DATA MINING 2
Exercises – Evaluation, NN, Ensemble

Riccardo Guidotti

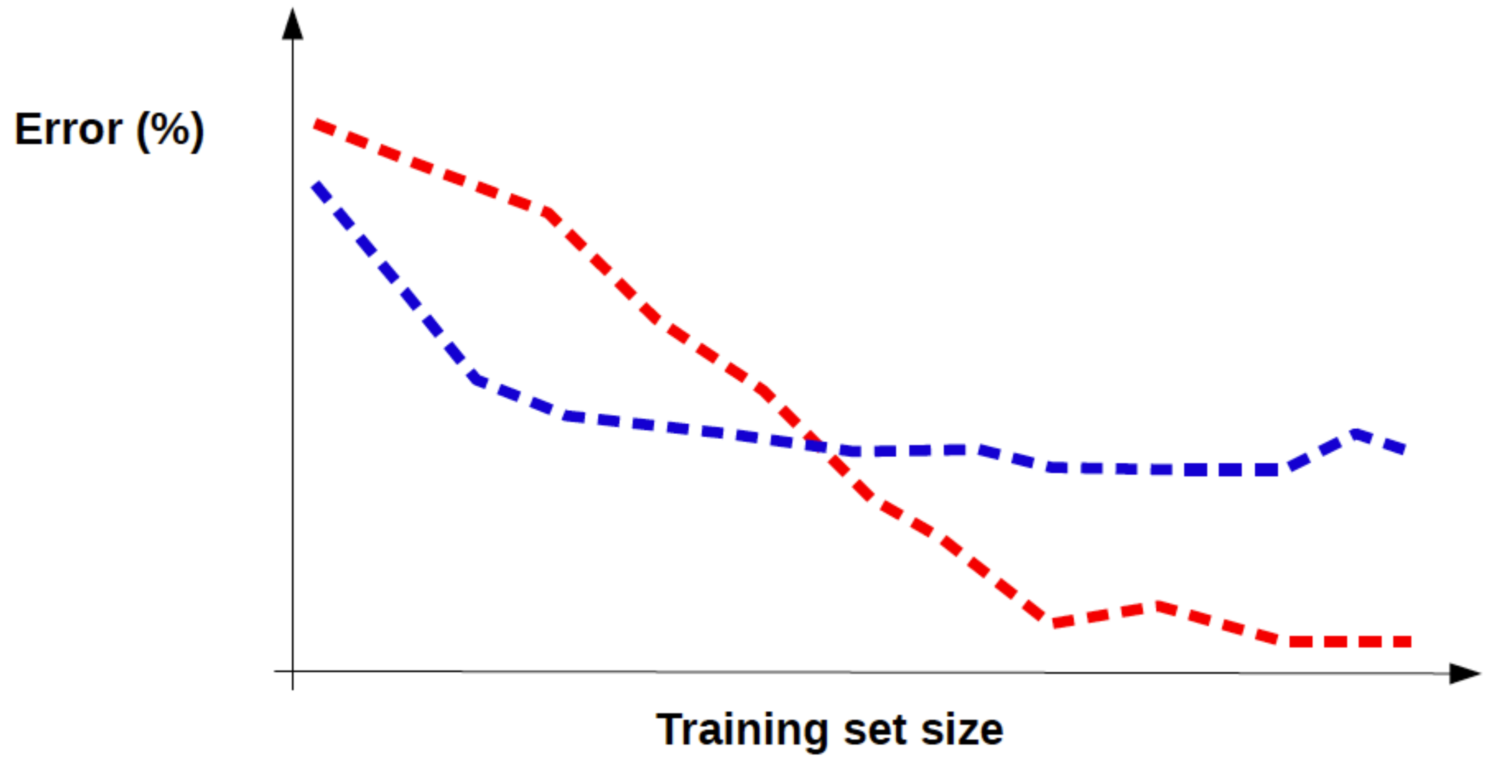
a.a. 2019/2020

UNIVERSITÀ DI PISA

Evaluation

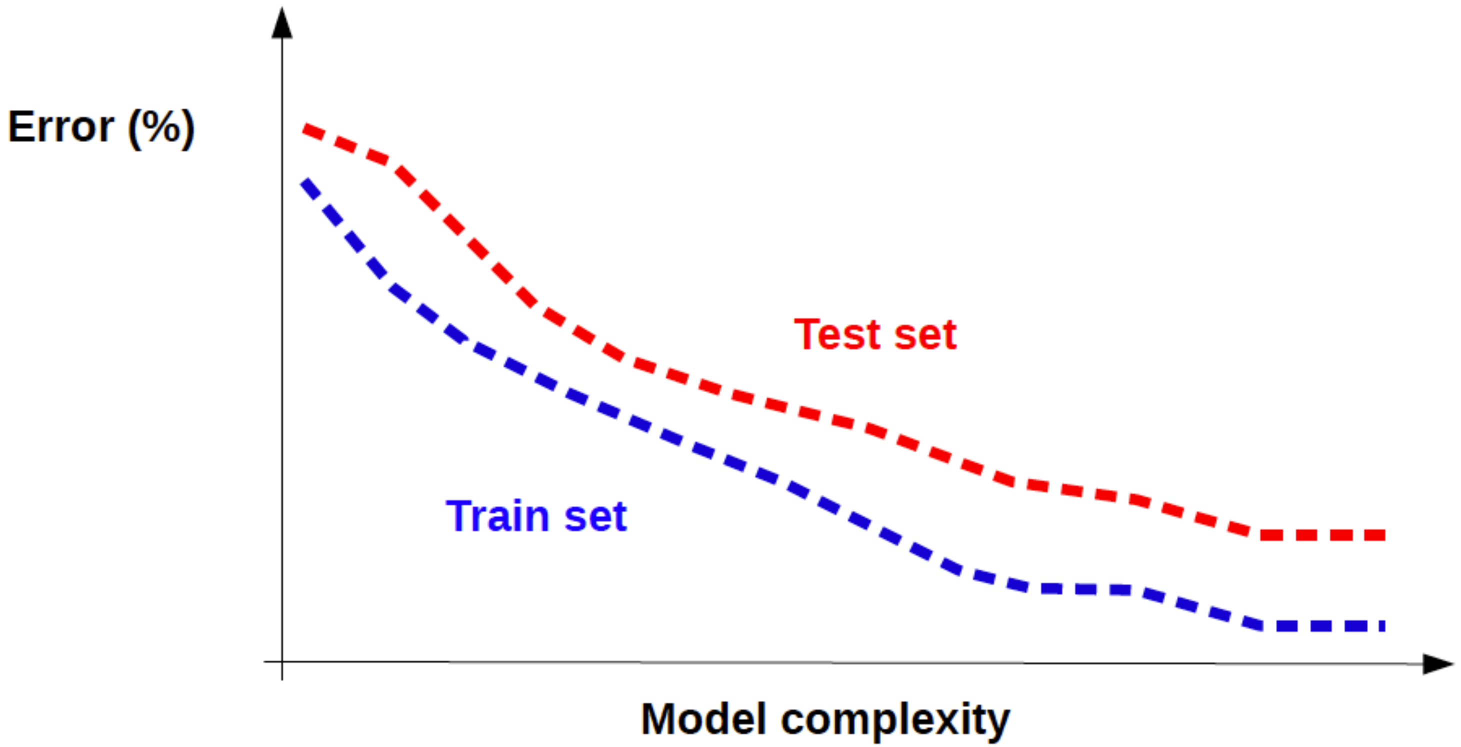
Learning Curves

- Two classification methods produce the following learning curves. What can we conclude about them?



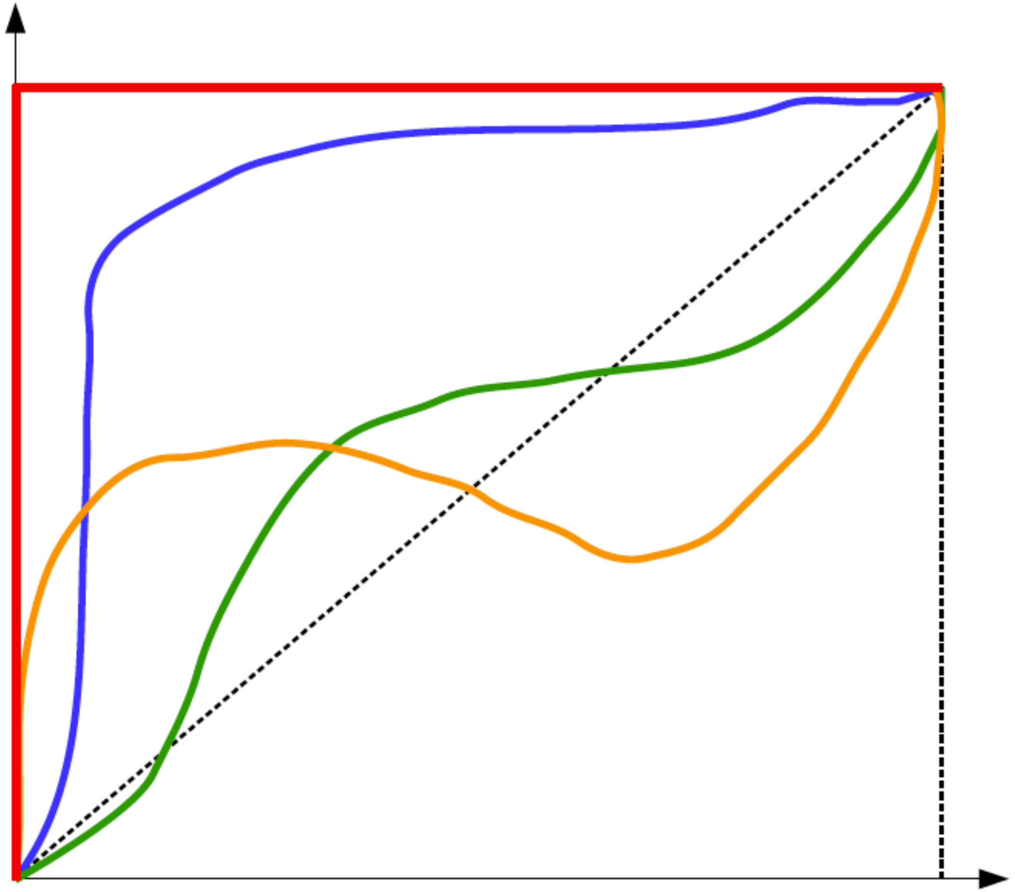
Fitting Graphs

- Our classification method produces the following fitting graph. What can we conclude about the model and/or the dataset?



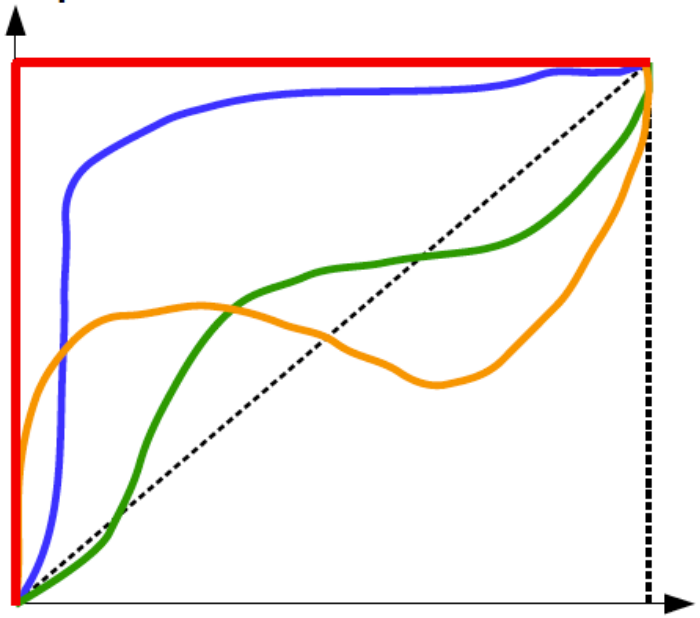
Evaluation Curves

- Which of the following curves could be a ROC?
- Which could be a Lift chart?



Evaluation Curves - Solution

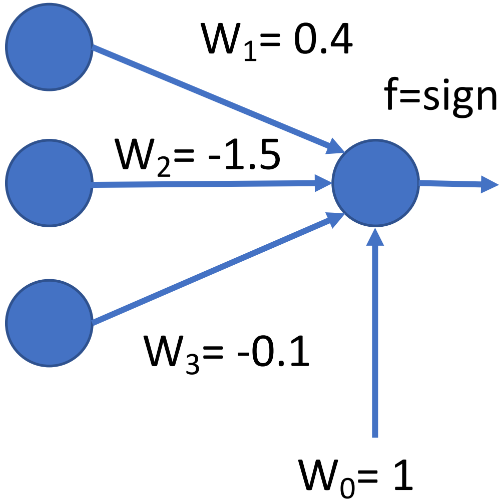
- Which of the following curves could be a ROC?
  - Answer: all, excepted the orange one: TPR and FPR (on the axes) never decrease
- Which could be a Lift chart?
  - Answer: as for ROC, but now also the red one is impossible: you need to classify as positive several records (X axis) to reach 100% of TPR (Y axis)



Neural Networks

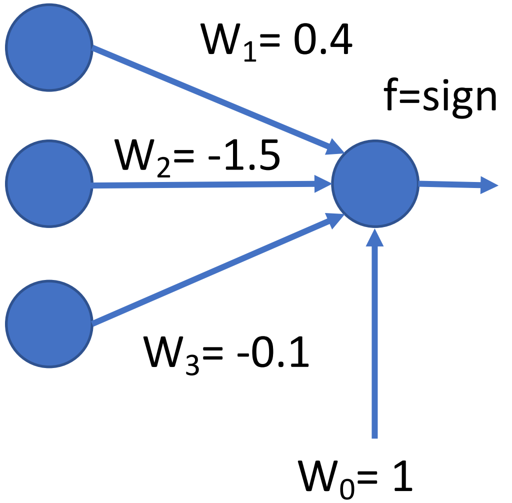
Predict with Perceptron

id	X_1	X_2	X_3	Y
1	0	0	0	
2	1	1	1	
3	1	0	1	
4	0	2	0	



Predict with Perceptron - Solution

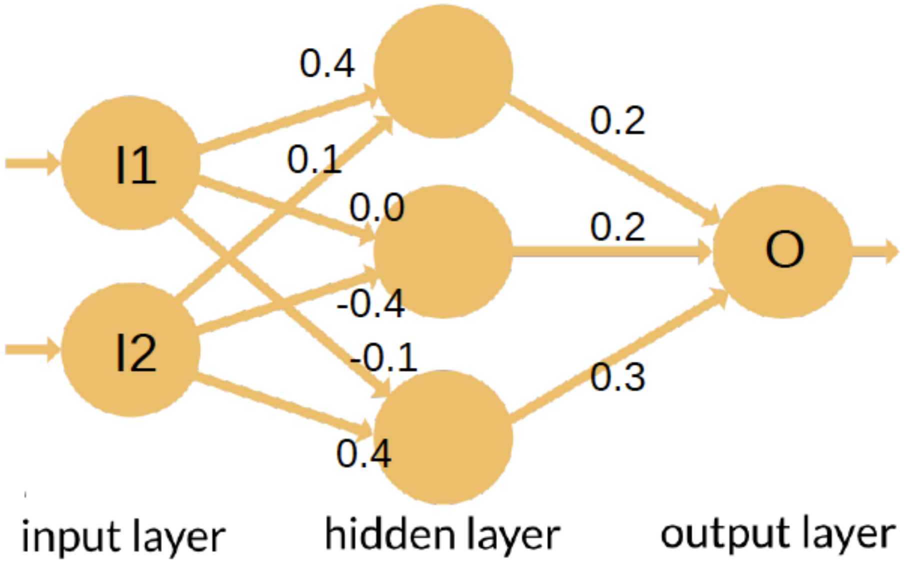
id	X_1	X_2	X_3	Y
1	0	0	0	1
2	1	1	1	-1
3	1	0	1	1
4	0	2	0	-1



Y_1 = sign(1 + 0.4 * 0 + -1.5 * 0 + -0.1 * 0) = sign(1) = 1
Y_2 = sign(1 + 0.4 * 1 + -1.5 * 1 + -0.1 * 1) = sign(-0.2) = -1
Y_3 = sign(1 + 0.4 * 1 + -1.5 * 0 + -0.1 * 0) = sign(1.3) = 1
Y_4 = sign(1 + 0.4 * 0 + -1.5 * 2 + -0.1 * 0) = sign(-2) = -1

Predict with a Neural Network

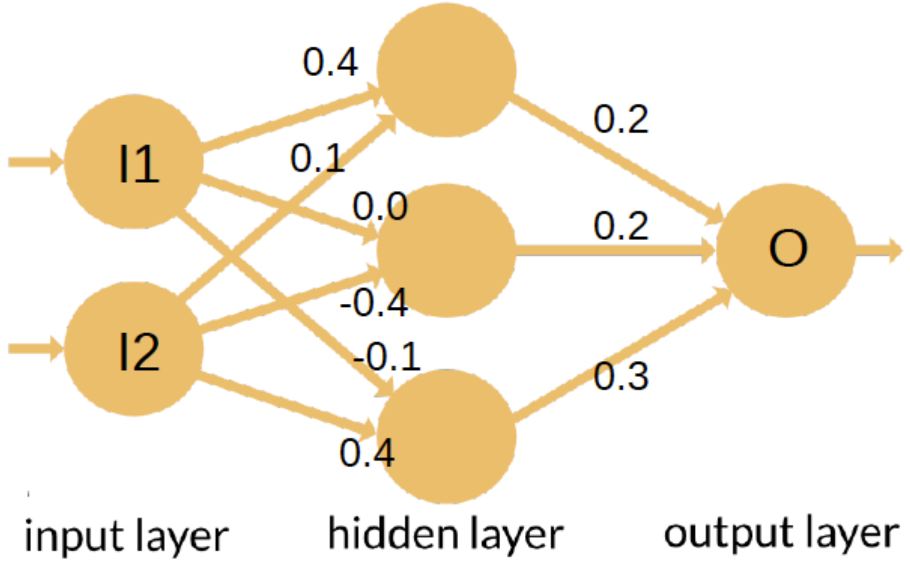
- Given the following NN with
  - assigned weights (see figure)
  - activation function f(S) = sign(S-0.2) for all nodes
- Label the test set on the right, then compute accuracy, and precision & recall for both classes



I1	I2	O
-1	+1	
+1	+1	
+1	-1	
+1	-1	
-1	+1	
+1	+1	
-1	-1	
+1	+1	
-1	-1	
+1	+1	

Predict with a Neural Network - Solution

- Given the following NN with
  - assigned weights (see figure)
  - activation function f(S) = sign(S-0.2) for all nodes
- Label the test set on the right, then compute accurac[y] and precision & recall for both classes



I1	I2	C
-1	+1	-1
+1	+1	+1
+1	-1	-1
+1	-1	+1
-1	+1	+1
+1	+1	+1
-1	-1	-1
+1	+1	-1
-1	-1	-1
+1	+1	+1

H_1 = sign(0.4 * -1 + 0.1 * 1 -0.2) =
= sign(-0.5) = -1

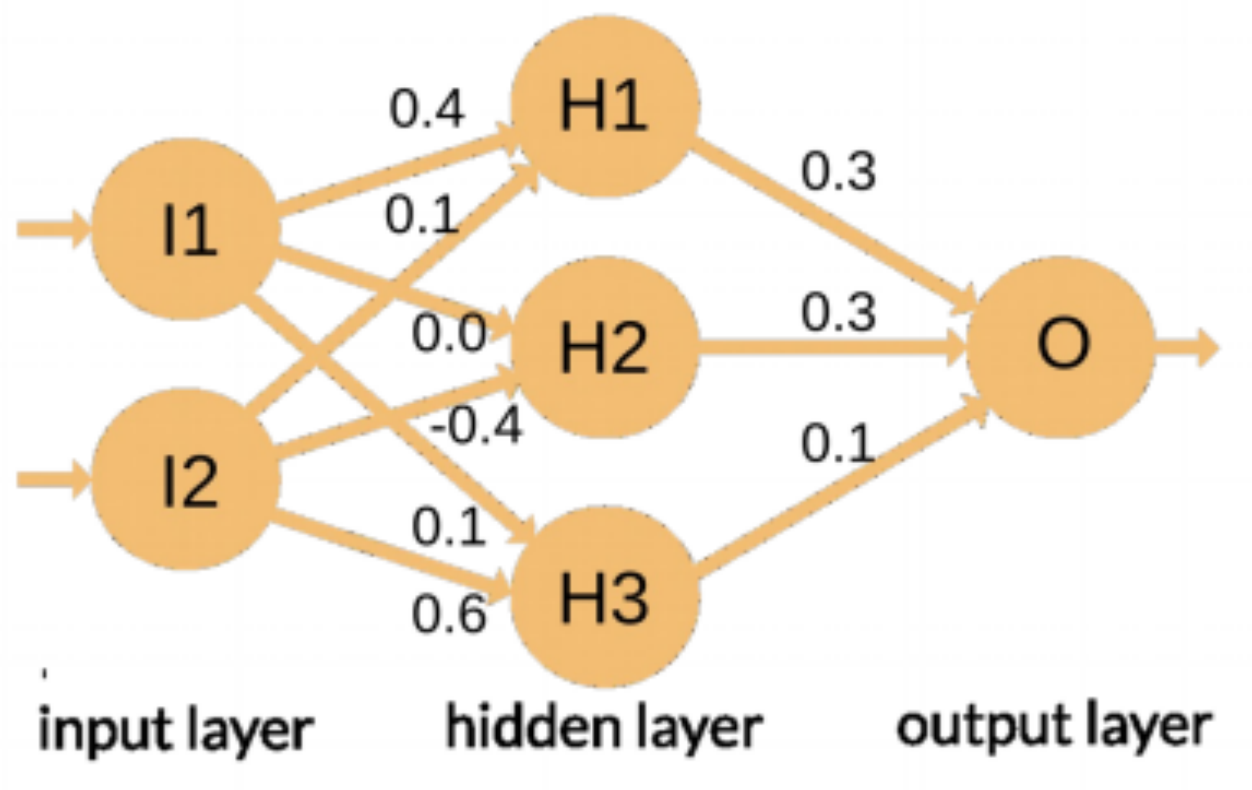
H_2 = sign(0.0 * -1 + -0.4 * 1 -0.2) =
= sign(-0.6) = -1

H_3 = sign(-0.1 * -1 + 0.4 * 1 -0.2) =
= sign(0.3) = 1

Y_1 = sign(0.2 * -1 + 0.2* -1 + 0.3 * 1 -0.2) =
= sign(-0.3) = -1

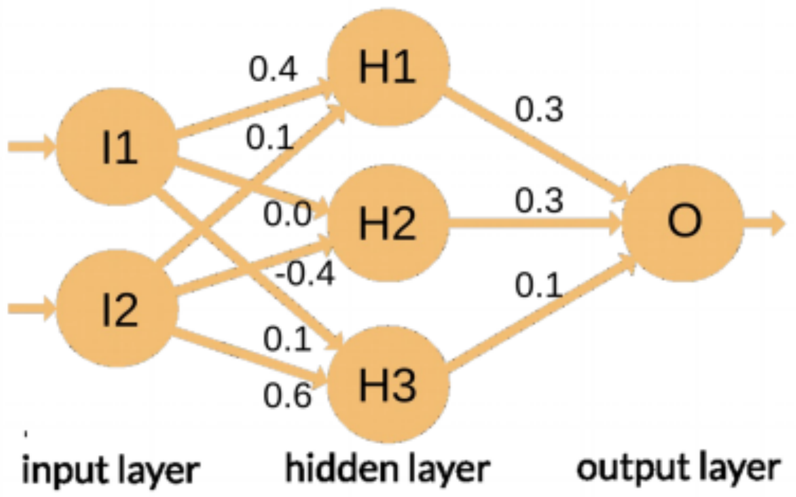
Predict with a Neural Network

Given the neural network below (on the left), apply it to the test set provided (on the right). The weights are reported beside each connection, while the activation function is simply f(S) = sign(S), i.e. -1 for positive values, +1 for positive ones and 0 for S=0. For each case, show the output also of the nodes on the hidden layer.



I1	I2	O
+0	-1	
+1	+0	
-1	+1	
+1	+1	
+1	-1	

Predict with a Neural Network - Solution



Answer:

I1	I2	O
+0	-1	-1
+1	+0	+1
-1	+1	-1
+1	+1	+1
+1	-1	+1

Input1	0	1	-1	1	1
Input2	-1	0	1	1	-1
H1	-1	1	-1	1	1
H2	1	0	-1	-1	1
H3	-1	1	1	1	-1
Output	-1	1	-1	1	1

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0						
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1					
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1				
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0			
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0		
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1								
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0							
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0						
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1					
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1				
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1	2			
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1	2	0.6		
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1	2	0.6	0	
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1	2	0.6	0	0
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1	2	0.6	0	0
3	-0.4								
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1	2	0.6	0	0
3	-0.4	0	0						
4									
5									
6									
7									
8									
9									
10									
11									
12									

Train Linear Perceptron - Solution

id	X_1	X_2	Y
a	0	1	-1
b	0	0	1
c	1	2	-1

Lambda = 0.3

f = sign

it	W_0	W_1	W_2	X.W	f(X.W)	error	$delta_0$	$delta_1$	$delta_2$
1	-1	0	0	-1	-1	0	0	0	0
2	-1	0	0	-1	-1	2	0,6	0	0
3	-0,4	0	0	-0,4	-1	0	0	0	0
4	-0,4	0	0	-0,4	-1	0	0	0	0
5	-0,4	0	0	-0,4	-1	2	0,6	0	0
6	0,2	0	0	0,2	1	-2	-0,6	-0,6	-1,2
7	-0,4	-0,6	-1,2	-1,6	-1	0	0	0	0
8	-0,4	-0,6	-1,2	-0,4	-1	2	0,6	0	0
9	0,2	-0,6	-1,2	-2,8	-1	0	0	0	0
10	0,2	-0,6	-1,2	-1	-1	0	0	0	0
11	0,2	-0,6	-1,2	0,2	1	0	0	0	0
12	0,2	-0,6	-1,2	-2,8	-1	0	0	0	0

Ensemble

Ensemble

- We have 3 independent models for the same data, with poor performances
  - Error1 = 45%
  - Error2 = 40%
  - Error3 = 35%
- Is it better to use Model3 alone or to make bagging with all the three models?

Ensemble - Solution

- TODO: compute the probability of error of the ensemble
- Standard formula (case for 25 models):

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

- Implicitly enumerates all cases with more errors than correct answers (i >= 13 errors against 25-i <=12 correct ones)
  - However, it works only when all models have the same error ε
- Here we have to explicitly enumerate all cases

Ensemble - Solution

- Probability of success ✔ and failure ✖ of each:
  - Model1 = ✖ .45 ✔ .55
  - Model2 = ✖ .40 ✔ .60
  - Model3 = ✖ .35 ✔ .65

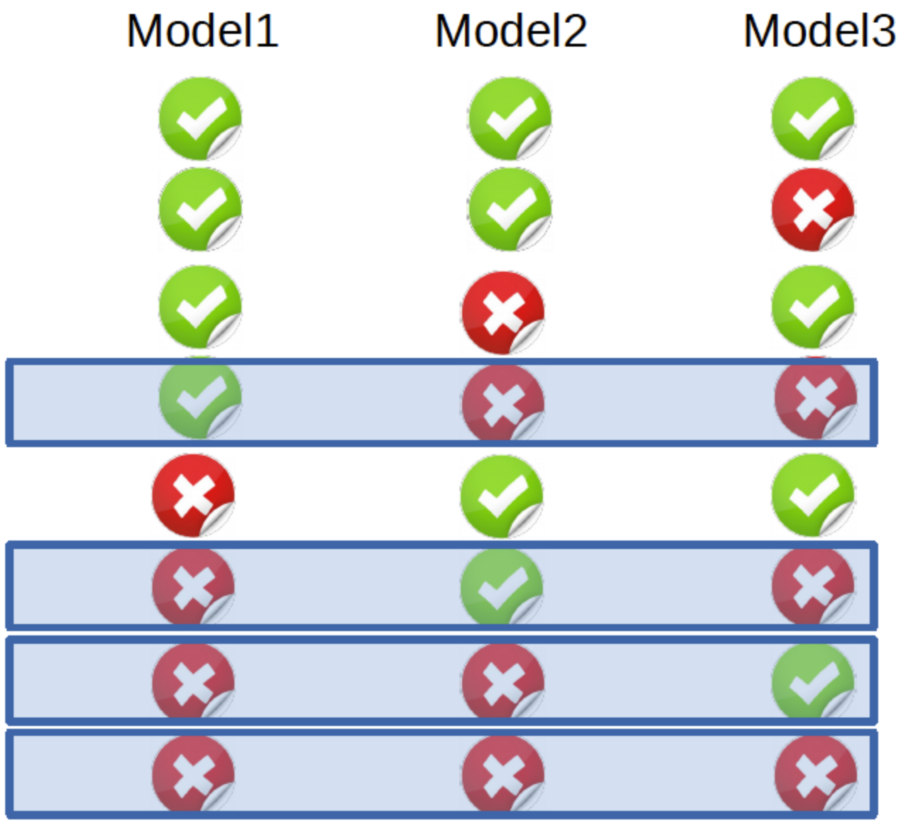
Ensemble - Solution

- We have 8 possible cases

Model1	Model2	Model3
✓	✓	✓
✓	✓	✗
✓	✗	✓
✓	✗	✗
✗	✓	✓
✗	✓	✗
✗	✗	✓
✗	✗	✗

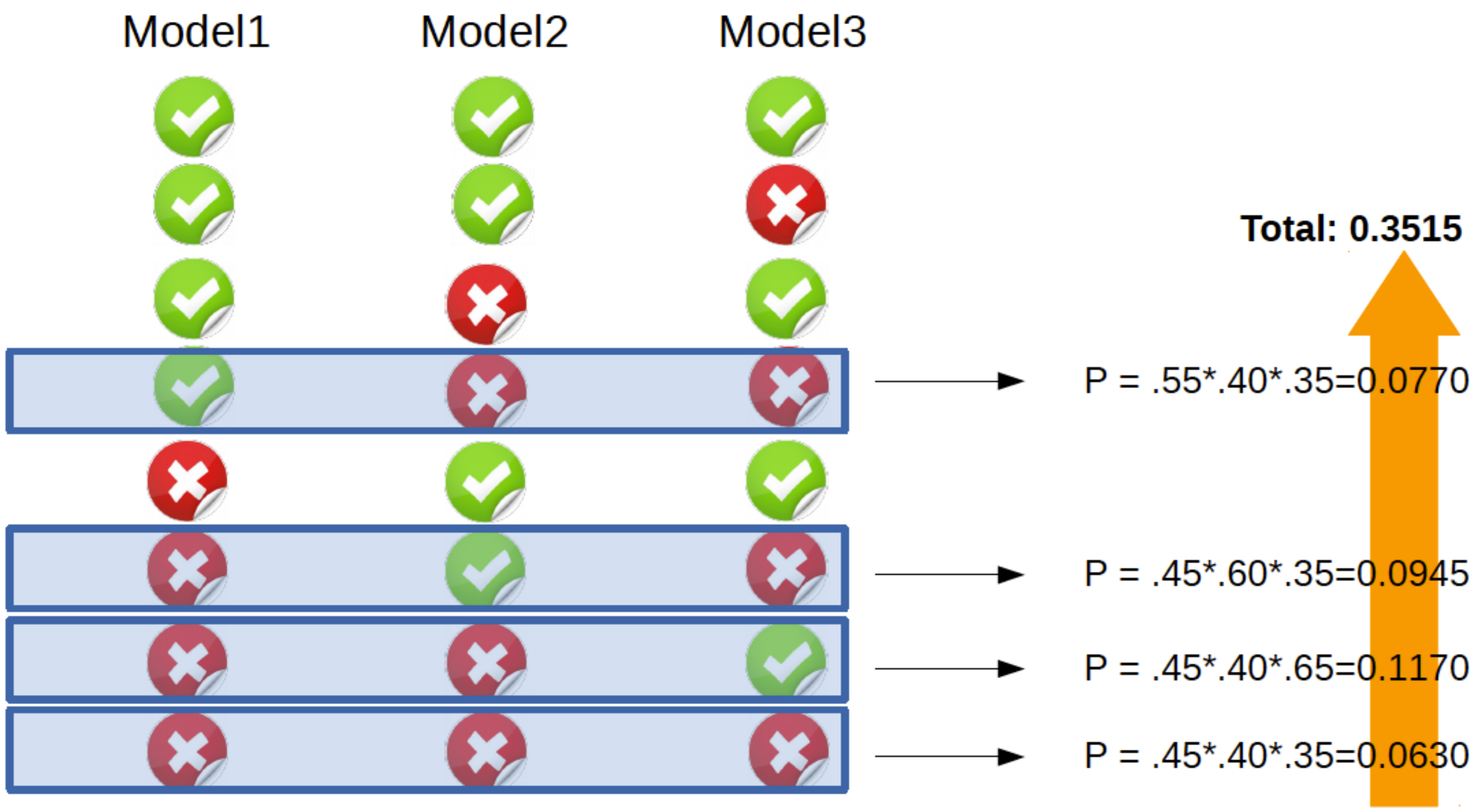
Ensemble - Solution

- We have 8 possible cases



Ensemble - Solution

- We have 8 possible cases



- Probability of success and failure of each:
  - Model1 = .45 .55
  - Model2 = .40 .60
  - Model3 = .35 .65

Ensemble - Solution

- Outcome:
  - The “expert” model (Model3) has 35% of error
  - The bagging model has 35.15% of error
  - In this specific case Bagging is not better than the “expert” alone...

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

1. Find the best split for AdaBoost
2. Which is the value of the importance (alpha) for the best split?
3. Which are the values of the new weights?
4. Which are the values of the new weights normalized?
5. Provide a correct training for the next iteration based on the new weights.

Gain Function = Misclassification Error

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

State	Y	N
Italy	2/3	1/3
France	1/2	1/2
Germ	1/2	1/2

Gain Function = Misclassification Error

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

State	Y	N
Italy	2/3	1/3
France	1/2	1/2
Germ	1/2	1/2

Delta Gain = 3/7*1/3 + 2/7*1/2 + 2/7*1/2 = 3/7

Gain Function = Misclassification Error

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

State	Y	N
Italy	2/3	1/3
France	1/2	1/2
Germ	1/2	1/2

Delta Gain = 3/7*1/3 + 2/7*1/2 + 2/7*1/2 = 3/7

Travel	Y	N
Yes	3/4	1/4
No	1/3	2/3

Delta Gain = 4/7*1/4 + 3/7*1/3 = 2/7

Gain Function = Misclassification Error

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

Gain Function = Misclassification Error

<u>State</u>	Y	N
Italy	<u>2/3</u>	1/3
France	<u>1/2</u>	1/2
Germ	<u>1/2</u>	1/2

Delta Gain = 3/7*1/3 + 2/7*1/2 + 2/7*1/2 = 3/7

<u>Travel</u>	Y	N
Yes	<u>3/4</u>	1/4
No	1/3	<u>2/3</u>

Delta Gain = 4/7*1/4 + 3/7*1/3 = 2/7

<u>Sex</u>	Y	N
M	<u>2/4</u>	2/4
F	<u>2/3</u>	1/3

Delta Gain = 4/7*2/4 + 3/7*1/3 = 3/7

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

Gain Function = Misclassification Error

State	Y	N
Italy	2/3	1/3
France	1/2	1/2
Germ	1/2	1/2

Delta Gain = 3/7*1/3 + 2/7*1/2 + 2/7*1/2 = 3/7

Travel	Y	N
Yes	3/4	1/4
No	1/3	2/3

Delta Gain = 4/7*1/4 + 3/7*1/3 = 2/7

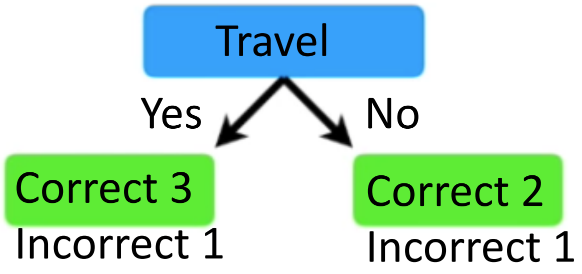
Sex	Y	N
M	2/4	2/4
F	2/3	1/3

Delta Gain = 4/7*2/4 + 3/7*1/3 = 3/7

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

Split by Travel



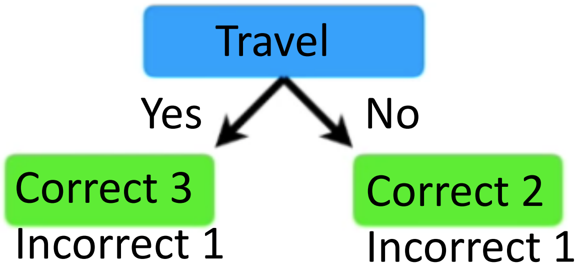
Error = 2/7

Gain Function = Misclassification Error

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7		
Germ	Yes	M	Y	1/7		
Italy	No	M	N	1/7		
Italy	No	F	Y	1/7		
France	Yes	M	Y	1/7		
Germ	No	F	N	1/7		
France	Yes	M	N	1/7		

Split by Travel



Error = 2/7

Gain Function = Misclassification Error

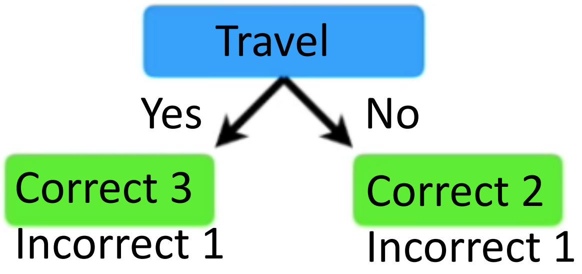
Alpha = 1/2 * ln((1-2/7)/(2/7)) = 1/2 ln(5/2) = 0.458

AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7	0.63	
Germ	Yes	M	Y	1/7	0.63	
Italy	No	M	N	1/7	0.63	
Italy	No	F	Y	1/7	1.58	
France	Yes	M	Y	1/7	0.63	
Germ	No	F	N	1/7	0.63	
France	Yes	M	N	1/7	1.58	

Gain Function = Misclassification Error

Split by Travel



Error = 2/7

Alpha = 1/2 * ln((1-2/7)/(2/7)) = 1/2 ln(5/2) = 0.458

1/7 * $e^{0.458}$ = 1.58 misclassified

1/7 * $e^{-0.458}$ = 0.63 correctly classified

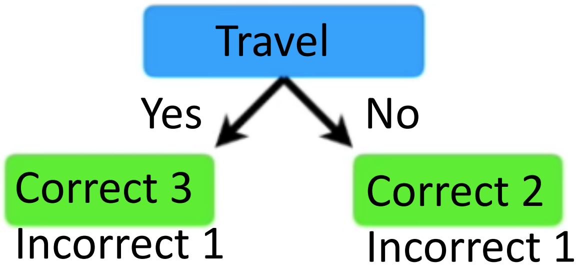
AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7	0.09	
Germ	Yes	M	Y	1/7	0.09	
Italy	No	M	N	1/7	0.09	
Italy	No	F	Y	1/7	0.23	
France	Yes	M	Y	1/7	0.09	
Germ	No	F	N	1/7	0.09	
France	Yes	M	N	1/7	0.23	

Z = 0.91

Gain Function = Misclassification Error

Split by Travel



Error = 2/7

Alpha = 1/2 * ln((1-2/7)/(2/7)) = 1/2 ln(5/2) = 0.458

$1/7 * e^{0.458}$ = 0.23 misclassified

$1/7 * e^{-0.458}$ = 0.09 correctly classified

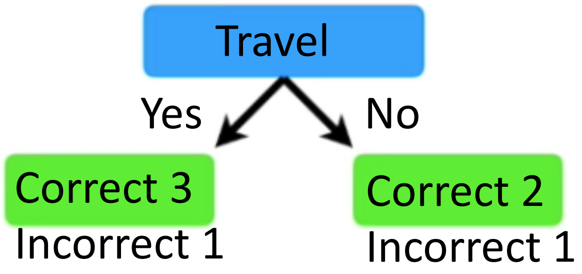
AdaBoost

state	travel	sex	class	weight	new weight	norm weight
Italy	Yes	F	Y	1/7	0.09	0.09
Germ	Yes	M	Y	1/7	0.09	0.09
Italy	No	M	N	1/7	0.09	0.09
Italy	No	F	Y	1/7	0.23	0.25
France	Yes	M	Y	1/7	0.09	0.09
Germ	No	F	N	1/7	0.09	0.09
France	Yes	M	N	1/7	0.23	0.25

Z = 0.91

Gain Function = Misclassification Error

Split by Travel



Error = 2/7

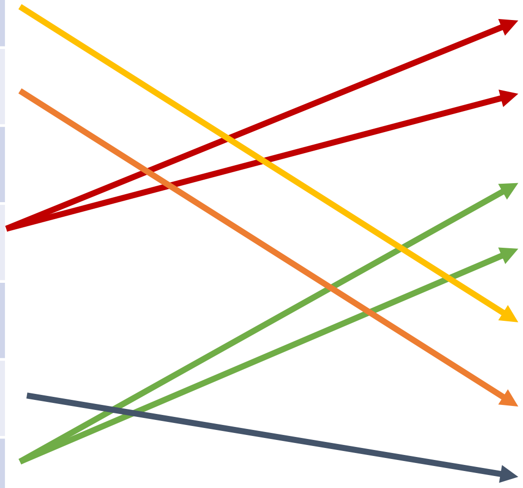
Alpha = 1/2 * ln((1-2/7)/(2/7)) = 1/2 ln(5/2) = 0.458

1/7 * $e^{0.458}$ = 0.23 misclassified

1/7 * $e^{-0.458}$ = 0.09 correctly classified

AdaBoost

state	travel	sex	class	weight
Italy	Yes	F	Y	0.09
Germ	Yes	M	Y	0.09
Italy	No	M	N	0.09
Italy	No	F	Y	0.25
France	Yes	M	Y	0.09
Germ	No	F	N	0.09
France	Yes	M	N	0.25



state	travel	sex	class	weight
Italy	No	F	Y	1/7
Italy	No	F	Y	1/7
France	Yes	M	N	1/7
France	Yes	M	N	1/7
Italy	Yes	F	Y	1/7
Germ	Yes	M	Y	1/7
Germ	No	F	N	1/7

AdaBoost - Classify

state	travel	sex	class
Italy	Yes	F	
Germ	Yes	M	
France	No	M	

If travel = Yes then Y else N - alpha = 0.46
If sex = F then Y else N - alpha = 0.53
If state = Germ then Y else N - alpha = 0.32

AdaBoost - Classify

state	travel	sex	class
Italy	Yes	F	Y
Germ	Yes	M	Y
France	No	M	N

1. N 0.32 Y 0.46 Y 0.53 -> Y
2. N 0.53 Y 0.32 + Y 0.46 -> Y
3. N 0.32 + N 0.46 + N 0.53 -> N

If travel = Yes then Y else N - alpha = 0.46
If sex = F then Y else N - alpha = 0.53
If state = Germ then Y else N - alpha = 0.32