Problem Statement - classify the Size_Categorie using SVM

1. Import Necessary Libraries

In [1]:
```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import tensorflow as tf
```

2. Import Data

In [2]:
```
Forest_Fires_SVM = pd.read_csv('forestfires.csv')
Forest_Fires_SVM
```

Out[2]:

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

3. Data Understanding

In [3]:
```
Forest_Fires_SVM.shape
```

Out[3]:
```
(517, 31)
```

In [4]:
```
Forest_Fires_SVM.isna().sum()
```

Out[4]:
```
month            0
day              0
FFMC             0
DMC              0
DC               0
ISI              0
temp             0
RH               0
wind             0
rain             0
area             0
dayfri           0
daymon           0
daysat           0
daysun           0
daythu           0
daytue           0
daywed           0
monthapr         0
monthaug         0
monthdec         0
monthfeb         0
monthjan         0
monthjul         0
monthjun         0
monthmar         0
monthmay         0
monthnov         0
monthoct         0
monthsep         0
size_category    0
dtype: int64
```

In [5]:
```
Forest_Fires_SVM.describe()
```

Out[5]:

	FFMC	DMC	DC	ISI	temp	RH	wind	
count	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.00
mean	90.644681	110.872340	547.940039	9.021663	18.889168	44.288201	4.017602	0.02
std	5.520111	64.046482	248.066192	4.559477	5.806625	16.317469	1.791653	0.29
min	18.700000	1.100000	7.900000	0.000000	2.200000	15.000000	0.400000	0.00
25%	90.200000	68.600000	437.700000	6.500000	15.500000	33.000000	2.700000	0.00
50%	91.600000	108.300000	664.200000	8.400000	19.300000	42.000000	4.000000	0.00
75%	92.900000	142.400000	713.900000	10.800000	22.800000	53.000000	4.900000	0.00
max	96.200000	291.300000	860.600000	56.100000	33.300000	100.000000	9.400000	6.40

8 rows × 28 columns

In [6]:
```
Forest_Fires_SVM.dtypes
```

Out[6]:
```
month            object
day              object
FFMC            float64
DMC             float64
DC              float64
ISI             float64
temp            float64
RH                int64
wind            float64
rain            float64
area            float64
dayfri            int64
daymon            int64
daysat            int64
daysun            int64
daythu            int64
daytue            int64
daywed            int64
monthapr          int64
monthaug          int64
monthdec          int64
monthfeb          int64
monthjan          int64
monthjul          int64
monthjun          int64
monthmar          int64
monthmay          int64
monthnov          int64
monthoct          int64
monthsep          int64
size_category    object
dtype: object
```

4. Data Preparation

In [7]:
```
from sklearn import preprocessing
```

In [8]:
```
LabelEncoder = preprocessing.LabelEncoder()
Forest_Fires_SVM['month'] = LabelEncoder.fit_transform(Forest_Fires_SVM['month'])
Forest_Fires_SVM['day'] = LabelEncoder.fit_transform(Forest_Fires_SVM['day'])
Forest_Fires_SVM['size_category'] = LabelEncoder.fit_transform(Forest_Fires_SVM['
```

In [9]:
```
Forest_Fires_SVM
```

Out[9]:

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthfeb	monthjan	mont
0	7	0	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	...	0	0	
1	10	5	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	...	0	0	
2	10	2	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	...	0	0	
3	7	0	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	...	0	0	
4	7	3	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
512	1	3	81.6	56.7	665.6	1.9	27.8	32	2.7	0.0	...	0	0	
513	1	3	81.6	56.7	665.6	1.9	21.9	71	5.8	0.0	...	0	0	
514	1	3	81.6	56.7	665.6	1.9	21.2	70	6.7	0.0	...	0	0	
515	1	2	94.4	146.0	614.7	11.3	25.6	42	4.0	0.0	...	0	0	
516	9	5	79.5	3.0	106.7	1.1	11.8	31	4.5	0.0	...	0	0	

517 rows × 31 columns

In [10]:
```
Forest_Fires_SVM.head(30)
```

Out[10]:

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthfeb	monthjan	month
0	7	0	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	...	0	0	
1	10	5	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	...	0	0	
2	10	2	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	...	0	0	
3	7	0	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	...	0	0	
4	7	3	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	...	0	0	
5	1	3	92.3	85.3	488.0	14.7	22.2	29	5.4	0.0	...	0	0	
6	1	1	92.3	88.9	495.6	8.5	24.1	27	3.1	0.0	...	0	0	
7	1	1	91.5	145.4	608.2	10.7	8.0	86	2.2	0.0	...	0	0	
8	11	5	91.0	129.5	692.6	7.0	13.1	63	5.4	0.0	...	0	0	
9	11	2	92.5	88.0	698.6	7.1	22.8	40	4.0	0.0	...	0	0	
10	11	2	92.5	88.0	698.6	7.1	17.8	51	7.2	0.0	...	0	0	
11	11	2	92.8	73.2	713.0	22.6	19.3	38	4.0	0.0	...	0	0	
12	1	0	63.5	70.8	665.3	0.8	17.0	72	6.7	0.0	...	0	0	
13	11	1	90.9	126.5	686.5	7.0	21.3	42	2.2	0.0	...	0	0	
14	11	6	92.9	133.3	699.6	9.2	26.4	21	4.5	0.0	...	0	0	
15	11	0	93.3	141.2	713.9	13.9	22.9	44	5.4	0.0	...	0	0	
16	7	2	91.7	35.8	80.8	7.8	15.1	27	5.4	0.0	...	0	0	
17	10	1	84.9	32.8	664.2	3.0	16.7	47	4.9	0.0	...	0	0	
18	7	6	89.2	27.9	70.8	6.3	15.9	35	4.0	0.0	...	0	0	
19	0	2	86.3	27.4	97.1	5.1	9.3	44	4.5	0.0	...	0	0	
20	11	5	91.0	129.5	692.6	7.0	18.3	40	2.7	0.0	...	0	0	
21	11	1	91.8	78.5	724.3	9.2	19.1	38	2.7	0.0	...	0	0	
22	6	3	94.3	96.3	200.0	56.1	21.0	44	4.5	0.0	...	0	0	
23	1	2	90.2	110.9	537.4	6.2	19.5	43	5.8	0.0	...	0	0	
24	1	2	93.5	139.4	594.2	20.3	23.7	32	5.8	0.0	...	0	0	
25	1	3	91.4	142.4	601.4	10.6	16.3	60	5.4	0.0	...	0	0	
26	11	0	92.4	117.9	668.0	12.2	19.0	34	5.8	0.0	...	0	0	
27	11	1	90.9	126.5	686.5	7.0	19.4	48	1.3	0.0	...	0	0	
28	11	2	93.4	145.4	721.4	8.1	30.2	24	2.7	0.0	...	0	0	
29	11	3	93.5	149.3	728.6	8.1	22.8	39	3.6	0.0	...	0	0	

30 rows × 31 columns

In [11]:
```
Forest_Fires_SVM.dtypes
```

Out[11]:
```
month              int32
day                int32
FFMC             float64
DMC              float64
DC               float64
ISI              float64
temp             float64
RH                 int64
wind             float64
rain             float64
area             float64
dayfri             int64
daymon             int64
daysat             int64
daysun             int64
daythu             int64
daytue             int64
daywed             int64
monthapr           int64
monthaug           int64
monthdec           int64
monthfeb           int64
monthjan           int64
monthjul           int64
monthjun           int64
monthmar           int64
monthmay           int64
monthnov           int64
monthoct           int64
monthsep           int64
size_category      int32
dtype: object
```

4. Model Building

In [12]:
```
X = Forest_Fires_SVM.drop(labels='size_category',axis=1)
y = Forest_Fires_SVM[['size_category']]
```

In [13]:
```
X
```

0	7	0	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	...	0	0
1	10	5	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	...	0	0
2	10	2	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	...	0	0
3	7	0	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	...	0	0
4	7	3	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	...	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
512	1	3	81.6	56.7	665.6	1.9	27.8	32	2.7	0.0	...	0	0
513	1	3	81.6	56.7	665.6	1.9	21.9	71	5.8	0.0	...	0	0
514	1	3	81.6	56.7	665.6	1.9	21.2	70	6.7	0.0	...	0	0
515	1	2	94.4	146.0	614.7	11.3	25.6	42	4.0	0.0	...	0	0
516	9	5	79.5	3.0	106.7	1.1	11.8	31	4.5	0.0	...	0	0

517 rows × 30 columns

In [14]:
```
y
```

0	1
1	1
2	1
3	1
4	1
...	...
512	0
513	0
514	0
515	1
516	1

517 rows × 1 columns

In [15]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y, test_size=0.20, random_stat
```

In [16]:
```
#For test data
X_train.shape, y_train.shape
```

Out[16]: ((413, 30), (413, 1))

In [17]:
```
X_test.shape,y_test.shape
```

Out[17]: ((104, 30), (104, 1))

6. Model Training, Testing & Evaluation

In [42]:
```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, confus
```

Kernel Trics

In [55]:
```
import warnings
warnings.filterwarnings('ignore')
svc_classifier = SVC(kernel = 'linear')
svc_classifier.fit(X_train,y_train)
y_pred = svc_classifier.predict(X_test)
print('Overall Accuracy       :', round(accuracy_score(y_test,y_pred),2))
print('Precision Score        :', round(precision_score(y_test,y_pred),2))
print('Recall Score           :', round(recall_score(y_test,y_pred),2))
print('AUC Score              :', round(roc_auc_score(y_test,y_pred),2))
print('confusion Matrix       :', confusion_matrix(y_test,y_pred))
print('Classification Report  :', classification_report(y_test,y_pred))
```

```
Overall Accuracy       : 0.99
Precision Score        : 1.0
Recall Score           : 0.99
AUC Score              : 0.99
confusion Matrix       : [[29  0]
 [ 1 74]]
Classification Report  : /n               precision    recall  f1-score   supp
ort

           0       0.97      1.00      0.98        29
           1       1.00      0.99      0.99        75

    accuracy                           0.99       104
   macro avg       0.98      0.99      0.99       104
weighted avg       0.99      0.99      0.99       104
```

In [56]:
```
svc_classifier = SVC(kernel = 'poly')
svc_classifier.fit(X_train,y_train)
y_pred = svc_classifier.predict(X_test)
print('Overall Accuracy       :', round(accuracy_score(y_test,y_pred),2))
print('Precision Score        :', round(precision_score(y_test,y_pred),2))
print('Recall Score           :', round(recall_score(y_test,y_pred),2))
print('AUC Score              :', round(roc_auc_score(y_test,y_pred),2))
print('confusion Matrix       :', confusion_matrix(y_test,y_pred))
print('Classification Report  :', classification_report(y_test,y_pred))
```

```
Overall Accuracy       : 0.76
Precision Score        : 0.75
Recall Score           : 1.0
AUC Score              : 0.57
confusion Matrix       : [[ 4 25]
 [ 0 75]]
Classification Report  :               precision    recall  f1-score   support

           0       1.00      0.14      0.24        29
           1       0.75      1.00      0.86        75

    accuracy                           0.76       104
   macro avg       0.88      0.57      0.55       104
weighted avg       0.82      0.76      0.69       104
```

In [57]:
```
svc_classifier = SVC(kernel = 'rbf')
svc_classifier.fit(X_train,y_train)
y_pred = svc_classifier.predict(X_test)
print('Overall Accuracy       :', round(accuracy_score(y_test,y_pred),2))
print('Precision Score        :', round(precision_score(y_test,y_pred),2))
print('Recall Score           :', round(recall_score(y_test,y_pred),2))
print('AUC Score              :', round(roc_auc_score(y_test,y_pred),2))
print('confusion Matrix       :', confusion_matrix(y_test,y_pred))
print('Classification Report  :', classification_report(y_test,y_pred))
```

```
Overall Accuracy       : 0.74
Precision Score        : 0.74
Recall Score           : 1.0
AUC Score              : 0.53
confusion Matrix       : [[ 2 27]
 [ 0 75]]
Classification Report  :               precision    recall  f1-score   support

           0       1.00      0.07      0.13        29
           1       0.74      1.00      0.85        75

    accuracy                           0.74       104
   macro avg       0.87      0.53      0.49       104
weighted avg       0.81      0.74      0.65       104
```

Conclusion - For linear kernel trics Accuracy, Precision score, Recall Score, Roc_Auc Score is good

In [ ]: