p3

March 28, 2024

[ ]:
```
import pandas as pd
df= pd.read_csv('dataSet.csv')
df
```

[ ]:

	roll	name	class	marks	age
0	0	anil	TE	56.77	22
1	1	amit	TE	59.77	21
2	2	aniket	BE	76.88	19
3	3	ajinkya	TE	69.66	20
4	4	asha	TE	63.28	20
5	5	ayesha	BE	49.55	20
6	6	amar	BE	65.34	19
7	7	amita	BE	68.33	23
8	8	amol	TE	56.75	20
9	9	anmol	BE	78.66	21

[51]:
```
print("Mean is: \m", df.mean())
```

```
Mean is:  roll      4.500
marks    64.499
age      20.500
dtype: float64
```

```
<ipython-input-51-b5400c99e283>:1: FutureWarning: The default value of
numeric_only in DataFrame.mean is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  print("Mean is: ", df.mean())
```

[53]:
```
print("Median is: \n", df.median())
```

```
Median is:
 roll      4.50
marks    64.31
age      20.00
dtype: float64
```

```
<ipython-input-53-10a8908516d2>:1: FutureWarning: The default value of
numeric_only in DataFrame.median is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  print("Median is: \n", df.median())
```

[54]:
```
print("Standard Deviation is: \n", df.std())
```

```
Standard Deviation is:
 roll     3.027650
marks    9.207179
age      1.269296
dtype: float64
```

```
<ipython-input-54-ca3f7944f318>:1: FutureWarning: The default value of
numeric_only in DataFrame.std is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  print("Standard Deviation is: \n", df.std())
```

[55]:
```
print("Max value :\n", df.max())
```

```
Max value :
 roll          9
name     ayesha
class        TE
marks     78.66
age          23
dtype: object
```

[56]:
```
print("Min value: \n ", df.min())
```

```
Min value:
  roll           0
name     ajinkya
class         BE
marks      49.55
age           19
dtype: object
```

[57]:
```
import numpy as np
print("Standard Deviation of marks: ", np.std(df['marks']))
```

```
Standard Deviation of marks:  8.734696846485285
```

[58]:
```
# Group by class and TE
gr1 = df.groupby('class')
te = gr1.get_group('TE')
```

[59]:
```
print("Min of TE: \n", te.min())
```

```
Min of TE:
 roll           0
name      ajinkya
class          TE
marks       56.75
age            20
dtype: object
```

[60]:
```
print("Max of TE: \n", te.max())
```

```
Max of TE:
 roll         8
name      asha
class       TE
marks    69.66
age         22
dtype: object
```

[62]:
```
# Grouping by age
gr2 = df.groupby('age')
print("Grouping by age: ", gr2.groups)
```

```
Grouping by age:  {19: [2, 6], 20: [3, 4, 5, 8], 21: [1, 9], 22: [0], 23: [7]}
```

[64]:
```
import seaborn as sns
# we can load the iris example dataset from seaborn
df1 = sns.load_dataset('iris')
df1
```

[64]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
..	…	…	…	…	…
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

```
[150 rows x 5 columns]
```

[66]:
```
# Grouping by species
gr = df1.groupby("species")

# grouping by species types
se = gr.get_group('setosa')
ve = gr.get_group('versicolor')
vi = gr.get_group('virginica')
```

[67]:
```
print("Shape of setosa: ", se.shape)
```

```
Shape of setosa:  (50, 5)
```

[68]:
```
print("Shape of versicolor: ", se.shape)
```

```
Shape of versicolor:  (50, 5)
```

[70]:
```
print("Shape of virginica: ", vi.shape)
```

```
Shape of virginica:  (50, 5)
```

[71]:
```
se.describe()
```

[71]:

	sepal_length	sepal_width	petal_length	petal_width
count	50.00000	50.000000	50.000000	50.000000
mean	5.00600	3.428000	1.462000	0.246000
std	0.35249	0.379064	0.173664	0.105386
min	4.30000	2.300000	1.000000	0.100000
25%	4.80000	3.200000	1.400000	0.200000
50%	5.00000	3.400000	1.500000	0.200000
75%	5.20000	3.675000	1.575000	0.300000
max	5.80000	4.400000	1.900000	0.600000

[72]:
```
ve.describe()
```

[72]:

	sepal_length	sepal_width	petal_length	petal_width
count	50.000000	50.000000	50.000000	50.000000
mean	5.936000	2.770000	4.260000	1.326000
std	0.516171	0.313798	0.469911	0.197753
min	4.900000	2.000000	3.000000	1.000000
25%	5.600000	2.525000	4.000000	1.200000
50%	5.900000	2.800000	4.350000	1.300000
75%	6.300000	3.000000	4.600000	1.500000
max	7.000000	3.400000	5.100000	1.800000

[73]:
```
vi.describe()
```

[73]:

	sepal_length	sepal_width	petal_length	petal_width
count	50.00000	50.000000	50.000000	50.00000
mean	6.58800	2.974000	5.552000	2.02600
std	0.63588	0.322497	0.551895	0.27465
min	4.90000	2.200000	4.500000	1.40000
25%	6.22500	2.800000	5.100000	1.80000
50%	6.50000	3.000000	5.550000	2.00000
75%	6.90000	3.175000	5.875000	2.30000
max	7.90000	3.800000	6.900000	2.50000