COMP5121
Data Mining and Data Warehousing Applications

Week 8: Course Review for Mid-term Exam

Dr. Fengmei Jin

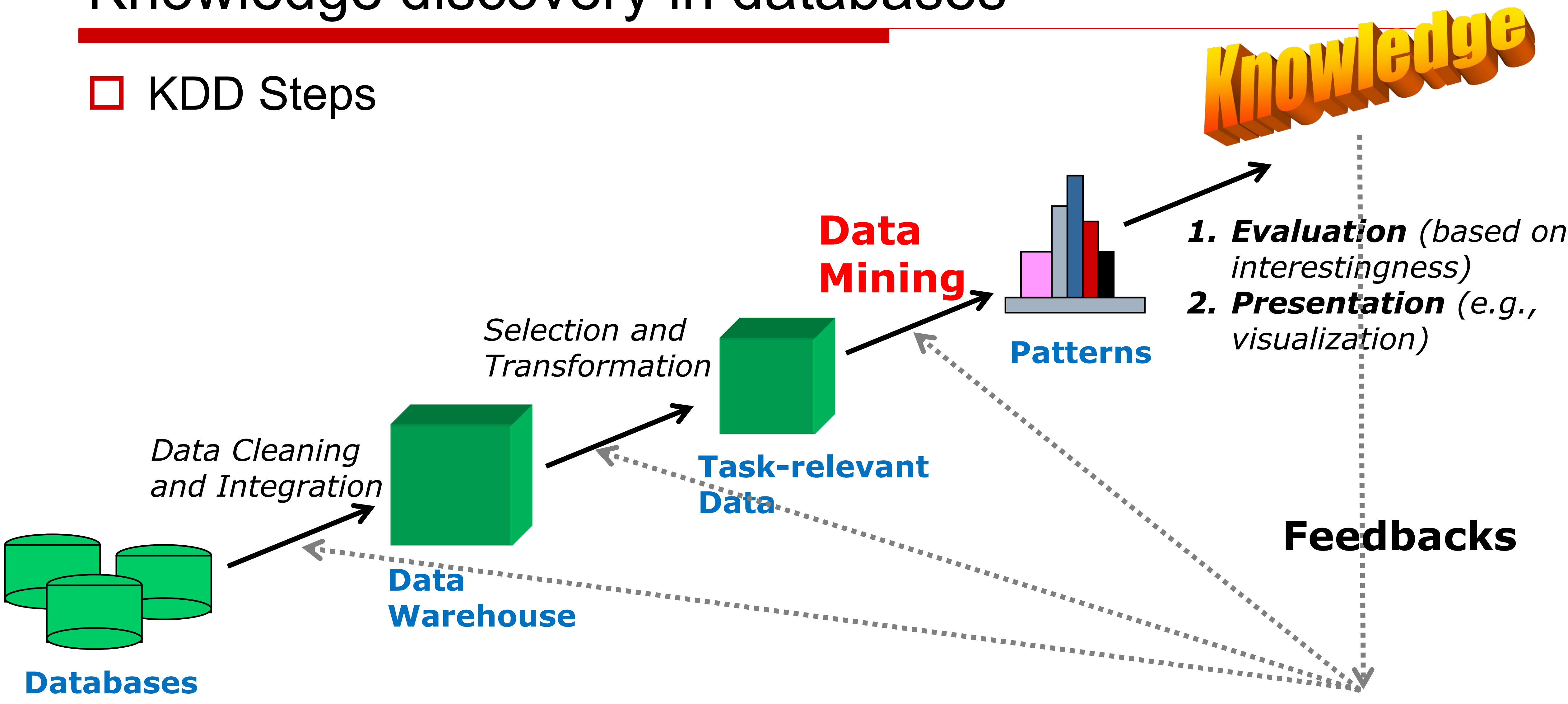
- Email: fengmei.jin@polyu.edu.hk
- Office: PQ747 (+852 3400 3327)
- Consultation Hours: 2:30-4:30pm every Thursday

The KDD process

KNOWLEDGE DISCOVERY FROM DATA

Knowledge discovery in databases

- KDD Steps



Data Objects

- Databases/Datasets are made up of data objects.
- A data object represents an entity.
  - Sales DB: customers, store items, sales
  - Medical DB: patients, treatments
  - University DB: students, professors, courses
- Database rows → data objects, described by attributes
  - Also called as *samples*, *examples*, *instances*, *data points*, *tuples*
- Database columns → attributes
  - Also called as *data field*, *characteristic*, *dimension*, *feature*, *variable*

Classify Attribute Types

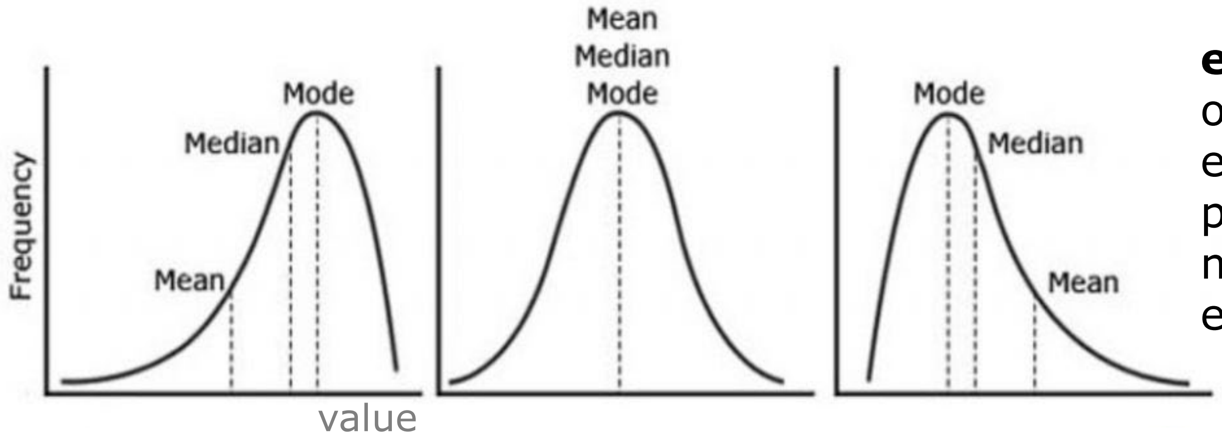
- To describe a **qualitative** feature of an object that does not provide actual size or quantity – **nominal, binary, ordinal**
  - Values are typically words representing categories.
  - Integers are used to embed categories as codes.
    - 0 for small drink size, 1 for medium, and 2 for large.

- To provide **quantitative** measurements of an object – **numeric**
  - **Interval-scaled**: No true zero.
  - **Radio-scaled**: True zero, enabling meaningful ratios.

Basic Statistical Descriptions of Data (I)

- **Motivation**: To better understand the data, identify properties of the data, and highlight what values shall be treated as *noise*
  - Central tendency: to measure the middle or center of the data
    - Mean: The average of the data (sensitive to extremes/outliers)
    - Median: The middle value when data is ordered (a more robust measure when data is *skewed*)
    - Mode: The most frequently occurring value

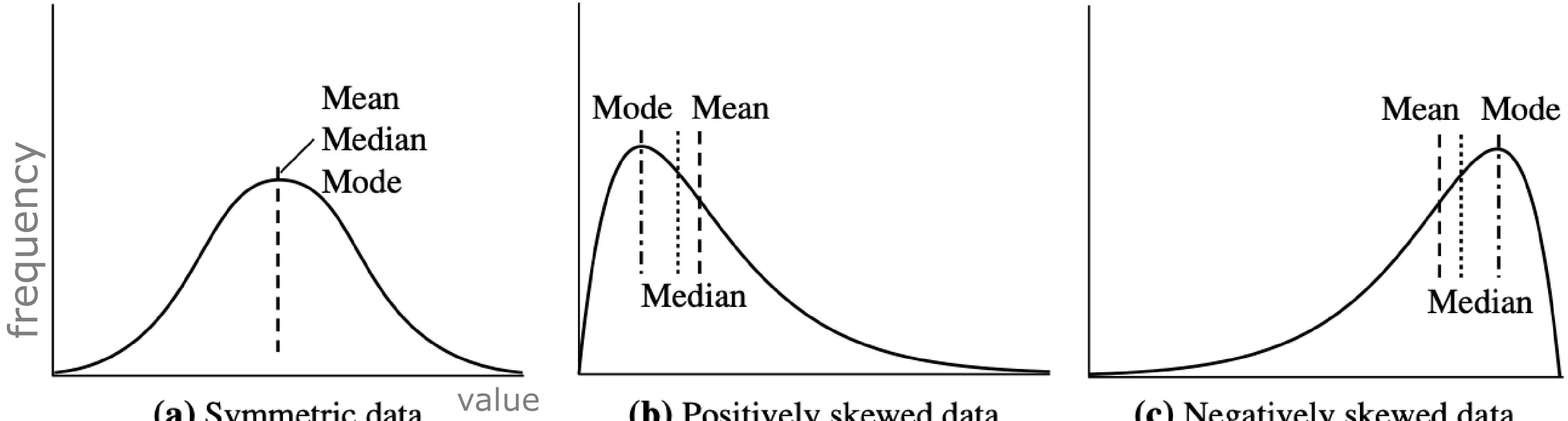
example: a strong middle class and fewer low-income households, e.g., Sweden, Finland, Denmark.



example: a small group of extremely high-income earners and a large population of low- to middle-income workers, e.g., New York, HK

Symmetric vs. Skewed Data

- Compare the central tendency (i.e., median, mean and mode) of symmetric, positively-skewed and negatively-skewed data

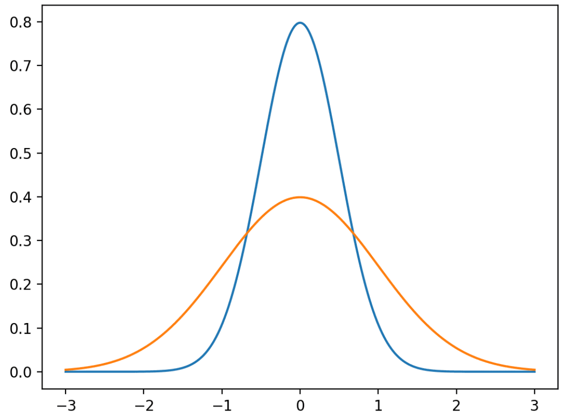


(a) Symmetric data **(b)** Positively skewed data **(c)** Negatively skewed data

the long tail is on the* ***positive* *side (higher values)*

Basic Statistical Descriptions of Data (II)

- **Motivation**: To better understand the data, identify properties of the data, and highlight what values shall be treated as *noise*
  - Data dispersion: how are the data spread out?
    - Range: difference between max and min values
    - Interquartile Range (IQR): Measures spread around the **median**
    - Variance / Standard Deviation: Indicate deviation from the **mean**

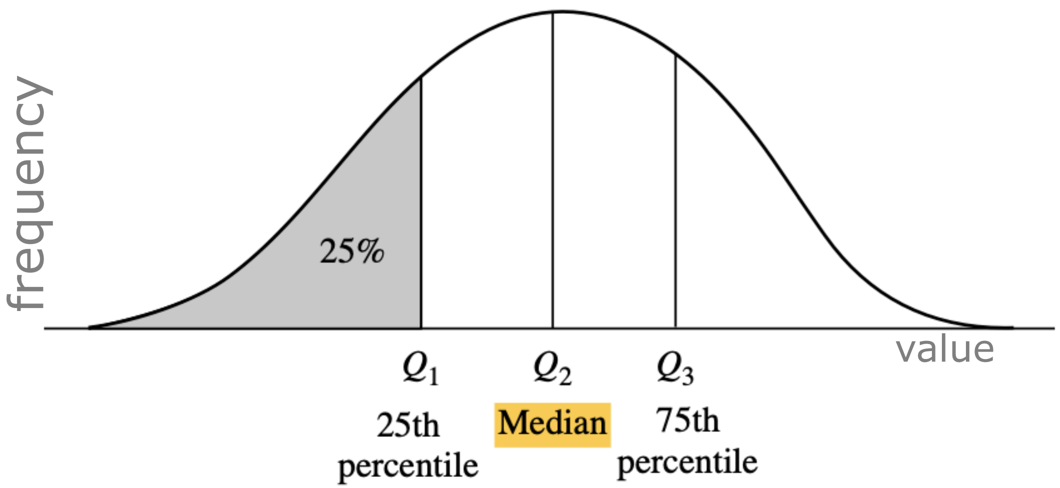


Measures of Data Dispersion

30, 36, $\boxed{47}$, 50, 52, $\boxed{52}$, 56, 60, $\boxed{63}$, 70, 70, 110.

Q1 Q3

- ***q*-Quantiles**: $q-1$ data points where the data distribution is split into q equal-size consecutive sets, e.g., 2-quantile (i.e., *median*), 4-quantiles (called *quartile*), 100-quantiles (called *percentiles*)



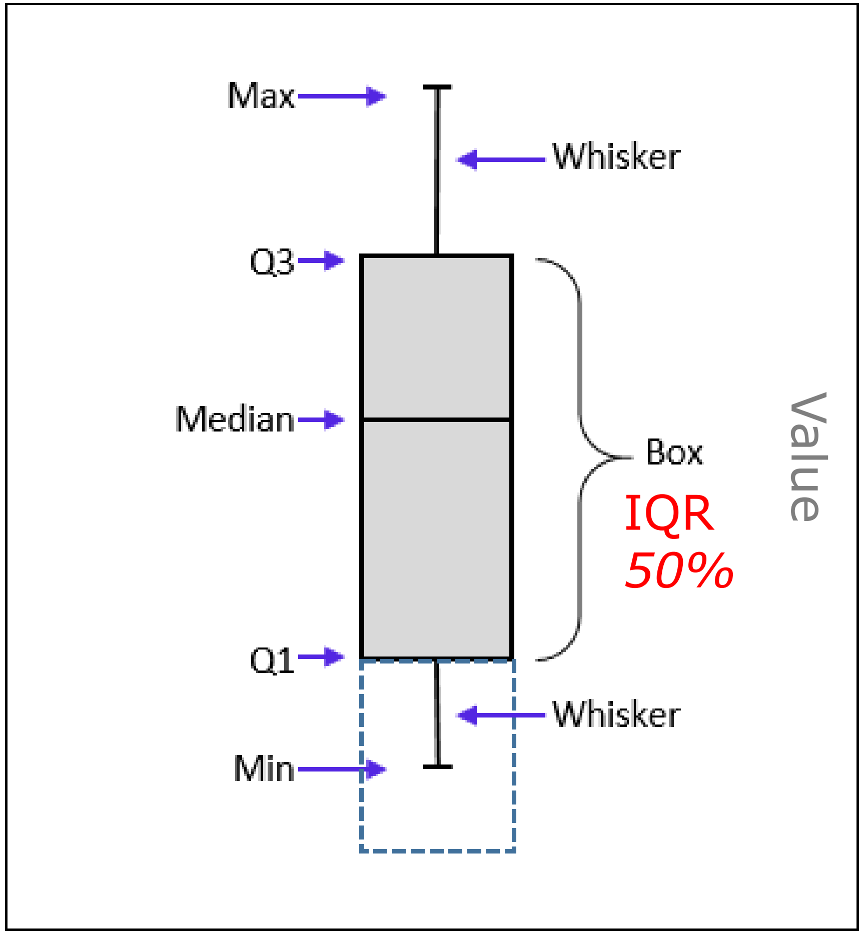
A plot of the data distribution for some attribute X. The quantiles plotted are quartiles. The three quartiles divide the distribution into four equal-size consecutive subsets. The second quartile corresponds to the median.

Interquartile Range (IQR)

- to identify the spread of the central portion of a dataset
- calculated as the difference between:
  - Upper quartile, Q3
  - Lower quartile, Q1
  - **IQR = Q3 – Q1** = 63 – 47 = 16

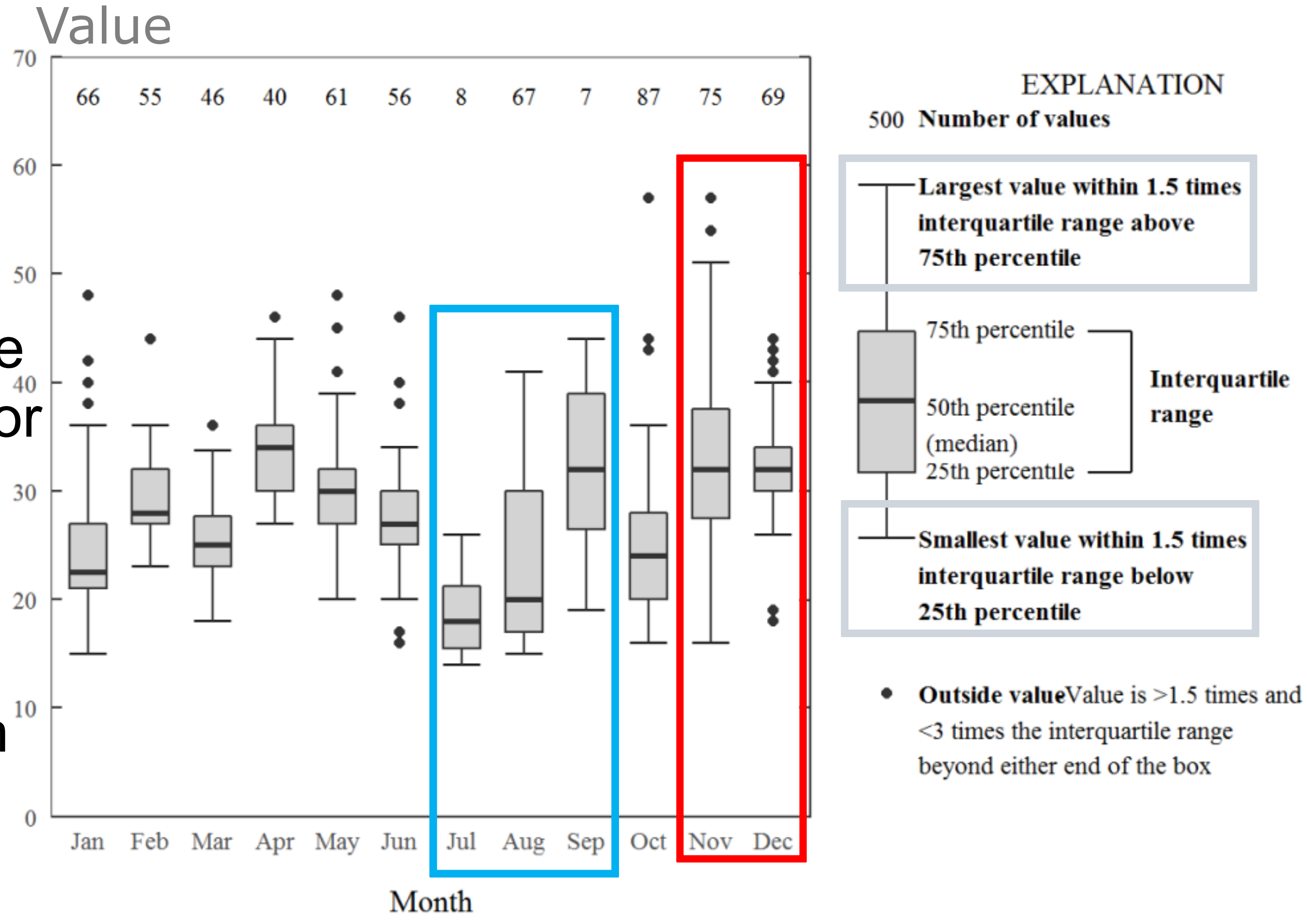
Graphic Displays: **Boxplot**

- **Quartiles** (i.e., 4-quantiles)
  - **Five-number summary**: min, Q1, median (Q2), Q3, max
  - **Boxplot**: data is represented by a box
    - IQR: the two ends of the box are at Q1 and Q3, i.e., the height of the box is IQR
    - Median: marked by a line within the box
    - Whiskers: two lines outside the box extended to min and max



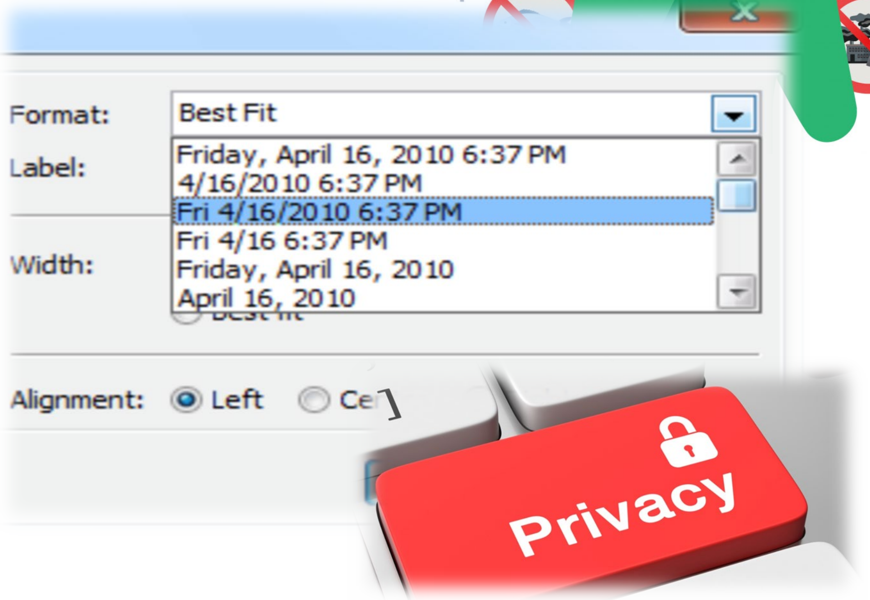
Graphic Displays: **Boxplot's Application**

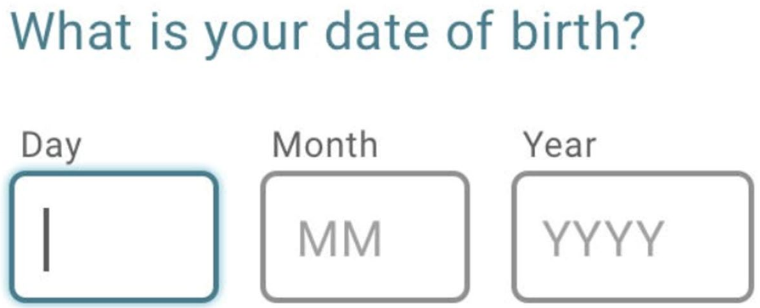
- **Outliers**
  - data points beyond a specified threshold
    - Usually, outside values are $1.5 \times IQR$ higher than Q3 or lower than Q1
  - Plotted individually
    - The whiskers shall stop at the most extreme low/high observations within $1.5 \times IQR$ of the quartile.
    - Then, outliers show up.



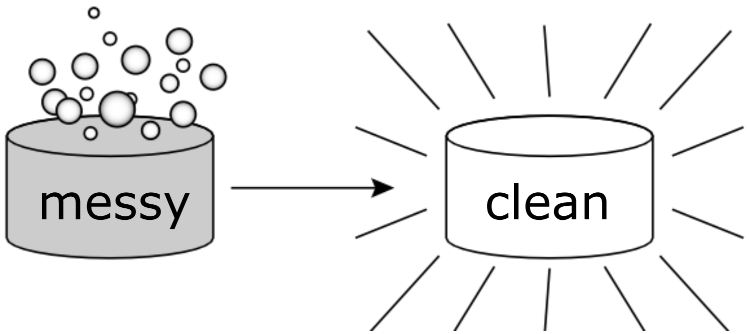
Why Preprocess the Data? Data Quality!

- Data quality depends on the intended use of data.
- Multidimensional views of data quality:
  - **Accuracy**: data must correctly reflect the real-world scenario without errors or noise.
  - **Completeness**: all required data fields should be present and valid.
  - **Consistency**: data should follow the same rules and format across all records.
  - **Timeliness**: data should be up-to-date.
  - **Believability**: data should be credible and from trusted sources.
  - **Interpretability**: data should be clear and understandable.

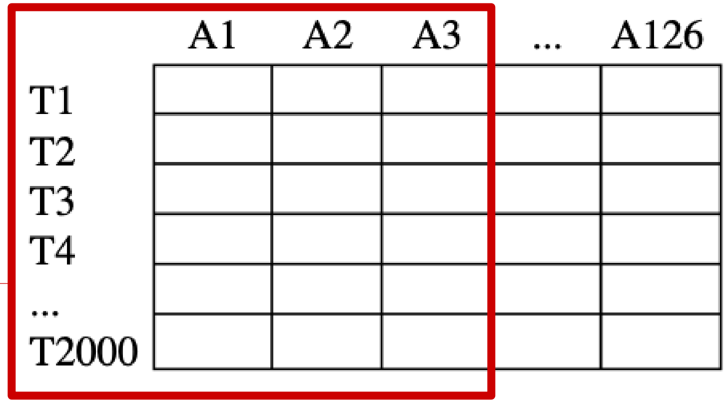




Major Tasks of Data Preprocessing



- Data **Cleaning**
  - To fill in missing data, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- Data **Integration** (e.g., Bill Gates, William Gates, B. Gates, …)
  - To merge multiple databases into a coherent data store
- Data **Reduction** (efficiency of mining process)
  - To obtain a reduced representation of the data with similar results
- Data **Transformation**
  - To normalize data for similarity-based mining (e.g., age vs salary)



data cube

DATA WAREHOUSE AND OLAP

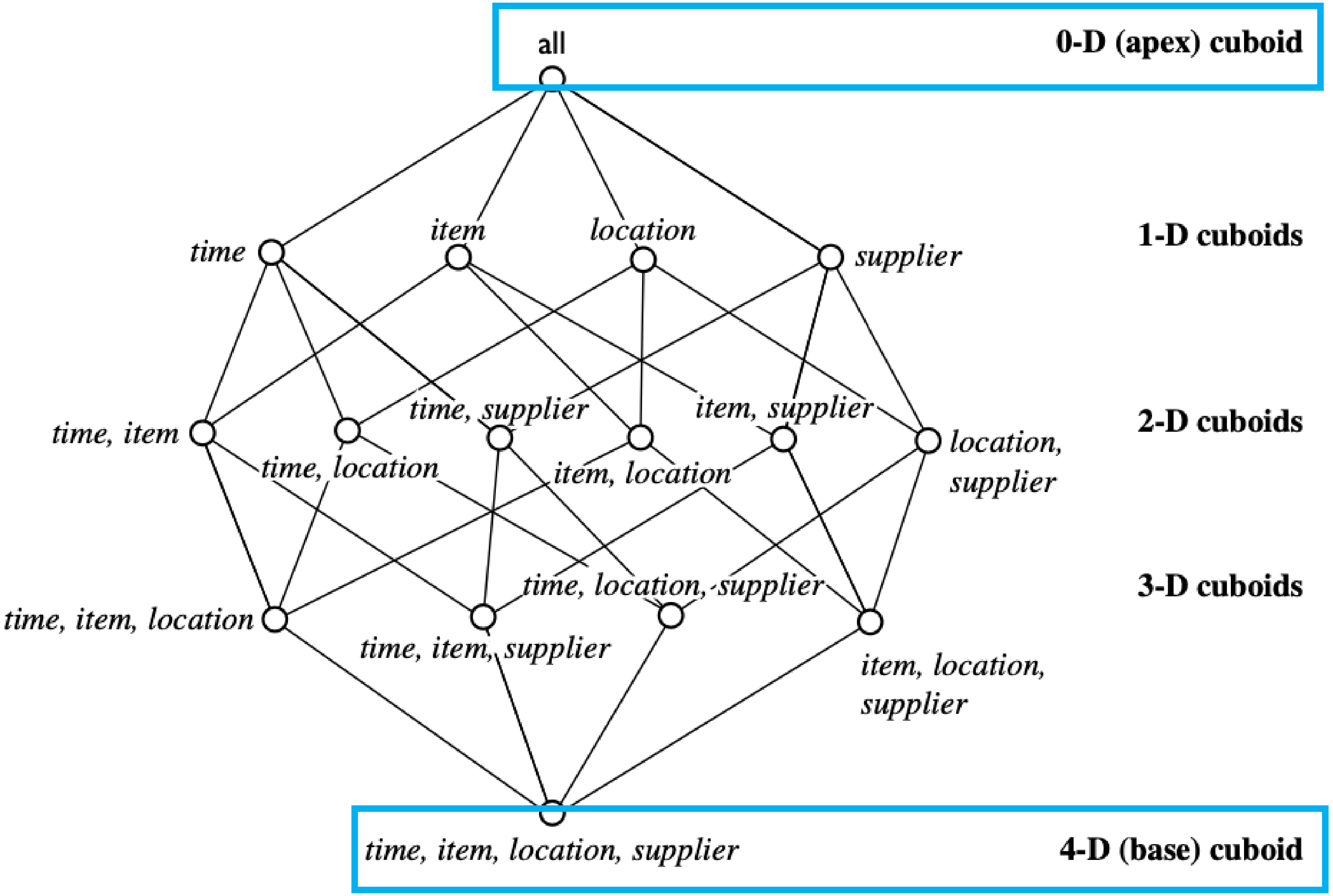
Why a Separate Data Warehouse?

- High performance for both systems:
  - DBMS – tuned for OLTP: access methods, indexing, hashing, concurrency control, recovery
  - Warehouse – tuned for OLAP: complex OLAP queries, consolidation, multi-dimensional view
- Different data and functions:
  - Data warehouses are structured for analysis, with standardized schemas and consolidated information from diverse sources.
  - Data warehouses support complex analytics on historical data. Operational databases handle frequent transactions and updates.
- *Some systems perform OLAP directly on DBs, but performance and scalability may be limited.*

From Tables and Spreadsheets to Data Cubes

- A data warehouse is based on a multi-dimensional data mode, which *views data* in the form of a data cube, defined by:
  - **Dimension tables**: to describe a dimension, e.g., item (*item_name*, *brand*, *type*), or time (*day*, *week*, *month*, *quarter*, *year*)
  - **Fact table**: to store numeric measures (e.g., *dollars_sold*) and keys linking to dimension tables – analyze relationships between dimensions
- Data cube is typically n-dimensional.
  - The n-dimensional base cube is called a **base cuboid**.
  - The topmost 0-dimensional cuboid, which provides the highest-level summarization, is called the **apex cuboid**.
  - All levels of cuboids form the entire data cube.

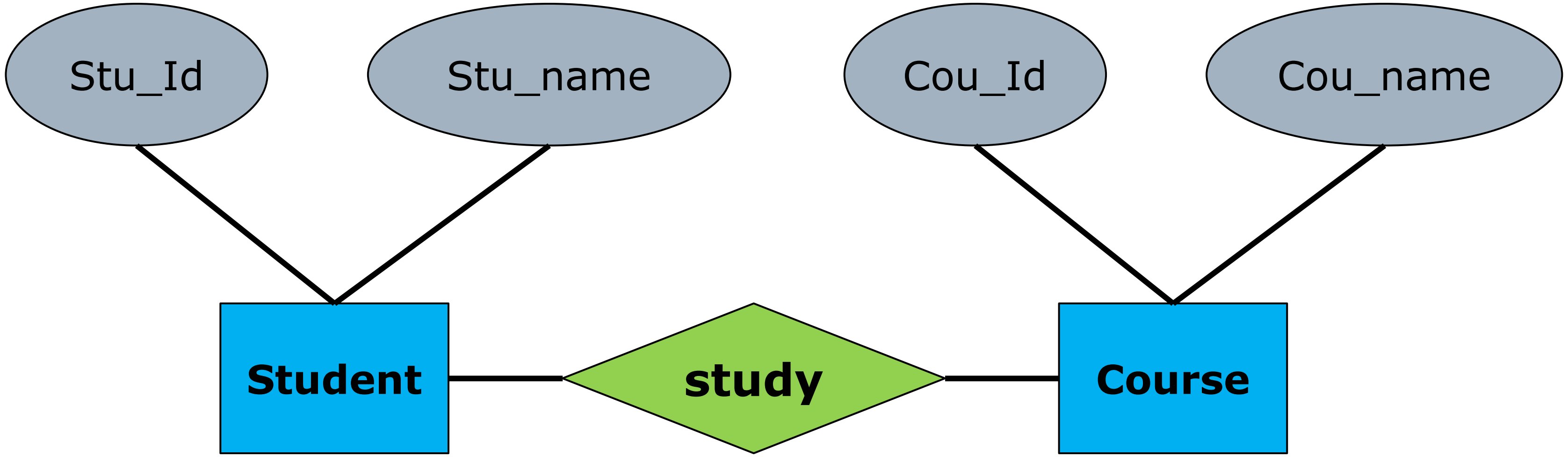
Example: Structure of Data Cube



Lattice of cuboids, making up a 4-D data cube for *time*, *item*, *location*, and *supplier*. Each cuboid represents a different degree of summarization.

Schemas for Multi-dimensional Data Models

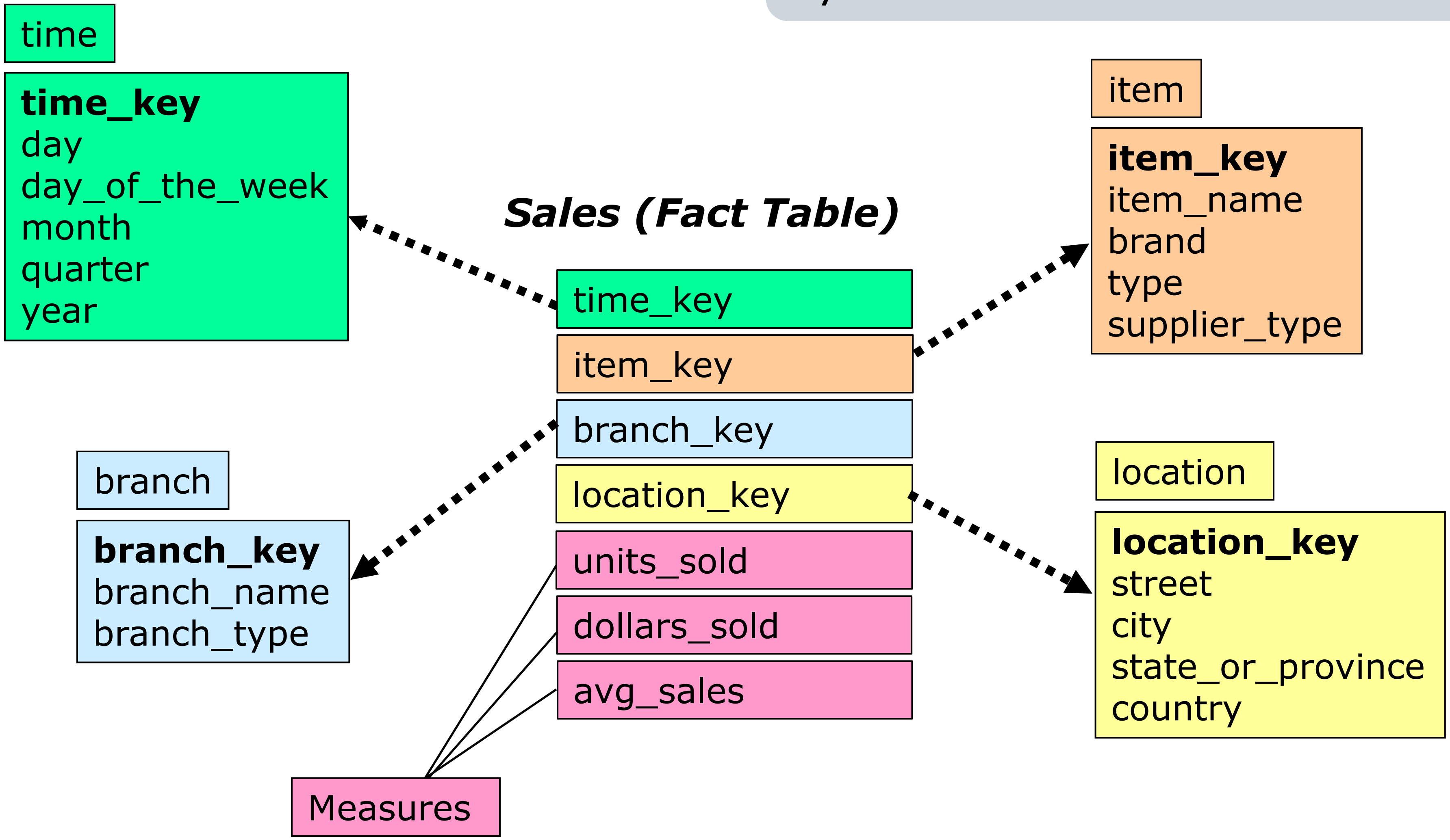
- Entity-Relationship (ER) model and the schema
  - a set of entities and their relationships – *appropriate for OLTP*



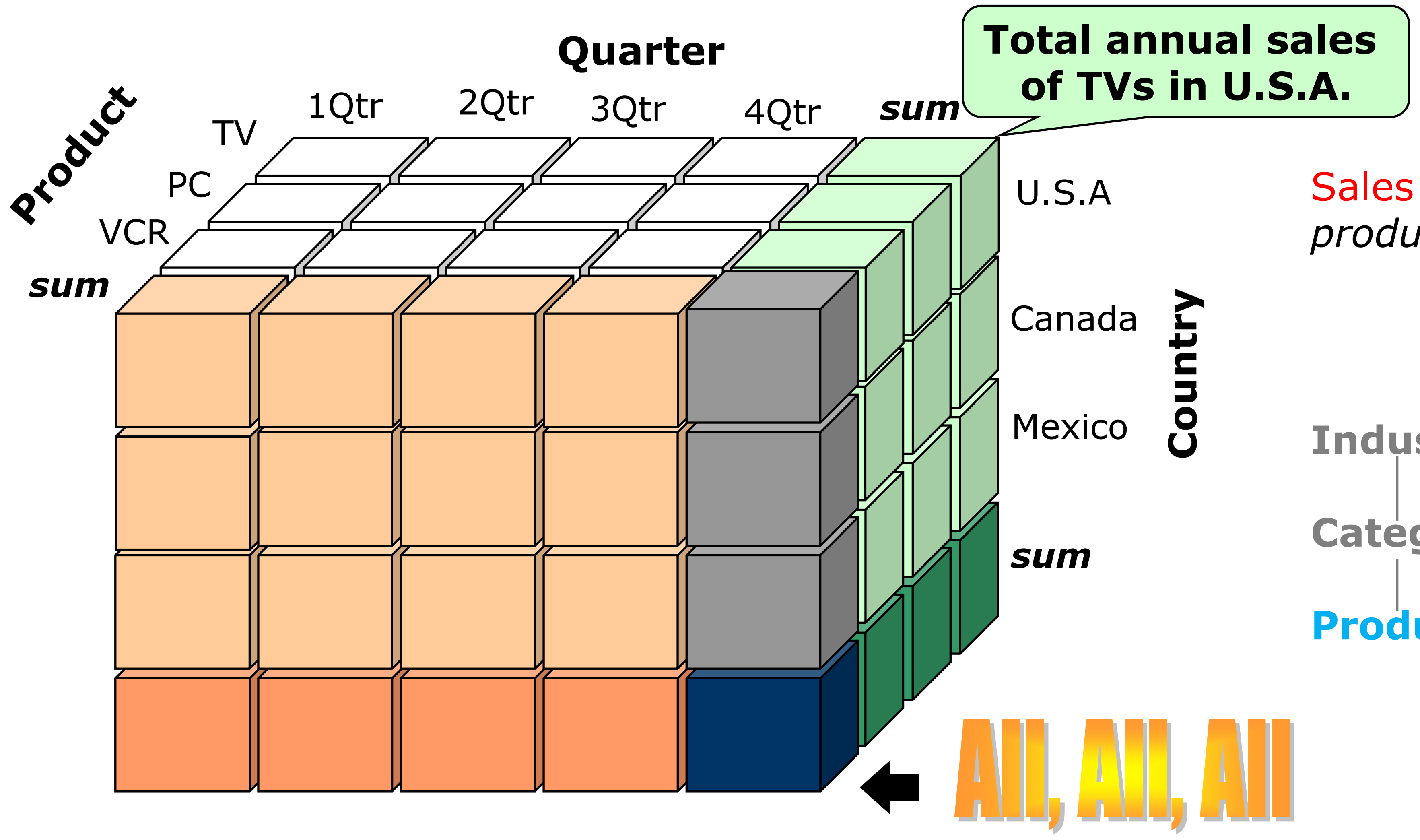
- A multi-dimensional model for data warehouses: focus on dimensions and measures, in the form of:
  - star schema, snowflake schema, fact constellation schema

(1) Star Schema

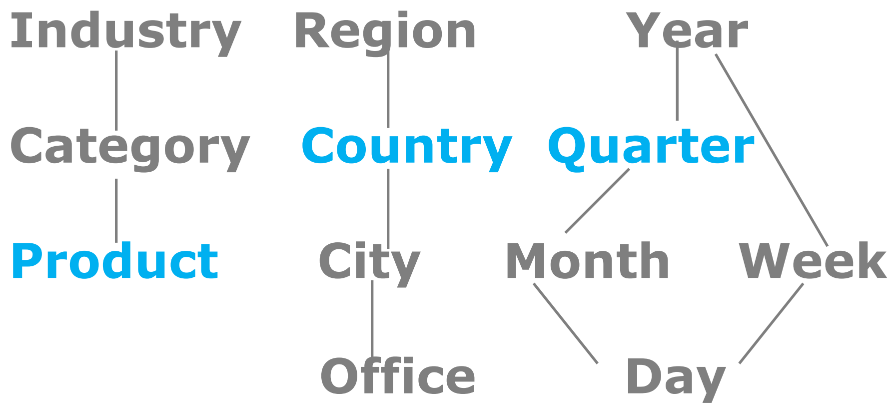
A fact table in the center, surrounded by a set of dimension tables



A Sample Data Cube



Sales volume as a function of *product*, *quarter*, and *country*

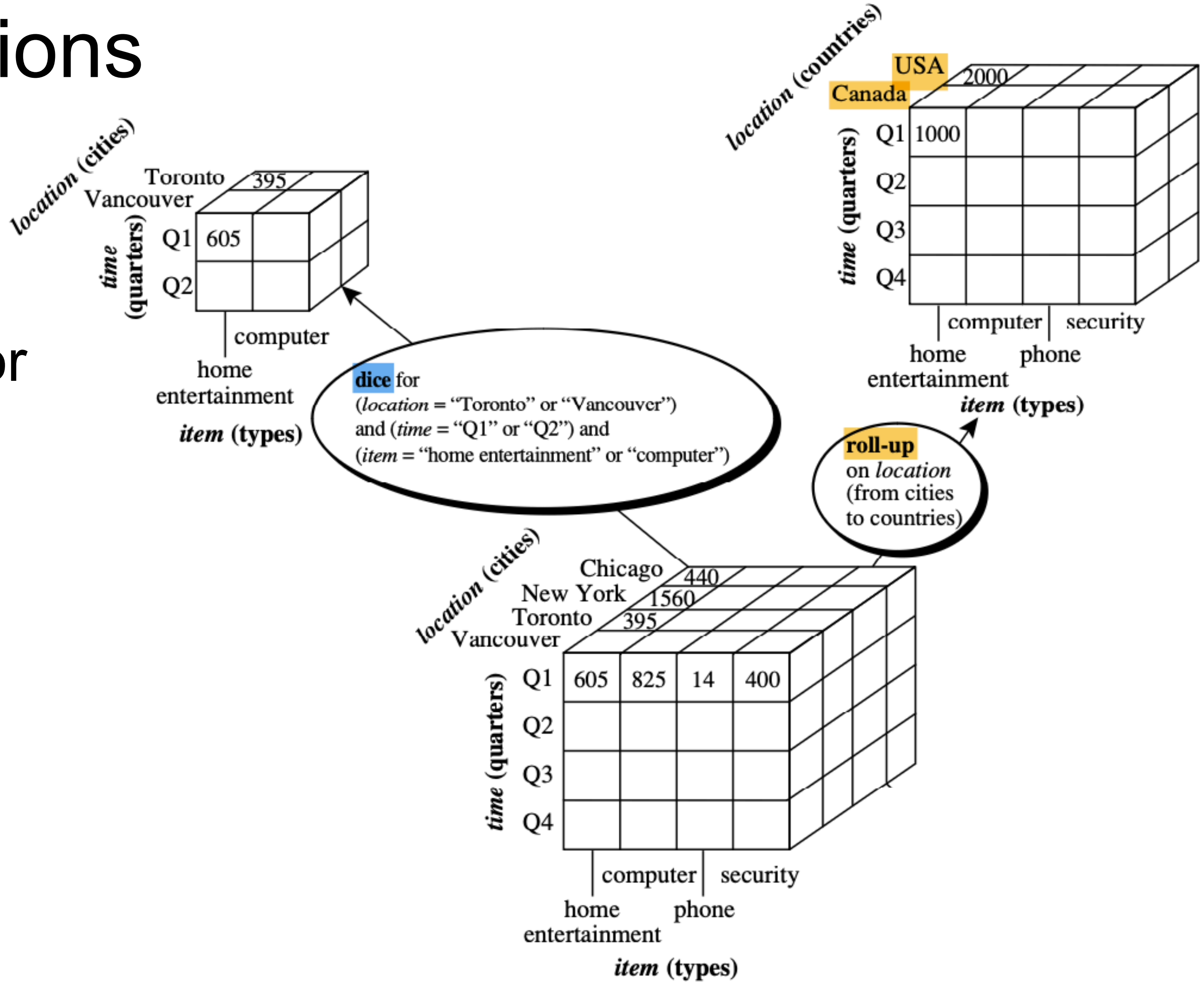


Typical OLAP Operations

- Roll up (drill-up): summarize data by climbing up hierarchy or by dimension reduction techniques
- Drill down (roll-down): reverse of roll-up
  - from higher-level summary to lower-level summary or detailed data, or introducing new dimensions
- Slice and dice: project and select
- Pivot (rotate): reorient the cube, visualization, 3D to series of 2D planes
- Other operations:
  - Drill-across: involving (across) more than one fact table
  - Drill-through: through the bottom level of the cube to its back-end relational tables (using SQL)

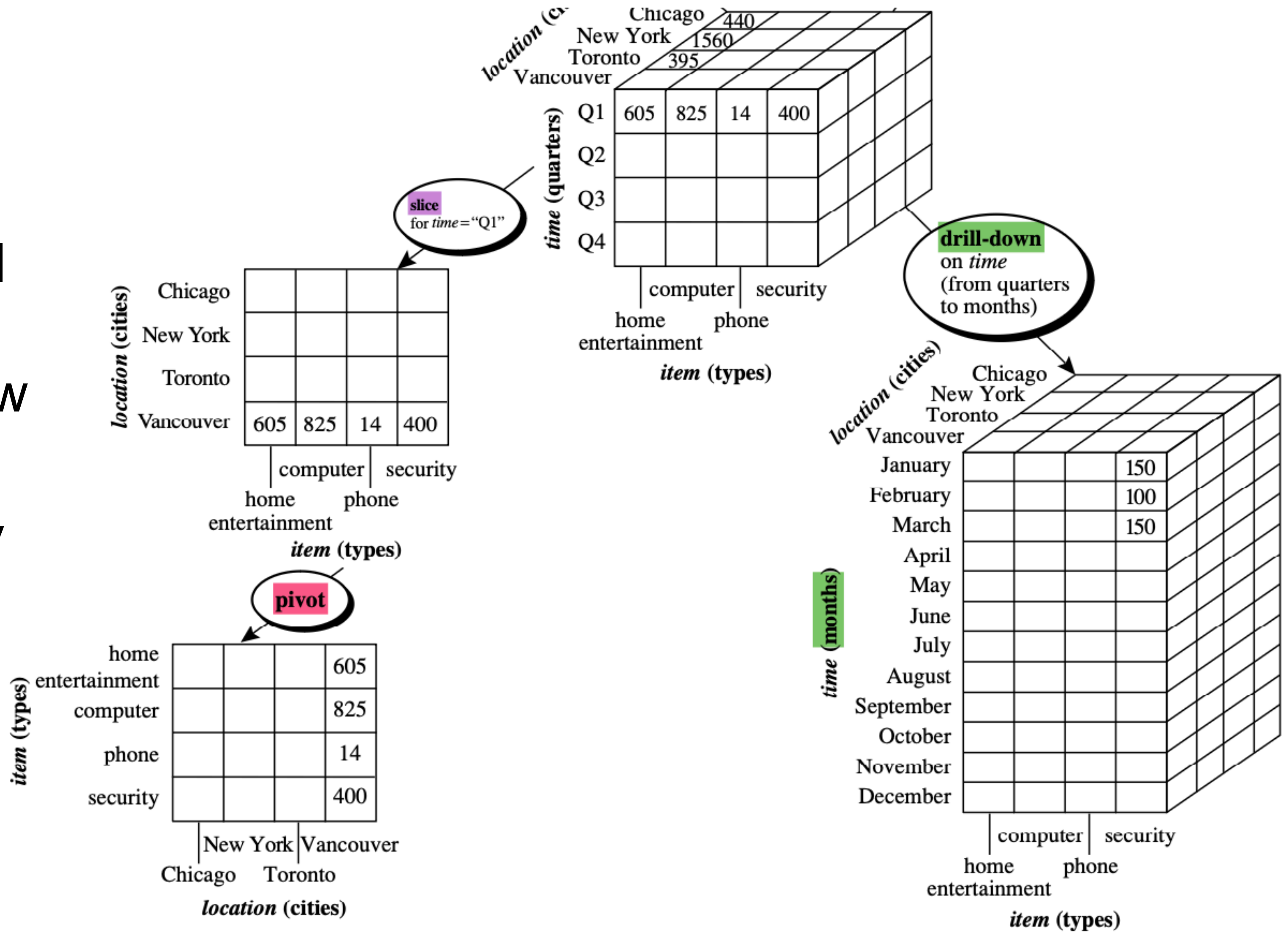
Typical OLAP Operations

- Roll up (drill-up)
  - **summarize data** by climbing up hierarchy for a dimension or by dimension reduction
- Dice
  - define a subcube by performing a **selection** on two or more dimensions

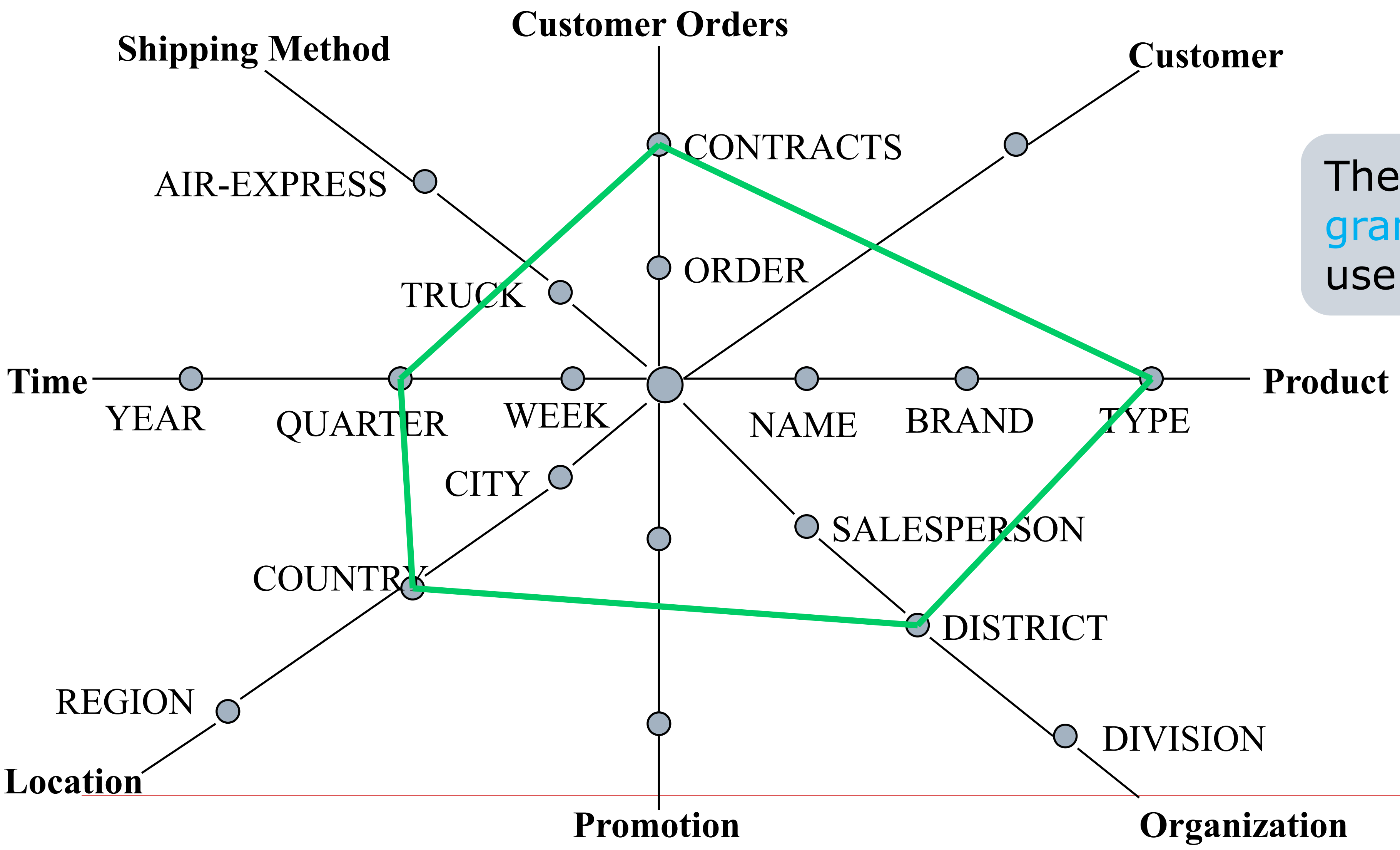


Typical OLAP Operations

- Roll-down (drill-down): reverse of roll-up
  - from higher-level summary to lower-level summary or detailed data, or introducing new dimensions
- Slice: define a subcube by performing a **selection** on one dimension
- Pivot (rotate): reorient the cube, visualization, 3D to series of 2D planes



A Star-Net Query Model

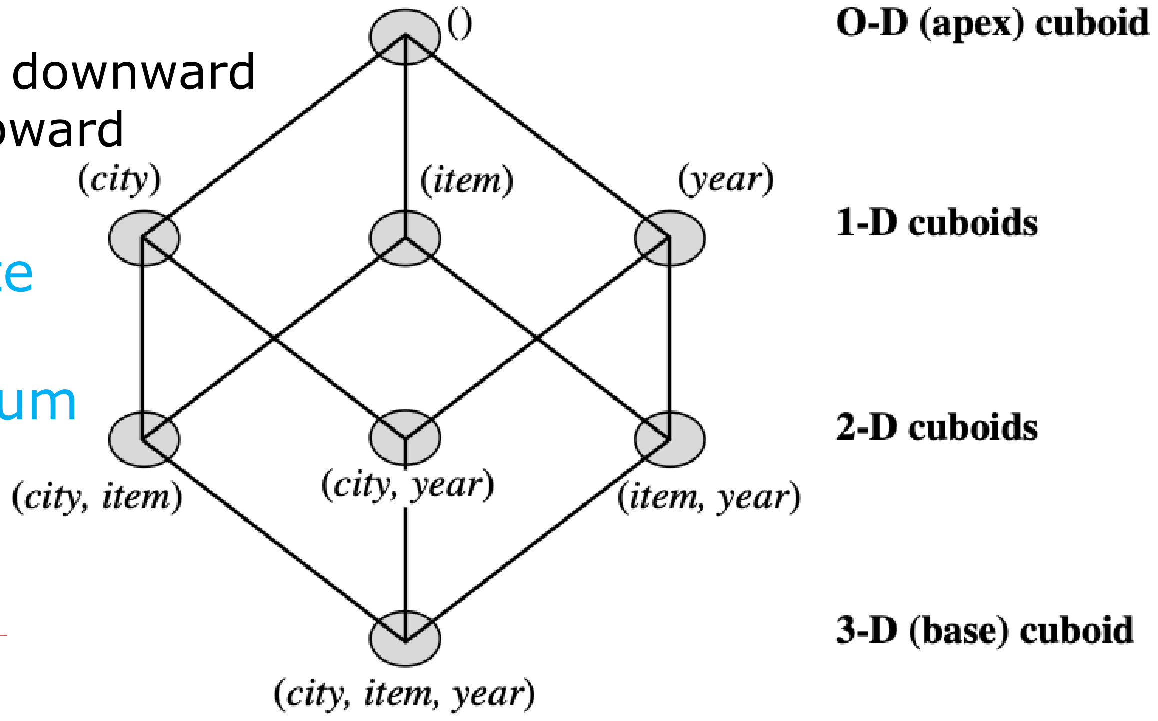


These represent the granularities available for use by OLAP operations.

Efficient Data Cube Computation

- Data cube can be viewed as a lattice of cuboids
  - The bottom-most cuboid is the **base** cuboid – the most specific
  - The top-most cuboid (**apex**) contains only one cell – the most generalized (all)

- **Drilling down**: start from apex cuboid and explore downward
- **Rolling up**: start at the base cuboid and explore upward



- **0-D op**: i.e., no group-by SQL, like "compute the sum of total sales"
- **1-D op**: one group-by, e.g., "compute the sum of sales, group-by city"
- …
- The cube operator is the n-dimensional generalization of the **group-by** operator.

Apriori algorithm, support, confidence, …

FREQUENT ITEMSETS & ASSOCIATION RULE MINING

Basic Concepts: Frequent Itemsets

- **Itemset**: A set of one or more items
  - ***k*-itemset**: $X = \{x_1, \dots, x_k\}$ with k items
- Support of an itemset
  - **Absolute Support (Count)**: the number of transactions containing the given itemset X
  - **Relative Support**: the fraction of transactions containing X (i.e., the probability that a transaction contains X)

- **Frequent Itemset:** An itemset X is *frequent* if the support of X is no less than σ – a *minsup* threshold.

The Apriori Algorithm: Framework

- Outline of **Apriori**: level-wise, candidate generation and test

> - Initially, scan DB once to get frequent 1-itemset
> - **Repeat**
>   - Generate length-$(k+1)$ candidate itemsets based on frequent k-itemsets
>   - Test the candidates against DB to find frequent $(k+1)$-itemsets
>   - Set $k := k + 1$
> - **Until** no frequent or candidate set can be generated
> - Return all the frequent itemsets derived

From Frequent Itemsets to Association Rules

- **Association Rules** written as X → Y [support, confidence]
  - Both X and Y are non-empty itemsets, and $X \cap Y = \emptyset$.
  - It describes an '*if-then*' relationship between two sets of items.

  - **Support**: The percentage of transactions containing both X and Y

$$\sup(X \rightarrow Y) = \mathrm{P}(X \cup Y)$$

    - $P(X \cup Y)$: the percentage of transactions that contains every item in X and Y, i.e., how frequently both X and Y appear together in the dataset
  - **Confidence**: The conditional probability that a transaction having X also contains Y, that is,

$$\mathrm{conf}(X \rightarrow Y) = \mathrm{P}(Y|X) = \sup(X \rightarrow Y)/\sup(X)$$

Final Step: Rule Generation via Frequent Itemsets

- Support (*min-sup*): used to mine the frequent itemsets
- Confidence (*min-conf*): used by the rule generation step to qualify the strength of the derived association rules
  - For each frequent itemset F, generate F's all non-empty subsets
  - For every non-empty subset s, generate a rule:

$$R: s \rightarrow (F - s)$$

  - If the rule R satisfies the minimum confidence, i.e.,

$$\mathrm{conf}(s \rightarrow F - s) = \frac{\sup(F)}{\sup(s)} \geq min\_conf$$

  then R is a strong association rule and should be output.

Limitation of the **Support-Confidence** Framework

- Strong rules are not necessarily interesting: "$A \rightarrow B$" $[s, c]$
- **Example**: Suppose a school may have the following statistics on # students related to playing basketball and/or eating cereal:

	Play basketball	**Not play basketball**	***sum***
Eat cereal	400	350	750
Not eat cereal	200	50	250
sum	600	400	1000 (TOTAL)

  - **Association rule mining** may generate a rule:

    play-basketball → *eat-cereal* [40%, 66.7%]

  - But this strong association rule is misleading → The overall % of students eating cereal is 75% > 66.7%.
  - **A more telling rule**:

    not play-basketball → *eat-cereal* [35%, 87.5%] (high s & c)

Interestingness Measure: **Lift**

- Measure of dependent / correlated events:

$$\boldsymbol{lift(B,C)} = \frac{\mathrm{P}(B \cup C)}{P(B)P(C)} = \frac{\sup(B \rightarrow C)}{\sup(B)\sup(C)} = \frac{\mathrm{conf}(B \rightarrow C)}{\sup(C)}$$

- Tell how B and C are correlated
  - $lift(B,C) = 1$: B and C are independent
  - $lift(B,C) > 1$: positively correlated
  - $lift(B,C) < 1$: negatively correlated

	B	**Not B**	***sum***
C	400	350	750
Not C	200	50	250
sum	600	400	1000

lift is more telling than **s & c**

- Example:

$$lift(B,C) = \frac{400/1000}{600/1000 \times 750/1000} = 0.89 \quad lift(B,\neg C) = \frac{200/1000}{600/1000 \times 250/1000} = 1.33$$

  - Thus, B and C are negatively correlated since $lift(B,C) < 1$.
  - B and $\neg C$ are positively correlated since $lift(B,\neg C) > 1$.

Interestingness Measure: χ^2

- To test correlated events: $\chi^2 = \frac{\sum(Observed - Expected)^2}{Expected}$

> - $\chi^2 = 0$: independent
> - $\chi^2 > 0$: correlated, either positive or negative → needs additional test

	B	Not B	*sum*
C	400 (450)	350 (300)	750
Not C	200 (150)	50 (100)	250
sum	600	400	1000

Expected value

Observed value

$$\chi^2 = \frac{(400-450)^2}{450} + \frac{(350-300)^2}{300} + \frac{(200-150)^2}{150} + \frac{(50-100)^2}{100} = 55.56$$

- Thus, B and C are negatively correlated since the expected value is 450 but the observed is only 400.
- χ^2 is also more telling than the support-confidence framework

Lift and χ^2: *Are They Always Good Measures?*

- Null transactions: Transactions that contain **neither B nor C**

Examine the dataset:

- BC $(100, 0.1\%)$ is much rarer than $B\neg C$ (1000) and $\neg BC$ (1000)
- There are many $\neg B\neg C$ $(100000, 98\%)$.
- Unlikely B & C will happen together!

	B	¬B	Σ_{row}
C	100	1000	1100
¬C	1000	100000	101000
$\Sigma_{col.}$	1100	101000	102100

null transactions

- However, B and C seem to be strongly positively correlated based on:
  - $lift(B, C) = 8.44 \gg 1$
  - $\chi^2(B, C) = 670$ and Observed (100) >> Expected (11.85)
- Too many null transactions may "spoil the soup"!

Contingency table with expected values added

	B	¬B	Σ_{row}
C	100 (11.85)	1000	1100
¬C	1000 (988.15)	100000	101000
$\Sigma_{col.}$	1100	101000	102100

Interestingness Measures: Null-Invariant

- Null invariance: value does not change with # null-transactions
  - χ^2 and $lift$ are NOT null-invariant with the range of $[0, \infty]$.
- Null-invariant Measures:
  - **All Confidence**: the minimum confidence of the two association rules related to A and B, namely, "A → B" and "B → A"

$$all\_conf(A,B) = \frac{sup(A \cup B)}{max\{sup(A), sup(B)\}} = min\{P(A|B), P(B|A)\}$$

$$max\_conf(A, B) = max\{P(A|B), P(B|A)\}$$

  - **Max Confidence**: the maximum confidence of the two rules
  - **Kulczynski** (*Kulc*): an average of two confidence values
  - **Cosine**: a harmonized *lift* measure (unaffected by # total transactions)

$$Kulc(A, B) = \frac{1}{2}(P(A|B) + P(B|A))$$

$$cosine(A, B) = \frac{P(A \cup B)}{\sqrt{P(A) \times P(B)}} = \frac{sup(A \cup B)}{\sqrt{sup(A) \times sup(B)}} = \sqrt{P(A|B) \times P(B|A)}.$$

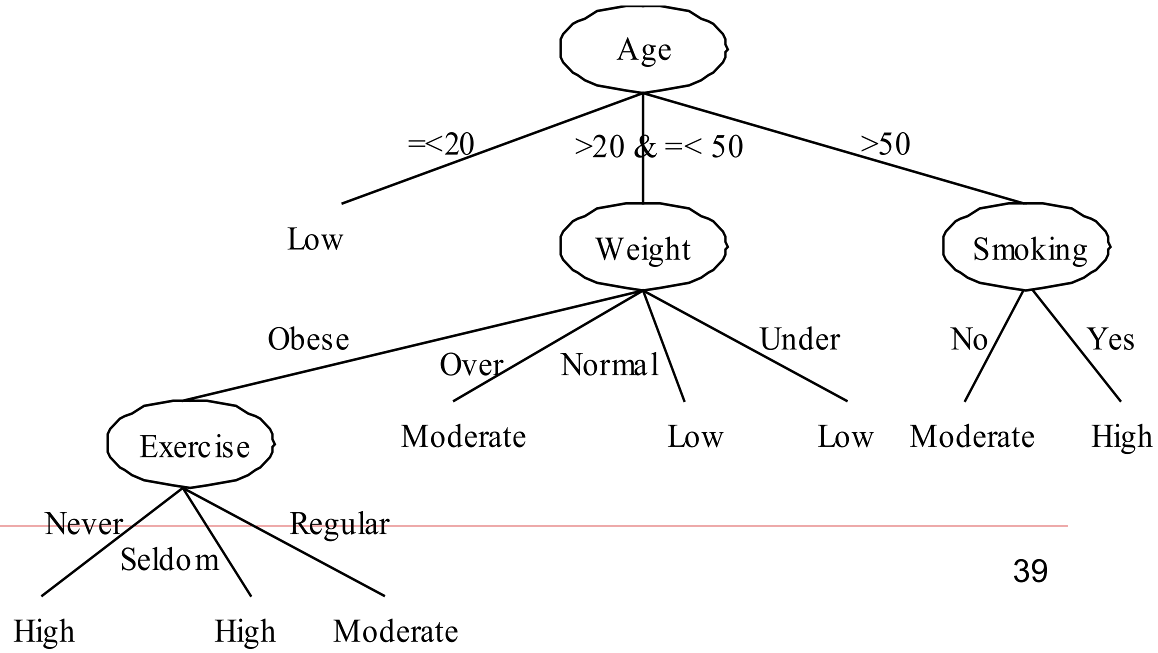
Decision tree: ID3 algorithm driven by entropy and information gain

CLASSIFICATION

Decision Tree Structure

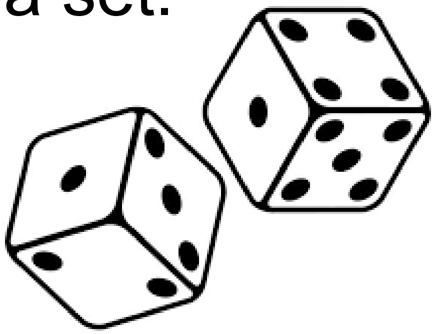
- A flow-chart-like structure used for classification
  - **Internal node**: a test on an attribute (e.g., age, exercise, weight, smoking)
  - **Branch**: an outcome of the test
  - **Leaf nodes**: class labels (e.g., high-, moderate-, and low-risk)

How it works:
An object is classified by traversing the tree from its root to a leaf.



Entropy

- A measure of randomness, uncertainty, and disorder in a system with probability distributions of outcome.
- Entropy is formulated as a *function* that measures disorder.
  - *“The higher the entropy, the greater the disorder.”*
  - For classification, it tells how diverse the classes are in a set.
- Let $\boldsymbol{D}$ be a set of examples from $\boldsymbol{m}$ classes.



$$Info(D) = -\sum_{i=1}^{m} p_i \cdot \log_2(p_i)$$

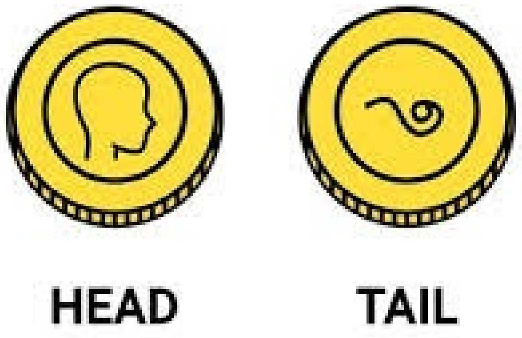
- **Input**: Distribution of outcomes
- **Output** : A value indicating how disordered the outcomes are
- p_i: The proportion of examples observed in D that belong to i-th class within [0,1].

Example: Tossing Coins in Casino

- **Casino A** with real coins (50/50 chances):

$$Info(Coin\ Toss) = -p(head)\log_2 p(head) - p(tail)\log_2 p(tail)$$

$$= -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1$$



- **Casino B** with fake coins (75/25 chances):

$$Info(Coin\ Toss) = -p(head)\log_2 p(head) - p(tail)\log_2 p(tail)$$

$$= -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) = 0.811$$

Entropy is a measure of randomness and disorder.
Higher entropy means higher uncertainty.

Information Gain and Iterative Dichotomiser (ID3)

- **Classification Goal**: To split the dataset in a way that reduces entropy the most.
- **Information Gain:** To measure the reduction in entropy after splitting the dataset on an attribute A

$$Gain(D, A) = Info(D) - Info_A(D)$$

  - Weighted entropy after split: $Info_A(D) = \sum_{j=1}^{n} p(D_j|A) Info(D_j)$
    - D_j: subsets of D created by splitting on A

ID3 Algorithm: Repeatedly selects the attribute with the highest information gain at each step to build the decision tree.

ID3 Example (Decision: buy computer or not)

- $\mathbf{Info}(D) = \sum -p_i \times \log_2 p_i$
- $\mathbf{Info}_A(D) = \sum[p(D_j|A) \times \mathbf{Info}(D_j)]$
- $\mathbf{Gain}(D, A) = \mathbf{Info}(D) - \mathbf{Info}_A(D)$

- **Class P**: buys_computer = 'yes' → 9
- **Class N**: buys_computer = 'no' → 5

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

$\boxed{\frac{5}{14}I(2,3)}$ means 'age <=30' has 5 out of 14 samples, with 2 'yes' and 3 'no'.

Hence,

$$\boxed{Gain(age) = Info(D) - Info_{age}(D) = 0.246}$$

Similarly, $Gain(income) = 0.029$

$Gain(student) = 0.151$

$Gain(credit\_rating) = 0.048$

Bayesian Theorem

- $P(H|E)$: Posterior probability, the probability of H holds given E
  - E: Evidences (e.g., a data tuple) with attribute description
  - H: Hypothesis to be verified (e.g., a class label that E belongs to)

$$\boldsymbol{P(H|E) = \frac{P(E|H)\,P(H)}{P(E)}}$$

- $P(H)$: prior probability, i.e., the initial probability of hypothesis H before observing evidence E
- $P(E)$: marginal probability, i.e., the total probability of observing evidence E under all possible hypotheses
- $P(E|H)$: likelihood, i.e., the probability of observing evidence E given that the hypothesis $H = true$

Bayesian Classification

- A data tuple: $X = (A_1 = x_1, A_2 = x_2, A_3 = x_3, \ldots, A_n = x_n)$
- To classify X, we need to estimate $P(C_i \mid X)$
  - C_i represents the **hypothesis** that X belongs to C_i.
  - We say X belongs to C_i iff: $P(C_i|X) > P(C_j|X), for\ all\ j \neq i$

- How to estimate $P(C_i \mid X)$ for classifying X?
  - **Bayesian theorem**: $P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$
  - The problem becomes → estimating $P(X|C_i)$ and $P(C_i)$

Bayesian Classification

- Estimate the priori probability of the i-th class C_i from the training set D: $P(C_i) = \frac{|C_i|}{|D|}$

- Independence Assumption: For $P(X \mid C_i)$, we assume that the effect of each attribute A_j is independent to others:

$$P(X = (A_1 = x_1, A_2 = x_2, \dots, A_n = x_n) \mid C_i)$$
$$= P(A_1 = x_1 | C_i) \times P(A_2 = x_2 | C_i) \times \cdots$$
$$\times P(A_n = x_n | C_i)$$

where $P\left(A_j = x_j \middle| C_i\right)$ can also be estimated from the training set D.

Example

Bayesian: $P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$

- Given a training set, predict if a person X will buy a computer
  - $\boldsymbol{X}$: {*age* = youth, *income* = medium, *student* = yes, *credit_rating* = fair}
  - Yes or No? $P(buy\_computer|X)$

Priori Probability in training Data:

- $P(buy\_computer = yes) = 9/14 = 0.643$
- $P(buy\_computer = no) = 5/14 = 0.357$

age	*buys_computer* yes	no
youth	2	3
middle_aged	4	0
senior	3	2

income	*buys_computer* yes	no
low	3	1
medium	4	2
high	2	2

student	*buys_computer* yes	no
yes	6	1
no	3	4

credit_ratting	*buys_computer* yes	no
fair	6	2
excellent	3	3

To calculate $\boldsymbol{P(X \mid buy\_computer = yes)}$:

- $P(age = youth \mid yes) = 2/9 = 0.222$
- $P(income = medium \mid yes) = 4/9 = 0.444$
- $P(student = yes \mid yes) = 6/9 = 0.667$
- $P(credit\_rating = fair \mid yes) = 6/9 = 0.667$

→ $P(X \mid buy\_computer = yes) = 0.044$

→ Similarly, $P(X \mid buy\_computer = no) = 0.019$

Through Bayesian:

- $P(X \mid yes) \times P(buy\_computer = yes) = 0.028$
- $P(X \mid no) \times P(buy\_computer = no) = 0.007$

Conclusion: X will buy a computer.

Evaluation Measures

- To assess how "accurate" your classifier is at predicting the class label of tuples compared to actual labels
  - **True Positives TP**: positive tuples that were correctly labeled
    - Positive tuples: tuples of the main class of interest
  - **True Negatives TN**: negative tuples that were correctly labeled
  - **False Positives FP**: negative tuples that were incorrectly labeled as positive (e.g., people who do not buy computers but are labeled as $buys\_computer = yes$)
  - **False Negatives FN**: positive tuples that were mislabeled as negative (e.g., people who really buy computers but are labeled as $buys\_computer = no$)

Predicted class

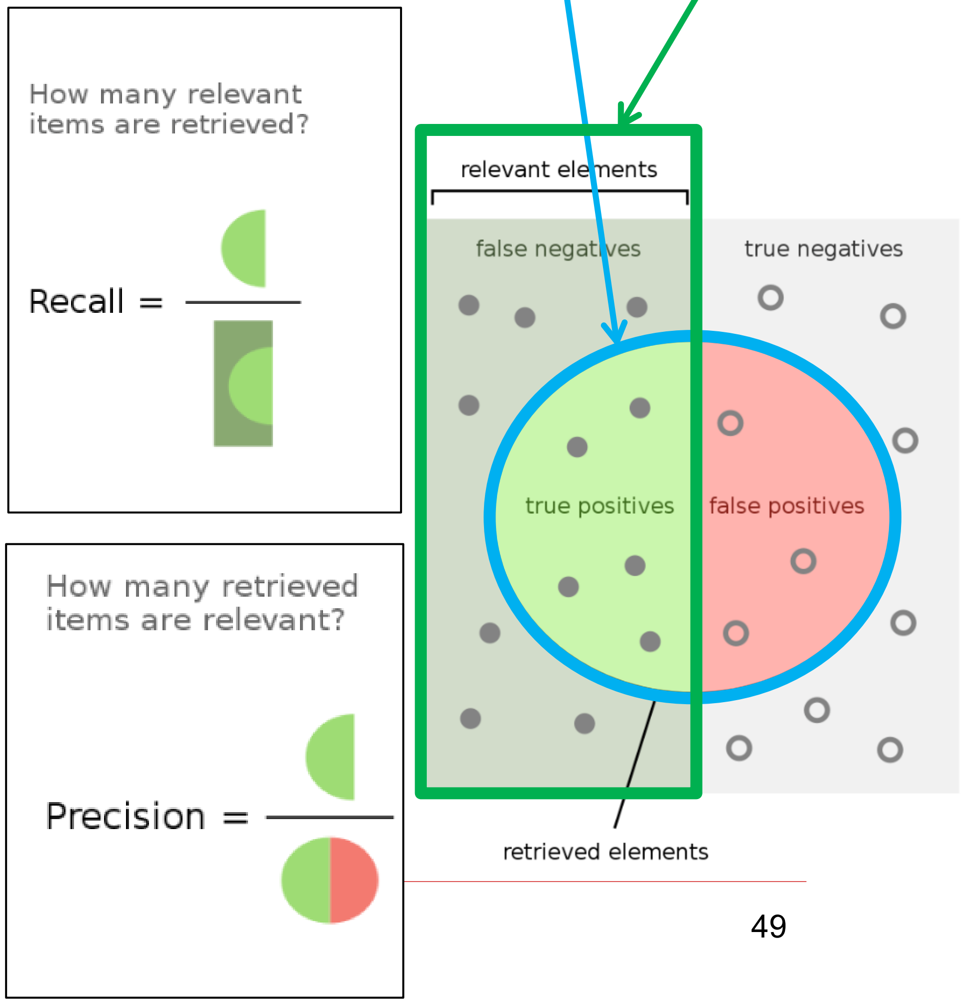
Actual class		*yes*	*no*	Total
	yes	TP	FN	P
	no	FP	TN	N
	Total	P'	N'	$P+N$

Evaluation Measures

Measure	Formula
accuracy, recognition rate	$\frac{TP+TN}{P+N}$
error rate, misclassification rate	$\frac{FP+FN}{P+N}$
sensitivity, true positive rate, recall	$\frac{TP}{P}$
specificity, true negative rate	$\frac{TN}{N}$
precision	$\frac{TP}{TP+FP}$
F, F_1, F-score, harmonic mean of precision and recall	$\frac{2 \times precision \times recall}{precision + recall}$
F_β, where β is a non-negative real number	$\frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$

Predicted class

Actual class	*yes*	*no*	Total
yes	TP	FN	P
no	FP	TN	N
Total	P'	N'	$P+N$



CLUSTERING

Partitioning Algorithms: Basic Concepts

- **Partitioning method**
  - Discover groupings in the data by optimizing a specific objective function and iteratively improving the quality of partitions
- **K-partitioning method**
  - Objective: Divide a dataset D of n objects into a set of K clusters, so that an objective function is optimized (e.g., minimizing the sum of distances within clusters)
  - Typical objective function: **Sum of Squared Errors (SSE)**

$$SSE(C) = \sum_{k=1}^{K} \sum_{x_{i \in C_k}} \| x_i - c_k \|^2$$

where c_k is the centroid or medoid of cluster C_k

The K-Means Clustering Algorithm

- **Idea**: each cluster is represented by the centroid, which is the mean position of all data points in the cluster
  - *It may not correspond to an actual data point in the dataset!*

- Given K, the number of clusters, the K-Means clustering algorithm is outlined as follows:

Initialization: Select K data points as initial centroids
Repeat
- Form K clusters by assigning each point to its closest centroid
- Re-compute the centroids (i.e., mean point) of each cluster

Until centroids no longer change or convergence criterion is met

Discussion on K-Means Clustering (I)

- **Limitations**
  - Need to specify K in advance
    - There are ways to automatically determine the '*best*' K.
    - In practice, one often runs a range of values and selected the '*best*'.
  - Only for objects in a continuous data space: K-modes for nominal data
  - K-means clustering often terminates at a local optimum.
    - Poor initialization can lead to suboptimal clusters.
  - Sensitive to noisy data and outliers (extreme values)

Measuring Clustering Quality

- **Evaluation**: Evaluating the goodness of clustering results
  - No universally recognized 'best' measure in practice!
- **Three categorization of measures**
  - **Internal**: Unsupervised, criteria derived from data itself
    - How well the clusters are separated and how compact the clusters are
  - **External**: Supervised, employ criteria not inherent to the dataset
    - Compare a clustering against prior or expert-specified knowledge (i.e., the ground truth) using certain clustering quality measures
  - **Relative**: Directly compare different clustering, usually those obtained by varying parameters for the same algorithm

Email: fengmei.jin@polyu.edu.hk

Office: PQ747

THANK YOU!