Experiment No : 09

Aim : To perform data analysis using essential Python libraries (Pandas and NumPy) by importing, manipulating, and summarizing datasets through descriptive statistics.

Theory :

1. Data Loading
   - Importing Structured Data:
     Pandas reads structured datasets (e.g., CSV files) into a DataFrame, a tabular structure with rows (observations) and columns (variables). This enables efficient handling of heterogeneous data types (numeric, categorical).
   - Initial Exploration:
     Techniques like viewing the first few rows (head()) and dataset metadata (info()) help identify columns, data types (e.g., integers, floats), and non-null counts. This step ensures data integrity and suitability for analysis.
2. Data Manipulating
   - Feature Engineering:
     New variables are derived from existing data (e.g., BMI from height and weight) to enhance analytical depth. This step contextualizes raw data for domain-specific insights (e.g., health metrics).
   - Filtering Subsets:
     Logical conditions (e.g., filtering rows where age > 30) isolate relevant subsets, enabling focused analysis on specific groups.
   - Handling Missing Values:
     Identifying missing data (isnull().sum()) is critical for deciding mitigation strategies (e.g., imputation or removal).
   - Array Operations with NumPy:
     Data is converted to NumPy arrays for numerical efficiency. Operations like reshaping (converting 1D arrays to 2D) and concatenation (merging arrays column-wise) prepare data for advanced computations (e.g., machine learning).
3. Data Summarizing
   - Descriptive Statistics:
     - Central Tendency:
       - Mean (average value) and median (midpoint value) describe data centrality.
       - Mode (most frequent value) is critical for categorical data analysis.
     - Variability:
       - Standard deviation and variance quantify data spread.
       - Percentiles (25th, 75th) highlight distribution skewness and outlier thresholds.
   - Correlation and Covariance:
     - Correlation matrices measure linear relationships between variables (strength and direction).

    - Covariance matrices indicate how variables change together (positive/negative trends).
- Data Distribution Insights:
  Statistical summaries reveal patterns like skewness, outliers, and clusters, guiding hypotheses for deeper analysis.

Code :

```
import pandas as pd
import numpy as np

df = pd.read_csv('/content/synthetic_dataset.csv')
print("Loaded Dataset:")
print(df.head())

df['BMI'] = df['Weight'] / (df['Height'] / 100) ** 2

print("\nData Info (Data types, non-null counts):")
print(df.info())

print("\nSummary Statistics:")
print(df.describe())

print("\nMissing Values (null counts):")
print(df.isnull().sum())

df_older_than_30 = df[df['Age'] > 30]
print("\nFiltered Data (Age > 30):")
print(df_older_than_30)

print("\nDescriptive Statistics (Mean, Median, Mode):")
mean_values = df.mean()
median_values = df.median()
mode_values = df.mode().iloc[0]

print("Mean Values:")
print(mean_values)
print("\nMedian Values:")
print(median_values)
print("\nMode Values:")
print(mode_values)

print("\nStandard Deviations and Variances using NumPy:")
std_devs = np.std(df[['Age', 'Height', 'Salary', 'Weight', 'BMI']], axis=0)
variances = np.var(df[['Age', 'Height', 'Salary', 'Weight', 'BMI']], axis=0)
```

```
percentiles_25 = {col: np.percentile(df[col], 25) for col in ['Age', 'Height', 'Salary', 'Weight', 'BMI']}
percentiles_75 = {col: np.percentile(df[col], 75) for col in ['Age', 'Height', 'Salary', 'Weight', 'BMI']}

print("Standard Deviations:")
print(std_devs)
print("\nVariances:")
print(variances)
print("\n25th Percentiles:")
print(percentiles_25)
print("\n75th Percentiles:")
print(percentiles_75)

print("\nCorrelation Matrix:")
print(df.corr())

print("\nCovariance Matrix:")
print(df.cov())

ages_first_5 = df['Age'].values[5:7]
print("First 7 Ages:")
print(ages_first_5)

age_array = df['Age'].values.reshape(-1, 1)
print("\nReshaped Age Array (10 rows, 1 column):")
print(age_array)

salary_array = df['Salary'].values
concatenated_data = np.column_stack((age_array, salary_array))
print("\nConcatenated Age and Salary:")
print(concatenated_data)
```

Output:

```
Loaded Dataset:
   Age  Height  Salary  Weight        BMI
0   23     175   50000      70  22.857143
1   45     180   60000      80  24.691358
2   36     165   55000      60  22.038567
3   50     170   80000      75  25.951557
4   23     160   45000      55  21.484375
```

```
Data Info (Data types, non-null counts):
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Age     10 non-null     int64
 1   Height  10 non-null     int64
 2   Salary  10 non-null     int64
 3   Weight  10 non-null     int64
 4   BMI     10 non-null     float64
dtypes: float64(1), int64(4)
memory usage: 532.0 bytes
None
```

```
Summary Statistics:
             Age      Height        Salary     Weight        BMI
count  10.000000   10.000000     10.000000  10.000000  10.000000
mean   36.100000  172.200000  61000.000000  69.500000  23.392514
std    10.071412    6.494442  14491.376746   8.031189   1.896276
min    23.000000  160.000000  40000.000000  55.000000  20.061728
25%    29.250000  169.250000  51250.000000  65.750000  22.209035
50%    35.500000  172.500000  60000.000000  70.000000  23.259291
75%    43.750000  177.250000  68750.000000  74.250000  25.079741
max    50.000000  180.000000  85000.000000  80.000000  25.951557
```

```
Missing Values (null counts):
Age       0
Height    0
Salary    0
Weight    0
BMI       0
dtype: int64
```

```
Filtered Data (Age > 30):
   Age  Height  Salary  Weight        BMI
1   45     180   60000      80  24.691358
2   36     165   55000      60  22.038567
3   50     170   80000      75  25.951557
5   40     172   70000      70  23.661439
6   50     178   85000      80  25.249337
9   35     173   65000      68  22.720438
```

```
Descriptive Statistics (Mean, Median, Mode):
Mean Values:
Age          36.100000
Height      172.200000
Salary    61000.000000
Weight       69.500000
BMI          23.392514
dtype: float64
```

```
Median Values:
Age          35.500000
Height      172.500000
Salary    60000.000000
Weight       70.000000
BMI          23.259291
dtype: float64
```

```
Mode Values:
Age          23.000000
Height      180.000000
Salary    60000.000000
Weight       70.000000
BMI          20.061728
Name: 0, dtype: float64
```

```
Standard Deviations and Variances using NumPy:
Standard Deviations:
Age           9.554580
Height        6.161169
Salary    13747.727085
Weight        7.619055
BMI           1.798966
dtype: float64
```

```
Variances:
Age       9.129000e+01
Height    3.796000e+01
Salary    1.890000e+08
Weight    5.805000e+01
BMI       3.236277e+00
dtype: float64
```

```
25th Percentiles:
{'Age': np.float64(29.25), 'Height': np.float64(169.25), 'Salary': np.float64(51250.0), 'Weight': np.float64(65.75), 'BMI': np.float64(22.209035212537444)}

75th Percentiles:
{'Age': np.float64(43.75), 'Height': np.float64(177.25), 'Salary': np.float64(68750.0), 'Weight': np.float64(74.25), 'BMI': np.float64(25.07974052504473)}
```

```
Correlation Matrix:
             Age    Height    Salary    Weight       BMI
Age     1.000000  0.349599  0.844285  0.715003  0.694892
Height  0.349599  1.000000  0.422658  0.709382  0.130050
Salary  0.844285  0.422658  1.000000  0.553727  0.409766
Weight  0.715003  0.709382  0.553727  1.000000  0.789009
BMI     0.694892  0.130050  0.409766  0.789009  1.000000
```

```
 Covariance Matrix:
                   Age        Height        Salary        Weight           BMI
 Age        101.433333     22.866667  1.232222e+05     57.833333     13.271167
 Height      22.866667     42.177778  3.977778e+04     37.000000      1.601596
 Salary  123222.222222  39777.777778  2.100000e+08  64444.444444  11260.231931
 Weight      57.833333     37.000000  6.444444e+04     64.500000     12.016099
 BMI         13.271167      1.601596  1.126023e+04     12.016099      3.595863
```

```
First 7 Ages:
[40 50]
```

```
Reshaped Age Array (10 rows, 1 column):
[[23]
 [45]
 [36]
 [50]
 [23]
 [40]
 [50]
 [29]
 [30]
 [35]]
```

```
Concatenated Age and Salary:
[[   23 50000]
 [   45 60000]
 [   36 55000]
 [   50 80000]
 [   23 45000]
 [   40 70000]
 [   50 85000]
 [   29 60000]
 [   30 40000]
 [   35 65000]]
```

Conclusion :
Thus, we have successfully performed data analysis using Pandas and NumPy to import, manipulate, and summarize the dataset through descriptive statistics.