Bringing together visual analytics and probabilistic programming languages

Jonas Aaron Gütter
Friedrich Schiller Universität Jena
Matrikelnr 152127
Prof.Dr. Joachim Giesen
M. Sc. Phillip Lucas

8. Oktober 2018

Zusammenfassung

A probabilistic programming language (PPL) provides methods to represent a probabilistic model by using the full power of a general purpose programming language. Thereby it is possible to specify complex models with a comparatively low amount of code. With Uber, Microsoft and DARPA focusing research efforts towards this area, PPLs are likely to play an important role in science and industry in the near future. However in most cases, models built by PPLs lack appropriate ways to be properly visualized, although visualization is an important first step in detecting errors and assessing the overall fitness of a model. This could be resolved by the software Lumen, developed by Philipp Lucas, which provides several visualization methods for statistical models. PPLs are not yet supported by Lumen, and the goal of the master thesis at hand is to change that by implementing an interface between Lumen and a chosen PPL, so that exploring PPL models by visual analytics becomes possible. The thesis will be divided into two main parts, the first part being an overview about how PPLs work and what existing PPLs there are available. Out of these, the most appropriate one will be chosen for the task. The second, more practical part will then document the actual implementation of the interface.

Inhaltsverzeichnis

1 Road Map **3**

2 Rules of Probabilistic Inference **4**
2.1 Chain rule 4
2.2 Total probability rule 4
2.3 Bayes' rule 4

3 Bayesian models **4**

4 What is Probabilistic Programming **5**

5 Existing PPLs **6**
5.1 Stan for python 6
5.2 Pymc3 6
5.3 Edward 7
5.4 Pyro 7

6 Lumen **7**

7 Literatur **7**

1 Road Map

1. Getting started
    - set up Master thesis document
    - Probabilistic Programming Languages
        - play at least with: PyMC3, Stan
        - read the docs, wiki, ...
        - download the libraries
        - reproduce the getting started tutorials
        - –> understand the ideas and how to use it, get a feeling for it
    - theoretic background: Read Bayesian Data Analysis part I, Chapter 1,2 and part II, chapter 6,7
    - Lumen
        - install locally and play with
        - understand main idea of Lumen and what we want to do with it
    - Start filling up your MA thesis document
        - understand and write down in MA thesis the "why & what"
        - describe the background of the work, e.g. summarize PPLs
    - give a short presentation
        - what have you done and learned
        - what do you plan to do?
        - why is it relevant?
        - how do you plan to measure the success?
2. First connection of PPLs to Lumen
    - Start from small, very simple and specific example. Generalize later.
    - Choose PPL to work with
        - work out requirements on PPL
        - work out preferred features of PPL
        - choose a PPL based on these requrements and preferences
    - design wrapper of PPL with Lumen
        - work out requirement and interface
        - identify necessary work on Lumen
        - identify necessary work
    - Connect chosen specific example with lumen
    - Continue to work on your master thesis document!
3. Improve, generalize and clean up the connection of your PPL to Lumen

2 Rules of Probabilistic Inference

There are three rules of probabilistic inference: The chain rule, the total probability rule, and the Bayes' rule. The following explanations are taken from [1].

2.1 Chain rule

The chain rule is used to calculate a joint probability distribution of several variables from local conditional probability distributions of these variables:

$$P(X_1, X_2, ...X_n) = P(X_1)P(X_2|X_1)P(X_3|X_1, X_2)...P(X_n|X_1, X_2, ...X_{n-1})) \tag{1}$$

2.2 Total probability rule

The total probability rule calculates the probability distribution over a subset of variables, also called a marginal distribution, by summing out all the other variables, that is by summing the probability distributions for each combination of values of these variables:

$$P(\boldsymbol{X}|\boldsymbol{Z}) = \sum_{\boldsymbol{y}} P(\boldsymbol{X}, \boldsymbol{Y} = \boldsymbol{y}|\boldsymbol{Z}) \tag{2}$$

2.3 Bayes' rule

Bayes' rule calculates the probability of a cause, given aneffect, by using the prior probability of the cause and the probability of the effect, given the cause. The Bayes' rule can be derived from the chain rule.

$$P(X|Y) = (P(Y|X) * P(X))/P(Y) \tag{3}$$

3 Bayesian models

According to [2], there are 3 basic steps in Bayesian data analysis: Setting up a joint probability model, calculating a posterior distribution, and assessing the model performance

"Classicaläpproach: Evaluate the procedure used to estimate the parameters over the distribution of possible outcome values conditional on the true unknown parameters.

Bayesian approach: Estimate the parameters conditioned on the outcomes.

Bayesian models are about finding a full probability model for a given problem, whereas conventional models only deal with estimation of the most likely parameters. The model parameters β themselves are also considered as random variables which depend on hyperparameters α.

According to [3], the likelihood of a new datapoint can be calculated by integrating the product of the prior likelihood and conditional probability of β over the space of β, as shown in equation 4. This formula can also be derived using the chain rule (I think).

$$p(x_{new}|\boldsymbol{x}, \alpha) = \int p(x_{new}|\beta) * p(\beta|\boldsymbol{x}, \alpha) d\beta \tag{4}$$

4 What is Probabilistic Programming

Modelle spezifizieren/beschreiben

[4]

effizienter in der Beschreibung von Modellen als herkömmliche Programmiersprachen [5]

unifying general purpose programming with probabilistic modeling [6]

A probabilistic reasoning system uses prior knowledge in the form of a probabilistic model to answer a certain query. The particular propertes of the query as well as the prior knowledge are given to an inference algorithm which returns an answer in the form of probabilities. Example is shown in figure 1. Probabilistic Programming is the implementation of a probabilistic reasoning system by using a programming language.

Traditional means for representing models are not always sufficient for probabilistic models. Therefore, probabilistic programming languages were introduced to be able to represent models with the full power of a programming language (http://www.probabilistic-programming.org/wiki/Home).

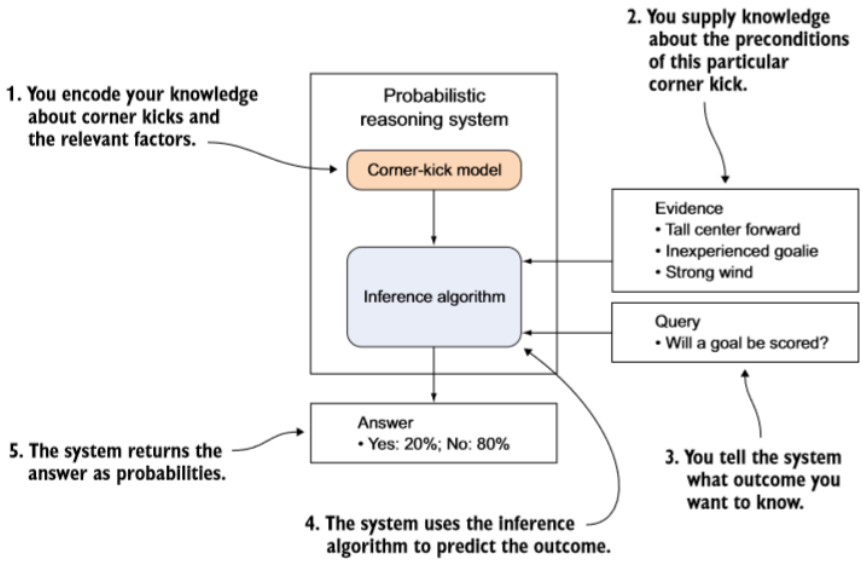

Abbildung 1: General workflow example of a probabilistic reasoning system

5 Existing PPLs

- stan for python: https://pystan.readthedocs.io/en/latest/
- pymc3: https://docs.pymc.io/notebooks/getting_started.html#Case-study-2:-Coal-mining-disasters
- edward: http://edwardlib.org/getting-started
- pyro: http://pyro.ai/

5.1 Stan for python

transformed parameters are parameters which depend on hyperparameters.

5.2 Pymc3

python library

fit Bayesian models, including Markov Chai Monte Carlo (MCMC) and variational inference (VI)

5.3 Edward

5.4 Pyro

6 Lumen

Abbildungsverzeichnis

1 General workflow example of a probabilistic reasoning system. Source: [1] 6

7 Literatur

Literatur

[1] A. Pfeffer, *Practical Probabilistic Programming*. Manning Publications, 2016.

[2] D. B. D. A. V. John B. Carlin, Hal S. Stern and D. B. R. A. Gelman, *Bayesian Data Analysis, 3Rd Edn.* T&F/Crc Press, 2014.

[3] C. Wang and D. M. Blei, "A general method for robust bayesian modeling," *Bayesian Analysis*, jan 2018.

[4] Wikipedia contributors, "Probabilistic programming language — Wikipedia, the free encyclopedia," 2018. [Online; accessed 23-August-2018].

[5] L. Hardesty, "Probabilistic programming does in 50 lines of code what used to take thousands," Apr. 2015. [Online; accessed 23-August-2018].

[6] "probabilistic-programming.org." [Online; accessed 23-August-2018].