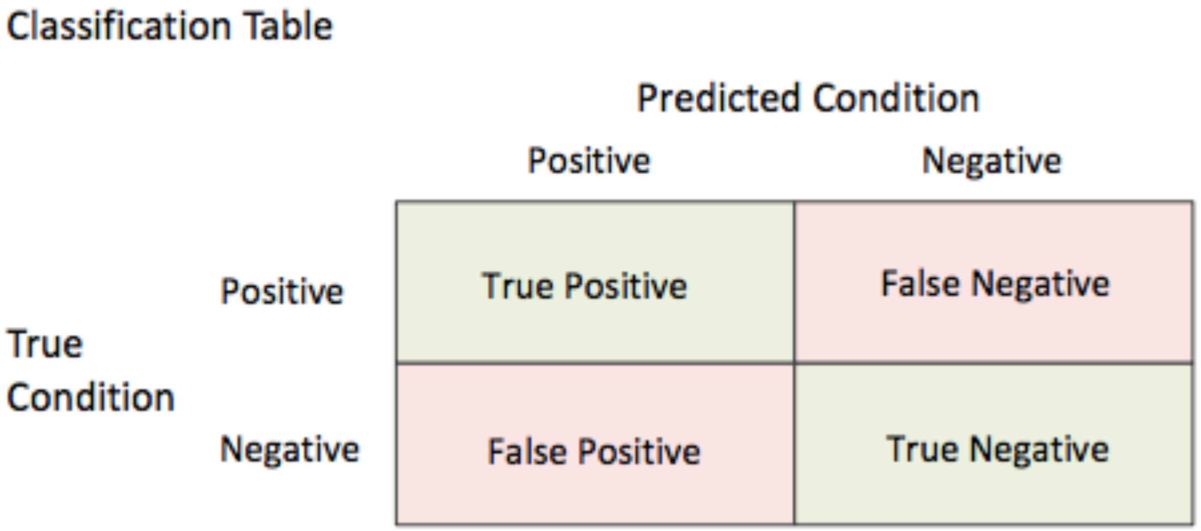
Lect. 7: Model Assessment & Ensemble methods

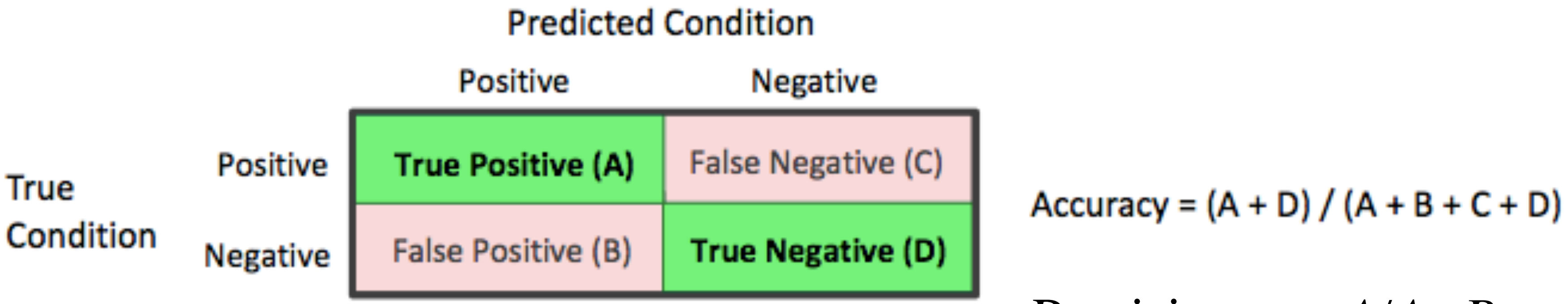
Rida Moustafa

Model Assessment



Classification Table

		Predicted Condition	
		Positive	Negative
True Condition	Positive	True Positive	False Negative
	Negative	False Positive	True Negative



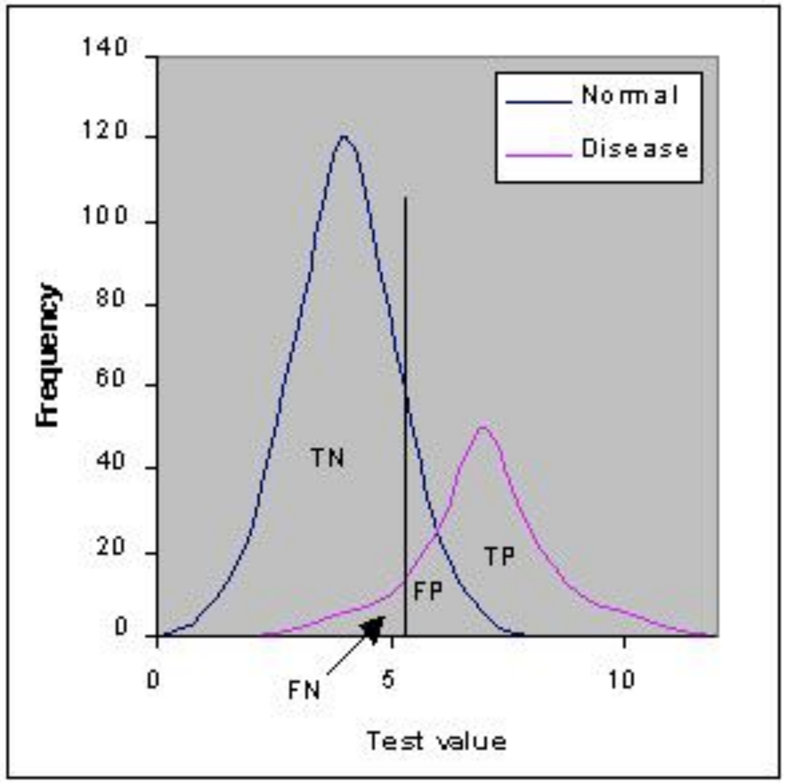
		Predicted Condition	
		Positive	Negative
True Condition	Positive	**True Positive (A)**	False Negative (C)
	Negative	False Positive (B)	**True Negative (D)**

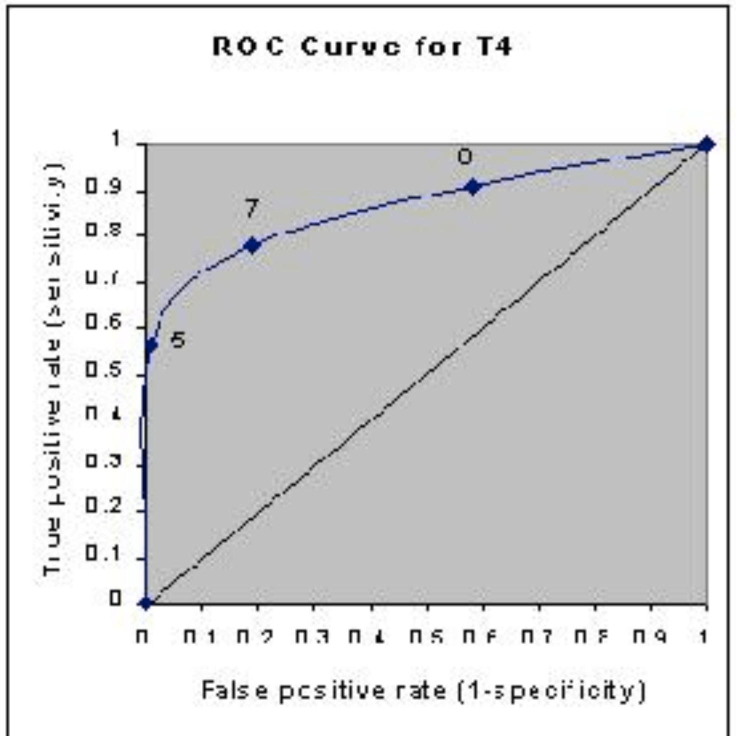
Accuracy = (A + D) / (A + B + C + D)

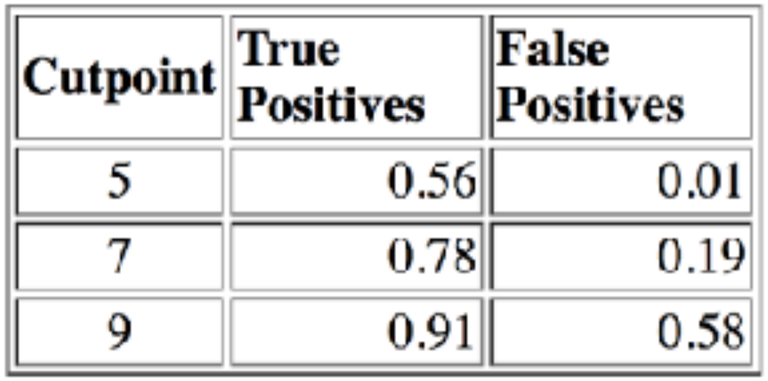
Precision, $p=A/A+B$

Recall, $r=A/A+C$

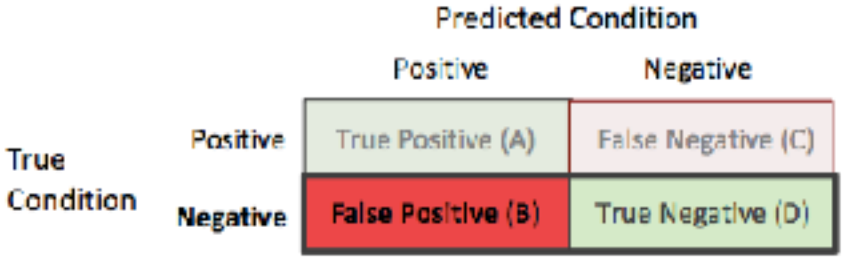
Roc Curve







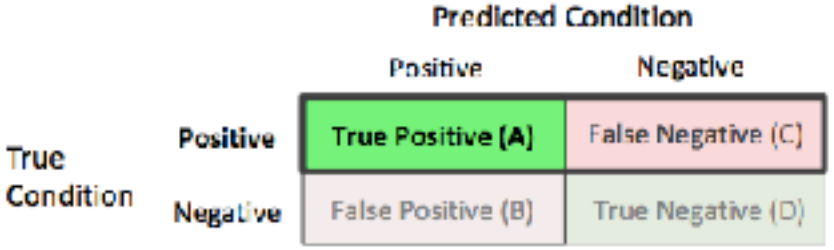
Cutpoint	True Positives	False Positives
5	0.56	0.01
7	0.78	0.19
9	0.91	0.58



		Predicted Condition	
		Positive	Negative
True Condition	Positive	True Positive (A)	False Negative (C)
	Negative	False Positive (B)	True Negative (D)

FPR = B / (B + D)

(X Axis on ROC Curve)

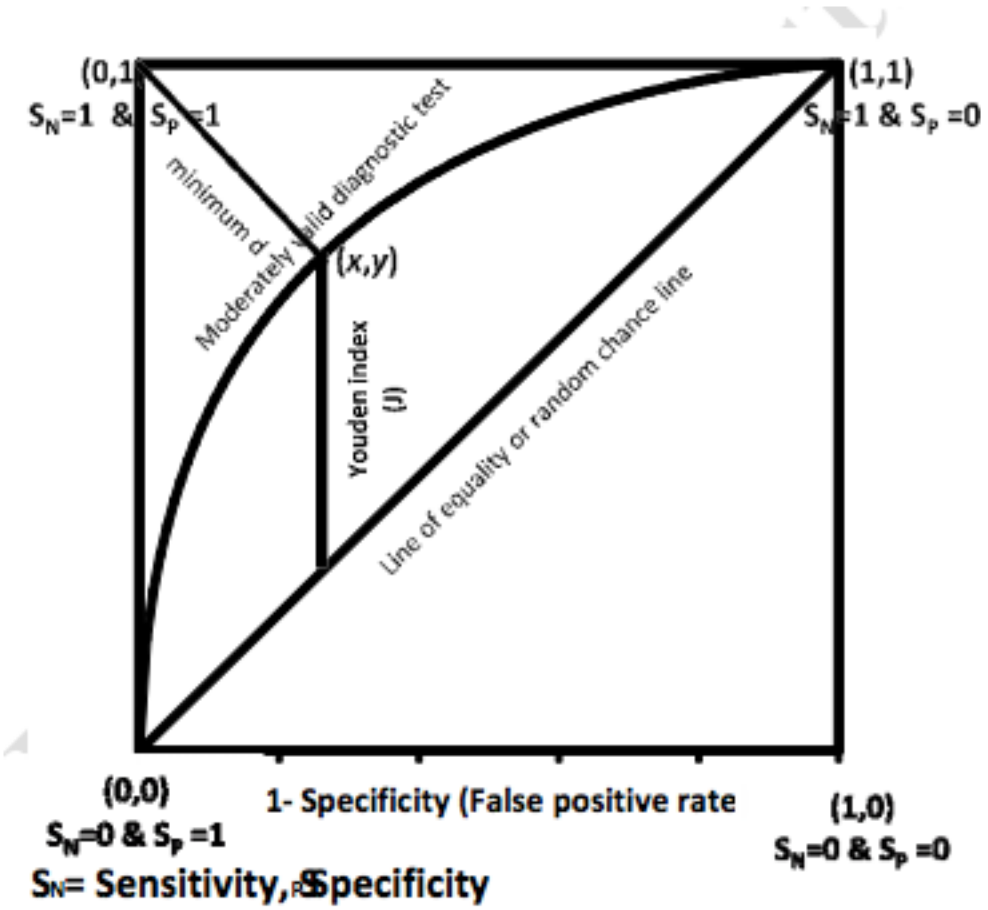


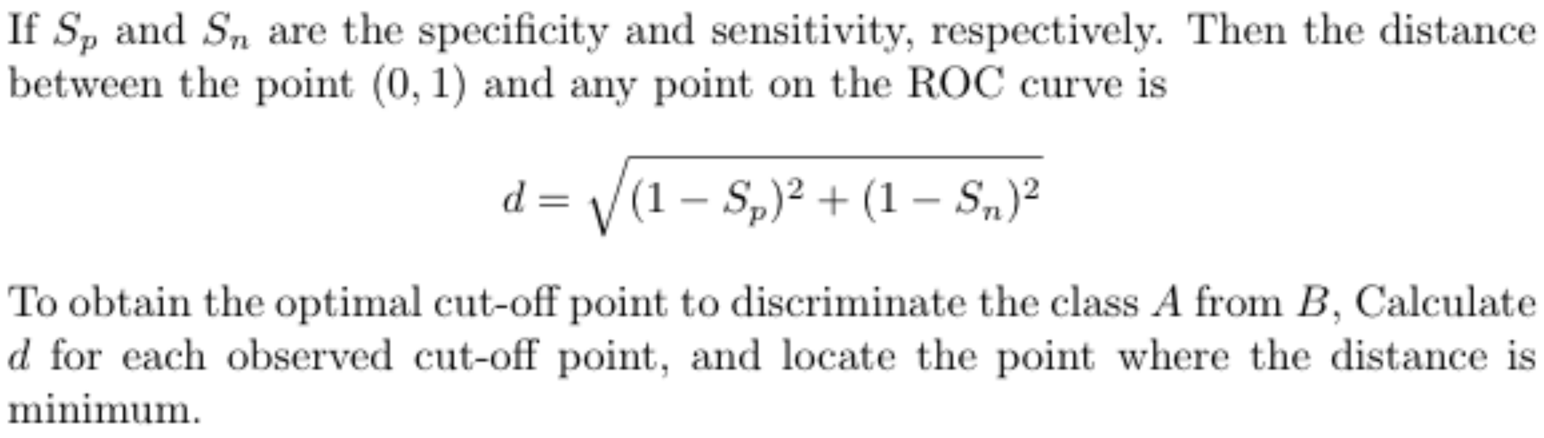
		Predicted Condition	
		Positive	Negative
True Condition	Positive	True Positive (A)	False Negative (C)
	Negative	False Positive (B)	True Negative (D)

TPR = A / (A + C)

(Y Axis on ROC Curve)

Find the best cut from ROC

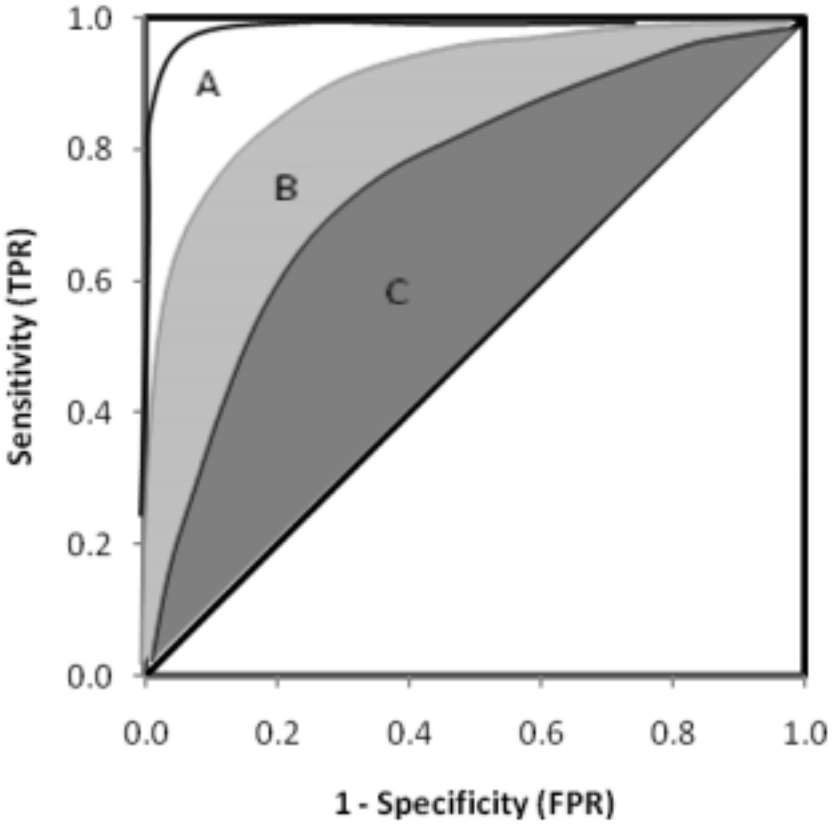

(0,1)
S_N=1 & S_P=1
minimum d
Moderately valid diagnostic test
(x,y)
Youden index (J)
Line of equality or random chance line
(1,1)
S_N=1 & S_P =0
(0,0)
S_N=0 & S_P =1
1- Specificity (False positive rate
(1,0)
S_N=0 & S_P =0
S_N= Sensitivity, S_P Specificity


If S_p and S_n are the specificity and sensitivity, respectively. Then the distance between the point $(0, 1)$ and any point on the ROC curve is

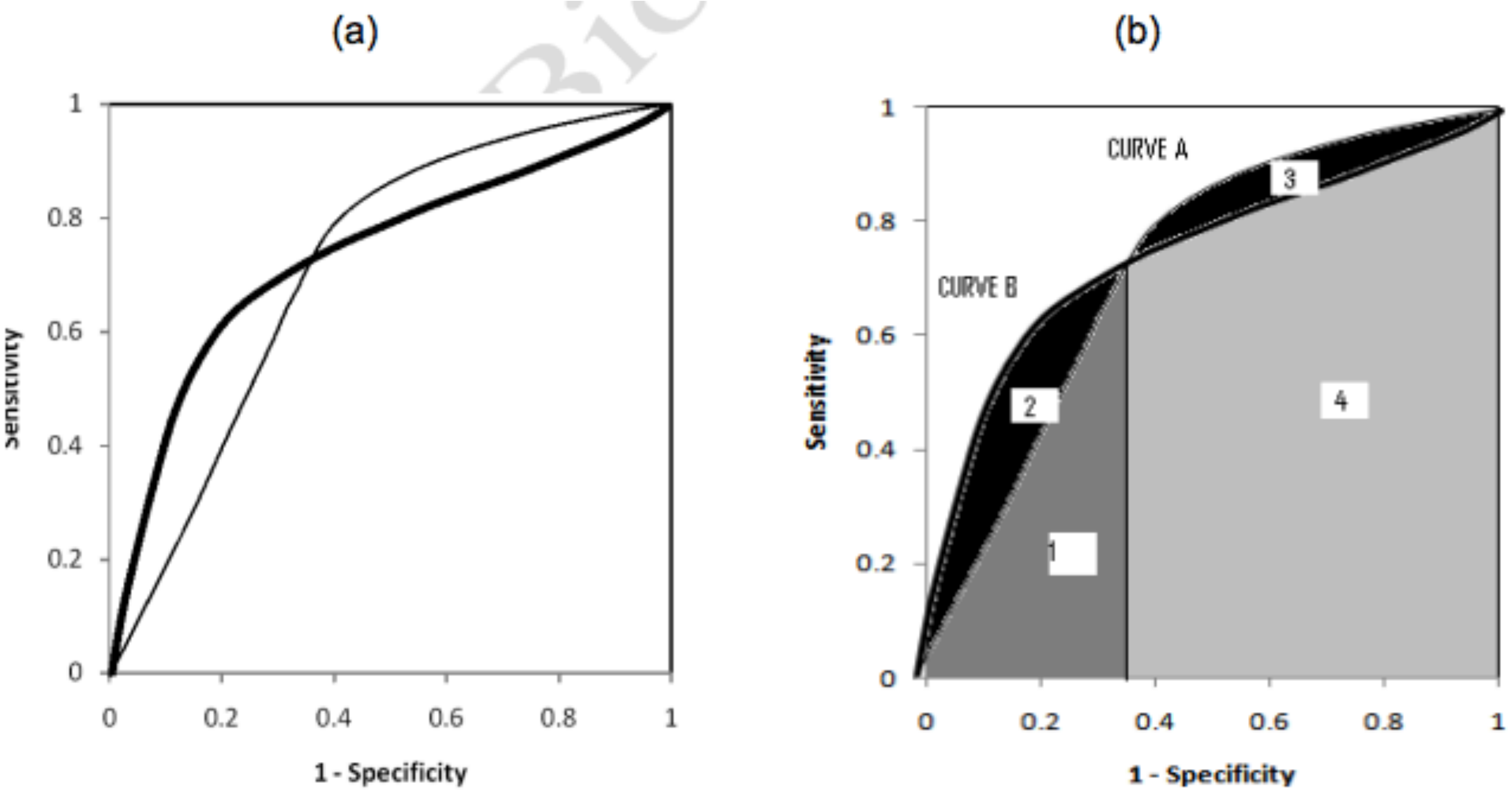
$$d = \sqrt{(1 - S_p)^2 + (1 - S_n)^2}$$

To obtain the optimal cut-off point to discriminate the class A from B, Calculate d for each observed cut-off point, and locate the point where the distance is minimum.

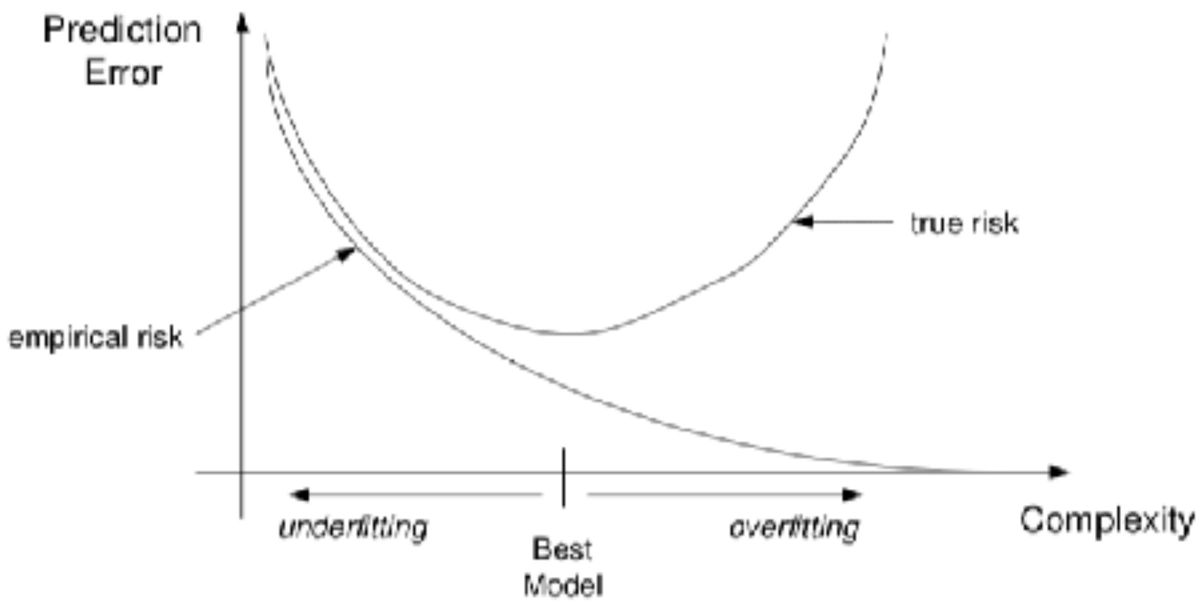
Area under the curve (AUC)



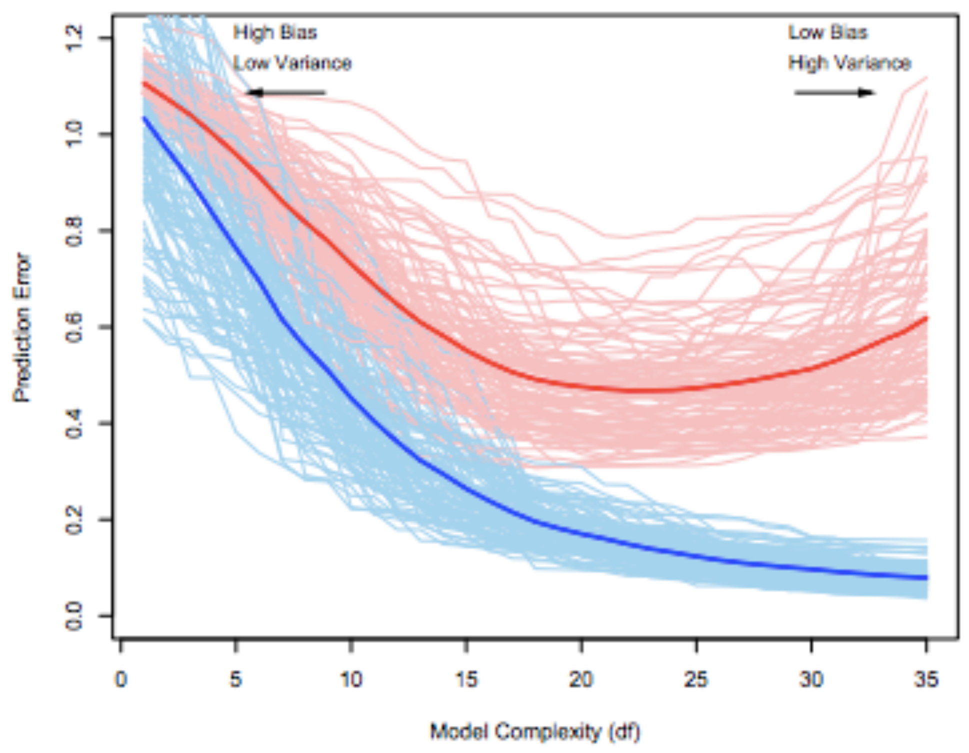
Comparing classifiers using ROC



Bias-Variance Tradeoff: fighting overfitting



What happened for the testing error and the training error for different complexity and different number of points?
Why?



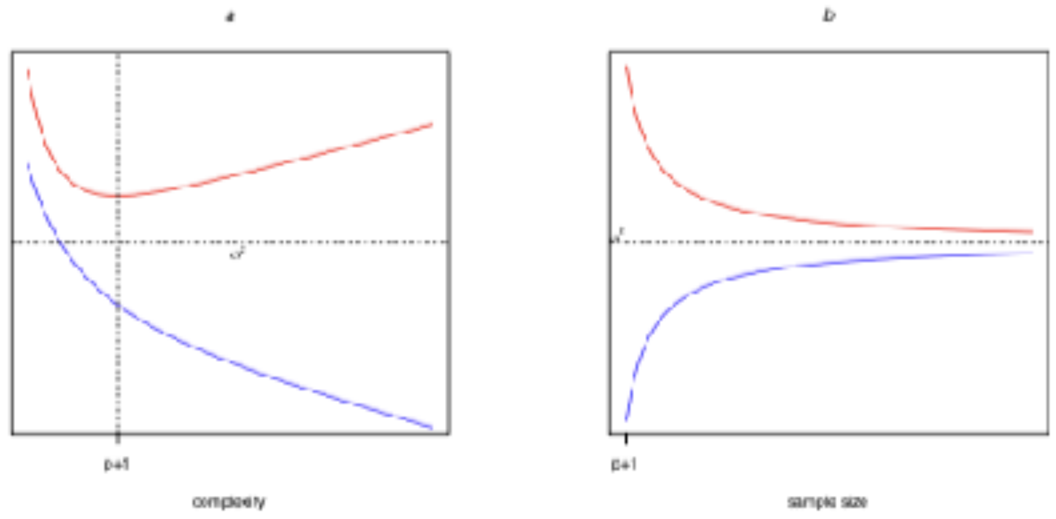
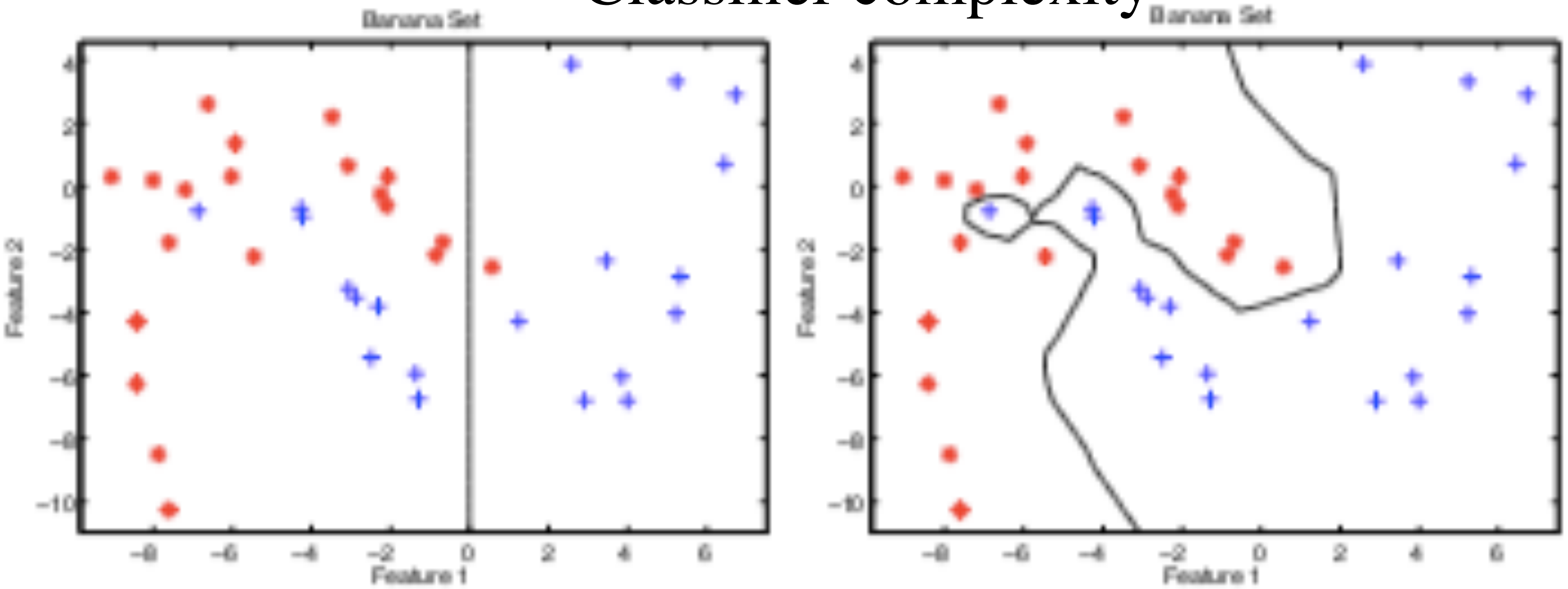


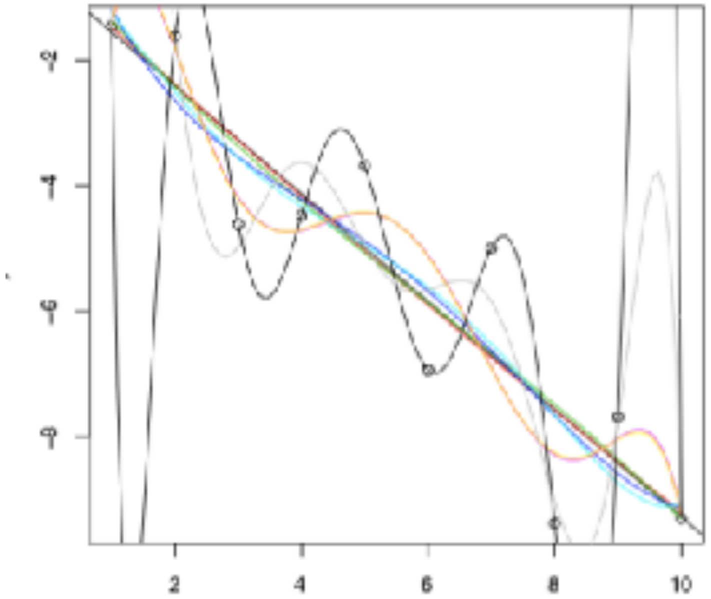
Figure 5: The *mse* on the training and testing data for ols as a function of the complexity p (a) and a function of sample size n (b). We assume the training and testing samples have the same sizes. This graph also represents the low complexity.

Bias-Variance Tradeoff: fighting overfitting

Classifier complexity



Overfitting



The complexity of a classifier indicates the ability to fit to any data distribution

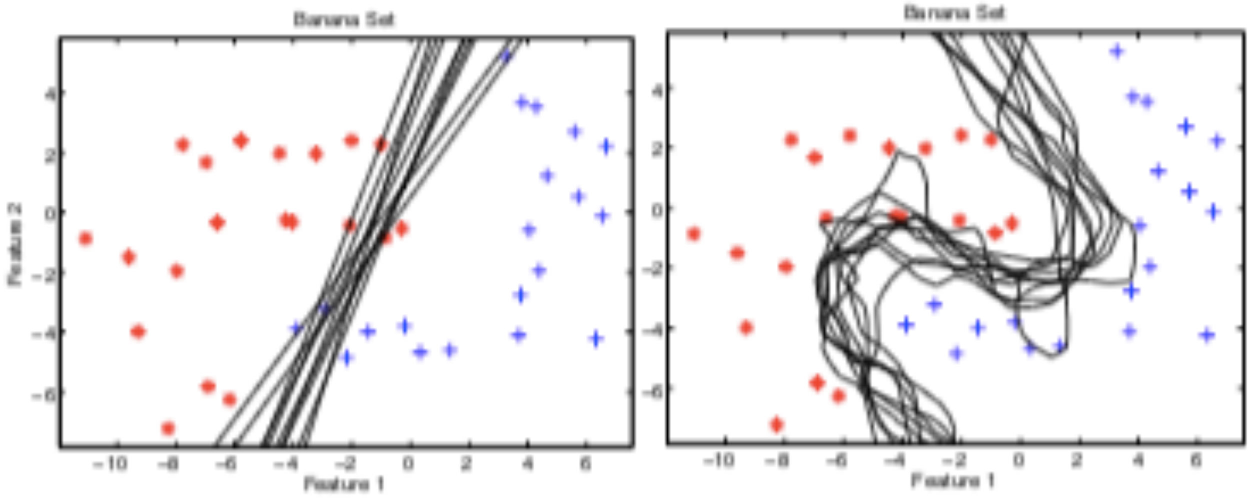
Bias-Variance Tradeoff: fighting overfitting

- Assume, our function f tries to model the relation between x and y given dataset $\mathcal{X} = \{\mathbf{x}_i, y_i\}, i = 1 \ldots N$
- The mean-squared error can be decomposed:

$$\varepsilon = E_{\mathcal{X}}\left[(f_{\mathcal{X}}(\mathbf{x}) - y(\mathbf{x}))^2\right]$$

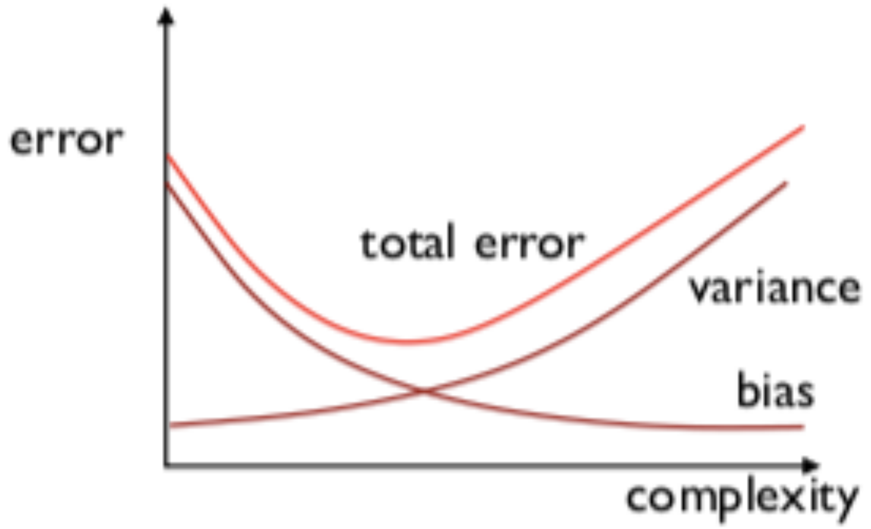
$$= (E_{\mathcal{X}}[f_{\mathcal{X}}(\mathbf{x}) - y(\mathbf{x})])^2 + E_{\mathcal{X}}\left[(f_{\mathcal{X}}(\mathbf{x}) - E_{\mathcal{X}}[y(\mathbf{x})])^2\right]$$

(squared) bias variance



- error = squared bias + variance
- This tradeoff is very general

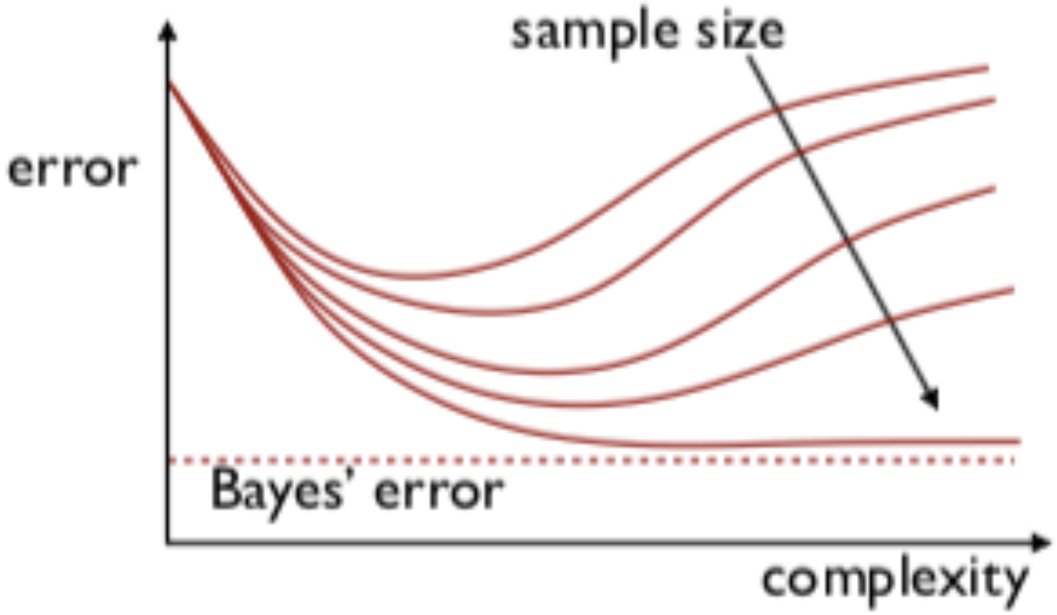
- More flexible models have lower bias, but higher variance
- Simple models have high bias, but low variance



Bias-variance explains the same peaking phenomenon

Bias-Variance Tradeoff: fighting overfitting

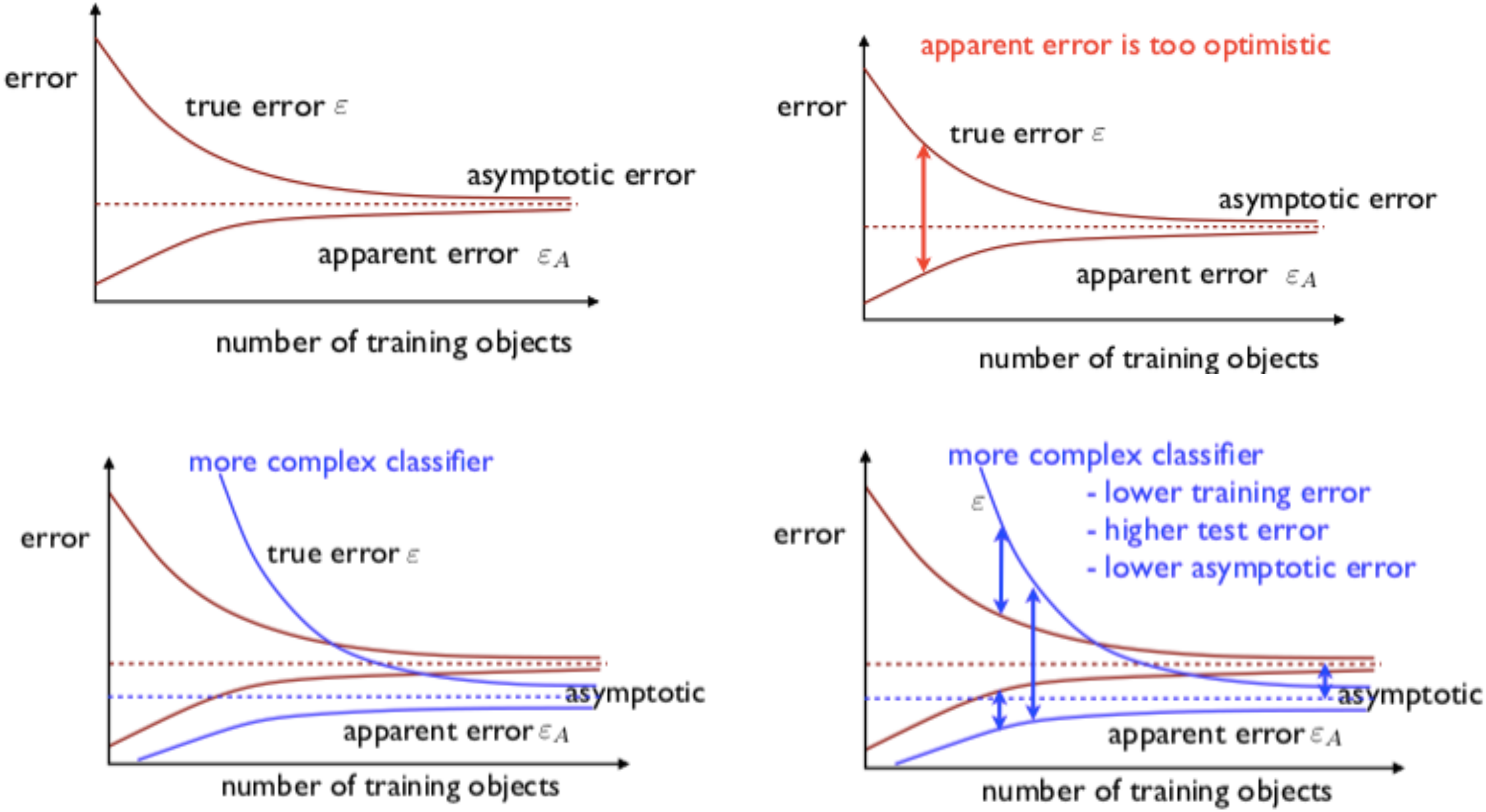
Peaking Phenomena



The minimum error for a given number of samples is obtained for a specific complexity

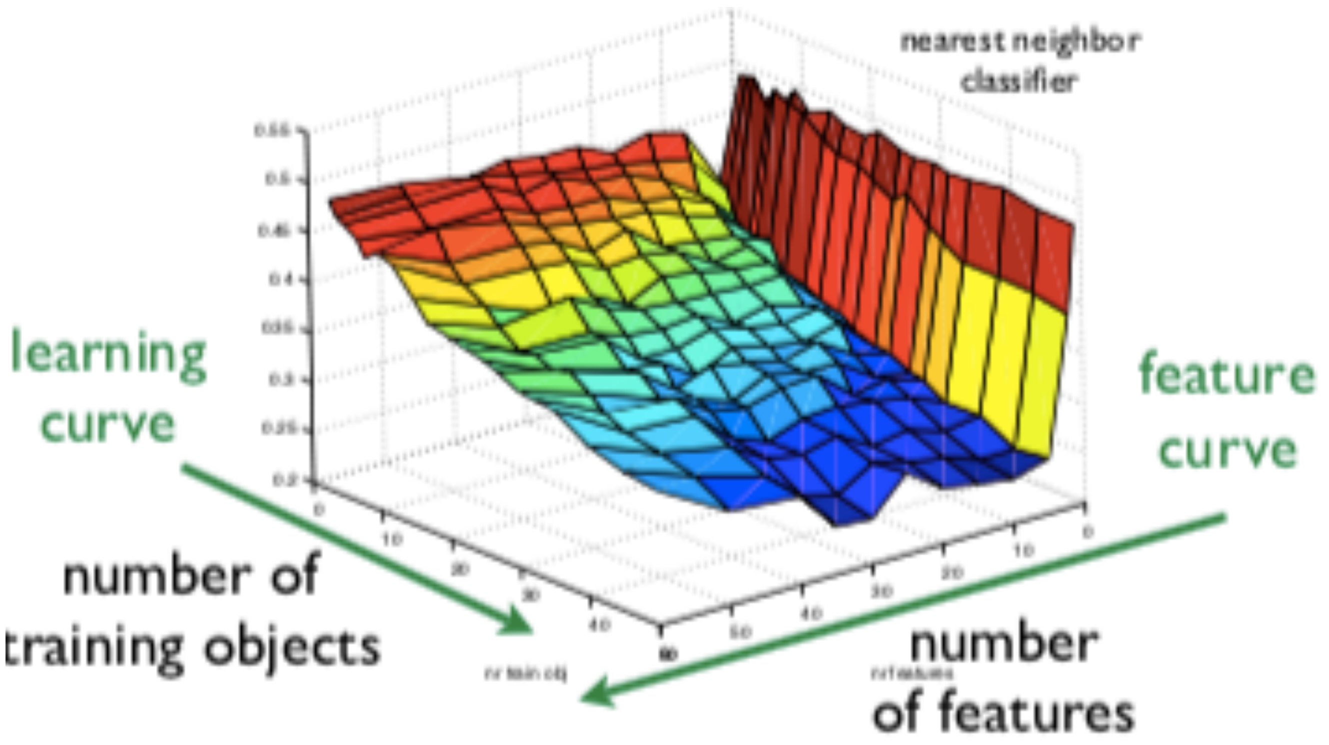
Bias-Variance Tradeoff: fighting overfitting

Learning curve and classifier complexity



Bias-Variance Tradeoff: fighting overfitting

Learning vs Feature Curve



So, what do you recommend?

- Complex classifiers are good when you have sufficient number of training objects
- When a small number of training objects is available, you overtrain
- Use a simple classifier when you don't have many training examples

Choose the complexity according to the available training set size

Fighting overfitting: Regularization/shrinkage methods

$$min_w \left\{ \sum_i (y_i - f(x_i, \beta))^2 + \lambda Pen(\beta) \right\}.$$

$Pen(\beta)$ is a penalty function that controls the model parameters β and λ is the regularization/shrinkage factor.

If $Pen(\beta) = \sum_j \beta_j^2$ then we have L_2 regularization.

If $Pen(\beta) = \sum_j |\beta_j|$ then we have L_1 regularization.

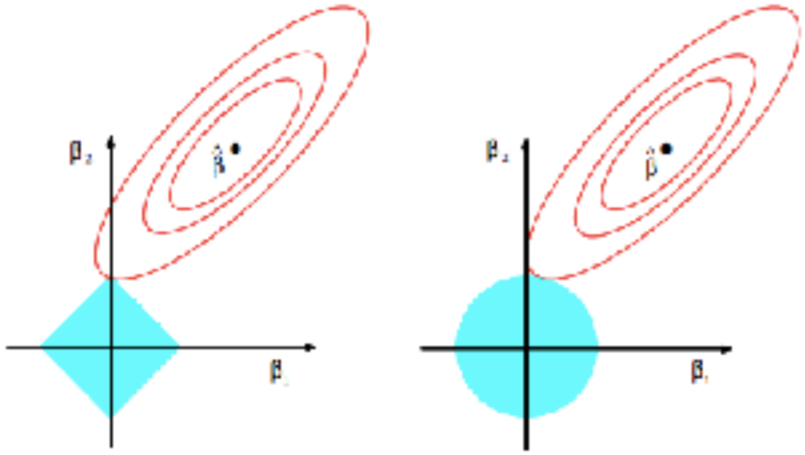


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \le t$ and $\beta_1^2 + \beta_2^2 \le t^2$, respectively, while the red ellipses are the contours of the least squares error function.*

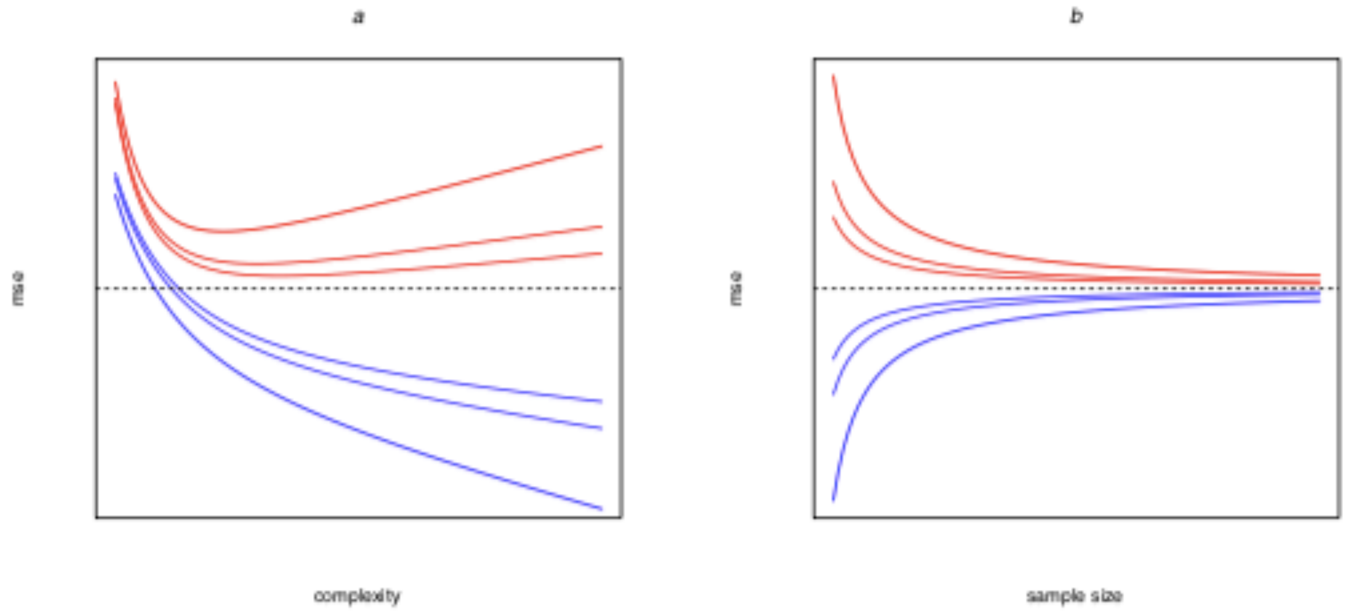


Figure 12: The mse on the training and testing data for ridge regression as function of p and n. Assuming the training and testing samples have the same sizes. The convergence to σ^2 is faster for increasing λ.

Ensemble Methods: Motivation

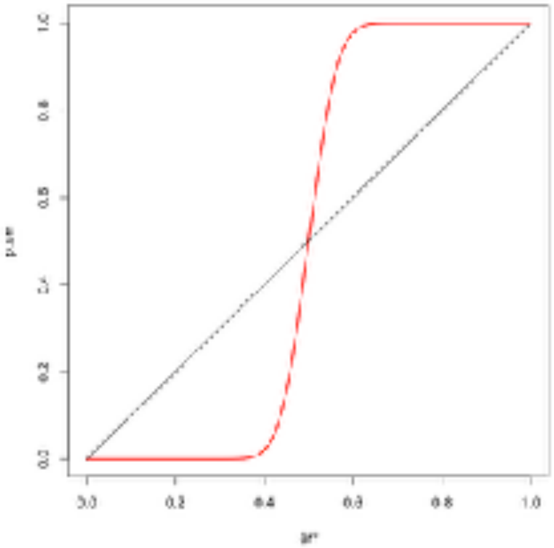
Ensemble methods more accurate than any individual members:

- Accurate (better than guessing)
- Diverse (different errors on new examples)

Independent errors:

prob *k* of *N* classifiers (independent error rate *ε*) wrong:

$$P(\#\text{errrors} = k) = \binom{N}{k} \varepsilon^{k} (1-\varepsilon)^{N-k}$$

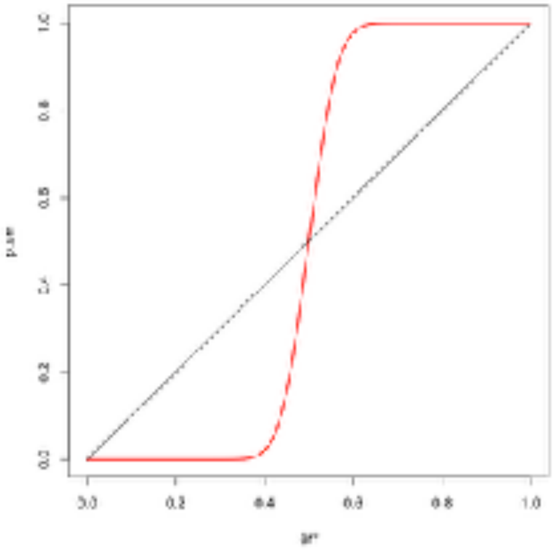


Ensembles intuition

Example 1:

- Suppose there are 25 base classifiers
  - Each classifier has error rate, ε = 0.35
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^{i} (1-\varepsilon)^{25-i} = 0.06$$



Example 2:

- Suppose there are *N=5* base classifiers
  - Each classifier has accuracy is *ε=0.70%*
  - Assume classifiers are independent
  - What is the accuracy of majority vote:

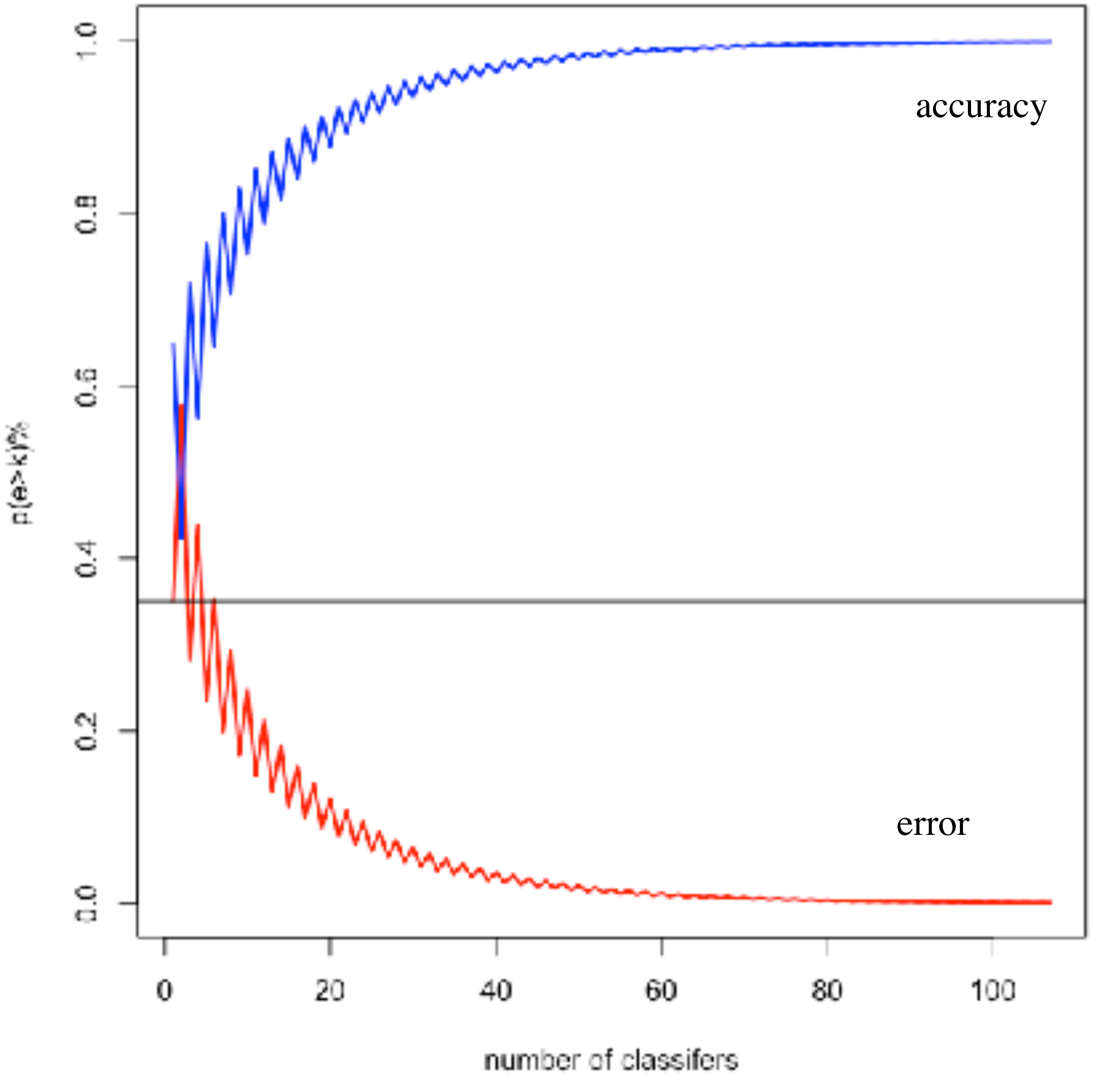
$$\begin{aligned} Pr(X > k) &= \sum_{i=k}^{N} \binom{N}{i} \epsilon^{i} (1-\epsilon)^{N-i} \\ &= \sum_{i=3}^{5} \binom{5}{3} (.65)^{i} (.35)^{5-i} \\ &= 10(.65)^3(.35)^2 + 5(.65)^4(.35) + (.65)^5 = 0.7648306 \end{aligned}$$

What is the accuracy of majority vote if *N=107?*

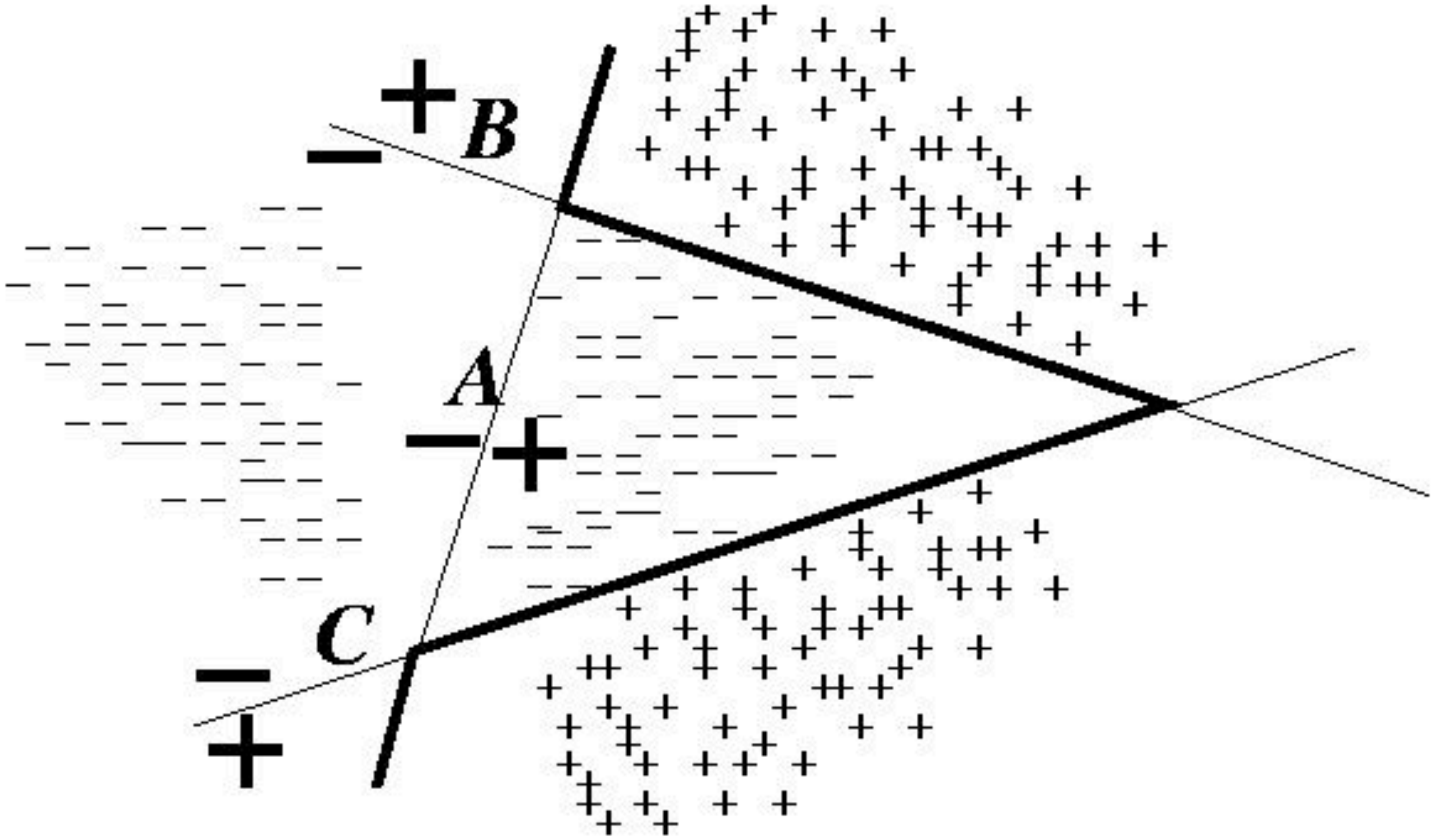
Ensembles and Netflix

Original progress prize winner (BellKor) was ensemble of 107 models!

10% increase in accuracy



How Ensemble improves accuracy



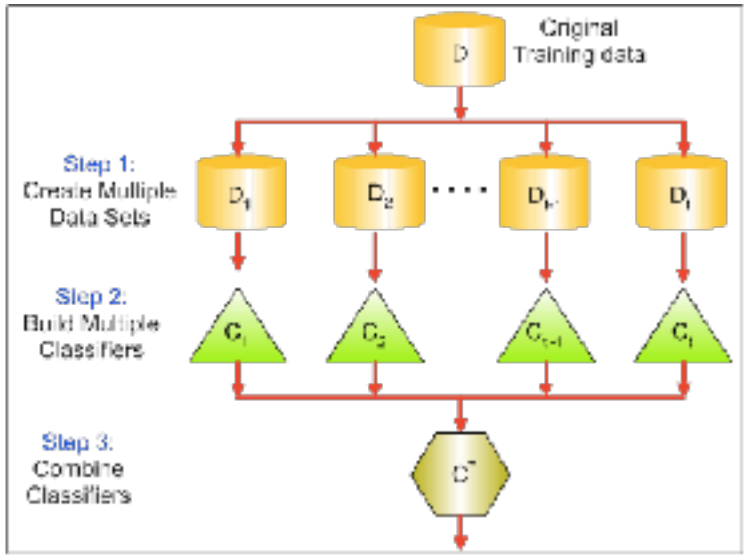
This figure depicts a classification problem in which the goal is to separate the points marked with plus signs from points marked with minus signs. None of the three individual linear classifiers (marked A, B, and C) is able to separate the two classes of points. However, a majority vote over all three linear classifiers yields the piecewise-linear classifier shown as a thick line. This classifier is able to separate the two classes perfectly.

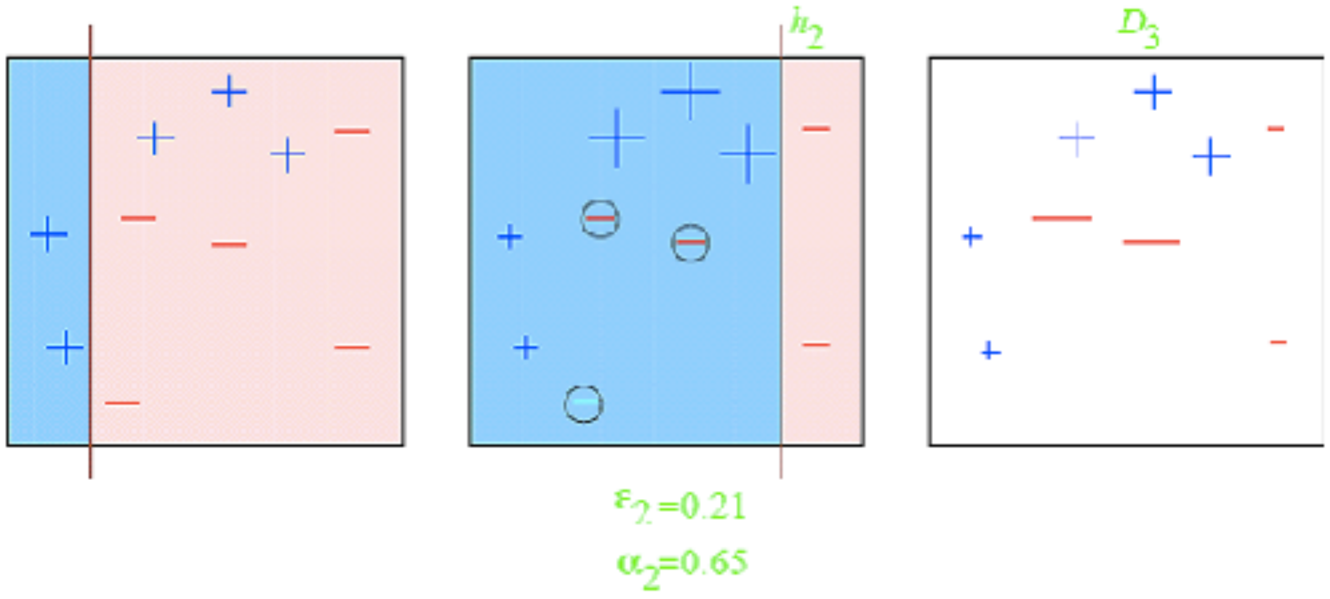
An ensemble of linear classifiers. Each line---A, B, and C---is a linear classifier.
The boldface line is the ensemble that classifies new examples by returning the majority vote of A, B, and C.

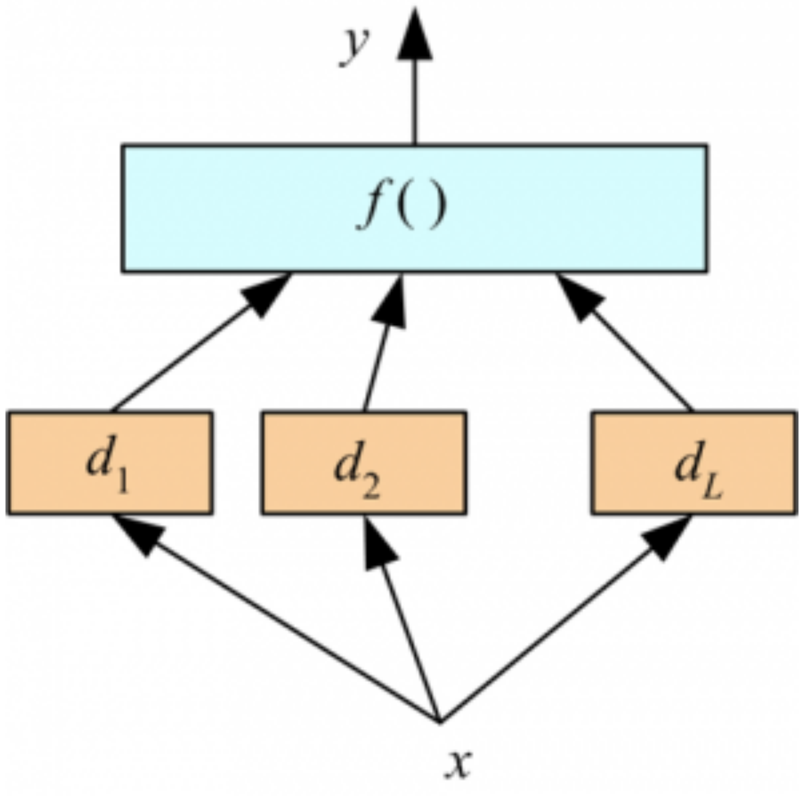
Ensemble Models

1. What is an Ensemble Model?

2. What are Bagging, Boosting and Stacking?


D
Original Training data
Step 1: Create Multiple Data Sets
D1
D2
Dt-1
Dt
Step 2: Build Multiple Classifiers
C1
C2
Ct-1
Ct
Step 3: Combine Classifiers
C*



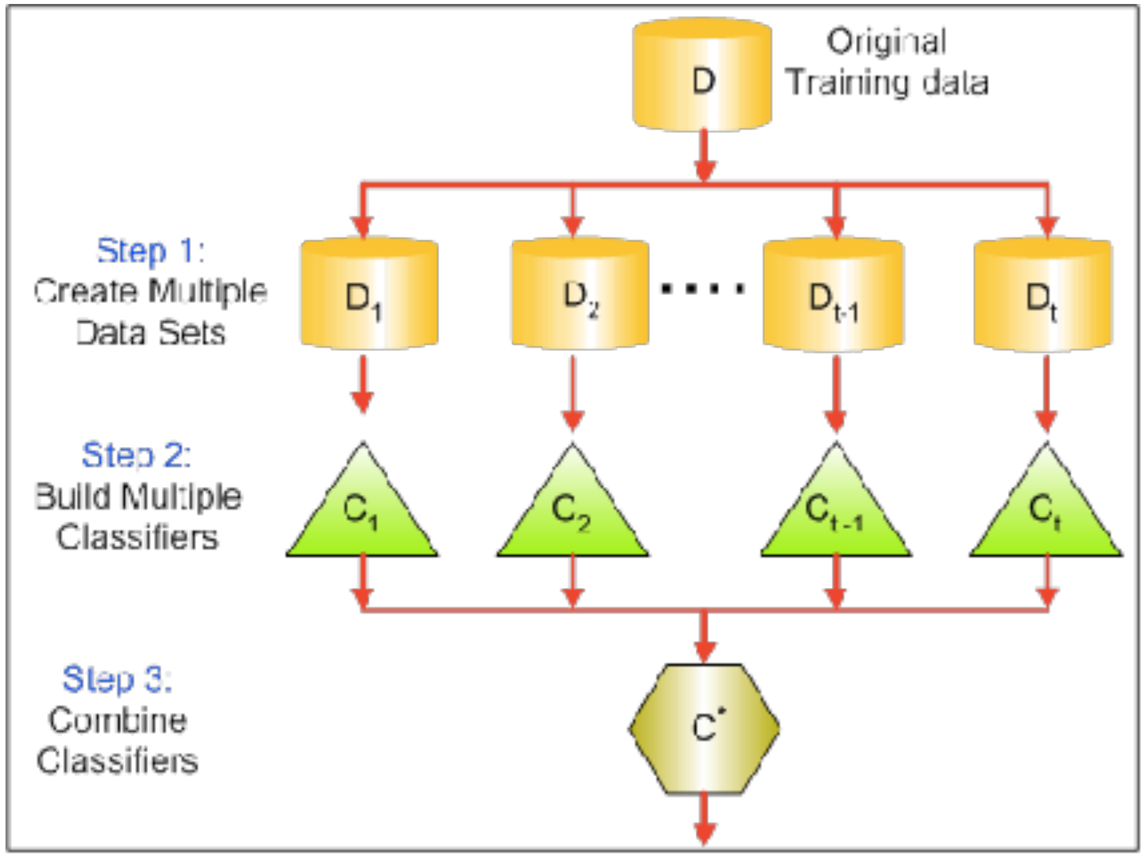


What are ***Bagging***, Boosting and Stacking?

Bagging (Bootstrap Aggregating)

1. Create random samples of the training data set (sub sets of training data set).
2. Build a classifier for each sample.
3. Results of these multiple classifiers are combined using average or majority voting.

Bagging helps to reduce the variance error.


D
Original Training data
Step 1: Create Multiple Data Sets
D_1
D_2
....
D_{t-1}
D_t
Step 2: Build Multiple Classifiers
C_1
C_2
C_{t-1}
C_t
Step 3: Combine Classifiers
C^*

Bagging (Constructing for Diversity)

1. Use random samples of the examples to construct the classifiers
2. Use random feature sets to construct the classifiers
   - Random Decision Forests

- Bagging: ***B****ootstrap* ***Agg****regation*

Leo Breiman

Random Forest

- Sample a data set with replacement
- Select ***m*** variables at random from ***p*** variables
- Create a tree
- Similarly create more trees
- Combine the results

Reference: – Hastie, Tibshirani, Friedman, The Elements of Statistical Learning, Chapter 15

Random Forest

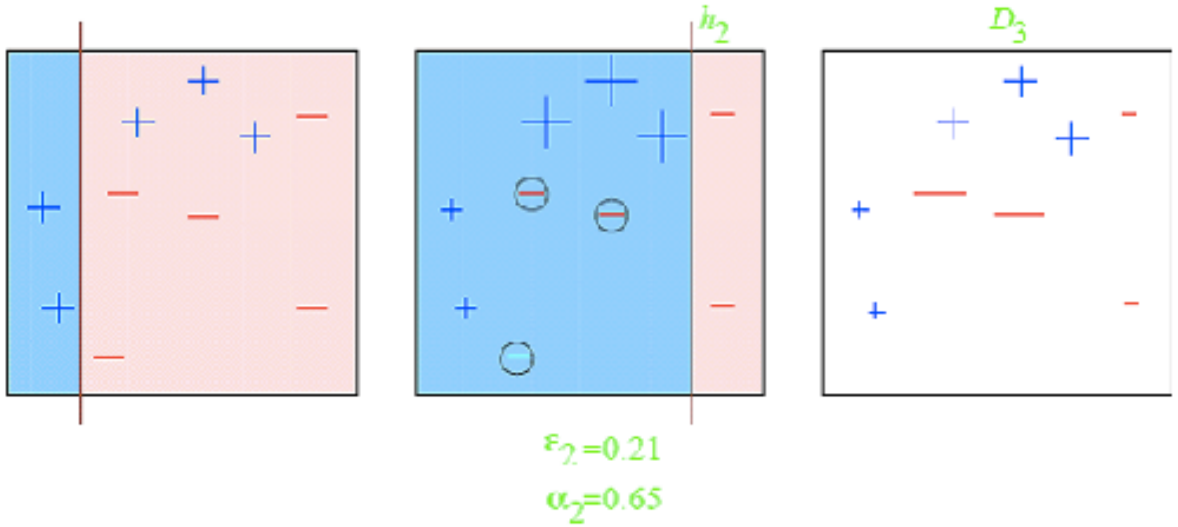
Advantages:

- Only for decision trees Lowers generalization error
- Uses randomization in tree construction: #features= log2(d+1)
- Equivalent accuracy to AdaBoost, but faster

See table in Tan et al p. 294 for comparison of ensemble methods.

What are Bagging, *Boosting* and Stacking?

Most common example of boosting is AdaBoost and Gradient Boosting.

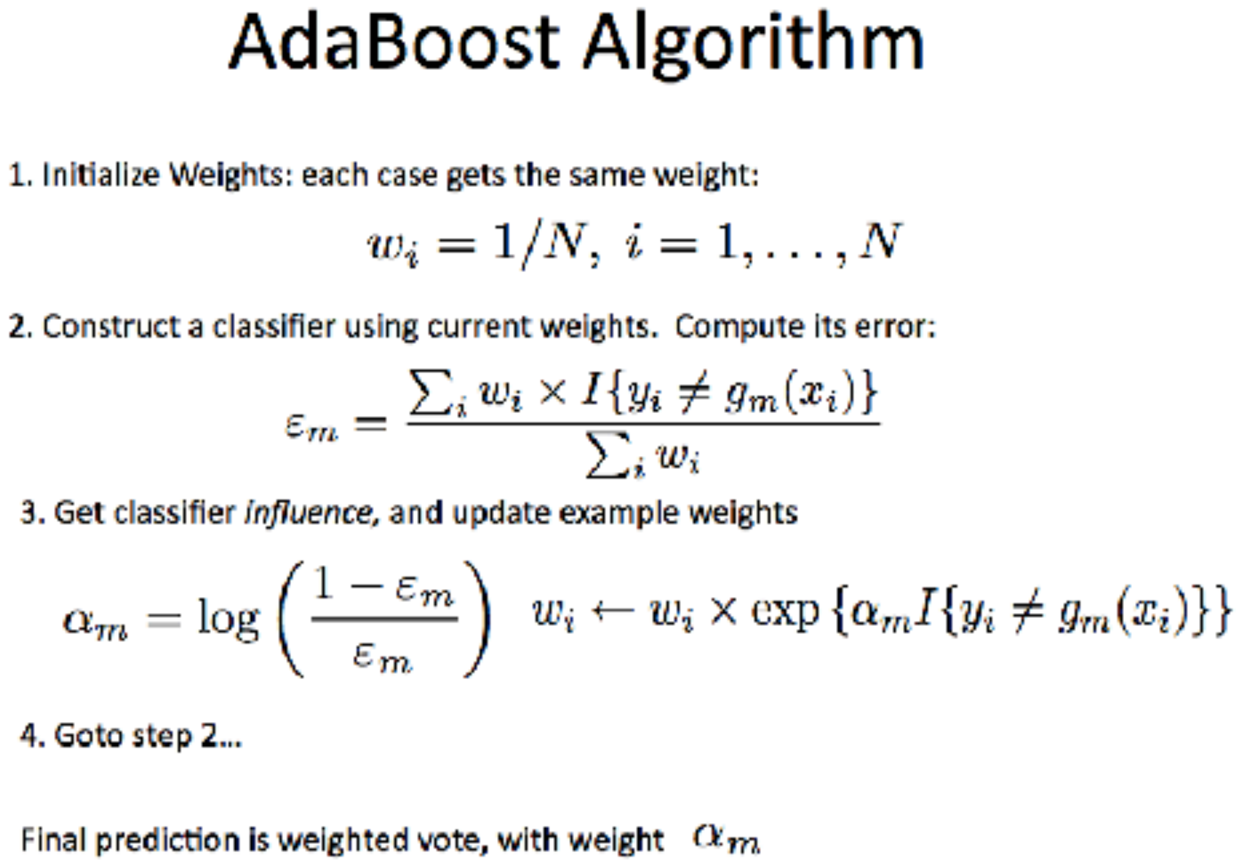


1. Create a sequence of classifiers, giving higher influence to more accurate classifier

2. At each iteration, make examples currently misclassified more important (get larger weights in the construction of next classifier)

3. Combine classifier by weighted vote (weight given by classifier accuracy).

Boosting has shown better predictive accuracy than bagging, but it also tends to over-fit the training data as well.



AdaBoost Algorithm

1. Initialize Weights: each case gets the same weight:

$$w_i = 1/N, \; i = 1, \ldots, N$$

2. Construct a classifier using current weights. Compute its error:

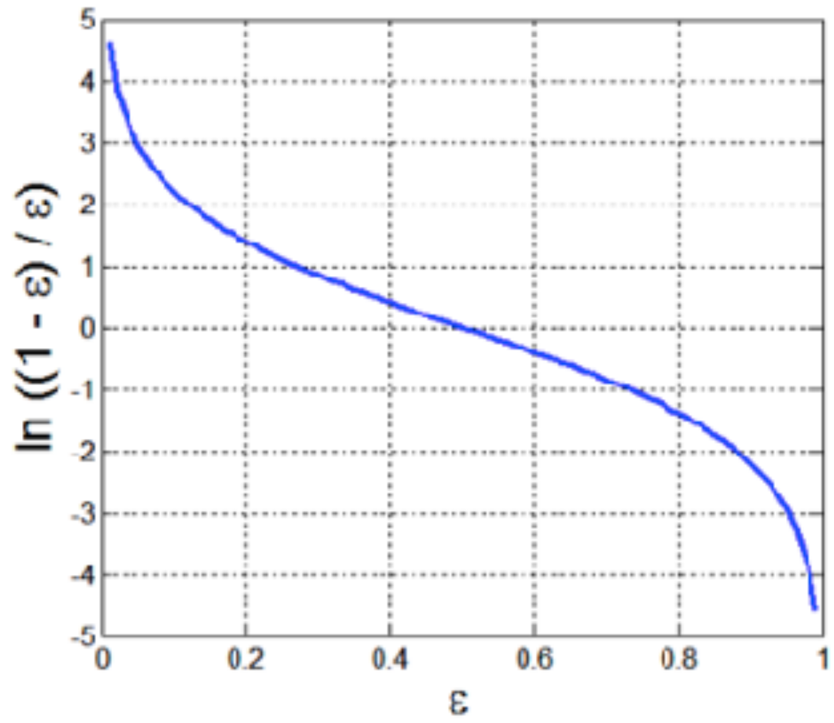
$$\varepsilon_m = \frac{\sum_i w_i \times I\{y_i \neq g_m(x_i)\}}{\sum_i w_i}$$

3. Get classifier *influence*, and update example weights

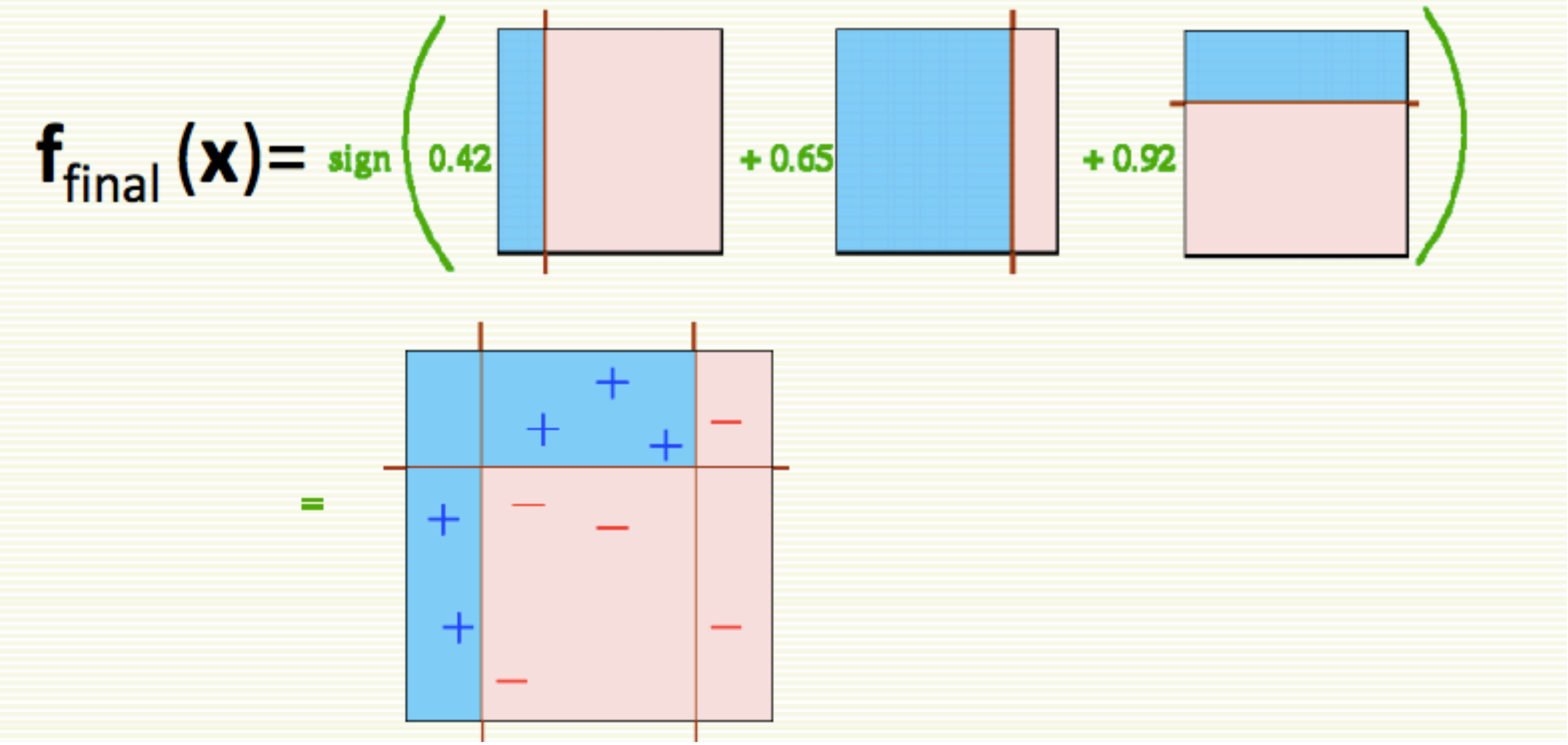
$$\alpha_m = \log\left(\frac{1-\varepsilon_m}{\varepsilon_m}\right) \qquad w_i \leftarrow w_i \times \exp\left\{\alpha_m I\{y_i \neq g_m(x_i)\}\right\}$$

4. Goto step 2...

Final prediction is weighted vote, with weight α_m



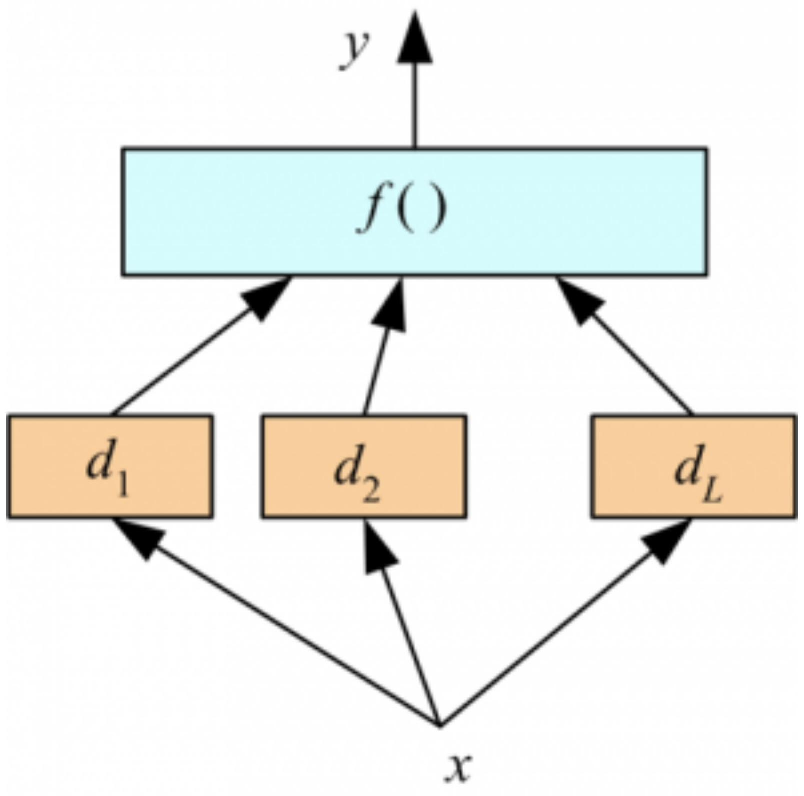
AdaBoost(1996)


$f_{final}(x) = \text{sign}\left(0.42 \; + 0.65 \; + 0.92\right)$
=
+
+
+
+
+
−
−
−
−
−

What are Bagging, Boosting and *Stacking*?

Stacking works in two phases.

1. Use multiple base classifiers to predict the class.
2. Use a new learner to combine their predictions with the aim of reducing the generalization error.



Ensemble Models

Questions:

1. Can we ensemble multiple models of same ML algorithm?
2. Let’s say we have three models (A, B and C). A, B and C have prediction accuracy of 85%, 80% and 55% respectively. But A and B are found to be highly correlated where as C is meagerly correlated with both A and B. Should we combine A and B?

Ensemble Models

Question 1: Can we ensemble multiple models of same ML algorithm?

Answer:

1. we can combine multiple models of same ML algorithms, but combining multiple predictions generated by different algorithms would normally give you better predictions.
    1. It is due to the diversification or independent nature as compared to each other.
    2. For example, the predictions of a random forest, a KNN, and a Naive Bayes may be combined to create a stronger final prediction set as compared to combining three random forest model.
    3. The key to creating a powerful ensemble is model diversity.
    4. An ensemble with two techniques that are very similar in nature will perform poorly than a more diverse model set.

Ensemble Models

Question 2:

Let’s say we have three models (A, B and C). A, B and C have prediction accuracy of 85%, 80% and 55% respectively. But A and B are found to be highly correlated where as C is meagerly correlated with both A and B. Should we combine A and B?

Answer:

No, we shouldn’t, because these models are highly correlated. We shouldn’t combine these two as this ensemble will not help to reduce any generalization error. I would prefer to combine A & C or B & C.