### 🧾 Cell 1: Loading Libraries

python

CopyEdit

import pandas as pd

import matplotlib.pyplot as plt

* **Problem Step**: "Import all required Python Libraries."
* **Definition**:
  + import pandas as pd: Imports the **pandas** library, which is used for working with **dataframes** (tabular data).
  + import matplotlib.pyplot as plt: Imports **matplotlib**, a library used for **data visualization** like plots and graphs.
* **Why**: We'll use pandas to load, clean, and explore the dataset, and matplotlib for drawing graphs like boxplots.

**🧾 Cell 2: Load the Dataset**

python

CopyEdit

df = pd.read\_csv('datasets/AcademicPerformance\_New.csv')

* **Problem Step**: "Load the Dataset into pandas data frame."
* **Definition**:
  + pd.read\_csv() reads a **CSV (Comma Separated Values)** file and loads it into a pandas **DataFrame**.
* **Why**: This allows us to analyze and manipulate the dataset programmatically.

**🧾 Cell 3: View First Few Rows**

python

CopyEdit

df.head()

* **Problem Step**: Used in **data inspection** during preprocessing.
* **Definition**:
  + .head() displays the **first 5 rows** of the dataset.
* **Why**: Helps get a quick understanding of what the data looks like — column names, structure, and example values.

**🧾 Cell 4: Data Overview**

python

CopyEdit

df.info()

* **Problem Step**: "Check dimensions and variable types."
* **Definition**:
  + .info() shows the **number of entries**, **column names**, **non-null counts**, and **data types**.
* **Why**: Helps you detect **missing values** and understand what type each column is (e.g., integer, float, object).

**🧾 Cell 5: Handle Missing Values**

python

CopyEdit

df['WT'].fillna(df['WT'].mean(), inplace=True)

df['DSBDA'].fillna(df['DSBDA'].mean(), inplace=True)

df['AI'].fillna(df['DSBDA'].mean(), inplace=True)

* **Problem Step**: "Find and handle missing values in the data."
* **Definition**:
  + .fillna(value) replaces missing values (NaN) with the specified value.
  + .mean() calculates the **average** value of the column.
* **Why**: Missing values can cause errors. Here, we're filling them using the **mean** of the respective columns to keep the data consistent and usable.

### ✅ Cell 6: Check Dataset Again

python

CopyEdit

df.head()

* **Why**: To confirm if missing values are filled properly.

**✅ Cell 7: Drop Missing Roll Numbers**

python

CopyEdit

df.dropna(subset=['Roll No'], inplace=True)

* **Definition**:
  + dropna(subset=...) removes rows where 'Roll No' is missing.
* **Why**: 'Roll No' is a unique identifier and should not be null.

**✅ Cell 8: Confirm Deletion**

python

CopyEdit

df.head()

* **Why**: To verify that rows with missing Roll Numbers are removed.

**✅ Cell 9: Check for Duplicates**

python

CopyEdit

df.duplicated().sum()

* **Definition**: .duplicated().sum() counts duplicate rows.
* **Why**: Duplicate rows can bias the analysis.

**✅ Cell 10: Convert Roll No to Integer**

python

CopyEdit

df['Roll No'].astype(int)

* **Definition**: .astype(int) converts values to integers.
* **Why**: Ensures data type consistency for identification numbers.

**✅ Cell 11: Initial Boxplot for WT**

python

CopyEdit

plt.boxplot(df['WT'])

plt.title('WT')

plt.grid(True)

plt.show()

* **Definition**:
  + boxplot() plots a distribution with median, quartiles, and outliers.
* **Why**: Visual check for outliers in the WT column.

**✅ Cell 12: Z-Score Normalization (WT)**

python

CopyEdit

df['WT\_z'] = (df['WT'] - df['WT'].mean()) / df['WT'].std()

* **Definition**: Z-Score = (value - mean) / std deviation.
* **Why**: To identify how far each value is from the mean.

**✅ Cell 13: Remove Outliers (WT)**

python

CopyEdit

df = df[~((df['WT\_z'] < -2.5) | (df['WT\_z'] > 2.5))]

* **Definition**: Filters out values beyond ±2.5 standard deviations.
* **Why**: Removes outliers to clean data.

**✅ Cell 14: Post-Cleaning WT Boxplot**

python

CopyEdit

plt.boxplot(df['WT'])

plt.title('WT')

plt.grid(True)

plt.show()

* **Why**: Check the distribution after removing outliers.

**✅ Cell 15: Boxplot for DSBDA**

python

CopyEdit

plt.boxplot(df['DSBDA'])

plt.title('DSBDA')

plt.grid(True)

plt.show()

* **Why**: Check DSBDA distribution and outliers visually.

**✅ Cell 16: Z-Score (DSBDA)**

python

CopyEdit

df['DSBDA\_z'] = (df['DSBDA'] - df['DSBDA'].mean()) / df['DSBDA'].std()

* **Why**: To normalize and detect outliers in DSBDA scores.

**✅ Cell 17: Remove DSBDA Outliers**

python

CopyEdit

df = df[~((df['DSBDA\_z'] < -2.5) | (df['DSBDA\_z'] > 2.5))]

* **Why**: Clean data by removing outliers.

**✅ Cell 18: Boxplot after DSBDA Cleaning**

python

CopyEdit

plt.boxplot(df['DSBDA'])

plt.title('DSBDA')

plt.grid(True)

plt.show()

* **Why**: Check cleaned DSBDA distribution.

**✅ Cell 19: Boxplot for AI**

python

CopyEdit

plt.boxplot(df['AI'])

plt.title('AI')

plt.grid(True)

plt.show()

* **Why**: Visualize AI scores and outliers.

**✅ Cell 20: Z-Score (AI)**

python

CopyEdit

df['AI\_z'] = (df['AI'] - df['AI'].mean()) / df['AI'].std()

* **Why**: Normalize AI column to detect outliers.

**✅ Cell 21: Remove Outliers (AI)**

python

CopyEdit

df = df[~((df['AI\_z'] < -2.5) | (df['DSBDA\_z'] > 2.5))]

* **Note**: Likely a typo — should filter based on 'AI\_z', not 'DSBDA\_z'.

**✅ Cell 22: AI Boxplot After Cleaning**

python

CopyEdit

plt.boxplot(df['AI'])

plt.title('AI')

plt.grid(True)

plt.show()

* **Why**: Ensure AI column is cleaned correctly.

**✅ Cell 23: Drop Z-Score Columns**

python

CopyEdit

df.drop(['WT\_z','AI\_z','DSBDA\_z'], axis=1, inplace=True)

* **Definition**:
  + drop(columns) removes unnecessary columns.
* **Why**: Clean final dataset before further use.

**✅ Cell 24: Create Performance Categories**

python

CopyEdit

bins = [0, 40, 60, 80, 100]

labels = ['Low', 'Average', 'Good', 'Excellent']

df['Performance\_Level'] = pd.cut(df['Average'], bins=bins, labels=labels, right=False)

* **Definition**:
  + pd.cut() categorizes numerical data into bins.
* **Why**: Transforms numeric average scores into performance labels for interpretation.

**✅ Cell 25: Final Data Check**

python

CopyEdit

df.head()

* **Why**: Confirm performance categories added correctly.

**✅ Cell 26: Final Boxplot**

python

CopyEdit

plt.boxplot(df['Average'])

plt.title('Average')

plt.grid(True)

plt.show()

* **Why**: Visualize overall student average performance distribution.