

Unit 3

Data Understanding and Preprocessing

- 3.1 Types of data: Structured, unstructured, semi-structured
- 3.2 Data preprocessing requirements
- 3.3 Data sources and collection methods
- 3.4 Data cleaning and preparation
- 3.5 Data wrangling and associated tools
- 3.6 Data enrichment, validation and publishing
- 3.7 Data transformation and normalization
- 3.8 Dimensionality reduction linear factor model, principal component analysis (PCA)

Data understanding and preprocessing are crucial steps in the data science pipeline, often consuming a large portion of a data scientist's time. Why is it so crucial? In essence, data is messy. Real-world data, the kind that companies and organizations collect every day, is filled with inaccuracies, inconsistencies, and missing entries. As the saying goes, "Garbage in, garbage out." If we feed our predictive models with dirty, inaccurate data, the performance and accuracy of our models will be compromised.

In a broader sense, data understanding and preprocessing are necessary to ensure data integrity. They enable us to refine and organize the raw data into a more suitable format that can be analyzed effectively and reliably. The integrity of the data we use in our analyses directly affects the validity of our conclusions. Therefore, spending time on this stage of the pipeline can save us from drawing incorrect conclusions, making poor decisions, or developing ineffective models.

3.1 Data Types

In data science, data can be broadly categorized into multiple ways:

1. Based on structure: **Structured Data, Unstructured Data, Semi-Structured Data**
2. Based on Nature: **Qualitative and Quantitative data.**
3. Based on Source: **Primary and Secondary source.**

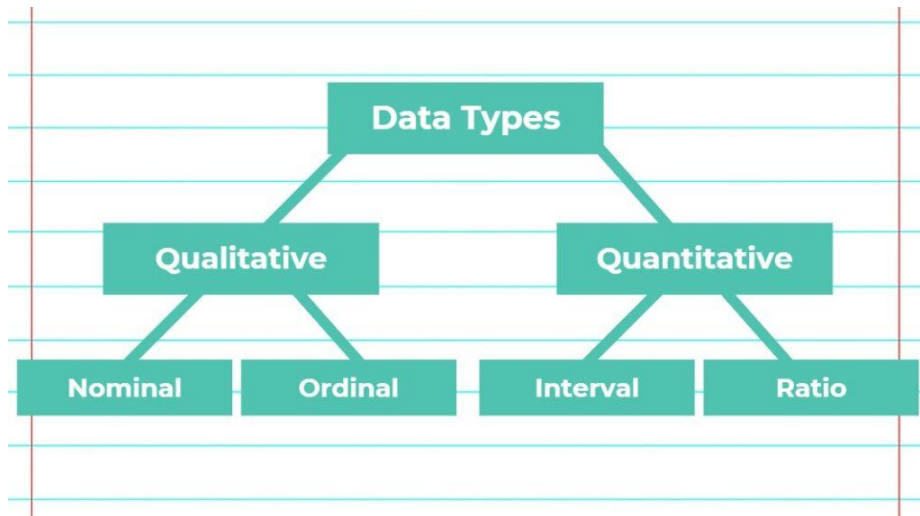
Based on structure, in data science data can be broadly categorized into three types:

Structured Data: This refers to data that is organized into rows and columns, such as in relational databases or spreadsheets. Structured data is easy to analyze using traditional statistical methods. Examples include sales records, employee data, tables in SQL databases, and transaction logs.

Unstructured Data: Unstructured data lacks a predefined format, making it more challenging to process. Examples include images, videos, social media posts, and audio files. Analyzing unstructured data requires advanced techniques like natural language processing (NLP) or image recognition.

Semi-Structured Data: This type lies between structured and unstructured data. It doesn't conform to a rigid structure but contains tags or markers that make it easier to process. Common examples include JSON files, XML files, and log data from servers.

In data science, data can be broadly categorized into two types(based on nature): Qualitative and Quantative data.



Quantitative Data

- Numbers, measurements, and quantities
- How much or how many?
- Easily quantified and manipulated mathematically
- Can perform various mathematical operations

Qualitative Data

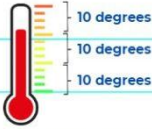
- Qualities, attributes, and characteristics
- What kind or what category?
- Can not be easily subjected to mathematical operations

Real World Scenarios

- Data often combines both quantitative and qualitative elements
- Provides a more comprehensive picture of a situation or problem
- Quantitative - numbers and measurements
- Qualitative - qualities and attributes

Quantitative Data	vs. Qualitative Data
• Deals with numbers and measurable quantities	• Deals with descriptive qualities and attributes
• How much or how many?	• What kind or what category?
• Examples:	• Examples:
◦ Age	◦ Gender
◦ Height	◦ Color
◦ Income	◦ Job Type

Qualitative Data: Two Types		
Nominal Data	vs.	Ordinal Data
• Categorizes things into different groups or labels		• Has categories with a specific order or rank
• Does not imply order or rank		• Examples:
• Examples:		◦ High school degree
◦ Colors		◦ Bachelor's degree
◦ Animal Species		◦ Master's degree
◦ Types of fruits		• Has a clear hierarchy

Quantitative Data: Two Types		
Interval Data	vs.	Ratio Data
• Has a consistent interval or gap between values		• Has the qualities of interval data
		• Includes a true zero point
• Allows us to quantify and compare differences between two points		• Examples:
		◦ Height
		◦ Weight
		◦ Income

The four levels of data:

1. **Nominal Level:** The nominal level represents data that can be categorized into distinct groups or labels but lacks any inherent order or ranking. It answers the question: "What category?"

Examples:

- Gender (Male, Female, Other)
- Eye color (Blue, Brown, Green)
- Types of cuisine (Italian, Chinese, Indian)

2. **Ordinal Level:** The ordinal level represents data that can be categorized and ranked in a meaningful order, but the intervals between ranks are not uniform or measurable. It answers the question: "What order?"

Examples:

- Education levels (High School, Bachelor's, Master's, PhD)

- Customer satisfaction (Very Unsatisfied, Unsatisfied, Neutral, Satisfied, Very Satisfied)
- Race rankings (1st place, 2nd place, 3rd place)

3. **Interval Level:** The interval level represents data that can be ordered and has equal, meaningful intervals between values, but it lacks a true zero point. It answers the question: "What is the difference?"

Examples:

- Temperature in Celsius or Fahrenheit ($0^{\circ}\text{C} \neq$ no temperature)
- Dates on a calendar (e.g., 2025, 2026)
- IQ scores

4. **Ratio Level:** The ratio level represents data that has all the properties of the interval level, but it also has a true zero point, meaning zero represents the complete absence of the variable. It answers the question: "How much of it?"

Examples:

- Height (e.g., 170 cm, 180 cm)
- Weight (e.g., 60 kg, 75 kg)
- Income (e.g., \$0, \$10,000, \$50,000)
- Time (e.g., 0 seconds, 30 seconds)

3.2 Data preprocessing requirements :

Real-world data are collected from practical sources such as sensors, surveys, social media, databases, or user interactions. Unlike idealized datasets, real-world data often has issues like noise, missing values, or inconsistencies that can negatively impact analysis and decision-making. Because of this we need data preprocessing.

Need for Data Preprocessing

Data preprocessing is a crucial step in preparing raw data for analysis or machine learning tasks. It ensures that the dataset is clean, consistent, and usable.

- Improving Data Quality
- Reduces Noise and Errors
- Handles Missing Values
- Enhances Model Performance
- Compatibility: It converts heterogeneous data into a format suitable for specific tools or models.

Data preprocessing requirements refer to the essential steps and transformations needed to prepare raw data for analysis or modeling. This process addresses issues like inconsistencies, noise, and missing values to ensure data is clean, relevant, and properly formatted for further use.

General Steps of Data Preprocessing:

1. Data Collection
2. Data Cleaning
3. Data Integration
4. Data Transformation and normalization
5. Data Reduction
6. Data enrichment, validation and publishing

Data Collection: Collect the raw data from various sources. The data might come from databases, APIs, web scraping, manual entry, etc.

Data Cleaning: This requirement involves identifying and fixing inaccuracies, inconsistencies, and errors within the data.

It includes:

Handling Missing Values: Filling in or removing empty values that can disrupt analysis and cause errors.

Outlier Management: Detecting and addressing extreme values that may skew results or are the result of errors.

Removing Duplicates: Ensuring each data point is unique and avoiding redundant information that can bias results.

Data Integration: Data integration combines data from multiple sources into a coherent dataset. This is critical in industries where data is collected from various systems, such as customer data from multiple sales channels. Integration ensures that the merged dataset is complete and avoids data redundancy.

Data Transformation Data needs to be reshaped into a format that can be interpreted by analytical tools or machine learning algorithms. It includes:

Normalization: Adjusting data to a common scale (e.g., using min-max scaling or Z-score normalization).

Encoding: Converting categorical data into numerical format (e.g., one-hot encoding).

Feature Engineering: Creating or modifying features to add predictive power, such as extracting the month from a date or combining two features.

Example: In customer data, transforming age into age ranges (e.g., 18-25, 26-35) can improve model interpretability.

Data Reduction: To make data easier to process and analyze, irrelevant or redundant data should be minimized. This step includes:

Dimensionality Reduction: Using techniques like PCA to simplify data by reducing the number of features.

Feature Selection: Choosing only the most relevant features, which can improve model performance and reduce computational costs.

Data Wrangling: Data wrangling involves reshaping and transforming the dataset into the desired format for analysis. This step is often used when datasets are complex or unstructured.

Example: Transforming a dataset that logs timestamps into separate date, time, and year columns to enable easier analysis.

Effective data wrangling ensures that datasets are clean, well-structured, and ready for model input.

Note: Data transformation is a part of data wrangling

3.3 Data sources and collection methods:

Data sources refer to the origins or locations where data can be collected for analysis, research, or decision-making.

Generally, they can be categorized into primary and secondary data sources.

1. Primary Data Sources:

Primary Data Sources refer to data that has been collected from first-hand-sources. Primary data has not been published yet and is more reliable, authentic and objective. Primary data has not been changed or altered by human beings; therefore its validity is greater.

Sources of Primary Data: Sources for primary data are limited and at times it becomes difficult to obtain data from primary source because of either scarcity of population or lack of cooperation.

Characteristics of Primary Data:

- **Original and Firsthand**
- **Time-Specific:** Created at the time of the event or soon after.
- **Authentic and Raw:** Contains unprocessed, unedited, or original information.
- **Subjective:** Reflects the creator's personal perspective or experience.

Following are some of the sources of primary data.

1. Experiments / Case studies
2. Survey/ Field Studies
3. Questionnaire
4. Interview
5. Observations
6. IoT and Sensor Data

Advantages of Using Primary Data

- ✓ The investigator collects data specific to the problem under study.
- ✓ There is no doubt about the quality of the data collected (for the investigator).
- ✓ If required, it may be possible to obtain additional data during the study period.

Disadvantages of Using Primary Data

1. The investigator has to contend with all the hassles of data collection-
 - ✓ deciding why, what, how, when to collect;
 - ✓ getting the data collected (personally or through others);
 - ✓ getting funding and dealing with funding agencies;
 - ✓ ethical considerations (consent, permissions, etc.).
2. Ensuring the data collected is of a high standard-
 - ✓ all desired data is obtained accurately, and in the format it is required in;

- ✓ there is no fake/ cooked up data;
 - ✓ unnecessary/ useless data has not been included.
3. Cost of obtaining the data is often the major expense in studies.

3. Secondary Data Sources:

Secondary Data Sources refer to data that has already been collected, processed, and published by someone else for a purpose other than the current research. This data can be reused by researchers for different studies or purposes. For example, census data might be used to analyze the impact of education on career choices and earnings, even though the original collection was for demographic analysis.

Characteristics of Secondary Data:

- **Created Later:** Produced after the event or study, based on primary sources.
- **Accessible:** Easier to obtain.
- **Synthesize Information:** Secondary sources often combine multiple primary sources or other secondary sources to create a more comprehensive understanding.

Sources of Secondary Data: The following are some ways of collecting secondary data –

1. Books
2. Records
3. Biographies
4. Newspapers
5. Published censuses or other statistical data
6. Data archives
7. Internet articles
8. Research articles by other researchers (journals)
9. Databases, etc.

Advantages of Using Secondary Data

- ✓ No hassles of data collection.
- ✓ It is less expensive.
- ✓ The investigator is not personally responsible for the quality of data ('I didn't do it').

Disadvantages of Using Secondary Data

- ✓ The data collected by the third party may not be a reliable party so the reliability and accuracy of data go down.

- ✓ Data collected in one location may not be suitable for the other one due variable environmental factor.
- ✓ With the passage of time the data becomes obsolete and very old.
- ✓ Secondary data collected can distort the results of the research. For using secondary data a special care is required to amend or modify for use.
- ✓ Secondary data can also raise issues of authenticity and copyright.

For more detailed info:

https://www.researchgate.net/publication/325846997_METHODS_OF_DATA_COLLECTION

Need for Data Collection:

- Data collection ensures informed decision-making by providing factual and reliable information.
- It supports research and development by uncovering trends and validating hypotheses.
- Monitoring and evaluation of processes or outcomes rely on accurate data collection.
- It helps in identifying and solving problems by analyzing root causes.
- Planning and forecasting depend on collected data for accurate predictions and resource allocation.
- Businesses use data collection to gain insights into customer preferences and behavior.
- It ensures compliance with regulations and supports reporting for audits and accountability.
- Data collection provides a competitive advantage by identifying market opportunities and improving strategies.
- It reduces risk and uncertainty by offering insights into potential challenges and opportunities.

Internal, external, and open-source data:

Internal data: Internal data refers to the information generated within an organization or system. It is typically collected from sources such as company records, operations, or transactions.

Characteristics:

- Originates from within the organization.
- Includes data from sales, financials, human resources, or customer databases.

- Provides insights into business performance, operations, and customer behavior.
- Often more accurate, reliable, and tailored to the organization's needs.

Examples:

- Sales transactions
- Employee performance records
- Financial statements
- Internal surveys or feedback

External data: External data comes from sources outside of the organization. It is used to complement internal data, offering insights from the broader environment in which the organization operates.

Characteristics:

- Collected from external entities or third parties.
- Provides a broader context or market conditions.
- Includes publicly available data, government records, or data purchased from vendors.
- Can be used for benchmarking, market analysis, and competitive intelligence.

Examples:

- Government statistics or census data
- Social media data
- Market research reports
- Publicly available data (e.g., from websites, news outlets)

Open-Source data: Open-source data refers to data that is freely available for public use, typically provided by governments, research institutions, or communities.

Characteristics:

- Accessible to anyone without restrictions or cost.
- Often provided in standardized formats for ease of use.
- Can be used for analysis, research, or application development.
- Promotes transparency and innovation through shared resources.

Examples:

- Open government datasets (e.g., data.gov)

- Open data from scientific research communities (e.g., datasets from academic journals)
- Publicly available databases like Wikipedia, open repositories, or Kaggle.

3.4 Data cleaning and preparation:

Data cleaning and preparation are **crucial steps** in the data analysis process. Real-world data tend to be **incomplete, noisy, and inconsistent**. This dirty data can **cause an error** while doing data analysis. Data cleaning is done to handle irrelevant or missing data.

Data cleaning also known as **data cleansing or scrubbing**. They involve identifying and addressing quality issues like **missing data, outliers, and inconsistencies that can skew results and lead to incorrect conclusions**.

Effective data cleaning techniques include handling missing data through **deletion or imputation**, **smoothing** any noisy data, **identifying and removing** outliers, and resolving inconsistencies. Data transformations and scaling methods further refine the dataset, ensuring it's primed for accurate analysis and meaningful insights.

Data Quality Issues and Impact:

Common Data Quality Issues:

- ✓ Missing data occurs when certain values are absent from the dataset caused by data collection issues, system failures, or respondents choosing not to provide information.
- ✓ Outliers are data points that significantly deviate from the majority of the data, often due to measurement errors, data entry mistakes, or genuine extreme values.
- ✓ Inconsistencies in data refer to contradictory or conflicting information within the dataset, such as discrepancies between related variables or violations of logical constraints.
- ✓ Data entry errors and duplicates are other common data quality issues that can affect the accuracy and reliability of the dataset.

Impact of Data Quality Issues

- ✓ Data quality issues can lead to biased or incorrect conclusions, reduced model performance, and ineffective decision-making based on the flawed data
- ✓ Missing data can reduce the representativeness of the sample and introduce bias if not handled appropriately

- ✓ Outliers can distort statistical measures like mean and variance and influence the results of data analysis techniques.
- ✓ Inconsistencies can arise from data integration issues, human errors, or changes in data collection methods over time.
- ✓ Identifying and addressing data quality issues is crucial for ensuring the reliability and credibility of data-driven insights and decisions.

Handling Data Challenges:

1. Handling Missing Data:

- ✓ **Deletion methods** include listwise deletion (removing entire records with missing values) and pairwise deletion (using available data for each analysis). Deletion can lead to reduced sample size and potential bias if missingness is related to the variables of interest.
- ✓ **Imputation methods** estimate missing values based on available data, such as mean imputation, median imputation, or regression imputation. Imputation preserves sample size but may introduce bias if the imputation model is misspecified or if the missing data mechanism is not properly accounted for.
- ✓ Advanced techniques like **multiple imputation** or maximum likelihood estimation can handle missing data more effectively by considering the uncertainty associated with the missing values.

2. Handling Outliers and Inconsistencies:

Outlier detection techniques identify data points that deviate significantly from the majority of the data, such as using statistical measures (Z-scores, interquartile range), visual inspection (box plots, scatter plots), or machine learning algorithms.

So, Handling outliers depends on the context and the nature of the outliers.

- ✓ Options include removing outliers if they are confirmed errors, transforming variables to reduce the impact of outliers (logarithmic transformation, Square root transformation or others), or using robust statistical methods that are less sensitive to outliers (median instead of mean).

Log Transformation:

Example:

Original data: [1, 10, 100, 1000, 10000]

Transformed data: $[\log(1), \log(10), \log(100), \log(1000), \log(10000)] = [0, 1, 2, 3, 4]$

Square root Transformation:

Original data: [1, 4, 9, 16, 25]

Transformed data: $[\sqrt{1}, \sqrt{4}, \sqrt{9}, \sqrt{16}, \sqrt{25}] = [1, 2, 3, 4, 5]$

Mean Sensitive to Outlier than Median:

For example:

- Dataset 1: [10, 12, 14, 15, 100]
 - Mean = $(10 + 12 + 14 + 15 + 100) / 5 = 30.2$
 - Median = 14

Several **Inconsistency resolution** techniques are used to identify and fix contradictory or conflicting information in a dataset. This is crucial because inconsistent data can lead to inaccurate conclusions or model predictions. These techniques typically involve:

- ✓ **Data Validation Rules:** These rules ensure that the data follows expected patterns. For example, you might set a rule that age values must be between 0 and 120, or that a product's price cannot be negative.
- ✓ **Logical Checks:** These checks look for data that doesn't make sense based on relationships between different columns. For example, if a "start date" comes after an "end date," it's a logical inconsistency.
- ✓ **Data Reconciliation:** This method is used when data comes from different sources. If the sources provide conflicting information, reconciliation techniques are used to decide which version is correct, such as choosing the most recent data or averaging conflicting values.

Also, It's also important to document how missing data, outliers, and inconsistencies are handled in your data analysis. This documentation ensures that others can understand and repeat your analysis, and that the steps you took to clean and prepare the data are transparent. This way, everyone can follow the same process and verify the accuracy of the results.

Benefits of Data Cleaning and Preparation:

1. Data cleaning improves the quality of data by ensuring it is accurate, consistent, and reliable, which reduces errors in analysis and decision-making.
2. Clean data leads to more precise insights and better analysis by uncovering meaningful patterns and trends that might be hidden in messy data.
3. Machine learning models perform significantly better when trained on high-quality, cleaned data, resulting in more accurate predictions.
4. Proper data preparation saves time and reduces costs by minimizing the need for corrections later in the process, preventing errors that could lead to costly decisions.
5. It ensures consistency across different datasets, making it easier to combine them for more comprehensive and reliable analysis.
6. Clean data improves decision-making by providing more trustworthy insights, which lead to better strategies and outcomes.
7. Data cleaning ensures compliance with industry regulations, such as GDPR, by addressing data privacy and security concerns.
8. By providing accurate and organized customer data, it helps businesses deliver personalized services and improve the overall customer experience.

Challenges of Data Cleaning and Preparation:

1. **Data inconsistency** across different sources can make it difficult to standardize data for analysis.
2. **Missing or incomplete data** can be problematic, as improper handling of gaps can lead to biased analysis or the loss of valuable information.
3. Identifying and removing **duplicate or redundant data** can be a time-consuming task that is critical for ensuring accurate analysis.
4. The **variety of data types**, such as numerical, categorical, and textual, **complicates** the cleaning process as each type requires different techniques.

5. Data cleaning is often **resource-intensive**, requiring significant time and expertise, especially with large datasets.
6. Ensuring **privacy and security** during data cleaning, particularly when handling sensitive information, is essential to avoid legal issues and breaches.
7. The **lack of standardization** in data formats and naming conventions across multiple systems can create significant challenges in making data uniform.
8. Manual data cleaning is prone to **human error**, which can lead to mistakes, such as accidentally removing important data or making incorrect adjustments.
9. Scaling data cleaning processes to handle large datasets efficiently can be difficult, often requiring automation or advanced tools.
10. **Time constraints** often lead to rushed data cleaning, which may result in incomplete preparation or lower-quality data being used for analysis.

3.5 Data wrangling and associated tools

Data wrangling is a more advanced step where we perform transformations to make data into a structured and usable format ready for analysis or machine learning. This includes reshaping, merging, aggregating, and manipulating the data to derive the desired results.

1. Reshaping Data:

✓ **Pivoting and Unpivoting:**

Pivoting: Converts unique values from a column into multiple columns,

Date	Product	Sales
2024-01-01	A	100
2024-01-02	A	150
2024-01-01	B	200
2024-01-02	B	250

turning data from a long format into a wide format.

Example:

the above table is converted into

Date	A	B
2024-01-01	100	200
2024-01-02	150	250

This method is called Pivoting.

While, **Unpivoting (or Melting)**: Converts columns into rows, turning wide data into a long format.

Tools like pandas (pivot() and melt()) can be used.

✓ Flattening Hierarchical Data:

If you're working with nested JSON, libraries like json_normalize() in pandas can help to flatten it into a tabular form.

```
python
data = [
    {"id": 1, "name": {"first": "Coleen", "last": "Volk"}},
    {"name": {"given": "Mark", "family": "Regner"}},
    {"id": 2, "name": "Faye Raker"},
]
pd.json_normalize(data)
```

Output:

```
r
      id name.first name.last name.given name.family      name
0  1.0    Coleen     Volk      NaN      NaN      NaN
1  NaN      NaN      NaN      Mark     Regner      NaN
2  2.0      NaN      NaN      NaN      NaN  Faye Raker
```

2. Data Merging and Joining:

Inner Join, Outer Join, Left Join, Right Join

- ✓ Inner Join: Combines rows where there are matching values in both datasets.
- ✓ Outer Join: Includes all rows from both datasets, filling in missing values with NULL.
- ✓ Left Join: Includes all rows from the left dataset, and matching rows from the right dataset.

- ✓ Right Join: Includes all rows from the right dataset, and matching rows from the left dataset.

Tools: Pandas merge(), SQL JOIN operations.

Example:

- Dataset 1 (Customers):

CustomerID	Name
1	Alice
2	Bob
3	Charlie

- Dataset 2 (Orders):

OrderID	CustomerID	Product
101	1	Laptop
102	2	Phone
103	4	Tablet

Inner Join on CustomerID :

- Result:

CustomerID	Name	OrderID	Product
1	Alice	101	Laptop
2	Bob	102	Phone

- Left Join on CustomerID :

- Result:

CustomerID	Name	OrderID	Product
1	Alice	101	Laptop
2	Bob	102	Phone
3	Charlie	NULL	NULL

- **Outer Join on CustomerID :**

- **Result:**

CustomerID	Name	OrderID	Product
1	Alice	101	Laptop
2	Bob	102	Phone
3	Charlie	NULL	NULL
4	NULL	103	Tablet

- **Right Join on CustomerID :**

- **Result:**

CustomerID	Name	OrderID	Product
1	Alice	101	Laptop
2	Bob	102	Phone
4	NULL	103	Tablet

3. Aggregating Data:

To aggregate data in pandas,

- ✓ you typically use methods like `groupby()`, `add()`, or `pivot_table()` depending on the kind of aggregation you want to perform.

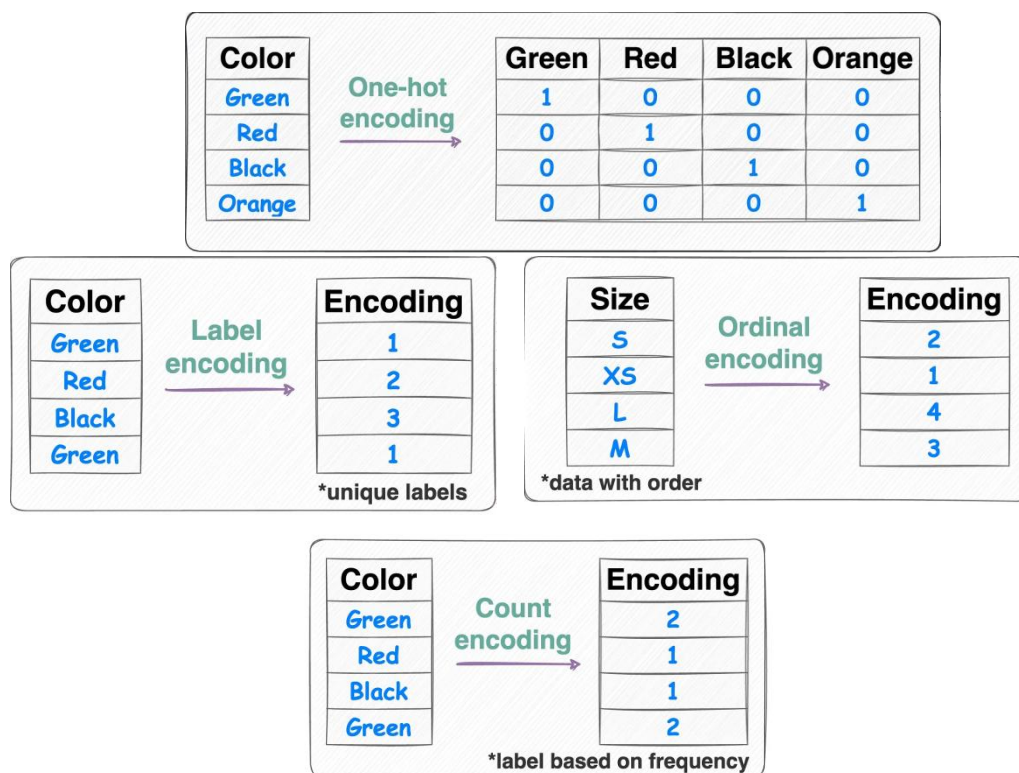
While in SQL,

- ✓ Use `GROUP BY` and aggregation functions like `AVG()`, `SUM()`, and `COUNT()` in SQL queries.

4. Handling Categorical Data:

We can use several methods to handle categorical data,

- ✓ One-hot encoding transforms categorical values into a binary vector (0s and 1s) where each category is represented by a unique column.
- ✓ Label encoding assign each category a unique label.
- ✓ Ordinal encoding Similar to label encoding — assign a unique integer value to each category but they have an inherent order.
- ✓ Count encoding encodes categorical features based on the frequency of each category.



For more info: <https://blog.dailydoseofds.com/p/7-must-know-techniques-for-encoding-017>

Associated Tools:

- ✓ **Pandas (Python):** Offers a wide range of data manipulation capabilities such as merging, reshaping, aggregating, etc.
- ✓ **dplyr (R):** Provides a simple syntax for data manipulation, allowing operations like `mutate()`, `select()`, and `summarize()`.
- ✓ **SQL:** Useful for joining tables, filtering data, and performing complex aggregation.
- ✓ **OpenRefine:** An open-source tool that facilitates data wrangling with a user-friendly interface for cleaning, transforming, and integrating data.

3.6 Data enrichment, validation and publishing:

Data enrichment is the process of enhancing or augmenting your existing dataset with additional relevant information from external or internal sources. The goal is to improve the quality and context of the data by adding new features, attributes, or metadata.

Methods:

- ✓ **External Data Sources:** Enriching the dataset with external data such as public APIs, third-party datasets (weather data, social media data, demographic data, etc.), or commercial datasets (e.g., customer data from external sources).
- ✓ **Data Fusion/ Crowdsourcing :** Combining multiple datasets from internal or external sources to create a more comprehensive dataset. Example: Merging transactional data with customer data to provide richer insights.
- ✓ **Data Augmentation:** Generate synthetic data by transforming or creating new data points, such as rotating images or generating new sentences for NLP tasks.
- ✓ **Feature Engineering:** Deriving new features from the existing data based on domain knowledge or advanced techniques. Example: From a timestamp column, you could create new features like "day of the week," "month," or "holiday indicator."
- ✓ **Web Scraping:** Pulling additional information from websites to supplement the dataset. Example: Scraping product reviews from an e-commerce website to add sentiment or popularity metrics to a product dataset.

Tools:

- ✓ APIs (e.g., Google Maps API, Twitter API)
- ✓ Web scraping libraries (e.g., BeautifulSoup, Scrapy)
- ✓ Data integration tools.

Use Cases in Various Sectors:

- **Financial Sector:** Enriching customer data with credit scores, financial history, or social media activity to assess risk or personalize offers.
- **Social Media Data:** Enhancing user profiles with demographic data, interests, and behavioral data for more targeted advertising and content recommendations.
- **Customer Data:** Adding behavioral data, purchase history, or customer service interactions to build a 360-degree view of the customer for personalized marketing.
- **E-commerce:** Enriching product data with reviews, user ratings, or competitor data to improve product recommendations and optimize pricing strategies.

Data Validation refers to the process of ensuring that the data meets specific quality standards and is consistent, accurate, and meaningful. Validation

helps to identify errors, inconsistencies, or invalid entries before data is used for analysis or modeling.

Techniques:

- ✓ **Schema Validation:** Ensuring that the data adheres to the expected structure (correct column names, data types, and formats). This is particularly useful for structured data.
Example: Checking if a Date field contains valid dates or if a Price field contains only numeric values.
- ✓ **Range and Constraint Validation:** Ensuring that data falls within acceptable ranges or adheres to predefined constraints.
Example: Ensuring that an Age field is within a valid range (e.g., 0 to 120), or that a Salary field is not negative.
- ✓ **Cross-Field Validation:** Compare values across different fields (e.g., a column that should have a certain relationship with another column).
Example: Ensure that a Discount column is never greater than the Price column.
- ✓ **Duplicate Detection:** Identifying and removing duplicate rows or records to maintain data integrity.
Example: Removing multiple entries for the same person or transaction ID.
- ✓ **Outlier Detection:** Identifying extreme values that deviate significantly from the rest of the data and may represent errors or anomalies.
Example: Flagging a salary of \$1,000,000 for an entry where the typical salary is on range of \$25000 to \$30,000.

Data publishing is the process of preparing and distributing the data for external use or making it available for analysis, reporting, or sharing with stakeholders. It involves making data accessible in formats that can be used by different systems or users.

Key Steps for consideration:

- ✓ **Format Conversion:** Converting the data into an appropriate format for the target audience or system. Common formats include:
CSV, JSON, or Parquet for tabular data.
APIs or dashboards for real-time access.

Excel or PDF for reports.

- ✓ **Data Documentation:** Creating documentation or metadata for the dataset to describe its structure, meaning, and limitations. This is critical for ensuring that users understand how to interpret the data.
Example: A data dictionary explaining the columns and their meanings.
- ✓ **Data Security and Privacy:** Ensuring that sensitive information is protected. This may involve anonymizing or aggregating data before sharing, especially for personal or confidential information.
Example: Removing personally identifiable information (PII) like Social Security numbers.
- ✓ **Access Control:** Deciding who has access to the data and enforcing policies to prevent unauthorized access.
Example: Using role-based access controls (RBAC) or authentication for APIs.
- ✓ **Data Publishing Platforms:** Making the data available via platforms such as:
 1. Public data repositories: (e.g., Kaggle, Huggingface, github).
 2. APIs: Providing data access via a REST API for integration with external applications. This is common for real-time data updates, such as stock prices or social media feeds.
 3. Data Portals: Creating centralized repositories where datasets are stored and made available for download or viewing. Often used by governments, research institutions, or public organizations.
 4. Cloud Platforms: Using cloud services like AWS, Google Cloud, or Azure to store and share datasets securely across teams or with third parties.
 5. Reports and Dashboards: Sharing insights through automated reports or visual dashboards that summarize key findings from the data. Common in business intelligence platforms.

In conclusion, Proper enrichment adds value, validation ensures correctness, and publishing facilitates easy access and use of the data.

3.7 Data transformation and normalization

Data transformation refers to changing the format, structure, or values of data to improve its quality or make it easier for algorithms to process.

Common transformations include:

1. Logarithmic Transformation:

The log transformation helps in:

- ✓ Reducing Skewness: It makes highly skewed data (e.g., income, population) more symmetrical.
- ✓ Compressing Large Values: It "compresses" high values, making them closer to smaller values. This can make analysis and visualization easier.

Formula:

For any value x , the transformed value x' is given by:

$$x' = \log(x + 1)$$

Adding $+1$ is a common adjustment to handle cases where $x = 0$, as the logarithm of zero is undefined.

Imagine we have an income dataset where values range from low (e.g., \$10,000) to high (e.g., \$1,000,000). Without a transformation, these high incomes could skew the analysis.

Original Income	Transformed Income ($\log(x + 1)$)
10,000	9.21
100,000	11.51
1,000,000	13.82

As seen, the gap between high and low incomes shrinks, making the data more balanced for analysis. This helps reduce the influence of outliers and provides a clearer view of overall trends in the data.

2. Square Root Transformation:

The square root transformation is a technique commonly used to handle count data or positively skewed data. By applying this transformation, you can stabilize the variance and make the data distribution more symmetric, which is often useful in statistical analysis and modeling.

Let's use Transactions dataset: Transactions=[1,4,9,16,25]

Step 1: Calculate the mean and variance before transformation.

- Mean:

$$\mu = \frac{1 + 4 + 9 + 16 + 25}{5} = \frac{55}{5} = 11$$

- Variance:

$$\sigma^2 = \frac{(1 - 11)^2 + (4 - 11)^2 + (9 - 11)^2 + (16 - 11)^2 + (25 - 11)^2}{5} = \frac{100 + 49 + 4 + 25 + 196}{5} = \frac{374}{5} = 74.8$$

Step 2: Apply the square root transformation:

$$\text{Square Root Transformed Transactions} = [\sqrt{1}, \sqrt{4}, \sqrt{9}, \sqrt{16}, \sqrt{25}] = [1, 2, 3, 4, 5]$$

Step 3: Calculate the mean and variance after transformation.

- Mean:

$$\mu' = \frac{1 + 2 + 3 + 4 + 5}{5} = \frac{15}{5} = 3$$

- Variance:

$$\sigma'^2 = \frac{(1 - 3)^2 + (2 - 3)^2 + (3 - 3)^2 + (4 - 3)^2 + (5 - 3)^2}{5} = \frac{4 + 1 + 0 + 1 + 4}{5} = \frac{10}{5} = 2$$

Result:

- **Before Transformation:**

- Mean = 11
- Variance = 74.8

- **After Transformation:**

- Mean = 3
- Variance = 2

The variance has decreased significantly, from 74.8 to 2. This stabilization happens because the square root transformation compresses the higher values and spreads out the lower values less, making the dataset less volatile and more stable.

3. Binning:

Binning is a technique that converts continuous data into categorical data by grouping values into bins or intervals. This can simplify the data and make models less sensitive to small fluctuations in the data, helping to highlight broader patterns or trends.

Consider a dataset with customer ages that are continuous (e.g., 23, 29, 35, 41, 55, etc.). Instead of treating each specific age as a separate value, you can group them into age bins (ranges). This can make analysis easier by categorizing people into age groups.

Customer ID	Age	Binned Age Group
1	23	18-25
2	30	26-35
3	35	26-35
4	41	36-45
5	50	46-55
6	55	46-55

Now, rather than using the exact age as a feature, the model uses the binned age group, making it easier to analyze patterns within broader age categories.

4. **Encoding Categorical Variables:** This process converts categorical data into numerical form, necessary for many algorithms. eg. One hot encoding, Label encoding, etc. (Described above)

Normalization:

Data normalization is the process of scaling or adjusting the data values to a common range or distribution, without changing their relative meaning or order. For example, you might normalize data from 0 to 100, or from -1 to 1, or to a standard normal distribution with mean 0 and standard deviation 1. Data normalization can help to reduce the impact of outliers, skewness, and scale differences on the data analysis.

Importance of Normalization

1. Improves Model Performance:

Many machine learning algorithms, particularly those that rely on distance calculations (e.g., k-NN, k-means, and SVM), perform better when the data is normalized. This ensures that features contribute equally to the model's performance, preventing bias toward features with larger numerical ranges.

2. Faster Convergence in Gradient-Based Algorithms:

Algorithms like gradient descent (used in neural networks, linear regression, etc.) converge faster when features are normalized. It helps in reducing the oscillations during the learning process, leading to more stable and efficient optimization.

3. Ensures Equal Weight for All Features:

Without normalization, features with larger numeric ranges can dominate the model's learning process, leading to an imbalance. Normalization ensures

that all features have an equal weight, making the model more accurate.

4. Prevents Numerical Instability:

When data has large values, it can cause numerical instability during calculations, especially in algorithms that involve matrix operations or inverse calculations. Normalizing the data reduces the risk of such instability.

5. Improves Interpretability:

In some cases, normalized data makes it easier to compare features on the same scale, especially when working with models where coefficients or weights are important, like in linear regression.

1. Min-Max Normalization:

This technique scales the data to a specific range, typically [0, 1].

Formula:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Given a dataset of house prices in thousands:

Prices=[100,200,300,400,500]

Applying Min-Max normalization to scale between [0,1]:

$$\text{Min-Max Normalized Prices} = \frac{\text{Price} - \min}{\max - \min}$$

Calculating:

$$\text{Min-Max Normalized Prices} = [0.0, 0.25, 0.5, 0.75, 1.0]$$

The house prices are now on a scale of [0,1], preserving relative relationships while making data comparable across features.

2. Standardization (Z-score Scaling):

Standardization (Z-score Scaling) is a technique used to scale data so that it has a mean of 0 and a standard deviation of 1. This is useful when you need to compare features with different units or scales, and it's often used when the data is approximately normally distributed.

The formula for standardization (Z-score scaling) is:

$$x' = \frac{x - \mu}{\sigma}$$

where,

x is the original value, μ is the mean of the feature, σ is the standard deviation of the feature, x' is the standardized value.

Suppose you have the following test scores for 5 students:

Student	Test Score (x)
1	45
2	50
3	55
4	60
5	65

By Calculating: Mean = 55 and standard deviation = 7.07

After performing standardization:

Student	Test Score (x)	Standardized Score (x')
1	45	-1.41
2	50	-0.71
3	55	0
4	60	0.71
5	65	1.41

The test scores now have a mean of 0 and a standard deviation of 1. This transformation ensures that the data is centered and scaled, making it easier to compare the scores or apply machine learning models that require features to be on the same scale.

3. Robust Scaling:

Robust Scaling is a technique for scaling features in a dataset that is less sensitive to outliers than traditional scaling methods, like Min-Max normalization or Z-score standardization. Instead of using the mean and standard deviation, which can be influenced by outliers, Robust Scaling uses the median and the interquartile range (IQR) to scale the data.

The formula for Robust Scaling is:

$$x' = \frac{x - \text{median}(x)}{\text{IQR}(x)}$$

where, x is the original value, $\text{median}(x)$ is the median of the feature, $\text{IQR}(x)$ is the interquartile range, calculated as $\text{IQR}(x) = Q3(x) - Q1(x)$, where $Q3$ is the third quartile and $Q1$ is the first quartile, and x' is the scaled value.

Consider test score of 5 student with one extremely high test score (1000).
5,15,20,25,30,1000

calculating we get Median= 22.5, $Q1=15$ and $Q3=30$.

also, $\text{IQR} = Q3 - Q1 = 30 - 15 = 15$

Now, we apply the formula to standardize each score:

Student	Test Score (x)	Scaled Test Score (x')
1	5	-1.17
2	15	-0.5
3	20	-0.17
4	25	0.17
5	30	0.5
6	1000	65.17

Student 6 (with the score of 1000) has been scaled to 65.17 in the transformed data. Despite being an extreme outlier, it is now much more proportionate compared to the other students' scaled values.

Robust Scaling reduces the impact of outliers. The median (22.5) and IQR (15) were used, so even if there are extreme values (like 1000), they do not heavily distort the scaling of the rest of the data.

In conclusion, The choice of normalization technique depends on the nature of the data and the requirements of the analysis algorithm.

Assignment:

**You have the following dataset representing the ages of 10 individuals:
{30,45,50,55,60,65,77,90,100,110}**

- Apply Min-Max normalization to scale these values to the range [0, 1].**
- Apply Z-Score normalization (standardization) to these values.**
- Apply Robust Scaling to these values.**

3.8 Dimensionality reduction linear factor model, principal component analysis (PCA)

Data Dimensionality refers to the number of features (variables or attributes) present in a dataset. As the number of features increases, so does the complexity of the data, which can make analysis more difficult.

The number of input features, variables, or columns present in a given dataset is known as **dimensionality**, and the process to reduce these features is called **dimensionality reduction**.

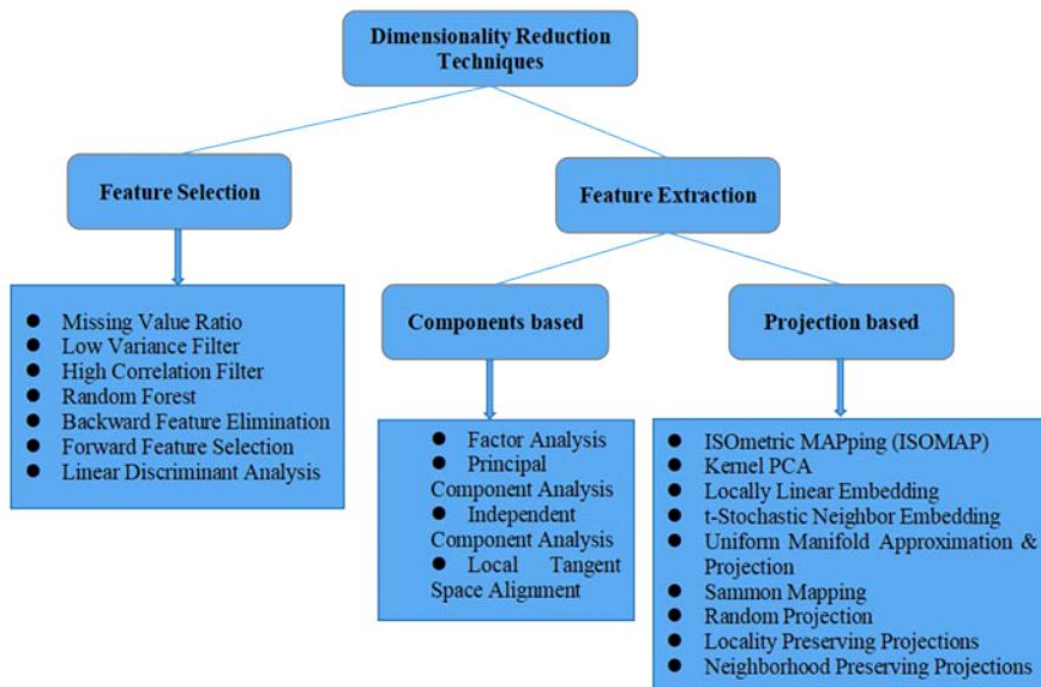
Curse of Dimensionality is a phenomenon that arises when the number of features (dimensions) increases in a dataset. It makes traditional machine learning algorithms less effective due to the exponential increase in volume of the feature space. As dimensionality grows:

1. Data points become sparse.
2. The volume of the space increases exponentially, diluting the density of available data.
3. Distance metrics (used in clustering, classification, etc.) become less meaningful.
4. It increases computational complexity.

A dataset contains a huge number of input features in various cases, which makes the predictive modeling task more complicated. Because it is very difficult to visualize or make predictions for the training dataset with a high number of features, for such cases, dimensionality reduction techniques are required to use.

Dimensionality reduction technique can be defined as, "It is a way of converting the higher dimensions dataset into lesser dimensions dataset ensuring that it provides similar information."

Dimensionality reduction may be implemented in two ways.



1. Feature selection:

In feature selection, we are interested in finding k of the total of n features that give us the most information and we discard the other $(n-k)$ dimensions.

2. Feature extraction:

In feature extraction, we are interested in finding a new set of k features that are the combination of the original n features. These methods may be supervised or unsupervised depending on whether or not they use the output information. The best known and most widely used feature extraction methods are Principal Components Analysis (PCA) and Linear Discriminant Analysis (LDA), which are both linear projection methods, unsupervised and supervised respectively.

Linear Factor Model (LFM)

The Linear Factor Model is a way to explain a set of observed data (like a collection of measurements or features) as a combination of underlying factors that we can't directly observe.

Imagine you have several pieces of data, like the height and weight of a person. The Linear Factor Model assumes that these observations (height, weight) are driven by some underlying factors (such as genetics, diet, exercise). These factors aren't directly measurable, but they influence the data.

The formula for the Linear Factor Model looks like this:

$$X = \Lambda F + \epsilon$$

Where, .

- X is the observed data (e.g., height and weight).
- Λ is a matrix of factor loadings—this shows how much each factor influences the observed data.
- F is a set of latent factors—these are the unobserved variables that drive the data (e.g., genetics, diet).
- ϵ is the error term, representing anything in the data not explained by the factors.

Introduction to Principal component analysis

Principal component analysis (PCA) is a method used to reduce the number of dimensions (features) in data while keeping as much information as possible.

Let's say you have a large dataset with many features (e.g., height, weight, age, income, etc.). PCA helps you reduce the number of features by finding new axes (directions) where the data has the most variance (spread).

- The first new axis (called the first principal component) is the direction where the data has the most variance (spread).
 - The second axis (the second principal component) is perpendicular to the first and captures the next largest spread in the data.
- and so on.

Connection Between the Linear Factor Model and PCA

1. Similar Goal: Both PCA and the Linear Factor Model aim to reduce the complexity of the data by capturing the most important structure. In a Linear Factor Model, the goal is to explain the data using fewer underlying factors. In PCA, the goal is to reduce dimensions by finding the directions that capture the most variance in the data.

2. Principal Components and Latent Factors:

- In the Linear Factor Model, the latent factors are the hidden or unobserved variables that you assume explain the relationships between the observed variables.
- In PCA, the principal components are like these latent factors. They are new variables that are derived from the data, and they represent directions of maximum variance in the data.

3. Factor Loadings and Eigenvectors:

- The factor loadings in the Linear Factor Model tell you how much each latent factor influences the observed data.
- In PCA, the eigenvectors (which define the principal components) act similarly to factor loadings, as they tell you how each original feature contributes to the new components.

4. Error Term in the Linear Factor Model vs. Residual Variance in PCA:

- In the Linear Factor Model, the error term (ϵ) represents the part of the data that is not explained by the latent factors.
- In PCA, after projecting the data onto the principal components, the remaining variance (not captured by the first few components) can be thought of as the "residual variance," similar to the error term.

Hence, PCA can be seen as a special case of the Linear Factor Model.

PCA algorithm

Step 1. Data

We consider a dataset having n features or variables denoted by X_1, X_2, \dots, X_n . Let there be N examples. Let the values of the i -th feature X_i be $X_{i1}, X_{i2}, \dots, X_{iN}$ (see Table 4.1).

Features	Example 1	Example 2	...	Example N
X_1	X_{11}	X_{12}	...	X_{1N}
X_2	X_{21}	X_{22}	...	X_{2N}
\vdots				
X_i	X_{i1}	X_{i2}	...	X_{iN}
\vdots				
X_n	X_{n1}	X_{n2}	...	X_{nN}

Table 4.1: Data for PCA algorithm

Step 2. Compute the means of the variables

We compute the mean \bar{X}_i of the variable X_i :

$$\bar{X}_i = \frac{1}{N} (X_{i1} + X_{i2} + \dots + X_{iN}).$$

Step 3. Calculate the covariance matrix

Consider the variables X_i and X_j (i and j need not be different). The covariance of the ordered pair (X_i, X_j) is defined as¹

$$\text{Cov}(X_i, X_j) = \frac{1}{N-1} \sum_{k=1}^N (X_{ik} - \bar{X}_i)(X_{jk} - \bar{X}_j). \quad (4.1)$$

We calculate the following $n \times n$ matrix S called the covariance matrix of the data. The element in the i -th row j -th column is the covariance $\text{Cov}(X_i, X_j)$:

$$S = \begin{bmatrix} \text{Cov}(X_1, X_1) & \text{Cov}(X_1, X_2) & \dots & \text{Cov}(X_1, X_n) \\ \text{Cov}(X_2, X_1) & \text{Cov}(X_2, X_2) & \dots & \text{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(X_n, X_1) & \text{Cov}(X_n, X_2) & \dots & \text{Cov}(X_n, X_n) \end{bmatrix}$$

Step 4. Calculate the eigenvalues and eigenvectors of the covariance matrix

Let S be the covariance matrix and let I be the identity matrix having the same dimension as the dimension of S .

i) Set up the equation:

$$\det(S - \lambda I) = 0. \quad (4.2)$$

This is a polynomial equation of degree n in λ . It has n real roots (some of the roots may be repeated) and these roots are the eigenvalues of S . We find the n roots $\lambda_1, \lambda_2, \dots, \lambda_n$ of Eq. (4.2).

- ii) If $\lambda = \lambda'$ is an eigenvalue, then the corresponding eigenvector is a vector

$$U = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}$$

such that

$$(S - \lambda' I)U = 0.$$

(This is a system of n homogeneous linear equations in u_1, u_2, \dots, u_n and it always has a nontrivial solution.) We next find a set of n orthogonal eigenvectors U_1, U_2, \dots, U_n such that U_i is an eigenvector corresponding to λ_i .²

- iii) We now normalise the eigenvectors. Given any vector X we normalise it by dividing X by its length. The length (or, the norm) of the vector

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

is defined as

$$\|X\| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}.$$

Given any eigenvector U , the corresponding normalised eigenvector is computed as

$$\frac{1}{\|U\|}U.$$

We compute the n normalised eigenvectors e_1, e_2, \dots, e_n by

$$e_i = \frac{1}{\|U_i\|}U_i, \quad i = 1, 2, \dots, n.$$

Step 5. Derive new data set

Order the eigenvalues from highest to lowest. The unit eigenvector corresponding to the largest eigenvalue is the first principal component. The unit eigenvector corresponding to the next highest eigenvalue is the second principal component, and so on.

- Let the eigenvalues in descending order be $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and let the corresponding unit eigenvectors be e_1, e_2, \dots, e_n .
- Choose a positive integer p such that $1 \leq p \leq n$.
- Choose the eigenvectors corresponding to the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_p$ and form the following $p \times n$ matrix (we write the eigenvectors as row vectors):

$$F = \begin{bmatrix} e_1^T \\ e_2^T \\ \vdots \\ e_p^T \end{bmatrix},$$

where T in the superscript denotes the transpose.

iv) We form the following $n \times N$ matrix:

$$X = \begin{bmatrix} X_{11} - \bar{X}_1 & X_{12} - \bar{X}_1 & \cdots & X_{1N} - \bar{X}_1 \\ X_{21} - \bar{X}_2 & X_{22} - \bar{X}_2 & \cdots & X_{2N} - \bar{X}_2 \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} - \bar{X}_n & X_{n2} - \bar{X}_n & \cdots & X_{nN} - \bar{X}_n \end{bmatrix}$$

v) Next compute the matrix:

$$X_{\text{new}} = FX.$$

Note that this is a $p \times N$ matrix. This gives us a dataset of N samples having p features.

Step 6. New dataset

The matrix X_{new} is the new dataset. Each row of this matrix represents the values of a feature. Since there are only p rows, the new dataset has only features.

Step 7. Conclusion

This is how the principal component analysis helps us in dimensional reduction of the dataset. Note that it is not possible to get back the original n -dimensional dataset from the new dataset.

Problem definition:

Given the data in Table, reduce the dimension from 2 to 1 using the Principal Component Analysis (PCA) algorithm.

Feature	Example 1	Example 2	Example 3	Example 4
X_1	4	8	13	7
X_2	11	4	5	14

Step1: Calculate mean:

$$\bar{X}_1 = \frac{1}{4}(4 + 8 + 13 + 7) = 8,$$

$$\bar{X}_2 = \frac{1}{4}(11 + 4 + 5 + 14) = 8.5.$$

Step2: Calculation of the covariance matrix

$$\text{Cov}(X_i, X_j) = \frac{1}{N-1} \sum_{k=1}^N (X_{ik} - \bar{X}_i)(X_{jk} - \bar{X}_j).$$

$$S = \begin{bmatrix} \text{Cov}(X_1, X_1) & \text{Cov}(X_1, X_2) & \cdots & \text{Cov}(X_1, X_n) \\ \text{Cov}(X_2, X_1) & \text{Cov}(X_2, X_2) & \cdots & \text{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(X_n, X_1) & \text{Cov}(X_n, X_2) & \cdots & \text{Cov}(X_n, X_n) \end{bmatrix}$$

$$\begin{aligned}\text{Cov}(X_1, X_1) &= \frac{1}{N-1} \sum_{k=1}^N (X_{1k} - \bar{X}_1)^2 \\ &= \frac{1}{3} ((4-8)^2 + (8-8)^2 + (13-8)^2 + (7-8)^2) \\ &= 14\end{aligned}$$

$$\begin{aligned}\text{Cov}(X_1, X_2) &= \frac{1}{N-1} \sum_{k=1}^N (X_{1k} - \bar{X}_1)(X_{2k} - \bar{X}_2) \\ &= \frac{1}{3} ((4-8)(11-8.5) + (8-8)(4-8.5) \\ &\quad + (13-8)(5-8.5) + (7-8)(14-8.5)) \\ &= -11\end{aligned}$$

$$\begin{aligned}\text{Cov}(X_2, X_1) &= \text{Cov}(X_1, X_2) \\ &= -11\end{aligned}$$

$$\begin{aligned}\text{Cov}(X_2, X_2) &= \frac{1}{N-1} \sum_{k=1}^N (X_{2k} - \bar{X}_2)^2 \\ &= \frac{1}{3} ((11-8.5)^2 + (4-8.5)^2 + (5-8.5)^2 + (14-8.5)^2) \\ &= 23\end{aligned}$$

$$\begin{aligned}S &= \begin{bmatrix} \text{Cov}(X_1, X_1) & \text{Cov}(X_1, X_2) \\ \text{Cov}(X_2, X_1) & \text{Cov}(X_2, X_2) \end{bmatrix} \\ &= \begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix}\end{aligned}$$

Step 3: Eigenvalues of the covariance matrix

The characteristic equation of the covariance matrix is,

$$\begin{aligned}0 &= \det(S - \lambda I) \\ &= \begin{vmatrix} 14 - \lambda & -11 \\ -11 & 23 - \lambda \end{vmatrix} \\ &= (14 - \lambda)(23 - \lambda) - (-11) \times (-11) \\ &= \lambda^2 - 37\lambda + 201\end{aligned}$$

Solving the characteristic equation we get,

$$\begin{aligned}\lambda &= \frac{1}{2}(37 \pm \sqrt{565}) \\ &= 30.3849, 6.6151 \\ &= \lambda_1, \lambda_2 \quad (\text{say})\end{aligned}$$

Step 4: Computation of the eigenvectors

To find the first principal components, we need only compute the eigenvector corresponding to the largest eigenvalue. In the present example, the largest eigenvalue is λ_1 and so we compute the eigenvector corresponding to λ_1 .

The eigenvector corresponding to $\lambda = \lambda_1$ is a vector

$$U = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

satisfying the following equation:

$$\begin{aligned} \begin{bmatrix} 0 \\ 0 \end{bmatrix} &= (S - \lambda_1 I) \cdot U \\ &= \begin{bmatrix} 14 - \lambda_1 & -11 \\ -11 & 23 - \lambda_1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \\ &= \begin{bmatrix} (14 - \lambda_1)u_1 - 11u_2 \\ -11u_1 + (23 - \lambda_1)u_2 \end{bmatrix} \end{aligned}$$

This is equivalent to the following two equations:

$$\begin{aligned} (14 - \lambda_1)u_1 - 11u_2 &= 0 \\ -11u_1 + (23 - \lambda_1)u_2 &= 0 \end{aligned}$$

Using the theory of systems of linear equations, we note that these equations are not independent and solutions are given by,

$$\frac{u_1}{11} = \frac{u_2}{14 - \lambda_1} = t,$$

$$u_1 = 11t, \quad u_2 = (14 - \lambda_1)t,$$

where, t is any real number.

Taking t = 1, we get an eigenvector corresponding to λ_1 as

$$U_1 = \begin{bmatrix} 11 \\ 14 - \lambda_1 \end{bmatrix}.$$

To find a unit eigenvector, we compute the length of U_1 which is given by,

$$\begin{aligned} \|U_1\| &= \sqrt{11^2 + (14 - \lambda_1)^2} \\ &= \sqrt{11^2 + (14 - 30.3849)^2} \\ &= 19.7348 \end{aligned}$$

Therefore, a unit eigenvector corresponding to λ_1 is

$$\begin{aligned}
 e_1 &= \begin{bmatrix} 11/\|U_1\| \\ (14 - \lambda_1)/\|U_1\| \end{bmatrix} \\
 &= \begin{bmatrix} 11/19.7348 \\ (14 - 30.3849)/19.7348 \end{bmatrix} \\
 &= \begin{bmatrix} 0.5574 \\ -0.8303 \end{bmatrix}
 \end{aligned}$$

By carrying out similar computations, the unit eigenvector e_2 corresponding to the eigenvalue $\lambda = \lambda_2$ can be shown to be,

$$e_2 = \begin{bmatrix} 0.8303 \\ 0.5574 \end{bmatrix}$$

Step 5: Computation of first principal components

Let,

$$\begin{bmatrix} X_{1k} \\ X_{2k} \end{bmatrix}$$

be the k th sample in the above Table (dataset). The first principal component of this example is given by (here “T” denotes the transpose of the matrix)

$$\begin{aligned}
 e_1^T \begin{bmatrix} X_{1k} - \bar{X}_1 \\ X_{2k} - \bar{X}_2 \end{bmatrix} &= \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} X_{1k} - \bar{X}_1 \\ X_{2k} - \bar{X}_2 \end{bmatrix} \\
 &= 0.5574(X_{1k} - \bar{X}_1) - 0.8303(X_{2k} - \bar{X}_2).
 \end{aligned}$$

For example, the first principal component corresponding to the first example

$$\begin{bmatrix} X_{11} \\ X_{21} \end{bmatrix} = \begin{bmatrix} 4 \\ 11 \end{bmatrix}$$

is calculated as follows:

$$\begin{aligned}
 \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} X_{11} - \bar{X}_1 \\ X_{21} - \bar{X}_2 \end{bmatrix} &= 0.5574(X_{11} - \bar{X}_1) - 0.8303(X_{21} - \bar{X}_2) \\
 &= 0.5574(4 - 8) - 0.8303(11 - 8, 5) \\
 &= -4.30535
 \end{aligned}$$

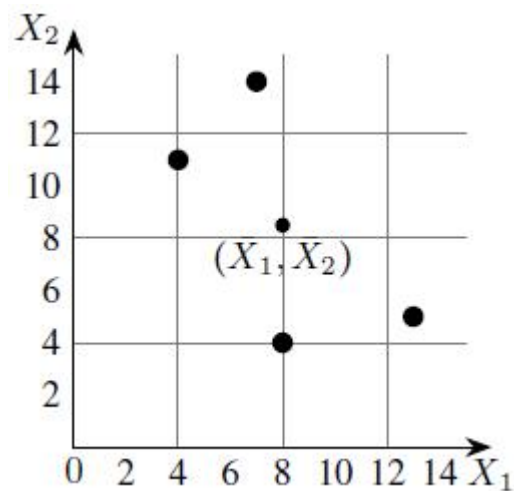
The results of the calculations are summarised in the below Table.

X_1	4	8	13	7
X_2	11	4	5	14
First Principle Components	-4.3052	3.7361	5.6928	-5.1238

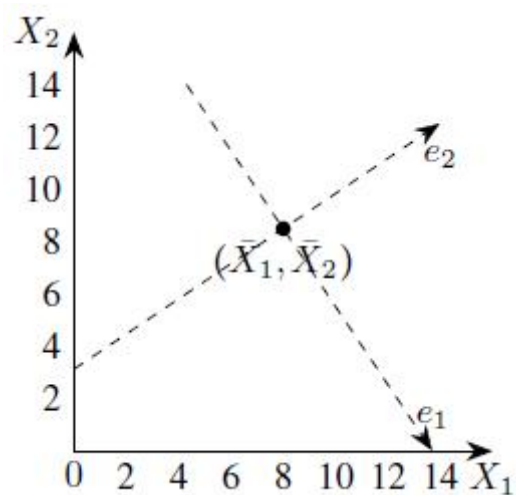
Email: sujan@ioepc.edu.np (for any Feedback)

Geometrical meaning:

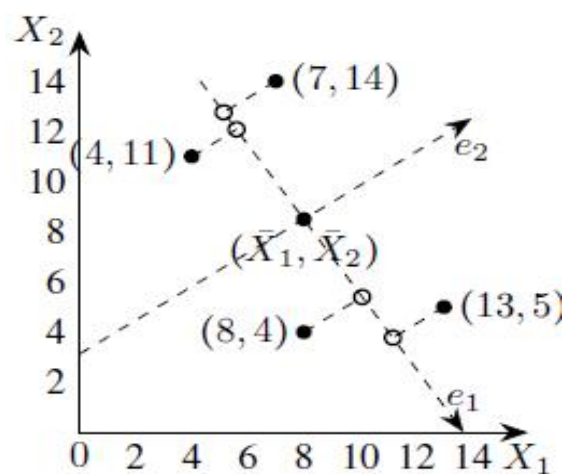
The figure shows the scatter plot of the given data points.



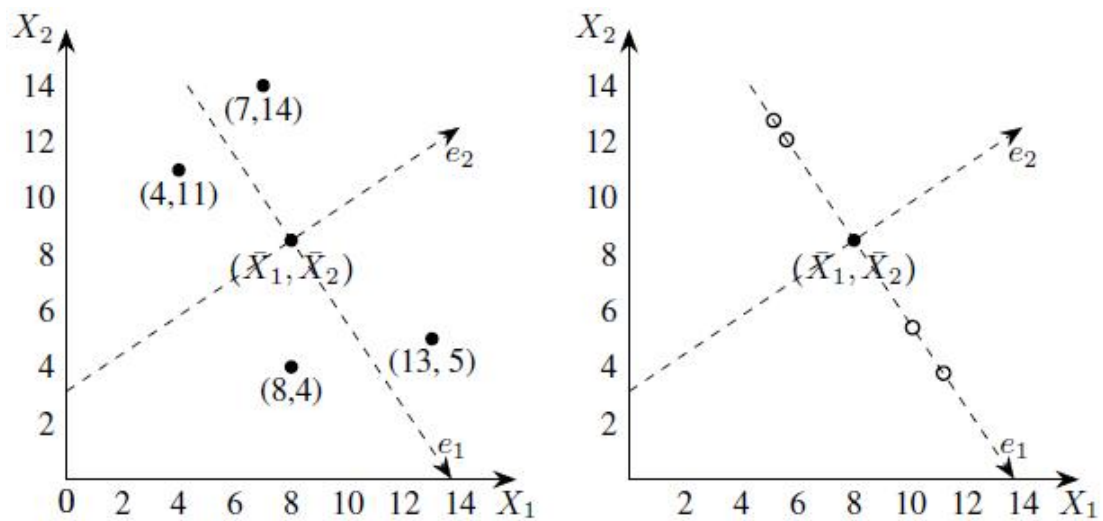
The coordinate system for principal components is as shown below:



Now , Projections of data points on the axis of the first principal component



Finally, two feature are reduced to 1 as shown below:



Assignment: For the purpose of justification and calculation, use the following data, where each row represents a customer and each column represents their preference score for different products and the goal is to reduce the dimensionality of the data (from 3 to 2 dimensions only) while retaining as much information as possible.

Customer ID		1	2	3	4	5
Customer Preferences	Product A	4	5	6	7	8
	Product B	5	4	3	6	7
	Product C	2	3	2	4	5

Application , Advantages and Limitation of PCA

Applications:

1. Data Compression: Data compression is the process of reducing the size of a data set by removing redundant or irrelevant information, while preserving the essential features and quality of the data. Data compression can help you save storage space, improve performance, and speed up processing and transmission of data. PCA can help you achieve data compression by selecting the principal components that explain most of the variance in the data, and discarding the ones that contribute little or nothing. For example, you can use

PCA to compress images by reducing the number of pixels or colors, while maintaining the main characteristics and details of the image.

2. Feature Extraction: Another common and important application of PCA is feature extraction. Feature extraction is the process of transforming or extracting a set of new features from the original data that are more relevant, meaningful, and informative for a specific task or goal. Feature extraction can help you improve the accuracy, efficiency, and interpretability of your data analysis, especially for machine learning and predictive modeling. PCA can help you perform feature extraction by creating principal components that are linear combinations of the original variables, and that are orthogonal and uncorrelated to each other. For example, you can use PCA to extract features that capture the main differences and similarities among groups of customers, products, or markets.

3. Noise Reduction: A less obvious but equally valuable application of PCA is noise reduction. Noise reduction is the process of removing or reducing the unwanted or random variation or error in the data that can affect the quality and reliability of your data analysis. Noise reduction can help you enhance the signal-to-noise ratio, and reveal the true underlying structure and patterns in the data. PCA can help you achieve noise reduction by filtering out the principal components that have low variance and high noise, and retaining the ones that have high variance and low noise. For example, you can use PCA to reduce noise in speech or audio signals by eliminating the components that are irrelevant or insignificant for the sound quality.

4. Data Visualization: A more fun and creative application of PCA is data visualization. Data visualization is the process of presenting and communicating your data in a graphical or visual form that can help you explore, understand, and share your data insights. Data visualization can help you discover new patterns, trends, and relationships in your data, and make your data analysis more engaging and accessible. PCA can help you create data visualizations by reducing the dimensionality of your data to two or three dimensions, which can be easily plotted on a screen or a paper. For example, you can use PCA to visualize complex and high-dimensional data sets such as gene expression, text documents, or social networks.

5. Anomaly Detection: A more advanced and challenging application of PCA is anomaly detection. Anomaly detection is the process of identifying and flagging the data points or observations that deviate significantly from the normal or expected behavior or pattern in the data. Anomaly detection can

help you monitor, detect, and prevent potential problems, risks, or opportunities in your data, such as fraud, errors, defects, or outliers. PCA can help you perform anomaly detection by measuring the distance or deviation of each data point from the principal components, and finding the ones that have unusually large or small values. For example, you can use PCA to detect anomalies in credit card transactions, sensor readings, or network traffic.

Advantages of PCA:

1. **Dimensionality Reduction:** One of the main advantages of PCA is its ability to reduce the dimensionality of the input variables. By transforming a large set of correlated variables into a smaller set of uncorrelated variables (principal components), PCA simplifies the modeling process and improves computational efficiency.
2. PCA removes noise. By reducing the number of dimensions in the data, PCA can help remove noisy and irrelevant features.
3. PCA reduces model training time. By reducing the number of dimensions, PCA simplifies the calculations involved in a model, leading to faster training times. i.e Machine learning algorithms converge faster when trained on principal components instead of the original dataset.
4. Removes Multicollinearity: Multicollinearity occurs when two or more features in a dataset are highly correlated, meaning they provide redundant or overlapping information. This can cause instability in statistical models like regression because it becomes difficult to determine the individual effect of each feature on the target variable. By removing correlations among the features, PCA ensures that multicollinearity is eliminated.
5. PCA enables the visualization of high-dimensional data in 2D or 3D by projecting it onto the first two or three principal components, which capture most of the variance.

Limitation of PCA

1. PCA assumes a correlation between features. If the features are not correlated, PCA will be unable to determine principal components.

2. PCA is sensitive to the scale of the features. Standardization is done before applying PCA.

Imagine we have two features - one takes values between 0 and 1000, while the other takes values between 0 and 1. PCA will be extremely biased towards the first feature being the first principle component, regardless of the actual maximum variance within the data. This is why it's so important to standardize the values first.

3. PCA is not robust against outliers. Similar to the point above, the algorithm will be biased in datasets with strong outliers. This is why it is recommended to remove outliers before performing PCA.

4. PCA assumes a linear relationship between features. The algorithm is not well suited to capturing non-linear relationships. That's why it's advised to turn non-linear features or relationships between features into linear, using the standard methods such as log transforms.

5. Low interpretability of principal components. Principal components are linear combinations of the features from the original data, but they are not as easy to interpret. For example, it is difficult to tell which are the most important features in the dataset after computing principal components.

6. The trade-off between information loss and dimensionality reduction. Although dimensionality reduction is useful, it comes at a cost. Information loss is a necessary part of PCA. Balancing the trade-off between dimensionality reduction and information loss is unfortunately a necessary compromise that we have to make when using PCA.