Journal of Economic Perspectives—Volume 31, Number 2—Spring 2017—Pages 87–106

# Machine Learning: An Applied Econometric Approach

Sendhil Mullainathan and Jann Spiess

Machines are increasingly doing "intelligent" things: Facebook recognizes faces in photos, Siri understands voices, and Google translates websites. The fundamental insight behind these breakthroughs is as much statistical as computational. Machine intelligence became possible once researchers stopped approaching intelligence tasks procedurally and began tackling them empirically. Face recognition algorithms, for example, do not consist of hard-wired rules to scan for certain pixel combinations, based on human understanding of what constitutes a face. Instead, these algorithms use a large dataset of photos labeled as having a face or not to estimate a function $f(x)$ that predicts the presence $y$ of a face from pixels $x$. This similarity to econometrics raises questions: Are these algorithms merely applying standard techniques to novel and large datasets? If there are fundamentally new empirical tools, how do they fit with what we know? As empirical economists, how can we use them?[1]

We present a way of thinking about machine learning that gives it its own place in the econometric toolbox. Central to our understanding is that machine learning

[1] In this journal, Varian (2014) provides an excellent introduction to many of the more novel tools and "tricks" from machine learning, such as decision trees or cross-validation. Einav and Levin (2014) describe big data and economics more broadly. Belloni, Chernozhukov, and Hanson (2014) present an econometrically thorough introduction on how LASSO (and close cousins) can be used for inference in high-dimensional data. Athey (2015) provides a brief overview of how machine learning relates to causal inference.

■ *Sendhil Mullainathan is the Robert C. Waggoner Professor of Economics and Jann Spiess is a PhD candidate in Economics, both at Harvard University, Cambridge, Massachusetts. Their email addresses are mullain@fas.harvard.edu and jspiess@fas.harvard.edu.*

† For supplementary materials such as appendices, datasets, and author disclosure statements, see the article page at
https://doi.org/10.1257/jep.31.2.87 doi=10.1257/jep.31.2.87

not only provides new tools, it solves a different problem. Machine learning (or rather "supervised" machine learning, the focus of this article) revolves around the problem of *prediction*: produce predictions of $y$ from $x$. The appeal of machine learning is that it manages to uncover generalizable patterns. In fact, the success of machine learning at intelligence tasks is largely due to its ability to discover complex structure that was not specified in advance. It manages to fit complex and very flexible functional forms to the data without simply overfitting; it finds functions that work well out-of-sample.

Many economic applications, instead, revolve around *parameter estimation*: produce good estimates of parameters $\beta$ that underlie the relationship between $y$ and $x$. It is important to recognize that machine learning algorithms are not built for this purpose. For example, even when these algorithms produce regression coefficients, the estimates are rarely consistent. The danger in using these tools is taking an algorithm built for $\hat{y}$, and presuming their $\hat{\beta}$ have the properties we typically associate with estimation output. Of course, prediction has a long history in econometric research—machine learning provides new tools to solve this old problem.[2] Put succinctly, machine learning belongs in the part of the toolbox marked $\hat{y}$ rather than in the more familiar $\hat{\beta}$ compartment.

This perspective suggests that applying machine learning to economics requires finding relevant $\hat{y}$ tasks. One category of such applications appears when using new kinds of data for traditional questions; for example, in measuring economic activity using satellite images or in classifying industries using corporate 10-K filings. Making sense of complex data such as images and text often involves a prediction pre-processing step. In another category of applications, the key object of interest is actually a parameter $\beta$, but the inference procedures (often implicitly) contain a prediction task. For example, the first stage of a linear instrumental variables regression is effectively prediction. The same is true when estimating heterogeneous treatment effects, testing for effects on multiple outcomes in experiments, and flexibly controlling for observed confounders. A final category is in direct policy applications. Deciding which teacher to hire implicitly involves a prediction task (what added value will a given teacher have?), one that is intimately tied to the causal question of the value of an additional teacher.

Machine learning algorithms are now *technically* easy to use: you can download convenient packages in R or Python that can fit decision trees, random forests, or LASSO (Least Absolute Shrinkage and Selection Operator) regression coefficients. This also raises the risk that they are applied naively or their output is misinterpreted. We hope to make them *conceptually* easier to use by providing a crisper

[2] While the ideas we describe as central to machine learning may appear unfamiliar to some, they have their roots and parallels in nonparametric statistics, including nonparametric kernel regression, penalized modeling, cross-validation, and sieve estimation. We refer to Györfi, Kohler, Krzyzak, and Walk (2002) for a general overview, and to Hansen (2014) more specifically for counterparts in sieve estimation.

understanding of how these algorithms work, where they excel, and where they can stumble—and thus where they can be most usefully applied.[3]

## How Machine Learning Works

Supervised machine learning algorithms seek functions that predict well out of sample. For example, we might look to predict the value $y$ of a house from its observed characteristics $x$ based on a sample of $n$ houses $(y_i, x_i)$. The algorithm would take a loss function $L(\hat{y}, y)$ as an input and search for a function $\hat{f}$ that has low expected prediction loss $E_{(y,x)}[L(\hat{f}(x), y)]$ on a *new* data point from the same distribution. Even complex intelligence tasks like face detection can be posed this way. A photo can be turned into a vector, say a 100-by-100 array so that the resulting $x$ vector has 10,000 entries. The $y$ value is 1 for images with a face and 0 for images without a face. The loss function $L(\hat{y}, y)$ captures payoffs from proper or improper classification of "face" or "no face."

Familiar estimation procedures, such as ordinary least squares, already provide convenient ways to form predictions, so why look to machine learning to solve this problem? We will use a concrete application—predicting house prices—to illustrate these tools. We consider 10,000 randomly selected owner-occupied units from the 2011 metropolitan sample of the American Housing Survey. In addition to the values of each unit, we also include 150 variables that contain information about the unit and its location, such as the number of rooms, the base area, and the census region within the United States. To compare different prediction techniques, we evaluate how well each approach predicts (log) unit value on a separate hold-out set of 41,808 units from the same sample. All details on the sample and our empirical exercise can be found in an online appendix available with this paper at http://e-jep.org.

Table 1 summarizes the findings of applying various procedures to this problem. Two main insights arise from this table. First, the table highlights the need for a hold-out sample to assess performance. In-sample performance may overstate performance; this is especially true for certain machine learning algorithms like random forests that have a strong tendency to overfit. Second, on out-of-sample performance, machine learning algorithms such as random forests can do significantly better than ordinary least squares, even at moderate sample sizes and with a limited number of covariates. Understanding machine learning, though, requires looking deeper than these quantitative gains. To make sense of how these

[3] This treatment is by no means exhaustive: First, we focus specifically on "supervised" machine learning where prediction is central, and do not discuss clustering or other "unsupervised" pattern recognition techniques. Second, we leave to more specialized sources the more hands-on practical advice, the discussion of computational challenges that are central to a computer-science treatment of the subject, and the overview of cutting-edge algorithms.

*Table 1*
**Performance of Different Algorithms in Predicting House Values**

| | *Prediction performance ($R^2$)* | | *Relative improvement over ordinary least squares by quintile of house value* | | | | |
|---|---|---|---|---|---|---|---|
| *Method* | *Training sample* | *Hold-out sample* | 1st | 2nd | 3rd | 4th | 5th |
| Ordinary least squares | 47.3% | 41.7%<br>[39.7%, 43.7%] | – | – | – | – | – |
| Regression tree tuned by depth | 39.6% | 34.5%<br>[32.6%, 36.5%] | –11.5% | 10.8% | 6.4% | –14.6% | –31.8% |
| LASSO | 46.0% | 43.3%<br>[41.5%, 45.2%] | 1.3% | 11.9% | 13.1% | 10.1% | –1.9% |
| Random forest | 85.1% | 45.5%<br>[43.6%, 47.5%] | 3.5% | 23.6% | 27.0% | 17.8% | –0.5% |
| Ensemble | 80.4% | 45.9%<br>[44.0%, 47.9%] | 4.5% | 16.0% | 17.9% | 14.2% | 7.6% |

*Note:* The dependent variable is the log-dollar house value of owner-occupied units in the 2011 American Housing Survey from 150 covariates including unit characteristics and quality measures. All algorithms are fitted on the same, randomly drawn training sample of 10,000 units and evaluated on the 41,808 remaining held-out units. The numbers in brackets in the hold-out sample column are 95 percent bootstrap confidence intervals for hold-out prediction performance, and represent measurement variation for a fixed prediction function. For this illustration, we do not use sampling weights. Details are provided in the online Appendix at http://e-jep.org.

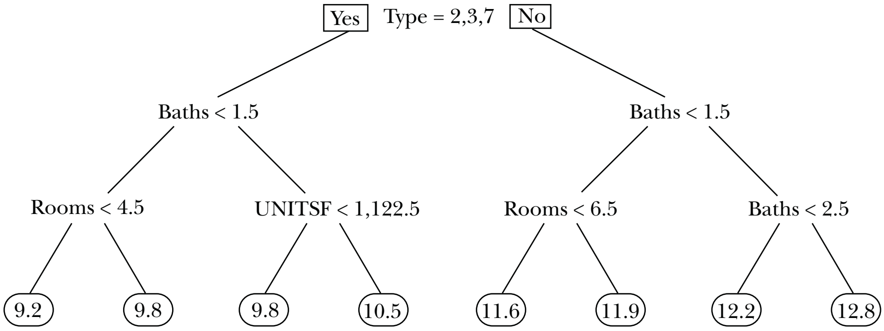
procedures work, we will focus in depth on a comparison of ordinary least squares and regression trees.

### From Linear Least-Squares to Regression Trees

Applying ordinary least squares to this problem requires making some choices. For the ordinary least squares regression reported in the first row of Table 1, we included all of the main effects (with categorical variables as dummies). But why not include interactions between variables? The effect of the number of bedrooms may well depend on the base area of the unit, and the added value of a fireplace may be different depending on the number of living rooms. Simply including all pairwise interactions would be infeasible as it produces more regressors than data points (especially considering that some variables are categorical). We would therefore need to hand-curate which interactions to include in the regression. An extreme version of this challenge appears in the face-recognition problem. The functions that effectively combine pixels to predict faces will be highly nonlinear and interactive: for example, "noses" are only defined by complex interactions between numerous pixels.

Machine learning searches for these interactions automatically. Consider, for example, a typical machine learning function class: regression trees. Like a linear function, a regression tree maps each vector of house characteristics to a predicted

*Figure 1*
**A Shallow Regression Tree Predicting House Values**



*Note:* Based on a sample from the 2011 American Housing Survey metropolitan survey. House-value predictions are in log dollars.

value. The prediction function takes the form of a tree that splits in two at every node. At each node of the tree, the value of a single variable (say, number of bathrooms) determines whether the left (less than two bathrooms) or the right (two or more) child node is considered next. When a terminal node—a leaf—is reached, a prediction is returned. An example of a tree is given in Figure 1. We could represent the tree in Figure 1 as a linear function, where each of the leaves corresponds to a product of dummy variables ($x_1 = 1_{TYPE=2,3,7} \times 1_{BATHS<1.5} \times 1_{ROOMS<4.5}$ for the leftmost leaf) with the corresponding coefficient ($\alpha_1 = 9.2$). Trees are thus a highly interactive function class.

### The Secret Sauce

How can a tree even be fitted here? A deep enough tree would fit perfectly—each observation would end up in its own leaf. That tree will have perfect *fit,* but of course this is really perfect *overfit:* out of sample, this tree would perform terribly for prediction. The (over)fitting conundrum is not specific to trees. The very appeal of machine learning is high dimensionality: flexible functional forms allow us to fit varied structures of the data. But this flexibility also gives so many possibilities that simply picking the function that fits best in-sample will be a terrible choice. So how does machine learning manage to do out-of-sample prediction?

The first part of the solution is *regularization.* In the tree case, instead of choosing the "best" overall tree, we could choose the best tree among those of a certain depth. The shallower the tree, the worse the in-sample fit: with many observations in each leaf, no one observation will be fit very well. But this also means there will be less overfit: the idiosyncratic noise of each observation is averaged out. Tree depth is an example of a regularizer. It measures the complexity of a function. As we regularize less, we do a better job at approximating the in-sample variation, but for the same reason, the wedge between in-sample and out-of-sample fit will typically increase. Machine learning algorithms typically have a regularizer associated with them. By choosing the level of regularization appropriately, we can have some benefits of flexible functional forms without having those benefits be overwhelmed by overfit.

How do we choose the optimal depth of the tree? In machine learning terms, how do we choose the level of regularization ("tune the algorithm")? This is the second key insight: *empirical tuning*. The essential problem of overfitting is that we would like the prediction function to do well *out of sample,* but we only fit in-sample. In empirical tuning, we create an out-of-sample experiment inside the original sample. We fit on one part of the data and ask which level of regularization leads to the best performance on the other part of the data.[4] We can increase the efficiency of this procedure through cross-validation: we randomly partition the sample into equally sized subsamples ("folds"). The estimation process then involves successively holding out one of the folds for evaluation while fitting the prediction function for a range of regularization parameters on all remaining folds. Finally, we pick the parameter with the best estimated average performance.[5] The second row of Table 1 shows the performance of a regression tree where we have chosen depth in this way.

This procedure works because prediction quality is observable: both predictions $\hat{y}$ and outcomes $y$ are observed. Contrast this with parameter estimation, where typically we must rely on assumptions about the data-generating process to ensure consistency. Observability by itself would not make prediction much easier since the algorithm would still need to sort through a very large function class. But regularization turns this choice into a low-dimensional one—we only need to choose the best tuning parameter. Regularization combines with the observability of prediction quality to allow us to fit flexible functional forms and still find generalizable structure.

### *Most of Machine Learning in One Expression*[6]

This structure—regularization and empirical choice of tuning parameters—helps organize the sometimes bewildering variety of prediction algorithms that one encounters. There is a function class $\mathcal{F}$ (in this case, trees) and a regularizer $R(f)$ (in the specific example, depth of tree) that expresses the complexity of a function

[4] One approach to the tuning problem is deriving the optimal level of regularization analytically for each procedure and under assumptions on the sampling process, such as AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), and SURE (Stein's Unbiased Risk Estimate). This theoretical guidance is helpful when available and applicable, but assumptions may prove hard to verify as the reason for undertaking nonparametric analysis may be that we are unsure about features of the data-generating processes in the first place. In other cases, theoretical results give only asymptotic guidance that remain an unverifiable promise in finite samples.

[5] In some cases, the researcher will adjust the empirical loss minimizer to account for measurement error and/or sample size differences in mapping observed performance to the level of regularization. An example is the "one standard-error rule" for LASSO tuning discussed in Hastie, Tibshirani, and Friedman (2009).

[6] We adapted the title of this section from a post on Francis X. Diebold's "No Hesitations" blog, http://fxdiebold.blogspot.com/2017/01/all-of-machine-learning-in-one.html.

*Table 2*
**Some Machine Learning Algorithms**

| *Function class $\mathcal{F}$ (and its parametrization)* | *Regularizer $R(f)$* |
|---|---|
| **Global/parametric predictors** | |
| Linear $\beta' x$ (and generalizations) | Subset selection $\|\|\beta\|\|_0 = \sum_{j=1}^{k} 1_{\beta_j \neq 0}$<br>LASSO $\|\|\beta\|\|_1 = \sum_{j=1}^{k} \|\beta_j\|$<br>Ridge $\|\|\beta\|\|_2^2 = \sum_{j=1}^{k} \beta_j^2$<br>Elastic net $\alpha \|\|\beta\|\|_1 + (1 - \alpha) \|\|\beta\|\|_2^2$ |
| **Local/nonparametric predictors** | |
| Decision/regression trees | Depth, number of nodes/leaves, minimal leaf size, information gain at splits |
| Random forest (linear combination of trees) | Number of trees, number of variables used in each tree, size of bootstrap sample, complexity of trees (see above) |
| Nearest neighbors | Number of neighbors |
| Kernel regression | Kernel bandwidth |
| **Mixed predictors** | |
| Deep learning, neural nets, convolutional neural networks | Number of levels, number of neurons per level, connectivity between neurons |
| Splines | Number of knots, order |
| **Combined predictors** | |
| Bagging: unweighted average of predictors from bootstrap draws | Number of draws, size of bootstrap samples (and individual regularization parameters) |
| Boosting: linear combination of predictions of residual | Learning rate, number of iterations (and individual regularization parameters) |
| Ensemble: weighted combination of different predictors | Ensemble weights (and individual regularization parameters) |

(more precisely the complexity of its representation).[7] Picking the prediction function then involves two steps: The first step is, conditional on a level of complexity, to pick the best in-sample loss-minimizing function.[8] The second step is to estimate the optimal level of complexity using empirical tuning (as we saw in cross-validating the depth of the tree). In Table 2, we give an incomplete overview of methods that follow this pattern.

[7] We write the regularizer as a mapping from the function itself. In cases where functions are not uniquely parametrized (and for practical purposes in many applications), we implicitly refer to a set of parameters that define a function for a given parametrization of the function class. Also, the complexity itself may be estimated from the training data.

[8] We summarize this central step in the expression

$$\text{minimize } \underbrace{\textstyle\sum_{i=1}^{n} L(f(x_i), y_i}_{\text{in-sample loss}}, \text{ over } \overbrace{f \in F}^{\text{function class}} \text{ subject to } \underbrace{R(f) \leq c}_{\text{complexity restriction}}.$$

For example, in our framework, the LASSO (probably the machine learning tool most familiar to economists) corresponds to 1) a quadratic loss function, 2) a class of linear functions (over some fixed set of possible variables), and 3) a regularizer which is the sum of absolute values of coefficients.[9] This effectively results in a linear regression in which only a small number of predictors from all possible variables are chosen to have nonzero values: the absolute-value regularizer encourages a coefficient vector where many are exactly zero. The third row of Table 1 shows the performance of LASSO in predicting house prices. Ridge regression is a close cousin: it simply uses a quadratic regularizer instead of the sum of absolute values.

In some of the most successful machine learning methods, multiple predictors from the same function class are combined into a single prediction function and tuned jointly. The fourth row in Table 1 shows the performance of a random forest; it outperforms ordinary least squares on the hold-out by over 9 percent in terms of overall $R^2$. The random forest is an average over many (in this case, 700) trees. Each tree is fitted on a bootstrap sample of the original training set and constrained to a randomly chosen subset of variables. The predictions of the trees are then averaged. The regularization variables in this algorithm include the complexity of each individual tree (such as its depth), the number of variables used in each tree, the size of each bootstrap sample, and the number of trees.

The last row in Table 1 lists an ensemble method that runs several separate algorithms (in this case tree, LASSO, and forest) and then averages their predictions, with weights chosen by cross-validation. The fact that the ensemble comes out on top in Table 1—with an out-of-sample $R^2$ of almost 46 percent—is no isolated case. While it may be unsurprising that such ensembles perform well *on average*—after all, they can cover a wider array of functional forms—it may be more surprising that they come on top in virtually *every* prediction competition.

Other models that we have not estimated in our data also fit this framework. For example, neural nets are popular prediction algorithms for image recognition tasks. For one standard implementation in binary prediction, the underlying function class is that of nested logistic regressions: The final prediction is a logistic transformation of a linear combination of variables ("neurons") that are themselves such logistic transformations, creating a layered hierarchy of logit regressions. The complexity of these functions is controlled by the number of layers, the number of neurons per layer, and their connectivity (that is, how many variables from one level enter each logistic regression on the next).

### Econometric Guidance

Viewed this way, there are several choices to be made when using a machine learning approach. First, this approach involves choosing the functions we fit and how we regularize them: Should I use a regression tree or linear functions? If I choose a tree, do I express its complexity by its depth, the minimal number of units

[9] For some readers, a more familiar equation for the LASSO is the Lagrangian dual formulation, where the Lagrange multiplier $\lambda$ plays the role of the tuning parameter.

in each leaf, or the minimal improvement in prediction quality at every split? Available guidance in the machine learning literature is largely based on a combination of simulation studies and expert intuition. They are complemented by recent theoretical results in econometrics that shed light on the comparative performance of different regularizers, such as Abadie and Kasy (2017) for LASSO and close relatives.

Practically, one must decide how to encode and transform the underlying variables. In our example of house prices, do we include base area *per room* as a variable, or just total area? Should we use logarithmic scales? Normalize to unit variances? These choices about how to represent the features will interact with the regularizer and function class: A linear model can reproduce the log base area per room from log base area and log room number easily, while a regression tree would require many splits to do so. In a traditional estimator, replacing one set of variables by a set of transformed variables from which it could be reconstructed would not change the predictions, because the set of functions being chosen from has not changed. But with regularization, including these variables can improve predictions because—at any given level of regularization—the set of functions might change. If the number of bathrooms *per bedroom* is what we suspect will matter in the price-setting process, creating that variable explicitly lowers the complexity cost for including this variable. Economic theory and content expertise play a crucial role in guiding where the algorithm looks for structure first. This is the sense in which "simply throw it all in" is an unreasonable way to understand or run these machine learning algorithms. For example, in visual tasks, some understanding of geometry proves crucial in specifying the way in which neurons are connected within neural nets.

A final set of choices revolves around the tuning procedure: Should out-of-sample performance be estimated using some known correction for overfitting (such as an adjusted $R^2$ when it is available) or using cross-validation? How many folds should be used in cross-validation, and how exactly should the final tuning parameter be chosen? While asymptotic results show that cross-validation tuning approximates the optimal complexity (Vaart, Dudoit, and Laan 2006), available finite-sample guidance on its implementation—such as heuristics for the number of folds (usually five to ten) or the "one standard-error rule" for tuning the LASSO (Hastie, Tibshirani, and Friedman 2009)—has a more ad-hoc flavor. Design choices must be made about function classes, regularizers, feature representations, and tuning procedures: there are no definitive and universal answers available. This leaves many opportunities for econometric research.

### Quantifying Predictive Performance

While these design choices leave plenty of freedom, having a reliable estimate of predictive performance is a nonnegotiable requirement for which strong econometric guarantees are available. In the house-price example, we divide the sample into a training and a test (hold-out) sample. This implements a *firewall principle*: none of the data involved in fitting the prediction function—which includes cross-validation to tune the algorithm—is used to evaluate the prediction function that is produced. As a result, inference on predictive performance of a fixed predictive function is a straightforward task of mean estimation: the distribution of realized loss in the hold-out (taking any clustering into account) directly yield consistent estimates of performance and confidence intervals.

Econometric theory plays a dual role here. First, econometrics can guide design choices, such as the number of folds or the function class. Guidance in these choices can help improve prediction quality and the power of any test based on it. Second, given the fitted prediction function, it must enable us to make inferences about estimated fit. The hold-out sample exactly allows us to form properly sized tests about predictive properties of the fitted function.

**What Do We (Not) Learn from Machine Learning Output?**

It is tempting to do more with the fitted function. Why not also use it to learn something about the "underlying model": specifically, why not use it to make inferences about the underlying data-generating process? Even if correlations are not causal, might they not reveal useful underlying structure? The LASSO regression of Table 1 ends up not using the number of dining rooms as a right-hand variable. Does that reveal that the number of dining rooms is an unimportant variable in determining house prices (given the other available variables)? It is tempting to draw such conclusions, and such conclusions could be economically meaningful: for example, in predicting wages, the weight placed on race by a machine learning algorithm seems like it could be a proxy for discrimination. Statistical packages contribute to these inferences by outputting measures of variable importance in the fitted functions.

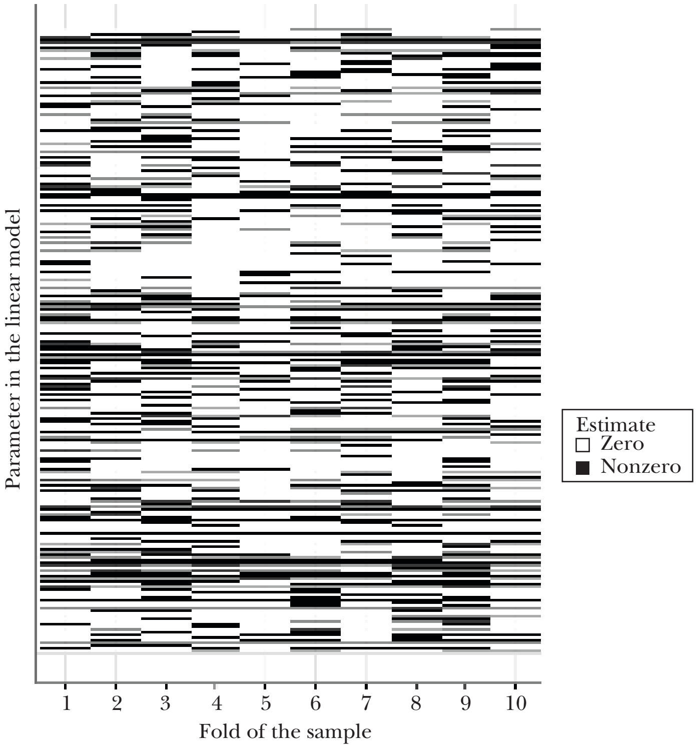
One obvious problem that arises in making such inferences is the lack of standard errors on the coefficients. Even when machine-learning predictors produce familiar output like linear functions, forming these standard errors can be more complicated than seems at first glance as they would have to account for the model selection itself. In fact, Leeb and Pötscher (2006, 2008) develop conditions under which it is impossible to obtain (uniformly) consistent estimates of the distribution of model parameters after data-driven selection.

But there is an even bigger challenge. To illustrate the problem, we repeated the house-value prediction exercise on subsets of our sample from the American Housing Survey. First, we randomly cut the sample into ten partitions of approximately 5,000 units each. On each partition, we re-estimate the LASSO predictor. Through its regularizer, LASSO produces a sparse prediction function, so that many coefficients are zero and are "not used"—in this example, we find that more than half the variables are unused in each run.

Figure 2 shows how the variables that are used vary from partition to partition. Each row represents one of *x* variables used. Each column represents a different partition. We color each cell black if that variable is used by the LASSO model in that partition. Figure 2 documents a fundamental problem: a variable used in one partition may be unused in another. In fact, there are few stable patterns overall.

These instabilities do not reflect instability in prediction quality—in fact, the $R^2$ remains roughly constant from partition to partition. The problem arises because if

*Figure 2*
**Selected Coefficients (Nonzero Estimates) across Ten LASSO Regressions**



*Note:* We repeated the house-value prediction exercise on subsets of our sample from the American Housing Survey. First, we randomly cut the sample into ten partitions of approximately 5,000 units each. On each partition, we re-estimate the LASSO predictor, with LASSO regularization parameter fixed. The figure shows how the variables that are used vary from partition to partition. Each row represents one of *x* variables used. Each column represents a different partition. We color each cell black if that variable is used by the LASSO model (has a nonzero coefficient) in that partition. The figure documents a fundamental problem: a variable used in one partition may be unused in another. In fact, there are few stable patterns overall. For details, see discussion in text and online appendix available with this paper at http://e-jep.org.

the variables are correlated with each other (say the number of rooms of a house and its square-footage), then such variables are substitutes in predicting house prices. Similar predictions can be produced using very different variables. Which variables are actually chosen depends on the specific finite sample. In traditional estimation, correlations between observed variables would be reflected in large standard errors that express our uncertainty in attributing effects to one variable over the other.

This problem is ubiquitous in machine learning. The very appeal of these algorithms is that they can fit many different functions. But this creates an Achilles' heel: more functions mean a greater chance that two functions with very different coefficients can produce similar prediction quality. As a result, how an algorithm chooses between two very different functions can effectively come down to the flip of a coin. In econometric terms, while the lack of standard errors illustrates the limitations to making inference *after* model selection, the challenge here is (uniform) model selection consistency itself.

Regularization also contributes to the problem. First, it encourages the choice of less complex, but wrong models. Even if the best model uses interactions of number of bathrooms with number of rooms, regularization may lead to a choice of a simpler (but worse) model that uses only number of fireplaces. Second, it can bring with it a cousin of omitted variable bias, where we are typically concerned with correlations between observed variables and unobserved ones. Here, when regularization excludes some variables, even a correlation between observed variables and other *observed* (but excluded) ones can create bias in the estimated coefficients.

### Recovering Structure: Estimation ($\hat{\beta}$) vs Prediction ($\hat{y}$)

We face a challenge. On the one hand, these machine learning algorithms by their very construction—tuning and evaluation out of sample—seek a generalizable structure and are evaluated on their capacity to find it. These algorithms do detect structure in $\hat{y}$: when predictive quality is high, some structure must have been found. Some econometric results also show the converse: when there is structure, it will be recovered at least asymptotically (for example, for prediction consistency of LASSO-type estimators in an approximately sparse linear framework, see Belloni, Chernozhukov, and Hansen 2011). On the other hand, we have seen the dangers of naively interpreting the estimated $\hat{\beta}$ parameters as indicating the discovered structure.

Of course, assumptions about the data-generating process would allow us to take the estimated $\hat{\beta}$ parameters more literally. The discussion above suggests that we must limit the correlations between the observed variables. This is seen clearly in Zhao and Yu (2006) who establish asymptotic model-selection consistency for the LASSO. Besides assuming that the true model is "sparse"—only a few variables are relevant—they also require the "irrepresentable condition" between observables: loosely put, none of the irrelevant covariates can be even moderately related to the set of relevant ones.

In practice, these assumptions are strong. The instability in Figure 2, for example, suggests that they are not realistic in the house price example. But since we know this model is finding some structure, can we characterize it? A key area of future research in econometrics and machine learning is to make sense of the estimated prediction function without making strong assumptions about the underlying true world.

## How Machine Learning Can Be Applied

Our starting point for applications of machine learning algorithms is guided by both the strength of machine learning—it provides a powerful, flexible way of making quality predictions—and its weakness: absent strong and mostly unverifiable assumptions, machine learning does not produce stable estimates of the underlying parameters. Therefore, we look for $\hat{y}$ problems, places where improved prediction has large applied value.

**New Data**

The phrase "big data" emphasizes a change in the scale of data. But there has been an equally important change in the *nature* of this data. Machine learning can deal with unconventional data that is too high-dimensional for standard estimation methods, including image and language information that we conventionally had not even thought of as data we can work with, let alone include in a regression.

Satellites have been taking images of the earth for decades, which we can now use not just as pixelated vectors, but as economically meaningful input. Donaldson and Storeygard (in this journal, 2016) provide an overview of the growing literature in economics using satellite data, including how luminosity at night correlates with economic output (Henderson, Storeygard, and Weil 2012) or estimating future harvest size (Lobell 2013). Satellite images do not directly contain, for example, measures of crop yield. Instead, they provide us with a large $x$ vector of image-based data; these images are then matched (in what we hope is a representative sample) to yield data which form the $y$ variable. This translation of satellite images to yield measures is a prediction problem. Machine learning is the essential tool by which we extract and scale economically meaningful signals from this data.

These new sources of data are particularly relevant where reliable data on economic outcomes are missing, such as in tracking and targeting poverty in developing countries (Blumenstock 2016). Jean et al. (2016) train a neural net to predict local economic outcomes from satellite data in five African countries. Machine learning also yields economic predictions from large-scale network data; for example, Blumenstock, Cadamuro, and On (2015) use cell-phone data to measure wealth, allowing them to quantify poverty in Rwanda at the individual level. Image recognition can of course be used beyond satellite data, and localized prediction of economic outcomes is relevant beyond the developing world: as one example, Glaeser, Kominers, Luca, and Naik (2016) use images from Google Street View to measure block-level income in New York City and Boston.

Language provides another new powerful source of data. As with satellite images, online posts can be made meaningful by labeling them with machine learning. Kang, Kuznetsova, Luca, and Choi (2013) use restaurant reviews on Yelp.com to predict the outcome of hygiene inspections. Antweiler and Frank (2004) classify text on online financial message boards as bullish, bearish, or neither. Their algorithm trains on a small number of manual classifications, and scales these labels up to 1.5 million messages as a basis for the subsequent analysis, which shows that online messages help explain market volatility, with statistically significant, if economically modest, effects on stock returns.

Financial economists rely heavily on corporate financial information, such as that available in Compustat. But companies release detailed reports on their financial positions above and beyond these numbers. In the United States, publicly traded companies must file annual 10-K forms. Kogan, Levin, Routledge, Sagi, and Smith (2009) predict volatility of roughly 10,000 such firms from market-risk disclosure text within these forms, and show that it adds significant predictive information to past volatility. Hoberg and Phillips (2016) extract similarity of firms from their 10-K business description texts, generating new time-varying industry classifications for these firms.

Machine learning can be useful in preprocessing and imputing even in traditional datasets. In this vein, Feigenbaum (2015a, b) applies machine-learning classifiers to match individuals in historical records: he links fathers and sons across censuses and other data sources, which allows him to quantify social mobility during the Great Depression. Bernheim, Bjorkegren, Naecker, and Rangel (2013) link survey responses to observable behavior: A subset of survey respondents take part in a laboratory experiment; a machine learning algorithm trained on this data predicts actual choices from survey responses, giving economists a tool to infer actual from reported behavior.

**Prediction in the Service of Estimation**

A second category of application is in tasks that we approach as estimation problems. Even when we are interested in a parameter $\hat{\beta}$, the tool we use to recover that parameter may contain (often implicitly) a prediction component. Take the case of linear instrumental variables understood as a two-stage procedure: first regress $x = \gamma' z + \delta$ on the instrument $z$, then regress $y = \beta' x + \epsilon$ on the fitted values $\hat{x}$. The first stage is typically handled as an estimation step. But this is effectively a prediction task: only the predictions $\hat{x}$ enter the second stage; the coefficients in the first stage are merely a means to these fitted values.

Understood this way, the finite-sample biases in instrumental variables are a consequence of overfitting. Overfitting means that the in-sample fitted values $\hat{x}$ pick up not only the signal $\gamma' z$, but also the noise $\delta$. As a consequence, $\hat{x}$ is biased towards $x$, and the second-stage instrumental variable estimate $\hat{\beta}$ is thus biased towards the ordinary least squares estimate of $y$ on $x$. Since overfit will be larger when sample size is low, the number of instruments is high, or the instruments are weak, we can see why biases arise in these cases (Bound, Jaeger, and Baker 1995; Bekker 1994; Staiger and Stock 1997).

This analogy carries through to some of the classical solutions to finite-sample bias. Above, we used hold-out sets (in evaluating the prediction function) or cross-validation (in choosing the tuning parameter) to separate the data used in the fitting of the function from the data used in the forming of predicted values; this ensured, for example, that our evaluations of a function's prediction quality were unbiased. These same techniques applied here result in split-sample instrumental variables (Angrist and Krueger 1995) and "jackknife" instrumental variables (Angrist, Imbens, and Krueger 1999). Overfitting has wider consequences: the flipside of excessive in-sample overfitting is bad out-of-sample prediction. In fact, predicting well requires managing overfitting, which was the goal of both regularization and empirical tuning. These techniques are applicable to the first stage of instrumental variable analysis as well. In particular, a set of papers has already introduced regularization into the first stage in a high-dimensional setting, including the LASSO (Belloni, Chen, Chernozhukov, and Hansen 2012) and ridge regression (Carrasco 2012; Hansen and Kozbur 2014). More recent extensions include nonlinear functional forms, all the way to neural nets (Hartford, Leyton-Brown, and Taddy 2016).

Practically, even when there appears to be only a few instruments, the problem is effectively high-dimensional because there are many degrees of freedom in how instruments are actually constructed. For example, several papers use college proximity as an instrument in estimating returns to schooling (for example, Card 1999, Table 4). How exactly college proximity is used, however, varies. After all, it can be included linearly, logarithmically, or as dummies (and if so, with different thresholds) and can be interacted with other variables (such as demographic groups most likely to be affected). The latitude in making these choices makes it even more valuable to consider the first stage as a prediction problem. It allows us to let the data explicitly pick effective specifications, and thus allows us to recover more of the variation and construct stronger instruments, provided that predictions are constructed and used in a way that preserves the exclusion restriction.[10]

Many other inference tasks also have a prediction problem implicit inside them. In controlling for observed confounders, we do not care about the parameters associated with the control variables as an end in themselves. For example, Lee, Lessler, and Stuart (2010) use machine-learning algorithms to estimate the propensity score. Chernozhukov, Chetverikov, Demirer, Duflo, Hansen, and Newey (2016) take care of high-dimensional controls in treatment effect estimation by solving two simultaneous prediction problems, one in the outcome and one in the treatment equation.

Similar opportunities arise even in cases where we have experimental data. Consider the problem of verifying balance between treatment and control groups (such as when there is attrition). Or consider the seemingly different problem of testing for effects on many outcomes. Both can be viewed as prediction problems (Ludwig, Mullainathan, and Spiess 2017). If treatment assignment can be predicted better than chance from pretreatment covariates, this is a sign of imbalance. If treatment assignment can be predicted from a set of outcomes, the treatment must have had an effect. Estimating heterogeneous treatment effects can also be viewed as a prediction problem, though the parallel is nonobvious and implementing the transformation is a major contribution of the papers in this literature. Typically, heterogeneous treatment effects might be estimated as coefficients on interaction terms in a linear regression. Consider instead the prediction task of mapping unit-level attributes to individual effect estimates. Of course, individual-level treatment effects are not directly observed. Despite this, machine-learning methods have been successfully applied to map out treatment effect heterogeneity. Athey and Imbens (2016) use sample-splitting to obtain valid (conditional) inference on

[10] In particular, we have to avoid "forbidden regressions" (Angrist and Pischke 2008) in which correlation between first-stage residuals and fitted values exists and creates bias in the second stage.

treatment effects that are estimated using decision trees, as previously suggested by Imai and Strauss (2011). Wager and Athey (2015) extend the construction to random forests, while Grimmer, Messing, and Westwood (2016) employ ensemble methods. These heterogenous treatment effects can be used to assign treatments; Misra and Dubé (2016) illustrate this on the problem of price targeting, applying Bayesian regularized methods to a large-scale experiment where prices were randomly assigned.

Expressing parts of these inference tasks as prediction problems also makes clear the limitation on their output. A carefully constructed heterogeneity tree provides valid estimates of treatment effects in every leaf (Athey and Imbens 2016). At the same time, one must be careful in interpreting the particular representation of the chosen tree, as it may suffer from instabilities similar to those in the American Housing Survey data above. Suppose the algorithm chooses a tree that splits on education but not on age. Conditional on this tree, the estimated coefficients are consistent. But that does not imply that treatment effects do not also vary by age, as education may well covary with age; on other draws of the data, in fact, the same procedure could have chosen a tree that split on age instead.

### Prediction in Policy

Consider the following policy problem: Shortly after arrest, a judge must decide where defendants will wait for trial, at home or in jail. This decision, by law, must be based on a prediction by the judge: If released, will the defendant return for their court appearance or will they skip court, and will they potentially commit further crimes? Statistical tools have helped improve policies in several ways (such as randomized control trials helping to answer "does the policy work?"). In this case, one might wonder whether a predictive algorithm could similarly help improve the judge's decision (Kleinberg et al. 2017).

Prediction policy problems, such as the bail problem, appear in many areas (Kleinberg, Ludwig, Mullainathan, and Obermeyer 2015). For instance, a large literature estimates the impact of hiring an additional teacher—this is meant to inform the decision of whether to hire more teachers. The decision of *which* teacher to hire, however, requires a prediction: the use of information available at time of hiring to forecast individual teacher quality (Kane and Staiger 2008; Dobbie 2011; Jacob et al. 2016). Chalfin et al. (2016) provide some preliminary evidence of how machine learning may improve predictive accuracy in these and other personnel decisions. Chandler, Levitt, and List (2011) predict highest-risk youth so that mentoring interventions can be appropriately targeted. Abelson, Varshney, and Sun (2014), McBride and Nichols (2016), and Engstrom, Hersh, and Newhouse (2016) use machine learning to improve poverty targeting relative to existing poverty scorecards. These predictive problems intimately relate to questions we already seek to answer: the impact of an extra teacher depends on how that teacher is chosen; the impact of a transfer program depends on how well targeted it is. Given the active research contributing to these policy discussions, expanding the focus to these adjacent prediction questions seems promising.

Economists can play a crucial role in solving prediction policy problems. First, even though prediction is important, machine learning is not enough: familiar econometric challenges arise. In deciding whether an algorithm could improve on the judge, one must resolve a basic counterfactual issue: we only know the crimes committed by those released. Many predictions problems share the feature that the available data is dictated by existing decision rules, and insights from the causal inference could prove helpful in tackling these problems; for example, Kleinberg et al. (2017) use pseudo-random assignment to judges of differing leniency in their application. Second, behavioral issues arise. Even when an algorithm can help, we must understand the factors that determine adoption of these tools (Dawes, Faust, and Meehl 1989; Dietvorst, Simmons, and Massey 2015; Yeomans, Shah, Mullainathan, and Kleinberg 2016). What factors determine faith in the algorithm? Would a simpler algorithm be more believed? How do we encourage judges to use their private information optimally? These questions combine problems of technology diffusion, information economics, and behavioral economics.

### Testing Theories

A final application of supervised machine learning is to test directly theories that are inherently about predictability. Within the efficient markets theory in finance, for example, the inability to make predictions about the future is a key prediction. Moritz and Zimmermann (2016) adapt methods from machine learning to show that past returns of US firms do have significant predictive power over their future stock prices.

Machine learning can also be used to construct a benchmark for how well theories are performing. A common concern is that even if a theory is accurate, it may only clear up a little of the systematic variation it aims to explain. The $R^2$ alone does not address this question, as some of the total variance may not be explainable from what is measured. Kleinberg, Liang, and Mullainathan (2015) propose to compare the predictive power of a theory to that of an optimal predictor. Relatedly, Peysakhovich and Naecker (2015) compare the out-of-sample performance of behavioral economics models for choices under risk and ambiguity to an atheoretical machine-learning benchmark.

## Conclusion

The problem of artificial intelligence has vexed researchers for decades. Even simple tasks such as digit recognition—challenges that we as humans overcome so effortlessly—proved extremely difficult to program. Introspection into how our mind solves these problems failed to translate into procedures. The real breakthrough came once we stopped trying to deduce these rules. Instead, the problem was turned into an inductive one: rather than hand-curating the rules, we simply let the data tell us which rules work best.

For empiricists, these theory- and data-driven modes of analysis have always coexisted. Many estimation approaches have been (often by necessity) based on top-down, theory-driven, deductive reasoning. At the same time, other approaches have aimed to simply let the data speak. Machine learning provides a powerful tool to hear, more clearly than ever, what the data have to say.

These approaches need not be in conflict. A structural model of poverty, for example, could be applied to satellite image data processed using machine learning. Theory could guide what variables to manipulate in an experiment; but in analyzing the results, machine learning could help manage multiple outcomes and estimate heterogeneous treatment effects.

In the long run, new empirical tools have also served to expand the kinds of problems we work on. The increased use of randomized control trials has also changed the kinds of questions empirical researchers work on. Ultimately, machine learning tools may also increase the scope of our work—not just by delivering new data or new methods but by focusing us on new questions.

■ *We are grateful to the editors Gordon Hanson, Enrico Moretti, and Timothy Taylor for their feedback and to Alberto Abadie, Peter Ganong, Lawrence Katz, Supreet Kaur, Jens Ludwig, and Andrei Shleifer for thoughtful and detailed comments.*

## References

**Abadie, Alberto, and Maximilian Kasy.** 2017. *The Risk of Machine Learning.* https://ideas.repec.org/p/qsh/wpaper/383316.html.

**Abelson, Brian, Kush R. Varshney, and Joy Sun.** 2014. "Targeting Direct Cash Transfers to the Extremely Poor." In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1563–72. ACM.

**Angrist, Joshua D., Guido W. Imbens, and Alan B. Krueger.** 1999. "Jackknife Instrumental Variables Estimation." *Journal of Applied Econometrics* 14(1): 57–67

**Angrist, Joshua D., and Alan B. Krueger.** 1995. "Split-Sample Instrumental Variables Estimates of the Return to Schooling." *Journal of Business and Economic Statistics* 13(2): 225–35.

**Angrist, Joshua D., and Jörn-Steffen S. Pischke.** 2008. *Mostly Harmless Econometrics: An Empiricist's Companion.* Princeton University Press.

**Antweiler, Werner, and Murray Z. Frank.** 2004. "Is All That Talk Just Noise? The Information Content of Internet Stock Message Boards." *Journal of Finance* 59(3): 1259–94.

**Athey, Susan.** 2015. "Machine Learning and Causal Inference for Policy Evaluation." In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 5–6. ACM.

**Athey, Susan, and Guido Imbens.** 2016. "Recursive Partitioning for Heterogeneous Causal Effects. *PNAS* 113(27): 7353–60.

**Bekker, Paul A.** 1994. "Alternative Approximations to the Distributions of Instrumental Variable Estimators." *Econometrica* 62(3): 657–81.

**Belloni, Alexandre, Daniel Chen, Victor Chernozhukov, and Christian Hansen.** 2012. "Sparse Models and Methods for Optimal Instruments with an Application to Eminent Domain." *Econometrica* 80(6): 2369–2429.

**Belloni, Alexandre, Victor Chernozhukov, and Christian Hansen.** 2011. "Inference for High-Dimensional Sparse Econometric Models." arXiv:1201.0220.

**Belloni, Alexandre, Victor Chernozhukov,**

**and Christian Hansen.** 2014. "Inference on Treatment Effects after Selection among High-Dimensional Controls." *Review of Economic Studies* 81(2): 608–650.

**Bernheim, Douglas, Daniel Bjorkegren, Jeffrey Naecker, and Anatonio Rangel.** 2013. "Non-Choice Evaluations Predict Behavioral Responses to Changes in Economic Conditions." NBER Working Paper 19269.

**Blumenstock, Joshua E., Gabriel Cadamuro, and Robert On.** 2015. "Predicting Poverty and Wealth from Mobile Phone Metadata." *Science* 350(6264): 1073–76.

**Blumenstock, Joshua Evan.** 2016. "Fighting Poverty with Data." *Science* 353(6301): 753–54.

**Bound, John, David A. Jaeger, and Regina M. Baker.** 1995. "Problems with Instrumental Variables Estimation When the Correlation between the Instruments and the Endogenous Explanatory Variable is Weak." *Journal of the American Statistical Association* 90(430): 443–50.

**Card, David.** 1999. "The Causal Effect of Education on Earnings." *Handbook of Labor Economics*, vol. 3A, edited by Orley C. Ashenfelter and David Card, 1801–1863. North-Holland, Elsevier.

**Carrasco, Marine** 2012. "A Regularization Approach to the Many Instruments Problem." *Journal of Econometrics* 170(2): 383–98.

**Chalfin, Aaron, Oren Danieli, Andrew Hillis, Zubin Jelveh, Michael Luca, Jens Ludwig, and Sendhil Mullainathan.** 2016. "Productivity and Selection of Human Capital with Machine Learning." *American Economic Review* 106(5): 124–27.

**Chandler, Dana, Steven D. Levitt, and John A. List.** 2011. "Predicting and Preventing Shootings among At-Risk Youth." *American Economic Review* 101(3): 288–92.

**Chernozhukov, Victor, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, and Whitney Newey.** 2016. "Double Machine Learning for Treatment and Causal Parameters." arXiv:1608.00060.

**Dawes, Robyn M., David Faust, and Paul E. Meehl.** 1989. "Clinical versus Actuarial Judgment." *Science* 243(4899): 1668–74.

**Dietvorst, Berkeley J., Joseph P Simmons, and Cade Massey.** 2015. "Algorithm Aversion: People Erroneously Avoid Algorithms after Seeing Them Err." *Journal of Experimental Psychology: General* 144(1): 114–26.

**Dobbie, Will.** 2011. "Teacher Characteristics and Student Achievement: Evidence from Teach For America." https://www.princeton.edu/~wdobbie/files/dobbie_tfa_2011.pdf.

**Donaldson, Dave, and Adam Storeygard.** 2016. "The View from Above: Applications of Satellite Data in Economics." *Journal of Economic Perspectives* 30(4): 171–98.

**Engstrom, Ryan, Jonathan Hersh, and David Newhouse.** 2016. "Poverty from Space: Using High Resolution Satellite Imagery for Estimating Economic Well-being and Geographic Targeting." Unpublished paper.

**Einav, Liran, and Jonathan Levin.** 2014. "Economics in the Age of Big Data." *Science* 346(6210): 1243089.

**Feigenbaum, James J.** 2015a. "Automated Census Record Linking." http://scholar.harvard.edu/jfeigenbaum/publications/automated-census-record-linking.

**Feigenbaum, James J.** 2015b. "Intergenerational Mobility during the Great Depression." http://scholar.harvard.edu/jfeigenbaum/publications/jmp.

**Glaeser, Edward L., Scott Duke Kominers, Michael Luca, and Nakhil Naik.** 2016. "Big Data and Big Cities: The Promises and Limitations of Improved Measures of Urban Life." *Economic Inquiry*, Early View Article, online July 12.

**Grimmer, Justin, Solomon Messing, and Sean J. Westwood.** 2016. "Estimating Heterogeneous Treatment Effects and the Effects of Heterogeneous Treatments with Ensemble Methods." https://stanford.edu/~jgrimmer/het.pdf.

**Györfi, L., M. Kohler, A. Krzyzak, and H. Walk.** 2002. *A Distribution-Free Theory of Nonparametric Regression*. Springer.

**Hansen, Bruce E.** 2014. "Nonparametric Sieve Regression: Least Squares, Averaging Least Squares, and Cross-Validation." In *The Oxford Handbook of Applied Nonparametric and Semiparametric Econometrics and Statistics,* edited by Jeffrey S. Racine, Liangjun Su, and Aman Ullah. Oxford Univerity Press.

**Hansen, Christian, and Damian Kozbur.** 2014. "Instrumental Variables Estimation with Many Weak instruments using Regularized JIVE." *Journal of Econometrics* 182(2): 290–308.

**Hartford, Jason, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy.** 2016. "Counterfactual Prediction with Deep Instrumental Variables Networks." arXiv:1612.09596.

**Hastie, Trevor, Robert Tibshirani, and Jerome Friedman.** 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Second edition. New York, NY: Springer.

**Henderson, J. Vernon, Adam Storeygard, and David N. Weil.** 2012. "Measuring Economic Growth from Outer Space." *American Economic Review* 102(2): 994–1028.

**Hoberg, Gerard, and Gordon Phillips.** 2016. "Text-based Network Industries and Endogenous Product Differentiation. *Journal of Political Economy*

124(5): 1423–65.

**Imai, Kosuke, and Aaron Strauss.** 2011. "Estimation of Heterogeneous Treatment Effects from Randomized Experiments, with Application to the Optimal Planning of the Get-Out-the-Vote Campaign." *Political Analysis* 19(1): 1–19.

**Jacob, Bryan, Jonah E. Rockoff, Eric S. Taylor, Benjamin Lindy, and Rachel Rosen.** 2016. "Teacher Applicant Hiring and Teacher Performance: Evidence from DC Public Schools." NBER Working Paper 22054.

**Jean, Neal, Marshall Burke, Michael Xie, W. Matthew Davis, David B. Lobell, and Stefano Ermon.** 2016. "Combining Satellite Imagery and Machine Learning to Predict Poverty." *Science* 353(6301): 790–94.

**Kane, Thomas J., and Douglas O. Staiger.** 2008. "Estimating Teacher Impacts on Student Achievement: An Experimental Evaluation." NBER Working Paper 14607.

**Kang, Jun Seok, Polina Kuznetsova, Michael Luca, and Yejin Choi.** 2013. "Where *Not* to Eat? Improving Public Policy by Predicting Hygiene Inspections Using Online Reviews." EMNLP 2013: 2013 Conference on Empirical Methods in Natural Language.

**Kleinberg, Jon, Annie Liang, and Sendhil Mullainathan.** 2015. "The Theory is Predictive, But is It Complete? An Application to Human Perception of Randomness." Unpublished paper.

**Kleinberg, Jon, Jens Ludwig, Sendhil Mullainathan, and Ziad Obermeyer.** 2015. "Prediction Policy Problems." *American Economic Review* 105(5): 491–95.

**Kleinberg, Jon, Himabindu Lakkaraju, Jure Leskovec, Jens Ludwig, and Sendhil Mullainathan.** 2017. "Human Decisions and Machine Predictions." NBER Working Paper 23180.

**Kogan, Shimon, Dimitry Levin, Byran R. Routledge, Jacob S. Sagi, and Noah A. Smith.** 2009. "Predicting Risk from Financial Reports with Regression." In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics,* pp. 272–80. ACM.

**Lee, Brian K., Justin Lessler, and Elizabeth A. Stuart.** 2010. "Improving Propensity Score Weighting using Machine Learning." *Statistics in Medicine* 29(3): 337–46.

**Leeb, Hannes, and Benedikt M. Pötscher.** 2006. "Can One Estimate the Conditional Distribution of Post-Model-Selection Estimators?" *Annals of Statistics* 34(5): 2554–91.

**Leeb, Hannes, and Bendikt M. Pötscher.** 2008. "Can One Estimate the Unconditional Distribution of Post-Model-Selection Estimators?" *Econometric Theory* 24(2): 338–76.

**Lobell, David B.** 2013. "The Use of Satellite Data for Crop Yield Gap Analysis." *Field Crops Research* 143: 56–64.

**Ludwig, Jens, Sendhil Mullainathan, and Jann Spiess.** 2017. "Machine Learning Tests for Effects on Multiple Outcomes." Unpublished paper.

**McBride, Linden, and Austin Nichols.** 2016. "Retooling Poverty Targeting Using Out-of-Sample Validation and Machine Learning." *World Bank Economic Review,* lhw056.

**Misra, Sanjog, and Jean-Pierre Dubé.** 2016. "Scalable Price Targeting." Unpublished paper.

**Moritz, Benjamin, and Tom Zimmermann.** 2016. "Tree-Based Conditional Portfolio Sorts: The Relation between Past and Future Stock Returns." Available at SSRN: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2740751.

**Peysakhovich, Alexander, and Jeffrey Naecker.** 2015. "Using Methods from Machine Learning to Evaluate Models of Human Choice."

**Staiger, Douglas, and James H. Stock.** 1997. "Instrumental Variables Regression with Weak Instruments." *Econometrica* 65(3): 557–86.

**van der Vaart, Aad W., Sandrine Dudoit, and Mark J. van der Laan.** 2006. "Oracle Inequalities for Multi-fold Cross Validation." *Statistics and Decisions* 24(3): 351–71.

**Varian, Hal R.** 2014. "Big Data: New Tricks for Econometrics." *Journal of Economic Perspectives* 28(2): 3–28.

**Wager, Stefan, and Susan Athey.** 2015. "Estimation and Inference of Heterogeneous Treatment Effects using Random Forests." arXiv:1510.04342

**Yeomans, Michael, Anuj K. Shah, Sendhil Mullainathan, and Jon Kleinberg.** 2016. "Making Sense of Recommendations." Unpublished paper.

**Zhao, Peng, and Bin Yu.** 2006. "On Model Selection Consistency of Lasso." *Journal of Machine Learning Research* 7: 2541–63.