**import** pandas **as** pd

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**from** IPython.core.interactiveshell **import** InteractiveShell

InteractiveShell**.**ast\_node\_interactivity **=** "all"

df **=** pd**.**read\_csv('Academic-Performance-Dataset.csv')

print("------------------Shape of Dataset------------------")

print(df**.**shape)

print("\n")

print("-----------------Data Types in Dataset-----------------")

print(df**.**dtypes)

print("\n")

print("=====================Data Cleanning===================\n")

print("-------------Handling Missing Values in Dataset-------------")

print(df**.**isnull()**.**sum())

print("\n"

print("--------------List of name of columns with missing values--------------")

cols\_with\_missing **=** df**.**columns[df**.**isnull()**.**any()]

print(cols\_with\_missing)

print("\n")

print("-------------Filling Missing Values with Mean/Mode Imputation-------------")

**for** col **in** cols\_with\_missing:

col\_dt **=** df[col]**.**dtype

**if** col\_dt **in** ['int64', 'float64']:

outliers **=** (df[col] **<** 0) **|** (df[col] **>** 100)

df**.**loc[outliers, col] **=** np**.**nan

df[col] **=** df[col]**.**fillna(df[col]**.**mean())

**else**:

df[col] **=** df[col]**.**ffill()

print(df**.**head())

print("\n")

print("---------------Correction in Percentage & Total Marks Columns---------------")

df['Total Marks']**=** (df['Phy\_marks']**+**df['Che\_marks']**+**df['EM1\_marks']**+**df['PPS\_marks']**+**df['SME\_marks'])**.**astype(int)

df['Percentage']**=** df['Total Marks']**/**5

print(df**.**head())

print("\n")

print("-------------After Handling Missing Values-------------")

print(df**.**isnull()**.**sum())

print("=================Handling Outliers===================\n")

print("-------------Identifying Outliers in Columns-------------")

plt**.**rcParams["figure.figsize"] **=** (8,5)

df\_list **=** ['Attendence', 'Phy\_marks', 'Che\_marks', 'EM1\_marks', 'PPS\_marks', 'SME\_marks']

fig, axes **=** plt**.**subplots(2, 3)

fig**.**set\_dpi(120)

count **=** 0

**for** r **in** range(2):

**for** c **in** range(3):

\_ **=** df[df\_list[count]]**.**plot(kind **=** 'box', ax**=**axes[r,c])

count**+=**1

plt**.**show()

print("-------------Removing Outliers from Che\_Makrs Column-------------")

q1 **=** df['Che\_marks']**.**quantile(0.25)

q3 **=** df['Che\_marks']**.**quantile(0.75)

Lower\_limit **=** q1 **-** 1.5 **\*** (q3 **-** q1)

Upper\_limit **=** q3 **+** 1.5 **\*** (q3 **-** q1)

print(f'q1 = {q1}, q3 = {q3}, IQR = {q3 **-**q1}, Lower\_limit = {Lower\_limit}, Upper\_limit = {Upper\_limit}')

print(df[(df['Che\_marks'] **<** Lower\_limit) **|** (df['Che\_marks'] **>** Upper\_limit)])

print("\n")

print("=====================Binning (convert into normal distribution)========================\n")

print("------------Gradding According to percentage")

**def** BinningFunction(column, cut\_points, labels **=** **None**) :

break\_points**=**[column**.**min()] **+** cut\_points **+** [column**.**max( )]

print('Gradding According to percentage \n<60 = F \n60-70 = B \n70-80 = A\n80-100 = O')

**return** pd**.**cut(column, bins**=**break\_points, labels**=**labels, include\_lowest**=True**)

cut\_points **=** [60, 70, 80]

labels **=** ['F', 'B', 'A', 'O']

df['Grade'] **=** BinningFunction(df['Percentage'], cut\_points, labels)

print(df**.**head(10))

print("\n")