AI4SEE PRIVATE LIMITED Assignment

-- by Saptaswa Basu

In [1]:
```
import numpy as np
```

In [2]:
```
import pandas as pd
```

In [3]:
```
import statistics as stats
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib notebook
```

In [4]:
```
df = pd.read_csv('task_dataset.csv')
```

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

Out[5]:

	AF3	F7	F3	FC5	T7	P	O1	O2	P8	T8	FC6	F4	F8	AF4	class
0	4329.23	4009.23	4289.23	4148.21	4350.26	4586.15	4096.92	4641.03	4222.05	4238.46	4211.28	4280.51	4635.90	4393.85	0
1	4324.62	4004.62	4293.85	4148.72	4342.05	4586.67	4097.44	4638.97	4210.77	4226.67	4207.69	4279.49	4632.82	4384.10	0
2	4327.69	4006.67	4295.38	4156.41	4336.92	4583.59	4096.92	4630.26	4207.69	4222.05	4206.67	4282.05	4628.72	4389.23	0
3	4328.72	4011.79	4296.41	4155.90	4343.59	4582.56	4097.44	4630.77	4217.44	4235.38	4210.77	4287.69	4632.31	4396.41	0
4	4326.15	4011.79	4292.31	4151.28	4347.69	4586.67	4095.90	4627.69	4210.77	4244.10	4212.82	4288.21	4632.82	4398.46	0

In [8]:
```
df.describe()
```

Out[8]:

	AF3	F7	F3	FC5	T7	P	O1	O2	P8	
count	14980.000000	14980.000000	14980.000000	14980.000000	14980.000000	14980.000000	14980.000000	14980.000000	14980.000000	149
mean	4321.917777	4009.767694	4264.022433	4164.946326	4341.741075	4644.022379	4110.400160	4616.056904	4218.826610	42
std	2492.072174	45.941672	44.428052	5216.404632	34.738821	2924.789537	4600.926543	29.292603	2136.408523	
min	1030.770000	2830.770000	1040.000000	2453.330000	2089.740000	2768.210000	2086.150000	4567.180000	1357.950000	18
25%	4280.510000	3990.770000	4250.260000	4108.210000	4331.790000	4611.790000	4057.950000	4604.620000	4190.770000	42
50%	4294.360000	4005.640000	4262.560000	4120.510000	4338.970000	4617.950000	4070.260000	4613.330000	4199.490000	42
75%	4311.790000	4023.080000	4270.770000	4132.310000	4347.180000	4626.670000	4083.590000	4624.100000	4209.230000	42
max	309231.000000	7804.620000	6880.510000	642564.000000	6474.360000	362564.000000	567179.000000	7264.100000	265641.000000	66

In [9]:
```
mean = df.mean()
mean
```

Out[9]:
```
AF3      4321.917777
F7       4009.767694
F3       4264.022433
FC5      4164.946326
T7       4341.741075
P        4644.022379
O1       4110.400160
O2       4616.056904
P8       4218.826610
T8       4231.316200
FC6      4202.456900
F4       4279.232774
F8       4615.205336
AF4      4416.435832
class       0.448798
dtype: float64
```

In [10]:
```
median = df.median()
median
```

Out[10]:
```
AF3       4294.36
F7        4005.64
F3        4262.56
FC5       4120.51
T7        4338.97
P         4617.95
O1        4070.26
O2        4613.33
P8        4199.49
T8        4229.23
FC6       4200.51
F4        4276.92
F8        4603.08
AF4       4354.87
class        0.00
dtype: float64
```

In [14]:
```
df.mode()
```

Out[14]:

	AF3	F7	F3	FC5	T7	P	O1	O2	P8	T8	FC6	F4	F8	AF4	class
0	4291.79	4003.59	4263.59	4122.56	4332.31	4616.41	4072.31	4610.77	4196.92	4224.62	4195.38	4273.85	4603.08	4352.31	0

In [11]:
```
std = df.std()
std
```

Out[11]:
```
AF3       2492.072174
F7          45.941672
F3          44.428052
FC5       5216.404632
T7          34.738821
P         2924.789537
O1        4600.926543
O2          29.292603
P8        2136.408523
T8          38.050903
FC6         37.785981
F4          41.544312
F8        1208.369958
AF4       5891.285043
class        0.497388
dtype: float64
```

In [15]:
```
import scipy
```

In [16]:
```
from scipy.stats import skew
```

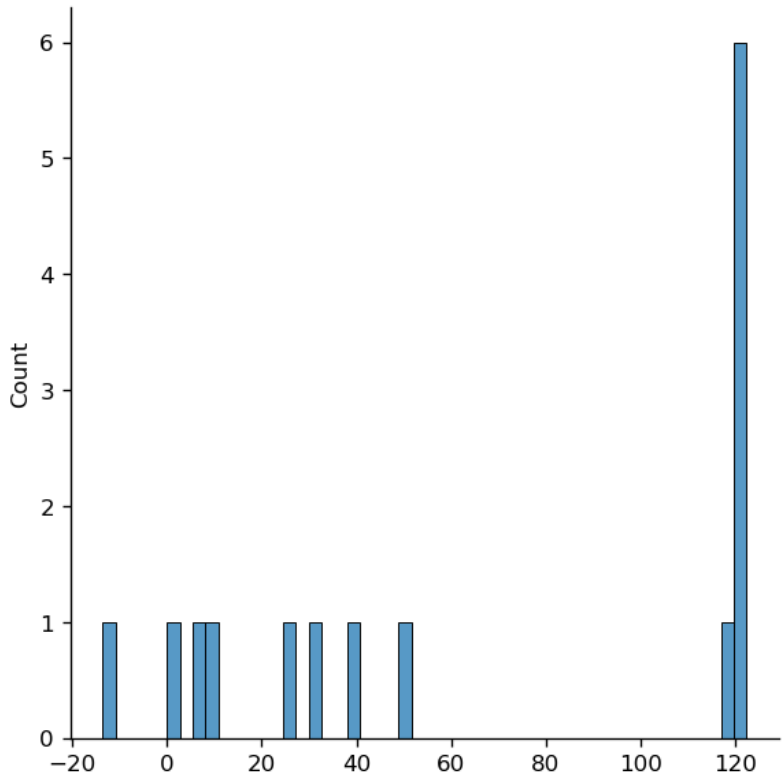
In [17]:
```
for col in df:
    print(col)
    print(skew(df[col]))
```

```
AF3
122.28161919534546
F7
39.04264771302619
F3
-13.613797368722386
FC5
122.37552142562957
T7
7.5611449017448935
P
122.3505575909479
O1
122.37133778860104
O2
51.09210233140153
P8
122.32242105912867
T8
10.229676557495747
FC6
31.645835610211982
F4
26.553809584445453
F8
121.89506508305209
AF4
118.11321639532063
class
0.20588877122691596
```

In [34]:
```
skew(df)
```

Out[34]:
```
array([122.2816192 ,  39.04264771, -13.61379737, 122.37552143,
         7.5611449 , 122.35055759, 122.37133779,  51.09210233,
       122.32242106,  10.22967656,  31.64583561,  26.55380958,
       121.89506508, 118.1132164 ,   0.20588877])
```

In [36]:
```
sns.displot(skew(df), kde = False, bins = 50)
```

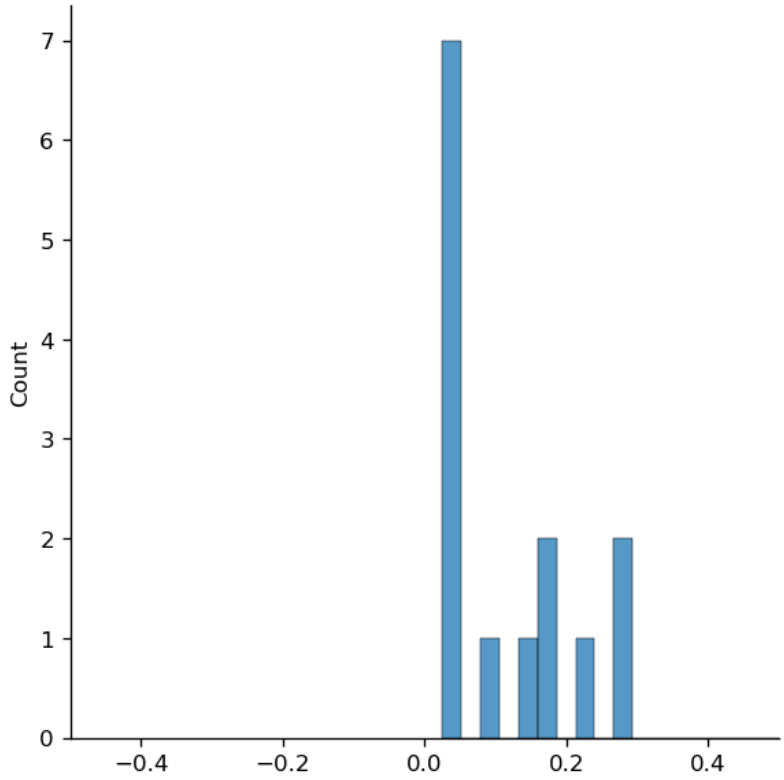


Out[36]:
```
<seaborn.axisgrid.FacetGrid at 0x1d3bd6a3fd0>
```

In [12]:
```
skewness = 3*(mean - median)/std
skewness
```

Out[12]:
```
AF3      0.033175
F7       0.269539
F3       0.098751
FC5      0.025556
T7       0.239307
P        0.026743
O1       0.026173
O2       0.279276
P8       0.027153
T8       0.164480
FC6      0.154573
F4       0.167010
F8       0.030103
AF4      0.031351
class    2.706931
dtype: float64
```

In [31]:
```
sns.displot(skewness, kde = False, bins = 100)
```



Out[31]:
```
(-0.5, 0.5)
```

As we can see the skewness of each column, we can say:

- Each and every columns has positive skewness
- And among them `F7, T7, O2` are highly skewed,
- Whereas `AF3, FC5, P, O1, P8, F8 and AF4` columns have very minimum skewness that we can ignore this as its very close to 0.
- Also we can see here that, `class` column has extrimely high skewness , it propably beacause it is the target column of our dataset.

Now to solve this, which columns have high skewness we can use square root on those columns to reduce the skewness. But before that we also need to check the correlation between them, we will only apply square root to those who have less correlaton and high skewness.

In [ ]:
```
```