Previously…

BN Structure

- BN structure learning is useful for building better predictive models
  - ‣ when domain experts don’t know the structure well, and
  - ‣ for Knowledge discovery
- Finding highest-scoring structures is NP-hard for structures beyond trees
  - ➡ Unlikely to find efficient algorithms
  - ➡ We can resort to simple heuristic search
    - Local steps: edge addition, deletion and reversal
    - Augmented hill climbing algorithms to avoid local maxima
  - ➡ There are better algorithms
    - Make larger progress on the search space
    - Computationally more expensive and harder to implement

Assessing how well a model explains the data

- Likelihood score computes log-likelihood of D relative to G, using MLE parameters for G
  - Parameters optimised for D

+ Nice information-theoretic interpretation in terms of (in)dependencies in G

– Guaranteed to overfill the training data if we do not impose constraints.

Scores with penalisation

- BIC score explicitly penalises model complexity
- BIC is asymptotically consistent:
  - ✤ If data is generated by a true graph G*, Markov equivalent networks to G* will have the highest BIC score as M (the number of data instances) grows to infinity

Bayesian Score

- the Bayesian approach : whenever we have uncertainty over anything, we should place a distribution over it.
- Uncertainty over the structure and the parameters
- Define priors
  - $P(G)$: a prior probability on different graph structures
  - $P(\theta_G \mid G)$: parameter prior probability on different choice of parameters once the graph G is given
- Using Baye's Rule

$$P(G \mid D) = \frac{P(D \mid G)P(G)}{P(D)}$$

Marginal Likelihood

$P(D \mid G)$

The posterior over parameters provides us with a **range of choices**, along with a **measure of how likely each of them is**.

Measures the expected likelihood, averaged over **different possible choices of parameters θ_G**.

- We are being more conservative in our estimate of the "goodness" of the model.

  ➡ thus, avoiding overfitting

Summary

- Score-based methods consider the whole structure at once.
- They are therefore less sensitive to individual failures and better at making compromises between independency of variables in the data and the “cost” of adding the edge.
- However, they pose a search problem that may not have an elegant and efficient solution.

Model Selection

Selecting a Hypothesis/Model

How can we **compare the performance** of the models in fitting a set of data $D = \{x_1, \ldots, x_N\}$?

Goal: Learn a hypothesis/model that fits "new" data "best"

"new" data:

- There is a (true) **stationary** probability distribution over the samples. **[stationary assumption]**
- Each data point is a random variable whose **observed value** is sampled from that distribution, and is **independent** of the previous examples, and each example has an **identical** prior probability distribution. **[i.i.d. assumption]**

Evaluating and choosing the best Hypothesis

How do we define "best fit"?

The **error rate** of a hypothesis is the proportion of mistakes it makes: that is the proportion of times that $h(x) \neq y$ for an (x, y)example.

Note: a low error rate does not necessarily means the hypothesis will perform well on new data.

✤ To get an accurate evaluation of a hypothesis, we need to test it on unseen examples

Holdout cross-validation: randomly split the available data into a **training** and **test** datasets.

K-fold cross-validation:

- Split the data into k equal subsets.
- Perform k rounds of learning, such that on each round 1/k of the data is used as a test dataset.

We can think of the task of finding the best hypothesis as two tasks:

- **model selection**: defines the hypothesis space
- **optimisation**: finds the best hypothesis within that space.

From Error Rates to Loss

- Minimising the the error rate seems reasonable when finding the best model. However,…

Consider the problem of **classifying** email messages as spam or non-spam, which of the following scenarios is worse?

- ✦ non-spam message classified as spam
- ✦ spam message classified as non-spam

Loss Functions

- Learners should maximise **utility**
- In ML utilities are expressed by means of a **loss function.**
  - ✤ That is, the amount of utility lost by predicting $h(x) = y'$ when the correct answer is $f(x) = y$

$L(x, y, y') =$ ***Utility***(result of using y given an input x) –
Utility(result of using y' given an input x)

Often a *simplified version* that is independent of x is used:

$$L(y, y')$$

For example: We can say that it is 10 times worse to classify non-spam as spam that the other way around by setting

L(spam, no-spam)= 1, L(nospam, spam)= 10

Loss Functions (example)

For functions with discrete outputs, we can enumerate a loss value for each possible misclassification.

For example:

A surgeon is trying to decide whether to operate a patient or not. Imagine that there are 3 states:

the patient has no cancer
the patient has lung cancer
the patient has breast cancer

We can represent the loss function *L*(x,y') a a loss matrix such as the following

	Surgery	No surgery
No cancer	20	0
Lung Cancer	10	50
Breast Cancer	10	60

Common Loss Functions

- In general, small error are better than larger ones

Absolute value loss: $L_1(y, y') = |y - y'|$

Square error loss: $L_2(y, y') = (y - y')^2$

- If we one to minimise error loss

0/1 loss: $L_{0/1}(y, y') = \begin{cases} 0 & \text{if } y = y' \\ 1 & \text{otherwise} \end{cases}$

- Theoretically, the learning agent can **maximise** the **expected utility** by choosing the hypothesis that **minimises** expected loss over **all input–output** pairs seen.
- Expectation can be captured by defining a **prior probability distribution** $p(X, Y)$ **over examples.**

The expected **generalisation loss** for a hypothesis h with respect to a loss function *L* is

$$\text{GenLoss}_L(h) = \sum_{(x,y)\in\mathbf{E}} L(x, h(x))p(x, y)$$

where $\mathbf{E}$ *is the set of all possible input-output examples*

and the **best hypothesis** is the one with the **minimum** expected generalisation loss:

$$h* = \arg\min_{h\in\mathbf{H}} \text{GenLoss}_L(h)$$

Note: $p(X, Y)$ is not known

But the learning agent can ***estimate*** generalisation loss via the **empirical loss** on a set of examples $\mathbf{E}$

$$\text{EmpLoss}_{\mathbf{L},\mathbf{E}}(h) = \frac{1}{N}\sum_{(x,y)\in\mathbf{E}} L(y, h(x))$$

The **estimated best hypothesis** is the one with minimum empirical loss:

$$h* = \arg\min_{h\in\mathbf{H}} \text{EmpLoss}_L(h)$$

Note: The estimated best hypothesis may differ form the **true function f**

Why?

- f may not be in the hypothesis space **[unrealizability]**
- a learning algorithm will return different hypothesis for different set of examples, and those hypothesis will make different predictions for new examples **[variance]**
- f may be **nondeterministic** or **noisy.** That is, it may return different values for x.
- If the hypothesis space is complex, it can be computationally intractable to search on it

Bayesian Model Selection

- If we have leaned a set of models $M_1, \ldots, M_m$ of possibly different complexity, how should we chose the best one?

  ➡ We can use cross-validation to estimate the empirical loss of all candidate models and pick the model that seem the best

  ‣ This requires fitting each model K times if we use K-fold cross validation

✤ A more efficient approach is to compute the posterior over models

$$p(M \mid D)$$

Bayesian Approach for comparing Models

We have uncertainty about the appropriateness of **models** $M_1, \ldots, M_m$

✤ Define a **prior probability** over each model M_i (the appropriateness of M_i)

$$p(M_i)$$

Our interest is the model **posterior probability** (given the data D)

Applying Bayes' rule to models

$$p(M_i \mid D) = \frac{p(D \mid M_i) p(M_i)}{p(D)}$$

Model M_i is parametrised by θ_i.

$$p(D \mid M_i) = \int p(D \mid \theta_i, M_i) p(\theta_i \mid M_i)\, d\theta_i$$

The **likelihood of the data** D

$$p(D) = \sum_{i=1}^{m} p(D \mid M_i) p(M_i)$$

Bayesian Approach for comparing Models

We have uncertainty about the appropriateness of **models** $M_1, \ldots, M_m$

✤ Define a **prior probability** over each model M_i (the appropriateness of M_i)

$$p(M_i)$$

Our interest is the model **posterior probability** (given the data D)

Applying Bayes' rule to models

$$p(M_i \mid D) = \frac{p(D \mid M_i)p(M_i)}{p(D)}$$

Model M_i is parametrised by θ_i. In **discrete parameter spaces**, the model marginal likelihood is given by

$$p(D \mid M_i) = \sum_{\theta_i} p(D \mid \theta_i, M_i)p(\theta_i \mid M_i)$$

The **likelihood of the data** D

$$p(D) = \sum_{i=1}^{m} p(D \mid M_i)p(M_i)$$

Bayesian Approach for comparing Models

We have uncertainty about the appropriateness of **models** $M_1, \ldots, M_m$

✤ Define a **prior probability** over each model M_i (the appropriateness of M_i)

$$p(M_i)$$

Our interest is the model **posterior probability** (given the data D)

$$p(M_i \mid D) = \frac{p(D \mid M_i) p(M_i)}{p(D)}$$

Notes:

✦ The number of parameters is not necessarily the same for each model

✦ $p(M_i \mid D)$ only refers to the probability relative to the set of models $M_1, \ldots, M_m$.

This is not the absolute probability that model M_i fits "well"

Comparing two models

- But, comparing two model hypotheses M_i and M_j only requires the **Bayes factor:**

$$\frac{p(M_i \mid D)}{p(M_j \mid D)} = \underbrace{\frac{p(D \mid M_i)}{p(D \mid M_j)}}_{\textbf{Bayes' Factor}} \frac{p(M_i)}{p(M_j)}$$

Which does not require summation over **all possible models**

Example: testing whether a coin is biased or not

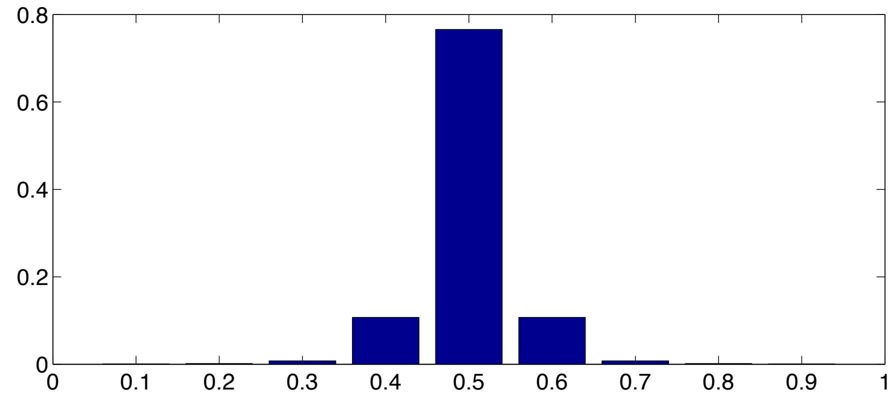
Let's assume θ is the probability that the coin will land heads. For a truly fair coin $\theta = 0.5$.

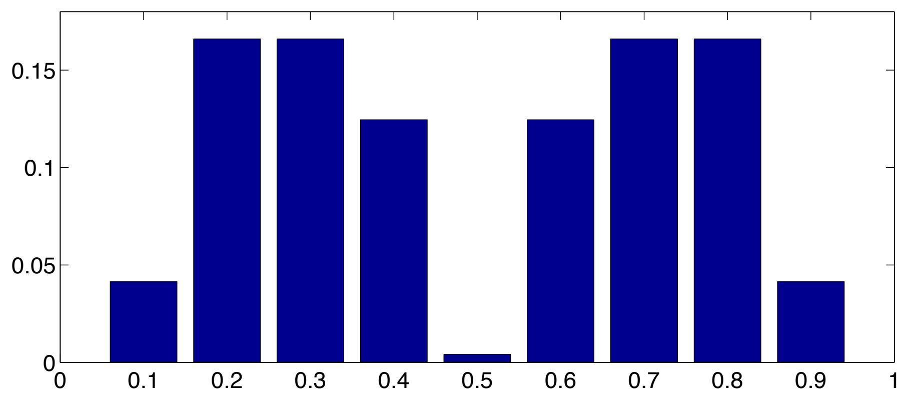
- Two models:

M_{fair}: the coin is fair

M_{biased}: the coin is biased

For simplicity we assume $dom(\theta) = \{0.1, 0.2, 0.3, \ldots, 0.9\}$





For each model M, the **likelihood** is given by

$$p(D \mid M) = \sum_{\theta} p(D \mid \theta, M) p(\theta \mid M) = \sum_{\theta} \theta^{N_H} (1-\theta)^{N_T} p(\theta \mid M)$$

This gives $0.1^{N_H} (1-0.1)^{N_T} p(\theta = 0.1 \mid M) + \ldots + 0.9^{N_H} (1-0.9)^{N_T} p(\theta = 0.9 \mid M)$

Exercise

- Assuming that $p(M_f | D) = p(M_b | D)$
- If we see 5 heads and 2 tails, what can we say about choosing between the two models?
- What if we saw 50 heads and 20 tails?

Complexity Penalisation

An unknown number of dice are rolled. Someone tells you that the sum of their scores t is 9. What is the posterior distribution of the dice scores ?

Assuming a priori that **any number n of dice is equally likely**, what is the **posterior distribution over n**?

From Bayes' rule, we need to compute the posterior distribution over models

$$p(n \mid t) = \frac{p(t \mid n) p(n)}{p(t)}$$

$$p(n \mid t) = \frac{p(t \mid n)p(n)}{p(t)}$$

Likelihood

$$p(t \mid n) = \sum_{s_1,\ldots,s_n} p(t, s_1, \ldots, s_n \mid n) = \sum_{s_1,\ldots,s_n} p(t \mid s_1, \ldots, s_n) \prod_i p(s_i)$$

$$= \sum \mathbb{1}\left[t = \sum_i^n s_i\right] \prod_i p(s_i)$$

where $p(s_i) = 1/6$ for all scores s_i

$$p(n \mid t) = \frac{p(t \mid n)p(n)}{p(t)}$$

Likelihood

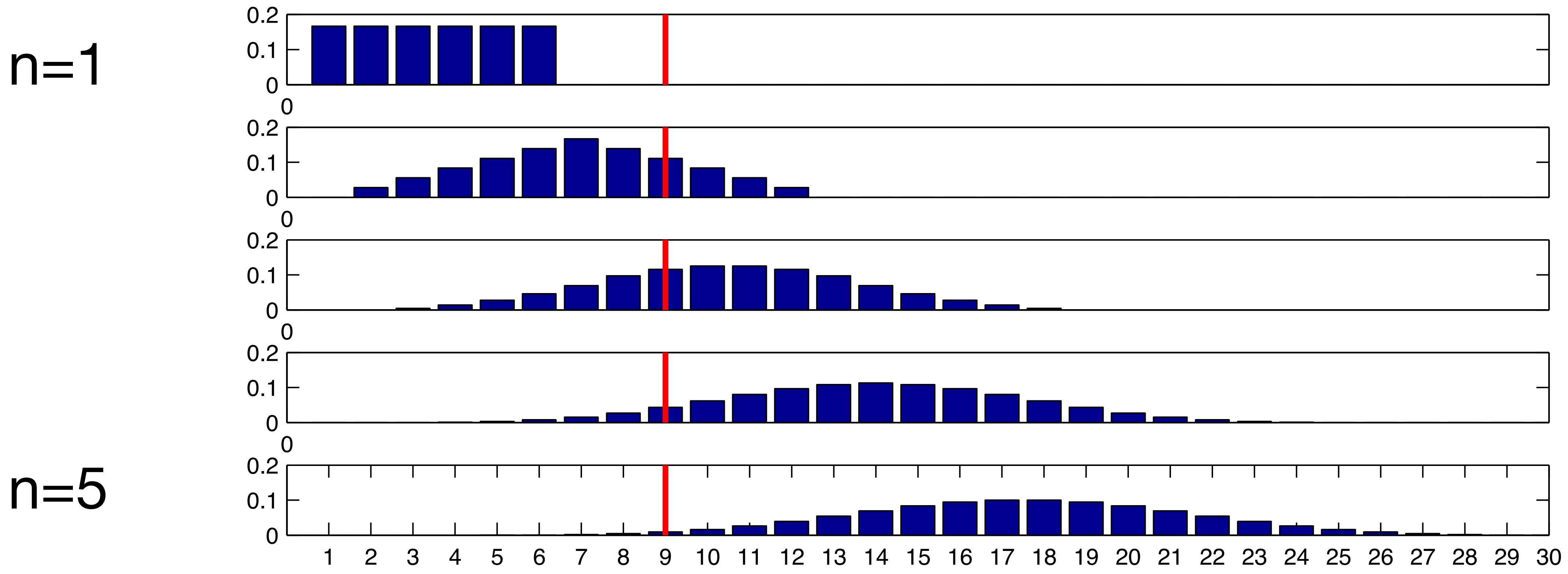
$$p(t \mid n) = \sum_{s_1,\ldots,s_n} p(t, s_1, \ldots, s_n \mid n) = \sum_{s_1,\ldots,s_n} p(t \mid s_1, \ldots, s_n) \prod_i p(s_i)$$

$$= \sum \Box \left[t = \sum_i^n s_i \right] \prod_i p(s_i)$$

where $p(s_i) = 1/6$ for all scores s_i

$$\Box \left[t = \sum_i^n s_i \right] = \begin{cases} 1 & \textbf{if } t = \sum_i^n s_i \\ 0 & \textbf{otherwise} \end{cases}$$

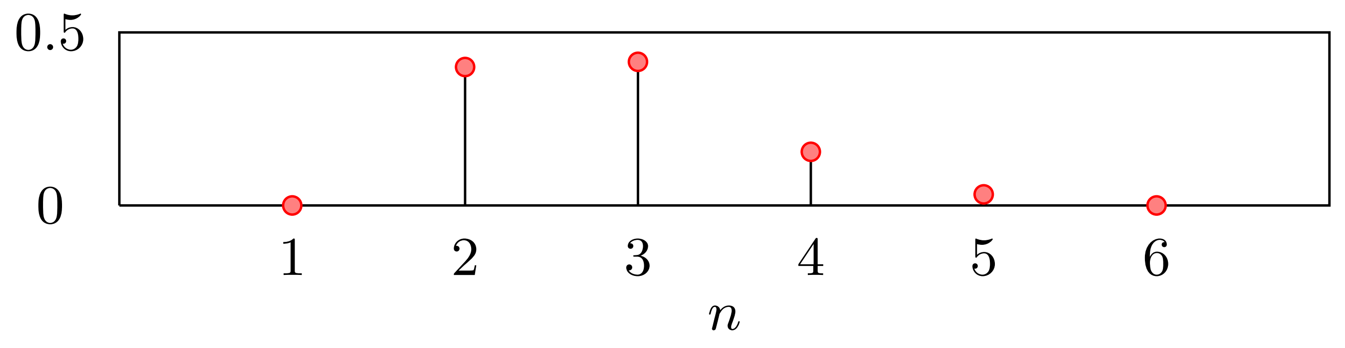
By enumerating all 6^n states, we can explicitly compute $p(t\,|\,n)$



As the models explaining the data become more 'complex', in this case as n increases, more states become accessible.
Models that can reach more states have lower likelihood due to normalisation over t.

Complexity penalisation

The posterior $p(n \mid t = 9)$



- A posteriori, there are only 3 plausible models, namely n = 2, 3, 4 since the rest are either too complex, or impossible.

- As the models become more 'complex' (n increases), more states become accessible and the probability mass typically reduces.

- This demonstrates the **Occam's razor** effect which **penalises models which are over complex.**

1. One way to understand the Ockham's razor effect in Bayesian model selection is to notice that the marginal likelihood can be rewritten as follows (*using the chain rule of probability*):

$$p(D \mid M) = p(y_1)p(y_2 \mid y_1)p(y_3 \mid y_1, y_2) \cdots p(y_N \mid y_1, \ldots, y_{N=1})$$

We predict each future point given all previous ones.

- If a model is too complex it will overfit the "early" examples and will then predict the remaining ones poorly

2. Another way is to note that the probabilities must sum up to one

- Complex models (which can predict many things) must 'spread' their probability mass thinly

➡ will not obtain as large a probability for any given data set as simpler models

Summary

- Model selection defines the hypothesis space
- In both Bayesian learning and MAP learning, the **hypothesis model prior** $P(M)$ plays an important role
- If the hypothesis space is too expressive it contains many hypotheses that fit the data set well, leading to **overfitting**
- Rather than placing an arbitrary limit on the hypotheses to be considered, Bayesian and MAP learning methods use the prior to penalise complexity.
- The hypothesis prior embodies a **tradeoff** between the complexity of a hypothesis and its degree of fit to the data.