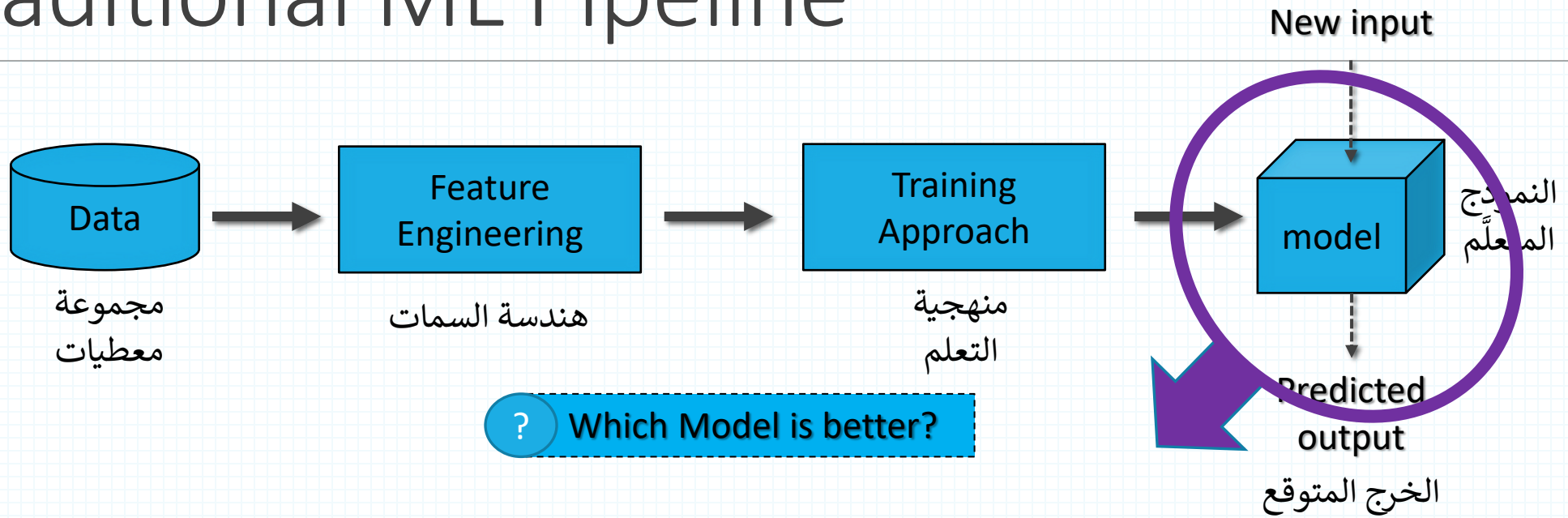
مقرر التعلم التلقائي | السنة الخامسة – هندسة المعلوماتية / الذكاء الصنعي | Week 3

Estimation Strategy and Evaluation Metrics

د. رياض سنبل

Access Course Materials

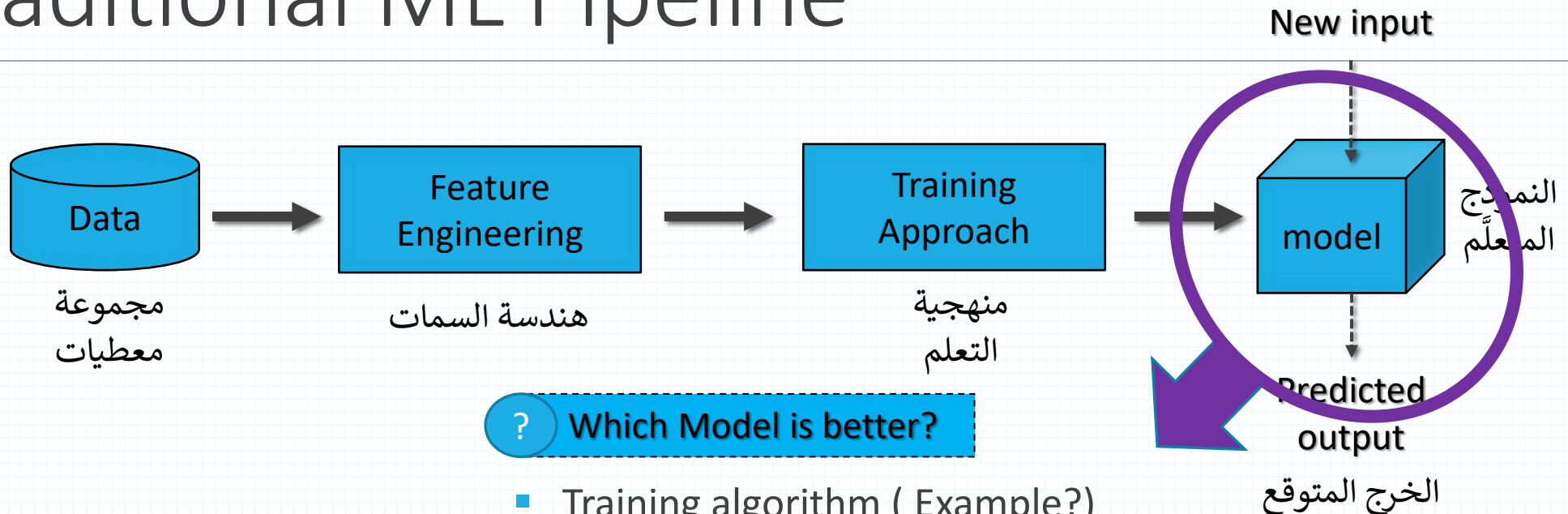
Traditional ML Pipeline



"All models are wrong; some are useful."

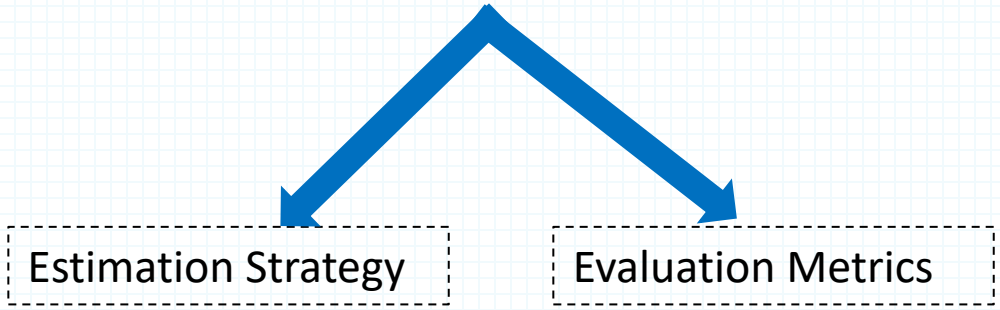
—George E. P. Box

Traditional ML Pipeline



? Which Model is better?

- Training algorithm (Example?)
- Parameters (Hyperparameters)?
- Handle unseen cases



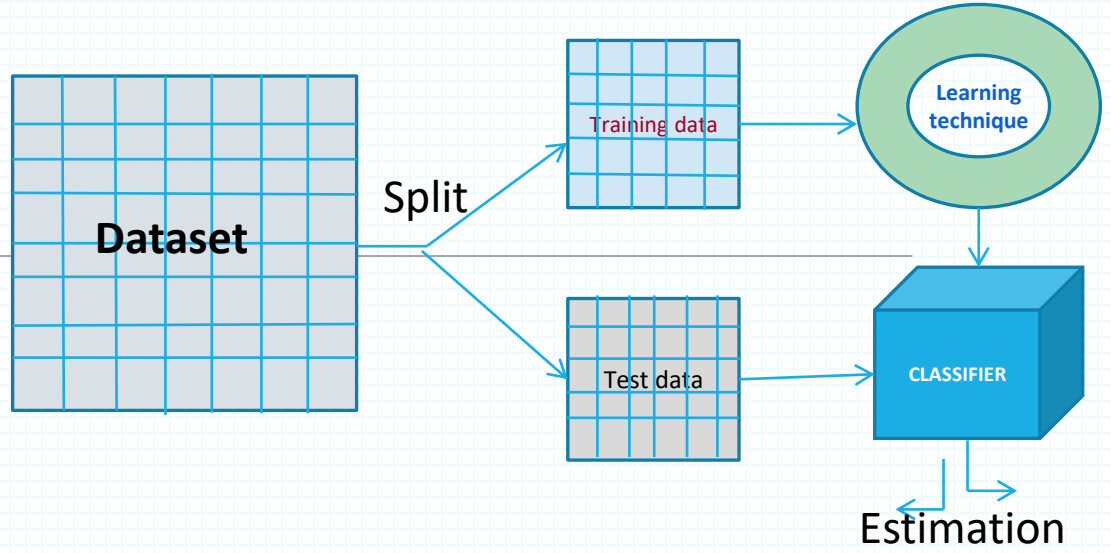
Estimation Strategy

Hyperparameters

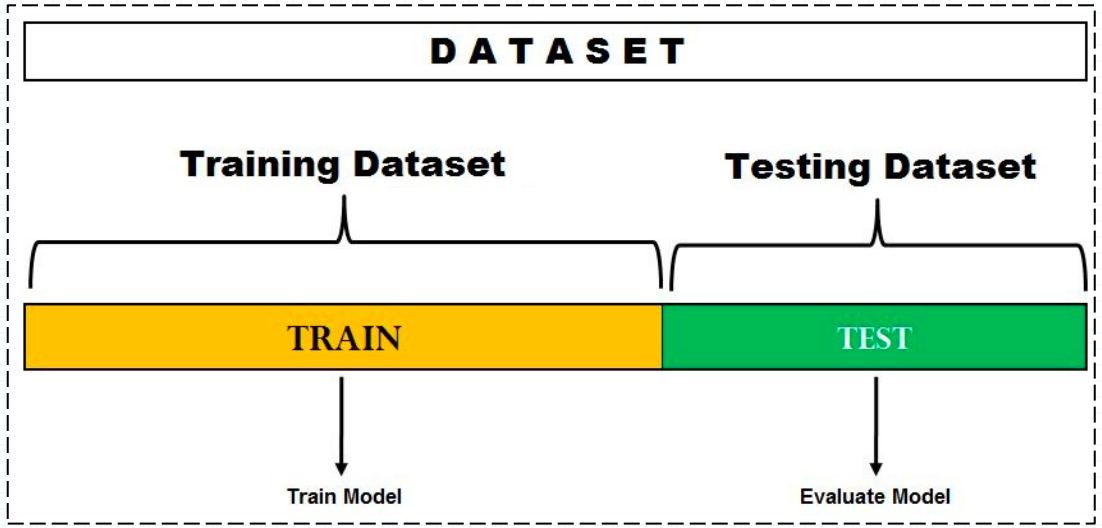
- Hyperparameters are the explicitly specified parameters that control the training process.
- What are the hyperparameters for decision trees?

class `sklearn.tree.`**`DecisionTreeClassifier`**(**, criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0*) [source]

The holdout method

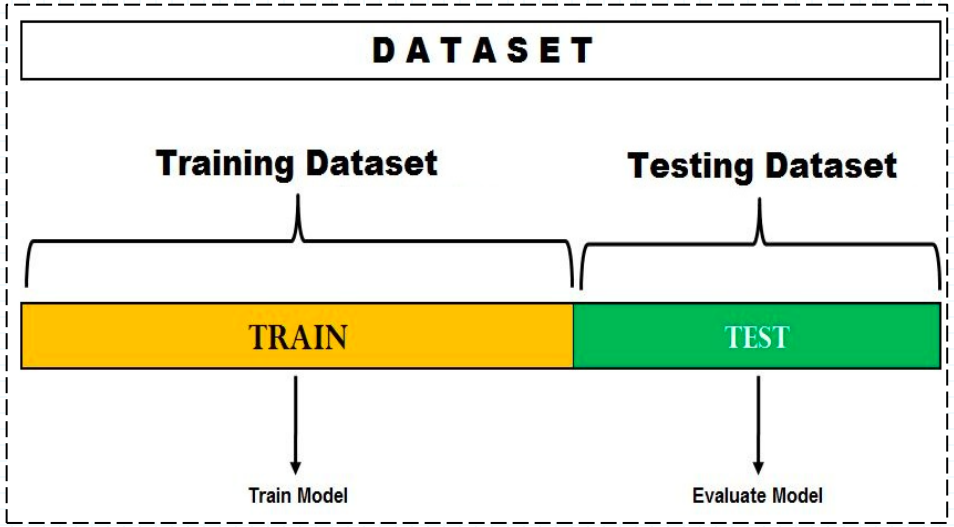


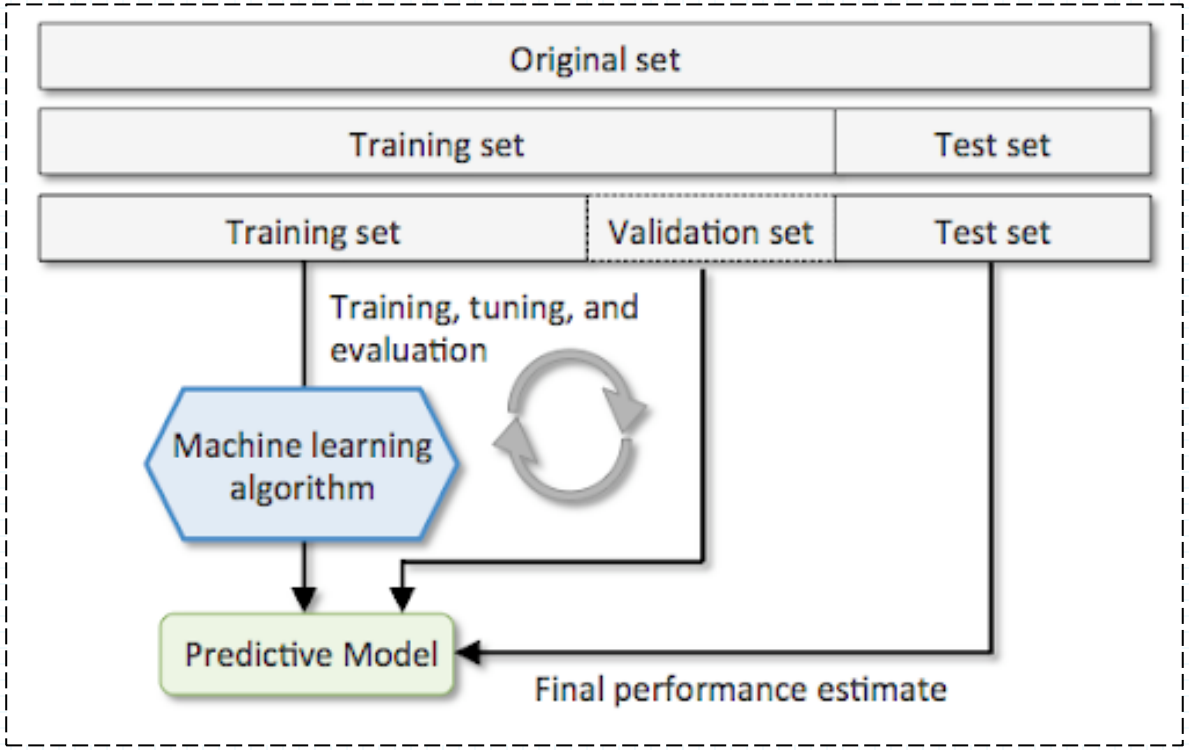
- Split dataset into two groups
  - Training set: used to train the classifier
  - Test set: used to estimate the error rate of the trained classifier.
- Ratio of training and testing sets is at the discretion of analyst;
  - Typically **1:1 or 2:1**, and there is a trade-off between these sizes of these two sets.
  - If the training set is too large, then model may be good enough, but estimation may be less reliable due to small testing set and vice-versa.



Holdout method

But how can we tune hyper-parameters?



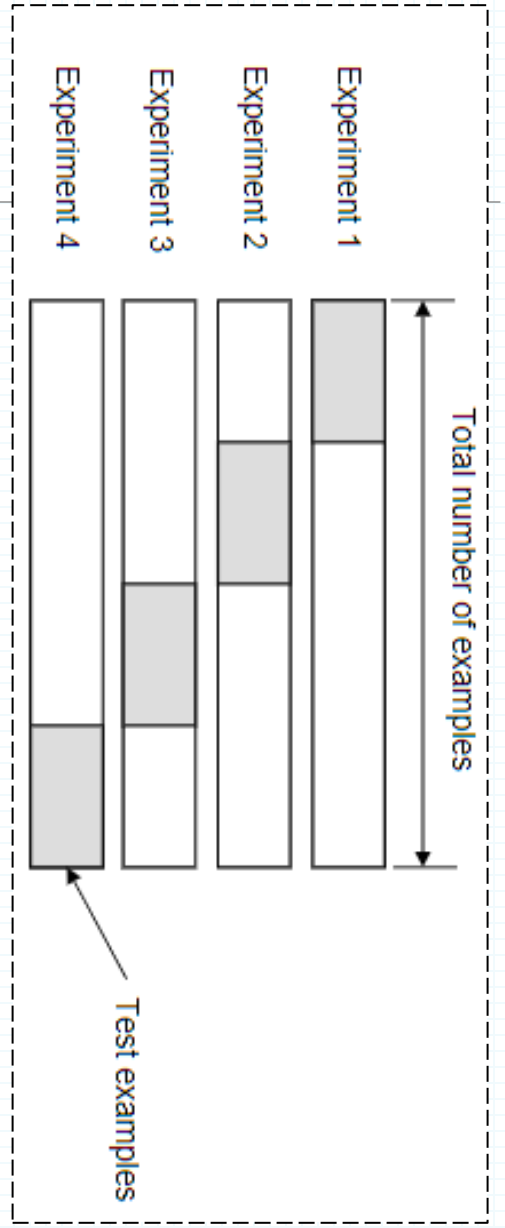


Holdout method - Drawbacks

- The holdout method has **two basic drawbacks**
  - In problems where we have a sparse dataset we may not be able to afford **the "luxury" of setting aside a portion** of the dataset for testing
  - Since it is **a single train-and-test experiment**, the holdout estimate of error rate will be misleading if we happen to get an "unfortunate" split
- The limitations of the holdout can be overcome with a family of re-sampling methods at the expense of higher computational cost
  - K-Fold Cross-Validation
  - Leave-one-out Cross-Validation

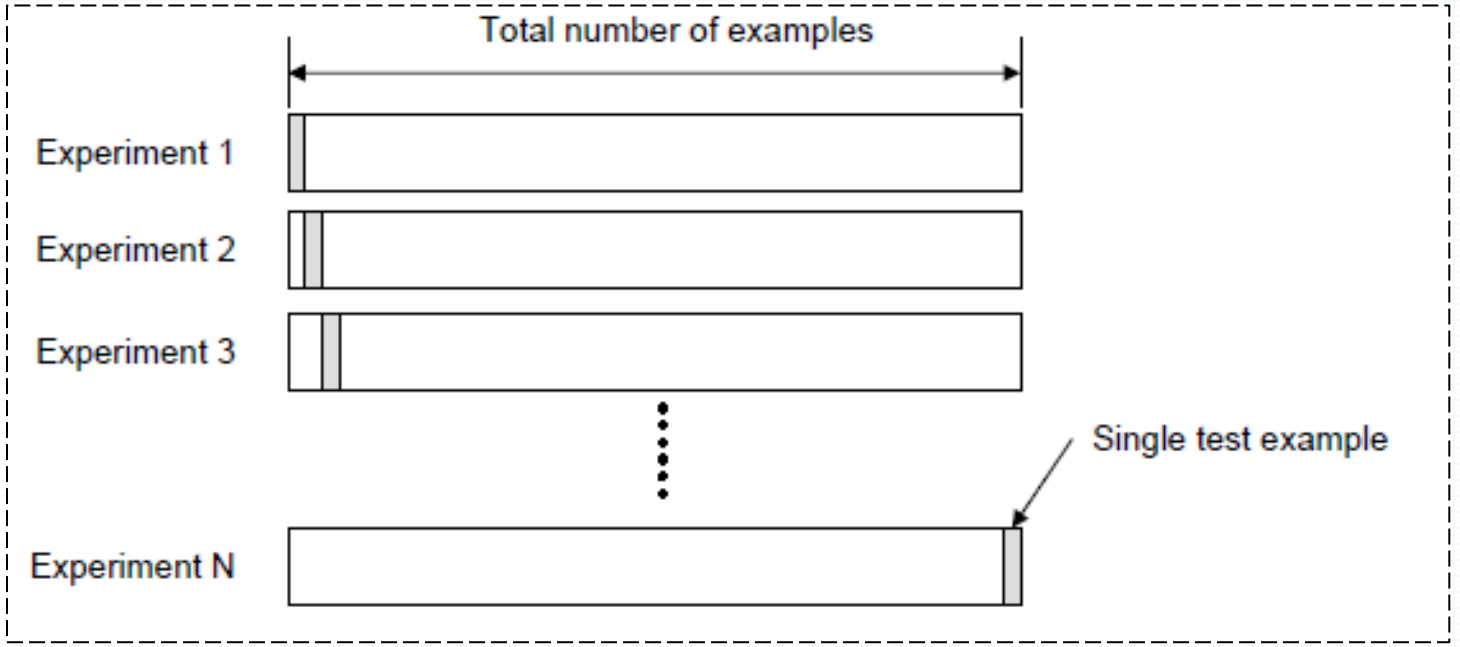
K-Fold Cross-validation

- Create a K-fold partition of the dataset
- For each of K experiments, use K-1 folds for training and a different fold for testing
- The advantage of K-Fold Cross validation is that **all the examples** in the dataset are eventually used for both training and testing
- The true error is estimated as the average error rate on test examples



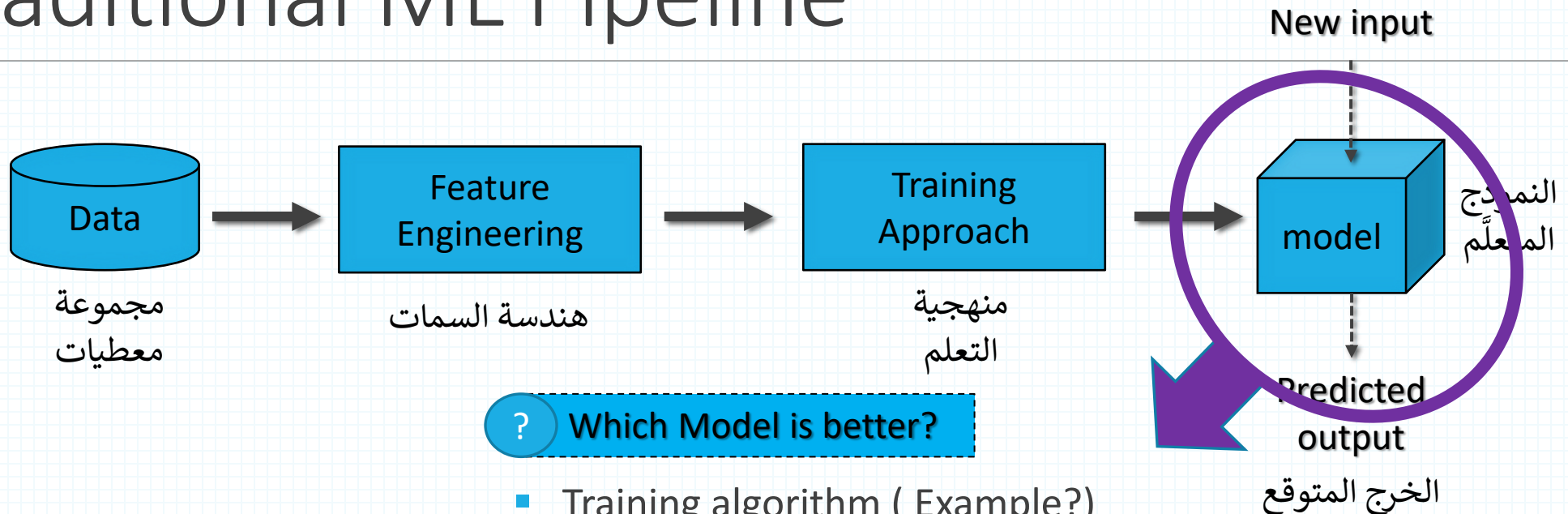
Leave-one-out Cross-Validation

- Leave-one-out is the degenerate case of K-Fold Cross Validation, where K is chosen as the total number of examples
  - For a dataset with N examples, perform N experiments
  - For each experiment use N-1 examples for training and the remaining example for testing



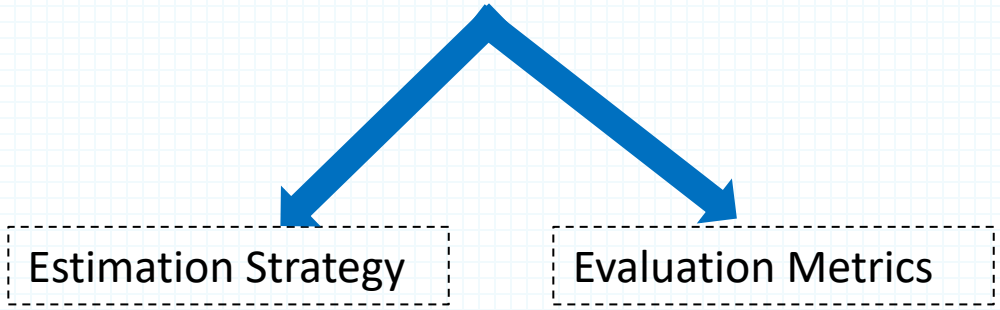
Evaluation Metrics

Traditional ML Pipeline



? Which Model is better?

- Training algorithm (Example?)
- Parameters (Hyperparameters)?
- Handle unseen cases

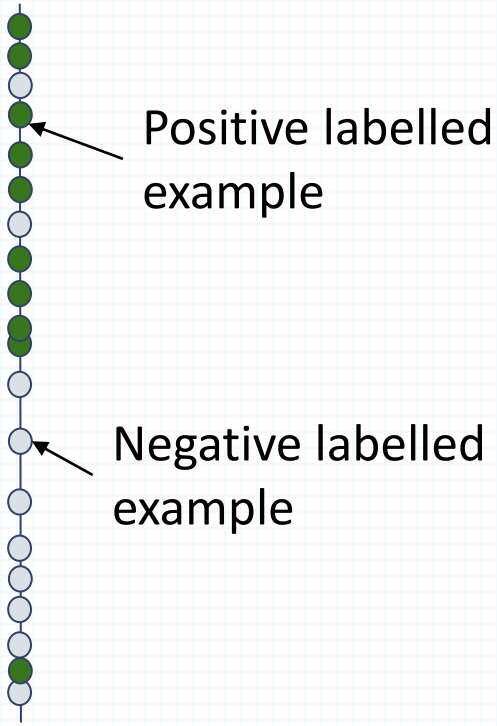


Two types of models

- Models that output a categorical class directly (K Nearest neighbor, Decision tree)
- Models that output a real valued score (SVM, Logistic Regression)
  - Score could be margin (SVM), probability (LR, NN)
  - Need to pick a threshold

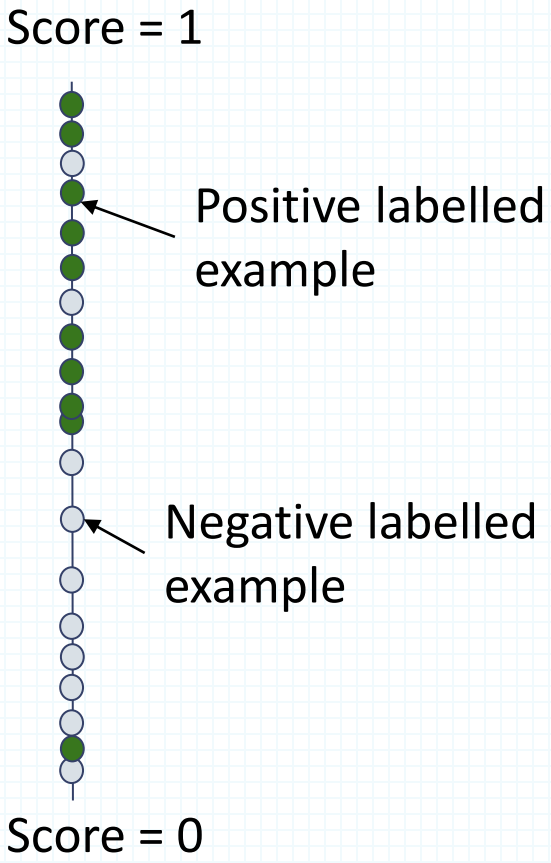
Example

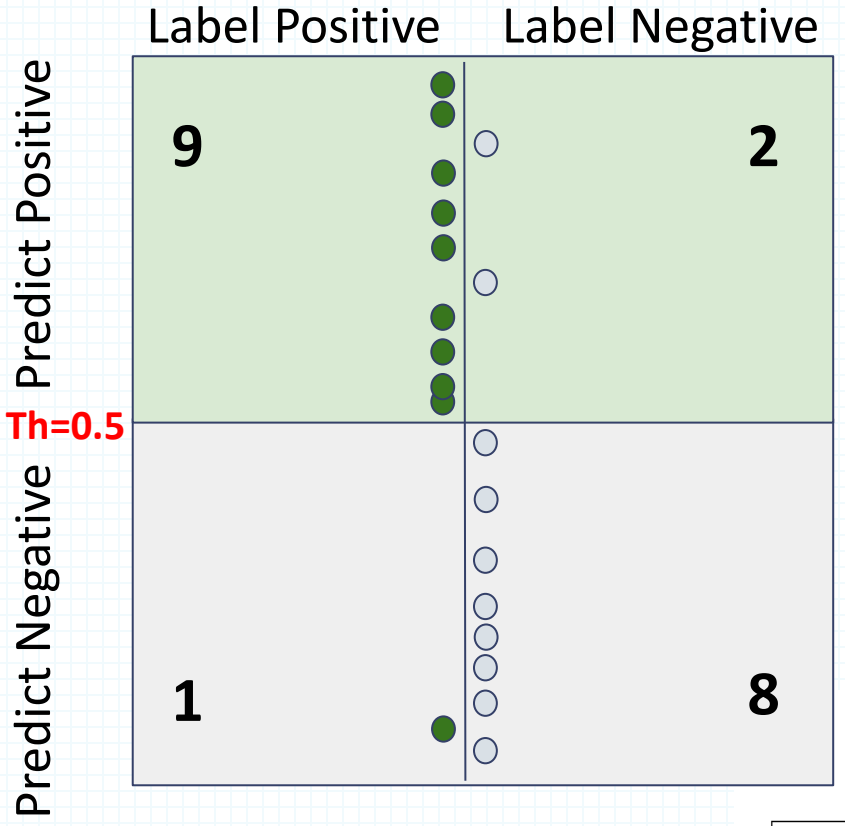
Score = 1



Score = 0

Example

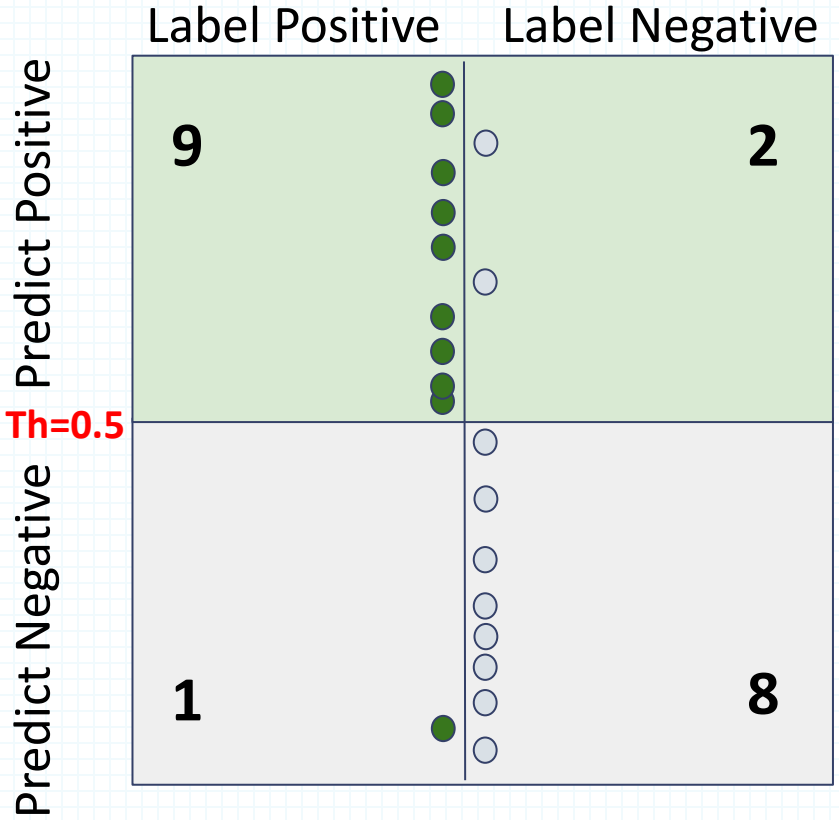




Th	TP	TN	FP	FN
0.5	9	8	2	1

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Evaluation Metrics



TP	TN	FP	FN	Acc
9	8	2	1	0.85

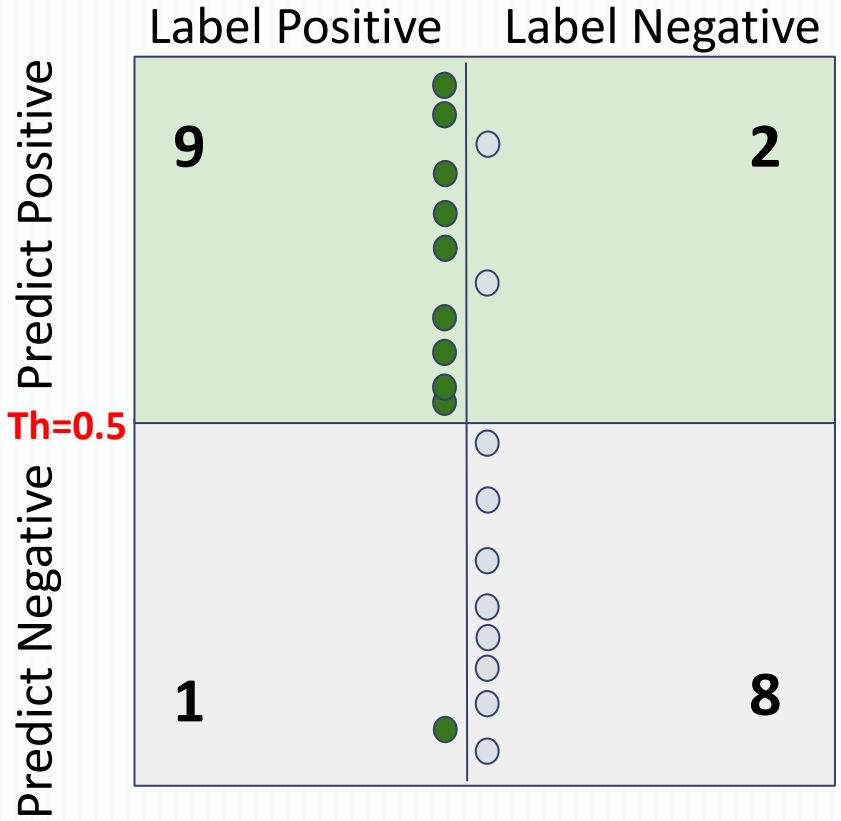
Limitation of Accuracy

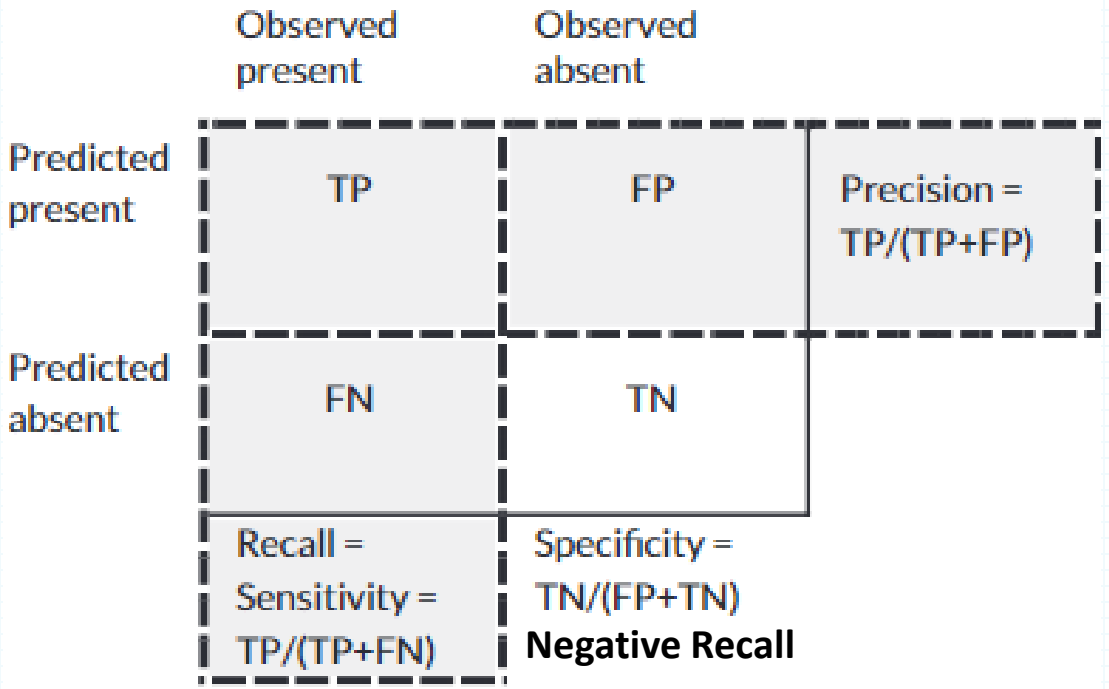
- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
- Accuracy is misleading because model does not detect any class 1 example

Evaluation Metrics

TP	TN	FP	FN	Acc	P	R	F1
9	8	2	1	0.85	0.81	0.90	0.857





F1-score

- It is usually better to compare models by means of one number only.
- The F1−score can be used to combine precision and recall

	Precision(P)	Recall (R)	Average	F_1 Score
Algorithm 1	0.5	0.4	0.45	0.444
Algorithm 2	0.7	0.1	0.4	0.175
Algorithm 3	0.02	1.0	0.51	0.0392

The best is Algorithm 1

Algorithm 3 predict always 1

Average says not correctly that Algorithm 3 is the best

$$\text{Average} = \frac{P + R}{2} \qquad F_1\text{score} = 2\frac{PR}{P + R}$$

- $P = 0$ or $R = 0 \Rightarrow F_1\text{score} = 0$
- $P = 1$ and $R = 1 \Rightarrow F_1\text{score} = 1$

Multiclass Classifier

- Having m classes, confusion matrix is a table of size m×m , where, element at (i, j) indicates the number of instances of class i but classified as class j.

Class	C_1	C_2	C_3	C_4	C_5	C_6
C_1	52	10	7	0	0	1
C_2	15	50	6	2	1	2
C_3	5	6	6	0	0	0
C_4	0	2	0	10	0	1
C_5	0	1	0	0	7	1
C_6	1	3	0	1	0	24

- What is the accuracy?

Multiclass Classifier

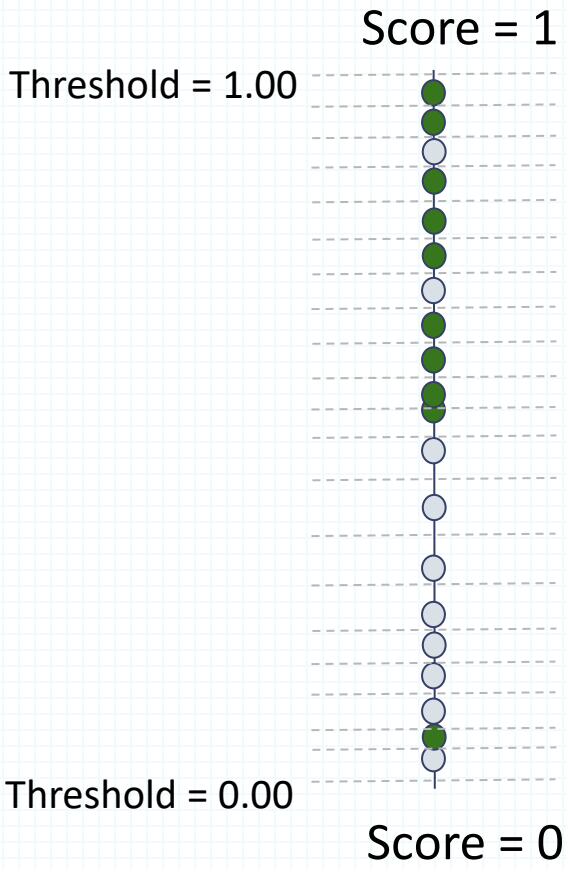
- Precision, Recall, and F1-score are calculated for each class.
  - A large confusion matrix of m*m can be considered into 2*2 matrix.

Class	C_1	C_2	C_3	C_4	C_5	C_6
C_1	52	10	7	0	0	1
C_2	15	50	6	2	1	2
C_3	5	6	6	0	0	0
C_4	0	2	0	10	0	1
C_5	0	1	0	0	7	1
C_6	1	3	0	1	0	24

C1	+	-
+	52	18
-	21	123

- Finally, we can merge overall scores (ex: using weighted average)
  - What is the overall F1-score in the previous example?

Precision vs Sensitivity

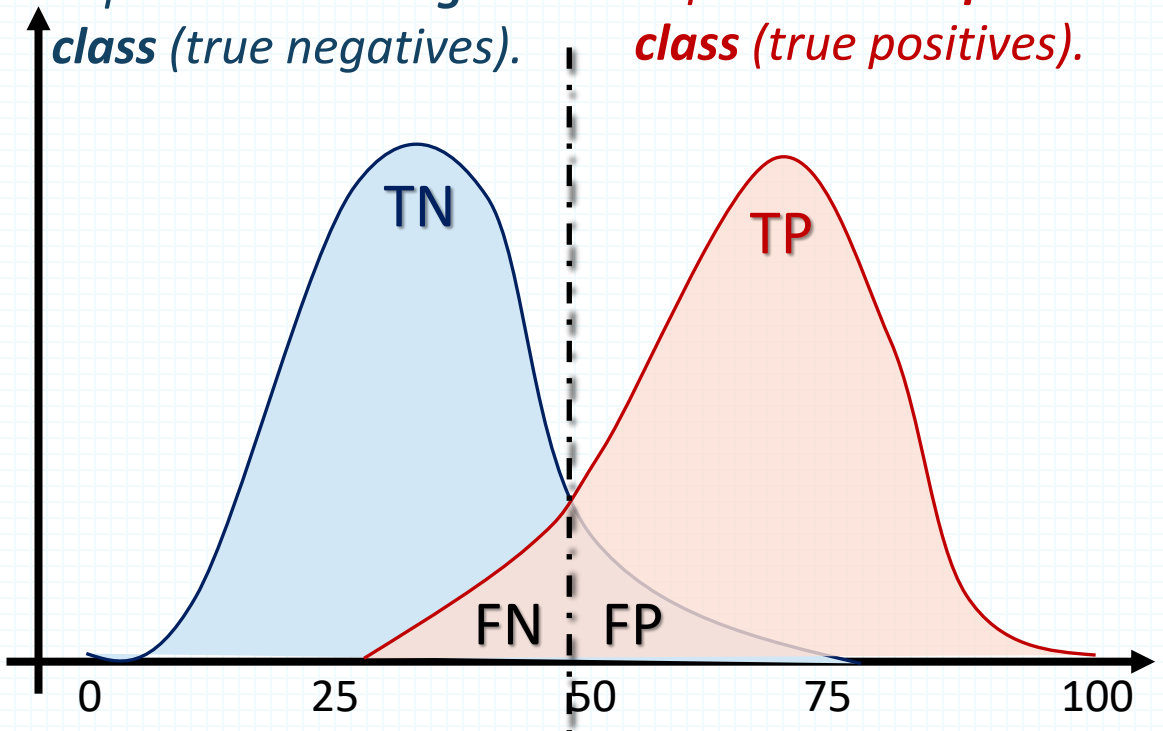


{Precision, Specificity} vs Recall/Sensitivity

Threshold	TP	TN	FP	FN	Accuracy	Precision	Recall	F1
1.00	0	10	0	10	0.50	1	0	0
0.95	1	10	0	9	0.55	1	0.1	0.182
0.90	2	10	0	8	0.60	1	0.2	0.333
0.85	2	9	1	8	0.55	0.667	0.2	0.308
0.80	3	9	1	7	0.60	0.750	0.3	0.429
0.75	4	9	1	6	0.65	0.800	0.4	0.533
0.70	5	9	1	5	0.70	0.833	0.5	0.625
0.65	5	8	2	5	0.65	0.714	0.5	0.588
0.60	6	8	2	4	0.70	0.750	0.6	0.667
0.55	7	8	2	3	0.75	0.778	0.7	0.737
0.50	8	8	2	2	0.80	0.800	0.8	0.800
0.45	9	8	2	1	0.85	0.818	0.9	0.857
0.40	9	7	3	1	0.80	0.750	0.9	0.818
0.35	9	6	4	1	0.75	0.692	0.9	0.783
0.30	9	5	5	1	0.70	0.643	0.9	0.750
0.25	9	4	6	1	0.65	0.600	0.9	0.720
0.20	9	3	7	1	0.60	0.562	0.9	0.692
0.15	9	2	8	1	0.55	0.529	0.9	0.667
0.10	9	1	9	1	0.50	0.500	0.9	0.643
0.05	10	1	9	0	0.55	0.526	1	0.690
0.00	10	0	10	0	0.50	0.500	1	0.667

Sensitivity vs Specificity

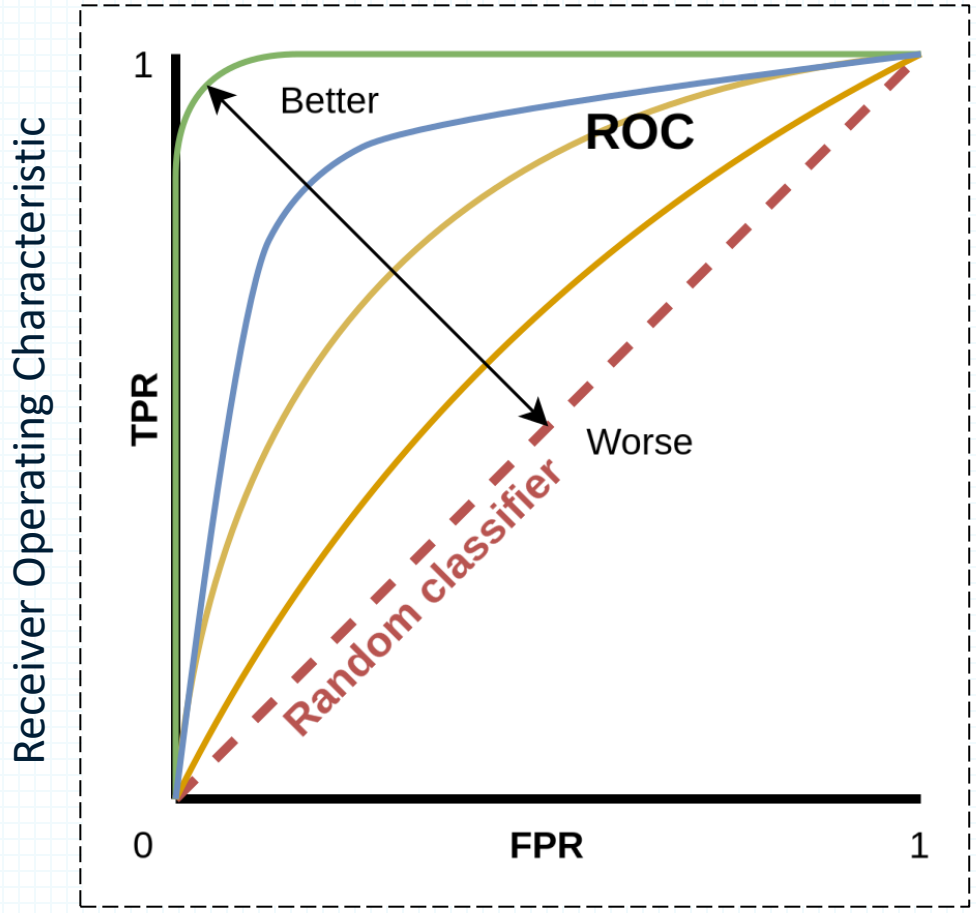
	Observed present	Observed absent	
Predicted present	TP	FP	Precision = TP/(TP+FP)
Predicted absent	FN	TN	
	Recall = Sensitivity = TP/(TP+FN)	Specificity = TN/(FP+TN) **Negative Recall**	



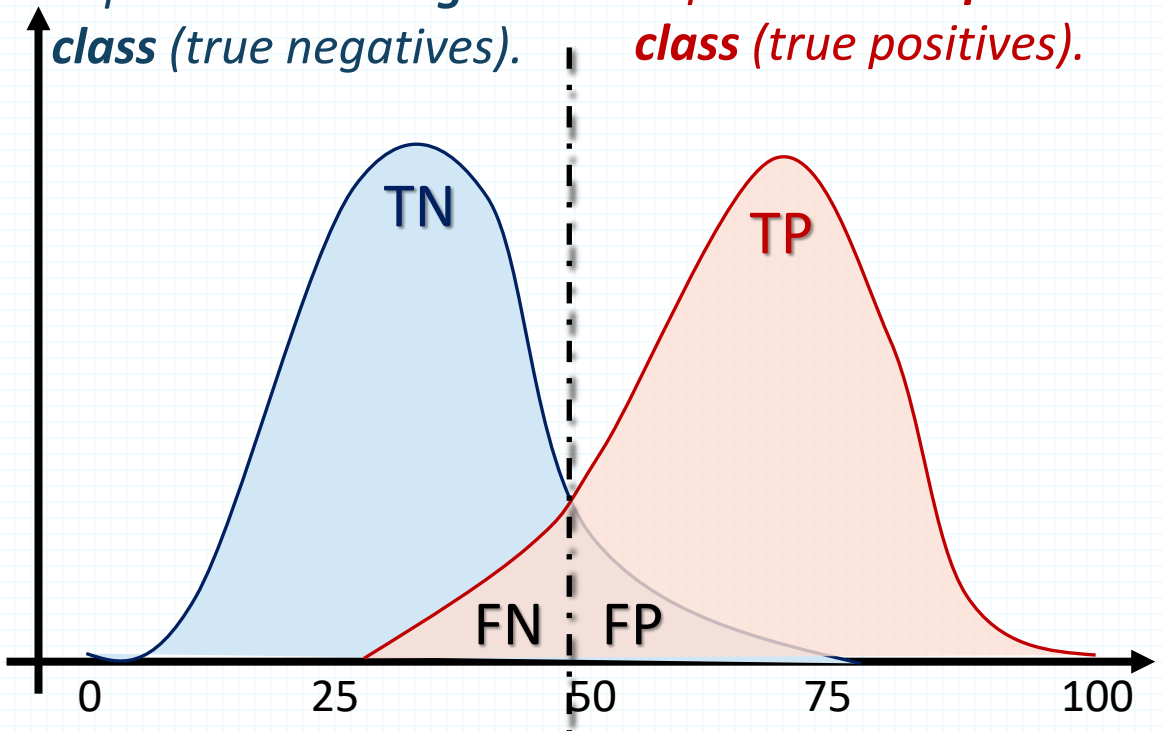
*increases **sensitivity** but may increase **FPs** (reduces specificity).*

*increases **specificity** but may increase **FNs** (reduces sensitivity).*

Sensitivity vs Specificity



TPR (= Sensitivity or Recall) vs FPR (= 1 - Specificity)



*increases **sensitivity** but may increase **FPs** (reduces specificity).*

*increases **specificity** but may increase **FNs** (reduces sensitivity).*

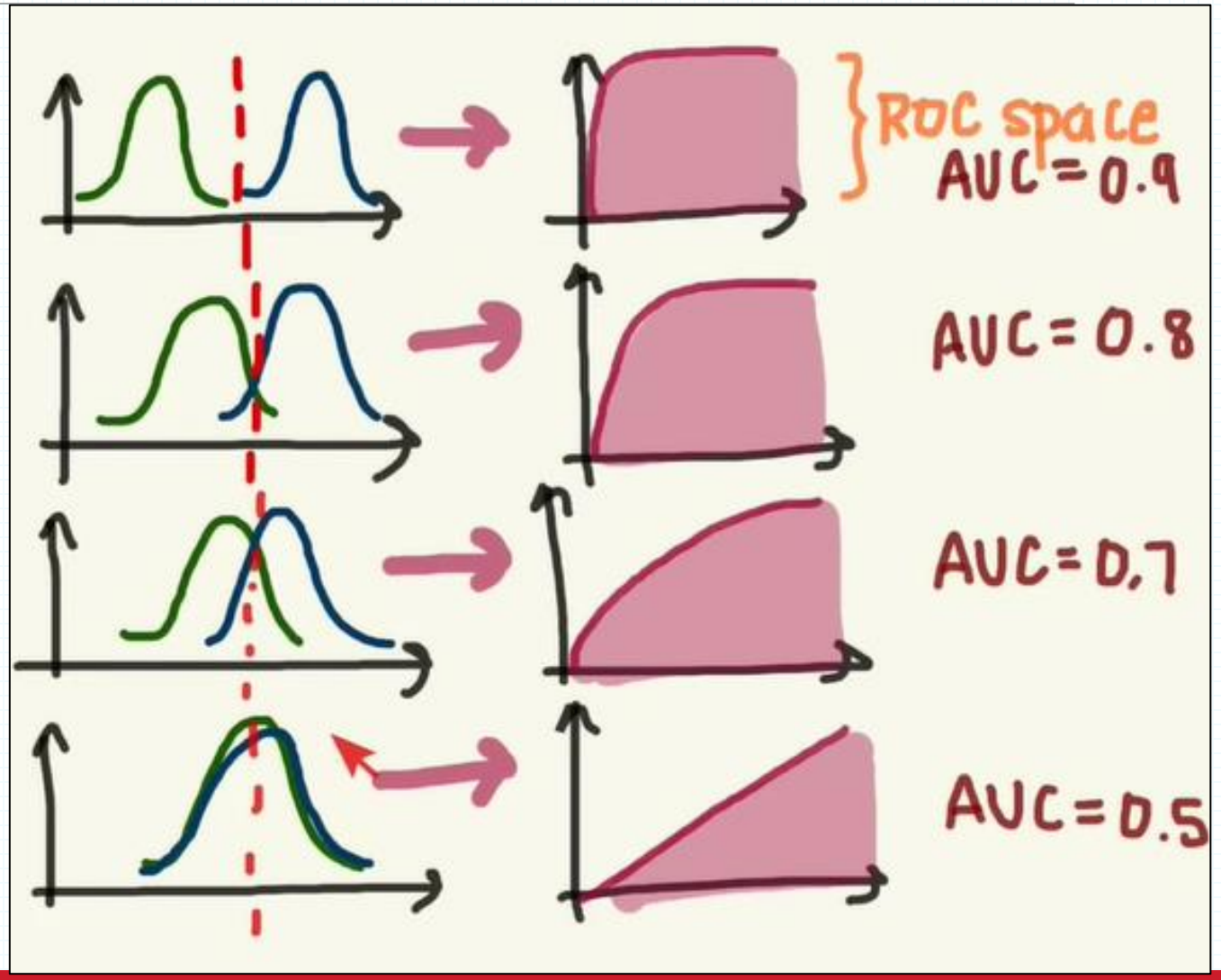
The Area Under the Curve (AUC)

What Does AUC Represent?

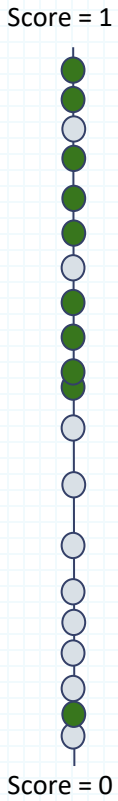
- **AUC = 1.0** → Perfect classifier
- **AUC = 0.5** → Random classifier
- **AUC < 0.5** → Worse than random

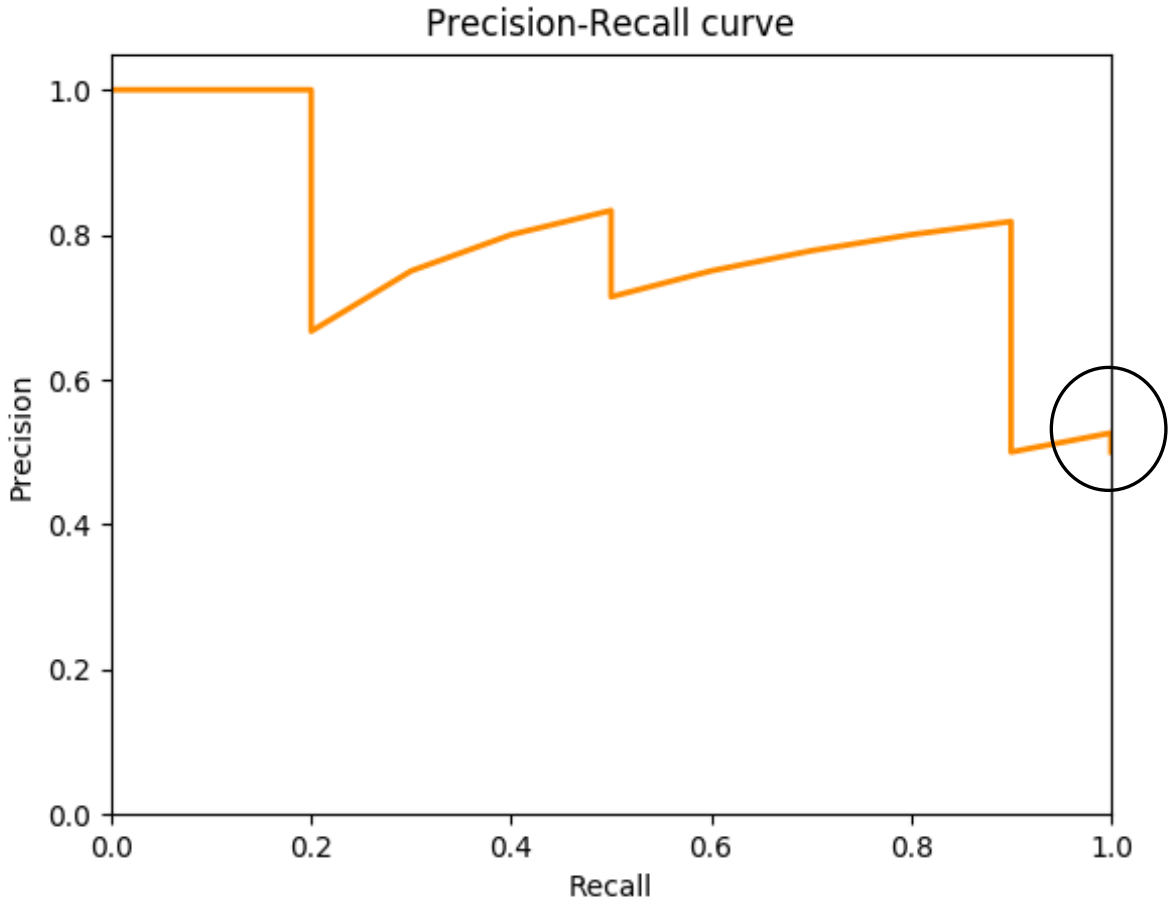
what AUC tells us about the model:

- ✓ Class Separation Power
- ✓ Stability Across Thresholds
- ✓ It doesn't tell you how good a specific threshold is.



Precision-Recall curve (PRC)





- Precision Recall curve represents a different tradeoff.
- End of curve at right cannot be lower than prevalence.

$$Prevalence = \frac{\#positives}{\#positives + \#negatives}$$

- Better for **Imbalanced** datasets (Focuses on positive class only)