Model Interpretability in Machine Learning

Antoine Ledoux[1], Erik Forseth[2], Ed Tricker[3]

Abstract
Interpretability is an increasingly vital issue in machine learning. Computerized statistical modeling has become the de facto paradigm for quantitative decision-making in any number of fields, including healthcare, advertising, investing, and more. And yet, the relative opacity of many of these techniques can pose a real issue in sensitive applications. Furthermore, the inability to interpret a model's behavior removes an essential part of the feedback loop for the practitioner, who needs to have a good understanding of the model to know when it's bound to fail, or where it can be improved. In this note, we first review the canonical statistical machine learning problem, before describing the issue of model interpretability and some of the recent developments. We list some examples of both interpretable and non-interpretable models and explain some of the differences.

Keywords
Machine Learning, Statistical Modeling, Interpretability, Complexity

[1] *Quantitative Research Analyst*
[2] *Senior Quantitative Research Analyst*
[3] *Chief Investment Officer of Quantitative Strategies*

1. Introduction

As machine learning (ML) methods have become more powerful and ever more ubiquitous, *interpretability* has come to the fore as a vital issue for practitioners and decision-makers whose work relies on these models. In many applications, the ability to understand the outputs generated by a model is as important as the outputs themselves, if not more so. Using models in a pure 'black box' fashion invites a host of potential issues. A model might encode some hidden implicit bias, for example, which isn't evident to the user. Such biases could prove disastrous in sensitive applications like medical diagnosis or assessing creditworthiness. Or, there might exist regions of the model's parameter space for which the model is especially sensitive and prone to instability. Perhaps most importantly: without a deep understanding of the model's inner workings, it will be difficult or even impossible to understand when and why the model might fail to work as intended.

All of these issues and more arise in the application of ML to trading and investing. When deploying a model, a manager needs to understand *why* the model does what it does; he or she needs to have some idea about the effects the model purports to capture and the risk factors on which the model loads. For a fund with external clients, the manager needs to be able to explain his/her positioning to investors. And if the manager doesn't understand the decisions made by the model, it may not be apparent when something 'breaks.'

Of course, potential pitfalls aren't the only reason to care about interpretability. Good researchers are inherently curious and wish to learn from their work. There might be situations when a model works unexpectedly well. If the model's designer can determine *why* that is, he or she might be able to garner some new insight or uncover something that wasn't obvious about their problem.

A model's interpretability is closely (and in some sense, inversely) related to the model's complexity. In this note, we review some basic results about capacity and regularization in ML, drawing connections to the notions of robustness and generalization. We point to recent work on interpretability and explainability in modern ML. Finally, we turn the discussion back to investing and give some more thought to how these issues arise in quantitative finance.

2. The Statistical Learning Problem

Fitting a statistical machine learning model boils down to finding some function f that gives a good fit to some sample data, and hopefully generalizes well to any new data it might see. Given some datapoints $\{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$, we wish to build a function f which approximately maps each x_i to the corresponding y_i, or $f(x_i) \approx y_i$. The most popular framework to guide our search revolves around an idea called *empirical risk minimization*.

2.1 Empirical Risk Minimization

The empirical risk[1] is generically some average *loss* over the data:

$$\mathcal{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell\left(y_i, f(x_i)\right),$$

and is supposed to serve as a proxy for the expected true risk of the model.

[1] Note that in this context, "risk" does not refer to any kind of financial or market risk. Here it simply refers to the quality (or lack thereof) of a model's fit to some data.

Here, the loss function ℓ represents some kind of error in the approximation that we therefore wish to minimize. To begin with, we need to specify a class of functions, or a *hypothesis space*, from which our solution will be drawn. We might consider the space of linear functions, for example, having the form

$$f(x_i) = x_i' w + b.$$

This is a relatively simple model class (but a highly interpretable one, as we'll discuss later on). In this case, a common measure for the empirical risk is the mean-squared error:

$$\mathcal{R}(f) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 = \frac{1}{n}\sum_{i=1}^{n}\left[y_i - (x_i' w + b)\right]^2.$$

Often, we wish to restrict the hypothesis space even further by *regularizing* the empirical risk, meaning that we add an extra term to be minimized:

$$\tilde{\mathcal{R}}(f) = \frac{1}{n}\sum_{i=1}^{n}\ell\left(y_i, f(x_i)\right) + \lambda\Omega(f).$$

Here Ω is a function which penalizes the complexity of f, and λ is a constant which controls the strength of that penalty. In the case of linear regression, a common choice for Ω is the squared norm of the solution vector w:

$$\tilde{\mathcal{R}}(f) = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i' w)^2 + \frac{1}{2}\lambda\|w\|^2. \tag{1}$$

This model[2] is known as *ridge regression*, and this penalty has the effect of discouraging large values of w[3].

Another way to think about this penalty is to consider its effect on the 'prior' probability we're implicitly giving to certain kinds of solutions. Eqn. (1) now says that we're interested in solutions w which minimize the usual mean-squared-error *while not being too large in magnitude*. We're essentially putting further restrictions on the space of functions we're interested in. These sorts of ideas – which we'll explore further in the next section – become even more important when we consider more powerful and complex models, such as deep neural networks.

2.2 Restricting the Hypothesis Space

Why would we ever wish to restrict our hypothesis space when building a model? Surely we want the best fit possible? As it turns out, that's not necessarily true. Depending on the problem at hand, there may be a good reason to expect the solution to have a certain form or at least certain characteristics. Furthermore, ML practitioners often need to keep in mind a crucial trade-off between capacity – the ability to model complicated phenomena – and regularization. Some kinds of

[2]Note that for simplicity we've omitted the intercept term b; pretend that we appended a column of 1's to our input data and folded the intercept into the vector w.

[3]It also has an interesting effect on the role played by the covariance of the x's when finding the solution.

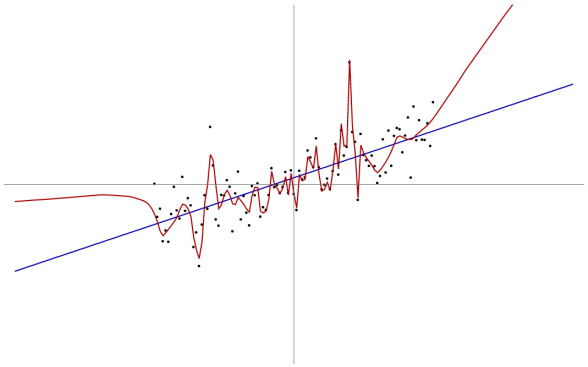

Figure 1. An example of the failure of a very high-capacity model versus a very simple one. The black data points were generated according to a linear model corrupted by noise and outliers. The blue line shows the result of a linear least squares fit to the data, whereas the red curve was generated by an un-regularized ten-layer feedforward neural network. The mean-squared-error of the neural network is (much) lower than the linear regression on the training data, but it's clearly not an appropriate model. If we were to naively trust this model without knowing better, we might infer some very complicated (and imaginary) dynamics from this dataset.

ML models (neural networks, for example) are known to be 'universal approximators,' capable of representing nearly any continuous function with arbitrary accuracy. But it's often the case that complicated functions aren't *robust*; they might be sensitive to outliers, or they might overfit the sample data, but become unstable or unreliable when presented with unseen inputs. Fig. 1 shows a basic illustration of this phenomena.

Nevertheless, some phenomena cannot be described by simple models. Many kinds of data have inherent nonlinear structure and require something more complicated to be modeled accurately. But as we use more powerful tools, we need to be wary. Because of their flexibility, we can often 'tune' high-capacity models until they give us the fit we want.

As we'll consider further in the next section, there is yet another important reason to consider restricting the hypothesis space for model building: we might wish to build models that we can *understand*. The capacity of a model, its ability to generalize well, and its interpretability are all closely related, but there are important distinctions. High-capacity models do not always overfit, simple models do not always generalize better, and interpretability is not antonymous with complexity. Nevertheless, without access to the data-generating process (or unlimited out-of-sample data), we need some way to feel confident that our models will behave robustly in the future. In this case, we often lean on our ability to grasp the model's output.

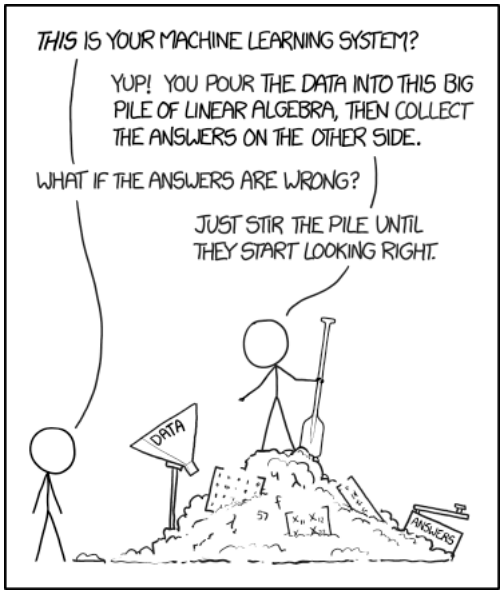

Figure 2. 'Machine Learning' by Randall Munroe, under a Creative Commons Attribution-NonCommercial 2.5 License. https://xkcd.com/1838/

3. Model Interpretability

Interpretability is not a formally defined concept but generally refers to the ability of a human to understand the cause of some output or decision. It's useful to make a distinction between two kinds of model interpretability [Molnar (2019)]. *Intrinsically* interpretable models are simple by construction; either they have some straightforward functional form, or their complexity has been restricted by regularization. The behavior of these models can be understood because their structure is not overly complicated. When models are not intrinsically interpretable, we can still talk about *post hoc* interpretability, referring to a set of methods and tools for understanding complex models *after* they've been trained. There is a growing body of work, for example, on distilling and understanding the output of very deep neural networks [Simonyan et al. (2014)].

Models are usually not categorized as either interpretable or not interpretable. Rather, they lie on a spectrum of interpretability depending upon their complexity and how well one can track down the cause of the model's output (see Figure 3).

3.1 Intrinsically Interpretable Models

Intrinsically interpretable models include linear models, rule-based models, and decision trees, where in each case the sequence of transformations used to transform the input to the output remain understandable throughout the entire process. These so-called 'white-box' models allow the user to stay in control each step of the way, so that each decision is traceable in a transparent way to the corresponding inputs.

In the case of linear models, the influence of each feature is simply encoded by the weight given to that feature. The model might use information about feature interactions in order to *derive* these weights (e.g. the covariance between inputs plays a key role in the ordinary least squares solution), but nevertheless once the model has been fit there is a very simple and clear relationship between perturbations of the input and resulting effects on the output.

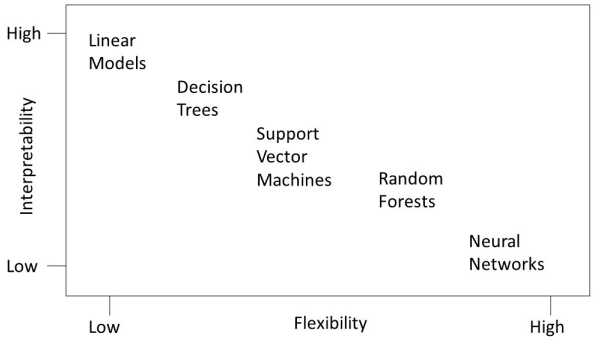

Figure 3. A representation of the tradeoff between flexibility and interpretability, using different statistical learning methods. In general, as the flexibility of a method increases, its interpretability decreases.

Decision trees are another popular method, in which rules are represented on a tree-like structure where each branch represents some "test" on the input and is therefore easy to interpret. Figure 4 shows an example where the objective is to classify points from two classes (TRUE or FALSE). Once we train a decision tree on this dataset, the resulting tree model (Figure 5) runs tests at each of its node on one of the two features x_1 and x_2, using cut-off values to branch and ultimately produce a final decision at the bottom of the tree. In this case, *training* the model amounts to finding the optimal cut-off values at each node in order to minimize some cost function of our choice.

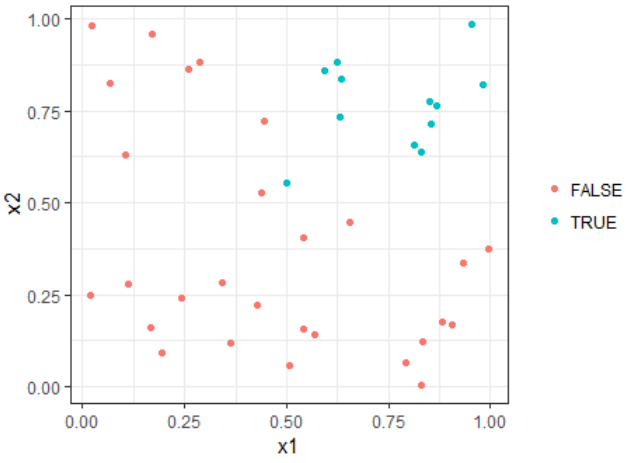

Figure 4. Simple example of a dataset with two features x_1 and x_2 spread over two classes (TRUE or FALSE).

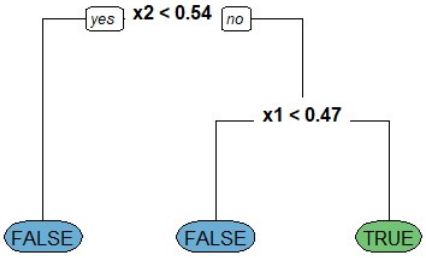

Figure 5. Example of a decision tree after it was trained on the data from Figure 4. Starting from the root node at the top, each node will test one of the two features from the data in order to correctly classify each point.

3.2 Black Box Models

Random forests and neural networks, on the other hand, are examples of models that are usually harder to interpret. Unlike the previous group, the complicated or intricate structure of these models means their decisions cannot be easily traced back to particular inputs or combinations of inputs.

Random forests, aptly named as they consist of large collections of individual decision trees, are an example of so-called *ensemble* models. Ensembles work by combining the outputs of a collection of sub-models to reduce idiosyncratic variance. Fundamental to this approach is the idea that the sub-models are meaningfully diverse from one another so that their outputs are roughly un-correlated. Random forests achieve this diversification by construction, training each tree on random subsets of the data, or sometimes using only random feature subsets when deriving node splits. The upshot is that where individual decision trees are generally quite interpretable, random forests introduce quite a lot of opacity by combining those trees (and introducing randomization along the way).

Neural networks perform a hierarchical feature engineering using chained nonlinear function compositions. Starting from the raw input data, successive 'layers' are fed the output from previous layers, mixing and deforming that output in such a way that the final layer can more easily make some task-specific decision. Figure (6) shows visually how these structures can become quite ornate. Through their successive layers, neural networks can fit the data in intricate and often new ways. However, this is at the expense of interpretability as the succession of layers renders the relationship between inputs and outputs opaque.

Finally, in addition to their more convoluted nature, these 'black-box' models are usually applied to data with many features or complicated structure, making it even harder to sift through the vast parameter space and gain insight. As described earlier, it may also come at the expense of a higher risk of overfitting [Nguyen et al. (2015)]. Besides, the large number of meta-parameters to fine-tune for these models also mean that they must be trained on very large datasets, which can be hard to come by in some applications.

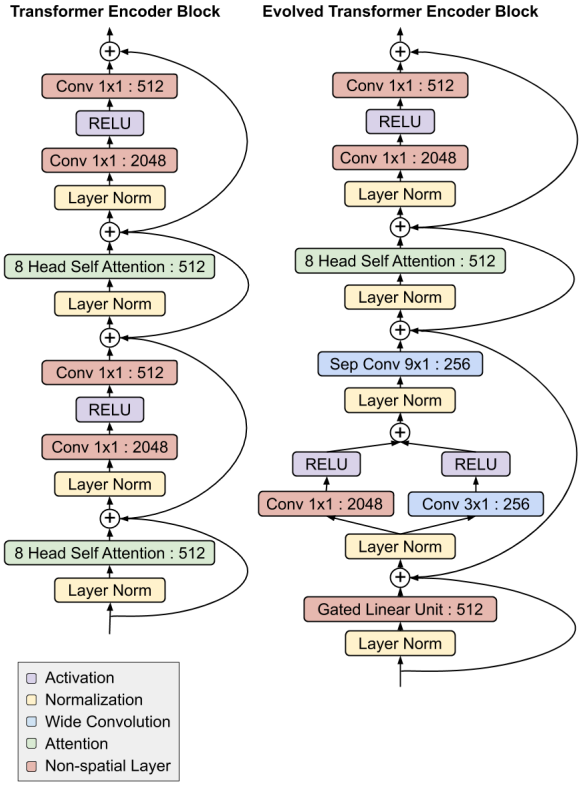

Figure 6. Examples of complicated neural network architectures [So (2019)]. Each block represents some kind of transformation of the data (occasionally nonlinear), and the arrows indicate the direction in which data passes through the network (notice that data occasionally "skips" some layers and gets recombined downstream). As one can imagine, there is no easy way to understand the relationship between the inputs and the final output in a model like this!

Significant research is being done in the field to make deep learning more interpretable and give the user a meaningful global view of each layer in the network. These techniques attempt to carry meaning from one layer to the next via the use of creative visualizations. This is often used side-by-side with the chain of transformations from the input features to the output of the model in order to gain some insights regarding the final decision.

3.3 Scope and Evaluation of Interpretability

In some instances, one may only need to interpret *locally* a specific decision from a 'black-box' model. These models, whose fitting function f may be quite complicated, can be approximated over some small region using a simpler model. In other words, one can fit a simple model between a subset of the input and *the output of the black-box model* to understand its behavior locally. Even if the simple model does not generalize well to the entire dataset, as long as it behaves similarly to the black-box model locally, it can help us understand things better. This procedure can even be extended globally by fitting the simpler model to the output of the black-box

model on the entire dataset so that the simple model becomes a *surrogate* model. Of course, due to the simpler nature of the surrogate model, it can be hard to extend it to the entire dataset while keeping the approximation faithful, and so local approximation is often preferred [Ribeiro et al. (2016)].

It not exactly clear how to express the concept of interpretability in terms of a statistical objective function. However, some research is being done in the field to 'grade' models and assess the interpretability of their decisions after the fact. One example is in the medical field, where neural networks can help radiologists to locate a tumor or nodules in the lungs. The models are graded in this case not only on their ability to correctly provide a 'yes/no' diagnosis, but on their ability to actually locate the tumor. Doctors can then be asked to evaluate *post hoc* the interpretabilty of the models by inspecting the indicated location. (Of course, such evaluation requires the user be at least as good as the model and have the required technical background.) Protocols of this kind place the model in a specific and understandable role so that doctors are able to provide patients with the best possible care.

Pure model accuracy is not the only goal in all machine learning applications. How the models come up with a result can often be just as important. Users may learn a lot by coming to understand how models make their decisions, gaining new insight, and getting valuable feedback for improving the performance of the model itself. The subject is an increasingly active area of research, including work on feature visualization [Olah et al. (2019), Goldstein et al. (2015)], example-based reasoning [Wachter et al. (2017), Kim et al. (2016), Goodfellow et al. (2014)], surrogate modeling [Ribeiro et al. (2016)], partial dependence plots [Zhao and Hastie (2019)], and even applications of game theory [Lundberg and Lee (2017)].

4. Conclusion

While the issue of interpretability has become nearly ubiquitous in all applications of statistical modeling, its relevance and the requirements of the modeler can vary widely by field. Frequently the task is too sensitive to be able to rely on the output of a black box. For other applications, the best choice is not always clear. In quantitative investing, the reasons for caring about interpretability are numerous. The financial analyst should always be concerned about model robustness, for example, given the fat tails and non-stationarities in his or her data. On the other side, investors might reasonably demand to know why they are positioned in a certain way.

Nevertheless, there are simultaneously good reasons for being interested in less interpretable models. As the field becomes more crowded, managers may need to push the envelope to find new sources of returns. Not to mention, the increasing size and variety of data that quantitative researchers are now processing necessitate more complicated methods in some cases.

There is no easy answer, and the problem is unlikely ever to be 'solved.' As machines become more powerful, these ideas will need to be better understood, and the trade-offs managed with careful attention.

References

A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 2015. doi: 10.1080/10618600.2014.907095.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples, 2014.

B. Kim, R. Khanna, and O. O. Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2280–2288. Curran Associates, Inc., 2016.

S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

C. Molnar. Interpretable machine learning: A guide for making black box models explainable. https://christophm.github.io/interpretable-ml-book/, 2019. Accessed: 2019-12-11.

A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

C. Olah, A. Mordvintsev, and L. Schubert. Feature visualization: how neural networks build up their understanding of images. https://distill.pub/2017/feature-visualization, 2019. Accessed: 2019-12-11.

M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?": Explaining the predictions of any classifier, 2016.

K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *ArXiv e-prints*, 2014.

D. So. Applying automl to transformer architectures. https://ai.googleblog.com/2019/06/applying-automl-to-transformer.html, 2019. Accessed: 2019-12-11.

S. Wachter, B. Mittelstadt, and C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr, 2017.

Q. Zhao and T. Hastie. Causal interpretations of black-box models. *Journal of Business & Economic Statistics*, 0(0): 1–10, 2019. doi: 10.1080/07350015.2019.1624293.

Legal Disclaimer