Assignment 3: Bayesian Networks, Inference and Learning

CS486/686 – Winter 2020

Out: February 20, 2020
Due: March 11, 2020 at 5pm

Submit your assignment via LEARN (CS486 site) in the Assignment 3 Dropbox folder.
No late assignments will be accepted

PART A [60pts]: INFERENCE IN BAYESIAN NETWORKS

Every year, credit card companies lose millions of dollars due to frauds resulting from lost or stolen cards. Recently, the financial industry has turned to AI for solutions to the fraud detection problem. Intuitively, credit card holders tend to make purchases following a certain pattern. A fraud is likely to happen when this pattern is broken. In this assignment you will implement a simple fraud detection system in two steps. First, you will implement the variable elimination algorithm. Second, you will define your fraud detection system as a Bayesian network and compute the likelihood of a fraud in different situations. Computations in your Bayesian network should be done with your implementation of the variable elimination algorithm. However, if you are not able to complete your implementation of variable elimination, you can also do the computations by hand since they are relatively simple.

1. **[0 pts]** Implement the variable elimination algorithm by coding the following 4 functions in the programming language of your choice.

   (a) **restrictedFactor = restrict(factor, variable, value):** function that restricts a variable to some value in a given factor.

   (b) **productFactor = multiply(factor1, factor2):** function that multiplies two factors.

   (c) **resultFactor = sumout(factor, variable):** function that sums out a variable in a given factor.

   (d) **normalizedFactor = normalize(factor):** function that normalizes a factor by dividing each entry by the sum of all the entries. This is useful when the factor is a distribution (i.e. sum of the probabilities must be 1).

   (e) **resultFactor = inference(factorList, queryVariables, orderedListOfHiddenVariables, evidenceList):** function that computes $Pr(queryVariables|evidenceList)$ by variable elimination. This function should restrict the factors in factorList according to the evidence in evidenceList. Next, it should sumout the hidden variables from the product of the factors in factorList. The products and sums should be interleaved so that each summation is done over the minimal product of factors necessary. The variables should be summed out in the order given in orderedListOfHiddenVariables. Finally, the answer can be normalized if a probability distribution that sums up to 1 is desired.

   Tip: factors are essentially multi-dimensional arrays, hence you may want to use this data-structure, but feel free to use a different data-structure.

   Tip: test each function individually using simple examples from the lecture slides. If you wait till the end to test your entire program it will be much harder to debug.

   What to hand in: hand in a printout of your code. Note that there are no marks given for Question 1, but you must hand in your code in order to get any marks for Question 2. Also, in Question 2, part of the marks will be given for using your variable elimination algorithm to do the computations.

A clear description of what the marking scheme is, complete with decision tree is therefore:

(a) you have to hand in your code to get any marks for Q.A.2

(b) if you do your calculations by hand in Q.A.2 you get a maximum of 2/3 the marks (attaining the maximum if you do the calculations all correctly and it is sufficiently clearly presented).

(c) If you do your calculations with your code (and this is verifiable), then you get a maximum of full marks for the question (if the calculations are all done correctly).

(d) If you hand in code, use the code, but produce incorrect results, then you get a maximum of 1/3 the mark (for the code)

2. Suppose you are working for a financial institution and you are asked to implement a fraud detection system. You plan to use the following information:

   - When the card holder is travelling abroad, fraudulent transactions are more likely since tourists are prime targets for thieves. More precisely, 1% of transactions are fraudulent when the card holder is travelling, where as only 0.4% of the transactions are fraudulent when she is not travelling. On average, 5% of all transactions happen while the card holder is travelling. If a transaction is fraudulent, then the likelihood of a foreign purchase increases, unless the card holder happens to be travelling. More precisely, when the card holder is not travelling, 10% of the fraudulent transactions are foreign purchases where as only 1% of the legitimate transactions are foreign purchases. On the other hand, when the card holder is travelling, then 90% of the transactions are foreign purchases regardless of the legitimacy of the transactions.
   - Purchases made over the internet are more likely to be fraudulent. This is especially true for card holders who don't own any computer. Currently, 60% of the population owns a computer and for those card holders, 1% of their legitimate transactions are done over the internet, however this percentage increases to 2% for fraudulent transactions. For those who don't own any computer, a mere 0.1% of their legitimate transactions is done over the internet, but that number increases to 1.1% for fraudulent transactions. Unfortunately, the credit card company doesn't know whether a card holder owns a computer, however it can usually guess by verifying whether any of the recent transactions involve the purchase of computer related accessories. In any given week, 10% of those who own a computer purchase with their credit card at least one computer related item as opposed to just 0.1% of those who don't own any computer.

   (a) **[15 pts]** Construct a Bayes Network that your fraud detection system can use to indentify fraudulent transactions.

   What to hand in: Show the graph defining the network and the Conditional Probability Tables associated with each node in the graph. This network should encode the information stated above. Your network should contain exactly six nodes, corresponding to the following binary random variables:

   - OC – card holder owns a computer.
   - $Fraud$ – current transaction is fraudulent.
   - $Trav$ – card holder is currently travelling.
   - FP – current transaction is a foreign purchase.
   - IP – current purchase is an internet purchase.
   - CRP – a computer related purchase was made in the past week.

   The arcs defining your Bayes Network should accurately capture the probabilistic dependencies between these variables.

   (b) **[20 pts]** What is the prior probability (i.e., before we search for previous computer related purchases and before we verify whether it is a foreign and/or an internet purchase) that the current transaction is a fraud? What is the probability that the current transaction is a fraud once we have verified that it is a foreign transaction, but not an internet purchase and that the card holder purchased computer related accessories in the past week?

What to hand in: Indicate what queries (i.e., $Pr(variables|evidence)$) you used to compute those probabilities. Whether you answer the queries by hand or using the code you wrote for Question 1, provide a printout of the factors computed at each step of variable elimination. Use the following variable ordering when summing out variables in variable elimination: $Trav$, FP, $Fraud$, IP, OC, CRP. Note that a maximum of two thirds of the marks are earned if you answer correctly the question by doing the computations by hand instead of using your program. No marks are awarded for answering the questions using some other software.

(c) **[20 pts]** After computing those probabilities, the fraud detection system raises a flag and recommends that the card holder be called to confirm the transaction. An agent calls at the domicile of the card holder but she is not home. Her spouse confirms that she is currently out of town on a business trip. How does the probability of a fraud change based on this new piece of information?

What to hand in: Same as for Question 2b.

(d) **[5 pts]** Suppose you are not a very honest employee and you just stole a credit card. You know that the fraud detection system uses the Bayes net designed earlier but you still want to make an important purchase over the internet. What can you do prior to your internet purchase to reduce the risk that the transaction will be rejected as a possible fraud?

What to hand in: Tell me the action taken and indicate by how much the probability of a fraud gets reduced. Follow the same instructions as for Question 2b.

PART B [40pts]: NAÏVE BAYES LEARNING

In assignment 2, you learned a decision tree to classify text documents in two sets given a labeled training set. Here you will learn a Naïve Bayes classifier for the same data. The data is made from a subset of Reddit posts sourced from `https://files.pushshift.io/reddit/` and processed it using Google BigQuery. The dataset includes the first 1500 comments of August 2019 of each of the r/books and r/atheism subreddits, cleaned by removing punctuation and some offensive language, and limiting the words to only those used more than 3 times among all posts. These 3000 comments are split evenly into training and testing sets (with 1500 documents in each).

To simplify your implementation, these posts have been pre-processed and converted to the *bag of words* model. More precisely, each post is converted to a vector of binary values such that each entry indicates whether the document contains a specific word or not. Each line of the files trainData.txt and testData.txt are formatted ”docId wordId” which indicates that word wordId is present in document docId. The files trainLabel.txt and testLabel.txt indicate the label/category (1=`atheism` or 2=`books`) for each document (docId = line#). The file words.txt indicates which word corresponds to each wordId (denoted by the line#). If you are using Matlab, the file loadScript.m provides a simple script to load the files into appropriate matrices. At the Matlab prompt, just type ”loadScript” to execute the script. Feel free to use any other language and to build your own loading script for the data if you prefer.

Implement code to learn a naïve Bayes model by maximum likelihood[1]. More precisely, learn a Bayesian network where the root node is the label/category variable with one child variable per word feature. The word variables should be binary and represent whether that word is present or absent in the document. Learn the parameters of the model by maximizing the likelihood of the training set only. This will set the class probability to the fraction of documents in the training set from each category, and the probability of a word given a document category as the fraction of documents in that category that contain that word. You should use a *Laplace correction* by adding 1 to numerator and 2 to the denominator, in order to avoid situations where both classes have probability of 0. Classify documents by computing the label/category with the highest posterior probability $\Pr(label|\text{words in document})$. Report the training and testing accuracy (i.e., percentage of correctly classified articles).

[1] For the precise equations for this, see the note on the course webpage `https://cs.uwaterloo.ca/~jhoey/teaching/cs486/naivebayesml.pdf`

What to hand in:

- **[10 pts]** A printout of your code.
- **[10 pts]** A printout listing the 10 most discriminative word features measured by

  $$\max_{word} |\log \Pr(word|label_1) - \log \Pr(word|label_2)|$$

  Since the posterior of each label is formulated by multiplying by the conditional probability $\Pr(word|label_i)$, a word feature should be more discriminative when the ratio $\Pr(word|label_1)/\Pr(word|label_2)$ is large or small and therefore when the absolute difference between $\log \Pr(word|label_1)$ and $\log \Pr(word|label_2)$ is large. In your opinion, are these good word features?
- **[10 pts]** Training and testing accuracy (i.e., two numbers indicating the percentage of correctly classified articles for the training and testing set).
- **[5 pts]** The naïve Bayes model assumes that all word features are independent. Is this a reasonable assumption? Explain briefly.
- **[5 pts]** What could you do to extend the Naïve Bayes model to take into account dependencies between words?
- What if, instead of using ML learning, you were to use MAP learning. Explain what you would need to add and how it would work.