WikiCities: a Feature Engineering Educational Resource

Pablo Duboue
Textualization Software Ltd.
Vancouver, BC, CANADA

Abstract

Teaching ML through competitions is an accepted and even recommended learning path in the field. Learning feature engineering is difficult as it involves domain knowledge and iterative improvements. This abstract describes WikiCities, a dataset put together to showcase feature engineering techniques. This dataset is a prime candidate for having feature engineering-centric ML competitions.

1 Introduction

Feature engineering and ensemble learning are consistently among the top techniques for successful submissions at ML competitions [1]. Ensemble learning is straightforward to teach and exercise [2], but how to teach feature engineering? For the book "The Art of Feature Engineering,"[3] I borrowed a page from business school playbook and focused on case studies as a technique to teach domain-dependent skills. The WikiCities dataset is central to the case studies. It consists of a dataset of 80,000 small towns and cities together with structured (relational) data, textual descriptions, satellite imaginery and historical variants of the relational data.

The task is to predict the total population of a town or city. This task is complex enough to be interesting but it is also close to everyday intuitions. In the words of a colleague professor that reviewed the book:

> [the problem is] complex enough to be non-trivial, everyone knows and uses the data, and there's enough noise and interesting features there to make it challenging.

2 Dataset Description

The list of towns and cities is obtained from GeoNames [4], to increase the reliability of the information. Only towns and cities with a population greater than 1,000 inhabitants are used. For **structured data**, it uses semantic information extracted from Wikipedia infoboxes, as made available in semantic web [5] format by the DBpedia project [6]. The use of graph data in ML has attracted plenty of attention in recent years [7] and there are AutoML approaches that excel at it [8]. Moreover, it also involves handling and representing variable-length raw data as fixed-length feature vectors. The full dataset contains 2 million triples (edges in the graph). In each triple, one of its elements is a town or city and the relation is one from 43 relations that appear in at least 5% of the places. The full DBpedia dump is also available if there is interest in exposing more information from the graph, as enhancing the data by retracing its source is an accepted feature engineering technique [9]. The relations used have the target feature ("population") removed plus other potential target leaks (e.g., "urban population") but proxies involving computable features are left for experimentation (e.g., "population density" and "area"). A baseline system achieves a RMSE of 0.3298 on a logscale.

The natural textual data for each place is the **full text** of their associated Wikipedia page. Both the detagged and fully tagged versions are kept, for further experimentation. The detagged version is

Preprint. Under review.

Table 1: Data Sizes (Compressed)

Section	Base Size	Extended Size
Structured	18 Mib	341 Mib
Textual	65 Mib	151 Mib
Timestamped	9 Mib	87 Mib
Images	1536 Mib	6554 Mib
Total	1628 Mib	7133 Mib

obtained using wikiextractor [10] and it totals 44 million words and over 270 million characters (an average of 558 words per document, or 3445 characters). The total number of characters with markup climbs up to over 730 million characters. A number of documents contains the population mentioned in the text (53%) which means that simple information extraction techniques will not solve the problem fully. Moreover, to obtain reasonable features, care has to be taken to handle numbers and number variants (with commas, etc) carefully, which stresses **tokenization** and **discretization** issues. With a vocabulary size in excess of 408 thousand word types, it also invites feature selection use [11]. Using this data produces a RMSE of 0.3267 in a baseline system.

As Wikipedia (and its InfoBoxes) change over time, six years of DBpedia dumps make for a dataset that contains insights on the velocity of the data. Because of **ontological changes** from different versions of the DBpedia extraction scripts, it also allows for exercising the type of temporal smoothing and imputation common on timestamped data [12]. No baseline improvement has been recorded on this variation of the data.

Finally, WikiCities includes sensor imaging as provided by NASA on the 31.25 meters per pixel range. These images were downloaded from the GIBS tile server. Using a tile server has the advantage that the tiles are directly in the **latitude** and **longitude** format, but it also means the tiles are squeezed in the vertical direction depending on how close to the poles is the city. This allows to exercise some popular affine transformation techniques. The files contain sensor data captured in 14 bands of the electromagnetic spectrum using two cameras to record elevation features. These are images but not in the visible spectrum. That also means that pre-trained NN models would be of little expected value.[13] The raw data in this case study is a set of tiles, downloaded from NASA, based on the GPS coordinates distributed by GeoNames. The tiles are then transformed into 64×64 pixel range surrounding the latitude and longitude centre of each city as provided by GeoNames. A baseline system on this dataset produces a RMSE of 0.3188.

The sizes of each sub-part are described in Table 1. As usual with datasets, image data dominates the total size. While large overall, it is still suitable for Internet distribution (needing to rely, potentially, on Academic Torrents [14] or other such services). The computation power needed to process these datasets fits in a laptop with 8Gb of RAM without the need of a GPU.

Care has been taken when putting together the dataset to find data under open licenses that allow for redistribution. 10,000 lines of Jupyter notebooks exemplifying different feature engineering techniques are also being released.[1] The feature engineering methodology proposed in the book and showcased in the case studies relies heavily in an human-in-the-loop approach based on both exploratory data analysis and error analysis. Feature engineering is known to be laborious and domain knowledge-intensive. The available code highlights its iterative nature by engaging in several iterations of featurization and piece-wise improvement of the feature set.

3 Conclusions

It would be interesting to find CiML partners to have competitions over this data, focusing not necessarily only on improving the final regressor results, but also showcase smart feature engineering approaches across data sources. We echo the need for non-featurized datasets [15] but not only for AutoML tasks, but also for humans to improve their feature engineering skills. The baseline code provides the full transformation of the raw data into "tidy" featurized data [16], but this is a guideline for the students to benchmark themselves against.

[1] Currently available at `http://artoffeatureengineering.com/review_notebooks`.

Acknowledgements

We acknowledge the use of imagery provided by services from the Global Imagery Browse Services (GIBS), operated by NASA's Earth Science Data and Information System (ESDIS) Project.

References

[1] Victor Lavrenko. Machine Learning = Feature Engineering. https://www.youtube.com/watch?v=CAnEJ42eEYA, 2016. Accessed: 2018-06-06.

[2] James Robert Lloyd. Sensible allocation of computation for ensemble construction. Presentation at CiML 2015. https://docs.google.com/a/chalearn.org/viewer?a=v&pid=sites&srcid=Y2hhbGVhcm4ub3JnfHdvcmtzaG9wfGd4OjQwZjdmYTRhNjRhMzBiYzE, 2015.

[3] Pablo Duboue. *The Art of Feature Engineering: Essentials for Machine Learning*. Cambridge University Press, 2020. ISBN 1108709389.

[4] Marc Wick. Geonames ontology. http://www.geonames.org/about.html, 2015. Accessed: 2015-04-22.

[5] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284 (5):34–43, May 2001.

[6] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2):167–195, 2015.

[7] Palash Goyal and Emilio Ferrara. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151:78 – 94, 2018. ISSN 0950-7051.

[8] James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2015, Paris, France, October 19-21, 2015*, pages 1–10. IEEE, 2015.

[9] Jason Brownlee. Discover feature engineering, how to engineer features and how to get good at it. https://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get-good-at-it/, oct 2014. Accessed: 2018-05-02.

[10] Giuseppe Attardi. wikiextractor. https://github.com/attardi/wikiextractor, 2018. Accessed: 2018-12-12.

[11] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[12] Henrik Brink, Joseph W Richards, and Mark Fetherolf. *Real-world machine learning*. Manning, 2017. ISBN 1-61729-192-7.

[13] Ragav Venkatesan, Vijetha Gatupalli, and Baoxin Li. On the generality of neural image features. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 41–45. IEEE, 2016.

[14] Henry Z. Lo and Joseph Paul Cohen. Academic torrents: Scalable data distribution. Presentation at CiML 2015. https://docs.google.com/a/chalearn.org/viewer?a=v&pid=sites&srcid=Y2hhbGVhcm4ub3JnfHdvcmtzaG9wfGd4OjFiNGQyMmU5YjE1ODUyYzA, 2015.

[15] Richard Lippmann, Swaroop Vattam, Pooya Khorrami, and Cagri Dagli. A new data corpus to promote more complete autonomous machine learning pipelines. Presentation at CiML 2018. https://docs.google.com/a/chalearn.org/viewer?a=v&pid=sites&srcid=Y2hhbGVhcm4ub3JnfHdvcmtzaG9wfGd4OjNiODFkODQ2ODZjMGQyZjY, 2018.

[16] Hadley Wickham. Tidy data. *Journal of Statistical Software*, 59(1):1–23, 2014.