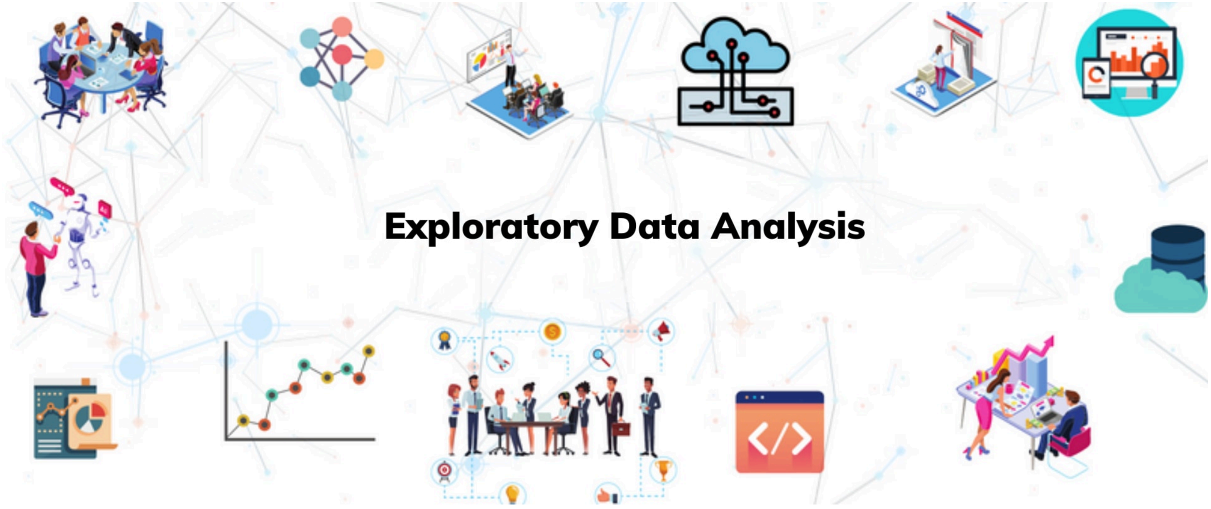


MOOC 1: Exploratory Data Analysis for Machine Learning

> **Original Notion link:** MOOC 1: Exploratory Data Analysis for Machine Learning

---

Mục lục

Module 1 — A Brief History of Modern AI and its Applications

1.1 Learning goals

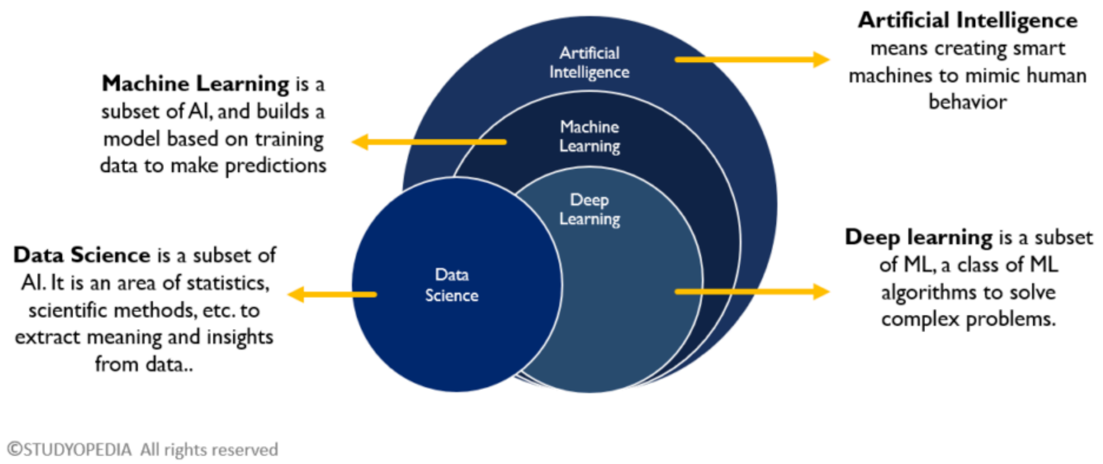
- Distinguish AI vs ML vs DL
- Understand the end-to-end ML workflow

1.2 Core concepts

Artificial Intelligence: A branch of computer science simulating intelligent behavior in computers.

Machine Learning: Learning patterns from data to make predictions without explicit programming.

Deep Learning: A subset of ML using multi-layer neural networks trained on large datasets.

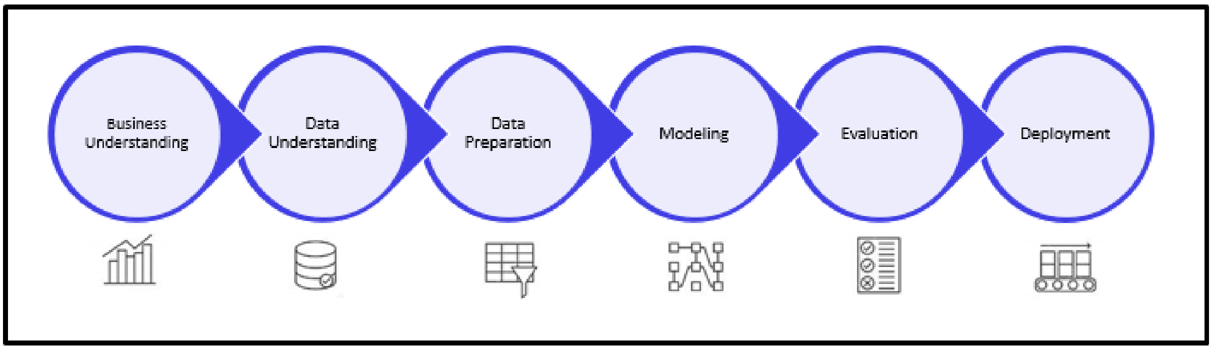


1.3 History and drivers

- Alternating cycles of AI winters and AI booms
- Modern applications: speech, vision, diagnosis, robotics
- Drivers: big data, faster compute, open-source, diverse NN architectures

1.4 ML workflow

1. Problem definition
2. Data collection
3. Data exploration and preprocessing
4. Modeling
5. Validation
6. Deployment



1.5 Data taxonomy

Term	Definition
Target	The category or value to predict
Features	Explanatory variables used for prediction
Example	A single observation or data point
Label	The target value for one observation

Module 2 — Retrieving and Cleaning Data

2.1 Learning goals

- Know common data sources and file formats
- Clean data, handle missing values and outliers

2.2 Sources and formats

- SQL, NoSQL, APIs, Cloud (AWS, GCP, Azure)
- Files: CSV, TSV, JSON

2.3 Read data — SQL

> Tip: Prefer parameterized queries. Close connections when done.

```
# Title: Read a full table from SQLite
import sqlite3
import pandas as pd

conn = sqlite3.connect('path/to/database.db')
query = "SELECT * FROM table_name"
df =
```

2.4 Read data — JSON

```
# Title: Load JSON from file and normalize to a DataFrame
import json
import pandas as pd

with open('data.json', 'r') as f:
    data = json.load(f)

df = pd.json_normalize(data)
# Or read directly with pandas
# df =
```

2.5 Cleaning — missing values

Strategy	Implementation	Use case
Removal	Drop rows or columns	Missing completely at random
Imputation	Mean, median, mode, or predictions	When deletion loses too much data
Masking	Flag or "Missing" category	When missingness is informative

```
# Title: Basic missing-value handling
# Count missing values
df.isna().sum()
```

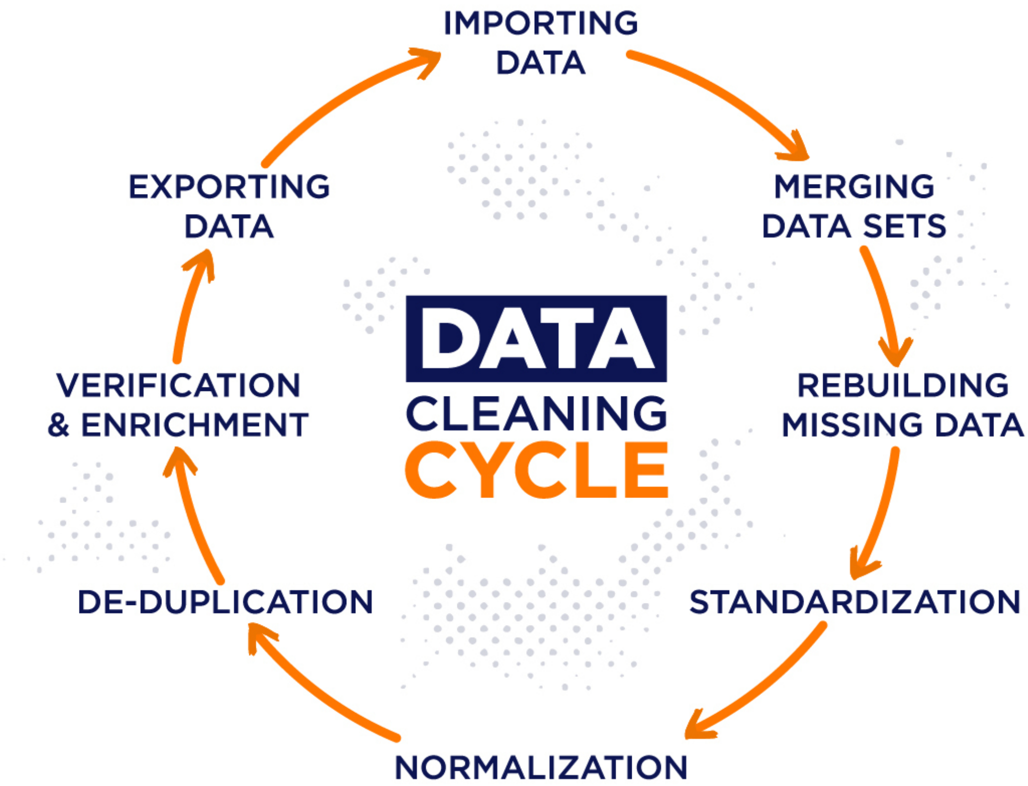
```
# Drop rows with missing
df_clean = df.dropna()

# Impute with statistics
df['column'] = df['column'].fillna(df['column'].median())
```

2.6 Cleaning — outliers

```
# Title: Detect outliers via Z-score
from scipy import stats
z = stats.zscore(df['column'])
outliers = df[abs(z) > 3]

# Title: Detect outliers via IQR
Q1 = df['column'].quantile(0.25)
Q3 = df['column'].quantile(0.75)
IQR = Q3 - Q1
lb, ub = Q1 - 1.5*IQR, Q3 + 1.5*IQR
outliers = df[(df['column'] < lb) | (df['column'] > ub)]
```



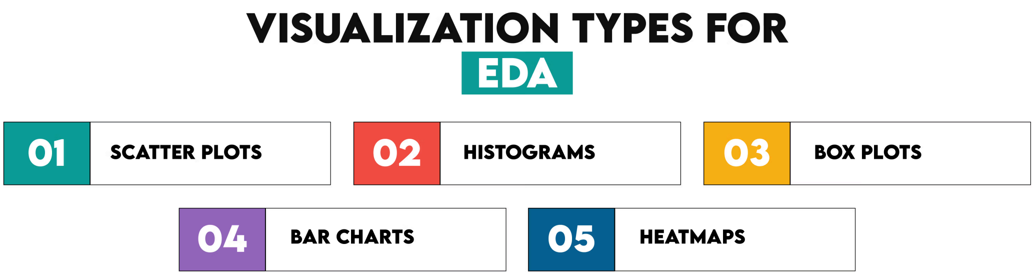
Module 3 — Exploratory Data Analysis and Feature Engineering

3.1 Learning goals

- Summarize data with statistics and visuals
- Spot patterns and hypotheses to guide feature engineering

3.2 EDA toolbox

- Summary stats: mean, median, range, variance
- Distributions: histograms, density plots
- Relationships: correlation matrices, scatter plots
- Time series: line plots
- Categorical: bar charts



3.3 Examples

```
# Title: Summary stats and correlation heatmap
summary = df.describe()

import seaborn as sns
import matplotlib.pyplot as plt
corr = df.corr(numeric_only=True)
plt.figure(figsize=(12, 10))
sns.heatmap(corr, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Matrix')
```

```
# Title: Pair plot by target
sns.pairplot(df[['col1', 'col2', 'col3', 'target']], hue='target')
```

Module 4 — Inferential Statistics and Hypothesis Testing

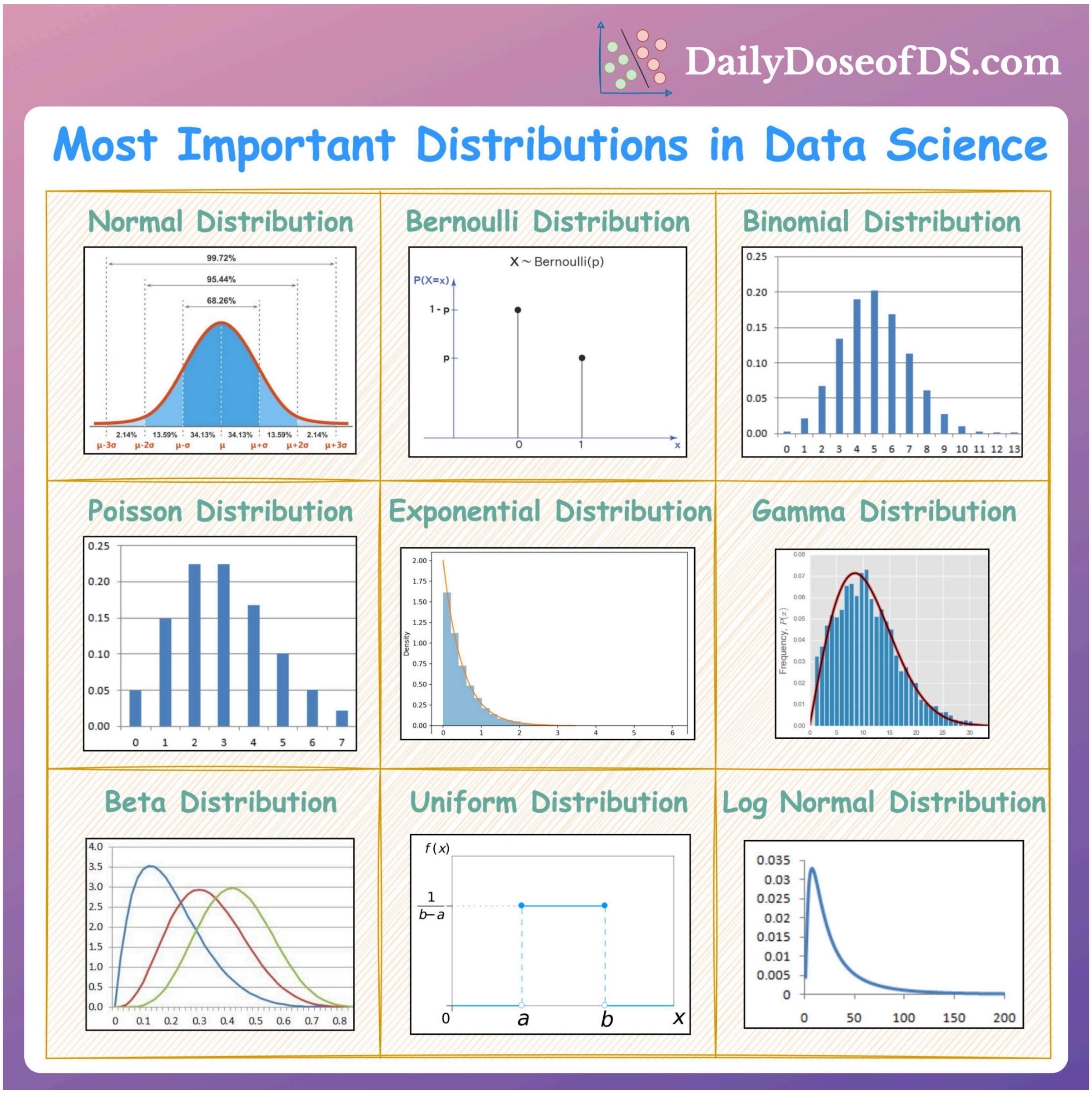
4.1 Learning goals

- Grasp estimation, inference, and hypothesis testing basics

4.2 Distribution quick table

Distribution	Properties	Applications
Uniform	Equal probability across range	Random sampling
Normal	Bell-shaped, defined by mean and std	Natural phenomena
Log-normal	log(x) is Normal	Income, asset prices
Exponential	Time between Poisson events	Failure times

Distribution	Properties	Applications
Poisson	Discrete counts	Rare events



4.3 Estimation and testing

$$\text{MLE: } \hat{\theta} = \arg\max_{\theta} L(\theta \mid data)$$

$$\text{Bayes' Theorem: } P(\theta \mid data) = \frac{P(data \mid \theta)\, P(\theta)}{P(data)}$$

```
# Title: t-tests and Pearson correlation
from scipy import stats

# One-sample t-test
t_stat, p = stats.ttest_1samp(df['column'], popmean=0)

# Two-sample t-test
t_stat2, p2 = stats.ttest_ind(group1, group2)

# Pearson correlation
r, p3 = stats.pearsonr(df['col1'], df['col2'])
```

$$\text{Pearson } r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$