Classification Metrics

How to check the Classification you have applied is accurate or not by using Classification Metrics

video — video — video

github

Accuracy

Types: Accuracy, Confusion Matrix, Precision, Recall, F1 score

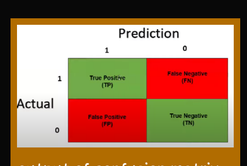
$$ACCURACY = \frac{\text{Number of correct Prediction}}{\text{Number of total prediction}}$$

code:

```
from sklearn.metrics import accuracy_score
# Initialize the logistic regression model
logreg = LogisticRegression()
# Train the model on the training data
logreg.fit(X_train, y_train)
# Make predictions on the testing data
y_pred = logreg.predict(X_test)
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

- How much Accuracy is good — It depends on problem
- Problem with Accuracy — It tell us your system is 90% accurate but does not tell what type of error is coming in rest of the 10%
- When accuracy is misleading — For imbalance data set

Confusion Matrix



output of confusion matrix

- Type 2 error = False Negative
- Type 1 error = False Positive

For binary Class Classification (output have only 2 class -> 0,1)

code:

```
from sklearn.metrics import confusion_matrix
y_pred = logreg.predict(X_test)
array([[26,  6],
       [ 0, 29]])
print(confusion_matrix(y_test, y_pred))
```

For Multi Class Classification ((output have more then 2 class -> 0,1,3))

Actual\Predicted	Cat	Dog	Rabbit
Cat	42	8	2
Dog	10	35	5
Rabbit	3	6	45

output of confusion matrix

Precision , Recall and F1 score

Precision & Recall

Precision

Out of all the positive prediction , how much of them truly belong their

Formula: $Precision = TP/(TP+FP)$

Here
TP=True Positive
FP=False Positive

Recall

Formula: $Recall = TP/(TP+FN)$

Here
TP=True Positive
FP=Fasle Positive

How much of a categories cases get correctly classified

When to use Precision and Recall

- Case 1 - Email spam classifier
  - Let suppose accuracy of the two model you are comparing are the same , but they have difference in Type 1 and Type 2 error
  - Type 1 error : You have predicted a non spam mail as spam
    Type 2 error : You have predicted a spam mail as non spam
  - Here type 1 error is more dangerous so we use precision
- Case 2 - Cancer Detector
  - Let suppose accuracy of the two model you are comparing are the same , but they have difference in Type 1 and Type 2 error
  - Type 1 error : You have predicted a non cancer patients as cancer
    Type 2 error : You have predicted a cancer patients as non patients
  - Here type 2 error is more dangerous so we use Recall

For binary Class Classification (output have only 2 class -> 0,1)

code:

```
from sklearn.metrics import recall_score,precision_score
y_pred = logreg.predict(X_test)
print("Precision - ",precision_score(y_test,y_pred1))
print("Recall - ",recall_score(y_test,y_pred1))
```

F1 score

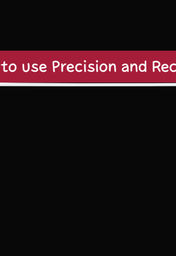
Formula: $\frac{2PR}{P+R}$

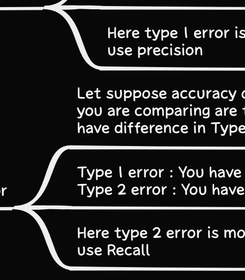
P= Precision
R= Recall

When to use: When both type 1 and type 2 are equally important -> Example = Dog and cat classifier

code:

```
from sklearn.metrics import f1_score
y_pred = logreg.predict(X_test)
print("F1 score - ",f1_score(y_test,y_pred1))
```





Implementation

For Multi Class Classification ((output have more then 2 class -> 0,1,3))

```
from sklearn.metrics import precision_score,recall_score,f1_score
```

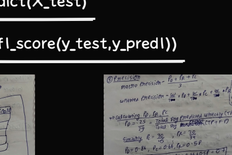
way 1:

- weighted
  - `precision_score(y_test,y_pred1,average='weighted')`
  - `recall_score(y_test,y_pred1,average='weighted')`
  - `f1_score(y_test,y_pred1,average='weighted')`
- macro
  - `precision_score(y_test,y_pred1,average='macro')`
  - `recall_score(y_test,y_pred1,average='macro')`
  - `f1_score(y_test,y_pred1,average='macro')`

	precision	recall	f1-score	support
0	0.95	0.96	0.96	821
1	0.95	0.98	0.96	962
2	0.91	0.90	0.91	829
3	0.90	0.88	0.89	864
4	0.93	0.94	0.94	856
5	0.87	0.84	0.85	729
6	0.93	0.94	0.94	839
7	0.91	0.93	0.92	873
8	0.86	0.88	0.87	793
9	0.89	0.88	0.89	834
accuracy			0.91	8400
macro avg	0.91	0.91	0.91	8400
weighted avg	0.91	0.91	0.91	8400

way 2 (Better way):

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred1))
```



Interpretation