

Unit IV Advanced Predictive Analytics Algorithms and Python


4.1 Introduction of Exploratory Data Analytics (EDA) -Definition, Motivation, Steps in data exploration, data types.

Introduction to Exploratory Data Analysis (EDA)

Definition:


EDA is the process of **examining and understanding data** before applying any models. It helps you find:

- Patterns
- Trends
- Relationships
- Anomalies (outliers)
- Missing values

 **In simple words:** EDA is like “**getting to know your data**” before you use it.

Motivation: Why Do EDA?

- To understand the **structure** and **quality** of the data.
- To **detect errors** or **missing values**.
- To decide which **features are important**.
- To choose the right **data transformation** or **modeling technique**.
- To make **visualizations** that explain your data story.

 It helps in making better decisions for **data preprocessing** and **model selection**.

Steps in Data Exploration (EDA Process)

Here are the common steps involved in EDA:

1. Understand Your Data

- What is each column?
- What do the values mean?

2. Check Data Types

- Is it numeric, categorical, date, or text?

3. Check for Missing Data

- Are there any NA or empty values?
- How much data is missing?

4. Check for Duplicates

- Are there repeated rows?

5. Summary Statistics

- Mean, median, mode, min, max, standard deviation.

6. Univariate Analysis (One column at a time)

- Histograms
- Bar plots
- Boxplots

7. Bivariate/Multivariate Analysis (Two or more columns)

- Scatter plots
- Correlation matrices
- Pair plots

8. Outlier Detection

- Find and analyze strange or extreme values.

Data Types in EDA

Type	Description	Example
Numeric	Numbers that can be added, averaged	10, 23.5, 1000
Categorical	Groups or categories	Male/Female, Yes/No
Ordinal	Categories with order	Low, Medium, High
Date/Time	Represents dates or timestamps	2023-12-01, 10:30 PM
Text	Free-form text	"Hello world", comments

Summary:

Topic	Description
EDA	Process to explore and understand your data
Why EDA?	To find patterns, fix issues, and guide model choice
Key Steps	Understand, summarize, visualize, detect outliers
Data Types	Numeric, Categorical, Ordinal, Date, Text


4.2 Techniques to Improve Classification Accuracy: Introducing Ensemble Methods, Bagging, Boosting and AdaBoost, Random Forest

Goal: Improve Classification Accuracy

Sometimes, a single machine learning model (like a decision tree or logistic regression) may not give great results. So, we **combine multiple models** to get better performance. This is where **ensemble methods** come in.

1. Ensemble Methods – What Are They?

Ensemble methods combine the predictions of **multiple models** to create a **stronger overall model**.

 Think of it like a **team**: one model might be weak, but a group of models can make a better decision.

2. Bagging (Bootstrap Aggregating)

What it does:

- It builds **many models (like decision trees)** on **different random samples** of the training data.
- Then, it combines their predictions by **voting** (for classification) or **averaging** (for regression).

Goal:

- **Reduce variance** (i.e., avoid overfitting).

Example:

- **Random Forest** is a popular algorithm that uses bagging with decision trees.
-

3. Boosting

What it does:

- Builds models **one at a time**, where each new model **tries to correct the mistakes** made by the previous model.
- The models are **combined** to make better predictions.


Goal:

- **Reduce bias and variance**, and increase accuracy.
-

4. AdaBoost (Adaptive Boosting)

Special Type of Boosting:

- Assigns **more weight** to **incorrectly predicted data points**, so the next model focuses more on them.
- Final prediction is a **weighted sum** of all models.

 **Works well with simple models like small decision trees (called stumps).**

5. Random Forest

What it is:

- A type of **bagging algorithm** that uses **multiple decision trees**.
- Each tree is trained on a **random subset of data and features**.

- Final prediction: **majority vote** (classification) or **average** (regression).

Why it's good:

- **Very accurate**
- Handles missing values
- **Less overfitting** than a single decision tree

Summary Table:

Technique	How It Works	Goal	Example Algorithm
Bagging	Train models on random data subsets & combine	Reduce overfitting	Random Forest
Boosting	Train models in sequence to fix errors	Improve accuracy	Gradient Boosting
AdaBoost	Focus on hard examples with weighted errors	Boost weak learners	AdaBoost
Random Forest	Multiple decision trees with bagging	Strong + stable model	Random Forest

4.3 Model Evaluation and Selection - Confusion Matrix, Dataset Partitioning Methods-Holdout Method and Random Subsampling, Cross Validation.

Model Evaluation and Selection

Once you've trained a machine learning model, you need to **test how well it performs**. This is done using evaluation techniques and metrics.

1. Confusion Matrix

A **confusion matrix** helps to evaluate a **classification model** by comparing **predicted vs actual values**.

✓ It looks like this:

	Predicted: Yes	Predicted: No
Actual: Yes	✓ True Positive (TP)	✗ False Negative (FN)
Actual: No	✗ False Positive (FP)	✓ True Negative (TN)

From this, we can calculate:

- **Accuracy** = $(TP + TN) / \text{Total}$
 - **Precision** = $TP / (TP + FP)$
 - **Recall** = $TP / (TP + FN)$
 - **F1-Score** = $2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$
-

2. Dataset Partitioning Methods

Before testing, we need to **split the dataset** into parts to train and evaluate the model.

a. Holdout Method

- **Split the data** into:

- **Training set** (e.g., 70%)
- **Testing set** (e.g., 30%)
- Train the model on the training set and evaluate it on the test set.

📌 **Simple, but results can vary** depending on the random split.

b. Random Subsampling

- Perform the **holdout method multiple times** with different random splits.
- Take the **average performance** across all splits.

📌 Better than single holdout, but still not the most reliable.

3. Cross-Validation (K-Fold)

- The **best method** for model evaluation.

How it works:

1. Split data into **K equal parts** (say $K = 5$).
2. Train the model **K times**, each time using a different part as test data and the rest as training data.
3. **Average the performance** over all K runs.

📌 **Advantage:** Uses all data for training and testing — gives **more reliable** results.

Summary Table

Method	Description	Pros	Cons
Confusion Matrix	Compares actual vs predicted classes	Gives detailed metrics	Only for classification tasks
Holdout Method	One-time split into train/test	Simple and quick	Can be inaccurate
Random Subsampling	Multiple random splits and averages	Reduces bias slightly	Still may miss some patterns
Cross-Validation	Uses all data in training and testing (K-fold)	Most reliable & robust	More time-consuming