Bayesian Networks for Predictive Modeling

by Jeff Heaton

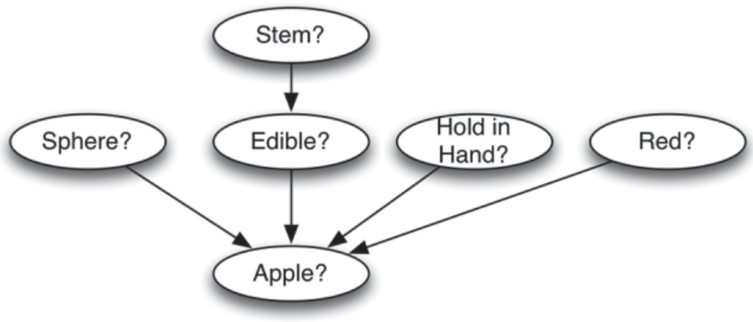
Bayesian Networks represent a convergence of Artificial Intelligence (AI) and Statistics. These networks function by creating a probabilistic model that can be used to query possible outcomes from input data. Bayesian Networks were created by Dr. Judea Pearl in 1983. Pearl is well known for championing the probabilistic movement in the field of Artificial Intelligence. Bayesian Networks can be used for predictive modeling, pattern recognition, classification and regression.

INTRODUCING BAYESIAN NETWORKS

Bayesian Networks model probabilistic relations among random variables. A random variable is a term from Statistics that means the items involved vary in some random, or unexplained manner. A Bayesian Network assigns probability factors to various results based upon an analysis of a set of input data. Like many other Machine Learning algorithms a Bayesian Network is taught using training data. Once trained, a Bayesian Network can be queried to make predictions about new data that was not represented by the training set. To see how a Bayesian Network functions consider if a trained Bayesian Network were presented with the following input data describing an object:

"I am holding an object that is red, editable, essentially spherical, fits in the palm of my hand and has a stem," you tell the network.

Figure 1: A Bayesian Network



The Bayesian network would report the probability that you are holding an apple. This probability is not 100 percent, since a cherry also fits this description. Figure 1 shows how this Bayesian network might be represented.

The above diagram indicates some relationships between the random variables. For example, the variable edible is dependent on stem. Many of the objects in our world that are edible have stems, so this makes sense. Additionally, the probability of the object being an apple is dependent on sphere, edible, handheld and red. The arrows in Figure 1 represent probabilities. The above network implies the following total probability.

```
Pr(Sphere) Pr(Edible|Stem) Pr(Hand)
Pr(Red) Pr(Apple| Sphere, Edible, Hand,
Red)
```

Because all of these variables are discrete, their probabilities can be represented by truth tables. Discrete random variables have a finite number of values. For this example each random variable has only two values—either true or false.

It is very common to represent random variables as real numbers. For numeric values there are two common methods for representation. First, real numbers can be banded into discrete ranges. Second, real numbers can be represented continuously using a probability distribution.

WHY USE A BAYESIAN NETWORK?

A Bayesian Network fills a role very similar to other Machine Learning algorithms such as an Artificial Neural Network (ANN), Decision Tree, or Support Vector Machine (SVM). However, a Bayesian Network has several unique advantages over some of the other Machine Learning algorithms.

First, a Bayesian Network handles missing values very well. Consider the previous example. You might only tell the Bayesian network that the object was red and had a stem. In this case the probability that the object is an apple would be lower. Not all Machine Learning algorithms handle missing data so well. Many Machine Learning algorithms require that incomplete data be eliminated or extrapolated.

Second, a Bayesian Network can be queried. An SVM or ANN is typically trained to predict a specific outcome given specific input. Following on the same example, you might tell the Bayesian network that you are holding an apple and that it has a stem. But this time, you query to see if the object is red. The random variables do not have fixed roles of input or output.

PREDICTIVE MODELING EXAMPLE

Now I will show you an example of how Bayesian networks can be used for predictive modeling. For this example, we will attempt to predict the direction that a particular exchange rate will move. To keep this example simple we will keep the amount of data relatively small. I will use the daily highs of the big six currency pairs. These pairs are given here.

- EUR/USD
- USD/JPY
- GBP/USD
- AUS/USD
- USD/CHF
- USD/CAD

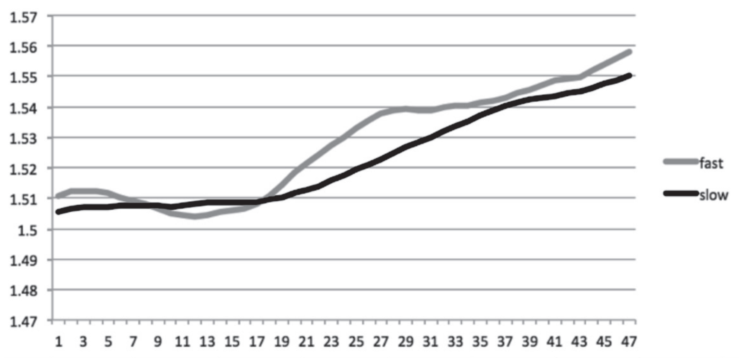
The data that I had available for these six currencies were minute bars since 2001. This is a fairly large dataset. Quite a few minutes have passed since 2001! Using SQL I decreased the granularity of the data down to just the daily highs for each of the six currencies. It is possible to deal with the individual minute bars. It is even possible to deal with the underlying ticks that made up the minute bars. Using such large amounts of data has the potential to yield better results. However, dealing with individual ticks over such a time period is a Big Data problem. Big Data often poses problems dealing with the immense size of the data set. Big Data is beyond the scope of this article, so I will stick with daily highs.

We are not going to present the data in its raw format to the Bayesian network. Like many Machine Learning problems the first step is to determine how to represent the data. Excel is a great program for preprocessing the data for the Bayesian network. There are only about 4,000 rows in my data set. Excel can easily handle this.

The goal of this Bayesian network is to use the current exchange rates of the big six exchanges and provide the probability that the EUR/USD pair will go up over the next 10 days. I would like the Bayesian network to use the rise and fall of the daily heights of all six. This considers that the various currencies may influence each other. I do not want the Bayesian network to consider the actual numbers, just the trends. Ideally the indicator of an upward trend will signal this regardless whether the exchange rate is 1.0 or 0.5.

To do this we will preprocess the data into two Simple Moving Averages (SMA). SMA is simply the average of a fixed number of days. We will make use of a fast (10-day) and slow (20-day) SMA. We are particularly interested in when these two SMAs cross. This is a well-known, and relatively

Figure 2: Fast and Slow SMA for EUR/USD



CONTINUED ON PAGE 9

simple, means of predicting trends. However, we are adding the extra complexity of using this data from six exchange rates. Figure 2 (pg. 7) shows these two SMAs.

As you can see from the chart above, the fast SMA was above the slow SMA for most of the chart. There were two crossovers.

The training data will consist of a total of 13 columns. The first column indicates the direction of the EUR/USD pair over the next day, either up or down. Each of the six currents also provides two columns. For each currency we track if the fast or slow SMA is greater. We also track if a crossover occurred on that day. You can see some of the training data in Figure 3.

Figure 3: Select Training Data for the Bayesian Network (select columns)

EURUSD Direction	EURUSD SMA	EURUSD Cross	USDCAD SMA	USDCAD Cross
Down	Slow	Cont	Slow	Cont
Down	Slow	Cont	Slow	Cont
Down	Slow	Cont	Slow	Cont
Up	Slow	Cont	Slow	Cont
Up	Fast	Cross	Slow	Cont
Up	Fast	Cont	Slow	Cont

For brevity only, the EURUSD and USDCAD exchange rates are shown in Figure 3. Likewise, only six days are shown. The training data shown above corresponds to just before the first crossover shown in Figure 2. This is all of the data provided to the Bayesian Network. No dates are provided, nor is the order of the rows. All short-range temporal position is encoded by the fact that we are encoding two averages and the crossover point.

TRAINING THE BAYESIAN NETWORK

Training is the process where training data is used to construct a Bayesian network. The Bayesian network is constructed so that the probabilities produced by the final network closely match the training data. It is very rare that you will get a Bayesian network that can exactly reproduce the training data. The success of the Bayesian network at producing results consistent with the training data is called the error rate of the Bayesian network. Training seeks to minimize this error.

Training a Bayesian network is typically divided into two distinct stages. The first stage creates the structure of the Bayesian network. The second stage creates the probabilities between the Bayesian network's nodes. The structure of the Bayesian network and the probabilities between nodes make up the entire memory of what the Bayesian network "knows."

There are several different techniques for creating the structure of the Bayesian network. Some of the most popular methods are listed here.

- Force Naïve Bayes
- Genetic Algorithm
- Hill Climbing Algorithm
- K2 Algorithm
- Simulated Annealing
- Tabu Search

The K2 algorithm is the most common, and is what I used to generate the Bayesian network for this article. Another common method is simply to force a Naïve Bayesian structure. The Naïve Bayesian structure is a very effective structure for Bayesian networks. Naive Bayes was the structure that K2 chose for this article's Bayesian network. You can see the structure of the Bayesian network in Figure 4.

In figure 4 you can see our 13 columns represented as random variables in a probability structure. A Naïve Bayes network contains only connections from leaf nodes to a central node. For Figure 4, the EURUSD next-day direction is dependent on the other six exchange rates slow and fast SMA position and crossover. There are no other dependencies. In reality there may well be such dependencies, however, the Bayesian network remains Naïve to them.

After the structure is in place, the probabilities between the variables must be determined. There are several different methods for performing this estimation.

- Bayes Model Averaging
- Multi-Nominal Bayes Model Averaging
- Simple Probability Estimation

For this article I used simple probability estimation. This allowed the Bayesian network to train to a theoretical accuracy of 62 percent. Of course this is only an example, and does not constitute investment advice. This is merely an example of how you might apply a Bayesian network for predictive modeling. This Bayesian network was modeled on the Encog Machine Learning Framework (http://www.encog.org).

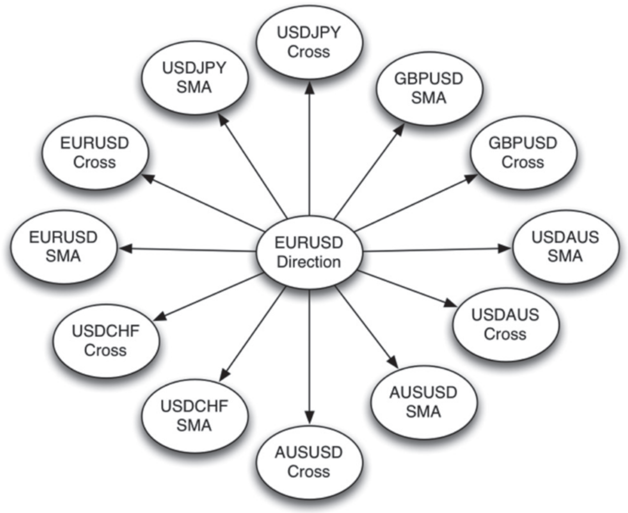
QUERYING THE BAYESIAN NETWORK

Once training is complete, the Bayesian network is ready to use. Using the Bayesian network typically involves querying the network. To query the network, you provide values for the known variables and the desired values for any output variables. You will be provided with the probability that the given input will produce the desired output.

Because we are dealing with a Naïve Bayesian network we can use the following relatively simple formula to perform the query.

$$\Pr(C|A_1, A_2 \dots A_n) = \frac{(\prod_{i=1}^{n} \Pr(A_i|C))\, P(C)}{P(A_1, A_2 \dots A_n)}$$

Figure 4: A Naïve Bayes Network for Currency Predictive Modeling



The above formula is a generalized form of Bayes Theorem. It shows how to calculate a case C based on several conditions A. For the currency example the case is the direction of EURUSD, and the conditions are the other 12 variables from Figure 4.

You can see a sample of this worked out in an Excel file (www.soa.org/BayesianNetworks/). Let's assume you enter the following values.

EURUSDSMA=slow, EURUSD_CR=cross,
USDCADSMA=fast, USDCAD_CR=cont,
USDCHFSMA=fast, USDCHF_CR=cont,
USDJPYSMA=fast, USDJPY_CR=cont,
AUDUSDSMA=slow, AUDUSD_CR=cont,
GPBUSDSMA=slow, GPBUSD_CR=cross

The Bayesian Network in this case predicts the EURUSD exchange rate direction as up with a probability of 63 per-

CONTINUED ON **PAGE 10**

cent. Once again, I am not making any guarantees here. Anyone who has spent hours shoveling a driveway full of snow after a "partly cloudy" weather forecast understands that probabilities are not certainties. Your model accuracy will depend upon the validity of your basic assumptions, the type and quantity of data you use for training, and external influences.

For a non-naïve Bayesian Network there are many different ways to implement querying of a Bayesian network. One of the most simple is called enumeration. Enumeration is a brute force iteration through all values of the truth tables. This process can be streamlined by various techniques.

CONCLUSIONS

For the purposes of illustration this article used a simple dataset for predictive modeling. For real-world use you would most likely do considerably more preprocessing of the data. Additionally, we only tested how well the Bayesian network performed on its training data. It is also important to use techniques such as cross validation to ensure the Bayesian network is not simply "memorizing" the training data.

The convergence of artificial intelligence and Statistics is behind some amazing advances in technology. Bayesian networks are just one of probability inspired Machine Learning algorithms. Hidden Markov Models, Particle Filters and Bayesian networks show that probability has a very important place in the field of AI.

REFERENCES

Artificial Intelligence: A Modern Approach (3rd Edition) by Peter Norvig & Stuart Russell, Prentice Hall, December 11, 2009

Data Mining: Practical Machine Learning Tools and Techniques, Third Edition by Ian H. Witten, et al, Morgan Kaufmann, January 20, 2011

The Theory That Would Not Die: How Bayes' Rule Cracked the Enigma Code, Hunted Down Russian Submarines, and Emerged Triumphant from Two Centuries of Controversy by Sharon Bertsch McGrayne, August 6, 2012

Illustration of the K2 Algorithm for Learning Bayes Net Structures by Carolina Ruiz, *http://web.cs.wpi.edu/~cs539/s05/Projects/k2_algorithm.pdf* ▼

Jeff Heaton

Jeff Heaton is EHR informatics scientist at RGA Reinsurance Company in Chesterfield, Mo. He can be reached at *jheaton@rgare.com.*