Nancy Espinosa Castillo - A01206782
Joel Alexis Bustamante Calvario - A00226215

REPORT 4

Bayesian networks are probabilistic graphical models that are used to organize the knowledge about a domain into a network of causes and effects between key variables. That factor can help to represent a joint distribution function on a finite set of variables and take decisions with this information.

Bayesian networks consist of two parts:

- The qualitative part, that is a graphic structure represented by a graph, describes the possible variables and dependencies between them.
- The quantitative part is composed of conditioned probabilities and it represents the beliefs of the cause-effect relations between nodes.

As an example, the Figure 1 is a Bayesian Network that shows the sales of a company that can have a high or low budget to make a product, also this affects directly to the cost of the product and the quality that can be high or low. Each node of the graph represents a domain variable: Budget, Cost and Quality.

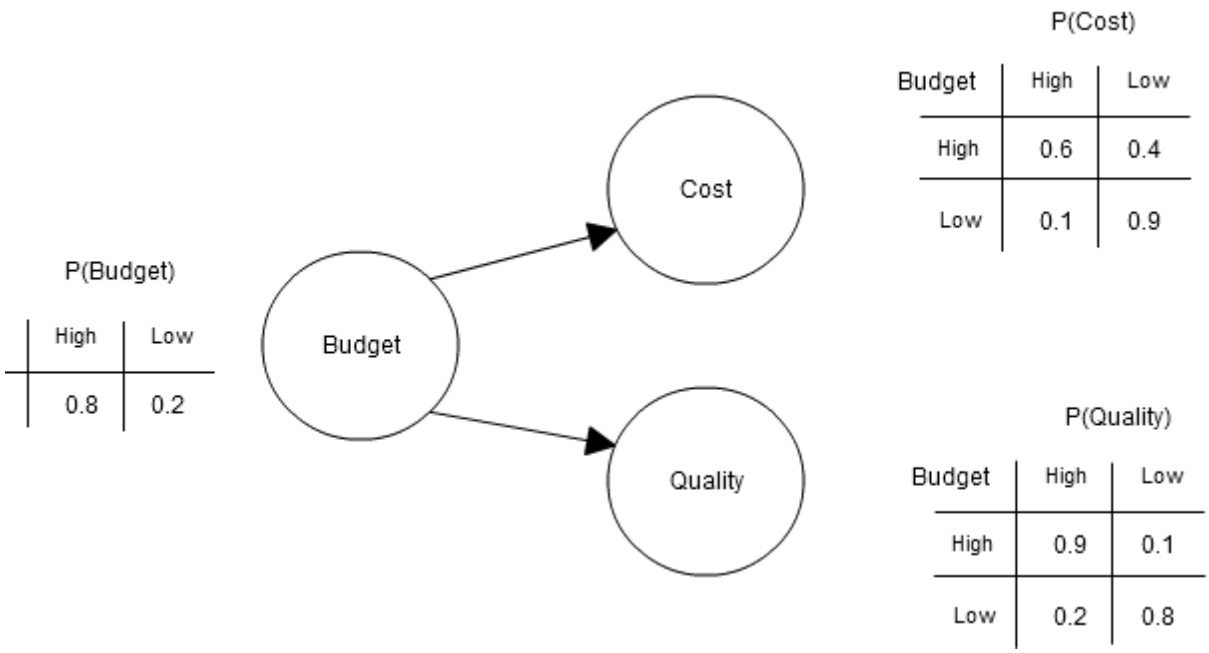

Figure 1. Example problem

We made a Python program before creating this simple Bayesian Network to calculate the conditional probability, we compared the result of our program with a specialized tool, HUGIN Expert.

In the Python program the input sample was

[Nodes]
Budget,Quality,Cost

[Probabilities]
+Budget = 0.8
+Cost|+Budget = 0.6
+Cost|-Budget = 0.1
+Quality|+Budget = 0.9
+Quality|-Budget = 0.2

On the other hand, after creating the Bayesian Network in HUGIN Expert, we obtained the next graphic representation.

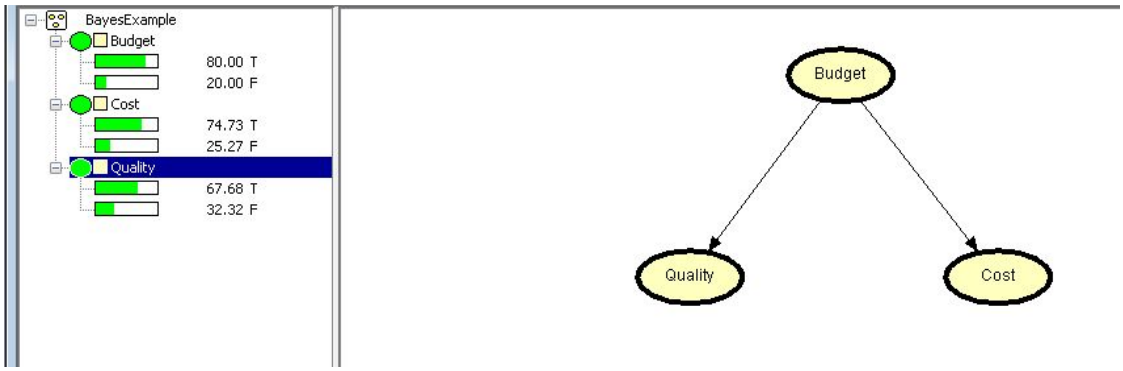

Figure 2

Both programs created the Bayesian Network and the results were the next ones:

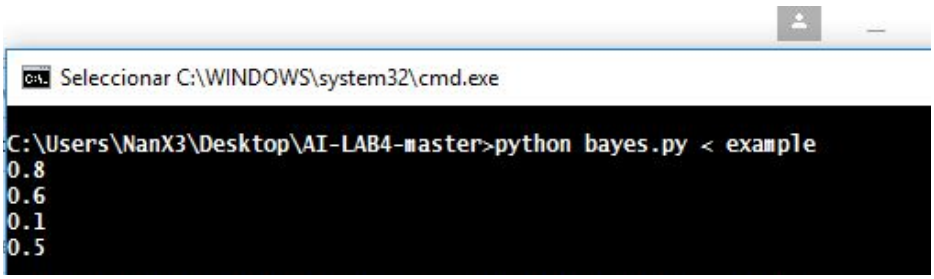

Figure 3. Python program

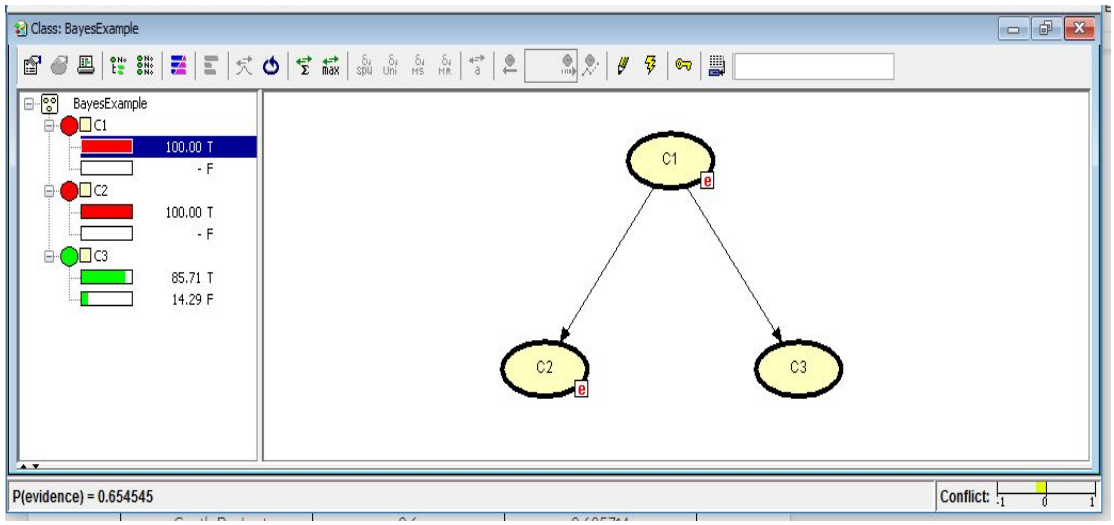

Figure 4. HUGIN Expert where C1 is Budget, C2 is Quality and C3 is Cost

Queries	Python program	HUGIN Expert
+Budget	0.8	0.8
+Cost|+Budget	0.6	0.685714
-Quality|+Budget	0.1	0.145455
+Cost	0.5	0.7447
+Cost|-Budget,+Quality	0.1	0.0068

If we select a fact in a node, then Hugin Expert calculates the total probability of each node, we assume that it does with the junction tree algorithm, in contrast to our implementation, which needs to calculate them using the enumeration algorithm. The results are almost equal in this example, but other cases can present inconsistencies because even though the Hugin Expert tool can calculate total probabilities e.g. $p(+Budget)$, joint probabilities e.g $p(+Budget,+Cost)$ and conditional probabilities e.g , when exist more than one query node the results are not correct. Due this factor, it will be necessary to make a middle step, converting the conditional probability to the division of two joint probabilities.

An important fact is that using Hugin Lite Expert, if we need to calculate more query nodes then it can be a problem with this tool because it can lose accuracy with each iteration. However, it can be an effective tool to represent the problem in a graphical way that makes more easy understand and its algorithm makes faster calculations that the Python program developed by us.

In conclusion, Hugin Lite is an interesting tool that permit us to calculate in a easy way a conditional probability as long as our nodes would not be more of 1 query node, also the algorithm that it uses, it is more efficient and offers a graphical representation. On the other hand, if we need to calculate more than one query node in a directly form then the python program developed by us can be a more useful tool because have the capacity to solve more complex queries but with the disadvantage that is slower.