EDA, FE and Logistic Regression Model (Algerian Forest Fires Dataset)

Shubham Verma

Follow me on Linkedin: https://lnkd.in/gPxctEja

Follow me on GitHub: https://lnkd.in/gky-wyFJ

For the code please check out my Machine Learning repository on GitHub

1. EDA and FE

1. Data Profiling
2. Stastical analysis
3. Graphical Analysis
4. Data Cleaning
5. Data Scaling

2. Logistic Regression Model (Classification)

1. Logistic Regression Model
2. Performance metrics for above model
3. Creating Imbalanced dataset
4. Balancing dataset using imbalanced library
5. EDA on balanced dataset
6. New Logistic Regression Model for Imbalanced dataset
7. Performance metrics for new model
8. Performance comparison for both models

Dataset: https://archive.ics.uci.edu/ml/datasets/Algerian+Forest+Fires+Dataset++#

In [1]:
```
from IPython import display
display.Image("forrestfire.png")
```

Out[1]:

1.0 Importing required libraries

In [2]:
```
### Pandas and Numpy
import pandas as pd
import numpy as np

### Visualisation libraries
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

### For Q-Q Plot
import scipy.stats as stats

### To ignore warnings
import warnings
warnings.filterwarnings('ignore')

### Machine Learning libraries
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report

### To be able to see maximum columns on screen
pd.set_option('display.max_columns', 500)

### To save the model
import pickle
```

2.0 Importing Dataset and Cleaning the Data

In [3]:
```
### reading csv file
dataset=pd.read_csv('Algerian_forest_fires_dataset_UPDATE.csv',header=1 )

dataset.iloc[121:].head(4) # index 122, 123 need to be removed from dataset
```

Out[3]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
121	30	09	2012	25	78	14	1.4	45	1.9	7.5	0.2	2.4	0.1	not fire
122	Sidi-Bel Abbes Region Dataset	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
123	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
124	01	06	2012	32	71	12	0.7	57.1	2.5	8.2	0.6	2.8	0.2	not fire

2.1 Info about dataset and its attributes

1. The dataset includes 244 instances that regroup a data of two regions of Algeria, namely the Bejaia region located in the northeast of Algeria and the Sidi Bel-abbes region located in the northwest of Algeria.
2. 122 instances for each region.
3. The period from June 2012 to September 2012.
4. The dataset includes 11 attribues and 1 output attribue (classes)
5. The 244 instances have been classified into fire (138 classes) and notfire (106 classes) classes.

Attributes

1. Date : (DD/MM/YYYY) Day, month ('june' to 'september'), year (2012)

Weather data observations

1. Temp : temperature noon (temperature max) in Celsius degrees: 22 to 42
2. RH : Relative Humidity in %: 21 to 90
3. Ws :Wind speed in km/h: 6 to 29
4. Rain: total day in mm: 0 to 16.8

FWI Components

1. Fine Fuel Moisture Code (FFMC) index from the FWI system: 28.6 to 92.5
2. Duff Moisture Code (DMC) index from the FWI system: 1.1 to 65.9
3. Drought Code (DC) index from the FWI system: 7 to 220.4
4. Initial Spread Index (ISI) index from the FWI system: 0 to 18.5

5. Buildup Index (BUI) index from the FWI system: 1.1 to 68
6. Fire Weather Index (FWI) Index: 0 to 31.1
7. Classes: two classes, namely fire and not fire

2.2 Dropping rows which have no information

In [4]:
```
#dropping rows having region name and headder
dataset.drop(index=[122,123], inplace=True) # droping row 122,123 from dataset
dataset.reset_index(inplace=True)
dataset.drop('index', axis=1, inplace=True)

dataset.iloc[121:].head()
```

Out[4]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
121	30	09	2012	25	78	14	1.4	45	1.9	7.5	0.2	2.4	0.1	not fire
122	01	06	2012	32	71	12	0.7	57.1	2.5	8.2	0.6	2.8	0.2	not fire
123	02	06	2012	30	73	13	4	55.7	2.7	7.8	0.6	2.9	0.2	not fire
124	03	06	2012	29	80	14	2	48.7	2.2	7.6	0.3	2.6	0.1	not fire
125	04	06	2012	30	64	14	0	79.4	5.2	15.4	2.2	5.6	1	not fire

2.3 Creating Region feature

In [5]:
```
### creating feature called Region 0 for Bejaia region and 1 for Sidi Bel-abbes region
dataset.loc[:122,'Region']=0
dataset.loc[122:, 'Region']=1

dataset.iloc[120:].head(8)
```

Out[5]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
120	29	09	2012	26	80	16	1.8	47.4	2.9	7.7	0.3	3	0.1	not fire	0.0
121	30	09	2012	25	78	14	1.4	45	1.9	7.5	0.2	2.4	0.1	not fire	0.0
122	01	06	2012	32	71	12	0.7	57.1	2.5	8.2	0.6	2.8	0.2	not fire	1.0
123	02	06	2012	30	73	13	4	55.7	2.7	7.8	0.6	2.9	0.2	not fire	1.0
124	03	06	2012	29	80	14	2	48.7	2.2	7.6	0.3	2.6	0.1	not fire	1.0
125	04	06	2012	30	64	14	0	79.4	5.2	15.4	2.2	5.6	1	not fire	1.0
126	05	06	2012	32	60	14	0.2	77.1	6	17.6	1.8	6.5	0.9	not fire	1.0
127	06	06	2012	35	54	11	0.1	83.7	8.4	26.3	3.1	9.3	3.1	fire	1.0

2.4 Datatypes and describe

In [6]:
```
# here it is visible that all datatypes are in object
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          244 non-null    object 
 1   month        244 non-null    object 
 2   year         244 non-null    object 
 3   Temperature  244 non-null    object 
 4    RH          244 non-null    object 
 5    Ws          244 non-null    object 
 6   Rain         244 non-null    object 
 7   FFMC         244 non-null    object 
 8   DMC          244 non-null    object 
 9   DC           244 non-null    object 
 10  ISI          244 non-null    object 
 11  BUI          244 non-null    object 
 12  FWI          244 non-null    object 
 13  Classes      243 non-null    object 
 14  Region       244 non-null    float64
dtypes: float64(1), object(14)
memory usage: 28.7+ KB
```

In [7]:
```
dataset.describe(include='all').T
```

Out[7]:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
day	244	31	01	8	NaN	NaN	NaN	NaN	NaN	NaN	NaN
month	244	4	07	62	NaN	NaN	NaN	NaN	NaN	NaN	NaN
year	244	1	2012	244	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Temperature	244	19	35	29	NaN	NaN	NaN	NaN	NaN	NaN	NaN
RH	244	62	64	10	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Ws	244	18	14	43	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Rain	244	39	0	133	NaN	NaN	NaN	NaN	NaN	NaN	NaN
FFMC	244	173	88.9	8	NaN	NaN	NaN	NaN	NaN	NaN	NaN
DMC	244	166	7.9	5	NaN	NaN	NaN	NaN	NaN	NaN	NaN
DC	244	198	8	5	NaN	NaN	NaN	NaN	NaN	NaN	NaN
ISI	244	106	1.1	8	NaN	NaN	NaN	NaN	NaN	NaN	NaN
BUI	244	174	3	5	NaN	NaN	NaN	NaN	NaN	NaN	NaN
FWI	244	127	0.4	12	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Classes	243	8	fire	131	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Region	244.0	NaN	NaN	NaN	0.5	0.501028	0.0	0.0	0.5	1.0	1.0

2.5 Data Cleaning

In [8]:
```
# here it is visible that some columns have spaces in the names like RH, Ws
dataset.columns
```

Out[8]:
```
Index(['day', 'month', 'year', 'Temperature', ' RH', ' Ws', 'Rain ', 'FFMC',
       'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'Classes  ', 'Region'],
      dtype='object')
```

In [9]:
```
# stripping spaces from column names
dataset.columns= [col_name.strip() for col_name in dataset.columns]
dataset.columns
```

Out[9]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Ws', 'Rain', 'FFMC',
       'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'Classes', 'Region'],
      dtype='object')
```

In [10]:
```
### converting all feature values to string so that we can do data cleaning as shown below.
dataset=dataset.astype(str)
```

In [11]:
```
### somes values in colums also have space
for feature in ['Rain', 'FFMC',
       'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'Classes']:
    dataset[feature]= dataset[feature].str.replace(" ","")
```

In [12]:
```
### index no 165 for feature name FWI has value fire
dataset[dataset['FWI']== 'fire'].index
```

Out[12]:
```
Int64Index([165], dtype='int64')
```

In [13]:
```
### replacing fire value witha float value
dataset.loc[165,'FWI']=' 0.1'
```

In [14]:
```
### replacing nan value wit fire to make data equal to the info given in dataset
dataset[dataset['Classes']== 'nan'].index
dataset.loc[165,'Classes']='fire'
```

In [15]:
```
### encoding classes feature
dataset['Classes']=dataset['Classes'].str.replace('notfire','0')
dataset['Classes']=dataset['Classes'].str.replace('fire','1')
```

In [16]:
```
### Dropping year feature as data is realted to year 2012.
dataset.drop('year', axis=1, inplace=True)
```

2.6 Changing datatype to Numerical from Object

In [17]:
```
### changing datatypes of features to numerical for numerical features as all are in object

datatype_convert={'day':'int64', 'month':'int64', 'Temperature':'int64', 'RH':'int64', 'Ws':'int64',
                  'Rain':'float64', 'FFMC':'float64', 'DMC':'float64', 'DC':'float64', 'ISI':'float64', 'BUI':'float64',
                  'FWI':'float64', 'Classes':'int64', 'Region':'float64'}
```

```
dataset=dataset.astype(datatype_convert)
dataset.dtypes
```

Out[17]:
```
day             int64
month           int64
Temperature     int64
RH              int64
Ws              int64
Rain          float64
FFMC          float64
DMC           float64
DC            float64
ISI           float64
BUI           float64
FWI           float64
Classes         int64
Region        float64
dtype: object
```

In [18]:
```
dataset.shape
```

Out[18]:
```
(244, 14)
```

2.7 Checking Null values and Duplicates

In [19]:
```
### checking for null values

dataset.isnull().sum()
```

Out[19]:
```
day            0
month          0
Temperature    0
RH             0
Ws             0
Rain           0
FFMC           0
DMC            0
DC             0
ISI            0
BUI            0
FWI            0
Classes        0
Region         0
dtype: int64
```

In [20]:
```
dataset[dataset.duplicated()]
```

Out[20]:

day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region

Observation

1. There is no null value in dataset.
2. Total 244 roww and 15 columns is present.
3. There is no duplicate observation in dataset.

3.0 Analysis of Features

3.1 Comparing Classes and Region features

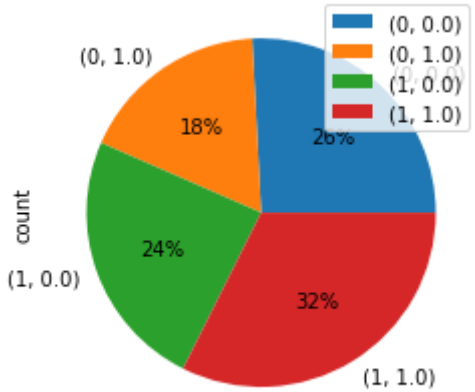
In [21]:
```
### 0 is not fire and 1 i fire for Classes
### 0 is Bejaia region and 1 is Sidi Bel-abbes region
data_pie=dataset.groupby(['Classes','Region']).agg({'Classes':['count']})
data_pie
```

Out[21]:

		Classes
		count
Classes	**Region**	
0	**0.0**	63
	1.0	43
1	**0.0**	59
	1.0	79

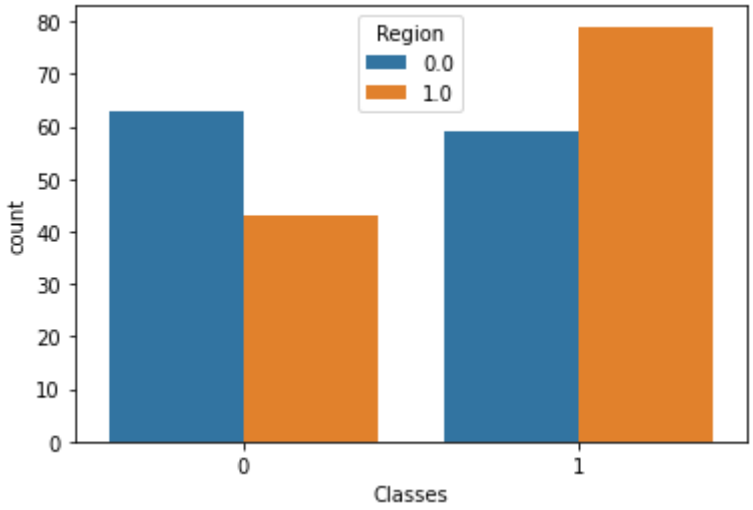
In [22]:
```
### Plotting Pie chart
### Here 0, 1 is not fire and fire respectively
### Here 0.0 and 1.0 is Bejaia region and Sidi Bel-abbes region respectively
plt.figure(figsize=(7,6))
data_pie.plot(kind='pie', y='Classes', autopct='%1.0f%%')
```

Out[22]:
```
<AxesSubplot:ylabel='count'>

<Figure size 504x432 with 0 Axes>
```



In [23]:
```
### 0 is not fire and 1 i fire for Classes
### 0 is Bejaia region and 1 is Sidi Bel-abbes region
sns.countplot(data=dataset, x='Classes', hue='Region')
```

Out[23]:
```
<AxesSubplot:xlabel='Classes', ylabel='count'>
```



Observation

1. It is evident that Sidi Bel-abbes region has more occurance of fire than Bejaia region.
2. It is also evident that there is more cases of fire than not fire.

3.2 Numerical features

In [24]:
```
### Getting list of numerical features excluding Classes and Region

numerical_features=[feature for feature in dataset.columns if feature not in ['Classes', 'Region']]
print(numerical_features)
```

```
['day', 'month', 'Temperature', 'RH', 'Ws', 'Rain', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI', 'FWI']
```

In [25]:
```
### Getting uniques values in each features

dataset[dataset.columns].nunique()
```

Out[25]:
```
day             31
month            4
Temperature     19
RH              62
Ws              18
Rain            39
FFMC           173
DMC            166
DC             198
ISI            106
BUI            174
FWI            125
Classes          2
Region           2
dtype: int64
```

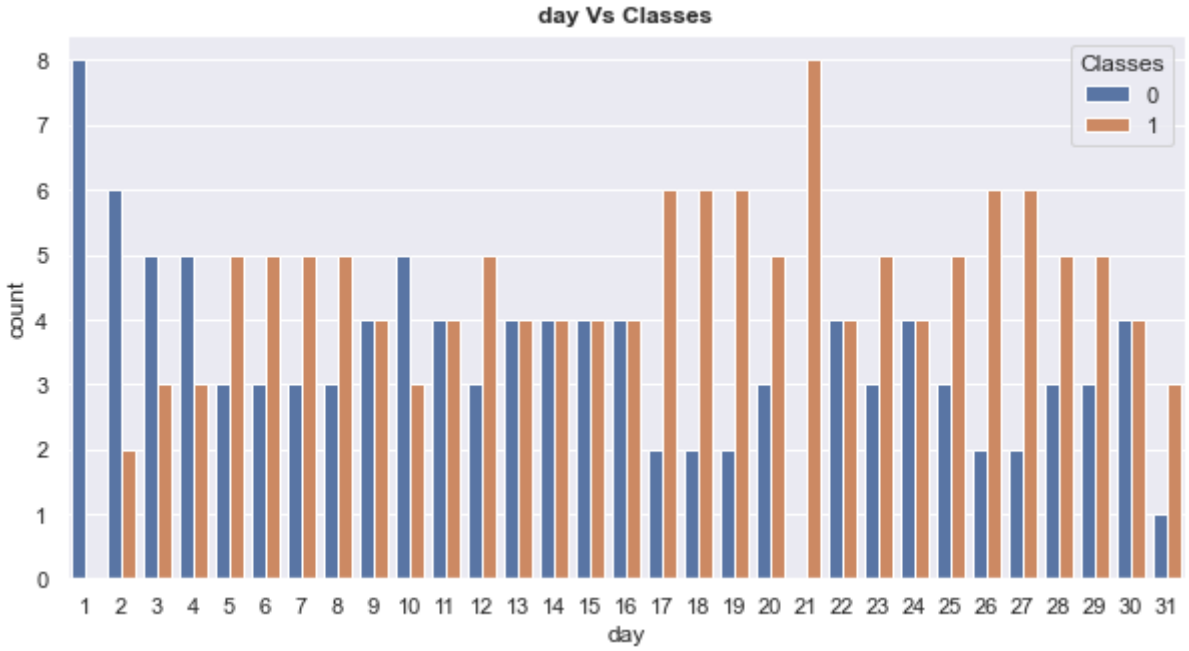
3.3.0 Discrete Numerical Features

In [26]:
```
# here the assumption to consider a feature discrete is that it should have less than 35 unique values otherwise it will be
# considered continuous feature

discrete_features=[feature for feature in numerical_features+['Region'] if len(dataset[feature].unique())<35]
discrete_features
```

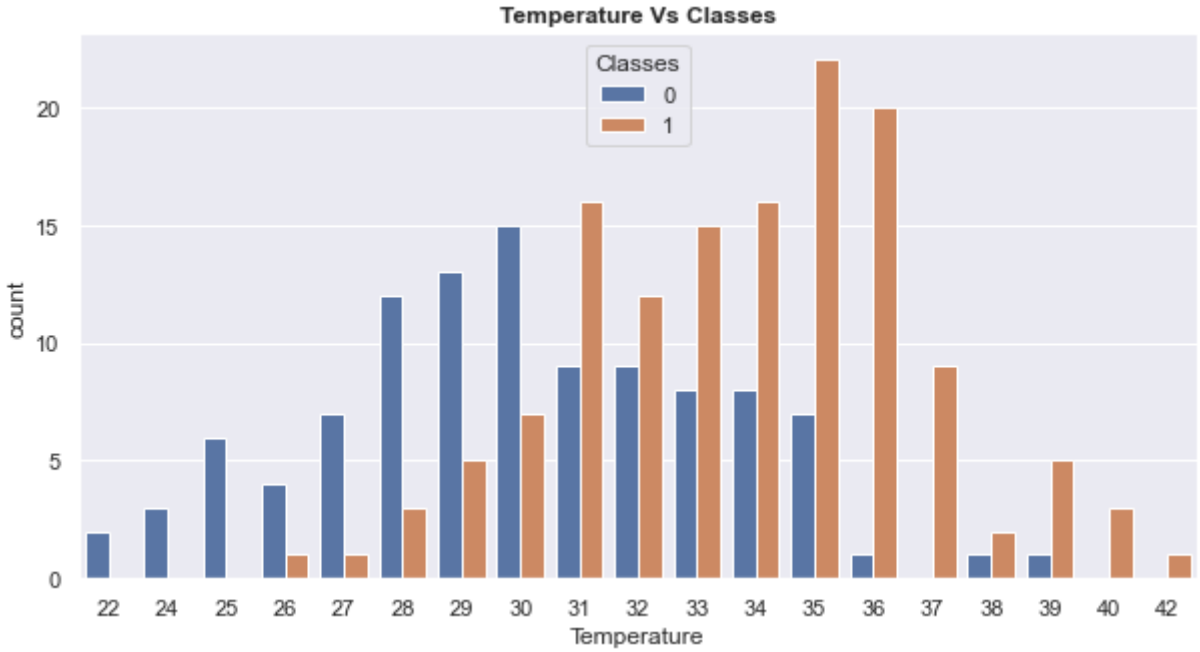
Out[26]:
```
['day', 'month', 'Temperature', 'Ws', 'Region']
```

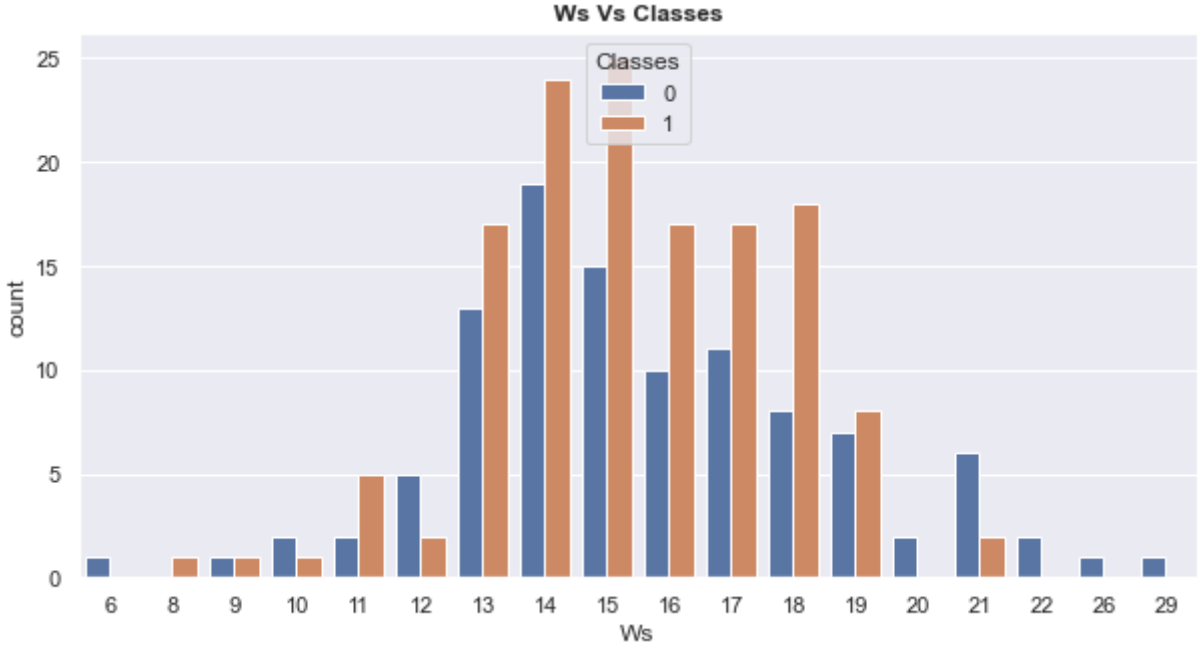
3.3.1 Discrete Numerical Feature vs Target Feature

In [27]:
```
### this is bivariate analysis between target feature classes and discrete numerical features
### for this we plot count plot
for feature in discrete_features:
    sns.set(rc={'figure.figsize':(10,5)})
    sns.countplot(data=dataset, x=feature, hue='Classes')
    plt.title("{} Vs Classes".format(feature),fontweight="bold")
    plt.show();
```


day Vs Classes
Classes
0
1
count
day
0 1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31


month Vs Classes
Classes
0
1
count
month
0 10 20 30 40 50
6 7 8 9


Temperature Vs Classes
Classes
0
1
count
Temperature
22 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 42
0 5 10 15 20


Ws Vs Classes
Classes
0
1
count
Ws
6 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 26 29
0 5 10 15 20 25

Observation

1. From day vs Classes plot it is visible that on almost all days the occurance of fire is there, and its count is more than or equal to the count of no fire cases.
2. From month vs Classes plot it is visible that july and august month have more cases of occurance of fire as compared to other two months of june and september where occurance of fire is less as compared to no fire.
3. The month of august has highest no of cases of occurance of fire.
4. Overall cases of occurance of fire is more than the cases of no occurance of fire.
5. From temperature vs Classes plot it is visible that temperature between 30 to 37 degree celcius have most no of cases of occurance of fire.
6. From windspeed vs Classes plot it is visible that for wind speed between 13 to 19 Km/hr range there is most no of occurance of fire.
7. From Region vs Class plot it is visible that in Bejaia region, the no of cases of occurance of fire is less compared to no fire.
8. In Sidi Bel-abbes region the no of cases of occurance of fire is more compared to no fire. Also Overall no of cases of occurance of fire is more in Sidi Bel-abbes region as compared to Bejaia region.
9. Most no of fires occured on 21st of the month.
10. Least no of fires occured on 2nd of the month.

11. For most days either fire occured was greater than or equal to no fire occured.

12. Most no of cases of fire occured are in the month of august and least no of cases of fire occured is in month of september.

13. July and august have more cases of fire as compared to no fire.
14. June and september have more cases of no fire as compared to fire.

15. In Bejaia region, the no of cases of occurance of fire is less compared to no of cases of occurance of no fire.

16. In Sidi Bel-abbes region the no of cases of occurance of fire is more compared to no fire.
17. Also Overall no of cases of occurance of fire is more in Sidi Bel-abbes region as compared to Bejaia region.

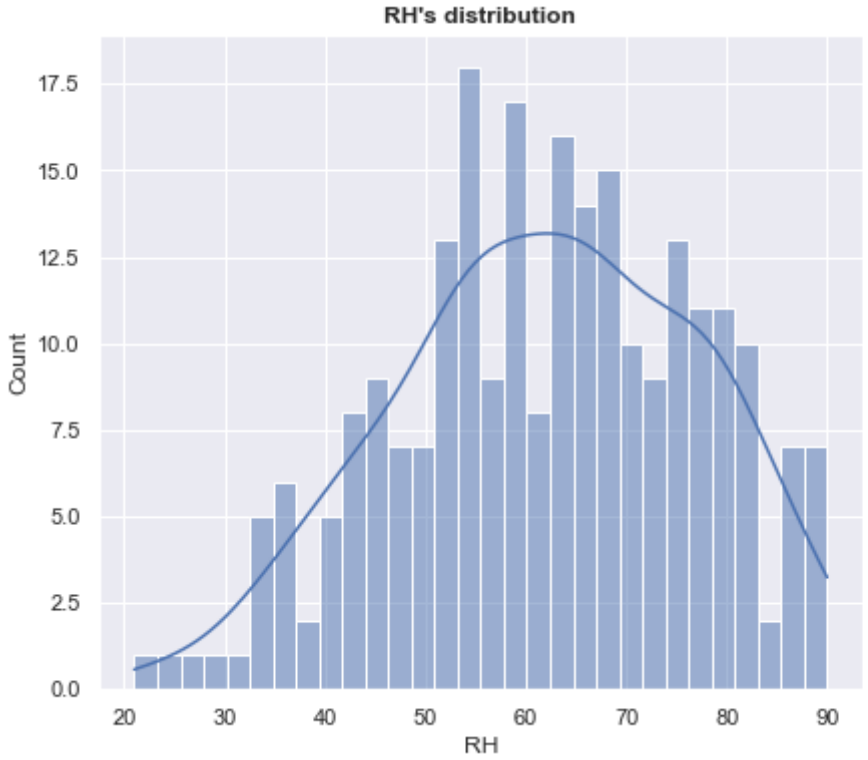
3.4.0 Continuous Numerical Features

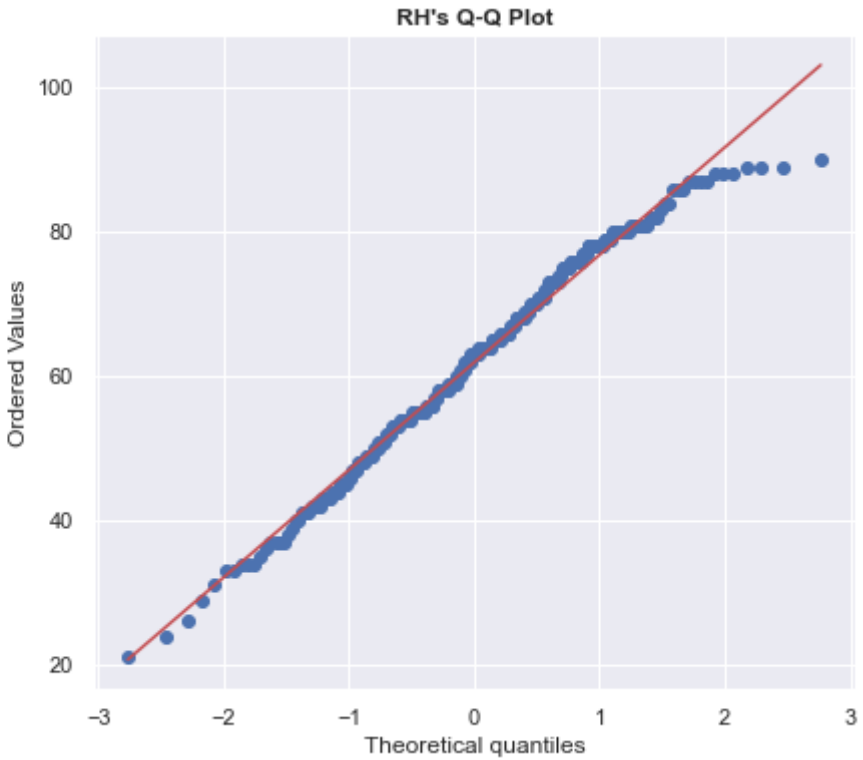
In [28]:
```
continuous_features=[feature for feature in numerical_features if feature not in discrete_features]
print(continuous_features)
```

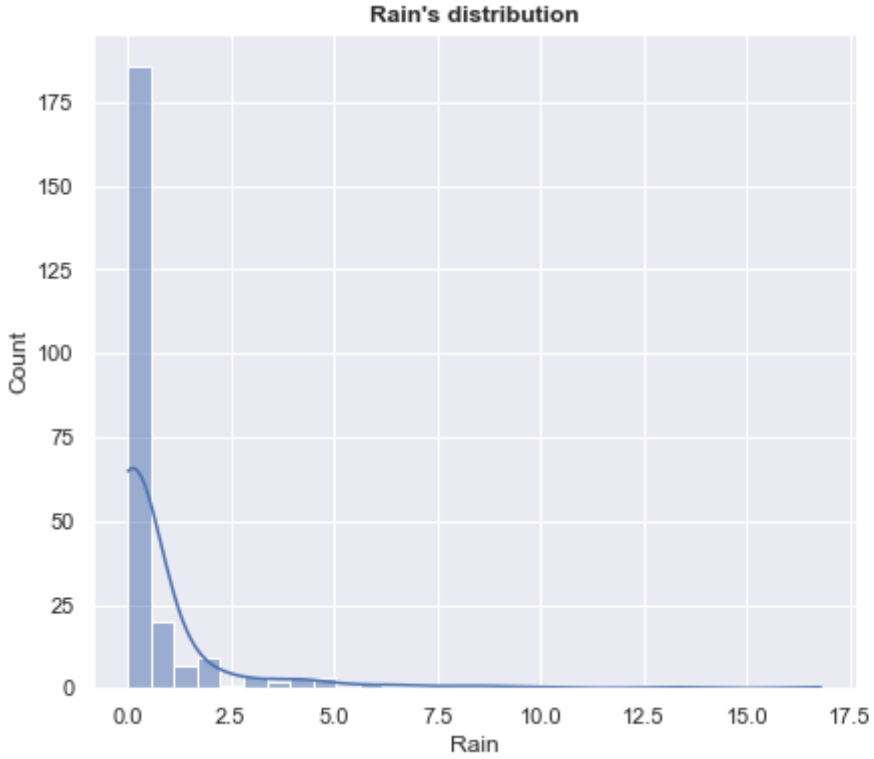
```
['RH', 'Rain', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI', 'FWI']
```

3.4.1 Distribution of Continuous Numerical Features

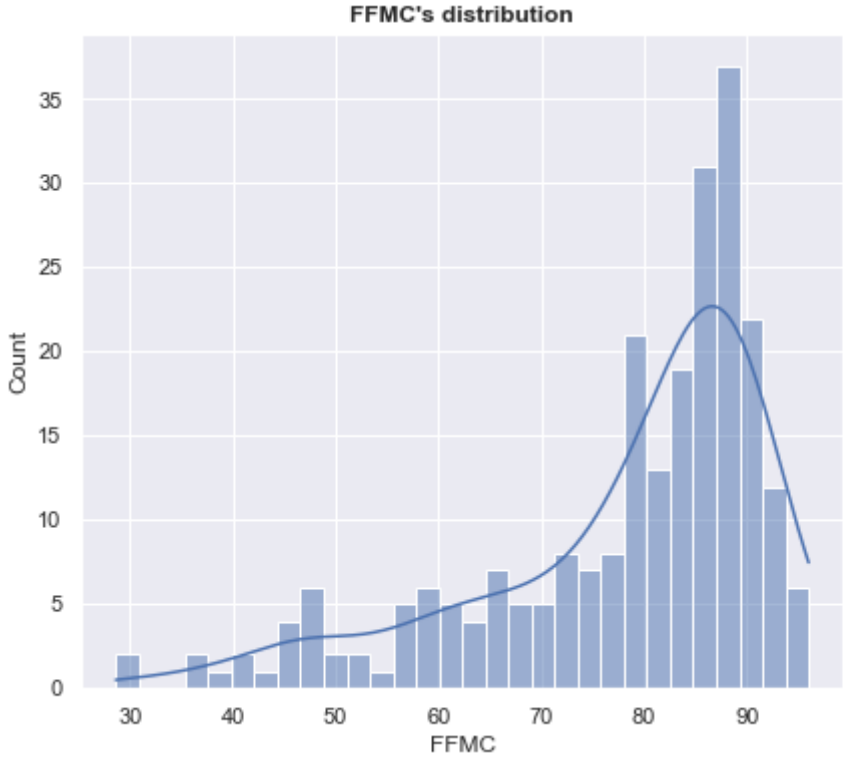
In [29]:
```
### Checking distribution of Continuous numerical features

for i in continuous_features:
    plt.figure(figsize=(15,6))
    plt.subplot(121)
    sns.histplot(data=dataset, x=i, kde=True, bins=30)
    plt.title("{}'s distribution".format(i),fontweight="bold")

    plt.subplot(122)
    stats.probplot(dataset[i], dist='norm', plot=plt)
    plt.title("{}'s Q-Q Plot".format(i),fontweight="bold")
```

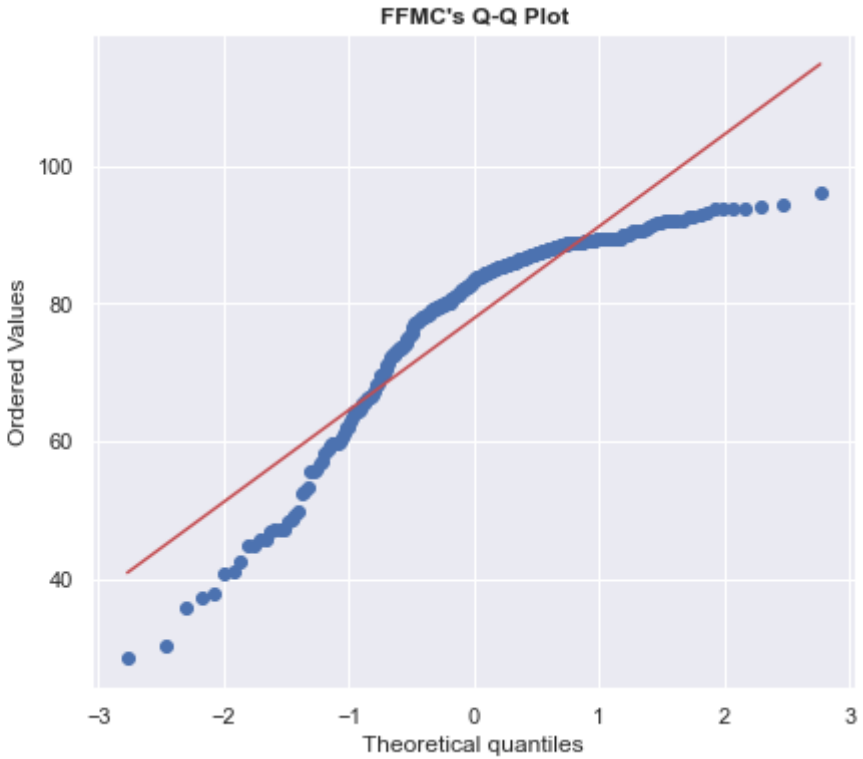

RH's distribution
Count
RH
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20
30
40
50
60
70
80
90

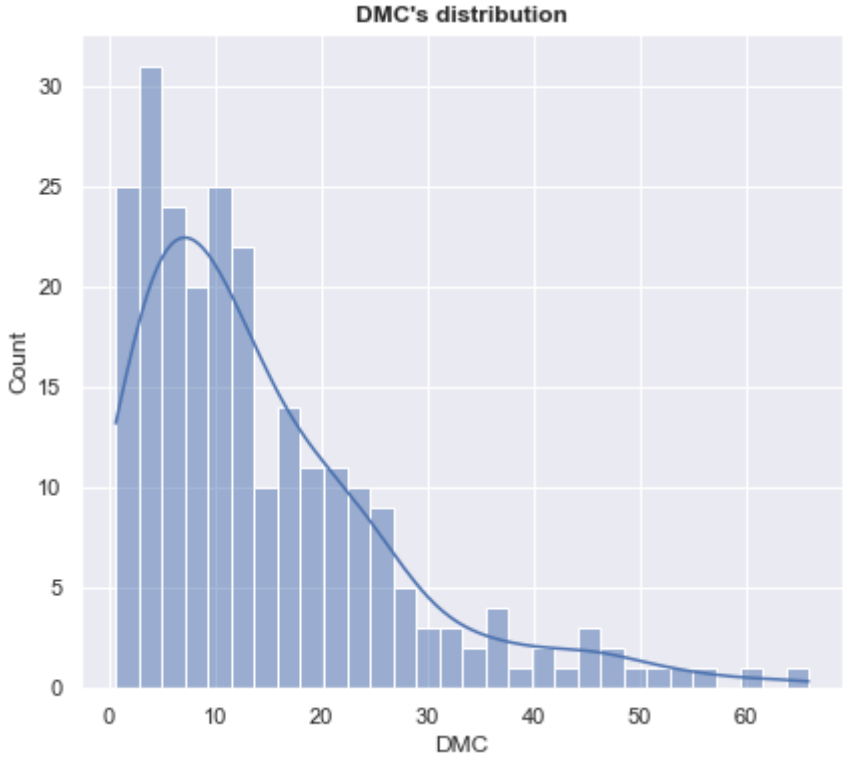

RH's Q-Q Plot
Ordered Values
Theoretical quantiles
20
40
60
80
100
−3
−2
−1
0
1
2
3

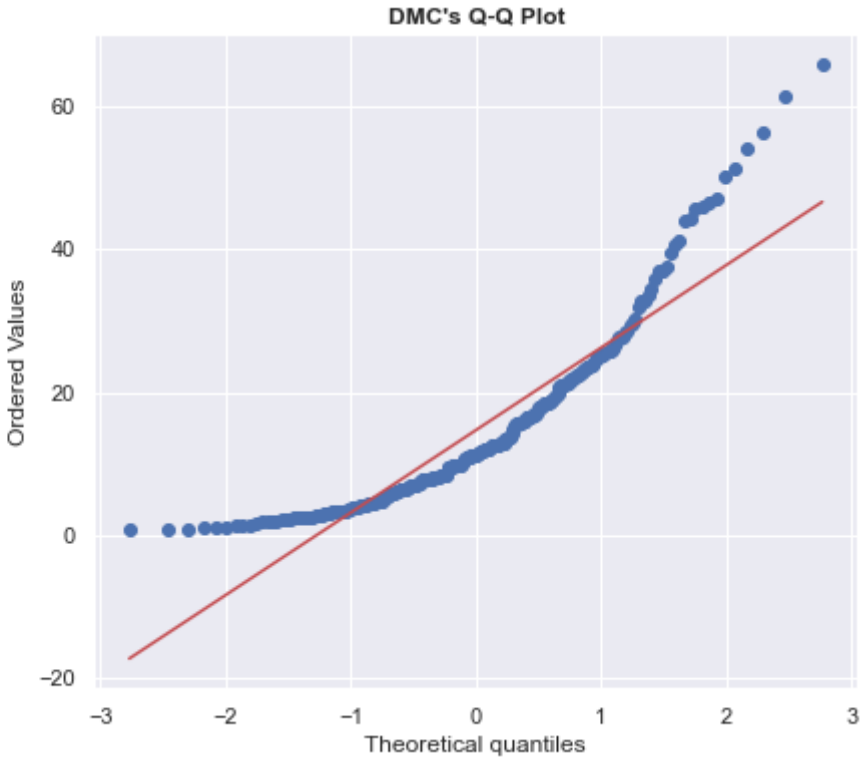

Rain's distribution
Count
Rain
0
25
50
75
100
125
150
175
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

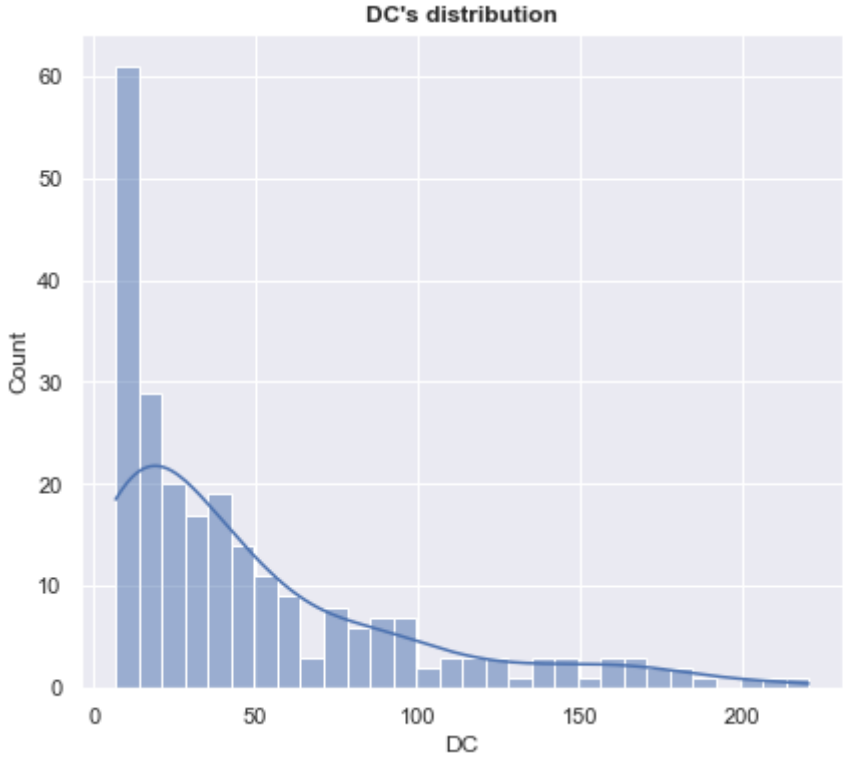

Rain's Q-Q Plot
Ordered Values
Theoretical quantiles
−2.5
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
−3
−2
−1
0
1
2
3

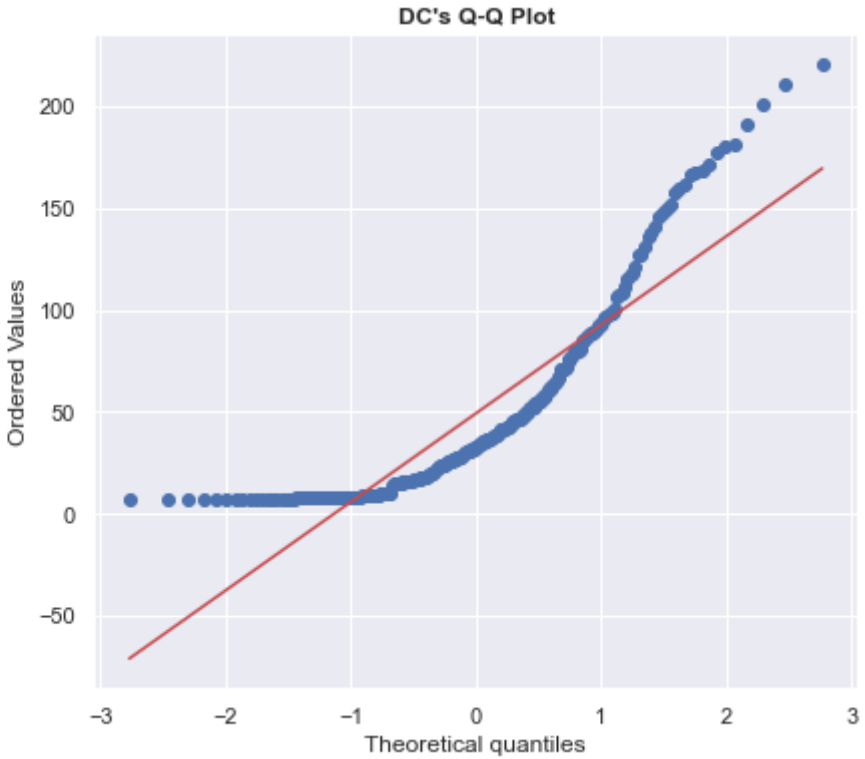

FFMC's distribution
Count
FFMC
0
5
10
15
20
25
30
35
30
40
50
60
70
80
90

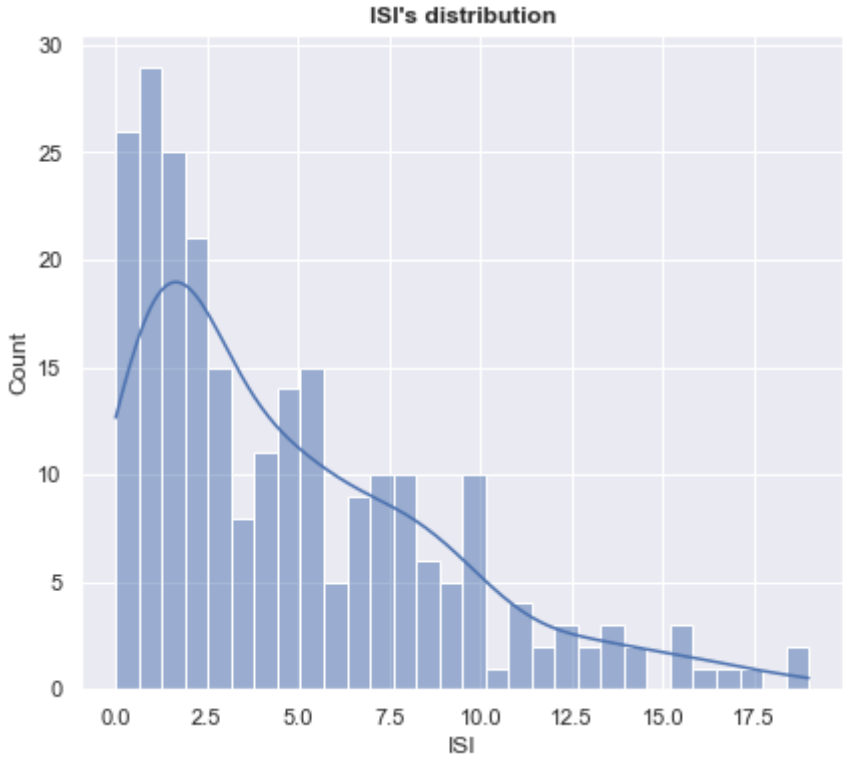

FFMC's Q-Q Plot
Ordered Values
Theoretical quantiles
40
60
80
100
−3
−2
−1
0
1
2
3

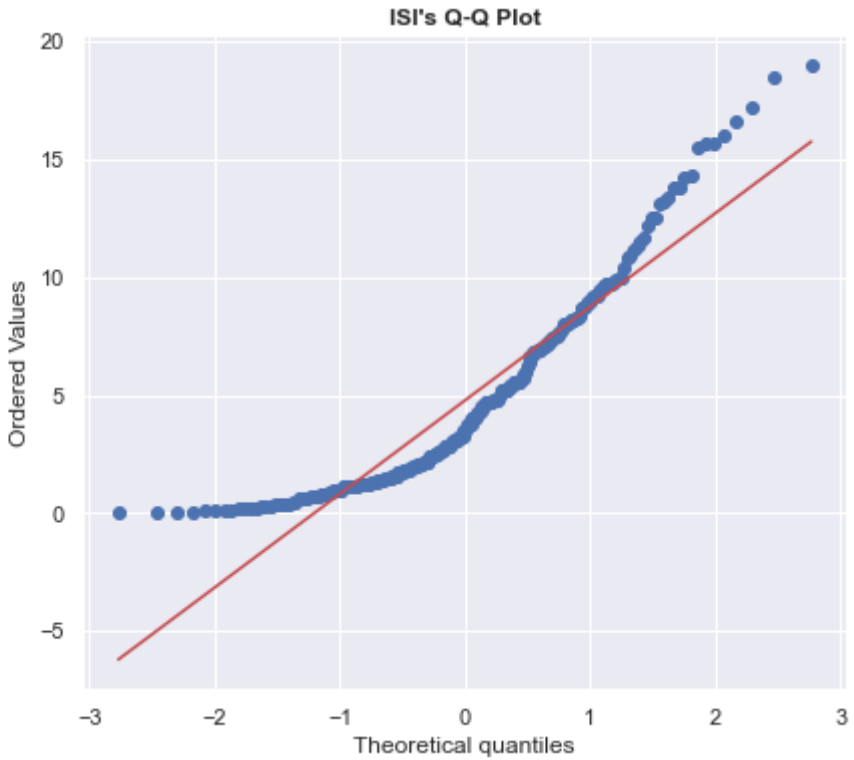

DMC's distribution
Count
DMC
0
5
10
15
20
25
30
0
10
20
30
40
50
60

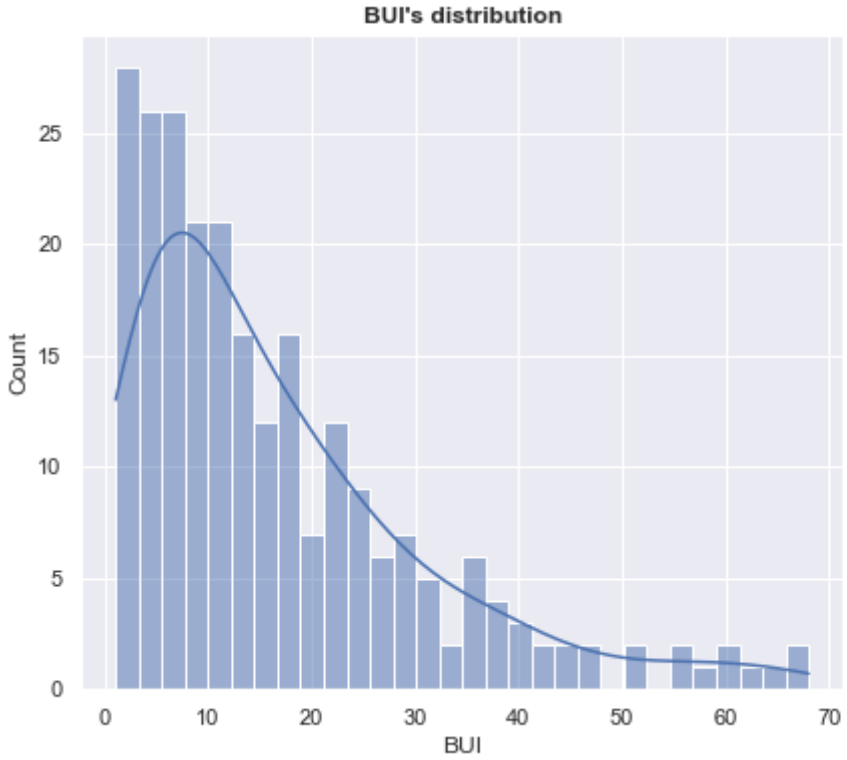

DMC's Q-Q Plot
Ordered Values
Theoretical quantiles
−20
0
20
40
60
−3
−2
−1
0
1
2
3

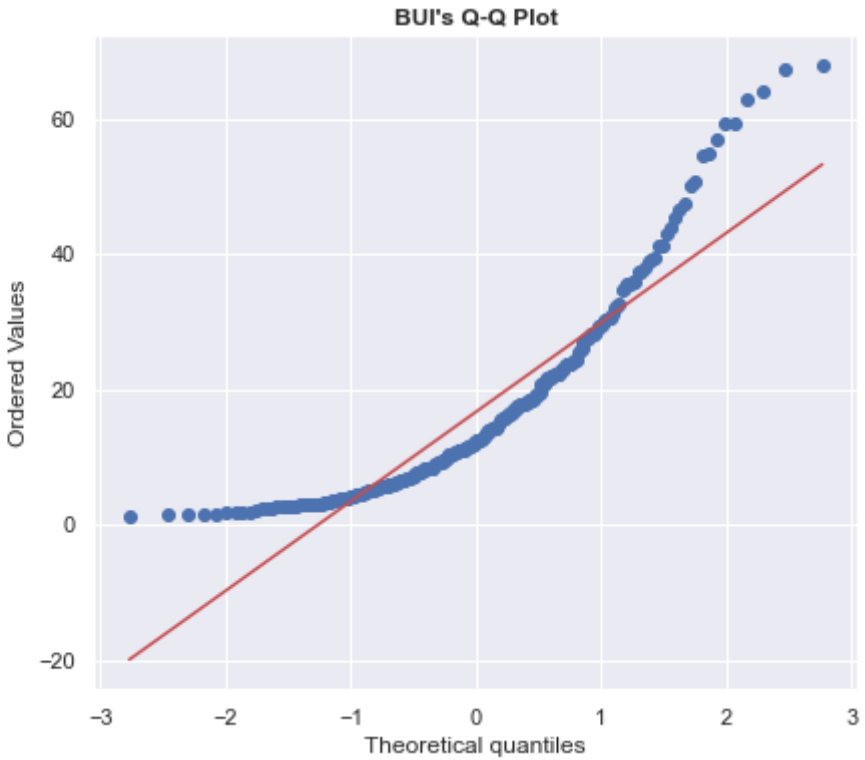

DC's distribution
Count
DC
0
10
20
30
40
50
60
0
50
100
150
200

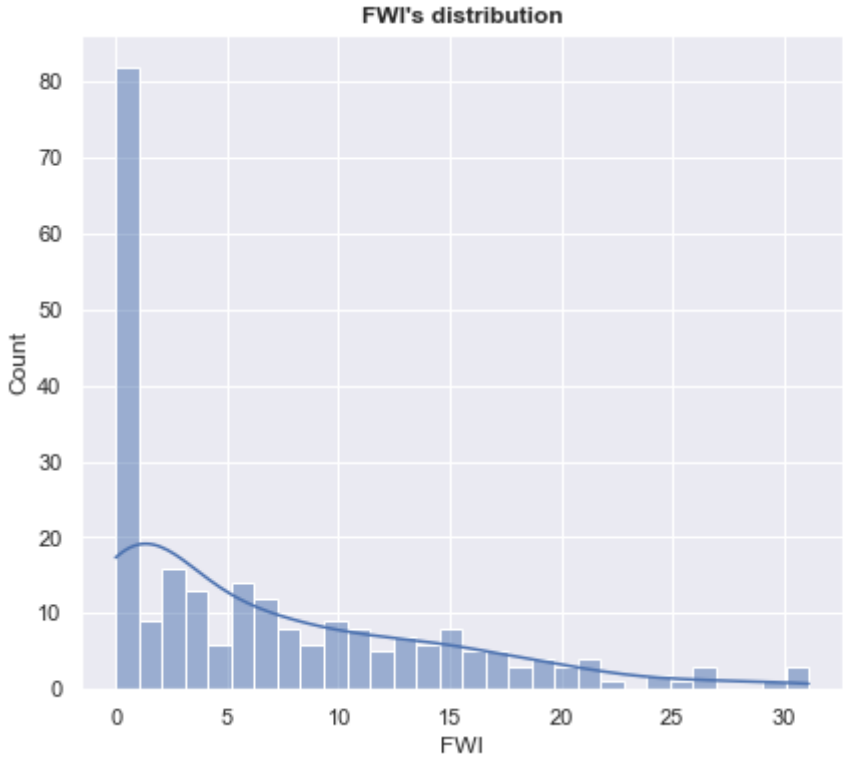

DC's Q-Q Plot
Ordered Values
Theoretical quantiles
−50
0
50
100
150
200
−3
−2
−1
0
1
2
3

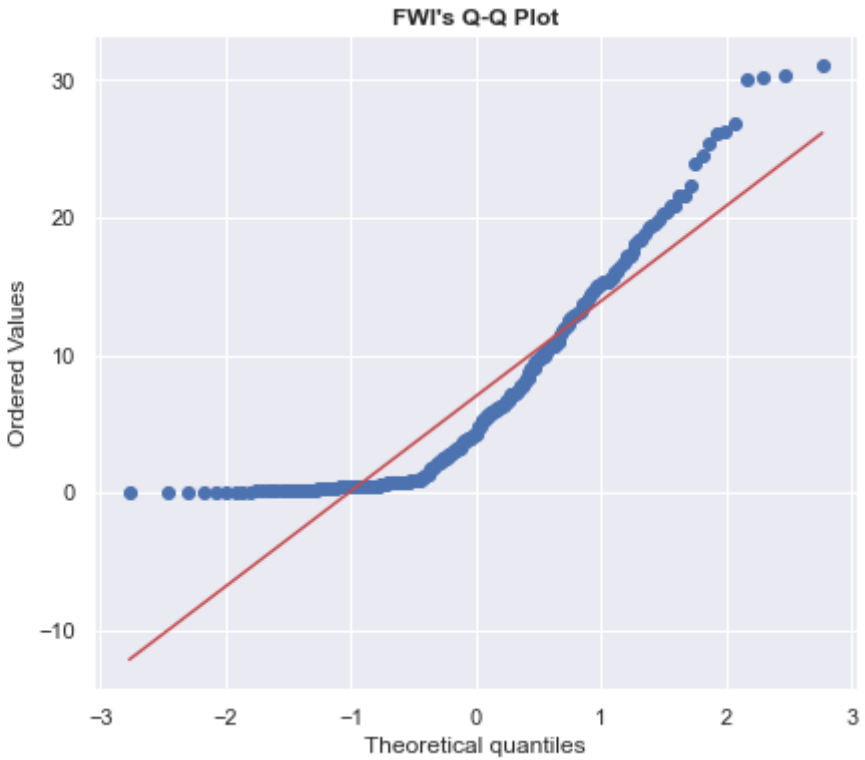

ISI's distribution
Count
ISI
0
5
10
15
20
25
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


ISI's Q-Q Plot
Ordered Values
Theoretical quantiles
−5
0
5
10
15
20
−3
−2
−1
0
1
2
3


BUI's distribution
Count
BUI
0
5
10
15
20
25
0
10
20
30
40
50
60
70


BUI's Q-Q Plot
Ordered Values
Theoretical quantiles
−20
0
20
40
60
−3
−2
−1
0
1
2
3





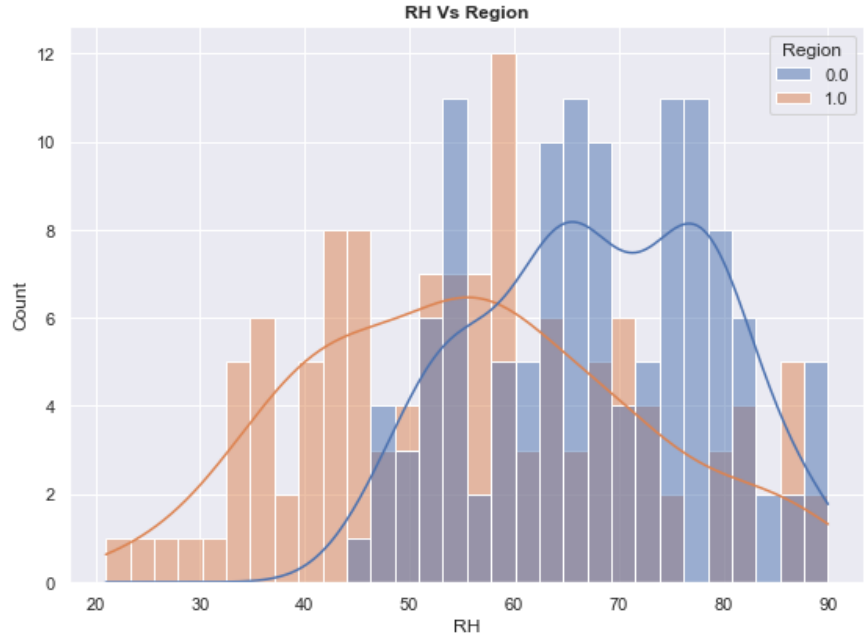
Observation

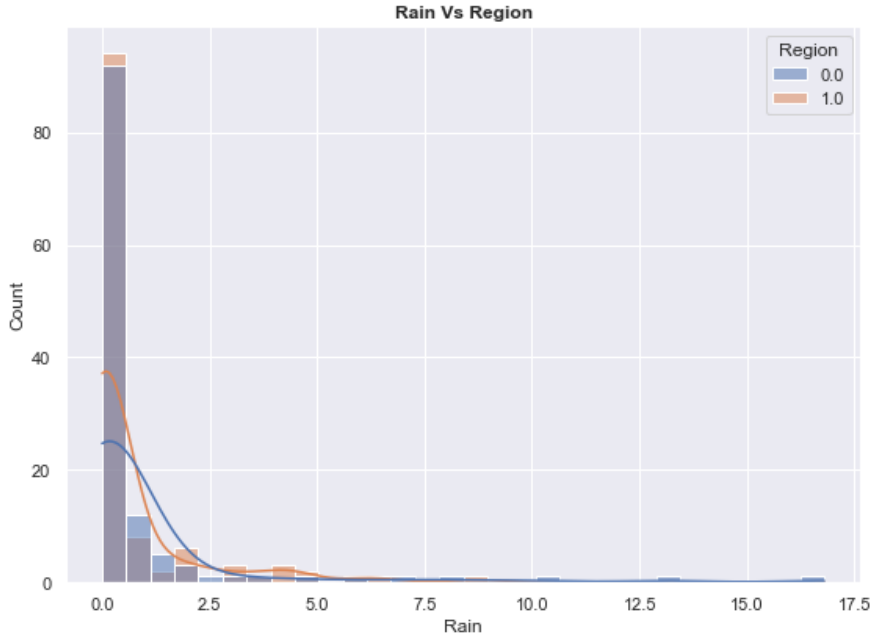
1. Relative humidity is following gaussian distribution.
2. Rain, DMC, DC, ISI, BUI, FWI are following right skewed distribution(Log-Normal distribution).
3. FFMC feature follows left skwed distribution.

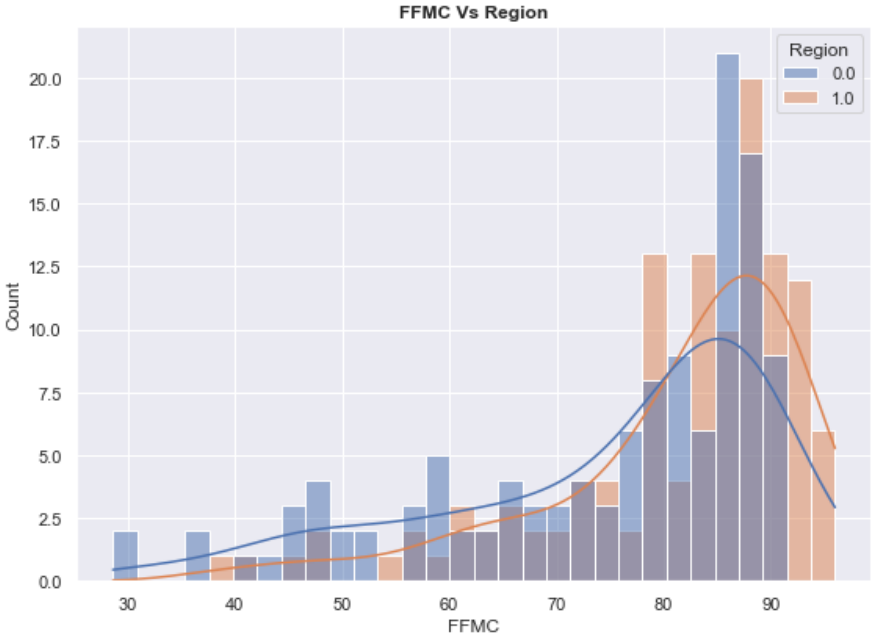
3.4.2 Comparing Continuous Numerical Features with Region

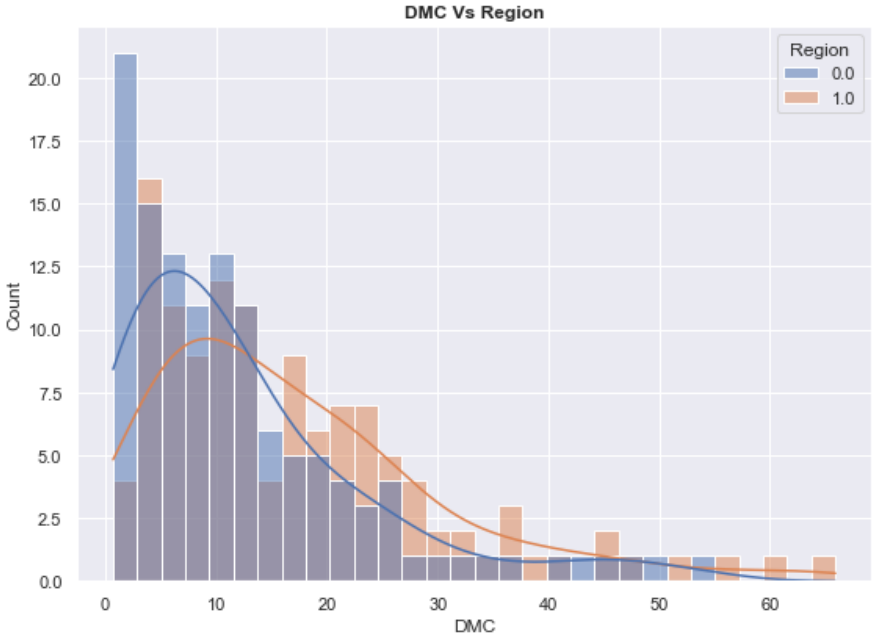
In [30]:
```
### Comparing Continuous numerical features with Region

plt.figure(figsize=(20,30))
for i in enumerate(continuous_features):
    plt.subplot(4, 2, i[0]+1)
    sns.set(rc={'figure.figsize':(7,7)})
```

```
    sns.histplot(data=dataset, x=i[1], kde=True, bins=30, color='blue', hue='Region')
    plt.title("{} Vs Region".format(i[1]),fontweight="bold")
```

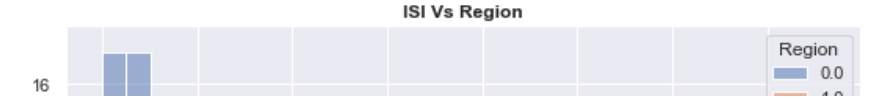

RH Vs Region
Region
0.0
1.0
Count
RH

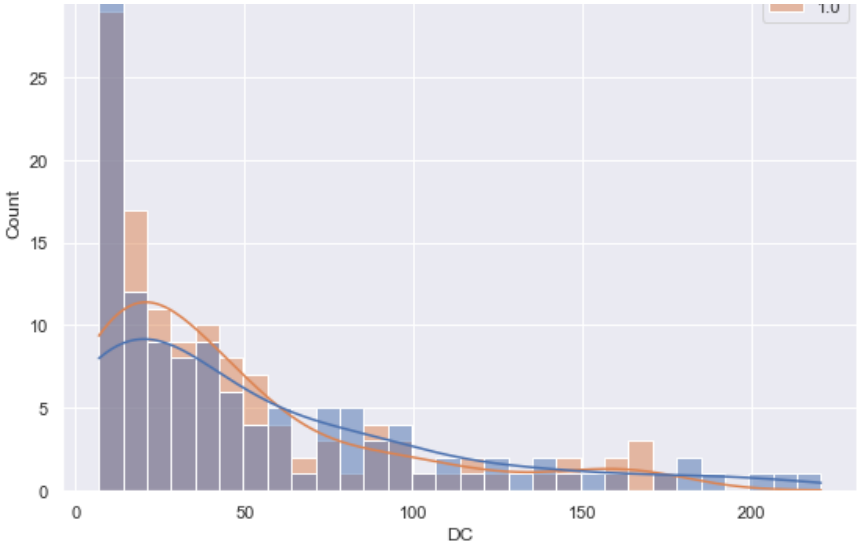

Rain Vs Region
Region
0.0
1.0
Count
Rain

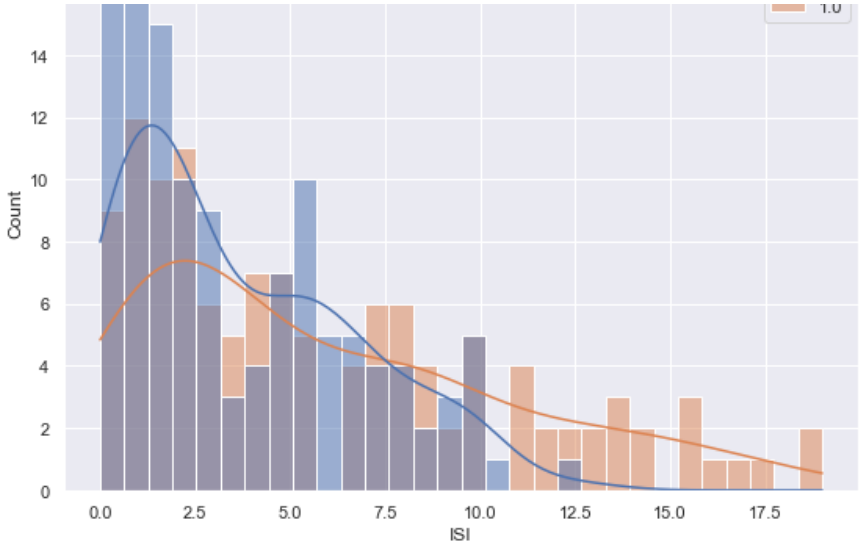

FFMC Vs Region
Region
0.0
1.0
Count
FFMC

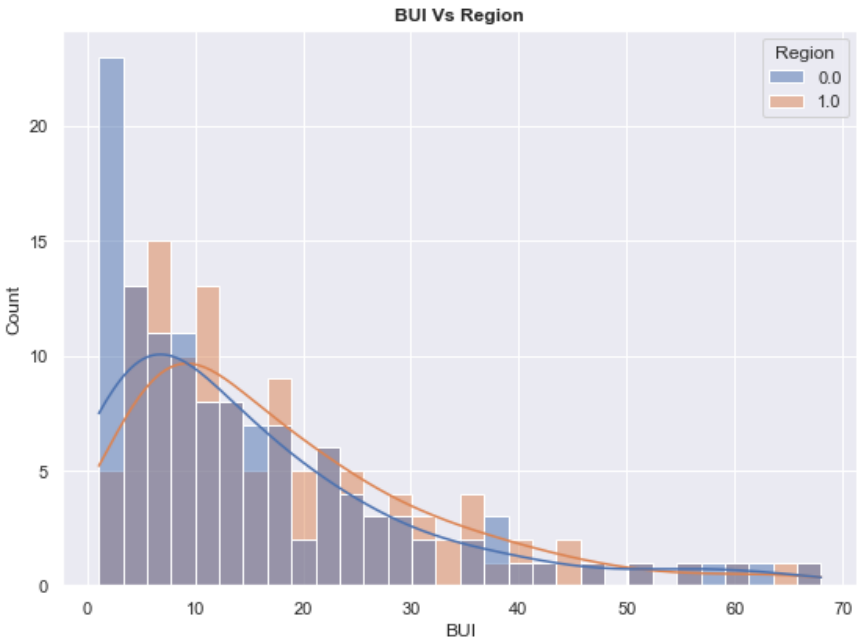

DMC Vs Region
Region
0.0
1.0
Count
DMC

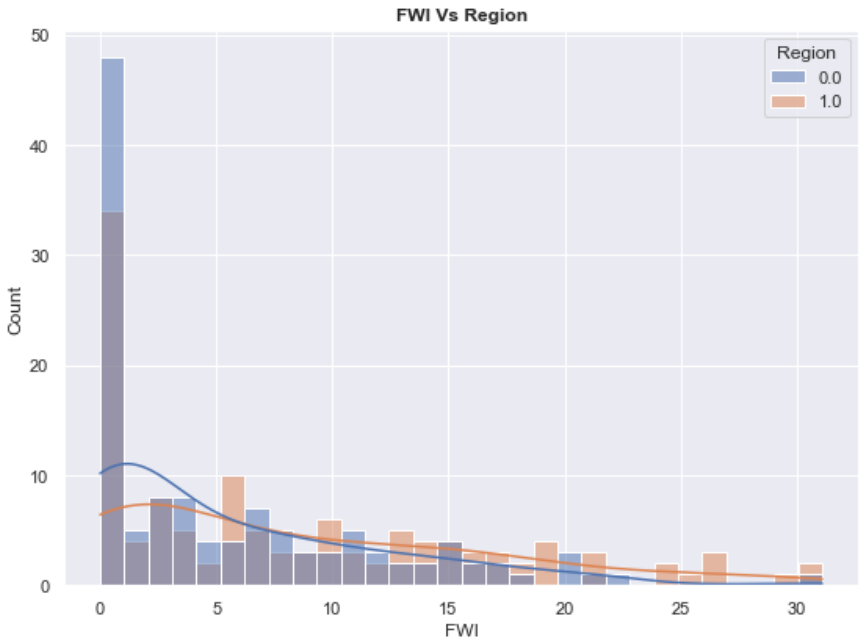

DC Vs Region
Region
0.0


ISI Vs Region
Region
0.0





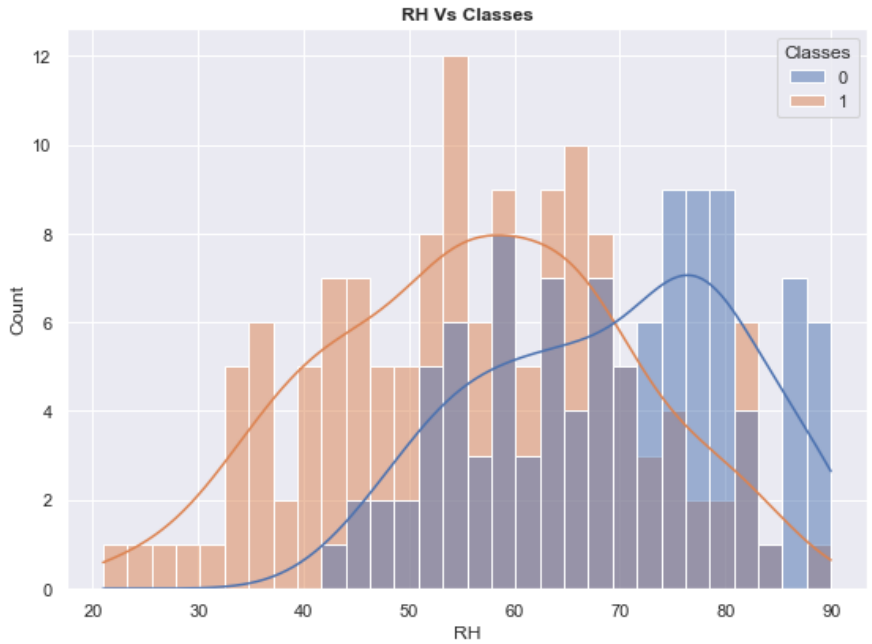


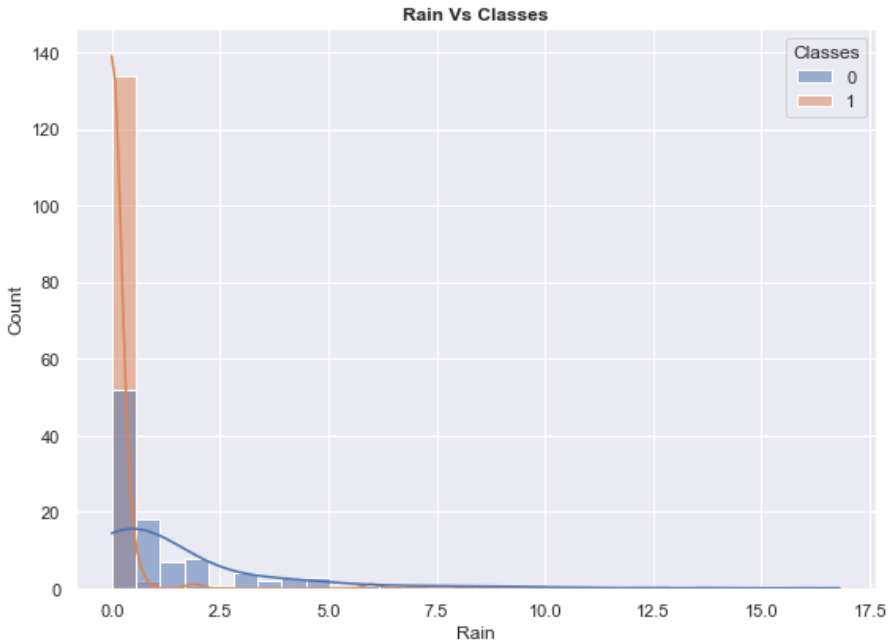


3.4.3 Comparing Continuous numerical features with Classes

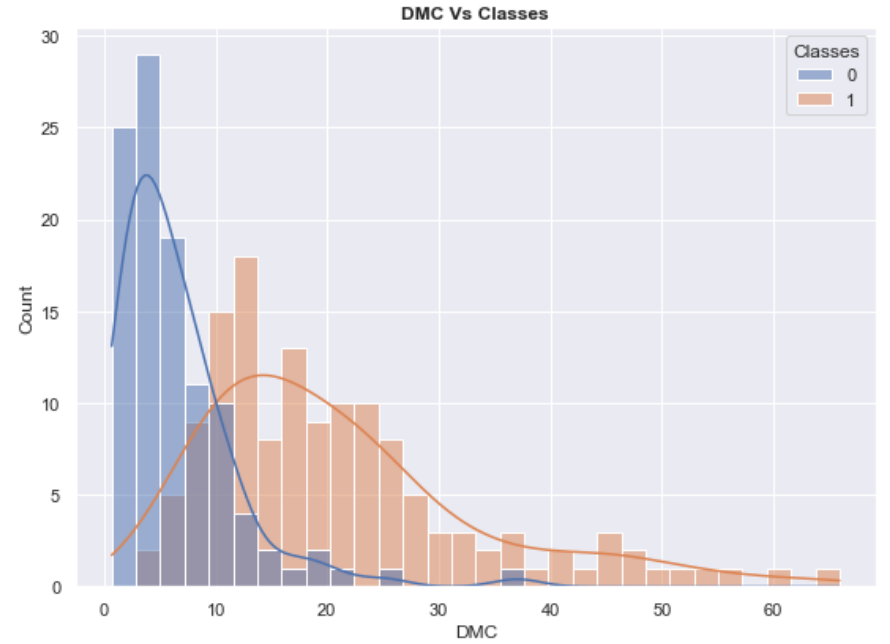
In [31]:
```
### Comparing Continuous numerical features with Classes
```

```
plt.figure(figsize=(20,30))
for i in enumerate(continuous_features):
    plt.subplot(4, 2, i[0]+1)
    sns.set(rc={'figure.figsize':(7,7)})
    sns.histplot(data=dataset, x=i[1], kde=True, bins=30, color='blue', hue='Classes')
    plt.title("{} Vs Classes".format(i[1]),fontweight="bold")
```

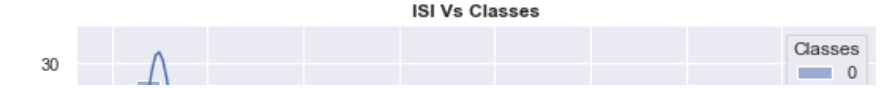

RH Vs Classes
Classes
0
1
Count
RH

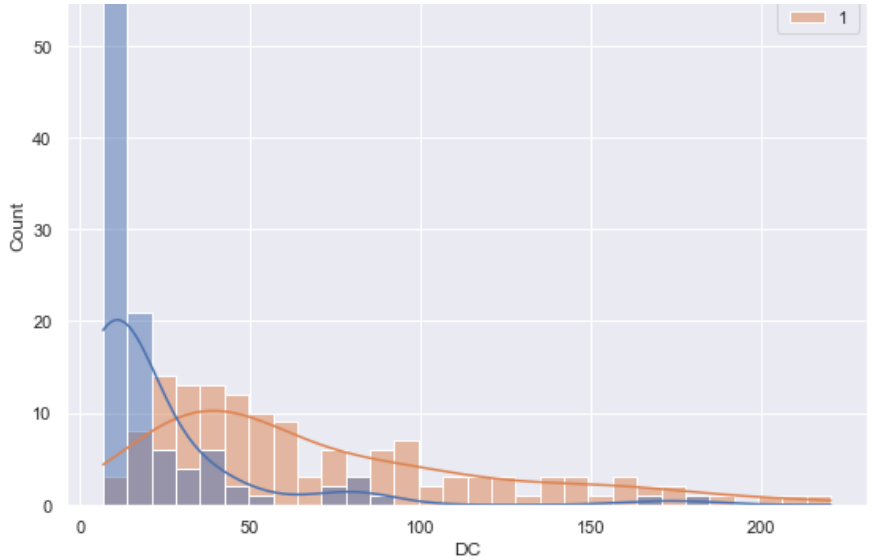

Rain Vs Classes
Classes
0
1
Count
Rain

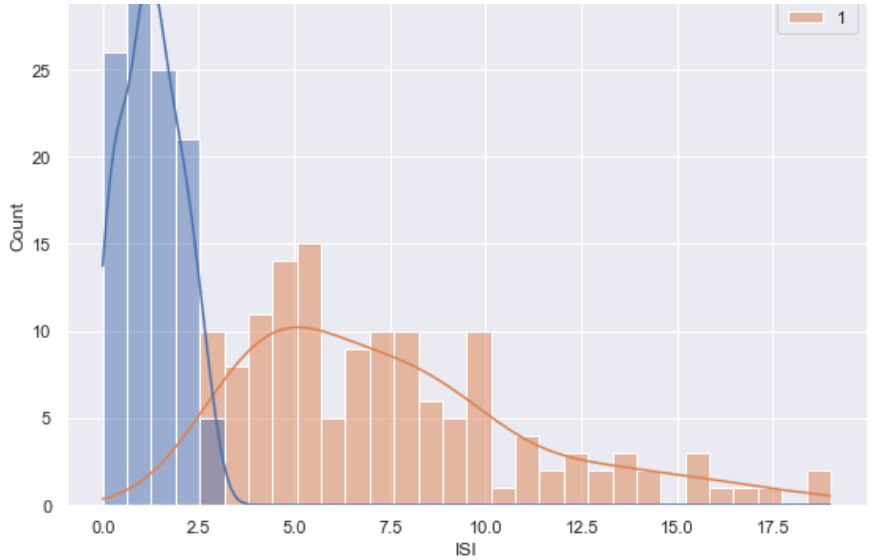

FFMC Vs Classes
Classes
0
1
Count
FFMC

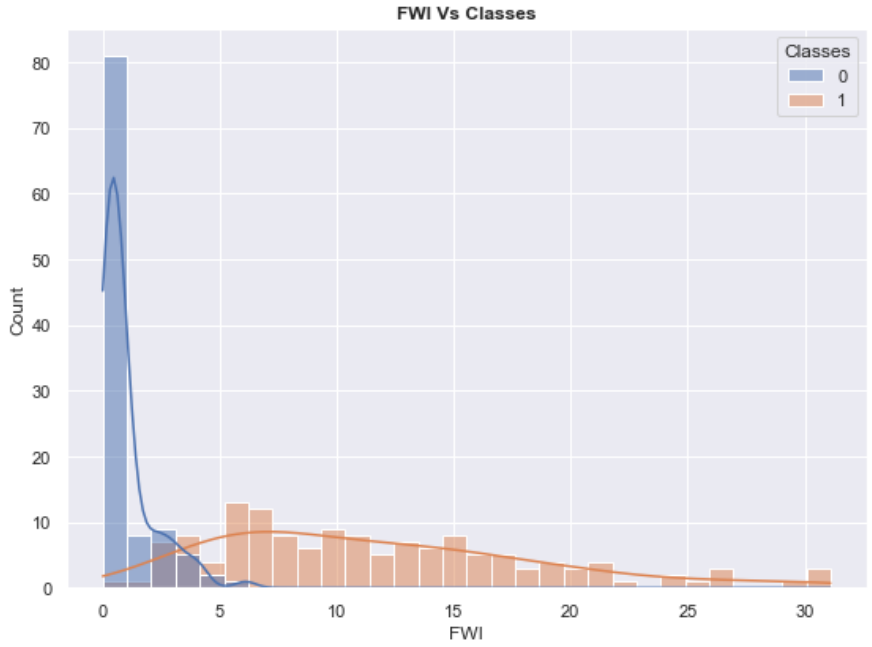

DMC Vs Classes
Classes
0
1
Count
DMC


DC Vs Classes
Classes
0


ISI Vs Classes
Classes
0



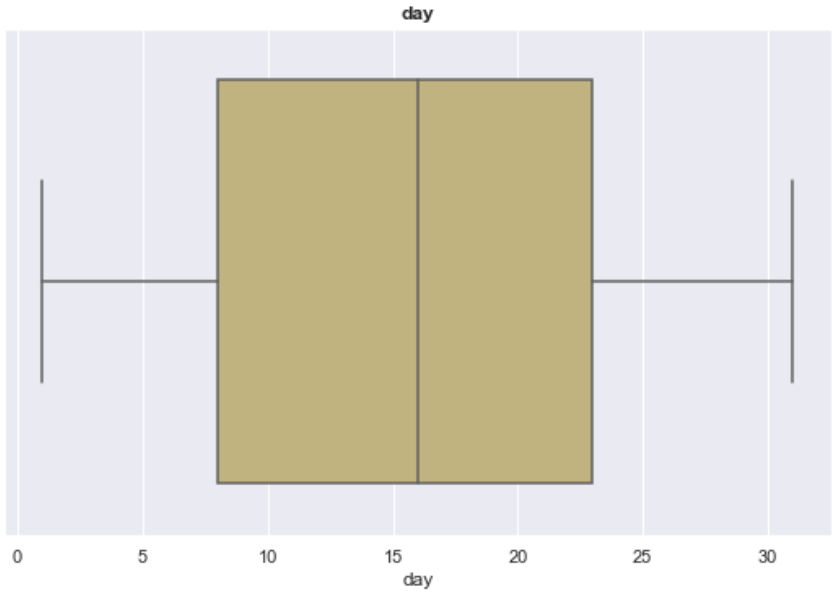


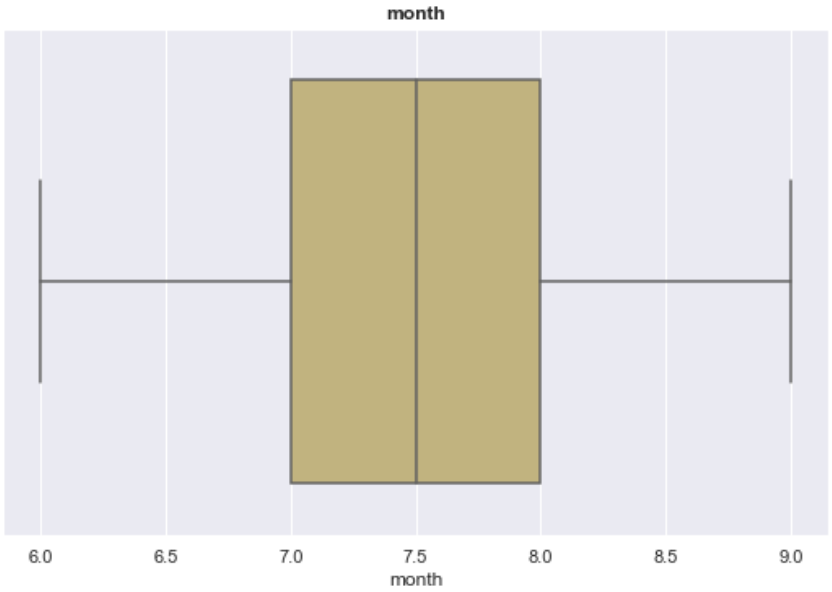


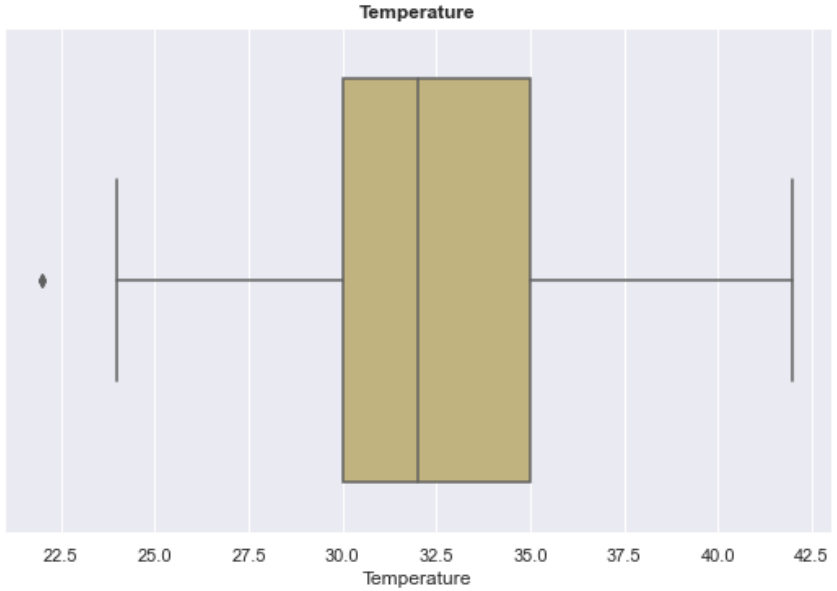
3.5 Checking Outliers

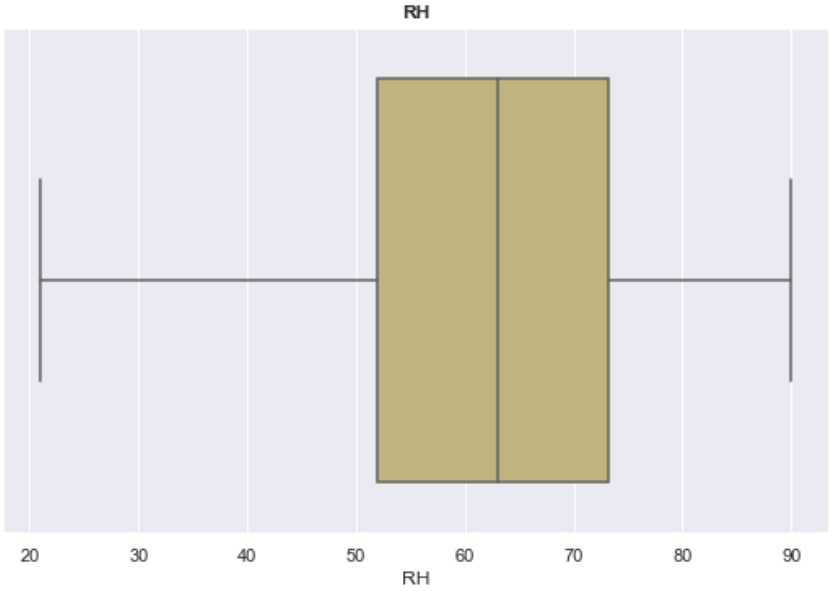
In [32]:
```
### Checking outliers in numerical features
```

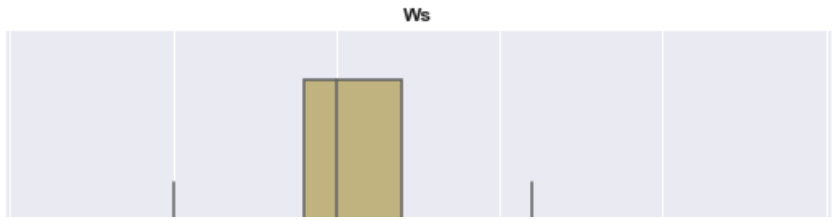
```
plt.figure(figsize=(20,40))
for i in enumerate(numerical_features):
    plt.subplot(6, 2, i[0]+1)
    sns.set(rc={'figure.figsize':(7,7)})
    sns.boxplot(data=dataset, x=i[1], color='y')
    plt.title("{}".format(i[1]), fontweight="bold")
```


day
0
5
10
15
20
25
30
day

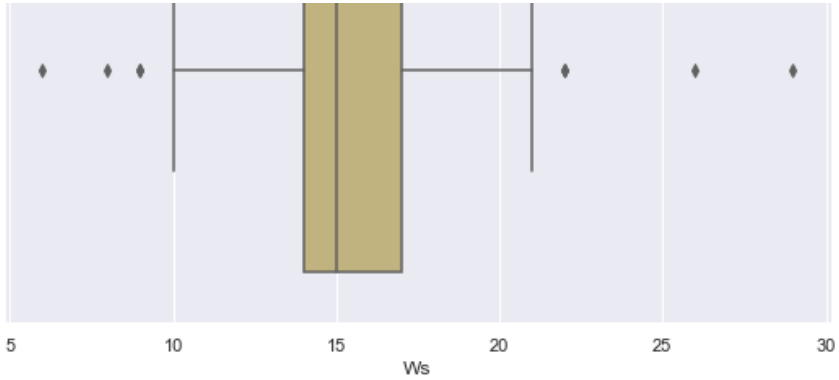

month
6.0
6.5
7.0
7.5
8.0
8.5
9.0
month

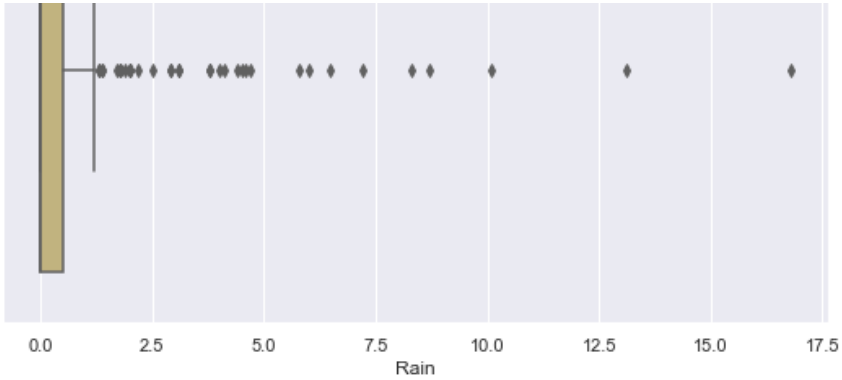

Temperature
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
42.5
Temperature

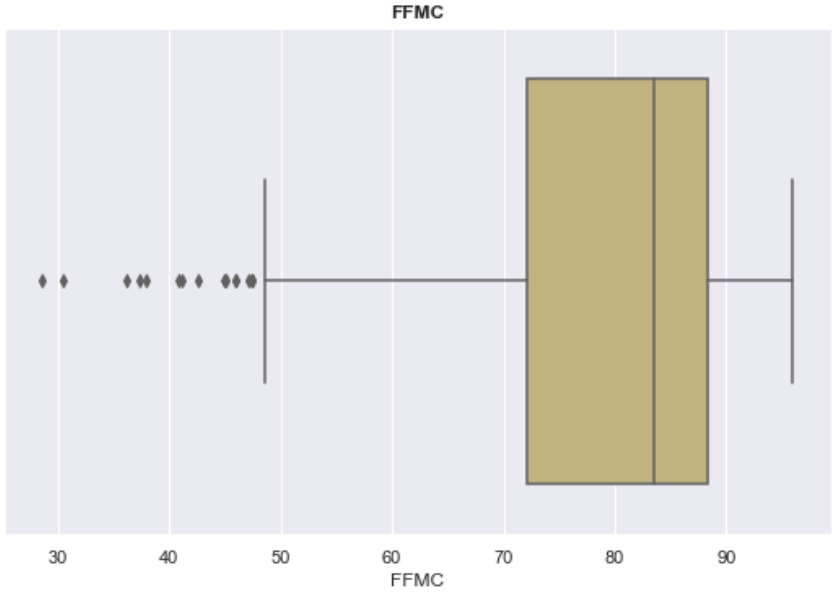

RH
20
30
40
50
60
70
80
90
RH

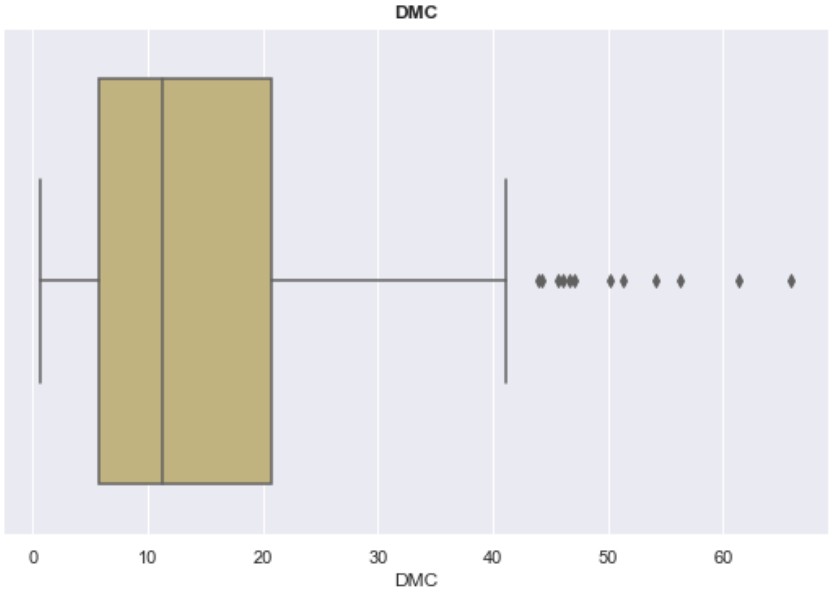

Ws

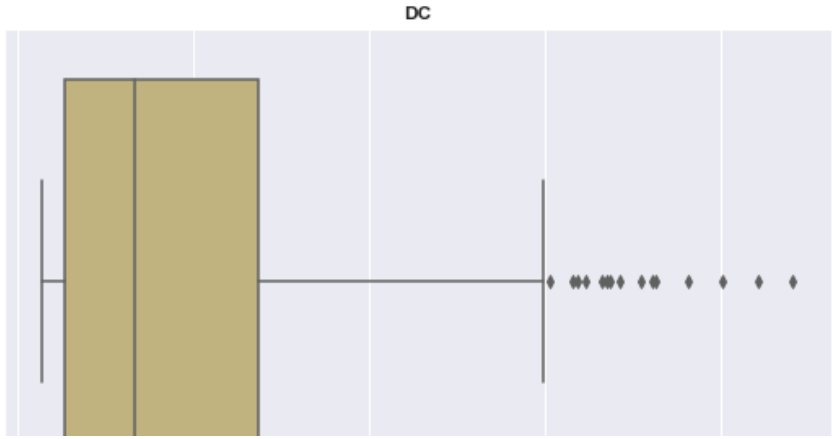

Rain

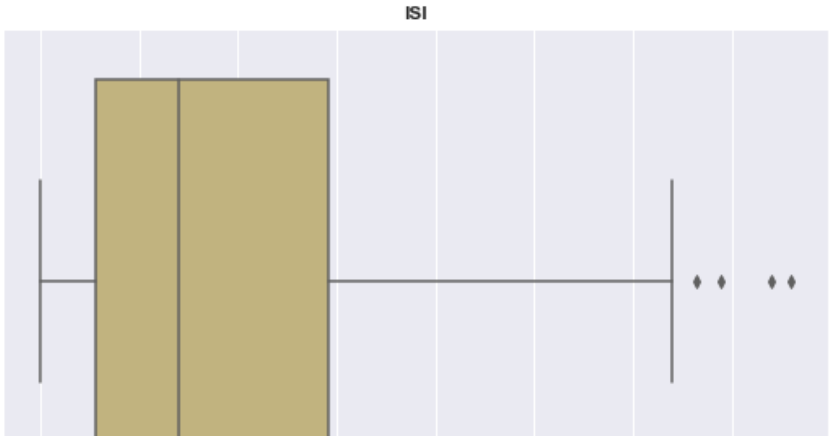

5
10
15
20
25
30
Ws

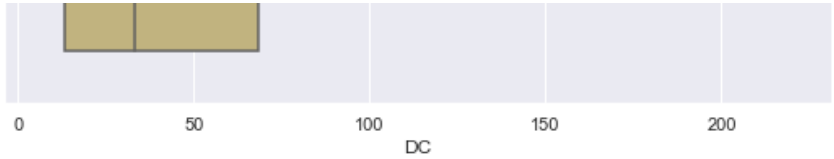

0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Rain

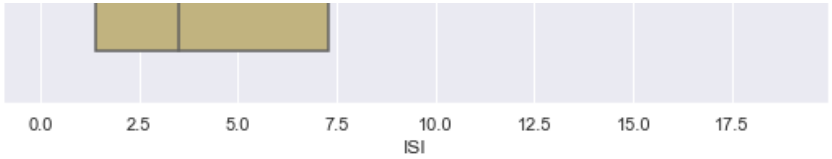

FFMC
30
40
50
60
70
80
90
FFMC

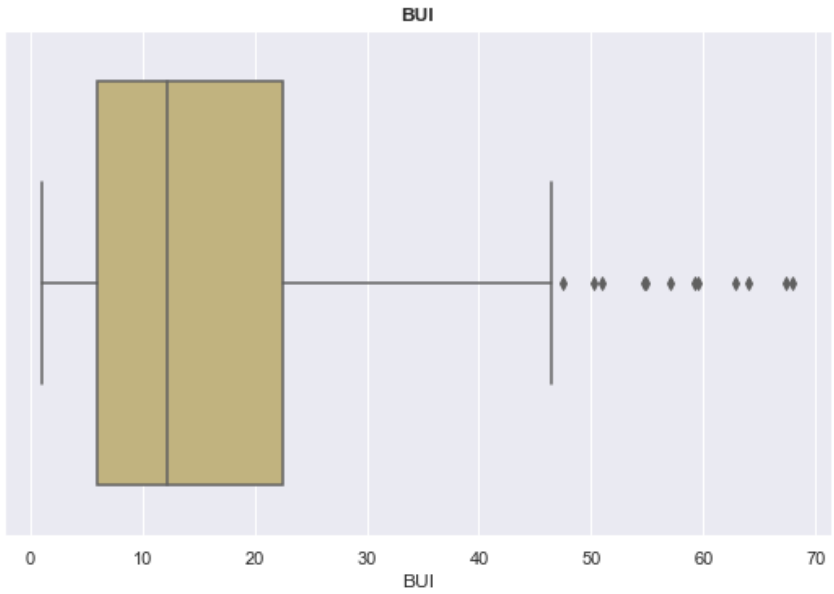

DMC
0
10
20
30
40
50
60
DMC

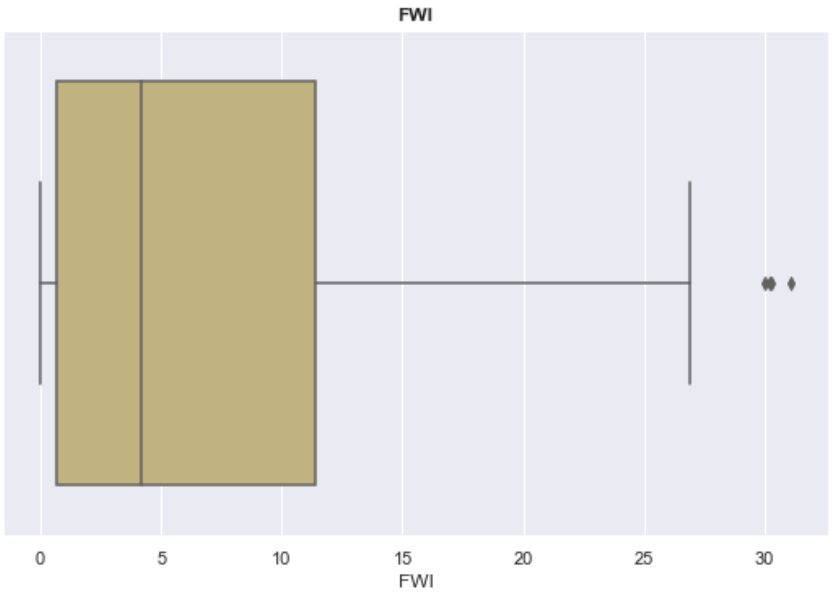

DC


ISI









Observation

1. Relative Humidity, RH feature doesnt have outliers.
2. Temperature and FFMC have outliers in lower boundary side.
3. Wind Speed, Ws has outliers on both sides(Upper and lower boundary).
4. Rain, DMC,DC, ISI, BUI and FWI have outilers in upper boundary side.

4.0 Correlation between each Numerical features

In [196…

```
corr= round(dataset[numerical_features+['Classes']].corr(),2)
corr
```

Out[196]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
day	1.00	0.00	0.10	-0.07	0.05	-0.11	0.22	0.49	0.53	0.18	0.52	0.35	0.20
month	0.00	1.00	-0.06	-0.04	-0.04	0.04	0.02	0.07	0.13	0.06	0.09	0.08	0.02
Temperature	0.10	-0.06	1.00	-0.65	-0.28	-0.33	0.68	0.48	0.37	0.61	0.46	0.56	0.52
RH	-0.07	-0.04	-0.65	1.00	0.24	0.22	-0.65	-0.41	-0.22	-0.69	-0.35	-0.57	-0.44
Ws	0.05	-0.04	-0.28	0.24	1.00	0.17	-0.16	-0.00	0.08	0.02	0.03	0.03	-0.07
Rain	-0.11	0.04	-0.33	0.22	0.17	1.00	-0.54	-0.29	-0.30	-0.35	-0.30	-0.32	-0.38
FFMC	0.22	0.02	0.68	-0.65	-0.16	-0.54	1.00	0.60	0.50	0.74	0.59	0.69	0.77
DMC	0.49	0.07	0.48	-0.41	-0.00	-0.29	0.60	1.00	0.88	0.67	0.98	0.87	0.58
DC	0.53	0.13	0.37	-0.22	0.08	-0.30	0.50	0.88	1.00	0.50	0.94	0.74	0.51
ISI	0.18	0.06	0.61	-0.69	0.02	-0.35	0.74	0.67	0.50	1.00	0.64	0.91	0.74
BUI	0.52	0.09	0.46	-0.35	0.03	-0.30	0.59	0.98	0.94	0.64	1.00	0.86	0.58
FWI	0.35	0.08	0.56	-0.57	0.03	-0.32	0.69	0.87	0.74	0.91	0.86	1.00	0.71
Classes	0.20	0.02	0.52	-0.44	-0.07	-0.38	0.77	0.58	0.51	0.74	0.58	0.71	1.00

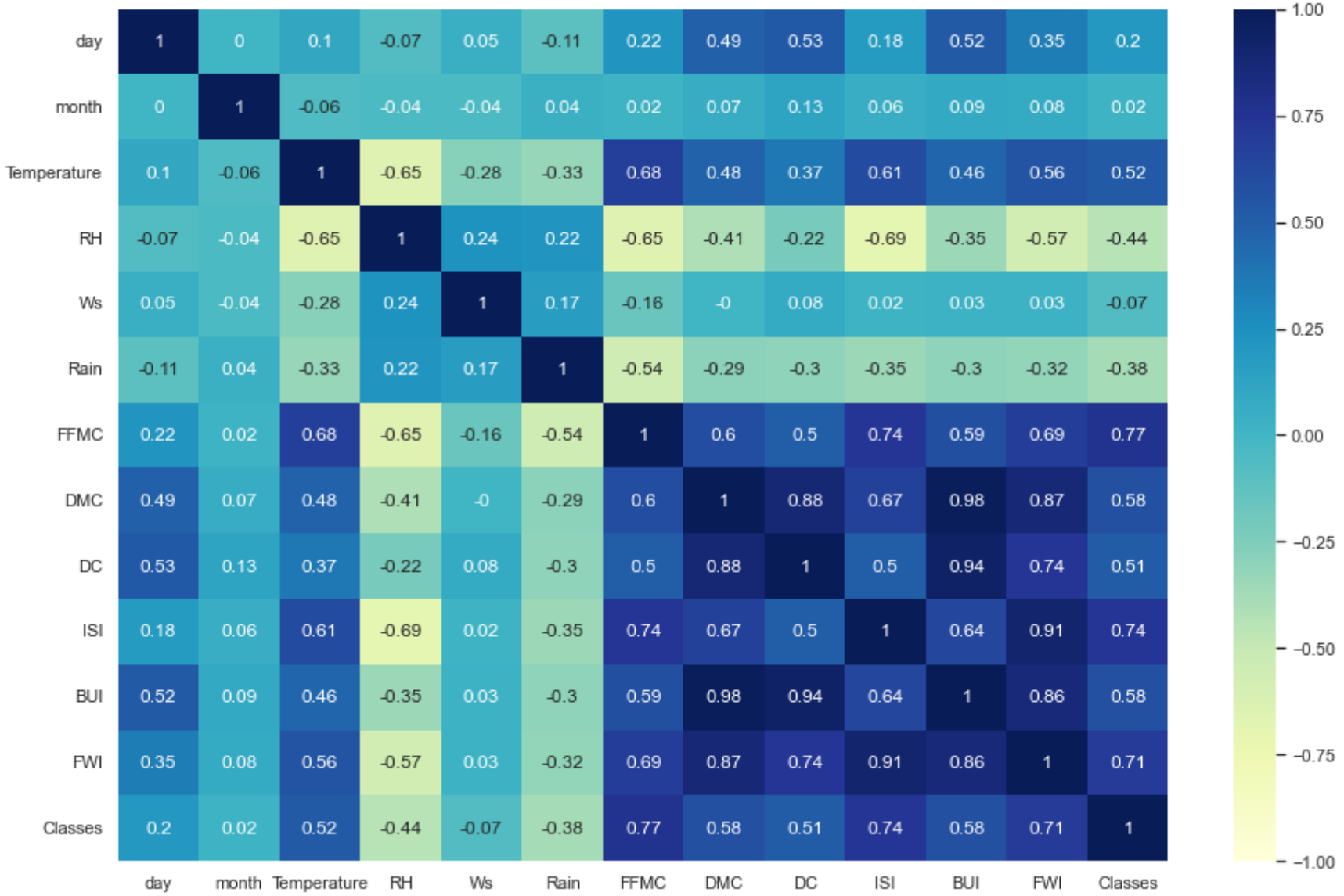
4.1 Heatmap to visualise the Correlation

In [197…

```
### Plotting heatmap for visualising the correlation between features
sns.set(rc={'figure.figsize':(15,10)})
sns.heatmap(data=corr, annot=True,  vmin=-1, vmax=1, cmap="YlGnBu")
```

Out[197]:

```
<AxesSubplot:>
```



Note (For both positive and negative side)

1. Correlation coefficients between 0.9 and 1.0, very highly correlated.
2. Correlation coefficients between 0.7 and 0.9, highly correlated.
3. Correlation coefficients between 0.5 and 0.7, moderately correlated.

4. Correlation coefficients between 0.3 and 0.5, low correlation.
5. Correlation coefficients less than 0.3, little correlation

Observations

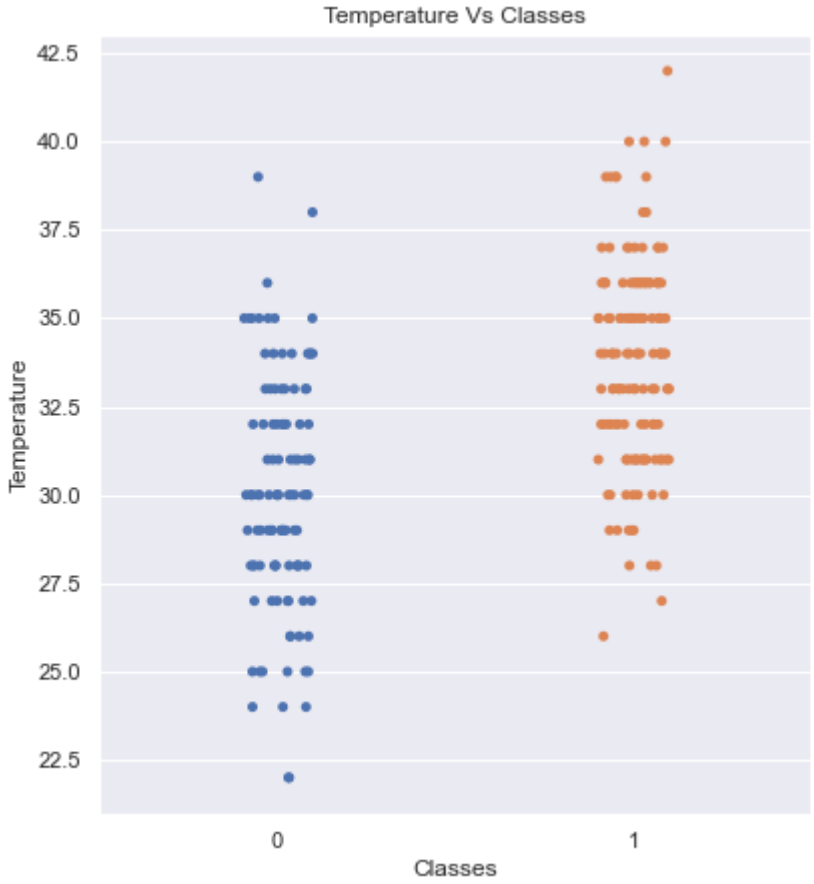
1. Very highly Correlated features: DMC-BUI, DC-BUI, ISI-FWI
2. Highly correlated features: FFMC-ISI, DC-DMC, FWI-DMC, FWI-DC, FWI-BUI

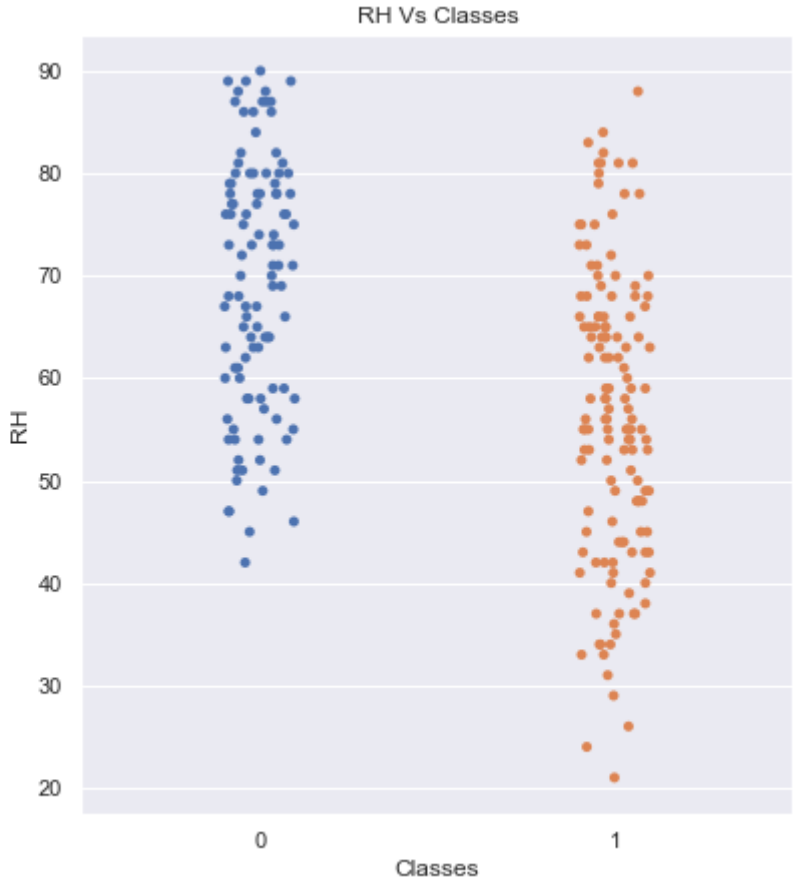
Note: Features with very hihg and high correlation are more linearly dependent and hence have almost the same effect on the dependent variable. So, we can drop one of the two features.

4.2 Relationship between Numerical Feature and Target Feature

4.2.1 Strip Plot

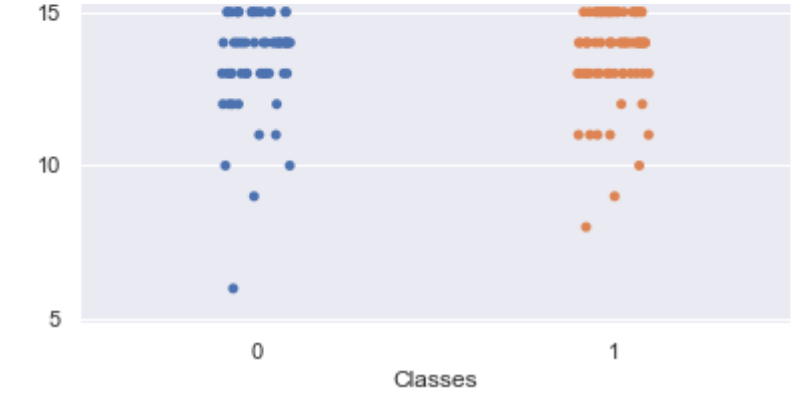
In [35]:
```
num_feature_custom=[feature for feature in numerical_features if feature not in ['day', 'month']]

plt.figure(figsize=(14,50))
for i in enumerate(num_feature_custom):
    plt.subplot(6, 2, i[0]+1)
    sns.set(rc={'figure.figsize':(7,8)})
    sns.stripplot(data=dataset, y=i[1], x='Classes')
    plt.title("{} Vs Classes".format(i[1]))
```

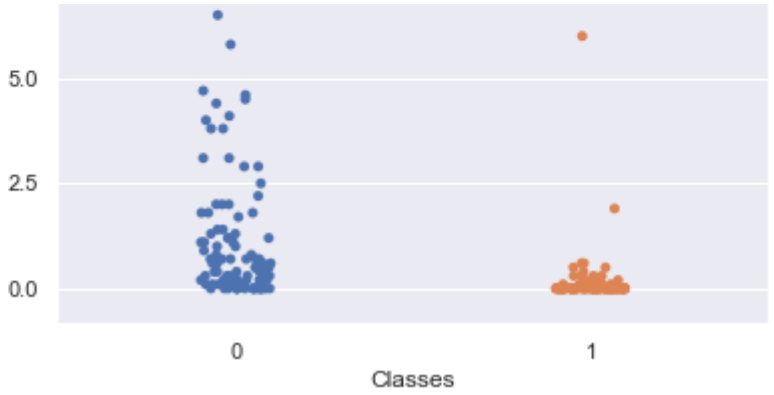

Temperature Vs Classes
Temperature
42.5
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
0
1
Classes


RH Vs Classes
RH
90
80
70
60
50
40
30
20
0
1
Classes


Ws Vs Classes
Ws
30
25
20

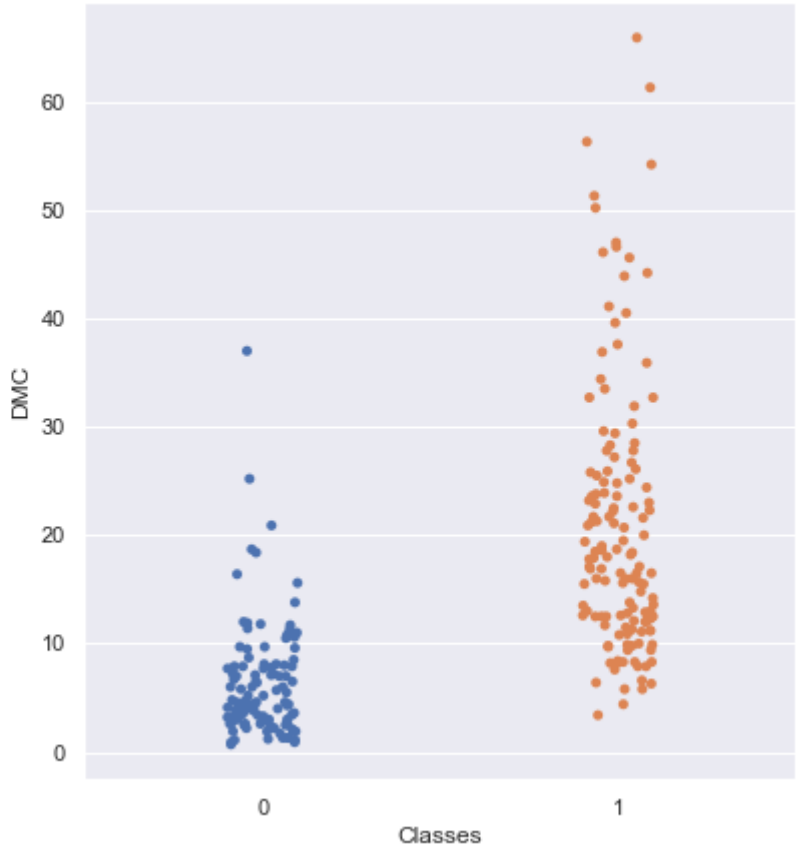

Rain Vs Classes
Rain
17.5
15.0
12.5
10.0
7.5





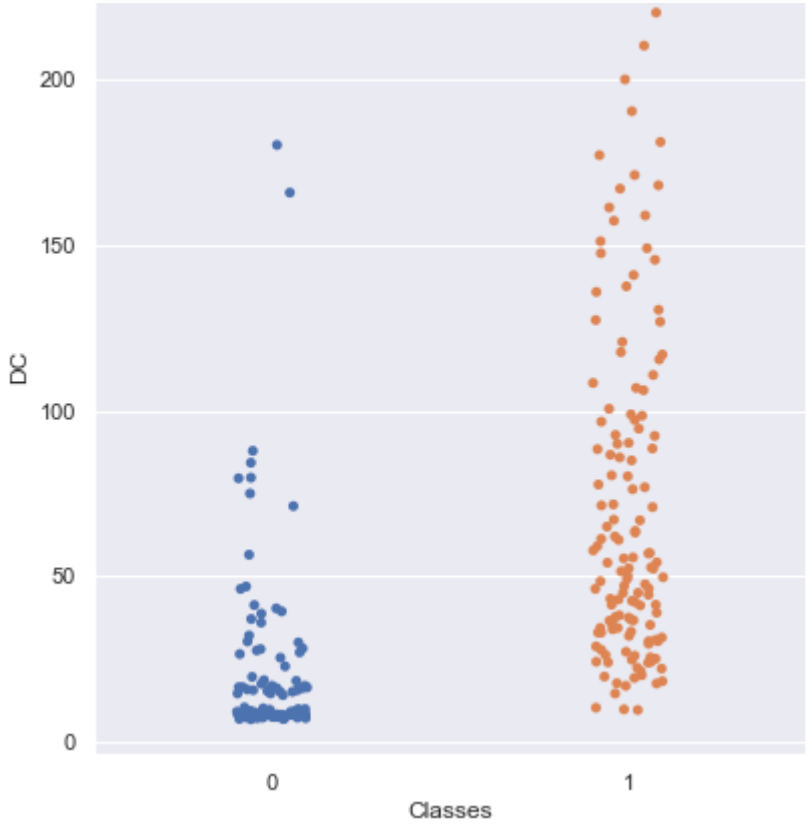
FFMC Vs Classes

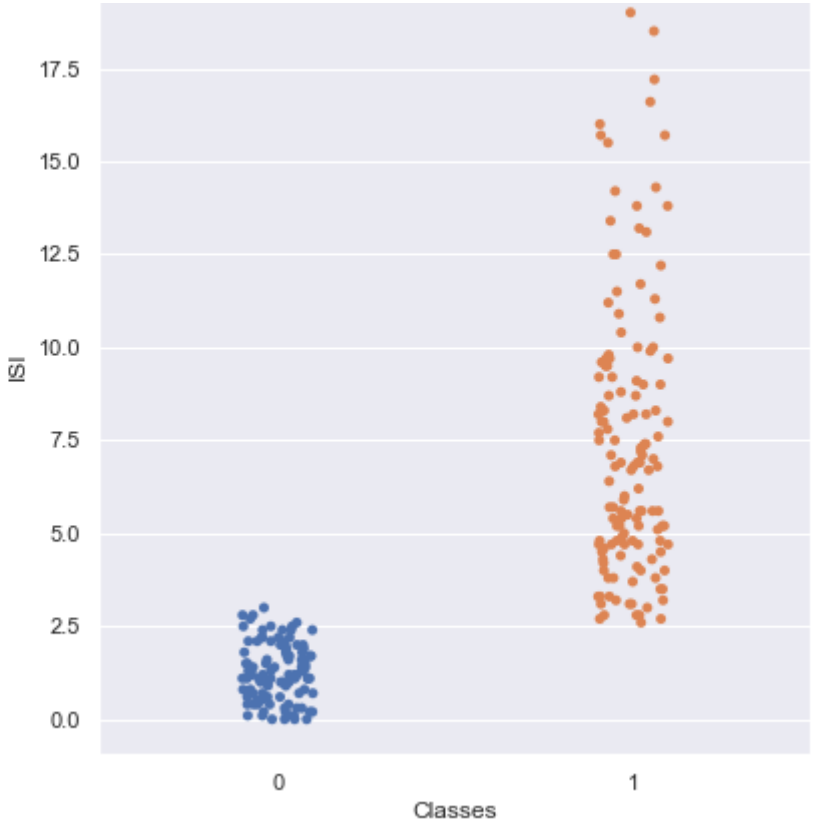
DMC Vs Classes

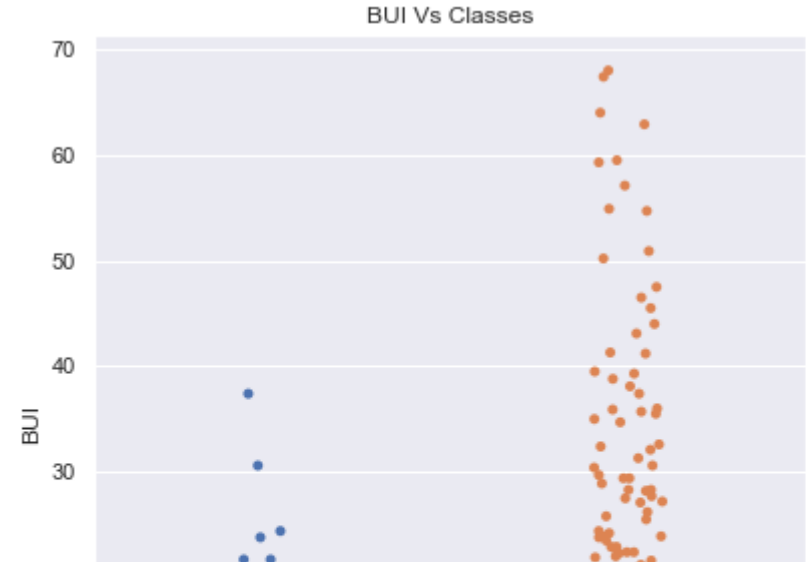


DC Vs Classes

ISI Vs Classes


DC
200
150
100
50
0
0
1
Classes


ISI
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
1
Classes


BUI Vs Classes
BUI
70
60
50
40
30

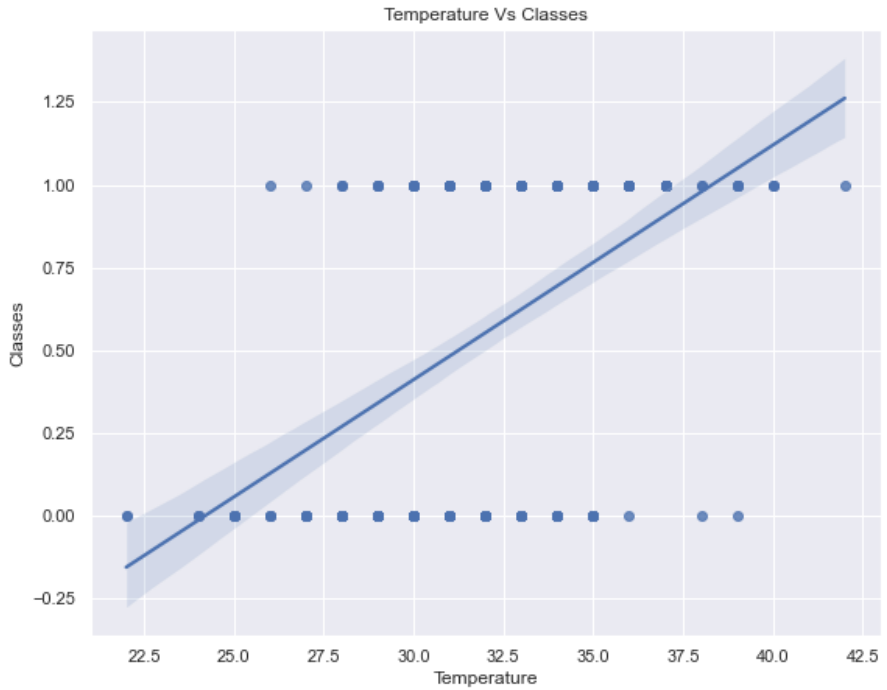

FWI Vs Classes
FWI
30
25
20
15
10

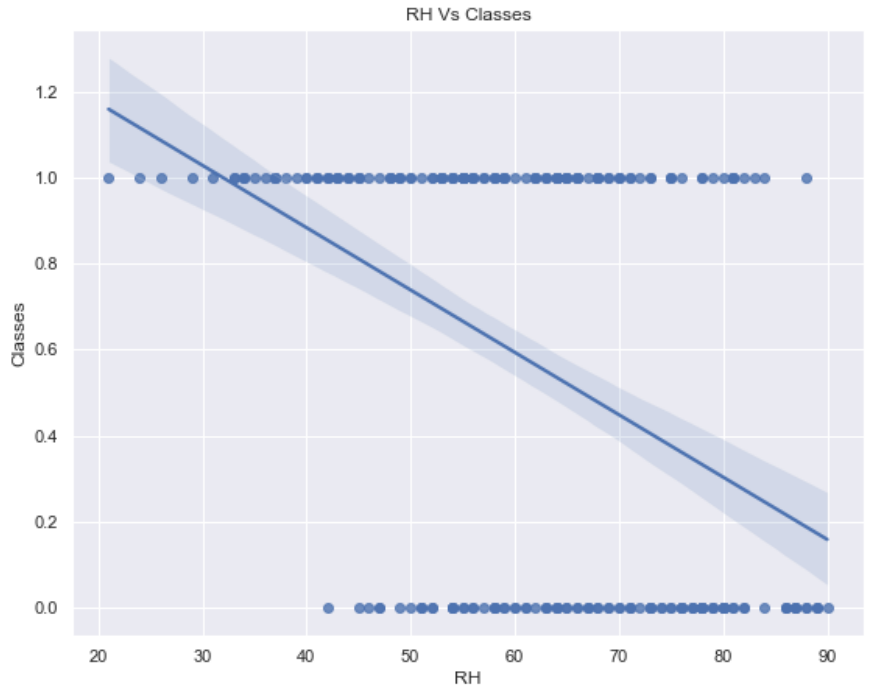
Observation

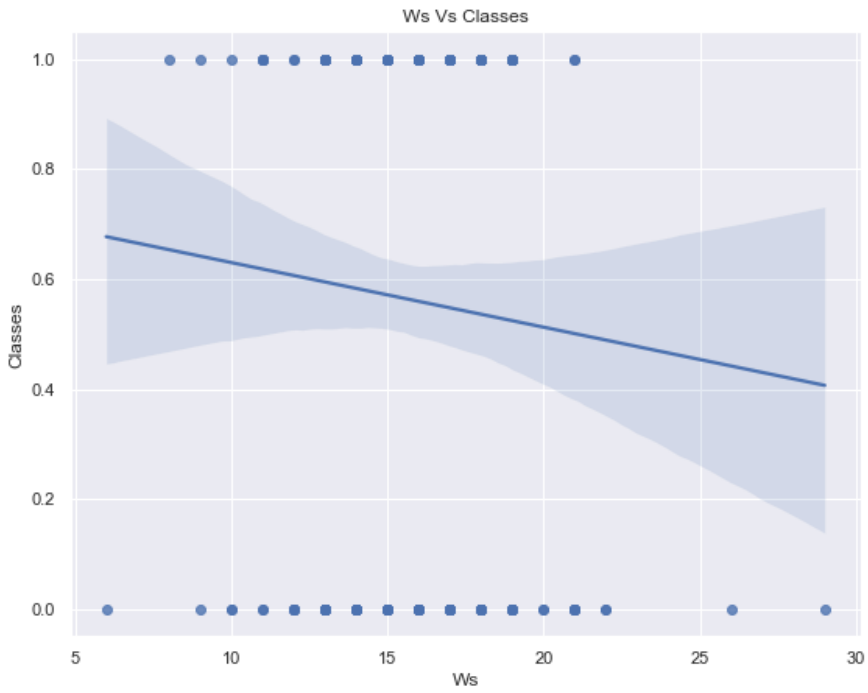
1. It is visible that for temperature between 30 to 37, there is most no of cases of occurance of fire, i.e Hot regions are more prone to forrest fires.
2. For RH 40 to 70 and wind speed between 13 to 19 Km/h, Most no of cases of occurance of fire is reported, i.e dry regions are more prone to forrest fires.
3. Almost all cases of occurance of fire is for region having rain less than 1 mm, i.e dry regions are more prone to forrest fires.
4. For FFMC(Fine Fuel Moisture Code) greater than 80, almost all cases of fire is reported.
5. DMC (Duff Moisture Code) >30 and DC (Drought code) >100, almost all cases of occurance of fire reported, this means drought affected areas are more prone to forrest fires.

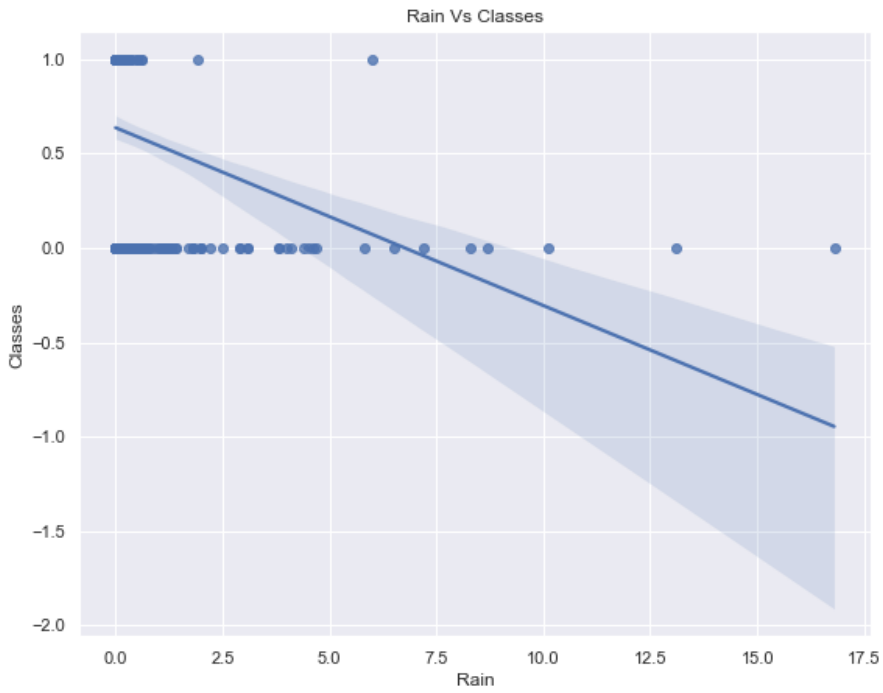
4.2.2 Regplot Plot

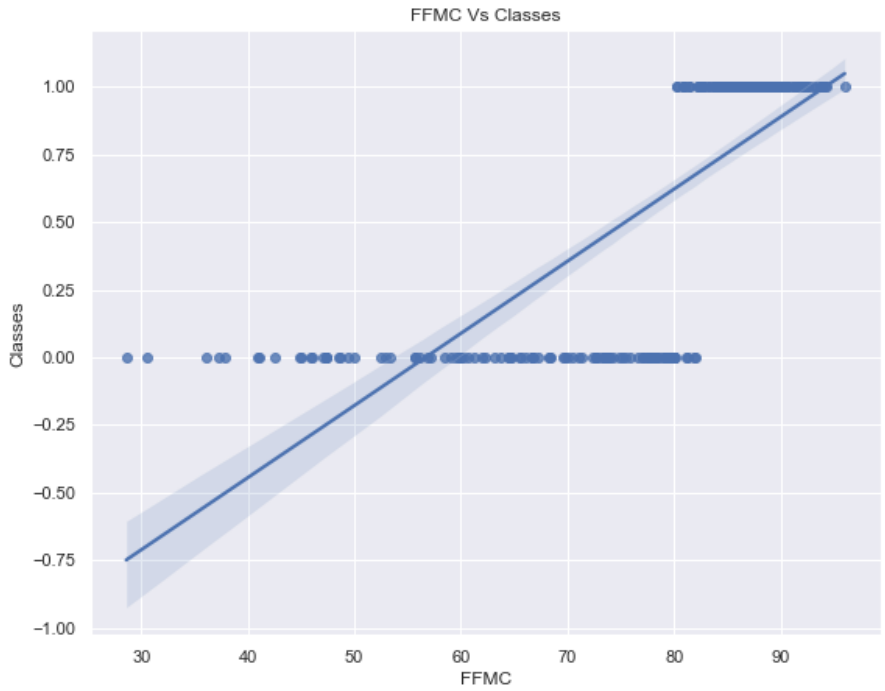
In [36]:
```
plt.figure(figsize=(20,50))
for i in enumerate(num_feature_custom):
    plt.subplot(6, 2, i[0]+1)
    sns.set(rc={'figure.figsize':(8,10)})
    sns.regplot(data=dataset, x=i[1], y='Classes')
    plt.xlabel(i[1])
    plt.ylabel("Classes")
    plt.title("{} Vs Classes".format(i[1]))
```

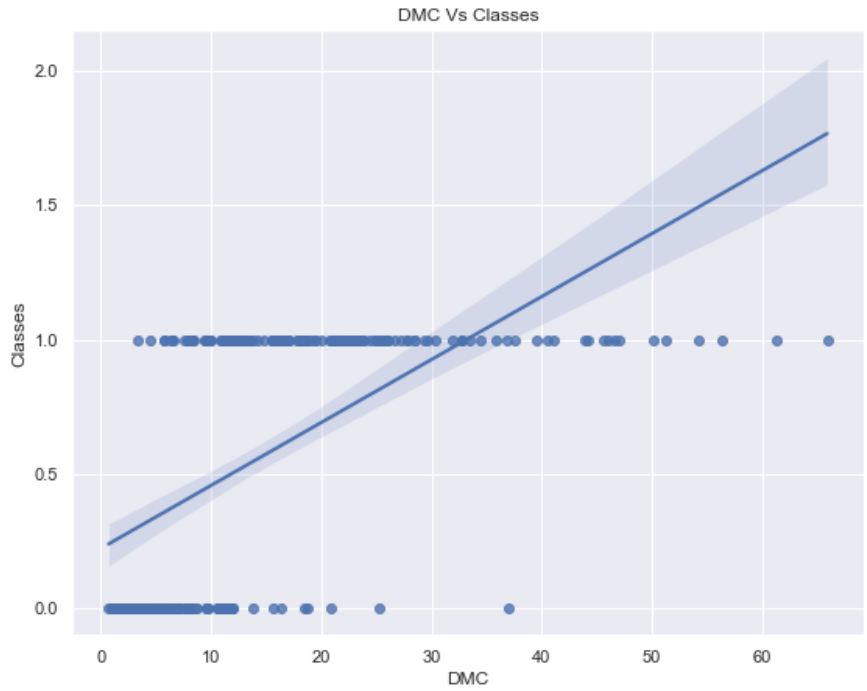

Temperature Vs Classes
Classes
Temperature
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
42.5

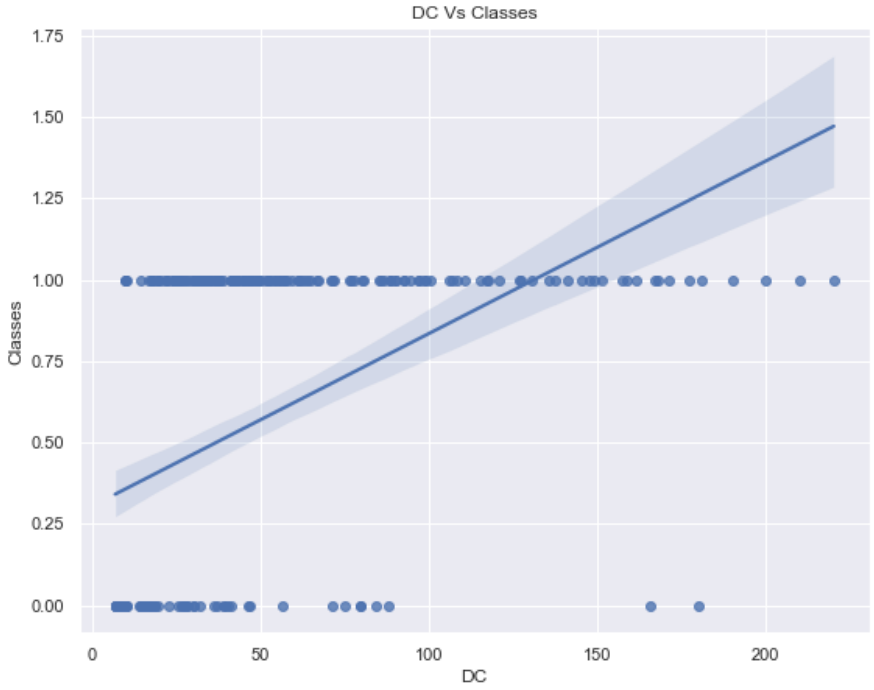

RH Vs Classes
Classes
RH
1.2
1.0
0.8
0.6
0.4
0.2
0.0
20
30
40
50
60
70
80
90

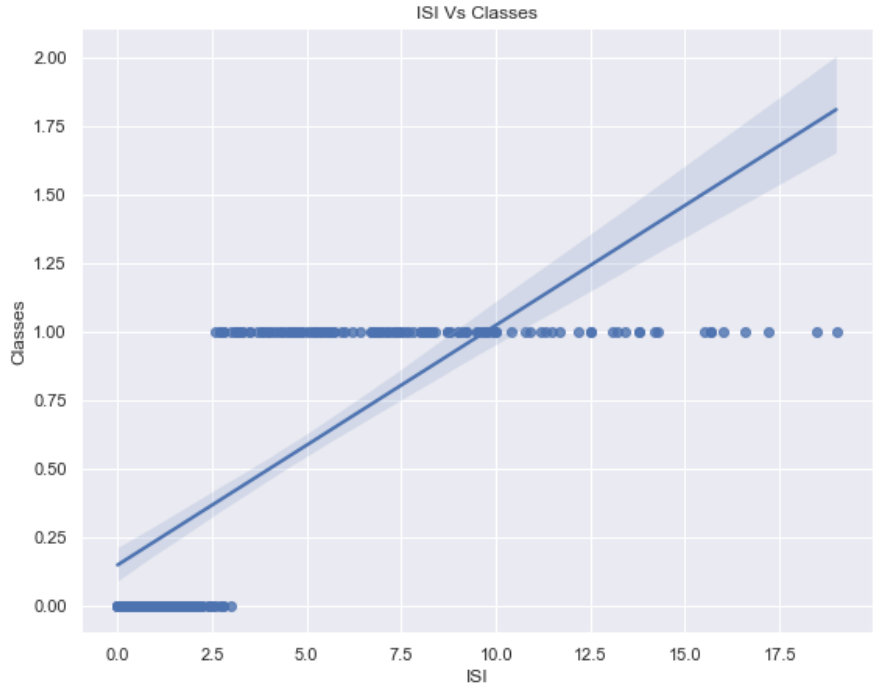

Ws Vs Classes
Classes
Ws
1.0
0.8
0.6
0.4
0.2
0.0
5
10
15
20
25
30

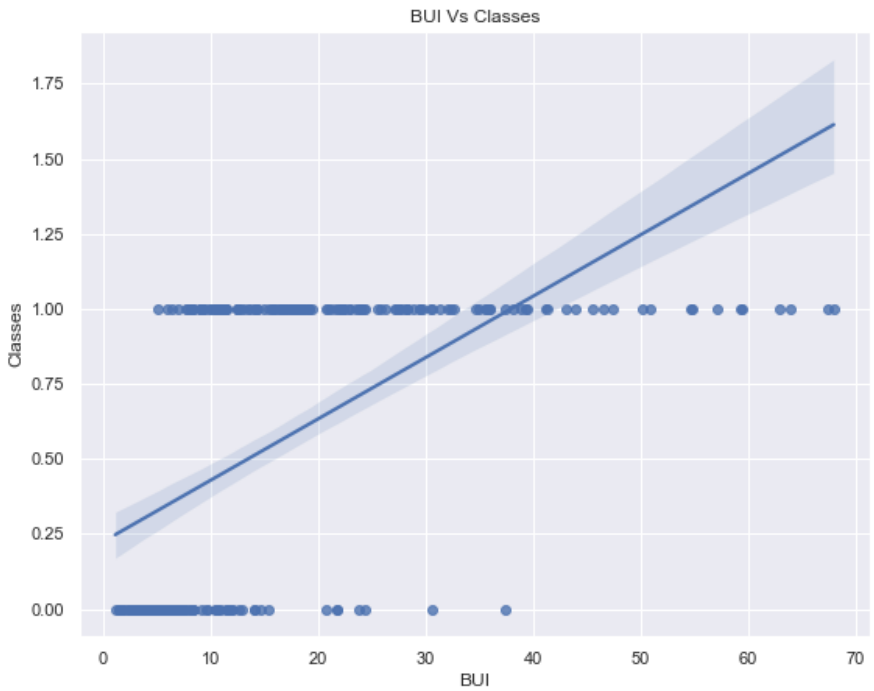

Rain Vs Classes
Classes
Rain
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

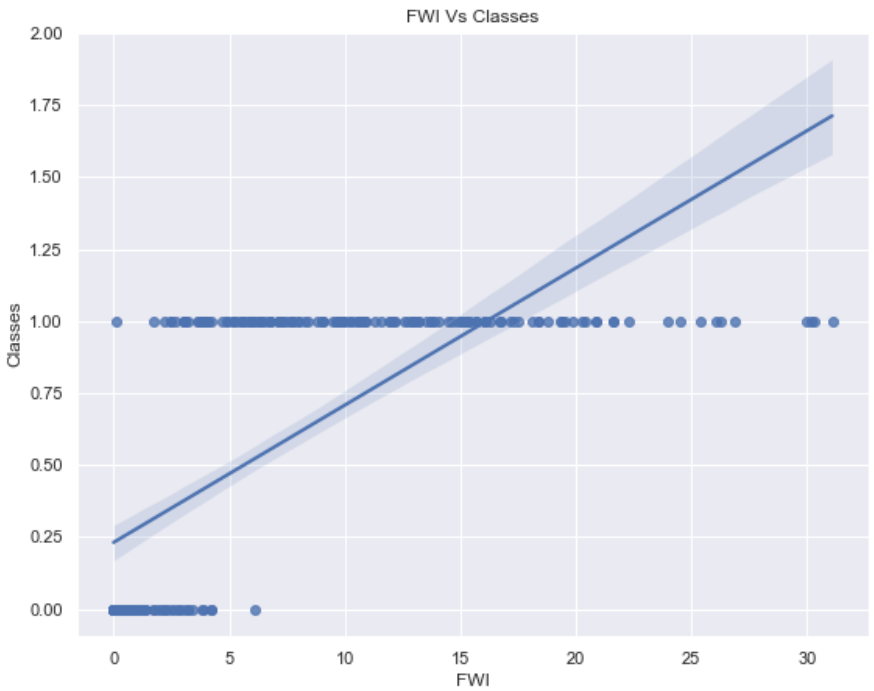

FFMC Vs Classes
Classes
FFMC
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
30
40
50
60
70
80
90


DMC Vs Classes
Classes
DMC
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
50
60


DC Vs Classes
Classes
DC
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
50
100
150
200


ISI Vs Classes
Classes
ISI
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5





Final Report

1. Very highly Correlated features: DMC-BUI, DC-BUI, ISI-FWI
2. Highly correlated features: FFMC-ISI, DC-DMC, FWI-DMC, FWI-DC, FWI-BUI
3. Temperature between 30 to 37 degree celcius have most no of cases of occurance of fire.
4. Wind speed between 13 to 19 Km/hr range there is most no of occurance of fire.
5. Almost all cases of occurance of fire is for region having rain less than 1 mm, i.e dry regions are more prone to forrest fires.
6. For FFMC(Fine Fuel Moisture Code) greater than 80, almost all cases of fire is reported.
7. DMC (Duff Moisture Code) >30 and DC (Drought code) >100, almost all cases of occurance of fire reported, this means drought affected areas are more prone to forrest fires.
8. In Bejaia region, the no of cases of occurance of fire is less compared to no of cases of occurance of no fire.
9. In Sidi Bel-abbes region the no of cases of occurance of fire is more compared to no fire.
10. Also Overall no of cases of occurance of fire is more in Sidi Bel-abbes region as compared to Bejaia region.
11. Most no of cases of fire occured are in the month of august and least no of cases of fire occured is in month of september.

12. July and august have more cases of fire as compared to no fire.
13. June and september have more cases of no fire as compared to fire.
14. Relative Humidity, RH feature doesnt have outliers whereas Temperature, FFMC, wind speed, Rain, DMC,DC, ISI, BUI and FWI have outilers.
15. There is no null vales in dataset.

5.0 Model Building

5.1 Getting independent features in dataset(X) and dependent feature in series(y)

In [37]:
```
### Creating copy of dataset
data=dataset.copy()
```

In [38]:
```
data['Class']=data['Classes']
```

In [39]:
```
data.drop('Classes', axis=1, inplace=True)
```

In [40]:
```
data['Classes']=data['Class']
data.drop('Class', axis=1, inplace=True)
```

In [41]:
```
data.head()
```

Out[41]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	1	6	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0.0	0
1	2	6	29	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0.0	0
2	3	6	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0.0	0
3	4	6	25	89	13	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0.0	0
4	5	6	27	77	16	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0.0	0

In [42]:
```
X=data.iloc[:,:-1]
y=data.iloc[:,-1]

X.head()
```

Out[42]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
0	1	6	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0.0
1	2	6	29	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0.0
2	3	6	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0.0
3	4	6	25	89	13	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0.0
4	5	6	27	77	16	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0.0

In [43]:
```
y.head()
```

Out[43]:
```
0    0
1    0
2    0
3    0
4    0
Name: Classes, dtype: int64
```

5.2 Splitting data into Training and Test data

In [44]:
```
### random state train test split will be same with all people using random_state=16

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=16)

X_train.head()
```

Out[44]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
216	3	9	28	75	16	0.0	82.2	4.4	24.3	3.3	6.0	2.5	1.0
199	17	8	42	24	9	0.0	96.0	30.3	76.4	15.7	30.4	24.0	1.0
156	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0
175	24	7	33	63	17	1.1	72.8	20.9	56.6	1.6	21.7	2.5	1.0
228	15	9	32	51	13	0.0	88.7	16.0	50.2	6.9	17.8	9.8	1.0

In [45]:
```
y_train.head()
```

Out[45]:
```
216    1
199    1
156    1
175    0
228    1
Name: Classes, dtype: int64
```

In [46]:
```
X_test.head()
```

Out[46]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
72	12	8	35	51	13	0.3	81.3	15.6	75.1	2.5	20.7	4.2	0.0
149	28	6	37	37	13	0.0	92.5	27.2	52.4	11.7	27.1	18.4	1.0
29	30	6	33	50	14	0.0	88.7	22.9	92.8	7.2	28.3	12.9	0.0
191	9	8	39	43	12	0.0	91.7	16.5	30.9	9.6	16.4	12.7	1.0
10	11	6	31	65	14	0.0	84.5	12.5	54.3	4.0	15.8	5.6	0.0

In [47]:
```
y_test.head()
```

Out[47]:
```
72     0
149    1
29     1
191    1
10     1
Name: Classes, dtype: int64
```

In [48]:
```
### both will have same shape
X_train.shape, y_train.shape
```

Out[48]:
```
((170, 13), (170,))
```

In [49]:
```
### both will have same shape
X_test.shape, y_test.shape
```

Out[49]:
```
((74, 13), (74,))
```

5.3 Standardisation/ Feature scaling the dataset

In [50]:
```
### Creating a standard scaler object
scaler=StandardScaler()
scaler
```

Out[50]:
```
StandardScaler
StandardScaler()
```

In [51]:
```
### using fit_transform to Standardize the train data
X_train=scaler.fit_transform(X_train)
X_train
```

Out[51]:
```
array([[-1.46667507,  1.32918909, -1.20880747, ..., -0.74565569,
        -0.62033172,  0.95399809],
       [ 0.12870275,  0.44306303,  2.71075899, ...,  0.96241799,
         2.25514379,  0.95399809],
       [-1.23876395, -0.44306303,  0.47100673, ...,  0.14338266,
         1.05145637,  0.95399809],
       ...,
       [ 1.610125  ,  1.32918909, -2.04871457, ..., -0.99766657,
        -0.94131503, -1.04822013],
       [ 1.04034721,  1.32918909, -1.20880747, ..., -0.17863123,
        -0.5534602 ,  0.95399809],
       [ 0.24265831, -0.44306303,  0.19103769, ...,  0.50739836,
         0.26237239,  0.95399809]])
```

In [52]:
```
### here using transform only to avoid data leakage
### (training mean and training std will be used for standardisation when we use transform)
X_test=scaler.transform(X_test)
X_test
```

Out[52]:
```
array([[-4.41075042e-01,  4.43063032e-01,  7.50975758e-01,
        -7.32557183e-01, -9.32937185e-01, -2.16223446e-01,
         2.27131581e-01,  6.22267177e-02,  5.59740154e-01,
        -5.58457335e-01,  2.83388695e-01, -3.92968541e-01,
        -1.04822013e+00],
       [ 1.38221389e+00, -1.32918909e+00,  1.31091382e+00,
        -1.64681313e+00, -9.32937185e-01, -3.58512423e-01,
         9.91597781e-01,  1.00182157e+00,  8.34900180e-02,
         1.54517290e+00,  7.31408021e-01,  1.50618273e+00,
         9.53998092e-01],
       [ 1.61012500e+00, -1.32918909e+00,  1.91037693e-01,
        -7.97861179e-01, -5.60638308e-01, -3.58512423e-01,
         7.32225321e-01,  6.53523476e-01,  9.31089379e-01,
         5.16223326e-01,  8.15411645e-01,  7.70595968e-01,
        -1.04822013e+00],
       [-7.82941717e-01,  4.43063032e-01,  1.87085189e+00,
        -1.25498915e+00, -1.30523606e+00, -3.58512423e-01,
         9.36993053e-01,  1.35126318e-01, -3.67583899e-01,
         1.06499643e+00, -1.76242897e-02,  7.43847358e-01,
         9.53998092e-01],
       [-5.55030601e-01, -1.32918909e+00, -3.68900372e-01,
         1.81698766e-01, -5.60638308e-01, -3.58512423e-01,
         4.45550495e-01, -1.88871906e-01,  1.23352364e-01,
        -2.15474145e-01, -5.96261015e-02, -2.05728275e-01,
        -1.04822013e+00],
       [ 1.15430277e+00, -1.32918909e+00, -3.68900372e-01,
         1.16394770e-01,  9.28557198e-01, -3.58512423e-01,
         6.02539090e-01,  2.40425741e-01,  4.90505553e-01,
         4.01895596e-01,  3.46391413e-01,  4.62986960e-01,
        -1.04822013e+00],
       [ 3.56613864e-01,  4.43063032e-01,  7.50975758e-01,
        -1.42132227e-02,  1.30085608e+00, -3.58512423e-01,
         7.80004458e-01,  6.77823342e-01,  1.52063250e+00,
         1.08786198e+00,  1.02542070e+00,  1.34569107e+00,
        -1.04822013e+00],
       [-2.13163926e-01, -1.32918909e+00, -1.48877650e+00,
         1.09595472e+00,  1.83959445e-01, -2.65048095e-02,
        -1.67720833e+00, -6.82969197e-01, -8.62716199e-01,
        -1.01576825e+00, -7.38655392e-01, -9.14566424e-01,
         9.53998092e-01],
       [-8.96897275e-01,  1.32918909e+00, -6.48869405e-01,
         7.04130737e-01,  5.56258322e-01,  6.83545087e-02,
        -1.09020750e+00, -9.90767509e-01, -8.39637999e-01,
        -8.78574978e-01, -9.55664753e-01, -9.01192120e-01,
```

```
 -1.04822013e+00],
[-1.69458618e+00, -1.32918909e+00, -9.28838438e-01,
 -3.40733205e-01,  9.28557198e-01, -3.58512423e-01,
 -8.37660627e-01, -9.25967864e-01, -8.56422144e-01,
 -8.32843886e-01, -9.27663546e-01, -8.87817815e-01,
 -1.04822013e+00],
[ 1.61012500e+00,  4.43063032e-01,  4.71006726e-01,
 -8.63165175e-01, -1.88339432e-01, -3.58512423e-01,
  7.66353276e-01,  8.07422632e-01,  2.32207546e+00,
  7.22013240e-01,  1.33343399e+00,  1.18519941e+00,
  9.53998092e-01],
[-1.12480839e+00, -4.43063032e-01,  7.50975758e-01,
 -1.32029315e+00, -1.88339432e-01, -2.16223446e-01,
  4.59201677e-01,  5.41267621e-02, -6.96653121e-02,
 -1.46877507e-01,  3.37661624e-03, -1.12108142e-01,
  9.53998092e-01],
[-3.27119484e-01,  1.32918909e+00, -2.04871457e+00,
  1.55308269e+00,  2.04545383e+00,  1.82325190e+00,
 -2.53040721e+00, -1.09606693e+00, -8.58520163e-01,
 -1.10723044e+00, -1.03966838e+00, -9.54689338e-01,
 -1.04822013e+00],
[-1.23876395e+00,  1.32918909e+00, -9.28838438e-01,
  8.34738730e-01,  1.83959445e-01, -3.58512423e-01,
  1.93003625e-01, -9.25967864e-01, -5.12347156e-01,
 -4.89860697e-01, -8.08658412e-01, -7.27326159e-01,
 -1.04822013e+00],
[-6.68986159e-01, -4.43063032e-01,  1.91037693e-01,
  4.42914752e-01, -9.32937185e-01, -2.65048095e-02,
 -7.76230308e-01, -7.15369019e-01, -8.20755835e-01,
 -8.78574978e-01, -7.59656298e-01, -8.87817815e-01,
 -1.04822013e+00],
[ 8.12436096e-01, -4.43063032e-01, -1.48877650e+00,
  2.47002763e-01,  2.41775271e+00, -1.68793787e-01,
 -6.67020850e-01, -3.50871017e-01,  4.80015462e-01,
 -7.18516157e-01, -8.76273093e-02, -6.73828940e-01,
 -1.04822013e+00],
[-1.23876395e+00, -1.32918909e+00, -8.89313398e-02,
 -1.44821215e-01, -5.60638308e-01, -2.63653105e-01,
 -5.95432445e-02, -7.15369019e-01, -6.46620323e-01,
 -7.18516157e-01, -7.10654185e-01, -8.34320596e-01,
  9.53998092e-01],
[-1.46667507e+00, -4.43063032e-01, -8.89313398e-02,
  9.00042726e-01,  1.67315495e+00, -2.65048095e-02,
 -1.01512600e+00, -9.90767509e-01, -8.22853853e-01,
```

```
 -8.32843886e-01, -9.55664753e-01, -8.87817815e-01,
 -1.04822013e+00],
[ 1.15430277e+00, -4.43063032e-01,  7.50975758e-01,
 -2.75429208e-01, -2.04983382e+00, -2.63653105e-01,
  2.23638484e-02, -3.26571151e-01, -6.02561940e-01,
 -7.64247249e-01, -4.16641502e-01, -8.20946291e-01,
  9.53998092e-01],
[-1.12480839e+00,  4.43063032e-01, -6.48869405e-01,
 -5.36645194e-01, -5.60638308e-01,  1.11180701e+00,
 -5.10032256e-01, -3.10371239e-01, -8.24951871e-01,
 -8.32843886e-01, -4.30642106e-01, -8.47694901e-01,
  9.53998092e-01],
[ 1.26825833e+00, -4.43063032e-01,  1.03094479e+00,
 -9.28469172e-01, -9.32937185e-01, -3.58512423e-01,
  8.41434778e-01,  5.96823787e-01,  1.26047824e+00,
  8.59206516e-01,  8.92414966e-01,  1.09157928e+00,
 -1.04822013e+00],
[ 1.61012500e+00,  4.43063032e-01,  7.50975758e-01,
  5.08218748e-01,  5.56258322e-01,  2.09248496e-02,
 -3.59869252e-01,  8.39822454e-01,  2.76895334e+00,
 -7.41381703e-01,  1.45243912e+00, -3.92968541e-01,
 -1.04822013e+00],
[ 6.98480538e-01, -4.43063032e-01, -1.20880747e+00,
  1.09595472e+00,  9.28557198e-01, -3.11082764e-01,
 -3.12090114e-01,  1.27026362e-01,  6.60445028e-01,
 -7.18516157e-01,  3.53391715e-01, -5.80208807e-01,
 -1.04822013e+00],
[ 5.84524980e-01, -1.32918909e+00, -8.89313398e-02,
 -4.71341197e-01, -5.60638308e-01, -3.58512423e-01,
  5.61585544e-01, -5.29070040e-01, -6.29836177e-01,
  1.31813147e-02, -5.91649051e-01, -2.99348408e-01,
  9.53998092e-01],
[-8.96897275e-01, -1.32918909e+00, -6.48869405e-01,
  7.04130737e-01, -1.88339432e-01, -3.58512423e-01,
  5.88887908e-01, -2.21271728e-01, -2.12330551e-01,
  1.50374591e-01, -2.20633047e-01, -5.11370474e-03,
 -1.04822013e+00],
[ 6.98480538e-01,  1.32918909e+00,  1.91037693e-01,
  1.16394770e-01, -9.32937185e-01, -3.58512423e-01,
  7.45876503e-01,  9.12722055e-01,  1.21432184e+00,
  4.93357780e-01,  1.10242403e+00,  8.77590405e-01,
  9.53998092e-01],
[ 1.26825833e+00,  1.32918909e+00, -1.20880747e+00,
  1.61838669e+00, -1.88339432e-01,  1.72839258e+00,
```

```
 -2.51675603e+00, -6.74869241e-01, -8.48030072e-01,
 -1.10723044e+00, -7.31655090e-01, -9.54689338e-01,
  9.53998092e-01],
[-6.68986159e-01, -4.43063032e-01,  4.71006726e-01,
 -7.32557183e-01,  1.83959445e-01,  1.44381462e+00,
 -3.22408802e-02, -5.53369907e-01, -8.16559798e-01,
 -6.72785065e-01, -6.26650561e-01, -7.80823377e-01,
  9.53998092e-01],
[-4.41075042e-01,  1.32918909e+00, -9.28838438e-01,
  1.68369068e+00, -9.32937185e-01, -3.58512423e-01,
 -4.75904300e-01, -9.90767509e-01, -6.67600505e-01,
 -8.55709432e-01, -9.06662640e-01, -8.87817815e-01,
 -1.04822013e+00],
[-9.92083682e-02,  4.43063032e-01,  7.50975758e-01,
 -1.05907716e+00, -9.32937185e-01, -2.16223446e-01,
  4.04596949e-01,  1.67526140e-01,  1.21254346e-01,
 -3.29801875e-01,  1.64383562e-01, -2.19102580e-01,
  9.53998092e-01],
[ 8.12436096e-01,  4.43063032e-01,  1.03094479e+00,
 -6.01949190e-01,  1.83959445e-01, -3.58512423e-01,
  7.86830049e-01,  1.84421695e+00,  2.37242790e+00,
  1.24792080e+00,  2.15946962e+00,  2.02778061e+00,
 -1.04822013e+00],
[-5.55030601e-01,  4.43063032e-01,  7.50975758e-01,
  5.10907736e-02, -9.32937185e-01, -3.58512423e-01,
  7.45876503e-01,  5.56324009e-01,  5.99602500e-01,
  4.93357780e-01,  6.19403189e-01,  6.63601530e-01,
 -1.04822013e+00],
[ 4.70569422e-01, -1.32918909e+00, -6.48869405e-01,
  1.16125871e+00,  1.83959445e-01, -1.68793787e-01,
 -1.24037050e+00, -9.25967864e-01, -4.47308592e-01,
 -9.24306070e-01, -8.08658412e-01, -9.01192120e-01,
 -1.04822013e+00],
[-1.69458618e+00,  4.43063032e-01,  1.03094479e+00,
 -1.12438116e+00, -5.60638308e-01, -3.58512423e-01,
  5.64918038e-02, -8.12568486e-01, -8.01873671e-01,
 -6.72785065e-01, -8.36659620e-01, -8.34320596e-01,
 -1.04822013e+00],
[-1.35271951e+00, -1.32918909e+00, -2.04871457e+00,
  1.74899468e+00, -9.32937185e-01,  8.27229054e-01,
 -3.36995492e+00, -1.09606693e+00, -8.71108272e-01,
 -1.13009598e+00, -1.04666868e+00, -9.54689338e-01,
 -1.04822013e+00],
[-6.68986159e-01, -1.32918909e+00, -1.20880747e+00,
```

```
  1.09595472e+00, -1.30523606e+00, -3.58512423e-01,
 -3.25741297e-01, -4.31870573e-01, -4.44890934e-02,
 -8.32843886e-01, -2.83635764e-01, -8.34320596e-01,
 -1.04822013e+00],
[ 3.56613864e-01,  1.32918909e+00, -9.28838438e-01,
 -1.38559715e+00, -2.79443157e+00, -3.11082764e-01,
  4.04596949e-01,  8.15522588e-01,  7.88424140e-01,
 -5.12726243e-01,  8.57413456e-01, -2.05728275e-01,
  9.53998092e-01],
[ 4.70569422e-01,  4.43063032e-01,  1.03094479e+00,
  1.22656271e+00, -1.88339432e-01, -3.58512423e-01,
  3.90945767e-01,  1.58501837e+00,  1.22900797e+00,
 -2.61205237e-01,  1.50144124e+00,  2.48998085e-01,
  9.53998092e-01],
[-1.69458618e+00,  4.43063032e-01,  1.59088286e+00,
 -6.67253186e-01, -5.60638308e-01, -3.58512423e-01,
  2.23638484e-02, -8.44968308e-01, -7.95579616e-01,
 -6.72785065e-01, -8.57660526e-01, -8.47694901e-01,
  9.53998092e-01],
[-1.69458618e+00, -4.43063032e-01, -1.20880747e+00,
 -2.75429208e-01,  9.28557198e-01,  6.84940077e-01,
 -9.74172449e-01, -9.42167776e-01, -8.37539981e-01,
 -8.55709432e-01, -9.34663848e-01, -8.87817815e-01,
  9.53998092e-01],
[-3.27119484e-01, -1.32918909e+00, -6.48869405e-01,
 -6.67253186e-01, -1.88339432e-01,  5.90080759e-01,
 -3.87171616e-01, -2.77971417e-01, -8.52226108e-01,
 -8.09978340e-01, -4.02640898e-01, -8.34320596e-01,
  9.53998092e-01],
[ 2.42658306e-01,  4.43063032e-01,  1.03094479e+00,
 -5.36645194e-01,  9.28557198e-01, -3.58512423e-01,
  7.80004458e-01,  4.18624764e-01,  1.31083068e+00,
  1.08786198e+00,  7.59409229e-01,  1.19857372e+00,
 -1.04822013e+00],
[-8.96897275e-01,  4.43063032e-01, -8.89313398e-02,
 -1.44821215e-01,  9.28557198e-01, -2.16223446e-01,
 -5.95432445e-02, -2.86071373e-01, -2.98029659e-02,
 -6.27053973e-01, -1.78631235e-01, -6.06957416e-01,
 -1.04822013e+00],
[-1.01085283e+00,  4.43063032e-01,  4.71006726e-01,
  5.10907736e-02, -9.32937185e-01,  1.01694769e+00,
 -5.64636984e-01, -6.18169552e-01, -8.10265744e-01,
 -8.55709432e-01, -6.82652977e-01, -8.74443510e-01,
  9.53998092e-01],
```

```
[-1.35271951e+00, -4.43063032e-01,  1.91037693e-01,
  1.03065072e+00,  5.56258322e-01, -3.58512423e-01,
  1.45224488e-01, -8.28768397e-01, -6.27738159e-01,
 -5.12726243e-01, -7.66656600e-01, -7.27326159e-01,
 -1.04822013e+00],
[ 1.26825833e+00,  4.43063032e-01,  1.03094479e+00,
 -5.36645194e-01, -5.60638308e-01, -3.58512423e-01,
  8.89213915e-01,  4.13650438e+00,  2.70391477e+00,
  1.15645861e+00,  3.59453153e+00,  2.53600419e+00,
  9.53998092e-01],
[-9.92083682e-02, -4.43063032e-01,  4.71006726e-01,
 -1.12438116e+00,  5.56258322e-01, -3.58512423e-01,
  8.55085960e-01,  2.56625652e-01, -5.10249138e-01,
  1.36224853e+00,  7.33796359e-02,  9.31087624e-01,
  9.53998092e-01],
[ 1.47471899e-02,  4.43063032e-01,  2.15082092e+00,
 -1.38559715e+00, -2.04983382e+00, -3.11082764e-01,
  9.57469826e-01,  6.29223609e-01,  3.49938332e-01,
  1.04213088e+00,  5.28399264e-01,  1.02470776e+00,
  9.53998092e-01],
[ 2.42658306e-01, -1.32918909e+00,  1.91037693e-01,
 -1.42132227e-02, -2.04983382e+00,  3.76786792e+00,
 -8.51311810e-01, -8.28768397e-01, -8.41736017e-01,
 -9.24306070e-01, -8.57660526e-01, -9.01192120e-01,
  9.53998092e-01],
[ 1.15430277e+00,  4.43063032e-01,  1.91037693e-01,
 -1.64681313e+00,  1.83959445e-01, -3.58512423e-01,
  9.71121008e-01,  3.76390642e+00,  2.49201493e+00,
  1.86529054e+00,  3.31451945e+00,  3.09772499e+00,
  9.53998092e-01],
[ 6.98480538e-01, -4.43063032e-01, -8.89313398e-02,
 -9.28469172e-01,  9.28557198e-01, -3.58512423e-01,
  9.23341871e-01,  2.37881402e+00,  8.74442887e-01,
  1.88815609e+00,  1.91445905e+00,  2.44238406e+00,
  9.53998092e-01],
[ 1.26825833e+00,  1.32918909e+00, -3.68900372e-01,
  2.47002763e-01, -1.67753494e+00, -3.58512423e-01,
  5.27457588e-01, -5.29070040e-01, -4.93464992e-01,
 -2.15474145e-01, -5.35646635e-01, -4.06342846e-01,
 -1.04822013e+00],
[ 3.56613864e-01, -4.43063032e-01,  7.50975758e-01,
 -2.10125212e-01,  5.56258322e-01, -3.58512423e-01,
  6.91271774e-01, -2.29371684e-01,  9.18820909e-02,
  6.30551056e-01,  1.08381146e-01,  5.03109874e-01,
```

```
 -1.04822013e+00],
[ 4.70569422e-01,  1.32918909e+00, -1.20880747e+00,
  1.42247470e+00,  9.28557198e-01, -3.58512423e-01,
  3.97771358e-01, -1.07872350e-01,  1.84514532e-02,
 -1.01146415e-01, -4.56254975e-02, -1.12108142e-01,
 -1.04822013e+00],
[ 1.72408056e+00,  4.43063032e-01, -6.48869405e-01,
 -2.10125212e-01,  1.30085608e+00, -3.58512423e-01,
  7.59527685e-01,  1.05042130e+00,  2.51299512e+00,
  1.11072752e+00,  1.58544486e+00,  1.63992577e+00,
  9.53998092e-01],
[ 1.47471899e-02, -4.43063032e-01, -1.20880747e+00,
  9.00042726e-01,  2.04545383e+00, -3.58512423e-01,
 -3.66694843e-01, -6.34369463e-01, -4.80876883e-01,
 -9.70037162e-01, -5.84648749e-01, -9.01192120e-01,
 -1.04822013e+00],
[-4.41075042e-01, -1.32918909e+00, -1.76874554e+00,
  1.22656271e+00,  1.30085608e+00, -3.58512423e-01,
  4.11422540e-01, -8.35724830e-02,  2.72311658e-01,
 -3.25497772e-02,  7.33796359e-02, -5.11370474e-03,
 -1.04822013e+00],
[ 5.84524980e-01,  1.32918909e+00, -3.68900372e-01,
 -4.71341197e-01, -1.67753494e+00, -3.58512423e-01,
  6.70795001e-01,  1.35126318e-01,  1.98881020e-01,
  1.04643499e-01,  1.78384165e-01,  1.55377952e-01,
 -1.04822013e+00],
[ 1.61012500e+00,  1.32918909e+00, -2.32868360e+00,
  1.16394770e-01, -1.88339432e-01, -2.63653105e-01,
 -7.28451170e-01, -8.93568042e-01, -6.69698523e-01,
 -8.55709432e-01, -8.29659318e-01, -8.87817815e-01,
  9.53998092e-01],
[-1.69458618e+00, -4.43063032e-01, -9.28838438e-01,
  3.77610755e-01,  1.30085608e+00,  1.15784168e-01,
 -1.23354491e+00, -9.98867465e-01, -8.35441962e-01,
 -8.78574978e-01, -9.62665055e-01, -9.01192120e-01,
 -1.04822013e+00],
[ 1.49616944e+00,  1.32918909e+00, -2.32868360e+00,
 -5.36645194e-01,  9.28557198e-01, -3.11082764e-01,
  1.17922124e-01, -8.53068264e-01, -6.96972760e-01,
 -7.41381703e-01, -8.08658412e-01, -8.61069206e-01,
  9.53998092e-01],
[-3.27119484e-01,  4.43063032e-01,  7.50975758e-01,
 -1.84272512e+00,  1.83959445e-01, -2.63653105e-01,
  7.04922956e-01,  1.67526140e-01, -6.96653121e-02,
```

```
  5.84819964e-01,  5.93790320e-02,  4.49612655e-01,
  9.53998092e-01],
[ 6.98480538e-01,  1.32918909e+00, -3.68900372e-01,
 -7.97861179e-01,  1.30085608e+00, -7.39344686e-02,
 -1.17641070e-02, -3.42771062e-01, -1.47291986e-01,
 -5.81322881e-01, -2.62634859e-01, -5.80208807e-01,
 -1.04822013e+00],
[-1.58063062e+00, -1.32918909e+00, -9.28838438e-01,
 -7.95172191e-02, -9.32937185e-01,  2.58073145e-01,
 -9.26393311e-01, -8.69268175e-01, -8.56422144e-01,
 -9.01440524e-01, -8.92662036e-01, -9.01192120e-01,
 -1.04822013e+00],
[-1.35271951e+00, -4.43063032e-01,  4.71006726e-01,
 -2.75429208e-01,  9.28557198e-01, -3.58512423e-01,
  6.84446183e-01, -9.97723942e-02, -2.43800824e-01,
  6.99147694e-01, -1.78631235e-01,  3.69366827e-01,
  9.53998092e-01],
[ 2.42658306e-01,  1.32918909e+00,  1.03094479e+00,
 -1.90802912e+00, -9.32937185e-01, -3.11082764e-01,
  8.61911551e-01,  8.88422188e-01,  6.16386646e-01,
  9.27803154e-01,  8.08411343e-01,  1.10495359e+00,
  9.53998092e-01],
[-1.35271951e+00,  1.32918909e+00, -9.28838438e-01,
  7.04130737e-01,  5.56258322e-01, -3.11082764e-01,
 -6.53369668e-01, -1.04746720e+00, -6.86482669e-01,
 -8.09978340e-01, -9.62665055e-01, -8.87817815e-01,
 -1.04822013e+00],
[-3.27119484e-01,  4.43063032e-01,  7.50975758e-01,
  5.10907736e-02, -1.88339432e-01, -3.58512423e-01,
  6.16190272e-01,  3.37625208e-01,  7.69541976e-01,
  2.18971229e-01,  5.42399868e-01,  4.09489741e-01,
 -1.04822013e+00],
[ 2.42658306e-01, -4.43063032e-01, -3.68900372e-01,
  3.77610755e-01, -5.60638308e-01, -3.58512423e-01,
  5.06980815e-01, -2.21271728e-01, -1.11625676e-01,
 -7.82808692e-02, -1.71630933e-01, -1.52231056e-01,
 -1.04822013e+00],
[ 3.56613864e-01, -1.32918909e+00, -8.89313398e-02,
  3.12306759e-01, -5.60638308e-01,  1.77582224e+00,
 -9.12742129e-01, -8.44968308e-01, -8.43834035e-01,
 -9.01440524e-01, -8.71661130e-01, -9.01192120e-01,
  9.53998092e-01],
[-1.23876395e+00, -4.43063032e-01,  1.91037693e-01,
  2.47002763e-01, -5.60638308e-01, -3.58512423e-01,
```

```
        5.41108770e-01, -5.85769730e-01, -4.30524446e-01,
       -3.25497772e-02, -5.28646333e-01, -2.99348408e-01,
       -1.04822013e+00],
      [-2.13163926e-01,  1.32918909e+00, -2.88862167e+00,
        9.00042726e-01,  3.90694821e+00,  3.57814928e+00,
       -2.08674379e+00, -1.11226684e+00, -8.69010254e-01,
       -1.03863380e+00, -1.05366898e+00, -9.41315034e-01,
       -1.04822013e+00],
      [ 1.28702748e-01,  4.43063032e-01,  1.31091382e+00,
       -6.67253186e-01,  9.28557198e-01, -3.58512423e-01,
        7.73178867e-01,  9.46265400e-02,  1.09683282e+00,
        1.08786198e+00,  4.37395338e-01,  9.97959148e-01,
       -1.04822013e+00],
      [ 1.38221389e+00, -4.43063032e-01,  1.91037693e-01,
       -3.40733205e-01,  1.83959445e-01, -3.58512423e-01,
        6.50318228e-01,  7.03266732e-02, -2.27016679e-01,
        4.01895596e-01, -6.66264034e-02,  2.48998085e-01,
        9.53998092e-01]])
```

6.0 Model

1.0 Logistic Regression

In [140…
```
### Creating a Logistic regression object
logistic_reg=LogisticRegression()
logistic_reg
```

Out[140]:
```
▾ LogisticRegression

LogisticRegression()
```

In [141…
```
### Passing independant and dependant training data to the model
logistic_reg.fit(X_train,y_train)
```

Out[141]:
```
▾ LogisticRegression

LogisticRegression()
```

1.1 Using Above Model to get prediction for test data

In [142…
```
logistic_reg_pred=logistic_reg.predict(X_test)
logistic_reg_pred
```

Out[142]:
```
array([1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1,
       0, 1, 1, 0, 1, 0, 1, 1], dtype=int64)
```

1.2.0 Performance Metrics

1.2.1 Confusion Matrix

In [143…
```
confusion_mat=confusion_matrix(y_test, logistic_reg_pred)
confusion_mat
```

Out[143]:
```
array([[34,  2],
       [ 1, 37]], dtype=int64)
```

In [144…
```
truly_positive=confusion_mat[0][0]
falsely_positive=confusion_mat[0][1]
falsely_negative=confusion_mat[1][0]
truly_negative=confusion_mat[1][1]
```

1.2.2 Accuracy Score

In [145…
```
### accuracy using accuracy_score
accuracy=round(accuracy_score(y_test, logistic_reg_pred),4)
accuracy
```

Out[145]:
```
0.9595
```

In [146…
```
### manual calcualtion for accuracy
accuracy_manual=round(((truly_positive+truly_negative)/(truly_positive+falsely_positive+falsely_negative+truly_negative)),4)
print("Accuracy of our model is {}".format(accuracy_manual))
```

```
Accuracy of our model is 0.9595
```

1.2.3 Precision Score

In [147…
```
precision_manual=round(truly_positive/(truly_positive+falsely_positive),4)
print("Precision of our model is {}".format(precision_manual))
```

```
Precision of our model is 0.9444
```

1.2.4 Recall Score

In [148…
```
recall_manual=round(truly_positive/(truly_positive+falsely_negative),4)
print("Recall of our model is {}".format(recall_manual))
```

```
Recall of our model is 0.9714
```

1.2.5 F-1 Score

1. Giving equal importance to falsely positive and falsely negative

In [149…
```
f1_score=2*(precision_manual*recall_manual)/(precision_manual+recall_manual)
print("F-1 Score of our model is {} ".format(round(f1_score,4)))
```

```
F-1 Score of our model is 0.9577
```

1.2.6 Classification Report

In [150…
```
print(classification_report(y_test, logistic_reg_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.94      0.96        36
           1       0.95      0.97      0.96        38

    accuracy                           0.96        74
   macro avg       0.96      0.96      0.96        74
weighted avg       0.96      0.96      0.96        74
```

7.0 Saving the Model

In [151…
```
### Writing model to a file that will be used while deployment
with open('model_Logistic_regression_algerian_ff.sav','wb') as f:
    pickle.dump(logistic_reg,f)
```

8.0 Creating Imbalance in dataset

In [65]:
```
data.head()
```

Out[65]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	1	6	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0.0	0
1	2	6	29	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0.0	0
2	3	6	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0.0	0
3	4	6	25	89	13	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0.0	0
4	5	6	27	77	16	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0.0	0

In [66]:
```
data.shape
```

Out[66]:
```
(244, 14)
```

In [67]:
```
### Creating imbalance
### 1. splitting data in 90:10 percent ratio using train test split
X1=data.iloc[:,:-1]
y1=data.iloc[:,-1]
```

In [68]:
```
X_train_imb, X_test_imb, y_train_imb, y_test_imb = train_test_split(X1, y1, test_size=0.10, random_state=17)
```

In [69]:
```
### both will have same shape
X_train_imb.shape, y_train_imb.shape
```

Out[69]:
```
((219, 13), (219,))
```

In [70]:
```
### both will have same shape
X_test_imb.shape, y_test_imb.shape
```

Out[70]:
```
((25, 13), (25,))
```

In [71]:
```
### Replacing all values as 1 in y_train and all values as zero in y_test to create imbalance
```

```
y_train_imb=y_train_imb.replace(0,1)
```

In [72]:
```
y_train_imb.head()
```

Out[72]:
```
156    1
183    1
11     1
75     1
130    1
Name: Classes, dtype: int64
```

In [73]:
```
y_test_imb=y_test_imb.replace(1,0)
y_test_imb.head()
```

Out[73]:
```
48     0
216    0
101    0
38     0
86     0
Name: Classes, dtype: int64
```

In [74]:
```
X_train_imb.head()
```

Out[74]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
156	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0
183	1	8	38	52	14	0.0	78.3	4.4	10.5	2.0	4.4	0.8	1.0
11	12	6	26	81	19	0.0	84.0	13.8	61.4	4.8	17.7	7.1	0.0
75	15	8	36	55	13	0.3	82.4	15.6	92.5	3.7	22.0	6.3	0.0
130	9	6	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	1.0

In [75]:
```
### Combining X_train_imb and y_train_imb
train_imb=X_train_imb.join(pd.DataFrame(y_train_imb))
train_imb.head()
```

Out[75]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
156	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0	1
183	1	8	38	52	14	0.0	78.3	4.4	10.5	2.0	4.4	0.8	1.0	1
11	12	6	26	81	19	0.0	84.0	13.8	61.4	4.8	17.7	7.1	0.0	1
75	15	8	36	55	13	0.3	82.4	15.6	92.5	3.7	22.0	6.3	0.0	1
130	9	6	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	1.0	1

In [76]:
```
### Combining X_test_imb and y_test_imb
test_imb=X_test_imb.join(pd.DataFrame(y_test_imb))
test_imb.head()
```

Out[76]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
48	19	7	35	59	17	0.0	88.1	12.0	52.8	7.7	18.2	10.9	0.0	0
216	3	9	28	75	16	0.0	82.2	4.4	24.3	3.3	6.0	2.5	1.0	0
101	10	9	33	73	12	1.8	59.9	2.2	8.9	0.7	2.7	0.3	0.0	0
38	9	7	32	68	14	1.4	66.6	7.7	9.2	1.1	7.4	0.6	0.0	0
86	26	8	31	78	18	0.0	85.8	45.6	190.6	4.7	57.1	13.7	0.0	0

In [77]:
```
### Checking shape of imbalance dataset
train_imb.shape, test_imb.shape
```

Out[77]:
```
((219, 14), (25, 14))
```

In [78]:
```
### Combining train_imb dataset and test_imb dataset into data_imb dataset
data_imb=pd.concat([train_imb, test_imb], ignore_index=True, sort=False)
```

In [79]:
```
data_imb.head()
```

Out[79]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0	1
1	1	8	38	52	14	0.0	78.3	4.4	10.5	2.0	4.4	0.8	1.0	1
2	12	6	26	81	19	0.0	84.0	13.8	61.4	4.8	17.7	7.1	0.0	1
3	15	8	36	55	13	0.3	82.4	15.6	92.5	3.7	22.0	6.3	0.0	1
4	9	6	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	1.0	1

In [80]:
```
data_imb.shape
```

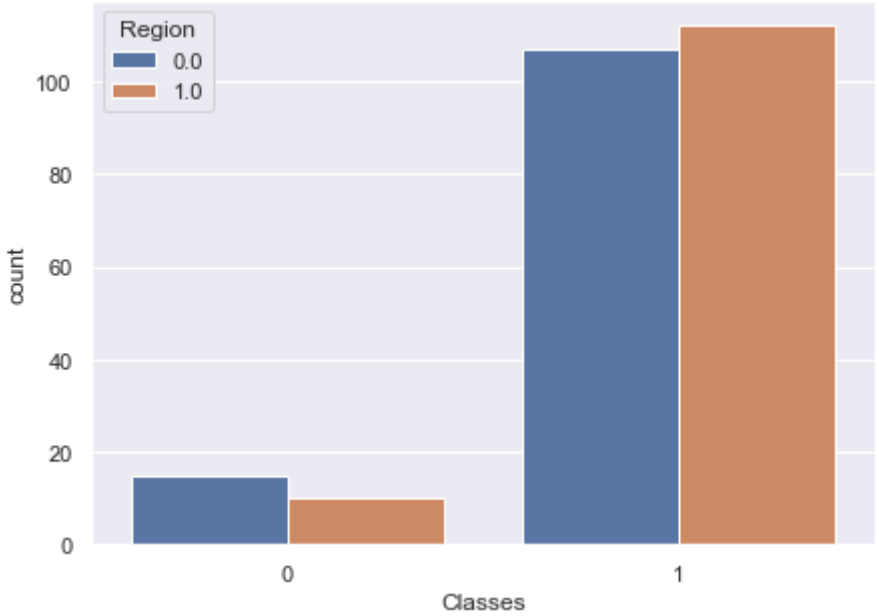
Out[80]:
```
(244, 14)
```

In [81]:
```
### there is imbalence in our dataset
data_imb.Classes.value_counts()
```

Out[81]:
```
1    219
0     25
Name: Classes, dtype: int64
```

In [168…
```
### 0 is not fire and 1 i fire for Classes
### 0 is Bejaia region and 1 is Sidi Bel-abbes region
plt.figure(figsize=(7,5))
sns.countplot(data=data_imb, x='Classes', hue='Region')
```

Out[168]:
```
<AxesSubplot:xlabel='Classes', ylabel='count'>
```



9.0 Logistic Regression Model on Imbalence dataset

9.1 Seperating Independent Features and Dependent Feature

In [83]:
```
data_imb.head()
```

Out[83]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0	1
1	1	8	38	52	14	0.0	78.3	4.4	10.5	2.0	4.4	0.8	1.0	1
2	12	6	26	81	19	0.0	84.0	13.8	61.4	4.8	17.7	7.1	0.0	1
3	15	8	36	55	13	0.3	82.4	15.6	92.5	3.7	22.0	6.3	0.0	1
4	9	6	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	1.0	1

In [84]:
```
X1=data_imb.iloc[:,:-1]
y1=data_imb.iloc[:,-1]
```

In [85]:
```
X1.head()
```

Out[85]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
0	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0
1	1	8	38	52	14	0.0	78.3	4.4	10.5	2.0	4.4	0.8	1.0
2	12	6	26	81	19	0.0	84.0	13.8	61.4	4.8	17.7	7.1	0.0
3	15	8	36	55	13	0.3	82.4	15.6	92.5	3.7	22.0	6.3	0.0
4	9	6	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	1.0

In [86]:
```
y1.head()
```

Out[86]:
```
0    1
1    1
2    1
3    1
4    1
Name: Classes, dtype: int64
```

9.2 Handling Imbalance dataset by Doing Upsampling

In [89]:
```
### for upsampling
from imblearn.combine import SMOTETomek
```

In [90]:
```
### Creating SMOTETomek object for over sampling
smk=SMOTETomek()
smk
```

Out[90]:
```
SMOTETomek()
```

In [92]:
```
X_bal,y_bal=smk.fit_resample(X1,y1)
```

In [93]:
```
X_bal.head()
```

Out[93]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
0	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0
1	1	8	38	52	14	0.0	78.3	4.4	10.5	2.0	4.4	0.8	1.0
2	12	6	26	81	19	0.0	84.0	13.8	61.4	4.8	17.7	7.1	0.0
3	15	8	36	55	13	0.3	82.4	15.6	92.5	3.7	22.0	6.3	0.0
4	9	6	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	1.0

In [94]:
```
y_bal.head()
```

Out[94]:
```
0    1
1    1
2    1
3    1
4    1
Name: Classes, dtype: int64
```

In [98]:
```
from collections import Counter
print('Original dataset shape {}'.format(Counter(y1)))
print('Resampled dataset shape {}'.format(Counter(y_bal)))
```

```
Original dataset shape Counter({1: 219, 0: 25})
Resampled dataset shape Counter({1: 206, 0: 206})
```

In [99]:
```
### Now our dataset is balanced
X_bal.shape, y_bal.shape
```

Out[99]:
```
((412, 13), (412,))
```

In [173…
```
### Creating data_bal for basic EDA on dataset after balancing imbalanced dataset
data_bal=X_bal.join(pd.DataFrame(y_bal))
data_bal.head()
```

Out[173]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	5	7	34	45	18	0.0	90.5	18.7	46.4	11.3	18.7	15.0	1.0	1
1	1	8	38	52	14	0.0	78.3	4.4	10.5	2.0	4.4	0.8	1.0	1
2	12	6	26	81	19	0.0	84.0	13.8	61.4	4.8	17.7	7.1	0.0	1
3	15	8	36	55	13	0.3	82.4	15.6	92.5	3.7	22.0	6.3	0.0	1
4	9	6	27	59	18	0.1	78.1	8.5	14.7	2.4	8.3	1.9	1.0	1

9.3 EDA on balanced dataset

9.3.1 Discrete and Continuous features

In [174…
```
data_bal[data_bal.columns].nunique()
```

Out[174]:
```
day             31
month            4
Temperature     19
RH              63
Ws              18
Rain           138
FFMC           350
DMC            339
DC             367
ISI            283
BUI            346
FWI            298
Region         102
Classes          2
dtype: int64
```

In [175…
```
### Getting list of numerical features excluding Classes and Region
print(numerical_features)
```

```
['day', 'month', 'Temperature', 'RH', 'Ws', 'Rain', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI', 'FWI']
```

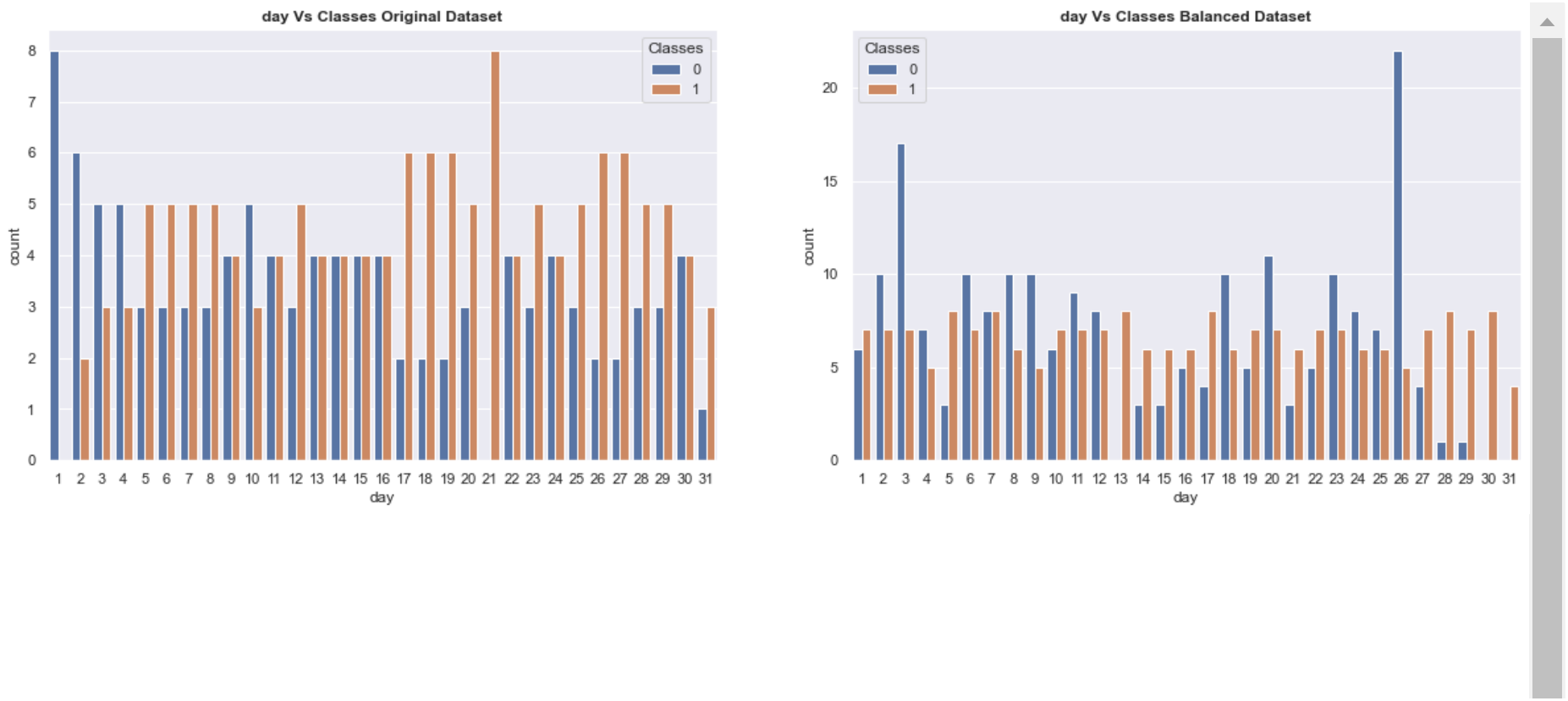
In [176…
```
# here the assumption to consider a feature discrete is that it should have less than 35 unique values otherwise it will be
# considered continuous feature
print(discrete_features)
```

```
['day', 'month', 'Temperature', 'Ws', 'Region']
```

9.3.2 Comparing Discrete features for Original and Balanced Dataset

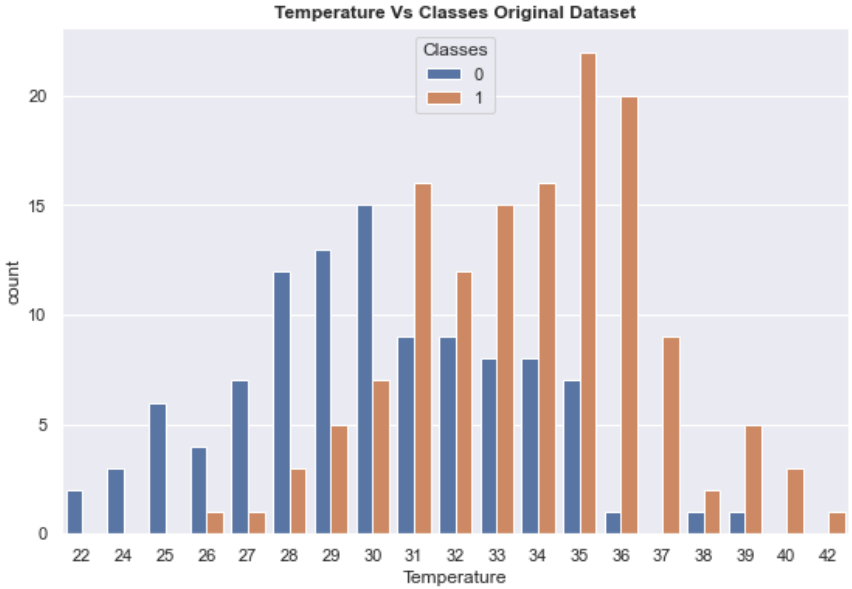
In [187…

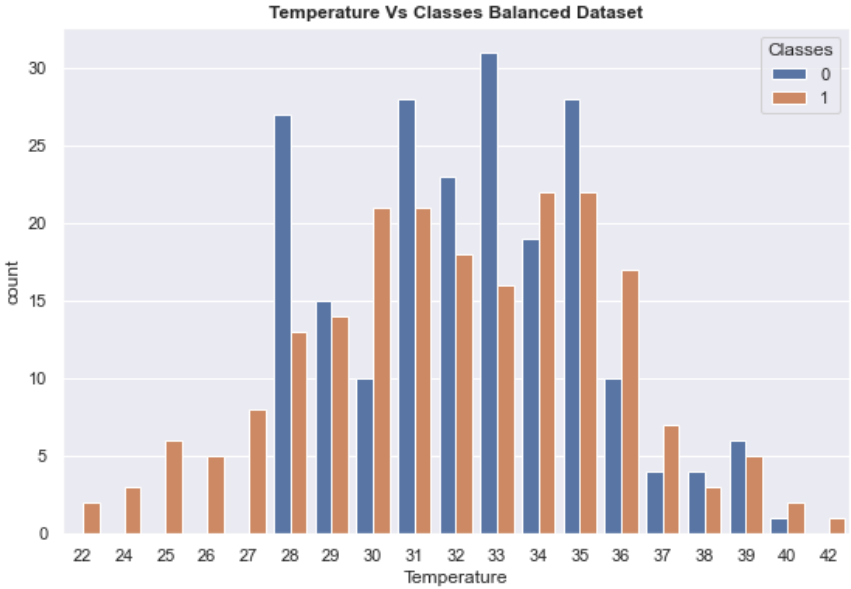
```
### this is bivariate analysis between target feature classes and discrete numerical features
### for this we plot count plot
count_feat=[feature for feature in discrete_features if feature not in ['Region']]

for feature in count_feat:
    plt.figure(figsize=(20,6))
    plt.subplot(121)
    sns.countplot(data=dataset, x=feature, hue='Classes')
    plt.title("{} Vs Classes Original Dataset".format(feature),fontweight="bold")

    plt.subplot(122)
    sns.countplot(data=data_bal, x=feature, hue='Classes')
    plt.title("{} Vs Classes Balanced Dataset".format(feature),fontweight="bold")
```

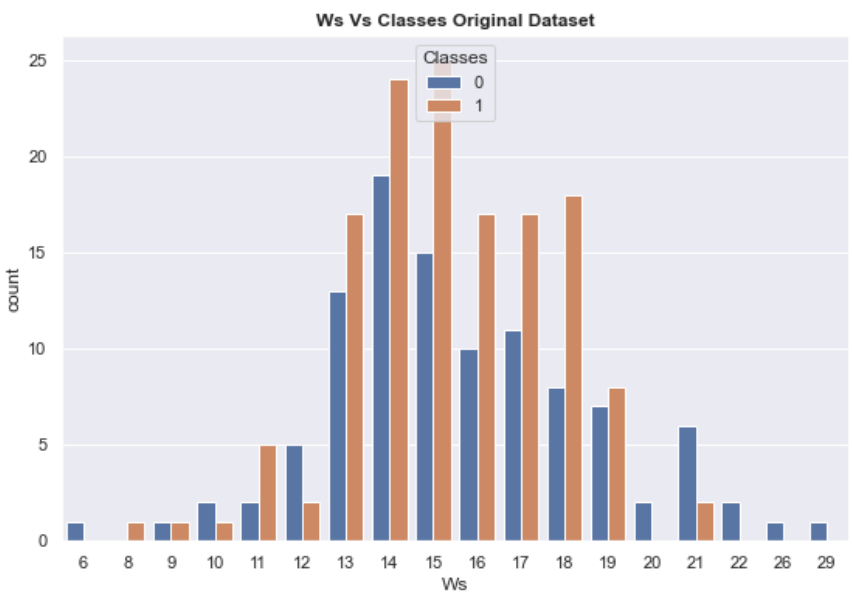


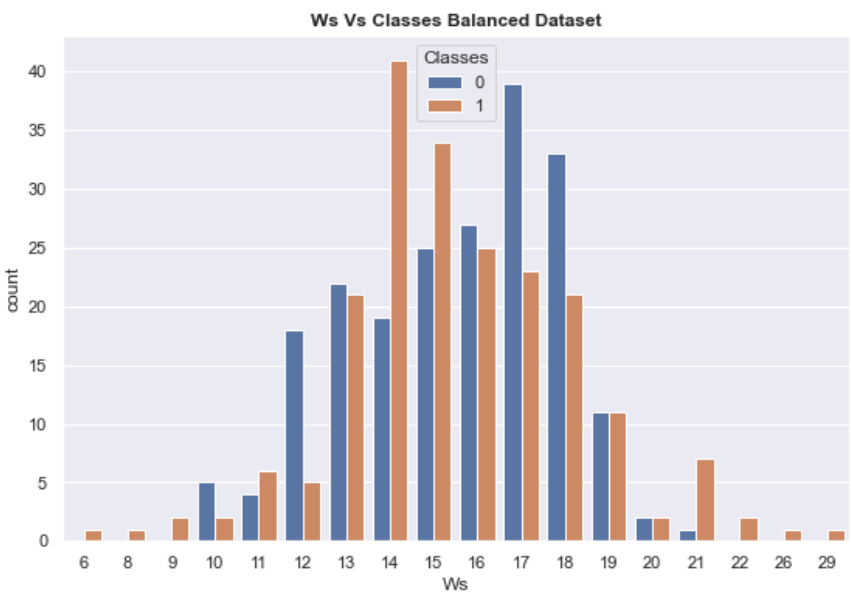

month Vs Classes Original Dataset
Classes
0
1
count
month
0
10
20
30
40
50
6
7
8
9


month Vs Classes Balanced Dataset
Classes
0
1
count
month
0
10
20
30
40
50
60
70
6
7
8
9


Temperature Vs Classes Original Dataset
Classes
0
1
count
Temperature
0
5
10
15
20
22
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
42


Temperature Vs Classes Balanced Dataset
Classes
0
1
count
Temperature
0
5
10
15
20
25
30
22
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
42





Observations

1. After balencing imbalance dataset there is more addition of cases of non fire records than fire records.
2. Original dataset had more fire cases daily than non fire cases, but it is reversed in balanced dataset.
3. Originally there were more No. of non fire cases in month of June and September, but now there are more fire cases in respective months.
4. Originally there were more No. of fire cases in month of July and August, but now there are more non fire cases in respective months.
5. Originally between temperature range of 30 to 37 degree celsius there were more fire cases, but now for the same temperature range there are more non fire cases.
6. For wind speed range og 16 to 18 Km/Hr in original dataset there were more cases of fire, but now for balanced dataset there are more non fire cases.

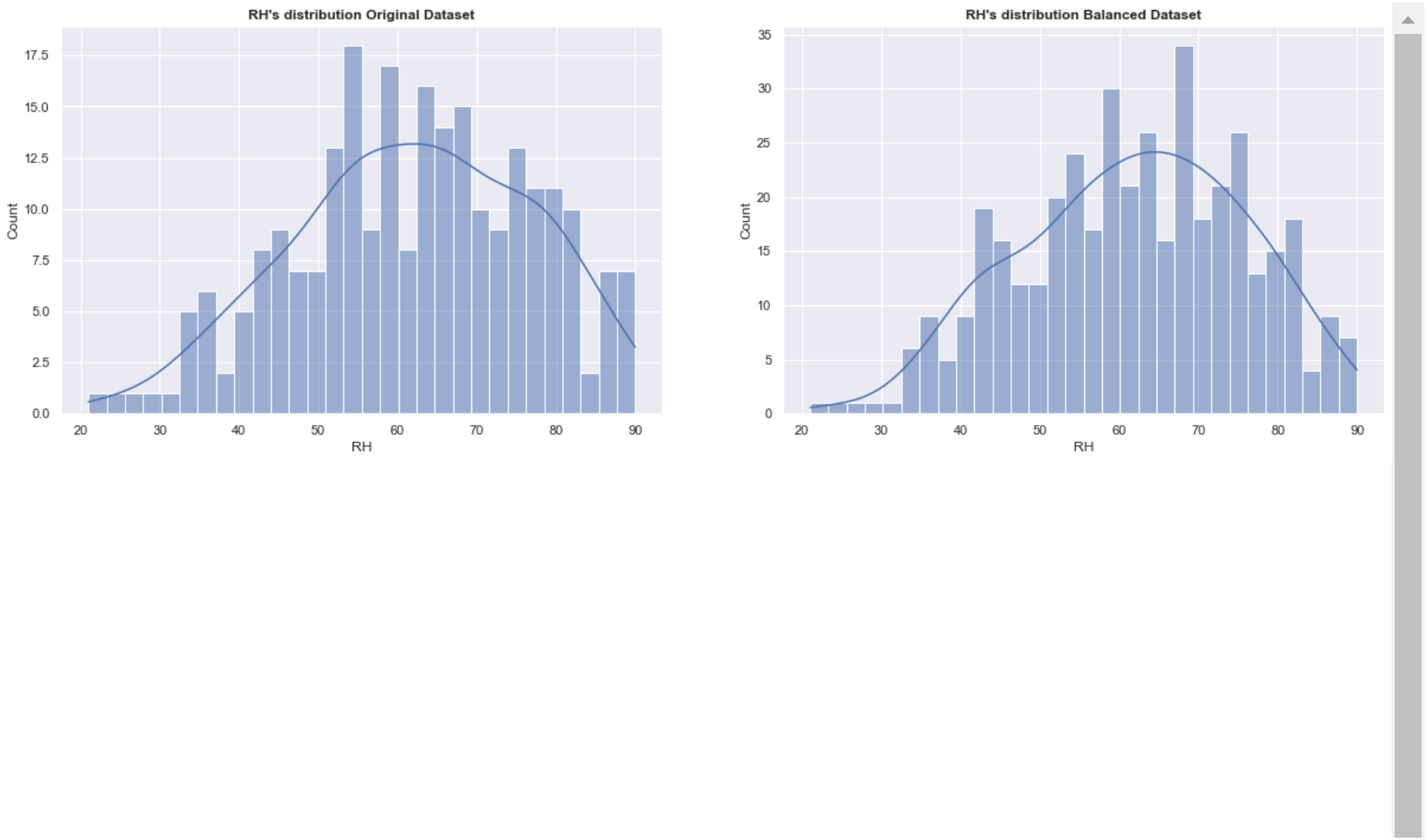
9.3.3 Comparing Distribution of Continuous features for Original and Balanced Dataset

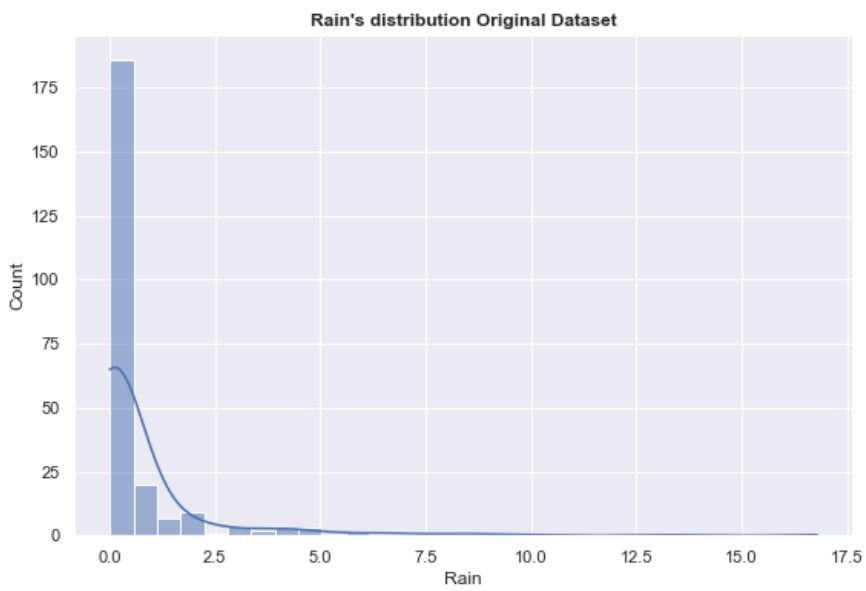
In [179…
```
### Continuous Features
print(continuous_features)
```

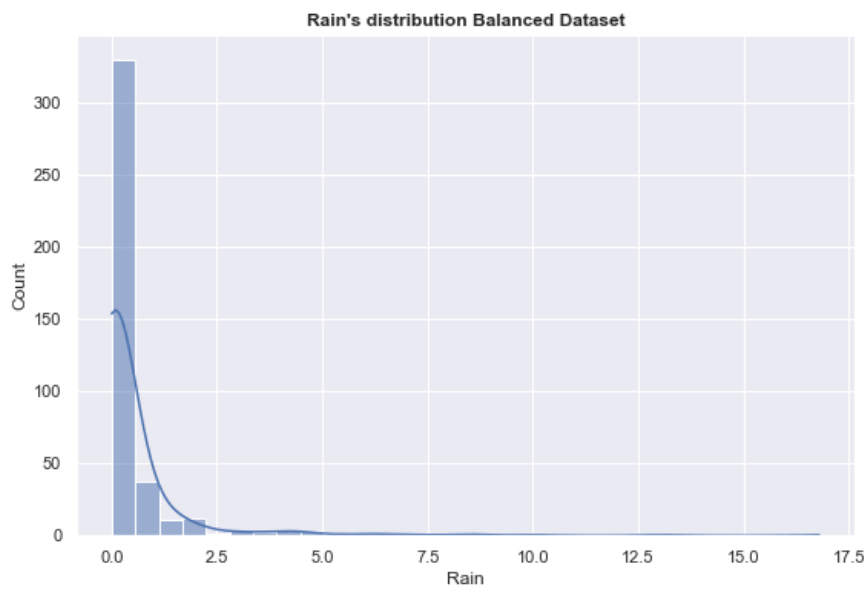
```
['RH', 'Rain', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI', 'FWI']
```

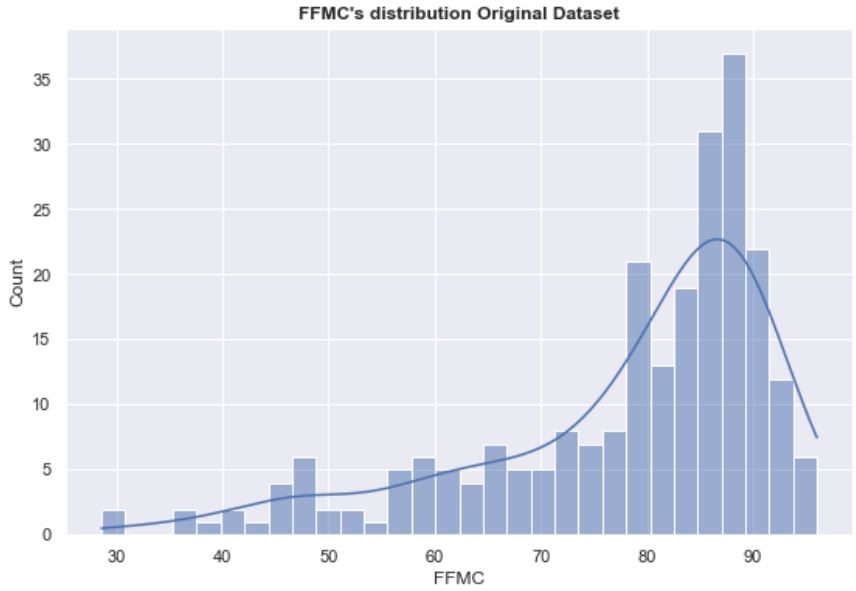
In [188…
```
### Checking distribution of Continuous numerical features
```

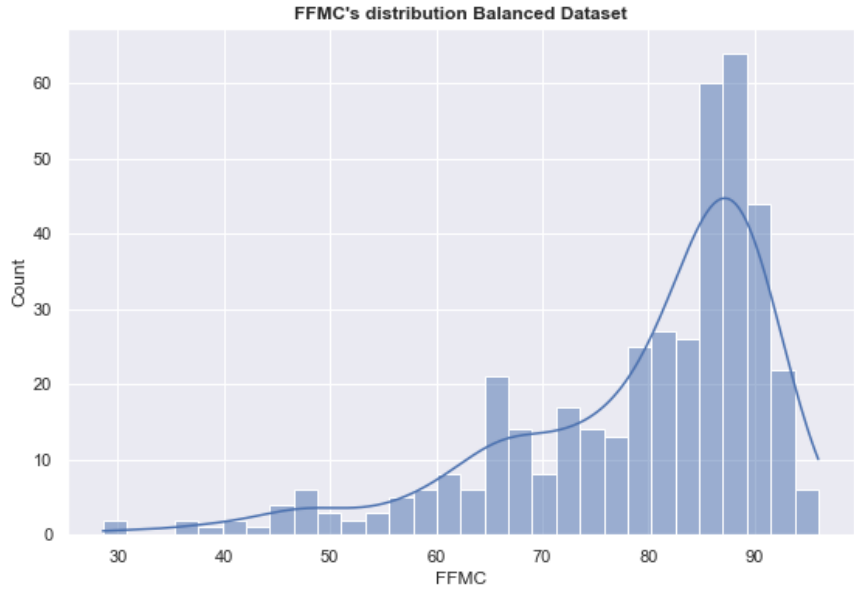
```
for i in continuous_features:
    plt.figure(figsize=(20,6))
    plt.subplot(121)
    sns.histplot(data=dataset, x=i, kde=True, bins=30)
    plt.title("{}'s distribution Original Dataset".format(i),fontweight="bold")

    plt.subplot(122)
    sns.histplot(data=data_bal, x=i, kde=True, bins=30)
    plt.title("{}'s distribution Balanced Dataset".format(i),fontweight="bold")
```

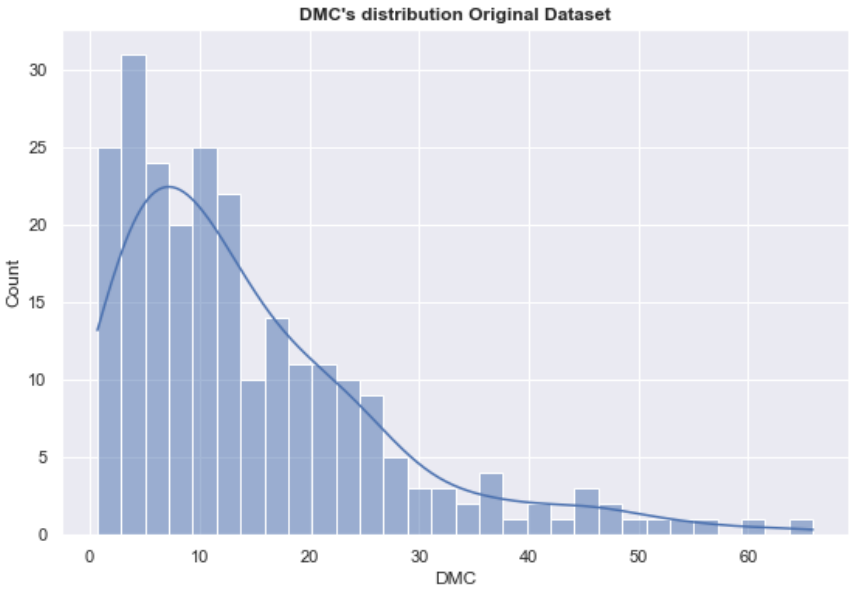


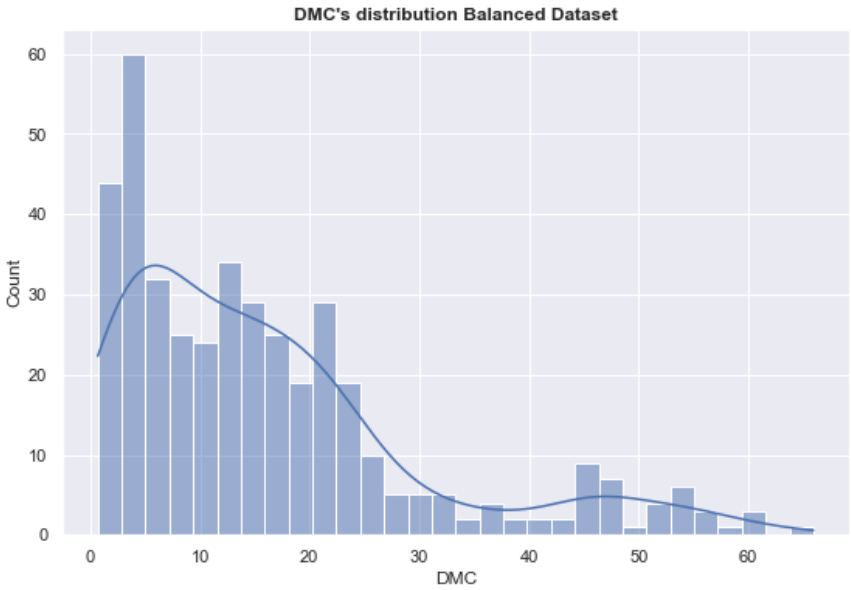

Rain's distribution Original Dataset
Count
Rain
0
25
50
75
100
125
150
175
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

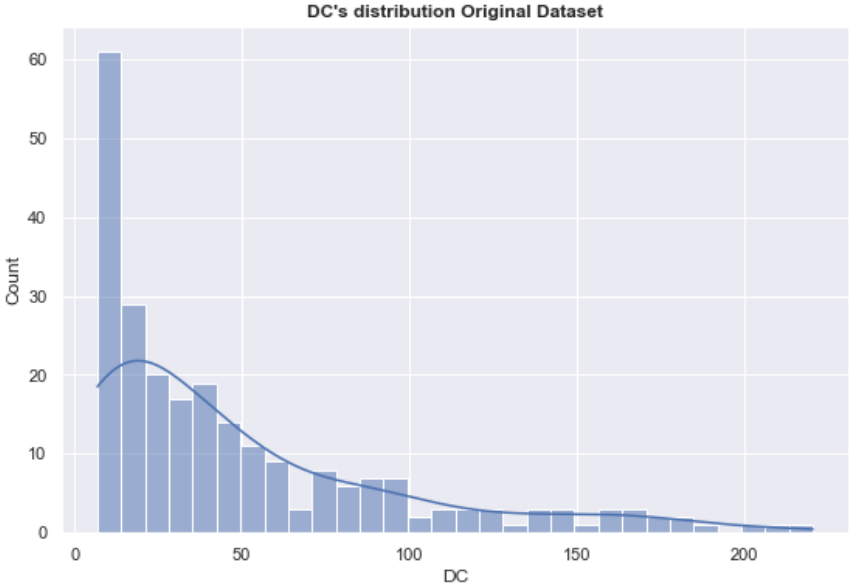

Rain's distribution Balanced Dataset
Count
Rain
0
50
100
150
200
250
300
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

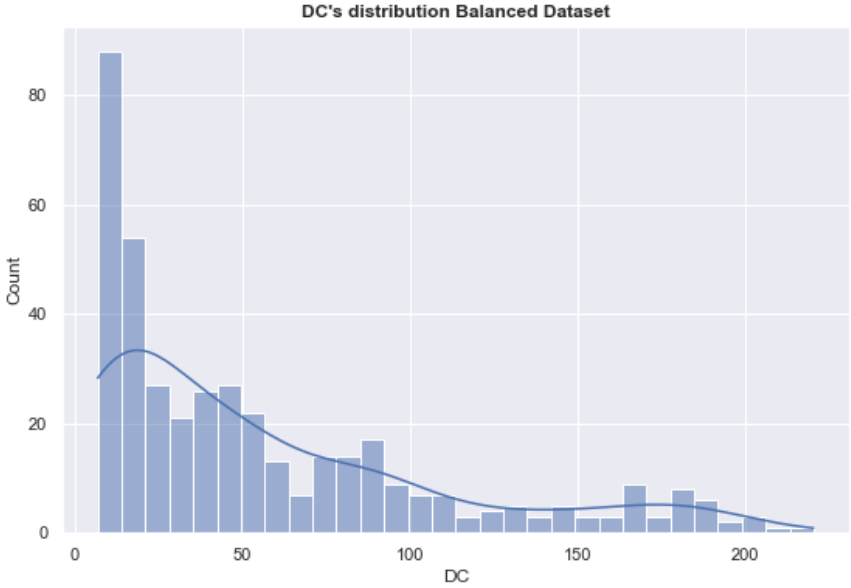

FFMC's distribution Original Dataset
Count
FFMC
0
5
10
15
20
25
30
35
30
40
50
60
70
80
90

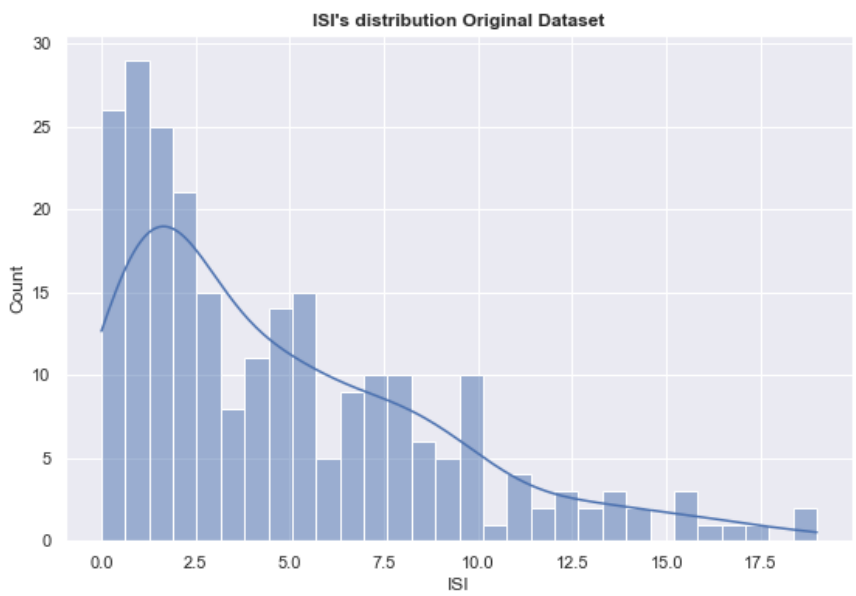

FFMC's distribution Balanced Dataset
Count
FFMC
0
10
20
30
40
50
60
30
40
50
60
70
80
90

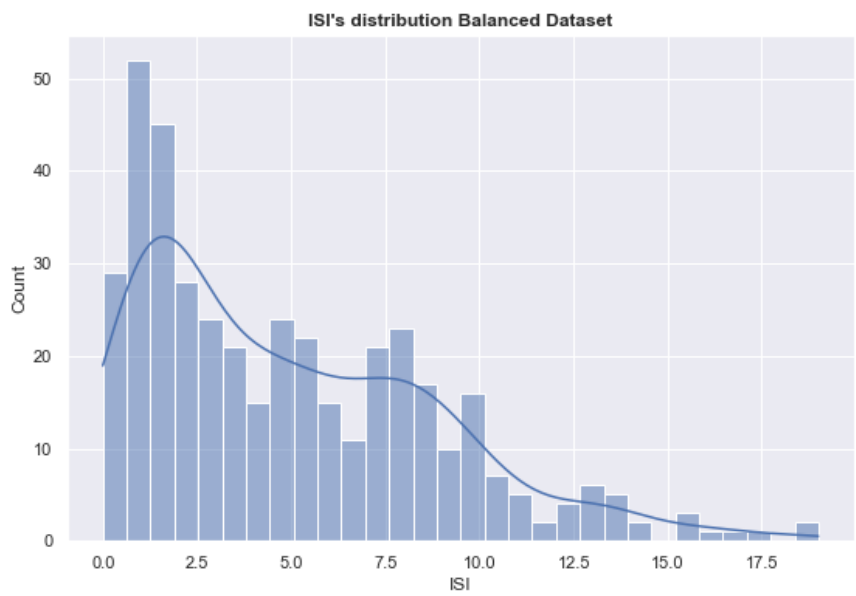

DMC's distribution Original Dataset
Count
DMC

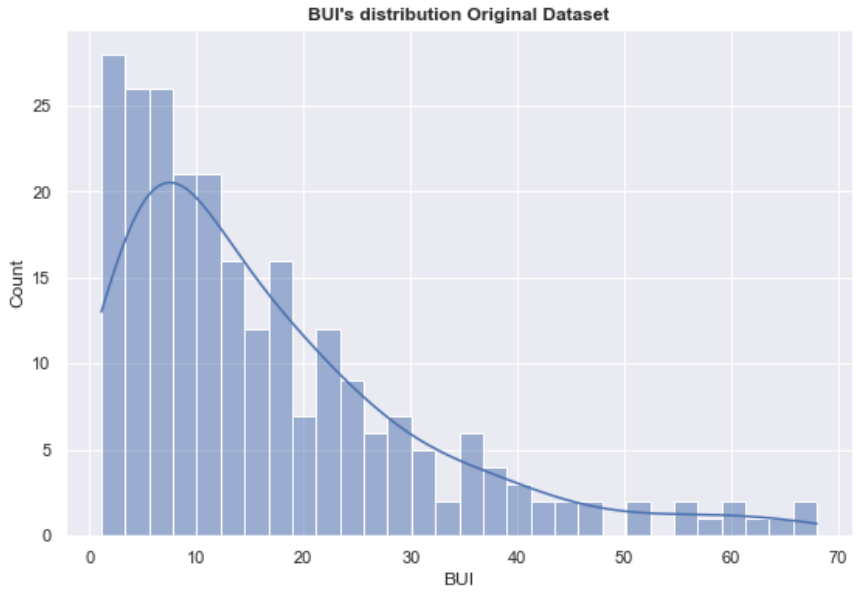

DMC's distribution Balanced Dataset
Count
DMC

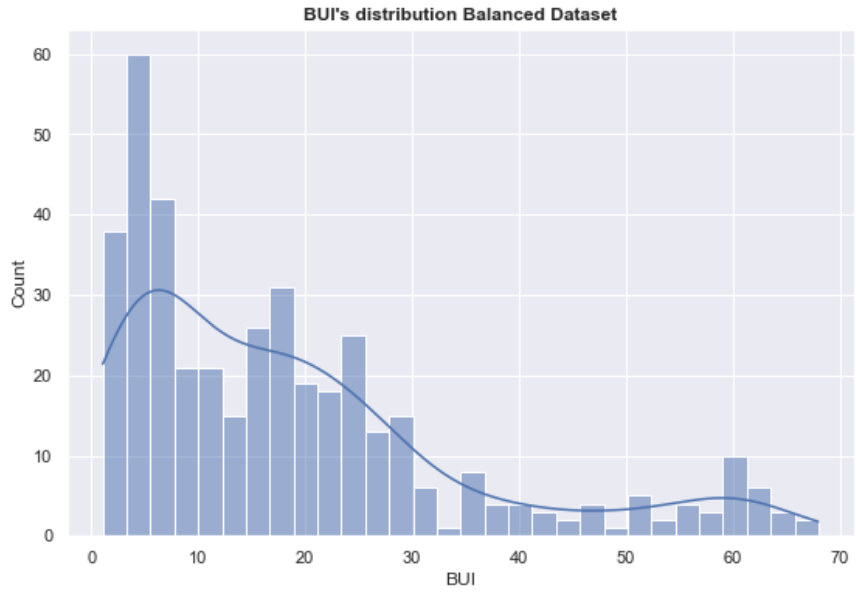

DC's distribution Original Dataset
Count
DC

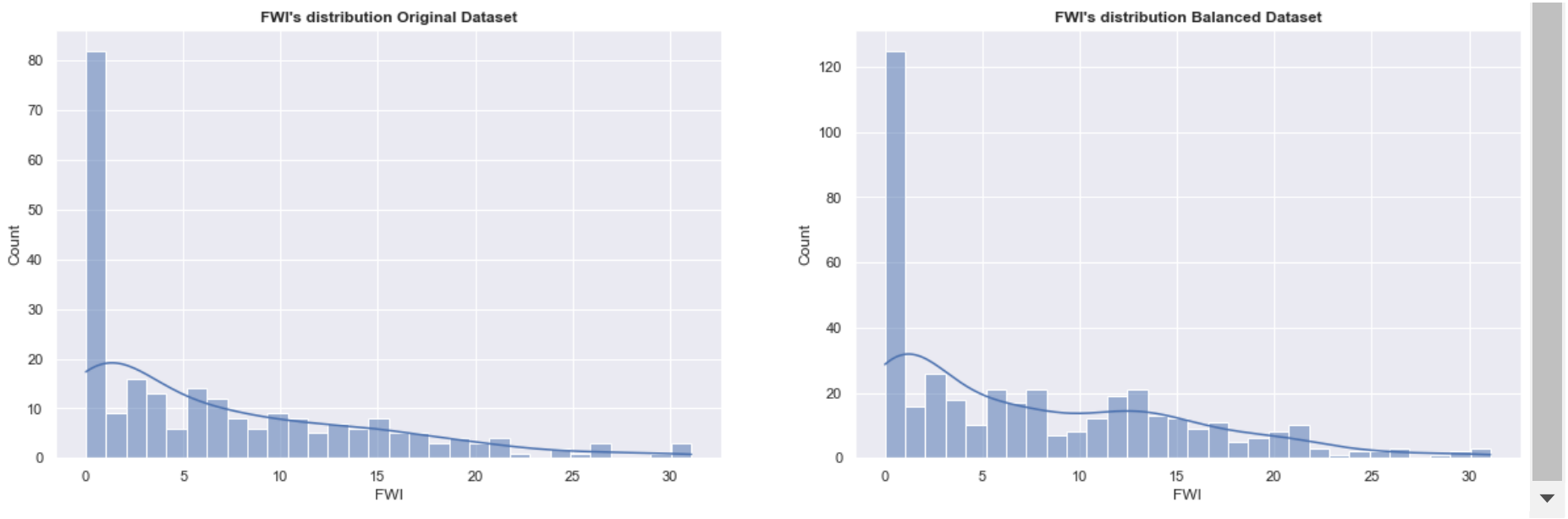

DC's distribution Balanced Dataset
Count
DC


ISI's distribution Original Dataset
Count
ISI
0
5
10
15
20
25
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


ISI's distribution Balanced Dataset
Count
ISI
0
10
20
30
40
50
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


BUI's distribution Original Dataset
Count
BUI
0
5
10
15
20
25
0
10
20
30
40
50
60
70


BUI's distribution Balanced Dataset
Count
BUI
0
10
20
30
40
50
60
0
10
20
30
40
50
60
70



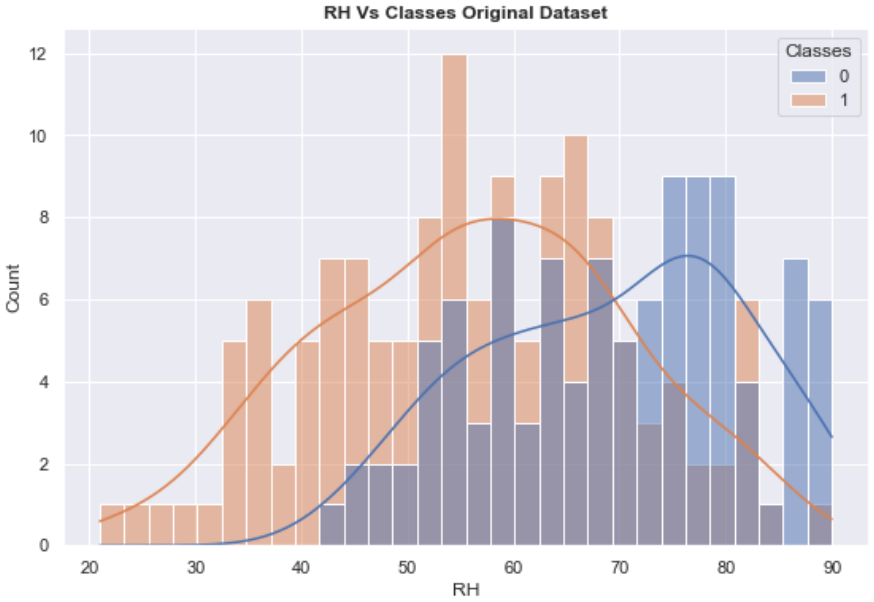
Observations

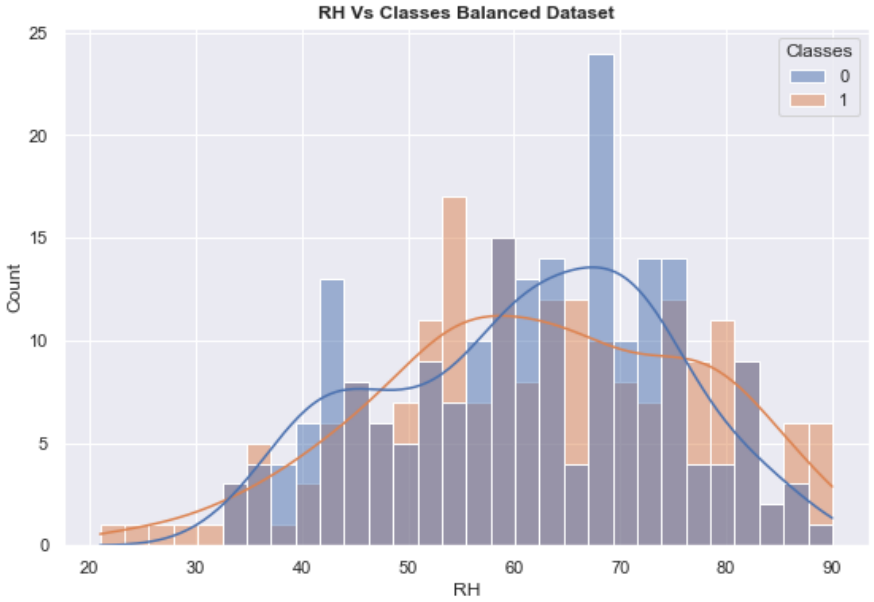
1. All continuous features have similar distribution, the only difference is increase in count of observation in each feature with subtle change in distribution.

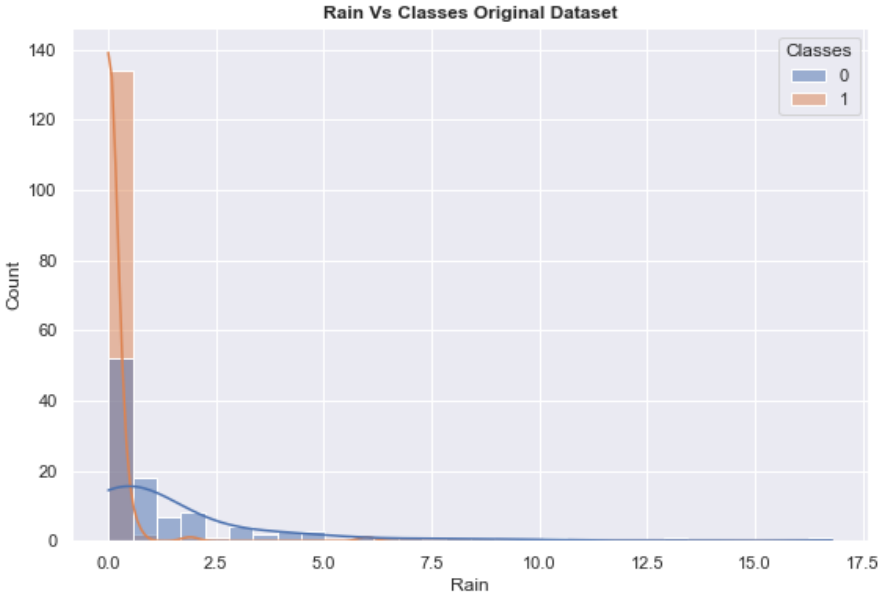
9.3.4 Comparing Continuous features with Target feature for Original and Balanced Dataset

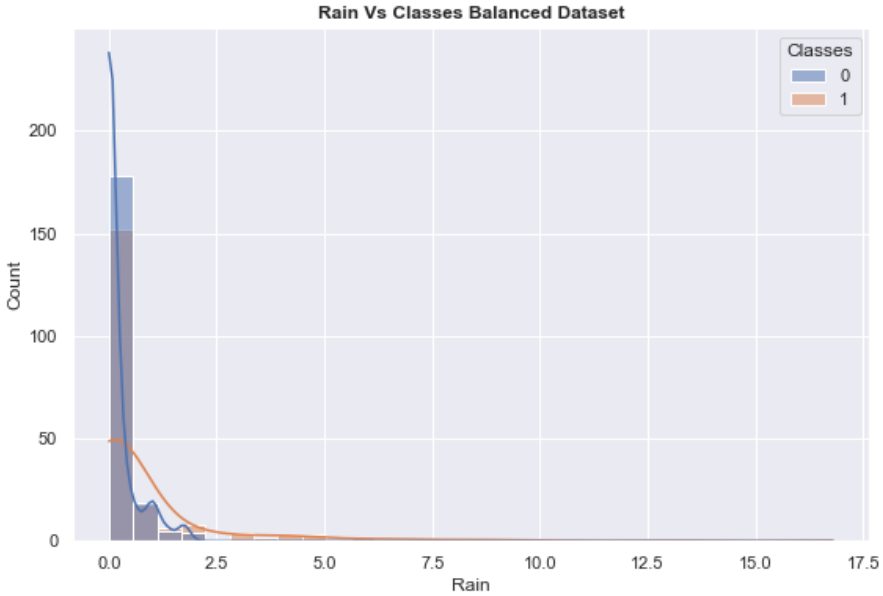
In [190…

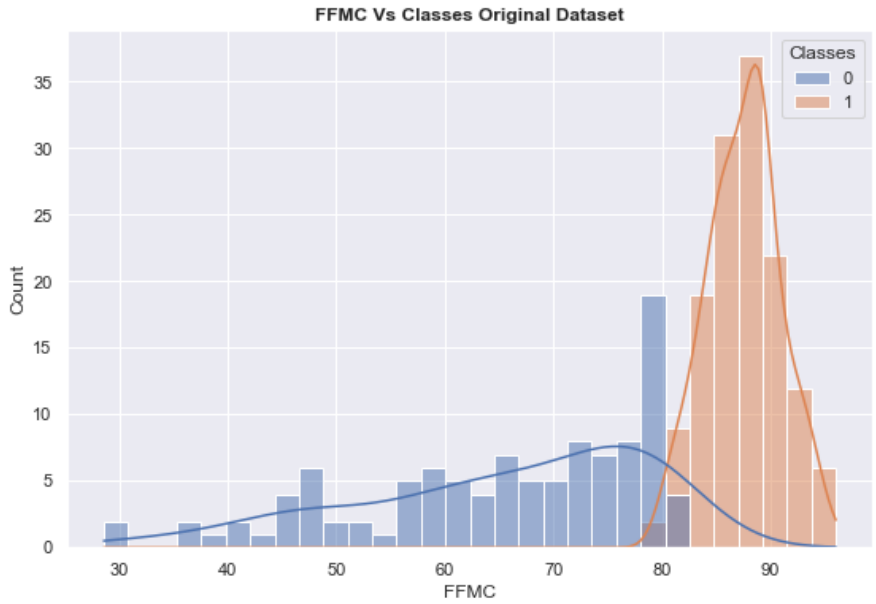
```
### Comparing Continuous numerical features with Classes

for i in continuous_features:
    plt.figure(figsize=(20,6))
    plt.subplot(121)
    sns.histplot(data=dataset, x=i, kde=True, bins=30, color='blue', hue='Classes')
    plt.title("{} Vs Classes Original Dataset".format(i),fontweight="bold")

    plt.subplot(122)
    sns.histplot(data=data_bal, x=i, kde=True, bins=30, color='blue', hue='Classes')
    plt.title("{} Vs Classes Balanced Dataset".format(i),fontweight="bold")
```

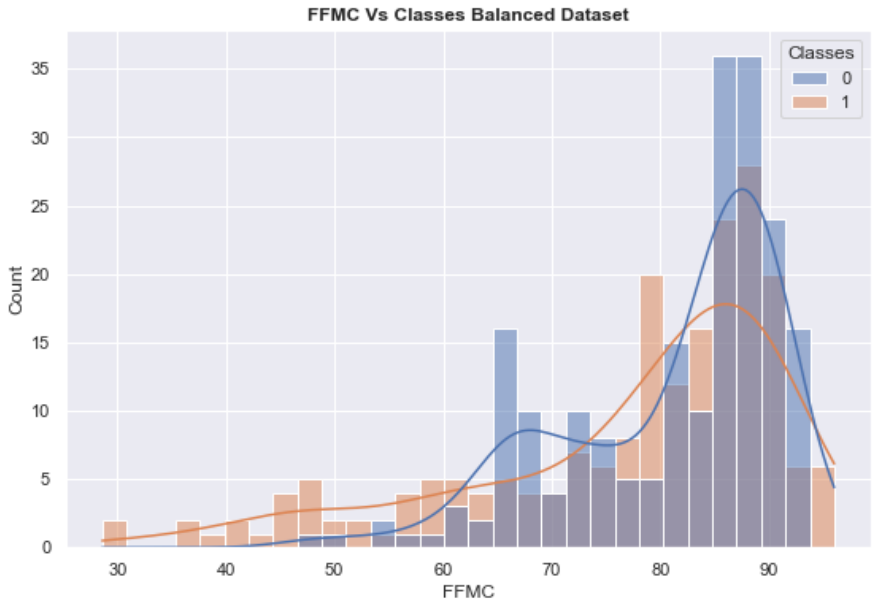

RH Vs Classes Original Dataset
Classes
0
1
Count
RH

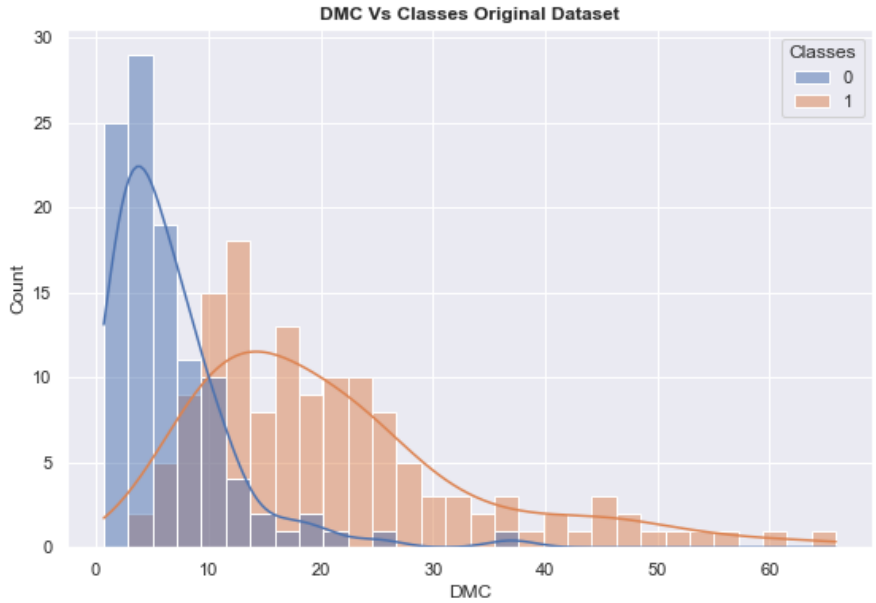

RH Vs Classes Balanced Dataset
Classes
0
1
Count
RH

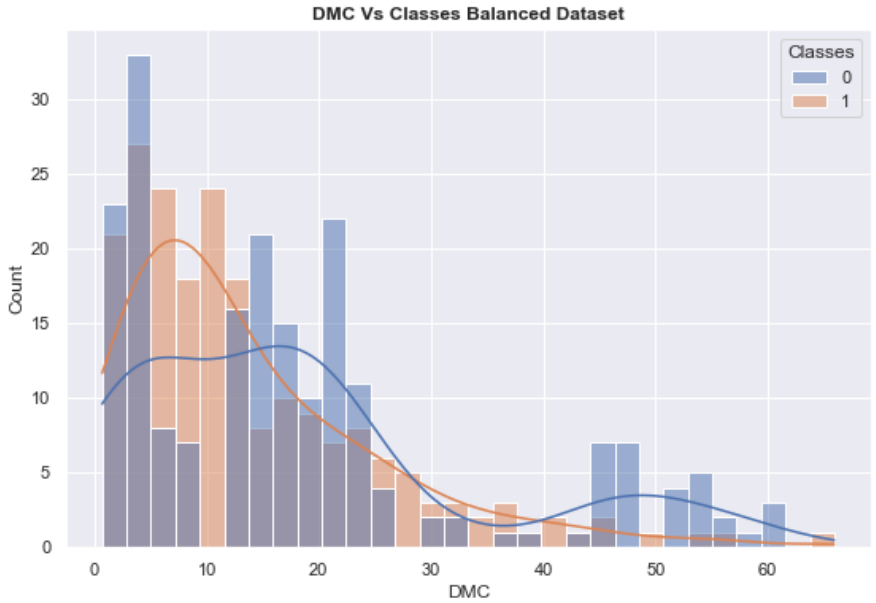

Rain Vs Classes Original Dataset
Classes
0
1
Count
Rain

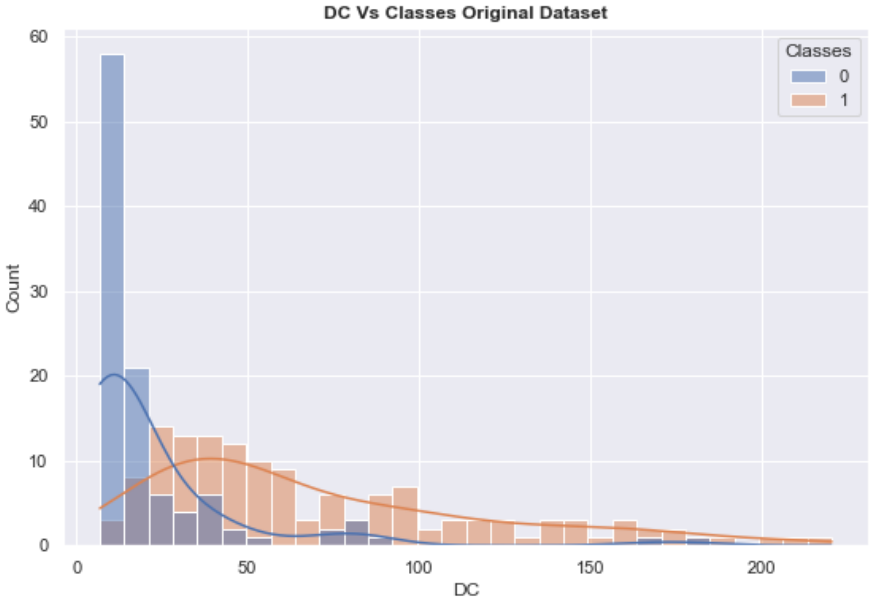

Rain Vs Classes Balanced Dataset
Classes
0
1
Count
Rain

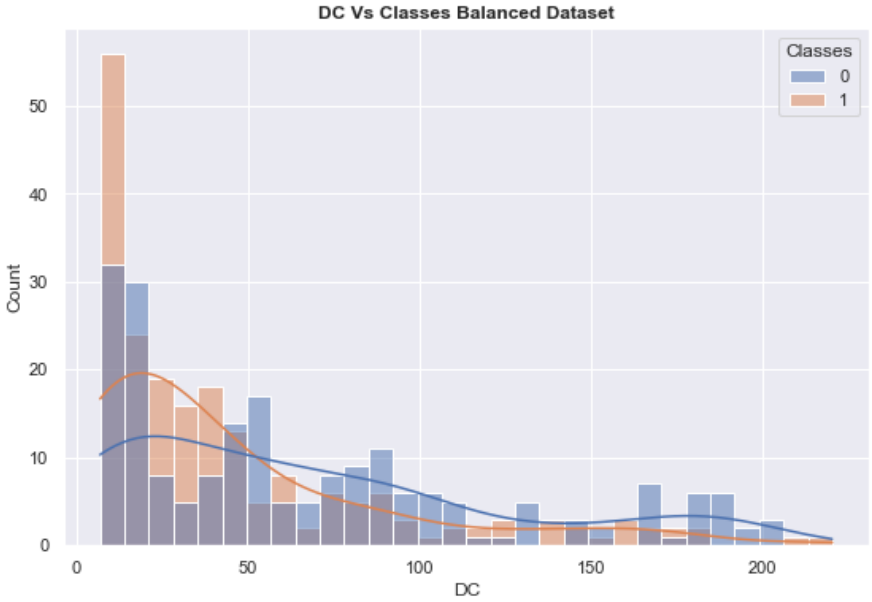

FFMC Vs Classes Original Dataset
Classes
0
1
Count
FFMC


FFMC Vs Classes Balanced Dataset
Classes
0
1
Count
FFMC

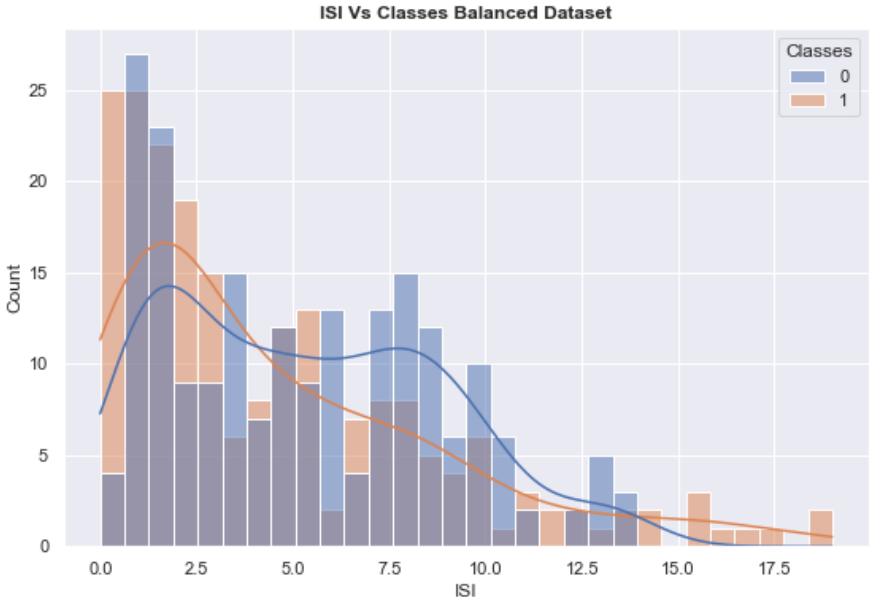

DMC Vs Classes Original Dataset
Classes
0
1
Count
DMC

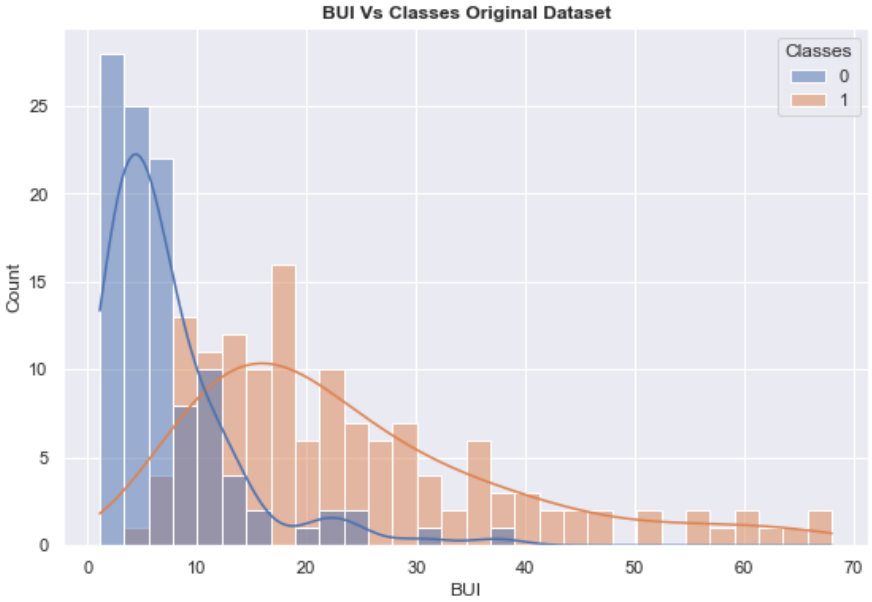

DMC Vs Classes Balanced Dataset
Classes
0
1
Count
DMC

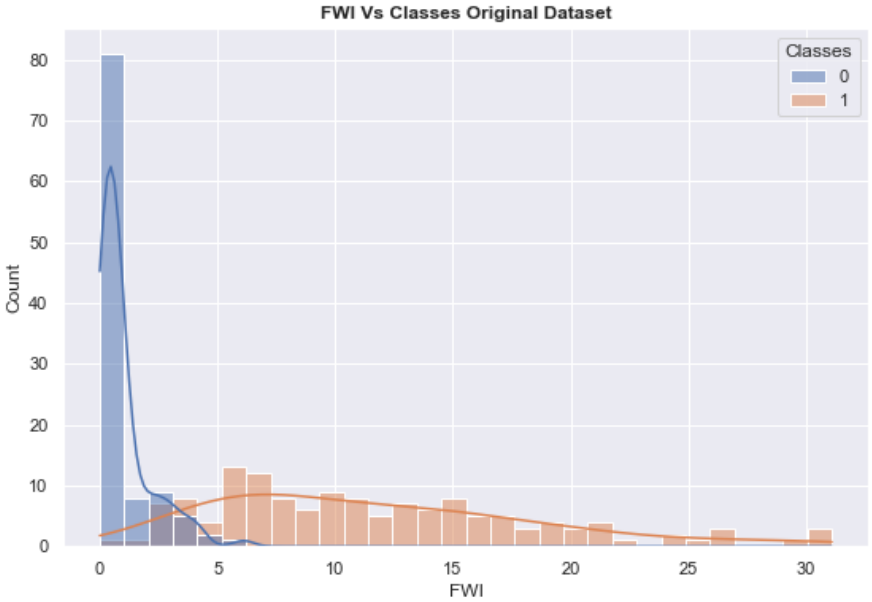

DC Vs Classes Original Dataset
Classes
0
1
Count
DC
0
10
20
30
40
50
60
0
50
100
150
200

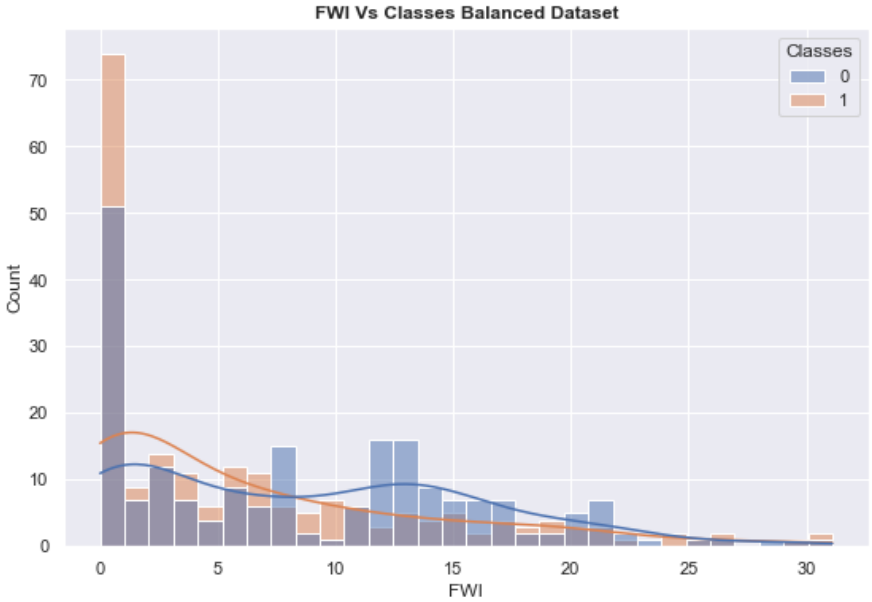

DC Vs Classes Balanced Dataset
Classes
0
1
Count
DC
0
10
20
30
40
50
0
50
100
150
200


ISI Vs Classes Original Dataset
Classes
0
1
Count
ISI
0
5
10
15
20
25
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


ISI Vs Classes Balanced Dataset
Classes
0
1
Count
ISI
0
5
10
15
20
25
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5







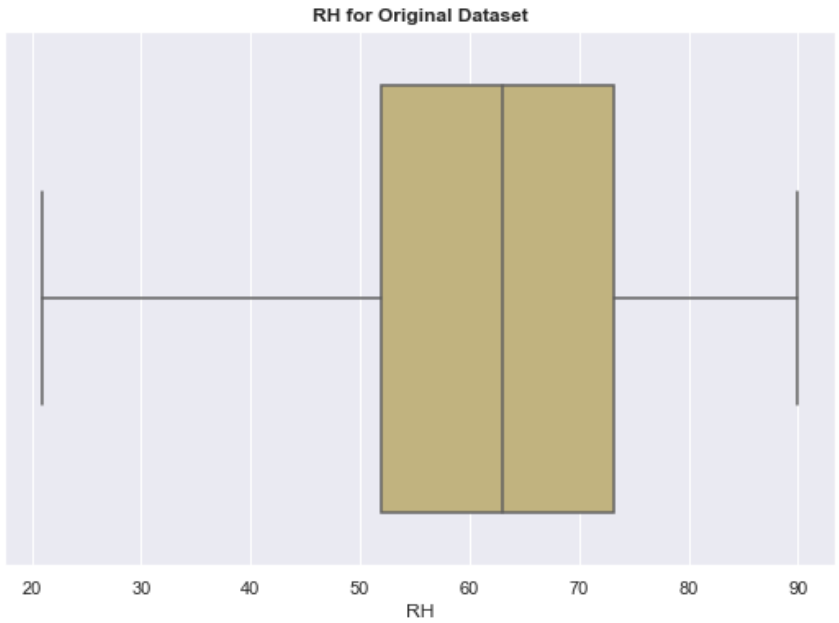
Observations

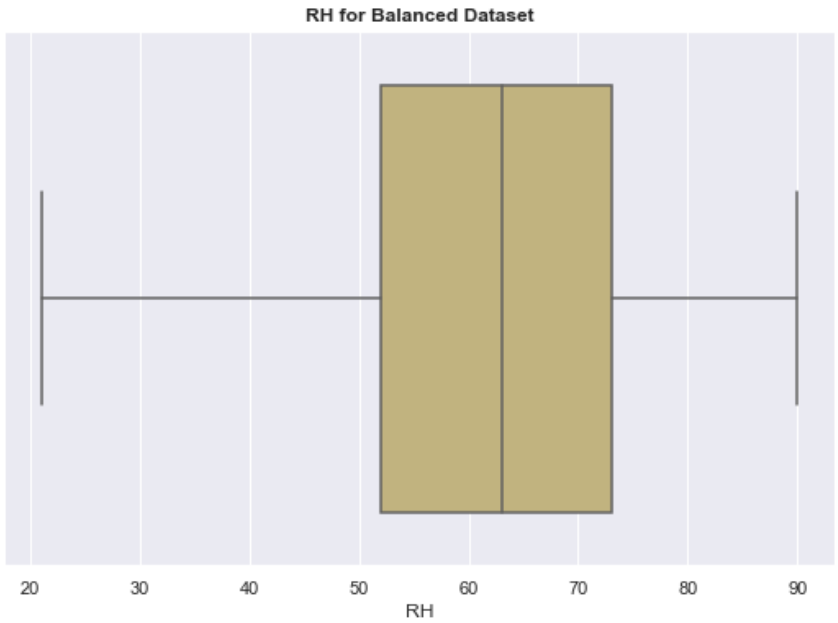
1. Originally for RH in range of 20 to 70 there were more fire cases, but now for the same range there are more non fire cases.

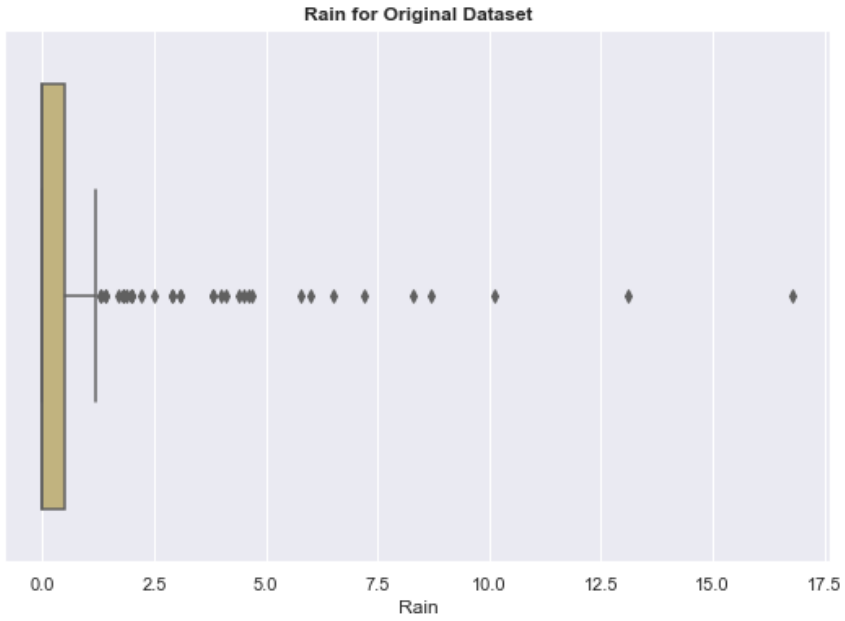
2. Originally for Rain in range 0 to 1 mm there were more fire Cases, but noe for the same range there are more non fire cases.
3. For FFMC there were more non fire cases in range of 30 to 80, but now the ratio of fire to non fire in same range is almost equal.
4. For FFMC in range of 80 to 90, now there are more non fire cases than fire cases, which is in contradiction to original dataset.
5. For DMC the ratio of fire to non fire in complete range is now almost equal.
6. For DC there are more fire cases than non fire cases now in range of 0 to 40.
7. For ISI now there are more no. of fire cases than non fire cases in range of 0 to 3, where as from 5 to 13 the no of cases of non fire have increased significantly.
8. For BUI in range of 18 to 70 there is significant increase in cases of non fire than fire.
9. For FWI now there are more cases of fire in range of 0 to 7 as compared to non fire, where as there is significant increase in non fire cases in range 10 to 30.

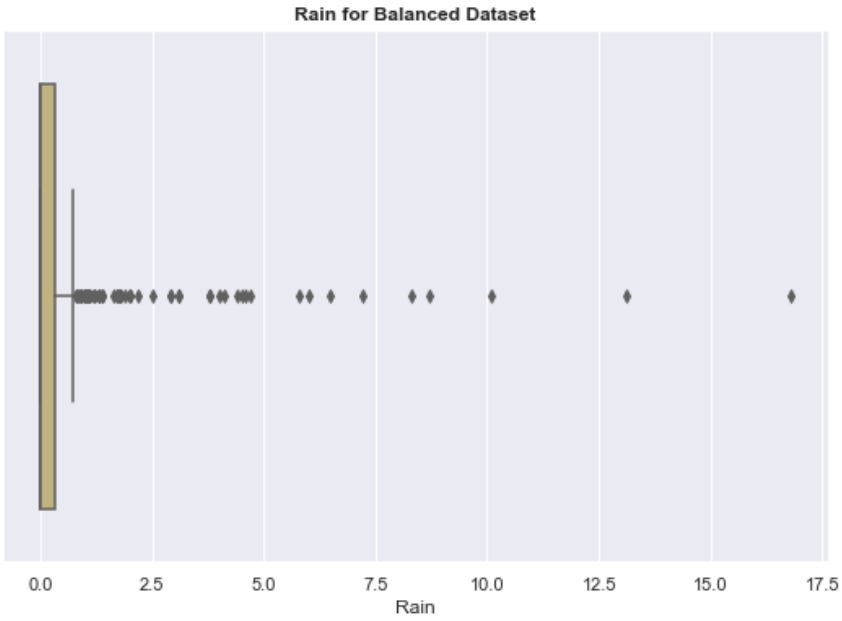
In [191…

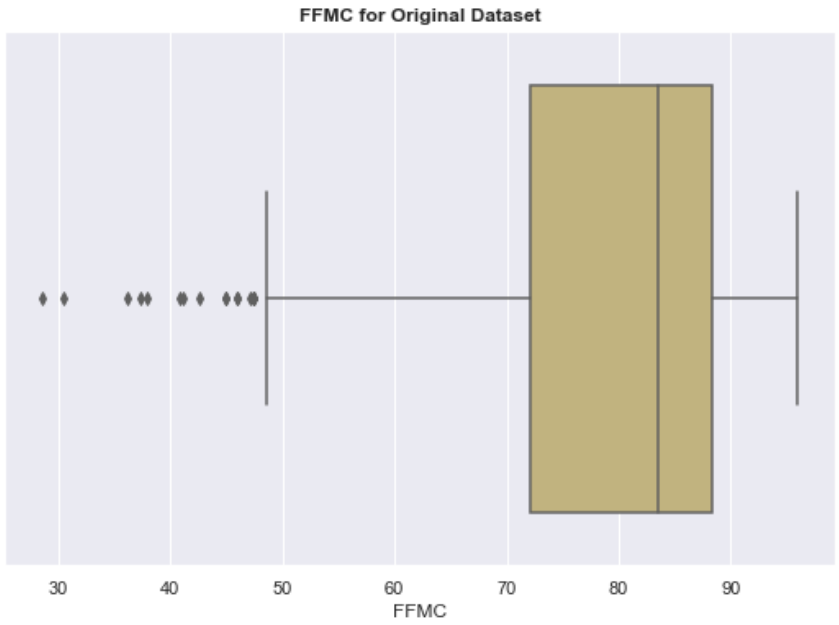
```
### Checking outliers in numerical features

for i in continuous_features:
    plt.figure(figsize=(20,6))
    plt.subplot(121)
    sns.boxplot(data=dataset, x=i, color='y')
    plt.title("{} for Original Dataset".format(i), fontweight="bold")

    plt.subplot(122)
    sns.boxplot(data=data_bal, x=i, color='y')
    plt.title("{} for Balanced Dataset".format(i), fontweight="bold")
```

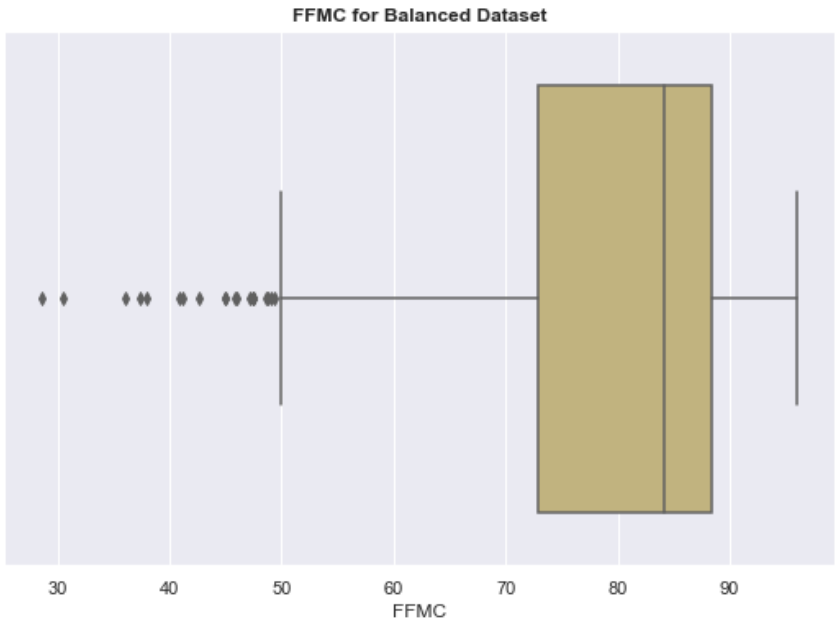

RH for Original Dataset
20
30
40
50
60
70
80
90
RH

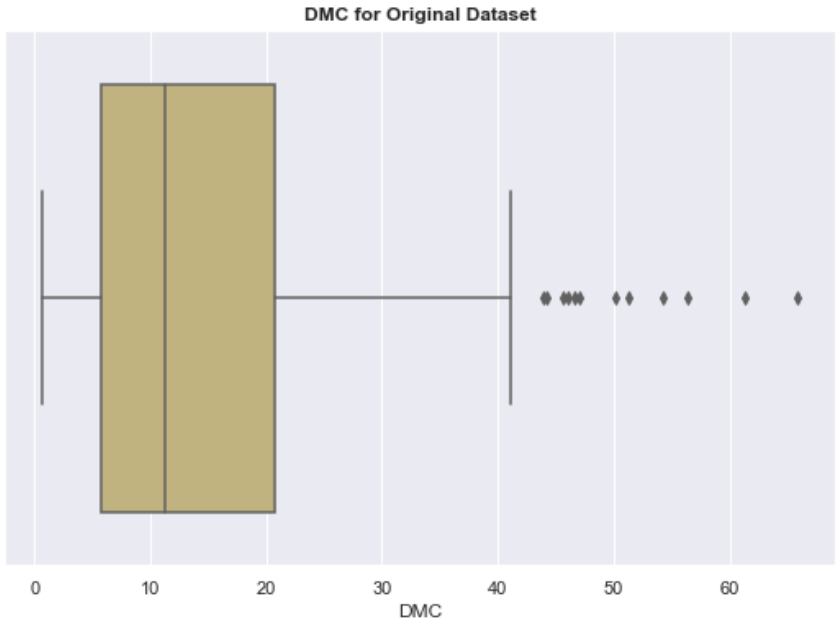

RH for Balanced Dataset
20
30
40
50
60
70
80
90
RH

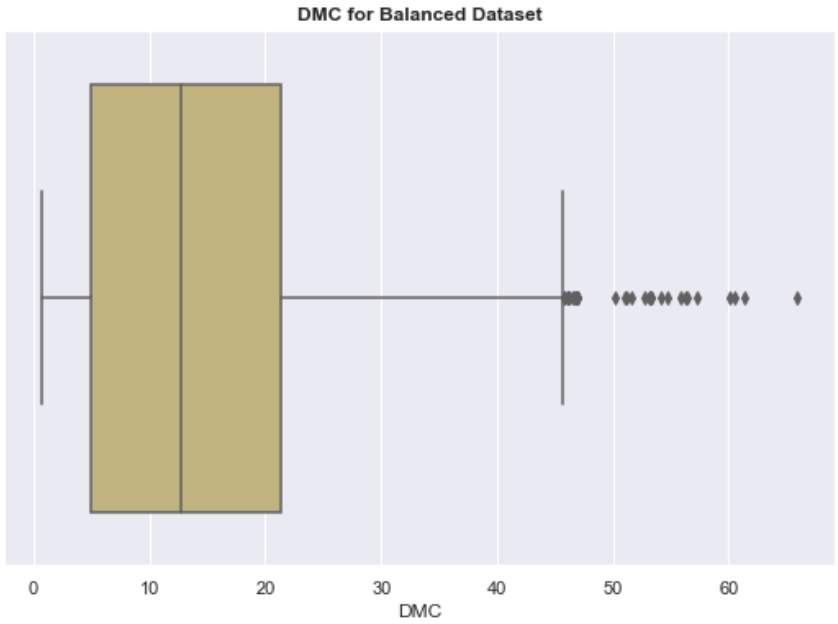

Rain for Original Dataset
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Rain

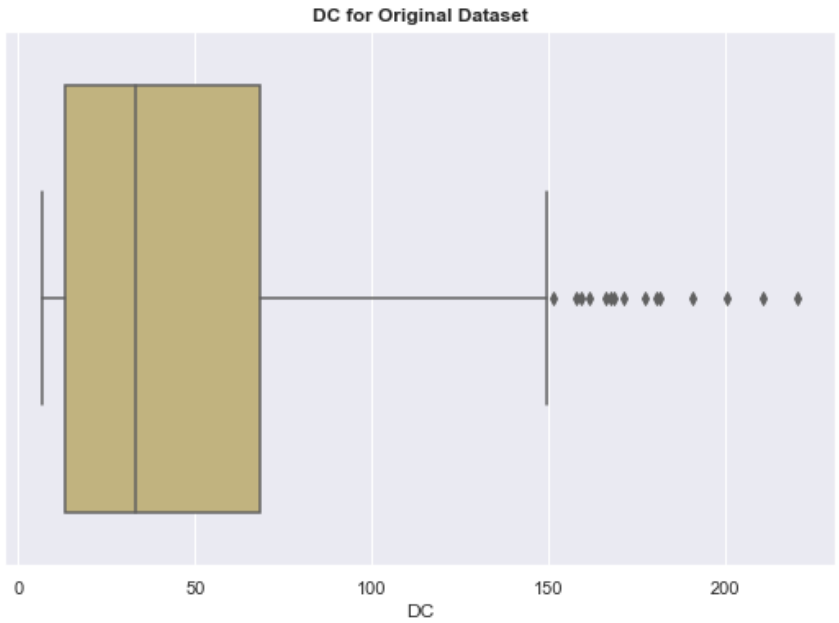

Rain for Balanced Dataset
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Rain

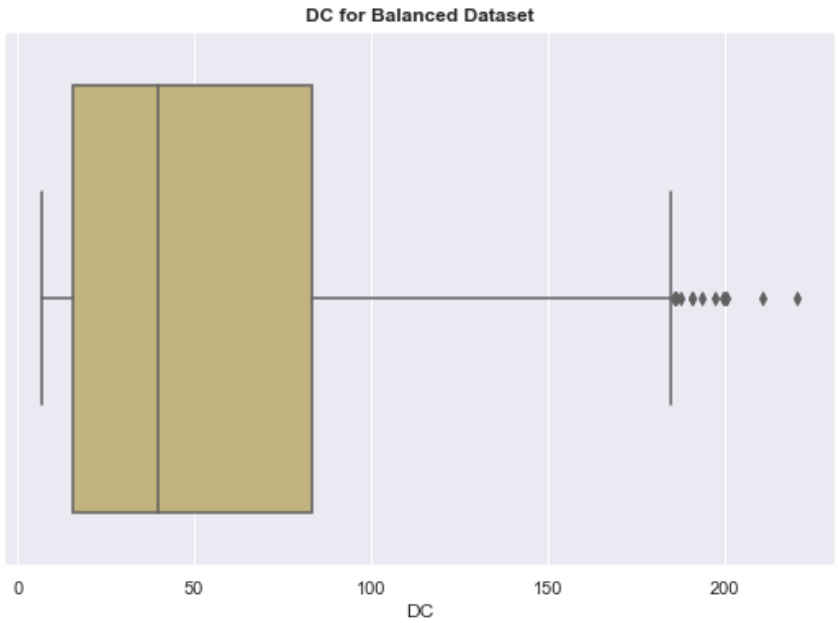

FFMC for Original Dataset
30
40
50
60
70
80
90
FFMC

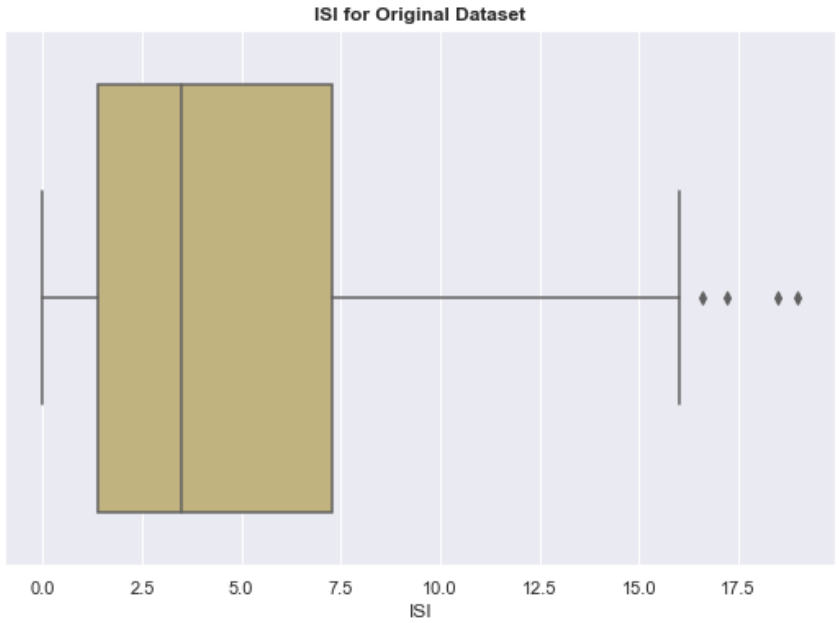

FFMC for Balanced Dataset
30
40
50
60
70
80
90
FFMC

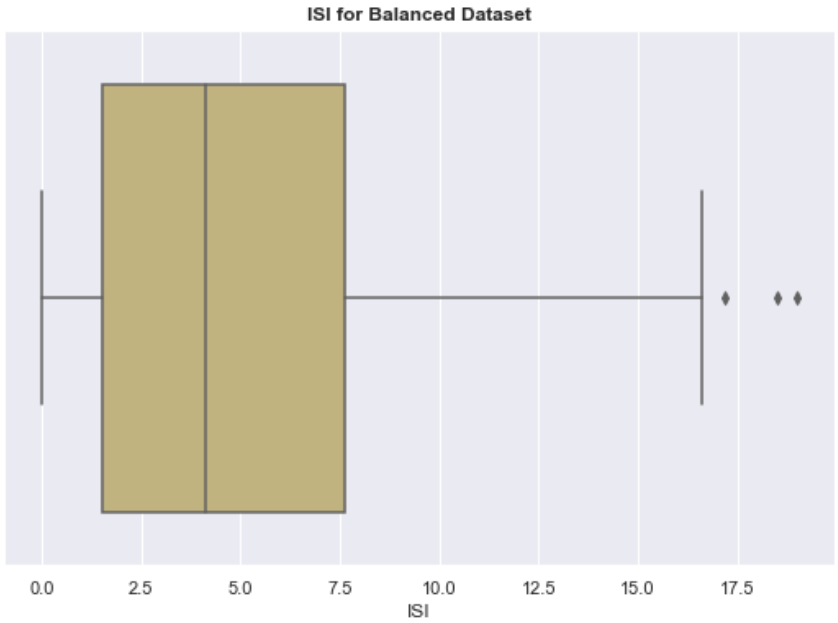

DMC for Original Dataset
0
10
20
30
40
50
60
DMC

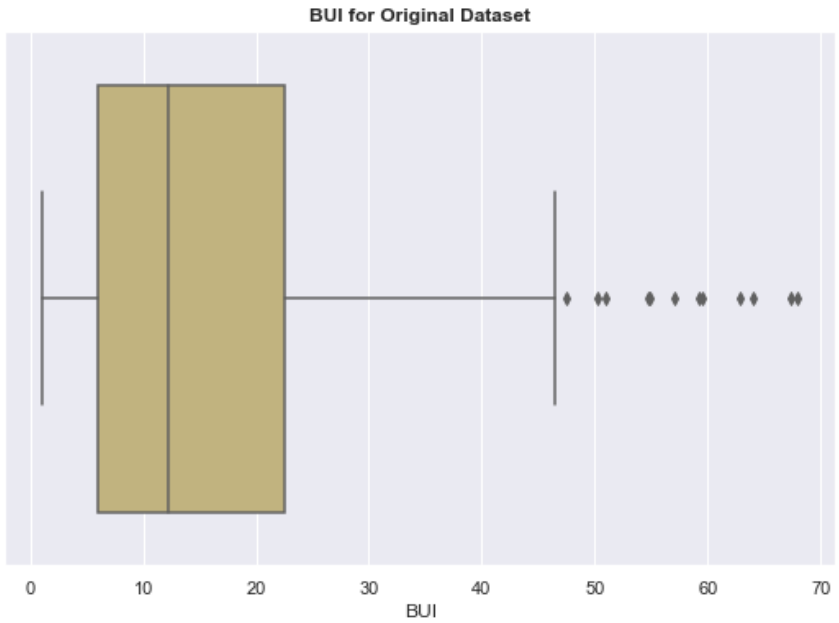

DMC for Balanced Dataset
0
10
20
30
40
50
60
DMC

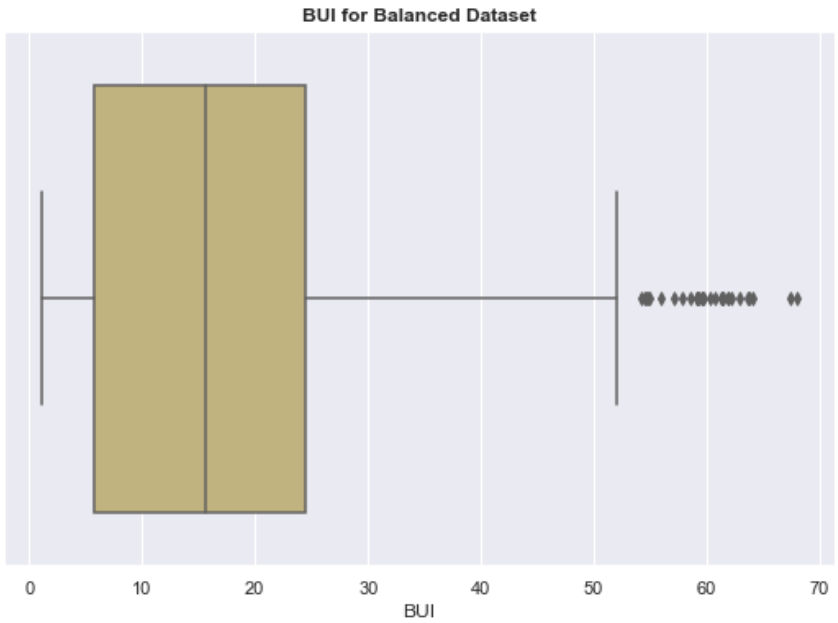

DC for Original Dataset
0
50
100
150
200
DC

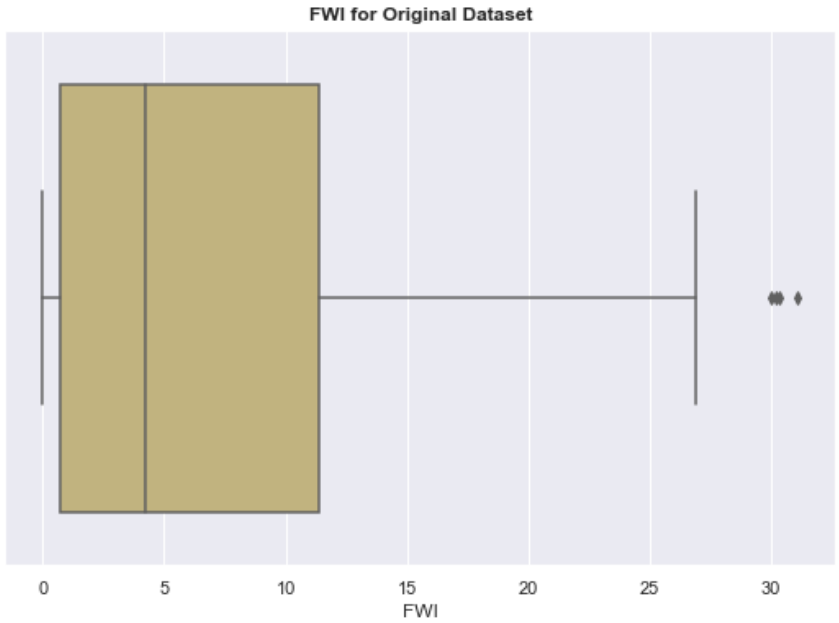

DC for Balanced Dataset
0
50
100
150
200
DC

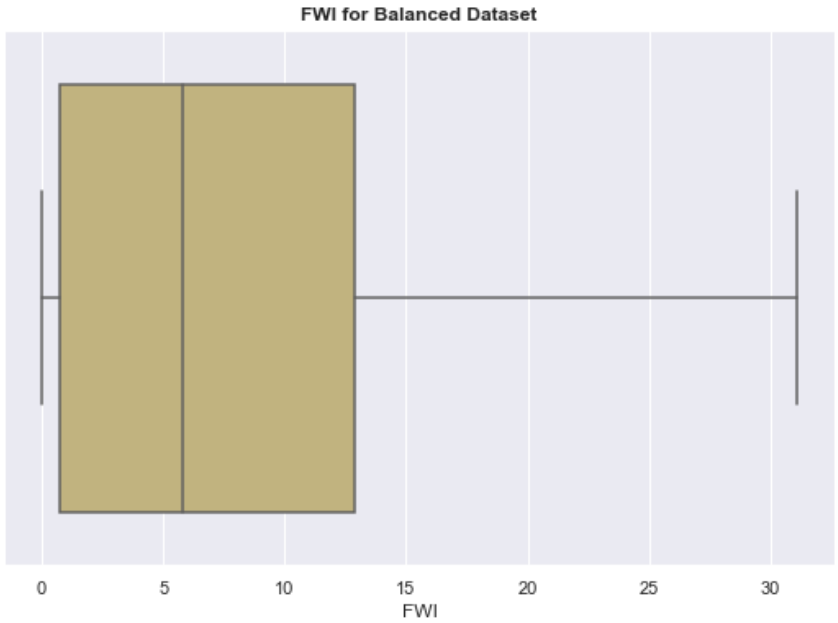

ISI for Original Dataset
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
ISI


ISI for Balanced Dataset
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
ISI









Observations

1. As no of Observations has increased in balance dataset, so there is significant increase in outliers in features that had outliers in original dataset.
2. There is no outlier addition in features that had no outliers in original dataset.
3. There is reduction in outliers for DC, ISI and FWI in comparision to original dataset.

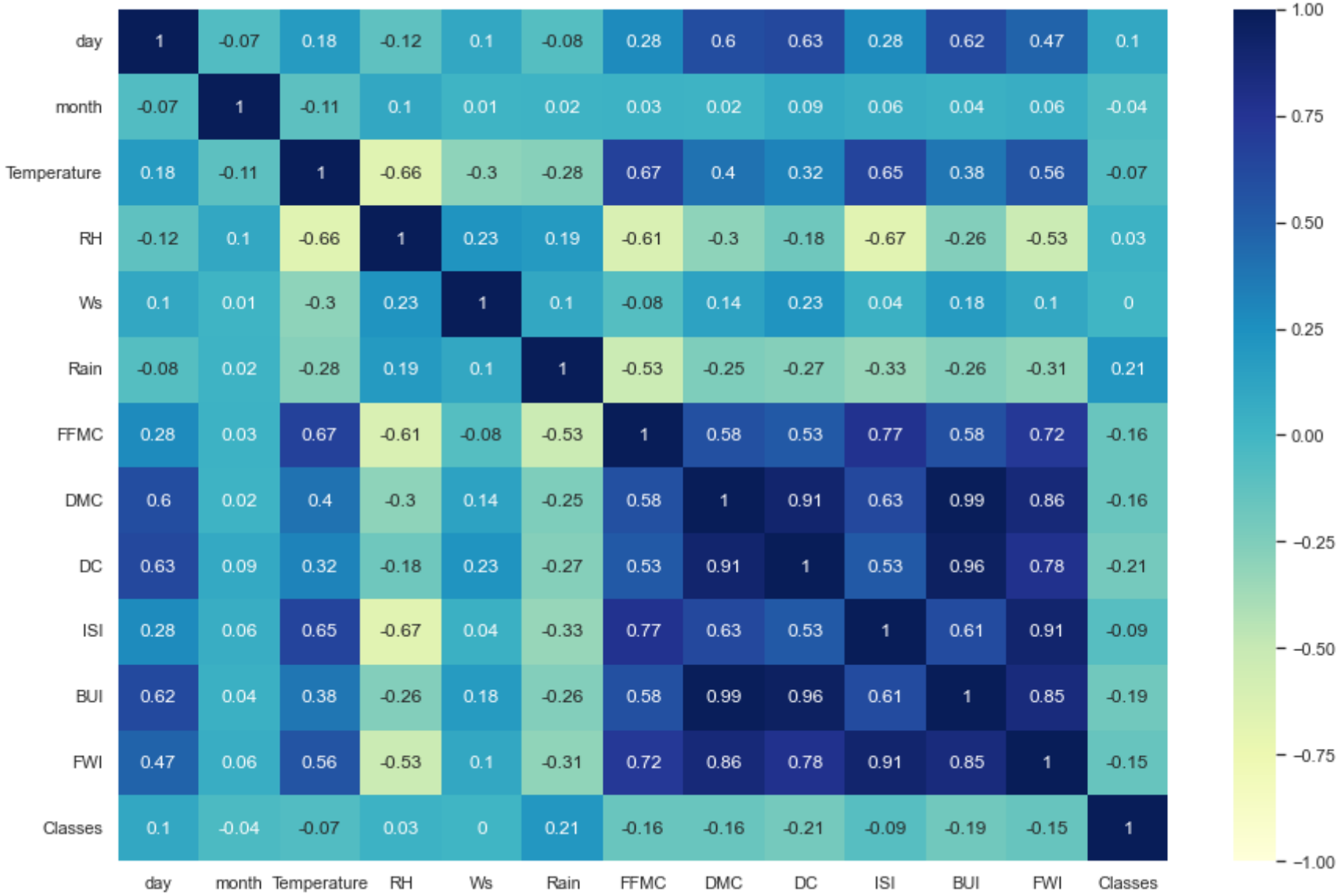
In [194…
```
corr1= round(data_bal[numerical_features+['Classes']].corr(),2)
corr1
```

Out[194]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
day	1.00	-0.07	0.18	-0.12	0.10	-0.08	0.28	0.60	0.63	0.28	0.62	0.47	0.10
month	-0.07	1.00	-0.11	0.10	0.01	0.02	0.03	0.02	0.09	0.06	0.04	0.06	-0.04
Temperature	0.18	-0.11	1.00	-0.66	-0.30	-0.28	0.67	0.40	0.32	0.65	0.38	0.56	-0.07
RH	-0.12	0.10	-0.66	1.00	0.23	0.19	-0.61	-0.30	-0.18	-0.67	-0.26	-0.53	0.03
Ws	0.10	0.01	-0.30	0.23	1.00	0.10	-0.08	0.14	0.23	0.04	0.18	0.10	0.00
Rain	-0.08	0.02	-0.28	0.19	0.10	1.00	-0.53	-0.25	-0.27	-0.33	-0.26	-0.31	0.21
FFMC	0.28	0.03	0.67	-0.61	-0.08	-0.53	1.00	0.58	0.53	0.77	0.58	0.72	-0.16
DMC	0.60	0.02	0.40	-0.30	0.14	-0.25	0.58	1.00	0.91	0.63	0.99	0.86	-0.16
DC	0.63	0.09	0.32	-0.18	0.23	-0.27	0.53	0.91	1.00	0.53	0.96	0.78	-0.21
ISI	0.28	0.06	0.65	-0.67	0.04	-0.33	0.77	0.63	0.53	1.00	0.61	0.91	-0.09
BUI	0.62	0.04	0.38	-0.26	0.18	-0.26	0.58	0.99	0.96	0.61	1.00	0.85	-0.19
FWI	0.47	0.06	0.56	-0.53	0.10	-0.31	0.72	0.86	0.78	0.91	0.85	1.00	-0.15
Classes	0.10	-0.04	-0.07	0.03	0.00	0.21	-0.16	-0.16	-0.21	-0.09	-0.19	-0.15	1.00

In [195…
```
### Plotting heatmap for visualising the correlation between features
sns.set(rc={'figure.figsize':(15,10)})
sns.heatmap(data=corr1, annot=True,  vmin=-1, vmax=1, cmap="YlGnBu")
```

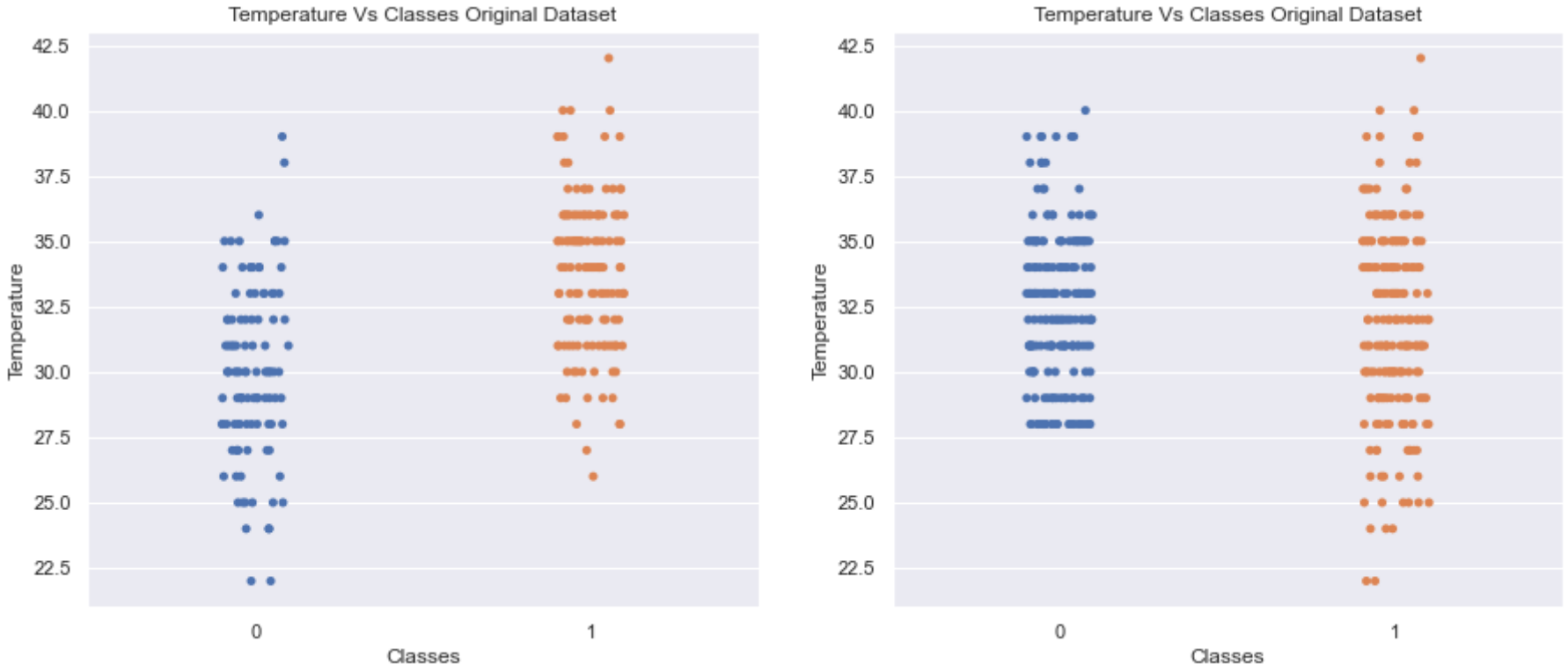
Out[195]:
```
<AxesSubplot:>
```

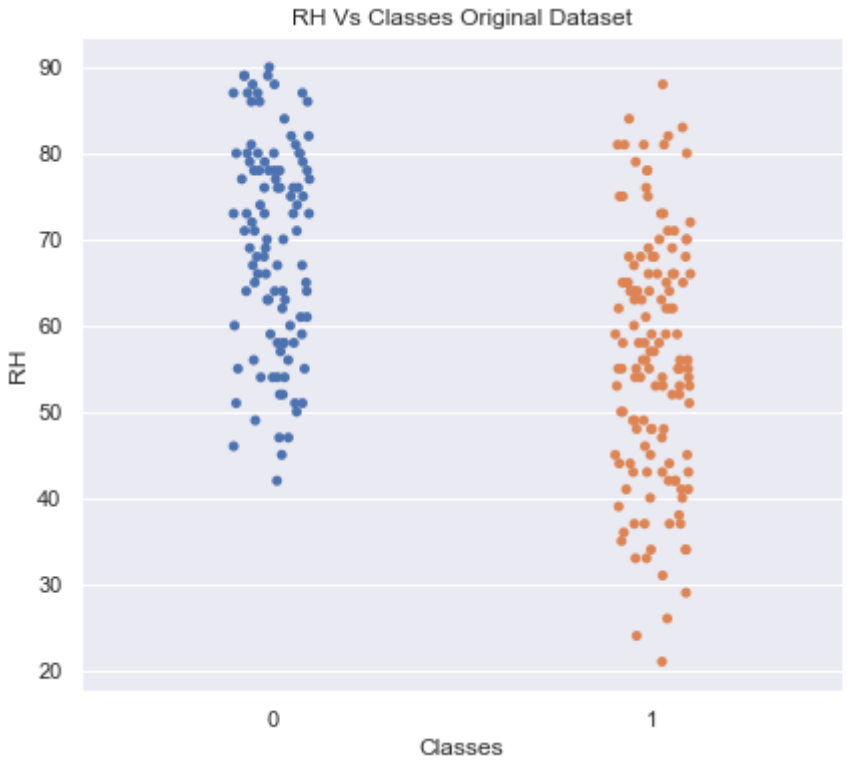


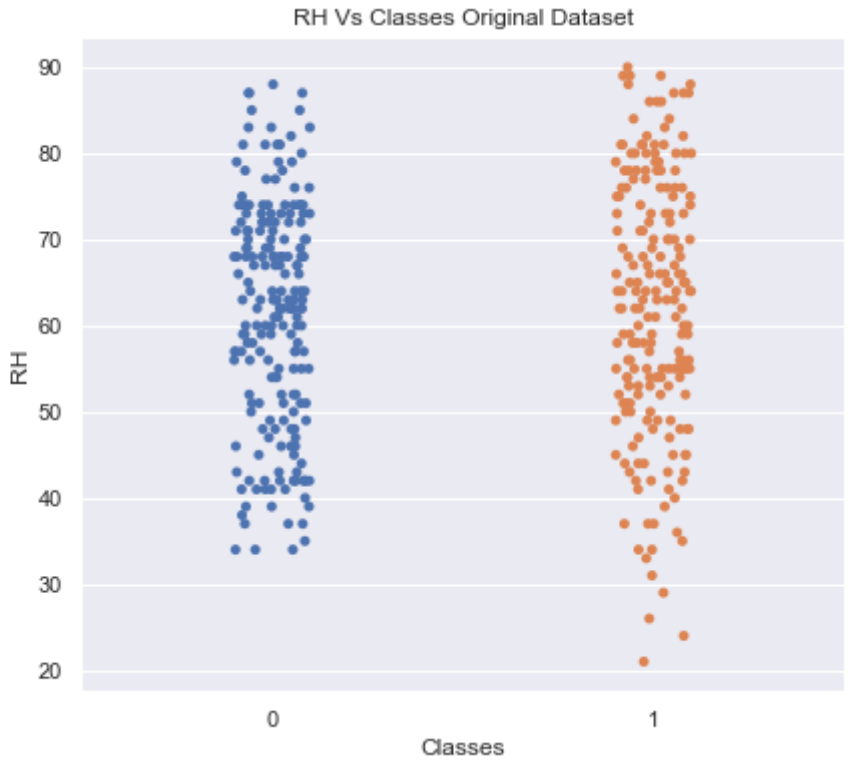
Observations

1. The only significant change is that FFMC, DMC, DC, ISI, BUI, FWI had correlation coefficient more than 0.55 with Classes, now there is sginifcant decrease in correlation coefficient of these features with Classes.
2. Due to this there may be reduction in performance of new model with respect to old model.

In [193…
```
num_feature_custom=[feature for feature in numerical_features if feature not in ['day', 'month']]

for i in num_feature_custom:
    plt.figure(figsize=(15,6))
    plt.subplot(121)
    sns.stripplot(data=dataset, y=i, x='Classes')
    plt.title("{} Vs Classes Original Dataset".format(i))

    plt.subplot(122)
    sns.stripplot(data=data_bal, y=i, x='Classes')
    plt.title("{} Vs Classes Original Dataset".format(i))
```



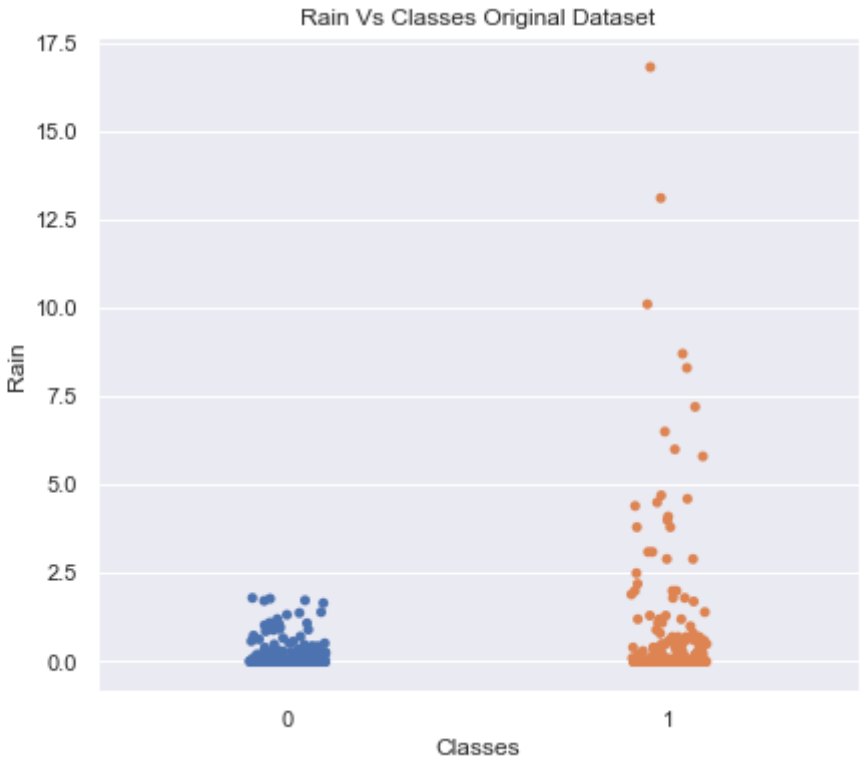

RH Vs Classes Original Dataset
RH
Classes
0
1
90
80
70
60
50
40
30
20

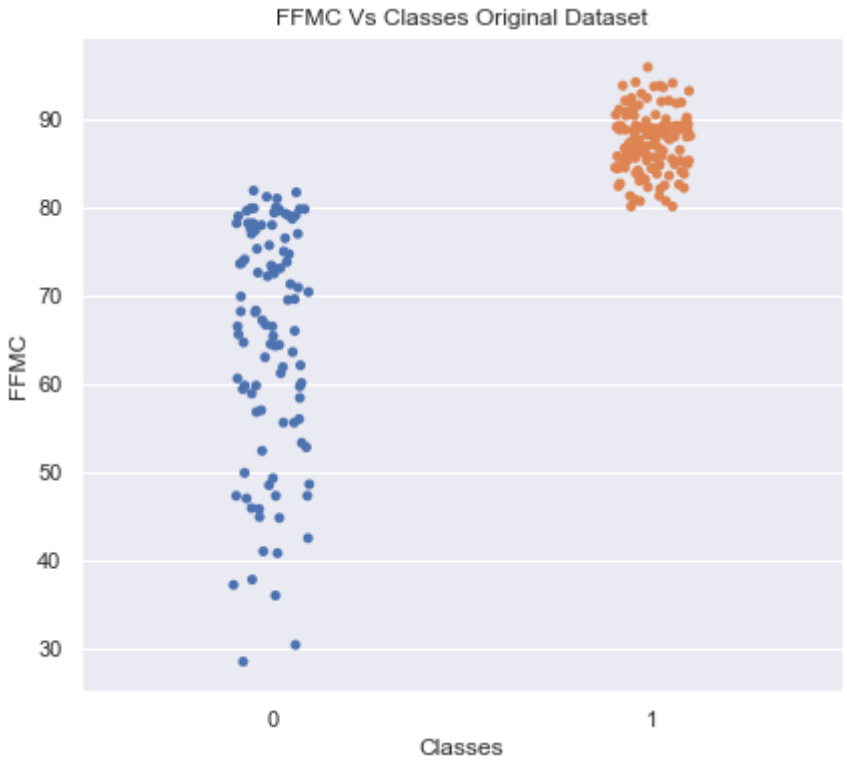

RH Vs Classes Original Dataset
RH
Classes
0
1
90
80
70
60
50
40
30
20

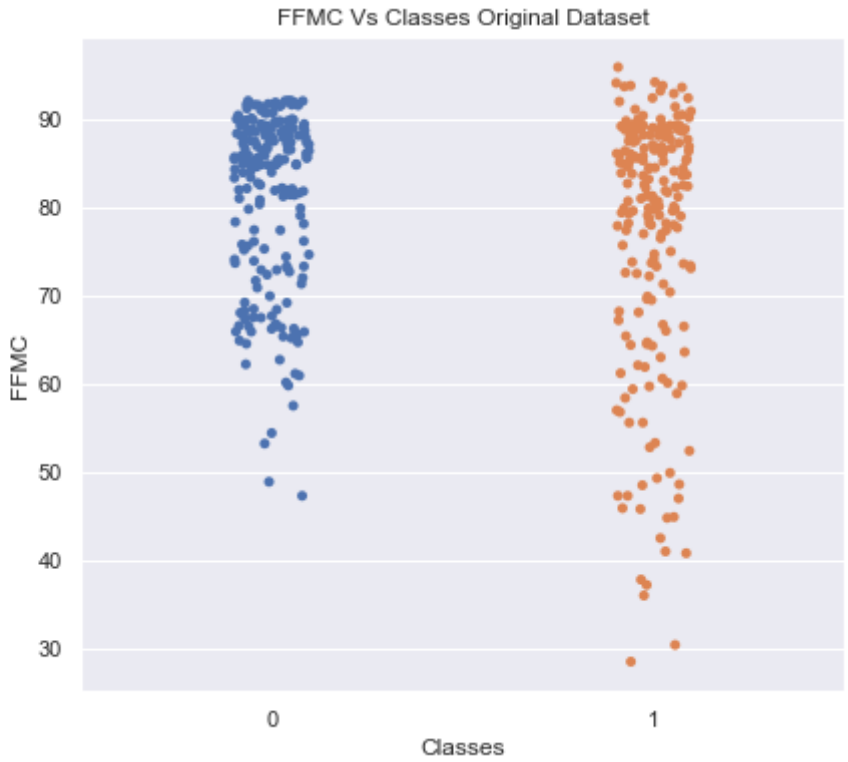

Ws Vs Classes Original Dataset
Ws
Classes
30
25
20
15
10
5
0
1

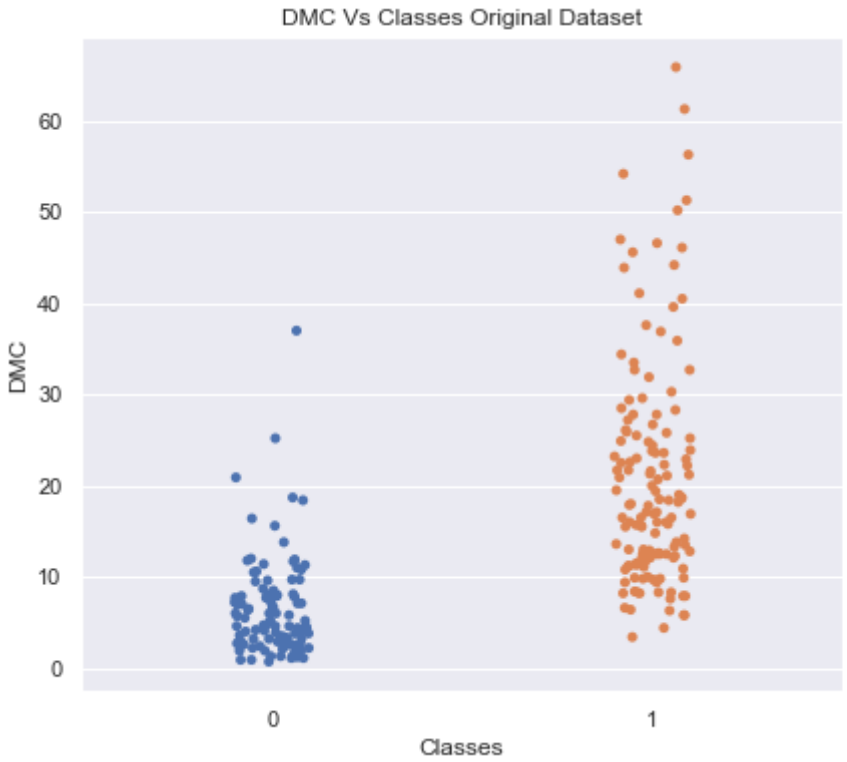

Ws Vs Classes Original Dataset
Ws
Classes
30
25
20
15
10
5
0
1

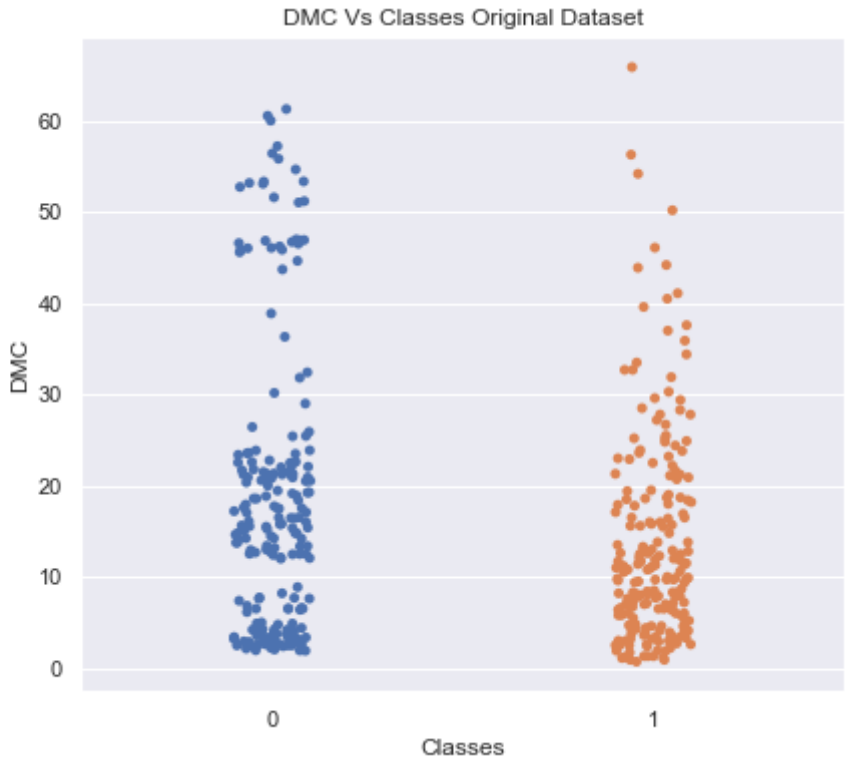

Rain Vs Classes Original Dataset
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Rain
0
1
Classes

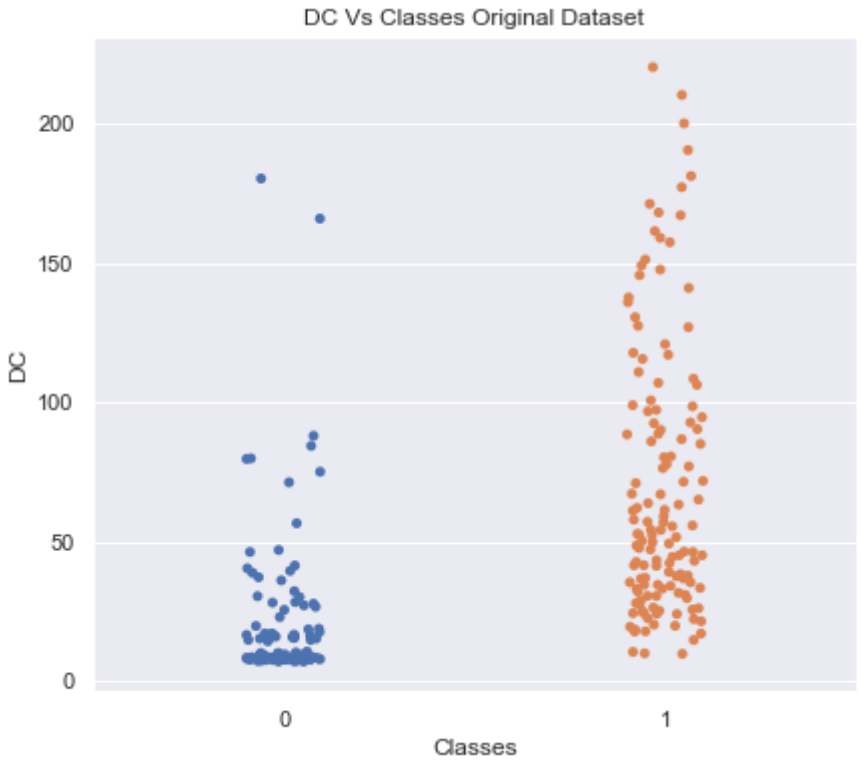

Rain Vs Classes Original Dataset
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Rain
0
1
Classes

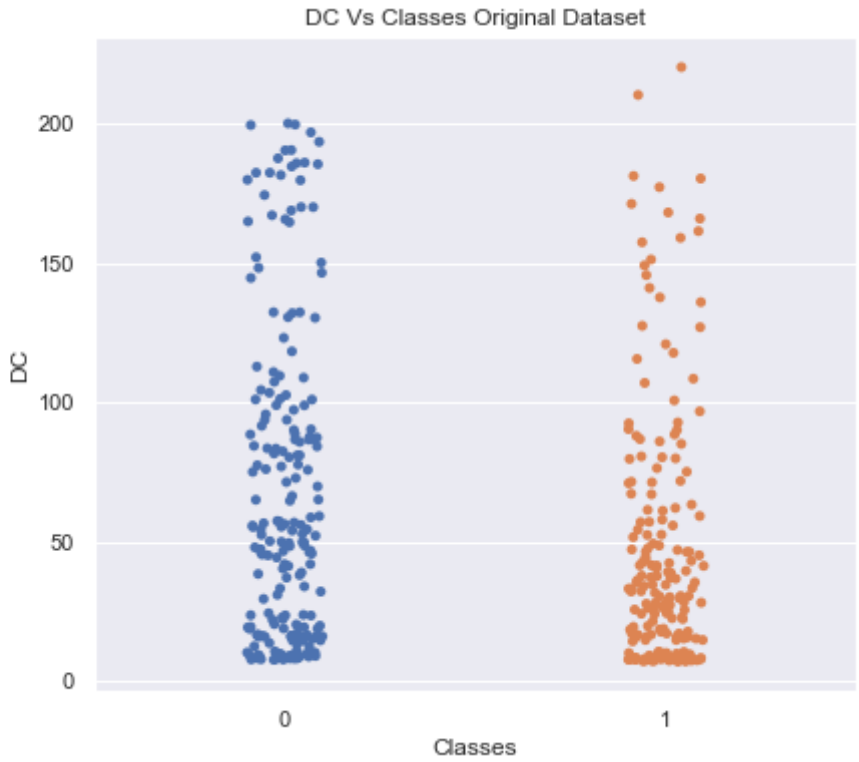

FFMC Vs Classes Original Dataset
FFMC
90
80
70
60
50
40
30
0
1
Classes


FFMC Vs Classes Original Dataset
FFMC
90
80
70
60
50
40
30
0
1
Classes


DMC Vs Classes Original Dataset
DMC
Classes
0
10
20
30
40
50
60
0
1

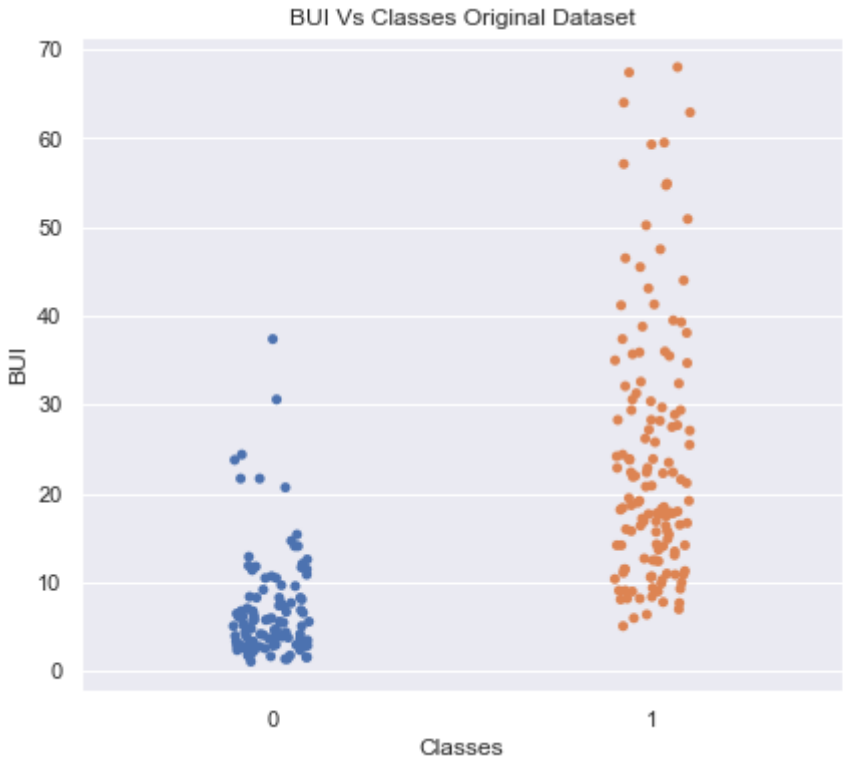

DMC Vs Classes Original Dataset
DMC
Classes
0
10
20
30
40
50
60
0
1

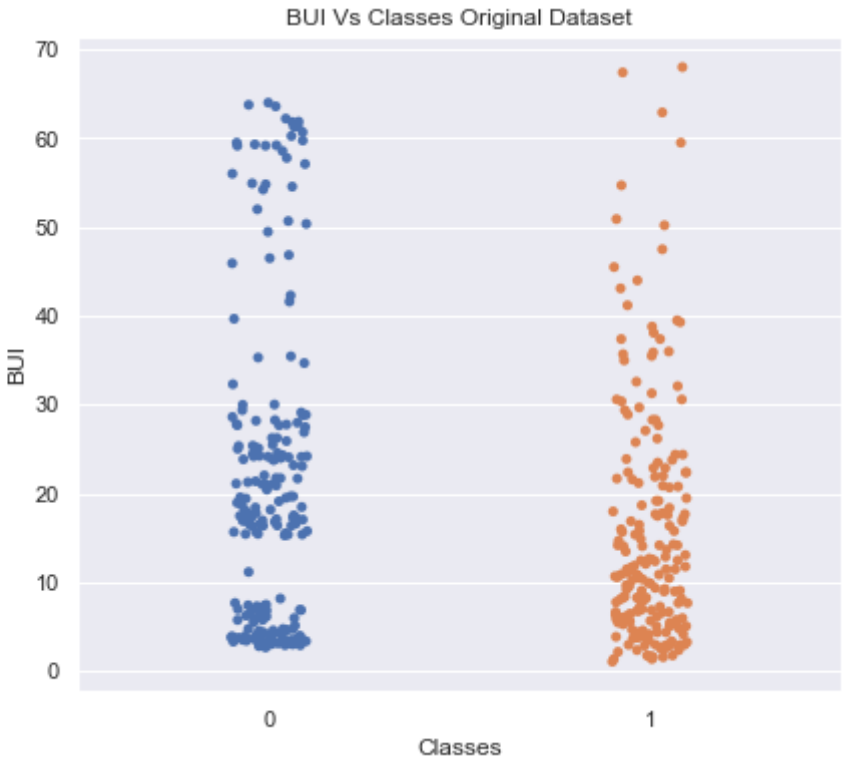

DC Vs Classes Original Dataset
DC
200
150
100
50
0
0
1
Classes

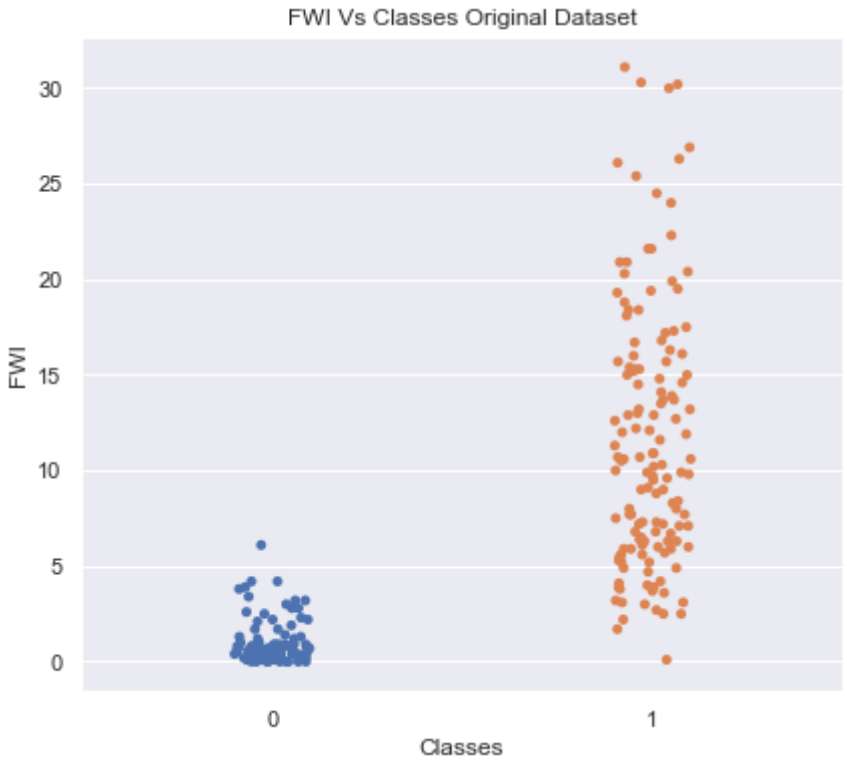

DC Vs Classes Original Dataset
DC
200
150
100
50
0
0
1
Classes

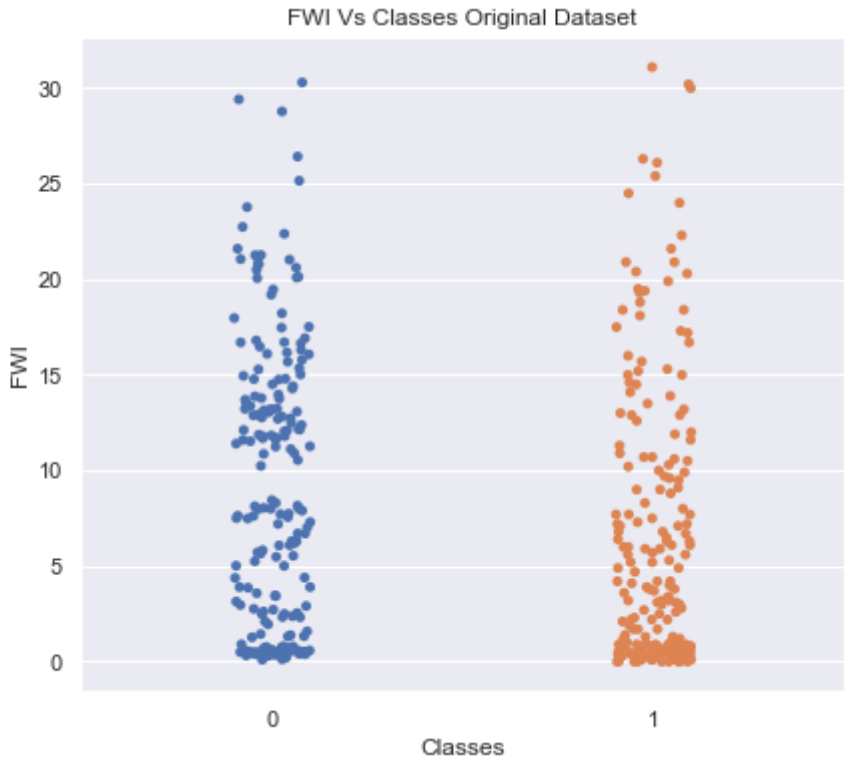

ISI Vs Classes Original Dataset
ISI
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
1
Classes


ISI Vs Classes Original Dataset
ISI
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
1
Classes


BUI Vs Classes Original Dataset
BUI
Classes
0
1
0
10
20
30
40
50
60
70


BUI Vs Classes Original Dataset
BUI
Classes
0
1
0
10
20
30
40
50
60
70





Observations

1. Now there are more No. of cases of fire in temperature range between 22 to 27 as compared to original dataset, whereas the ratio of cases of fire to non fire has almost balenced out in temperature range of 27 to 37, which was in favour of fire for original dataset.
2. There is significant increase in No of cases of fire for RH greater than 60 as compared to original dataset.
3. There is increase in No of cases of fire for FFMC less than 80 as compared to original dataset.
4. There is Significant increase in No of cases of non fire for DMC greater than 20 as compared to original dataset.
5. There is Significant increase in No of cases of non fire for DC greater than 40 as compared to original dataset.
6. There is Significant increase in No of cases of non fire for ISI greater than 2.5 as compared to original dataset.
7. There is Significant increase in No of cases of fire for ISI less than 2.5 as compared to original dataset.
8. There is Significant increase in No of cases of non fire for BUI greater than 18 as compared to original dataset.
9. There is Significant increase in No of cases of fire for BUI less than 10 as compared to original dataset.
10. There is Significant increase in No of cases of non fire for FWI greater than 5 as compared to original dataset.

11. There is Significant increase in No of cases of fire for FWI less than 1 as compared to original dataset.

9.4 Splitting data into Training and Test data

In [106…
```
### random state train test split will be same with all people using random_state=16

X_train1, X_test1, y_train1, y_test1 = train_test_split(X_bal, y_bal, test_size=0.30, random_state=16)

X_train1.head()
```

Out[106]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
332	27	6	31	52	14	0.201401	81.346126	16.460877	85.813259	3.054529	22.058076	5.017461	0.000000
347	16	7	34	61	17	0.000000	87.530776	12.113845	52.116931	7.312928	17.562469	10.239700	0.000000
285	12	7	34	64	13	0.000000	88.661216	25.416078	80.283282	7.070152	28.634042	12.726808	0.149240
328	3	9	28	72	17	0.086970	69.312093	2.030299	15.778179	1.478179	3.043329	0.539090	0.130299
190	18	6	31	78	14	0.300000	56.900000	1.900000	8.000000	0.700000	2.400000	0.200000	0.000000

In [107…
```
y_train1.head()
```

Out[107]:
```
332    0
347    0
285    0
328    0
190    1
Name: Classes, dtype: int64
```

In [108…
```
X_test1.head()
```

Out[108]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
327	1	6	29	57	17	0.074373	65.460355	3.350418	7.653714	1.275209	3.371077	0.491736	0.0
330	17	7	34	60	17	0.000000	87.795511	12.060898	52.434613	7.492948	17.858972	10.546793	0.0
26	14	6	30	78	20	0.500000	59.000000	4.600000	7.800000	1.000000	4.400000	0.400000	0.0
319	18	8	35	55	18	0.000000	89.291521	20.581845	112.842769	9.611244	28.210048	16.169032	0.0
30	21	9	31	55	11	0.000000	87.800000	16.500000	57.900000	5.400000	19.200000	8.300000	0.0

In [109…
```
y_train1.head()
```

Out[109]:
```
332    0
347    0
285    0
328    0
190    1
Name: Classes, dtype: int64
```

In [110…
```
### both will have same shape
X_train1.shape, y_train1.shape
```

Out[110]:
```
((288, 13), (288,))
```

In [111…
```
### both will have same shape
X_test1.shape, y_test1.shape
```

Out[111]:
```
((124, 13), (124,))
```

9.4 Standardisation/ Feature scaling the dataset

In [112…
```
### using fit_transform to Standardize the train data
X_train1=scaler.fit_transform(X_train1)
X_train1
```

Out[112]:
```
array([[ 1.35962522, -1.42233545, -0.34932259, ...,  0.19270021,
        -0.36344564, -0.97986971],
       [ 0.10458656, -0.45164206,  0.55161687, ..., -0.08467537,
         0.33626637, -0.97986971],
       [-0.35179114, -0.45164206,  0.55161687, ...,  0.59843243,
         0.66950651, -0.6487928 ],
       ...,
       [-1.15045211,  0.51905132,  0.55161687, ..., -0.45872519,
        -0.21839951, -0.97986971],
       [ 0.56096425,  0.51905132,  0.85193002, ...,  0.68479639,
         1.78148924,  0.97325949],
       [ 1.70190849,  1.48974471, -2.1512015 , ..., -1.0201884 ,
        -1.02232137, -0.97986971]])
```

In [113…
```
### here using transform only to avoid data leakage
### (training mean and training std will be used for standardisation when we use transform)
X_test1=scaler.transform(X_test1)
X_test1
```

Out[113]:
```
array([[-1.6068298 , -1.42233545, -0.9499489 , ..., -0.96027366,
        -0.9698338 , -0.97986971],
       [ 0.21868098, -0.45164206,  0.55161687, ..., -0.06638133,
         0.37741282, -0.97986971],
       [-0.12360229, -1.42233545, -0.64963574, ..., -0.89678989,
        -0.98212528, -0.97986971],
       ...,
       [ 0.3327754 ,  0.51905132,  1.15224317, ...,  0.52846289,
         1.12147026, -0.97986971],
       [ 0.7891531 ,  0.51905132,  1.45255632, ...,  1.63904947,
         1.38944422,  1.23854831],
       [ 0.67505867,  0.51905132,  1.15224317, ...,  1.22566446,
         1.42964031, -0.97986971]])
```

9.5 Model

1.0 Logistic Regression 2.0

In [152…
```
### Creating a Logistic regression object
logistic_reg1=LogisticRegression()
logistic_reg1
```

Out[152]:
```
LogisticRegression
LogisticRegression()
```

In [153…
```
### Passing independant and dependant training data to the Logistic regression model created initially
logistic_reg1.fit(X_train1,y_train1)
```

Out[153]:
```
LogisticRegression
LogisticRegression()
```

1.1 Using Above Model to get prediction for test data

In [154…
```
logistic_reg_pred1=logistic_reg1.predict(X_test1)
logistic_reg_pred1
```

Out[154]:
```
array([0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1,
       0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1,
       0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0], dtype=int64)
```

1.2.0 Performance Metrics

1.2.1 Confusion Matrix

In [155…
```
confusion_mat1=confusion_matrix(y_test1, logistic_reg_pred1)
confusion_mat1
```

Out[155]:
```
array([[45, 19],
       [25, 35]], dtype=int64)
```

In [156…
```
truly_positive=confusion_mat1[0][0]
falsely_positive=confusion_mat1[0][1]
falsely_negative=confusion_mat1[1][0]
truly_negative=confusion_mat1[1][1]
```

1.2.2 Accuracy Score

In [157…
```
### accuracy using accuracy_score
accuracy1=round(accuracy_score(y_test1, logistic_reg_pred1),4)
accuracy1
```

Out[157]:
```
0.6452
```

In [158…
```
### manual calcualtion for accuracy
accuracy_manual1=round(((truly_positive+truly_negative)/(truly_positive+falsely_positive+falsely_negative+truly_negative)),4)
print("Accuracy of our model is {}".format(accuracy_manual1))
```

```
Accuracy of our model is 0.6452
```

1.2.3 Precision Score

In [159…
```
precision_manual1=round(truly_positive/(truly_positive+falsely_positive),4)
print("Precision of our model is {}".format(precision_manual1))
```

```
Precision of our model is 0.7031
```

1.2.4 Recall Score

In [160…
```
recall_manual1=round(truly_positive/(truly_positive+falsely_negative),4)
print("Recall of our model is {}".format(recall_manual1))
```

```
Recall of our model is 0.6429
```

1.2.5 F-1 Score

1. Giving equal importance to falsely positive and falsely negative

In [161…
```
f1_score1=2*(precision_manual1*recall_manual1)/(precision_manual1+recall_manual1)
print("F-1 Score of our model is {} ".format(round(f1_score1,4)))
```

```
F-1 Score of our model is 0.6717
```

1.2.6 Classification Report

In [162…
```
print(classification_report(y_test1, logistic_reg_pred1))
```

```
              precision    recall  f1-score   support

           0       0.64      0.70      0.67        64
           1       0.65      0.58      0.61        60

    accuracy                           0.65       124
   macro avg       0.65      0.64      0.64       124
weighted avg       0.65      0.65      0.64       124
```

10.0 Saving the Model for imbalanced dataset

In [163…
```
### Writing model to a file that will be used while deployment
with open('model_Logistic_regression_algerian_ff_imbalanced.sav','wb') as f:
    pickle.dump(logistic_reg1,f)
```

11.0 Comparing Logistic Regression Model for Original and Imbalanced Dataset

11.1 Performance of Logistic Regression Model for Original Dataset

In [166…
```
print("The Performance of Model for Original dataset: \n{}".format(classification_report(y_test, logistic_reg_pred)))
```

```
The Performance of Model for Original dataset:
              precision    recall  f1-score   support

           0       0.97      0.94      0.96        36
           1       0.95      0.97      0.96        38

    accuracy                           0.96        74
   macro avg       0.96      0.96      0.96        74
weighted avg       0.96      0.96      0.96        74
```

11.2 Performance of Logistic Regression Model for Imbalanced Dataset

In [165…
```
print("The Performance of Model for Imbalanced dataset: \n{}".format(classification_report(y_test1, logistic_reg_pred1)))
```

```
The Performance of Model for Imbalanced dataset:
              precision    recall  f1-score   support

           0       0.64      0.70      0.67        64
           1       0.65      0.58      0.61        60

    accuracy                           0.65       124
   macro avg       0.65      0.64      0.64       124
weighted avg       0.65      0.65      0.64       124
```

Observations

1. It is clearly visible that creating imbalance on purpose degrades the dataset when we try to balance the imbalanced dataset.
2. This results in decrease in the performance of model.