Journal of Economic Perspectives—Volume 28, Number 2—Spring 2014—Pages 3–28

Big Data: New Tricks for Econometrics[†]

Hal R. Varian

Computers are now involved in many economic transactions and can capture data associated with these transactions, which can then be manipulated and analyzed. Conventional statistical and econometric techniques such as regression often work well, but there are issues unique to big datasets that may require different tools.

First, the sheer size of the data involved may require more powerful data manipulation tools. Second, we may have more potential predictors than appropriate for estimation, so we need to do some kind of variable selection. Third, large datasets may allow for more flexible relationships than simple linear models. Machine learning techniques such as decision trees, support vector machines, neural nets, deep learning, and so on may allow for more effective ways to model complex relationships.

In this essay, I will describe a few of these tools for manipulating and analyzing big data. I believe that these methods have a lot to offer and should be more widely known and used by economists. In fact, my standard advice to graduate students these days is go to the computer science department and take a class in machine learning. There have been very fruitful collaborations between computer scientists and statisticians in the last decade or so, and I expect collaborations between computer scientists and econometricians will also be productive in the future.

■ *Hal Varian is Chief Economist, Google Inc., Mountain View, California, and Emeritus Professor of Economics, University of California, Berkeley, California. His email address is hal@ischool.berkeley.edu.*

[†] To access the Appendix and disclosure statements, visit
http://dx.doi.org/10.1257/jep.28.2.3

doi=10.1257/jep.28.2.3

Tools to Manipulate Big Data

Economists have historically dealt with data that fits in a spreadsheet, but that is changing as new more-detailed data becomes available (see Einav and Levin 2013, for several examples and discussion). If you have more than a million or so rows in a spreadsheet, you probably want to store it in a relational database, such as MySQL. Relational databases offer a flexible way to store, manipulate, and retrieve data using a Structured Query Language (SQL), which is easy to learn and very useful for dealing with medium-sized datasets.

However, if you have several gigabytes of data or several million observations, standard relational databases become unwieldy. Databases to manage data of this size are generically known as "NoSQL" databases. The term is used rather loosely, but is sometimes interpreted as meaning "not only SQL." NoSQL databases are more primitive than SQL databases in terms of data manipulation capabilities but can handle larger amounts of data.

Due to the rise of computer-mediated transactions, many companies have found it necessary to develop systems to process billions of transactions per day. For example, according to Sullivan (2012), Google has seen 30 trillion URLs, crawls over 20 billion of those a day, and answers 100 billion search queries a month. Analyzing even one day's worth of data of this size is virtually impossible with conventional databases. The challenge of dealing with datasets of this size led to the development of several tools to manage and analyze big data.

A number of these tools are proprietary to Google, but have been described in academic publications in sufficient detail that open-source implementations have been developed. Table 1 contains both the Google name and the name of related open-source tools. Further details can be found in the Wikipedia entries associated with the tool names.

Though these tools can be run on a single computer for learning purposes, real applications use large clusters of computers such as those provided by Amazon, Google, Microsoft, and other cloud-computing providers. The ability to rent rather than buy data storage and processing has turned what was previously a fixed cost of computing into a variable cost and has lowered the barriers to entry for working with big data.

Tools to Analyze Data

The outcome of the big-data processing described above is often a "small" table of data that may be directly human readable or can be loaded into an SQL database, a statistics package, or a spreadsheet. If the extracted data is still inconveniently large, it is often possible to select a subsample for statistical analysis. At Google, for example, I have found that random samples on the order of 0.1 percent work fine for analysis of business data.

Once a dataset has been extracted, it is often necessary to do some exploratory data analysis along with consistency and data-cleaning tasks. This is something

Table 1
Tools for Manipulating Big Data

Google name	*Analog*	*Description*
Google File System	Hadoop File System	This system supports files so large that they must be distributed across hundreds or even thousands of computers.
Bigtable	Cassandra	This is a table of data that lives in the Google File System. It too can stretch over many computers.
MapReduce	Hadoop	This is a system for accessing and manipulating data in large data structures such as Bigtables. MapReduce allows you to access the data in parallel, using hundreds or thousands of machines to extract the data you are interested in. The query is "mapped" to the machines and is then applied in parallel to different shards of the data. The partial calculations are then combined ("reduced") to create the summary table you are interested in.
Sawzall	Pig	This is a language for creating MapReduce jobs.
Go	None	Go is flexible open-source, general-purpose computer language that makes it easier to do parallel data processing.
Dremel, BigQuery	Hive, Drill, Impala	This is a tool that allows data queries to be written in a simplified form of of Structured Query Language (SQL). With Dremel it is possible to run an SQL query on a petabtye of data (1,000 terabytes) in a few seconds.

of an art, which can be learned only by practice, but data-cleaning tools such as OpenRefine and DataWrangler can be used to assist in data cleansing.

Data analysis in statistics and econometrics can be broken down into four categories: 1) prediction, 2) summarization, 3) estimation, and 4) hypothesis testing. Machine learning is concerned primarily with prediction; the closely related field of data mining is also concerned with summarization, and particularly with finding interesting patterns in the data. Econometricians, statisticians, and data mining specialists are generally looking for insights that can be extracted from the data. Machine learning specialists are often primarily concerned with developing high-performance computer systems that can provide useful predictions in the presence of challenging computational constraints. Data science, a somewhat newer term, is concerned with both prediction and summarization, but also with data manipulation, visualization, and other similar tasks. Note that terminology is not standardized in these areas, so these descriptions reflect general usage, not hard-and-fast definitions. Other terms used to describe computer-assisted data analysis include knowledge extraction, information discovery, information harvesting, data archaeology, data pattern processing, and exploratory data analysis.

Much of applied econometrics is concerned with detecting and summarizing relationships in the data. The most common tool used for summarization is (linear) regression analysis. As we shall see, machine learning offers a set of tools that can usefully summarize various sorts of nonlinear relationships in the data. We will focus on these regression-like tools because they are the most natural for economic applications.

In the most general formulation of a statistical prediction problem, we are interested in understanding the conditional distribution of some variable y given some other variables $x = (x_1, \ldots, x_P)$. If we want a point prediction, we can use the mean or median of the conditional distribution.

In machine learning, the x-variables are usually called "predictors" or "features." The focus of machine learning is to find some function that provides a good prediction of y as a function of x. Historically, most work in machine learning has involved cross-section data where it is natural to think of the data being independent and identically distributed (IID) or at least independently distributed. The data may be "fat," which means lots of predictors relative to the number of observations, or "tall" which means lots of observations relative to the number of predictors.

We typically have some observed data on y and x, and we want to compute a "good" prediction of y given new values of x. Usually "good" means it minimizes some loss function such as the sum of squared residuals, mean of absolute value of residuals, and so on. Of course, the relevant loss is that associated with *new* out-of-sample observations of x, not the observations used to fit the model.

When confronted with a prediction problem of this sort an economist would think immediately of a linear or logistic regression. However, there may be better choices, particularly if a lot of data is available. These include nonlinear methods such as 1) classification and regression trees (CART); 2) random forests; and 3) penalized regression such as LASSO, LARS, and elastic nets. (There are also other techniques, such as neural nets, deep learning, and support vector machines, which I do not cover in this review.) Much more detail about these methods can be found in machine learning texts; an excellent treatment is available in Hastie, Tibshirani, and Friedman (2009), which can be freely downloaded. Additional suggestions for further reading are given at the end of this article.

General Considerations for Prediction

Our goal with prediction is typically to get good *out-of-sample predictions*. Most of us know from experience that it is all too easy to construct a predictor that works well in-sample but fails miserably out-of-sample. To take a trivial example, n linearly independent regressors will fit n observations perfectly but will usually have poor out-of-sample performance. Machine learning specialists refer to this phenomenon as the "overfitting problem" and have come up with several ways to deal with it.

First, since simpler models tend to work better for out-of-sample forecasts, machine learning experts have come up with various ways to penalize models for excessive complexity. In the machine learning world, this is known as "regularization," and we will describe some examples below. Economists tend to prefer simpler models for the same reason, but have not been as explicit about quantifying complexity costs.

Second, it is conventional to divide the data into separate sets for the purpose of training, testing, and validation. You use the training data to estimate a model, the validation data to choose your model, and the testing data to evaluate how well your chosen model performs. (Often validation and testing sets are combined.)

Third, if we have an explicit numeric measure of model complexity, we can view it as a parameter that can be "tuned" to produce the best out of sample predictions. The standard way to choose a good value for such a tuning parameter is to use *k-fold cross-validation*.

1. Divide the data into k roughly equal subsets (folds) and label them by $s = 1, \ldots, k$. Start with subset $s = 1$.
2. Pick a value for the tuning parameter.
3. Fit your model using the $k - 1$ subsets other than subset s.
4. Predict for subset s and measure the associated loss.
5. Stop if $s = k$, otherwise increment s by 1 and go to step 2.

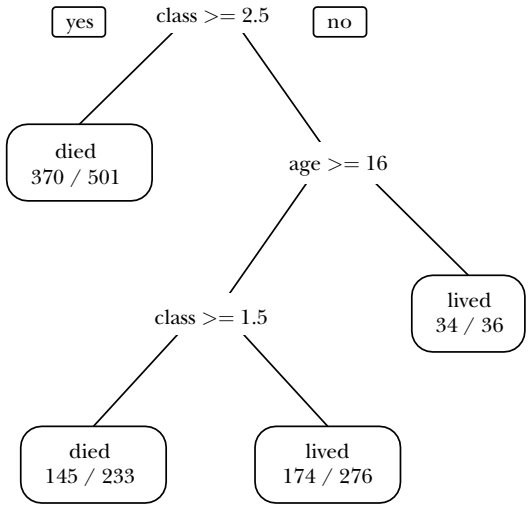
Common choices for k are 10, 5, and the sample size minus 1 ("leave one out"). After cross-validation, you end up with k values of the tuning parameter and the associated loss which you can then examine to choose an appropriate value for the tuning parameter. Even if there is no tuning parameter, it is prudent to use cross-validation to report goodness-of-fit measures since it measures out-of-sample performance, which is generally more meaningful than in-sample performance.

The test-train cycle and cross-validation are very commonly used in machine learning and, in my view, should be used much more in economics, particularly when working with large datasets. For many years, economists have reported in-sample goodness-of-fit measures using the excuse that we had small datasets. But now that larger datasets have become available, there is no reason not to use separate training and testing sets. Cross-validation also turns out to be a very useful technique, particularly when working with reasonably large data. It is also a much more realistic measure of prediction performance than measures commonly used in economics.

Classification and Regression Trees

Let us start by considering a discrete variable regression where our goal is to predict a 0–1 outcome based on some set of features (what economists would call explanatory variables or predictors). In machine learning, this is known as a

Figure 1
A Classification Tree for Survivors of the *Titanic*



Note: See text for interpretation.

classification problem. A common example would be classifying email into "spam" or "not spam" based on characteristics of the email. Economists would typically use a generalized linear model like a logit or probit for a classification problem.

A quite different way to build a classifier is to use a decision tree. Most economists are familiar with decision trees that describe a sequence of decisions that results in some outcome. A tree classifier has the same general form, but the decision at the end of the process is a choice about how to classify the observation. The goal is to construct (or "grow") a decision tree that leads to good out-of-sample predictions.

Ironically, one of the earliest papers on the automatic construction of decision trees (Morgan and Sonquist 1963) was coauthored by an economist. However, the technique did not really gain much traction until 20 years later in the work of Breiman, Friedman, Olshen, and Stone (1984). Nowadays this prediction technique is known as "classification and regression trees," or "CART."

To illustrate the use of tree models, I used the **R** package **rpart** to find a tree that predicts *Titanic* survivors using just two variables: age and class of travel.[1] The resulting tree is shown in Figure 1, and the rules depicted in the tree are shown in Table 2. The rules fit the data reasonably well, misclassifying about 30 percent of the observations in the testing set.

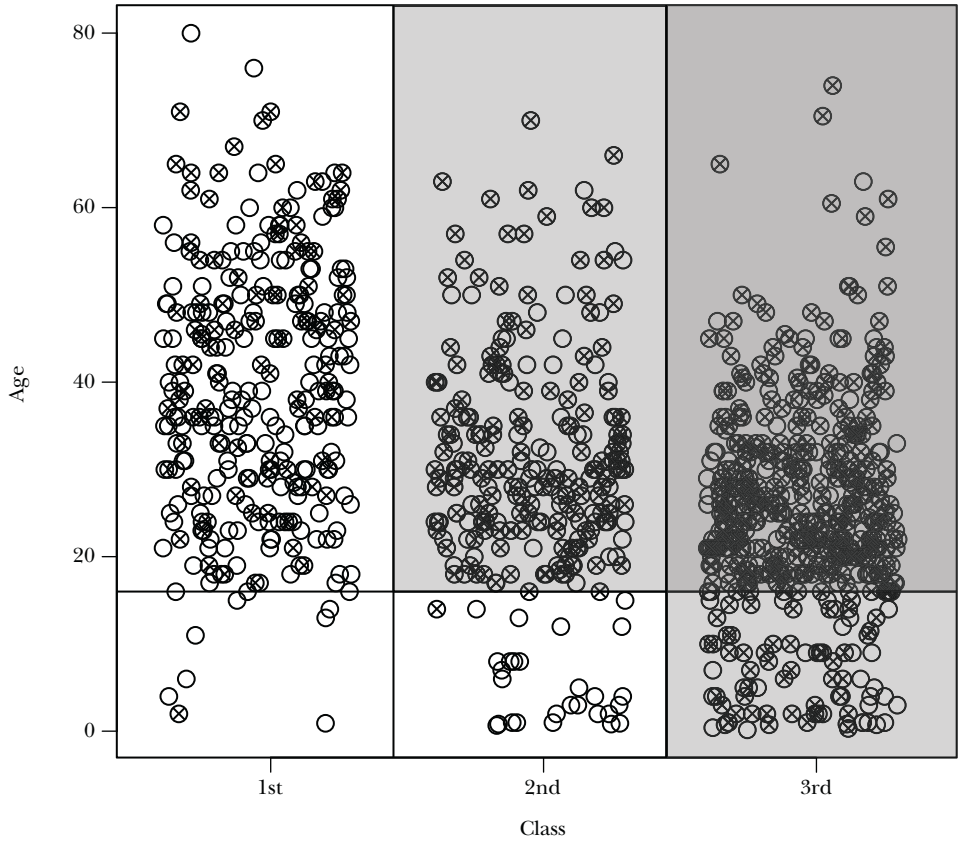
This classification can also be depicted in the "partition plot" (Figure 2), which shows how the tree divides up the space of age and class pairs into rectangular

[1] All data and code used in this paper can be found in the online Appendix available at http://e-jep.org.

Table 2
Tree Model in Rule Form

Features	*Predicted*	*Actual/Total*
Class 3	Died	370/501
Class 1–2, younger than 16	Lived	34/36
Class 2, older than 16	Died	145/233
Class 1, older than 16	Lived	174/276

Figure 2
The Simple Tree Model Predicts Death in Shaded Region
(empty circles indicate survival; circles with x *'s indicate death)*



regions. Of course, the partition plot can only be used for two variables, while a tree representation can handle an arbitrarily large number.

It turns out that there are computationally efficient ways to construct classification trees of this sort. These methods generally are restricted to binary trees (two branches

Table 3
Logistic Regression of Survival versus Age

Coefficient	*Estimate*	*Standard error*	*t value*	*p value*
Intercept	0.465	0.0350	13.291	0.000
Age	−0.002	0.001	−1.796	0.072

Note: Logistic regression relating survival (0 or 1) to age in years.

at each node). They can be used for classification with multiple outcomes ("classification trees") or with continuous dependent variables ("regression trees").

Trees tend to work well for problems where there are important nonlinearities and interactions. As an example, let us continue with the *Titanic* data and create a tree that relates survival to age. In this case, the rule generated by the tree is very simple: predict "survive" if age < 8.5 years. We can examine the same data with a logistic regression to estimate the probability of survival as a function of age, with results reported in Table 3.

The tree model suggests that age is an important predictor of survival, while the logistic model says it is barely important. This discrepancy is explained in Figure 3 where we plot survival rates by age bins. Here we see that survival rates for the youngest passengers were relatively high, and survival rates for older passengers were relatively low. For passengers between these two extremes, age didn't matter very much. So what mattered for survival is not so much age, but whether the passenger was a child or elderly. It would be difficult to discover this pattern from a logistic regression alone.[2]

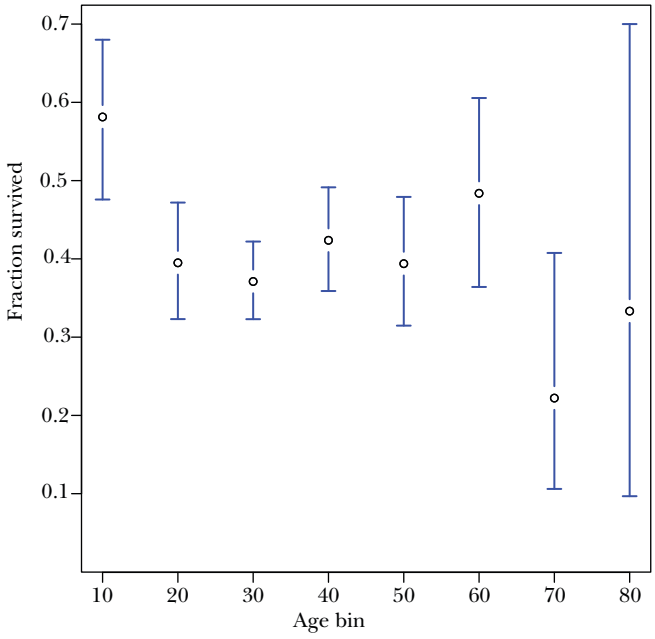
Trees also handle missing data well. Perlich, Provost, and Simonoff (2003) examined several standard datasets and found that "logistic regression is better for smaller data sets and tree induction for larger data sets." Interestingly enough, trees tend *not* to work very well if the underlying relationship really is linear, but there are hybrid models such as RuleFit (Friedman and Popescu 2005) that can incorporate both tree and linear relationships among variables. However, even if trees may not improve on predictive accuracy compared to linear models, the age example shows that they may reveal aspects of the data that are not apparent from a traditional linear modeling approach.

Pruning Trees

One problem with trees is that they tend to overfit the data. Just as a regression with n observations and n variables will give you a good fit in-sample, a tree with many branches will also fit the training data well. In either case, predictions using new data, such as the test set, could be very poor.

[2] It is true that if you *knew* that there was a nonlinearity in age, you could use age dummies in the logit model to capture this effect. However the tree formulation made this nonlinearity immediately apparent.

Figure 3
***Titanic* Survival Rates by Age Group**



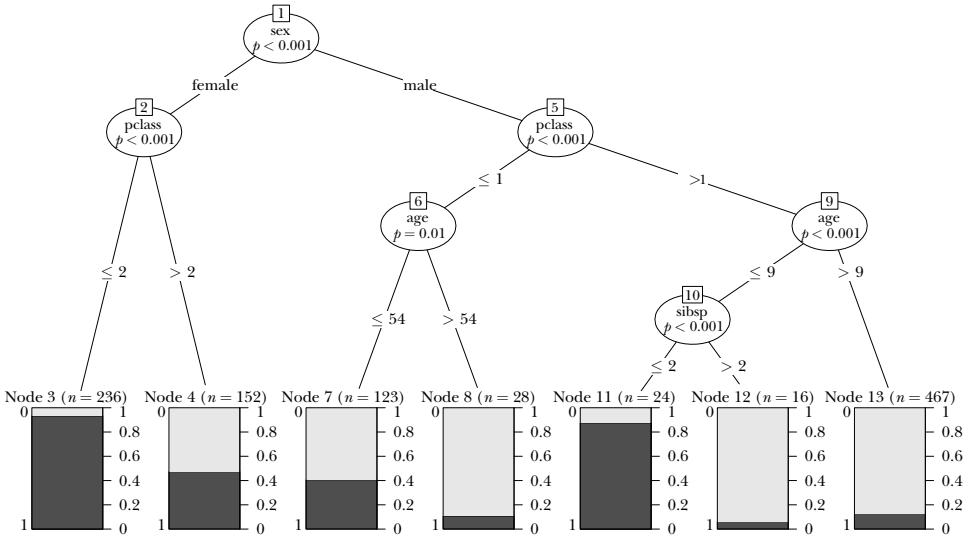
Notes: The figure shows the mean survival rates for different age groups along with confidence intervals. The age bin 10 means "10 and younger," the next age bin is "older than 10 through 20," and so on.

The most common solution to this problem is to "prune" the tree by imposing a cost for complexity. There are various measures of complexity, but a common one is the number of terminal nodes (also known as "leafs"). The cost of complexity is a tuning parameter that is chosen to provide the best out-of-sample predictions, which is typically measured using the 10-fold cross-validation procedure mentioned earlier.

A typical tree estimation session might involve dividing your data into ten folds, using nine of the folds to grow a tree with a particular complexity, and then predict on the excluded fold. Repeat the estimation with different values of the complexity parameter using other folds and choose the value of the complexity parameter that minimizes the out-of-sample classification error. (Some researchers recommend being a bit more aggressive and advocate choosing the complexity parameter that is one standard deviation lower than the loss-minimizing value.)

Of course, in practice, the computer program handles most of these details for you. In the examples in this paper, I mostly use default choices to keep things simple, but in practice these defaults will often be adjusted by the analyst. As with any other statistical procedure, skill, experience, and intuition are helpful in coming up with a good answer. Diagnostics, exploration, and experimentation are just as useful with these methods as with regression techniques.

Figure 4
A ctree for Survivors of the *Titanic*
(black bars indicate fraction of the group that survived)



Note: See text for interpretation.

There are many other approaches to creating trees, including some that are explicitly statistical in nature. For example, a "conditional inference tree," or ctree for short, chooses the structure of the tree using a sequence of hypothesis tests. The resulting trees tend to need very little pruning (Hothorn, Hornik, and Zeileis 2006). An example for the *Titanic* data is shown in Figure 4.

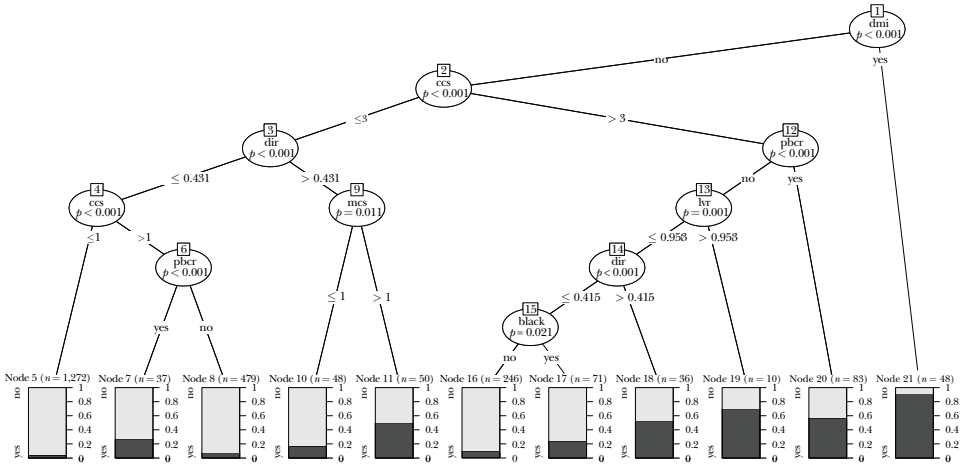
The first node divides by gender. The second node then divides by class. In the right-hand branches, the third node divides by age, and a fourth node divides by the number of siblings plus spouse aboard. The bins at the bottom of the figure show the total number of people in that leaf and a graphical depiction of their survival rate. One might summarize this tree by the following principle: "women and children first . . . particularly if they were traveling first class." This simple example again illustrates that classification trees can be helpful in summarizing relationships in data, as well as predicting outcomes.[3]

An Economic Example Using Home Mortgage Disclosure Act Data

Munnell, Tootell, Browne, and McEneaney (1996) examined mortgage lending in Boston to see if race played a significant role in determining who was approved for a mortgage. The primary econometric technique was a logistic regression where

[3] For two excellent tutorials on tree methods that use the *Titanic* data, see Stephens and Wehrley (2014).

Figure 5
Home Mortgage Disclosure Act (HMDA) Data Tree



Notes: Figure 5 shows a conditional tree estimated using the **R** package **party**. The black bars indicate the fraction of each group who were denied mortgages. The most important determinant of this is the variable "dmi," or "denied mortgage insurance." Other variables are: "dir," debt payments to total income ratio; "hir," housing expenses to income ratio; "lvr," ratio of size of loan to assessed value of property; "ccs," consumer credit score; "mcs," mortgage credit score; "pbcr," public bad credit record; "dmi," denied mortgage insurance; "self," self-employed; "single," applicant is single; "uria," 1989 Massachusetts unemployment rate applicant's industry; "condominium," unit is condominium; "black," race of applicant black; and "deny," mortgage application denied.

race was included as one of the predictors. The coefficient on race showed a statistically significant negative impact on probability of getting a mortgage for black applicants. This finding prompted considerable subsequent debate and discussion; see Ladd (1998) for an overview.

Here I examine this question using the tree-based estimators described in the previous section. The data consists of 2,380 observations of 12 predictors, one of which was race. Figure 5 shows a conditional tree estimated using the **R** package **party**.

The tree fits pretty well, misclassifying 228 of the 2,380 observations for an error rate of 9.6 percent. By comparison, a simple logistic regression does slightly better, misclassifying 225 of the 2,380 observations, leading to an error rate of 9.5 percent. As you can see in Figure 5, the most important variable is "dmi" = "denied mortgage insurance." This variable alone explains much of the variation in the data. The race variable ("black") shows up far down the tree and seems to be relatively unimportant.

One way to gauge whether a variable is important is to exclude it from the prediction and see what happens. When this is done, it turns out that the accuracy of the tree-based model doesn't change at all: exactly the same cases are misclassified. Of course, it is perfectly possible that there was racial discrimination

elsewhere in the mortgage process, or that some of the variables included are highly correlated with race. But it is noteworthy that the tree model produced by standard procedures that omits race fits the observed data just as well as a model that includes race.

Boosting, Bagging, Bootstrap

There are several useful ways to improve classifier performance. Interestingly enough, some of these methods work by *adding* randomness to the data. This seems paradoxical at first, but adding randomness turns out to be a helpful way of dealing with the overfitting problem.

Bootstrap involves choosing (with replacement) a sample of size n from a dataset of size n to estimate the sampling distribution of some statistic. A variation is the "m out of n bootstrap" which draws a sample of size m from a dataset of size $n > m$.

Bagging involves averaging across models estimated with several different bootstrap samples in order to improve the performance of an estimator.

Boosting involves repeated estimation where misclassified observations are given increasing weight in each repetition. The final estimate is then a vote or an average across the repeated estimates.[4]

Econometricians are well-acquainted with the bootstrap but rarely use the other two methods. Bagging is primarily useful for nonlinear models such as trees (Friedman and Hall 2007). Boosting tends to improve predictive performance of an estimator significantly and can be used for pretty much any kind of classifier or regression model, including logits, probits, trees, and so on.

It is also possible to combine these techniques and create a "forest" of trees that can often significantly improve on single-tree methods. Here is a rough description of how such "random forests" work.

Random Forests

Random forests is a technique that uses multiple trees. A typical procedure uses the following steps.

1. Choose a bootstrap sample of the observations and start to grow a tree.
2. At each node of the tree, choose a random sample of the predictors to make the next decision. Do not prune the trees.
3. Repeat this process many times to grow a forest of trees.
4. In order to determine the classification of a new observation, have each tree make a classification and use a majority vote for the final prediction.

This method produces surprisingly good out-of-sample fits, particularly with highly nonlinear data. In fact, Howard and Bowles (2012) claim "ensembles of decision trees (often known as 'Random Forests') have been the most successful general-purpose algorithm in modern times." They go on to indicate that

[4] Boosting is often used with decision trees, where it can dramatically improve their predictive performance.

"the algorithm is very simple to understand, and is fast and easy to apply." See also Caruana and Niculescu-Mitzil (2006) who compare several different machine learning algorithms and find that ensembles of trees perform quite well. There are a number of variations and extensions of the basic "ensemble of trees" model such as Friedman's "Stochastic Gradient Boosting" (Friedman 2002).

One defect of random forests is that they are a bit of a black box—they don't offer simple summaries of relationships in the data. As we have seen earlier, a single tree can offer some insight about how predictors interact. But a forest of a thousand trees cannot be easily interpreted. However, random forests can determine which variables are "important" in predictions in the sense of contributing the biggest improvements in prediction accuracy.

Note that random forests involves quite a bit of randomization; if you want to try them out on some data, I strongly suggest choosing a particular seed for the random number generator so that your results can be reproduced. (See the online supplement for examples.)

I ran the random forest method on the HMDA data and found that it misclassified 223 of the 2,380 cases, a small improvement over the logit and the ctree. I also used the importance option in random forests to see how the predictors compared. It turned out that "dmi" was the most important predictor and race was second from the bottom, which is consistent with the ctree analysis.

Variable Selection

Let us return to the familiar world of linear regression and consider the problem of variable selection. There are many such methods available, including stepwise regression, principal component regression, partial least squares, Akaike information criterion (AIC) and Bayesian information criterion (BIC) complexity measures, and so on. Castle, Qin, and Reed (2009) describe and compare 21 different methods.

LASSO and Friends

Here we consider a class of estimators that involves penalized regression. Consider a standard multivariate regression model where we predict y_t as a linear function of a constant, b_0, and P predictor variables. We suppose that we have standardized all the (nonconstant) predictors so they have mean zero and variance one.

Consider choosing the coefficients $(b_1, \ldots, b_P)$ for these predictor variables by minimizing the sum of squared residuals plus a penalty term of the form

$$\lambda \sum_{p=1}^{P} \left[(1 - \alpha) \, | b_p | + \alpha \, | b_p |^2 \right].$$

This estimation method is called *elastic net regression*; it contains three other methods as special cases. If there is no penalty term ($\lambda = 0$), this is *ordinary least squares*. If $\alpha = 1$, so that there is only the quadratic constraint, this is *ridge regression*.

If $\alpha = 0$, this is called the *LASSO*, an acronym for "least absolute shrinkage and selection operator."

These penalized regressions are classic examples of regularization. In this case, the complexity is the number and size of predictors in the model. All of these methods tend to shrink the least squares regression coefficients towards zero. The LASSO and elastic net typically produces regressions where some of the variables are set to be exactly zero. Hence this is a relatively straightforward way to do variable selection.

It turns out that these estimators can be computed quite efficiently, so doing variable selection on reasonably large problems is computationally feasible. They also seem to provide good predictions in practice.

Spike-and-Slab Regression

Another approach to variable selection that is novel to most economists is spike-and-slab regression, a Bayesian technique. Suppose that you have P possible predictors in some linear model. Let γ be a vector of length P composed of zeros and ones that indicate whether or not a particular variable is included in the regression.

We start with a Bernoulli prior distribution on γ; for example, initially we might think that all variables have an equally likely chance of being in the regression. Conditional on a variable being in the regression, we specify a prior distribution for the regression coefficient associated with that variable. For example, we might use a Normal prior with mean 0 and a large variance. These two priors are the source of the method's name: the "spike'" is the probability of a coefficient being nonzero; the "slab" is the (diffuse) prior describing the values that the coefficient can take on.

Now we take a draw of γ from its prior distribution, which will just be a list of variables in the regression. Conditional on this list of included variables, we take a draw from the prior distribution for the coefficients. We combine these two draws with the likelihood in the usual way, which gives us a draw from posterior distribution on both probability of inclusion and the coefficients. We repeat this process thousands of times using a Markov Chain Monte Carlo (MCMC) technique which gives us a table summarizing the posterior distribution for γ (indicating variable inclusion), β (the coefficients), and the associated prediction of y. We can summarize this table in a variety of ways. For example, we can compute the average value of γ_p which shows the posterior probability that the variable p is included in the regressions.

An Economic Example: Growth Regressions

We illustrate these different methods of variable selection using data from Sala-i-Martín (1997). This exercise involved examining a dataset of 72 counties and 42 variables in order to see which variables appeared to be important predictors of economic growth. Sala-i-Martín (1997) computed at all possible subsets of regressors of manageable size and used the results to construct an importance measure he called CDF(0). Ley and Steel (2009) investigated the same question using Bayesian

Table 4

Comparing Variable Selection Algorithms: Which Variables Appeared as Important Predictors of Economic Growth?

Predictor	*Bayesian model averaging*	*CDF(0)*	*LASSO*	*Spike-and-Slab*
GDP level 1960	1.000	1.000	-	0.9992
Fraction Confucian	0.995	1.000	2	0.9730
Life expectancy	0.946	0.942	-	0.9610
Equipment investment	0.757	0.997	1	0.9532
Sub-Saharan dummy	0.656	1.000	7	0.5834
Fraction Muslim	0.656	1.000	8	0.6590
Rule of law	0.516	1.000	-	0.4532
Open economy	0.502	1.000	6	0.5736
Degree of capitalism	0.471	0.987	9	0.4230
Fraction Protestant	0.461	0.966	5	0.3798

Source: The table is based on that in Ley and Steel (2009); the data analyzed is from Sala-i-Martín (1997).
Notes: We illustrate different methods of variable selection. This exercise involved examining a dataset of 72 counties and 42 variables in order to see which variables appeared to be important predictors of economic growth. The table shows ten predictors that were chosen by Sala-i-Martín (1997) using a CDF(0) measure defined in the 1997 paper; Ley and Steel (2009) using Bayesian model averaging, LASSO, and spike-and-slab regressions. Metrics used are not strictly comparable across the various models. The "Bayesian model averaging" and "Spike-and-Slab" columns are posterior probabilities of inclusion; the "LASSO" column just shows the ordinal importance of the variable or a dash indicating that it was not included in the chosen model; and the CDF(0) measure is defined in Sala-i-Martín (1997).

model averaging, a technique related to, but not identical with, spike-and-slab. Hendry and Krolzig (2004) examined an iterative significance test selection method.

Table 4 shows ten predictors that were chosen by Sala-i-Martín (1997) using his two million regressions, Ley and Steel (2009) using Bayesian model averaging, LASSO, and spike-and-slab. The table is based on that in Ley and Steel (2009) but metrics used are not strictly comparable across the various models. The "Bayesian model averaging" and "spike-slab" columns show posterior probabilities of inclusion; the "LASSO" column just shows the ordinal importance of the variable or a dash indicating that it was not included in the chosen model; and the CDF(0) measure is defined in Sala-i-Martín (1997).

The LASSO and the Bayesian techniques are very computationally efficient and would likely be preferred to exhaustive search. All four of these variable selection methods give similar results for the first four or five variables, after which they diverge. In this particular case, the dataset appears to be too small to resolve the question of what is "important" for economic growth.

Variable Selection in Time Series Applications

The machine learning techniques described up until now are generally applied to cross-sectional data where independently distributed data is a plausible assumption. However, there are also techniques that work with time series. Here we

describe an estimation method that we call Bayesian Structural Time Series (BSTS) that seems to work well for variable selection problems in time series applications.

Our research in this area was motivated by Google Trends data, which provides an index of the volume of Google queries on specific terms. One might expect that queries on "file for unemployment" might be predictive of the actual rate of filings for initial claims, or that queries on "Orlando vacation" might be predictive of actual visits to Orlando. Indeed, in Choi and Varian (2009, 2012), Goel, Hofman, Lahaie, Pennock, and Watts (2010), Carrière-Swallow and Labbé (2011), McLaren and Shanbhoge (2011), Artola and Galan (2012), Hellerstein and Middeldorp (2012), and other papers, many researchers have shown that Google queries do have significant short-term predictive power for various economic metrics.

The challenge is that there are billions of queries so it is hard to determine exactly which queries are the most predictive for a particular purpose. Google Trends classifies the queries into categories, which helps a little, but even then we have hundreds of categories as possible predictors so that overfitting and spurious correlation are a serious concern. Bayesian Structural Time Series is designed to address these issues. We offer a very brief description here; more details are available in Scott and Varian (2013a, 2013b).

Consider a classic time series model with *constant* level, linear time trend, and regressor components:

$$yt = \mu + bt + \beta xt + et.$$

The "local linear trend" is a stochastic generalization of this model where the level and time trend can vary through time.

$$\text{Observation: } y_t = \mu_t + z_t + e_{1t} = \text{level} + \text{regression}$$

$$\text{State variable 1: } \mu_t = \mu_{t-1} + b_{t-1} + e_{2t} = \text{random walk} + \text{trend}$$

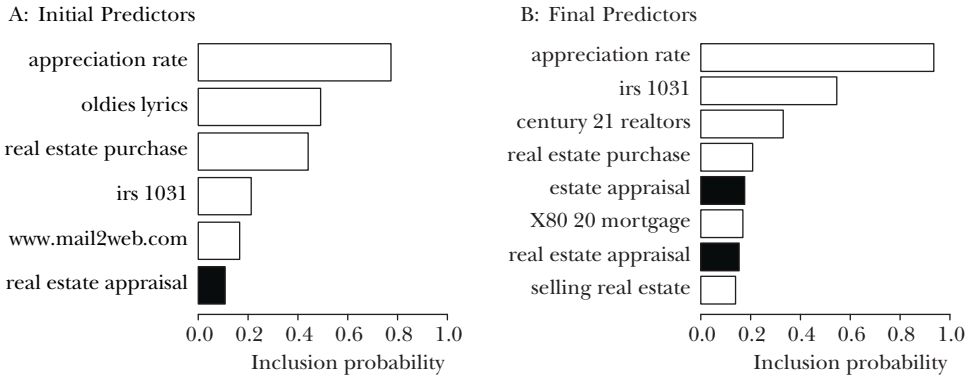
$$\text{State variable 2: } z_t = \beta x_t = \text{regression}$$

$$\text{State variable 3: } b_t = b_{t-1} + e_{3t} = \text{random walk for trend}$$

It is easy to add an additional state variable for seasonality if that is appropriate. The parameters to estimate are the regression coefficients β and the variances of (e_{it}) for $i = 1, \ldots 3$. We can then use these estimates to construct the optimal forecast based on techniques drawn from the literature on Kalman filters.

For the regression, we use the spike-and-slab variable choice mechanism described above. A draw from the posterior distribution now involves a draw of variances of (e_{1t}, e_{2t}, e_{3t}) a draw of the vector γ that indicates which variables are in the regression, and a draw of the regression coefficients β for the included variables. The draws of μ_t, b_t, and β can be used to construct estimates of y_t and forecasts for y_{t+1}. We end up with an (estimated) posterior distribution for each parameter of

Figure 6
An Example Using Bayesian Structural Time Series (BSTS)
(finding Google queries that are predictors of new home sales)



Source: Author using HSN1FNSA data from the St. Louis Federal Reserve Economic Data.
Notes: Consider the nonseasonally adjusted data for new homes sold in the United States, which is (HSN1FNSA) from the St. Louis Federal Reserve Economic Data. This time series can be submitted to Google Correlate, which then returns the 100 queries that are the most highly correlated with the series. We feed that data into the BSTS system, which identifies the predictors with the largest posterior probabilities of appearing in the housing regression; these are shown in Figure 6A. In these figures, black bars indicate a negative relationship, and white bars indicate a positive relationship. Two predictors, "oldies lyrics" and "www.mail2web" appear to be spurious so we remove them and re-estimate, yielding the results in Figure 6B.

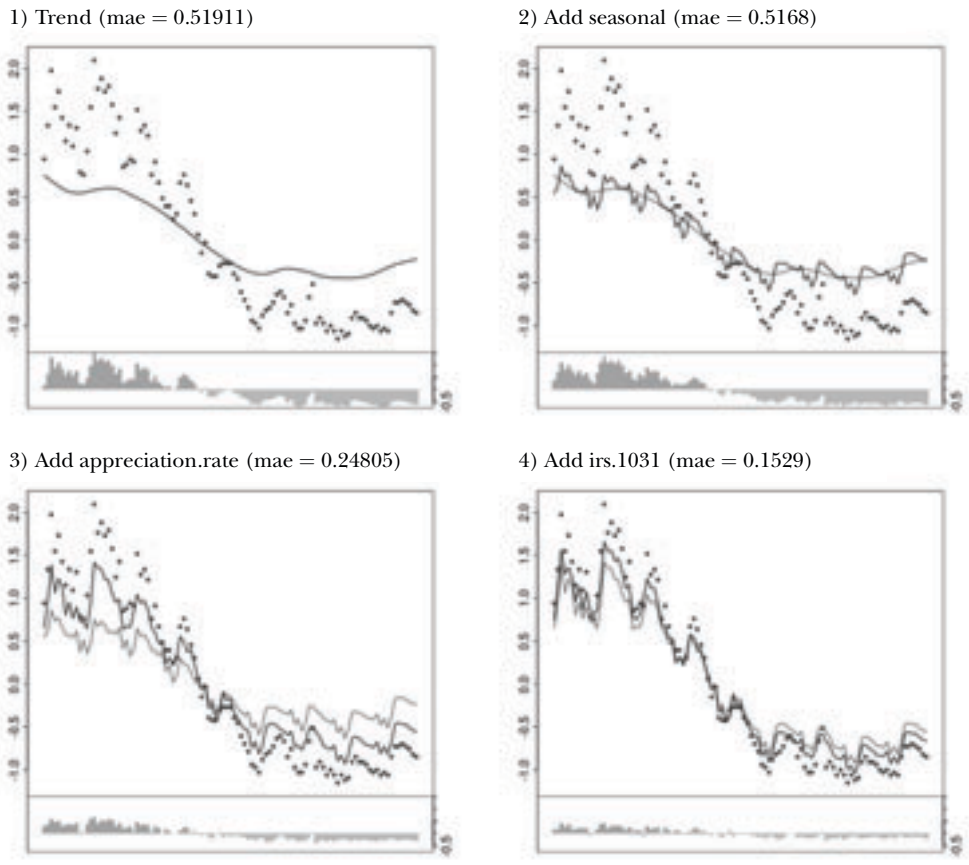
interest. If we seek a point prediction, we can average over these draws, which is essentially a form of Bayesian model averaging.

As an example, consider the nonseasonally adjusted data for new homes sold in the United States, which is (HSN1FNSA) from the St. Louis Federal Reserve Economic Data. This time series can be submitted to Google Correlate, which then returns the 100 queries that are the most highly correlated with the series. We feed that data into the BSTS system, which identifies the predictors with the largest posterior probabilities of appearing in the housing regression; these are shown in Figure 6A.In these figures, black bars indicate a negative relationship and white bars indicate a positive relationship. Two predictors, "oldies lyrics" and "www.mail2web" appear to be spurious so we remove them and re-estimate, yielding the results in Figure 6B.

The fit is shown in Figure 7, which shows the incremental contribution of the trend, seasonal, and two of the regressors. Even with only two predictors, queries on "appreciation rate" and queries on "irs 1031," we get a pretty good fit.[5]

[5] IRS section 1031 has to do with deferring capital gains on certain sorts of property exchange.

Figure 7
Fit for the Housing Regression: Incremental Contribution of Trend, Seasonal, and Two Regressors



Source: Author using (HSN1FNSA) data from the St. Louis Federal Reserve.
Notes: The plots show the impact of the trend, seasonal, and a few individual regressors. Data has been standardized to have mean zero and variance 1. The residuals are shown on the bottom. The abbreviation "mae" stands for "mean absolute error."

Econometrics and Machine Learning

There are a number of areas where there would be opportunities for fruitful collaboration between econometrics and machine learning. I mentioned above that most machine learning uses independent and identically distributed data. However, the Bayesian Structural Time Series model shows that some of these techniques can be adopted for time series models. It is also possible to use machine learning techniques to look at panel data, and there has been some work in this direction.

However, the most important area for collaboration involves causal inference. Econometricians have developed several tools for causal inference such as

instrumental variables, regression discontinuity, difference-in-differences, and various forms of natural and designed experiments (Angrist and Krueger 2001). Machine learning work has, for the most part, dealt with pure prediction. In a way, this is ironic, since theoretical computer scientists, such as Pearl (2009a, b) have made significant contributions to causal modeling. However, it appears that these theoretical advances have not as yet been incorporated into machine learning practice to a significant degree.

Causality and Prediction

As economists know well, there is a big difference between correlation and causation. A classic example: there are often more police in precincts with high crime, but that does not imply that increasing the number of police in a precinct would increase crime.

The machine learning models we have described so far have been entirely about prediction. If our data were generated by policymakers who assigned police to areas with high crime, then the observed relationship between police and crime rates could be highly predictive for the *historical* data but not useful in predicting the causal impact of explicitly *assigning* additional police to a precinct.

To enlarge on this point, let us consider an experiment (natural or designed) that attempts to estimate the impact of some policy, such as adding police to precincts. There are two critical questions.

1) How will police be assigned to precincts in both the experiment and the policy implementation? Possible assignment rules could be 1) random, 2) based on perceived need, 3) based on cost of providing service, 4) based on resident requests, 5) based on a formula or set of rules, 6) based on asking for volunteers, and so on. Ideally the assignment procedure in the experiment will be similar to that used in the policy. Developing accurate predictions about which precincts will receive additional police under the proposed policy based on the experimental data can clearly be helpful in predicting the expected impact of the policy.

2) What will be the impact of these additional police in both the experiment and the policy? As Rubin (1974) and many subsequent authors have emphasized, when we want to estimate the *causal* impact of some treatment we need to compare the outcome with the intervention to what *would have happened* without the intervention. But this counterfactual cannot be observed, so it must be predicted by some model. The better predictive model you have for the counterfactual, the better you will be able to estimate the causal effect, a rule that is true for both pure experiments and natural experiments.

So even though a predictive model will not necessarily allow one to conclude anything about causality by itself, such models may help in estimating the causal impact of an intervention when it occurs.

To state this in a slightly more formal way, consider the identity from Angrist and Pischke (2009, p. 11):

$$\begin{aligned}\text{observed difference in outcome} &= \text{average treatment effect on the treated} \\ &\quad + \text{selection bias.}\end{aligned}$$

If you want to model the average treatment effect as a function of other variables, you will usually need to model both the observed difference in outcome and the selection bias. The better your predictive model for those components, the better your estimate of the average treatment effect will be. Of course, if you have a true randomized treatment–control experiment, selection bias goes away and those treated are an unbiased random sample of the population.

To illustrate these points, let us consider the thorny problem of estimating the causal effect of advertising on sales (Lewis and Rao 2013). The difficulty is that there are many confounding variables, such as seasonality or weather, that cause both increased ad exposures and increased purchases by consumers. For example, consider the (probably apocryphal) story about an advertising manager who was asked why he thought his ads were effective. "Look at this chart," he said. "Every December I increase my ad spend and, sure enough, purchases go up." Of course, in this case, seasonality can be included in the model. However, generally there will be other confounding variables that affect both exposure to ads and the propensity of purchase, which make causal interpretations of observed relationships problematic.

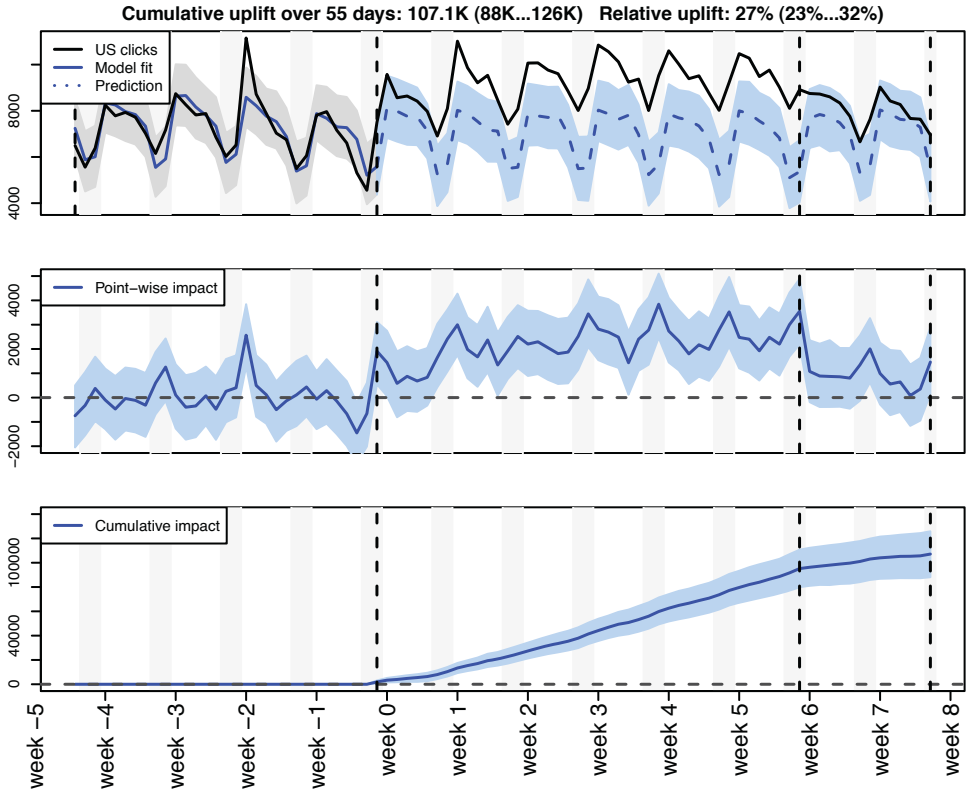
The ideal way to estimate advertising effectiveness is, of course, to run a controlled experiment. In this case the control group provides an estimate of the counterfactual: what would have happened without ad exposures. But this ideal approach can be quite expensive, so it is worth looking for alternative ways to predict the counterfactual. One way to do this is to use the Bayesian Structural Time Series (BSTS) method described earlier.

Suppose a given company wants to determine the impact of an advertising campaign on visits to its website. It first uses BSTS (or some other technique) to build a model predicting the time series of visits as a function of its past history, seasonal effects, and other possible predictors such as Google queries on its company name, its competitors' names, or products that it produces. Since there are many possible choices for predictors, it is important to use some variable selection mechanism such as those described earlier.

It next runs an ad campaign for a few weeks and records visits during this period. Finally, it makes a forecast of what visits *would have been* in the absence of the ad campaign using the model developed in the first stage. Comparing the actual visits to the counterfactual visits gives us an estimate of the causal effect of advertising.

Figure 8, shows the outcome of such a procedure. It is based on the approach proposed in Brodersen, Gallusser, Koehler, Remy, and Scott (2013), but the covariates are chosen automatically from Google Trends categories using Bayesian Structural Time Series (BSTS). Panel A shows the actual visits and the prediction

Figure 8
Actual and Predicted Website Visits



Source: This example is based on the approach proposed in Brodersen, Gallusser, Koehler, Remy, and Scott (2013), but the covariates are chosen automatically from Google Trends categories using Bayesian Structural Time Series (BSTS).
Notes: Suppose a given company wants to determine the impact of an advertising campaign on its website visits. Panel A shows the actual visits and the prediction of what the visits would have been without the campaign based on the BSTS forecasting model. Panel B shows the difference between actual and predicted visits, and Panel C shows the cumulative difference.

of what the visits would have been without the campaign based on the BSTS forecasting model. Panel B shows the difference between actual and predicted visits, and Panel C shows the cumulative difference. It is clear from this figure that there was a significant causal impact of advertising, which can then be compared to the cost of the advertising to evaluate the campaign.

This procedure does not use a control group in the conventional sense. Rather it uses a general time series model based on trend extrapolation, seasonal effects, and relevant covariates to forecast what would have happened without the ad campaign.

A good predictive model can be better than a randomly chosen control group, which is usually thought to be the gold standard. To see this, suppose that you run

an ad campaign in 100 cities and retain 100 cities as a control. After the experiment is over, you discover the weather was dramatically different across the cities in the study. Should you add weather as a predictor of the counterfactual? Of course! If weather affects sales (which it does), then you will get a more accurate prediction of the counterfactual and thus a better estimate of the causal effect of advertising.

Model Uncertainty

An important insight from machine learning is that averaging over many small models tends to give better out-of-sample prediction than choosing a single model.

In 2006, Netflix offered a million dollar prize to researchers who could provide the largest improvement to their existing movie recommendation system. The winning submission involved a "complex blending of no fewer than 800 models," though they also point out that "predictions of good quality can usually be obtained by combining a small number of judiciously chosen methods" (Feuerverger, He, and Khatri 2012). It also turned out that a blend of the best- and second-best submissions outperformed either of them.

Ironically, it was recognized many years ago that averages of macroeconomic model forecasts outperformed individual models, but somehow this idea was rarely exploited in traditional econometrics. The exception is the literature on Bayesian model averaging, which has seen a steady flow of work; see Steel (2011) for a survey.

However, I think that model uncertainty has crept into applied econometrics through the back door. Many papers in applied econometrics present regression results in a table with several different specifications: which variables are included in the controls, which variables are used as instruments, and so on. The goal is usually to show that the estimate of some interesting parameter is not very sensitive to the exact specification used.

One way to think about it is that these tables illustrate a simple form of model uncertainty: how an estimated parameter varies as different models are used. In these papers, the authors tend to examine only a few representative specifications, but there is no reason why they couldn't examine many more if the data were available.

In this period of "big data," it seems strange to focus on *sampling uncertainty,* which tends to be small with large datasets, while completely ignoring *model uncertainty,* which may be quite large. One way to address this is to be explicit about examining how parameter estimates vary with respect to choices of control variables and instruments.

Summary and Further Reading

Since computers are now involved in many economic transactions, big data will only get bigger. Data manipulation tools and techniques developed for small datasets will become increasingly inadequate to deal with new problems. Researchers in machine learning have developed ways to deal with large datasets and economists

interested in dealing with such data would be well advised to invest in learning these techniques.

I have already mentioned Hastie, Tibshirani, and Friedman (2009), who provide detailed descriptions of all the methods discussed here but at a relatively advanced level. James, Witten, Hastie, and Tibshirani (2013) describe many of the same topics at an undergraduate-level, along with **R** code and many examples. (There are several economic examples in the book where the tension between predictive modeling and causal inference is apparent.) Murphy (2012) examines machine learning from a Bayesian point of view.

Venables and Ripley (2002) offer good discussions of these topics with emphasis on applied examples. Leek (2013) presents a number of YouTube videos with gentle and accessible introductions to several tools of data analysis. Howe (2013) provides a somewhat more advanced introduction to data science that also includes discussions of SQL and NoSQL databases. Wu and Kumar (2009) give detailed descriptions and examples of the major algorithms in data mining, while Williams (2011) provides a unified toolkit. Domingos (2012) summarizes some important lessons including "pitfalls to avoid, important issues to focus on and answers to common questions."

■ *Thanks to Jeffrey Oldham, Tom Zhang, Rob On, Pierre Grinspan, Jerry Friedman, Art Owen, Steve Scott, Bo Cowgill, Brock Noland, Daniel Stonehill, Robert Snedegar, Gary King, Fabien Curto-Millet, and the editors of this journal for helpful comments on earlier versions of this paper. The author works for Google, and Google had the right to review this paper before publication.*

References

Angrist, Joshua D., and Alan B. Krueger. 2001. "Instrumental Variables and the Search for Identification: From Supply and Demand to Natural Experiments." *Journal of Economic Perspectives* 5(4): 69–85.

Angrist, Joshua D., and Jörn-Steffen Pischke. 2009. *Mostly Harmless Econometrics: An Empiricist's Companion*. Princeton University Press.

Artola, Concha, and Enrique Galan. 2012. "Tracking the Future on the Web: Construction of Leading Indicators Using Internet Searches." Documentos Ocasionales 1203T, Bank of Spain. http://www.bde.es/webbde/SES/Secciones/Publicaciones/PublicacionesSeriadas/DocumentosOcasionales/12/Fich/do1203e.pdf.

Breiman, Leo, Jerome H. Friedman, R. A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees*. Wadsworth and Brooks/Cole, Monterey.

Brodersen, Kay H., Fabian Gallusser, Jim Koehler, Nicolas Remy, and Steven L. Scott. 2013. "Inferring Causal Impact Using Bayesian Structural Time-Series Models." http://research.google.com/pubs/pub41854.html.

Carrière-Swallow, Yan, and Felipe Labbé. 2011. "Nowcasting with Google Trends in an Emerging Market." *Journal of Forecasting* 32(4): 289–98.

Caruana, Rich, and Alexandru Niculescu-Mizil. 2006. "An Empirical Comparison of Supervised Learning Algorithms." In *Proceedings of the 23rd*

International Conference on Machine Learning, Pittsburgh, PA. Available at: http://www.autonlab.org/icml2006/technical/accepted.html.

Castle, Jennifer L., Xiaochuan Qin, and W. Robert Reed. 2009. "How to Pick the Best Regression Equation: A Review and Comparison of Model Selection Algorithms." Working Paper 13/2009, Department of Economics, University of Canterbury. http://www.econ.canterbury.ac.nz/RePEc/cbt/econwp/0913.pdf.

Choi, Hyunyoung, and Hal Varian. 2009. "Predicting the Present with Google Trends." http://google.com/googleblogs/pdfs/google_predicting_the_present.pdf.

Choi, Hyunyoung, and Hal Varian. 2012. "Predicting the Present with Google Trends." *Economic Record* 88(1): 2–9.

Domingos, Pedro. 2012. "A Few Useful Things to Know about Machine Learning." *Communications of the ACM* 55(10): 78–87.

Einav, Liran, and Jonathan D. Levin. 2013. "The Data Revolution and Economic Analysis." Technical report, NBER Innovation Policy and the Economy Conference, 2013. NBER Working Paper 19035.

Feuerverger, Andrey, Yu He, and Shashi Khatri. 2012. "Statistical Significance of the Netflix Challenge." *Statistical Science* 27(2): 202–231.

Friedman, Jerome. 2002. "Stochastic Gradient Boosting." *Computational Statistics & Data Analysis* 38(4): 367–78.

Friedman, Jerome, and Peter Hall. 2007. "On Bagging and Nonlinear Estimation." *Journal of Statistical Planning and Inference* 137(3): 669–83.

Friedman, Jerome H., and Bogdan E. Popescu. 2005. "Predictive Learning via Rule Ensembles." Technical report, Stanford University. http://www-stat.stanford.edu/~jhf/ftp/RuleFit.pdf

Goel, Sharad, Jake M. Hofman, Sébastien Lahaie, David M. Pennock, and Duncan J. Watts. 2010. "Predicting Consumer Behavior with Web Search." *PNAS* 107(41).

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edition. Springer-Verlag.

Hellerstein, Rebecca, and Menno Middeldorp. 2012. "Forecasting with Internet Search Data." *Liberty Street Economics* Blog of the Federal Reserve Bank of New York, January 4. http://libertystreeteconomics.newyorkfed.org/2012/01/forecasting-with-internet-search-data.html.

Hendry, David F., and Hans-Martin Krolzig. 2004. "We Ran One Regression." *Oxford Bulletin of Economics and Statistics* 66(5): 799–810.

Hothorn, Torsten, Kurt Hornik, and Achim Zeileis. 2006. "Unbiased Recursive Partitioning: A Conditional Inference Framework." *Journal of Computational and Graphical Statistics* 15(3): 651–74.

Howard, Jeremy, and Mike Bowles. 2012. "The Two Most Important Algorithms in Predictive Modeling Today." Strata Conference presentation, February 28. http://strataconf.com/strata2012/public/schedule/detail/22658.

Howe, Bill. 2013. Introduction to Data Science. A course from the University of Washington. https://class.coursera.org/datasci-001/lecture/index.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning: With Applications in R*. New York: Springer.

Ladd, Helen F. 1998. "Evidence on Discrimination in Mortgage Lending." *Journal of Economic Perspectives* 12(2): 41–62.

Leek, Jeff. 2013. Data Analysis. Videos from the course. http://blog.revolutionanalytics.com/2013/04/coursera-data-analysis-course-videos.html.

Lewis, Randall A., and Justin M. Rao. 2013. "On the Near Impossibility of Measuring the Returns to Advertising." Unpublished paper, Google, Inc. and Microsoft Research. http://justinmrao.com/lewis_rao_nearimpossibility.pdf.

Ley, Eduardo, and Mark F. J. Steel. 2009. "On the Effect of Prior Assumptions in Bayesian Model Averaging with Applications to Growth Regression." *Journal of Applied Econometrics* 24(4): 651–74.

McLaren, Nick, and Rachana Shanbhoge. 2011. "Using Internet Search Data as Economic Indicators." *Bank of England Quarterly Bulletin* 51(2): 134–40.

Morgan, James N., and John A. Sonquist. 1963. "Problems in the Analysis of Survey Data, and a Proposal." *Journal of the American Statistical Association* 58(302): 415–34.

Munnell, Alicia H., Geoffrey M. B. Tootell, Lynne E. Browne, and James McEneaney. 1996. "Mortgage Lending in Boston: Interpreting HMDA Data." *American Economic Review* 86(1): 25–53.

Murphy, Kevin P. 2012. *Machine Learning: A Probabilistic Perspective*. MIT Press.

Pearl, Judea. 2009a. *Causality: Models, Reasoning, and Inference*, 2nd edition. Cambridge University Press.

Pearl, Judea. 2009b. "Causal Inference in Statistics: An Overview." *Statistics Surveys* 3: 96–146.

Perlich, Claudia, Foster Provost, and Jeffrey S. Simonoff. 2003. "Tree Induction vs. Logistic Regression: A Learning-Curve Analysis." *Journal of Machine Learning Research* 4: 211–55.

Rubin, Donald B. 1974. "Estimating Causal Effects of Treatments in Randomized and Nonrandomized Studies." *Journal of Educational Psychology* 66(5): 689–701.

Sala-i-Martín, Xavier. 1997. "I Just Ran Two Million Regressions." *American Economic Review* 87(2): 178–83.

Scott, Steve, and Hal Varian. 2013a. "Bayesian Variable Selection for Nowcasting Economic Time Series." NBER Working Paper 19567.

Scott, Steve, and Hal Varian. 2013b. "Predicting the Present with Bayesian Structural Time Series." NBER Working Paper 19567.

Steel, Mark F. J. 2011. "Bayesian Model Averaging and Forecasting." *Bulletin of E.U. and U.S. Inflation and Macroeconomic Analysis*, 200: 30–41.

Stephens, Revor, and Curt Wehrley. 2014. "Getting Started with R." *Kaggle*, September 28. https://www.kaggle.com/c/titanic-gettingStarted/details/new-getting-started-with-r.

Sullivan, Danny. 2012. "Google: 100 Billion Searches per Month, Search to Integrate Gmail, Launching Enhanced Search App for iOS." Search Engine Land. http://searchengineland.com/google-search-press-129925.

Venables, W. N., and B. D. Ripley. 2002. *Modern Applied Statistics with S,* 4th edition. New York: Springer.

Williams, Graham. 2011. *Data Mining with Rattle and R.* New York: Springer.

Wu, Xindong, and Vipin Kumar, eds. 2009. *The Top Ten Algorithms in Data Mining.* CRC Press.