Data Set Information:

The dataset includes 244 instances that regroup a data of two regions of Algeria,namely the Bejaia region located in the northeast of Algeria and the Sidi Bel-abbes region located in the northwest of Algeria.

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 fire (138 classes) and not fire (106 classes) classes.

Attribute Information:

1. **Date : (DD/MM/YYYY) Day, month ('june' to 'september'), year (2012) #### Weather data observations**
2. **Temp : temperature noon (temperature max) in Celsius degrees: 22 to 42**
3. **RH : Relative Humidity in %: 21 to 90**
4. **Ws :Wind speed in km/h: 6 to 29**
5. **Rain: total day in mm: 0 to 16.8 #### FWI Components**
6. **Fine Fuel Moisture Code (FFMC) index from the FWI system: 28.6 to 92.5**
7. **Duff Moisture Code (DMC) index from the FWI system: 1.1 to 65.9**
8. **Drought Code (DC) index from the FWI system: 7 to 220.4**
9. **Initial Spread Index (ISI) index from the FWI system: 0 to 18.5**
10. **Buildup Index (BUI) index from the FWI system: 1.1 to 68**
11. **Fire Weather Index (FWI) Index: 0 to 31.1**
12. **Classes: two classes, namely â€œFireâ€ and â€œnot Fireâ€**

In [1]:

```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
import warnings

warnings.filterwarnings("ignore")
%matplotlib inline
```

In [2]:

```
# Import the Algerian Forest Fire Dataset
# Link: https://archive.ics.uci.edu/ml/datasets/Algerian+Forest+Fires+Dataset++#

forest_data = pd.read_csv("Algerian_forest_fires_dataset_UPDATE.csv",header=1)
```

In [3]:

```
# Cloning the original dataset
df = forest_data.copy()
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

Data Cleaning

In [4]:

```
# Checking the null values
df.isnull().sum()
```

Out[4]:

```
day            0
month          1
year           1
Temperature    1
 RH            1
 Ws            1
Rain           1
FFMC           1
DMC            1
DC             1
ISI            1
BUI            1
FWI            1
Classes        2
dtype: int64
```

In [5]:

```
# Categorizing the dataset into two halves for two regions for better visulization
df.loc[:122,'Region']=1
df.loc[122:,'Region']=2
df[['Region']] = df[['Region']].astype(int)
```

In [6]:

```
df.info() # Checking the datatypes
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 246 entries, 0 to 245
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   day          246 non-null    object
 1   month        245 non-null    object
 2   year         245 non-null    object
 3   Temperature  245 non-null    object
 4    RH          245 non-null    object
 5    Ws          245 non-null    object
 6   Rain         245 non-null    object
 7   FFMC         245 non-null    object
 8   DMC          245 non-null    object
 9   DC           245 non-null    object
 10  ISI          245 non-null    object
 11  BUI          245 non-null    object
 12  FWI          245 non-null    object
 13  Classes      244 non-null    object
 14  Region       246 non-null    int32 
dtypes: int32(1), object(14)
memory usage: 28.0+ KB
```

It is showing almost every column as categorical column though it has numerical values, so we'll perform few operations to classify into Numerical and Categorical Features.

In [7]:

```
df.shape
```

Out[7]:

```
(246, 15)
```

In [8]:

```
df[df.isnull().any(axis=1)] #Checking the row containing the NaN values
```

Out[8]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
122	**Sidi-Bel Abbes Region Dataset**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**NaN**	**2**
167	**14**	**07**	**2012**	**37**	**37**	**18**	**0.2**	**88.9**	**12.9**	**14.69**	**12.5**	**10.4**	**fire**	**NaN**	**2**

The Missing values are found at 122th index.Thus, we need to seprate the dataset into two regions.

1 : Bejaia Region Dataset

2 : Sidi-Bel Abbes Region Dataset

In [9]:

```
df.loc[:122,'Region']=1
df.loc[122:,'Region']=2
df[['Region']] = df[['Region']].astype(int)
```

In [10]:

```
df.head()
```

Out[10]:

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

In [11]:

```
df.tail()
```

Out[11]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
241	**26**	**09**	**2012**	**30**	**65**	**14**	**0**	**85.4**	**16**	**44.5**	**4.5**	**16.9**	**6.5**	**fire**	**2**
242	**27**	**09**	**2012**	**28**	**87**	**15**	**4.4**	**41.1**	**6.5**	**8**	**0.1**	**6.2**	**0**	**not fire**	**2**
243	**28**	**09**	**2012**	**27**	**87**	**29**	**0.5**	**45.9**	**3.5**	**7.9**	**0.4**	**3.4**	**0.2**	**not fire**	**2**
244	**29**	**09**	**2012**	**24**	**54**	**18**	**0.1**	**79.7**	**4.3**	**15.2**	**1.7**	**5.1**	**0.7**	**not fire**	**2**
245	**30**	**09**	**2012**	**24**	**64**	**15**	**0.2**	**67.3**	**3.8**	**16.5**	**1.2**	**4.8**	**0.5**	**not fire**	**2**

In [12]:

```
df.isnull().sum()
```

Out[12]:

```
day            0
month          1
year           1
Temperature    1
 RH            1
```

```
 Ws             1
Rain            1
FFMC            1
DMC             1
DC              1
ISI             1
BUI             1
FWI             1
Classes         2
Region          0
dtype: int64
```

In [13]:

```
df =df.dropna().reset_index(drop=True)
df.shape
```

Out[13]:

```
(244, 15)
```

In [14]:

```
# Column which has string
df.iloc[[122]]
```

Out[14]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
122	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	2

In [15]:

```
df[df.duplicated()]
```

Out[15]:

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

In [16]:

```
#removing 122th column
df1 = df.drop(122).reset_index(drop=True)
df1.head()
```

Out[16]:

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

In [17]:

```
df1.shape
```

Out[17]:

```
(243, 15)
```

In [18]:

```
df1[df1.isnull().any(axis=1)]
```

Out[18]:

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

In [19]:

```
df1.columns
```

Out[19]:

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

In [20]:

```
# Spaces were fixed in the column names
df1.columns = df1.columns.str.strip()
df1.columns
```

Out[20]:

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

Converting the categorical values into int type

In [21]:

```
df1[['month', 'day', 'year', 'Temperature','RH', 'Ws']] = df1[['month', 'day', 'year', '
Temperature','RH', 'Ws']].astype(int)
```

In [22]:

```
# define numerical & categorical columns
numeric_features = [feature for feature in df1.columns if df1[feature].dtype != 'O']
categorical_features = [feature for feature in df1.columns if df1[feature].dtype == 'O']

# print columns
print('We have {} numerical features : {}'.format(len(numeric_features), numeric_feature
s))
print('\nWe have {} categorical features : {}'.format(len(categorical_features), categor
ical_features))
```

```
We have 7 numerical features : ['day', 'month', 'year', 'Temperature', 'RH', 'Ws', 'Regio
n']

We have 8 categorical features : ['Rain', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'Clas
ses']
```

In [23]:

```
# proportion of count data on categorical columns
for col in categorical_features:
    print(df1[col].value_counts(normalize=True) * 100)
    print('----------------------------')
```

```
0       54.732510
0.1      7.407407
0.2      4.526749
0.3      4.115226
0.4      3.292181
0.7      2.469136
0.6      2.469136
0.5      2.057613
1.1      1.234568
1.2      1.234568
2        1.234568
1.8      1.234568
0.8      0.823045
```

```
2.9      0.823045
1.3      0.823045
3.8      0.823045
1.4      0.823045
1        0.823045
3.1      0.823045
16.8     0.411523
4.5      0.411523
6.5      0.411523
4.1      0.411523
13.1     0.411523
1.9      0.411523
6        0.411523
2.2      0.411523
1.7      0.411523
2.5      0.411523
4.7      0.411523
8.7      0.411523
7.2      0.411523
4        0.411523
5.8      0.411523
8.3      0.411523
4.6      0.411523
0.9      0.411523
10.1     0.411523
4.4      0.411523
Name: Rain, dtype: float64
---------------------------
88.9    2.880658
89.4    2.057613
89.3    1.646091
85.4    1.646091
89.1    1.646091
          ...
82.4    0.411523
81.3    0.411523
86.4    0.411523
76.6    0.411523
67.3    0.411523
Name: FFMC, Length: 173, dtype: float64
---------------------------
7.9     2.057613
12.5    1.646091
1.9     1.646091
3.4     1.234568
4.6     1.234568
          ...
29.6    0.411523
25.9    0.411523
23.2    0.411523
20      0.411523
4.3     0.411523
Name: DMC, Length: 165, dtype: float64
---------------------------
8        2.057613
7.6      1.646091
7.8      1.646091
8.4      1.646091
7.5      1.646091
           ...
92.5     0.411523
90.4     0.411523
100.7    0.411523
110.9    0.411523
16.5     0.411523
Name: DC, Length: 197, dtype: float64
---------------------------
1.1     3.292181
1.2     2.880658
0.4     2.057613
4.7     2.057613
5.2     2.057613
```

```
          ...
11.7    0.411523
11.3    0.411523
7.6     0.411523
8.8     0.411523
11.2    0.411523
Name: ISI, Length: 106, dtype: float64
---------------------------
3       2.057613
5.1     1.646091
8.3     1.234568
2.9     1.234568
11.5    1.234568
          ...
67.4    0.411523
62.9    0.411523
59.3    0.411523
57.1    0.411523
4.8     0.411523
Name: BUI, Length: 173, dtype: float64
---------------------------
0.4    4.938272
0.8    4.115226
0.5    3.703704
0.1    3.703704
0      3.703704
          ...
7.5    0.411523
8.3    0.411523
9.7    0.411523
8.4    0.411523
6.5    0.411523
Name: FWI, Length: 126, dtype: float64
---------------------------
fire             53.909465
not fire         41.563786
fire              1.646091
fire              0.823045
not fire          0.823045
not fire          0.411523
not fire          0.411523
not fire          0.411523
Name: Classes, dtype: float64
---------------------------
```

Data set is balanced i.e, equally divided into its categories

In [24]:

```
# Changing the existing data types into the required data types for the some features for better EDA
objects = [features for features in df1.columns if df1[features].dtypes=='O']
for i in objects:
    if i != 'Classes':
        df1[i] = df1[i].astype(float)
```

In [25]:

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 243 entries, 0 to 242
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   day          243 non-null    int32
 1   month        243 non-null    int32
 2   year         243 non-null    int32
 3   Temperature  243 non-null    int32
 4   RH           243 non-null    int32
```

```
 5   ws            243 non-null    int32
 6   Rain          243 non-null    float64
 7   FFMC          243 non-null    float64
 8   DMC           243 non-null    float64
 9   DC            243 non-null    float64
 10  ISI           243 non-null    float64
 11  BUI           243 non-null    float64
 12  FWI           243 non-null    float64
 13  Classes       243 non-null    object
 14  Region        243 non-null    int32
dtypes: float64(7), int32(7), object(1)
memory usage: 22.0+ KB
```

In [26]:

```
df1.describe()
```

Out[26]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC
count	243.000000	243.000000	243.0	243.000000	243.000000	243.000000	243.000000	243.000000	243.000000	243.000000
mean	15.761317	7.502058	2012.0	32.152263	62.041152	15.493827	0.762963	77.842387	14.680658	49.430864
std	8.842552	1.114793	0.0	3.628039	14.828160	2.811385	2.003207	14.349641	12.393040	47.665606
min	1.000000	6.000000	2012.0	22.000000	21.000000	6.000000	0.000000	28.600000	0.700000	6.900000
25%	8.000000	7.000000	2012.0	30.000000	52.500000	14.000000	0.000000	71.850000	5.800000	12.350000
50%	16.000000	8.000000	2012.0	32.000000	63.000000	15.000000	0.000000	83.300000	11.300000	33.100000
75%	23.000000	8.000000	2012.0	35.000000	73.500000	17.000000	0.500000	88.300000	20.800000	69.100000
max	31.000000	9.000000	2012.0	42.000000	90.000000	29.000000	16.800000	96.000000	65.900000	220.400000

In [27]:

```
df1.describe(include = 'all')
```

Out[27]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC
count	243.000000	243.000000	243.0	243.000000	243.000000	243.000000	243.000000	243.000000	243.000000	243.000000
unique	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	15.761317	7.502058	2012.0	32.152263	62.041152	15.493827	0.762963	77.842387	14.680658	49.430864
std	8.842552	1.114793	0.0	3.628039	14.828160	2.811385	2.003207	14.349641	12.393040	47.665606
min	1.000000	6.000000	2012.0	22.000000	21.000000	6.000000	0.000000	28.600000	0.700000	6.900000
25%	8.000000	7.000000	2012.0	30.000000	52.500000	14.000000	0.000000	71.850000	5.800000	12.350000
50%	16.000000	8.000000	2012.0	32.000000	63.000000	15.000000	0.000000	83.300000	11.300000	33.100000
75%	23.000000	8.000000	2012.0	35.000000	73.500000	17.000000	0.500000	88.300000	20.800000	69.100000
max	31.000000	9.000000	2012.0	42.000000	90.000000	29.000000	16.800000	96.000000	65.900000	220.400000

In [28]:

```
df1["Classes"].value_counts()
```

Out[28]:

```
fire              131
not fire          101
fire                4
fire                2
```

```
fire            2
not fire        2
not fire        1
not fire        1
not fire        1
Name: Classes, dtype: int64
```

The dependent variables (Classes) contains only two categories (fire & not fire), due to unwanted spacing, it is showing as many categories, so we need to reduce the spacings to get the correct values

In [29]:

```
df1.Classes = df1.Classes.str.strip()
```

In [30]:

```
df1["Classes"].value_counts()
```

Out[30]:

```
fire        137
not fire    106
Name: Classes, dtype: int64
```

Here we got the correct value counts of our dependent variables

In [31]:

```
df1[:122].head()
```

Out[31]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	2012	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	1
1	2	6	2012	29	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	not fire	1
2	3	6	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	1
3	4	6	2012	25	89	13	2.5	28.6	1.3	6.9	0.0	1.7	0.0	not fire	1
4	5	6	2012	27	77	16	0.0	64.8	3.0	14.2	1.2	3.9	0.5	not fire	1

In [32]:

```
df1[122:].head()
```

Out[32]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
122	1	6	2012	32	71	12	0.7	57.1	2.5	8.2	0.6	2.8	0.2	not fire	2
123	2	6	2012	30	73	13	4.0	55.7	2.7	7.8	0.6	2.9	0.2	not fire	2
124	3	6	2012	29	80	14	2.0	48.7	2.2	7.6	0.3	2.6	0.1	not fire	2
125	4	6	2012	30	64	14	0.0	79.4	5.2	15.4	2.2	5.6	1.0	not fire	2
126	5	6	2012	32	60	14	0.2	77.1	6.0	17.6	1.8	6.5	0.9	not fire	2

In [33]:

```
df1.corr()
```

Out[33]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	
day	1.000000	-0.000369	NaN	0.097227	-0.076034	0.047812	-0.112523	0.224956	0.491514	0.527952	0.180543	0

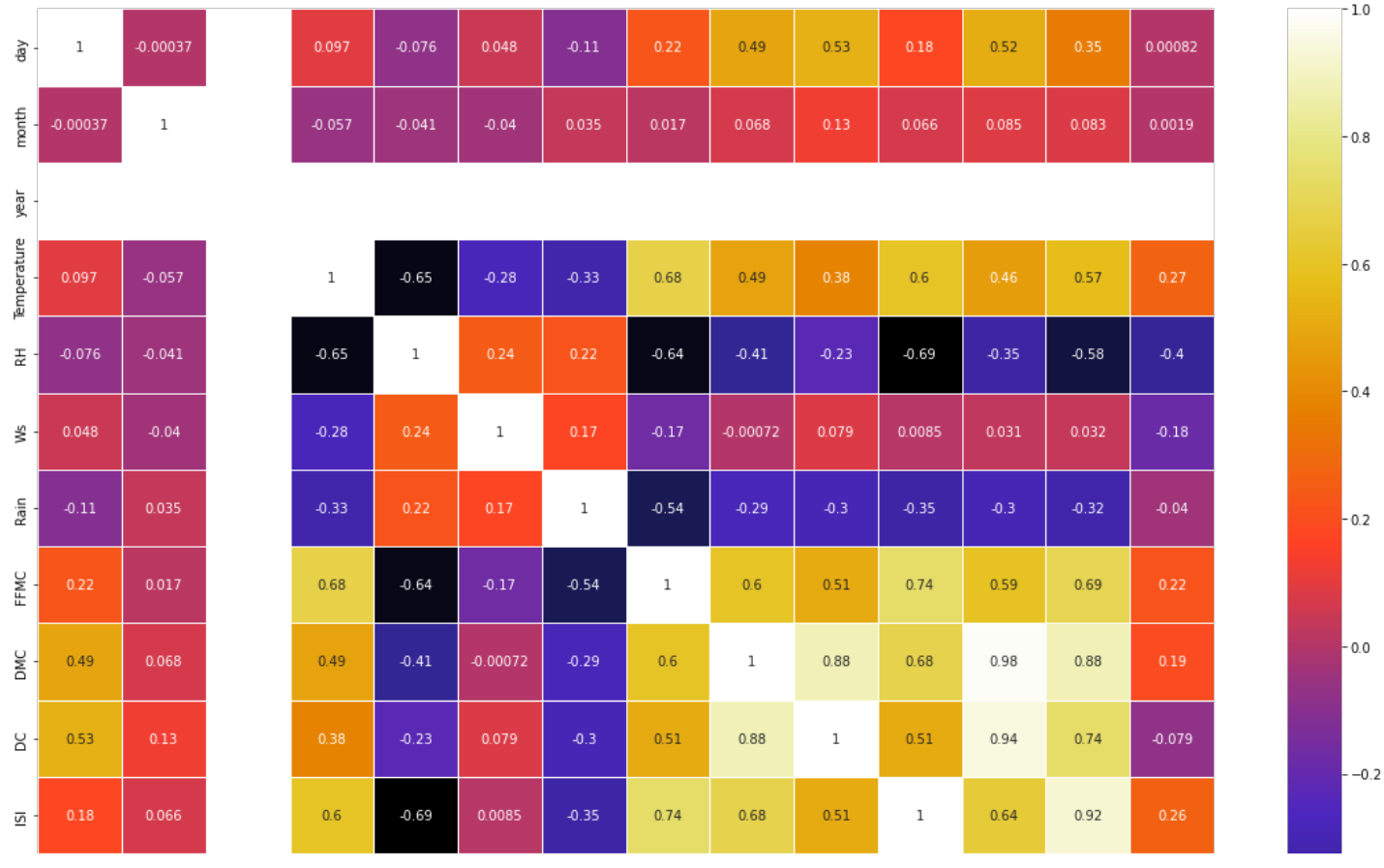
	month	day	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	
month	1.000000	0.000369	NaN	-0.056781	0.041252	0.039880	0.034822	0.017030	0.067943	0.126511	0.065608	0
year	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Temperature	0.097227	-0.056781	NaN	1.000000	-0.651400	-0.284510	-0.326492	0.676568	0.485687	0.376284	0.603871	0
RH	-0.076034	-0.041252	NaN	-0.651400	1.000000	0.244048	0.222356	-0.644873	-0.408519	-0.226941	-0.686667	0
Ws	0.047812	-0.039880	NaN	-0.284510	0.244048	1.000000	0.171506	-0.166548	-0.000721	0.079135	0.008532	0
Rain	-0.112523	0.034822	NaN	-0.326492	0.222356	0.171506	1.000000	-0.543906	-0.288773	-0.298023	-0.347484	0
FFMC	0.224956	0.017030	NaN	0.676568	-0.644873	-0.166548	-0.543906	1.000000	0.603608	0.507397	0.740007	0
DMC	0.491514	0.067943	NaN	0.485687	-0.408519	-0.000721	-0.288773	0.603608	1.000000	0.875925	0.680454	0
DC	0.527952	0.126511	NaN	0.376284	-0.226941	0.079135	-0.298023	0.507397	0.875925	1.000000	0.508643	0
ISI	0.180543	0.065608	NaN	0.603871	-0.686667	0.008532	-0.347484	0.740007	0.680454	0.508643	1.000000	0
BUI	0.517117	0.085073	NaN	0.459789	-0.353841	0.031438	-0.299852	0.592011	0.982248	0.941988	0.644093	1
FWI	0.350781	0.082639	NaN	0.566670	-0.580957	0.032368	-0.324422	0.691132	0.875864	0.739521	0.922895	0
Region	0.000821	0.001857	NaN	0.269555	-0.402682	-0.181160	-0.040013	0.222241	0.192089	-0.078734	0.263197	0

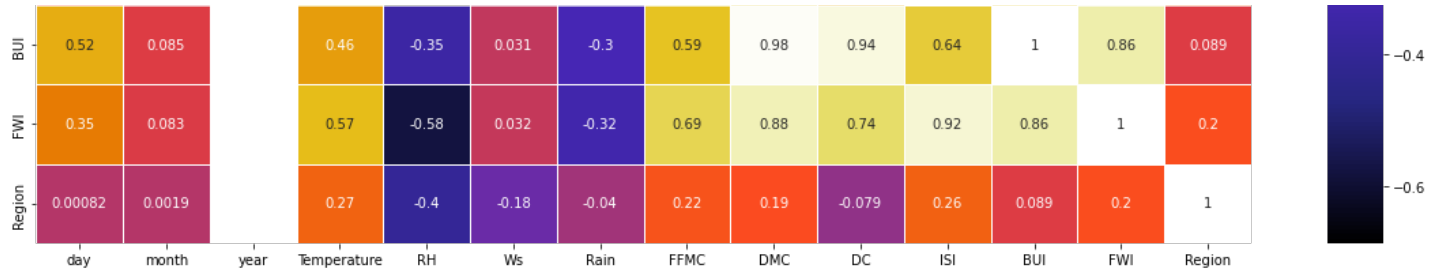
In [34]:

```
plt.figure(figsize=(20,15))
sns.heatmap(df1.corr(),annot= True,linewidths=1, linecolor="white", cbar=True,
            cmap = "CMRmap",xticklabels="auto", yticklabels="auto")
```

Out[34]:

<AxesSubplot:>





In [ ]:

```
```

In [35]:

```
# Encoding Not fire as 0 and Fire as 1
df1['Classes']= np.where(df1['Classes']== 'not fire',0,1)
df1.head()
```

Out[35]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	2012	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0	1
1	2	6	2012	29	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0	1
2	3	6	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0	1
3	4	6	2012	25	89	13	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0	1
4	5	6	2012	27	77	16	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0	1

In [36]:

```
# Check counts
df1.Classes.value_counts()
```

Out[36]:

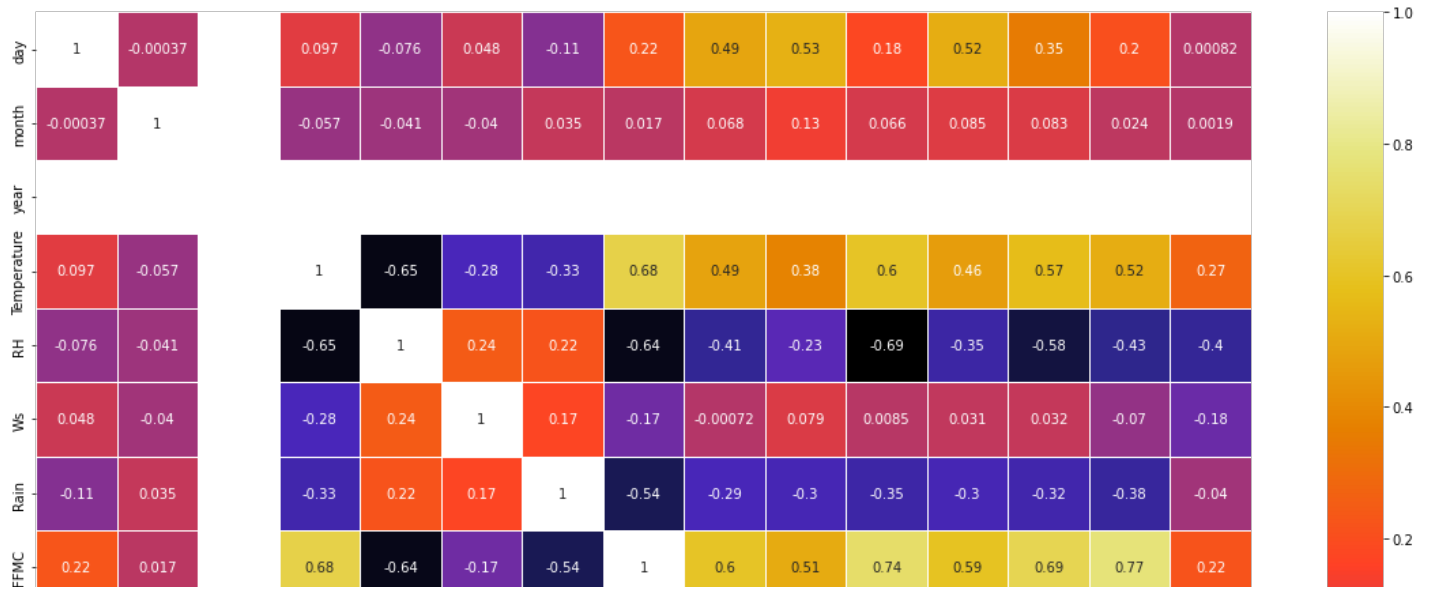
```
1    137
0    106
Name: Classes, dtype: int64
```

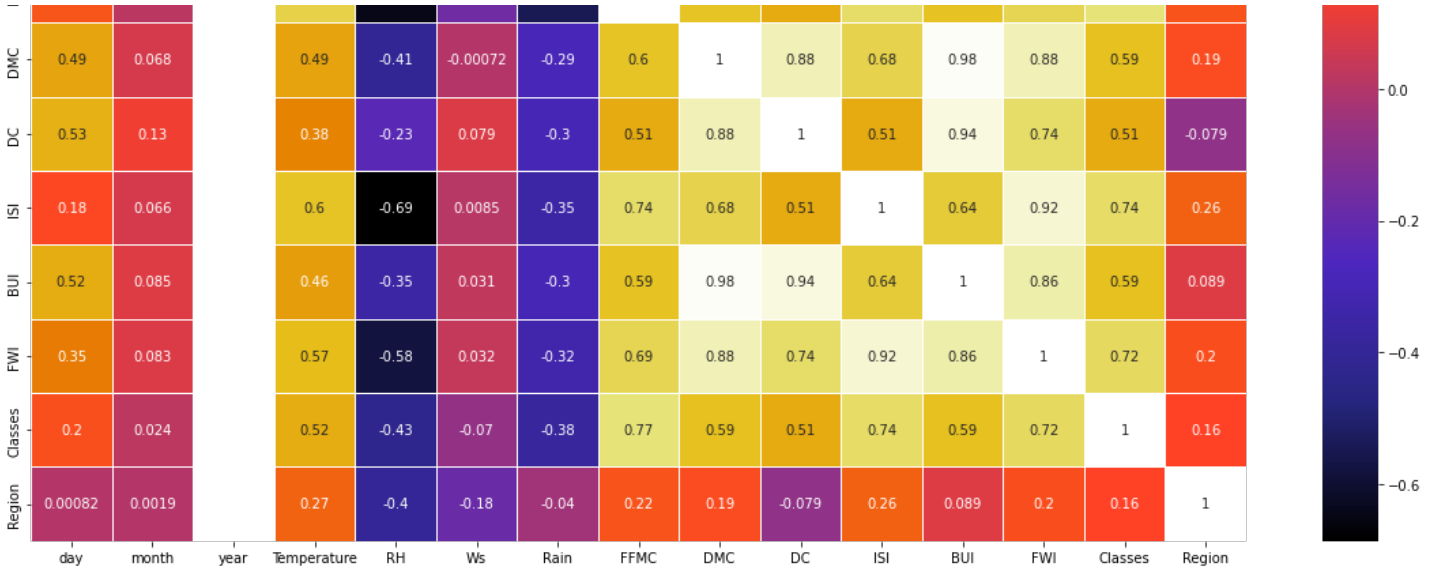
In [37]:

```
plt.figure(figsize=(20,15))
sns.heatmap(df1.corr(),annot= True,linewidths=1, linecolor="white", cbar=True, cmap = "CM
Rmap",xticklabels="auto", yticklabels="auto")
```

Out[37]:

```
<AxesSubplot:>
```



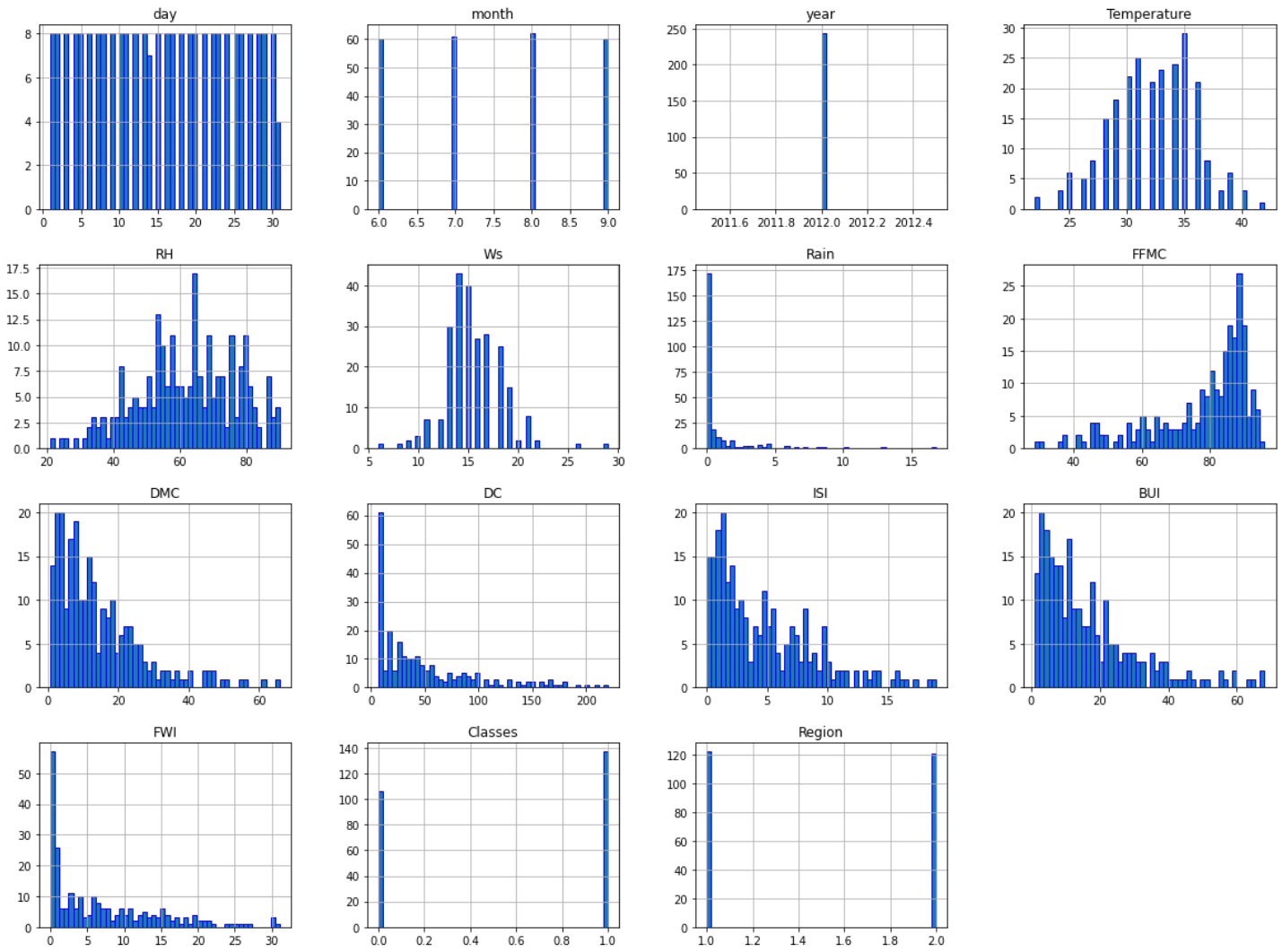


In [38]:

```
df1.to_csv('Algerian_forest_fire_cleaned-data.csv', index=False)
```

In [39]:

```
df1.hist(bins=50, figsize=(20,15), ec = 'b')
plt.show()
```

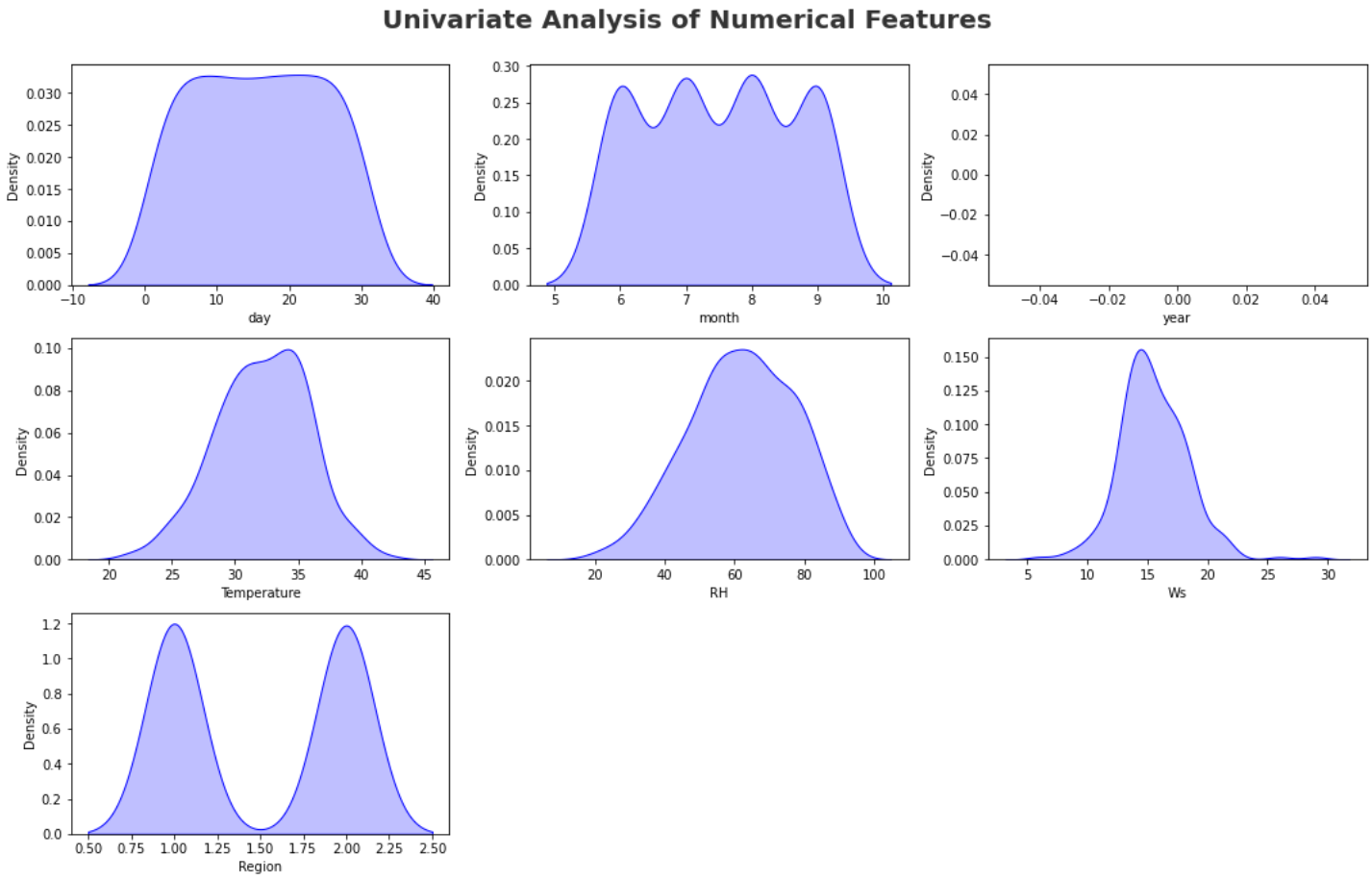


In [40]:

```
plt.figure(figsize=(15, 15))
plt.suptitle('Univariate Analysis of Numerical Features', fontsize=20, fontweight='bold',
alpha=0.8, y=1.)

for i in range(0, len(numeric_features)):
```

```
    plt.subplot(5, 3, i+1)
    sns.kdeplot(x=df1[numeric_features[i]],shade=True, color='b')
    plt.xlabel(numeric_features[i])
    plt.tight_layout()
```



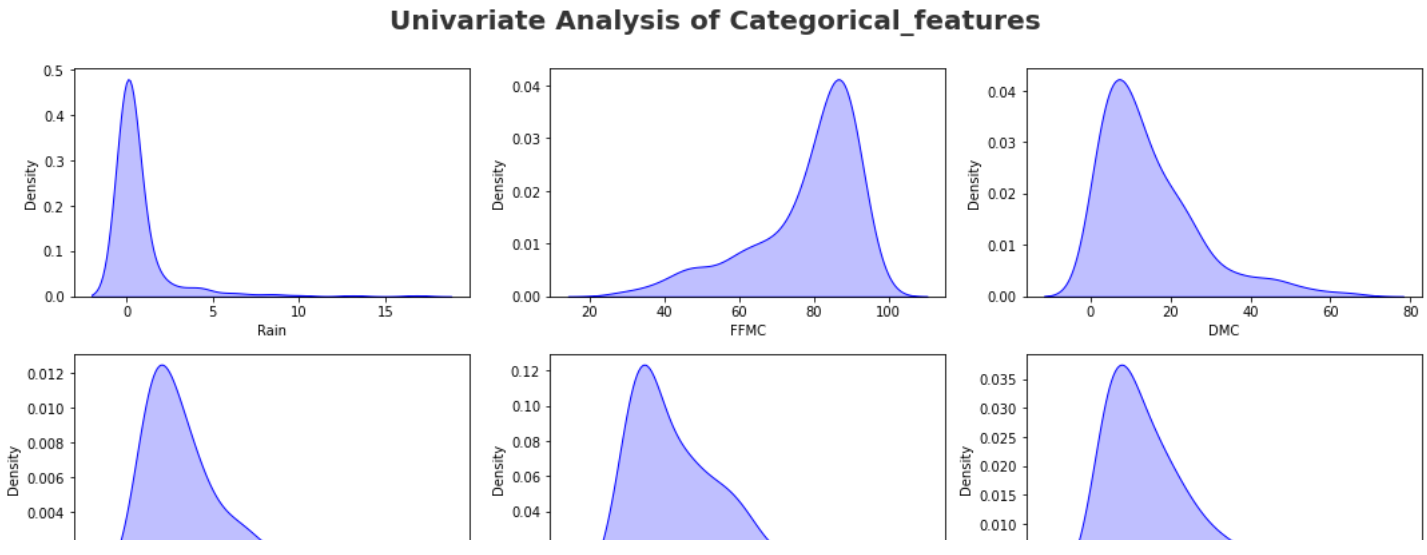
Observations

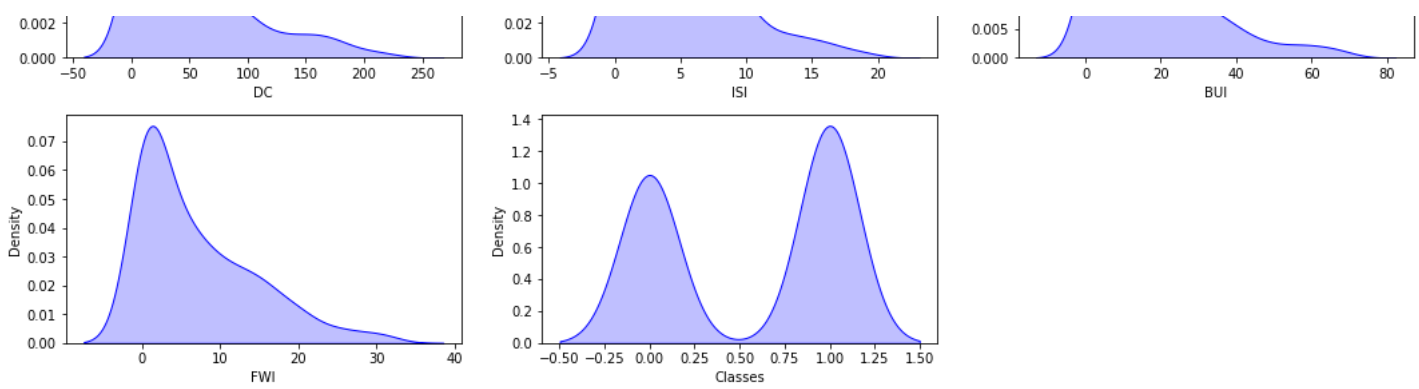
1. **Temperature, RH, Region are Normally Distributed**
2. **Ws has few outliers**

In [41]:

```
plt.figure(figsize=(15, 15))
plt.suptitle('Univariate Analysis of Categorical_features', fontsize=20, fontweight='bold
', alpha=0.8, y=1.)

for i in range(0, len(categorical_features)):
    plt.subplot(5, 3, i+1)
    sns.kdeplot(x=df1[categorical_features[i]],shade=True, color='b')
    plt.xlabel(categorical_features[i])
    plt.tight_layout()
```





Observations

1. **Rain, DMC, DC, ISI, BUI, FWI, are Right-Skewed**
2. **FFMC is Left-Skewed**

In [42]:

```
percentage = df1.Classes.value_counts(normalize=True)*100
percentage
```

Out[42]:

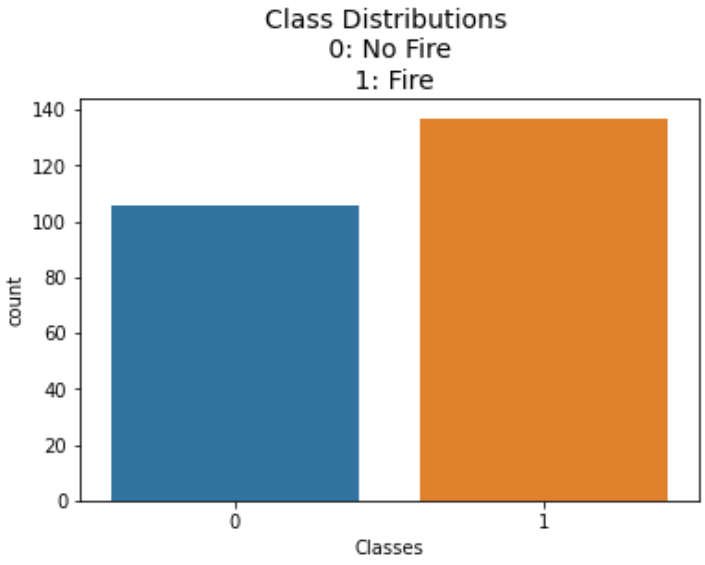
```
1    56.378601
0    43.621399
Name: Classes, dtype: float64
```

In [43]:

```
sns.countplot('Classes', data=df1, palette="tab10")
plt.title('Class Distributions \n 0: No Fire \n 1: Fire', fontsize=14)
```

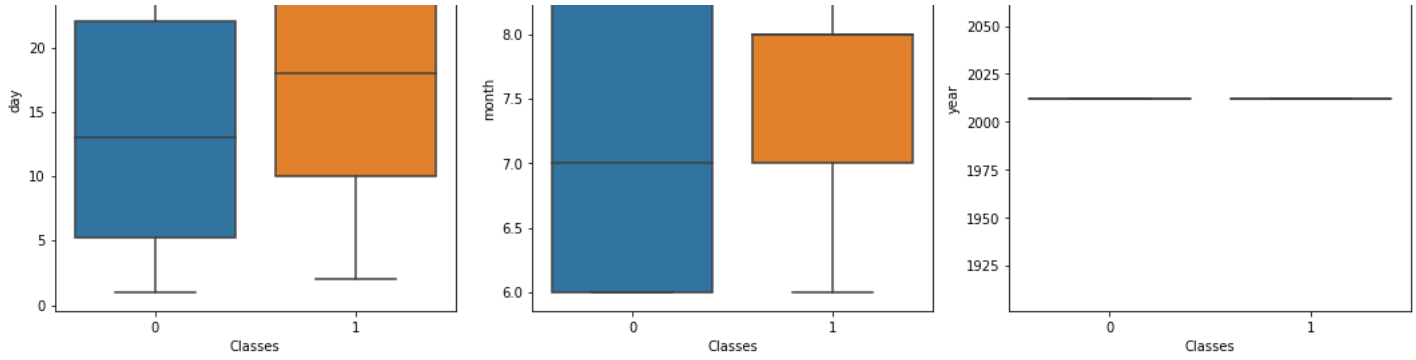
Out[43]:

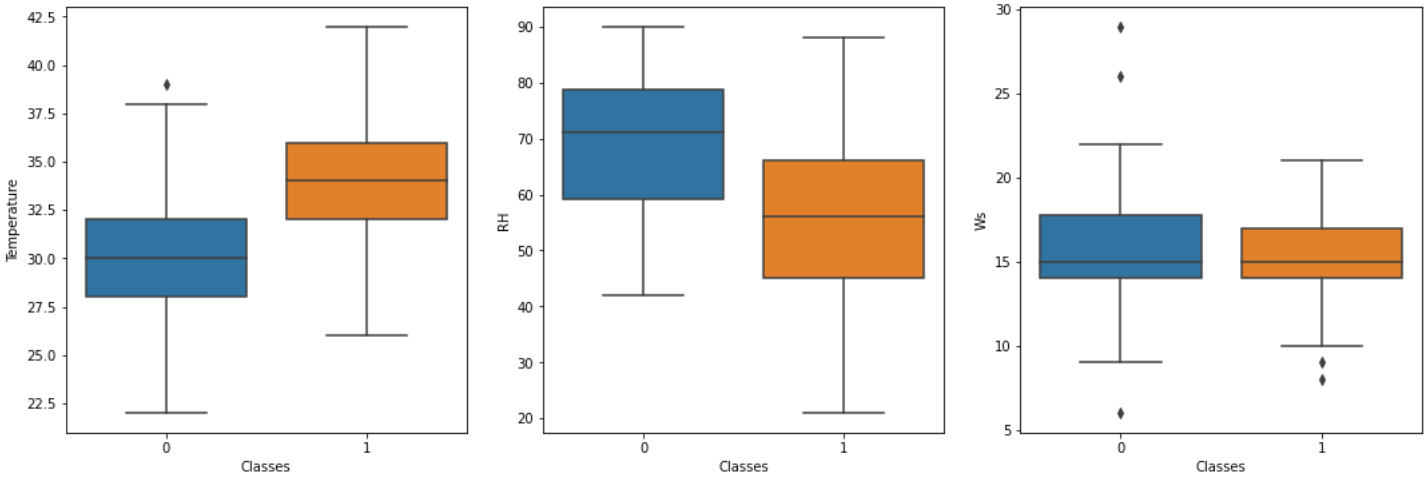
```
Text(0.5, 1.0, 'Class Distributions \n 0: No Fire \n 1: Fire')
```

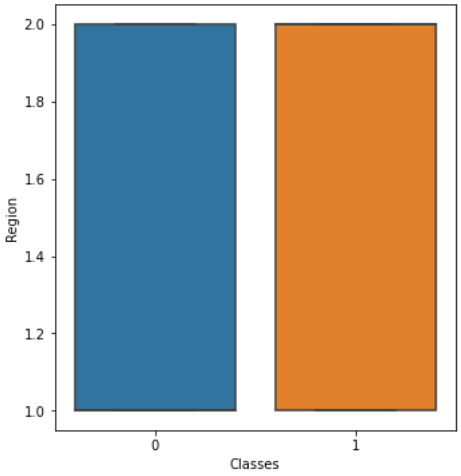


In [44]:

```
fig = plt.figure(figsize=(15, 50))
for i in range(0, len(numeric_features)):
    ax = plt.subplot(10, 3, i+1)
    sns.boxplot(data = df1, x = 'Classes',
    y = df1[numeric_features[i]])
    plt.tight_layout()
```

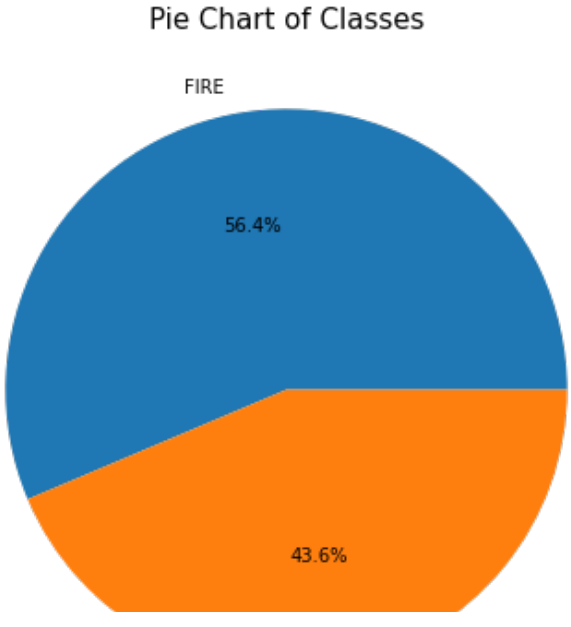






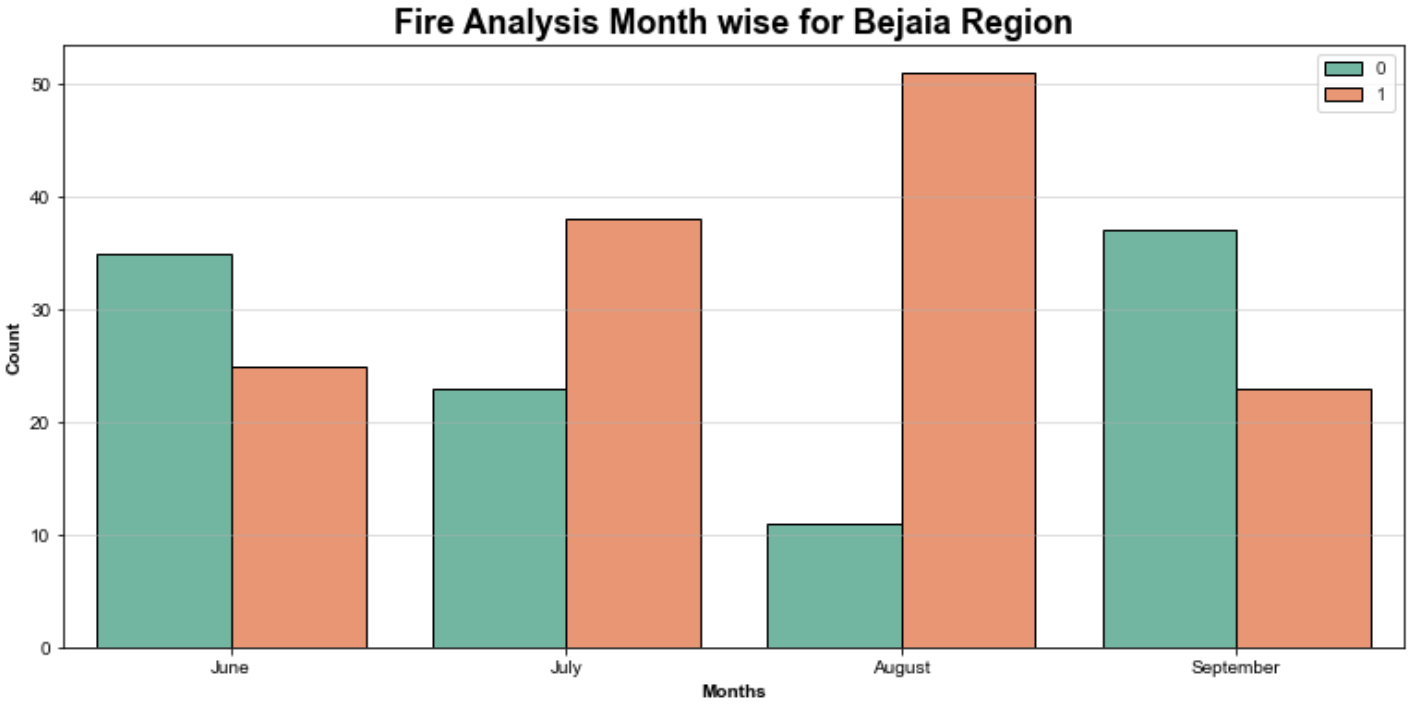
In [45]:

```
classeslabels = ["FIRE", "NOT FIRE"]
plt.figure(figsize =(15, 7))
plt.pie(percentage,labels = classeslabels,autopct='%1.1f%%')
plt.title ("Pie Chart of Classes", fontsize = 15)
plt.show()
```



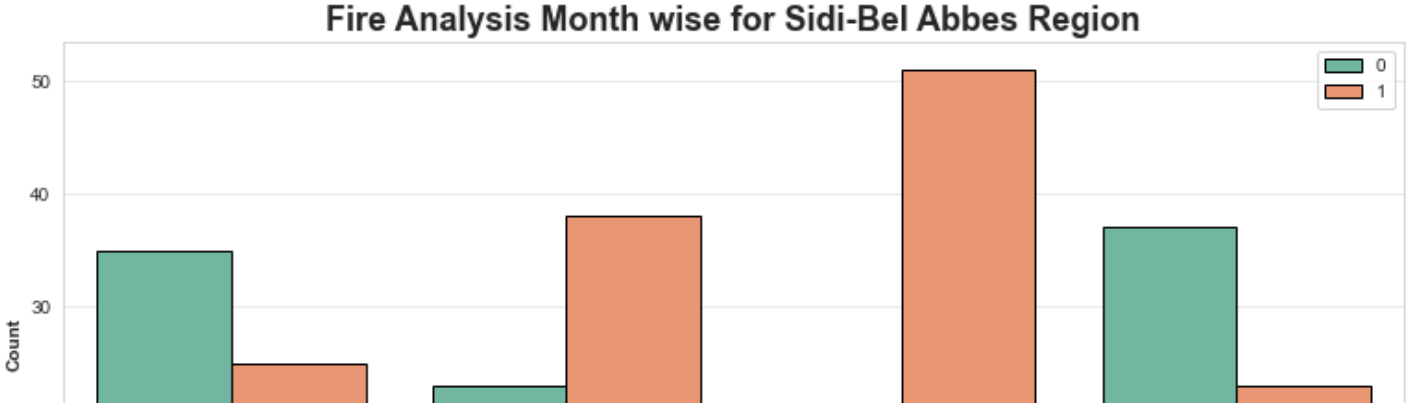
In [46]:

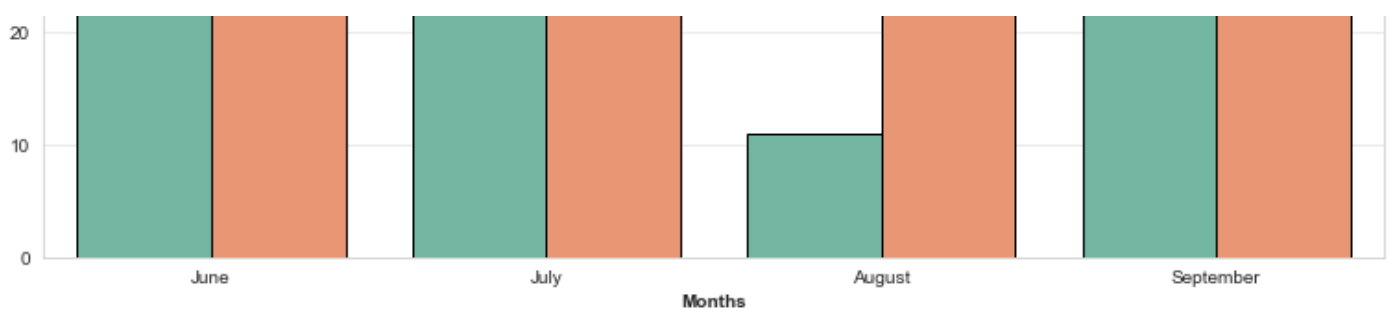
```
temp_df= df1.loc[df1['Region']== 1]
plt.subplots(figsize=(13,6))
sns.set_style('whitegrid')
sns.countplot(x='month',hue='Classes',data= df1,ec = 'black', palette= 'Set2')
plt.title('Fire Analysis Month wise for Bejaia Region', fontsize=18, weight='bold')
plt.ylabel('Count', weight = 'bold')
plt.xlabel('Months', weight= 'bold')
plt.legend(loc='upper right')
plt.xticks(np.arange(4), ['June','July', 'August', 'September',])
plt.grid(alpha = 0.5,axis = 'y')
plt.show()
```



In [47]:

```
temp_df= df1.loc[df1['Region']== 2]
plt.subplots(figsize=(13,6))
sns.set_style('whitegrid')
sns.countplot(x='month',hue='Classes',data= df1,ec = 'black', palette= 'Set2')
plt.title('Fire Analysis Month wise for Sidi-Bel Abbes Region', fontsize=18, weight='bold
')
plt.ylabel('Count', weight = 'bold')
plt.xlabel('Months', weight= 'bold')
plt.legend(loc='upper right')
plt.xticks(np.arange(4), ['June','July', 'August', 'September',])
plt.grid(alpha = 0.5,axis = 'y')
plt.show()
```





Conclusion

1. **Temperature rise being found the main reason to increase the chances of forest fire.**
2. **However, a slight increase in Humidity also decreases the chances of forest fire.**
3. **Wind Speed is negatively correlated, thus it has no such significance in the role of forest fire.**
4. **Good amount of rain can compensate the high chances of increase in forest fire.**
5. **If FFMC > 80 chances of forest fire becomes somewhat high**
6. **If DMC > 10 chances of forest fire becomes somewhat high**
7. **If DC > 50 chances of forest fire becomes somewhat high**
8. **If ISI > 5 chances of forest fire becomes somewhat high**
9. **If BUI > 15 chances of forest fire becomes somewhat high**
10. **If FWI > 5 chances of forest fire becomes somewhat high**
11. **Evidently it is seen from the above two graphs that forest fires occur high in the months of July and August. The management hence should be more careful and take precautionary measures to prevent most of the chances.**

In [ ]: