Advanced Machine Learning

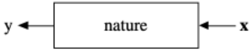
András Kornai

AML 2024/11/13

Modeling: the bird's eye view

1. INTRODUCTION

Statistics starts with data. Think of the data as being generated by a black box in which a vector of input variables $\mathbf{x}$ (independent variables) go in one side, and on the other side the response variables $\mathbf{y}$ come out. Inside the black box, nature functions to associate the predictor variables with the response variables, so the picture is like this:



There are two goals in analyzing the data:

Prediction. To be able to predict what the responses are going to be to future input variables;
Information. To extract some information about how nature is associating the response variables to the input variables.

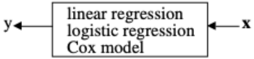
There are two different approaches toward these goals:

The Data Modeling Culture

The analysis in this culture starts with assuming a stochastic data model for the inside of the black box. For example, a common data model is that data are generated by independent draws from

response variables = f(predictor variables, random noise, parameters)

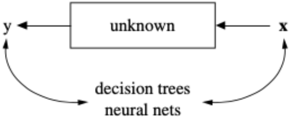
The values of the parameters are estimated from the data and the model then used for information and/or prediction. Thus the black box is filled in like this:



Model validation. Yes–no using goodness-of-fit tests and residual examination.
Estimated culture population. 98% of all statisticians.

The Algorithmic Modeling Culture

The analysis in this culture considers the inside of the box complex and unknown. Their approach is to find a function $f(\mathbf{x})$—an algorithm that operates on $\mathbf{x}$ to predict the responses $\mathbf{y}$. Their black box looks like this:



Model validation. Measured by predictive accuracy.
Estimated culture population. 2% of statisticians, many in other fields.

The "data modeling" and "algorithmic modeling" schools

- Data modelers start with a class of mathematical models: these are highly parametrized, and have only few parameters.
- How few? "With four parameters I can fit an elephant, and with five I can make him wiggle his trunk" (Neumann János)
- They also tend to believe that these models actually reveal something about the internals of the black box
- Main problems: low-hanging fruit all gone, statistical tests become meaningless for millions of datapoints
- Biggest problem (Breiman): fit is no good!

Algorithmic modeling

- The key is prediction accuracy on unseen (future) data
- We don't care if we don't understand the model, black box is good enough
- Many parameters: millions are common, GPT 4o has $2 \cdot 10^{11}$ numerical parameters
- Very loosely structured models, e.g. neural nets
- The approach benefits from theorems that show these are universal approximators
- Problems: even low-hanging fruit require very significant CPU reseources
- You may not care if you can't understand the model, but your sponsors will

Decision trees: where the tire meets the road

- Random forests (typically obtained by bagging/boosting) are good, but not interpretable
- Single trees (CART, C5.0) are more interpretable

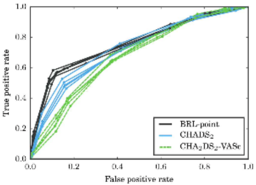


Fig. 4. *ROC curves for stroke prediction on the MDCD database for each of 5 folds of cross-validation, for the BRL point estimate, CHADS$_2$ and CHA$_2$DS$_2$-VASc.*

- But even trees may be too general
- Decision lists may be a good compromise

Enrichment

- The best form of learning is memorization 'memory is all you need'
- Everything else is about generalization ability. This is really necessary when there is a long tail, so training samples don't cover the problem space well, e.g. in NLP, where practically every sentence is new.
- Recent autonomous driving examples are all like this, exotic signage and traffic blocking at road repair, unusual vehicles ... horse-driven carriage
- When there is no data *enrichment* makes sense
- One clever trick is *bootstrap aggregating* 'bagging' you have only n datapoints, but you resample from these uniformly with replacement, train new models that way, and vote in the end.

Decision lists

if hemiplegia **and** age > 60 **then** *stroke risk* 58.9% (53.8%–63.8%)
else if cerebrovascular disorder **then** *stroke risk* 47.8% (44.8%–50.7%)
else if transient ischaemic attack **then** *stroke risk* 23.8% (19.5%–28.4%)
else if occlusion and stenosis of carotid artery without infarction **then** *stroke risk* 15.8% (12.2%–19.6%)
else if altered state of consciousness **and** age > 60 **then** *stroke risk* 16.0% (12.2%–20.2%)
else if age ≤ 70 **then** *stroke risk* 4.6% (3.9%–5.4%)
else *stroke risk* 8.7% (7.9%–9.6%)

Model comparison

Table 2

Mean, and in parentheses standard deviation, of AUC and training time across 5 folds of cross-validation for stroke prediction. Note that the $CHADS_2$ and CHA_2DS_2-VASc models are fixed, so no training time is reported

	AUC	**Training time (mins)**
BRL-point	0.756 (0.007)	21.48 (6.78)
$CHADS_2$	0.721 (0.014)	no training
CHA_2DS_2-VASc	0.677 (0.007)	no training
CART	0.704 (0.010)	12.62 (0.09)
C5.0	0.704 (0.011)	2.56 (0.27)
ℓ_1 logistic regression	0.767 (0.010)	0.05 (0.00)
SVM	0.753 (0.014)	302.89 (8.28)
Random forests	0.774 (0.013)	698.56 (59.66)
BRL-post	0.775 (0.015)	21.48 (6.78)

Decision trees: where the tire meets the road

- Random forests (typically obtained by bagging/boosting) are good, but not interpretable
- Single trees (CART, C5.0) are more interpretable

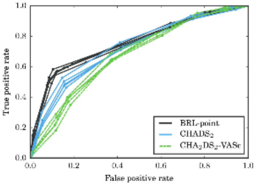


Fig. 4. *ROC curves for stroke prediction on the MDCD database for each of 5 folds of cross-validation, for the BRL point estimate, $CHADS_2$ and CHA_2DS_2-VASc.*

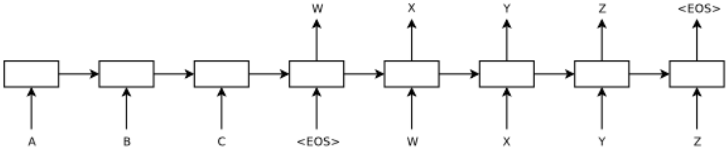
- But even trees may be too general
- Decision lists may be a good compromise

LSTM (long short-term memory)

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) && (1)\\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) && (2)\\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) && (3)\\
\tilde{c}_t &= \sigma_c(W_c x_t + U_c h_{t-1} + b_c) && (4)\\
c_t &= f_t \circ c_{t-1} + i_t \circ \tilde{c}_t && (5)\\
h_t &= o_t \circ \sigma_h(c_t) && (6)
\end{aligned}
$$

- Historically LSTM (Hochreiter and Schmidhuber, 1997) preceded GRU (Cho et al., 2014) who wanted to simplify LSTMs
- LSTMs have more power (do better on long dependencies)
- Computational power of architectures investigated by (Weiss, Goldberg, and Yahav, 2018)
- Key idea: keep a very contentful state vector
- Best line of attack: information bottleneck method (Tishby, Pereira, and Bialek, 2000)

Seq2seq

- Using LSTMs as elementary building blocks (Sutskever, Vinyals, and Le, 2014)
- 

- Stacked 5 deep, state vectors 8000 dim
- Does MT (English-French) quite well
- Relies on reversing input
- Encoder-decoder architecture (can be retrojected on LSTM)

Attention

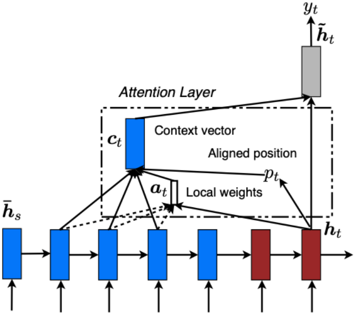
- Bahdanau, Cho, and Bengio, 2015; Luong, Pham, and Manning, 2015
- 

  Figure 3: **Local attention model** – the model first predicts a single aligned position p_t for the current target word. A window centered around the source position p_t is then used to compute a context vector c_t, a weighted average of the source hidden states in the window. The weights a_t are inferred from the current target state h_t and those source states $\bar{h}_s$ in the window.
- Self-attention (Lin et al., 2017) current form Vaswani et al
- Also developed for MT (which remains the canonical case)
- Introduces 'multi-head' model: several attention layers running in parallel
- Positional encoding: mixing sinusoids of different frequencies

Seq2seq + Attention = Transformers

- The first dynamic word vector system was CoVe (McCann et al., 2017)
- This was an encoder-decoder model trained on various MT datasets (but no effort to mix them in a single model)
- Trained on MT data (7m sentence pairs)
- Encoder output concatenated to a static (GloVe) embedding
- CoVe had sophisticated bidirectional attention, but not as good as Transformers

ELMO

- Encoder-decoder trained on LM task (monolingual – much more data) Peters et al., 2018
- Multi-head (transformer-style) attention
- Concatenates all (not just the top) LSTM states
- For specific tasks, it may make sense to re-train the LM itself
- ELMO training used 1G words of English text, GPT-2 on about 8G words, GPT-3 on over 100G words (45 TB compressed from CommonCrawl, plus curated datasets)
- GPT-3 175G parameters trained in $3.14 \cdot 10^{23}$ flops (a third yottaflop)
- Energy usage alone 500MWh

BERT

- Introduced in Devlin et al., 2019
- Similar to ELMO, but trained on much less data than GPT: 800m words from the Google Books Corpus and 2.5G words from WP
- Fully bidirectional, with 15% of tokens masked out
- m-BERT (multilingual, 104 languages), RoBertA (Liu et al., 2019), etc etc
- National BERT's: CememBERT (Martin et al., 2019), HuBERT (Nemeskey, 2021) 2020), ...
- Generalizations, BERTology

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2015). "Neural machine translation by jointly learning to align and translate". In: *International Conference on Learning Representations (ICLR 2015)*.

Cho, Kyunghyun et al. (2014). *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. arXiv: 1409.1259. URL: https://arxiv.org/abs/1409.1259.

Devlin, Jacob et al. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proc. of NAACL*.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

Lin, Zhouhan et al. (2017). *A Structured Self-attentive Sentence Embedding*. arXiv: 1703.03130. URL: https://arxiv.org/abs/1703.03130.

Liu, Yinhan et al. (2019). *RoBERTa: A robustly optimized bert pretraining approach*. arXiv: 1907.11692 [cs.CL].

Luong, Thang, Hieu Pham, and Christopher D. Manning (2015). "Effective Approaches to Attention-based Neural Machine Translation". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 1412–1421. DOI: 10.18653/v1/D15-1166. URL: http://www.aclweb.org/anthology/D15-1166.

Martin, Louis et al. (2019). "Camembert: a tasty french language model". In: *arXiv preprint arXiv:1911.03894*.

McCann, Bryan et al. (2017). "Learned in translation: Contextualized word vectors". In: *Advances in Neural Information Processing Systems*, pp. 6294–6305.

Nemeskey, Dávid Márk (2021). "Introducing huBERT". In: *XVII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2021)*. Szeged, pp. 3–14.

Peters, Matthew et al. (2018). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, Volume 1 (Long Papers). New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202. URL: http://aclweb.org/anthology/N18-1202.

- Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). "Sequence to Sequence Learning with Neural Networks". In: *Proc. NIPS*. Montreal, CA, pp. 3104–3112. URL: http://arxiv.org/abs/1409.3215.
- Tishby, Naftali, Fernando C. Pereira, and William Bialek (2000). *The information bottleneck method*. arXiv: physics/0004057 [physics.data-an]. URL: https://arxiv.org/abs/physics/0004057.
- Vaswani, Ashish et al. (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., pp. 5998–6008. arXiv: 1706.03762 [cs.CL]. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Weiss, Gail, Yoav Goldberg, and Eran Yahav (2018). "On the Practical Computational Power of Finite Precision RNNs for Language Recognition". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Ed. by Iryna Gurevych and Yusuke Miyao. Melbourne, Australia: Association for Computational Linguistics, pp. 740–745. DOI: 10.18653/v1/P18-2117. URL: https://aclanthology.org/P18-2117.