Devanshu Surana
PC-23, Panel C, Batch C1
1032210755

DEC Lab Assignment 1

Problem statement: Data Handling. Locate any open source data. Load data into data frame. Perform Data frame operations, Perform basic statistical operations like mean, median, standard deviation.

Objectives:

1) To explore various Data Sources and Data Repositories
2) To explore the operation on a dataset file using data frame with basic statistical operations in python.

Methodology

1) Identify and study various data sources.

A-1) Public Data Sources.
- Open data Portal
- Publicly Accessible websites
- Research publications

Private Data Sources
- Corporate Data
- Market Research firm
- Subscription services

Page No.
Date

Government Data Sources
- Government databases
- Census data
- Geospatial data
- Weather and environmental data

It is important to note that access to certain government and private data sources may be restricted or subject to privacy regulations and legal requirement

2) To study a dataset with various operation using Python command.

- Load Dataset
- Read csv file
- Display:
  Head: Displays first five values of dataset.
  Tail: Displays last five values of dataset
  Describe: Gets summary statistics for numeric cols.
  ✓ Summary: Depending upon your or specific analysis you can create a custom summary of your dataset using Pandas.

Data Handling.
Remove duplicates
By, 'drop_duplicates()' keeps the first occurence of each duplicated row and removes subsequent duplicates.

Teacher's Sign.: ____________

Identify and display missing values.

→ Using the 'isna()' or 'isnull()' both return a dataset frame of the same shape as the original. where each element is 'True' if the corresponding element in original Dataframe is missing (NaN or None) and 'false' otherwise.

Statistical basic operations to handle missing values.

→ for ex: rows with missing values : we can use the command dropna.

df. dropna (inplace = True).

Execution
Program implementation and output study

Conclusion :
Basic operations were performed on the csv data file using Python.

FAQ's

Q1) state the significance of handling missing values in a dataset.

→ Some significant reasons are :
1) Preservation of data Integrity
2) Avoidance of Bias.
3) Improved model ~~Prefer~~ Performance
4) Enhanced Data Visualisation

5) Reduced Noise in Analysis.

Q2) Explain Central tendency measures with examples.

→ 1) Mean : Avg of all values divided by the no. of values.

Ex : Consider scores of test = {85, 92, 88, 78, 95}

$$\text{mean} = \frac{85+92+88+78+95}{5} = 88{\cdot}6$$

mean is 88·6

2) Median : Middle value of a dataset when its arranged in ascending or descending order.

for dataset : 78, 85, 88, 92, 95 median is 88.

3) Mode : Mode is most frequently occured element in a dataset

for ex : In dataset (1, 1, 2, 2, 2, 3, 4, 4, 4, 4)
mode = 4

Q3) Describe various methods to handle missing values in a dataset.

→ Mean/Median/Mode imputation : Replace null/missing values with median/mode/mean.

K-nearest Neighbours (KNN) Imputation: Impute missing values by considering the values of their k-nearest neighbors in dataset.

Page No.
Date

- Create a New Category: for Categorised data, you can create a new category (for eg: "Unknown" or "others").

- Machine-learning based Imputation: Train machine learning models (eg: decision trees, random forests, or neutral networks) to predict missing values on other features in dataset.

Q4) Explain different datatype in Python.

→ Numeric:

int (integer): Represent whole numbers for ex: '+5', '-3'.

float : represent decimal point numbers for ex: '3.14' '-0.5'

Complex: represent complex with a real and imaginary part for ex: 3+2j.

Text

Str (string): represents text or sequence of characters enclosed in single or double quotes. for ex: "Hello World".

Boolean Data Type

bool (Boolean): Represents either True / False.

Sequence Types

List: Represents an ordered collection of items that can be of different data types.

Teacher's Sign.: ____________

Tuple : Represent an ordered collection of items, similar to lists. but tuples are immutable, meaning their elements cannot be changed once defined

range : Represents an immutable sequence of numbers and is often used for looping.

```
import numpy as np
import pandas as pd
```

```
ds=pd.read_csv("/content/Iris.csv")
```

```
ds.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

```
ds.tail()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

```
ds.describe()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

```
ds.value_counts("Species")
```

```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

```
ds.max()
```

```
Id                          150
SepalLengthCm               7.9
SepalWidthCm                4.4
PetalLengthCm               6.9
PetalWidthCm                2.5
Species          Iris-virginica
dtype: object
```

```
ds.min()
```

```
Id                        1
SepalLengthCm           4.3
SepalWidthCm            2.0
PetalLengthCm           1.0
PetalWidthCm            0.1
Species         Iris-setosa
dtype: object
```

```
ds.isnull()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
145	False	False	False	False	False	False
146	False	False	False	False	False	False
147	False	False	False	False	False	False
148	False	False	False	False	False	False
149	False	False	False	False	False	False

150 rows × 6 columns

```
ds.isnull().sum()
```

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

```
ds.to_numpy()
```

```
array([[1, 5.1, 3.5, 1.4, 0.2, 'Iris-setosa'],
       [2, 4.9, 3.0, 1.4, 0.2, 'Iris-setosa'],
       [3, 4.7, 3.2, 1.3, 0.2, 'Iris-setosa'],
       [4, 4.6, 3.1, 1.5, 0.2, 'Iris-setosa'],
       [5, 5.0, 3.6, 1.4, 0.2, 'Iris-setosa'],
       [6, 5.4, 3.9, 1.7, 0.4, 'Iris-setosa'],
       [7, 4.6, 3.4, 1.4, 0.3, 'Iris-setosa'],
       [8, 5.0, 3.4, 1.5, 0.2, 'Iris-setosa'],
       [9, 4.4, 2.9, 1.4, 0.2, 'Iris-setosa'],
       [10, 4.9, 3.1, 1.5, 0.1, 'Iris-setosa'],
       [11, 5.4, 3.7, 1.5, 0.2, 'Iris-setosa'],
       [12, 4.8, 3.4, 1.6, 0.2, 'Iris-setosa'],
       [13, 4.8, 3.0, 1.4, 0.1, 'Iris-setosa'],
       [14, 4.3, 3.0, 1.1, 0.1, 'Iris-setosa'],
       [15, 5.8, 4.0, 1.2, 0.2, 'Iris-setosa'],
       [16, 5.7, 4.4, 1.5, 0.4, 'Iris-setosa'],
       [17, 5.4, 3.9, 1.3, 0.4, 'Iris-setosa'],
       [18, 5.1, 3.5, 1.4, 0.3, 'Iris-setosa'],
       [19, 5.7, 3.8, 1.7, 0.3, 'Iris-setosa'],
       [20, 5.1, 3.8, 1.5, 0.3, 'Iris-setosa'],
       [21, 5.4, 3.4, 1.7, 0.2, 'Iris-setosa'],
       [22, 5.1, 3.7, 1.5, 0.4, 'Iris-setosa'],
       [23, 4.6, 3.6, 1.0, 0.2, 'Iris-setosa'],
       [24, 5.1, 3.3, 1.7, 0.5, 'Iris-setosa'],
       [25, 4.8, 3.4, 1.9, 0.2, 'Iris-setosa'],
       [26, 5.0, 3.0, 1.6, 0.2, 'Iris-setosa'],
       [27, 5.0, 3.4, 1.6, 0.4, 'Iris-setosa'],
       [28, 5.2, 3.5, 1.5, 0.2, 'Iris-setosa'],
       [29, 5.2, 3.4, 1.4, 0.2, 'Iris-setosa'],
       [30, 4.7, 3.2, 1.6, 0.2, 'Iris-setosa'],
       [31, 4.8, 3.1, 1.6, 0.2, 'Iris-setosa'],
       [32, 5.4, 3.4, 1.5, 0.4, 'Iris-setosa'],
       [33, 5.2, 4.1, 1.5, 0.1, 'Iris-setosa'],
       [34, 5.5, 4.2, 1.4, 0.2, 'Iris-setosa'],
       [35, 4.9, 3.1, 1.5, 0.1, 'Iris-setosa'],
       [36, 5.0, 3.2, 1.2, 0.2, 'Iris-setosa'],
       [37, 5.5, 3.5, 1.3, 0.2, 'Iris-setosa'],
       [38, 4.9, 3.1, 1.5, 0.1, 'Iris-setosa'],
       [39, 4.4, 3.0, 1.3, 0.2, 'Iris-setosa'],
       [40, 5.1, 3.4, 1.5, 0.2, 'Iris-setosa'],
       [41, 5.0, 3.5, 1.3, 0.3, 'Iris-setosa'],
       [42, 4.5, 2.3, 1.3, 0.3, 'Iris-setosa'],
       [43, 4.4, 3.2, 1.3, 0.2, 'Iris-setosa'],
       [44, 5.0, 3.5, 1.6, 0.6, 'Iris-setosa'],
       [45, 5.1, 3.8, 1.9, 0.4, 'Iris-setosa'],
       [46, 4.8, 3.0, 1.4, 0.3, 'Iris-setosa'],
       [47, 5.1, 3.8, 1.6, 0.2, 'Iris-setosa'],
       [48, 4.6, 3.2, 1.4, 0.2, 'Iris-setosa'],
       [49, 5.3, 3.7, 1.5, 0.2, 'Iris-setosa'],
       [50, 5.0, 3.3, 1.4, 0.2, 'Iris-setosa'],
       [51, 7.0, 3.2, 4.7, 1.4, 'Iris-versicolor'],
       [52, 6.4, 3.2, 4.5, 1.5, 'Iris-versicolor'],
       [53, 6.9, 3.1, 4.9, 1.5, 'Iris-versicolor'],
       [54, 5.5, 2.3, 4.0, 1.3, 'Iris-versicolor'],
       [55, 6.5, 2.8, 4.6, 1.5, 'Iris-versicolor'],
       [56, 5.7, 2.8, 4.5, 1.3, 'Iris-versicolor'],
       [57, 6.3, 3.3, 4.7, 1.6, 'Iris-versicolor'],
```

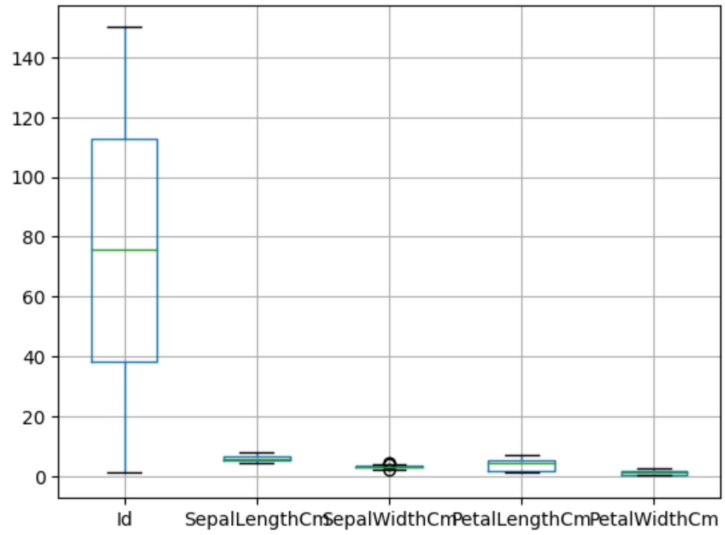
```
ds.duplicated()
```

```
    0      False
    1      False
    2      False
    3      False
    4      False
           ...
    145    False
    146    False
    147    False
    148    False
    149    False
    Length: 150, dtype: bool
```

```
import matplotlib.pyplot as plt
```

```
ds.boxplot()
```

```
    <Axes: >
```


Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm