Autokit: automatic machine learning via representation and model search

Tadej Štajner, Jožef Stefan International Postgraduate School TADEJ@TDJ.SI
Jamova cesta 39, 1000 Ljubljana, Slovenia

Abstract

This paper describes an approach for automatic machine learning that explores both the spaces of preprocessing components, as well as prediction models, and defining the search space as a function of dataset properties. We focus on using kernel approximations as a performance-efficient way to include non-linearities in the preprocessing layer, in combination with stochastic gradient descent linear models and decision tree ensemble models.

Keywords: Hyper-parameter optimization, kernel approximation, automatic machine learning

1. Introduction

We describe the AUTOKIT library, a system for automatically guiding the model selection process. It is our the submission to the AutoML 2015 (Guyon et al.) challenge, a competition where such systems compete on modeling previously unobserved problems. The method is based on the HYPEROPT (Bergstra et al., 2013), SCIKIT-LEARN (Pedregosa et al., 2011) and HYPEROPT-SKLEARN(Komer et al., 2014) libraries to pose the automatic machine learning problem as a hyperparameter optimization problem. The approach extends the hyperopt-sklearn model to include additional learning model selection that is able to determine admissible learning models given the problem description.

We focus on generating high-performing models through two strategies: either via learning decision tree ensembles, or applying a kernel approximation step in preprocessing. While decision trees are good at representing complex relationships among variables, kernel approximations are an efficient way to describe the kernel space among the data points. These approaches are complementary: decision trees express relationships among variable values, kernel maps quanitfy relationships among data points, and Naive Bayes for the cases where the independence assumption holds.

We also try to reduce the amount of processing, so that many configuration can be evaluated: if a certain expressive power can be reached with a faster model, we use that. For instance, we use kernel approximations instead of a full kernelized SVM, and online solvers of many popular classification and regression models.

2. Model

The sampling from the model selection space takes place in two steps: preprocessing layer selection, and learning model selection. If not expressed otherwise, we use the paramterer space definitions from HYPEROPT-SKLEARN.

Round	Rank	Set 1	Set 2	Set 3	Set 4	Set 5	Duration (s)
1	4.20 (3)	0.431 (9)	0.621 (3)	0.747 (1)	0.555 (2)	0.875 (6)	2727 (61)

Table 1: Results on the ChaLearn AutoML Challenge - Round 1 new data release

For preprocessing, provide an uniform prior distribution of one of several options. The first option is **no preprocessing**, directly learning on the data. Second, **normalization** of the feature values of a sample with its l_1 or l_2 norm, each having equal probability. Third, **scaling** features to have a standard deviation of one. Fourth, **Nyström kernel approximation**Yang et al. (2012), computing an approximate kernel map by approximating the full kernel matrix K by first sampling m examples and then construct a low rank decomposition, that can be used to construct an m-dimensional representation of the input. Learning a linear machine is equivalent to using the kernel κ directly with a SVM regressor that predicts directly.

The parameter space is also a function of the dimensionality of the dataset: the prior of the number of components is sampled on a log-uniform distribution from 4 to $\min(30, \frac{\#\text{features}}{4})$. The kernel chosen as either sigmoid, RBF or polynomial, sampling the γ kernel parameter. Fifth, **PCA** can be applied, but only for dense data. The number of components is selected the same way as with Nyström kernel approximation. As a prior, we also uniformly pick whether to whiten the data

For classification problems, we uniformly sample among three models. **Stochastic gradient descent** for both sparse and dense data where the loss is chosen as either log or Huber los, with either l_1, l_2 or a linear combination of them (elastic net). We used the inverse scaling learning rate model, which decays from the initial learning rate η_0 parametrized by *pow*. For sparse problems only, we use **Multinomial Naive Bayes**. When dealing with dense data, also use **Random forest classifiers** or **Extra tree classifiers**. For regression problems, we use either **Stochastic gradient descent regressors** for both sparse and dense data, and **Random forest regressors** or **Extra Tree regressors** for dense data. The pipeline space is explored using the Tree-structured Parzen Estimator (TPE) approach using the alotted time budget.

Table 1 shows the results of this approach among the round 1 of the challenge, reaching third place overall out of 68. While being one of the approaches with the most time budget spent, it consistently ranked among the top within all five datasets, reaching first place on Set 3.

3. Conclusions

We have designed a model configuration space with different pre-processing steps and learning models. First, we propose a fast preprocessing step that increases the expressive power via kernel approximation. Second, we configure our model space so that different properties of the problem can be captured by appropriate algorithms. For future work, we will consider other approximations of expressive power that can be estimated quickly and memoized, like randomized trees or representation learning approaches. We will also explore the effect of changing the parameter configuration while the model is still learning in order to reduce the warm-up time.

References

James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *Proceedings of The 30th ICML*, pages 115–123, 2013.

Isabelle Guyon, Kristin Bennett, Gavin Cawley, Hugo Jair Escalante, Sergio Escalera, Tin Kam Ho, Núria Macia, Bisakha Ray, Mehreen Saeed, Alexander Statnikov, et al. Design of the 2015 chalearn automl challenge.

Brent Komer, James Bergstra, and Chris Eliasmith. Hyperopt-sklearn: Automatic hyperparameter configuration for scikit-learn. In *ICML workshop on AutoML*, 2014.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *JMLR*, 12:2825–2830, 2011.

Tianbao Yang, Yu-Feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In *NIPS*, pages 476–484, 2012.