Classification Metrics

In [ ]:
```
import numpy as np
import pandas as pd
from sklearn import preprocessing
```

In [ ]:
```
data=pd.read_csv('payment_fraud.csv')
data
```

In [ ]:
```
data.head(2)
```

In [ ]:
```
data.label.value_counts(normalize=True)
```

In [ ]:
```
data.shape
```

In [ ]:
```
x=data.iloc[:,1:6]
y=data.iloc[:,5]
```

In [ ]:
```
x.head(3)
```

In [ ]:
```
y.head(3)
```

In [ ]:
```
#data['paymentMethod'].replace(['paypal','storecredit','creditcard'],[2,1,0],inplace=Tru
#data.head(3)
```

In [ ]:
```
result=data.dtypes
print(result)
```

In [ ]:
```
minmax=preprocessing.MinMaxScaler(feature_range=(0,1))
minmax.fit(x).transform(x)
```

In [ ]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2)
```

In [ ]:
```
x_train.head(10)
```

In [ ]:
```
y_train = y_train.astype('int')
y_test = y_test.astype('int')
```

In [ ]:
```
x_test.tail(10)
```

In [ ]:
```
y_test.head(2)
```

In [ ]:
```
from sklearn.neighbors import KNeighborsClassifier
clf = KNeighborsClassifier()
clf.fit(x_train,y_train)
```

In [ ]:
```
#Predict on test data
y_pred=clf.predict(x_test)
```

In [ ]:
```
print("Actual fraud : ")
print(y_test.values)
```

In [ ]:
```
print("\nPredicted fraud : ")
print(y_pred)
```

In [ ]:
```
from sklearn.metrics import accuracy_score, recall_score, roc_auc_score, confusion_matri

print("\nAccuracy score: %f" %(accuracy_score(y_test,y_pred) * 100))
```

In [ ]:
```
print("precision_score : %f" %(precision_score(y_test, y_pred) * 100))
```

In [ ]:
```
print("Recall score : %f" %(recall_score(y_test, y_pred) * 100))
```

In [ ]:
```
print(confusion_matrix(y_test, y_pred))
```

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

In [ ]:
```
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_pred, y_test)
roc_auc = auc(fpr, tpr)

#plt.figure()
plt.plot(fpr, tpr, color='darkorange',
         label='ROC curve (area = %0.3f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy',  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('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```

In [ ]:
```
print("ROC score : %f\n" %(roc_auc_score(y_test, y_pred) * 100))
```

In [ ]:
```

```