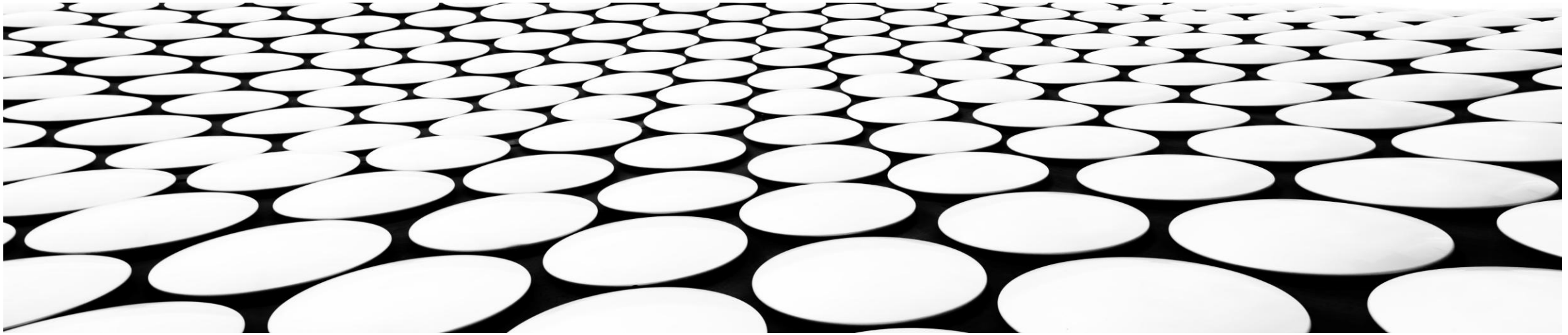
DATA MINING AND PREDICTIVE DATA ANALYSIS

CHAPTER-2

DATA PREPROCESSING



INTRODUCTION TO DATA PREPROCESSING

- **Data in the real world are - Noisy, Incomplete and Inconsistent**
  - Due to their typically huge size (often several gigabytes or more).
  - As their likely origin from multiple, heterogeneous sources
  - Historical/past un-attended data- often contains field values that have expired, are no longer relevant, or are simply missing.
- Real-world (raw data) data is often:
  - **Incomplete:** missing values, lacking attributes, or containing only partial information**.**
  - **Noisy:** containing errors, outliers, or irrelevant information.
  - **Inconsistent:** conflicting values, different units/formats, or duplication**.**
  - **High-dimensional:** too many features, making analysis complex**.**

WHY DATA PREPROCESSING

- **No quality data, no quality mining results!**
  - Quality decisions must be based on quality data.
  - "garbage in, garbage out": if raw data is noisy, incomplete, or inconsistent, the mining results will be poor.

❑ **Data Preprocessing**

- It prepares raw data into a clean, consistent, and suitable format for further analysis.
- Is the first and most critical step in the data mining process.
- Improve the quality of the data and, consequently the quality of the mining results and lead to huge payoffs for decision making.
- Data preprocessing can take 20–60% of the total time and effort in the data mining process based on the size and complexity of the data set.

WHY DATA PREPROCESSING

- **Importance of Data Preprocessing**
  - Ensures better quality results (reliable, efficient, and meaningful mining results)
  - Reduces computational cost and time.
  - Increases accuracy of models (classification, clustering, prediction).
- In order to be useful for data mining purposes, the databases need to undergo preprocessing, in the form of
  - **Data Cleaning,** and
  - **Data Transformation**

DATA CLEANING

- let us take a look at some of the kinds of errors that could creep into even a tiny data set

Customer ID	Zip	Gender	Income	Age	Marital Status	Transaction Amount
1001	10048	M	75,000	C	M	5000
1002	J2S7K7	F	−40,000	40	W	4000
1003	90210		10,000,000	45	S	7000
1004	6269	M	50,000	0	S	1000
1005	55101	F	99,999	30	D	3000

- Without proper cleaning, data mining algorithms may produce misleading patterns- leading to inaccurate, inconsistent, and unreliable outcomes

DATA CLEANING

- Data cleaning (also called data cleansing or data scrubbing) is the process of detecting and correcting (or removing) corrupt, inaccurate, incomplete, or irrelevant records from a dataset.
- Causes of Dirty Data
  - Human errors – Typos, inconsistent spellings, missing entries.
  - Data integration issues – Different formats, units, or coding standards across sources.
  - Noise – Random variations or outliers in data.
  - Missing values – Due to non-response, equipment failure, or improper data entry.
  - Redundancy and duplicates – Same data recorded multiple times.

DATA CLEANING

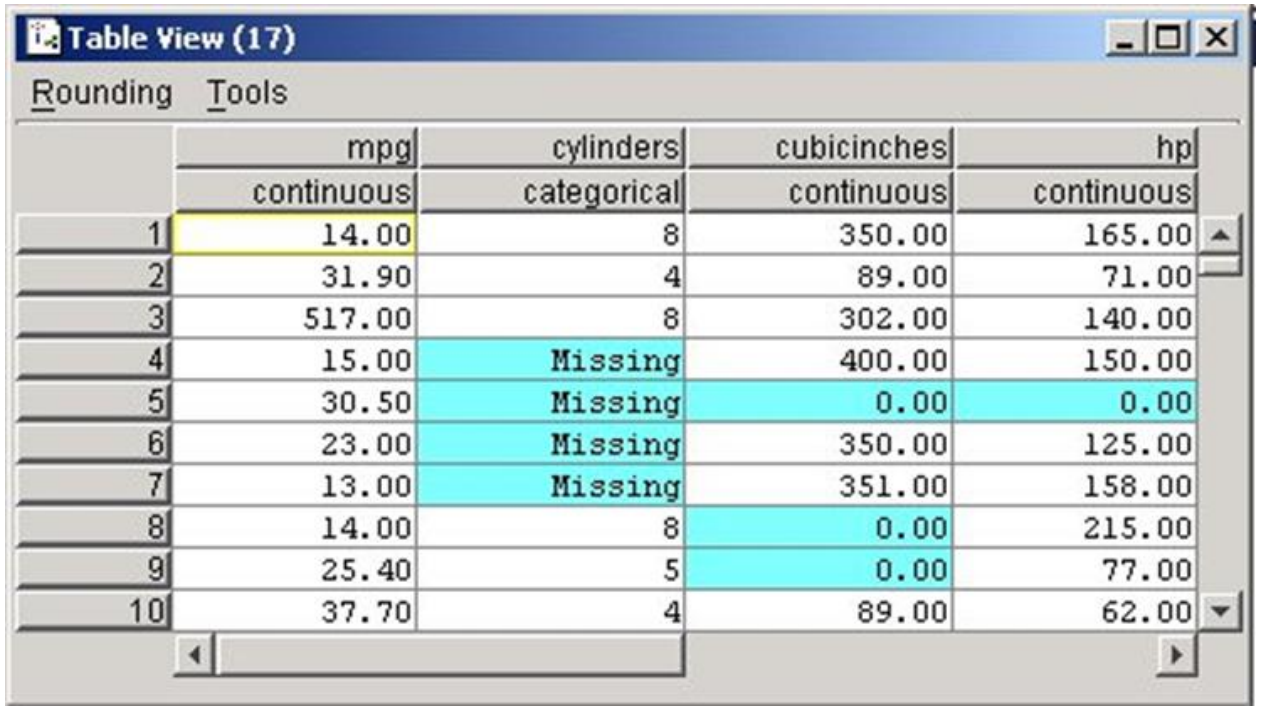
- Types of Data Quality Issues
  - Missing Data: Null values, blanks, unknowns.
  - Noisy Data: Errors, outliers, misrecorded values.
  - Inconsistent Data: Conflicting values (e.g., "Male" vs. "M").
  - Duplicate Data: Repeated records.
  - Irrelevant Data: Attributes not useful for analysis.

HANDLING MISSING DATA

- **Missing Values**: Some tuples may not have recorded value for several attributes.
- A common method of “handling” missing values is simply to omit the records or fields with missing values -- However, this may be dangerous
  - Deleting the records with missing values would lead to a biased subset of the data.
  - Wasteful to omit the information in all other fields, just because one field value is missing.
- This method may be effective, if the tuple contains several attributes with missing values.
- This is usually done when the class label is missing in classification problems.

HANDLING MISSING DATA

- Replace the missing value with some constant, specified by the analyst.
  - Missing numeric values replaced with 0.0
  - Missing categorical values replaced with “Missing”

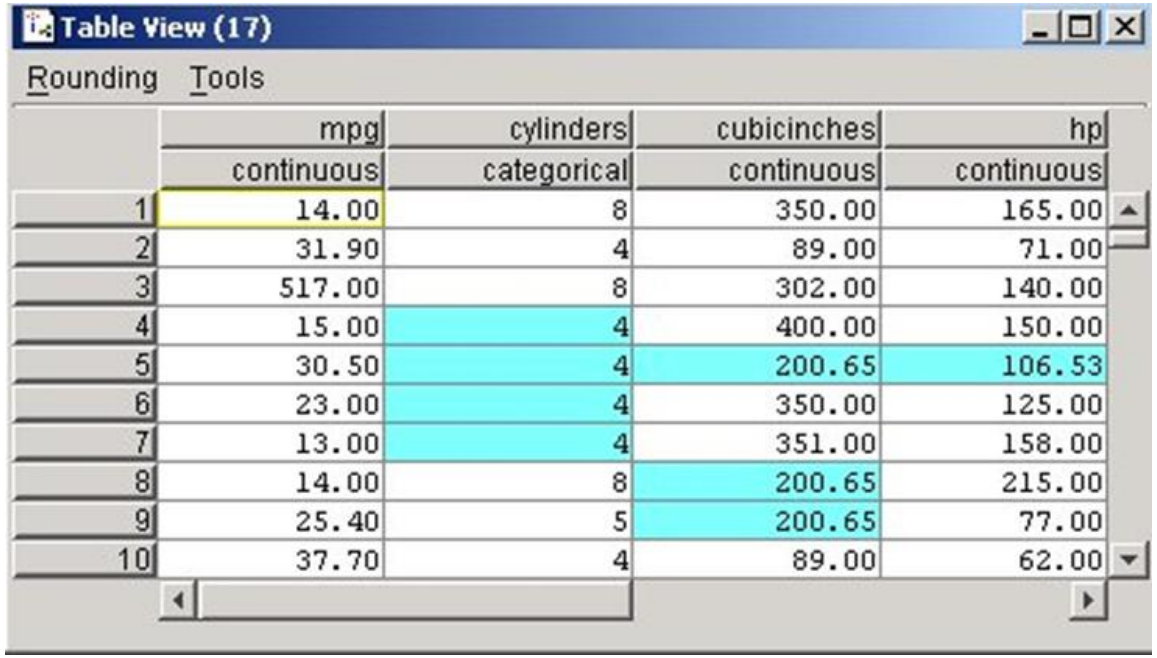


	mpg	cylinders	cubicinches	hp
	continuous	categorical	continuous	continuous
1	14.00	8	350.00	165.00
2	31.90	4	89.00	71.00
3	517.00	8	302.00	140.00
4	15.00	Missing	400.00	150.00
5	30.50	Missing	0.00	0.00
6	23.00	Missing	350.00	125.00
7	13.00	Missing	351.00	158.00
8	14.00	8	0.00	215.00
9	25.40	5	0.00	77.00
10	37.70	4	89.00	62.00

HANDLING MISSING DATA

❑Replace Missing Values with Mode or Mean

- Attribute mean- for numeric variables
- Attribute mode -for categorical variables



	mpg	cylinders	cubicinches	hp
	continuous	categorical	continuous	continuous
1	14.00	8	350.00	165.00
2	31.90	4	89.00	71.00
3	517.00	8	302.00	140.00
4	15.00	4	400.00	150.00
5	30.50	4	200.65	106.53
6	23.00	4	350.00	125.00
7	13.00	4	351.00	158.00
8	14.00	8	200.65	215.00
9	25.40	5	200.65	77.00
10	37.70	4	89.00	62.00

- Mode of categorical field cylinders = 4
- Mean for non-missing values in numeric field cubicinches = 200.65

HANDLING MISSING DATA

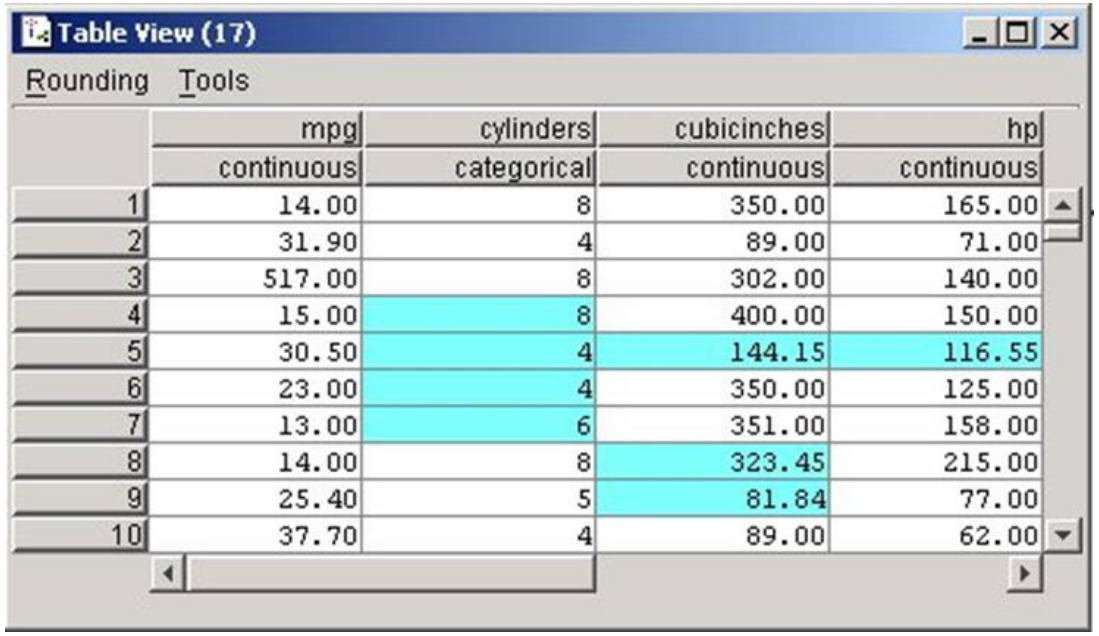
- Substituting mode or mean for missing values sometimes works well.
- Mean not always best choice for “typical” value.
- Resulting confidence levels for statistical inference become overoptimistic.
- Domain experts should be consulted regarding approach to replace missing values.
- Benefits and drawbacks resulting from the replacement of missing values must be carefully evaluated.

HANDLING MISSING DATA

❑Replace Missing Values with Random

- Replace the missing values with a value generated at random from the observed distribution of the variable.
  - Value for cylinders, cubicinches, and hp randomly drawn proportionately from each field's distribution
  - Method superior compared to mean substitution
  - Measures of location and spread remain closer to original

No guarantee resulting records make sense



	mpg	cylinders	cubicinches	hp
	continuous	categorical	continuous	continuous
1	14.00	8	350.00	165.00
2	31.90	4	89.00	71.00
3	517.00	8	302.00	140.00
4	15.00	8	400.00	150.00
5	30.50	4	144.15	116.55
6	23.00	4	350.00	125.00
7	13.00	6	351.00	158.00
8	14.00	8	323.45	215.00
9	25.40	5	81.84	77.00
10	37.70	4	89.00	62.00

HANDLING MISSING DATA

❑ Replace Missing Values using - Data Imputation Method

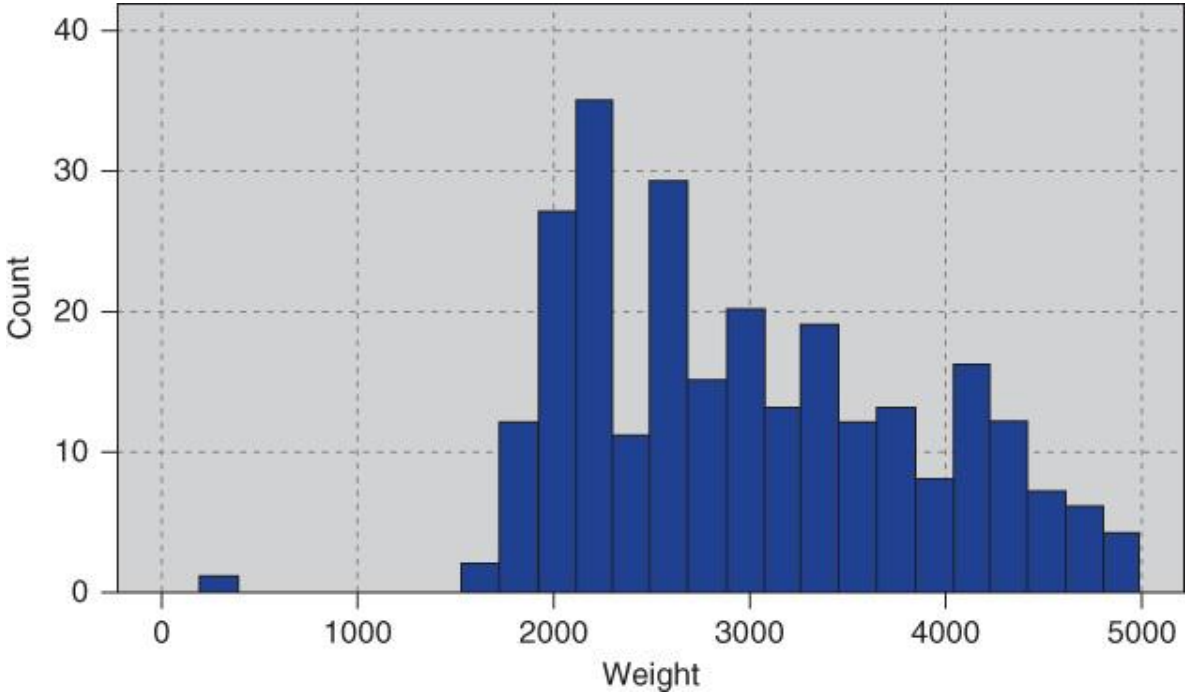
- Replace the missing values with imputed values based on the other characteristics of the record.
- What would be the most likely value for this missing value - given all the other attributes for a particular record
- Example:
  - Regression-based imputation.
  - Using **k-Nearest Neighbors (kNN)** for estimation.
  - inference-based tools using a Bayesian inference, or decision tree

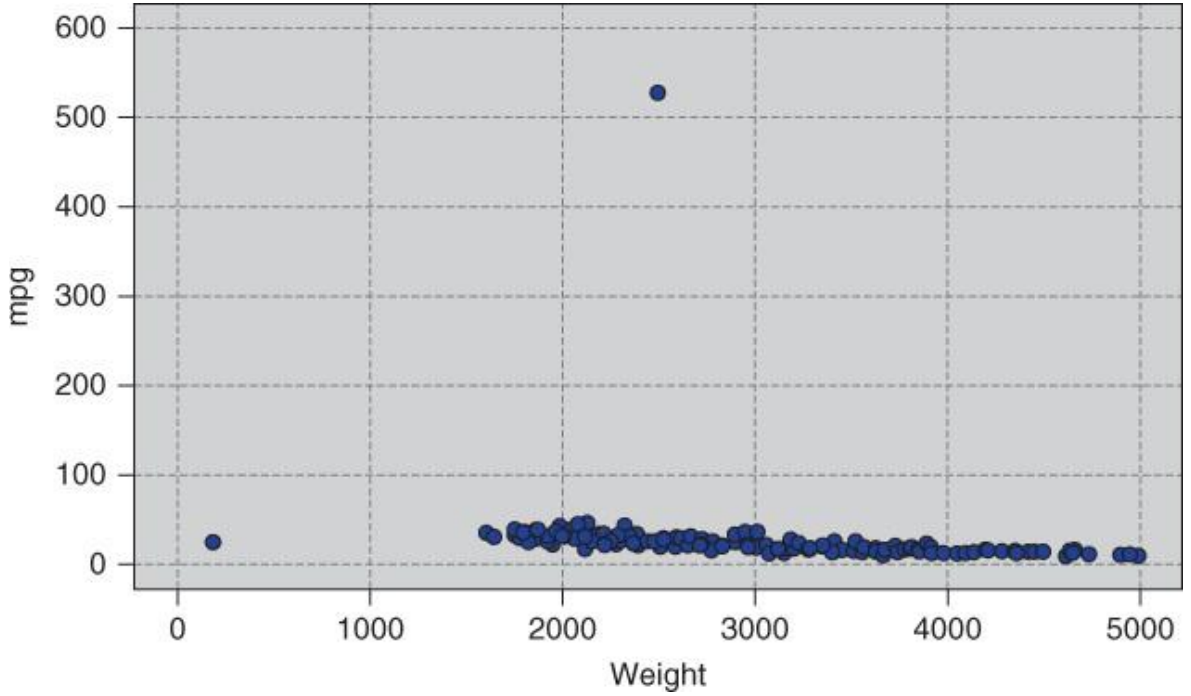
GRAPHICAL METHODS FOR IDENTIFYING OUTLIERS

- **Outlier : A data point that significantly deviates from the overall pattern of the dataset.**
- **Outliers are extreme values that go against the trend of the remaining data.**
- **Outliers may arise due to:**
  - **Errors in data collection or entry.**
  - **Natural variation in data.**
  - **Rare events or anomalies worth studying (Outlier Analysis)**
- **Identifying outliers is important because they may that can distort analysis..**
- **Statistical methods are sensitive to the presence of outliers**

GRAPHICAL METHODS FOR IDENTIFYING OUTLIERS

- Graphical Method For Identifying Outliers
  - **Histogram** - for numeric variables
  - **Scatter Plot** - to reveal outliers in more than one variable





MEASURES OF CENTER AND SPREAD

- **Central Tendency:** Central tendency refers to a single value that attempts to describe a dataset by identifying the central point within that dataset.
- Purpose in Data Mining:
  - Provides quick insight into the distribution of values.
  - Helps detect skewness and outliers.
  - Serves as a baseline for further statistical analysis and preprocessing.
- Most fundamental descriptive measures for “central tendency" of data- **Mean, Median**, and Mode,

MEASURES OF CENTER AND SPREAD

- **Mean (Arithmetic Average):** The sum of all data values divided by the number of values.
  - Example:Dataset = {2, 4, 6, 8, 10} Mean = (2 + 4 + 6 + 8 + 10)/5 = 30/5 = 6
  - Advantages: Easy to compute, uses all data points.
  - Limitations: Sensitive to outliers (e.g., extreme values can distort the mean).
- **Median:** The middle value when the data is arranged in ascending (or descending) order.
  - Calculation:If nn is odd ⟶ Median = middle value.
  - If nn is even ⟶ Median = average of the two middle values.
  - Example: Dataset = {3, 5, 7, 9, 11} ⟶ Median = 7

    Dataset = {3, 5, 7, 9} ⟶ Median = (5+7)/2 = 6
  - Advantages: Not affected by outliers or skewed data.
  - Limitations: Ignores the magnitude of values (uses only positional data).

MEASURES OF CENTER AND SPREAD

- **Mode:** The value(s) that appear most frequently in a dataset.
  - Types:
    - Unimodal: One mode
    - Bimodal: Two modes
    - Multimodal: More than two modes
    - No mode: If all values occur with the same frequency
  - Example:Dataset = {2, 4, 4, 6, 8, 8, 8, 10} ⟶ Mode = 8
  - Advantages: Useful for categorical data (e.g., "most purchased product").
  - Limitations: May not exist or may not be unique.

MEASURES OF CENTER AND SPREAD

Measure	Best Used When	Sensitive to Outliers?	Data Type
Mean	Symmetric, numeric data	Yes	Interval/Ratio
Median	Skewed data or with outliers	No	Ordinal, Interval/Ratio
Mode	Most frequent item needed	No	Nominal, Ordinal, Interval

DATA TRANSFORMATION

- **Data Transformation**
  - Refers to a set techniques used to transform raw data into a more appropriate format to enable efficient data mining and model building.
  - In data transformation, the raw data are transformed or consolidated into forms appropriate for mining.
  - With data transformation,
    - The resulting mining process becomes more efficient and faster
    - Enhances compatibility in model building
    - The patterns found are of better quality and may be easier to understand.

DATA TRANSFORMATION

- **Strategies for data transformation include the following:**

1. **Data Smoothing**: works to remove noise from the data. Techniques include binning, regression, and clustering.
2. **Attribute construction:** new attributes are constructed and added from the given set of attributes to help the mining process.
3. **Aggregation**: summary or aggregation operations applied to the data.
   - For example: The *daily sales* data - aggregated to - *monthly* and *annual sales* amounts.
   - Typically used in constructing a **data cube** for data analysis

DATA TRANSFORMATION

4. Normalization: the attribute data are scaled so as to fall within a smaller range, such as --1.0 to 1.0, or 0.0 to 1.0.

5. Discretization: the raw values of a numeric attribute (e.g., *age*) are replaced by interval labels (e.g., 0–10, 11–20, etc.) or conceptual labels (e.g., *youth, adult*, *senior*).

- The labels, in turn, can be recursively organized into higher-level concepts, resulting in a *concept hierarchy* for the numeric attribute.

DATA TRANSFORMATION BY NORMALIZATION

Data Normalization

- The measurement unit used can affect the data analysis.
  - Attributes tend to have ranges different from each other
  - Expressing an attribute in smaller units will lead to a larger range, and thus tend to give such an attribute greater effect or “weight.”
- Variables with greater ranges tend to have larger influence on data model’s results.
- To help avoid dependence on the choice of measurement units (and thereby - the range), the data (numeric data) should be *normalized* or *standardized*.

DATA TRANSFORMATION BY NORMALIZATION

- Normalizing the data attempts to give all attributes an equal weight.
  - Particularly useful for classification algorithms involving *neural networks* or distance measurements such as nearest-neighbor classification and *clustering*.
- For distance-based methods, normalization helps prevent attributes with initially large ranges (e.g., *income*) from outweighing attributes with initially smaller ranges (e.g., binary attributes).

DATA TRANSFORMATION BY NORMALIZATION

❑Methods for data normalization.

- There are several methods for data normalization.
  - *min-max normalization,*
  - *z-score normalization,*
  - *normalization by decimal scaling.*

DATA TRANSFORMATION BY NORMALIZATION

❑Min-max normalization

- It performs a linear transformation on the original data.
- Min-max normalization maps a value, *v*, of *A* to v' in the range $[new\_min_A, new\_max_A]$ by computing:

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

- min_A : minimum values of an attribute, *A*.
- max_A : maximum values of an attribute, *A*.

DATA TRANSFORMATION BY NORMALIZATION

- Example (Min-max normalization)
  - Suppose that the minimum and maximum values for the attribute *income* are $12,000 and $98,000, respectively.
  - Map income to the range [0.0, 1.0].
  - By min-max normalization, a value of $73,600 for income is transformed to

$$\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716$$

  - Min-max normalization preserves the relationships among the original data values. It will encounter an “out-of-bounds” error if a future input case for normalization falls outside of the original data range for A

DATA TRANSFORMATION BY NORMALIZATION

❑ z-score normalization

- The values for an attribute are normalized based on the *mean* and *standard deviation*.
- A value, *v*, for an attribute *A*, is normalized to v' by computing:

$$v' = \frac{v - \bar{A}}{\sigma_A}$$

  - $\bar{A}$= mean of *A*
  - σ_A= standard deviation of *A*
- This method of normalization is useful when the *actual minimum* and *maximum* of attribute *A* are unknown.

DATA TRANSFORMATION BY NORMALIZATION

- Example (z-score normalization)
  - Suppose for the attribute *income*

    mean = $54,000

    standard deviation =$16,000

  A value of $73,600 for *income* is transformed to

$$\frac{73{,}600-54{,}000}{16{,}000} = 1.225$$

Note:

- Data values that lie below the mean will have a negative Z-score standardization.
- Data values that lie above the mean will have a positive Z-score standardization
- Data values falling exactly on the mean will have a *Z*-score standardization of zero

DATA TRANSFORMATION BY NORMALIZATION

❑Normalization by Decimal Scaling

- Normalizes by moving the decimal point of values of attribute *A*.
- The number of decimal points moved depends on the *maximum absolute value* of *A*.
- A value, *v*, of *A* is normalized to v' by computing

$$v' = \frac{v}{10^j}$$

- Where, *j* is the smallest integer such that *Max*(|v' |) < 1.
- Decimal scaling ensures that every normalized value lies between −1 and 1

DATA TRANSFORMATION BY NORMALIZATION

- Example (Normalization by Decimal Scaling)
  - Suppose that the recorded values of *A* range from

    -986 to 917.
  - The maximum absolute value of *A* is 986.
  - To normalize by decimal scaling, we therefore divide each value by 1,000 (i.e., $j = 3$) so that -986 normalizes to -0.986 and 917 normalizes to 0.917.

FLAG VARIABLES

- **Flag Variables (Dummy Variables Or Indicator Variables)**
  - In many analytical methods (e.g., regression, classification), predictors must be numeric.
  - However, real-world data often contains categorical variables.
  - To use them in models, we convert categories into numeric form using flag variables.
  - **Definition:** A **flag variable** is a binary variable that takes values:
  - 1 → if the condition is satisfied (category present)
  - 0 → if the condition is not satisfied (category absent)
  - **Example:** Smoker_Flag
    - Smoker = Yes → 1 , Smoker = No → 0

FLAG VARIABLES

- **Handling Multiple Categories with Flags**
  - If a categorical variable has **k categories**:
    - We typically create **k – 1 flag variables**.
    - The unassigned category becomes the **reference category**.
  - This avoids redundancy and the **dummy variable trap** (perfect multicollinearity in regression).

- Example
- Suppose we have a categorical variable '**region**' with 4 categories: {north, east, south, west}.
- Define k – 1 = 3 flag variables:
- **north_flag**:

  If region = north ⟶ 1, Otherwise ⟶ 0
- **east_flag**:

  If region = east ⟶ 1, Otherwise ⟶ 0
- **south_flag**:

  If region = south ⟶ 1, Otherwise ⟶ 0
- The **west region** is the **reference category**.

  If all three flags = 0, then region = west.

FLAG VARIABLES

- **Advantages**
  - Converts categorical data into numeric form, suitable for algorithms.
  - Easy to interpret (1 = present, 0 = absent).
- **Limitations**
  - Can increase dataset dimensionality if categories are many.
  - Requires careful choice of the **reference category**.\
- **Applications in Data Mining**
  - **Regression models**: Encode categorical predictors as numeric.
  - **Classification**: Enable algorithms to handle non-numeric inputs.
  - **Clustering**: Represent categorical features in binary form.

TRANSFORMING CATEGORICAL VARIABLES INTO NUMERICAL VARIABLES

- Many data mining and machine learning algorithms require predictors to be numeric.
- However, real-world data often contains **categorical variables**
- To make such data usable, we must **transform categorical variables into numerical form**.
- Two common approaches:
  - Assign numeric codes directly (often wrong).
  - Use **flag variables (dummy/indicator variables)**

TRANSFORMING CATEGORICAL VARIABLES INTO NUMERICAL VARIABLES

- **The Wrong Approach: Arbitrary Numeric Coding**
  - Let's have a categorical variable **region = {North, East, South, West}**.
  - If we assign numbers:
    - North = 1, East = 2, South = 3, West = 4
  - This creates a **false ordering**: West > South > East > North.
  - West is "three times closer" to South compared to North.
  - Algorithms wrongly assume **ordinal relationships** that do not exist.
  - This can lead to **biased or misleading model results**.

TRANSFORMING CATEGORICAL VARIABLES INTO NUMERICAL VARIABLES

- **The Correct Approach: Flag (Dummy) Variables**
  - Instead of coding categories with numeric variable, we use **flag variables**:
  - For **k categories**, create **k – 1 binary flags** (1 if present, 0 if absent).
  - This avoids false ordering and makes the data suitable for numeric algorithms.
- **Exception: Ordered (Ordinal) Categorical Variables**
  - Sometimes, categories have a **natural order (Ordinal Variables)**.
  - Example: Survey responses → {Always, Usually, Sometimes, Never}.
  - In such cases, numeric assignment is meaningful:
    - Always = 4, Usually = 3, Sometimes = 2, Never = 1

BINNING NUMERICAL VARIABLES

- Binning (also called discretization) is used to convert numerical variables into categories (bins or bands).
- Some algorithms prefer categorical rather than continuous predictors
- In such any numerical predictors are partitioned into bins or bands.
- For example, numerical variables ‘income’ value are portioned into - *low, medium*, and *high*.
- There are the following four common methods for binning numerical variables:

BINNING NUMERICAL VARIABLES

- **Equal Width Binning**
  - The numerical range is divided into k intervals of equal width.
  - Example:
    - Suppose, X={1,1,1,1,1, 2,2,11,11,12,12,44}, k=3.
    - Range = 44 – 1 = 43, Width = 43 / 3 ≈ 14.3
    - Bins:
      - Low: [1, 15.3) → contains most values {1,1,1,1, 1, 2,2,11,11,12,12}
      - Medium: [15.3, 29.6) → contains no values
      - High: [29.6, 44] → contains {44}
  - Drawback: Outliers can distort bins, leaving some bins empty and others overloaded.

BINNING NUMERICAL VARIABLES

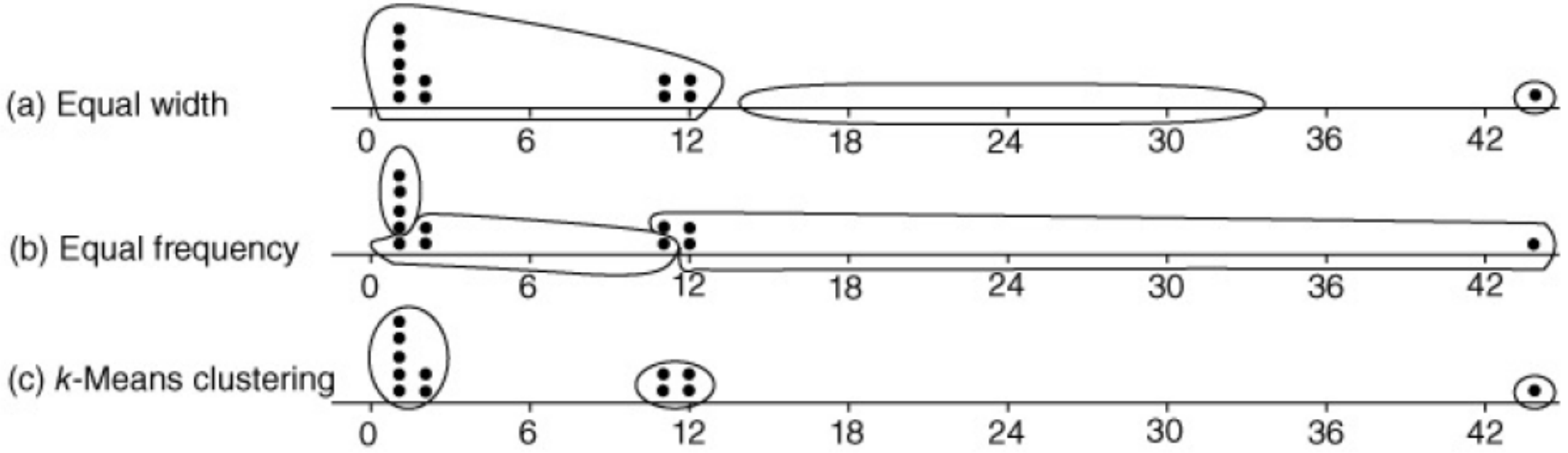
- **Equal Frequency Binning**
  - Each bin contains approximately the **same number of records**.
  - If n = total records, and k = number of bins, then each bin has n/k records.
  - **Example** (same dataset X={1,1,1,1,1, 2,2,11,11,12,12,44},
  - n = 12, k = 3 → each bin = 4 records):
    - Low: {1,1,1,1}
    - Medium: {1,2,2,11}
    - High: {11,12,12,44}
  - **Issue**: Identical values may fall into **different bins**, violating the principle that equal values should belong to the same category.

BINNING NUMERICAL VARIABLES

- **Binning by Clustering**
  - A clustering algorithm (e.g., **k-means**) is applied to group values.
  - Automatically determines “natural” partitions in data.
  - **Example:** Clustering the dataset X={1,1,1,1,1, 2,2,11,11,12,12,44},

    Cluster 1: {1,1,1,1,1,2,2} (Low)
    - Cluster 2: {11,11,12,12} (Medium)
    - Cluster 3: {44} (High)
  - □ Better than binning because it respects data distribution.

BINNING NUMERICAL VARIABLES



BINNING NUMERICAL VARIABLES

- **Binning Based on Predictive Value**
  - Partitions are created with respect to the target variable (class label or outcome).
  - Example: If ‘income’ value is used to predict "Loan Default," bins are chosen based on how values separate defaulters vs. non-defaulters.
  - Best method when prediction is the goal.
  - Considers target variable.

BINNING NUMERICAL VARIABLES

Method	Approach	Pros	Cons
Equal Width Binning	Divides range into equal widths	Simple	Distorted by outliers; empty bins possible
Equal Frequency Binning	Equal number of records per bin	Balances counts	Identical values may split
Clustering-based Binning	Uses clustering (e.g., k-means)	Respects data distribution	More complex
Predictive Value Binning	Based on target relationship	Improves prediction	Needs labeled data

RECLASSIFYING CATEGORICAL VARIABLES

- **Categorical variables** often contain **too many categories (levels or field values)**.
- Example: A variable like **State** in the U.S. has **50 categories**.
- Having too many categories can:
  - Increase model complexity and Reduce model accuracy.
  - Slow down computation.
  - Cause algorithms (e.g., logistic regression, decision trees) to perform **suboptimally**.
- To solve this, we use Reclassification (the categorical equivalent of binning numerical variables).

RECLASSIFYING CATEGORICAL VARIABLES

- **Reclassifying categorical variables** means combining or regrouping categories into **more meaningful groups** that are easier to analyze.
- Purpose: Reduce the number of categories while preserving or enhancing predictive power.
- Analogy: Just as binning numerical variables groups numbers into ranges, reclassifying categorical variables groups levels into categories.
- Example
  - **Original variable**: State (50 field values)
  - **Reclassified variable**: Region (5 categories)
    - Northeast, Southeast, North Central, Southwest, West
- Instead of 50 unique values, the model now sees only 5.

RECLASSIFYING CATEGORICAL VARIABLES

- **Methods of Reclassification**
  - **Domain Knowledge–Based Grouping**
    - Use expert knowledge or business rules.
    - Example: Grouping states into **geographic regions** or **market zones**.
  - **Data-Driven Grouping**
    - Use clustering or similarity measures to group categories.
    - Example: Group products based on **sales patterns** or **customer demographics**.
  - **Target-Oriented Grouping**
    - Group based on relationship with the target variable.
    - Example: If predicting **loan default**, classify occupations into **High Risk**, **Medium Risk**, and **Low Risk**.

REMOVING VARIABLES THAT ARE NOT USEFUL

- Not all collected variables contribute meaningfully to the analysis.
- Some variables may be redundant, irrelevant, or constant across the dataset.
- These variables must be removed; As, keeping such variables can lead to:
  - Increased computational cost
  - Noise in the data
  - Misleading results
- Two such variables include
  - unary variables and
  - variables that are very nearly unary.

REMOVING VARIABLES THAT ARE NOT USEFUL

- **Unary Variables:** A unary variable is a variable that takes on only one value across all observation.
  - Example: If a dataset contains students from an all-girls private school, the variable *sex* would have only one value, “female”.
    - This variable has no effect on classification, regression, or clustering algorithms.
- **Nearly Unary Variables:** A variable that is almost constant, with one value dominating nearly all records.
  - Example: In a field hockey league dataset: 99.95% of players - female and 0.05% of players -male
    - Algorithms may treat such variables as essentially unary.
    - For example, a classifier could be >99.9% confident that a new record is “female,” rendering the variable uninformative for predictive modeling.

VARIABLES THAT SHOULD PROBABLY NOT BE REMOVED

- Analysts often remove problematic variables during preprocessing,
- But some imperfect ones may still hold valuable insights.
- Two common examples are:
  - Variables with a high proportion of missing values
  - Variables that are strongly correlated
- Removing them blindly can:
  - Reduce predictive power
  - Oversimplify models
  - Discard valuable signals
- Key Principle: Data preprocessing should be thoughtful, not automatic. Preserve potential sources of insight whenever possible.

VARIABLES THAT SHOULD PROBABLY NOT BE REMOVED

- Variables with Many Missing Values
  - Common Practice: Remove variables with ≥90% missing values.
  - Problem: Missingness itself may carry useful information and hidden patterns.
  - Example: *donation_dollars* in a self-reported survey.
    - People who donate more may report proudly.
    - People who donate little may skip the question.
    - Thus, the missingness has a pattern and is not random.

VARIABLES THAT SHOULD PROBABLY NOT BE REMOVED

- Strategies Instead of Removal:
  - Create a flag variable (e.g., *donation_flag* = reported/not reported).
    - Captures missingness as potential predictor.
  - Impute missing values if the observed data are representative.
    - Use regression-based or decision tree-based imputation methods.
  - Evaluate representativeness of non-missing cases before imputing.
- Key Point:
  - Missingness may indicate characteristics, making it potentially predictive.
  - Avoid dropping variables simply because they contain lots of missing data.

VARIABLES THAT SHOULD PROBABLY NOT BE REMOVED

- **Strongly Correlated Variables**
  - Common Practice: Remove one of the variables.
  - Problem: Removing correlated variables may discard important information.
  - Example: *Precipitation* vs. *Beach Attendance*
    - Negatively correlated (more rain → fewer visitors).
    - Dropping one variable may ignore useful relationships.
  - Alternative Approach:
    - Use PCA or Use algorithms that can handle correlation (e.g., tree-based models).
      - Transforms correlated predictors into uncorrelated principal components.
      - Retains information while avoiding multicollinearity.