SVM_Forest Fire

Classify the Size_Categorie using SVM

month month of the year: 'jan' to 'dec'

day day of the week: 'mon' to 'sun'

FFMC FFMC index from the FWI system: 18.7 to 96.20

DMC DMC index from the FWI system: 1.1 to 291.3

DC DC index from the FWI system: 7.9 to 860.6

ISI ISI index from the FWI system: 0.0 to 56.10

temp temperature in Celsius degrees: 2.2 to 33.30

RH relative humidity in %: 15.0 to 100

wind wind speed in km/h: 0.40 to 9.40

rain outside rain in mm/m2 : 0.0 to 6.4

Size_Categorie the burned area of the forest (Small , Large)

Importing the libraries

In [1]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer, TfidfV
from sklearn.preprocessing import StandardScaler

from sklearn import svm
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report


from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.model_selection import train_test_split, cross_val_sco

import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Importing the Dataset

In [2]:
```
forest=pd.read_csv("/Users/chethantumkur/Desktop/Data Science/ASSSI
forest
```

Out[2]:

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthfeb	monthja
0	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	...	0	
1	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	...	0	
2	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	...	0	
3	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	...	0	
4	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
512	aug	sun	81.6	56.7	665.6	1.9	27.8	32	2.7	0.0	...	0	
513	aug	sun	81.6	56.7	665.6	1.9	21.9	71	5.8	0.0	...	0	
514	aug	sun	81.6	56.7	665.6	1.9	21.2	70	6.7	0.0	...	0	
515	aug	sat	94.4	146.0	614.7	11.3	25.6	42	4.0	0.0	...	0	
516	nov	tue	79.5	3.0	106.7	1.1	11.8	31	4.5	0.0	...	0	

517 rows × 31 columns

In [3]:
```
forest.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 517 entries, 0 to 516
Data columns (total 31 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   month          517 non-null    object 
 1   day            517 non-null    object 
 2   FFMC           517 non-null    float64
 3   DMC            517 non-null    float64
 4   DC             517 non-null    float64
 5   ISI            517 non-null    float64
 6   temp           517 non-null    float64
 7   RH             517 non-null    int64  
 8   wind           517 non-null    float64
 9   rain           517 non-null    float64
 10  area           517 non-null    float64
 11  dayfri         517 non-null    int64  
 12  daymon         517 non-null    int64  
 13  daysat         517 non-null    int64  
 14  daysun         517 non-null    int64  
 15  daythu         517 non-null    int64  
 16  daytue         517 non-null    int64  
 17  daywed         517 non-null    int64  
 18  monthapr       517 non-null    int64  
 19  monthaug       517 non-null    int64  
 20  monthdec       517 non-null    int64  
 21  monthfeb       517 non-null    int64  
 22  monthjan       517 non-null    int64  
 23  monthjul       517 non-null    int64  
 24  monthjun       517 non-null    int64  
 25  monthmar       517 non-null    int64  
 26  monthmay       517 non-null    int64  
 27  monthnov       517 non-null    int64  
 28  monthoct       517 non-null    int64  
 29  monthsep       517 non-null    int64  
 30  size_category  517 non-null    object 
dtypes: float64(8), int64(20), object(3)
memory usage: 125.3+ KB
```

In [4]:
```
forest.describe()
```

Out[4]:

	FFMC	DMC	DC	ISI	temp	RH	wind
count	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000
mean	90.644681	110.872340	547.940039	9.021663	18.889168	44.288201	4.017602
std	5.520111	64.046482	248.066192	4.559477	5.806625	16.317469	1.791653
min	18.700000	1.100000	7.900000	0.000000	2.200000	15.000000	0.400000
25%	90.200000	68.600000	437.700000	6.500000	15.500000	33.000000	2.700000
50%	91.600000	108.300000	664.200000	8.400000	19.300000	42.000000	4.000000
75%	92.900000	142.400000	713.900000	10.800000	22.800000	53.000000	4.900000
max	96.200000	291.300000	860.600000	56.100000	33.300000	100.000000	9.400000

8 rows × 28 columns

In [5]:
```
forest[forest.duplicated()].shape
```

Out[5]: (8, 31)

In [6]:
```
forest[forest.duplicated()]
```

Out[6]:

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthfeb	monthja
53	aug	wed	92.1	111.2	654.1	9.6	20.4	42	4.9	0.0	...	0	
100	aug	sun	91.4	142.4	601.4	10.6	19.8	39	5.4	0.0	...	0	
215	mar	sat	91.7	35.8	80.8	7.8	17.0	27	4.9	0.0	...	0	
303	jun	fri	91.1	94.1	232.1	7.1	19.2	38	4.5	0.0	...	0	
426	aug	thu	91.6	248.4	753.8	6.3	20.4	56	2.2	0.0	...	0	
461	aug	sat	93.7	231.1	715.1	8.4	18.9	64	4.9	0.0	...	0	
501	aug	tue	96.1	181.1	671.2	14.3	21.6	65	4.9	0.8	...	0	
508	aug	fri	91.0	166.9	752.6	7.1	25.9	41	3.6	0.0	...	0	

8 rows × 31 columns

In [7]:
```
Forest = forest.drop_duplicates()
```

In [8]:
```
Forest
```

Out[8]:

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthfeb	monthja
0	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	...	0	
1	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	...	0	
2	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	...	0	
3	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	...	0	
4	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	.
512	aug	sun	81.6	56.7	665.6	1.9	27.8	32	2.7	0.0	...	0	
513	aug	sun	81.6	56.7	665.6	1.9	21.9	71	5.8	0.0	...	0	
514	aug	sun	81.6	56.7	665.6	1.9	21.2	70	6.7	0.0	...	0	
515	aug	sat	94.4	146.0	614.7	11.3	25.6	42	4.0	0.0	...	0	
516	nov	tue	79.5	3.0	106.7	1.1	11.8	31	4.5	0.0	...	0	

509 rows × 31 columns

In [9]:
```
Forest['size_category'].value_counts()
```

Out[9]:
```
small    371
large    138
Name: size_category, dtype: int64
```

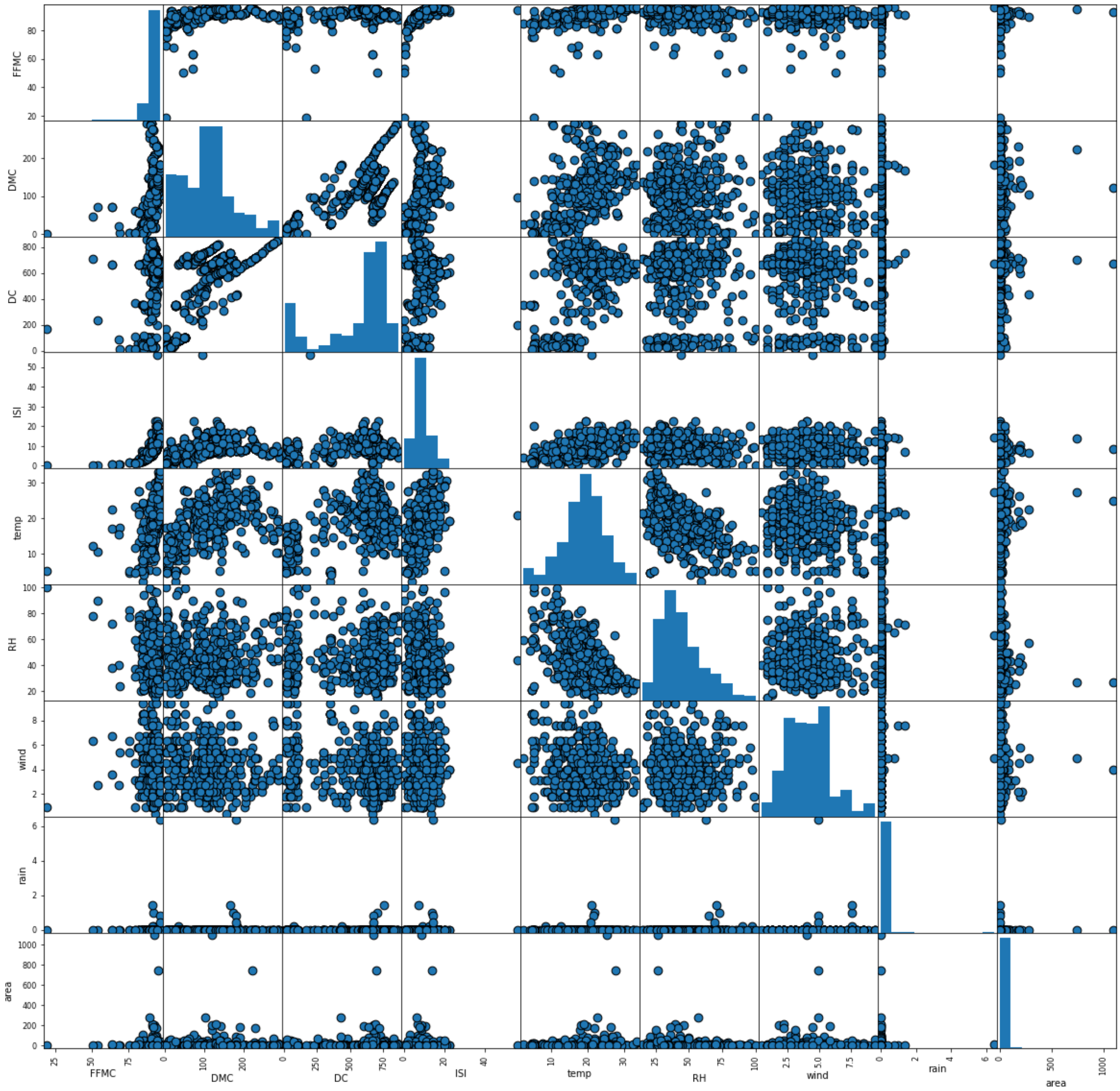
In [10]:
```
pd.crosstab(Forest['size_category'], Forest['month'])
```

Out[10]:

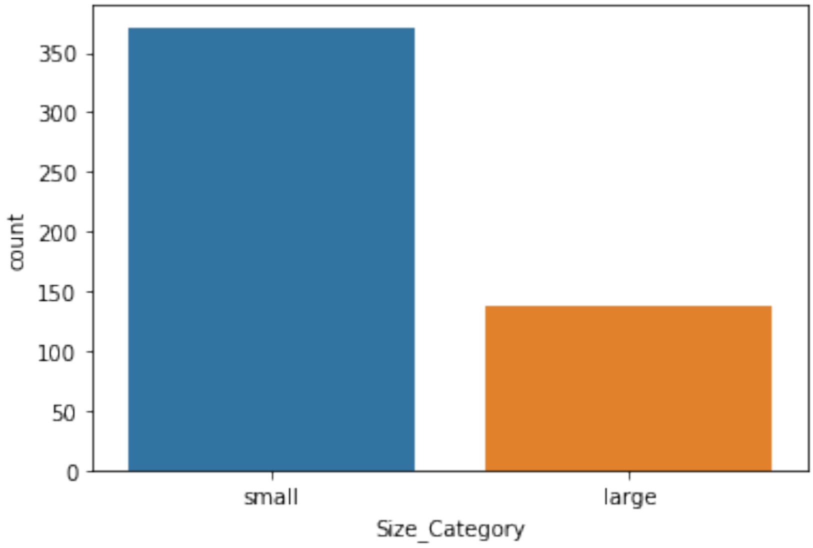
month	apr	aug	dec	feb	jan	jul	jun	mar	may	nov	oct	sep
size_category												
large	2	43	8	6	0	9	3	11	1	0	4	51
small	7	135	1	14	2	23	13	42	1	1	11	121

scatter matrix to observe relationship between every colomn attribute.

In [11]:
```
pd.plotting.scatter_matrix(Forest.iloc[:,:11],
                                              figsize= [20,20],
                                              diagonal='hist',
                                              alpha=1,
                                              s = 300,
                                              marker = '.',
                                              edgecolor= "black")
plt.show()
```



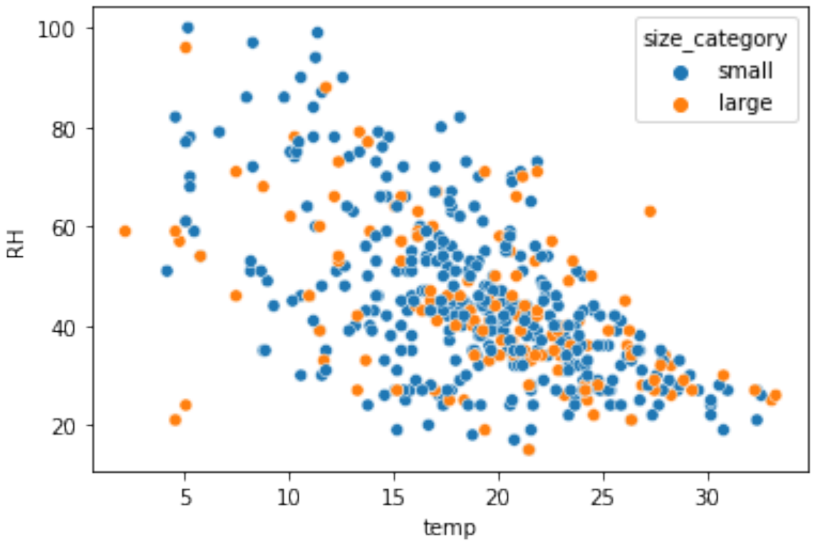
In [12]:
```
sns.countplot(x='size_category',data= Forest)
plt.xlabel('Size_Category')
plt.ylabel('count')
plt.show()
Forest['size_category'].value_counts()
```



Out[12]:
```
small    371
large    138
Name: size_category, dtype: int64
```

In [13]:
```
sns.scatterplot(Forest['temp'],Forest['RH'],hue=Forest['size_catego
```

Out[13]:
```
<AxesSubplot:xlabel='temp', ylabel='RH'>
```



In [14]:
```
x= Forest.iloc[:,2:30]
y = Forest['size_category']
```

In [15]:
```
x
```

Out[15]:

	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area	dayfri	...	monthdec	monthf
0	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	0.00	1	...	0	
1	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.00	0	...	0	
2	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.00	0	...	0	
3	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.00	1	...	0	
4	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.00	0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
512	81.6	56.7	665.6	1.9	27.8	32	2.7	0.0	6.44	0	...	0	
513	81.6	56.7	665.6	1.9	21.9	71	5.8	0.0	54.29	0	...	0	
514	81.6	56.7	665.6	1.9	21.2	70	6.7	0.0	11.16	0	...	0	
515	94.4	146.0	614.7	11.3	25.6	42	4.0	0.0	0.00	0	...	0	
516	79.5	3.0	106.7	1.1	11.8	31	4.5	0.0	0.00	0	...	0	

509 rows × 28 columns

Normalising the data as there is scale difference Splitting the dataset into training and test samples

In [16]:
```
def norm_func(i):
    x= (i-i.min())/(i.max()-i.min())
    return (x)
```

In [17]:
```
X = norm_func(x)
```

In [18]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size
```

In [19]:
```
X_train
```

Out[19]:

	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
144	0.990968	0.340455	0.592706	0.235294	0.694534	0.200000	0.555556	0.0	0.000706
474	0.958710	0.353894	0.362144	0.192513	0.778135	0.235294	0.255556	0.0	0.009241
135	0.965161	0.476568	0.687581	0.361854	0.495177	0.435294	0.600000	0.0	0.000000
284	0.858065	0.013094	0.009265	0.112299	0.170418	0.364706	0.844444	0.0	0.022221
364	0.944516	0.381116	0.894101	0.115865	0.607717	0.235294	0.255556	0.0	0.005179
...	...	...	...	...	...	...	...	...	...
512	0.811613	0.191592	0.771315	0.033868	0.823151	0.200000	0.255556	0.0	0.005904
167	0.997419	0.434183	0.659787	0.294118	0.681672	0.211765	0.455556	0.0	0.002301
7	0.939355	0.497243	0.703999	0.190731	0.186495	0.835294	0.200000	0.0	0.000000
222	0.889032	0.176085	0.112466	0.089127	0.282958	0.364706	0.600000	0.0	0.033781
330	0.948387	0.348725	0.872053	0.149733	0.707395	0.141176	0.300000	0.0	0.006032

356 rows × 28 columns

In [20]:
```
X_test
```

Out[20]:

	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
475	0.967742	0.415575	0.401431	0.320856	0.659164	0.294118	1.000000	0.0	0.002924
357	0.952258	0.416609	0.916852	0.181818	0.440514	0.470588	0.355556	0.0	0.000000
325	0.948387	0.348725	0.872053	0.149733	0.707395	0.141176	0.300000	0.0	0.000000
470	0.932903	0.046520	0.020758	0.219251	0.495177	0.141176	0.600000	0.0	0.000000
249	0.960000	0.538249	0.772605	0.240642	0.627010	0.294118	0.000000	0.0	0.002264
...	...	...	...	...	...	...	...	...	...
175	0.922581	0.330117	0.722763	0.158645	0.581994	0.282353	0.500000	0.0	0.004153
489	0.985806	0.483115	0.701184	0.315508	0.591640	0.505882	0.100000	0.0	0.000000
345	0.939355	0.444521	0.937258	0.133690	0.469453	0.329412	0.300000	0.0	0.005345
443	0.929032	0.274983	0.422657	0.299465	0.405145	0.741176	0.844444	0.0	0.000000
375	0.935484	0.321158	0.863727	0.149733	0.424437	0.494118	0.500000	0.0	0.036073

153 rows × 28 columns

In [21]:
```
y_train
```

Out[21]:
```
144    small
474    large
135    small
284    large
364    small
       ...
512    large
167    small
7      small
222    large
330    large
Name: size_category, Length: 356, dtype: object
```

In [22]:
```
y_test
```

Out[22]:
```
475    small
357    small
325    small
470    small
249    small
       ...
175    small
489    small
345    small
443    small
375    large
Name: size_category, Length: 153, dtype: object
```

In [23]:
```
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

Out[23]:
```
((356, 28), (356,), (153, 28), (153,))
```

GRID SEARCH

In [24]:
```
#Grid search
clf = SVC()
param_grid = [{'kernel':['rbf'],'gamma':[50,5,10,0.5],'C':[15,14,13
gsv = GridSearchCV(clf,param_grid,cv=10)
gsv.fit(X_train,y_train)
```

Out[24]:
```
GridSearchCV(cv=10, estimator=SVC(),
             param_grid=[{'C': [15, 14, 13, 12, 11, 10, 0.1, 0.001
],
                          'gamma': [50, 5, 10, 0.5], 'kernel': ['r
bf']}])
```

In [25]:
```
gsv.best_params_ , gsv.best_score_
```

Out[25]:
```
({'C': 0.1, 'gamma': 50, 'kernel': 'rbf'}, 0.7162698412698413)
```

In [26]:
```
clf = SVC(C= 15, gamma = 50)
clf.fit(X_train , y_train)
y_pred = clf.predict(X_test)
acc = accuracy_score(y_test, y_pred) * 100
print("Accuracy =", acc)
confusion_matrix(y_test, y_pred)
```

```
Accuracy = 74.50980392156863
```

Out[26]:
```
array([[  5,  32],
       [  7, 109]])
```

In [27]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

       large       0.42      0.14      0.20        37
       small       0.77      0.94      0.85       116

    accuracy                           0.75       153
   macro avg       0.59      0.54      0.53       153
weighted avg       0.69      0.75      0.69       153
```

KERNAL For Accuracy

KERNAL= LINEAR

In [28]:
```
model_linear = SVC(kernel = "linear")
model_linear.fit(X_train,y_train)
```

Out[28]:
```
SVC(kernel='linear')
```

In [29]:
```
pred_test_linear = model_linear.predict(X_test)
```

In [35]:
```
np.mean(pred_test_linear==y_test)
```

Out[35]:
```
0.7647058823529411
```

In [36]:
```
acc = accuracy_score(y_test, pred_test_linear) * 100
print("Accuracy =", acc)
confusion_matrix(y_test, pred_test_linear)
```

```
Accuracy = 76.47058823529412
```

Out[36]:
```
array([[  2,  35],
       [  1, 115]])
```

KERNAL= POLYNOMIAL

In [37]:
```
model_poly = SVC(kernel = "poly")
model_poly.fit(X_train,y_train)
```

Out[37]:
```
SVC(kernel='poly')
```

In [38]:
```
pred_test_poly = model_poly.predict(X_test)
```

In [39]:
```
np.mean(pred_test_poly==y_test)
```

Out[39]:
```
0.7581699346405228
```

KERNAL= Radial Basis Function

In [40]:
```
model_rbf = SVC(kernel = "rbf")
model_rbf.fit(X_train,y_train)
```

Out[40]:
```
SVC()
```

In [41]:
```
pred_test_rbf = model_rbf.predict(X_test)
```

In [42]:
```
np.mean(pred_test_rbf==y_test) #Accuracy = 75.81%
```

Out[42]:
```
0.7581699346405228
```

KERNAL= Sigmoid

In [43]:
```
#'sigmoid'
model_sig = SVC(kernel = "sigmoid")
model_sig.fit(X_train,y_train)
```

Out[43]:
```
SVC(kernel='sigmoid')
```

In [44]:
```
pred_test_sig = model_rbf.predict(X_test)
```

In [45]:
```
np.mean(pred_test_sig==y_test)
```

Out[45]:
```
0.7581699346405228
```

Conclusion:

Kernal Linear is more Accurate.

In [ ]:
```
```