Chapter 9
The Automatic Statistician

Christian Steinruecken, Emma Smith, David Janz, James Lloyd, and Zoubin Ghahramani

Abstract The Automatic Statistician project aims to automate data science, producing predictions and human-readable reports from raw datasets with minimal human intervention. Alongside basic graphs and statistics, the generated reports contain a curation of high-level insights about the dataset that are obtained from (1) an automated construction of models for the dataset, (2) a comparison of these models, and (3) a software component that turns these results into natural language descriptions. This chapter describes the common architecture of such Automatic Statistician systems, and discusses some of the design decisions and technical challenges.

9.1 Introduction

Machine Learning (ML) and data science are closely related fields of research, that are focused on the development of algorithms for automated learning from data. These algorithms also underpin many of the recent advances in artificial intelligence (AI), which have had a tremendous impact in industry, ushering in a new golden age of AI. However, many of the current approaches to machine learning, data science, and AI, suffer from a set of important but related limitations.

Firstly, many of the approaches used are complicated black-boxes that are difficult to interpret, understand, debug, and trust. This lack of interpretability hampers the deployment of ML systems. For example, consider the major legal, technical and ethical consequences of using an uninterpretable black-box system that arrives at a prediction or decision related to a medical condition, a criminal justice setting, or in a self-driving car. The realisation that black-box ML methods are severely limited in such settings has led to major efforts to develop "explainable AI", and systems that offer interpretability, trust, and transparency.

C. Steinruecken (✉) · E. Smith · D. Janz · J. Lloyd · Z. Ghahramani
Department of Engineering, University of Cambridge, Cambridge, UK
e-mail: tcs27@cam.ac.uk

Secondly, the development of ML systems has turned into a cottage industry where ML experts tackle problems by hand-designing solutions that often reflect a set of ad-hoc manual decisions, or the preferences and biases of the expert. It is ironic that machine learning, a field dedicated to building systems that automatically learn from data, is so dependent on human experts and manual tuning of models and learning algorithms. Manual search over possible models and methods can result in solutions that are sub-optimal across any number of metrics. Moreover, the tremendous imbalance between the supply of experts and the demand for data science and ML solutions is likely resulting in many missed opportunities for applications that could have a major benefit for society.

The vision of the Automatic Statistician is to automate many aspects of data analysis, model discovery, and explanation. In a sense, the goal is to develop an *AI for data science* – a system that can reason about patterns in data and explain them to the user. Ideally, given some raw data, such a system should be able to:

- automate the process of feature selection and transformation,
- deal with the messiness of real data, including missing values, outliers, and different types and encodings of variables,
- search over a large space of models so as to automatically discover a good model that captures any reliable patterns in the data,
- find such a model while avoiding both overfitting and underfitting,
- explain the patterns that have been found to the user, ideally by having a conversation with the user about the data, and
- do all of this in a manner that is efficient and robust with respect to constraints on compute time, memory, amount of data, and other relevant resources.

While this agenda is obviously a very ambitious one, the work to date on the Automatic Statistician project has made progress on many of the above desiderata. In particular, the ability to discover plausible models from data and to explain these discoveries in plain English, is one of the distinguishing features of the Automatic Statistician [18]. Such a feature could be useful to almost any field or endeavour that is reliant on extracting knowledge from data.

In contrast to much of the machine learning literature that has been focused on extracting increasing performance improvements on pattern recognition problems (using techniques such as kernel methods, random forests, or deep learning), the Automatic Statistician needs to build models that are composed of interpretable components, and to have a principled way of representing uncertainty about model structures given data. It also needs to be able to give reasonable answers not just for big data sets, but also for small ones.
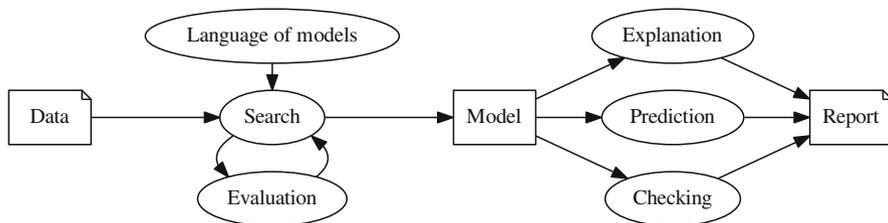
Fig. 9.1 A simplified flow diagram outlining the operation of a report-writing Automatic Statistician. Models for the data are automatically constructed (from the open-ended language of models), and evaluated on the data. This evaluation is done in a way that allows models to be compared to each other. The best models are then inspected to produce a report. Each model can be used to make extrapolations or predictions from the data, and the construction blue-print of the model can be turned into a human-readable description. For some models, it is also possible to generate *model criticism*, and report on where the modelling assumptions do not match the data well

9.2 Basic Anatomy of an Automatic Statistician

At the heart of the Automatic Statistician is the idea that a good solution to the above challenges can be obtained by working in the framework of *model-based machine learning* [2, 9]. In model-based ML, the basic idea is that probabilistic models are explanations for patterns in data, and that the probabilistic framework (or Bayesian Occam's razor) can be used to discover models that avoid both overfitting and underfitting [21]. Bayesian approaches provide an elegant way of trading off the complexity of the model and the complexity of the data, and probabilistic models are compositional and interpretable as described previously. Moreover, the model-based philosophy maintains that tasks such as data pre-processing and transformation are all parts of the model and should ideally all be conducted at once [35].

An Automatic Statistician contains the following key ingredients:

1. **An open-ended language of models** – expressive enough to capture real-world phenomena, and to allow applying the techniques used by human statisticians and data scientists.
2. **A search procedure** to efficiently explore the language of models.
3. **A principled method of evaluating models**, trading off complexity, fit to data, and resource usage.
4. **A procedure to automatically explain the models**, making the assumptions of the models explicit in a way that is simultaneously accurate and intelligible to non-experts.

Fig. 9.1 shows a high-level overview of how these components could be used to produce a basic version of a report-writing Automatic Statistician.

As will be discussed later in this chapter, it is possible to build Automatic Statistician systems that exchange ingredient (4) for procedures that produce other

desirable outputs, for example raw predictions or decisions. In such cases, the language, search, and evaluation components may be modified appropriately to prioritise the chosen objective.

9.2.1 Related Work

Important earlier work includes statistical expert systems [11, 37], and equation learning [26, 27]. The *Robot Scientist* [16] integrates machine learning and scientific discovery in a closed loop with an experimental platform in microbiology to automate the design and execution of new experiments. *Auto-WEKA* [17, 33] and *Auto-sklearn* [6] are projects that automate learning classifiers, making heavy use of Bayesian optimisation techniques. Efforts to automate the application of machine learning methods to data have recently gained momentum, and may ultimately result in practical AI systems for data science.
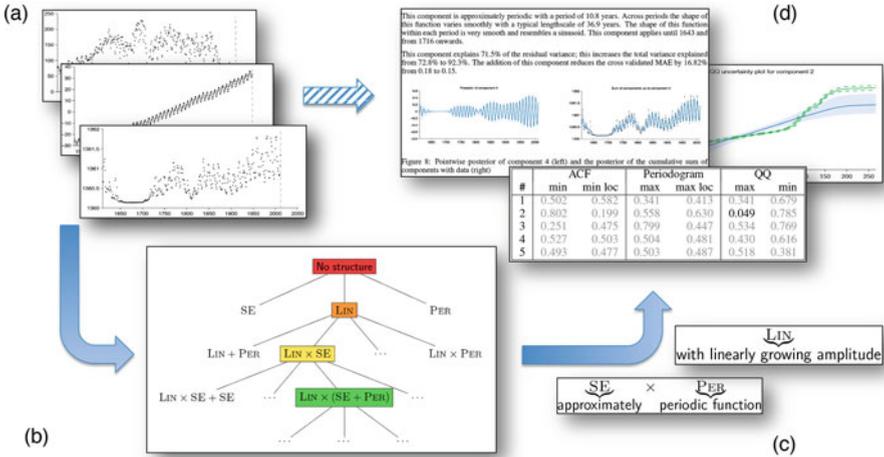
9.3 An Automatic Statistician for Time Series Data

Automatic Statistician systems can be defined for a variety of different objectives, and can be based on different underlying model families. We'll start by describing one such system, and discuss the wider taxonomy later, with notes on common design elements and general architecture.

An early Automatic Statistician for one-dimensional regression tasks was described by Lloyd et al. [18]. Their system, called *Automatic Bayesian Covariance Discovery (ABCD)*, uses an open-ended language of Gaussian process models through a compositional grammar over kernels. A Gaussian process (GP) defines a distribution over functions, and the parameters of the GP – its mean and its kernel – determine the properties of the functions [25]. There is a broad choice of available kernels that induce function distributions with particular properties; for example distributions over functions that are linear, polynomial, periodic, or uncorrelated noise. A pictorial overview of this system is shown in Fig. 9.2.

9.3.1 The Grammar over Kernels

As mentioned above, a grammar over GP kernels makes it possible to represent many interesting properties of functions, and gives a systematic way of constructing distributions over such functions. This grammar over kernels is compositional: it comprises a set of fixed *base kernels*, and *kernel operators* that make it possible to compose new kernels from existing ones. This grammar was carefully chosen to be *interpretable*: each expression in the grammar defines a kernel that can be described with a simple but descriptive set of words in human language.

Fig. 9.2 A flow diagram describing a report-writing Automatic Statistician for time-series data. (**a**) The input to the system is data, in this case represented as time series. (**b**) The system searches over a grammar of models to discover a good interpretation of the data, using Bayesian inference to score models. (**c**) Components of the model discovered are translated into English phrases. (**d**) The end result is a report with text, figures and tables, describing in detail what has been inferred about the data, including a section on model checking and criticism [8, 20]

The base kernels in the grammar are: C (constant), LIN (linear), SE (squared exponential), PER (periodic), and WN (white noise). The kernel operators are: + (addition), × (multiplication), and CP (a change point operator), defined as follows:

$$(k_1 + k_2)(x, x') = k_1(x, x') + k_2(x, x')$$

$$(k_1 \times k_2)(x, x') = k_1(x, x') \times k_2(x, x')$$

$$\mathrm{CP}(k_1, k_2)(x, x') = k_1(x, x')\,\sigma(x)\,\sigma(x') \;+\; k_2(x, x')\,(1 - \sigma(x))\,(1 - \sigma(x'))$$

where $\sigma(x) = \frac{1}{2}\left(1 + \tanh \frac{l-x}{s}\right)$ is a sigmoidal function, and l and s are parameters of the change point. The base kernels can be arbitrarily combined using the above operators to produce new kernels.

The infinite space of kernels defined by this grammar allows a large class of interesting distributions over functions to be searched, evaluated, and described in an automated way. This type of grammar was first described in [10] for matrix factorization problems, and then refined in [5] and [18] for GP models.

9.3.2   The Search and Evaluation Procedure

ABCD performs a greedy search over the space of models (as defined by the grammar). The kernel parameters of each proposed model are optimised by a

conjugate-gradient method; the model with optimised parameters is then evaluated using the Bayesian Information Criterion [29]:

$$\text{BIC}\,(M) = -2\log p\,(D \mid M) \, + \, |M|\log N \qquad (9.1)$$

where M is the optimised model, $p\,(D \mid M)$ is the marginal likelihood of the model integrating out the latent GP function, $|M|$ is the number of kernel parameters in M, and N is the size of the dataset. The Bayesian Information Criterion trades off model complexity and fit to the data, and approximates the full marginal likelihood (which integrates out latent functions and hyperparameters).

The best-scoring model in each round is used to construct new proposed models, either by: (1) expanding the kernel with production rules from the grammar, such as introducing a sum, product, or change point; or (2) mutating the kernel by swapping out a base kernel for a different one. The new set of proposed kernels is then evaluated in the next round. It is possible with the above rules that a kernel expression gets proposed several times, but a well-implemented system will keep records and only ever evaluate each expression once. The search and evaluation procedure stops either when the score of all newly proposed models is worse than the best model from the previous round, or when a pre-defined search depth is exceeded.

This greedy search procedure is not guaranteed to find the best model in the language for any given dataset: a better model might be hiding in one of the subtrees that weren't expanded out. Finding the globally best model isn't usually essential, as long as a good interpretable models is found in a reasonable amount of time. There are other ways of conducting the search and evaluation of models. For example, Malkomes et al. [22] describe a kernel search procedure based on Bayesian optimisation. Janz et al. [14] implemented a kernel search method using particle filtering and Hamiltonian Monte Carlo.

9.3.3 Generating Descriptions in Natural Language

When the search procedure terminates, it produces a list of kernel expressions and their scores on the dataset. The expression with the best score is then used to generate a natural-language description. To convert a kernel to a description in natural language, the kernel is first converted to a canonical form, using the following process:

1. Nested sums and products are flattened into a sum of products form.
2. Some products of kernels can be simplified into base kernels with modified parameters, for example: SE $\times$ SE $\rightarrow$ SE*, C $\times$ k $\rightarrow$ k^* for any k, and WN $\times$ k $\rightarrow$ WN* for any $k \in$ {C, SE, WN, PER}.

After applying these rules, the kernel expression is a sum of product terms, where each product term has the following canonical form:

$$k \times \prod_m \mathrm{LIN}^{(m)} \times \prod_n \sigma^{(n)} \tag{9.2}$$

where $\sigma(x, x') = \sigma(x)\,\sigma(x')$ is a product of two sigmoid functions, and k has one of the following forms: 1, WN, C, SE, $\prod_j \mathrm{PER}^{(j)}$, or SE $\times \prod_j \mathrm{PER}^{(j)}$. The notation $\prod_j k^{(j)}$ stands for products of kernels, each with separate parameters.

In this canonical form, the kernel is a sum of products, and the number of terms in the sum is described first: "The structure search algorithm has identified N additive components in the data." This sentence is then followed by a description of each additive component (i.e. each product in the sum), using the following algorithm:

1. Choose *one* of the kernels in the product to be the noun descriptor. A heuristic recommended by Lloyd et al. [18] is to pick according to the following preference: PER > {C, SE, WN} > $\prod_j \mathrm{LIN}^{(j)}$ > $\prod_j \sigma^{(j)}$, where PER is the most preferred.
2. Convert the chosen kernel type to a string using this table:

WN	"uncorrelated noise"	SE	"smooth function"
PER	"periodic function"	LIN	"linear function"
C	"constant"	$\prod_j \mathrm{LIN}^{(j)}$	"polynomial"

3. The other kernels in the product are converted to *post-modifier* expressions that are appended to the noun descriptor. The post modifiers are converted using this table:

SE	"whose shape changes smoothly"
PER	"modulated by a periodic function"
LIN	"with linearly varying amplitude"
$\prod_j \mathrm{LIN}^{(j)}$	"with polynomially varying amplitude"
$\prod_j \sigma^{(j)}$	"which applies from / until [changepoint]"

4. Further refinements to the description are possible, including insights from kernel parameters, or extra information calculated from the data. Some of these refinements are described in [18].

More details on the translation of kernel expressions to natural language can be found in [18] and [19]. An example extract from a generated report is shown in Fig. 9.3.

This component is approximately periodic with a period of 10.8 years. Across periods the shape of this function varies smoothly with a typical lengthscale of 36.9 years. The shape of this function within each period is very smooth and resembles a sinusoid. This component applies until 1643 and from 1716 onwards.

This component explains 71.5% of the residual variance; this increases the total variance explained from 72.8% to 92.3%. The addition of this component reduces the cross validated MAE by 16.82% from 0.18 to 0.15.
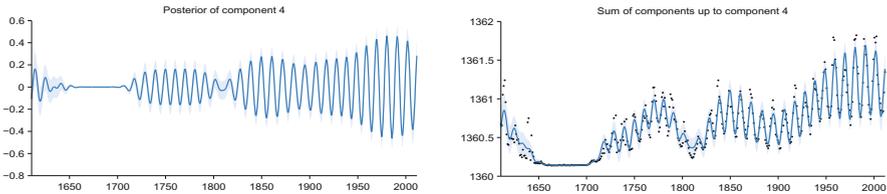
Fig. 9.3 Extract from an automatically generated report that describes the model components discovered by ABCD. This part of the report isolates and describes the approximately 11-year sunspot cycle, also noting its disappearance during the sixteenth century, a time period known as the Maunder minimum. (This figure is reproduced from [18])

9.3.4 Comparison with Humans

An interesting question to consider is to what extent predictions made by an Automated Statistician (such as the ABCD algorithm) are human-like, and how they compare to predictions made with other methods that are also based on Gaussian processes. To answer that question, Schulz et al. [28] presented participants with the task of extrapolating from a given set of data, and choosing a preferred extrapolation from a given set. The results were encouraging for composite kernel search in two ways: Firstly, the participants preferred the extrapolations made by ABCD over those made with Spectral Kernels [36], and over those made with a simple RBF (radial basis function) kernel. Secondly, when human participants were asked to extrapolate the data themselves, their predictions were most similar to those given by ABCD's composite search procedure.

One of the design goals of a report-writing Automatic Statistician is the ability to explain its findings in terms that are understandable by humans. The system described earlier restricts itself to a space of models that can be explained in human language using simple terms, even though this design choice may come at the cost of predictive accuracy. In general, it is not straight-forward to measure the interpretability of machine learning systems; one possible framework is suggested by Doshi-Velez and Kim [4]. We note in passing that not all machine learning systems require such functionality. For example, when the results of a system have little impact on society, especially in terms of social norms and interactions, it is acceptable to optimise for performance or accuracy instead (e.g. recognising post codes for automatic mail sorting).

9.4   Other Automatic Statistician Systems

The ability to generate human-readable reports is perhaps one of the distinguishing features of Automatic Statistician systems. But, as mentioned earlier, software of this nature can serve other purposes as well. For example, users might be interested in raw predictions from the data (with or without explanations), or they might want the system to make data-driven decisions directly on their behalf.

Also, it is possible to build Automatic Statistician systems for model families that are different from Gaussian processes or grammars. For example, we built Automated Statistician systems for regression [5, 18], classification [12, 23], univariate and multivariate data; systems based on various different model classes, and systems with and without intelligent resource control. This section discusses some of the design elements that are shared across many Automatic Statistician systems.

9.4.1   Core Components

One of the key tasks that an Automatic Statistician has to perform is to select, evaluate, and compare models. These types of task can be run concurrently, but they have interdependencies. For example, the evaluation of one set of models might influence the selection of the next set of models.

Most generally, the **selection strategy** component in our system is responsible for choosing models to evaluate: it might choose from a fixed or open-ended family of models, or it might generate and refine models based on the evaluation and comparison of previously chosen models. Sometimes, the *types* of the variables in the dataset (whether inferred from the data or annotated by the user) influence which models might be chosen by the selection strategy. For example, one might want to distinguish continuous and discrete data, and to use different treatments for categorical and ordinal data.

The **model evaluation** task trains a given model on part of the user-supplied dataset, and then produces a score by testing the model on held-out data. Some models do not require a separate training phase and can produce a log-likelihood for the entire dataset directly. Model evaluation is probably one of the most important tasks to parallelise: at any given time, multiple selected models can be evaluated simultaneously, on multiple CPUs or even multiple computers.

The **report curator** component is the piece of software that decides which results to include in the final report. For example, it might include sections that describe the best fitting models, along with extrapolations, graphs, or data tables. Depending on the evaluation results, the report curator might choose to include additional material, such as data falsification/model criticism sections, recommendations, or a summary. In some systems the deliverable might be something other than a report, such as raw predictions, parameter settings, or model source code.

In interactive systems, a **data loading stage** provides an instant summary about the uploaded dataset, and allows the user to correct any assumptions about the format of the data. The user can make type annotations, remove columns from the dataset, choose an output variable (e.g. for classification), and specify the analyses that should be run.

9.4.2 Design Challenges

9.4.2.1 User Interaction

While the aim of an Automatic Statistician is to automate *all* aspects of data handling (from low-level tasks such as formatting and clean-up, to high-level tasks such as model construction, evaluation, and criticism), it is also useful to give users the option to interact with the system and influence the choices it makes. For example, users might want to specify which parts or which aspects of the data they are interested in, and which parts can be ignored. Some users might want to choose the family of models that the system will consider in the model construction or evaluation phase. Finally, the system may want to engage in a dialogue with the user to explore or explain what it found in the data. Such interactivity needs to be supported by the underlying system.

9.4.2.2 Missing and Messy Data

A common problem with real-world datasets is that they may have missing or corrupt entries, unit or formatting inconsistencies, or other kinds of defects. These kinds of defects may require some pre-processing of the data, and while many decisions could be made automatically, some might benefit from interaction with the user. Good models can handle missing data directly, and as long as the missing data is detected correctly by the data loading stage, everything should be fine. But there are some data models that cannot handle missing data natively. In such cases, it might be useful to perform *data imputation* to feed these models a version of the dataset that has the missing values filled in. This imputation task itself is performed by a model that is trained on the data. Examples of such techniques include e.g. MissForest [31], MissPaLasso [30], mice [3], KNNimpute [34], and Bayesian approaches [1, 7].

9.4.2.3 Resource Allocation

Another important aspect of an Automatic Statistician is *resource usage*. For example, a user might only have a limited number of CPU cores available, or might be interested to get the best possible report within a fixed time limit,

e.g. before a given deadline. To make good model selection and evaluation choices, an intelligent system might take into account such resource constraints. The ability to do so will affect the overall usability of the system.

Even when there are no direct constraints on computation time, CPU cores, or memory usage, an intelligent system might benefit from allocating resources to models whose evaluation is promising for the chosen deliverable. Such functionality can be implemented for models that support some form of gradual evaluation, for example by training incrementally on increasingly large subsets of the dataset. One of our systems used a variant of Freeze-thaw Bayesian optimisation [32] for this purpose.

9.5 Conclusion

Our society has entered an era of abundant data. Analysis and exploration of the data is essential for harnessing the benefits of this growing resource. Unfortunately, the growth of data currently outpaces our ability to analyse it, especially because this task still largely rests on human experts. But many aspects of machine learning and data analysis can be automated, and one guiding principle in pursuit of this goal is to "apply machine learning to itself".

The Automatic Statistician project aims to automate data science by taking care of all aspects of data analysis, from data pre-processing, modelling and evaluation, to the generation of useful and transparent results. All these tasks should be performed in a way that requires little user expertise, minimises the amount of user interaction, and makes intelligent and controlled use of computational resources.

While this aim is ambitious, and a lot of the work still needs to happen, encouraging progress has been made towards the creation of such automated systems. Multiple Automatic Statistician systems have been built, each with slight differences in purpose and underlying technology, but they all share the same intent and much of the same design philosophy. We hope that the creation of such instruments will bring the ability to gain insights from data to a larger group of people, and help empower society to make great use of our data resources.

Bibliography

1. Allingham, J.U.: Unsupervised automatic dataset repair. Master's thesis in advanced computer science, Computer Laboratory, University of Cambridge (2018)
2. Bishop, C.M.: Pattern recognition and machine learning. Information science and statistics, Springer (2006)
3. van Buuren, S., Groothuis-Oudshoorn, K.: `mice`: Multivariate imputation by chained equations in R. Journal of Statistical Software 45(3) (2011)
4. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning (Mar 2017), http://arxiv.org/abs/1702.08608

5. Duvenaud, D., Lloyd, J.R., Grosse, R., Tenenbaum, J.B., Ghahramani, Z.: Structure discovery in nonparametric regression through compositional kernel search. In: Proceedings of the 30th International Conference on Machine Learning (Jun 2013)

6. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28, pp. 2962–2970. Curran Associates, Inc. (2015)

7. Garriga Alonso, A.: Probability density imputation of missing data with Gaussian Mixture Models. MSc thesis, University of Oxford (2017)

8. Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., Rubin, D.B.: Bayesian Data Analysis, Third Edition. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis (2013)

9. Ghahramani, Z.: Probabilistic machine learning and artificial intelligence. Nature 521, 452–459 (2015)

10. Grosse, R.B., Salakhutdinov, R., Tenenbaum, J.B.: Exploiting compositionality to explore a large space of model structures. In: Uncertainty in Artificial Intelligence (2012)

11. Hand, D.J.: Patterns in statistical strategy. In: Gale, W.A. (ed.) Artificial intelligence and statistics (1986)

12. He, Q.: The Automatic Statistician for Classification. Master's thesis, Department of Engineering, University of Cambridge (May 2016)

13. Hwang, Y., Tong, A., Choi, J.: Automatic construction of nonparametric relational regression models for multiple time series. In: Balcan, M.F., Weinberger, K.Q. (eds.) ICML 2016: Proceedings of the 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 48, pp. 3030–3039. PLMR (2016)

14. Janz, D., Paige, B., Rainforth, T., van de Meent, J.W., Wood, F.: Probabilistic structure discovery in time series data (2016), https://arxiv.org/abs/1611.06863

15. Kim, H., Teh, Y.W.: Scaling up the Automatic Statistician: Scalable structure discovery using Gaussian processes. In: Storkey, A., Perez-Cruz, F. (eds.) Proceedings of the 21st International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 84, pp. 575–584. PLMR (2018)

16. King, R.D., Whelan, K.E., Jones, F.M., Reiser, P.G.K., Bryant, C.H., Muggleton, S.H., Kell, D.B., Oliver, S.G.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature 427(6971), 247–252 (2004)

17. Kotthoff, L., Thornton, C., Hoos, H.H., Hutter, F., Leyton-Brown, K.: Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. Journal of Machine Learning Research 18(25), 1–5 (2017)

18. Lloyd, J.R., Duvenaud, D., Grosse, R., Tenenbaum, J.B., Ghahramani, Z.: Automatic construction and natural-language description of nonparametric regression models. In: Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI-14) (2014)

19. Lloyd, J.R.: Representation, learning, description and criticism of probabilistic models with applications to networks, functions and relational data. Ph.D. thesis, Department of Engineering, University of Cambridge (Dec 2014)

20. Lloyd, J.R., Ghahramani, Z.: Statistical model criticism using kernel two sample tests. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28. pp. 829–837. Curran Associates, Inc. (2015)

21. MacKay, D.J.C.: Bayesian interpolation. Neural Computation 4(3), 415–447 (1992), see [24] for additional discussion and illustration.

22. Malkomes, G., Schaff, C., Garnett, R.: Bayesian optimization for automated model selection. In: Lee, D.D., Sugiyama, M., von Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 2900–2908. Curran Associates, Inc. (2016)

23. Mrkšić, N.: Kernel Structure Discovery for Gaussian Process Classification. Master's thesis, Computer Laboratory, University of Cambridge (Jun 2014)

24. Murray, I., Ghahramani, Z.: A note on the evidence and Bayesian Occam's razor. Tech. Rep. GCNU-TR 2005-003, Gatsby Computational Neuroscience Unit, University College London (2005)

25. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press (2006), http://www.gaussianprocess.org/gpml/
26. Schmidt, M., Lipson, H.: Distilling free-form natural laws from experimental data. Science 324(5923), 81–85 (2009)
27. Schmidt, M., Lipson, H.: Symbolic regression of implicit equations. In: Riolo, R., O'Reilly, U.M., McConaghy, T. (eds.) Genetic Programming Theory and Practice VII, pp. 73–85. Springer, Boston, MA (2010)
28. Schulz, E., Tenenbaum, J., Duvenaud, D.K., Speekenbrink, M., Gershman, S.J.: Probing the compositionality of intuitive functions. In: Lee, D.D., Sugiyama, M., von Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 3729–3737. Curran Associates, Inc. (2016)
29. Schwarz, G.: Estimating the dimension of a model. The Annals of Statistics 6(2), 461–464 (1978)
30. Städler, N., Stekhoven, D.J., Bühlmann, P.: Pattern alternating maximization algorithm for missing data in high-dimensional problems. Journal of Machine Learning Research 15, 1903–1928 (Jun 2014)
31. Stekhoven, D.J., Bühlmann, P.: MissForest – non-parametric missing value imputation for mixed-type data. Bioinformatics 28(1), 112–118 (Nov 2011)
32. Swersky, K., Snoek, J., Adams, R.P.: Freeze-thaw Bayesian optimization (Jun 2014), http://arxiv.org/abs/1406.3896
33. Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 847–855. KDD '13, ACM, New York, NY, USA (2013)
34. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for DNA microarrays. Bioinformatics pp. 520–525 (Jun 2001)
35. Valera, I., Ghahramani, Z.: Automatic discovery of the statistical types of variables in a dataset. In: Precup, D., Teh, Y.W. (eds.) ICML 2017: Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 3521–3529. PLMR (2017)
36. Wilson, A.G., Adams, R.P.: Gaussian process kernels for pattern discovery and extrapolation. In: Dasgupta, S., McAllester, D. (eds.) ICML 2013: Proceedings of the 30th International Conference on Machine Learning. JLMR Proceedings, vol. 28, pp. 1067–1075. JLMR.org (Jun 2013)
37. Wolstenholme, D.E., O'Brien, C.M., Nelder, J.A.: GLIMPSE: a knowledge-based front end for statistical analysis. Knowledge-Based Systems 1(3), 173–178 (1988)