LINFO2364: Mining Patterns in Data
Project 3: Bayesian Networks

Due May 17 2024, 23:55

1 Context

In this project, you will use a Bayesian Network to predict missing values in tabular data. You will first write code for calculating the probability of a cell according to a given neural network; subsequently, you will write code for learning the probabilities of a network, and finally, code for learning the structure of the network.

1.1 Datasets

The datasets we consider in this project are all categorical, but not necessarily binary. Each dataset is provided in the standard CSV format. All datasets that will be provided for training are complete, that is, they do not contain missing data. For the missing value imputation task we will provide data with missing values. The missing values are indicated in these files using the `NaN` string.

Note that while Bayesian Networks can be used for classification, we will not use them for this task. Hence, the datasets provided in this project are unsupervised.

1.2 Tasks

Your final task is to perform missing value imputation using a Bayesian Network learned from training data.

We expect you to do so in the following steps:

1. Bayesian Network inference for missing value imputation: we provide you a template class that already allows you to read a Bayesian Network and represent it in Python code. Your task is to implement a function in this class that allows you to calculate the distribution for one or two variables, given as evidence an assignment to all other variables. This allows you to perform missing value imputation if one or two values in a row are missing.

2. Parameter learning: for a given dataset and a given Bayesian Network over this dataset without parameters, your task is to implement the parameter learning task: the task of filling in the parameters of the network, by filling in the Conditional Probability Tables (CPTs).

3. Structure learning: finally, for a given dataset, your task is to learn a Bayesian Network with both its parameters and its structure.

4. The combination of the previous steps allows you to learn from a given datataset a Bayesian Network, and to use this Bayesian subsequently to predict missing values for instances in which one or two values are missing. You can evaluate the quality of the Bayesian Network by evaluating how well it manages to predict the missing values in a validation dataset.

Your prime objective is to create an inference algorithm that calculates the required probability correctly. Your second objective is to determine an algorithm that imputes missing values as accurately as possible. Efficiency is the last objective.

After implementing the different algorithms, you are expected to write a report summarizing your experiments and the algorithmic choices you have made.

For each of these phases, more details are provided below.

Bayesian Network inference for Missing Value Imputation

A Bayesian Network defines a joint distribution over all variables in the network. This makes it possible to calculate a marginal distribution for any set of variables Y given evidence for variables X in a network, using the equation

$$P(Y = y|X = x) = \frac{P(X = x \wedge Y = y)}{P(X = x)} = \frac{P(X = x \wedge Y = y)}{\sum_{y'} P(X = x \wedge Y = y')}.$$

The more variables there are in Y, the more complex this calculation becomes due to the sum over possible assignments to the variables in Y.

Your task is first to implement the simplest setting, in which $|Y| = 1$, and all other variables are in X; then implement the slightly more complex cersion in which $|Y| = 2$ and all other variables are in X. We will not consider the case that $|Y| > 2$ in the project, although Bayesian Networks could also be used for this case.

To help you with this task, we already provide you some Bayesian Networks in the .BIF format, as well as a Bayesian Network class that can read these files; a full description of this format can be found here:

`https://www.cs.washington.edu/dm/vfml/appendixes/bif.htm`

Roughly speaking, this format first defines the variables in the network, followed by the CPTs. In the resulting class we essentially store a list of variables, with for every variable its domain, its parents, and the probability tables.

Parameter Learning

Assume we are given a Bayesian Network and a dataset for the variables in this Bayesian Network; then we can fill in the CPTs (that is, the parameters) of this Bayesian Network using a maximum likelihood estimation: for instance, for a probability $P(Y = y|X = x)$ we can determine the count $count_D(X = x \wedge Y = y)$ and the count $count_D(X = x)$ and estimate $P(Y = y|X = x)$ with $\frac{count_D(X=x \wedge Y=y)}{count_D(X=x)}$ for the given dataset D.

In this step, you are expected to implement a parameter estimation algorithm, where you can asume that the training data does not have missing values and is provided in the CSV format. The parameter estimation algorithm should be added to the Bayesian Network class, and should take as argument the name of the file that should be used to estimate the parameters. The function should assume that the Bayesian Network class has already been initialized with variables for the columns in the file, except for the CPTs, which should be filled in.

You may add multiple parameter learning functions if you wish to evaluate alternative estimation strategies. For instance, we encourage you to also consider *Laplace smoothing*, in which the estimation is:

$$\frac{count_D(X = x \wedge Y = y) + \alpha}{count_D(X = x) + \alpha K},$$

where $\alpha = 1$ and $K =$ the number of variables are reasonable defaults.

Structure Learning

Finding the structure of a Bayesian Network is a nontrivial task. In this phase you implement an algorithm for this task, where you are free to decide how far you wish to go. The algorithm we expect you to write is a local search algorithm, in which you start from one Bayesian Network and subsequently execute local moves to improve the quality of this network. The parameter learning algorithm can each time be used to fill in the parameters of the network. Subsequently, a scoring function is required to evaluate the quality of the Bayesian Network with its parameters and operations for moving from one Bayesian Network to another are needed.

We expect you to implement the following baselines:

- for the initialization: the baseline is a network in which variable is independent from the other variables; i.e. every variable is a node in the network, and there are no dependencies between the nodes.
- for the local moves: the simplest form of local move is one in which you take one isolated variable (that is, the variable is not dependent on any other variable yet, and no other variable depends on it yet) and you make it dependent on one other variable; this requires considering pairs of variables, and allows to create Bayesian Networks with linear structures (no variable can get two parents by only using this move).

- for the scoring function: for a structure, you can fill in the parameters using the parameter learning algorithm; using these parameters, you can calculate the loglikelihood of the training data given the network:

$$\sum_{e \in D} -\log(P(e)),$$

where $P(e)$ is the probability of the instance e according to the Bayesian Network; a good network could be considered one that gives the training data a large loglikelihood. Among all possible local moves, you can pick the one that scores best; this process can be repeated for a certain number of steps.

All these choices are simplistic choices: the local move does not allow for the creation of complex Bayesian Networks; the scoring function does not take into account the complexity of the network, which may lead to overfitting. You are asked to implement at least one more complex option concerning the local move or the scoring function, or both.

Missing Value Imputation

For the learning strategy implemented before, the resulting Bayesian Network can be used to perform missing value imputation. Hence, the quality of the Bayesian Network can be evaluted in terms of its accuracy in perform missing value imputation. You are asked to evaluate the performance of your Bayesian Network on this missing value imputation task.

2 Directives

- The project will be done by groups of at most two students.
- As in the earlier projects, the project will be done in Python.
- You will have to submit a single `.py` file that contains a main method that receives the following command line arguments:
  - the training dataset to be used for the learning process;
  - the test dataset to be used for evaluation;
  - the file in which the data with missing values should be stored;
  - the file in which the Bayesian Network should be stored.

  The dataset should be stored in the CSV format, similar to the input data; the Bayesian Network should be stored using the BIF format, used also in the input.
- Your report must not exceed 4 pages. It has to contain a short description of your implementation. You have to explain and justify your implementation choices. The report should include a discussion of the parameter learning used, the structure learning used, including the local moves and scoring function used, as well as a discussion of the results.

  Optionally, you can briefly discuss the difficulties that you encountered during the project.

  Your report must contain the number of your group as well as the names and NOMAs of each member. It should be written in correct English. Be precise and concise in your explanations. Do not hesitate to use tables, schemas or graphics to illustrate.
- You will be graded based on several criteria:
  - The correctness and performances of your implementations (8/20).
  - The quality and relevance of your report, justifications and performance analysis. The quality and relevance of your report, justifications and performance analysis. Your source code will also be checked and graded here (e.g. hard-coding the solutions will give a grade of 0) (12/20).
- The deadline for this project is **Friday 17th of May 2024 at 23:55**. Your submission should be uploaded on INGInious (but not graded). The INGInious tasks to upload your source code and report will be made available later.

3 Tips

- Make full use of the time allocated. Do not start the project just before the deadline!
- Do not forget to comment your code.
- Plagiarism is forbidden and will be checked against! Do not share code between groups. If you use online resources, cite them.