Performance metrices for classification based problem statement:

For classifiaction based problem statement we generally use below metrices for measuring the performance of respective classification based learning model:

- Accuracy
- Precision
- Recall
- F-Beta score

Note: These metrices are only used in case of classification problem statement and are not applicable in case regression probelm statement.

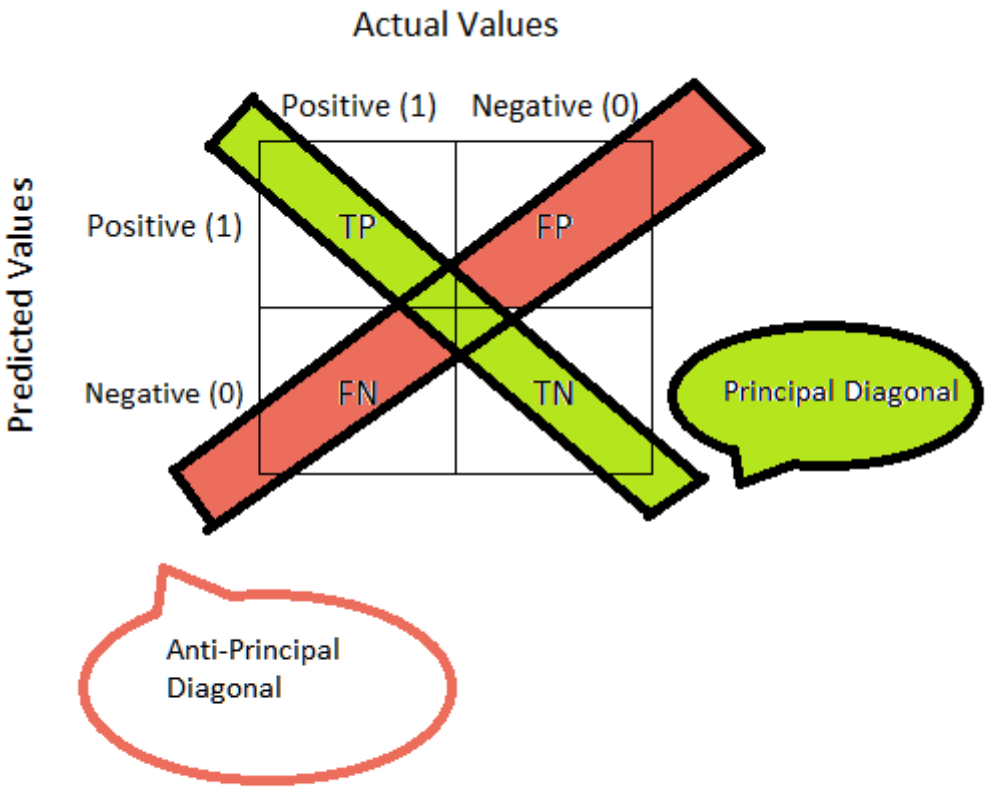
Before discussing about these metrices let's discuss about confusion matirx.

Confusion matrix:

Confusion matrix is the matrix that can be used to visualize following 4 parameters in case of classifiaction problem statement:

- **True positive:** If the classification based learning model makes true prediction by classifying any record as positive that actually belongs to the positive class, then that case is called True positive case.
- **True negative:** If the classification based learning model makes true prediction by classifying any record as negative that actually belongs to the negative class, then that case is called True negative case.
- **False positive:** If the classification based learning model makes false prediction by classifying any record as positive that actually belongs to the negative class, then that case is called False positive case.
- **False negative:** If the classification based learning model makes false prediction by classifying any record as negative that actually belongs to the positive class, then that case is called False negative case.

Please note that confusion matrix is a square matrix (i.e. number of row = number of column) where *Principal Diagonal* represents True predictions and *Anti-principal Diagonal* represents False predictions.



Now, using parameters (TP,TN,FP,FN) defined in confusion matrix we can discuss the different Performance metrices used in classification problem statement.

1. Accuracy:

- Out of all the classifications made by any classification based learning model, the number of classifiactions that were correct is called accuracy.

That is, **Accuracy = (TP+TN) / (TP+TN+FP+FN)**

- **Understanding accuracy in terms if data imbalance:** Suppose, we are provided with around 1000 datapoints from which around 900 are classified as 1 and only 100 are classified as 0. Due to this data imbalance trained model will most likely behave in a biased manner that is if model is passed with any new data that acctually belongs to the to 0 category, model may falsely predict it as 1 as model is trained with a lot of 1s. If we calculate the

accuracy in this case it will come around 90%. Although accuracy is high but in this case where there is huge data imbalance accuracy cannot be used as an only parameter to determine the performance of the ML model. That's why we are also using other learning parameters such as Precision and Recall to judge the performance of model instead of just relying on accuracy score.

2. Precision:

- Out of all the actual values the number datapoints that are correctly predicted is called Precision.
- That is, **Precision = (TP) / (TP+FP)**
- Cases where false positive is important, we use Precision where we want to reduce false positive cases.

3. Recall:

- Out of all the predicted values the number of datapoints that are correctly predicted with actual values is called Recall.
- That is, **Recall = (TP) / (TP+FN)**
- Cases where false negative is important, we use Recall where we want to reduce false negative cases.

Let's understand Precision and Recall using real life use cases.

Use Case1: Email spam classification(Gmail, Outlook etc.):

- Suppose email is actually SPAM and learning model predicted it as SPAM. Then this will be True positive case which seems fine and hence no special attention needed in this case.
- Suppose email is actually NOT SPAM and learning model predicted it as NOT SPAM. Then this will be True Negative case which seems fine and hence no special attention needed in this case as well.
- Suppose email is actually SPAM and learning model predicted it as NOT SPAM. Then this will be False Negative case. In this case some of the SPAM email may get delivered inside our regular INBOX which we can manage. This case seems to be less important as compared to case dicussed below.
- Suppose email is actually NOT SPAM and learning model predicted it as SPAM. Then this will be False Postive case. It's obvious that if our learning model for spam classification is lying under this category then this would be a big blunder. One can miss their important emails as it will be moved to JUNK folder since they are classified as SPAM. Therefore, we should attempt to reduce the False Postive cases for Email classification use case.
- Hence, Precision becomes an important parameter to judge performance of our Email Spam classification model.

Use Case2: Deciding whether a person is having Diabetes or not?

- Suppose a person is actually HAVING Diabetes and learning model predicted it as Diabetes. Then this will be True positive case which seems fine and hence no special attention is needed in this case.

- Suppose a person is actually NOT HAVING Diabetes and learning model predicted it as NO Diabetes. Then this will be True Negative case which seems fine and hence no special attention needed in this case as well.
- Suppose a person is actually NOT HAVING Diabetes and learning model predicted it as Diabetes. Then this will be False Postive case. In this case a person will may be start taking neccesary precautions in order to counter wrongly diagonsed Diabetes symptoms, that will not have have adverse effect on his/her health. This case seems to be less important as compared to case dicussed below.
- Suppose a person is actually HAVING Diabetes and learning model predicted it as NO Diabetes. Then this will be False Negative case. It's obvious that if our learning model for Diabetes prediction is lying under this category then this would be a big blunder. A person with Diabetes will not get diagonsed which will impact his/her health. Therefore, we should attempt to reduce the False Negative cases for Diabetes prediction use case.
- Hence, Recall becomes an important parameter to judge performance of our Diabetes prediction based classification model.

Please note that generally:

1. Cases where **Protection of People** is an important factor we attempt to reduce False Negative cases.
2. Whereas, the cases where **Protection of Companies** is an important factor we attempt to reduce False Positive cases.

4. F-Beta score:

- Precision and Recall are valuable metrics that are widely used across the industry. Still, they have a massive drawback. They produce two values that must be first analyzed separately and then together to better understand an algorithm’s performance.
- So, to overcome this disadvantage, Data Scientists came up with a way to combine Precision and Recall into an aggregate quality metric.
    - To define the term, in Machine Learning, the F-beta score (or F-measure) is a Classification metric featuring a harmonic mean of Precision and Recall.
    - It is given by the below formula where Beta parameter represents the weight of Precision in the metric:

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

- In general, there are three most common values for the beta parameter:

    1. **F0.5 score** (beta = 0.5): Such a beta makes a Precision value more important than a Recall one. In other words, it focuses on minimizing False Positives than minimizing False Negatives;
    2. **F1 score** (beta = 1): True harmonic mean of Precision and Recall. In the best-case scenario, if Precision and Recall are equal to 1, the F-1 score will also be equal to 1. F1 score is the one that is most commonly used;
    3. **F2 score** (beta = 2): Such a beta makes a Recall value more important than a Precision one. In other words, it focuses on minimizing False Negatives than minimizing False Positives.
- For calculating **F-Beta score** in case of **Multi Class Classification** we use below approaches:
    1. Micro
    2. Macro
    3. Weighted
- For any beta parameter, the best possible value for F-measure is 1, and the worst is 0.
- Follow this link to understand how **Micro F1 score** and **Macro F1 score** is calculated: *https://hasty.ai/docs/mp-wiki/metrics/f-beta-score#:~:text=In%20the%20F%2Dbeta%20score,the%20measured%20value%2C%20the%20better*
- To understand more about F-Beta calculation approaches in case Multi class classification follow this sklearn.metrices.fbeta_score documentation: *https://scikit-learn.org/stable/modules/generated/sklearn.metrics.fbeta_score.html?highlight=fbeta#sklearn.metrics.fbeta_score*

All the sklearn methods needed for executing Performance metrices in case of classification problem statement can be found under this sklearn.metrices documentation:

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics

IMPLEMENTATION

In [30]:
```
# Importing pythonic libraries needed for implmentation

from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import fbeta_score
from sklearn.metrics import f1_score
```

In [13]:
```
# y_true represents the Ground truth (correct) target values.
# Whereas, y_pred represents the Estimated targets as returned by a classifier.

y_true = [0, 1, 2, 0, 1, 2, 1, 2, 0, 0, 2]
y_pred = [0, 2, 1, 0, 0, 1, 0, 2, 1, 0, 2]
```

In [18]:
```
# Creating Confusion matrix
cm=confusion_matrix(y_true, y_pred)
print("Confusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[3 1 0]
 [2 0 1]
 [0 2 2]]
```

In the binary case, we can extract true positives, etc as follows:

In the binary case, we can extract true positives, etc as follows:

tn, fp, fn, tp = confusion_matrix([0, 1, 0, 1], [1, 1, 1, 0]).ravel()

(tn, fp, fn, tp)

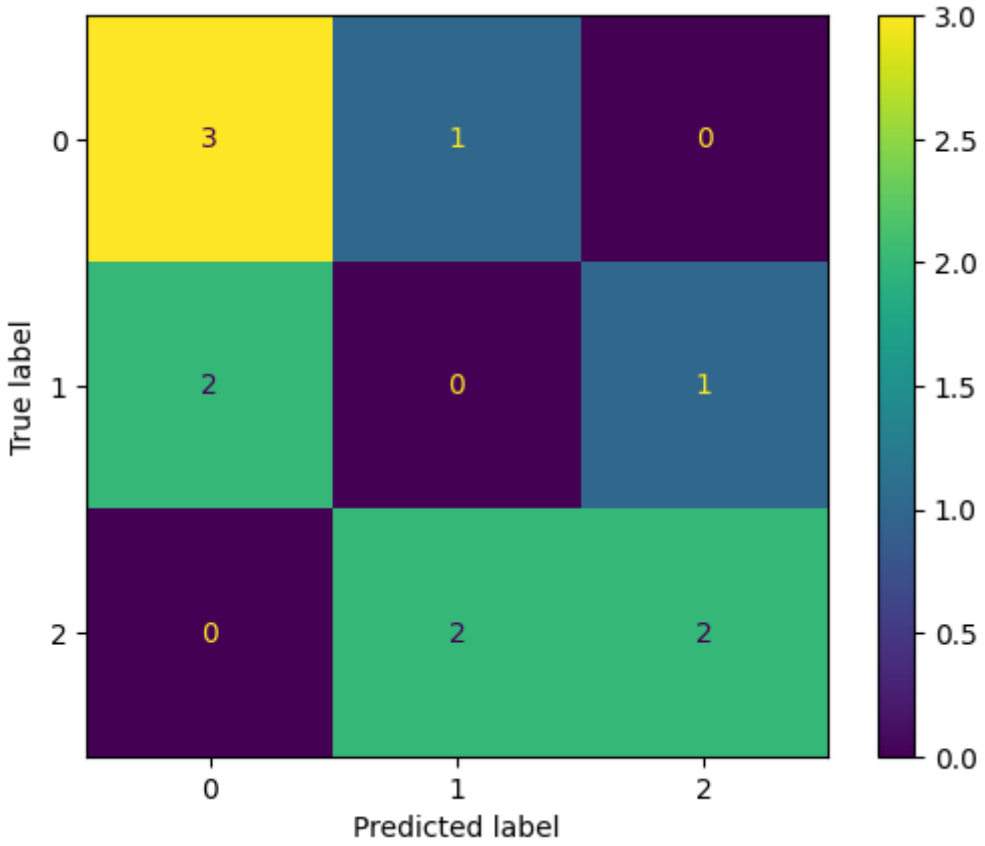
Output:

(0, 2, 1, 1)

Please note TP, TN, FP and FN cannot be computed in case of Multi class classification since there are more than 2 classes. It is applicable only in case of binary classification where a class can be represented as Positive class and another class as Negative class.

In [15]:
```
# Visualizing the Confusion matrix more accurately
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()
```

Out[15]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1e6cafe52d0>
```



In [23]:
```
#Calculating the Accuracy score

accuracy = accuracy_score(y_true, y_pred)
print("Accuracy score is:\n", round(accuracy*100,4))
```

```
Accuracy score is:
 45.4545
```

In [27]:
```
#Calculating the Precision score. Since, this is multi label classification we are using macro approach for calculation. One
#use other approaches as well like; micro, weighted etc.

precision = precision_score(y_true, y_pred, average="macro")
print("Precision score is:\n", round(precision*100,4))
```

```
Precision score is:
 42.2222
```

In [29]:
```
#Calculating the Recall score. Since, this is multi label classification we are using macro approach for calculation. One car
#use other approaches as well like; micro, weighted etc.

recall = recall_score(y_true, y_pred, average="macro")
print("Recall score is:\n", round(recall*100,4))
```

```
Recall score is:
 41.6667
```

In [33]:
```
# Calculating the metrics and printing the result for F1 and F0.5 scores

f1 = f1_score(y_true, y_pred, average='macro')
f_half = fbeta_score(y_true, y_pred, average='macro', beta=0.5)
print("F1 score is:\n", round(f1*100,4))
print("F0.5 score is:\n", round(f_half*100,4))
```

```
F1 score is:
 41.2698
F0.5 score is:
 41.6667
```

In [41]:
```
# Summarized report

print("SUMMARIZED REPORT\n")
print("Accuracy score is:\n", round(accuracy*100,4))
print("="*50+'\n')
print("Precision score is:\n", round(precision*100,4))
print("="*50+'\n')
print("Recall score is:\n", round(recall*100,4))
print("="*50+'\n')
print("F1 score is:\n", round(f1*100,4))
print("="*50+'\n')
print("F0.5 score is:\n", round(f_half*100,4))
print("="*50)
```

```
SUMMARIZED REPORT

Accuracy score is:
 45.4545
==================================================

Precision score is:
 42.2222
==================================================

Recall score is:
```

```
 41.6667
================================================

F1 score is:
 41.2698
================================================

F0.5 score is:
 41.6667
================================================
```

References:

1. https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics

2. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.fbeta_score.html?highlight=fbeta#sklearn.metrics.fbeta_score

3. https://hasty.ai/docs/mp-wiki/metrics/f-beta-score#:~:text=In%20the%20F%2Dbeta%20score,the%20measured%20value%2C%20the%2

In [ ]: