



# **Principles of Artificial Intelligence(305450)**

## **Lecture 10: Machine Learning III**



# Naïve Bayes

- Founded on Bayes' rule for probability inference (Thomas Bayes)
- Update probability of hypotheses based on evidence
- Choose hypothesis with the maximum probability after the evidence has been incorporated
- Algorithm is particularly useful for domains with lots of features

# Example

$f_1$	$f_2$	$f_3$	$f_4$	$Y$
0	1	1	0	1
0	0	1	1	1
1	0	1	0	1
0	0	1	1	1
0	0	0	0	1
1	0	0	1	0
1	1	0	1	0
1	0	0	0	0
1	1	0	1	0
1	0	1	1	0

- $R_1(1,1)=1/5$ : fraction of all **positive** examples that have feature 1 **on**
- $R_1(0,1)=4/5$ : fraction of all **positive** examples that have feature 1 **off**

- We're going to try to characterize, for each feature individually, how it is related to the class of an example.
- First, we look at the positive examples, and count up what fraction of them have feature 1 on and what fraction have feature 1 off. We'll call these fractions  $R_1(1, 1)$  and  $R_1(0,1)$ .
- We can see here that most positive examples have this feature 1 off.

# Example

$f_1$	$f_2$	$f_3$	$f_4$	$y$
0	1	1	0	1
0	0	1	1	1
1	0	1	0	1
0	0	1	1	1
0	0	0	0	1
1	0	0	1	0
1	1	0	1	0
1	0	0	0	0
1	1	0	1	0
1	0	1	1	0

- $R_1(1,1)=1/5$ : fraction of all **positive** examples that have feature 1 **on**
- $R_1(0,1)=4/5$ : fraction of all **positive** examples that have feature 1 **off**
- $R_1(1,0)=5/5$ : fraction of all **negative** examples that have feature 1 **on**
- $R_1(0,0)=0/5$ : fraction of all **negative** examples that have feature 1 **off**

- Now, we look at the negative examples, and figure out what fraction of negative examples have feature 1 on and what fraction have it off. We call these fractions  $R_1(1, 0)$  and  $R_1(0, 0)$ .
- Here we see that **all** negative examples have this feature on.

# Example

$f_1$	$f_2$	$f_3$	$f_4$	$y$
0	1	1	0	1
0	0	1	1	1
1	0	1	0	1
0	0	1	1	1
0	0	0	0	1
1	0	0	1	0
1	1	0	1	0
1	0	0	0	0
1	1	0	1	0
1	0	1	1	0

$$R_1(1,1)=1/5 \quad R_1(0,1)=4/5$$

$$R_1(1,0)=5/5 \quad R_1(0,0)=0/5$$

$$R_2(1,1)=1/5 \quad R_2(0,1)=4/5$$

$$R_2(1,0)=2/5 \quad R_2(0,0)=3/5$$

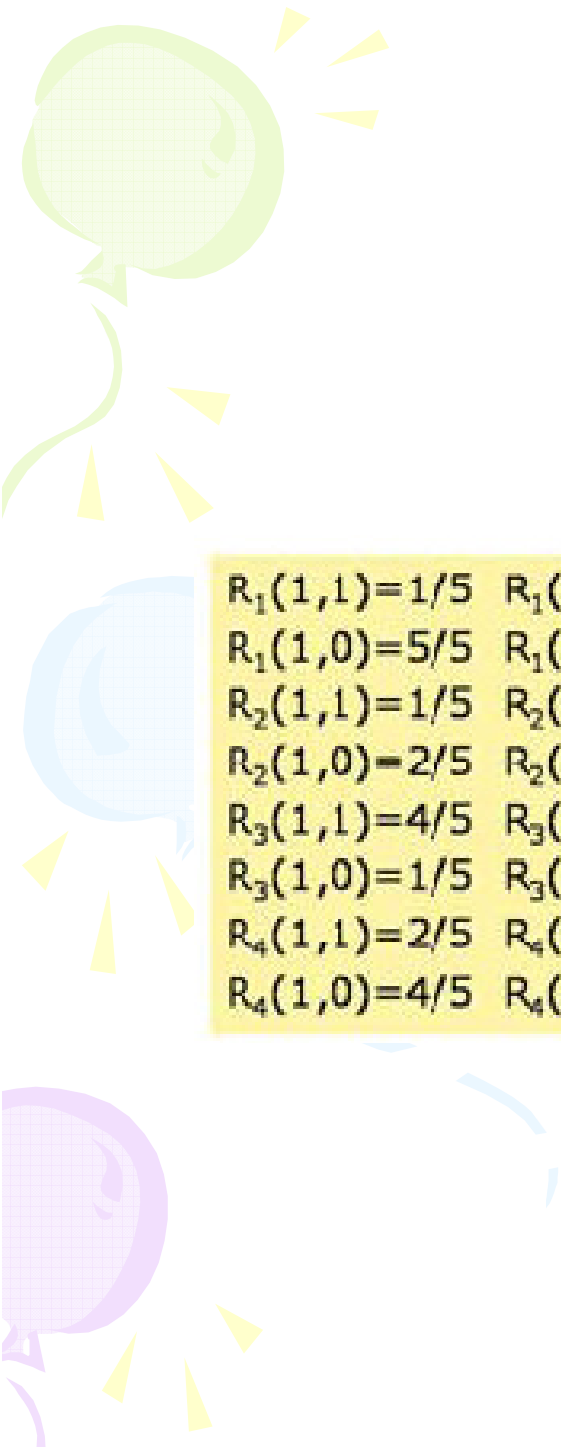
$$R_3(1,1)=4/5 \quad R_3(0,1)=1/5$$

$$R_3(1,0)=1/5 \quad R_3(0,0)=4/5$$

$$R_4(1,1)=2/5 \quad R_4(0,1)=3/5$$

$$R_4(1,0)=4/5 \quad R_4(0,0)=1/5$$

# Prediction



$R_1(1,1)=1/5$	$R_1(0,1)=4/5$
$R_1(1,0)=5/5$	$R_1(0,0)=0/5$
$R_2(1,1)=1/5$	$R_2(0,1)=4/5$
$R_2(1,0)=2/5$	$R_2(0,0)=3/5$
$R_3(1,1)=4/5$	$R_3(0,1)=1/5$
$R_3(1,0)=1/5$	$R_3(0,0)=4/5$
$R_4(1,1)=2/5$	$R_4(0,1)=3/5$
$R_4(1,0)=4/5$	$R_4(0,0)=1/5$

- These R values actually represent our hypothesis, which means that, given a new input  $x$ , we can use the R values to compute an output value  $Y$ .

# Prediction

$$\begin{array}{ll} R_1(1,1)=1/5 & R_1(0,1)=4/5 \\ R_1(1,0)=5/5 & R_1(0,0)=0/5 \\ R_2(1,1)=1/5 & R_2(0,1)=4/5 \\ R_2(1,0)=2/5 & R_2(0,0)=3/5 \\ R_3(1,1)=4/5 & R_3(0,1)=1/5 \\ R_3(1,0)=1/5 & R_3(0,0)=4/5 \\ R_4(1,1)=2/5 & R_4(0,1)=3/5 \\ R_4(1,0)=4/5 & R_4(0,0)=1/5 \end{array}$$

- New  $x = \langle 0, 0, 1, 1 \rangle$
- $S(1) = R_1(0,1) * R_2(0,1) * R_3(1,1) * R_4(1,1) = .205$

- Assume that we get a new  $x = \langle 0, 0, 1, 1 \rangle$ . We start out by computing a "score" for this example being a positive example.
- Each of the factors in the score represents the degree to which this feature tends to have this value in positive examples. Multiplied all together, they give us a measure of how likely it is that this example is positive.

# Prediction

$$\begin{array}{ll} R_1(1,1)=1/5 & R_1(0,1)=4/5 \\ R_1(1,0)=5/5 & R_1(0,0)=0/5 \\ R_2(1,1)=1/5 & R_2(0,1)=4/5 \\ R_2(1,0)=2/5 & R_2(0,0)=3/5 \\ R_3(1,1)=4/5 & R_3(0,1)=1/5 \\ R_3(1,0)=1/5 & R_3(0,0)=4/5 \\ R_4(1,1)=2/5 & R_4(0,1)=3/5 \\ R_4(1,0)=4/5 & R_4(0,0)=1/5 \end{array}$$

- New  $x = \langle 0, 0, 1, 1 \rangle$
- $S(1) = R_1(0,1) * R_2(0,1) * R_3(1,1) * R_4(1,1) = .205$
- $S(0) = R_1(0,0) * R_2(0,0) * R_3(1,0) * R_4(1,0) = 0$

- We can do the same thing to compute a score for  $x$  being a negative example.
- Here, we have  $R_1$  of  $0, 0$  equal to  $0$ . We've never seen a negative example with feature 1 off, so we have concluded, essentially, that it's impossible for that to happen.



# Prediction

- $R_1(1,1)=1/5$   $R_1(0,1)=4/5$
- $R_1(1,0)=5/5$   $R_1(0,0)=0/5$
- $R_2(1,1)=1/5$   $R_2(0,1)=4/5$
- $R_2(1,0)=2/5$   $R_2(0,0)=3/5$
- $R_3(1,1)=4/5$   $R_3(0,1)=1/5$
- $R_3(1,0)=1/5$   $R_3(0,0)=4/5$
- $R_4(1,1)=2/5$   $R_4(0,1)=3/5$
- $R_4(1,0)=4/5$   $R_4(0,0)=1/5$

- New  $x = \langle 0, 0, 1, 1 \rangle$
- $S(1) = R_1(0,1) * R_2(0,1) * R_3(1,1) * R_4(1,1) = .205$
- $S(0) = R_1(0,0) * R_2(0,0) * R_3(1,0) * R_4(1,0) = 0$
- $S(1) > S(0)$ , so predict class 1

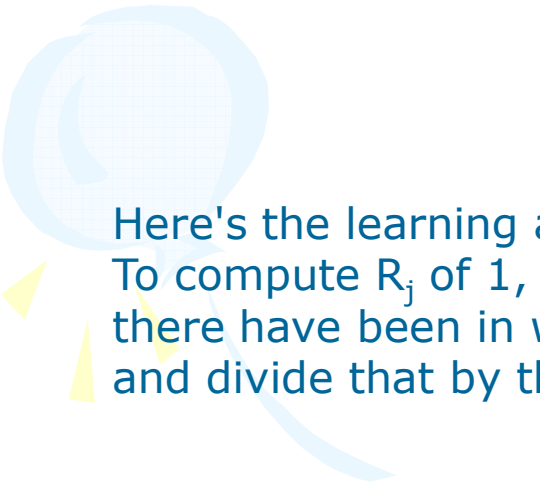
Finally, we compare score 1 to score 0, and generate output 1 because score 1 is larger than score 0.



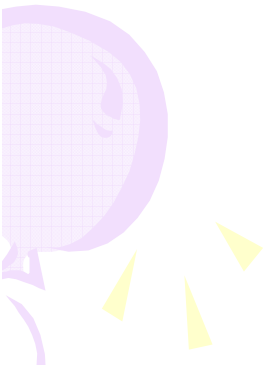
# Learning Algorithm

- Estimate from the data, for all  $j$ :

$$R_j(1,1) = \frac{\#(x_j = 1 \wedge y' = 1)}{\#(y' = 1)}$$



Here's the learning algorithm written out just a little bit more generally. To compute  $R_j$  of 1, 1, we just count, in our data set, how many examples there have been in which feature  $j$  has had value 1 and the output was also 1, and divide that by the total number of samples with output 1.





# Learning Algorithm

- Estimate from the data, for all  $j$ :

$$R_j(1,1) = \frac{\#(x_j^i = 1 \wedge y^i = 1)}{\#(y^i = 1)}$$

$$R_j(0,1) = 1 - R_j(1,1)$$

Now,  $R_j$  of 0, 1 is just 1 minus  $R_j$  of 1, 1.



# Learning Algorithm

- Estimate from the data, for all  $j$ :

$$R_j(1,1) = \frac{\#(x_j^i = 1 \wedge y^i = 1)}{\#(y^i = 1)}$$

$$R_j(0,1) = 1 - R_j(1,1)$$

$$R_j(1,0) = \frac{\#(x_j^i = 1 \wedge y^i = 0)}{\#(y^i = 0)}$$

$$R_j(0,0) = 1 - R_j(1,0)$$

Similarly,  $R_j$  of 1, 0 is the number of examples in which feature  $j$  had value 1 and the output was 0, divided the total number of examples with output 0. And  $R_j$  of 0, 0 is just 1 minus  $R_j$  of 1, 0.



# Prediction Algorithm

- Given a new  $x$ ,

$$S(\mathbf{1}) = \prod_j \begin{cases} R_j(1, 1) & \text{if } x_j = 1 \\ R_j(0, 1) & \text{otherwise} \end{cases}$$



# Prediction Algorithm

- Given a new  $x$ ,

$$S(1) = \prod_j \begin{cases} R_j(1,1) & \text{if } x_j = 1 \\ R_j(0,1) & \text{otherwise} \end{cases}$$

$$S(0) = \prod_j \begin{cases} R_j(1,0) & \text{if } x_j = 1 \\ R_j(0,0) & \text{otherwise} \end{cases}$$



# Prediction Algorithm

- Given a new  $x$ ,

$$S(1) = \prod_j \begin{cases} R_j(1, 1) & \text{if } x_j = 1 \\ R_j(0, 1) & \text{otherwise} \end{cases}$$

$$S(0) = \prod_j \begin{cases} R_j(1, 0) & \text{if } x_j = 1 \\ R_j(0, 0) & \text{otherwise} \end{cases}$$

- Output 1 if  $S(1) > S(0)$



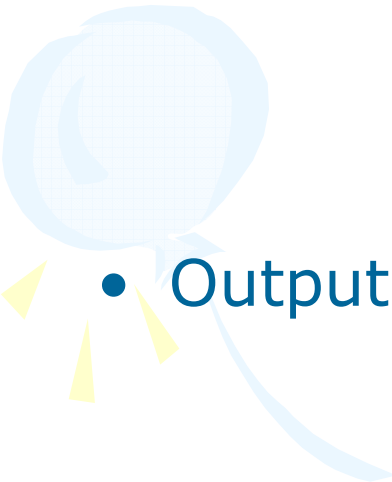
# Prediction Algorithm

- Given a new  $x$ ,

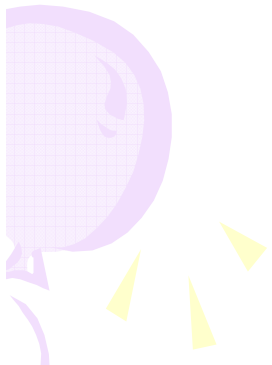
$$\log S(1) = \sum_j \begin{cases} \log R_j(1,1) & \text{if } x_j = 1 \\ \log R_j(0,1) & \text{otherwise} \end{cases}$$

$$\log S(0) = \sum_j \begin{cases} \log R_j(1,0) & \text{if } x_j = 1 \\ \log R_j(0,0) & \text{otherwise} \end{cases}$$

- Output 1 if  $\log S(1) > \log S(0)$



Better to add logs than to multiply small probabilities





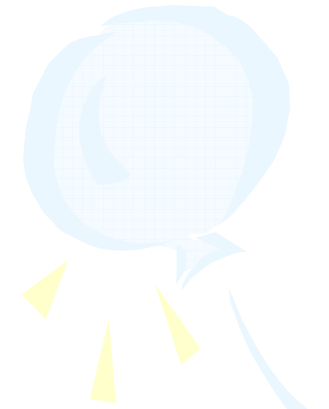


# Laplace Correction

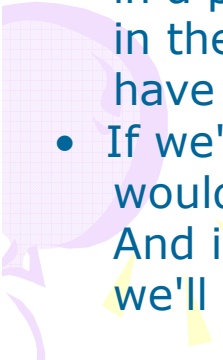
- Avoid getting 0 or 1 as an answer

$$R_j(1,1) = \frac{\#(x_j^i = 1 \wedge y^i = 1) + 1}{\#(y^i = 1) + 2}$$

$$R_j(0,1) = 1 - R_j(1,1)$$


$$R_j(1,0) = \frac{\#(x_j^i = 1 \wedge y^i = 0) + 1}{\#(y^i = 0) + 2}$$

$$R_j(0,0) = 1 - R_j(1,0)$$

- In our example, we saw that if we had never seen a feature take value 1 in a positive example, our estimate for how likely that would be to happen in the future was 0. This is not a good conclusion, especially when we only have had a few examples to learn from.
  - If we've never seen any positive instances, for example, our  $R(1,1)$  values would be  $1/2$ , which seems sort of reasonable in the absence of any information. And if we see lots and lots of examples, this 1 and 2 will be washed out, and we'll get the same estimate that we would have gotten without the correction.
- 



# Example with correction

$f_1$	$f_2$	$f_3$	$f_4$	$y$
0	1	1	0	1
0	0	1	1	1
1	0	1	0	1
0	0	1	1	1
0	0	0	0	1
1	0	0	1	0
1	1	0	1	0
1	0	0	0	0
1	1	0	1	0
1	0	1	1	0

- $R_1(1,1)=2/7$   $R_1(0,1)=5/7$
- $R_1(1,0)=6/7$   **$R_1(0,0)=1/7$**
- $R_2(1,1)=2/7$   $R_2(0,1)=5/7$
- $R_2(1,0)=3/7$   $R_2(0,0)=4/7$
- $R_3(1,1)=5/7$   $R_3(0,1)=2/7$
- $R_3(1,0)=2/7$   $R_3(0,0)=5/7$
- $R_4(1,1)=3/7$   $R_4(0,1)=4/7$
- $R_4(1,0)=5/7$   $R_4(0,0)=2/7$



# Prediction with correction

- $R_1(1,1)=2/7$   $R_1(0,1)=5/7$
- $R_1(1,0)=6/7$   $R_1(0,0)=1/7$
- $R_2(1,1)=2/7$   $R_2(0,1)=5/7$
- $R_2(1,0)=3/7$   $R_2(0,0)=4/7$
- $R_3(1,1)=5/7$   $R_3(0,1)=2/7$
- $R_3(1,0)=2/7$   $R_3(0,0)=5/7$
- $R_4(1,1)=3/7$   $R_4(0,1)=4/7$
- $R_4(1,0)=5/7$   $R_4(0,0)=2/7$

- New  $x = \langle 0, 0, 1, 1 \rangle$
- $S(1) = R_1(0,1) * R_2(0,1) * R_3(1,1) * R_4(1,1) = .156$
- $S(0) = R_1(0,0) * R_2(0,0) * R_3(1,0) * R_4(1,0) = .017$
- $S(1) > S(0)$ , so predict class 1



# Hypothesis Space

- Output 1 if

$$\prod_j \alpha_j x_j + (1 - \alpha_j)(1 - x_j) > \prod_j \beta_j x_j + (1 - \beta_j)(1 - x_j)$$

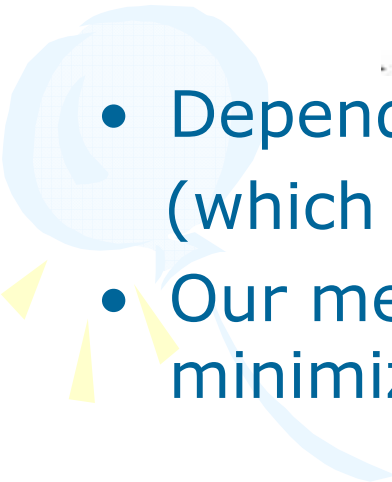
- Depends on parameters  $\alpha_1 \dots \alpha_n, \beta_1 \dots \beta_n$   
(which we set to be the  $R_j$  value)
- 
- 

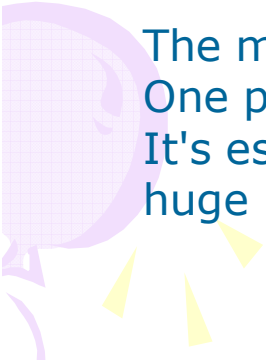


# Hypothesis Space

- Output 1 if

$$\prod_j \alpha_j x_j + (1 - \alpha_j)(1 - x_j) > \prod_j \beta_j x_j + (1 - \beta_j)(1 - x_j)$$

- 
- Depends on parameters  $\alpha_1 \dots \alpha_n, \beta_1 \dots \beta_n$   
(which we set to be the  $R_j$  value)
  - Our method of computing parameters does not minimize the training set error, but it's fast



The main reason we like this algorithm is that it's easy to train. One pass through the data and we can compute all the parameters. It's especially useful in things like text categorization, where there are huge numbers of attributes and we can't possibly look at them many times.

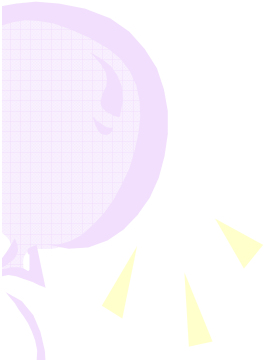


# Hypothesis Space

- Output 1 if

$$\prod_j \alpha_j x_j + (1 - \alpha_j)(1 - x_j) > \prod_j \beta_j x_j + (1 - \beta_j)(1 - x_j)$$

- Depends on parameters  $\alpha_1 \dots \alpha_n, \beta_1 \dots \beta_n$   
(which we set to be the  $R_j$  value)
- Our method of computing parameters does not minimize the training set error, but it's fast
- Weight of feature  $j$ 's vote in favor of output 1:

$$\log \frac{\alpha_j}{1 - \alpha_j} - \log \frac{\beta_j}{1 - \beta_j}$$


# Exclusive OR

$f_1$	$f_2$	$f_3$	$f_4$	$y$
0	1	1	0	0
1	0	1	0	0
1	0	0	1	0
0	1	0	1	0
1	1	1	0	1
0	0	0	1	1

# Exclusive OR

$f_1$	$f_2$	$f_3$	$f_4$	$\gamma$
0	1	1	0	0
1	0	1	0	0
1	0	0	1	0
0	1	0	1	0
1	1	1	0	1
0	0	0	1	1

- $R_1(1,1)=2/4$   $R_1(0,1)=2/4$
- $R_1(1,0)=3/6$   $R_1(0,0)=3/6$
- $R_2(1,1)=2/4$   $R_2(0,1)=2/4$
- $R_2(1,0)=3/6$   $R_2(0,0)=3/6$
- $R_3(1,1)=2/4$   $R_3(0,1)=2/4$
- $R_3(1,0)=3/6$   $R_3(0,0)=3/6$
- $R_4(1,1)=2/4$   $R_4(0,1)=2/4$
- $R_4(1,0)=3/6$   $R_4(0,0)=3/6$



# Exclusive OR

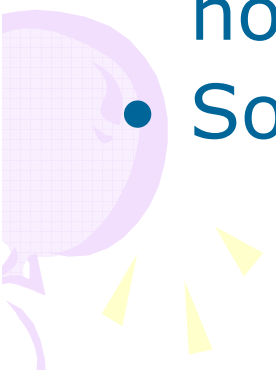
$f_1$	$f_2$	$f_3$	$f_4$	$y$
0	1	1	0	0
1	0	1	0	0
1	0	0	1	0
0	1	0	1	0
1	1	1	0	1
0	0	0	1	1

- $R_1(1,1)=2/4$   $R_1(0,1)=2/4$
- $R_1(1,0)=3/6$   $R_1(0,0)=3/6$
- $R_2(1,1)=2/4$   $R_2(0,1)=2/4$
- $R_2(1,0)=3/6$   $R_2(0,0)=3/6$
- $R_3(1,1)=2/4$   $R_3(0,1)=2/4$
- $R_3(1,0)=3/6$   $R_3(0,0)=3/6$
- $R_4(1,1)=2/4$   $R_4(0,1)=2/4$
- $R_4(1,0)=3/6$   $R_4(0,0)=3/6$

- For any new  $x$
- $S(1) = .5 * .5 * .5 * .5 = .0625$
- $S(0) = .5 * .5 * .5 * .5 = .0625$
- We're indifferent between classes



# Congressional Voting

- Accuracy on the congressional voting domain is about 0.91
  - Somewhat worse than decision trees (0.95)
  - Decision trees can express more complex hypotheses over combinations of attributes
  - Domain is small enough so that speed is not an issue
  - So, prefer trees or DNF in this domain
- 



# Congressional Voting

-6.82	physician-fee-freeze
-4.20	el-salvador-aid
-4.20	crime
-3.56	education-spending
3.36	adoption-of-the-budget-resolution
3.25	aid-to-nicaraguan-contras
3.07	mx-missile
-2.51	superfund-right-to-sue
2.40	duty-free-exports
2.14	anti-satellite-test-ban
-2.07	religious-groups-in-schools
2.01	export-administration-act-south-africa
1.66	synfuels-corporation-cutback
1.63	handicapped-infants
-0.17	immigration
-0.08	water-project-cost-sharing

republican  
democrat