# Statistics and Data Analysis

## Unit 01 – Lecture 03: Preprocessing Pipelines and Exploratory Data Analysis (EDA)

Tofik Ali

School of Computer Science, UPES Dehradun

February 14, 2026

https://github.com/tali7c/Statistics-and-Data-Analysis

# Quick Links

Workflow EDA Checklist Plots Demo Summary

# Agenda

# Learning Outcomes

- Explain what a preprocessing pipeline is and why it matters

# Learning Outcomes

- Explain what a preprocessing pipeline is and why it matters
- Apply a simple end-to-end workflow: load $\rightarrow$ clean $\rightarrow$ validate $\rightarrow$ summarize

# Learning Outcomes

- Explain what a preprocessing pipeline is and why it matters
- Apply a simple end-to-end workflow: load $\rightarrow$ clean $\rightarrow$ validate $\rightarrow$ summarize
- Perform basic EDA: missingness, summary stats, group summaries, correlations

# Learning Outcomes

- Explain what a preprocessing pipeline is and why it matters
- Apply a simple end-to-end workflow: load $\rightarrow$ clean $\rightarrow$ validate $\rightarrow$ summarize
- Perform basic EDA: missingness, summary stats, group summaries, correlations
- Choose appropriate plots for numeric and categorical variables

# What is a Pipeline?

A pipeline is an ordered set of steps applied consistently to data.

- Makes analysis **reproducible** (same input $\Rightarrow$ same output)

# What is a Pipeline?

A pipeline is an ordered set of steps applied consistently to data.

- Makes analysis **reproducible** (same input $\Rightarrow$ same output)
- Reduces mistakes (steps are documented and repeatable)

# What is a Pipeline?

A pipeline is an ordered set of steps applied consistently to data.

- Makes analysis **reproducible** (same input $\Rightarrow$ same output)
- Reduces mistakes (steps are documented and repeatable)
- Helps avoid **data leakage** (train/test separation)

# Typical End-to-End Workflow (Practical)

1. Understand the question (what do you want to learn/decide?)

# Typical End-to-End Workflow (Practical)

1. Understand the question (what do you want to learn/decide?)
2. Acquire data (files, DB, API)

# Typical End-to-End Workflow (Practical)

1. Understand the question (what do you want to learn/decide?)
2. Acquire data (files, DB, API)
3. Inspect: shape, dtypes, missingness

# Typical End-to-End Workflow (Practical)

1. Understand the question (what do you want to learn/decide?)
2. Acquire data (files, DB, API)
3. Inspect: shape, dtypes, missingness
4. Clean: duplicates, invalid ranges, inconsistent categories

# Typical End-to-End Workflow (Practical)

1. Understand the question (what do you want to learn/decide?)
2. Acquire data (files, DB, API)
3. Inspect: shape, dtypes, missingness
4. Clean: duplicates, invalid ranges, inconsistent categories
5. Validate: check constraints (0–100%, 0–10 CGPA, etc.)

# Typical End-to-End Workflow (Practical)

1. Understand the question (what do you want to learn/decide?)
2. Acquire data (files, DB, API)
3. Inspect: shape, dtypes, missingness
4. Clean: duplicates, invalid ranges, inconsistent categories
5. Validate: check constraints (0–100%, 0–10 CGPA, etc.)
6. EDA: summary tables + plots + simple relationships

# Typical End-to-End Workflow (Practical)

1. Understand the question (what do you want to learn/decide?)
2. Acquire data (files, DB, API)
3. Inspect: shape, dtypes, missingness
4. Clean: duplicates, invalid ranges, inconsistent categories
5. Validate: check constraints (0–100%, 0–10 CGPA, etc.)
6. EDA: summary tables + plots + simple relationships
7. Save outputs (cleaned dataset, plots, summary tables)

# Example: “Pipeline” in Code (Concept)

```
df = read_raw()
df = clean_strings(df)
df = coerce_types(df)
df = range_check(df)
df = impute_missing(df)
save_clean(df)
eda_report(df)
```

This is a simple pipeline: each step has a clear purpose.

# Exercise 1: Put Steps in Order

Arrange these steps in a reasonable order:

1. EDA plots
2. Load raw data
3. Fix data types + invalid ranges
4. Save cleaned dataset
5. Check missingness

# Solution 1

One reasonable order:

1. Load raw data
2. Check missingness
3. Fix data types + invalid ranges
4. EDA plots
5. Save cleaned dataset

# What is EDA?

Exploratory Data Analysis (EDA) is the first structured look at your data.

- Understand distribution (shape, spread, outliers)

# What is EDA?

Exploratory Data Analysis (EDA) is the first structured look at your data.

- Understand distribution (shape, spread, outliers)
- Understand relationships (scatter plots, correlation)

# What is EDA?

Exploratory Data Analysis (EDA) is the first structured look at your data.

- Understand distribution (shape, spread, outliers)
- Understand relationships (scatter plots, correlation)
- Compare groups (e.g., program-wise summaries)

# What is EDA?

Exploratory Data Analysis (EDA) is the first structured look at your data.

- Understand distribution (shape, spread, outliers)
- Understand relationships (scatter plots, correlation)
- Compare groups (e.g., program-wise summaries)
- Identify issues early (missingness, strange values)

# EDA Checklist (Minimum)

- **Data quality:** missingness %, duplicates, invalid ranges

# EDA Checklist (Minimum)

- **Data quality:** missingness %, duplicates, invalid ranges
- **Univariate:** histograms/boxplots for numeric; bar charts for categorical

# EDA Checklist (Minimum)

- **Data quality:** missingness %, duplicates, invalid ranges
- **Univariate:** histograms/boxplots for numeric; bar charts for categorical
- **Bivariate:** scatter plot for numeric–numeric; boxplot for numeric by category

# EDA Checklist (Minimum)

- **Data quality:** missingness %, duplicates, invalid ranges
- **Univariate:** histograms/boxplots for numeric; bar charts for categorical
- **Bivariate:** scatter plot for numeric–numeric; boxplot for numeric by category
- **Multivariate (basic):** correlation matrix/heatmap for numeric columns

# EDA Checklist (Minimum)

- **Data quality:** missingness %, duplicates, invalid ranges
- **Univariate:** histograms/boxplots for numeric; bar charts for categorical
- **Bivariate:** scatter plot for numeric–numeric; boxplot for numeric by category
- **Multivariate (basic):** correlation matrix/heatmap for numeric columns
- **Group summaries:** mean/median/std by program or gender

# Plot Selection (Quick Guide)

- Numeric (one variable): histogram, boxplot

# Plot Selection (Quick Guide)

- Numeric (one variable): histogram, boxplot
- Categorical (one variable): bar chart (counts)

# Plot Selection (Quick Guide)

- Numeric (one variable): histogram, boxplot
- Categorical (one variable): bar chart (counts)
- Numeric vs numeric: scatter plot

# Plot Selection (Quick Guide)

- Numeric (one variable): histogram, boxplot
- Categorical (one variable): bar chart (counts)
- Numeric vs numeric: scatter plot
- Numeric vs categorical: boxplot (numeric grouped by category)

# Plot Selection (Quick Guide)

- Numeric (one variable): histogram, boxplot
- Categorical (one variable): bar chart (counts)
- Numeric vs numeric: scatter plot
- Numeric vs categorical: boxplot (numeric grouped by category)
- Many numeric features: correlation heatmap

# Exercise 2: Choose the Plot

Pick a good plot for each:

1. Distribution of `final_marks` (numeric)
2. Compare `final_marks` across `program` (categorical)
3. Relationship between `study_hours_week` and `final_marks`

# Solution 2

- (1) Histogram or boxplot
- (2) Boxplot of marks grouped by program
- (3) Scatter plot (hours vs marks)

# Exercise 3: Spot Data Leakage

A student computes mean/std for scaling using the **entire dataset**, then splits into train/test and trains a model.

**Question:** Is this correct? If not, what should be done instead?

# Solution 3

Not correct: it uses test information during training (**leakage**).
Correct approach:

- split into train/test first
- compute scaling parameters on **train only**
- apply the same parameters to test

# Mini Demo (Python)

Run from the lecture folder:

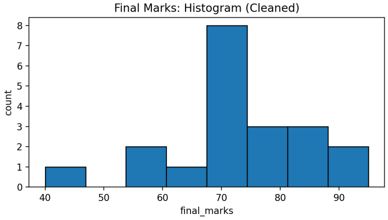
```
python demo/pipeline_eda_demo.py
```

Outputs:

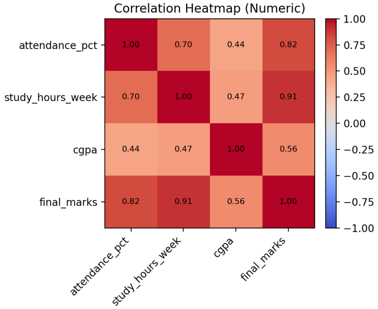
- `data/case_study_clean.csv`
- `data/summary_by_program.csv`
- `data/corr_matrix.csv`
- plots in `images/` (histogram, boxplot, scatter, heatmap)

# Demo Output (Example)

## Histogram




## Correlation Heatmap



# Summary

- Pipelines make preprocessing repeatable and reduce mistakes

**Exit question:** Name two checks you must do before trusting a dataset for analysis.

# Summary

- Pipelines make preprocessing repeatable and reduce mistakes
- EDA is about understanding quality, distributions, and relationships

**Exit question:** Name two checks you must do before trusting a dataset for analysis.

# Summary

- Pipelines make preprocessing repeatable and reduce mistakes
- EDA is about understanding quality, distributions, and relationships
- Pick plots based on variable types (numeric vs categorical)

**Exit question:** Name two checks you must do before trusting a dataset for analysis.

# Summary

- Pipelines make preprocessing repeatable and reduce mistakes
- EDA is about understanding quality, distributions, and relationships
- Pick plots based on variable types (numeric vs categorical)
- Save cleaned data, plots, and summary tables as reusable artifacts

**Exit question:** Name two checks you must do before trusting a dataset for analysis.