[1]:
```
import pandas as pd
```

[2]:
```
import numpy as np
```

[3]:
```
df= pd.read_csv("/home/admin1/Downloads/income.csv")
```

[4]:
```
df.shape
```

[4]:
```
(1500, 5)
```

[5]:
```
df.head()
```

[5]:

	ID	Income	Age	Education	Gender
0	1	113	69	12	1
1	2	91	52	18	0
2	3	121	65	14	0
3	4	81	58	12	0
4	5	68	31	16	1

[6]:
```
df.tail()
```

[6]:

	ID	Income	Age	Education	Gender
1495	1496	84	50	16	0
1496	1497	81	40	16	0
1497	1498	62	30	16	1
1498	1499	99	67	16	0
1499	1500	91	57	16	1

[7]:
```
df.describe()
```

[7]:

	ID	Income	Age	Education	Gender
count	1500.000000	1500.000000	1500.000000	1500.000000	1500.000000
mean	750.500000	75.986000	43.582000	14.681333	0.490000
std	433.157015	20.005215	15.169466	2.693812	0.500067
min	1.000000	14.000000	18.000000	10.000000	0.000000
25%	375.750000	62.000000	30.000000	12.000000	0.000000
50%	750.500000	76.000000	44.000000	15.000000	0.000000
75%	1125.250000	91.000000	57.000000	16.000000	1.000000
max	1500.000000	134.000000	70.000000	20.000000	1.000000

[9]:
```
def summary_statistics_by_group(df, categorical_var, numeric_var):
    # Group by the categorical variable and calculate summary statistics
    grouped = df.groupby(categorical_var)[numeric_var].describe()

    # Calculate additional statistics such as standard deviation and range
    grouped['standard_deviation'] = df.groupby(categorical_var)[numeric_var].std()
    grouped['range'] = df.groupby(categorical_var)[numeric_var].apply(lambda x: x.max() - x.min())

    return grouped
```

[11]:
```
# Call the function with the dataframe, categorical variable, and numeric variable
result = summary_statistics_by_group(df, 'Age', 'Income')

# Display the results
print(result)
```

```
     count        mean         std   min    25%    50%     75%     max  \
Age
18    17.0   50.470588   10.380978  30.0  43.00   49.0   57.00   71.0
19    33.0   53.666667   10.899159  32.0  48.00   54.0   60.00   75.0
20    31.0   52.645161   13.182181  18.0  44.50   52.0   60.50   73.0
21    37.0   55.594595   11.875417  32.0  48.00   55.0   64.00   79.0
22    31.0   50.806452   15.751020  14.0  38.00   51.0   63.50   89.0
23    27.0   55.703704   13.294923  31.0  47.00   58.0   61.00   92.0
24    27.0   61.185185   14.639349  36.0  53.00   59.0   67.50   98.0
25    34.0   54.941176   14.180527  16.0  48.25   57.5   65.25   79.0
26    33.0   58.151515   12.833358  29.0  48.00   58.0   68.00   81.0
27    29.0   59.103448   14.859689  26.0  50.00   58.0   69.00   81.0
28    32.0   57.593750   12.714887  30.0  47.75   58.0   68.25   87.0
29    32.0   60.625000   10.416333  42.0  53.75   61.5   66.75   88.0
30    27.0   67.851852   15.881057  34.0  61.50   70.0   77.00   99.0
31    31.0   61.580645   11.820249  42.0  54.00   61.0   68.00   91.0
32    23.0   64.260870   13.056581  35.0  55.00   66.0   74.50   87.0
33    22.0   65.590909   14.516523  26.0  57.25   67.0   73.50   92.0
38    30.0   72.500000   12.686539  48.0  65.25   71.5   78.75  102.0
39    26.0   70.692308   13.376903  37.0  62.50   69.0   75.75  105.0
40    22.0   70.909091   12.701365  43.0  65.00   70.5   77.75   96.0
41    36.0   75.694444   12.621568  45.0  68.75   77.0   88.00   99.0
42    31.0   70.516129   13.306317  43.0  62.00   70.0   79.00  100.0
43    28.0   75.821429   15.405755  51.0  63.00   73.5   91.00  103.0
44    25.0   76.480000   13.401244  54.0  68.00   74.0   82.00  114.0
45    30.0   74.133333   11.509317  50.0  66.25   73.0   83.25  100.0
46    28.0   79.821429    9.217751  50.0  74.25   80.5   86.25   95.0
47    20.0   77.400000   11.094807  53.0  72.25   80.0   85.00   91.0
48    36.0   80.722222   13.582786  53.0  70.75   81.5   91.00  105.0
49    20.0   80.300000   14.885882  54.0  68.25   79.5   91.00  105.0
50    43.0   84.023256   11.839177  60.0  75.50   84.0   92.50  107.0
51    24.0   80.666667    8.716135  61.0  74.75   80.0   88.25   96.0
52    36.0   85.027778   13.357686  55.0  80.25   86.0   93.25  110.0
53    36.0   84.555556   12.164220  59.0  77.00   82.0   93.00  119.0
54    24.0   86.208333   13.213890  43.0  81.00   87.0   95.00  104.0
55    21.0   87.619048   10.331874  69.0  81.00   84.0   99.00  103.0
56    33.0   89.151515   15.091639  63.0  77.00   92.0   99.00  126.0
57    29.0   90.965517   11.422042  65.0  87.00   91.0   99.00  110.0
58    25.0   90.200000   12.409674  63.0  81.00   92.0   99.00  112.0
59    28.0   94.178571   13.463403  64.0  83.75   94.5  103.50  116.0
60    33.0   96.393939   13.303992  74.0  88.00   94.0  102.00  130.0
61    25.0   91.000000   13.168776  67.0  85.00   88.0   99.00  118.0
62    24.0   95.166667   13.321009  60.0  87.00   97.5  102.00  119.0
63    34.0   94.205882   11.951365  71.0  84.50   97.0  102.50  121.0
64    30.0   98.900000   14.935147  60.0  91.50   99.5  107.25  127.0
65    26.0   96.807692   11.210778  81.0  86.25   95.0  103.75  121.0
66    32.0   95.500000   11.927199  74.0  89.75   96.0  103.25  128.0
67    23.0   99.782609   12.830916  69.0  91.00  101.0  105.00  123.0
68    32.0   98.968750   13.587196  69.0  90.75   99.0  105.25  134.0
69    24.0  104.708333   12.037908  85.0  94.50  104.0  113.25  124.0
70    13.0  102.153846   11.480753  89.0  94.00   98.0  111.00  128.0
```

Age	standard_deviation	range
18	10.380978	41
19	10.899159	43
20	13.182181	55
21	11.875417	47
22	15.751020	75
23	13.294923	61
24	14.639349	62
25	14.180527	63
26	12.833358	52
27	14.859689	55
28	12.714887	57
29	10.416333	46
30	15.881057	65
31	11.820249	49
32	13.056581	52
33	14.516523	66
53	12.164220	60
54	13.213890	61
55	10.331874	34
56	15.091639	63
57	11.422042	45
58	12.409674	49
59	13.463403	52
60	13.303992	56
61	13.168776	51
62	13.321009	59
63	11.951365	50
64	14.935147	67
65	11.210778	40
66	11.927199	54
67	12.830916	54
68	13.587196	65
69	12.037908	39
70	11.480753	39

[12]:
```
# Create a list of incomes grouped by Age Group
income_list = df.groupby('Age')['Income'].apply(list)

print(income_list)
```

```
Age
18    [60, 56, 40, 44, 65, 54, 50, 49, 43, 39, 60, 3...
19    [58, 32, 52, 42, 49, 36, 67, 65, 70, 50, 75, 4...
20    [41, 52, 70, 45, 37, 44, 55, 67, 52, 70, 51, 6...
21    [44, 77, 62, 64, 68, 50, 48, 48, 56, 33, 58, 4...
22    [58, 64, 61, 38, 72, 41, 57, 62, 65, 44, 89, 6...
23    [44, 50, 58, 38, 59, 52, 61, 61, 62, 54, 39, 5...
24    [98, 59, 61, 69, 66, 53, 58, 81, 55, 49, 50, 4...
25    [41, 66, 34, 71, 51, 61, 69, 39, 55, 71, 48, 7...
26    [47, 41, 75, 56, 68, 74, 59, 77, 48, 49, 46, 7...
27    [26, 76, 69, 69, 44, 51, 66, 79, 79, 45, 69, 5...
28    [61, 45, 87, 51, 47, 71, 55, 54, 47, 33, 50, 4...
29    [70, 50, 61, 51, 65, 56, 57, 66, 49, 65, 74, 6...
30    [75, 87, 64, 61, 84, 82, 76, 99, 76, 70, 50, 3...
31    [68, 62, 49, 56, 48, 57, 44, 73, 64, 61, 48, 4...
32    [55, 84, 87, 73, 49, 55, 57, 75, 66, 54, 63, 5...
33    [53, 71, 68, 62, 74, 92, 65, 68, 66, 51, 83, 6...
34    [46, 68, 50, 84, 62, 67, 85, 82, 59, 92, 55, 7...
35    [70, 37, 95, 62, 38, 75, 48, 96, 85, 94, 63, 6...
36    [69, 65, 81, 52, 65, 77, 59, 83, 67, 61, 51, 5...
37    [91, 62, 71, 63, 87, 84, 34, 51, 57, 59, 83, 3...
38    [74, 70, 55, 102, 98, 83, 86, 71, 72, 64, 73, ...
39    [64, 73, 105, 67, 66, 37, 75, 81, 57, 78, 75, ...
40    [69, 51, 43, 77, 93, 72, 68, 78, 55, 73, 63, 7...
41    [58, 83, 82, 78, 88, 78, 89, 97, 63, 88, 64, 7...
42    [56, 51, 64, 100, 79, 77, 62, 81, 56, 73, 52, ...
43    [98, 63, 84, 54, 85, 51, 72, 91, 60, 73, 80, 6...
44    [67, 68, 60, 99, 73, 67, 91, 73, 114, 54, 60, ...
45    [67, 70, 89, 61, 73, 71, 66, 67, 67, 84, 81, 7...
46    [88, 50, 76, 78, 81, 90, 72, 87, 78, 90, 79, 7...
```

```
47    [84, 85, 89, 67, 79, 78, 89, 80, 91, 53, 65, 8...
48    [72, 101, 86, 98, 83, 85, 66, 66, 75, 61, 87, ...
49    [78, 105, 91, 101, 74, 61, 54, 87, 64, 69, 86,...
50    [77, 65, 81, 66, 80, 76, 72, 86, 92, 87, 87, 6...
51    [92, 74, 75, 96, 72, 87, 77, 89, 81, 76, 72, 8...
52    [91, 99, 87, 104, 90, 86, 110, 81, 81, 84, 82,...
53    [75, 74, 83, 98, 72, 93, 69, 119, 88, 96, 81, ...
54    [84, 82, 79, 93, 97, 95, 100, 43, 95, 61, 77, ...
55    [81, 99, 94, 86, 80, 84, 76, 102, 101, 77, 83,...
56    [76, 96, 96, 92, 65, 95, 92, 103, 112, 84, 103...
57    [91, 95, 94, 72, 99, 99, 101, 70, 101, 101, 11...
58    [81, 101, 63, 99, 103, 89, 98, 79, 89, 92, 99,...
59    [116, 94, 83, 97, 92, 90, 84, 109, 99, 91, 92,...
60    [106, 94, 109, 79, 80, 88, 89, 85, 93, 90, 99,...
61    [115, 88, 118, 107, 87, 97, 86, 75, 107, 74, 1...
62    [97, 87, 91, 100, 119, 100, 115, 102, 88, 60, ...
63    [103, 71, 97, 74, 80, 86, 106, 100, 96, 89, 10...
64    [89, 119, 102, 96, 88, 60, 110, 97, 103, 127, ...
65    [121, 85, 114, 115, 100, 86, 86, 108, 93, 84, ...
66    [105, 92, 108, 90, 94, 89, 102, 96, 76, 79, 97...
67    [86, 97, 108, 115, 123, 104, 102, 106, 69, 117...
68    [102, 108, 113, 91, 90, 101, 93, 105, 83, 89, ...
69    [113, 104, 90, 124, 104, 124, 90, 99, 124, 87,...
70    [89, 108, 112, 95, 128, 114, 96, 111, 93, 99, ...
Name: Income, dtype: object
```

2. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of 'Iris-setosa', 'Iris-versicolor' and 'Iris-versicolor' of iris.csv dataset

[12]:
```
df = pd.read_csv(r"C:\\Users\\UNKNOWN_CODER\\DSDBA\\Assign3\\iris.csv")
```

[13]:
```
df.head()
```

[13]:

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

[14]:
```
grouped_species = df.groupby('Species')
```

[15]:
```
print("\nBasic Statistical Details of each Species:")
for species, group in grouped_species:
    print(f"\nStatistics for {species}:")
    numeric_group = group.select_dtypes(include=[np.number])
    # Calculate and display basic statistics: mean, standard deviation, min, and max
    print("Mean:")
    print(numeric_group.mean())

    print("\nStandard Deviation:")
    print(numeric_group.std())

    print("\nMinimum Values:")
    print(numeric_group.min())

    print("\nMaximum Values:")
    print(numeric_group.max())

    print("\nPercentiles:")
    percentiles = np.percentile(numeric_group, [25, 50, 75], axis=0)
    print(f"25th Percentile: {percentiles[0]}")
    print(f"50th Percentile (Median): {percentiles[1]}")
    print(f"75th Percentile: {percentiles[2]}")
    print("="*40)
```

```
Basic Statistical Details of each Species:

Statistics for Iris-setosa:
Mean:
SepalLengthCm    5.006
SepalWidthCm     3.418
PetalLengthCm    1.464
PetalWidthCm     0.244
dtype: float64
```

```
Standard Deviation:
SepalLengthCm    0.352490
SepalWidthCm     0.381024
PetalLengthCm    0.173511
PetalWidthCm     0.107210
dtype: float64
Maximum Values:
SepalLengthCm    7.0
SepalWidthCm     3.4
PetalLengthCm    5.1
PetalWidthCm     1.8
dtype: float64

Percentiles:
25th Percentile: [5.6   2.525 4.    1.2  ]
50th Percentile (Median): [5.9  2.8  4.35 1.3 ]
75th Percentile: [6.3 3.  4.6 1.5]
========================================

Statistics for Iris-virginica:
Mean:
SepalLengthCm    6.588
SepalWidthCm     2.974
PetalLengthCm    5.552
PetalWidthCm     2.026
dtype: float64

Standard Deviation:
SepalLengthCm    0.635880
SepalWidthCm     0.322497
PetalLengthCm    0.551895
PetalWidthCm     0.274650
dtype: float64

Minimum Values:
SepalLengthCm    4.9
SepalWidthCm     2.2
PetalLengthCm    4.5
PetalWidthCm     1.4
dtype: float64

Maximum Values:
SepalLengthCm    7.9
SepalWidthCm     3.8
PetalLengthCm    6.9
PetalWidthCm     2.5
dtype: float64

Percentiles:
25th Percentile: [6.225 2.8   5.1   1.8  ]
50th Percentile (Median): [6.5  3.   5.55 2.  ]
75th Percentile: [6.9   3.175 5.875 2.3  ]
========================================

Minimum Values:
SepalLengthCm    4.3
SepalWidthCm     2.3
PetalLengthCm    1.0
PetalWidthCm     0.1
dtype: float64

Maximum Values:
SepalLengthCm    5.8
SepalWidthCm     4.4
PetalLengthCm    1.9
PetalWidthCm     0.6
dtype: float64

Percentiles:
25th Percentile: [4.8   3.125 1.4   0.2  ]
50th Percentile (Median): [5.  3.4 1.5 0.2]
75th Percentile: [5.2   3.675 1.575 0.3  ]
========================================

Statistics for Iris-versicolor:
Mean:
SepalLengthCm    5.936
SepalWidthCm     2.770
PetalLengthCm    4.260
PetalWidthCm     1.326
dtype: float64

Standard Deviation:
SepalLengthCm    0.516171
SepalWidthCm     0.313798
PetalLengthCm    0.469911
PetalWidthCm     0.197753
dtype: float64

Minimum Values:
SepalLengthCm    4.9
SepalWidthCm     2.0
PetalLengthCm    3.0
PetalWidthCm     1.0
dtype: float64
```