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

In [3]:
```
data = pd.read_csv("C:\\Users\\GOVIND SINGH\\Downloads\\01.Data Cleaning and Preprc
```

In [5]:
```
data
```

Out[5]:

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.545	1.443
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.709	NaN
...	...	...	...	...	...	...	...	...	...
319	10-16:00	23.75	12.667	93.450	1178.252	276.955	347.286	310.970	1.523
320	9-19:00	19.80	12.558	94.352	1184.119	297.071	399.135	319.576	1.451
321	9-20:00	23.01	12.550	90.842	1188.517	289.826	373.633	314.591	1.457
322	9-21:00	24.32	13.083	88.910	1192.879	318.006	364.081	308.559	1.523
323	9-22:00	25.75	13.417	85.451	1186.342	248.312	356.289	310.482	1.474

324 rows × 23 columns

In [6]:
```
#Now we will deal with the missing values in our data
#.info() gives us both data-types and the sum of null values
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 324 entries, 0 to 323
Data columns (total 23 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Observation      324 non-null    object 
 1   Y-Kappa          324 non-null    float64
 2   ChipRate         319 non-null    float64
 3   BF-CMratio       307 non-null    float64
 4   BlowFlow         308 non-null    float64
 5   ChipLevel4       323 non-null    float64
 6   T-upperExt-2     322 non-null    float64
 7   T-lowerExt-2     322 non-null    float64
 8   UCZAA            299 non-null    float64
 9   WhiteFlow-4      323 non-null    float64
 10  AAWhiteSt-4      173 non-null    float64
 11  AA-Wood-4        323 non-null    float64
 12  ChipMoisture-4   323 non-null    float64
 13  SteamFlow-4      323 non-null    float64
 14  Lower-HeatT-3    322 non-null    float64
 15  Upper-HeatT-3    322 non-null    float64
 16  ChipMass-4       323 non-null    float64
 17  WeakLiquorF      323 non-null    float64
 18  BlackFlow-2      322 non-null    float64
 19  WeakWashF        323 non-null    float64
 20  SteamHeatF-3     322 non-null    float64
 21  T-Top-Chips-4    323 non-null    float64
 22  SulphidityL-4    173 non-null    float64
dtypes: float64(22), object(1)
memory usage: 58.3+ KB
```

In [7]:
```
#It will give us the sum of null values in corresponding column of our data
data.isnull().sum()
```

Out[7]:
```
Observation          0
Y-Kappa              0
ChipRate             5
BF-CMratio          17
BlowFlow            16
ChipLevel4           1
T-upperExt-2         2
T-lowerExt-2         2
UCZAA               25
WhiteFlow-4          1
AAWhiteSt-4        151
AA-Wood-4            1
ChipMoisture-4       1
SteamFlow-4          1
Lower-HeatT-3        2
Upper-HeatT-3        2
ChipMass-4           1
WeakLiquorF          1
BlackFlow-2          2
WeakWashF            1
SteamHeatF-3         2
T-Top-Chips-4        1
SulphidityL-4      151
dtype: int64
```

In [8]:
```
#It will give us the total number of null values present in our data
data.isnull().sum().sum()
```

Out[8]:
```
386
```

In [10]:
```
#.describe() is very useful if we are trying to get a simple statistical report for

data.describe()
```

Out[10]:

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	U
count	324.000000	319.000000	307.000000	308.000000	323.000000	322.000000	322.000000	299.0
mean	20.635370	14.347937	87.464456	1237.837614	258.164483	356.904295	324.020180	1.49
std	3.070036	1.499095	7.995012	100.593735	87.987452	9.209290	7.621402	0.1(
min	12.170000	9.983000	68.645000	0.000000	0.000000	339.168000	284.633000	1.18
25%	18.382500	13.358000	81.823000	1193.215250	213.527000	350.241250	321.420000	1.43
50%	20.845000	14.308000	86.739000	1273.138500	271.792000	356.843000	325.669000	1.49
75%	23.032500	15.517000	92.372000	1289.196000	321.680000	362.242250	329.175000	1.5(
max	27.600000	16.958000	121.717000	1351.240000	419.014000	399.135000	337.012000	1.74

8 rows × 22 columns

In [13]:
```
#Droping the duplicate values from our data
data = data.drop_duplicates()
data
```

Out[13]:

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.545	1.443
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.709	NaN
...	...	...	...	...	...	...	...	...	...
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.041	1.635
299	12-10:00	24.98	NaN	85.034	1278.345	368.564	357.723	321.387	NaN
300	12-11:00	21.00	NaN	88.013	1307.722	278.842	357.438	323.757	NaN
301	12-12:00	21.40	NaN	85.490	1255.986	273.484	361.365	322.689	NaN
307	31-05:00	20.89	14.308	94.172	1327.832	251.120	351.263	332.485	1.522

301 rows × 23 columns

In [14]:
```
#replacing the null values with 0
data2 = data.fillna(value=0)
data2
```

Out[14]:

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.545	1.443
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.709	0.000
...	...	...	...	...	...	...	...	...	...
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.041	1.635
299	12-10:00	24.98	0.000	85.034	1278.345	368.564	357.723	321.387	0.000
300	12-11:00	21.00	0.000	88.013	1307.722	278.842	357.438	323.757	0.000
301	12-12:00	21.40	0.000	85.490	1255.986	273.484	361.365	322.689	0.000
307	31-05:00	20.89	14.308	94.172	1327.832	251.120	351.263	332.485	1.522

301 rows × 23 columns

In [15]:
```
#We can use dropna() to remove all the rows with missing data.
data3 = data.dropna()
data3
```

Out[15]:

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604
5	1-08:00	14.23	15.350	85.518	1171.604	198.538	344.014	325.195	1.436
7	31-06:00	22.65	14.100	91.887	1307.852	288.989	352.321	331.162	1.468
9	31-08:00	24.70	13.850	96.208	1334.892	362.511	352.372	327.358	1.515
...	...	...	...	...	...	...	...	...	...
290	12-01:00	19.90	11.333	87.405	1033.565	369.383	343.515	302.364	1.592
292	12-03:00	22.00	11.858	93.199	1171.206	366.787	345.261	310.115	1.513
294	12-05:00	19.00	12.425	92.905	1272.030	316.226	345.811	307.806	1.633
296	12-07:00	20.50	13.358	97.662	1304.597	377.678	347.672	313.147	1.546
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.041	1.635

131 rows × 23 columns

In [16]:
```
data2.isnull().sum().sum()
```

Out[16]:
```
0
```

In [18]:
```
#filling null values with the next value
data4=data.fillna(method='bfill')
data4
```

```
C:\Users\GOVIND SINGH\AppData\Local\Temp\ipykernel_20372\1594516589.py:1: FutureWa
rning: DataFrame.fillna with 'method' is deprecated and will raise in a future ver
sion. Use obj.ffill() or obj.bfill() instead.
  data4=data.fillna(method='bfill')
```

Out[18]:

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.545	1.443
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.709	1.436
...	...	...	...	...	...	...	...	...	...
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.041	1.635
299	12-10:00	24.98	14.308	85.034	1278.345	368.564	357.723	321.387	1.522
300	12-11:00	21.00	14.308	88.013	1307.722	278.842	357.438	323.757	1.522
301	12-12:00	21.40	14.308	85.490	1255.986	273.484	361.365	322.689	1.522
307	31-05:00	20.89	14.308	94.172	1327.832	251.120	351.263	332.485	1.522

301 rows × 23 columns

In [19]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
```

In [ ]:
```
#detect the outliers using IQR
```

In [20]:
```
data2.columns
```

Out[20]:
```
Index(['Observation', 'Y-Kappa', 'ChipRate', 'BF-CMratio', 'BlowFlow',
       'ChipLevel4 ', 'T-upperExt-2 ', 'T-lowerExt-2  ', 'UCZAA',
       'WhiteFlow-4 ', 'AAWhiteSt-4 ', 'AA-Wood-4  ', 'ChipMoisture-4 ',
       'SteamFlow-4 ', 'Lower-HeatT-3', 'Upper-HeatT-3 ', 'ChipMass-4 ',
       'WeakLiquorF ', 'BlackFlow-2 ', 'WeakWashF ', 'SteamHeatF-3 ',
       'T-Top-Chips-4 ', 'SulphidityL-4 '],
      dtype='object')
```

In [21]:
```
data2.drop(['Observation'], axis=1, inplace=True)
```

In [22]:
```
data2.columns
```

Out[22]:
```
Index(['Y-Kappa', 'ChipRate', 'BF-CMratio', 'BlowFlow', 'ChipLevel4 ',
       'T-upperExt-2 ', 'T-lowerExt-2  ', 'UCZAA', 'WhiteFlow-4 ',
       'AAWhiteSt-4 ', 'AA-Wood-4  ', 'ChipMoisture-4 ', 'SteamFlow-4 ',
       'Lower-HeatT-3', 'Upper-HeatT-3 ', 'ChipMass-4 ', 'WeakLiquorF ',
       'BlackFlow-2 ', 'WeakWashF ', 'SteamHeatF-3 ', 'T-Top-Chips-4 ',
       'SulphidityL-4 '],
      dtype='object')
```

In [23]:
```
Q1= data2.quantile(0.25)
Q3= data2.quantile(0.75)
IQR=Q3-Q1
print(IQR)
```

```
Y-Kappa              4.550
ChipRate             2.233
BF-CMratio          10.912
BlowFlow            96.766
ChipLevel4         105.868
T-upperExt-2        11.994
T-lowerExt-2         7.609
UCZAA                0.152
WhiteFlow-4        100.098
AAWhiteSt-4          6.143
AA-Wood-4            1.486
ChipMoisture-4       2.186
SteamFlow-4          8.840
Lower-HeatT-3        8.585
Upper-HeatT-3        7.852
ChipMass-4          19.347
WeakLiquorF        180.613
BlackFlow-2        280.829
WeakWashF          267.219
SteamHeatF-3         6.903
T-Top-Chips-4        2.044
SulphidityL-4       30.420
dtype: float64
```

In [24]:
```
data2=data2[~((data2<(Q1-1.5*IQR))|(data2>(Q3+1.5*IQR))).any(axis=1)]
data2
```

Out[24]:

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA	WhiteFlow-4
1	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549	537.201
2	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600	549.611
3	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604	623.362
5	14.23	15.350	85.518	1171.604	198.538	344.014	325.195	1.436	628.245
6	13.49	13.700	98.186	1243.688	116.275	346.208	326.982	1.434	696.766
...	...	...	...	...	...	...	...	...	...
276	22.70	15.517	83.008	1288.010	306.886	350.155	322.485	1.590	568.752
296	20.50	13.358	97.662	1304.597	377.678	347.672	313.147	1.546	496.460
297	20.40	14.233	89.790	1278.006	379.458	354.290	315.558	1.515	491.374
298	20.90	15.167	84.640	1283.706	339.440	354.803	311.041	1.635	532.419
307	20.89	14.308	94.172	1327.832	251.120	351.263	332.485	1.522	631.514

226 rows × 22 columns

In [26]:
```
#tranforming dataset
import scipy
import sklearn
```

```
from sklearn import preprocessing
from sklearn.preprocessing import scale
```

In [27]:
```
data2.describe()
```

Out[27]:

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	U
count	226.000000	226.000000	226.000000	226.000000	226.000000	226.000000	226.000000	226.0
mean	20.690487	14.673491	85.882181	1255.288916	264.664912	356.861681	325.341124	1.4
std	2.982916	1.297369	7.033155	47.896055	74.345135	7.466897	5.557537	0.1
min	12.480000	10.833000	68.645000	1084.083000	61.783000	340.222000	310.421000	1.1
25%	18.457500	13.850000	80.984000	1221.926000	220.356000	350.704250	322.355500	1.4
50%	20.775000	14.729000	84.967000	1280.291500	270.965000	357.560500	326.508500	1.4
75%	23.010000	15.708000	91.178750	1289.254000	322.492000	361.555000	329.264500	1.5
max	27.600000	16.958000	108.104000	1351.240000	419.014000	375.047000	337.012000	1.7

8 rows × 22 columns

In [28]:
```
data2.matrix=data2.values.reshape(-1,1)
scaled=preprocessing.MinMaxScaler(feature_range=(0,10))
scaled_data=scaled.fit_transform(data2)
data2
```

```
C:\Users\GOVIND SINGH\AppData\Local\Temp\ipykernel_20372\2476949266.py:1: UserWarn
ing: Pandas doesn't allow columns to be created via a new attribute name - see htt
ps://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  data2.matrix=data2.values.reshape(-1,1)
```

Out[28]:

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA	WhiteFlow-4
1	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549	537.201
2	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600	549.611
3	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604	623.362
5	14.23	15.350	85.518	1171.604	198.538	344.014	325.195	1.436	628.245
6	13.49	13.700	98.186	1243.688	116.275	346.208	326.982	1.434	696.766
...	...	...	...	...	...	...	...	...	...
276	22.70	15.517	83.008	1288.010	306.886	350.155	322.485	1.590	568.752
296	20.50	13.358	97.662	1304.597	377.678	347.672	313.147	1.546	496.460
297	20.40	14.233	89.790	1278.006	379.458	354.290	315.558	1.515	491.374
298	20.90	15.167	84.640	1283.706	339.440	354.803	311.041	1.635	532.419
307	20.89	14.308	94.172	1327.832	251.120	351.263	332.485	1.522	631.514

226 rows × 22 columns