Applied Machine Learning - Exercise 6 (02.06.2017)

Ben Wulf, Lie Hong, Amnon Bleich

Task 1)

$$p(x1,x2,x3,x4,x5,x6,x7)=p(x1)*p(x2)*p(x3)*p(x5|x1,x3)*p(x4|x1,x2,x3)*p(x6|x4,x2,x1,x3)*p(x7|x1,x2,x3,x4,x5)$$

Task 2)

$p(R|Y)=1/3$, $p(B|Y)=1/3$, $p(R\cap B|Y)=1/9 \Rightarrow p(R\cap B|Y)= p(R|Y)*p(B|Y) \Rightarrow$

R and B are conditionally independent

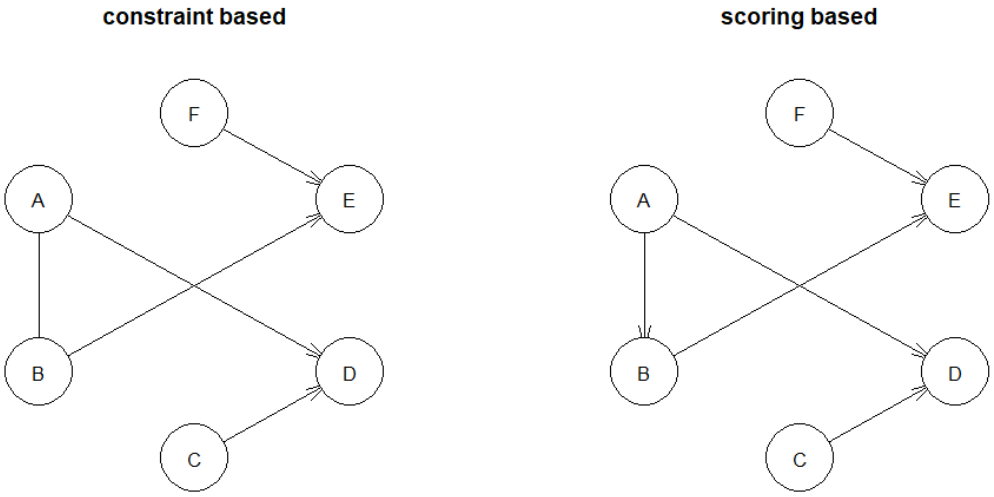
Task 3)

A) If we don't have any information about C or D, then A and B are conditionally independent.
B) Knowing D directly implies information about C, which makes A and B conditionally independent (correlated)
   * for the right figure, controlling F separates the path that existed between A and B. Therefore A and B are d-separated.

Task 4A)

Nothing to report

Task 4B)

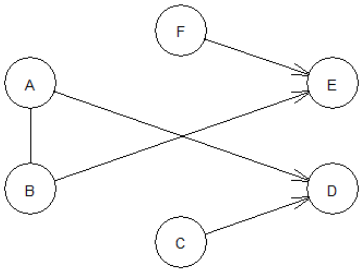


The only different difference is that in the constraint based model the edge between A and B is undirected whether in the scoring based it is directed. (causality vs. correlation). The change from the arc A-B to A B causes an increase of the branching factor from 0.67 (cb) to 0.83 (sb).

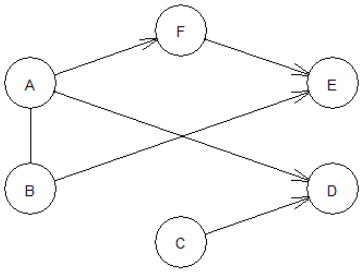
Task 4C)

Add, reverse and remove edges form the graph.

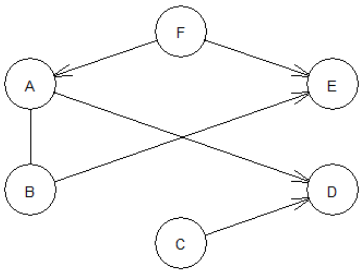
constraint based



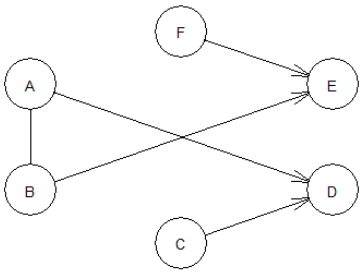
constraint based add edge from A to F



constraint based reverse edge from A to F

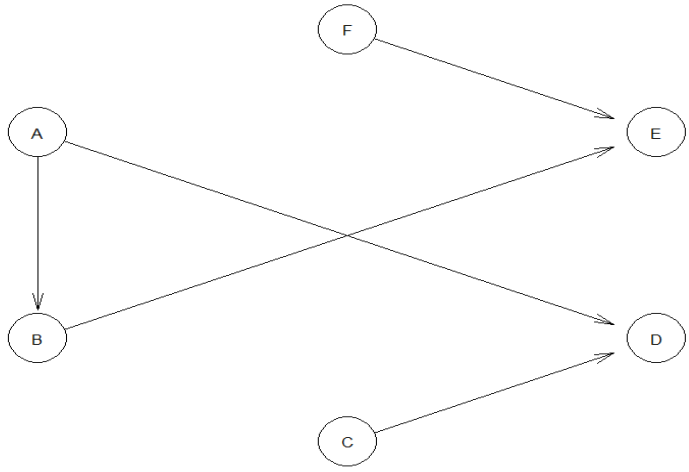


constraint based remove edge from F to A



To turn the constraint based graph into a DAG we use the set.arc function to get an directed A→B edge instead of the undirected A-B edge. With the parameters check.cycles = TRUE,check.illegal = TRUE prevent us to create cycles in bigger examples.

turned to DAG



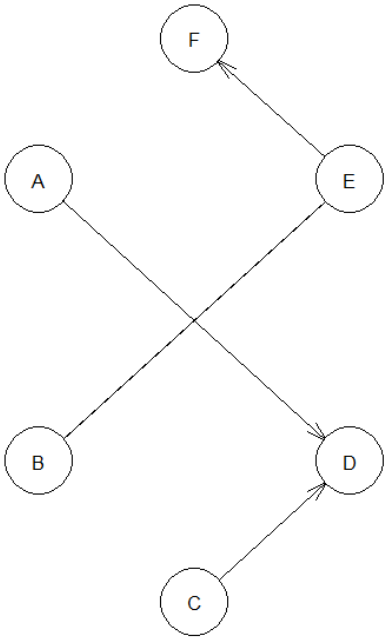
Task 4D)

The bayesian-networks B_1 and B_2 are Markov equivalent if and only if both describe the same conditional dependence and independence statements. B_1 and B_2 have the same V-structures and inferred edges.

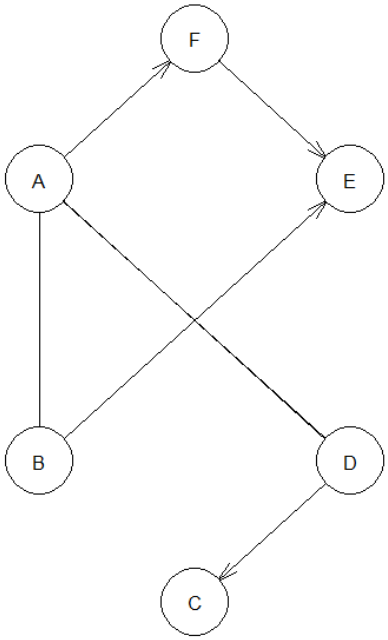
In a first try we blacklist the undirected edge A-B and whitelist the edge E→ F. The leads to a removal of the edge A-B, the edge B→E becomes undirected and the edge F→E is reversed.

In a second try we white list the new edge A→ F and blacklist C→D. This lead to the reverse of edge C->D and to the new edge A→

Blacklisted arc A-B; whitelisted arc E->F



whitelist A->F; blacklist C->D



Task 4E)

We start with an empty graph and add and remove single edges randomly. If the new graph is better we keep it otherwise we use the old one.
The result of our own Hill climbing algorithm looks nearly like the build in algorithm, but the edge A→B is reversed.

own simple HC