Logistic Regression on Algerian Forest Fire Dataset

Submitted by : Nilutpal Das

Life Cycle of a Machine Learning Model

- **Understanding the Problem Statement**
- **Data Collection**
- **Exploratory Data Analysis**
- **Feature Engineering**
- **Model Training & Evaluation**

1. Problem Statement

- **Wildfires are a natural occurrence within some forest ecosystems, but in a few years, the fires are becoming more extreme and widespread. Hotter and drier weather caused by climate change and poor land management create conditions favorable for**

frequent, larger, and high-intensity forest fires. The incidence of forest fires is increasing year by year. Our Objective is to identify the prime factors of causing forest fires and this will help in minimizing the chances of forest fires and taking preventive measures against this problem.

2. Data Collection

- **In this dataset 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.**
- **This dataset comprises of 122 instances for each region. The period from June 2012 to September 2012. The dataset includes 11 attribues and 1 output attribue (class). The 244 instances have been classified into â€˜fireâ€™ (138 classes) and â€˜not fireâ€™ (106 classes) classes.**
- **Link for the dataset is as follows - https://archive.ics.uci.edu/ml/datasets/Algerian+Forest+Fires+Dataset++**

Dataset Attributes Information

- **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**

- **Fire Weather Index (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**

Importing required packages and libraries

In [214…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")

%matplotlib inline
```

Loading the dataset

In [2]:
```
df= pd.read_csv('D:\Data Science\Algerian dataset\Algerian_forest_fires_dataset.csv',
```

Showing top 5 records

In [3]:
```
df.head()
```

Out[3]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire

Showing top 5 records

In [4]:
```
df.tail()
```

Out[4]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
241	26	09	2012	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire
242	27	09	2012	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire
243	28	09	2012	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire
244	29	09	2012	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire
245	30	09	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire

Checking total rows and columns

In [5]:
```
df.shape
```

Out[5]:
```
(246, 14)
```

Observation

- **246 rows and 14 columns**

3. Exploratory Data Analysis

Checking all unique values of column 'day'

In [6]:
```
df['day'].unique()
```

Out[6]:
```
array(['01', '02', '03', '04', '05', '06', '07', '08', '09', '10', '11',
       '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '22',
       '23', '24', '25', '26', '27', '28', '29', '30', '31',
       'Sidi-Bel Abbes Region Dataset', 'day'], dtype=object)
```

Checking total rows of column 'day'

In [7]:
```
df['day'].shape
```

Out[7]:
```
(246,)
```

Checking all data is numeric or not

In [8]:
```
df['day'].str.isnumeric().sum()
```

Out[8]:
```
244
```

Showing particular row having the categorical value

In [9]:
```
df[~df['day'].str.isnumeric()]
```

Out[9]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	C
122	Sidi-Bel Abbes Region Dataset	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	C

Observation

- **row 122 and 123 are having categorical data**

Removing unwanted rows from the dataset

In [10]:
```
df.drop(index=[122,123], inplace=True)
df.reset_index(inplace=True)
df.drop('index', axis=1, inplace=True)
df.loc[122:].head(4)
```

Out[10]:

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

Observation

- **column 'day' is fixed by removing unwanted records from some rows**

Checking all unique values of column 'month'

In [11]:
```
df['month'].unique()
```

Out[11]:
```
array(['06', '07', '08', '09'], dtype=object)
```

Checking total rows of column 'month'

In [12]:
```
df['month'].shape
```

Out[12]:
```
(244,)
```

Checking all data is numeric or not

In [13]:
```
df['month'].str.isnumeric().sum()
```

Out[13]:
```
244
```

Shows particular row having categorical data

In [14]:
```
df[~df['month'].str.isnumeric()]
```

Out[14]:

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

Creating a new feature name 'Region' in the dataset

In [15]:
```
# creating feature called Region 0 for Bejaia region and 1 for Sidi Bel-abbes region
for i in range(len(df)):
    if i<=121:
        df['Region'] = '0'
    else:
        df['Region'][i] = '1'

df.loc[120:]
```

Out[15]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
120	29	09	2012	26	80	16	1.8	47.4	2.9	7.7	0.3	3	0.1	not fire
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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
239	26	09	2012	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire
240	27	09	2012	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire
241	28	09	2012	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire
242	29	09	2012	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire
243	30	09	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire

124 rows × 15 columns

Observation

- **this solves the anomaly found in column 'day'**

In [16]:
```
df['year'].unique()
```

Out[16]:
```
array(['2012'], dtype=object)
```

3.3 Renaming the names of the columns

In [17]:
```
df.columns= [col_name.strip() for col_name in df.columns]
df.columns
```

Out[17]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Ws', 'Rain', 'FFMC',
       'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'Classes', 'Region'],
      dtype='object')
```

converting all feature values to string for data cleaning

In [18]:
```
df=df.astype(str)
```

In [19]:
```
df.columns
```

Out[19]:
```
Index(['day', 'month', 'year', 'Temperature', 'RH', 'Ws', 'Rain', 'FFMC',
       'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'Classes', 'Region'],
      dtype='object')
```

Observation

- **spaces in features solved**

3.4 Stripping the classes features

In [20]:
```
df.Classes = df.Classes.str.strip()
df['Classes'].unique()
```

Out[20]:
```
array(['not fire', 'fire', 'nan'], dtype=object)
```

In [21]:
```
df.head()
```

Out[21]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Reg
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire	
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire	
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire	

In [22]:
```
df['Temperature'].unique()
```

Out[22]:
```
array(['29', '26', '25', '27', '31', '33', '30', '28', '32', '34', '35',
       '36', '37', '22', '24', '38', '39', '40', '42'], dtype=object)
```

In [23]:
```
df['Temperature'].shape
```

Out[23]:
```
(244,)
```

In [24]:
```
df['Temperature'].str.isnumeric().sum()
```

Out[24]:
```
244
```

In [25]:
```
df['RH'].unique()
```

Out[25]:
```
array(['57', '61', '82', '89', '77', '67', '54', '73', '88', '79', '65',
       '81', '84', '78', '80', '55', '62', '66', '64', '53', '47', '50',
       '68', '75', '76', '63', '69', '70', '59', '48', '45', '60', '51',
       '52', '58', '86', '74', '71', '49', '44', '41', '42', '90', '87',
       '72', '46', '37', '36', '56', '43', '83', '29', '34', '33', '35',
       '39', '31', '21', '40', '24', '38', '26'], dtype=object)
```

In [26]:
```
df['RH'].shape
```

Out[26]:
```
(244,)
```

In [27]:
```
df['RH'].str.isnumeric().sum()
```

Out[27]:
```
244
```

In [28]:
```
df['Ws'].unique()
```

Out[28]:
```
array(['18', '13', '22', '16', '14', '15', '12', '19', '21', '20', '17',
       '26', '11', '10', '9', '8', '6', '29'], dtype=object)
```

In [29]:
```
df['Ws'].str.isnumeric().sum()
```

Out[29]:
```
244
```

In [30]:
```
df['Rain'].unique()
```

Out[30]:
```
array(['0', '1.3', '13.1', '2.5', '0.2', '1.2', '0.5', '3.1', '0.7',
       '0.6', '0.3', '0.1', '0.4', '1', '1.4', '0.8', '16.8', '7.2',
       '10.1', '3.8', '0.9', '1.8', '4.6', '8.3', '5.8', '4', '2', '4.7',
       '8.7', '4.5', '1.1', '1.7', '2.2', '6', '1.9', '2.9', '4.1', '6.5',
       '4.4'], dtype=object)
```

In [31]:
```
df['FFMC'].unique()
```

Out[31]:
```
array(['65.7', '64.4', '47.1', '28.6', '64.8', '82.6', '88.2', '86.6',
       '52.9', '73.2', '84.5', '84', '50', '59', '49.4', '36.1', '37.3',
       '56.9', '79.9', '59.8', '81', '79.1', '81.4', '85.9', '86.7',
       '86.8', '89', '89.1', '88.7', '59.9', '55.7', '63.1', '80.1', '87',
       '80', '85.6', '66.6', '81.1', '75.1', '81.8', '73.9', '60.7',
       '72.6', '82.8', '85.4', '88.1', '73.4', '68.2', '70', '84.3',
       '89.2', '90.3', '86.5', '87.2', '78.8', '78', '76.6', '85', '86.4',
       '77.1', '87.4', '88.9', '81.3', '82.4', '80.2', '89.3', '89.4',
       '88.3', '88.6', '89.5', '85.8', '84.9', '90.1', '72.7', '52.5',
       '46', '30.5', '42.6', '68.4', '80.8', '75.8', '69.6', '62', '56.1',
       '58.5', '71', '40.9', '47.4', '44.9', '78.1', '87.7', '83.8',
       '87.8', '77.8', '73.7', '68.3', '48.6', '82', '85.7', '77.5', '45',
       '57.1', '48.7', '79.4', '83.7', '71.4', '90.6', '72.3', '53.4',
       '66.8', '62.2', '65.5', '64.6', '60.2', '86.2', '78.3', '74.2',
       '85.3', '86', '92.5', '79.7', '63.7', '87.6', '84.7', '88', '90.5',
       '82.3', '74.8', '85.2', '84.6', '86.1', '89.9', '93.9', '91.5',
       '87.3', '72.8', '73.8', '87.5', '93.3', '93.7', '93.8', '70.5',
       '69.7', '91.7', '94.2', '93', '91.9', '83.9', '92', '96', '94.3',
       '82.7', '91.2', '92.1', '92.2', '91', '79.2', '37.9', '75.4',
       '82.2', '73.5', '66.1', '64.5', '83.3', '82.5', '83.1', '59.5',
       '84.2', '79.5', '61.3', '41.1', '45.9', '67.3'], dtype=object)
```

In [33]:
```
df['FFMC'].shape
```

Out[33]:
```
(244,)
```

In [34]:
```
df['FFMC'].str.isnumeric().sum()
```

Out[34]:
```
27
```

In [35]:
```
df[~df['FFMC'].str.isnumeric()]
```

Out[35]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
239	26	09	2012	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire
240	27	09	2012	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire
241	28	09	2012	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire
242	29	09	2012	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire
243	30	09	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire

217 rows × 15 columns

Observation

- here 215 remaining data are from FFMC are in float and 27 are in int

In [36]:
```
df['DMC'].unique()
```

Out[36]:
```
array(['3.4', '4.1', '2.5', '1.3', '3', '5.8', '9.9', '12.1', '7.9',
       '9.5', '12.5', '13.8', '6.7', '4.6', '1.7', '1.1', '1.9', '4.5',
       '6.3', '7', '8.2', '11.2', '14.2', '17.8', '21.6', '25.5', '18.4',
       '22.9', '2.4', '2.6', '7.6', '10.9', '9.7', '7.7', '6', '8.1',
       '7.8', '5.2', '9.4', '12', '12.3', '18.5', '16.4', '10.5', '9.6',
       '17.1', '22.2', '24.4', '26.7', '28.5', '31.9', '4.8', '5.7',
       '11.1', '13', '15.5', '11.3', '14.8', '18.6', '21.7', '15.6', '19',
       '11.7', '16', '20', '23.2', '25.9', '29.6', '33.5', '37.6', '40.5',
       '43.9', '45.6', '47', '50.2', '54.2', '25.2', '8.7', '0.7', '1.2',
       '3.6', '3.2', '2.1', '2.2', '0.9', '6.4', '9.8', '13.5', '16.5',
       '10.6', '5.5', '8.3', '7.1', '2.9', '2.7', '8.4', '8.5', '13.3',
       '18.2', '21.3', '11.4', '7.2', '4.2', '3.9', '4.4', '3.8', '10',
       '12.8', '20.9', '27.2', '17.9', '13.6', '18.7', '8', '12.6',
       '12.9', '18', '19.4', '21.1', '23.9', '27.8', '32.7', '39.6',
       '44.2', '46.6', '10.8', '11.8', '15.7', '19.5', '23.8', '28.3',
       '23', '23.6', '11', '15.8', '22.5', '16.9', '22.3', '22.6', '30.3',
       '35.9', '34.4', '36.9', '41.1', '46.1', '51.3', '56.3', '61.3',
       '65.9', '37', '20.7', '24.8', '4', '3.3', '6.6', '4.7', '6.5',
       '11.5', '21.2', '25.8', '24.9', '26.1', '29.4', '11.9', '3.5',
       '4.3'], dtype=object)
```

In [37]:
```
# df['DMC'].tolist()
```

In [38]:
```
df['DMC'].shape
```

Out[38]:
```
(244,)
```

In [39]:
```
df['DMC'].str.isnumeric().sum()
```

Out[39]:
```
26
```

Observation

- **Remaining data are in float and 26 data are in int.**

In [40]:
```
df['DC'].unique()
```

Out[40]:
```
array(['7.6', '7.1', '6.9', '14.2', '22.2', '30.5', '38.3', '38.8',
       '46.3', '54.3', '61.4', '17', '7.8', '7.4', '8', '16', '27.1',
       '31.6', '39.5', '47.7', '55.8', '63.8', '71.8', '80.3', '88.5',
       '84.4', '92.8', '8.6', '8.3', '9.2', '18.5', '27.9', '37', '40.4',
       '49.8', '9.3', '18.7', '27.7', '37.2', '22.9', '25.5', '34.1',
       '43.1', '52.8', '62.1', '71.5', '79.9', '71.3', '79.7', '88.7',
       '98.6', '108.5', '117.8', '127', '136', '145.7', '10.2', '10',
       '19.8', '29.7', '39.1', '48.6', '47', '57', '67', '77', '75.1',
       '85.1', '94.7', '92.5', '90.4', '100.7', '110.9', '120.9', '130.6',
       '141.1', '151.3', '161.5', '171.3', '181.3', '190.6', '200.2',
       '210.4', '220.4', '180.4', '8.7', '7.5', '7', '15.7', '24', '32.2',
       '30.1', '8.4', '8.9', '16.6', '7.3', '24.3', '33.1', '41.3',
       '49.3', '57.9', '41.4', '30.4', '15.2', '7.7', '16.3', '24.9',
       '8.8', '8.2', '15.4', '17.6', '26.3', '28.9', '14.7', '22.5',
       '37.8', '18.4', '25.6', '34.5', '43.3', '52.4', '36.7', '8.5',
       '17.8', '27.3', '36.8', '46.4', '45.1', '35.4', '9.7', '9.9',
       '9.5', '19.4', '10.4', '14.6 9', '24.1', '42.3', '51.6', '61.1',
       '71', '80.6', '90.1', '99', '56.6', '15.9', '19.7', '28.3', '37.6',
       '47.2', '57.1', '67.2', '10.5', '21.4', '32.1', '42.7', '52.5',
       '9.1', '9.8', '20.2', '30.9', '41.5', '55.5', '54.2', '65.1',
       '76.4', '86.8', '96.8', '107', '117.1', '127.5', '137.7', '147.7',
       '157.5', '167.2', '177.3', '166', '149.2', '159.1', '168.2',
       '26.6', '17.7', '26.1', '25.2', '33.4', '50.2', '59.2', '63.3',
       '77.8', '86', '88', '97.3', '106.3', '115.6', '28.1', '36.1',
       '44.5', '7.9', '16.5'], dtype=object)
```

In [42]:
```
df['DC'].shape
```

Out[42]:
```
(244,)
```

In [43]:
```
df['DC'].str.isnumeric().sum()
```

Out[43]:
```
27
```

In [44]:
```
df[~df['FFMC'].str.isnumeric()]
```

Out[44]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
239	26	09	2012	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire
240	27	09	2012	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire
241	28	09	2012	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire
242	29	09	2012	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire
243	30	09	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire

217 rows × 15 columns

Observation

- **Remaining data are in float and 27 are in int.**

In [45]:
```
df['ISI'].unique()
```

Out[45]:
```
array(['1.3', '1', '0.3', '0', '1.2', '3.1', '6.4', '5.6', '0.4', '4',
       '4.8', '0.5', '0.7', '2.5', '0.9', '2.6', '2.4', '3.3', '5.7',
       '6.7', '9.2', '7.6', '2.2', '7.2', '1.1', '0.8', '2.7', '2.8', '6',
       '1.5', '3', '1.4', '3.2', '4.6', '7.7', '5.2', '1.8', '10', '8.7',
       '4.7', '6.8', '2', '1.7', '5.5', '6.9', '7.4', '7.1', '5.9', '3.7',
       '9.7', '8.8', '9.9', '10.4', '9', '8.2', '4.4', '7.3', '12.5',
       '0.6', '0.2', '0.1', '2.1', '1.9', '6.2', '7.8', '4.5', '5.4',
       '8.4', '13.4', '5', '1.6', '4.9', '7', '8', '11.7', '11.3', '4.3',
       '4.1', '8.3', '4.2', '10.9', '9.5', '18.5', '13.2', '13.8', '17.2',
       '15.7', '19', '9.6', '16.6', '15.5', '7.5', '10.8', '3.5', '16',
       '3.8', '5.1', '11.5', '12.2', '14.3', '13.1', '8.1', '9.8', '9.1',
       '14.2', '11.2'], dtype=object)
```

In [46]:
```
df['ISI'].shape
```

Out[46]:
```
(244,)
```

In [47]:
```
df['ISI'].str.isnumeric().sum()
```

Out[47]:
```
30
```

Observation

- **Remaining are in float and 30 are in int.**

In [48]:
```
df['BUI'].unique()
```

Out[48]:
```
array(['3.4', '3.9', '2.7', '1.7', '7', '10.9', '13.5', '10.5', '12.6',
       '15.8', '17.7', '6.7', '4.4', '3', '2.2', '1.6', '2.4', '5.3',
       '5.1', '8.4', '9.7', '11.5', '14.9', '18.3', '21.6', '25.8',
       '29.7', '23.8', '28.3', '2.9', '2.8', '5.7', '9.1', '12.5', '12.1',
       '15.4', '7.4', '5.8', '8.1', '9.2', '11.7', '5.9', '8.3', '11.1',
       '14.2', '18.2', '16.5', '22.4', '21.7', '14.7', '18.5', '23.9',
       '29.4', '32.1', '35', '37.4', '41.2', '4.7', '5.5', '8.2', '17.2',
       '14.1', '17.9', '21.9', '25.5', '20.7', '24.4', '27.2', '22',
       '17.6', '22.9', '27.5', '31.3', '34.7', '38.8', '43.1', '47.5',
       '50.9', '54.7', '57.1', '59.3', '62.9', '67.4', '1.8', '1.1',
       '5.6', '2.6', '3.7', '1.4', '4.2', '7.7', '11.3', '16', '19.2',
       '12.9', '9.6', '6.2', '9', '6.8', '6.5', '9.3', '10.7', '7.3',
       '13.1', '18', '21.2', '6.1', '7.1', '4.1', '3.8', '9.9', '12.7',
       '16.4', '20.8', '27.1', '17.8', '3.3', '7.8', '10.3', '18.7',
       '16.7', '13.7', '9.4', '10.4', '20.9', '27.7', '32.6', '39.5',
       '44', '46.5', '11.4', '11.8', '15.7', '19.5', '10.6', '16.9',
       '23.5', '6.9', '11', '18.4', '17.5', '22.3', '19', '24.2', '30.4',
       '35.9', '35.5', '38.1', '41.3', '45.5', '50.2', '54.9', '59.5',
       '64', '68', '30.6', '35.7', '39.3', '4', '6', '3.5', '6.4', '10',
       '4.6', '6.6', '12.4', '14.3', '26.2', '28.2', '28.9', '32.4', '36',
       '11.9', '4.8'], dtype=object)
```

In [49]:
```
df['BUI'].shape
```

Out[49]:
```
(244,)
```

In [50]:
```
df['BUI'].str.isnumeric().sum()
```

Out[50]:
```
22
```

Observation

- **Remaining are in float and 22 are in int.**

In [51]:
```
df['FWI'].unique()
```

Out[51]:
```
array(['0.5', '0.4', '0.1', '0', '2.5', '7.2', '7.1', '0.3', '0.9', '5.6',
       '7.1 ', '0.2', '1.4', '2.2', '2.3', '3.8', '7.5', '8.4', '10.6',
       '15', '13.9', '3.9', '12.9', '1.7', '4.9', '6.8', '3.2', '8',
       '0.6', '3.4', '0.8', '3.6', '6', '10.9', '4', '8.8', '2.8', '2.1',
       '1.3', '7.3', '15.3', '11.3', '11.9', '10.7', '15.7', '6.1', '2.6',
       '9.9', '11.6', '12.1', '4.2', '10.2', '6.3', '14.6', '16.1',
       '17.2', '16.8', '18.4', '20.4', '22.3', '20.9', '20.3', '13.7',
       '13.2', '19.9', '30.2', '5.9', '7.7', '9.7', '8.3', '0.7', '4.1',
       '1', '3.1', '1.9', '10', '16.7', '1.2', '5.3', '6.7', '9.5', '12',
       '6.4', '5.2', '3', '9.6', '4.7', 'fire   ', '14.1', '9.1', '13',
       '17.3', '30', '25.4', '16.3', '9', '14.5', '13.5', '19.5', '12.6',
       '12.7', '21.6', '18.8', '10.5', '5.5', '14.8', '24', '26.3',
       '12.2', '18.1', '24.5', '26.9', '31.1', '30.3', '26.1', '16',
       '19.4', '2.7', '3.7', '10.3', '5.7', '9.8', '19.3', '17.5', '15.4',
       '15.2', '6.5'], dtype=object)
```

In [53]:
```
df['FWI'].shape
```

Out[53]:
```
(244,)
```

In [54]:
```
df['FWI'].str.isnumeric().sum()
```

Out[54]:
```
28
```

Observation

- **Remaining are in float and 28 are in int.**

In [55]:
```
df['Classes'].unique()
```

Out[55]:
```
array(['not fire', 'fire', 'nan'], dtype=object)
```

In [57]:
```
df['Classes'].shape
```

Out[57]:
```
(244,)
```

In [58]:
```
df['Classes'].str.isnumeric().sum()
```

Out[58]:
```
0
```

Observation

- **All are string.**

In [59]:
```
df['Region'].unique()
```

Out[59]:
```
array(['0', '1'], dtype=object)
```

In [60]:
```
df['Region'].shape
```

Out[60]:
```
(244,)
```

In [61]:
```
df['Region'].str.isnumeric().sum()
```

Out[61]:
```
244
```

4. Feature Engineering

Checking datatypes and Null values

In [62]:
```
df.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      244 non-null    object
 14  Region       244 non-null    object
dtypes: object(15)
memory usage: 28.7+ KB
```

Checking null values

In [63]:
```
df.isnull().sum()
```

Out[63]:
```
day            0
month          0
year           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
```

Fixing the null values

In [64]:
```
df.iloc[165]
```

Out[64]:
```
day                14
month              07
year             2012
Temperature        37
RH                 37
Ws                 18
Rain              0.2
FFMC             88.9
DMC              12.9
DC             14.6 9
ISI              12.5
BUI              10.4
FWI             fire
Classes              nan
Region                 1
Name: 165, dtype: object
```

In [65]:
```
df.at[165, 'DC'] = 14.6
df.at[165, 'ISI'] = 9
df.at[165, 'BUI'] = 12.5
df.at[165, 'FWI'] = 10.4
df.at[165, 'Classes'] = 'fire'
```

In [66]:
```
df.isnull().sum()
```

Out[66]:
```
day            0
month          0
year           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
```

Handling categorical values

In [67]:
```
df['Classes']=df['Classes'].map({'not fire':0,'fire':1})
df.head()
```

Out[67]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Reg
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	0	
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	0	
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0	
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	0	
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	0	

Changing the datatypes

In [68]:
```
df['day']=df['day'].astype(int)
df['month']=df['month'].astype(int)
df['year']=df['year'].astype(int)
df['Temperature']=df['Temperature'].astype(int)
df['RH']=df['RH'].astype(int)
df['Rain']=df['Rain'].astype(float)
df['FFMC']=df['FFMC'].astype(float)
df['DMC']=df['DMC'].astype(float)
df['DC']=df['DC'].astype(float)
df['FWI']=df['FWI'].astype(float)
df['BUI']=df['BUI'].astype(float)
df['ISI']=df['ISI'].astype(float)
df['Ws']=df['Ws'].astype(float)
df['Classes']=df['Classes'].astype(int)
df['Region']=df['Region'].astype(int)
```

Basic Information

In [69]:
```
df.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    int32  
 1   month        244 non-null    int32  
 2   year         244 non-null    int32  
 3   Temperature  244 non-null    int32  
 4   RH           244 non-null    int32  
 5   Ws           244 non-null    float64
 6   Rain         244 non-null    float64
 7   FFMC         244 non-null    float64
 8   DMC          244 non-null    float64
 9   DC           244 non-null    float64
 10  ISI          244 non-null    float64
 11  BUI          244 non-null    float64
 12  FWI          244 non-null    float64
 13  Classes      244 non-null    int32  
 14  Region       244 non-null    int32  
dtypes: float64(8), int32(7)
memory usage: 22.0 KB
```

Observation

- **Corrected datatypes.**
- **No null values.**

Missing values

In [70]:
```
df.isnull().sum()
```

Out[70]:
```
day            0
month          0
year           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
```

Observation

- **No null values**

Duplicate values

In [71]:
```
df[df.duplicated()]
```

Out[71]:

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

Observation

- **No duplicated values**

Creating copy dataframe from original dataframe

In [72]:
```
df_copy=df.copy()
```

In [73]:
```
df_copy
```

Out[73]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
0	1	6	2012	29	57	18.0	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0
1	2	6	2012	29	61	13.0	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0
2	3	6	2012	26	82	22.0	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0
3	4	6	2012	25	89	13.0	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0
4	5	6	2012	27	77	16.0	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
239	26	9	2012	30	65	14.0	0.0	85.4	16.0	44.5	4.5	16.9	6.5	1
240	27	9	2012	28	87	15.0	4.4	41.1	6.5	8.0	0.1	6.2	0.0	0
241	28	9	2012	27	87	29.0	0.5	45.9	3.5	7.9	0.4	3.4	0.2	0
242	29	9	2012	24	54	18.0	0.1	79.7	4.3	15.2	1.7	5.1	0.7	0
243	30	9	2012	24	64	15.0	0.2	67.3	3.8	16.5	1.2	4.8	0.5	0

244 rows × 15 columns

Dropping column 'year' as the dataset records are from 2012 only.

In [74]:
```
df_copy.drop('year', axis=1, inplace=True)
```

Statistical analysis

In [75]:
```
df_copy.describe().T
```

Out[75]:

	count	mean	std	min	25%	50%	75%	max
day	244.0	15.754098	8.825059	1.0	8.000	16.00	23.000	31.0
month	244.0	7.500000	1.112961	6.0	7.000	7.50	8.000	9.0
Temperature	244.0	32.172131	3.633843	22.0	30.000	32.00	35.000	42.0
RH	244.0	61.938525	14.884200	21.0	52.000	63.00	73.250	90.0
Ws	244.0	15.504098	2.810178	6.0	14.000	15.00	17.000	29.0
Rain	244.0	0.760656	1.999406	0.0	0.000	0.00	0.500	16.8
FFMC	244.0	77.887705	14.337571	28.6	72.075	83.50	88.300	96.0
DMC	244.0	14.673361	12.368039	0.7	5.800	11.30	20.750	65.9
DC	244.0	49.288115	47.619662	6.9	13.275	33.10	68.150	220.4
ISI	244.0	4.759836	4.154628	0.0	1.400	3.50	7.300	19.0
BUI	244.0	16.673361	14.201648	1.1	6.000	12.45	22.525	68.0
FWI	244.0	7.049180	7.428366	0.0	0.700	4.45	11.375	31.1
Classes	244.0	0.565574	0.496700	0.0	0.000	1.00	1.000	1.0
Region	244.0	0.500000	0.501028	0.0	0.000	0.50	1.000	1.0

Numerical features

In [76]:
```
num = [feature for feature in df_copy.columns if df_copy[feature].dtype != 'O']
num
```

Out[76]:
```
['day',
 'month',
 'Temperature',
 'RH',
 'Ws',
 'Rain',
 'FFMC',
 'DMC',
 'DC',
 'ISI',
 'BUI',
 'FWI',
 'Classes',
 'Region']
```

Creating dataframe of numerical features

In [215…
```
num_df=df_copy[num]
num_df
```

Out[215]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Regio
0	1	6	29	57	18.0	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0	
1	2	6	29	61	13.0	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0	
2	3	6	26	82	22.0	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0	
3	4	6	25	89	13.0	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0	
4	5	6	27	77	16.0	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0	
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
239	26	9	30	65	14.0	0.0	85.4	16.0	44.5	4.5	16.9	6.5	1	
240	27	9	28	87	15.0	4.4	41.1	6.5	8.0	0.1	6.2	0.0	0	
241	28	9	27	87	29.0	0.5	45.9	3.5	7.9	0.4	3.4	0.2	0	
242	29	9	24	54	18.0	0.1	79.7	4.3	15.2	1.7	5.1	0.7	0	
243	30	9	24	64	15.0	0.2	67.3	3.8	16.5	1.2	4.8	0.5	0	

244 rows × 14 columns

Counting the proportion of all features

In [78]:
```
df_copy['day'].value_counts().sort_values(ascending=False)
```

Out[78]:
```
1     8
2     8
3     8
4     8
5     8
6     8
7     8
8     8
9     8
10    8
11    8
12    8
13    8
14    8
15    8
16    8
17    8
18    8
19    8
20    8
21    8
22    8
23    8
24    8
25    8
26    8
27    8
28    8
29    8
30    8
31    4
Name: day, dtype: int64
```

In [79]:
```
df_copy['month'].value_counts().sort_values(ascending=False)
```

Out[79]:
```
7    62
8    62
6    60
9    60
Name: month, dtype: int64
```

In [80]:
```
df_copy['Temperature'].value_counts().sort_values(ascending=False)
```

Out[80]:
```
35    29
31    25
34    24
33    23
30    22
32    21
36    21
29    18
28    15
37     9
27     8
25     6
39     6
26     5
24     3
38     3
40     3
22     2
42     1
Name: Temperature, dtype: int64
```

In [81]:
```
df_copy['RH'].value_counts().sort_values(ascending=False)
```

Out[81]:
```
64    10
55    10
58     8
54     8
78     8
      ..
21     1
90     1
24     1
38     1
26     1
Name: RH, Length: 62, dtype: int64
```

In [82]:
```
df_copy['Ws'].value_counts()
```

Out[82]:
```
14.0    43
15.0    40
13.0    30
17.0    28
16.0    27
18.0    26
19.0    15
21.0     8
11.0     7
12.0     7
10.0     3
9.0      2
20.0     2
22.0     2
26.0     1
8.0      1
6.0      1
29.0     1
Name: Ws, dtype: int64
```

In [83]:
```
df_copy['Rain'].value_counts()
```

Out[83]:
```
0.0     133
0.1      18
0.2      12
0.3      10
0.4       8
0.7       6
0.6       6
0.5       5
1.1       3
1.2       3
2.0       3
1.8       3
0.8       2
2.9       2
1.3       2
3.8       2
1.4       2
1.0       2
3.1       2
16.8      1
4.5       1
6.5       1
4.1       1
13.1      1
1.9       1
6.0       1
2.2       1
1.7       1
2.5       1
4.7       1
8.7       1
7.2       1
4.0       1
5.8       1
8.3       1
4.6       1
0.9       1
10.1      1
4.4       1
Name: Rain, dtype: int64
```

In [84]:
```
df_copy['FFMC'].value_counts()
```

Out[84]:
```
88.9    8
89.4    5
89.3    4
85.4    4
89.1    4
       ..
82.4    1
81.3    1
86.4    1
76.6    1
67.3    1
Name: FFMC, Length: 173, dtype: int64
```

In [85]:
```
df_copy['DMC'].value_counts()
```

Out[85]:
```
7.9     5
12.5    4
1.9     4
3.4     3
4.6     3
       ..
10.9    1
2.1     1
3.6     1
1.2     1
4.3     1
Name: DMC, Length: 166, dtype: int64
```

In [86]:
```
df_copy['DC'].value_counts()
```

Out[86]:
```
8.0      5
7.6      4
7.8      4
8.4      4
7.5      4
        ..
90.4     1
100.7    1
110.9    1
120.9    1
16.5     1
Name: DC, Length: 198, dtype: int64
```

In [87]:
```
df_copy['ISI'].value_counts()
```

Out[87]:
```
1.1     8
1.2     7
5.2     5
1.5     5
2.8     5
       ..
4.9     1
7.0     1
11.7    1
11.3    1
11.2    1
Name: ISI, Length: 106, dtype: int64
```

In [88]:
```
df_copy['BUI'].value_counts()
```

Out[88]:
```
3.0     5
5.1     4
14.2    3
2.9     3
11.5    3
       ..
67.4    1
62.9    1
59.3    1
57.1    1
4.8     1
Name: BUI, Length: 173, dtype: int64
```

In [89]:
```
df_copy['FWI'].value_counts()
```

Out[89]:
```
0.4     12
0.8     10
0.5      9
0.1      9
0.0      9
        ..
30.2     1
20.3     1
22.3     1
20.4     1
6.5      1
Name: FWI, Length: 126, dtype: int64
```

In [90]:
```
df_copy['Classes'].value_counts().sort_values(ascending=False)
```

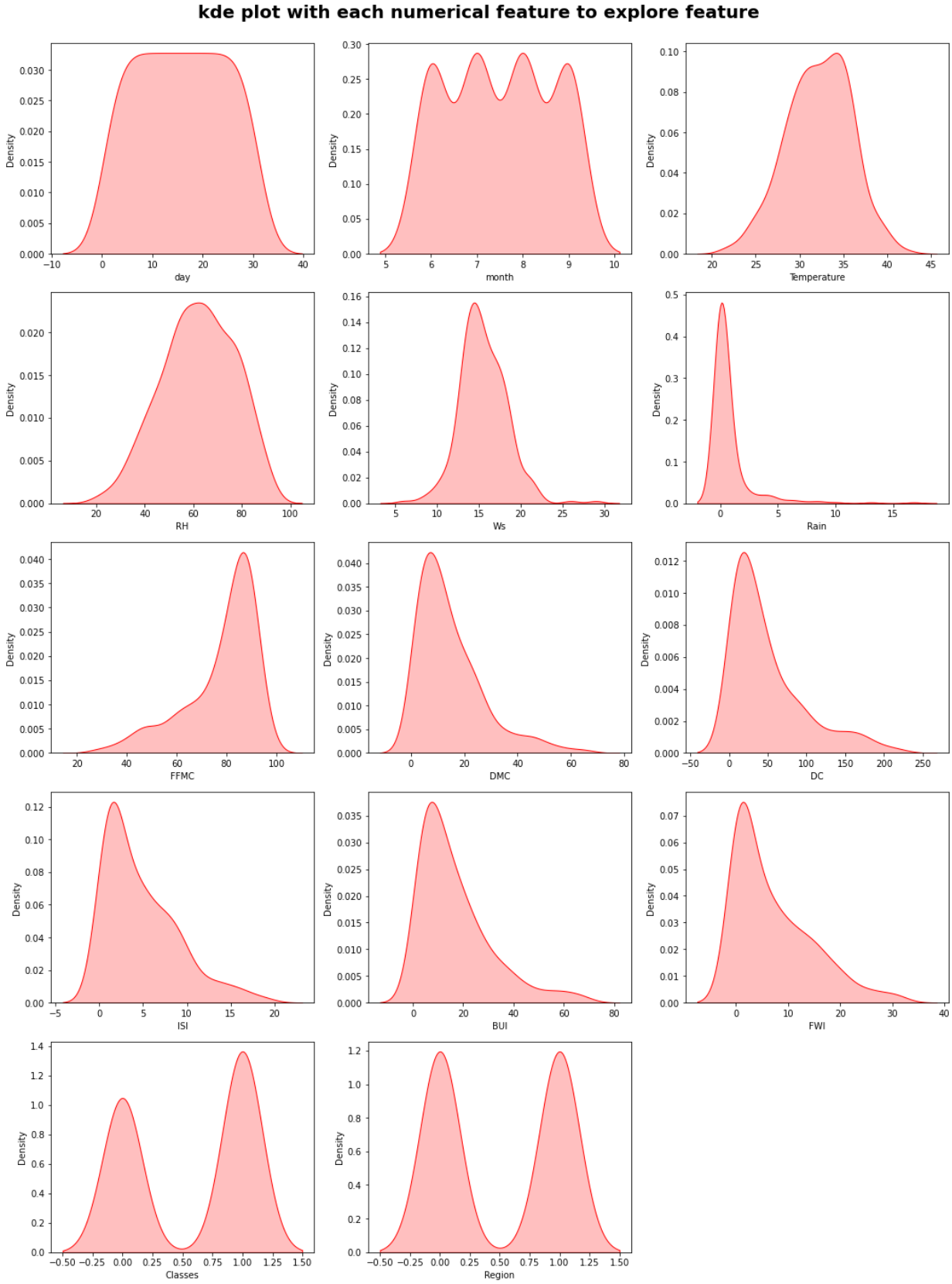
Out[90]:
```
1    138
0    106
Name: Classes, dtype: int64
```

In [91]:
```
df_copy['Region'].value_counts().sort_values(ascending=False)
```

Out[91]:
```
0    122
1    122
Name: Region, dtype: int64
```

Univariate Analysis

In [211…
```
plt.figure(figsize=(15, 20))
plt.suptitle('kde plot with each numerical feature to explore feature', fontsize=20, f

for i in range(0, len(num)):
    plt.subplot(5, 3, i+1)
    sns.kdeplot(x=df_copy[num[i]], data=df_copy, shade=True, color='r')
    plt.tight_layout()
```

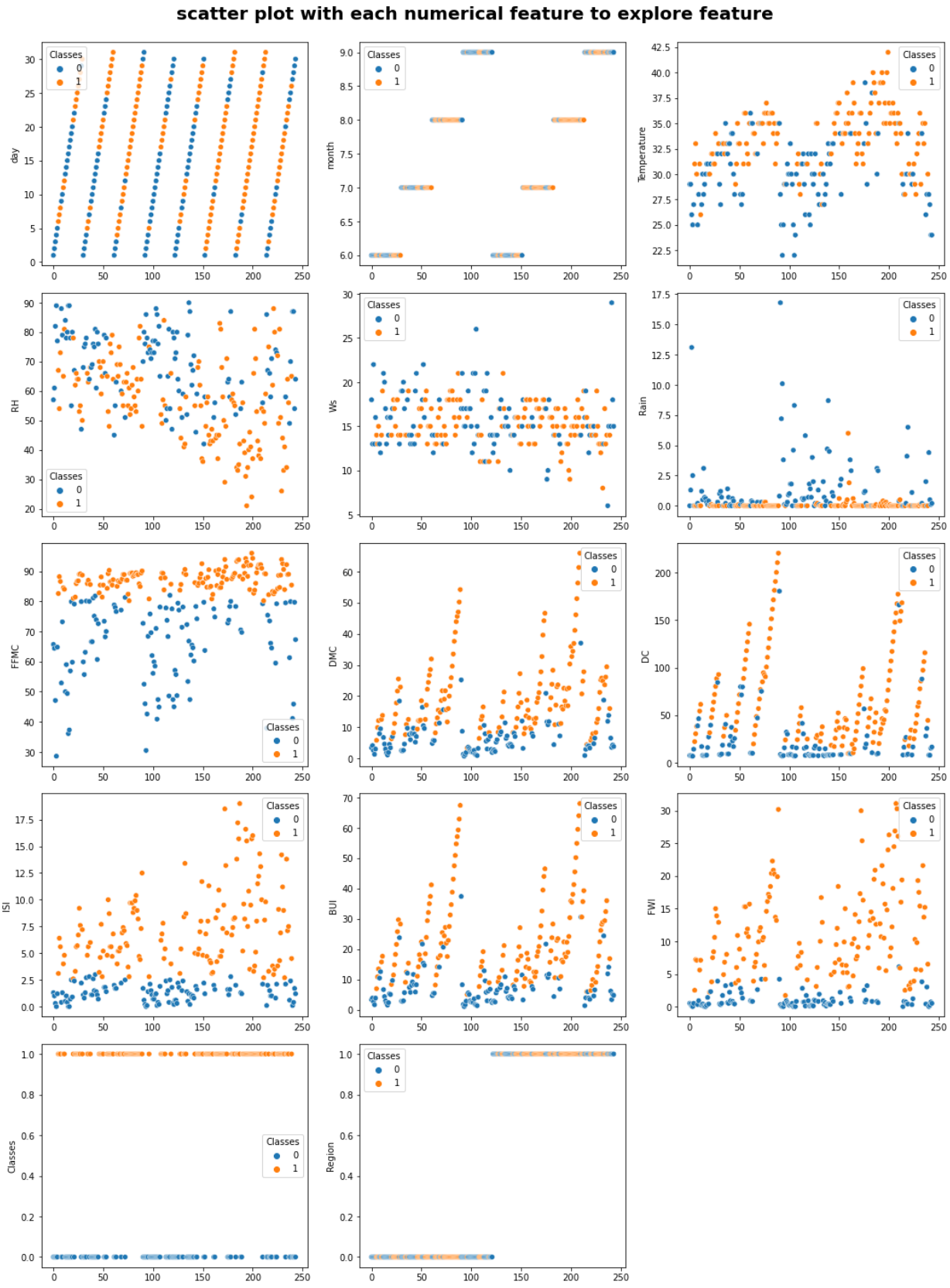


Observation

- **Rain, DMC, DC, ISI, BUI, FWI are log normal distribution.**

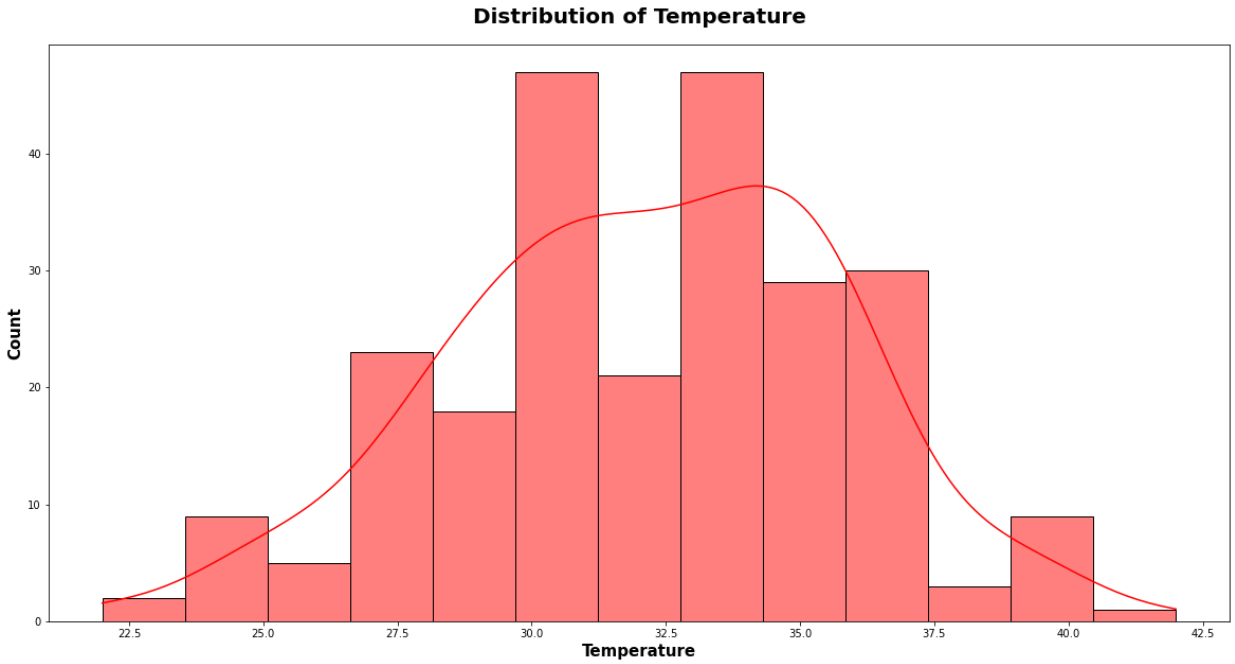
In [93]:
```
plt.figure(figsize=(15, 20))
plt.suptitle('scatter plot with each numerical feature to explore feature', fontsize=2

for i in range(0, len(num)):
    plt.subplot(5, 3, i+1)
```

```
    sns.scatterplot(y=num[i], x=df_copy.index, data=df_copy, hue='Classes')
    plt.tight_layout()
```



Relation between Temperature and Relative Humidity

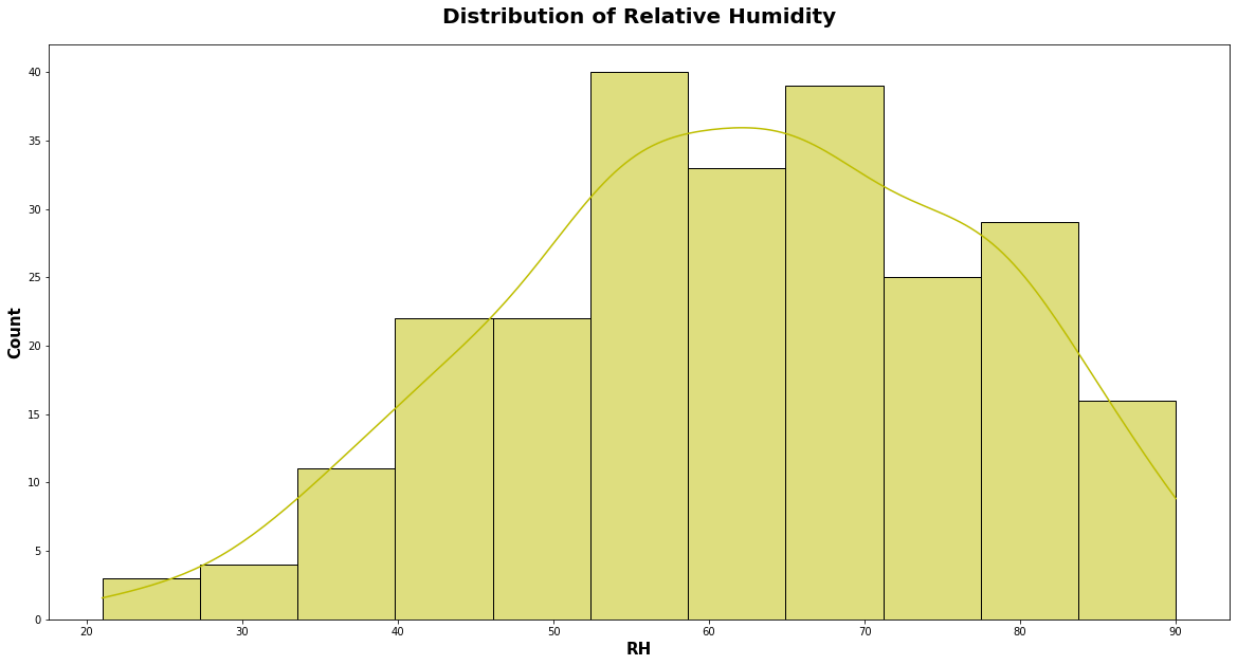
In [94]:
```
plt.subplots(figsize=(20,10))
sns.histplot("Distribution of Temperature",x=df_copy.Temperature,color='r',kde=True)
plt.title("Distribution of Temperature",weight='bold',fontsize=20,pad=20)
plt.xlabel("Temperature",weight='bold',fontsize=15)
plt.ylabel("Count",weight='bold',fontsize=15)
plt.show()
```



Observation

- **Temperature range is between 30.5 to 35.0 degree celcius.**

In [95]:
```
plt.subplots(figsize=(20,10))
sns.histplot("Distribution of Relative Humidity",x=df_copy.RH,color='y',kde=True)
plt.title("Distribution of Relative Humidity",weight='bold',fontsize=20,pad=20)
plt.xlabel("RH",weight='bold',fontsize=15)
plt.ylabel("Count",weight='bold',fontsize=15)
plt.show()
```



discomfort index = relation between temperature and relative humidity

In [96]:
```
discomfort_index=df_copy.groupby(['Temperature', 'RH']).size()
```

In [97]:
```
discomfort_index.head(5)
```

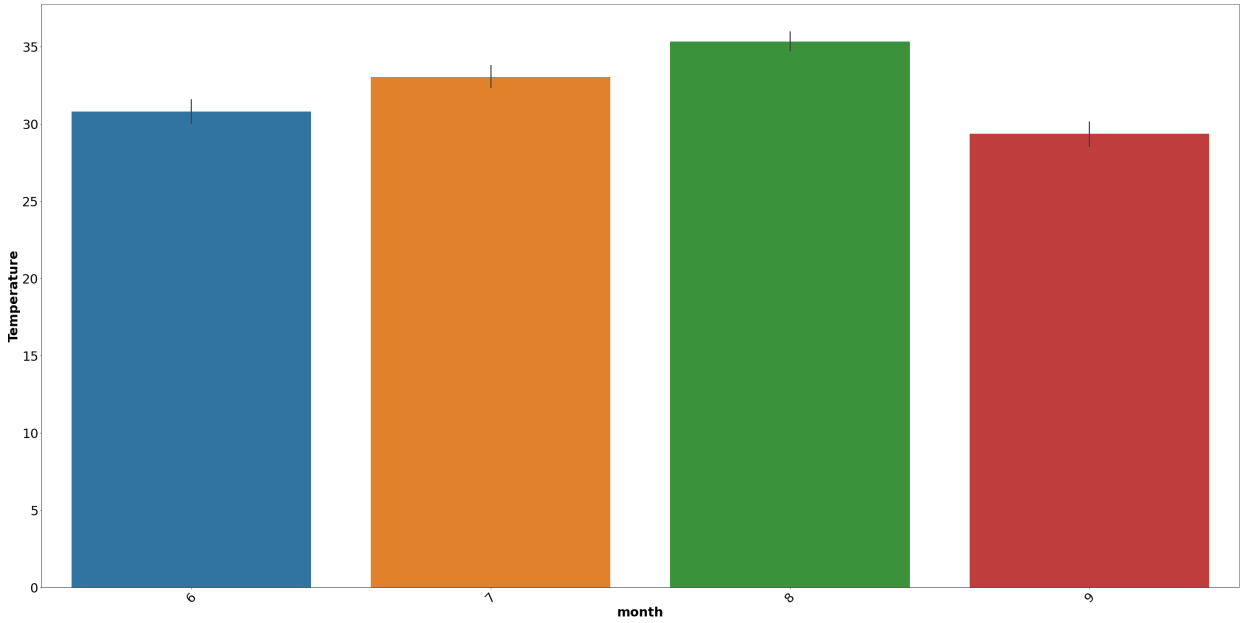
Out[97]:
```
Temperature  RH
22           76    1
             86    1
24           54    1
             64    1
             82    1
dtype: int64
```

Observation

- **this output also shows same that as the temperature goes down the relative humidity goes up.**

Relation between temperature and month

In [212…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='month', y='Temperature', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('month',weight='bold', fontsize=30);
plt.ylabel('Temperature',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

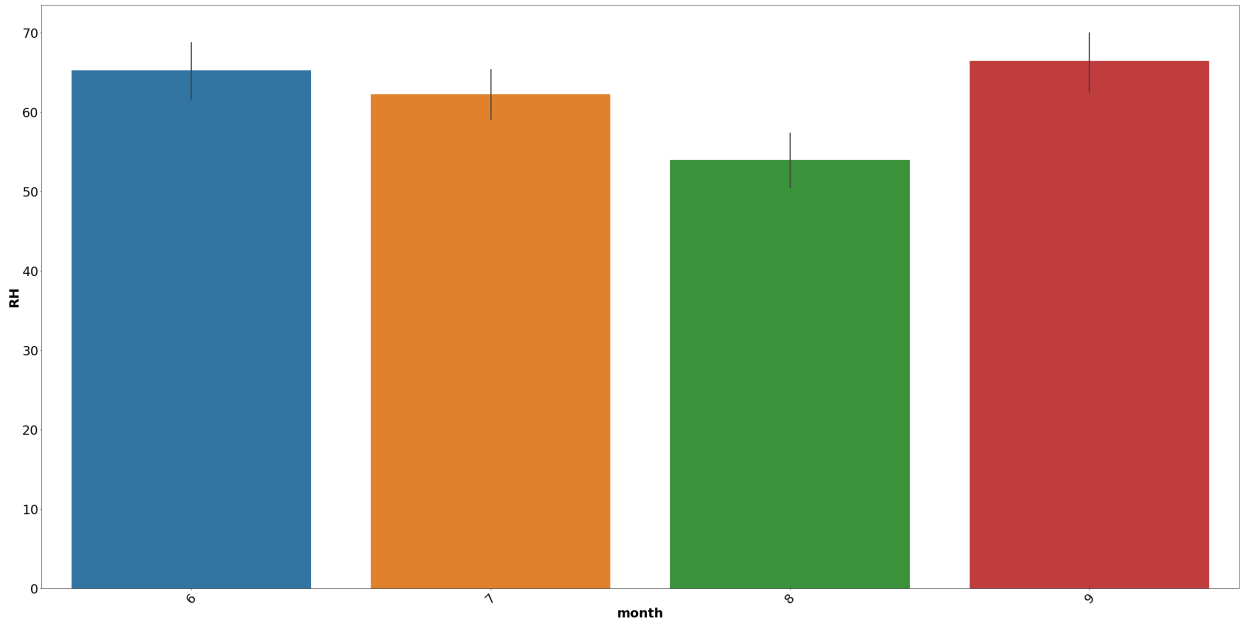


Observation

- **the month of july and August experiences highest temperature in the entire year.**

Relation between Relative Humidity (RH) and month

In [99]:
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='month', y='RH', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('month',weight='bold', fontsize=30);
plt.ylabel('RH',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```



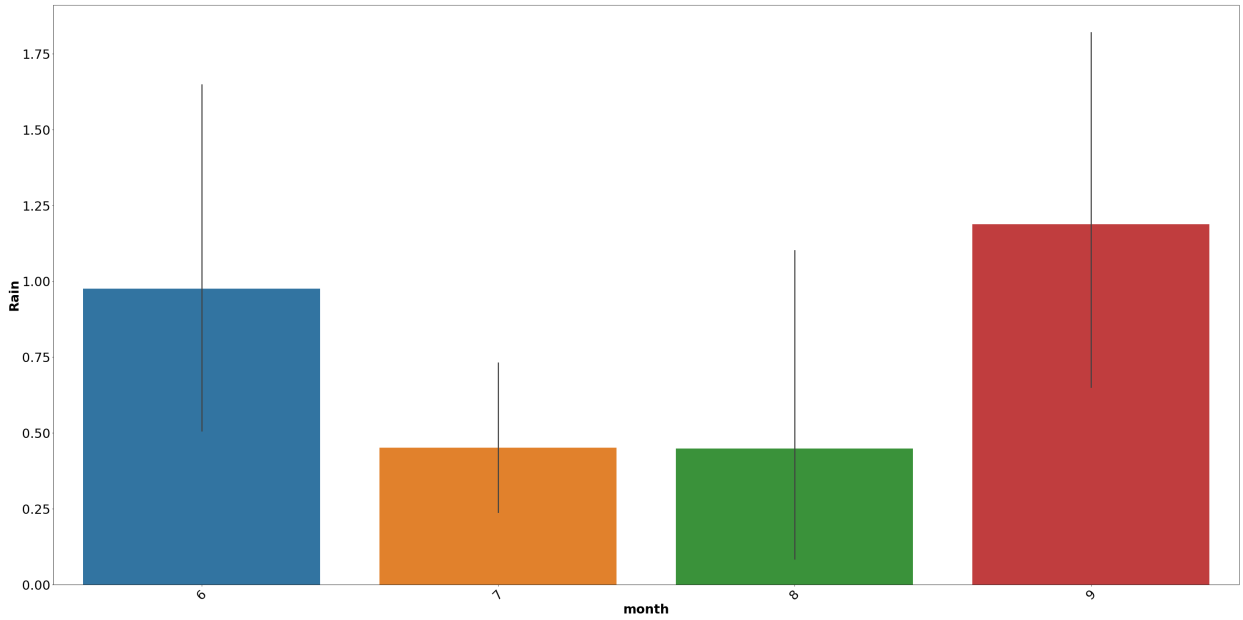
Observation

- **As per the temperature chart the lowest temperature experienced on 9th month i.e. september.**
- **So the Relative Humidity would be highest on 9th month i.e. september.**

Relation between Rain and month

In [100…

```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='month', y='Rain', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('month',weight='bold', fontsize=30);
plt.ylabel('Rain',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

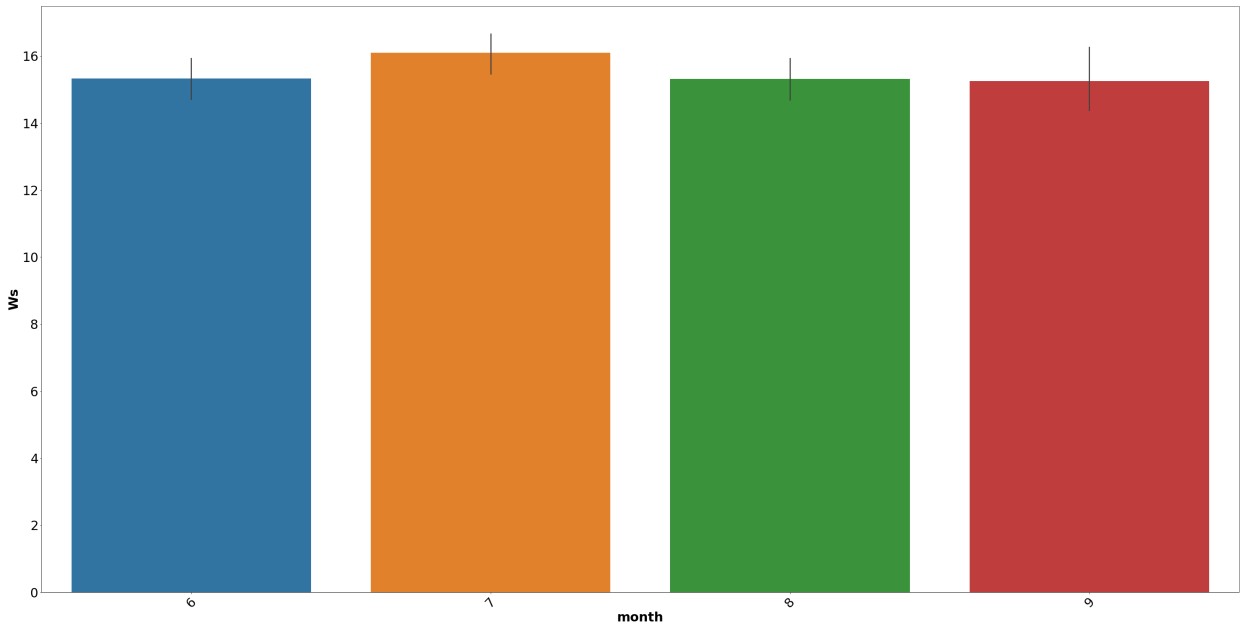


Observation

- **9th month i.e. september of the year 2012 experiences highest rainfall.**

Relation between Wind speed (Ws) and month

In [101…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='month', y='Ws', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('month',weight='bold', fontsize=30);
plt.ylabel('Ws',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

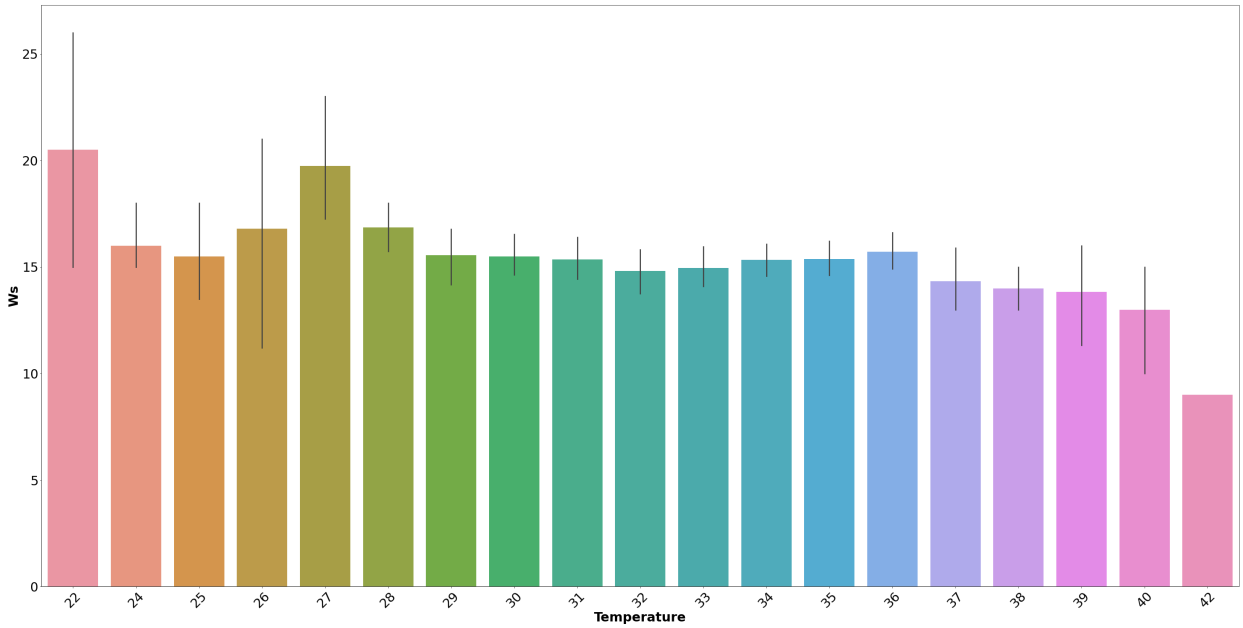


Observation

- **7th month i.e. july experiences of the year 2012 highest wind speed.**

Relation between temperature and Wind speed (Ws)

In [102…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Temperature', y='Ws', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Temperature',weight='bold', fontsize=30);
plt.ylabel('Ws',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```



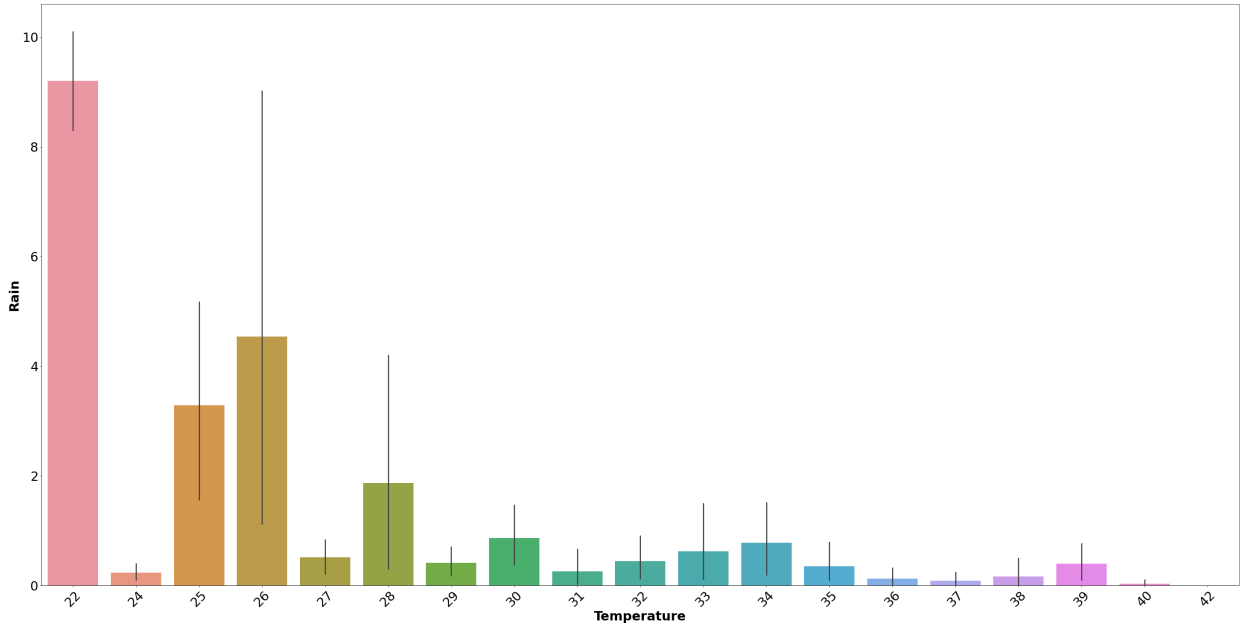
Observation

- **On temperature 22 degree celcius the Wind speed is highest.**

Relation between temperature and Rain

In [103…

```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Temperature', y='Rain', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Temperature',weight='bold', fontsize=30);
plt.ylabel('Rain',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

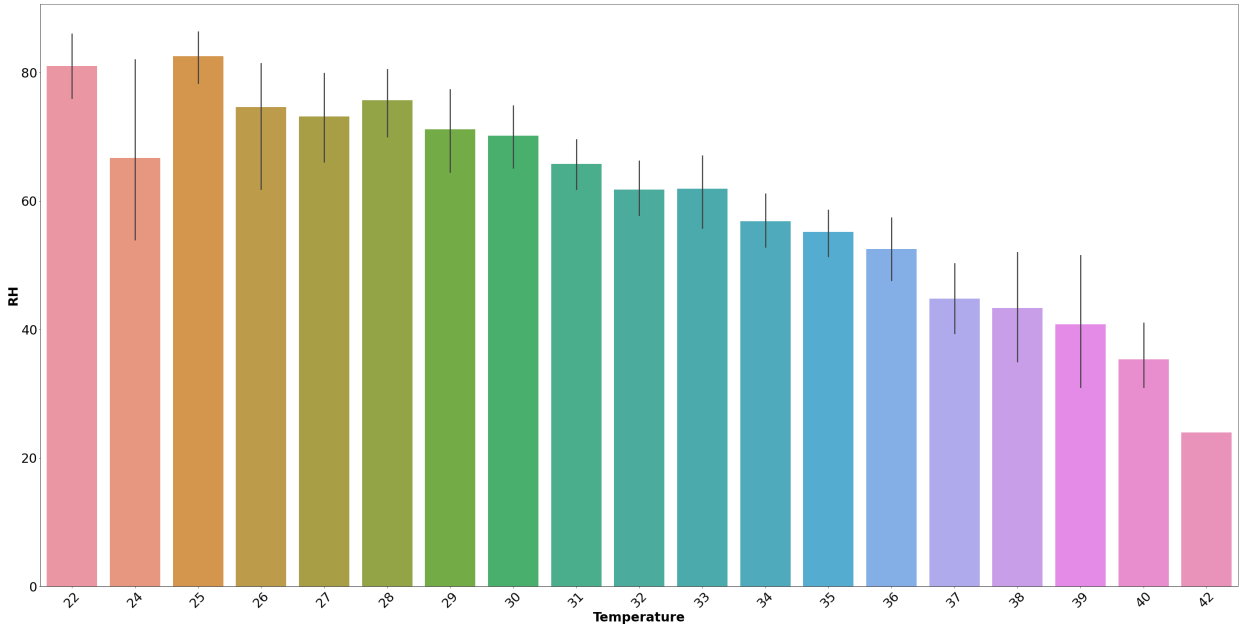


Observation

- **On temperature 22 degree celcius the Rainfall is highest.**

Relation between temperature and Relative Humidity (RH)

In [104…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Temperature', y='RH', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Temperature',weight='bold', fontsize=30);
plt.ylabel('RH',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

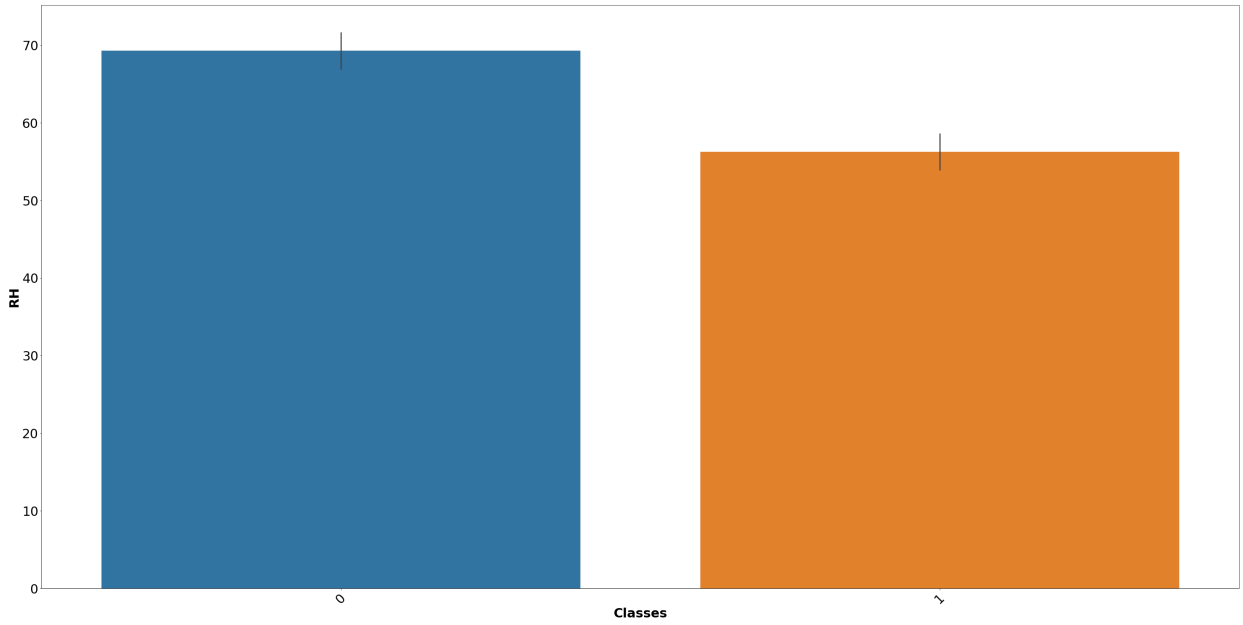


Observation

- **As we know that the Temperature and RH are inversely proportional.**
- **At 25 degree celcius we have highest relative humidity as compared to 42 degree celcius.**

Relation between Classes and Relative Humidity

In [105…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Classes', y='RH', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Classes',weight='bold', fontsize=30);
plt.ylabel('RH',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```



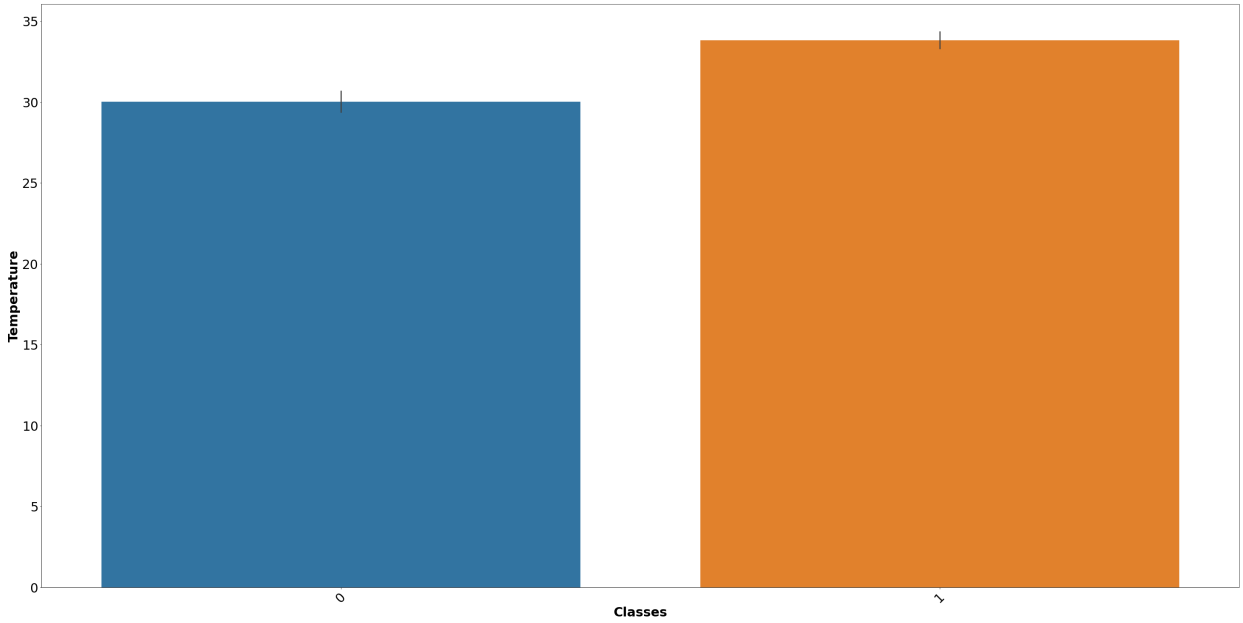
Observation

- **As the Relative Humidity goes high the water droplets in the atmosphere gets increases the chances of catching forest fire decreases.**

Relation between Classes and Temperature

In [106…

```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Classes', y='Temperature', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Classes',weight='bold', fontsize=30);
plt.ylabel('Temperature',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

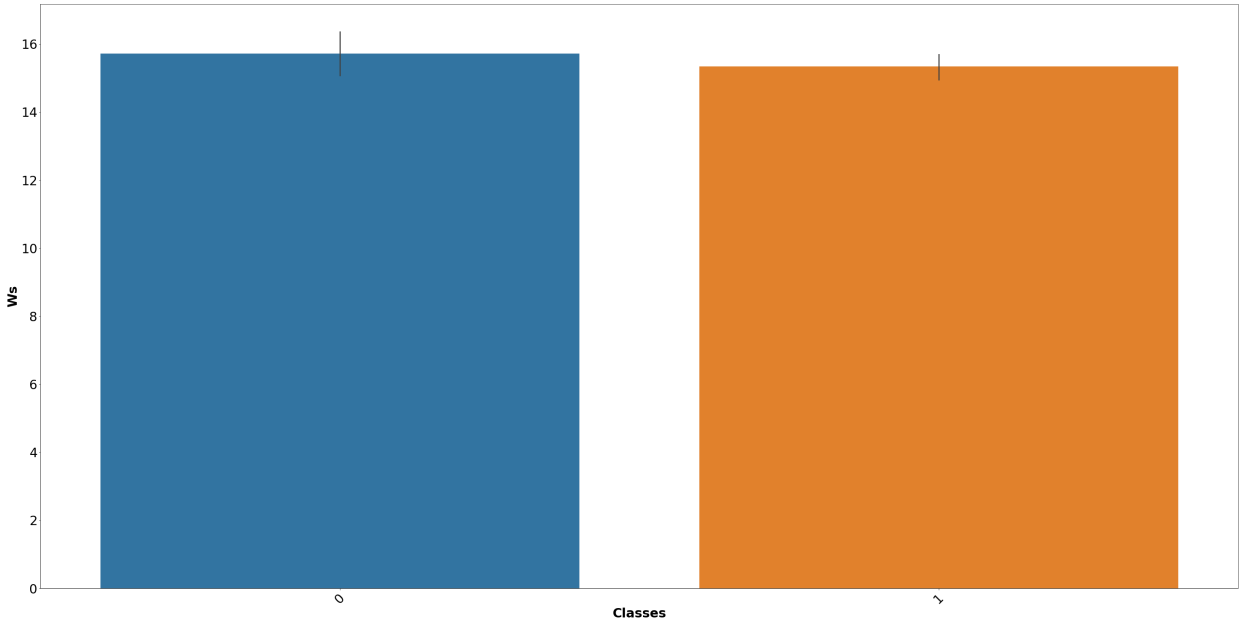


Observation

- **Here, as the temperature goes up the chances of catching fire will also increases.**

Relation between Classes and Wind speed (Ws)

In [107…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Classes', y='Ws', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Classes',weight='bold', fontsize=30);
plt.ylabel('Ws',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

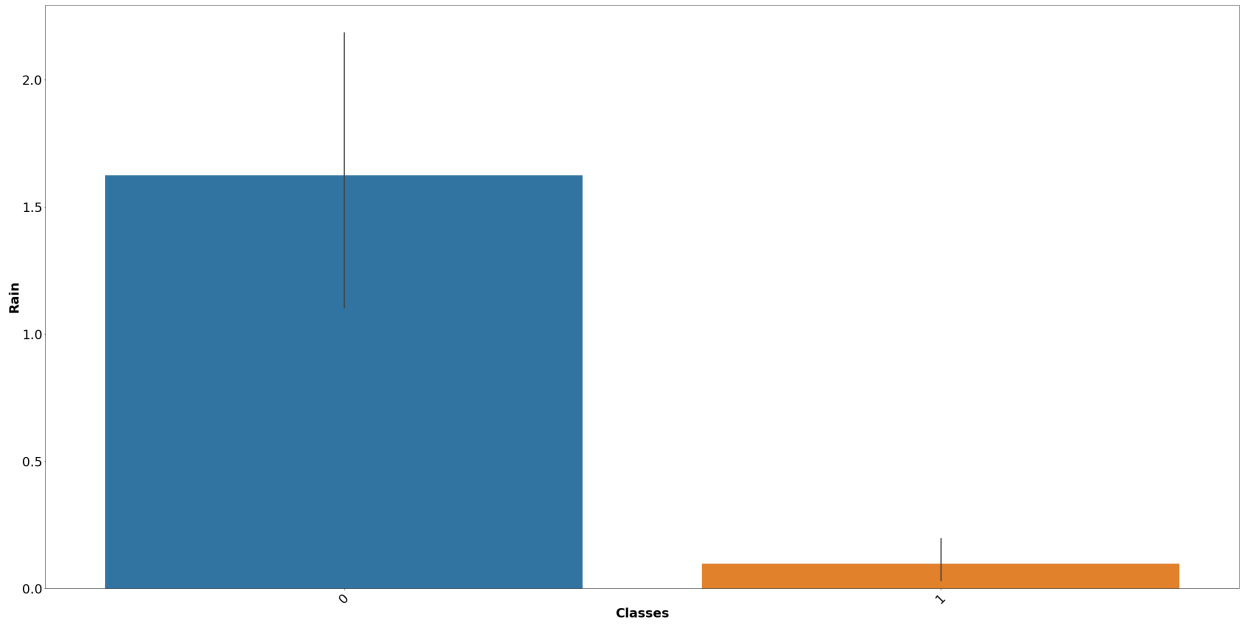


Observation

- **Here the chances of catching forest fire is somewhat equal with respect to Wind speed.**

Relation between Classes and Rain

In [108…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Classes', y='Rain', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Classes',weight='bold', fontsize=30);
plt.ylabel('Rain',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```



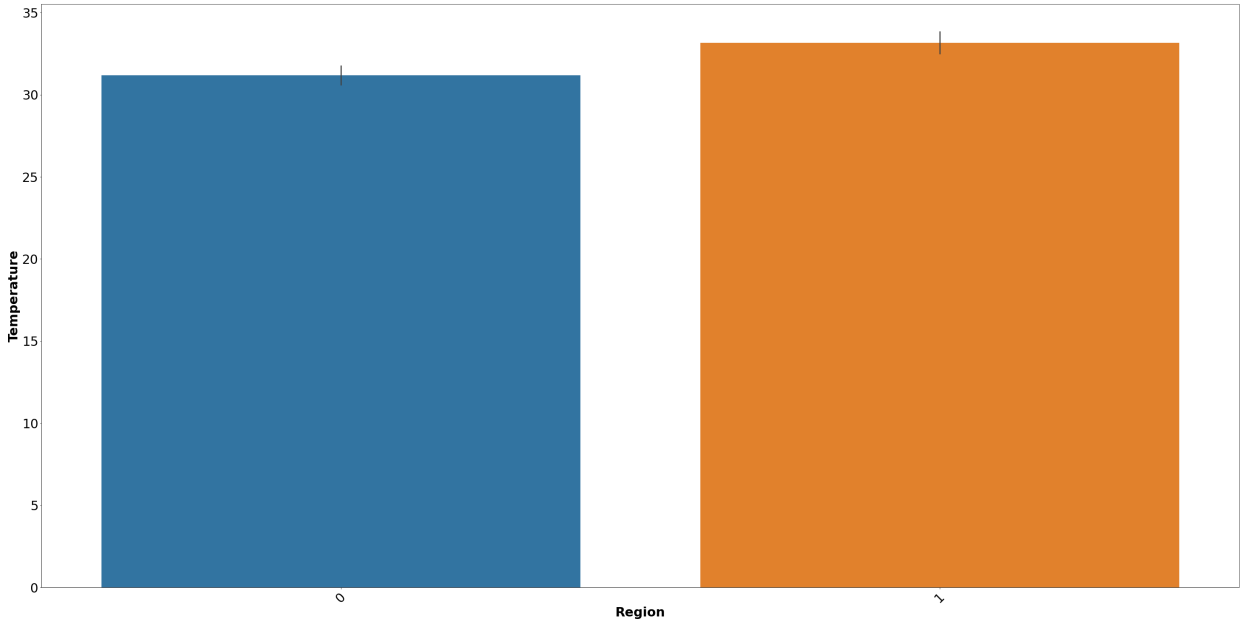
Observation

- **It is obvious that when chances of getting rainfall is high, then chances of catching fire will be going low.**

Relation between Region and temperature

In [109…

```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Region', y='Temperature', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Region',weight='bold', fontsize=30);
plt.ylabel('Temperature',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

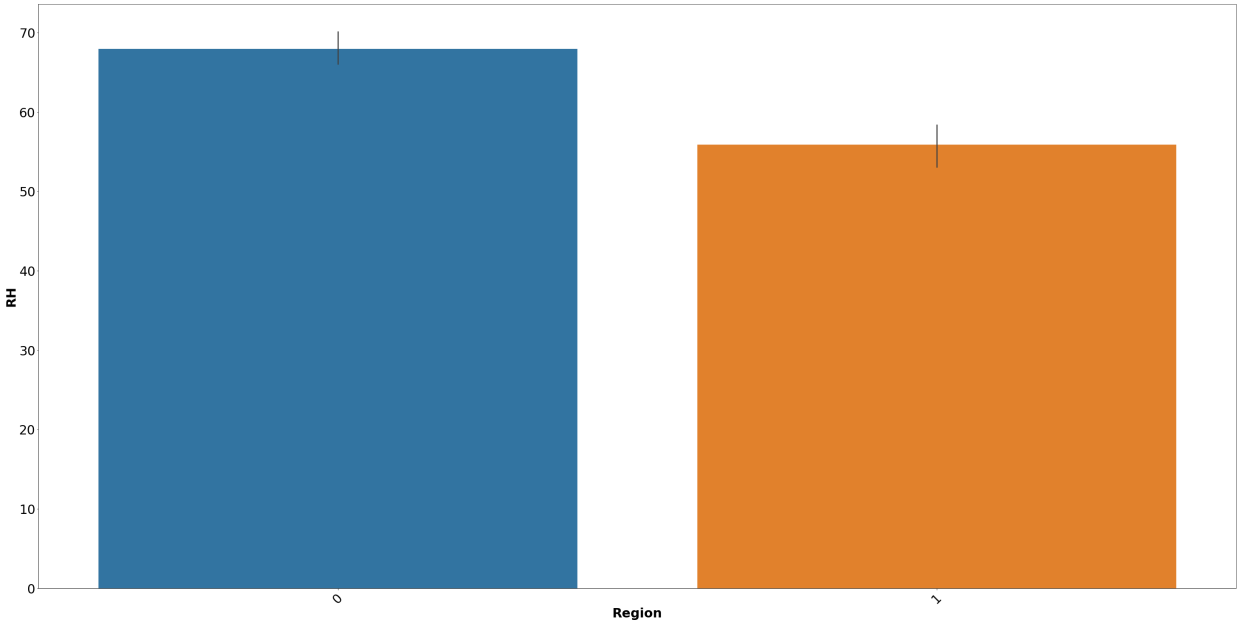


Observation

- **Sidi Bel-abbes Region experiences higher Temperature than Bejaia region.**

Relation between Region and Relative Humidity (RH)

In [110…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Region', y='RH', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Region',weight='bold', fontsize=30);
plt.ylabel('RH',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

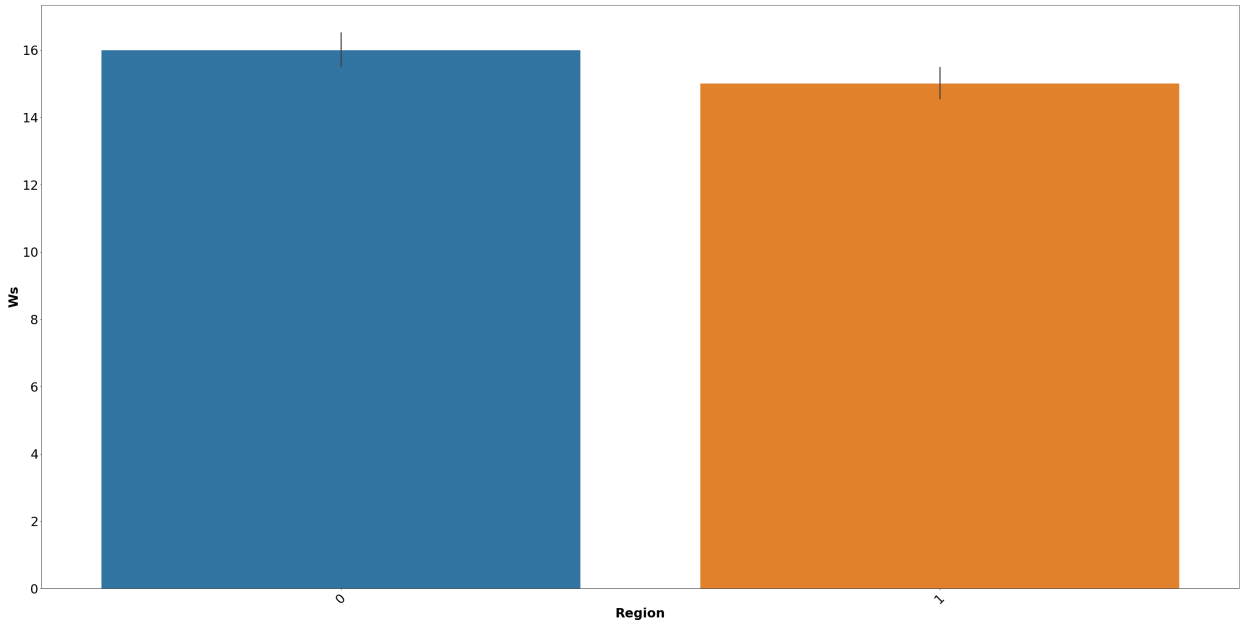


Observation

- **Sidi Bel-abbes Region experiences lower Relative Humidity than Bejaia region.**

Relation between Region and Wind Speed (Ws)

In [111…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Region', y='Ws', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Region',weight='bold', fontsize=30);
plt.ylabel('Ws',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```



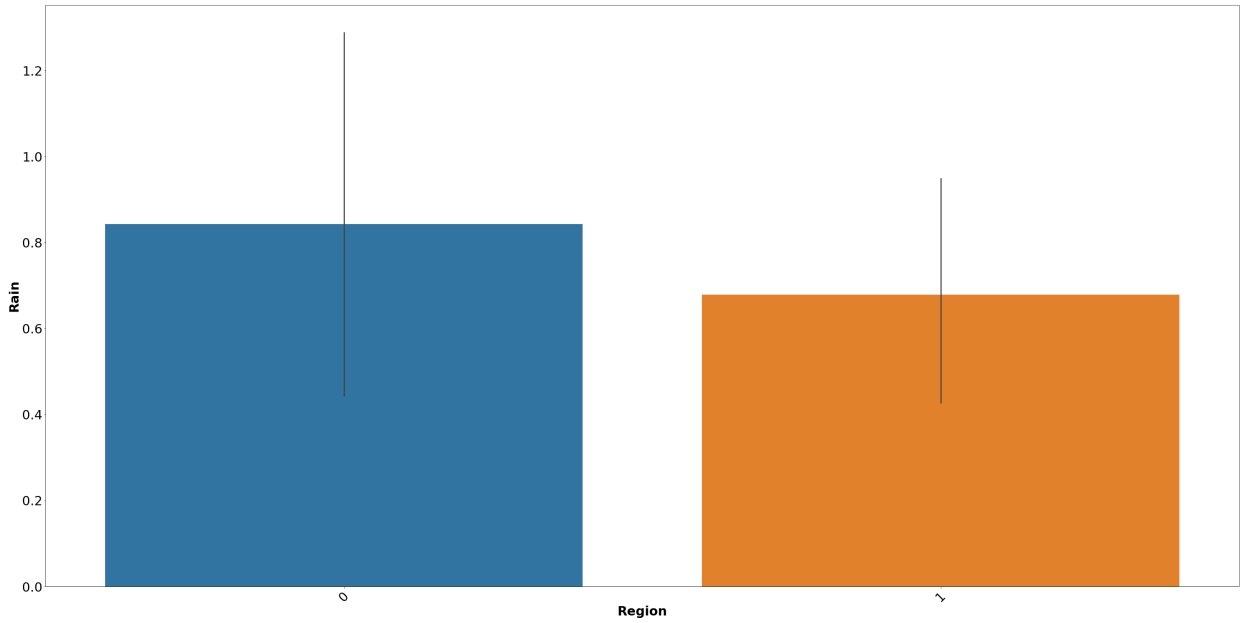
Observation

- **Sidi Bel-abbes Region experiences lesser Wind speed than Bejaia region.**

Relation between Region and Rain

In [112…

```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Region', y='Rain', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Region',weight='bold', fontsize=30);
plt.ylabel('Rain',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```

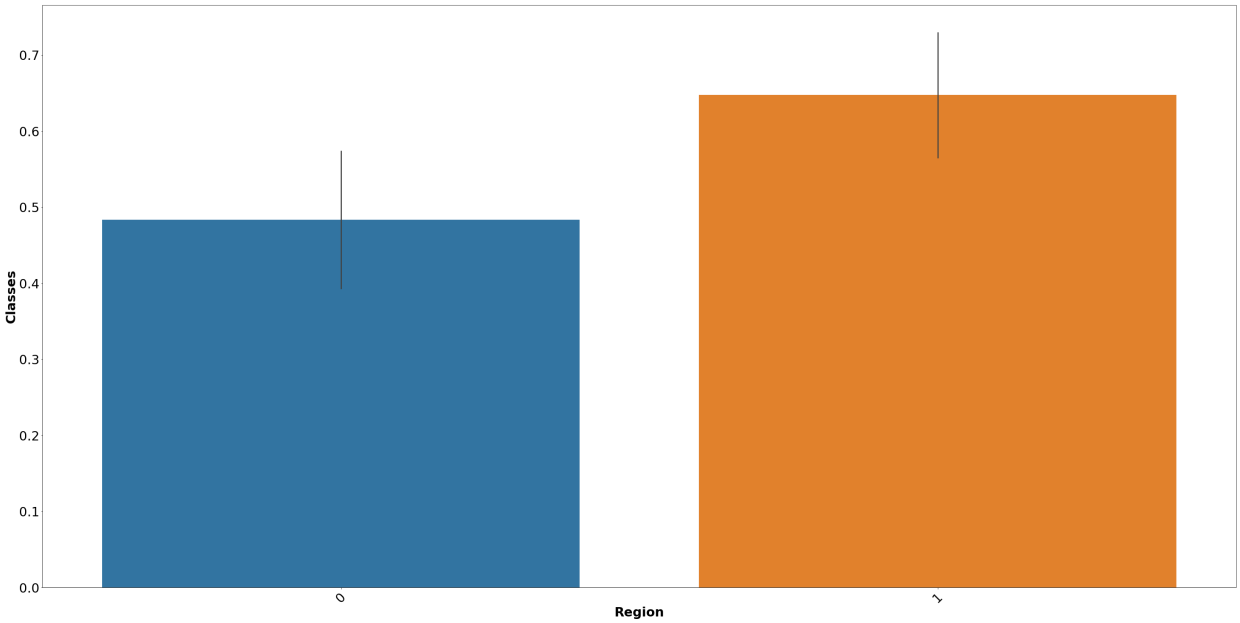


Observation

- **Sidi Bel-abbes Region experiences lesser Rainfall than Bejaia region.**

Relation between Region and Classes

In [113…
```
# plotting a barplot
plt.figure(figsize=(40,20))
sns.barplot(x='Region', y='Classes', data=df_copy)
plt.xticks(rotation=45)
plt.xlabel('Region',weight='bold', fontsize=30);
plt.ylabel('Classes',weight='bold', fontsize=30);
plt.tight_layout()
plt.tick_params(axis='both', which='major', labelsize=30)
```



Observation

- **Sidi Bel-abbes Region experiences more forest fires than Bejaia region.**

Covariance

In [114…
```
df_copy.corr()
```

Out[114]:

	day	month	Temperature	RH	Ws	Rain	FFM
day	1.000000e+00	2.232788e-17	0.095772	-0.074209	0.047001	-0.112265	0.22403
month	2.232788e-17	1.000000e+00	-0.059017	-0.037884	-0.041447	0.035322	0.01557
Temperature	9.577222e-02	-5.901677e-02	1.000000	-0.654443	-0.278132	-0.326786	0.67749
RH	-7.420934e-02	-3.788419e-02	-0.654443	1.000000	0.236084	0.222968	-0.64565
Ws	4.700086e-02	-4.144673e-02	-0.278132	0.236084	1.000000	0.170169	-0.16325
Rain	-1.122654e-01	3.532207e-02	-0.326786	0.222968	0.170169	1.000000	-0.54404
FFMC	2.240321e-01	1.557668e-02	0.677491	-0.645658	-0.163255	-0.544045	1.00000
DMC	4.915710e-01	6.817778e-02	0.483105	-0.405133	-0.001246	-0.288548	0.60239
DC	5.279285e-01	1.276719e-01	0.370498	-0.220330	0.076245	-0.296804	0.50391
ISI	1.793008e-01	6.354476e-02	0.605971	-0.688268	0.012245	-0.347862	0.74075
BUI	5.172239e-01	8.556743e-02	0.456415	-0.349685	0.030303	-0.299409	0.59025
FWI	3.502343e-01	8.173226e-02	0.566839	-0.580457	0.033957	-0.324755	0.69143
Classes	2.017844e-01	2.233266e-02	0.518119	-0.435023	-0.066529	-0.379449	0.77011
Region	4.662229e-16	-9.586232e-17	0.273496	-0.406424	-0.176829	-0.041080	0.22468

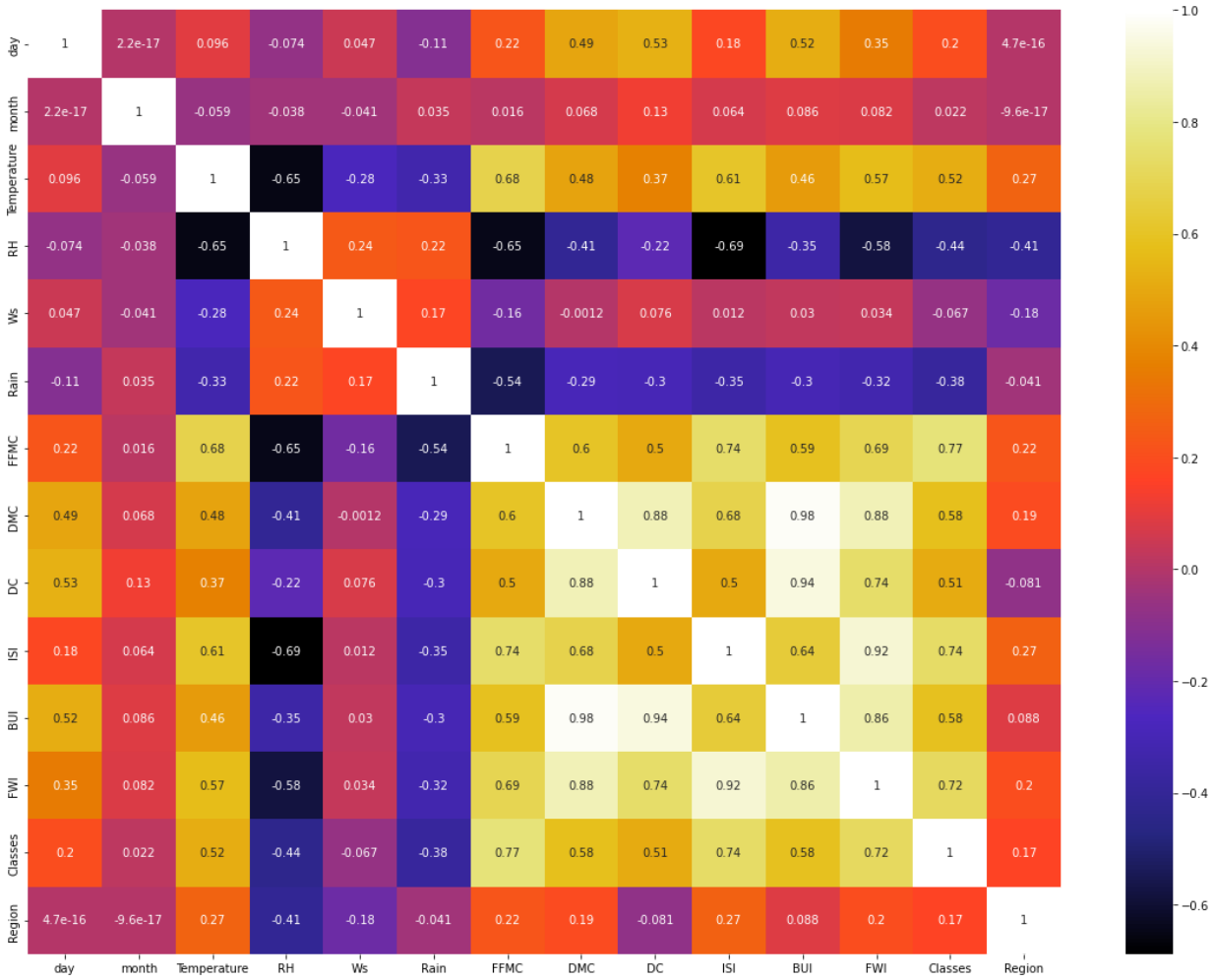
Heatmap for visualisation of correlation

In [115…

```
plt.figure(figsize = (20,15))
sns.heatmap(df_copy.corr(),cmap="CMRmap", annot=True)
```

Out[115]:

```
<AxesSubplot:>
```



Observation

- **We can clearly see that from correalation chart and heatmap, the BUI, FWI, ISI, DMC and DC are highly correlated.**

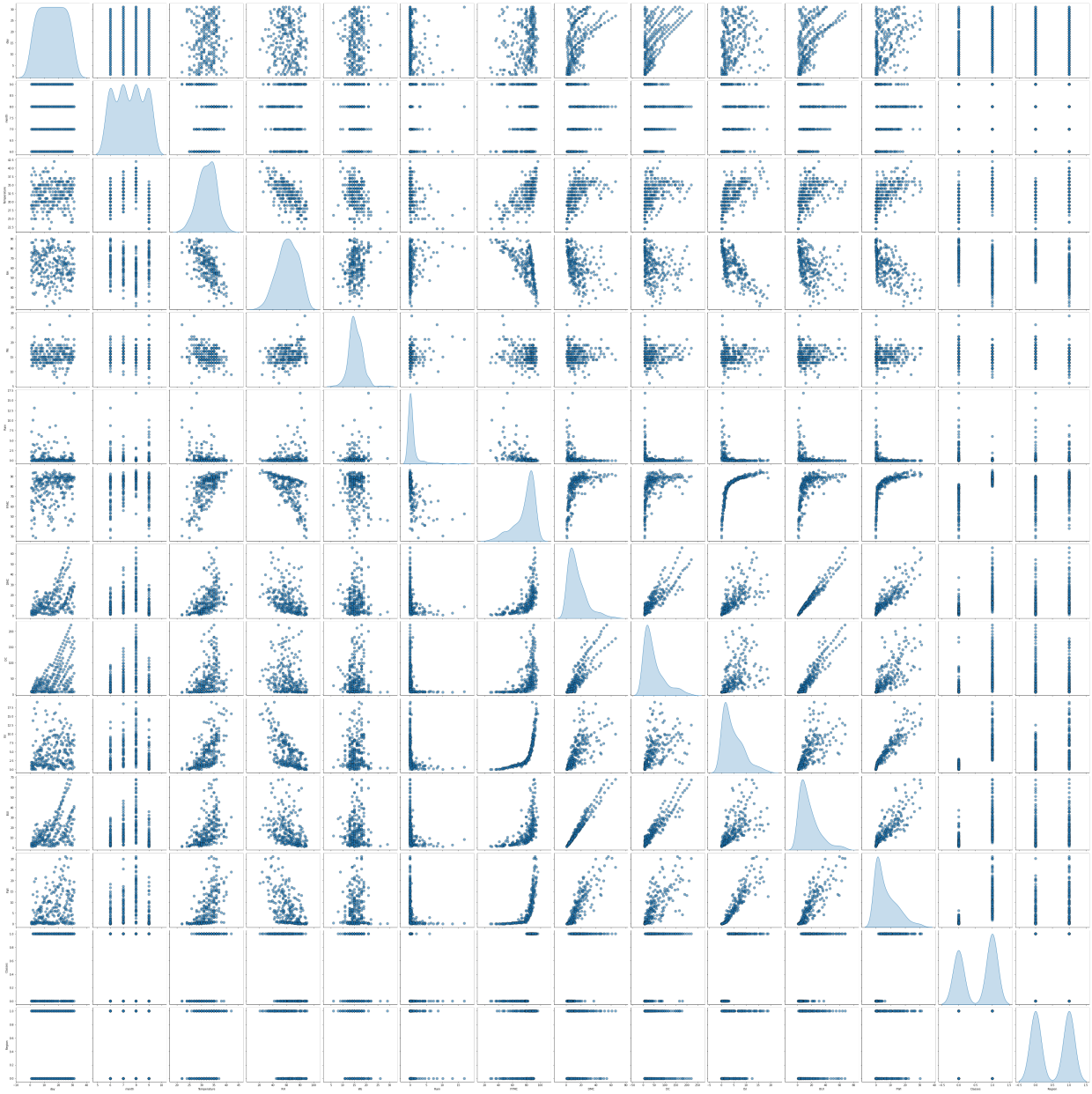
Multivariate Analysis

In [116…

```
plt.figure(figsize=(15,15))
plt.suptitle('Multivariate Analysis', fontsize=20, fontweight='bold', alpha=0.8, y=1.)
sns.pairplot(df_copy,  diag_kind = 'kde',
             plot_kws = {'alpha': 0.6, 's': 80, 'edgecolor': 'k'},
             size = 4)
```

Out[116]:

```
<seaborn.axisgrid.PairGrid at 0x23a701cd520>

<Figure size 1080x1080 with 0 Axes>
```

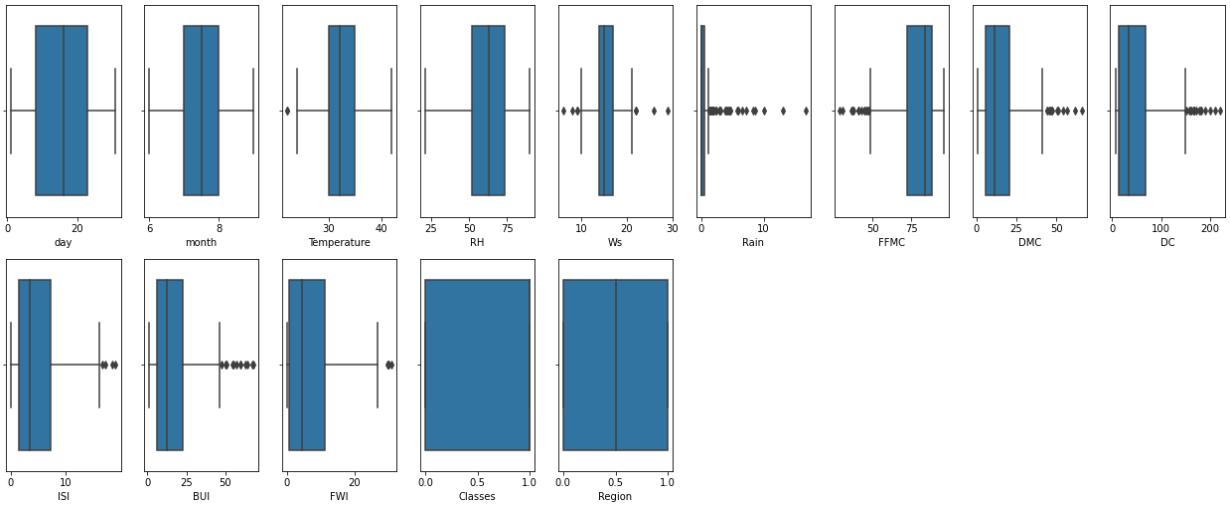


Outlier detection

Boxplot for outlier detection

In [117…

```
plt.figure(figsize=(22,18))
for i,col in enumerate(num_df.columns):
    plt.subplot(4,9,i+1)
    sns.boxplot(num_df[col])
```



Observation

- **Temperature, Ws, Rain, FFMC, DMC, DC, ISI, BUI and FWI having outliers.**

In [118…
```
df1=df_copy.copy()
df1.head(5)
```

Out[118]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	29	57	18.0	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0	0
1	2	6	29	61	13.0	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0	0
2	3	6	26	82	22.0	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0	0
3	4	6	25	89	13.0	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0	0
4	5	6	27	77	16.0	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0	0

Remove outlier function

In [119…
```
def remove_outliers_IQR(col):
    # Finding the IQR
    percentile25 = df1[col].quantile(0.25)
    percentile75 = df1[col].quantile(0.75)
    print("percentile25",percentile25)
    print("percentile75",percentile75)
    iqr = percentile75 - percentile25
    upper_limit = percentile75 + 1.5 * iqr
    lower_limit = percentile25 - 1.5 * iqr
    print("Upper limit",upper_limit)
    print("Lower limit",lower_limit)
    print("IQR", iqr)
    df1[col] = np.where(df1[col]>upper_limit, upper_limit, np.where(df1[col]<lower_lim
    return df1[df1[col] > upper_limit]
```

Comparison plot function

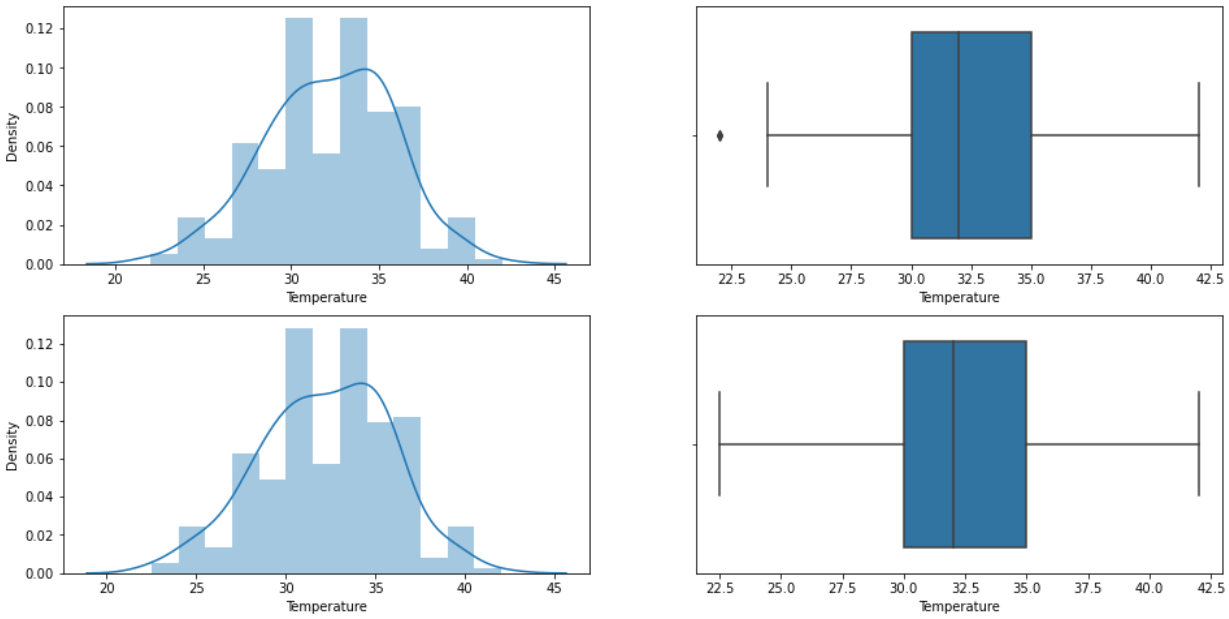
In [120…
```
def create_comparison_plot(df_copy,df1,column):
    # Comparing
    plt.figure(figsize=(16,8))
    plt.subplot(2,2,1)
    sns.distplot(df_copy[column])
```

```
    plt.subplot(2,2,2)
    sns.boxplot(df_copy[column])

    plt.subplot(2,2,3)
    sns.distplot(df1[column])

    plt.subplot(2,2,4)
    sns.boxplot(df1[column])

    plt.show()
```

Removing outliers from every columns

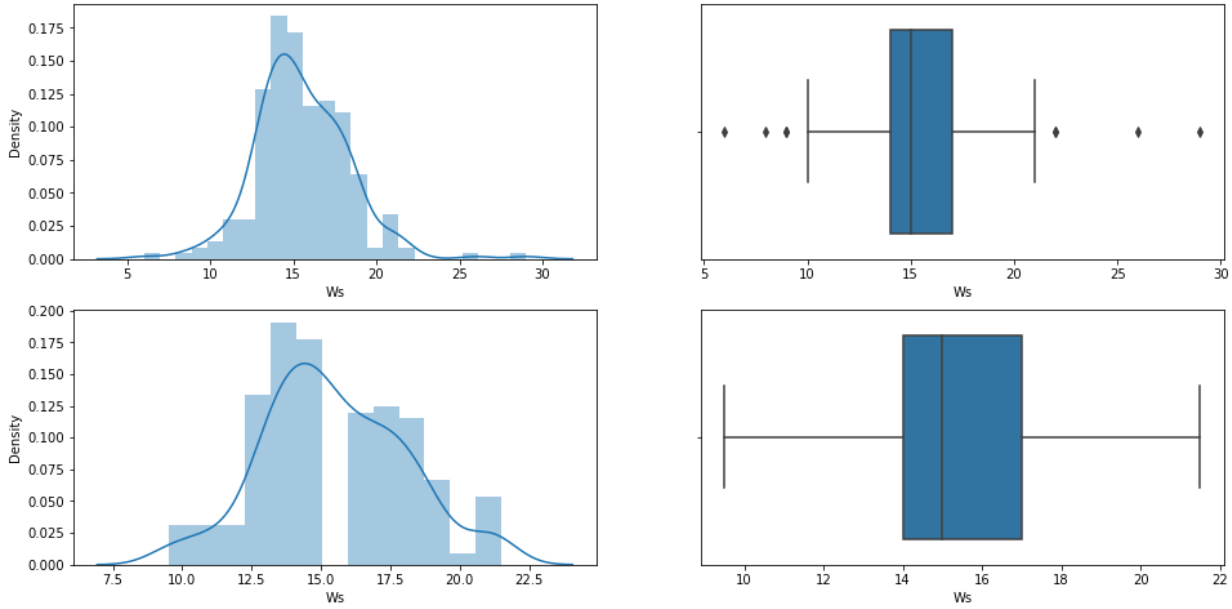
In [121…
```
remove_outliers_IQR('Temperature')
create_comparison_plot(df_copy,df1,"Temperature")
```

```
percentile25 30.0
percentile75 35.0
Upper limit 42.5
Lower limit 22.5
IQR 5.0
```



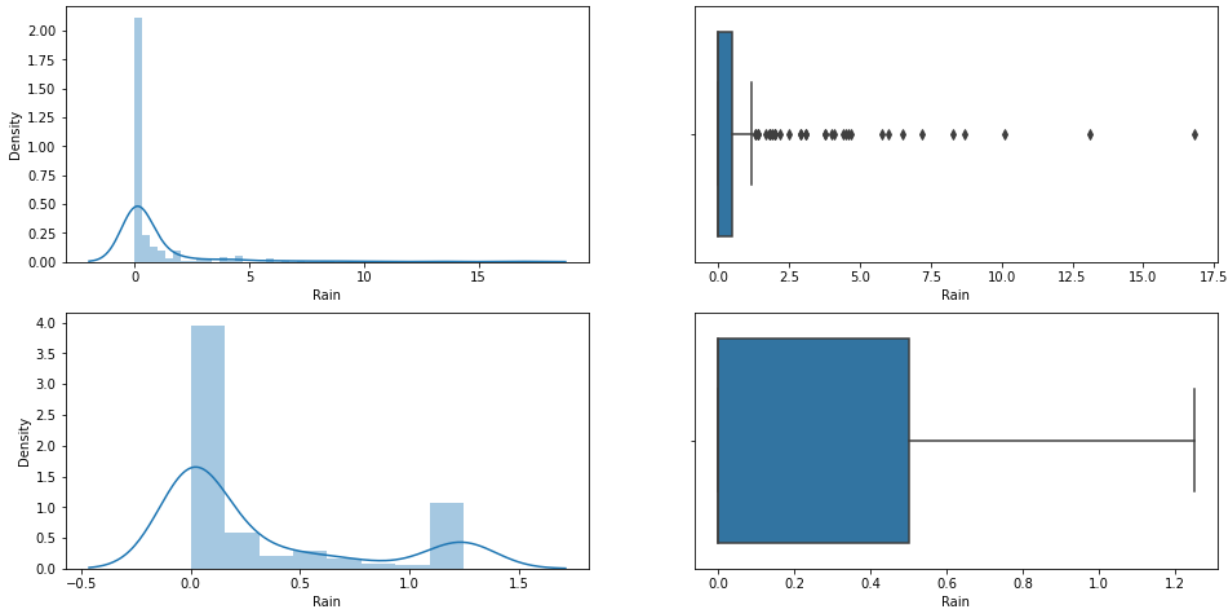
In [122…
```
remove_outliers_IQR('Ws')
create_comparison_plot(df_copy,df1,"Ws")
```

```
percentile25 14.0
percentile75 17.0
Upper limit 21.5
Lower limit 9.5
IQR 3.0
```



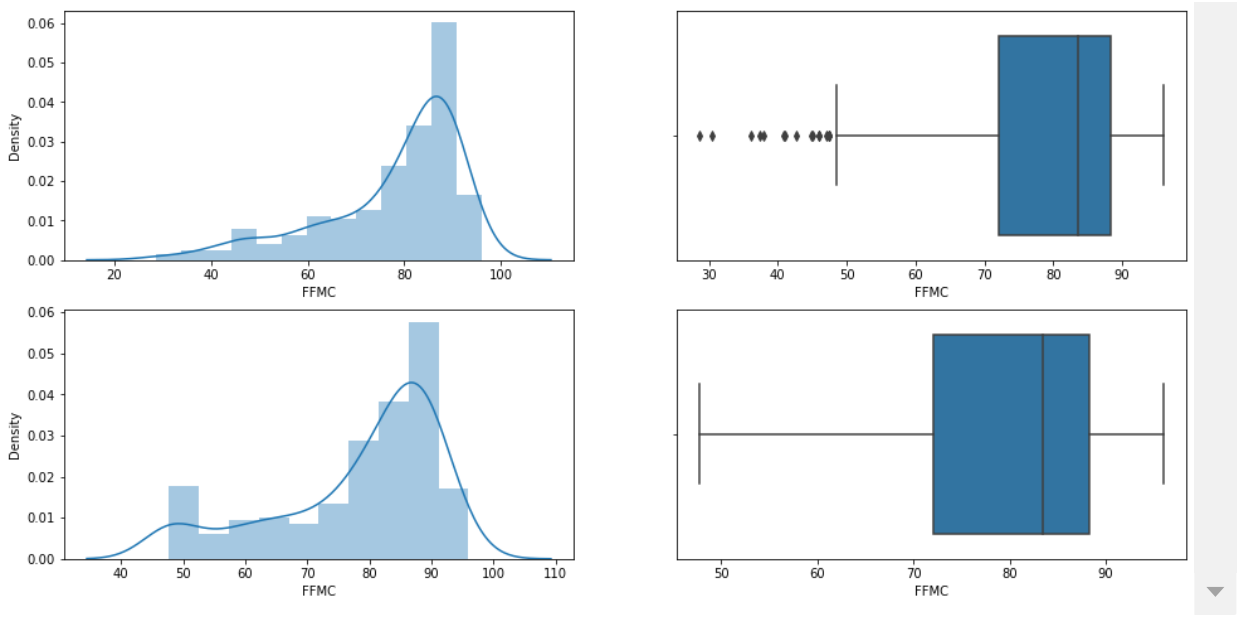
In [123…
```
remove_outliers_IQR('Rain')
create_comparison_plot(df_copy,df1,"Rain")
```

```
percentile25 0.0
percentile75 0.5
Upper limit 1.25
Lower limit -0.75
IQR 0.5
```



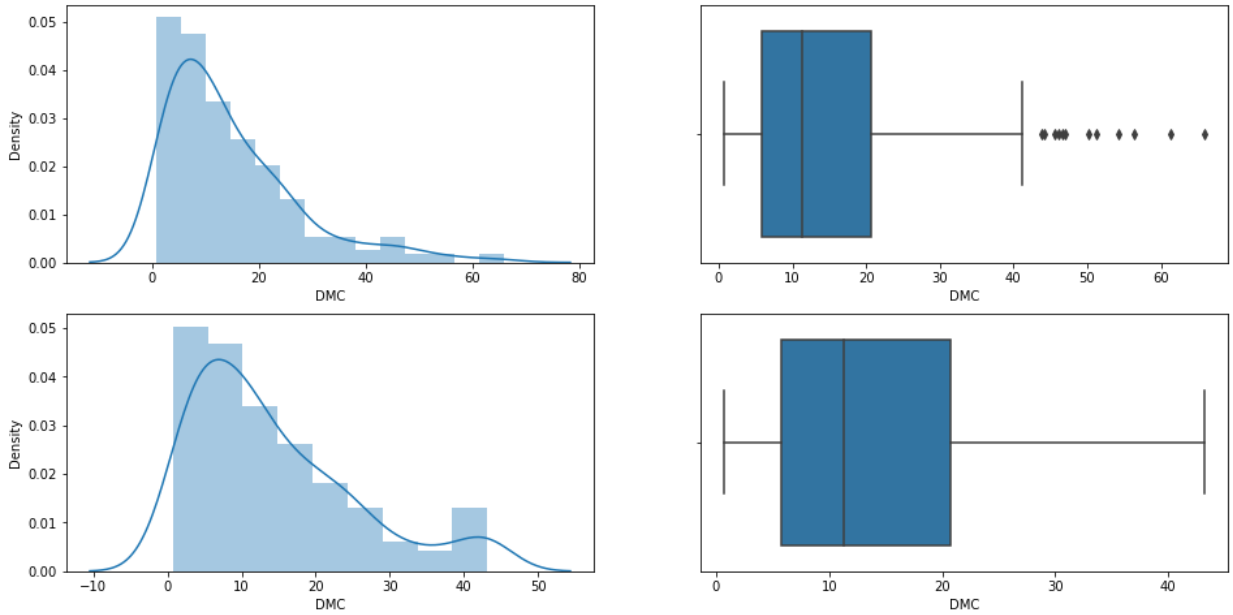
In [124…
```
remove_outliers_IQR('FFMC')
create_comparison_plot(df_copy,df1,"FFMC")
```

```
percentile25 72.075
percentile75 88.3
Upper limit 112.63749999999999
Lower limit 47.73750000000001
IQR 16.224999999999994
```



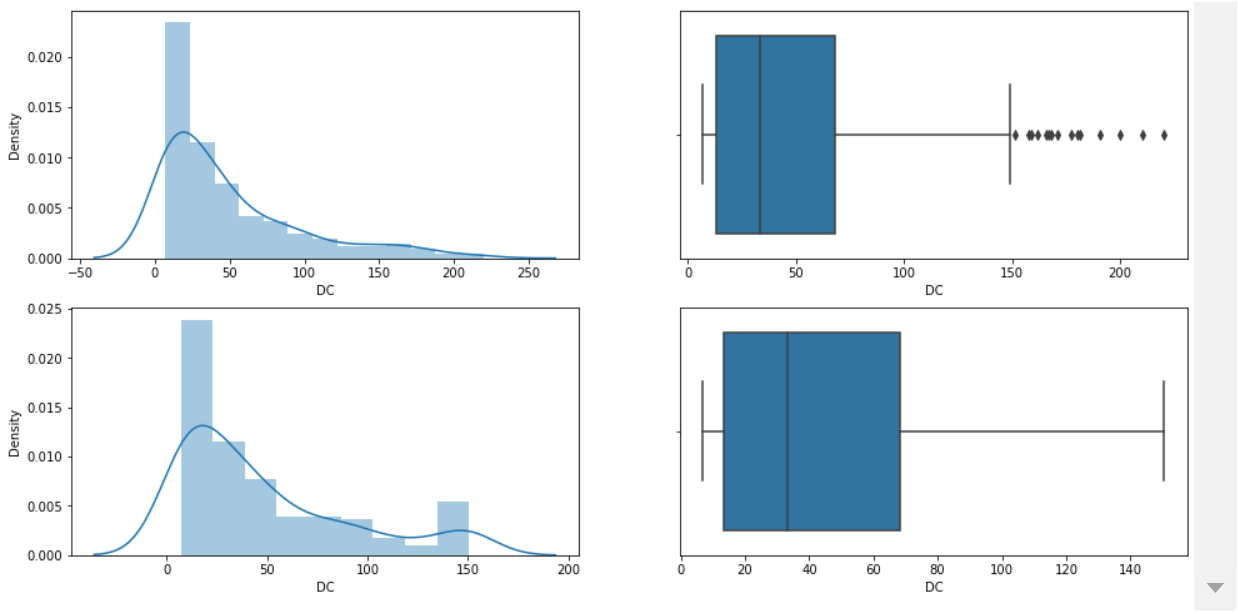
In [125…
```
remove_outliers_IQR('DMC')
create_comparison_plot(df_copy,df1,"DMC")
```

```
percentile25 5.8
percentile75 20.75
Upper limit 43.175
Lower limit -16.624999999999996
IQR 14.95
```



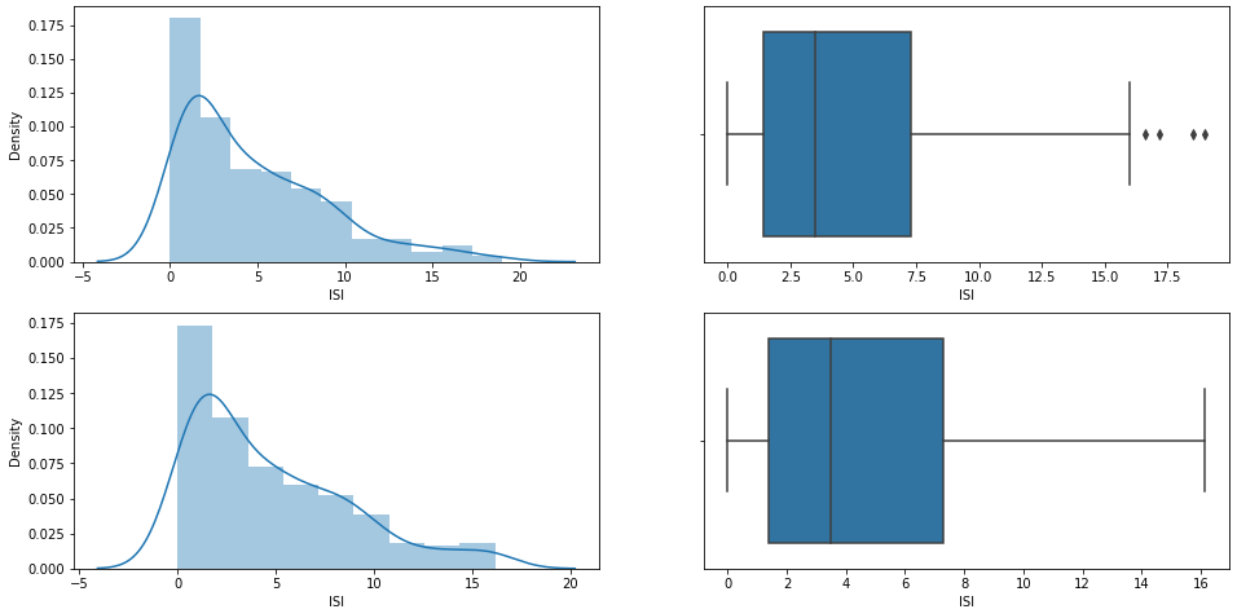
In [126…
```
remove_outliers_IQR('DC')
create_comparison_plot(df_copy,df1,"DC")
```

```
percentile25 13.274999999999999
percentile75 68.15
Upper limit 150.46250000000003
Lower limit -69.03750000000002
IQR 54.87500000000001
```



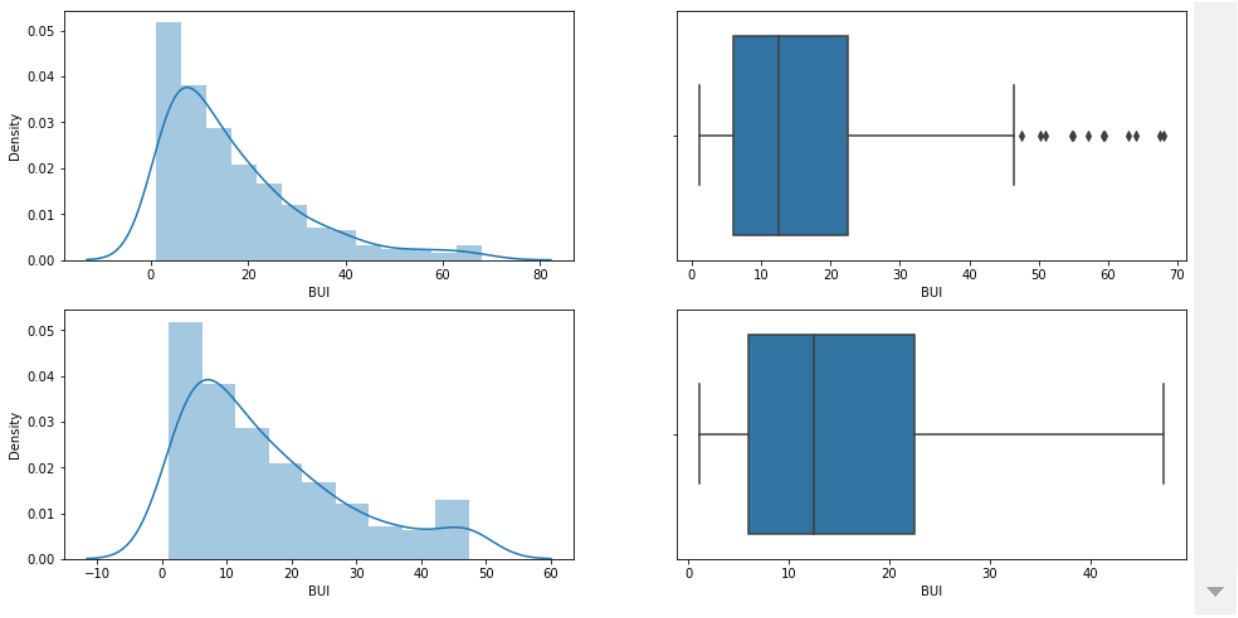
In [127…
```
remove_outliers_IQR('ISI')
create_comparison_plot(df_copy,df1,"ISI")
```

```
percentile25 1.4
percentile75 7.3
Upper limit 16.150000000000002
Lower limit -7.450000000000001
IQR 5.9
```



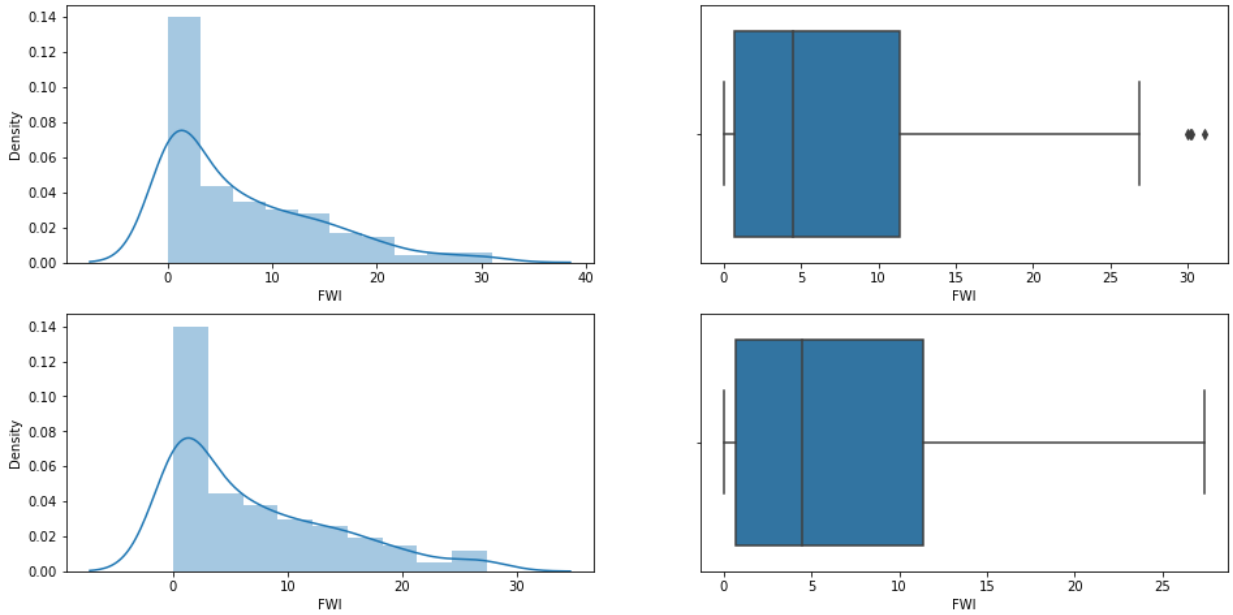
In [128…
```
remove_outliers_IQR('BUI')
create_comparison_plot(df_copy,df1,"BUI")
```

```
percentile25 6.0
percentile75 22.525
Upper limit 47.3125
Lower limit -18.787499999999998
IQR 16.525
```



In [129…
```
remove_outliers_IQR('FWI')
create_comparison_plot(df_copy,df1,"FWI")
```

```
percentile25 0.7
percentile75 11.375
Upper limit 27.387500000000003
Lower limit -15.312500000000004
IQR 10.675
```

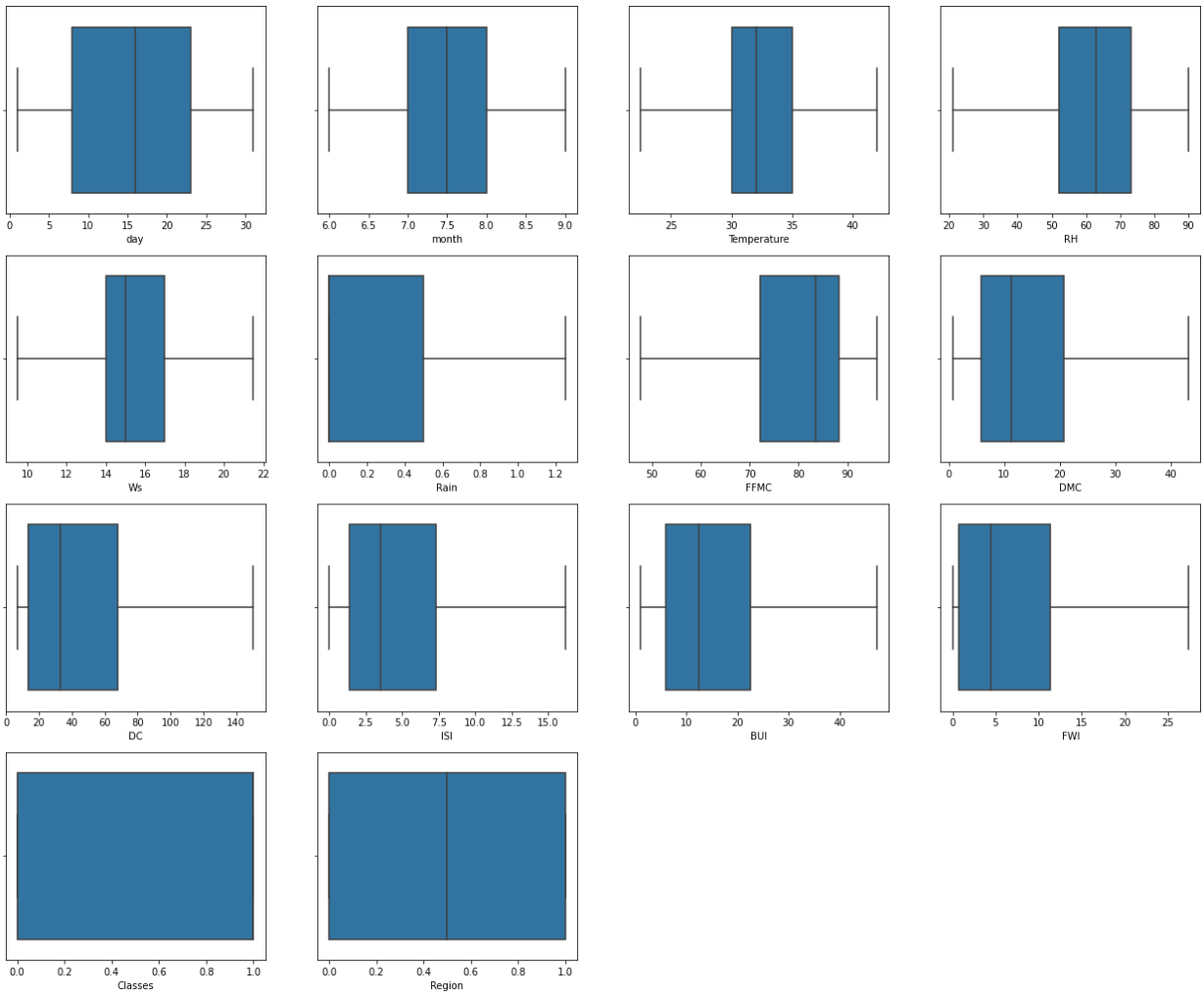


Observation

- **Using IQR method the outliers have been removed from all the features.**
- **In the above results we can clearly see before and after applying IQR method on outliers.**

Rechecking all columns after removing outliers

In [130…
```
plt.figure(figsize=(22,18))
for i,col in enumerate(df1.columns):
    plt.subplot(4,4,i+1)
    sns.boxplot(df1[col])
```



Observation

- **Outliers removed from all the mentioned features.**

Compare skewness

In [131…
```
df_copy.skew()
```

Out[131]:
```
day            0.002806
month          0.000000
Temperature   -0.196309
RH            -0.237964
Ws             0.545881
Rain           4.579071
FFMC          -1.325633
DMC            1.527652
DC             1.479042
ISI            1.126950
BUI            1.458466
FWI            1.143243
Classes       -0.266220
Region         0.000000
dtype: float64
```

In [132…
```
df1.skew()
```

Out[132]:
```
day             0.002806
month           0.000000
Temperature    -0.175783
RH             -0.237964
Ws              0.177613
Rain            1.246290
FFMC           -1.073835
DMC             1.089909
DC              1.159322
ISI             1.024383
BUI             1.020148
FWI             1.048454
Classes        -0.266220
Region          0.000000
dtype: float64
```

Observation

- **Skewness is reduced after we have removed ouliers using IQR Method.**

5. Model Building

Creating independent and dependent variables

In [133…
```
x = df1.drop(columns = ['Classes'])
y = df1['Classes']
```

Independent variable

In [134…
```
x
```

Out[134]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
0	1	6	29.0	57	18.0	0.00	65.7000	3.4	7.6	1.3	3.4	0.5	0
1	2	6	29.0	61	13.0	1.25	64.4000	4.1	7.6	1.0	3.9	0.4	0
2	3	6	26.0	82	21.5	1.25	47.7375	2.5	7.1	0.3	2.7	0.1	0
3	4	6	25.0	89	13.0	1.25	47.7375	1.3	6.9	0.0	1.7	0.0	0
4	5	6	27.0	77	16.0	0.00	64.8000	3.0	14.2	1.2	3.9	0.5	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…
239	26	9	30.0	65	14.0	0.00	85.4000	16.0	44.5	4.5	16.9	6.5	1
240	27	9	28.0	87	15.0	1.25	47.7375	6.5	8.0	0.1	6.2	0.0	1
241	28	9	27.0	87	21.5	0.50	47.7375	3.5	7.9	0.4	3.4	0.2	1
242	29	9	24.0	54	18.0	0.10	79.7000	4.3	15.2	1.7	5.1	0.7	1
243	30	9	24.0	64	15.0	0.20	67.3000	3.8	16.5	1.2	4.8	0.5	1

244 rows × 13 columns

In [135…
```
x.shape
```

Out[135]:
```
(244, 13)
```

In [136…
```
y
```

Out[136]:
```
0      0
1      0
2      0
3      0
4      0
      ..
239    1
240    0
241    0
242    0
243    0
Name: Classes, Length: 244, dtype: int32
```

In [137…
```
y.shape
```

Out[137]:
```
(244,)
```

Importing Machine Learning libraries

In [139…
```
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model  import  LogisticRegression
from sklearn.model_selection import train_test_split
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.metrics import accuracy_score, confusion_matrix, roc_curve, roc_auc_score
```

Train Test Split

In [140…
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=42)
```

Logistic Regression

In [141…
```
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
```

In [142…
```
from sklearn.model_selection import GridSearchCV
parameter={'penalty':['l1','l2','elasticnet'],'C':[1,2,3,4,5,6,10,20,30,40,50],'max_it
```

In [143…
```
classifier_regressor=GridSearchCV(classifier,param_grid=parameter,scoring='accuracy',c
```

Standardization

In [144…
```
classifier_regressor.fit(x_train,y_train)
```

Out[144]:
```
GridSearchCV
estimator: LogisticRegression
LogisticRegression
```

In [145…
```
print(classifier_regressor.best_params_)
```

```
{'C': 1, 'max_iter': 300, 'penalty': 'l2'}
```

In [146…
```
print(classifier_regressor.best_score_)
```

```
0.9669669669669669
```

Prediction

In [147…
```
y_pred = classifier_regressor.predict(x_test)
```

Accuracy

In [148…
```
from sklearn.metrics import accuracy_score,classification_report
score=accuracy_score(y_pred,y_test)
print(score)
```

```
0.9836065573770492
```

Classification Report

In [149…
```
print(classification_report(y_pred,y_test))
```

```
              precision    recall  f1-score   support

           0       1.00      0.96      0.98        25
           1       0.97      1.00      0.99        36

    accuracy                           0.98        61
   macro avg       0.99      0.98      0.98        61
weighted avg       0.98      0.98      0.98        61
```

Performance Metrics

Confusion metrics

In [150…
```
conf_mat=confusion_matrix(y_pred,y_test)
```

In [151…
```
conf_mat
```

Out[151]:
```
array([[24,  1],
       [ 0, 36]], dtype=int64)
```

In [152…
```
true_positive = conf_mat[0][0]
false_positive = conf_mat[0][1]
false_negative = conf_mat[1][0]
true_negative = conf_mat[1][1]
```

Precision

In [153…
```
Precision = true_positive/(true_positive+false_positive)
Precision
```

Out[153]:
```
0.96
```

Recall

In [154…
```
Recall = true_positive/(true_positive+false_negative)
Recall
```

Out[154]:
```
1.0
```

F1 Score

In [155…
```
F1_Score = 2*(Recall * Precision) / (Recall + Precision)
F1_Score
```

Out[155]:
```
0.9795918367346939
```

AUC

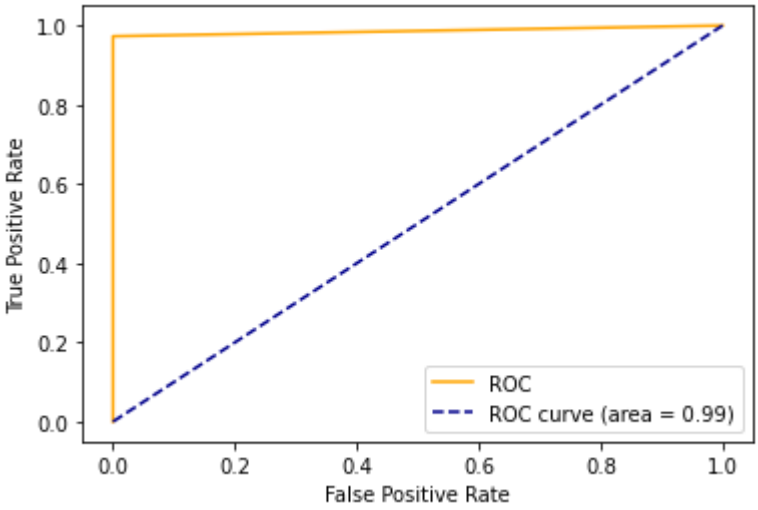
In [156…
```
auc = roc_auc_score(y_test, y_pred)
auc
```

Out[156]:
```
0.9864864864864865
```

ROC

In [157…
```
fpr, tpr, thresholds = roc_curve(y_test, y_pred)
```

In [158…
```
plt.plot(fpr, tpr, color='orange', label='ROC')
plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--',label='ROC curve (area = %0.
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend()
plt.show()
```



Creating Imbalance dataset

In [159…
```
df_copy.head()
```

Out[159]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	29	57	18.0	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0	0
1	2	6	29	61	13.0	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0	0
2	3	6	26	82	22.0	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0	0
3	4	6	25	89	13.0	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0	0
4	5	6	27	77	16.0	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0	0

In [160…
```
df_copy.shape
```

Out[160]:
```
(244, 14)
```

Splitting data in 90:10 percent ratio using train test split

In [161…
```
x1 = pd.DataFrame(df_copy, columns = ['day', 'month','Temperature','RH','Ws','Rain','F
y1=pd.DataFrame(df_copy,columns = ['Classes'])
```

In [162…
```
x_train_imb, x_test_imb, y_train_imb, y_test_imb = train_test_split(x1, y1, test_size=
```

In [163…
```
x_train_imb.shape, y_train_imb.shape
```

Out[163]:
```
((219, 13), (219, 1))
```

Replacing all values as 1 in y_train and all values as 0 in y_test to creating imbalance

In [164…
```
y_train_imb=y_train_imb.replace(0,1)
y_train_imb.head()
```

Out[164]:

	Classes
134	1
13	1
163	1
65	1
87	1

In [165…
```
y_test_imb=y_test_imb.replace(1,0)
y_test_imb.head()
```

Out[165]:

	Classes
141	0
218	0
185	0
20	0
132	0

In [166…
```
x_train_imb.head()
```

Out[166]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
134	13	6	30	52	15.0	2.0	72.3	11.4	7.8	1.4	10.9	0.9	1
13	14	6	30	78	20.0	0.5	59.0	4.6	7.8	1.0	4.4	0.4	0
163	12	7	36	44	13.0	0.0	90.1	12.6	19.4	8.3	12.5	9.6	1
65	5	8	34	65	13.0	0.0	86.8	11.1	29.7	5.2	11.5	6.1	0
87	27	8	33	82	21.0	0.0	84.9	47.0	200.2	4.4	59.3	13.2	0

Combining x_train_imb and y_train_imb

In [167…
```
train_imb=x_train_imb.join(pd.DataFrame(y_train_imb))
train_imb.head()
```

Out[167]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Class
134	13	6	30	52	15.0	2.0	72.3	11.4	7.8	1.4	10.9	0.9	1	
13	14	6	30	78	20.0	0.5	59.0	4.6	7.8	1.0	4.4	0.4	0	
163	12	7	36	44	13.0	0.0	90.1	12.6	19.4	8.3	12.5	9.6	1	
65	5	8	34	65	13.0	0.0	86.8	11.1	29.7	5.2	11.5	6.1	0	
87	27	8	33	82	21.0	0.0	84.9	47.0	200.2	4.4	59.3	13.2	0	

Combining x_test_imb and y_test_imb

In [168…
```
test_imb=x_test_imb.join(pd.DataFrame(y_test_imb))
test_imb.head()
```

Out[168]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Class
141	20	6	31	72	14.0	0.2	60.2	3.8	8.0	0.8	3.7	0.3	1	
218	5	9	30	58	12.0	4.1	66.1	4.0	8.4	1.0	3.9	0.4	1	
185	3	8	39	33	17.0	0.0	93.7	17.1	32.1	17.2	16.9	19.5	1	
20	21	6	30	78	14.0	0.0	81.0	6.3	31.6	2.6	8.4	2.2	0	
132	11	6	31	42	21.0	0.0	90.6	18.2	30.5	13.4	18.0	16.7	1	

In [169…
```
train_imb.shape, test_imb.shape
```

Out[169]:
```
((219, 14), (25, 14))
```

Combining train_imb dataset and test_imb dataset into data_imb dataset

In [170…
```
df_imb=pd.concat([train_imb, test_imb], ignore_index=True, sort=False)
df_imb.head()
```

Out[170]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	13	6	30	52	15.0	2.0	72.3	11.4	7.8	1.4	10.9	0.9	1	1
1	14	6	30	78	20.0	0.5	59.0	4.6	7.8	1.0	4.4	0.4	0	1
2	12	7	36	44	13.0	0.0	90.1	12.6	19.4	8.3	12.5	9.6	1	1
3	5	8	34	65	13.0	0.0	86.8	11.1	29.7	5.2	11.5	6.1	0	1
4	27	8	33	82	21.0	0.0	84.9	47.0	200.2	4.4	59.3	13.2	0	1

In [171…
```
df_imb.shape
```

Out[171]:
```
(244, 14)
```

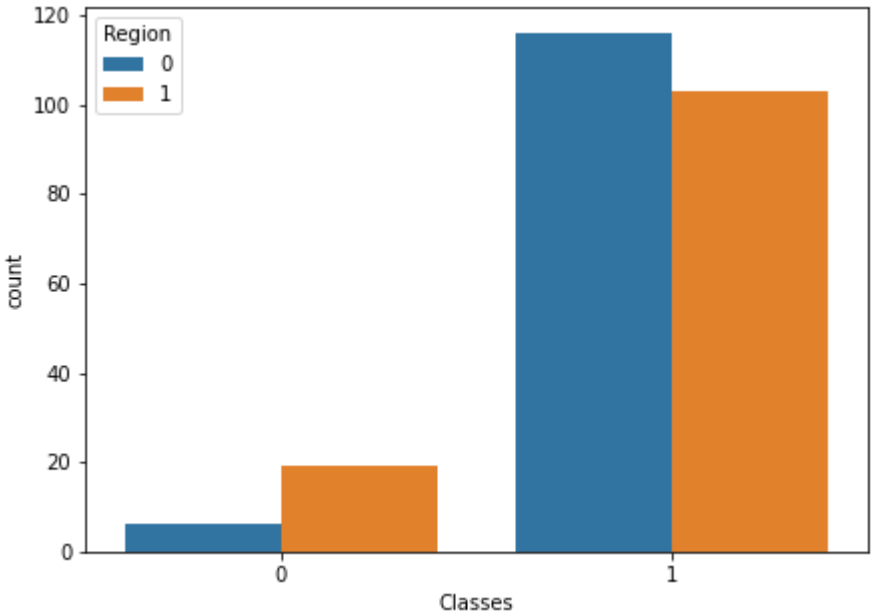
Checking imbalancing

In [172…
```
df_imb.Classes.value_counts()
```

Out[172]:
```
1    219
0     25
Name: Classes, dtype: int64
```

In [173…
```
## 0 is 'Bejaia' and 1 is 'Sidi Bel-abbes region'
plt.figure(figsize=(7,5))
sns.countplot(data=df_imb,x='Classes',hue='Region')
```

Out[173]:
```
<AxesSubplot:xlabel='Classes', ylabel='count'>
```



Logistic Regression on imbalance dataset

In [174…
```
df_imb.head()
```

Out[174]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	13	6	30	52	15.0	2.0	72.3	11.4	7.8	1.4	10.9	0.9	1	1
1	14	6	30	78	20.0	0.5	59.0	4.6	7.8	1.0	4.4	0.4	0	1
2	12	7	36	44	13.0	0.0	90.1	12.6	19.4	8.3	12.5	9.6	1	1
3	5	8	34	65	13.0	0.0	86.8	11.1	29.7	5.2	11.5	6.1	0	1
4	27	8	33	82	21.0	0.0	84.9	47.0	200.2	4.4	59.3	13.2	0	1

In [175…
```
x1 = df_imb.drop(columns = ['Classes'])
y1 = df_imb['Classes']
```

In [176…
```
x1
```

Out[176]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
0	13	6	30	52	15.0	2.0	72.3	11.4	7.8	1.4	10.9	0.9	1
1	14	6	30	78	20.0	0.5	59.0	4.6	7.8	1.0	4.4	0.4	0
2	12	7	36	44	13.0	0.0	90.1	12.6	19.4	8.3	12.5	9.6	1
3	5	8	34	65	13.0	0.0	86.8	11.1	29.7	5.2	11.5	6.1	0
4	27	8	33	82	21.0	0.0	84.9	47.0	200.2	4.4	59.3	13.2	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…
239	30	8	34	49	15.0	0.0	89.2	24.8	159.1	8.1	35.7	16.0	1
240	11	9	30	73	14.0	0.0	79.2	6.5	16.6	2.1	6.6	1.2	1
241	22	9	33	64	13.0	0.0	88.9	26.1	106.3	7.1	32.4	13.7	1
242	1	9	25	76	17.0	7.2	46.0	1.3	7.5	0.2	1.8	0.1	0
243	30	6	34	42	15.0	1.7	79.7	12.0	8.5	2.2	11.5	2.2	1

244 rows × 13 columns

In [177…
```
y1
```

Out[177]:
```
0      1
1      1
2      1
3      1
4      1
      ..
239    0
240    0
241    0
242    0
243    0
Name: Classes, Length: 244, dtype: int32
```

In [178…
```
### For upsampling
import imblearn
from imblearn.combine import SMOTETomek
```

In [179…
```
smt=SMOTETomek()
```

```
smt
```

Out[179]:
```
SMOTETomek()
```

In [180…
```
x_bal,y_bal=smt.fit_resample(x1,y1)
x_bal.head()
```

Out[180]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region
0	14	6	30	78	20.0	0.5	59.0	4.6	7.8	1.0	4.4	0.4	0
1	5	8	34	65	13.0	0.0	86.8	11.1	29.7	5.2	11.5	6.1	0
2	27	8	33	82	21.0	0.0	84.9	47.0	200.2	4.4	59.3	13.2	0
3	5	9	29	75	16.0	0.0	80.8	3.4	24.0	2.8	5.1	1.7	0
4	2	7	27	75	19.0	1.2	55.7	2.4	8.3	0.8	2.8	0.3	0

In [181…
```
y_bal.head()
```

Out[181]:
```
0    1
1    1
2    1
3    1
4    1
Name: Classes, dtype: int32
```

In [182…
```
x_bal.shape, y_bal.shape
```

Out[182]:
```
((426, 13), (426,))
```

Creating Balanced data from imbalanced data

In [183…
```
data_bal=x_bal.join(pd.DataFrame(y_bal))
data_bal.head()
```

Out[183]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Region	Classes
0	14	6	30	78	20.0	0.5	59.0	4.6	7.8	1.0	4.4	0.4	0	1
1	5	8	34	65	13.0	0.0	86.8	11.1	29.7	5.2	11.5	6.1	0	1
2	27	8	33	82	21.0	0.0	84.9	47.0	200.2	4.4	59.3	13.2	0	1
3	5	9	29	75	16.0	0.0	80.8	3.4	24.0	2.8	5.1	1.7	0	1
4	2	7	27	75	19.0	1.2	55.7	2.4	8.3	0.8	2.8	0.3	0	1

Exploratory Data Analaysis on balanced dataset

In [184…
```
data_bal.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 426 entries, 0 to 425
Data columns (total 14 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          426 non-null    int32  
 1   month        426 non-null    int32  
 2   Temperature  426 non-null    int32  
 3   RH           426 non-null    int32  
 4   Ws           426 non-null    float64
 5   Rain         426 non-null    float64
 6   FFMC         426 non-null    float64
 7   DMC          426 non-null    float64
 8   DC           426 non-null    float64
 9   ISI          426 non-null    float64
 10  BUI          426 non-null    float64
 11  FWI          426 non-null    float64
 12  Region       426 non-null    int32  
 13  Classes      426 non-null    int32  
dtypes: float64(8), int32(6)
memory usage: 36.7 KB
```

In [185…
```
data_bal.describe().T
```

Out[185]:

	count	mean	std	min	25%	50%	75%	max
day	426.0	14.781690	8.485101	1.0	8.000000	14.000000	21.000000	31.0
month	426.0	7.309859	1.032528	6.0	7.000000	7.000000	8.000000	9.0
Temperature	426.0	32.373239	3.431225	22.0	30.000000	32.000000	35.000000	42.0
RH	426.0	60.288732	14.102940	21.0	50.000000	62.000000	71.000000	90.0
Ws	426.0	15.360322	2.499838	6.0	13.917993	15.000000	17.000000	29.0
Rain	426.0	0.722659	1.721984	0.0	0.000000	0.000000	0.595842	16.8
FFMC	426.0	78.808667	13.267024	28.6	71.100000	84.600000	88.656776	96.0
DMC	426.0	15.097374	12.158386	0.7	5.651194	11.932237	21.700000	65.9
DC	426.0	45.921786	43.504251	6.9	12.206045	32.599091	67.150000	220.4
ISI	426.0	5.134252	4.292499	0.0	1.425000	4.146053	7.795710	19.0
BUI	426.0	16.628410	13.613676	1.1	5.800000	12.747891	24.125000	68.0
FWI	426.0	7.534845	7.442861	0.0	0.700000	5.550000	13.547627	31.1
Region	426.0	0.537559	0.499174	0.0	0.000000	1.000000	1.000000	1.0
Classes	426.0	0.500000	0.500588	0.0	0.000000	0.500000	1.000000	1.0

In [186…
```
data_bal.corr()
```

Out[186]:

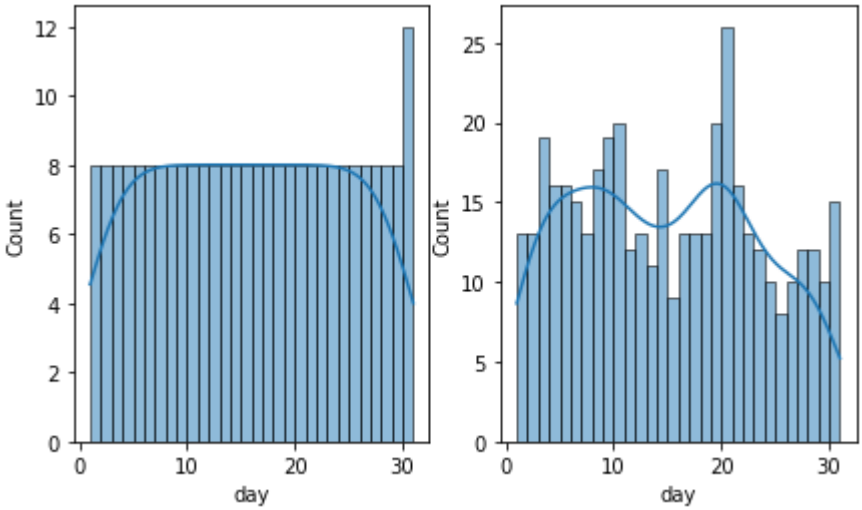
	day	month	Temperature	RH	Ws	Rain	FFMC	
day	1.000000	-0.039260	0.129285	-0.027452	0.013711	-0.203083	0.259873	0.52
month	-0.039260	1.000000	-0.148945	0.094832	-0.072144	0.128725	-0.076403	-0.02
Temperature	0.129285	-0.148945	1.000000	-0.702663	-0.194329	-0.369768	0.707620	0.50
RH	-0.027452	0.094832	-0.702663	1.000000	0.116871	0.232448	-0.647288	-0.40
Ws	0.013711	-0.072144	-0.194329	0.116871	1.000000	0.066955	-0.014755	0.10
Rain	-0.203083	0.128725	-0.369768	0.232448	0.066955	1.000000	-0.587217	-0.34
FFMC	0.259873	-0.076403	0.707620	-0.647288	-0.014755	-0.587217	1.000000	0.65
DMC	0.524441	-0.020152	0.509020	-0.400904	0.108789	-0.345135	0.654142	1.00
DC	0.578618	0.094313	0.368880	-0.186408	0.129471	-0.330467	0.530587	0.86
ISI	0.163053	-0.084769	0.637503	-0.721426	0.187971	-0.390822	0.771070	0.69
BUI	0.560115	0.013849	0.477850	-0.340209	0.122513	-0.350150	0.636625	0.98
FWI	0.355736	-0.045694	0.600112	-0.608776	0.180322	-0.376885	0.739576	0.88
Region	-0.010559	-0.072850	0.389499	-0.470306	-0.065476	-0.104785	0.335068	0.27
Classes	0.102758	0.191196	-0.123974	0.175143	0.092606	0.029954	-0.116010	-0.08

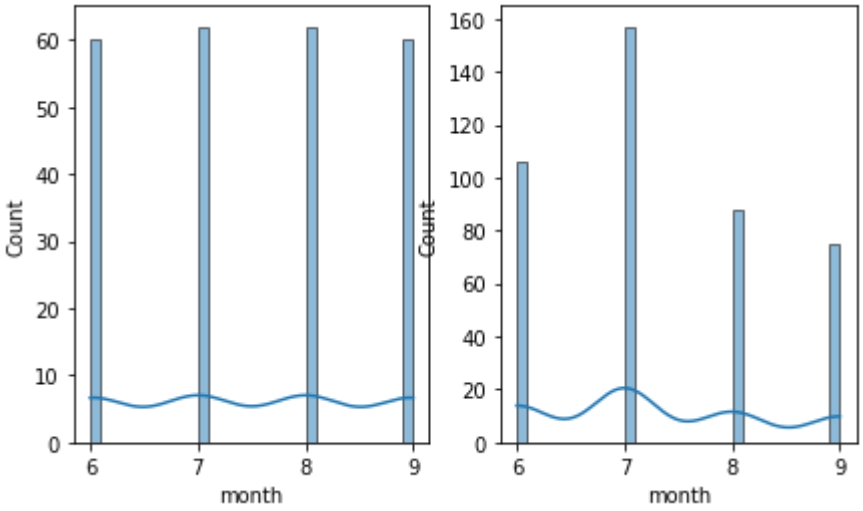
In [187…
```
num_bal_col=[feature for feature in data_bal.columns if data_bal[feature].dtype != 'O'
num_bal_col
```

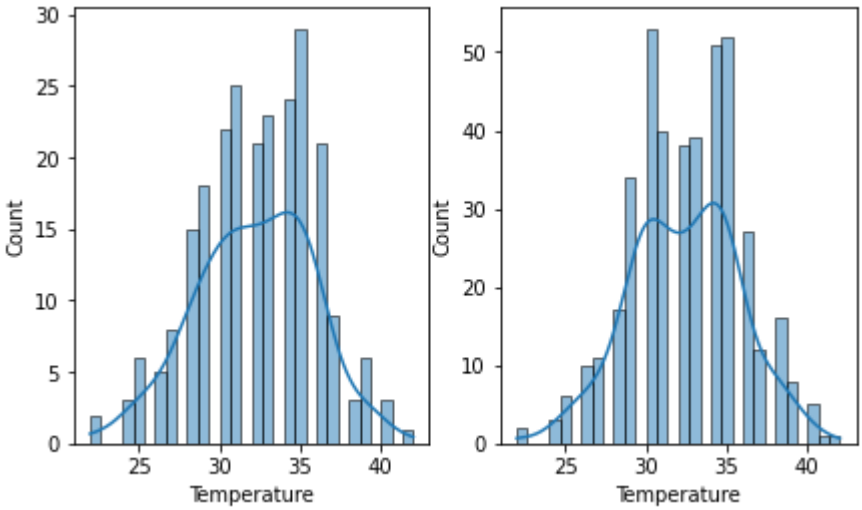
Out[187]:
```
['day',
 'month',
 'Temperature',
 'RH',
 'Ws',
 'Rain',
 'FFMC',
 'DMC',
 'DC',
 'ISI',
 'BUI',
 'FWI',
 'Region',
 'Classes']
```

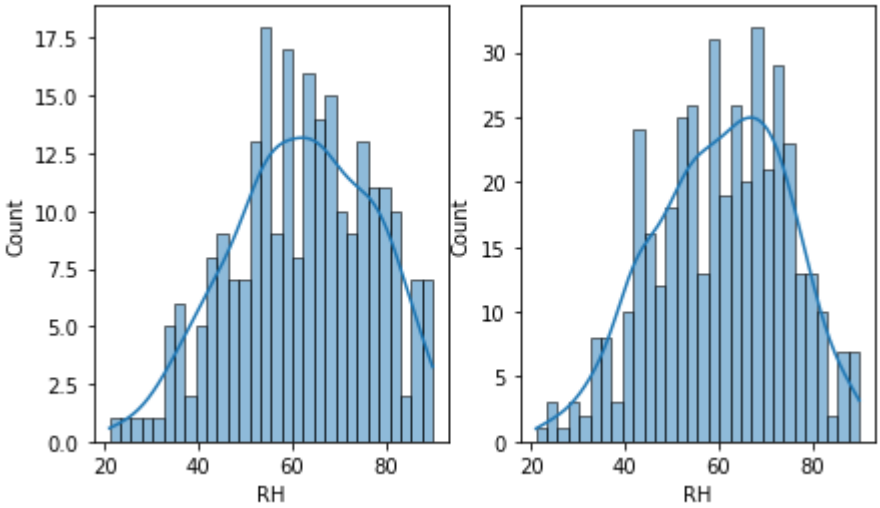
Dataset - original vs balanced

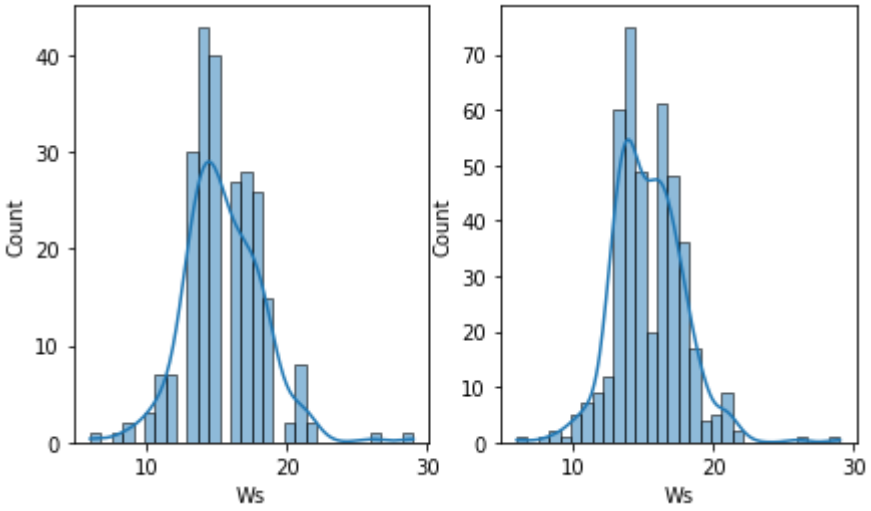
In [216…
```
for i in num_df:
    plt.figure(figsize=(7,4))
    plt.subplot(121)
    sns.histplot(data=df_copy,x=i,kde=True,bins=30)

    plt.subplot(122)
    sns.histplot(data=data_bal,x=i,kde=True,bins=30)
```

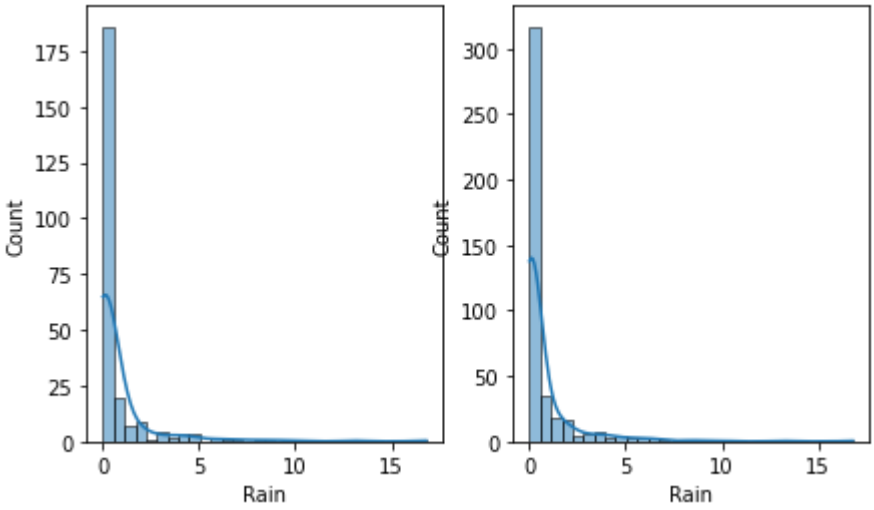

Count
day
Count
day

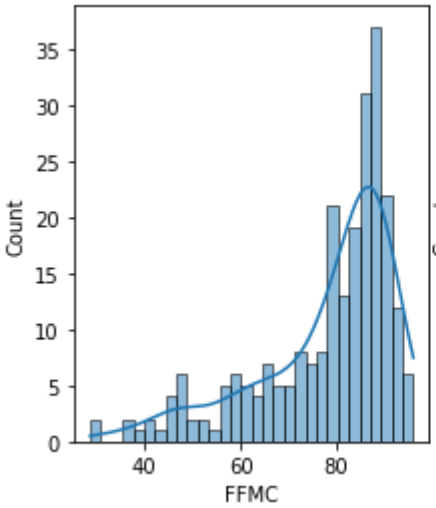

Count
month
Count
month

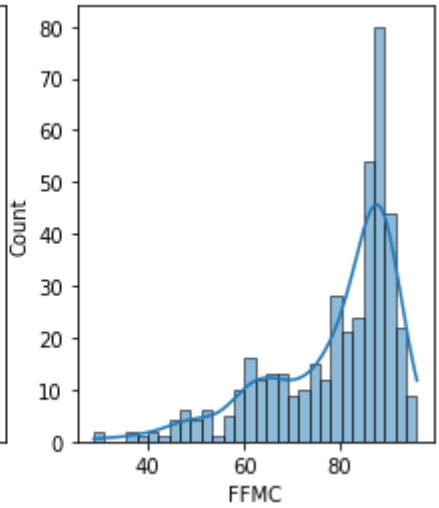

Count
Temperature
Count
Temperature

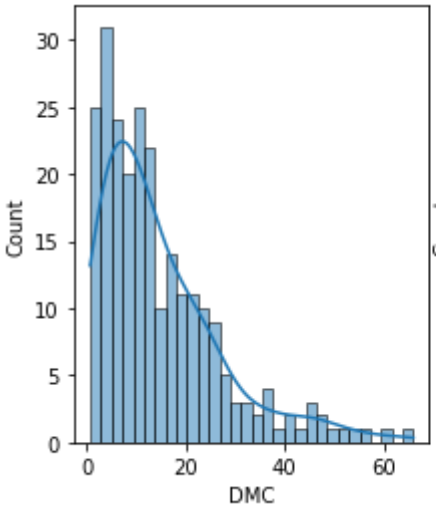

Count
RH
Count
RH

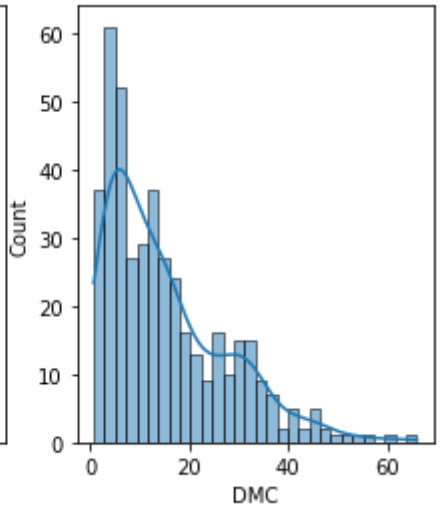

Count
Ws
Count
Ws

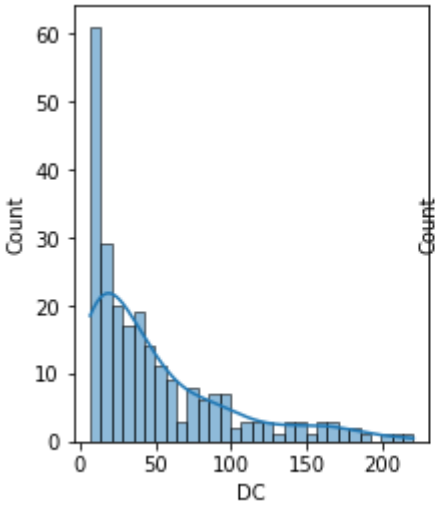

Count
Rain
Count
Rain

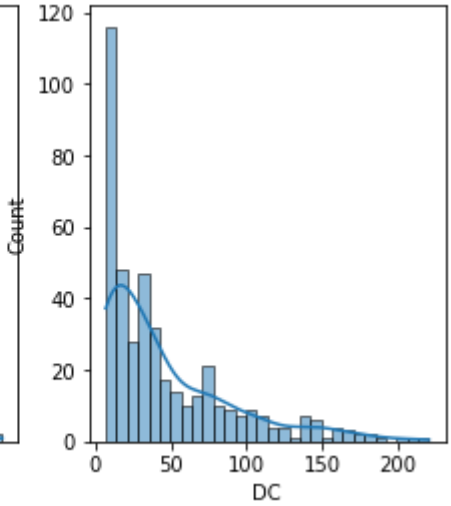

Count
FFMC

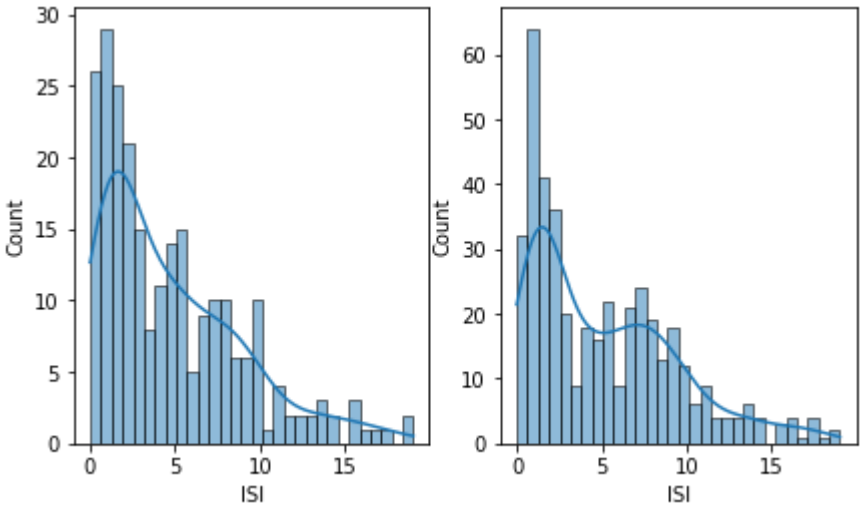

Count
FFMC

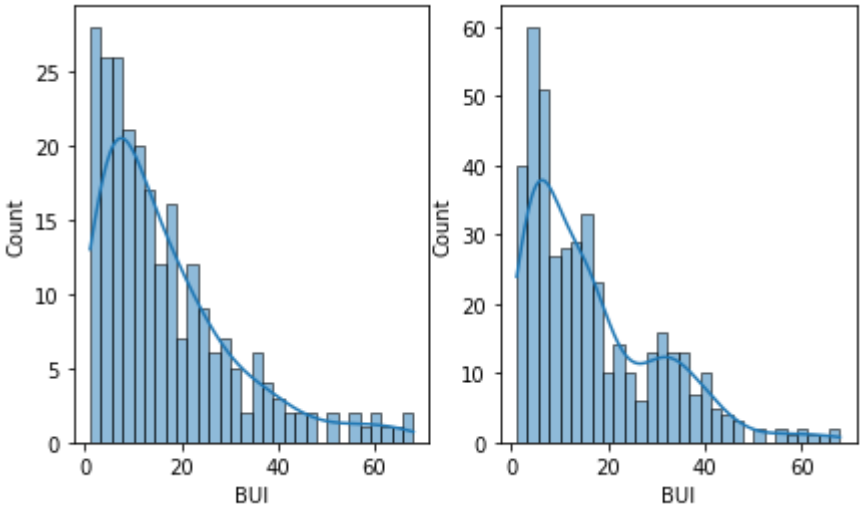

Count
DMC

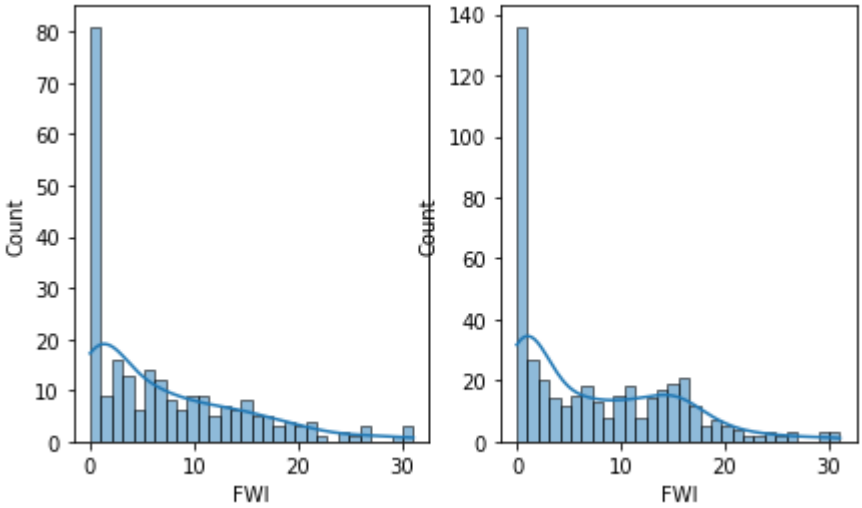

Count
DMC

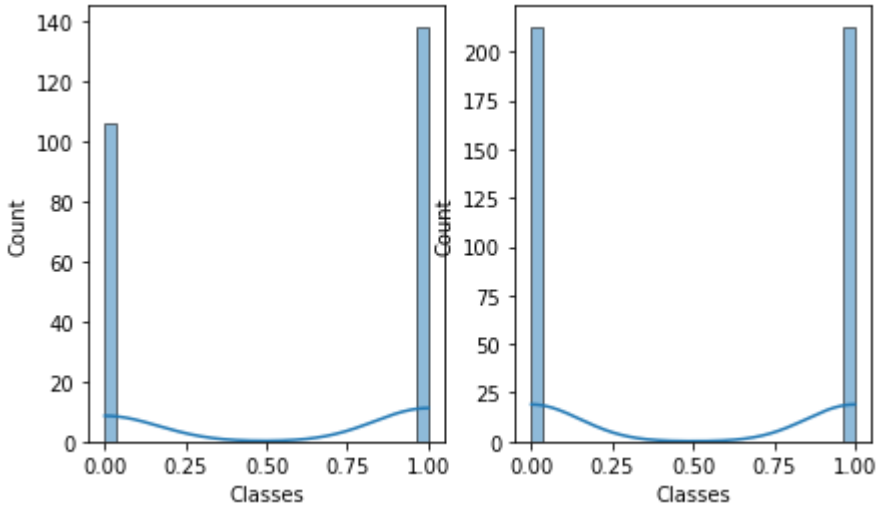

Count
DC

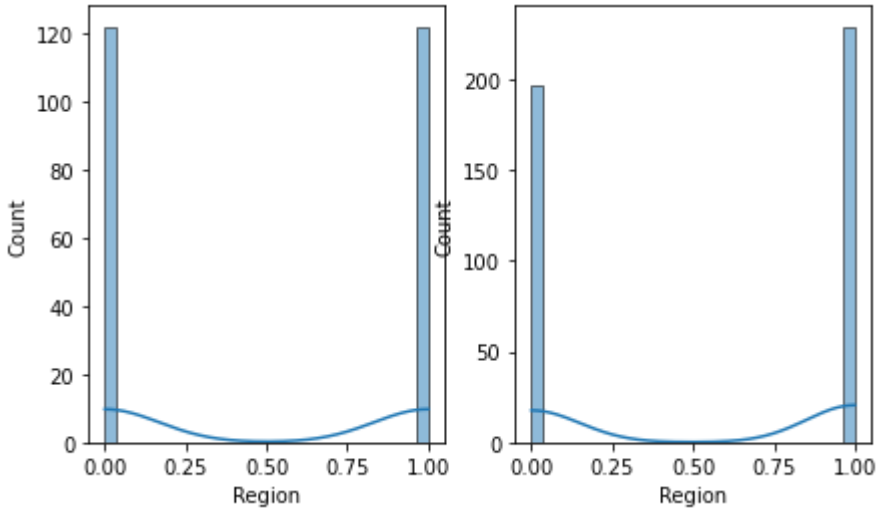

Count
DC


Count
ISI
Count
ISI


Count
BUI
Count
BUI


Count
FWI
Count
FWI

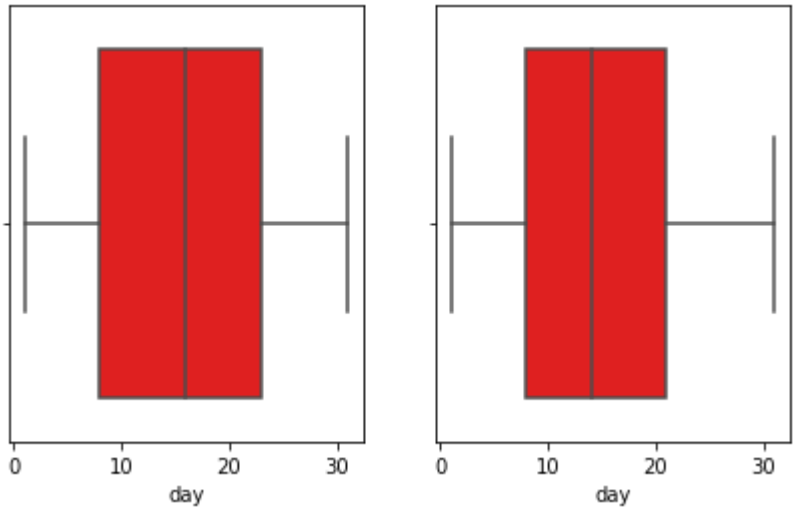


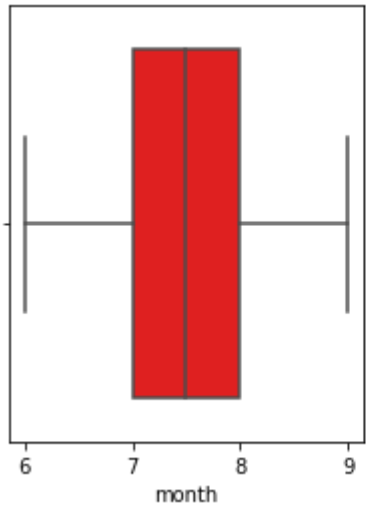


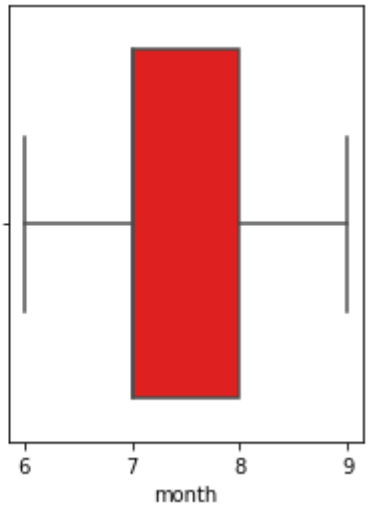
Outliers - original vs balanced

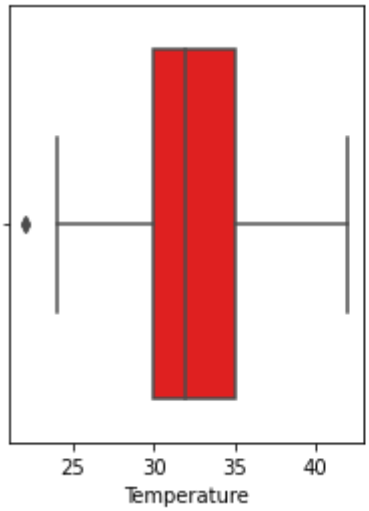
In [218…

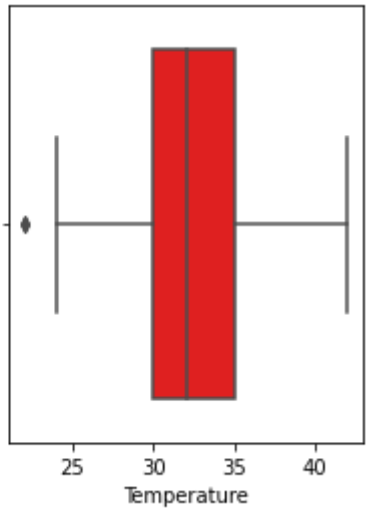
```
for i in num_df:
    plt.figure(figsize=(7,4))
    plt.subplot(121)
    sns.boxplot(data=df_copy,x=i,color='r')


    plt.subplot(122)
    sns.boxplot(data=data_bal,x=i,color='r')
```

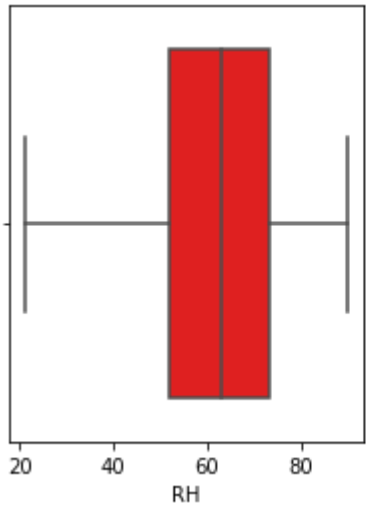


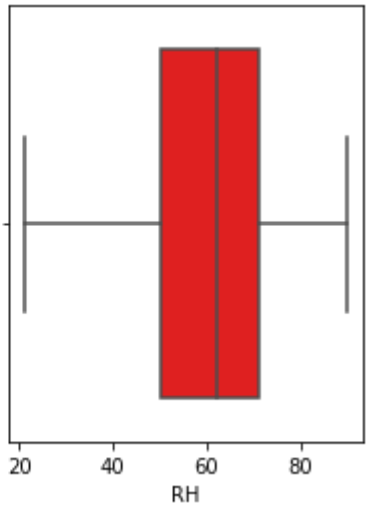

6
7
8
9
month

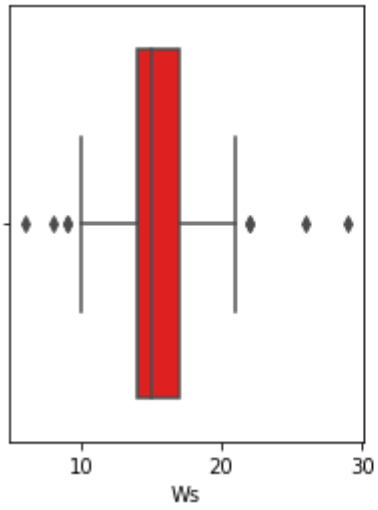

6
7
8
9
month

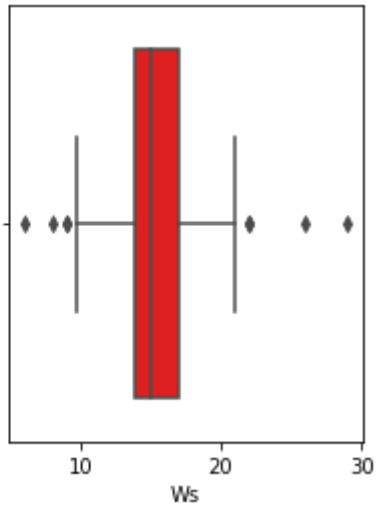

25
30
35
40
Temperature

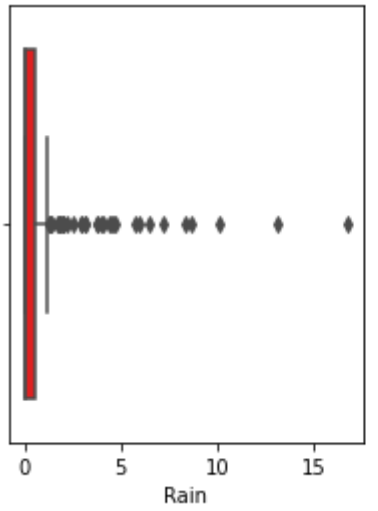

25
30
35
40
Temperature

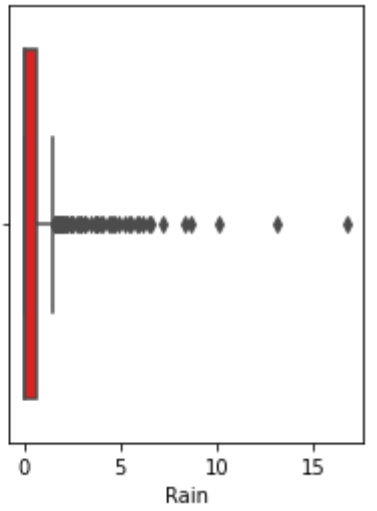

20
40
60
80
RH

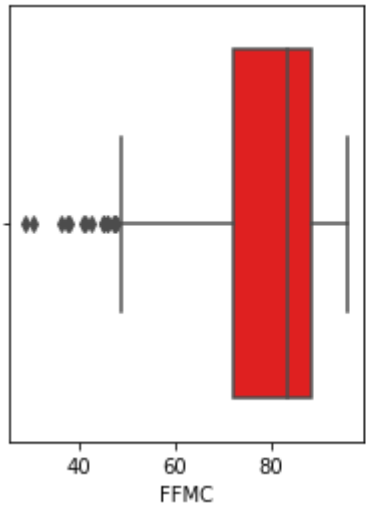

20
40
60
80
RH

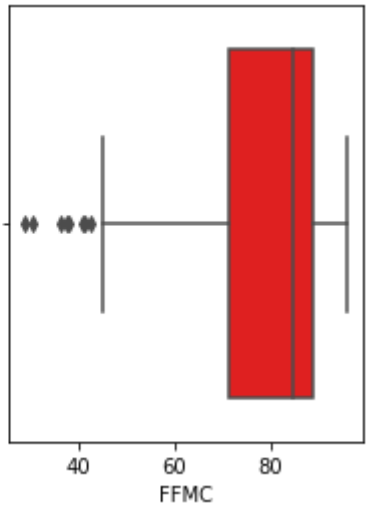

10
20
30
Ws

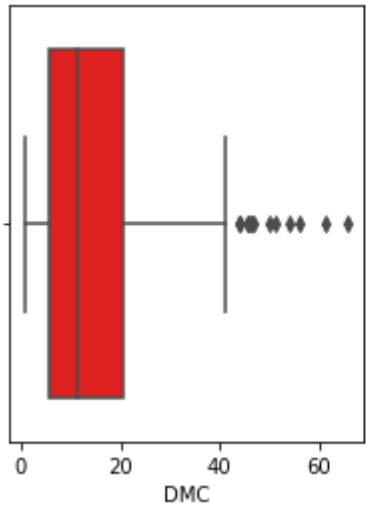

10
20
30
Ws

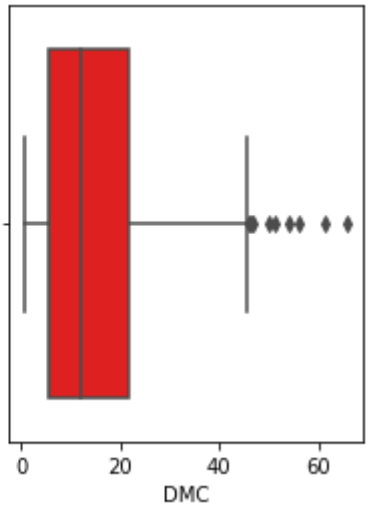

0
5
10
15
Rain

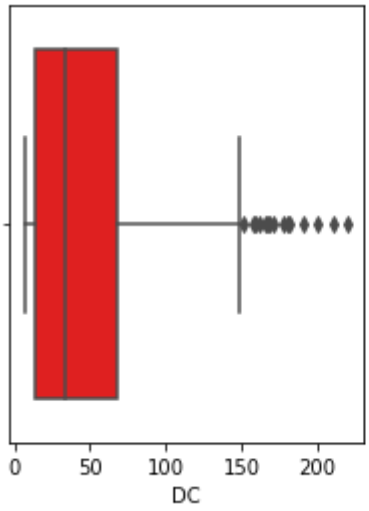

0
5
10
15
Rain

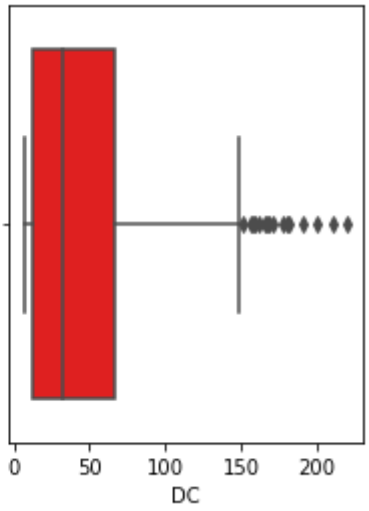

40
60
80
FFMC

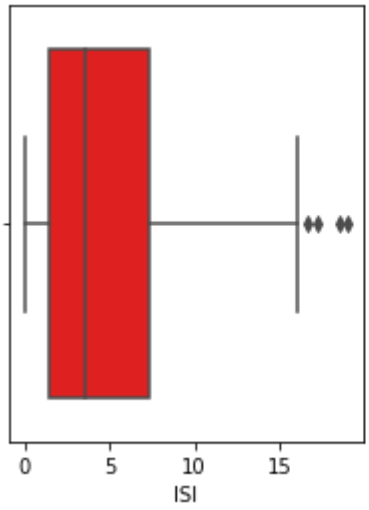

40
60
80
FFMC

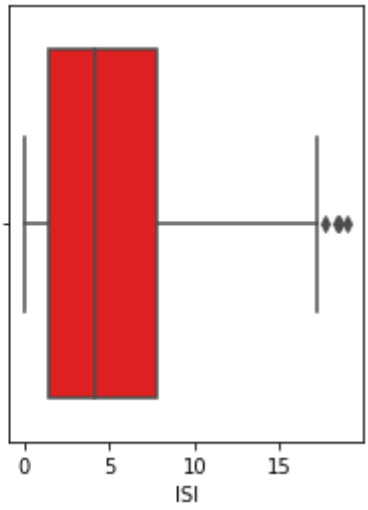

0
20
40
60
DMC

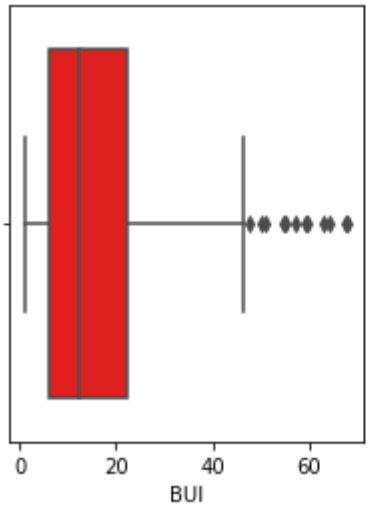

0
20
40
60
DMC

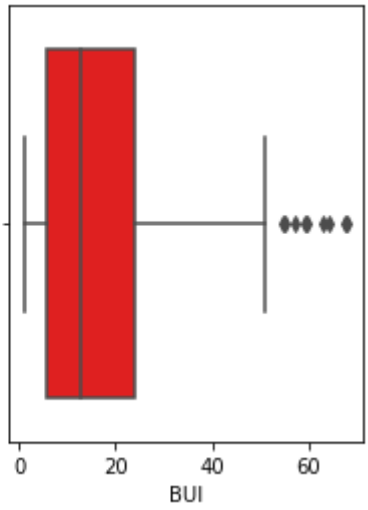

0
50
100
150
200
DC

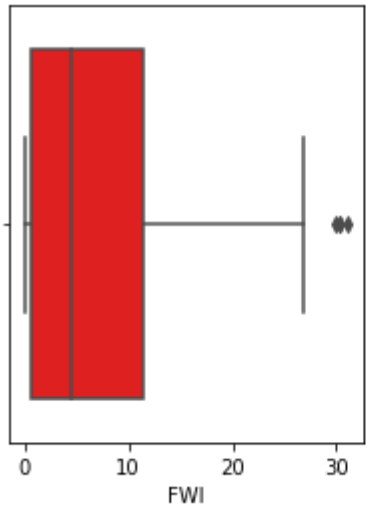

0
50
100
150
200
DC

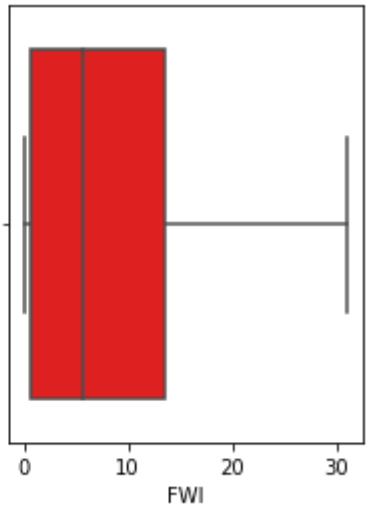

0
5
10
15
ISI

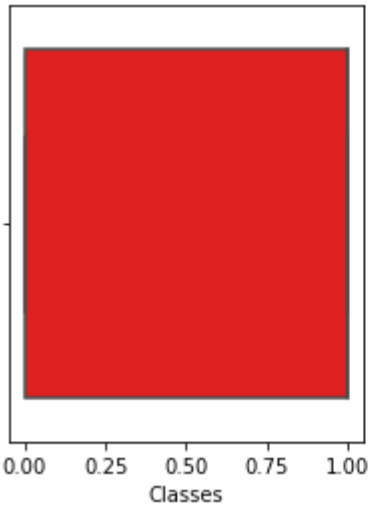

0
5
10
15
ISI

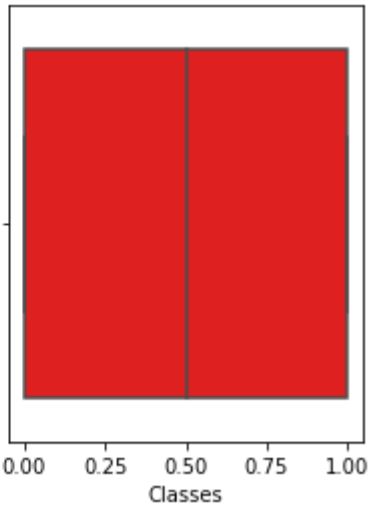

0
20
40
60
BUI

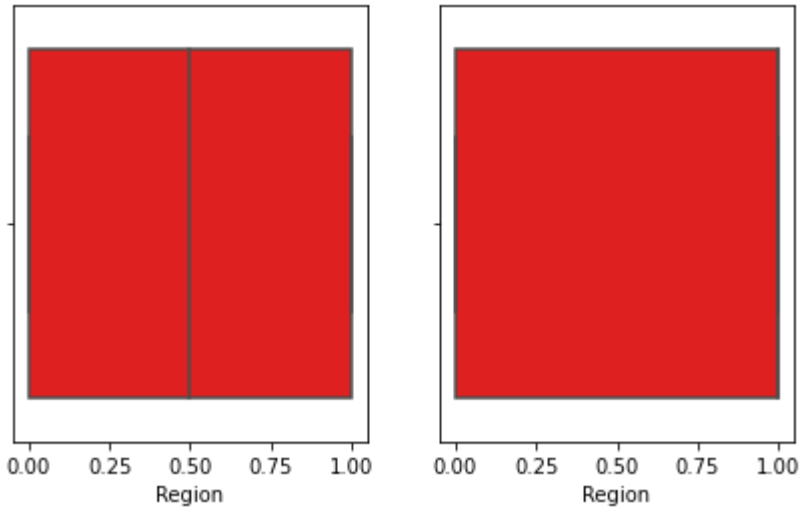

0
20
40
60
BUI


0
10
20
30
FWI


0
10
20
30
FWI


0.00
0.25
0.50
0.75
1.00
Classes


0.00
0.25
0.50
0.75
1.00
Classes



Train test split

In [190…
```
from sklearn.model_selection import train_test_split
x_train1,x_test1,y_train1,y_test1=train_test_split(x_bal,y_bal,test_size=0.30,random_s
```

In [191…
```
x_train1
```

Out[191]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	
3	5	9	29	75	16.000000	0.000000	80.800000	3.400000	24.000000	2.8
92	27	7	36	48	13.000000	0.000000	90.300000	22.200000	108.500000	8.7
12	8	8	37	56	11.000000	0.000000	87.400000	11.200000	20.200000	5.2
149	15	7	30	80	19.000000	0.400000	60.700000	5.200000	17.000000	1.1
7	8	7	33	68	19.000000	0.000000	85.600000	12.500000	49.800000	6.0
...	...	...	...	...	...	...	...	...	...	...
321	19	6	30	72	14.059384	0.198021	60.362317	3.762390	8.152420	0.8
69	18	8	36	54	18.000000	0.000000	89.400000	20.000000	110.900000	9.7
121	13	8	35	34	16.000000	0.200000	88.300000	16.900000	45.100000	7.5
238	3	7	38	33	17.094732	0.000000	93.626583	17.126051	32.062107	17.1
169	26	7	35	58	10.000000	0.200000	78.300000	10.800000	19.700000	1.6

298 rows × 13 columns

In [192…
```
x_test1
```

Out[192]:

	day	month	Temperature	RH	Ws	Rain	FFMC	DMC	DC	
103	20	8	35	68	19.000000	0.000000	88.300000	25.900000	130.600000	8.8
374	5	9	29	73	16.327981	0.077599	70.819267	2.930428	15.901606	1.5
325	11	6	31	43	20.493039	0.000000	90.346519	19.173366	33.602603	12.7
322	14	7	30	74	14.000000	0.000000	79.858084	6.426880	22.084030	2.2
133	19	9	29	41	8.000000	0.100000	83.900000	24.900000	86.000000	2.7
...	...	...	...	...	...	...	...	...	...	
55	13	7	34	81	15.000000	0.000000	81.800000	9.700000	37.200000	3.0
25	27	8	36	54	14.000000	0.000000	91.000000	65.900000	177.300000	10.0
168	22	7	32	48	18.000000	0.000000	91.500000	44.200000	90.100000	13.2
194	16	7	31	83	17.000000	0.000000	84.500000	19.400000	33.100000	4.7
386	6	7	37	45	13.914151	0.408585	85.548491	13.109698	35.655962	4.8

128 rows × 13 columns

In [193…
```
y_train1
```

Out[193]:
```
3      1
92     1
12     1
149    1
7      1
      ..
321    0
69     1
121    1
238    0
169    1
Name: Classes, Length: 298, dtype: int32
```

In [194…
```
y_test1
```

Out[194]:
```
103    1
374    0
325    0
322    0
133    1
      ..
55     1
25     1
168    1
194    1
386    0
Name: Classes, Length: 128, dtype: int32
```

logistic regression model

In [195…
```
from sklearn.linear_model import LogisticRegression
classifier_bal=LogisticRegression()
classifier_bal
```

Out[195]:
```
LogisticRegression()
```

In [196…
```
from sklearn.model_selection import GridSearchCV
parameter_bal={'penalty':['l1','l2','elasticnet'],'C':[1,2,3,4,5,6,10,20,30,40,50],'ma
```

In [197…
```
classifier_regressor_bal=GridSearchCV(classifier,param_grid=parameter,scoring='accurac
```

Standardization

In [198…
```
classifier_regressor_bal.fit(x_train1,y_train1)
```

Out[198]:
```
GridSearchCV
estimator: LogisticRegression
LogisticRegression
```

In [199…
```
print(classifier_regressor_bal.best_params_)
```

```
{'C': 2, 'max_iter': 300, 'penalty': 'l2'}
```

In [200…
```
print(classifier_regressor_bal.best_score_)
```

```
0.5743502824858757
```

Prediction

In [201…
```
y_bal_pred = classifier_regressor_bal.predict(x_test1)
y_bal_pred
```

Out[201]:
```
array([1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0,
       0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0])
```

Accuracy

In [202…
```
from sklearn.metrics import accuracy_score,classification_report
bal_score=accuracy_score(y_bal_pred,y_test1)
print(bal_score)
```

```
0.640625
```

Classification Report

In [203…
```
print(classification_report(y_bal_pred,y_test1))
```

	precision	recall	f1-score	support
0	0.56	0.75	0.64	55
1	0.75	0.56	0.64	73
accuracy			0.64	128
macro avg	0.65	0.65	0.64	128
weighted avg	0.67	0.64	0.64	128

Performance Metrics

Confusion Metrics

In [204…
```
conf_mat_bal=confusion_matrix(y_bal_pred,y_test1)
conf_mat_bal
```

Out[204]:
```
array([[41, 14],
       [32, 41]], dtype=int64)
```

In [205…
```
true_positive = conf_mat_bal[0][0]
false_positive = conf_mat_bal[0][1]
false_negative = conf_mat_bal[1][0]
true_negative = conf_mat_bal[1][1]
```

Precision

In [206…
```
bal_Precision = true_positive/(true_positive+false_positive)
bal_Precision
```

Out[206]:
```
0.7454545454545455
```

Recall

In [207…
```
bal_recall = true_positive/(true_positive+false_negative)
bal_recall
```

Out[207]:
```
0.5616438356164384
```

F1 Score

In [208…
```
F1_Score_bal = 2*(bal_recall * bal_Precision) / (bal_recall + bal_Precision)
F1_Score_bal
```

Out[208]:
```
0.640625
```

Conclusion

Performance of Logistic Regression on Original Dataset

In [209…
```
print(classification_report(y_pred,y_test))
```

	precision	recall	f1-score	support
0	1.00	0.96	0.98	25
1	0.97	1.00	0.99	36
accuracy			0.98	61
macro avg	0.99	0.98	0.98	61
weighted avg	0.98	0.98	0.98	61

Perfromance of Logistic Regression on Balanced Dataset

In [210…
```
print(classification_report(y_bal_pred,y_test1))
```

	precision	recall	f1-score	support
0	0.56	0.75	0.64	55
1	0.75	0.56	0.64	73
accuracy			0.64	128
macro avg	0.65	0.65	0.64	128
weighted avg	0.67	0.64	0.64	128

Observation

- **Original dataset model predicts good results.**
- **Balanced dataset (created from imbalanced dataset) model predicts bad results.**

In [ ]:
```
```