Teaching a computer to be a data scientist

Kalyan Veeramachaneni

Over the past 2 years, I have been chasing a foundational question: "How can I make a computer do what my peers and I do as data scientists?" How can I teach it to recognize data as having come from, say, a *bank* or an *online education platform*, without human instruction or supervision? How can it ask the questions we would ask of this data, and how can it take steps to answer those questions? This line of inquiry was inspired by a similar one from few decades ago, which eventually gave rise to what we now know as computer vision–a thriving field with huge implications for many of today's societal endeavors.

Quixotic as it sounds, *teaching computers to be data scientists* has a number of practical implications:

- **Increased Efficiency**: A computerized data scientist will help us handle many problems simultaneously. This will take some of the weight off of data scientists who, as data becomes more and more of a part of everyday life, have been asked to handle increasingly diverse problems at a rapidly expanding rate.
- **Broaden User Base**: Complete automation will lower the barrier to entry, allowing humans from an unprecedented number of fields and backgrounds to derive insights from data.
- **Interdisciplinary Connections**: A computerized data scientist will build a much-needed bridge between big data processing engines and machine learning algorithms, eliminating the huge amount of time experts spend transforming data.

Earlier projects that paved the way

I was originally motivated by a strong desire to make human-data interactions easier. To this end, I founded a series of projects across a wide variety of data types and domains, including the following:

- **The Gigabeats project**: Development of predictive models using *physiological* signals is time-consuming; even a modest study usually takes from 6 to 12 months. In response, I developed Gigabeats, which judiciously pre-computes features per beat for reuse across investigations, and offers parameterizations that allow for task-specific computing and storage tradeoff decisions [1]. It resolves questions such as: what data abstractions make for efficient data storage, retrieval and processing? Which features should be pre-processed and stored vs. computed "on the fly"? [2] What interfaces for machine learning and analytics are generally useful? Where and how can the process be more agile, modular, and efficient? With these questions answered, we were able to derive 700 predictive models against different outcome definitions in a single day (the previous rate was 1 model per 2 weeks). [1]
- **The MOOCdb project**: Online learning platforms now attract millions of students, and commensurate data complexity. I brought together multiple universities' platforms and developed a data description standard called MOOCdb [3]. This has allowed us and several other groups to develop applications, predictive modeling software, and additional reusable modules [4]. With this software, we can develop models on data from one course, and examine whether they work in other contexts [5, 6][2].
- **Estimating wind resources**: We created a high fidelity, non-gaussian, nonlinear modeling method (using *copulas*) to estimate wind speed distribution at a particular test site [7]. This required developing a system that synchronized publicly available data with measurements collected from sensors around the state [8]. Aggregating data from many sources and building copula-based models allowed us to derive accurate wind speed estimates with 3 months of data (industry standard is 8 months). [3]

After repeatedly hearing industry leaders express concerns about efficiency, I aggressively sought industrial projects as well, and led the development of several data-driven predictive analytics platforms for diverse industries. These include predicting driver behavior for Jaguar and Land Rover using 7000 event-driven signals, predicting delays in software delivery for Accenture using 510 different signals collected weekly or monthly, and predicting fraud for Banco Bilbao Vizcaya Argentaria (BBVA) using a billion transactions across millions of credit cards.

Each of these projects led to novel approaches that improved the quality and efficacy of data-based discovery in that particular field. While working on them, we noticed some common challenges:

[1] The Gigabeats project has culminated in a 3-year, 700K project with Philips (2015-2019).

[2] The MOOCdb project led to a 5-year NSF project (2015-2020) called CIF21 DIBBs: Building a Scalable Infrastructure for Data-Driven Discovery and Innovation in Education. MIT's share of this project is 750K.

[3] This project has spun out as a startup, Evervest, founded by my student Teasha Feldman-Fitzthum.

- It took a long time to get from data to models, and most of this delay was due not to computational bottlenecks, but to a lack of human capital. Without human ingenuity, raw data cannot be transformed into the data representations required by machine learning platforms.
- It was cognitively challenging to hop back and forth between problems, each of which required us to understand a particular set of data and its context, and to remember the nuances of all the fields that composed that data.
- Despite the diversity of the projects' goals, each required the same sequence of steps: organizing the data, linking and cleaning the data, extracting features by writing customized scripts (while considering the nuances of bursty, diverse timescales and types), and then selecting and building models.

Thinking more generally

Having noted all these common challenges, I shifted my focus and began to think more broadly. I wondered: can we develop general-purpose tools and utilities for use in data science? How can we make the process more systematic and structured while still considering the maximum amount of complexity? Over the past year, I laid the first steps toward a number of relevant projects, each of which required building a platform, developing an algorithm, or structuring an otherwise ad-hoc process. The next phase, spanning 3-5 years, for each will involve experimentation, demonstration of wide-scale applicability, and generation of user interfaces designed for broad adoption, ultimately making human interactions with data easy and enjoyable. I highlight these projects below.

- **Feature Factory**: An early and critical step in data science is transforming information from raw, unorganized data into features (*explanatory variables*) that describe the entities (*students*, *cars*, *customers* etc.)[9]. [4] Traditionally, feature engineering depends heavily on human intuition and domain expertise. Gathering as many diverse features as possible is often beneficial for the problem at hand–in one experiment, we showed that collecting ideas from a group of people outside of our research team increased the predictive accuracy of a model by up to 20% [10].

  To enable both idea and software generation, we developed *Feature Factory*–a web-based platform that enables users to define, extract, and test features on any given machine learning problem. A server hosting a sample dataset allows participants to write scripts within pre-defined software abstractions. The platform then invokes a learning algorithm and gives participants feedback. We tested the platform on 3 different data science problems [11]. The platform also gives us insight into how data scientists materialize features, a critical step towards automating the feature engineering step.[5]
- **Machine Learning Blocks (MLBlocks)**: A data modeling pipeline can be abstracted into a number of discrete steps. We turned each step into a reusable, tunable software module, or "*block*". These blocks include feature *extraction*, *transformation*, and *selection*, clustering, feature pre-processing and modeling. Data scientists can put multiple *blocks* together to construct an end-to-end modeling pipeline, and can also develop and reuse their own *blocks*. With modular definition of the *blocks* and ability to store intermediary state of data, this streamlines the process and allows data scientists to make a variety of predictive models without needing to revisit the raw data [12].
- **Deep Mining**: A later but equally vital step in data science is tuning–honing the parameters of different steps in the pipeline to achieve more accurate models. Traditionally, data scientists must hand-tune these parameters, making this another time-consuming and iterative part of the process. The Deep Mining algorithm simultaneously tunes the parameters of the entire pipeline using a Gaussian copula process[6]. Since the parameters for the entire pipeline are tuned, each iteration is computationally intensive. This requires novel sub-sampling of data, caching of computations for future re-use, and the ability to work with uncertain estimates of pipeline performance. We initially optimized three work flows - sentiment analysis, handwritten digits classification and relational data problems from KAGGLE. With a system in place, we are now looking at ways to optimize pipelines without computing on all the data.

[4] "At the end of the day, some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used." Pedro Domingos in *A Few Useful Things to Know about Machine Learning*

[5] While KAGGLE also seeks to crowdsource feature engineering, it doesn't allow data scientists to see each other's work, and requires them to write learning algorithms as well, which takes time and focus away from feature engineering proper.

[6] For example, for a Handwritten digit recognition, it tunes kernel size and standard deviation for Gaussian Blur based feature extraction, number of components to be used after PCA and degree and gamma coefficient for the SVM classifier.

- **Deep Feature Synthesis**: After observing how data scientists materialize features from relational data, we wanted to automate feature generation for this data type[7]. To this end, we developed the Deep Feature Synthesis algorithm [13]. The algorithm follows data relationships to a base field, and then sequentially applies mathematical functions along that path to create the final feature. With this we are able to produce *thousands* of features for every entity in the data automatically.
- **Delphi**: We wanted to develop something that would allow us to take what we learned from one data science project (*what worked well* vs. *what didn't*) and apply it to others. We decided to build a *recommendation system* focused on classifier design [14]. For Delphi, we enumerated a large classifier space, and learned roughly 5000 classifiers for 22 datasets (including cross-validation, this involved learning roughly 1 million models). Then, for each new dataset, we learn k classifiers first, rank them according to their cross-validated accuracy, and then compare to see which of these datasets the new one most resembles in the ranked space. We then use the best classifier for the that data set on this new one. This simple system has allowed us to cut through the search space by an order of magnitude.
- **The Data Science Machine**: By combining Deep Feature Synthesis and Deep Mining, we developed the Data Science Machine– an end-to-end automated system that can turn raw data into predictive models with minimal human input. This machine beat a majority of human competitors in 3 competitions held at premier machine learning conferences. These conferences featured 906 other teams, and our approach beat 615 of them. In 2 of the 3 competitions, we beat a majority of competitors, and in the third, we achieved 94% of the best competitor's score. In the best case, with an ongoing competition (at that time), we beat 85.6% of teams and achieved 95.7% of the top submission's score.

As I continue to innovate on these projects, I am also working on methods that automatically generate questions and phrase them as machine learning problems [15], and that automate generative modeling [16] (the DSM only tackles predictive modeling). Automating any of these otherwise human-driven endeavors requires building a system that enables wide-scale human participation across many datasets (much like the Feature Factory), uses these to learn how data scientists do a specific task, creates abstractions and algorithms (like Deep Feature Synthesis) and ultimately automates and demonstrates its efficacy compared to humans (like the DSM). A completely automated process will encourage humans to opt in again, but in different roles–for example, Deep Feature Synthesis allows humans to become feature *selectors* instead of feature *creators*. The ultimate achievement would be to supply different interfaces to different people (economists, analysts, educators) in order to bring them closer to their goals–and to make it easier, more effective, and more enjoyable for people to work closely with data.

Reflection

Before these latest endeavors, I spent my days creating newer versions of latent variable models, or distributing existing learning algorithms over cloud-based infrastructure. But as I worked on those problems, I realized that I was ignoring issues that, if properly addressed, would lead to a real revolution in data science. If we widened the well-known bottlenecks that slow down the journey from raw data to conclusions, we could use this efficiency to enable more people to interact with data in productive ways. While there was widespread acknowledgement in the machine learning community [9, 17], not much was being done to address the problems. Perhaps because it requires rethinking feature engineering, scaling human understanding of data, and figuring out how to deal with the cognitive overload that results from encountering data over multiple domains.

Taking on such broad questions was a bold move in my career, and was difficult to justify [8]. Hence, I soon realized that a big part of my new task was communicate in an imaginative way about what I was doing, to bring in some rigor, define metrics of success, and develop systems that could capture the interest of my peers and the larger scientific community. After two years of work, we developed something which not only addressed the problems I set out to solve, but did so in a rigorous, measurable, and imaginative way–the Data Science Machine (DSM) [18].

In my time as a researcher, I have been lucky enough to enjoy a number of institutional resources, including a highly talented pool of students, access to industry, and exemplary peers who challenge me to communicate better and aim higher. All of these have been critical to my success. However, while I did what I did by

[7] Note that feature generation for images, text and signals is somewhat automated already *via* signal and image transforms, LDA and LSA for text, and ultimately Deep Learning.

[8] A quintessential thought I had several times - *whether I should work and learn more about Deep Learning?*

simply by learning from my peers, to fully realize the bold vision of *teaching a computer to be a data scientist* and ultimately *democratizing data science*, I look forward to working more closely with experts in *big data*, *machine learning*, *programming languages* and *human-computer interaction*.

References

[1] F. Dernoncourt, K. Veeramachaneni, and U.-M. O'Reilly, "beatdb: A large scale waveform feature repository," in *NIPS Workshop on Machine Learning for Clinical Data Analysis and Healthcare*, 2013.

[2] F. Dernoncourt, K. Veeramachaneni, and U.-M. OReilly, "Gaussian process-based feature selection for wavelet parameters: Predicting acute hypotensive episodes from physiological signals," in *IEEE 28th International Symposium on Computer-Based Medical Systems*, 2015.

[3] K. Veeramachaneni, F. Dernoncourt, C. Taylor, Z. Pardos, and U.-M. O'Reilly, "Moocdb: Developing data standards for mooc data science," in *AIED 2013 Workshops Proceedings Volume*. Citeseer, 2013, p. 17.

[4] S. Boyer, B. U. Gelman, B. Schreck, and K. Veeramachaneni, "Data science foundry for moocs," in *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*. IEEE, 2015, pp. 1–10.

[5] C. Taylor, K. Veeramachaneni, and U.-M. O'Reilly, "Likely to stop? predicting stopout in massive open online courses," *arXiv preprint arXiv:1408.3382*, 2014.

[6] S. Boyer and K. Veeramachaneni, "Transfer learning for predictive models in massive open online courses," in *Artificial Intelligence in Education*. Springer, 2015, pp. 54–63.

[7] K. Veeramachaneni, A. Cuesta-Infante, and U.-M. O'Reilly, "Copula graphical models for wind resource estimation," in *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press, 2015, pp. 2646–2654.

[8] K. Veeramachaneni, T. Feldman-Fitzthum, A. C. Infante, and U.-M. O'Reilly, "Computer-implemented data analysis methods and systems for wind energy assessments," Patent US 0 160 373, 06 11, 2015. [Online]. Available: http://www.google.com/patents/US20150160373?cl=en

[9] P. Domingos, "A few useful things to know about machine learning," *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, 2012.

[10] K. Veeramachaneni, K. Adl, and U.-M. O'Reilly, "Feature factory: Crowd sourced feature discovery," in *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*. ACM, 2015, pp. 373–376.

[11] A. C. Wang, "Feature factory: A collaborative, crowd-sourced machine learning system," Master's thesis, Massachusetts Institute of Technology, 2015.

[12] B. Santiago Omar Collazo, "Machine learning blocks," Master's thesis, Massachusetts Institute of Technology, 2015.

[13] J. M. Kanter and K. Veeramachaneni, "Deep feature synthesis: Towards automating data science endeavors," in *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*. IEEE, 2015, pp. 1–10.

[14] W. Drevo, K. Veeramachaneni, and U.-M. O'Reilly, "A distributed, multi-model, self-learning platform for machine learning," Patent.

[15] B. Schreck, "What would a human ask?: Automatic generation of data science inquires." Master's thesis, Massachusetts Institute of Technology, 2016 - in progress.

[16] N. Patki, "The synthetic data vault: Automating generative modeling for databases." Master's thesis, Massachusetts Institute of Technology, 2016 - in progress.

[17] K. Wagstaff, "Machine learning that matters," *arXiv preprint arXiv:1206.4656*, 2012.

[18] R. Feltman, "New mit algorithm rubs shoulders with human intuition in big data analysis," *The Washington Post*, October 2015.