In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
```

In [2]:

```
zero_one_colourmap = ListedColormap(('red','green'))
```

In [3]:

```
forest_data = pd.read_csv('forestfires.csv')
forest_data
```

Out[3]:

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthfeb	monthjan	mont
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	

517 rows × 31 columns

In [4]:

```
# initial analysis
forest_data.shape
```

Out[4]:

```
(517, 31)
```

In [5]:

```
forest_data.isna().sum()
```

Out[5]:

```
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 [6]:
```
forest_data.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
```

In [7]:
```
# converting category column to numeric form.
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [8]:

```
forest_data['month'] = le.fit_transform(forest_data['month'])
forest_data['day'] = le.fit_transform(forest_data['day'])
forest_data['size_category'] = le.fit_transform(forest_data['size_category'])
forest_data
```

Out[8]:

	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	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
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	

In [9]:
```
forest_data.dtypes
```

Out[9]:
```
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
```

In [10]:
```
#splitting the data for training and testing.
X = forest_data.drop(['size_category'], axis=1)
y= forest_data['size_category']
```

In [11]:
```
X
```

Out[11]:

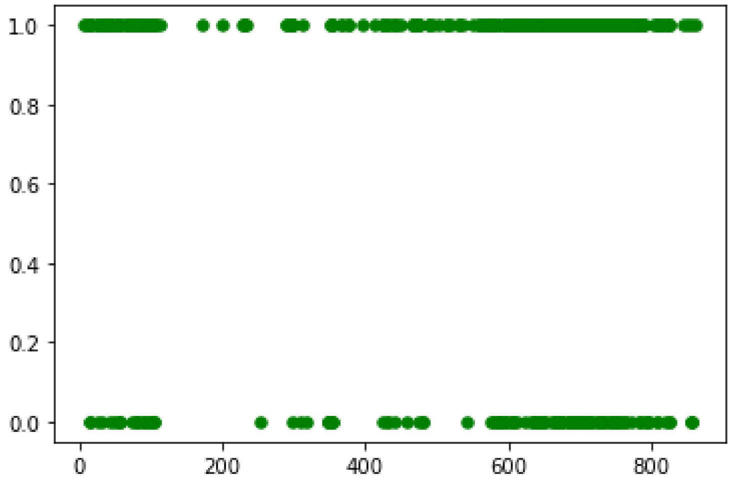
	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthdec	monthfeb	mon
0	7	0	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	...	0	0	
1	10	5	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	...	0	0	
2	10	2	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	...	0	0	
3	7	0	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	...	0	0	
4	7	3	89.3	51.3	102.2	9.6	11.4	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

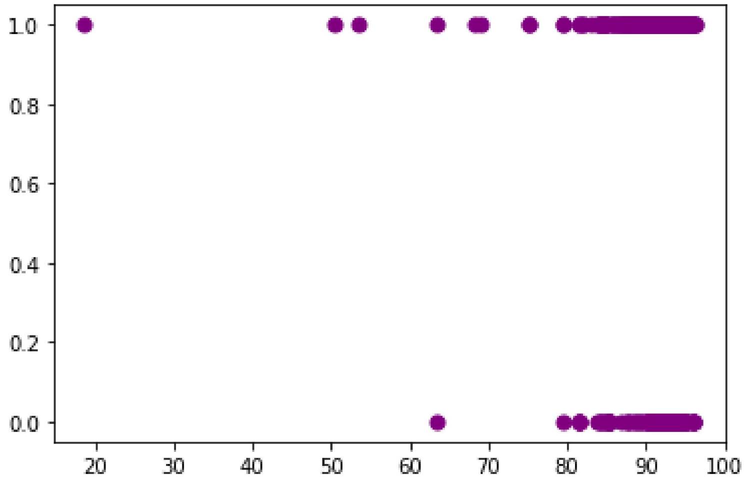
In [12]:
```
y
```

Out[12]:
```
0      1
1      1
2      1
3      1
4      1
      ..
512    0
513    0
514    0
515    1
516    1
Name: size_category, Length: 517, dtype: int32
```

In [77]:
```
plt.scatter(forest_data['DC'], y, s=30 , alpha = 1, c= 'green')
plt.show()
```



In [68]:
```
plt.scatter(forest_data['FFMC'], y, s=50, c= 'purple')
plt.show()
```



Model Building

In [32]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import accuracy_score
```

In [33]:
```
X_train,X_test,y_train,y_test =  train_test_split(X,y,test_size=0.20, random_stat
```

In [34]:
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[34]:
```
((413, 30), (104, 30), (413,), (104,))
```

In [35]:
```
from sklearn.svm import SVC
```

Linear classification

Linear classification

```
In [43]: classifier = SVC(kernel = 'linear', C = 0.01, gamma = 0.1)
         classifier.fit(X_train, y_train)
         y_test_pred = classifier.predict(X_test)
```

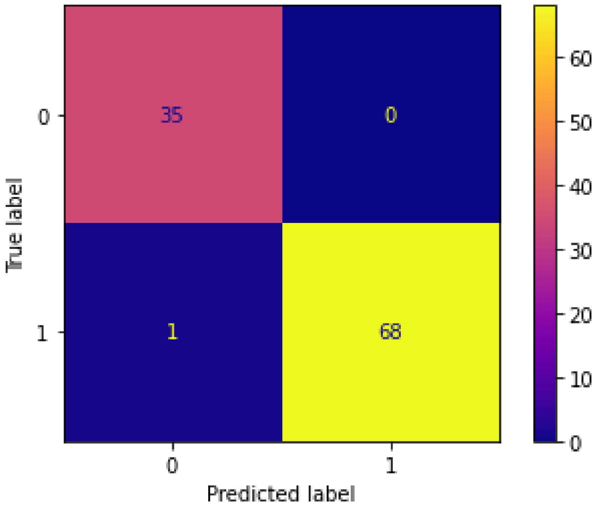
```
In [44]: accuracy_score(y_test, y_test_pred)

Out[44]: 0.9903846153846154
```

```
In [45]: confusion_matrix(y_test, y_test_pred)

Out[45]: array([[35,  0],
                [ 1, 68]], dtype=int64)
```

```
In [46]: plot_confusion_matrix(classifier, X_test, y_test, cmap = 'plasma')
         plt.show()
```



Non- Linear classification

```
In [79]: rbf_classifier = SVC(kernel='rbf',C= 0.01,gamma=0.1)
         rbf_classifier
         rbf_classifier.fit(X_train, y_train)
         y_test_pred = rbf_classifier.predict(X_test)
```

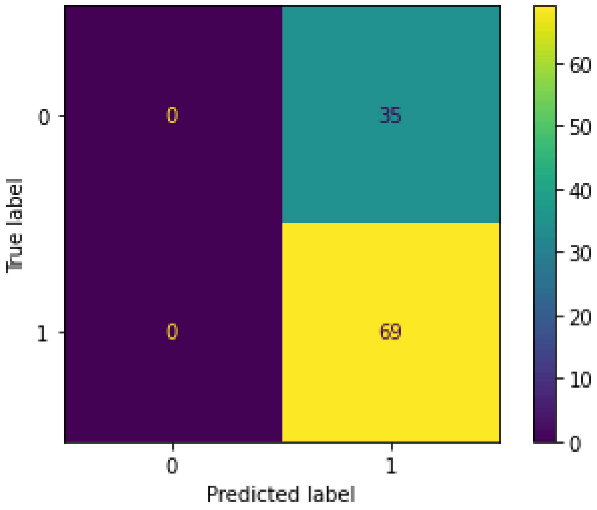
```
In [80]: accuracy_score(y_test,y_test_pred)

Out[80]: 0.6634615384615384
```

```
In [81]: confusion_matrix(y_test,y_test_pred)

Out[81]: array([[ 0, 35],
                [ 0, 69]], dtype=int64)
```

In [82]:
```
plot_confusion_matrix(classifier, X_test, y_test, cmap = 'viridis')
plt.show()
```



Scaling the data.

In [52]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_X = scaler.fit_transform(X)
```

In [53]:
```
X_train,X_test,y_train,y_test= train_test_split(scaled_X, y, test_size=0.20, rand
```

In [54]:
```
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

Out[54]:
```
((413, 30), (413,), (104, 30), (104,))
```

In [55]:
```
X_train
```

Out[55]:
```
array([[ 0.28422225,  1.69668174, -0.08063453, ..., -0.04402255,
        -0.17285971, -0.70608125],
       [ 0.05533922,  0.65674759,  0.42709293, ..., -0.04402255,
        -0.17285971, -0.70608125],
       [ 1.19975437,  0.13678051, -0.18943327, ..., -0.04402255,
        -0.17285971,  1.41626761],
       ...,
       [-1.31795895, -1.42312073, -1.38621943, ..., -0.04402255,
        -0.17285971, -0.70608125],
       [-1.08907592,  0.65674759,  0.22762857, ..., -0.04402255,
        -0.17285971, -0.70608125],
       [-0.63130986, -0.90315365, -1.07795633, ..., -0.04402255,
        -0.17285971, -0.70608125]])
```

In [56]:
```
X_test
```

Out[56]:
```
array([[-0.17354381,  0.13678051,  0.59029104, ..., -0.04402255,
        -0.17285971, -0.70608125],
       [-1.08907592, -0.38318657,  0.51775855, ..., -0.04402255,
        -0.17285971, -0.70608125],
       [ 1.19975437,  0.65674759,  0.08256358, ..., -0.04402255,
        -0.17285971,  1.41626761],
       ...,
       [ 1.19975437, -1.42312073,  0.26389482, ..., -0.04402255,
        -0.17285971,  1.41626761],
       [ 1.19975437,  0.65674759,  0.31829419, ..., -0.04402255,
        -0.17285971,  1.41626761],
       [ 1.19975437,  1.69668174,  0.35456044, ..., -0.04402255,
        -0.17285971,  1.41626761]])
```

In [ ]:

In [57]:
```
classifier_1 = SVC(C=0.1, gamma = 0.1, kernel='rbf')
classifier_1
classifier_1.fit(X_train, y_train)
y_test_pred = classifier.predict(X_test)
```

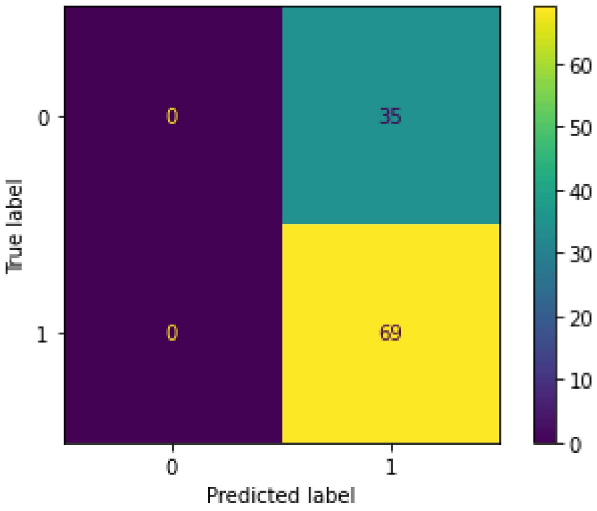
In [58]:
```
accuracy_score(y_test,y_test_pred)
```

Out[58]:
```
0.6634615384615384
```

In [59]:
```
confusion_matrix(y_test,y_test_pred)
```

Out[59]:
```
array([[ 0, 35],
       [ 0, 69]], dtype=int64)
```

In [98]:
```
plot_confusion_matrix(classifier, X_test, y_test, cmap = 'viridis')
plt.show()
```



In [ ]: