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

17. 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

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 Introduction

3 The Bayesian Approach

Statistics in a Bayesian way refers to using data(evidence) to assess and adjust prior knowledge, as well as calculating posterior distributions from the prior and the data to draw inference. Conditional probabilities play a big role in Bayesian statistics.

According to [1], there are 3 basic steps in Bayesian data analysis: Setting up a joint probability model, calculating a posterior distribution, and assessing the model performance
The posterior distribution is a compromise between the prior distribution and the sample distribution, with the prior distribution becoming less important as the sample size increases [1]. posterior distributions can be described numerically by mean, median, modes. The uncertainty can be described by quantiles. The highest posterior density region is also a possibility, it is the area that for example contains 95% of the posterior probability density, like quantiles, but has the additional constraint that the density on each point inside the area has to be bigger than the density on any point outside the area. It does not have to be one single connected area [1].
Often it is about which distribution class should be chosen for the prior and the likelihood. There is (in practice) an important separation between conjugate and nonconjugate distributions. Conjugate distributions are more convenient since posterior and prior distributions have the same form.
Standard, convenient distributions for single-parameter models: normal, binomial, Poisson, exponential. Those can also be combined to represent more complex distributions. For different classes of sample distributions there are corresponding conjugate prior distributions which lead in turn to posterior distributions of the same form. [1]
What is the difference between Bayesian inference and Bayesian data analysis in [1]?

3.1 Choosing an appropriate prior distribution

There two interpretations of prior distributions. The *population* interpretation, where the prior distribution is thought of as a population of possible parameters, from where the current parameter is drawn. This, as far as I understand, requires the range of possible values to be known, e.g from past experience. On the other hand, the *state of knowledge* interpretation looks at the prior distribution as an expression of the user's knowledge or uncertainty, so that the assumption is plausible, that the real parameter value is taken from a random realization of the prior distribution.

Laplace's principle of insufficient reason: When nothing is known about the prior distribution, it is assumed that a uniform distribution over all possible values is most appropriate. One difficulty of this principle is the question, on which parametrization should it apply? E.g. applying it to $p(x)$ gives a non-uniform distribution when looking at $p(x^2)$ and the other way round.

[1]

The parameters of the prior distributions are called hyperparameters. The property that prior and posterior distribution are of the parametric form (e.g., both are a beta distribution) (for a given likelihood distribution), is called conjugacy. Conjugate prior distributions have the advantages of being computationally convenient and being intertpretable as additional data.
For the posterior distribution, a normal distribution is often assumed.
models can be chosen for mathematical convenience. One could estimate hyperparameters from the data in some cases. This is a bit of a circular reasoning, but apparently it is appropriate for [1]. When we have lots of knowledge about the parameter already, it makes sense to choose an $i$nformative prior, which has a big influence on the posterior distribution. If there is no sufficient data to estimate a prior, it is desirable to choose a prior that is $n$oninformative, meaning that it will contribute very little to the posterior distribution ('let the data speak for itself'). Besides that there is also the *weakly informative* prior distribution. This kind of prior does affect the posterior distribution in terms of regularization (e.g. it prevents extreme outliers), but it does not contain any further special knowledge about the parameter.
A prior distribution is called $p$roper if it does not depend on data and sums to 1 [1]. Proper distribution can be normalized. The uniform prior for example is $i$mproper, since it can't be normalized.
Jeffrey's approach to find noninformative prior distributions: A problem of the uniform distribution is, that for different parameterizations(choice of parameter) , distributions for the same parameter can be contradictory (see https://eventuallyalmosteverywhere.wordpress.c inference-and-the-jeffreys-prior/). Jeffreys' prior has the property of being invariant of the parameterization.

3.2 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 [2].

3.2.1 Chain rule

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

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

3.2.2 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:

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

3.2.3 Bayes' rule

Bayes' rule calculates the probability of a cause, given an effect, 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.

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

3.3 Comparison to frequentist approach

Classical approach: 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. [1]
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.

3.4 Drawing inference

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).
Making predictions for a new data points requires both the probability distribution of the new data point, given a parameter value, as well as the probability distribution of the parameter, given the old data points.
If it is not possible to perform calculations directly from the posterior distribution (e.g. if there is no closed form but only a discrete approximation of the posterior distribution), one can simulate data points from the posterior distribution instead and perform calculations on them [1].

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

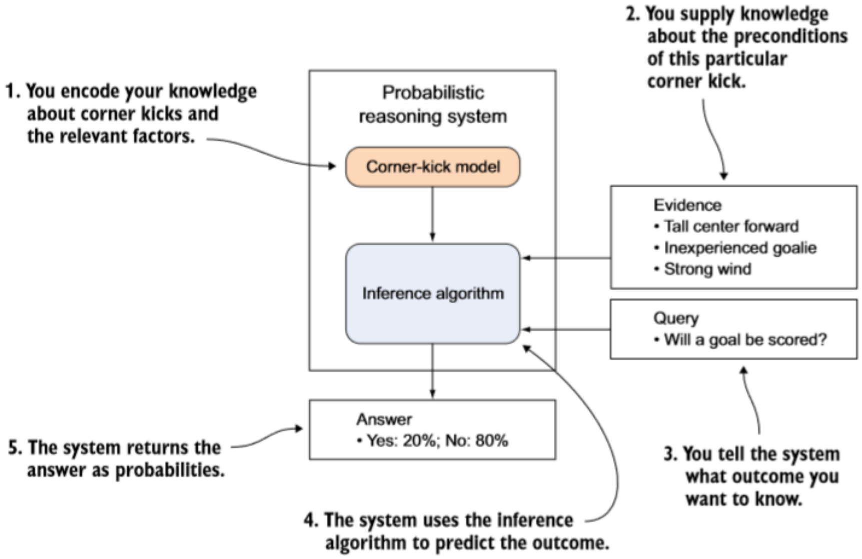
1. You encode your knowledge about corner kicks and the relevant factors.
Probabilistic reasoning system
Corner-kick model
Inference algorithm
2. You supply knowledge about the preconditions of this particular corner kick.
Evidence
• Tall center forward
• Inexperienced goalie
• Strong wind
Query
• Will a goal be scored?
3. You tell the system what outcome you want to know.
Answer
• Yes: 20%; No: 80%
5. The system returns the answer as probabilities.
4. The system uses the inference algorithm to predict the outcome.

Abbildung 1: General workflow example of a probabilistic reasoning system

4 Probabilistic Programming

stochastic data types (–> probability distributions) make it easier to perform bayesian data analysis

4.1 What are Probabilistic Programming Languages

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).

5 Lumen

5.1 Functionality

5.2 Requirements for a PPL

6 Comparing Different Probabilistic Programming Languages

- 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/

6.1 Stan for python

transformed parameters are parameters which depend on hyperparameters.

6.2 Pymc3

python library

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

6.3 Edward

6.4 Pyro

6.5 Choose the PPL for the task at hand

7 Practical implementation

8 Fallbeispiele

Abbildungsverzeichnis

9 Literatur

Literatur

[1] 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.

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

[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].