Assignment No 1

Aim:

Perform the following operations using R/Python on suitable data sets:

a) read data from different formats (like csv, xls)

b) Find Shape of Data

c) Find Missing Values

d) Find data type of each column

e) Finding out Zero's

f) Indexing and selecting data, sort data,

g) Describe attributes of data, checking data types of each column,

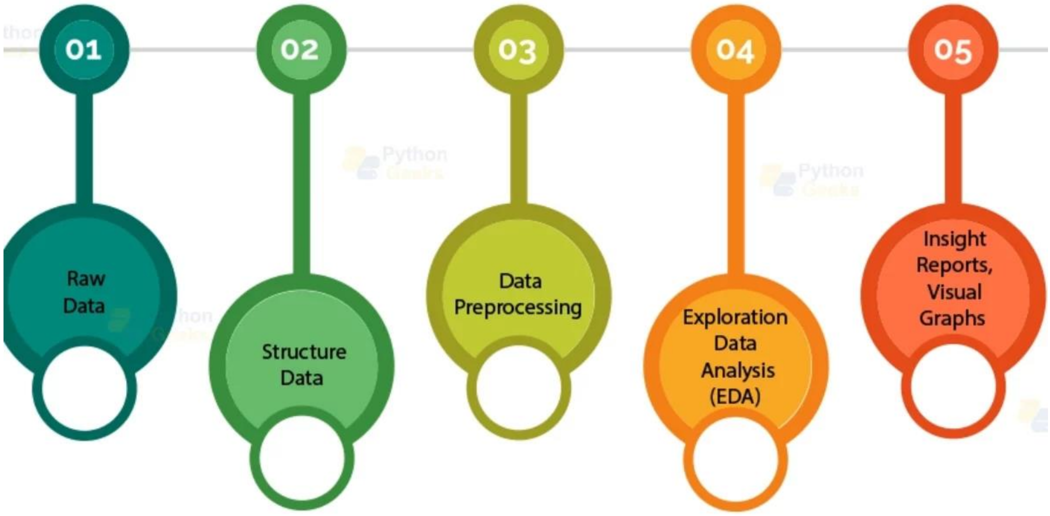
h) counting unique values of data, format of each column, converting variable

Theory:

What is Data Preprocessing?

Data preprocessing is a crucial step in the data analysis pipeline. It includes cleaning, transforming, and organizing raw data into a usable format. Without proper preprocessing, data analysis or machine learning models may yield inaccurate results.

Data is usually collected from multiple sources and stored in formats like CSV, XLS/XLSX, JSON, etc. Before conducting any statistical or machine learning task, we need to inspect, clean, and understand the data through exploratory techniques.


01
Raw Data
02
Structure Data
03
Data Preprocessing
04
Exploration Data Analysis (EDA)
05
Insight Reports, Visual Graphs

Methods and Explanations of Operations

a) Reading Data from Different Formats (CSV, XLS)

- - CSV (Comma-Separated Values): A plain text file format that stores tabular data.
- - XLS/XLSX: Excel spreadsheet files. Reading these formats requires additional libraries in Python like pandas with openpyxl or xlrd, and readxl in R.
- - Why Important: Helps bring external data into your working environment for processing and analysis.

b) Find Shape of Data

- - Represents the number of rows and columns in a dataset.
- - Syntax: (rows, columns)
- - Why Important: Understanding the shape is the first step in knowing what you're working with.

c) Find Missing Values

- - Real-world datasets often have missing entries.
- - Missing values are typically represented as NaN, NA, or blanks.
- - Why Important: They can skew analysis, so you need to either fill, impute, or drop them.

d) Find Data Type of Each Column

- - Helps in understanding how each column is interpreted: numerical, string, boolean, datetime, etc.
- - Why Important: Many operations are data-type sensitive. For instance, you can't compute the mean of a text field.

e) Finding Out Zeros

- - Zero values may or may not be significant.
- - Sometimes zeros indicate missing values or errors (especially in medical or survey data).
- - Why Important: Differentiating actual values from placeholders helps with data cleaning.

f) Indexing and Selecting Data, Sorting Data

- - Indexing: Refers to accessing specific rows or columns.
- - Selection: Filter data based on condition.
- - Sorting: Reorder data based on column values (ascending/descending).
- - Why Important: Gives flexibility in analyzing data slices or ordering it for better visibility.

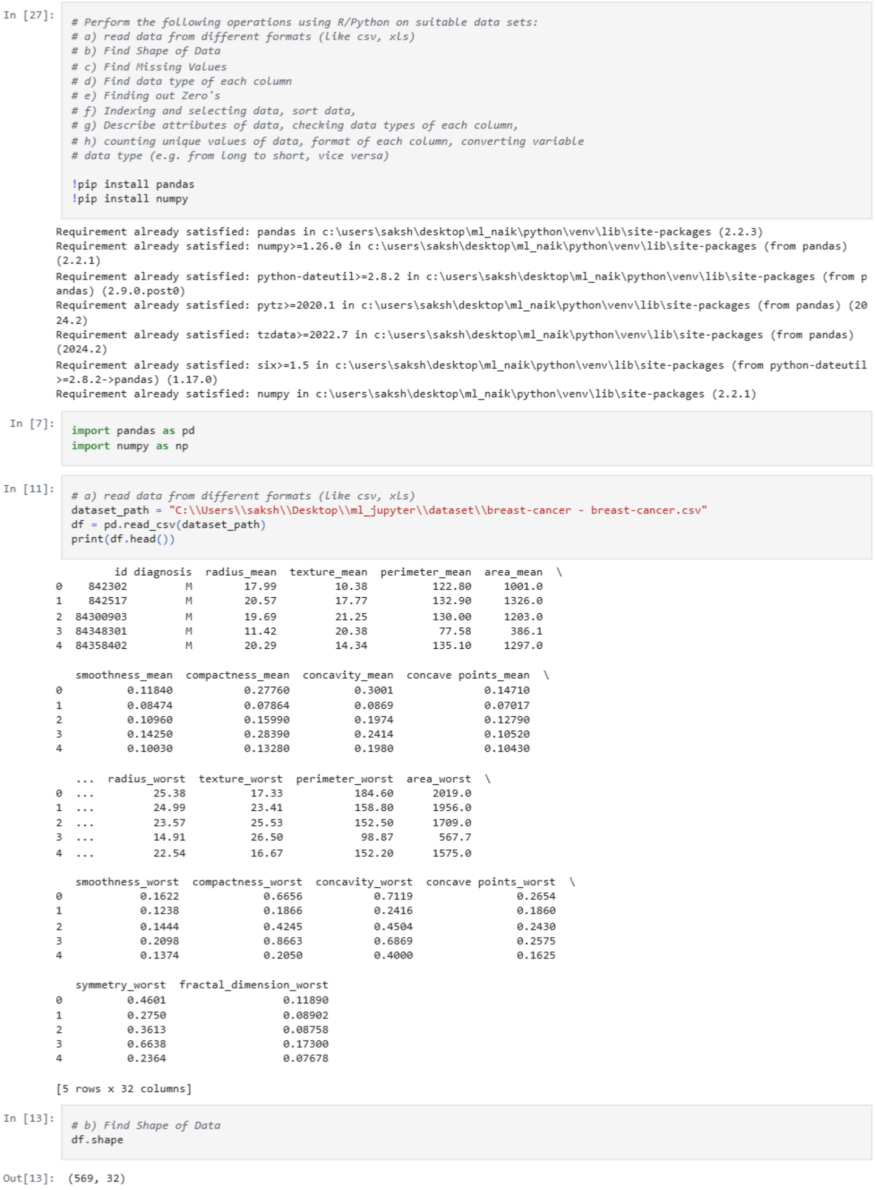
g) Describe Attributes of Data, Checking Data Types of Each Column

- - Includes mean, median, min, max, standard deviation, quartiles.
- - Helps to get a statistical summary of each column.
- - Why Important: Quickly provides distribution and spread of numerical data, helping to identify outliers and trends.

h) Counting Unique Values, Format of Each Column, Converting Variable Data Type

- - Unique Counts: Useful for categorical variables to know diversity.
- - Format: Consistency check (e.g., dates, strings).
- - Conversion: Sometimes needed for modeling (e.g., converting float to integer, or string to category).
- - Why Important: Improves model performance and ensures compatibility between tools and libraries.

Outputs:



```
In [27]:
# Perform the following operations using R/Python on suitable data sets:
# a) read data from different formats (like csv, xls)
# b) Find Shape of Data
# c) Find Missing Values
# d) Find data type of each column
# e) Finding out Zero's
# f) Indexing and selecting data, sort data,
# g) Describe attributes of data, checking data types of each column,
# h) counting unique values of data, format of each column, converting variable
# data type (e.g. from long to short, vice versa)

!pip install pandas
!pip install numpy
```

```
Requirement already satisfied: pandas in c:\users\saksh\desktop\ml_naik\python\venv\lib\site-packages (2.2.3)
Requirement already satisfied: numpy>=1.26.0 in c:\users\saksh\desktop\ml_naik\python\venv\lib\site-packages (from pandas)
(2.2.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\saksh\desktop\ml_naik\python\venv\lib\site-packages (from p
andas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\saksh\desktop\ml_naik\python\venv\lib\site-packages (from pandas) (20
24.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\saksh\desktop\ml_naik\python\venv\lib\site-packages (from pandas)
(2024.2)
Requirement already satisfied: six>=1.5 in c:\users\saksh\desktop\ml_naik\python\venv\lib\site-packages (from python-dateutil
>=2.8.2->pandas) (1.17.0)
Requirement already satisfied: numpy in c:\users\saksh\desktop\ml_naik\python\venv\lib\site-packages (2.2.1)
```

```
In [7]:
import pandas as pd
import numpy as np
```

```
In [11]:
# a) read data from different formats (like csv, xls)
dataset_path = "C:\\Users\\saksh\\Desktop\\ml_jupyter\\dataset\\breast-cancer - breast-cancer.csv"
df = pd.read_csv(dataset_path)
print(df.head())
```

```
         id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
0    842302         M        17.99         10.38          122.80     1001.0   
1    842517         M        20.57         17.77          132.90     1326.0   
2  84300903         M        19.69         21.25          130.00     1203.0   
3  84348301         M        11.42         20.38           77.58      386.1   
4  84358402         M        20.29         14.34          135.10     1297.0   

   smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0          0.11840           0.27760          0.3001              0.14710   
1          0.08474           0.07864          0.0869              0.07017   
2          0.10960           0.15990          0.1974              0.12790   
3          0.14250           0.28390          0.2414              0.10520   
4          0.10030           0.13280          0.1980              0.10430   

   ...  radius_worst  texture_worst  perimeter_worst  area_worst  \
0  ...         25.38          17.33           184.60      2019.0   
1  ...         24.99          23.41           158.80      1956.0   
2  ...         23.57          25.53           152.50      1709.0   
3  ...         14.91          26.50            98.87       567.7   
4  ...         22.54          16.67           152.20      1575.0   

   smoothness_worst  compactness_worst  concavity_worst  concave points_worst  \
0            0.1622             0.6656           0.7119                0.2654   
1            0.1238             0.1866           0.2416                0.1860   
2            0.1444             0.4245           0.4504                0.2430   
3            0.2098             0.8663           0.6869                0.2575   
4            0.1374             0.2050           0.4000                0.1625   

   symmetry_worst  fractal_dimension_worst  
0          0.4601                  0.11890  
1          0.2750                  0.08902  
2          0.3613                  0.08758  
3          0.6638                  0.17300  
4          0.2364                  0.07678  

[5 rows x 32 columns]
```

```
In [13]:
# b) Find Shape of Data
df.shape
```

```
Out[13]: (569, 32)
```

In [17]:

```python
# c) Find Missing Values
print("Sum of all null values:")
df.isnull().sum()
```

```
Sum of all null values:
```

Out[17]:

```
id                         0
diagnosis                  0
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
symmetry_mean              0
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
radius_worst               0
texture_worst              0
perimeter_worst            0
area_worst                 0
smoothness_worst           0
compactness_worst          0
concavity_worst            0
concave points_worst       0
symmetry_worst             0
fractal_dimension_worst    0
dtype: int64
```

In [21]:

```python
# d) Find data type of each column
print("Datatypes of each column:")
df.info()
```

```
Datatypes of each column:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       569 non-null    int64  
 1   diagnosis                569 non-null    object 
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
dtypes: float64(30), int64(1), object(1)
```

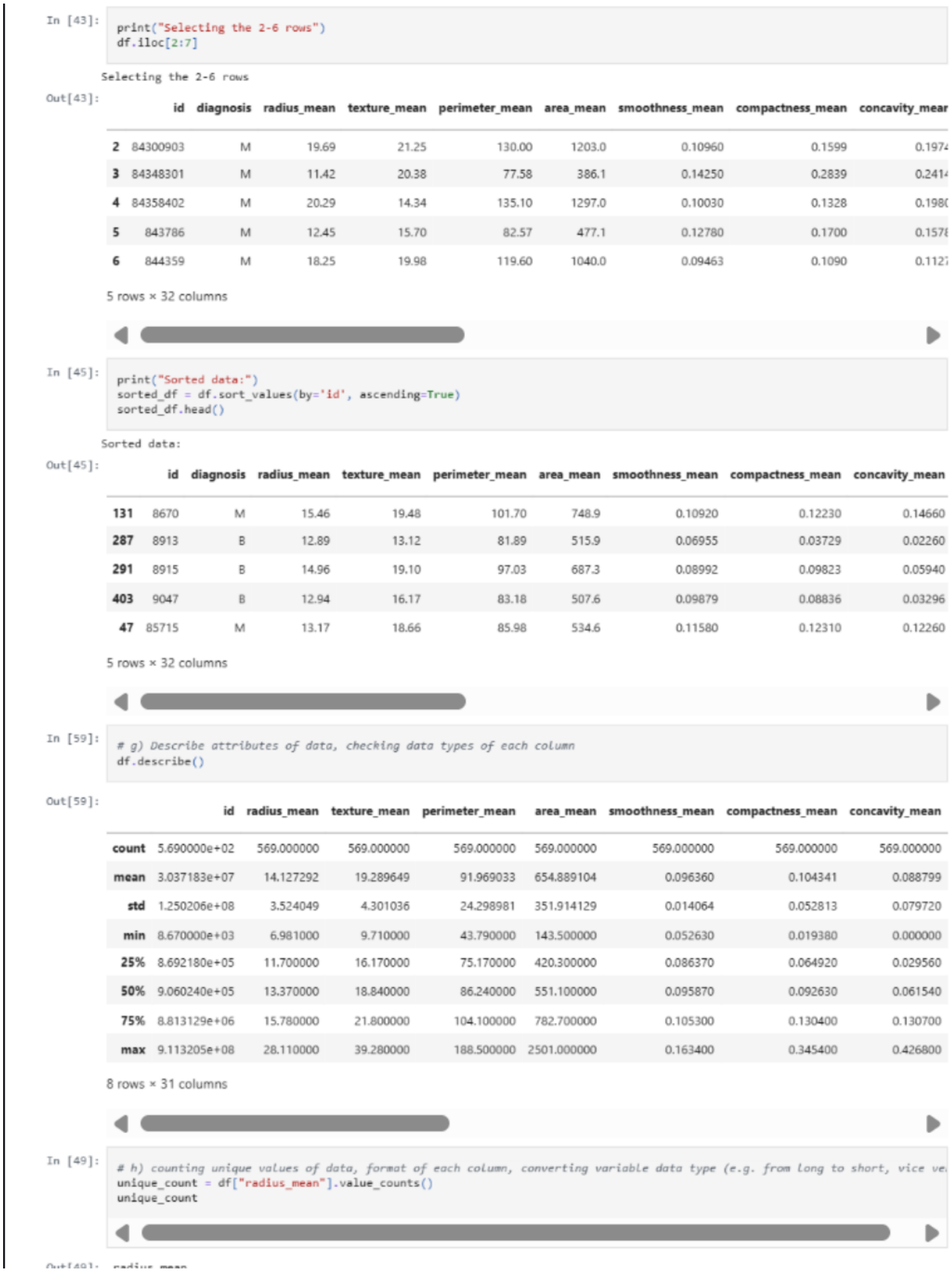
```
In [25]:
# e) Finding out Zero's
print("Finding no. of zero in column:")
num_of_zero=(df==0).sum()
print(num_of_zero)
```

```
Finding no. of zero in column:
id                          0
diagnosis                   0
radius_mean                 0
texture_mean                0
perimeter_mean              0
area_mean                   0
smoothness_mean             0
compactness_mean            0
concavity_mean             13
concave points_mean        13
symmetry_mean               0
fractal_dimension_mean      0
radius_se                   0
texture_se                  0
perimeter_se                0
area_se                     0
smoothness_se               0
compactness_se              0
concavity_se               13
concave points_se          13
symmetry_se                 0
fractal_dimension_se        0
radius_worst                0
texture_worst               0
perimeter_worst             0
area_worst                  0
smoothness_worst            0
compactness_worst           0
concavity_worst            13
concave points_worst       13
symmetry_worst              0
fractal_dimension_worst     0
dtype: int64
```

```
In [41]:
# f) Indexing and selecting data, sort data
print("Selecting the 1st row:")
df.iloc[0]
```

```
Selecting the 1st row:
```

```
Out[41]: id                           842302
         diagnosis                         M
         radius_mean                   17.99
         texture_mean                  10.38
         perimeter_mean                122.8
         area_mean                    1001.0
         smoothness_mean              0.1184
         compactness_mean             0.2776
         concavity_mean               0.3001
         concave points_mean          0.1471
         symmetry_mean                0.2419
         fractal_dimension_mean      0.07871
         radius_se                     1.095
         texture_se                   0.9053
         perimeter_se                  8.589
         area_se                       153.4
         smoothness_se              0.006399
         compactness_se              0.04904
         concavity_se                0.05373
         concave points_se           0.01587
         symmetry_se                 0.03003
         fractal_dimension_se       0.006193
         radius_worst                  25.38
         texture_worst                 17.33
         perimeter_worst               184.6
         area_worst                   2019.0
         smoothness_worst             0.1622
         compactness_worst            0.6656
         concavity_worst              0.7119
         concave points_worst         0.2654
         symmetry_worst               0.4601
         fractal_dimension_worst      0.1189
         Name: 0, dtype: object
```



In [43]:

```
print("Selecting the 2-6 rows")
df.iloc[2:7]
```

```
Selecting the 2-6 rows
```

Out[43]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mear
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.1599	0.1974
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.2839	0.2414
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.1328	0.1980
5	843786	M	12.45	15.70	82.57	477.1	0.12780	0.1700	0.1578
6	844359	M	18.25	19.98	119.60	1040.0	0.09463	0.1090	0.1127

5 rows × 32 columns

In [45]:

```
print("Sorted data:")
sorted_df = df.sort_values(by='id', ascending=True)
sorted_df.head()
```

```
Sorted data:
```

Out[45]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean
131	8670	M	15.46	19.48	101.70	748.9	0.10920	0.12230	0.14660
287	8913	B	12.89	13.12	81.89	515.9	0.06955	0.03729	0.02260
291	8915	B	14.96	19.10	97.03	687.3	0.08992	0.09823	0.05940
403	9047	B	12.94	16.17	83.18	507.6	0.09879	0.08836	0.03296
47	85715	M	13.17	18.66	85.98	534.6	0.11580	0.12310	0.12260

5 rows × 32 columns

In [59]:

```
# g) Describe attributes of data, checking data types of each column
df.describe()
```

Out[59]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	3.037183e+07	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799
std	1.250206e+08	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720
min	8.670000e+03	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000
25%	8.692180e+05	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560
50%	9.060240e+05	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540
75%	8.813129e+06	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700
max	9.113205e+08	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800

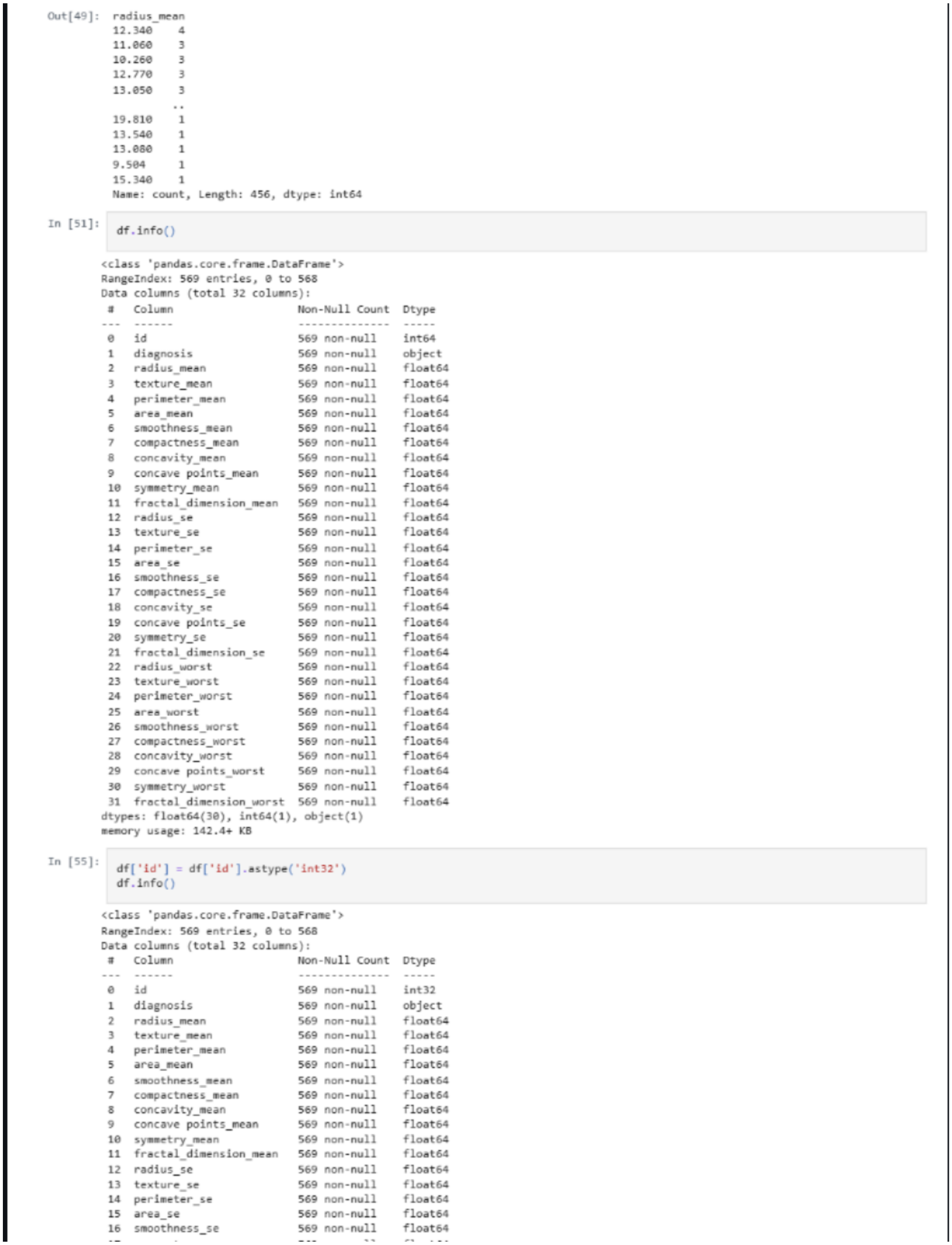
8 rows × 31 columns

In [49]:

```
# h) counting unique values of data, format of each column, converting variable data type (e.g. from long to short, vice ve
unique_count = df["radius_mean"].value_counts()
unique_count
```

Out[49]:

```
radius_mean
```



```
Out[49]: radius_mean
         12.340    4
         11.060    3
         10.260    3
         12.770    3
         13.050    3
                  ..
         19.810    1
         13.540    1
         13.080    1
         9.504     1
         15.340    1
         Name: count, Length: 456, dtype: int64
```

```
In [51]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   id                       569 non-null    int64
 1   diagnosis                569 non-null    object
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
dtypes: float64(30), int64(1), object(1)
memory usage: 142.4+ KB
```

```
In [55]: df['id'] = df['id'].astype('int32')
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   id                       569 non-null    int32
 1   diagnosis                569 non-null    object
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
```

Conclusion

We have successfully performed all essential data preprocessing tasks using R/Python as part of this assignment. We started with importing data from various formats and explored different techniques such as identifying missing values, checking and converting data types, indexing and sorting, detecting zeros, and summarizing attributes.Through this assignment, I have learned how crucial preprocessing is to prepare raw datasets for analysis or modeling. These skills are fundamental for ensuring the accuracy and reliability of results in any data-driven project.