

# Confusion Matrix

Suppose you have following 16 labelled data that you feed to a classification ML model.

Feature_1 (ht)	Feature_2 (wt)	Actual_Target
----------------	----------------	---------------

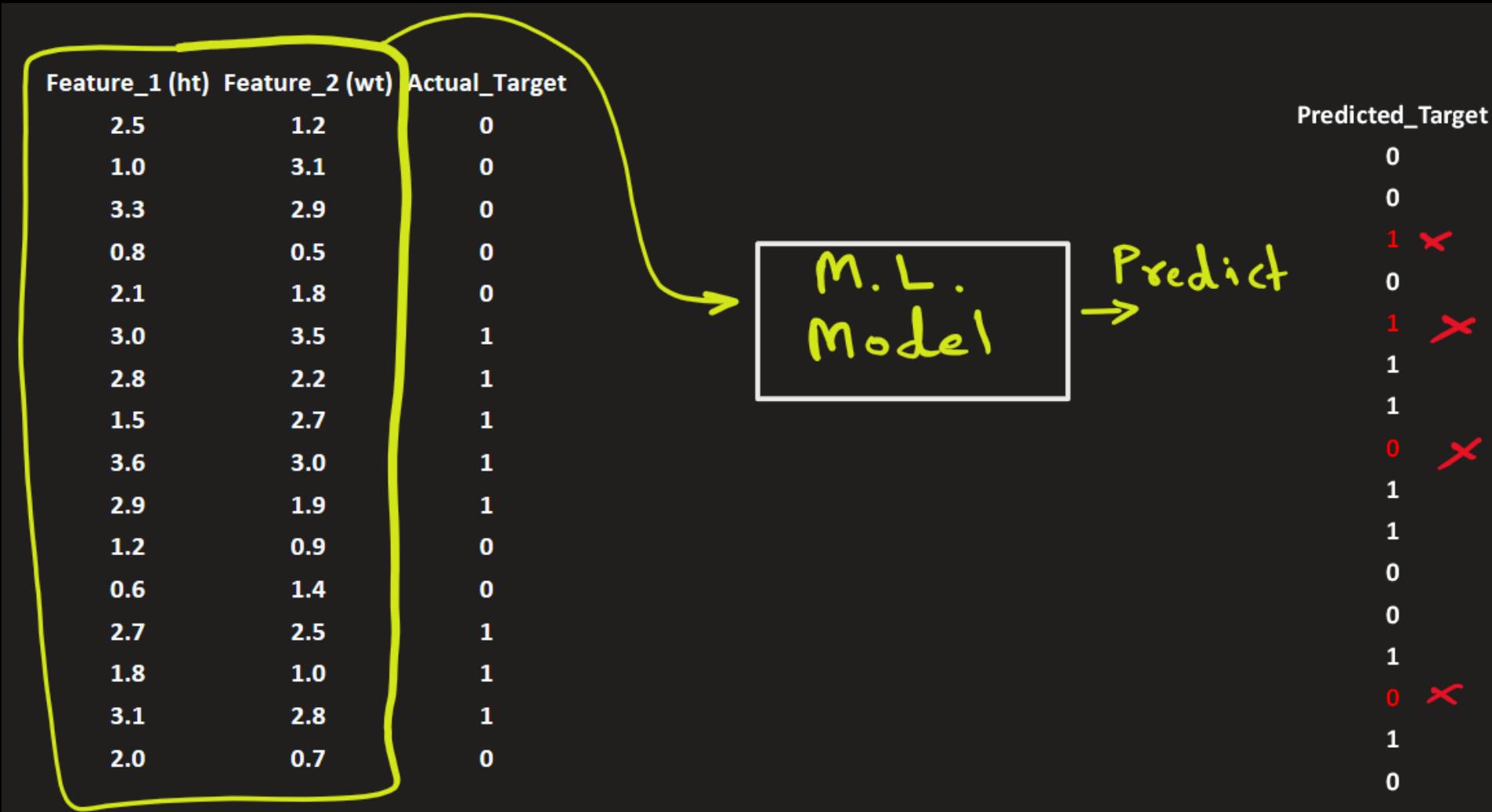
2.5	1.2	0
1.0	3.1	0
3.3	2.9	0
0.8	0.5	0
2.1	1.8	0
3.0	3.5	1
2.8	2.2	1
1.5	2.7	1
3.6	3.0	1
2.9	1.9	1
1.2	0.9	0
0.6	1.4	0
2.7	2.5	1
1.8	1.0	1
3.1	2.8	1
2.0	0.7	0

# Confusion Matrix

Suppose you have following labelled data that you feed to a classification ML model.

It makes a prediction.

You compare the predicted target with the actual target. You find 4 misclassified targets



# Confusion Matrix



Feature_1 (ht)	Feature_2 (wt)	Actual_Target	Predicted_Target
2.5	1.2	0	0
1.0	3.1	0	0
3.3	2.9	0	1
0.8	0.5	0	0
2.1	1.8	0	1
3.0	3.5	1	1
2.8	2.2	1	1
1.5	2.7	1	0
3.6	3.0	1	1
2.9	1.9	1	1
1.2	0.9	0	0
0.6	1.4	0	0
2.7	2.5	1	1
1.8	1.0	1	0
3.1	2.8	1	1
2.0	0.7	0	0

You create a matrix that show how many were classified correctly and incorrectly:

		Predicted	
		0	1
		0	6
Actual		0	2
1		2	6

The on-diagonal were classified correctly.  
The off-diagonal were misclassified.  
This matrix is called confusion matrix.



# Confusion Matrix



Confusion Matrix tells you how many were

- correctly classified (i.e. True Negative and True Positive)
- incorrectly classified (i.e. False Negative and False Positive)

		Predicted	
		<u>0</u>	<u>1</u>
Actual	<u>0</u>	TN = 6	FP = 2
	<u>1</u>	FN = 2	TP = 6

Here,  
TN = True Negative,  
FP = False Positive,  
etc..



# Confusion Matrix

Or you can see the percentage instead of raw numbers.  
Here you divide by total records, 16.

		Predicted	
		<u>0</u>	<u>1</u>
Actual	<u>0</u>	$6/16 = 37.5\%$	$2/16 = 12.5\%$
	<u>1</u>	$2/16 = 12.5\%$	$6/16 = 37.5\%$



# Confusion Matrix



Confusion matrix is not about accuracy — it is about **understanding errors**.

		Predicted	
		<u>0</u>	<u>1</u>
Actual	<u>0</u>	$6/16 = 37.5\%$	$2/16 = 12.5\%$
	<u>1</u>	$2/16 = 12.5\%$	$6/16 = 37.5\%$



# Confusion Matrix

```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

y_actual = [ 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0] # Actual target values
y_pred   = [ 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0] # Predicted target values

cm = confusion_matrix(y_actual, y_pred)

print(cm)
```

```
[[6 2]
 [2 6]]
```

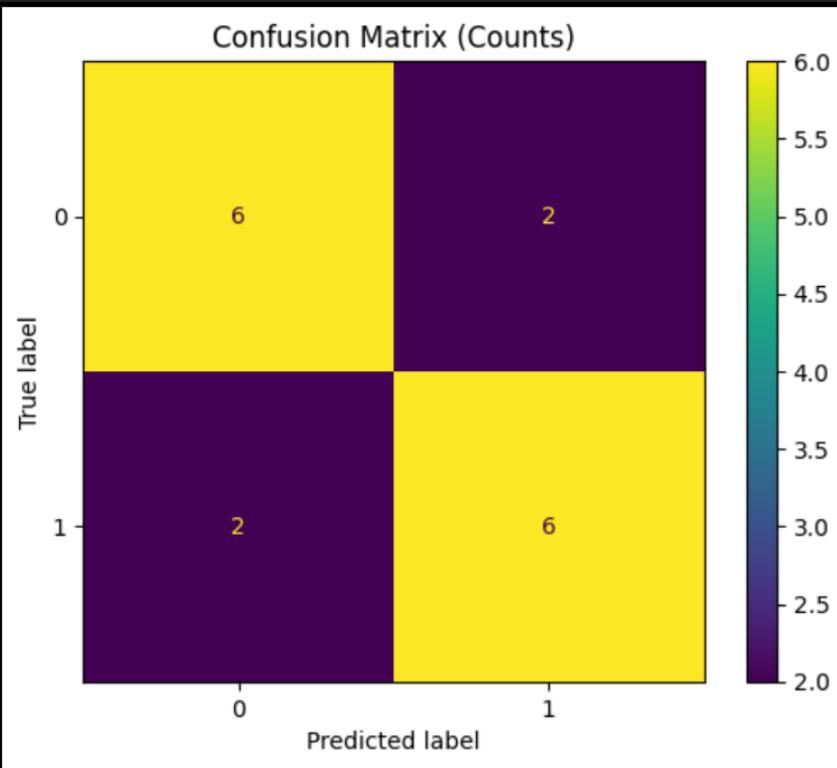
```
cm_normalized = confusion_matrix(y_actual, y_pred, normalize='all')
print(cm_normalized)
```

```
[[0.375 0.125]
 [0.125 0.375]]
```

# Confusion Matrix

```
import matplotlib.pyplot as plt
```

```
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()
plt.title("Confusion Matrix (Counts)")
plt.show()
```





# Confusion Matrix: 4 classes

```
y_actual = [ 0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3]  
y_pred   = [ 0, 1, 0, 0, 1, 1, 2, 1, 2, 2, 2, 3, 3, 0, 0, 3]
```

```
cm = confusion_matrix(y_actual, y_pred)  
print(cm)
```

```
[[3 1 0 0]  
 [0 3 1 0]  
 [0 0 3 1]  
 [2 0 0 2]]
```

# Confusion Matrix: 4 classes



# Confusion Matrix



Fcsdf



# Heading Goes Here



Fhdsklf  
Fjdsklf  
Fjskldf

