JOHNS HOPKINS
UNIVERSITY

Course Project Report

Bayesian Network Classifiers

Azita Dadresan

Reasoning Under Uncertainty 605.745

December 5, 2022

Abstract

Bayesian networks can be used as machine learning classifiers. A popular example is the Naive Bayes classifier. Given the target class label, the features are assumed to be conditionally independent in a Naive Bayes classifier. For several classification problems, Naive Bayes has shown competitive performance with more advanced machine learning classifiers such as decision trees and Support Vector Machines (SVM). The Tree Augmented Network (TAN) is an extension of the Naive Bayes classifier to incorporate correlations between features. This project studies the Tree Augmented Networks and implements the Chow-Liu algorithm for training and testing bayesian classifiers. The method is illustrated on a synthetic toy dataset (the Lung Cancer dataset). Further experiments are conducted on publicly available datasets from the UCI machine learning repository. Experiments show that the Chow-Liu algorithm outperforms the Naive Bayes algorithm significantly and is at par with or better than decision tree and SVM classifiers.

Contents

1 Introduction

Bayesian networks are directed graphical models used for probabilistic reasoning, such as causal reasoning and evidential reasoning. They can be directly used for classification problems by setting the variable to be inferred as the target/class variable. A major challenge of using bayesian networks is that they can quickly become intractable as the number of features increases. The Naive Bayes classifier is a simple bayesian network that has a strong conditional independence assumption.

Naive Bayes classifiers are simple, yet powerful classification models. It makes the assumption that the features are conditionally independent given the class label. Figure 1 shows the graphical model of an example Naive Bayes classifier. The class or target variable is pneumonia which can either be positive or negative. The features are the results from X-ray results, the presence of fever in the patient, and whether the patient is lethargic or not. A Naive Bayes classifier assumes that all these features are independent given the class label (if the information that pneumonia is positive or negative is known for a patient). This is clearly a very simplistic assumption as we know the features could be correlated with each other.

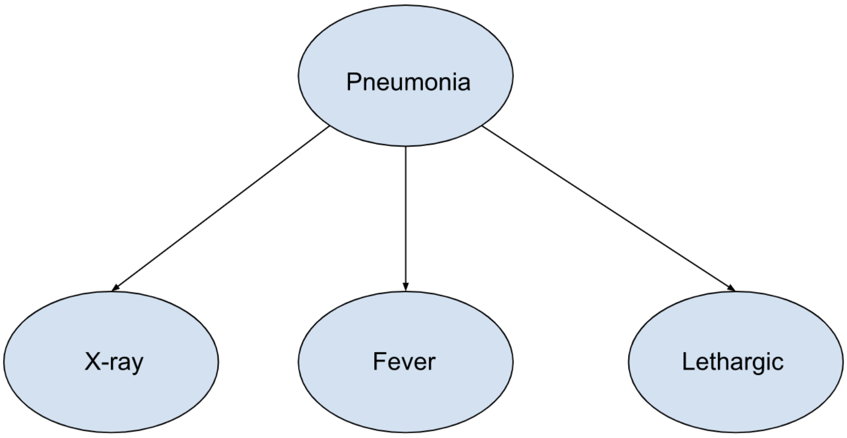


Figure 1: Graphical Model of a Naive Bayes Classifier

The Tree Augmented Network (TAN) [1] extends the capability of the simple Naive Bayes classifier by introducing edges between the features to capture correlations among them. But to make the computations tractable, the edges are assumed to follow a tree structure.

Figure 2 shows an example of a Tree Augmented Network. Note that a directed edge has been added from the Fever node to the Lethargic node. This makes sense because lethargy is often accompanied by fever. Also, note that all the features are connected to the class node. Hence in a TAN, a node can have zero or 1 parent node in addition to the class node. (Here, the class node is the pneumonia node). The advantage of TAN over Naive Bayes is that it can capture richer relationships between features and the larger class variable, yet the tree structure ensures that the computations are polynomial time and hence tractable for datasets involving a large number of features.

The Tree Augmented Network (TAN) is trained using the Chow-Liu algorithm [2, 3]. This makes use of mutual information between features while training. The mathematical details are explained in the next section. In essence, this is a structure learning algorithm. In addition to the conditional probability tables, the Chow-Liu algorithm also learns the structure of the bayesian network in terms of the directed edges between the features. Note that each node can have only one other node as a parent in addition to the class node which is a parent to all the nodes. This is a first-order structure learning algorithm (Naive Bayes can be thought of as a zeroth-order structure learning algorithm as it learns only the conditional probability tables). It is possible to extend the algorithm for learning richer structures and hence better correlations between the features but with additional computational complexity.

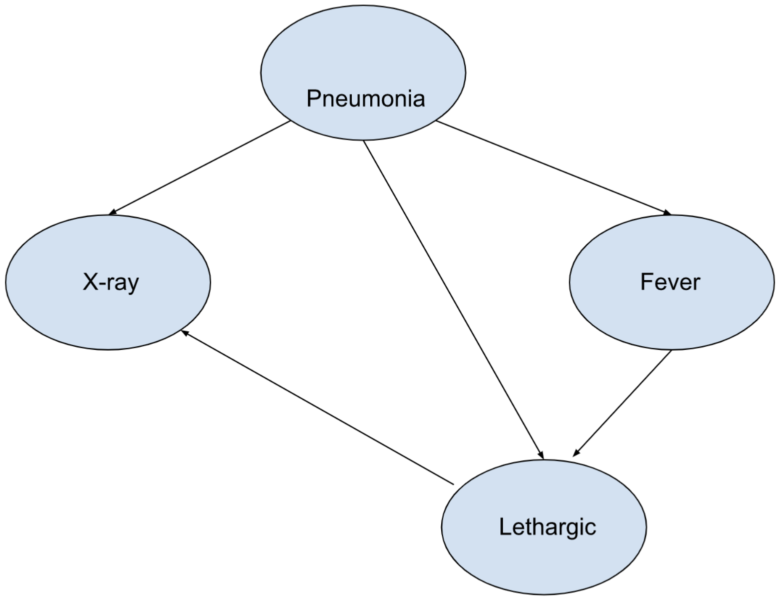


Figure 2: Graphical Model of a Naive Bayes Classifier

The following section describes the Naive Bayes model and the Tree Augmented Network in detail. The training and testing procedure in both models are described mathematically. Experiments are conducted on standard machine learning datasets available from the UCI machine learning repository [1]. Also, an experiment is conducted using a synthetic dataset generated with a toy bayesian network, the lung cancer bayesian network. The toy dataset is generated using a predefined conditional probability table and sampling from this conditional probability distribution. The experiments show that TAN outperforms Naive Bayes as well as performs competitively with the other machine learning methods such as decision trees and SVMs.

[1] UCI machine learning repository: https://archive.ics.uci.edu/ml/datasets.php

2 Literature review

Bayesian networks or the directed graphical models are introduced by Judea Pearl [3]. They are quite useful in knowledge representation involving uncertainty and enable reasoning through probabilistic inference. There are exact and approximate inference algorithms [4] for Bayesian networks. But as the size of the graph becomes large, exact inference become infeasible. Approximate inference techniques such as loopy belief propagation and methods based on sampling are popular.

Another challenge with using Bayesian networks is that expert knowledge is required to identify the structure of the graphical model. But structure learning algorithms that can learn the structure of the graph in addition to the conditional probability tables (CPT) are also proposed [4]. A major challenge with structure learning algorithms is that it quickly becomes infeasible as the size of the network grows. Hence certain relaxations have been made on the structure requirements. For example, in a tree augmented network [1], an augmented tree structure is learned. This makes learning the structure efficient and tractable. Inference in TAN is polynomial in time complexity which makes it robust and at the same time more powerful than Naive Bayes classifiers. The structure learning is attained using an algorithm known as the Chow-Liu algorithm [2] which makes use of the Prim's algorithm for finding a maximum spanning tree.

Machine learning classifiers are supervised algorithms that learns from examples of labeled data to make predictions on unseen test data. We use several datasets to compare the Bayesian classifiers against some of the standard machine learning algorithms such as the decision trees and support vector machine (SVM) classifiers. These experiments for comparison are conducted using the popular scikit-learn library [5].

3 Methodology

3.1 Naive Bayes Classifier

The graphical model representation of a Naive Bayes classifier is shown in Figure 1. Let's say the class node takes two values 'Yes' and 'No' for Pneumonia. The inference problem in a naive Bayes classifier is to compare the two conditional probabilities $P(\text{Pneumonia} = \text{'Yes'}|\text{X-ray, fever, lethargic})$ and $P(\text{Pneumonia} = \text{'No'}|\text{X-ray, fever, lethargic})$. We will predict the output as the value with the highest posterior probability.

Let's denote the class variable by C and the features by f_1, f_2 & f_3. Then using Bayes theorem, we can write,

$$P(C = \text{'Yes'}|f_1, f_2, f_3) > P(C = \text{'No'}|f_1, f_2, f_3)$$
$$\iff$$
$$P(f_1, f_2, f_3|C = \text{'Yes'}) * P(C = \text{'Yes'}) > P(f_1, f_2, f_3|C = \text{'No'}) * P(C = \text{'No'})$$

That means the posterior is computed as a product of the prior and the class conditional probability.

In a naive Bayes classifier, we assume that the features are conditionally independent. That means,

$$P(f_1, f_2, f_3|C) = P(f_1|C) * P(f_2|C) * P(f_3|C)$$

The prior probabilities and the conditional probabilities for each of the classes are learned using maximum likelihood estimation (MLE) from the training data. For example, the prior probability of class 'yes' is computed as the ratio of the number of occurrences of the class 'yes' in the training data. Smoothing such as add-one smoothing is also employed with the MLE techniques.

The class conditional probabilities are estimated as the ratio of occurrence of each feature f1 in each class. The learned prior and class conditional probabilities are typically stored as matrices or hash tables.

3.2 Tree Augmented Network

A Tree Augmented Network (TAN) is a Bayesian network with the constraint that every node can have at most one parent node in addition to the conditioning node (the target node). The structure of a tree augmented network is learned using the Chow-Liu algorithm [2].

The idea is to construct a graph with the features as the nodes and the edges between the features with weights as the mutual information between the features as learned from the data. Then using the mutual information as edge weights, a tree is learned (using either Kruskal's algorithm or Prim's algorithm). The vertex C is the class/category node and A_i's are the attribute nodes. The class conditional mutual information I, is calculated as

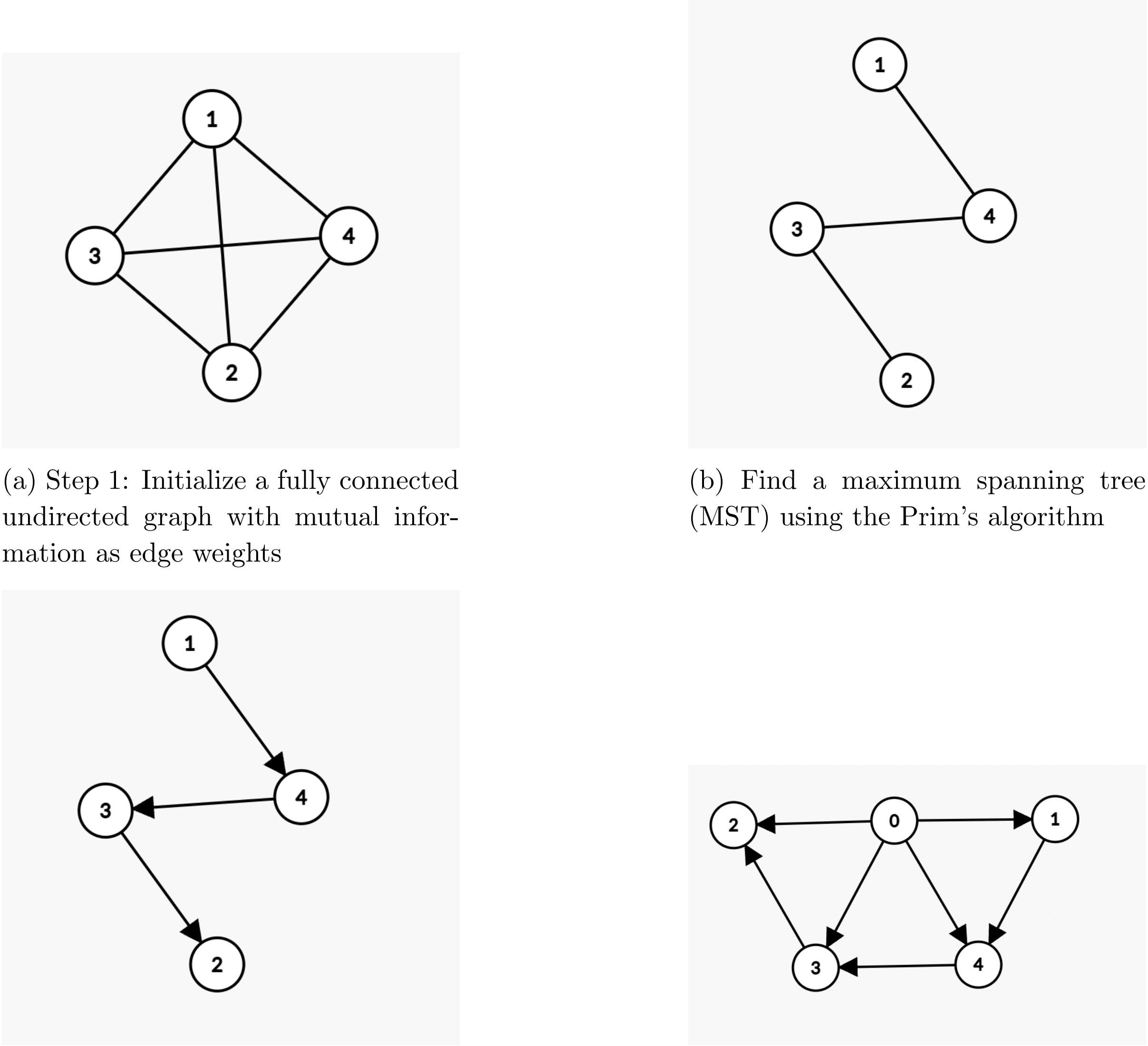
$$I(A_i; A_j|C) = \sum_{x,y,c} P(x, y, c) log \frac{P(x, y|c)}{P(x|c)P(y|c)} \tag{1}$$

where, x, y & c are the values taken by the variables A_i, A_j & C respectively.

Algorithm 1 The Construct-TAN procedure

```
1: Compute mutual information, $I(A_i; A_j|C)$ between each pair of attributes, $i \neq j$.
2: Build a complete undirected graph in which the vertices are the attributes $A_1, ..., A_n$.
   Set the weight of the edge connecting $A_i$ to $A_j$ to be the mutual information between
   them, $I(A_i; A_j|C)$, calculated using equation 1.
3: Build a maximum weighted spanning tree using Prim's algorithm.
4: Transform the resulting undirected tree to a directed one by choosing a root variable
   and setting the direction of all edges to be outward from it.
5: Construct a TAN model by adding a vertex labeled by $C$ and adding an arc from $C$
   to each $A_i$.
```

An illustration of the Chow Liu algorithm for a simple example graph with 4 feature nodes and a target node is shown in figure 3.



(a) Step 1: Initialize a fully connected undirected graph with mutual information as edge weights

(b) Find a maximum spanning tree (MST) using the Prim's algorithm

(c) Choose a random node and assign directions out of it

(d) Add the target node 0 and add directed edges to all other nodes from the target node

Figure 3: An Illustration of the Chow Liu Algorithm

3.2.1 Parameter Learning in Tree Augmented Networks

In order to calculate mutual information between variables (features of the dataset) using equation 1, we need to know the values of the individual conditional probabilities $P(X|C)$ as well as the joint conditional probabilities $P(X, Y|C)$. These can be estimated using the Maximum Likelihood Estimation (MLE) from the training data. Add one smoothing (also called Laplace smoothing) is applied to avoid any zero probabilities. All the probability calculation is performed in the log domain to avoid any underflow issues. Once the conditional probability tables (CPT) are learned, then the tree structure can be learned using the Chow-Liu algorithm as outlined in the algorithm 1.

3.2.2 Inference in Tree Augmented Networks

In order to make predictions using a tree augmented network (TAN) classifier, we need to compute the posterior probability of each class given the test data feature values. Same as in Naive Bayes classifier, the posterior probability can be written as a product of the prior and class conditional probabilities using Bayes theorem. The chain structure of the tree makes the calculation of the class conditional linear in time complexity. For example, in figure 3 (c), the class conditional probability can be decomposed as;

$$\begin{aligned} P(1,2,3,4|0) &= P(4|1,0)P(3|4,0)P(2|3,0)P(2|0) \\ &= \frac{P(1,4|0)P(3,4|0)P(2,3|0)}{P(4|0)P(3|0)} \end{aligned}$$

Since we have already estimated each of the above conditional probabilities, the posterior probabilities can be easily computed.

4 Results

4.1 Synthetic Dataset

A synthetic dataset is created from a toy bayesian network. The following is a popular example of a bayesian network shown in Figure 4.

We set Cancer as the target variable. Hence the objective is to predict whether a patient has cancer given the pollution level (H or L) of the city where they live, whether the patient is a smoker (T or F), whether the Xray result is positive or negative, and whether the patient has dyspnoea or not.

We can generate data by following the conditional probability sampling given in figure 3. For instance, the probability of pollution is high (H) is 0.1. The probability of smoking is 0.3. Then using the conditional probability for the Cancer variable, generate a sample conditioned on the generated values (0 or 1 corresponding to each node). This way a complete data point can be generated. One problem with using the given values is that the probability of Cancer when given the values of the features, $P(\text{Cancer} = \text{True}|\text{Pollution} = \text{High}, \text{Smoker} = \text{True}, \text{Xray=pos}, \text{Dysponea=T})$ is

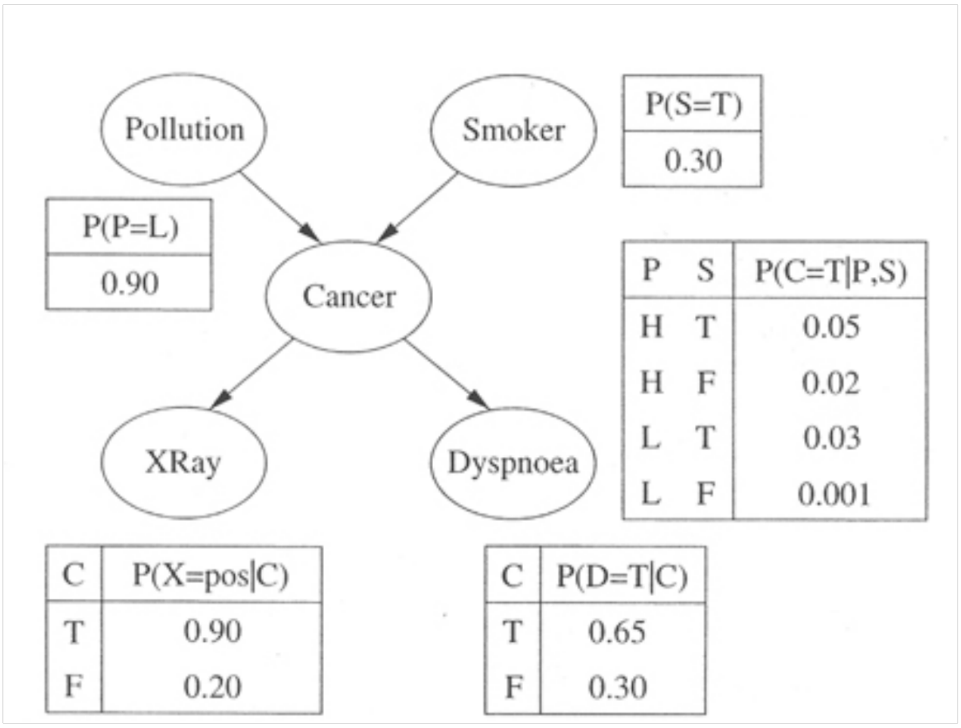


Figure 4: Lung Cancer Bayesian Network

very small (< 0.1). Hence any classifier trained on the above data always predicts the class label (Cancer) as False.

This is quite reasonable since the false positive rate of the X-rays is quite high (0.2) as well as Dyspnoea is also not a good indicator of the target (Cancer). Hence the given conditional probabilities are tweaked slightly as follows.

$p_1 = 0.1$ (prob. of pollution)
$s_1 = 0.2$ (prob. of being a smoker)
$c_{11} = 0.5$ (prob. of cancer if pollution smoker)
$c_{12} = 0.2$ (if pollution but not smoker)
$c_{21} = 0.3$ (if smoker but no pollution)
$c_{22} = 0.01$ (prob. cancer if no smoking or pollution)
$x_1 = 0.95$ (x-ray sensitivity)
$x_2 = 0.01$ (x-ray false positive rate)
$d_1 = 0.85$ (dyspnoea if cancer)
$d_2 = 0.02$ (dysnpnoea if no cancer)

Now the posterior probability of cancer given all the features are positive (pollution=H, smoker=T, Xray=Pos, Dyspnoea=T) is about 0.9997. Hence this bayesian network can generate data that can be used for training machine learning classifiers.

100,000 training data points and 10,000 test data points are generated using the procedure described above. Note that the dataset is highly imbalanced. Only about 8.8% of the dataset has the positive target class (Cancer = True). Of the rest, 91.2% of the patients are not having cancer. Hence it is not advised to use accuracy as the predictive performance measure. A good measure in the case of imbalanced data is the f1-score. The f1-score is defined as the harmonic mean between precision and recall.

$$\text{f1-score} = 2 \times \frac{\text{precision} \times \text{recall}}{(\text{precision} + \text{recall})}$$

The table 1 shows the performance of different machine learning methods on the dataset described above.

Classifier	F1-score
Naive Bayes	82.708%
Decision Trees	95.769%
SVM	95.598%
TAN	**99.280%**

Table 1: F1 scores of different ML algorithms for the Lung Cancer Dataset

The proposed method, TAN, is a clear winner for the synthetic dataset generated. The structure of the graphical model as learned by the Chow-Liu algorithm is shown in Figure 5. It has learned some correlations between the features.

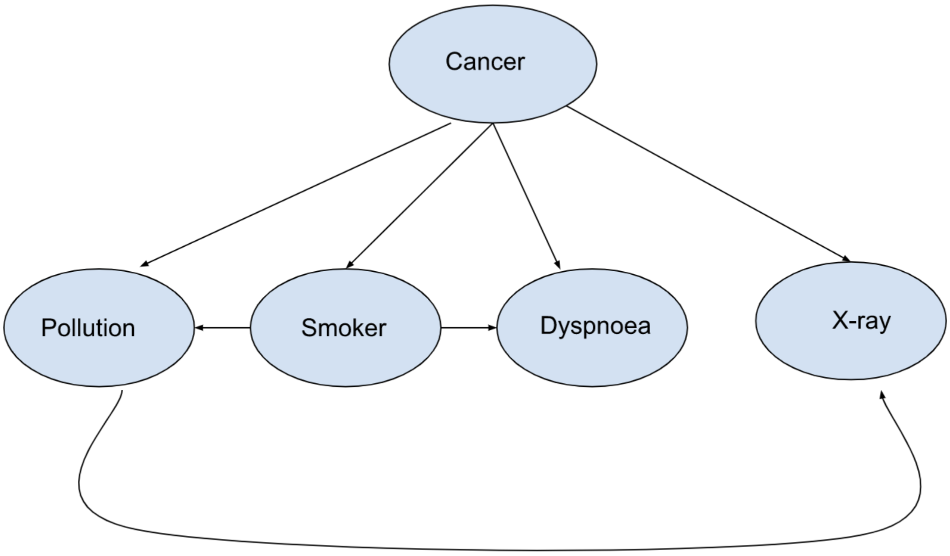


Figure 5: Lung Cancer Bayesian Network

4.2 Experiments on the UCI Dataset

The UCI machine learning repository has several datasets. The iris dataset, chess dataset, and diabetes dataset are used for experiments. The f1 score of various ML algorithms on these datasets is tabulated in Table 2.

Datasets	Naive Bayes	Decision Tree	SVM	TAN (Chow-Liu)
Iris	93.8%	93.5%	93.1%	94.1%
Chess	87.2%	92.7%	92.4%	92.3%
Diabetes	74.3%	74.8%	75.2%	75.3%

Table 2: F1 scores of different ML algorithms on UCI Dataset

We can see that the Chow-Liu algorithm gives competitive performance over other standard ML algorithms.

5 Conclusion

Naive Bayes and Tree Augmented Networks (TAN) are bayesian networks that can be used for machine learning classification. We have seen that TAN outperforms NB classifier significantly and performs comparably with other standard machine learning algorithms. TAN is able to learn structure in bayesian networks which may not be very much realistic but can learn the correlation between features.

References

[1] N. Friedman, D. Geiger, and M. Goldzmidt, "Bayesian network classifiers," *Machine Learning*, vol. 29, pp. 131–163, 1997.

[2] C. K. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Trans. Inf. Theory*, vol. 14, pp. 462–467, 1968.

[3] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[4] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.