

Chapter 3

Causal Inference Using Bayesian Networks

Iftekhar Ahmed, Jeffrey Proulx, and Andrew Pilny

3.1 Introduction

The availability of new computational technologies, data collection opportunities, and data size is profoundly changing the nature social scientific analysis. Although traditional social scientific analysis (Content analysis, ANOVA, Regression, etc.) is still very much at the core of scholarly choice, newly found avenues are expanding analytical possibilities for social scientists. Prediction and network analyses are two of the areas impacted by newly found opportunities. Social scientists are now able to generate predictive results beyond traditional regression methods, thus are able to increase the power of social analysis. Hard sciences (i.e., Biology or Physics) have already developed a rich practice of collecting and analyzing massive amounts of data (Lazer et al., 2009). The possibility of dramatic changes in “analyzing, understanding, and addressing many major societal problems” became a reality due to an increase in the availability of informative social science data (King, 2011, p. 719). This data driven social scientific approach, popularly known as “computational social science”, is a slowly growing field within social sciences largely spearheaded by interdisciplinary scientific teams (Lazer et al., 2009).

I. Ahmed (✉)
University of North Texas, Denton, TX, USA
e-mail: Iftekhar.Ahmed@unt.edu

J. Proulx
University of Illinois, Urbana, IL, USA
e-mail: proulx2@illinois.edu

A. Pilny
University of Kentucky, Lexington, KY, USA
e-mail: andy.pilny@uky.edu

A number of new techniques utilized by present day computational social scientists are borrowed from computer science or information technology. Machine learning classification algorithm (MLCA) is one of such technique. MLCA is an umbrella term that consists of a variety of classification algorithms. The actual choice of a MLCA technique depends upon the theoretical and predictive interests of the researcher and the nature of data. Instead of looking for patterns in the dataset, MLCAs use cross-validation techniques to verify patterns in the data. MLCAs divide the data sample into several random samples, search for patterns in the earlier samples (except the last one), create probabilities or rules based on these patterns and then test those rules on the last sample. Bayesian network classifiers are one group of these MLCAs. Bayesian network MLCAs use *posterior probabilities (PP)* to generate classifications using Bayes' formula.

MLCAs became more user friendly for social scientists with the availability of Analytical Graphical User Interfaces (GUI). Weka is one of these available GUI for researchers. Developed at the University of Waikato, New Zealand, "Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization." (<http://www.cs.waikato.ac.nz/ml/weka/>). This chapter introduces Bayesian Network Analysis using WEKA.

"A Bayesian network consists of a graphical structure and a probabilistic description of the relationships among variables in a system. The graphical structure explicitly represents cause-and-effect assumptions that allow a complex causal chain linking actions to outcomes to be factored into an articulated series of conditional relationships" (Borsuk, Stow, & Reckhow., 2004, p. 219). Because of these links between actions and outcomes, social scientists can generate predictive results and develop network structure among variables beyond traditional social scientific approaches to increase the power of analysis. Conditional independence is at the core of Bayesian networks (Pe'er, 2005). Theoretically speaking, variable X is conditionally independent of variable Z given variable Y if the probability distribution of X conditioned on both Y and Z is the same as the probability distribution of X conditioned only on Y: $P(X|Y,Z) = P(X|Y)$. We represent this statement as $(X \perp Z|Y)$. Bayesian networks encode these conditional independencies with a graph structure (Pe'er, 2005, p. 1). A Bayesian network MLAs use *posterior probabilities (PP)* to generate classifications using Bayes' formula (Eq. 3.1):

$$P(y|x) = \frac{P(x|y) \cdot P(y)}{P(x|y) \cdot P(y) + P(x|not\ y) \cdot P(not\ y)} \quad (3.1)$$

whereas $P(y|x)$ is a posterior probability of y (dependent variable) given x (independent variable), calculated by multiplying the likelihood of an attribute (x) given y ($P(x|y)$) and the class prior probability of y ($P(y)$) over that value time the probability of a false positive ($P(x|not\ y)$) and the probability of a case not being y ($P(not\ y)$).

For instance, imagine you wanted to know if group performance (i.e., HIGH) is contingent on whether or not the group implemented a participative decision making (PDM) structure (i.e., TRUE). The Bayesian formula (Eq. 3.2) would try to determine $P(\text{HIGH} \mid \text{TRUE})$:

$$P(\text{HIGH} \mid \text{TRUE}) = \frac{P(\text{TRUE} \mid \text{HIGH}) \cdot P(\text{HIGH})}{P(\text{TRUE} \mid \text{HIGH}) \cdot P(\text{HIGH}) + P(\text{TRUE} \mid \text{not HIGH}) \cdot P(\text{not HIGH})} \quad (3.2)$$

For the current example, we will look at situations where a number of people are working together in a complex environment. A number of these situations like Military training or Firefighting are potentially dangerous and costly. The development of new technology such as games provides us an opportunity to train people in a safer environment. Because of technological development, we can make these training simulations very close to real world actions. We use the term multiteam system (MTS) to describe nested teams engaged in military or firefighting operations. MTSs are “teams of teams” where each team is nested within a larger collaborative group (Marks, DeChurch, Mathieu, Panzer, & Alonso, 2005). The purpose of the experiment was to investigate MTS collaboration dynamics in response to changes in the accuracy of the information environment surrounding teams.

Now consider a Military training simulation using games. Our experiment was conducted using a computer game called Virtual Battlespace 2 (VBS2) (see Pilny, Yahja, Poole, & Dobosh, 2014). VBS2 is a customizable combat simulation environment and is used globally for military training and simulation as it allows researchers to create custom scenarios where the researcher can add or remove stimuli in the simulation environment. Each experiment session lasted approximately three hours during which all participants engaged in two missions. All sessions were implemented in seven, sequential phases. For each mission, participants either played a scenario that contained entirely accurate information or a scenario that contained partially inaccurate information. In the experimental scenario, each MTS contained four participants divided into two teams of two people. Teams were tasked with navigating a map that contained landmarks and hazards along a route to the MTS’s rendezvous point. As each team’s location was unknown to the other team, therefore frequent communication was needed to coordinate activities.

This experiment uses participant’s survey responses to see if we can predict which information condition the MTS assigned to groups. Each survey in this experiment was large and contained many scales and single response items, consequently providing a robust dataset. These are the types of data sets that we earlier mentioned as new possibilities for social sciences. As investigators, our interest is to identify factors that can predict information manipulation. However, the amount of data that we get and the research interest that we have together persuades us to explore new possibilities of social scientific research under the broader term “computational social science”. Here, our particular interest is to see how Bayesian Network Analysis helps us in our investigation.

3.2 Scenario

This tutorial uses data from an experiment investigating MTSs. The purpose of the experiment was to investigate MTS collaboration dynamics in response to changes in the accuracy of the information environment surrounding teams. This chapter uses a subset of the original data. The following sections will walk the reader through the procedures used during data collection, variable selection, and the steps taken to prepare data for analysis.

Participants ($n = 129$) included undergraduate students from a large Midwestern university who were recruited through flyers and course announcements. A total of 38 MTS experimental sessions were conducted. Each MTS was comprised of four participants, divided into two teams of two. Of these 38 sessions, 33 yielded usable data as five sessions needed to be discarded due to recording errors.

Teams were tasked with navigating a map that contained landmarks and hazards along a route to the MTS's rendezvous point. As teams' location was unknown to each other in the MTS, field teams needed to use radio communication to coordinate a synchronous arrival at the rendezvous point. Once arriving at the rendezvous point, teams were given the task of eliminating a group of enemy insurgents. As a collective, the MTS was given three tasks: (1) to record landmarks the team navigated to for reconnaissance, (2) to successfully disarm and neutralize hazards such as explosive devices and insurgent ambushes, and (3) to coordinate a synchronous arrival at the rendezvous point.

While traveling to their rendezvous point, teams were exposed to pre-recorded radio messages that were intended to represent orders from the MTS's commanding officer. These radio messages took the form of audio played through each participant's headset that played when teams reached certain locations in the simulation. These messages gave teams information regarding the path that lie directly ahead of them and were also used to assign teams tasks such as confirming that an object exists along the path (e.g., a suspicious looking backpack) or exiting their vehicle to disarm an explosive device. Teams were exposed to ten messages during each mission. In the accurate condition all ten messages contained correct information that teams could verify within the simulation (e.g., if they are told there is a suspicious abandoned vehicle ahead, the suspicious vehicle actually existed). In the inaccurate condition two of the ten messages were inaccurate (e.g., if a team is told there is a suspected explosive device ahead, there was no explosive device).

All self-report items were measured at the individual level and observed measures were coded for each team in each session. As observed data were aggregated to the team level, there are two observations per session, one for each team. Screen recordings (videos) were used to construct a behaviorally anchored coding system and each MTS was coded for the five outcomes used in this analysis. Two independent raters coded each video and coding was largely objective. Kappa was used to measure interrater reliabilities and exceeded .90 in all cases thus suggesting an acceptable level of agreement.

In this tutorial, we will be using the participant's survey responses to predict manipulation of information condition in the MTS missions. Each survey in this

experiment was large and contained many scales and single response items. Additionally, there are several observed outcome variables recorded for each mission. In this case, we chose 118 variables that are a mixture of scales, single survey items, and outcome variables to demonstrate how Bayesian networks can be used to accommodate robust datasets.

3.2.1 Variables

118 variables from the dataset were initially explored in this tutorial (1 dependent and 117 independent). We will be using *information accuracy* as our dependent variable (it is called MA_Accurate). Information accuracy was manipulated as a counter-balanced fixed effect in the experiment as each team was randomly assigned to either the accurate or inaccurate condition for their first mission. For the accurate information condition, all information given to teams was accurate, for the inaccurate condition, information given to teams contained two erroneous pieces of information. The remaining variables are treated as independent variables in this tutorial.

3.2.2 Data Preparation

Prior to analysis, data need to be cleaned and formatted. In this case, our data preparation involved two steps. First, all data were merged into a single file containing all of the variables we will use in our analysis. This means that survey responses, observed outcomes, and a dummy coded variable indicating which information condition the participant was in were combined into a single file. All variables are numeric.

After merging data, we removed any cases that containing missing values. In this case, a recording error occurred resulting in five sessions with partially mission data. These sessions were removed list wise. In order to analyze the dataset in Weka environment, we created a comma delimited file (.csv). MLCs work best with Binary Dependent Variable that we are going to predict. However, we can also use Nominal Variable with more than two categories.

3.3 Description of Weka Environment

This section describes the Weka GUI and how to explore different options to run an analysis. Figure 3.1 shows the opening window. The Explorer button allows us to locate and choose the data file that we are going to use. Once you click the explorer tab, it will open a window that provides an Open File option (Fig. 3.2). That option helps us to explore our data location and choose the file we will use for this experiment.

Fig. 3.1 WEKA main GUI

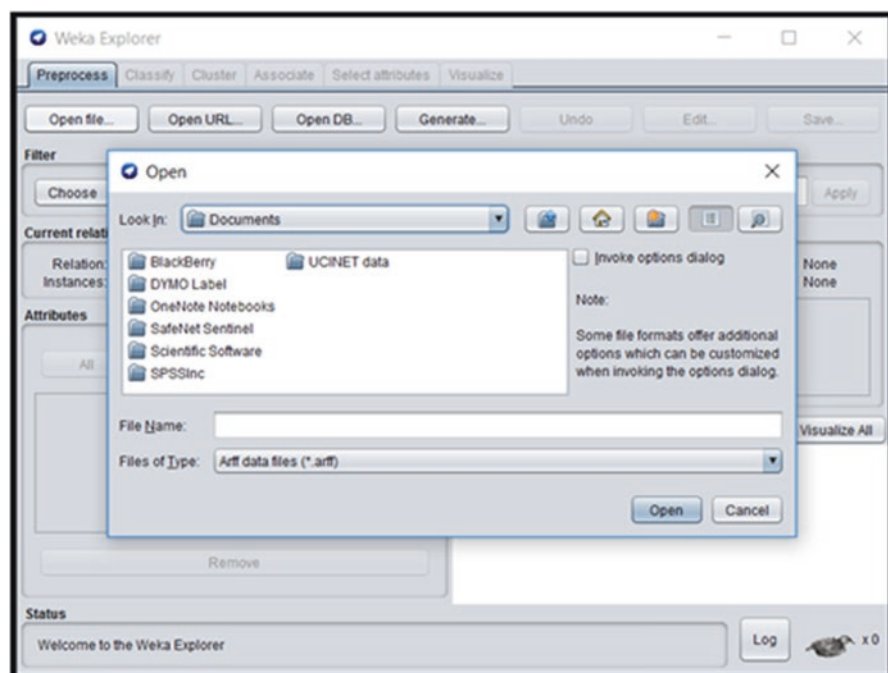


Fig. 3.2 WEKA explorer window

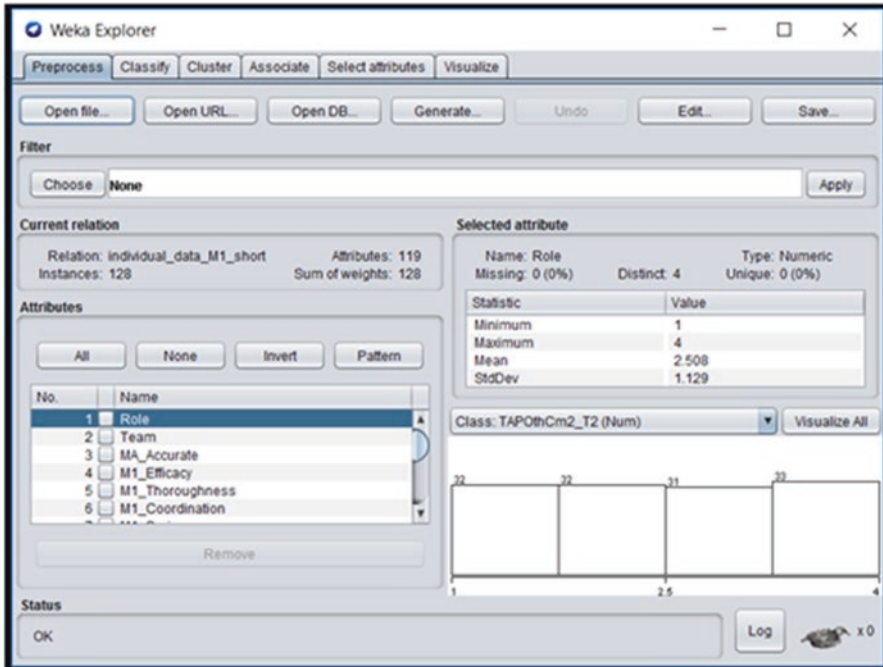


Fig. 3.3 Variables and basic statistics

Once the file is loaded in Weka, you can click any variable and see the basic statistics including maximum and minimum value, mean, standard deviation, and also visual representation of the data (Fig. 3.3).

Figure 3.4 shows the variable that we are going to predict. It shows that the variable is nominal with two categories. As you can see, category I has 63 cases and category A 65 cases. It is recommended to have either exact or very close number of cases in categories for better prediction results because classification problems can occur from imbalanced data. A data set is imbalanced if the classes are not represented equally within the data set. It is common to have data sets with small imbalances. However, large imbalances would definitely cause a problem. The best way to tackle the problem is by collecting new data. If that is not possible, then the option is to generate synthetic samples to balance the classes. This synthetic sample generation usually randomly sample attributes from minority class instances. If there are discrepancies, Weka allows to ‘under-sample’ or ‘over-sample’ a category. Over-sampling in Weka resamples datasets by applying the Synthetic Minority Oversampling TEchnique (SMOTE).

The Classify tab allows us to run classification algorithms. Figure 3.5 shows us a list of classifiers available based on the nature of our data. The first option here provides different Bayes classifiers. For this experiment, we are using BayesNet classifier.

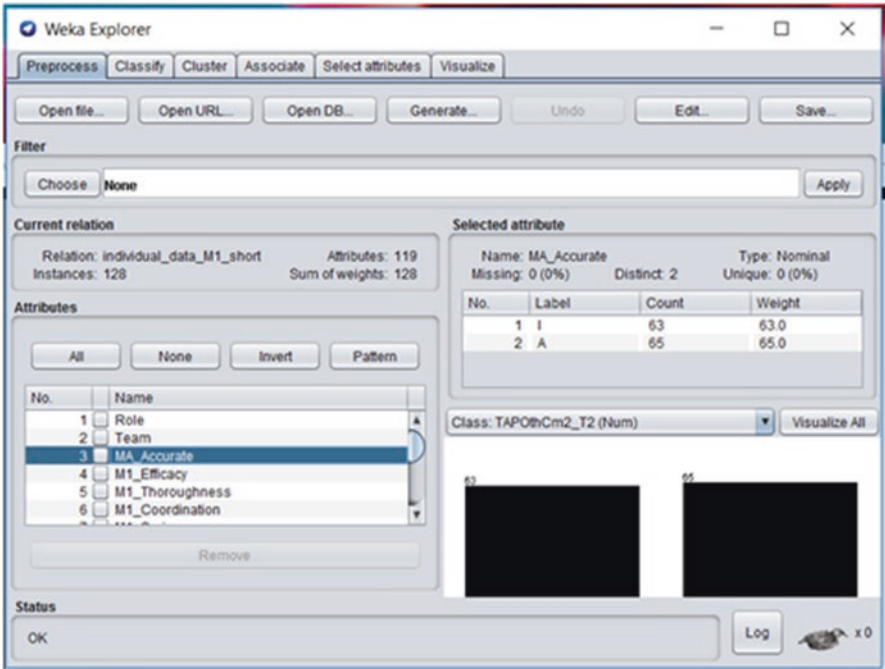


Fig. 3.4 Variables in prediction analysis

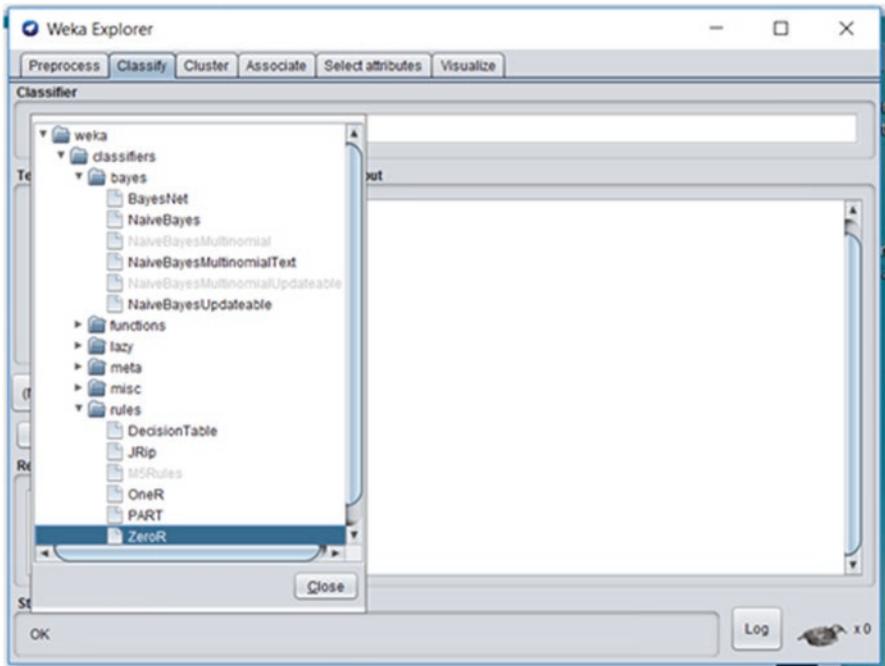


Fig. 3.5 List of available classifiers

3.4 Running Bayesian Network Analysis in Weka

3.4.1 Analysis with All Variables

There are 118 variables in our data set. MA_Accurate is the variable that provides us binary condition that we are interested to know. The rest of the variables will be used to predict MA_Accurate. There are three steps of running this experiment. First, in order to see the prediction power of our data set, we are going to run an analysis with all the variables. Having so many predictor variables is not a good practice as it makes explanation complicated. We are using it for two reasons. First to demonstrate advanced analysis possibilities so that we can use the technique if experimental situation demands such a robust analysis. Second, we like to compare results between all variables and few important variables that we are going to select later. Figure 3.6 shows us the basic window. As you can see, the classifier choice is BayesNet and the variable button shows MA_Accurate. It also shows that the variable is Nominal. To run the classification algorithm, we simply need to hit the start button. There is one additional step to remember. Figure 3.6 shows that, to determine the predictors of information accuracy, we are using a tenfold cross-validation method. It means that the algorithm will divide the sample into ten random samples. Then, it will use the first nine to create probabilities and search for patterns and

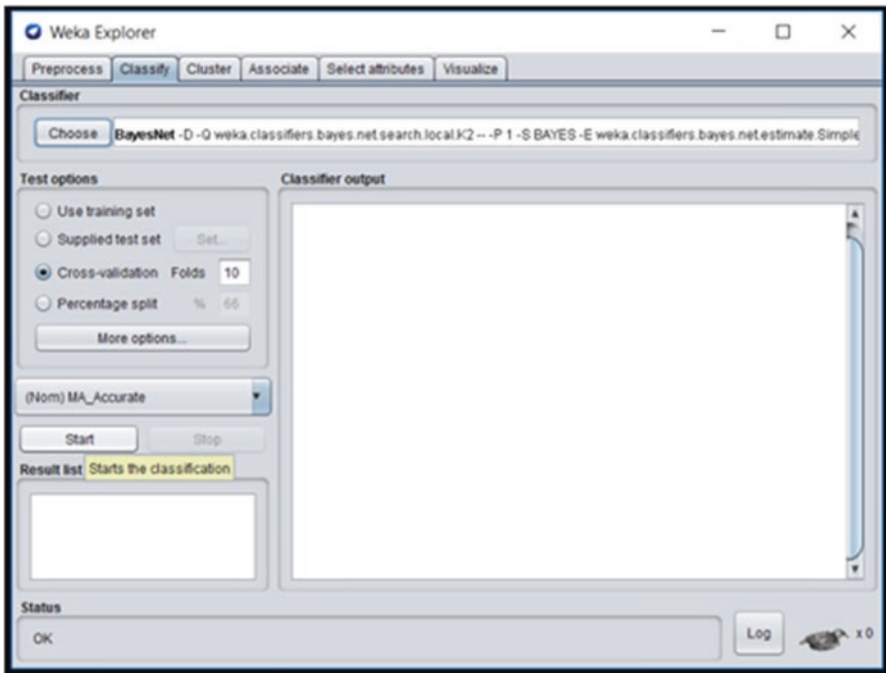


Fig. 3.6 Basic run window

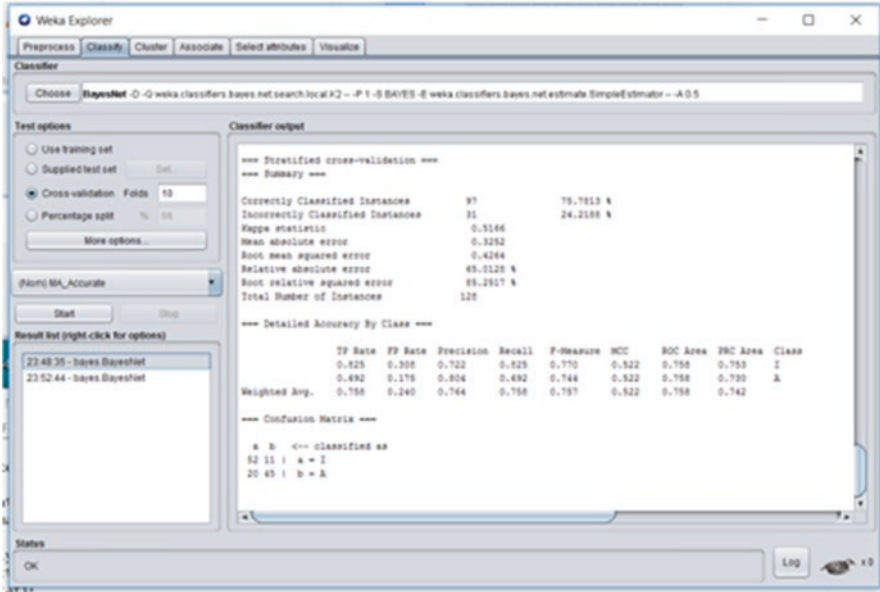


Fig. 3.7 Output window with run results

develop rules based on those patterns. Finally it will test those derived rules on the tenth sample. The number of cross validation choice depends upon the researcher and research interest (e.g., smaller samples may need smaller folds). A user can also supply a completely different data set to test the MLCA. Once we hit the start button, the classifier output window provides the result (Fig. 3.7).

3.4.2 Understanding Weka Output

There are three important sections of the output that together provides us a clear picture of our analysis. First is the *Stratified cross-validation Summary*. This section provides detail into the number of correctly and incorrectly classified instances and total number of instances. For us these were 90 (70.31 %), 38 (29.68 %) and 128.

The most important output for us is the second part of result - *Detailed Accuracy By Class*. Five important statistics for us are the *Precision*, *Recall*, *F-Measure*, *ROC Area*, and *Class* (Table 3.1).

First, the Weka output table provides the rate of true positives (TP Rate) or the ratio of instances of a given class that was correctly classified and the rate of false positives (FP Rate) or the ratio of instances of a given class that was falsely classified. Then it provides *Precision*, *Recall*, *F-Measure*, *ROC Area*, and *Class*. Precision is the ratio calculated by dividing proportion of true instances of a class by the total number of instances classified as that class. Recall is the ratio calculated by dividing

Table 3.1 Weka output for full model

	Precision	Recall	F-Measure	ROC area	Class
	0.667	0.794	0.725	0.770	I
	0.755	0.615	0.678	0.770	A
Weighted Avg.	0.711	0.703	0.701	0.770	

Table 3.2 Confusion matrix for full model

a	b	← classified as
50	13	a = I
25	40	b = A

the proportion of instances classified as a given class by the actual total of that class. The F-Measure (Eq. 3.3) is calculated by combining precision and recall in the following manner:

$$F = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (3.3)$$

The overall F-Measure (mentioned in the output as Weighted Avg.) is the model accuracy. The accuracy of our test is shown by the ROC Area. A ROC Area value of 1 denotes a perfect test whereas a value of .5 is equal to random guessing. So, a worthwhile value is the one above .5 and better if that is closer to 1. In this test, our value is 0.701, which is good enough to accept results. Class denotes the values of our binary classes. Here ‘I’ represents *Inaccurate* and ‘A’ represents *Accurate* classes of our MA_Accurate variable. Based on our results, we can say that the test can identify whether the information scenario given by the researchers were accurate or inaccurate about seventy six percent of the time. Finally, the *Confusion Matrix* provides the statistics of how many times a particular class was classified rightly or wrongly. Our results indicate that in 50 cases of class I were classified as I (right classification) and in 13 cases as A (Table 3.2).

3.4.3 Assessing Information Gain

Although we have a good model that has a 70 % prediction power, a question of the power of individual variables in prediction remains. Although we have 118 variables in the test, it is good to find out how much each of these variables is contributing to the prediction analysis. *Assessing Information Gain* is one way that allows us exactly to do that. The reason behind this test is to identify and exclude variables that are not contributing much to prediction, eliminating them, thus make the model more parsimonious.

In order to run *Information Gain*, we need to go to *Select attributes* tab and choose *InfoGainAttributeEval* (Fig. 3.8). The select method will automatically

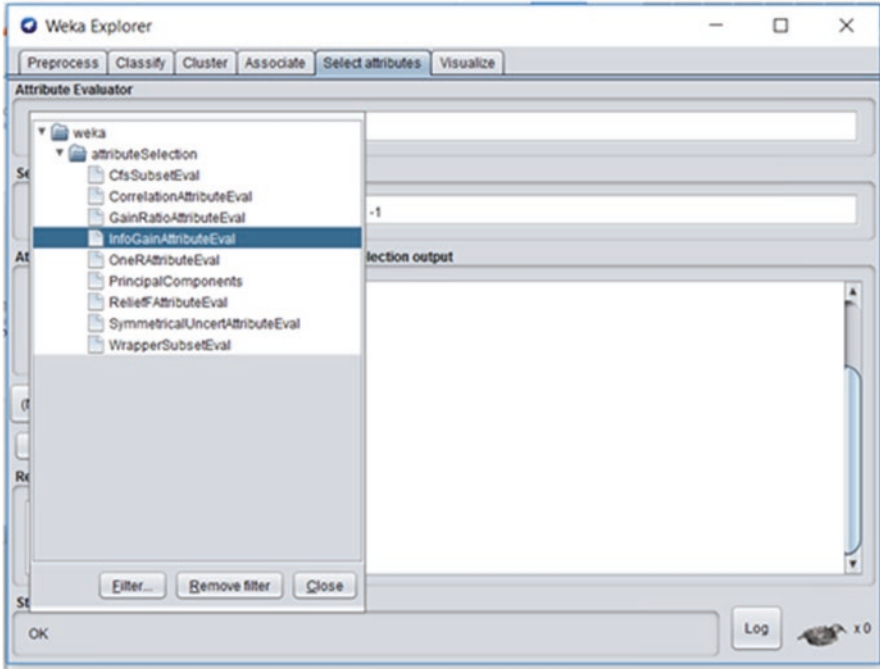


Fig. 3.8 Selection of *InfoGainAttributeEval* in *Select attributes* tab

change to ranker and will provide a window to accept that choice. Once we click OK it will be set. By clicking the start button, we will get the *Information Gain* results in the output area (Fig. 3.9).

In the *Ranked attributes* section, our results identified three variables with some information gain number (Fig. 3.10). The numbers are the amount of information gained from that particular variable (on the left side). Starting from variable '118 TAP0thCm2_T2' the numbers are all 0. It means that those variables are not contributing to any prediction analysis. Three variables our information gain test identified were team efficacy, team thoroughness, and speed. All variables were observed and measured at the interval level.

Team efficacy measures the degree to which teams accomplished their task of neutralizing hazards in the field. High scores of team efficacy were obtained by MTS's that identified and neutralized threats such as explosive devices and insurgent ambushes efficiently and quickly. Low scores of team efficacy indicate an MTS that did not neutralize threats, needed multiple attempts to eliminate threats, or took damage while completing a task. MTSs were placed into three categories based on their scores: (1) High, (2) Average, and (3) Low.

Team thoroughness measures the extent to which teams completed their task of recording the location of landmarks and hazards during their mission. High scores

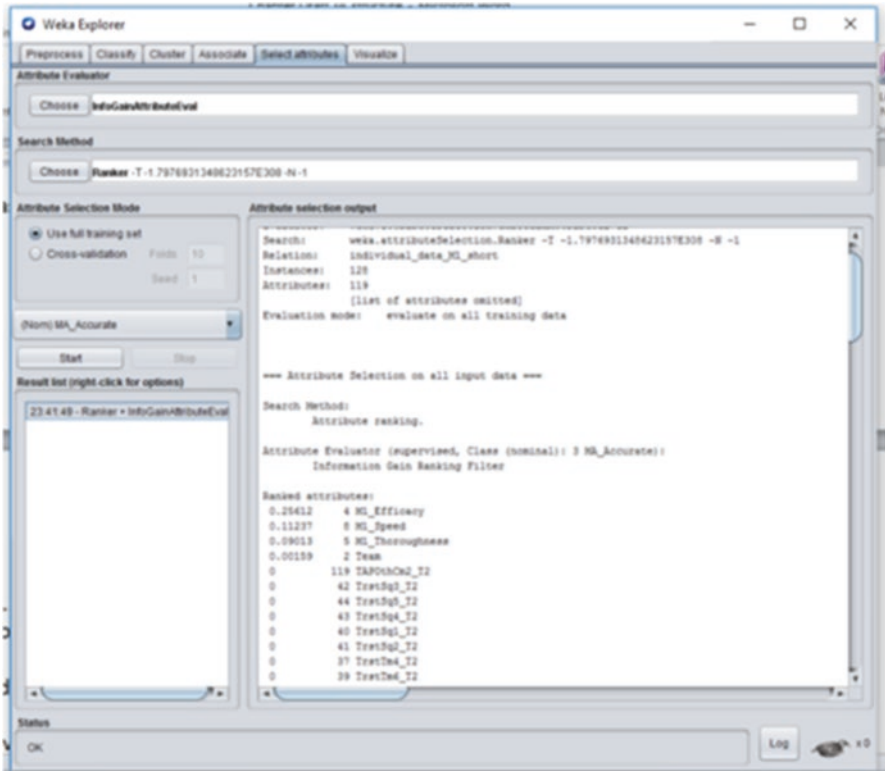


Fig. 3.9 Information Gain results output

of team thoroughness indicate an MTS that correctly identified the name and location of mission landmarks and hazards. Low scores of team thoroughness indicate an MTS that did not accurately record the name or location of landmarks and hazards that they were tasked to locate. They were similarly placed into.

Speed was measured as the time in seconds that it took each team to complete the mission. Completion of the mission was denoted by the moment at which each team first arrived at the rendezvous point and was similarly placed into three categories based on one standard deviation: (1) Long, (2) Average, and (3) Short.

An analysis with only three identified variables with information gain statistics would yield almost similar result. As such, it is time for us to re-run the test with selected variables.

```
=== Run information ===
Evaluator:   weka.attributeSelection.InfoGainAttributeEval
Search:     weka.attributeSelection.Ranker -T -1.7976931348623157E308 -N -1
Relation:   individual_data_M1_short-weka.filters.unsupervised.attribute.Remove-R2
Instances:  128
Attributes: 118
            [list of attributes omitted]
Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:
    Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 2 MA_Accurate):
    Information Gain Ranking Filter

Ranked attributes:
0.2561    3 M1_Efficacy
0.1124    7 M1_Speed
0.0901    4 M1_Thoroughness
0         118 TAP0thCm2_I2
0         41 TrstSq3_I2
0         43 TrstSq5_I2
```

Fig. 3.10 Ranked attributes section of Information Gain result

3.4.4 Re-run with Selected Variables

In order to re-run the test, we need to go back to the Processes tab and select the variables that we need. Here we need M1_Efficacy, M1_Speed, and M1_Thoroughness. We also need our main variable MA_Accurate (Fig. 3.11). Once we select these four variables by clicking the checkbox beside them, we need to click the Invert button right above the list of variables area. This will reverse the selection and will select all variables other than the four we need. Now we can click the remove button right under the list area and remove all unnecessary variables from our analysis (Fig. 3.12). Once this selection process is done, we can replicate the analysis exactly as before.

Table 3.3 shows us our re-run results. As you can see, there is a slight decrease in the overall F-Measure from 0.701 to 0.656. However, the important part to know is that we have significantly decreased much of the noise in the data (i.e., variables that do not predict well), making the data much more interpretable and more substantially (rather than statistically) significant. A look at the probability distribution table can tell us more about the specific odds used to make prediction based on Bayes' theorem.

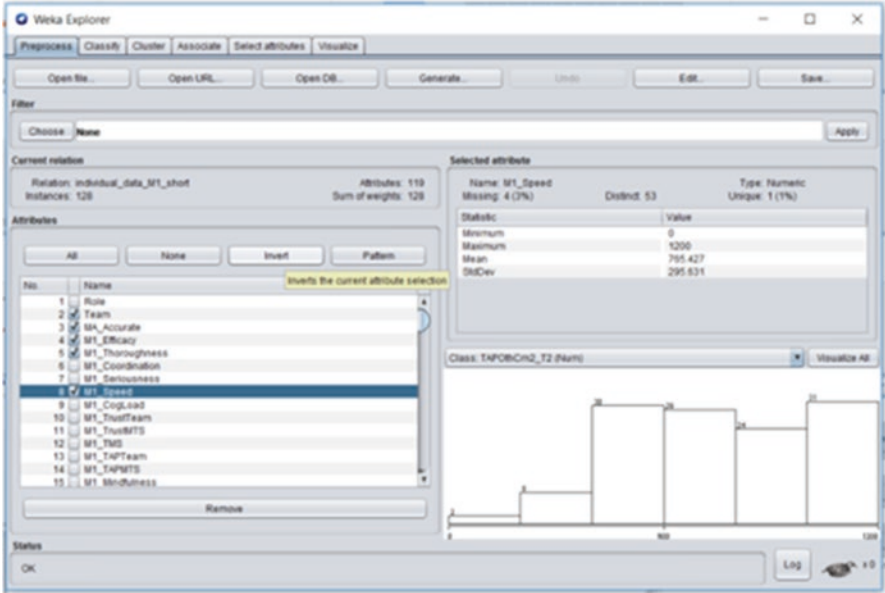


Fig. 3.11 Selection of variables with information gain

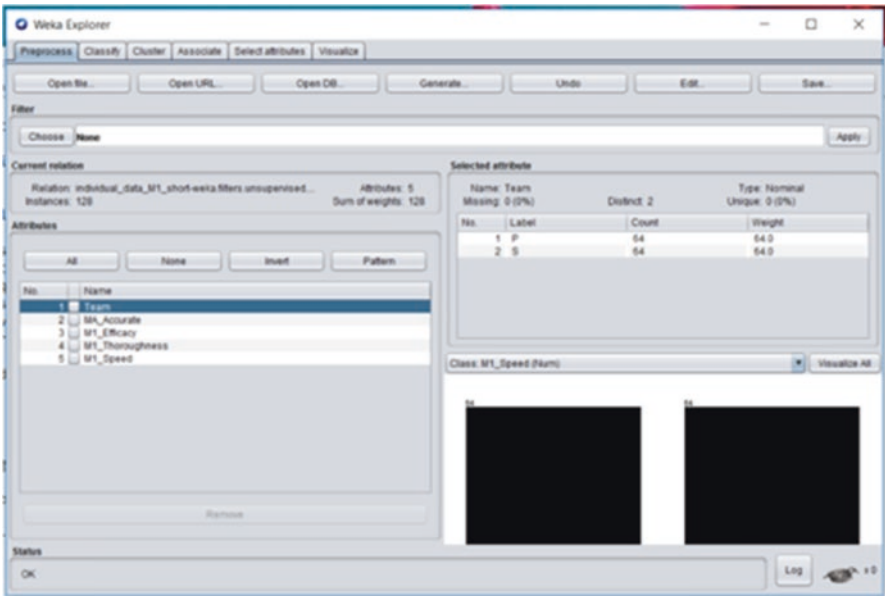


Fig. 3.12 Selected variables for final analysis

Table 3.3 Weka output for reduced model

	Precision	Recall	F-Measure	ROC Area	Class
	0.727	0.508	0.598	0.701	I
	0.631	0.815	0.711	0.701	A
Weighted Avg.	0.678	0.664	0.656	0.701	

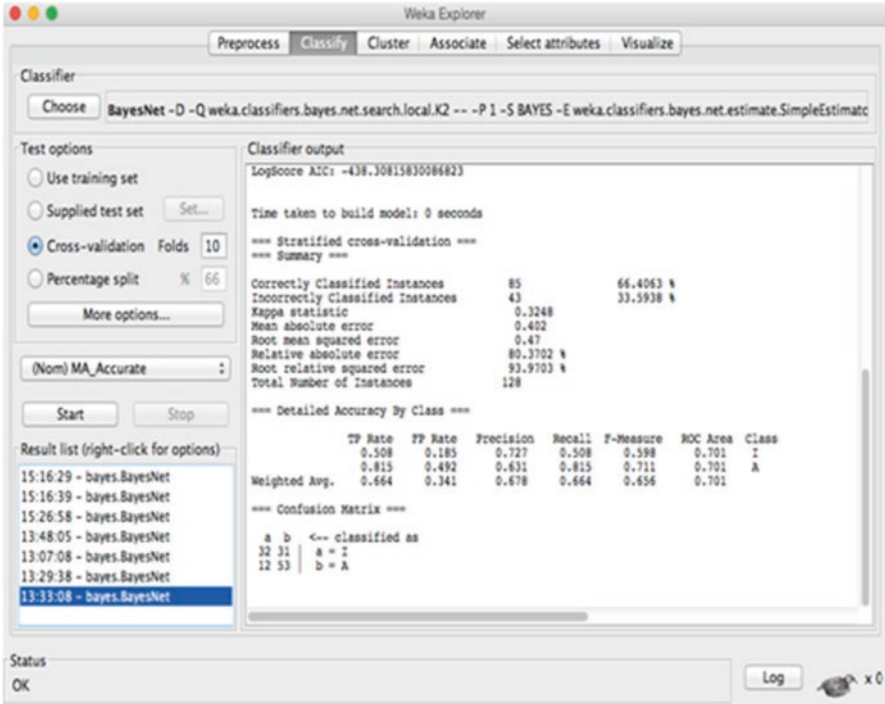


Fig. 3.13 Final prediction model

3.4.5 Probability Distribution

Weka allows viewing the graphical structure of the network. By right clicking the result in the explorer and a drop-down menu appears with a “Visualize graph” option (Fig. 3.13). The graph represents the graphical network with relationship among nodes (Fig. 3.14). This window allows inspecting both the network structure and probability tables. This graph is very useful to identify relationship between nodes. Each node in the graph represents a variable or condition and their relationships represent the network. It is similar to any other network structure.

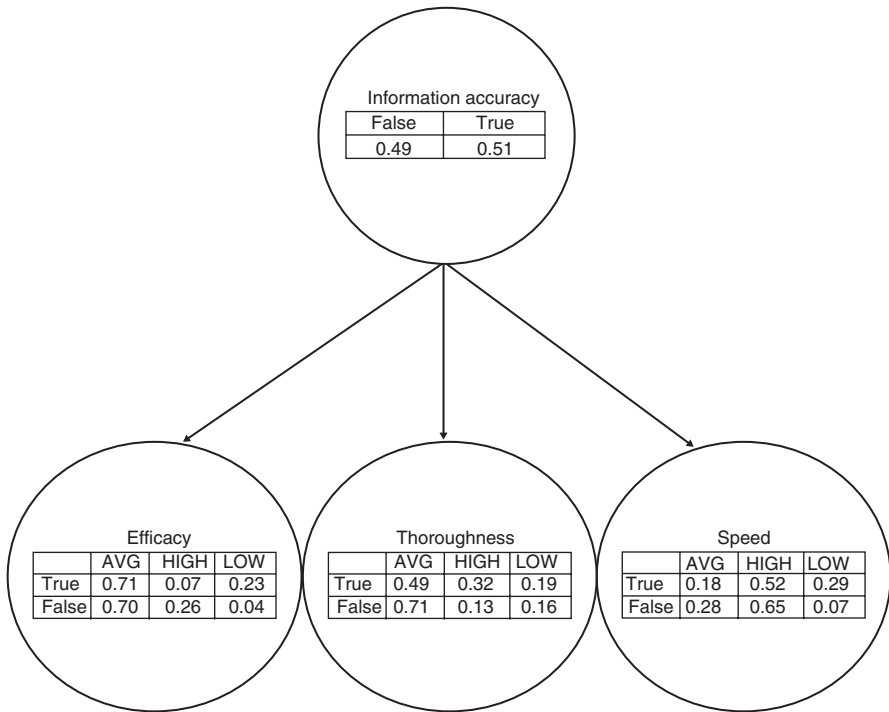


Fig. 3.14 Probability distribution table of one-parent model

Moreover, the network in the graph are directional indicating a directional relationship. If you place your cursor on any node, it will get high lighted. Clicking that node provides the probability table (Fig. 3.15). “The left side shows the parent attributes and lists the values of the parents, the right side shows the probability of the node clicked conditioned on the values of the parents listed on the left” (Bouckaert, 2004, p.29).

Using these probabilities, it is possible to calculate odds using Bayes’ theorem. For instance, consider whether or not there was a relationship between the information manipulation and thoroughness (78 = AVG, 22 = HIGH, 28 = LOW). The corresponding probability of having a high thoroughness score and being in the true information accuracy group was 0.32. If we plug this into Bayes’ theorem, we can determine the posterior probability of an MTS in the true information condition having a high score based on the probabilities given in Fig. 3.14 (see also Witten, Frank, & Hall, 2011, p. 260). To calculate theses, observe that Fig. 3.14 gives the odds of being in the True condition as 0.51 and 0.32 when thoroughness is high. The same odd when information accuracy is False is 0.49 and 0.13. To obtain conditional probabilities, we can use adapt Eqs. 3.1 into 3.4:

$$P(y|x) = \frac{P(x|y) \cdot P(y)}{P(x|y) \cdot P(y) + P(x|\text{not } y) \cdot P(\text{not } y)} \quad (3.1)$$

$$\frac{PP(\text{True Info}) \cdot P(\text{High Thorough|True Info})}{PP(\text{True Info}) \cdot P(\text{High Thorough|True Info}) + P(\text{High Thorough|False}) \cdot P(\text{False})} \quad (3.4)$$

The values from Fig. 3.14 help us solve this equation:

$$\frac{0.32 \cdot 0.51}{(0.32 \cdot 0.51) + (0.13 \cdot 0.49)} = 0.7193$$

As such, the conditional probability that an MTS had a high thoroughness score and was in the true information condition was 71.93 %, suggesting a significant relationship between having true information and better performance.

3.4.6 Re-run with Two Parent Nodes

Many social scientists and group researchers are interested in moderation or in other words, interaction effects. One way to get at this type of analysis is through increasing the parent nodes from one to two. This allows the predictor variables to interact with one another to create joint probabilities. Indeed, one of the reasons it is called Naïve Bayes is because the predictor variables operate independent from one another.

To increase the amount of parent nodes from one to two, simply click on the BayesNet classifier next to the “Choose” button in Weka to open the generic object editor (Fig. 3.15). Then click on the “searchAlgorithm” box next to the “Choose” button and increase the “maxNrOfParents” from one to two (Fig. 3.15). Finally, re-run the analysis (Table 3.4).

The table here is promising because the F-Measure has substantially increased from 0.656 to 0.702. Similarly, clicking on visualize graph will give us a probability distribution table (see Fig. 3.16). For instance, consider if we looked at efficacy and speed and wanted to determine is those groups who had high efficacy and average speeds:

$$\frac{0.51 \cdot 0.07 \cdot 0.838}{(0.51 \cdot 0.07 \cdot 0.838) + (0.49 \cdot 0.26 \cdot 0.091)} = 0.721$$

Here, we see that MTSs that had high efficacy and completed the mission in average times (i.e., not too long or short) had a 72.1 % chance of being in the true condition, demonstrating a significant relationship in how the manipulation may have influenced group performance.

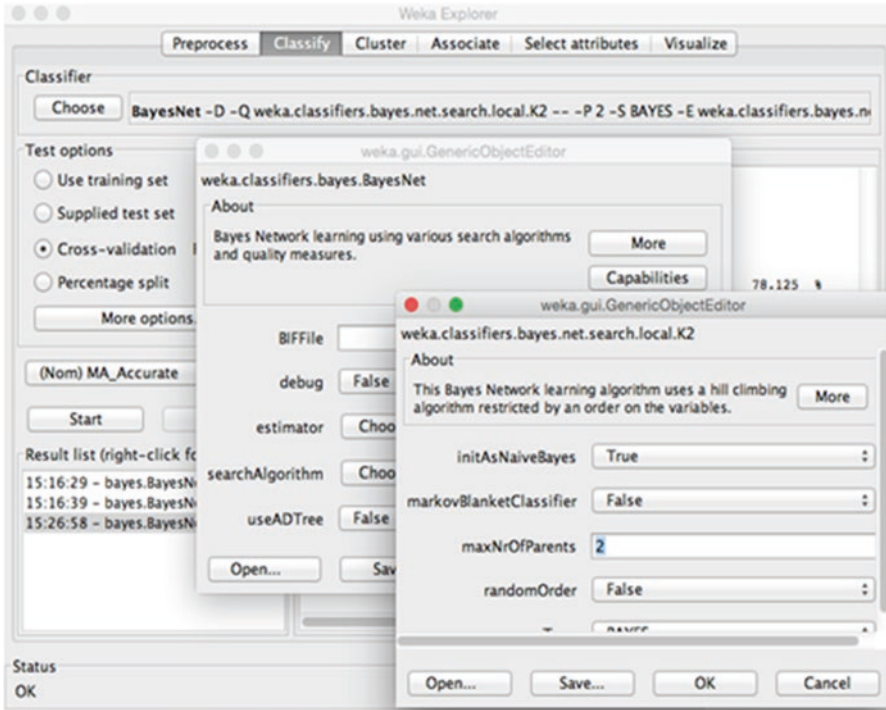


Fig. 3.15 Setting up a two-parent model

Table 3.4 Weka output for reduced model with two parent nodes

	Precision	Recall	F-Measure	ROC area	Class
	0.727	0.635	0.678	0.737	I
	0.685	0.769	0.725	0.737	A
Weighted Avg.	0.706	0.703	0.702	0.737	

3.5 Conclusion

This article demonstrated the opportunities offered by a data driven social scientific approach, popularly known as “computational social science”. Here we explored a situation where a number of people were working together in a complex environment. These people constituted true groups as they were interdependent with common goal and fate. Receiving accurate information was vital in their success. However, information accuracy was manipulated to see the effect on group processes. It was a simulation of a real world group oriented problem, and due to recent technological developments, the simulation was very close to real world actions.

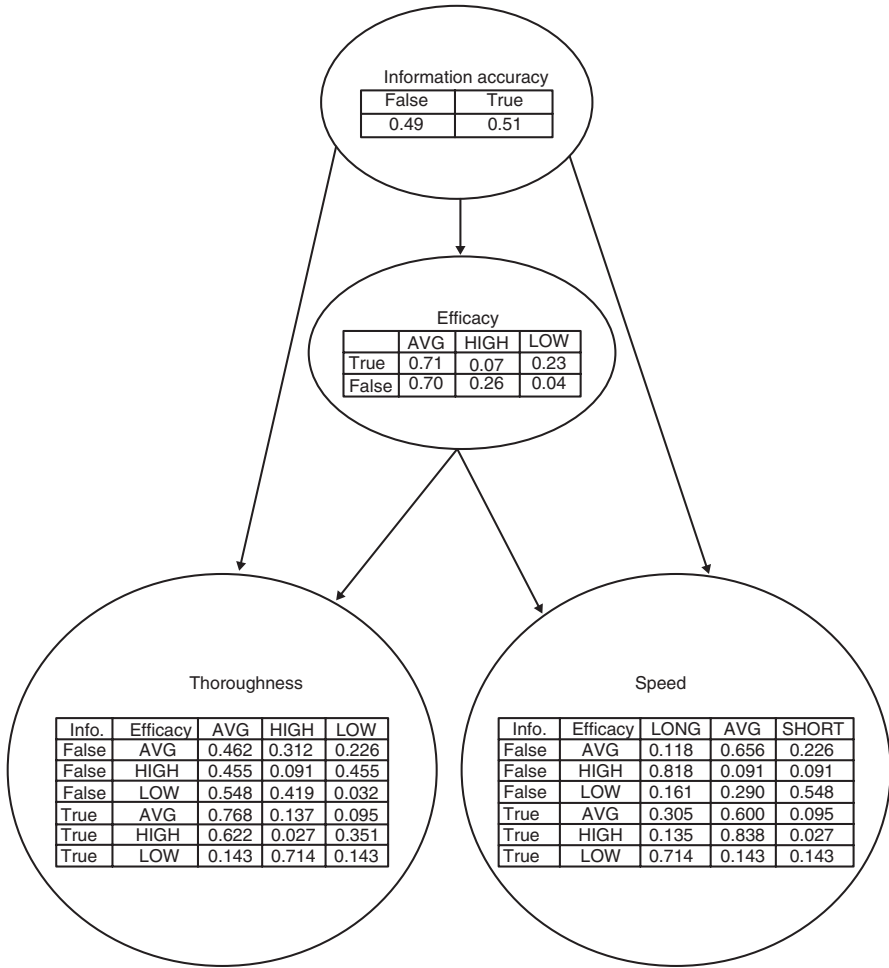


Fig. 3.16 Probability distribution table of two-parent model

Group communication scholars has been exploring and analyzing such situations for a long time. What made this situation unique is the number of variables that our system collected. We had 118 variables in the data set. What we observe here is the opportunity of collecting massive data. Previously, social scientists would limit the number of variables because of the complication that would arise in analysis and explanation. Data collection in those cases would be limited based on existing theories. Although theoretically sound, this line of research would be conservative in exploring many variables, limiting the possibility of discovering novel effects. Computational social science helps us to address this barrier.

Another possibility that comes forward is the opposite of theory driven analysis. Instead of an a-priori approach, now we can let the data show us relationships and

relational patterns and make sense of the relationship later based on existing theories. During the process, this article demonstrates that the MLCA analysis could actually discriminate variables based on their importance in understanding the situation.

This article also demonstrates new ways of interpreting and presenting social scientific results. Here we not only see that the conditional probability that an MTS had a high thoroughness score and was in the true information condition suggesting a relationship between having true information and better performance, we knew that the probability was 71.93 %. Such accuracy derived from complex situations could be considered as a major improvement in social scientific analysis.

This demonstration represents one of many novel possibilities offered by computational social science methods to social scientific scholars. Together with traditional approaches, new methods would definitely enhance our explorations and analysis of social situations. The significance of considering the approaches is even higher when we consider the nature of data sets with numerous associations and layers that we get from new and emerging media.

References

- Bouckaert, R. R. (2004). Bayesian networks in Weka. *Technical Report 14/2004*. Computer Science Department, University of Waikato, 1–43.
- Borsuk, M. E., Stow, C. A., & Reckhow, K. H. (2004). A Bayesian network of eutrophication models for synthesis, prediction, and uncertainty analysis. *Ecological Modelling*, 173(2), 219–239.
- King, G. (2011). Ensuring the data-rich future of the social sciences. *Science*, 331(6018), 719–721.
- Lazer, D., Pentland, A. S., Adamic, L., Aral, S., Barabasi, A. L., Brewer, D., ... Jebara, T. (2009). Life in the network: The coming age of computational social science. *Science*, 323(5915), 721.
- Marks, M. A., DeChurch, L. A., Mathieu, J. E., Panzer, F. J., & Alonso, A. (2005). Teamwork in multiteam systems. *Journal of Applied Psychology*, 90(5), 964.
- Pe'er, D. (2005). Bayesian network analysis of signaling networks: A primer. *Sci STKE*, 281, 1–12.
- Pilny, A., Yahja, A., Poole, M.S., & Dobosh, M. (2014). A dynamic social network experiment with multiteam systems. *Big Data and Cloud Computing, Proceedings of 2014 Social Computing*, 587–593.
- Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data mining: Practical machine learning tools and techniques* (3rd ed.,). San Francisco, CA: Morgan Kaufmann.