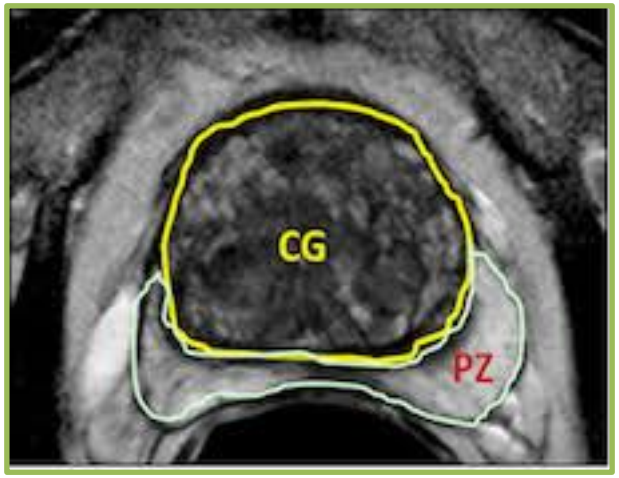
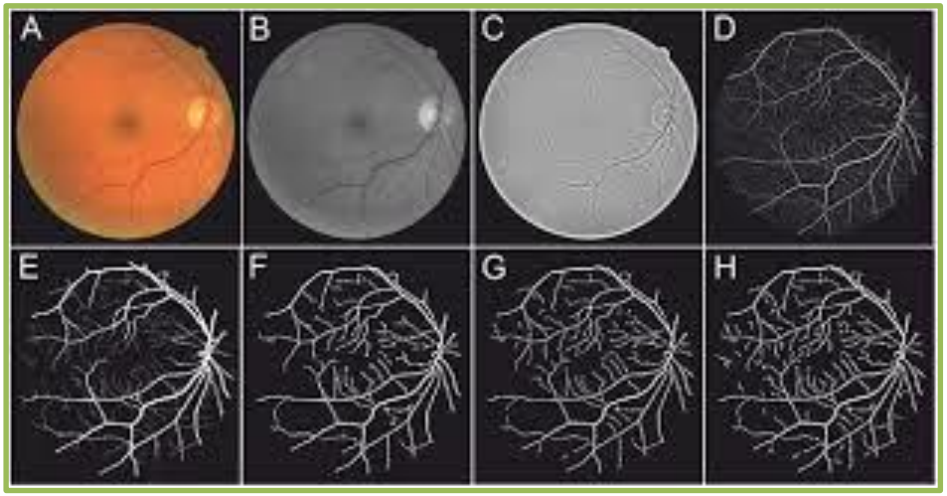
Classification/Segmentation/CAD

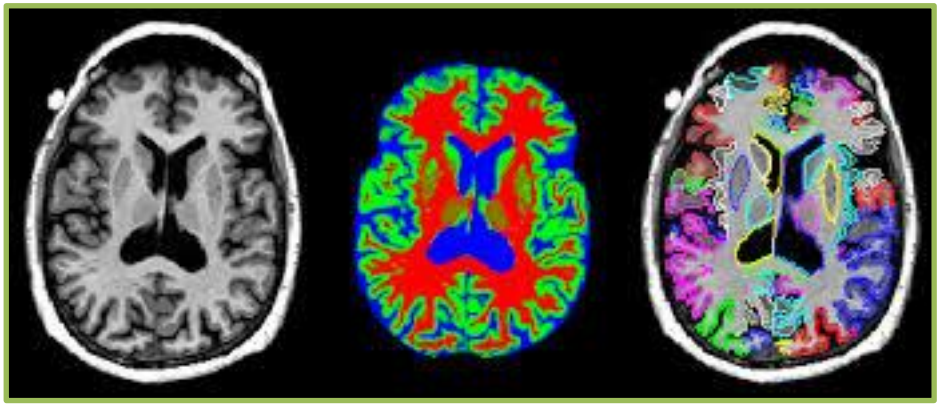
Background Information

Prof F Meriaudeau
Université de Bourgogne

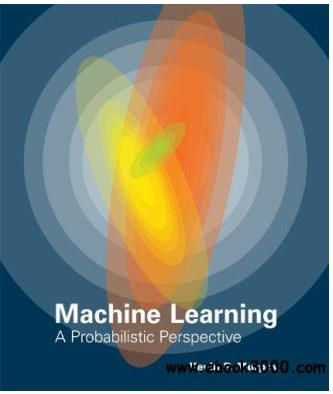

CG
PZ

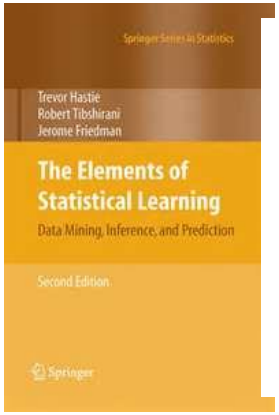

A
B
C
D
E
F
G
H

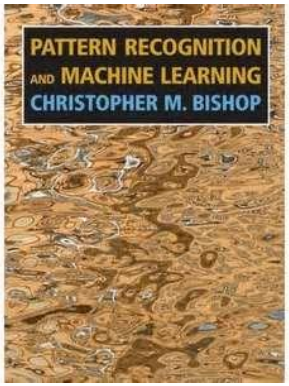

Normal lung

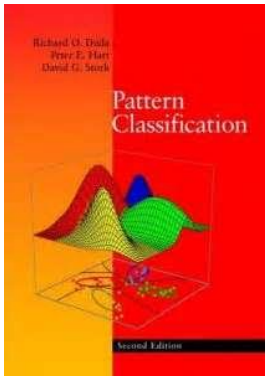


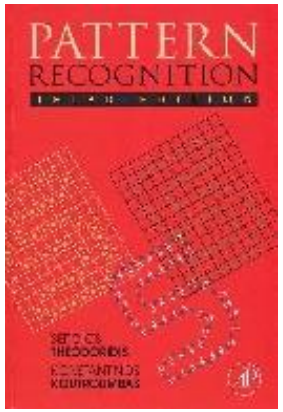
Material


Machine Learning
A Probabilistic Perspective


Springer Series in Statistics
Trevor Hastie
Robert Tibshirani
Jerome Friedman
The Elements of Statistical Learning
Data Mining, Inference, and Prediction
Second Edition
Springer

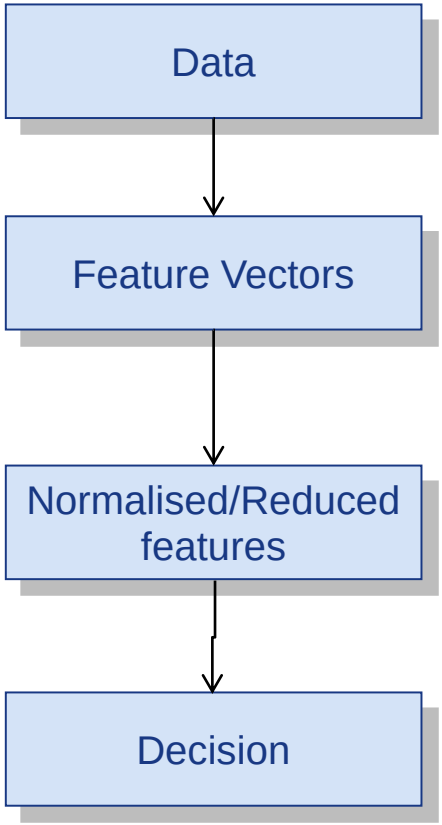

PATTERN RECOGNITION
AND MACHINE LEARNING
CHRISTOPHER M. BISHOP


Pattern Classification


PATTERN RECOGNITION

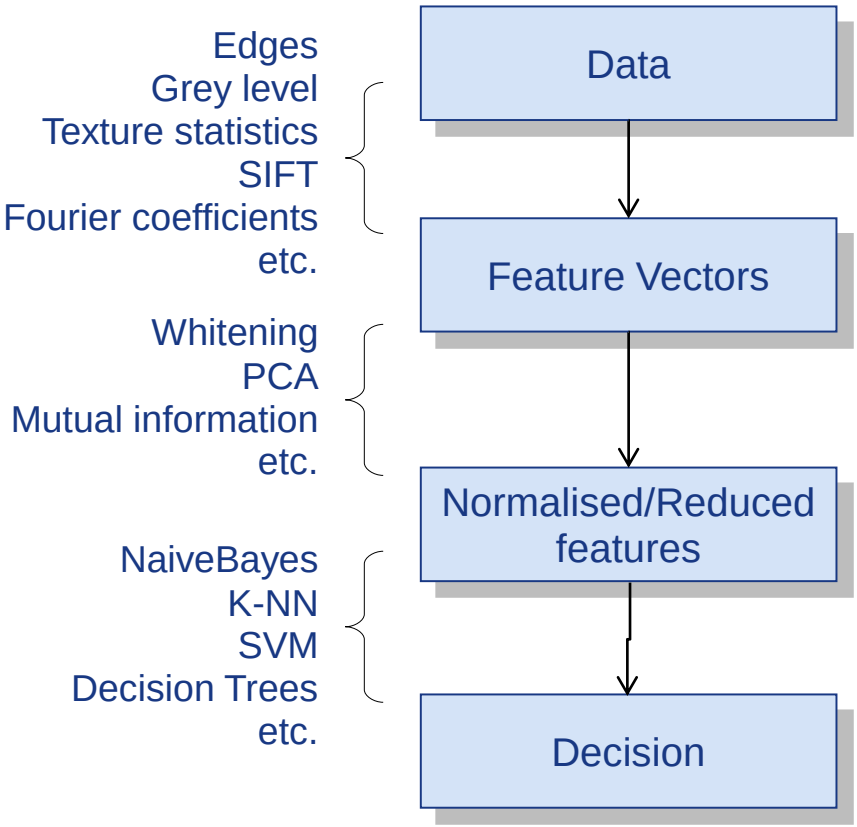
Introduction

- How do we make decisions?



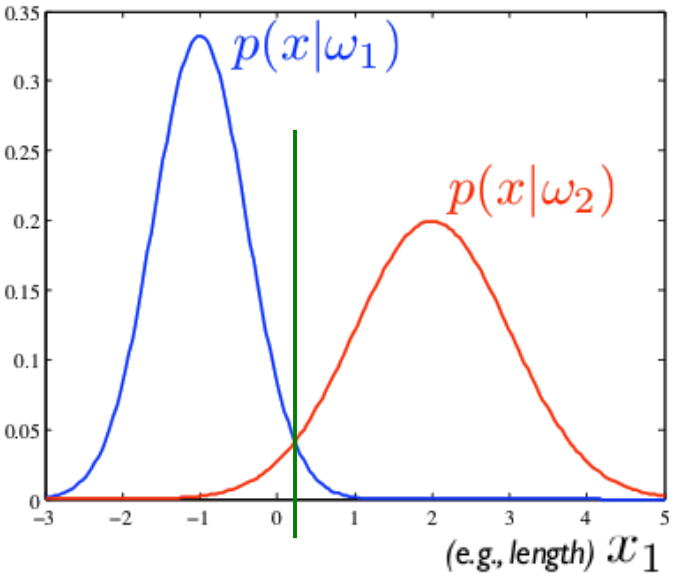
Introduction

- How do we make decisions?

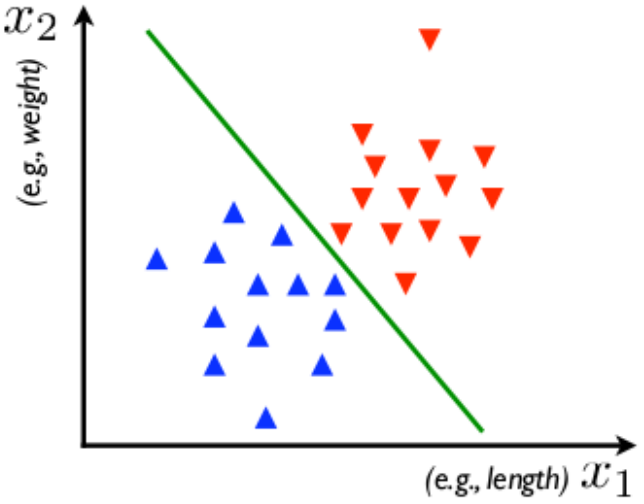


Introduction

- Decisions in a pattern recognition framework
  - We find the sample distribution in the space
  - Find rules or classifiers that define decision boundaries (**hard** or **soft**).



1-dimensional samples



2-dimensional samples

Introduction

- Pattern recognition approach
  - We measure properties from our sample population
  - These properties are called features
  - Sample with more than one feature → **feature vector**

Classification Phase

- Feature vectors are used a input for the classifier
- Classification results in a discrete class index
  - Cancer/Non Cancer (image level)
  - Cancer/Non Cancer (pixel level)
- Different types of classifiers:
  - Statitiscal
  - Parametric
  - Non Parametric
  - Linear
  - Non Linear
  - ….

Supervised/Unsupervised

- Supervised classification
  - Samples of our data with known labels.
  - Construct or train classifier to differentiate between these training samples.
- Unsupervised classification
  - it is the role of the classifier to find the natural groupings or **clusters** in the input patterns
- Semi-supervised
  - A mixture of the two

Unsupervised learning

- This week we will see only training samples that are “labelled” by their category membership → **supervised**.
- However:
  - Why would one even be interested in learning with unlabelled samples?
  - Is it even possible in principle to learn anything of value from unlabelled samples?
- Reasons
  - Collecting and labelling a large set of sample patterns can be surprisingly costly
  - Train a classifier on a small set of samples, then tune it up to make it run without supervision on a large, unlabelled set
  - Or, in the reverse direction, let a large set of unlabelled data group automatically, then label the groupings found (bag of words...)
  - To detect the gradual change of pattern over time

Clustering

- One of the key components of unsupervised learning is Clustering

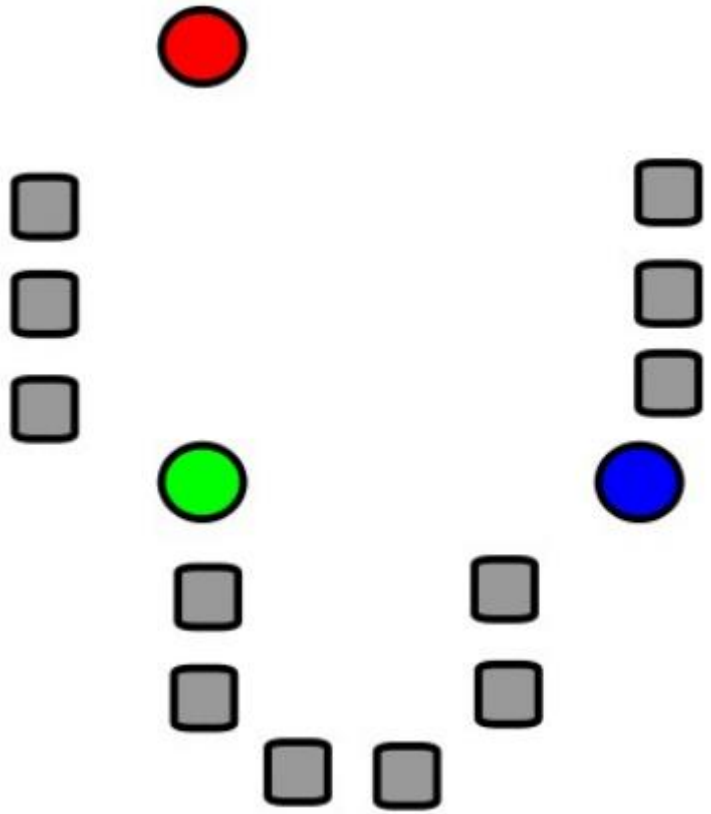
Clustering is a mathematical tool that attempts to discover structures or certain patterns in a data set, where the objects inside each cluster show a certain degree of similarity

- Hard clustering
  - Assign each sample to one class only
- Fuzzy clustering
  - Each sample has a degree of membership to all classes

K-Means

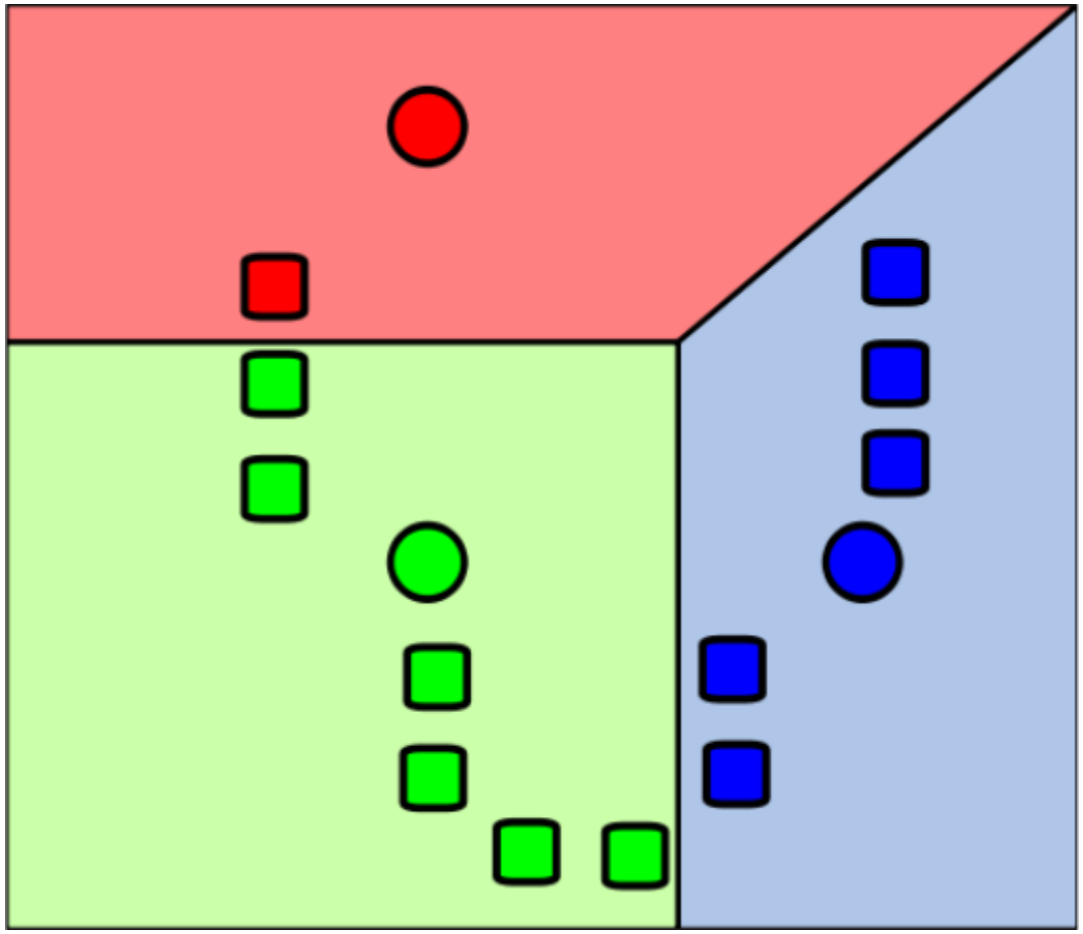
- The most known example of hard clustering
- Composed of three steps
  - Initialisation
    - The number of cluster and cluster centres are identified
  - Assignment step
    - Assign each observation to the cluster with the closest mean
  - Update step
    - Calculate the new means to be the centroid of the observations in the cluster
  - Repeat Assignment/Update steps until convergence is found

K-Means



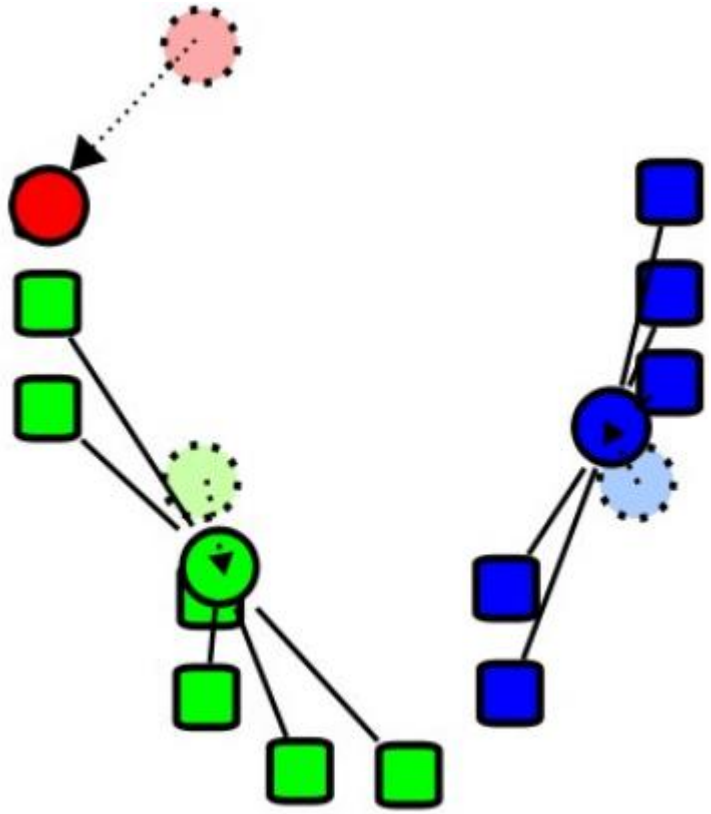
Initialisation

K-Means



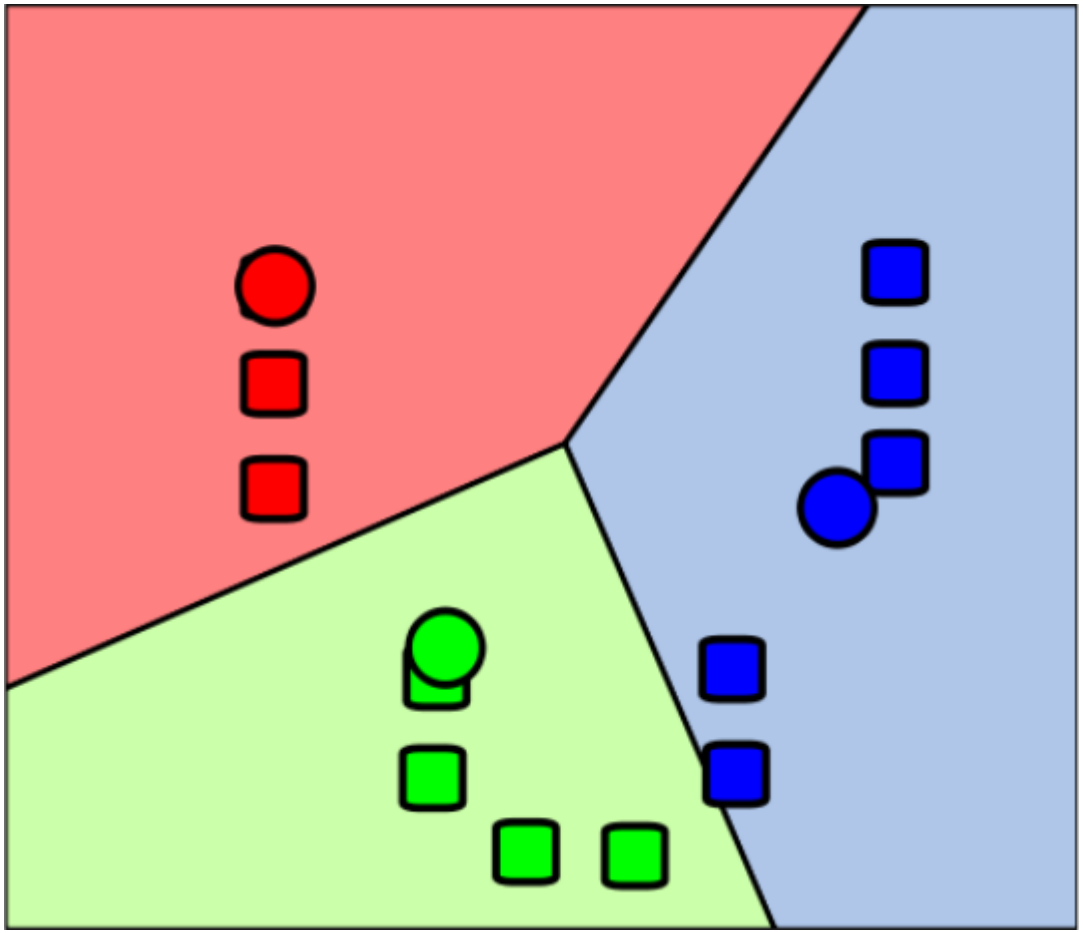
Assignment

K-Means



Update

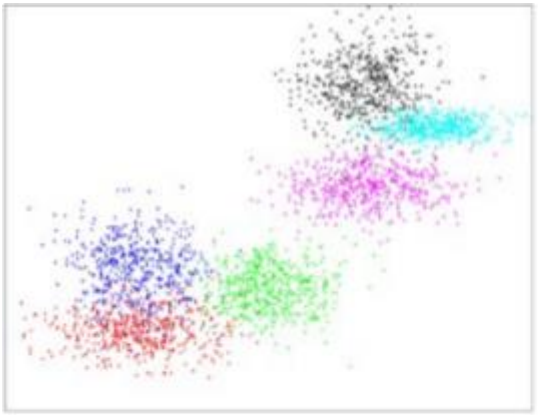
K-Means



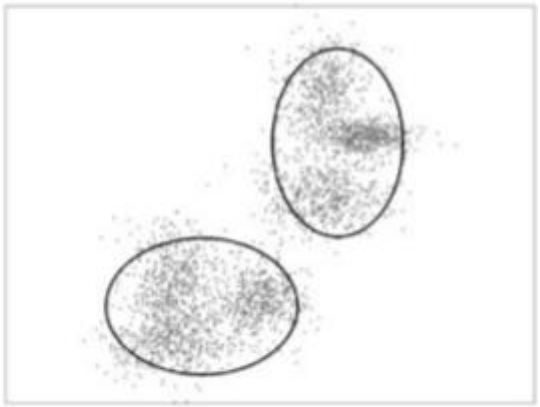
Check convergence

K-Means

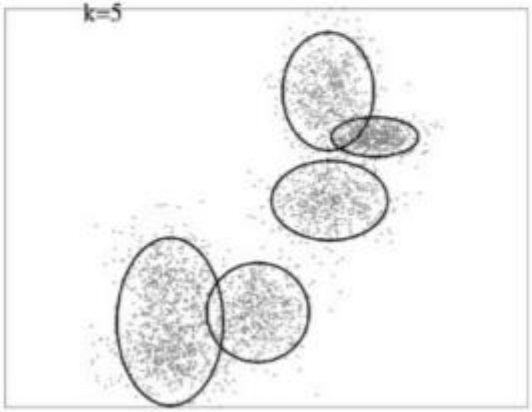
- Importance of the selection of K



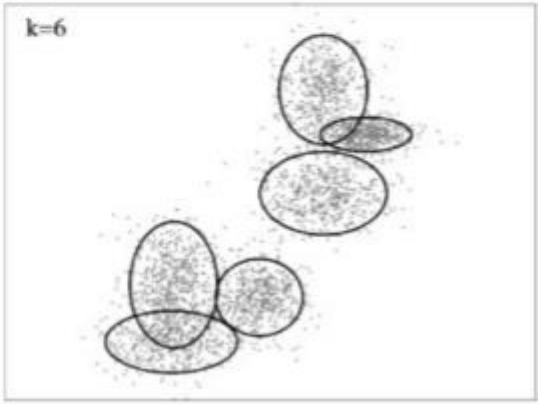
ground truth



$k = 2$



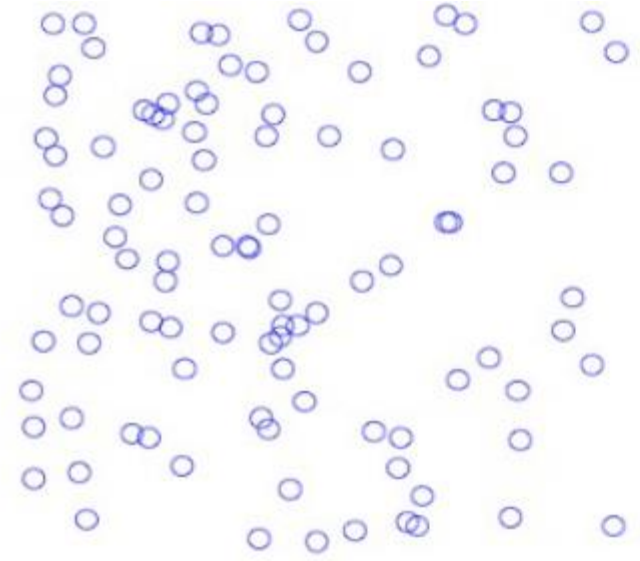
$k = 5$



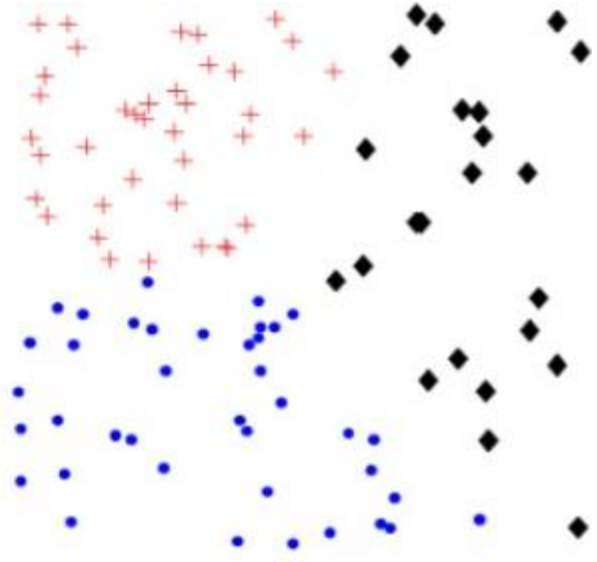
$k = 6$

K-Means

- Clustering algorithms find clusters, even if there are no natural clusters in the data



100 2D uniform data points



k-Means with k=3

Fuzzy c-means

- A widely known example of fuzzy clustering
- Composed of the same three steps
  - Initialisation
  - Assignment step
  - Update step
  - Repeat Assignment/Update steps until convergence is found
- Differences
  - Each sample *x* has a degree of membership to the class *k* represented as $u_k(x)$
  - The sum of $\forall x \left(\sum_{k=1}^{\text{num. clusters}} u_k(x) = 1 \right)$ ship must be 1

Fuzzy c-means

- The centroid of a cluster is the mean of all points, weighted by their degree of belonging to the cluster

$$\text{center}_k = \frac{\sum_x u_k(x)^m x}{\sum_x u_k(x)^m}.$$

- The degree of belonging is related to the inverse of the distance to the cluster centre

$$u_k(x) = \frac{1}{d(\text{center}_k, x)},$$

- the degree of belonging can be weighted with a parameter *m*

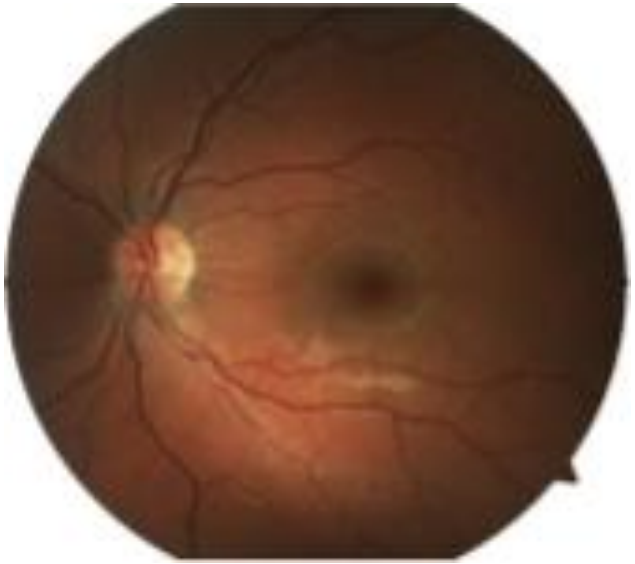
$$u_k(x) = \frac{1}{\sum_j \left(\frac{d(\text{center}_k, x)}{d(\text{center}_j, x)} \right)^{2/(m-1)}}$$

Clustering for segmentation

- Clustering is often used for segmentation
  - As an initial step
  - To provide an estimate of the structures available
- Features used are various, for example:
  - Pixel values
    - RGB
    - Grey
    - HSV
    - Other colour spaces (L*a*b* !!!)
  - Coordinates
    - From origin or from other interest point
  - Output of a filter
    - Low/High pass
    - Texture descriptors
    - Morphological operators
    - Combination of filters/scales

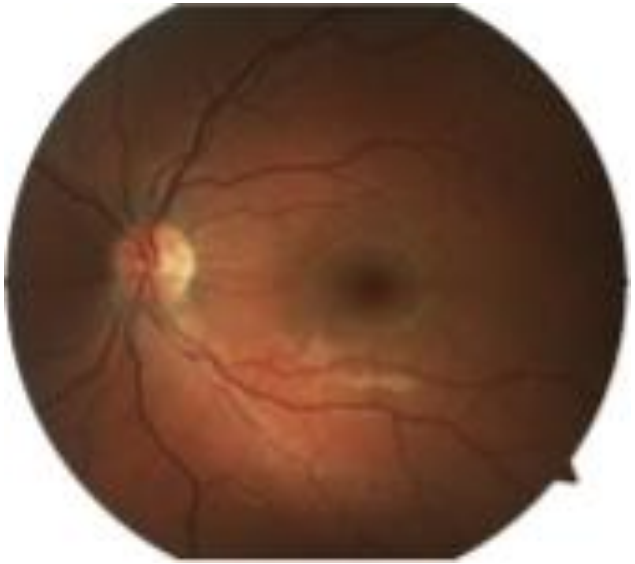
Clustering for segmentation

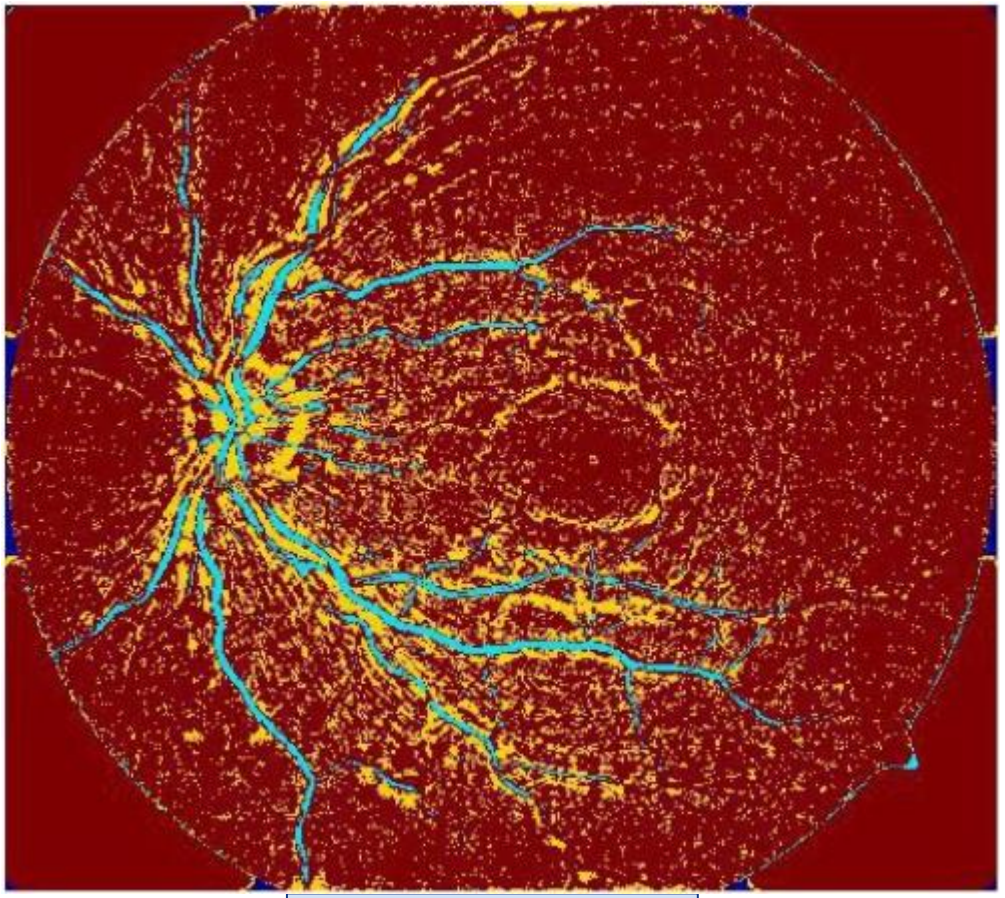
Original Image



Clustering for segmentation

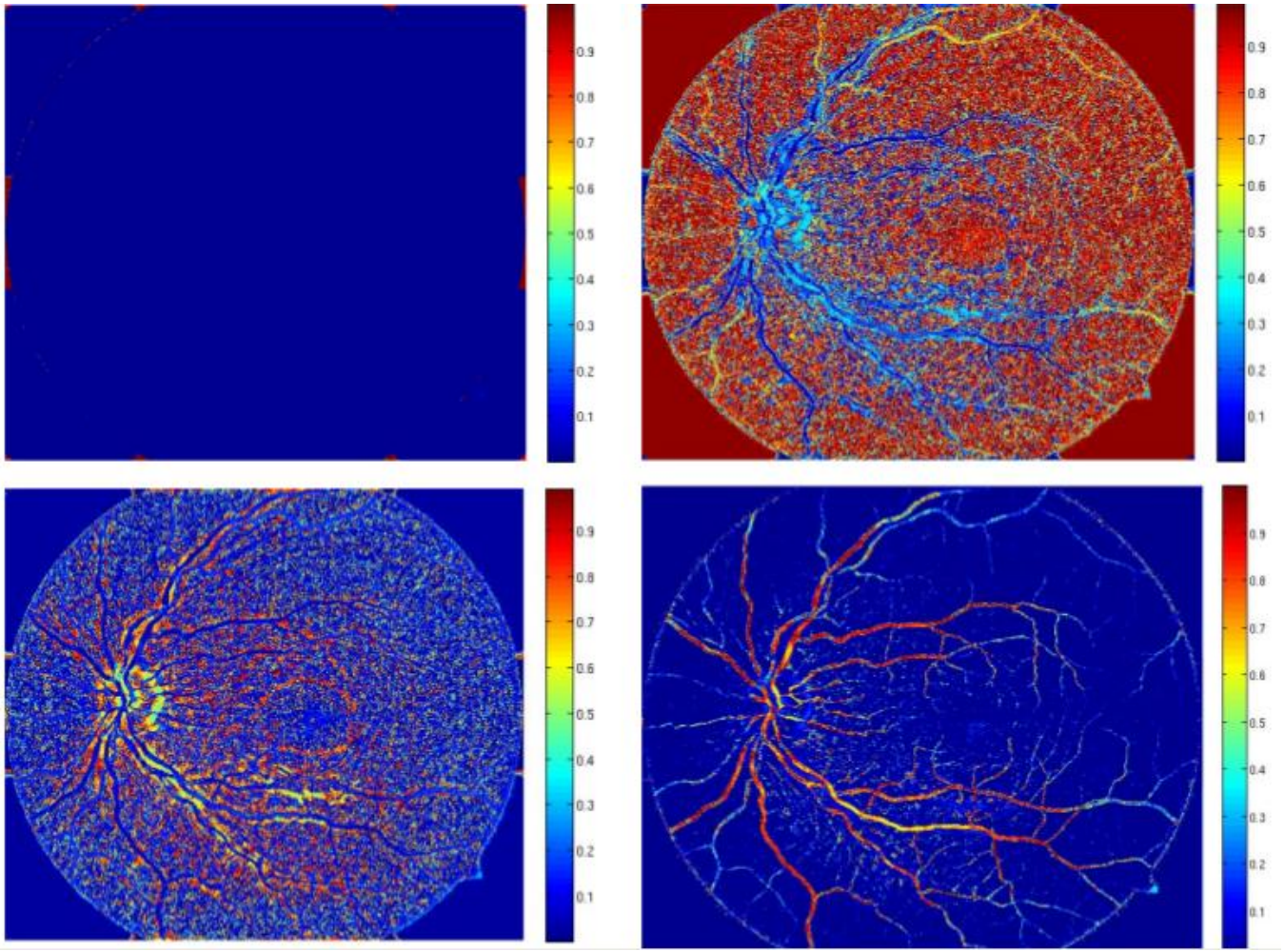
- Feature vector *x* composed of: R,G,B, k=4





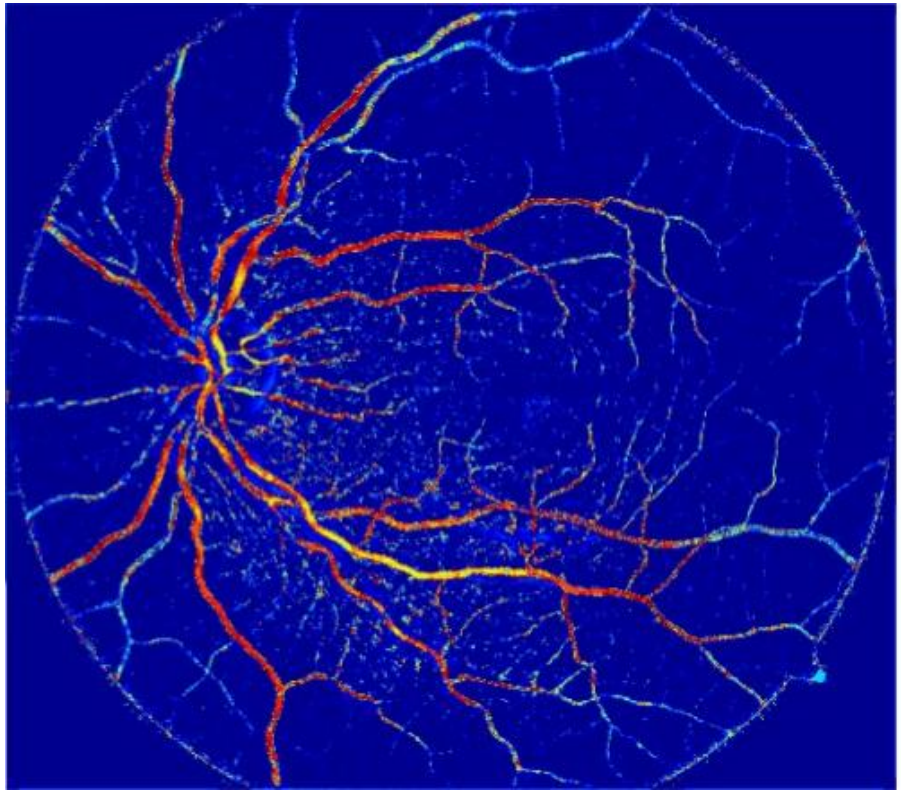
K-Means

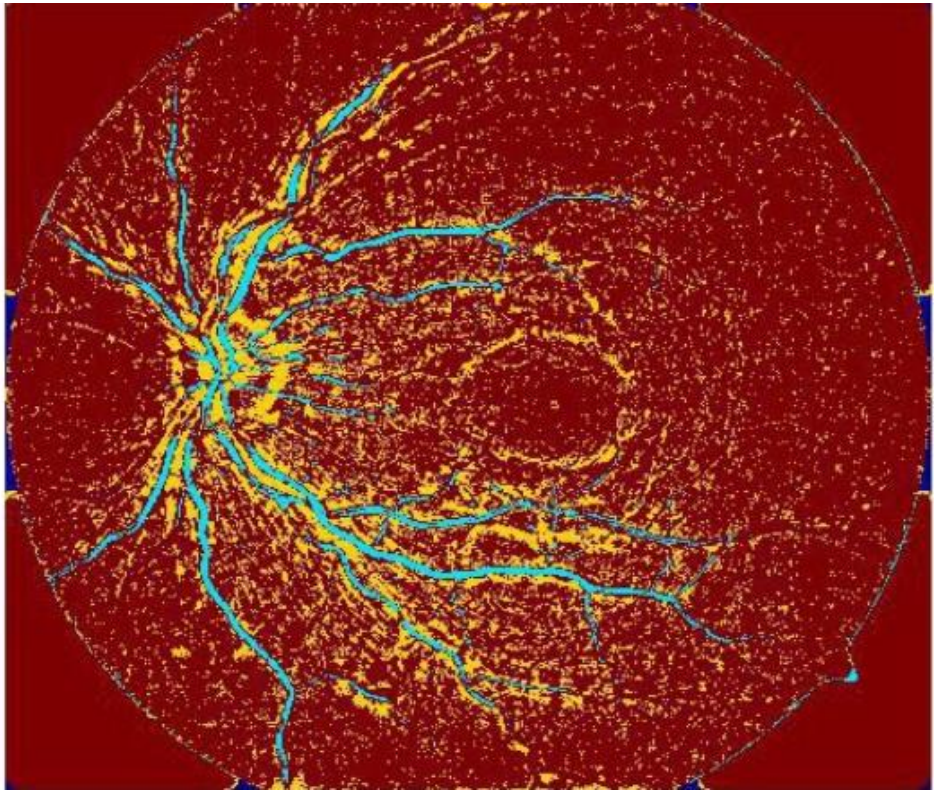
Clustering for segmentation



Fuzzy C-Means

Clustering for segmentation



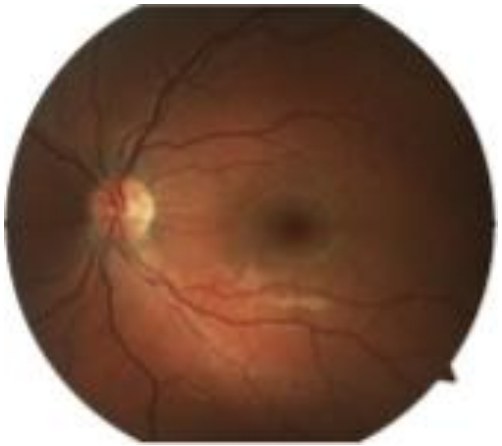


Comparison for vessel segmentation

Supervised/Unsupervised

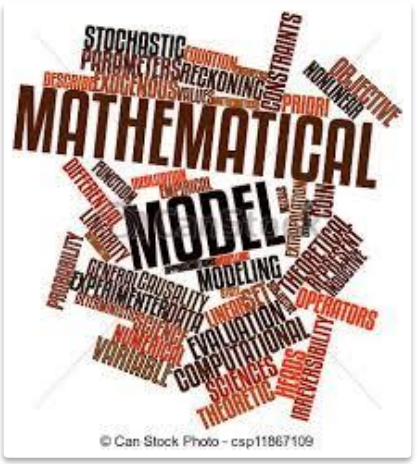
- Supervised classification
  - Samples of our data with known labels.
  - Construct or train classifier to differentiate between these training samples.

- Unsupervised classification
  - it is the role of the classifier to find the natural groupings or **clusters** in the input patterns
- Semi-supervised
  - A mixture of the two



features

Labels



Disease

Non Disease

Classifier Evaluation ?

Classifier Evaluation

- Each problem different way to be evaluated
- Training / test set
  - Standard division
  - 2-fold
  - N-fold
  - Leave-one-out / hold-one-out (LOO/HOO)
  - Bootstrapping (or Bagging)
- **NEVER use test set for training**
- Importance of sampling
  - Your training set must be a good representation of the problem (i.e. cover as much feature space as possible)

Classifier Evaluation

Almost invariably, all the pattern recognition techniques that you will see , have one or more free parameters

–The number of neighbors in a kNN classifier

–The bandwidth of the kernel function in kernel density estimation

–The number of features to preserve in a subset selection problem

•Two issues arise at this point

–**Model Selection: How do we select the "optimal" parameter(s) for a given classification problem?**

–**Validation: Once we have chosen a model, how do we estimate its true error rate?**

•The true error rate is the classifier's error rate when tested on the ENTIRE POPULATION

•If we had access to an unlimited number of examples, these questions would have a straightforward answer

–Choose the model that provides the lowest error rate on the entire population

–And, of course, that error rate is the true error rate

•However, in real applications only a finite set of examples is available

–This number is usually smaller than we would hope for!

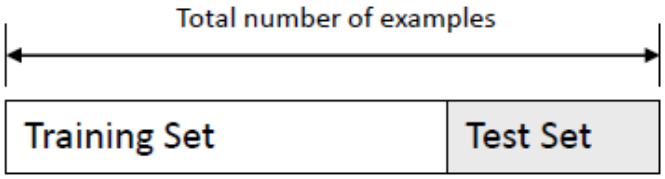
–Why? Data collection is a very expensive process

Classifier Evaluation

The Hold Out 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

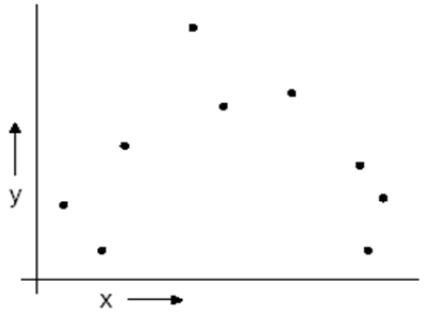


Classifier Evaluation

The Hold Out Method

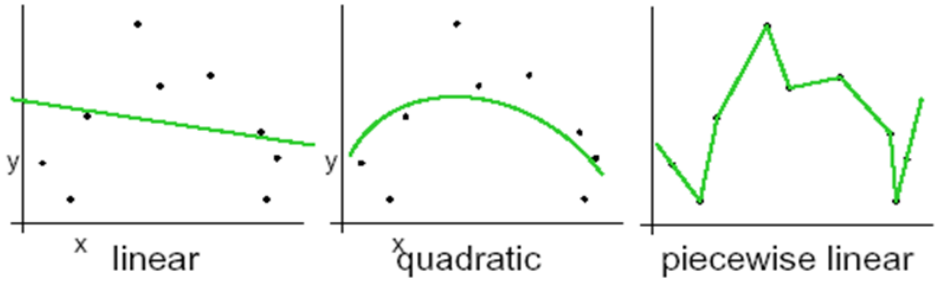
Want to learn model from data

Toy example



Can we learn f(x) that fits our data, y=f(x)+noise?

Which is best?

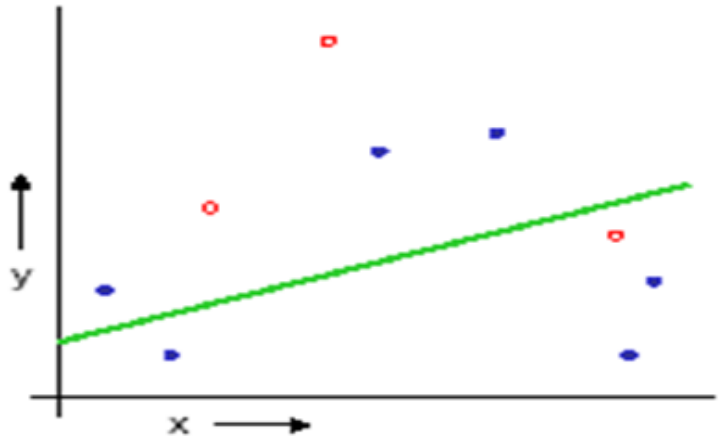


Why not choose the method with the best fit to the data?

Classifier Evaluation

The Hold Out Method

The test set method



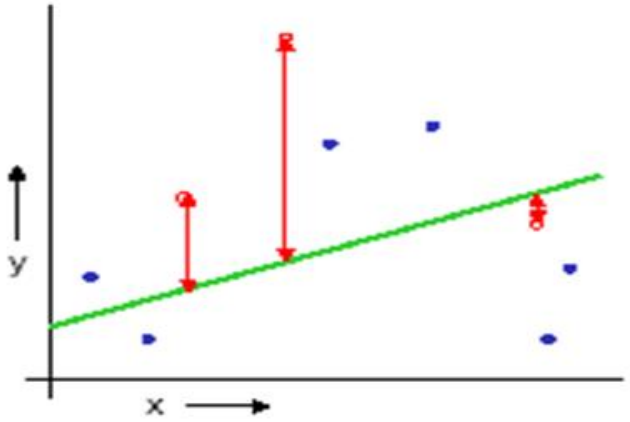
(Linear function example)

1. Randomly choose 30% of the data to be in a test set
2. The remainder is a training set
3. Learn model from the training set

Classifier Evaluation

The Hold Out Method

The test set method



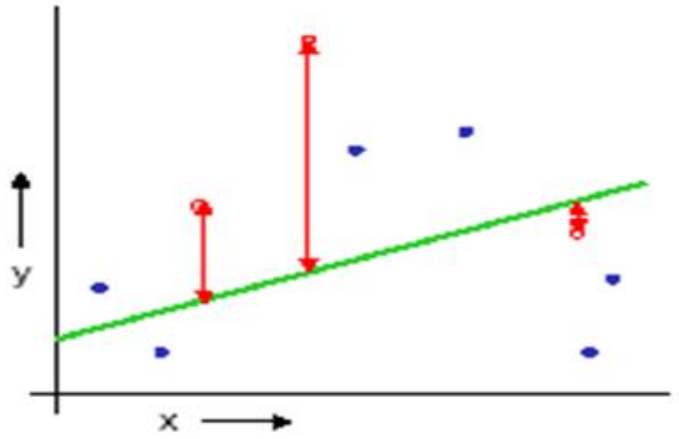
1. Randomly choose 30% of the data to be in a test set
2. The remainder is a training set
3. Learn the model from the training set
4. Estimate your future performance with the test set

How to tell how good is the prediction?

Classifier Evaluation

The Hold Out Method

The test set method



(Linear function example)

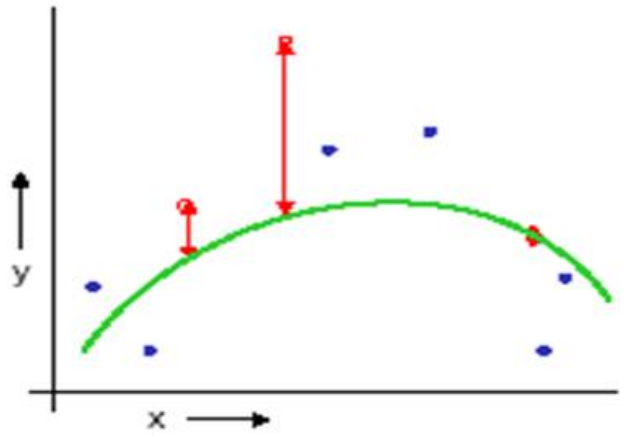
Mean Squared Error = 2.4

1. Randomly choose 30% of the data to be in a test set
2. The remainder is a training set
3. Learn the model from the training set
4. Estimate your future performance with the test set

Classifier Evaluation

The Hold Out Method

The test set method



(Quadratic function)

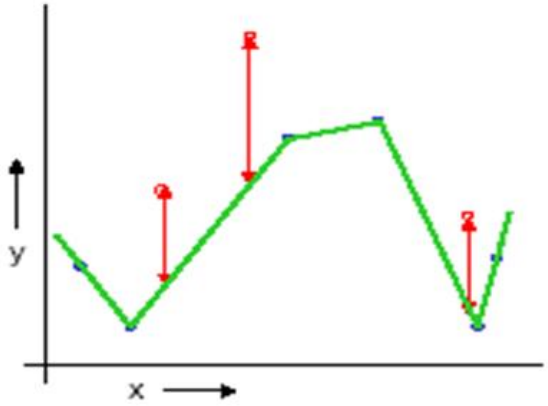
Mean Squared Error = 0.9

1. Randomly choose 30% of the data to be in a test set
2. The remainder is a training set
3. Perform model fitting on the training set
4. Estimate your future performance with the test set

Classifier Evaluation

The Hold Out Method

The test set method



(Piecewise Linear function)

Mean Squared Error = 2.2

1. Randomly choose 30% of the data to be in a test set
2. The remainder is a training set
3. Perform model fitting on the training set
4. Estimate your future performance with the test set

Classifier Evaluation

The Hold Out Method

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

These limitations of the holdout can be overcome with a family of resampling methods at the expense of higher computational cost

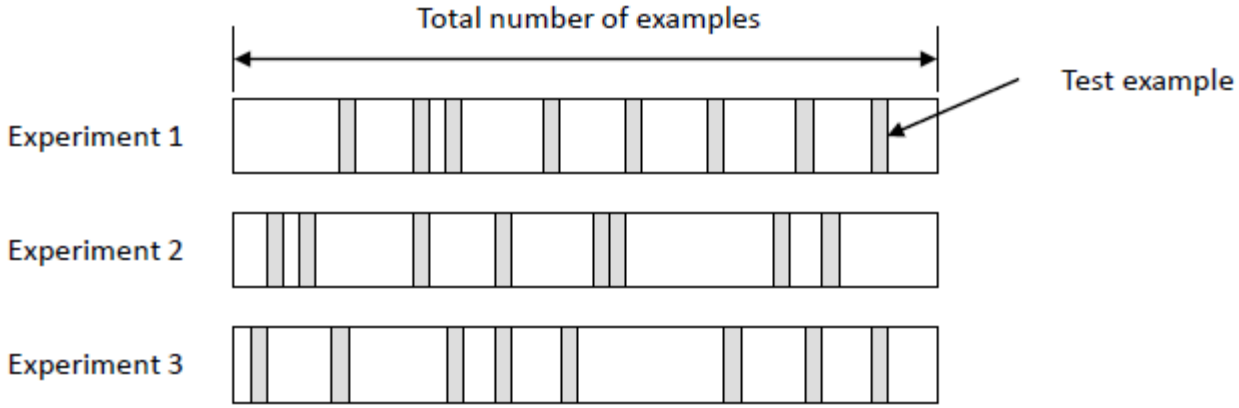
- Cross validation
  - Random subsampling
  - K-fold cross-validation
  - Leave-one-out cross-validation
- Bootstrap

Classifier Evaluation

Random Subsampling

Random subsampling performs K data splits of the entire dataset

- Each data split randomly selects a (fixed) number of examples without replacement
- For each data split we retrain the classifier from scratch with the training examples and then estimate E_i with the test examples



Classifier Evaluation

Random Subsampling

- The true error estimate is obtained as the average of the separate estimates E_i
- This estimate is significantly better than the holdout estimate

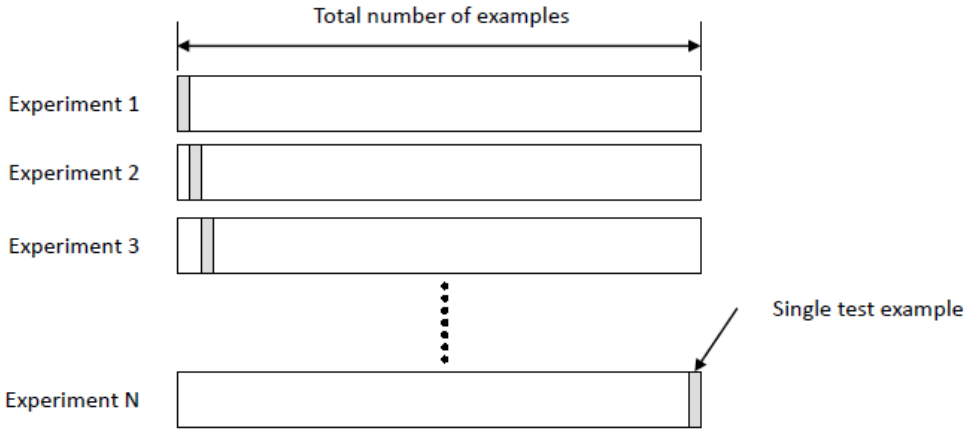
$$E = \frac{1}{K}\sum_{i=1}^{K} E_i$$

Classifier Evaluation

Leave One Out Cross Validation

LOO is the degenerate case of KFCV, 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



Classifier Evaluation

Leave One Out Cross Validation

- As usual, the true error is estimated as the average error rate on test examples

$$E = \frac{1}{N}\sum_{i=1}^{N} E_i$$

Classifier Evaluation

Cross-Validation – How Many Folds?

With a large number of folds

+ The bias of the true error rate estimator will be small (the estimator will be very accurate)
– The variance of the true error rate estimator will be large
– The computational time will be very large as well (many experiments)

With a small number of folds

+ The number of experiments and, therefore, computation time are reduced
+ The variance of the estimator will be small
– The bias of the estimator will be large (conservative or larger than the true error rate)

In practice, the choice for K depends on the size of the dataset

– For large datasets, even 3-fold cross validation will be quite accurate
– For very sparse datasets, we may have to use leave-one-out in order to train on as many examples as possible

A common choice for is K=10

Classifier Evaluation

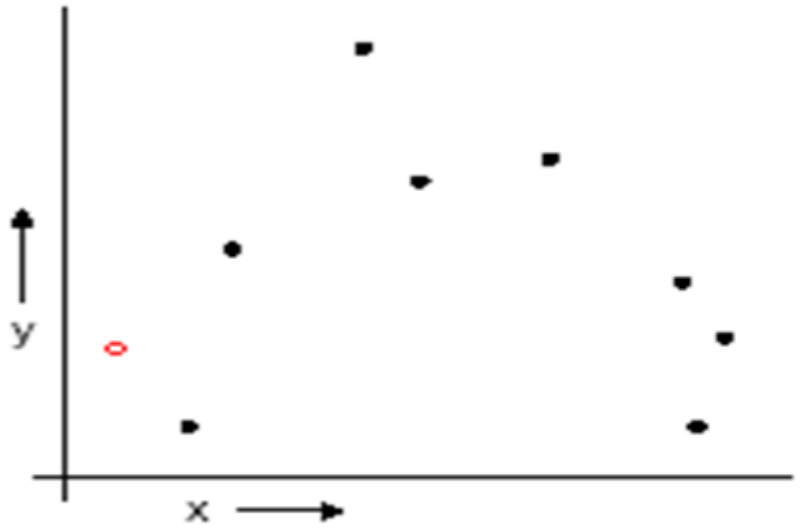
Leave One Out Cross Validation

Example……

LOOCV (Leave-one-out Cross Validation)

For k=1 to R, R = # of records

1. Let (x_k, y_k) be the k^{th} record

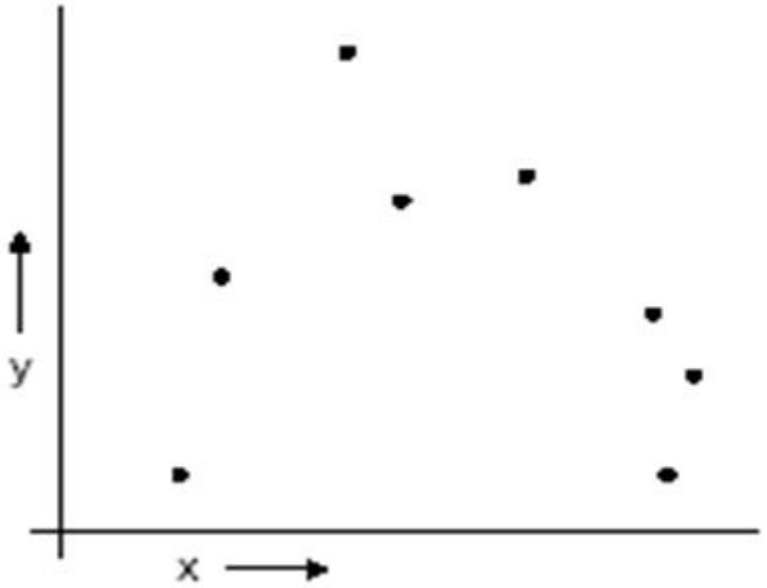


Classifier Evaluation

Leave One Out Cross Validation

Example……

LOOCV (Leave-one-out Cross Validation)



For k=1 to R

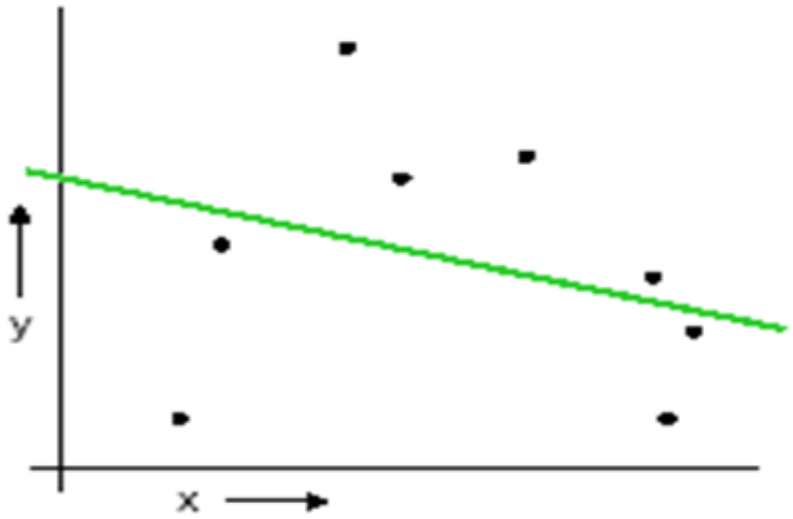
1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset

Classifier Evaluation

Leave One Out Cross Validation

Example……

LOOCV (Leave-one-out Cross Validation)



For k=1 to R

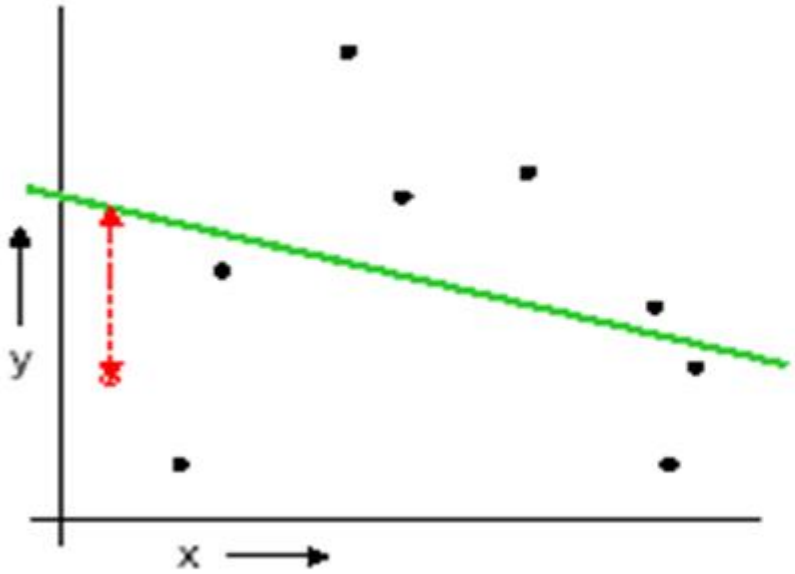
1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining R-1 datapoints

Classifier Evaluation

Leave One Out Cross Validation

Example……

LOOCV (Leave-one-out Cross Validation)



For k=1 to R

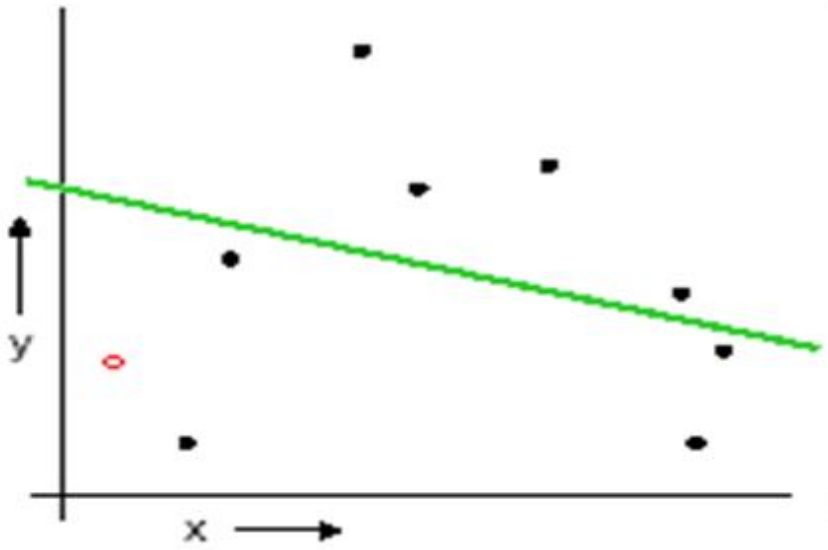
1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining R-1 datapoints
4. Note your error (x_k, y_k)

Classifier Evaluation

Leave One Out Cross Validation

Example……

LOOCV (Leave-one-out Cross Validation)



For k=1 to R

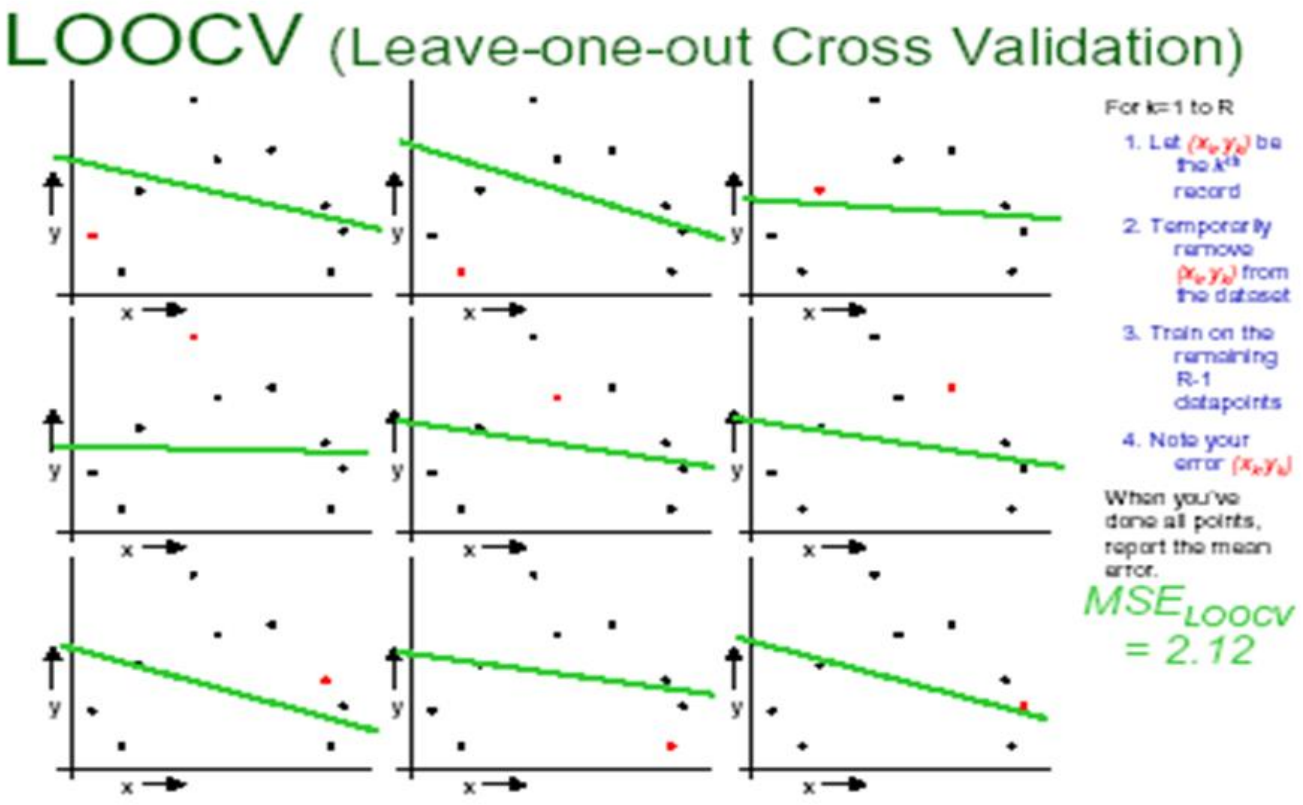
1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining R-1 datapoints
4. Note your error (x_k, y_k)

When you've done all points, report the mean error.

Classifier Evaluation

Leave One Out Cross Validation

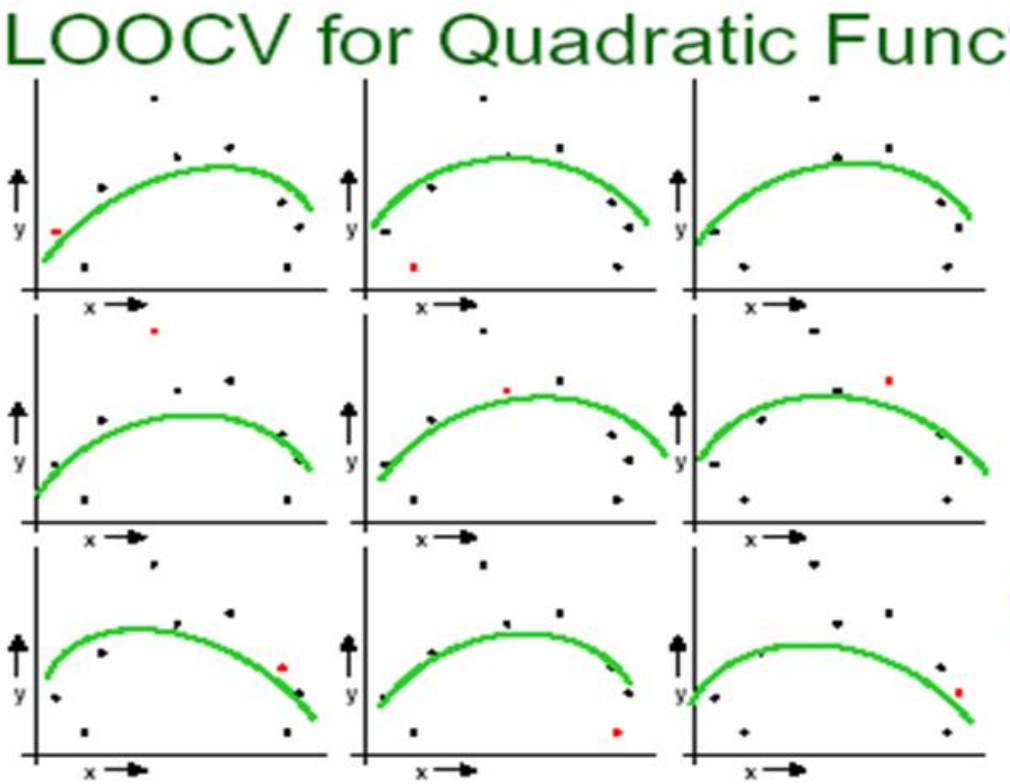
Example……



Classifier Evaluation

Leave One Out Cross Validation

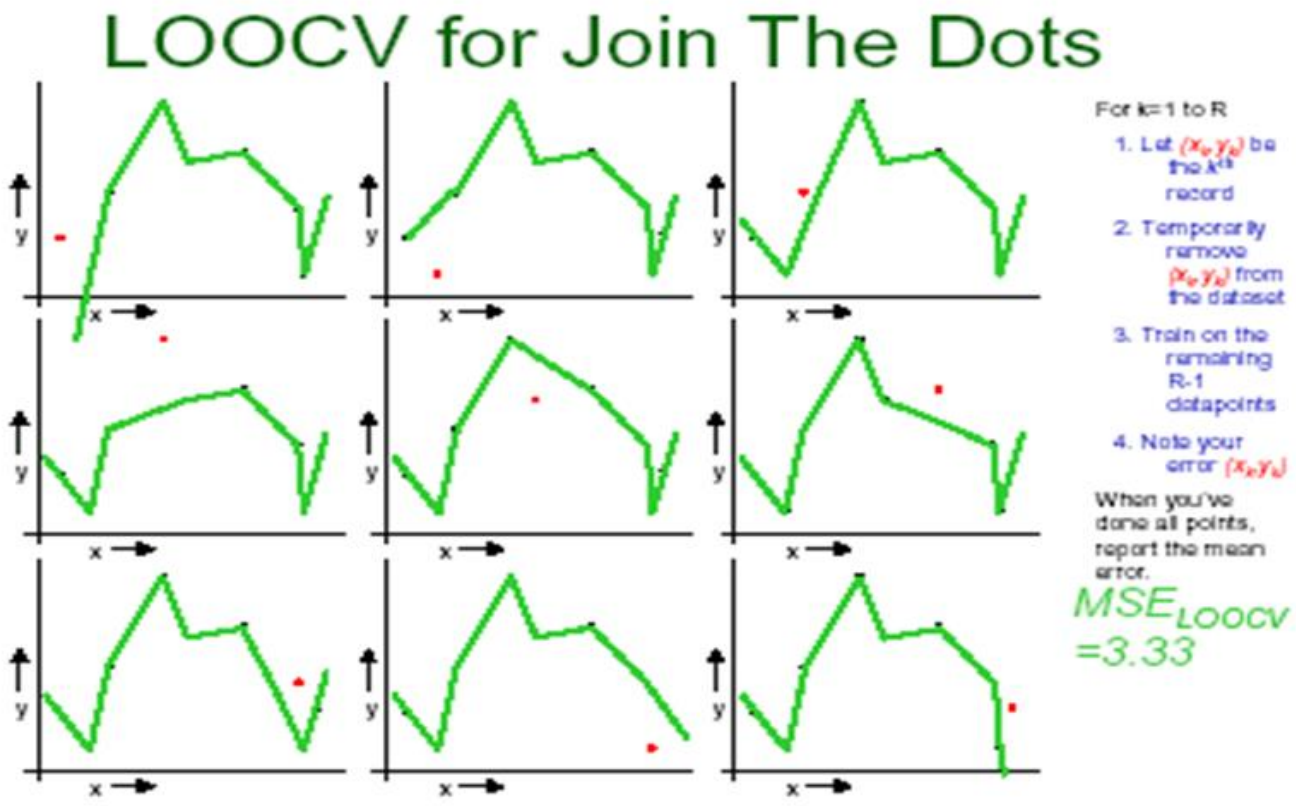
Example……



Classifier Evaluation

Leave One Out Cross Validation

Example……

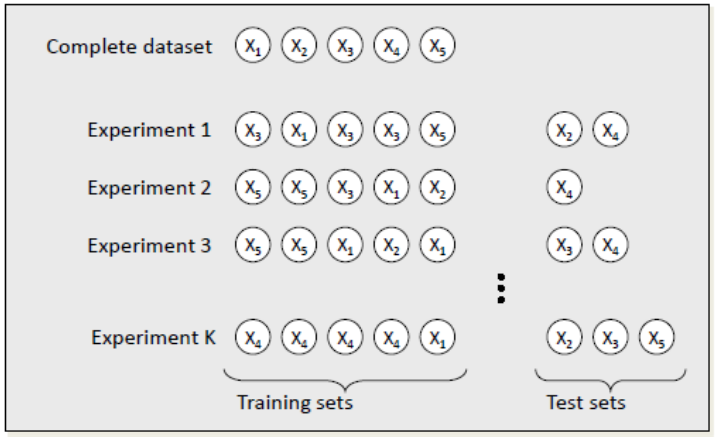


Classifier Evaluation

The Bootstrap

The bootstrap is a resampling technique with replacement

- From a dataset with N examples
  - Randomly select (with replacement) N examples and use this set for training
  - The remaining examples that were not selected for training are used for testing
  - This value is likely to change from fold to fold
- Repeat this process for a specified number of folds (K)
- As before, the true error is estimated as the average error rate on test data



Classifier Evaluation

The Three Ways Data Split

If model selection and true error estimates are to be computed simultaneously, the data should be divided into three disjoint sets [Ripley, 1996]

- **Training set**: used for learning, e.g., to fit the parameters of the classifier
  - In an MLP, we would use the training set to find the "optimal" weights with the back-prop rule
- **Validation set**: used to select among several trained classifiers
  - In an MLP, we would use the validation set to find the "optimal" number of hidden units or determine a stopping point for the back-propagation algorithm
- **Test set**: used only to assess the performance of a fully-trained classifier
  - In an MLP, we would use the test to estimate the error rate after we have chosen the final model (MLP size and actual weights)

Classifier Evaluation

The Three Ways Data Split

Why separate test and validation sets?

- The error rate of the final model on validation data will be biased (smaller than the true error rate) since the validation set is used to select the final model
- After assessing the final model on the test set, YOU MUST NOT tune the model any further!

Classifier Evaluation

The Three Ways Data Split

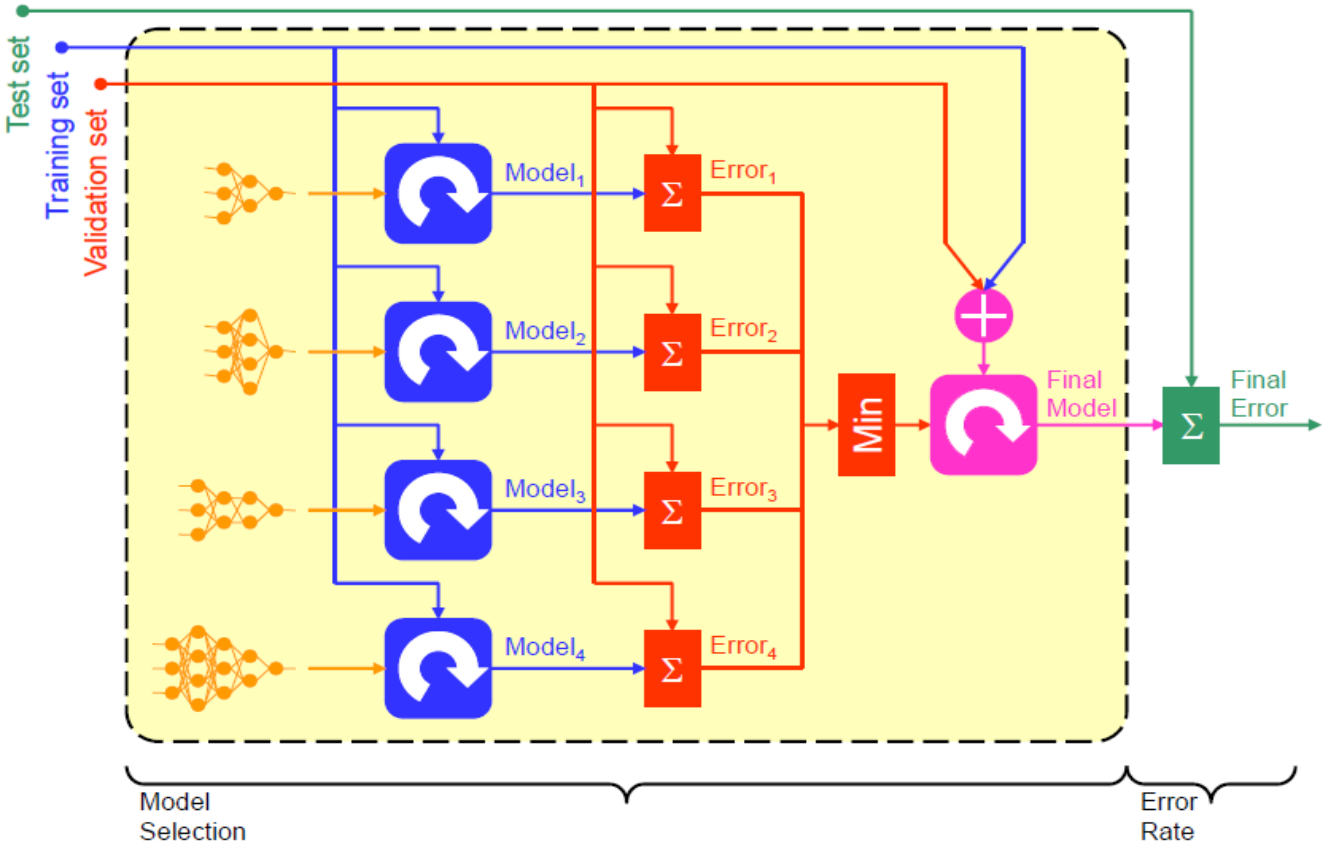
- Procedure outline

1. Divide the available data into training, validation and test set
2. Select architecture and training parameters
3. Train the model using the training set
4. Evaluate the model using the validation set
5. Repeat steps 2 through 4 using different architectures and training parameters
6. Select the best model and train it using data from the training and validation sets
7. Assess this final model using the test set

- This outline assumes a holdout method
- If CV or bootstrap are used, steps 3 and 4 have to be repeated for each of the K folds

Classifier Evaluation

The Three Ways Data Split

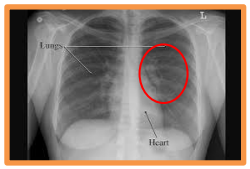


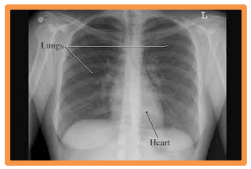
Classifier Evaluation

Cancer **predicted** and **there is cancer**

Hypothesized class	True class: p	True class: n
Y	True Positives	False Positives
N	False Negatives	True Negatives
Column totals:	P	N

Cancer **predicted** and **there is no cancer**

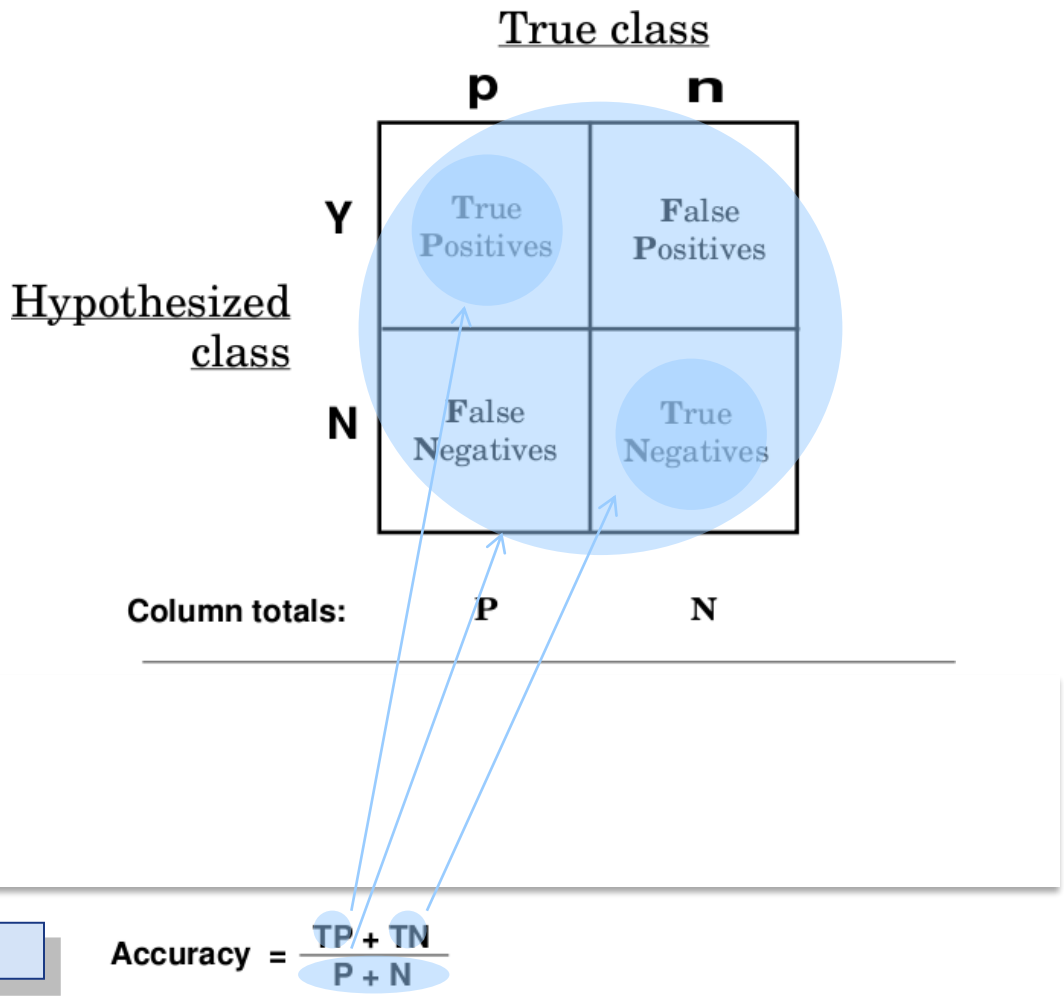




No Cancer **predicted** and **there is no cancer**

$$\text{Accuracy} = \frac{TP + TN}{P + N}$$

Classifier Evaluation

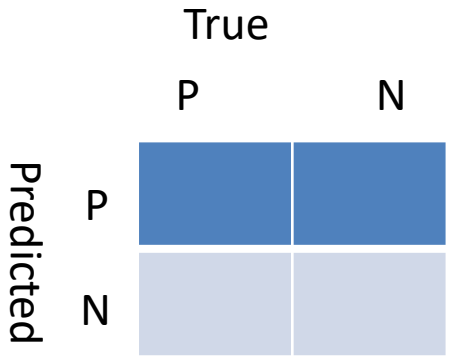


$$\text{Accuracy} = \frac{TP + TN}{P + N}$$

# of correct answers

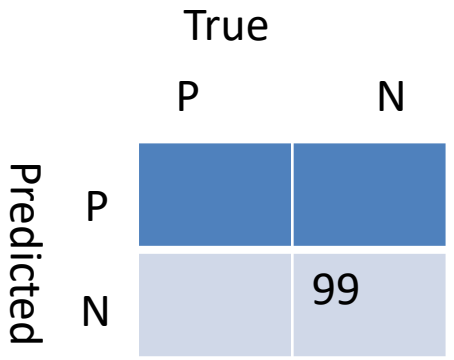
Problems w/ accuracy

- Ex. Mammography
  - The disease occurs < 1%.
  - So let a trained monkey always say there is no disease (without even looking at the film/image)!



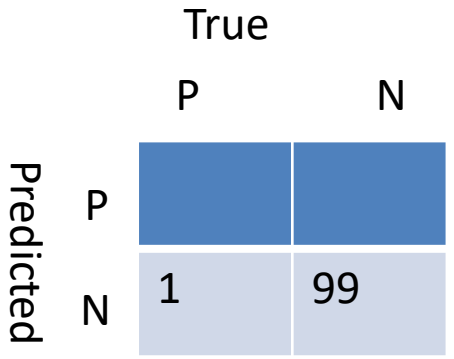
Problems w/ accuracy

- Ex. Mammography
  - The disease occurs < 1%.
  - So let a trained monkey always say there is no disease (without even looking at the film/image)!



Problems w/ accuracy

- Ex. Mammography
  - The disease occurs < 1%.
  - So let a trained monkey always say there is no disease (without even looking at the film/image)!



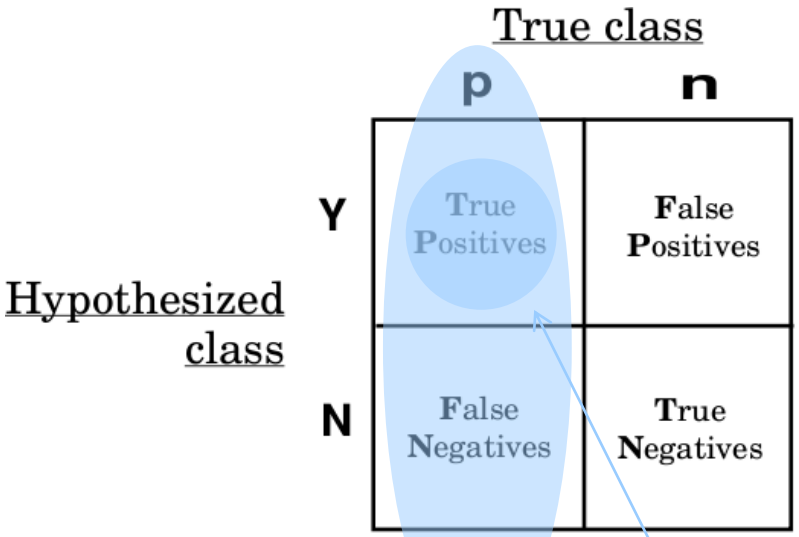
Problems w/ accuracy

- Ex. Mammography
  - The dise
  - So let a t s say there is no disease (at the film/image)!
  - Our mon -99)/(0+99+0+1) = 99% wit edical school!

		True	
		P	N
Predicted	P	0	0
	N	1	99

- We need a different measure!

Classifier Evaluation



FP rate = $\frac{FP}{N}$

TP rate = $\frac{TP}{P}$ = Recall

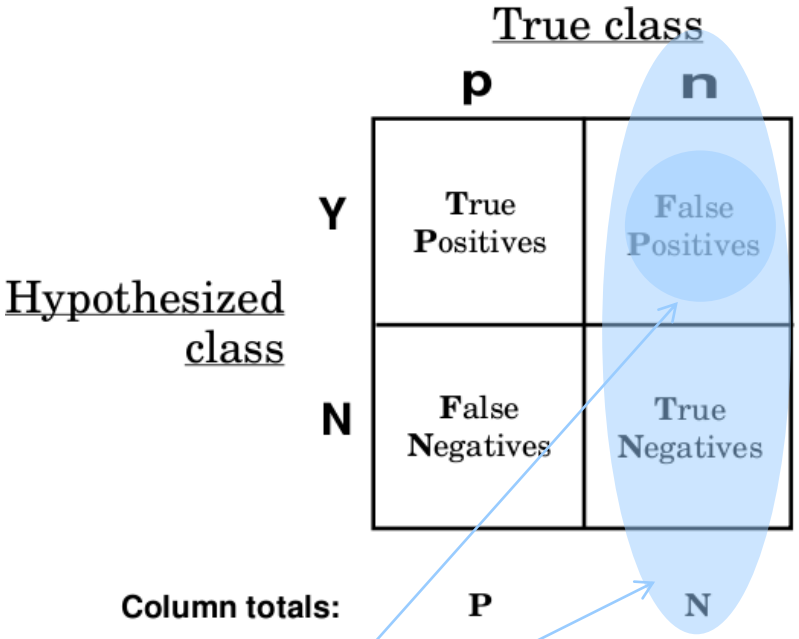
Sensitivity

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

Accuracy = $\frac{TP + TN}{P + N}$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Classifier Evaluation



1-Specificity

$$\text{FP rate} = \frac{FP}{N}$$

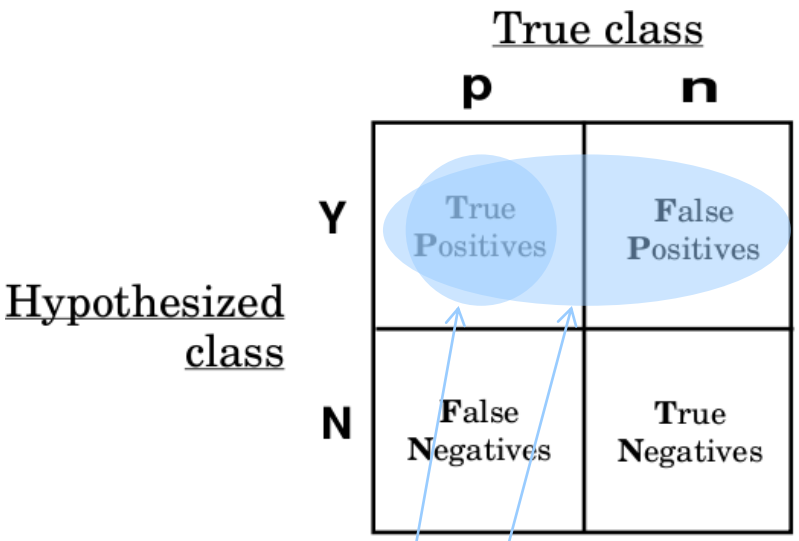
$$\text{TP rate} = \frac{TP}{P} = \text{Recall}$$

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

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{Accuracy} = \frac{TP + TN}{P + N}$$

Classifier Evaluation



Column totals: P N

$$\text{FP rate} = \frac{FP}{N} \qquad \text{TP rate} = \frac{TP}{P} = \text{Recall}$$

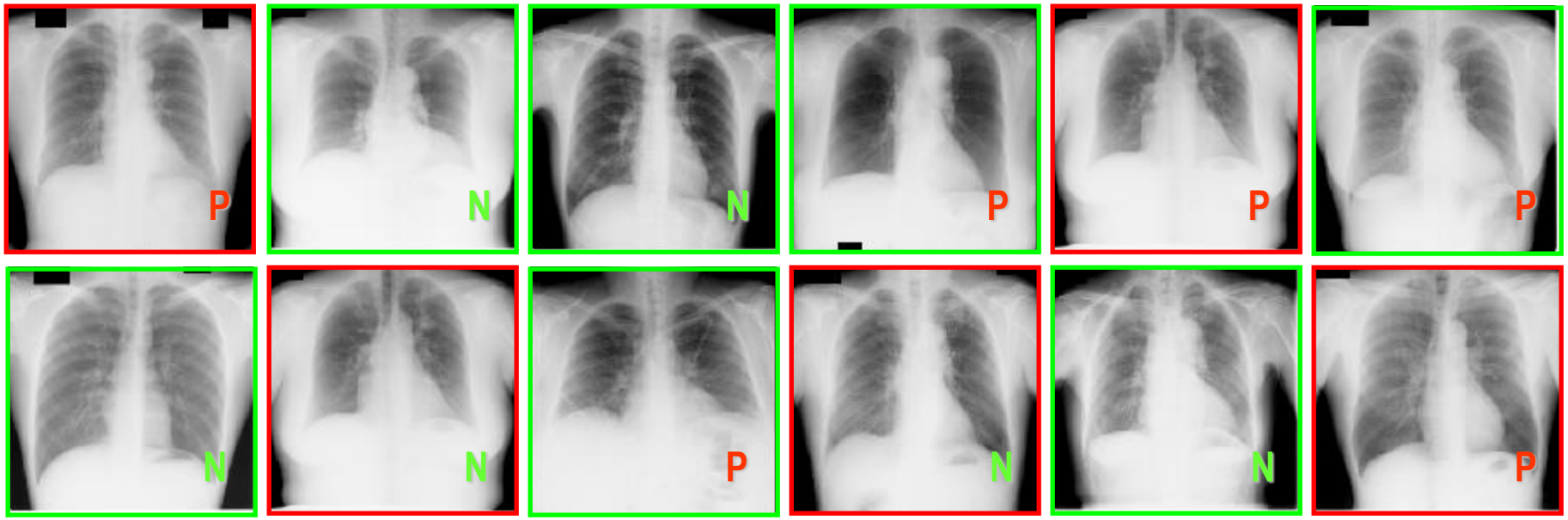
$$\text{Precision} = \frac{TP}{TP + FP}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{Accuracy} = \frac{TP + TN}{P + N}$$

Positive Predictive Value

Sensitivity & Specificity



P true positives (TP)

P false positives (FP)

N false negatives (FN)

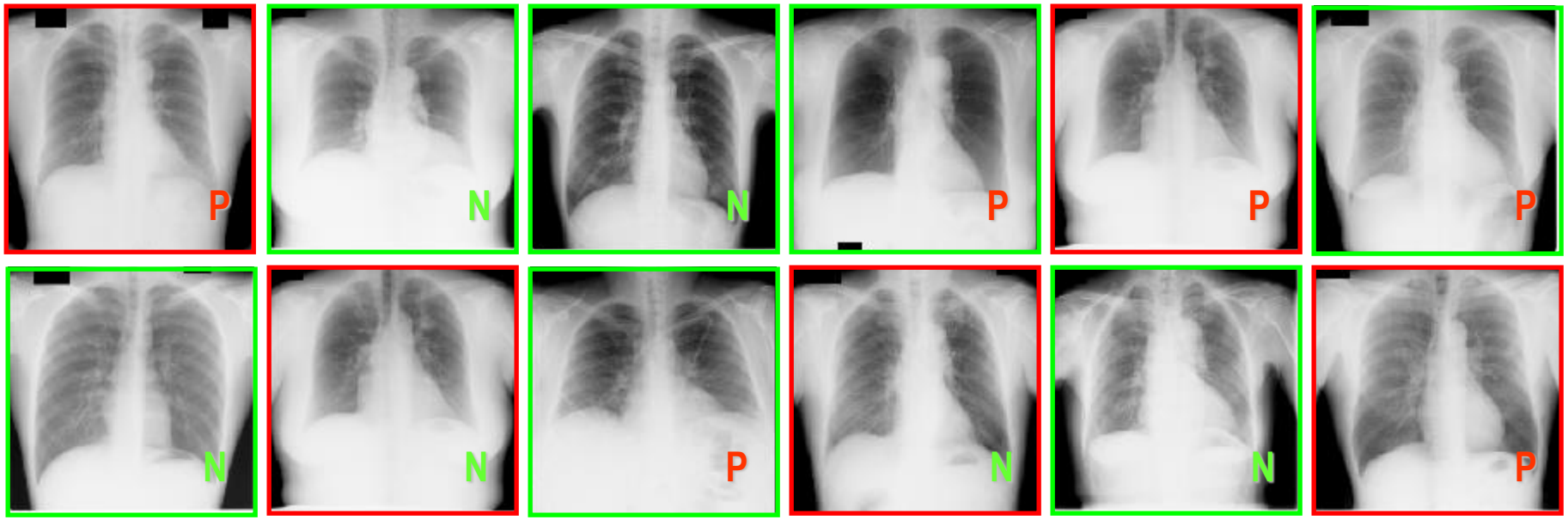
N true negatives (TN)

sensitivity = true positive fraction
= 1 – false negative fraction
= TP / (TP + FN)

specificity = true negative fraction
= 1 – false positive fraction
= TN / (TN + FP)

accuracy = (TP+TN) / (TP+TN+FP+FN)

Sensitivity & Specificity



P true positives (TP) = 3

P false positives (FP) = 3

N false negatives (FN) = 2

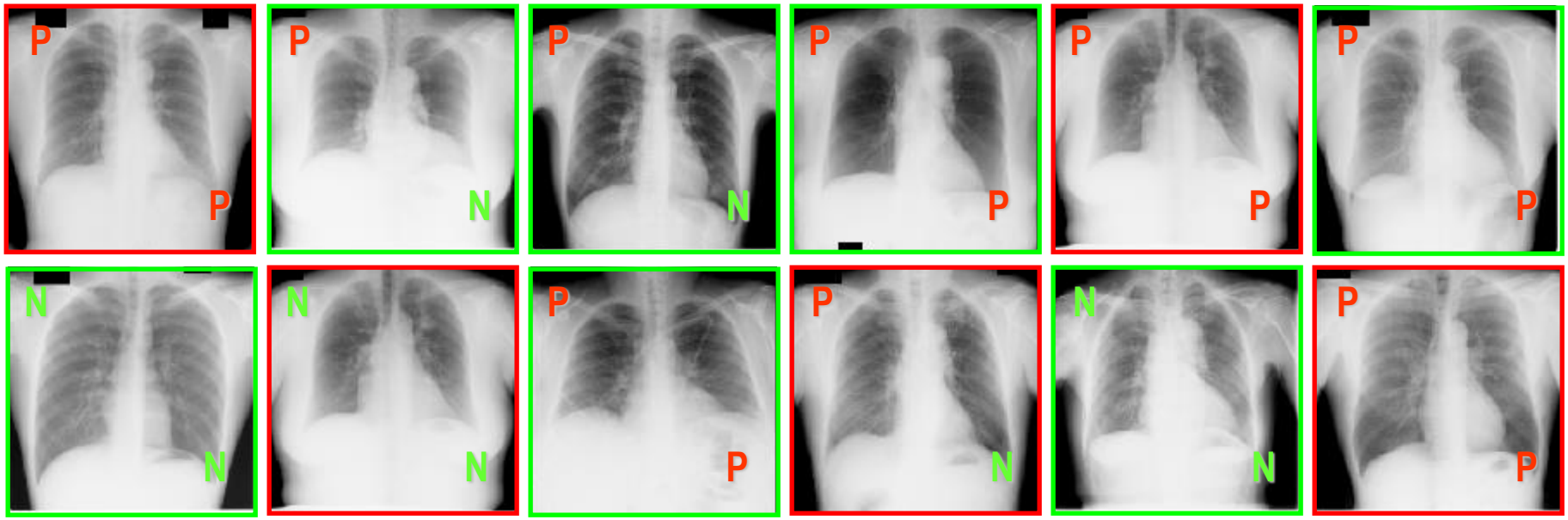
N true negatives (TN) = 4

sensitivity = TP / (TP + FN)
= 3 / 5 = 0.6

specificity = TN / (TN + FP)
= 4 / 7 = 0.57

accuracy = (TP+TN) /
(TP+TN+FP+FN) = 7 / 12 = 0.58

Sensitivity & Specificity



algorithm 1

P = 3 N = 2

P = 3 N = 4

sensitivity = 3 / 5 = 0.6
specificity = 4 / 7 = 0.57
accuracy = 7 / 12 = 0.58

algorithm 2

P = 4 N = 1

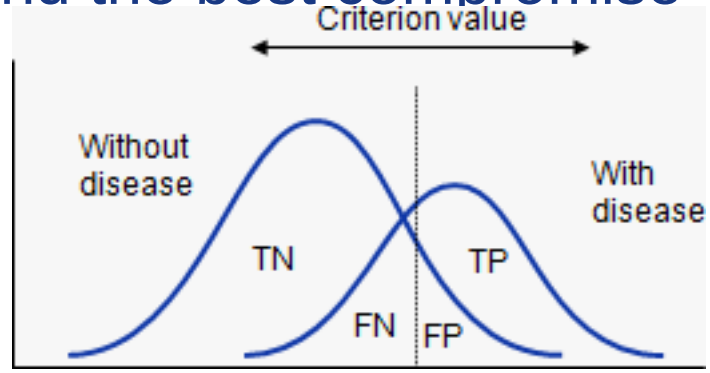
P = 5 N = 2

sensitivity = 4 / 5 = 0.8
specificity = 2 / 7 = 0.29
accuracy = 6 / 12 = 0.5

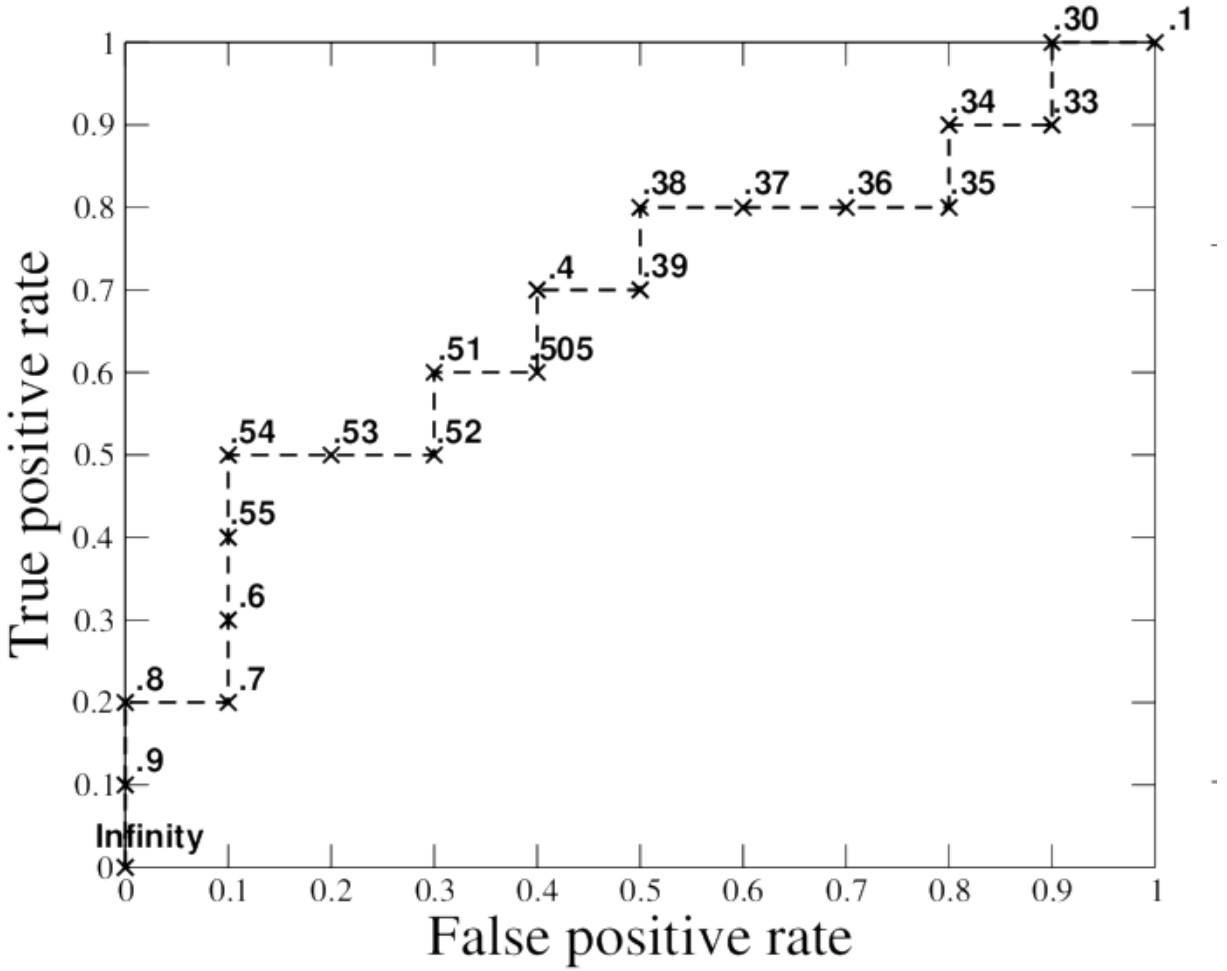
Which system is better?

ROC Curve

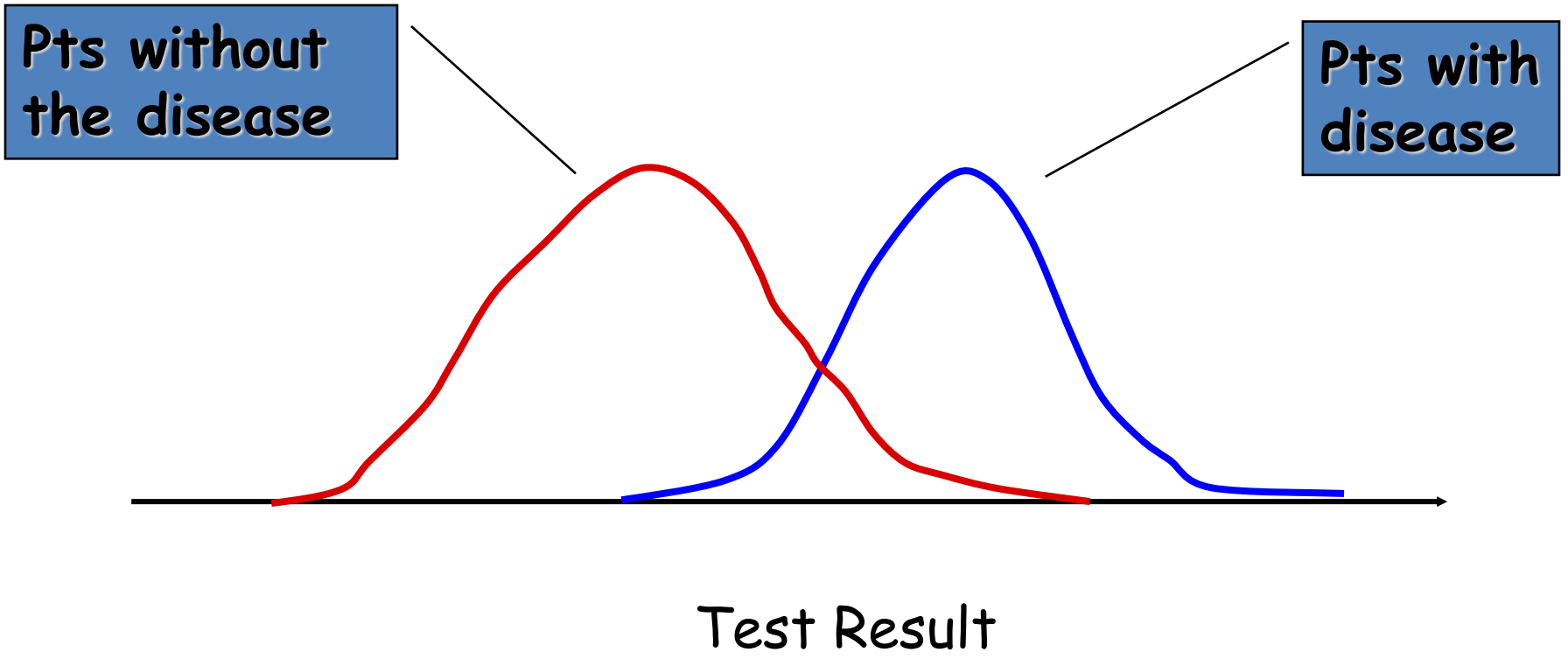
- Receiving Operating Characteristic curve (ROC)
  - Standard way to evaluate performance of a 2-class classifier
  - Evaluates Sensitivity and Specificity
  - Interdisciplinary (especially in the medical field)
  - Can be summarised by a single number Area Under roC (AUC)
- Used to find the best compromise



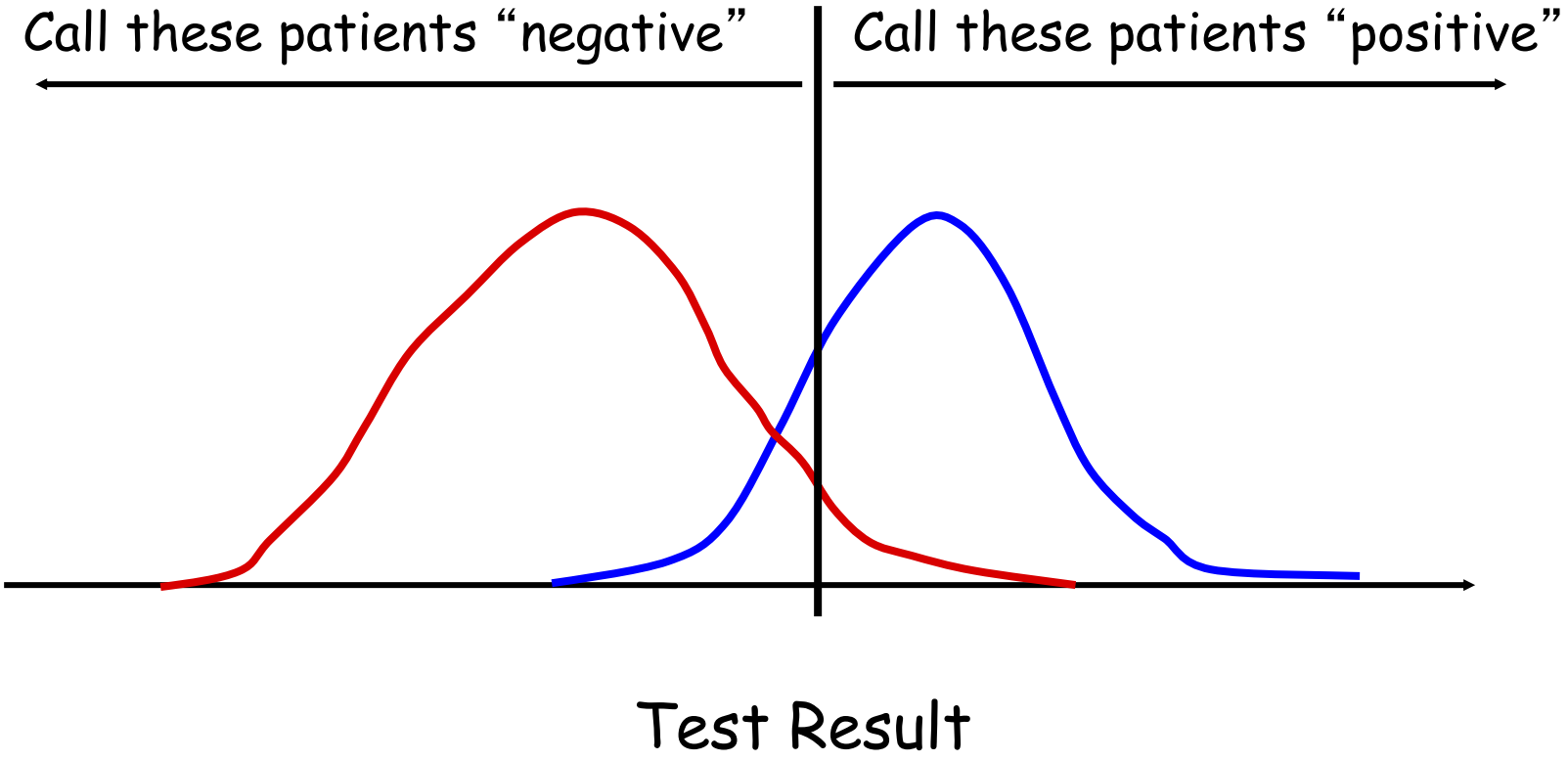
Building a ROC Curve



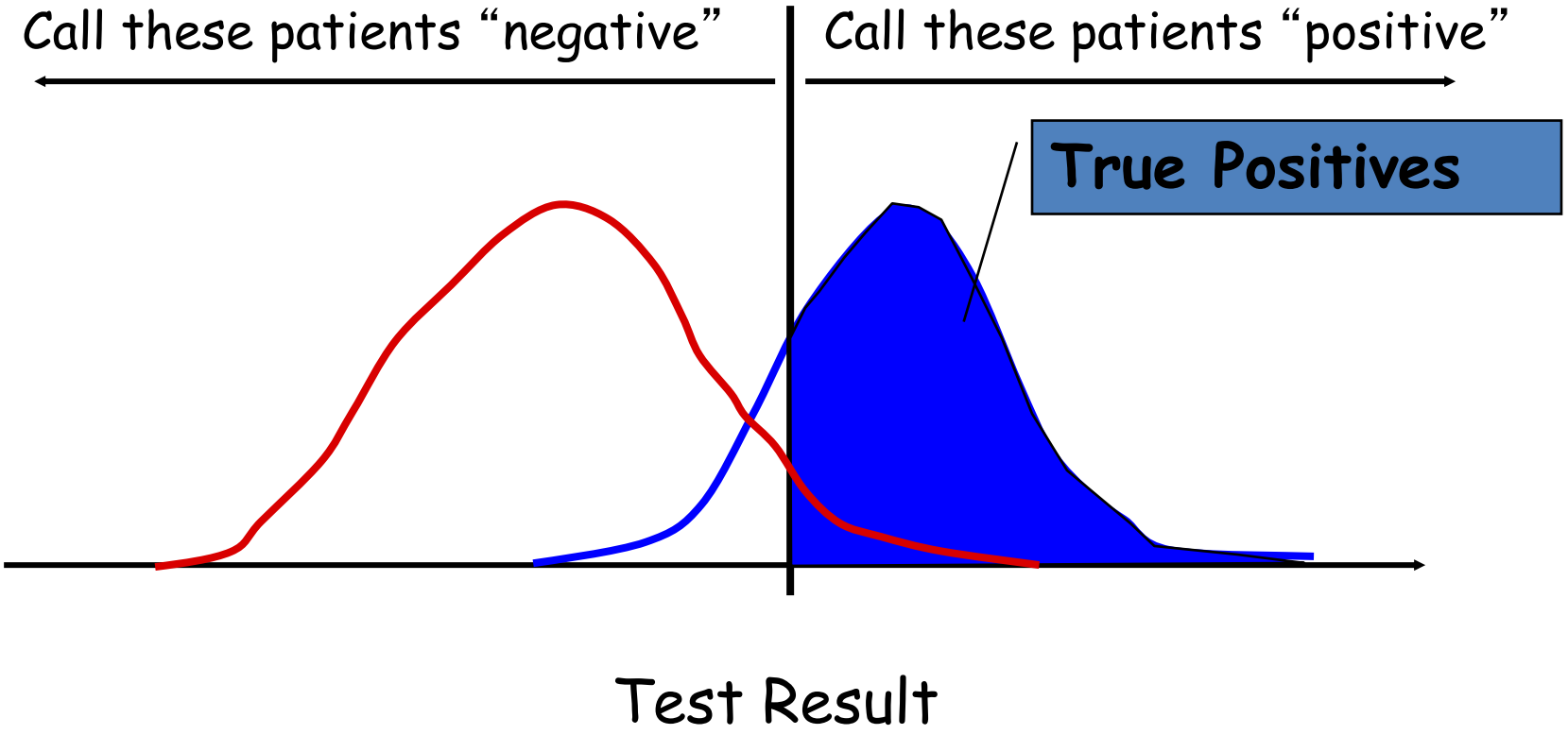
Specific Example



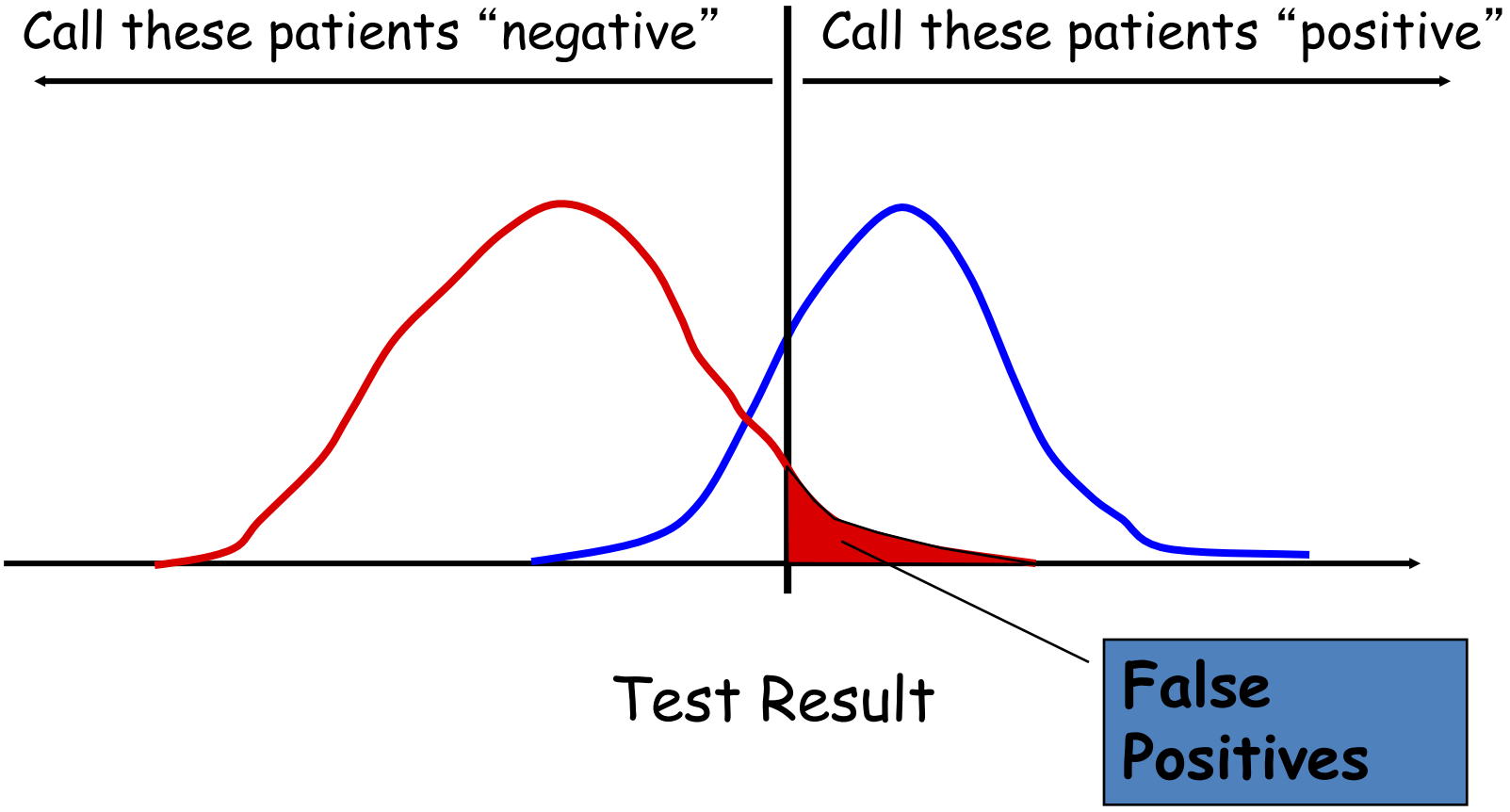
Threshold


Call these patients “negative”
Call these patients “positive”
Test Result

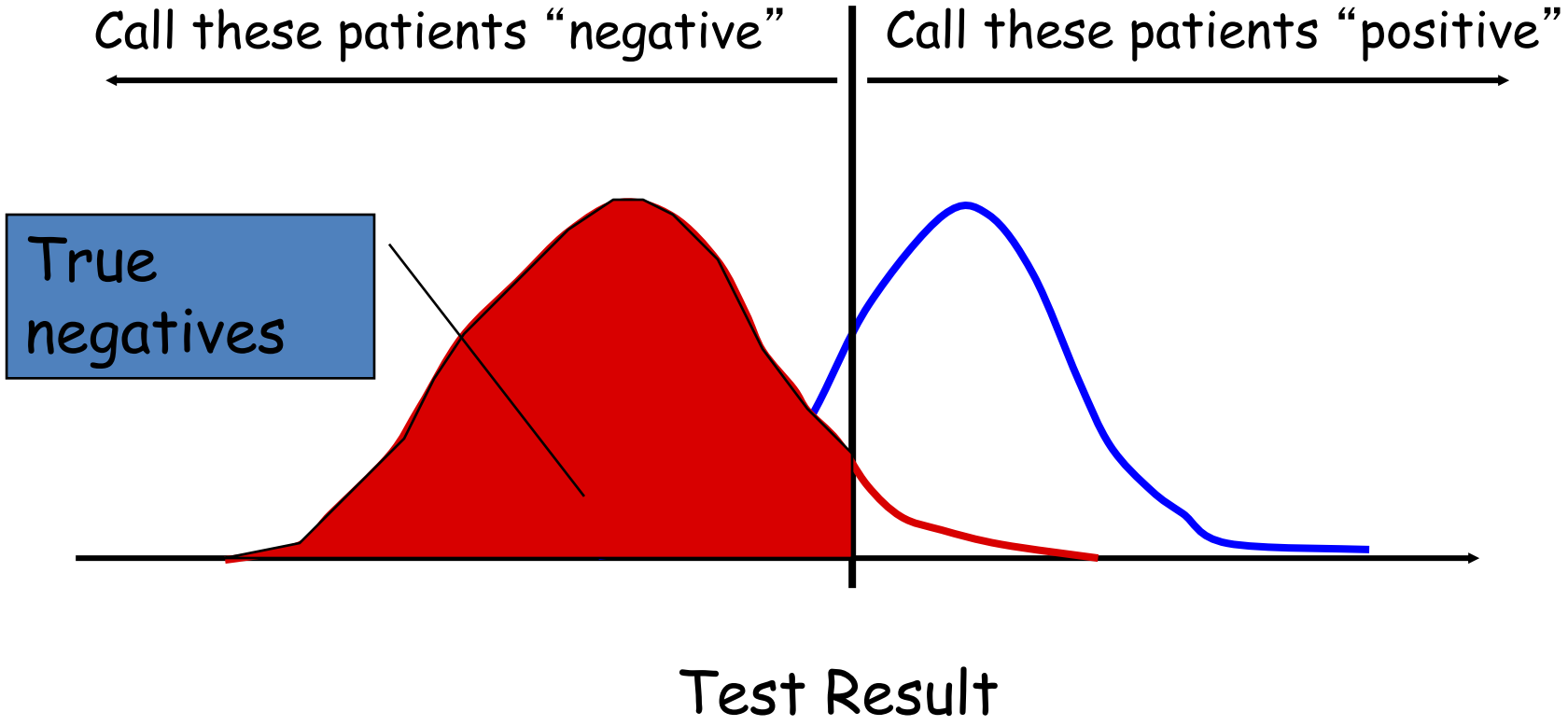
Some definitions ...



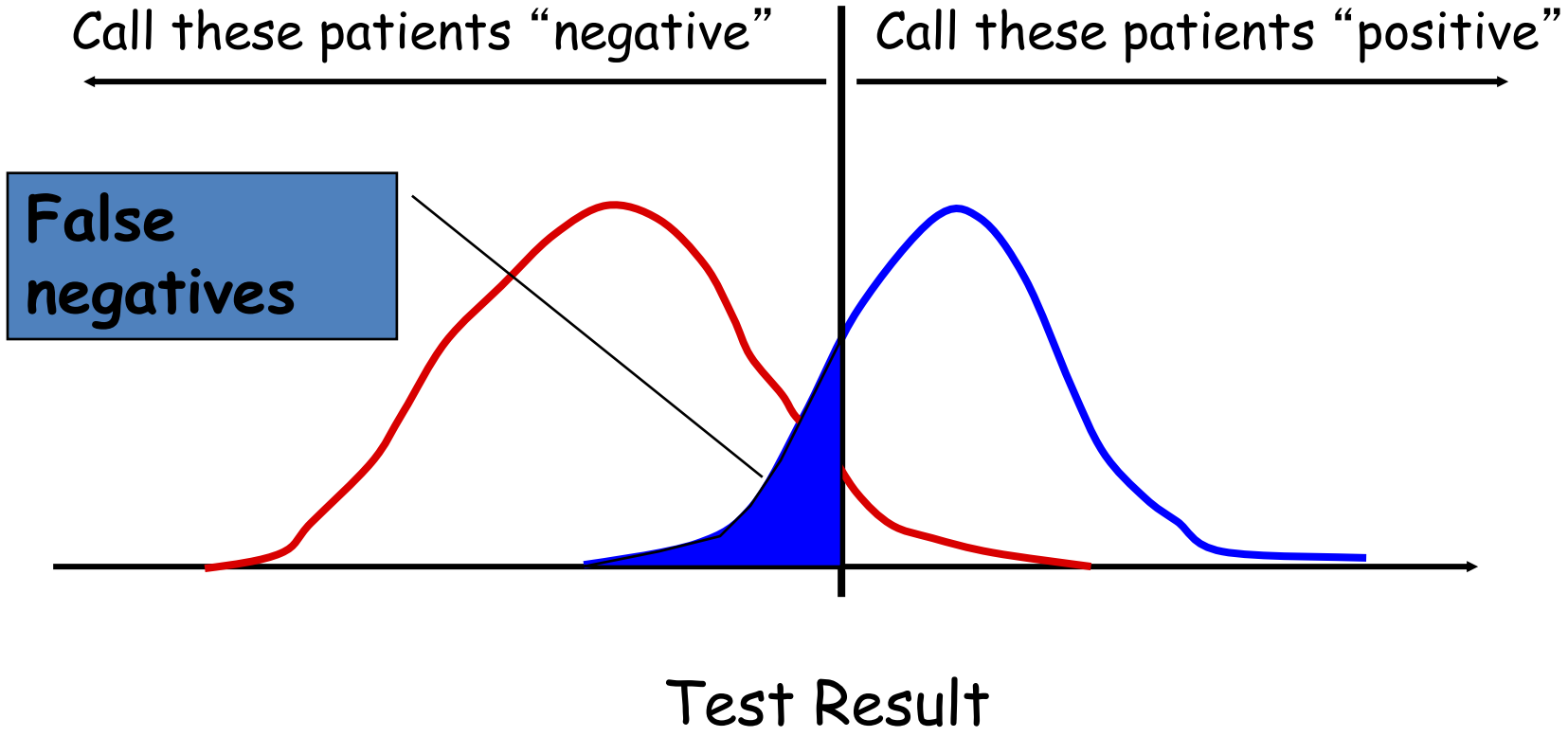
without the disease
with the disease



without the disease
with the disease

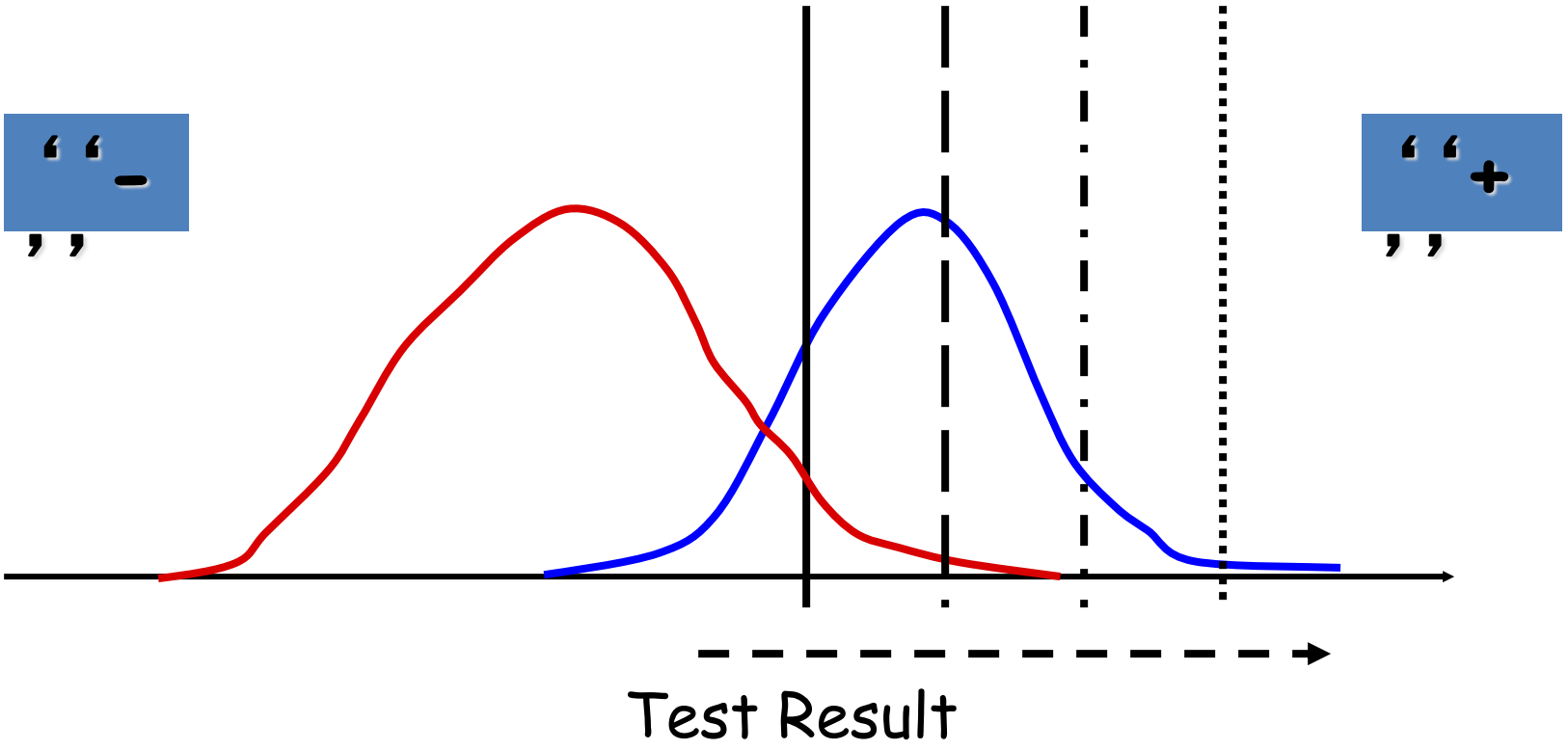


without the disease
with the disease



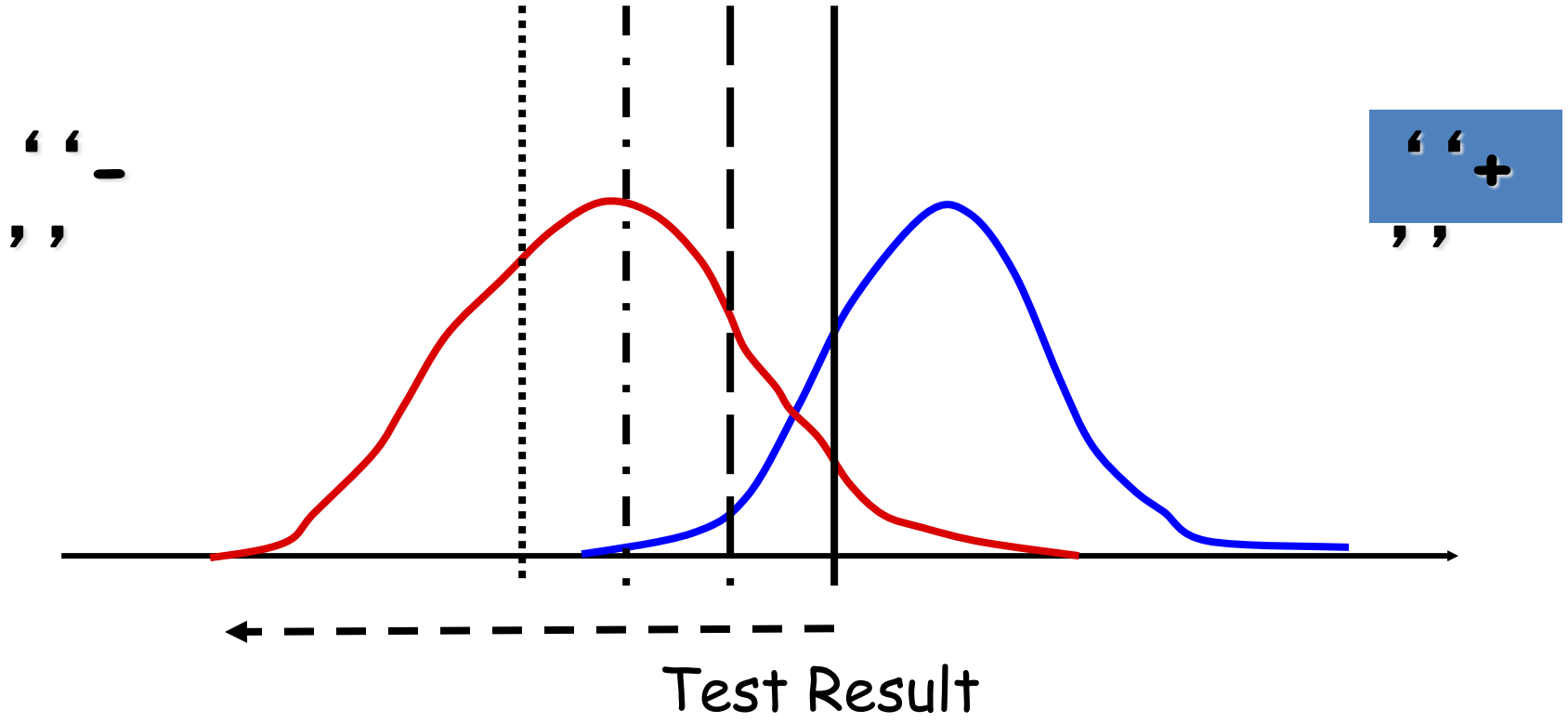
without the disease
with the disease

Moving the Threshold: right



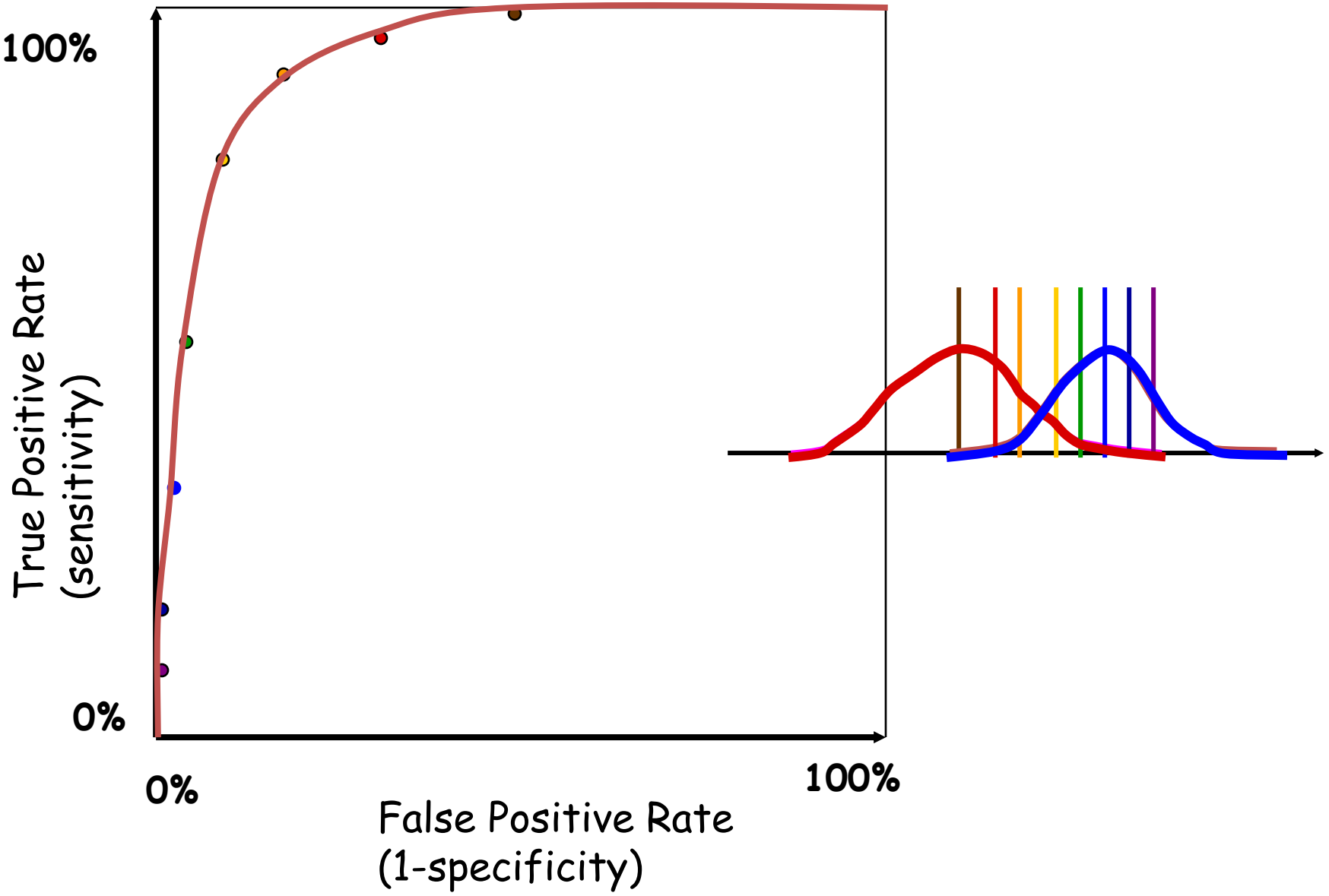
without the disease
with the disease

Moving the Threshold: left



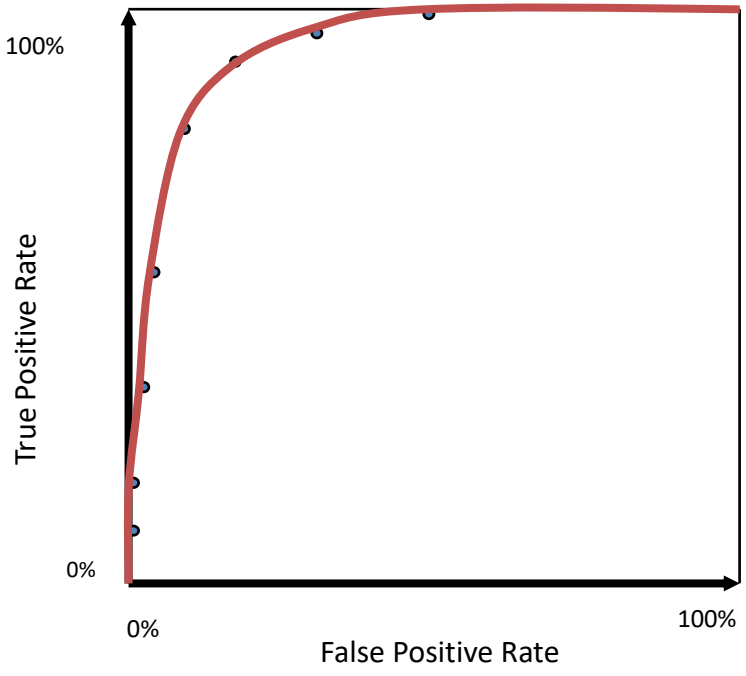
without the disease
with the disease

ROC curve

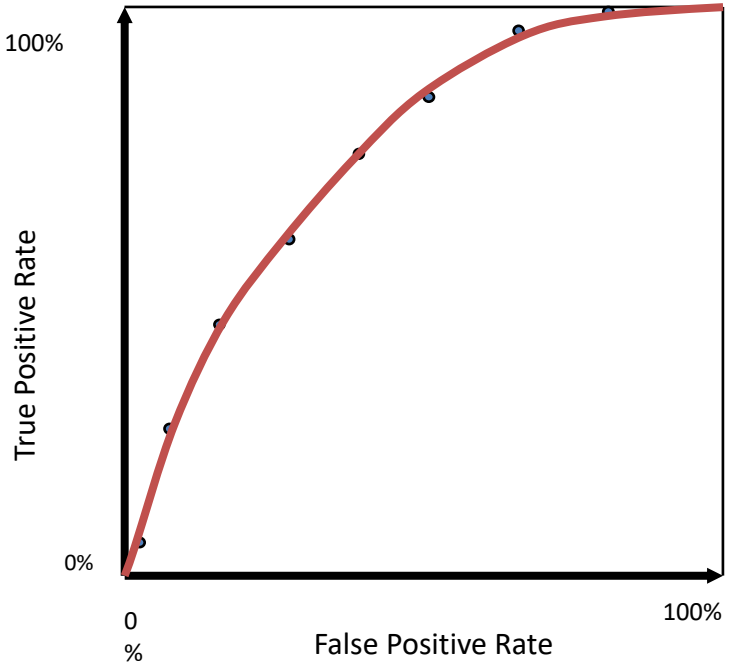

100%
True Positive Rate
(sensitivity)
0%
0%
100%
False Positive Rate
(1-specificity)

ROC curve comparison

A good test:

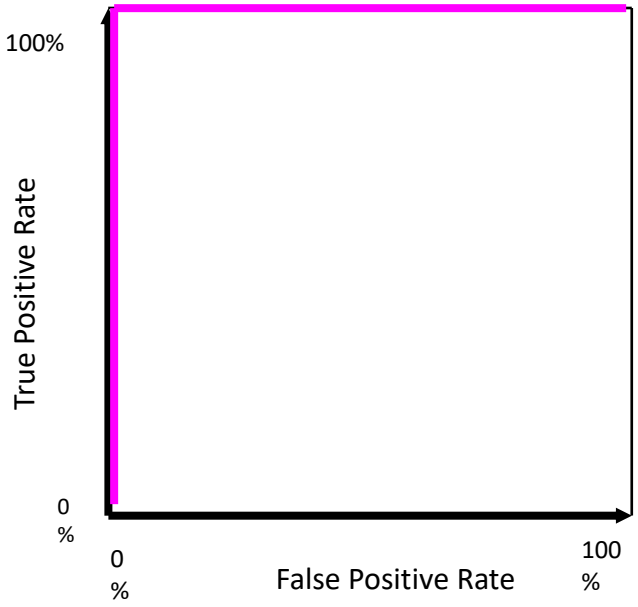


A poor test:



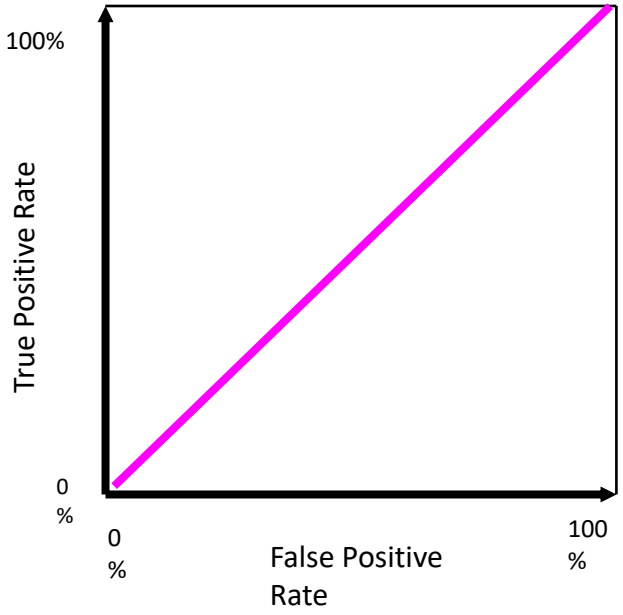
ROC curve extremes

Best Test:



The distributions don't overlap at all

Worst test:

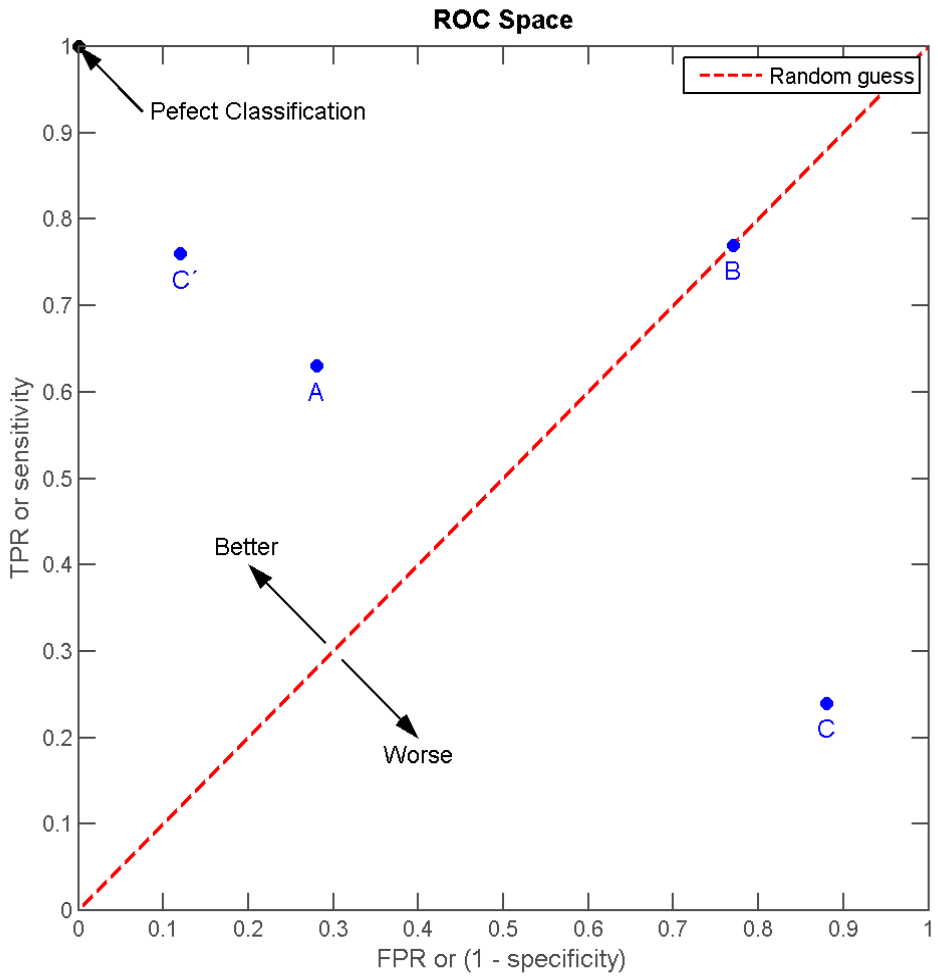


The distributions overlap completely

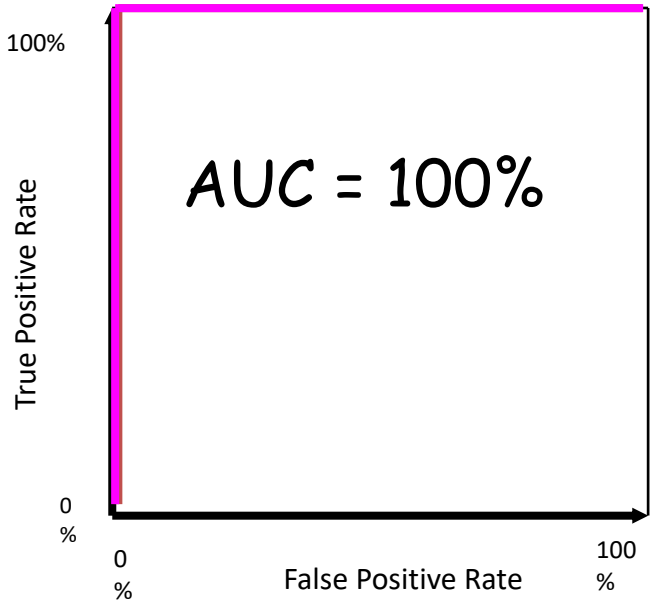
Area under ROC curve (AUC)

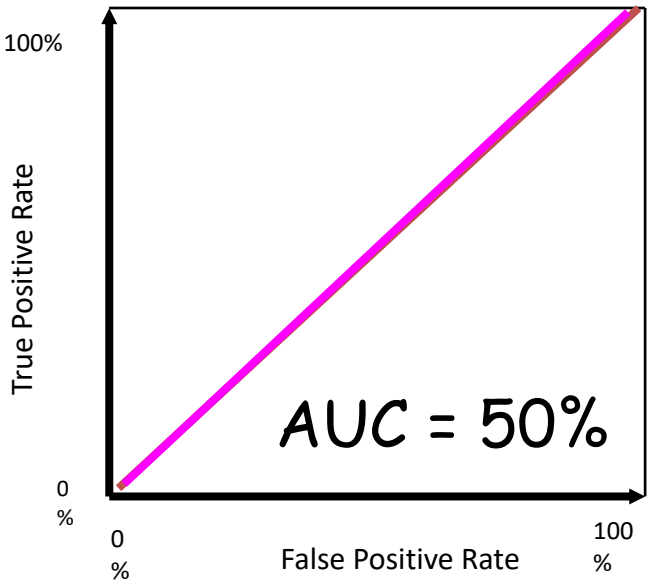
- *Overall measure* of test performance
- *Comparisons* between two tests based on differences between (estimated) AUC

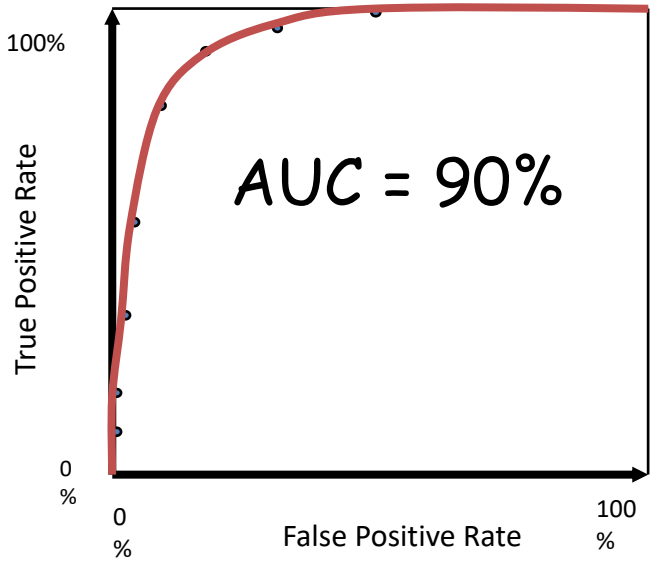
ROC Curve interpretation

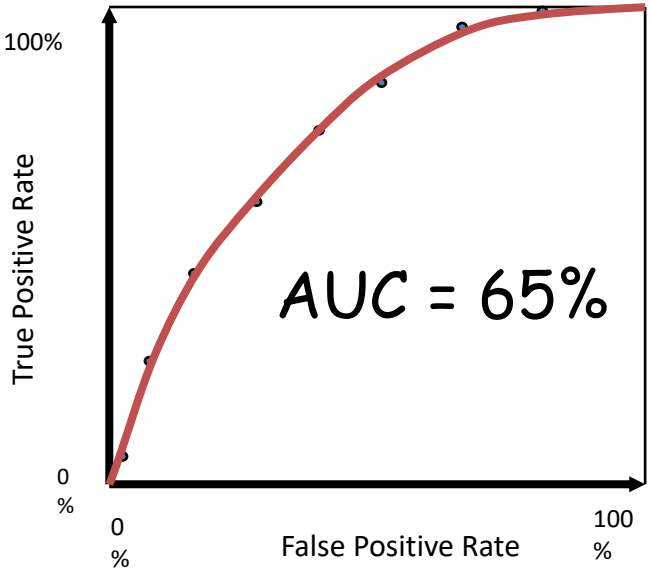


AUC for ROC curves

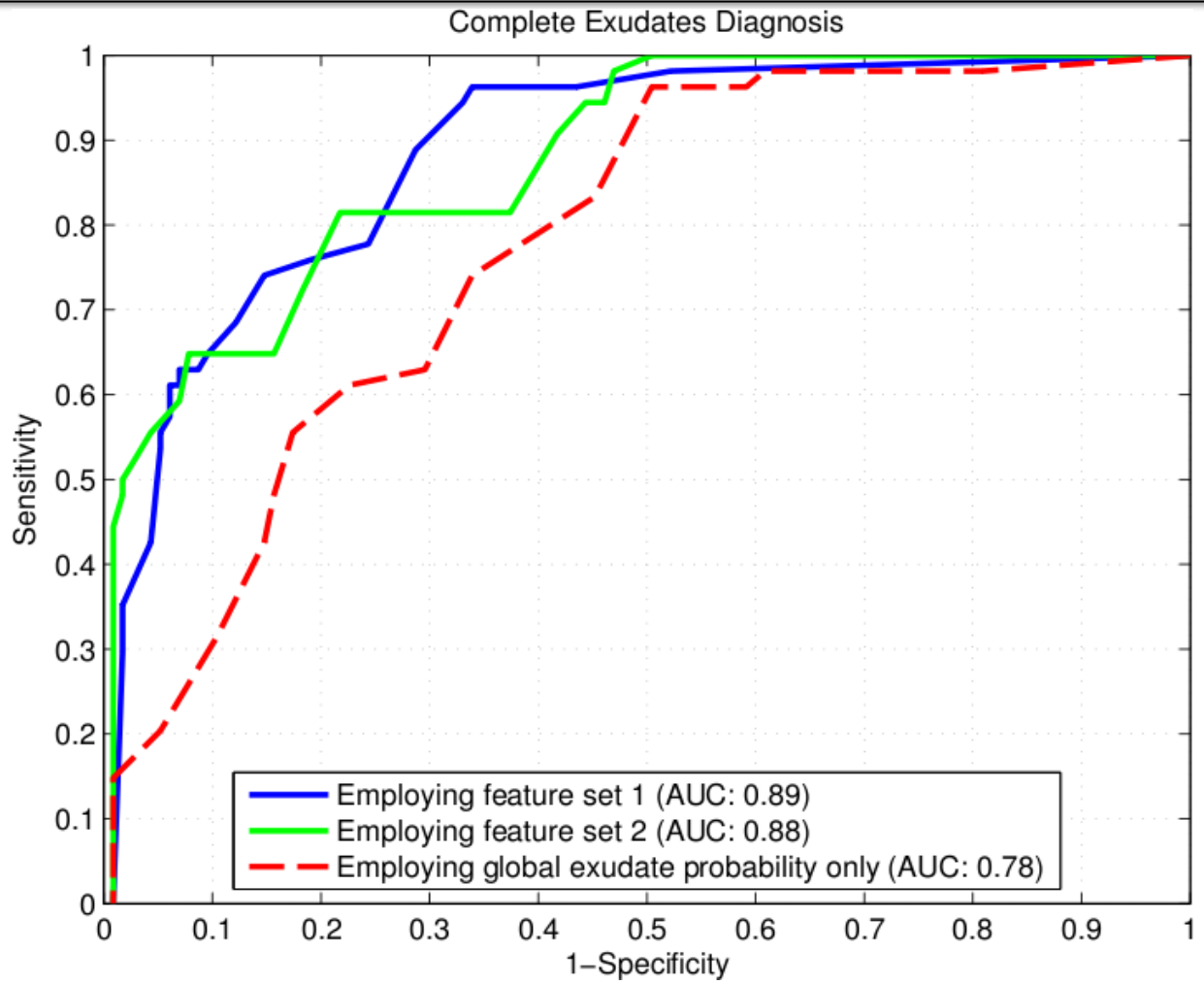






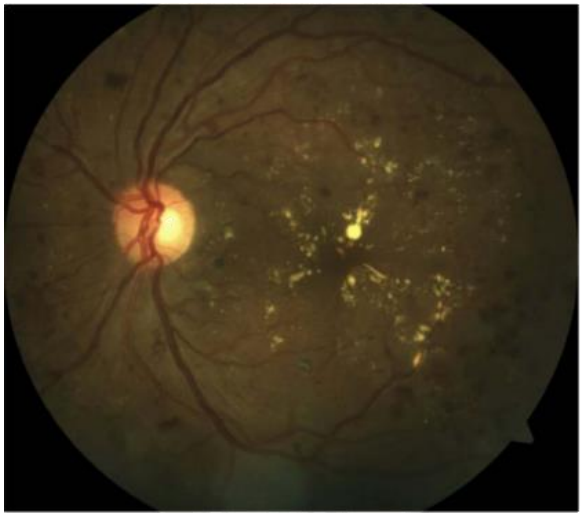


ROC Curves examples



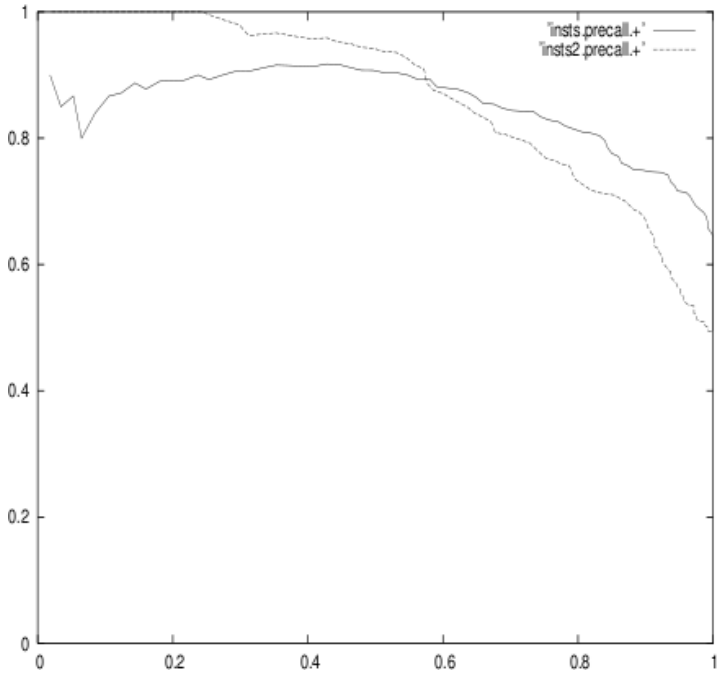
Issues with ROC

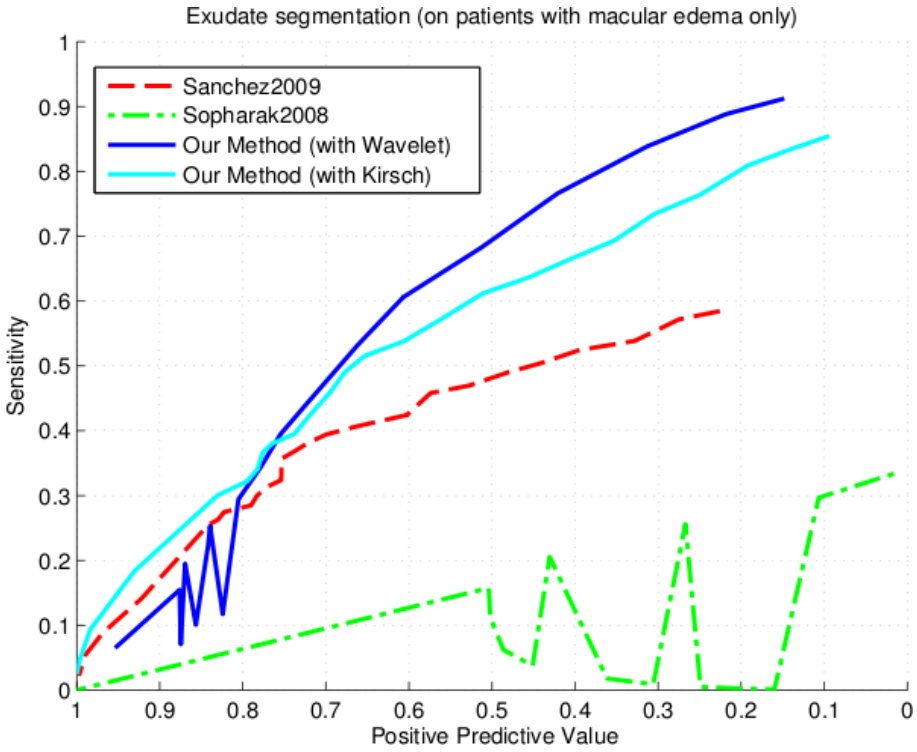
- Not always the ROC curve is the ideal
- Typical Examples
  - Multiclass classification
  - Fraction of TP/FP vastly exceed the other
  - Lesions detection (number)/ real size
- The selection of TP/FP is not always obvious



Issues with ROC

- If TN not available
  - Precision-recall curve or Sensitivity-PPV, or FROC

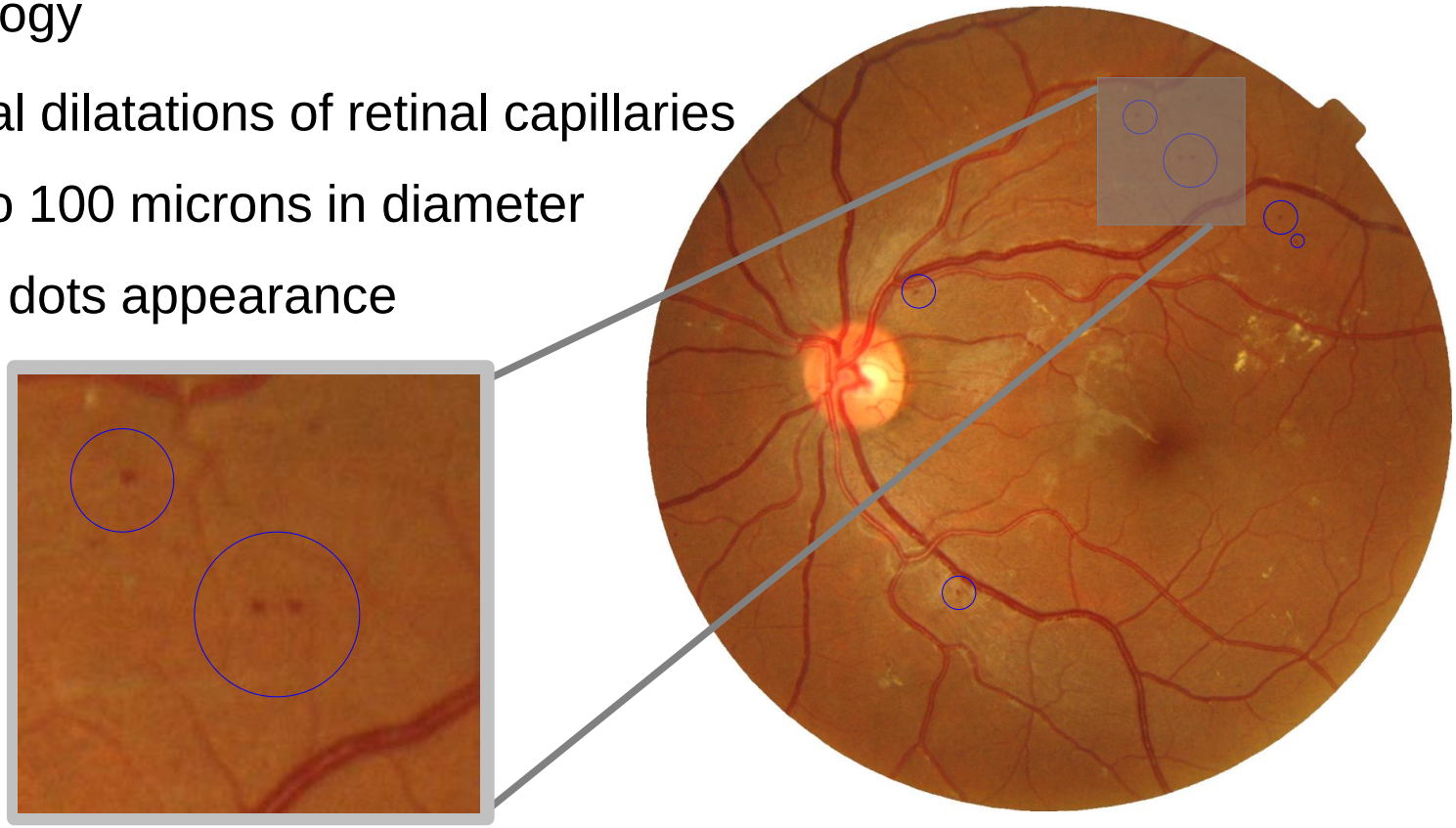




FROC

Examples :Microaneurysms

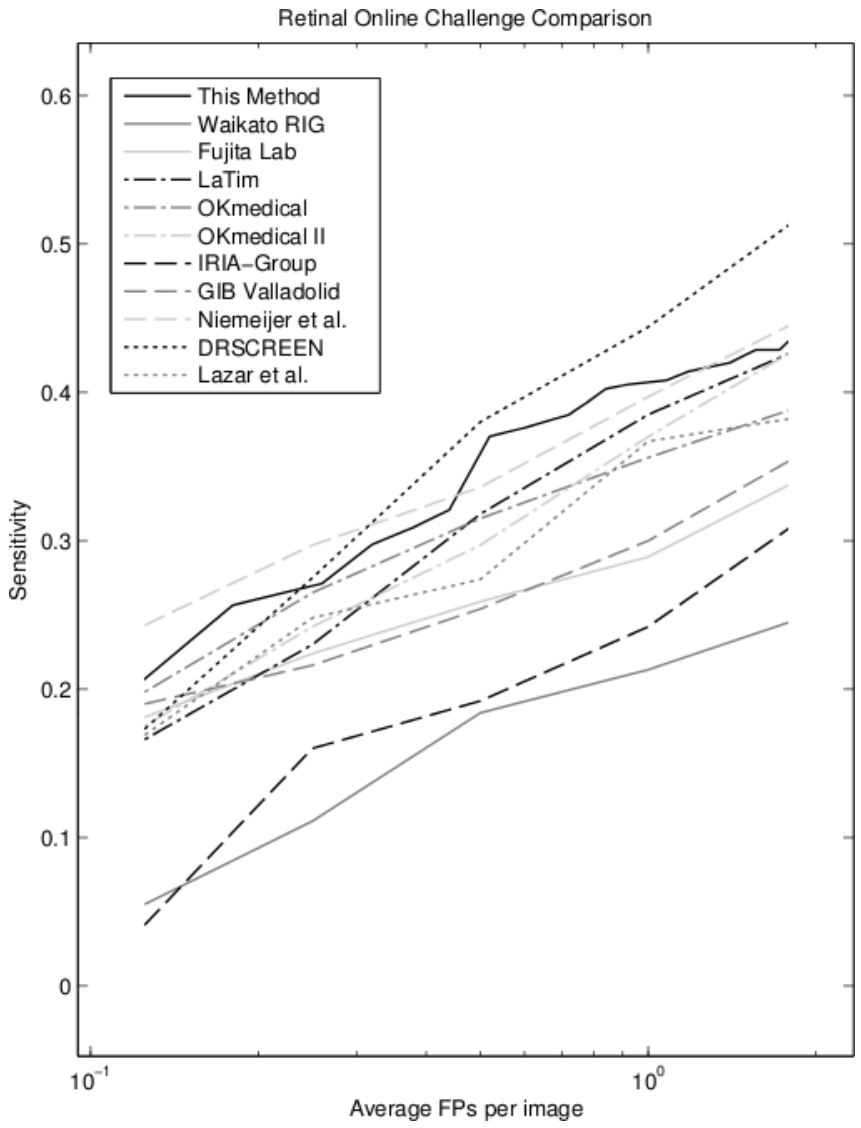
- First detectable sign of Diabetic Retinopathy
- Morphology
  - Focal dilatations of retinal capillaries
  - 10 to 100 microns in diameter
  - Red dots appearance



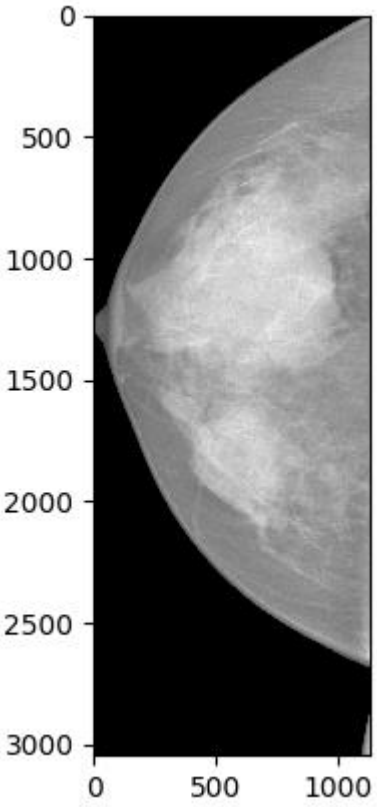
FROC

Examples :Microaneurysms

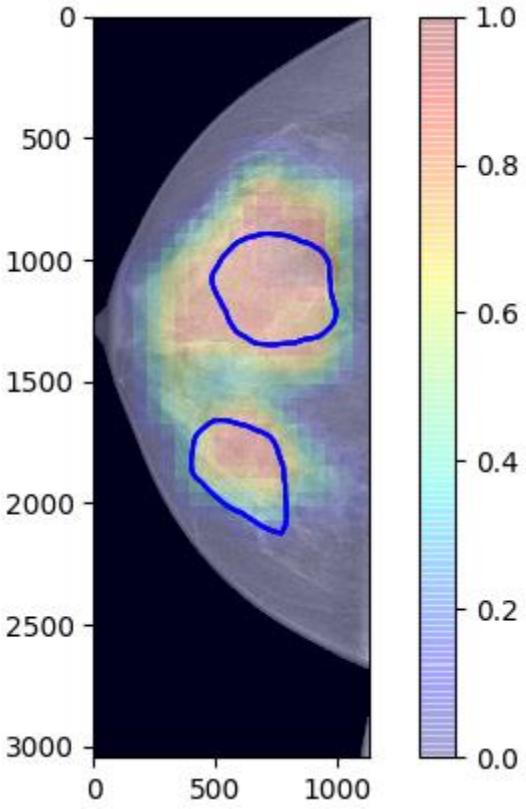
MA Detection Results



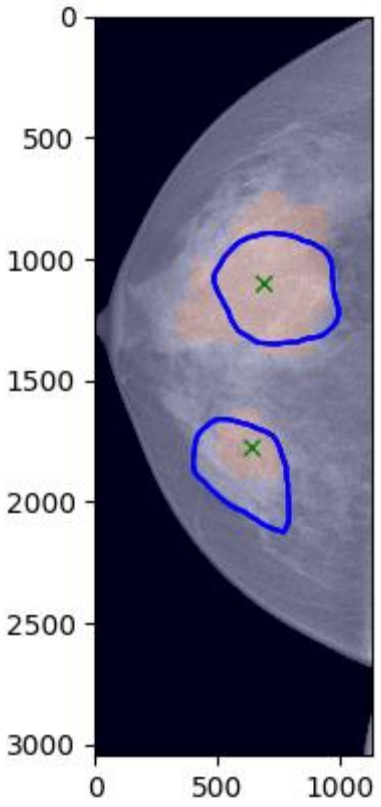
FROC



original image



original image with mass probability colormap and ground truth (blue)



original image with probability map thresholded at 0.75 and mass centroid represented with a cross

Confusion Matrix

- Useful to visualise multiple classes

		Predicted class		
		Cat	Dog	Rabbit
Actual class	Cat	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

6

8

13

Confusion Matrix→ Table of Confusion

		Predicted class		
		Cat	**Dog**	**Rabbit**
Actual class	**Cat**	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

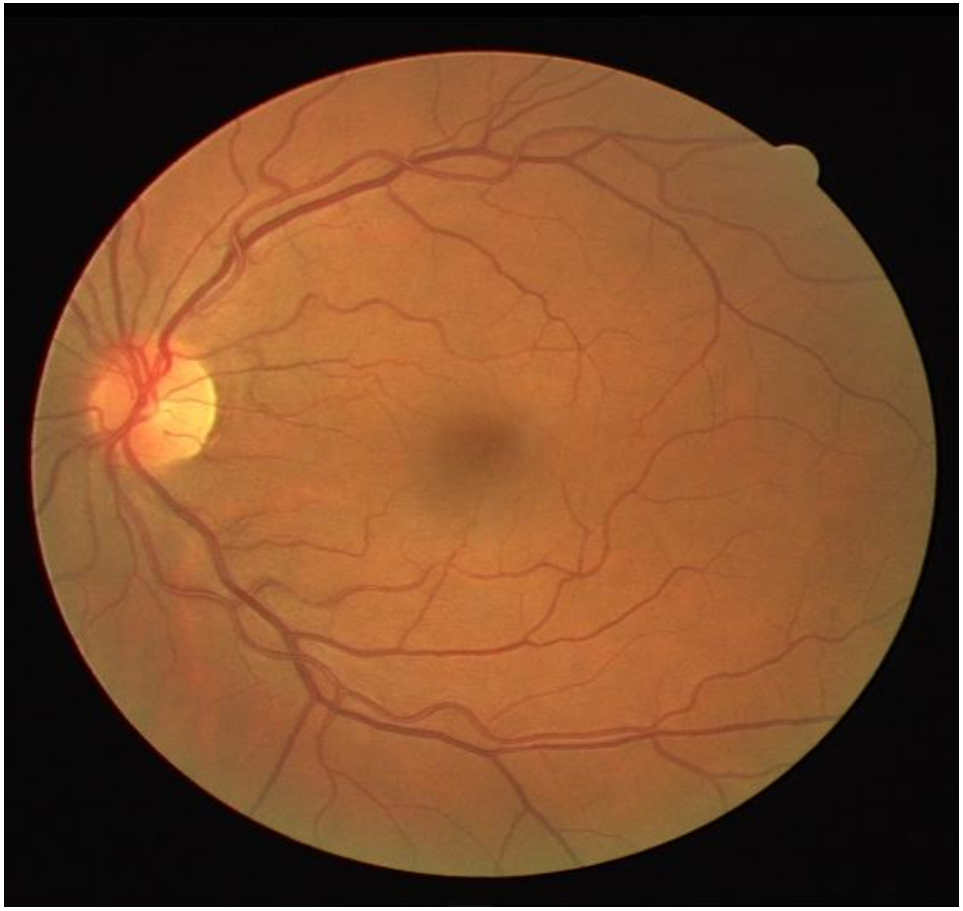
5 true positives (actual cats that were correctly classified as cats)	3 false negatives (cats that were incorrectly marked as dogs)
2 false positives (dogs that were incorrectly labeled as cats)	17 true negatives (all the remaining animals, correctly classified as non-cats)

6

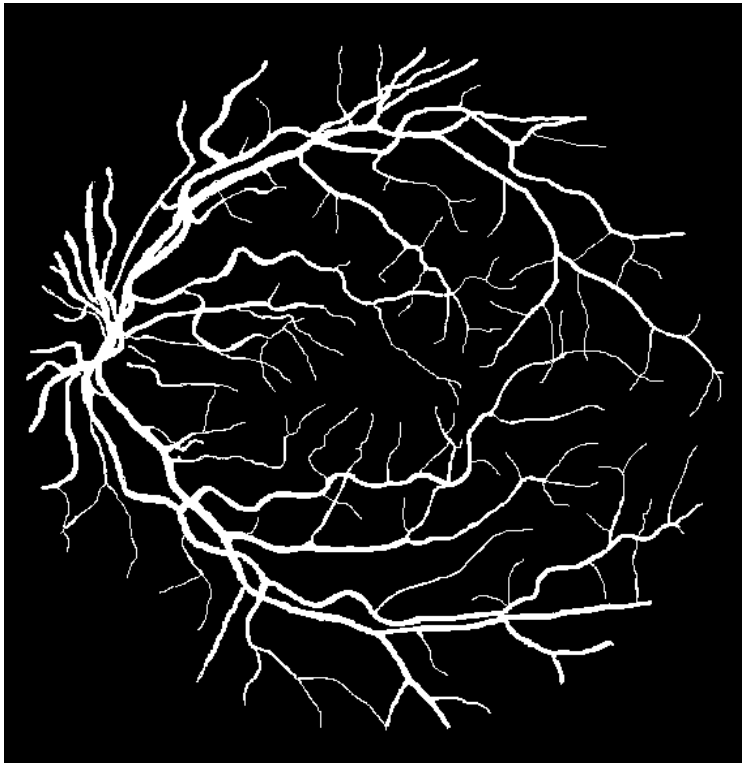
8

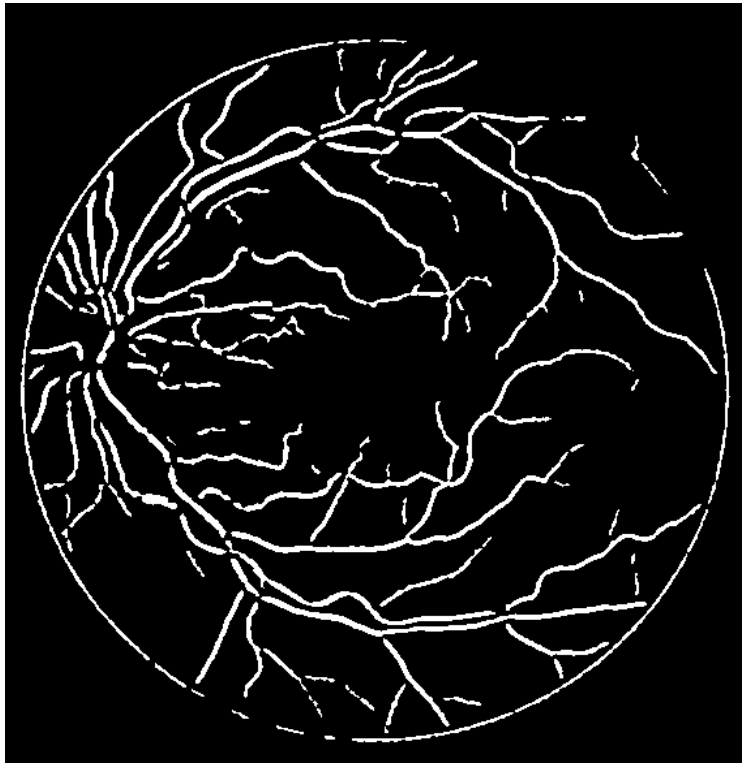
13

Evaluating segmentation



Reference standard & segmentation





Segmentation performance

- Qualitative/subjective evaluation → the easy way out, sometimes the only option
- Quantitative evaluation preferable in general
- A wild variety of performance measures exists
- Many measures are applicable outside the segmentation domain as well
- Focus here is on two class problems

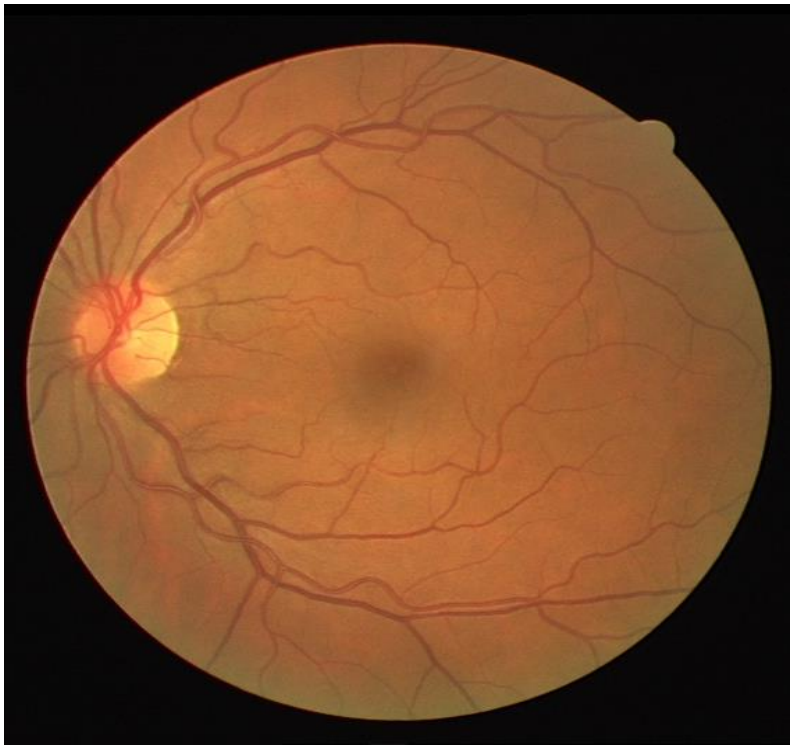
Some terms

- **Ground truth** = the real thing
- **Gold standard** = the best we can get
- **Bronze standard** = gold standard with limitations
- **Reference standard** = preferred term for gold standard in the medical community

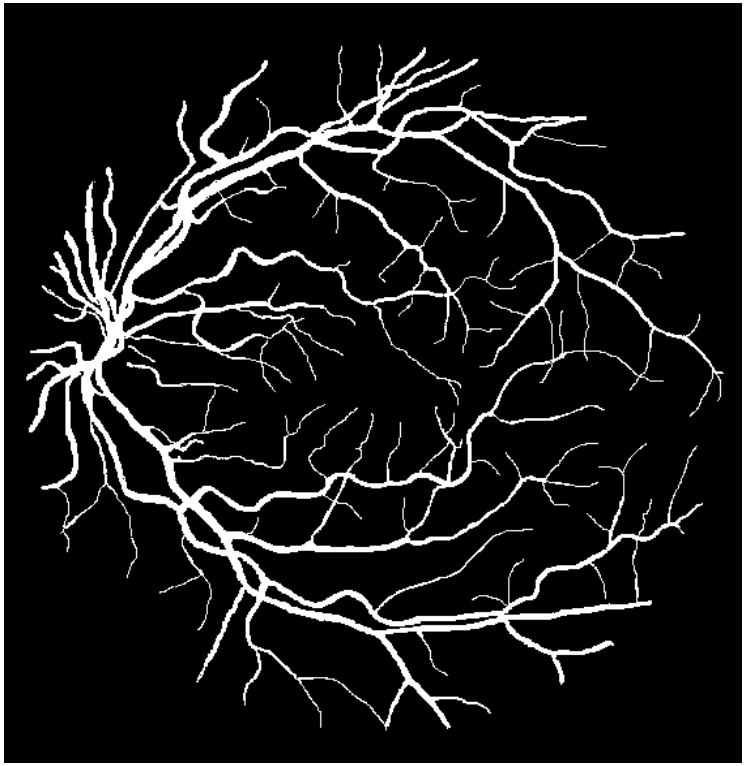
What to evaluate?

- Without reference standard, subjective or qualitative evaluation is hard to avoid
- Region/pixel based comparisons
- Border/surface comparisons
- (a selection of) Points
- Global performance measures versus local measures

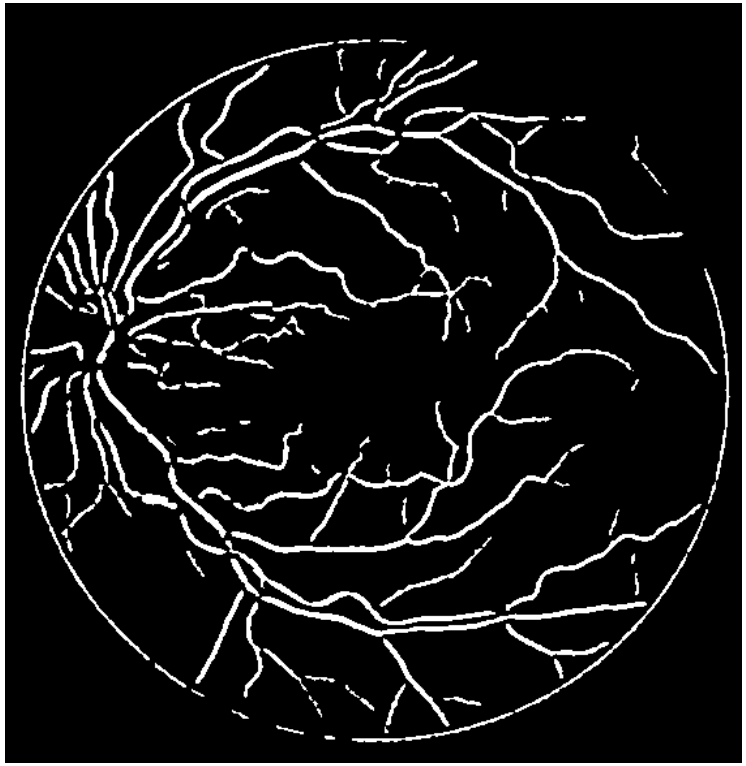
Example



Example

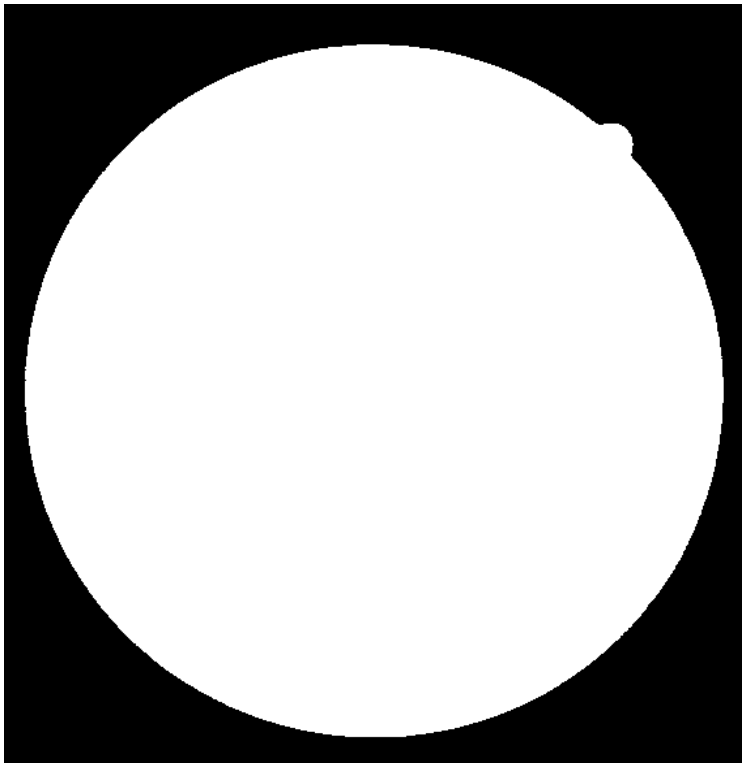


Reference standard

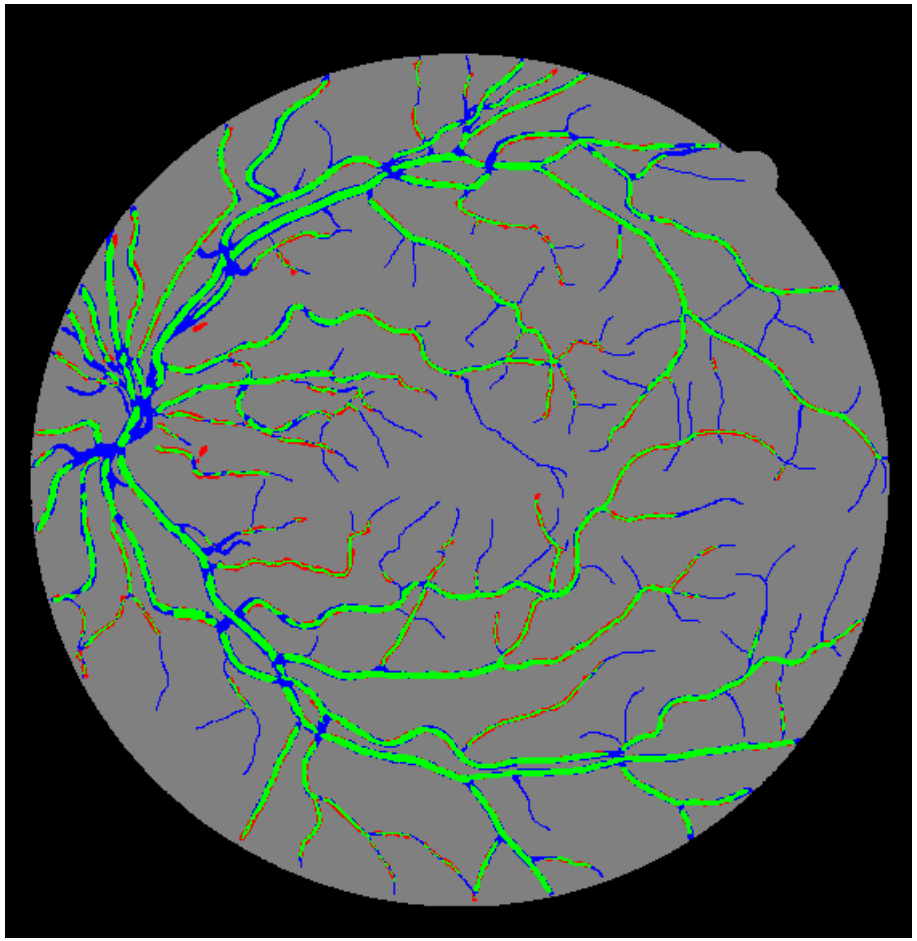


Segmentation

What region to evaluate over?



Combination of reference and result



masked

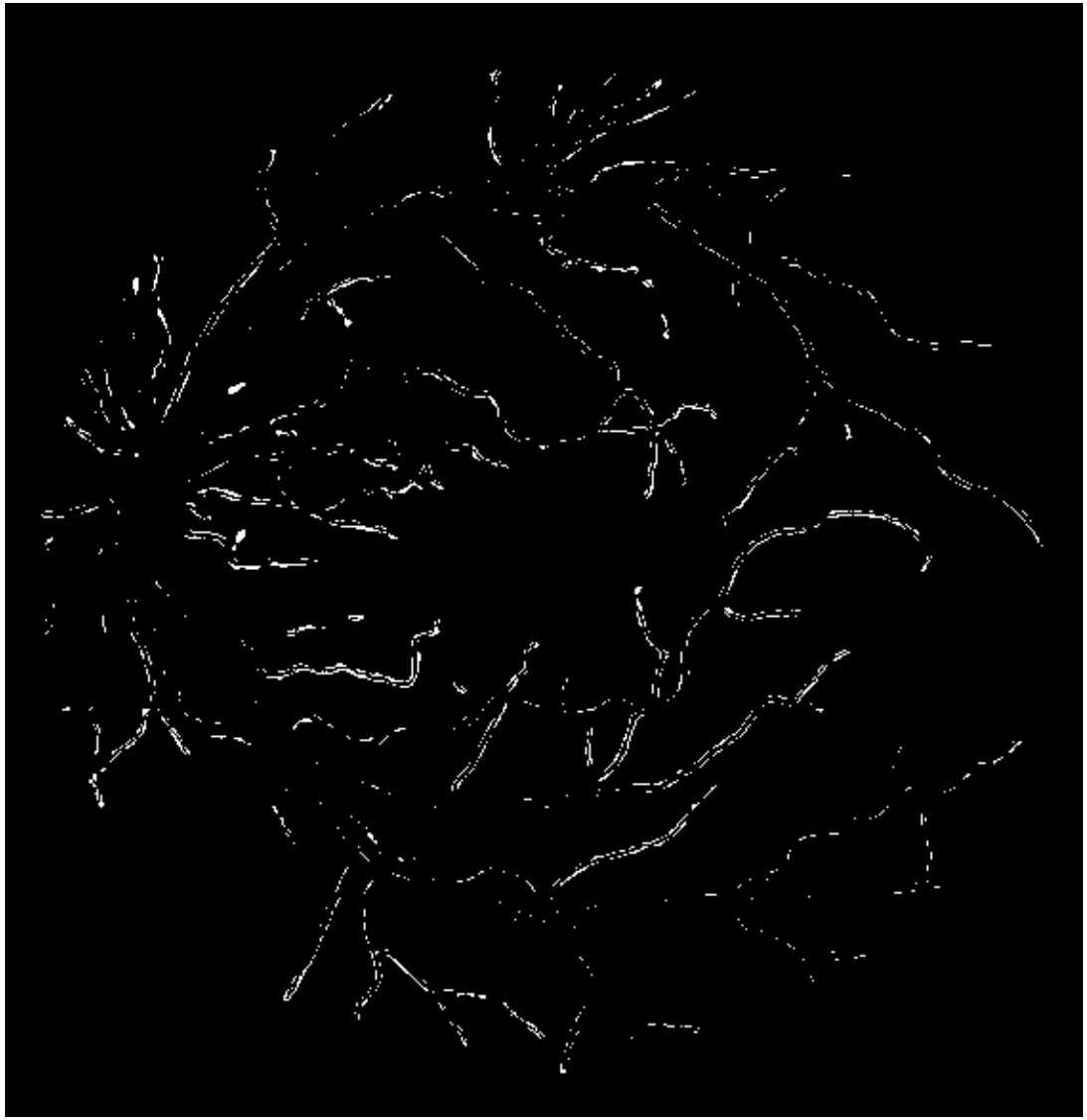
true positive

true negative

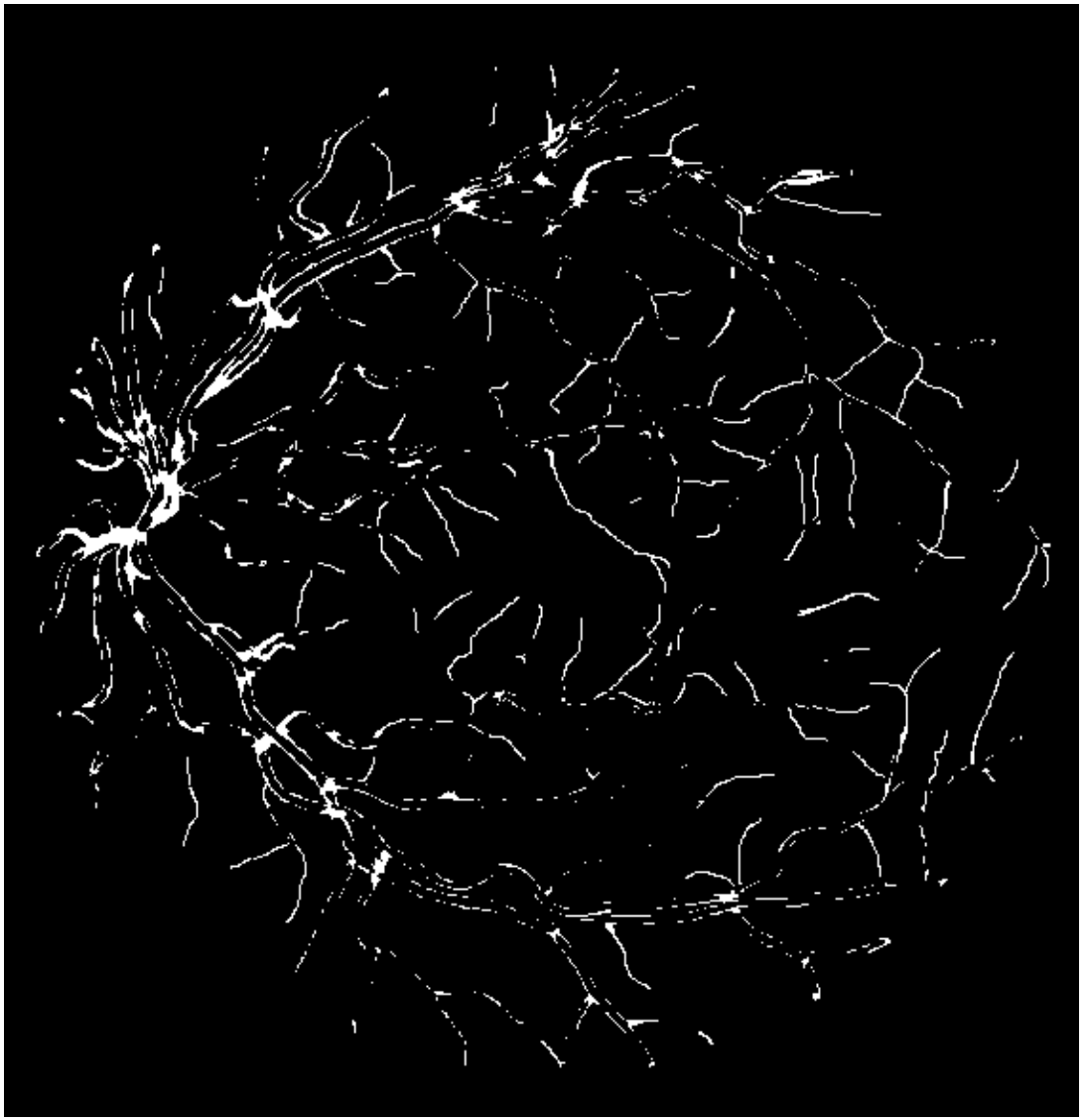
false negative

false positive

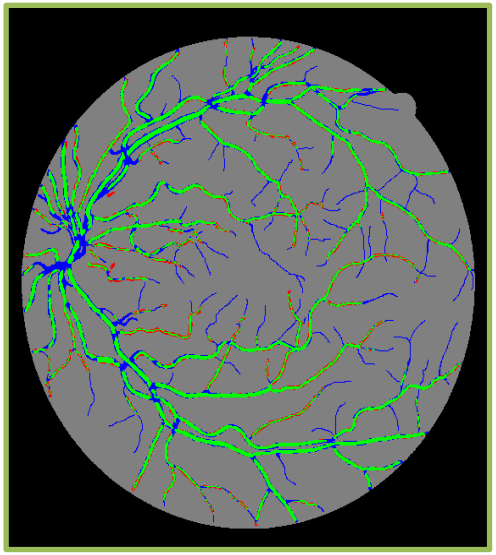
False positives



False negatives



Confusion matrix (Contingency table)



Segmentation

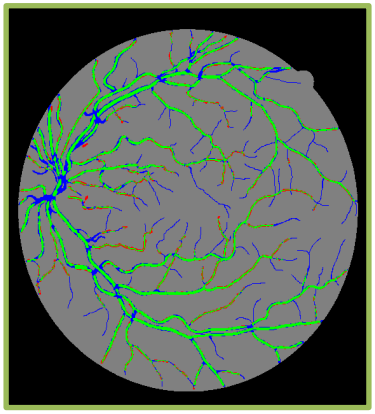
Reference

	negative	positive
negative	191152 **TN**	3813 **FP**
positive	9764 **FN**	19648 **TP**

Do not get confused!

- False positives are actually negative
- False negatives are actually positives

Confusion matrix (Contingency table)



Segmentation

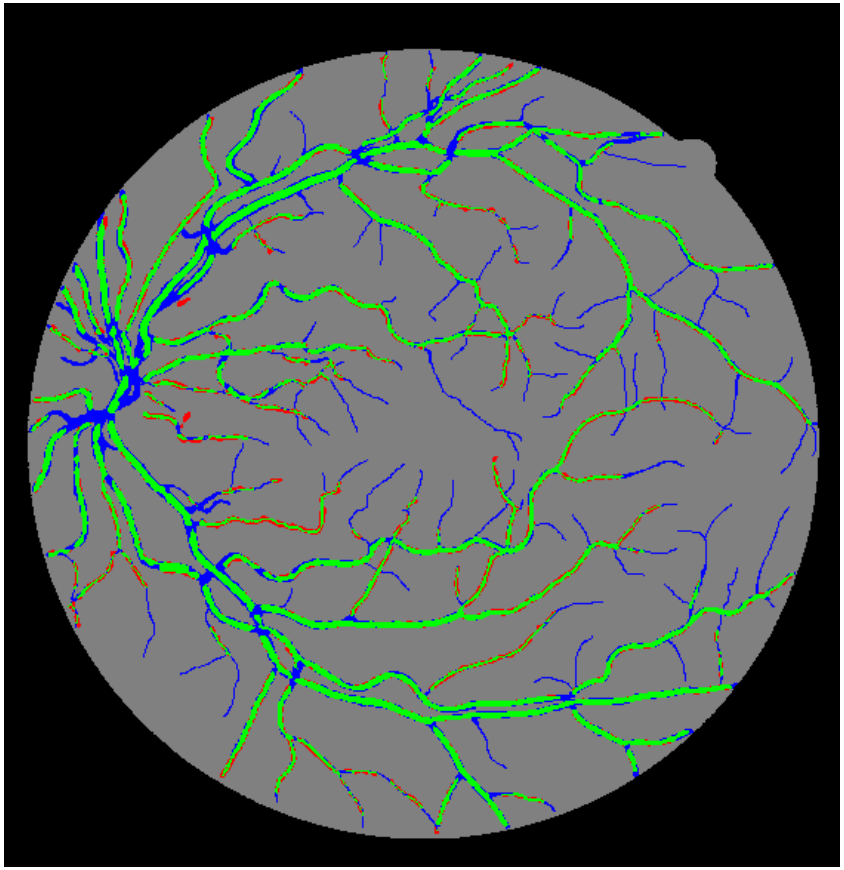
Reference

	negative	positive
negative	.852 **TN**	.017 **FP**
positive	.044 **FN**	.088 **TP**

Back to the retinal image…

result / reference	negative	positive
negative	.852 TN	.017 FP
positive	.044 FN	.088 TP

Accuracy: 0.93949
Sensitivity: 0.668027
Specifity: 0.980443



Kappa

- Accuracy would not be zero if we used a system that is ‘guessing’
- A ‘guessing’ system should get a ‘zero’ mark (remember multiple choice exams…)
- **Kappa** is an attempt to measure ‘**accuracy in excess of accuracy expected by chance**’

Kappa

- accguess = the accuracy of a randomly guessing system with a given positive (or negative) rate

kappa = (acc – accguess) / (1 – accguess)

Kappa

Result / Reference	negative	positive	
negative	191152	3813	194965
positive	9764	19648	29412
	200916	23461	224377

System accuracy:
(191152 + 19648)/
224377 = .939

Total number of positives

System positive rate:
23461/224377 = .105

System negative rate:
200916/224377=0.895

True positives of a guessing system:
.105 * 29412 = 3075

True Negative of a guessing system
0.895*194965= 174 579

Accuracy guessing (174579+3075)/224377
system: .792

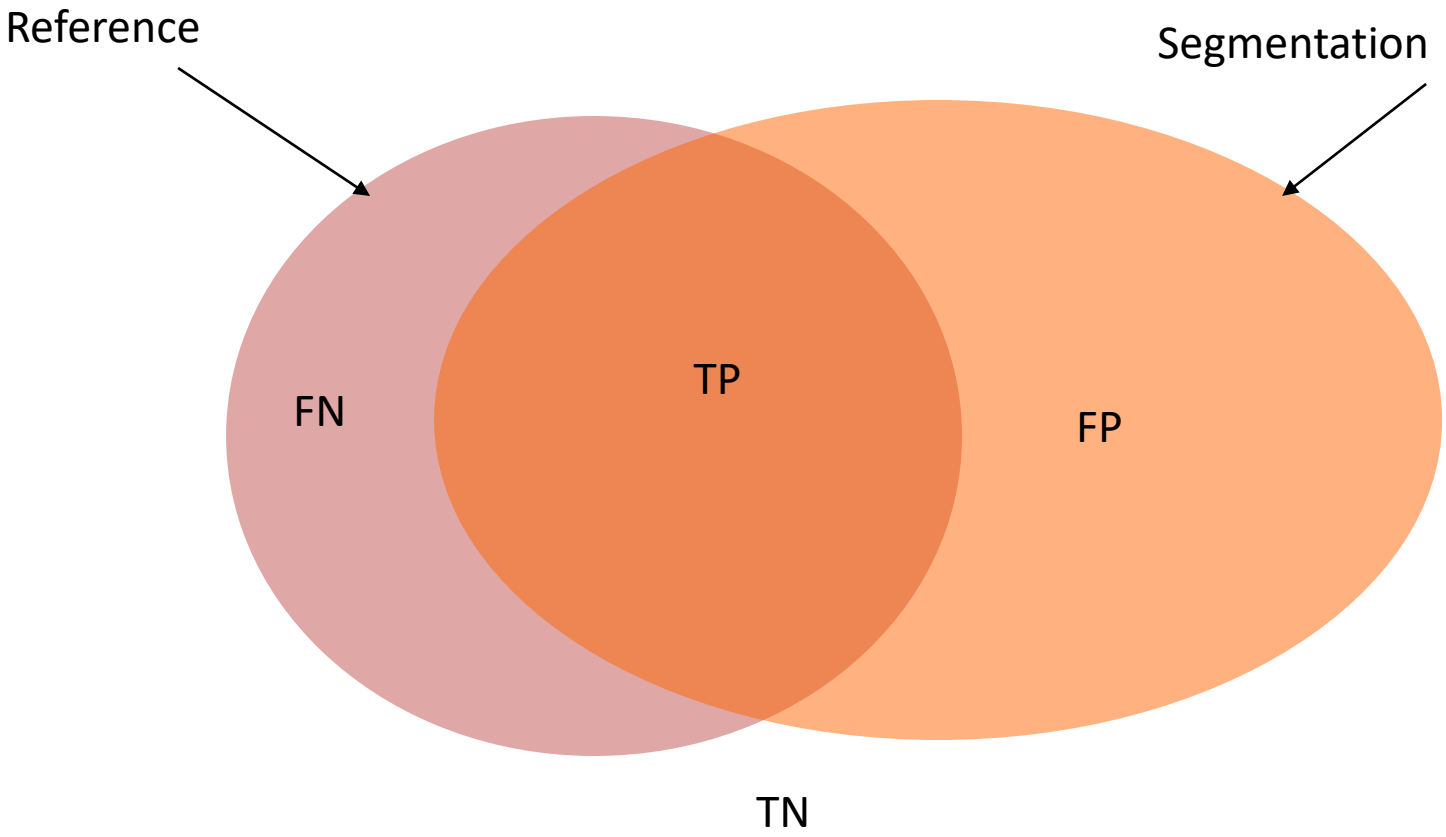
Kappa

- accguess = the accuracy of a randomly guessing system with a given positive (or negative) rate
- kappa = (acc – accguess) / (1 – accguess)
- In our case: kappa = (.939 - .792)/(1 - .792) = .707

Kappa

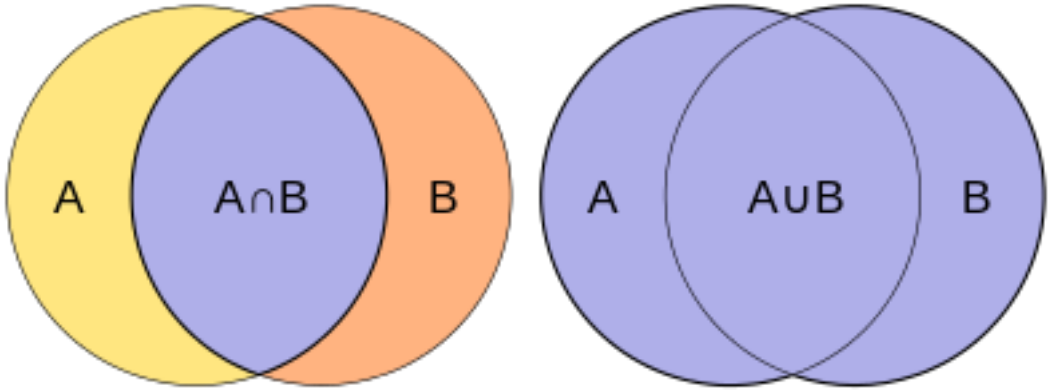
- Maximum value is 1, can be negative
- A ‘guessing’ system has kappa = 0
- ‘Stupid systems’ (‘all background’ or ‘most likely class’) have kappa = 0
- Systems with negative kappa have ‘worse than chance’ performance

Overlap =IoU= intersection / union = TP/(TP+FP+FN)



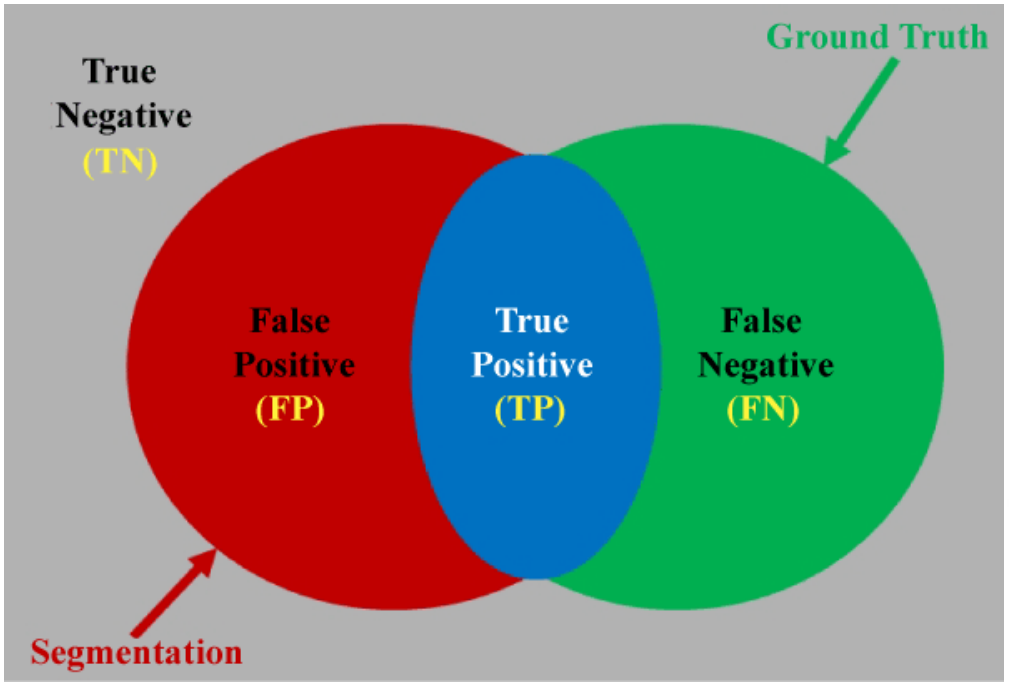
Overlap

- Overlap ranges from 0 (no overlap) to 1 (complete overlap)
- The background (TN) is disregarded in the overlap measure
- Small objects with irregular borders have lower overlap values than big compact objects



Jaccard Index or Tanimoto Coefficient or IoU

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$



Dice Index

$$Dice = \frac{2 \times TP}{(TP + FP) + (TP + FN)}$$

Distances and norms

- There are various ways to measure distances between two samples
- Particularly important in high dimensions (because of curse of dimensionality)
- In pattern recognition the two most useful are
  - Euclidean
  - Mahalanobis
- Various other are available
  - Standardized Euclidean
  - City block
  - Minkowski
  - Hamming
  - And others...

Distances and norms

- L_1-norm: sum of absolute values

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{d} |x_i|$$

- L_2-norm: sum of squared values

$$\|\boldsymbol{x}\|_2 = \left(\sum_{i=1}^{d} x_i^2 \right)^{\frac{1}{2}}$$

- L_∞-norm: maximum norm

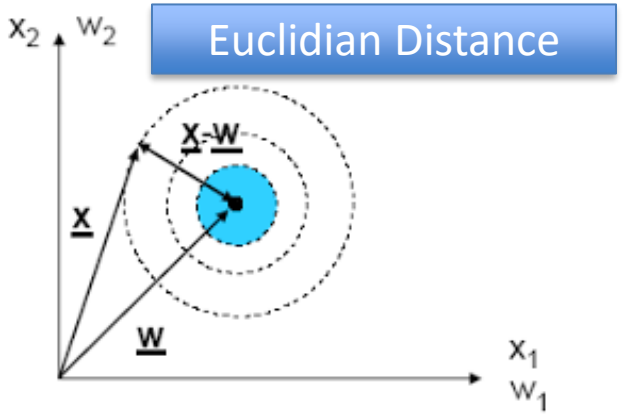
$$\|\boldsymbol{x}\|_\infty = \lim_{p \to \infty} \left(\sum_{i=1}^{d} |x_i|^p \right)^{\frac{1}{p}} = \max_i \{ |x_i| \; ; \; i = 1, 2, \ldots, d \}$$

Euclidean distance

- The most widely used
- A convenient way to calculate it:

$$d_{euci} = \sqrt{(x_1 - x_{ref})^t (x_1 - x_{ref})} \qquad x_1 = \begin{bmatrix} 3 \\ 4 \\ 10 \end{bmatrix} \qquad x_{ref} = \begin{bmatrix} 6 \\ 8 \\ 10 \end{bmatrix}$$

- It considers between two samples only

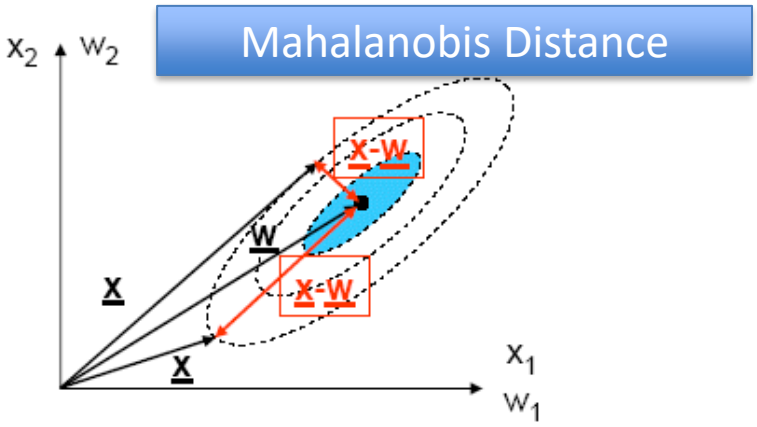


Mahalanobis distance

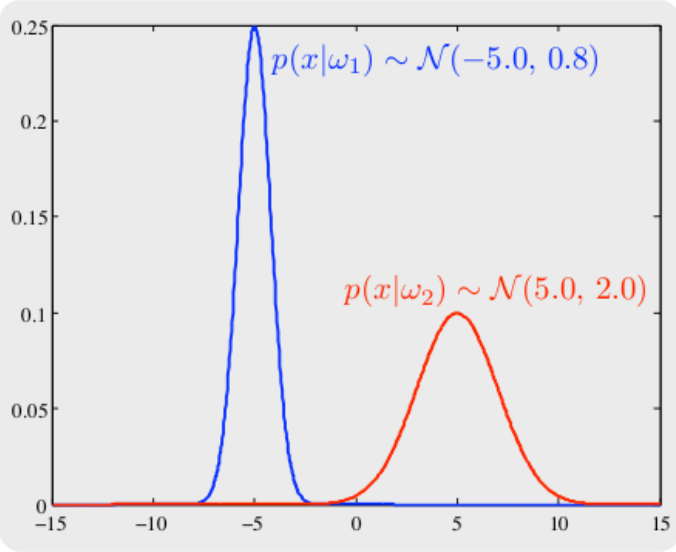
- Calculate the distance of a sample to a Gaussian distribution
- A convenient way to calculate it:

$$d_{maha} = \sqrt{(x_1 - x_2)^t \Sigma^{-1} (x_1 - x_2)} \qquad x_1 = \begin{bmatrix} 3 \\ 4 \\ 10 \end{bmatrix} \qquad x_2 = \begin{bmatrix} 6 \\ 8 \\ 10 \end{bmatrix}$$

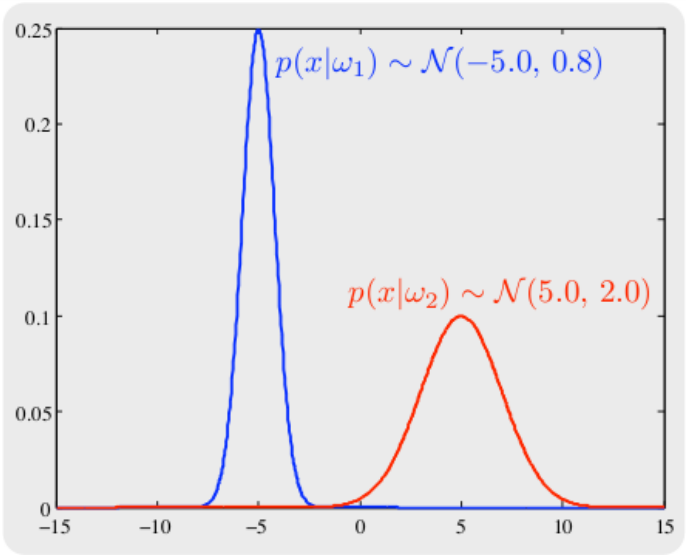
- $\Sigma \rightarrow$ Covariance matrix of the distribution



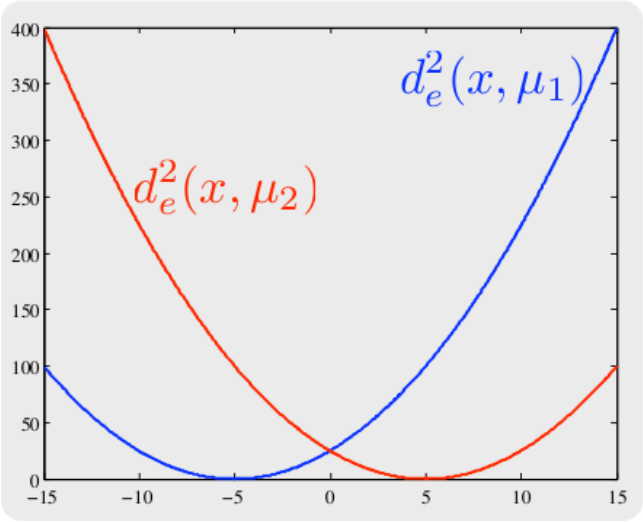
Distance comparison



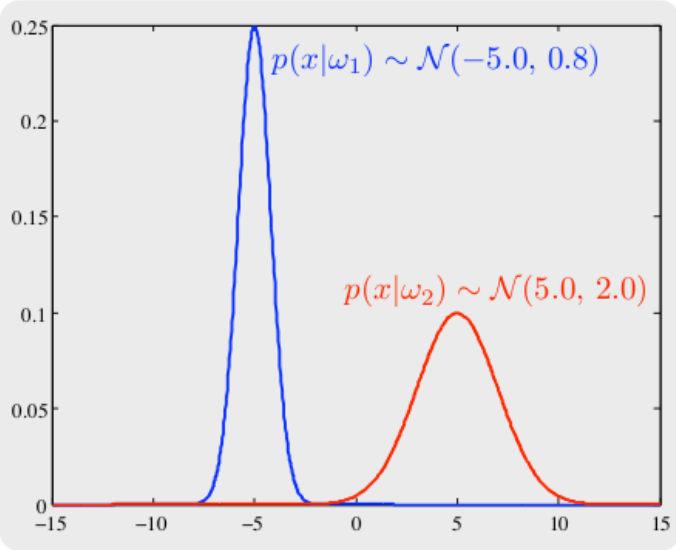
Distance comparison



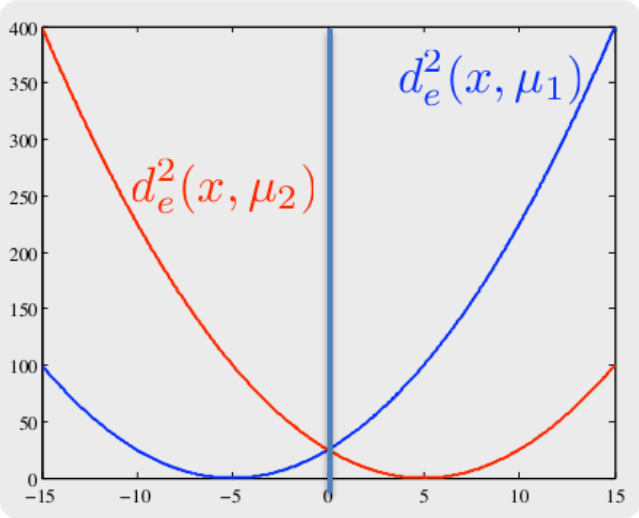
Euclidean



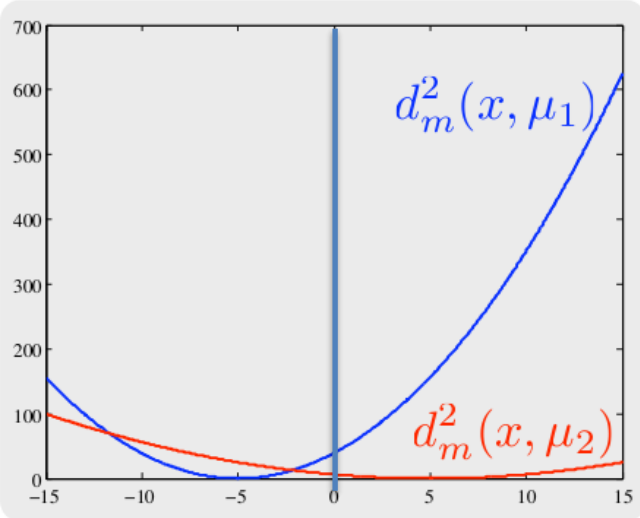
Distance comparison



Euclidean

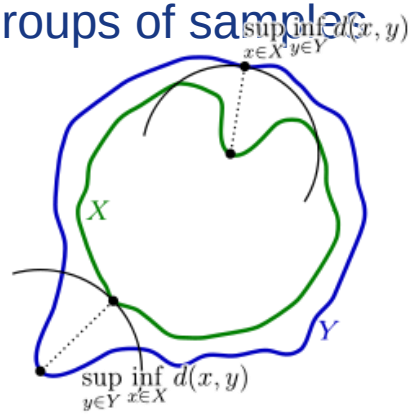


Mahalanobis



Other Distances → Hausdorff

- So far only a sample has been used as reference
- It is possible to evaluate a distances between groups of samples (without knowing the distributions!)
- Hausdorff distance
  - *X*, *Y* → two sets of samples
  - *tmpLst* → buffer
  - For each *x* (belonging to *X*)
    - Find minimum euclidean distance to Y
    - add minimum euclidean distance to Y to *TmpLst*
  - End for
  - *leftHausdorff* := max of *TmpLst*
  - Calculate *rightHausdorff* inverting X and Y
  - *Hausdorff := max* between *leftHausdorff* and *rightHausdorff*



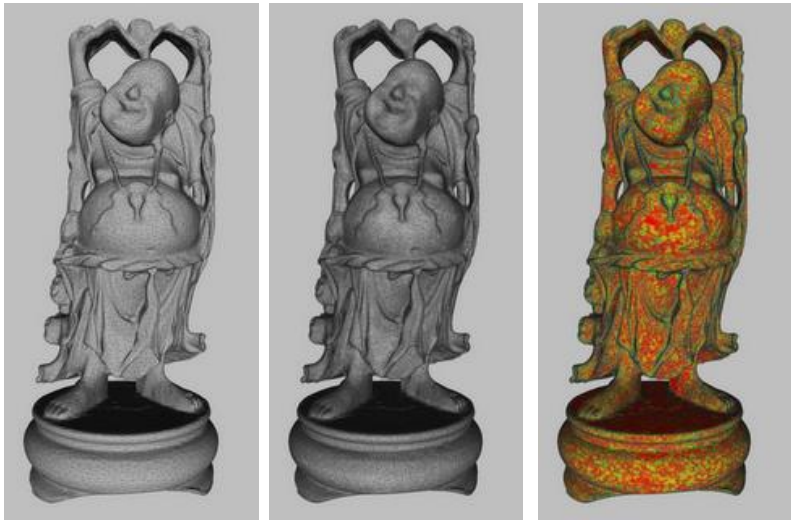
https://en.wikipedia.org/wiki/Hausdorff_distance

Hausdorff Distance

- More formally:

$$d_{\mathrm{H}}(X,Y) = \max\{\sup_{x\in X}\inf_{y\in Y} d(x,y),\ \sup_{y\in Y}\inf_{x\in X} d(x,y)\},$$

- Rarely employed for classification
- But often employed to evaluate:
  - Segmentations
  - 3D reconstructions



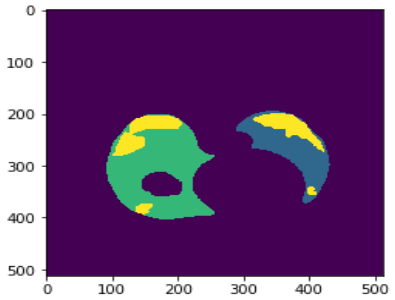
Performance matrices Segmentation

Validation or testing the performance of model for segmentation

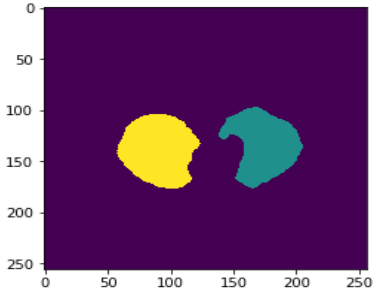
Performance Metrics:

1. $Sensitivity = TPR = \frac{TP}{TP+FN}$

2. $Specificity = TNR = \frac{TN}{TN + FP}$



GT(A)



Predicted(B)

3. Jaccard index

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

where A is ground-truth mask and B is predicted segmentation map)

4. Dice Coefficients (DC)

$$Dice(A,B) = \frac{2|A \cap B|}{|A| + |B|}$$

where A is ground-truth mask and B is predicted segmentation map)

Performance matrices Segmentation

Validation or testing the performance of model for segmentation

Performance Metrics:

5. Volume Overlap Error (VOE)

$$VOE(A,B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

6. Relative Volume Difference (RVD)

$$RVD(A,B) = \frac{|B| - |A|}{|A|}$$

6. Surface Distance Metrics

Let S(A) denotes the set of surface voxels of A and the shortest distance of an arbitrary voxel v to S(A) is shown in equation

$$d(v, S(A)) = min_{S_A \epsilon S(A)} \|v - S_A\|$$

$$ASD(A,B) = \frac{1}{|S(A)| + |S(B)|}\left(\sum_{S_A \epsilon S(A)} d(S_A, S(B)) + \sum_{S_B \epsilon S(B)} d(S_B, S(A))\right)$$

7. Hausdorff Distance

Symmetric Hausdorff Distance is used to compute the (symmetric) Hausdorff Distance (HD) between the binary objects in two segmentation masks. It is defined as the maximum surface distance (MSD) between the objects.

$$MSD(A,B)$$
$$= max\{max_{S_A \epsilon S(A)} d(S_A, S(B)), max_{S_B \epsilon S(B)} d(S_B, S(A))\}$$

Performance matrices Segmentation

Validation or testing the performance of model for segmentation

Figure 1, for two point sets X and Y, the one-sided HD from X to Y is defined as [7]:

$$\mathrm{hd}(X,Y) = \max_{x\in X} \min_{y\in Y} \|x-y\|_2, \qquad (1)$$

and similarly for $\mathrm{hd}(Y,X)$:

$$\mathrm{hd}(Y,X) = \max_{y\in Y} \min_{x\in X} \|x-y\|_2. \qquad (2)$$

The bidirectional HD between these two sets is then:

$$\mathrm{HD}(X,Y) = \max\big(\mathrm{hd}(X,Y), \mathrm{hd}(Y,X)\big) \qquad (3)$$

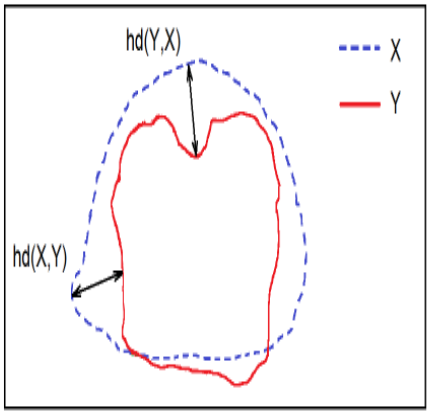


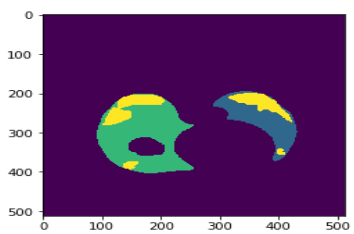
Figure 1. A schematic showing the Hausdorff Distance between points sets X and Y.

Performance matrices Segmentation

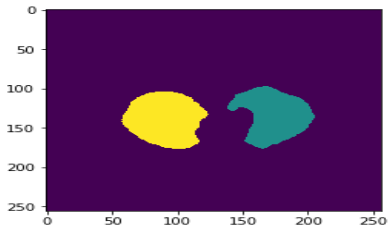
Validation or testing the performance of model for segmentation

1. *Sensitivity*
2. Specificity
3. Jaccard Index
4. Dice similarity Coefficients

Should be high



Manual Mask(GT)



Predicted by model

1. Volume Overlap Error
2. Relative Volume Difference
3. Surface Metrics(Asymmetric distance(ASD))
4. Hausdorff Distance(HD)

Distance Surface

Should be low

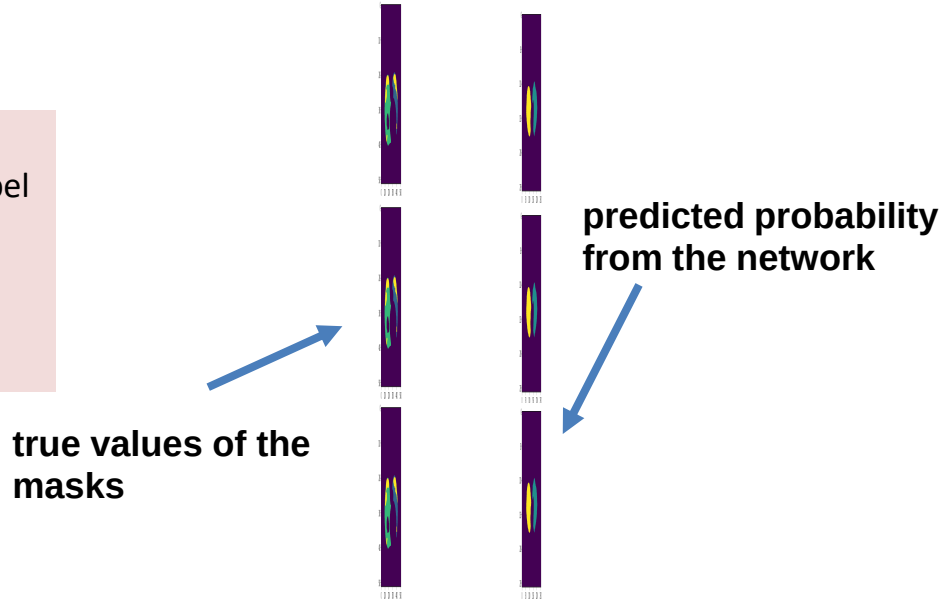
Performance matrices Segmentation

ROC computation for segmentation:

Each .txt file contain the total images after flattening where “true_xx.txt” contains the true values of the masks [0,1] and “predict_xx.txt” contains the predicted probability from the network. The dimension of the contents in a .txt file is "Image_Number x Rows x Columns"

```
fpr_fcn, tpr_fcn, thresholds_model =
roc_curve(np.loadtxt('true_labels.txt'),np.loadtxt('predict_label
s.txt'))

auc_model = roc_auc_score(np.loadtxt('true_labels.txt'),
                np.loadtxt('predict_labels.txt'))
```



Performance matrices Segmentation

https://stackoverflow.com/questions/57830697/calculating-the-area-covered-by-the-objects-of-irregular-shapes-in-an-image

Surface area:
Compute the area of non-zero pixels and divides with total No of image height and width
Surface area=count(non_zero_pixels)/(width*height)


```
import cv2
import numpy as np
img = cv2.imread('img.jpg')
height = img.shape[0]
width = img.shape[1]
gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(gray,0,255,cv2.THRESH_BINARY)
cv2.imshow("Mask", thresh)
cv2.waitKey(0)
cv2.destroyAllWindows()
count = cv2.countNonZero(thresh)
area = count/(width*height)
print(area)
```

Performance matrices Segmentation

Surface area:
Compute the area of non-zero pixels and divides with total
No of image height and width
Surface area=count(non_zero_pixels)/(width*height)

calculate surface area of all test predicted mask produced by model
And segmentation mask produced manually

Two surface area vector obtained (predicted segmentation masks area, Gt segmentation masks area)

Predictedarea=1,…….n, where n is the length of test images

Gtarea=1,…….n, where n is the length of test images

We can used these two vectors in order to compute t-test, p-test or band-Altman plot between two surface area vectors