```
import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd from sklearn.preprocessing

import power_transform

#Importing data and storing it in variable called 'data'
data = pd.read_excel('D:\Gulzar\Miscll\Personal\Project\AIML\ML\
Project\Project-Anamoly_Detection\Data_Set\AnomaData.xlsx')
data
```

```
<>:2: SyntaxWarning: invalid escape sequence '\A'
<>:2: SyntaxWarning: invalid escape sequence '\A'
C:\Users\Gulzar.alam\AppData\Local\Temp\
ipykernel_16240\3623776104.py:2: SyntaxWarning: invalid escape
sequence '\A'
   data = pd.read_excel('D:\Gulzar.alam\Gulzar\Miscll\Personal\
Project\AIML\ML\Project\Project-Anamoly_Detection\Data_Set\
AnomaData.xlsx')
```

```
                    time y        x1        x2        x3         x4
\
0       1999-05-01 00:00:00 0  0.376665 -4.596435 -4.095756  13.497687
1       1999-05-01 00:02:00 0  0.475720 -4.542502 -4.018359  16.230659
2       1999-05-01 00:04:00 0  0.363848 -4.681394 -4.353147  14.127997
3       1999-05-01 00:06:00 0  0.301590 -4.758934 -4.023612  13.161566
4       1999-05-01 00:08:00 0  0.265578 -4.749928 -4.333150  15.267340
...                     ... ..       ...       ...       ...        ...
18393 1999-05-28 23:58:00 0 -0.877441  0.786430  0.406426 135.301215
18394 1999-05-29 00:00:00 0 -0.843988  0.633086  0.561918 133.228949
18395 1999-05-29 00:02:00 0 -0.826547  0.450126  0.334582 134.977973
18396 1999-05-29 00:04:00 0 -0.822843  0.419383  0.387263 135.658942
18397 1999-05-29 00:06:00 0 -0.840981  0.582710  0.593416 136.339880


            x5         x6        x7        x8 ...        x51
x52   \
0     -0.118830 -20.669883 0.000732 -0.061114 ... ...  29.984624
10.091721
1     -0.128733 -18.758079 0.000732 -0.061114        29.984624
```

10.095871
2 -0.138636 -17.836632 0.010803 -0.061114 ... ... 29.984624
10.100265 ... ... 29.984624
3 -0.148142 -18.517601 0.002075 -0.061114 ... ... 29.984624
10.104660 ... ...
4 -0.155314 -17.505913 0.000732 -0.061114 ... 29.984624
10.109054 29.984624
... ... 18393 ... ... 29.984624
0.112295 26.300392 -0.159185 0.058823 29.984624
0.058823 29.984624 -
0.773514 0.048752 -
18394 0.141332 25.678597 -0.159185 0.048752 -
0.773514 0.048752 -
18395 0.170370 25.056801 -0.159185 -
0.773514
18396 0.199422 24.435005 -0.159185
0.773514
18397 0.228460 24.712960 -0.159185
0.773514

x54 x55 x56 x57 x58 x59
x60 \
0 -4.936434 -24.590146 18.515436 3.473400 0.033444 0.953219
0.006076
2.682933 0.033536 1.090502
1 -4.937179 -32.413266 22.760065
0.006083
3.537487 0.033629 1.840540
2 -4.937924 -34.183774 27.004663
3.986095 0.033721 2.554880
0.006090
3 -4.938669 -35.954281 21.672449
3.601573 0.033777 1.410494
0.006097
4 -4.939414 -37.724789 21.907251
... ...
0.006105
6.944644 -37.795661 -0.860218 0.010220
... ...
0.507755 -39.357199 -0.915698 0.010620
...
18393 2.682413
2.809146 -39.357199 -1.409596 0.013323 0.895685 -
0.011242
2.164859 -39.357199 -0.860218 0.012888 0.175348 -
18394 2.683338
0.011235
1.416690 -39.357199 -0.732044 0.012453 0.621020 -
18395 2.684263
0.011228
1.390902 -
18396 2.685189
0.418993 -
0.011221
18397 2.686114
0.011214

	y.1
0	0
1	0
2	0

```
3   4   ...      0
18393            0
18394            ...
18395            0
18396            0
18397            0
                 0
                0

[18398 rows x 62 columns]

# Converting the excel data into the dataframe
# Why ?
	#1. DataFrames are designed for easy handling of structured data
(rows and columns), Handling missing data, Sorting data etc.
	#2. Pandas provides a wide range of built-in functions that are
optimized for DataFrame objects
df = pd.DataFrame(data)
df

                     time y        x1        x2        x3          x4
\
0     1999-05-01 00:00:00 0  0.376665 -4.596435 -4.095756   13.497687

1     1999-05-01 00:02:00 0  0.475720 -4.542502 -4.018359   16.230659

2     1999-05-01 00:04:00 0  0.363848 -4.681394 -4.353147   14.127997

3     1999-05-01 00:06:00 0  0.301590 -4.758934 -4.023612   13.161566

4     1999-05-01 00:08:00 0  0.265578 -4.749928 -4.333150   15.267340

...                   ... ..      ...       ...       ...         ...

18393 1999-05-28 23:58:00 0 -0.877441  0.786430  0.406426  135.301215

18394 1999-05-29 00:00:00 0 -0.843988  0.633086  0.561918  133.228949

18395 1999-05-29 00:02:00 0 -0.826547  0.450126  0.334582  134.977973

18396 1999-05-29 00:04:00 0 -0.822843  0.419383  0.387263  135.658942

18397 1999-05-29 00:06:00 0 -0.840981  0.582710  0.593416  136.339880


             x5         x6        x7        x8 ...        x51
x52    \
0     -0.118830 -20.669883 0.000732 -0.061114 ... ...  29.984624
10.091721
1     -0.128733 -18.758079 0.000732 -0.061114        29.984624
10.095871
```

```
2          -0.138636 -17.836632 0.010803 -0.061114        ...  ...   29.984624
10.100265                                                 ...  ...   29.984624
3          -0.148142 -18.517601 0.002075 -0.061114        ...  ...   29.984624
10.104660                                                 ...  ...          ...
4          -0.155314 -17.505913 0.000732 -0.061114        ...        29.984624
10.109054                                                            29.984624
... ... 18393      ...          ...            ...        ...        29.984624
                 0.112295 26.300392 -0.159185   0.058823             29.984624
                                                0.058823            29.984624  -
0.773514                                        0.048752                       -
18394   0.141332 25.678597 -0.159185            0.048752                       -
0.773514                                        0.048752                       -
18395   0.170370 25.056801 -0.159185                                           -
0.773514
18396   0.199422 24.435005 -0.159185
0.773514
18397   0.228460 24.712960 -0.159185
0.773514

                 x54            x55            x56          x57         x58          x59
x60   \
0           -4.936434 -24.590146 18.515436   3.473400  0.033444    0.953219
0.006076
                                             2.682933   0.033536    1.090502
1          -4.937179 -32.413266 22.760065
0.006083                                     3.537487  0.033629   1.840540
2          -4.937924 -34.183774 27.004663
                                            3.986095   0.033721   2.554880
0.006090
3          -4.938669 -35.954281 21.672449    3.601573  0.033777   1.410494
0.006097
                                                  ...          ...          ...
4          -4.939414 -37.724789 21.907251
0.006105              6.944644 -37.795661 -0.860218   0.010220
...                ...
...                   0.507755 -39.357199 -0.915698   0.010620
         2.682413
18393                 2.809146 -39.357199 -1.409596   0.013323  0.895685   -
0.011242
                      2.164859 -39.357199 -0.860218   0.012888  0.175348   -
18394   2.683338
0.011235              1.416690 -39.357199 -0.732044   0.012453  0.621020   -
18395   2.684263
                                                                1.390902   -
0.011228
18396   2.685189                                                0.418993 -
0.011221
18397   2.686114
0.011214

           y.1
0            0
1            0
2            0
3            0
```

```
4 0 ... ... 18393 0 18394 0 18395 0 18396 0
18397 0

[18398 rows x 62 columns]
```

Exploratory Data Analysis

```
#Analyzing Rows and Columns df.shape
```

```
(18398, 62)
```

```
#Information about non-null values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18398 entries, 0 to 18397
Data columns (total 62 columns):

 #   Column Non-Null Count   Dtype
---  ------ --------------   -----
0    time    18398 non-null   datetime64[ns]
 1   y       18398 non-null   int64
2    x1      18398 non-null   float64
3    x2      18398 non-null   float64
4    x3      18398 non-null   float64
5    x4      18398 non-null   float64
6    x5      18398 non-null   float64
7    x6      18398 non-null   float64
8    x7      18398 non-null   float64
9    x8      18398 non-null   float64
10   x9      18398 non-null   float64
11   x10     18398 non-null   float64
12   x11     18398 non-null   float64
13   x12     18398 non-null   float64
14   x13     18398 non-null   float64
15   x14     18398 non-null   float64
16   x15     18398 non-null   float64
17   x16     18398 non-null   float64
18   x17     18398 non-null   float64
19   x18     18398 non-null   float64
20   x19     18398 non-null   float64
21   x20     18398 non-null   float64
22   x21     18398 non-null   float64
23   x22     18398 non-null   float64
24   x23     18398 non-null   float64
```

```
 25  x24        18398    non-null  float64
 26  x25        18398    non-null  float64
 27  x26        18398    non-null  float64
 28  x27        18398    non-null  float64
 29  x28        18398    non-null  int64
 30  x29        18398    non-null  float64
 31  x30        18398    non-null  float64
 32  x31        18398    non-null  float64
 33  x32        18398    non-null  float64
 34  x33        18398    non-null  float64
 35  x34        18398    non-null  float64
 36  x35        18398    non-null  float64
 37  x36        18398    non-null  float64
 38  x37        18398    non-null  float64
 39  x38        18398    non-null  float64
 40  x39        18398    non-null  float64
 41  x40        18398    non-null  float64
 42  x41        18398    non-null  float64
 43  x42        18398    non-null  float64
 44  x43        18398    non-null  float64
 45  x44        18398    non-null  float64
 46  x45        18398    non-null  float64
 47  x46        18398    non-null  float64
 48  x47        18398    non-null  float64
 49  x48        18398    non-null  float64
 50  x49        18398    non-null  float64
 51  x50        18398    non-null  float64
 52  x51        18398    non-null  float64
 53  x52        18398    non-null  float64
 54  x54        18398    non-null  float64
 55  x55        18398    non-null  float64
 56  x56        18398    non-null  float64
 57  x57        18398    non-null  float64
 58  x58        18398    non-null  float64
 59  x59        18398    non-null  float64
 60  x60        18398    non-null  float64
 61  y.1        18398 non-null     int64
dtypes: datetime64[ns](1), float64(58), int64(3)
memory usage: 8.7 MB
```

```
#Counting the no. of Null values for each column
df.isnull().sum()
```

```
0
time
0
y
0
x1
0
x2
0
x3
..
0
x57
```

```
x58      0
x59      0
x60      0
y.1      0
Length: 62, dtype: int64
```

```
#Finding Duplicate values
print(df.duplicated())
```

```
0        False
1        False
2        False
3        False
4        False
         ...
18393    False
18394    False
18395    False
18396    False
18397    False
Length: 18398, dtype: bool
```

```
#Dropping any duplicate values
df.drop_duplicates()
```

```
                      time y        x1        x2        x3          x4  \
0      1999-05-01 00:00:00 0  0.376665 -4.596435 -4.095756   13.497687
1      1999-05-01 00:02:00 0  0.475720 -4.542502 -4.018359   16.230659
2      1999-05-01 00:04:00 0  0.363848 -4.681394 -4.353147   14.127997
3      1999-05-01 00:06:00 0  0.301590 -4.758934 -4.023612   13.161566
4      1999-05-01 00:08:00 0  0.265578 -4.749928 -4.333150   15.267340
...                    ... ..       ...       ...       ...         ...
18393 1999-05-28 23:58:00 0 -0.877441  0.786430  0.406426  135.301215
18394 1999-05-29 00:00:00 0 -0.843988  0.633086  0.561918  133.228949
18395 1999-05-29 00:02:00 0 -0.826547  0.450126  0.334582  134.977973
18396 1999-05-29 00:04:00 0 -0.822843  0.419383  0.387263  135.658942
18397 1999-05-29 00:06:00 0 -0.840981  0.582710  0.593416  136.339880

              x5          x6          x7          x8  ...        x51
```

```
x52 \
0 -0.118830 -20.669883 0.000732 -0.061114 ... ... 29.984624
10.091721 ... ... 29.984624
1 -0.128733 -18.758079 0.000732 -0.061114 ... ... 29.984624
10.095871 ... ... 29.984624
2 -0.138636 -17.836632 0.010803 -0.061114 ... ... 29.984624
10.100265 ... ...
3 -0.148142 -18.517601 0.002075 -0.061114 29.984624
10.104660 29.984624
4 -0.155314 -17.505913 0.000732 -0.061114 29.984624
10.109054 29.984624
... ... 18393 ... ... ... ... 29.984624
0.112295 26.300392 -0.159185 0.058823
0.058823 -
0.773514 0.048752 -
18394 0.141332 25.678597 -0.159185 0.048752 -
0.773514 0.048752 -
18395 0.170370 25.056801 -0.159185 -
0.773514
18396 0.199422 24.435005 -0.159185
0.773514
18397 0.228460 24.712960 -0.159185
0.773514

x54 x55 x56 x57 x58 x59
x60 \
0 -4.936434 -24.590146 18.515436 3.473400 0.033444 0.953219
0.006076 2.682933 0.033536 1.090502
1 -4.937179 -32.413266 22.760065
0.006083 3.537487 0.033629 1.840540
2 -4.937924 -34.183774 27.004663 3.986095 0.033721 2.554880
0.006090
3 -4.938669 -35.954281 21.672449 3.601573 0.033777 1.410494
0.006097
4 -4.939414 -37.724789 21.907251 ... ... ...
0.006105 6.944644 -37.795661 -0.860218 0.010220
... ... 0.507755 -39.357199 -0.915698 0.010620
...
18393 2.682413 2.809146 -39.357199 -1.409596 0.013323 0.895685 -
0.011242 2.164859 -39.357199 -0.860218 0.012888 0.175348 -
18394 2.683338
0.011235 1.416690 -39.357199 -0.732044 0.012453 0.621020 -
18395 2.684263
0.011228 1.390902 -
18396 2.685189 0.418993 -
0.011221
18397 2.686114
0.011214
```

```
              y.1
0  1  2  3     0
4       ...    0
18393          0
18394          0
18395          0
18396          ...
18397          0
               0
               0
               0
              0

[18398 rows x 62 columns]
```

```
#Counting Unique value in each column
print(df.nunique())
```

```
time      18398
y             2
x1        14091
x2        15768
x3        16615
            ...
x57        1112
x58       13025
x59       12225
x60       10800
y.1           2
Length: 62, dtype: int64
```

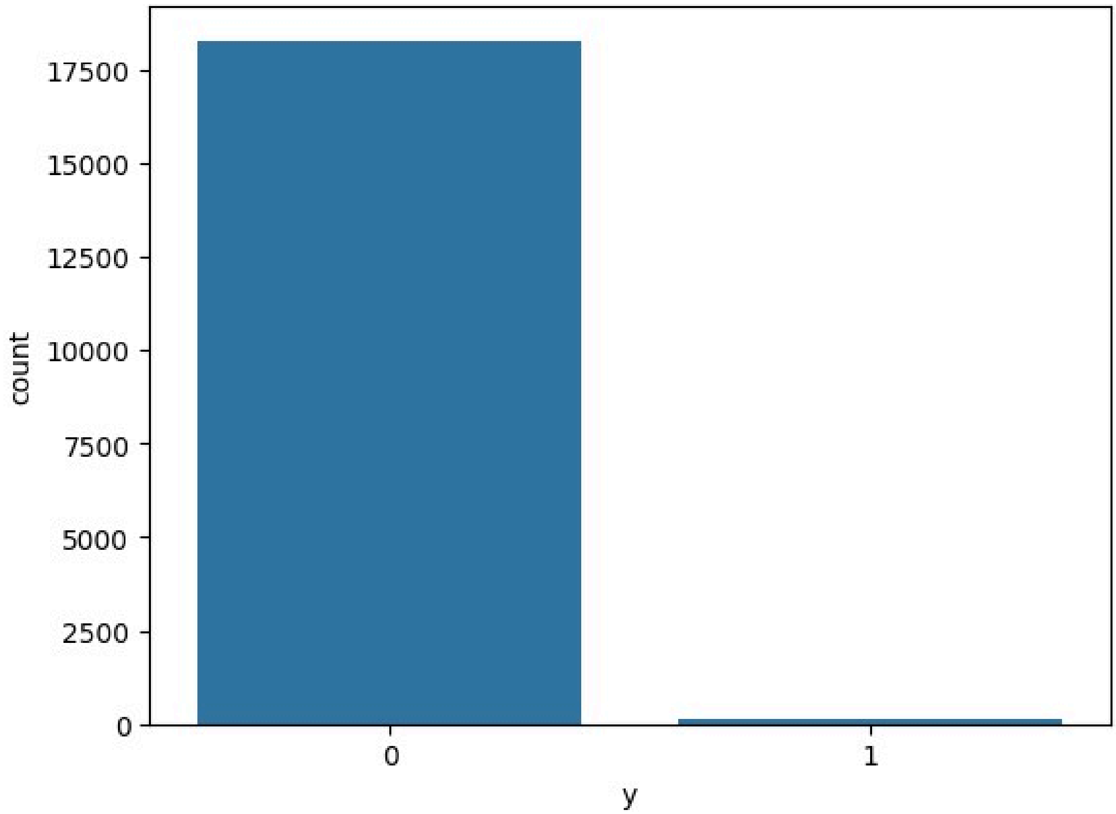
```
### Seperating Input and Output columns
X = df.drop(['y','time'],axis = 1)
Y = df['y']
```

```
##Zeroes and ones count in column 'y'
df['y'].value_counts()
```

```
y
0        18274
1          124
Name: count, dtype: int64
```

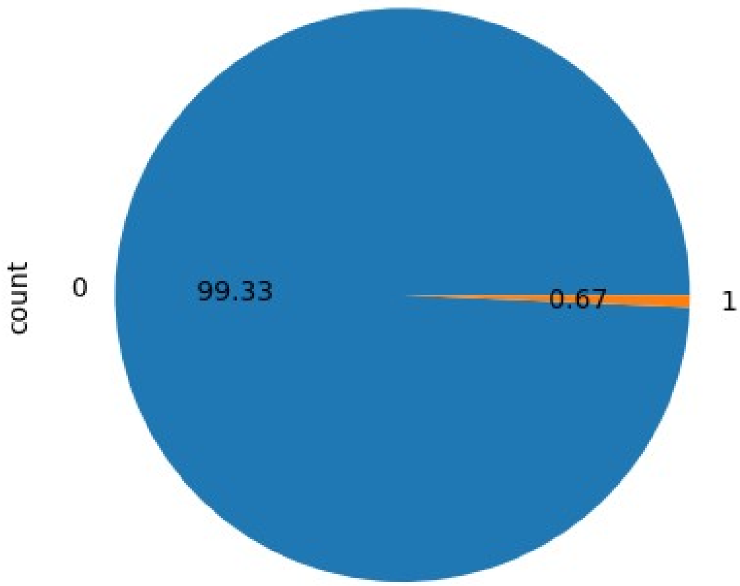
```
# Representing Zeroes and ones count in bar chart
sns.countplot(x ='y',data = df)
```

```
<Axes: xlabel='y', ylabel='count'>
```




```
#Representing Zeroes and ones count in pie plot
df['y'].value_counts().plot.pie(autopct = '% .2f')
```

```
<Axes: ylabel='count'>
```




```
# % of ones' count in the dataset
print(f"{124/18274*100 :.3f} %")
```

```
0.679 %
```

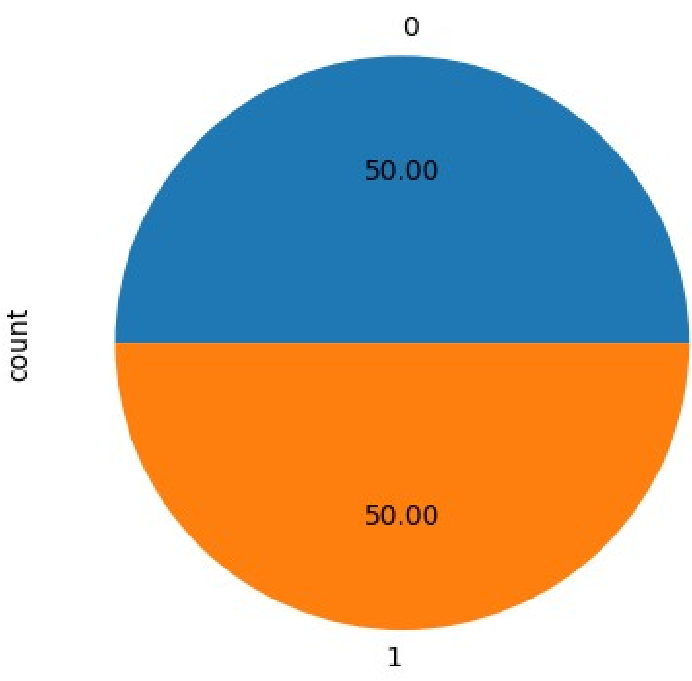
Data is extremely imbalanced

Percentage of data belonging to minority class	Degree of imbalance
20-40% of the dataset	Mild
1-20% of the dataset	Moderate
<1% of the dataset	Extreme

```
# Now Install library which will help to balance the dataset !pip install -U imbalanced-learn
```

```
Requirement already satisfied: imbalanced-learn in c:\users\
Gulzar.alam\appdata\local\anaconda3\envs\Gulzar\lib\site-packages
(0.12.4)
Requirement already satisfied: numpy>=1.17.3 in c:\users\
Gulzar.alam\appdata\local\anaconda3\envs\Gulzar\lib\site-packages
(from imbalanced-learn) (1.26.4)
Requirement already satisfied: scipy>=1.5.0 in c:\users\Gulzar.alam\
appdata\local\anaconda3\envs\Gulzar\lib\site-packages (from
imbalanced-learn) (1.14.1)
```

```
Requirement already satisfied: scikit-learn>=1.0.2 in c:\users\Gulzar.alam\appdata\local\anaconda3\envs\Gulzar\lib\site-packages (from imbalanced-learn) (1.5.2) Requirement already satisfied: joblib>=1.1.1 in c:\users\Gulzar.alam\appdata\local\anaconda3\envs\Gulzar\lib\site-packages (from imbalanced-learn) (1.4.2) Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\Gulzar.alam\appdata\local\anaconda3\envs\Gulzar\lib\site-packages (from imbalanced-learn) (3.5.0)

""" We will use Oversampling technique - SMOTE, to balance the dataset
which will scale up the minority classes to match up with the majority
class """

' We will use Oversampling technique - SMOTE, to balance the dataset
which will scale up the minority classes to match up with the majority
class '

from imblearn.over_sampling import SMOTE

sm = SMOTE(random_state=42)
X_res, Y_res = sm.fit_resample(X, Y)
ax = Y_res.value_counts().plot.pie(autopct='%.2f')
```




```
X_res
```

```
                  x1        x2         x3          x4         x5          x6
\ 0         0.376665 -4.596435 -4.095756   13.497687 -0.118830 -20.669883
1           0.475720 -4.542502 -4.018359   16.230659 -0.128733 -18.758079
2           0.363848 -4.681394 -4.353147   14.127997 -0.138636 -17.836632
3           0.301590 -4.758934 -4.023612   13.161566 -0.148142 -18.517601
4           0.265578 -4.749928 -4.333150   15.267340 -0.155314 -17.505913
...                                   ...                                ...
36543       0.123190 -5.907143 -5.243223         ...   0.050505 -20.217502
                                          -70.876037
36544       1.713375 -0.701378 -9.644510   35.501460 -0.492367 -80.683381
36545 -0.613582 -1.478861 -1.963563                                9.743904
36546 -0.325275 -9.779309 -11.727837  145.395851 -0.103702       36.268184
                                       89.769610   0.062049
36547 -0.301383 0.568078 -9.652514    -86.446607 -0.125511 -64.649525

                  x7        x8         x9        x10 ...          x51
x52    \
0           0.000732 -0.061114 -0.059966 -0.038189 ...  ...  29.984624
10.091721
                                                    ...  ...  29.984624
1       0.000732 -0.061114 -0.059966 -0.038189
10.095871                                           ...  ...  29.984624
2       0.010803 -0.061114 -0.030057 -0.018352
                                                    ...  ...  29.984624
10.100265
3       0.002075 -0.061114 -0.019986 -0.008280      ...  ...  29.984624
10.104660
                                                    ...             ...
4       0.000732 -0.061114 -0.030057 -0.008280
10.109054                                                     31.469783
...            ...       ...       ...        ...                        .
                                                              29.984624
..
36543       0.208618 0.089437 -0.058252 -0.009017
                                                              29.984624
0.886796
                                                              31.309173
36544   0.095394 0.018844 0.072664     0.034269                          -
4.665145                                                      29.554408
                                                                         -
36545   0.005235 0.023728 -0.110563 -0.078383
1.272212
36546 -0.059389 -0.037282 -0.001641 -0.031279
0.481610
36547 -0.001525 -0.085744 -0.049731 -0.035550                            -
1.780379

                 x54        x55        x56        x57        x58        x59
```

```
\ 0
            -4.936434 -24.590146   18.515436  3.473400  0.033444  0.953219
1
            -4.937179 -32.413266   22.760065  2.682933  0.033536  1.090502
2
            -4.937924 -34.183774   27.004663  3.537487  0.033629  1.840540
3
            -4.938669 -35.954281   21.672449  3.986095  0.033721  2.554880
4
            -4.939414 -37.724789   21.907251  3.601573  0.033777  1.410494
...
                 ...         ...         ...       ...       ...       ...
36543
             1.348418 147.223957  -46.214293  1.460570 -0.100675  0.989833
36544 -5.850424 -87.113012 -117.867780 -2.174484
                                                       0.030733  0.512385

                                                       0.021017  3.595538
36545        1.887315 -34.705985  -40.410241 -0.712796
                                                       0.015123  3.079882
36546 -4.983965 -39.008923         61.291453 -0.140618
                                                       0.016106  2.684745
36547 -3.012029 -28.471037        -71.361523 -1.177984


                x60 y.1
0       0.006076    0
1       0.006083    0
2       0.006090    0
3       0.006097    0
4       0.006105    0
...          ...  ...
36543   0.000255    0
36544   0.000855    0
36545   0.006391    0
36546   0.010029    0
36547   0.007003    0

[36548 rows x 60 columns]

Y_res

0        0
1        0
2        0
3        0
4        0
        ..
36543    1
36544    1
36545    1
36546    1
```

```
36547      1
Name: y, Length: 36548, dtype: int64
```

```
Y_res.value_counts()
```

```
y
0       18274
1       18274
Name: count, dtype: int64
```

Data Splitting

```
# Splitting the data into training and testing dataset from
sklearn.model_selection import train_test_split X_train, X_test, Y_train,
Y_test = train_test_split(X_res, Y_res,
test_size = 0.2, random_state=123)
X_train
```

```
\
15236     x3 0.340890 0.954831 -7.371992          x4          x5          x6
                                          190.530829     1.019521   -9.557884
3467      0.112985 -1.955501 -3.326003 -112.113960 -0.016917
                                                                      6.881905
8732
                                                                    -16.111809
15750       0.299043 -2.908362 -6.682374  193.899542 -0.272105
                                                                     22.860329
24044 -0.090876 4.521304 -9.385014918  189.186743 -0.243647
                                                                     32.608301
                                          218.398094    0.097490
                                                                           ...
...              ...           ...              ...          ...          ...
                                                                     26.937905
7763          -0.509073 1.612657 -4.989895    244.561713 -0.687235
                                                                    -13.812401
15377           0.421222 -1.813448 -6.310049    187.502356 -0.000941
                                                                     36.687660
17730           0.069523 2.977344 3.374720  56.548499     0.078710
                                              -23.532758 -0.203797 -128.292789
28030       1.484900 -3.953427 -9.692695

15725           0.703519 2.901412 -6.339217  193.932379    0.194051     24.198830


                x7          x8          x9         x10 ...         x51
\x52
15236     0.020874 -0.005866 -0.049021 -0.048260 ... ...    29.984624 -
3.128250
                                                             29.984624
3467   -0.069155 -0.031206 -0.070037 -0.068402
```

```
4.137131
8732     0.130741 -0.081257 -0.070037 -0.078168        ...   ...  28.660161
0.294846                                                   ...   ...  29.984624  -
15750 -0.019409 0.008773 -0.000149        0.001791     ...   ...  29.984624 ...
10.099442                                                  ...   ...  29.984624
24044 -0.016794 0.027563 -0.018847 -0.052779         ...         29.984624   -
6.906621                                                          29.931645
...                  ...              ...              ...                 ...            29.984624              .
..                 0.090558 0.202372 -0.095656 -0.075637           29.984624
7763
4.262131
15377 -0.049319 -0.061114 -0.040129 -0.058331                                -
3.753738                                                                      -
17730 -0.039246 0.008773 -0.040129 -0.002573                                 -
1.689041                                                                      -
28030   0.075331 0.001054 0.245567          0.003837
7.755202                                    0.171779
15725 -0.019409 -0.011064 0.049901
9.184891

                  x54              x55               x56              x57              x58
x59    \
15236 -4.951394 -67.547421             15.551843 -2.047413   0.009474    0.150034

3467          0.626401 30.327945    94.529779    2.972911       0.029428    0.338144

8732       -4.904975 -57.939633    32.866907    1.642284        0.011228     1.421931

             15750 -4.972492 -96.391781 -51.338217 -3.149097 -0.005344    0.021227

           24044 -4.841939 -87.818713 -39.707157 -3.459498 -0.006421    0.479616

...                   ...                    ...                  ...              ...               ...               ...

7763       -4.982663 -20.718075       68.139123      2.222118      0.014912      1.160845

15377 -4.941107 -66.666318 -51.538504 -1.775807              0.016013      1.632204

17730    1.343881 75.917057              65.735406 -1.775807    0.013303      0.180338

28030 -4.918541 -75.584446 -150.120545 -1.862119            0.015198      0.878159

            15725 -4.935096 -94.496762 -50.603201 -3.622180 -0.010034 -0.047285


                       x60 y.1
15236      0.003592       0
3467       0.006674       0
8732       -0.001253      0
  15750 -0.000550         0
```

```
24044 -0.000550        0
...                  ... ...
7763      -0.001995    0
15377      0.008914    0
17730 -0.006130        0
28030 0.000315         0
15725 -0.000550       0

[29238 rows x 60 columns]
```

Y_train

```
18609    1
21881    1
6061     0
17156    0
13041    0
         ..
7763     0
15377    0
17730    0
28030    1
15725    0
Name: y, Length: 25583, dtype: int64
```

Feature Engineering

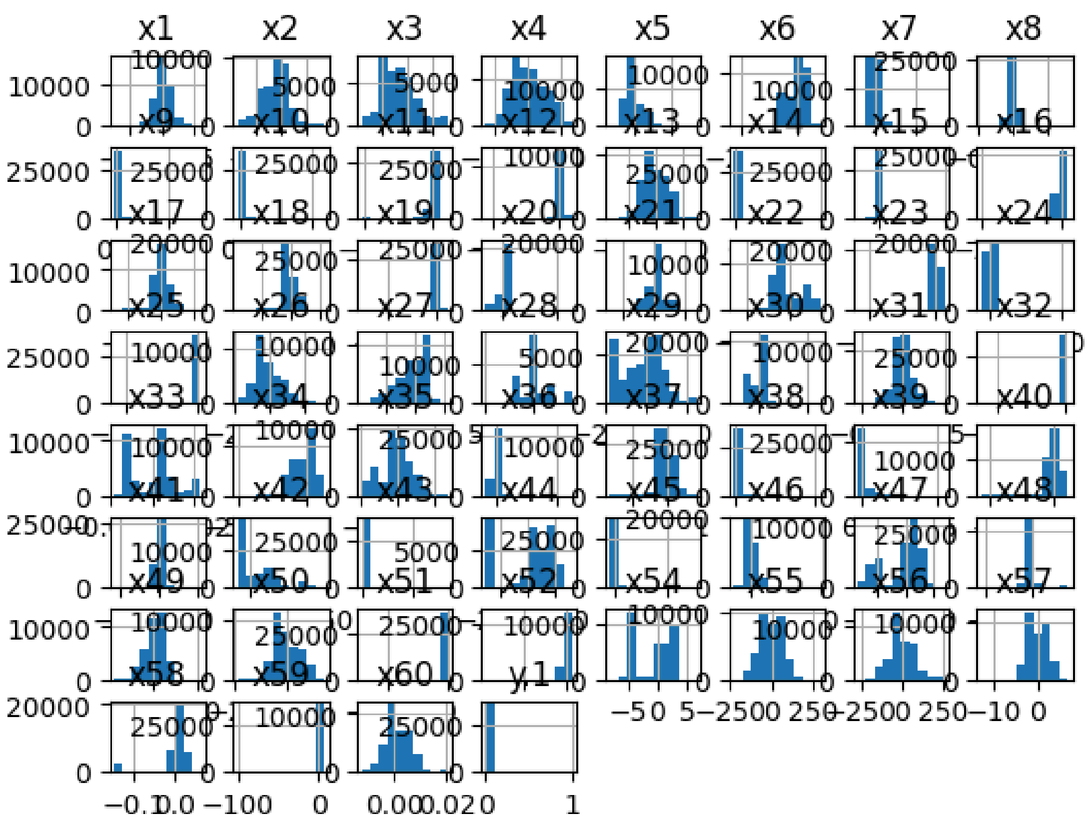
```
#Will convert raw data to useful features for ML model #success of a ML model largely depends on the quality of the features used in the model.

# Extracting all numeric features in one variable
num_cols = X_res._get_numeric_data().columns
num_cols
```

```
Index(['x1', 'x2', 'x3', 'x4', 'x5', 'x6', 'x7', 'x8', 'x9', 'x10',
'x11',
'x21',    'x12', 'x13', 'x14', 'x15', 'x16', 'x17', 'x18', 'x19', 'x20', 'x22', 'x23', 'x24', 'x25',
'x31',    'x26', 'x27', 'x28', 'x29', 'x30', 'x32', 'x33', 'x34', 'x35', 'x36', 'x37', 'x38', 'x39',
'x41',    'x40', 'x42', 'x43', 'x44', 'x45', 'x46', 'x47', 'x48', 'x49', 'x50', 'x52', 'x54',
'x51',    'x55', 'x56', 'x57', 'x58', 'x59', 'x60', 'y.1'],

      dtype='object')
```

```python
# Plot the histograms
X_res[num_cols].hist(bins=10)
```

```
array([[<Axes: title={'center': 'x1'}>, <Axes: title={'center':
'x2'}>,
          <Axes: title={'center': 'x3'}>, <Axes: title={'center': <Axes: title=
'x4'}>,
          {'center': 'x5'}>, <Axes: title={'center': <Axes: title={'center':
'x6'}>,
          'x7'}>, <Axes: title={'center':
'x8'}>],
        [<Axes: title={'center': 'x9'}>, <Axes: title={'center':
'x10'}>,
          <Axes: title={'center': 'x11'}>, <Axes: title={'center': <Axes: title=
'x12'}>,
          {'center': 'x13'}>, <Axes: title={'center': <Axes: title={'center':
'x14'}>,
          'x15'}>, <Axes: title={'center':
'x16'}>],
        [<Axes: title={'center': 'x17'}>, <Axes: title={'center':
'x18'}>,
          <Axes: title={'center': 'x19'}>, <Axes: title={'center': <Axes: title=
'x20'}>,
          {'center': 'x21'}>, <Axes: title={'center': <Axes: title={'center':
'x22'}>,
          'x23'}>, <Axes: title={'center':
'x24'}>],
        [<Axes: title={'center': 'x25'}>, <Axes: title={'center':
'x26'}>,
          <Axes: title={'center': 'x27'}>, <Axes: title={'center': <Axes: title=
'x28'}>,
          {'center': 'x29'}>, <Axes: title={'center': <Axes: title={'center':
'x30'}>,
          'x31'}>, <Axes: title={'center':
'x32'}>],
        [<Axes: title={'center': 'x33'}>, <Axes: title={'center':
'x34'}>,
          <Axes: title={'center': 'x35'}>, <Axes: title={'center': <Axes: title=
'x36'}>,
          {'center': 'x37'}>, <Axes: title={'center': <Axes: title={'center':
'x38'}>,
          'x39'}>, <Axes: title={'center':
'x40'}>],
        [<Axes: title={'center': 'x41'}>, <Axes: title={'center':
'x42'}>,
          <Axes: title={'center': 'x43'}>, <Axes: title={'center': <Axes: title=
'x44'}>,
          {'center': 'x45'}>, <Axes: title={'center': <Axes: title={'center':
'x46'}>,
          'x47'}>, <Axes: title={'center':
```

```
'x48'}>],
          [<Axes: title={'center': 'x49'}>, <Axes: title={'center':
'x50'}>,
           <Axes: title={'center': 'x51'}>, <Axes: title={'center': <Axes: title=
'x52'}>,   {'center':  'x54'}>,  <Axes:  title={'center':  <Axes:  title={'center':
           'x56'}>, <Axes: title={'center':
'x55'}>,

'x57'}>],
          [<Axes: title={'center': 'x58'}>, <Axes: title={'center':
'x59'}>,
           <Axes: title={'center': 'x60'}>, <Axes: title={'center':
'y.1'}>,   <Axes: >, <Axes: >, <Axes: >, <Axes: >]], dtype=object)
```



Dimensionality reduction - Principal Componenet analysis

Basically Reducing the number of features by transforming the data into a lower-dimensional space while retaining important information.

```
#Observing correlation b/w all features
X_train[num_cols].corr()
```

x1 x2 x3 x4 x5 x6
x7 \ x1 1.000000 0.084069 -0.105779 0.163079 -0.025989 -0.174184
0.206807

x2 0.084069 1.000000 0.521845 -0.060098 0.053239 0.003550 - 0.111763
0.105085 0.309316 - 1.000000
x3 -0.105779 0.521845 1.000000 -0.200288 -0.028036 -0.068325 - 1.000000 -
0.053907
x4 0.163079 -0.060098 -0.200288 1.000000 0.007818 0.078818
0.007755 0.309316 -0.068325
x5 -0.025989 0.053239 -0.028036
0.041858
x6 -0.174184 0.003550 0.111763
0.185740
x7 0.206807 -0.105085 0.053907 -0.007755 -0.041858 -0.185740
1.000000
x8 0.164196 0.170058 0.088091 0.063350 0.224671 0.064808
0.417490 0.088119 -0.112473
x9 0.007810 -0.087101 -0.058924 -0.008695 0.104906 -0.082165
0.305754 0.071768
x10 0.000231 -0.013258 0.026922 -0.062910 0.090622
0.129676 0.667859 -
x11 0.320249 0.266533 0.207619 0.004112 -0.059185
0.210842 0.059572
x12 0.428606 0.245025 0.189053 -0.026617 0.177492 -0.080192
0.309224
x13 0.014091 -0.063304 -0.011347
0.022321
x14 0.145603 0.049442 0.013414 -0.130440 -0.173737 -0.000178
0.012144
x15 -0.036054 -0.083182 -0.023861 -0.153299 0.064212 -0.237913 -
0.093415 0.020649 -0.047121
x16 0.339653 0.250920 0.255780 -0.081105
0.251208
x17 0.050072 0.323387 0.107934 0.116949 0.149184 0.169883 - 0.152054 -
0.192899 0.148363 0.124581 0.123331 0.017039
x18 0.081340 0.327707 0.113617 0.194335 -0.113567
0.239691
x19 -0.103546 0.073601 0.121728 -0.005982 -0.193920
0.034020
x20 0.014971 -0.073041 -0.034480 -0.065070 -0.151415
0.159769
x21 0.250143 0.057564 -0.181906 0.214502
0.042947 0.181531
x22 0.100446 0.222220 -0.014500 0.098037 0.566459 0.040472 -
0.138232 0.306208 0.140359
x23 -0.075332 0.004249 0.010131
0.068384
x24 -0.117655 -0.048825 0.021131 0.010696 -0.190925 -0.143766 -
0.099946

x25 0.354985 0.169349 0.165143 -0.041633 -0.019531 -0.109918
0.315253
x26 -0.126878 0.014028 0.237707 -0.275519 -0.034688 0.092735
0.096636 0.191164
x27 -0.033811 0.072479 0.187933 -0.097783 0.002436 0.023218 -
0.055576 0.072629 0.371444 -
x28 -0.225437 -0.098550 0.268833 -0.033342 0.045342
0.016594
x29 -0.355779 0.054562 0.356976 -0.603911
0.136738
x30 -0.145379 -0.188276 0.103788 -0.217991 -0.604530 -0.027546
0.140615
x31 0.010293 0.105333 0.086008 0.071137 -0.034827 0.099671 -
0.032405
x32 0.388889 0.204339 0.166043 0.007985 -0.023205 -0.058750
0.293818
x33 -0.291524 -0.208444 0.135354 -0.321745 -0.432638 -0.087390
0.067388
x34 -0.250226 -0.018225 0.137856 0.264800 -0.086218 0.967869 -
0.223604
x35 -0.038592 -0.088866 0.149440 -0.323515 -0.263395 -0.164077
0.098228
x36 0.029763 -0.010939 0.401174 -0.168193 0.106319 0.092242
0.518706 0.397789 0.169903
x37 0.184506 0.224059 0.241953 -0.010991 0.055504 -0.097919
0.139277 0.049226 -0.051425
x38 0.006614 -0.092014 -0.028417 -0.007408
0.291839
x39 0.031982 -0.055870 -0.026558 -0.021551
0.116548
x40 0.123050 0.204192 0.266117 -0.184548 [illegible] 0.029249
0.162333 -0.063147 0.089733
x41 0.049128 -0.013641 -0.039084 0.026326 -0.248708 0.205902
0.021674 0.011934 -0.033896 -
x42 0.170231 0.072263 -0.140190 0.041439
0.136795
x43 -0.048023 0.056959 0.067263
0.010286
x44 -0.213603 -0.143747 0.086158 -0.483883 -0.236929 -0.162198
0.153753
x45 -0.348889 -0.221391 -0.142414 0.027515 -0.011268 0.029843 -
0.207878
x46 0.042921 -0.206694 -0.020279 0.124609 -0.094658 -0.086779
0.244275
x47 0.160704 0.119619 0.017012 0.138110 [illegible] 0.292972
0.072692 0.059263 [illegible] -0.016863
x48 0.244082 -0.123402 -0.057050 0.022677
0.196465
x49 0.019593 -0.170183 -0.039402
0.192068

```
x50    0.134269 0.061094 0.019675 -0.275122 -0.358936              0.008763 -
0.043239
x51    0.372485 0.189591 0.153676              0.002815 -0.000060 -0.061855
0.286642                                       0.067635
x52    0.063473 0.076846 0.064336 -0.103462 0.250196 -0.285335
0.018765                                       0.210419 -
x54 -0.365939 0.127888 0.429812 -0.467034 0.342087 - 0.159760
0.158177                                       0.043529    0.195624
x55 -0.182015 0.118343 0.071921 -0.283370
0.003098
x56 -0.034292 0.045621 0.050182 -0.047020 -0.337245
0.154005
x57    0.186036 -0.018981 0.000267 -0.106942 -0.416979
0.067389
x58    0.062366 0.161531 -0.241157             0.014033 -0.042278 -0.146758 -
0.342651
x59    0.351140 0.182400 0.166398 -0.009013 -0.011512 -0.083605
0.289545
x60 -0.175277 -0.145551 -0.060453 -0.324576            0.023247    -0.398500
0.120603                                                 0.119758 -0.125980
y.1 -0.051528 -0.158613 -0.147230 -0.084354
0.160705

                 x8          x9         x10   ...          x51         x52
x54    \
x1     0.164196 0.007810 0.000231        ...       0.372485  0.063473 -0.365939

x2     0.170058 -0.087101 -0.013258      ...        0.189591  0.076846    0.127888

x3     0.088091 -0.058924 0.026922       ...       0.153676  0.064336    0.429812

x4     0.063350 -0.008695 -0.062910      ...         0.002815 -0.103462 -0.467034

x5     0.224671 0.088119 0.104906        ... -0.000060 -0.285335          0.159760

x6     0.064808 -0.112473 -0.082165      ... -0.061855        0.067635    0.250196

x7     0.417490 0.305754 0.129676        ...      0.286642         0.018765 -0.158177

x8     1.000000 0.054682 0.084748        ...      0.381006 -0.104123        0.183303

x9     0.054682 1.000000 0.623213        ...         0.053660 -0.046796 -0.055306

x10    0.084748 0.623213 1.000000        ...      0.067390 -0.039241       0.071604

x11    0.290835 0.050497 0.054981        ...        0.491499  0.050085     0.015080

x12    0.525598 0.054003 0.103940        ...       0.786865  0.064981     0.097660

x13 -0.010814 -0.082140 -0.072492        ... -0.059703          0.234750 -0.004515
```

x14 -0.011147 -0.026306 -0.025038 ... 0.013041 0.127976 -0.031930 x15 -0.134794 0.038151 0.006471 ... -0.001058 -0.048362 0.046950 x16 0.443987 0.012746 0.085831 ... 0.803574 0.007357 0.198592 x17 0.094280 -0.094052 0.024152 ... -0.016968 0.014475 0.142548 x18 0.075986 -0.115021 -0.016863 ... -0.039415 0.117424 0.144883 x19 -0.060798 -0.171601 -0.011407 ... 0.003066 0.110315 -0.012959 x20 -0.091784 0.252598 0.214279 ... 0.012938 0.078401 0.030746 x21 0.198948 -0.043518 -0.035521 ... 0.034520 -0.244845 -0.184157 x22 0.330107 -0.020311 0.062431 ... -0.007581 -0.291862 0.211410 x23 0.164155 -0.002448 0.093070 ... 0.000498 -0.267033 0.177342 x24 -0.123213 0.076803 -0.079980 ... -0.030638 -0.291327 -0.072906 x25 0.358927 0.055546 0.075390 ... 0.922441 0.061714 -0.065007 x26 0.126905 -0.040943 0.068967 ... 0.011307 0.051549 0.513467 x27 0.156947 -0.067218 0.070485 ... -0.004076 -0.046091 0.544594 x28 0.029582 -0.030765 0.023507 ... 0.004319 -0.169310 0.383947 x29 0.090017 -0.075062 0.019028 ... -0.055557 -0.015076 0.801815 x30 -0.335454 -0.030024 -0.085533 ... 0.019464 0.330261 -0.165822 x31 -0.031196 -0.022835 -0.053922 ... 0.006695 0.016740 0.019551 x32 0.383546 0.048066 0.066471 ... 0.930353 0.049949 -0.062486 x33 -0.303485 -0.048883 -0.065796 ... 0.024437 0.259457 -0.059335 x34 0.009224 -0.113706 -0.087405 ... -0.060816 0.105932 0.274402 x35 -0.064917 -0.113463 -0.017757 ... 0.077740 0.184269 0.121565 x36 0.358481 0.133217 0.130591 ... 0.269864 -0.087199 0.473111 x37 0.693164 -0.015669 0.081525 ... 0.455623 -0.142176 0.467503 x38 0.083042 0.765825 0.401966 ... 0.054277 -0.029996 -0.063736 x39 0.116466 0.356443 0.673197 ... 0.068700 -0.006022 0.022128 x40 x41 0.325660 0.014510 -0.023894 ... ... 0.452826 -0.004302 0.365446

x42 x43 x44 0.128385

-0.004185 0.042573 -0.024395 -0.396583 ... ... 0.004772 -0.064031 -0.085608

-0.025150 x45 0.049651

-0.027371 0.160590 0.009080 -0.068729 ... 0.034360 -0.220459 -0.123855

-0.012990 x46 0.085615

0.029632 -0.039901 -0.016504 -0.007140 0.008495 -0.004159 0.117603

0.039968

0.079299 0.254257 0.001486

... -0.890010 -0.027871 -0.013835

... 0.022119 0.010894 -0.189148

x47 x48 -0.342789 0.076564 0.115299 ... -0.020322 -0.089631 0.334569

0.017290 0.004251 -0.012529

x49 -0.192752 -0.046775 -0.136604 x50 ... ... 0.004677 ... ... -0.046108 0.115783 -0.287198

-0.181244 -0.171232 -0.048931 -0.115684 0.020227 0.283265 -0.313285

0.039721 0.560490 -0.104824

x51 0.381006 0.053660 0.067390 1.000000 0.018709 -0.046100

x52 -0.104123 -0.046796 -0.039241 0.018709 1.000000 -0.115684

x54 x55 0.163303 -0.175463 -0.087804 1.000000

x56 0.105306 0.075904

-0.087521 0.291674 0.173043 -0.128037 ... -0.029572 ... 0.089787 0.548485

x57 -0.011310 -0.158456

-0.083554 x58 -0.142560 -0.164641 -0.025544 0.319665 -0.001489

-0.169038 ... 0.046344 0.537975 -0.243889

... -0.017314 0.074990 -0.291449

x59 0.379018 0.058884 0.066013 ... ... 0.924150 0.027308 -0.062852

x60 -0.131704 0.041663 0.044417 ... 0.063602 0.061383 -0.123407

y.1 0.069936 0.222279 0.085825 0.009521 -0.187576 -0.014380

	x55	x56	x57	x58	x59	x60
y.1						
x1	-0.182015	-0.034292	0.186036	0.062366	0.351140	-0.175277 - 0.051528
x2	0.118343	0.045621	-0.018981	0.161531	0.182400	-0.145551 - 0.158613
x3	0.071921	0.050182	0.000267	-0.241157	0.166398	-0.060453 - 0.147230

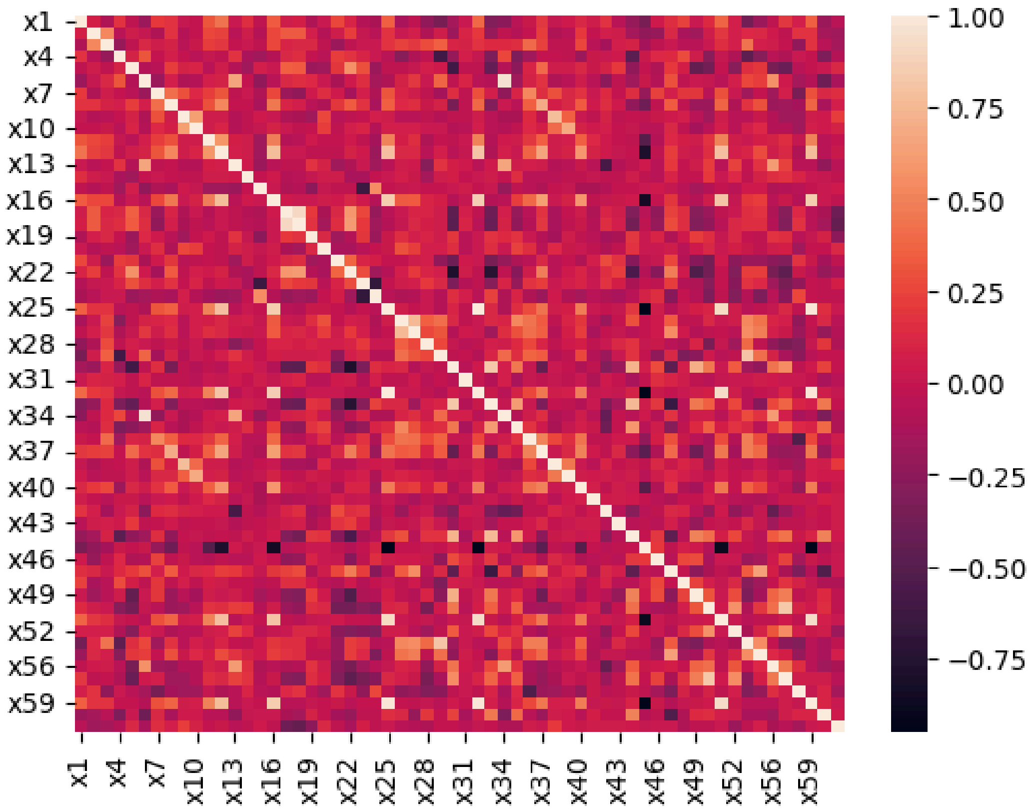
x4 -0.283370 -0.047020 -0.106942 0.014033 -0.009013 -0.324576 -
0.084354
x5 0.195624 -0.337245 -0.416979 -0.042278 -0.011512 0.023247
0.119758
x6 0.210419 0.572087 0.043529 -0.146758 -0.083605 -0.398500 -
0.125980
x7 -0.003098 -0.154005 0.067389 -0.342651 0.289545 0.120603
0.160705 0.379018 -0.131704
x8 0.291674 -0.175460 -0.173570 -0.142560
0.069936
x9 0.031043 -0.085944 -0.168036 -0.164641 0.058884 0.041663
0.222279 0.066013 0.044417
x10 0.127447 -0.087521 -0.083554 -0.169038
0.085825
x11 0.082135 0.085998 0.156182 0.007240 0.490636 -0.052056 -
0.001547 0.786132 -0.052224
x12 0.269005 0.057399 0.110558 -0.103582
0.029879
x13 0.178890 0.605483 0.238720 -0.058988 -0.065637 -0.239735 -
0.064467
x14 0.090877 0.140848 0.183529 0.107387 0.007361 - 0.010752
0.021963 0.017537
x15 0.045139 -0.090591 -0.090070 -0.0530973 -0.002769
0.045821 0.129937 -0.044476 -0.117180 -
x16 0.217869 -0.039012 0.030577 -0.078578 0.173498 -0.084759 0.067938 -0.146219 -
0.019926 0.091063 -
x17 0.171025 -0.101337 -0.279541 0.000261
0.399310
x18 0.144260 -0.081619 -0.305192
0.436355
x19 -0.028375 0.165336 0.151449 -0.047569 0.007148
0.256022
x20 -0.014645 0.058310 0.081878 -0.172903 -0.003189
0.120473
x21 -0.181867 -0.255696 -0.297027 0.180442 0.028349 -0.141292 -
0.027083 0.025962 -0.044263 -0.215280 -
x22 0.223985 -0.341842 -0.466326
0.134504
x23 0.129072 -0.160297 -0.334252 -0.233377 -0.012237 -0.024053
0.083527
x24 -0.333829 -0.042308 -0.268046 0.259437 -0.012956 -0.036271 -
0.036701
x25 -0.020787 0.001700 0.119576 -0.016771 0.988968 0.148081
0.018361 0.000762 0.110648 -
x26 0.421020 0.097478 0.072004 -0.221604
0.053731
x27 0.477857 0.071358 0.029206 -0.195111 -0.032318 -0.210280 -
0.085124
x28 -0.013892 -0.226716 -0.335661 -0.289299 0.008185 0.194042 -
0.102578

```
x29   0.419014 0.092074 -0.132074 -0.243121 -0.064137 -0.002107 -
0.001569
x30 -0.230699 0.434090 0.545362 -0.031721         0.045274      0.235431 -
0.001771                                          0.023608 -0.005124 -0.132893 -
x31   0.060473 0.046751 0.024088
0.053024
x32 -0.008163 0.005395 0.099245 -0.016853         0.991002    0.054479
0.008068                                          0.075821    0.447921 -
x33 -0.269966 0.265064 0.373454          0.103329
0.007298
x34   0.180294 0.605800 0.076459 -0.129058 -0.078073 -0.327000 -
0.111284
x35 -0.064795 0.171000 0.350463          0.125916     0.101222     0.384607 -
0.103372
x36   0.277621 -0.122127 -0.119594 -0.539611          0.262133      -0.061030
0.060406                                          0.455697 -0.176571
x37   0.384038 -0.182399 -0.215887 -0.177817
0.024127
x38   0.019173 -0.071275 -0.113096 -0.154447          0.063194      0.057358
0.203679                                          0.066431      0.041439
x39   0.086571 -0.022208 0.016623 -0.111179              0.470878 -0.051925
0.041757
x40   0.264165 0.012409 -0.036151 -0.201568
0.028963
x41 -0.093318 -0.060674 0.028081          0.029110     0.004881     0.080686
0.056631                                               0.026783 -0.012778
x42   0.006569 -0.434727 -0.187898 -0.045558          0.002485 -0.009164
0.062176
x43   0.004460 -0.162864 -0.033873 -0.027297
0.051532
x44 -0.055312 0.330996 0.425781          0.028908   0.104252    0.489931
0.041074
x45 -0.057850 0.030890 -0.000098 -0.028042 -0.947100 -0.070187
0.005229
x46 -0.148149 0.039551 0.242155 -0.144471             0.026767 -0.033001
0.021510
x47   0.647607 0.081878 -0.114531 -0.315054 -0.076627 -0.533136
0.009289
x48 -0.152061 -0.011870 0.172929          0.092620    0.002689 -0.103130 -
0.090816
x49 -0.136331 0.381148 0.469330 -0.032198             0.030427    0.137015     -
0.024065                                          0.045470    0.143944    -
x50   0.189267 0.410046 0.828841          0.087008   0.924150    0.063602
0.030883                                          0.027308    0.061383 -
x51 -0.029572 -0.025544 0.046344 -0.017314
0.009521
x52   0.089787 0.319665 0.537975          0.074990
0.187576
x54   0.548485 -0.001489 -0.243889 -0.291449 -0.062852 -0.123407 -
0.014380
```

```
x55    1.000000 0.130555 0.081863 -0.218429 -0.043398 -0.134590
0.068141
x56    0.130555 1.000000 0.447214        0.140274 -0.037379 -0.161349 -
0.114680
x57    0.081863 0.447214 1.000000        0.061533     0.050000     0.059524  -
0.054019                                 1.000000     0.005363     0.096954  -
x58 -0.218429 0.140274 0.061533          0.005363     1.000000     0.103409
0.148272                                 0.096954     0.103409     1.000000
x59 -0.043398 -0.037379 0.050000                      0.016150     0.037684
0.016150
x60 -0.134590 -0.161349 0.059524
0.037684
y.1     0.068141 -0.114680 -0.054019 -0.148272
1.000000

[60 rows x 60 columns]
```

```python
#Observing Correlation via heat map
sns.heatmap(data= X_train[num_cols].corr())
```

```
<Axes: >
```




```python
X_train.describe()
```

```
                     x1              x2               x3               x4
x5   \     29238.000000 29238.000000  29238.000000 29238.000000
count                                       -3.479189         5.436828
29238.000000                                 6.767939       129.591824
mean            0.074386       -2.388121  -18.198509      -322.781610   -
0.031725        0.731781        5.559442   -8.966241       -94.817109
std            -3.787279      -17.316550   -4.116914        -6.119270
0.584093       -0.304898       -6.269648    0.672001       102.899725
min             0.113318       -1.451957   15.900116       334.694098   -
1.623988        0.443885        0.784189                                -
25%             3.053444       16.742105                                -
0.404666
50%
0.141337
75%
0.211119
max
4.239385

                     x6              x7               x8               x9
\x10
count      29238.000000 29238.000000  29238.000000 29238.000000
29238.000000
                                            -0.004039         0.012431
mean           -4.680699        0.011974                                -
0.000600                                     0.082882         0.173839
std            40.940411        0.108611
                                            -0.451141        -0.120087
0.102495                       -0.429273
min          -279.408440                    -0.051043        -0.059966   -
0.098310                       -0.049319
                                            -0.011064        -0.029299   -
25%           -39.600153        0.000732
0.048260                                     0.038986         0.010131   -
               5.844685         0.060853
50%
0.018352                                     0.788826         3.206675
              26.849670         1.705590
75%
0.012368       96.060768
max
2.921802

           x52              x51                          x54          x55 \
count   ... 29238.000000 29238.000000           29238.000000  29238.000000
mean    ...        11.638219        -0.812641      -0.665653     -3.857103
std     ...       258.719785        10.461626       3.316290     65.721542
min     ...     -3652.989000      -187.702316      -6.569237   -209.886410
25%     ...        29.984624        -4.573589      -4.952808    -48.612269
50%     ...        29.984624        -1.454222       0.604562     -1.943421
75%     ...        29.984624         3.818460       2.394793     41.291979
max     ...        40.152348        14.180588       6.475137    287.252017

                    x56            x57                    x58          x59
x60   \
```

```
count   29238.000000 29238.000000   29238.000000  29238.000000
29238.000000                                -0.000157       0.475719
mean        -3.017051          0.037325     0.044493        7.737157
0.001734    75.588047          2.252481    -0.149790    -100.810500
std                          -12.640370    -0.000449        0.391867
0.004767                      -1.726978     0.013693        0.804750
min       -269.039500         -0.219349     0.020921        1.275744     -
0.012229                       1.874218     0.067249        6.985460     -
25%          -51.596782        6.922008
0.001514      -16.215734
50%           48.139368
0.000972    252.147455
75%
0.005536
max
0.020495

                   y.1
count   29238.000000
mean        0.018264
std         0.133906
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000

[8 rows x 60 columns]
```

Scaling down the features

```
#Scaling the columns in order to have more uniform distribution - We apply Yeo-Johnson Transform from sklearn.preprocessing import PowerTransformer boxcox = PowerTransformer() X_scaled = boxcox.fit_transform(X_res) X_scaled
```

```
array([[ 3.81889646e-01, -3.98515966e-01, -1.46922678e-03, ...,


         -1.64173622e-01, 9.37183457e-01, -1.35716344e-01],
       [ 5.26239506e-01, -3.88792501e-01,        1.11631202e-02, ...,
         -6.064070736e-02, 9.38506198e-01, -1.35716344e-01],
       [ 3.63340078e-01, -4.13832850e-01, -4.36263146e-02, ...,
          5.89206738e-01, 9.39828687e-01, -1.35716344e-01],
       ...,
       [-9.54098977e-01, 1.63743754e-01,         3.37751551e-01, ...,
          2.61684068e+00, 9.96536882e-01, -1.35716344e-01],
       [-5.84283564e-01, -1.33257519e+00, -1.32131555e+00, ...,
          1.95163403e+00, 1.64554634e+00, -1.35716344e-01],
```

```
       [-5.53047055e-01, 5.33311277e-01, -9.50691774e-01, ...,
          1.48013870e+00, 1.11025352e+00, -1.35716344e-01]])

## To display X_scaled as a DataFrame
X_scaled_df = pd.DataFrame(X_scaled, columns=X_res.columns)
print(X_scaled_df)

              x1          x2          x3         x4         x5          x6
x7   \
0         0.381890 -0.398516 -0.001469   0.132479
                                     0.018342 -0.568235 -
0.038212
1       0.526240 -0.388793 0.011163   0.152415 -0.001269 -0.532933 -
0.038212
                                     0.137084 -0.021018 -0.515783
2       0.363340 -0.413833 -0.043626
0.059892                             0.130022 -0.040104 -0.528466 -
3       0.273669 -0.427812 0.010306  0.145397 -0.054588 -0.509605 -
0.025045
4       0.222136 -0.426188 -0.040343
0.038212
...          ...         ...         ...            ...       ...            ...
...
36543        0.020918 -0.634793 -0.191039 -0.561167    0.332184 -0.559914
1.733541
                                        0.291396 -0.814875 -1.576482
36544  2.449727 0.304055 -0.949277
0.830347
36545 -0.954099 0.163744 0.337752      1.063217    0.548023 1.130400   -
0.005827                                           0.154534
36546 -0.584284 -1.332575 -1.321316   0.675342    0.302128 -1.319566 -
0.654801                                           0.359089
36547 -0.553047 0.533311 -0.950692 -0.694546
0.060399

              x8          x9         x10  ...          x51         x52         x54
\
0         -0.689910 -0.622008 -0.338508  ...     0.080069     1.833449 -1.209384

1         -0.689910 -0.622008 -0.338508  ...     0.080069     1.834403 -1.209509

2          -0.689910 -0.087960 0.067208  ...     0.080069      1.835412 -1.209634

3           -0.689910 0.068052 0.248779  ...     0.080069      1.836422 -1.209759

4          -0.689910 -0.087960 0.248779  ...     0.080069      1.837432 -1.209885

...              ...         ...           ...  ...            ...          ...          ...

36543       1.141652 -0.588320 0.236020  ...
                                               1.521700 -0.005931     0.440542
36544        0.309154 1.080125 0.865474  ...      0.080069 -0.617209 -1.358684

36545       0.368190 -1.814855 -1.393327  ...
                                             0.080069 -0.285506    0.709579
```

```
36546 -0.385691  0.324870 -0.189516  ...  1.357514 -0.066459 -1.217363
36547 -1.010032 -0.426622 -0.280589  ... -0.306348 -0.340106 -0.861056

            x55       x56       x57       x58       x59       x60       y.1
0     -0.334412  0.392335  1.511999  1.575970 -0.164174  0.937183 -0.135716
1     -0.449437  0.441441  1.169347  1.584385 -0.060641  0.938506 -0.135716
2     -0.475430  0.490149  1.539732  1.592911  0.589207  0.939829 -0.135716
3     -0.501410  0.428900  1.733683  1.601364  1.332612  0.941151 -0.135716
4     -0.527378  0.431610  1.567458  1.606518  0.199535  0.942662 -0.135716
...         ...       ...       ...       ...       ...       ...       ...
36543  2.359045 -0.534413  0.636896 -2.093521 -0.137045 -0.254866 -0.135716
36544 -1.248274 -1.697927 -0.982147  1.336030 -0.462287 -0.122919 -0.135716
36545 -0.483094 -0.442690 -0.323708  0.592399  2.616841  0.996537 -0.135716
36546 -0.546205  0.874209 -0.068251  0.218555  1.951634  1.645546 -0.135716
36547 -0.391511 -0.937148 -0.532500  0.277252  1.480139  1.110254 -0.135716

[36548 rows x 60 columns]
```

```
## Now, further splitting the data into training and testing based on scaled values
X_train_sc, X_test_sc, Y_train_sc, Y_test_sc = train_test_split(X_scaled_df, Y_res, test_size = 0.2, random_state=123)

X_train_sc
```

```
             x1        x2        x3        x4        x5        x6        x7  \
15236  0.330189  0.603209 -0.552396  1.375394  1.605863 -0.356665  0.156556
3467   0.006659  0.077748  0.123198 -0.915552  0.212106  0.059107 -0.760029
8732   0.270016 -0.094126 -0.434108  1.398622 -0.300498 -0.483424  1.126667
15750  1.103654  0.846468 -0.169641  1.366123 -0.238843  0.621058 -
```

```
0.238843    24044    -0.273255    1.580371
-0.850291 0.212451 ... ... 7763                 1.567282    0.412282 0.988937 -
                                                      ...           ...
            ...          ...             ...    1.746937 -1.323684          ...
           -0.821480 0.722105 -0.148838                           0.773220

0.788706
15377    0.446606 0.103376 -0.370712        1.354504    0.241137 -0.439705 -
0.547782                                    0.441239
17730 -0.053813 0.968870 1.046099                     0.380602   1.146716 -
0.442265
28030   2.080371 -0.282583 -0.957799 -0.160547 -0.154380 -2.309889
0.655773
15725    0.864459 0.955137 -0.375666        1.398848    0.568669   0.670708 -
0.238843

              x8          x9          x10 ...          x51        x52         x54
\
15236     0.006987 -0.413570 -0.571424 ...      0.080069 -0.475127 -1.211898

3467     -0.309002 -0.828158 -1.098683 ...   0.080069   0.564684     0.117708

8732      -0.951280 -0.828158 -1.386757   ... -1.066749 -0.093162 -1.204088

15750       0.186706 0.344309 0.415446 ...     0.080069 -1.063350 -1.215439

24044     0.414388 0.084996 -0.682344 ...    0.080069 -0.809584 -1.193444

...            ...         ...          ... ...            ...       ...         ...

7763        2.389024 -1.420573 -1.309931 ...  0.080069      0.588845 -1.217145

    15377 -0.689910 -0.255504 -0.824317 ...     0.080069 -0.534201 -1.210169

17730      0.186706 -0.255504 0.344981 ...   0.031746 -0.330472      0.438372

28030       0.092198 1.887042 0.447601 ...    0.080069 -0.879002 -1.206373

15725 -0.057326 0.888957 1.872794       ...    0.080069 -0.992603 -1.209159


             x55         x56         x57         x58         x59         x60
y.1
15236 -0.963344 0.357764 -0.924658 -0.093350 -0.665835          0.451535 -
0.135716
                                                                 1.049281 -
3467     0.503471 1.236892 1.295178        1.225886 -0.565033
0.135716                                               0.209313 -0.595793 -
8732    -0.823146 0.556877 0.716288 -0.001015
0.135716
15750 -1.383172 -0.615817 -1.424266 -0.734708 -0.728555 -0.435098 -
```

```
0.135716 24044 -1.258539 -0.431617 -1.565465 -0.772945 -0.482283 -0.435098
- 0.135716 ... ... 7763

                    ...              ...                   ...        ...              ...              ...
                -0.277358 0.949527 0.969035
                                                            0.206117 -0.005688 -0.768783 -
0.135716
15377 -0.950495 -0.619007 -0.801917        0.271670    0.394823      1.453370 -
0.135716                                                  0.113386 -0.650324 -1.799586 -
17730   1.223906 0.923128 -0.801917              0.222943 -0.218674 -0.241566 -
0.135716
28030 -1.080471 -2.234050 -0.840901
0.135716
15725 -1.355632 -0.604117 -1.639535 -0.894073 -0.759767 -0.435098 -
0.135716

[29238 rows x 60 columns]

X_scaled_df.describe()

                          x1                     x2                   x3                   x4
x5    \
count     3.654800e+04 3.654800e+04  3.654800e+04  3.654800e+04
3.654800e+04
mean    6.221234e-18 -6.843358e-17 -1.244247e-17 -3.421679e-17 -
4.665926e-17
std       1.000014e+00 1.000014e+00     1.000014e+00    1.000014e+00
1.000014e+00
min     -4.543615e+00 -2.690137e+00 -2.515620e+00 -2.759972e+00 -
4.437926e+00
25%     -5.534697e-01 -7.032891e-01 -8.326755e-01 -7.616707e-01 -
6.044112e-01
50%       9.229517e-03 1.669920e-01 -5.721958e-03 -2.032956e-02 -
2.830933e-02
75%       4.816622e-01 5.692326e-01        7.143310e-01     7.655529e-01
5.908173e-01
max       4.716381e+00 3.460680e+00        2.388219e+00    2.362790e+00
3.794102e+00

                          x6                     x7                   x8                   x9
\x10
count     3.654800e+04 3.654800e+04  3.654800e+04  3.654800e+04
3.654800e+04
mean  -6.221234e-18 3.110617e-17          2.099667e-17 -1.088716e-17 -
2.391287e-17
std       1.000014e+00 1.000014e+00     1.000014e+00    1.000014e+00
1.000014e+00
min     -4.475132e+00 -5.692855e+00 -6.424467e+00 -2.087710e+00 -
2.056124e+00
25%     -9.049585e-01 -5.477824e-01 -5.606866e-01 -6.220080e-01 -
```

```
5.714242e-01
50%      2.542399e-02 -3.821154e-02 -5.732608e-02 -7.713820e-02
6.720843e-02
75%      7.707192e-01 5.267675e-01      5.489173e-01     4.815339e-01
5.775432e-01                              7.684405e+00  2.405039e+00
max      3.604507e+00 7.719614e+00
2.513861e+00

          ...                 x51              x52            x54             x55 \
count        ... 3.654800e+04 3.654800e+04  3.654800e+04   3.654800e+04
mean             ... 1.050611e-15 2.799555e-17    2.799555e-17    3.110617e-17
std          ... 1.000014e+00 1.000014e+00  1.000014e+00   1.000014e+00
min           ... -7.961188e+00 -9.783648e+00 -1.711381e+00 -3.323567e+00
25%             ... 8.006864e-02 -6.086076e-01 -1.212105e+00 -6.910733e-01
50%      ... 8.006864e-02 -3.061275e-01        1.092092e-01    5.019114e-03
75%      ... 8.006864e-02 4.960211e-01         9.933361e-01    6.769392e-01
max      ... 1.388752e+01 2.809207e+00         3.812686e+00    4.602692e+00

                      x56                x57              x58               x59
\x60
count    3.654800e+04 3.654800e+04  3.654800e+04  3.654800e+04
3.654800e+04
                                               3.897992e-17    6.843358e-17
mean  -2.488494e-17 6.221234e-18
2.021901e-17                                  1.000014e+00  1.000014e+00
std      1.000014e+00 1.000014e+00
1.000014e+00
min     -4.250150e+00 -5.791120e+00 -2.212102e+00 -5.295292e+00 -
3.547702e+00
25%     -6.194874e-01 -7.784508e-01 -5.463565e-01 -5.354201e-01 -
6.589753e-01
50%     -6.979482e-02 -1.032931e-01      1.369049e-01 -2.708045e-01 -
1.034212e-01
75%      7.237340e-01 8.152391e-01      5.870559e-01     8.523002e-02
8.324430e-01
max      2.898191e+00 2.996698e+00      6.279333e+00    8.282744e+00
3.195467e+00

                  y.1
count    3.654800e+04
mean    -4.043802e-17
std      1.000014e+00
min     -1.357163e-01
25%     -1.357163e-01
50%     -1.357163e-01
75%     -1.357163e-01
max      7.368309e+00

[8 rows x 60 columns]
```

```
#Importing PCA library #Telling PCA to retain 90% of useful features and then create new dimensions from sklearn.decomposition import PCA pca = PCA(0.90) X_pca = pca.fit_transform(X_scaled_df) X_pca.shape
```

```
(36548, 26)
```

```
## Each value will tell how much % of usefulness it contributes to the entire dataset
pca.explained_variance_ratio_
```

```
array([0.16405375, 0.1145583 , 0.08261669, 0.07065509, 0.05393095,
       0.04271356, 0.03966432, 0.03552791, 0.0340095 , 0.02983506,
       0.02528292, 0.02266675, 0.02099167, 0.01786314, 0.01699457,
       0.0161314 , 0.01576427, 0.01515727, 0.01428581, 0.01337401,
       0.01195731, 0.01145649, 0.00998132, 0.00936646, 0.00900109,
       0.00774108])
```

```
#To display how many dimensions we will feed now in our model
pca.n_components_
```

```
26
```

```
# Will now again do train test split but now with the X_pca
X_train_sc_pca, X_test_sc_pca, Y_train_sc_pca, Y_test_sc_pca = train_test_split(X_pca, Y_res, test_size = 0.2, random_state=123)
```

```
X_train_sc_pca
```

```
array([[-1.85904913, -3.43825123, 1.57555331, ..., -0.09662878,
        -0.22846121, 0.61013392],
       [ 3.3451392 , 1.9950538 , 1.76736295, ..., -0.11005107,
        0.23110504, -0.12187116],
       [ 2.52467669, -1.31695532, 1.13665655, ..., -0.25767517,
        0.50666258, -0.27040914],
       ...,
       [-2.90046972, 0.88216013, 3.47542271, ..., -1.00079079,
        -1.51671085, -0.5722476 ],
       [-0.16058809, -5.23636177, -1.71785398, ..., 0.31846568,
        1.30601642, 2.4340082 ],
       [-3.42476415, -4.85333061, 0.51266168, ..., -0.45545897,
        0.97360167, -0.06335646]])
```

Model Selection

```
#Since it is a classification problem, We will use the following models for it and will compare their accuracy on test data against each other.
```

```
#1. Logistic Regression from sklearn.linear_model import
LogisticRegression model1 = LogisticRegression()
model1.fit(X_train_sc_pca,Y_train_sc_pca)
model1.score(X_test_sc_pca,Y_test_sc_pca)
```

0.8383036935704514

```
#2. Decision Tree
from sklearn import tree
model2 = tree.DecisionTreeClassifier()
model2.fit(X_train_sc_pca,Y_train_sc_pca)
model2.score(X_test_sc_pca,Y_test_sc_pca)
```

0.9800273597811218

```
#3. KNN
from sklearn.neighbors import KNeighborsClassifier
model3 = KNeighborsClassifier(n_neighbors=3)
model3.fit(X_train_sc_pca,Y_train_sc_pca)
model3.score(X_test_sc_pca,Y_test_sc_pca)
```

0.9941176470588236

```
#4. Random Forest
from sklearn.ensemble import RandomForestClassifier
model4 = RandomForestClassifier(n_estimators=30)
model4.fit(X_train_sc_pca,Y_train_sc_pca)
model4.score(X_test_sc_pca,Y_test_sc_pca)
```

0.9991792065663475

Model Validation

To check where model performs good and where it performs bad

1. For Logistic Regression

```
Y_predicted_md1 = model1.predict(X_test_sc_pca)
Y_predicted_md1
```

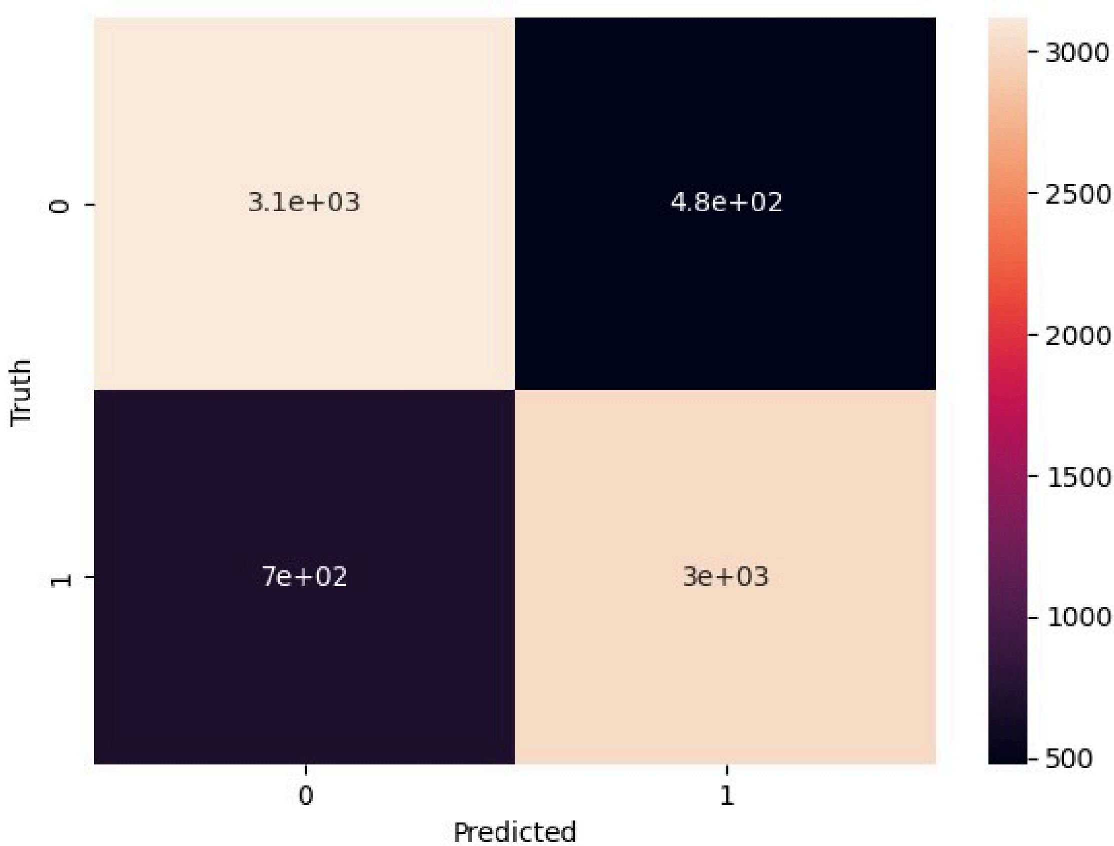
array([1, 0, 0, ..., 1, 0, 1], dtype=int64)

```
from sklearn.metrics import confusion_matrix
cm1 = confusion_matrix(Y_test_sc_pca,Y_predicted_md1)
cm1
```

```
array([[3118, 479],
       [ 703, 3010]], dtype=int64)
```

```
plt.figure(figsize = (7,5))
sns.heatmap(cm1,annot = True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

Text(58.222222222222214, 0.5, 'Truth')




```
from sklearn.metrics import classification_report
print(classification_report(Y_test_sc_pca,Y_predicted_md1))
```

	precision	recall	f1-score	support
0	0.82	0.87	0.84	3597
1	0.86	0.81	0.84	3713
accuracy			0.84	7310
macro avg	0.84	0.84	0.84	7310
weighted avg	0.84	0.84	0.84	7310

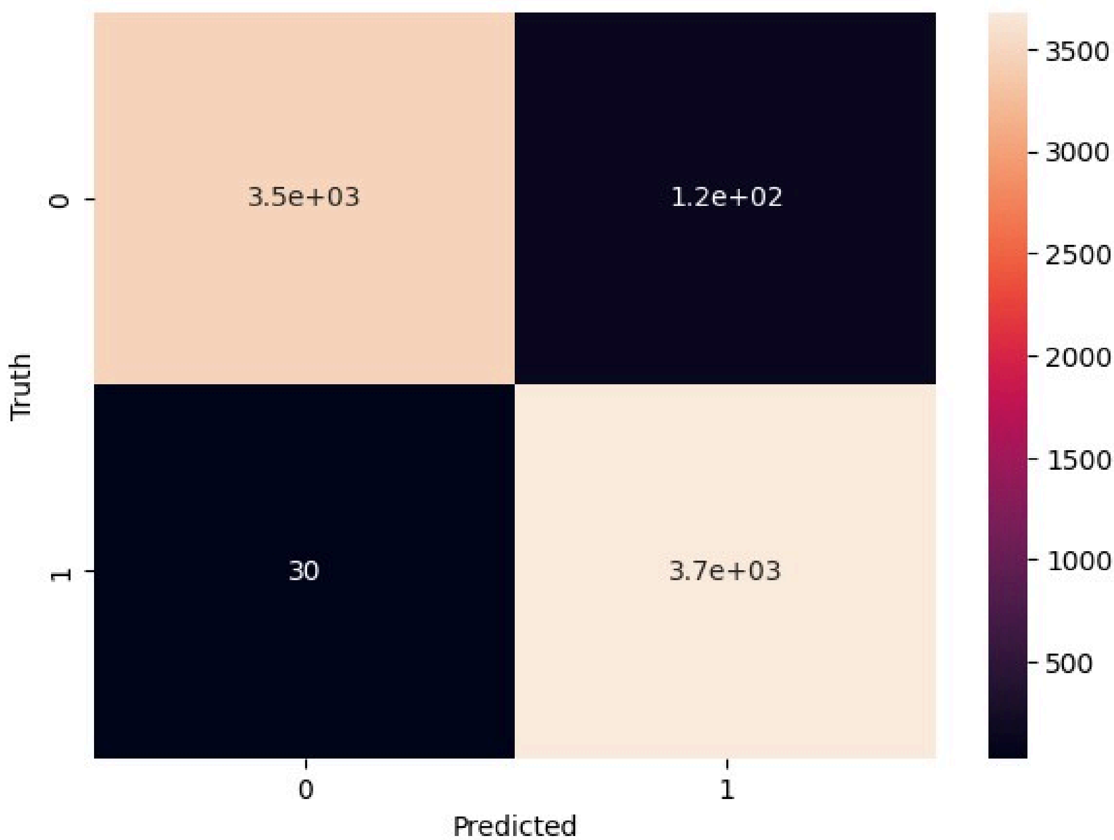
2. For Decision Tree

```
Y_predicted_md2 = model2.predict(X_test_sc_pca)
Y_predicted_md2
cm2 = confusion_matrix(Y_test_sc_pca,Y_predicted_md2)
cm2
```

```
array([[3481, 116],
       [ 30, 3683]], dtype=int64)
```

```
plt.figure(figsize = (7,5))
sns.heatmap(cm2,annot = True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

```
Text(58.222222222222214, 0.5, 'Truth')
```




```
from sklearn.metrics import classification_report
print(classification_report(Y_test_sc_pca,Y_predicted_md2))
```

	precision	recall	f1-score	support
0	0.99	0.97	0.98	3597
1	0.97	0.99	0.98	3713

```
 accuracy   0.98   7310     macro   avg   0.98   0.98   0.98   7310
weighted avg 0.98 0.98 0.98 7310
```

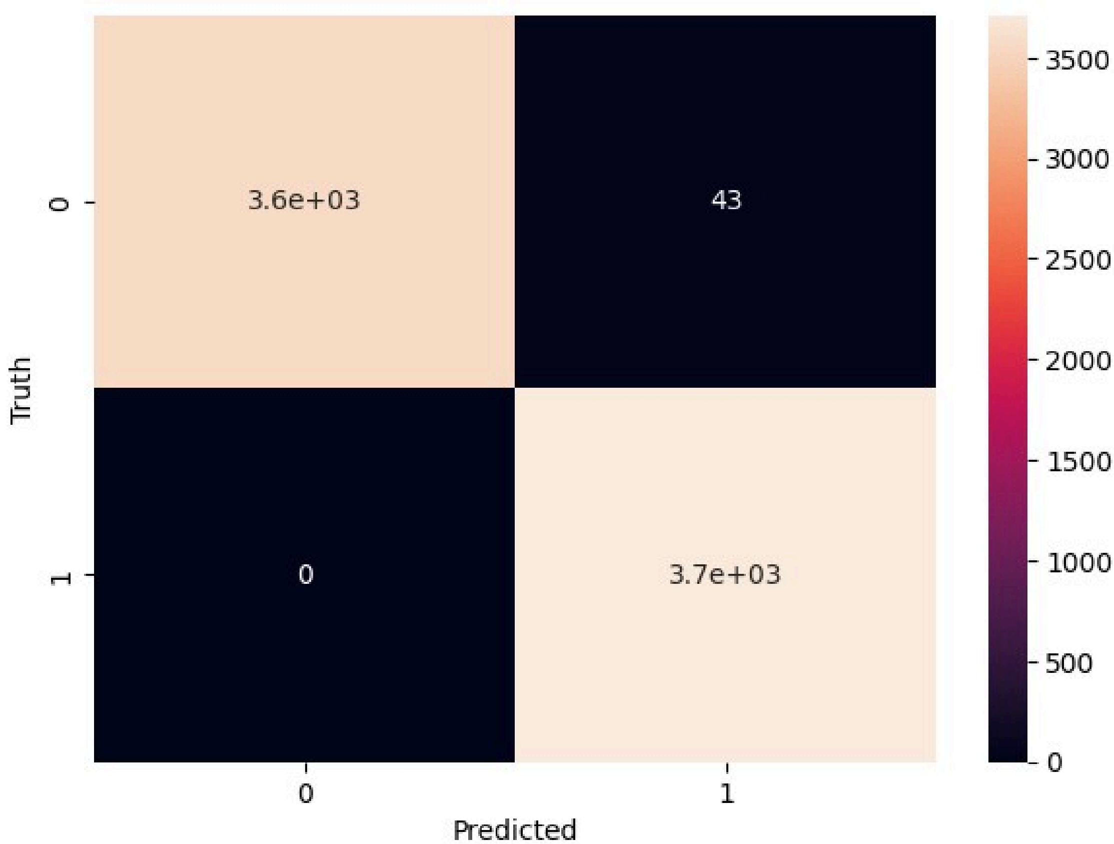
3. For KNN

```
Y_predicted_md3 = model3.predict(X_test_sc_pca)
Y_predicted_md3
cm3 = confusion_matrix(Y_test_sc_pca,Y_predicted_md3)
cm3
```

```
array([[3554,     43],
       [   0, 3713]], dtype=int64)
```

```
plt.figure(figsize = (7,5))
sns.heatmap(cm3,annot = True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

```
Text(58.222222222222214, 0.5, 'Truth')
```



```
from sklearn.metrics import classification_report
print(classification_report(Y_test_sc_pca,Y_predicted_md3))
```

```
              precision    recall  f1-score   support

           0       1.00      0.99      0.99      3597
           1       0.99      1.00      0.99      3713

    accuracy                           0.99      7310
   macro avg       0.99      0.99      0.99      7310
weighted avg       0.99      0.99      0.99      7310
```

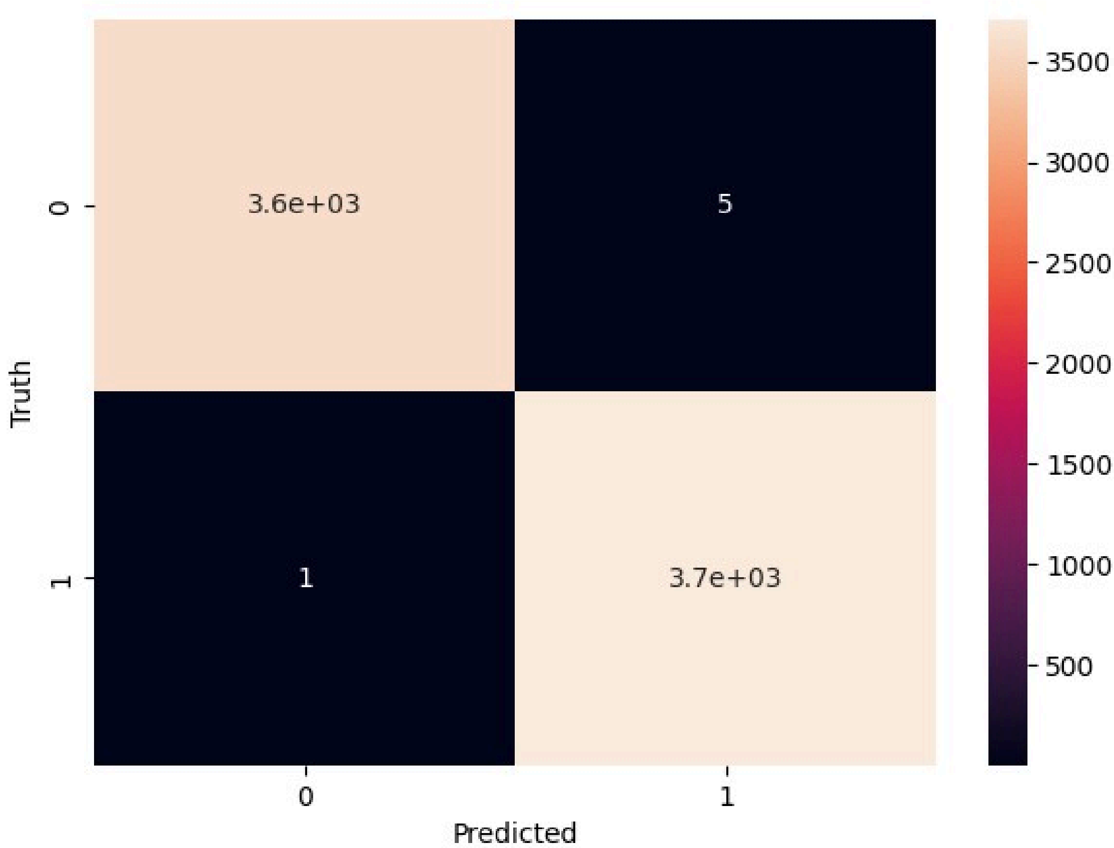
4. For Random Forest

```
Y_predicted_md4 = model4.predict(X_test_sc_pca)
Y_predicted_md4
cm4 = confusion_matrix(Y_test_sc_pca,Y_predicted_md4)
cm4
```

```
array([[3592,    5],
       [   1, 3712]], dtype=int64)
```

```
plt.figure(figsize = (7,5))
sns.heatmap(cm4,annot = True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

```
Text(58.222222222222214, 0.5, 'Truth')
```




```
from sklearn.metrics import classification_report
print(classification_report(Y_test_sc_pca,Y_predicted_md4))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	3597
1	1.00	1.00	1.00	3713
accuracy			1.00	7310
macro avg	1.00	1.00	1.00	7310
weighted avg	1.00	1.00	1.00	7310

Conclusion

Considering the Classification report from the above four models, we found Random Forest Algorithm - accuracy, precision, f1-score is acheieving 99.99 % accuracy on test data set. With a dataset comprising over 18,000 rows and utilizing binary labels for anomaly identification, the model has demonstrated exceptional performance, achieving an outstanding accuracy of 99.99%. This high level of accuracy underscores the model's reliability in detecting anomalies and predicting machine breakdowns, which can significantly reduce downtime, minimize risks, and optimize maintenance schedules across industries.