Metrics that can be derived from confusion matrix

Before knowing about the metrics lets have an idea about what is a confusion matrix.Lets not get confused with it.

A confusion matrix is a table used to evaluate the performance of a machine learning classification model. It is used to summarize the actual and predicted classification results obtained by a model on a set of test data.

The matrix consists of four components: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The rows of the matrix represent the actual class labels, and the columns represent the predicted class labels.

Why is it called as confusion matrix:

The term "confusion matrix" comes from the fact that the matrix helps to identify the confusion that may arise between different classes in a classification problem.

The matrix provides a visual representation of the different types of errors that a model can make and their corresponding frequencies. This can help to identify patterns in the errors and to understand which classes are most often confused with each other.

The confusion matrix provides information that can be used to calculate various performance metrics of a classification model. Some commonly used metrics include:

1. Accuracy: Accuracy measures the proportion of correctly classified instances out of the total number of instances. It is calculated as (TP + TN) / (TP + TN + FP + FN).

   Let us take the dataset as

Name	Body Temp	Actual labels	Predicted labels
A	90	Not Covid	Not Covid
B	80	Not Covid	Not Covid
C	92	Not Covid	Not Covid
D	93	Not Covid	Not Covid
E	94	Not Covid	Covid
G	95	Not Covid	Covid
H	92	Not Covid	Covid
I	100	Covid	Covid
J	100	Covid	Covid
K	100	Covid	Not Covid

TP – machine predicts Covid as Covid= 2

TN- Machine predicts not Covid as Not Covid=4

FP- Machine predicts not Covid as Covid=3

FN-Machine predicts Covid as Not Covid=1

Therefore Accuracy=(2+4)/4=6/4=1.5

2. Precision: Precision measures the proportion of true positives among the instances that were classified as positive. It is calculated as TP / (TP + FP).=2/(3+2)=0.4
3. Recall: Recall (also known as sensitivity or true positive rate) measures the proportion of true positives among all actual positives. It is calculated as TP / (TP + FN).=2/(2+1)=0.6
4. F1-score: F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall and is often used as a summary metric for classification performance. It is calculated as 2 * (precision * recall) / (precision + recall).

   F1-score= 2*(0.4*0.6)/(0.4+0.6)=0.48

5. Specificity: Specificity (also known as true negative rate) measures the proportion of true negatives among all actual negatives. It is calculated as TN / (TN + FP).
6. False positive rate: False positive rate measures the proportion of instances that were incorrectly classified as positive out of all actual negatives. It is calculated as FP / (TN + FP).
7. False negative rate: False negative rate measures the proportion of instances that were incorrectly classified as negative out of all actual positives. It is calculated as FN / (TP + FN).

These metrics provide valuable information about the performance of a classification model, and can help to diagnose problems and improve the model's accuracy and effectiveness.

How to choose which metrics: It is completely based on the usecases. in our covid usecase,if the prediction is all covid or all not covid then it is a bad model, for other cases use case: Spam detection Model, Recall is a better choice as in recall emphasis is laid on FN. It is justified saying that a false negative is always better than false positive in a spam detection where as false positive is better in Covid detection use case,therefore precision is better than recall in this scenario.

For our understanding ,Just looted from my handson

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

#train_model
clf_0 = LogisticRegression().fit(train_x,train_y)
pred_y_0 = clf_0.predict(val_x)
accuracy_score(pred_y_0,val_y)
```

Out[47]:

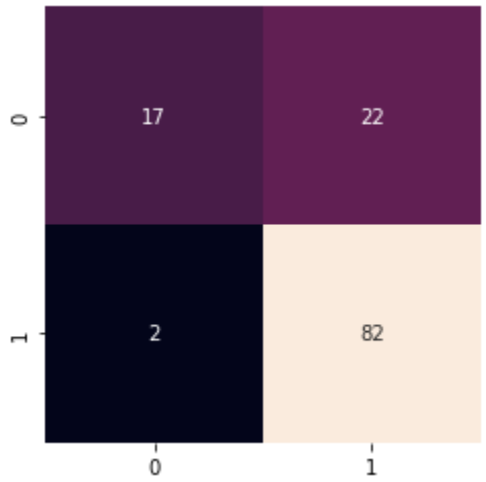
```
0.8048780487804879
```

```
from sklearn.metrics import accuracy_score,confusion_matrix
cm = confusion_matrix(val_y,pred_y_0)

sns.heatmap(cm,square = True,annot = True,cbar= False)
```

Out[48]:

<AxesSubplot:>



val_y (*****test data)

pred_y_0(*****train data)

square = True(*****to display square shape)

annot = True(to mark the axis with numerical values)

cbar= False(removes the colour bar)

Just like precision we can also calculate the other metrics too:

accuracy = accuracy_score(y_test, y_pred)

precision = precision_score(y_test, y_pred)

recall = recall_score(y_test, y_pred)

f1 = f1_score(y_test, y_pred)

Additional piece of information:

The range of Accuracy,Precision,Recall and F1 score ranges from 0 to 1

0 being the least and 1 being the max.