Universität Potsdam

Institut für Erd- und Umweltwissenschaften

Einführung in Bayessche Netze für Geowissenschaftler

Author:

Silvio Schwarz

Matrikelnr.:

743289

Email:

silvio.schwarz@uni-potsdam.de

June 2015

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The following assignment is an exam for the master course "MGEW23: Einführung in Bayessche Netze für Geowissenschaftler" in the scope of the master program "Geowissenschaften" at the Universität Potsdam.
The purpose of this work is to use Bayesian Networks in an example that resembles simplified questions one would encounter in the assessment of natural hazards. The task is to build ground motion models from synthetic data and quantify their ability to predict values of peak ground acceleration (PGA) given the input variables.
In the context of this paper four discrete Bayesian Networks have been learned. These include a causal network, a Naive Bayes network, a constraint-based network using a Grow-Shrink algorithm and a score-based one from a Hill-Climber method. Special care is given to the evaluation of the prediction performance. A mean squared error and a log-likelihood score are computed on a validation set in order to get an estimate of the out-of-sample performance. In a next step the concept of testing on unseen data is expanded to cross-validation and finally an attempt of a bias-variance decomposition is made to further develop this concept and to draw conclusions where potential for improvement lies.
The computations were performed using R (R Core Team, 2015) in combination with the IDE RStudio (RStudio Team, 2012). For setting up and working with Bayesian Networks the R package bnlearn (Scutari, 2010) was used.

DISCLAIMER:
There are some footnotes throughout this paper. These are personal comments and reflect more of an opinion or thought and are a bit more colloquial than the rest of this document. They are intended as discussion points.

Chapter 2

Learning Bayesian Networks

Bayesian networks are directed acyclic graphs (DAG). They are representations of the dependence structure among a set of random variables. The conditional dependencies between these random variables are visualized via directed edges and the random variables themselves are the nodes of the network. Through the directed edges it is possibly to identify a hierarchy. Nodes having edges leading to another node are called parents of the receiving node. Similarly, the receiving node is a child of the source node. Besides the dependence structure the nodes in a Bayesian Network carry a probability distribution conditional on their parents. In the case of discrete distributions this becomes a probability table.

The advantage of using a Bayesian Network is that it represents the joint probability over all the random variables considered. It is possible to answer conditional queries through the laws of probability theory. Through the structure of the network the joint probability distribution factorizes, leading to less parameters to be estimated and a more efficient computation.

For the task of the evaluation of natural hazard Bayesian Networks provide a consistent and efficient way to compute the full distribution of the target variable and thus to deal with the associated uncertainties.

2.1 Choosing the Structure of Bayesian Networks

The task for this assignment is to build different Bayesian Networks, use synthetic data to learn the corresponding parameters and evaluate their performance of predicting on the target value.
The data consists of a set of six variables commonly employed for defining ground motion models and the prediction of ground motion values. Each of the variables has been sampled from a distribution according to table 1 and the stochastic model of Boore (2003) was used to compute the corresponding values of PGA. In total the data set comprises 10.000 "observations". Since there is no simple analytic form of the stochastic model and it involves dealing with distributions from different families, the data has been discretized into intervals according to table 2 and discrete Bayesian Networks are learned using multinomial distributions throughout.

TABLE 1: Overview over the variables used and their according distributions.

X_i	Description	Distribution$_{[range]}$
	Variables	
M	Moment Magnitude	$\mathcal{U}_{[5,7.5]}$
R	Distance to Source	$\text{Exp}_{[1km,200km]}$
SD	Stress drop	$\text{Exp}_{[0bar,500bar]}$
Q_0	Attenuation of seismic waves in deep strata	$\text{Exp}_{[0s^{-1},5000s^{-1}]}$
κ_0	Attenuation of seismic waves close to the surface	$\text{Exp}_{[0s,0.1s]}$
V_s30	Average shear wave velocity in the upper 30m	$\mathcal{U}_{[600ms^{-1},2800ms^{-1}]}$
	Ground Motion Variable	
$\log PGA$	logarithm of peak horizontal ground acceleration	synthetic calculated through the stochastic model of Boore (Boore, 2003)

Table 2: Discretization of the Variables. The intervals have been chosen in order to minimize information loss.

Variable	Interval Borders								
SD	0	0.8792	5.438	14.92	58	500			
Q0	0	330	5000						
κ_0	0	0.01053	0.0345	0.1					
V_S30	600	1704.5	2800						
M	5	6.271	7.5						
R	1	4.38	15.4885	55.84	200				
log PGA	-Inf	-5.135	-3.722	-2.627	-1.20742	0.145	1.657	3.175	Inf

2.1.1 Naive Bayes Network

There are different methods for choosing the structure of a Bayesian Network. The easiest and simplest way is to construct a Naive Bayes network. This means that the target value is connected to all explanatory values and there are no other dependencies. This is "naive" as it makes the assumption that all explanatory variables are independent from each other. Thus, the joint distribution factorizes simply to the following product:

$$\text{P(PGA, SD, MAG, DIST, } Q_0, \kappa_0, V_s30) = \text{P(PGA)* P(SD|PGA)*}$$
$$\text{P(MAG|PGA)*P(DIST|PGA)* P(}Q_0\text{ |PGA)* P(}\kappa_0\text{ |PGA)* P(}V_s30\text{ |PGA)}$$

Considering the Naive Bayes independence assumption can be thought of the simplest model that can be created by using all variables. Neglecting all dependencies is a strong assumption which might result in an unrealistic model. Nevertheless, it is a attractive choice because it needs very few parameters to be estimated and it shows a reasonable performance in real-world applications such as email spam filter. It is also a good candidate as a starting point because it is questionable to build more complex models which perform worse. The network in figure 1 represents the Naive Bayes network for the given variables and was used in the further computations.

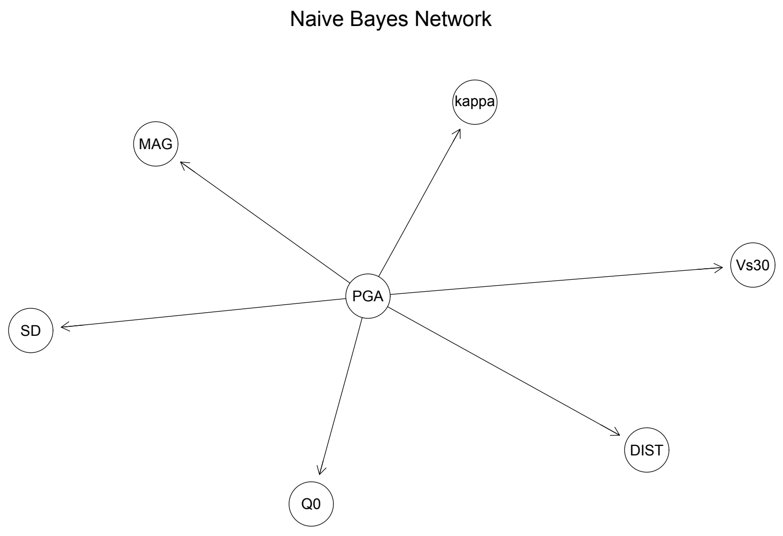


FIGURE 1: The Naive Bayes network of the variables.

2.1.2 Causal Network

Another way of setting up the structure of a Bayesian Network is to rely on expert judgment to define the dependencies between the variables. This is called a causal network since one tries to capture the causal relationships among the variables. For the case of predicting PGA from a set of explanatory variables the following causal network (Figure 2) can be reasoned.

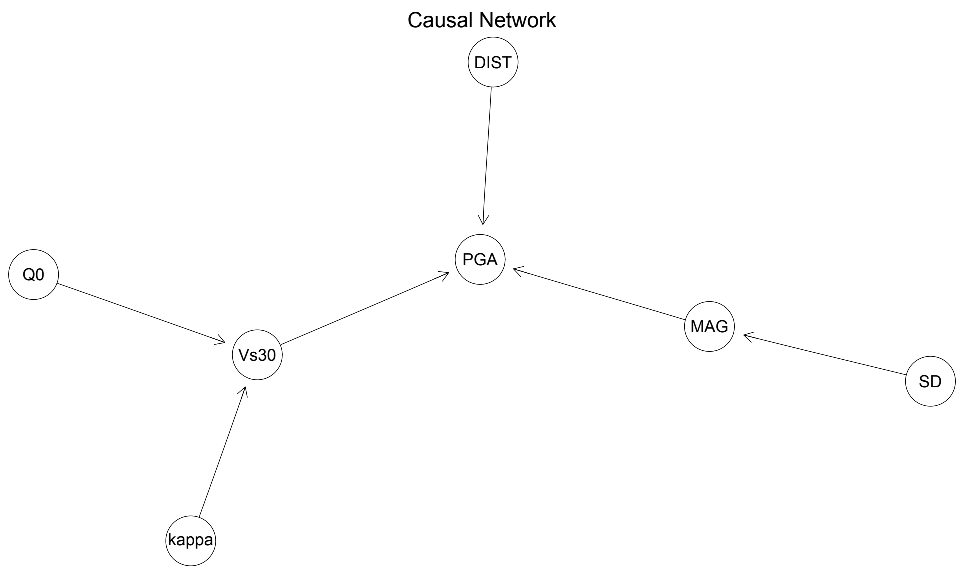


FIGURE 2: A causal network representing the beliefs in dependency based on expert knowledge.

One can argue that the attenuation behavior in deep layer (Q0) is independent from the one in shallow layers (kappa) because this difference reflects the varying materials and environment conditions. Nevertheless, since both are material properties, as well as the shear wave velocity in the first 30m (V_s30), one can imagine that their influence is mitigated by the shear wave velocity in the first 30m. The distance from the source (DIST) does not seem dependent on any other explanatory variable because one can argue having the same earthquake but choosing a different location on the earth's surface freely. Moment magnitude (MAG) and stress drop (SD) are dependent since they both refer to the energy that is released during an earthquake. Since the moment magnitude is proportional to the ruptured area the same value can be achieved by a wide range of possible combinations in the rupture's width and length. This is the reason why it is dependent on the stress drop, meaning one value of stress drop can have a distribution of moment magnitude values associated but not the other way around. [1]
Another way to set up causal networks is to consult literature about the topic. In the case of Bayesian Networks for ground motion prediction sources could be Kuehn (2010) or Vogel (2014).

2.1.3 The constraint-based Grow-Shrink Algorithm

A third way of defining the structure of a Bayesian Network is to learn it from the data itself. Often, not all dependencies between the variables are known and human domain knowledge can also be misleading. So it is a natural extension to ask whether there are principled ways in the framework of Bayesian Networks to let also the structure come from the data. This can be done by extending the Bayesian paradigm so that the structure of a network becomes a random variable too. Now the task is to jointly estimate the parameters and the structure from the data. In the scope of this paper the constraint-based Grow-Shrink algorithm (Margaritis, 2003) and the score-based Hill-Climber, using the Bayesian Information Criterion (Schwarz et al., 1978) as score, are discussed.

Constraint-based algorithms perform independence tests between the random variables and then set up a network according to the found independencies. The task is one of finding the best minimal I-map. An I-map or independence map is a graph whose

[1] After setting up the nets and running the computations, which took almost 2 weeks, I realized that this is not the best causal network I could design through my knowledge of the earthquake process. Q0 and kappa are attenuation parameters and only influence the amplitude of the seismic waves. It is questionable how this can have an influence on the seismic wave velocity. There also might be a dependency between the distance and the magnitude since larger earthquakes appear in greater depths and the distance employed in the Boore model is hypocentral distance and not epicentral. So this is more an example of a poorly designed causal net but changing this would have resulted in a re-computation.

independence statements hold for the probability distribution among the variables. In the case where the graph captures all independence statements this is a perfect I-map. A minimal I-map is graph that is rendered not an I-map anymore by the removal of one edge. This is an important definition because the complete graph over a set of random variables is also an I-map but does not reveal any independencies and therefore carries parameters that are redundant. In practice not one single best minimal I-map is found but a class of graphs that carry the same independence statements and are therefore called I-equivalent (Koller and Friedman, 2009). The Grow-Shrink algorithm tries to construct the structure of a network by finding the Markov Blankets of the variables. A Markov Blanket of one variable is a set of variables that renders that variable to be d-separated from all other variables. That means that knowing the state of any variable not in the Markov Blanket has no effect on knowing the state of the variable in interest. One could say that the Markov Blanket "shields" a variable from the influence of all other variables. Graphically, it is the set of parents, children and parents of the children of the variable in interest (Koller and Friedman, 2009). In the growing phase of the Grow-Shrink algorithm independence tests between variables are performed which are the basis to decide whether a variable should be included in the Markov Blanket. These tests occur given the state of the Markov Blanket. Depending on the initial ordering of the variables this can lead to including redundant variables in the Markov Blanket which are subsequently removed by the independence test of the shrinking phase (Margaritis, 2003). In this assignment the mutual information (Equation 2.1) is used as independence test.

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)\,p(y)} \right), \tag{2.1}$$

It estimates the dependence between two variables by comparing the joint distribution to the product of the marginal distributions, since in the case of independence the joint distribution factorizes to the product of the marginal distributions. From a Venn-diagram point of view it calculates the area shared by two variables relative to the total area of the variables.

The learned network is visualized in Fig.3. There are no direct dependencies between the explanatory variables. Even more, the variable V_s30 is completely ignored. By comparing this result to work of (Vogel, 2014),which uses a similar data set[2], one can find that this seems to be a consistent result when learning the structure of a Bayesian Network for ground motion prediction from data. One reason might be that V_s30 is merely a proxy in quantifying the capability of the soil to amplify the amplitudes of seismic waves.

[2] ;)

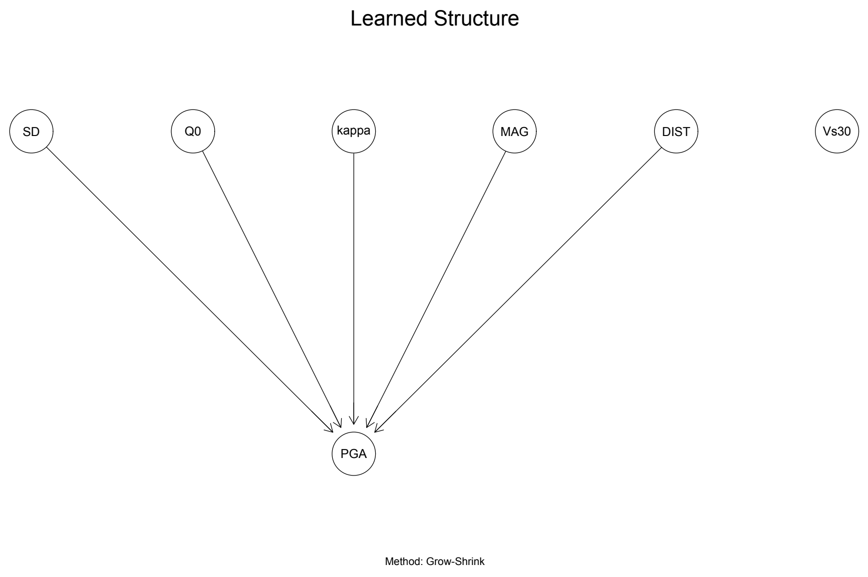


FIGURE 3: Bayesian Network constructed by the Grow-Shrink algorithm.

2.1.4 The score-based Hill-Climber Algorithm

Score-based algorithms view the problem of finding the structure of a Bayesian Network from an optimization point of view. In contrast to the constraint-based algorithms, score-based ones do not try to construct the structure from information about single dependencies between variables, but take the network as a whole, compute a score and try to find the network that maximizes that score. Consequently, score-based algorithms pose a search problem in the space of possible network structures. Depending on the number of variables and the underlying probability distributions this is a NP-hard problem in most cases and requires some approximation techniques (Koller and Friedman, 2009).
A Hill-Climber can be thought as the opposite of gradient descent since it tries to maximize a predefined score in contrast to minimizing an error term. For the construction of a Bayesian Network using the Hill-Climber algorithm a score consisting of the maximized likelihood L (Equation 2.2), that gives the probability of the data being generated by the graph G and the parameters θ, and the Bayesian Information Criterion (BIC) (Equation 2.3 (Schwarz et al., 1978)) as a regularization term, consisting of the number of free parameters k and the size of the data n, is used.

$$L = \arg\max_{\theta} P(x \mid \theta, G) \tag{2.2}$$

$$BIC = -2 * lnL + k * ln(n) \tag{2.3}$$

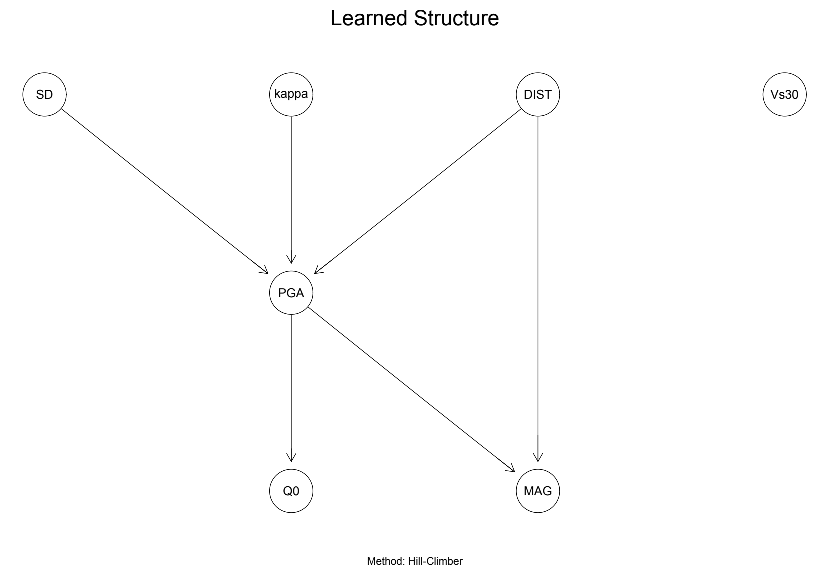


FIGURE 4: Bayesian Network constructed by the Hill-Climber algorithm.

The learned network from the Hill-Climber algorithm looks similar to the one learned by the Grow-Shrink algorithm. The only difference is a dependency between distance and moment magnitude. Since in the scope of this paper cross-validation is discussed, it should be noted that actually a series of networks was learned (Appendices A.1 and A.2). Each of these networks is learned from a different set of data and besides the first fold of the Hill-Climber method the results for each method are consistent. This means that the structure of the learned network is not dependent on which data points are used to come up with the structure and it is a result that is expected by using synthetic data and learning the structure by random permutations of the data.

2.2 Parameter Learning

There are different ways to learn the associated parameters and therefore the probability distributions of a Bayesian Network. As noted before, because the model of Boore (2003) has no simple analytic form and involves distributions from different families, the data has been discretized into intervals and discrete Bayesian Networks are learned using multinomial distributions.

In this assignment the parameters of the different networks have been learned using Bayes' Law. This is also called a maximum aposterior estimator (MAP) since the prior belief in the parameters are updated through the likelihood of the data been generated by a model with this parameters. With the posterior probability of the parameters being:

$$P(\theta \mid x) = \frac{P(x \mid \theta)P(\theta)}{P(x)}$$

the maximum aposteriori estimates of the parameters:

$$\theta_{MAP} = \arg\max_{\theta}\ P(\theta \mid x)$$

Here, the structure of the Bayesian Network comes in handy because it simplifies the computations by making use of the independencies between variables. For each node in the network a probability table is calculated which depends only on the parent nodes and not on the full set of all variables. All the computations are performed with the bnlearn package for R (Scutari, 2010).

Chapter 3

Testing Bayesian Networks

Testing the outcomes of a model is an integral part of the modeling since it gives an estimate on the expected performance in future scenarios. It is also a requirement for scientific work to provide some sense how well the reality has been captured and to provide a way to falsify one's findings.
For doing so, many different metrics and methodologies have been developed, all targeting a different part of the question what a "good" model should be. There are measures like the mean squared error that quantify the difference between the observed data and the values predicted from the model. These can be thought of as an in-sample metric because the data to construct the model is also used to estimate the error. Intuitively, this seems appropriate because it gives an easy to use measure that establishes a link between the model and reality. But focusing too much on getting a low in-sample-error can cause a phenomenon called "overfitting", where the model is so much adjusted to the data at hand, it fails to do well on unseen data. This can even lead to the conviction that some of the data has to be excluded because the fit is worsened.
Here lies the difference between function approximation and learning a model. In function approximation the goal is to estimate the parameters of a known model so that the final function matches the given data the closest. In learning a model the underlying relationship is usually not or not fully known and the data only represent a subset of the whole range of possible values, often including some noise. Hence, the task in learning a model is to match the model complexity to the data resources (Abu-Mostafa et al., 2012). Particular in a scenario of estimating natural hazard it would be desirable to have a measure that can quantify the model's ability for forecasting[1].

[1]Here I switch from predicting to forecasting because predicting seems to have more of a meaning of a wise oracle that knows something for sure in mysterious ways inaccessible to mortal beings. Forecasting, like weather forecasting, captures more the notion of dealing with uncertainty and deciding under uncertainty

In the scope of this paper the mode, median and mean are used as point estimates and consequently the mean squared error is computed. Additionally, the log-likelihood given by the model to the observed PGA values is used. These metrics are employed into different methodologies starting from a simple validation case over to cross-validation and to an attempt of a bias-variance decomposition. This captures a process from getting a simple handle on the out-of-sample performance over to more sophisticated methods that reveal insights into the prediction ability of the model.

3.1 Validation

One common approach to get an estimate of a models' prediction performance is to divide the data into a learning set and a validation set. Then the learning data is used to estimate the parameters and the performance is tested on the validation set. This has the effect of simulating unseen data since the validation data has not been used for learning the models parameters.
The data has been randomly divided into learning and validation data with 90% of the data going into the learning set and 10% are kept for validation. Table 3 shows the results of this analysis.

Table 3: Errors on the validation set calculated for the different nets and error measures. Mean, median and mode are the mean squared errors between the mean, median and mode of the resulting PGA distribution and the "observed values" in the data set. Probability is the sum over the logarithm of the probability that was predicted for the "observed" values of PGA in the validation data.

	error metric			
net	mean	median	mode	probability
Causal	4.816	4.933	9.739	-1.772
Naive Bayes	1.488	1.801	1.937	-1.35
Hill-Climber	1.068	1.339	1.503	-1.016
Grow-Shrink	0.844	1.124	1.149	-0.874

For all networks, using the mean as a point estimator yields the lowest errors. This might seem counter-intuitive because the mode of a distribution is by definition the value with the highest probability. But over the long run the mean is the value that establishes the best balance between values with high probabilities and those in the tails of the distribution. Taking a game of gambling as analogy, the mean is the "fair price" with which one would neither lose nor win any money in the long run.
Considering the different ways the structures of the networks have been chosen the causal net show the worst performance which might have to do with the fact that it was not properly thought through. The Naive Bayes network has quite a good performance given the fact that it neglects any dependency between the explanatory variables. The best

predictions come from the networks which have learned their structure from the data. One might be a bit puzzled because these networks do not seem to show much of the dependencies one would think of considering the earthquake process but, again, the task is not to come up with a model that causally explains how the different variables interact but to find a model that, given the data resources, best would have generated the data. These two approaches, modeling nature by observing cause and effect and focusing on a good prediction performance, should complement and at some point come very close to each other. But they do not have to be the same and it is a bit of a religious argument what ”real” science ought to be.

3.2 Cross-validation

Dividing the available data into a learning and a validation set makes the compromise to spend a certain portion of the data for validation and to not use this data in the estimation of the model's parameters. One has to find a balance between not setting aside too much data for the validation and consequently learning a poor model from fewer data points and not using enough data for validation so that the estimated future performance is not well defined. The method of cross-validation addresses this dilemma by not learning a single model and testing on one validation set but performing this procedure several times and average the results. Therefore, the data is divided into so called folds. In the course of this assignment the data has been randomly divided into ten folds which are subsets of the data that together span the whole range of the data. Consequently, nine of these ten folds have been used to learn the model and one was left for validation. Since there are ten folds this procedure can be repeated nine times, each time with another fold for validation and the rest for learning. This means that for each network type ten different networks have been learned and tested on a different data set. One can think of cross-validation similar to bootstrapping but without using a data point over again. This is the same as the procedure in the Validation section because any other combination of learning and validation data set would have been possible. So without changing anything, ten times the amount of models have been learned and validated. For the Grow-Shrink and Hill-Climber networks this means learning ten structure which possibly could be different (see Appendices A.1 and A.2).
The resulting errors are then the averages of the mean squared errors or log-likelihood for each network type. This is like learning from ten times more data with the restriction that the data is not really independent from each other because the learning data sets overlap. Cross-validation originally comes from model selection where one trains models and varies one parameter in order to determine the best choice for that parameter.

TABLE 4: Average errors on the folds using the cross-validation method. Errors are calculated for the different nets and error measures. Mean, median and mode are the mean squared errors between the mean, median and mode of the resulting PGA distribution and the "observed values" in the data set. Probability is the sum over the logarithm of the probability that was predicted for the "observed" values of PGA in the validation data.

	error metric			
net	mean	median	mode	probability
Causal	4.904	5.066	9.421	-1.764
Naive	1.482	1.703	1.762	-1.342
Hill-Climber	1.041	1.488	1.503	-1.017
Grow-Shrink	0.823	1.202	1.149	-0.875

The results in Table 4 confirm the findings of the validation section but now the confidence in these findings can be higher since they are averaged over a larger number of data. There is another advantage to cross-validation. Now that a good and reliable out-of-sample estimate is calculated one can go back and learn the final model on the whole data set. This way the error estimates become a kind of upper limit and one gains the advantage of learning on the whole data (Abu-Mostafa et al., 2012).[2]

3.3 Bias-Variance Decomposition

Until now, two methods for estimating the prediction performance have been given. Both start from the conviction that what really matters is the out-of-sample performance, with cross-validation having the advantage of getting a reliable estimate by averaging the validation error of a set of models and nevertheless being able to learn the model on the entire data set.
Now it would be interesting, given it is known that there are errors associated with the models, to have some guidance on how to improve the models. For this reason the bias-variance decomposition is introduced. It shares some similarities to cross-validation since both rely on learning a multitude of model. First, the bias-variance decomposition takes a step back to the in-sample mean squared error:

$$\mathbb{E}_{out}(g^{(D)}) = \mathbb{E}_x[(g^{(D)}(x) - f(x))^2])$$

where the expected out-of sample error $\mathbb{E}_{out}(g^{(D)})$ is the expected value with respect to the probability distribution of the input features x of the difference between the

[2]Another way of determining the reliability of the learned networks would be to analyze how strong they differ from each other. Good models should have a low diversity between each other although learned from different data sets. This idea is explored in the next section "Bias and Variance Decomposition".

hypothesis $g^{(D)}(x)$ and the target function f(x). The dependence of g on a certain data set D is made explicit. This can be eliminated by taking the expectation with respect to all data sets:

$$\begin{aligned}\mathbb{E}_D[\mathbb{E}_{out}(g^{(D)})] &= \mathbb{E}_D[\mathbb{E}_x[(g^{(D)}(x) - f(x))^2]] \\ &= \mathbb{E}_x[\mathbb{E}_D[(g^{(D)}(x) - f(x))^2]] \\ &= \mathbb{E}_x[\mathbb{E}_D[(g^{(D)}(x)^2] - 2\mathbb{E}_D[g^{(D)}(x)]f(x) + f(x)^2]\end{aligned}$$

The term $\mathbb{E}_D[g^{(D)}(x)]$ is a kind of average function which can be seen as generating different hypotheses from different data sets and averaging across these. By denoting the average function with $\bar{g}(x)$:

$$\begin{aligned}\mathbb{E}_D[\mathbb{E}_{out}(g^{(D)})] &= \mathbb{E}_x[\mathbb{E}_D[(g^{(D)}(x)^2] - 2\bar{g}(x)f(x) + f(x)^2] \\ &= \mathbb{E}_x[\mathbb{E}_D[(g^{(D)}(x)^2] - \bar{g}(x)^2 + \bar{g}(x)^2 - 2\bar{g}(x)f(x) + f(x)^2] \\ &= \mathbb{E}_x[\mathbb{E}_D[(g^{(D)}(x) - \bar{g}(x))^2 + (\bar{g}(x) - f(x))^2]\end{aligned}$$

which can be divided into $bias(x) = (\bar{g}(x) - f(x))^2$ and $var(x) = \mathbb{E}_D[(g^{(D)}(x) - \bar{g}(x))^2$ so that:

$$\begin{aligned}\mathbb{E}_D[\mathbb{E}_{out}(g^{(D)})] &= \mathbb{E}_x[bias(x) + var(x)] \\ &= bias + var\end{aligned}$$

The bias can be thought of how close the average function $\bar{g}(x)$ is to the target function f(x). The variance describes how much the models from different data sets vary with respect to each other.

Through this decomposition of the mean squared error it is possible to identify potential for improvement. Consider the case where the bias is large. This means, even learning from models of different data sets, the average of these models is still far from the target function. The model is too simple and would benefit from some additional features. This corresponds to the case of underfitting. If the variance is large that suggests that models from different sub-samples of the data lead to very different predictions. This is a case of overfitting (Abu-Mostafa et al., 2012). Diagnosing bias and variance can also be done by comparing the training or in-sample error with a cross-validation error.

A setting of high bias would correspond to high values in both the training and cross-validation error because the hypothesis is to simple to catch the underlying relationships and consequently underfitting. A high variance scenario is characterized by a low training error and a high cross-validation error, showing that the hypothesis has overfit the target function.
It should be noted that this definition of the bias-variance decomposition assumes no noise in the data. This can contribute to the variance. Besides, the errors are calculate using the middle of the discrete intervals for PGA, but the differences are calculated with the continuous data. Some of the errors account for the fact that discrete Bayesian Networks where used and not continuous ones. For calculating the bias and variance the networks learned by cross-validation were used. It might be good to have a look how these estimates change using more than ten different data sets. This would be very time consuming considering the duration of the computations.

Table 5 and 6 show the results for the bias and variance, respectively. The vast majority of the error is made up by the bias. This suggests that the learning algorithm suffers from underfitting. One reason could be that, although using a Bayesian approach to ground motion modeling, no prior knowledge about the parameters of the variables was specified[3]. This can suggest that the prior used was not very informative and acted as a strong regularization term.

Table 5: Estimates of the Bias for different error metrics and different networks.

	error metric		
net	mean	median	mode
Causal	4.876	5.007	9.233
Naive Bayes	1.472	1.682	1.709
Hill-Climber	1.029	1.341	1.478
Grow-Shrink	0.786	1.115	1.152

Table 6: Estimates of the Variance for different error metrics and different networks.

	error metric		
net	mean	median	mode
Causal	0.01	0.185	1.661
Naive Bayes	0.0091	0.136	0.546
Hill-Climber	0.0032	0.0694	0.128
Grow-Shrink	0.003	0.0622	0.092

[3]I haven't found any hint how the bnlearn package treats parameter estimation through maximum aposteriori probabilities. I am just assuming that a Dirichlet distribution with equal probability for different values of the parameters is used as prior. Essentially a uniform distribution. This would fit the observation that the data was underfit.

Chapter 4

Conclusions

Four different Bayesian Networks have been learned from synthetic data in an attempt to answer questions related to natural hazard assessment. These computations where complemented by using different metrics to derive point estimates from a distribution and different methodologies to estimate the out-of-sample performance of the models.

The best results throughout all tests where accomplished by networks which learned their structure from the data. They do not include V_S30 which is surprising but consistent with findings by Vogel (2014).
It was shown that the mean is a better point estimator than the mode or median of a distribution, at least when considering a larger number of predictions.
The complexity of the causal network seems unjustified, considering the prediction performance, although the cause may lie more on the side of a bad reasoning process.
For a scenario where quick decisions have to be made by relying on the outcomes of a Bayesian Network the Naive Bayes network has the advantage of a very low computation duration[1] and it has the benefit of not needing any expert judgment[2] while still performing quite well.
It also should be noted that some of the errors are due to the discretization and not using continuous Bayesian Networks. It would be interesting performing similar computations with different degrees of interval size to see what influence this feature has.
Another desirable feature for the future would be some kind of absolute error metric. Right now, it is only possible to compare models to each other but the number alone does not tell so much.

[1]Although I am a bit shocked how long it took for the computations to be done. Right now, I can't imagine this being used in the framework of an early warning system, for example. And I am sure Carsten showed me something that was much more responsive concerning Tsunami.

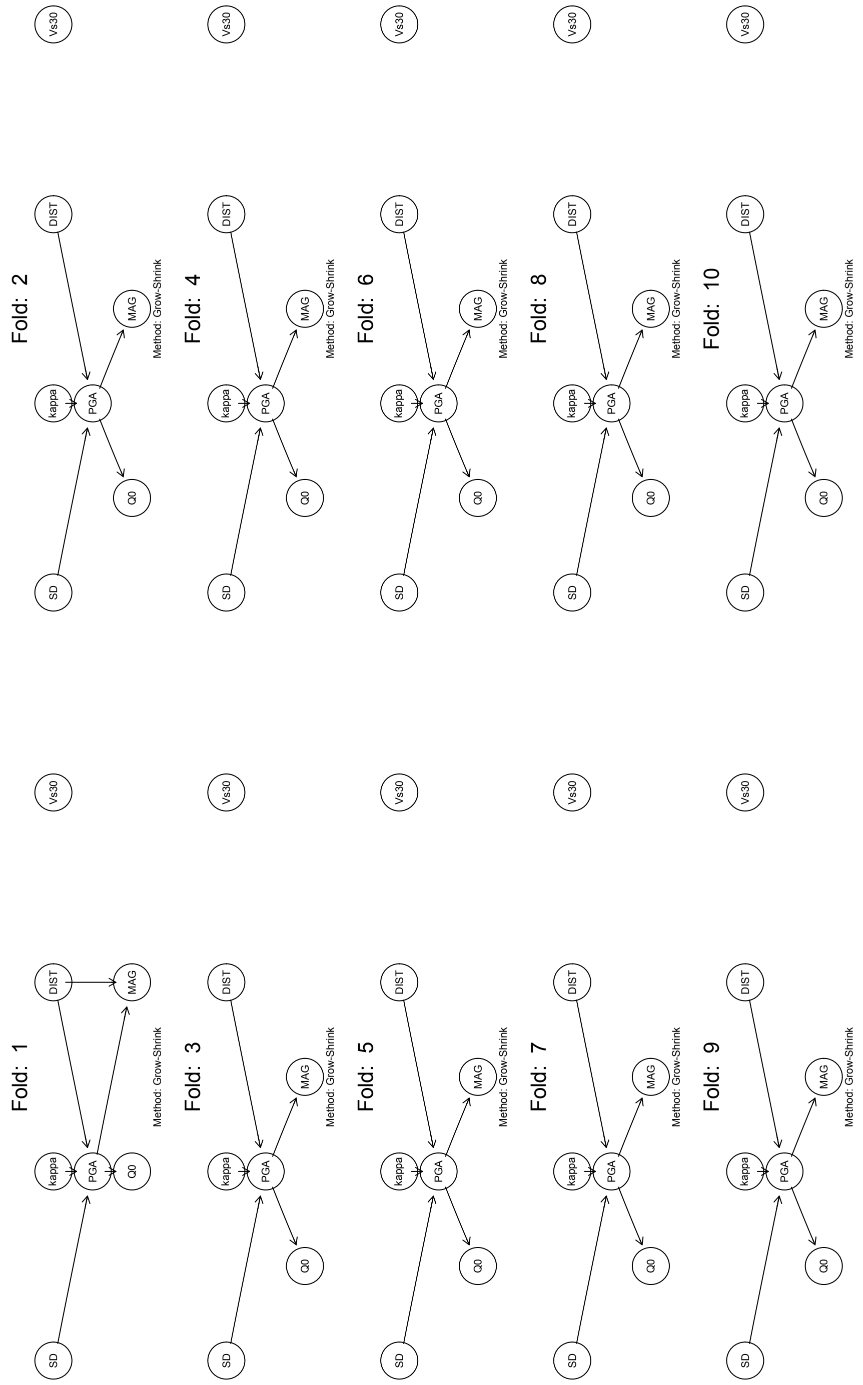
[2]and years of committees and meetings. But actually that's not really encouraging from a Bayesian point of view. Assuming everything is independent and not using expert knowledge...

DISCLAIMER:
All the code, graphics, images, this document and most of the used references can be found in my personal repository under:
https://github.com/silvioschwarz/bayes-nets.
It is not as clean and self explaining as some repos of big engineering companies but I think it will do the trick

Appendix A

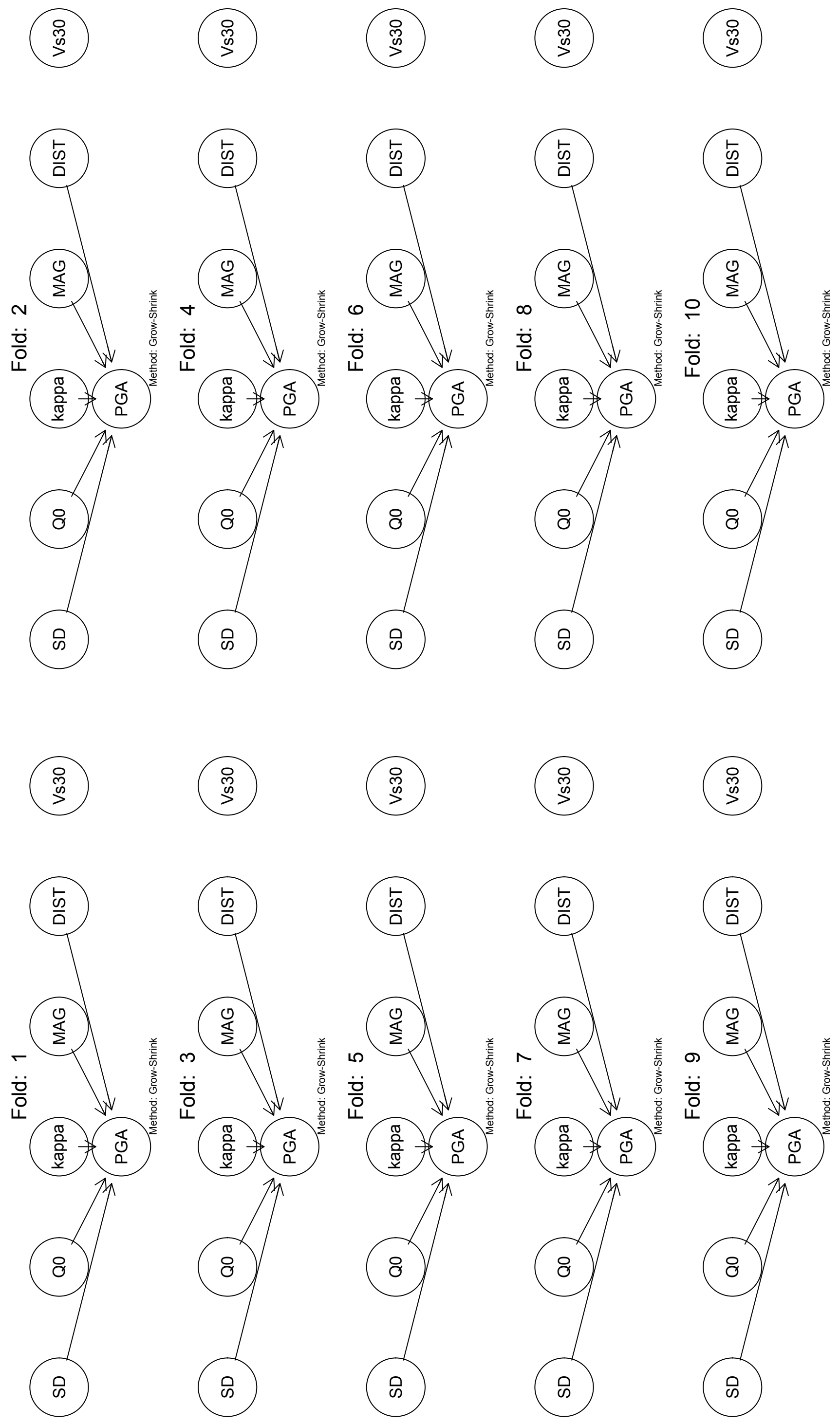
Additional Figures

A.1 Folds of Learned Networks: Hill-Climber



Folds of Learned Networks: Hill-Climber

A.2 Folds of Learned Networks: Grow-Shrink



Folds of Learned Networks: Grow-Shrink

References

Abu-Mostafa, Y. S., Magdon-Ismail, M., and Lin, H.-T. (2012). *Learning From Data*. AMLBook.

Boore, D. M. (2003). Simulation of ground motion using the stochastic method. *Pure and applied geophysics*, 160(3-4):635–676.

Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

Kuehn, N. (2010). *Empirical Ground-motion Models for Probabilistic Seismic Hazard Analysis: A Graphical Model Perspective*. PhD thesis, Universität Potsdam.

Margaritis, D. (2003). *Learning Bayesian network model structure from data*. PhD thesis, University of Pittsburgh.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

RStudio Team (2012). *Rstudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464.

Scutari, M. (2010). Learning bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35(3):1–22.

Vogel, K. (2014). *Applications of Bayesian networks in natural hazard assessments*. PhD thesis, Universität Potsdam.