Lecture 7: Model Accuracy

One way of defining the accuracy of a model is how well it represents a data set. This is the likelihood of a data set Ds given a model Bn and is written as:

$$P(Ds|Bn) = \prod_{data} P(Bn)$$

Where $P(Bn)$ is the joint probability of the model variables. The usual intention is that the data set represents the relationships between the variables accurately.

We can illustrate this concept of model accuracy with a very simple example. Suppose we have a Bayesian network with just two nodes, X and Y.

The joint probability of the variables is $P(X\&Y) = P(X)P(Y|X)$ and so the model accuracy with a set of data represented by $[x_i, y_i]$ is:

$$P(Ds|Bn) = \prod_i P(x_i)P(y_i|x_i)$$

For example, suppose we have the following data set:

X	Y
x_1	y_1
x_2	y_1
x_2	y_2
x_2	y_2

$$P(X) = \begin{bmatrix} 1/4 & 3/4 \end{bmatrix}$$

$$P(Y|X) = \begin{bmatrix} 1 & 1/3 \\ 0 & 2/3 \end{bmatrix}$$

The likelihood of the data set given the model can be calculated as the product of the probability of each data point:

$$P(Ds|Bn) = (1/4 \times 1) \times (3/4 \times 1/3) \times (3/4 \times 2/3) \times (3/4 \times 2/3) = 1/64 = 0.0156$$

Now suppose that we want to calculate the likelihood of a different model given the same data. This second model assumes that X and Y are completely independent:

The data is the same, but this time we have just the prior probabilities in the model:

$$P(X) = [1/4 \quad 3/4]$$
$$P(Y) = [1/2 \quad 1/2]$$

and so

$$P(Ds|Bn) = (1/4 \times 1/2) \times (3/4 \times 1/2) \times (3/4 \times 1/2) \times (3/4 \times 1/2) = 0.0066$$

The second model has lower liklihood than the first, and so we can conclude that the first model fits the data better.

Log Likelihood

Clearly, the value of $P(Ds|Bn)$ goes down dramatically with the number of data points, and in practice we will be dealing with data sets with thousands of points. Consequently, in order to keep the model accuracy measure in a reasonable numeric range, and avoid numerical underflow, it is more common to use the log likelihood:

$$log_2(P(Ds|Bn))$$

Taking log_2 creates what is commonly called an 'information measure since $log_2 N$ bits are required to represent integers up to N. Taking the log likelihood for our simple examples gives:

$P(Ds|Bn) = 0.0156$
$log_2(P(Ds|Bn)) = -6$

$P(Ds|Bn) = 0.0066$
$log_2(P(Ds|Bn)) = -7.24$

Model Size

It is not surprising that the first model represents the data better, since it uses six conditional probabilities rather than four. Thus there are more probability values to represent the same data. This means that in general, log likelihood is not sufficient on its own to compare competing models since the one with the most arcs will always be the most likely. Hence, we want also to include some measure of model size on the basis that we would like to choose the smallest model that represents the data well.

One way to estimate the model size is to count the number of parameters. The simplest way of doing this would be to count the total number of entries in the link matrices and the prior probability vectors, but for the purists this is not quite right. For example, each prior probability is a vector of m probability values, but it can be represented by $m-1$ parameters. The m^{th} probability value can be calculated from the others, since the sum of the prior probabilities is always 1. Similarly, each link matrix has $n \times m$ probability values which can be represented by $(n-1) \times m$ parameters where the n dimension refers to the number of states in the child node.

A further subtlety is that as the data set increases in size, the parameters can be represented more accurately. We noted before that to represent integers up to N requires at most $\lceil log_2 N \rceil$ bits. We note that on average we need $(log_2 N)/2$ bits to represent a set of integers up to and including N. This results in the following information measure of model size:

$$Size(Bn) = |Bn|(log_2 N)/2$$

Where $|Bn|$ is the smallest number of parameters we need to represent all the probabilities (prior and conditional) and N is the number of data cases used to calculate them.

Minimum Description Length

The minimum description length principle states that the best model to represent a set of data is the smallest that will produce the required accuracy. This suggests a metric as follows:

MDLScore = ModelSize - ModelAccuracy

Thus using the measures that we have developed so far we get:

$$MDLScore(Bn\&Ds) = |Bn|(log_2 N)/2 - log_2(P(Ds|Bn))$$

We can now assert that between competing models the best one has the lowest MDL score, although we need to exercise caution since the MDL is only one possible measure of model quality. The MDL score is also referred to as the BIC (Bayesian Information Criterion) score.

Returning to our simple example, we can compute a MDL score for each of the two models. In each case we have that the number of data points N is 4, and hence $log_2 N/2 = 1$.

For the connected model the number of parameters is 3 - for example, we can construct $P(X)$ from $P(x_0)$ and $P(Y|X)$ from $P(y_0|x_0)$ and $P(y_0|x_1)$). Thus:

$$|Bn|(log_2 N)/2 = 3$$
$$log_2(P(Ds|Bn) = -6$$
$$MDLScoreScore(Bn\&Ds) = 9$$

For the independent model the number of parameters is 2 - for example, we can construct $P(X)$ from $P(x_0)$ and $P(Y)$ from $P(y_0)$. Thus:

$$|Bn|(log_2 N)/2 = 2$$
$$log_2(P(Ds|Bn) = -7.24$$
$$MDLScore(Bn\&Ds) = 9.24$$

Thus comparing the MDL scores we find that the dependent network is only marginally the better model to represent this data.

The data set in the above example did not show much dependency between the variables, and the connected network proved to be only marginally better. If the data set has no dependency for example if our data points were $[x_0, y_0][x_0, y_1][x_1, y_0][x_1, y_1]$ then the MDL metric would have preferred the independent network. You can verify this as an exercise if you wish. As a second example we will look at a different data set where there is clear dependency:

X	Y
x_0	y_0
x_1	y_0
x_1	y_1
x_1	y_1
x_1	y_1
x_0	y_0
x_0	y_0
x_1	y_1

$$P(X) = \begin{bmatrix} 3/8 & 5/8 \end{bmatrix}$$
$$P(Y|X) = \begin{bmatrix} 1 & 1/5 \\ 0 & 4/5 \end{bmatrix}$$

$$P(X) = \begin{bmatrix} 3/8 & 5/8 \end{bmatrix}$$
$$P(Y) = \begin{bmatrix} 1/2 & 1/2 \end{bmatrix}$$

for the connected model we have:

$$log_2 P(Ds|Bn) = 3log_2(3/8) + log_2(1/8) + 4log_2(1/2) = 3log_2 3 - 9 - 3 - 4 = -11.245$$

and for the independent model we have:

$$log_2 P(Ds|Bn) = 3log_2(3/16) + log_2(5/16) + 4log_2(5/16) = -15.6$$

The model size for this data set is bigger, reflecting the fact that we have twice as many data points to represent: for the connected model:

$$|Bn|(log_2 N)/2 = 3 \times 3/2 = 4.5$$
$$MDLScore = 4.5 + 11.25 = 15.75$$

for the independent model:

$$|Bn|(logN)/2 = 2.3/2 = 3$$
$$MDLScore = 3 + 15.6 = 18.6$$

The connected model fits the data much better, and the MDL metric definitely prefers it despite its greater size.

Search for the best network

Having defined a metric on the network and the data, we can now adopt a new approach to finding networks from data. Since we can choose between competing models, we can search the space containing all possible models to find the best one. Exhaustive search is possible for small numbers of variables. It would proceed, breadth first, by constructing a tree of all possible networks.

Let us suppose we have four variables A,B,C and D. We start from the independent network (no arcs), and calculate its MDL score.

Now we consider adding one arc. There are $\binom{4}{2}$ possible ways in which we can do this: AB, AC, AD, BC, BD and CD. For each of these possibilities we create a new network and measure its MDL.

We now continue expanding the tree in the same way until all possible networks have been made. at which point we select the network with the lowest MDL score.

Unfortunately exhaustive search is only possible for small networks. To see this we note that if a network has n variables, then the total number of possible arcs is $(n-1) + (n-2) + ... + 2 + 1 = n(n-1)/2$. In a particular network each of the possible arcs is either present of absent, and so there are in total $2^{(n(n-1)/2)}$ different networks that can be constructed with n nodes.

Thus for the four node example above there are $2^6 = 64$ possible networks. However for six variables there are 32K possibilities and for eight variables there are 268M possible networks making the computation infeasible. In practice, if we exclude all networks which are not singly connected then we will reduce the number of cases considerably. However the number is still rising exponentially with the number of nodes.

Another possibility is to prune the tree during the depth first search using some heuristic rules. Possibilities could include:

1. If a network has a higher MDLScore that its parent remove it from the search tree.
2. Add arcs only if the mutual information between the variables is high
3. At each level expand only networks with low MDL scores

These strategies can dramatically reduce the search time, but they all run the risk of finding a sub-optimal solution.

MDL is not an absolute measure

Consider a data set where:

$$log_2(P(Ds|Bn) = s$$

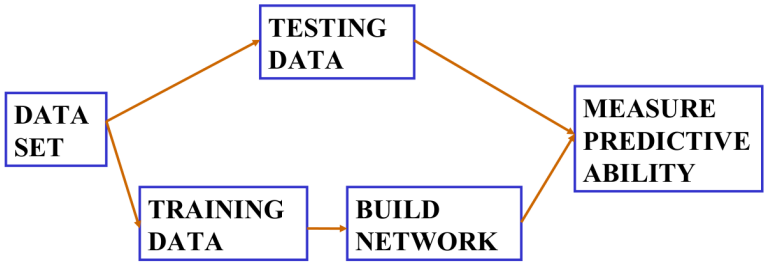
Suppose now that we create a new data set by duplicating every data point in the first, then, for this second set:

$$log_2(P(Ds|Bn) = 2s$$

Furthermore, for the second data set $log_2 N$ increases by 1, and thus the overall MDL score increases even though the network is the same and the data distribution is the same. Thus MDL can only be used to compare two networks using the same data set.

Using Prediction accuracy to determine a classifier

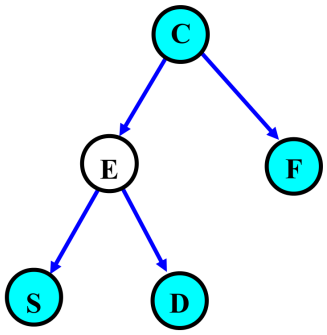
Prediction accuracy is an alternative way measuring model accuracy which can give a more concrete absolute result. The idea is to split the training data into two - a training set and a testing set and use the latter to measure the accuracy of the network.



Choose a target variable eg E.

For each point in the testing data set:

- Instantiate C, S, D and F
- Calculate $P'(E)$
- Calculate $|P'(E) - D(E)|$ (where $D(E)$ is the data value of E



The smaller the average $|P'(E) - D(E)|$ the more accurate the network

One example of using prediction accuracy, which is a variant on the spanning tree for cases where the class node is known, is a model proposed by Enrique Sucar. It requires a measure of quality of the tree. This can be achieved by testing the predictive ability of the network. This can be done by splitting the training data into two, one set used for training and one for testing.

The steps are as follows:

- Build a spanning tree and obtain an ordering of the nodes starting at the root.
- Remove all arcs
- Add arcs in the order found in step 1
- If the predictive ability of the network is good enough (or the nodes are all joined) stop, otherwise go to step 3