USC Viterbi School of Engineering

INF 553: Foundations and Applications of Data Mining

Dr. Anna Farzindar

farzinda@usc.edu

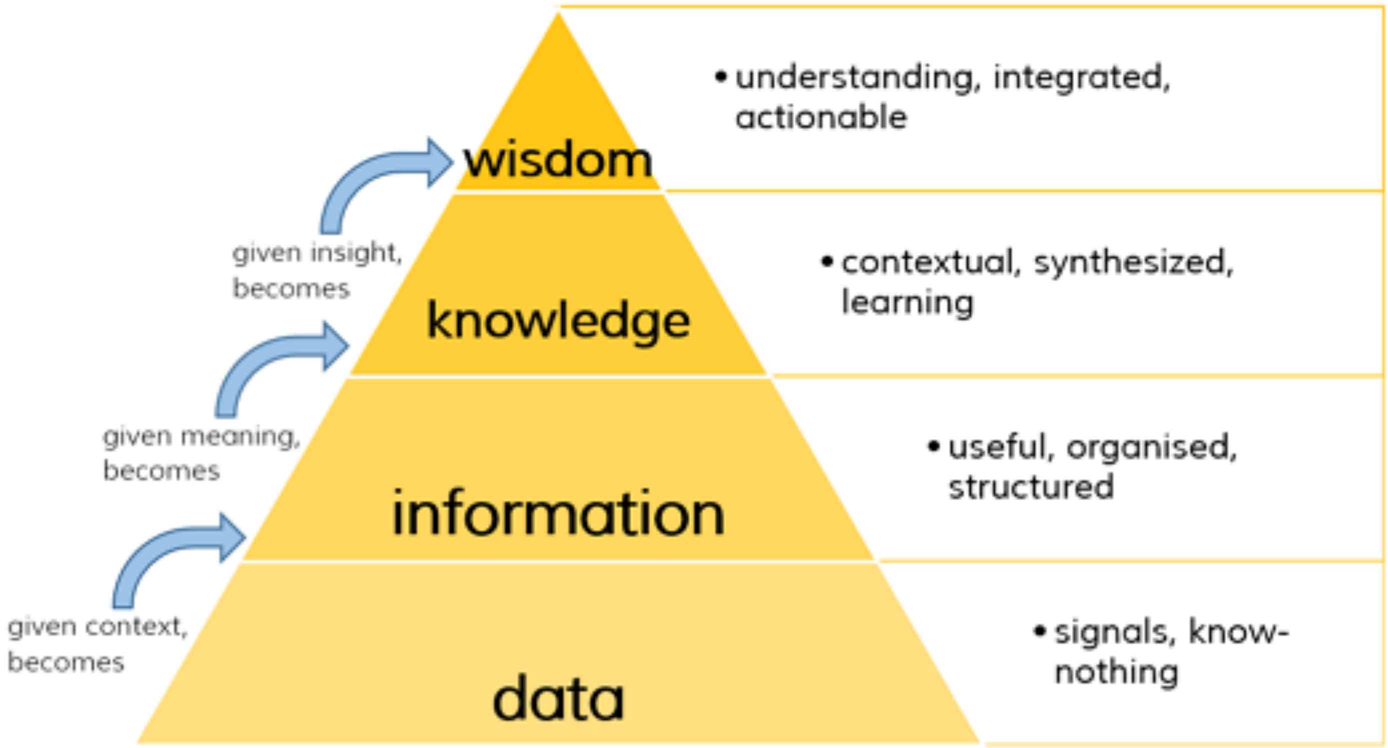
Basic Course Information

- **Lectures**
  - Mon. & Wed.
- **Instructor**
  - Dr. Anna Farzindar, farzinda@usc.edu
    - Office Hours: Monday before class, by appointment, Office: PHE 310
- **Grader/Course Supervisor**
  - Course producers: Youanbin Cheng yuanbinc@usc.edu and Anirudh Kashi kashia@usc.edu, Prasad Bhagwat pbhagwat@usc.edu
  - Graders: Rasika Guru rguru@usc.edu and Roshani Mangalore rmangalo@usc.edu

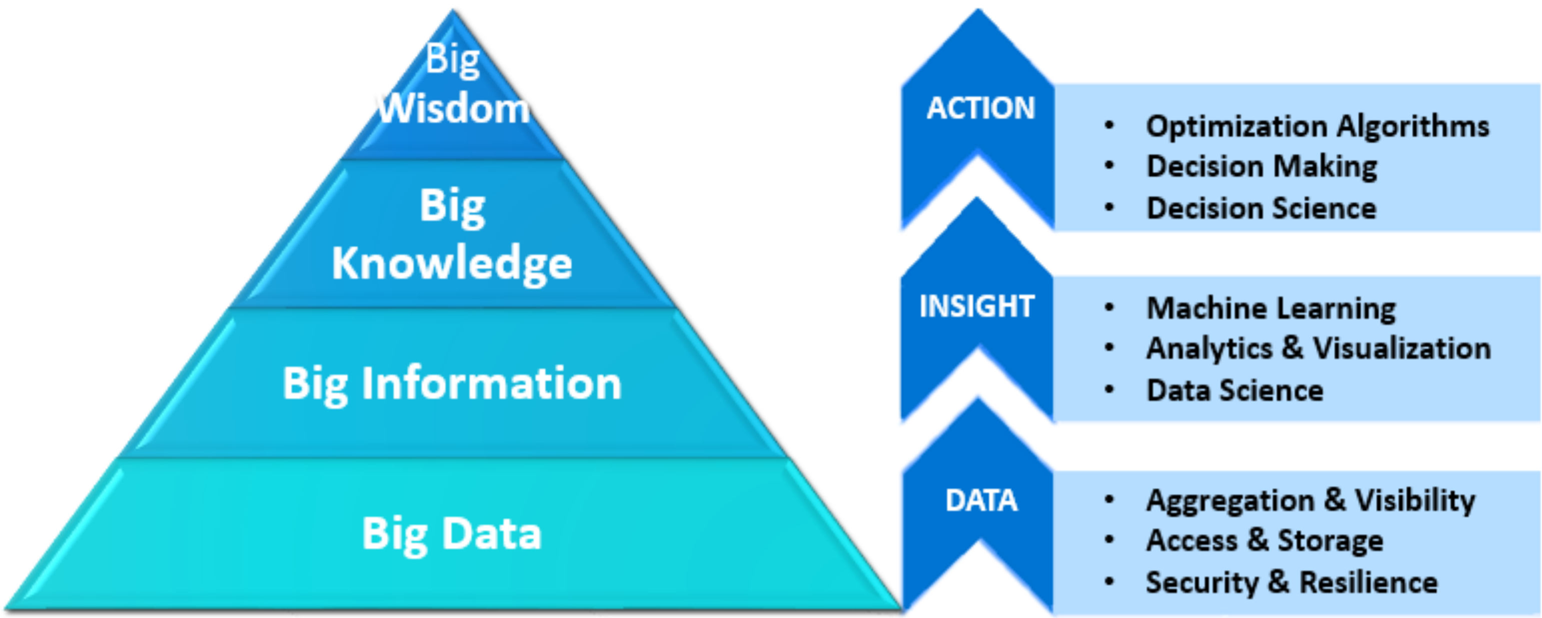
What This Course Is About

- *Data mining* = extraction of **actionable information** from (usually) very large datasets, is the subject of extreme hype, fear, and interest
- It's not all about machine learning
- But a lot of it is.

Data, Information, Knowledge and wisdom (DIKW Pyramid)



Big Data and Data mining



Modeling

- Often, especially for ML-type algorithms, the result is a *model* = a simple **representation** of the data, typically used for **prediction**.
- Example: PageRank is a number Google assigns to each Web page, representing the "**importance**" of the page.
  - **Calculated** from the link structure of the Web
  - **Summarizes** in one number, all the links leading to one page
  - **Used to help decide** which pages Google shows you.

Rules Versus Models

- In many applications, all we want is an algorithm that will say **"yes" or "no"**
- **Example:** a model for **email spam** based on weighted occurrences of words or phrases
  - Would give high weight to words like "**Lottery**" or phrases like "**Nigerian Inheritance**"
- **Problem:** when the weights are in favor of spam, there is no obvious reason why it is spam
  - Sometimes, no one cares; other times understanding is vital.

Rules – (2)

- Rules like “Nigerian Inheritance” -> spam are understandable and actionable
- But the downside is that every email with that phrase will be considered spam
- Next lecture will talk about these *Association Rules*, and how they are used in managing (brick and mortar) stores, where understanding the meaning of a rule is essential.

Outline of Course

- Map-Reduce and Hadoop
- Finding similar sets
  - Minhashing, Locality-Sensitive hashing
- Recommendation systems
  - Collaborative filtering
- Association rules, frequent itemsets
- PageRank and related measures of importance on the Web (link analysis)
  - Spam detection
  - Topic-specific search.

Outline (cont.)

- Extracting structured data (relations) from the Web
- Clustering data
- Managing Web advertisements
- Mining data streams.

Prerequisites

- A basic understanding of engineering principles
- Programming skills
  - Familiarity with the **Scala** and **Python** language is desirable
  - Code Academy Python tutorials: http://www.codecademy.com/tracks/python
  - Google Python Class: https://developers.google.com/edu/python/,
  - Apache **SPARK**
    - An open source big data processing framework built around speed, ease of use, and sophisticated analytics
- Most assignments are designed for the Unix environment
  - Basic Unix skills will make programming assignments easier
- Mathematical background: probability, statistics, and linear algebra
- Some knowledge of machine learning is helpful.

Class Communication and Collaborative Learning

- Blackboard at USC will be used for most class communication
  - assigning and submitting homework
  - posting lecture slides
  - Posting some grades
  - Discussion forum
  - Students are *strongly* encouraged to post questions and respond to other students' postings
  - Active participation can help those students with borderline final grade.

Textbook

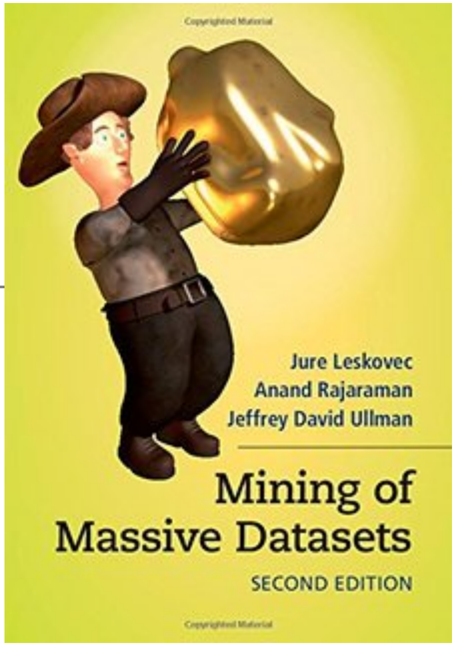
- Rajaraman, J. Leskovec and J. D. Ullman

Mining of Massive Datasets

Cambridge University Press, 2012

- Available free online at:

http://infolab.stanford.edu/~ullman/mmds/book.pdf



- In addition to the textbook, students may be given additional reading materials such as research papers.
- Students are responsible for all assigned reading assignments.

Course Grading

Grading for the course will be based on student performance on:

- Programming assignments (5 assignments during the semester), Possible additional homework assignments
- **Hackathon Competition!!!**
- No midterm examination
- Comprehensive exam Nov 28th, 2018 Wed – 2:00-3:50pm
- Hackathon Competition e-presentation Friday Dec 7th or Mon Dec 10th.
- Class participation, Weekly qizzes & activity on class discussion forums
  - Subscribe to Blackboard -Disscussion Forum

Grading Allocations

- Programming Assignments, Homework: 45%
- weekly quizzes: 30%
- Final: 25%

Total	**100%**

Grading Scale

- 94 – 100 = A
- 90 – 93 = A-
- 87 – 89 = B+
- 84 – 86 = B
- 80 – 83 = B-
- 77 – 79 =C+

74 - 76 = C

70 - 73 = C-

67 - 69 = D+

64 - 66 = D

60 - 63 = D-

Below 60 is an F

Programming Assignments

- All homework assignments are to be submitted to BlackBoard
- To obtain maximum points on the homework assignment, follow the **assignment guideline** and grading rubric carefully
- **Late Work**
  - For each day the homework assignment is late, the student will lose 1/3 of the grade for the assignment.
- In extenuating circumstances, such as a serious medical ailment or a family emergency, students must communicate and make arrangement with the instructors **in advance**
- In case of a serious medical ailment, an original doctor's note must accompany the late submission.

Grading Corrections

ATTENTION!

- **Grades are not negotiable**
- Any student who wastes the instructor's time with non-legitimate requests for additional points on an assignment or exams risks losing additional points as well as having their behavior affect their class participation grades

- Any legitimate request for re-grading **must be submitted in writing, with carefully worked out explanation** of why it is believed that an assignment has not been properly graded.

Academic Integrity

- **Cheating will not be tolerated**
- **All parties involved will receive**

a grade of F for the course and be reported to SJACS (WITHOUT EXCEPTION)

- It is fine to answer questions from other students on the class discussion board, but DO NOT post your solution to an assignment
- **We will be using the Moss system to detect software plagiarism**

  http://theory.stanford.edu/~aiken/moss/

- If you have questions or concerns regarding what is permitted in terms of collaboration or teamwork, please ask the instructor/grader for clarifications.

Example Moss Output

```python
import sys

mr = MapReduce.MapReduce()

def mapper(record):
    # key: name of the matrix
    # value: location and value of element
    key = record[0]  #matrix A or B

    if key=='a': #check the name of matrix
        mr.emit_intermediate(key, [record[1],record[2],record[3]])    #emit matrix name as key and location,value of element as value


    if key=='b':
           mr.emit_intermediate(key, [record[2],record[1],record[3]]) #i have scanned matrix b vertically

def reducer(key, list_of_values):
    #key: name of matrix
    #list_of_values: element location and its value
    matrix_a={}
    matrix_b={}
    if key=='a':    #if this condition is not checked, then the multiplication executes twice, once for 'a' and once for 'b'
        for a_values in mr.intermediate['a']:   #access the intermediate dictionary with key 'a'


            matrix_a[(a_values[0], a_values[1])]=a_values[2]

        for b_values in mr.intermediate['b']:
            matrix_b[(b_values[0], b_values[1])]=b_values[2]

        #considering matrices to be sparse, fill the missing values with zero
        for i in range(5):
            for j in range(5):
                    if (i,j) not in matrix_a:
                        matrix_a[(i,j)]=0
        for j in range(5):
            for i in range(5):
                    if (i,j) not in matrix_b:
                        matrix_b[(i,j)]=0
        result=0

        for i in range(5):
             for k in range(5):
                  for j in range(5):


                       result=result+(matrix_a[(i,j)]*matrix_b[(k,j)])
                  mr.emit((i,k,result))
                  result=0

if __name__ == '__main__':
  inputdata = open(sys.argv[1])
  mr.execute(inputdata, mapper, reducer)
```

```python
import sys

"""
Word Count Example in the Simple Python MapReduce Framework
"""

mr = MapReduce.MapReduce()

# =============================
# Do not modify above this line

def mapper(record): # receive record as the matrix values in input file
      mat = record[0] # assign mat as the type 'a' or 'b' matrix
      if mat == 'a': # check if the input data is of matrix 'a' else 'b'
           mr.emit_intermediate(mat, [record[1],record[2],record[3]]) # pass mat 'a' as key and it's i,j,number data as value


      else:
           mr.emit_intermediate(mat, [record[2],record[1],record[3]]) # pass mat 'b' as key and it's j,i,number data as value

def reducer(mat, values):
      mata={} # create dictionary for matrix a
      matb={} # create dictionary for matrix b
      if mat == 'a': # check for matrix to be a
           # populate both dictionaries with known values
           for value in values:
              for value in values:


                  mata[(value[0], value[1])] = value[2]
              for value in mr.intermediate['b']:
                  matb[(value[0], value[1])] = value[2]

           # fill in '0' for missing numbers from input file
           for i in range(0,5):
                for j in range(0,5):
                     if (i,j) not in mata.keys():
                          mata[(i,j)] = 0
                     if (j,i) not in matb.keys():
                          matb[(j,i)] = 0

           # multiply the matrices to emit the result
           matres = 0
           for i in range(0,5):
                for j in range(0,5):
                     for k in range(0,5):


                          matres+= mata[(i,k)] * matb[(j,k)]
                     mr.emit((i,j,matres))
                     matres = 0

# Do not modify below this line
# =============================
if __name__ == '__main__':
  inputdata = open(sys.argv[1])
```

What is Data Mining?

Knowledge Discovery from Data

Thanks for source slides and material to: J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

$600 to buy a disk drive that can store all of the world's music

5 billion mobile phones in use in 2010

30 billion pieces of content shared on Facebook every month

40% projected growth in global data generated per year vs. **5%** growth in global IT spending

$5 million vs. $400
Price of the fastest supercomputer in 1975[1] and an iPhone 4 with equal performance

235 terabytes data collected by the US Library of Congress by April 2011

15 out of 17 sectors in the United States have more data stored per company than the US Library of Congress

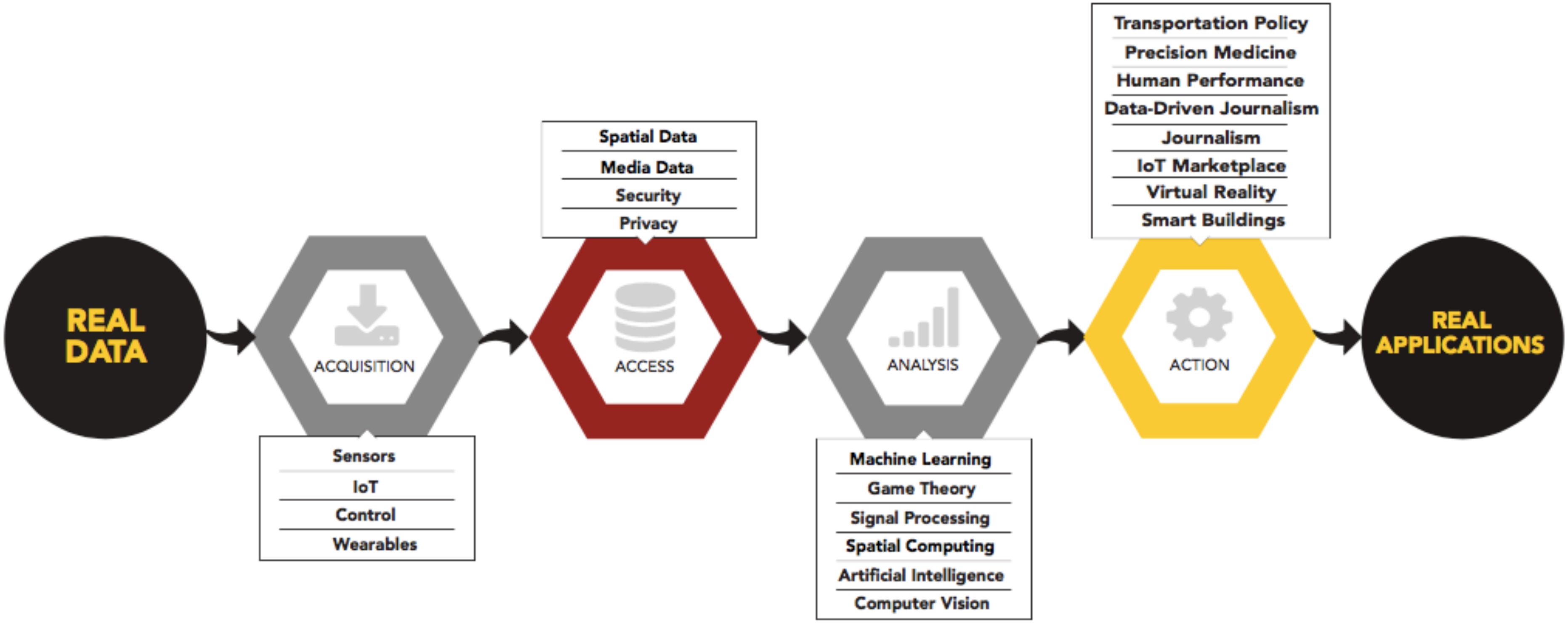
Data contains value and knowledge

Data Mining

- But to extract the knowledge data needs to be
  - Stored
  - Managed
  - And ANALYZED ← this class

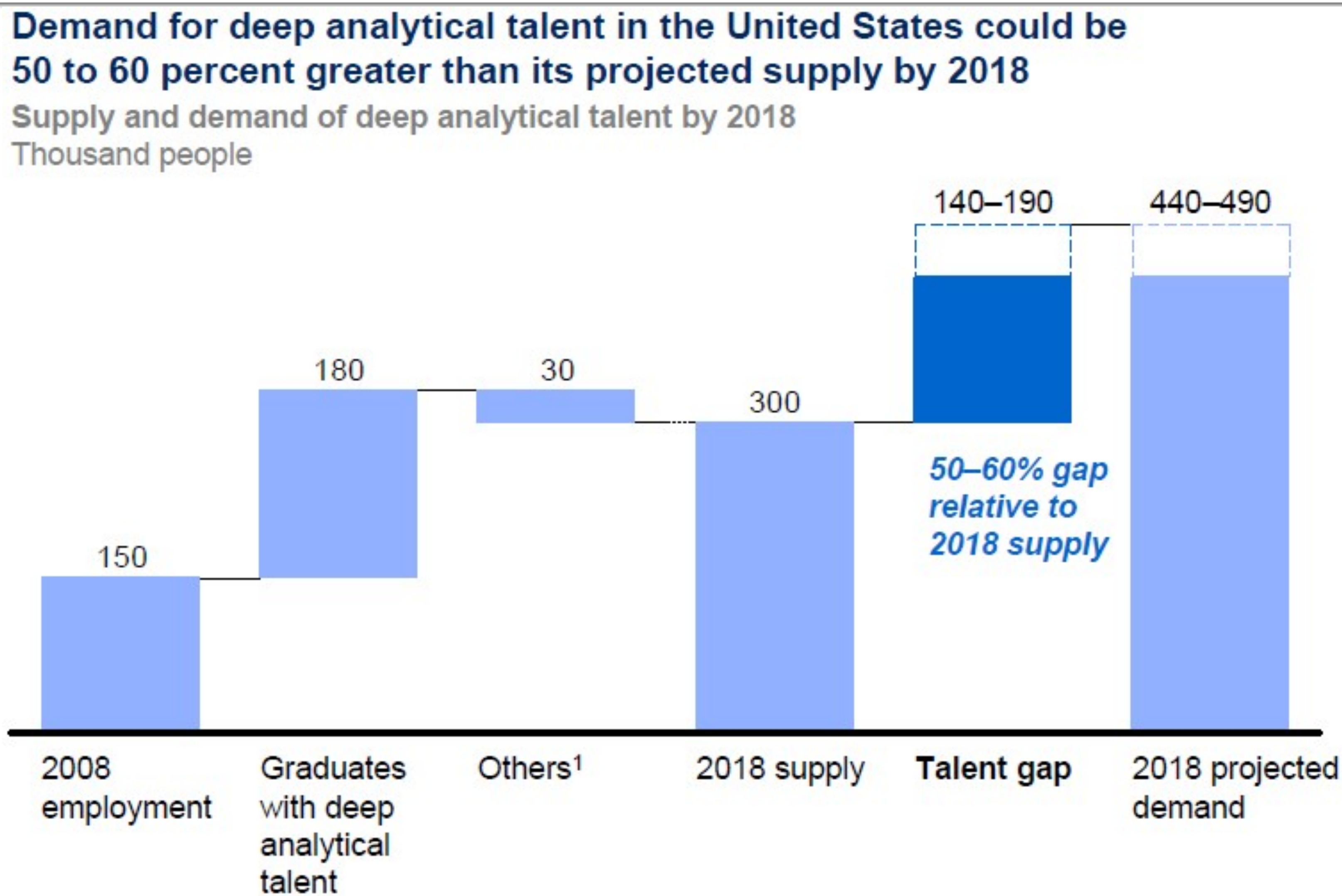
Data Mining ≈ Big Data ≈ Predictive Analytics ≈ Data Science

The A4 Data Science Pipeline



https://imsc.usc.edu/

Good news: Demand for Data Mining



1 Other supply drivers include attrition (-), immigration (+), and reemploying previously unemployed deep analytical talent (+).

SOURCE: US Bureau of Labor Statistics; US Census; Dun & Bradstreet; company interviews; McKinsey Global Institute analysis

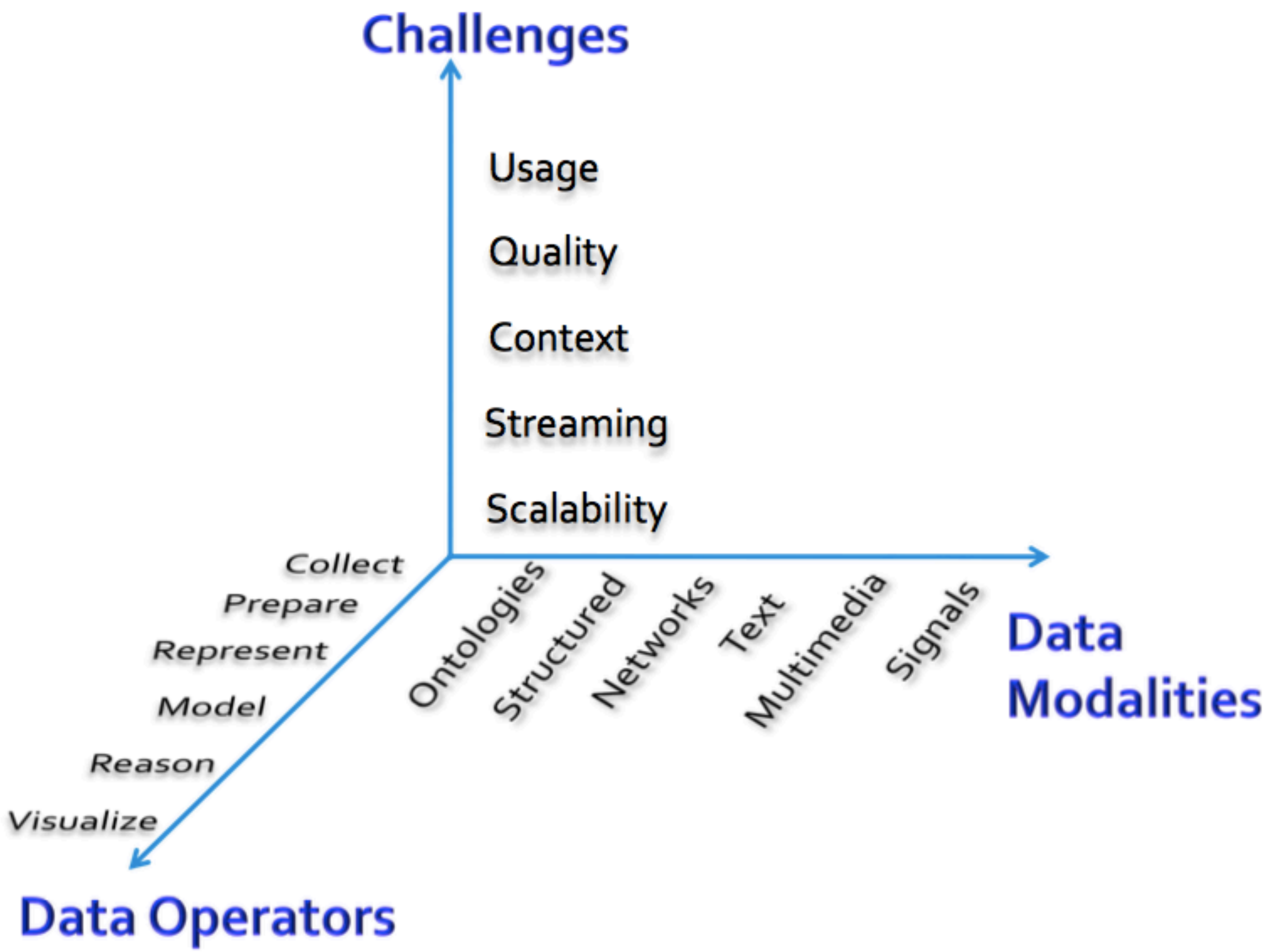
What is Data Mining?

- **Given lots of data**
- **Discover patterns and models that are:**
  - **Valid:** hold on **new data** with some certainty
  - **Useful:** should be possible to **act** on the item
  - **Unexpected:** **non-obvious** to the system
  - **Understandable:** **humans** should be able to **interpret the pattern**.

Data Mining Tasks

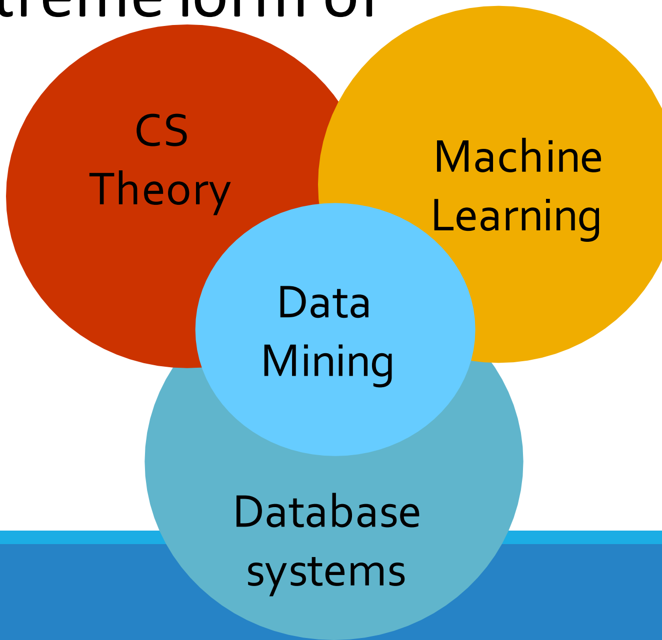
- **1- Descriptive methods**
  - Find human-interpretable patterns that describe the data
    - **Example:** Clustering.
- **2- Predictive methods**
  - Use some variables to predict unknown or future values of other variables
    - **Example:** Recommender systems.

What matters when dealing with data?



Data Mining: Cultures

- **Data mining overlaps with:**
  - **Databases:** Large-scale data, simple queries
  - **Machine learning:** Small data, Complex models
  - **CS Theory:** (Randomized) Algorithms
- **Different cultures:**
  - To a DB person, data mining is an extreme form of **analytic processing** – queries that examine large amounts of data
    - Result is the query answer
  - To a ML person, data-mining is the **inference of models**
    - Result is the parameters of the model.

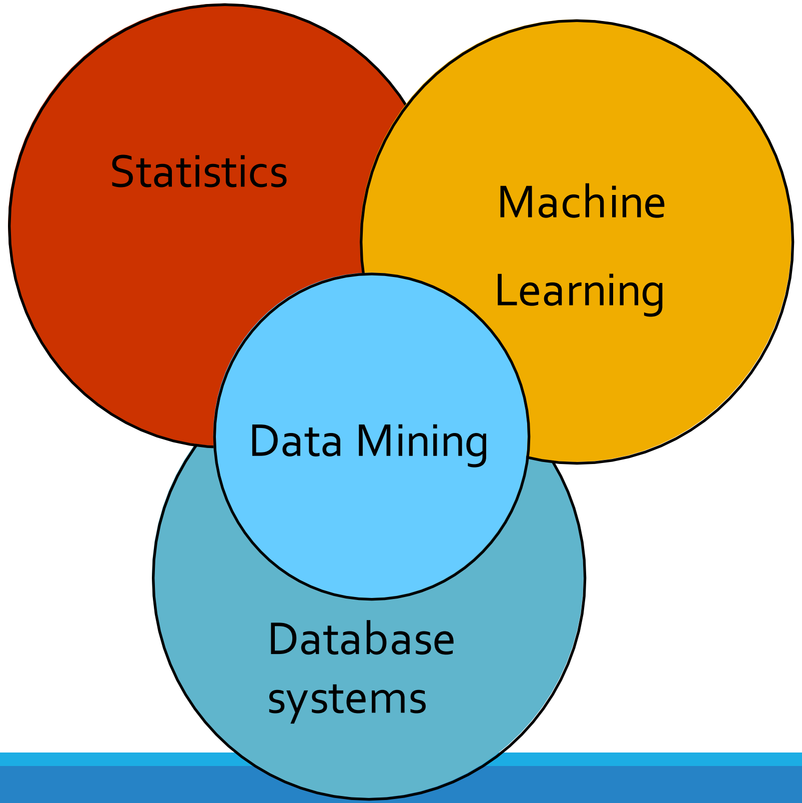


Cultures

- Databases: concentrate on large-scale (non-main-memory) data
  - Computational approach to modeling
  - Model of the data is the answer to a complex query,
- Artificial Intelligence/Machine-learning: concentrate on complex methods, small data
  - Some data mining uses machine learning algorithms
  - Works best when little idea of what **we are looking for** in the data (e.g., algorithm to predict movie ratings for users),
- Statistics: construct a statistical model: an underlying distribution from which visible data are drawn.

This Class

- **This class overlaps with machine learning, statistics, artificial intelligence, databases but more stress on**
  - **Scalability** (big data)
  - **Algorithms**
  - **Computing architectures**
  - Automation for handling **large data.**



What will we learn?

- **We will learn to** mine different types of data:
  - Data is high dimensional
  - Data is a graph
  - Data is infinite/never-ending
  - Data is labeled,
- **We will learn to use different models of computation:**
  - MapReduce
  - Streams and online algorithms
  - Single machine in-memory.

What will we learn?

- **We will learn to solve real-world problems:**
  - Recommender systems
  - Market Basket Analysis
  - Spam detection
  - Duplicate document detection,
- **We will learn various “tools”:**
  - Linear algebra (SVD, Rec. Sys., Communities)
  - Dynamic programming (frequent itemsets)
  - Hashing (LSH, Bloom filters)
  - Optimization (stochastic gradient descent).

How the Class Fits Together

High dim. data	Graph data	Infinite data	Machine learning	Apps
Locality sensitive hashing	PageRank, SimRank	Filtering data streams	SVM	Recommender systems
Clustering	Network Analysis	Web advertising	Decision Trees	Association Rules
Dimensionality reduction	Spam Detection	Queries on streams	Perceptron, kNN	Duplicate document detection

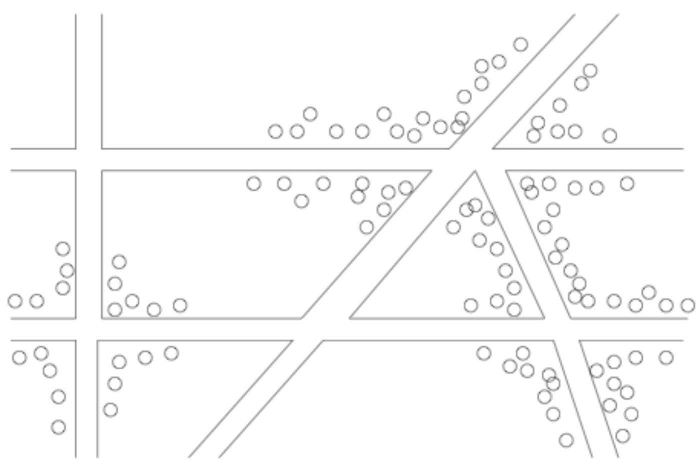
Modeling Data: **Summarization**

- Summarize the data
- A) PageRank (Chapter 5)
  - Structure of the web is **summarized** by a **single number** for each web page (its PageRank)
  - Probability that a random walker on the graph would be on the page at any given time
  - Property: the PageRank reflects the ***importance*** of the page
  - How much a typical **searcher** wants that page returned as **an answer** to their search query.

Modeling Data: Summarization

- B) Clustering (Chapter 7)
  - Data viewed as **points in multidimensional space**
  - Points "close" in space assigned to **same cluster**
  - Clusters are summarized: e.g., by **centroid of cluster** and **average distance** from centroid of points in cluster
  - Cluster summaries become **summary of data set**,

Example: identify clusters of cholera cases around London intersections due to contaminated wells



Modeling Data: Feature Extraction

- A complex **relationship** between objects is represented by finding the **strongest statistical dependencies** among objects and using those to represent statistical connections
- **A) Frequent itemsets** (Chapter 6)
  - Model for data that consists of **"baskets"** of small sets of items (e.g., for brick and mortar stores or shopping sites)
  - Look for small sets of items that **appear together in many baskets**
  - These "frequent itemsets" **characterize** the data
  - Identify sets of items that people tend to **buy together**, can use to set prices,
    - E.g., hamburger and ketchup

Modeling Data:
Feature Extraction

- **B) Similar items** (Chapter 3)
  - Data looks like **a collection of sets**
  - Objective: find **pairs of sets** that have fairly large number of **items in common**
  - E.g. treat customers as **set of items they have bought**, look for similar customers to ***recommend*** additional items those customers have bought
  - Or, for each customer, identify small number of customers with **similar tastes**
  - Recommendation Systems, Chapter 9.

Meaningfulness of Analytic Answers

- A **risk** with "Data mining" is that an analyst can "discover" **patterns that are meaningless**

- Statisticians call it **Bonferroni's principle**:
  - Roughly, if you look in **more places** for interesting patterns than your amount of data will support, **you are bound to find crap.**

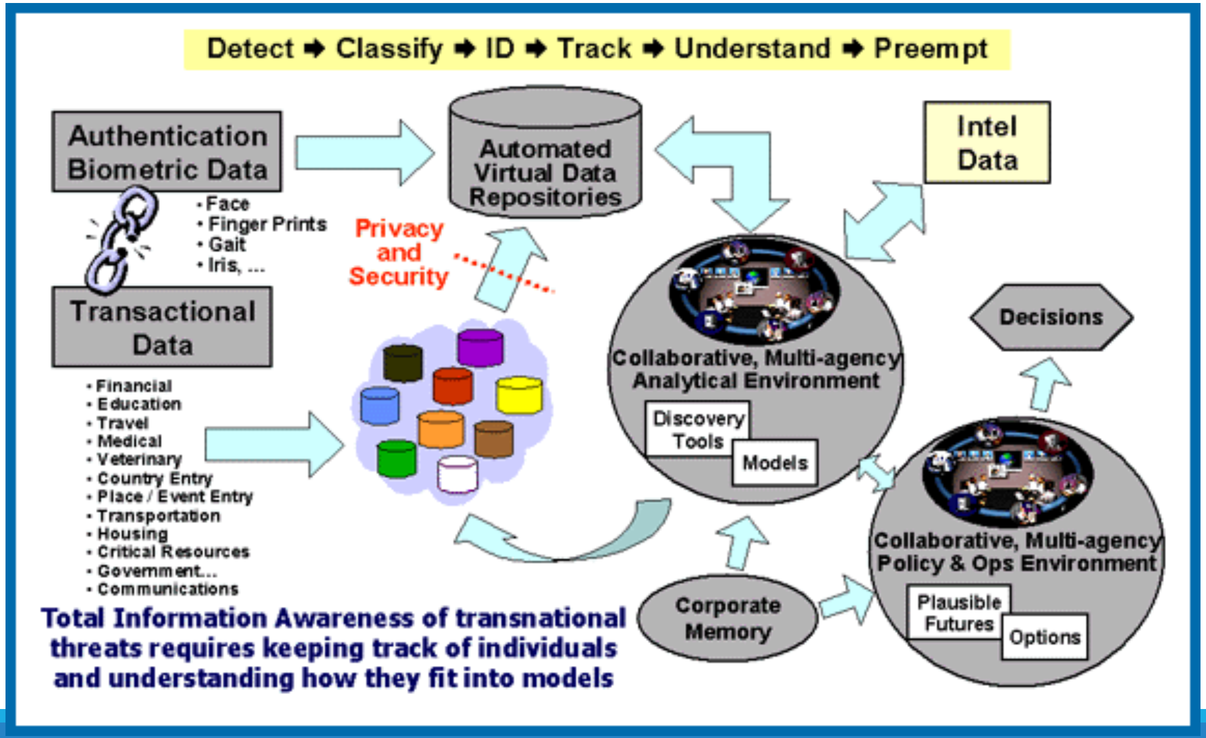
Bonferroni's Principle

- **Bonferroni's Principle** is an informal presentation of a statistical theorem
  - that states if your **method of finding** significant items **returns significantly more items** that you would expect in the actual population,
  - you can assume **most of the items** you find with it are **bogus**.
- This essentially means that an algorithm or method we think is **useful** for finding a particular set of data actually returns more **false positives** as it returns larger portions of the data than should be within that category.

http://rationalwiki.org/wiki/Bonferroni's_principle

Example of Bonferroni's Principle

- **Total Information Awareness** (searching for **suspicious activity**), US IAO 2003
- **Using Predictive policing**: the usage of mathematical, predictive and analytical techniques to identify potential criminal activity.



https://commons.wikimedia.org/wiki/File:Total_Information_Awareness_--_system_diagram.gif

Example of Bonferroni's Principle (Count.)

- A big objection to **Total Information Awareness**
- Was that it was looking for **so many vague connections** that it was sure to find things that were **bogus and thus violate privacy of innocent people.**

Scenario

- Suppose we believe that certain groups of evil-doers are **meeting occasionally in hotels** to plot doing evil.
- We want to find (unrelated) people who at least **twice** have stayed at the **same hotel** on **the same day**.

The Details

- One billion (10^9) people being tracked
- 1,000 days
- Each person stays in a hotel 1% of the time (1 days out of 100).
- Hotels hold 100 people (so 10^5 hotels)
- **If everyone behaves randomly (i.e., no evil-doers) will the data mining detect anything suspicious?**
- **What would be Expected number of "suspicious" pairs of people?**

Calculations

- 10^9 people being tracked
- 1000 days
- Each person stays in a hotel 1% of the time (10 days out of 1000).
- Hotels hold 100 people
- 10^5 hotels to hold 1% of 10^9 people

p at some hotel

q at some hotel

Same hotel

Probability that given persons *p* and *q* will be at the same hotel on given day *d* :

$$\underbrace{\boxed{1/100}}_{p \text{ at some hotel}} \times \underbrace{\boxed{1/100}}_{q \text{ at some hotel}} \times \underbrace{\boxed{10^{-5}}}_{\text{Same hotel}} = 10^{-9}$$

Probability that *p* and *q* will be at the same hotel on given days d_1 and d_2:

- $10^{-9} \times 10^{-9} = 10^{-18}$ **(A)**

Recall: for large n days , $\binom{n}{2}$ is about $n^2/2$

Pairs of days: $\binom{1000}{2}$ is about 5 x 10^5 **(B)**

- 10^9 people being tracked
- 1000 days
- Each person stays in a hotel 1% of the time (10 days out of 1000).
- Hotels hold 100 people
- 10^5 hotels to hold 1% of 10^9 people

Calculations

- Probability that *p* and *q* will be at the same hotel on some two days:
  - (number of pairs of days) x (prob p and q at same hotel for 2 days)
  - **(B) X (A)** from previous slide
  - $(5\times10^5) \times 10^{-18} = 5\times10^{-13}$ **(C)**
- Pairs of people:
  - 10^9 people
  - About $n^2/2$ pairs of people: 5×10^{17} **(D)**
- Expected number of **"suspicious"** pairs of people:
  - **(C)** X **(D)**
  - $5\times10^{17} \times 5\times10^{-13} = 250{,}000$

Conclusion

- Suppose there are (say) **10 pairs of evil-doers** who definitely stayed at the same hotel twice,
- Analysts have to sift through **250,000 (a quarter of million) candidates** to find the 10 real cases
  - Not realistic
- **Expected number of "suspicious" pairs of people:**
- 250,000
- ... too many combinatons to check – we need to have some additonal evidence to find "suspicious" pairs of people in some more efficient way.

Moral

- When looking for a property (e.g., "two people stayed at the same hotel twice"), make sure that the property does not allow so many possibilities that random data will surely produce facts "of interest."