# Transparent Model Distillation

**Sarah Tan** [1] **Rich Caruana** **Giles Hooker** **Albert Gordo**

## Abstract

Model distillation was originally designed to distill knowledge from a large, complex teacher model to a faster, simpler student model without significant loss in prediction accuracy. We investigate model distillation for another goal – transparency – investigating if fully-connected neural networks can be distilled into models that are transparent or interpretable in some sense. Our teacher models are multilayer perceptrons, and we try two types of student models: (1) tree-based generalized additive models (GA2Ms), a type of boosted, short tree (2) gradient boosted trees (GBTs). More transparent student models are forthcoming. Our results are not yet conclusive. GA2Ms show some promise for distilling binary classification teachers, but not yet regression. GBTs are not "directly" interpretable but may be promising for regression teachers. GA2M models may provide a computationally viable alternative to additive decomposition methods for global function approximation.

## 1. Introduction

Model distillation was originally proposed to distill knowledge from a large, complex model (teacher) to a faster, simpler model (student) without significant loss in prediction accuracy (Bucilua et al., 2006; Hinton et al., 2015; Ba & Caruana, 2014). Practical reasons for model distillation include test-time evaluation on memory-constrained devices or time-critical applications, model selection and reduction in the cost of collecting features, and theoretical reasons include cleaning noisy labels, richer supervision from soft targets compared to hard 0/1 targets for classification, etc. (Bucilua et al., 2006).

We investigate model distillation for another reason – transparency. Specifically, we are interested in whether fully connected neural network teacher models could be distilled into student models that are transparent or interpretable in some sense. We call this setting transparent model distillation and describe in Section 1.1 related work where this has been performed.

The notion of transparency or interpretability in machine learning is still not well-defined (Lipton, 2016; Doshi-Velez & Kim, 2017). While we started this project with the goal of investigating if a particular class of models, tree-based generalized additive models (Lou et al., 2012; 2013; Caruana et al., 2015), claimed to be more interpretable than tree ensembles while achieving comparable accuracy results on several data sets could be good student models for the transparent model distillation setting, it is debatable whether certain classes of models[1] could be claimed to be more interpretable than others, and if complex versions[2] of these models could still be claimed as interpretable. It could still be interesting to investigate if non-neural networks could distill neural networks and we investigate this in this paper.

### 1.1. Related Work

**Global function approximation:** We note the distinction between the global model distillation approaches examined here and local sensitivity analysis (Ribeiro et al., 2016), which aim to provide explanations of individual predictions. In particular, for non-linear response surfaces, local explanations may miss globally important features that have little local influence. By contrast, transparent distillation models may provide a diagnostic of larger effects in the data that can be used to understand the influences of individual features on the global pattern of predictions. The additive model methods we employ here are related to the functional ANOVA decomposition defined, for example, in (Gu, 2003), adapted for machine learning in (Hooker, 2007) and used in settings such as hyperparameter optimization to examine the importance and interactions between hyperparameters (Hutter et al., 2014). Rather than directly decomposing a learned function, we propose to distill it with a generalized additive model, thereby avoiding the need for the high dimensional integrals employed in a formal additive decomposition (c.f. (Hooker, 2007)).

[1]Working paper. Comments welcome.. Correspondence to: Sarah Tan <ht395@cornell.edu>, Rich Caruana <rcaruana@microsoft.com>.

[1]Some typical classes of models claimed to be interpretable are linear regression and decision trees.

[2]For example, linear regression with a vast number of features or decision trees with a large number of nodes.

**Neural network student models:** Bucilua, Caruana and Niculescu-Mizil used a fully-connected net to mimic a large ensemble of models by matching logits between teacher and student models (Bucilua et al., 2006). Ba and Caruana distilled an ensemble of deep convolutional nets into a shallow fully-connected net (Ba & Caruana, 2014). Hinton, Vinyals and Dean distilled an ensemble of deep non-convolutional nets into a single deep fully-connected net, and introduced the notion of temperature distillation which generalizes matching logits (Hinton et al., 2015). Many other papers distilling neural networks into other variants of neural networks followed (Romero et al., 2015; Urban et al., 2017).

**Tree-based student models:** Perhaps the first to distill neural networks into trees were Craven and Shavit who distilled a one-hidden-layer perceptron into a decision tree (Craven & Shavlik, 1995). Che et al. distilled a multilayer perceptron into a gradient boosted tree, then used partial dependence plots (Friedman, 2001) to interpret the predictions from the tree (Che et al., 2016). Most recently, Frosst and Hinton attempted to distill a shallow convolutional net into a type of soft decision tree they propose (Frosst & Hinton, 2017), however the student model did not attain accuracy close to its teacher.

## 2. Method

Let $\mathcal{D} = \{(y_i, \boldsymbol{x}_i)\}_{i=1}^N$ be a data set of size $N$, where $y$ is a target and $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})$ is a vector of $p$ features for observation $i$, and $x_j$ is the $j$th variable in feature space. $y$ can be continuous or binary. The features are column-based features (not structured features such as text or images), for example age, medical history, etc.

We train non-convolutional, fully-connected neural networks, also called multilayer perceptrons (MLPs), to predict $y_i$ from $\boldsymbol{x}_i$. These are our teacher models. We try two kinds of transparent student models: (1) tree-based generalized additive models (GA2Ms), a type of bagged, short tree learned using gradient boosting (Lou et al., 2012; 2013; Caruana et al., 2015); (2) gradient boosted trees (GBTs), as used by Che et al. (Che et al., 2016). More transparent student models are forthcoming.

Section 2.1 describes the training procedure for the MLP teacher models and Section 2.2 details how the MLP teacher models are distilled into the transparent student models. Section 2.3 elaborates on one of the student models - GA2Ms.

### 2.1. Training MLP Teacher Models

We are interested in whether modern neural networks that are deeper (e.g. than the 1-hidden-layer multilayer perceptrons of (Craven & Shavlik, 1995)), have more complex architectures, and trained using modern techniques, including dropout (Srivastava et al., 2014), batch normalization (Ioffe & Szegedy, 2015), weight decay, etc. could be distilled into transparent student models. Hence, it is imperative that we attempt to train MLP teacher models that are as accurate as possible.

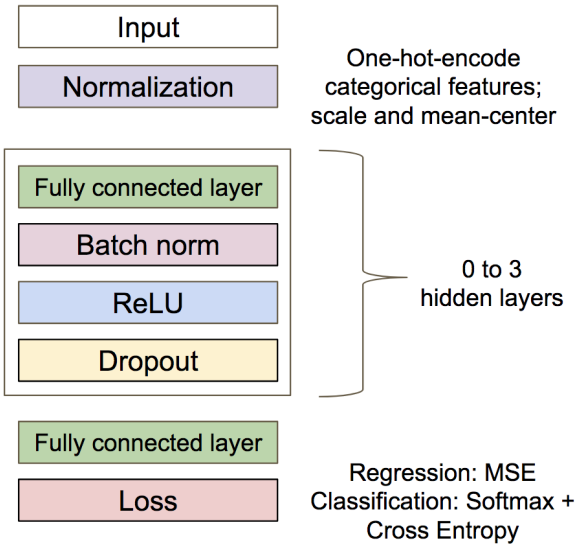


*Figure 1.* Architecture of the MLP.

**Architecture:** Figure 1 describes the MLP architecture. Each MLP teacher model contains up to three hidden layers consisting of a fully connected layer followed by batch normalization, a ReLU nonlinearity, and dropout. The last fully-connected layer makes the final prediction by projecting the data into a single dimension (for regression) or into two dimensions (for binary classification). The batch normalization layer was helpful to train models of two and three hidden layers, while dropout was used to reduce overfitting in some data sets.

**Training:** We use mean square error loss for regression and cross entropy with softmax for classification. The optimization is done with Adam (Kingma & Ba, 2015), which led to better results than SGD with momentum on our data sets. We train for 300 epochs, compute the validation loss at the end of every epoch, and finally keep the model from the epoch that yielded the lowest validation error. We also apply learning rate decay, dividing the current learning rate by ten whenever the validation loss did not improve after $n$ epochs. In classification data sets where the data is unbalanced we oversample the minority class.

We used random search to find the optimal architecture and training parameters for the MLP teacher models. In particular, we validated the number of hidden units per layer, the initial learning rate, the batch size, and the dropout probability. The optimal number of hidden layers was found using exhaustive search in the range $[0 - 3]$. The optimal

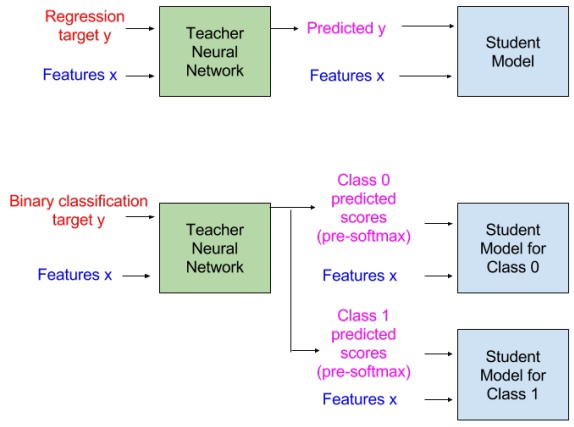


*Figure 2.* Distillation setup. Top: Distilling a regression teacher Bottom: Distilling a binary classification teacher

parameters are in Table 2.

### 2.2. Distillation Setup

We investigate two settings: (1) distilling a regression teacher model; (2) distilling a binary classification teacher. Several distillation approaches for the latter case have been proposed, including:

1. Use class probabilities predicted by the teacher model as "soft" targets to train the student model on
2. Use logits (scores before applying the softmax) predicted by the teacher model as soft targets, minimizing squared error between these logits and the student model's logits
3. Temperature distillation: raise the temperature of the teacher model's softmax until the teacher produces sufficiently soft targets, then use the same temperature when training the student model.

Hinton et al. observed that matching logits is actually a special case of temperature distillation (Hinton et al., 2015). We try matching logits. The more general case of temperature distillation is to be investigated[3]. We are also investigating the use of generative adversarial networks for distillation. The case of distilling a regression teacher model has been less investigated in the literature. We use the predicted targets from the regression teacher model as targets to train the student model on. Figure 2 describes the teacher and student models' inputs and outputs for both regression and binary classification.

[3]In some public talks, Hinton suggested that matching logits performs similarly to temperature distillation in several cases.

### 2.3. Tree-Based Generalized Additive Models (GA2Ms)

GA2Ms were proposed by Lou and Caruana (Lou et al., 2012; 2013; Caruana et al., 2015). These models, claimed to be more interpretable than tree ensembles while achieving comparable accuracy on several data sets, are a type of bagged, short trees learned using gradient boosting. Their claim to transparency stems from their additive form[4]:

$$g(y) = \beta_0 + \sum h_j(x_j) + \sum h_{jk}(x_j, x_k)$$

where each term $h_j(x_j)$ is a shallow tree restricted to only operate on one feature, and $h_{jk}(x_j, x_k)$ is again a shallow tree but operating on two features. $h_j$ is called the shape function of feature $x_j$ and can be plotted against $x_j$ in graphs such as the red or green lines in the right hand side of Figure 3. $h_{jk}$ is the pairwise interaction of $x_j$ and $x_k$ and can be visualized in a heat map. This allows the contribution of any one feature to the prediction to be directly examined, making the model transparent. Multiple terms are learned together using gradient boosting to obtain an additive formulation, hence the name generalized additive models (GAMs). However, unlike classical GAMs where features are shaped using splines, tree-based GAMs shape features using short trees.

## 3. Experimental Setup

**Data:** We use publicly available data sets from UCI and Kaggle, two each for regression and binary classification. Table 1 describes them. The car lemon data is from Kaggle, and its target is whether the car is a lemon (bad buy) or not. The UCI adult income data set (Asuncion & Newman, 2007) predicts if an adult earns above or below a certain income threshold. As for regression, the bikeshare count data set is from UCI and the store sales data set is from Kaggle.

**Data normalization:** All categorical features are one-hot-encoded. Every feature is then normalized by scaling and mean-centering, specifically:

$$\tilde{x}_j = \frac{x_j}{s_j} - m_j$$

where $s_j = \max_{i \in \text{train-set}} (|x_{ij}|)$ and $m_j = \text{mean}(\frac{x_j}{s_j})$

**Data partitioning:** We split all data sets into three partitions: train (70% of observations), validation (15%), and test (15%). We perform this partitioning four times to obtain different train / validation / test splits. We use only one of these splits to tune, on the validation-set, the parameters for the MLP teacher and transparent student models. With the optimal parameters tuned, we then use the other three splits to train the models to get a sense of the variability of the

[4]$g$ is the logistic link function for classification. For regression, $g$ is the identity function.

*Table 1.* Data sets used. For data with binary classification tasks, class imbalance is described. For regression, the range of the continuous target is described.

| Data | Task | # Observations | # Features | Class imbalance / Range |
|---|---|---|---|---|
| Car lemona | Binary Classification | 69k | 25 | Yes: 90%; No: 10% |
| Adult income | Binary Classification | 49k | 14 | Yes: 24%; No: 76% |
| Bikeshare count | Regression | 17k | 12 | [0.01 - 9.77] |
| Store sales | Regression | 576k | 14 | [0 - 38.7] |

*Table 2.* Optimal parameters for number of hidden units of each layer (# hidden units, in $[64 - 512]$), learning rate (LR, in $10^{[-5,-2]}$), batch size (BS, in $2^{\{5,6,7,8\}}$), and dropout probability (DP, in $[0 - 0.7]$) in each dataset.

| Data | # hidden units | LR | BS | DP |
|---|---|---|---|---|
| Car lemon | [251, 511] | 2.36e-5 | 32 | 0.7 |
| Adult income | [236, 470] | 1.31e-5 | 64 | 0.09 |
| Bikeshare count | [505, 395] | 1.06e-5 | 64 | - |
| Store sales | [452, 462, 291] | 2.47e-5 | 32 | - |

results. In the MLP case, each of these splits has a different random initialization of the model weights. We report the MLP parameters that works best on average in Table 2. The GA2M and GBT models were trained using 5000 gradient boosting iterations, with the optimal number of iterations ($\leq$ 5,000) selected based on minimum validation-set loss.

**Evaluation Metrics:** We assess the student models generated using three criteria: (1) accuracy, the notion that if the student model is to replace the teacher model, it should perform at comparable or greater accuracy than the teacher; (2) fidelity, the notion that the student model should match the teacher model (Craven & Shavlik, 1995), using the closeness of the student model to the teacher's predictions; (3) interpretability. We elaborate on interpretability metrics directly in Section 4.4 as we discuss the results.

# 4. Results

## 4.1. Learned Architectures for MLP Teacher Models

Table 2 shows the learned parameters. In general, using 2 or 3 hidden layers outperformed models with 0 or 1 hidden layer. We also found the number of optimal hidden units per layer to be quite high compared to the number dimensionality of the inputs, probably to allow for different feature itneractions. The optimal learning rate is quite similar between all datasets and tasks, as is the optimal batch size. We applied dropout to the classification models (which initially overfitted significantly more than the regression models), and found that the chosen dropout probability significantly reduced the gap between the validation and train accuracy.

## 4.2. Accuracy of Student Models

To determine if the student models can replace their teacher models, we evaluated the student models on the original test-set targets. Table 3 provides the results. For binary classification tasks, GA2M students achieved AUC numbers equal to or slightly higher than their MLP teachers. GBT students also achieved comparable AUC numbers. For regression tasks, however, GA2M models were not able to achieve RMSE numbers comparable to their MLP teachers whereas GBT could for one data set - store sales.

We also train GBT and GA2M models directly on the original targets instead of soft target outputs from the MLP teachers. Comparing these models trained on the original targets and their student counterparts trained on soft targets, for the car lemon data (binary classification) and bikeshare count data (regression), both GBT and GA2M models improved slightly when trained on soft targets compared to original targets, but the opposite was true for the adult income data (binary classification) and store sales data (regression).

## 4.3. Fidelity of Student Models

To determine how close the student models are to their teachers, we compare, on the test set, the difference between the teachers' predictions (which serve as inputs to the student models) and the students' predictions. The MLP teacher predictions used are pre-softmax scores for classification and predicted targets for regression. Table 4 provides the results. For classification tasks, fidelity is assessed for each class. In general, the GBT models are higher fidelity than GA2M models. This is unsurprising as GBT models also model higher-order interactions, not just pairwise interactions that GA2M models stop at. This may, however, come at the expense of interpretability which we study further in Section 4.4.

## 4.4. Interpretability of Student Models

Since GBTs are not "directly" interpretable (Che et al., 2016), Che et al. used partial dependence plots (Friedman,

*Table 3.* Accuracy of MLP teacher model (first column of numbers) and two student models - GBT and GA2M (last two columns). For comparison, accuracy results of these two student model classes, when trained on the original targets, are also provided (second and third columns of numbers). The first two rows are AUC numbers; the last two rows are RMSE numbers.

| | | Teacher | | | Student [tested on original target] | |
|---|---|---|---|---|---|---|
| Teacher task | Data | MLP | GBT | GA2M | GBT | GA2M |
| Binary Classification (AUC) | Car lemon | $0.68 \pm 0.01$ | $0.66 \pm 0.01$ | $0.69 \pm 0.01$ | $0.68 \pm 0.01$ | $\mathbf{0.70 \pm 0.02}$ |
| | Adult income | $\mathbf{0.90 \pm 0.01}$ | $0.92 \pm 0.02$ | $0.93 \pm 0.01$ | $0.89 \pm 0.02$ | $\mathbf{0.90 \pm 0.01}$ |
| Regression (RMSE) | Bikeshare count | $\mathbf{0.43 \pm 0.02}$ | $0.57 \pm 0.01$ | $0.69 \pm 0.02$ | $0.54 \pm 0.01$ | $0.68 \pm 0.02$ |
| | Store sales | $\mathbf{1.14 \pm 0.02}$ | $0.84 \pm 0.03$ | $1.03 \pm 0.01$ | $\mathbf{1.14 \pm 0.03}$ | $1.2 \pm 0.02$ |

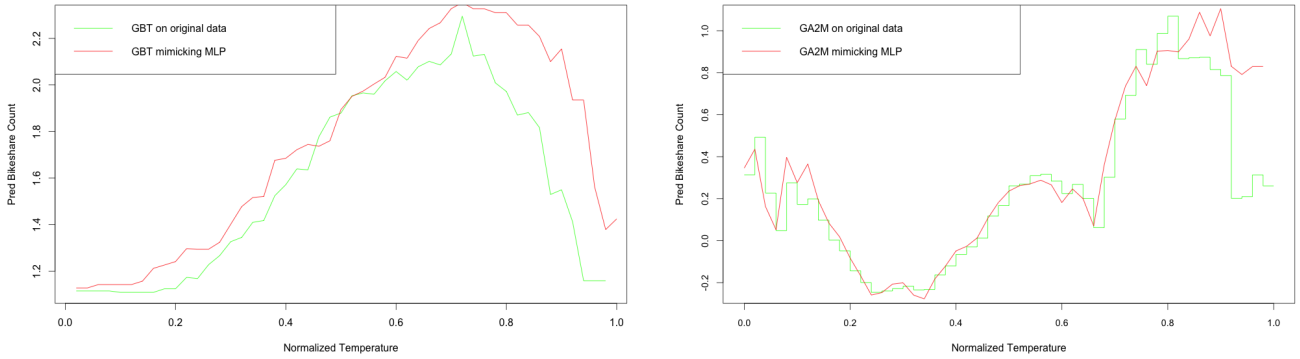


*Figure 3.* Interpreting the relationship between one of the features in the bikeshare count data - weather temperature - and the predicted target of bikeshare count. Left: GBT partial dependence plot for the feature and predicted target. Right: GA2M learned relationship between the feature and predicted target.

2001) to understand the relationship between a feature and predicted targets. Partial dependence (PD) plots how the average predicted target changes as the feature changes over its marginal distribution. Let $x_j^C$ be the complement set of $x_j$, namely the set of all features besides $x_j$. Concretely, partial dependence of feature $x_j$ is defined as:

$$PD_{x_j} = \int f(x_j, x_j^C) dP(X_j^C) \tag{1}$$

where $dP(X_j^C)$ is the marginal distribution of $x_j^C$. However, neither the true prediction function $f$ nor $dP(X_j^C)$ is known (Goldstein et al., 2015), hence Equation 1 has to be estimated, typically using:

$$PD_{x_j} = \frac{1}{|\text{train-set}|} \sum_{i \in \text{train-set}} \hat{f}(x_j, x_j^C) \tag{2}$$

We calculate partial dependence for our GBT models. For GA2Ms, we directly plot the learned relationship between features and predicted targets. Figure 3 is for one of the 12 features - weather temperature (normalized) - in the bikeshare count data. On the left of the figure are the partial dependence plots for the GBT models - the green line is for the GBT model learned directly on the original targets, the red line is for the GBT student model learned on the soft targets; on the right is how the GA2M model shaped the relationship between this feature and the target. Similarly, the green line is for the GA2M model learned directly on the original targets and the red line is for the GA2M student model. We draw the red and green lines on the same plot to provide an idea of how a particular model class performs when trained on the original target compared to soft targets.

At first glance, the relationship described in the partial dependence plot could makes sense - at lower temperatures, the demand for bikes is low. This demand increases as temperature increases, then tepers off as temperature increases too much. However, that the partial dependence plot and the the GA2M plot differs significantly suggests that a more nuanced approach to interpret the feature-target relationships is needed; it is known that partial dependence can be misleading when $x_j$ and $x_j^C$ are correlated or interact, and it likely extrapolates off the manifold of the training data (Hooker, 2007; Goldstein et al., 2015). More investigation is needed

*Table 4.* Fidelity of student models to MLP teacher, measured in terms of difference (in RMSE) between MLP teacher predictions (which serve as inputs to the student model) and the student models' predictions.

| | Fidelity (RMSE) | |
|---|---|---|
| Data | GBT | GA2M |
| Car lemon | Class 0: $0.07 \pm 0.02$<br>Class 1: $0.07 \pm 0.03$ | Class 0: $0.15 \pm 0.03$<br>Class 1: $0.16 \pm 0.04$ |
| Adult income | Class 0: $0.33 \pm 0.08$<br>Class 1: $0.40 \pm 0.09$ | Class 0: $0.32 \pm 0.07$<br>Class 1: $0.40 \pm 0.10$ |
| Bikeshare count | $0.40 \pm 0.01$ | $0.58 \pm 0.01$ |
| Store sales | $0.46 \pm 0.02$ | $0.70 \pm 0.03$ |

to determine if this feature is highly correlated with another, and while partial dependence ignores this correlation, the GA2M model does not.

## 5. Discussion and Ongoing Work

The results are not conclusive thus far. The GA2M student models achieve good accuracy on some data sets (binary classification) and are able to replace their teachers, but not yet regression teachers. We are trying the approach on more data. GBTs are not "directly" interpretable but may be promising for regression teachers.

However, that the GBT partial dependence plot and GA2M plot differ significantly for some feature values means that detailed investigation into features that potentially correlate is needed. Partial dependence plots can be constructed for any prediction function, hence we can attempt to construct them for neural network teachers as well to "directly" interpret the teacher. That partial dependence plots for neural networks have been less popular may be due to the computational load of querying the neural network for each point that makes up the partial dependence plot.

We note that all our neural network teacher models only have up to three hidden layers so far. We are investigating the case of deeper teacher models.

Finally, the distillation setup tried was a simple approach of matching logits. The more general approach of temperature distillation as well as distillation using GANs is ongoing work.

## References


Asuncion, Arthur and Newman, David. Uci machine learning repository, 2007.

Ba, Jimmy and Caruana, Rich. Do deep nets really need to be deep? In *NIPS*, 2014.

Bucilua, Cristian, Caruana, Rich, and Niculescu-Mizil, Alexandru. Model compression. In *ICDM*, 2006.

Caruana, Rich, Lou, Yin, Gehrke, Johannes, Koch, Paul, Sturm, Marc, and Elhadad, Noemie. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD*, 2015.

Che, Zhengping, Purushotham, Sanjay, Khemani, Robinder G., and Liu, Yan. Interpretable deep models for ICU outcome prediction. In *American Medical Informatics Association (AMIA) Annual Symposium*, 2016.

Craven, Mark W. and Shavlik, Jude W. Extracting tree-structured representations of trained networks. In *NIPS*, 1995.

Doshi-Velez, Finale and Kim, Been. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Friedman, Jerome H. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29 (5):1189–1232, 2001.

Frosst, Nicholas and Hinton, Geoffrey. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

Goldstein, Alex, Kapelner, Adam, Bleich, Justin, and Pitkin, Emil. 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.

Gu, C. *Smoothing Spline ANOVA Models*. Springer, New York, 2003.

Hinton, Geoff, Vinyals, Oriol, and Dean, Jeff. Distilling the knowledge in a neural network. *NIPS Deep Learning Workshop*, 2015.

Hooker, Giles. Generalized functional ANOVA diagnostics for high dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics*, 16: 709–732, 2007.

Hutter, Frank, Hoos, Holger, and Leyton-Brown, Kevin. An efficient approach for assessing hyperparameter importance. In *ICML*, 2014.

Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Kingma, Diederik P. and Ba, Jimmy. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Lipton, Zachary C. The mythos of model interpretability. In *ICML Workshop on Human Interpretability in Machine Learning*, 2016.

Lou, Yin, Caruana, Rich, and Gehrke, Johannes. Intelligible models for classification and regression. In *KDD*, 2012.

Lou, Yin, Caruana, Rich, Gehrke, Johannes, and Hooker, Giles. Accurate intelligible models with pairwise interactions. In *KDD*, 2013.

Ribeiro, Marco Tulio, Singh, Sameer, and Guestrin, Carlos. "why should i trust you?": Explaining the predictions of any classifier. In *KDD*, 2016.

Romero, Adriana, Ballas, Nicolas, Ebrahimi Kahou, Samira, Chassang, Antoine, Gatta, Carlo, and Bengio, Yoshua. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.

Urban, Gregor, Geras, Krzysztof J., Ebrahimi Kahou, Samira, Aslan, Ozlem, Wang, Shengjie, Mohamed, Abdelrahman, Philipose, Matthai, Richardson, Matt, and Caruana, Rich. Do deep convolutional nets really need to be deep and convolutional? In *ICLR*, 2017.