Big Data Analytics: Introduction

Alexander Schönhuth

UNIVERSITÄT BIELEFELD
Faculty of Technology

Bielefeld University
April 15, 2021

Learning Goals Today

- None of today's topics plays an explicit role in assignments/exercises or the exam
- But they may reappear in other topics, and then play an implicit role
- Goal today is to get fundamental ideas about the following crucial topics

Organizational matters

What is Data Mining?

Statistical Limits

Useful Things

LEARNING GOALS TODAY

- *Organization:*
  - How do lectures, tutorials etc work
  - What tools will be used
- What does *Data Mining* mean? What is the meaning of
  - Statistical/Computational Modeling
  - Summarization
  - Feature Extraction
- What are *Statistical Limits on Data Mining*
  - Bonferroni's Principle
- Which are *Useful Things to Know*
  - Word importance (example): the TD.IDF measure
  - Hash functions
  - Secondary storage and the effects on runtime
  - The natural logarithm and important identities based on it
  - Power laws

Learning Goals Today

- *Organization:*
  - How do lectures, tutorials etc work
  - What tools will be used
- What does *Data Mining* mean? What is the meaning of
  - Statistical/Computational Modeling
  - Summarization
  - Feature Extraction
- What are *Statistical Limits on Data Mining*
  - Bonferroni's Principle
- Which are *Useful Things to Know*
  - Word importance (example): the TD.IDF measure
  - Hash functions
  - Secondary storage and the effects on runtime
  - The natural logarithm and important identities based on it
  - Power laws

Learning Goals Today

- *Organization:*
  - How do lectures, tutorials etc work
  - What tools will be used
- What does *Data Mining* mean? What is the meaning of
  - Statistical/Computational Modeling
  - Summarization
  - Feature Extraction
- What are *Statistical Limits on Data Mining*
  - Bonferroni's Principle
- Which are *Useful Things to Know*
  - Word importance (example): the TD.IDF measure
  - Hash functions
  - Secondary storage and the effects on runtime
  - The natural logarithm and important identities based on it
  - Power laws

Learning Goals Today

- *Organization:*
  - How do lectures, tutorials etc work
  - What tools will be used
- What does *Data Mining* mean? What is the meaning of
  - Statistical/Computational Modeling
  - Summarization
  - Feature Extraction
- What are *Statistical Limits on Data Mining*
  - Bonferroni's Principle
- Which are *Useful Things to Know*
  - Word importance (example): the TD.IDF measure
  - Hash functions
  - Secondary storage and the effects on runtime
  - The natural logarithm and important identities based on it
  - Power laws

Organizational matters

What is Data Mining?

Statistical Limits

Useful Things

Prerequisites, Lectures, Exercises

- Course prerequisites: Databases I (Datenbanken I)
- Lectures: Thursdays, 10-12, via Zoom meetings as per links provided
- Exercises: 5 assignments + 1 exam preparation session

Prerequisites, Lectures, Exercises

- Course prerequisites: Databases I (Datenbanken I)
- Lectures: Thursdays, 10-12, via Zoom meetings as per links provided
- Exercises: 5 assignments + 1 exam preparation session

Prerequisites, Lectures, Exercises

- Course prerequisites: Databases I (Datenbanken I)
- Lectures: Thursdays, 10-12, via Zoom meetings as per links provided
- Exercises: 5 assignments + 1 exam preparation session

Assignments, Exam

- *Tutorials/Assignments:*
  - New exercise sheets provided on Thursdays April 22, May 6, May 27, June 10, June 24, July 15) after the lecture
  - Exercises to be submitted by Tuesday, **23:59** twelve days thereafter, discussion on Wednesday, Thursday same week
  - Submission of exercises in groups of (approximately) 5 people
  - Each group is supposed to present one exercise sheet in one of the tutorials (ideal scenario)
  - Upload to corresponding folder in the "Lernraum Plus"
  - First exercise sheet uploaded on 22nd of April (next week)
- *Exam:*
  - Likely online exam, planned for **Thursday, July 29, 2021 10:00-12:00** (may be subject to changes due to situation; we will communicate changes as timely as possible)
  - Admitted: everyone exceeding 50% of total exercise points

TUTORIALS

- Every **Wednesday, 16-18** and **Thursday, 16-18**
- 4 tutorials, 3 tutors: Maren Knop, Swen Simon and Harsha Manjunath
- Assignment of people to the 4 tutorials via Lernraum Plus (details will follow soon)
- One tutorial per day (Wednesday or Thursday) in English, the other one in German (ideal scenario)
- Zoom meetings, link will be provided in time
- Presentation of individual solutions during the online meeting, by groups of 2-3 people

Course material

- ... available on course website: `https://gds.techfak.uni-bielefeld.de/teaching/2021summer/bda`
  - Slides and pointers to literature
  - Excercise sheets
- Lernraum Plus: `https://lernraumplus.uni-bielefeld.de/course/view.php?id=9839`
  - Submission of exercise solutions
  - Self-managed forum

LITERATURE AND LINKS

- Jure Leskovec, Anand Rajaraman, Jeffrey David Ullman (2019). *Mining of Massive Datasets*. 3rd Edition, Cambridge University Press.
- *Download:* `http://infolab.stanford.edu/~ullman/mmds/book0n.pdf`
- *Materials:* `http://www.mmds.org/`
- *Other Books:* See eKVV. For maximum consistency in online times, other books less relevant.
- *Further Links:* To be provided during course.

Course Curriculum

Part 1: Foundations

- Finding Similar Items I + II
- MapReduce / Workflow Systems I + II
- Mining Data Streams I + II
- Mining Frequent Itemsets
- Clustering

Part 2: Applications

- Link Analysis (PageRank) I + II
- Recommendation Systems
- Web Advertisements
- Social Networks

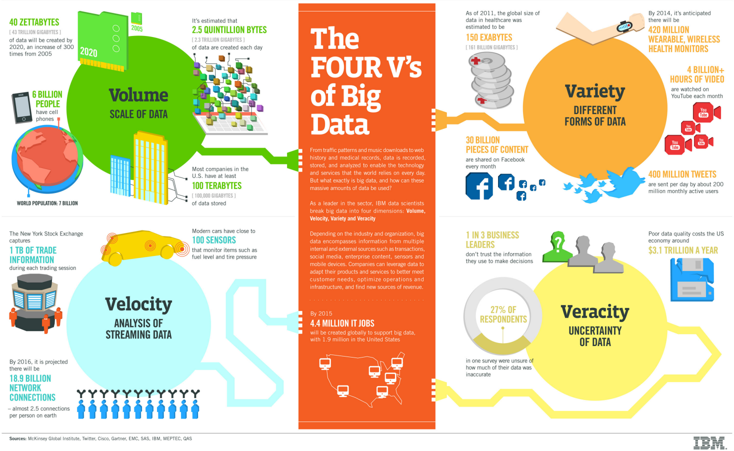
Organizational matters

What is Data Mining?

Statistical Limits

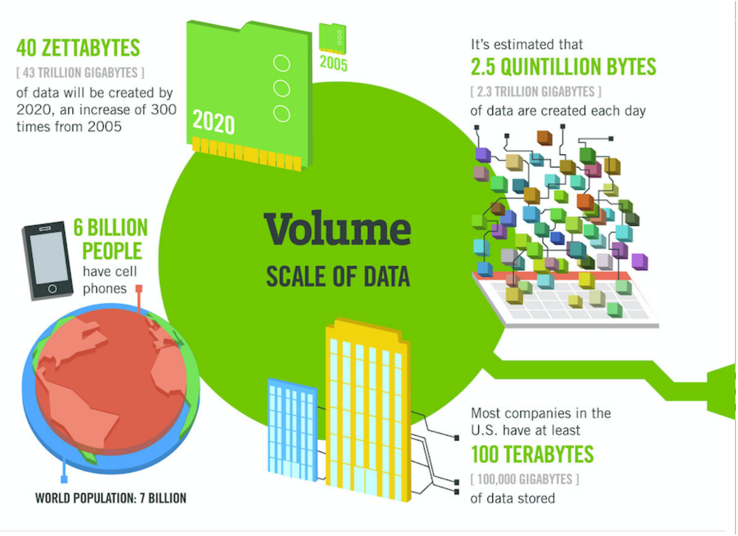
Useful Things

The 4 V's of Big Data



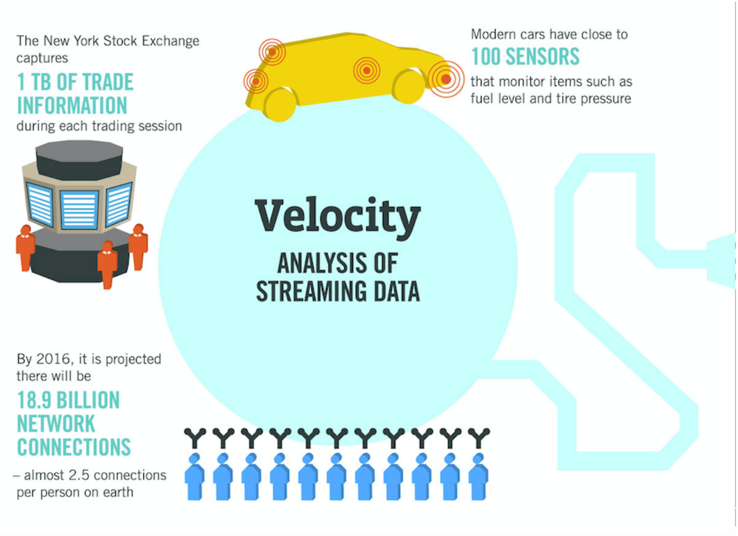
Provided by IBM Big Data & Analytics Hub

THE 4 V'S OF BIG DATA: VOLUME



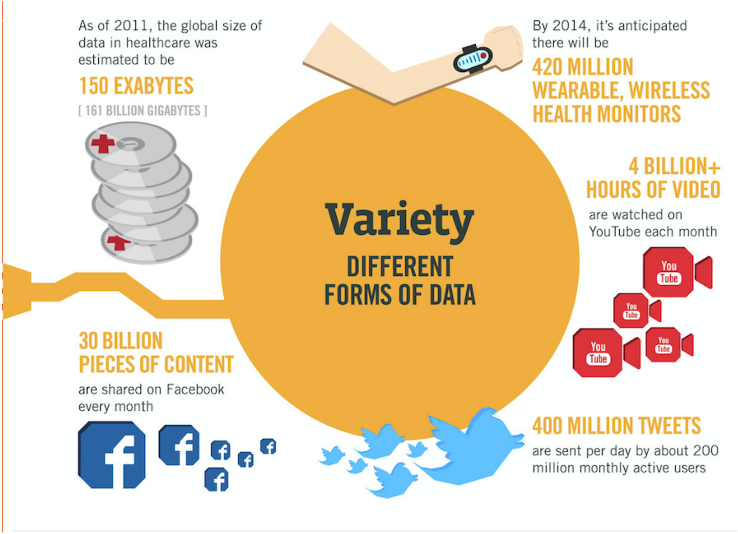
Provided by IBM Big Data & Analytics Hub

The 4 V's of Big Data: Velocity



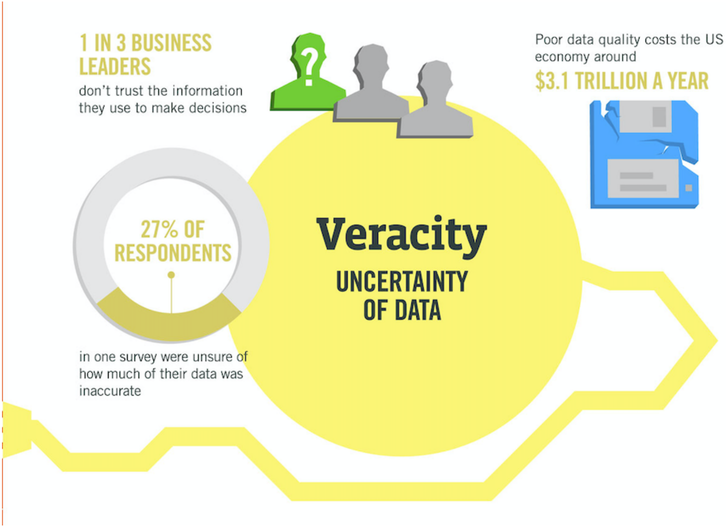
Provided by IBM Big Data & Analytics Hub

THE 4 V'S OF BIG DATA: VARIETY



Provided by IBM Big Data & Analytics Hub

The 4 V's of Big Data: Veracity



Provided by IBM Big Data & Analytics Hub

Data Mining – Meaning

- Data Mining (from 1990) is used interchangeably with
  - Big Data (from 2010)
  - Data Science (today)
- Data mining / Data Science / Big Data is about how to
  - store big data
  - manage big data
  - *analyze* big data ☞ THIS COURSE!

Data Mining – Meaning

- Data Mining (from 1990) is used interchangeably with
  - Big Data (from 2010)
  - Data Science (today)
- Data mining / Data Science / Big Data is about how to
  - store big data
  - manage big data
  - *analyze* big data ☞ THIS COURSE!

Data Mining – Meaning

- Data Mining (from 1990) is used interchangeably with
  - Big Data (from 2010)
  - Data Science (today)
- Data mining / Data Science / Big Data is about how to
  - store big data
  - manage big data
  - *analyze* big data ☞ THIS COURSE!

Data Mining – Modeling

- Often, data mining means to construct a map

$$f : \text{Data} \to \mathcal{S}$$

  where $\mathcal{S}$ is a set of useful labels, values, or similar, and analyze this map.
- Such a map is a *model*.
- *Example:* Detection of phishing emails

Data Mining – Modeling

- Often, data mining means to construct a map

$$f : \text{Data} \to \mathcal{S}$$

  where $\mathcal{S}$ is a set of useful labels, values, or similar, and analyze this map.
- Such a map is a *model*.
- *Example:* Detection of phishing emails

Modeling: Example

- Consider a weighting scheme that assigns a real number $w(x)$ to words or phrases x
- The larger $w(x)$ the more x is indicative of phishing emails
- For example, $w(x)$ is large for x equal to "verify account"
- Consider the map f that maps emails E to real numbers where

$$f(E) = \sum_{x \in E} w(x)$$

  that is, f sums up weights of all words/phrases in the email E

UNIVERSITÄT BIELEFELD

Modeling: Example

- Consider a weighting scheme that assigns a real number $w(x)$ to words or phrases x
- The larger $w(x)$ the more x is indicative of phishing emails
- For example, $w(x)$ is large for x equal to "verify account"
- Consider the map f that maps emails E to real numbers where

$$f(E) = \sum_{x \in E} w(x)$$

that is, f sums up weights of all words/phrases in the email E

MODELING: EXAMPLE

- Consider a weighting scheme that assigns a real number $w(x)$ to words or phrases x
- The larger $w(x)$ the more x is indicative of phishing emails
- For example, $w(x)$ is large for x equal to "verify account"
- Consider the map f that maps emails E to real numbers where

$$f(E) = \sum_{x \in E} w(x)$$

that is, f sums up weights of all words/phrases in the email E

Data Mining – Statistical Modeling

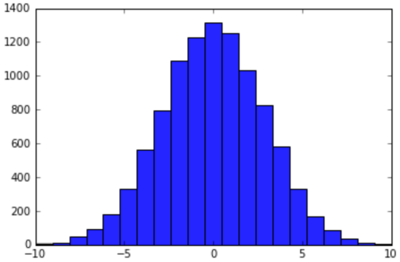
- A *statistical model* of the data is a *probability distribution* that describes the data.
- A *generative model* describes how the data is generated.
- *Example:*
  - Data is a set of integers
  - A statistical model may be a Gaussian distribution that fits the empirical distribution

Data Mining – Statistical Modeling

- A *statistical model* of the data is a *probability distribution* that describes the data.
- A *generative model* describes how the data is generated.
- *Example:*
  - Data is a set of integers
  - A statistical model may be a Gaussian distribution that fits the empirical distribution

Statistical Modeling – Basic Example

Set of Numbers



From stackoverflow.com:

- First fit a Gaussian to the empirical distribution of integers
- Mean and standard deviation sufficient for generating more numbers
  ☞ generative model

Machine Learning

- *Supervised Learning:* Computationally infer model f from data points x for which $f(x)$ is known
- *Unsupervised Learning:* Computationally infer generative statistical model $P(x)$
- Or: computationally infer combinations of the two
- *Possible advantage*: model highly accurate
- *Possible disadvantage*: model too complex to be explainable ☞ *deep learning*

Machine Learning

- *Supervised Learning:* Computationally infer model f from data points x for which $f(x)$ is known
- *Unsupervised Learning:* Computationally infer generative statistical model $P(x)$
- Or: computationally infer combinations of the two
- *Possible advantage*: model highly accurate
- *Possible disadvantage*: model too complex to be explainable
  ☞ *deep learning*

MODELING: COMPUTATIONAL APPROACHES

- Provide probability distribution that reflects to have generated the data (see above)
- *Summarize* all data succinctly and approximately
  - *Example:* Compute the mean and standard deviation of numerical data
- *Extract* only the most *prominent features* of the data, and ignore the rest
  - Consider patient data: keep only height, age, gender, and blood pressure, and discard the rest

Modeling: Computational Approaches

- Provide probability distribution that reflects to have generated the data (see above)
- *Summarize* all data succinctly and approximately
  - *Example:* Compute the mean and standard deviation of numerical data
- *Extract* only the most *prominent features* of the data, and ignore the rest
  - Consider patient data: keep only height, age, gender, and blood pressure, and discard the rest

Modeling: Computational Approaches

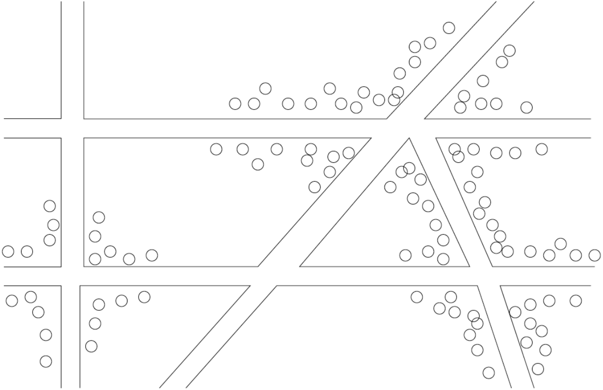
- Provide probability distribution that reflects to have generated the data (see above)
- *Summarize* all data succinctly and approximately
  - *Example:* Compute the mean and standard deviation of numerical data
- *Extract* only the most *prominent features* of the data, and ignore the rest
  - Consider patient data: keep only height, age, gender, and blood pressure, and discard the rest

SUMMARIZATION

Interesting Examples

- *PageRank*: Summarize each web page into one number
  - PageRank computes the number of times a random "web walker" hits a page; the more often, the more "important"
  - PageRank indicates relevance of web page (relative to a search)
- *Clustering*:
  - Group data points, and choose a summarizing representative for each group

Clustering – Example



From http://www.mmds.org.
Cholera cases on a map of London:

Clusters forming around contaminated wells

UNIVERSITÄT BIELEFELD

Feature Extraction: Frequent Itemsets

- Model: “baskets” containing (relatively small) sets of items
- Example: super market. Baskets = shoppers, items = items chosen for purchase.
- *Frequent itemsets*: Small groups of items re-appearing in many baskets.
- Example: burgers and ketchup form a frequent itemset consisting of two items.
- The set of frequent itemsets describes the “behaviour” (characterizes) the data.

Feature Extraction: Frequent Itemsets

- Model: “baskets” containing (relatively small) sets of items
- Example: super market. Baskets = shoppers, items = items chosen for purchase.
- *Frequent itemsets*: Small groups of items re-appearing in many baskets.
- Example: burgers and ketchup form a frequent itemset consisting of two items.
- The set of frequent itemsets describes the “behaviour” (characterizes) the data.

Feature Extraction: Similar Items

- Model: Data = collection of sets
- *Similar items:* Pairs of sets that are sufficiently similar.
- Example: Amazon buyers, mining similar items refers to identifying shoppers that have purchased similar goods
- Used for recommending items to buyers; process is called *collaborative filtering*

Organizational matters

What is Data Mining?

Statistical Limits

Useful Things

Discovering Unusual Events in Big Data

- The more one searches, the more likely "unusual" events are discovered
- Are they still unusual?
- *Issue:* When looking at too many things at a time, one discovers things that are interesting, just because they are statistical artifacts
- *Example*: Total Awareness Information
  - American response to 9-11.
  - Attempt to spot "unusual" (terrorist like) behaviour in credit-card receipts, flight schedule records, hotel information, and so on.
  - Vast majority of "terrorist like" behaviour spotted harmless
- *Bonferroni's principle* deals with the corresponding limits

Discovering Unusual Events in Big Data

- The more one searches, the more likely "unusual" events are discovered
- Are they still unusual?
- *Issue:* When looking at too many things at a time, one discovers things that are interesting, just because they are statistical artifacts
- *Example*: Total Awareness Information
  - American response to 9-11.
  - Attempt to spot "unusual" (terrorist like) behaviour in credit-card receipts, flight schedule records, hotel information, and so on.
  - Vast majority of "terrorist like" behaviour spotted harmless
- *Bonferroni's principle* deals with the corresponding limits

Discovering Unusual Events in Big Data

- The more one searches, the more likely "unusual" events are discovered
- Are they still unusual?
- *Issue:* When looking at too many things at a time, one discovers things that are interesting, just because they are statistical artifacts
- *Example*: Total Awareness Information
  - American response to 9-11.
  - Attempt to spot "unusual" (terrorist like) behaviour in credit-card receipts, flight schedule records, hotel information, and so on.
  - Vast majority of "terrorist like" behaviour spotted harmless
- *Bonferroni's principle* deals with the corresponding limits

Discovering Unusual Events in Big Data

- The more one searches, the more likely "unusual" events are discovered
- Are they still unusual?
- *Issue:* When looking at too many things at a time, one discovers things that are interesting, just because they are statistical artifacts
- *Example*: Total Awareness Information
  - American response to 9-11.
  - Attempt to spot "unusual" (terrorist like) behaviour in credit-card receipts, flight schedule records, hotel information, and so on.
  - Vast majority of "terrorist like" behaviour spotted harmless
- *Bonferroni's principle* deals with the corresponding limits

Discovering Unusual Events in Big Data

- The more one searches, the more likely "unusual" events are discovered
- Are they still unusual?
- *Issue:* When looking at too many things at a time, one discovers things that are interesting, just because they are statistical artifacts
- *Example*: Total Awareness Information
  - American response to 9-11.
  - Attempt to spot "unusual" (terrorist like) behaviour in credit-card receipts, flight schedule records, hotel information, and so on.
  - Vast majority of "terrorist like" behaviour spotted harmless
- *Bonferroni's principle* deals with the corresponding limits

Bonferroni's Principle

- The number of unlikely events to occur randomly will grow when data grows.
- So, when data is big, many "interesting" things may be bogus, because they are statistical artifacts.
- *Bonferroni's principle* computes the probability of unlikely events to occur by chance.

Bonferroni's Principle – Example

Spot group of "evil-doers" who regularly meet in a hotel.

- There are one billion (10^9) people to be watched
- On average: random people stay in a hotel 1 out of 100 days
- On average: a hotel holds 100 people
- So we can deal with 100 000 hotels, because

$$100\,000 \times 100 = \frac{10^9}{100}$$

- Data: hotel records for 1000 days.

BONFERRONI'S PRINCIPLE – EXAMPLE

- *Definition of evil-doers:*
  Pairs meet in two different hotels on two different days
- *Let us assume that* there aren't any evil-doers
- *Question:* What is the probability to spot a pair of "evil-doers" although there aren't any, just by random effects?

RANDOM EVIL-DOERS: CALCULATION

- Probability that two randomly picked people visit a hotel on one particular day:

$$0.01 \times 0.01 = 10^{-4}$$

- Probability that they choose the same hotel:

$$1 \times 10^{-5} = 10^{-5}$$

- Probability that two random people meet in the same hotel on one day is:

- Probability that two random people meet in the same hotel on two particular, different days is:

Random Evil-Doers: Calculation

- Probability that two randomly picked people visit a hotel on one particular day:

$$0.01 \times 0.01 = 10^{-4}$$

- Probability that they choose the same hotel:

$$1 \times 10^{-5} = 10^{-5}$$

- Probability that two random people meet in the same hotel on one day is:

- Probability that two random people meet in the same hotel on two particular, different days is:

Random Evil-Doers: Calculation

- Probability that two randomly picked people visit a hotel on one particular day:

$$0.01 \times 0.01 = 10^{-4}$$

- Probability that they choose the same hotel:

$$1 \times 10^{-5} = 10^{-5}$$

- Probability that two random people meet in the same hotel on one day is:

- Probability that two random people meet in the same hotel on two particular, different days is:

Random Evil-Doers: Calculation

- Probability that two randomly picked people visit a hotel on one particular day:
$$0.01 \times 0.01 = 10^{-4}$$
- Probability that they choose the same hotel:
$$1 \times 10^{-5} = 10^{-5}$$
- Probability that two random people meet in the same hotel on one day is:
$$10^{-4} \times 10^{-5} = 10^{-9}$$
- Probability that two random people meet in the same hotel on two particular, different days is:

Random Evil-Doers: Calculation

- Probability that two randomly picked people visit a hotel on one particular day:
$$0.01 \times 0.01 = 10^{-4}$$
- Probability that they choose the same hotel:
$$1 \times 10^{-5} = 10^{-5}$$
- Probability that two random people meet in the same hotel on one day is:
$$10^{-4} \times 10^{-5} = 10^{-9}$$
- Probability that two random people meet in the same hotel on two particular, different days is:

Random Evil-Doers: Calculation

- Probability that two randomly picked people visit a hotel on one particular day:
$$0.01 \times 0.01 = 10^{-4}$$
- Probability that they choose the same hotel:
$$1 \times 10^{-5} = 10^{-5}$$
- Probability that two random people meet in the same hotel on one day is:
$$10^{-4} \times 10^{-5} = 10^{-9}$$
- Probability that two random people meet in the same hotel on two particular, different days is:
$$10^{-9} \times 10^{-9} = 10^{-18}$$

Bonferroni's Principle – Example

- Probability that two random people meet in the same hotel on two different days is
$$10^{-9} \times 10^{-9} = 10^{-18}$$
- Clearly the more people and the more days, the greater the chance that two random people meet in the same hotel on the same day.
- Number of pairs of people and pairs of days is:
$$\binom{10^9}{2} = 5 \times 10^{17} \quad \text{and} \quad \binom{1000}{2} = 5 \times 10^5$$
- So, number of random(!) events that meet the definition of "evil-doing" is
$$10^{-18} \times (5 \times 10^{17}) \times (5 \times 10^5) = 250\,000$$
- **Summary:** A quarter million pairs of people look like "doing evil" just by chance

Bonferroni's Principle – Example

- Probability that two random people meet in the same hotel on two different days is
$$10^{-9} \times 10^{-9} = 10^{-18}$$
- Clearly the more people and the more days, the greater the chance that two random people meet in the same hotel on the same day.
- Number of pairs of people and pairs of days is:
$$\binom{10^9}{2} = 5 \times 10^{17} \quad \text{and} \quad \binom{1000}{2} = 5 \times 10^5$$
- So, number of random(!) events that meet the definition of "evil-doing" is
$$10^{-18} \times (5 \times 10^{17}) \times (5 \times 10^5) = 250\,000$$
- **Summary:** A quarter million pairs of people look like "doing evil" just by chance

Bonferroni's Principle – Example

- Probability that two random people meet in the same hotel on two different days is

$$10^{-9} \times 10^{-9} = 10^{-18}$$

- Clearly the more people and the more days, the greater the chance that two random people meet in the same hotel on the same day.
- Number of pairs of people and pairs of days is:

$$\binom{10^9}{2} = 5 \times 10^{17} \quad \text{and} \quad \binom{1000}{2} = 5 \times 10^5$$

- So, number of random(!) events that meet the definition of "evil-doing" is

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

- **Summary:** A quarter million pairs of people look like "doing evil" just by chance

Bonferroni's Principle – Example

- Probability that two random people meet in the same hotel on two different days is
$$10^{-9} \times 10^{-9} = 10^{-18}$$
- Clearly the more people and the more days, the greater the chance that two random people meet in the same hotel on the same day.
- Number of pairs of people and pairs of days is:
$$\binom{10^9}{2} = 5 \times 10^{17} \quad \text{and} \quad \binom{1000}{2} = 5 \times 10^5$$
- So, number of random(!) events that meet the definition of "evil-doing" is
$$10^{-18} \times (5 \times 10^{17}) \times (5 \times 10^5) = 250\,000$$
- **Summary:** A quarter million pairs of people look like "doing evil" just by chance

BONFERRONI'S PRINCIPLE – EXAMPLE

- Probability that two random people meet in the same hotel on two different days is
$$10^{-9} \times 10^{-9} = 10^{-18}$$
- Clearly the more people and the more days, the greater the chance that two random people meet in the same hotel on the same day.
- Number of pairs of people and pairs of days is:
$$\binom{10^9}{2} = 5 \times 10^{17} \quad \text{and} \quad \binom{1000}{2} = 5 \times 10^5$$
- So, number of random(!) events that meet the definition of "evil-doing" is
$$10^{-18} \times (5 \times 10^{17}) \times (5 \times 10^5) = 250\,000$$
- **Summary:** A quarter million pairs of people look like "doing evil" just by chance

Organizational matters

What is Data Mining?

Statistical Limits

Useful Things

Useful Things to Know

- The TD.IDF measure of word importance
- Hash functions
- Secondary storage (disk) and running time of algorithms
- The natural logarithm
- Power laws

TD.IDF: INTRODUCTION

- *Goal:* Find words in documents (such as emails, news articles) that are characteristic of the contents
- Example: in texts on the corona virus, you may see “corona”, “virus”, “infection”, “cough”, “fever” more often than usual
- However: the most frequent words are likely to be “the” and “and” (or the likes)
- So, words indicative of topics are rather rare.

TD.IDF: Introduction

- *Goal:* Find words in documents (such as emails, news articles) that are characteristic of the contents
- Example: in texts on the corona virus, you may see “corona”, “virus”, “infection”, “cough”, “fever” more often than usual
- However: the most frequent words are likely to be “the” and “and” (or the likes)
- So, words indicative of topics are rather rare.

TD.IDF: INTRODUCTION

- However: the most frequent words are likely to be "the" and "and" (or the likes)
- So, words indicative of topics are rather rare.
- While, of course, there are also many rare words (such as "albeit", "notwithstanding" or similar) that are not indicative of the topic, because rather generic.
- How to find words indicative of topics of interest?
- Compute the TF.IDF = Term Frequency times Inverse Document Frequency!

TD.IDF: Introduction

- However: the most frequent words are likely to be "the" and "and" (or the likes)
- So, words indicative of topics are rather rare.
- While, of course, there are also many rare words (such as "albeit", "notwithstanding" or similar) that are not indicative of the topic, because rather generic.
- How to find words indicative of topics of interest?
- Compute the TF.IDF = Term Frequency times Inverse Document Frequency!

TD.IDF: Introduction

- However: the most frequent words are likely to be “the” and “and” (or the likes)
- So, words indicative of topics are rather rare.
- While, of course, there are also many rare words (such as “albeit”, “notwithstanding” or similar) that are not indicative of the topic, because rather generic.
- How to find words indicative of topics of interest?
- Compute the TF.IDF = Term Frequency times Inverse Document Frequency!

Computing the TF.IDF

- Compute the *Term Frequency* TF_{ij}

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \quad (1)$$

where f_{ij} is the number of occurrences of word i in document j.

- Note: the most frequent term in document j gets a TF of 1.
- Compute the *Inverse Document Frequency* IDF_i of i as

$$IDF_i = \log_2(\frac{N}{n_i}) \quad (2)$$

where N is the number of documents overall, and n_i is the number of documents in which word i appears.

- So, $n_i \leq N$ and $IDF_i \geq 0$
- TF.IDF for term i in document j is defined to be

$$TF_{ij} \times IDF_i \quad (3)$$

Computing the TF.IDF

- Compute the *Term Frequency* TF_{ij}

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \tag{1}$$

  where f_{ij} is the number of occurrences of word i in document j.
- Note: the most frequent term in document j gets a TF of 1.
- Compute the *Inverse Document Frequency* IDF_i of i as

$$IDF_i = \log_2(\frac{N}{n_i}) \tag{2}$$

  where N is the number of documents overall, and n_i is the number of documents in which word i appears.
- So, $n_i \leq N$ and $IDF_i \geq 0$
- TF.IDF for term i in document j is defined to be

$$TF_{ij} \times IDF_i \tag{3}$$

Computing the TF.IDF

- Compute the *Term Frequency* TF_{ij}

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \quad (1)$$

where f_{ij} is the number of occurrences of word i in document j.

- Note: the most frequent term in document j gets a TF of 1.
- Compute the *Inverse Document Frequency* IDF_i of i as

$$IDF_i = \log_2(\frac{N}{n_i}) \quad (2)$$

where N is the number of documents overall, and n_i is the number of documents in which word i appears.

- So, $n_i \leq N$ and $IDF_i \geq 0$
- TF.IDF for term i in document j is defined to be

$$TF_{ij} \times IDF_i \quad (3)$$

TF.IDF: Explanations

- Terms with highest TF.IDF are often the terms that explain the document best. Why?
- If a word i appears in all documents:

$$IDF_i = \log_2(\frac{N}{n_i}) \overset{n_i=N}{=} \log_2(1) = 0$$

so that word cannot be characteristic of any document

TF.IDF: Explanations

- Terms with highest TF.IDF are often the terms that explain the document best. Why?
- If a word i appears in all documents:

$$IDF_i = \log_2(\frac{N}{n_i}) \overset{n_i=N}{=} \log_2(1) = 0$$

so that word cannot be characteristic of any document

TF.IDF: Explanations

- Terms with highest TF.IDF are often the terms that explain the document best. Why?
- Suppose we have 2^{20} documents
  - Suppose word w appears in 2^{10} documents:

$$IDF_w = \log_2(2^{20}/2^{10}) = \log_2(2^{10}) = 10$$

  - Consider document j in which w appears 20 times, which is the maximum of appearances in one document:

$$TF_{wj} = \frac{20}{20} = 1, \text{ so } TF.IDF_{wj} = 10$$

  - Consider document k, in which w appears once:

$$TF.IDF_{wk} = \frac{1}{2}$$

TF.IDF: Explanations

- Terms with highest TF.IDF are often the terms that explain the document best. Why?
- Suppose we have 2^{20} documents
  - Suppose word w appears in 2^{10} documents:

$$IDF_w = \log_2(2^{20}/2^{10}) = \log_2(2^{10}) = 10$$

  - Consider document j in which w appears 20 times, which is the maximum of appearances in one document:

$$TF_{wj} = \frac{20}{20} = 1, \text{ so } TF.IDF_{wj} = 10$$

  - Consider document k, in which w appears once:

$$TF.IDF_{wk} = \frac{1}{2}$$

TF.IDF: Explanations

- Terms with highest TF.IDF are often the terms that explain the document best. Why?
- Suppose we have 2^{20} documents
  - Suppose word w appears in 2^{10} documents:

$$IDF_w = \log_2(2^{20}/2^{10}) = \log_2(2^{10}) = 10$$

  - Consider document j in which w appears 20 times, which is the maximum of appearances in one document:

$$TF_{wj} = \frac{20}{20} = 1, \text{ so } TF.IDF_{wj} = 10$$

  - Consider document k, in which w appears once:

$$TF.IDF_{wk} = \frac{1}{2}$$

TF.IDF: EXPLANATIONS

- Terms with highest TF.IDF are often the terms that explain the document best. Why?
- Suppose we have 2^{20} documents
  - Suppose word w appears in 2^{10} documents:

$$IDF_w = \log_2(2^{20}/2^{10}) = \log_2(2^{10}) = 10$$

  - Consider document j in which w appears 20 times, which is the maximum of appearances in one document:

$$TF_{wj} = \frac{20}{20} = 1, \text{ so } TF.IDF_{wj} = 10$$

  - Consider document k, in which w appears once:

$$TF.IDF_{wk} = \frac{1}{2}$$

Hash Functions

- A hash function takes a *hash-key* x as input and maps it to a bucket number.
- The bucket number is a an integer in the range from 0 to B-1, where B is the number of buckets.
- *Example*: Hash-keys are positive integers.

$$h(x) = x \mod B$$

which is the remainder of x when dividing it by B. Often, B is a prime.

HASH FUNCTIONS

- A hash function takes a *hash-key* x as input and maps it to a bucket number.
- The bucket number is a an integer in the range from 0 to B-1, where B is the number of buckets.
- *Example*: Hash-keys are positive integers.

$$h(x) = x \mod B$$

which is the remainder of x when dividing it by B. Often, B is a prime.

HASH FUNCTIONS

- If hash-keys are not integers, they are often converted to integers.
- Example: if hash-keys are strings, one can map each character to its ASCII code, and sum them up, before dividing them by B.
- If hash-keys have several components (such as arrays), convert each component to integer, and sum them up.
- Let $h(x) := x \mod 5$. *Example:*

$$h("AB") = h(ord('A') + ord('B')) = h(65 + 66) = h(131) = 1$$

Hash Functions

- If hash-keys are not integers, they are often converted to integers.
- Example: if hash-keys are strings, one can map each character to its ASCII code, and sum them up, before dividing them by B.
- If hash-keys have several components (such as arrays), convert each component to integer, and sum them up.
- Let $h(x) := x \mod 5$. *Example:*

$$h(\text{"}AB\text{"}) = h(ord('A') + ord('B')) = h(65 + 66) = h(131) = 1$$

Hash Functions

- If hash-keys are not integers, they are often converted to integers.
- Example: if hash-keys are strings, one can map each character to its ASCII code, and sum them up, before dividing them by B.
- If hash-keys have several components (such as arrays), convert each component to integer, and sum them up.
- Let $h(x) := x \mod 5$. *Example:*

$$h(\text{"}AB\text{"}) = h(ord('A') + ord('B')) = h(65 + 66) = h(131) = 1$$

Hash Functions

- If hash-keys are not integers, they are often converted to integers.
- Example: if hash-keys are strings, one can map each character to its ASCII code, and sum them up, before dividing them by B.
- If hash-keys have several components (such as arrays), convert each component to integer, and sum them up.
- Let $h(x) := x \mod 5$. *Example:*

$$h(\text{"}AB\text{"}) = h(ord('A') + ord('B')) = h(65 + 66) = h(131) = 1$$

Number of Keys vs Number of Buckets

- Usually, there are more than B hash-keys conceivable; but usually not all of them are in use.
- If only less than B hash-keys are in use, with only little probability, hash collisions

$$x_1 \neq x_2 \quad \text{but} \quad h(x_1) = h(x_2)$$

  happen to occur.
- If number of hash-keys is much larger than B, then hash functions "randomize" keys, by distributing them (optimally) uniformly across the whole range [0,B-1]
- That is more likely to happen when B is a prime

Number of Keys vs Number of Buckets

- Usually, there are more than B hash-keys conceivable; but usually not all of them are in use.
- If only less than B hash-keys are in use, with only little probability, hash collisions

$$x_1 \neq x_2 \quad \text{but} \quad h(x_1) = h(x_2)$$

happen to occur.
- If number of hash-keys is much larger than B, then hash functions "randomize" keys, by distributing them (optimally) uniformly across the whole range [0,B-1]
- That is more likely to happen when B is a prime

Number of Keys vs Number of Buckets

- Usually, there are more than B hash-keys conceivable; but usually not all of them are in use.
- If only less than B hash-keys are in use, with only little probability, hash collisions
$$x_1 \neq x_2 \quad \text{but} \quad h(x_1) = h(x_2)$$
happen to occur.
- If number of hash-keys is much larger than B, then hash functions "randomize" keys, by distributing them (optimally) uniformly across the whole range [0,B-1]
- That is more likely to happen when B is a prime

Indexes

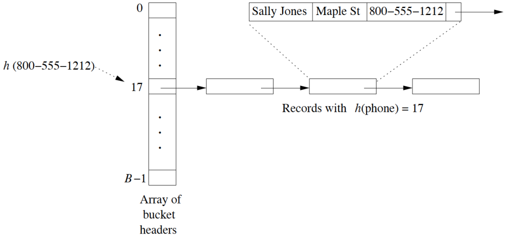
- Data structure that enables to retrieve all records specified by a particular feature.
- *Example*: Consider an address book with entries (name, address, phone number). We would like to retrieve all entries with a particular phone number.
- One solution is to use a hash table:

Indexes

- Data structure that enables to retrieve all records specified by a particular feature.
- *Example*: Consider an address book with entries (name, address, phone number). We would like to retrieve all entries with a particular phone number.
- One solution is to use a hash table:

INDEXES

- Data structure that enables to retrieve all records specified by a particular feature.
- *Example*: Consider an address book with entries (name, address, phone number). We would like to retrieve all entries with a particular phone number.
- One solution is to use a hash table:



Hash table used as index for retrieving address records based by their phone number

Secondary Storage

- Important to keep in mind when dealing with big data: accessing data from disks (hard drives) costs time (and energy).
- Disks are organized into blocks; e.g. blocks of 64K bytes.
- Takes approx. 10 milliseconds to *access* and read a disk block.
- About 10^5 times slower than accessing data in main memory.
- And taking a block to main memory costs more time than executing the computations on the data when being in main memory.

SECONDARY STORAGE

- Important to keep in mind when dealing with big data: accessing data from disks (hard drives) costs time (and energy).
- Disks are organized into blocks; e.g. blocks of 64K bytes.
- Takes approx. 10 milliseconds to *access* and read a disk block.
- About 10^5 times slower than accessing data in main memory.
- And taking a block to main memory costs more time than executing the computations on the data when being in main memory.

Secondary Storage

- Important to keep in mind when dealing with big data: accessing data from disks (hard drives) costs time (and energy).
- Disks are organized into blocks; e.g. blocks of 64K bytes.
- Takes approx. 10 milliseconds to *access* and read a disk block.
- About 10^5 times slower than accessing data in main memory.
- And taking a block to main memory costs more time than executing the computations on the data when being in main memory.

SECONDARY STORAGE

- One can alleviate problem by putting related data on a single *cylinder*, where accessing all blocks on a cylinder costs considerably less time per block.
- This establishes a limit of 100MB per second to transfer blocks to main memory.
- If data is in the hundreds of gigabytes, let alone terabytes, this is an issue.
- *Integrate this knowledge into runtime considerations when dealing with big data!*

SECONDARY STORAGE

- One can alleviate problem by putting related data on a single *cylinder*, where accessing all blocks on a cylinder costs considerably less time per block.
- This establishes a limit of 100MB per second to transfer blocks to main memory.
- If data is in the hundreds of gigabytes, let alone terabytes, this is an issue.
- *Integrate this knowledge into runtime considerations when dealing with big data!*

Secondary Storage

- One can alleviate problem by putting related data on a single *cylinder*, where accessing all blocks on a cylinder costs considerably less time per block.
- This establishes a limit of 100MB per second to transfer blocks to main memory.
- If data is in the hundreds of gigabytes, let alone terabytes, this is an issue.
- *Integrate this knowledge into runtime considerations when dealing with big data!*

The Natural Logarithm I

- Euler constant:

$$e = \lim_{x\to\infty}(1+\frac{1}{x})^x \approx 2.71828 \quad (4)$$

- Consider computing $(1+a)^b$ where a is small:

$$(1+a)^b = (1+a)^{(1/a)(ab)} \overset{a=1/x}{=} (1+\frac{1}{x})^{x(ab)} = ((1+\frac{1}{x})^x)^{ab} \overset{x \text{ large}}{\approx} e^{ab}$$

- Consider computing $(1-a)^b$ where a is small:

$$(1-a)^b = ((1-\frac{1}{x})^x)^{ab} \overset{x \text{ large}}{\approx} e^{-ab}$$

The Natural Logarithm I

- Euler constant:

$$e = \lim_{x\to\infty}(1+\frac{1}{x})^x \approx 2.71828 \tag{4}$$

- Consider computing $(1+a)^b$ where a is small:

$$(1+a)^b = (1+a)^{(1/a)(ab)} \overset{a=1/x}{=} (1+\frac{1}{x})^{x(ab)} = ((1+\frac{1}{x})^x)^{ab} \overset{x \text{ large}}{\approx} e^{ab}$$

- Consider computing $(1-a)^b$ where a is small:

$$(1-a)^b = ((1-\frac{1}{x})^x)^{ab} \overset{x \text{ large}}{\approx} e^{-ab}$$

The Natural Logarithm I

- Euler constant:

$$e = \lim_{x \to \infty} (1 + \frac{1}{x})^x \approx 2.71828 \tag{4}$$

- Consider computing $(1 + a)^b$ where a is small:

$$(1 + a)^b = (1 + a)^{(1/a)(ab)} \stackrel{a=1/x}{=} (1 + \frac{1}{x})^{x(ab)} = ((1 + \frac{1}{x})^x)^{ab} \stackrel{x \text{ large}}{\approx} e^{ab}$$

- Consider computing $(1 - a)^b$ where a is small:

$$(1 - a)^b = ((1 - \frac{1}{x})^x)^{ab} \stackrel{x \text{ large}}{\approx} e^{-ab}$$

Euler Constant: Taylor Expansion of e^x

- The Taylor expansion of e^x is

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \ldots \quad (5)$$

- Convergence slow on large x, so not helpful.
- Convergence fast on small (positive and negative) x.
- *Example:* $x = 1/2$

$$e^{1/2} = 1 + \frac{1}{2} + \frac{1}{8} + \frac{1}{48} + \frac{1}{384} + \ldots \approx 1.64844$$

- *Example:* $x = -1$

$$e^{-1} = 1 - 1 + \frac{1}{2} - \frac{1}{6} + \frac{1}{24} - \frac{1}{120} + \frac{1}{720} - \frac{1}{5040} \ldots \approx 0.36786$$

Euler Constant: Taylor Expansion of e^x

- The Taylor expansion of e^x is

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \ldots \quad (5)$$

- Convergence slow on large x, so not helpful.
- Convergence fast on small (positive and negative) x.
- *Example:* $x = 1/2$

$$e^{1/2} = 1 + \frac{1}{2} + \frac{1}{8} + \frac{1}{48} + \frac{1}{384} + \ldots \approx 1.64844$$

- *Example:* $x = -1$

$$e^{-1} = 1 - 1 + \frac{1}{2} - \frac{1}{6} + \frac{1}{24} - \frac{1}{120} + \frac{1}{720} - \frac{1}{5040} \ldots \approx 0.36786$$

Euler Constant: Taylor Expansion of e^x

- The Taylor expansion of e^x is

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \dots \tag{5}$$

- Convergence slow on large x, so not helpful.
- Convergence fast on small (positive and negative) x.
- *Example:* $x = 1/2$

$$e^{1/2} = 1 + \frac{1}{2} + \frac{1}{8} + \frac{1}{48} + \frac{1}{384} + \dots \approx 1.64844$$

- *Example:* $x = -1$

$$e^{-1} = 1 - 1 + \frac{1}{2} - \frac{1}{6} + \frac{1}{24} - \frac{1}{120} + \frac{1}{720} - \frac{1}{5040} \dots \approx 0.36786$$

Euler Constant: Taylor Expansion of e^x

- The Taylor expansion of e^x is

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \ldots \qquad (5)$$

- Convergence slow on large x, so not helpful.
- Convergence fast on small (positive and negative) x.
- *Example:* $x = 1/2$

$$e^{1/2} = 1 + \frac{1}{2} + \frac{1}{8} + \frac{1}{48} + \frac{1}{384} + \ldots \approx 1.64844$$

- *Example:* $x = -1$

$$e^{-1} = 1 - 1 + \frac{1}{2} - \frac{1}{6} + \frac{1}{24} - \frac{1}{120} + \frac{1}{720} - \frac{1}{5040} \ldots \approx 0.36786$$

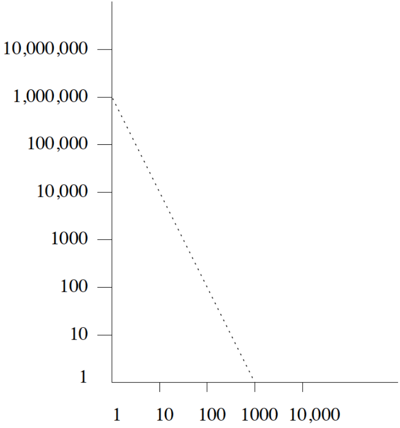
Power Laws

- Consider two variables y and x and their functional relationship.
- General form of a power law is

$$\log y = b + a \log x \tag{6}$$

so a linear relationship between the logarithms of x and y.

Power Law: Example



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

Power Laws

- Power law:

$$\log y = b + a \log x \tag{7}$$

- Transforming yields:

$$y = e^b \cdot e^{a \log x} = e^b \cdot e^{\log x^a} = e^b \cdot x^a$$

so power law expresses polynomial relationship $y = cx^a$

Real World Scenarios

- *Node degrees in web graph*
  - Nodes are web pages
  - Nodes are linked when there are links between pages
  - Order pages by numbers of links: number of links as a function of the order number is power law
- *Sales of products: y* is the number of sales of the *x*-th most popular item (books at amazon.com, say)
- *Sizes of web sites: y* is number of pages at the *x*-th largest web site
- *Zipf's Law:* Order words in document by frequency, and let *y* be the number of times the *x*-th word appears in the document.
  - Zipf found the relationship to approximately reflect $y = cx^{-1/2}$.
  - Other relationships follow that law, too. For example, *y* is population of *x*-th most populous (American) state.
- Summary: *The Matthew Effect* = "The rich get ever richer"

Real World Scenarios

- *Node degrees in web graph*
  - Nodes are web pages
  - Nodes are linked when there are links between pages
  - Order pages by numbers of links: number of links as a function of the order number is power law
- *Sales of products: y* is the number of sales of the x-th most popular item (books at amazon.com, say)
- *Sizes of web sites: y* is number of pages at the x-th largest web site
- *Zipf's Law:* Order words in document by frequency, and let y be the number of times the x-th word appears in the document.
  - Zipf found the relationship to approximately reflect $y = cx^{-1/2}$.
  - Other relationships follow that law, too. For example, y is population of x-th most populous (American) state.
- Summary: *The Matthew Effect* = "The rich get ever richer"

Real World Scenarios

- *Node degrees in web graph*
  - Nodes are web pages
  - Nodes are linked when there are links between pages
  - Order pages by numbers of links: number of links as a function of the order number is power law
- *Sales of products: y* is the number of sales of the x-th most popular item (books at amazon.com, say)
- *Sizes of web sites: y* is number of pages at the x-th largest web site
- *Zipf's Law:* Order words in document by frequency, and let y be the number of times the x-th word appears in the document.
  - Zipf found the relationship to approximately reflect $y = cx^{-1/2}$.
  - Other relationships follow that law, too. For example, y is population of x-th most populous (American) state.
- Summary: *The Matthew Effect* = "The rich get ever richer"

Real World Scenarios

- *Node degrees in web graph*
  - Nodes are web pages
  - Nodes are linked when there are links between pages
  - Order pages by numbers of links: number of links as a function of the order number is power law
- *Sales of products: y* is the number of sales of the x-th most popular item (books at amazon.com, say)
- *Sizes of web sites: y* is number of pages at the x-th largest web site
- *Zipf's Law:* Order words in document by frequency, and let y be the number of times the x-th word appears in the document.
  - Zipf found the relationship to approximately reflect $y = cx^{-1/2}$.
  - Other relationships follow that law, too. For example, y is population of x-th most populous (American) state.
- Summary: *The Matthew Effect* = "The rich get ever richer"

Real World Scenarios

- *Node degrees in web graph*
  - Nodes are web pages
  - Nodes are linked when there are links between pages
  - Order pages by numbers of links: number of links as a function of the order number is power law
- *Sales of products: y* is the number of sales of the x-th most popular item (books at amazon.com, say)
- *Sizes of web sites: y* is number of pages at the x-th largest web site
- *Zipf's Law:* Order words in document by frequency, and let y be the number of times the x-th word appears in the document.
  - Zipf found the relationship to approximately reflect $y = cx^{-1/2}$.
  - Other relationships follow that law, too. For example, y is population of x-th most populous (American) state.
- Summary: *The Matthew Effect* = "The rich get ever richer"

Real World Scenarios

- *Node degrees in web graph*
  - Nodes are web pages
  - Nodes are linked when there are links between pages
  - Order pages by numbers of links: number of links as a function of the order number is power law
- *Sales of products: y* is the number of sales of the x-th most popular item (books at amazon.com, say)
- *Sizes of web sites: y* is number of pages at the x-th largest web site
- *Zipf's Law:* Order words in document by frequency, and let y be the number of times the x-th word appears in the document.
  - Zipf found the relationship to approximately reflect $y = cx^{-1/2}$.
  - Other relationships follow that law, too. For example, y is population of x-th most populous (American) state.
- Summary: *The Matthew Effect* = "The rich get ever richer"

Materials / Outlook

- See *Mining of Massive Datasets*, chapter 1
- See further `http://www.mmds.org/` in general for further resources
- Next lecture: "Finding Similar Items"
  - See *Mining of Massive Datasets* 3.1–3.6