# Chapter 2 Data and Data Processing

Types of Data

Data Quality

Data Preprocessing

- Dimensionality Reduction, Feature Subset Selection
- Discretization and Binarization
- Variable Transformation

Measures of Similarity and Dissimilarity

- Similarity and Dissimilarity between Simple Attributes
- Similarity and Dissimilarity between Data Objects

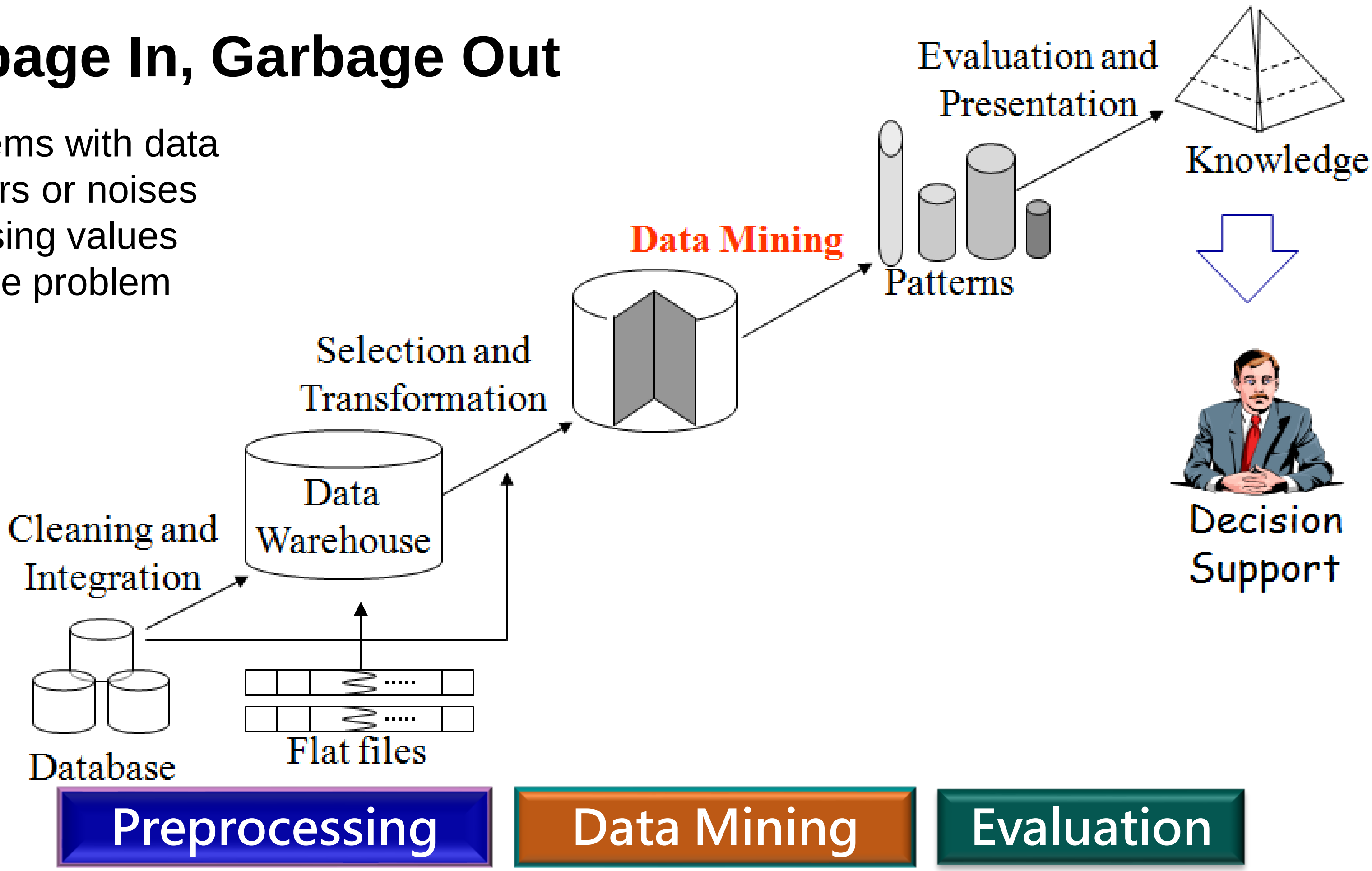
# Why Data Preprocessing?

## Garbage In, Garbage Out

Problems with data

- Errors or noises
- Missing values
- Scale problem

…




t1: Tom, M, 400, 60000, Master, Sports, Engineering, …

t2: Mary, F, 30, 50000, , Music, Marketing, …

# What is Data?

- Collection of data objects and their attributes
- An attribute is a property or characteristic of an object
  - Examples: eye color of a person, temperature, etc.
  - **Attribute** is also known as **variable**, **field**, **characteristic**, or **feature**
- A collection of attributes describe an object
  - **Object** is also known as **instance**, **observation**, **point**, **case**, **record**, **entity**, or **example**

**Attributes**

**Objects**

| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

**Feature Attributes**

**Class Attribute**

# Types of Attributes

- Nominal
  - ID numbers, eye color, zip codes, …
- Ordinal
  - rankings (e.g., taste of potato chips on a scale from 1~10);
  - grades in {A, B, C, D, F};
  - height in {tall, medium, short},
  - …
- Interval
  - calendar dates, temperatures in Celsius or Fahrenheit, …
- Ratio
  - temperature in Kelvin, length, time, counts, …

# Properties of Attribute Values

- The type of an attribute depends on which of the following properties it possesses:
  - Distinctness: $= \ \neq$
  - Order: $< \ >$
  - Addition: $+ \ -$
  - Multiplication: $* \ /$

- Properties for different attributes
  - **Nominal:** distinctness (eye color, sex); e.g., yellow $\neq$ red
  - **Ordinal:** distinctness & order (rankings, grade); e.g., A > B
  - **Interval:** distinctness, order & addition (calendar date); e.g., (June 3) + 1
  - **Ratio:** all 4 properties (length, height, weight); e.g., 100 cm = 2* 50cm

| **Attribute Type** | **Description** | **Examples** | **Operations** |
|---|---|---|---|
| Nominal | The values of a nominal attribute are just different names, i.e., nominal attributes provide only enough information **to distinguish one object from another**. ($=$, $\neq$) | zip codes, employee ID numbers, eye color, sex: {*male*, *female*} | mode, entropy, contingency correlation, $\chi^2$ test |
| Ordinal | The values of an ordinal attribute provide enough information **to order objects**. ($<$, $>$) | hardness of minerals, {*good*, *better*, *best*}, grades, street numbers | median, percentiles, rank correlation, run tests, sign tests |
| Interval | For interval attributes, the **differences** between values are meaningful, i.e., a unit of measurement exists. ($+$, $-$) | calendar dates, temperature in Celsius or Fahrenheit | mean, standard deviation, Pearson's correlation, $t$ and $F$ tests |
| Ratio | For ratio variables, both **differences** and **ratios** are meaningful. ($+$, $-$, $*$, $/$) | temperature in Kelvin, monetary quantities, counts, age, mass, length, electrical current | geometric mean, harmonic mean, percent variation |

# Discrete and Continuous Attributes

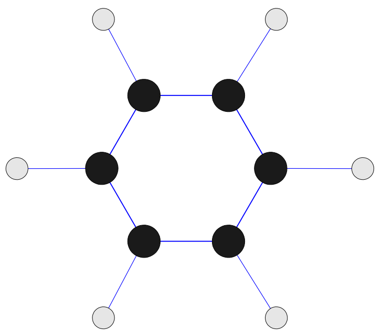
- Discrete Attribute
  - Has only a **finite** or **countably infinite** set of values
  - Often represented as integer variables.
  - Binary attributes are a special case of discrete attributes
  - Examples:
    - zip codes, counts, or the set of words in a collection of documents

- Continuous Attribute
  - Has **real numbers** as attribute values
  - Continuous attributes are typically represented as floating-point variables.
  - Practically, real values can only be measured and represented using a finite number of digits.
  - Examples:
    - temperature, height, or weight.

# Types of data sets

- **Record**
  - Data Matrix
  - Document Data
  - Transaction Data
- **Graph**
  - World Wide Web
  - Molecular Structures
- **Ordered**
  - Spatial Data
  - Temporal Data
  - Sequential Data
  - Genetic Sequence Data

| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | No |
| 2 | No | Married | 100K | No |
| 3 | No | Single | 70K | No |
| 4 | Yes | Married | 120K | No |
| 5 | No | Divorced | 95K | Yes |
| 6 | No | Married | 60K | No |
| 7 | Yes | Divorced | 220K | No |
| 8 | No | Single | 85K | Yes |
| 9 | No | Married | 75K | No |
| 10 | No | Single | 90K | Yes |



GGTTCCGCCTTCAGCCCCGCGCC
CGCAGGGCCCGCCCCGCGCCGTC
GAGAAGGGCCCGCCTGGCGGGCG
GGGGGAGGCGGGGCCGCCCGAGC
CCAACCGAGTCCGACCAGGTGCC
CCCTCTGCTCGGCCTAGACCTGA
GCTCATTAGGCGGCAGCGGACAG
GCCAAGTAGAACACGCGAAGCGC
TGGGCTGCCTGCTGCGACCAGGG

# Record Data

- Data that consists of a collection of records,
- Each record consists of a fixed set of attributes

Attributes

Records (objects)

| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 6 | No | Married | 60K | **No** |
| 7 | Yes | Divorced | 220K | **No** |
| 8 | No | Single | 85K | **Yes** |
| 9 | No | Married | 75K | **No** |
| 10 | No | Single | 90K | **Yes** |

# Data Matrix

- If data objects have the same fixed set of numeric attributes, then the data objects can be thought of as points in a multi-dimensional space, where each dimension represents a distinct attribute

- Such data set can be represented by an $m$ by $n$ matrix, where there are $m$ rows, one for each object, and $n$ columns, one for each attribute

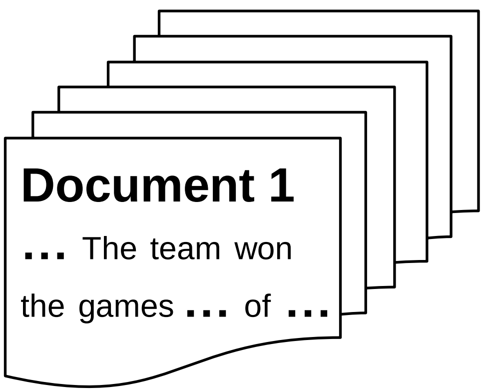
attributes ($n$)

objects ($m$)

| Projection of x Load | Projection of y load | Distance | Load | Thickness |
|---|---|---|---|---|
| 10.23 | 5.27 | 15.22 | 2.7 | 1.2 |
| 12.65 | 6.25 | 16.22 | 2.2 | 1.1 |

# Document Data

- Transfer each document to become a 'term' vector,
  - each term (keyword) is a component (attribute) of the vector,
  - the value of each component is the number of times the corresponding term occurs in the document.
  - referred to as Vector Space Model (VSM) or Document-Term Matrix




1. **Stemming (won→win, games→game, …)**
2. **Remove stop words (of, the, a, …)**
3. **Extract keywords (team, win, …)**
4. **Form the keyword vector [team, coach, …, season]**
5. **Convert documents to Document-Term matrix**

**Terms (keywords) of the document collection**

| documents | team | coach | play | ball | score | game | win | lost | timeout | season |
|---|---|---|---|---|---|---|---|---|---|---|
| Document 1 | 3 | 0 | 5 | 0 | 2 | 6 | 0 | 2 | 0 | 2 |
| Document 2 | 0 | 7 | 0 | 2 | 1 | 0 | 0 | 3 | 0 | 0 |
| Document 3 | 0 | 1 | 0 | 0 | 1 | 2 | 2 | 0 | 3 | 0 |

**Document similarity can be measured by:**

$$\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1||\ ||d_2|| = 0.1113$$

# Transaction Data

- A special type of record data, where
  - each record (transaction) involves a set of items.
  - E.g., consider a grocery store.

    The set of products purchased by a customer during one shopping trip constitute a transaction, while the individual products that were purchased are the items.
  - Application: frequently purchased itemsets,

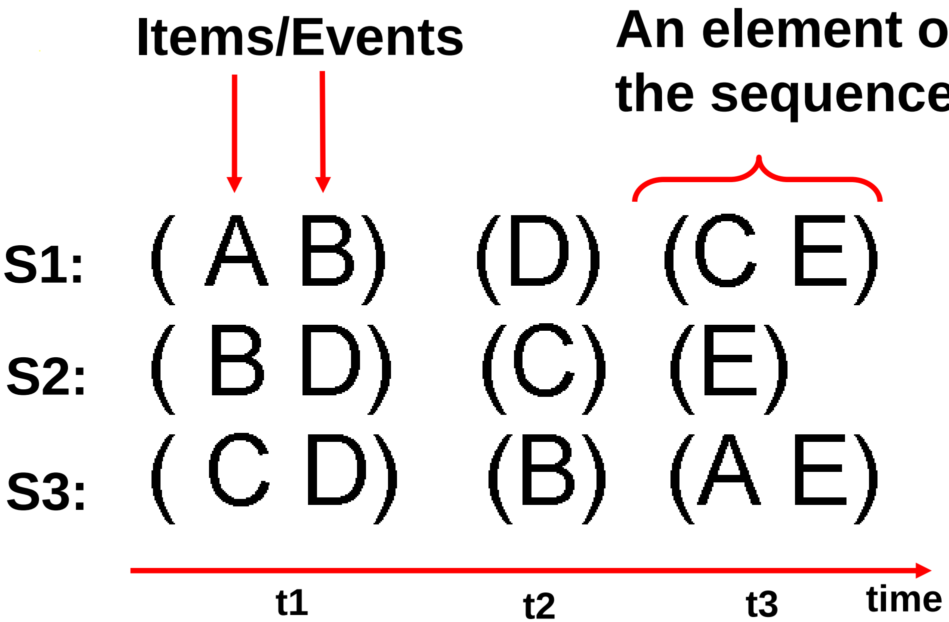
    e.g., {Diaper, Milk} with *support* = 60% (i.e., 3/5)

transaction

| *TID* | *Items* |
|---|---|
| 1 | Bread, Coke, Milk |
| 2 | Beer, Bread |
| 3 | Beer, Coke, Diaper, Milk |
| 4 | Beer, Bread, Diaper, Milk |
| 5 | Coke, Diaper, Milk |

# Ordered Data

- Sequences of transactions
  – Book checkouts, movie rental




- Pattern:
  E always follows D
- Application:
  marketing

# Data Quality

- What kinds of data quality problems?
- How can we detect problems with the data?
- What can we do about these problems?

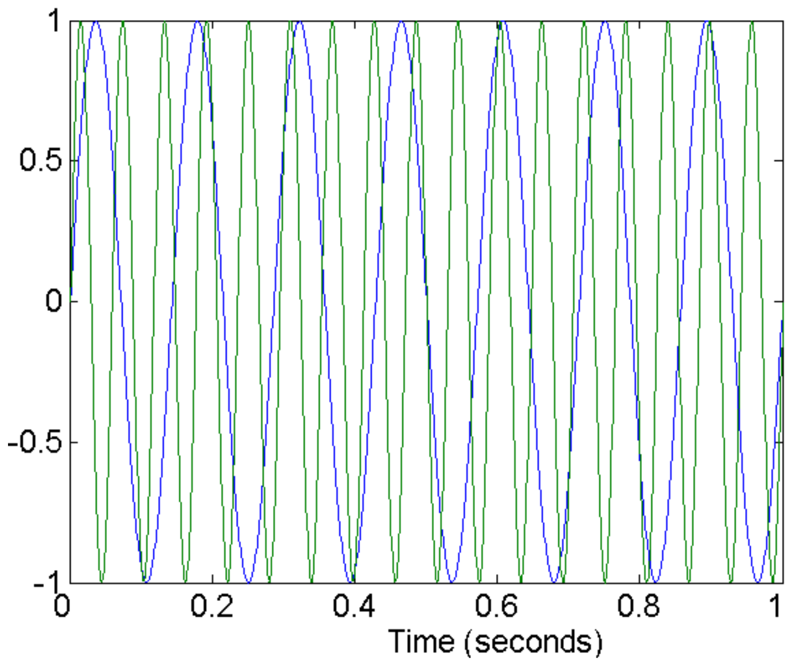
- Examples of data quality problems:
  - Noises and outliers
  - missing values
  - duplicate data

t1: Tom, M, 400, 60000, Master, Sports, Engineering, …

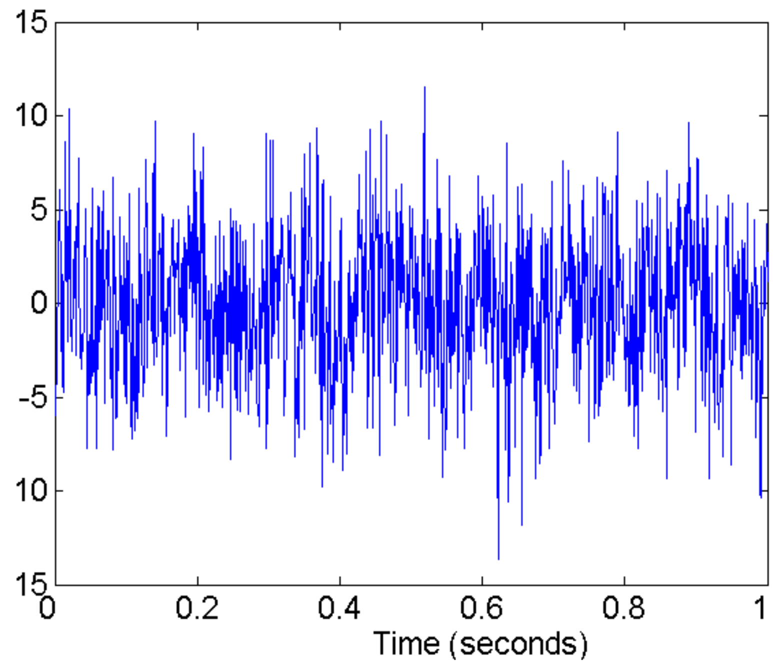
t2: Mary, F, 30, 50000, , Music, Marketing, …

# Noises

- Noise refers to modification of original values
  - E.g., distortion of a person's voice when talking on a poor phone and "snow" on television screen




**Two Sine Waves**

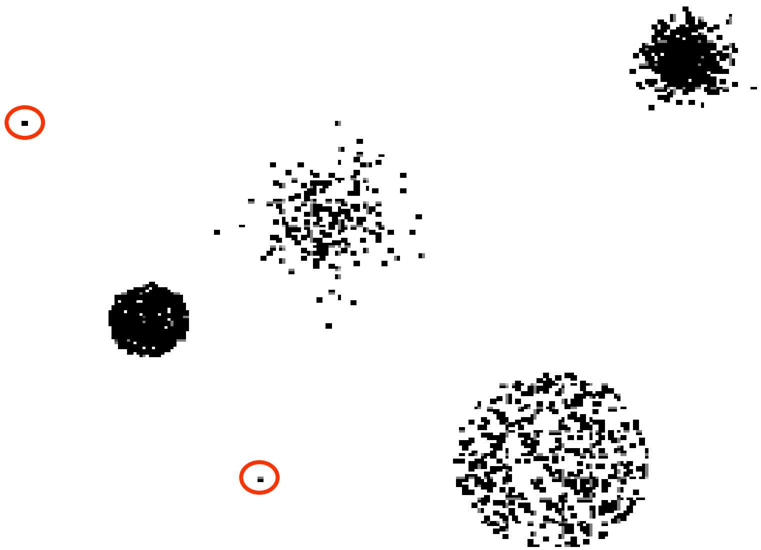



**Two Sine Waves + Noises**

# Outliers

- Data objects with characteristics that are considerably different than most of the other data objects in the data set
- Can be legitimate data objects and be of interest

E.g., in fraud and network intrusion detection, the goal is to find unusual objects or events.



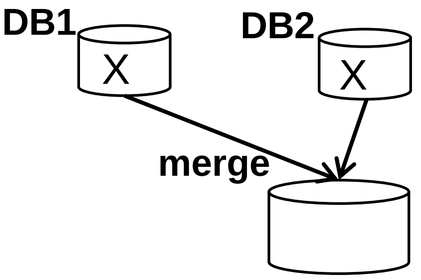
# Missing Values

- Reasons for missing values
  - Information is not collected
    (e.g., people decline to give their age and weight)
  - Attributes may not be applicable to all cases
    (e.g., annual income is not applicable to children)
- Handling missing values
  - Eliminate data objects
  - Ignore the missing value during analysis
  - Replace with
    - average of the dataset, average of the same class
    - mode of the dataset, mode of the same class, or
    - all possible values (weighted by their probabilities)

t1: Tom, M, 40, 40000, Master, Sports, Engineering, …, Engineer
t2: Mary, F, 25, , , Music, Marketing, …, Sales

# Duplicate Data

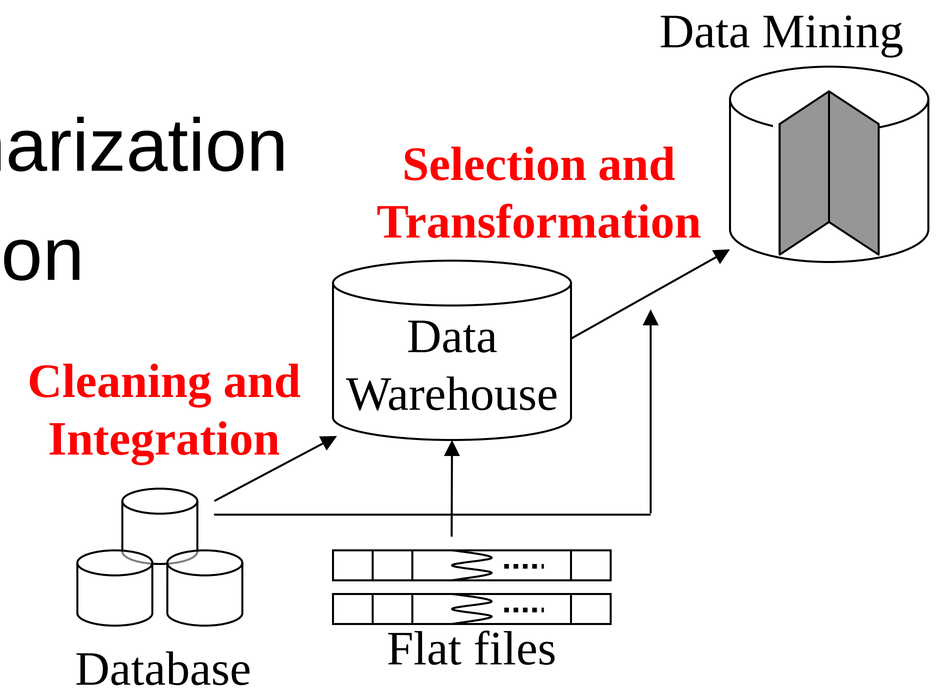
- Data set may include data objects that are duplicates, or almost duplicates of one another
  - Major issue when merging data from heterogeneous sources




- Examples:
  - Same person with multiple email addresses
- Data cleaning
  - Process of dealing with duplicate data issues

# Data Preprocessing Techniques

- Aggregation
- Sampling
- Dimensionality Reduction
- Feature subset selection
- Feature creation
- Discretization and Binarization
- Attribute Transformation



# Aggregation

- Combining two or more attributes (or objects) into a single attribute (or object)
- Purpose
  - Data reduction
    - Reduce the number of attributes or objects
  - Change of scale
    - Cities aggregated into regions, states, countries, etc
  - More “stable” data
    - Aggregated data tends to have less variability, e.g., moving average.

# Aggregation – Data Reduction

- Reduce the number of attributes

H/W Ratio

| … | Weight | Height | … |
|---|---|---|---|
| … | 60 | 160 | … |
| … | 50 | 155 | … |

| … | H/W Ratio | … |
|---|---|---|
| … | 16/6 | … |
| … | 155/50 | … |

Reduce the number of objects

| Id | Score |
|---|---|
| A | 70 |
| A | 90 |
| B | 80 |
| B | 60 |

| Id | Avg-Score |
|---|---|
| A | 80 |
| B | 70 |

# Aggregation – Change of Scale

- Change of Scale
  - Cities aggregated into regions, states, countries, etc
  - Temp aggregated into cold, mild, hot

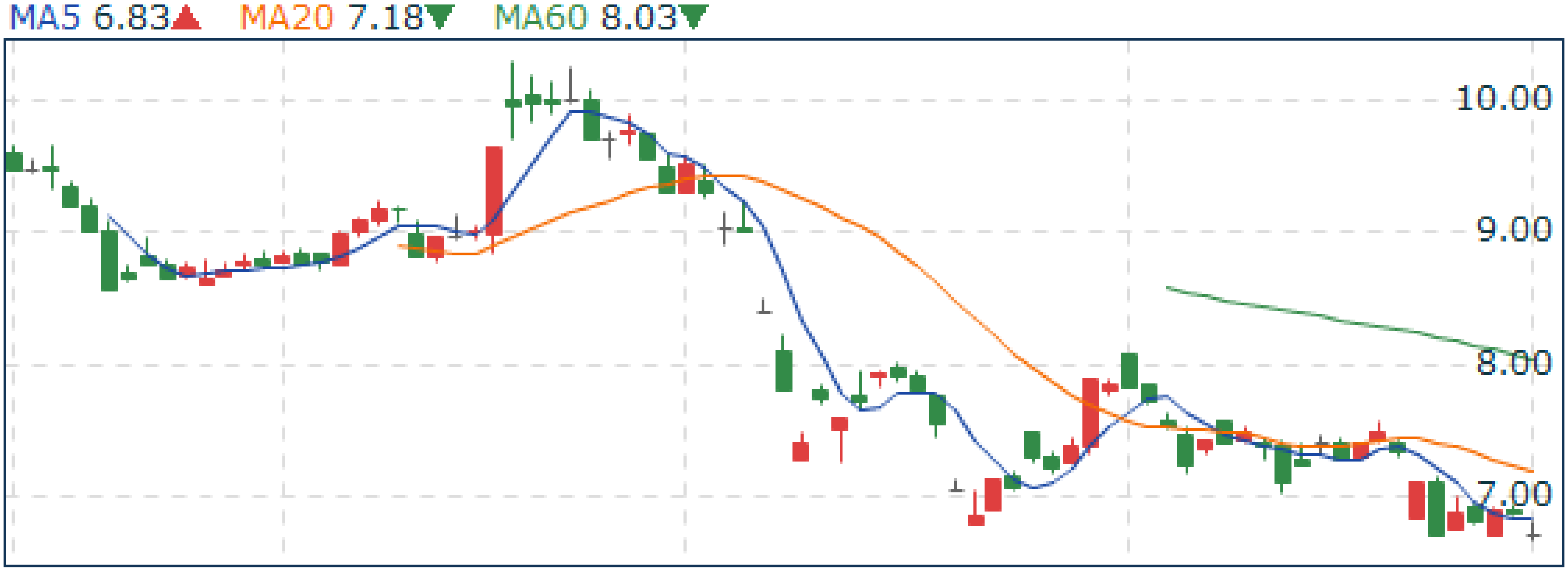
| Location | Temp. |
|---|---|
| Taichung | 25 |
| Yunlin | 26 |
| Miaoli | 24 |
| Changhua | 25 |
| Tainan | 33 |
| Kaohsung | 34 |
| Pindon | 36 |

| Location | Temp. | Count |
|---|---|---|
| Central_Taiwan | mild | 4 |
| Southern_Taiwan | hot | 3 |

# Aggregation – More Stable Data

- Aggregated data tends to have less variability (smoothing)

  e.g., moving average.



# Important Characteristics of Structured Data

## Dimensionality

- Curse of Dimensionality

## Sparsity

- Only presence counts

## Resolution

- Patterns depend on the scale

In the case of high dimensionality and high sparsity, cosine measure is better than Euclidean measure.

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}.$$

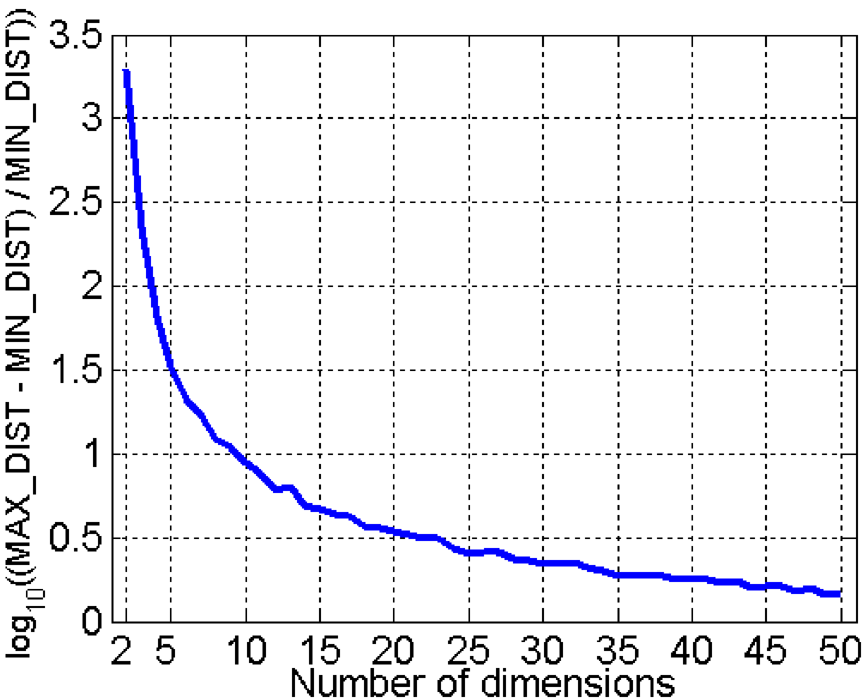
| | team | coach | pla y | ball | score | game | wi n | lost | timeout | season |
|---|---|---|---|---|---|---|---|---|---|---|
| Document 1 | 3 | 0 | 5 | 0 | 2 | 6 | 0 | 2 | 0 | 2 |
| Document 2 | 0 | 7 | 0 | 2 | 1 | 0 | 0 | 3 | 0 | 0 |
| Document 3 | 0 | 1 | 0 | 0 | 1 | 2 | 2 | 0 | 3 | 0 |

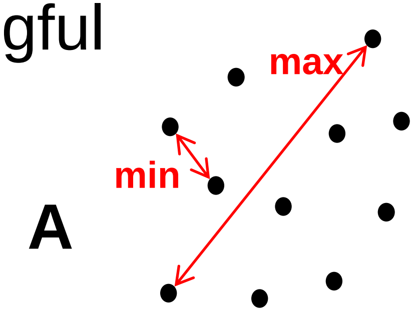
**Document-Term Matrix (Vector Space Model, or VSM)**

**Similarity between documents can be measured by inner product (or cosine index)**

# Curse of Dimensionality

- When dimensionality increases, data becomes increasingly sparse in the space that it occupies
- Definitions of density and distance between points, which is critical for clustering and outlier detection, become less meaningful




- Dataset A by randomly generating 500 data points
- Compute difference between max and min distance in the dataset A

$\log_{10}((\text{MAX\_DIST} - \text{MIN\_DIST}) / \text{MIN\_DIST}))$

# Dimensionality Reduction

- Purpose:
  - Avoid curse of dimensionality
  - Reduce amount of time and memory required by data mining algorithms
  - Allow data to be more easily visualized
  - May help to eliminate irrelevant features or reduce noise
- Techniques
  - Principal Component Analysis (PCA)
  - Singular Value Decomposition (SVD)
  - Others: supervised and non-linear techniques

| $A_1$ | $A_2$ | … | … | … | … | $A_n$ |
|---|---|---|---|---|---|---|
| … | … | … | … | … | … | … |

➡

| $A_1$ | $A_2$ | … | $A_k$ |
|---|---|---|---|
| … | … | … | … |

where $k < n$

# Reduction from Two to One Dimension

## Reduction by projection to X1-axis

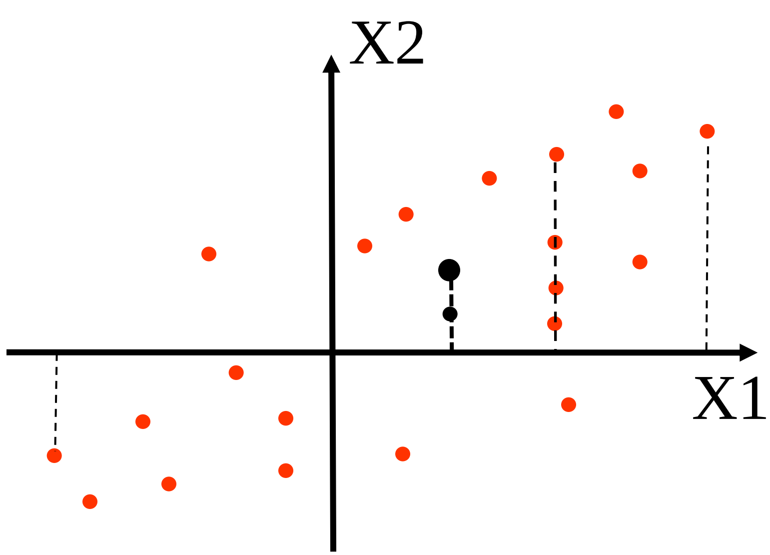
$[x_1\ x_2] \rightarrow [x_1]$

e.g.

a = [1, 1] → [1]

b = [1, 0.5] → [1]

***a*** **and** ***b*** **become indistinguishable after projection.**




## Reduction by PCA

$y_1 = a_{11}x_1 + a_{12}x_2$ (principle component)

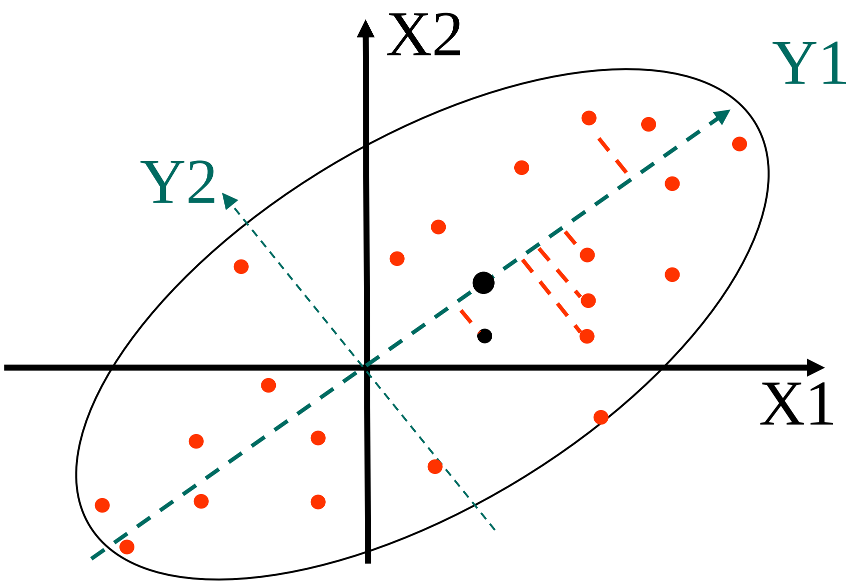
$y_2 = a_{21}x_1 + a_{22}x_2$

[x1 x2] → [y1 y2] → [y1]

e.g.,

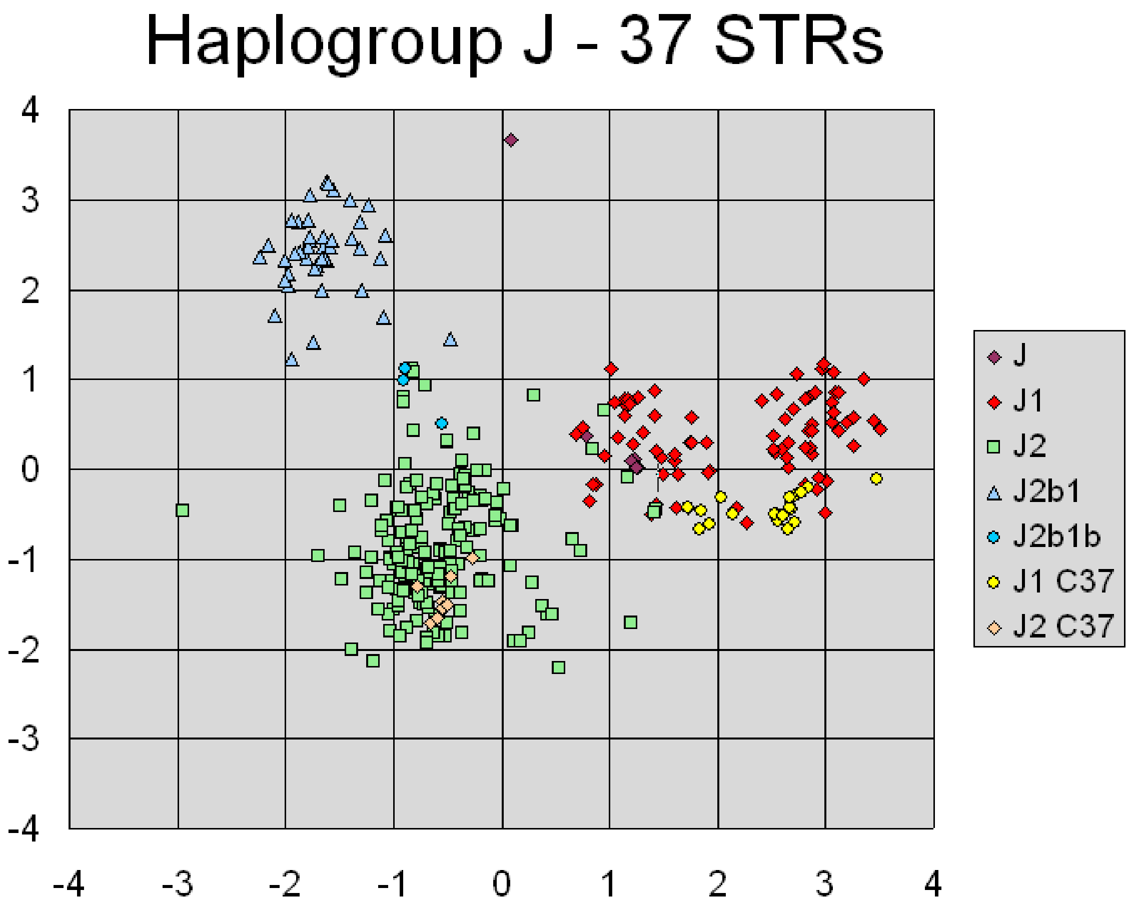
a = [1, 1] → [1.414, 0] → [1.414]

b = [1, 0.5] → [1.2, -0.3] → [1.2]




**Y1 captures the largest amount of variation in the data.**

# Application of PCA—from 37 to 2 Dimensions




A principal components analysis scatterplot of Y-STR haplotypes calculated from repeat-count values for 37 Y-chromosomal STR markers from 354 individuals. PCA has successfully found linear combinations of the different markers, that separate out different clusters corresponding to different lines of individuals' Y-chromosomal genetic descent.

# Principal Component Analysis (PCA)

- Goal is to find a projection that captures the largest amount of variation in data
- Given $N$ data vectors from $k$-dimensions, find $c$ ($\leq k$) orthogonal vectors that can be best used to represent data
  - The original data set is reduced to one consisting of $N$ data vectors on $c$ principal components (reduced dimensions)
- Each data vector is a linear combination of the $c$ principal component vectors
- Works for numeric data only


$[p_1\ p_2\ p_3\ p_4\ p_5\ p_6] \rightarrow [c_1\ c_2\ c_3]$

$c_1 = a_{11}p_1 + a_{12}p_2 + a_{13}p_3 + a_{14}p_4 + a_{15}p_5 + a_{16}p_6$

$c_2 = a_{21}p_1 + a_{22}p_2 + a_{23}p_3 + a_{24}p_4 + a_{25}p_5 + a_{26}p_6$

$c_3 = a_{31}p_1 + a_{32}p_2 + a_{33}p_3 + a_{34}p_4 + a_{35}p_5 + a_{36}p_6$

$c_4 = a_{41}p_1 + a_{42}p_2 + a_{43}p_3 + a_{44}p_4 + a_{45}p_5 + a_{46}p_6$

…

Calculate $a_{ij}$ from training data

1. Construct covariance matrix of the training data
2. Find the eigenvectors of the covariance matrix
3. The eigenvectors define the new space, $\mathbf{a}_1$, $\mathbf{a}_2$, …, $\mathbf{a}_6$ where $\mathbf{a}_i = \langle a_{i1}, a_{i2}, \ldots, a_{i6} \rangle$.

# Eigenvalues and Eigenvectors

Eigenvalues $\lambda$ and eigenvectors v of a square matrix A:

$$Av = \lambda v$$

E.g.:

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 4 \\ 0 & 4 & 9 \end{bmatrix}$$

$\lambda$: 11, 2, 1

v: $[0\ 1\ 2]^T$, $[1\ 0\ 0]^T$, $[0\ 2\ -1]^T$

Trace(A) = 14

To obtain $\lambda$ and v:

$$Av - \lambda v = 0,$$

$$(A - \lambda I)v = 0,$$

$$\det(A - \lambda I) = 0$$

$$\det(A - \lambda I) = \det\left(\begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 4 \\ 0 & 4 & 9 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right)$$

$$= \det \begin{bmatrix} 2-\lambda & 0 & 0 \\ 0 & 3-\lambda & 4 \\ 0 & 4 & 9-\lambda \end{bmatrix}$$

$$= (2-\lambda)\left[(3-\lambda)(9-\lambda) - 16\right]$$

$$= -\lambda^3 + 14\lambda^2 - 35\lambda + 22 = 0$$

Therefore,

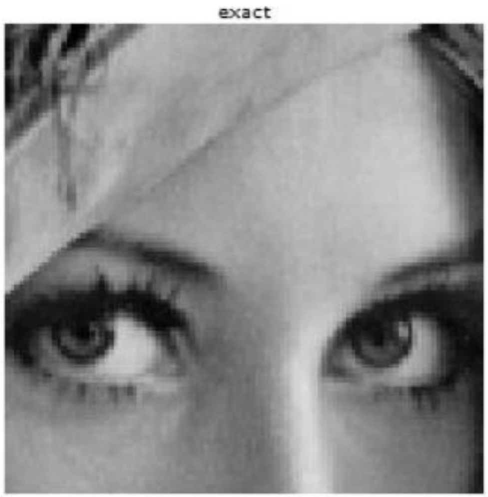
$\lambda = 11, 2, \text{or } 1$

**Trace(A): the sum of all its eigenvalues of matrix A; also the sum of the elements on the main diagonal.**

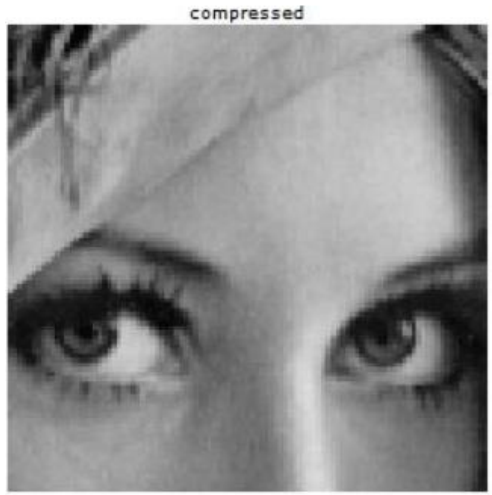
# Image Compression via Singular Value Decomposition

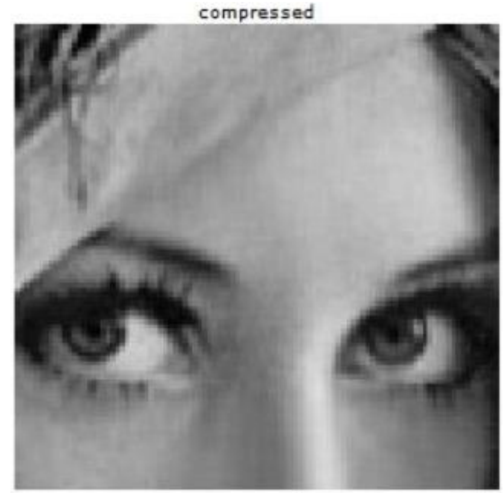
**From 61 singular values** **From 29 singular values**

340 pixels

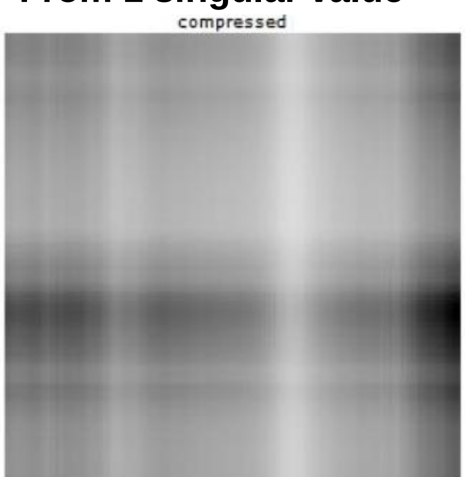



280 pixels




**From 1 singular value**




**Original Image**

- Grey levels of 1 pixel are represented by 8 bits (0~255).
- Image resolution: 340 × 280 pixels (= 95,200 pixels)
- File size: 340 × 280 × 8 bits = 95,200 bytes

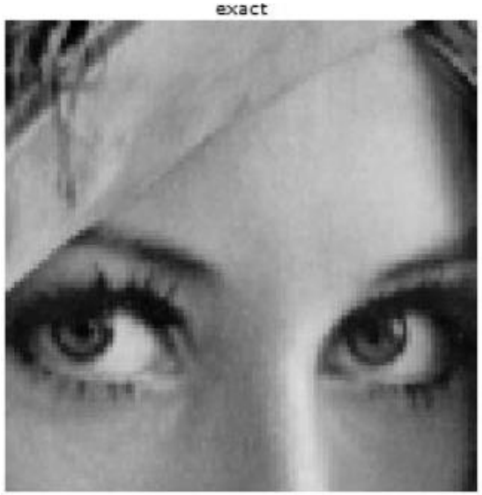
**Compression by using SVD**

- For 61 S.V.: (1+340+280) × 61 × 8 bits = 37,881 bytes.
- For 29 S.V.: (1+340+280) × 29 × 8 bits = 18,009 bytes.
- For 1 S.V.: (1+340+280) × 1 × 8 bits = 621 bytes.

# Compression via Singular Value Decomposition

exact

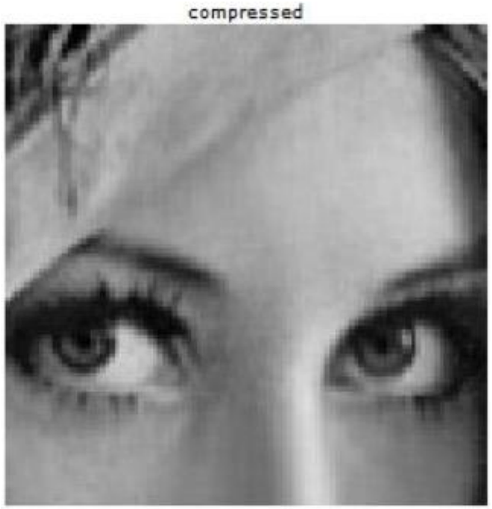
**340 pixels**



**280 pixels**

**From 29 singular values**

compressed



$$\underbrace{340\left[\begin{matrix} 200 & 180 & 70 & \dots \\ 198 & 173 & 87 & \dots \\ 124 & 36 & 94 & \dots \\ \dots & \dots & \dots & \dots \end{matrix}\right]}_{280} = 25 \underbrace{\begin{bmatrix} 20 \\ 14 \\ 8 \\ \dots \end{bmatrix}}_{340} \underbrace{\begin{bmatrix} 25 \\ 4 \\ 27 \\ \dots \end{bmatrix}^T}_{280} + \dots + 0.4 \underbrace{\begin{bmatrix} 15 \\ 7 \\ 32 \\ \dots \end{bmatrix}}_{\mathrm{u}_r} \underbrace{\begin{bmatrix} 3 \\ 54 \\ 6 \\ \dots \end{bmatrix}^T}_{\mathrm{v}_r^T}$$

Singular Value Decomposition :

$$M = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \dots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T + \dots + \sigma_r \mathbf{u}_r \mathbf{v}_r^T$$

Keep only the first $k$ terms :

$$M_k = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \dots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T, \qquad \text{where } k \le r$$

- $M_k$ is an approximation to $M$ that corresponds to keeping only the first $k$ singular values and the corresponding singular vectors.
- Note: singular vectors $\mathbf{u}_\mathbf{i} \in R^{340}$, $\mathbf{v}_\mathbf{i} \in R^{280}$.

- Original Image:
  file size: 340 × 280 × 8 bits = 95,200 bytes
- Compression by SVD
  For $k$ = 29, file size: (1+340+280) × 29 = 18,009 bytes

# Singular Value Decomposition

- Every matrix (symmetric or not, square or not) has a factorization of the form $M = U\Sigma V^T$, where U and V are $m\times m$ and $n\times n$ orthogonal matrices, and $\Sigma$ is an $m\times n$ "diagonal" matrix.

$$M=\begin{bmatrix}1&0&0&0&2\\0&0&3&0&0\\0&0&0&0&0\\0&4&0&0&0\end{bmatrix}=\begin{array}{c}\mathbf{U}\\ \begin{bmatrix}0&0&1&0\\0&1&0&0\\0&0&0&-1\\1&0&0&0\end{bmatrix}\\ \begin{array}{cccc}\mathbf{u}_1&&&\mathbf{u}_r\end{array}\end{array}\begin{array}{c}\mathbf{\Sigma}\\ \begin{bmatrix}4&0&0&0&0\\0&3&0&0&0\\0&0&\sqrt{5}&0&0\\0&0&0&0&0\end{bmatrix}\\ \\ \end{array}\begin{array}{c}\mathbf{V^T}\\ \begin{bmatrix}0&1&0&0&0\\0&0&1&0&0\\\sqrt{0.2}&0&0&0&\sqrt{0.8}\\0&0&0&1&0\\-\sqrt{0.8}&0&0&0&\sqrt{0.2}\end{bmatrix}\end{array}\begin{array}{l}\leftarrow\mathbf{v}_1^T\\ \\ \\ \\ \leftarrow\mathbf{v}_r^T\end{array}$$

$$M = U\Sigma V^T = \sigma_1 \mathrm{u}_1 \mathrm{v}_1^T + \ldots + \sigma_r \mathrm{u}_r \mathrm{v}_r^T$$

$$M=\begin{bmatrix}1&0&0&0&2\\0&0&3&0&0\\0&0&0&0&0\\0&4&0&0&0\end{bmatrix}=4\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\begin{bmatrix}0\\1\\0\\0\\0\end{bmatrix}^T+3\begin{bmatrix}0\\1\\0\\0\end{bmatrix}\begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix}^T+\sqrt{5}\begin{bmatrix}1\\0\\0\\0\end{bmatrix}\begin{bmatrix}\sqrt{0.2}\\0\\0\\0\\\sqrt{0.8}\end{bmatrix}^T$$

Size:
$M$ = 4*5 = 20 bytes
$M_{k=1}$ = 10 bytes

- Singular values in **Σ**, $\boldsymbol{\sigma_1, \ldots, \sigma_n}$, of $M_{m\times n}$ are the square roots of the eigenvalues of $M^TM$, denoted by $\boldsymbol{\sigma_1, \ldots, \sigma_n}$ and $\boldsymbol{\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n}$.

# Matrix Operations (Complimentary)

***M^T*: Transpose of a matrix *M***

***M^-1*: Inverse of a matrix *M***

**$MM^{-1} = M^{-1}M = I$**

***I* = Identity matrix**

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{2\times 3} = \begin{bmatrix} 1 & 3 & 0 \\ 4 & 2 & 1 \end{bmatrix} \qquad M^{T}{}_{3\times 2} = \begin{bmatrix} 1 & 4 \\ 3 & 2 \\ 0 & 1 \end{bmatrix}$$

$$MM^T = ?$$

Dimension of $MM^T = ?$

Dimension of $M^T M = ?$

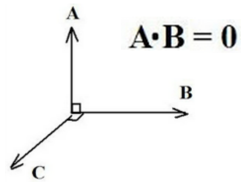
$$row\,vector : q_{1\times 4} = \begin{bmatrix} 2 & 6 & 3 & 1 \end{bmatrix}$$

$$column\,vector : p_{4\times 1} = \begin{bmatrix} 4 \\ 2 \\ 3 \\ 1 \end{bmatrix}$$

*pq* = ?  *qp* = ?  // Product (or Multiplication) of two matrice
$q \cdot p^T = ?$  // Dot product of two vectors

# Orthogonal and Diagonal Matrix

- A square matrix $M_{m \times m}$ whose columns form an orthonormal set is called an orthogonal matrix. // orthonormal: perpendicular and unit length
- A square matrix $M$ is orthogonal iff $M^{-1} = M^T$




$$UU^T = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \equiv I_4$$

// $\therefore U^T = U^{-1}$

$$VV^T = \begin{bmatrix} 0 & 0 & \sqrt{0.2} & 0 & -\sqrt{0.8} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \sqrt{0.8} & 0 & \sqrt{0.2} \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \equiv I_5$$

- A matrix is diagonal if its nondiagonal entries are all zero.
- An identity matrix, denoted by $I$, is a matrix whose diagonal entries are all one and nondiagonal entries are all zero.

$$\begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- $M^{-1}$ : inverse of $M$
- $MM^{-1} = I$
- $M^T$ : transpose of $M$

# Feature Subset Selection

- Another way to reduce dimensionality of data
- Redundant features
  - 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

| … | Purchase price | Sale tax |
|---|---|---|
| “” | 3,000 | 300 |
| “” | 20,000 | 2,000 |

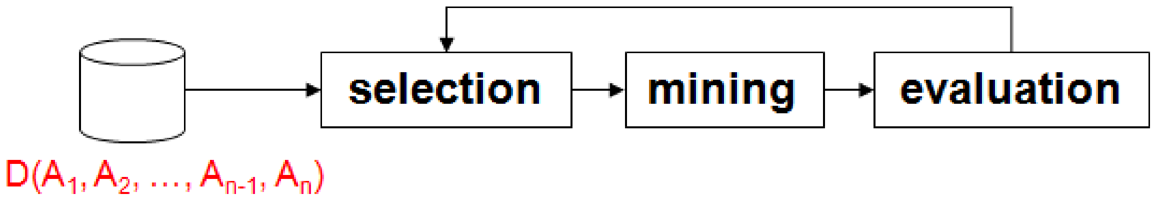
- Irrelevant features
  - contain no information that is useful for the data mining task at hand

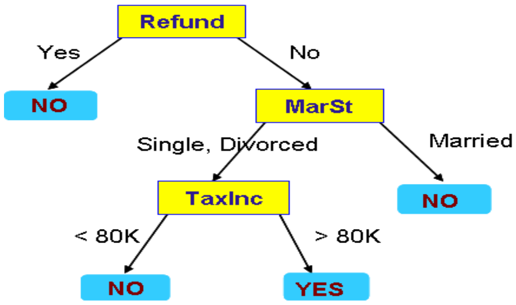
    e.g.: students' ID and Name are often irrelevant to the task of predicting students' GPA

| ID | Name | … | GPA |
|---|---|---|---|
| 1 | John | “” | 4.5 |
| 2 | Mary | “” | 3.2 |

# Feature Subset Selection Techniques

- Brute-force approach:
  - Try all possible feature subsets as input to data mining algorithm. (Problem: time complexity, e.g., pick 5 from 100 attributes)
- Embedded approaches:
  - Feature selection occurs naturally as part of the data mining algorithm (e.g. decision tree algorithm)
- Filter approaches:
  - Features are selected before data mining algorithm is run
- Wrapper approaches:
  - Use the data mining algorithm as a black box to find best subset of attributes



# Heuristic Feature Selection Methods

- Several heuristic feature selection methods:
  - Best single features under the feature independence assumption: choose by significance tests.
  - **Best step-wise feature selection**:
    - The best single-feature is picked first
    - Then next best feature condition to the first, ...
  - **Worst step-wise feature elimination:**
    - Repeatedly eliminate the worst feature
  - Best combined feature selection and elimination:
  - Optimal branch and bound:
    - Use feature elimination and backtracking

{A1, A2, A3, A4, A5, A6}

forward selection

Selected: {A5}

{A1, A2, A3, A4, A6}

forward selection

Selected: {A5, A2}

… …

**Greedy strategy?**

**Backward Elimination**

1. Eliminate A1? A2? A3? …

{A1, ~~A2~~, A3, A4, A5, A6}

2. Eliminate A1? A3? A4? …

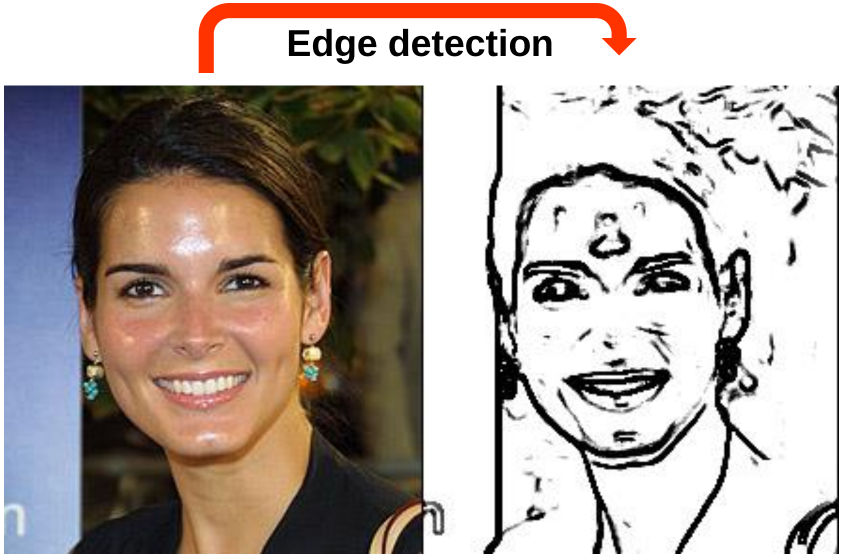
{A1, A3, A4, ~~A5~~, A6}

3. …

# Feature Creation

- Create new attributes that can capture the important information in a data set much more efficiently than the original attributes
- Three general methodologies:
  - Feature Extraction
    - extract certain type of edges related with presence of faces
    - domain-specific
  - Feature Construction
    - combining features

      (e.g., **BMI** by **Height / Weight**)
  - Mapping Data to New Space
    - e.g, Fourier, Wavelet transform

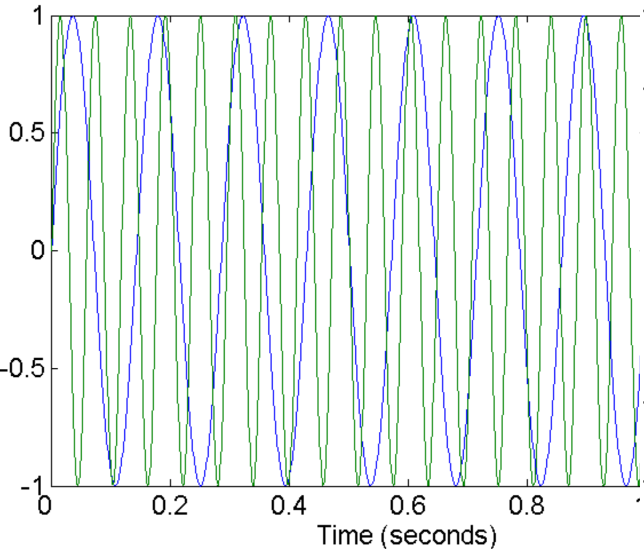


# Mapping Data to a New Space

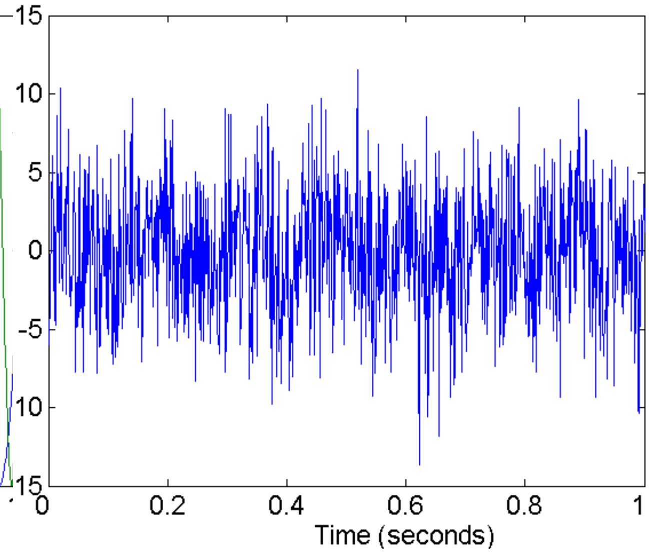
Fourier transform

Wavelet transform

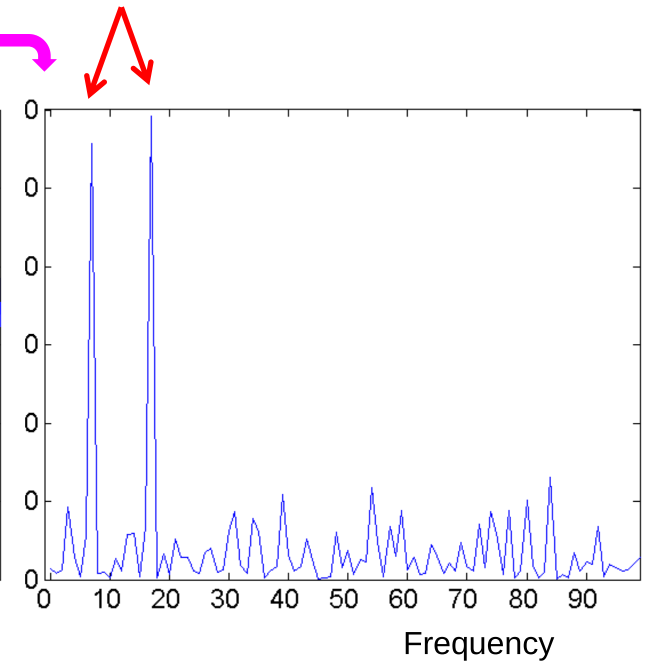
After the transformation, we can discover two obvious patterns




**Two Sine Waves**




**Two Sine Waves + Noises**




**Frequencies**

Transform time domain to frequency domain by Fourier transformation

# Conversion of A Categorical Attribute

- Some algorithms accepts only numeric values, e.g., neural networks.
- Convert ordinal values to integers.
- Convert nominal values to asymmetric binary attributes (or 1-of-*k* coding)
  - For nominal values: the semantics is lost. ***d*(Coke, Pepsi) < *d*(Coke, Latte)?**

| Ordinal values | Integer value |
|---|---|
| Awful | 0 |
| Poor | 1 |
| OK | 2 |
| Good | 3 |
| Great | 4 |

| Nominal values | x1 | x2 | x3 | x4 | x5 |
|---|---|---|---|---|---|
| Coke | 1 | 0 | 0 | 0 | 0 |
| Pepsi | 0 | 1 | 0 | 0 | 0 |
| Sprint | 0 | 0 | 1 | 0 | 0 |
| Mocha | 0 | 0 | 0 | 1 | 0 |
| Latte | 0 | 0 | 0 | 0 | 1 |

**1-of-*k* coding**

t1: Tom, M, 60, 60000, Good, Coke

t2: Mary, F, 30, 50000, Poor, Mocha

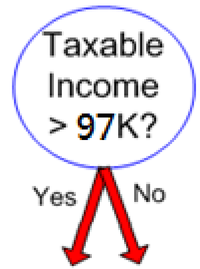
t1': Tom, 1, 0, 60, 60000, 3, **1, 0, 0, 0, 0**

t2': Mary, 0, 1, 30, 50000, 1, **0, 0, 0, 1, 0**

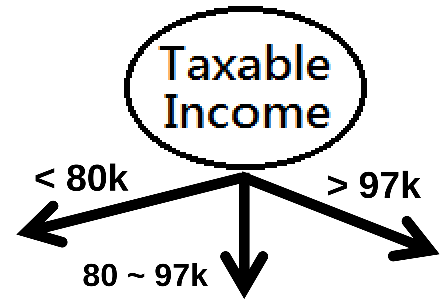
# Discretization of Continuous Attributes

Some algorithms accept only discrete attributes, e.g., decision trees

| *Tid* | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | No |
| 2 | No | Married | 100K | No |
| 3 | No | Single | 70K | No |
| 4 | Yes | Married | 120K | No |
| 5 | No | Divorced | 95K | Yes |
| 6 | No | Married | 60K | No |
| 7 | Yes | Divorced | 220K | No |
| 8 | No | Single | 85K | Yes |
| 9 | No | Married | 75K | No |
| 10 | No | Single | 90K | Yes |




**Binary discretizaiton**




**Multi-way discretizaiton**

| Cheat | No | No | No | Yes | Yes | Yes | No | No | No | No |
|---|---|---|---|---|---|---|---|---|---|---|
| Taxable Income (Sorted Values) | 60 | 70 | 75 | 85 | 90 | 95 | 100 | 120 | 125 | 220 |

| Split Positions | 55 | | 65 | | 72 | | 80 | | 87 | | 92 | | 97 | | 110 | | 122 | | 172 | | 230 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > |
| Yes | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 1 | 2 | 2 | 1 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 0 |
| No | 0 | 7 | 1 | 6 | 2 | 5 | 3 | 4 | 3 | 4 | 3 | 4 | 3 | 4 | 4 | 3 | 5 | 2 | 6 | 1 | 7 | 0 |
| Gini | 0.420 | | 0.400 | | 0.375 | | 0.343 | | 0.417 | | 0.400 | | <u>*0.300*</u> | | 0.343 | | 0.375 | | 0.400 | | 0.420 | |

**Discretization by Gini index:**
1. **Sort the values**
2. **Determine the discretization boundary by Gini value**

# Entropy-Based Discretization

Given a set of samples S, if S is partitioned into two intervals S1 and S2 using boundary T.

The boundary that minimizes the entropy function over all possible boundaries is selected as a binary discretization.

- If more partitions are desired, the process is recursively applied until some stopping criterion is met.

$$E(S,T)=\frac{|S_1|}{|S|}Ent(S_1)+\frac{|S_2|}{|S|}Ent(S_2)$$

$$\max_i\left(Ent(S)-E(S,T_i)\right)$$

**Discretize A3 by entropy of class labels:**

1. **Sort the values of A3**
2. **Determine the discretization boundary by entropy value of the Class attribute**

**To discretize A3** ⬇ **$S$**

| A2 | A3 | Class |
|---|---|---|
| | 2 | A |
| | 5 | A |
| | 6 | B |
| | 8 | A |
| | 9 | A |
| | 12 | A |
| | 15 | B |
| | 18 | B |
| | 20 | B |
| | 22 | B |
| | 24 | B |
| | 26 | A |
| | 30 | B |

$T_6$ $S_1$ $S_2$

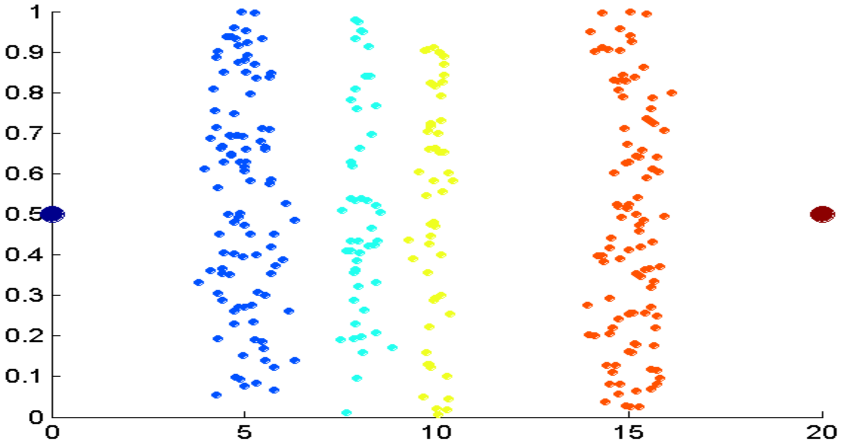
**To discretize A3** ⬇ **$S$**

| A2 | A3 | Class |
|---|---|---|
| | 2 | A |
| | 5 | A |
| | 6 | B |
| | 8 | A |
| | 9 | A |
| | 12 | A |
| | 15 | B |
| | 18 | B |
| | 20 | B |
| | 22 | B |
| | 24 | B |
| | 26 | A |
| | 30 | B |

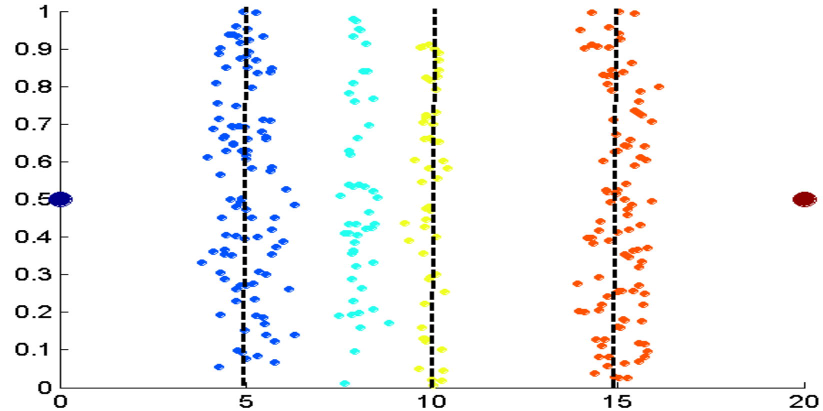
$T_{10}$ $S_1$ $S_2$

# Discretization Without Using Class Labels

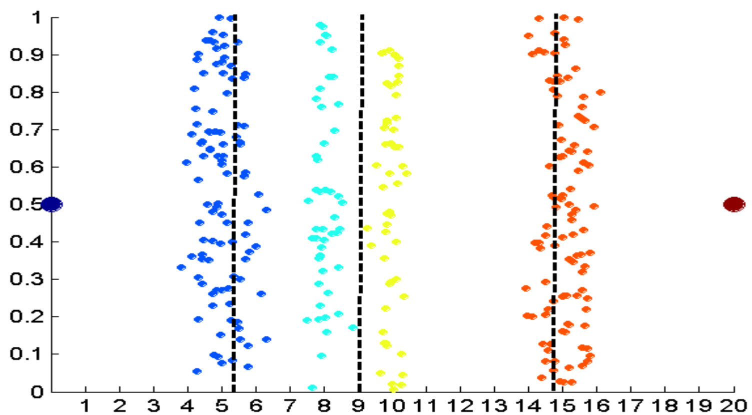
**D: { 2, 3, 4, 4, 5, 6, 10, 13, 14, 15, 25, 26, 30, 40, 41, 42, 43, 44, 45}**



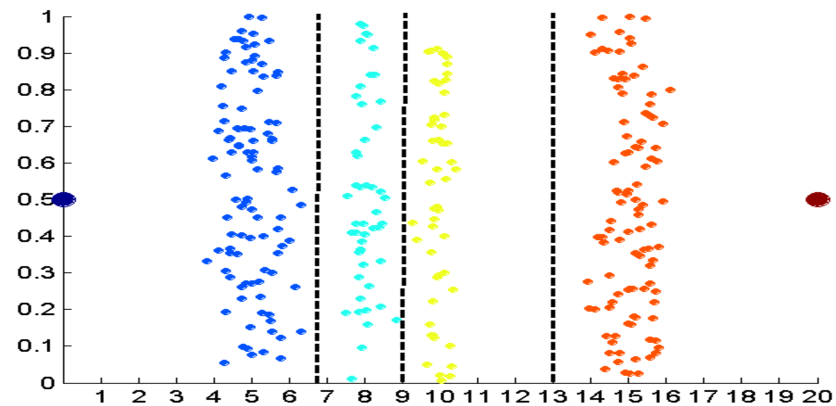

**Data**




**Equal interval width (of x axis)**



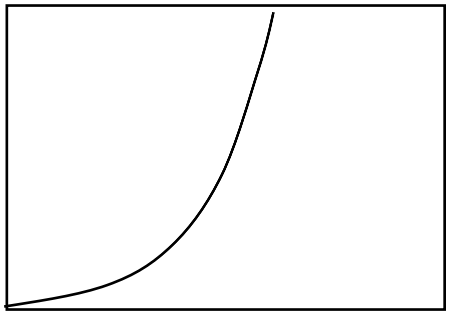

**Equal frequency (data count)**

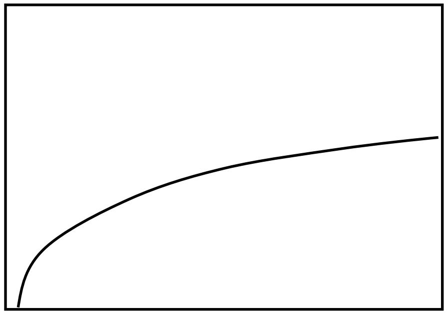


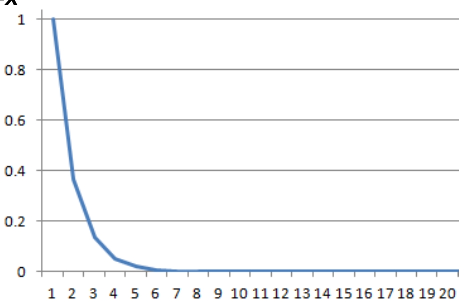

***K*-means**

# Attribute Transformation

- A function that maps the entire set of values of a given attribute to a new set of replacement values such that each old value ***x*** can be identified with one of the new values
  - incremental functions: $x^k$, $e^x$, $\log_n(x)$
  - decreasing functions: $x^{-n}$, $e^{-x}$



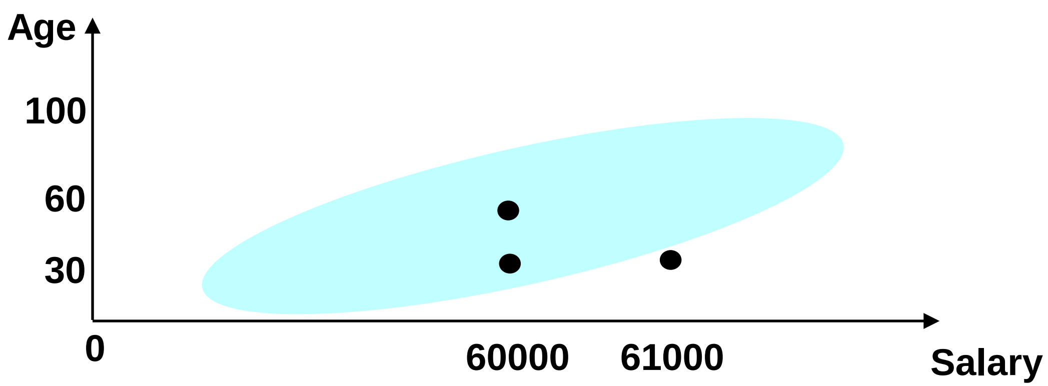

Order preserving (but range changed) transformations

e.g.: 0.000001, 0.001, 1000, 1000000 $\xrightarrow[\log(x)]{}$ -6, -3, 3, 6

# Scale Problem: Normalization (or Standardization)

Why is normalization necessary?
Which two are more similar?
i.e., $d$(T1,T2) < $d$(T2,T3)?

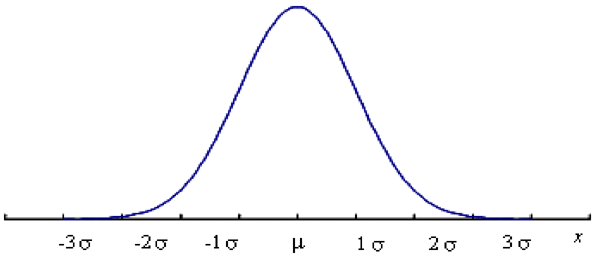
| | Age | Salary |
|---|---|---|
| **T1** | 60 | 60000 |
| **T2** | 30 | 60000 |
| **T3** | 30 | 61000 |



# Scale Problem: Normalization (or Standardization)

- z-score normalization

$$v' = \frac{v - mean_A}{stand\_dev_A}$$




**$\mu$ : mean**
**$\sigma$ : standard deviation**

| | Age | Salary |
|---|---|---|
| **T1** | 60 | 60000 |
| **T2** | 30 | 60000 |
| **T3** | 30 | 61000 |

**Age: ($\mu$ : 50, $\sigma$ : 10)**
**60→*v'*? 30→*v''*?**

- min-max normalization (usually 0~1 or -1~1)

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

**Age: (0, 120) → (0, 1)  60→*v'*? 30→?**
**Salary: (20000, 100000) → (0, 1)  60000→?**

- normalization by decimal scaling

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

**{578, 85, 4627} → {0.0578, 0.0085, 0.4627}**

# Distance between Two Objects

T1: Tom, M, 60, 60000, Good, Coke
T2: Mary, F, 30, 50000, Poor, Mocha

*d*(T1, T2)?

**Convert categorical attributes**

T1’: Tom, 1, 0, 60, 60000, 3, 1, 0, 0, 0, 0
T2’: Mary, 0, 1, 30, 50000, 1, 0, 0, 0, 1, 0

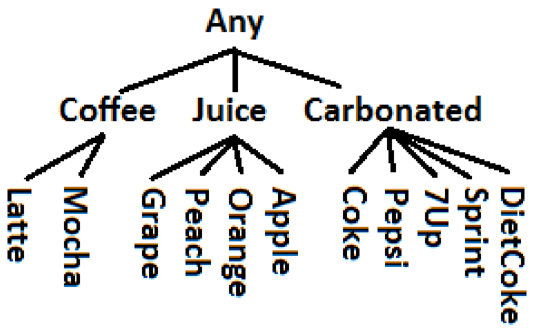
*d*(T1’, T2’)?

**Discard irrelevant attribute NAME, and normalize attributes.**
**e.g. normalize the range of each attribute to (0, 1)**

T1”: 1, 0, 0.50, 0.50, 0.75, 1, 0, 0, 0, 0
T2”: 0, 1, 0.25, 0.38, 0.25, 0, 0, 0, 1, 0

*d*(T1”, T2”)?

**Note:**
**Other alternatives for categorical attributes are possible, e.g., use simple matching or distance hierarchy**



# Exercise

1. What are the distances *d*(T1,T2) and *d*(T2,T3) under z-score normalization? Assume mean and standard deviation for Age and Salary are (70, 10) and (50000, 1000), respectively.
2. What are the distances *d*(T1,T2) and *d*(T2,T3) by min-max normalization to the range (0, 1)? Assume (min, max) for Age and Salary are (0, 120) and (20000, 100000), respectively.
3. What are the distances *d*(T1,T2) and *d*(T2,T3) if normalization by decimal scaling is applied to Age and Salary, respectively?

| | **Age** | **Salary** |
|---|---|---|
| **T1** | 60 | 60000 |
| **T2** | 30 | 60000 |
| **T3** | 30 | 61000 |