Fall 2018: Introduction to Data Science

GIRI NARASIMHAN, SCIS, FIU

Data Wrangling

Database *Join* (Python merge)

```
unames = ['user_id', 'gender', 'age', 'occupation', 'zip']
users = pd.read_table('data/ml-1m/users.dat', sep='::', header=None,
                      names=unames, engine='python')
rnames = ['user_id', 'movie_id', 'rating', 'timestamp']
ratings = pd.read_table('data/ml-1m/ratings.dat', sep='::', header=None,
                        names=rnames, engine='python')
mnames = ['movie_id', 'title', 'genres']
movies = pd.read_table('data/ml-1m/movies.dat', sep='::', header=None,
                       names=mnames, engine='python')
```

Statistical Preliminaries

- Notes available on Course webpage
- Data Mining: task of **modeling** the data?
  - Discrete distributions
    - **Uniform** – equiprobable events
    - **Binomial** – number of successes in independent trials
    - **Negative binomial** – number of successes to have m successes
    - **Geometric** – number of successes before failure
    - **Poisson** – limiting form of binomial (large n and small p); # of events in a range

Continuous Distributions

- **Normal**, **Gaussian** – bell curve shaped distribution
- **Exponential** – generalizes geometric
- **Gamma** – generalizes exponential, sum of Poisson distributions
- **Beta** – generalizes uniform distribution

Law of Large Numbers

- Say, we want average rating for the movie "Blackkklansman"
- Cannot ask everyone who saw the movie; so survey a sample cohort
- Compute a sample average and report it?
- How much off are we from the true average?
- Law of large numbers:

$$\text{Prob}\left(\left|\frac{x_1 + x_2 + \cdots + x_n}{n} - E(x)\right| > \epsilon\right) \leq \frac{\text{Var}(x)}{n\epsilon^2}.$$

- Proof uses Markov's and Chebyshev's Inequalities

Case History: Believable Hunches

- Say, we want to identify a member of a terrorist cell
- Say, we have a hunch that they regularly meet at some hotel in city
- Population = 1B; # of hotels = 100,000
- Every person visits a hotel once every 100 days
- Every hotel holds 100 people;
- We have hotel data on 1000 days
- Look for terrorists that were at same hotel on same day at least twice
- If we found a pair of terrorists, how sure are we?

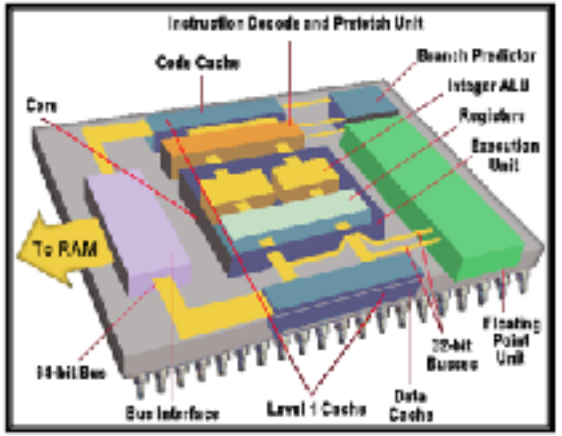
Bonferroni Principle

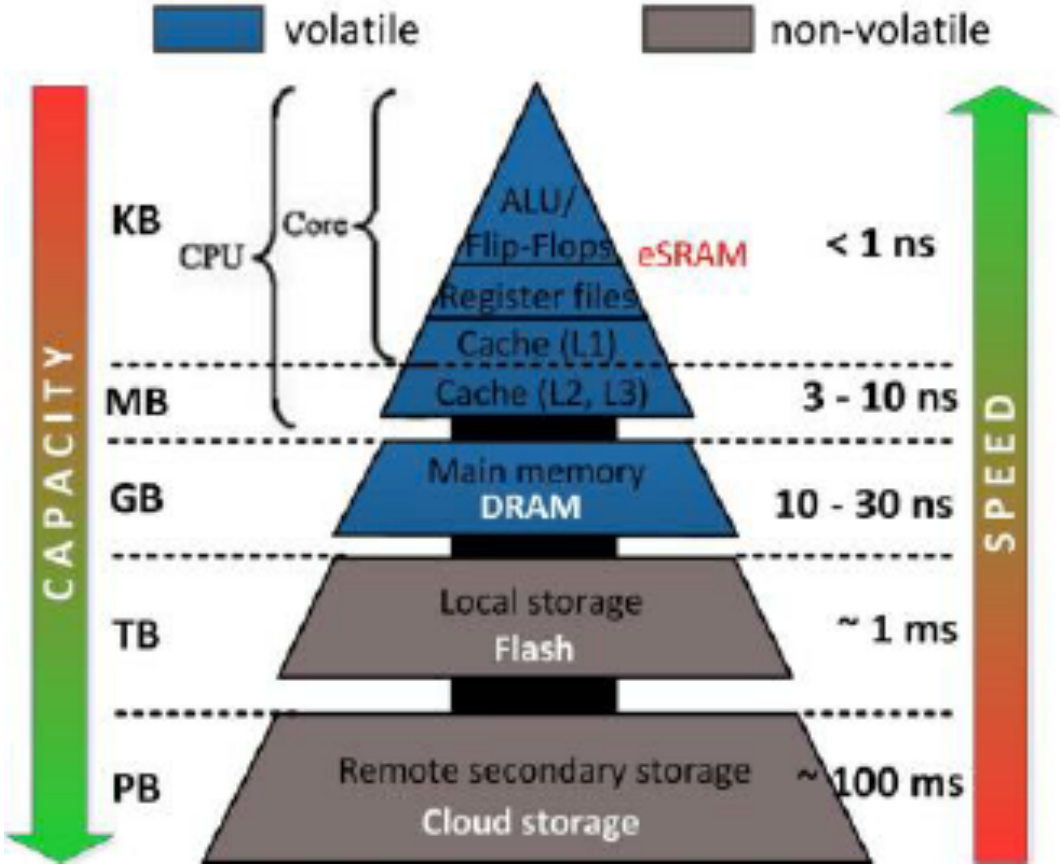
- **Population** = 1B; **# of hotels** = 100,000; Hotel visit = once every 100 days
- Every hotel holds 100 people; We have hotel data on 1000 days
- Prob { X and Y going on hotel trip } = 10^{-2} X 10^{-2}
- Prob { X and Y going on hotel trip to same hotel } = 10^{-4} / 10^{5} = 10^{-9}
- Prob { X and Y going to same hotel on 2 days } = 10^{-9} X 10^{-9} = 10^{-18}
- # of random terrorist-like events = # of pairs of people X # of pairs of days X probability of terrorist-like behavior

$$5 \times 10^{17} \times 5 \times 10^{5} \times 10^{-18} = 250,000$$

Memory Hierarchy in Computers

- Actual computation happens on a CPU
  - Fastest when the operands are in registers
- Data and programs are in main memory
  - Frequent items can be found in cache
- Beyond MM are 2° and 3° storage
  - Disk
  - Flash
  - Magnetic tapes
  - cloud





How are x and y related?

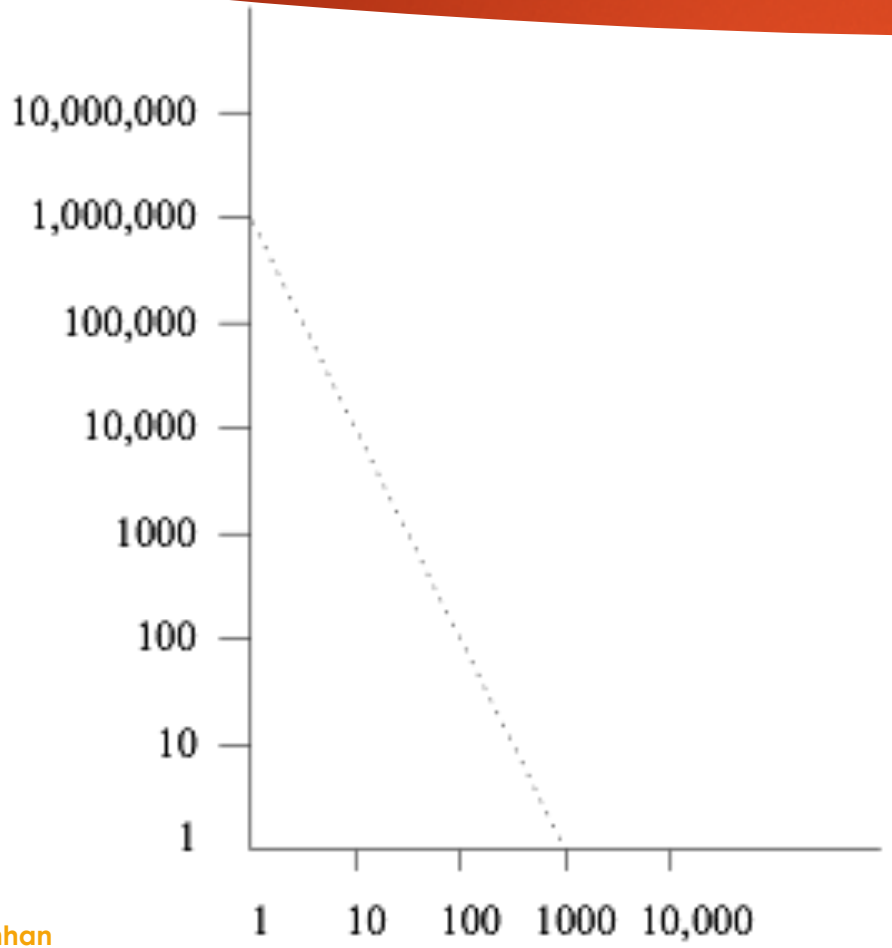


Figure 1.3: A power law with a slope of – 2

$$\log_{10} y = 6 - 2 \log_{10} x$$

Power Laws

The Matthew Effect

Often, the existence of power laws with values of the exponent higher than 1 are explained by the Matthew effect. In the biblical Book of Matthew, there is a verse about "the rich get richer." Many phenomena exhibit this behavior, where getting a high value of some property causes that very property to increase. For example, if a Web page has many links in, then people are more likely to find the page and may choose to link to it from one of their pages as well. As another example, if a book is selling well on Amazon, then it is likely to be advertised when customers go to the Amazon site. Some of these people will choose to buy the book as well, thus increasing the sales of this book.

"Wordle" Images capture documents

Higher Education

World War

Physics of Motion

Text Mining: Characterize Documents

- What words characterize the topic of a document? TF-IDF
- Finding words with high frequency is not enough (e.g., the, and, etc.)
- Finding rare words is not enough either (e.g., albeit, consequently, etc.)
- TF: Term Frequency of word i in document j
  - Normalized by largest frequency
- IDF: Inverse Document Frequency for term i =
  - Assume term i appears in n_i of N documents

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

$$IDF_i = \log_2(N/n_i)$$

TF-IDF example

- Repository size = 2^{20} = 1,048,576 documents
- Word w appears in 2^{10} documents
  - IDF = 10
- Word w appears 20 times in document j & max = 200
  - TF = 0.1
- TF.IDF = 1
- If word appears in $2^{19.5}$ documents and 200 times in document j
  - TF.IDF = 0.5

Case History: Using TF-IDF with LDA

- Read https://towardsdatascience.com/topic-modeling-and-latent-dirichlet-allocation-in-python-9bf156893c24
- Can you find and download the Kaggle dataset?
- Follow the code for preprocessing after loading libraries
  - Tokenization, filter, stopwords, lemmatized, stemmed
- Perform further filtering (15 < n < N) using filter extremes
- Compute TF-IDF
- Run LDA using Bag of Words & measure performance