Out[1062]:

	Discrete Features	No of Unique Values	Unique Values
0	month	4	[7, 6, 9, 8]
1	year	1	[2012]
2	Region	2	[0.0, 1.0]
3	Classes_notfire	2	[1, 0]

In [106…

```
categorical_features = [fea for fea in dataset.columns if dataset[fea].dtype == 'O
print("Categorical feature Count {}".format(len(categorical_features)))
print(categorical_features)
data = dataset.copy()
pd.DataFrame(list(zip(categorical_features,[len(data[feature].unique()) for featur
```

```
Categorical feature Count 0
[]
```

Out[1063]:

	Categorical Features	No. of Categories	Categories

Tranformation of Data

Shapiro Wick Test

- **The Shapiro-Wilk test is a way to tell if a random sample comes from a normal distribution.**
- Ho : Data is normally distributed
- H1 : Data is not normally distributed

In [106…

```
from scipy.stats import shapiro
shapiro_wick_test = []
for column in numerical_features:
    dataToTest = data[column]
    stat,p = shapiro(dataToTest)
    if p > 0.05:
        shapiro_wick_test.append("Normally Distributed")
    else:
        shapiro_wick_test.append("Not Normally Distributed")
result = pd.DataFrame(data=[numerical_features, shapiro_wick_test]).T
result.columns = ['Column Name', 'Shapiro Hypothesis Result']
result
```

Out[1064]:

	Column Name	Shapiro Hypothesis Result
0	day	Not Normally Distributed
1	month	Not Normally Distributed
2	year	Normally Distributed
3	Temperature	Normally Distributed
4	RH	Not Normally Distributed
5	Rain	Not Normally Distributed
6	Region	Not Normally Distributed
7	WsiqrO	Not Normally Distributed
8	RainextO	Not Normally Distributed
9	RainiqrO	Not Normally Distributed
10	FFMCiqrO	Not Normally Distributed
11	DMCiqrO	Not Normally Distributed
12	DCiqrO	Not Normally Distributed
13	ISIiqrO	Not Normally Distributed
14	BUIiqrO	Not Normally Distributed
15	FWIiqrO	Not Normally Distributed
16	Classes_notfire	Not Normally Distributed

K^2 Normality Test

- **Test aims to establish whether or not the given sample comes from a normally distributed population. Test is based on transformations of the sample kurtosis and skewness**
- Ho : Data is normally distributed
- H1 : Data is not normally distributed

In [106…

```
from scipy.stats import normaltest
normaltest_test = []
for column in numerical_features:
    dataToTest = data[column]
    stat,p = normaltest(dataToTest)
    if p > 0.05:
        normaltest_test.append("Normally Distributed")
    else:
        normaltest_test.append("Not Normally Distributed")
result = pd.DataFrame(data=[numerical_features, normaltest_test]).T
result.columns = ['Column Name', 'normaltest Hypothesis Result']
result
```

Out[1065]:

	Column Name	normaltest Hypothesis Result
0	day	Not Normally Distributed
1	month	Not Normally Distributed
2	year	Not Normally Distributed
3	Temperature	Normally Distributed
4	RH	Normally Distributed
5	Rain	Not Normally Distributed
6	Region	Not Normally Distributed
7	WsiqrO	Normally Distributed
8	RainextO	Not Normally Distributed
9	RainiqrO	Not Normally Distributed
10	FFMCiqrO	Not Normally Distributed
11	DMCiqrO	Not Normally Distributed
12	DCiqrO	Not Normally Distributed
13	ISIiqrO	Not Normally Distributed
14	BUIiqrO	Not Normally Distributed
15	FWIiqrO	Not Normally Distributed
16	Classes_notfire	Not Normally Distributed

In [106…

```
#### If you want to check whether feature is guassian or normal distributed
#### Q-Q plot
def plot_data(df,feature):
    plt.figure(figsize=(14,6))
    plt.subplot(1,2,1)
    df[feature].hist()
    plt.subplot(1,2,2)
    stat.probplot(df[feature],dist='norm',plot=pylab)
    plt.show()


def plot_qq_plot(df,column):
    plt.figure(figsize=(40,7))
    plt.subplot(131)
    sns.distplot(df[column])
    plt.title("{} PDF".format(column))
    plt.subplot(132)
    stat.probplot(df[column], dist="norm", plot=plt)
    plt.title('{} QQ Plot'.format(column))
    plt.subplot(133)
    sns.scatterplot(x=df[column], y = data['Temperature'])
    plt.title(column)
    plt.show()


def transform_data(df,feature):
    plt.figure(figsize=(10,6))

    if 0 in data[feature].unique():  #
        pass
    else:
```

```
        df[feature +'_log']=numpy.log(df[feature])
        print('----------------- Log Transformation -------------------')
        plot_qq_plot(df,feature +'_log')

        df[feature +'_reciprocal']=1/df[feature]
        print('----------------- Reciprocal Transformation -------------------')
        plot_qq_plot(df,feature +'_reciprocal')
        df[feature +'_square']=df[feature]**(1/2)
        print('----------------- Square Transformation -------------------')
        plot_qq_plot(df,feature +'_square')
        df[feature +'_exponential']=df[feature]**(1/1.2)
        print('----------------- Exponential Transformation -------------------')
        plot_qq_plot(df,feature +'_exponential')
        df[feature +'_Boxcox'],parameters=stat.boxcox(df[feature])
        print('----------------- Boxcox Transformation -------------------')
        plot_qq_plot(df,feature +'_Boxcox')
        plt.show()



def transform_data_nl(df,feature):
    plt.figure(figsize=(10,6))

    df[feature +'_square']=df[feature]**(1/2)
    print('----------------- Square Transformation -------------------')
    plot_qq_plot(df,feature +'_square')

    df[feature +'_exponential']=df[feature]**(1/1.2)
    print('----------------- Exponential Transformation -------------------')
    plot_qq_plot(df,feature +'_exponential')

    df[feature +'_Boxcox'],parameters=stat.boxcox(df[feature])
    print('----------------- Boxcox Transformation -------------------')
    plot_qq_plot(df,feature +'_Boxcox')
    plt.show()
```

In [106…

```
data.corr()['Temperature'].sort_values()
```

Out[1067]:

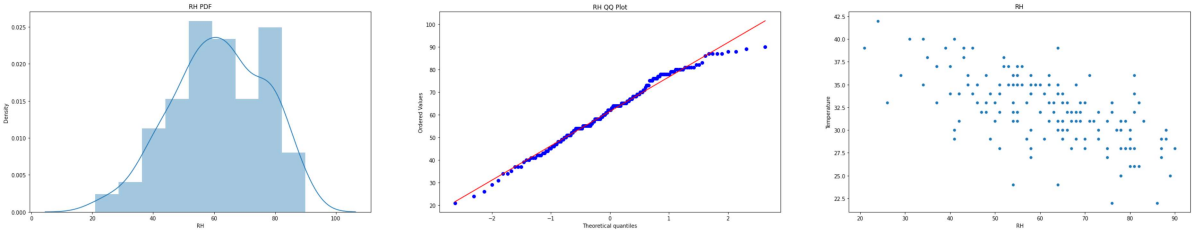
```
RH               -0.640282
Classes_notfire  -0.503021
RainextO         -0.398826
RainiqrO         -0.370476
Rain             -0.355614
WsiqrO           -0.323498
month            -0.053159
day               0.069888
Region            0.265069
DCiqrO            0.381636
BUIiqrO           0.484925
DMCiqrO           0.509677
FWIiqrO           0.577653
ISIiqrO           0.616460
FFMCiqrO          0.676236
Temperature       1.000000
year                   NaN
Name: Temperature, dtype: float64
```

RH Tranformation

In [106…

```
import scipy.stats as stat
print('----------------- Original Distribution -------------------')
plot_qq_plot(data, 'RH')
```

```
---------------- Original Distribution -------------------
```



RH has Normal distribution

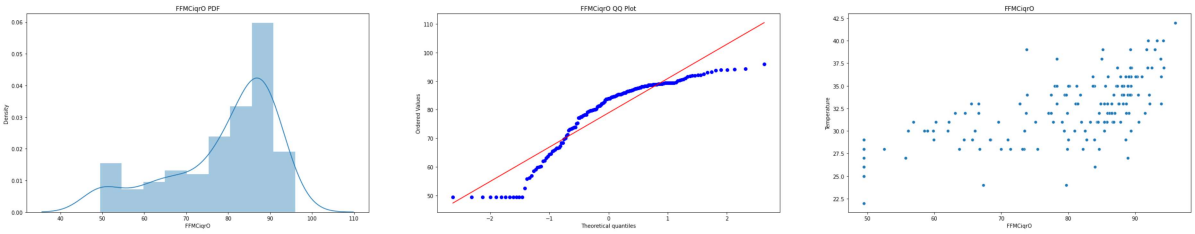
FFMCiqrO Transformation

In [106…
```
data.columns
```

Out[1069]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Rain', 'Region', 'WsiqrO',
       'RainextO', 'RainiqrO', 'FFMCiqrO', 'DMCiqrO', 'DCiqrO', 'ISIiqrO',
       'BUIiqrO', 'FWIiqrO', 'Classes_notfire'],
      dtype='object')
```

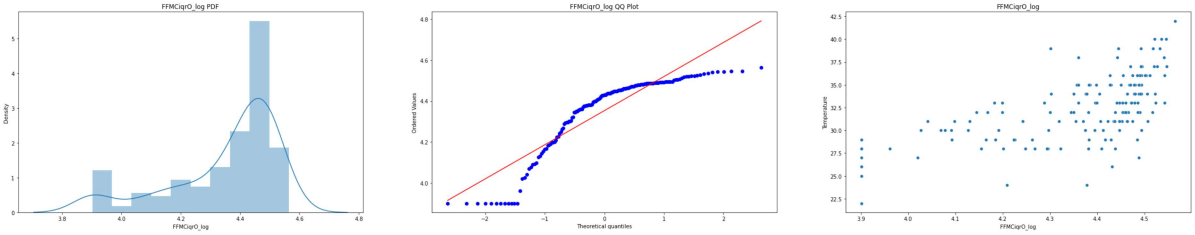
In [107…
```
import scipy.stats as stat
print('---------------- Original Distribution -------------------')
plot_qq_plot(data, 'FFMCiqrO')
```

```
---------------- Original Distribution -------------------
```

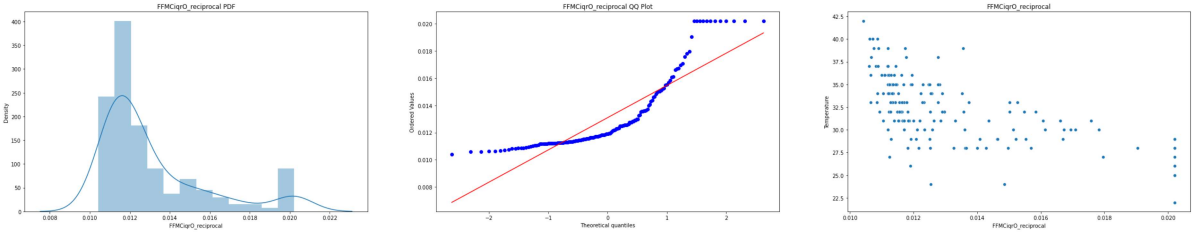


In [107…
```
import pylab
import scipy.stats as stat
transform_data(data,'FFMCiqrO')
```

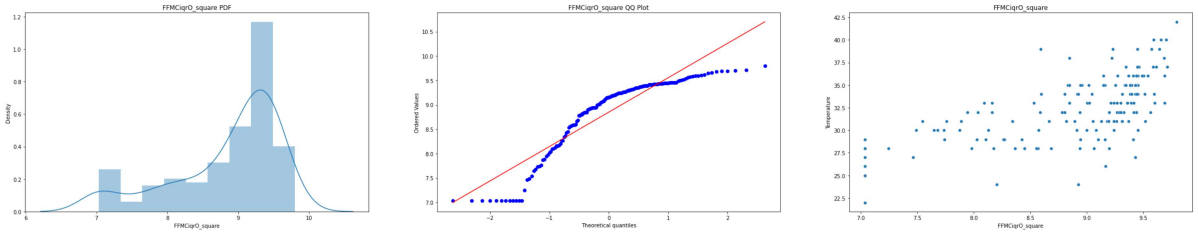
```
---------------- Log Transformation -------------------
<Figure size 720x432 with 0 Axes>
```



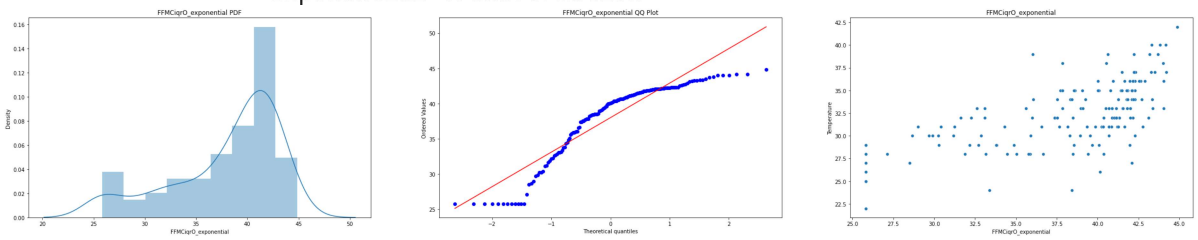

```
---------------- Reciprocal Transformation -------------------
```



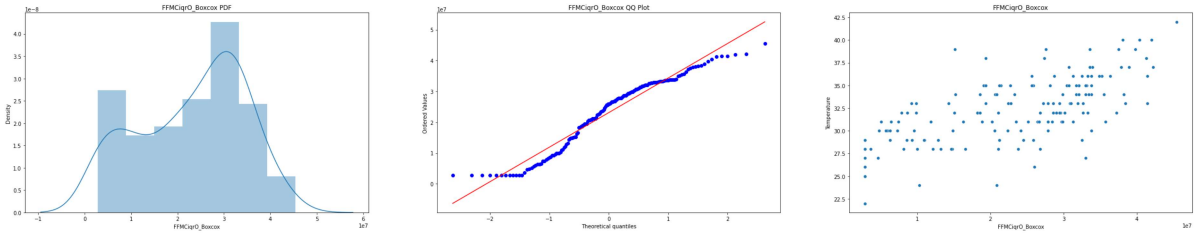

```
---------------- Square Transformation -------------------
```



---------- Exponential Transformation ----------



---------- Boxcox Transformation ----------



In [107…
```
data.columns
```

Out[1072]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Rain', 'Region', 'WsiqrO',
       'RainextO', 'RainiqrO', 'FFMCiqrO', 'DMCiqrO', 'DCiqrO', 'ISIiqrO',
       'BUIiqrO', 'FWIiqrO', 'Classes_notfire', 'FFMCiqrO_log',
       'FFMCiqrO_reciprocal', 'FFMCiqrO_square', 'FFMCiqrO_exponential',
       'FFMCiqrO_Boxcox'],
      dtype='object')
```

In [107…
```
data.drop(['FFMCiqrO_log',
       'FFMCiqrO_reciprocal', 'FFMCiqrO_square', 'FFMCiqrO_exponential','FFMCiqrO'
data.columns
```

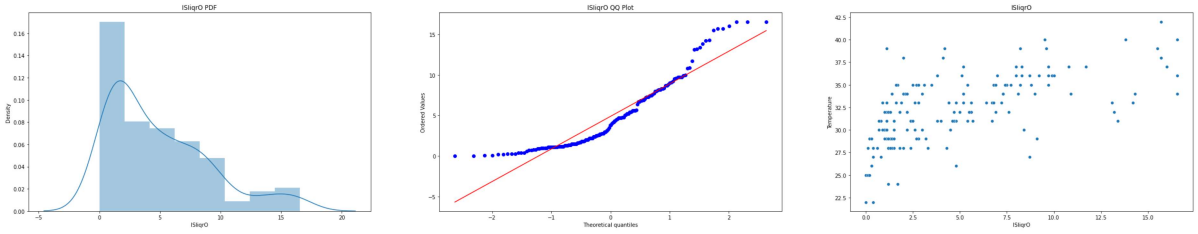
Out[1073]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Rain', 'Region', 'WsiqrO',
       'RainextO', 'RainiqrO', 'DMCiqrO', 'DCiqrO', 'ISIiqrO', 'BUIiqrO',
       'FWIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox'],
      dtype='object')
```

FFMC boxcox perform well.

ISIiqrO TRansformation

In [107…
```
import scipy.stats as stat
print('---------- Original Distribution ----------')
plot_qq_plot(data, 'ISIiqrO')
```

---------- Original Distribution ----------



In [916…
```
#data[data['ISIiqrO']<=0]
```

Out[916]:

	day	month	year	Temperature	RH	Rain	Region	WsiqrO	RainextO	RainiqrO	DMCiqrO
2	4	6	2012	25	89	2.5	0.0	13.0	1.9	1.1875	1.3
53	2	9	2012	22	86	10.1	0.0	15.0	1.9	1.1875	0.7

In [917…
```
#data.ISIiqrO.unique()
```

Out[917]:
```
array([ 2.6 ,  1.  ,  0.  ,  0.2 ,  5.7 ,  1.1 ,  6.8 ,  8.7 ,  2.  ,
        1.8 ,  0.9 ,  8.3 ,  8.2 ,  4.8 ,  7.5 ,  8.8 ,  4.1 ,  4.7 ,
        2.2 ,  1.5 ,  1.2 ,  0.3 ,  0.4 ,  7.6 ,  5.2 ,  3.  ,  3.2 ,
        3.8 ,  4.5 , 16.55,  1.7 ,  6.7 ,  5.1 ,  4.3 ,  1.4 ,  9.6 ,
        4.  ,  5.4 ,  1.6 , 11.7 ,  2.1 ,  2.4 ,  1.3 ,  0.1 ,  6.9 ,
        2.8 ,  7.7 ,  0.8 ,  5.6 ,  1.9 ,  9.7 ,  0.7 ,  8.4 ,  6.4 ,
        9.  , 14.3 ,  3.5 ,  0.6 , 13.1 ,  7.3 , 15.7 , 13.4 ,  2.7 ,
        4.2 , 13.8 , 16.  ,  3.1 ,  3.3 ,  9.1 ,  5.  ,  8.  ,  7.  ,
        7.2 ,  9.9 ,  9.5 , 10.  ,  2.5 , 10.9 ,  9.2 , 14.2 , 15.5 ,
        7.4 ,  4.4 ,  8.1 , 10.8 ,  5.5 ,  4.6 , 13.2 ])
```

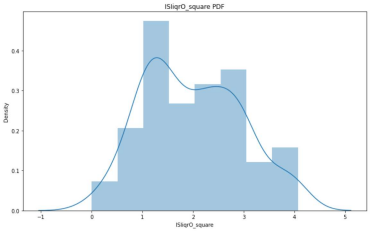
In [107…
```
#data[data.ISIiqrO<=0]=0.1
data[data['ISIiqrO']<=0]
```

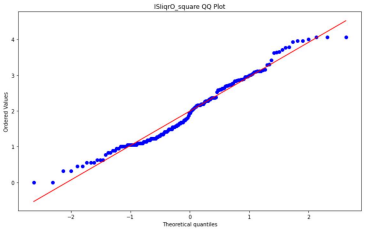
Out[1075]:

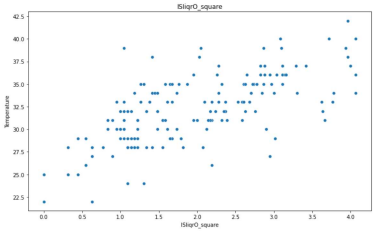
	day	month	year	Temperature	RH	Rain	Region	WsiqrO	RainextO	RainiqrO	DMCiqrO
2	4	6	2012	25	89	2.5	0.0	13.0	1.9	1.1875	1.3
53	2	9	2012	22	86	10.1	0.0	15.0	1.9	1.1875	0.7

In [107…
```
import pylab
import scipy.stats as stat
transform_data_nl(data,'ISIiqrO')
```

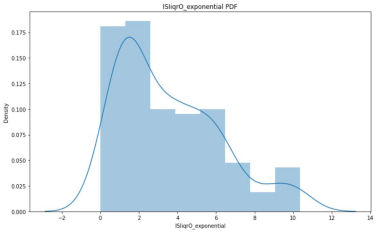
```
----------------- Square Transformation -------------------
<Figure size 720x432 with 0 Axes>
```

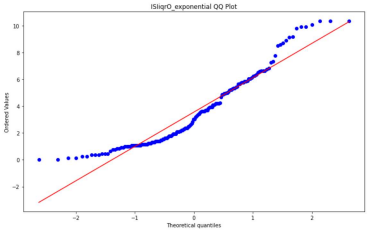


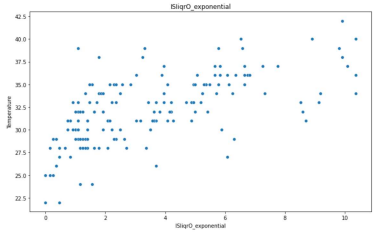





```
----------------- Exponential Transformation -------------------
```







In [109…
```
data.drop(['FWIiqrO',
       'FWIiqrO_exponential'],axis=1,inplace=True)
data.columns
```

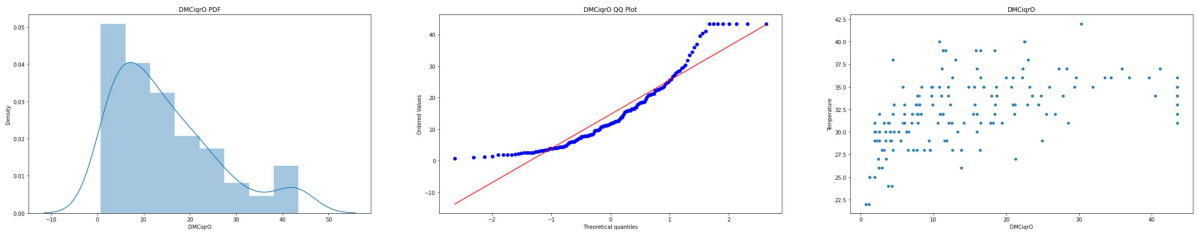
Out[1094]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DMCiqrO', 'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square'],
      dtype='object')
```

In [ ]:

DMC

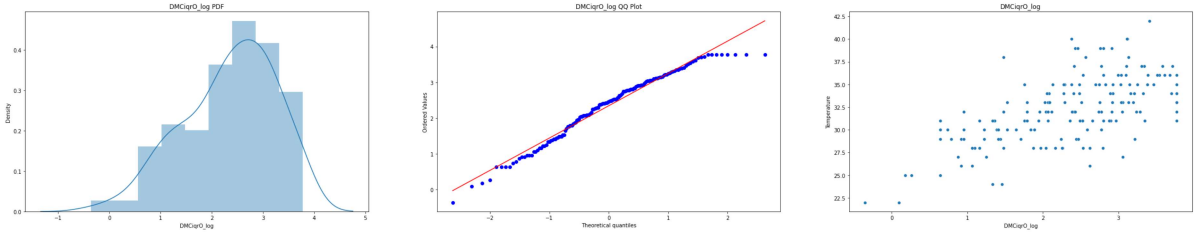
In [109…
```
import scipy.stats as stat
print('----------------- Original Distribution -------------------')
plot_qq_plot(data, 'DMCiqrO')
```

```
----------------- Original Distribution -------------------
```

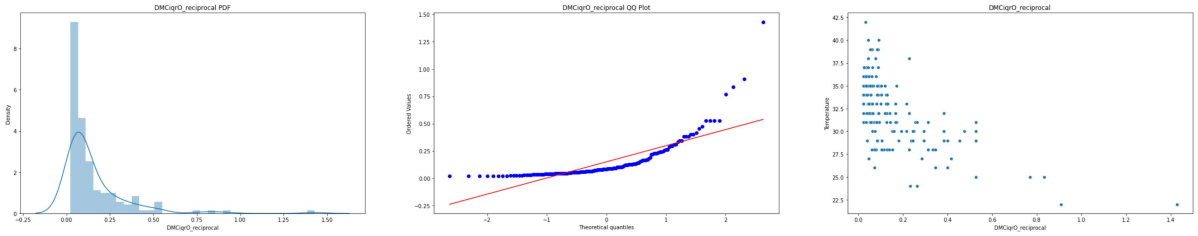


In [109…
```
import pylab
import scipy.stats as stat
transform_data(data,'DMCiqrO')
```

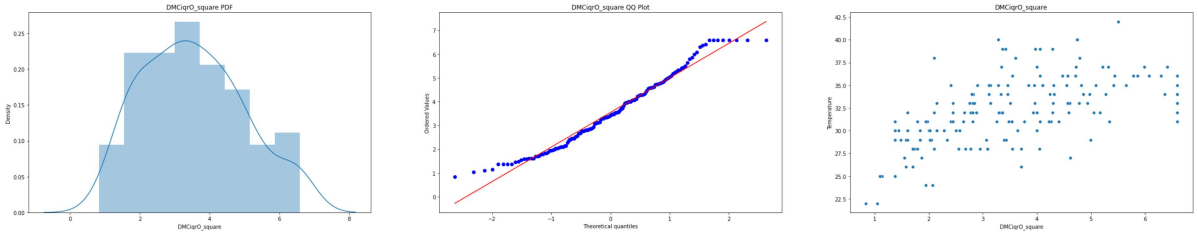
```
----------------- Log Transformation -------------------
<Figure size 720x432 with 0 Axes>
```



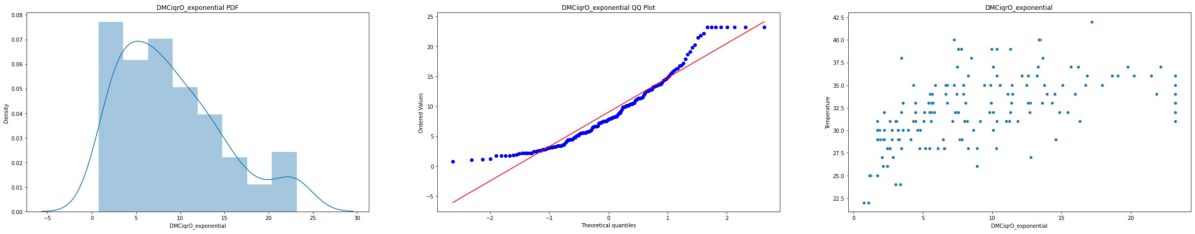

```
----------------- Reciprocal Transformation -------------------
```



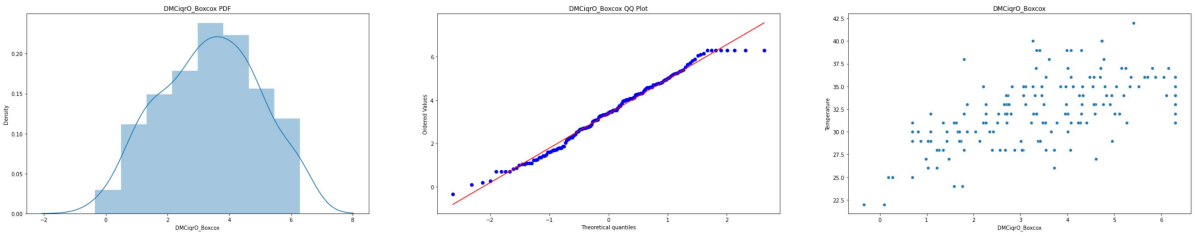

```
----------------- Square Transformation -------------------
```




```
----------------- Exponential Transformation -------------------
```




```
---------------- Boxcox Transformation -------------------
```



In [109…
```
data.columns
```

Out[1097]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DMCiqrO', 'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_log', 'DMCiqrO_reciprocal',
       'DMCiqrO_square', 'DMCiqrO_exponential', 'DMCiqrO_Boxcox'],
      dtype='object')
```

In [109…
```
data.drop(['DMCiqrO_log', 'DMCiqrO_reciprocal', 'DMCiqrO_exponential','DMCiqrO'],a
data.columns
```

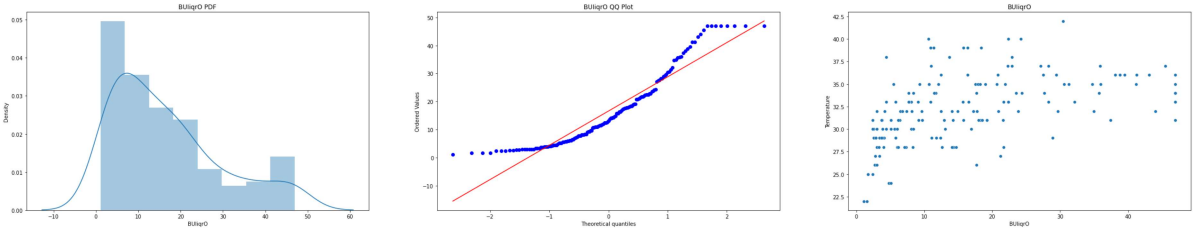
Out[1098]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox'],
      dtype='object')
```

In [ ]:

BUI

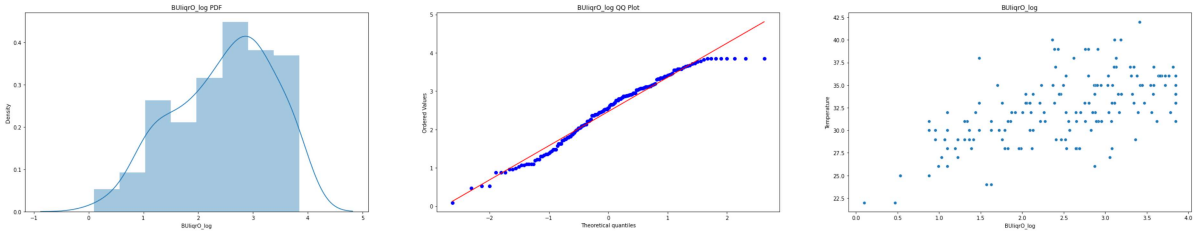
In [109…
```
import scipy.stats as stat
print('----------------- Original Distribution -------------------')
plot_qq_plot(data, 'BUIiqrO')
```

```
----------------- Original Distribution -------------------
```

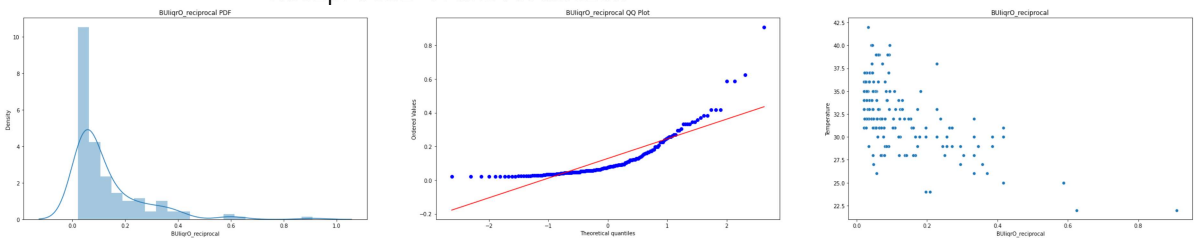


In [110…
```
import pylab
import scipy.stats as stat
transform_data(data,'BUIiqrO')
```

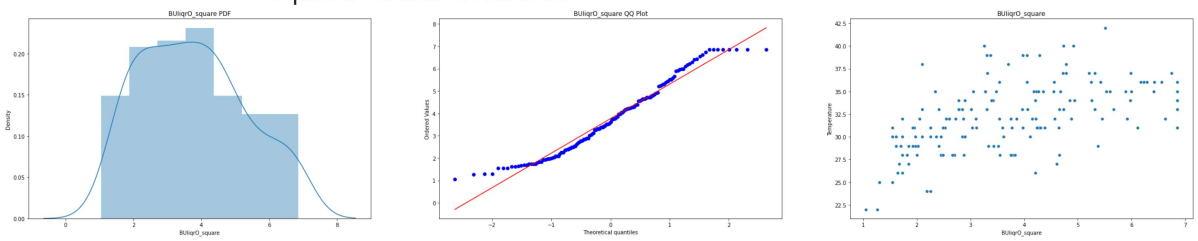
```
----------------- Log Transformation -------------------
<Figure size 720x432 with 0 Axes>
```



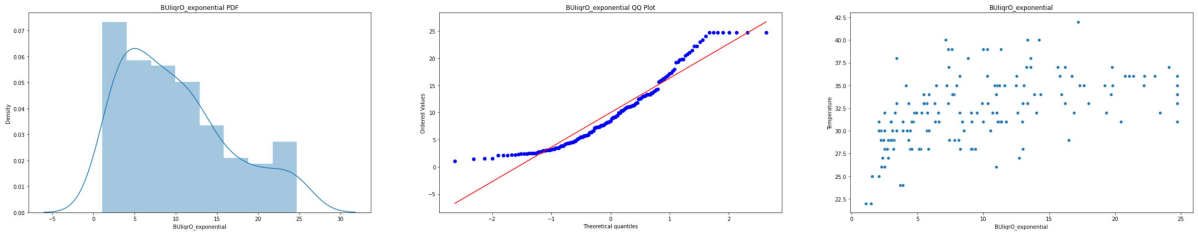
---------- Reciprocal Transformation ----------



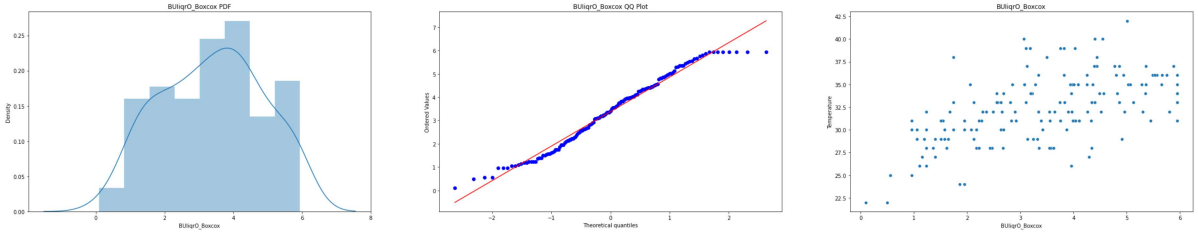
---------- Square Transformation ----------



---------- Exponential Transformation ----------



---------- Boxcox Transformation ----------



In [110…
```
data.columns
```

Out[1101]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox',
       'BUIiqrO_log', 'BUIiqrO_reciprocal', 'BUIiqrO_square',
       'BUIiqrO_exponential', 'BUIiqrO_Boxcox'],
      dtype='object')
```

In [110…
```
data.drop(['BUIiqrO_exponential','BUIiqrO_log', 'BUIiqrO_reciprocal'],axis=1,inpla
data.columns
```

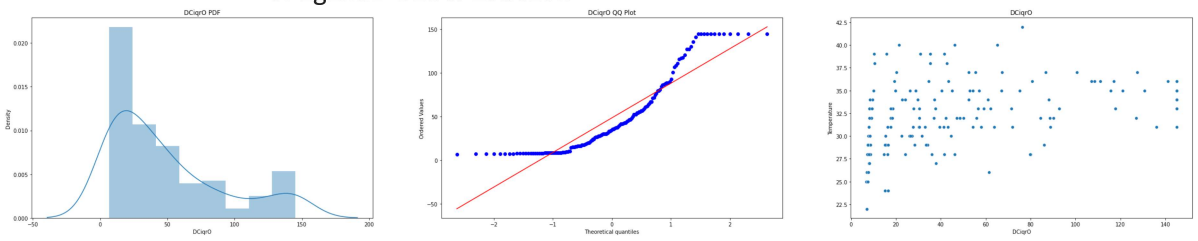
Out[1102]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox',
       'BUIiqrO_square', 'BUIiqrO_Boxcox'],
      dtype='object')
```

In [ ]:

DC

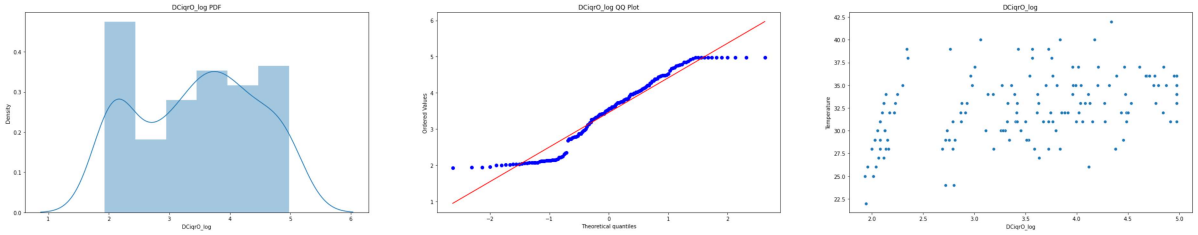
In [110…
```
import scipy.stats as stat
print('----------------- Original Distribution -------------------')
plot_qq_plot(data, 'DCiqrO')
```

```
----------------- Original Distribution -------------------
```

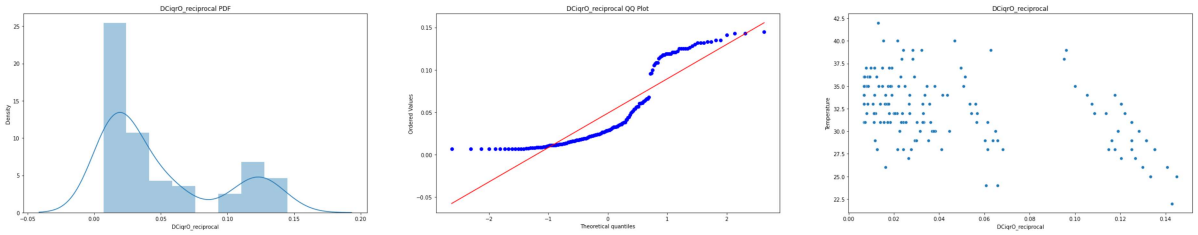


In [110…
```
import pylab
import scipy.stats as stat
transform_data(data,'DCiqrO')
```

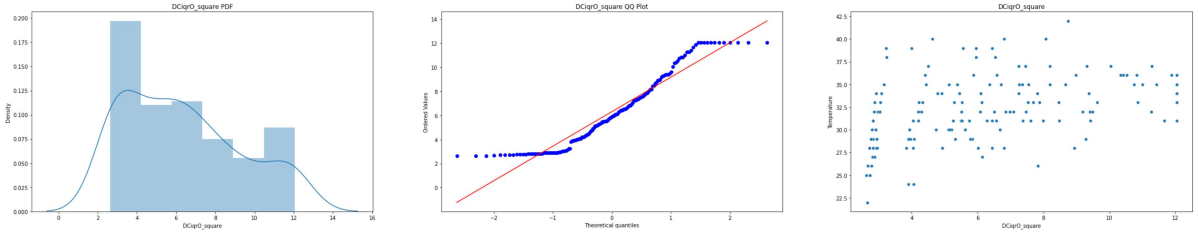
```
----------------- Log Transformation -------------------
<Figure size 720x432 with 0 Axes>
```



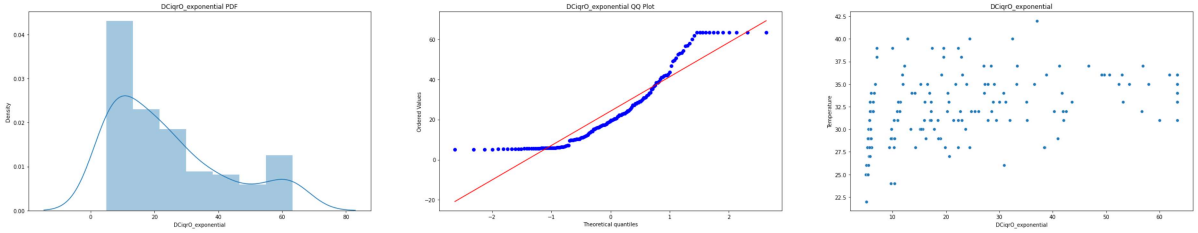

```
----------------- Reciprocal Transformation -------------------
```



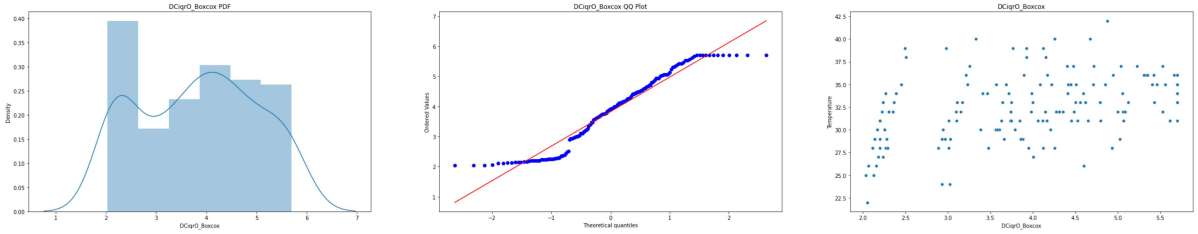

```
----------------- Square Transformation -------------------
```




```
----------------- Exponential Transformation -------------------
```




```
----------------- Boxcox Transformation -------------------
```



In [110…
```
data.columns
```

Out[1105]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox',
       'BUIiqrO_square', 'BUIiqrO_Boxcox', 'DCiqrO_log', 'DCiqrO_reciprocal',
       'DCiqrO_square', 'DCiqrO_exponential', 'DCiqrO_Boxcox'],
      dtype='object')
```

In [110…
```
data.drop(['DCiqrO_exponential','DCiqrO_log', 'DCiqrO_reciprocal'],axis=1,inplace=
data.columns
```

Out[1106]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox',
       'BUIiqrO_square', 'BUIiqrO_Boxcox', 'DCiqrO_square', 'DCiqrO_Boxcox'],
      dtype='object')
```

In [110…
```
data.month.unique()
```

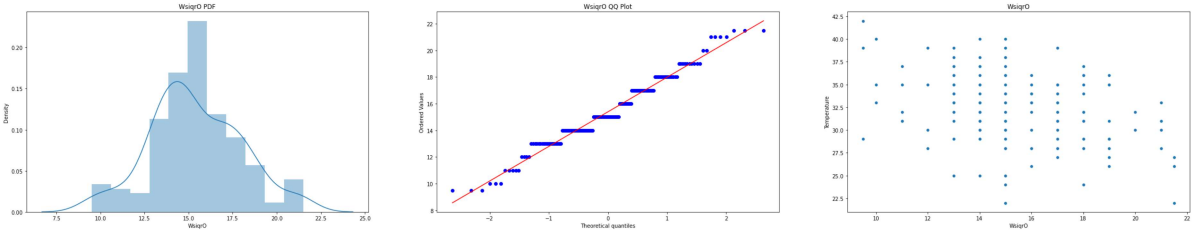
Out[1107]:
```
array([7, 6, 9, 8], dtype=int64)
```

In [ ]:

WsiqrO

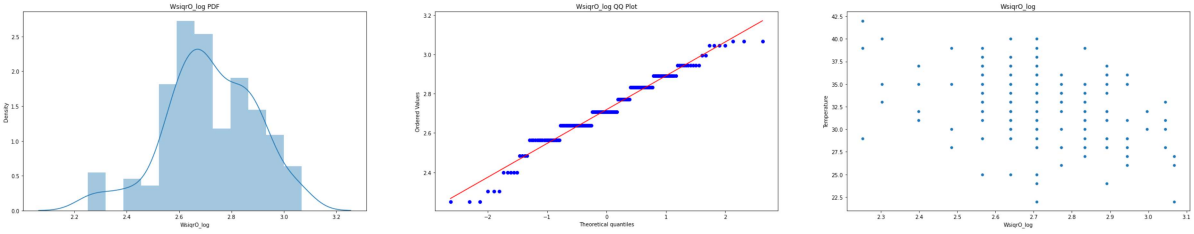
In [110…
```
import scipy.stats as stat
print('----------------- Original Distribution -------------------')
plot_qq_plot(data, 'WsiqrO')
```

```
----------------- Original Distribution -------------------
```

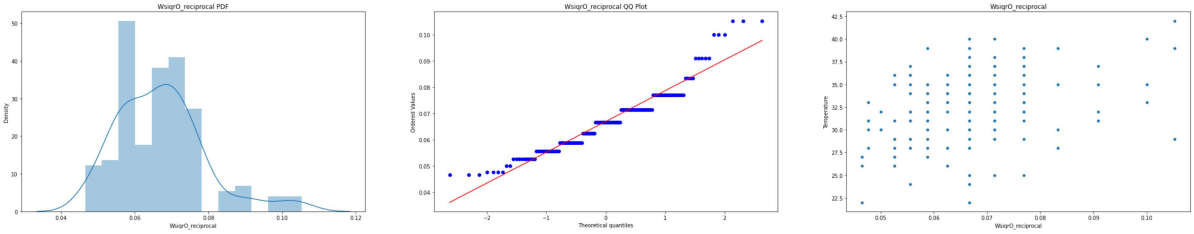


In [110…
```
import pylab
import scipy.stats as stat
transform_data(data,'WsiqrO')
```

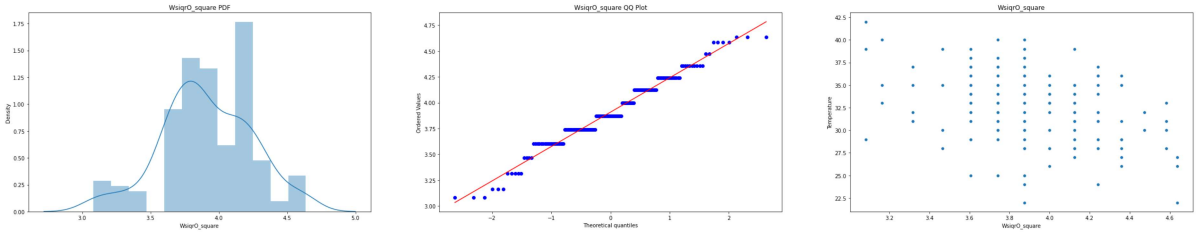
```
----------------- Log Transformation -------------------
<Figure size 720x432 with 0 Axes>
```



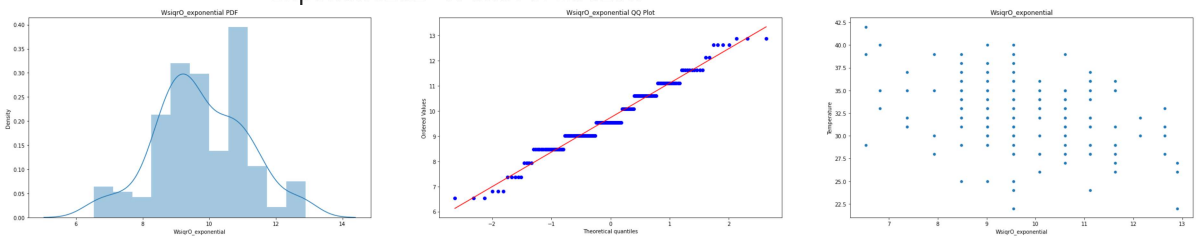

```
----------------- Reciprocal Transformation -------------------
```



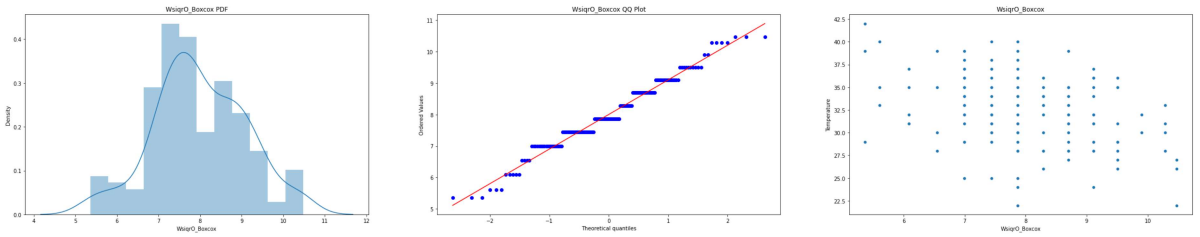

```
----------------- Square Transformation -------------------
```



---------- Exponential Transformation ----------



---------- Boxcox Transformation ----------



In [111…
```
data.columns
```

Out[1110]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox',
       'BUIiqrO_square', 'BUIiqrO_Boxcox', 'DCiqrO_square', 'DCiqrO_Boxcox',
       'WsiqrO_log', 'WsiqrO_reciprocal', 'WsiqrO_square',
       'WsiqrO_exponential', 'WsiqrO_Boxcox'],
      dtype='object')
```

In [111…
```
data.drop(['WsiqrO_log', 'WsiqrO_reciprocal', 'WsiqrO_square',
       'WsiqrO_exponential'],axis=1,inplace=True)
data.columns
```

Out[1111]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region', 'WsiqrO',
       'DCiqrO', 'BUIiqrO', 'Classes_notfire', 'FFMCiqrO_Boxcox',
       'ISIiqrO_square', 'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox',
       'BUIiqrO_square', 'BUIiqrO_Boxcox', 'DCiqrO_square', 'DCiqrO_Boxcox',
       'WsiqrO_Boxcox'],
      dtype='object')
```

In [111…
```
data.head()
```

Out[1112]:

	day	month	year	Temperature	RH	Region	WsiqrO	DCiqrO	BUIiqrO	Classes_notfire	FFM
0	11	7	2012	33	76	0.0	14.0	18.7	8.1	1	
1	14	6	2012	30	78	0.0	20.0	7.8	4.4	1	
2	4	6	2012	25	89	0.0	13.0	6.9	1.7	1	
3	16	6	2012	29	87	1.0	15.0	8.0	4.1	1	
4	25	6	2012	31	64	0.0	15.0	63.8	18.3	0	

In [265…
```
#data.corr()['Temperature'].sort_values().index
```

In [266…
```
#data.corr()['Temperature'].sort_values().values
```

In [267…
```
#data.corr()['Temperature'].nlargest(38)
```

In [268…
```
#data.corr()['Temperature'].nsmallest(12)
```

In [269…
```
#list(data.corr()['Temperature'].nlargest(38).index) + list(data.corr()['Temperatu
```

In [270…
```
#data = data[list(data.corr()['Temperature'].nlargest(38).index) + list(data.corr(
```

In [111…
```
data
```

Out[1113]:

	day	month	year	Temperature	RH	Region	WsiqrO	DCiqrO	BUIiqrO	Classes_notfire	F
0	11	7	2012	33	76	0.0	14.0	18.7	8.1	1	
1	14	6	2012	30	78	0.0	20.0	7.8	4.4	1	
2	4	6	2012	25	89	0.0	13.0	6.9	1.7	1	
3	16	6	2012	29	87	1.0	15.0	8.0	4.1	1	
4	25	6	2012	31	64	0.0	15.0	63.8	18.3	0	
…	…	…	…	…	…	…	…	…	…	…	
157	7	8	2012	32	69	0.0	16.0	48.6	17.2	0	
158	11	8	2012	40	31	1.0	15.0	46.3	22.4	0	
159	26	9	2012	31	54	0.0	11.0	16.3	6.2	1	
160	18	7	2012	31	68	0.0	14.0	43.1	14.2	0	
161	22	7	2012	32	48	1.0	18.0	90.1	44.0	0	

162 rows × 20 columns

In [111…
```
data=data.drop(['WsiqrO'        ,'DCiqrO'       ,'BUIiqrO'],axis = 1)
data
```

Out[1114]:

	day	month	year	Temperature	RH	Region	Classes_notfire	FFMCiqrO_Boxcox	ISIiqrO_sq
0	11	7	2012	33	76	0.0	1	2.246811e+07	1.61
1	14	6	2012	30	78	0.0	1	5.950505e+06	1.00
2	4	6	2012	25	89	0.0	1	2.849455e+06	0.00
3	16	6	2012	29	87	1.0	1	2.849455e+06	0.44
4	25	6	2012	31	64	0.0	0	2.969391e+07	2.38
...	...	...	...	...	...	...	...	...	
157	7	8	2012	32	69	0.0	0	2.940889e+07	2.34
158	11	8	2012	40	31	1.0	0	4.198955e+07	4.06
159	26	9	2012	31	54	0.0	1	2.352788e+07	1.58
160	18	7	2012	31	68	0.0	0	2.787833e+07	2.14
161	22	7	2012	32	48	1.0	0	3.718753e+07	3.63

162 rows × 17 columns

Visualizing Scatter Plot: Is it Linear with the Target or Not.

In [111…

```
data.columns
```

Out[1115]:

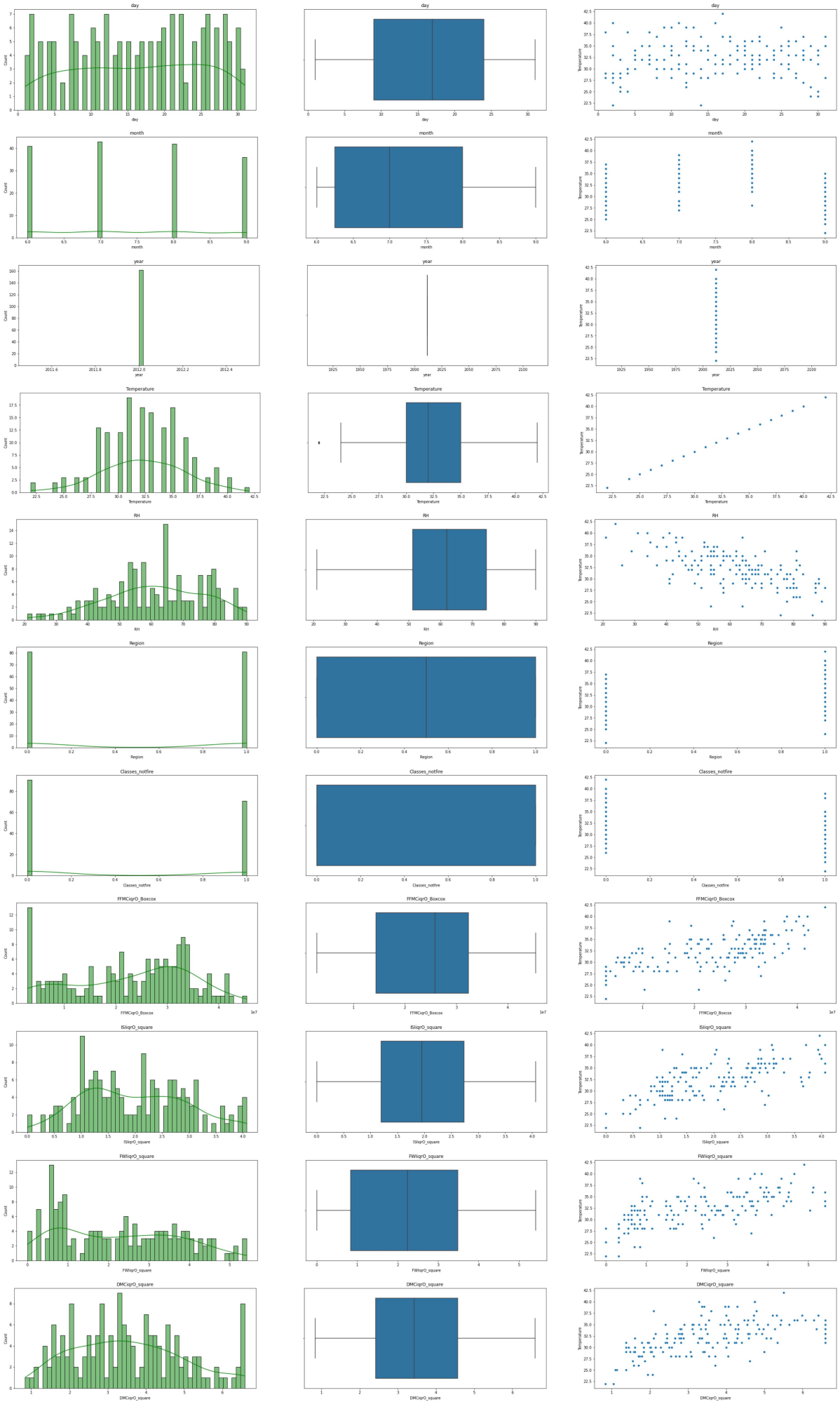
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Region',
       'Classes_notfire', 'FFMCiqrO_Boxcox', 'ISIiqrO_square',
       'FWIiqrO_square', 'DMCiqrO_square', 'DMCiqrO_Boxcox', 'BUIiqrO_square',
       'BUIiqrO_Boxcox', 'DCiqrO_square', 'DCiqrO_Boxcox', 'WsiqrO_Boxcox'],
      dtype='object')
```

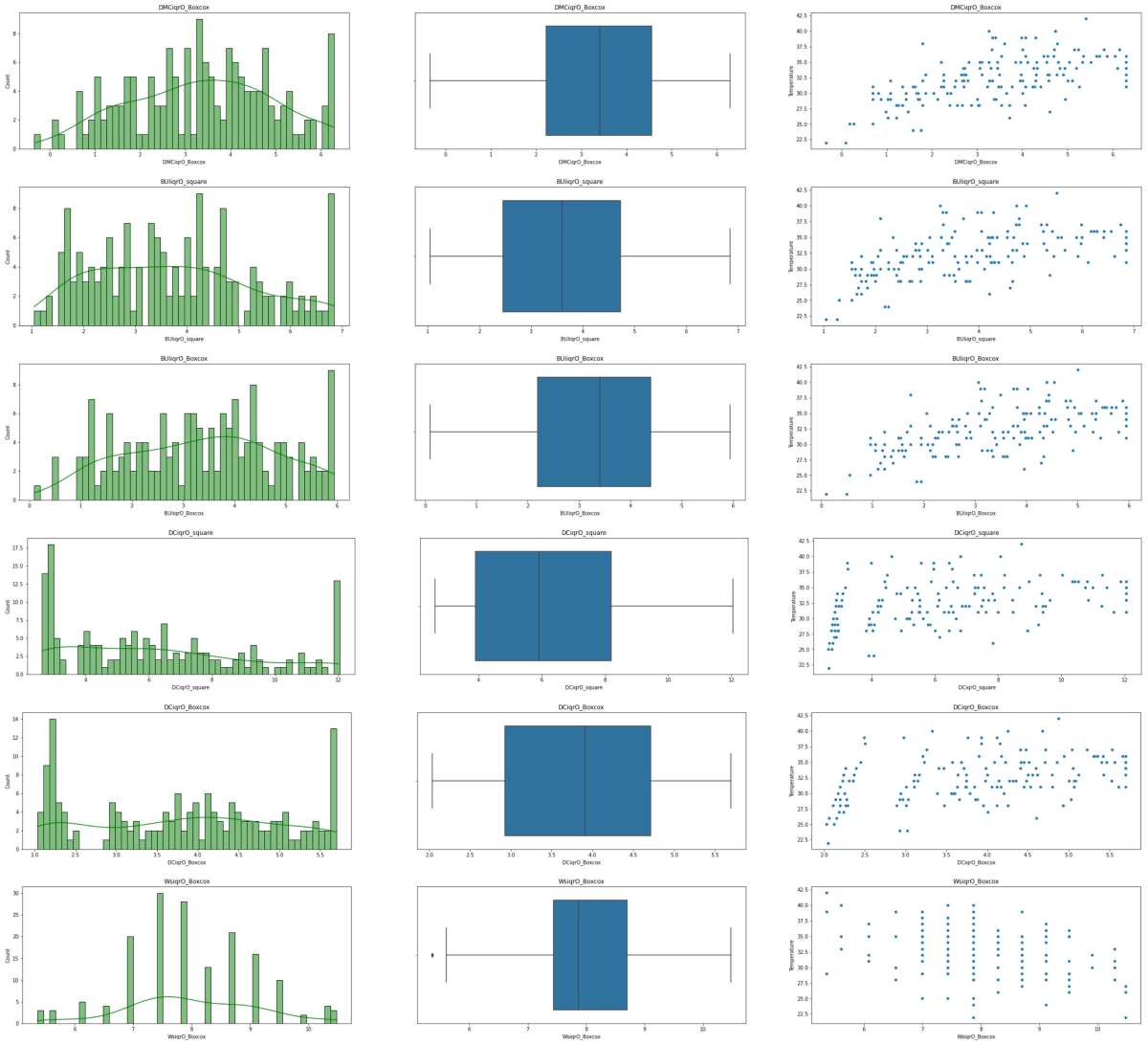
In [111…

```
# Compairing plots
def create_comparison_plot(df,col1):
    '''
    This function compairs the distribution of the features with or without outlier
    '''
    plt.figure(figsize=(40,5))
    plt.subplot(1,3,1)
    sns.histplot(df[col1], bins=50, kde=True,color='g')
    plt.title(col1,loc='center')

    plt.subplot(1,3,2)
    sns.boxplot(df[col1])
    plt.title(col1)

    plt.subplot(1,3,3)
    sns.scatterplot(x=data[col1], y = data['Temperature'])
    plt.title(col1)



    plt.show()
```

In [111…

```
for col in data.columns:
    create_comparison_plot(df = data,col1= col)
```


day
day
day
month
month
month
year
year
year
Temperature
Temperature
Temperature
RH
RH
RH
Region
Region
Region
Classes_notfire
Classes_notfire
Classes_notfire
FFMCiqrO_Boxcox
FFMCiqrO_Boxcox
FFMCiqrO_Boxcox
ISIiqrO_square
ISIiqrO_square
ISIiqrO_square
FWIiqrO_square
FWIiqrO_square
FWIiqrO_square
DMCiqrO_square
DMCiqrO_square
DMCiqrO_square



In [111…
```
data.month.unique()
```

Out[1118]:
```
array([7, 6, 9, 8], dtype=int64)
```

In [111…
```
from scipy.stats import shapiro
shapiro_wick_test = []
for column in data.columns:
    dataToTest = data[column]
    stat,p = shapiro(dataToTest)
    if p > 0.05:
        shapiro_wick_test.append("Normally Distributed")
    else:
        shapiro_wick_test.append("Not Normally Distributed")
result = pd.DataFrame(data=[data.columns, shapiro_wick_test]).T
result.columns = ['Column Name', 'Shapiro Hypothesis Result']
result
```

Out[1119]:

	Column Name	Shapiro Hypothesis Result
0	day	Not Normally Distributed
1	month	Not Normally Distributed
2	year	Normally Distributed
3	Temperature	Normally Distributed
4	RH	Not Normally Distributed
5	Region	Not Normally Distributed
6	Classes_notfire	Not Normally Distributed
7	FFMCiqrO_Boxcox	Not Normally Distributed
8	ISIiqrO_square	Not Normally Distributed
9	FWIiqrO_square	Not Normally Distributed
10	DMCiqrO_square	Not Normally Distributed
11	DMCiqrO_Boxcox	Not Normally Distributed
12	BUIiqrO_square	Not Normally Distributed
13	BUIiqrO_Boxcox	Not Normally Distributed
14	DCiqrO_square	Not Normally Distributed
15	DCiqrO_Boxcox	Not Normally Distributed
16	WsiqrO_Boxcox	Not Normally Distributed

In [112…

```
from scipy.stats import normaltest
normaltest_test = []
for column in data.columns:
    dataToTest = data[column]
    stat,p = normaltest(dataToTest)
    if p > 0.05:
        normaltest_test.append("Normally Distributed")
    else:
        normaltest_test.append("Not Normally Distributed")
result = pd.DataFrame(data=[data.columns, normaltest_test]).T
result.columns = ['Column Name', 'normaltest Hypothesis Result']
result
```

Out[1120]:

	Column Name	normaltest Hypothesis Result
0	day	Not Normally Distributed
1	month	Not Normally Distributed
2	year	Not Normally Distributed
3	Temperature	Normally Distributed
4	RH	Normally Distributed
5	Region	Not Normally Distributed
6	Classes_notfire	Not Normally Distributed
7	FFMCiqrO_Boxcox	Not Normally Distributed
8	ISIiqrO_square	Not Normally Distributed
9	FWIiqrO_square	Not Normally Distributed
10	DMCiqrO_square	Not Normally Distributed
11	DMCiqrO_Boxcox	Not Normally Distributed
12	BUIiqrO_square	Not Normally Distributed
13	BUIiqrO_Boxcox	Not Normally Distributed
14	DCiqrO_square	Not Normally Distributed
15	DCiqrO_Boxcox	Not Normally Distributed
16	WsiqrO_Boxcox	Normally Distributed

In [ ]:

In [ ]:

In [ ]:

In [112…

```
# Saving dataset after handling outliers data
data.to_csv('prepared_tranformed_new_data_train.csv',index=False)
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: