Combining Multiple Learners Part 1

Ethem Chp. 15
Hastie Chp 8
Haykin Chp. 7, pp. 351-370

Overview

- **Introduction**
  - □ Motivation

- **Static structures:** the responses of several *experts* (individual networks) are combined in a way that **does not** involve the input signal.
  - □ ***Ensemble averaging / Majority Voting***
  - □ ***Boosting***
  - □ ***Stacking***
  - □ *Error orrecting Output Codes*

- **Dynamic structures**: the input signal actuates the mechanism that combines the responses of the experts.
  - □ ***Mixture of experts***
  - □ *Hierarchical mixture of experts*

Motivation

- When designing a learning machine, we generally make some choices:
  - parameters of machine, training data, representation, etc…
- This implies some sort of variance in performance
- **Why not keep all machines and combine their predictions?**
- Intuition: Combining experts opinions, votes,...

- Ensemble Learning / Classifier Combination:
  - do not learn a single classifier but learn a set of base classifiers
  - combine the predictions of multiple classifiers

- Hope to….:
  - reduce variance: results are less dependent on peculiarities of a single training set
  - reduce bias: a combination of multiple classifiers may learn a more expressive concept class than a single classifier

- Important:
  - formation of an ensemble of diverse classifiers from a single training set

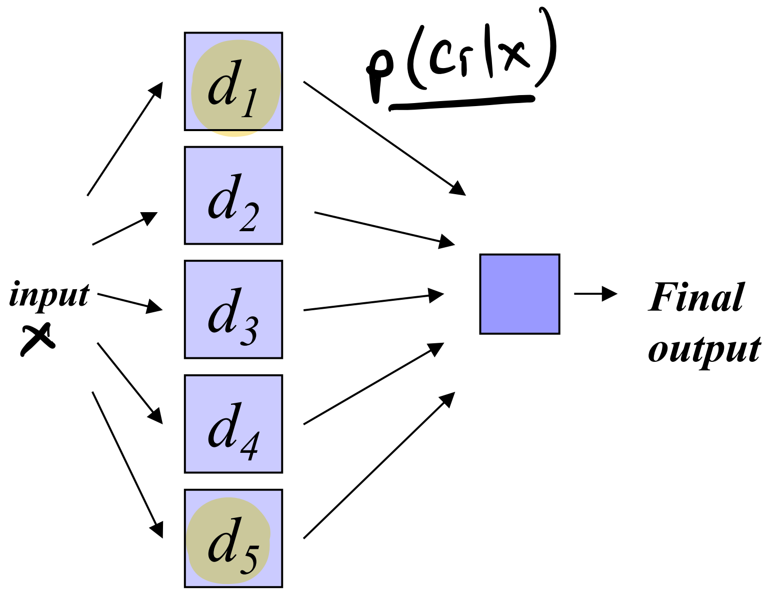
Rationale

- No Free Lunch thm: “There is no algorithm that induces the most accurate learner in any domain, all the time.”
  - □ http://www.no-free-lunch.org/

- Generate a group of base-learners which when combined has higher accuracy

- Different learners use different
  - □ Algorithms: making different assumptions
  - □ Hyperparameters: e.g number of hidden nodes in NN, k in k-NN
  - □ Representations: diff. features, multiple sources of information
  - □ Training sets: small variations in the sets or diff. subproblems

Reasons to Combine Learning Machines



Terminology:
Ensemble formed
from base classifiers

Lots of different combination methods:

Most popular are *ensemble averaging* and *majority voting.*

$$\frac{1}{L}\sum_{J} p(C_{kJ}|x)$$

Ensemble Averaging

- Regression

$$y = \sum_{j=1}^{L} w_j d_j$$

$$w_j \geq 0 \text{ and } \sum_{j=1}^{L} w_j = 1$$

- Classification

$$y_i = \sum_{j=1}^{L} w_j d_{ji}$$

d_j is the output of the jth base classifier in regression

d_{ji} is the output of the jth base classifier for ith class in classification

w_j often inversely proportional to error rate of classifier:

Learned over a validation set

Ensemble Averaging

The output of the ensemble is the (weighted) average of the base classifiers` outputs.

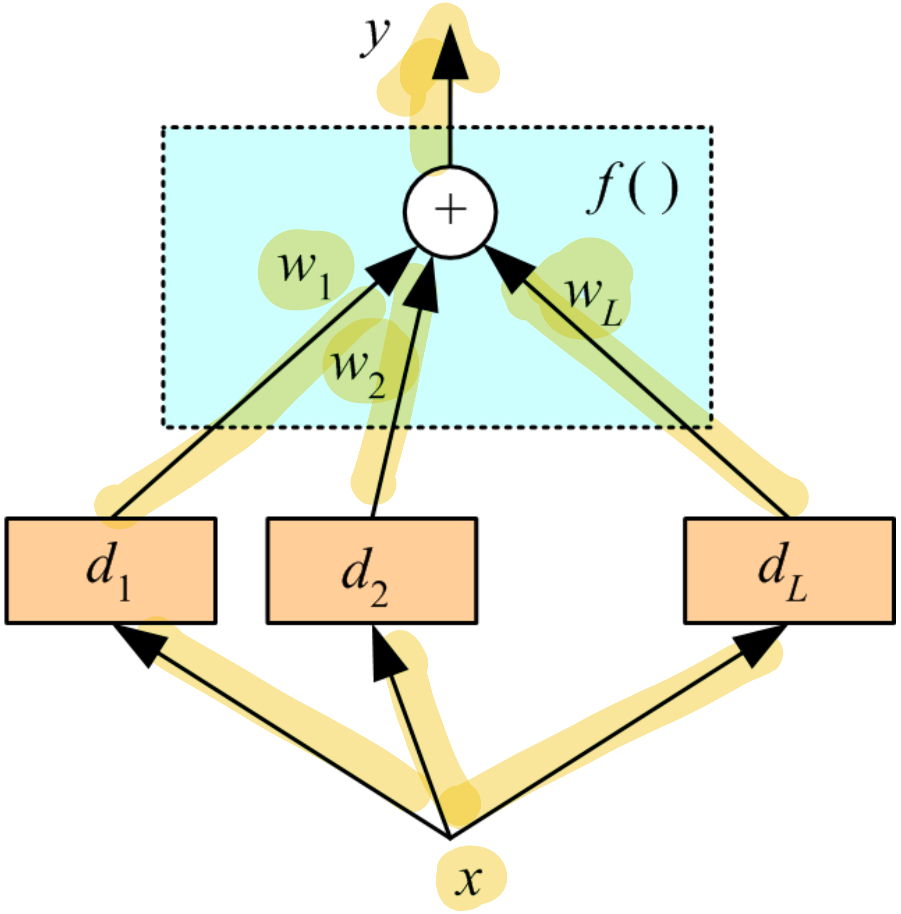
- Regression

$$y = \sum_{j=1}^{L} w_j d_j$$

$$w_j \geq 0 \text{ and } \sum_{j=1}^{L} w_j = 1$$

- Classification

$$y_i = \sum_{j=1}^{L} w_j d_{ji}$$



Ensemble Averaging

If we use a committee machine f$_{com}$ whose output is the average:

$$f_{com} = \frac{1}{L}\sum_{i=1}^{i=L} d_i$$ base learners

Error of combination is <u>guaranteed</u> to be lower than the <u>average</u> error:
ensemble

$$(f_{com}-t)^2 = \underbrace{\frac{1}{L}\sum_i \underline{(d_i-t)^2}}_{\text{average error}} - \underbrace{\frac{1}{L}\sum_i (d_i - f_{com})^2}_{\text{always positive}}$$

(Krogh & Vedelsby 1995)

$\mathrm{Avg}(e_{d_1}, e_{d_3}, e_{d_3})$

$\wedge$ e_{fcomm}.

- Similarly, we can show that if d_j are iid:

$$\text{Bias}^2: \quad (E_D(d)-f)^2$$
$$\text{Variance}: \; E_D[(E_D(d)-d)^2]$$

$$E\left[f_{com}\right]=E\left[\sum_j \frac{1}{L}d_j\right]=\frac{1}{L}L\cdot E\left[d_j\right]=E\left[d_j\right]$$

$$\text{Var}\left(f_{com}\right)=\text{Var}\left(\sum_j \frac{1}{L}d_j\right)=\frac{1}{L^2}\text{Var}\left(\sum_j d_j\right)=\frac{1}{L^2}L\cdot\text{Var}\left(d_j\right)=\frac{1}{L}\text{Var}\left(d_j\right)$$

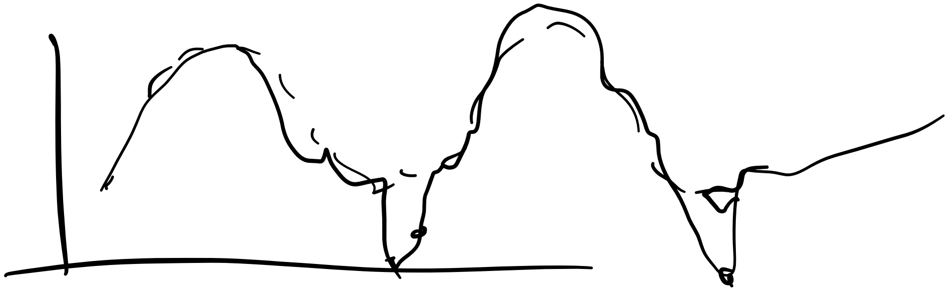
- Bias does not change, variance decreases by 1/*L*

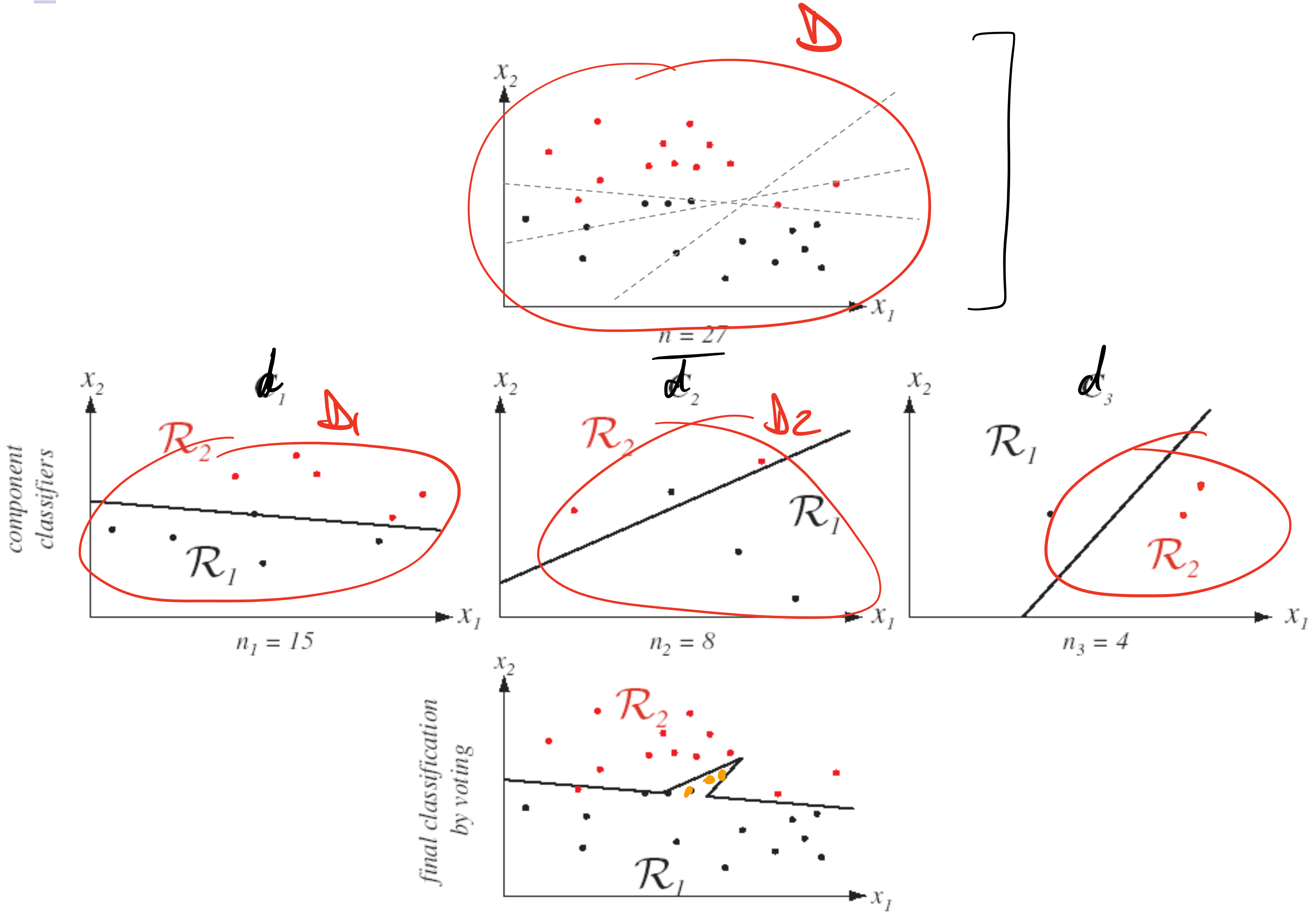
Ensemble Averaging

What we can exploit from this fact:

- Combine multiple experts with the same bias and variance, using ensemble-averaging
  - □ the bias of the ensemble-averaged system would the **same as the bias of one of the individual experts**
  - □ the variance of the ensemble-averaged system would be **less than the variance of one of the individual experts.**

➢ We can purposefully use complex/flexible models, the variance will be reduced due to averaging.




D
x2
x1
n = 27
component classifiers
d1
d2
d3
D1
D2
R1
R2
x2
x1
n1 = 15
n2 = 8
n3 = 4
final classification by voting

Majority Voting

In classification, majority voting chooses the class that gets the most votes from the base classifiers (not averaging the output probabilities).

- Majority voting is also considered a form of ensemble averaging (**hard vs soft averaging**).

ensemble "

averaging posterior probs $\{P(C_k | x)\}$

scores

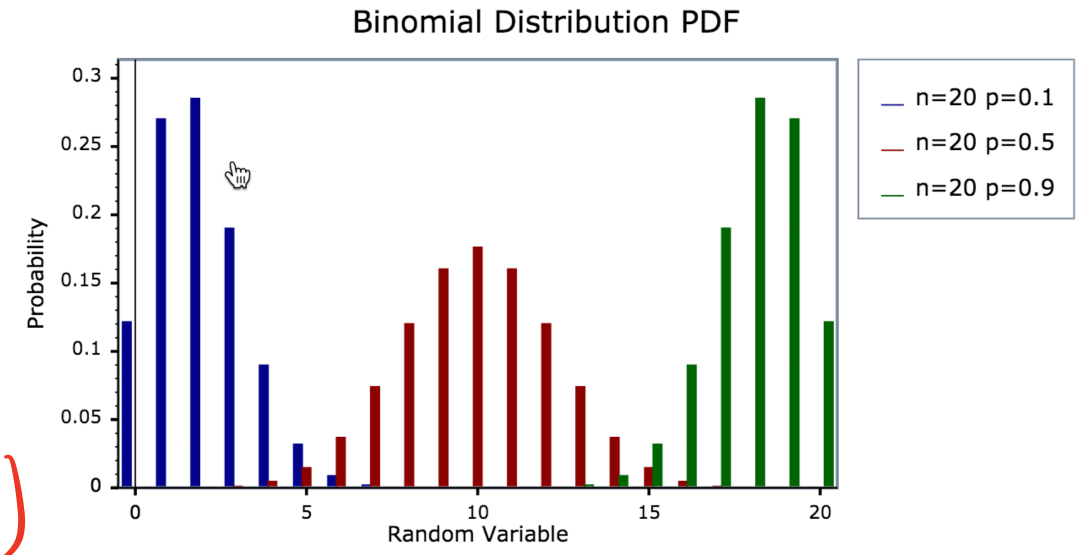
confidences

Majority voting ensemble makes an error when more than half of the base classifiers make a mistake. What is this probability?

	Class A	Class B
d_1	1	0
d_2	1	0
d_3	0	1

ens. = 1 (chooses Class A w/ majority vote (hard averaging))
fcomm

Hint:



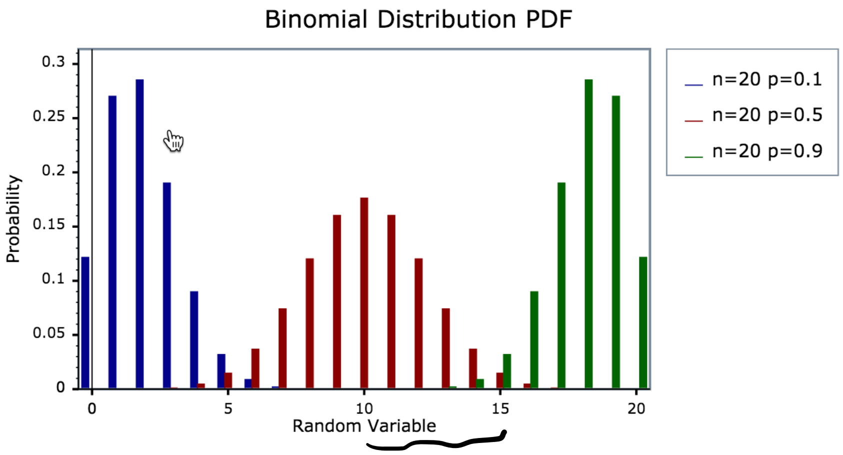
Reasons to Combine Learning Machines

If the base classifiers are all independent, the probability of error of the **majority voting ensemble** is:

$$\overset{\text{ensemble}}{P(error)} = \sum_{k=\frac{N}{2}+1}^{N} \binom{N}{k} p^{k} (1-p)^{N-k}$$

p : probability that d_i makes an error.

In prev. example, sum goes from $k=2$ to $k=3$.



Example

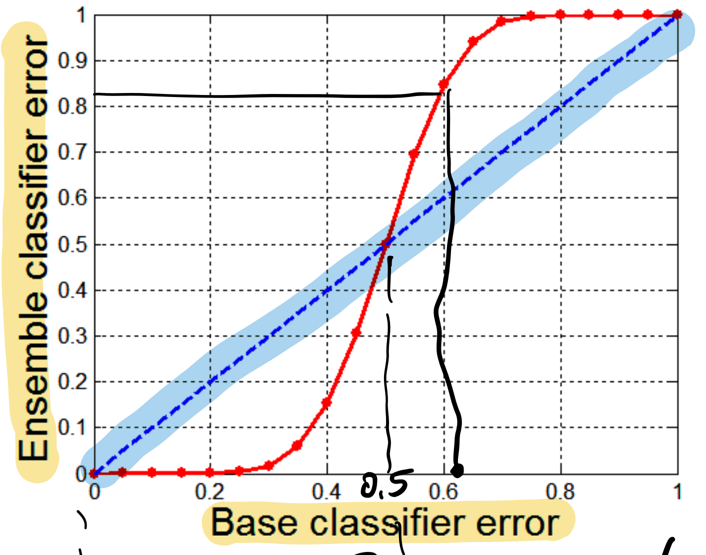
- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume classifiers are independent
    - i.e., probability that a classifier makes a mistake does not depend on whether other classifiers made a mistake
    - **Note:** in practice they are not independent!
- Probability that the ensemble classifier makes a wrong prediction
  - The ensemble makes a wrong prediction if the majority of the classifiers makes a wrong prediction
  - The probability that 13 or more classifiers err is

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

Based on a slide by Kumar et al.

Why Majority Voting works?

- Suppose there are 25 base classifier, and
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Errors made by classifiers are uncorrelated



ε

ens. error is less than ε

ens. err. is higher than ε

– Probability that the ensemble classifier makes a wrong prediction:

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

Note: Ensemble does not help when base classifier error is high!
(where red curve is higher than the blue line)

- **We want the base learners to be complementary**
  - What if they were all the same or very similar?
    - □ No use...

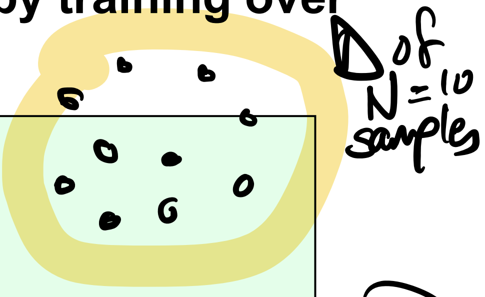
- **Reasonably accurate, but not necessarily very accurate**

Overview

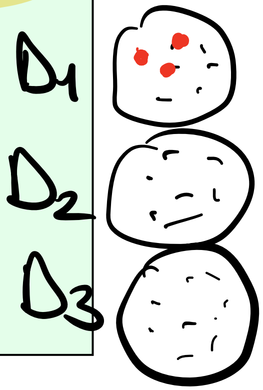
- Introduction
  - Rationale
- Combination Methods
  - Static Structures
    - Ensemble averaging / Majority Voting
    - **Bagging**
    - Boosting
    - Error Correcting Output Codes
  - Dynamic structures
    - Mixture of Experts
    - Hierarchical Mixture of Experts

Ensemble Methods > Bagging

Voting method where base-learners are made different by training over slightly different training sets.



```
Bagging (Bootstrap Aggregating) - Breiman, 1996
    take a training set D, of size N
    for each network / tree / k-nn /  etc…base learner
            - build a new training set by sampling N examples,
                randomly with replacement, from D
            - train your machine with the new dataset
    end for
    output is average/vote from all machines trained
```



- □ Resulting base-learners are similar because they are drawn from the same original sample
- □ Resulting base-learners are slightly different due to chance

- Generate new training sets using sampling with replacement

D	Original Data	1	2	3	4	5	6	7	8	9	10
→ D_1	Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
→ D_2	Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
→ D_3	Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- some examples may appear in more than one set (samples 1, 3, 4, 6 are not in D_1)
- for each set, the probability that a given example doesn't appear in it is

(x e.g. sample 4)

$$\Pr(x \notin D_i) = \left(1 - \frac{1}{n}\right)^{\boxed{n}} \rightarrow 0.3678$$

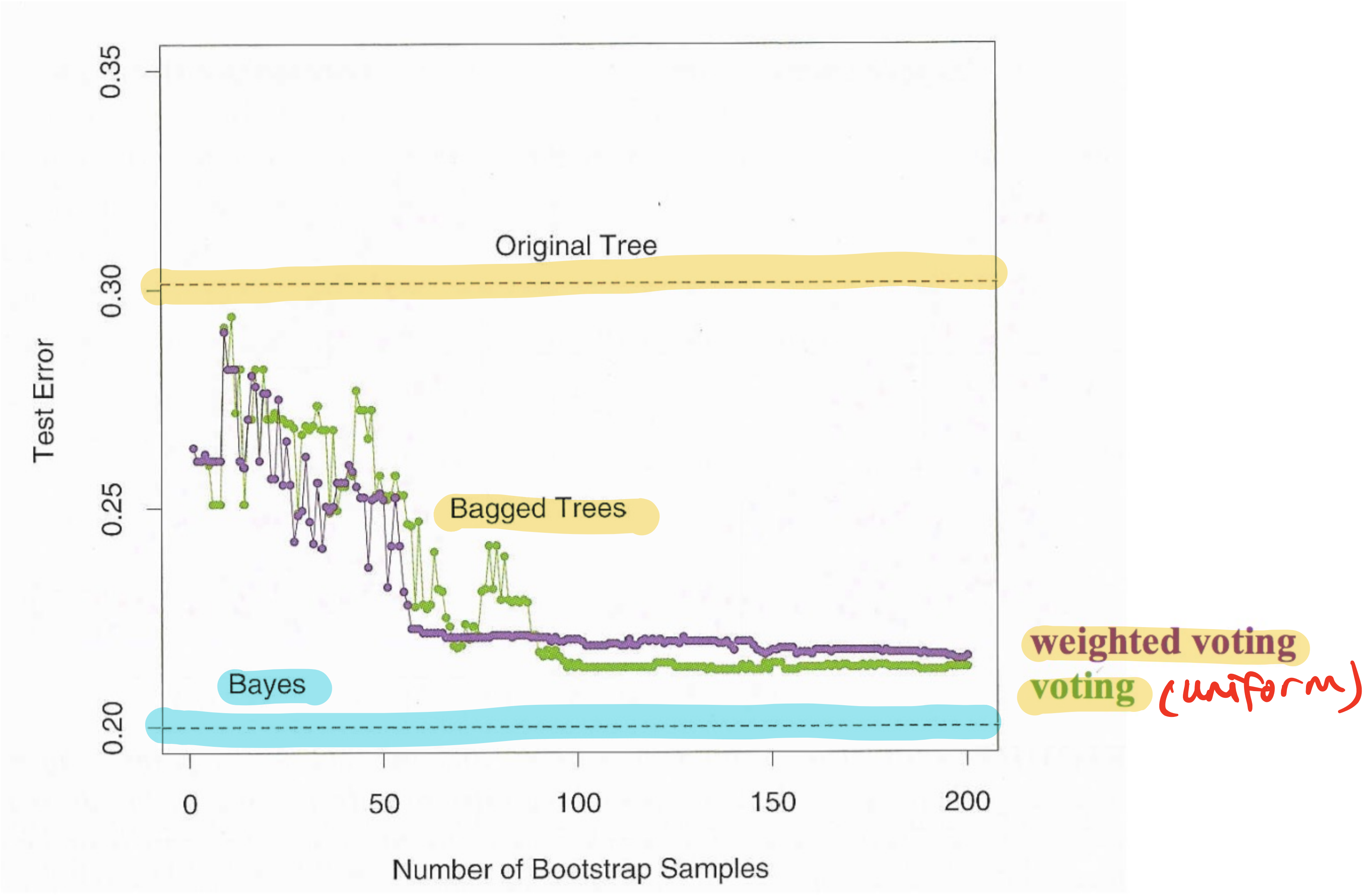
- i.e., less than 2/3 of the examples appear in one bootstrap sample

Based on a slide by Kumar et al.

Bagging

- Not all data points will be used for training
  - □ Waste of training set
    - Each sample has a probability of 0.37 of not being selected in any one bootsrap training set.
    - I.e. only about 2/3rd of the samples are used in any one bootsrap sample.
    - They can be used for testing (see Out-of-Bag error in Random Forests)

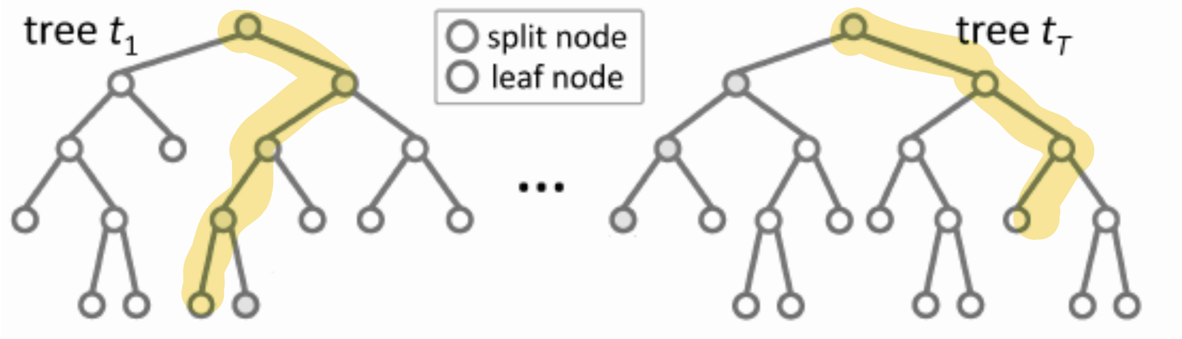
- Bagging is suitable for **unstable learning** algorithms
  - □ Unstable algorithms change significantly due to small changes in the data.
  - □ Such as MLPs, decision trees



from Hastie, Tibshirani, Friedman: The Elements of Statistical Learning, Springer Verlag 2001

Random Forests

Random Forests

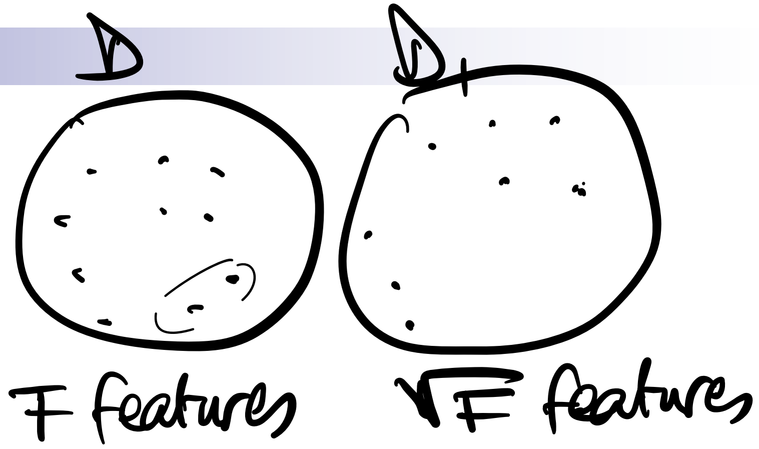


- An improved method over simple bagged trees
- All trees vote to produce a final answer.

Motivation

- **With decision trees, it is found that optimal cut points can strongly depend on the training set used.**
  - This suggested using multiple trees and use voting to combine the results.
- Averaging the outputs of the trees reduces overfitting.
- For the use of multiple trees to be most effective, the trees should be independent as possible.
  - Splitting using a random subset of features hopefully achieves this.
  - If the trees really are independent, the performance should improve with more trees

The Random Forests Algorithm



//Random forest with k trees

Given a training set S

- For i = 1 to k do:
- Build subset S_i by sampling with replacement from S

  Learn tree T_i from S_i as:

  At each node:

  Choose best split from random subset of F features

  Each tree grows to the largest extend, no pruning
- Make predictions according to **majority vote** of the set of k trees.

- **Bagging**

  Generate randomized training sets by sampling with replacement from the full training set (bootstrap sampling)

Full training set	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}	D_{11}	D_{12}
Random "bag"	D_4	D_9	D_3	D_4	D_{12}	D_{10}	D_{10}	D_7	D_3	D_1	D_6	D_1

- **Feature subset selection**

  Choose different random subsets of the full feature vector to generate each tree

Full feature vector	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	F
Feature subset	f_4	f_6	f_7	f_9	f_{10}						$\sqrt{F}$

features

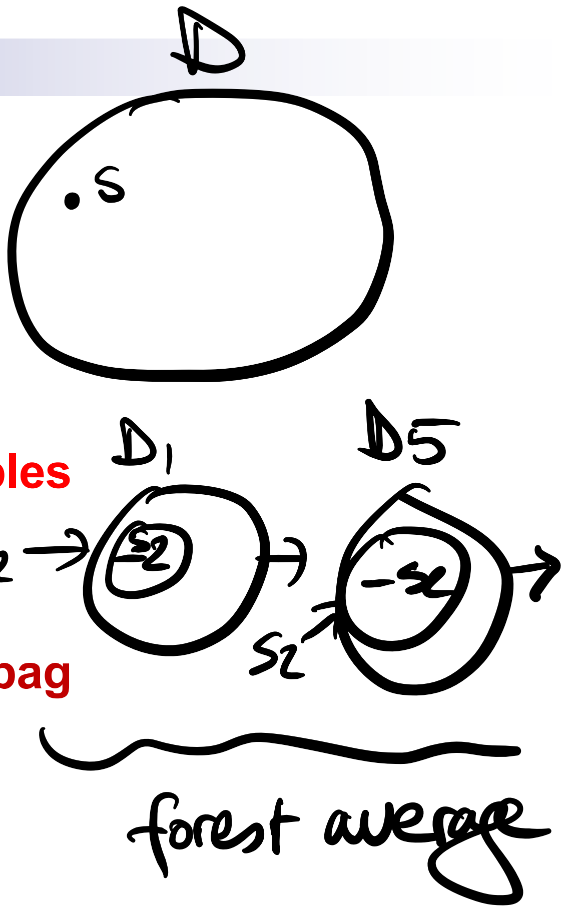
- Typically 5 – 100 trees are used. Often only a few trees are needed.

- Results seem fairly insensitive to the number of random attributes that are tested for each split. A common default is to use the square root of the number of attributes.

- Trees are fast to generate because fewer attributes have to be tested for each split and no pruning is needed.

Out-of-Bag Error

- To grow one tree
  - Bootstrap sample set from learning set L
  - Remaining samples are called **out-of-bag samples**
- For each sample $\overset{-s_2}{\underline{S}}$ of the learning set
  - **Look for all the trees for which S was out-of-bag**
  - Build the corresponding sub-forest
  - Predict the class of S with it
  - Measure error on S



- **Out-of-bag error** = average over all samples of S
  - Predictions not made using the whole forest... but with some aggregation
- Provides an estimation of the generalization error
  - Can be used to decide when to stop adding trees to the forest

Features of Random Forests

- It gives some of the best accuracy among current algorithms.
- It is efficient.
- It generates an internal unbiased estimate of the generalization error as the forest is built.

Plus…

- It handles missing data effectively.
  - A tree doesn't use all the features, plus other methods when a feature value is unknown…

- It gives estimates of what variables are important in the classification.
  - Impurity beased importance (average info gain from that feature over all trees)
  - Permutation importance

- It has methods for balancing error in class population unbalanced data sets.
  - Bootstrap sampling introduces variation in class proportions, or one can use a sample with equal class priors