```
import pandas as pd


data = {'feature': [1,2,3,6,6,7,10,11],
        'Class':    [1,1,2,2,2,1,1,1]}


df = pd.DataFrame(data)
df
```

	feature	Class
0	1	1
1	2	1
2	3	2
3	6	2
4	6	2
5	7	1
6	10	1
7	11	1

```
from sklearn.model_selection import train_test_split
# We are interested to consider 50 % of the total data as Testing data

X_train, X_test, y_train, y_test = train_test_split(df.drop('Class',axis=1),
                                                    df['Class'], test_size=0.50,
                                                    random_state=101)


from sklearn.neighbors import KNeighborsClassifier
clf = KNeighborsClassifier(n_neighbors = 3)
clf.fit(X_train, y_train)
```

```
KNeighborsClassifier(n_neighbors=3)
```

```
# Predicted outputs for Test Samples
y_pred = clf.predict(X_test)
y_pred
```

```
array([1, 1, 1, 1])
```

```
#Checking the accuracy score , confusion matrix ,
from sklearn.metrics import accuracy_score,confusion_matrix
accuracy = accuracy_score(y_test,y_pred)
cm = confusion_matrix(y_test,y_pred)


# Assigning columns names
```

```
cm_df = pd.DataFrame(cm,
            columns = ['Predicted Negative', 'Predicted Positive'],
            index = ['Actual Negative', 'Actual Positive'])
# Showing the confusion matrix
cm_df
```

	Predicted Negative	Predicted Positive
Actual Negative	2	0
Actual Positive	2	0

```
## Calculating Specificity & Sensitivity

# Creating a function to report confusion metrics
def confusion_metrics (conf_matrix):
# save confusion matrix and slice into four pieces
    TP = conf_matrix[1][1]
    TN = conf_matrix[0][0]
    FP = conf_matrix[0][1]
    FN = conf_matrix[1][0]

    # calculate accuracy
    conf_accuracy = (float (TP+TN) / float(TP + TN + FP + FN))

    # calculate mis-classification
    conf_misclassification = 1- conf_accuracy

    # calculate the sensitivity
    conf_sensitivity = (TP / float(TP + FN))
    # calculate the specificity
    conf_specificity = (TN / float(TN + FP))

    # calculate precision
    conf_precision = (TN / float(TN + FP))
    # calculate f_1 score
    conf_f1 = 2 * ((conf_precision * conf_sensitivity) / (conf_precision + conf_sensitivit
    print('-'*50)
    print(f'Accuracy: {round(conf_accuracy,2)}')
    print(f'Mis-Classification: {round(conf_misclassification,2)}')
    print(f'Sensitivity: {round(conf_sensitivity,2)}')
    print(f'Specificity: {round(conf_specificity,2)}')
    print(f'Precision: {round(conf_precision,2)}')
    print(f'f_1 Score: {round(conf_f1,2)}')

confusion_metrics(cm)
```

```
--------------------------------------------------
Accuracy: 0.5
Mis-Classification: 0.5
Sensitivity: 0.0
Specificity: 1.0
Precision: 1.0
f_1 Score: 0.0
```

Colab paid products - Cancel contracts here

✓ 0s completed at 10:49 AM