CS228 Homework 2 Solutions

Instructor: Stefano Ermon – `ermon@stanford.edu`

Available: 01/24/2017; Due: 02/03/2017

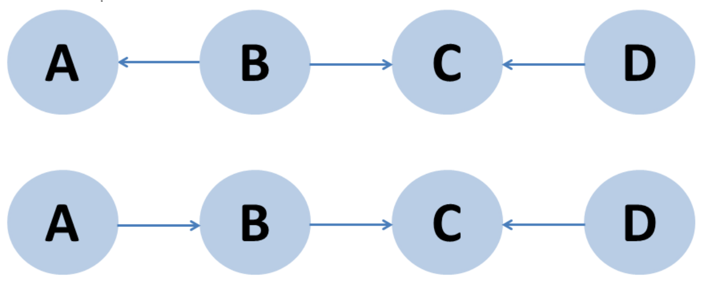
---

1. **[8 points]** *(I-Maps and P-Maps)* Consider a probability distribution P over the variables A, B, C and D that satisfies only the following independencies:

   1. $A \perp C|B$.
   2. $A \perp C|B, D$.
   3. $A \perp D|B$.
   4. $A \perp D|B, C$.
   5. $B \perp D$.
   6. $B \perp D|A$.
   7. $A \perp D$.

   (a) **[3 points]** Draw a Bayesian network that is a perfect map for P.

   Answers: The 2 graphs below are both perfect maps for P. Either of them is valid, and the other is the I-equivalent graph for part (b).



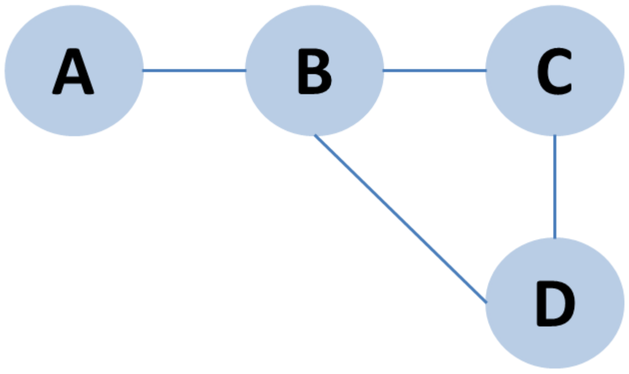
   (b) **[2 points]** Does this perfect map have any I-equivalent graphs? If so, draw them. If not, explain why not.

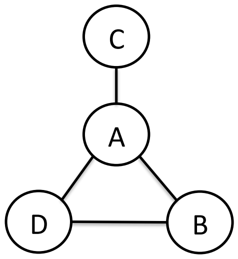
   Answers: Yes. See the solution to part (a) for the graph.

   (c) **[3 points]** Draw a Markov network that is a minimal I-map for P and explain why the Markov network is or is not also a perfect map.

   Answers: A Markov net with these undirected edges and additional $B - D$ edge is a valid I-map. This Markov net is not a perfect map because B and D are not independent in this network, while they are independent in the original probability distribution.

2. **[9 points]** *(I-Maps and P-Maps)* Consider a distribution P over four binary random variables (A, B, C, D), which gives probability 1/8 to assignments $(0, 0, 0, 0)$, and $(1, 1, 0, 0)$, and gives probability 1/4 to assignments $(1, 1, 1, 0)$, $(0, 1, 0, 1)$, and $(1, 0, 1, 1)$. A skeleton for the Bayesian network G is also provided; the skeleton contains all the nodes and edges, but the directions of the edges are missing.





(a) **[2 points]** Decide whether the following two independencies are in $I(P)$: $(C \perp D), (C \perp B)$
Answers: First, compute the marginal distribution on (C, D):

$$P(C=0, D=0) = (1/8) + (1/8) = 1/4; \quad P(C=0, D=1) = 1/4;$$
$$P(C=1, D=0) = 1/4; \quad P(C=1, D=1) = 1/4.$$

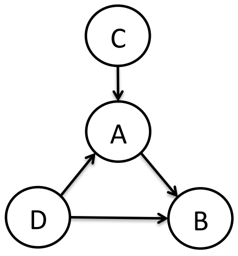
One can easily check that $P(C, D) = P(C)P(D)$.
Then, by definition of independence, $(C \perp D) \in I(P)$.

Second, compute the marginal distribution on (B, C):

$$P(B=0, C=0) = 1/8; \quad P(B=0, C=1) = 1/4;$$
$$P(B=1, C=0) = (1/8) + (1/4) = 3/8; \quad P(B=1, C=1) = 1/4.$$

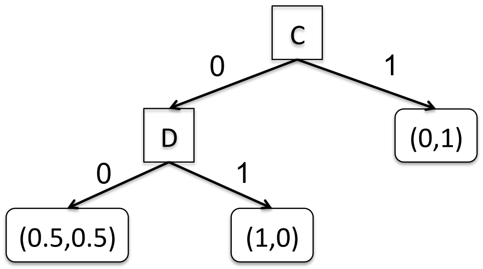
From above, $P(B=0) = 3/8$ and $P(C=0) = 1/2$. So, $P(B=0, C=0) \neq P(B=0)P(C=0)$.
Then, again by definition of independence, $(C \perp B) \notin I(P)$.

(b) **[5 points]** Give a direction to each individual edge in G, such that the resulting Bayesian network is a perfect map of P. Is your solution unique? Briefly state why.
Answers:
The only way to have $(C \perp D)$ is to have a V-structure: $D \to A \leftarrow C$; otherwise there will be an active trail from C to D via A.
Moreover, the only possible direction of the edge between A and B is: $A \to B$; otherwise, we will have a V-structure: $B \to A \leftarrow C$, which contradicts the fact that $(C \perp B) \notin I(P)$.
Lastly, we must have $D \to B$; otherwise, we'll have a triangular loop (A, B, D), which contradicts the acyclicity requirement of Bayesian network.
From the description above, one can see that such result is unique.



(c) [**2 points**] Give the CPDs for each node in your Bayesian network specified in (b).
Answers:

- CPD for variable C:
$$P(C=0)=P(C=1)=1/2.$$
- CPD for variable D:
$$P(D=0)=P(D=1)=1/2.$$
So, both C and D are Bernoulli(1/2). We'll just use tabular CPDs for C, D.
- CPD for variable B: from probability assignments mentioned in the problem description, one can easily convince himself/herself that $B = (A \text{ XOR } D)$. So a deterministic function (XOR) will be sufficient to characterize this CPD.
- CPD for variable A:
$$\begin{aligned}
&P(A=0 \mid C=0, D=0)=1/2, && P(A=1 \mid C=0, D=0)=1/2;\\
&P(A=0 \mid C=1, D=0)=0, && P(A=1 \mid C=1, D=0)=1;\\
&P(A=0 \mid C=0, D=1)=1, && P(A=1 \mid C=0, D=1)=0;\\
&P(A=0 \mid C=1, D=1)=0, && P(A=1 \mid C=1, D=1)=1.
\end{aligned}$$
We'll use the following tree CPD to represent it:



3. [**26 points**] *(Undirected Graphical Models, Inference)* **The Restricted Boltzmann Machine**
Restricted Boltzmann machines (RBMs) have been widely used as a generative model in many fields of machine learning: image processing, speech recognition and collaborative filtering. Concretely, an RBM is an undirected graphical model defined on variables $(\mathbf{v}, \mathbf{h})$, $\mathbf{v} \in \{0,1\}^m$ and $\mathbf{h} \in \{0,1\}^n$. The joint probability is given by
$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z}\exp(\phi(\mathbf{v}, \mathbf{h}))$$
where
$$\phi(\mathbf{v}, \mathbf{h}) = -\alpha^T\mathbf{v} - \beta^T\mathbf{h} - \mathbf{v}^T W\mathbf{h}$$
is a potential function. Here $\alpha \in \mathbb{R}^m$, $\beta \in \mathbb{R}^n$, $W \in \mathbb{R}^{m\times n}$ and Z is the normalizing constant. You can interpret it as a fully connected *bipartite* network with two layers: one for visible variables $\mathbf{v}$ and one for hidden variables $\mathbf{h}$. In this problem, you will explore inference and learning for RBMs.

(a) **[5 points]** What is the marginal conditional distribution of $P(\mathbf{h}_i \mid \mathbf{v})$ for a single hidden variable $\mathbf{h}_i$? Can it be computed tractably?

Answer:

$$P(h_i|v) = \exp(-h_i * (\sum_j W_{ji}v_j) - \beta_i * h_i)/(\sum_{h_i} \exp(-h_i * (\sum_j W_{ji}v_j) - \beta_i * h_i))$$

(b) **[5 points]** What is the conditional distribution $P(\mathbf{h} \mid \mathbf{v})$? Can it be expressed in a compact (factored) form? **Answer:** Yes, product of the above marginals

(c) **[2 points]** Suppose we are given samples for the visible units $\mathcal{D} = \{\mathbf{v}^1, \cdots, \mathbf{v}^K\}$ (e.g., the MNSIT digits dataset), and we want to learn the parameters of the RBM to maximize the likelihood of these samples. Since we don't know the values of hidden variables $\mathbf{h}^k$, a natural approach is to look for parameters that maximize the marginal likelihood of each sample $\mathbf{v}^k$, given by

$$L(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{h}, \mathbf{v}) = \sum_{\mathbf{h}} \frac{1}{Z} \exp(\phi(\mathbf{v}, \mathbf{h})$$

For a fixed $\mathbf{v}$, can

$$\sum_{\mathbf{h}} \exp(\phi(\mathbf{v}, \mathbf{h}))$$

be computed efficiently (in time not exponential in n)?

Answer: Yes, the sum factors because of conditional independency

(d) **[2 points]** For a fixed $\mathbf{h}$, can

$$\sum_{\mathbf{v}} \exp(\phi(\mathbf{v}, \mathbf{h}))$$

be computed efficiently (in time not exponential in m)?

Answer: Yes, the sum factors because of conditional independency

(e) **[2 points]** Can the normalization constant

$$Z = \sum_{\mathbf{h}} \sum_{\mathbf{v}} \exp(\phi(\mathbf{v}, \mathbf{h}))$$

be computed efficiently (in time not exponential in m or n)?

Answer: No

4. **[10 points]** [**Maximum Likelihood Learning of Bayes Nets**] Let G be a valid Bayes Net structure, and $\mathcal{D}$ be some training data. It can be shown that the likelihood of $\mathcal{D}$ for a given choice of the parameters θ (specifying the conditional probability tables in G) is

$$\ell_G(\theta; \mathcal{D}) = \sum_{i=1}^{n} \sum_{\boldsymbol{u}_i \in Val(Pa(X_i))} \sum_{x_i} M[x_i, \boldsymbol{u}_i] \log \theta_{x_i|\boldsymbol{u}_i}$$

We have seen in class that the maximum likelihood estimate $\theta^{ML} = \arg\max_\theta \ell_G(\theta; \mathcal{D})$ of the parameters of G can be computed as follows

$$\theta^{ML}_{x_i|\boldsymbol{u}_i} = \frac{M[x_i, \boldsymbol{u}_i]}{M[\boldsymbol{u}_i]} \tag{1}$$

where $M[x_i, \boldsymbol{u}_i]$ is the number of times the tuple $(X_i = x_i, Pa(X_i) = \boldsymbol{u}_i)$ appears in the data $\mathcal{D}$.

Let G' be a valid Bayes net structure obtained by adding an edge to G. Show that

$$\max_{\theta'} \ell_{G'}(\theta'; \mathcal{D}) \geq \max_{\theta} \ell_G(\theta; \mathcal{D})$$

Note that G' is stricly more expressive than G (G' makes less conditional independence assumptions). This is confirmed by the fact that to specify a Bayes Net with structure G' we need more parameters, i.e., the vector θ' has more components than θ. The goal of this exercise is to show directly that the more "complex" a Bayesian Network is, the better we can fit it to data.

You may assume that G and G' are identical, except for an extra $x_1 \to x_n$ edge in G'.

Hint: There are several ways to show this. Jensen's inequality might come handy.

Answers

$$\ell_{G'}(\theta_{G'}^{ML};\mathcal{D}) - \ell_G(\theta_G^{ML};\mathcal{D}) =$$

$$\sum_{\widehat{\boldsymbol{u}}_n \in Val(Pa(X_n)\cup X_1)} \sum_{x_n} M[x_n, \widehat{\boldsymbol{u}}_n] \log \frac{M[x_n, \widehat{\boldsymbol{u}}_n]}{M[\widehat{\boldsymbol{u}}_n]} - \sum_{\boldsymbol{u}_n \in Val(Pa(X_n))} \sum_{x_n} M[x_n, \boldsymbol{u}_n] \log \frac{M[x_n, \boldsymbol{u}_n]}{M[\boldsymbol{u}_n]} =$$

$$\sum_{x_1} \sum_{\boldsymbol{u}'_n \in Val(Pa(X_n))} \sum_{x_n} M[x_n, \boldsymbol{u}'_n, x_1] \log \frac{M[x_n, \boldsymbol{u}'_n, x_1]}{M[\boldsymbol{u}'_n, x_1]} - \sum_{\boldsymbol{u}_n \in Val(Pa(X_n))} \sum_{x_n} M[x_n, \boldsymbol{u}_n] \log \frac{M[x_n, \boldsymbol{u}_n]}{M[\boldsymbol{u}_n]} =$$

$$\sum_{x_1} \sum_{\boldsymbol{u}'_n \in Val(Pa(X_n))} \sum_{x_n} M[x_n, \boldsymbol{u}'_n, x_1] \log \frac{M[x_n, \boldsymbol{u}'_n, x_1]}{M[\boldsymbol{u}'_n, x_1]} - \sum_{\boldsymbol{u}_n \in Val(Pa(X_n))} \sum_{x_n} \sum_{x_1} M[x_n, \boldsymbol{u}_n, x_1] \log \frac{M[x_n, \boldsymbol{u}_n]}{M[\boldsymbol{u}_n]} =$$

$$\sum_{x_1} \sum_{\boldsymbol{u}'_n \in Val(Pa(X_n))} \sum_{x_n} M[x_n, \boldsymbol{u}'_n, x_1] \left(\log \frac{M[x_n, \boldsymbol{u}'_n, x_1]}{M[\boldsymbol{u}'_n, x_1]} - \log \frac{M[x_n, \boldsymbol{u}'_n]}{M[\boldsymbol{u}'_n]} \right) =$$

$$\sum_{x_1} \sum_{\boldsymbol{u}'_n \in Val(Pa(X_n))} \sum_{x_n} M[x_n, \boldsymbol{u}'_n, x_1] \log \frac{M[x_n, \boldsymbol{u}'_n, x_1] M[\boldsymbol{u}'_n]}{M[\boldsymbol{u}'_n, x_1] M[x_n, \boldsymbol{u}'_n]} =$$

$$|\mathcal{D}| \sum_{x_1} \sum_{\boldsymbol{u}'_n \in Val(Pa(X_n))} \sum_{x_n} \frac{M[x_n, \boldsymbol{u}'_n, x_1]}{|\mathcal{D}|} \log \frac{M[x_n, \boldsymbol{u}'_n, x_1] M[\boldsymbol{u}'_n]}{M[\boldsymbol{u}'_n, x_1] M[x_n, \boldsymbol{u}'_n]} =$$

$$|\mathcal{D}| \sum_{x_1} \sum_{\boldsymbol{u}'_n \in Val(Pa(X_n))} \sum_{x_n} \widehat{P}[x_n, x_1, \boldsymbol{u}'_n] \log \frac{M[x_n, \boldsymbol{u}'_n, x_1] M[\boldsymbol{u}'_n]}{M[\boldsymbol{u}'_n, x_1] M[x_n, \boldsymbol{u}'_n]} =$$

$$|\mathcal{D}| \sum_{\boldsymbol{u}'_n \in Val(Pa(X_n))} \sum_{x_1} \sum_{x_n} \widehat{P}[x_n, x_1, \boldsymbol{u}'_n] \log \frac{\widehat{P}[x_n, x_1 | \boldsymbol{u}'_n]}{\widehat{P}[x_1 | \boldsymbol{u}'_n] \widehat{P}[x_n | \boldsymbol{u}'_n]} = |\mathcal{D}| I_{\widehat{P}}(X_n; X_1 | Pa(X_n) \geq 0$$

5. **[47 points]** *(Programming Assignment)* **Tree Augmented Naive Bayes**
Naive Bayes (NB) classifiers are often competitive classifiers even though their strong independence assumptions may be unrealistic. We will explore an extension of this classification framework, Tree Augmented Naive Bayes (TANB), which will aim to improve the results of NB by modelling correlations between attributes while still being computationally tractable. If C denotes the class variable and $(A_1, \ldots, A_N)$ the attributes, then a NB model can be represented as a directed graph with these variables as nodes and edges $\{(C, A_i) : 1 \leq i \leq n\}$. A TANB extends this graph structure by allowing each child A_i to have one additional node A_j as a parent with the restriction that these additional edges (A_i, A_j) must form a tree. Figure 1 illustrates the difference in the graph structure.

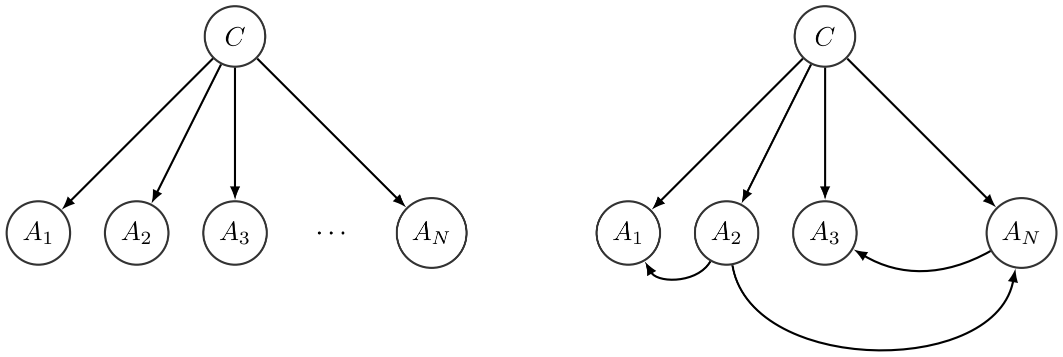


Figure 1: A Naive Bayes model and Tree Augmented Naive Bayes model

In general, learning the structure of this graph can be hard, but we will cover this topic later in the course. For now we have supplied starter code necessary to learn the structure of the graph for the dataset.

In this problem, you will implement the parameter learning for both NB and TANB classifiers and explore the trade-offs of both. You will apply these classifiers to predict the party affiliation of either Democrat or Republican of US Congressmen (the class variable) based on their votes for 16 different measures (the attribute variables) shown in Table 1. Not all Congressmen voted on all 16 measures so sometimes entries in this dataset will have missing attributes, however, we will still be able to utilize our Bayes Network to accurately classify these examples. To keep things simple, the class and attribute variables are all binary with 0, 1 corresponding to a no, yes vote respectively. Missing attributes will have a value of -1 assigned in the raw data.

When training the models, some of the parameters may not have enough examples for accurate estimation. To mitigate this, we will perform Laplace smoothing with a pseudocount of $\alpha = 0.1$ for every value of an attribute given its parents when learning the parameters.

In order to evaluate the performance of our classifiers on the dataset, we will use 10-fold cross-validation. Under 10-fold cross-validation, the dataset is first partitioned into 10 equally sized partitions. Of these 10 partitions, one of them is used for the test set while the rest of the data are used as the training set to compute test error on this partition. This process is repeated for the other nine partitions, and we can take the average of the resulting test errors to obtain the 10-fold CV test error. We have implemented this procedure for you in the function `evaluate`.

(a) **[8 points]**
Implement a NB classifier that both learns the parameters from the training data and can use these parameters to score and classify examples in the training data. What is your test error rate using 10-fold cross-validation?

Note: you can use the `evaluate` *function in the starter code, but leave the optional argument* `train_subset` *to its default value until part d.*

Answers 10-fold cross validation total test accuracy 0.9181 on 232 examples

(b) **[12 points]**

Attribute	Name	Incomplete Entry 1
A_1	handicapped infants	1
A_2	water project cost sharing	1
A_3	adoption of the budget resolution	0
A_4	physician fee freeze	0
A_5	El Salvador aid	0
A_6	religious groups in schools	0
A_7	anti satellite test ban	0
A_8	aid to Nicaraguan Contras	?
A_9	mx missile	?
A_{10}	immigration	0
A_{11}	synfuels corporation cutback	0
A_{12}	education spending	?
A_{13}	superfund right to sue	?
A_{14}	crime	?
A_{15}	duty free exports	0
A_{16}	export administration act south Africa	0

Table 1: Attribute names for Congressional Voting Records together with an incomplete example that has some voting records missing for a particular Congressman.

We have provided starter code that implements the Chow-Liu algorithm (see Friedman et al. 1997) to learn the tree structure from the training data for a TANB. Using this tree, implement a TANB classifier that learns the parameters from the training data and uses these to score examples. What is your test error rate using 10-fold cross-validation?

Answers 10-fold cross validation total test accuracy 0.9526 on 232 examples

(c) **[15 points]**

In general, working with data where the values of attributes and labels are missing is difficult when learning model parameters. However, we can still use our generative model from a fully trained Bayes Network to classify examples in which some of the attributes may be unobserved. Suppose A_i is unobserved. We can still compute $P(C|A_1, \cdots, A_{i-1}, A_{i+1}, \cdots, A_N)$ by computing $P(C, A_i|A_1, \cdots, A_{i-1}, A_{i+1}, \cdots, A_N)$ and marginalizing over A_i. Update your TANB implementation to handle the case where some attributes may have missing values and use this new implementation to classify Incomplete Entry 1 in Table 1[1]. Given the observed attributes, what is the marginal probability this Congressman is Democrat ($C = 1$) given the votes we did observe? Can you predict how this Congressman voted on education spending (A_{12})?

Note the power of a generative model: it can easily handle missing data and can be used to answer all sorts of probabilistic queries. In contrast, this would *not* be possible with a discriminative model, e.g., if you trained a logistic regression or a random forest classifier to directly predict the affiliation of a Congressman (C), because these models would only provide you with the conditional distribution $P(C|A_1, \cdots, A_N)$.

Answers

[1] Your implementation does not have to be fully general. It's fine as long as it works on that particular example.

TANB Classifier on missing data
$P(C = 1|A_{\text{observed}}) = 0.9899$
$P(A_{12} = 1|A_{\text{observed}}) = 0.1022$

Note: You should train your classifier on the full dataset. You can use the function `evaluate_incomplete_entry`*, which both trains on the full dataset and loads Incomplete Entry 1 for classification.* `evaluate`.

(d) [**12 points**]

What is the test error when you train both classifiers on a smaller subset of the training data? Explain why the test error for the TANB may not strictly be better than NB.

Note: set the arguments `train_subset=True` *when calling* `evaluate` *so that the classifiers are trained on a much smaller subset of the data.*

Answers

Naive Bayes
10-fold cross validation total test accuracy 0.9009 on 232 examples
TANBClassifier
10-fold cross validation total test accuracy 0.8922 on 232 examples

Training on a smaller subset of the data could lead to the TANB overfitting and learning dependencies present in the training data that do not hold across the dataset.