



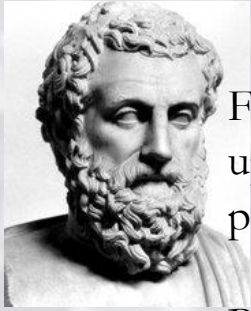
Learning Theory & Theory of Generalization  
CS 446/546

# Outline

- Learning Feasibility
- VC Dimension
- Theory of Generalization
- Bayesian Concept Learning
- Beta-Binomial Model
- Dirichlet-Multinomial Model



# Just a little history: Baconian Method



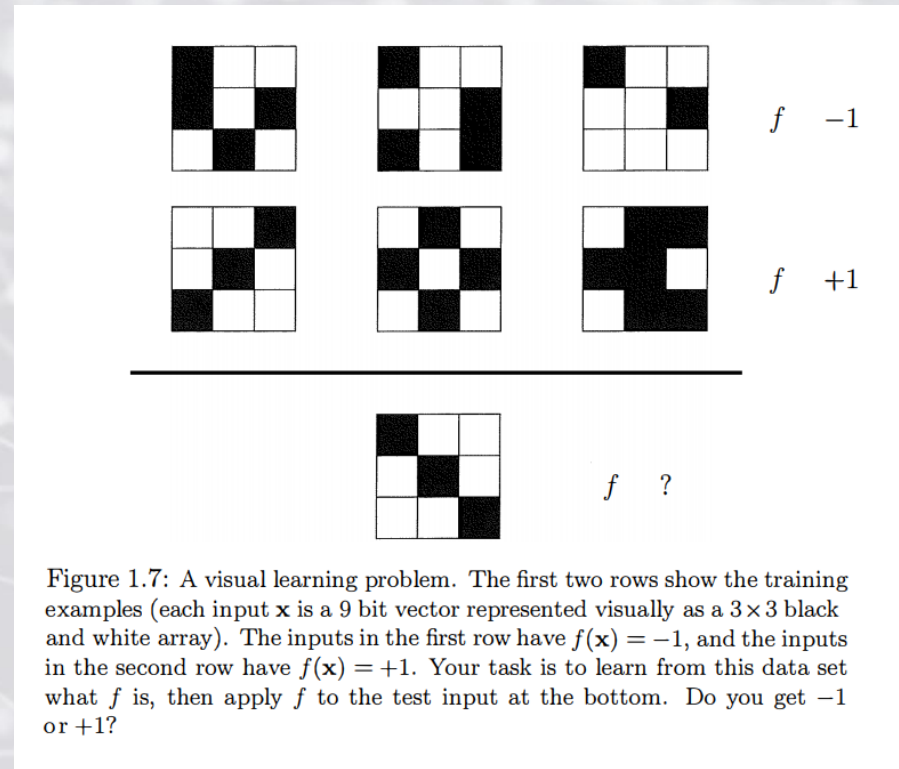
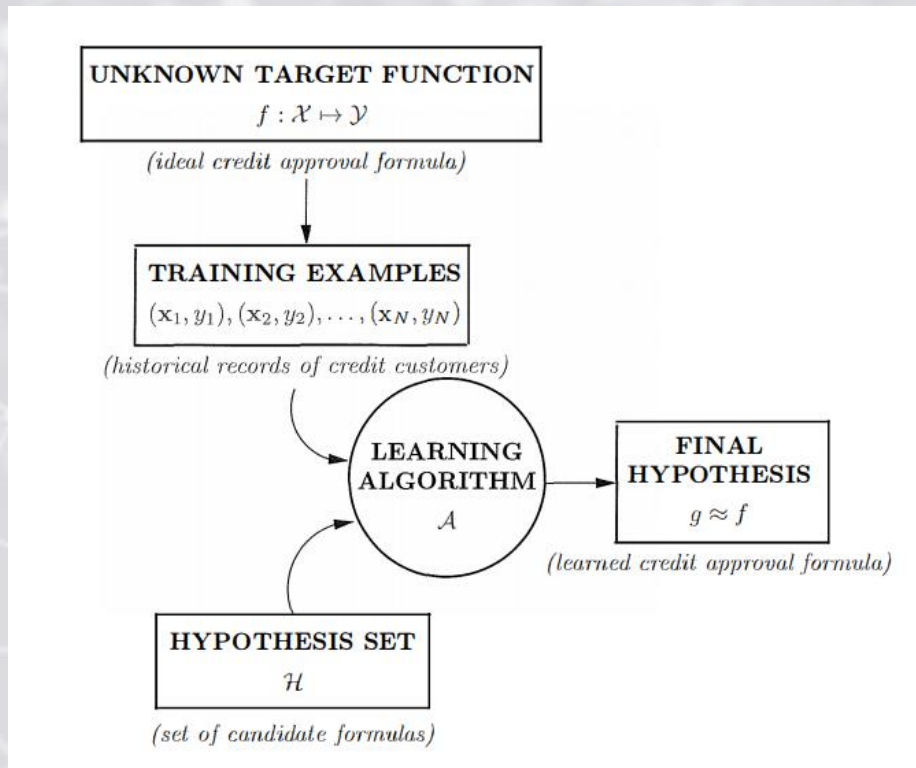
Francis Bacon's text *Novum Organum* (1620) -- "New Method" -- was highly influential upon the development of modern science, chiefly for rejecting medieval *Aristotelianism* and promulgating inductive reasoning in the sciences.

Bacon's method of induction is much more complex than the essential inductive process of making generalizations from observations. His method includes a meticulous description of the requirements for making the careful, systematic observations necessary to produce quality facts. He then proceeds to use induction, the ability to generalize from a set of facts to one or more axioms. However, he stresses the necessity of not generalizing beyond what the facts truly demonstrate. The next step may be to gather additional data, or the researcher may use existing data and the new axioms to establish additional axioms. The whole process is repeated in a stepwise fashion to build an increasingly complex base of knowledge, but one which is always supported by observed facts/empirical data.

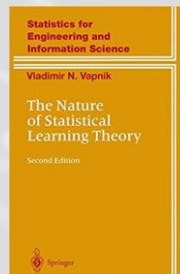


# Learning Feasibility

- In Machine Learning we wish to learn an unknown target function  $f$ . A natural question arises: how can a limited data set reveal enough information to pin down an entire target function?



(\*) Recommended reading:  
Vapnik, The Nature of  
Statistical Learning Theory



# Learning Feasibility

- In Machine Learning we wish to learn an unknown target function  $f$ . A natural question arises: how can a limited data set reveal enough information to pin down an entire target function?

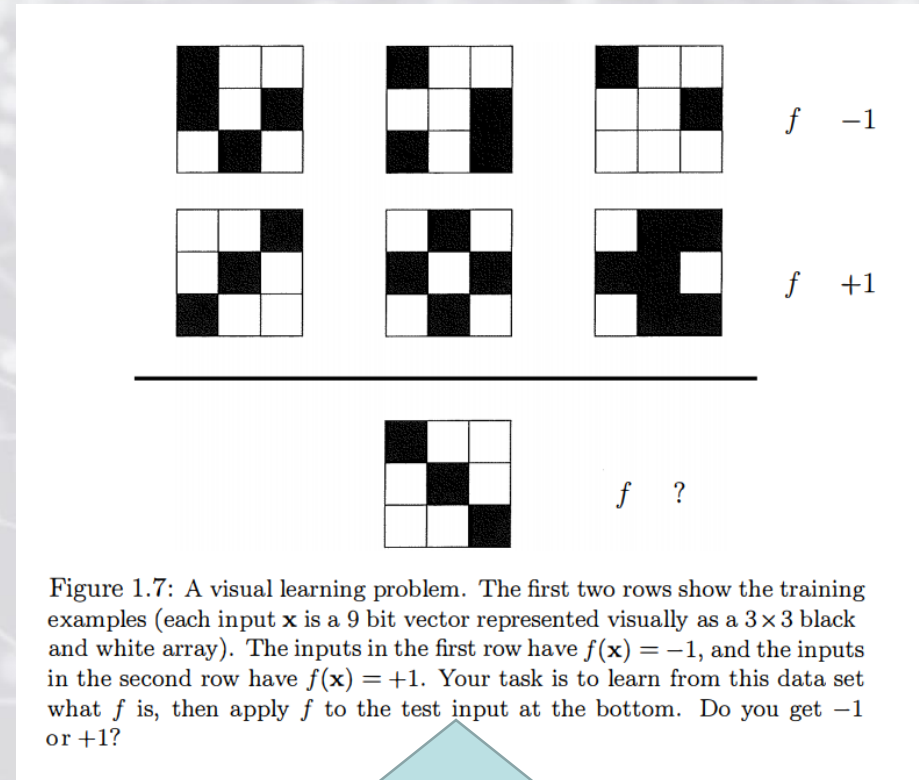
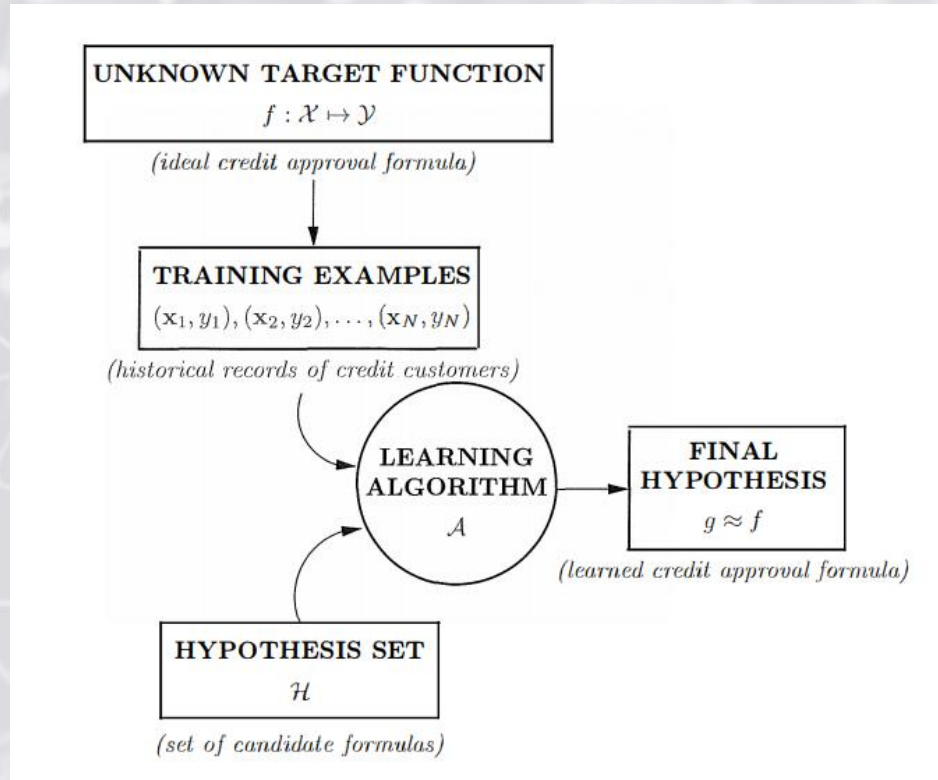
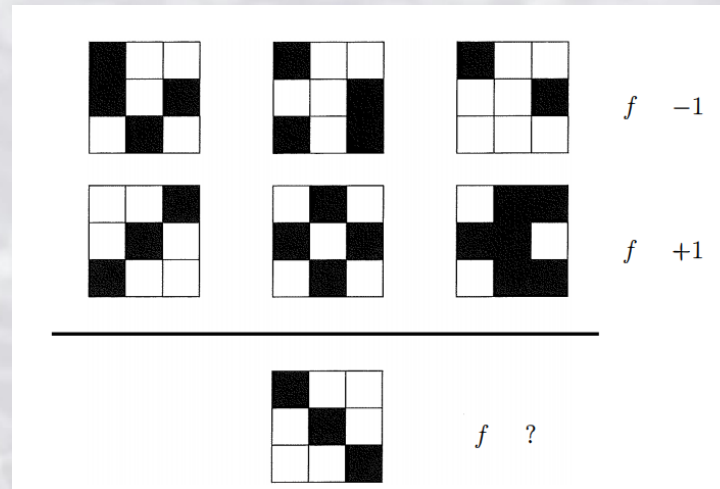


Figure 1.7: A visual learning problem. The first two rows show the training examples (each input  $\mathbf{x}$  is a 9 bit vector represented visually as a  $3 \times 3$  black and white array). The inputs in the first row have  $f(\mathbf{x}) = -1$ , and the inputs in the second row have  $f(\mathbf{x}) = +1$ . Your task is to learn from this data set what  $f$  is, then apply  $f$  to the test input at the bottom. Do you get  $-1$  or  $+1$ ?

$f$  is in fact ambiguous here;  $f=1$  for symmetric pattern works – but so does  $f=-1$  when top-left square is white.



# Learning Feasibility



- A formative question in ML: Does the data set  $D$  tell us anything outside of  $D$  that we didn't know before? If the answer is yes, then we have learned something; otherwise, if the answer is no, we can conclude that learning is not feasible.
- Because we are never privy to the function  $f$  (otherwise learning itself would be needless), it stands to reason that  $f$  remains unknown outside of  $D$ . We demonstrate this notion with an example.

# Learning Feasibility

- Consider a Boolean target function over a 3-D input space  $X=\{0,1\}^3$ . We are given a data set  $D$  of five examples represented in the table.

| $\mathbf{x}_n$ | $y_n$ |
|----------------|-------|
| 0 0 0          | ○     |
| 0 0 1          | ●     |
| 0 1 0          | ●     |
| 0 1 1          | ○     |
| 1 0 0          | ●     |

- Where  $y_n=f(\mathbf{x}_n)$  for  $n = 1, 2, 3, 4, 5$ . Note that there are  $2^8=256$  distinct Boolean functions defined on 3 Boolean inputs.

# Learning Feasibility

- Consider a Boolean target function over a 3-D input space  $X = \{0,1\}^3$ . We are given a data set  $D$  of five examples represented in the table.

| $\mathbf{x}_n$ | $y_n$ |
|----------------|-------|
| 0 0 0          | ○     |
| 0 0 1          | ●     |
| 0 1 0          | ●     |
| 0 1 1          | ○     |
| 1 0 0          | ●     |

- Where  $y_n = f(\mathbf{x}_n)$  for  $n = 1, 2, 3, 4, 5$ . Note that there are  $2^8 = 256$  distinct Boolean functions defined on 3 Boolean inputs.

- Let's consider the problem of learning  $f$  concretely. Since  $f$  is unknown except inside  $D$ , any function that agrees with  $D$  could conceivably be  $f$ . The table below shows all such functions  $f_1, \dots, f_8$ ;  $g$  is the final hypothesis.

| $\mathbf{x}$ | $y$ | $g$ | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ | $f_7$ | $f_8$ |
|--------------|-----|-----|-------|-------|-------|-------|-------|-------|-------|-------|
| 0 0 0        | ○   | ○   | ○     | ○     | ○     | ○     | ○     | ○     | ○     | ○     |
| 0 0 1        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 0 1 0        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 0 1 1        | ○   | ○   | ○     | ○     | ○     | ○     | ○     | ○     | ○     | ○     |
| 1 0 0        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 1 0 1        |     | ?   | ○     | ○     | ○     | ○     | ●     | ●     | ●     | ●     |
| 1 1 0        |     | ?   | ○     | ○     | ●     | ●     | ○     | ○     | ●     | ●     |
| 1 1 1        |     | ?   | ○     | ●     | ○     | ●     | ○     | ●     | ○     | ●     |



# Learning Feasibility

| $\mathbf{x}$ | $y$ | $g$ | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ | $f_7$ | $f_8$ |
|--------------|-----|-----|-------|-------|-------|-------|-------|-------|-------|-------|
| 0 0 0        | ○   | ○   | ○     | ○     | ○     | ○     | ○     | ○     | ○     | ○     |
| 0 0 1        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 0 1 0        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 0 1 1        | ○   | ○   | ○     | ○     | ○     | ○     | ○     | ○     | ○     | ○     |
| 1 0 0        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 1 0 1        |     | ?   | ○     | ○     | ○     | ○     | ●     | ●     | ●     | ●     |
| 1 1 0        |     | ?   | ○     | ○     | ●     | ●     | ○     | ○     | ●     | ●     |
| 1 1 1        |     | ?   | ○     | ●     | ○     | ●     | ○     | ●     | ○     | ●     |

- We cannot exclude any of the  $f_1, \dots, f_8$  from being the true  $f$  – this poses a natural dilemma: choosing an appropriate  $g$  function (to be used for predictions on unseen data) is **deterministically ill-posed**.
  - Furthermore, the performance of  $g$  on  $D$  makes no difference whatsoever as far as the performance *outside of*  $D$  is concerned. Nevertheless, the performance outside  $D$  is all that matters for learning.
- (\*) As we show, inference can be done effectively when we introduce a probabilistic framework.

# Learning Feasibility

- Consider a Boolean target function over a 3-D input space  $X=\{0,1\}^3$ . We are given a data set  $D$  of five examples represented in the table.

| $\mathbf{x}_n$ | $y_n$ |
|----------------|-------|
| 0 0 0          | ○     |
| 0 0 1          | ●     |
| 0 1 0          | ●     |
| 0 1 1          | ○     |
| 1 0 0          | ●     |

- Where  $y_n=f(\mathbf{x}_n)$  for  $n = 1, 2, 3, 4, 5$ . Note that there are  $2^8=256$  distinct Boolean functions defined on 3 Boolean inputs.

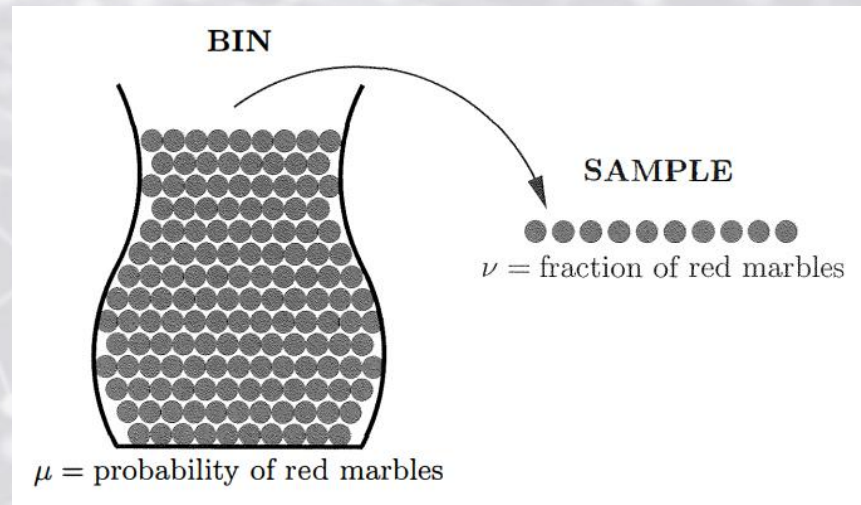
- Let's consider the problem of learning  $f$  concretely. Since  $f$  is unknown except inside  $D$ , any function that agrees with  $D$  could conceivably be  $f$ . The table below shows all such functions  $f_1, \dots, f_8$ ;  $g$  is the final hypothesis.

| $\mathbf{x}$ | $y$ | $g$ | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ | $f_7$ | $f_8$ |
|--------------|-----|-----|-------|-------|-------|-------|-------|-------|-------|-------|
| 0 0 0        | ○   | ○   | ○     | ○     | ○     | ○     | ○     | ○     | ○     | ○     |
| 0 0 1        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 0 1 0        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 0 1 1        | ○   | ○   | ○     | ○     | ○     | ○     | ○     | ○     | ○     | ○     |
| 1 0 0        | ●   | ●   | ●     | ●     | ●     | ●     | ●     | ●     | ●     | ●     |
| 1 0 1        |     | ?   | ○     | ○     | ○     | ○     | ●     | ●     | ●     | ●     |
| 1 1 0        |     | ?   | ○     | ○     | ●     | ●     | ○     | ○     | ●     | ●     |
| 1 1 1        |     | ?   | ○     | ●     | ○     | ●     | ○     | ●     | ○     | ●     |

(\*) To infer something outside of  $D$  we need to incorporate **probabilistic bounds**.

# Learning Feasibility: Hoeffding Inequality

- As a simple example, consider a bin consisting of green and red marbles; let  $\mu$  equal the true proportion of red marbles – so  $1 - \mu$  represents the true proportion of green marbles. We pick a random sample of  $N$  independent marbles (*with replacement*) from the bin and observe  $\nu$ , the fraction of red marbles within the sample.





# Learning Feasibility: Hoeffding Inequality

- As a simple example, consider a bin consisting of green and red marbles; let  $\mu$  equal the true proportion of red marbles – so  $1 - \mu$  represents the true proportion of green marbles. We pick a random sample of  $N$  independent marbles (*with replacement*) from the bin and observe  $v$ , the fraction of red marbles within the sample.
- To quantify the relationship between  $\mu$  (true proportion of red) and  $v$  (sample proportion of red), we use a simple bound called the **Hoeffding Inequality**, which states that for any sample size  $N$ :

$$P(|\mu - v| > \varepsilon) \leq 2e^{-2\varepsilon^2 N} \quad \text{for any } \varepsilon > 0$$

- Note:  $v$  is a random variable;  $\mu$  is a fixed constant. The bound (RHS) is independent of  $\mu$ . The utility of the Hoeffding Inequality is to infer the value of  $\mu$  using the *point-estimate*  $v$ ; only the size of the sample  $N$  (not the bin size itself) affects the bound; a sample value  $v$  “close” to  $\mu$  necessitates a large sample size.

(\*) For bounds for sums of independent random variables, see: *Chernoff bounds*.

# Learning Feasibility: Hoeffding Inequality

Q: How does the Hoeffding Inequality relate to the general learning problem?

- Take any single hypothesis  $h \in H$  and compare it to  $f$  on each point  $\mathbf{x} \in X$ . If  $h(\mathbf{x}) = f(\mathbf{x})$ , color the point  $\mathbf{x}$  green (we got it right) – otherwise if  $h(\mathbf{x}) \neq f(\mathbf{x})$  color the point  $\mathbf{x}$  red.
- The color that each point gets is not known to us, since  $f$  is unknown. However, if we pick  $\mathbf{x}$  at random according to some probability distribution  $P$  over the input space  $X$ , we know that  $\mathbf{x}$  will be red with some probability, call it  $\mu$ , and green with probability  $1 - \mu$ . Regardless of the value of  $\mu$ , the space  $X$  now behaves like the bin we previously described.

# Learning Feasibility: Hoeffding Inequality

Q: How does the Hoeffding Inequality relate to the general learning problem?

- Take any single hypothesis  $h \in H$  and compare it to  $f$  on each point  $\mathbf{x} \in X$ . If  $h(\mathbf{x}) = f(\mathbf{x})$ , color the point  $\mathbf{x}$  green (we got it right) – otherwise if  $h(\mathbf{x}) \neq f(\mathbf{x})$  color the point  $\mathbf{x}$  red.
- The color that each point gets is not known to us, since  $f$  is unknown. However, if we pick  $\mathbf{x}$  at random according to some probability distribution  $P$  over the input space  $X$ , we know that  $\mathbf{x}$  will be red with some probability, call it  $\mu$ , and green with probability  $1 - \mu$ . Regardless of the value of  $\mu$ , the space  $X$  now behaves like the bin we previously described.

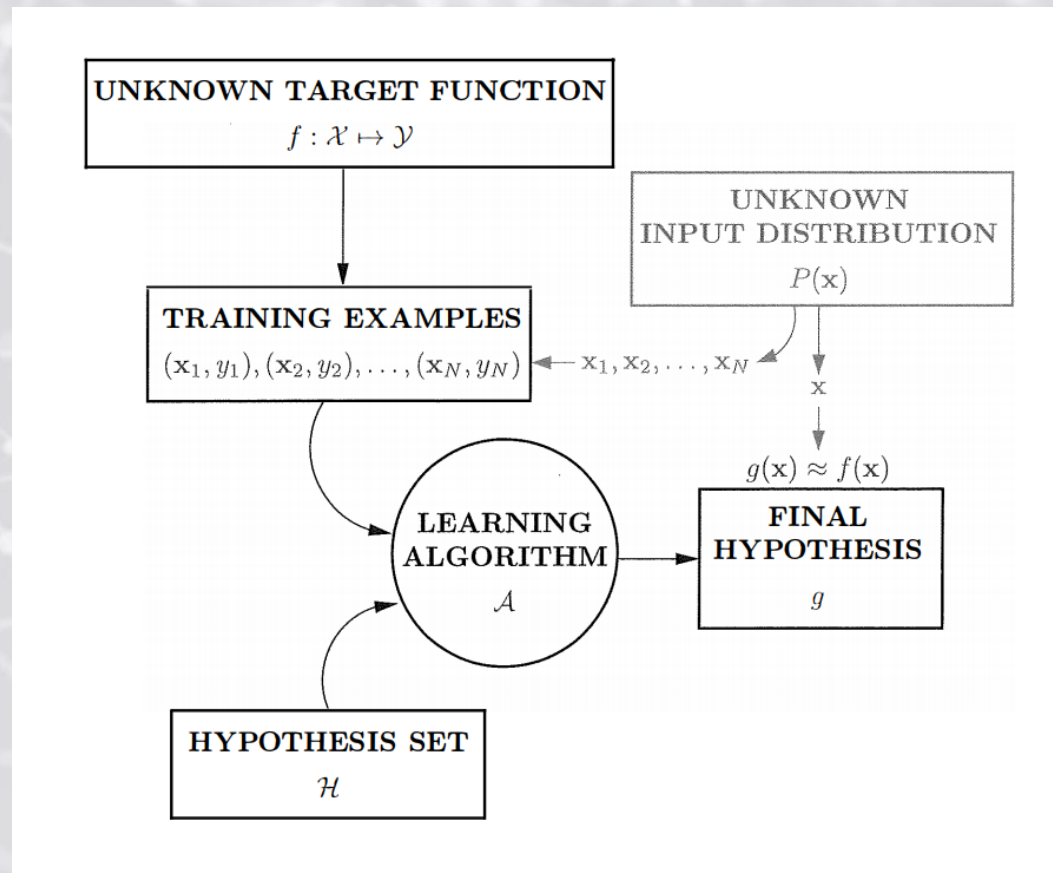
(\*) The training examples play the role of a sample from the bin. If the inputs  $\mathbf{x}_1, \dots, \mathbf{x}_N$  in  $D$  are picked independently according to  $P$ , we will get a random sample of red ( $h(\mathbf{x}) \neq f(\mathbf{x})$ ) and green ( $h(\mathbf{x}) = f(\mathbf{x})$ ) points. The color of each point will be known to us, since both  $h(\mathbf{x}_n)$  and  $f(\mathbf{x}_n)$  are known for  $n = 1, \dots, N$ . The learning problem is thus reduced to the bin problem.



# Learning Feasibility: Hoeffding Inequality

(\*) With this equivalence, the Hoeffding Inequality can be applied to the learning problem – which enables us to make predictions outside of  $D$ .

- Using  $v$  to predict  $\mu$  tells us something about  $f$ , although it doesn't tell us what  $f$  is. What  $\mu$  tells us is the error rate  $h$  makes in approximating  $f$ . If  $v$  happens to be close to zero, we can predict that  $h$  will approximate  $f$  well over the entire input space.



# Learning Feasibility: Hoeffding Inequality

- For a more complete equivalence, we want to consider the case where we have multiple hypotheses, in order to capture real learning.

Define the **in-sample error** for hypothesis  $h$ :

$$E_{in}(h) = \frac{1}{N} \sum_{i=1}^N I[h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]$$

where  $I$  denotes the indicator function. Define the **out-of-sample error**:

$$E_{out}(h) = P[h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]$$

which corresponds to  $\mu$  in the bin model. The probability is based on the distribution  $P$  over  $X$  which is used to sample the data points  $\mathbf{x}$ .

# Learning Feasibility: Hoeffding Inequality

$$E_{in}(h) = \frac{1}{N} \sum_{i=1}^N I[h(\mathbf{x}_n) \neq f(\mathbf{x}_n)] \quad E_{out}(h) = P[h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]$$

- With these definitions, we can apply the Hoeffding Inequality using  $E_{in}$  for  $v$  and  $E_{out}$  for  $\mu$ :

$$P(|E_{in}(h) - E_{out}(h)| > \varepsilon) \leq 2e^{-2\varepsilon^2 N} \quad \text{for any } \varepsilon > 0$$

- Recall that the in-sample error  $E_{in}$ , just like  $v$ , is a random variable that depends on the sample. The out-of-sample error,  $E_{out}$ , just like  $\mu$ , is unknown but fixed.



# Learning Feasibility: Hoeffding Inequality

$$E_{in}(h) = \frac{1}{N} \sum_{i=1}^N I[h(\mathbf{x}_i) \neq f(\mathbf{x}_i)] \quad E_{out}(h) = P[h(\mathbf{x}) \neq f(\mathbf{x})]$$

- With these definitions, we can apply the Hoeffding Inequality using  $E_{in}$  for  $v$  and  $E_{out}$  for  $\mu$ :

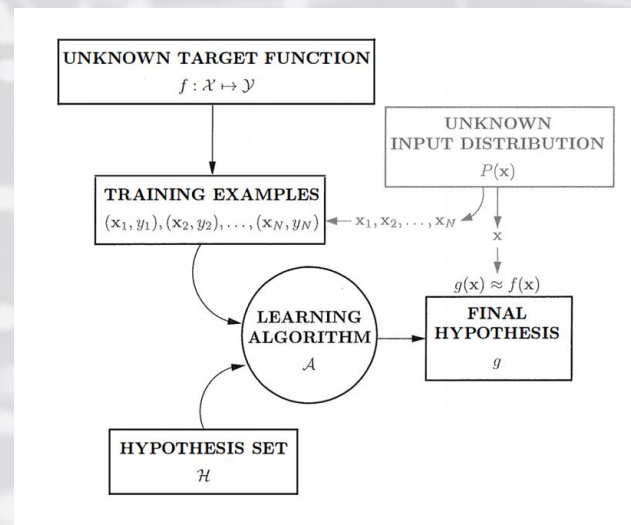
$$P(|E_{in}(h) - E_{out}(h)| > \varepsilon) \leq 2e^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

- Recall that the in-sample error  $E_{in}$ , just like  $v$ , is a random variable that depends on the sample. The out-of-sample error,  $E_{out}$ , just like  $\mu$ , is unknown but fixed.

(\*) Notice that this bound makes a subtle assumption. Here we presume that the hypothesis  $h$  is fixed **before** we generate the data set. If you are allowed to change  $h$  after you generate the data set, the assumptions that are needed to prove the Hoeffding Inequality no longer hold!

- (\*) We are in fact most interested in a bound (as it applies to the general learning problem) of the form:

$$P(|E_{in}(g) - E_{out}(g)| > \varepsilon) \text{ "is small" for the final hypothesis } g$$



# Learning Feasibility: Hoeffding Inequality

$$E_{in}(h) = \frac{1}{N} \sum_{i=1}^N I[h(\mathbf{x}_i) \neq f(\mathbf{x}_i)] \quad E_{out}(h) = P[h(\mathbf{x}) \neq f(\mathbf{x})]$$

- With these definitions, we can apply the Hoeffding Inequality using  $E_{in}$  for  $v$  and  $E_{out}$  for  $\mu$ :

$$P(|E_{in}(h) - E_{out}(h)| > \varepsilon) \leq 2e^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

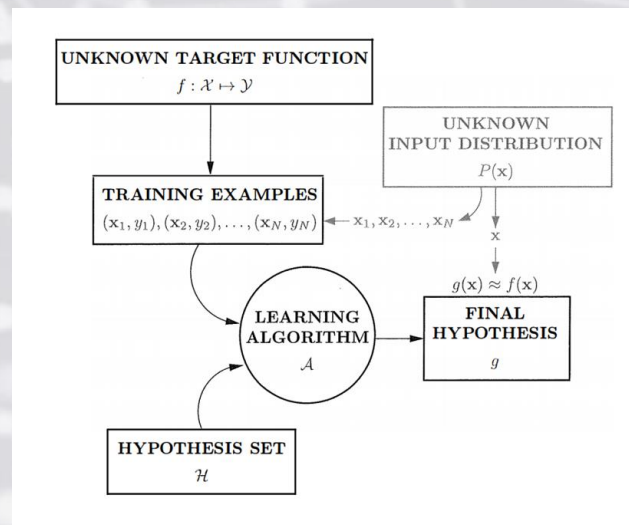
- Recall that the in-sample error  $E_{in}$ , just like  $v$ , is a random variable that depends on the sample. The out-of-sample error,  $E_{out}$ , just like  $\mu$ , is unknown but fixed.

(\*) Notice that this bound makes a subtle assumption. Here we presume that the hypothesis  $h$  is fixed **before** we generate the data set. If you are allowed to change  $h$  after you generate the data set, the assumptions that are needed to prove the Hoeffding Inequality no longer hold!

- (\*) We are in fact most interested in a bound (as it applies to the general learning problem) of the form:

$$P(|E_{in}(g) - E_{out}(g)| > \varepsilon) \text{ "is small" for the final hypothesis } g$$

Where  $g$  is **not fixed** before generating the data – all the more: which hypothesis is selected to be  $g$  depends on the data.



# Learning Feasibility: Hoeffding Inequality

Q: How do we generalize the Hoeffding Inequality to the case of  $g$ , the final hypothesis that we choose (which depends on the data)?



# Learning Feasibility: Hoeffding Inequality

Q: How do we generalize the Hoeffding Inequality to the case of  $g$ , the final hypothesis that we choose (which depends on the data)?

A: There is an elementary method. Notice that it follows that:

$$\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon \xrightarrow{\text{implies}} \left| E_{in}(h_1) - E_{out}(h_1) \right| > \varepsilon \text{ OR } \left| E_{in}(h_2) - E_{out}(h_2) \right| > \varepsilon \dots \text{OR } \left| E_{in}(h_M) - E_{out}(h_M) \right| > \varepsilon$$

where we assume  $H = \{h_1, h_2, \dots, h_M\}$ , i.e., the hypothesis space is finite.

# Learning Feasibility: Hoeffding Inequality

Q: How do we generalize the Hoeffding Inequality to the case of  $g$ , the final hypothesis that we choose (which depends on the data)?

A: There is an elementary method. Notice that it follows that:

$$\left|E_{in}(g) - E_{out}(g)\right| > \varepsilon \xrightarrow{\text{implies}} \left|E_{in}(h_1) - E_{out}(h_1)\right| > \varepsilon \text{ OR } \left|E_{in}(h_2) - E_{out}(h_2)\right| > \varepsilon \dots \text{ OR } \left|E_{in}(h_M) - E_{out}(h_M)\right| > \varepsilon$$

where we assume  $H = \{h_1, h_2, \dots, h_M\}$ , i.e., the hypothesis space is finite.

This implies:

$$P\left(\left|E_{in}(g) - E_{out}(g)\right| > \varepsilon\right) \leq P\left(\left|E_{in}(h_1) - E_{out}(h_1)\right| > \varepsilon \text{ OR } \left|E_{in}(h_2) - E_{out}(h_2)\right| > \varepsilon \dots \text{ OR } \left|E_{in}(h_M) - E_{out}(h_M)\right| > \varepsilon\right)$$

# Learning Feasibility: Hoeffding Inequality

Q: How do we generalize the Hoeffding Inequality to the case of  $g$ , the final hypothesis that we choose (which depends on the data)?

A: There is an elementary method. Notice that it follows that:

$$\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon \xrightarrow{\text{implies}} \left| E_{in}(h_1) - E_{out}(h_1) \right| > \varepsilon \text{ OR } \left| E_{in}(h_2) - E_{out}(h_2) \right| > \varepsilon \dots \text{OR } \left| E_{in}(h_M) - E_{out}(h_M) \right| > \varepsilon$$

where we assume  $H = \{h_1, h_2, \dots, h_M\}$ , i.e., the hypothesis space is finite.

This implies:

$$P\left(\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon\right) \leq P\left(\left| E_{in}(h_1) - E_{out}(h_1) \right| > \varepsilon \text{ OR } \left| E_{in}(h_2) - E_{out}(h_2) \right| > \varepsilon \dots \text{OR } \left| E_{in}(h_M) - E_{out}(h_M) \right| > \varepsilon\right)$$
$$\leq \sum_{m=1}^M P\left(\left| E_{in}(h_m) - E_{out}(h_m) \right| > \varepsilon\right)$$



Why?



# Learning Feasibility: Hoeffding Inequality

Q: How do we generalize the Hoeffding Inequality to the case of  $g$ , the final hypothesis that we choose (which depends on the data)?

A: There is an elementary method. Notice that it follows that:

$$\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon \xrightarrow{\text{implies}} \left| E_{in}(h_1) - E_{out}(h_1) \right| > \varepsilon \text{ OR } \left| E_{in}(h_2) - E_{out}(h_2) \right| > \varepsilon \dots \text{OR } \left| E_{in}(h_M) - E_{out}(h_M) \right| > \varepsilon$$

where we assume  $H = \{h_1, h_2, \dots, h_M\}$ , i.e., the hypothesis space is finite.

This implies:

$$P\left(\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon\right) \leq P\left(\left| E_{in}(h_1) - E_{out}(h_1) \right| > \varepsilon \text{ OR } \left| E_{in}(h_2) - E_{out}(h_2) \right| > \varepsilon \dots \text{OR } \left| E_{in}(h_M) - E_{out}(h_M) \right| > \varepsilon\right)$$
$$\leq \sum_{m=1}^M P\left(\left| E_{in}(h_m) - E_{out}(h_m) \right| > \varepsilon\right)$$



Why? The “union bound”:  
 $P(A_1 \text{ OR } A_2) \leq P(A_1) + P(A_2)$

# Learning Feasibility: Hoeffding Inequality

Q: How do we generalize the Hoeffding Inequality to the case of  $g$ , the final hypothesis that we choose (which depends on the data)?

A: There is an elementary method. Notice that it follows that:

$$\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon \xrightarrow{\text{implies}} \left| E_{in}(h_1) - E_{out}(h_1) \right| > \varepsilon \text{ OR } \left| E_{in}(h_2) - E_{out}(h_2) \right| > \varepsilon \dots \text{ OR } \left| E_{in}(h_M) - E_{out}(h_M) \right| > \varepsilon$$

where we assume  $H = \{h_1, h_2, \dots, h_M\}$ , i.e., the hypothesis space is finite.

This implies:

$$\begin{aligned} P\left(\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon\right) &\leq P\left(\left| E_{in}(h_1) - E_{out}(h_1) \right| > \varepsilon \text{ OR } \left| E_{in}(h_2) - E_{out}(h_2) \right| > \varepsilon \dots \text{ OR } \left| E_{in}(h_M) - E_{out}(h_M) \right| > \varepsilon\right) \\ &\leq \sum_{m=1}^M P\left(\left| E_{in}(h_m) - E_{out}(h_m) \right| > \varepsilon\right) \end{aligned}$$

(\*) Now we can apply the Hoeffding Inequality to the  $M$  terms one at a time, bounding each by  $2e^{-2\varepsilon^2 N}$ .

$$P\left(\left| E_{in}(g) - E_{out}(g) \right| > \varepsilon\right) \leq 2Me^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

(\*) This yields a “uniform” bound; the downside for this uniform estimate is that the probability bound is a factor of  $M$  looser than the bound for a single hypothesis, and is only meaningful if  $M$  is finite.

# Learning Feasibility

- Let's now reconcile the issue of whether  $D$  tells us anything outside of  $D$  that we didn't know before.

(\*) If we insist on a deterministic answer, which means that  $D$  tells us something *certain* about  $f$  outside of  $D$  – then the answer is NO.

(\*) However, if we accept a probabilistic answer, which means that  $D$  tells us something *likely* about  $f$  outside of  $D$  then the answer is YES. (the only assumption we made was that the examples in  $D$  are generated independently)



# Learning Feasibility

- Next, let's pin down what we mean by the **feasibility of learning**.
- If learning is successful, then  $g$  should approximate  $f$  well, which means that  $E_{out}(g) \approx 0$ . However, this is not what we get from the probabilistic analysis. What we get instead is:  
 $E_{out}(g) \approx E_{in}(g)$ .
- We still have to make  $E_{in}(g) \approx 0$  in order to conclude that  $E_{out}(g) \approx 0$ . We cannot guarantee that we will find a hypothesis that achieves  $E_{in}(g) \approx 0$ . The Hoeffding Inequality nevertheless assures us that  $E_{out}(g) \approx E_{in}(g)$ , so we can use  $E_{in}$  as a proxy for  $E_{out}$ .

In general, the feasibility of learning boils down to (2) conditions:

- (1) Can we make sure that  $E_{out}(g)$  is close enough to  $E_{in}(g)$ ? (Hoeffding Inequality addresses this)
  - (2) Can we make  $E_{in}(g)$  small enough?
- (\*) The second question is answered after we run the learning algorithm on the actual data and see how small we can get  $E_{in}$  to be.

# Learning Feasibility

In general, the feasibility of learning boils down to (2) conditions:

(1) Can we make sure that  $E_{\text{out}}(g)$  is close enough to  $E_{\text{in}}(g)$ ? (Hoeffding Inequality addresses this)

$$P(|E_{\text{in}}(g) - E_{\text{out}}(g)| > \varepsilon) \leq 2Me^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

(2) Can we make  $E_{\text{in}}(g)$  small enough?

(\*) The second question is answered after we run the learning algorithm on the actual data and see how small we can get  $E_{\text{in}}$  to be.

**Complexity of  $H$ :** If the number of hypotheses  $M$  goes up, we run more risk that  $E_{\text{in}}(g)$  will be a poor estimator of  $E_{\text{out}}(g)$  according to the Hoeffding Inequality.  $M$  can thus be thought of as a measure of the “complexity” of the hypothesis set  $H$  that we use.

If we want to satisfy issue (1) above, we need a relatively small  $M$ . Conversely, if we want to satisfy condition (2), it is better to have a large hypothesis space  $H$ .

(\*) This tradeoff is, naturally, a major theme in ML.

# Learning Feasibility

In general, the feasibility of learning boils down to (2) conditions:

(1) Can we make sure that  $E_{\text{out}}(g)$  is close enough to  $E_{\text{in}}(g)$ ? (Hoeffding Inequality addresses this)

$$P(|E_{\text{in}}(g) - E_{\text{out}}(g)| > \varepsilon) \leq 2Me^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

(2) Can we make  $E_{\text{in}}(g)$  small enough?

(\*) The second question is answered after we run the learning algorithm on the actual data and see how small we can get  $E_{\text{in}}$  to be.

**Complexity of  $f$ :** If  $f$  is complex, this should be harder to learn than a simple  $f$ . Note that the complexity of  $f$  does not affect how well  $E_{\text{in}}(g)$  approximate  $E_{\text{out}}(g)$ .

However, this doesn't imply that we can learn complex functions as easily as simple functions. If the target function is complex, then this affects condition (2), since the data are hard to fit for a complex  $f$ . In order to decrease  $E_{\text{in}}(g)$  for a complex  $f$  we need to increase  $M$  (the size of the hypothesis set), which in turn makes learning more difficult.



# Theory of Generalization

- We have previously discussed how the value of  $E_{\text{in}}$  does not always generalize to a similar value of  $E_{\text{out}}$ . Generalization is a key issue in learning. One can define the **generalization error** as the discrepancy between  $E_{\text{in}}$  and  $E_{\text{out}}$ .

The Hoeffding Inequality furnished a way to characterize the generalization error with a probabilistic bound:

$$P\left(\left|E_{\text{in}}(g) - E_{\text{out}}(g)\right| > \varepsilon\right) \leq 2Me^{-2\varepsilon^2 N} \quad \text{for any } \varepsilon > 0$$

We can rephrase this as follows: given a **tolerance level**  $\delta$ , one can assert with probability *at least*  $1 - \delta$  that:

$$E_{\text{out}}(g) \leq E_{\text{in}}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

(\*) This is known as a *generalization bound* because it bounds  $E_{\text{out}}$  in terms of  $E_{\text{in}}$ .

# Theory of Generalization

- We have previously discussed how the value of  $E_{in}$  does not always generalize to a similar value of  $E_{out}$ . Generalization is a key issue in learning. One can define the **generalization error** as the discrepancy between  $E_{in}$  and  $E_{out}$ .

The Hoeffding Inequality (HI) furnished a way to characterize the generalization error with a probabilistic bound:

$$P\left(\left|E_{in}(g) - E_{out}(g)\right| > \varepsilon\right) \leq 2Me^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

We can rephrase this as follows: given a **tolerance level**  $\delta$ , one can assert with probability *at least*  $1 - \delta$  that:

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

(\*) This is known as a *generalization bound* because it bounds  $E_{out}$  in terms of  $E_{in}$ .

**Pf.**

By HI, we have: with probability at least  $1 - 2Me^{-2\varepsilon^2 N}$ ,  $|E_{out} - E_{in}| \leq \varepsilon$ .

# Theory of Generalization

- We have previously discussed how the value of  $E_{in}$  does not always generalize to a similar value of  $E_{out}$ . Generalization is a key issue in learning. One can define the **generalization error** as the discrepancy between  $E_{in}$  and  $E_{out}$ .

The Hoeffding Inequality (HI) furnished a way to characterize the generalization error with a probabilistic bound:

$$P\left(\left|E_{in}(g) - E_{out}(g)\right| > \varepsilon\right) \leq 2Me^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

We can rephrase this as follows: given a **tolerance level**  $\delta$ , one can assert with probability *at least*  $1 - \delta$  that:

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

(\*) This is known as a *generalization bound* because it bounds  $E_{out}$  in terms of  $E_{in}$ .

**Pf.**

By HI, we have: with probability at least  $1 - \delta$ ,  $|E_{out} - E_{in}| \leq \varepsilon$ .

This implies that  $E_{out} \leq E_{in} + \varepsilon$ ; we then identify  $\delta = 2Me^{-2\varepsilon^2 N}$ ; solving for  $\varepsilon$  yields the result.



# Theory of Generalization

$$P\left(\left|E_{in}(g) - E_{out}(g)\right| > \varepsilon\right) \leq 2Me^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

- Given a **tolerance level**  $\delta$ , one can assert with probability *at least*  $1 - \delta$  that:

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

- This is known as a *generalization bound* because it bounds  $E_{out}$  in terms of  $E_{in}$ .

(\*) Observe that the “other” inequality suggested by the HI is also useful. Namely:  $|E_{out} - E_{in}| \leq \varepsilon$  also implies  $E_{out} \geq E_{in} - \varepsilon$  for all  $h \in H$ .

This second inequality assures us that every hypothesis with a higher  $E_{in}$  than  $g$  (our chosen model) will also have a comparably higher  $E_{out}$ .

# Theory of Generalization

$$P\left(\left|E_{in}(g) - E_{out}(g)\right| > \varepsilon\right) \leq 2Me^{-2\varepsilon^2 N} \text{ for any } \varepsilon > 0$$

- Given a **tolerance level**  $\delta$ , one can assert with probability *at least*  $1 - \delta$  that:

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

- This is known as a *generalization bound* because it bounds  $E_{out}$  in terms of  $E_{in}$ .

(\*) Notice that the “error bar”,  $\sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$ , in the bound above, depends explicitly on  $M$ , the size of the hypothesis set  $H$ . When  $M$  is infinite this bound becomes meaningless. Next we consider how to deal with infinite  $H$ .

# Theory of Generalization: Effective Number of Hypotheses

- We begin by defining the **growth function** (i) – a combinatorial quantity that captures how different the hypotheses in  $H$  are, and hence how much overlap they contain.
- Next we give a bound for the growth function (ii); finally, we show that we can subsequently replace  $M$  in the generalization bound with the growth function (iii). Together, these steps will yield a *generalization bound* that applies to the case when  $H$  is finite or *infinite*.



# Theory of Generalization: Effective Number of Hypotheses

- We begin by defining the **growth function** – a combinatorial quantity that captures how different the hypotheses in  $H$  are, and hence how much overlap they contain.
- Next we give a bound for the growth function; finally, we show that we can subsequently replace  $M$  in the generalization bound with the growth function. Together, these steps will yield a *generalization bound* that applies to the case when  $H$  is finite or *infinite*.
- For mathematical simplicity, consider the case of binary target functions, so that for each  $h \in H$ ,  $h$  maps  $X$  to  $\{-1, +1\}$ .
- If  $h$  is applied to a finite sample of points  $\mathbf{x}_1, \dots, \mathbf{x}_N \in X$ , we get an  $N$ -tuple:  $h(\mathbf{x}_1), \dots, h(\mathbf{x}_N)$  of  $+/-1$ 's.
- We call such an  $N$ -tuple a **dichotomy** on  $\mathbf{x}_1, \dots, \mathbf{x}_N$ , since it *splits*  $\mathbf{x}_1, \dots, \mathbf{x}_N$  into two groups: those whose points for which  $h$  is  $-1$  and those for which  $h$  is  $+1$ . In this way, each  $h \in H$  generates a dichotomy on  $\mathbf{x}_1, \dots, \mathbf{x}_N$ .

# Theory of Generalization: Effective Number of Hypotheses

**Definition.** Let  $\mathbf{x}_1, \dots, \mathbf{x}_N \in X$ . The *dichotomies* generated by the hypothesis set  $H$  on these points are defined by:

$$H(\mathbf{x}_1, \dots, \mathbf{x}_N) = \left\{ (h(\mathbf{x}_1), \dots, h(\mathbf{x}_N)) \mid h \in H \right\}$$

- One can think of the dichotomies  $H(\mathbf{x}_1, \dots, \mathbf{x}_N)$  as a set of hypotheses just like  $H$  is, except that the hypotheses are seen through the eyes of  $N$  points only. A larger  $H(\mathbf{x}_1, \dots, \mathbf{x}_N)$  signifies a more “diverse”  $H$ .

# Theory of Generalization: Effective Number of Hypotheses

**Definition.** Let  $\mathbf{x}_1, \dots, \mathbf{x}_N \in X$ . The *dichotomies* generated by the hypothesis set  $H$  on these points are defined by:

$$H(\mathbf{x}_1, \dots, \mathbf{x}_N) = \left\{ (h(\mathbf{x}_1), \dots, h(\mathbf{x}_N)) \mid h \in H \right\}$$

- One can think of the dichotomies  $H(\mathbf{x}_1, \dots, \mathbf{x}_N)$  as a set of hypotheses just like  $H$  is, except that the hypotheses are seen through the eyes of  $N$  points only. A larger  $H(\mathbf{x}_1, \dots, \mathbf{x}_N)$  signifies a more “diverse”  $H$ .

**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

$$m_H = \max_{\mathbf{x}_1, \dots, \mathbf{x}_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

(\*) In other words,  $m_H(N)$  is the maximum number of dichotomies that can be generated by  $H$  on any  $N$  points.



# Theory of Generalization: Effective Number of Hypotheses

**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

$$m_H = \max_{x_1, \dots, x_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

(\*) In other words,  $m_H(N)$  is the maximum number of dichotomies that can be generated by  $H$  on any  $N$  points.

- To compute  $m_H(N)$ , we consider all possible choices of  $N$  points  $\mathbf{x}_1, \dots, \mathbf{x}_N$  from  $X$  and pick the one that gives us the most dichotomies. Like  $M$ ,  $m_H(N)$  is a measure of the number of hypotheses in  $H$ , except that a hypothesis is now considered on  $N$  points instead of the entire  $X$ .

It follows that  $m_H(N) \leq 2^N$ . (why?)

- If  $H$  is capable of generating all possible dichotomies on  $\mathbf{x}_1, \dots, \mathbf{x}_N$ , then  $H(\mathbf{x}_1, \dots, \mathbf{x}_N) = \{-1, +1\}^N$  and we say that  $H$  can **shatter**  $\mathbf{x}_1, \dots, \mathbf{x}_N$ . This shows that  $H$  is as diverse as can be on this particular sample.

# Theory of Generalization: Effective Number of Hypotheses

**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

$$m_H = \max_{x_1, \dots, x_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

**Example.** If  $X$  is the Euclidean plane, and  $H$  is a 2-D perceptron, what are  $m_H(3)$  and  $m_H(4)$ ?

# Theory of Generalization: Effective Number of Hypotheses

**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

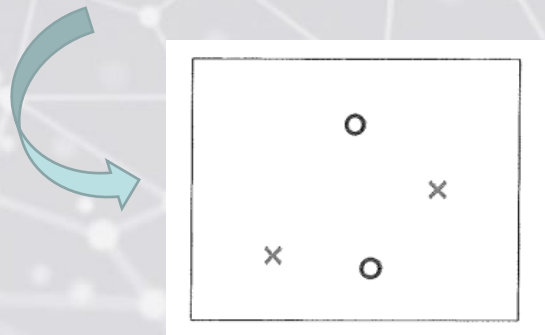
$$m_H = \max_{x_1, \dots, x_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

**Example.** If  $X$  is the Euclidean plane, and  $H$  is a 2-D perceptron, what are  $m_H(3)$  and  $m_H(4)$ ?

$m_H(3) = 8$  for a perceptron, because it can shatter all possible combinations of 3 points, which yields 8 total dichotomies.

$m_H(4) = 14$  (less obvious); notice that there are  $2^4=16$  possible dichotomies for 4 points; recall that the XOR (and its symmetric counterpart) cannot be shattered by a 2-D perceptron.





# Theory of Generalization: Effective Number of Hypotheses

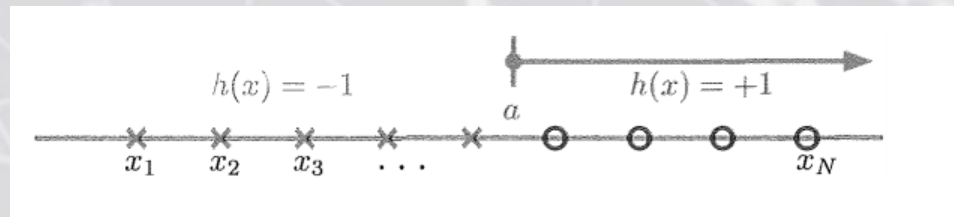
**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

$$m_H = \max_{x_1, \dots, x_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

**Example.** Consider  $m_H(N)$  for the following cases:

- (1) *Positive rays*:  $H$  consists of all hypotheses  $h: \mathbb{R} \rightarrow \{-1, +1\}$  of the form  $h(x) = \text{sign}(x-a)$ , i.e. the hypotheses are defined in a 1-D input space, and they return -1 to the left of some value  $a$  and +1 to the right of  $a$ .



Q: What is  $m_H(N)$  in this case?

# Theory of Generalization: Effective Number of Hypotheses

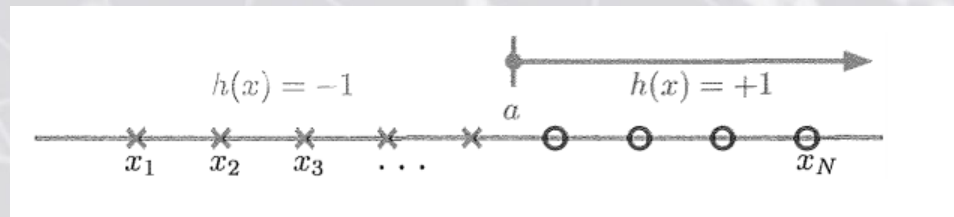
**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

$$m_H = \max_{x_1, \dots, x_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

**Example.** Consider  $m_H(N)$  for the following cases:

- (1) *Positive rays*:  $H$  consists of all hypotheses  $h: \mathbb{R} \rightarrow \{-1, +1\}$  of the form  $h(x) = \text{sign}(x-a)$ , i.e. the hypotheses are defined in a 1-D input space, and they return -1 to the left of some value  $a$  and +1 to the right of  $a$ .



Q: What is  $m_H(N)$  in this case? A:  $m_H(N) = N + 1$ ; since  $m_H(N)$  is defined as the maximum number of dichotomies, and as we vary  $a$  we get  $N+1$  different dichotomies.

# Theory of Generalization: Effective Number of Hypotheses

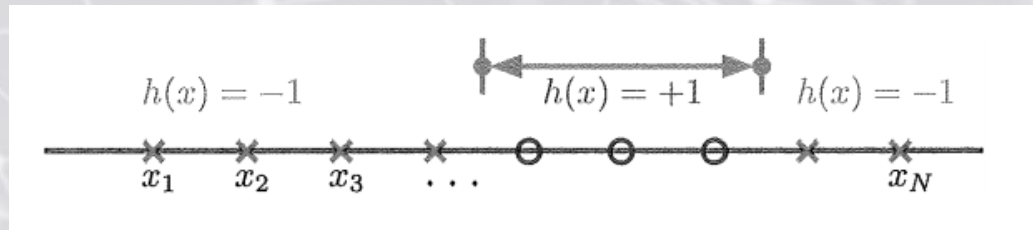
**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

$$m_H = \max_{x_1, \dots, x_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

**Example.** Consider  $m_H(N)$  for the following cases:

(2) *Positive Intervals:*  $H$  consist of all hypotheses in one dimension that return +1 within some interval and -1 otherwise. Each hypothesis is specified by two end points.



Q: What is  $m_H(N)$ ?



# Theory of Generalization: Effective Number of Hypotheses

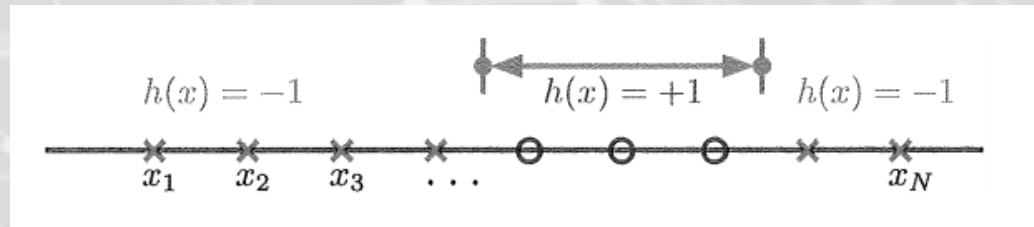
**Definition.** The **growth function** is defined for a hypothesis set  $H$ , by:

$$m_H = \max_{x_1, \dots, x_N \in X} |H(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

Where  $|\cdot|$  denotes the cardinality of a set.

**Example.** Consider  $m_H(N)$  for the following cases:

(2) *Positive Intervals*:  $H$  consist of all hypotheses in one dimension that return +1 within some interval and -1 otherwise. Each hypothesis is specified by two end points.



Q: What is  $m_H(N)$ ? A: Given  $N$  points, the line is split by the points into  $N+1$  regions. The dichotomy we get is decided by which two regions contain the end values of the interval resulting in  $C(N+1, 2)$  different dichotomies; lastly, if both end points fall in the same region, the resulting hypothesis is the constant -1 regardless of which region it is. This gives:  $m_H(N) = C(N+1, 2) + 1$ .

# Theory of Generalization: Effective Number of Hypotheses

- Even with a few relatively simple examples, we start to see that computing  $m_H(N)$  exactly can be challenging – if not impossible – in general. Instead, we prefer to compute a more amenable upper-bound based on the notion of a *break point*.

**Definition.** If no data of size  $k$  can be shattered by  $H$ , then  $k$  is said to be a **break point** for  $H$ .

(\*) Naturally, if  $k$  is a break point, then  $m_H(k) < 2^k$ . For example, for 2-D perceptrons,  $k=4$  is a break point. Consequently  $m_H(4) < 2^4=16$  (previously we noted that  $m_H(4)=14$ ).

- Next we use the break point  $k$  to derive a bound on the growth function  $m_H(N)$  for all values of  $N$ .

# Theory of Generalization: Bounding the Growth Function

- Crucially, if the condition  $m_H(N)=2^N$  breaks at any point, we can bound  $m_H(N)$  for all values of  $N$  by a simple polynomial based on this break point (for intuitive clarity:  $m_H(N)$  breaks for  $N=4$  for the 2-D perceptron, which means that  $m_H(N)$  is likewise bounded for  $N=5$ , etc.).

(\*) The fact that the bound is polynomial is significant. Absent a polynomial bound (due to the presence of a break point), the bound  $\sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$ , on the generalization error would not go to zero regardless of how many training examples  $N$  we have!



# Theory of Generalization: Bounding the Growth Function

**Theorem.** Given a break point  $k$ ,  $m_H(k) < 2^k$ , and it follows that:

$$m_H(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

(\*) We omit the proof for brevity; notice that the RHS is a polynomial in  $N$  of degree  $k-1$ . *Main point:* If  $H$  has a break point, we have what we want to ensure good generalization; a polynomial bound on  $m_H(N)$ . The smaller the break point, the better the bound.

# VC Dimension



V. Vapnik

- *VC dimension* is an important and deep concept in ML.
- VC dimension is a measure of the **capacity** (i.e. *complexity*, *expressive power*, *richness*, or *flexibility*) of a hypothesis set  $H$  that can be learned by a classification algorithm.

**Definition. VC dimension** (of a hypothesis set), denoted  $d_{vc}(H)$ , is the largest value of  $N$  for which  $m_H(N) = 2^N$ . In other words maximum number of points that can be shattered by  $H$ . If  $m_H(N) = 2^N$  for all  $N$ , then  $d_{vc}(H) = \infty$ .

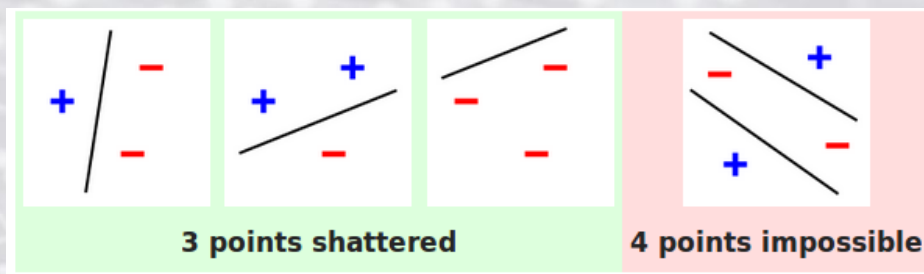
# VC Dimension

- Example: If  $H$  corresponds with the set of constant classifiers, its VC dimension is **zero**, since it cannot shatter even a single point.



# VC Dimension

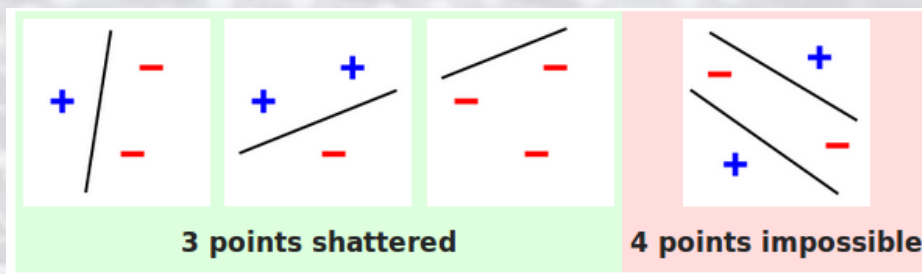
- Example: If  $H$  corresponds with the set of constant classifiers, its VC dimension is **zero**, since it cannot shatter even a single point.
- Example: If  $H$  is a straight line classifier (in 2-D). There exist sets of 3 points that can indeed be shattered using the model (any 3 points that are non-collinear can be shattered). However, no set of 4 points can be shattered. Thus the **VC dimension of a straight line is 3**.



Note we only show 3 of the  $2^3=8$  possible binary labelings for 3 points.

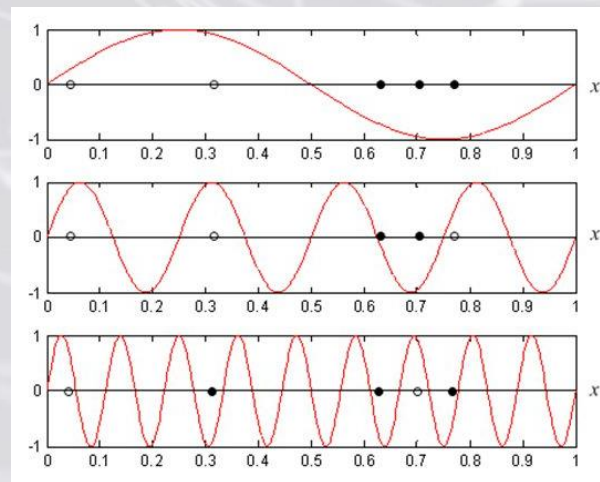
# VC Dimension

- Example: If  $H$  corresponds with the set of constant classifiers, its VC dimension is **zero**, since it cannot shatter even a single point.
- Example: If is a straight line classifier (in 2-D). There exist sets of 3 points that can indeed be shattered using the model (any 3 points that are non-collinear can be shattered). However, no set of 4 points can be shattered. Thus the **VC dimension of a straight line is 3**.



Note we only show 3 of the  $2^3=8$  possible binary labelings for 3 points.

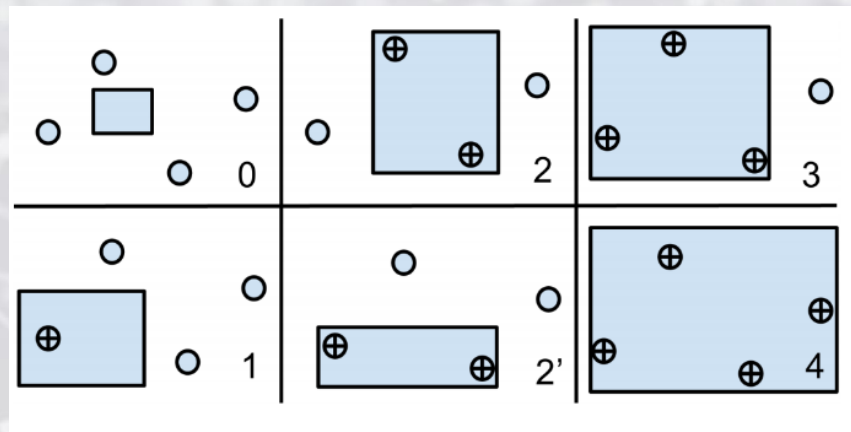
- Example: consider the set of hypotheses defined by a single-parameter “sine classifier”, i.e. for a certain parameter  $\theta$ , the classifier  $f_\theta$  returns 1 if the input number  $x$  is larger than  $\sin(\theta x)$  and 0 otherwise.
- The VC dimension of  $f$  is **infinite**, since it can shatter the set:  $\{2^{-m} \mid m \in \mathbb{N}\}$  for any positive  $m$ .



(\*) Note the last example shows, importantly, that VC dimension is not directly related to the number of model parameters!

# VC Dimension

- Another example: the VC dimension of a *rectangle* in  $\mathbf{R}^2$  (where the rectangle encompasses all data of a particular class).



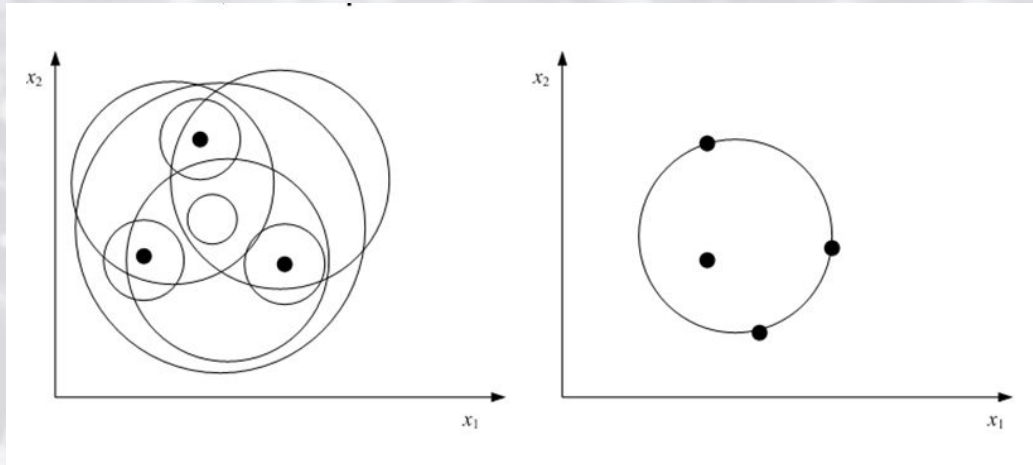
- The diagram shows that **a rectangle shatters at least 4 points in the plane;** for 5 points there exists a counter-example (try it).
- In general, the **VC dimension of a hyperplane in  $\mathbf{R}^d$  is  $d+1$ .**  
(e.g. in  $\mathbf{R}^2$  we previously showed a hyperplane has VC dimension = 3).



# VC Dimension

- Another example: the VC dimension of a *spherical indicator function* in  $\mathbf{R}^2$  (where the sphere encompasses all data of a particular class).

$$f(\mathbf{x}, \mathbf{c}, r) = I\left((\mathbf{x} - \mathbf{c})^2 \leq r^2\right)$$



- Three points in  $\mathbf{R}^2$  can be shattered, but four cannot; thus **VC dim(H) = 3** in  $\mathbf{R}^2$  (useful for nearest neighbors classifiers, radial basis functions).
- The VC dimension of a d-dimensional perceptron is d+1.

For the curious, on VC dimension of NNs:

<https://pdfs.semanticscholar.org/48ff/5cecb0f23714c50f2641e8565a3dba5eec88.pdf>

# VC Dimension

Again, here is our polynomial bound for  $m_H(\mathbb{N})$ , based on a break point value  $k$ . Given a break point  $k$ ,  $m_H(k) < 2^k$ , and it follows that:

$$m_H(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

- If  $d_{vc}(H)$  is the VC dimension of  $H$ , then  $k=d_{vc}+1$  is a break point for  $m_H$ , since  $m_H(\mathbb{N})$  cannot equal  $2^N$  for any  $N > d_{vc}$  by definition. Also, no smaller break point exists since  $H$  can shatter  $d_{vc}$  points, hence it can also shatter any subset of these points.

# VC Dimension

Again, here is our polynomial bound for  $m_H(\mathbb{N})$ , based on a break point value  $k$ . Given a break point  $k$ ,  $m_H(k) < 2^k$ , and it follows that:

$$m_H(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

- If  $d_{vc}(H)$  is the VC dimension of  $H$ , then  $k=d_{vc}+1$  is a break point for  $m_H$ , since  $m_H(\mathbb{N})$  cannot equal  $2^N$  for any  $N > d_{vc}$  by definition. Also, no smaller break point exists since  $H$  can shatter  $d_{vc}$  points, hence it can also shatter any subset of these points.

- Putting this together, it follows that:

$$m_H(N) \leq \sum_{i=0}^{d_{vc}} \binom{N}{i}$$

Thus, the VC dimension is the order of the polynomial bound on  $m_H(\mathbb{N})$  (since no smaller break point than  $k = d_{vc}+1$  exists). In fact one can show that:  $M_H(\mathbb{N}) \leq N^{d_{vc}+1}$ .




# VC Dimension

- Putting this together, it follows that:

$$m_H(N) \leq \sum_{i=0}^{d_{vc}} \binom{N}{i}$$

Thus, the VC dimension is the order of the polynomial bound on  $m_H(N)$  (since no smaller break point than  $k = d_{vc} + 1$  exists). In fact one can show that:  $m_H(N) \leq N^{d_{vc}+1}$  (using induction).

Notice that the growth function is bounded in terms of VC dimension, as desired. Our last required task is to replace the number of hypotheses  $M$  in the generalization bound with the growth function  $m_H(N)$ .


$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N} \ln \frac{2M}{\delta}}$$

(\*) When  $d_{vc}(H)$  is finite this procedure will basically work; however, we still need to sort out the case when  $d_{vc}(H) = \infty$ .

# The VC Generalization Bound

Theorem. (**VC generalization bound**). For any tolerance  $\delta > 0$ ,

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

with probability  $\geq 1 - \delta$ .

(\*) The VC generalization bound is the most important mathematical result in the theory of learning. It establishes the feasibility of learning – even in the case of infinite hypothesis sets.

• Since  $m_H(2N)$  is a polynomial of order  $d_{VC}$  in  $N$ , with enough data, each and every hypothesis in an infinite  $H$  **with a finite VC dimension** will generalize well from  $E_{in}$  to  $E_{out}$ .

(\*) Notice that with an infinite  $d_{VC}$ , no matter how large the data set is, we cannot make generalization conclusion from  $E_{in}$  to  $E_{out}$  based on the VC analysis.

# The VC Generalization Bound

Theorem. (**VC generalization bound**). For any tolerance  $\delta > 0$ ,

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

with probability  $\geq 1 - \delta$ .

- The formal proof of the VC generalization bound is technical, so let's only consider a very brief sketch.

The bulk of the VC proof deals with how to account for overlaps of hypotheses.

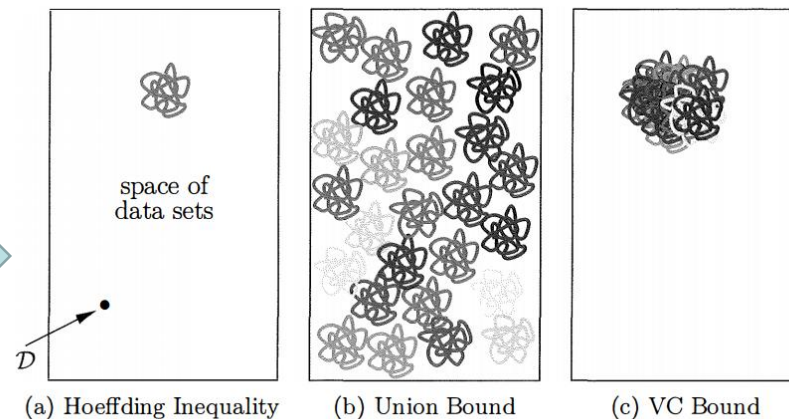


Figure 2.2: Illustration of the proof of the VC bound, where the 'canvas' represents the space of all data sets, with areas corresponding to probabilities. (a) For a given hypothesis, the colored points correspond to data sets where  $E_{in}$  does not generalize well to  $E_{out}$ . The Hoeffding Inequality guarantees a small colored area. (b) For several hypotheses, the union bound assumes no overlaps, so the total colored area is large. (c) The VC bound keeps track of overlaps, so it estimates the total area of bad generalization to be relatively small.



# The VC Generalization Bound: Interpretation

Theorem. (**VC generalization bound**). For any tolerance  $\delta > 0$ ,

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

with probability  $\geq 1 - \delta$ .

- The VC generalization bound is a universal result in the sense that it applies to all hypothesis sets, learning algorithms, input spaces, probability distributions, and binary target functions (it can be extended to other types of target functions as well).

Because of this generality, the VC bound is quite loose. Why? Three main reasons:

- (1) It uses the Hoeffding Inequality which already incorporates some slack.
- (2) Using  $m_H(N)$  to quantify the number of dichotomies on  $N$  points, regardless of which  $N$  points are in the data set, gives us a worst-case estimate.
- (3) Bounding  $m_H(N)$  by a simple polynomial of order  $d_{VC}$  also contributes to the slack in the VC bound.



# The VC Generalization Bound: Interpretation

Theorem. (**VC generalization bound**). For any tolerance  $\delta > 0$ ,

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

with probability  $\geq 1 - \delta$ .

- On-going analysis has only slightly diminished the looseness of the VC bound.

Q: So why bother using it?

A: (i) VC analysis establishes the feasibility of learning for infinite hypothesis sets – the only ones we use in practice.

(ii) Despite the bound being loose, it tends to be equally loose for different learning models. It is therefore effective for comparing generalization performance of these models.

# The VC Generalization Bound: Sample Complexity

Theorem. (**VC generalization bound**). For any tolerance  $\delta > 0$ ,

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}}$$

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4((2N)^{d_{VC}} + 1)}{\delta}}$$

with probability  $\geq 1 - \delta$ .



VC bound as explicit  
function of  $d_{VC}$

- The **sample complexity** denotes how many training examples  $N$  are needed to achieve a certain generalization performance.
- This performance is specified by two parameters:  $\epsilon$  and  $\delta$ . The *error tolerance*  $\epsilon$  determines the allowed generalization error, and the confidence parameter  $\delta$  determines how often the error tolerance is violated. How fast  $N$  grows as  $\epsilon$  and  $\delta$  become smaller indicates how much data are needed to render good generalization.

Solving the inequality:  $\sqrt{\frac{8}{N} \ln \frac{4m_H(2N)}{\delta}} \leq \epsilon$  for  $N$  gives:  $N \geq \frac{8}{\epsilon^2} \ln \left( \frac{4((2N)^{d_{VC}} + 1)}{\delta} \right)$

# The VC Generalization Bound: Sample Complexity

- The **sample complexity** denotes how many training examples  $N$  are needed to achieve a certain generalization performance.
- This performance is specified by two parameters:  $\epsilon$  and  $\delta$ . The *error tolerance*  $\epsilon$  determines the allowed generalization error, and the confidence parameter  $\delta$  determines how often the error tolerance is violated. How fast  $N$  grows as  $\epsilon$  and  $\delta$  become smaller indicates how much data are needed to render good generalization.

$$N \geq \frac{8}{\epsilon^2} \ln \left( \frac{4 \left( (2N)^{d_{VC}} + 1 \right)}{\delta} \right)$$

**Example.** Suppose that we have a learning model with  $d_{VC}=3$  (e.g. a line in  $\mathbb{R}^2$ ) and we would like the generalization error to be at most 0.1 with confidence 90% (i.e.  $\epsilon = 0.1$  and  $\delta = 0.1$ ). How big does our data set need to be?

We need: 
$$N \geq \frac{8}{0.1^2} \ln \left( \frac{4 \left( (2N)^3 + 1 \right)}{0.1} \right) \approx 30,000$$



# The VC Generalization Bound: Model Complexity

- Sample complexity fixes the performance parameters  $\epsilon$  (generalization) and  $\delta$  (confidence parameter) and estimates how many examples  $N$  are needed.

In most practical situations, however, we are given a fixed data set  $D$ , so  $N$  is also fixed. In this case, the relevant question is what performance can we expect given this particular  $N$ . The VC generalization bound quantifies this. With probability at least  $1 - \delta$ :

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4((2N)^{d_{vc}} + 1)}{\delta}}$$

**Example.** Suppose that  $N=100$  and we have a 90% confidence requirement ( $\delta=0.1$ ). We could ask what error bar can we offer with this confidence if  $H$  has  $d_{VC}=1$ .

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{100} \ln \frac{4((201))}{0.1}} \approx E_{in}(g) + 0.848$$

With confidence  $\geq 90\%$ .



# The VC Generalization Bound: Model Complexity

- Let's consider the two parts that make up the bound on  $E_{out}$  in the VC generalization bound more closely.

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4((2N)^{d_{vc}} + 1)}{\delta}}$$

The first part is  $E_{in}$ , and the second part is a term that increases as the VC dimension of  $H$  increases.

$$E_{out}(g) \leq E_{in}(g) + \Omega(N, H, \delta)$$

where

$$\Omega(N, H, \delta) = \sqrt{\frac{8}{N} \ln \frac{4(m_H(2N))}{\delta}} \leq \sqrt{\frac{8}{N} \ln \frac{4((2N)^{d_{vc}} + 1)}{\delta}}$$

One way to think of  $\Omega(N, H, \delta)$  is that it is a penalty for model complexity. It penalizes us by worsening the bound on  $E_{out}$  when we use a more complex  $H$  (i.e. larger  $d_{VC}$ ). The penalty  $\Omega(N, H, \delta)$  gets worse if we insist on high confidence (i.e. lower  $\delta$ ), and it gets better when we have more training examples, as we would expect.

# The VC Generalization Bound: Model Complexity

- The first part is  $E_{in}$ , and the second part is a term that increases as the VC dimension of  $H$  increases.

$$E_{out}(g) \leq E_{in}(g) + \Omega(N, H, \delta)$$

- One way to think of  $\Omega(N, H, \delta)$  is that it is a penalty for model complexity. It penalizes us by worsening the bound on  $E_{out}$  when we use a more complex  $H$  (i.e. larger  $d_{VC}$ ). The penalty  $\Omega(N, H, \delta)$  gets worse if we insist on high confidence (i.e. lower  $\delta$ ), and it gets better when we have more training examples, as we would expect.

Although  $\Omega(N, H, \delta)$  goes up when  $H$  has higher VC dimension,  $E_{in}$  is likely to go down with A higher VC dimension as we have more choices within  $H$  to fit the data. Therefore, we have a tradeoff: more complex models help  $E_{in}$  and hurt  $\Omega(N, H, \delta)$ .

(\*) The optimal model is a compromise that minimizes a combination of the two.

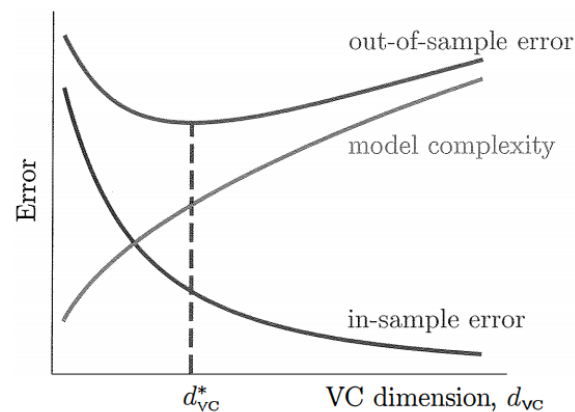


Figure 2.3: When we use a more complex learning model, one that has higher VC dimension  $d_{VC}$ , we are likely to fit the training data better resulting in a lower in sample error, but we pay a higher penalty for model complexity. A combination of the two, which estimates the out of sample error, thus attains a minimum at some intermediate  $d_{VC}^*$ .

# Generative and Discriminative Models

- A **discriminative classifier** directly fits the class posterior, e.g. SVMs, decision trees, logistic regression:

$$p(y = c | \mathbf{x})$$

- A **generative classifier** specifies how to generate data using the *class-conditional density*  $p(\mathbf{x} | y = c)$  and the class prior  $p(y = c)$ , e.g., Naïve Bayes, GMM, HMM, *generative adversarial networks* (GAN).

$$p(y = c | \mathbf{x}) = \frac{p(y = c) p(\mathbf{x} | y = c)}{\sum_{c'} p(y = c' | \boldsymbol{\theta}) p(\mathbf{x} | y = c')}$$

# Bayesian Concept Learning

- We begin with a simple example (due to Tenenbaum\*) of *Bayesian concept learning* called **the number game**.
- The game proceeds as follows: I choose some simple arithmetical concept  $C$ , such as “prime number” or “a number between 1 and 10”; I then give you a series of randomly chosen positive examples:  $D = \{x_1, \dots, x_N\}$  drawn from  $C$ , and ask you whether some new test case  $\hat{x}$  belong to  $C$ , i.e., I ask you to classify  $\hat{x}$ .

\*<https://dspace.mit.edu/bitstream/handle/1721.1/16714/42471842-MIT.pdf>

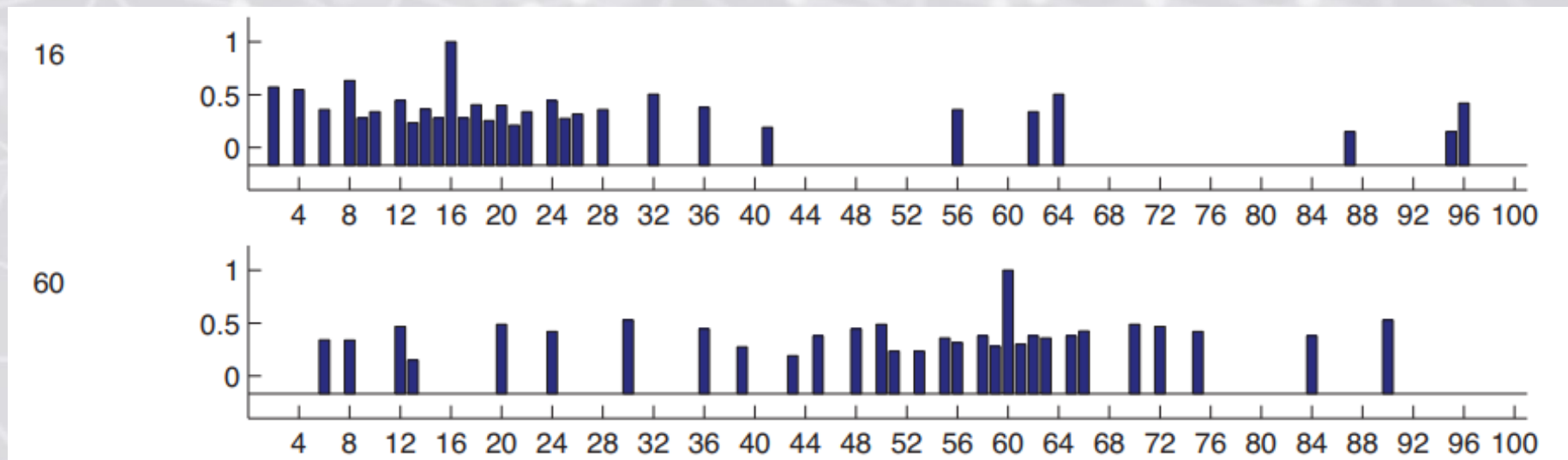


# Bayesian Concept Learning

- We begin with a simple example (due to Tenenbaum\*) of *Bayesian concept learning* called **the number game**.
- The game proceeds as follows: I choose some simple arithmetical concept  $C$ , such as “prime number” or “a number between 1 and 10”; I then give you a series of randomly chosen positive examples:  $D = \{x_1, \dots, x_N\}$  drawn from  $C$ , and ask you whether some new test case  $\hat{x}$  belong to  $C$ , i.e., I ask you to classify  $\hat{x}$ .
- For simplicity, let's consider the case that we are dealing with the domain of integers between 1 and 100. Now suppose I tell you “16” is a positive example of the concept. What other numbers do you think are positive? 17? 6? 32? 99? It's obviously hard to tell with one example.
- Presumably numbers that are *similar* in some sense to 16 are more likely. But similar in what way? 17 is “close by”; 6 has a digit in common; 32 is similar because it also even and a power of 2; 99 on the other hand seems “dissimilar.”

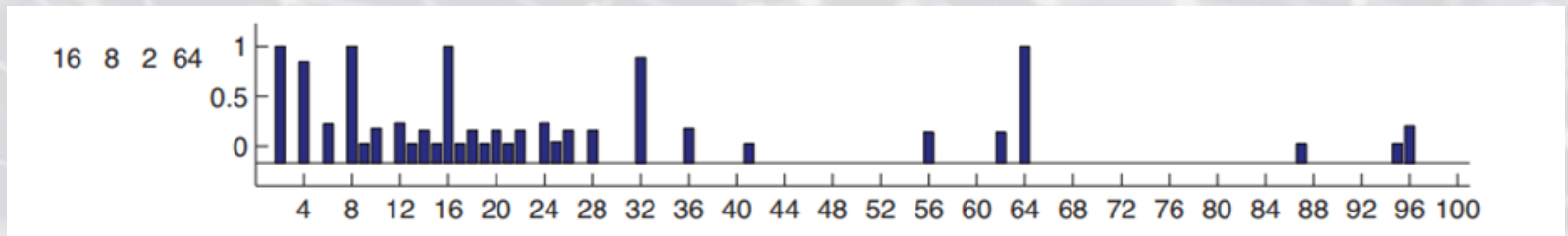
# Bayesian Concept Learning

- Thus some numbers are more likely than others. Naturally, we can encapsulate this concept via probability distribution:  $p(\hat{x} \mid D)$ , which is the probability that  $\hat{x} \in C$  given the data  $D$  for any  $\hat{x} \in \{1, \dots, 100\}$ . This is called the **posterior predictive distribution**.

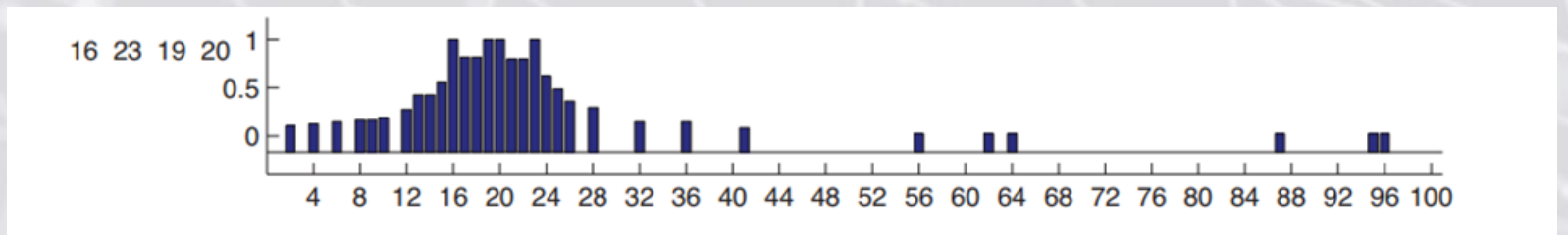


# Bayesian Concept Learning

- If I tell you that in addition, 8, 2 and 64 are also positive examples. Then you are likely to guess that the hidden concept is “powers of 2.” This is an example of **induction**.
- Given this hypothesis, the predictive distribution is quite specific, and puts most of its mass on powers of 2.



- Conversely, if I tell you that the data is  $D = \{16, 23, 19, 20\}$ , you will get a different kind of **generalization gradient**.



# Bayesian Concept Learning

- How do we arrange these ideas formally? In the framework of induction, we suppose that there is a **hypothesis space** of concepts  $H$ , such as: odd numbers, even numbers, all numbers between 1 and 100, powers of two, all numbers ending in the digit  $j$  (e.g.  $j = 3$ ).
- The subsets of  $H$  that are consistent with the data  $D$  is called the **version space**. As we observe data the version space becomes smaller and smaller, and we presumably become increasingly certain about the underlying concept.



# Bayesian Concept Learning

- How do we arrange these ideas formally? In the framework of induction, we suppose that there is a **hypothesis space** of concepts  $H$ , such as: odd numbers, even numbers, all numbers between 1 and 100, powers of two, all numbers ending in the digit  $j$  (e.g.  $j = 3$ ).
- The subsets of  $H$  that are consistent with the data  $D$  is called the **version space**. As we observe data the version space becomes smaller and smaller, and we presumably become increasingly certain about the underlying concept.
- Notice that the version space is only part of the story. After observing  $D = \{16\}$ , for example, there are still many consistent rules. We still need to resolve the question of determining the probability that a new datum belongs to a particular concept, in addition to determining the most likely concept, given a set of observations. A Bayesian template will allow us to reasonably answer these questions.

# Bayesian Concept Learning

## Likelihood

- We need to formalize why, for instance, we might choose  $h_{two}$  := “powers of two” and not, say  $h_{even}$  := “even numbers” after observing  $D = \{16, 8, 2, 64\}$ , despite the fact that both hypotheses are consistent with the evidence.

(\*) The key intuition is that we want to eschew **suspicious coincidences**.

- To this end, let us assume that examples are sampled uniformly at random from the **extension of a concept** (where the extension of a concept is just the set of numbers that belong to it).

- Given this assumption, the probability of independently sampling  $N$  items (with replacement) from hypothesis  $h$  is given by:

$$P(D | h) = \left[ \frac{1}{\text{size}(h)} \right]^N = \left[ \frac{1}{|h|} \right]^N$$

# Bayesian Concept Learning

## Likelihood

- Given this assumption, the probability of independently sampling N items (with replacement) from hypothesis  $h$  is given by:

$$P(D | h) = \left[ \frac{1}{\text{size}(h)} \right]^N = \left[ \frac{1}{|h|} \right]^N$$

- Notice that this equation encompasses *Occam's Razor*, in that the model favors the simplest (i.e. smallest) hypothesis consistent with the data.

Example: Let  $D = \{16\}$ ; then  $p(D | h_{\text{two}}) = 1/6$ , since there are only 6 powers of two less than 100, but  $p(D | h_{\text{even}}) = 1/50$ . So the likelihood that  $h = h_{\text{two}}$  is higher than if  $h = h_{\text{even}}$ .

- After 4 examples, the likelihood of  $h_{\text{two}}$  is  $(1/6)^4 = 7.7 \cdot 10^{-4}$ , whereas the likelihood of even is  $(1/50)^4 = 1.6 \cdot 10^{-7}$ . This is a **likelihood ratio** of almost 5000:1 in favor of  $h_{\text{two}}$ . This quantifies our earlier intuition that  $D = \{16, 8, 2, 64\}$  would indeed be a very suspicious coincidence if generated by  $h_{\text{even}}$ .



# Bayesian Concept Learning

## Prior

- Suppose that  $D = \{16, 8, 2, 64\}$ . Given this data, the concept  $h' =$  “powers of two except 32” is more likely than  $h =$  “power of two”, since  $h'$  does not need to explain the coincidence that 32 is missing from the set of examples.
- However, the hypothesis  $h' =$  “powers of 2 except 32” seems conceptually unnatural. We can capture this intuition by assigning low prior probability to unnatural concepts. The prior – while *potentially subjective* – is the mechanism by which background knowledge can be brought to bear on a problem.
- In practice, priors can be **subjective** (but nevertheless a prior is usually informed by experience or background knowledge), **data-driven**, or **uninformative** (when, without compelling evidence, there is no reason to favor one outcome over another).



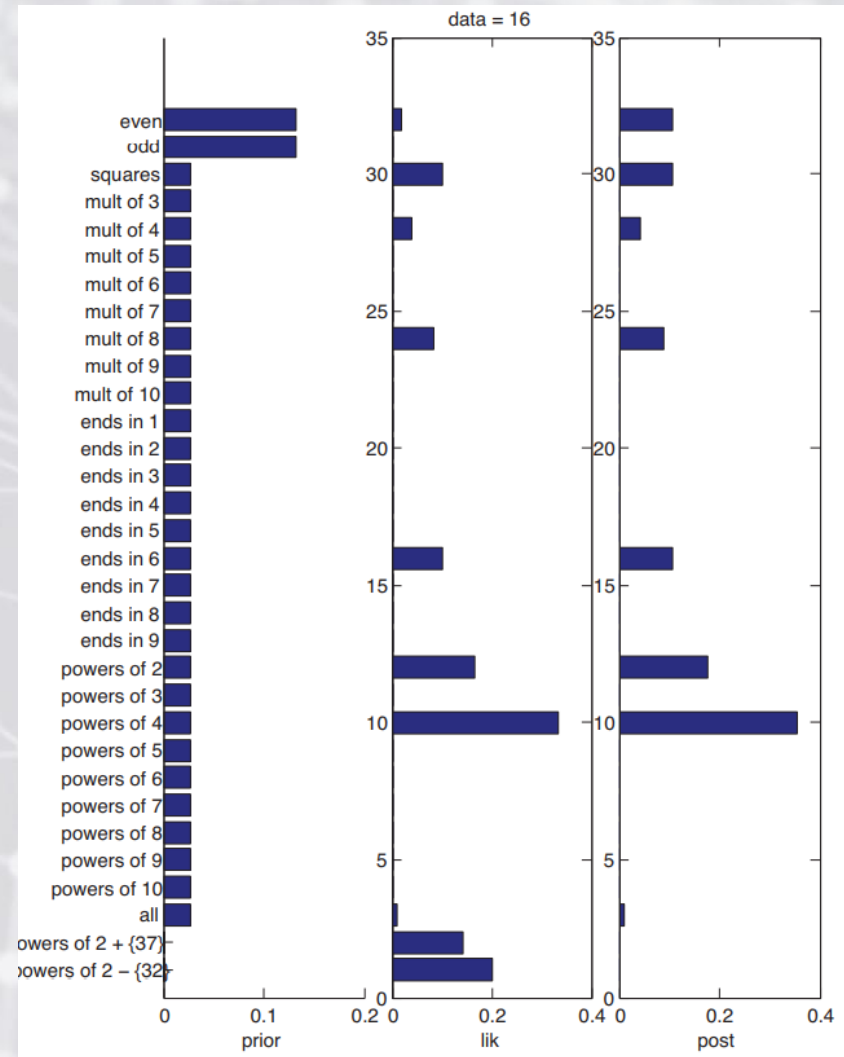
# Bayesian Concept Learning

## Posterior

- Recall that the posterior is the likelihood times the prior, normalized:

$$p(h|D) = \frac{p(D|h)p(h)}{\sum_{h' \in H} p(D,h')} = \frac{p(h)I(D \in h)/|h|^N}{\sum_{h' \in H} p(h')I(D \in h')/|h'|^N}$$

where  $I$  is the indicator function, so  $I(D \in Ch) = 1$  iff all the data are in the extension of the hypothesis  $h$ . The figure on the right shows the prior, likelihood and posterior after having observed  $D = \{16\}$ .

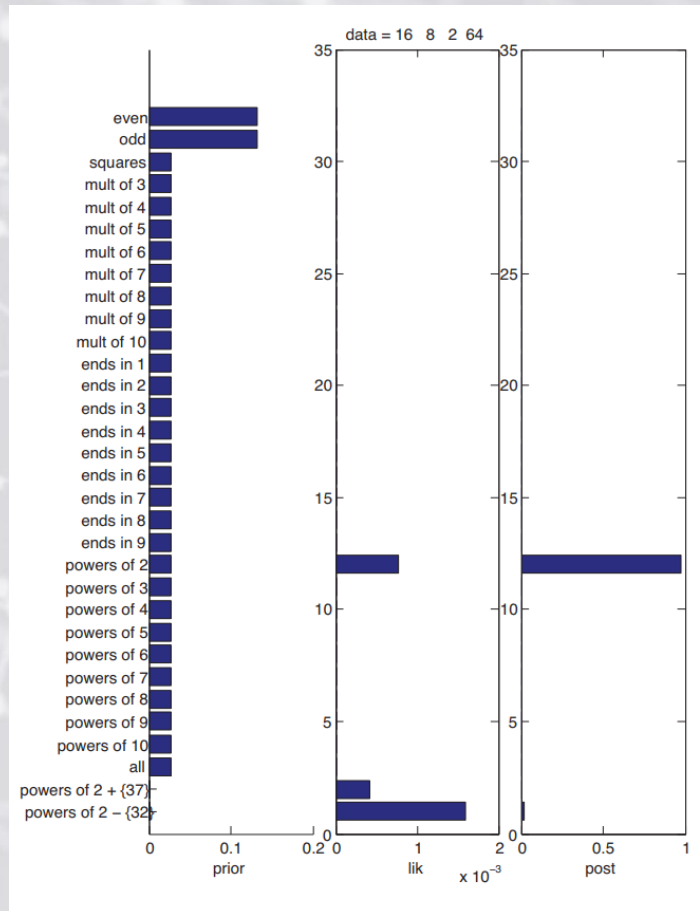


# Bayesian Concept Learning

## Posterior

$$p(h|D) = \frac{p(D|h)p(h)}{\sum_{h' \in H} p(D,h')} = \frac{p(h)I(D \in h)/|h|^N}{\sum_{h' \in H} p(h')I(D \in h')/|h'|^N}$$

- The figure plots the prior, likelihood and posterior after seeing  $D = \{16, 8, 2, 64\}$ . We see the likelihood is much more peaked on the powers of two concept, so this dominates the posterior. (This is akin to the learner having a **Gestalt moment**).



# Bayesian Concept Learning

## Posterior

- In general, when we have a sufficient amount of data, the posterior  $p(h | D)$  becomes peaked on a single concept (*viz.*, **the data overwhelms the prior**) – namely the *MAP* estimate. In other words:

$$p(h | D) \rightarrow \delta_{\hat{h}^{MAP}}(h)$$

Where  $\hat{h}^{MAP} = \operatorname{argmax}_h p(h | D)$  is the posterior mode, and  $\delta$  is the **Dirac measure** defined by:

$$\delta_x(A) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases}$$

(\*) Note that the MAP estimate can be written as:

$$\hat{h}^{MAP} = \operatorname{argmax}_h p(D | h) p(h) = \operatorname{argmax}_h \left[ \log p(D | h) + \log p(h) \right]$$

# Bayesian Concept Learning

## Posterior

$$P(D|h) = \left[ \frac{1}{\text{size}(h)} \right]^N = \left[ \frac{1}{|h|} \right]^N$$

$$\hat{h}^{MAP} = \underset{h}{\operatorname{argmax}} p(D|h) p(h) = \underset{h}{\operatorname{argmax}} \left[ \log p(D|h) + \log p(h) \right]$$

- In our example of the “number game” the prior stays constant and the likelihood term depends exponentially on N, and thus the MAP estimate converges toward the MLE (maximum likelihood estimate):

$$\hat{h}^{mle} = \underset{h}{\operatorname{argmax}} p(D|h) = \underset{h}{\operatorname{argmax}} \left[ \log p(D|h) \right]$$



## Posterior

# Bayesian Concept Learning

(\*) This formally shows an example for which the “data overwhelms the prior.” This captures a fundamental idea in ML: (for independent data) in general, **as the data sample grows in size, the MAP converges to the MLE.**

Pf.

$$\begin{aligned}\hat{h}^{MAP} &= \operatorname{argmax}_h p(D|h) p(h) \\ &= \operatorname{argmax}_h \left[ \log p(D|h) + \log p(h) \right] \\ &\propto \operatorname{argmax}_h \left[ N \log c_1 + c_2 \right]\end{aligned}$$

Ergo,

$$\lim_{N \rightarrow \infty} \hat{h}^{MAP} = \operatorname{argmax}_h \left[ \log p(D|h) \right] = \hat{h}^{mle}$$

(\*) Note that when the data set is small in size, then the prior “protects” us from incomplete observations. Both the MLE and MAP are consistent estimators, meaning that they converge to the correct hypothesis as the amount of data increases.

# Bayesian Concept Learning

## Posterior Predictive Distribution

- The posterior encompasses our internal **belief state** about the world. The way to test if our beliefs are justified is to use them to predict objectively observable quantities – this is the basis of the scientific method.

Specifically, the **posterior predictive distribution** in this context is given by:

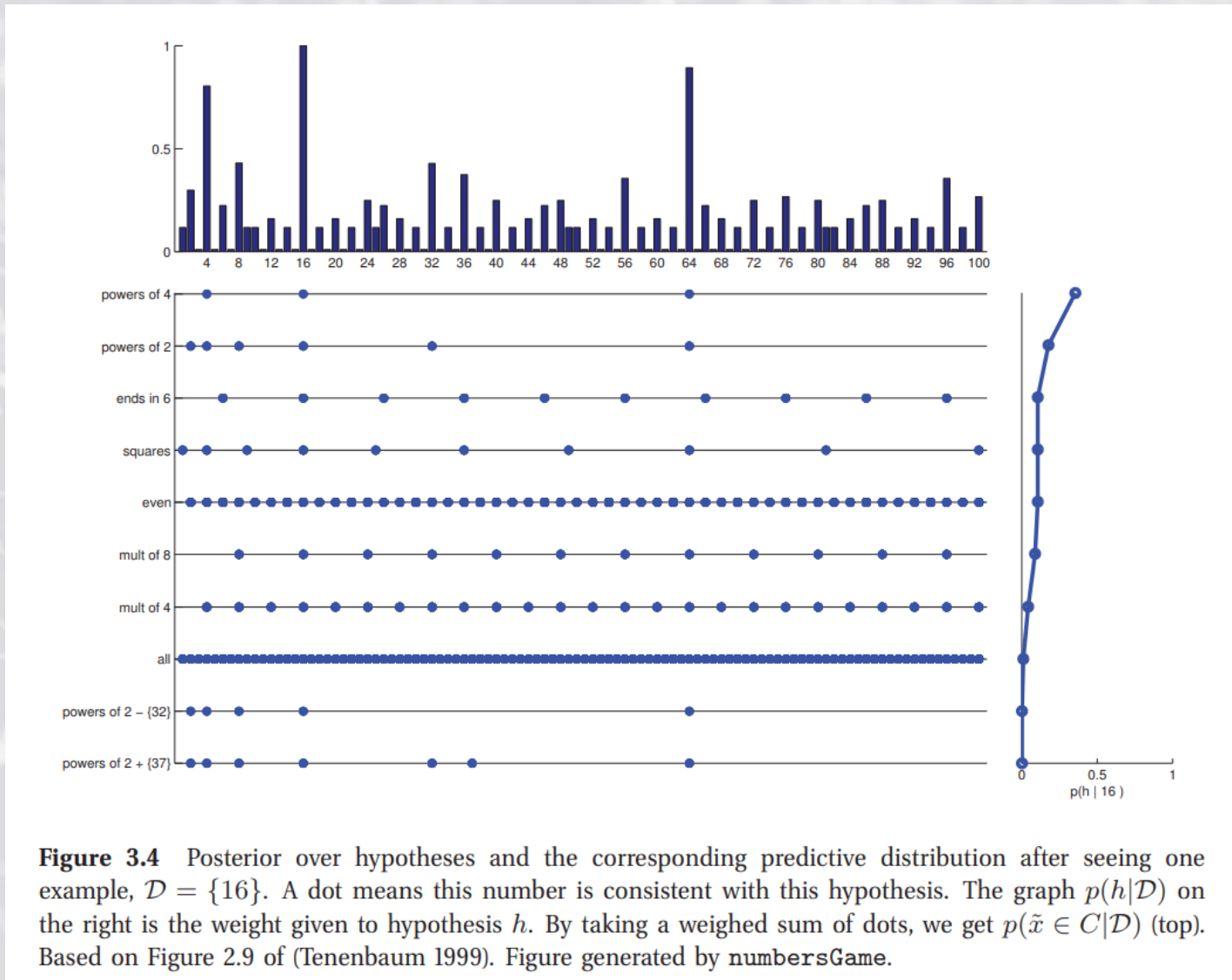
$$p(\tilde{x} \in C | D) = \sum_h p(y = 1 | \tilde{x}, h) p(h | D)$$

This is just the weighted average of the predictions of each individual hypothesis and is called **Bayes model averaging**.

# Bayesian Concept Learning

## Posterior Predictive Distribution

$$p(\tilde{x} \in C | D) = \sum_h p(y = 1 | \tilde{x}, h) p(h | D)$$



# Bayesian Concept Learning

## Posterior Predictive Distribution

$$p(\tilde{x} \in C | D) = \sum_h p(y=1 | \tilde{x}, h) p(h | D)$$

When we have a small and/or ambiguous data set, the posterior  $p(h | D)$  is vague, which induces a broad predictive distribution. However, once we have “figured things out”, the posterior becomes a delta function centered at the MAP estimate.



# Bayesian Concept Learning

## Posterior Predictive Distribution

$$p(\tilde{x} \in C \mid D) = \sum_h p(y = 1 \mid \tilde{x}, h) p(h \mid D)$$

When we have a small and/or ambiguous data set, the posterior  $p(h \mid D)$  is vague, which induces a broad predictive distribution. However, once we have “figured things out”, the posterior becomes a delta function centered at the MAP estimate.

- In this case, the predictive distribution becomes:

$$p(\tilde{x} \in C \mid D) = \sum_h p(\tilde{x} \mid h) \delta_{\hat{h}}(h) = p(\tilde{x} \mid \hat{h})$$

- This is called a **plug-in approximation** to the predictive density and is very widely used, due to its simplicity. (NB: in practice the plug-in approximation can underrepresent our true degree of uncertainty).

# Bayesian Concept Learning

## Posterior Predictive Distribution

$$\underbrace{p(\tilde{x} \in C | D) = \sum_h p(y = 1 | \tilde{x}, h) p(h | D)}_{\text{posterior predictive distribution}}$$

$$\underbrace{p(\tilde{x} \in C | D) = \sum_h p(\tilde{x} | h) \delta_{\hat{h}}(h) = p(\tilde{x} | \hat{h})}_{\text{plug-in approximation}}$$

- Although MAP learning is simple (this is part of its intrinsic appeal), it nonetheless cannot explain the gradual shift from *similarity-based reasoning* (i.e. with uncertain posteriors) to *rule-based reasoning* (with certain posteriors).
- For instance, suppose we observe  $D = \{16\}$ ; if we use the simple prior (from the previous examples), the minimal consistent hypothesis is “all powers of 4”, so only 4 and 16 get non-zero probability of being predicted. This is an example of **overfitting**.

# Bayesian Concept Learning

## Posterior Predictive Distribution

$$\underbrace{p(\tilde{x} \in C | D) = \sum_h p(y=1 | \tilde{x}, h) p(h | D)}_{\text{posterior predictive distribution}} \quad \underbrace{p(\tilde{x} \in C | D) = \sum_h p(\tilde{x} | h) \delta_{\hat{h}}(h) = p(\tilde{x} | \hat{h})}_{\text{plug-in approximation}}$$

- For instance, suppose we observe  $D = \{16\}$ ; if we use the simple prior (from the previous examples), the minimal consistent hypothesis is “all powers of 4”, so only 4 and 16 get non-zero probability of being predicted. This is an example of **overfitting**.
- Given  $D = \{16, 8, 2, 64\}$ , the MAP hypothesis is “all powers of 2”. Thus the plug-in predictive distribution gets broader (or stays the same) as we see more data: it starts narrow, but is forced to broaden as it sees more data.
- In contrast, in the Bayesian approach, we start broad and then narrow down as we learn more. In particular, given  $D = \{16\}$ , there are many hypotheses with non-negligible posterior support, so the predictive distribution is broad. However, when we see  $D = \{16, 8, 2, 64\}$ , the posterior concentrates its mass on one hypothesis, so the predictive distribution becomes narrower.

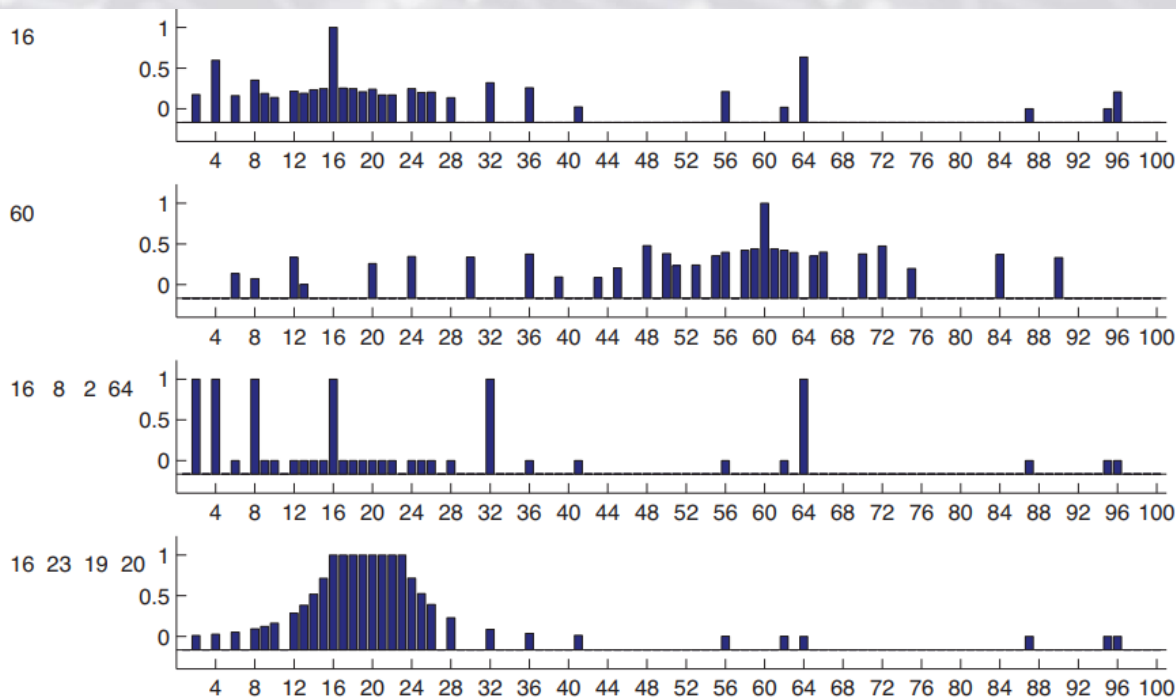
# Bayesian Concept Learning

## Posterior Predictive Distribution

$$\underbrace{p(\tilde{x} \in C | D) = \sum_h p(y = 1 | \tilde{x}, h) p(h | D)}_{\text{posterior predictive distribution}}$$

$$\underbrace{p(\tilde{x} \in C | D) = \sum_h p(\tilde{x} | h) \delta_{\hat{h}}(h) = p(\tilde{x} | \hat{h})}_{\text{plug-in approximation}}$$

- In summary: the predictions made by a plug-in approach and a Bayesian approach are quite different in the small data regime – although they converge to the same answer as we see more data.



**Figure 3.5** Predictive distributions for the model using the full hypothesis space. Compare to Figure 3.1. The predictions of the Bayesian model are only plotted for those values of  $\tilde{x}$  for which human data is available; this is why the top line looks sparser than Figure 3.4. Source: Figure 5.6 of (Tenenbaum 1999).



# The Beta-Binomial Model

## The Beta Distribution

- The *beta distribution* is one of the most commonly applied distributions in statistics; it is frequently used as a prior for a model parameter(s), as it has support over  $[0,1]$ . It is defined as follows:

$$Beta(x | a, b) = \frac{1}{B(a, b)} x^{a-1} (1-x)^{b-1}$$

- Where  $B(p, q)$  is the **beta function** (it simply serves to normalize the distribution), defined in terms of the *gamma function*:

$$B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

where  $a, b > 0$  are **shape parameters** (we demonstrate their effects on the next slide); the **gamma function** is defined:

$$\Gamma(x) = \int_0^{\infty} u^{x-1} e^{-u} du$$

The Beta function was first introduced by Pearson, one of founders of mathematical statistics.



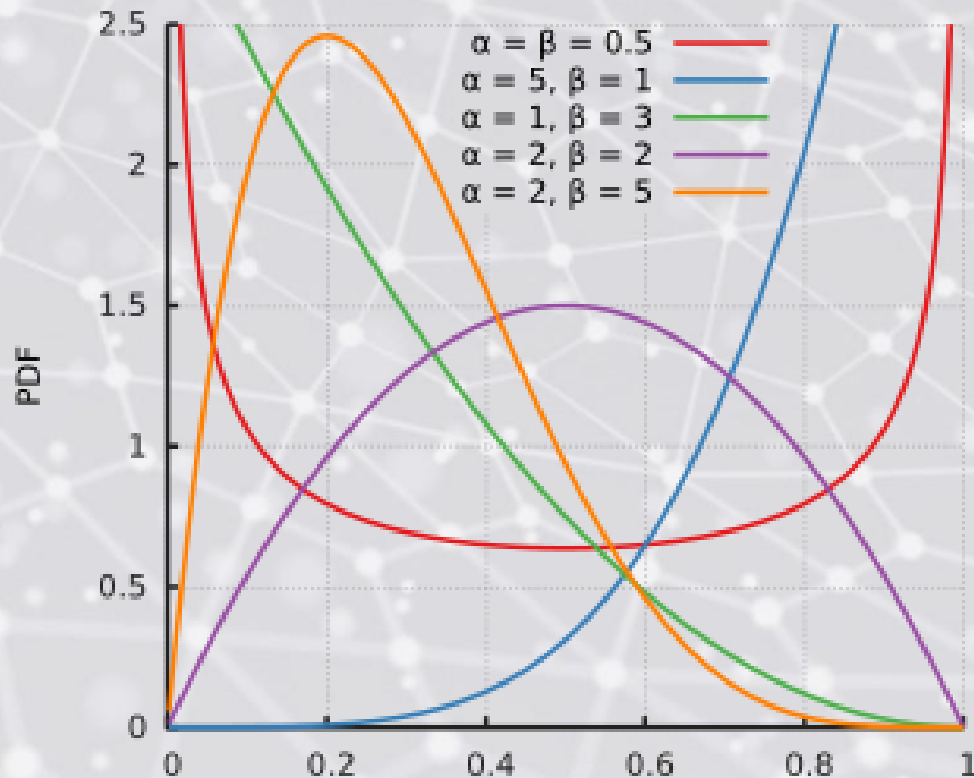
Pearson

# The Beta-Binomial Model

## The Beta Distribution

$$Beta(x | a, b) = \frac{1}{B(a, b)} x^{a-1} (1-x)^{b-1}$$

(\*) When  $a = b = 1$ , the beta distribution reduces to the **uniform distribution** (check this); if  $a$  and  $b$  are both less than 1, we get a **bimodal distribution** with “spikes” at 0 and 1; if  $a$  and  $b$  are both greater than 1, the distribution is **unimodal**.



$$\text{mean} : \frac{a}{a+b}$$

$$\text{mode} : \frac{a-1}{a+b-2}$$

$$\text{variance} : \frac{ab}{(a+b)^2 (a+b+1)}$$

# The Beta-Binomial Model

## The Bernoulli & Binomial Distributions

- The **Bernoulli distribution** is the probability distribution of a random variable which takes the value 1 with probability  $\theta$  and the value 0 with probability  $1 - \theta$ . That is, the probability distribution of any single experiment that asks a *yes-no question*; the question results in a Boolean-valued outcome/a single bit of information.

The Bernoulli *pmf* (probability mass function) is typically expressed as:

$$\text{Ber}(x | \theta) = \begin{cases} \theta & \text{if } x = 1 \\ 1 - \theta & \text{if } x = 0 \end{cases}$$

Using indicator functions, one can also express the Bernoulli pmf:

$$\text{Ber}(x | \theta) = \theta^{I(x=1)} (1 - \theta)^{I(x=0)}$$



J. Bernoulli

# The Beta-Binomial Model

## The Bernoulli & Binomial Distributions

- The Bernoulli *pmf* (probability mass function) is typically expressed as:

$$Ber(x | \theta) = \begin{cases} \theta & \text{if } x = 1 \\ 1 - \theta & \text{if } x = 0 \end{cases}$$

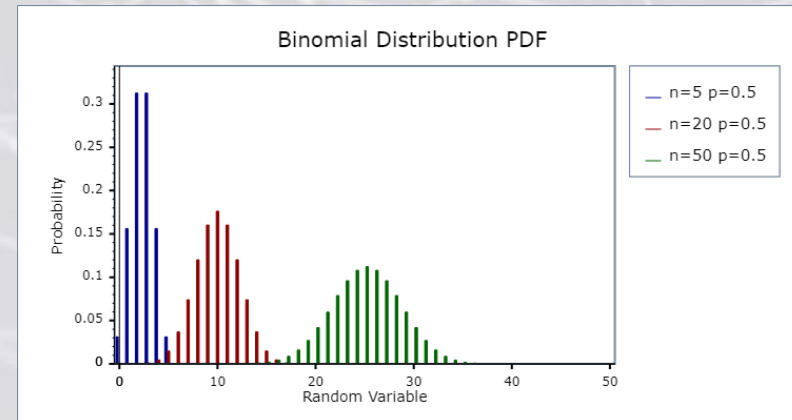
- Using indicator functions, one can also express the Bernoulli pmf:

$$Ber(x | \theta) = \theta^{I(x=1)} (1 - \theta)^{I(x=0)}$$

- In the case of  $n$  independent Bernoulli trials with shared parameters, we define the **binomial distribution** for the random variable  $X \in \{0, \dots, n\}$  as the total number of “successes” (i.e.  $x=1$  counts) in  $n$  trials. The binomial pmf is defined:

$$Bin(X = k | n, \theta) = \binom{n}{k} \theta^k (1 - \theta)^{n-k}$$

with summary statistics: mean:  $n\theta$ ; variance:  $n\theta(1 - \theta)$

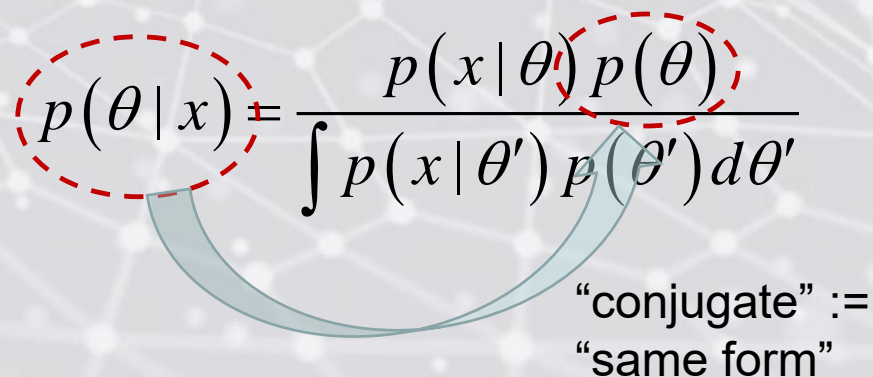




# The Beta-Binomial Model: Conjugacy

- In combination, the beta-binomial model is a very useful statistical model – particularly in Bayesian analysis. Beta-binomial models are common in many ML applications, including the naïve Bayes classifier, Markov models and many NLP settings. Its historical genesis dates back to Bayes' original paper (1763).

- The beta-binomial model is perhaps the most straightforward example of **conjugate distributions**. In Bayesian theory, if the *posterior distributions*  $p(\theta | x)$  are in the same family (e.g. such as the *exponential family* of distributions) as the prior distribution  $p(\theta)$ , we say the prior and posterior are then called conjugate distributions, and the prior is called a **conjugate prior** for the likelihood. More concretely: when the prior and the posterior have the same closed-form, we say the prior is a conjugate prior for the corresponding likelihood.



The diagram illustrates the concept of conjugacy using the equation for the posterior distribution  $p(\theta | x)$ . The equation is 
$$p(\theta | x) = \frac{p(x | \theta) p(\theta)}{\int p(x | \theta') p(\theta') d\theta'}$$
 A red dashed oval encircles the numerator  $p(x | \theta) p(\theta)$ , and another red dashed oval encircles the denominator  $\int p(x | \theta') p(\theta') d\theta'$ . A light blue curved arrow points from the  $p(\theta)$  term in the numerator to the  $p(\theta')$  term in the denominator, highlighting that both the prior and the posterior share the same functional form. Below the equation, the text "“conjugate” := “same form”" explains this relationship.

“conjugate” :=  
“same form”

# The Beta-Binomial Model: Conjugacy

- The beta-binomial model is perhaps the most straightforward example of **conjugate distributions**. In Bayesian theory, if the *posterior distributions*  $p(\theta | x)$  are in the same family (e.g. such as the *exponential family* of distributions) as the prior distribution  $p(\theta)$ , we say the prior and posterior are then called conjugate distributions, and the prior is called a **conjugate prior** for the likelihood. More concretely: when the prior and the posterior have the same closed-form, we say **the prior is a conjugate prior for the corresponding likelihood**.

- In addition to the beta-binomial model, the *Gaussian-Gaussian* and *Dirichlet-multinomial* models are also examples of conjugate distributions.

(\*) Conjugacy is an attractive quality in statistics because it means that the posterior distribution can be tractably calculated – which is to say we can represent it compactly in a recognizable closed-form.

# The Beta-Binomial Model

## Likelihood

• Suppose  $X_i \sim \text{Ber}(\theta)$  where  $X_i=1$  represents “heads”,  $X_i=0$  represents “tails”, and  $\theta$  is the rate parameter (i.e. probability of heads). If the data are **IID** (*independent and identically distributed*), the likelihood has the form:

$$p(D | \theta) = \theta^{N_1} (1 - \theta)^{N_0}$$

where we define  $N_1 = \sum_{i=1}^N I(x_i = 1)$  for the total “heads count” and  $N_0 = \sum_{i=1}^N I(x_i = 0)$  for the total “tails count.” These two counts are known as the **sufficient statistics** of the data, because this is all we need to know about  $D$  to infer  $\theta$  (note that there are alternative choices for sufficient statistics in this case:  $N_0$  and  $N = N_0 + N_1$  is another valid choice).



# The Beta-Binomial Model

## Prior

- We need a prior with support over  $[0, 1]$ ; furthermore, we wish for the prior to have the same form as the likelihood, namely:

$$p(\theta | D) \propto \theta^{\gamma_1} (1 - \theta)^{\gamma_2}$$

- Naturally, this is a beta distribution. In this case, the posterior computation is facile:

$$p(\theta | D) \propto p(D | \theta) p(\theta) = \theta^{N_1} (1 - \theta)^{N_0} \theta^{\gamma_1} (1 - \theta)^{\gamma_2} = \theta^{N_1 + \gamma_1} (1 - \theta)^{N_0 + \gamma_2}$$

(\*) Note that the posterior is of the same form as the beta prior, confirming that the beta prior is a conjugate prior for the binomial likelihood.



# The Beta-Binomial Model

## Posterior

- In summary, when we multiply the binomial likelihood by the beta prior we get a posterior of the following form:

$$p(\theta | D) \propto \text{Bin}(N_1 | \theta, N) \text{Beta}(\theta | a, b) \propto \text{Beta}(\theta | N_1 + a, N_0 + b)$$

- In particular, the posterior is obtained by adding the prior hyper-parameters to the empirical counts (i.e.  $N_1$  and  $N_0$ ). For this reason, the hyper-parameters are known as **pseudo counts**. The strength of the prior, also known as the **equivalent sample size of the prior**, is the sum of the pseudo counts:  $\alpha_0 = a + b$ ; this plays a role analogous to the data set size,  $N_1 + N_0 = N$ .

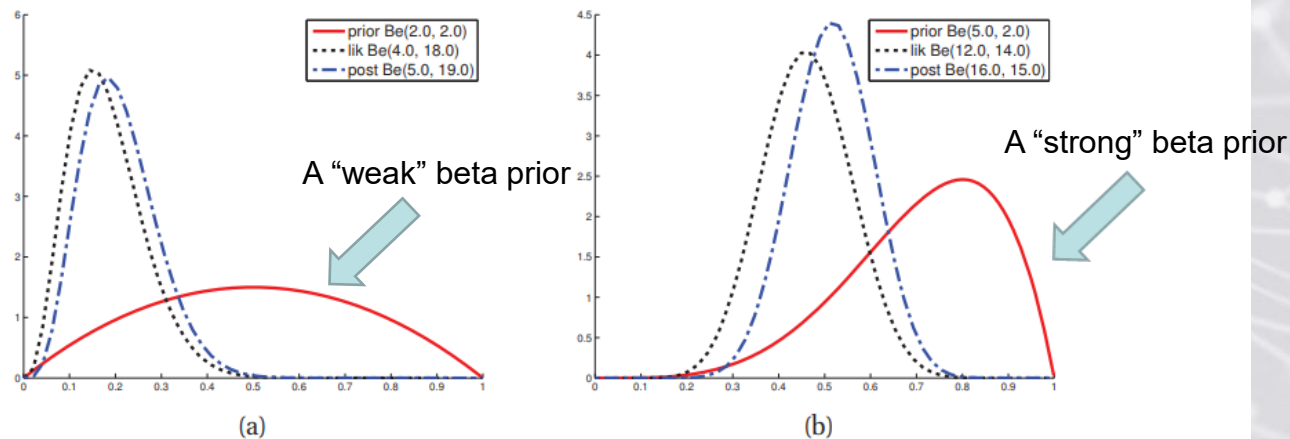
# The Beta-Binomial Model

## Posterior

- In summary, when we multiply the binomial likelihood by the beta prior we get a posterior of the following form:

$$p(\theta | D) \propto \text{Bin}(N_1 | \theta, N) \text{Beta}(\theta | a, b) \propto \text{Beta}(\theta | N_1 + a, N_0 + b)$$

- In particular, the posterior is obtained by adding the prior hyper-parameters to the empirical counts (i.e.  $N_1$  and  $N_0$ ). For this reason, the hyper-parameters are known as **pseudo counts**. The strength of the prior, also known as the **equivalent sample size of the prior**, is the sum of the pseudo counts:  $\alpha_0 = a + b$ ; this plays a role analogous to the data set size,  $N_1 + N_0 = N$ .



(\*) Note that the posterior in each case is a "compromise" between prior/likelihood

**Figure 3.6** (a) Updating a Beta(2, 2) prior with a Binomial likelihood with sufficient statistics  $N_1 = 3, N_0 = 17$  to yield a Beta(5,19) posterior. (b) Updating a Beta(5, 2) prior with a Binomial likelihood with sufficient statistics  $N_1 = 11, N_0 = 13$  to yield a Beta(16, 15) posterior. Figure generated by

# The Beta-Binomial Model

## Posterior

$$p(\theta | D) \propto \text{Bin}(N_1 | \theta, N) \text{Beta}(\theta | a, b) \propto \text{Beta}(\theta | N_1 + a, N_0 + b)$$

Beta distribution summary statistics:

$$\text{mean} : \frac{a}{a+b}$$

$$\text{mode} : \frac{a-1}{a+b-2}$$

$$\text{variance} : \frac{ab}{(a+b)^2(a+b+1)}$$

- The *MAP estimate* is given by:

$$\hat{\theta}_{MAP} = \frac{a + N_1 - 1}{a + b + N - 2}$$

- If we use a uniform prior (i.e.  $a = b = 1$ ), then the MAP estimate reduces to the MLE, which is just the empirical fraction of heads:

$$\hat{\theta}_{MLE} = \frac{N_1}{N}$$

- This makes intuitive sense. The *posterior mean* is given by:

$$\bar{\theta}_{MAP} = \frac{a + N_1}{a + b + N}$$

# The Beta-Binomial Model

## Posterior

$$p(\theta | D) \propto \text{Bin}(N_1 | \theta, N) \text{Beta}(\theta | a, b) \propto \text{Beta}(\theta | N_1 + a, N_0 + b)$$

- Importantly, it is possible to express the posterior mean for the beta-binomial model as a convex combination of the prior mean and the MLE estimate; let  $\alpha_0 = a + b$  and  $m = a/(a+b)$ , i.e., we set  $m$  equal to the prior mean. Then:

$$E[(\theta | D)] = \frac{a + N_1}{a + b + N} = \frac{\alpha_0}{N + \alpha_0} m + \frac{N}{N + \alpha_0} \frac{N_1}{N} = \lambda m + (1 - \lambda) \hat{\theta}_{MLE}$$

where  $\lambda = \alpha_0 / (N + \alpha_0)$ , is the ratio of the prior to posterior *equivalent sample size*. Thus, the weaker the prior, the smaller  $\lambda$  is, and hence the closer to the posterior mean is to the MLE as  $N \rightarrow \infty$ .



# The Beta-Binomial Model

$$p(\theta | D) \propto \text{Bin}(N_1 | \theta, N) \text{Beta}(\theta | a, b) \propto \text{Beta}(\theta | N_1 + a, N_0 + b)$$

- Recall that the Bayesian solution to the “problem of induction”/Black swan paradox (viz., what to do in the absence of an observation corresponding with a particular variable configuration – or how to draw general conclusions about the future from specific observations from the past) is to introduce a prior.
- Under the aegis of the *principle of insufficient reason* (PIR), we use a **uniform or non-informative prior** when there is no compelling evidence to do/presume otherwise.
- The uniform beta corresponds with  $a = b = 1$ ; which yields a posterior:  $\text{Beta}(\theta | N_1 + 1, N_0 + 1)$ .  
From the previous slide, this yields a posterior mean:

$$\bar{\theta}_{MAP} = \frac{N_1 + 1}{N + 2}$$

Which is precisely the common expression used in **Laplace/add-one smoothing** (this formula easily generalizes for multi-class settings).

# The Dirichlet-Multinomial Model

- The **Dirichlet-multinomial** model is another useful model that is common in ML applications – particularly in NLP and biosequence data. It is likewise a conjugate model (as we show subsequently) which renders a closed-form posterior.

(\*) **Simply put**: the Dirichlet-multinomial model is a natural generalization of the beta-binomial model to a multi-class setting; respectively, the multinomial distribution is simply a generalization of the binomial distribution to a multi-class setting, and, similarly, the Dirichlet distribution generalizes the beta distribution.

# The Dirichlet-Multinomial Model

## The Multinomial Distribution

While the binomial distribution can be used to model the outcomes of coin tosses, by extension, the multinomial distribution generalizes to model the outcomes of tossing a  $K$ -sided die. Let  $\mathbf{x}=(x_1,\dots,x_K)$  be random vector, where  $x_j$  is the number of times side  $j$  of the die occurs. Then  $\mathbf{x}$  has the following pmf:

$$Mu(\mathbf{x} | n, \theta) = \left( \frac{n!}{x_1! \cdots x_K!} \right) \prod_{j=1}^K \theta_j^{x_j}$$

Where  $\theta_j$  is the probability that side  $j$  shows up; the left-most factor in this expression represents the number of ways to divide a set of size  $n = x_1 + \dots + x_K$  into subsets with sizes  $x_1$  up to  $x_K$ .

# The Dirichlet-Multinomial Model

## The Multinomial Distribution

Let  $\mathbf{x}=(x_1,\dots,x_K)$  be random vector, where  $x_j$  is the number of times side  $j$  of the die occurs. Then  $\mathbf{x}$  has the following pmf:

$$Mu(\mathbf{x} | n, \theta) = \left( \frac{n!}{x_1! \cdots x_K!} \right) \prod_{j=1}^K \theta_j^{x_j}$$

**Example 1:** An experiment of drawing a random card from an ordinary playing cards deck is done with replacing it back. This was done ten times. Find the probability of getting 2 spades, 3 diamond, 3 club and 2 hearts.

**Solution:**

We use the multinomial probability here to solve this problem.

In this case, the number of trials,  $n = 10$

Now since four types of outputs are there means  $k = 4$  with  $n_1 = 2, n_2 = 3, n_3 = 3, n_4 = 2$

Now the probability of drawing a spade, diamond, club or a heart is  $\frac{13}{52} = 0.25$  for each one of them. Thus we get that  $p_1 = 0.25, p_2 = 0.25 = p_3 = p_4$ .

Now we substitute the values in the multinomial probability formula and we get,

$$P = \left[ \frac{n!}{(n_1! \cdot n_2! \cdot n_3! \cdot n_4!)} \right] \cdot p_1^{n_1} \cdot p_2^{n_2} \cdot p_3^{n_3} \cdot p_4^{n_4}$$

$$\rightarrow P = \left[ \frac{10!}{(2! \cdot 3! \cdot 3! \cdot 2!)} \right] \cdot (0.25)^2 \cdot (0.25)^3 \cdot (0.25)^3 \cdot (0.25)^2$$

$$\rightarrow P = 0.024$$



# The Dirichlet-Multinomial Model

## The Dirichlet Distribution

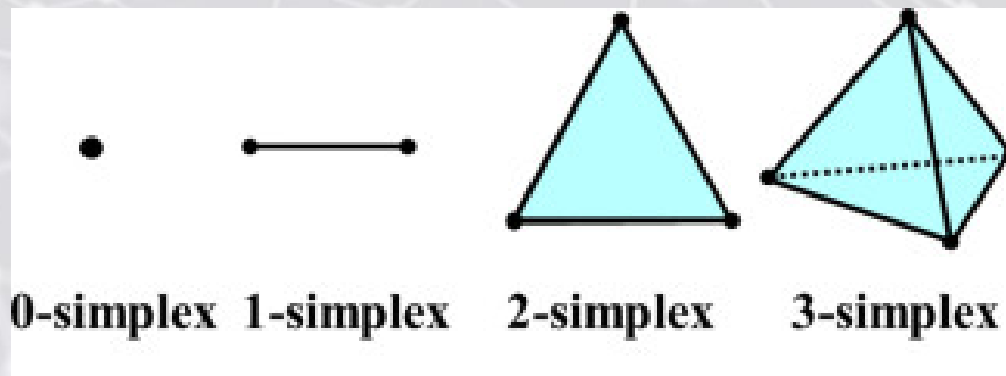
The Dirichlet distribution is a multi-variate generalization of the beta distribution, which has support over the **probability simplex** (plural: simplices) defined by:

$$S_K = \left\{ x : 0 \leq x_k \leq 1, \sum_{k=1}^K x_k = 1 \right\}$$

A *simplex* is a generalization of the notion of a triangle or tetrahedron to arbitrary dimensions. Formally, a  $K$ -simplex is a  **$K$ -dimensional polytope** which is the **convex hull** of its  $K + 1$  vertices. A 2-simplex is a triangle; a 3-simplex is a tetrahedron, etc.



Dirichlet



# The Dirichlet-Multinomial Model

## The Dirichlet Distribution

$$S_K = \left\{ \mathbf{x} : 0 \leq x_k \leq 1, \sum_{k=1}^K x_k = 1 \right\}$$

- The *pdf* of the Dirichlet distribution is defined as follows:

$$Dir(\mathbf{x} | \boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^K x_k^{\alpha_k - 1} I(\mathbf{x} \in S_K)$$

Where  $B(\alpha_1, \dots, \alpha_K)$  is the natural generalization of the **beta function** to  $K$  variables:

$$B(\boldsymbol{\alpha}) = \frac{\prod_{k=1}^K \Gamma(\alpha_k)}{\Gamma(\alpha_0)}, \text{ for which the } \textit{gamma function} \text{ is defined: } \Gamma(x) = \int_0^{\infty} u^{x-1} e^{-u} du$$

where  $\alpha_0 = \sum_{k=1}^K \alpha_k$ ;  $I(\mathbf{x} \in S_K)$  denotes the **indicator function for the  $K$ -simplex**; this just means that the support of the Dirichlet distribution corresponds to the  $K$ -simplex.

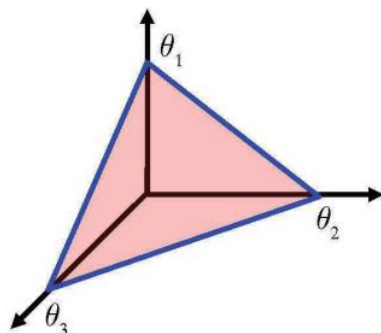
$$E[x_k] = \frac{\alpha_k}{\alpha_0}, \text{ mode}[x_k] = \frac{\alpha_k - 1}{\alpha_0 - K}, \text{ var}[x_k] = \frac{\alpha_k (\alpha_0 - \alpha_k)}{\alpha_0^2 (\alpha_0 + 1)}$$

# The Dirichlet-Multinomial Model

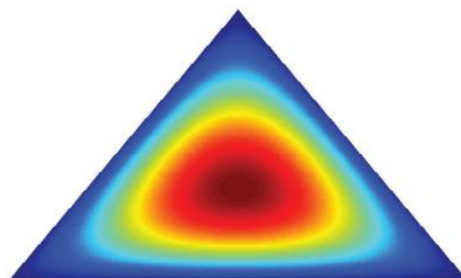
## The Dirichlet Distribution

$$\text{Dir}(\mathbf{x} | \boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^K x_k^{\alpha_k - 1} I(\mathbf{x} \in S_K)$$

$$S_K = \left\{ \mathbf{x} : 0 \leq x_k \leq 1, \sum_{k=1}^K x_k = 1 \right\}$$



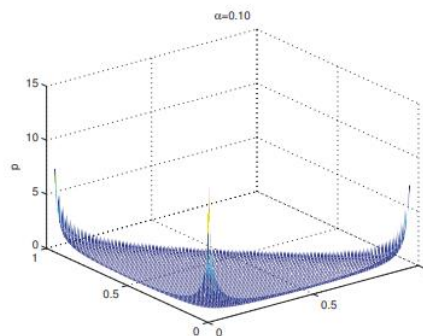
(a)



(b)



(c)



(d)

The figure shows some plots of the Dirichlet when  $K=3$ . We see that  $\alpha_0 = \sum_{k=1}^K \alpha_k$  controls the strength of the distribution (i.e. how peaked it is), and the  $\alpha_k$  control where the peaks occur.

For example,  $\text{Dir}(1,1,1,)$  is a **uniform distribution**;  $\text{Dir}(2,2,2)$  is a broad distribution centered at  $(1/3, 1/3, 1/3)$ , and  $\text{Dir}(20,20,20)$  is a narrow distribution with the same center. If  $\alpha_k < 1$  for all  $k$ , we get “spikes” at the corners of the simplex.

**Figure 2.14** (a) The Dirichlet distribution when  $K = 3$  defines a distribution over the simplex, which can be represented by the triangular surface. Points on this surface satisfy  $0 \leq \theta_k \leq 1$  and  $\sum_{k=1}^3 \theta_k = 1$ . (b) Plot of the Dirichlet density when  $\boldsymbol{\alpha} = (2, 2, 2)$ . (c)  $\boldsymbol{\alpha} = (20, 2, 2)$ . Figure generated by visDirichletGui, by Jonathan Huang. (d)  $\boldsymbol{\alpha} = (0.1, 0.1, 0.1)$ . (The comb-like structure on the edges is a plotting artifact.) Figure generated by dirichlet3dPlot.

# The Dirichlet-Multinomial Model

## Likelihood

- Suppose we observe  $N$  dice rolls,  $D = \{x_1, \dots, x_N\}$ , where  $x_i \in \{1, \dots, K\}$ . If we assume the data are IID, the likelihood has the form:

$$p(X | \theta) = \prod_{k=1}^K \theta_k^{N_k}$$

where  $N_k = \sum_{i=1}^N I(y_i=k)$  is the number of times even  $k$  occurred.



# The Dirichlet-Multinomial Model

## Likelihood

- Suppose we observe  $N$  dice rolls,  $D = \{x_1, \dots, x_N\}$ , where  $x_i \in \{1, \dots, K\}$ . If we assume the data are IID, the likelihood has the form:

$$p(D | \theta) = \prod_{k=1}^K \theta_k^{N_k}$$

where  $N_k = \sum_{i=1}^N I(y_i=k)$  is the number of times even  $k$  occurred.

## Prior

- Since the parameter vector lives in the  $K$ -dimensional probability simplex, we need a prior that has support over this simplex. Ideally it would also be **conjugate**. Happily, the Dirichlet distribution satisfies both criteria. Hence we use the following prior:

$$Dir(\theta | \alpha) = \frac{1}{B(\alpha)} \prod_{k=1}^K x_k^{\alpha_k - 1} I(\mathbf{x} \in S_K)$$

# The Dirichlet-Multinomial Model

## Likelihood

$$p(D | \boldsymbol{\theta}) = \prod_{k=1}^K \theta_k^{N_k}$$

## Prior

$$Dir(\boldsymbol{\theta} | \boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^K x_k^{\alpha_k - 1} I(\mathbf{x} \in S_K)$$

## Posterior

$$\begin{aligned} p(\boldsymbol{\theta} | D) &\propto p(D | \boldsymbol{\theta}) p(\boldsymbol{\theta}) \propto \prod_{k=1}^K \theta_k^{N_k} \theta_k^{\alpha_k - 1} = \prod_{k=1}^K \theta_k^{\alpha_k + N_k - 1} \\ &= Dir(\boldsymbol{\theta} | \alpha_1 + N_1, \dots, \alpha_K + N_K) \end{aligned}$$

(\*) We see that the prior and posterior have the same form (i.e. they are both Dirichlet distributions) – namely, the prior is a conjugate prior for the corresponding multinomial likelihood. Note that the posterior is obtained simply by adding the prior hyper-parameters (pseudo-counts)  $\alpha_k$ 's to the empirical counts,  $N_k$ 's.

# The Dirichlet-Multinomial Model

## Posterior

$$\begin{aligned} p(\boldsymbol{\theta} | D) &\propto p(D | \boldsymbol{\theta}) p(\boldsymbol{\theta}) \propto \prod_{k=1}^K \theta_k^{N_k} \theta_k^{\alpha_k - 1} = \prod_{k=1}^K \theta_k^{\alpha_k + N_k - 1} \\ &= \text{Dir}(\boldsymbol{\theta} | \alpha_1 + N_1, \dots, \alpha_K + N_K) \end{aligned}$$

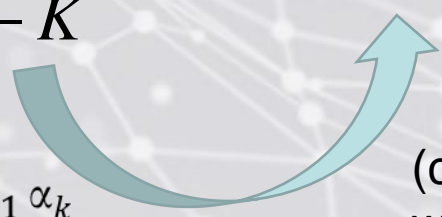
Using the method of **Lagrange Multipliers** (we'll walk through this method in more detail in future lectures to brush up), one can solve for the posterior mode, i.e. MAP for the Dirichlet-multinomial model:

$$\hat{\theta}_k = \frac{N_k + \alpha_k - 1}{N + \alpha_0 - K}$$

$\alpha_0 = \sum_{k=1}^K \alpha_k$

$\text{mode}[x_k] = \frac{\alpha_k - 1}{\alpha_0 - K}$

(consistent with previous formula)



# The Dirichlet-Multinomial Model

## Posterior

$$\begin{aligned} p(\boldsymbol{\theta} | D) &\propto p(D | \boldsymbol{\theta}) p(\boldsymbol{\theta}) \propto \prod_{k=1}^K \theta_k^{N_k} \theta_k^{\alpha_k - 1} = \prod_{k=1}^K \theta_k^{\alpha_k + N_k - 1} \\ &= \text{Dir}(\boldsymbol{\theta} | \alpha_1 + N_1, \dots, \alpha_K + N_K) \end{aligned}$$

Using the method of **Lagrange Multipliers** (we'll walk through this method in more detail in future lectures to brush up), one can solve for the posterior mode, i.e. MAP for the Dirichlet-multinomial model:

$$\hat{\theta}_k = \frac{N_k + \alpha_k - 1}{N + \alpha_0 - K}$$

Observe that if we use a uniform prior,  $\boldsymbol{\alpha}=\mathbf{1}$  (i.e. the alpha vector consists of all 1's), then the MAP reduces to the MLE (once again), whereby:

$$\hat{\theta}_k = \frac{N_k}{N}$$

This is just the empirical fraction of times face  $k$  shows up.



# The Dirichlet-Multinomial Model: Example

- One application of Bayesian analysis using the Dirichlet-multinomial model is to language modeling, which means predicting which words might occur next in a sequence.
- Here we will use a standard **bag of words** model and assume that the  $i$ th word  $X_i \in \{1, \dots, K\}$  is sampled independently from all other words.
- Given a past sequence of words, how can we predict which one is likely to come next?

# The Dirichlet-Multinomial Model:

## Example

For example, suppose we observe the following sequence (part of a children's nursery rhyme):

Mary had a little lamb, little lamb, little lamb,  
Mary had a little lamb, its fleece as white as snow

Furthermore, suppose our vocabulary consists of the following words:

|      |      |        |     |        |       |       |      |      |     |
|------|------|--------|-----|--------|-------|-------|------|------|-----|
| mary | lamb | little | big | fleece | white | black | snow | rain | unk |
| 1    | 2    | 3      | 4   | 5      | 6     | 7     | 8    | 9    | 10  |

Here **unk** stands for unknown, and represents all other words that do not appear elsewhere on the list. To encode each line of the nursery rhyme, we first strip off punctuation, and remove any **stop words** such as “a”, “as”, “the”, etc. We can also perform **stemming**, which means reducing words to their base form, such as stripping off the final *s* in plural words, or the *ing* from verbs (e.g., *running* becomes *run*). In this example, no words need stemming. Finally, we replace each word by its index into the vocabulary to get:

|   |    |   |   |    |   |    |   |   |
|---|----|---|---|----|---|----|---|---|
| 1 | 10 | 3 | 2 | 3  | 2 | 3  | 2 |   |
| 1 | 10 | 3 | 2 | 10 | 5 | 10 | 6 | 8 |

We now ignore the word order, and count how often each word occurred, resulting in a histogram of word counts:

# The Dirichlet-Multinomial Model: Example

|       |      |      |        |     |        |       |       |      |      |     |
|-------|------|------|--------|-----|--------|-------|-------|------|------|-----|
| Token | 1    | 2    | 3      | 4   | 5      | 6     | 7     | 8    | 9    | 10  |
| Word  | mary | lamb | little | big | fleece | white | black | snow | rain | unk |
| Count | 2    | 4    | 4      | 0   | 1      | 1     | 0     | 1    | 0    | 4   |

Denote the above counts by  $N_j$ . If we use a  $\text{Dir}(\alpha)$  prior for  $\theta$ , the posterior predictive is just

$$p(\tilde{X} = j|D) = E[\theta_j|D] = \frac{\alpha_j + N_j}{\sum_{j'} \alpha_{j'} + N_{j'}} = \frac{1 + N_j}{10 + 17} \quad (3.52)$$

If we set  $\alpha_j = 1$ , we get

$$p(\tilde{X} = j|D) = (3/27, 5/27, 5/27, 1/27, 2/27, 2/27, 1/27, 2/27, 1/27, 5/27) \quad (3.53)$$

The modes of the predictive distribution are  $X = 2$  (“lamb”) and  $X = 10$  (“unk”). Note that the words “big”, “black” and “rain” are predicted to occur with non-zero probability in the future, even though they have never been seen before. Later on we will see more sophisticated language models.

$$E[x_k] = \frac{\alpha_k}{\alpha_0}, \quad \text{mode}[x_k] = \frac{\alpha_k - 1}{\alpha_0 - K} \quad \text{Dir}(\boldsymbol{\theta} \mid \alpha_1 + N_1, \dots, \alpha_K + N_K)$$

(\*) Notice that the posterior predictive is equivalent to “Laplace smoothing” in a multi-class setting, i.e. it is the posterior mean.



# The Dirichlet-Multinomial Model: Example

|       |      |      |        |     |        |       |       |      |      |     |
|-------|------|------|--------|-----|--------|-------|-------|------|------|-----|
| Token | 1    | 2    | 3      | 4   | 5      | 6     | 7     | 8    | 9    | 10  |
| Word  | mary | lamb | little | big | fleece | white | black | snow | rain | unk |
| Count | 2    | 4    | 4      | 0   | 1      | 1     | 0     | 1    | 0    | 4   |

Denote the above counts by  $N_j$ . If we use a  $\text{Dir}(\alpha)$  prior for  $\theta$ , the posterior predictive is just

$$p(\tilde{X} = j|D) = E[\theta_j|D] = \frac{\alpha_j + N_j}{\sum_{j'} \alpha_{j'} + N_{j'}} = \frac{1 + N_j}{10 + 17} \quad (3.52)$$

If we set  $\alpha_j = 1$ , we get

$$p(\tilde{X} = j|D) = (3/27, 5/27, 5/27, 1/27, 2/27, 2/27, 1/27, 2/27, 1/27, 5/27) \quad (3.53)$$

The modes of the predictive distribution are  $X = 2$  (“lamb”) and  $X = 10$  (“unk”). Note that the words “big”, “black” and “rain” are predicted to occur with non-zero probability in the future, even though they have never been seen before. Later on we will see more sophisticated language models.

$$E[x_k] = \frac{\alpha_k}{\alpha_0}, \quad \text{mode}[x_k] = \frac{\alpha_k - 1}{\alpha_0 - K} \quad \text{Dir}(\theta | \alpha_1 + N_1, \dots, \alpha_K + N_K)$$

(\*) Notice that the posterior predictive is equivalent to “Laplace smoothing” in a multi-class setting, i.e. it is the posterior mean.



# The Dirichlet-Multinomial Model: Research Example

Elkan, C., “Clustering Documents with an Exponential-Family Approximation of the Dirichlet Compound Multinomial Distribution”, ICML, 2006.

The authors apply the Dirichlet compound multinomial (DCM) as model for text documents that takes into account “burstiness” (i.e. the fact that if a word occurs more than once in a document, it is likely to occur repeatedly). They derive a new family of distributions that are approximations to DCM (called EDCM distributions). Their new algorithm is competitive with state-of-the-art (at the time).

Clustering documents with the EDCM

Table 1. Clustering results on two document collections ( $D$ : number of documents,  $\bar{n}_d$ : average document length,  $W$ : vocabulary size,  $k$ : number of classes,  $p$ : perplexity  $\pm$  standard error, MI: mutual information  $\pm$  standard error, time measured in seconds).

| name    | $D$ | $\bar{n}_d$ | $W$  | $k$ | DCM               |                       |       | EDCM              |                       |      |
|---------|-----|-------------|------|-----|-------------------|-----------------------|-------|-------------------|-----------------------|------|
|         |     |             |      |     | $p$               | MI                    | time  | $p$               | MI                    | time |
| Classic | 400 | 78.8        | 6205 | 3   | $853.96 \pm 1.91$ | $0.77197 \pm 0.01063$ | 49.7  | $877.64 \pm 1.87$ | $0.77203 \pm 0.01136$ | 2.35 |
| NIPS    | 391 | 1332.4      | 6871 | 9   | $804.19 \pm 0.24$ | $0.84364 \pm 0.00659$ | 751.3 | $875.17 \pm 0.15$ | $0.83705 \pm 0.00737$ | 6.61 |

Perplexity is a common metric in NLP applications that computes the average uncertainty the model assigns to each word in a document collection; low perplexity is preferred.

<http://cseweb.ucsd.edu/~elkan/edcm.pdf>

*Fin*

