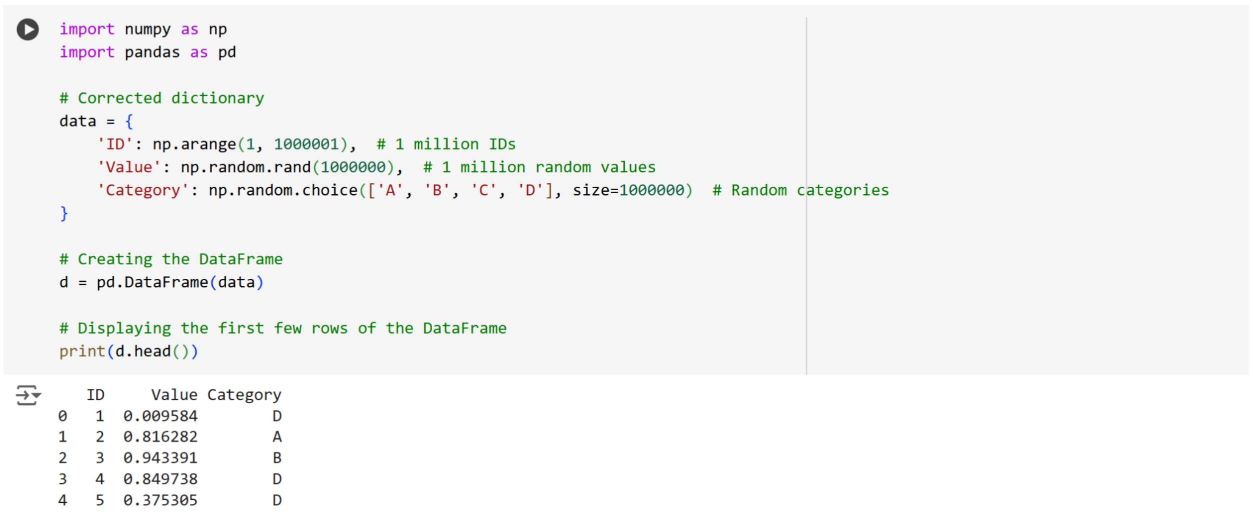
ICP3 REPORT



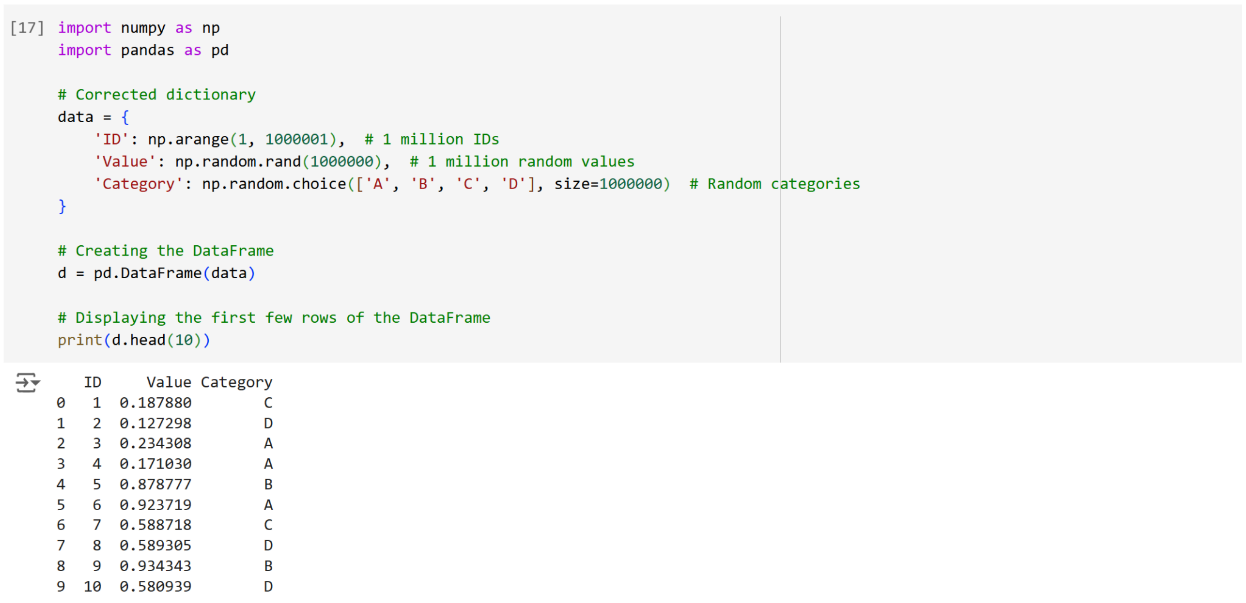
```python
import numpy as np
import pandas as pd

# Corrected dictionary
data = {
    'ID': np.arange(1, 1000001),  # 1 million IDs
    'Value': np.random.rand(1000000),  # 1 million random values
    'Category': np.random.choice(['A', 'B', 'C', 'D'], size=1000000)  # Random categories
}

# Creating the DataFrame
d = pd.DataFrame(data)

# Displaying the first few rows of the DataFrame
print(d.head())
```

```
   ID     Value Category
0   1  0.009584        D
1   2  0.816282        A
2   3  0.943391        B
3   4  0.849738        D
4   5  0.375305        D
```



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

# Corrected dictionary
data = {
    'ID': np.arange(1, 1000001),  # 1 million IDs
    'Value': np.random.rand(1000000),  # 1 million random values
    'Category': np.random.choice(['A', 'B', 'C', 'D'], size=1000000)  # Random categories
}

# Creating the DataFrame
d = pd.DataFrame(data)

# Displaying the first few rows of the DataFrame
print(d.head(10))
```

```
   ID     Value Category
0   1  0.187880        C
1   2  0.127298        D
2   3  0.234308        A
3   4  0.171030        A
4   5  0.878777        B
5   6  0.923719        A
6   7  0.588718        C
7   8  0.589305        D
8   9  0.934343        B
9  10  0.580939        D
```

```
[20] import numpy as np
     import pandas as pd

     # Corrected dictionary
     data = {
         'ID': np.arange(1, 1000001),  # 1 million IDs
         'Value': np.random.rand(1000000),  # 1 million random values
         'Category': np.random.choice(['A', 'B', 'C', 'D'], size=1000000)  # Random categories
     }

     # Creating the DataFrame
     d = pd.DataFrame(data)

     # Displaying the first few rows of the DataFrame
     print(d['Value'].head(10))
```

```
0    0.097612
1    0.424642
2    0.591496
3    0.880452
4    0.006170
5    0.600627
6    0.494344
7    0.269559
8    0.161614
9    0.938304
Name: Value, dtype: float64
```

```
import numpy as np
import pandas as pd

# Create the dictionary
data = {
    'ID': np.arange(1, 1000001),  # 1 million IDs
    'Value': np.random.rand(1000000),  # 1 million random values
    'Category': np.random.choice(['A', 'B', 'C', 'D'], size=1000000)  # Random categories
}

# Create the DataFrame
d = pd.DataFrame(data)

# Rename the columns
d.rename(columns={
    'ID': 'ID number',
    'Value': 'Random value',
    'Category': 'Choice'
}, inplace=True)

# Display the first five rows
print(d.head())
```

```
   ID number  Random value Choice
0          1      0.855382      D
1          2      0.583019      B
2          3      0.324775      C
3          4      0.861327      C
4          5      0.653837      B
```

```
import pandas as pd

# Set display options
pd.set_option('display.max_rows', None)

# Create DataFrame with corrected syntax
student_data = pd.DataFrame({
    'school_code': ['s001', 's002', 's003', 's001', 's002', 's004'],
    'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],
    'name': ['Alberto Franco', 'Gino Mcneill', 'Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],
    'date_of_birth': ['15/05/2002', '17/05/2002', '16/02/1999', '25/09/1998', '11/05/2002', '15/09/1997'],
    'age': [12, 12, 13, 13, 14, 12],
    'height': [173, 192, 186, 167, 151, 159],
    'weight': [35, 32, 33, 30, 31, 32],
    'address': ['street1', 'street2', 'street3', 'street4', 'street5', 'street6']
}, index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

print("Original Dataframe:")
print(student_data)

print('\nSplit the said data on school_code, class wise:')

# Group by 'school_code' and 'class'
result = student_data.groupby(['school_code', 'class'])

for name, group in result:
    print("\nGroup:")
    print(name)
    print(group)
```

```
Original Dataframe:
   school_code class            name date_of_birth  age  height  weight  \
S1        s001     V  Alberto Franco    15/05/2002   12     173      35
S2        s002     V    Gino Mcneill    17/05/2002   12     192      32
S3        s003    VI     Ryan Parkes    16/02/1999   13     186      33
S4        s001    VI    Eesha Hinton    25/09/1998   13     167      30
S5        s002     V    Gino Mcneill    11/05/2002   14     151      31
S6        s004    VI    David Parkes    15/09/1997   12     159      32
```

[23]

```
    address
S1  street1
S2  street2
S3  street3
S4  street4
S5  street5
S6  street6

Split the said data on school_code, class wise:

Group:
('s001', 'V')
   school_code class            name date_of_birth  age  height  weight  \
S1        s001     V  Alberto Franco    15/05/2002   12     173      35

    address
S1  street1

Group:
('s001', 'VI')
   school_code class          name date_of_birth  age  height  weight  address
S4        s001    VI  Eesha Hinton    25/09/1998   13     167      30  street4

Group:
('s002', 'V')
   school_code class          name date_of_birth  age  height  weight  address
S2        s002     V  Gino Mcneill    17/05/2002   12     192      32  street2
S5        s002     V  Gino Mcneill    11/05/2002   14     151      31  street5

Group:
('s003', 'VI')
   school_code class         name date_of_birth  age  height  weight  address
S3        s003    VI  Ryan Parkes    16/02/1999   13     186      33  street3

Group:
('s004', 'VI')
   school_code class          name date_of_birth  age  height  weight  address
S6        s004    VI  David Parkes    15/09/1997   12     159      32  street6
```

[36]
```python
import pandas as pd
# Load the CSV file
file_path = '/content/data.csv'
d = pd.read_csv(file_path)

# Show the first few rows of the DataFrame to understand its structure
d.head()
```

	Duration	Pulse	Maxpulse	Calories
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0

Next steps: Generate code with d View recommended plots New interactive sheet

[32]
```python
basic_stats = d.describe()
print("The statistical discription:")
print(basic_stats)
```

```
The statistical discription:
            ID number    Random value
count  1000000.000000  1000000.000000
mean    500000.500000        0.499997
std     288675.278932        0.288617
min          1.000000        0.000002
25%     250000.750000        0.249924
50%     500000.500000        0.499795
75%     750000.250000        0.749826
max    1000000.000000        1.000000
```

[33]
```python
null_values = d.isnull().sum()
print("\nNull Values in Each Column:")
print(null_values)

# a. Replace the null values with the mean
for col in d.columns:
    if pd.api.types.is_numeric_dtype(d[col]):
        d[col].fillna(d[col].mean(), inplace=True)
```

```

Null Values in Each Column:
ID number       0
Random value    0
Choice          0
dtype: int64
```

[37]
```python
aggregation = d.agg({
    'Pulse': ['min', 'max', 'count', 'mean'],
    'Calories': ['min', 'max', 'count', 'mean']
})
print("\nAggregation of Pulse and Calories:")
print(aggregation)
```

```

Aggregation of Pulse and Calories:
            Pulse     Calories
min     80.000000    50.300000
max    159.000000  1860.400000
count  169.000000   164.000000
mean   107.461538   375.790244
```

```
[38] filtered_d_500_1000 = d[(d['Calories'] >= 500) & (d['Calories'] <= 1000)]
     print("\nRows with Calories lying in  between 500 and 1000:")
     print(filtered_d_500_1000)
```

```
Rows with Calories lying in  between 500 and 1000:
     Duration  Pulse  Maxpulse  Calories
51         80    123       146     643.1
62        160    109       135     853.0
65        180     90       130     800.4
66        150    105       135     873.4
67        150    107       130     816.0
72         90    100       127     700.0
73        150     97       127     953.2
75         90     98       125     563.2
78        120    100       130     500.4
83        120    100       130     500.0
90        180    101       127     600.1
99         90     93       124     604.1
101        90     90       110     500.0
102        90     90       100     500.0
103        90     90       100     500.4
106       180     90       120     800.3
108        90     90       120     500.3
```

```
[42] filtered_d_calories_pulse = d[(d['Calories'] > 500) & (d['Pulse'] < 100)]
     print("\nRows with Calories > 500 and Pulses < 100:")
     print(filtered_d_calories_pulse)
```

```
Rows with Calories > 500 and Pulses < 100:
     Duration  Pulse  Maxpulse  Calories
65        180     90       130     800.4
70        150     97       129    1115.0
73        150     97       127     953.2
75         90     98       125     563.2
99         90     93       124     604.1
103        90     90       100     500.4
106       180     90       120     800.3
108        90     90       120     500.3
```

```
[43] d_modified = d.drop(columns=['Maxpulse'])
     print("DataFrame 'd_modified' with out 'Maxpulse':")
     print(d_modified.head())
```

```
DataFrame 'd_modified' with out 'Maxpulse':
   Duration  Pulse  Calories
0        60    110     409.1
1        60    117     479.0
2        60    103     340.0
3        45    109     282.4
4        45    117     406.0
```

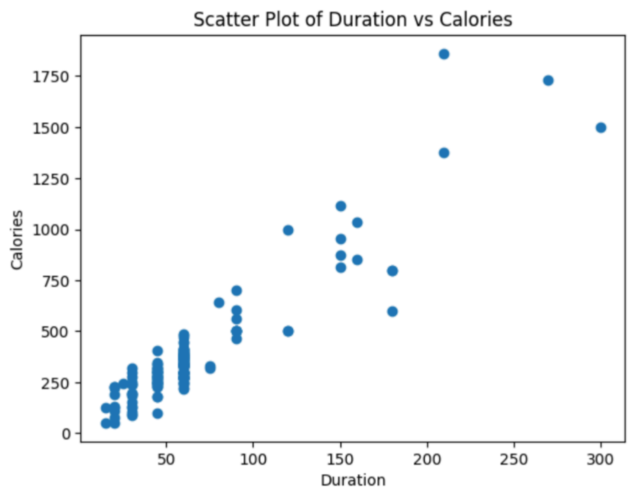
```
[44] d.drop(columns=['Maxpulse'], inplace=True)
     print("\nUpdated DataFrame 'd' without 'Maxpulse':")
     print(d.head())
```

```
Updated DataFrame 'd' without 'Maxpulse':
   Duration  Pulse  Calories
0        60    110     409.1
1        60    117     479.0
2        60    103     340.0
3        45    109     282.4
4        45    117     406.0
```

```
[46] d['Calories'] = d['Calories'].fillna(0).astype(int)
     print("\nDataFrame with 'Calories' as integer:")
     print(d.dtypes)
```

```
DataFrame with 'Calories' as integer:
Duration     int64
Pulse        int64
Calories     int64
dtype: object
```

```
[47] import matplotlib.pyplot as plt
     plt.scatter(df['Duration'], df['Calories'])
     plt.title('Scatter Plot of Duration vs Calories')
     plt.xlabel('Duration')
     plt.ylabel('Calories')
     plt.show()
```



My GitHub repository link:- https://github.com/akshaykumarpathem/bda.git

My youtube link:-https://youtu.be/oo8vOGoGyIk