Data Preprocessing

Data Mining & Neural Networks

Dr. Wilmer Garzón

Director, Master's Program in Data Science
Department of Computer Engineering

Escuela Colombiana de Ingeniería
Universidade da Coruña

2025

Data Preprocessing

- **Preparing Raw Data for Analysis and Learning**
- Data preprocessing is a fundamental step in any data science or machine learning project.
- Raw data is often messy, incomplete, and inconsistent.

This session will guide you through the essential steps to clean, structure, and prepare data so it can be used effectively in mining and neural network models.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD

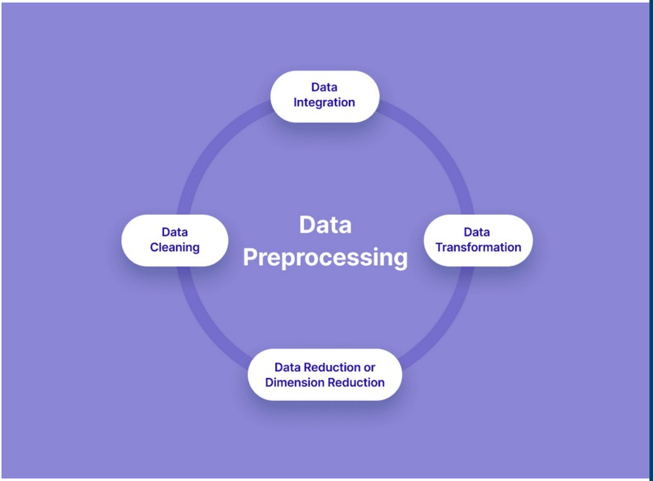
Learning Objectives

By the end of this session, you will be able to:

- Understand the role and goals of **data preprocessing**.
- Identify and correct common issues in **raw datasets**.
- Apply techniques such as **imputation**, **normalization**, and **encoding**.
- Evaluate the impact of data preprocessing on **model performance**.
- Use tools and libraries to implement **preprocessing workflows**.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
UNIVERSIDAD

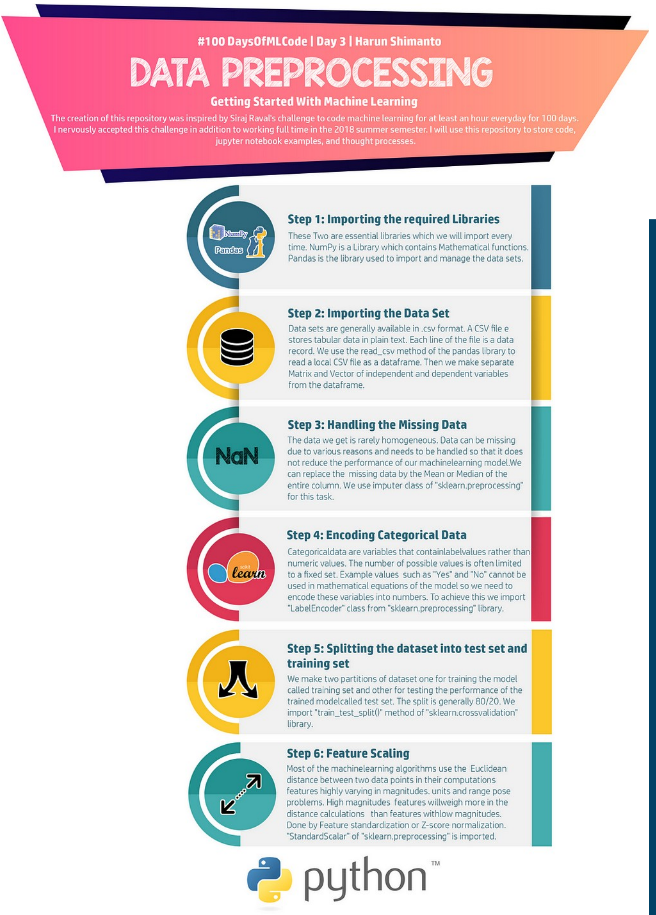
What is Data Preprocessing?



- DP is the process of transforming raw data into a clean and usable format.
- Real-world data is often noisy, incomplete, and unformatted.
- Preprocessing includes task:
  - handling missing values,
  - removing duplicates,
  - standardizing formats,
  - encoding categorical features,
  - scaling numeric data.
- DP ensures better learning and generalization in models.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD



#100 DaysOfMLCode | Day 3 | Harun Shimanto

DATA PREPROCESSING

Getting Started With Machine Learning

The creation of this repository was inspired by Siraj Raval's challenge to code machine learning for at least an hour everyday for 100 days. I nervously accepted this challenge in addition to working full time in the 2018 summer semester. I will use this repository to store code, jupyter notebook examples, and thought processes.

Step 1: Importing the required Libraries

These Two are essential libraries which we will import every time. NumPy is a Library which contains Mathematical functions. Pandas is the library used to import and manage the data sets.

Step 2: Importing the Data Set

Data sets are generally available in .csv format. A CSV file e stores tabular data in plain text. Each line of the file is a data record. We use the read_csv method of the pandas library to read a local CSV file as a dataframe. Then we make separate Matrix and Vector of independent and dependent variables from the dataframe.

Step 3: Handling the Missing Data

The data we get is rarely homogeneous. Data can be missing due to various reasons and needs to be handled so that it does not reduce the performance of our machinelearning model.We can replace the missing data by the Mean or Median of the entire column. We use imputer class of "sklearn.preprocessing" for this task.

Step 4: Encoding Categorical Data

Categoricaldata are variables that containlabelvalues rather than numeric values. The number of possible values is often limited to a fixed set. Example values such as "Yes" and "No" cannot be used in mathematical equations of the model so we need to encode these variables into numbers. To achieve this we import "LabelEncoder" class from "sklearn.preprocessing" library.

Step 5: Splitting the dataset into test set and training set

We make two partitions of dataset one for training the model called training set and other for testing the performance of the trained modelcalled test set. The split is generally 80/20. We import "train_test_split()" method of "sklearn.crossvalidation" library.

Step 6: Feature Scaling

Most of the machinelearning algorithms use the Euclidean distance between two data points in their computations features highly varying in magnitudes. units and range pose problems. High magnitudes features willweigh more in the distance calculations than features withlow magnitudes. Done by Feature standardization or Z-score normalization. "StandardScalar" of "sklearn.preprocessing" is imported.

Why Preprocessing Matters

- Preprocessing is essential because AI algorithms require structured, clean, and numerical data.
- If data is not properly cleaned, models may underperform or produce misleading results.
- For instance, a dataset with inconsistent labels or outliers may confuse a classifier.
- Data quality directly impacts model accuracy, fairness, and interpretability.

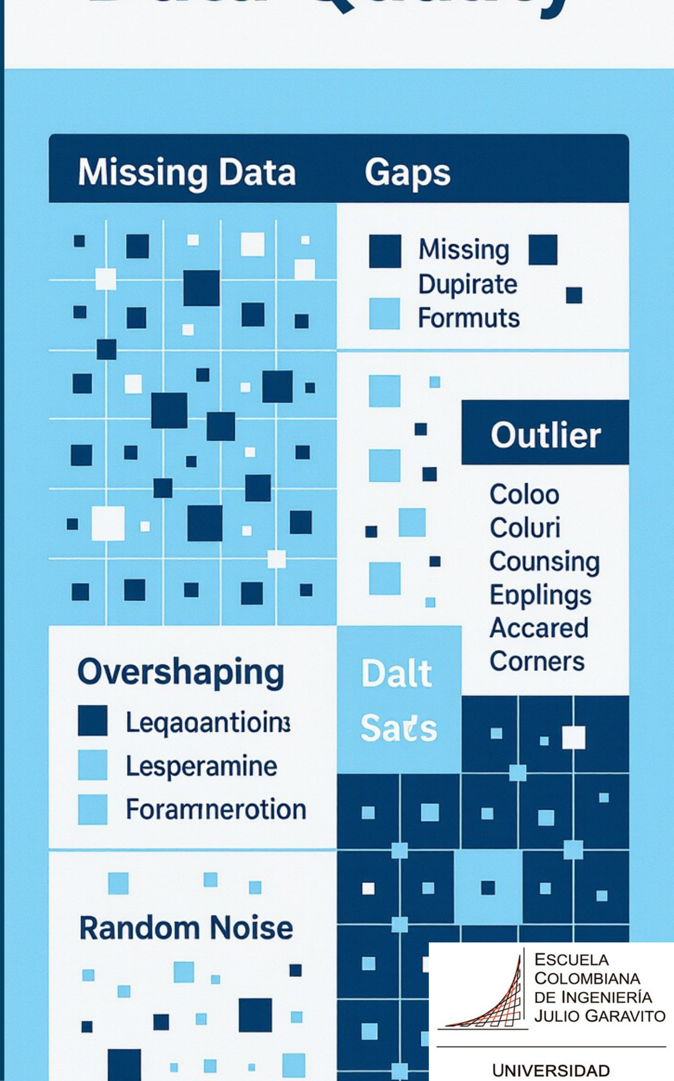
Escuela Colombiana de Ingeniería Julio Garavito
Universidad

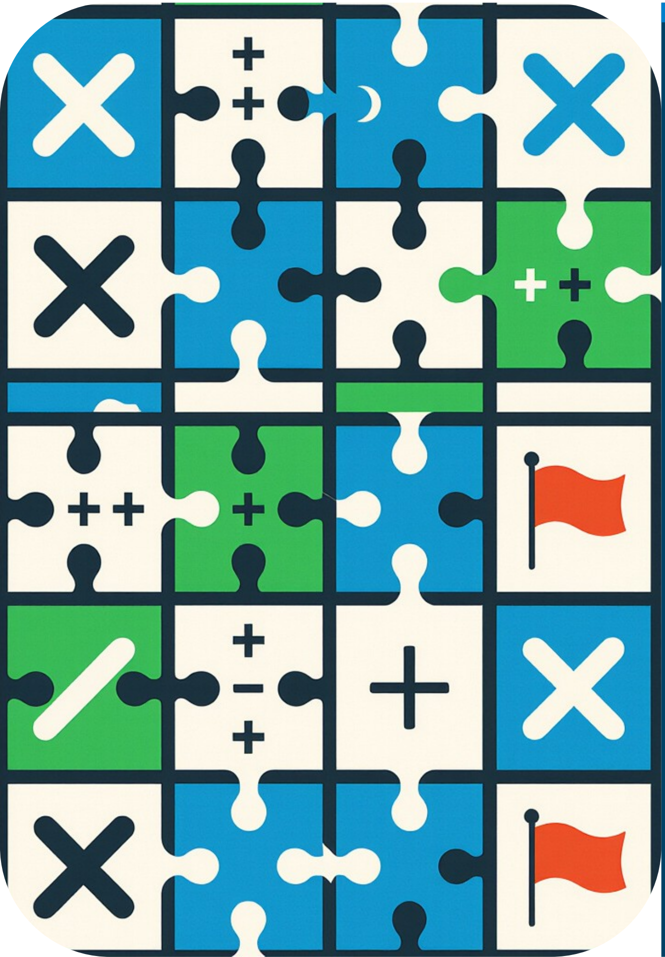
Common Data Issues

Typical problems found in raw data include:

- **Missing values**: Cells with no recorded data.
- **Duplicates**: Repeated rows that skew distributions.
- **Inconsistent formats**: Mixed cases like “yes”, “Yes”, “Y”.
- **Outliers**: Extremely high or low values.
- **Noise**: Irrelevant or erroneous data points.

Each issue requires specific strategies for resolution before modeling.





Handling Missing Values

Missing data is a common problem. Options include:

- **Deletion**: Remove rows or columns with too many NAs.
- **Imputation**: Replace missing values using statistical measures (**mean**, **median**) or machine learning techniques.
- **Flagging**: Create binary indicators for missingness.

Example: In a health dataset, missing BMI values might be imputed using the median of patients in the same age group.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD

Removing Duplicates

Duplicates can inflate patterns and introduce bias. Steps:

- Use tools like **pandas.drop_duplicates()** to identify and remove exact matches.
- For near-duplicates, apply **fuzzy matching algorithms**.

Be cautious: not all repetitions are duplicates. For example, multiple transactions by the same customer are valid and should not be removed blindly.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
UNIVERSIDAD

Normalization and Scaling

Machine learning models are sensitive to feature scales. Techniques include:

- **Min-Max Scaling**: Transforms values to a 0–1 range.
- **Standardization**: Converts features to have mean = 0 and std = 1.
- **Robust Scaling**: Uses median and IQR to handle outliers.

Example: When predicting prices, features like area and number of rooms should be scaled to avoid dominance by large values.

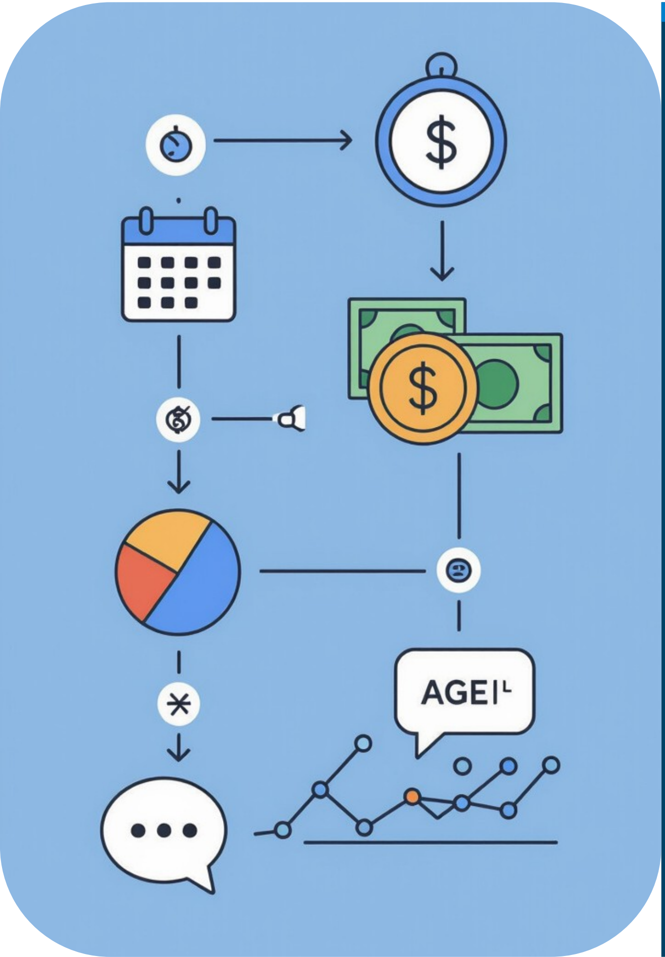
Encoding Categorical Variables

Models require numeric inputs. Convert categories using:

- **Label Encoding**: Assign integers to categories (e.g., Red=0, Blue=1).
- **One-Hot Encoding**: Create binary columns for each category.
- **Target Encoding**: Use average target value for each category.

Example: For a “City” column with 3 cities, one-hot encoding creates 3 binary features: City_A, City_B, City_C.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
UNIVERSIDAD



Feature Engineering

Feature engineering creates new variables that help models. Examples:

- From “Date of Birth,” extract “Age.”
- Combine “Income” and “Family Size” to get “Income per Person.”
- From text, extract sentiment or keyword presence.

Good feature engineering improves model accuracy and interpretability and often requires domain knowledge.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

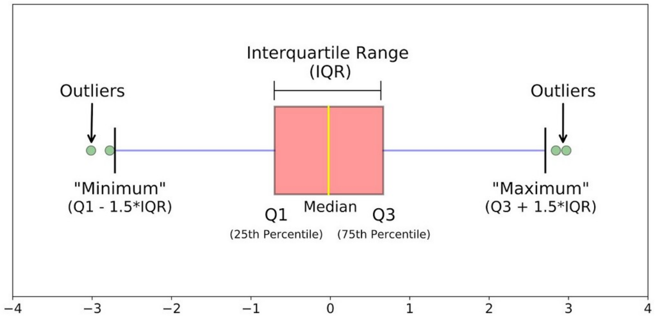
UNIVERSIDAD

Outlier Detection and Treatment

Outliers can distort statistical models. Detection methods:

- **Z-score**: Values more than 3 standard deviations from the mean.
- **IQR method**: Values outside Q1 – 1.5×IQR or Q3 + 1.5×IQR.
- **Visuals**: Boxplots, scatterplots.

Treatment: transform (e.g., log), cap values, or remove them, depending on the use case and domain knowledge.



ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

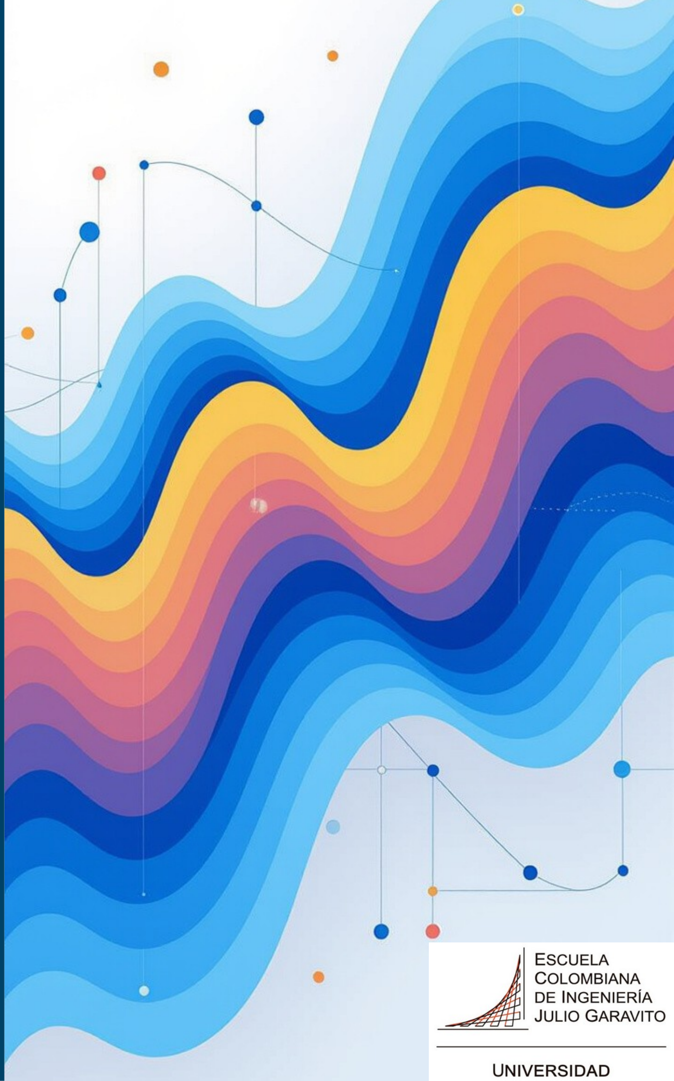
UNIVERSIDAD

Noise Reduction

Noise = random errors or irrelevant features. Reduction techniques:

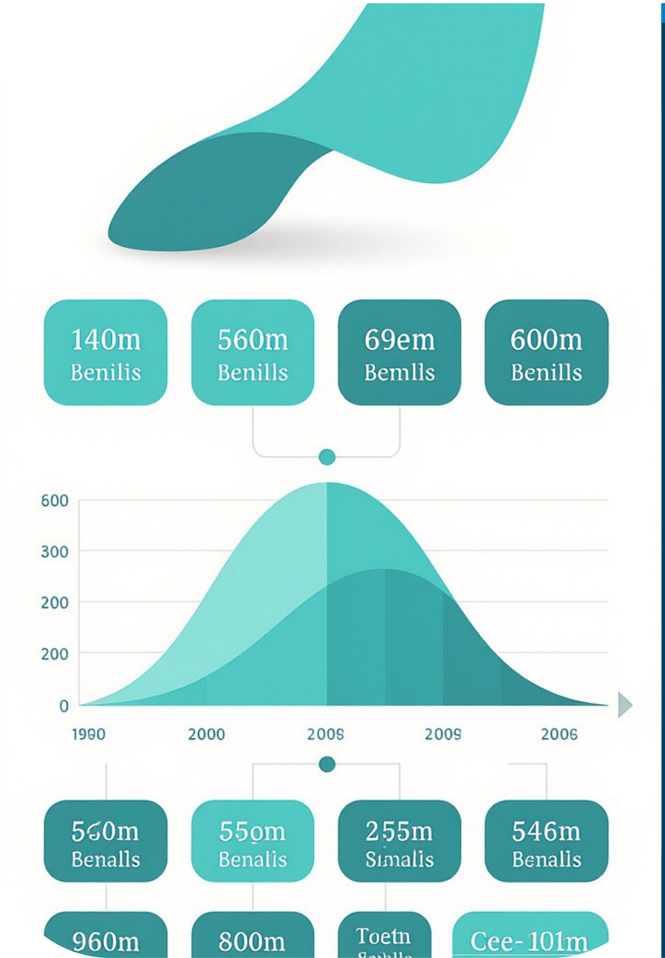
- **Smoothing**: Moving averages on time series.
- **Binning**: Group similar values to reduce variation.
- **Dimensionality Reduction**: Use PCA or feature selection.

Example: In a sensor dataset, noise from faulty readings can be smoothed using a rolling mean to reveal trends.



ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD



Data Transformation

Transformations help make data more usable for models. Examples:

- **Log Transform**: Reduces skew in long-tailed distributions.
- **Binning**: Converts continuous values into intervals.
- **Text Vectorization**: Converts raw text into word counts or embeddings.

These changes improve model performance and meet algorithm assumptions.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD

Feature Selection

Not all features are helpful. Feature selection:

- Improves model accuracy.
- Reduces overfitting.
- Makes models faster and easier to interpret.
- Methods include:
- **Correlation matrix**
- **Recursive Feature Elimination (RFE)**
- **Tree-based importance**

Avoid irrelevant or redundant features that add noise to the model.

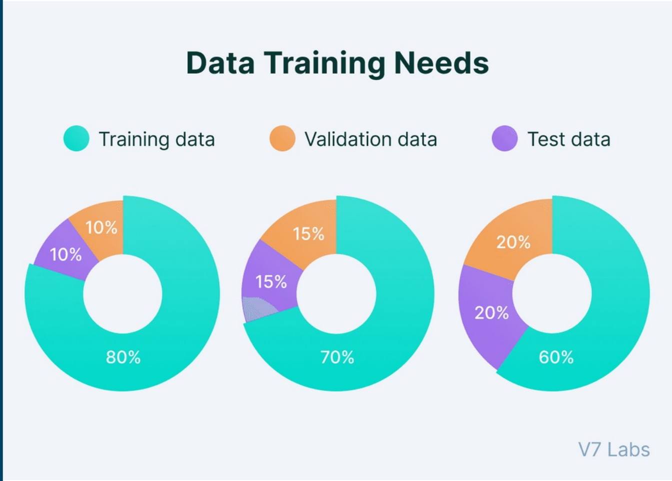
ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
UNIVERSIDAD

Data Splitting

Before training, split data into:

- **Training set** (e.g., 70%): used to build the model.
- **Validation set** (e.g., 15%): used to tune hyperparameters.
- **Test set** (e.g., 15%): used to assess final model performance.

Use stratified sampling in classification to maintain class balance.



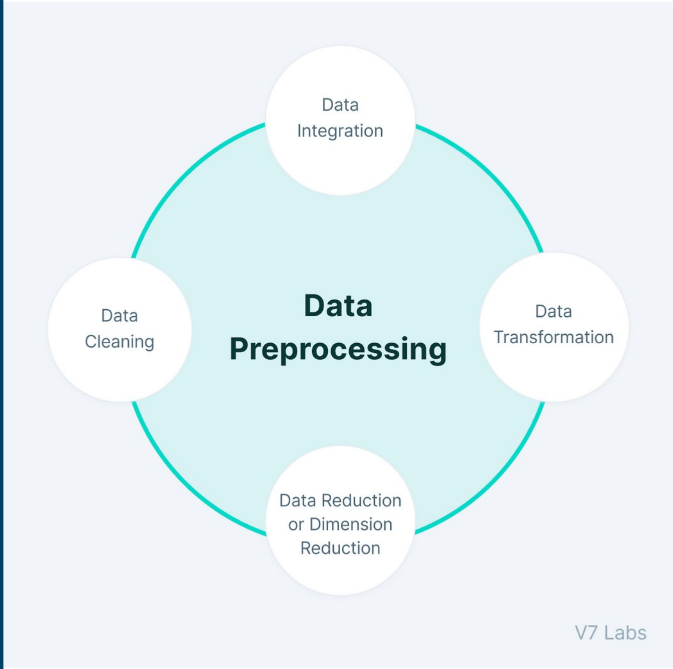
ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
UNIVERSIDAD

Preprocessing for Neural Networks

Neural networks require highly curated inputs:

- Scale features to [0, 1] or standard normal.
- Encode categories with one-hot encoding.
- Pad sequences to fixed lengths for NLP.
- Normalize pixel values (e.g., divide by 255).

Good preprocessing improves convergence and generalization of deep learning models.



Tools and Libraries

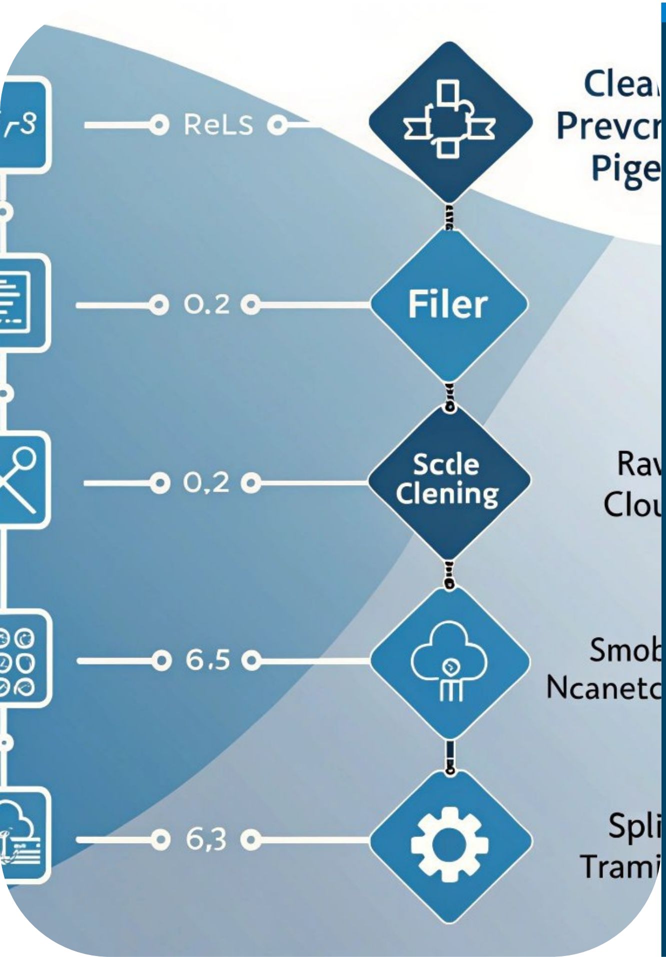
Preprocessing tools:

- **Python**: pandas, NumPy, scikit-learn’s Pipeline
- **TensorFlow/Keras**: tf.data, preprocessing layers
- **PyTorch**: torchvision.transforms for image preprocessing
- **R**: tidyverse, caret
- GUI Tools: KNIME, RapidMiner

Use pipelines to standardize and automate your workflow.

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD



Preprocessing Pipelines

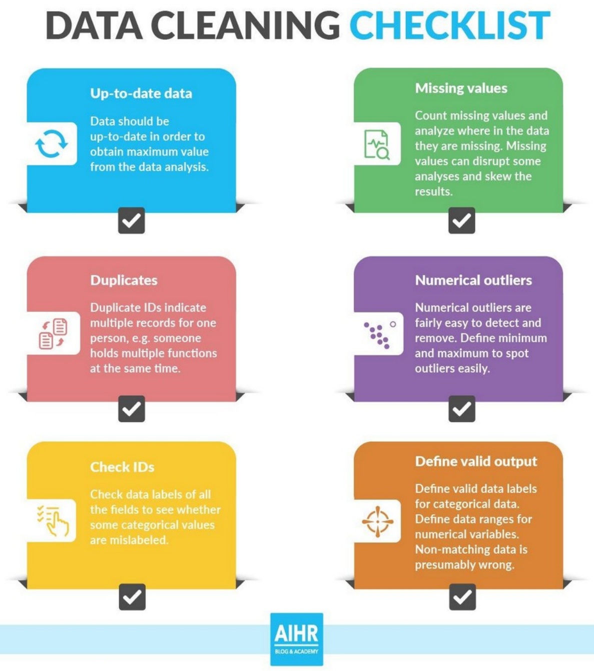
Automate the sequence of steps with pipelines:

- Load raw data
- Clean (handle missing, duplicates)
- Normalize/scale
- Encode categories
- Feature engineer
- Split data

Example in scikit-learn: Pipeline([('scale', StandardScaler()), ('model', SVC())])

Common Mistakes to Avoid

- **Preprocessing test data** before splitting
- **Ignoring missing values**
- **Not scaling numerical features**
- **Encoding categories inconsistently**
- **Overfitting with excessive features**
- **Solution**: Validate each step and use reproducible scripts and pipelines.



DATA CLEANING CHECKLIST

Up-to-date data
Data should be up-to-date in order to obtain maximum value from the data analysis.

Missing values
Count missing values and analyze where in the data they are missing. Missing values can disrupt some analyses and skew the results.

Duplicates
Duplicate IDs indicate multiple records for one person, e.g. someone holds multiple functions at the same time.

Numerical outliers
Numerical outliers are fairly easy to detect and remove. Define minimum and maximum to spot outliers easily.

Check IDs
Check data labels of all the fields to see whether some categorical values are mislabeled.

Define valid output
Define valid data labels for categorical data. Define data ranges for numerical variables. Non-matching data is presumably wrong.

AIHR BLOG & ACADEMY

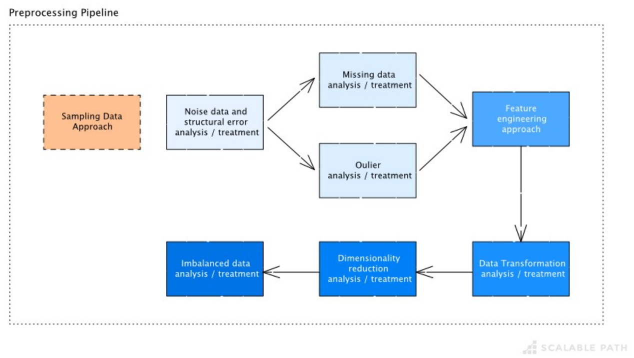
ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
UNIVERSIDAD

Summary and Next Steps

Preprocessing ensures data is reliable and ready for learning.

- We covered:
- **Handling missing data** and outliers
- **Scaling and encoding**
- **Feature engineering** and selection
- **Pipelines and tools**
- **Next session**: hands-on preprocessing with Python (pandas and scikit-learn) using a real-world dataset.

Figure 1: Data Preprocessing Pipeline


Preprocessing Pipeline
Sampling Data Approach
Noise data and structural error analysis / treatment
Missing data analysis / treatment
Oulier analysis / treatment
Feature engineering approach
Data Transformation analysis / treatment
Dimensionality reduction analysis / treatment
Imbalanced data analysis / treatment
SCALABLE PATH

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD

INTER NATIONAL SUMMER SCHOOL
UNIVERSIDADE DA CORUÑA

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

UNIVERSIDAD