CSE 4/587

Data Intensive Computing

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Dr. Shamshad Parvin
shamsadp@buffalo.edu
313 Davis Hall

Course Recap

Announcements

- Phase 3 deadline extended to Wednesday
- Evaluation Progress:
    - 487: 85%
    - 587: 85%
    - Thank you!

Prelude:
Motivation and Goals

Deluge of Data (data is everywhere)

Bioinformatics data: from about 3.3 billion base pairs in a human genome to huge number of sequences of proteins and the analysis of their behaviors

The internet: web logs, facebook, twitter, maps, blogs, etc.

Financial applications: volumes of data for trends and other deeper knowledge

Healthcare: huge amount of patient data, drug and treatment data

The universe: the Hubble ultra deep telescope shows 100s of galaxies each with billions of stars. Sloan Digital Sky Survey: https://www.sdss.org/

Combined with more powerful methods...

Tremendous advances have taken place in **statistical methods and tools**, **machine learning** and **data mining approaches**, and **internet-based dissemination tools** for analysis and visualization.

- Many tools are open source and freely available for anybody to use.
- Is there an easy entry-point into learning these technologies?
- Can we make these tools easily accessible to the students, researchers and decision makers similar to how “office” productivity software is used?

High Level Goals

1. Understand foundations of data analytics so that you can **interpret and communicate results to make informed decisions**
2. Study and learn to apply common **statistical methods** and **machine learning algorithms** to solve business problems
3. Learn to work with popular **tools** to analyze and visualize data
4. **Understand distributed techniques** for storage and deployment of applications
5. **Transform complex analytics into routine processes**

What is Data Intensive Computing?

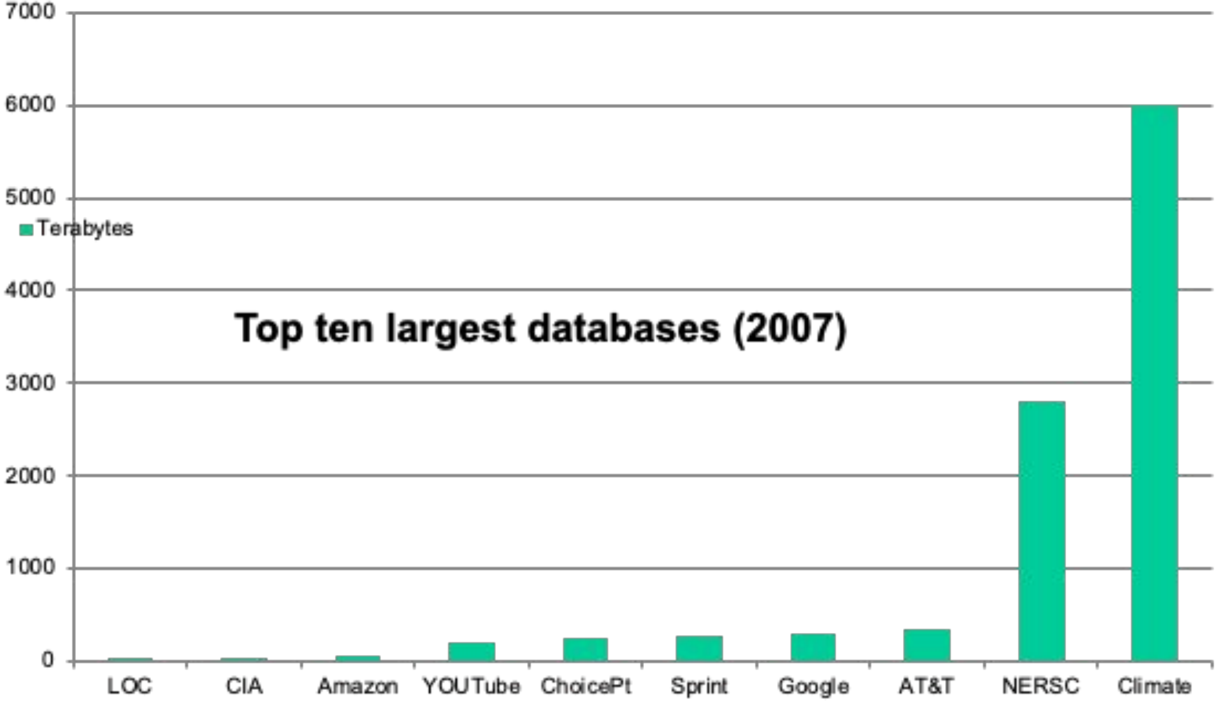
The phrase was initially coined by National Science Foundation (NSF)

- What is it?
  - Volume, velocity, variety, veracity (uncertainty) (Gartner, IBM)
- What do you expect to extract by processing this large data?
  - Intelligence for decision making
- What is different now?
  - Storage models, processing models
  - Big Data, analytics and cloud infrastructures

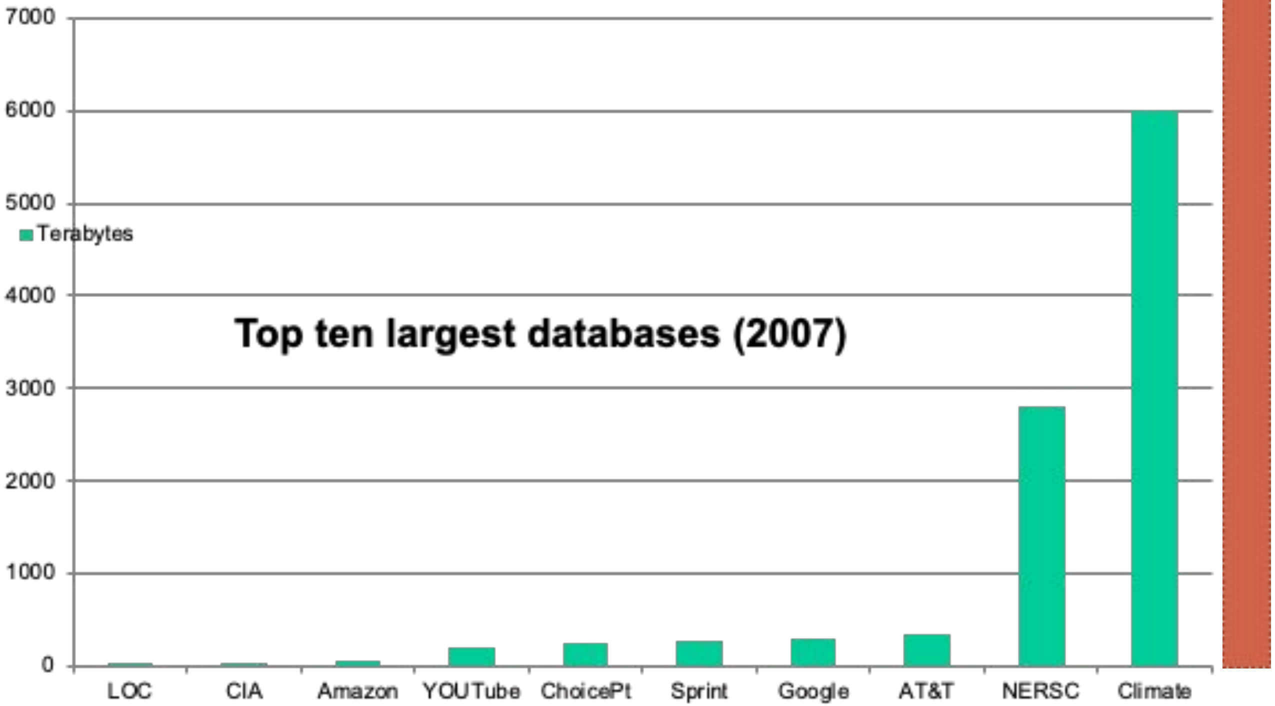
Examples of Data Intensive Applications

- Search engines
- Recommendation systems
  - Netflix: movie recommendations
  - Amazon: book/product recommendations
- Biological systems
  - Analysis (ie disease-gene match)
  - Query/search for gene sequences
- Space exploration
- Financial analysis

…and many more…



Ref: http://www.comparebusinessproducts.com/fyi/10-largest-databases-in-the-world/



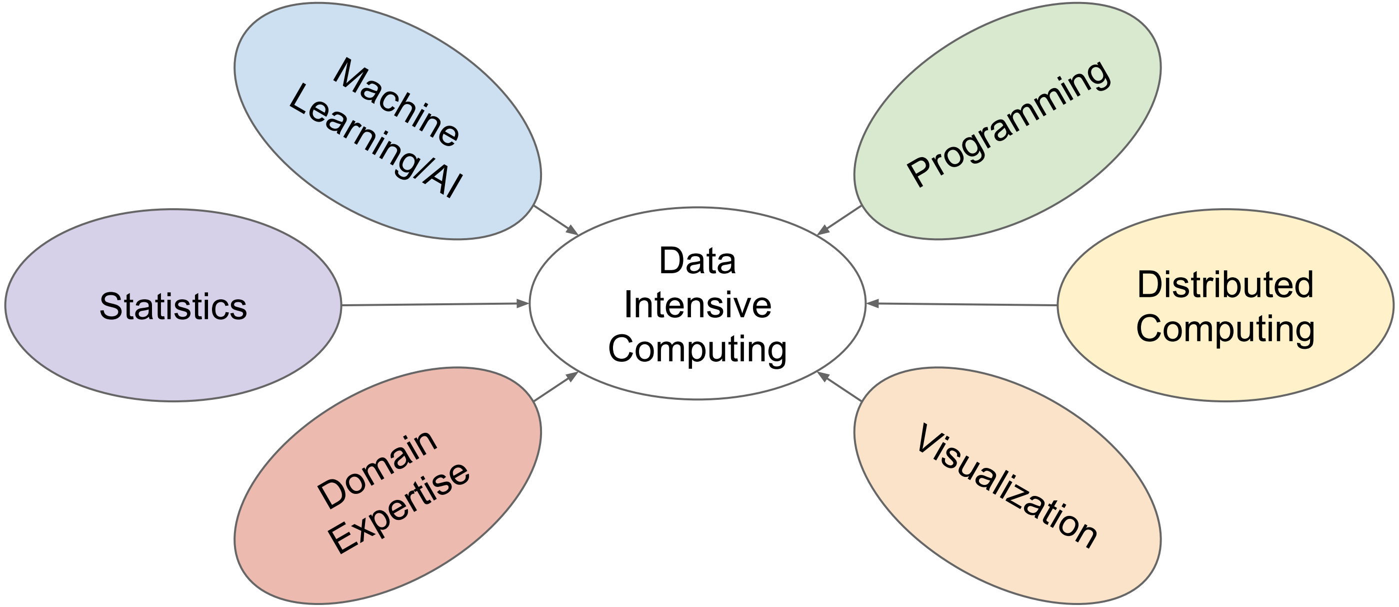
Facebook
@ 21
petabytes
in 2010

Ref: http://www.comparebusinessproducts.com/fyi/10-largest-databases-in-the-world/

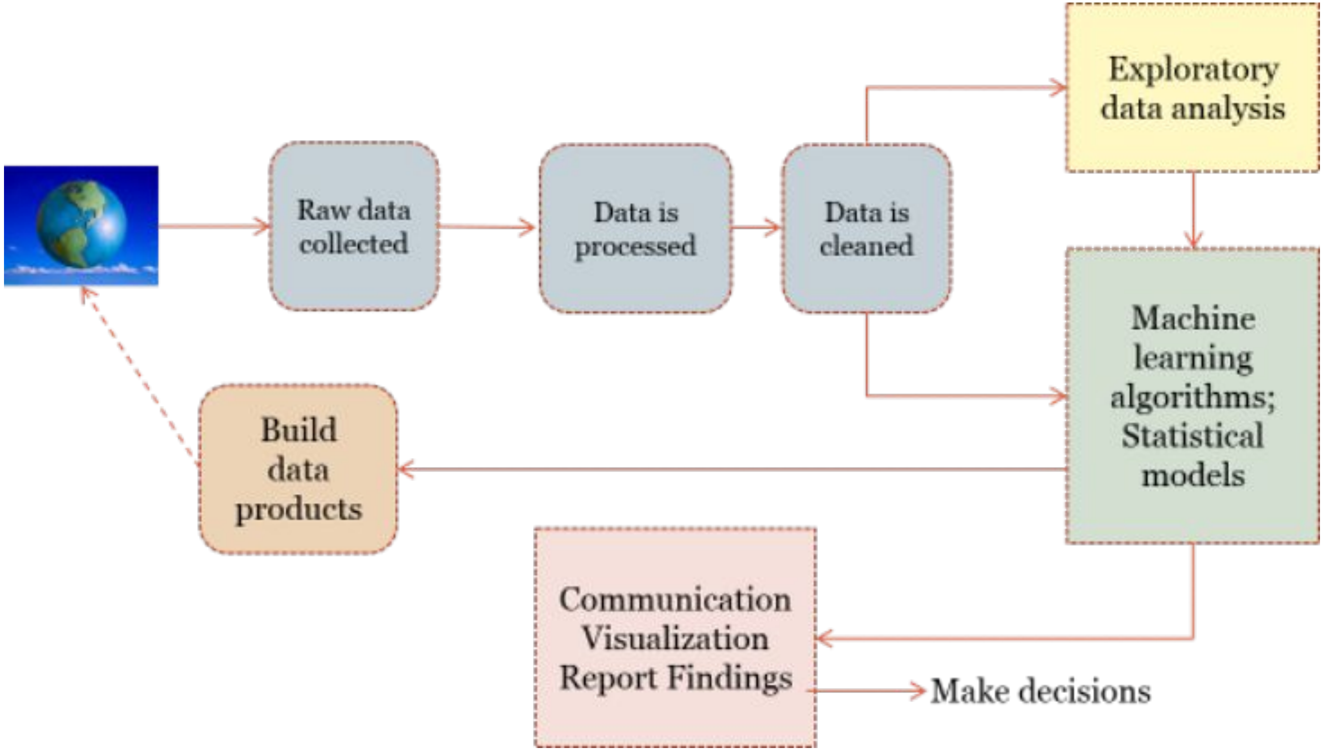
DIC Skill Diversity

Data Intensive Computing requires a diverse set of skills

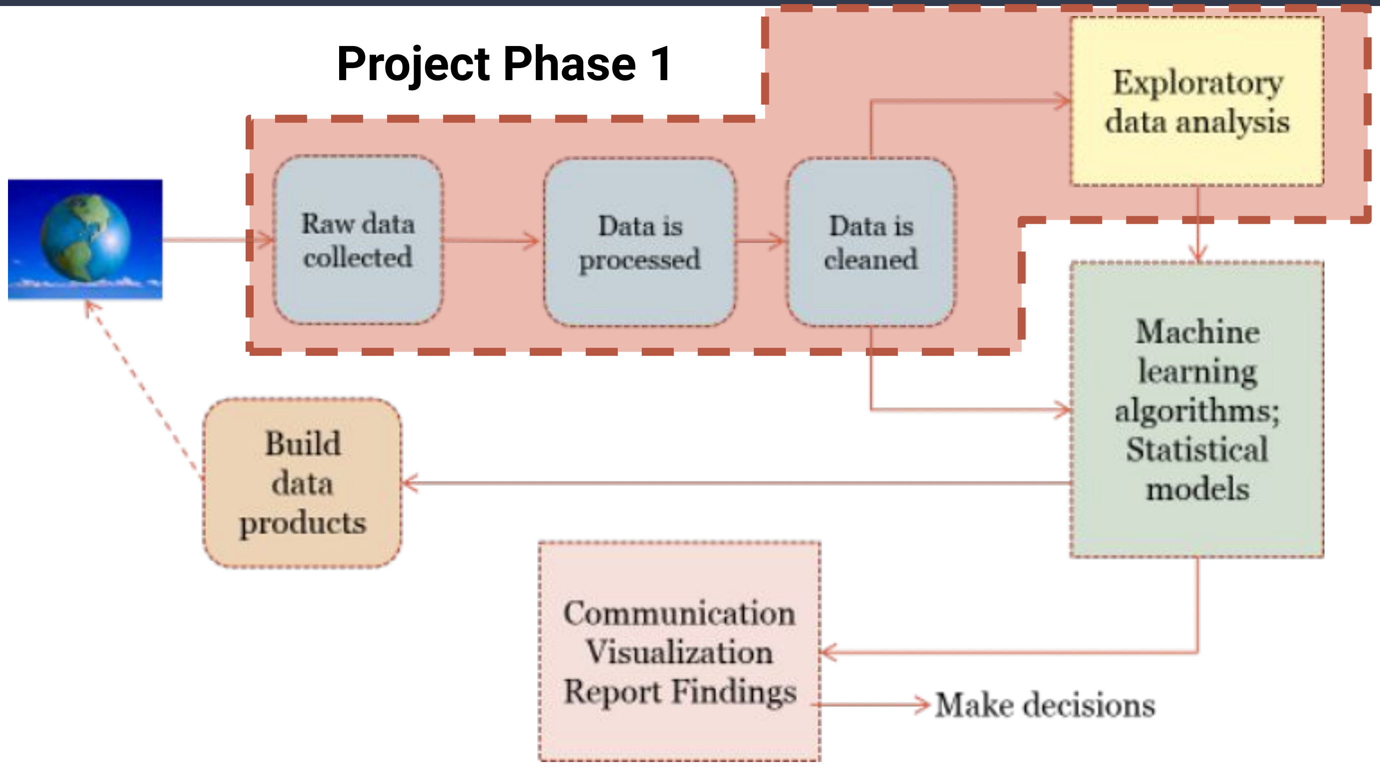
(and no one person will likely be an expert in all of them)



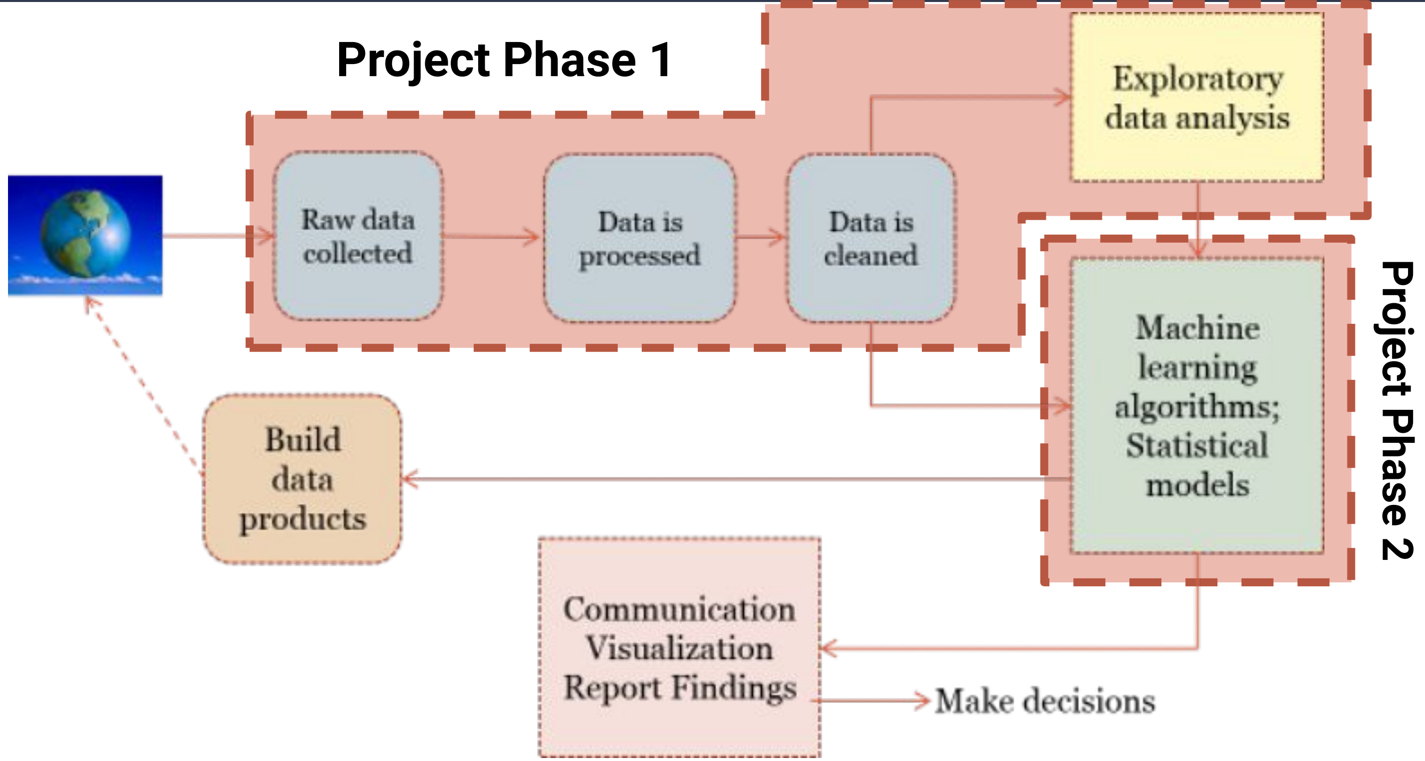
Data Science Pipeline


Raw data collected
Data is processed
Data is cleaned
Exploratory data analysis
Machine learning algorithms; Statistical models
Build data products
Communication Visualization Report Findings
Make decisions

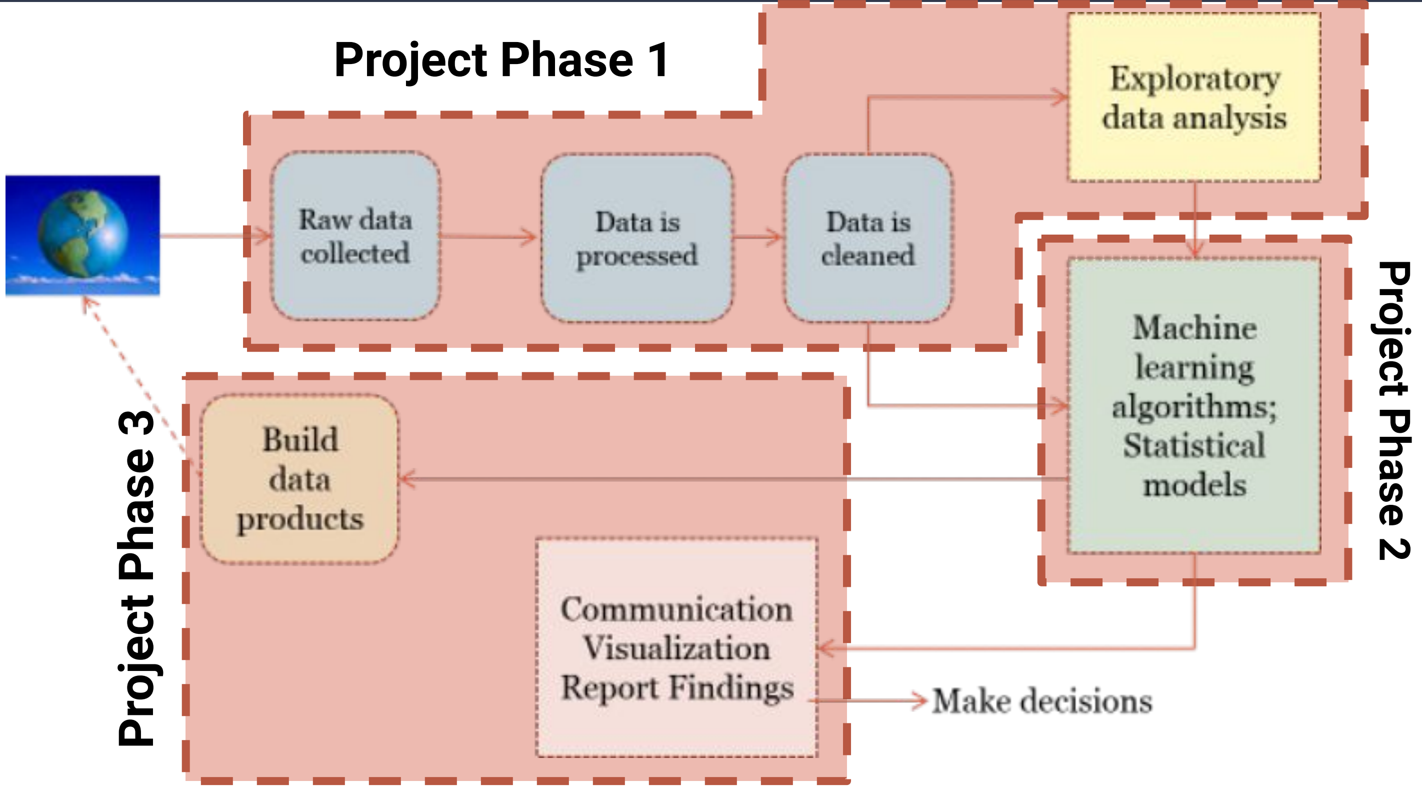
Data Science Pipeline


Project Phase 1
Raw data collected
Data is processed
Data is cleaned
Exploratory data analysis
Machine learning algorithms; Statistical models
Build data products
Communication Visualization Report Findings
Make decisions

Data Science Pipeline


Project Phase 1
Raw data collected
Data is processed
Data is cleaned
Exploratory data analysis
Machine learning algorithms; Statistical models
Project Phase 2
Build data products
Communication Visualization Report Findings
Make decisions

Data Science Pipeline



Phase 1:
Understand Your Data

Understand the Data

- Data represents the traces of real-world processes
  - What traces we collect depends on the sampling methods
  - You build models to understand the data and extract meaning and information from the data: statistical inference
- Two sources of randomness and uncertainty
  - The process that generates data is random
  - The sampling process itself is random

Questions to ask

- How big is the data?
- Any outliers?
- Missing data? How to address it? (Clean our data…)
- Sparse or dense?
- Collision of identifiers in different sets of data

Population and Sample

- Population is complete set of traces/data points
  - US population: 314 Million, world population: 7 billion for example
  - All voters, all things
- Sample is a subset of the complete set (or population): **how we select the sample introduces biases into the data**

Big Data vs Statistical Inference

- Sample size N
  - For statistical inference N < All
  - For big data N == All
  - For some atypical big data analysis N == 1
    - World model through the eyes of a prolific twitter user
    - Followers of Ashton Kutcher: If you analyze the twitter data you may get a world view from his point of view

Big Data Context

- Sampling is still a valid solution, it depends on your needs
  - For quick analysis, or inference purposes you don’t need all the data
  - At Google (at the originator big data algs.) people sample all the time.
- However, if you want to serve and render information in a UI, you cannot sample.
- Some DNA-based search you cannot sample.
- **Just because you have an entire population does not mean there is no bias**

Exploratory Data Analysis (EDA)

- By doing EDA, you achieve two things to get you started:
  - You get an intuitive feel for the data
  - You can start to make a list of hypotheses
- EDA is the prototype phase of ML and other sophisticated approaches
- Basic tools of EDA are plots, graphs, and summary stats (a lot of histograms)...
- It is a method for “systematically” going through data, plotting distributions, plotting time series, looking at pairwise relationships using scatter plots, generating summary stats.eg. mean, min, max, upper, lower quartiles, identifying outliers.
- **EDA is done to understand big data before using expensive big data methodology.**

Example from Data Science from Scratch (Ch. 5)

- Consider dividing up data scientists you know into bins based on which coast they are from and finding the average number of friends in for each bin

Coast	# of members	Avg. # of friends
West Coast	101	8.2
East Coast	103	6.5

It would appear that the West Coast scientists are, by this metric, “friendlier”

…however

Example from Data Science from Scratch (Ch. 5)

- Consider dividing up data scientists you know into bins based on which coast they are from and finding the average number of friends in for each bin

Coa		
Wes		
Eas		

If we also bin by degree, we see a different story…

East coast has a higher percentage of PhD members, but each bin is “friendlier”

Coast	Degree	# of members	Avg. # of friends
West Coast	PhD	35	3.1
East Coast	PhD	70	**3.2**
West Coast	No PhD	66	10.9
East Coast	No PhD	33	**13.4**

Example from Data Science from Scratch (Ch. 5)

- Consider dividing up data scientists you know into bins based on which coast they are from and finding the average number of friends in for each bin

If we also bin by degree, we see a different story…

East coast has a higher percentage of PhD members, but each bin is “friendlier”

Coa		
Wes		
Eas		

Coast	Degree	# of members	Avg. # of friends
W			3.1
Ea			**3.2**
W			10.9
East Coast	No PhD	33	**13.4**

From this we can conclude that the degree may also be a relevant factor in “friendliness” and form new hypotheses

Phase 2:
Apply algorithms to learn from your data

Models/Algorithms

Supervised

Prediction

- Linear Regression

Classification

- k-Nearest Neighbor
- Naive Bayes
- Logistic Regression

Unsupervised

Clustering

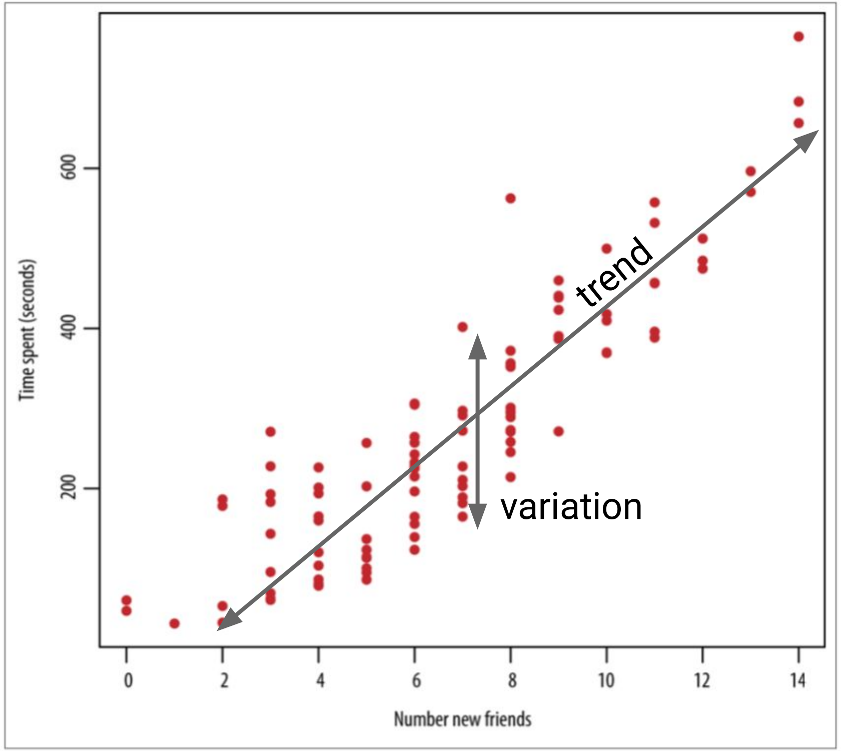
- k-Means

Other

- PageRank

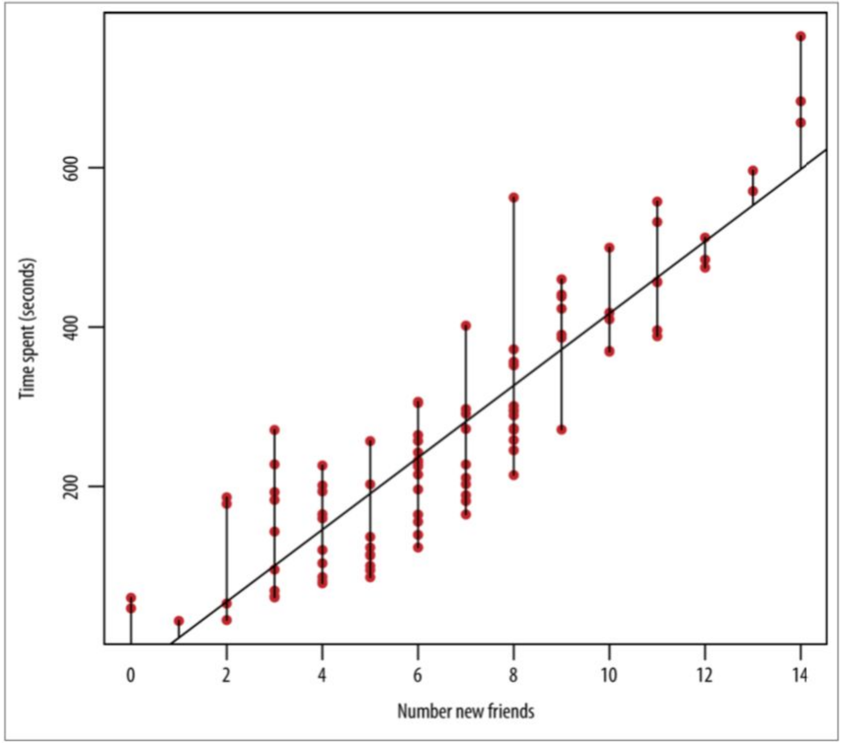
Linear Regression

- We want to capture 2 factors: trend and variation
- Assume a linear relationship ($y=\beta_0 + \beta_1 x$)
- Now we must *"fit"* the model - use an algorithm to find the best values of β_0 and β_1



Linear Regression

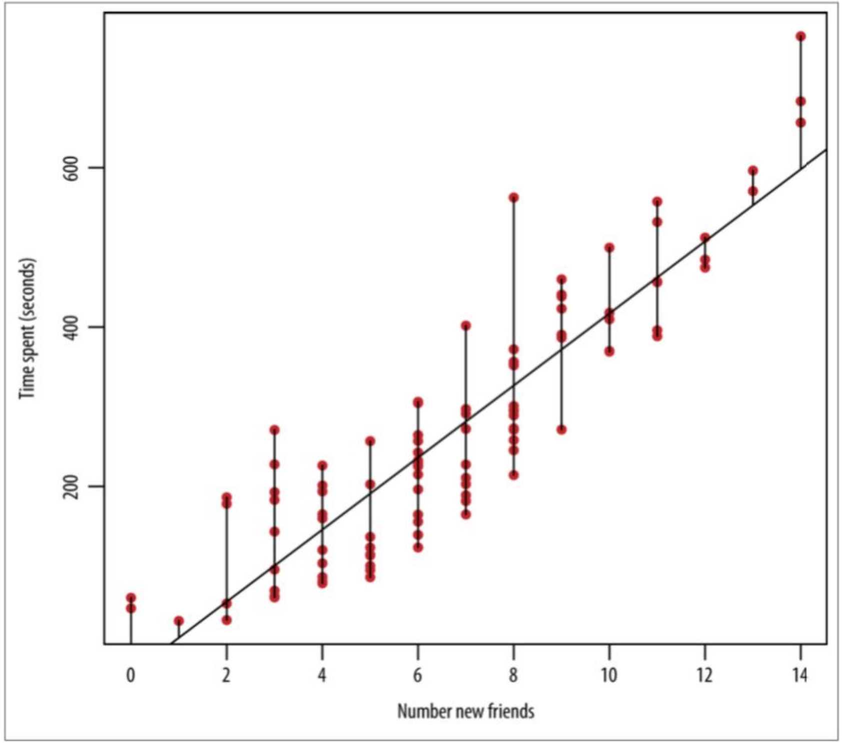
- Running the data through a solver yields β_0 = -32.08 and β_1= 45.92
- How confident are we in this model?
- If we have a new user, with 5 new friends, can we predict how much time they'll spend?



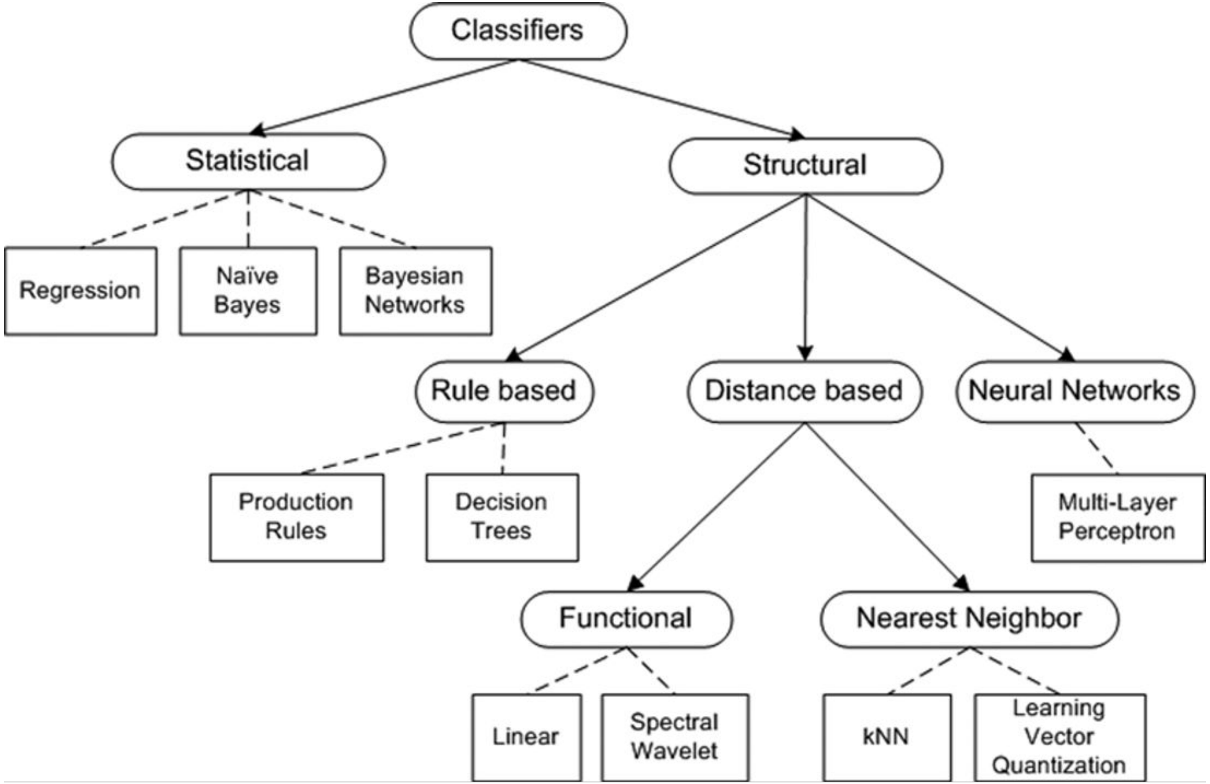
Linear Regression

- Running the data through a solver yields β_0 = -32.08 and β_1= 45.92
- How confident are we in this model?
- If we have a new user, with 5 new friends, can we predict how much time they'll spend?

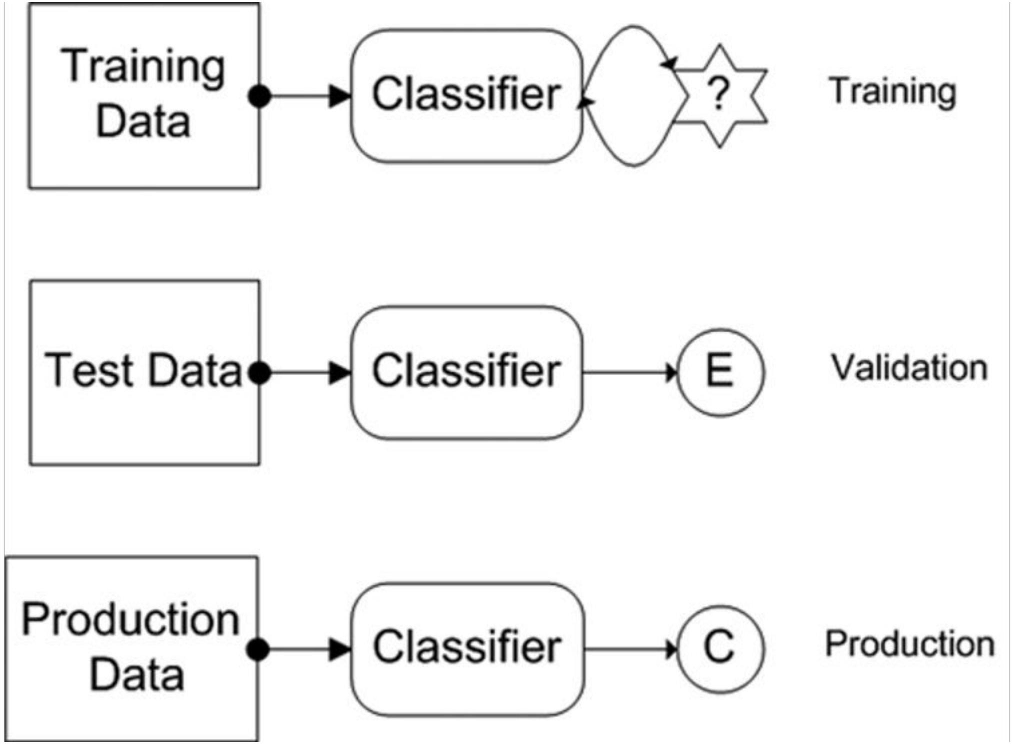
...This, afterall, is the whole goal for modeling in the first place, right?



Classification of Classification Algorithms

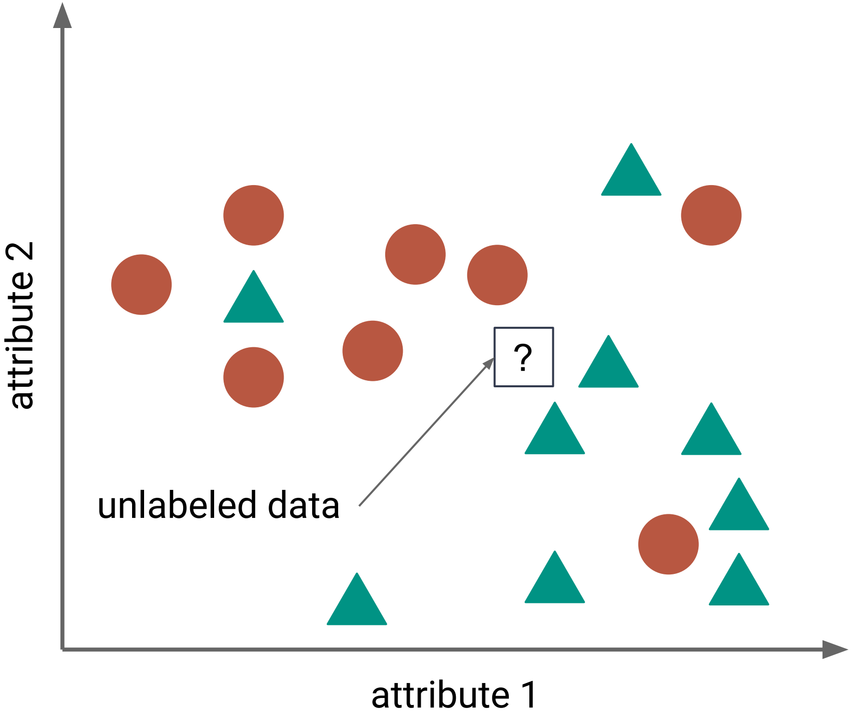


Life Cycle of Classifiers



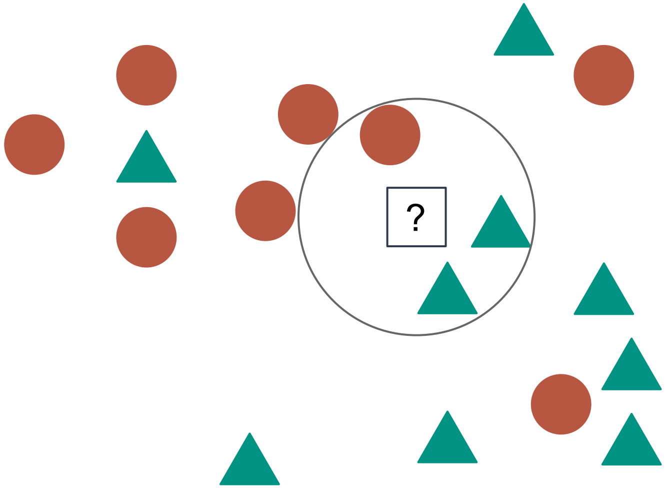
k-Nearest Neighbors

- For the example to the left, we have a number of data points labeled as either red circles, or green triangles
- How do we label the new unknown data point?
- Depends on the value of k



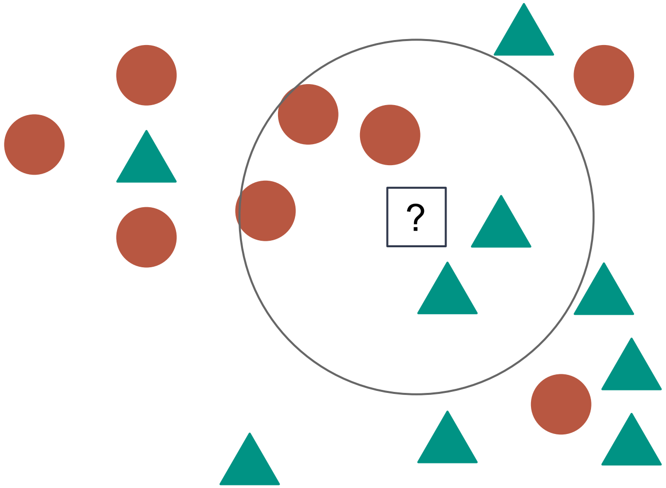
k-Nearest Neighbors

- If k = 3:
  - Green triangles have 2 votes
  - Red circles have 1 vote
  - The new point will be labeled green triangle



k-Nearest Neighbors

- If k = 5:
  - Green triangles have 2 votes
  - Red circles have 3 votes
  - The new point will be labeled red circle



Naive Bayes and Logistic Regression

Basic Idea: Make a probabilistic model – have many ***simple rules***, and aggregate those rules together to provide a probability.

Naive Bayes

Basic principle: P(*H* | *E*) = P(*E* | *H*) * P(*H*) / P(*E*)

Posterior probability is proportional to likelihood times prior

- *H* – hypothesis   *E* – evidence
- **Prior** = probability of the *E* given *H*; P(*E* | *H*)
- **Likelihood** = P(*H*) / P(*E*)
- **Posterior** = Probability of *H* given *E*; P(*H* | *E*)

Logistic Regression

Odds Ratio: $\frac{p}{1-p}$

The ***logit*** function is the basic building block of Logistic Regression

$$logit(p) = log(\frac{p}{1-p}) = log(p) - log(1-p)$$

It takes an ***x*** value in the range [0,1] (ie a probability) and transforms it to ***y*** values ranging across all real numbers

Classifiers Summary

- k-NN works well for a low number of dimensions (features)
- Naive Bayes and Log Regression work well with large number of features
- Naive Bayes is generative and requires independent features
- Log Regression is discriminative and does not require independent features

http://robotics.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf

k-Means

1. Choose the number of clusters
2. Initialize centroids to some value
    a. Could be via some special algorithm, or could be random
3. Then repeat the following steps…
    a. Reassign all points to the closest centroid
    b. Recalculate the centroids position based on this assignment
4. …until there is no change in centroid values or points stop switching

Interactive Example: Visualizing K-Means Clustering

PageRank

Googles solution for PageRank:

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{N}$$

In matrix notation:

$$A = \beta\, M + (1 - \beta) \left[\frac{1}{N}\right]_{N \times N}$$

Interlude:
What if our data gets too big?

Scaling Up: Distributed Techniques in DIC

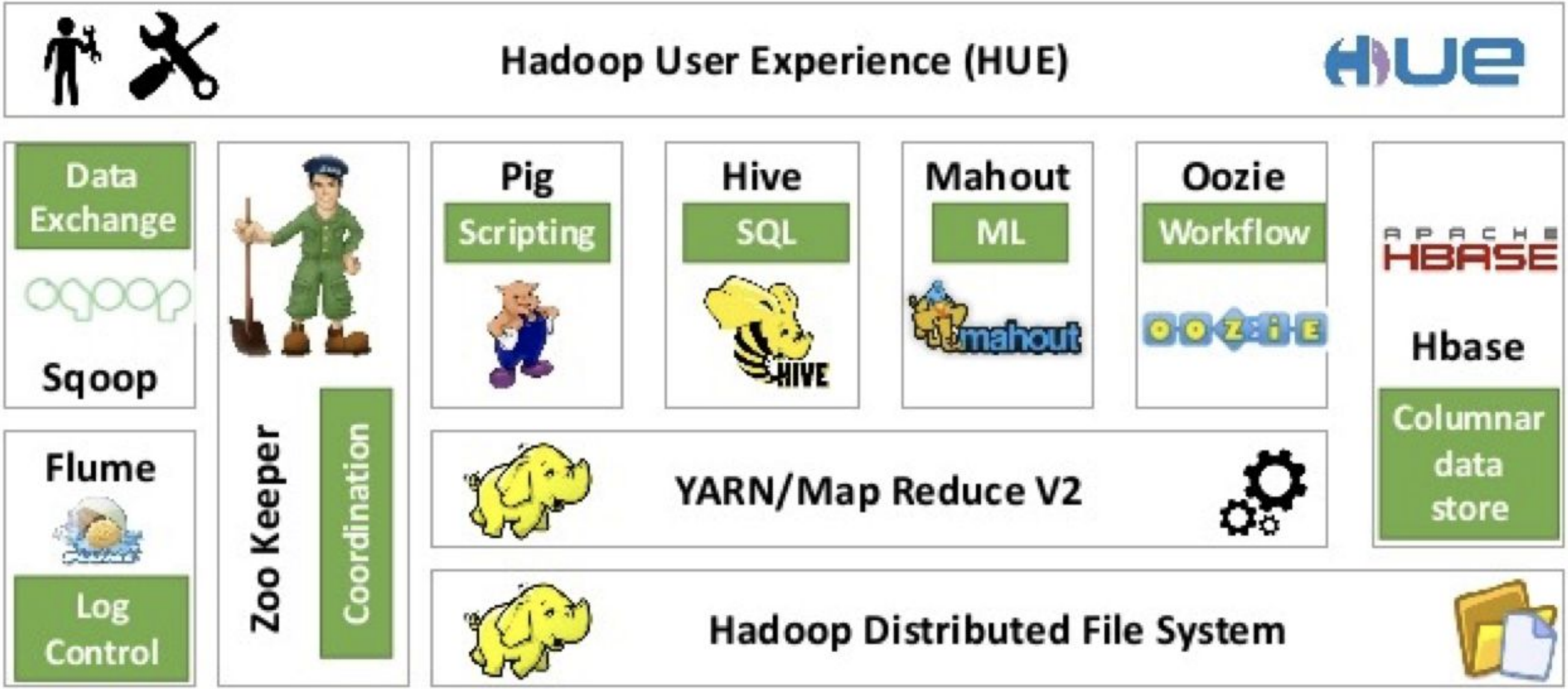
Hadoop

- **HDFS:** Distributed file system for storing large data files
    - Fault tolerance via replication
- **MapReduce:** Distributed data processing via map and reduce operations over large datasets
- **Hive/Pig:** Abstractions on top of MR to make programming more productive

Spark

- Alternative to MapReduce
- Good support for iterative applications
- Fault tolerance via lineage graphs

The Apache Hadoop Stack



Hadoop User Experience (HUE)

Data Exchange

Sqoop

Flume

Log Control

Zoo Keeper

Coordination

Pig

Scripting

Hive

SQL

Mahout

ML

Oozie

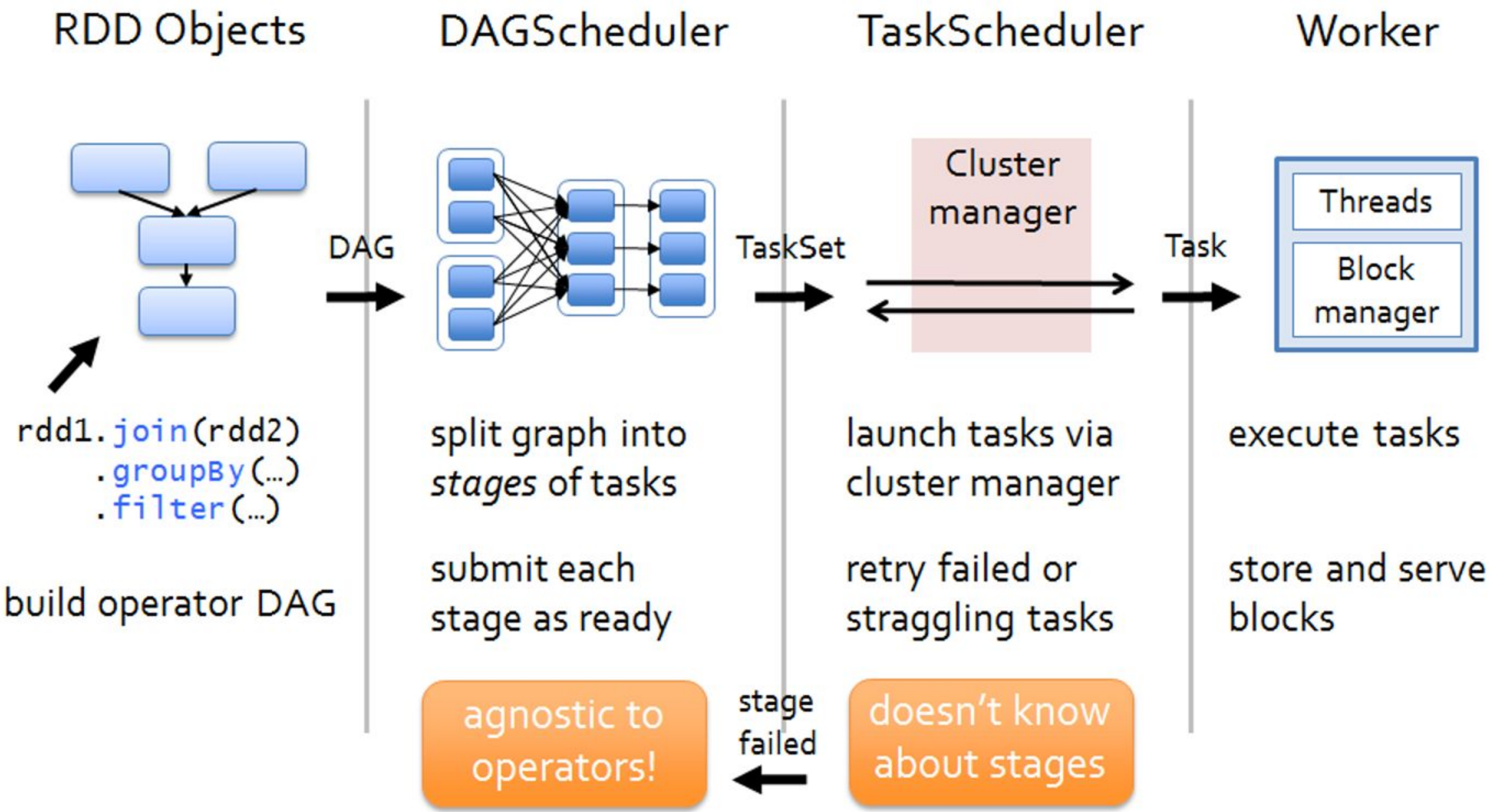
Workflow

Hbase

Columnar data store

YARN/Map Reduce V2

Hadoop Distributed File System



RDD Objects

```
rdd1.join(rdd2)
    .groupBy(…)
    .filter(…)
```

build operator DAG

DAGScheduler

split graph into *stages* of tasks

submit each stage as ready

agnostic to operators!

TaskScheduler

launch tasks via cluster manager

retry failed or straggling tasks

doesn't know about stages

Worker

execute tasks

store and serve blocks

Fault Tolerance

Remember: As we run on larger and larger machines, the probability that something will fail approaches 100%

Our systems must have some way of tolerating faults!

Hadoop/HDFS
Resilience by *replicating data*
If a node/rack goes down, the data can still be read from a replica

Spark
Resilience by *recomputation*
If data is lost, we have the chain of operations needed to recompute it

Phase 3:
Make your analysis accessible!

Bringing it all together

Finally, we bring it all together in a data product

Remember the goal: convey the intelligence your models can glean from the data, without requiring the end users to be data science experts. Make the power of data intensive computing accessible!

But also…consider the possible implications your product will have

Ethics in DIC: Biases in Data

- **Historical biases** in society are present in your data
- **Representation biases:** certain groups in the population are not fairly represented in your datasets
- **Measurement biases:** the variables you use do not match the variables you actually want to capture
- **Aggregation biases:** a "one-size-fits-all" model may not work across multiple diverse groups
- **Evaluation biases:** evaluation metrics may ignore certain groups
- **Deployment biases:** assumptions you make during development may not match conditions in the real world

Moving Forward

Technology is always advancing…

Tools and methods will change and evolve…

We CANNOT teach you every framework, model, or tool you may need

YOU need to be ready to evolve and learn as well