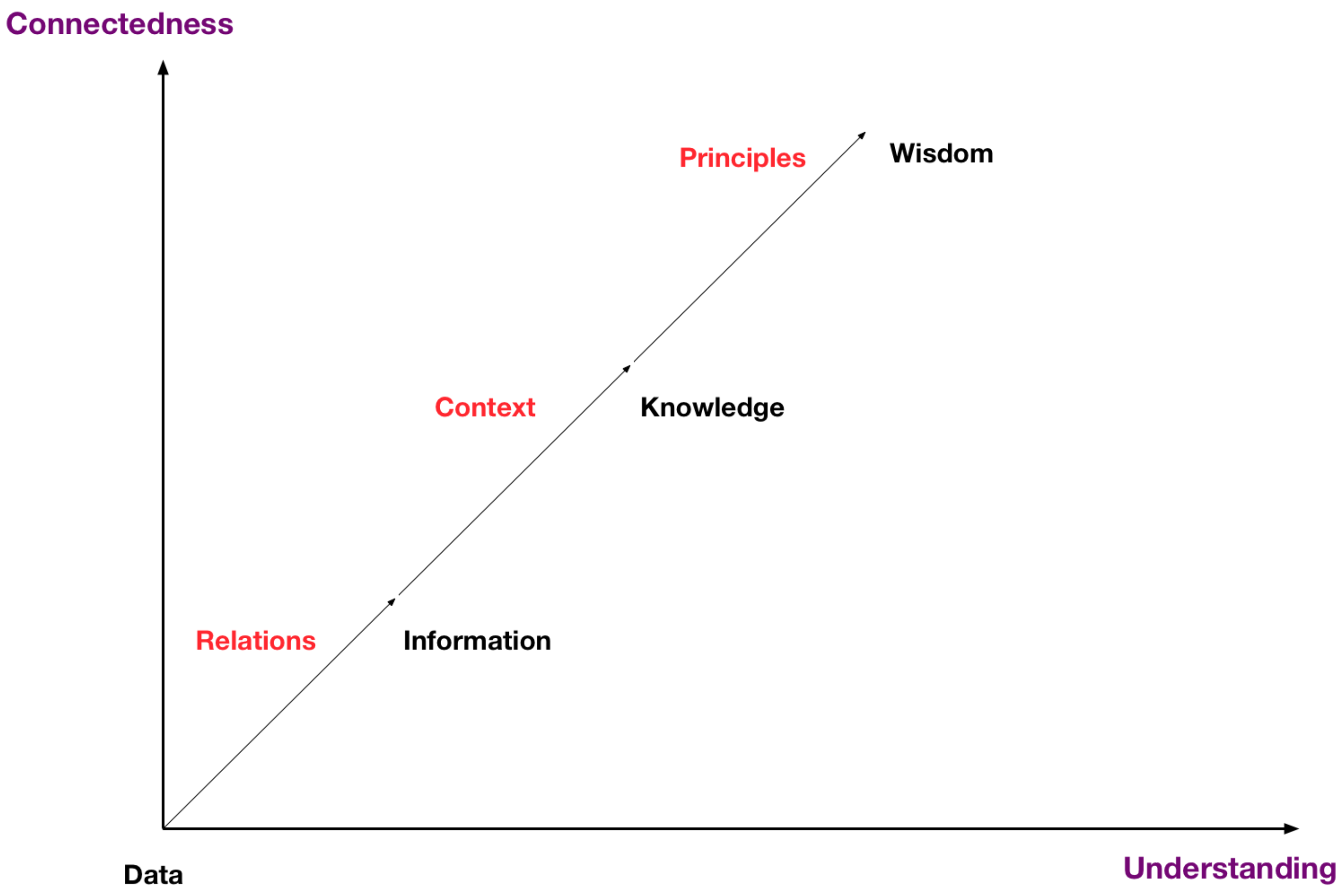
- data is raw
  - i.e., not connected to anything
- data is a necessary but insufficient starting point



- relate data to other data we have which turns the data into information
- a *descriptive* perspective of the world



- shift from a description of the world to a *predictive* nature based on a prioritization of information as a result of context
- what makes information knowledge is the application of context



- patterns that we discover across the knowledge show us principles that result in wisdom

What is the Problem?

Wasting Time

- Data professionals spend 60% of their time "getting to insight"
  - 37% of this time is spent searching for data
  - 36% of this time is spent preparing data
  - Only 27% of that time is actual analysis
- 50% of the time is spent searching for or replicating existing data
- 30-50% of organizations are not where they want to be
- Costs to organizations annually
  - $1.7 million/100 employees in U.S.
  - €1.1 million/100 employees in Europe

https://tinyurl.com/ybwdq34u

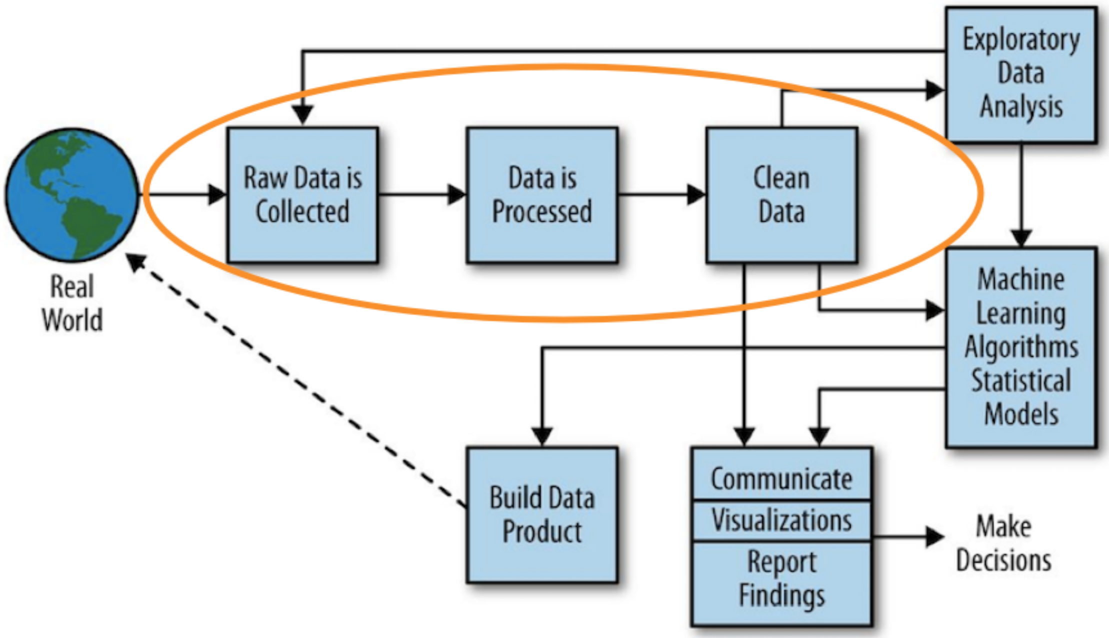
"It is evident that many professionals are not aware of what resources are available within data assets like data lakes, how to access the data, where it came from, or how to glean trusted insights..."

https://tinyurl.com/ybwdq34u

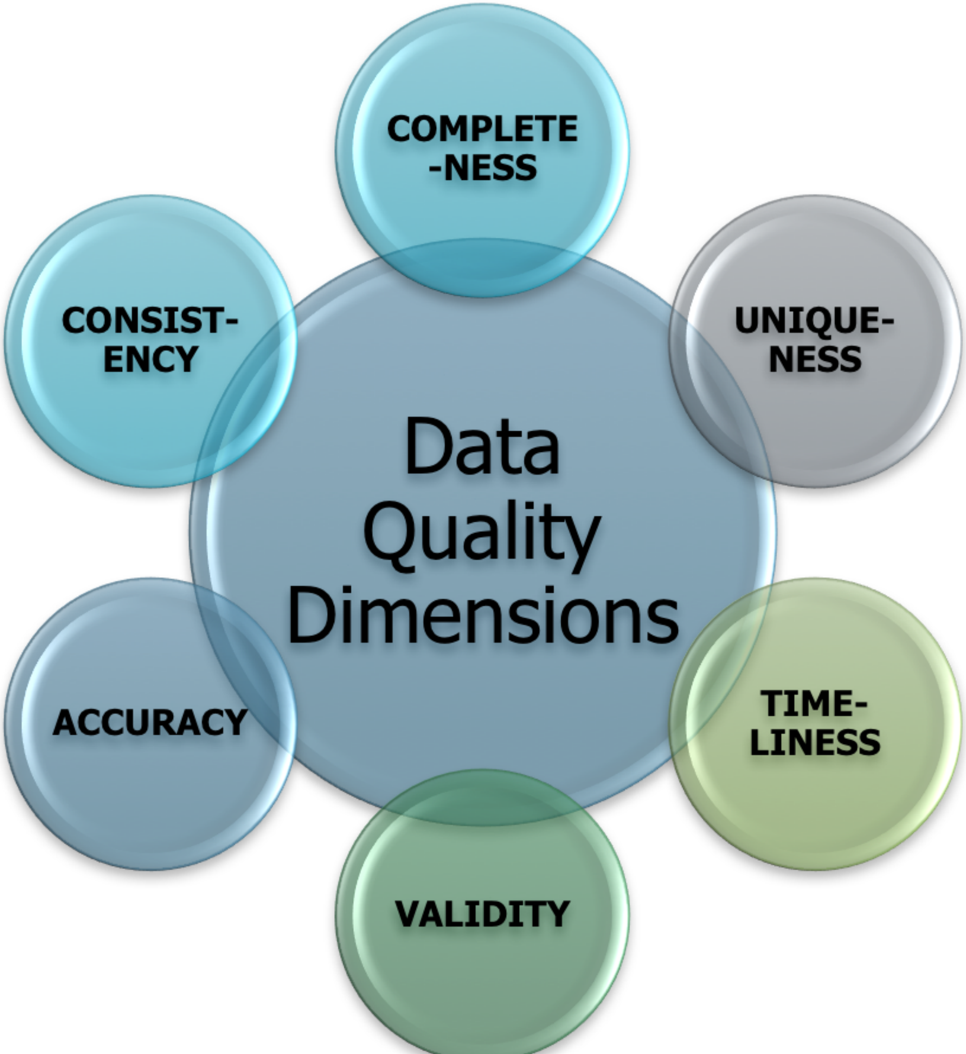
"Unless organizations make changes to their infrastructure now, and close the gaps on data discovery, integrity and cataloging, processes will only become more inefficient as data volume and variety continues to grow."

https://tinyurl.com/ybwdq34u

Data Science Pipeline



"Doing Data Science"


COMPLETE
-NESS
CONSIST-
ENCY
UNIQUE-
NESS
Data
Quality
Dimensions
ACCURACY
TIME-
LINESS
VALIDITY

Metadata

- Data about data
- Types of metadata
  - Business metadata
  - Technical metadata
  - Media metadata
  - Semantic metadata

Business Metadata

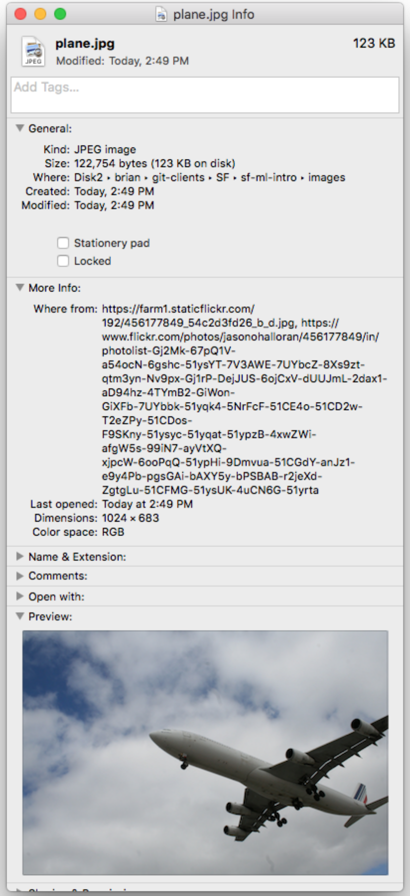
Term	Description
Airline	A company that owns one or more airplanes that are used to transport passengers from one airport to another.
Airport	A transportation destination at which airlines have routes to transport passengers.
Airplane	A vehicle that flies and lands at airports to pick up passengers.
Routes	A connection between two airports that an airline flies its passengers.

Technical Metadata

```
sqlite> .tables
airlines  airports  routes
```

```
sqlite> .schema airports
CREATE TABLE airports (
[index] INTEGER,
  [id] TEXT,
  [name] TEXT,
  [city] TEXT,
  [country] TEXT,
  [code] TEXT,
  [icao] TEXT,
  [latitude] TEXT,
  [longitude] TEXT,
  [altitude] TEXT,
  [offset] TEXT,
  [dst] TEXT,
  [timezone] TEXT
);
CREATE INDEX ix_airports_index ON airports ([index]);
```

Media Metadata



Semantic Metadata

- One approach is JSON-LD (JSON + Linked Data)
- JSON-LD organizes and connects messy and disconnected data

```
{  "@context": "http://schema.org",
   "@type": "Organization",
   "url": "http://united.com",
   "contactPoint":
   [
      {
        "@type": "ContactPoint",
        "telephone": "+1-800-864-8331",
        "contactType": "customer service"
      }
    ]
  }
```

```
@prefix schema: <http://schema.org/> .

[ a schema:Organization ;
  schema:contactPoint  [ a schema:ContactPoint ;
                          schema:contactType  "customer service" ;
                          schema:telephone    "+1-800-864-8331"
                       ] ;
  schema:url <http://united.com>
] .
```

https://json-ld.org/

http://schema.org/Organization

- This is metadata
  - drives the cost of integration down to almost nothing, which lets you find, ingest, and use data
- if you have **`httpie`** installed, you can do **`http http://schema.org/Organization Accept:application/ld+json`**
- when you get a 303 redirect, add **`--follow`** and do it again

Example of Semi-Structured

- Semantic Metadata added to an Unstructured HTML Document

```
<html>
  <head>
    <script type="application/ld+json">
    {"@context": "http://schema.org",
       "@type": "Organization",
       "url": "http://united.com",
       "contactPoint":
       [
          {
            "@type": "ContactPoint",
            "telephone": "+1-800-864-8331",
            "contactType": "customer service"
          }
        ]
      }
    </script>
  </head>
...
</html>
```

As a Developer, Google lets you...

- Add actions to emails
- Mark up events so users can discover them through Google Search results (and other Google products like Google Maps)

Data Sources

Objective

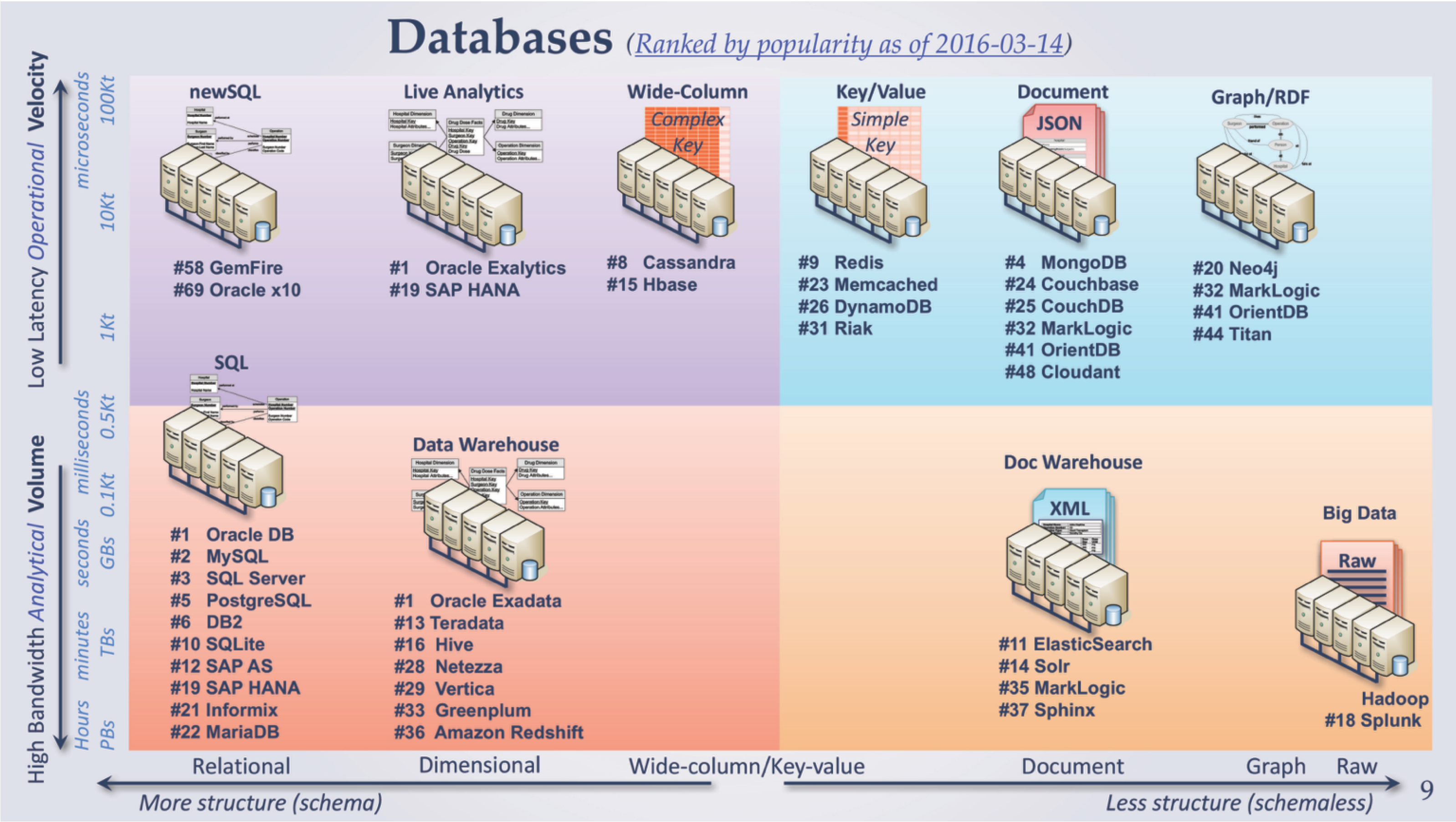
- Consider where data comes from
- Understand some of the integration issues we will face
- Consider our data storage options
- Read data into Python in a convenient format

What did you do Saturday night?

- Can you prove it?
- Consider data "contrails"

Data Modeling Choices

- Relational
- Key-Value (e.g., Redis)
- Column (e.g., Cassandra)
- Document (e..g, CouchDB)
- Graph (e.g., Giraph)

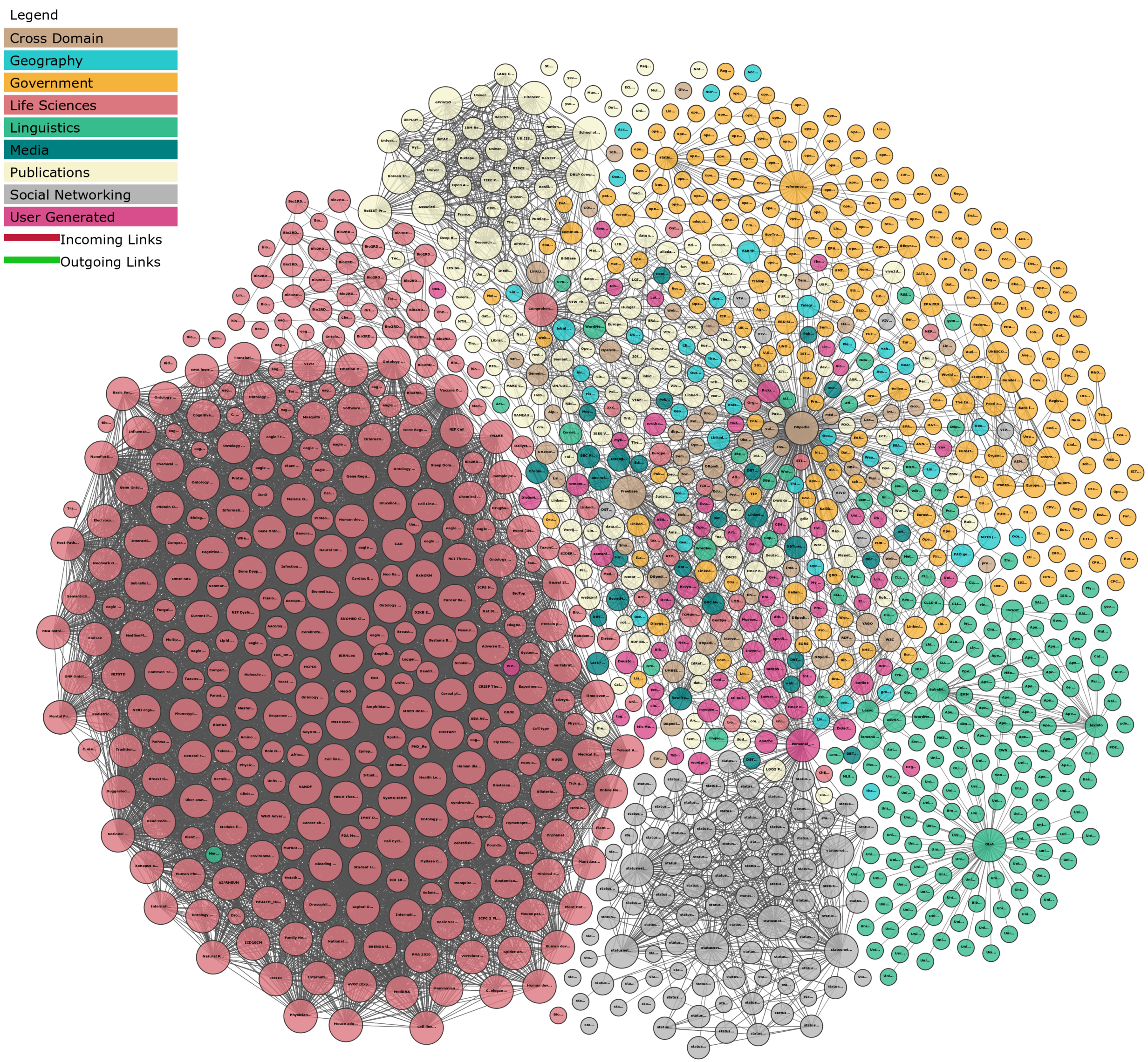

Databases (Ranked by popularity as of 2016-03-14)
Low Latency Operational Velocity
High Bandwidth Analytical Volume
microseconds
100Kt
10Kt
1Kt
milliseconds
0.5Kt
0.1Kt
seconds
GBs
minutes
TBs
Hours
PBs
newSQL
#58 GemFire
#69 Oracle x10
Live Analytics
#1 Oracle Exalytics
#19 SAP HANA
Wide-Column
Complex Key
#8 Cassandra
#15 Hbase
Key/Value
Simple Key
#9 Redis
#23 Memcached
#26 DynamoDB
#31 Riak
Document
JSON
#4 MongoDB
#24 Couchbase
#25 CouchDB
#32 MarkLogic
#41 OrientDB
#48 Cloudant
Graph/RDF
#20 Neo4j
#32 MarkLogic
#41 OrientDB
#44 Titan
SQL
#1 Oracle DB
#2 MySQL
#3 SQL Server
#5 PostgreSQL
#6 DB2
#10 SQLite
#12 SAP AS
#19 SAP HANA
#21 Informix
#22 MariaDB
Data Warehouse
#1 Oracle Exadata
#13 Teradata
#16 Hive
#28 Netezza
#29 Vertica
#33 Greenplum
#36 Amazon Redshift
Doc Warehouse
XML
#11 ElasticSearch
#14 Solr
#35 MarkLogic
#37 Sphinx
Big Data
Raw
Hadoop
#18 Splunk
Relational
Dimensional
Wide-column/Key-value
Document
Graph
Raw
More structure (schema)
Less structure (schemaless)
9

RDBMS

- Established technology, well-understood
- Works best for stable domains

Spreadsheets

- Easy tooling
- Familiarity
- Rapid experimentation
- Customization


Legend
Cross Domain
Geography
Government
Life Sciences
Linguistics
Media
Publications
Social Networking
User Generated
Incoming Links
Outgoing Links

Setting up a Data Science Environment

Let's Install Some Essential Tools!

- Remember that one of our goals is to gain some familiarity with commonly-used data science tools
- We're going to use a tool called Jupyter notebooks to do our hands-on work
- Let's go...

Step 1: Install Anaconda

- Anaconda is an open source distribution of Python and essential data science tools. We use the 5.2 version (Anaconda Navigator 1.8.7)

1. Go to https://www.anaconda.com/download
2. Make sure you download the Python 3 version

3. OSX Install - accept the defaults EXCEPT

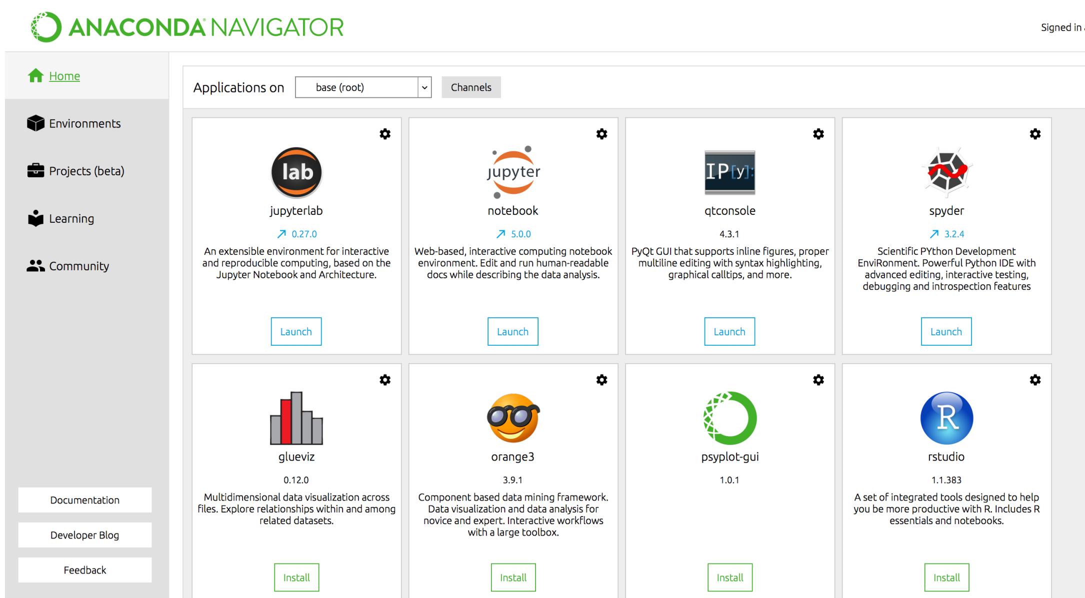
    1. At the **Installation Type** phase:
    2. Click **Change Install Location** and choose **Install for me only**
    3. Choose **Customize** at the **Installation Type** phase, and disable the **Modify PATH** option.
    4. Microsoft VSCode is *not required*

4. Windows Install - accept the defaults EXCEPT

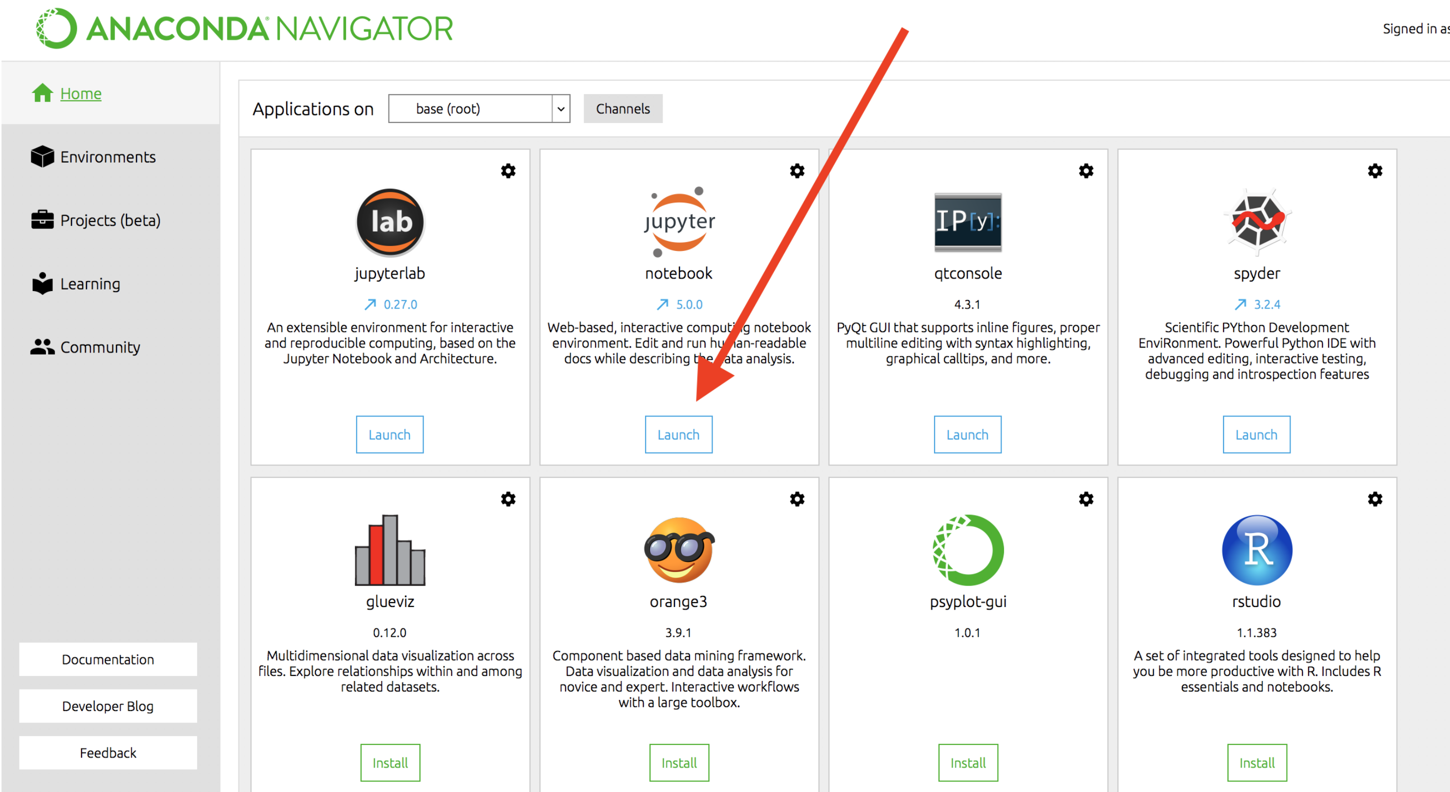
    1. At the *Destination Select* phase pick "Just Me"
    2. Microsoft VSCode is *not required*
    3. Start Menu | Anaconda Prompt
        1. conda install -c conda-forge sparqlwrapper
        2. conda install -c anaconda graphviz

Step 2: Launch the Anaconda Navigator

- You should have an application called Anaconda Navigator in Applications or Start menu



Step 3: Launch Jupyter



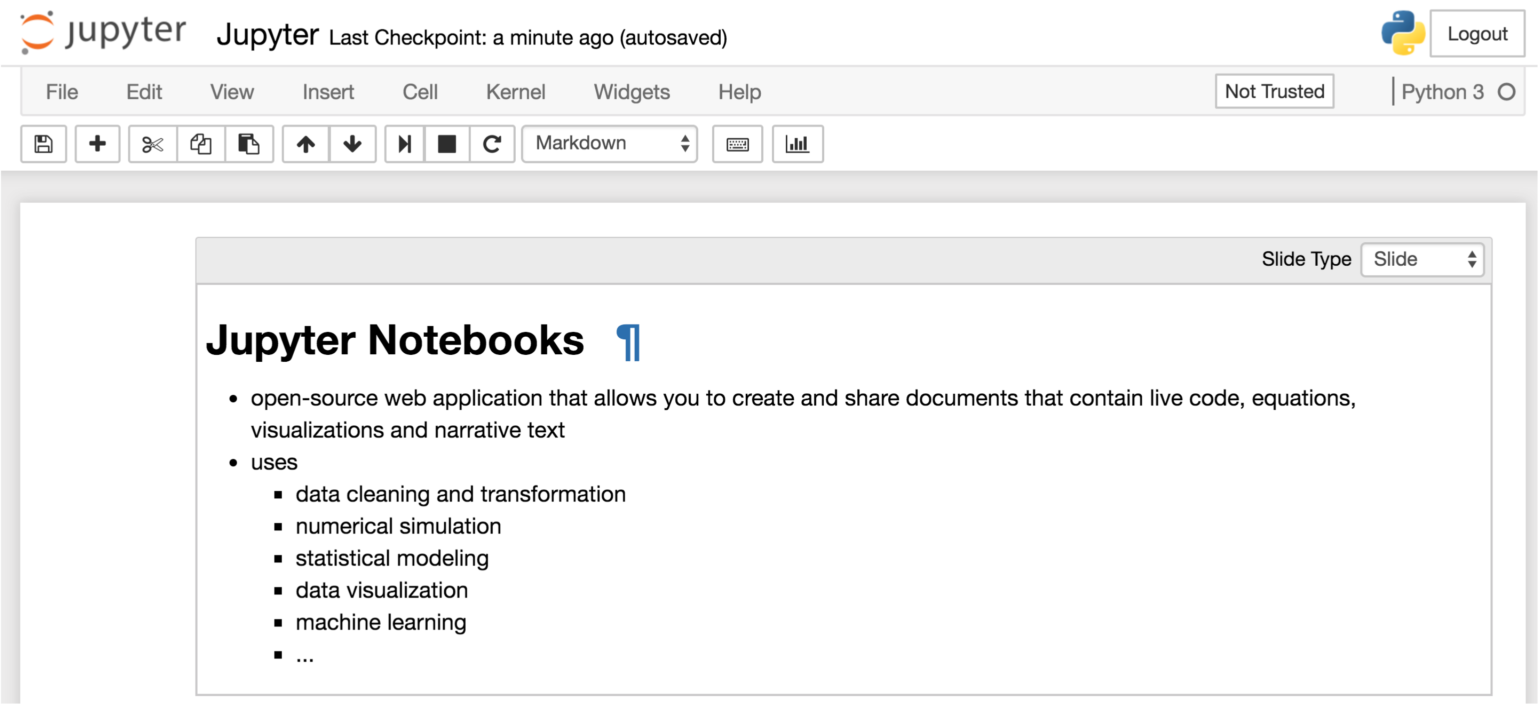
Step 4: Copy/Paste URL into Your Browser

Generally, the prior step will launch a browser window. If that did not happen…

```
[I 16:25:50.469 NotebookApp] JupyterLab alpha preview extension loaded from /Users/dws
/anaconda3/lib/python3.6/site-packages/jupyterlab
JupyterLab v0.27.0
Known labextensions:
[I 16:25:50.472 NotebookApp] Running the core application with no additional extension
s or settings
[I 16:25:50.478 NotebookApp] Serving notebooks from local directory: /Users/dws
[I 16:25:50.478 NotebookApp] 0 active kernels
[I 16:25:50.478 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888
/?token=39a5ad5e45dbd4d6a7e78287693858ba145ea5dbaaf65856
[I 16:25:50.478 NotebookApp] Use Control-C to stop this server and shut down all kerne
ls (twice to skip confirmation).
[C 16:25:50.4[illegible] NotebookApp]

    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://localhost:8888/?token=39a5ad5e45dbd4d6a7e78287693858ba145ea5dbaaf65856
0:97: execution error: "http://localhost:8888/tree?token=f29cdf589b571[illegible]f2553e30ea03
b6d5947fbe8a511170e" doesn't understand the "open location" message. (-1708)
```

Step 5: Navigate to the file **`Demo - Jupyter.ipynb`**



Demo: Pandas

Demo: Pandas

- contraction of "**pan**el **da**ta"
- de facto data analysis toolkit (https://pandas.pydata.org)
- there are screenshots in this presentation to maintain continuity, but let's open the notebook named `Demo - Pandas.ipynb` and go through it together
- when done, click [here]() to skip screenshots

Pandas Series

```
>>> population_dict = {'California': 38332521,
                       'Texas': 26448193,
                       'New York': 19651127,
                       'Florida': 19552860,
                       'Illinois': 12882135}
>>> population = pd.Series(population_dict)
>>> population
California    38332521
Florida       19552860
Illinois      12882135
New York      19651127
Texas         26448193
dtype: int64
```

```
>>> area_dict = {'California': 423967, 'Texas': 695662, 'New York': 141297,
                 'Florida': 170312, 'Illinois': 149995}
>>> area = pd.Series(area_dict)
>>> area
California    423967
Florida       170312
Illinois      149995
New York      141297
Texas         695662
dtype: int64
```

Pandas DataFrame

```
>>> states = pd.DataFrame({'population': population,
                           'area': area})
>>> states
              area  population
California  423967    38332521
Florida     170312    19552860
Illinois    149995    12882135
New York    141297    19651127
Texas       695662    26448193
```

```
>>> states.index
Index(['California', 'Florida', 'Illinois', 'New York', 'Texas'], dtype='object')
```

```
>>> states.columns
Index(['area', 'population'], dtype='object')
```

```
>>> states['area']
California    423967
Florida       170312
Illinois      149995
New York      141297
Texas         695662
Name: area, dtype: int64
```

```
>>> states[area > 200000]
              area  population
California  423967    38332521
Texas       695662    26448193
```

```
>>> states.values
array([[  423967, 38332521],
       [  170312, 19552860],
       [  149995, 12882135],
       [  141297, 19651127],
       [  695662, 26448193]])
```

Reading CSV Files into Pandas

```
In [1]: import pandas as pd
        data = pd.read_csv('data/agg_database_daily.csv')
```

```
In [2]: data.head()
```

	20161031	WAS	SP3	na17	12.881275	87	47.8	1329935.5	2.3832552	24.30115	59.996094	46.06433	13.931762	8.10202	51.894073	59.996
0	20161025	CHI	SP2	cs15	65.872505	69.13125	30.184166	1189676.4	1.6051193	14.272053	\N	\N	\N	\N	\N	
1	20161031	CHI	SP4	cs40	30.010305	86	22.05	863386.7	1.1656047	9.44478	106.87152	87.29087	19.580648	15.005844	91.86568	106.8
2	20161031	WAS	SP3	na23	6.484551	72	35.066666	1139595.2	1.7977492	13.583612	59.996094	32.447983	27.54811	20.739609	39.256485	59.99
3	20161031	PHX	SP1	na45	4.7593827	50	17.758333	744373.94	1.1349427	9.120685	67.4989	26.249228	41.249672	30.199581	37.29932	67.
4	20161031	CHI	SP1	gs0	10.942432	37	8.622222	1118318	1.1859665	4.268443	362.25	33.72686	328.52313	240.68625	121.56374	36

```
In  [3]: data.shape
Out [3]: (55285, 20)
```

Specifying No Headers

```
In [3]: import pandas as pd
        data = pd.read_csv('data/agg_database_daily.csv', header=None)
```

```
In [4]: data.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
0	20161031	WAS	SP3	na17	12.881275	87	47.8	1329935.5	2.3832552	24.30115	59.996094	46.06433	13.931762	8.10202	51.894073	59.9960
1	20161025	CHI	SP2	cs15	65.872505	69.13125	30.184166	1189676.4	1.6051193	14.272053	\N	\N	\N	\N	\N	
2	20161031	CHI	SP4	cs40	30.010305	86	22.05	863386.7	1.1656047	9.44478	106.87152	87.29087	19.580648	15.005844	91.86568	106.871
3	20161031	WAS	SP3	na23	6.484551	72	35.066666	1139595.2	1.7977492	13.583612	59.996094	32.447983	27.54811	20.739609	39.256485	59.9960
4	20161031	PHX	SP1	na45	4.7593827	50	17.758333	744373.94	1.1349427	9.120685	67.4989	26.249228	41.249672	30.199581	37.29932	67.49

```
In  [5]: data.shape
Out [5]: (55286, 20)
```

Specifying Column Names

```
In  [6]: data.columns = ['date_key','datacenter','superpod','pod','max_redo_size',
  'max_active_sessions','max_db_cpu_user','max_peak_buffer','avg_db_cpu_system',
  'avg_db_cpu_user','total_db_size_in_tb','used_db_space_in_tb',
  'free_db_space_in_tb', 'asm_free_db_space_in_tb','asm_used_db_space_in_tb',
  'asm_total_db_size_in_tb', 'last_modified','asm_used_db_space_perc',
  'oem_cpu_util','oem_read_io_latency']
```

```
In  [7]: data.head()
```

	date_key	datacenter	superpod	pod	max_redo_size	max_active_sessions	max_db_cpu_user	max_peak_buffer	avg_db_cpu_system	avg_db_cpu_user	total_
0	20161031	WAS	SP3	na17	12.881275	87	47.8	1329935.5	2.3832552	24.30115	
1	20161025	CHI	SP2	cs15	65.872505	69.13125	30.184166	1189676.4	1.6051193	14.272053	
2	20161031	CHI	SP4	cs40	30.010305	86	22.05	863386.7	1.1656047	9.44478	
3	20161031	WAS	SP3	na23	6.484551	72	35.066666	1139595.2	1.7977492	13.583612	
4	20161031	PHX	SP1	na45	4.7593827	50	17.758333	744373.94	1.1349427	9.120685	

Missing Values

```
In  [7]: data['max_active_sessions']
Out [7]: 0                  87
         1            69.13125
         2                  86
         3                  72
         4                  50
         5                  37
         6                  38
         7                  \N
         8                  89
         9                  94
         ...
```

Handling Missing Values

```
In  [8]: data = pd.read_csv('data/agg_database_daily.csv', header=None,
                            na_values=[r'\N'])
         data.columns = ['date_key','datacenter','superpod','pod','max_redo_size',
'max_active_sessions','max_db_cpu_user','max_peak_buffer','avg_db_cpu_system',
'avg_db_cpu_user','total_db_size_in_tb','used_db_space_in_tb',
'free_db_space_in_tb','asm_free_db_space_in_tb','asm_used_db_space_in_tb',
'asm_total_db_size_in_tb','last_modified','asm_used_db_space_perc',
'oem_cpu_util','oem_read_io_latency']
```

```
In  [7]: data['max_active_sessions']
Out [7]: 0              87
         1        69.13125
         2              86
         3              72
         4              50
         5              37
         6              38
         7             NaN
         8              89
         9              94
         ...
```

Dropping Missing Values

```
In  [9]: data['max_active_sessions'].dropna()
Out [9]: 0          87.000000
         1          69.131250
         2          86.000000
         3          72.000000
         4          50.000000
         5          37.000000
         6          38.000000
         8          89.000000
         9          94.000000
         10         35.542683
         11         57.950000
         12         44.000000
         14         46.000000
         15         47.000000
         16        107.000000
         17         83.000000
         18         77.470290
         19         89.000000
         ...
```

Reading SQLite into Pandas

```
>>> import pandas as pd
>>> import sqlite3
>>> conn = sqlite3.connect("data/flights.db")
>>> data = pd.read_sql_query("select * from airlines", conn)
>>> data.shape
(6048, 9)
```

```
>>> data.head()
   index id                                          name alias  iata  icao  \
0      0  1                                Private flight    \N     -  None
1      1  2                                   135 Airways    \N  None   GNL
2      2  3                                 1Time Airline    \N    1T   RNX
3      3  4  2 Sqn No 1 Elementary Flying Training School    \N  None   WYT
4      4  5                               213 Flight Unit    \N  None   TFU

  callsign         country active
0     None            None      Y
1  GENERAL   United States      N
2  NEXTIME    South Africa      Y
3     None  United Kingdom      N
4     None          Russia      N
```

```
>>> conn.close()
```

Exercise: Pandas

(open the notebook named `Exercise 1 - Pandas.ipynb`)

Exercise: Reading TTL Data

(open the notebook named `Exercise 2 - Reading TTL Data.ipynb`)

Recap: Data Sources Recap

- What are trying do?
  - we're reading data from relational databases, spreadsheets, etc.
- Why would we choose to put data into these different types of storage?
  - differently-shaped data
  - high-volume data
  - high-velocity data
  - in other words, there are compelling business and technical reasons
- We also talked about *linked data*
  - e.g., DBpedia is a dataset drawn from the editorial process of Wikipedia
  - for anything in Wikipedia, such as a city, there is structural information (e.g., when city was founded, geographic location, etc.)
  - the entities in DBpedia are generated from the pages of Wikipedia
  - they're resolvable through URLs, and the relationships themselves are also resolvable
    - node ids/links are like global, disambiguable, resolvable DB keys
    - relationship ids/links are like global, disambiguable, resolvable column names

Demo: Reading Linked Data

Demo: Reading Linked Data

- there are screenshots in this presentation to maintain continuity, but let's open the notebook named `Demo - Reading Linked Data.ipynb` and go through it together
- when done, click here to skip screenshots

```python
import pandas as pd
import json
from SPARQLWrapper import SPARQLWrapper, JSON
```

```python
def get_sparql_dataframe(service, query):
    """
    Helper function to convert SPARQL results into a Pandas data frame.
    """
    sparql = SPARQLWrapper(service)
    sparql.setQuery(query)
    sparql.setReturnFormat(JSON)
    result = sparql.query()

    processed_results = json.load(result.response)
    cols = processed_results['head']['vars']

    out = []
    for row in processed_results['results']['bindings']:
        item = []
        for c in cols:
            item.append(row.get(c, {}).get('value'))
        out.append(item)

    return pd.DataFrame(out, columns=cols)
```

```python
wds = "https://query.wikidata.org/sparql"
```

```
# This SPARQL query will be sent to the SPARQL endpoint defined in the previous
# step. It mixes three vocabularies that each have their own definitions but is
# ultimately a selection from the Wikidata graph. We're looking for distinct rows
# of individuals who have an orcid (https://orcid.org) and any English description
# and labels we might also have about them. We're matching a pattern in the graph
# for any node that is connected to other nodes with these relationships.

# To understand what __`wdt:P496 means`__, expand it into its full URL by applying
# the prefix for wdt and then issuing an HTTP request to
# http://www.wikidata.org/prop/direct/P496.

# For more info, consult Bob DuCharme Learning SPARQL (2nd Edition)

rq = """
PREFIX bd: <http://www.bigdata.com/rdf#>
PREFIX wikibase: <http://wikiba.se/ontology#>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>

select distinct
   ?item
   ?itemLabel
   ?orcid
   ?description
WHERE {
  ?item wdt:P496 ?orcid
  OPTIONAL {
      ?item schema:description ?description filter (lang(?description) = "en")
  }
  SERVICE wikibase:label {
        bd:serviceParam wikibase:language "en" .
  }
}
"""
```

Exercise: Querying Linked Data

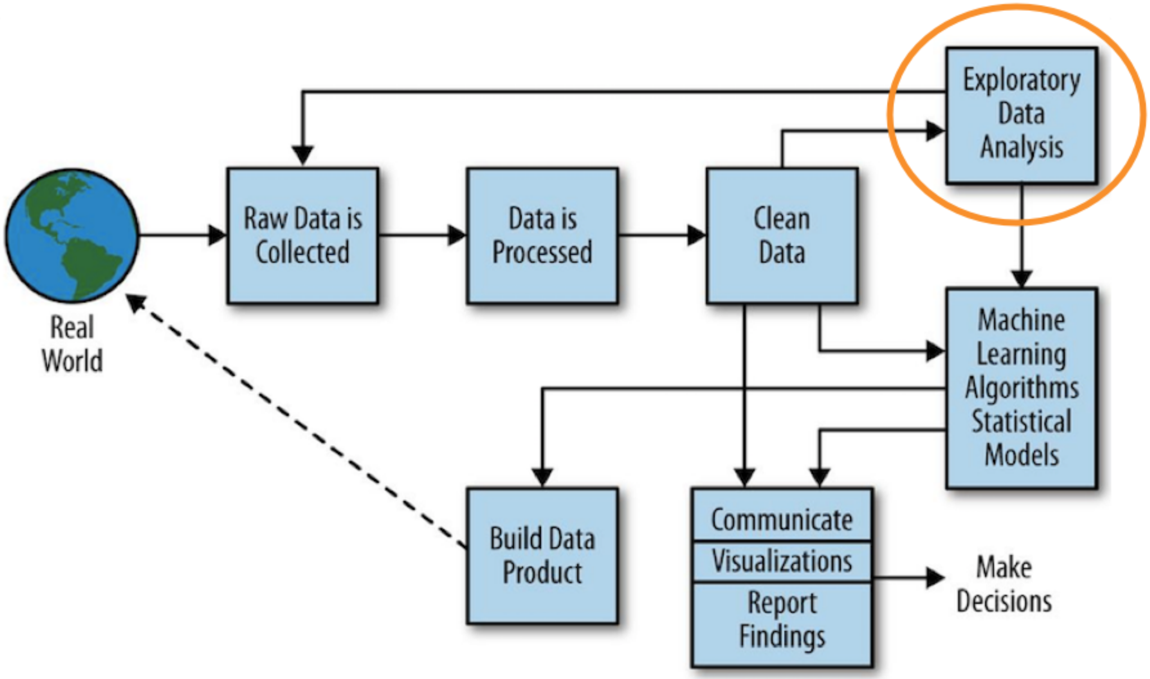
(open the notebook named `Exercise 3 - Querying Linked Data.ipynb`)

Data-Driven

Objective

- Consider the mental shift in letting the data drive our activities
- Explore the discipline needed to understand our data
- Friendly statistics intro/re-familiarization
- Understand some of the challenges that emerge with being data driven

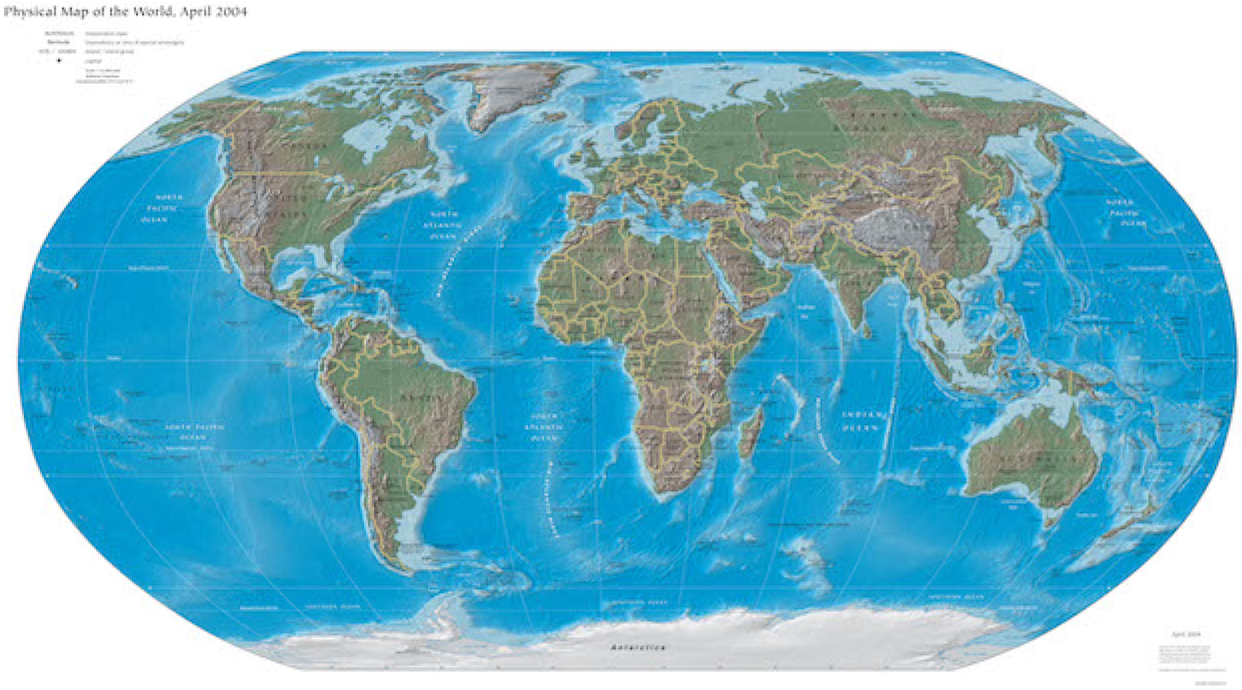
Data Science Pipeline



"Doing Data Science"

Which university degree led to the highest average starting salary at the University of North Carolina in the 1980's?

Geography!



Intro to (or Review of) Statistics

Can you explain these terms?

- Mean
- Variance
- Interquartile range
- Skew

Demo: Let's Make Some Data

- there are screenshots in this presentation to maintain continuity, but let's open the notebook named **`Demo - Let's Make Some Data.ipynb`** and go through it together
- when done, click here to skip screen shots

```
>>> # Generate a list of numbers 20 to 59
>>> ages = range(20, 60)
```

```
>>> # Select 100 numbers at random from the set of numbers 20 to 59
>>> random_ages = [random.choice(ages) for _ in range(100)]
```

```
>>> random_ages
[27, 21, 20, 22, 21, 27, 25, 52, 56, 43, 33, 28, 31, 30, 41, 29, 38, 52,
 48, 24, 50, 31, 29, 56, 45, 54, 37, 33, 22, 33, 23, 40, 37, 23, 28, 38,
 41, 34, 35, 40, 25, 57, 32, 48, 56, 54, 39, 25, 57, 24, 26, 52, 26, 35,
 57, 39, 26, 32, 50, 52, 54, 53, 30, 53, 56, 25, 43, 38, 53, 46, 51, 51,
 35, 38, 20, 52, 47, 53, 45, 27, 22, 26, 25, 47, 36, 52, 54, 58, 31, 42,
 46, 46, 43, 33, 35, 48, 49, 55, 36, 36]
```

```
>>> max(random_ages)
58
```

```
>>> min(random_ages)
20
```

Range

```
>>> def calc_range(x):
...     return max(x) - min(x)
...
```

```
>>> calc_range(random_ages)
38
```

```
>>> nums = [10, 10, 100, 100]
>>> calc_range(nums)
90
```

```
>>> nums = [10, 50, 50, 50, 50, 100]
>>> calc_range(nums)
90
```

```
# Peak to Peak - returns the range, the min is a "peak" and the max is a "peak"
>>> numpy.ptp(random_ages)
38
```

Mean

```
>>> def mean(x):
...     return sum(x) / len(x)
...
```

```
>>> mean(random_ages)
38.99
```

```
>>> import numpy
>>> numpy.mean(random_ages)
38.990000000000002
```

Median

```
def median(x) :
  n = len(x)
  sorted_x = sorted(x)
  mid = n // 2

  if n % 2 == 0:
      return (sorted_x[mid - 1] + sorted_x[mid]) / 2
  else:
      return (sorted_x[mid])
```

```
>>> median(random_ages)
38
```

```
>>> numpy.median(random_ages)
38.0
```

Percentile

```
>>> numpy.percentile(random_ages, 33)
32.0
```

```
>>> numpy.percentile(random_ages, 80)
52.0
```

```
>>> numpy.percentile(random_ages, 50)
38.0
```

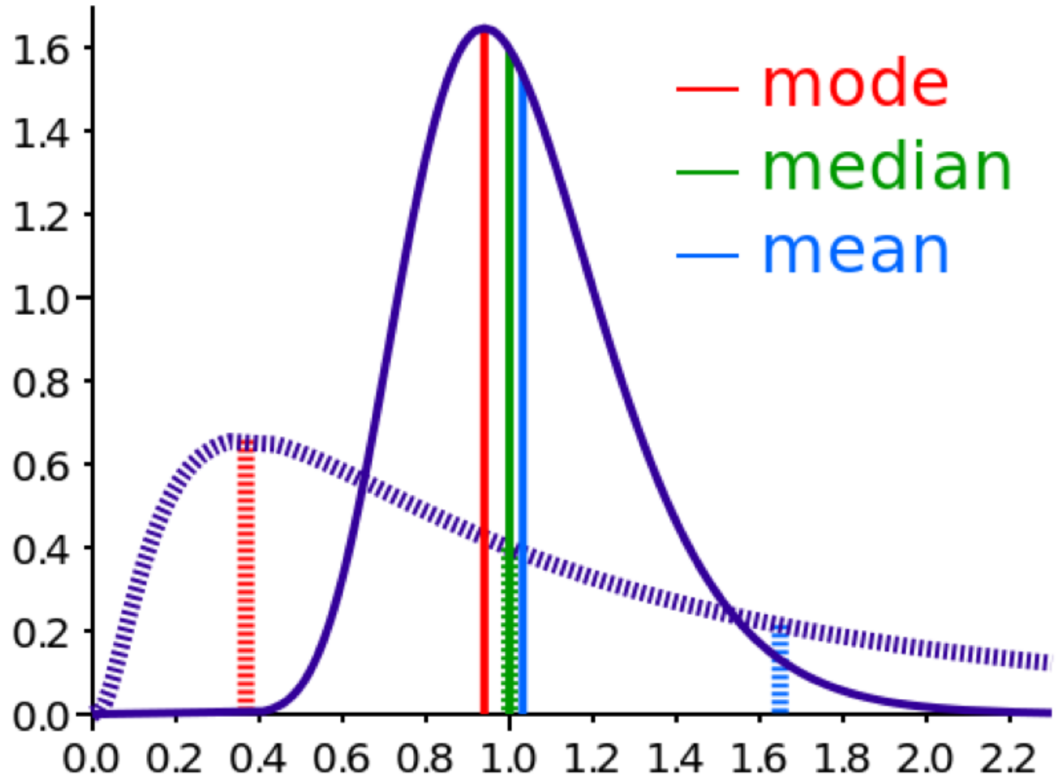
Interquartile Range (IQR)

$$IQR = Q_3 - Q_1$$

```
>>> from scipy import stats
>>> stats.iqr(random_ages)
21.5
```

Mode

```
>>> from scipy import stats
>>> stats.mode(random_ages)
ModeResult(mode=array([52]), count=array([6]))
```



- consider the spread of data in two hypothetical distributions
- how can we identify/quantify different spreads?
- focus on the **mean**, **median**, and **mode**

Variance

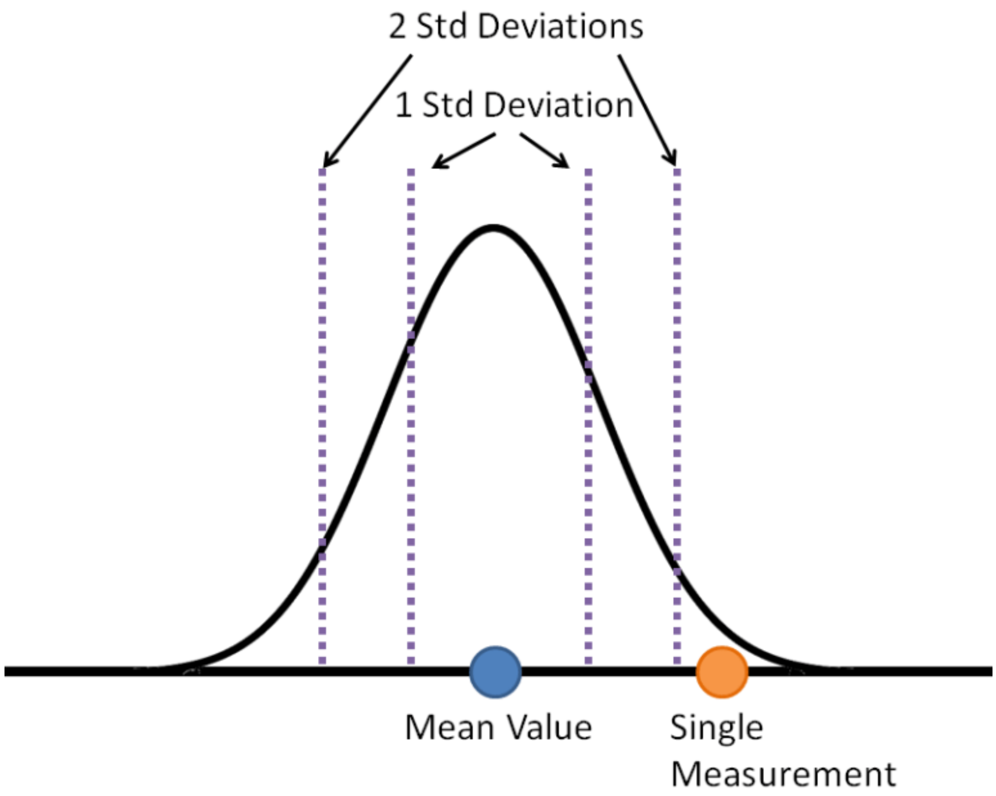
$$Var(X) = \frac{1}{n}\sum_{i=1}^{n}(a_i - \bar{x})^2$$

```
>>> numpy.var(random_ages)
131.82989999999998
```

Standard Deviation

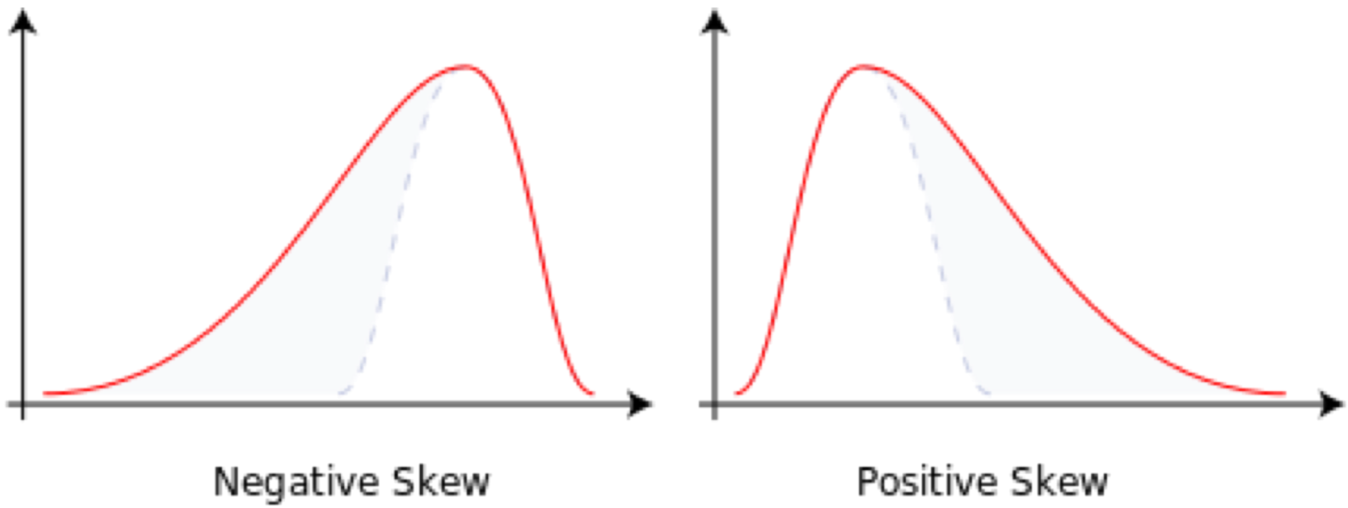
$$\sqrt{Var(X)}$$

```
>>> numpy.std(random_ages)
11.481720254386969
```



http://www.fairlynerdy.com/intuitive-guide-statistical-significance/

Skewness




```
>>> stats.skew(random_ages)
0.037925216234465986
```

Descriptive Statistics

- Descriptive Statistics *describe* our data in various ways
- Measures of center ("The Three M's")
  - mean
  - median
  - mode
- Measures of dispersion
  - range
  - percentiles/quartiles
  - IQR
  - variance
  - standard deviation
  - skewness

Demo: Summary Statistics of TTL Data

Demo: Summary Statistics of TTL Data

- let's open the notebook named **`Demo - Summary Statistics of TTL Data.ipynb`** and go through it together

Exercise: Summary Statistics of TTL Data

(open the notebook named **`Exercise 4 - Summary Statistics of TTL Data.ipynb`**)

Understanding
Relationships between Variables

Covariance

- Measure of joint variability
  - i.e., how does one variable change as the other changes?
  - this draws our attention to a connection
- Non-zero covariance implies?
  - there is some connection
- Zero covariance implies?
  - independence

"Correlation is not causation."

Anyone who has ever taken statistics

"Empirically observed covariation is a necessary but not sufficient condition for causality."

Edward Tufte

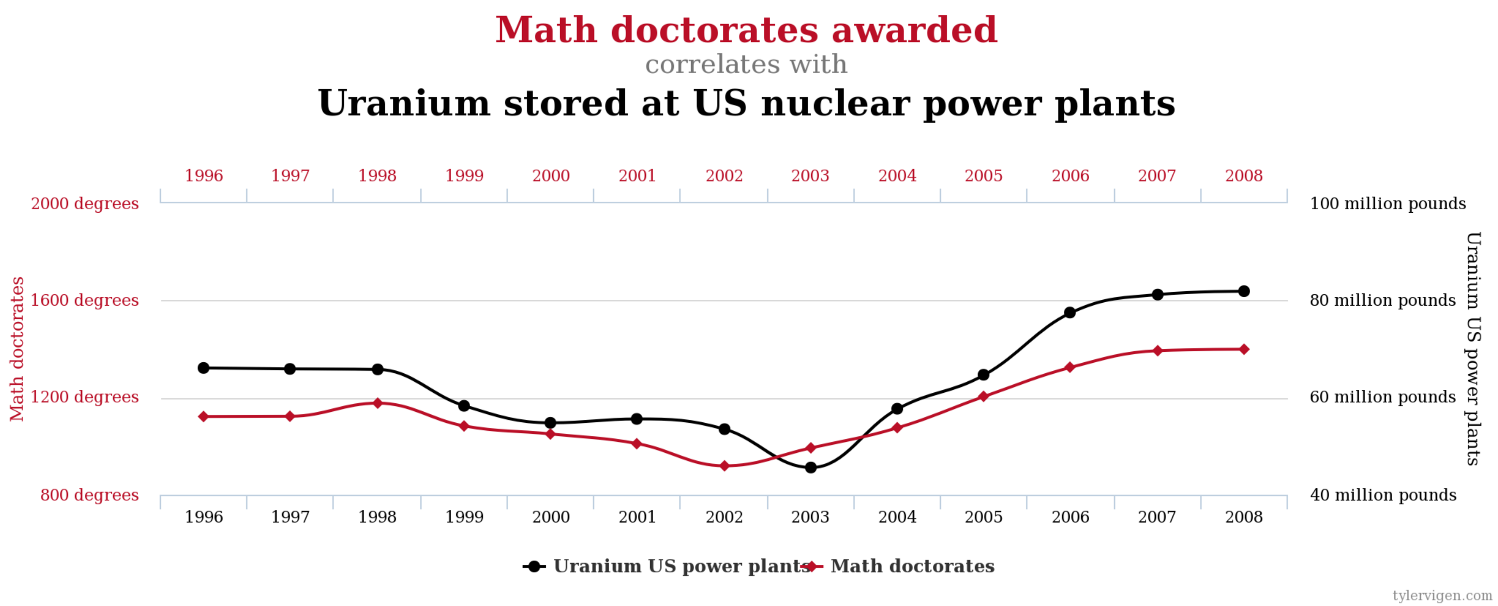
"Correlation is not causation but it sure is a hint."

Edward Tufte

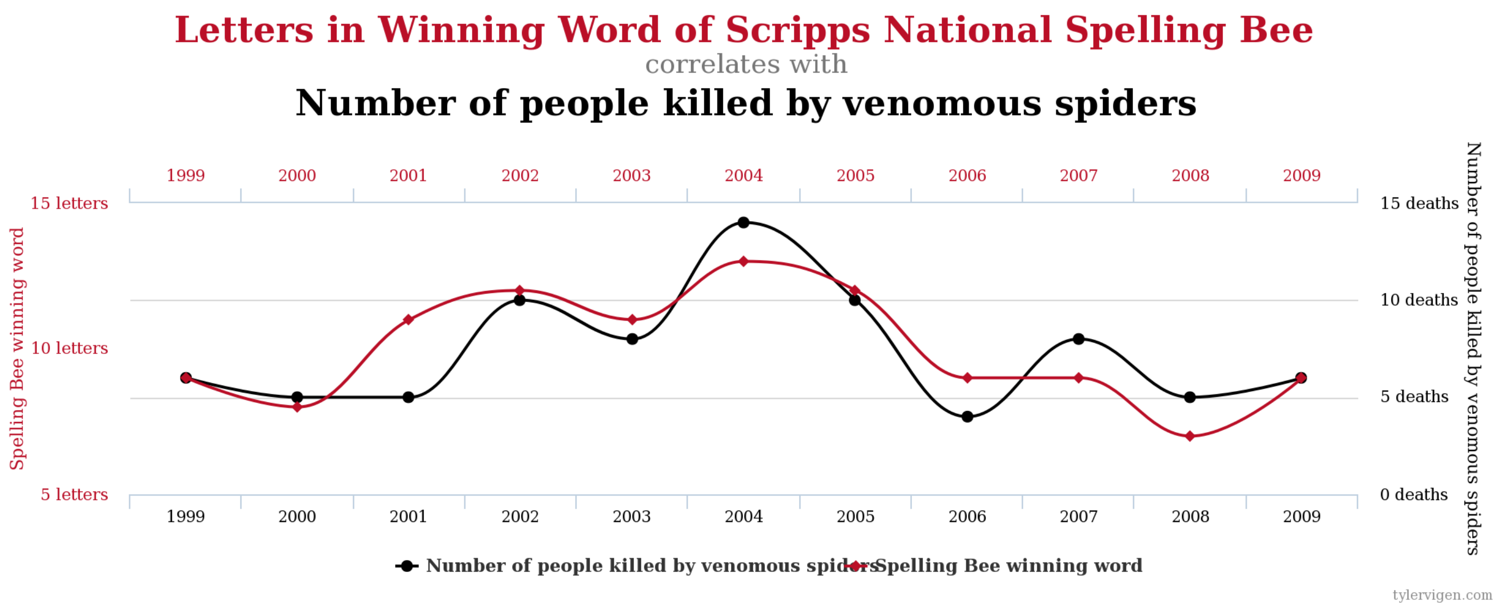
Correlation/Causation

- What are the possible causal relationships?
    - could be unrelated, i.e., coincidence
    - reverse causation
        - every time windmills are spinning it's really windy
    - missed variable, i.e., some other factor causes both
        - whenever I go to sleep with my shoes on I wake up with a headache
        - when it rains, every time I see a flash I hear a boom
    - bi-directional relationships
        - temperature and pressure
    - or they are actually the same thing
        - ºF goes up as ºC goes up

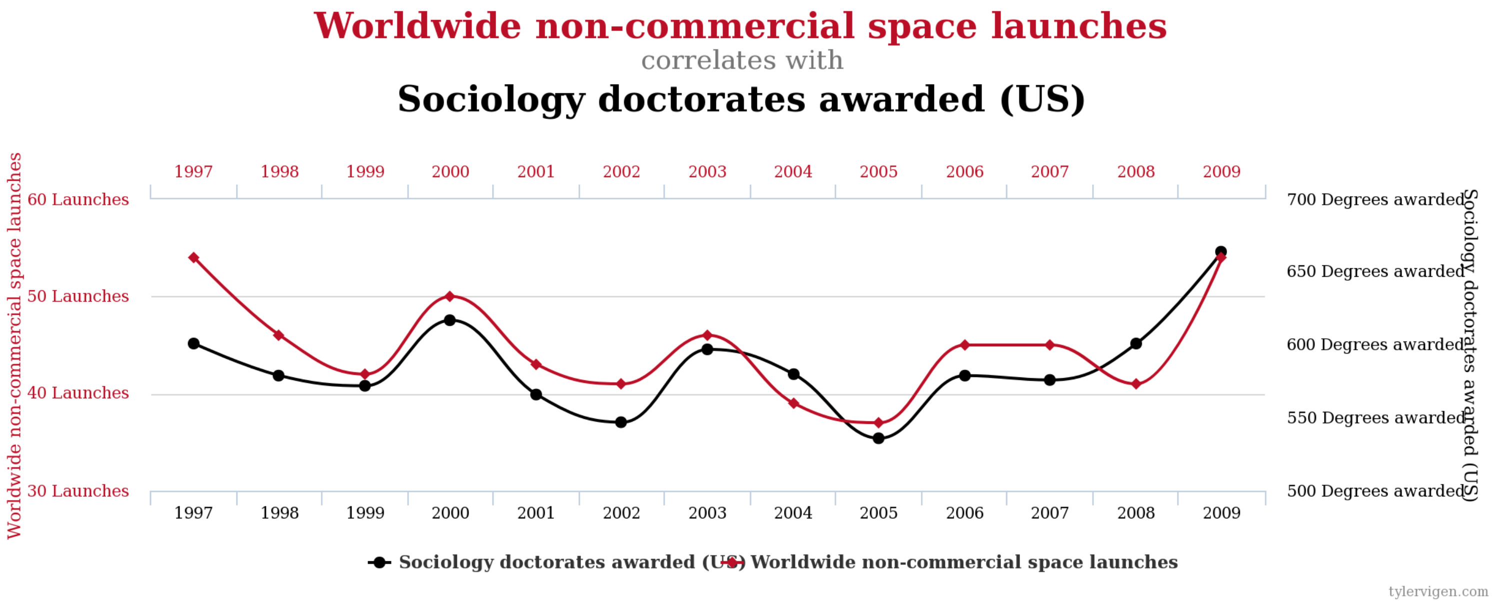
Let's Look at Some Correlations



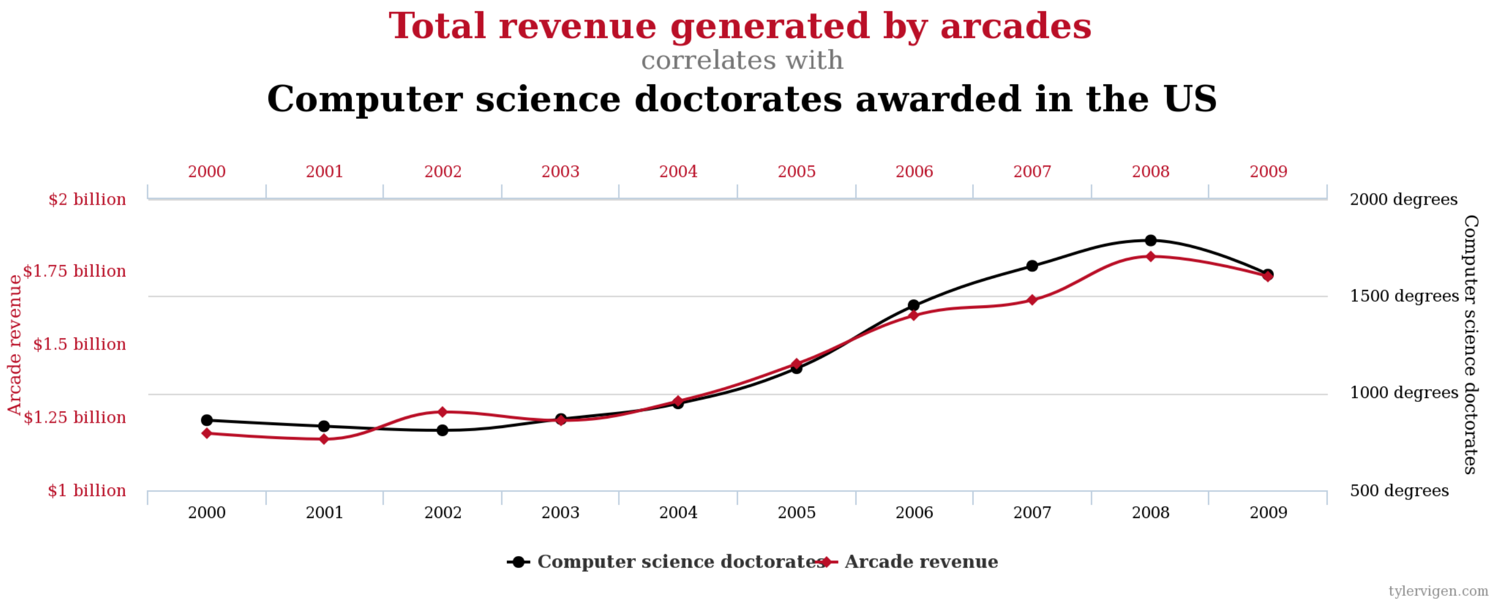
http://www.tylervigen.com/spurious-correlations



http://www.tylervigen.com/spurious-correlations



http://www.tylervigen.com/spurious-correlations



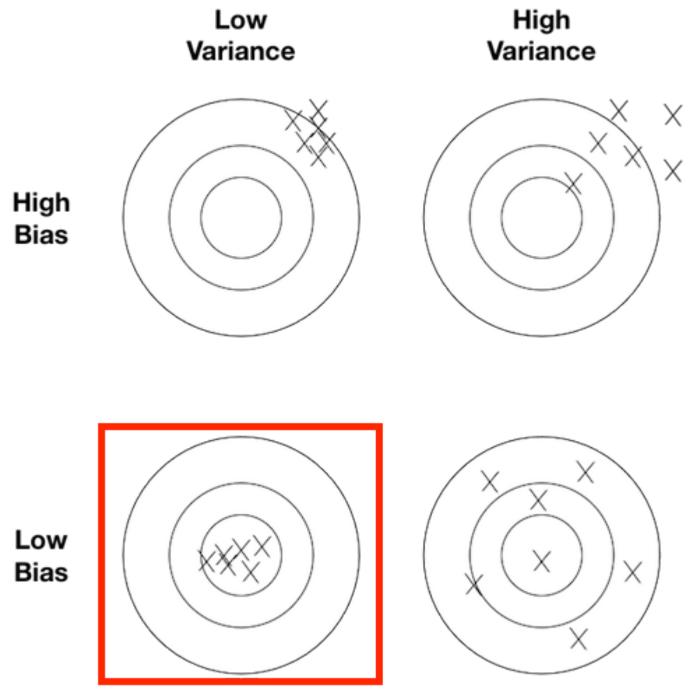
http://www.tylervigen.com/spurious-correlations

Choosing Machine Learning Algorithms:
Bias/Variance Tradeoff

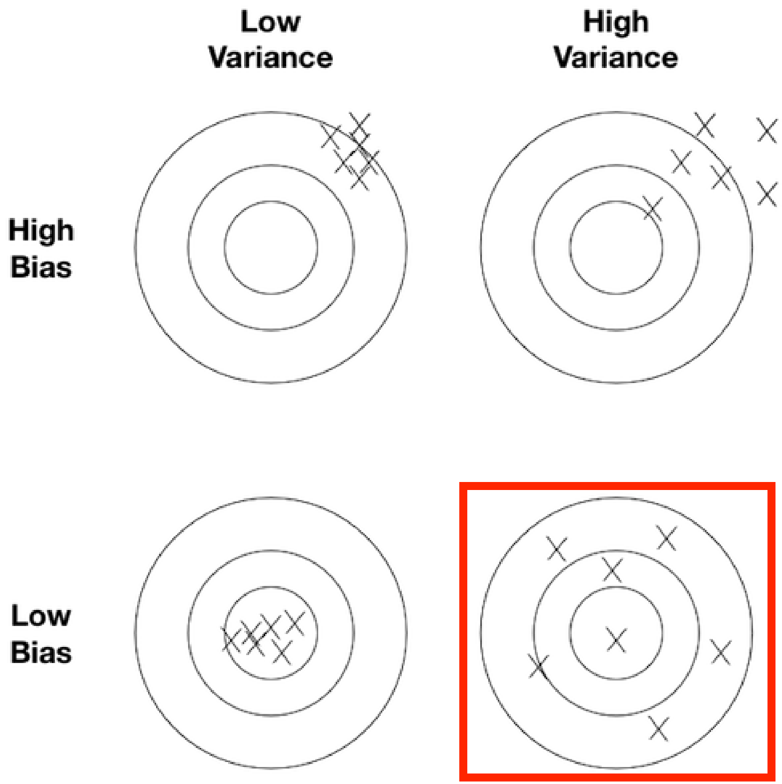
Choosing a Machine Learning Algorithm

- from an exploratory data analysis standpoint, we might try things and see what works
  - but if we can't map it back to the business function, it doesn't matter
  - if it does work, but doesn't work from an operational standpoint, you can't use it either
    - e.g., can't run on a phone, or breaks email
  - therefore, not everything that works is available to us
  - sometimes things work well in one aspect don't work well in another aspect
- A **model** is the "function" which is produced from a machine learning algorithm
- **bias** refers to errors from incorrect assumptions in the our algorithm
  - high bias means our model is pushing us in the wrong direction
  - = "underfitting"
- **variance** refers to errors from poor generalization to new, unseen inputs
  - = sensitivity to small fluctuations in the training data
  - = modeling random noise in the training data
  - = "overfitting"

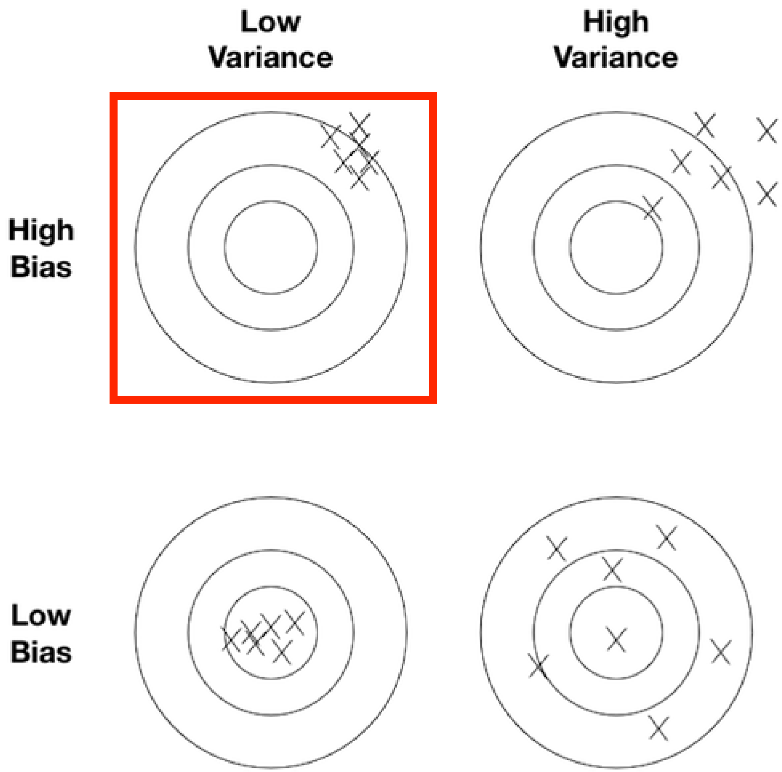
Understanding Model Bias and Variance



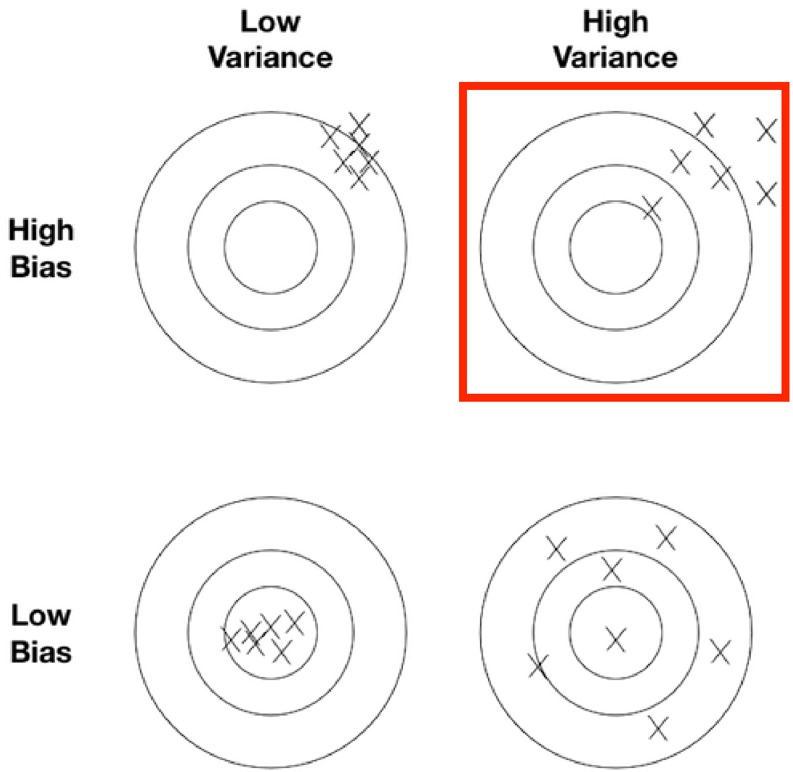
- *low bias* means our model is not pushing us in the wrong direction
- *low variance* means we're tightly clustered
- like someone who is good at playing darts, has natural hand-eye coordination, and is sober–they can put the darts wherever they want
- imagine data set that reflects the population, the features that we are trying to train off of, the method we are using to train (model we are choosing)–that should work



- let's say our dart thrower is drunk
    - darts will be all over the place, which maps to noise in the system
    - *high variance* = the model is too tightly tied to the data we trained on
    - if we pick different data, we'll get different results
    - we may need to choose a simpler model (this will be come clearer later)



- simpler models are more like our friends who have some natural ability in dart throwing bit no formal training
- we'll be wrong, but maybe right enough
- e.g., if we assume there is a linear relationship when there isn't, we'll be off from the real results



- our friend who has no skill, no training, and is drunk will have no hope of doing well
- imagine trying to build a model of credit worthiness based on your favorite color
- good algorithms + bad data = garbage

Introduction to Probability

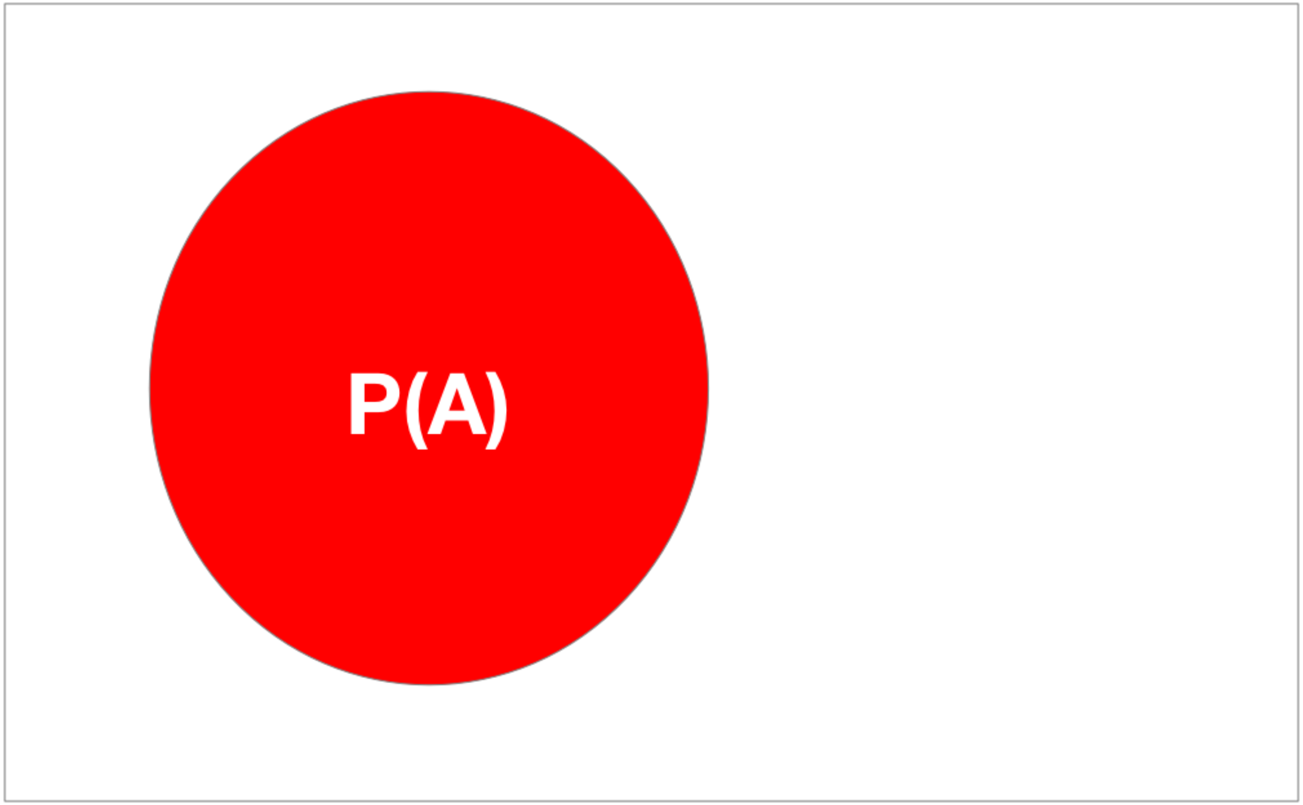
Flipping a Coin

- How many outcomes are there? (…assuming it doesn't land on its edge)
- What is the likelihood of each of those outcomes?
- We intuitively understand this when it comes to coin flipping, but let's formalize it

Definitions

- **Random Experiment** - Observing something uncertain
- **Outcome** - Result of an experiment
  - heads, tails, rolling a 3, picking an even number
- **Sample space** - Set of all possible outcomes
  - Coin: heads, tails
  - A di(c)e: 1, 2, 3, 4, 5, 6

Probability that Event A Happens



Probability of Event A: **Roll a 1**

$$P(A) = \frac{|Outcome|}{|SampleSpace|}$$

$$P(A) = \frac{|\{1\}|}{|\{1, 2, 3, 4, 5, 6\}|}$$

$$P(A) = \frac{1}{6}$$

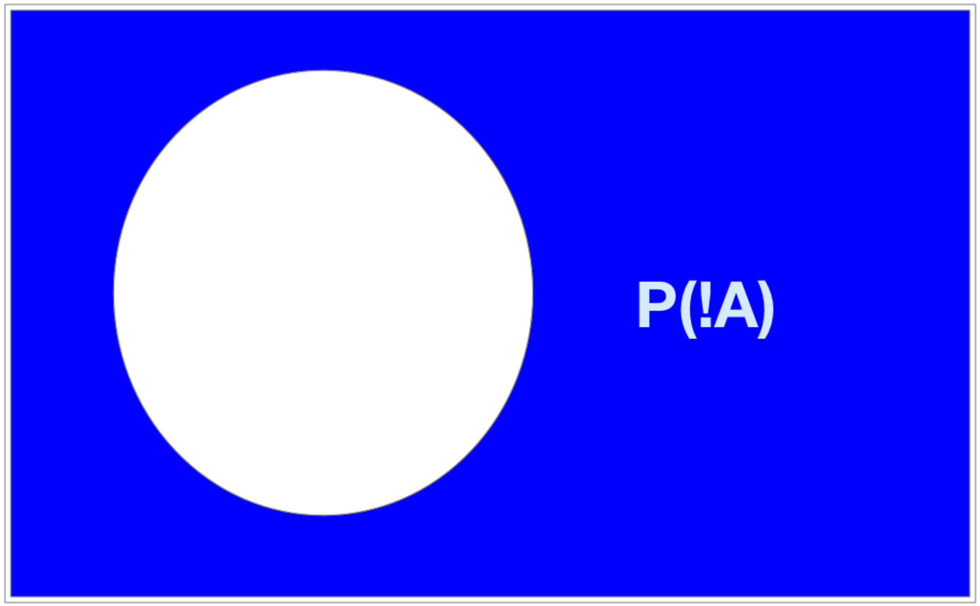
Probability of Event A: **Roll an Even Number**

$$P(A) = \frac{|Outcome|}{|SampleSpace|}$$

$$P(A) = \frac{|\{2, 4, 6\}|}{|\{1, 2, 3, 4, 5, 6\}|}$$

$$P(A) = \frac{3}{6} = \frac{1}{2}$$

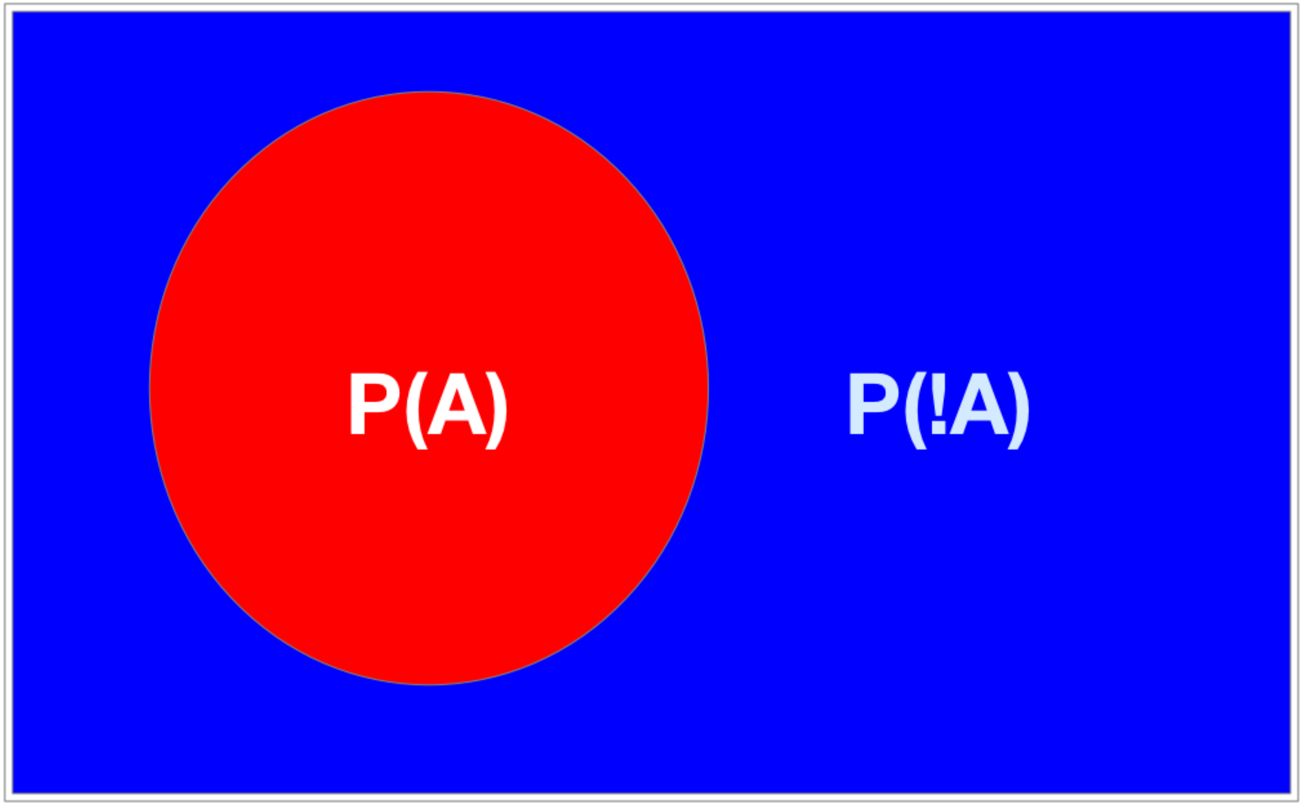
Probability that Event A Doesn't Happen



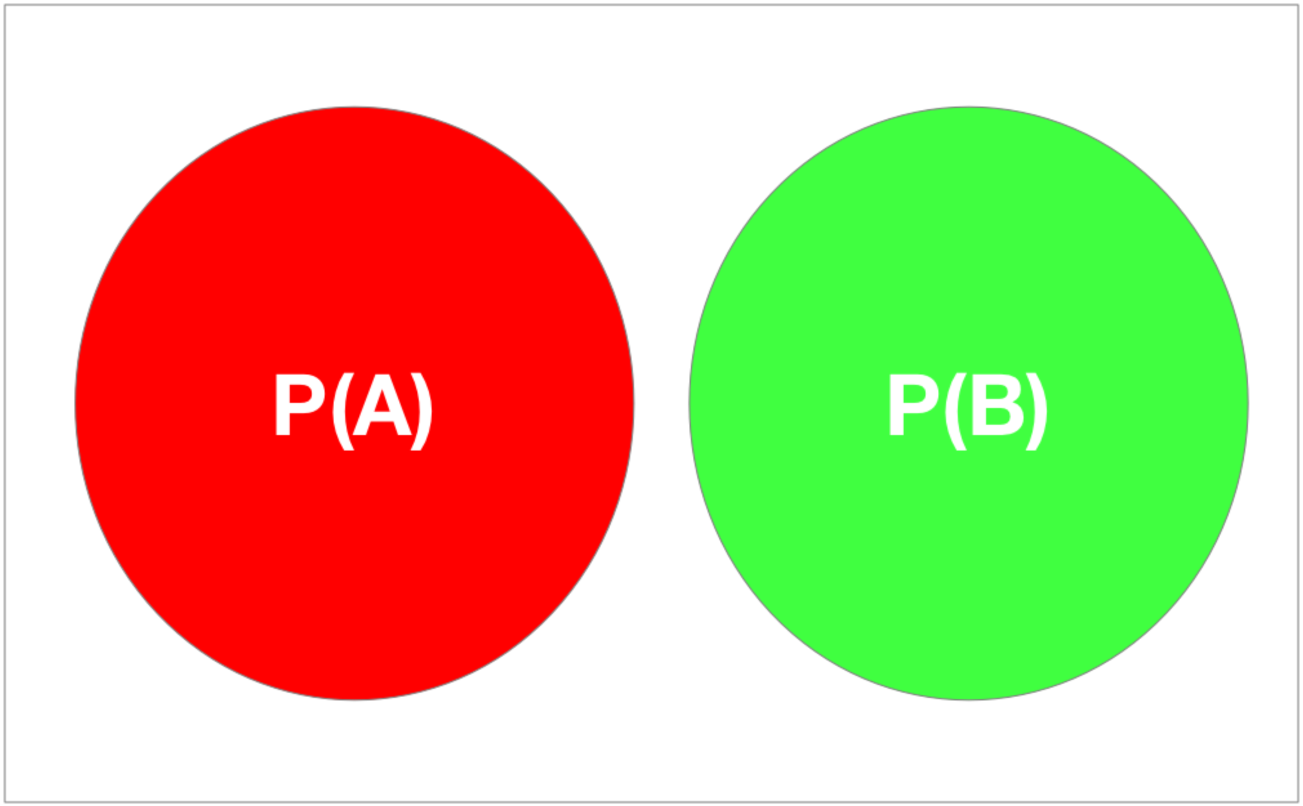
$$P(!A) = \ ?$$

$$P(!A) = 1 - P(A)$$

Whole Sample Space



Consider Two Independent Events, A and B



Event A = roll a 3

Event B = roll a 4

$$P(A) = P(\text{roll a 3})$$

$$P(B) = P(\text{roll a 4})$$

$$P(A) = P(\text{roll a } 3) = \frac{1}{6} = 0.167$$

$$P(B) = P(\text{roll a } 4) = \frac{1}{6} = 0.167$$

$$P(!A) = 1 - P(A) = \frac{5}{6} = 0.833$$

$$P(!B) = 1 - P(B) = \frac{5}{6} = 0.833$$

Probability of A and B Happening (Two Rolls)

$$P(A \cap B) = ?$$

Assuming Independent Events

$$P(A \cap B) = P(A) \cdot P(B)$$

$$= 0.167 \cdot 0.167$$

$$= 0.02789$$

Conditional Probability

Probability of "A Given B"

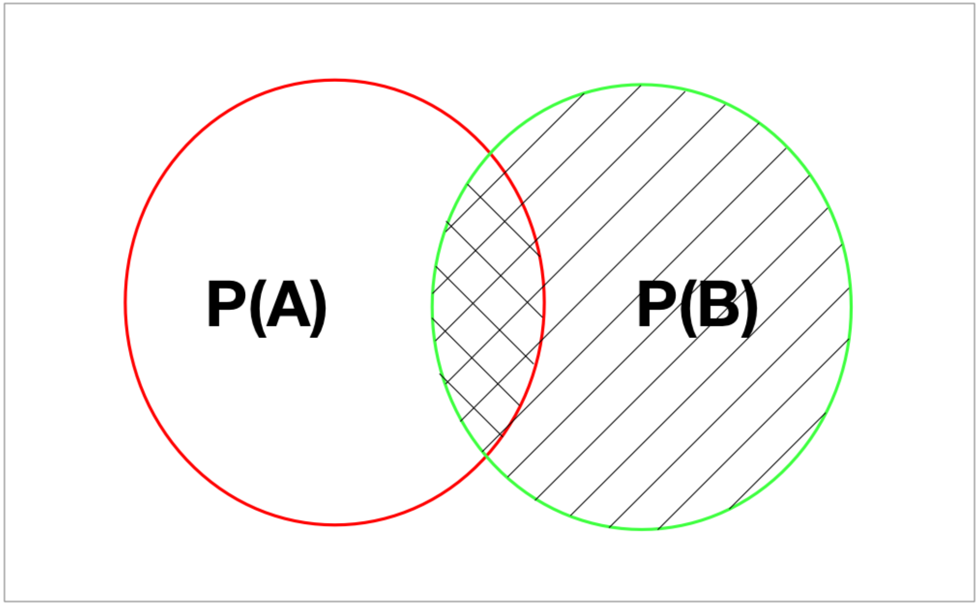
(i.e., probability of A happening, given that B has happened)

$$P(A|B) = \ ?$$

$$P(A|B) = P(\text{roll a 3}|\text{roll a 4})$$

We know intuitively that the first roll cannot affect the second, so the answer must be P(A)

P(A|B) for Non-Independent Events



$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

P(A|B) for Independent Events

$$P(A) = \text{roll a } 3$$

$$P(B) = \text{roll a } 4$$

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

$$P(A|B) = \frac{P(A) \cdot P(B)}{P(B)}$$

$$= P(A)$$

P(A|B) for Non-Independent Events

$$A = \text{roll an odd number}$$

$$B = \text{roll a number} <= 3$$

(only one roll of the die here)

$$P(A) = \frac{1}{2}$$

$$P(B) = \frac{1}{2}$$

$$P(event) = \frac{|Outcome|}{|Sample\ Space|}$$

This time, the event in question is A | B
(i.e., die shows an odd number given that the roll was <= 3)

$$P(A|B) = \frac{|A \cap B|}{|B|}$$

(We can derive the definition of conditional probability here...)

$$P(A|B) = \frac{|A \cap B|}{|B|}$$

Now divide both top and bottom by `|Sample Space|`

$$= \frac{\frac{|A \cap B|}{|Sample\ Space|}}{\frac{|B|}{|Sample\ Space|}}$$

$$= \frac{P(A \cap B)}{P(B)}$$

A = "Odd" and B = "Rolled 1, 2, or 3"

$$B = \{1, 2, 3\}$$

$$A \cap B = \{1, 3\}$$

$$P(A|B) = \frac{|A \cap B|}{|B|}$$

$$= \frac{2}{3}$$

Exercise

1. Go to setosa.io

2. Change the *perspective* from **world** to **P(B|A)** and **P(A|B)** to be sure you understand what is happening in the simulation

3. Adjusting the *drop frequency* can also make it easier to see what's going on

4. Adjust P(A) so that the red shelf completely covers the blue shelf and be sure you understand how that affects P(A|B)

5. Reload the page and perform a similar adjustment for P(B)

6. Adjust the P(A n B) slider up and down and be sure you understand the effect on the probabilites

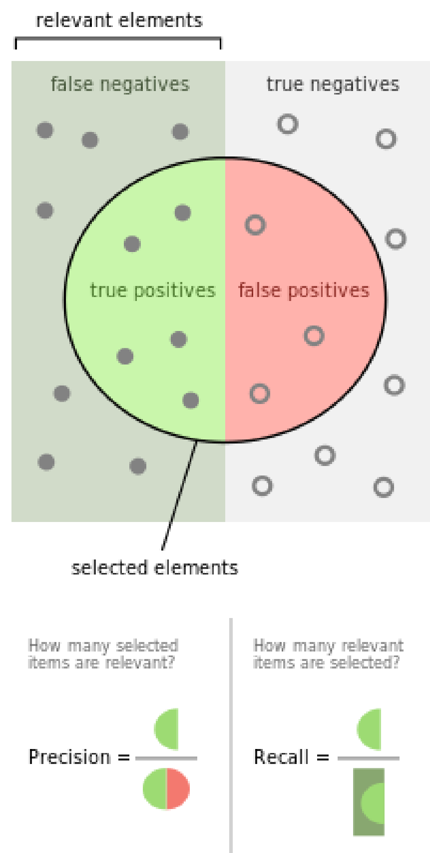
2. Solve the following problem

- Out of a sample of 1000 Tesla Model 3 pre-orders, 400 purchased the long range battery (LRB), 325 purchased the all-wheel drive (AWD) option, and 180 buyers purchased both options. If a random buyer from this group purchased the LRB, what is the probability that the buyer also purchased the AWD?

Understanding Classification Results

Precision, Recall and Accuracy of a Model

- How accurate are we?
- How many things might we have missed?



Example: Hypothetical Terrorist Identification System

- It is guaranteed not to miss a single terrorist
- **You're all terrorists**
- We did not miss a single terrorist
- PROBLEM: Too many false positives
- Let's say there are 100 terrorists in the country…
    - Our system got them all, but wasn't very precise!
- Recall is a function of how many we got right–we got all 100 terrorists so recall was great!

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

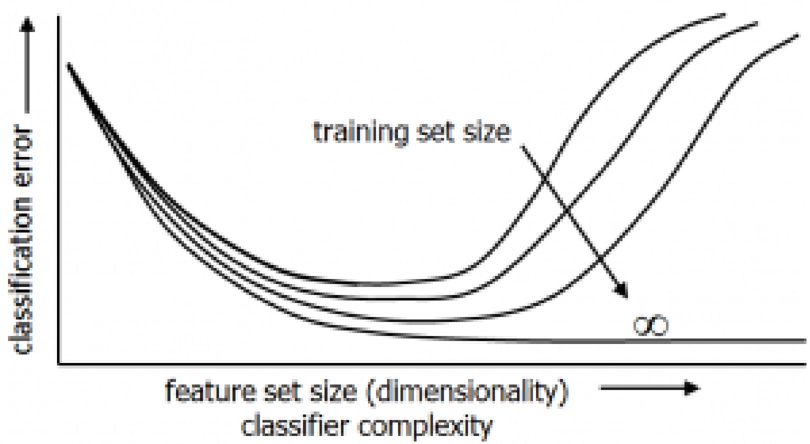
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

In what kind of real-world system would we prefer recall over precision?

Curse of Dimensionality

Curse of Dimensionality

- Our intuition is that more is always better, but that is not always true
- On the other hand, many things that were not possible before are now possible with Big Data
- The more features you look at, the more data you need to justify it, the harder it is to draw conclusions
- In other words, as the dimensionality increases, the volume of the space increases so fast that the available data become sparse
- Hughes Phenomenon: predictive power of a classifier/regressor increases as number of dimensions/features considered increases, then decreases



http://37steps.com/2279/the-peaking-paradox

Example Dataset

$$X^T = \{x_1, \ldots, x_N\}, N = 20 \text{ (i.e., observations)}$$

$$T^T = \{t_1, \ldots t_N\}, t \in \{g, r\}$$

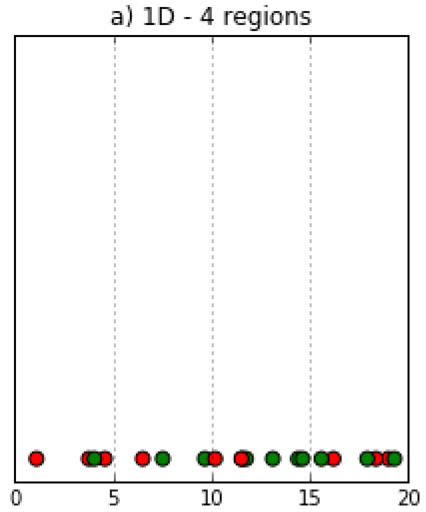
(i.e., target class of each input, either red or green)

- we want a classifier to output the probability that each x is green

$$p(t_n = g | x_n)$$

- as a first approach, we'll partition X into four equal-sized regions

http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/



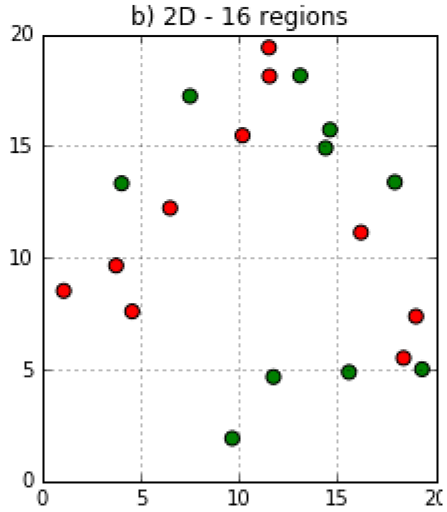
$\mathbf{R_1}$**:** 4 dots, 1 green, 3 red $\quad$ $\mathbf{R_3}$**:** 7 dots, 4 green, 3 red

$\mathbf{R_2}$**:** 3 dots, 2 green, 1 red $\quad$ $\mathbf{R_4}$**:** 6 dots, 3 green, 3 red

$$p(t_n = g|R_1) = \frac{1}{4} = 0.25$$

$$\frac{20}{4} = 5 \text{ obs per region on average}$$

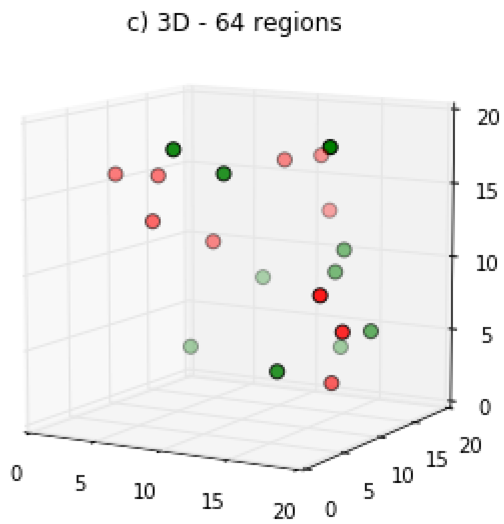
http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/



$$X^T = \{(x_{11}, x_{12}), \ldots, (x_{N1}, x_{N2})\}$$

$$\frac{20}{16} = 1.25 \text{ obs per region}$$

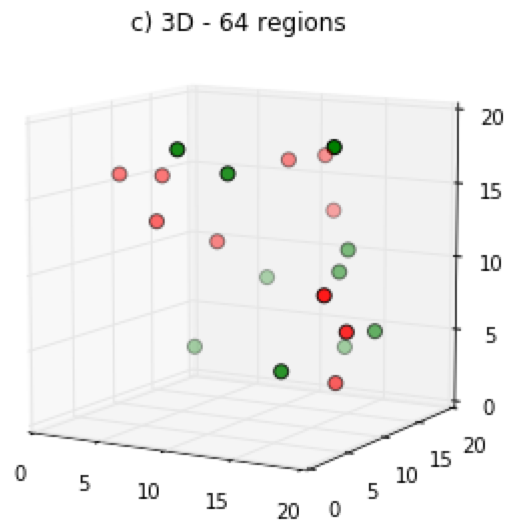
http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/



$$X^T = \{(x_{11}, x_{12}, x_{13}), \ldots, (x_{N1}, x_{N2}, x_{N3})\}$$

$$\frac{20}{64} \approx 0.31 \text{ obs per region}$$

http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/

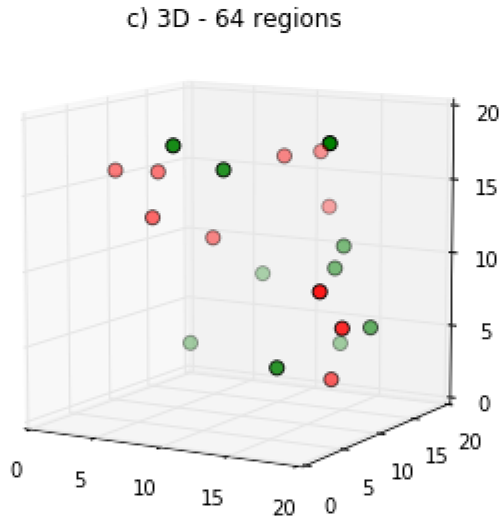


Sampling density is proportional to $N^{\frac{1}{D}}$

$$20^{\frac{1}{1}} = 8000^{\frac{1}{3}}$$

As the number of dimensions increase the number of samples we need grows exponentially

http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/



Sampling density is proportional to $N^{\frac{1}{D}}$

$$20^{\frac{1}{1}} = 8000^{\frac{1}{3}}$$

As the number of dimensions increase the number of samples we need grows exponentially

We need enough dimensions to have ample variation to detect the classification of each input–too many dimensions requires too much data, two few does not have enough feature variation to detect anything

http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/