Performance measure

Performance metrics

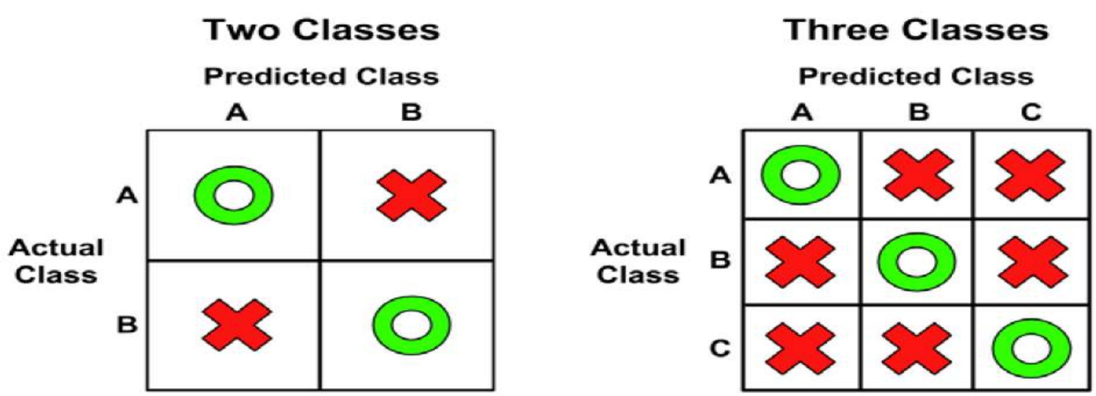
- Sensitivity/specificity
- Precision/Recall, F1 score
- ROC curve, AUC

SENSITIVITY & SPECIFICITY

Performance measure

- For a test data *X*, measure of closeness between true label Y_{true} and predicted Y_{pred}
  - Rather than how fast it takes to classify or learn the classifier, scalability, etc.

- Confusion matrix



Binary Classification

1100 test images

Classifier 1

	Predicted '2'	Predicted 'Not 2'
True '2'	70	30
True 'Not 2'	140	860

Classifier 2

	Predicted '2'	Predicted 'Not 2'
True '2'	20	80
True 'Not 2'	50	950

Which classifier is better?

Performance measure

- The class of interest is known as the positive class
- All the others are known as negative

- True Positive (TP): Correctly classified as the class of interest
- False Negative (FN): Incorrectly classified as not the class of interest
- False Positive (FP): Incorrectly classified as the class of interest
- True Negative (TN): Correctly classified as not the class of interest

	Predicted class		
Actual class		Class=1	Class=0
	Class=1	TP	FN
	Class=0	FP	TN

Actual class	Predicted class
1	0
0	0
0	0
1	1
1	0
0	0
0	0
0	1

X	FN
O	TN
O	TN
O	TP
X	FN
O	TN
O	TN
X	FP

$$Accuracy = \frac{5}{8} = 62.5\%$$

Metrics for Performance Evaluation

	Predicted class		
Actual class		Class=1	Class=0
	Class=1	A	B
	Class=0	C	D

- Widely-used metric:

$$Accuracy = \frac{A+D}{A+B+C+D} = \frac{TP+TN}{TP+TN+FP+FN}$$

Num. correctly classified / total num. of test data

Errror = 1 - Accuracy

Limitation of Accuracy

- In binary classification
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10
- If a classifier predicts everything to be class 0,
  accuracy is:
  - Accuracy is misleading because the classifier does not detect any Class 1 example

		Predicted class	
		Class=1	Class=0
Actual class	Class=1		
	Class=0		

Sensitivity & Specificity

	Predicted class		
Actual class		Class=1	Class=0
	Class=1	TP	FN
	Class=0	FP	TN

$$Sensitivity = \frac{TP}{TP + FN}$$

True Positive rate

$$Specificity = \frac{TN}{FP + TN}$$

True Negative rate

Example

- Confusion matrix

	Predicted class	
Actual class	Class=1	Class=0
Class=1	4	1
Class=0	2	3

  - Accuracy=
  - Misclassification error=1-Accuracy=
  - Sensitivity (true positive, recall) =
  - Specificity (true negative)=

$$Sensitivity = \frac{TP}{TP + FN}$$ True Positive rate

$$Specificity = \frac{TN}{FP + TN}$$ True Negative rate

- High sensitivity = Few false negatives
- High specificity = Few false positives

$\boxed{\text{Tradeoff}}$

e.g. airport alarm system

PRECISION AND RECALL

Precision & Recall

	Predicted class	
Actual class	Class=1	Class=0
Class=1	TP	FN
Class=0	FP	TN

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$ (= Sensitivity)

Precision

- Precision
  - $Precision = \frac{TP}{TP + FP}$
  - In a document retrieval example
    - the ratio between the documents that match the user expectation and the total number of documents returned by the system.
    - A system can *cheat* the precision score up to 100% by only returning documents about which it is extremely confident. By doing this, the amount of returned documents is severely reduced

    ➔ Consider Recall together with Precision

Recall

- The ratio between the documents that match the user expectation and the total number of expected documents from the user.
- How much good information that a system really misses

(* Precision: how much good information in the search result the user can use)

F1 score

- Harmonic mean of precision and recall

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

- Between 0 and 1

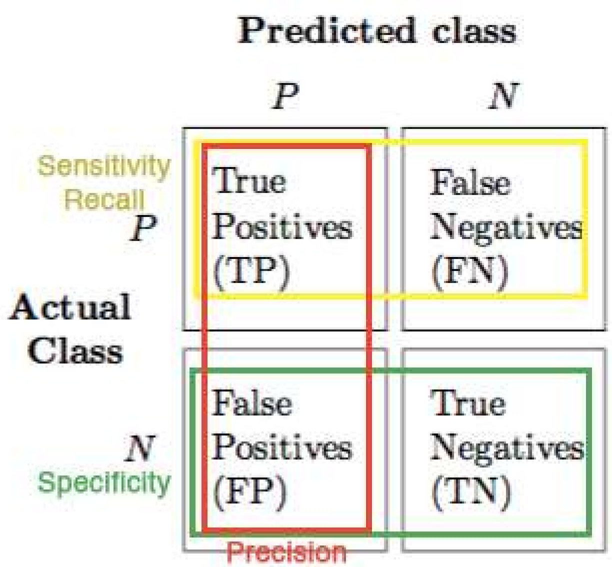
Classification Performances

- Confusion matrix

	Predicted class		
Actual class		Class=1	Class=0
	Class=1	4	1
	Class=0	2	3

  - Precision =
  - Recall =
  - F1 score =

Performance metrics



Which one to use

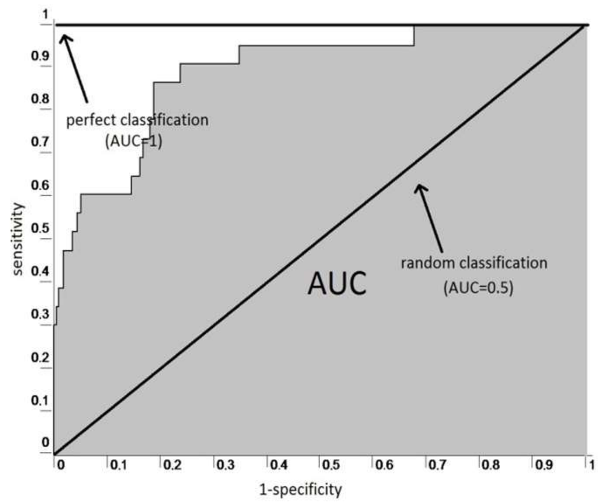
- Depends on the problem domain, e.g.
  - Search or information retrieval domain
    - precision & recall
  - Medical predictive system
    - Precision, recall, and specificity

ROC CURVE

- What if you change a threshold, or a hyper-parameter to your algorithm?

ROC (Receiver Operating Characteristic) curve

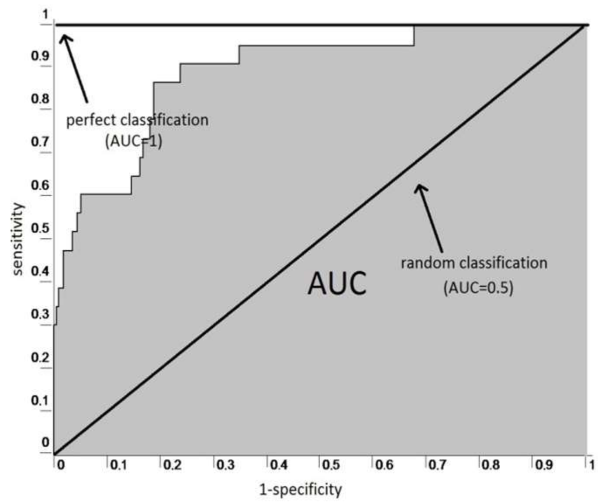
- ROC curve plots (1-specificity) (or FP rate) on the x-axis against sensitivity (or TP rate) on the y-axis



AUC: area under the curve

ROC (Receiver Operating Characteristic) curve

- ROC curve plots (1-specificity) (or FP rate) on the x-axis against sensitivity (or TP rate) on the y-axis



AUC: area under the curve

How to construct an ROC curve

+	Instance	P(+|A)	True Class
+	1	0.95	+
+	2	0.93	+
+	3	0.87	-
+	4	0.85	-
+	5	0.85	-
-	6	0.85	+
	7	0.76	-
	8	0.53	+
	9	0.43	-
	10	0.25	+

- Use classifier that produces posterior probability for each test instance P(+|A)
- Sort the instances according to P(+|A) in decreasing order
- Apply threshold at each unique value of P(+|A)
- Count TP,FP,TN,FN at each threshold
- Compute TP rate, FP rate

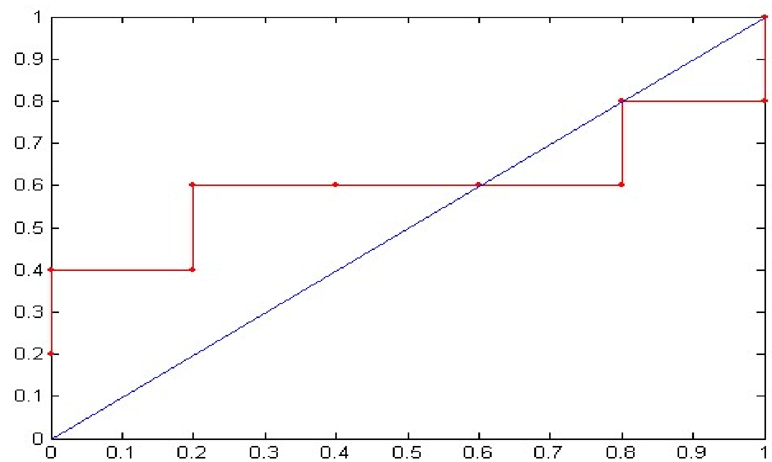
How to construct an ROC curve

	Instance	Sorted	True Class
+	1	28.9	1
+	2	23.4	0
+	3	23.2	1
+	4	21.7	1
+	5	21.6	1
+	6	21.5	0
–	7	19.9	1
	8	15.7	0
	9	10	0
	10	8.9	0

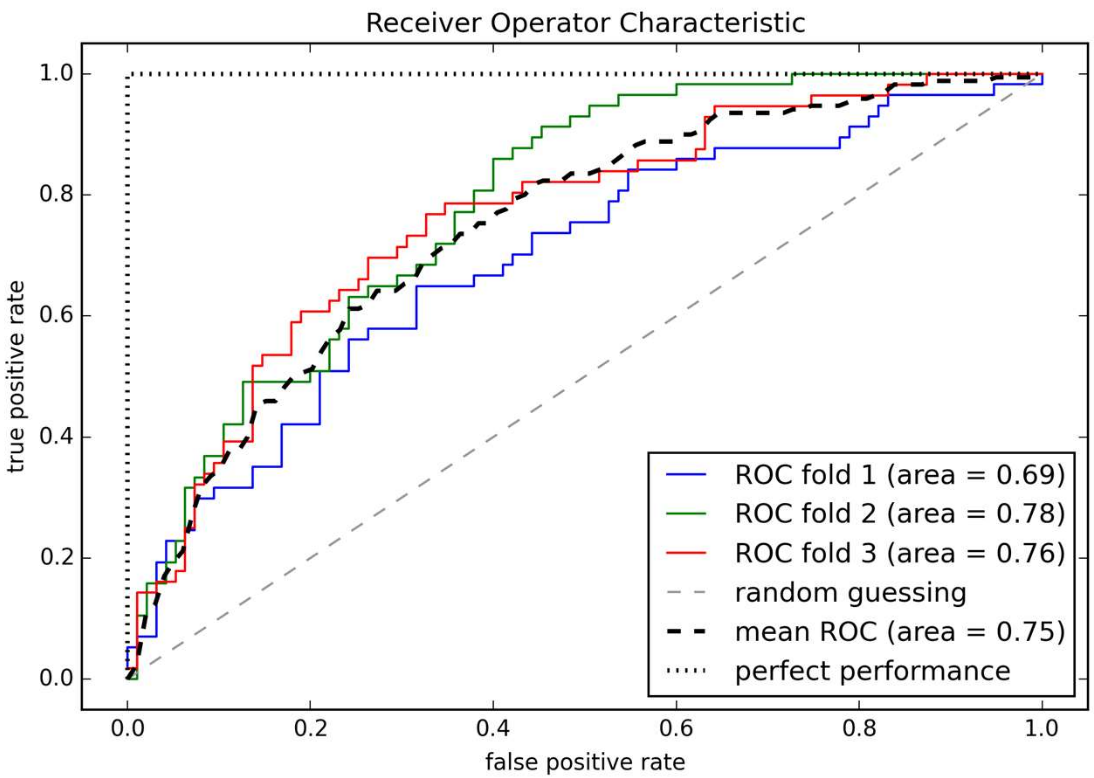
TP	FN	TN	FP	Sensitivity	Specificity
1	4	5	0	0.2	1
1	4	4	1	0.2	0.8

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



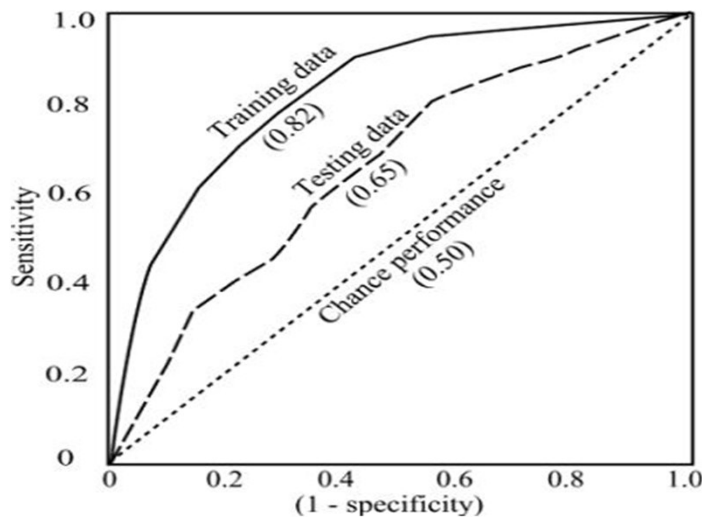
Example



Receiver Operator Characteristic
true positive rate
false positive rate
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (area = 0.69)
ROC fold 2 (area = 0.78)
ROC fold 3 (area = 0.76)
random guessing
mean ROC (area = 0.75)
perfect performance

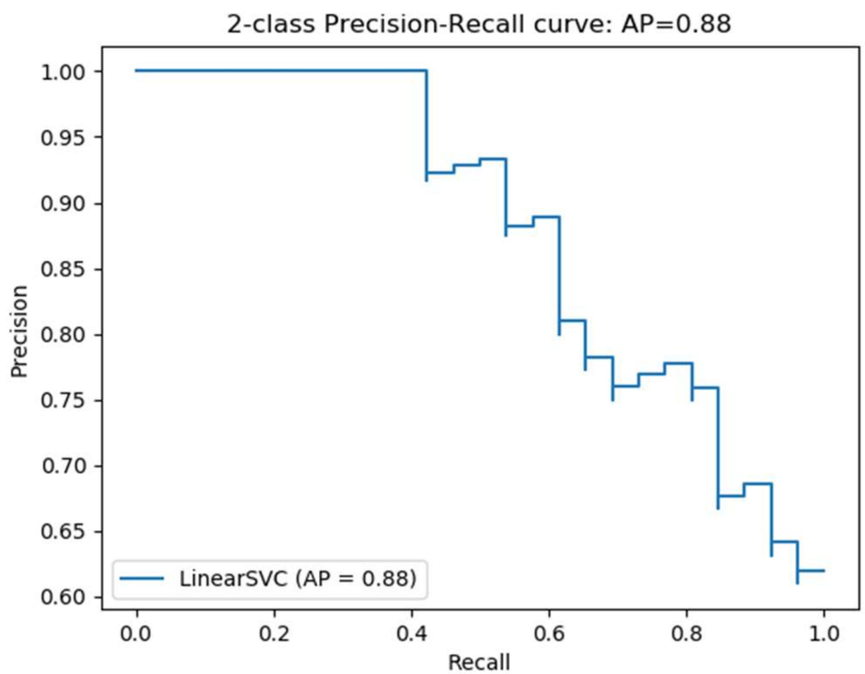
ROC curves

- Typically,



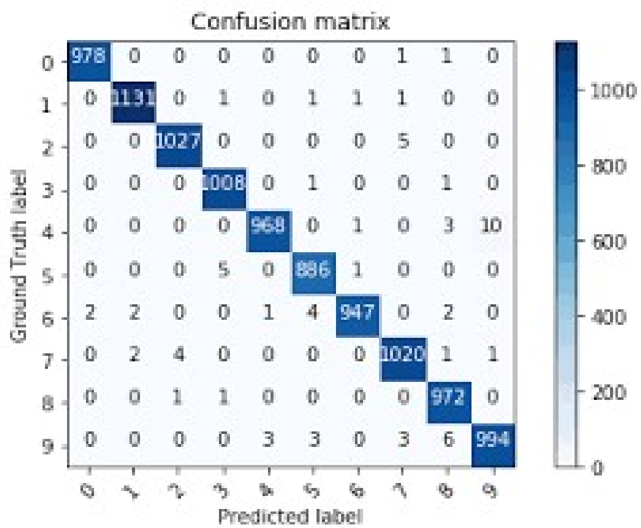
- AUC value between 0 and 1
  - Random guessing: 0.5
  - Perfect classification: 1

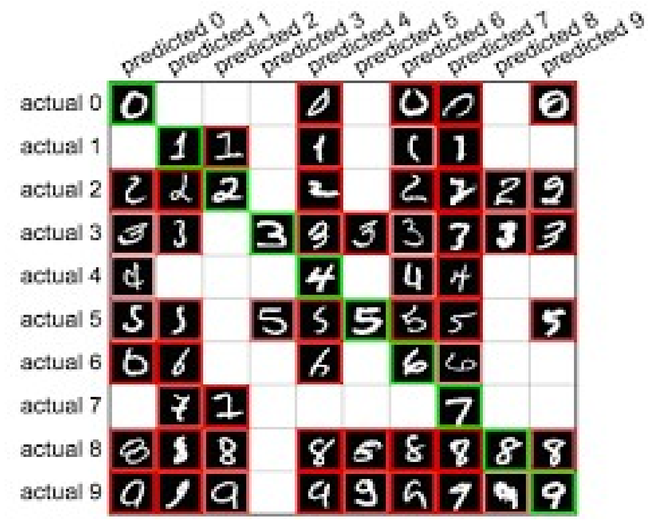
Precision-recall curve



MULTICLASS CLASSIFICATION

Multi-class Classification: MNIST

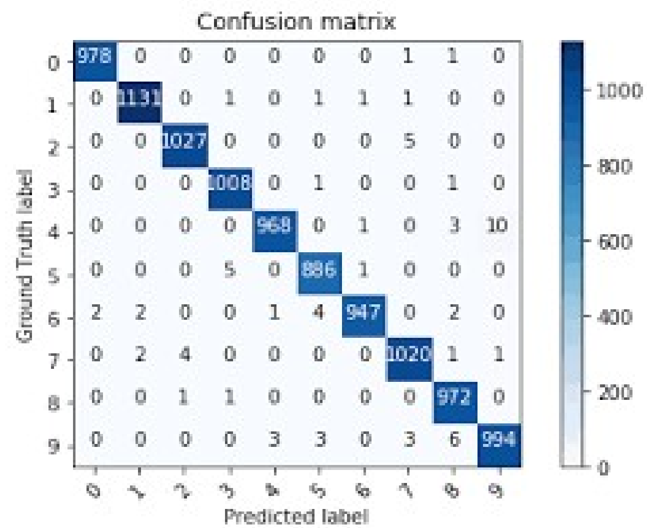




Scoring metrics

- Averaging methods to score multiclass problems via one-vs.all classification

- Micro-average
- Macro-average



Micro-averaging

- Calculated from the individual TPs, TNs, FPs, and FNs

Micro-precision = $(TP_1+\ldots TP_k)/(TP_1+\ldots TP_k+FP_1+\ldots FP_k)$

- Useful if we want to weight each sample equally

Macro-averaging

- Calculated as the average scores of the k different classes (or systems)

  Macro-precision = $(Pre_1 + \ldots Pre_k)/k$

- Weights all classes equally to evaluate the overall performance

Micro-F1: Example

Predicted

True

	Cat	Fish	Hen
Cat	4	1	1
Fish	6	2	2
Hen	3	0	6

- Looking at all classes together:

  4+2+6 = 12 correctly predicted sample (TP = 12)

  FP=6+3+1+0+1+2=13

  Micro-precision = 12/(12+13)= 48.0

  Micro-recall = ?

Macro-F1: Example

Predicted

True

	Cat	Fish	Hen
Cat	4	1	1
Fish	6	2	2
Hen	3	0	6

Class-wise result

Class	Precision	Recall	F1-score
Cat	30.8%	66.7%	42.1%
Fish	66.7%	20.0%	30.8%
Hen	66.7%	66.7%	66.7%

F1-score(Cat) = 2 × (30.8% × 66.7%) / (30.8% + 66.7%) = 42.1%

Macro-F1: Example

Class	Precision	Recall	F1-score
Cat	30.8%	66.7%	42.1%
Fish	66.7%	20.0%	30.8%
Hen	66.7%	66.7%	66.7%

Macro-F1 = (42.1% + 30.8% + 66.7%) / 3 = 46.5%

Macro-precision = (31% + 67% + 67%) / 3 = 54.7%
Macro-recall = (67% + 20% + 67%) / 3 = 51.1%

Weighted macro-average

- Calculated by weighting the score of each class by the number of true instances
- Useful when dealing with **class imbalances**

Class	Precision	Recall	F1-score
Cat	30.8%	66.7%	42.1%
Fish	66.7%	20.0%	30.8%
Hen	66.7%	66.7%	66.7%

Weighted-precision=(6 × 30.8% + 10 × 66.7% + 9 × 66.7%)/25 = 58.1%
Weighted-recall = (6 × 66.7% + 10 × 20.0% + 9 × 66.7%) / 25 = 48.0%

Weighted-F1 = (6 × 42.1% + 10 × 30.8% + 9 × 66.7%) / 25 = 46.4%

Class imbalance problem

- One or multiple classes are over-represented (have much more samples than others)
- Common problem when working with real-world data
  - Spam filtering, fraud detection, disease screening

Dealing with class imbalances

- Assign a larger penalty to wrong predictions on the minority class
- Upsampling the minority class, downsampling the majority class, or generation of synthetic training examples
- No universal best solution
  - Try different methods, evaluate, and choose the best one for your own application