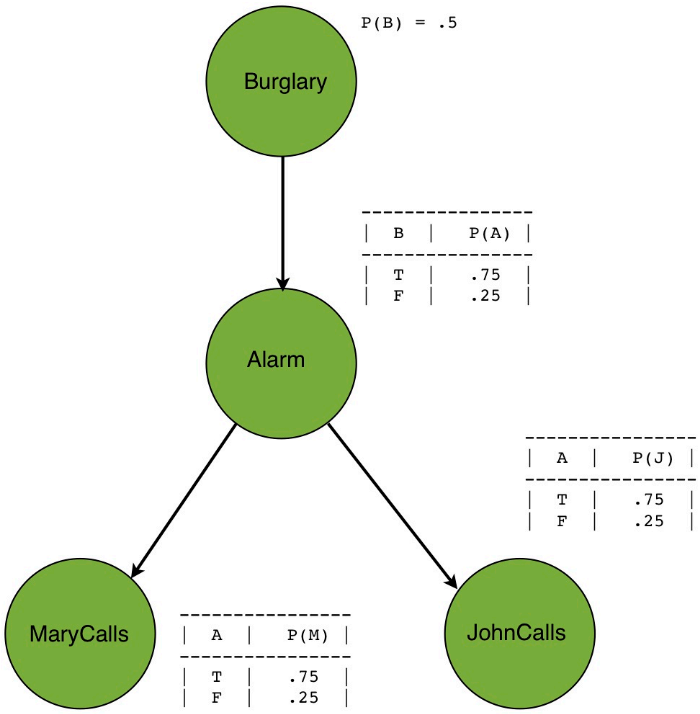
Consider that there exists a network with the following underlying distribution.



For convenience let B = Burglary, E = Earthquake, A = Alarm, M = Marycalls, and J = Johncalls. The probabilities have been simplified to make the math easy to follow.

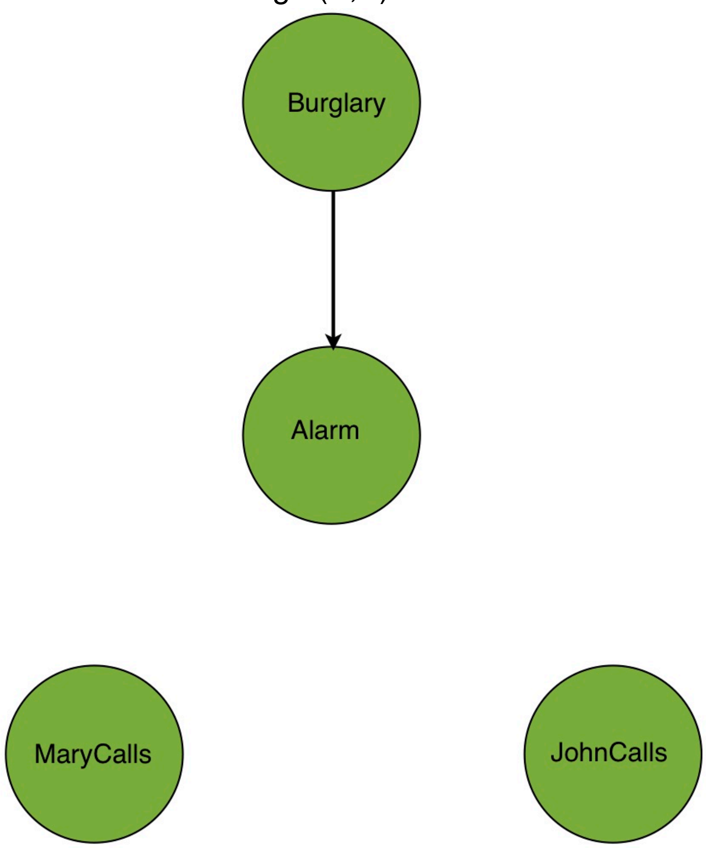
Now say that we gather the following set of data from this network (1 = True, 0 = False):

B	A	M	J
1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	0
1	1	1	0
1	1	1	0
1	1	1	0
1	1	1	1
1	1	1	1
1	1	0	1
1	1	0	1
1	0	1	0
1	0	0	1
1	0	0	0
1	0	0	0
0	1	1	1
0	1	1	1
0	1	0	1
0	1	0	1

0	0	1	0
0	0	1	0
0	0	1	0
0	0	0	1
0	0	0	1
0	0	0	1
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

In inferring the network we assume we know nothing about the network. We have four nodes and our data. We begin with a graph with no edges and try adding edges one at a time, each time calculating the likelihood score. As long as the score increases by adding an edge, we keep the edge.

First we add the edge (A,B):



Ignoring the constant M, we calculate the sum of the mutual information between connected nodes (remember that nodes without parents have a mutual information of 0):

I_p (A;B) = P(A,B) * log (P(B|A)/P(B))
+ P(¬A,B) * log (P(B|¬A)/P(B))
+ P(A,¬B) * log (P(¬B|A)/P(¬B))
+ P(¬A,¬B) * log (P(¬B|¬A)/P(¬B))

= 12/32 * log ((12/16)/(16/32))

+ 4/32 * log((4/16)/(16/32))
+ 4/32 * log((4/16)/(16/32))
+ 12/32 * log((12/16)/(16/32))

= 0.0568109454
I_p(A;B) = 0.0568109454

Because this value is larger than the likelihood score before the edge was added (which was zero), we keep the edge. Adding the edge (A,M) we get the following I_p score:

I_p(A;B) = 0.0568109454
I_p(M;A) = P(M,A) * log (P(A|M)/P(A))
+ P(¬M,A) * log (P(A|¬M)/P(A))
+ P(M,¬A) * log (P(¬A|M)/P(¬A))
+ P(¬M,¬A) * log (P(¬A|¬M)/P(¬A))

= 12/32 * log ((12/16)/(16/32))
+ 4/32 * log((4/16)/(16/32))
+ 4/32 * log((4/16)/(16/32))
+ 12/32 * log((12/16)/(16/32))

= 0.0568109454

I_p(A;B) + I_p(M;A) = 0.113621891

The score increased so we keep the edge. Adding the edge (J,A) increases the sum of the mutual information scores to 0.170432836. Now let us add an edge which should not be in the network, (J,M), and see if the scoring is able to detect that it does not belong:

I_p(A;B) = 0.0568109454
I_p(M;A) = 0.0568109454
I_p(J;A) = 0.0568109454
I_p(J;M) = P(J,M) * log (P(M|J)/P(M))
+ P(¬J,M) * log (P(M|¬J)/P(M))
+ P(J,¬M) * log (P(¬M|J)/P(¬M))
+ P(¬J,¬M) * log (P(¬M|¬J)/P(¬M))

= 8/32 * log ((8/16)/(16/32))
+ 8/32 * log((8/16)/(16/32))
+ 8/32 * log((8/16)/(16/32))
+ 8/32 * log((8/16)/(16/32))

= 0
I_p(A;B) + I_p(M;A) + I_p(J;A) + I_p(J;M) = 0.170432836

The score did not change by adding the edge and therefore the algorithm does not include the edge. This example illustrates an important concept with likelihood scoring: the only way an edge is not added using likelihood scoring is if there is absolutely no mutual information shared between the two nodes. The data in this example were contrived to reflect perfect independence between J and M; in real-world data true independence is rarely observed. For example, consider when we try to add the edge (B,J) to our network (note that mutual information is independent of the edge direction):

$I_p(A;B) = 0.0568109454$
$I_p(M;A) = 0.0568109454$
$I_p(J;A) = 0.0568109454$

$$
\begin{aligned}
I_p(B;J) = &\ P(B,J) * \log (P(J|B)/P(J)) \\
&+ P(\neg B,J) * \log (P(J|\neg B)/P(J)) \\
&+ P(B,\neg J) * \log (P(\neg J|B)/P(\neg J)) \\
&+ P(\neg B,\neg J) * \log (P(\neg J|\neg B)/P(\neg J)) \\
\\
&= 9/32 * \log ((9/16)/(16/32)) \\
&+ 7/32 * \log((7/16)/(16/32)) \\
&+ 7/32 * \log((7/16)/(16/32)) \\
&+ 9/32 * \log((9/16)/(16/32))
\end{aligned}
$$

$= 0.00340181707$
$I_p(A;B) + I_p(M;A) + I_p(J;A) + I_p(B;J) = 0.173834653$

Our score increased slightly by the adding of the edge and the edge would be erroneously kept. This exhibits the limitation of the maximum likelihood score: a simple network will only be preferred over a more complex network when for two nodes X and Y, X and Y are truly independent in the training data. Otherwise the algorithm will try to build into the inferred graph every trace of dependence that it can, thereby overfitting the data.