

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

INTRODUCTION TO DATA PREPROCESSING

CO – 3

Session –
15

AIM



To familiarize the students with the concept of data preprocessing and various data preprocessing techniques.

INSTRUCTIONAL OBJECTIVES



This session is designed to:

1. Demonstrate the importance of data preprocessing.
2. Demonstrate various data preprocessing techniques.

LEARNING OUTCOMES



At the end of this session, students will be able to:

1. Learn the importance of data preprocessing.
2. Learn various data preprocessing techniques.

OUTLINE

- ☐ Why preprocess data?
- ☐ Why data are dirty?
- ☐ Why data preprocessing is important in the machine learning context?
- ☐ Major tasks in data preprocessing
 - Data cleaning
 - Data integration
 - Data transformation
 - Data reduction
 - Data splitting, batching, and shuffling
- ☐ Overfitting and underfitting
- ☐ Summary

WHY PREPROCESS DATA?

- Data in the real world is dirty
 - **incomplete**: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
 - e.g., occupation=" "
 - **noisy**: containing errors or outliers
 - e.g., Salary="-10"
 - **inconsistent**: containing discrepancies in codes or names
 - e.g., Age="42" Birthday="03/07/1997"
 - e.g., Was rating "1,2,3", now rating "A, B, C"
 - e.g., discrepancy between duplicate records

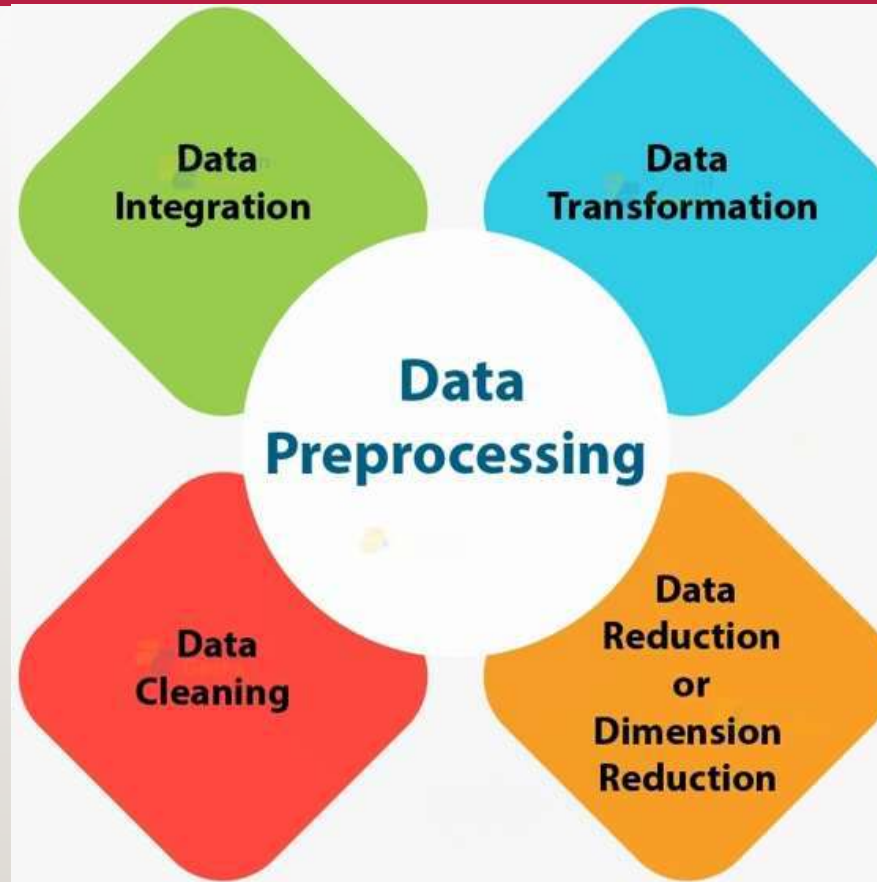
WHY DATA ARE DIRTY?

- Incomplete data may come from
 - “Not applicable” data value when collected
 - Different considerations between the time when the data was collected and when it is analyzed.
 - Human/hardware/software problems
- Noisy data (incorrect values) may come from
 - Faulty data collection instruments
 - Human or computer error at data entry
 - Errors in data transmission
- Inconsistent data may come from
 - Different data sources
 - Functional dependency violation (e.g., modify some linked data)
- Duplicate records also need data cleaning

WHY DATA PREPROCESSING IS IMPORTANT IN MACHINE LEARNING CONTEXT?

- No quality data, no quality results!
 - Quality decisions must be based on quality data
 - e.g., duplicate or missing data may cause incorrect or even misleading statistics.
- In machine learning, preprocessing is a crucial step to ensure that the data is in a format that the algorithm can understand and that it is free of errors or outliers that can negatively impact the model's performance.

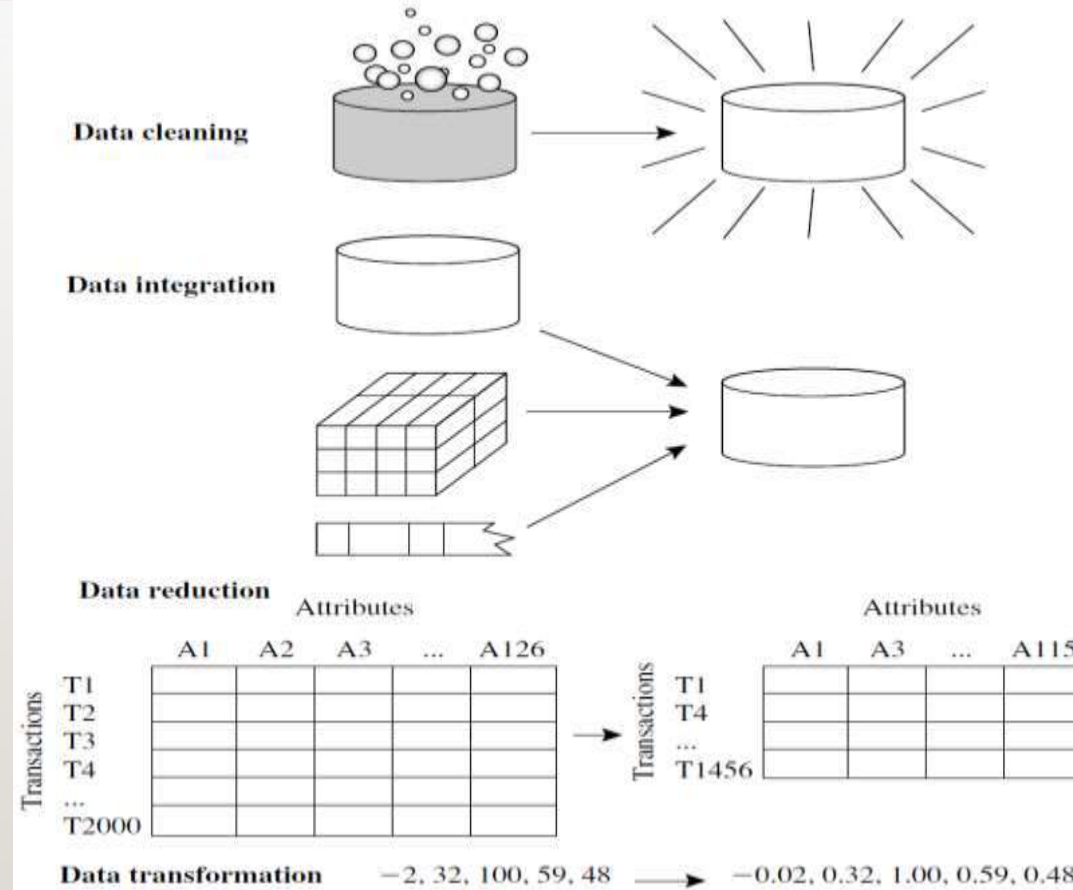
MAJOR TASKS IN DATA PREPROCESSING



MAJOR TASKS IN DATA PREPROCESSING

- Data cleaning
 - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- Data integration
 - Integration of multiple databases, data cubes, or files
- Data transformation
 - Normalization and aggregation
- Data reduction
 - Obtains reduced representation in volume but produces the same or similar analytical results

ILLUSTRATION OF MAJOR TASKS IN DATA PREPROCESSING



DATA CLEANING

- Data cleaning tasks
 - Fill in missing values
 - Identify outliers and smooth out noisy data
 - Correct inconsistent data
 - Resolve redundancy caused by data integration

MISSING DATA

- Data is not always available
 - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be due to
 - equipment malfunction
 - inconsistent with other recorded data and thus deleted
 - data not entered due to misunderstanding
 - certain data may not be considered important at the time of entry

Name	Age	Sex	Income	Class
Mike	40	Male	150k	Big spender
Jenny	20	Female	?	Regular

HOW TO HANDLE MISSING DATA?

- **Ignore the tuple:** usually done when class label is missing (assuming the tasks in classification—not effective when the percentage of missing values per attribute varies considerably).
- **Fill in the missing value manually:** tedious + infeasible?
- **Fill in it automatically** with
 - a global constant : e.g., “unknown”, a new class?!
 - the attribute mean
 - the attribute mean for all samples belonging to the same class: smarter
 - the most probable value

HOW TO HANDLE NOISY DATA?

- **Binning:** This method is to smooth or handle noisy data. First, the data is sorted, and then the sorted values are separated and stored in the form of bins. There are three methods for smoothing data in the bin.
 - **Smoothing by bin mean method:** In this method, the values in the bin are replaced by the mean value of the bin;
 - **Smoothing by bin median:** In this method, the values in the bin are replaced by the median value;
 - **Smoothing by bin boundary:** In this method, the using minimum and maximum values of the bin values are taken, and the closest boundary value replaces the values.
- **Regression:** This is used to smooth the data and will help to handle data when unnecessary data is present. For the analysis, purpose regression helps to decide the variable that is suitable for our analysis.
- **Clustering:** This is used for finding the outliers and also in grouping the data. Clustering is generally used in unsupervised learning.

DATA TRANSFORMATION: BINNING

Sorted data for *price* (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34

Partition into (equal-frequency) bins:

Bin 1: 4, 8, 15

Bin 2: 21, 21, 24

Bin 3: 25, 28, 34

Smoothing by bin means:

Bin 1: 9, 9, 9

Bin 2: 22, 22, 22

Bin 3: 29, 29, 29

Smoothing by bin boundaries:

Bin 1: 4, 4, 15

Bin 2: 21, 21, 24

Bin 3: 25, 25, 34

DATA INTEGRATION

- Data integration:
 - Combines data from multiple sources into a coherent store
- Schema integration: e.g., $A.cust-id \equiv B.cust-#$
 - Integrate metadata from different sources
- Entity identification problem:
 - Identify real-world entities from multiple data sources, e.g., Bill Clinton = William Clinton
- Detecting and resolving data value conflicts
 - For the same real-world entity, attribute values from different sources are different
 - Possible reasons: different representations, different scales, e.g., metric vs. British units (e.g., GPA in US and China)

DATA TRANSFORMATION

- Smoothing: remove noise from data
- Aggregation: summarization
- Generalization: concept hierarchy climbing
- Normalization: scaled to fall within a small, specified range
 - min-max normalization
 - z-score normalization
 - normalization by decimal scaling
- Attribute/feature construction
 - New attributes constructed from the given ones

DATA TRANSFORMATION

- **Smoothing:**

- With the help of algorithms, we can **remove noise from the dataset**, which helps in knowing the important features of the dataset. By smoothing, we can find even a simple change that helps in prediction.

- **Aggregation:**

- In this method, the **data is stored and presented in the form of a summary.**

- **Discretization:**

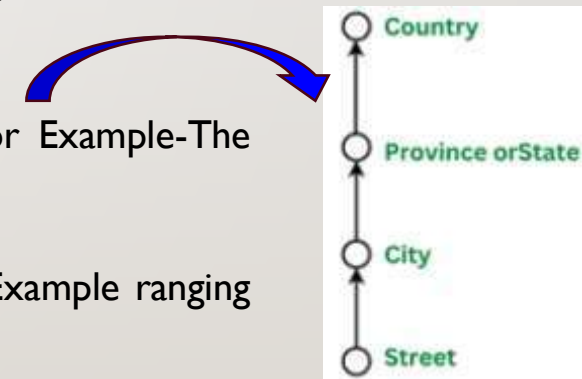
- The **continuous data** here is **split into intervals.**
- Discretization reduces the data size. For example, rather than specifying the class time, we can set an interval like (3 pm-5 pm or 6 pm-8 pm).

- **Concept Hierarchy Generation:**

- Here **attributes are converted from lower level to higher level in the hierarchy.** For Example-The attribute “street” can be converted to “country”.

- **Normalization:**

- It is the method of scaling the **data** so that it **can be represented in a smaller range.** Example ranging from -1.0 to 1.0.



DATA TRANSFORMATION: NORMALIZATION

- **Min-max normalization:** to $[new_min_A, new_max_A]$

$$v' = \frac{v - min_A}{max_A - min_A} (new_max_A - new_min_A) + new_min_A$$

- The min-max normalization method converts a dataset into a scale ranging from 0 to 1.
- Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped

to

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

- **Z-score normalization (μ : mean, σ : standard deviation):**

$$v' = \frac{v - \mu_A}{\sigma_A}$$

- Ex. Let $\mu = 54,000$, $\sigma = 16,000$. Then $\frac{73,600 - 54,000}{16,000} = 1.225$

DATA TRANSFORMATION: NORMALIZATION

- **Normalization by Decimal Scaling:**

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

- Let the input data be: -10, 201, 301, -401, 501, 601, 701

To normalize the above data,

Step 1: Maximum absolute value in given data (m): 701

Step 2: Divide the given data by 1000 (i.e., $j=3$)

Why $j=3$? We will count total numbers in our maximum value and then put 1 and after 1 we can put zeros equal to the length of the maximum value

Result: The normalized data is: -0.01, 0.201, 0.301, -0.401, 0.501, 0.601, 0.701

DATA REDUCTION

- **Dimensionality reduction:**

- In this method, **irrelevant features are removed**
- Avoid the curse of dimensionality
- Reduce time and space required during the learning process
- Allow easier visualization
- Techniques for dimensionality reduction are principal component analysis, feature selection, and wavelet transforms

- **Numerosity reduction:**

- In this method, the **data representation is made smaller by reducing the volume**. There will not be any loss of data in this reduction
- Techniques for numerosity reduction are parametric (regression, log-linear models) and non-parametric methods (histograms, clustering, sampling)

- **Data compression:**

- The **compressed form of data** is called data compression
- This **compression can be lossless or lossy**. When there is no loss of information during compression, it is called lossless compression. Whereas lossy compression reduces information, but it removes only the unnecessary information

DATA REDUCTION

- **Problem:** Feature selection (i.e., **attribute subset selection**):
 - Select a **minimum set of features** such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features
 - **Nice side-effect:** reduces # of attributes in the discovered patterns (which are now easier to understand)
- **Solution:** Heuristic methods (due to the exponential # of choices) are usually greedy:
 - step-wise forward selection
 - step-wise backward elimination
 - combining forward selection and backward elimination
 - decision-tree induction

DATA REDUCTION: FEATURE SELECTION

Stepwise forward selection:

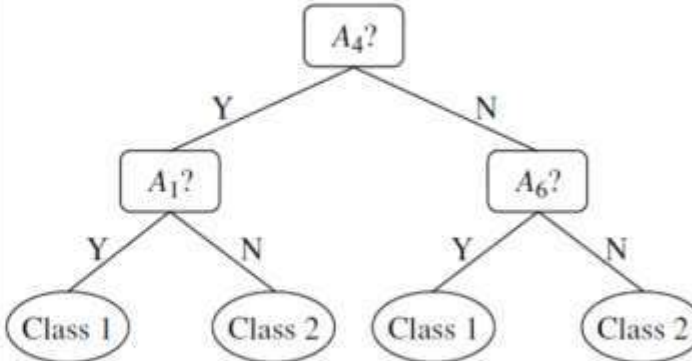
The procedure starts with an empty set of attributes as the reduced set. The best of the original attributes is determined and added to the reduced set. **At each subsequent iteration or step, the best of the remaining original attributes is added to the set.**

Stepwise backward elimination:

The procedure starts with the full set of attributes. **At each step, it removes the worst attribute remaining in the set.**

Combination of forward selection and backward elimination:

The stepwise forward selection and backward elimination methods can be combined so that, at each step, the procedure **selects the best attribute and removes the worst from among the remaining attributes.**

Forward selection	Backward elimination	Decision tree induction
<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p> <p>Initial reduced set: $\{\}$ $\Rightarrow \{A_1\}$ $\Rightarrow \{A_1, A_4\}$ \Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>	<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p> <p>$\Rightarrow \{A_1, A_3, A_4, A_5, A_6\}$ $\Rightarrow \{A_1, A_4, A_5, A_6\}$ \Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>	<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p>  <p>\Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>

DATA REDUCTION: PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis

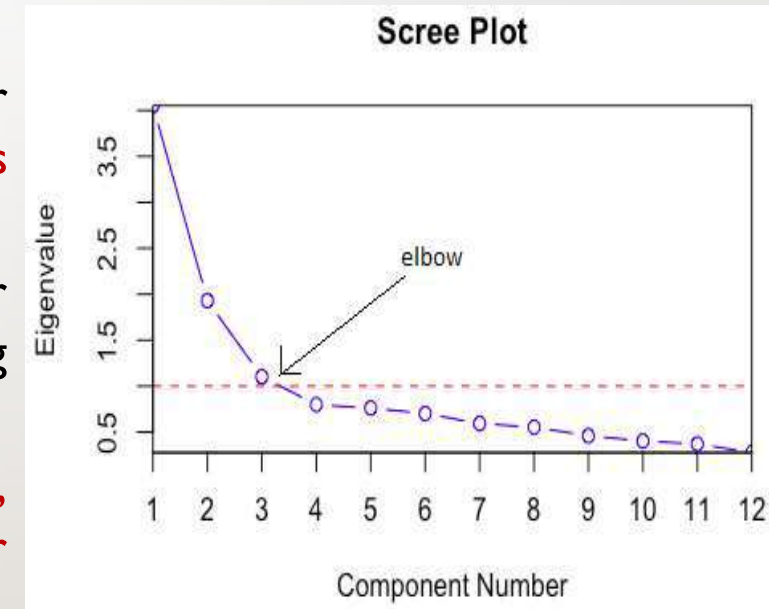
- Principal component analysis or **PCA is a dimensionality reduction method** that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one

How do you do a principal component analysis?

- Standardize the range of continuous initial variables
- Compute the covariance matrix to identify correlations
- Compute the eigenvectors and eigenvalues of the covariance matrix to identify the principal components
- Create a feature vector to decide which principal components to keep
- Recast the data along the axes of the principal component that still contains most of the information in the large set.

PCA – SCREE PLOT

- A **common method for determining the number of PCs to be retained** is a graphical representation known as a scree plot.
- A Scree Plot is a simple line segment plot that shows the eigenvalues for each PC. **It shows the eigenvalues on the y-axis and the number of factors on the x-axis.**
- **It always displays a downward curve.** Most scree plots look broadly similar in shape, starting high on the left, falling rather quickly, and then flattening out at some point.
- This is because **the first component usually explains much of the variability, the next few components explain a moderate amount, and the latter components only explain a small fraction of the overall variability.**
- The scree plot criterion looks for the “elbow” in the curve and **selects all components just before the line flattens out.**



PCA – APPLICATIONS, CONSIDERATIONS, AND LIMITATIONS

■ Applications of PCA

- ❖ Dimensionality reduction and feature extraction: PCA can simplify complex datasets by reducing the number of variables while retaining meaningful information.
- ❖ Data preprocessing: PCA can be used to preprocess data before applying other machine learning algorithms to improve their performance.
- ❖ Noise reduction: PCA can remove noise and extract signals from data.
- ❖ Visualization: PCA can help visualize high-dimensional data in a lower-dimensional space.

■ Considerations and Limitations

- ❖ PCA assumes linearity and normality in the data.
- ❖ Outliers can affect the results of PCA.
- ❖ Scaling and standardization of variables are important to avoid the dominance of certain features.
- ❖ PCA is a linear technique and may not capture nonlinear relationships in the data.

DATA SPLITTING

- **Data splitting** is a crucial step in the machine learning workflow. It involves dividing the dataset into separate subsets for training, validation, and testing to ensure that the model generalizes well to new, unseen data. The following are the types of splitting.
- **Train-Test Split**
 - Purpose: To evaluate the model's performance on unseen data.
 - Method:
 - The dataset is split into two parts: a training set and a testing set.
 - Common split ratio is 80% for training and 20% for testing, but this can vary (e.g., 70-30, 60-40).
- **Train-Validation-Test Split**
 - Purpose: To tune the model's hyperparameters and evaluate its performance.
 - Method:
 - The dataset is split into three parts: training set, validation set, and testing set.
 - Common split ratio is 60% for training, 20% for validation, and 20% for testing.

DATA SPLITTING

- Cross-Validation
 - Purpose: To ensure the model performs well on different subsets of the data.
 - Method:
 - The dataset is split into k equally sized folds.
 - The model is trained on $k-1$ folds and tested on the remaining fold.
 - This process is repeated k times with each fold used exactly once as the test data.
 - Common values for k are 5 or 10.
- Stratified Splitting
 - Purpose: To maintain the same proportion of classes in each subset as in the original dataset, particularly useful for imbalanced datasets.
 - Method:
 - Similar to train-test split but ensures the distribution of classes is consistent across training and testing sets.

DATA BATCHING

- **Data batching** is a technique in machine learning where data is divided into smaller subsets, or batches, for processing. This is particularly useful for training models on large datasets, as it allows for more efficient memory usage and can speed up the training process. The following are the data batching practices.
- **Mini-Batch Gradient Descent**
 - Purpose: To update model parameters more frequently compared to batch gradient descent while maintaining computational efficiency.
- **Data Batching with PyTorch**
 - Purpose: To efficiently handle batching during model training with automatic handling of data loading and shuffling.
- **Data Batching with TensorFlow/Keras**
 - Purpose: To handle data batching using built-in functions for model training.
- **Custom Data Batching**
 - Purpose: To implement custom batching logic when built-in libraries are not sufficient.

DATA SHUFFLING

- **Data shuffling** is an important preprocessing step in machine learning that helps ensure that the data is randomly distributed. This reduces the risk of bias in the training process, especially in scenarios where the data may have inherent order or patterns.
- Basic Data Shuffling
 - Purpose: To randomly shuffle the dataset to ensure that the training and testing subsets are representative of the overall dataset.
- Shuffling Before Train-Test Split
 - Purpose: To ensure that the training and testing sets are not biased by any inherent order in the data.
- Shuffling During Cross-Validation
 - Purpose: To ensure that the folds in cross-validation are representative of the entire dataset.
- Shuffling in Stratified Cross-Validation
 - Purpose: To maintain class distribution while ensuring random distribution of samples across folds.

OVERFITTING AND UNDERFITTING

- **Overfitting** occurs when a model learns the training data too well, capturing noise and details that are not relevant to the general problem. This leads to a model that performs exceptionally well on training data but poorly on new, unseen data.
- Causes of Overfitting:
 - Complex Models: Models with too many parameters relative to the number of training samples.
 - Insufficient Data: Too little training data can cause the model to learn the noise and random fluctuations.
 - Too Many Features: Using too many irrelevant features can lead to overfitting.
- Indicators of Overfitting:
 - High accuracy on training data.
 - Low accuracy on validation/testing data.
 - Large gap between training and validation/testing performance.

OVERFITTING AND UNDERFITTING

- Solutions to Overfitting:
 - Simplify the Model: Use fewer parameters or features.
 - Regularization: Techniques like L1 (Lasso) or L2 (Ridge) regularization add a penalty for large coefficients.
 - Cross-Validation: Use techniques like k-fold cross-validation to ensure the model generalizes well.
 - Pruning (for Decision Trees): Reduce the size of the tree by removing sections that provide little power.
 - Early Stopping (for Neural Networks): Stop training when the performance on a validation set starts to degrade.

OVERFITTING AND UNDERFITTING

- **Underfitting** happens when a model is too simple to capture the underlying patterns in the data. This results in poor performance on both the training data and new data.
- Causes of Underfitting:
 - Too Simple Models: Models that are not complex enough to capture the data's underlying structure.
 - Insufficient Training: Not training the model for enough epochs or iterations.
 - Poor Feature Selection: Using features that do not capture the underlying patterns.
- Indicators of Underfitting:
 - Low accuracy on both training and validation/testing data.
 - Minimal gap between training and validation/testing performance.

OVERFITTING AND UNDERFITTING

- Solutions to Underfitting:
 - Increase Model Complexity: Use a more complex model with more parameters.
 - Train Longer: Train the model for more epochs or iterations.
 - Feature Engineering: Create better features or use more relevant data.
 - Reduce Regularization: If using regularization techniques, reduce the penalty to allow the model to better fit the data.

SUMMARY

- Data preprocessing techniques are responsible for converting raw data into a format understandable by the machine learning algorithms.
- Data cleaning, data transformation, data integration, data reduction, data splitting, data batching, and data shuffling are the major steps in data preprocessing.
- It is essential to find the balance between overfitting and underfitting to create a model that generalizes well to new, unseen data. Techniques like cross-validation, proper feature selection, and regularization help in achieving this balance.

BOOKS

Textbooks:

- Best Practices in Data Cleaning: A Complete Guide to Everything You Need to Do Before and After Collecting Your Data, 1st Edition, Jason W. Osborne, 2012
- Data Cleaning, 1st Edition, Ihab F Ilyas and Xu Chu, 2019
- Data mining concepts and techniques, 3rd Edition, Jiawei Han, Micheline Kamber, and Jian Pei

Reference Books :

- Bad Data Handbook, 1st Edition, Q. E McCallum, 2012
- Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists, 1st Edition, Alice Zheng and Amanda Casari, 2018
- Introduction to data mining, Pang-Ning Tan and Michael Steinbach

WEBLINKS

1. <https://www.scaler.com/topics/data-science/data-preprocessing/>
2. <https://neptune.ai/blog/data-preprocessing-guide>
3. <https://www.analyticsvidhya.com/blog/2021/08/data-preprocessing-in-data-mining-a-hands-on-guide/#:~:text=Data%20preprocessing%20is%20the%20process,learning%20or%20data%20mining%20algorithms.>
4. <https://wikidocs.net/185539>
5. <https://www.turing.com/kb/guide-to-principal-component-analysis>

Self-Assessment Questions

1. To remove noise and inconsistent data ____ is needed.

- A) Data cleaning
- B) Data transformation
- C) Data integration
- D) Data reduction

2. **Normalization by _____ normalizes by moving the decimal point of values of attributes.**

- A) Z-score
- B) Z-index
- C) Decimal scaling
- D) Min-max normalization

TERMINAL QUESTIONS

1. You are given a dataset with several missing values scattered across different features. How would you decide on the best strategy to handle these missing values? Discuss the considerations and techniques you would use.
2. What techniques would you use to reduce noise in your data? Explain how these techniques can improve model performance with an example.
3. How would you preprocess data that needs to be aggregated over certain periods or categories? Provide an example where data aggregation is necessary before modeling.
4. Why is it important to normalize your data before applying machine learning algorithms? Describe a scenario where normalization significantly improves the model performance.
5. Explain the difference between standard scaling and min-max scaling. Provide an example where one might be preferred over the other.
6. Discuss the importance of dimensionality reduction in data preprocessing. How would you apply PCA (Principal Component Analysis) to a high-dimensional dataset? What are the potential pitfalls of using PCA?
7. What considerations would you consider when splitting your data into training and testing sets?

THANK YOU



Team – ARTIFICIAL INTELLIGENCE AND MACHINE
LEARNING