Data-Mining-Module-2-Important-Topics-PYQs

    - 3. Cluster Sampling
    - 4. Stratified Sampling
- 6. Suppose a group of 12 sales price records has been sorted as follows: 5, 10, 11, 13, 15,35,50, 55,72,92,204,215. Sketch examples of each of the following sampling techniques: SRSWOR, SRSWR, cluster sampling, stratified sampling. Use samples of size 5 and the strata "youth," "middle-aged," and "senior."
    - 1. SRSWOR (Simple Random Sampling Without Replacement)
    - 2. SRSWR (Simple Random Sampling With Replacement)
    - 3. Cluster Sampling
    - 4. Stratified Sampling
- 7. Real-world data tend to be incomplete, noisy and inconsistent. What are the various approaches adopted to clean the data?
    - 1. Handling Missing Values
        - (a) Ignore the Tuple
        - (b) Fill Missing Value Manually
        - (c) Use a Global Constant
        - (d) Use Attribute Mean
        - (e) Use Class-specific Mean
        - (f) Use Most Probable Value
    - 2. Handling Noisy Data
        - (a) Binning
        - (b) Clustering
        - (c) Regression
    - 3. Handling Inconsistent Data
- 8. Describe the various techniques for numerosity reduction in data mining.
    - 1. Parametric Methods
        - (a) Regression
        - (b) Log-Linear Model
    - 2. Non-Parametric Methods
        - (a) Histograms
    - (b) Clustering
    - (c) Sampling

- 9. Why do we need data transformation? What are the different ways of data transformation?
  - Different Ways of Data Transformation
- 10. Suppose that the data for analysis includes the attribute cost price and the values for the data tuples are: 100,150,140,115,190,120,130,125,135,145,140,150, 165,160,170
  - (i) Use min-max normalization to transform the value of 145 for cost price onto the range [0,1].
  - (ii) Use Z-Score normalization to transform the value 145 for cost price where the standard deviation of cost price is 120.
  - Data:
  - (i) Min-Max Normalization for the value 145
  - (ii) Z-Score Normalization
- 11. Discuss the significance of data discretization in data mining. List and explain any four data discretization strategies.
  - Significance of Data Discretization in Data Mining
  - 1. Discretization by Binning
  - 2. Discretization by Histogram Analysis
  - 3. Discretization by Clustering, Decision Tree, and Correlation Analysis
    - a. Discretization by Clustering
    - b. Discretization by Decision Tree
    - c. Discretization by Correlation Analysis (ChiMerge)
- 12. Illustrate PCA for dimensionality reduction with an example.
  - What is PCA?
  - PCA Step-by-Step
  - Step 1: Normalize the data
  - Step 2: Find principal components
  - Step 3: Sort components by importance
  - Step 4: Reduce dimensions
- 13. Explain data normalization methods with necessary equations. Calculate the normalized value for 550 in the set of the data points 50, 150, 250, 350,450, 550, 650, 700, 850, 950, 1000 using
  - (a) Min-Max Normalization
  - (b) Z-Score Normalization

1. Perform data smoothing by bin means on 3 equi-width bins.Data: [24,27,29,16,17,31,33,29,36,37,35,44]

Step 1: Given Data

```
Data: [24, 27, 29, 16, 17, 31, 33, 29, 36, 37, 35, 44]
```

The question also says that **3**-equi width bins are needed

Step 2: Sort the Data

First, sort the data (if not already sorted):

```
Sorted Data: [16, 17, 24, 27, 29, 29, 31, 33, 35, 36, 37, 44]
```

Step 3: Find the Range and Width of Each Bin

- Minimum = 16
- Maximum = 44
- Range = 44 - 16 = **28**
- Since we need **3 bins**,
  - Bin Width = Range ÷ Number of bins
  - Bin Width = 28 ÷ 3 ≈ **9.33** → Let's round it to **10** for easy bins.
  Thus, each bin will cover **~10** units.

Step 4: Create 3 Bins

Bin No.	Bin Range	Values in Bin
1	16 – 25	16, 17, 24
2	26 – 35	27, 29, 29, 31, 33, 35
3	36 – 45	36, 37, 44

Step 5: Find Mean of Each Bin

Bin 1:

- Values: 16, 17, 24
- Mean = (16 + 17 + 24) ÷ 3 = 57 ÷ 3 = **19**
  Bin 2:
- Values: 27, 29, 29, 31, 33, 35
- Mean = (27 + 29 + 29 + 31 + 33 + 35) ÷ 6 = 184 ÷ 6 ≈ **30.67**
  Bin 3:
- Values: 36, 37, 44
- Mean = (36 + 37 + 44) ÷ 3 = 117 ÷ 3 = **39**

Step 6: Replace Each Value by Bin Mean

Original Value	Bin	Smoothed Value
16	1	19
17	1	19
24	1	19
27	2	30.67
29	2	30.67
29	2	30.67
31	2	30.67
33	2	30.67
35	2	30.67
36	3	39
37	3	39
44	3	39

Final Smoothed Data:

```
[19, 19, 19, 30.67, 30.67, 30.67, 30.67, 30.67, 30.67, 39, 39, 39]
```

2. Explain concept hierarchy with an example.

Concept Hierarchy means replacing **specific low-level data** (like exact numbers) with **higher-level, more general concepts** to **reduce the data size** and **make patterns easier to find**.

✅ **Purpose:**

- To simplify the data.
- To generalize detailed values into broader categories.

Example 1: Age Concept Hierarchy

Suppose you have **exact ages** like:

```
16, 24, 35, 43, 60, 70
```

You can replace them by **high-level age groups**:

Age Value	Concept (Generalized)
16	Young
24	Young
35	Middle-aged
43	Middle-aged
60	Senior
70	Senior

This reduces the complexity by removing exact ages and replacing them with categories like **Young**, **Middle-aged**, and **Senior**.

Example 2: Month to Quarter

```
Jan, Feb, Mar → Q1
Apr, May, Jun → Q2
Jul, Aug, Sep → Q3
Oct, Nov, Dec → Q4
```

Here, 12 months are **grouped into 4 quarters**, making the data much simpler.

How Concept Hierarchies Are Formed:

There are different methods:

Method	Description	Example
Binning	Splitting data into bins.	0–20, 21–40, etc.
Histogram Analysis	Group values into buckets.	Grouping 0–25, 26–50, etc.
Clustering	Group similar data points together.	Similar income ranges grouped.
Manual Hierarchy	Define by hand at schema level.	street < city < state < country

Example 3: Location Concept Hierarchy

Suppose you have address data:

```
Street → City → State → Country
```

You can generalize:

- Instead of storing each **street**, store just the **city** or **state** when needed.
  Example
- "221B Baker Street" becomes "London" (City level)
- "London" becomes "United Kingdom" (Country level)

3. What is the purpose of data discretization? List any two data discretization strategies

Data Discretization is the process of **converting continuous data** into a **small number of intervals** (bins or categories).
Main Purpose:

- To **reduce the number of distinct values** in continuous attributes.
- To **simplify the data** and **make data mining more efficient**.
- Helps algorithms (especially decision trees) to work better on the data.

Example:
Instead of having many different age values like 16, 17, 18, 19, 20, etc.,
we group them as:

- 15–20 → "Teen"
- 21–30 → "Young Adult"
- etc

Two Data Discretization Strategies

Strategy	Description
Top-Down Discretization	Start with the whole range and split it into intervals step-by-step. Also called **splitting**.
Bottom-Up Discretization	Start with all individual data points and **merge** neighboring values into bigger intervals.

4. Justify the significance of pre-processing the data before mining.

Data preprocessing is a **very important step** before applying any data mining techniques because:

- **Databases come from multiple heterogeneous sources**, meaning the data can be inconsistent, incomplete, or noisy.
- **Low-quality data** will result in **poor mining outcomes**, leading to **incorrect patterns** and **wrong decisions**.

Why Data Preprocessing is Crucial?

- Ensures **high-quality**, **consistent**, and **reliable** data.
- Improves **accuracy**, **efficiency**, and **interpretability** of mining results.
- Prevents **wrong patterns** and **misleading conclusions**.
- Makes the data **ready and optimized** for further analysis.

Simple Example:

- Imagine trying to mine customer behavior data with missing age or incorrect income values — the mining algorithm will give **wrong insights**.
- After **cleaning and transforming** the data, the mining will produce **meaningful patterns** like "young customers prefer online shopping".

5. Explain the two sampling methods used in data reduction.

Sampling is a technique in **data reduction** where we select a small representative subset of a large dataset, so that the analysis becomes faster and easier without much loss of information.

important sampling methods are:

1. Simple Random Sample Without Replacement (SRSWOR)

- We randomly select `s` tuples (records) from the dataset `D`, where `s < N` (`N` is the total number of tuples).
- **Important Point:** Once a tuple is selected, it is **not put back** into the dataset.
- So, each tuple can be selected **only once**.
- **Probability** of picking any tuple = 1/N (all are equally likely).
- **Example:**
  - If you have 1000 customer records and want a sample of 100, you randomly pick 100 different customers without repeating anyone.

2. Simple Random Sample With Replacement (SRSWR)

- We randomly select `s` tuples from dataset `D`, but **after selecting a tuple, it is put back** into the dataset.
- So, **the same tuple can be selected more than once**.
- Every time, the probability of selecting any tuple stays 1/N.
- **Example:**
  - If you randomly pick a customer record, you put it back into the pool after selection, so the same customer might get picked multiple times in your sample of 100.

Method	Replacement?	Same tuple selected twice?
SRSWOR	No	No
SRSWR	Yes	Yes

3. Cluster Sampling

- **Definition:**
  In **cluster sampling**, the entire data set (say, dataset **D**) is divided into **mutually disjoint groups** called **clusters**.

- **Process:**
  Instead of selecting individual tuples (records) randomly, you randomly select **entire clusters**.
- **Example:**
  Suppose you have a database of **10,000 students** from **100 schools**.
  Instead of sampling individual students, you randomly select, say, **10 schools** (clusters) and collect **all students** from these 10 schools.
- **Key Points:**
  - Reduces effort because entire groups are selected at once.
  - Objects within a cluster are usually **similar**.
  - It is useful when it is difficult or expensive to create a complete list of individuals but easy to list groups.

4. Stratified Sampling

- **Definition:**
  In **stratified sampling**, the data set **D** is divided into **strata** (groups) based on some important attribute.
  Then, **a simple random sample** is drawn **separately** from each stratum.
- **Process:**
  1. Divide the dataset into strata based on an attribute (like gender, age group, region, etc.).
  2. Perform random sampling within each stratum.
- **Example:**
  Suppose you have a database of employees where **60% are male** and **40% are female**.
  To maintain this proportion, you separately sample **males** and **females** — ensuring that your sample also reflects 60%-40% gender ratio.
- **Key Points:**
  - Helps ensure that **each subgroup** is properly represented.
  - Increases the **accuracy** and **reliability** of the sample.
  - It is useful when the population is **heterogeneous** (diverse).

6. Suppose a group of 12 sales price records has been sorted as follows: 5, 10, 11, 13, 15,35,50, 55,72,92,204,215.

Sketch examples of each of the following sampling techniques: SRSWOR, SRSWR, cluster sampling, stratified sampling. Use samples of size 5 and the strata "youth," "middle-aged," and "senior."

We have 12 sorted records:
`5, 10, 11, 13, 15, 35, 50, 55, 72, 92, 204, 215`

We are asked to sketch examples of these sampling techniques:

- SRSWOR (Simple Random Sampling Without Replacement)
- SRSWR (Simple Random Sampling With Replacement)
- Cluster Sampling
- Stratified Sampling (with strata: "youth," "middle-aged," "senior")

We need **samples of size 5**.

1. SRSWOR (Simple Random Sampling Without Replacement)

- Randomly pick **5 different records** — no repetition allowed.
- Example sample (randomly chosen):
  - → `{11, 15, 35, 55, 204}`
  - No repetition, each selected only once.

2. SRSWR (Simple Random Sampling With Replacement)

- Randomly pick **5 records**, **allowing repetition**.
- Example sample (randomly chosen):
  - → `{10, 15, 15, 204, 5}`
  - Notice here **15** was selected **twice**.

3. Cluster Sampling

- First, divide records into **clusters** (groups).
  (We can group nearby numbers together.)

Example clusters

- Cluster 1: `{5, 10, 11, 13}`
- Cluster 2: `{15, 35, 50, 55}`
- Cluster 3: `{72, 92, 204, 215}`

Now, **randomly pick 1 cluster** (or more depending on how big you want the sample).
Suppose we randomly pick **Cluster 2**:
→ `{15, 35, 50, 55}`

But Cluster 2 only has 4 records — since we want 5 samples, maybe pick **another cluster** randomly.

Suppose pick Cluster 1 too:
→ `{5, 10, 11, 13}`

Now from the union of these two clusters `{5, 10, 11, 13, 15, 35, 50, 55}`, randomly pick **5 records**.

Example:
→ `{5, 15, 35, 50, 55}`

4. Stratified Sampling

Given strata:

- **Youth**: Lower prices (say 5–50)
- **Middle-aged**: Medium prices (51–100)
- **Senior**: High prices (>100)

So, divide the 12 records:

- Youth: `{5, 10, 11, 13, 15, 35, 50}` (7 records)
- Middle-aged: `{55, 72, 92}` (3 records)
- Senior: `{204, 215}` (2 records)

Now sample separately from each stratum:
Suppose we select

- 2 records from Youth → `{10, 35}`
- 2 records from Middle-aged → `{55, 92}`
- 1 record from Senior → `{204}`

Final stratified sample:
→ `{10, 35, 55, 92, 204}`

✅ Each group is represented properly.

7. Real-world data tend to be incomplete, noisy and inconsistent. What are the various approaches adopted to clean the data?

Real-world data is often:

- **Incomplete** (missing values)
- **Noisy** (random errors)
- **Inconsistent** (contradictory values)
  Data cleaning aims to **fill missing values, remove noise, and fix inconsistencies**.

1. Handling Missing Values

If some data (like customer income) is missing, we can:

(a) Ignore the Tuple

- Just skip the record if too much information is missing.
- Useful **only** if missing data is rare.

(b) Fill Missing Value Manually

- Human experts fill in missing info.
- Not practical for **large datasets**.

(c) Use a Global Constant

- Replace missing values with a label like `"Unknown"`.
- Can mislead analysis because "Unknown" looks like a real value.

(d) Use Attribute Mean

- Fill missing value with the **average** of that attribute.
- E.g., if average income = $56,000, use $56,000.

(e) Use Class-specific Mean

- If data is classified (like credit risk), fill missing value with **mean for that class**.
- E.g., average income for "high-risk" customers.

(f) Use Most Probable Value

- Use techniques like **regression**, **Bayesian methods**, or **decision trees** to predict missing values.

2. Handling Noisy Data

Noise = random errors or variability in data.
Methods to smooth noisy data:

(a) Binning

- **Sort** data first.
- Divide into small groups ("bins").
- Then smooth:
  - **Bin mean:** Replace each value with the bin’s average.
  - **Bin median:** Replace with bin's median.
  - **Bin boundaries:** Replace each value with nearest boundary (min or max).

(b) Clustering

- Group similar data points into clusters.
- Values **far from any cluster** are treated as **outliers**.

(c) Regression

- Fit a line (or curve) to the data.
- Predict missing or noisy values using this line.
- **Linear regression** (2 variables) or **multiple regression** (many variables).

3. Handling Inconsistent Data

Inconsistent data = data that **conflicts** or **breaks rules**.
How to handle:

- **Manual correction:** Cross-check with papers or trusted sources.
- **Code standardization:** Fix different code usages (e.g., USA vs US).
- **Use knowledge rules:** Apply known rules or constraints.
    - Example: If **state** depends on **ZIP code**, mismatch indicates inconsistency.

8. Describe the various techniques for numerosity reduction in data mining.

Numerosity Reduction = Reducing the size of the data without losing important information. Instead of keeping all the original data, we keep

- Only a **smaller version**
- Or **a model** that summarizes the data.

1. Parametric Methods

- Assume that the data follows a certain model (like a line or curve).
- Store only the **parameters** of the model (not the full data).

(a) Regression

- Find a **line** or **equation** that fits the data.
    - **Simple linear regression**:
    - **Multiple linear regression**:
- Good for **predicting** values using other variables.

(b) Log-Linear Model

- Used for **categorical (discrete)** data.
- Studies the **probability** of different combinations in multi-dimensional space.
- Useful for **sparse** and **skewed** datasets.

2. Non-Parametric Methods

- **No assumption** about data's distribution.
- Directly **compress** or **transform** the data.

(a) Histograms

- Divide the range of data into **buckets** (bins).
- Two main types:
  - **Equal-width**: Buckets have the same size range.
  - **Equal-frequency**: Each bucket has the same number of data points.
- Instead of storing all values, store only:
  - Range of bucket
  - Count of values
  Example:
  If prices are between $0–$50, buckets could be: $0–$10, $10–$20, etc.

(b) Clustering

- Group similar data points into **clusters**.
- Only store **cluster centers** (centroids) and cluster size.
- Outliers can also be detected.

Example:
Customers living close to each other are grouped into clusters.

(c) Sampling

- Select only a **small part** (sample) of the data.
- Types of Sampling:
  - **SRSWOR (Simple Random Sample Without Replacement)**:
    - Pick randomly without putting the data back.
  - **SRSWR (Simple Random Sample With Replacement)**:
    - Pick randomly, but allow repetition.
  - **Cluster Sampling**:
    - Divide data into **clusters** and sample a few clusters.
  - **Stratified Sampling**:
    - Divide data into **strata** (groups like age groups) and sample from each group.

9. Why do we need data transformation? What are the different ways of data transformation?

- **Raw data** is often noisy, inconsistent, or unsuitable for mining.
- **Data Transformation** helps to **prepare** and **improve** data for better mining results.
- It **makes data:**
  - Cleaner
  - Simpler
  - Easier to analyze
  - More accurate in results

Different Ways of Data Transformation

1. **Smoothing**
   - Removes **noise** (errors/randomness) from data.
   - Methods:
     - **Binning**: Grouping values into bins.
     - **Regression**: Fitting data to a function.
     - **Clustering**: Grouping similar items.
2. **Aggregation**
   - **Summarizing** data.
   - Example:
     Daily sales → Monthly sales → Annual sales.
   - Used for building **data cubes** (multi-level data analysis).
3. **Generalization**
   - Replace detailed data with **higher-level concepts**.
   - Example:
     "Street" → "City" → "Country".
   - Helps to view **bigger patterns**.
4. **Normalization**
   - **Rescales** data into a specific range like [0,1] or [-1,1].
   - Needed because different measurement units (like meters vs inches) can mess up analysis.
   - Three types:

- **Min-max normalization**:
  - Scales between a minimum and maximum.
- **Z-score normalization**:
  - Based on **mean** and **standard deviation**.
- **Decimal scaling**:
  - Move the **decimal point** to normalize.

5. **Attribute Construction (Feature Construction)**
   - **Create new attributes** using existing ones.
   - Helps the model understand better and improve mining results.

Technique	Purpose	Example
Smoothing	Remove noise	Binning, regression
Aggregation	Summarize data	Daily → Monthly sales
Generalization	Higher-level concepts	Street → City
Normalization	Scale data to a range	Min-max, z-score, decimal scaling
Attribute Construction	Create new helpful attributes	Combine 'height' and 'weight' to make 'BMI'

Normalization types

Method	How It Works	When to Use
Min-max normalization	Linear scaling between [new_min, new_max]	Data has known min/max
Z-score normalization	Center data to mean = 0, std dev = 1	Data has unknown min/max or has outliers
Decimal scaling	Shift decimal point based on max value	Simple datasets

10. Suppose that the data for analysis includes the attribute cost price and the values for the data tuples are: 100,150,140,115,190,120,130,125,135,145,140,150, 165,160,170

(i) Use min-max normalization to transform the value of 145 for cost price onto the range [0,1].

(ii) Use Z-Score normalization to transform the value 145 for cost price where the standard deviation of cost price is 120.

Data:

```
100,150,140,115,190,120,130,125,135,145,140,150, 165,160,170
```

The **value to normalize** is 145.

(i) Min-Max Normalization for the value 145

Formula for Min-Max Normalization:

$$v' = \frac{(v - \min_A)}{(\max_A - \min_A)} \times (\text{new\_max}_A - \text{new\_min}_A) + \text{new\_min}_A$$

Where:

- v is the original value (145).
- $\min_A$ is the minimum value in the dataset.
- $\max_A$ is the maximum value in the dataset.
- new_min_A and new_max_A are the target range, which in this case is [0, 1].

Steps:

1. **Find min and max values:**
   - Min value $\min_A = 100$
   - Max value $\max_A = 190$
2. **Substitute values into the formula:**

$$v' = \frac{(145 - 100)}{(190 - 100)} \times (1 - 0) + 0$$

$$v' = \frac{45}{90} \times 1 = 0.5$$

So, the **Min-Max normalized value** for 145 is **0.5**.

(ii) Z-Score Normalization

Formula for **Z-Score Normalization**:

$$v' = \frac{(v - \mu)}{\sigma}$$

Where:

- v is the original value (145).
- μ is the mean of the data.
- σ is the standard deviation of the data, which is given as **120**.

Steps:

1. **Find the mean (μ) of the dataset:**

$$\mu = \frac{\sum \text{data values}}{n} = \frac{100 + 150 + 140 + 115 + 190 + 120 + 130 + 125 + 135 + 145 + 140 + 150 + 165 + 160 + 170}{15}$$

$$\mu = \frac{2,005}{15} = 133.67$$

2. **Apply the Z-score formula:**

$$v' = \frac{(145 - 133.67)}{120} = \frac{11.33}{120} = 0.0944$$

So, the **Z-Score normalized value** for 145 is approximately **0.0944**.

---

11. Discuss the significance of data discretization in data mining. List and explain any four data discretization strategies.

Significance of Data Discretization in Data Mining

- **Data Discretization** is the process of converting a large set of continuous attribute values into a smaller set of discrete intervals.
- It simplifies the data, making it **easier to analyze**, **manage**, and **mine**.
- Discretization **reduces data size** and helps in **improving the performance** of data mining algorithms.
- It also helps in **building concept hierarchies** for further analysis.
- **Example:** Instead of handling hundreds of exact Age values, we can create intervals like "Young", "Middle-aged", "Old".

1. Discretization by Binning

- Think of binning like **putting data into small boxes**.
- We **divide the whole range into small parts (bins)** — either all bins have the same size, or each bin has the same number of values.
- Inside each bin, we can **replace all the values with the average** (mean) or **middle value** (median) to make it even simpler.
- **Example:** If we have ages like 5, 7, 8, 9, 12, 13, 18 — we can group them into bins like:
  - Bin 1: 0–10 → (5,7,8,9)

- Bin 2: 11–20 → (12,13,18)

2. Discretization by Histogram Analysis

- Imagine drawing a **bar chart (histogram)** of the data.
- Here also, data is **grouped into bins** (small ranges).
- Two ways to do it:
  - **Equi-width:** Each bin has **same width** (like 0–10, 10–20, etc.).
  - **Equal-frequency:** Each bin has **same number of data points** (like 5 values in each bin).
- This helps **see the data pattern** and **simplify** it.

3. Discretization by Clustering, Decision Tree, and Correlation Analysis

a. Discretization by Clustering

- **Clustering means grouping similar things together**.
- Data points that are **close or similar** are put into one group (cluster).
- Each group acts like one interval (bin) for the data.
- It's like **putting students into groups** based on similar marks!

b. Discretization by Decision Tree

- Decision trees **divide data step-by-step** based on some conditions.
- It **uses class labels** (extra information) to decide **where to cut** the data.
- **Example:** In medical data, a tree can split patient records based on symptoms to predict a disease.
- It tries to **make groups as pure as possible** (mostly same class inside a group).

c. Discretization by Correlation Analysis (ChiMerge)

- This method **starts with each value as its own bin**.
- Then, it **merges bins** that are **similar** based on a special test called **Chi-square (χ^2) test**.
- It keeps merging until the groups are good enough.
- It's like **combining small groups into bigger ones** when they look similar.

12. Illustrate PCA for dimensionality reduction with an example.

What is PCA?

- PCA is a method to **reduce the number of attributes (features)** in your data.
- It **finds new directions (principal components)** along which the **data varies the most**.
- Then it **replaces** the original data with these **few important directions**.
- **Goal:** Keep the important patterns, **remove extra dimensions**.

PCA Step-by-Step

Suppose you have the following simple data of students:

Height (cm)	Weight (kg)
150	50
160	55
165	58
170	62
175	65

You have 2 attributes: **Height** and **Weight**.
Now, let's apply PCA:

Step 1: Normalize the data

- Make sure height and weight are on the **same scale**.

Step 2: Find principal components

- PCA will find **new axes (directions)** along which the data varies the most.
- Here, **Height and Weight are strongly related** (when height increases, weight increases).
- So PCA finds:
  - **1st Principal Component (Y_1):** The direction where data varies the most (Height and Weight together).

- **2nd Principal Component (Y_2):** A smaller variation direction (tiny differences not captured by Y_1).

Step 3: Sort components by importance

- Y_1 captures most of the pattern (Height ↔ Weight relationship).
- Y_2 captures very little (random noise).

Step 4: Reduce dimensions

- We **keep only Y_1** and **discard Y_2**.
- Now instead of recording 2 things (height and weight), we just record **one value** along Y_1.
- **Dimension reduced from 2D to 1D**

13. Explain data normalization methods with necessary equations. Calculate the normalized value for 550 in the set of the data points 50, 150, 250, 350,450, 550, 650, 700, 850, 950, 1000 using

- **(a) min-max normalization by setting min=0 and max=I**
- **(b) z-score normalization**
- **(c) Normalization by decimal scaling**

Normalization means **scaling** the data values into a **small, specified range** (like [0,1]) so that no feature dominates others.

(a) Min-Max Normalization

Idea:

- Scale the data value between a **new minimum and new maximum** (usually [0,1]).

Formula:

$$v' = \frac{(v - min_{data})}{(max_{data} - min_{data})} \times (new\_max - new\_min) + new\_min$$

Given:

- min = 0, max = 1 (so range = [0,1])
- data points: 50, 150, 250, 350, 450, 550, 650, 700, 850, 950, 1000
- $\min_{data} = 50$
- $\max_{data} = 1000$
- $v = 550$

Substitute into the formula:

$$v' = \frac{(550 - 50)}{(1000 - 50)} \times (1 - 0) + 0$$

$$v' = \frac{500}{950}$$

$$v' \approx 0.5263$$

✅ **Answer: 0.5263**

(b) Z-Score Normalization

Idea:

- Convert the value to how many **standard deviations** it is from the **mean**.

Formula:

$$v' = \frac{(v - \mu)}{\sigma}$$

where:

- μ = mean
- σ = standard deviation

Find each data point - mean

Data Point (x)	(x−μ)
50	50 - 540.91 = -490.91
150	150 - 540.91 = -390.91
250	250 - 540.91 = -290.91
350	350 - 540.91 = -190.91
450	450 - 540.91 = -90.91
550	550 - 540.91 = 9.09
650	650 - 540.91 = 109.09
700	700 - 540.91 = 159.09
850	850 - 540.91 = 309.09
950	950 - 540.91 = 409.09
1000	1000 - 540.91 = 459.09

Square each deviation

Deviation (x−μ)	Square (x−μ)^2
-490.91	240,991.63
-390.91	152,810.63
-290.91	84,628.63
-190.91	36,446.63

Deviation (x−μ)	Square (x−μ)^2
-90.91	8,264.63
9.09	82.64
109.09	11,901.64
159.09	25,280.64
309.09	95,528.64
409.09	167,776.64
459.09	210,195.64

Find the sum of all squared deviations

Sum = **1,033,907.4**

Divide by number of data points (n)

$$\sigma = \sqrt{\frac{\text{sum of squared deviations}}{n}}$$

Thus:

$$\sigma = \sqrt{\frac{1033907.4}{11}}$$

$$\sigma = \sqrt{94082.49}$$

$$\sigma \approx 306.74$$

✅ **Standard Deviation = 306.74**

Find z-score

$$z = \frac{550 - 540.91}{306.74}$$

$$z = \frac{9.09}{306.74}$$

$$z \approx 0.0296$$

(c) Normalization by Decimal Scaling

Idea:

- Move the decimal point of values to make all data values fall between -1 and 1.

Formula:

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

where **j** is the smallest integer such that max(|v'|) < 1.

You have to **shift the decimal point** (by dividing by 10^j) to make **all** data values **between -1 and 1**.
So:

- Find the **maximum absolute value** (biggest value ignoring + or - sign).

- Choose the **smallest** j so that after dividing by 10^j the maximum value becomes **< 1**.

  Your numbers:
  $50, 150, 250, 350, 450, 550, 650, 700, 850, 950, 1000$

  👉 **Max value = 1000**

  Now, how much should we divide 1000 by to make it **less than 1**?

  - $10^1 = 10 \rightarrow 1000/10 = 100$ ❌ (Still > 1)
  - $10^2 = 100 \rightarrow 1000/100 = 10$ ❌ (Still > 1)
  - $10^3 = 1000 \rightarrow 1000/1000 = 1$ ❌ (Not < 1, just equal)
  - $10^4 = 10000 \rightarrow 1000/10000 = 0.1$ ✅ (Now < 1!)

  But here **standard method** says:
  👉 We allow "1" — **equal to 1 is acceptable in most cases** when doing decimal scaling.

  Thus, **j = 3** is usually considered correct, because:

  - $1000/1000 = 1$ is **fine**.
  - No need to go up to $j = 4$.

  ✅ **Conclusion:** $j = 3$

$$v' = \frac{550}{10^3} = \frac{550}{1000} = 0.55$$

14. Explain Attribute Subset Selection

What is Attribute Subset Selection?

Attribute Subset Selection is a process used to select the most important attributes (or features) from a dataset, and discard the irrelevant or redundant ones. The goal is to reduce the complexity of the model by focusing on the most relevant attributes.

In simpler terms, instead of using every single piece of data in a dataset, we pick only the ones that matter the most for making predictions. This helps in:

- **Reducing complexity**: We don't have to deal with too many variables.
- **Improving performance**: The model becomes faster and may perform better.

- **Easier interpretation**: It becomes easier to understand which attributes are most important.

How Does Attribute Subset Selection Work?

The process typically follows these steps

1. **Start with all attributes**: Initially, you have all the attributes (or features) from the dataset.
2. **Evaluate the importance of each attribute**: Look at how much each attribute contributes to the prediction or classification task.
3. **Remove unnecessary or redundant attributes**: Eliminate the attributes that don't help much in making predictions.
4. **Repeat**: Keep selecting the best attributes until you have a small, efficient set.

There are **different methods** for selecting attributes:

- **Stepwise Forward Selection**
- **Stepwise Backward Elimination**
- **Decision Tree Induction** (which we'll cover in this example)

Example Dataset:

Let's say we have a small dataset of people with the following attributes:

Age	Income	Education Level	Health	Buy Product?
25	30,000	High School	Good	Yes
40	50,000	Bachelor's	Average	No
35	45,000	Master's	Good	Yes
50	70,000	Bachelor's	Poor	No
28	40,000	High School	Good	Yes

The task is to predict whether a person buys a product based on their **Age**, **Income**, **Education Level**, and **Health**.

The **target variable** is **Buy Product?** (Yes/No).

Stepwise Forward Selection:

In **Stepwise Forward Selection**, we start with no attributes, and then add the most useful attributes one at a time.

Step 1: Start with no attributes

At the beginning, we have no selected attributes.

Step 2: Evaluate the best attribute to start with

Now we evaluate the individual attributes (**Age**, **Income**, **Education Level**, **Health**) to see which one has the most significant impact on predicting whether someone buys the product.

- We test each attribute independently and calculate how well each one predicts **Buy Product?**. We might use a measure like **accuracy** or **information gain** to see how well each attribute splits the data.

Let's say that **Income** gives the best result, so we choose **Income** as the first attribute.

Step 3: Add the best attribute to the model

Now, **Income** is our first chosen attribute, so the selected attributes are:

- **Income**

Step 4: Evaluate the next best attribute to add

Next, we evaluate which additional attribute works best with **Income**. We now have to evaluate the combinations of **Income** and the remaining attributes (**Age**, **Education Level**, **Health**)

- Let’s say after testing combinations, **Income + Age** performs best.

Step 5: Add the best combination of attributes

Now, the selected attributes are:

- **Income, Age**

Step 6: Continue until no significant improvement

We keep adding attributes until we see no significant improvement in performance. For example, adding **Education Level** and **Health** might not improve the prediction much, so we stop.

Final Selected Attributes:

- **Income, Age**
  These are the attributes that give us the best prediction of whether someone will buy the product.

15. Explain Data Integration and Data Transformation

In data mining, **data integration** and **data transformation** are two important preprocessing steps that help prepare data for analysis

1. Data Integration

Data Integration refers to the process of merging data from multiple sources into a unified dataset. This step is important because data often resides in various places: different databases, files, or systems. Integrating this data ensures that all relevant information is brought together for analysis.

Key Considerations in Data Integration:

1. **Entity Identification Problem**:
   - When merging data from different sources, it's important to ensure that equivalent entities from various data stores are correctly matched. For example, a customer ID in one database might be represented as `customer_id`, and in another database, it might be `cust_number`. The system must recognize that these two fields refer to the same entity.
   - **Solution**: Use metadata (descriptive data about data) to help identify and match these entities correctly.
2. **Redundancy and Correlation Analysis**:
   - **Redundancy** occurs when the same information is stored in multiple places. For instance, if two attributes (e.g., `annual_income` and `yearly_salary`) provide the same information, one can be removed to avoid redundancy.
   - **Correlation Analysis** can be used to identify redundancies between attributes. For instance, if two attributes show a high positive correlation (close to +1), it suggests that one attribute might be redundant and can be removed.
3. **Data Value Conflicts**:
   - Sometimes, data values from different sources may conflict. For example, one database may store weights in kilograms, while another stores them in pounds. To handle this,

data needs to be **standardized** to a common format.

- **Example**: One database might record `price` in USD, while another records it in EUR. These need to be normalized to the same currency for accurate analysis.

Example

Suppose we have customer data from two sources:

- **Source 1**: `customer_id`, `name`, `age`, `address`
- **Source 2**: `cust_number`, `full_name`, `age`, `location`

When integrating the data, we need to handle issues like matching `customer_id` with `cust_number` and `name` with `full_name`. Additionally, we would handle potential redundancies and conflicts in fields like `address` and `location`.

2. Data Transformation

Data Transformation involves converting data into a format suitable for mining. This could mean scaling values, aggregating data, or simplifying complex data into more useful forms.

Key Data Transformation Methods:

1. **Smoothing**:
   - Smoothing helps to remove noise from the data (i.e., inconsistencies or outliers). Techniques like **binning**, **regression**, or **clustering** can be used to smooth the data and make it more reliable for analysis.
   - **Example**: If we have sales data with fluctuations due to occasional outliers, we might smooth the data to get a clearer trend.
2. **Aggregation**:
   - Aggregation involves summarizing data at a higher level. For example, instead of tracking daily sales, you might aggregate the data to show monthly or yearly sales.
   - **Example**: If the dataset contains daily sales data, aggregation could provide the total sales for each month.
3. **Generalization**:
   - Generalization replaces detailed data with higher-level concepts. For example, instead of having a specific street address, we could generalize to a city or country.
   - **Example**: Instead of tracking individual products, we might generalize the data to product categories like "Electronics" or "Clothing."

4. **Normalization**:
    - **Normalization** is the process of scaling data to fit within a specific range (like 0 to 1 or -1 to 1). This is particularly important when attributes have different units or scales. Common methods include **min-max normalization** and **z-score normalization**.
    - **Example**: If we have data about income (ranging from $12,000 to $98,000) and age (ranging from 18 to 70), normalization ensures that these attributes are on the same scale for comparison.
5. **Attribute Construction**:
    - In some cases, new attributes (or features) are created from existing ones to help improve the analysis. This process is known as **feature construction**.
    - **Example**: If we have data about height and weight, we might construct a new attribute called **BMI (Body Mass Index)**.