Ensemble Learning

CS4780/5780 – Machine Learning
Fall 2013

Igor Labutov
Cornell University

Ensemble Learning

A class of "meta" learning algorithms

Combining multiple classifiers to increase performance

Very effective in practice

Good theoretical guarantees

Easy to implement!

Ensemble

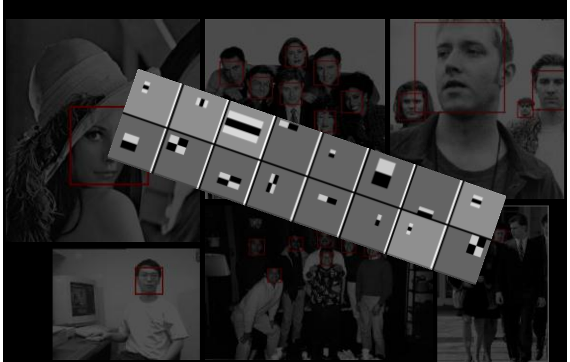
Problem : given T binary classification hypotheses (h_1,…, h_T), **find** a combined classifier:

$$h_S(x) = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

with better performance.

Teaser

Teaser



BAGGING

Bagging

Bagging (Boostrap aggregating). (Breiman, 1996)

```
BAGGING(S = ((x_1, y_1), ..., (x_m, y_m)))
  for t ← 1 to T do
      S_t ← BOOTSTRAP(S)  ▷ i.i.d. sampling with replacement from S.
      h_t ← TRAINCLASSIFIER(S_t)
  return h_S = x ↦ MAJORITYVOTE((h_1(x), ..., h_T(x)))
```

Bagging

Ensemble :

$$h_S(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Bagging : Special case where we fix:

$$\alpha_t = 1 \quad \text{and} \quad h_t = \mathbb{L}(S_t)^*$$

$^*\mathbb{L}$ is some learning algorithm

S_t is a training set drawn from distribution $P(<x,y>)$

Bias-Variance Tradeoff

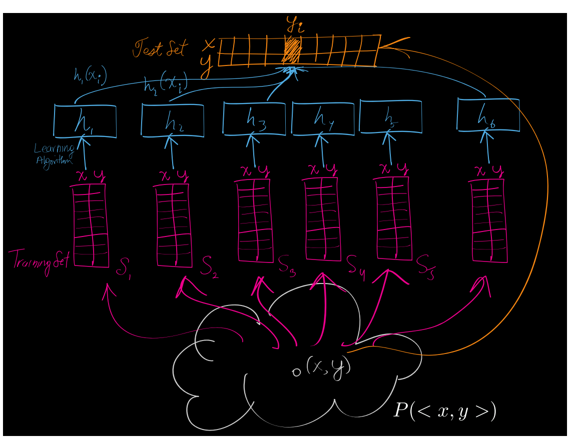
Generalization Error

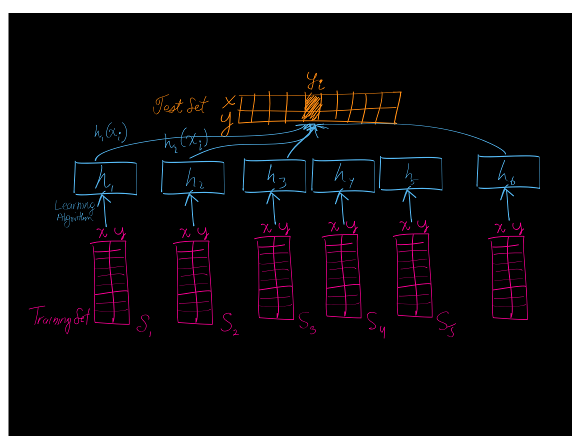
Classification :

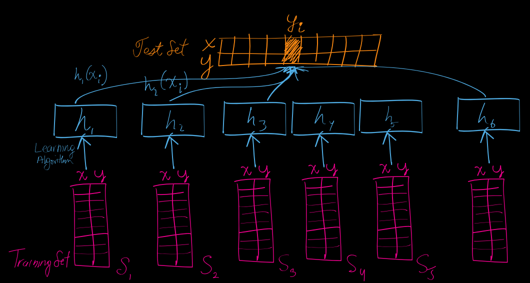
$$\epsilon_{test} = \frac{1}{n}\sum_{i}^{n} \text{Zero-One-Loss}(y_i, h(x_i))$$

Regression :

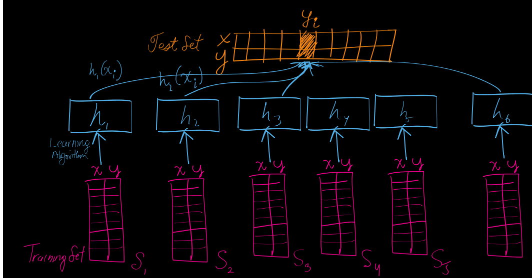
$$\epsilon_{test} = \frac{1}{n}\sum_{i}^{n} (y_i - h(x_i))^2$$







$$\bar{\epsilon}_{test}(x_i) = \frac{1}{T}\sum_{t}^{T}(y_i - h_t(x_i))^2$$



$$\bar{\epsilon}_{test}(x_i) = \frac{1}{T}\sum_{t}^{T}(y_i - h_t(x_i))^2$$

OR, as an expectation:

$$\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right]$$

For the entire test set:

$$\mathbb{E}_{X,Y}\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right]$$

CLAIM:

$$\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right] =$$

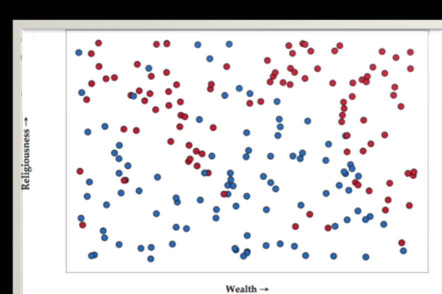
bias² $\quad (y_i - \mathbb{E}_S[h_S(x_i)])^2 \ +$

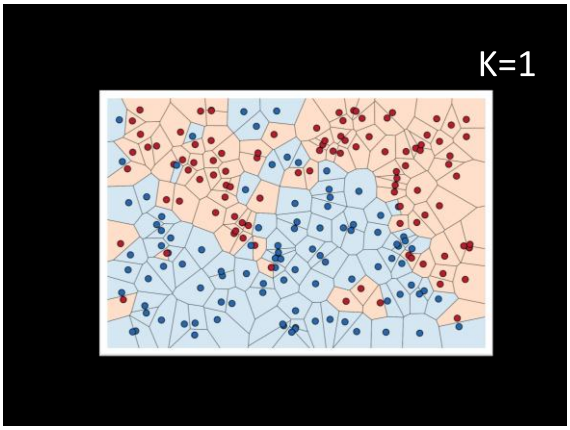
variance $\quad + \ \mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[(h_S(x_i))])^2]$

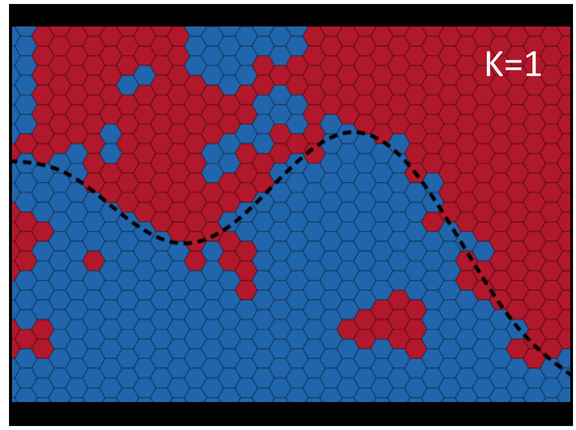
Example

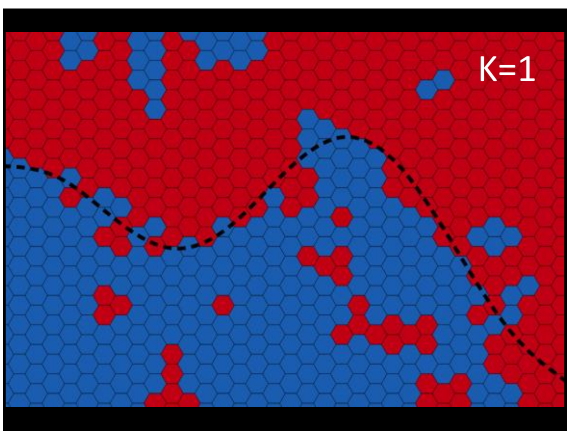
(kNN)

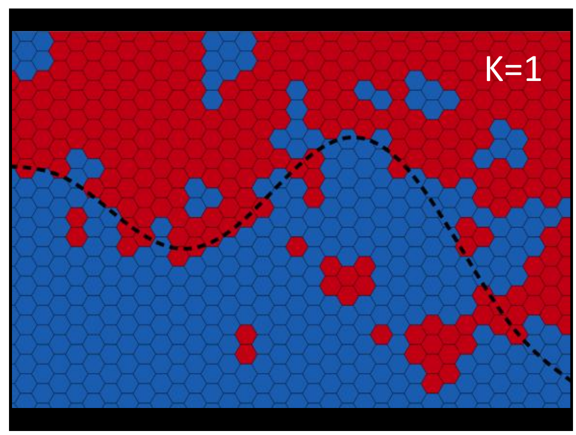
Democrat vs Republican party association

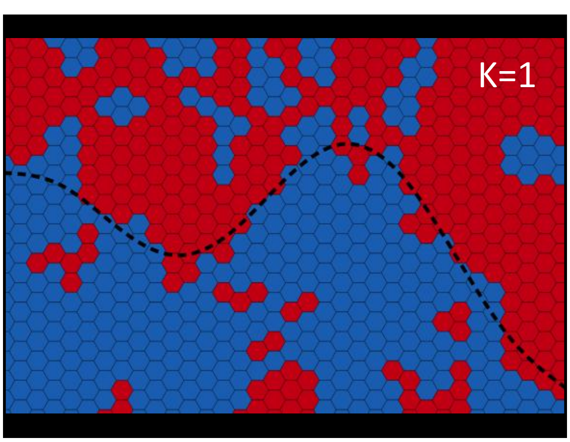


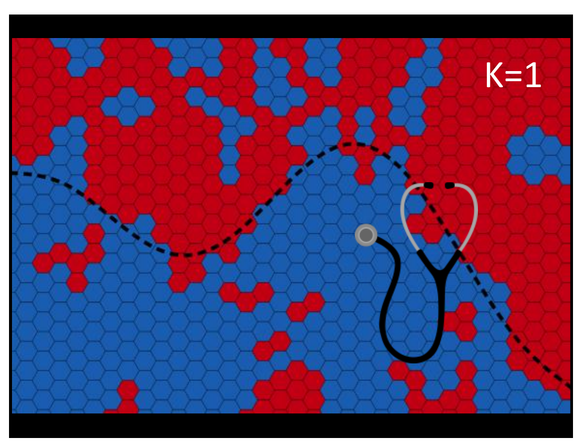

K=1

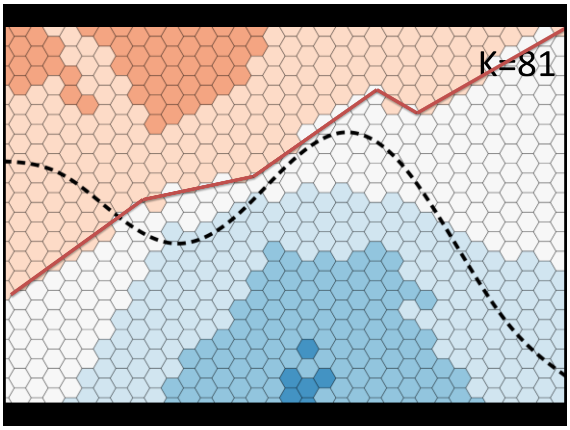

K=1

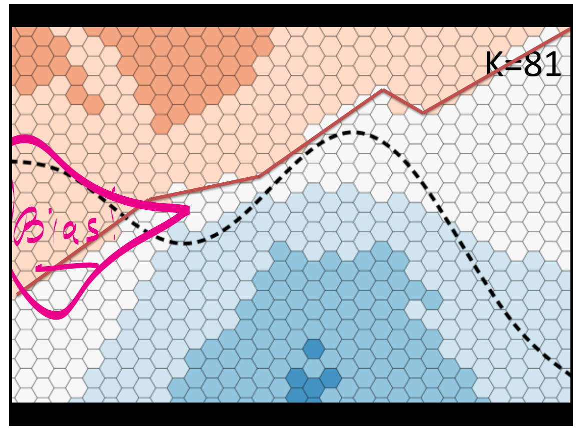

K=1

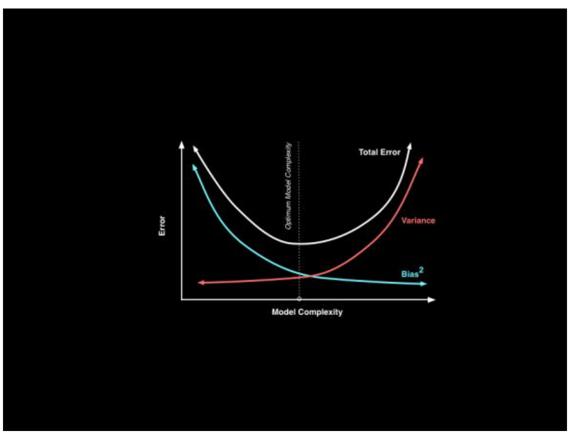

K=1

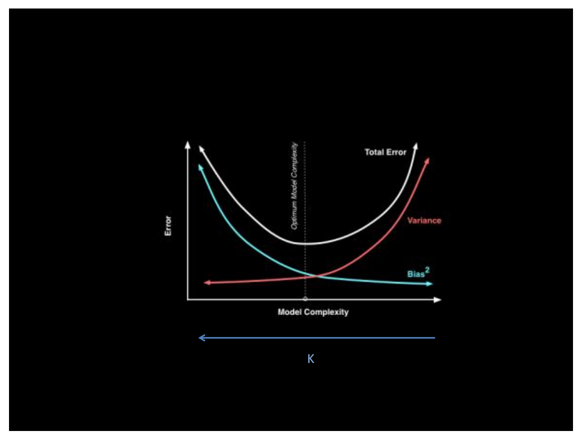

K=1

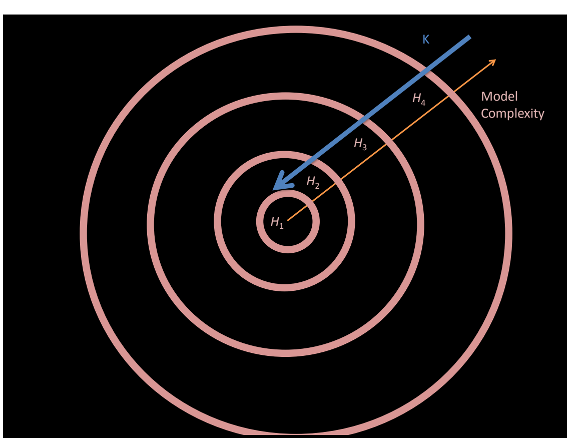

K=1

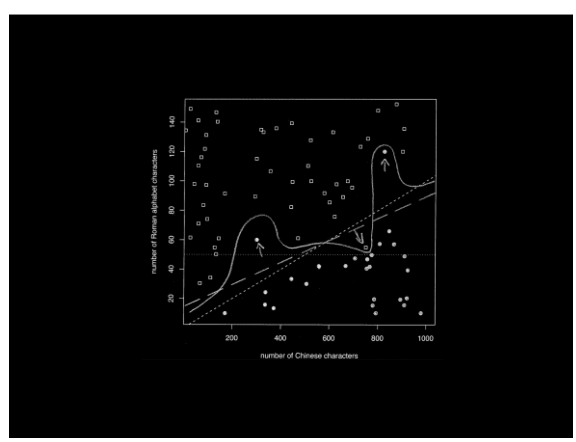

K=81


K=81
Bias


Total Error
Variance
Bias^2
Error
Optimum Model Complexity
Model Complexity


Total Error
Variance
Bias^2
Error
Optimum Model Complexity
Model Complexity
K


K
Model Complexity
H_4
H_3
H_2
H_1


number of Roman alphabet characters
number of Chinese characters
200
400
600
800
1000

CLAIM:

$$\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right] =$$

bias² $(y_i - \mathbb{E}_S[h_S(x_i)])^2 +$

variance $+ \mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[(h_S(x_i))])^2]$

USEFUL LEMMA:

$$\mathbb{E}[(\alpha - \mathbb{E}[\alpha])^2] = \mathbb{E}[\alpha^2] + \mathbb{E}[\alpha]^2$$

$$y_i = f(x_i) + \mathcal{N}(0, \sigma^2)$$

$$\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right] =$$

bias² $(y_i - \mathbb{E}_S[h_S(x_i)])^2 +$

variance $+ \mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[(h_S(x_i))])^2]$

$$y_i = f(x_i) + \mathcal{N}(0, \sigma^2)$$

$$\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right] =$$

bias² $(f(x_i) - \mathbb{E}_S[h_S(x_i)])^2 +$

variance $+ \mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[(h_S(x_i))])^2]$

noise $+ \mathbb{E}_S[(f(x_i) - y_i)^2]$

$$y_i = f(x_i) + \mathcal{N}(0, \sigma^2)$$

$$\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right] =$$

bias² $(f(x_i) - \mathbb{E}_S[h_S(x_i)])^2 +$

variance $+ \mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[(h_S(x_i))])^2]$

noise $+ \sigma^2$

$$\mathbb{E}_S\left[(y_i - h_S(x_i))^2\right] =$$

bias² $(y_i - \mathbb{E}_S[h_S(x_i)])^2 +$

variance $+ \mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[(h_S(x_i))])^2]$

BAGGING

revisited

Bagging

Bagging (Boostrap aggregating).

(Breiman, 1996)

```
BAGGING(S = ((x_1, y_1), ..., (x_m, y_m)))
  for t ← 1 to T do
      S_t ← BOOTSTRAP(S) ▷ i.i.d. sampling with replacement from S.
      h_t ← TRAINCLASSIFIER(S_t)
  return h_S = x ↦ MAJORITYVOTE((h_1(x), ..., h_T(x)))
```

Why does it work?

Bagging

Ensemble :

$$h_S(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Bagging : Special case where we fix:

$$\alpha_t = 1 \quad \text{and} \quad h_t = \mathbb{L}(S_t)^*$$

* $\mathbb{L}$ is some learning algorithm

S_t is a training set drawn from distribution $P(< x, y >)$

Bagging

Bagging Ensemble :

$$h_S(x) = \text{sign}\left(\sum_{t=1}^{T} h_t(x)\right)$$

What happens to *bias* and *variance?*

Bagging

Bagging Ensemble (regression) :

$$h_S(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$$

bias^2 $\quad (y_i - \mathbb{E}_S[h_S(x_i)])^2$

variance $\quad \mathbb{E}_S[(h_s(x_i) - \mathbb{E}_S[(h_S(x_i))])^2]$

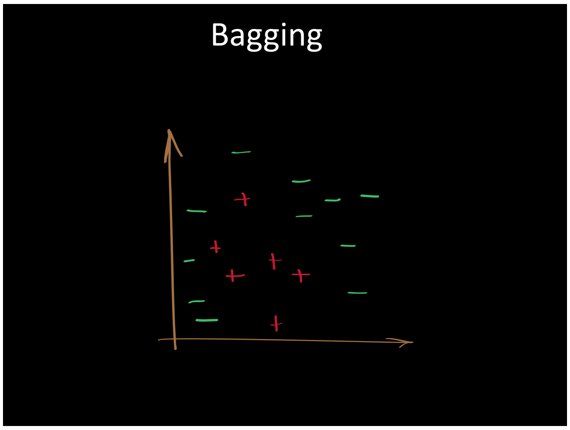
Bagging

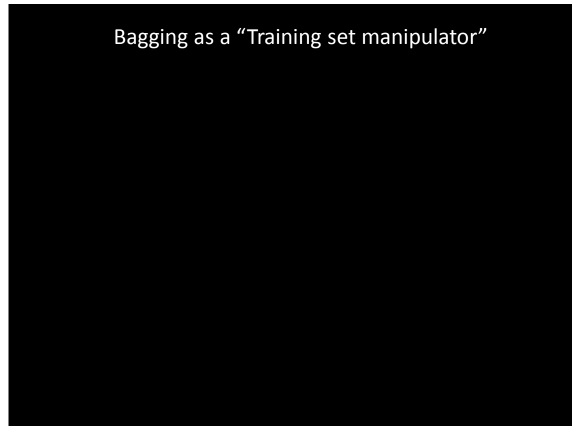
What happens to *bias* and *variance?*

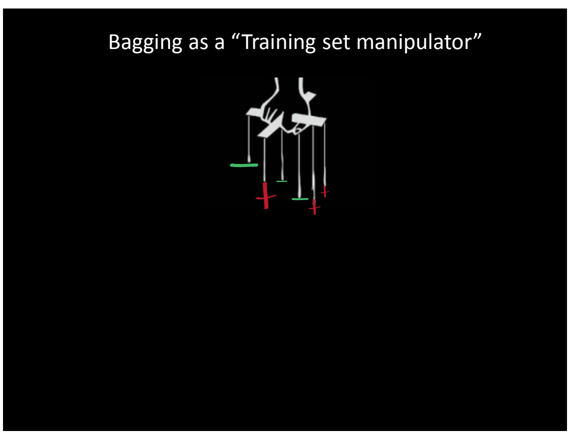
$$\text{Bias}(h_s, x_i) = \frac{1}{T}\sum_{t=1}^{T} \text{Bias}(h_t, x_i)$$

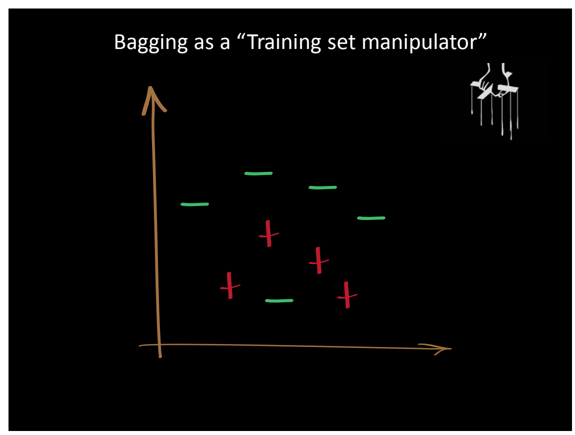
$$\text{Var}(h_s, x_i) \approx \frac{1}{T}\text{Var}(h_1, x_i)$$

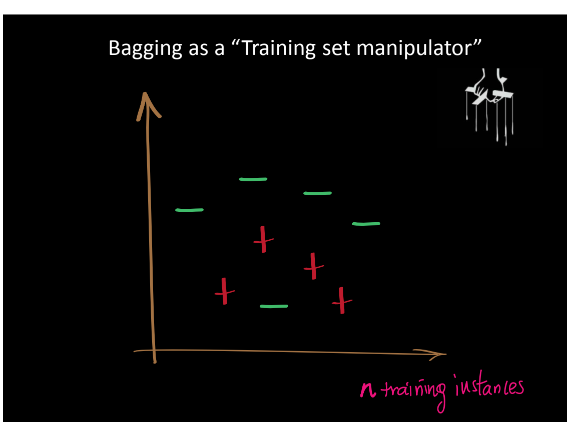
Bagging has approximately the same bias, but reduces variance of individual classifiers!

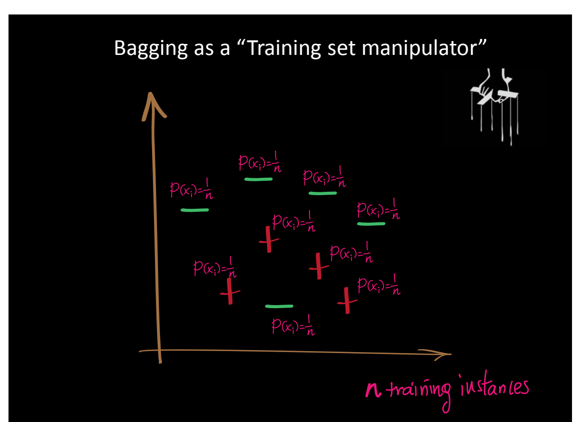

Bagging


Bagging as a “Training set manipulator”

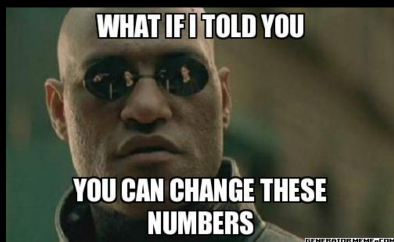

Bagging as a “Training set manipulator”


Bagging as a “Training set manipulator”

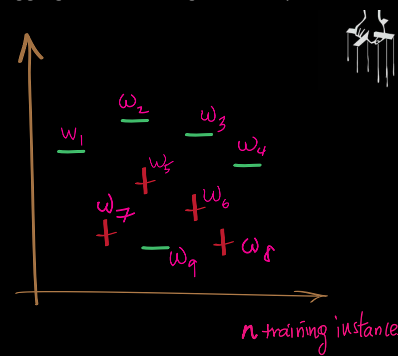

Bagging as a “Training set manipulator”
n training instances


Bagging as a “Training set manipulator”
P(x_i)=1/n
n training instances

Bagging as a "Training set manipulator"



Bagging as a "Training set manipulator"



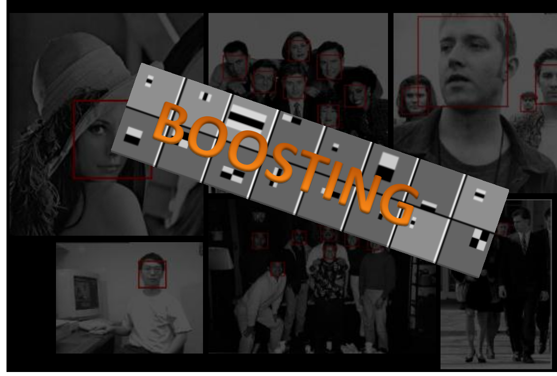
Ensemble

Problem : given *T* binary classification hypotheses (h_1,…, h_T), **find** a combined classifier:

$$h_S(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

with better performance.

Teaser



Hypothetical Algorithm

Hypothetical Algorithm
(incomplete)

Given $x_i \in X, y_i \in Y = \{-1, 1\}$

where $(x_1, y_1), \ldots, (x_n, y_n)$

Initialize $W_1(i) = 1/n$

Initialize set $H = \{h_1, \ldots, h_T\}$

For $t = 1, \ldots, T$:

- Pick hypothesis h_t out of the set H
- Compute error rate ϵ_t of h_t
- Assign new weights W_t to X
- Compute new weight α_t for h_t

Output $h_S(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Hypothetical Algorithm
(incomplete)

Given $x_i \in X, y_i \in Y = \{-1, 1\}$
where $(x_1, y_1), \ldots, (x_n, y_n)$
Initialize $W_1(i) = 1/n$
Learning algorithm $\mathbb{L}$
For $t = 1, \ldots, T$:
- Generate hypothesis h_t with $\mathbb{L}$
- Compute error rate ϵ_t of h_t
- Assign new weights W_t to X
- Compute new weight α_t for h_t

Output $h_S(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Hypothetical Algorithm
(incomplete)

Given $x_i \in X, y_i \in Y = \{-1, 1\}$
where $(x_1, y_1), \ldots, (x_n, y_n)$
Initialize $W_1(i) = 1/n$
Initialize set $H = \{h_1, \ldots, h_T\}$
For $t = 1, \ldots, T$:
- Pick hypothesis h_t out of the set H
- Compute error rate ϵ_t of h_t
- Assign new weights W_t to X
- Compute new weight α_t for h_t

Output $h_S(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Hypothetical Algorithm
(incomplete)

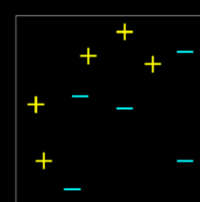
Given $x_i \in X, y_i \in Y = \{-1, 1\}$
where $(x_1, y_1), \ldots, (x_n, y_n)$
Initialize $W_1(i) = 1/n$
Initialize set $H = \{h_1, \ldots, h_T\}$
For $t = 1, \ldots, T$:
- Pick hypothesis h_t out of the set H
- Compute error rate ϵ_t of h_t
- Assign new weights W_t to X
- Compute new weight α_t for h_t

Output $h_S(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Hypothetical Algorithm

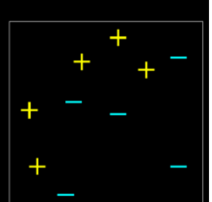
Toy Example

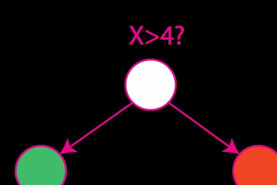
- Positive examples
- Negative examples
- 2-Dimensional plane
- Weak hyps: linear separators
- 3 iterations

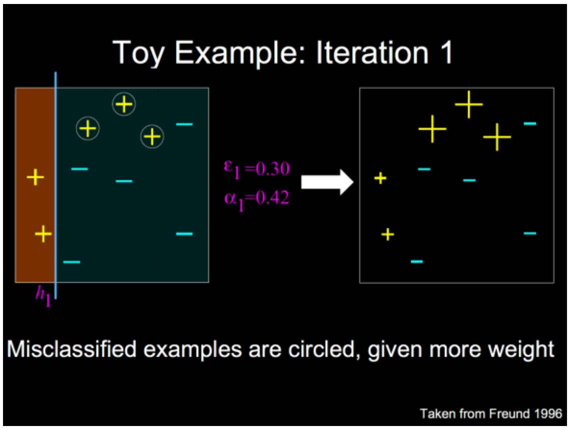


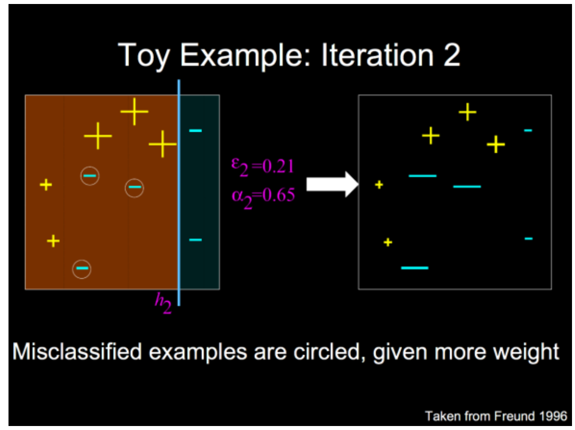
Toy Example

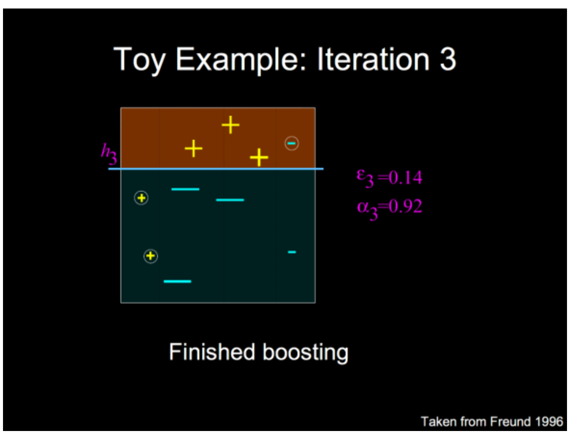
- Positive examples
- Negative examples
- 2-Dimensional plane
- Weak hyps: linear separators
- 3 iterations

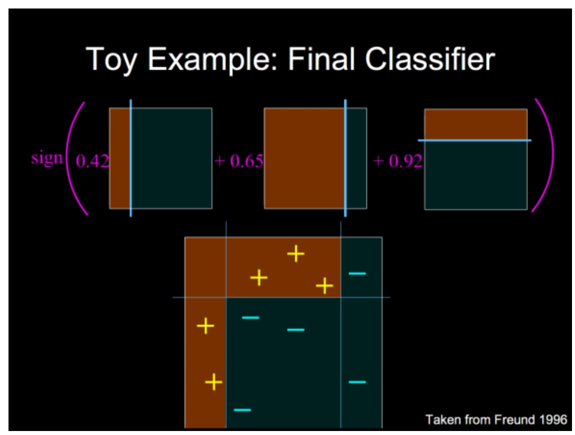












Questions

- Which hypothesis do we choose at every iteration?
- How should we weight the hypotheses?
- How should we weight the examples?

Answers

Choose h_t that maximizes $W_{correct}$ (minimizes ϵ_t)

Choose α_t according to:

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

Update the weight of instance x_i as follows:

$$w_t(i) = w_{t-1}(i) * e^{-\alpha_t} \quad \text{if} \quad y_i = h_t(x_i)$$

$$w_t(i) = w_{t-1}(i) * e^{\alpha_t} \quad \text{if} \quad y_i \neq h_t(x_i)$$

AdaBoost

Hypothetical Algorithm
(incomplete)

Given $x_i \in X, y_i \in Y = \{-1, 1\}$
where $(x_1, y_1), \ldots, (x_n, y_n)$
Initialize $W_1(i) = 1/n$
Initialize set $H = \{h_1, \ldots, h_T\}$
For $t = 1, \ldots, T$:

- Pick hypothesis h_t out of the set H
- Compute error rate ϵ_t of h_t
- Assign new weights W_t to X
- Compute new weight α_t for h_t

Output $h_S(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Hypothetical Algorithm
(incomplete)

Given $x_i \in X, y_i \in Y = \{-1, 1\}$
where $(x_1, y_1), \ldots, (x_n, y_n)$
Initialize $W_1(i) = 1/n$
Initialize set $H = \{h_1, \ldots, h_T\}$
For $t = 1, \ldots, T$:

- Pick hypothesis h_t with smallest ϵ_t
- Compute weight α_t on h_t
- Update weights W_t for X

Output $h_S(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Training Error for AdaBoost

Write for some h_t weighted error ϵ_t as:

$$\epsilon_t = \frac{1}{2} - \gamma_t$$

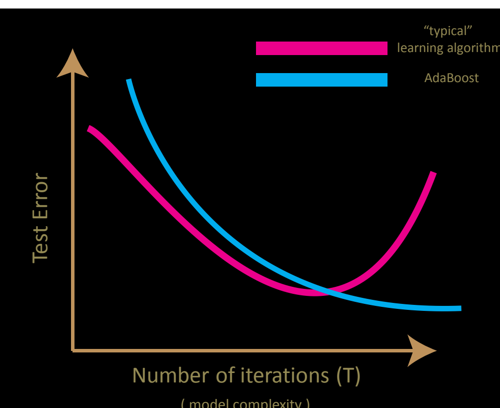
We can then bound the training error:

$$\text{training error} \leq \exp\left(-2T\gamma^2\right)$$

For some γ such that:

$$\gamma_t \geq \gamma > 0$$

What about Generalization Error?



Why?

Margin

$$\text{margin}_f(x,y) =$$

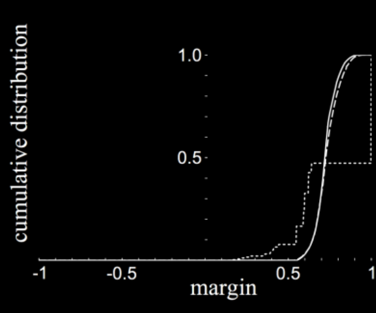
Margin

$$\text{margin}_f(x,y) = \frac{yf(x)}{\sum_t |\alpha_t|} =$$

Margin

$$\text{margin}_f(x,y) = \frac{yf(x)}{\sum_t |\alpha_t|} = \frac{y \sum_t \alpha_t h_t(x)}{\sum_t |\alpha_t|}$$

Margin

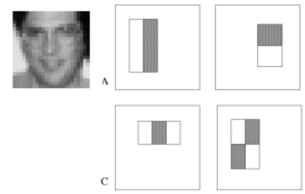


Viola Jones Classifier

Teaser

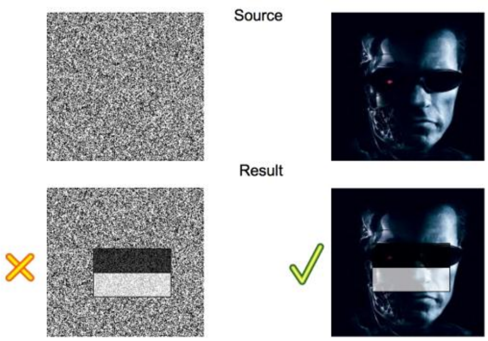
Image Features

"Rectangle filters"



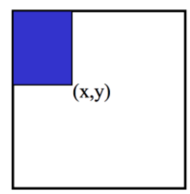
Value =

∑ (pixels in white area) –
∑ (pixels in black area)



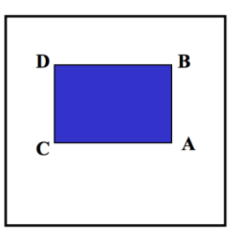
Fast computation with integral images

- The *integral image* computes a value at each pixel (x,y) that is the sum of the pixel values above and to the left of (x,y), inclusive
- This can quickly be computed in one pass through the image

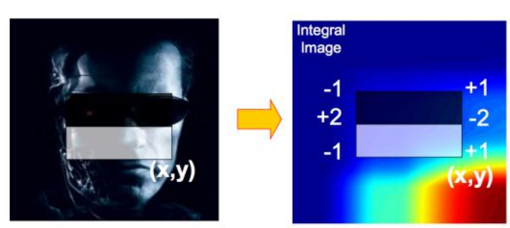


Computing sum within a rectangle

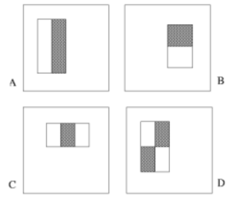
- Let A,B,C,D be the values of the integral image at the corners of a rectangle
- Then the sum of original image values within the rectangle can be computed as:
  sum = A – B – C + D
- Only 3 additions are required for any size of rectangle!
  - This is now used in many areas of computer vision



Example



"Rectangle filters"



Similar to Haar wavelets

Papageorgiou, et al.

$$h_t(x_i) = \begin{cases} \alpha_t & \text{if } f_t(x_i) > \theta_t \\ \beta_t & \text{otherwise} \end{cases}$$

$$C(x) = \theta\left(\sum_t h_t(x) + b\right)$$

60,000 features to choose from