Pandas

In [2]:

```
import pandas as pd
import numpy as np
import seaborn as sns
from scipy import stats
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler
```

In [3]:

```
data=pd.read_csv("DataStudents.csv")
df=pd.DataFrame(data)
df
```

Out[3]:

	RollNo	Name	Age	Maths	Science	English	Total	Average
0	1	Anish	18	25.0	35	56	116.0	NaN
1	2	Aasawari	20	33.0	78	34	145.0	48.3
2	3	Aditya	11	90.0	33	22	145.0	48.3
3	4	Apoorva	22	45.0	88	56	189.0	63.0
4	5	Anil	12	66.0	43	88	197.0	65.7
5	6	Bablu	13	NaN	35	53	110.0	NaN
6	7	Chetan	12	56.0	66	78	NaN	66.7
7	8	Chinmay	14	56.0	36	68	160.0	53.3
8	9	Chitra	15	67.0	22	32	121.0	40.3
9	10	Diya	16	54.0	78	88	220.0	73.3
10	11	Deepak	34	33.0	44	71	148.0	49.3
11	12	Dilip	18	87.0	89	37	213.0	NaN
12	13	Esha	19	89.0	53	40	182.0	60.7
13	14	Fatima	17	34.0	45	78	157.0	52.3
14	15	Faiz	11	32.0	51	27	110.0	36.7
15	16	Gaurav	19	80.0	30	28	138.0	46.0
16	17	Hitesh	20	NaN	74	30	NaN	64.7
17	18	Isha	14	50.0	90	56	196.0	65.3
18	19	Ishan	16	40.0	42	89	171.0	57.0
19	20	Jay	18	48.0	72	90	210.0	70.0

In [4]:

```
df.describe()
```

Out[4]:

	RollNo	Age	Maths	Science	English	Total	Average
count	20.00000	20.000000	18.000000	20.000000	20.000000	18.000000	17.000000
mean	10.50000	16.950000	54.722222	55.200000	56.050000	162.666667	56.523529
std	5.91608	5.155222	21.051214	22.018174	23.811707	36.328566	10.714799
min	1.00000	11.000000	25.000000	22.000000	22.000000	110.000000	36.700000
25%	5.75000	13.750000	35.500000	35.750000	33.500000	139.750000	48.300000
50%	10.50000	16.500000	52.000000	48.000000	56.000000	158.500000	57.000000
75%	15.25000	19.000000	66.750000	75.000000	78.000000	194.250000	65.300000
max	20.00000	34.000000	90.000000	90.000000	90.000000	220.000000	73.300000

In [5]:

```
df.isnull().head(7)
```

Out[5]:

	RollNo	Name	Age	Maths	Science	English	Total	Average
0	False	False	False	False	False	False	False	True
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
5	False	False	False	True	False	False	False	True
6	False	False	False	False	False	False	True	False

In [6]:

```
df.head(10)
```

Out[6]:

	RollNo	Name	Age	Maths	Science	English	Total	Average
0	1	Anish	18	25.0	35	56	116.0	NaN
1	2	Aasawari	20	33.0	78	34	145.0	48.3
2	3	Aditya	11	90.0	33	22	145.0	48.3
3	4	Apoorva	22	45.0	88	56	189.0	63.0
4	5	Anil	12	66.0	43	88	197.0	65.7
5	6	Bablu	13	NaN	35	53	110.0	NaN
6	7	Chetan	12	56.0	66	78	NaN	66.7
7	8	Chinmay	14	56.0	36	68	160.0	53.3
8	9	Chitra	15	67.0	22	32	121.0	40.3
9	10	Diya	16	54.0	78	88	220.0	73.3

Filling Null Values

In [7]:

```
df['Maths'] = df['Maths'].fillna(df['Maths'].mean())
df['Science'] = df['Science'].fillna(df['Science'].mean())
df['English'] = df['English'].fillna(df['English'].mean())
df['Total'] = df['Total'].fillna(df['Total'].mean())
df['Average'] = df['Average'].fillna(df['Average'].mean())
df.head(7)
```

Out[7]:

	RollNo	Name	Age	Maths	Science	English	Total	Average
0	1	Anish	18	25.000000	35	56	116.000000	56.523529
1	2	Aasawari	20	33.000000	78	34	145.000000	48.300000
2	3	Aditya	11	90.000000	33	22	145.000000	48.300000
3	4	Apoorva	22	45.000000	88	56	189.000000	63.000000
4	5	Anil	12	66.000000	43	88	197.000000	65.700000
5	6	Bablu	13	54.722222	35	53	110.000000	56.523529
6	7	Chetan	12	56.000000	66	78	162.666667	66.700000

Detecting Outliers

BoxPlot Method

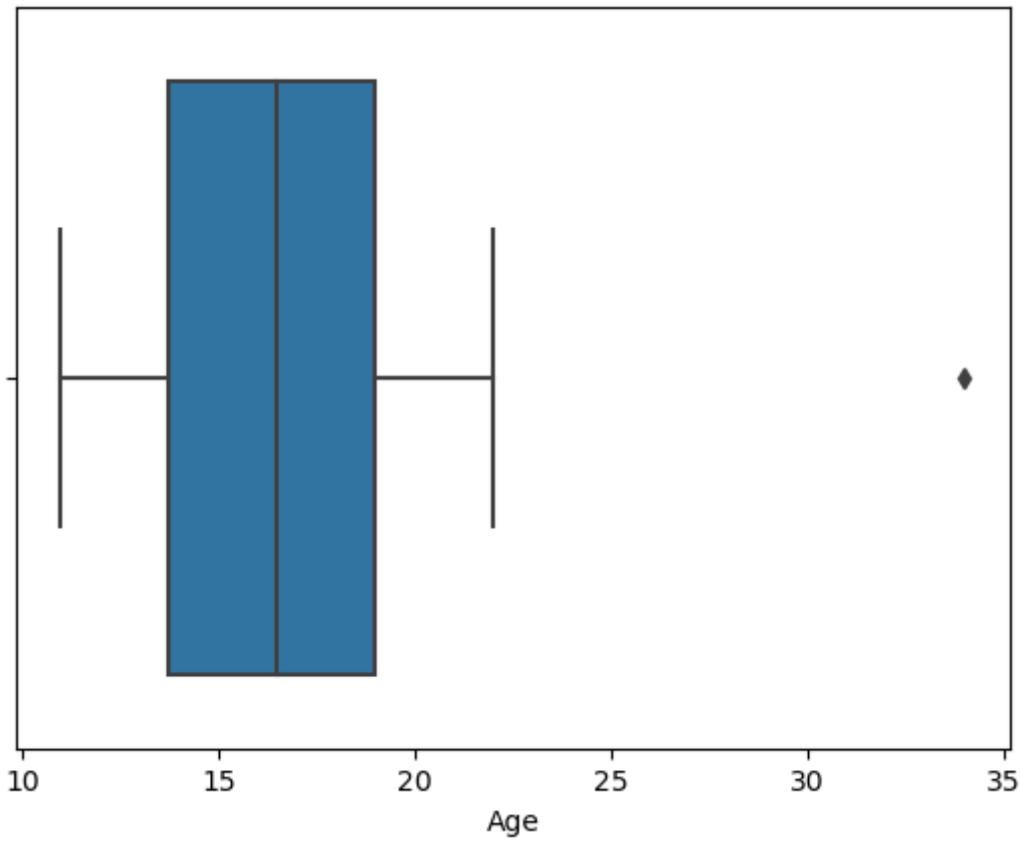
In [8]:

```
sns.boxplot(df['Age'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```

Out[8]:

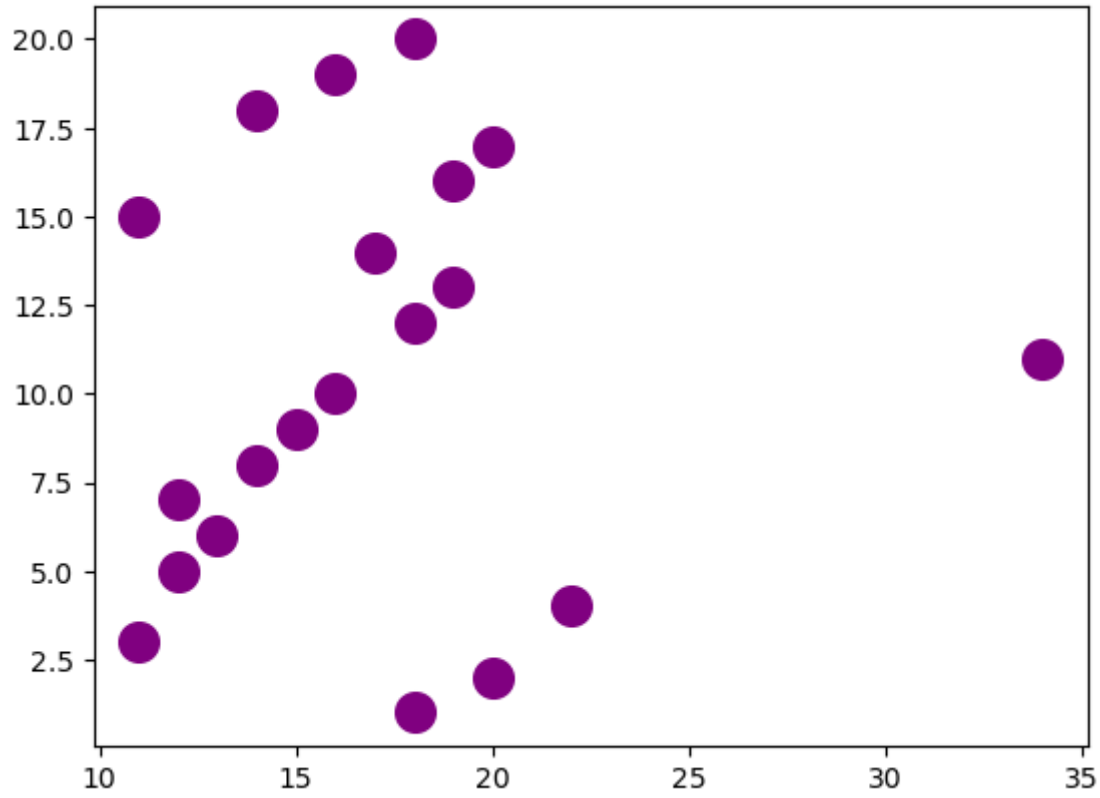
```
<AxesSubplot:xlabel='Age'>
```



Scatter Plot Method

In [9]:

```
x=df.Age
y=df.RollNo
plt.scatter(x,y,c="Purple",s=200)
plt.show()
```



Z-score Method

In [10]:

```
data=df.Age
m=np.mean(data)
sd=np.std(data)
print("Mean is :",m)
print("Standard Deviation is:",sd)
```

```
Mean is : 16.95
Standard Deviation is: 5.0246890451051796
```

In [11]:

```
'''
68% of the data points lie between +/- 1 standard deviation
95% of the data points lie between +/- 2 standard deviation
99.7% of the data points lie between +/- 3 standard deviation
'''
threshold = 3
outlier = []
for i in data:
    z = (i-m)/sd
    if z > threshold:
        outlier.append(i)
print('Outlier in dataset is', outlier)
```

```
Outlier in dataset is [34]
```

IQR (Inter Quartile Range)

In [12]:

```
data=np.sort(df.Age)
Q1 = np.percentile(data, 25, interpolation = 'midpoint')
Q2 = np.percentile(data, 50, interpolation = 'midpoint')
Q3 = np.percentile(data, 75, interpolation = 'midpoint')

print('Q1 25 percentile of the given data is, ', Q1)
print('Q1 50 percentile of the given data is, ', Q2)
print('Q1 75 percentile of the given data is, ', Q3)

IQR = Q3 - Q1
print('Interquartile range is', IQR)
```

```
Q1 25 percentile of the given data is,  13.5
Q1 50 percentile of the given data is,  16.5
Q1 75 percentile of the given data is,  19.0
Interquartile range is 5.5
```

In [13]:

```
low_lim = Q1 - 1.5 * IQR
up_lim = Q3 + 1.5 * IQR
print('low_limit is', low_lim)
print('up_limit is', up_lim)
outlier =[]
for x in data:
    if ((x> up_lim) or (x<low_lim)):
         outlier.append(x)
print('Outlier in the dataset is', outlier)
```

```
low_limit is 5.25
up_limit is 27.25
Outlier in the dataset is [34]
```

Removing the Outliers

In [14]:

```
df.drop(10, inplace = True)
df
```

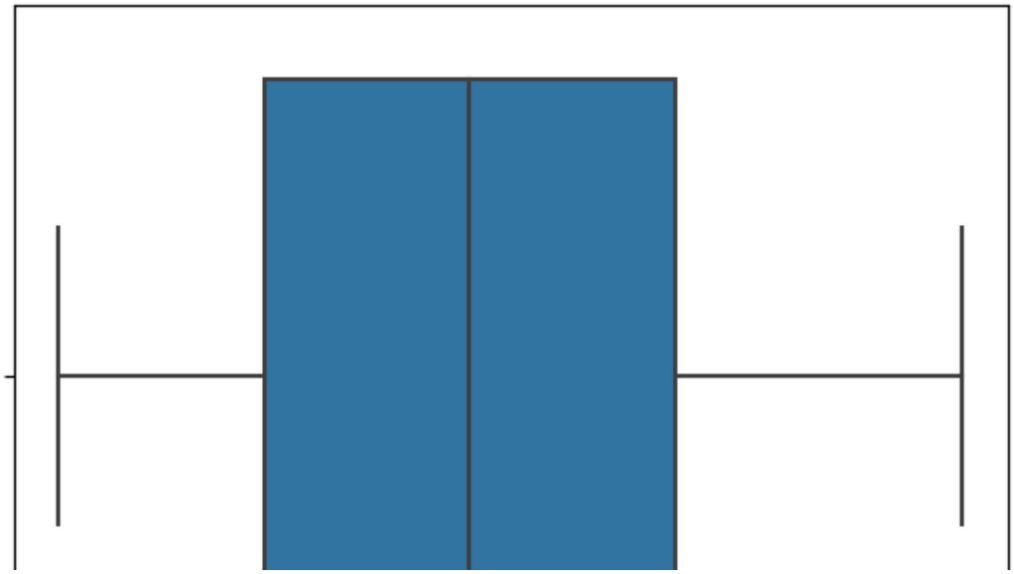
Out[14]:

	RollNo	Name	Age	Maths	Science	English	Total	Average
0	1	Anish	18	25.000000	35	56	116.000000	56.523529
1	2	Aasawari	20	33.000000	78	34	145.000000	48.300000
2	3	Aditya	11	90.000000	33	22	145.000000	48.300000
3	4	Apoorva	22	45.000000	88	56	189.000000	63.000000
4	5	Anil	12	66.000000	43	88	197.000000	65.700000
5	6	Bablu	13	54.722222	35	53	110.000000	56.523529
6	7	Chetan	12	56.000000	66	78	162.666667	66.700000
7	8	Chinmay	14	56.000000	36	68	160.000000	53.300000
8	9	Chitra	15	67.000000	22	32	121.000000	40.300000
9	10	Diya	16	54.000000	78	88	220.000000	73.300000
11	12	Dilip	18	87.000000	89	37	213.000000	56.523529
12	13	Esha	19	89.000000	53	40	182.000000	60.700000
13	14	Fatima	17	34.000000	45	78	157.000000	52.300000
14	15	Faiz	11	32.000000	51	27	110.000000	36.700000
15	16	Gaurav	19	80.000000	30	28	138.000000	46.000000
16	17	Hitesh	20	54.722222	74	30	162.666667	64.700000
17	18	Isha	14	50.000000	90	56	196.000000	65.300000
18	19	Ishan	16	40.000000	42	89	171.000000	57.000000
19	20	Jay	18	48.000000	72	90	210.000000	70.000000

In [15]:

```
sns.boxplot(df['Age'])
```

```
<AxesSubplot:xlabel='Age'>
```



Log Transform

In [21]:

```
df['log_total'] = np.log2(df['Total'])
df
```

Out[21]:

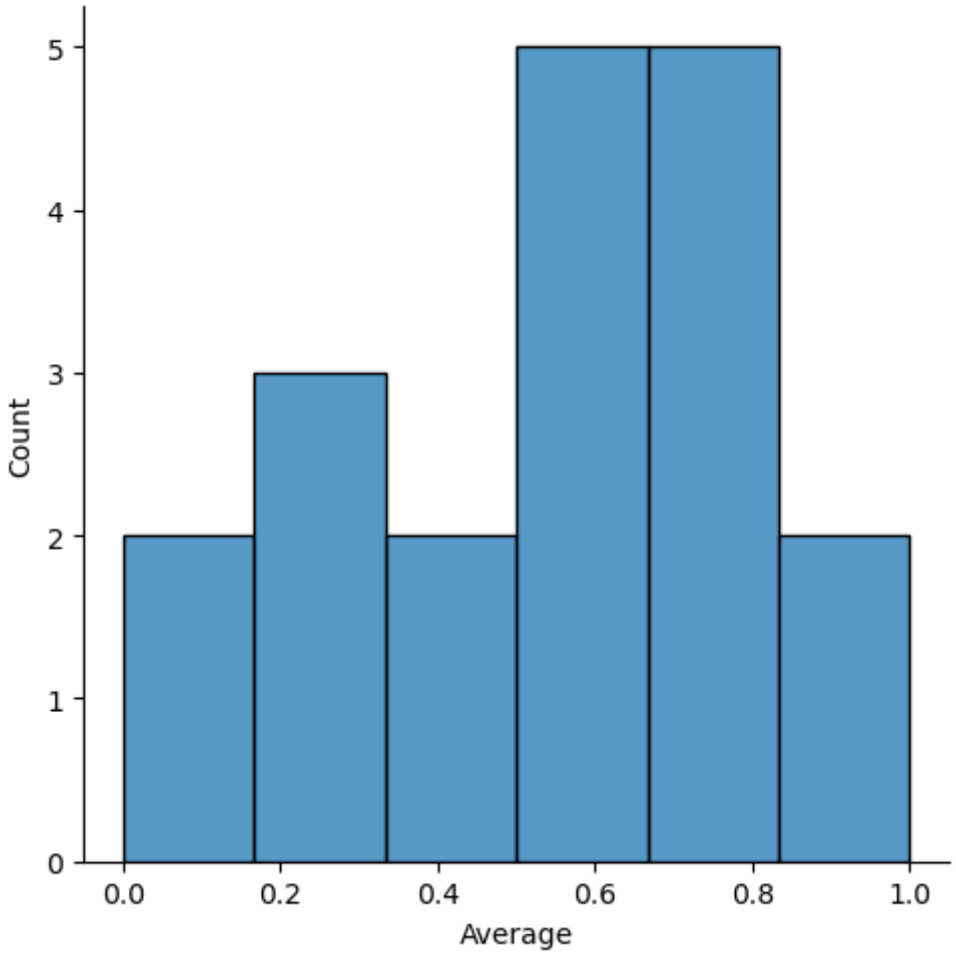
	RollNo	Name	Age	Maths	Science	English	Total	Average	log_total
0	1	Anish	18	25.000000	35	56	116.000000	56.523529	6.857981
1	2	Aasawari	20	33.000000	78	34	145.000000	48.300000	7.179909
2	3	Aditya	11	90.000000	33	22	145.000000	48.300000	7.179909
3	4	Apoorva	22	45.000000	88	56	189.000000	63.000000	7.562242
4	5	Anil	12	66.000000	43	88	197.000000	65.700000	7.622052
5	6	Bablu	13	54.722222	35	53	110.000000	56.523529	6.781360
6	7	Chetan	12	56.000000	66	78	162.666667	66.700000	7.345775
7	8	Chinmay	14	56.000000	36	68	160.000000	53.300000	7.321928
8	9	Chitra	15	67.000000	22	32	121.000000	40.300000	6.918863
9	10	Diya	16	54.000000	78	88	220.000000	73.300000	7.781360
11	12	Dilip	18	87.000000	89	37	213.000000	56.523529	7.734710
12	13	Esha	19	89.000000	53	40	182.000000	60.700000	7.507795
13	14	Fatima	17	34.000000	45	78	157.000000	52.300000	7.294621
14	15	Faiz	11	32.000000	51	27	110.000000	36.700000	6.781360
15	16	Gaurav	19	80.000000	30	28	138.000000	46.000000	7.108524
16	17	Hitesh	20	54.722222	74	30	162.666667	64.700000	7.345775
17	18	Isha	14	50.000000	90	56	196.000000	65.300000	7.614710
18	19	Ishan	16	40.000000	42	89	171.000000	57.000000	7.417853
19	20	Jay	18	48.000000	72	90	210.000000	70.000000	7.714246

Normal Distribution

Min-max Scaling

In [17]:

```
df_copy=df.copy()
df_copy['Average']=(df_copy['Average'] - df_copy['Average'].min())/(df_copy['Average'].m
sns.displot(df_copy['Average'])
plt.show()
```



Z-index Scaling

In [19]:

```
z_copy=df.copy()
for col in ['Average']:
    z_copy[col]=(z_copy[col]-z_copy[col].mean())/z_copy[col].std()
```

In [20]:

```
sns.displot(z_copy['Average'])
plt.show()
```