MSC BD 5002/IT 5210: Knowledge Discovery and Data Mining

Acknowledgement: Slides modified by Dr. Lei Chen based on the slides provided by Jiawei Han, Micheline Kamber, and Jian Pei

1

Chapter 3: Data Preprocessing

- Data Preprocessing: An Overview 
  - Data Quality
  - Major Tasks in Data Preprocessing
- Data Cleaning
- Data Integration
- Data Reduction
- Data Transformation and Data Discretization
- Summary

Data Quality: Why Preprocess the Data?

- Measures for data quality: A multidimensional view
  - Accuracy: correct or wrong, accurate or not
  - Completeness: not recorded, unavailable, …
  - Consistency: some modified but some not, dangling, …
  - Timeliness: timely update?
  - Believability: how trustable the data are correct?
  - Interpretability: how easily the data can be understood?

Major Tasks in Data Preprocessing

- **Data cleaning**
  - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- **Data integration**
  - Integration of multiple databases, data cubes, or files
- **Data reduction**
  - Dimensionality reduction
  - Numerosity reduction
  - Data compression
- **Data transformation and data discretization**
  - Normalization
  - Concept hierarchy generation

Chapter 3: Data Preprocessing

- Data Preprocessing: An Overview
  - Data Quality
  - Major Tasks in Data Preprocessing
- Data Cleaning 
- Data Integration
- Data Reduction
- Data Transformation and Data Discretization
- Summary

Data Cleaning

- Data in the Real World Is Dirty: Lots of potentially incorrect data, e.g., instrument faulty, human or computer error, transmission error
  - incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
    - e.g., *Occupation*=" " (missing data)
  - noisy: containing noise, errors, or outliers
    - e.g., *Salary*="−10" (an error)
  - inconsistent: containing discrepancies in codes or names, e.g.,
    - *Age*="42", *Birthday*="03/07/2010"
    - Was rating "1, 2, 3", now rating "A, B, C"
    - discrepancy between duplicate records
  - Intentional (e.g., *disguised missing* data)
    - Jan. 1 as everyone's birthday?

Incomplete (Missing) Data

- Data is not always available
  - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be due to
  - equipment malfunction
  - inconsistent with other recorded data and thus deleted
  - data not entered due to misunderstanding
  - certain data may not be considered important at the time of entry
  - not register history or changes of the data
- Missing data may need to be inferred

How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing (when doing classification)—not effective when the % of missing values per attribute varies considerably
- Fill in the missing value manually: tedious + infeasible?
- Fill in it automatically with
  - a global constant : e.g., "unknown", a new class?!
  - the attribute mean
  - the attribute mean for all samples belonging to the same class: smarter
  - the most probable value: inference-based such as Bayesian formula or decision tree

Noisy Data

- Noise: random error or variance in a measured variable
- Incorrect attribute values may be due to
  - faulty data collection instruments
  - data entry problems
  - data transmission problems
  - technology limitation
  - inconsistency in naming convention
- Other data problems which require data cleaning
  - duplicate records
  - incomplete data
  - inconsistent data

How to Handle Noisy Data?

- Binning
  - first sort data and partition into (equal-frequency) bins
  - then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.
- Regression
  - smooth by fitting the data into regression functions
- Clustering
  - detect and remove outliers
- Combined computer and human inspection
  - detect suspicious values and check by human (e.g., deal with possible outliers)

Chapter 3: Data Preprocessing

- Data Preprocessing: An Overview
  - Data Quality
  - Major Tasks in Data Preprocessing
- Data Cleaning
- Data Integration 
- Data Reduction
- Data Transformation and Data Discretization
- Summary

Data Integration

- **Data integration**:
  - Combines data from multiple sources into a coherent store
- Schema integration: e.g., A.cust-id ≡ B.cust-#
  - Integrate metadata from different sources
- Entity identification problem:
  - Identify real world entities from multiple data sources, e.g., Bill Clinton = William Clinton
- Detecting and resolving data value conflicts
  - For the same real world entity, attribute values from different sources are different
  - Possible reasons: different representations, different scales, e.g., metric vs. British units

Entity Resolution for Big Data

Lise Getoor

University of Maryland
College Park, MD

Ashwin Machanavajjhala

Duke University
Durham, NC

http://www.cs.umd.edu/~getoor/Tutorials/ER_KDD2013.pdf
http://goo.gl/7tKiiL

What is Entity Resolution?

Problem of identifying and linking/grouping different manifestations of the same real world object.

Examples of manifestations and objects:

- Different ways of addressing (names, email addresses, FaceBook accounts) the same person in text.
- Web pages with differing descriptions of the same business.
- Different photos of the same object.
- …

What is Entity Resolution?

Article | Talk | Read | Edit | View history | Search

Record linkage

From Wikipedia, the free encyclopedia

(Redirected from Entity resolution)

Record linkage (RL) refers to the task of finding records in a data set that refer to the same entity across different data sources (e.g., data files, books, websites, databases). Record linkage is necessary when joining data sets based on entities that may or may not share a common identifier (e.g., database key, URI, National identification number), as may

Name resolution

From Wikipedia, the free encyclopedia

Coreference

From Wikipedia, the free encyclopedia

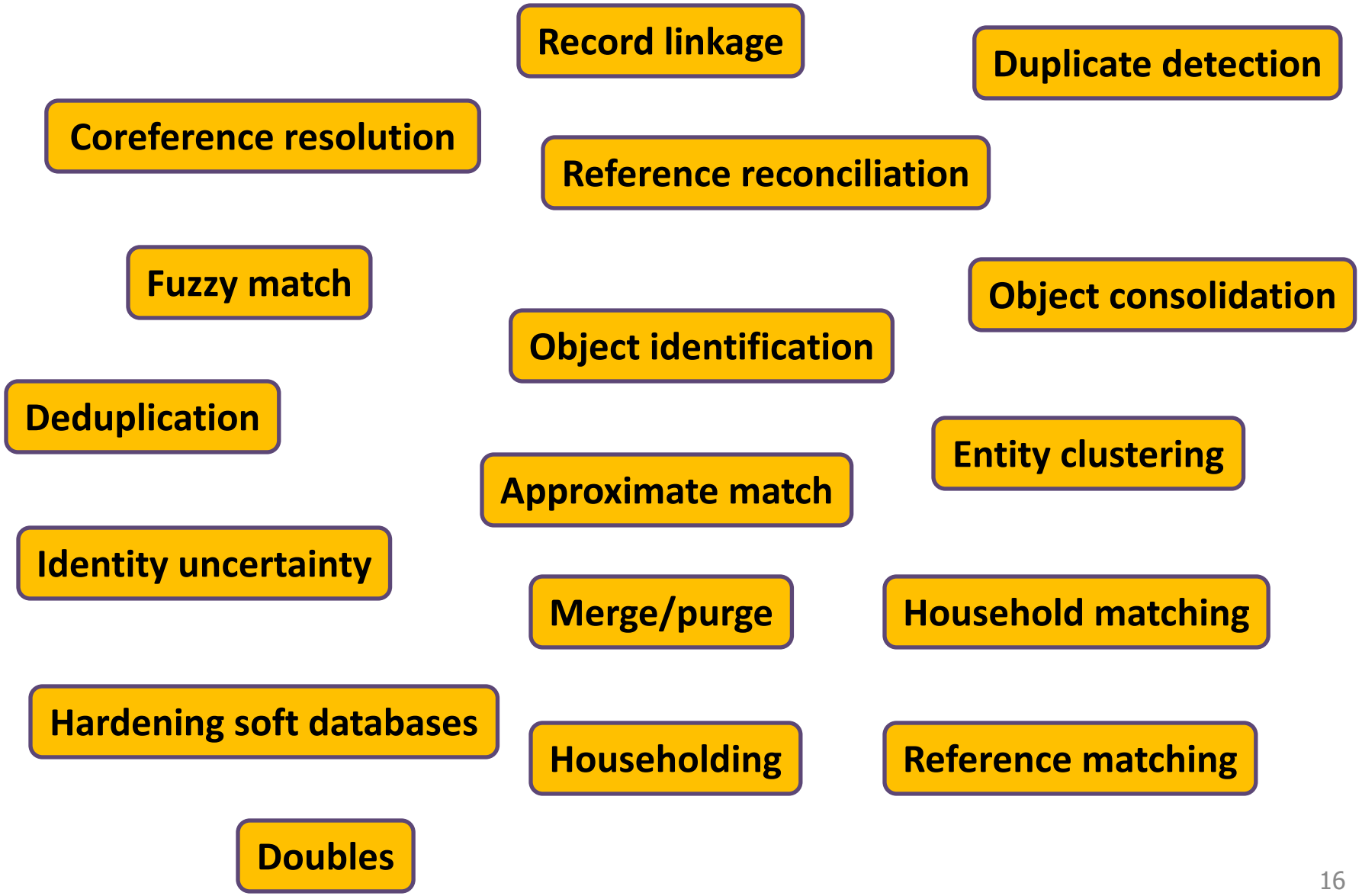
Deduplication

From Wikipedia, the free encyclopedia

Identity resolution

From Wikipedia, the free encyclopedia

Ironically, Entity Resolution has many duplicate names



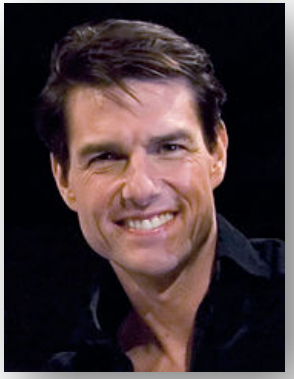
ER Motivating Examples

- *Linking Census Records*
- *Public Health*
- *Web search*
- *Comparison shopping*
- *Counter-terrorism*
- *Knowledge Graph Construction*
- *…*

Traditional Challenges in ER

- Name/Attribute ambiguity

Thomas Cruise



Michael Jordan

Traditional Challenges in ER

- Name/Attribute ambiguity
- Errors due to data entry

↓	C1	C2
	Total Cholesterol_1	Total Cholesterol_2
682	214.4	214.4
683	184.4	184.4
684	183.5	183.5
685	240.7	240.7
686	215.1	215.1
687	198.6	198.6
688	2800.0	280.0
689	210.8	210.8
690	182.5	182.5
691	192.6	192.6

Traditional Challenges in ER

- Name/Attribute ambiguity
- Errors due to data entry
- Missing Values

Exhibit 2: Examples of variables that are set to unknown values

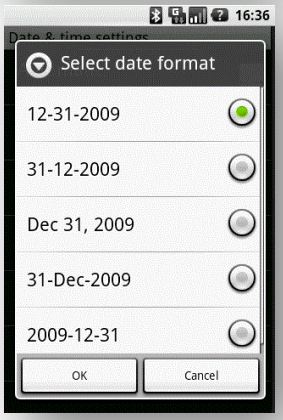
Administrative dates:	set to 0101YY, 010199, 999999
Date of Birth	0101YY, 1506YY, 3006YY, 0107YY, 1507YY, 0101YEAR
Names:	set to spaces, NK, UNKNOWN, or ZZZZ BABY, MALE, FEMALE, TWIN, TRIPLET, INFANT
Other variables:	set to 9, 99, 9999, -1 NK (Not Known) NA (Not applicable) NC (Not coded) U (Unknown)

[Gill et al; Univ of Oxford 2003]

Traditional Challenges in ER

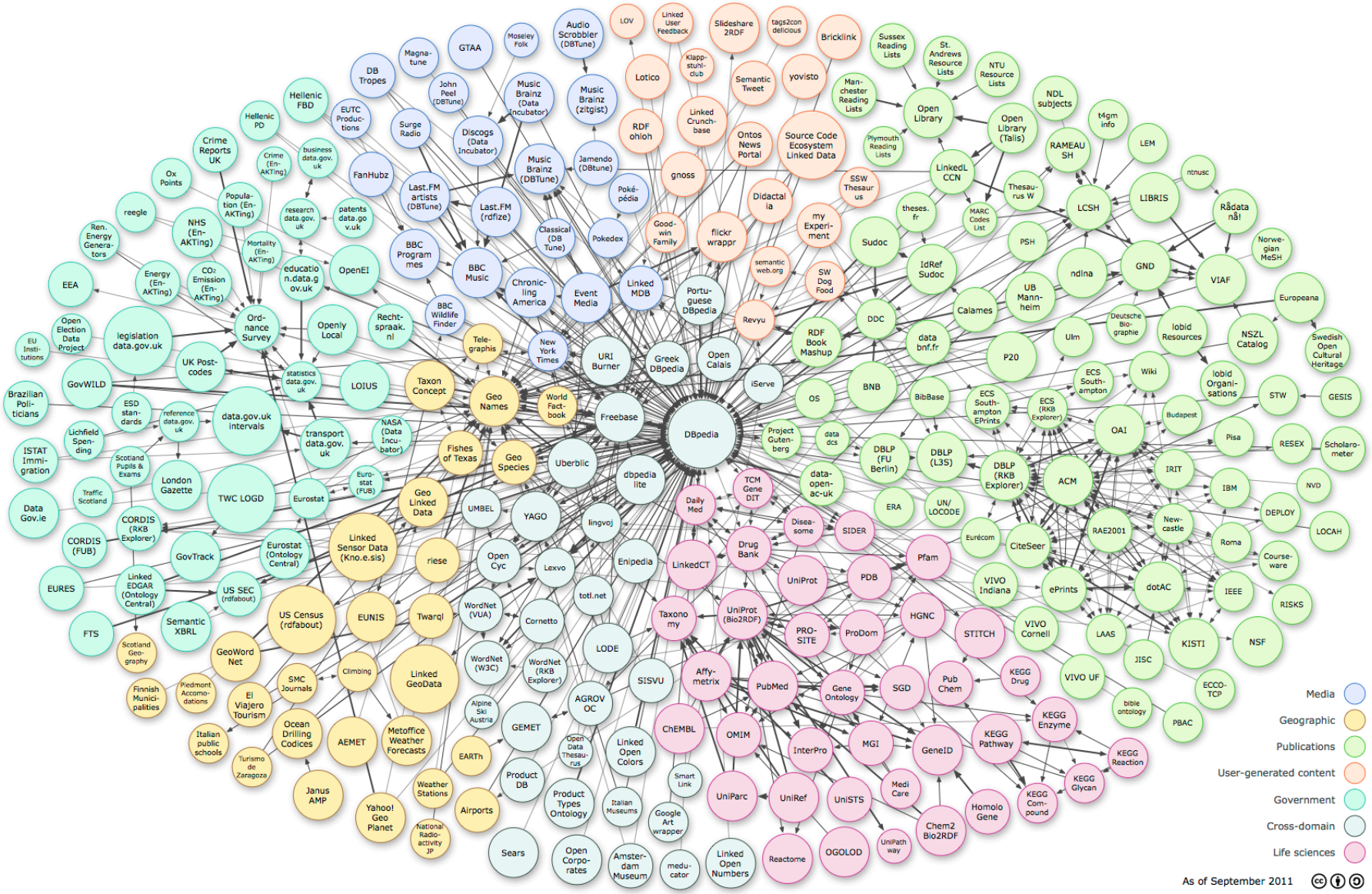
- Name/Attribute ambiguity
- Errors due to data entry
- Missing Values
- Changing Attributes

- Data formatting



- Abbreviations / Data Truncation

Big-Data ER Challenges



Big-Data ER Challenges

- Larger and more Datasets
  - Need efficient parallel techniques
- More Heterogeneity
  - Unstructured, Unclean and Incomplete data. Diverse data types.
  - No longer just matching names with names, but Amazon profiles with browsing history on Google and friends network in Facebook.

Big-Data ER Challenges

- Larger and more Datasets
  - Need efficient parallel techniques
- More Heterogeneity
  - Unstructured, Unclean and Incomplete data. Diverse data types.
- More linked
  - Need to infer relationships in addition to “equality”
- Multi-Relational
  - Deal with structure of entities (Are Walmart and Walmart Pharmacy the same?)
- Multi-domain
  - Customizable methods that span across domains
- Multiple applications (web search versus comparison shopping)
  - Serve diverse application with different accuracy requirements

ER References

- Book / Survey Articles
  - Data Quality and Record Linkage Techniques [T. Herzog, F. Scheuren, W. Winkler, Springer, '07]
  - Duplicate Record Detection [A. Elmagrid, P. Ipeirotis, V. Verykios, TKDE '07]
  - An Introduction to Duplicate Detection [F. Naumann, M. Herschel, M&P synthesis lectures 2010]
  - Evaluation of Entity Resolution Approached on Real-world Match Problems [H. Kopke, A. Thor, E. Rahm, PVLDB 2010]
  - Data Matching [P. Christen, Springer 2012]

- Tutorials
  - Record Linkage: Similarity measures and Algorithms [N. Koudas, S. Sarawagi, D. Srivatsava SIGMOD '06]
  - Data fusion--Resolving data conflicts for integration [X. Dong, F. Naumann VLDB '09]
  - Entity Resolution: Theory, Practice and Open Challenges http://goo.gl/Ui38o [L. Getoor, A. Machanavajjhala AAAI '12]

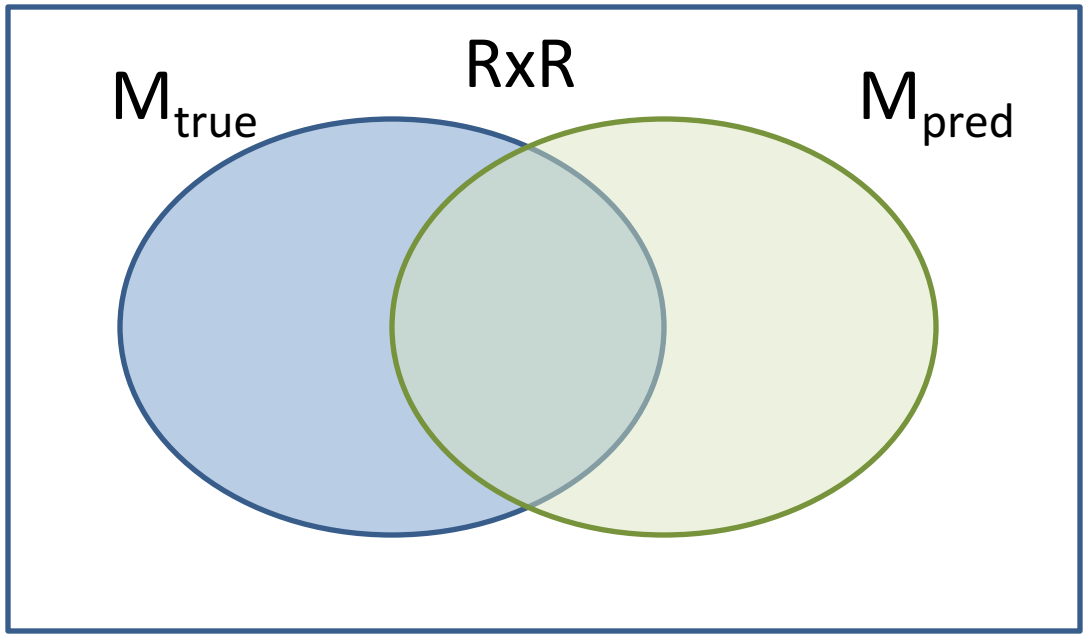
Notation

- *R*: set of records / mentions (typed)
- *H*: set of relations / hyperedges (typed)
- *M*: set of *matches* (record pairs that correspond to same entity)
- *N*: set of *non-matches* (record pairs corresponding to different entities)
- *E*: set of entities
- *L*: set of links

- True (M_{true}, N_{true}, E_{true}, L_{true}): according to real world
  vs Predicted (M_{pred}, N_{pred}, E_{pred}, L_{pred}): by algorithm

Relationship between M_{true} and M_{pred}

- M_{true} (SameAs , Equivalence)
- M_{pred} (Similar representations and similar attributes)

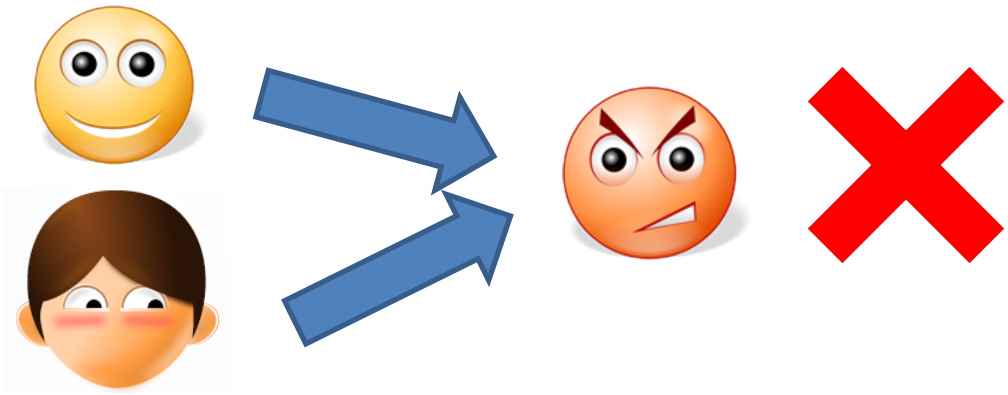


Metrics

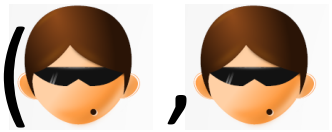
- Pairwise metrics
  - Precision/Recall, F1
  - # of predicted matching pairs
- Cluster level metrics
  - purity, completeness, complexity
  - Precision/Recall/F1: Cluster-level, closest cluster, MUC, B^3 , Rand Index
  - Generalized merge distance [Menestrina et al, PVLDB10]
- Little work that evaluates correct prediction of links

Typical Assumptions Made

- *Each record/mention is associated with a single real world entity.*



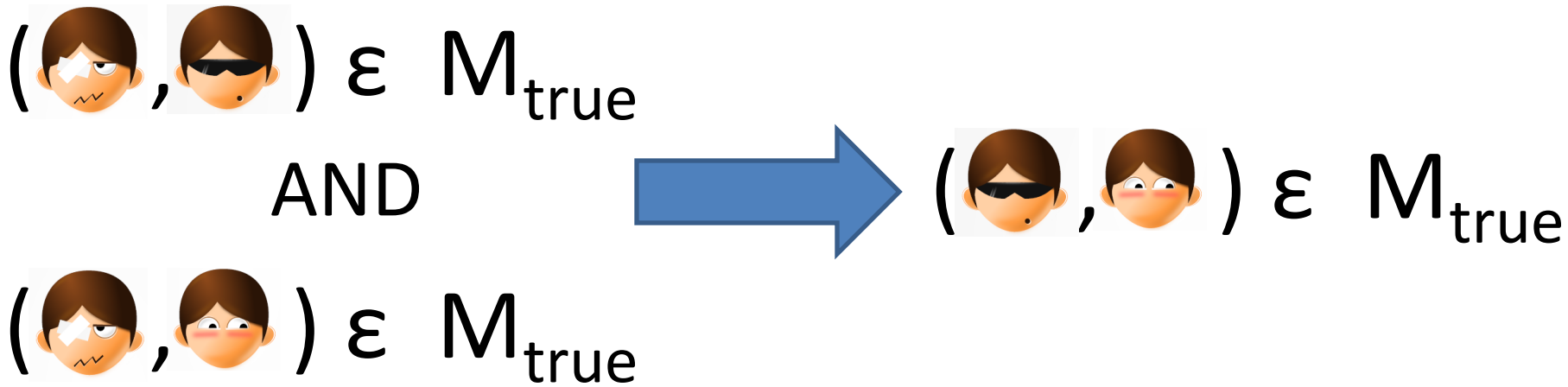
- *In record linkage, no duplicates in the same source*
- *If two records/mentions are identical, then they are true matches*

 ε M_{true}

ER versus Classification

Finding matches vs non-matches is a classification problem

- Imbalanced: typically O(R) matches, O(R^2) non-matches
- Instances are pairs of records. Pairs are not IID



ER vs (Multi-relational) Clustering

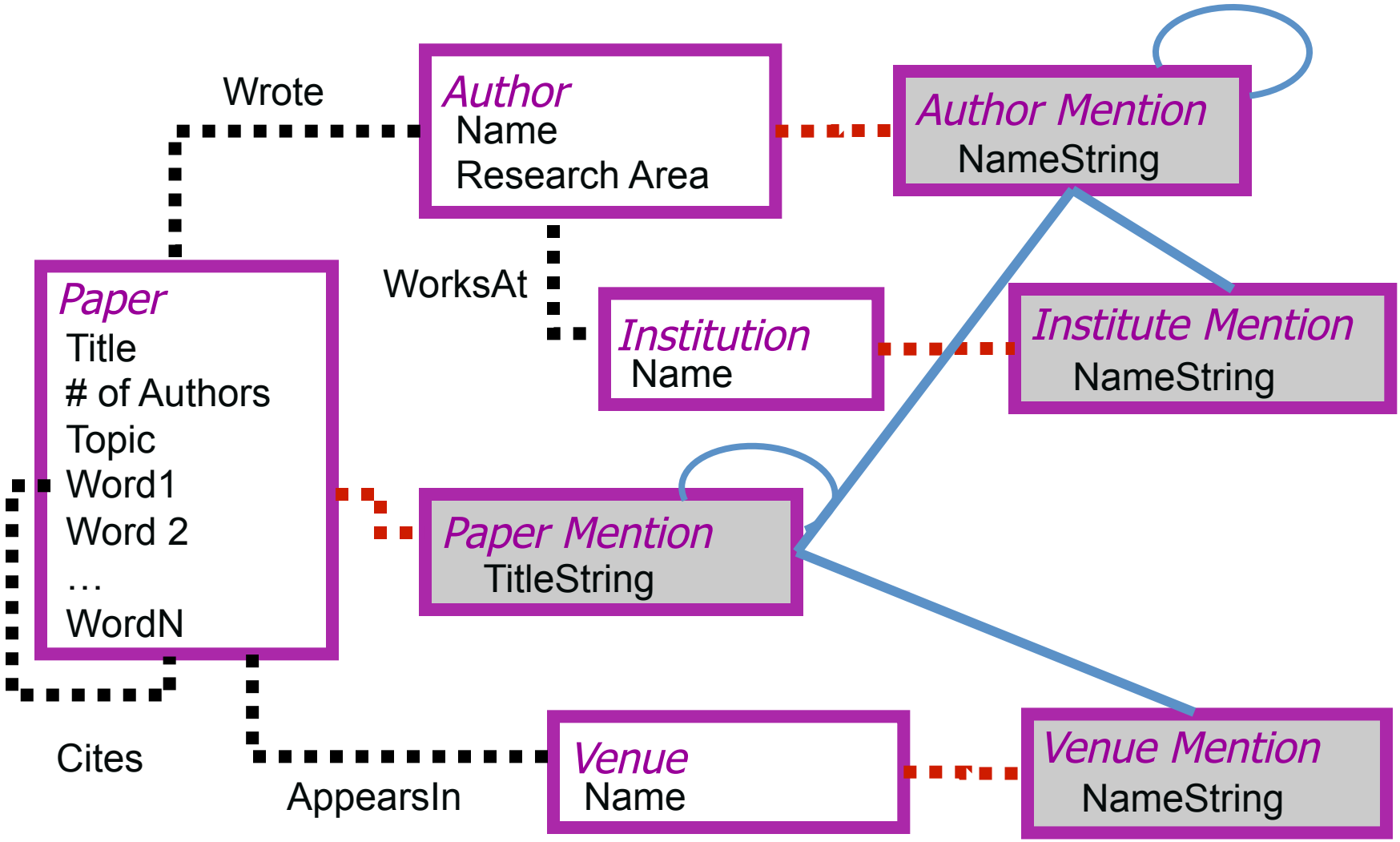
Computing entities from records is a clustering problem

- In typical clustering algorithms (k-means, LDA, etc.) *number of clusters is a constant or sub linear in R.*

- In ER: *number of clusters is linear in R, and average cluster size is a constant. Significant fraction of clusters are singletons.*

MOTIVATING EXAMPLE: BIBLIOGRAPHIC DOMAIN

Entities & Relations in Bibliographic Domain



PART 2-a

DATA PREPARATION & MATCH FEATURES

Normalization

- Schema normalization
  - Schema Matching – e.g., contact number and phone number
  - Compound attributes – full address vs str,city,state,zip
  - Nested attributes
    - List of features in one dataset (air conditioning, parking) vs each feature a boolean attribute
  - Set valued attributes
    - Set of phones vs primary/secondary phone
  - Record segmentation from text
- Data normalization
  - Often convert to all lower/all upper; remove whitespace
  - detecting and correcting values that contain known typographical errors or variations,
  - expanding abbreviations and replacing them with standard forms; replacing nicknames with their proper name forms
  - Usually done based on dictionaries (e.g., commercial dictionaries, postal addresses, etc.)

Normalization

- Schema normalization
  - Schema Matching – e.g., contact number and phone number
  - Compound attributes – full address vs str,city,state,zip
  - Nested attributes
    - List of features in one dataset (air cond… …each feature a boolean attribute
  - Set valued attributes
    - Set of phones v… …phone
  - Record segm…
- Data nor…
  - Often c… …all lower/all upper; remove whitespace
  - detecting and correcting values that contain known typographical errors or variations,
  - expanding abbreviations and replacing them with standard forms; replacing nicknames with their proper name forms
  - Usually done based on dictionaries (e.g., commercial dictionaries, postal addresses, etc.)

Initial data prep big part of the work; smart normalization can go long way!

Matching Features

- For two references x and y, compute a "comparison" vector of *similarity scores* of component attribute.
  - [1st-author-match-score,
    paper-match-score,
    venue-match-score,
    year-match-score, ….]

- Similarity scores
  - Boolean (match or not-match)
  - Real values based on distance functions

Summary of Matching Features

Handle Typographical errors

- Equality on a boolean predicate
- Edit distance
  - Levenstein, Smith-Waterman, Affine
- Set similarity
  - Jaccard, Dice
- Vector Based
  - Cosine similarity, TFIDF

Good for Text like reviews/ tweets

Good for Names

- Alignment-based or Two-tiered
  - Jaro-Winkler, Soft-TFIDF, Monge-Elkan
- Phonetic Similarity
  - Soundex
- Translation-based
- Numeric distance between values
- Domain-specific

Useful for abbreviations, alternate names.

- Useful packages:
  - SecondString, http://secondstring.sourceforge.net/
  - Simmetrics: http://sourceforge.net/projects/simmetrics/
  - LingPipe, http://alias-i.com/lingpipe/index.html

Relational Matching Features

- Relational features are often set-based
  - Set of coauthors for a paper
  - Set of cities in a country
  - Set of products manufactured by manufacturer

- Can use set similarity functions mentioned earlier
  - Common Neighbors: Intersection size
  - Jaccard's Coefficient: Normalize by union size
  - Adar Coefficient: Weighted set similarity

- Can reason about similarity in sets of values
  - Average or Max
  - Other aggregates

PART 2-b

PAIRWISE MATCHING

Pairwise Match Score

Problem: Given a vector of component-wise similarities for a pair of records (x,y), compute P(x and y match).

Solutions:

1. Weighted sum or average of component-wise similarity scores. Threshold determines match or non-match.
   - 0.5*1st-author-match-score + 0.2*venue-match-score + 0.3*paper-match-score.
   - Hard to pick weights.
     - Match on last name match *more predictive* than login name.
     - Match on "Smith" *less predictive* than match on "Getoor" or "Machanavajjhala".
   - Hard to tune a threshold.

Pairwise Match Score

Problem: Given a vector of component-wise similarities for a pair of records (x,y), compute P(x and y match).

Solutions:

1. Weighted sum or average of component-wise similarity scores. Threshold determines match or non-match.
2. Formulate rules about what constitutes a match.
   - (1^{st}-author-match-score > 0.7 AND venue-match-score > 0.8) OR (paper-match-score > 0.9 AND venue-match-score > 0.9)
   - Manually formulating the right set of rules is hard.

Fellegi & Sunter Model [FS, Science '69]

- Record pair: *r = (x,y)* in A x B

- $\gamma = \gamma(r)$ is a comparison vector
  - E.g., γ = ["Is x.name = y.name?", "Is x.address = y.address?" …]
  - Assume binary vector for simplicity

- *M* : set of matching pairs of records
- *U* : set of non-matching pairs of records

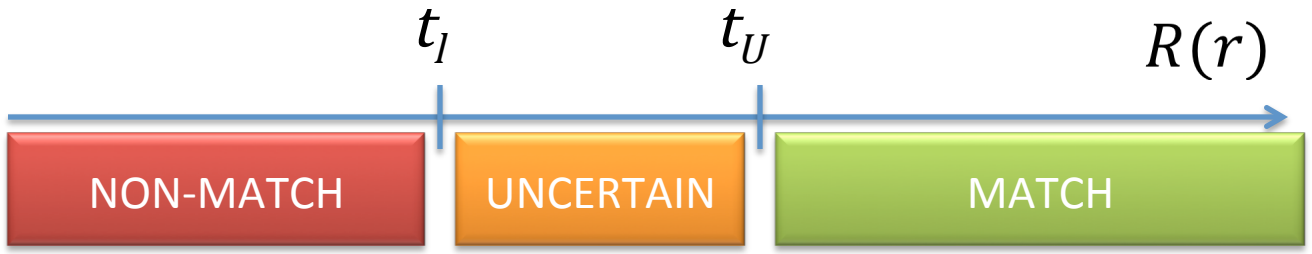
Fellegi & Sunter Model [FS, Science '69]

- *r = (x,y)* is record pair, γ is comparison vector, *M* matches, *U* non-matches

- Linkage decisions are based on:

$$R(r) = \frac{m(\gamma)}{u(\gamma)} = \frac{P(\gamma \mid r \in M)}{P(\gamma \mid r \in U)}$$

- **Linkage Rule: L(t_l, t_u)**



Error due to a Linkage Rule

- Type I Error: *r = (x,y)* in U, but the linkage rule calls it a match

$$P(L_{match}|U) = \sum_{\gamma \epsilon \Gamma} u(\gamma) \cdot P(L_{match}|\gamma)$$

- Type II Error: *r = (x,y)* in M, but the linkage rule calls it a non-match

$$P(L_{non}|M) = \sum_{\gamma \epsilon \Gamma} m(\gamma) \cdot P(L_{non}|\gamma)$$

Optimal Linkage Rule

- $L^* = (t_l^*, t_u^*)$ is an optimal decision rule for comparison space Γ with error bounds μ and λ, if
  - L^* meets the type I and type II requirements

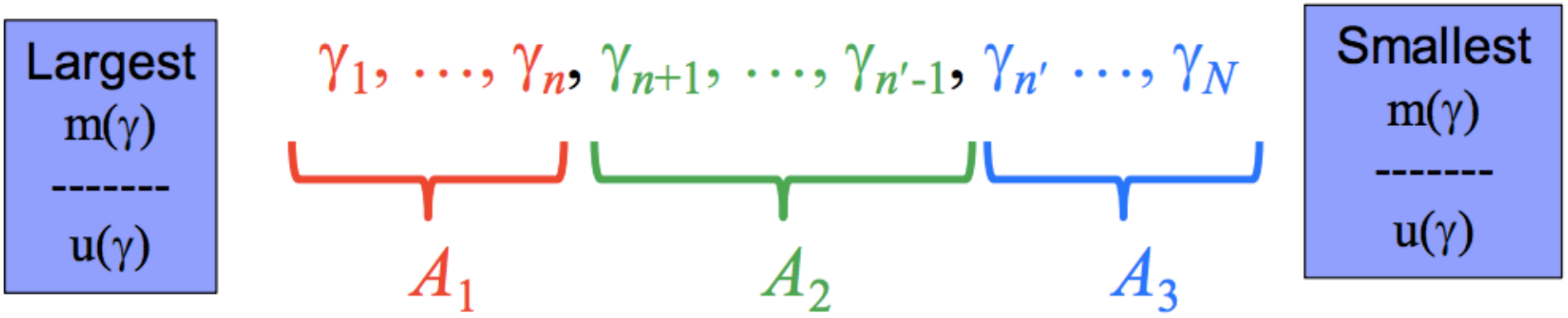
$$P(L_{match}|U) \leq \mu, \quad P(L_{non}|M) \leq \lambda$$

  - L^* has the least conditional probabilities of *not making a decision.* That is for all other decision rules L (with error bounds μ and λ),

$$P(L^*_{uncertain}|U) \leq P(L_{uncertain}|U)$$
$$P(L^*_{uncertain}|M) \leq P(L_{uncertain}|M)$$

Finding the Optimal Linkage Rule

- Suppose there are N comparison vectors
- Sort them in decreasing order of $m(\gamma)$ / $u(\gamma)$



- Pick the largest n and n' such that:

$$\mu \geq \sum_{i=1}^{n} u(\gamma_i), \qquad \lambda \geq \sum_{i=1}^{n} m(\gamma_i)$$

Using Fellegi Sunter in Practice

- Γ is usually high dimensional (computing m(γ) and u(γ) is inefficient)
  - Use conditional independence of features in γ given match or non-match
  - Naïve Bayes assumption

- Computing P(γ | r ε M) requires some knowledge of matches.
  - Supervised learning (assume a training set is provided)
  - EM-based techniques can used to learn the parameters jointly while identifying matches.

ML Pairwise Approaches

- Supervised machine learning algorithms
  - Decision trees
    - [Cochinwala et al, IS01]
  - Support vector machines
    - [Bilenko & Mooney, KDD03]; [Christen, KDD08]
  - Ensembles of classifiers
    - [Chen et al., SIGMOD09]
  - Conditional Random Fields (CRF)
    - [Gupta & Sarawagi, VLDB09]
  - … and many others.

- Issues:
  - **Training set generation**
  - Imbalanced classes – many more negatives than positives (even after eliminating obvious non-matches … using *Blocking*)
  - Misclassification cost

Creating a Training Set is a key issue

- Constructing a training set is hard – since most pairs of records are "easy non-matches".
  - 100 records from 100 cities.
  - Only 10^6 pairs out of total 10^8 (1%) come from the same city

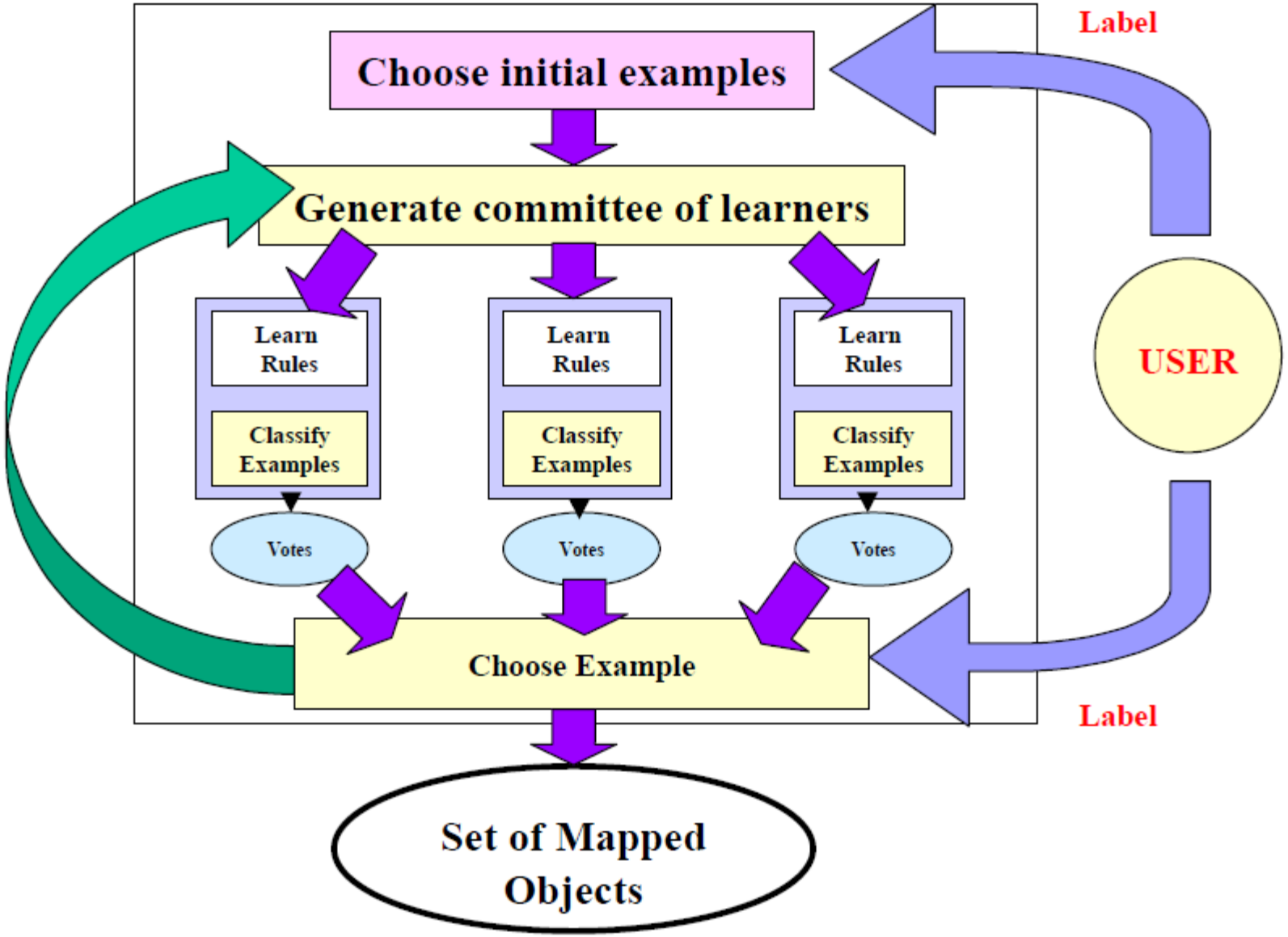
- Some pairs are hard to judge even by humans
  - Inherently ambiguous
    - E.g., Paris Hilton (person or business)
  - Missing attributes
    - Starbucks, Toronto vs Starbucks, Queen Street ,Toronto

Avoiding Training Set Generation

- Unsupervised / Semi-supervised Techniques
  - EM based techniques to learn parameters
    - [Winkler '06, Herzog et al '07]
  - Generative Models
    - [Ravikumar & Cohen, UAI04]

- Active Learning
  - Committee of Classifiers
    - [Sarawagi et al KDD '00, Tajeda et al IS '01]
  - Provably optimizing precision/recall
    - [Arasu et al SIGMOD '10, Bellare et al KDD '12]
  - Crowdsourcing
    - [Wang et al VLDB '12, Marcus et al VLDB '12, …]

Committee of Classifiers [Tejada et al, IS '01]



Active Learning with Provable Guarantees

- Most active learning techniques minimize 0-1 loss
  [Beygelzimer et al NIPS 2010].

$$\textbf{minimize } \frac{fn(h) + fp(h)}{n}$$

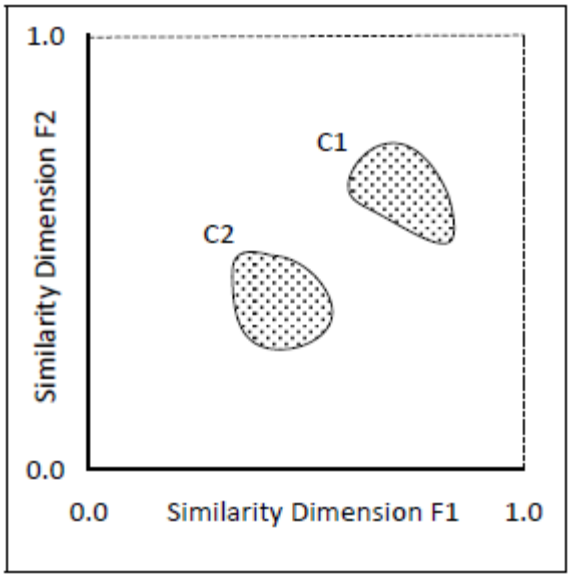
- However, ER is very imbalanced:
  - Number of non-matches > 100 * number of matches.
  - Classifying all pairs as "non-matches" has low 0-1 loss (< 1%).

- Hence, need active learning techniques that maximize precision/ recall.

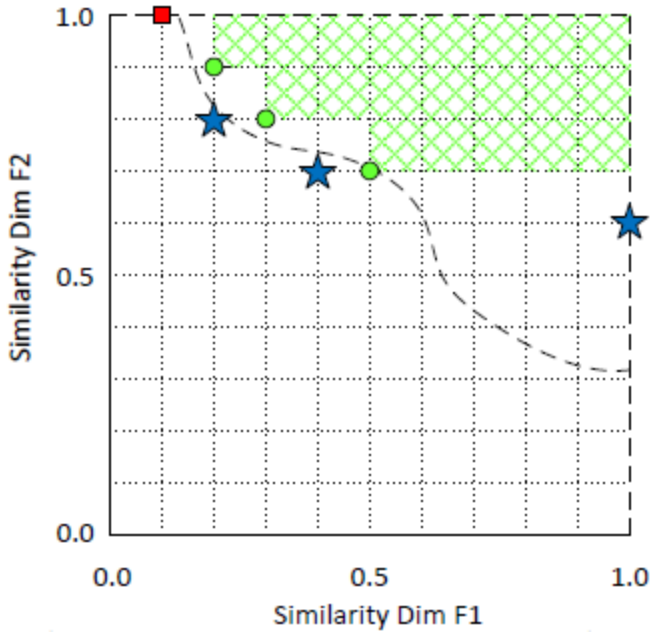
$$\begin{aligned} &\textbf{maximize} && recall(h) \\ &\textbf{subject to} && precison(h) \geq \tau \end{aligned}$$

Active Learning with Provable Guarantees

- Monotonicity of Precision [Arasu et al SIGMOD '10]



There is a larger fraction of matches in C1 than in C2.



Algorithm searches for the optimal classifier using binary search on each dimension

Active Learning with Provable Guarantees

[Bellare et al KDD '12]

$O(\log^2 n)$ calls to a blackbox 0-1 loss active learning algorithm.

Exponentially smaller label complexity than [Arasu et al SIGMOD '10] (in the worst case).

1. Precision Constrained → Weighted 0-1 Loss Problem (using a Lagrange Multiplier λ).
2. Given a fixed value for λ, weighted 0-1 Loss can be optimized by (one call to) a blackbox active learning classifier.
3. Right value of λ is computed by searching over all optimal classifiers.
   - Classifiers are embedded in a 2-d plane (precision/recall)
   - Search is along the convex hull of the embedded classifiers

Crowdsourcing

- Growing interest in integrating human computation in declarative workflow engines.
  - ER is an important problem (e.g., for evaluating fuzzy joins)
  - [Wang et al VLDB '12, Marcus et al VLDB '12, …]

- Opportunity: utilize crowdsourcing for creating training sets, or for active learning.

- Key open issue: Handling errors in human judgments
  - In an experiment on Amazon Mechanical Turk:
    - Pairwise matching judgment, each given to 5 different people
  - Majority of workers agreed on truth on only 90% of pairwise judgments.

Summary of Single-Entity ER Algorithms

- Many algorithms for independent classification of pairs of records as match/non-match

- ML based classification & Fellegi-Sunter
  - Pro: Advanced state of the art
  - Con: Building high fidelity training sets is a hard problem

- Active Learning & Crowdsourcing for ER are active areas of research.

Handling Redundancy in Data Integration

- Redundant data occur often when integration of multiple databases
  - *Object identification*: The same attribute or object may have different names in different databases
  - *Derivable data:* One attribute may be a "derived" attribute in another table, e.g., annual revenue
- Redundant attributes may be able to be detected by *correlation analysis* and *covariance analysis*
- Careful integration of the data from multiple sources may help reduce/avoid redundancies and inconsistencies and improve mining speed and quality

Correlation Analysis (Nominal Data)

- **X^2 (chi-square) test**

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

- The larger the X^2 value, the more likely the variables are related
- The cells that contribute the most to the X^2 value are those whose actual count is very different from the expected count
- Correlation does not imply causality
  - # of hospitals and # of car-theft in a city are correlated
  - Both are causally linked to the third variable: population

Chi-Square Calculation: An Example

	Play chess	Not play chess	Sum (row)
Like science fiction	250(90)	200(360)	450
Not like science fiction	50(210)	1000(840)	1050
Sum(col.)	300	1200	1500

- X^2 (chi-square) calculation (numbers in parenthesis are expected counts calculated based on the data distribution in the two categories)

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93$$

- It shows that like_science_fiction and play_chess are correlated in the group

Correlation Analysis (Numeric Data)

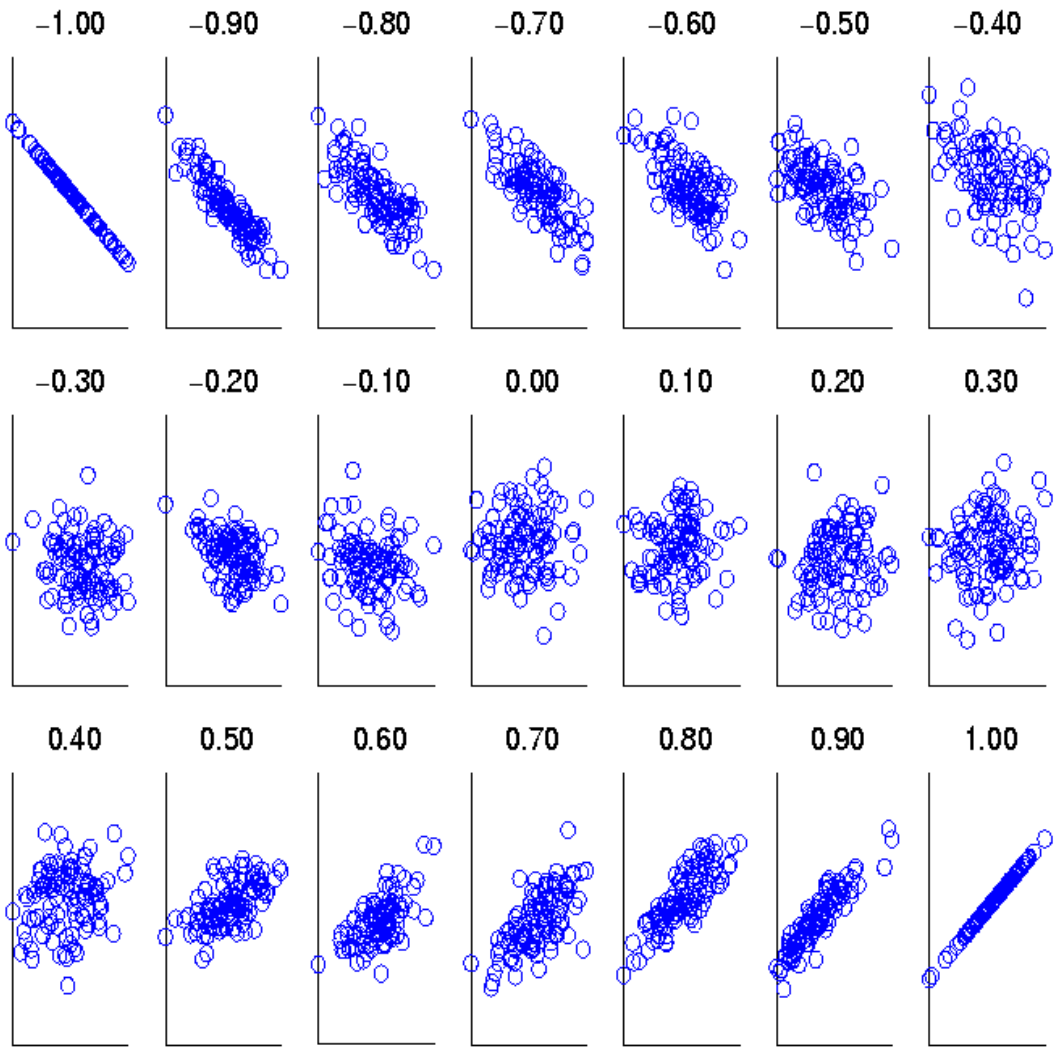
- Correlation coefficient (also called Pearson's product moment coefficient)

$$r_{A,B} = \frac{\sum_{i=1}^{n}(a_i - \bar{A})(b_i - \bar{B})}{(n-1)\sigma_A\sigma_B} = \frac{\sum_{i=1}^{n}(a_i b_i) - n\bar{A}\bar{B}}{(n-1)\sigma_A\sigma_B}$$

where n is the number of tuples, $\bar{A}$ and $\bar{B}$ are the respective means of A and B, σ_A and σ_B are the respective standard deviation of A and B, and $\Sigma(a_i b_i)$ is the sum of the AB cross-product.

- If $r_{A,B} > 0$, A and B are positively correlated (A's values increase as B's). The higher, the stronger correlation.
- $r_{A,B} = 0$: independent; $r_{AB} < 0$: negatively correlated

Visually Evaluating Correlation



Scatter plots showing the similarity from –1 to 1.

Correlation (viewed as linear relationship)

- Correlation measures the linear relationship between objects
- To compute correlation, we standardize data objects, A and B, and then take their dot product

$$a'_k = (a_k - mean(A)) / std(A)$$

$$b'_k = (b_k - mean(B)) / std(B)$$

$$correlation(A, B) = A' \bullet B'$$

Covariance (Numeric Data)

- Covariance is similar to correlation

$$Cov(A, B) = E((A - \bar{A})(B - \bar{B})) = \frac{\sum_{i=1}^{n}(a_i - \bar{A})(b_i - \bar{B})}{n}$$

Correlation coefficient: $$r_{A,B} = \frac{Cov(A, B)}{\sigma_A \sigma_B}$$

where n is the number of tuples, $\bar{A}$ and $\bar{B}$ are the respective mean or **expected values** of A and B, σ_A and σ_B are the respective standard deviation of A and B.

- **Positive covariance**: If $Cov_{A,B} > 0$, then A and B both tend to be larger than their expected values.
- **Negative covariance**: If $Cov_{A,B} < 0$ then if A is larger than its expected value, B is likely to be smaller than its expected value.
- **Independence**: $Cov_{A,B} = 0$ but the converse is not true:
  - Some pairs of random variables may have a covariance of 0 but are not independent. Only under some additional assumptions (e.g., the data follow multivariate normal distributions) does a covariance of 0 imply independence

Co-Variance: An Example

$$Cov(A,B) = E((A-\bar{A})(B-\bar{B})) = \frac{\sum_{i=1}^{n}(a_i-\bar{A})(b_i-\bar{B})}{n}$$

- It can be simplified in computation as

$$Cov(A,B) = E(A \cdot B) - \bar{A}\bar{B}$$

- Suppose two stocks A and B have the following values in one week: (2, 5), (3, 8), (5, 10), (4, 11), (6, 14).
- Question: If the stocks are affected by the same industry trends, will their prices rise or fall together?
  - E(A) = (2 + 3 + 5 + 4 + 6)/ 5 = 20/5 = 4
  - E(B) = (5 + 8 + 10 + 11 + 14) /5 = 48/5 = 9.6
  - Cov(A,B) = (2×5+3×8+5×10+4×11+6×14)/5 – 4 × 9.6 = 4
- Thus, A and B rise together since Cov(A, B) > 0.

Chapter 3: Data Preprocessing

- Data Preprocessing: An Overview
  - Data Quality
  - Major Tasks in Data Preprocessing
- Data Cleaning
- Data Integration
- Data Reduction 
- Data Transformation and Data Discretization
- Summary

Data Reduction Strategies

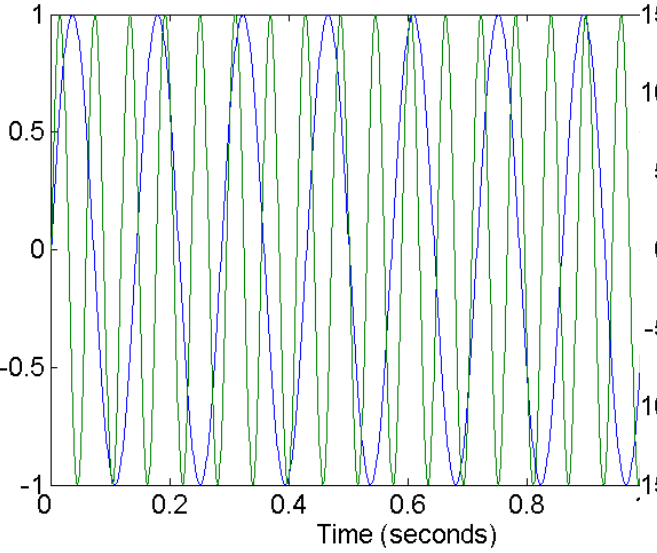
- **Data reduction**: Obtain a reduced representation of the data set that is much smaller in volume but yet produces the same (or almost the same) analytical results
- Why data reduction? — A database/data warehouse may store terabytes of data. Complex data analysis may take a very long time to run on the complete data set.
- Data reduction strategies
  - Dimensionality reduction, e.g., remove unimportant attributes
    - Wavelet transforms
    - Principal Components Analysis (PCA)
    - Feature subset selection, feature creation
  - Numerosity reduction (some simply call it: Data Reduction)
    - Regression and Log-Linear Models
    - Histograms, clustering, sampling
    - Data cube aggregation
  - Data compression

Data Reduction 1: Dimensionality Reduction

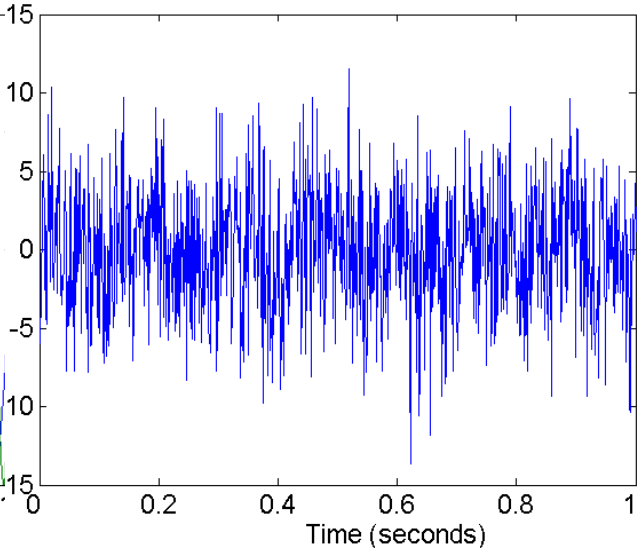
- **Curse of dimensionality**
  - When dimensionality increases, data becomes increasingly sparse
  - Density and distance between points, which is critical to clustering, outlier analysis, becomes less meaningful
  - The possible combinations of subspaces will grow exponentially
- **Dimensionality reduction**
  - Avoid the curse of dimensionality
  - Help eliminate irrelevant features and reduce noise
  - Reduce time and space required in data mining
  - Allow easier visualization
- **Dimensionality reduction techniques**
  - Wavelet transforms
  - Principal Component Analysis
  - Supervised and nonlinear techniques (e.g., feature selection)

Mapping Data to a New Space

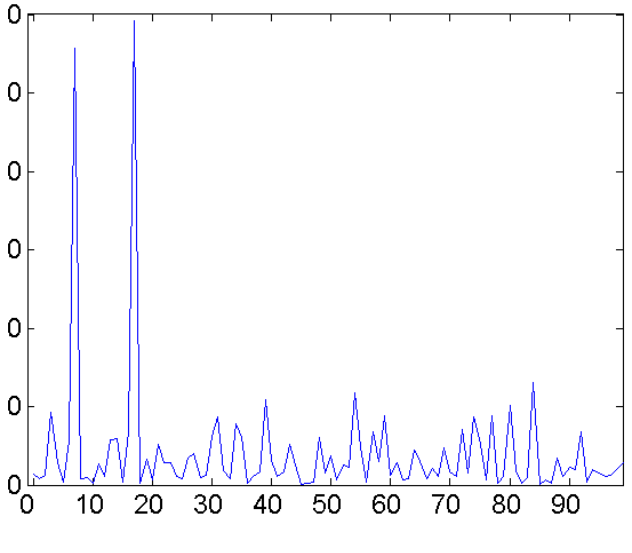
- **Fourier transform**
- **Wavelet transform**



Two Sine Waves



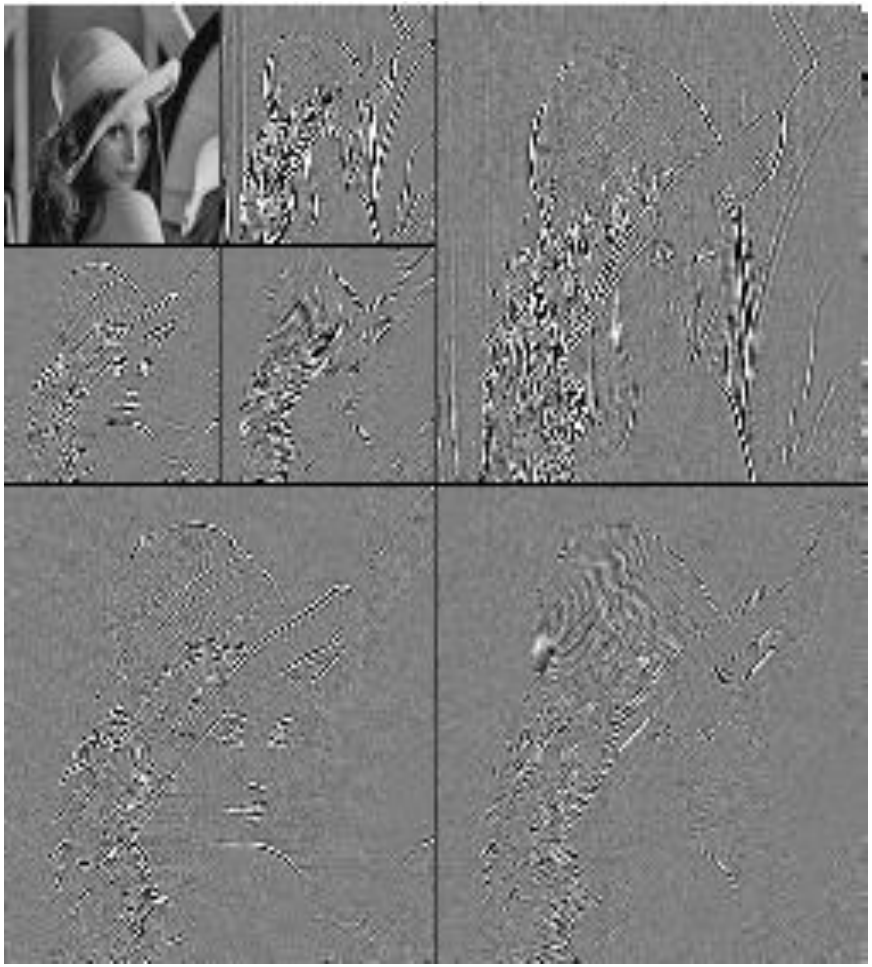
Two Sine Waves + Noise



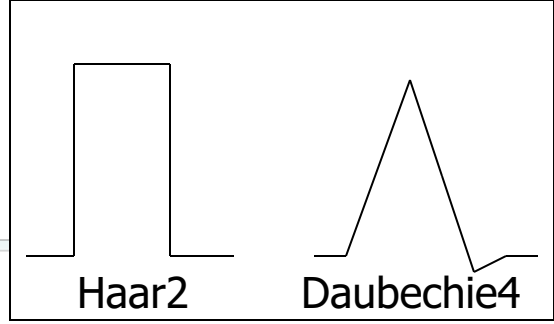
Frequency

What Is Wavelet Transform?

- Decomposes a signal into different frequency subbands
  - Applicable to n-dimensional signals
- Data are transformed to preserve relative distance between objects at different levels of resolution
- Allow natural clusters to become more distinguishable
- Used for image compression



Wavelet Transformation



- Discrete wavelet transform (DWT) for linear signal processing, multi-resolution analysis
- Compressed approximation: store only a small fraction of the strongest of the wavelet coefficients
- Similar to discrete Fourier transform (DFT), but better lossy compression, localized in space
- Method:
  - Length, L, must be an integer power of 2 (padding with 0's, when necessary)
  - Each transform has 2 functions: smoothing, difference
  - Applies to pairs of data, resulting in two set of data of length L/2
  - Applies two functions recursively, until reaches the desired length

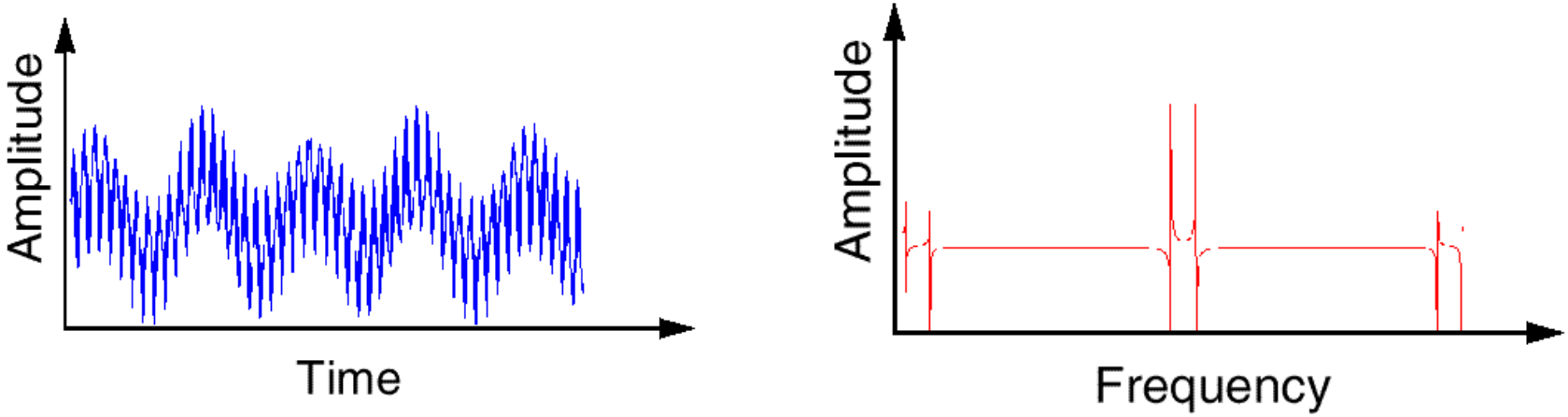
Introduction to Wavelets

http://www.math.tau.ac.il/~turkel/notes/wavelets.pdf

Background

Fourier Analysis

- Breaks down a signal into constituent sinusoids of different frequencies



In other words: Transform the view of the signal from time-base to frequency-base.

What's wrong with Fourier?

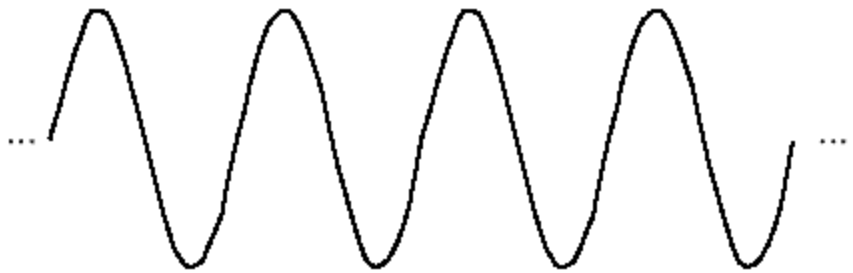
$$F(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t)\, e^{-j\omega t}\, d\, t\ ,$$

- **FT is only the function of ω , frequency.**
- By using Fourier Transform , we loose the time information : WHEN did a particular event take place ?
- **Fourier Transform** can not locate drift, trends, abrupt changes, beginning and ends of events, etc.
- **Fourier Transform is unable to pick out local frequency content. It has a "hard time" representing functions that are oscillatory or discontinuous (Gibbs phenomena)**
- **Frequency domain** ☺
- **Temporal domain** ☹

Wavelet Analysis

What is Wavelet Analysis ?

- And…what is a wavelet…?



Sine Wave

Wavelet (db10)

- A wavelet is a waveform of effectively limited duration that has an average value of zero.

Wavelet's properties

- Short time localized waves with zero integral value.
- Possibility of time shifting.
- Flexibility.

Wavelet Analysis Detail

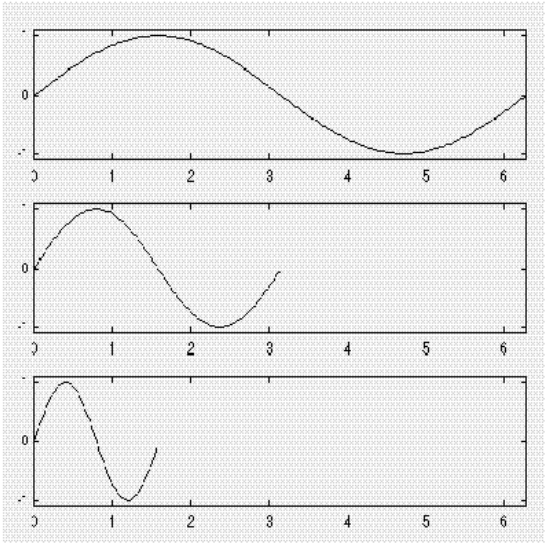
- Wavelet transform decomposes a signal into a set of basis functions (wavelets)
- Wavelets are obtained from a single prototype wavelet Ψ(t) called mother wavelet by dilations and shifting:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$$

- where a is the scaling parameter and b is the shifting parameter

Scaling

- Wavelet analysis produces a time-scale view of the signal.
- *Scaling* means stretching or compressing of the signal.
- scale factor (a) for sine waves:



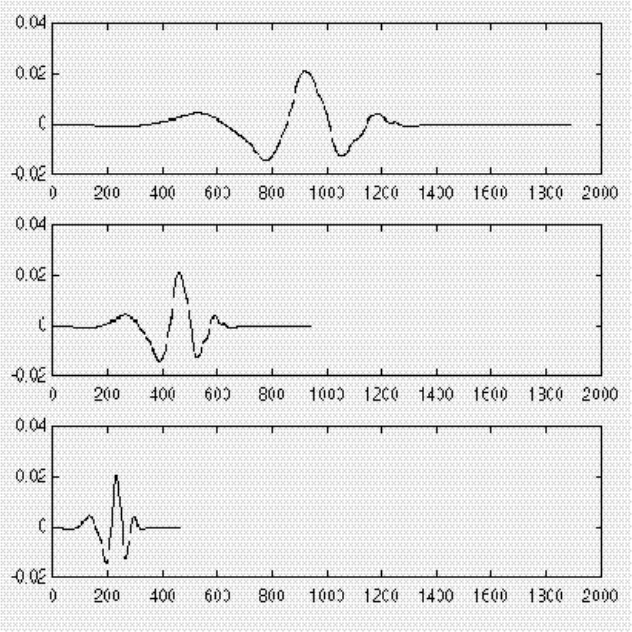
$$f(t) = \sin(t) \;\; ; \;\; a = 1$$

$$f(t) = \sin(2t) \;\; ; \;\; a = 1/2$$

$$f(t) = \sin(4t) \;\; ; \;\; a = 1/4$$

Scaling

- Scale factor works exactly the same with wavelets:



$$f(t) = \Psi(t) \;\; ; \;\; a = 1$$

$$f(t) = \Psi(2t) \;\; ; \;\; a = 1/2$$

$$f(t) = \Psi(4t) \;\; ; \;\; a = 1/4$$

Wavelet function

$$\Psi_{a,b}(x) = \frac{1}{\sqrt{a}} \Psi\left(\frac{x-b}{a}\right)$$

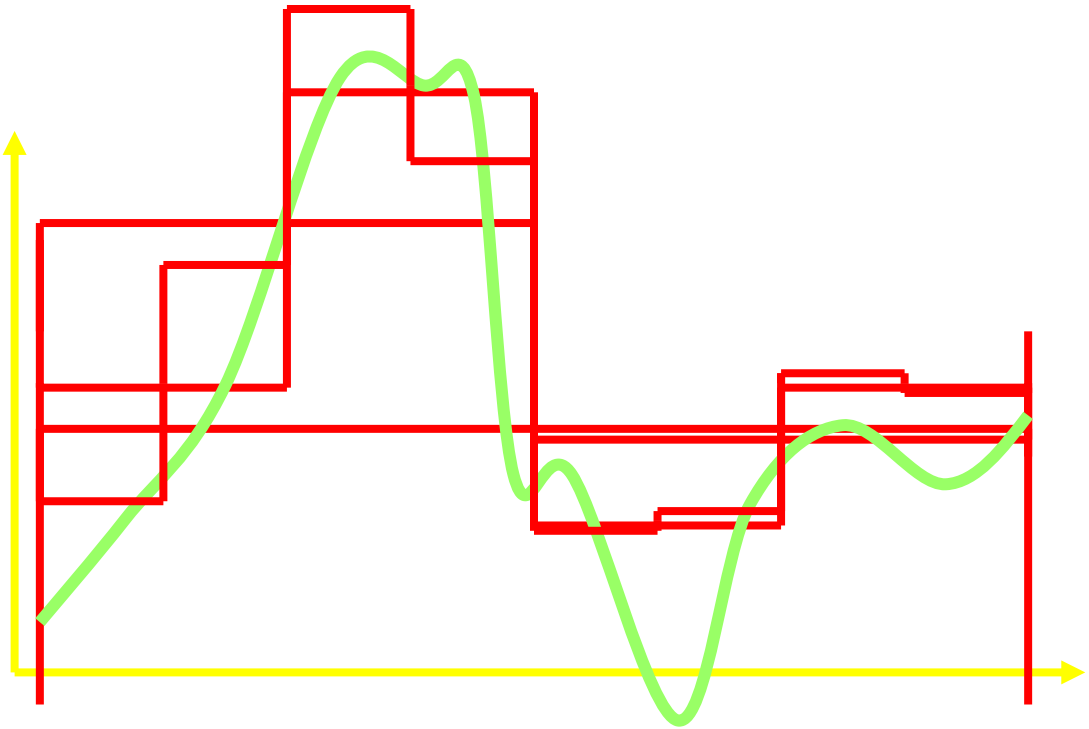
- b – shift coefficient
- a – scale coefficient

$$\Psi_{a,b_x,b_y}(x,y) = \frac{1}{|a|} \Psi\left(\frac{x-b_x}{a}, \frac{y-b_y}{a}\right)$$

- 2D function

Haar transform

- A Haar wavelet is the simplest type of wavelet.
- In discrete form, Haar wavelets are related to a mathematical operation called the Haar transform.
- The Haar transform serves as a prototype for all other wavelet transforms.



Haar transform

- Like all wavelet transforms, the Haar transform decomposes a discrete signal into two subsignals of half its length.
- One subsignal is a running average or trend; the other subsignal is a running difference or fluctuation.

Haar Transform Example

1. Find the average of each pair of samples. Fill the first half of the array with averages. **(first trend subsignal)**

Original Signal	**1**	**3**	**5**	**7**
1. Iteration	**2**	**6**		

(1+3)/2 = 2 **(5+7)/2 = 6**

Haar Wavelet Transform

1. Find the average of each pair of samples. Fill the first half of the array with averages.
2. Find the difference between the average and the samples. Fill the second half of the array with differences. **(first fluctuation subsignal)**

Original Signal	**1**	**3**	**5**	**7**
1. Iteration	**2**	**6**	**-1**	**-1**

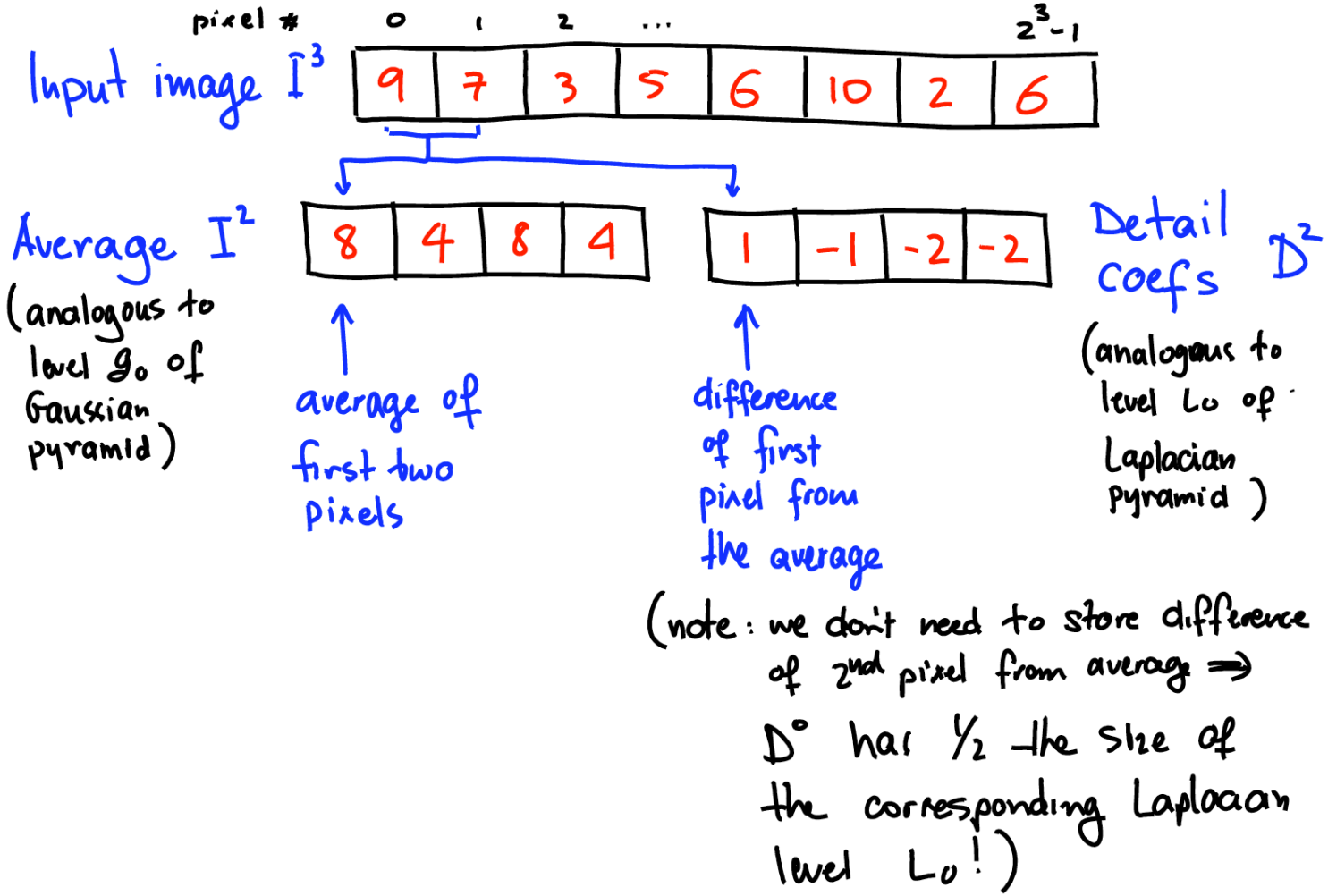
1-2 = -1 **5-6= -1**

Haar Wavelet Transform

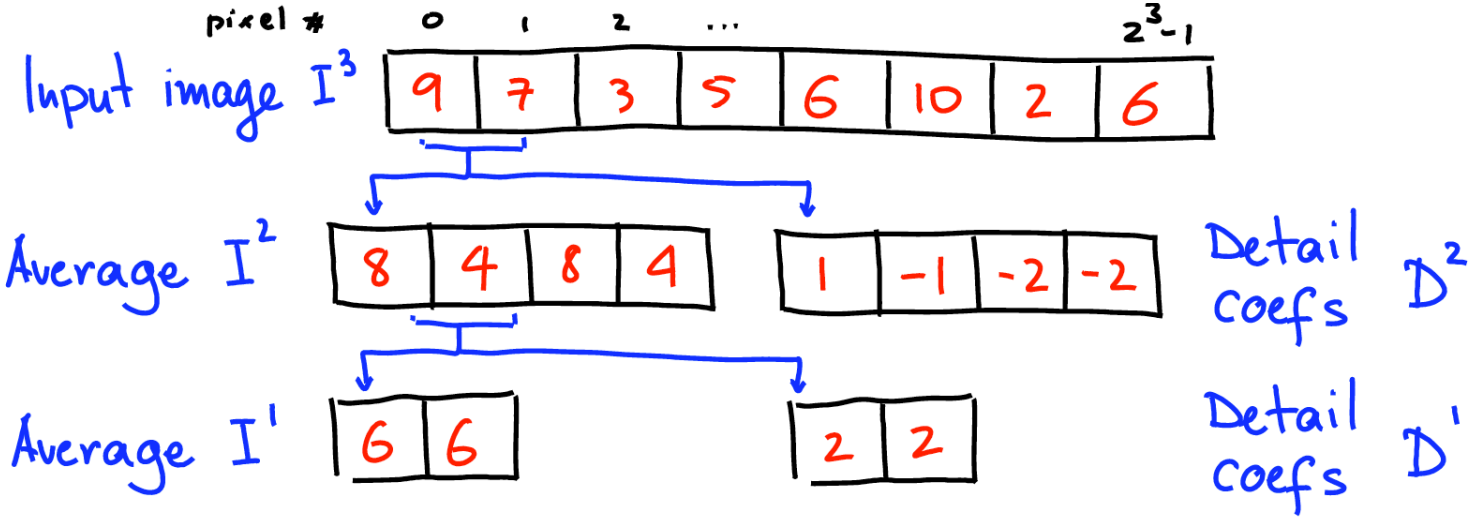
1. Find the average of each pair of samples. Fill the first half of the array with averages.
2. Find the difference between the average and the samples. Fill the second half of the array with differences.
3. Repeat the process on the first half of the array.

Original Signal	1	3	5	7
1. Iteration	**2**	**6**	**-1**	**-1**
2. Iteration	**4**	**-2**	**-1**	**-1**

A more complicated example



A more complicated example



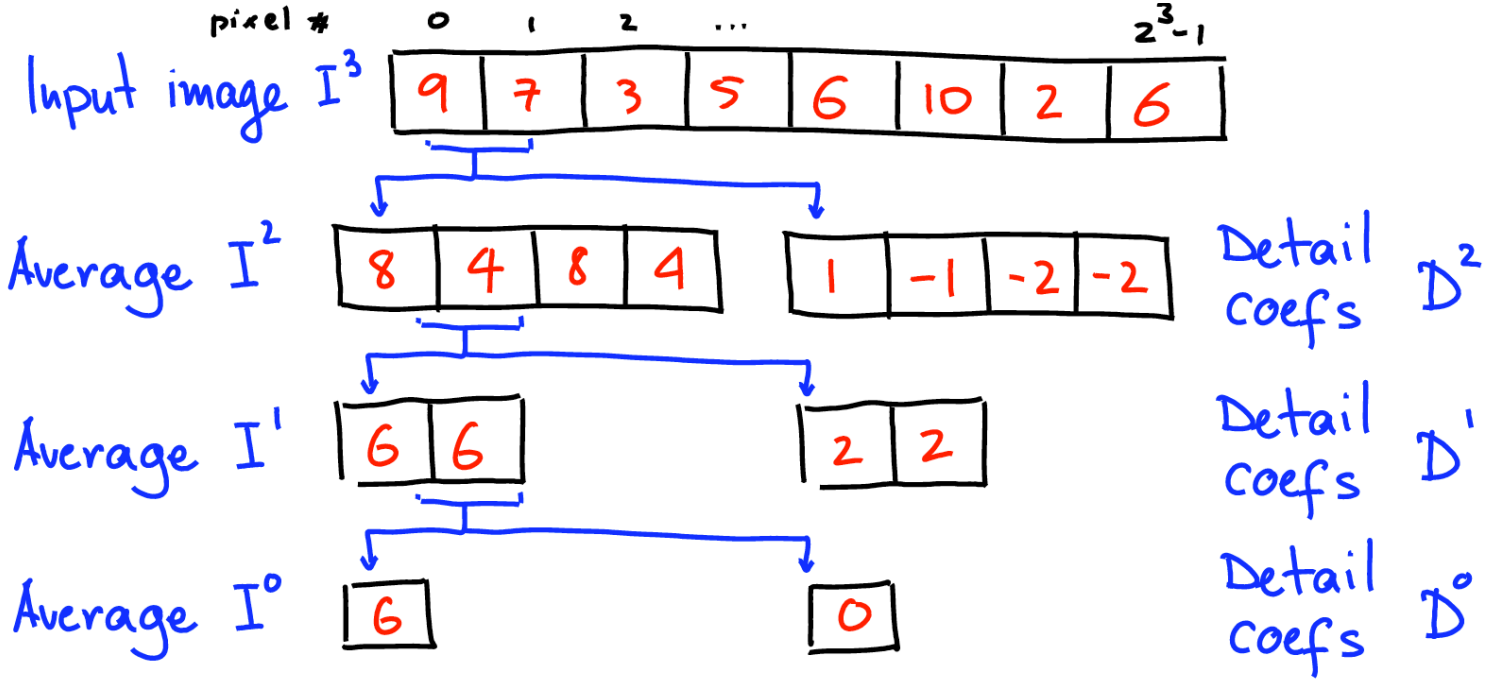
$$I_i^j = \frac{1}{2}\left(I_{2i}^{j+1} + I_{2i+1}^{j+1} \right)$$

j-th level of "pyramid" contains 2^j pixels

$$D_i^j = I_{2i}^{j+1} - \frac{1}{2}\left(I_{2i}^{j+1} + I_{2i+1}^{j+1} \right)$$

$$= \frac{1}{2}\left(I_{2i}^{j+1} - I_{2i+1}^{j+1} \right)$$

A more complicated example

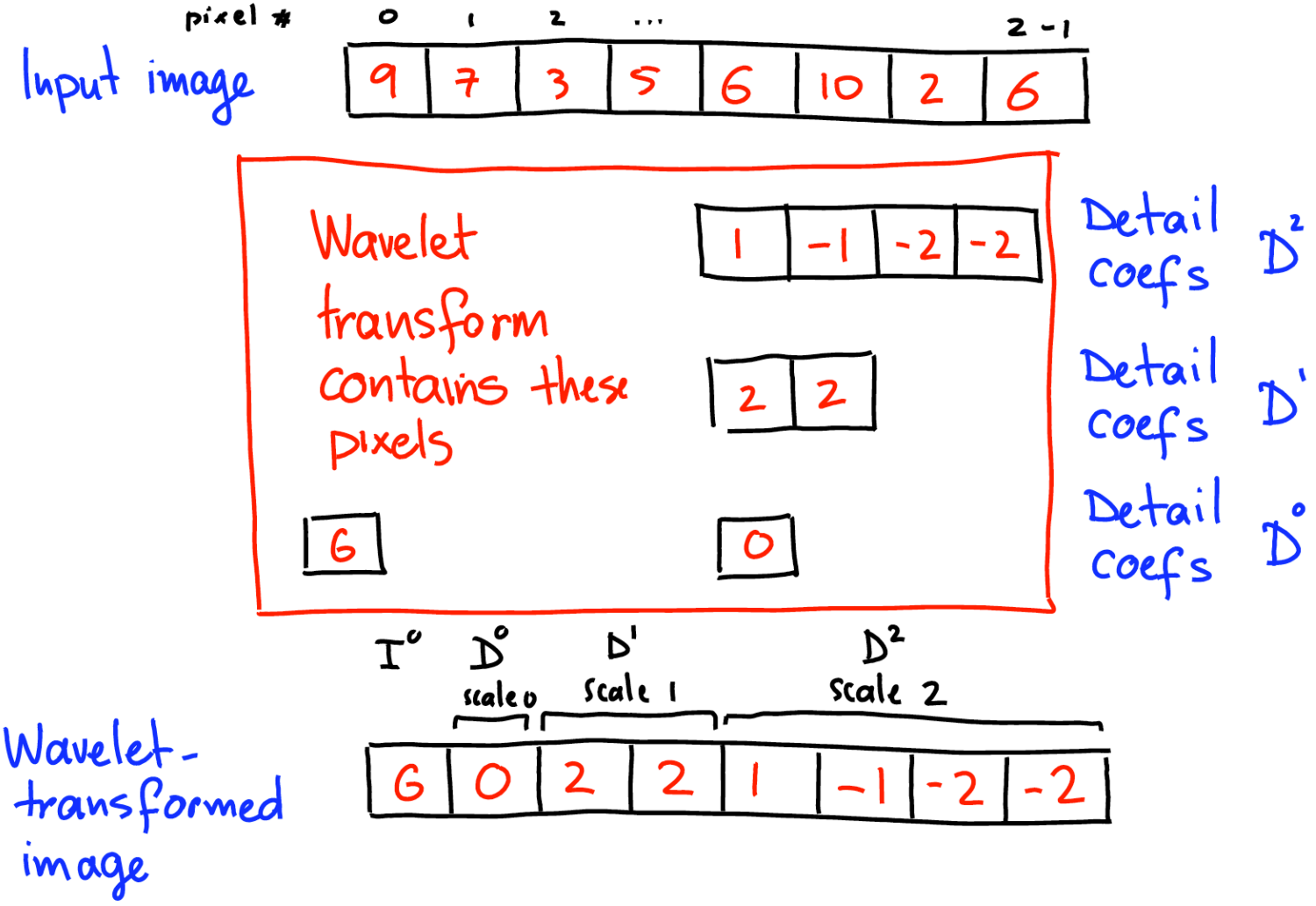


$$I_i^j = \frac{1}{2}\left(I_{2i}^{j+1} + I_{2i+1}^{j+1} \right)$$

$$D^j = \frac{1}{2}\left(I_{2i}^{j+1} - I_{2i+1}^{j+1} \right)$$

j-th level of "pyramid" contains 2^j pixels

A more complicated example



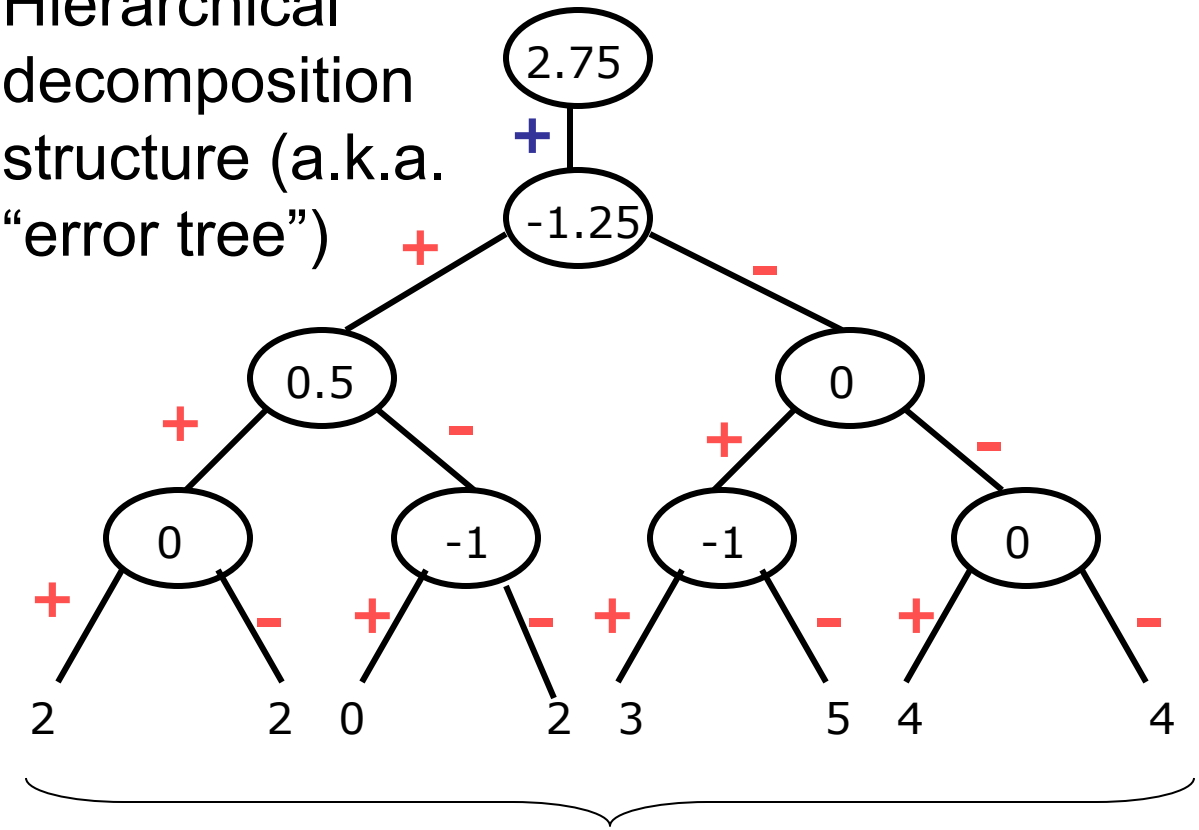
Wavelet Decomposition

- Wavelets: A math tool for space-efficient hierarchical decomposition of functions
- S = [2, 2, 0, 2, 3, 5, 4, 4] can be transformed to $S_{\wedge}$ = [$2^3/_4$, $-1^1/_4$, $^1/_2$, 0, 0, -1, -1, 0]
- Compression: many small detail coefficients can be replaced by 0's, and only the significant coefficients are retained

Resolution	Averages	Detail Coefficients
8	$[2, 2, 0, 2, 3, 5, 4, 4]$	
4	$[2, 1, 4, 4]$	$[0, -1, -1, 0]$
2	$[1\frac{1}{2}, 4]$	$[\frac{1}{2}, 0]$
1	$[2\frac{3}{4}]$	$[-1\frac{1}{4}]$

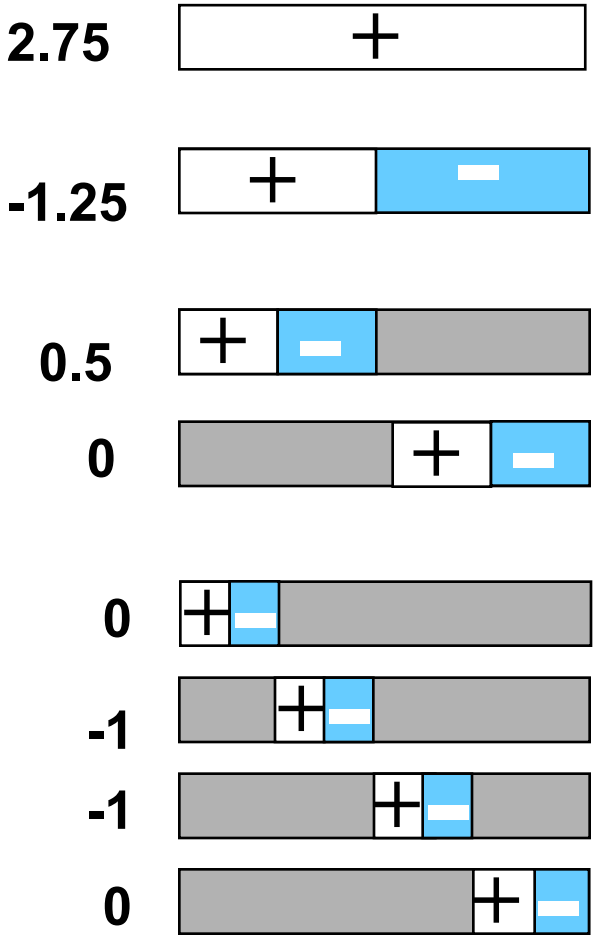
Haar Wavelet Coefficients

Hierarchical decomposition structure (a.k.a. "error tree")



Original frequency distribution

Coefficient "Supports"

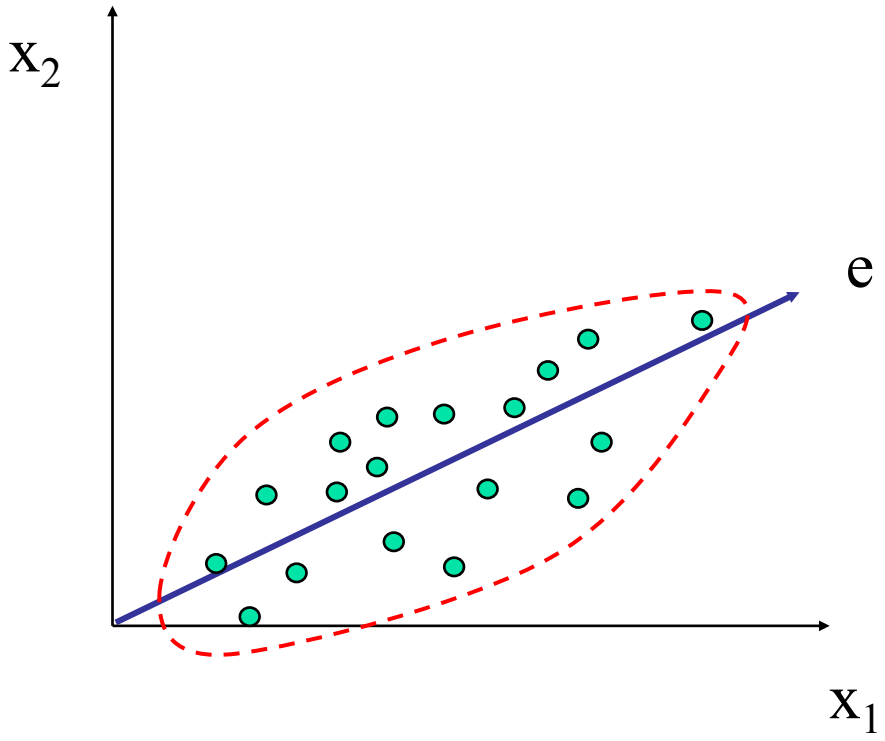


Why Wavelet Transform?

- Use hat-shape filters
  - Emphasize region where points cluster
  - Suppress weaker information in their boundaries
- Effective removal of outliers
  - Insensitive to noise, insensitive to input order
- Multi-resolution
  - Detect arbitrary shaped clusters at different scales
- Efficient
  - Complexity O(N)
- Only applicable to low dimensional data

Principal Component Analysis (PCA)

- Find a projection that captures the largest amount of variation in data
- The original data are projected onto a much smaller space, resulting in dimensionality reduction. We find the eigenvectors of the covariance matrix, and these eigenvectors define the new space



What is PCA?

- Principal components analysis (PCA): Dimensionality Reduction and Feature Construction
  - PCA is used to reduce dimensions of data without much loss of information.
  - Used in machine learning and in signal processing and image compression (among other things).

Goal of PCA

- We wish to explain/summarize the underlying variance-covariance structure of a large set of variables through a few linear combinations of these variables.

Background for PCA

- Suppose attributes are A_1 and A_2, and we have *n* training examples. *x*'s denote values of A_1 and *y*'s denote values of A_2 over the training examples.
- **Variance of an attribute A_1:**

$$\text{var}(A_1) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{(n-1)}$$

Background for PCA

- **Covariance of two attributes A_1 and A_2 :**

$$\mathrm{cov}(A_1, A_2) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{(n-1)}$$

  - If covariance is positive, both dimensions increase together.
  - If negative, as one increases, the other decreases.
  - Zero: independent of each other.

Background for PCA

- Covariance matrix
  - Suppose we have *n* attributes, A_1, ..., A_n.
  - **Covariance matrix:**

$$C^{n\times n} = (c_{i,j}), \text{where } c_{i,j} = \text{cov}(A_i, A_j)$$

Example

	Hours(H)	*Mark(M)*
Data	9	39
	15	56
	25	93
	14	61
	10	50
	18	75
	0	32
	16	85
	5	42
	19	70
	16	66
	20	80
Totals	167	749
Averages	13.92	62.42

Covariance:

H	M	$(H_i - \bar{H})$	$(M_i - \bar{M})$	$(H_i - \bar{H})(M_i - \bar{M})$
9	39	-4.92	-23.42	115.23
15	56	1.08	-6.42	-6.93
25	93	11.08	30.58	338.83
14	61	0.08	-1.42	-0.11
10	50	-3.92	-12.42	48.69
18	75	4.08	12.58	51.33
0	32	-13.92	-30.42	423.45
16	85	2.08	22.58	46.97
5	42	-8.92	-20.42	182.15
19	70	5.08	7.58	38.51
16	66	2.08	3.58	7.45
20	80	6.08	17.58	106.89
Total				1149.89
Average				104.54

Table 2.2: 2-dimensional data set and covariance calculation

$$\begin{pmatrix} \text{cov}(H,H) & \text{cov}(H,M) \\ \text{cov}(M,H) & \text{cov}(M,M) \end{pmatrix}$$

$$= \begin{pmatrix} \text{var}(H) & 104.5 \\ 104.5 & \text{var}(M) \end{pmatrix}$$

$$= \begin{pmatrix} 47.7 & 104.5 \\ 104.5 & 370 \end{pmatrix}$$

Covariance matrix

Background for PCA

- **Eigenvectors:**
  - Let **M be** an $n \times n$ matrix.
    - **v** is an *eigenvector* of M if M $\times$ **v** = λ**v**
    - λ is called the *eigenvalue* associated with **v**
  - For any eigenvector **v** of **M** and scalar *a*,

$$\mathbf{M} \times a\mathbf{v} = \lambda a\mathbf{v}$$

  - Thus you can always choose eigenvectors of length 1:

$$\sqrt{v_1^2 + \ldots + v_n^2} = 1$$

  - If **M** has any eigenvectors, it has *n* of them, and they are orthogonal to one another.
  - Thus eigenvectors can be used as a new basis for a n-dimensional vector space.

A 2D Numerical Example

PCA Example –STEP 1

- Subtract the mean
  - From each of the data dimensions, all the x values have $\bar{x}$ subtracted and y values have $\bar{y}$ subtracted from them. This produces a data set whose mean is zero.
  - Subtracting the mean makes variance and covariance calculation easier by simplifying their equations. The variance and co-variance values are not affected by the mean value.

PCA Example –STEP 1

- Given original data set S = {$\mathbf{x}^1$, ..., $\mathbf{x}^k$}, produce new set by subtracting the mean of attribute A_i from each x_i.

Data =

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9
Mean: 1.81	**1.91**

DataAdjust =

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01
Mean: **0**	**0**

PCA Example –STEP 2

- Calculate the covariance matrix

$$\text{cov} = \begin{array}{c} \\ x \\ y \end{array} \begin{array}{c} \begin{array}{cc} x & y \end{array} \\ \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix} \end{array}$$

- since the non-diagonal elements in this covariance matrix are positive, we should expect that both the x and y variable increase together.

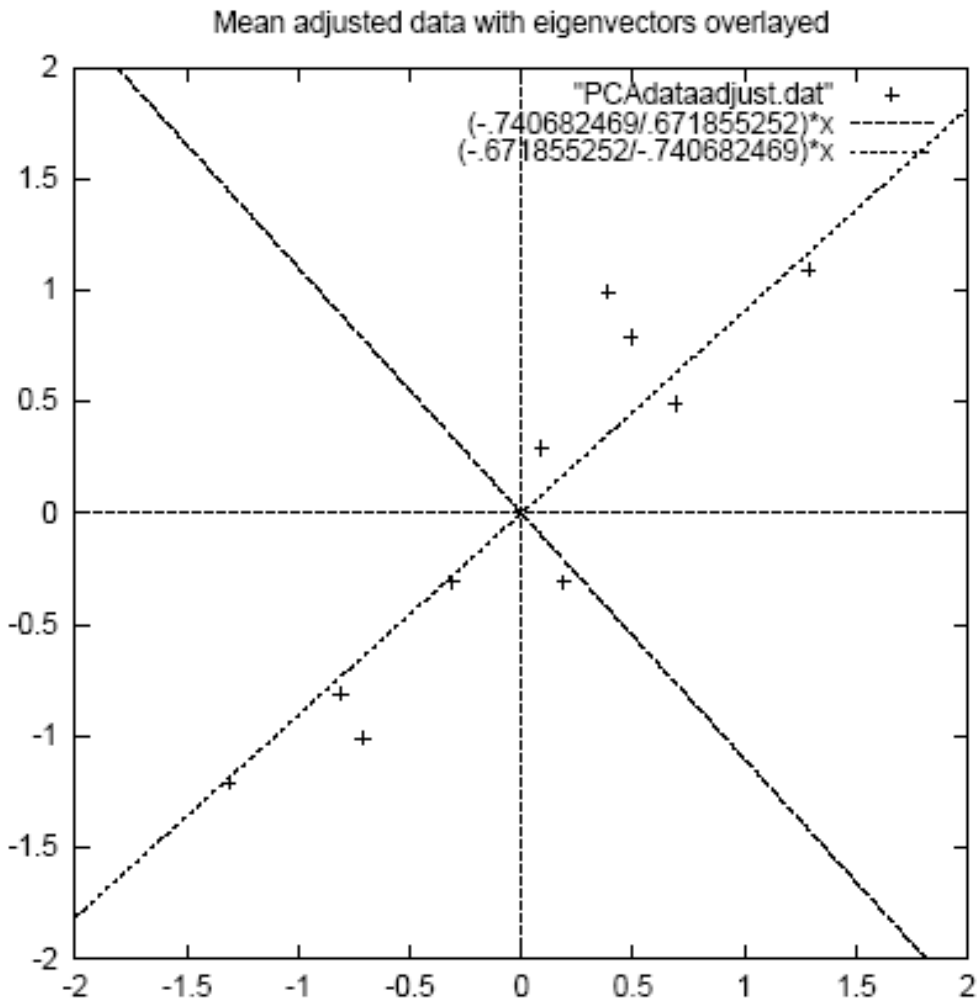
PCA Example –STEP 3

- Calculate the (unit)eigenvectors and eigenvalues of the covariance matrix

$$\text{eigenvalues} = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$\text{eigenvectors} = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

PCA Example –STEP 3



- **eigenvectors are plotted as diagonal dotted lines on the plot.**
- **Note they are perpendicular to each other.**
- **Note one of the eigenvectors goes through the middle of the points, like drawing a line of best fit.**
- **The second eigenvector gives us the other, less important, pattern in the data, that all the points follow the main line, but are off to the side of the main line by some amount.**

Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

PCA Example –STEP 4

- Reduce dimensionality and form *feature vector*

  the eigenvector with the *highest* eigenvalue is the *principle component* of the data set.

  In our example, the eigenvector with the larges eigenvalue was the one that pointed down the middle of the data.

  Once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. This gives you the components in order of significance.

PCA Example –STEP 4

Now, if you like, you can decide to *ignore* the components of lesser significance.

You do lose some information, but if the eigenvalues are small, you don't lose much

- n dimensions in your data
- calculate n eigenvectors and eigenvalues
- choose only the first p eigenvectors
- final data set has only p dimensions.

PCA Example –STEP 4

- Order eigenvectors by eigenvalue, highest to lowest.
- In general, you get *n* components.To reduce dimensionality to *p*, ignore *n–p* components at the bottom of the list.

$$\mathbf{v}_1 = \begin{pmatrix} -.677873399 \\ -.735178956 \end{pmatrix} \quad \lambda = 1.28402771$$

$$\mathbf{v}_2 = \begin{pmatrix} -.735178956 \\ .677873399 \end{pmatrix} \quad \lambda = .0490833989$$

PCA Example –STEP 4

- Feature Vector

  $$\text{FeatureVector} = (\text{eig}_1\ \text{eig}_2\ \text{eig}_3\ \ldots\ \text{eig}_n)$$

  We can either form a feature vector with both of the eigenvectors:

  $$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

  or, we can choose to leave out the smaller, less significant component and only have a single column:

  $$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

PCA Example –STEP 5

- Deriving the new data

  FinalData = RowFeatureVector x RowZeroMeanData

  RowFeatureVector is the matrix with the eigenvectors in the columns *transposed* so that the eigenvectors are now in the rows, with the most significant eigenvector at the top

  RowZeroMeanData is the mean-adjusted data *transposed*, ie. the data items are in each column, with each row holding a separate dimension.

PCA Example –STEP 5

- Deriving the new data

FinalData = RowFeatureVector x RowZeroMeanData (RowDataAdjust)

$$RowFeatureVector1 = \begin{pmatrix} -.677873399 & -.735178956 \\ -.735178956 & .677873399 \end{pmatrix}$$

$$RowFeatureVector2 = \begin{pmatrix} -.677873399 & -.735178956 \end{pmatrix}$$

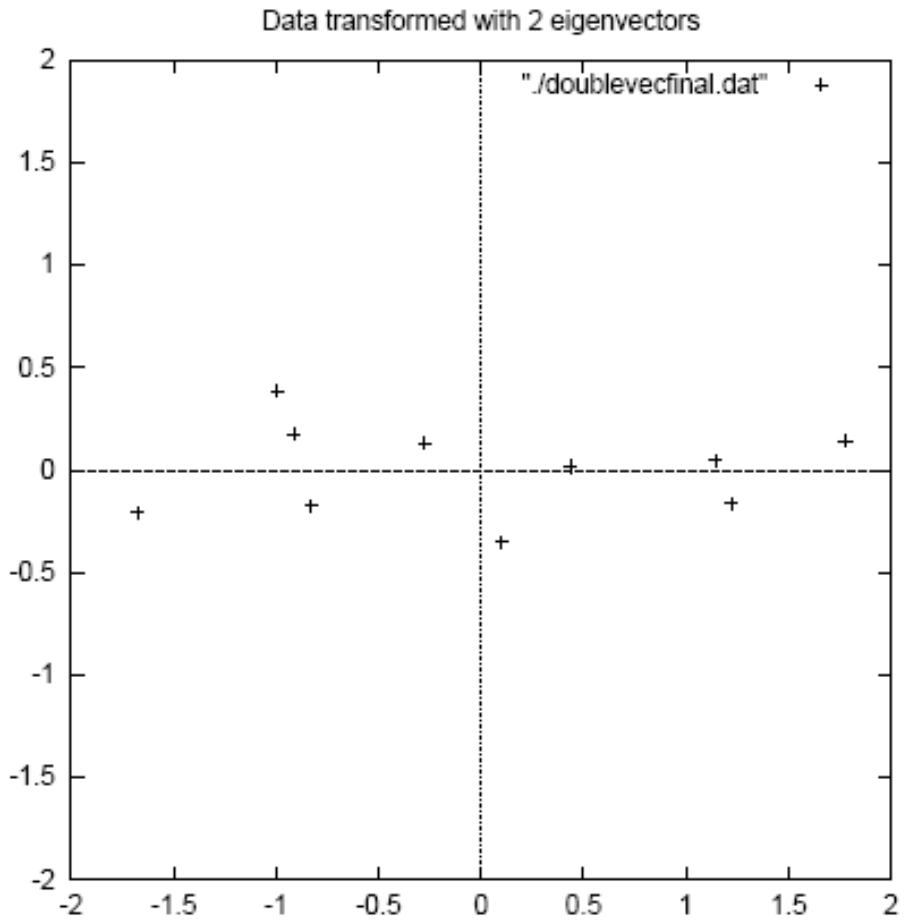
$$RowDataAdjust = \begin{pmatrix} .69 & -1.31 & .39 & .09 & 1.29 & .49 & .19 & -.81 & -.31 & -.71 \\ .49 & -1.21 & .99 & .29 & 1.09 & .79 & -.31 & -.81 & -.31 & -1.01 \end{pmatrix}$$

PCA Example –STEP 5

FinalData transpose:
dimensions along columns

x	y
-.827970186	-.175115307
1.77758033	.142857227
-.992197494	.384374989
-.274210416	.130417207
-1.67580142	-.209498461
-.912949103	.175282444
.0991094375	-.349824698
1.14457216	.0464172582
.438046137	.0177646297
1.22382056	-.162675287

PCA Example –STEP 5



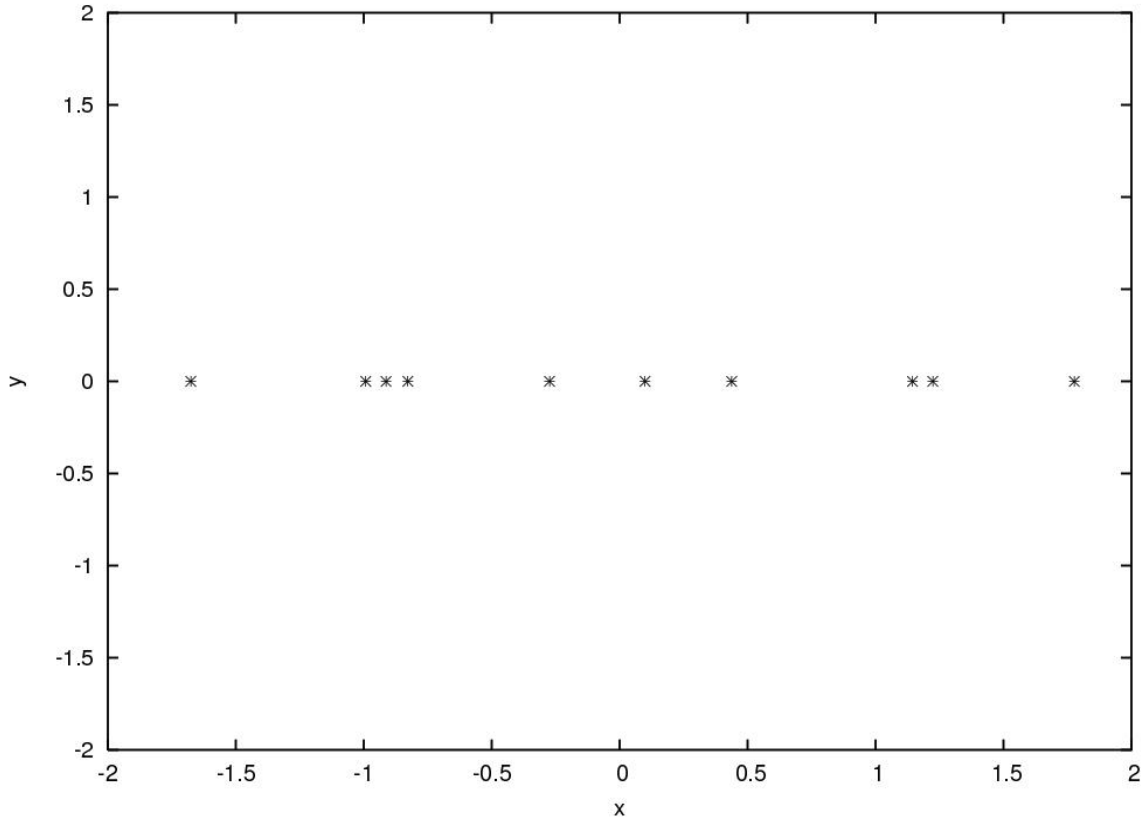
Transformed Data=

x	y
-.827970186	-.175115307
1.77758033	.142857227
-.992197494	.384374989
-.274210416	.130417207
-1.67580142	-.209498461
-.912949103	.175282444
.0991094375	-.349824698
1.14457216	.0464172582
.438046137	.0177646297
1.22382056	-.162675287

Data transformed with 2 eigenvectors

Figure 3.3: The table of data by applying the PCA analysis using both eigenvectors, and a plot of the new data points.

PCA Example –STEP 5



Transformed Data (Single eigenvector)

x
-.827970186
1.77758033
-.992197494
-.274210416
-1.67580142
-.912949103
.0991094375
1.14457216
.438046137
1.22382056

Reconstruction of original Data

- If we reduced the dimensionality, obviously, when reconstructing the data we would lose those dimensions we chose to discard. In our example let us assume that we considered only the x dimension…
- We did:

$$TransformedData = RowFeatureVector \times RowDataAdjust$$

so we can do

$$RowDataAdjust = RowFeatureVector^{-1} \times TransformedData$$

$$= RowFeatureVector^{T} \times TransformedData$$

and

$$RowDataOriginal = RowDataAdjust + OriginalMean$$

Reconstruction of original Data

x
-.827970186
1.77758033
-.992197494
-.274210416
-1.67580142
-.912949103
.0991094375
1.14457216
.438046137
1.22382056

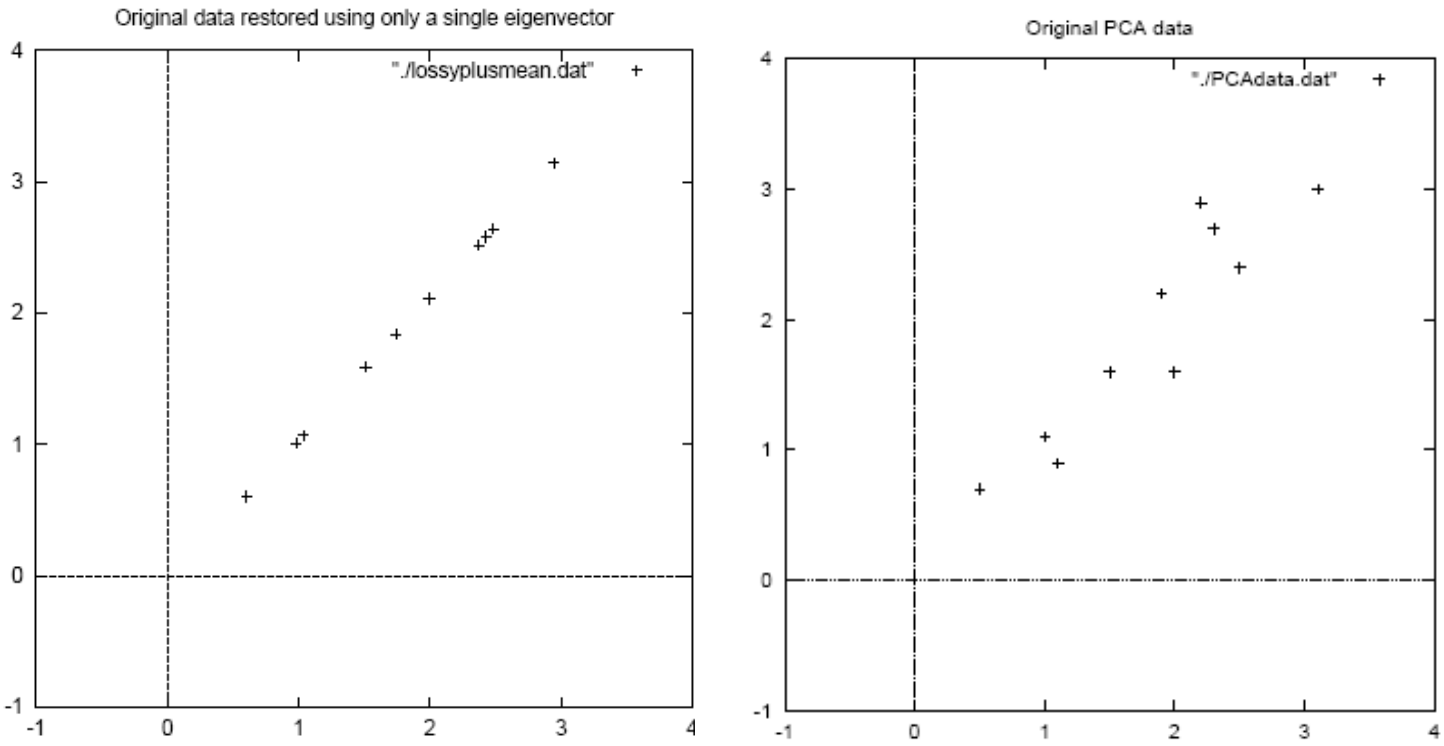


Figure 3.5: The reconstruction from the data that was derived using only a single eigenvector

Principal Component Analysis (Steps)

- Given *N* data vectors from *n*-dimensions, find $k \leq n$ orthogonal vectors (*principal components*) that can be best used to represent data
  - Normalize input data: Each attribute falls within the same range
  - Compute *k* orthonormal (unit) vectors, i.e., *principal components*
  - Each input data (vector) is a linear combination of the *k* principal component vectors
  - The principal components are sorted in order of decreasing "significance" or strength
  - Since the components are sorted, the size of the data can be reduced by eliminating the *weak components*, i.e., those with low variance (i.e., using the strongest principal components, it is possible to reconstruct a good approximation of the original data)
- Works for numeric data only

Attribute Subset Selection

- Another way to reduce dimensionality of data
- Redundant attributes
  - Duplicate much or all of the information contained in one or more other attributes
  - E.g., purchase price of a product and the amount of sales tax paid
- Irrelevant attributes
  - Contain no information that is useful for the data mining task at hand
  - E.g., students' ID is often irrelevant to the task of predicting students' GPA

Heuristic Search in Attribute Selection

- There are 2^d possible attribute combinations of d attributes
- Typical heuristic attribute selection methods:
  - Best single attribute under the attribute independence assumption: choose by significance tests
  - Best step-wise feature selection:
    - The best single-attribute is picked first
    - Then next best attribute condition to the first, ...
  - Step-wise attribute elimination:
    - Repeatedly eliminate the worst attribute
  - Best combined attribute selection and elimination
  - Optimal branch and bound:
    - Use attribute elimination and backtracking

Attribute Creation (Feature Generation)

- Create new attributes (features) that can capture the important information in a data set more effectively than the original ones
- Three general methodologies
  - Attribute extraction
    - Domain-specific
  - Mapping data to new space (see: data reduction)
    - E.g., Fourier transformation, wavelet transformation, manifold approaches (not covered)
  - Attribute construction
    - Combining features (see: discriminative frequent patterns in Chapter 7)
    - Data discretization

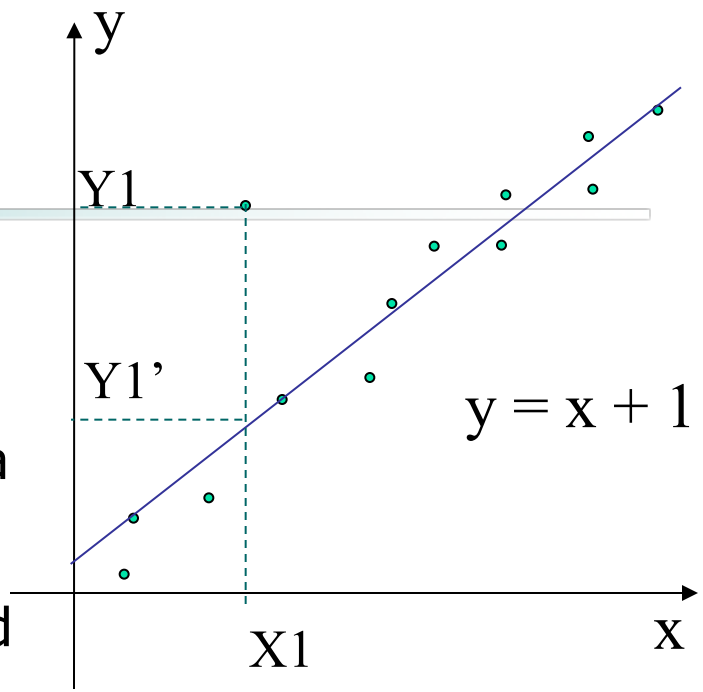
Data Reduction 2: Numerosity Reduction

- Reduce data volume by choosing alternative, *smaller forms* of data representation
- **Parametric methods** (e.g., regression)
  - Assume the data fits some model, estimate model parameters, store only the parameters, and discard the data (except possible outliers)
  - Ex.: Log-linear models—obtain value at a point in *m*-D space as the product on appropriate marginal subspaces
- **Non-parametric** methods
  - Do not assume models
  - Major families: histograms, clustering, sampling, …

Parametric Data Reduction: Regression and Log-Linear Models

- **Linear regression**
  - Data modeled to fit a straight line
  - Often uses the least-square method to fit the line
- **Multiple regression**
  - Allows a response variable Y to be modeled as a linear function of multidimensional feature vector
- **Log-linear model**
  - Approximates discrete multidimensional probability distributions

Regression Analysis



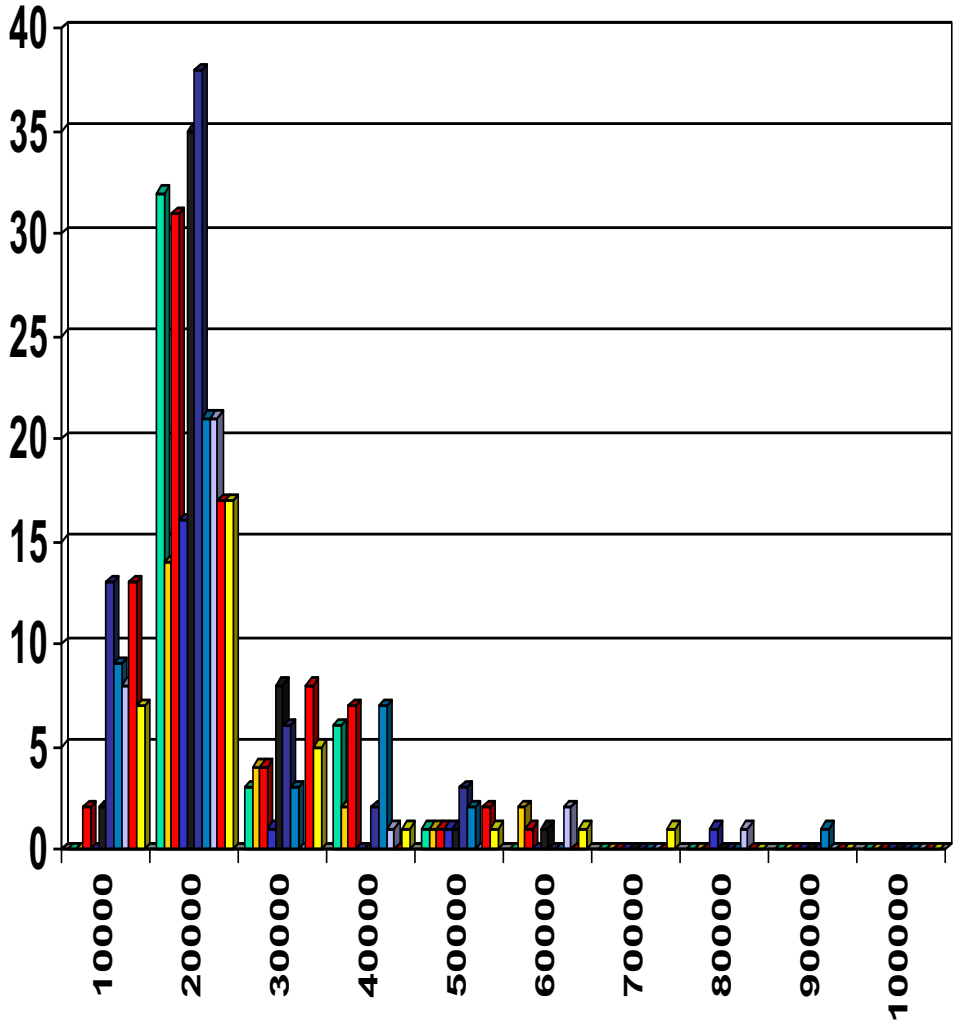
- Regression analysis: A collective name for techniques for the modeling and analysis of numerical data consisting of values of a ***dependent variable*** (also called ***response variable*** or *measurement*) and of one or more *independent variables* (aka. ***explanatory variables*** or ***predictors***)
- The parameters are estimated so as to give a "**best fit**" of the data
- Most commonly the best fit is evaluated by using the ***least squares method***, but other criteria have also been used
- Used for prediction (including forecasting of time-series data), inference, hypothesis testing, and modeling of causal relationships

Regress Analysis and Log-Linear Models

- Linear regression: $Y = w X + b$
  - Two regression coefficients, w and b, specify the line and are to be estimated by using the data at hand
  - Using the least squares criterion to the known values of $Y_1, Y_2, \ldots, X_1, X_2, \ldots$.
- Multiple regression: $Y = b_0 + b_1 X_1 + b_2 X_2$
  - Many nonlinear functions can be transformed into the above
- Log-linear models:
  - Approximate discrete multidimensional probability distributions
  - Estimate the probability of each point (tuple) in a multi-dimensional space for a set of discretized attributes, based on a smaller subset of dimensional combinations
  - Useful for dimensionality reduction and data smoothing

Histogram Analysis

- Divide data into buckets and store average (sum) for each bucket
- Partitioning rules:
  - Equal-width: equal bucket range
  - Equal-frequency (or equal-depth)



Clustering

- Partition data set into clusters based on similarity, and store cluster representation (e.g., centroid and diameter) only
- Can be very effective if data is clustered but not if data is "smeared"
- Can have hierarchical clustering and be stored in multi-dimensional index tree structures
- There are many choices of clustering definitions and clustering algorithms
- Cluster analysis will be studied in depth in Chapter 10

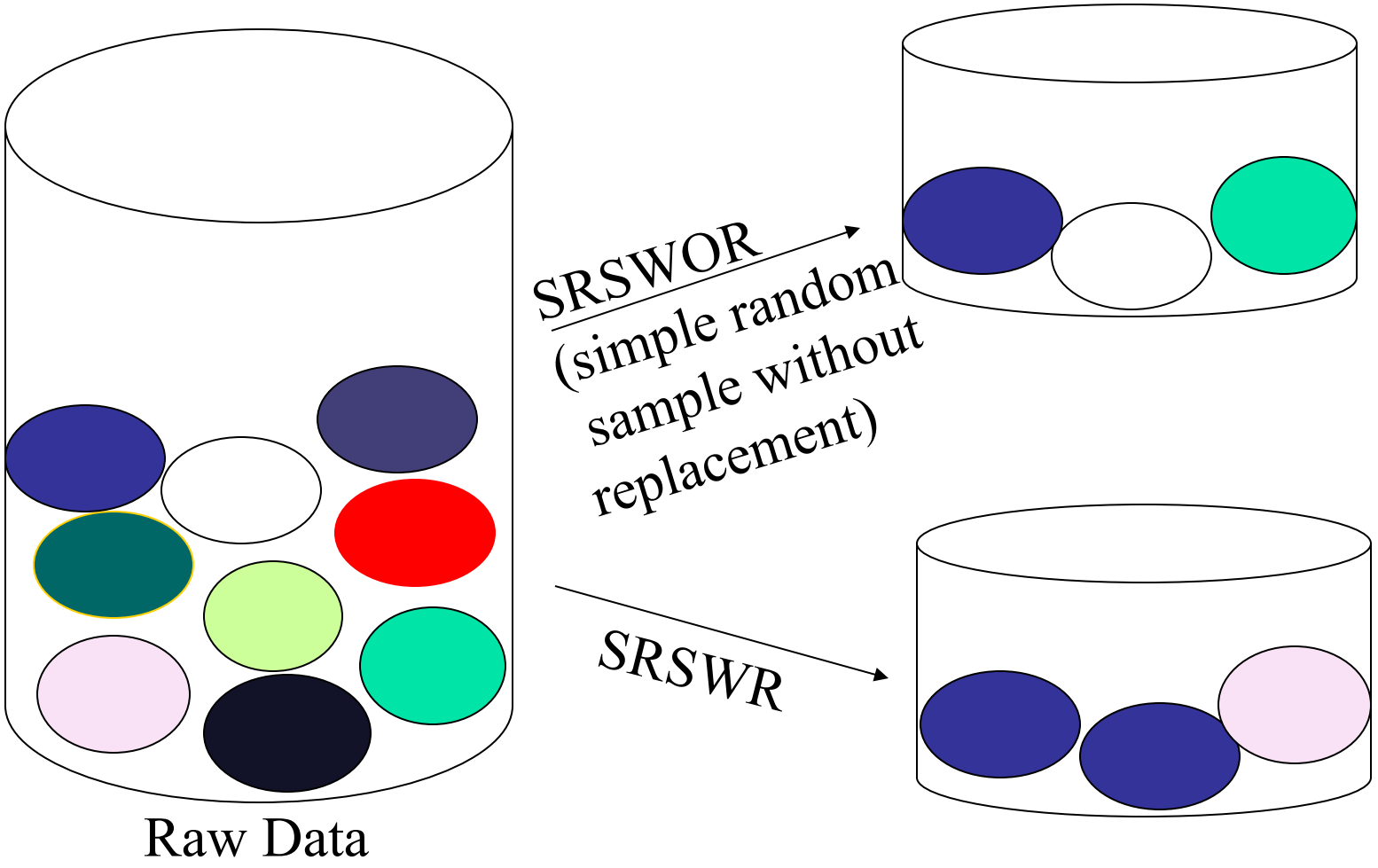
Sampling

- Sampling: obtaining a small sample *s* to represent the whole data set *N*
- Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data
- Key principle: Choose a representative subset of the data
  - Simple random sampling may have very poor performance in the presence of skew
  - Develop adaptive sampling methods, e.g., stratified sampling:
- Note: Sampling may not reduce database I/Os (page at a time)

Types of Sampling

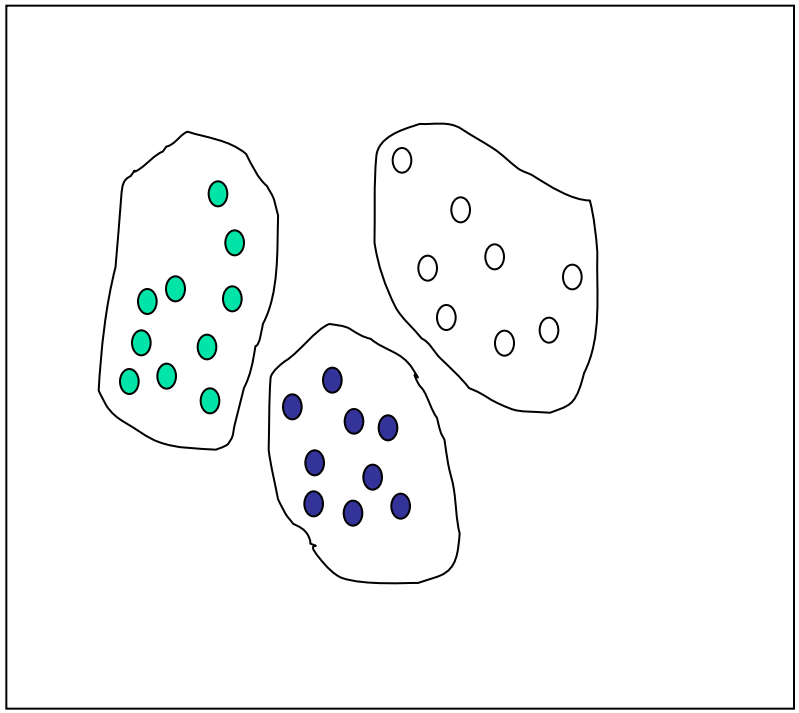
- **Simple random sampling**
  - There is an equal probability of selecting any particular item
- **Sampling without replacement**
  - Once an object is selected, it is removed from the population
- **Sampling with replacement**
  - A selected object is not removed from the population
- **Stratified sampling:**
  - Partition the data set, and draw samples from each partition (proportionally, i.e., approximately the same percentage of the data)
  - Used in conjunction with skewed data

Sampling: With or without Replacement

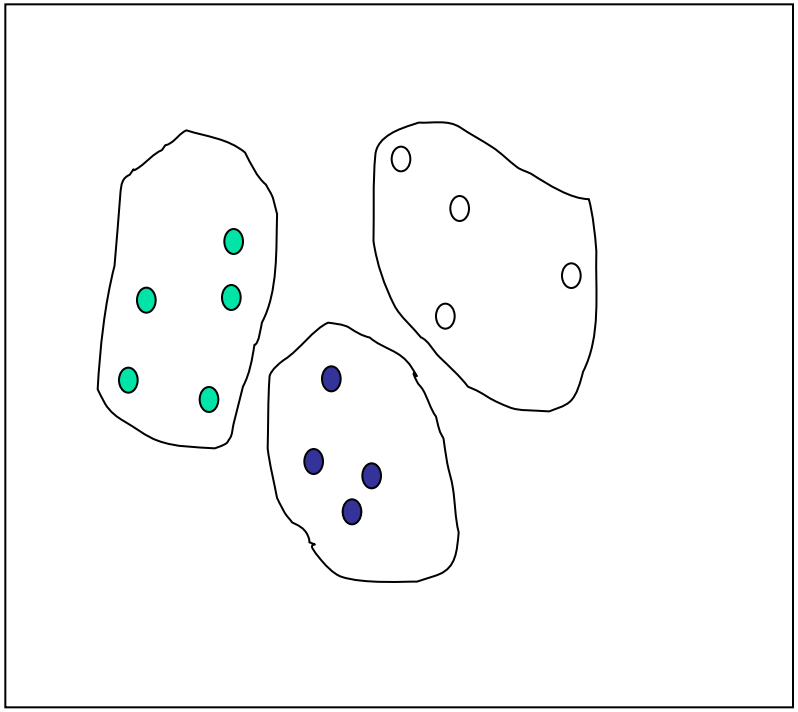


Sampling: Cluster or Stratified Sampling

Raw Data



Cluster/Stratified Sample



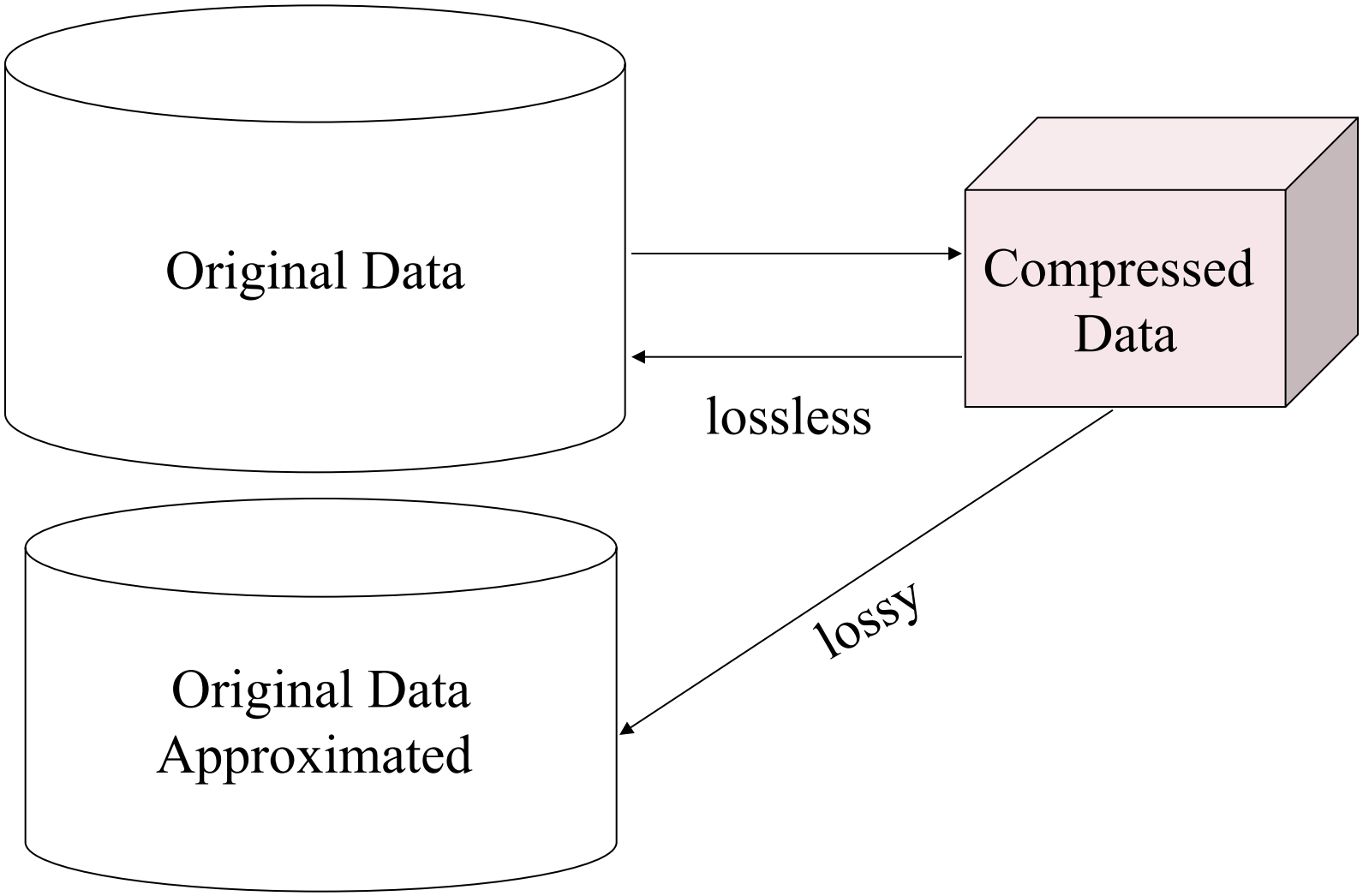
Data Cube Aggregation

- The lowest level of a data cube (base cuboid)
  - The aggregated data for an individual entity of interest
  - E.g., a customer in a phone calling data warehouse
- Multiple levels of aggregation in data cubes
  - Further reduce the size of data to deal with
- Reference appropriate levels
  - Use the smallest representation which is enough to solve the task
- Queries regarding aggregated information should be answered using data cube, when possible

Data Reduction 3: Data Compression

- String compression
  - There are extensive theories and well-tuned algorithms
  - Typically lossless, but only limited manipulation is possible without expansion
- Audio/video compression
  - Typically lossy compression, with progressive refinement
  - Sometimes small fragments of signal can be reconstructed without reconstructing the whole
- Time sequence is not audio
  - Typically short and vary slowly with time
- Dimensionality and numerosity reduction may also be considered as forms of data compression

Data Compression



Chapter 3: Data Preprocessing

- Data Preprocessing: An Overview
  - Data Quality
  - Major Tasks in Data Preprocessing
- Data Cleaning
- Data Integration
- Data Reduction
- Data Transformation and Data Discretization 
- Summary

Data Transformation

- A function that maps the entire set of values of a given attribute to a new set of replacement values s.t. each old value can be identified with one of the new values
- Methods
  - Smoothing: Remove noise from data
  - Attribute/feature construction
    - New attributes constructed from the given ones
  - Aggregation: Summarization, data cube construction
  - Normalization: Scaled to fall within a smaller, specified range
    - min-max normalization
    - z-score normalization
    - normalization by decimal scaling
  - Discretization: Concept hierarchy climbing

Normalization

- **Min-max normalization**: to [new_min$_A$, new_max$_A$]

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

  - Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped to $\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716$

- **Z-score normalization** (μ: mean, σ: standard deviation):

$$v' = \frac{v - \mu_A}{\sigma_A}$$

  - Ex. Let μ = 54,000, σ = 16,000. Then $\frac{73{,}600 - 54{,}000}{16{,}000} = 1.225$

- **Normalization by decimal scaling**

$$v' = \frac{v}{10^j}$$ Where j is the smallest integer such that Max(|v'|) < 1

Discretization

- Three types of attributes
  - Nominal—values from an unordered set, e.g., color, profession
  - Ordinal—values from an ordered set, e.g., military or academic rank
  - Numeric—real numbers, e.g., integer or real numbers
- Discretization: Divide the range of a continuous attribute into intervals
  - Interval labels can then be used to replace actual data values
  - Reduce data size by discretization
  - Supervised vs. unsupervised
  - Split (top-down) vs. merge (bottom-up)
  - Discretization can be performed recursively on an attribute
  - Prepare for further analysis, e.g., classification

Data Discretization Methods

- Typical methods: All the methods can be applied recursively
  - Binning
    - Top-down split, unsupervised
  - Histogram analysis
    - Top-down split, unsupervised
  - Clustering analysis (unsupervised, top-down split or bottom-up merge)
  - Decision-tree analysis (supervised, top-down split)
  - Correlation (e.g., χ^2) analysis (unsupervised, bottom-up merge)

Simple Discretization: Binning

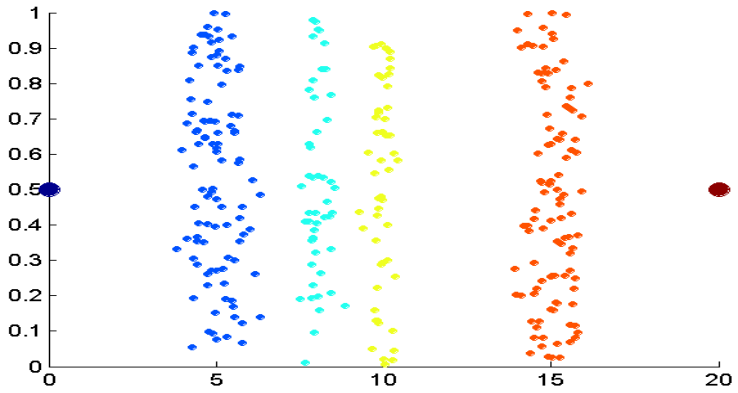
- Equal-width (distance) partitioning
  - Divides the range into *N* intervals of equal size: uniform grid
  - if *A* and *B* are the lowest and highest values of the attribute, the width of intervals will be: $W = (B - A)/N$.
  - The most straightforward, but outliers may dominate presentation
  - Skewed data is not handled well
- Equal-depth (frequency) partitioning
  - Divides the range into *N* intervals, each containing approximately same number of samples
  - Good data scaling
  - Managing categorical attributes can be tricky

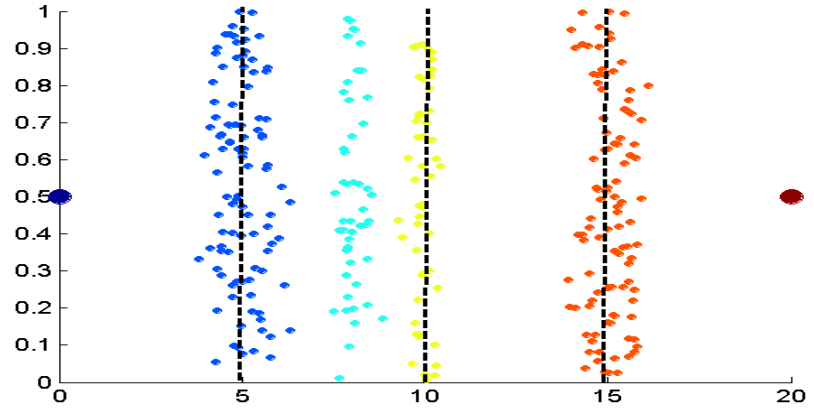
Binning Methods for Data Smoothing

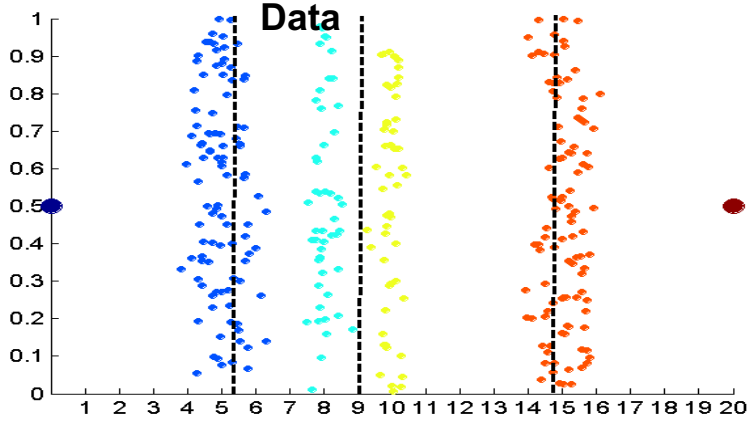
- Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

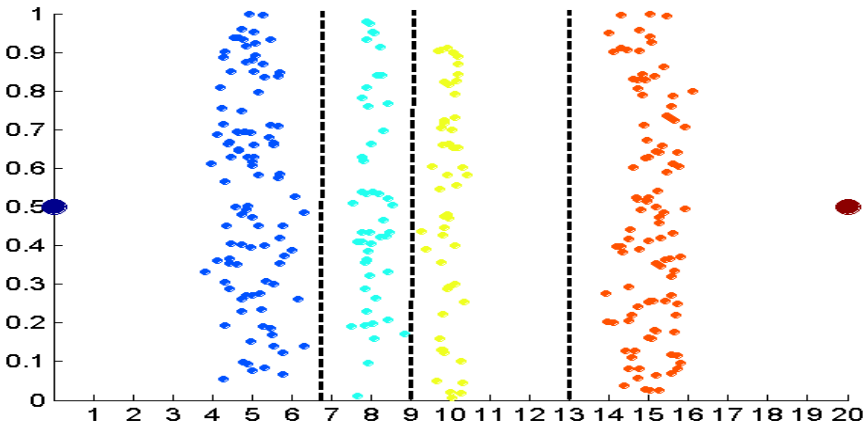
* Partition into equal-frequency (**equi-depth**) bins:
  - Bin 1: 4, 8, 9, 15
  - Bin 2: 21, 21, 24, 25
  - Bin 3: 26, 28, 29, 34
* Smoothing by **bin means**:
  - Bin 1: 9, 9, 9, 9
  - Bin 2: 23, 23, 23, 23
  - Bin 3: 29, 29, 29, 29
* Smoothing by **bin boundaries**:
  - Bin 1: 4, 4, 4, 15
  - Bin 2: 21, 21, 25, 25
  - Bin 3: 26, 26, 26, 34

Discretization Without Using Class Labels (Binning vs. Clustering)









Equal frequency (binning)

K-means clustering leads to better results

Discretization by Classification & Correlation Analysis

- Classification (e.g., decision tree analysis)
  - Supervised: Given class labels, e.g., cancerous vs. benign
  - Using *entropy* to determine split point (discretization point)
  - Top-down, recursive split
  - Details to be covered in Chapter 7
- Correlation analysis (e.g., Chi-merge: χ^2-based discretization)
  - Supervised: use class information
  - Bottom-up merge: find the best neighboring intervals (those having similar distributions of classes, i.e., low χ^2 values) to merge
  - Merge performed recursively, until a predefined stopping condition

Concept Hierarchy Generation

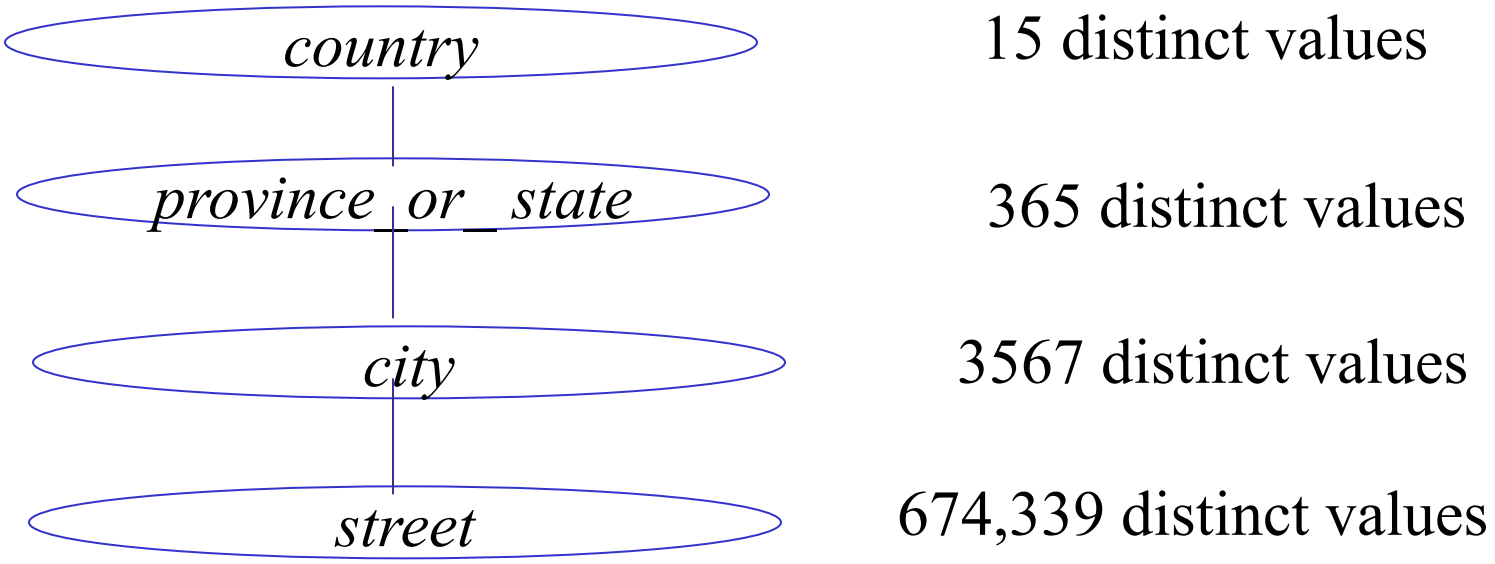
- **Concept hierarchy** organizes concepts (i.e., attribute values) hierarchically and is usually associated with each dimension in a data warehouse
- Concept hierarchies facilitate drilling and rolling in data warehouses to view data in multiple granularity
- Concept hierarchy formation: Recursively reduce the data by collecting and replacing low level concepts (such as numeric values for *age*) by higher level concepts (such as *youth, adult*, or *senior*)
- Concept hierarchies can be explicitly specified by domain experts and/or data warehouse designers
- Concept hierarchy can be automatically formed for both numeric and nominal data. For numeric data, use discretization methods shown.

Concept Hierarchy Generation for Nominal Data

- Specification of a partial/total ordering of attributes explicitly at the schema level by users or experts
  - *street* < *city* < *state* < *country*
- Specification of a hierarchy for a set of values by explicit data grouping
  - {Urbana, Champaign, Chicago} < Illinois
- Specification of only a partial set of attributes
  - E.g., only *street* < *city*, not others
- Automatic generation of hierarchies (or attribute levels) by the analysis of the number of distinct values
  - E.g., for a set of attributes: {*street, city, state, country*}

Automatic Concept Hierarchy Generation

- Some hierarchies can be automatically generated based on the analysis of the number of distinct values per attribute in the data set
  - The attribute with the most distinct values is placed at the lowest level of the hierarchy
  - Exceptions, e.g., weekday, month, quarter, year



Chapter 3: Data Preprocessing

- Data Preprocessing: An Overview
  - Data Quality
  - Major Tasks in Data Preprocessing
- Data Cleaning
- Data Integration
- Data Reduction
- Data Transformation and Data Discretization
- Summary

Summary

- **Data quality**: accuracy, completeness, consistency, timeliness, believability, interpretability
- **Data cleaning**: e.g. missing/noisy values, outliers
- **Data integration** from multiple sources:
  - Entity identification problem
  - Remove redundancies
  - Detect inconsistencies
- **Data reduction**
  - Dimensionality reduction
  - Numerosity reduction
  - Data compression
- **Data transformation and data discretization**
  - Normalization
  - Concept hierarchy generation

References

- D. P. Ballou and G. K. Tayi. Enhancing data quality in data warehouse environments. Comm. of ACM, 42:73-78, 1999
- A. Bruce, D. Donoho, and H.-Y. Gao. Wavelet analysis. *IEEE Spectrum*, Oct 1996
- T. Dasu and T. Johnson. Exploratory Data Mining and Data Cleaning. John Wiley, 2003
- J. Devore and R. Peck. *Statistics: The Exploration and Analysis of Data*. Duxbury Press, 1997.
- H. Galhardas, D. Florescu, D. Shasha, E. Simon, and C.-A. Saita. Declarative data cleaning: Language, model, and algorithms. *VLDB'01*
- M. Hua and J. Pei. Cleaning disguised missing data: A heuristic approach. *KDD'07*
- H. V. Jagadish, et al., Special Issue on Data Reduction Techniques. Bulletin of the Technical Committee on Data Engineering, 20(4), Dec. 1997
- H. Liu and H. Motoda (eds.). *Feature Extraction, Construction, and Selection: A Data Mining Perspective*. Kluwer Academic, 1998
- J. E. Olson. *Data Quality: The Accuracy Dimension*. Morgan Kaufmann, 2003
- D. Pyle. Data Preparation for Data Mining. Morgan Kaufmann, 1999
- V. Raman and J. Hellerstein. Potters Wheel: An Interactive Framework for Data Cleaning and Transformation, VLDB'2001
- T. Redman. *Data Quality: The Field Guide*. Digital Press (Elsevier), 2001
- R. Wang, V. Storey, and C. Firth. A framework for analysis of data quality research. IEEE Trans. Knowledge and Data Engineering, 7:623-640, 1995