Instituto de Computação

UNIVERSIDADE ESTADUAL DE CAMPINAS

Capacitação profissional em tecnologias de Inteligência Artificial

Machine Learning Overview

Prof. Edson Borin

https://www.ic.unicamp.br/~edson

Institute of Computing - UNICAMP

Model evaluation (II)
Performance measures

Data collection → Data preprocessing → Feature extraction, selection, and scaling → Model training → **Model evaluation** → Model deployment and integration

ML Process - Model evaluation

Performance measures: **regression tasks**

Usually distance between $h_\theta(x^{(i)})$ and $y^{(i)}$

- RMSE: Root Mean Square Error

$$\mathrm{RMSE}(\mathrm{X},h_\theta) = \sqrt{\frac{1}{\mathrm{m}}\sum_{i=1}^{\mathrm{m}} (h_\theta(x^{(i)}) - y^{(i)})^2}$$

- MAE: Mean Absolute Error

$$\mathrm{MAE}(\mathrm{X},h_\theta) = \frac{1}{\mathrm{m}}\sum_{i=1}^{\mathrm{m}} | h_\theta(x^{(i)}) - y^{(i)} |$$

ML Process - Model evaluation

Performance measures: **regression tasks**

Usually distance between $h_\theta(x^{(i)})$ and $y^{(i)}$

- MSE: Mean Square Error

RMSE = $\sqrt{\text{MSE}}$

$$\text{MSE}(X, h_\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

ML Process - Model evaluation

Performance measures: **regression tasks**

Usually distance between $h_\theta(x^{(i)})$ and $y^{(i)}$

- R^2: Coefficient of determination (a.k.a. "R squared")

$$\bar{y} = \frac{1}{m}\sum_{i=1}^{m} y^{(i)}$$

$$R^2(X,h_\theta) = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum_{i=1}^{m} (\, y^{(i)} - h_\theta(x^{(i)})\,)^2}{\sum_{i=1}^{m} (\, y^{(i)} - \bar{y}\,)^2}$$

ML Process - Model evaluation

Ranges from −∞ to 1 and indicates how well the model fits data.

egression tasks

Us distance between $h_\theta(x^{(i)})$ and $y^{(i)}$

- R^2: Coefficient of determination (a.k.a. "R squared")

$$\bar{y} = \frac{1}{m} \sum_{i=1}^{m} y^{(i)}$$

$$R^2(X, h_\theta) = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum_{i=1}^{m} (\, y^{(i)} - h_\theta(x^{(i)}) \,)^2}{\sum_{i=1}^{m} (\, y^{(i)} - \bar{y} \,)^2}$$

Performance measures: **classification tasks**

- $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{1, 2, \ldots, k\}$

Map features to classes

- Distance measures (*e.g.*, $h_\theta(x^{(i)}) - y^{(i)}$) do not reflect well the performance of the classifier

ML Process - Model evaluation

Performance measures: **classification tasks**

- $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{1, 2, \ldots, k\}$

Map features to classes

- Distance measures (*e.g.*, $h_\theta(x^{(i)}) - y^{(i)}$) do not reflect well the performance of the classifier

- Ex: for a given sample $(x^{(i)}, y^{(i)})$, assume:
  - $y^{(i)} = 2$
  - $h_{\theta'}(x^{(i)}) = 3 \quad \Rightarrow \quad h_{\theta'}(x^{(i)}) - y^{(i)} = 3-2 = 1$
  - $h_{\theta''}(x^{(i)}) = 7 \quad \Rightarrow \quad h_{\theta''}(x^{(i)}) - y^{(i)} = 7-2 = 5$

ML Process - Model evaluation

Performance measures: **classification tasks**

- $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{1, 2, \ldots, k\}$

Map features to classes

- Distance measures (*e.g.*, $h_\theta(x^{(i)}) - y^{(i)}$) do not reflect well the performance of the classifier

- Ex: for a given sample $(x^{(i)}, y^{(i)})$, assume:
  - $y^{(i)} = 2$
  - $h_{\theta'}(x^{(i)}) = 3 \quad \Rightarrow \quad h_{\theta'}(x^{(i)}) - y^{(i)} = 3\text{-}2 = 1$
  - $h_{\theta''}(x^{(i)}) = 7 \quad \Rightarrow \quad h_{\theta''}(x^{(i)}) - y^{(i)} = 7\text{-}2 = 5$

Is $h_{\theta'}(x^{(i)})$ better than $h_{\theta''}(x^{(i)})$?

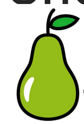
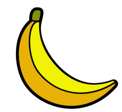
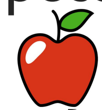
ML Process - Model evaluation

Performance measures: **classification tasks**

- Confusion matrix
- Accuracy / Error rate
- Precision
- Recall
- F-score

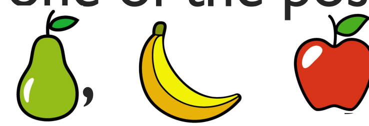
ML Process - Model evaluation

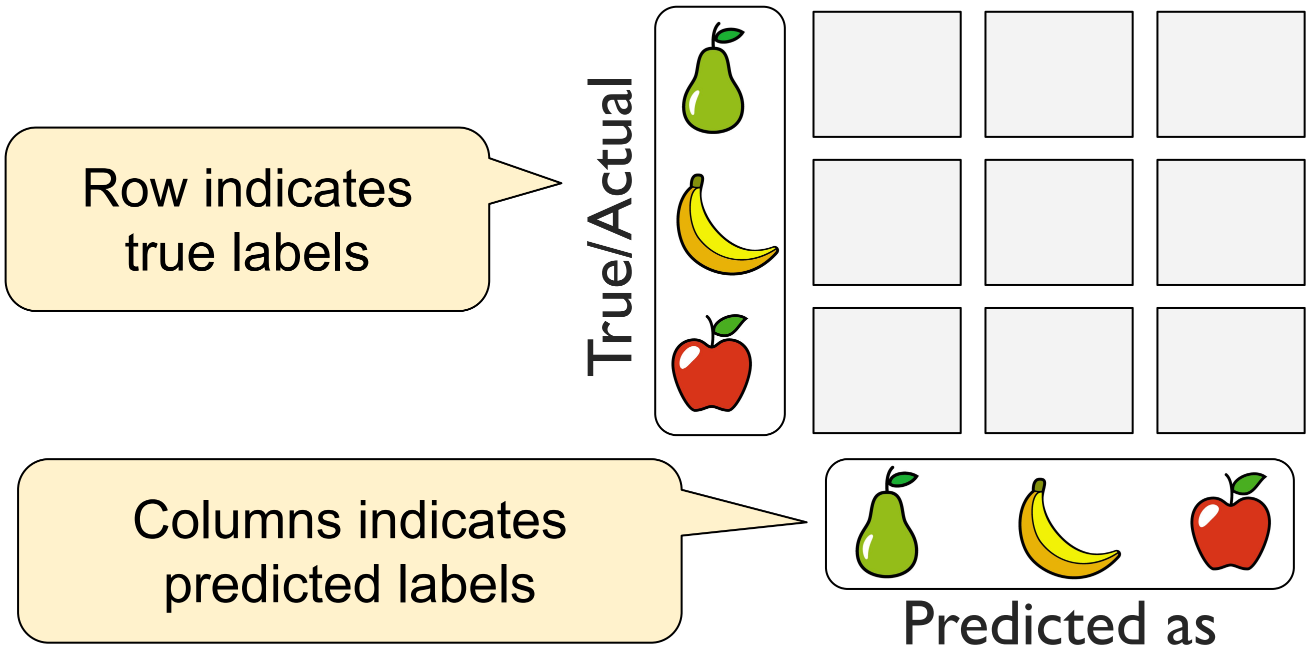
Performance measures: **classification tasks**

- Confusion matrix: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ ,  

ML Process - Model evaluation

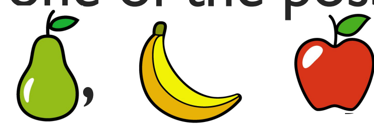
Performance measures: **classification tasks**

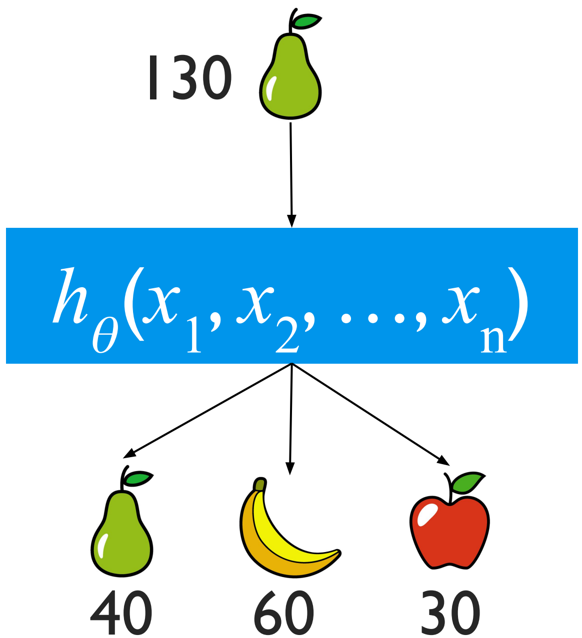
- <u>Confusion matrix</u>: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

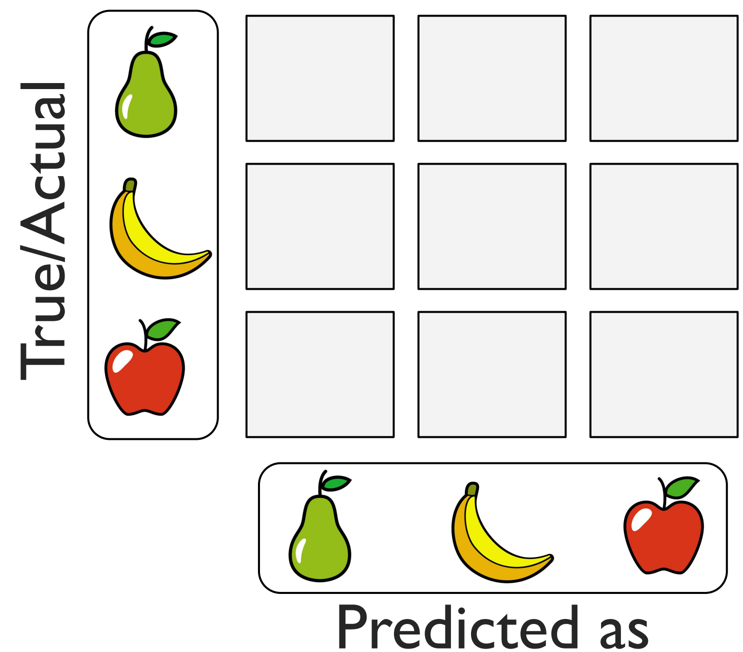


ML Process - Model evaluation

Performance measures: **classification tasks**

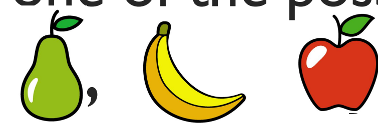
- Confusion matrix: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

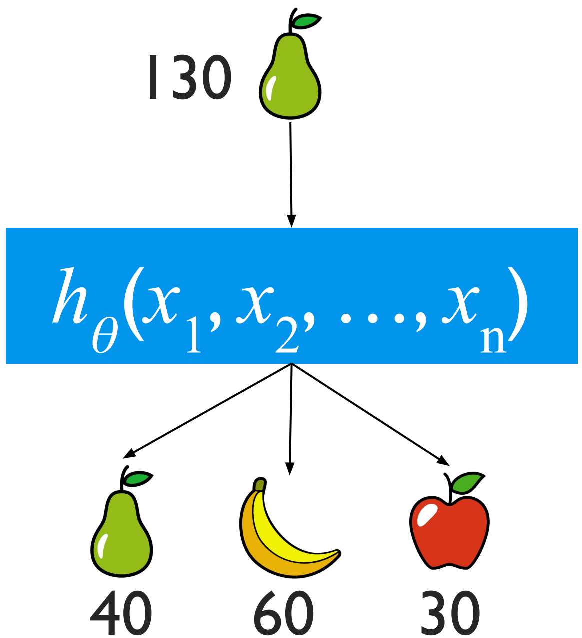


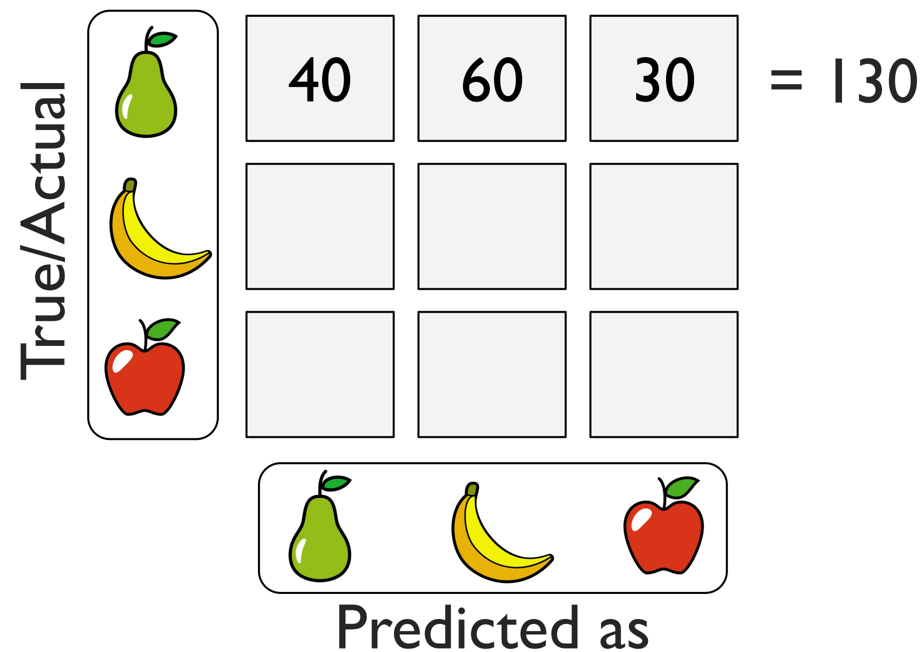


ML Process - Model evaluation

Performance measures: **classification tasks**

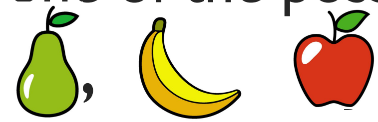
- Confusion matrix: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

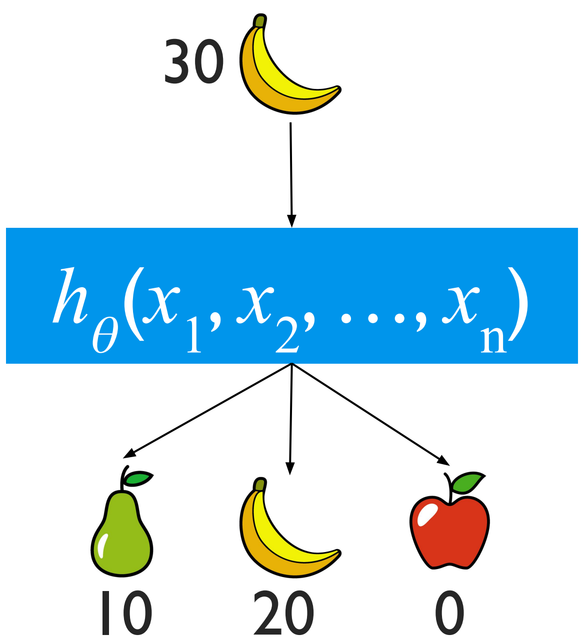


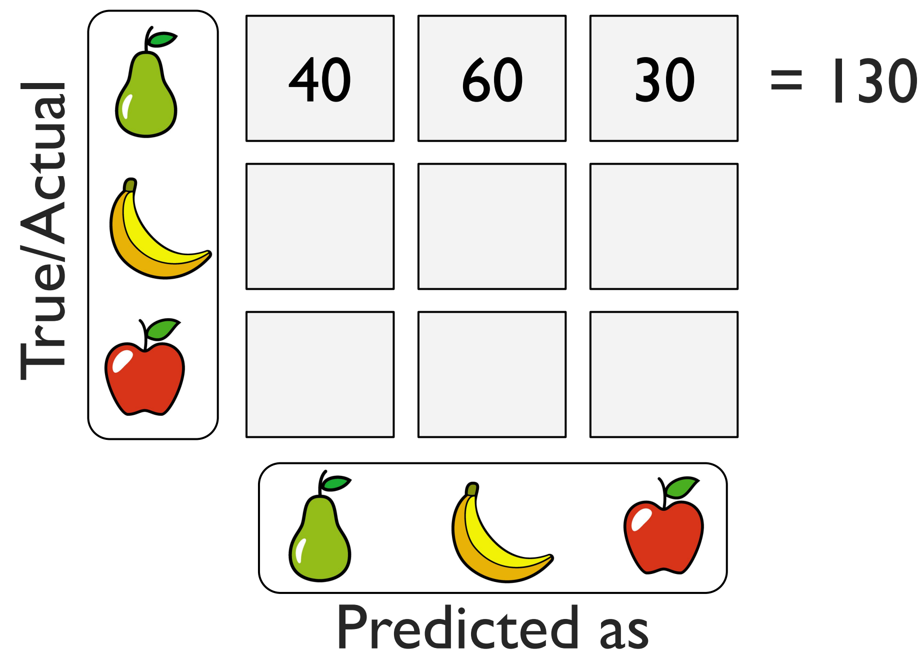


ML Process - Model evaluation

Performance measures: **classification tasks**

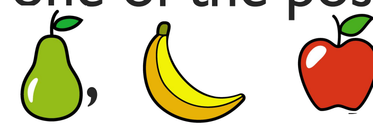
- Confusion matrix: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

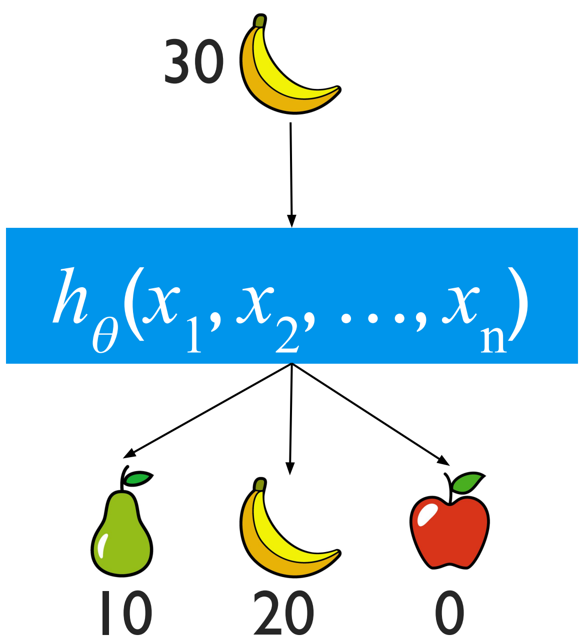


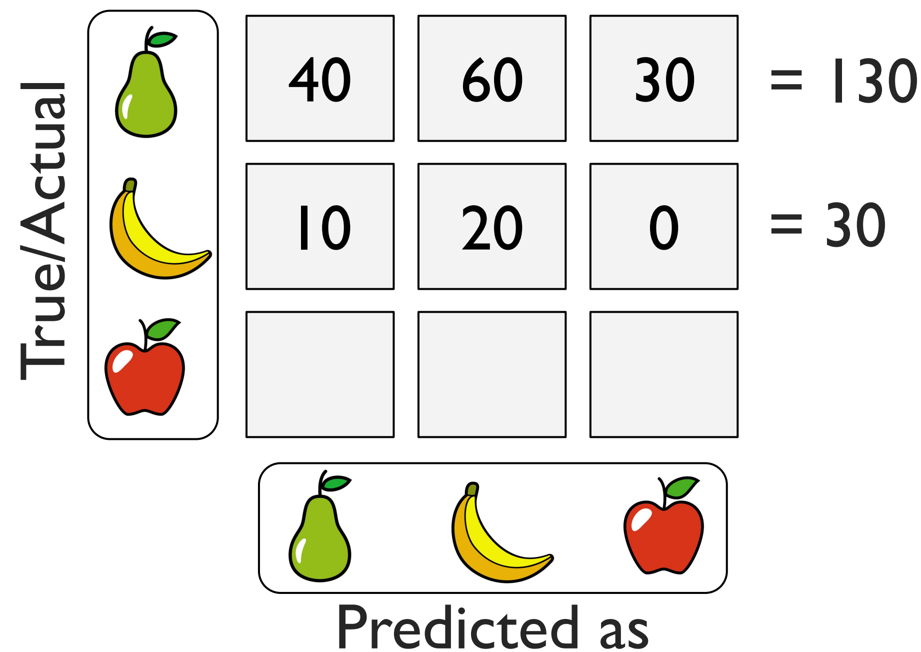


ML Process - Model evaluation

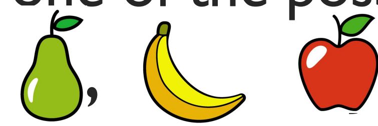
Performance measures: **classification tasks**

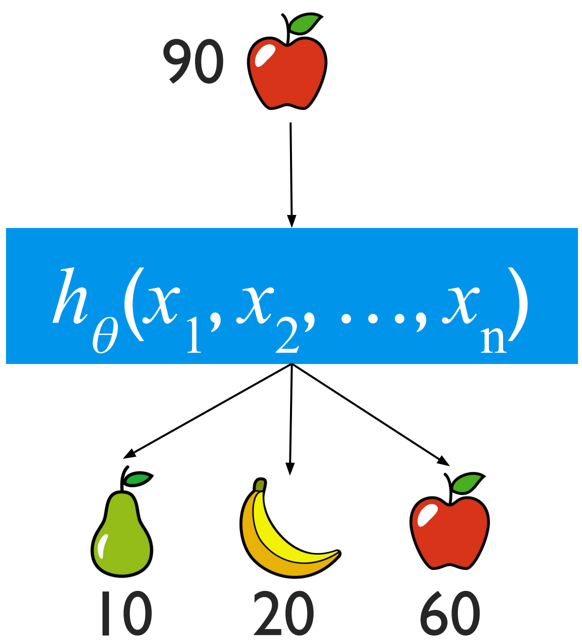
- <u>Confusion matrix</u>: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

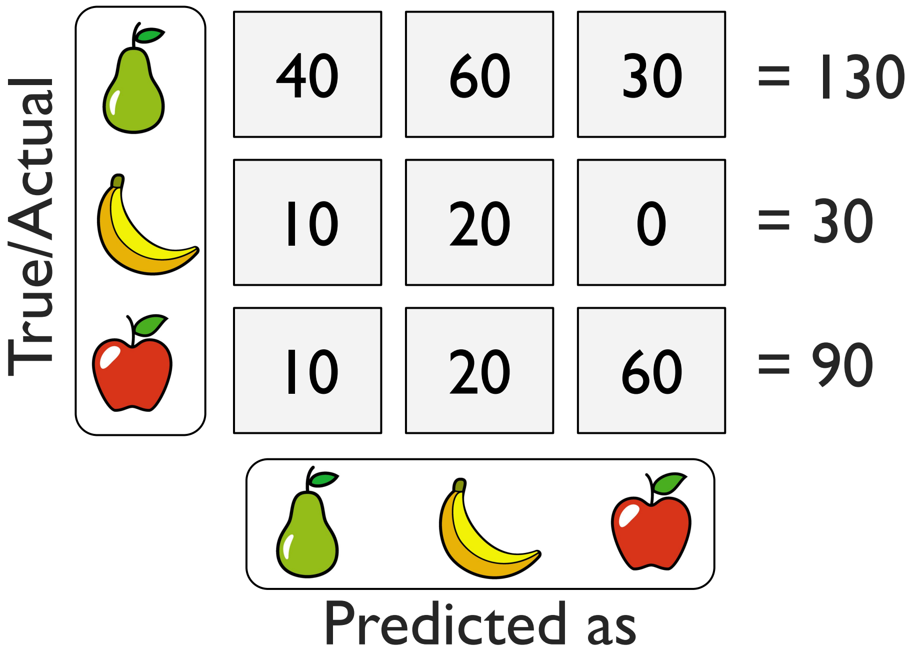




Performance measures: **classification tasks**

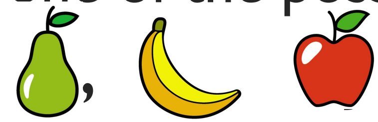
- <u>Confusion matrix</u>: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

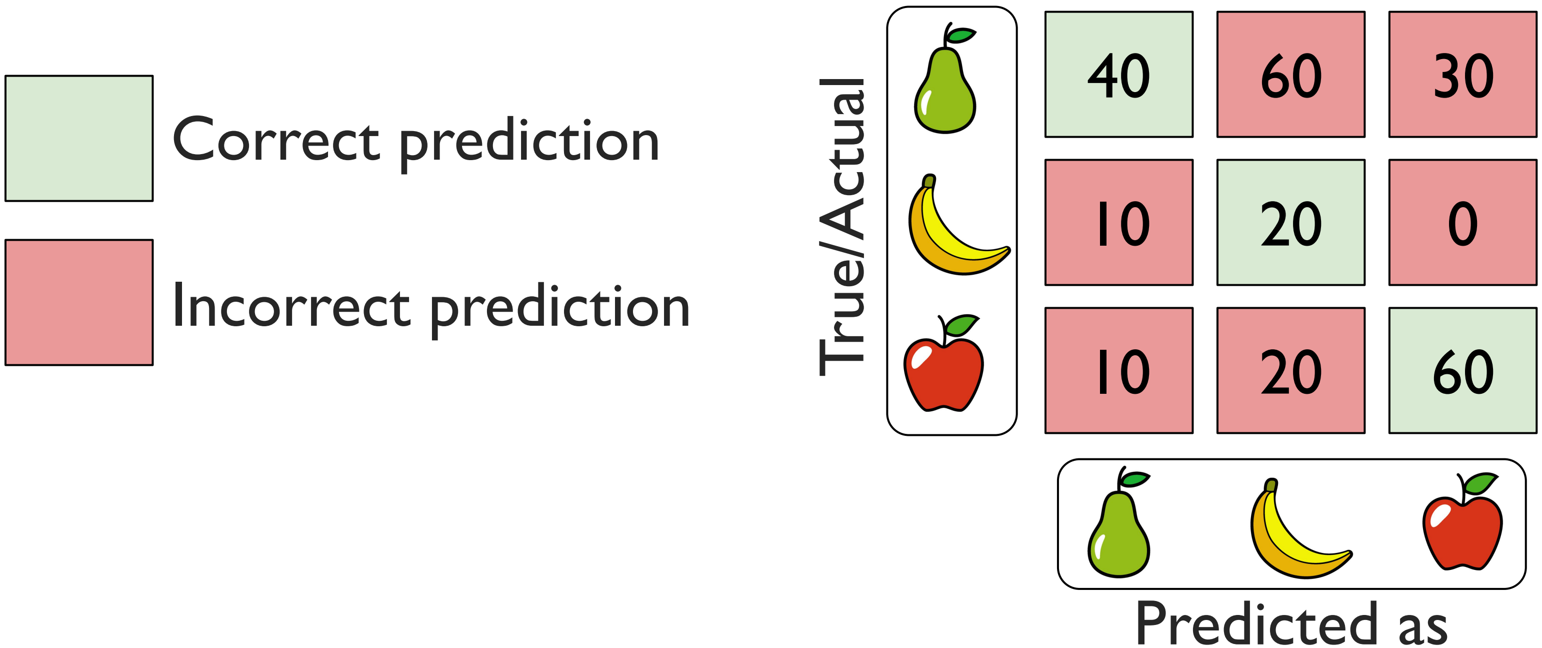




ML Process - Model evaluation

Performance measures: **classification tasks**

- <u>Confusion matrix</u>: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

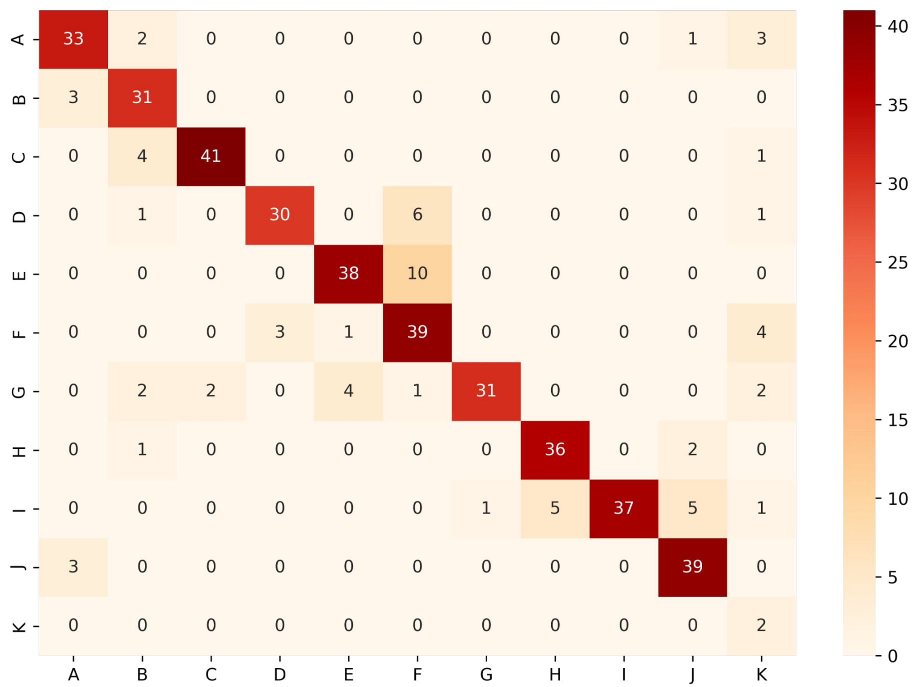


ML Process - Model evaluation

Performance measures: **classification tasks**

- <u>Confusion matrix</u>: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow$ {A,B,...,K}

Confusion matrix colored with **Heatmap**



ML Process - Model evaluation

Performance measures: **classification tasks**

- <u>Confusion matrix</u>: table that shows, for each class, how many of its samples where predicted as each one of the possible classes
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow$ {A,B,...,K}

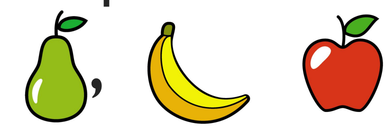
Confusion matrix colored with Heatmap

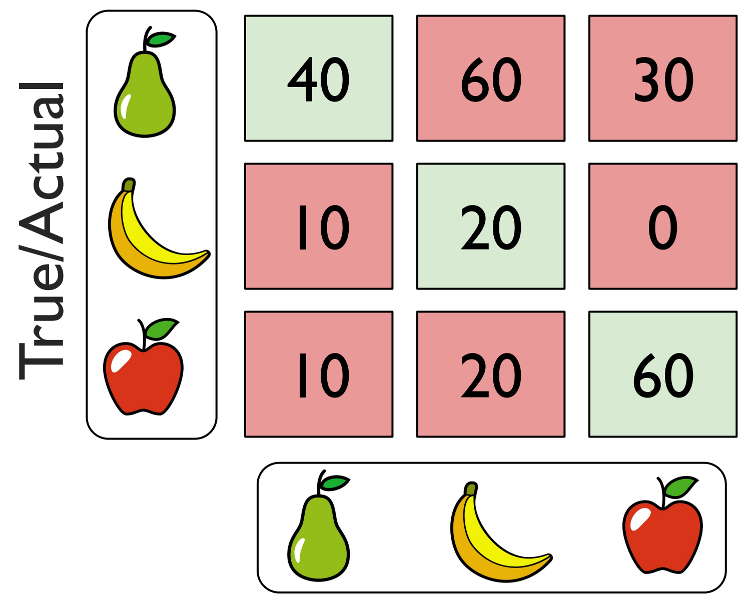
	A	B	C	D	E	F	G	H	I	J	K
A	33	2	0	0	0	0	0	0	0	1	3
B	3	31	0	0	0	0	0	0	0	0	0
C	0	4	41	0	0	0	0	0	0	0	1
D	0	1	0	30	0	6	0	0	0	0	1
E	0	0	0	0	38	10	0	0	0	0	0
F	0	0	0	3	1	39	0	0	0	0	4
G	0	2	2	0	4	1	31	0	0	0	2
H	0	1	0	0	0	0	0	36	0	2	0
I	0	0	0	0	0	0	1	5	37	5	1
J	3	0	0	0	0	0	0	0	0	39	0
K	0	0	0	0	0	0	0	0	0	0	2

Sometimes is useful to summarize the whole result as a single number (*e.g.*, accuracy)

ML Process - Model evaluation

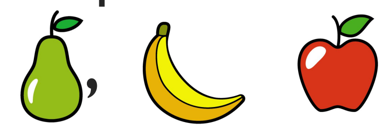
Performance measures: **classification tasks**

- Accuracy = all correct predictions / all predictions.
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 



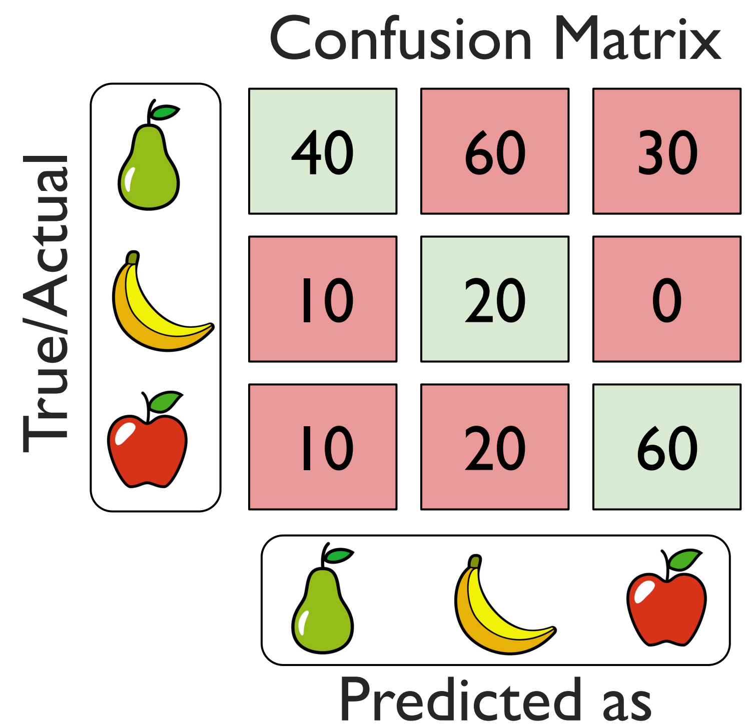
ML Process - Model evaluation

Performance measures: **classification tasks**

- Accuracy = all correct predictions / all predictions.
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

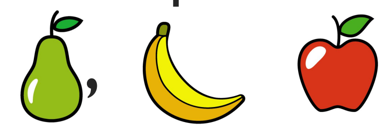
Accuracy = 120/250 = **48%**

Summarizes all results



ML Process - Model evaluation

Performance measures: **classification tasks**

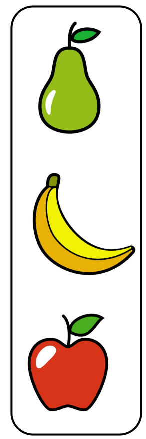
- Error rate = all incorrect predictions / all predictions.
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

Error rate = 130/250 = **52%**

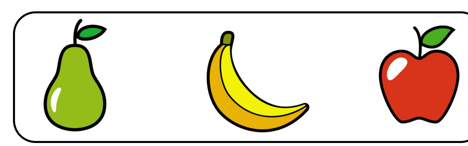
error rate = 1 - accuracy

Confusion Matrix

True/Actual



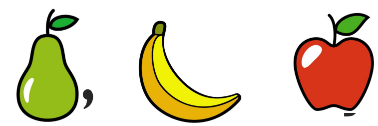
40	60	30
10	20	0
10	20	60



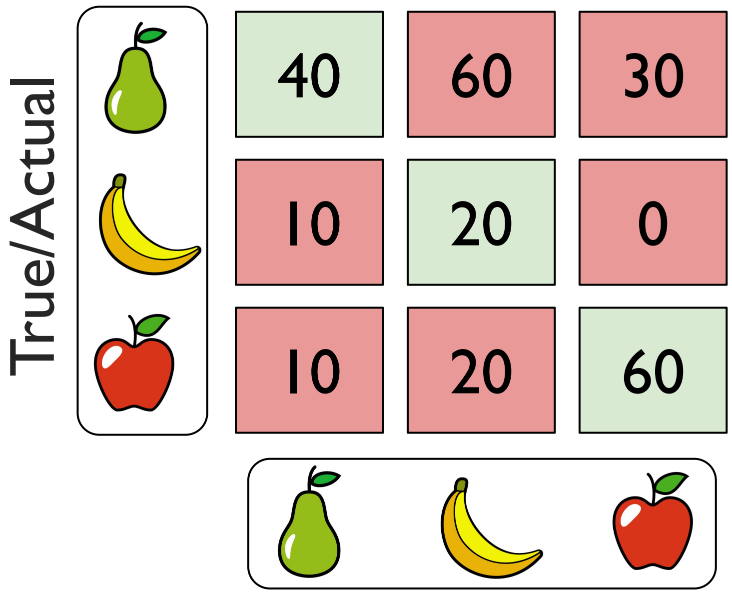
Predicted as

ML Process - Model evaluation

Performance measures: **classification tasks**

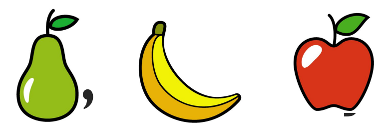
- Precision = proportion of samples predicted as class X that really belong to class X.
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

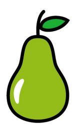
Confusion Matrix

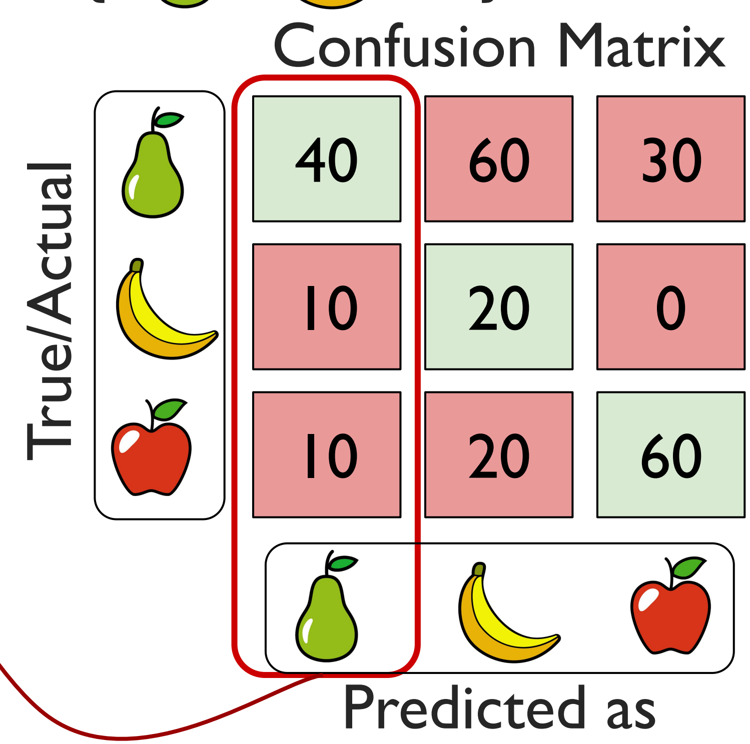


ML Process - Model evaluation

Performance measures: **classification tasks**

- Precision = proportion of samples predicted as class X that really belong to class X.
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

 precision = 40/60 = **66.7%**



ML Process - Model evaluation

Performance measures: **classification tasks**

- Precision = proportion of samples predicted as class X that really belong to class X.
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$,

Confusion Matrix

True/Actual

	40	60	30
	10	20	0
	10	20	60

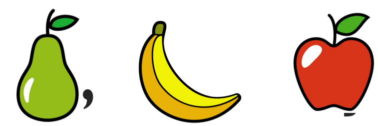
Predicted as

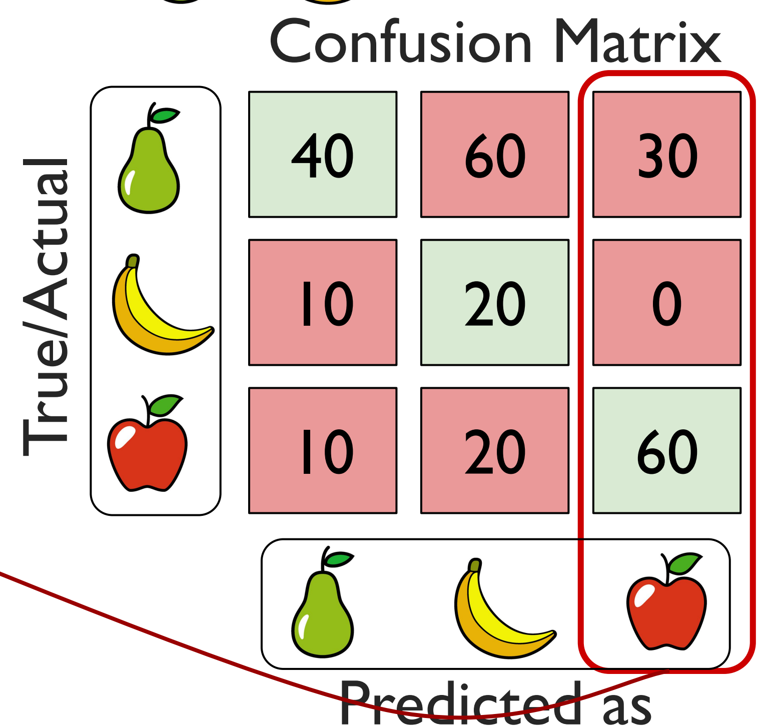
precision = 40/60 = **66.7%**

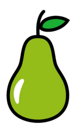
precision = 20/100 = **20%**

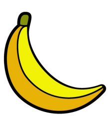
ML Process - Model evaluation

Performance measures: **classification tasks**

- Precision = proportion of samples predicted as class X that really belong to class X.
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

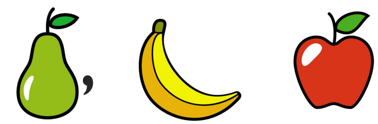


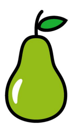
 precision = 40/60 = **66.7%**

 precision = 20/100 = **20%**

 precision = 60/90 = **66.7%**

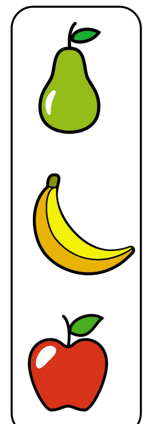
Performance measures: **classification tasks**

- Recall = proportion of class X samples classified correctly
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

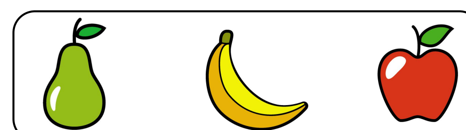
 recall = 40/130 = **30.8%**

Confusion Matrix

True/Actual



40	60	30
10	20	0
10	20	60

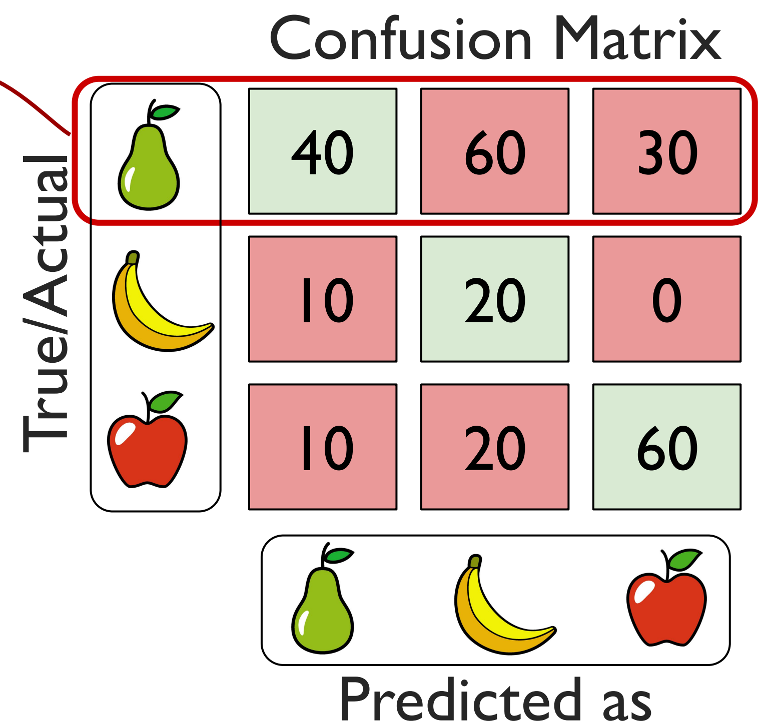


Predicted as

Performance measures: **classification tasks**

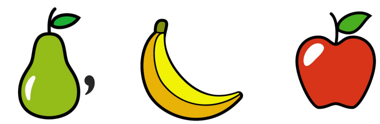
- <u>Recall</u> = proportion of class X samples classified correctly
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$, $\}$

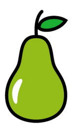
recall = 40/130 = **30.8%**

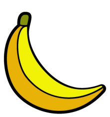


ML Process - Model evaluation

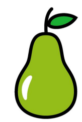
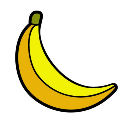
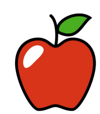
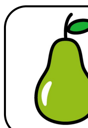
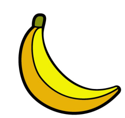
Performance measures: **classification tasks**

- Recall = proportion of class X samples classified correctly
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 

 recall = 40/130 = **30.8%**

 recall = 20/30 = **66.7%**

Confusion Matrix

True/Actual			
	40	60	30
	10	20	0
	10	20	60
			

Predicted as

ML Process - Model evaluation

Performance measures: **classification tasks**

- <u>Recall</u> = proportion of class X samples classified correctly
- Example: $h_\theta(x_1, x_2, \ldots, x_n) \rightarrow \{$ 🍐, 🍌 🍎

🍐 recall = 40/130 = **30.8%**

🍌 recall = 20/30 = **66.7%**

🍎 recall = 60/90 = **66.7%**

Confusion Matrix

True/Actual \ Predicted as	🍐	🍌	🍎
🍐	40	60	30
🍌	10	20	0
🍎	10	20	60

ML Process - Model evaluation

Performance measures: **classification tasks**

- <u>Summarizing Recall and Precision</u>
  - F_1-score = (2 × precision × recall) / (precision + recall)
  - macro vs weighted average

	Precision	Recall
Pear	66.7%	30.8%
Banana	20%	66.7%
Apple	66.7%	66.7%

ML Process - Model evaluation

Precision and recall
Harmonic mean

Performance measures: **classification**

- <u>Summarizing Recall and Precision</u>
  - **F_1-score = (2 × precision × recall) / (precision + recall)**
  - macro vs weighted average

	Precision	Recall	F_1-score
Pear	66.7%	30.8%	42.1%
Banana	20%	66.7%	30.8%
Apple	66.7%	66.7%	66.7%

Performance measures: **classification tasks**

- Summarizing Recall and Precision
  - F_1-score = (2 × precision × recall) / (precision + recall)
  - macro vs weighted average

		Precision	Recall	F_1-score
130	Pear	66.7%	30.8%	42.1%
30	Banana	20%	66.7%	30.8%
90	Apple	66.7%	66.7%	66.7%
	Macro avg	51.1%	54.73	46.5%
	Weighted avg	61.1%	48.0%	49.6%

Weighted by the number of samples

Accuracy = 120/250 = **48%**

ML Process - Model evaluation

Per

•

```
from sklearn import metrics
print(metrics.classification_report(y_true, y_pred, digits=3))

              precision    recall  f1-score   support

       Apple      0.667     0.667     0.667        90
      Banana      0.200     0.667     0.308        30
        Pear      0.667     0.308     0.421       130

    accuracy                          0.480       250
   macro avg      0.511     0.547     0.465       250
weighted avg      0.611     0.480     0.496       250
```

		Precision	Recall	F_1-score
130	Pear	66.7%	30.8%	42.1%
30	Banana	20%	66.7%	30.8%
90	Apple	66.7%	66.7%	66.7%
	Macro avg	51.1%	54.73	46.5%
	Weighted avg	61.1%	48.0%	49.6%

scikit learn classification report

Accuracy = 120/250 = **48%**

ML Process - Model evaluation

Performance measures: **classification tasks**

$$F_\beta\text{-score} = \frac{(1+\beta^2) \times \text{precision} \times \text{recall}}{(\beta^2 \times \text{precision}) + \text{recall}}$$

$$F_1\text{-score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

F_1-score is a special case of F_β-score ($\beta=1$)

Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- On binary classifiers, precision and recall are usually expressed as a function of Positives, Negatives, True Positive (TP), False Positives (FP), True Negative (TN), and False Negatives (FN)

- For a given class X
  - Positives: samples predicted as class X
    - TP (True Positives) samples correctly predicted as X
    - FP (False Positives) samples incorrectly predicted as X
  - Negatives: samples predicted as non-X (other classes)
    - TN (True Negative) samples correctly predicted as non-X
    - FN (False Negative) samples incorrectly predicted as non-X

ML Process - Model evaluation

Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction

Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction

Confusion Matrix

True/Actual		
	467	7
	6456	277879

Predicted as

Positives	6923
TP	
FP	
Negatives	
TN	
FN	

Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

ML Process - Model evaluation

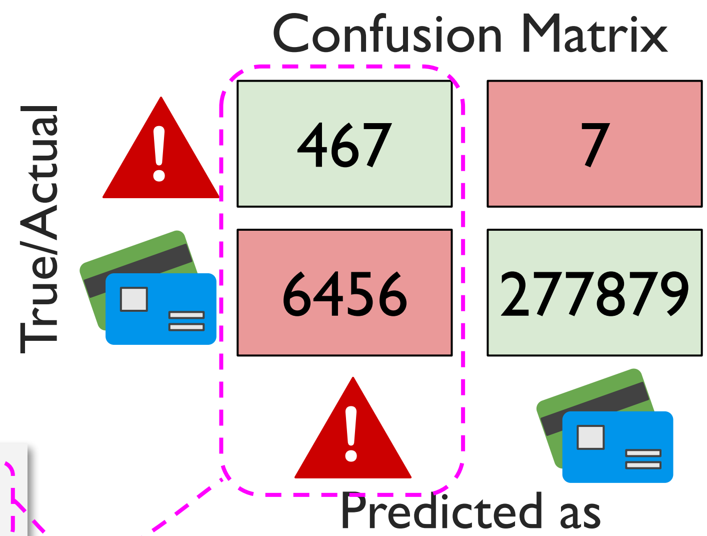
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction



Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

Positives	6923
TP	
FP	
Negatives	
TN	
FN	

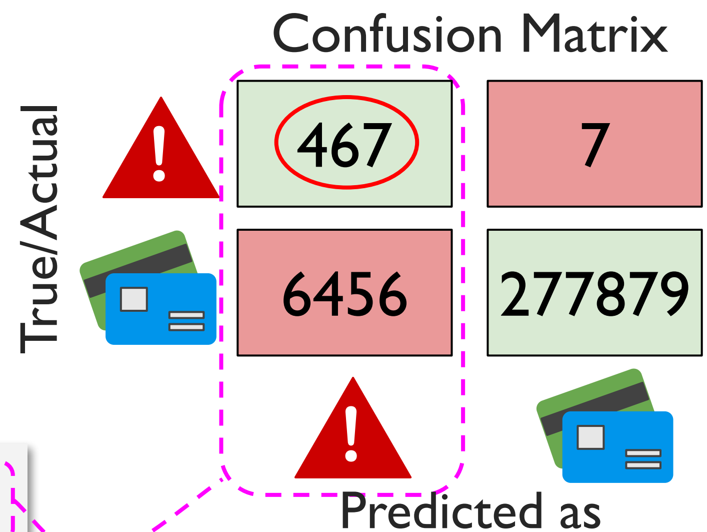
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction



Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

Positives	6923
TP	467
FP	
Negatives	
TN	
FN	

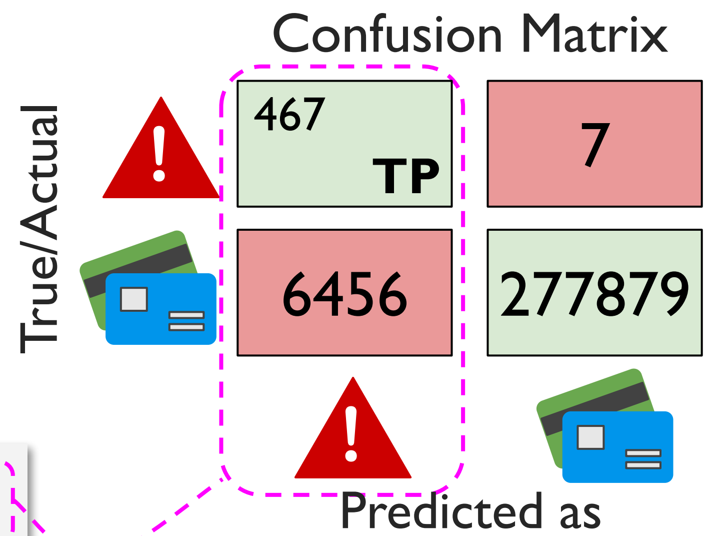
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction



Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

Positives	6923
TP	467
FP	
Negatives	
TN	
FN	

ML Process - Model evaluation

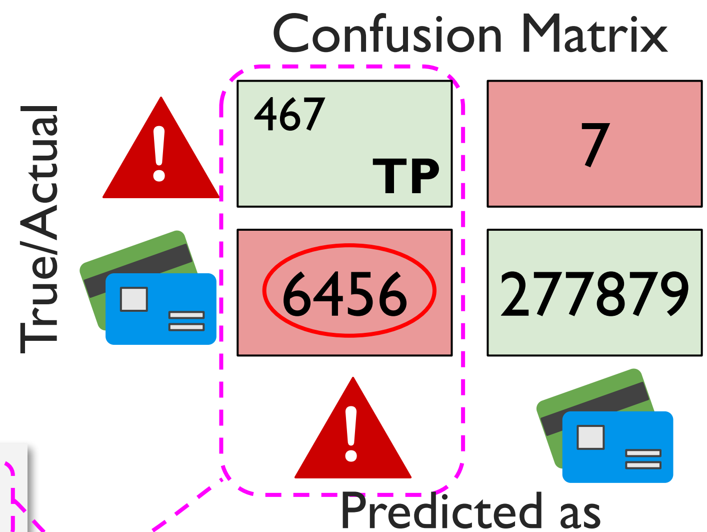
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

  Fraud

  Legitimate transaction



Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

Positives	6923
TP	467
FP	6456
Negatives	
TN	
FN	

ML Process - Model evaluation

Performance measures: **classification tasks**

<u>P, N, TP</u>, <u>FP</u>, <u>TN</u>, <u>FN</u>

- **Ex: Credit card fraud**

Fraud

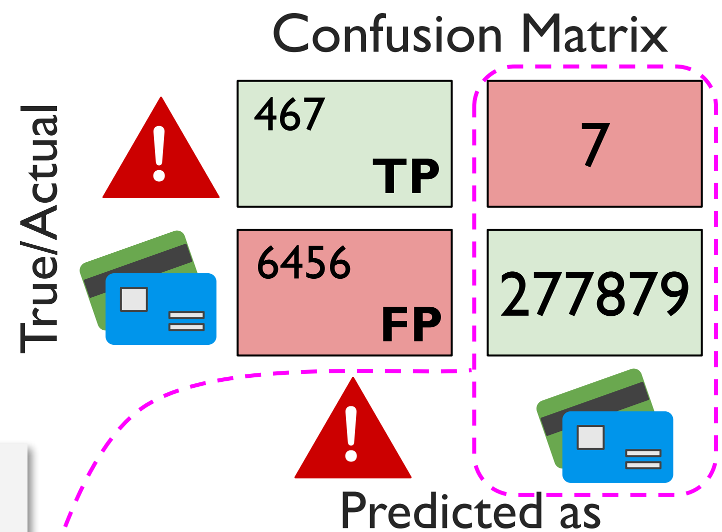
Legitimate transaction

<u>Positives</u>: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

<u>Negatives</u>: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)



Positives	6923
TP	467
FP	6456
Negatives	277886
TN	
FN	

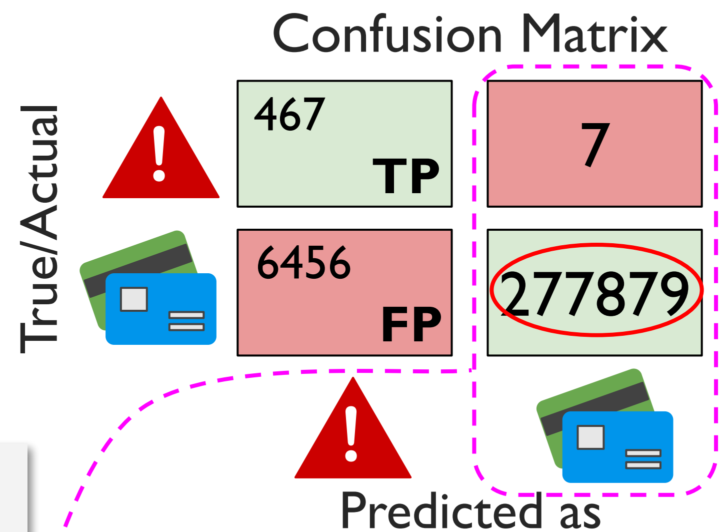
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

  Fraud

  Legitimate transaction



Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

Positives	6923
TP	467
FP	6456
Negatives	277886
TN	277879
FN	

ML Process - Model evaluation

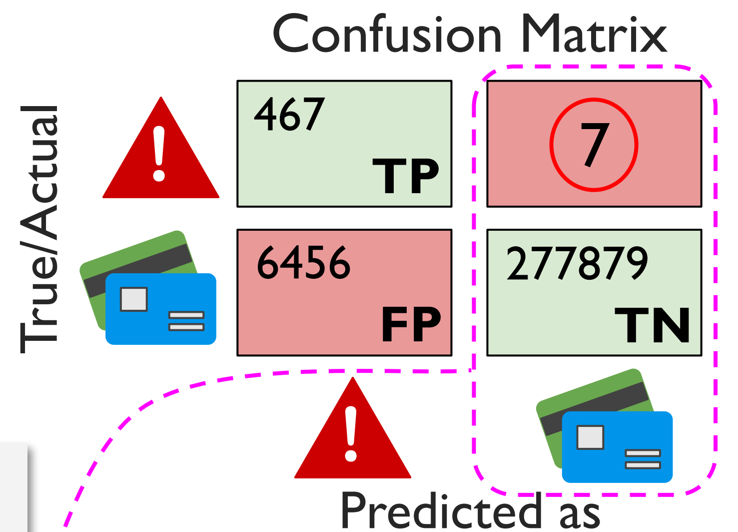
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction



Positives: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

Negatives: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

Positives	6923
TP	467
FP	6456
Negatives	277886
TN	277879
FN	7

ML Process - Model evaluation

Performance measures: **classification tasks**

<u>P, N, TP, FP, TN, FN</u>

- **Ex: Credit card fraud**

Fraud

Legitimate transaction

Confusion Matrix

True/Actual

467 **TP**	7 **FN**
6456 **FP**	277879 **TN**

Predicted as

Positives	6923
TP	467
FP	6456
Negatives	277886
TN	277879
FN	7

<u>Positives</u>: samples predicted as fraud

- TP: correctly predicted as fraud
- FP: incorrectly predicted as fraud

<u>Negatives</u>: samples predicted as non-fraud

- TN: correctly predicted as non-fraud (legit)
- FN: incorrectly predicted as non-fraud (legit)

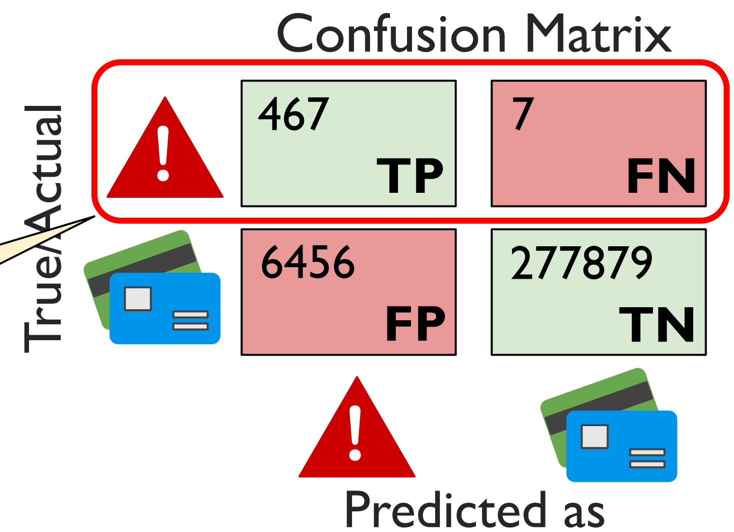
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction



$$\text{recall} = \frac{TP}{TP+FN} = 98.5\%$$

Positives	6923
	467
	6456
	277879
TN	277879
FN	7

Good: Most frauds are detected!

ML Process - Model evaluation

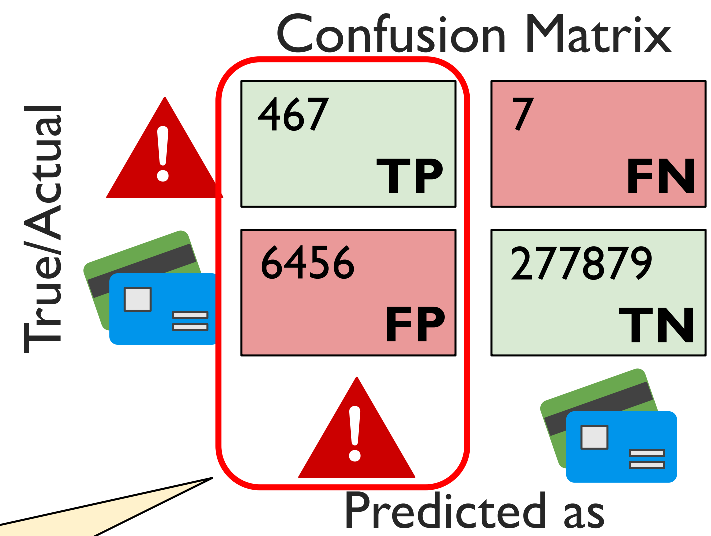
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction



$$\text{recall} = \frac{TP}{TP+FN} = 98.5\%$$

$$\text{precision} = \frac{TP}{TP+FP} = \frac{TP}{P} = 6.7\%$$

Positives	6923
TP	467

Problem: Many legit transactions flagged as fraud!

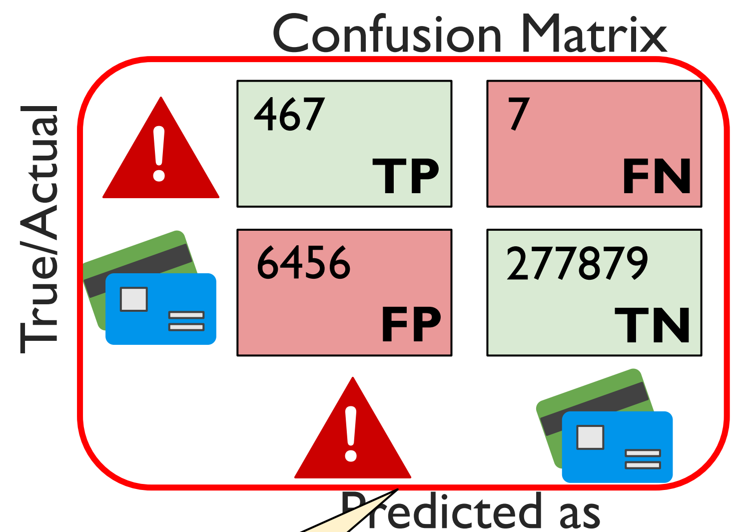
Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction



Positives	6923
TP	467
FP	6456
Negatives	277879
TN	277879
FN	7

$$\text{recall} = \frac{TP}{TP+FN} = 98.5\%$$

$$\text{precision} = \frac{TP}{TP+FP} = 6.7\%$$

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN} = 97.7\%$$

ML Process - Model evaluation

Performance measures: **classification tasks**

P, N, TP, FP, TN, FN

- **Ex: Credit card fraud**

Fraud

Legitimate transaction

$$\text{recall} = \frac{TP}{TP+FN} = 98.5\%$$

$$\text{precision} = \frac{TP}{TP+FP} = 6.7\%$$

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN} = 97.7\%$$

Confusion Matrix

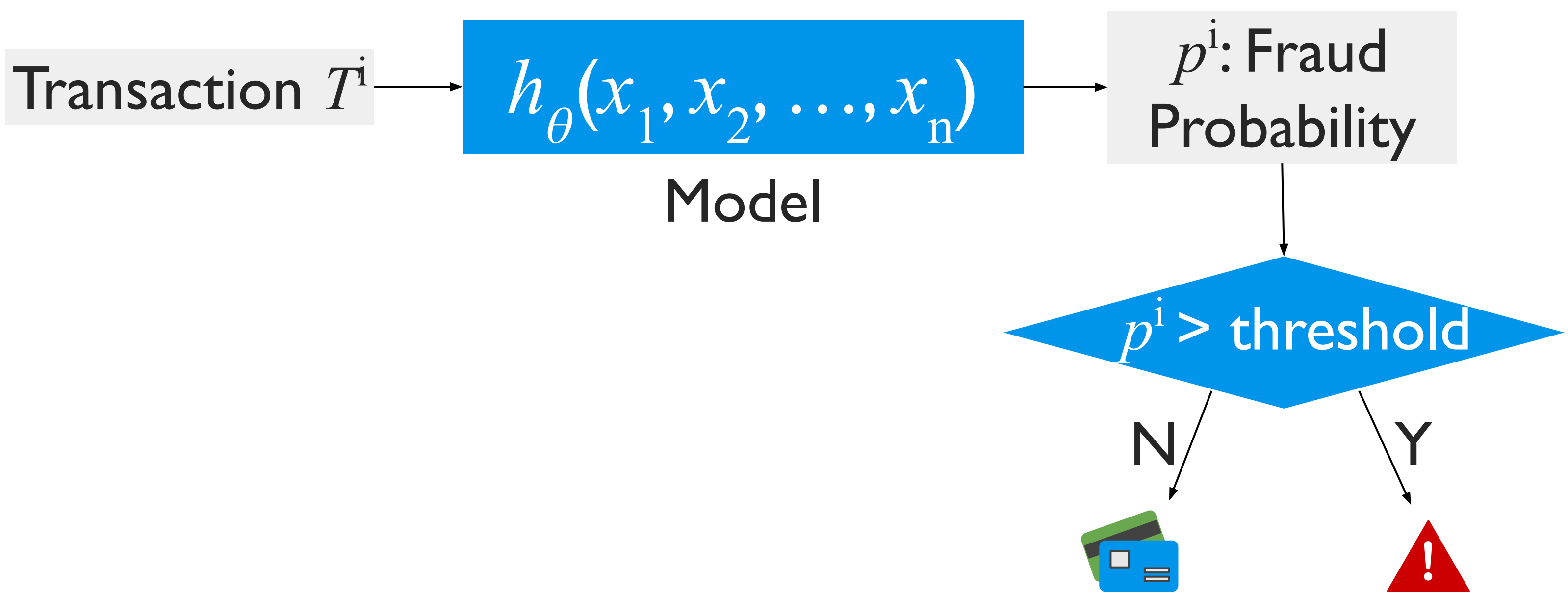
True/Actual \ Predicted as	Fraud	Legitimate
Fraud	467 **TP**	7 **FN**
Legitimate	6456 **FP**	277879 **TN**

Positives	6923
TP	467
FP	6456
Negatives	277879
TN	277879
FN	7

ML Process - Model evaluation

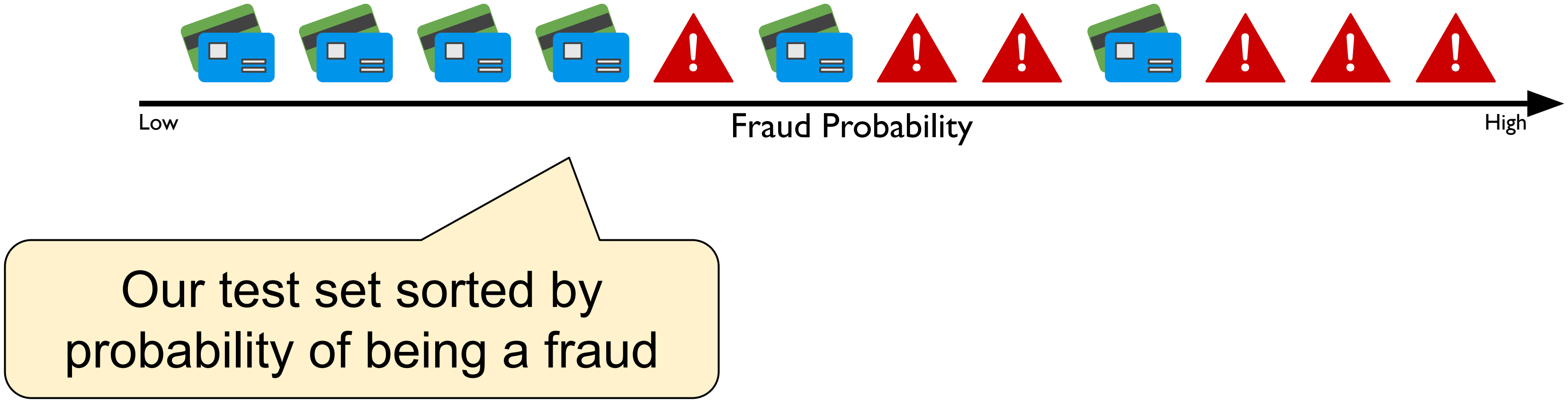
Performance measures: **classification tasks**

- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



Performance measures: **classification tasks**

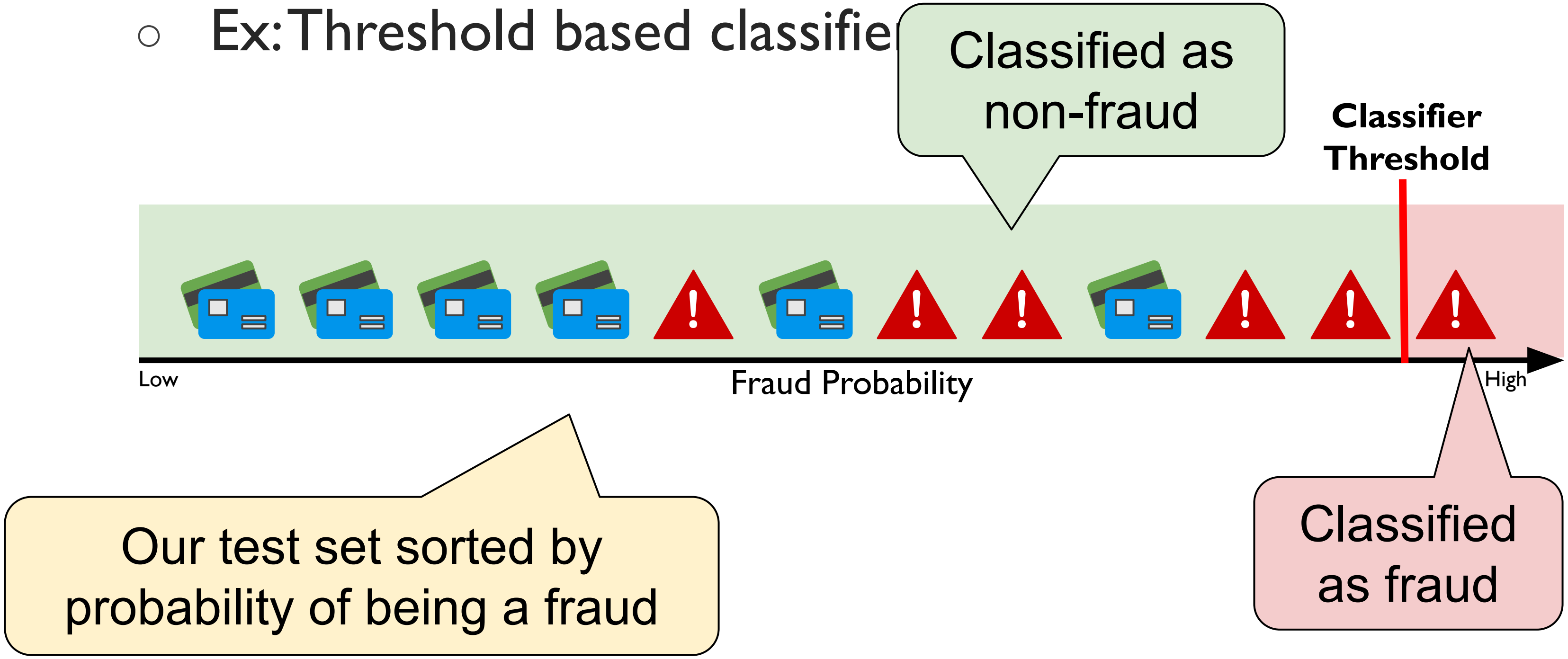
- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



ML Process - Model evaluation

Performance measures: **classification tasks**

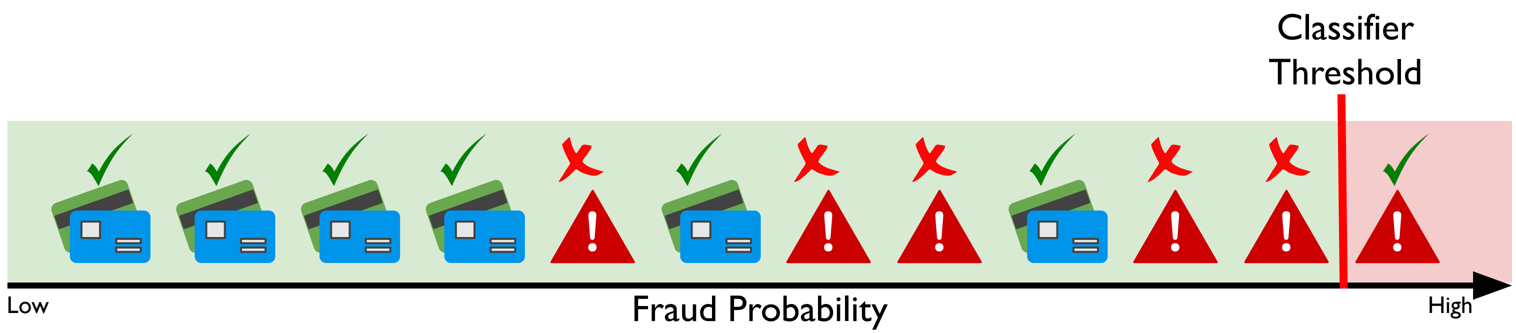
- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



ML Process - Model evaluation

Performance measures: **classification tasks**

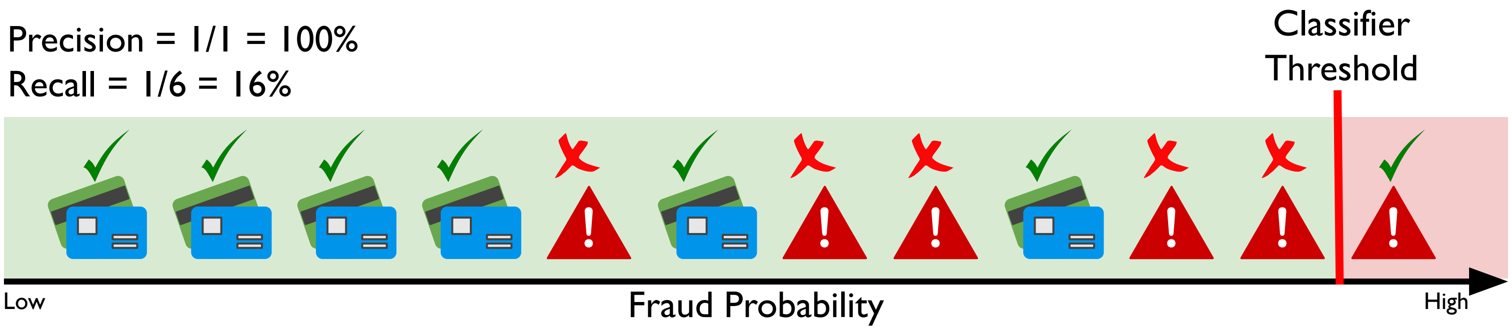
- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



ML Process - Model evaluation

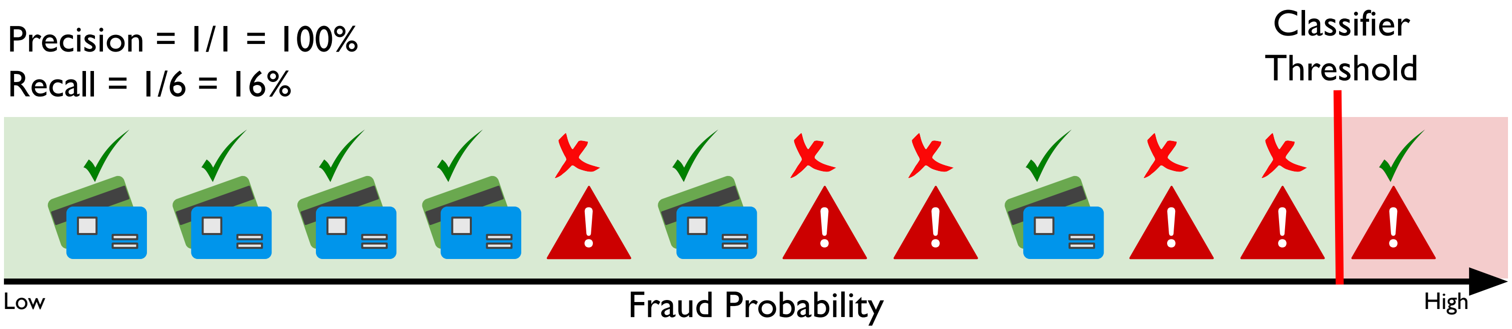
Performance measures: **classification tasks**

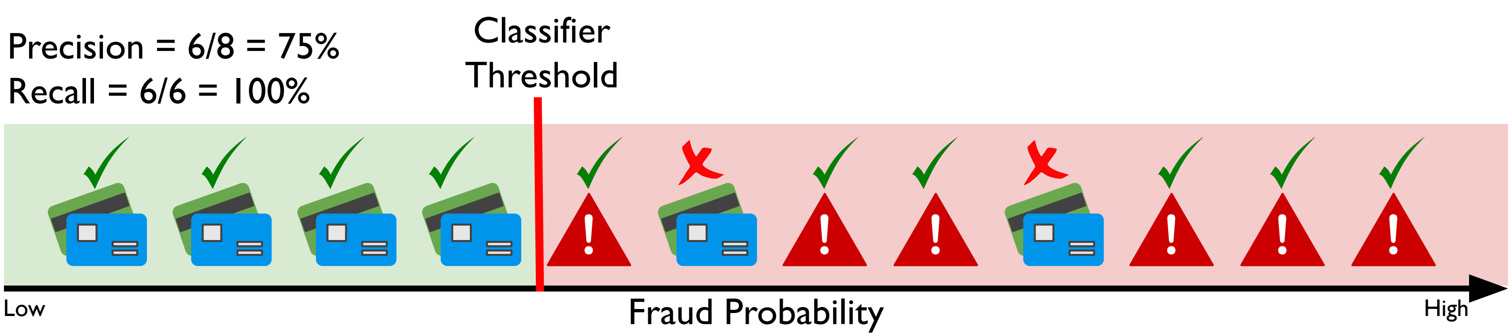
- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



Performance measures: **classification tasks**

- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier

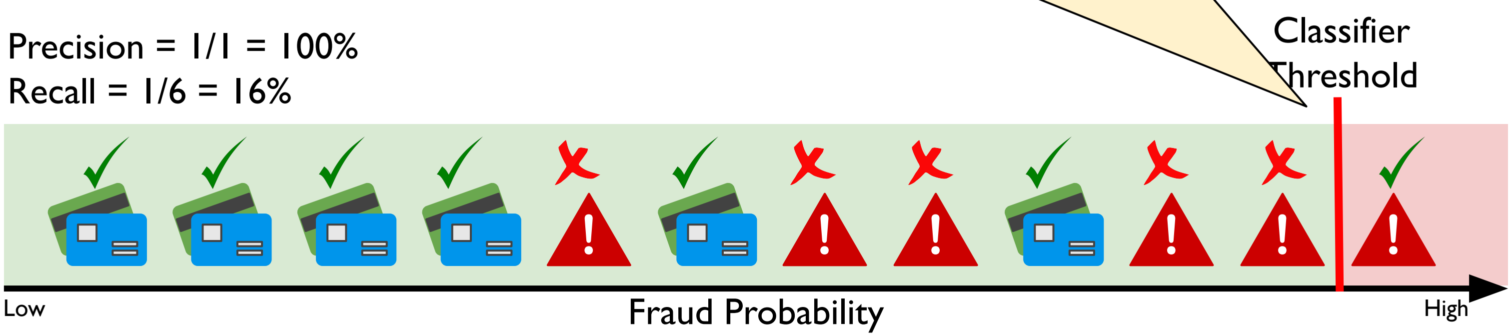


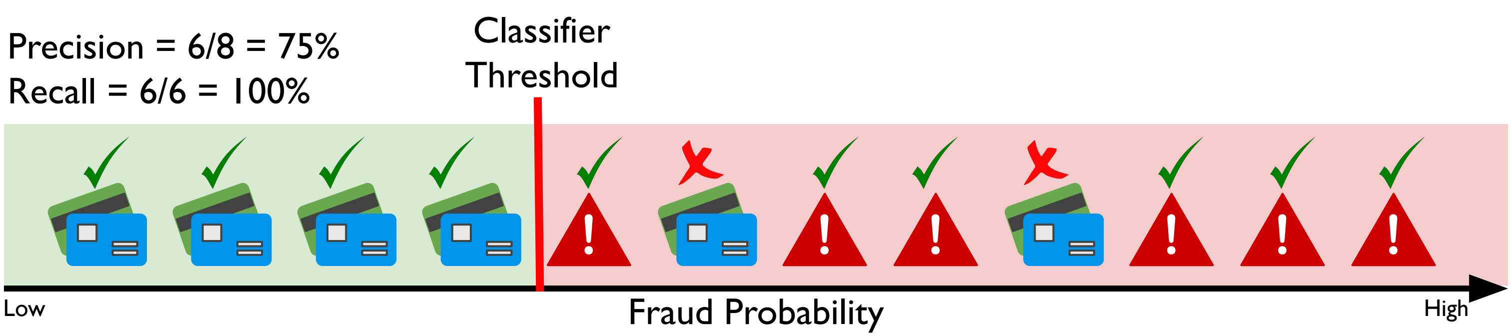


Performance measures: **classification tasks**

- Precision/R
  - Many class ... n or recall
  - Ex: Thresho

Increasing the threshold improves precision but decreases recall

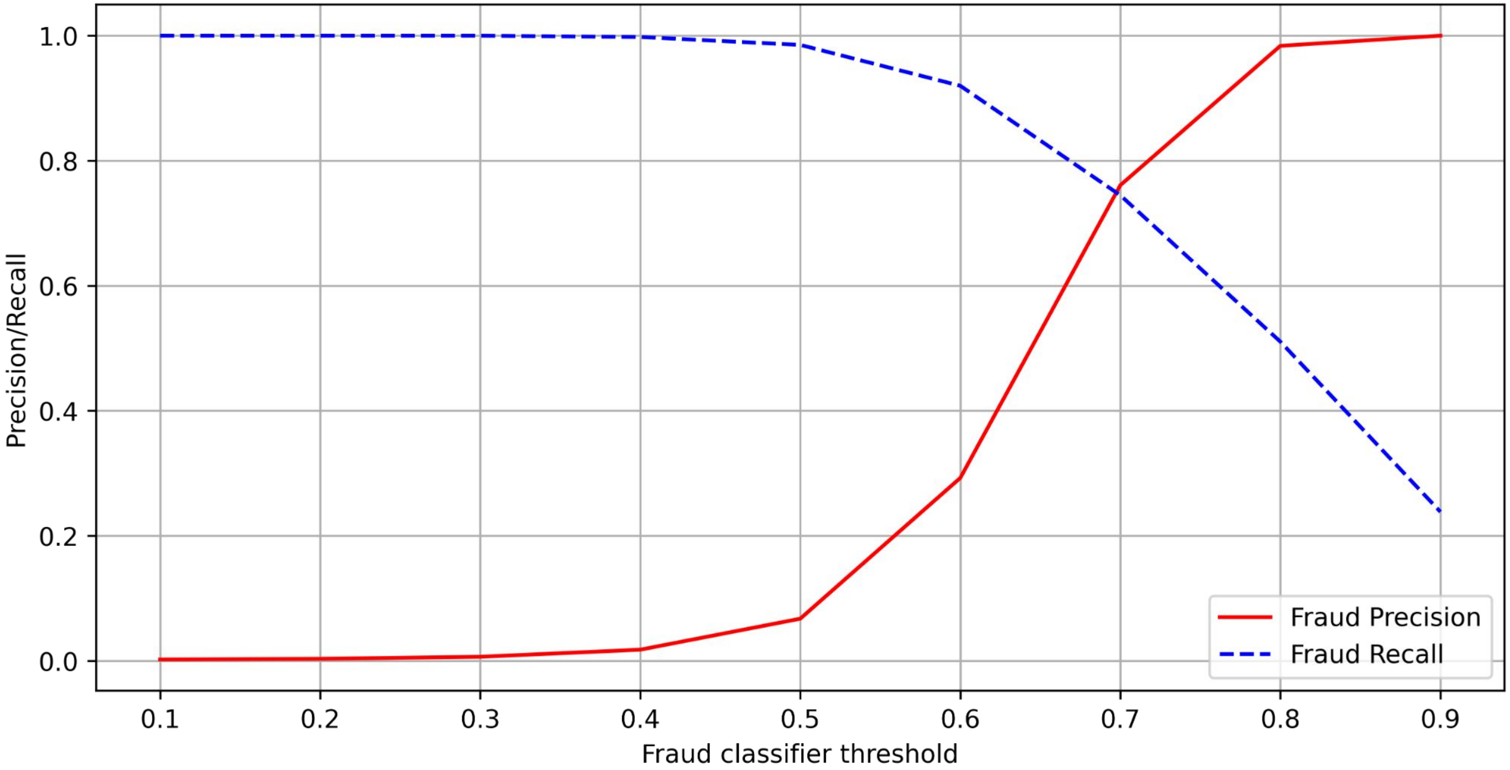




ML Process - Model evaluation

Performance measures: **classification tasks**

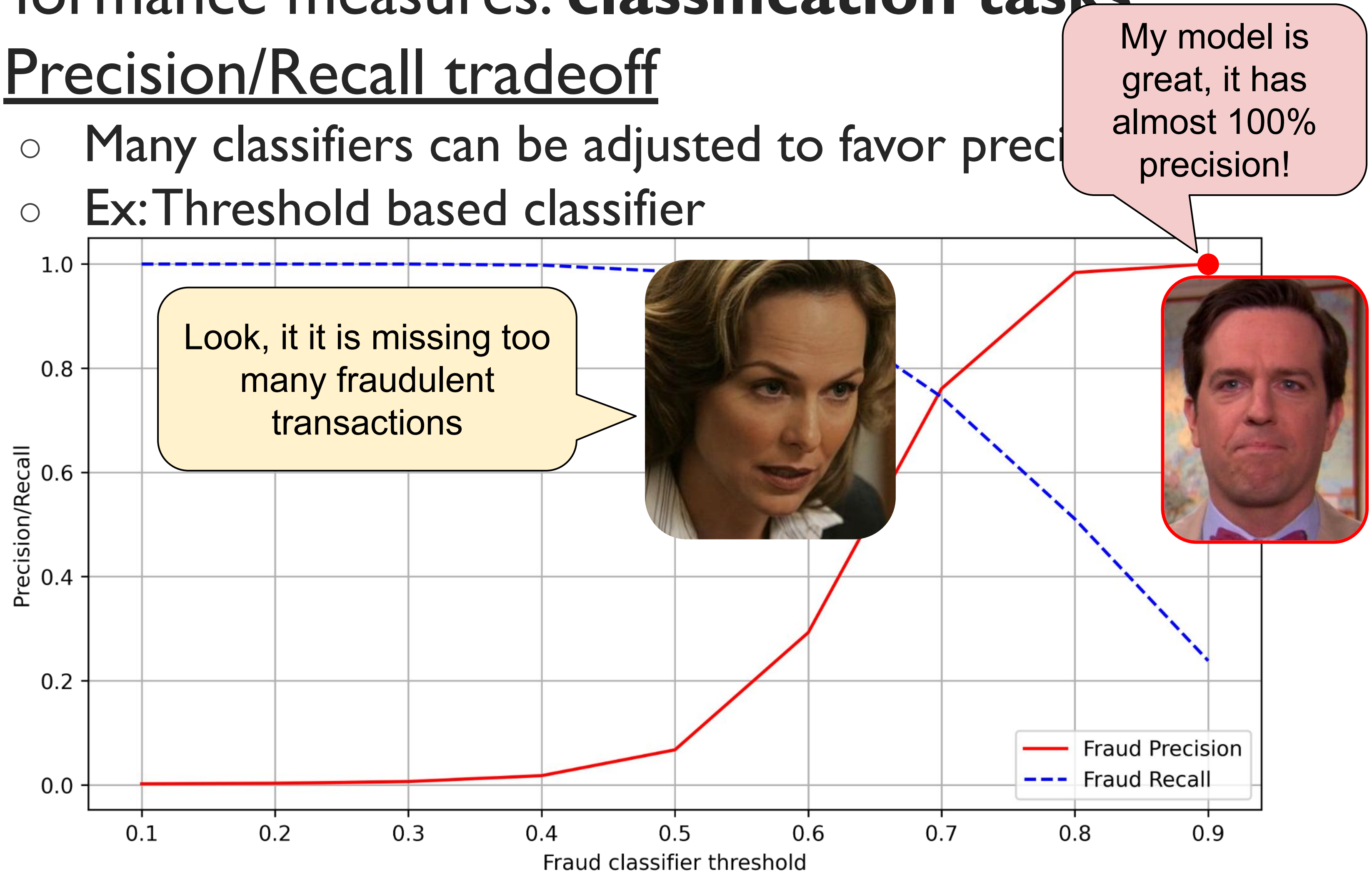
- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



ML Process - Model evaluation

Performance measures: **classification tasks**

- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor preci
  - Ex: Threshold based classifier



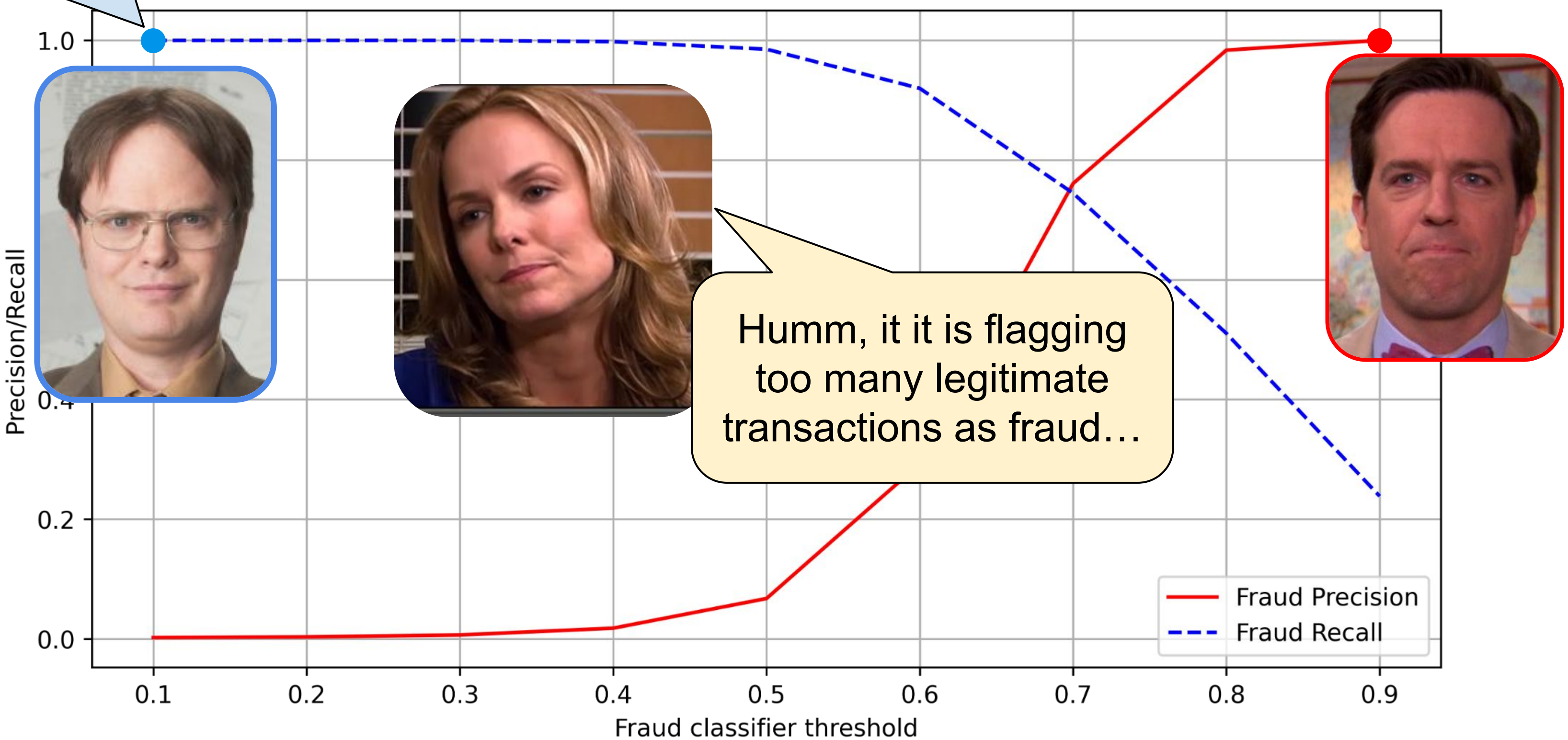
ML Process - Model evaluation

Performance measures: **classification tasks**

ion/Recall tradeoff

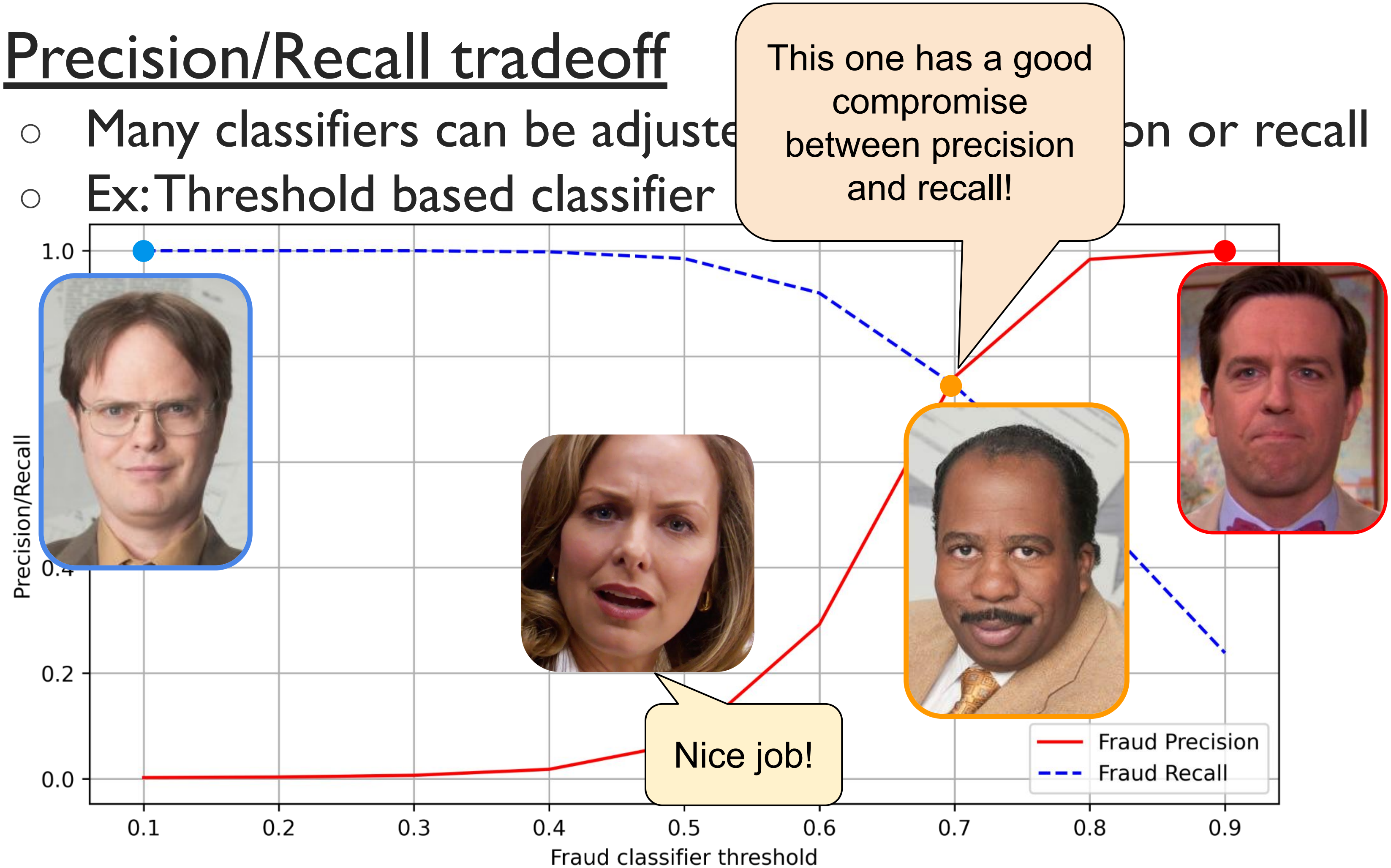
y classifiers can be adjusted to favor precision or recall

Threshold based classifier



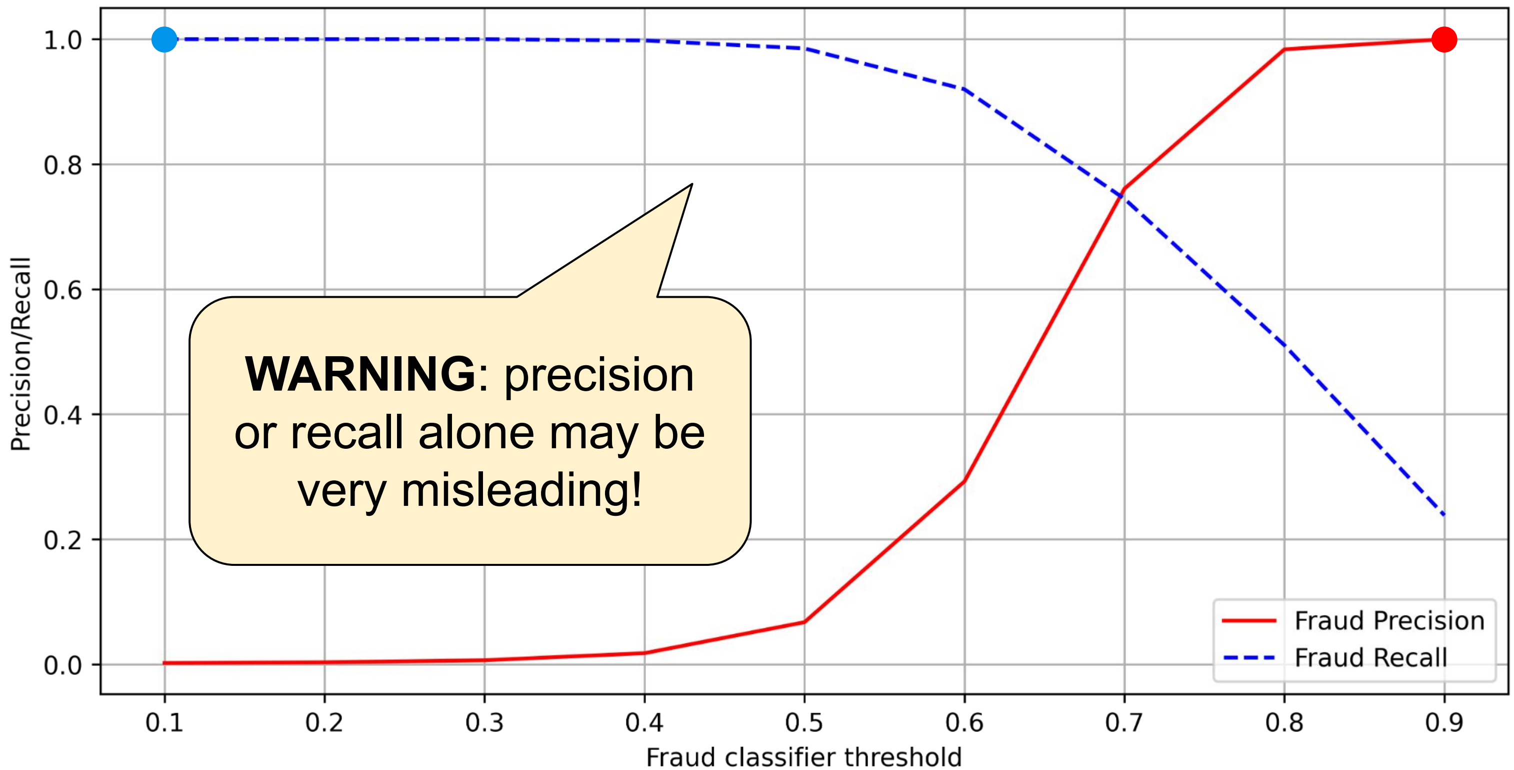
Performance measures: **classification tasks**

- Precision/Recall tradeoff
  - Many classifiers can be adjuste[…]on or recall
  - Ex: Threshold based classifier



Performance measures: **classification tasks**

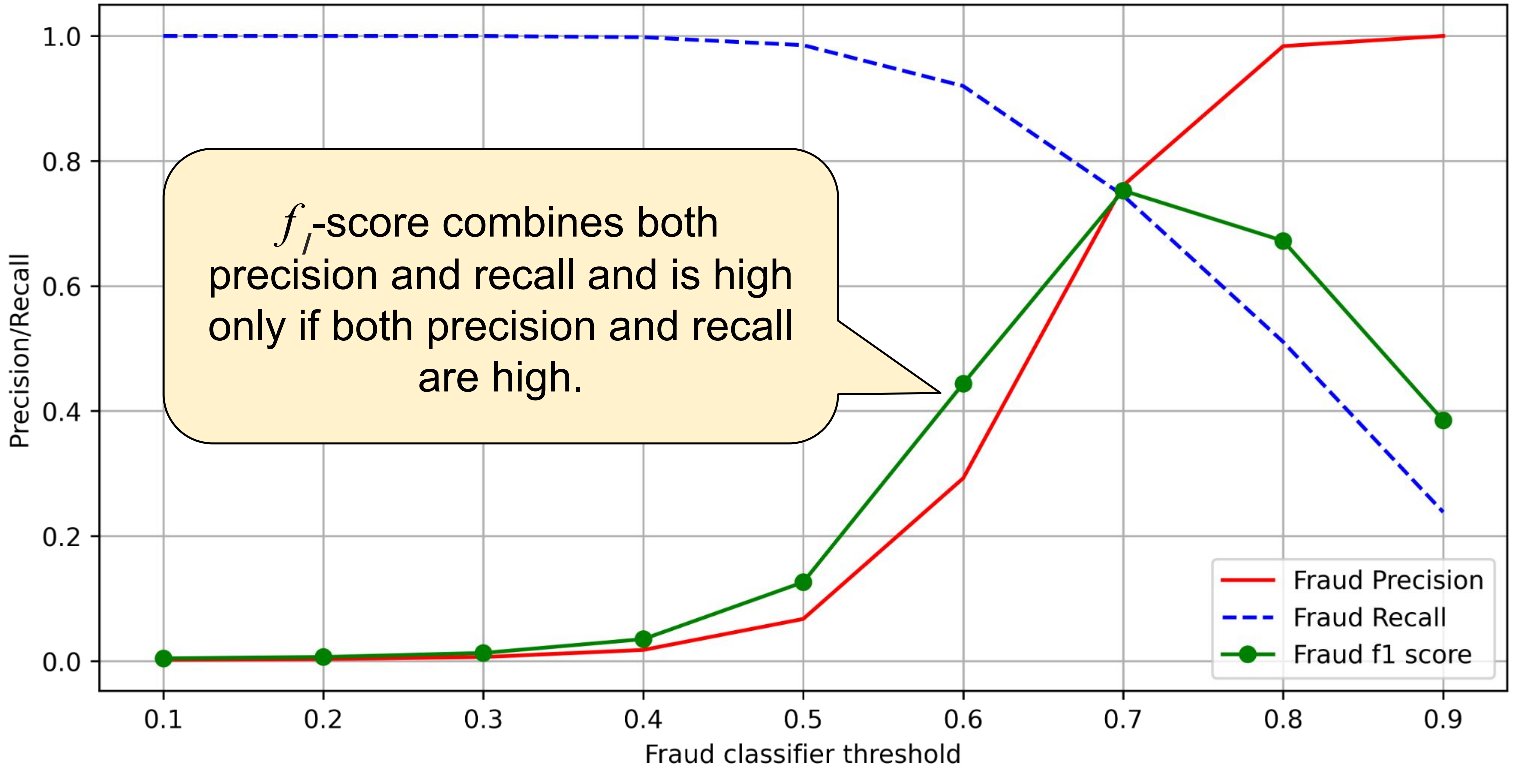
- Precision/Recall tradeoff
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



ML Process - Model evaluation

Performance measures: **classification tasks**

- <u>Precision/Recall tradeoff</u>
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier

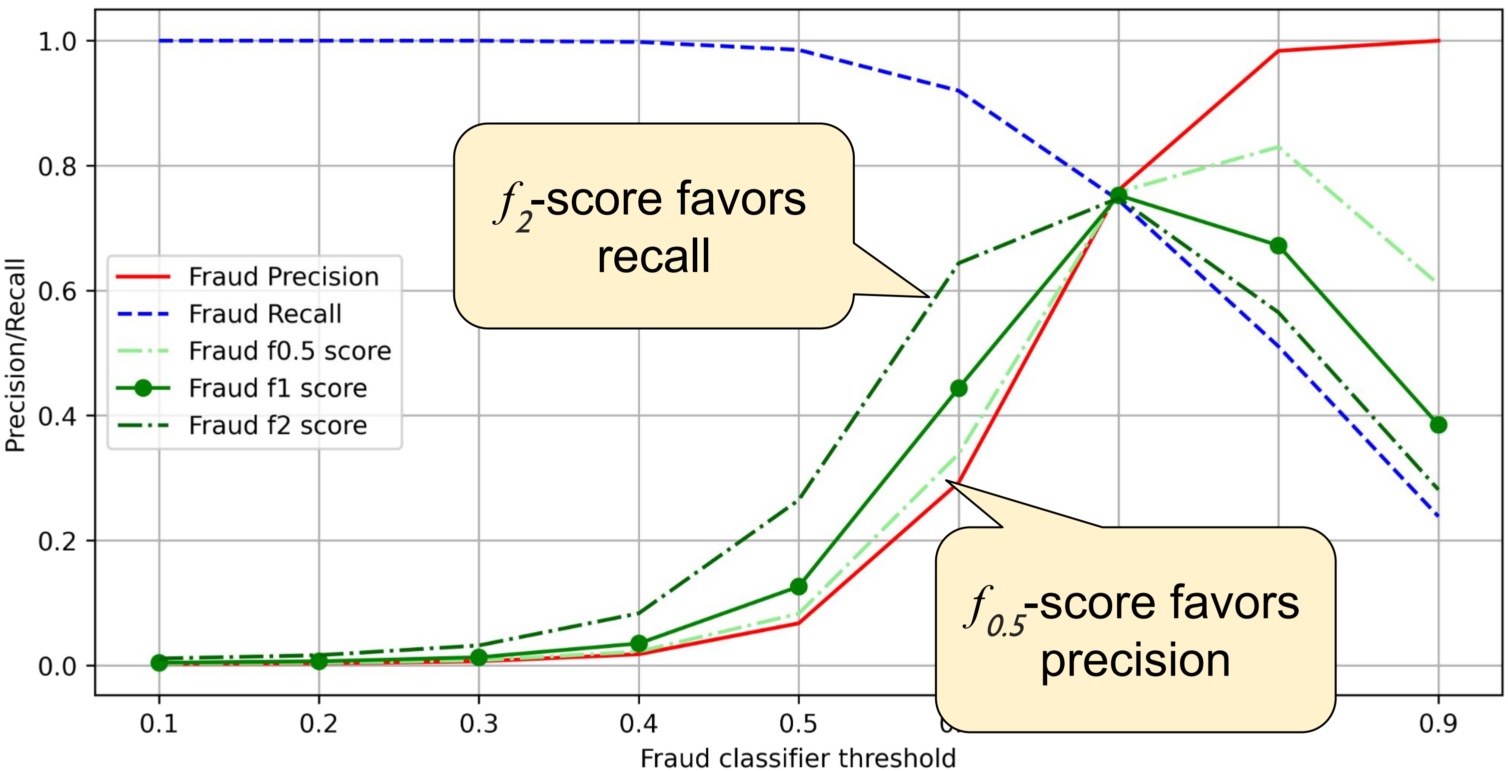


ML Process - Model evaluation

Performance measures: **classification tasks**

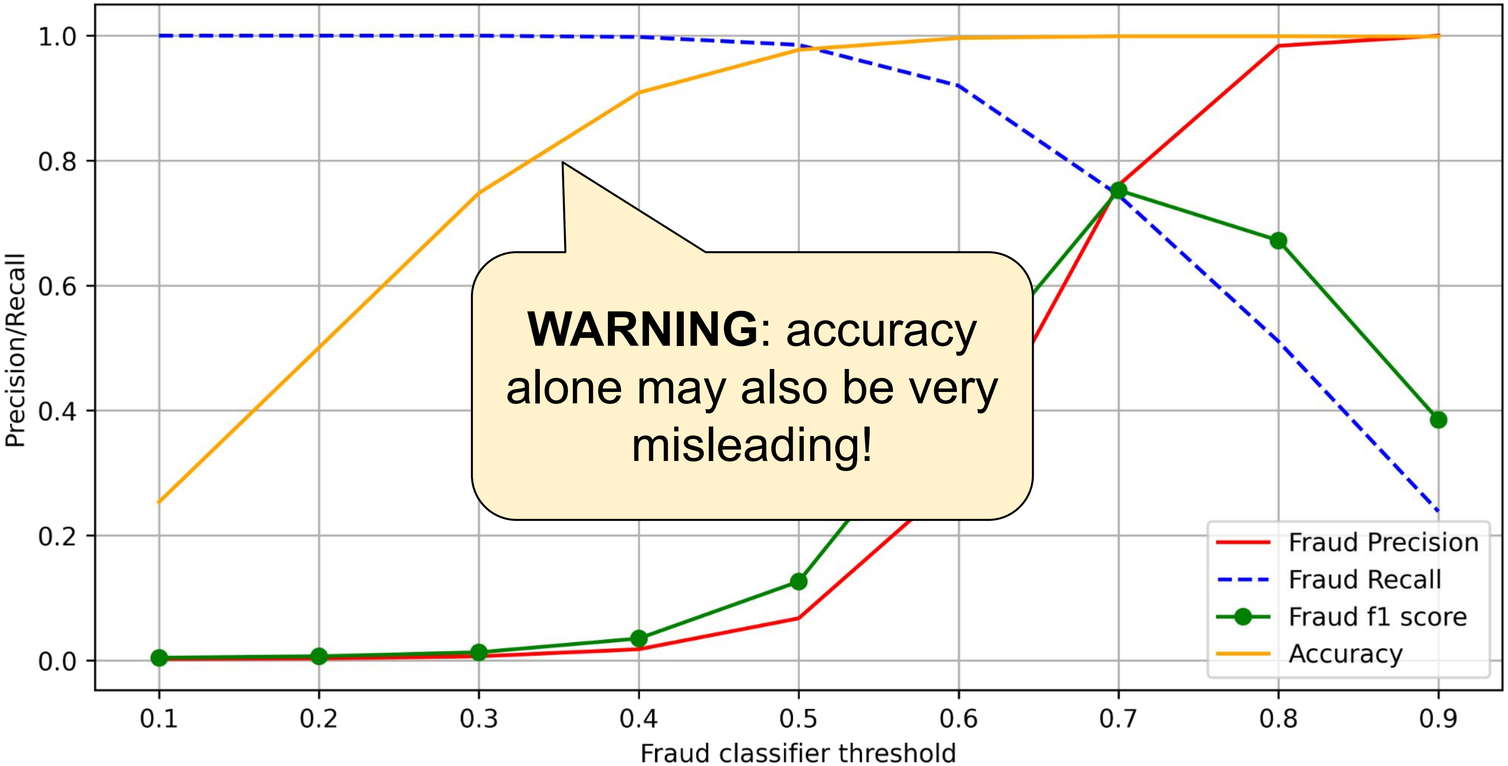
- Precision/Recall tradeoff
  - Many classifiers can be adjusted ... ll
  - Ex: Threshold based classifier

f_β-score can be used to assign more importance to precision or recall.



Performance measures: **classification tasks**

- <u>Precision/Recall tradeoff</u>
  - Many classifiers can be adjusted to favor precision or recall
  - Ex: Threshold based classifier



ML Process - Model evaluation

Performance measures: **classification tasks**

Error rate?

Precision?

Accuracy?

Recall?

F_β-score?

ML Process - Model evaluation

Performance measures: **classification tasks**

- No silver bullet
  - Best measure depends on the task being solved

Performance measures: **classification tasks**

- No silver bullet
  - Best measure depends on the task being solved

Task: Spam detection

Spam precision

Low ⟷ High

Low	High
High FP Miss important email	Low FP Do not miss important email

Spam recall

Low ⟷ High

Low	High
High FN Waste time going through spam	Low FN No time wasted reading spam

ML Process - Model evaluation

Performance measures: **classification tasks**

- N
  - pends on the task being solved

Focus on precision or f_β-score with low β (e.g., β=0.1)

Task: Spam ction

Spam precisio

Low ↔ High

✗ High FP Miss important email	Low FP Do not miss important email ✓

Spam recall

Low ↔ High

✗ High FN Waste time going through spam	Low FN No time wasted reading spam ✓

ML Process - Model evaluation

Performance measures: **classification tasks**

- No silver bullet
  - Best measure depends on the task being solved

Task: Spam detection

Spam precision

Low ⟷ High

High FP
Miss important email

Low FP
Do not miss important email

Spam recall

Low ⟷ High

High FN
Waste time going through spam

Low FN
No time wasted reading spam

Task: Illness detection

Illness precision

Low ⟷ High

High FP
Waste of doctors' time

Low FP
Not much waste of doctors' time

Illness recall

Low ⟷ High

High FN
Too many patients undiagnosed

Low FN
No (few) patients undiagnosed

ML Process - Model evaluation

Performance measures: **classification tasks**

- No silver bullet
  - Best measure depends on the task being sol

Focus on recall or f_β-score with high β (e.g., $\beta=10$)

Task: Spam detection

Spam precision

Low ← → High

High FP
Miss important email

Low FP
Do not miss important email

Spam recall

Low ← → High

High FN
Waste time going through spam

Low FN
No time wasted reading spam

Task: Illness detection

Illness precision

Low ← → High

High FP
Waste of doctors' time

Low FP
Not much waste of doctors' time

Illness recall

Low ← → High

High FN
Too many patients undiagnosed

Low FN
No (few) patients undiagnosed

Performance measures: **classification tasks**

- No silver bullet
  - Best measure depends on the task being solved

Precision — $f_{0.5}$-score — f_1-score — f_2-score — Recall

Instituto de Computação
UNIVERSIDADE ESTADUAL DE CAMPINAS

Capacitação profissional em tecnologias de Inteligência Artificial

Machine Learning Overview

Prof. Edson Borin
https://www.ic.unicamp.br/~edson
Institute of Computing - UNICAMP