In [1]:
```
from sklearn.datasets import make_classification

```

In [2]:
```
X,y=make_classification(n_samples=1000, n_features=10, n_classes=2, random_state=2)
```

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

Out[3]:
```
(1000, 10)
```

In [4]:
```
y.shape
```

Out[4]:
```
(1000,)
```

In [5]:
```
X[0]
```

Out[5]:
```
array([-3.10242081, -0.07342322,  0.41972891,  0.96616393, -0.24918304,
       -0.60499269,  0.34970061, -2.21131549, -1.7235762 ,  2.27126308])
```

In [6]:
```
y[0]
```

Out[6]:
```
1
```

In [7]:
```
y
```

...

In [8]:
```
from sklearn.model_selection import train_test_split

```

In [9]:
```
xtrain,xtest,ytrain,ytest = train_test_split(X,y,test_size=0.3, random_state=1)
```

In [10]:
```
xtrain.shape,xtest.shape,ytrain.shape,ytest.shape
```

Out[10]:
```
((700, 10), (300, 10), (700,), (300,))
```

In [11]:
```
from sklearn.linear_model import LogisticRegression
```

In [12]:
```
model=LogisticRegression()
```

In [13]:
```
model.fit(xtrain,ytrain)
```

Out[13]:
```
LogisticRegression()
```

In [14]:
```
y_probs = model.predict_proba(xtest)[:,1]
```

In [15]:
```
y_probs

```

...

In [16]:
```
y_pred=model.predict(xtest)
```

In [17]:
```
from sklearn.metrics import roc_auc_score,accuracy_score,precision_score,recall_score,f1_scor
```

In [18]:
```
roc_auc = roc_auc_score(ytest,y_pred)#receiver operating curve    area under curve
#roc and auc are used to evalute the performance of binary classifers
```

In [19]:
```
roc_auc
```

Out[19]:
```
0.8955506117908787
```

In [20]:
```
accuracy = accuracy_score(ytest,y_pred)
```

In [21]:
```
accuracy
```

Out[21]: 0.8966666666666666

In [22]:
```
recall = recall_score(ytest,y_pred)
```

In [31]:
```
recall
```

Out[31]: 0.8620689655172413

In [32]:
```
precission = precision_score(ytest,y_pred)
```

In [33]:
```
precission
```

Out[33]: 0.9191176470588235

In [34]:
```
f1= f1_score(ytest,y_pred)
```

In [35]:
```
f1
```

Out[35]: 0.889679715302491

In [36]:
```
from matplotlib import pyplot as plt
```

In [38]:
```
from sklearn.metrics import roc_curve,auc
```

In [39]:
```
fpr,tpr,threshold = roc_curve(ytest,y_probs)
```

In [40]:
```
fpr
```

...

In [41]:
```
tpr
```

...

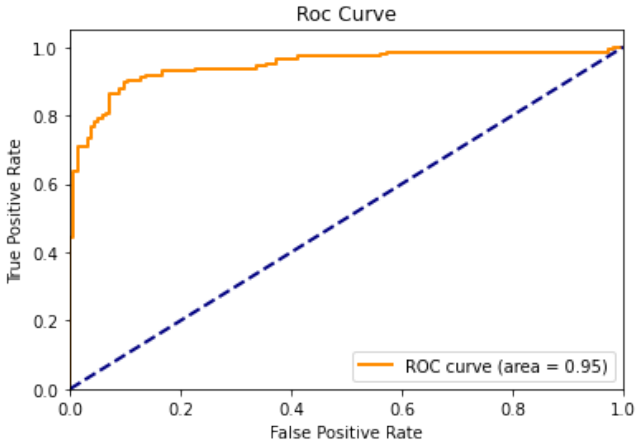
In [42]:
```
threshold
```

...

In [49]:
```
#ploting the roc

roc_auc = auc(fpr, tpr)
# Plot the ROC curve
plt.figure()
plt.plot(fpr, tpr, color='darkorange' ,lw = 2,label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy' , lw = 2, linestyle ='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Roc Curve')
plt.legend()
plt.show()


```



In [ ]:
```
1
```