Automating Feature Engineering

Udayan Khurana[1], Fatemeh Nargesian[2], Horst Samulowitz[1], Elias Khalil[3], and Deepak Turaga[1]

[1]IBM Watson Research Center
[2]University of Toronto
[3]Georgia Institute of Technology

Abstract

Feature Engineering is the task of transforming the feature space in a given learning problem to improve the performance of a trained model. It is a crucial but time-intensive and skillful process, involving a data scientist or a domain expert. It is often the key determinant of the time and cost required to build an effective learner. In this paper, we discuss our system for performing feature engineering in an automated manner using a combination of exploratory and learning techniques. We also mention our larger charter of an automated data science pipeline.

1 Introduction

An increasing number of information systems now rely on machine learning based predictive models to capture and predict behavior, outcomes and likelihoods. We have already witnessed the impact of using a wide array of learners by systems such as IBM Watson in various domains such as healthcare, weather, and finance, amongst others. At the core of building, managing, and adapting such predictive models through their lifecycles is a large amount of manual processing, analysis, tuning, and experimenting with the data. This involves cleaning data, dealing with missing values, training and testing the models using machine learning algorithms, engineering the features. The complexity of resulting combinatorial choices makes it a computationally hard problem. It requires extensive intellectual capital and human effort, making the process lengthy, bound by limited scalability, costly, and often prohibitive. In this paper, we describe some of our recent and ongoing work in automating data science components in order to proliferate its adoption in information systems.

Specifically, we focus on *feature engineering* or *feature construction*, which is a time-consuming albeit crucial step in the data science pipeline. In that respect, a data scientist performs *transformations*, compositions or subset-selection on given features in an iterative manner while observing changes in model accuracy for the desired predictive analysis task. The efficacy of this human driven process is heavily dependent on the domain and statistical expertise of the individual, and is constrained by the time to delivery. On the other hand, simple automations for exhaustive transformation and validation tend to be infeasible due to the inherent combinatorial complexity. For the want of space, we refer to a few representative related works on FICUS [4], Data Science Machine [5] and Brainwash [6].

2 System Overview

Upon a deeper understanding of what enables a data scientist to successfully perform feature engineering, we realized that it is a factor of two components. First is the wisdom of *"what works"* through experience over months and years of working with predictive modeling tasks. Secondly, the inherent problem solving drive of a trained human to *try, observe,* and *adapt* is essential in navigating through a vast number of possible choices on an unknown data and unforeseen behavior. Our system consists of two main components as shown in Figure 1: the `Explorer` and the `Learner-Predictor`.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

The `Explorer` [1] navigates through various feature construction choices in a hierarchical and non-exhaustive manner, while progressively maximizing the accuracy of the model through a greedy adaptive exploration strategy. A *Transformation Tree* is used to systematically enumerate the space of different data *transforms* (such as *logarithm*, *frequency*, *z-score*, *temporal aggregation*, and so on) that can be applied in sequence to a dataset, while search or pruning strategies are used to effectively find the optimal node in the tree. It is easily extensible to add new transform functions to the system. It works in a domain independent manner, yet enabling a domain expert to influence the search through specification of constraints relevant to the dataset.

The `Learner-Predictor` [3] generalizes the impact of different transformations on a range of historical datasets, and learns to predict the most suitable transformation for each feature in a given dataset. The generalization across feature vectors of different sizes and representing different quantities, however, is non-trivial. Moreover, different datasets belong to different domains and represent different learning targets. We canonicalize feature representations using a novel way to represent any feature vector using a method called *imagification*, which consists of normalizing, binning and histogramming. Through meta-learning a predictive model on such feature representations, we are able to predict the most suitable transform for each feature independently. The results show a good accuracy of prediction and fast execution time. When put together, the `Learner-Predictor` quickly provides recommendations for applying certain transformations. This is a relatively inexpensive step which helps the `Explorer` narrow down or prioritize its search space. The Transformation Tree in the explorer produces several intermediate transformed datasets and for each of those, it reaches out to the predictor for updating transformation recommendations or priorties.

Automating the Data Science Pipeline. While feature engineering is a critical and perhaps the most time consuming step in the data science pipeline, other steps such as model selection, data cleaning, data completion, are also essential. In the larger scope of this project, we work on a holistic automation of the data science process. For instance, the efficacy of engineered features also depends on the type of model being employed. Our systems for feature engineering [1] and model selection [2] using incremental sampling and estimation, work in tandem to solve the joint problem of finding the best model and feature set. Finally, the transformation tree extends to test the impact of various data preparation choices similar to data transformations in a performance driven manner.

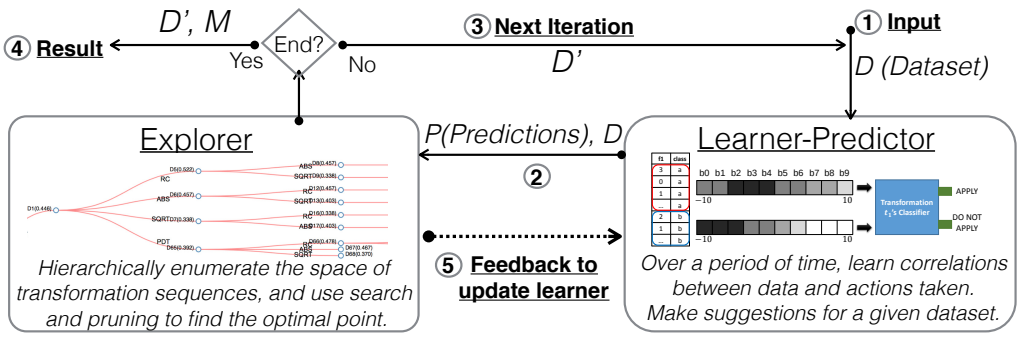

Figure 1: We perform automated feature engineering using an iterative two-phase process through (a) `Learner-Predictor` providing transformation hints based on past experience; and (b) `Explorer` navigating through space of weighted options to provide the optimal sequence of transformations.

References

[1] U. Khurana, et al. Cognito: Automated Feature Engineering in Supervised Learning. *ICDM*, 2016.

[2] A. Sabharwal, et al. Selecting Near-Optimal Learners via Incremental Data Allocation. *AAAI*, 2016.

[3] F. Nargesian, et al. Learning Feature Engineering. *Under review at SDM*, 2017.

[4] S. Markovitch, et al. Feature generation using general constructor functions. *Machine Learning*, 2002.

[5] J. Kanter, et al. Deep feature synthesis: Towards automating data science endeavors. In *IEEE DSAA*, 2015.

[6] M. Anderson, et al. Brainwash: A Data System for Feature Engineering. *CIDR*, 2013.