Essays on Machine Learning and Hedonic Models

Miaoyu Yang

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2016

Reading Committee:

Patrick Bajari, Chair

Gregory Duncan

Fahad Khalil

Program Authorized to Offer Degree:
Department of Economics

University of Washington

Abstract

Essays on Machine Learning and Hedonic Models

Miaoyu Yang

Chair of the Supervisory Committee:
Professor Patrick Bajari
Economics

Chapter 1 and 2: We survey and apply several techniques from the statistical and computer science literature to the problem of demand estimation. We derive novel asymptotic properties for several of these models. To improve out-of-sample prediction accuracy and obtain parametric rates of convergence, we propose a method of combining the underlying models via linear regression. We illustrate our method using a standard scanner panel data set to estimate promotional lift and find that our estimates are considerably more accurate in out-of-sample predictions of demand than some commonly-used alternatives. While demand estimation is our motivating application, these methods are widely applicable to other microeconometric problems.

Chapter 3: We collect high dimensional data and extract features from house descriptions and images to use as controls within a hedonic model to estimate the impact of fracking on house prices in Pennsylvania. Supplementing a structured dataset with high dimensional unstructured data in the form of descriptive words and images of homes can help to close the gap caused by omitted variable bias. We construct curb appeal scores based on aesthetic features of home images. We then compare four models: OLS, LASSO - OLS, random forest and gradient boosting. The ensemble tree models (random forest and gradient boosting) yield 10% improvements in prediction accuracy compared to LASSO and OLS. Our results imply that royalty payments exactly compensate for the negative environmental effects on

homes within 1 km of fracking wells but increase the price of houses farther away by up to 5%.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I want to thank my advisor Pat Bajari for his knowledgeable guidance and candid career advices. I thank my co-authers Pat Bajari, Stephen Ryan, Denis Nekipelov and Jenny Ho for their hard work and numerous discussions leading to these papers. I thank the rest of my committee, Greg Duncan, Fahad Khalil and Judith Thornton for their helpful guidance on my work. I appreciate Graduate School Representatives Gunther Uhlmann and Rebecca Aanerud for taking part in my exams. I also want to acknowledge my colleagues Konstantin Golyaev and Andrew Clayton for comments and discussions.

I owe thanks to Fahad Khalil for mentoring in the first three years of my graduate program, and to my undergraduate mentors Yan Zhang, Yuan Zhang and Changyuan Luo for inspiring me to pursue economics research.

Lastly I want to mention my friends and family for their emotional support. My parents who support me unconditionally are the main reason I survived graduate school. My husband Jie Chen took great responsibility to care for me and the family, and he also helped editing the dissertation. My friends Chunxiao Xue, Hui Mai and Xuyang Ma, supported me and each other to strive through the PhD program.

DEDICATION

To my parents

Songguo Yang and *Luxian Zhou*

my husband

Jie Chen

and my daughter

Arya Chen

Chapter 1

MACHINE LEARNING METHODS FOR DEMAND ESTIMATION

By Patrick Bajari, Denis Nekipelov, Stephen Ryan, and Miaoyu Yang [1]

Over the past decade, there has been a high level of interest in modeling consumer behavior in the fields of computer science and statistics. These applications are motivated in part by the availability of large data sets, and are commonly used by firms in the retail, health care, and internet industries to improve business decisions. In this paper, we compare these methods to standard econometric models that are used by practitioners to study demand.[2] We are motivated by the problem of finding practical tools that would be of use to applied econometricians in estimating demand with large numbers of observations and covariates, such as in a scanner panel data set.

Many economists are unfamiliar with these methods, so briefly sketch several commonly-used techniques from the machine learning literature.[3] We consider eight different models that can be used for estimating demand: linear regression, the conditional logit, and six machine learning methods, all of which differ from standard approaches by combining an element of model selection into the estimation procedure. Several of these models can be seen as variants on regularization schemes, which reduce the number of covariates in a regression which receive non-zero coefficients, such as stepwise regression, forward stagewise regression, LASSO, and support vector machines. We also consider two models based on regression trees, which are flexible methods for approximating arbitrary functions: bagging and random

[1] *American Economic Review: Papers & Proceedings 2015, 105(5):481-485*

[2] Hastie et al. [2009] is a comprehensive reference of these and related machine learning methods.

[3] These methods are beginning to diffuse in econometrics. See, for example, Belloni et al. [2014] and Varian [2014].

forests. While these models may be unfamiliar to many economists, they are surprisingly simple and are based on underlying methods that will be quite familiar. Also, all of the methods that we use are supported in statistical packages. We perform our computations in the open source software package R. Therefore, application of these methods will not require writing complex code from scratch. However, applied econometricians may have to familiarize themselves with alternative software.

We apply our method to a canonical demand estimation problem. We use data from IRI Marketing Research Bronnenberg et al. [2008] via an academic license at the University of Chicago. It contains scanner panel data from grocery stores within one grocery store chain for six years. We used sales data on salty snacks, which is one of the categories provided in the IRI data. The number of observations are 837,460, which includes 3,149 unique products.

If we allow for product and store level fixed effects, our model effectively has many thousands of explanatory variables. Therefore, variable selection will be an important problem. If we included all of these variables in a standard regression model, the parameters would be poorly estimated. Also, many of the regressors will be multi-collinear which will make the models predict poorly out of sample.

In our results, we find that the six models we use from the statistics and computer science literature predict demand out of sample in standard metrics much more accurately than a panel data or logistic model. We do not claim that these models dominate all methods proposed in the voluminous demand estimation literature. Rather, we claim that as compared to common methods an applied econometrician might use in off the shelf statistical software, these methods are considerably more accurate. Also, the methods that we propose are all available in the well documented, open software package R as well as commercially-available software.

Finally, we propose using an idea dating back at least to Bates and Granger [1969]. We treat each of these eight independent predictions as regressors and form a combined model by regressing the dependent variable on to the prediction of each component model. We use a three-way cross validation to avoid overfitting the models in practice. We split the sample

into three disjoint sets; we use the first set to fit all eight models, we use the second set to fit our regression on the eight independent model predictions, and we use the third set of the data to test the fit out of sample. We find that this combination procedure can lead to substantial improvements in fit with little additional work. And, as we detail in Bajari et al. [2015], the combined model exhibits standard asymptotic behavior, even though the component models may not, which simplifies the construction of standard errors.

1.1 Summary of Machine Learning Methods

We explore using machine learning techniques to predict demand. We briefly discuss each method in turn before applying them to estimation of demand using a scanner panel data set.

A typical specification for demand of product j in group h in market m at time t would be:

$$Y_{jhmt} = f(\mathbf{X}, \mathbf{D}, \mathbf{p})'\beta + \zeta_{hm} + \eta_{mt} + \epsilon_{jmt}, \tag{1.1}$$

where f generates arbitrary interactions between the observables ($\mathbf{X}$), demographics ($\mathbf{D}$), and prices ($\mathbf{p}$). Such a model may have thousands of right-hand side variables; an extreme example from Rajaraman and Ullman [2011] notes Google estimates the demand for a given webpage by using a model of the network structure of literally billions of other webpages on the right-hand side. Dummy variables on nests are captured by ζ_{hm}. Seasonality is captured by the term η_{mt}, which varies by time (say, quarters) across markets.

In ordinary least squares (OLS), the parameters of Equation 1.1 are typically estimated using the closed-form formula $\beta = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{Y})$. This formula requires an inversion of $(\mathbf{X}'\mathbf{X})$, imposing a rank and order condition on the matrix $\mathbf{X}$. We highlight this because in many settings, the number of right-hand side variables can easily exceed the number of observations. Even in the simplest univariate model, one can saturate the right-hand side by using a series of basis functions of $\mathbf{X}$. This restriction requires the econometrician to make choices about which variables to include in the regression. We will return to this below, as

some of machine learning methods we discuss below allow the econometrician to skirt the order condition by combining model selection and estimation simultaneously.

A large literature on differentiated products has focused on logit-type models, where the idiosyncratic error term is assumed to be distributed as a Type I Extreme Value. Under that restriction, market shares are given by:

$$s_{jhmt} = \frac{\exp(\theta' \mathbf{X}_{jhmt})}{\sum_{k \in J} \exp(\theta' \mathbf{X}_{khmt})}. \tag{1.2}$$

Quantities are then computed by multiplying through by market size.[4]

Stepwise regression starts with the intercept as the base model on a set of demeaned covariates. The algorithm then searches over the set of covariates, selects the one with the highest correlation with the residual, and adds that variable to the next model. The method then estimates OLS using that subset of covariates, then repeats the search for the covariate with the next highest correlation. The procedure produces a series of nested models and runs until no covariates have a sufficiently high correlation with the error term.

Forward stagewise regression is a variant on the stepwise regression. Whereas all of the coefficients can change at each step in the stepwise regression, forward stagewise regression only updates one coefficient at each step. The method finds the variable with the highest correlation with the error term and adds that covariance to the coefficient. This continues until none of the covariates have any correlation with the error term.

These methods build up the model over time in addition to estimating a fit. One advantage of this approach is that the methods can recover the true data-generating process when the number of covariates is larger than the number of observations and the true model is sparse, e.g. the number of coefficients with true non-zero values is less than the number of observations.

---

[4]Berry et al. [1995] extends this model to deal with unobserved heterogeneity and vertical characteristics that are observed to both the firm and consumers.

Support vector machines (SVM) are a penalized method of regression, using the following:

$$\min_{\beta} \sum_{i=1}^{n} V(y_i - \mathbf{X}_i'\beta) + \frac{\lambda}{2}\|\beta\|, \tag{1.3}$$

where the penalty function is:

$$V_\epsilon(r) = \begin{cases} 0 & \text{if } |r| < \epsilon, \\ |r| - \epsilon & \text{otherwise.} \end{cases} \tag{1.4}$$

The tuning parameter, ϵ, controls which errors are included in the regression. Errors of sufficiently small size are treated as zeros. Typically only a partial set of the covariates are assigned a non-zero value in SVM regression.

LASSO is another penalized regression method. The regression is given by:

$$\min_{\beta} \frac{1}{2} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda(t - \sum_{j=1}^{p} |\beta_j|), \tag{1.5}$$

where t is the tuning parameter governing how strictly additional regressors are penalized. LASSO typically results in a number of covariates being given zero weights.

Regression trees approximate functions by partitioning the characteristic space into a series of hyper-cubes and reporting the average value of the function in each of those partition. Regression trees generalize fixed effects to allow them to depend on values of $\mathbf{X}$. In the limit as the hypercubes grow infinitesimally small, the tree reports the average value $Y = f(\mathbf{X} = \mathbf{x})$, which is a perfect reconstruction of the underlying function f. In practice, the tree is expanded until the reduction in squared prediction error falls under some threshold. Often, the tree is grown until a specific number of splits are achieved.

The literature has proposed several variations on the regression tree estimator. One is bagging Breiman [1996], which uses resampling and model combination to obtain a predictor. The idea is to sample the data with replacement B times, train a regression tree on each

resampled set of data, and then predict the outcome at each x through a simple average of the prediction under each of the B trees.

A second approach, which we have found to work exceptionally well in practice, are random forests, as in Breiman [2001]. Random forests expand on the idea of using collections of predictors to make predictions by introducing randomness into the set of variables which are considered at node level for splitting. Before each split, only $m \leq p$ of the explanatory variables are included in the split search. Repeating this across B trees results in a forest of random trees. The regression predictor for the true function is then:

$$\hat{f}_{rf}^{B}(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} T_b(\mathbf{x}). \tag{1.6}$$

Trees of sufficient size can be unbiased but exhibit high variance, and therefore may benefit from averaging.

1.2 Empirical Application

This section compares econometric models with machine learning ones using a typical demand estimation scenario—grocery store sales. We find that the machine learning models in general produce better out-of-sample fits than linear models without loss of in-sample goodness of fit. If we combine all the models linearly with non-negative weights, the resulting combination of models produces better out-of-sample fit than any model in the combination.

The data we use is provided by IRI Marketing Research via an academic license at the University of Chicago. It contains scanner panel data from grocery stores within one grocery store chain for six years. We used sales data on salty snacks, which is one of the categories in the IRI data. A unit of observation is product j, uniquely defined by a UPC (Universal Product Code), in store m at week t. The number of observations are 1,510,563, which includes 3,149 unique products. Let q_{jmt} be the number of bags of salty snack j sold in store m at week t. If $q_{jmt} = 0$, we do not know if it is due to no sale or out-of-stock and the observation is not filled in. The price p_{jmt} is defined as the quantity weighted average

of prices for product j in store m at week t. Therefore if $q_{jmt} = 0$, the weight is also 0. In addition to price and quantity, the data contains attributes of the products (such as brand, volume, flavor, cut type, cooking method, package size, fat and salt levels) and promotional variables (promotion, display and feature).

The response variable is log of quantity sold per week. The covariates are log of price, product attributes variables, promotional variables, store fixed effects, and week fixed effects. We provide the same covariate matrix to all of the models expect for the logit model, where all the fixed effects are excluded.[5]

In order to estimate and compare models, we split our data into three sets: training, validation, and holdout. We estimate the model using the training set, and then use the validation set to assign weights to each model when building the combined model. This approach mitigates overfitting in the training model; for example, the linear model tends to get a good in-sample fit but a bad out-of-sample fit, and granting these model a large in-sample weight would produce poor predictions in the holdout sample. Finally, we each model to predict fit in a holdout sample. 25 percent of the data is used as the holdout sample, 15 percent is used as the validate set, and the remaining 60 percent is used as the training set.

Table 1.1 reports the root mean squared prediction error (RMSE) across the validation and out-of-sample data sets, along with the estimated weights of each model in the combined model. In the scenario of out-of-sample prediction error, the best two models are random forest and support vector machine. The combined model, where we regress the actual value of the response variable on a constrained linear model of the predictions from eight models, outperforms all the eight models, which follows the optimal combination of forecasts in Bates and Granger [1969]. Random forest receives the largest weight in the combined model (65.6 percent), and the stepwise and SVM models receive the majority of the rest. It is interesting

[5]For brevity, we have minimized the description of the data set and details of implementation of the machine learning methods; Bajari et al. [2015] provides more details about the construction of the data set.

to observe the combined model does not simply choose the submodel with the best RMSE; there are important covariances among the models which generate better fit in combination than any one given submodel.

Table 1.1: Model Comparison: Prediction Error

	Validation		Out-of-Sample		Percent
	RMSE	Std. Err.	RMSE	Std. Err.	Weight
Linear	1.169	0.022	1.193	0.020	6.62
Stepwise	0.983	0.012	1.004	0.011	12.13
Forward Stagewise	0.988	0.013	1.003	0.012	0.00
LASSO	1.178	0.017	1.222	0.012	0.00
Random Forest	0.943	0.017	0.965	0.015	65.56
SVM	1.046	0.024	1.068	0.018	15.69
Bagging	1.355	0.030	1.321	0.025	0.00
Logit	1.190	0.020	1.234	0.018	0.00
Combined	0.924		0.946		100.00
Number of Obs.	226,952		376,980		
Total Obs.	1,510,563				
Percent of Total	15.0		25.0		

1.3 Conclusion

In this paper, we review and apply several popular methods from the machine learning literature to the problem of demand estimation. Machine learning models bridge the gap between parametric models with user-selected covariates and completely non-parametric approaches. We demonstrate that these methods can produce superior predictive accuracy as compared to a standard linear regression or logit model. We also show that a linear combination of the underlying models can improve fit even further with very little additional work. While these methods are not yet commonly used in economics, we think that practitioners will find value in the flexibility, ease-of-use, and scalability of these methods to a wide variety of applied settings.

One concern has been the relative paucity of econometric theory for machine learning

models. In related work Bajari et al. [2015], we provide asymptotic theory results for rates of convergence of the underlying machine learning models. We show that while several of the machine learning models have non-standard asymptotics, with slower-than-parametric rates of convergence, the model formed by combining estimates retains standard asymptotic properties. This simplifies the construction of standard errors for both parameters and predictions, making the methods surveyed here even more accessible for the applied practitioner.

Chapter 2

DEMAND ESTIMATION WITH MACHINE LEARNING AND MODEL COMBINATION

By Patrick Bajari, Denis Nekipelov, Stephen Ryan, and Miaoyu Yang [1]

Abstract

We survey and apply several techniques from the statistical and computer science literature to the problem of demand estimation. We derive novel asymptotic properties for several of these models. To improve out-of-sample prediction accuracy and obtain parametric rates of convergence, we propose a method of combining the underlying models via linear regression. We illustrate our method using a standard scanner panel data set to estimate promotional lift and find that our estimates are considerably more accurate in out-of-sample predictions of demand than some commonly-used alternatives. While demand estimation is our motivating application, these methods are widely applicable to other microeconometric problems.

2.1 Introduction

Over the past decade, there has been a high level of interest in modeling consumer behavior in the fields of computer science and statistics. These applications are motivated in part by the availability of large data sets where the demand for stock keeping unit (SKU) or individual consumers can be observed. These methods are commonly used in industry in retail, health care or on the internet by firms to use data at large scale to make more rational business decisions. In this paper, we compare these methods to standard econometric models that are used by practitioners to study demand. We are motivated by the problem of finding

[1] *NBER Working Paper No. 20955*

practical tools that would be of use to applied econometricians in estimating demand with large numbers of observations and covariates, such as in a scanner panel data set.

Many economists are unfamiliar with these methods, so we begin by expositing some commonly used techniques from the machine learning literature. We consider 8 different models that can be used for estimating demand for an SKU. The first two models are well-known to applied econometricians—the conditional logit and a panel data regression model. We then turn to machine learning methods, all of which differ from standard approaches by combining an element of model selection into the estimation procedure. Several of these models can be seen as variants on regularization schemes, which reduce the number of covariates in a regression which receive non-zero coefficients, such as stepwise regression, forward stagewise regression, LASSO, and support vector machines. We also consider two models based on regression trees, which are flexible methods for approximating arbitrary functions: bagging and random forests. While these models may be unfamiliar to many economists, they are surprisingly simple and are based on underlying methods that will be quite familiar. Also, all of the methods that we use are supported in statistical packages. We perform our computations in the open source software package R. Therefore, application of these methods will not require writing complex code from scratch. However, applied econometricians may have to familiarize themselves with alternative software.

We derive novel results for the asymptotic theory for several of the models above. We show, somewhat unsurprisingly, that many of these models do not have standard asymptotics and converge more slowly than the standard square root rate. Since these models do not have standard normal asymptotics, common methods such as the bootstrap cannot be applied for inference. We also propose using an idea dating back at least to Bates and Granger [1969]. In a first step, we estimate all of the models on a training data set. In a second step, we then estimate a linear regression, where the regressors are predictions from each submodel. This process has two benefits: first, the linear combination has better predictive accuracy than any of its component models; and second, the linear combination also exhibits parametric rates of convergence under weak conditions.

To illustrate the usefulness of our approach, we apply our method to a canonical demand estimation problem. We use the Information Resources Inc, Inc. Marketing data set, detailed in Bronnenberg et al. [2008], to carry out the estimation. All estimates in this paper based on IRI data are by the authors and not by Information Resources Inc, Inc.. It contains scanner panel data from grocery stores within one grocery store chain for six years. We used sales data on salty snacks, which is one of the categories provided in the IRI data. We find that the 6 models from the machine learning literature predict demand out of sample in standard metrics much more accurately than a panel data or logistic model. The combined model does even better, increasing predictive fit by two percent against the best single model. We do not claim that these models dominate all methods proposed in the voluminous demand estimation literature. Rather, we claim that as compared to common methods an applied econometrician might use in off the shelf statistical software, these methods are considerably more accurate. Also, the methods that we propose are all available in the well documented, open software package R as well as commercially-available software.

Applied econometricians have sometimes voiced skepticism about machine learning models because they do not have a clear interpretation and it is not obvious how to apply them to estimate causal effects. In this paper, we use an idea proposed by Varian [2014] to estimate the marketing lift attributable to promotions in our scanner panel. The idea is similar to the idea of synthetic controls used in Abadie and Gardeazabal [2003]. We begin by training our model on the data where there is no promotion. We then hold the parameters of our model fixed and predict for the observations where there is a promotion. We then take the difference between the observed demand and the predicted demand for every observation in our data. By averaging over all observations in our sample, we construct an estimate of the average treatment effect on the treated.

We believe that this approach might be preferable to instrumental variable methods. In practice, it can be difficult to find instruments that are a priori plausible. When they exist, they may be subject to standard critiques such as the instruments may be weak or the identification may only be local. The logic behind our approach is simply to use lots of

data rather than rely on quasi-randomness. As mentioned above, in applied econometrics, there are many forms of data about products that are simply not exploited in empirical studies such as unstructured text or pictures. In some applications, simply using more data and more scalable computations may be a superior strategy to reducing bias in causal lift estimates.

We find quite interesting that a standard panel data model with fixed effects has the "wrong sign" on promotional lift, i.e. promotions decrease demand. By contrast, our models from the statistics and computer science literature have the anticipated sign. We conjecture that this is because they simply use more data and have less bias as suggested by standard omitted variable formulas.

Finally, we can use our model to search for heterogeneity in the treatment effects in an unstructured way. Our model generates a residual for each observation that is treated. We can regress this residual on covariates of interest such as store indicators, brand dummies, hedonic attributes or seasonal factors. Once again, this is a high dimensional regression problem and the estimates would be poorly estimated in a regression framework. We instead propose a method suggested by Belloni et al. [2012] and use a LASSO to select variable and then use standard methods for inference. We believe that this is attractive for applied econometricians since it allows us to learn about heterogeneity in the treatment effect. Also, it could be useful to applied marketers since these variable could be useful in marketing mix models because it allows us to identify a smaller set of variables that predict marketing return.

The paper is organized as follows: Section 2.2 introduces the underlying statistical model of demand we study; Section 2.3 discusses the rates of convergence of our estimators; Section 2.4 covers model combination; Section 2.5 discusses four additional machine learning models that we use in our application; Section 2.6 applies the techniques to a scanner data set; and Section 2.7 concludes.

2.2 Model

In this section we will base our analysis on the idea that the point of inference of individual models (that will be further used in averaging) is to estimate the conditional expectation $\theta(z) = E[Y \mid Z = z]$. In other words, the concrete machine learning methods will be considered new versions of nonparametric methods whose main goal is *prediction*. When we use the traditional L_2 norm the task of prediction reduces to finding the function that best approximates the conditional expectation of the outcome variable of interest. To fix ideas, we consider a model with a scalar outcome variable Y, a vector of inputs Z and a random disturbance ε, such that for each of n observations i the model for the data generating process is

$$y_i = f(z_i) + \epsilon_i,$$

where $E[\epsilon_i] = 0$ and $E[\epsilon_i^2] = \sigma^2 < \infty$. Our goal will be the inference for $f(\cdot)$.

Before turning to a discussion of the machine learning techniques we will use in this paper, we first consider two common empirical approaches to estimating conditional expectations. To fix ideas, suppose our goal is to estimate the following model of demand. Let there be J products, each endowed with observable characteristics X_j. Let product j have demand in market m at time t equal to:

$$\ln Q_{jhmt} = f(p_{mt}, a_{mt}, X_{mt}, D_{mt}, \epsilon_{jmt}; \theta), \tag{2.1}$$

where a is a matrix of advertising and promotional measures, D is a vector of demographics, p is a vector of prices, ϵ is an idiosyncratic shock, and θ is a vector of unknown parameters. The above specification is very general. It allows for nesting through the stratification of the error term. Suppose that there are H nests of products; continuing the automobile example, two nests might be entry-level sub-compacts (Ford Fiesta, Toyota Yaris, Mazda 2, Chevrolet Sonic) and luxury performance sedans (BMW M3, Mercedes-Benz AMG C63, Cadillac CTS-V). Nests allow the substitution patterns to vary in a reduced-form way across those different

classes of products. One can also extend the model to allow for non-trivial intertemporal shocks, such as seasonality in demand due to environmental conditions or holidays. The specification in Equation 2.1 is also consistent with models of discrete choice. For example, if the choice problem is discrete, then one obtains quantities by integrating over both the population of consumers and the distribution of errors.

The goal of our exercise is to estimate the relationships between the right-hand side variables and quantities demanded. We discuss several approaches to approximating the model in Equation 2.1. We first consider more familiar approaches like linear regression and logit models before describing several machine learning approaches which are less known in the econometrics literature. Here, we explain these models in enough detail to provide some intuition.

2.2.1 Linear Regression

A typical approach to estimating demand would be to approximate Equation 2.1 using a flexible functional form of the following type:

$$\ln Q_{jhmt} = \alpha' p_{mt} + \beta_1' X_{mt} + \beta_2' D_{mt} + \gamma' a_{mt} + \lambda' \mathcal{I}(X_{mt}, D_{mt}, p_{mt}, a_{mt}) + \zeta_{hm} + \eta_{mt} + \epsilon_{jmt}, \quad (2.2)$$

where $\mathcal{I}$ is an operator which generates interactions between the observables (e.g. interactions of X, p, and D, e.g. high income neighborhoods have higher demand for expensive imported beer). Dummy variables on nests are captured by ζ_{hm}, while seasonality is captured by the term η_{mt}, which varies by time across markets. In principle, such a model may have thousands of right-hand side variables; for example, an online retailer such as eBay may offer hundreds of competing products in a category, such as men's dress shirts. The demand for one particular good, say white Brooks Brothers dress shirts, may depend on the prices of the full set of competing products offered. In a more extreme example, as offered in Rajaraman and Ullman [2011], Google estimates the demand for a given webpage by using a model of the network structure of literally billions of other webpages on the right-hand side. In practice,

however, such models usually only consider a very small subset of all possible right-hand side variables, and those variables are typically chosen in an ad hoc manner by the researcher.

In ordinary least squares (OLS), the parameters of Equation 2.2, jointly denoted by β, are typically estimated using the closed-form formula:

$$\beta = (X'X)^{-1}(X'Y), \tag{2.3}$$

where X is the matrix of right-hand side variables and Y is the vector of outcomes.

We note that the formula requires an inversion of $(X'X)$. This imposes a rank and order condition on the matrix X. We highlight this because in many settings, the number of right-hand side variables can easily exceed the number of observations. Even in the simplest univariate model, one can saturate the right-hand side by using a series of basis functions of X. This restriction requires the econometrician to make choices about which variables to include in the regression. We will return to this below, as some of machine learning methods we discuss below allow the econometrician to skirt the order condition by combining model selection and estimation simultaneously.

2.2.2 Logit Models

A large literature on differentiated products has focused on providing a theoretical foundation for Equation 2.1 based on an underlying random utility model of discrete choice. This structure gives rise to predictions of choice probabilities that can be used to compute market shares when integrated over the market population; quantities are then computed by multiplying market shares with market size.

A typical discrete choice model is the logit model, where utilities are modeled as functions similar to the right-hand side of Equation 2.1 and the idiosyncratic error is a Type I Extreme Value. This gives rise to a particularly nice analytical form for the market share:

$$s_j = \frac{\exp(\theta' X_{jhmt})}{\sum_{k \in J} \exp(\theta' X_{khmt})}. \tag{2.4}$$

Since Berry et al. [1995], many empirical models of differentiated product demand concentrate on estimating a distribution of unobserved heterogeneity, $F(\theta)$, over a typically low-dimensional θ:

$$s_{jhmt} = \int \frac{\exp(\theta' X_{jhmt})}{\sum_{k \in J} \exp(\theta' X_{khmt})} dF(\theta). \tag{2.5}$$

An attractive feature of this approach is that the method is robust to the inclusion of unobserved heterogeneity and vertical characteristics that are observed to both the firm and consumers. However, this estimator places a significant amount of structure on the demand curve and is computationally burdensome to compute when the degree of heterogeneity is very flexible.

Next, we turn to discussing the machine learning methods that are used in this paper. We begin with a general overview of the motivation and implementation of each method in Section 2.3.1. Econometric details begin in Section **??**; the applied reader may wish to skip over this section.

2.3 Implementation and Properties of Selected Machine Learning Methods

2.3.1 Review of Machine Learning Methods

In this section we will base our analysis on the idea that the point of inference of individual models (that will be further used in averaging) is to estimate the conditional expectation $\theta(z) = E[Y \mid Z = z]$. In other words, the concrete machine learning methods will be considered new versions of nonparametric methods whose main goal is *prediction*. When we use the traditional L_2 norm the task of prediction reduces to finding the function that best approximates the conditional expectation of the outcome variable of interest. To fix ideas, we consider a model with a scalar outcome variable Y, a vector of inputs Z and a random disturbance ε, such that for each of n observations i the model for the data generating process is

$$y_i = f(z_i) + \epsilon_i,$$

where $E[\epsilon_i] = 0$ and $E[\epsilon_i^2] = \sigma^2 < \infty$. Our goal will be the inference for $f(\cdot)$.

2.3.2 Square-root LASSO

The general class of LASSO methods is designed to perform the computationally efficient and consistent approach to model selection and inference based on penalization. The idea of such a penalization if the following. Suppose that $X \in \mathcal{X}$ is a vector of the p-dimensional space (which is a nonlinear transformation of the space spanned by Z, possibly enlarging the dimension) where function $f(\cdot)$ can be well approximated by some linear function of X. The estimators are considered for the "self-normalized" regressors, such that $\frac{1}{n}\sum_{i=1}^n x_{ij}^2 = 1$ for each j. Under appropriate smoothness of $f(\cdot)$ and the appropriately chosen $\mathcal{X}$, the residual from the best linear approximation using s elements of $\mathcal{X}$, denoted $r_i = f(z_i) - x_i'\beta_0$ will be small as measured by the sample sum of squares $c_s^2 = \frac{1}{n}\sum_{i=1}^n r_i^2$. For instance, if $f(\cdot)$ is a smooth function and $\mathcal{X}$ is the space of polynomials of $\mathcal{Z}$ then the corresponding sum of squares for the approximation will be determined by order of the residual in the corresponding Taylor expansion of $f(\cdot)$. Provided that the goal of interest is the minimization of the prediction error **?** introduce the norm for estimators $\widehat{\beta}$ of β_0 such that

$$\|\widehat{\beta} - \beta_0\|_{2,n} = \sqrt{\frac{1}{n}\sum_{i=1}^n (x_i'\widehat{\beta} - x_i'\beta_0)^2}.$$

Note that if $f(\cdot)$ belongs to the linear span of $\mathcal{X}$, then norm $\|\cdot\|_{2,n}$ measures the prediction error for $f(\cdot)$. Otherwise, the error will also reflect the approximation error c_s.

The procedure for finding the tradeoff between the approximation error and estimation error is by choosing the appropriate penalization for the sample least squares objective

$$\widehat{Q}(\beta) = \frac{1}{n}\sum_{i=1}^n (y_i - x_i'\beta)^2.$$

Then the sample version of the problem of finding the parsimonious (sparse) linear function

can be written as

$$\min_{\beta \in \mathbb{R}^p} \widehat{Q}(\beta) + \frac{\sigma^2 \|\beta\|_0}{n}, \tag{2.6}$$

where $\|\beta\|_0 = \sum_{k=1}^{p} \mathbf{1}\{\beta_k \neq 0\}$ is the ℓ_0 norm counting the number of non-zero components of β. This estimator yields the minimum risk over sparse linear representations of $f(\cdot)$ when the errors are homoskedastic and Gaussian. In this respect this estimator parallels the standard OLS estimator in terms of efficiency. However, the empirical objective function that needs to be minimized is discontinuous with respect to the minimand. Morover it is not convex meaning that known optimization procedures will converge slowly. In fact, the computation problem corresponding to this minimization is known to be NP-hard.

The idea in the Machine Learning literature that allows one to transform a computationally hard problem (2.6) into a computationally efficient problem that also yields the solution that approximates the solution of (2.6) well is the following. Since the biggest problem in the objective of (2.6) is the presence of the ℓ_0 penalty, it gets replaced with the normalized ℓ_1 penalty $\lambda \|\beta\|_1$, where $\|\beta\|_1 = \sum_{k=1}^{p} |\beta|_1$. That instantly leads to a convex objective function that can be minimized efficiently. There are three concerns with this approach. First, the penalty constant λ is not known a priori and needs to be estimated as well. Second, in the presense of heteroskedastic errors the choice of unweighted norm $\|\beta\|_1$ does not lead to consistent estimates for parameters. Third, the presense of ℓ_1 penalty leads to the small sample bias in the estimated parameters. **?** address all three problems in the following way. First of all, they suggest to replace the objective function $\widehat{Q}(\beta)$ with $\sqrt{\widehat{Q}(\beta)}$. In this case the gradient of $\sqrt{\widehat{Q}(\beta)}$ with respect to β becomes "self-normalized". As a result, the optimal penalty λ no longer depends on σ and is only the function of dimensionality of $\mathcal{X}$ and the sample size. Second, they suggest a linear transformation of β in the ℓ_1 penalty that leads to consistency of the estimator. Finally, they suggest to use the penalized minimization procedure as the tool for selecting the dimensions of X that will be used the approximation and then run the post-estimation procedure that minimizes $\widehat{Q}(\cdot)$ only with respect to the selected coefficients.

Formally, the minimization problem can be characterized as

$$\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} (y_i - x_i'\beta)^2 + \lambda \|\widehat{\Psi}\beta\|_1, \quad \lambda = 2\sqrt{2\log(pn)/n} \tag{2.7}$$

The optimal choice for the penalty constant in this case is $\lambda = 2\sqrt{\frac{2\log(pn)}{n}}$, which means that no additional steps are required to obtain its value. The transformation matrix Ψ resembles the weights in the WLS procedure such that

$$\widehat{\Psi} = \text{diag}\left(\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_{ij}^2 \epsilon_i^2} + o_p(1),\ j = 1, \ldots, p\right).$$

Provided that the residuals ϵ_i are not immediately available, they are replaced with the estimates $\widehat{\epsilon}_i$ that are the outcome of the iterative procedure. The procedure initilizes at $\widehat{\epsilon}_i = y_i - \bar{y}$ leading to $\widehat{\Psi} = \text{diag}\left(\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_{ij}^2 \widehat{\epsilon}_i^2} + o_p(1),\ j = 1, \ldots, p\right)$. Then the we solve (2.7) using $\widehat{\Psi}$ as the transformation matrix yielding $\hat{\beta}$. Then at the next step we use the updated errors $\widehat{\epsilon}_i = y_i - x_i'\hat{\beta}$ to construct the new weighting matrix $\widehat{\Psi}$. The procedure iterates till convergence. For the non-zero components of $\widehat{\beta}$ we then run the regular OLS to find the post-selection estimates.

2.3.3 SVM regression

Support vector machines (SVMs) are a class of estimators designed to solve the problem of prediction of a discrete outcome using a vector of regressors X. We can easily illustrate the idea behind the SVM approach for the settings where the outcome is determined by a linear index of regressors. In other words the statistical model takes the form

$$Y = f(X) + \zeta,$$

where $f(X) = \sum_{k=1}^{p} w_k X_k + b = \langle w, X\rangle + b$ and $|\zeta| \leq \epsilon$. The problem as in LASSO is to find a parsimonious set of relevant regressors in terms of their weights w. Allowing for the error in the magnitude of at most ϵ, we can formulate the mathematical problem for finding w from the sample $\{(y_i, x_i)\}_{i=1}^{n}$ as $\min \|w\|^2$ subject to

$$y_i - \langle w, x_i\rangle - b \leq \epsilon, \text{ and } \langle w, x_i\rangle + b - y_i \leq \epsilon.$$

Provided that for a given ϵ the linear approximation may not exist **?** suggest introdusing "slack" variables ξ and ξ^* that transform the original problem that may not have a solution into a feasible problem

$$\min \|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

subject to

$$y_i - \langle w, x_i\rangle - b \leq \epsilon + \xi_i, \text{ and } \langle w, x_i\rangle + b - y_i \leq \epsilon + \xi_i^*$$

for some constant $C > 0$ and non-negative slack variables ξ and ξ^*. The key idea in the SVM is to construct the Lagrangian for the introduced constrained optimization problem and replace the original optimization problem with the corresponding dual problem. Introducing Lagrange multipliers α, α^*, η and η^* we can write the Lagrangian as

$$L = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) + \sum_{i=1}^{n}(\eta_i\xi_i + \eta_i^*\xi_i^*) + \sum_{i=1}^{n}\alpha_i(\epsilon + \xi_i - y_i + \langle w, x_i\rangle + b)$$
$$+ \sum_{i=1}^{n}\alpha_i(\epsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b).$$

The problem dual to the minimization of the Lagrangian with respect to the state variables w, ξ, ξ^* is the problem of maximization with respect to the co-state variables (Lagrange multipliers) α, α^*, η, η^*. After simple re-arrangement this problem can be written as maxi-

mization of

$$-\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\langle x_i,\, x_j\rangle-\epsilon\sum_{i=1}^{n}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{n}y_i(\alpha_i-\alpha_i^*)$$

subject to $\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)=0$ and $0\leq\alpha_i,\,\alpha_i^*\leq C$. This is the problem that can be efficiently solved using quadratic programming. The observations i for which α_i or α_i^* are not equal to zero are called the support vectors. The main insight from this formulation is that now the estimated function of interest can be written as

$$\widehat{f}(x)=\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)\langle x_i,\, x\rangle+b.$$

This means that the impact of regressors x on the predicted outcome is summarized by the inner product $\langle\cdot,\cdot\rangle$. In linear case the inner product is the standard inner product in the Euclidean space.

Now suppose that X corresponds to some nonlinear transformation of a smaller set of regressors Z. For instance, X can be different order polynomials of Z. In that case, the inner product $\langle x_i,\, x\rangle$ will also summarize the mapping $Z\mapsto X$. This means that such an inner product can be explicitly defined as a function of the original inputs z called the kernel function. In this case the estimated regression function can be explicitly written in terms of the kernel function

$$\widehat{f}(z)=\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)k(z_i,z)+b.$$

For a given function of two arguments to be a kernel it needs to be shown that it can be represented as an inner product in some transformed space $\mathcal{X}$ (corresponding to $Z\mapsto X$). Several functions has been shown to be kernels and have been applied in practice. For instance, the Gaussian function has been shown to be a kernel:

$$k(z,z')=\exp\left(-\frac{\|z-z'\|^2}{2\sigma^2}\right).$$

Another example of the kernel is

$$k(z, z') = B_{2r+1}(\|z - z'\|),$$

where B_{2r+1} is the B-spline of order $2r + 1$. The difference between the linear case (where the function is estimated as a linear combination of inputs) and the case where the kernel function is used is that the weights w are no longer defined explicitly. To give more intuition, we can consider the case where the kernel can be written explicitly as $k(z, z') = \langle \Phi(z), \Phi(z') \rangle$. Then $\Phi(\cdot)$ is the mapping that creates "regressors" X from Z. This will also allow to write the weights as

$$\widehat{w} = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \Phi(x_i).$$

We note that unlike LASSO, the regression SVM approach allows the space $\mathcal{X}$ to be explicitly infinite-dimensional via the use of kernels.

2.3.4 L_2 Boosting

Boosting algorithms are computationally fast methods that have been shown to have similar properties to the LASSO methods in application to high-dimensional linear models. Although initial applications of boosting were for the binary outcome variable, L_2 boosting with the associated quadratic loss function applies to the models with continuous outcome variables. Unlike LASSO L_2 boosting imposes an implicit penalty that is incorporated in the structure of the algorithm itself. The algorithm is designed via the iterative verification of the fit of the model over all of the dimensions of regressors. The algorithm then advances along the dimension of the "best improvement", i.e. the dimension where a given regressor provides the biggest contribution to the least squares residual. The componentwise structure eliminates the need to fit a high-dimensional least squares objective function replacing the estimation with a series of one dimensional least squares procedures. Provided that at each step one needs to verify the fit associated with each regressor in the model, it will be neces-

sary to compute p least squares minima. Due to the structure of the least squares objective function the minimum can be written explicitly via the ratio of two sums further reducing the computational burden.

The algorithm proceeds in four steps. First, the algorithm initializes at step $m = 0$ at some $\widehat{f}^{(0)}(\cdot)$ with the default value $\widehat{f}^{(0)}(\cdot) = \bar{Y}$. Second, at step $m+1$ compute the residuals

$$\widehat{\epsilon}_i^{(m)} = y_i - \widehat{f}^{(m)}(x_i).$$

The model is then fitted using a componentwise linear least squares. (i) Compute the coefficients

$$\widehat{\gamma}_j^{(m)} = \frac{\sum_{i=1}^n x_{ij}\widehat{\epsilon}_i^{(m)}}{\sum_{i=1}^n (x_{ij})^2}.$$

(ii) Find the dimension to be updated $\widehat{j}^{(m)} = \arg\min_{j \le p} \sum_{i=1}^n \left(\widehat{\epsilon}_i^{(m)} - \widehat{\gamma}_j^{(m)} x_{ij}\right)^2$. (iii) Update the estimated coefficients as $\widehat{f}^{(m+1)}(x) = \widehat{f}^{(m)}(x) + \nu\, \widehat{\gamma}_{\widehat{j}^{(m)}}^{(m)}\, x_{\widehat{j}^{(m)}}$. The choice of the step-length factor ν does not impact the asymptotic behavior of the algorithm and is usually selected to be "sufficiently small" (e.g. $\nu = 0.1$). The componentwise linear least squares thus updates only one dimension of the model at a time. Finally, the second step is iterated until the stopping iteration M has been reached. The recommended choice is $M = K\left(\sqrt{n/\log(p)}\right)^{(3-\delta)/(4-2\delta)}$ for $0 < K < \infty$ and $0 < \delta < 5/8$.

2.4 Econometric properties of averaged models

Above we discussed three machine learning methods. We note that the main difference between the SVM regression, L_2 boosting and square-root LASSO is that the SVM regression does not explicitly use the "generated features" $x = \Phi(z)$. Instead, the (large) vector of generated features is embedded in the kernel function. Our goal is to ensure that the three discussed methods have adequate performance when evaluated in the average model. That would in general require that each of the estimators attains the convergence rate of $o(n^{-1/4})$. That would allow us to use existing results regarding the performance of average predictions

and establish asymptotic normality and $O(n^{-1/2})$ convergence rate for the corresponding average predictor.

There are three sets of conditions that need to be imposed on the econometric model to ensure the adequate performance of the resulting estimator. First, we need to ensure that the distributions of regressors and the unobserved disturbances a sufficiently regular so that the concentration inequalities leading to limit theorems can be applied. Second, we need to verify that the sequence of functional spaces that we use to approximate the nonparametric model provides sufficiently good approximation. The properties of approximating functional spaces used for semi- and nonparametric inference have been extensively studied in Econometrics, e.g. including Andrews [1991], Newey [1997] and Chen [2007a]. Finally, we need to impose the constraint on the complexity of the approximating model such that it appropriately trades off the approximation and the estimation errors.

We start with the imposition of properties on the statistical components of the model.

ASSUMPTION 1 *Suppose that the higher order maximum and minimum moments of regressors and disturbances satisfy the following conditions.*

(i) *For* $q > 4$ $\frac{1}{n}\sum_{i=1}^{n} E[|\epsilon_i|^q] < \infty$

(ii) $\inf_{n\geq 1} \min_{1\leq j\leq p} \frac{1}{n}\sum_{i=1}^{n} P^{pj}(z_i)^2 E[\epsilon_i^2] > 0$ *and* $\sup_{n\geq 1} \max_{1\leq j\leq p} \frac{1}{n}\sum_{i=1}^{n} |P^{pj}(z_i)|^3 E[|\epsilon_i|^3] > 0$*, where* $P^{pj}(\cdot)$ *are basis functions of the approximating space described in the next assumption.*

Next we impose a high-level condition on the approximating functional space and the corresponding estimator. The estimator itself will need to be appropriately selected, e.g. using the guidelines in Chen [2007a] and will depend on the underlying properties of the function that is approximated (such as smoothness, monotonicity, etc.).

ASSUMPTION 2 *Suppose that*

(i) $f(\cdot) \in \mathcal{F}$ *is an element of a compact subset of separable Hilbert space* $\mathcal{H}$ *and there is a sequence of finite-dimensional vector spaces* $\mathcal{H}^p$ *such that* $\mathcal{H}^p \subset \mathcal{H}^{p+1} \subset \mathcal{H}$ *and there is a sequence of orthogonal bases* $\{P^{pk}(\cdot)\}_{k=0}^{p}$ *such that* $\mathcal{H}^p$ *is a completion of the linear span of* $\{P^{pk}(\cdot)\}_{k=0}^{p}$.

(ii) For each n *and for any* $f \in \mathcal{F}$ *and any* $p \gg n$ *there exists a subsequence* $\{P^{pk_j}(\cdot)\}_{j=1}^{s}$ *with* $s \ll p$ *such that the class of functions* $\mathcal{R}^p = \left\{r^p(\cdot) = f(\cdot) - proj\left(f \mid \{P^{pk_j}(\cdot)\}_{j=1}^{s}\right),\ f \in \mathcal{F}\right\}$ *has a finite envelope* R^p *with*

$$E[(R^p)^2] \leq C\, s \log(p)/n$$

(iii) The inner product in each $\mathcal{H}^p$ *and and an element* $\Phi^p(\cdot)$ *of* $\mathcal{H}^p$ *generate a kernel* $K^p(\cdot,\cdot) = \langle \Phi^p,\, \Phi^p \rangle_{\mathcal{H}^p} \leq \mathcal{K}^p$ *such that*

$$proj\left(f \mid \{P^{pk_j}(\cdot)\}_{j=1}^{s}\right) = \langle w,\, \Phi^p \rangle_{\mathcal{H}^p},$$

for $w \in \mathcal{H}^p$ *and* $\|w\|_{\mathcal{H}^p} < \Lambda < \infty$

Assumption 2 essentially specifies the properties of the feature map that approximates the nonparametric regression. Associated with this feature map is the corresponding kernel defined as an inner product in $\mathcal{H}^p$.

Finally, we provide the set of assumptions that ensure the appropriate tradeoff between the sample size and the complexity of the approximating functional space.

ASSUMPTION 3 *Let* Δ^p *be the subspace of* $\mathbb{R}^p$ *such that* $0 \notin \Delta^p$ *and for each* $\delta \in \Delta^p$ *the component* $\delta_{k_j} > 0$ *where* $0 < k_j \leq p$ *is the subsequence defined in Assumption 2 and* $\delta_k = 0$ *otherwise. Let* $l_n \to \infty$. *Define* $\underline{\lambda}_n = \min_{\delta \in \Delta^p, \|\delta\|_0 \leq s l_n} \frac{\delta'\left(\frac{1}{n}\sum_{i=1}^{n} P^p(z_i)_i P^p(z_i)'\right)\delta}{(\delta'\delta)^{1/2}}$ *and* $\overline{\lambda}_n = \max_{\delta \in \Delta^p, \|\delta\|_0 \leq s l_n} \frac{\delta'\left(\frac{1}{n}\sum_{i=1}^{n} P^p(z_i)_i P^p(z_i)'\right)\delta}{(\delta'\delta)^{1/2}}$. *Suppose that*

(i) $\sup_{n \geq 1} \frac{\overline{\lambda}_n}{\underline{\lambda}_n} < \infty$.

(ii) $\max_{i\leq n, j\leq p} |P^{pj}(z_i)|^2/l_n = o(1)$, $\log p \leq C(n/\log^2 n)^{1/3}$ *and* $l_n^2 s \log(p) \leq Cn/log\, n$.

(iii) $\mathcal{K}^p \to 0$ *as* $p \to \infty$ *and* $\mathcal{K}^p\sqrt{n} \to \infty$

Based on Assumption 2 we can integrate the results in **?**, **?** and **?** to provide the following theorem.

THEOREM 1 *Suppose that statistical model satisfies Assumptions 1, 2, and 3. Consider the* $\sqrt{LASSO}$ *algorithm with* $\lambda = 2\sqrt{2\log(pn)/n}$, *the SVM regression algorithm with* $\epsilon = o\left(\mathcal{K}^p\sqrt{n}\right)$ *and* $C \geq E[(R^p)^2]^{1/2}$, *and the* L_2 *boosting algorithm with* $M = O\left(\left(\sqrt{n/\log(p)}\right)^{(3-\delta)/(4-2\delta)}\right)$ *with* $0 < \delta < 5/8$. *Then for each resulting estimator*

$$\frac{1}{\sqrt{n}}\sum_{i=1}^{n}(\widehat{f}(z_i) - f(z_i))^2 = o_p(1).$$

The result of the theorem is based on the following previous results. Under Assumption 1 (i), (ii), Assumption 2 (i), (ii) and Assumption 3 (i), (ii) we can verify the conditions of Theorem 8 in Belloni, Chernozhukov and Wang (2014) that yields the desired result for square root LASSO. Under the statement of the theorem, Assumption 1 (i), (ii), Assumption 2 (i), (ii) and Assumption 3 (i), (ii) we can apply Theorem 12.2 in Buhlmann and van de Geer (2011) and yields the desired result for L_2-boosting. Under the statement of the theorem and Assumptions 1 (iii), Assumption 2 (i), (ii), (iii) and Assumption 3 (iii) we can apply Lemma 15.5 in Vapnik (1998) yielding the result for the SVM regression.

The property of the constructed estimator allows us to apply it in the settings of model averaging. We can formulate the object of interest as a single-dimensional parameter determined by a single conditional moment model

$$\theta = E[Y \mid Z = z].$$

In terminology of our previous discussion $f(\cdot) = E[Y \mid Z = z]$ In this case, the parameter of interest can also be expressed as weighted average prediction with weighing function $w(\cdot)$

$(E[w(Z)] = 1)$ such that the final parameter of interest can be expressed as

$$\theta = E[w(Z)f(Z)]. \tag{2.8}$$

Our next idea is to apply the result in Newey [1994] which assures that as long as model (2.8) is written in the "orthogonal form" with respect to the nonparametrically estimated function $f(\cdot)$ then the particular estimator of $f(\cdot)$ will not have an impact on the asymptotic distribution and thus the implementation details of the machine learning methods that we used to estimate $f(\cdot)$ will not affect the resulting average prediction.

THEOREM 2 *Suppose that $\widehat{f}(\cdot)$ is a machine learning method satisfying conditions of Theorem 1. Suppose that*

$$w(z) = Var(Y|Z = z)^{-1}$$

Then the resulting average prediction $\widehat{\theta}$ is asymptotically normal with

$$\sqrt{n}(\widehat{\theta} - \theta) \xrightarrow{d} N(0, V)$$

with

$$V = E[Var(Y|Z)^{-1}]^{-1}$$

Proof: Suppose as before that $z \in \mathcal{Z} \subset \mathbb{R}^d$ with $d \ll n$ and finite. Consider the likelihood of the model $\ell(y, z)$ and consider its parametrization $\ell_t(y, z)$ with some scalar parameter t. Denote $s_t(y|z)$ the score of the conditional distribution $\ell_t(y|z)$ and $s_t(z)$ the score of the unconditional distribution $\ell_t(z)$. The full score $s_t(y, z) = s_t(y|z) + s_t(z)$. Then we can write the tangent space of the model $\mathcal{T}_t = \{a(y, z) + t(z)\}$ where $E[a(Y, Z)^2] < \infty$ and $E[a(Y, Z)|Z = z] = 0$ and $E[t(Z)^2] < \infty$ and $E[t(Z)] = 0$.

Consider an arbitrary measurable weighting function $h(\cdot)$ such that $E[h(Z)^2] < \infty$. We can then write

$$E[h(Z)(\theta - f(Z))] = 0.$$

Now we consider the sequence of parameters θ_t corresponding to the sequence of distributions $\ell_t(y,z)$ such that for $t=0$ $\theta_t=\theta$ and $\ell_t(y,z)$ corresponds to the true distribution and consider the directional derivative

$$\frac{\partial}{\partial t}E_t[h(Z)(\theta_t - f_t(Z))] = 0,$$

where we highlight the elements of the moment function depending on parameter t. We can express this derivative using the fact that $f_t(z) = E_t[Y \mid Z=z]$:

$$E[h(Z)]\frac{\partial \theta_t}{\partial t}\big|_{t=0} = E[h(Z)Y\, s_t(Y|Z)] + E[h(Z)(f(Z)-\theta_t)s_t(Z)].$$

Now the idea to find the "orthogonal" form of the estimator will be to find an element of $\mathcal{T}_t$ such that the directional derivative of the parameter of interest can be represented as a projection of the score of the parametrized model on the tangent space. We recall that $E[s_t(Y|Z) \mid Z=z] = 0$ and $E[s_t(Z)] = 0$. Then the direct verification suggests that

$$\frac{\partial \theta_t}{\partial t} = E[\Psi(Y,Z)s_t(Y,Z)],$$

where

$$\Psi(y,z) = h(z)(y-f(z)) + h(z)(f(z)-\theta).$$

We find variance

$$\mathrm{Var}\left(\Psi(Y,Z)\right) = E[h^2(Z)\mathrm{Var}(Y|Z)].$$

Thus, the choice $h(z) = \mathrm{Var}(Y|Z)^{-1}$ minimizes the variance of $\Psi(Y,Z)$.

Next we apply Lemma 5.3 in Newey [1994] to the estimator defined by

$$\widehat{\theta} = \frac{1}{n}\sum_{i=1}^{n}\Psi(y_i,z_i)$$

with $f(\cdot)$ replaced with $\widehat{f}(\cdot)$. Provided that the estimator $\widehat{f}(\cdot)$ satisfies assumptions of The-

orem 1 is satisfies conditions of Lemma 5.3 in Newey [1994]. This leads to the result that

$$\sqrt{n}(\widehat{\theta} - \theta) \xrightarrow{d} N(0, V)$$

with

$$V = E[\mathrm{Var}(Y|Z)^{-1}]^{-1}$$

Q.E.D.

2.5 Additional empirical procedures

In addition to the three inference procedures described above we consider four additional machine learning procedures that we also interpret as methods for the estimation of the conditional expectation $E[Y \mid Z = z]$.

2.5.1 Forward stepwise and forward stagewise models

As in the square-root LASSO procedure and L_2 boosting we consider an explicit mapping $\mathcal{Z} \mapsto \mathcal{X}$. As in standard nonparametric models this mapping can be represented via a collection of basis functions $\{P^{pj}(\cdot)\}_{j=1}^{p}$ where the size of the set of functions p is allowed to slowly grow with the sample size n. Unlike LASSO, where the model is selected by fitting the penalized least squares objective function that leads to corner solutions making multiple coefficients to be equal to zero, the stepwise and stagewise models lead to sparse coefficient sets via the construction of the variable selection procedure.

The incremental version of the stagewise regression model uses the explicitly set step size ϵ that can be calibrated based on the convergence performance of the estimation procedure. The procedure itself is based on the iterative addition of the "explanatory variables" $P^{pj}(\cdot)$. The procedure is initialized from the baseline model $\widehat{f}^{(0)}(z) = \bar{y}$ with the corresponding coefficients of the nonparametric representation $\beta_j^{(0)} \equiv 0$. The corresponding residuals are computed in the standard way as $r_i^{(0)} = y_i - f(z_i)$. Then at iteration k we find the predictor $P^{jp}(\cdot)$

maximizing the correlation between $r_i^{(k)}$ and $P^{jp}(z_i)$. Then the corresponding coefficient in the nonparametric representation is updated as $\beta^{(k+1)} = \beta^{(k)} + \epsilon \operatorname{sign}\left(\widehat{\operatorname{corr}}(r^{(k)}, P^{pj}(\cdot))\right)$. The residual then is updated to offset the change in the coefficient β_j. The procedure then stops when the maximum correlation between $r_i^{(k)}$ and $P^{jp}(z_i)$ does not exceed the pre-set tolerance (or the maximum allowed number of steps has been achieved). In our analysis we choose the maximum number of steps at 16,000 to ensure that the procedure is stopped via the tolerance criterion.

In the stepwise algorithm, the set of regressors used for estimation is iteratively updated based on the minimization of the Akaike Information Criterion. The model is initialized at $\widehat{f}^{(0)}(z) = \bar{y}$ same as in the stagewise method. Then at step k we select one of the components $P^{jp}(\cdot)$ for which $\beta_j^{(k-1)} = 0$ that yields the highest contribution to the AIC. This re-computation can be done efficiently by invoking the QR factorization in the corresponding first order condition and thus does not require a repetitive multi-dimensional minimization. The procedure stops once the improvement in the AIC falls below the selected tolerance or the maximum set number of steps is exceed.

2.5.2 Regression Trees and the Random Forest

The random forest model is different from most of the models that we considered previously in the sense that it is explicitly not using the spectral properties of the functional space that contains $f(z) = E[Y \mid Z = z]$. Instead, the method is based on selecting the best iterative factorization of the state space $\mathcal{Z}$ such that within each element of the partition of the state space function $f(\cdot)$ can be approximated by a constant.

The basic element of the random forest algorithm is a regression tree. The regression tree is defined by two basic parameters: the maximum depth (also called the number of splits) and the size of the terminal nodes (also called leaves) determining the minimum number of points within each partition of the state space. Denote the tree parameters θ. Then given the sample $Z_n = \{z\}_{i=1}^n$ the regression tree is defined by the following algorithm. The algorithm is initialized at $\widehat{f}^{(0)}(z) = \bar{y}$. At step k the state space is split into rectangular areas $R_j^{(k)}$ and

the regression tree predicts as

$$\widehat{f}^{(k)}(z) = \sum_j \frac{\sum_{i=1}^n y_i \mathbf{1}\{z_i \in R_j^{(k)}\}}{\sum_{i=1}^n \mathbf{1}\{z_i \in R_j^{(k)}\}} \mathbf{1}\{z \in R_j^{(k)}\}.$$

Then the algorithm is updated by searching for the partition element $R_j^{(k)}$ and the dimension l of z such that if $R_j^{(k)} = [z_{jL}^1, z_{jU}^1] \times \ldots \times [z_{jL}^p, z_{jU}^p]$, then the split of this partitions into two $R_{j_1}^{(k+1)} = [z_{jL}^1, z_{jU}^1] \times \ldots \times [z_{jL}^l, 0.5(z_{jU}^l + z_{jU}^l)] \times \ldots \times [z_{jL}^p, z_{jU}^p]$ and $R_{j_2}^{(k+1)} = [z_{jL}^1, z_{jU}^1] \times \ldots \times [0.5(z_{jU}^l + z_{jU}^l), z_{jU}^l] \times \ldots \times [z_{jL}^p, z_{jU}^p]$ provides the greatest contribution to the sum of squared residuals corresponding to the new partition $R_j^{k+1)}$:

$$\min_{l=1,\ldots,p,\, R_j^{(k)}} \sum_{i=1}^n (y_i - \widehat{f}^{(k+1)}(z_i))$$

over all j and dimensions $l = 1, \ldots, p$. The iterations continue until one of the convergence criteria (the size of the terminal node or the depth of the tree) is met. It can be shown that when the outcome variable is binary, the minimization problem used to construct the tree is analogous to Manski's maximum score problem over the binary splits of the state space.

The random forest model is generated as an ensemble of trees via bootstrap. One draws B bootstrap samples from the original sample X_n. Each such a bootstrap sample X_b and the corresponding parameters θ_b yield the estimated regression tree $\widehat{f}_b(\cdot)$. Then the random forest model provides the value which is the average over the bootstrap samples:

$$\widehat{f}(z) = \frac{1}{B} \sum_{b=1}^B \widehat{f}_b(z).$$

2.6 Empirical Application

This section compares econometric models with machine learning ones using a typical demand estimation scenario—grocery store sales. The machine learning models in general produce better out-of-sample fits than linear models without loss of in-sample goodness of

fit. If we combine all the models linearly with non-negative weights, the resulting combination of models produces better out-of-sample fit than any model in the combination. This section also illustrates how machine learning models could work with unstructured data or sparse data. Unstructured data is not organized to feed into models directly like structured data. For instance, the text description of a bag of chips and the image of the bag are unstructured data. Sparse data is a type of data where most of the elements are zeros. Both of unstructured and sparse data would be hard to handle in econometric models. The last part of this section uses the same data and model structures to estimate the promotional lift of sales.

The IRI marketing data set, which is described in Bronnenberg et al. [2008], contains scanner panel data from grocery stores within one grocery store chain for six years. The product category we chose is salty snacks. A unit of observation is product j, uniquely defined by a UPC (Universal Product Code), in store m at week t. There are 3,045,513 observations, which includes 3,337 unique products. Let q_{jmt} be the number of bags of salty snack j sold in store m at week t. If $q_{jmt} = 0$, we do not know if it is due to no sale or out-of-stock and the observation is not filled in. The price p_{jmt} is defined as the quantity weighted average of prices for product j in store m at week t. Therefore if $q_{jmt} = 0$, the weight is also 0. In addition to price and quantity, the data contains attributes of the products (such as brand, volume, flavor, cut type, cooking method, package size, fat and salt levels) and promotional variables (promotion, display and feature). Table 2.1 shows the summary statistics of quantity, price and dollar spent per transaction. Table 2.2 shows the number of unique values for the category variables as well as the three most common values of product attributes: brand, volume, flavor, cut type, cooking method, package size, fat and salt levels.

In our application, we compare eight alternative models of demand to predict q_{jmt}. The eight models are linear model, stepwise, forward stagewise, LASSO, random forest, support vector machine, gradient boosting trees, and Logit. Linear model and Logit are traditional econometric models where the others are popular machine learning algorithms. We also perform a linear combination of all the models in the effort to increase prediction accuracy.

Table 2.1: Summary Statistics

Variable	Mean	Median
Price	2.12	1.99
Quantity	15.80	6.00
Dollars	28.11	12.19
#Stores	1,560	
#Weeks	313	
#UPC	3,337	
#Obs	3,045,513	

Table 2.2: Category Variables

Variable	#Levels	Three Most Frequent Values		
Brand	237	Pringles	Utz	Lays
Product Type	4	Potato Chip	Potato Crisp	Potato Chip and Dip
Packaging	20	Bag	Canister	Plastic Wrapped Cardboard Canister
Flavor	207	Original	Sour Cream & Onion	BBQ
Fat Content	16	Missing	Low Fat	Fat Free
Cooking Method	47	Missing	Kettle Cooked	Old Fashion Kettle Cooked
Salt Content	14	Missing	Lightly Salted	Sea Salt
Cutting Type	32	Flat	Missing	Ripple

We use R to compute all the results and a list of related R packages is provided in this section.

2.6.1 Linear Regression Models

The linear regression is a typical approach to estimate demand by approximating the demand using a function form of the following:

$$\ln q_{jmt} = \beta' X_{jmt} + \zeta_{mt} + \eta_{jm} + \epsilon_{jmt}$$

where X_{jmt} is the matrix of attributes including log of own prices, product attributes, advertising and promotion indicators, ζ_{mt} is the market specific seasonal factor, η_{jm} is the product

specific market effect and ϵ_{jmt} is an idiosyncratic shock to each product, market and time.

Table 2.3 shows the output of the linear model where only the significant coefficients are displayed.

2.6.2 Logit

We followed Berry et al. [1995] to do a logit-type model of market shares. We project estimated $\hat{q}_{jmt}$ on product attributes dummies, store fixed effects and week fixed effects. Then we sum $\hat{q}_{jmt}$ over stores and weeks. Assuming that market sizes are fixed, we calculate market share by dividing $\hat{q}_{jmt}$ by market size. The log of market share is taken as the dependent variable in our Logit model. Table 2.4 shows the output of Logit model with traditional regression where only the significant coefficients are displayed.

2.6.3 Stepwise, Forward Stagewise and Sqrt LASSO

In practice, stepwise and forward stagewise can be realized in R package **lars**. We take the default parameter t and λ in the package. These three model converge in similar ways where in each step, the most important variable is added to the model. We limit the maximum number of steps to 100 in our practice for demonstration purposes because it takes significantly longer to converge if the number of steps is larger.

Square Root Lasso is from the *slim* function of R package **flare**, where *slim* stands for Sparse Linear Regression using Non-smooth Loss Functions and L1 Regularization. In function *slim*, we set type=lq and q=2 (loss function) to use the Square Root Lasso. Default number of lambdas in the sequence is 5. We set it to be 40. Default min/max value of the sequence of lambdas is 0.3*max and $\pi\sqrt{(log(d)/n)}$. The authors of the **flare** package stated that SQRT Lasso is tuning insensitive based on Belloni et al. (2011).

2.6.4 Random Forest

Random forecast is implemented in R package **randomForest**. Random forest has a few parameters about the tree structure: *mtry* as the number of variables to randomly select at each tree, *ntree* as the number of trees in the model, *nodesize* as the minimum size of the terminal nodes of each tree, and *maxnodes* as the maximum number of terminal nodes each tree can have.

We first run a cross-validated version of random forest (function *rfcv*) to determine the best number of variables to sample at each tree. This function uses K-fold cross-validation. We set scale=log, step=0.5, which means we will drop 50% of the variables in every step.After we determine the best number of variables to try at each tree, a random forest model (function *randomFores*t) was executed with the optimal *mtry* and the other parameters at default values. The default value for *nodesize* is 5, *ntree* 500 and *maxnodes* NULL. The cross-validation function *rfcv* does not decide the optimal values for these parameters.

Table 2.5 displays the variable importance of the twelve most important variables in determining Log Quantity using two metrics. The percentage *increase in mean squared error* is the increased percentage of mean squared error if a variable is excluded from the model. Node purity measures how much the additional variable or tree split reduces the residual sum of squares. Thus, the *increase in node purity* measures the size change in node purity if a variable is excluded from the model.

2.6.5 Support Vector Machine

We use functions *svm* and *tune.svm* in R package **e1071** to implement the support vector machine model. We specify type=eps-regression as our dependent variable continuous.We specify a 10-fold cross-validation in the support vector machine.

We use a tuning function *tune.svm* to do the 10-fold cross-validation of SVM to determine the best parameters of *epsilon*, *cost* and *gamma*. *Epsilon* is ϵ in the loss function with default 0.1. We test *epsilon* from 0 to 1 with interval 0.2. *Cost* is the constant of the regularization

term in the Lagrange formulation. We test *cost* from 0.5 to 2 with interval 0.5. *Gamma* is the parameter in the kernel function. We test *gamma* from 0.01 to 0.1 with length of five. The kernel used in the training is *radial basis* - Gaussian. This is tested in our data to out perform the other kernels. The other kernel options are polynomial, linear and sigmoid. Here are the kernel functions:

- linear: $u'v$
- polynomial: $(\gamma u'v + coef0)^{degree}$
- radial basis: $exp(-\gamma|u - v|^2)$
- sigmoid: $tanh(\gamma u'v + coef0)$

2.6.6 L_2 Boost

L_2 Boost is from function *glmoost* of R package **mboost** for boosting with component-wise least squares. We define the loss function as the squares of errors and define the negative gradient and stopping parameter for the *glmoost* function.

Table 2.6 shows the coefficients from L_2 Boost where only 36 out of 2243 variables have non-zero coefficients (we show only a portion of the non-zero coefficients).

2.6.7 Combined Model

In order to compare models, we want to split the data into training and testing set, where we train the model using the training set and pretend the dependent variable in the test set is unknown and predict on the test set. Because we have eight models, we want to assign weight to each model when building a linear combination. However, the weight based on the training set is biased. Some models like linear model tend to get a good in-sample fit but a bad out-of-sample fit, and this grants these model a very large in-sample weight which could be misleading.

Therefore, we randomly split the data into three pieces to do cross validation, where the model weights are determined on the validating set. This three way data partitioning mitigates the possible large out-of-sample error for some models when over fitting happens. 25% of the data is used as the test set, 15% is used as the validate set, and the remaining 60% is used as the training set. Table 2.8 shows how the data is sliced into three pieces.

Combining Models with Asymptotic Properties

We combine six models that has asymptotic properties using a constrained linear regression, in Table 2.7. The six models are: linear regression, Square-root Lasso, Support Vector Machine, L_2 Boosting, Logit with market share predicted by a regression model. The response variable is log of quantity sold per week. The covariates are log of price, product attributes variables, promotional variables, store fixed effects, and week fixed effects. We provide the same covariate matrix to all of the models expect for the Logit model, where all the fixed effects are excluded.

We combine them using the following algorithm:

1. Fit the actual dependent variable in the training set on each of the five models.

2. Make out-of-sample prediction using the model coefficients from the first step on the validate set.

3. Take only the validation set. Treat the predicted values of the dependent variable from the five models as regressors, treat the actual value of the dependent variable as the response variable, then do a constrained linear regression. The sum of the coefficients are constrained to be one. Get all the coefficients and regard them as weights in the combined model.

4. Use the fitted models to predict in the test set, and apply the weights from validate set to each model, sum them up and form the linearly combined prediction.

In Table 2.7, support vector machine gets the most weight out of the five models due to its small prediction error (measured by RMSE) in the validation set. Its prediction error in the test set is also the smallest among all models.

Combining All Models

In Table 2.8, we compare eight models: two of them are traditional econometrics models and seven of them are more in the context of machine learning as introduced before. Our purpose is to run a horse race of models by comparing out-of-sample prediction errors.

Table 2.8 shows the comparison of the models. In the scenario of out-of-sample prediction error, the best two models are random forest and support vector machine. The combined model, where we regress the actual value of the response variable on a constrained linear model of the predictions from eight models, outperforms all the eight models, which follows the optimal combination of forecasts in Bates and Granger [1969]. Random forest and support vector machine get more weights in the combined model due to their good performance out-of-sample.

Based on Section 2.3, the combination of models converges to asymptotic normal distribution at $\sqrt{n}$ rate, regardless of the what individual models there are. Therefore, we could bootstrap the combined model to get the confidence interval, knowing that it is asymptotically normal.

Table 2.9 provides the summary statistics of the residual between predicted and actual values of quantity in both validation and testing set.

Controlling Endogeneity in Prices

We use the Berry et al. [1995] instrumental variables to control endogeneity in prices. The endogeneity is assumed to be due to variation in marginal costs. The instruments include own product characteristics, the sums of the values of the same characteristics of other products from the same brand, the sums of the values of the same characteristics of products from other brands.

In the first stage regression of log price on the instrumental variables, the sums of the values of the same characteristics of products from other brands are ignored due to colinearity. The summary of the first stage regression is displayed in Table 2.10, where we only show the coefficients on the numerical variables such as volume and scores of characteristics. The explanatory power of the instruments is satisfactory and a majority of the instruments are significant, so we move on to the second stage where we add the predicted residuals from the first stage regression to the right hand side explanatory variables.

The predicted residuals of log price are used in the models in addition to the log price variable in Table 2.7. This produces the new results in Table 2.11, which are similar to the original ones in three ways: 1) one of the machine learning models out performs the rest and gains the majority of weight in the combined model; 2) the validation set generates smaller errors than the test set; 3) linear model has similar weights in the combination in both cases.

2.6.8 Asymptotics

We dedicate this section to show empirically the asymptotics of the selected machine learning models mentioned in Section 2.3. We use our dataset but limit the number of observations to a sequence of smaller numbers. We run the same model specification as the main application for these datasets of different sizes and compute standard errors of the root mean square error in prediction using subsampling. As Table 2.12 Figure 2.1 shows, in both validate and test set, the standard errors are going to zero when the number of observation grows.

2.6.9 Discussion

Variable Selection

When the number of independent variables is large, it is common to have some degree of multicollinearity. The shrinkage models could help us reduce the multicollinearity intelligently. A usual statistics to determine multicollinearity is Variance Inflation Factors (VIF).

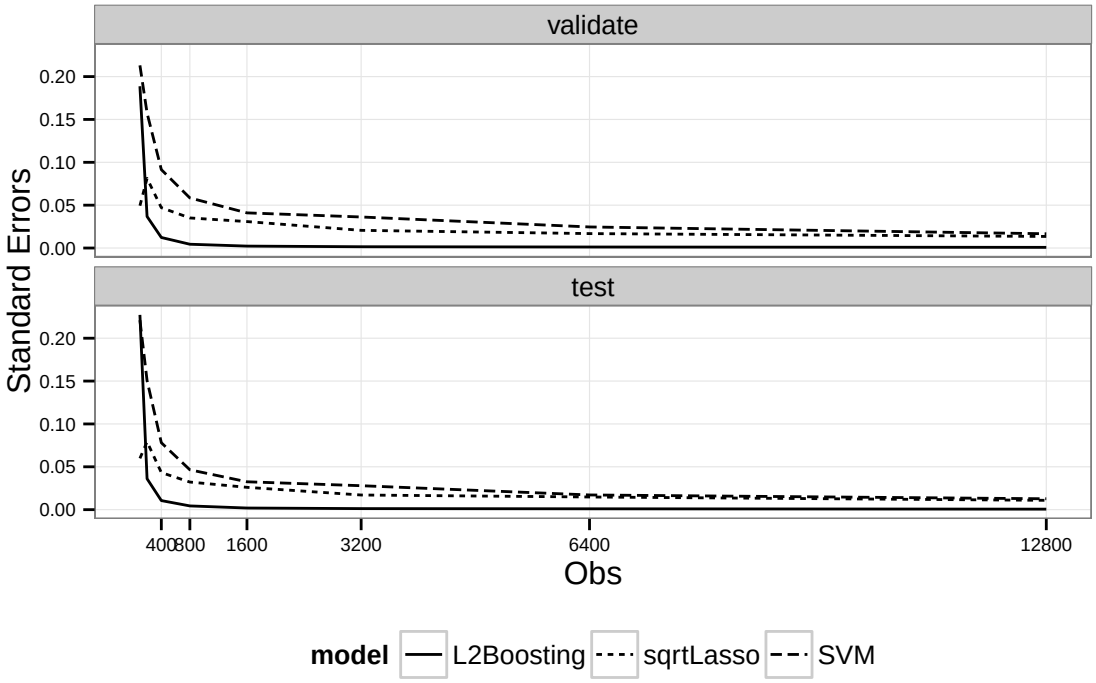

Figure 2.1: Standard Errors of RMSE

The VIF for covariate X_j is defined as:

$$VIF_j = \frac{1}{1 - R^2_{-j}} \tag{2.9}$$

where R^2_{-j} is the R-squared by regressing covariate X_j on the rest of covariates. Table 2.13 shows some VIFs for the independent variables used in the linear regression model. To compare, after LASSO selects variable, we also have the VIFs for the independent variables in the linear regression using only selected variables. The VIFs in the non-selecting case are much higher than the LASSO selection for many variables. Using 5 as a threshold for multicollinearity, LASSO successfully reduces multicollinearity in the independent variables. We believe it's more attractive than ad hoc variable selection procedures commonly used in practice.

Random forest on the individual model pieces

We use the predicted values of all the eight models from the validation set as input variables and use the actual log quantity as the output variable to train the random forest model. We do a 10-fold cross-validated version of random forest to determine the best parameters (same approach as described in Section 2.6.4) and then predict the output variable log quantity in the test set. Then we evaluate the prediction by the combined model with RMSE.

Table 2.14 shows combining the eight models using a random forest model instead of linear combination in Table 2.8. The importance of each individual model is shown in the right column, which is similar to the Weight column of Table 2.8.

Bag of Words

The mere feat of encoding a vector of hedonic attributes commonly used in demand estimation for this data could be cumbersome. In applied econometrics, researchers commonly restrict attention to the products with the largest demand and encode regressors for brands and a small vector of product attributes. In our application, we propose using unsupervised learning to construct product level regressors. In particular, we use the unstructured text that describes the product in the raw data and apply the bag of words model. This has the advantage of being a simpler computation and allows us to avoid the arduous task of encoding attributes for thousands of products. Since it is more scalable, it allows us to model the demand for all of the products rather than restricting attention to the products with the largest demand. Also, it could be viewed as less ad hoc than hand coding product attributes since it imposes fewer a priori restrictions on the hedonic attributes that we should include as regressors in our model. We note that there is a large literature on this form of unsupervised learning which sometimes goes by the name of feature extraction in computer science. With the exception of Gentzkow and Shapiro [2010], there has been relatively little use of this method in economics. We believe that this is promising for demand estimation because it allows a new source of data to construct covariates such as the raw text in product

descriptions or product reviews.

Aside from structured features like package type, cut type of potato chips, the hedonic attributes of the product can also be defined by unstructured texts that describe the product. A bag of words model could easily turn the unstructured texts into large amounts of features. Unsupervised learning (clustering) on these features could be a more scalable approach than hand coding them. The procedures following the unsupervised learning are the same as the way we deal with structured features. Yet this straightforward approach has better prediction power than structured models.

The bag of words model analyzes a corpus of K documents, comprising a dictionary of M words, and finds the relations of words and documents. In our case, the K documents are the n unique potato chips descriptions. We cluster the descriptions, via manipulation of the document-term matrix. A document-term matrix is a mathematical matrix that describes the frequency of terms that occur in a collection of descriptions. In a document-term matrix, rows correspond to each unique product in the collection and columns correspond to terms. Where it is tedious to encode all the features of a product, bag of words provides a simple way to exploit the rich features of products.

In a high dimensional learning problem, only some parts of an observation are important to prediction. For example, the information to correctly categorizing a product may lie in a handful of its words. The extraneous words could prove computational burdensome, so word regularization may be helpful. Possible methods for regularization include LASSO and other shrinkage models. Therefore it's natural to combine the technique of bag of words with our models.

Top/Tail Products

In marketing literature, people usually prune out the tail products (for example Nevo [2001]). But, with the new methods, we want to show that a training data set with all possible products predicts better than a data set with only the top products. To show that, we take only the top twenty products to train the model and predict the sales of the tail products

and compare the prediction fit to those in Table 2.8.

We use the same eight models as modeling examples to demonstrate the difference. We rank all the products by total units sold and only mark the top twenty as the training set. Using exactly the same methodology in our main application, we get the root mean squared error as a measure of fit for eight plus combine model. The prediction error and weights is presented in Table 2.15.

As Table 2.15 shows, when we only train on top twenty products, the in-sample predictions (top 20 products) are very good, but the out-of-sample predictions (other products) are not so good, in contrast to Table 2.8, where the in-sample and out-of-sample error are very close when we randomly split the data for training. As the top twenty products may not necessarily explain all the features of the rest of the products, the predicting power for the other products is therefore weakened.

Practical Advantages

There are some other practical advantages to the machine learning models or their generic approaches.

In random forest, the missing values could be imputed, for example, as the median of its column. This imputation will not effect accuracy much since the randomness of subsampling and the trees grown. Another approach for categorical variables is to simply create a new category called "missing." This new category might capture the behavioral differences in observations with missing values and the ones not missing.

If we have a large dataset and we want to do model selection or model assessment, the best approach is to randomly divide the dataset into three parts: a training set, a validation set, and a test set. The training set is used to fit the models; the validation set is used to estimate prediction error for model selection; the test set is used for assessment of the generalization error of the final chosen model. The test set should be kept separately and be brought out only at the end of the analysis.

If, however, we do not have enough data to split three ways, we can efficiently re-use the

sample data by cross-validation or bootstrap.

In addition to the models we mentioned in our paper, there are some other very popular machine learning approaches that could be helpful to readers in economics. For example, EM (Expectation-Maximization) algorithm for simplifying difficult maximum likelihood problems; Graphical models for complicated conditional expectations; Neural Networks for extracting linear combination of features and modeling response variable as a non-linear function of the features, and so on.

2.6.10 Lift Estimates

The other interesting problem we want to look at is estimating promotional lift. In our data, a product is tagged as on promotion if the price reduction is greater than 5%. By considering promotion as a randomized treatment, we follow the methodology suggested by Varian [2014] to estimate the promotional lift. Our models are trained on the control group (no promotion) and then used to predict out of sample on the treated group (on promotion). The control/treatment partition is based on whether the product is on price reduction promotion, instead of random assignment in the last example.

The most important variables we want to study are the price promotion as well as how stores feature and display the products. A product is tagged as in promotion if the price reduction is greater than 5%. There are four levels of feature ads: large, medium, small and no ad. There are three levels of display: major (including lobby and end-aisle), minor and none. Table 2.16 shows the frequency of promotion in the dataset.

If everything else is the same in the control and the treatment group, we believe the differences between predicted and actual in treatment group are our treatment effects. Based on model coefficients from the control group, we predict the quantities sold using the independent variables in the treated group. The differences between the predicted and the actual values of quantity sold in the treated group are therefore our lift estimates.

The lift estimates from eight models mentioned before are presented in Table 2.17. We split the control group where there is no promotion into a training and a validation set. We

fit the same models as we used in the main application using the training data and combine them linearly with the validation data set. Then we predict the model using the treated group where there is promotion. The weights form the linear combination of models with the validation set is used to construct the combined model in the treated group. All the machine-learning models calculate similar promotional lifts when the linear model produces a smaller lift for promotion.

2.6.11 Computation Tools

In this application we face constraints on memory and CPU when processing the data with millions of observations with complicated machine learning models. The time it takes to compute a Random Forest object with data of such size using a single work station could be days, even weeks. A solution to solve the computation constraints is to utilize high performance computing tools, such as parallel processing using Revolution R® and MATLAB® Parallel Computing Toolbox™. Both of them are available on Amazon Web Services(AWS) Marketplace at low costs.

Revolution R has two major packages - RevoR and ScaleR. RevoR automatically uses all available cores and processors to reduce computation times without modifying a standard R script. A real time simulation shows it speeds up computation by three to thirty times compared to a single core standard R process. ScaleR scales up data size to 1 to 16 tera-bytes easily by dividing data into pieces and accessing the pieces with different processors.

In these parallel processing frameworks, a common implementation is MapReduce. It is useful when the data is too large to be held or processed in memory. There are two stages in MapReduce: Map and Reduce. In the Map stage the program (Revolution R® or MATLAB®) pre-processes each distributed data trunk in parallel and performs preliminary statistical modeling on each data trunk in parallel. In the Reduce stage, the program gathers all the information for each distributed data trunk from Map stage, summarizes the information and returns it to the user. This is much faster and takes less memory than storing and accessing data directly from memory.

2.7 Conclusion

In this paper, we survey a set of models from statistics and computer science. We select a few machine learning models to compare with traditional econometric models. The models we focus on in this paper include linear regression as the baseline model, logit as the econometric model, stepwise, forward stagewise, LASSO, random forest, support vector machine and bagging as the machine learning models. We derive novel asymptotic properties for the machine learning models.We use Monte Carlo simulation to demonstrate the properties of the models and also show that combining all the underlying models with a linear regression improves out-of-sample prediction accuracy.

We illustrate the properties of these models by using a real world data set with scanner panel sales data of potato chips. First, we compare the prediction accuracy of these models and the machine learning models consistently give better out-of-sample prediction accuracy while holding in-sample prediction error comparable. By combining all the models via weighted linear regression, we are able to improve the out-of-sample prediction accuracy even more. Second, we compare two scenarios where one, as the traditional marketing literature suggests, prunes the market to only the top sold products, and the other, our approach, uses a mix of both top and tail products. Our approach has better prediction accuracy in demand. Third, we estimate the promotion lift using the predictions for treatment effects. Last, we explored the unstructured text of product description from the raw data and apply the bag of words model. This has the advantage of being a simpler computation and allows us to avoid the arduous task of encoding attributes for thousands of products.

Our approach is robust to a large number of potentially collinear regressors; it scales easily to large data sets; the linear combination method selects the best model automatically and produces the best in-sample and out-of-sample prediction accuracy; and the method can flexibly approximate arbitrary non-linear functions, even when the set of regressors is high dimensional and we also allow for fixed effects.While demand estimation is our motivating application, we believe that the approach we have proposed can be useful in many other

microeconometric settings.

Table 2.3: Linear Regression

Log Quantity	Estimate	Std. Error	t value	Pr($>\|t\|$)	
Log Price	-0.639	0.055	-11.708	$<$ 2e-16	***
Promotion	0.466	0.039	11.926	$<$ 2e-16	***
Feature: None	-0.630	0.067	-9.334	$<$ 2e-16	***
Display:					
Minor	0.708	0.049	14.341	$<$ 2e-16	***
Major	0.637	0.049	13.119	$<$ 2e-16	***
Brand:					
Herrs	-0.351	0.156	-2.253	0.024	*
Jays	-1.101	0.244	-4.516	0.000	***
Kettle Chips	-0.995	0.236	-4.217	0.000	***
Lays	-0.337	0.159	-2.124	0.034	*
Lays Bistro Gourmet	-0.656	0.188	-3.480	0.001	***
Lays Natural	-1.662	0.327	-5.079	0.000	***
Lays Stax	-1.481	0.183	-8.104	0.000	***
Lays Wow	-0.485	0.204	-2.379	0.017	*
Michael Seasons	-1.655	0.239	-6.921	0.000	***
Pringles	-0.794	0.156	-5.090	0.000	***
Pringles Cheezums	-0.644	0.211	-3.055	0.002	**
Pringles Fat Free	-0.624	0.189	-3.308	0.001	***
Pringles Prints	-1.876	0.314	-5.982	0.000	***
Pringles Right Crisps	-0.881	0.128	-6.892	0.000	***
Ruffles Natural	-1.379	0.389	-3.549	0.000	***
Ruffles Snack Kit	-1.555	0.307	-5.061	0.000	***
Utz	-0.543	0.149	-3.635	0.000	***
Wise	-0.505	0.165	-3.062	0.002	**
Wise Ridgies	-0.984	0.167	-5.888	0.000	***
Volume	0.469	0.113	4.142	0.000	***
Package:					
Canister	0.437	0.091	4.800	0.000	***
Canister In Box	0.453	0.130	3.494	0.000	***
Flavor:					
BBQ	0.167	0.066	2.534	0.011	*
Cheddar	0.241	0.080	3.026	0.002	**
Cheese	-0.443	0.205	-2.164	0.031	*
Ketchup	-0.680	0.244	-2.787	0.005	**
Onion	0.339	0.066	5.107	0.000	***
Original	0.704	0.061	11.588	$<$ 2e-16	***
Spicy	-0.211	0.105	-2.005	0.045	*
Salt: No Salt	-0.446	0.212	-2.099	0.036	*
Type of Cut: Flat	0.308	0.070	4.411	0.000	***
Store Fixed Effects	Yes				
Week Fixed Effects	Yes				
Adjusted R-squared	0.884				
Significance	0 ***	0.001 **	0.01 *	0.05 .	

Table 2.4: Logit with Regression Selection

Log Share	Estimate	Std. Error	t value	Pr($>\|t\|$)	
Log Price	0.296	0.113	2.624	0.009	**
Promotion	-0.441	5.192	-0.085	0.932	
Feature: None	0.263	0.151	1.745	0.081	.
Display:					
Minor	-0.215	0.104	-2.080	0.038	*
Major	-0.338	0.113	-3.000	0.003	**
Store Fixed Effects	No				
Week Fixed Effects	No				
AIC	6884.4				
Significance	0 ***	0.001 **	0.01 *	0.05 .	

Table 2.5: Random Forest Variable Importance

Log Quantity	%Increase in Mean Squared Error	Increase in Node Purity
Log Price	74.83	1196.68
Volume	56.81	855.79
Display: Minor	49.79	455.98
Promotion	43.76	519.72
Display: Major	43.29	267.43
Feature: None	42.05	592.37
Brand: Lays	39.82	367.29
Brand: Ruffles	33.21	76.97
Brand: Wavy Lays	32.95	143.46
Flavor: Classic	32.31	219.00
Flavor: Sour Cream & Onion	30.26	62.28
R-Squared	0.42	

Table 2.6: L_2 Boost Coefficients

Log Quantity	Coefficient
Log Price	-19.57
Promotion	18.24
Feature: Medium Ad	4.79
Feature: None	-19.85
Display: Minor	12.78
Display: Major	18.88
Brand: Kettle Chips	-3.41
Brand: Lance Thunder	-0.48
Brand: Lays	16.50
Brand: Lays Stax	-2.30
Brand: Ruffles	6.26
Brand: Wavy Lays	10.06
Flavor: Classic	11.30
Flavor: Sea Salt & Vinegar	-0.45
Type: Potato Chip and Dip	-0.49
Type: Potato Crisp	-1.10
Package: Canister in Box	-4.08
...	

Table 2.7: Linear Model Combination: Models with Asymptotics

	Train		Validation		Test		
	RMSE	Std. Err.	RMSE	Std. Err.	RMSE	Std. Err.	Weight
Linear	0.766	0.010	0.994	0.017	1.010	0.015	10.41%
Sqrt Lasso	0.977	0.007	0.984	0.013	0.995	0.009	1.71%
Support Vector Machine	0.543	0.007	0.889	0.018	0.900	0.012	87.57%
L2 Boosting	1.053	0.004	1.016	0.013	1.028	0.012	0.00%
Logit	3.282	0.170	3.509	0.340	3.915	0.263	0.32%
Linearly Combined			0.887		0.898		100.00%
# of Obs	1,827,308		456,827		761,379		
Total Obs	3,045,513						
% of Total	60%		15%		25%		

Table 2.8: Model Comparison: Prediction Error

	Train		Validation		Test		
	RMSE	Std. Err.	RMSE	Std. Err.	RMSE	Std. Err.	Weight
Linear	0.766	0.010	0.994	0.017	1.010	0.015	17.73%
Stepwise	0.930	0.008	0.969	0.017	0.980	0.014	0.00%
Forward Stagewise	0.977	0.007	0.985	0.015	0.995	0.013	0.00%
Sqrt Lasso	0.977	0.007	0.984	0.013	0.995	0.009	0.00%
Random Forest	0.927	0.007	0.914	0.017	0.916	0.013	37.46%
Support Vector Machine	0.543	0.007	0.889	0.018	0.900	0.012	44.79%
L2 Boosting	1.053	0.004	1.016	0.013	1.028	0.012	0.00%
Logit	3.282	0.170	3.509	0.340	3.915	0.263	0.02%
Linearly Combined			0.879		0.887		100.00%
# of Obs	1,827,308		456,827		761,378		
Total Obs	3,045,513						
% of Total	60%		15%		25%		

Table 2.9: Summary Statistics of Residual in Prediction

Validation Set	Mean	Std. Dev.	Median	Min	Max
Linear	-0.019	1.549	-0.015	-5.402	4.862
Stepwise	-0.023	1.440	0.018	-4.942	4.365
Stagewise	-0.023	1.353	0.040	-4.549	3.975
Sqrt Lasso	-0.023	1.347	0.011	-5.501	4.035
Random Forest	0.010	1.424	0.073	-4.591	4.618
Support Vector Machine	-0.010	1.462	0.008	-4.387	5.167
L2 Boost	-0.024	1.399	0.025	-4.952	4.661
Logit	-0.244	3.566	0.154	-39.141	3.135
Combined	-0.004	1.466	0.028	-4.811	5.053

Test Set	Mean	Std. Dev.	Median	Min	Max
Linear	-0.012	1.533	0.013	-5.385	5.387
Stepwise	-0.029	1.429	-0.026	-5.568	4.726
Stagewise	-0.072	1.402	-0.073	-4.521	3.869
sqrt Lasso	-0.056	1.375	-0.066	-6.262	4.069
Random Forest	0.088	1.489	0.168	-5.912	4.802
Support Vector Machine	0.004	1.515	0.012	-5.771	5.430
L2 Boost	-0.080	1.462	-0.031	-5.888	4.379
Logit	-0.322	3.910	0.154	-38.241	2.833
Combined	-0.033	1.435	-0.038	-5.126	4.698

Table 2.10: BLP Instruments - First Stage

Logprice	Estimate	Std. Err.	t value	P	
Own Product					
Volume	1.65	0.01	149.131	0.000	***
Product Characteristics Dummies			Yes		
Other Products - Same Brand					
Volume	0.00023	0.00013	1.760	0.078	.
Product Type	-0.00021	0.00007	-3.187	0.001	**
Package	0.00012	0.00003	3.519	0.000	***
Flavor	0.00000	0.00000	-0.160	0.873	
Fat	0.00017	0.00002	9.040	0.000	***
Cooking Method	0.00001	0.00000	4.497	0.000	***
Salt	-0.00114	0.00011	-9.972	0.000	***
Type of Cut	0.00000	0.00000	-0.305	0.761	
Other Products - Other Brands			Yes		
Intercept	2.18	0.18	12.041	0.000	***
R^2	0.7500				
Significance	0 ***	0.1% **	1% *	5% .	

Table 2.11: Linear Model Combination: Models with Asymptotics

Controlling Price Endogeneity with BLP Instruments

	Train		Validation		Test		
	RMSE	Std. Err.	RMSE	Std. Err.	RMSE	Std. Err.	Weight
Linear	1.037	0.007	1.351	0.013	1.371	0.011	13.62%
Sqrt Lasso	1.213	0.000	1.202	0.001	1.223	0.001	86.38%
Support Vector Machine	0.861	0.004	1.204	0.006	1.227	0.005	0.00%
L2 Boosting	1.200	0.001	1.202	0.002	1.221	0.001	0.00%
Logit	1.856	0.154	2.710	0.270	2.048	0.228	0.00%
Linearly Combined			1.198		1.219		100.00%
# of Obs	1,827,308		456,827		761,379		
Total Obs	3,045,513						
% of Total	60%		15%		25%		

Table 2.12: Convergence Rate for Three Machine Learning Models

Sample Size	100	200	400	800	1600	3200	6400	12800
				Validate				
Sqrt Lasso	0.2025	0.1380	0.0909	0.0566	0.0382	0.0320	0.0225	0.0162
Support Vector Machine	0.1613	0.1180	0.0767	0.0446	0.0375	0.0222	0.0162	0.0137
L2 Boosting	0.0487	0.0788	0.0467	0.0325	0.0318	0.0223	0.0177	0.0133
Combined	0.1944	0.1468	0.0828	0.0574	0.0417	0.0316	0.0258	0.0159
				Test				
Sqrt Lasso	0.2125	0.1364	0.0790	0.0492	0.0292	0.0259	0.0170	0.0130
Support Vector Machine	0.1635	0.1245	0.0765	0.0435	0.0299	0.0186	0.0142	0.0099
L2 Boosting	0.0621	0.0786	0.0466	0.0313	0.0261	0.0185	0.0138	0.0099
Combined	0.2251	0.1476	0.0813	0.0519	0.0367	0.0258	0.0199	0.0136

Table 2.13: Variance Inflation Factors

	VIFs after Selection	
Variable	**Linear**	**LASSO**
Product Type - Potato Chip And Dip	$+\infty$	3.5084
Brand - Ruffles Snack Kit	$+\infty$	3.4729
Logprice	4.1750	3.2319
Volumn	3.9775	3.1541
Cooking - Missing	$+\infty$	3.1100
Cooking - Kettle	$+\infty$	2.6495
Package - Canister	$+\infty$	1.8047
Fat - Regular	76.6610	1.5930
Brand - Lays	104.5904	1.5187
Promotion	1.4806	1.4388
Feature - None	2.3398	1.3369
Brand - Kettle Chips	27.3608	1.3222
Flavor - Original	2.8610	1.2875
Brand - Ruffles	50.1427	1.2802
Salt - Regular	3.0660	1.2732
...		

Table 2.14: Combining Models in Random Forest

	Train		Validation		Test		
	RMSE	Std. Err.	RMSE	Std. Err.	RMSE	Std. Err.	Var. Imp.
Linear	0.766	0.010	0.994	0.017	1.010	0.015	32.435
Stepwise	0.930	0.008	0.969	0.017	0.980	0.014	32.647
Forward Stagewise	0.977	0.007	0.985	0.015	0.995	0.013	24.607
Sqrt Lasso	0.977	0.007	0.984	0.013	0.995	0.009	23.135
Random Forest	0.927	0.007	0.914	0.017	0.916	0.013	34.845
Support Vector Machine	0.543	0.007	0.889	0.018	0.900	0.012	52.972
L2 Boosting	1.053	0.004	1.016	0.013	1.028	0.012	23.977
Logit	3.282	0.170	3.509	0.340	3.915	0.263	0.932
Combined by Random Forest			0.920		0.902		
# of Obs	1,827,308		456,827		761,378		
Total Obs	3,045,513						
% of Total	60%		15%		25%		

Note: In random forest, variable importance is defined as the average difference in out-of-bag mean squared error with and without permuting the variable, divided by the standard deviation of the difference. The bigger the variable importance value, the more important the variable is.

Table 2.15: Top 20 Products vs. the Other Products

	Top 20 Products		Other Products		
	RMSE	Std. Err.	RMSE	Std. Err.	Weight
Linear	0.397	0.034	2.037	0.037	35.37%
Stepwise	0.768	0.023	1.437	0.024	0.00%
Forward Stagewise	0.882	0.017	1.371	0.018	0.00%
Sqrt Lasso	0.935	0.015	1.374	0.017	0.00%
Random Forest	0.759	0.018	1.530	0.017	0.00%
Support Vector Machine	0.318	0.042	1.537	0.020	64.63%
L2 Boosting	0.920	0.021	1.378	0.019	0.00%
Logit	1.331	0.124	2.685	0.134	0.00%
Linearly Combined	0.277		1.433		100.00%
# of Obs	504,337		2,541,176		
Total Obs	3,045,513				
% of Total	16.56%		83.44%		

Table 2.16: Promotion Variables

Variable	Value	Frequency(%)
Promotion	Price Reduction < 5%	74.11
	Price Reduction >5%	25.89
Feature	Large Ad	5.35
	Medium Ad	4.98
	Small Ad	0.33
	None	89.33
Display	None	81.16
	Minor	10.66
	Major	8.18
Total Obs	3,045,513	

Table 2.17: Model Comparison:Promotional Lift (in units)

	Mean	t	95% Conf. Int.		Weight
Linear	9.646	8.171	7.332	11.960	23.37%
Stepwise	20.124	19.516	18.103	22.145	7.01%
Stagewise	22.458	21.018	20.363	24.552	0.00%
Sqrt Lasso	22.440	21.006	20.346	24.534	0.00%
Random Forest	18.276	17.705	16.253	20.299	68.00%
Support Vector Machine	25.920	23.428	23.752	28.089	0.00%
L2 Boost	22.995	21.386	20.887	25.102	0.00%
Logit	22.671	20.474	20.500	24.841	1.61%
Linear Combination	19.017	18.456	16.998	21.037	100.00%

Chapter 3

ESTIMATING THE IMPACT OF FRACKING ON HOUSE PRICES

By Jenny Ho and Miaoyu Yang

Abstract

We collect high dimensional data and extract features from house descriptions and images to use as controls within a hedonic model to estimate the impact of fracking on house prices in Pennsylvania. Supplementing a structured dataset with high dimensional unstructured data in the form of descriptive words and images of homes can help to close the gap caused by omitted variable bias. We construct curb appeal scores based on aesthetic features of home images. We then compare four models: OLS, LASSO - OLS, random forest and gradient boosting. The ensemble tree models (random forest and gradient boosting) yield 10% improvements in prediction accuracy compared to LASSO and OLS. Our results imply that royalty payments exactly compensate for the negative environmental effects on homes within 1 km of fracking wells but increase the price of houses farther away by up to 5%.

3.1 Introduction

Natural gas production has rose dramatically in the last decade, increasing by 40% from 2004 to 2014. The dramatic increase has been largely due to fracking. Fracking is the process of cracking shale rock with chemically treated water to release natural gas trapped beneath the rock. Shale gas as a percentage of total natural gas production increased from 1.6% of the total in 2000 to 23.1% in 2010. The US Energy Information Agency predicts shale gas production to double within the next 30 years and account for roughly half of all natural gas production by 2040.

Enhancements in fracking have been positive on the natural gas industry, allowing for the US to become independent of natural gas imports with potential to become a net exporter. However, the impact on local communities is more ambiguous. The effect on current communities has been two-fold: on the one hand, an increase in economic activity and financial compensation in the form of upfront checks and monthly royalty payments, while on the other hand, a flood of disamenities such as noise, air pollution, traffic, and increased risks of groundwater contamination. Fracking involves spraying large quantities of water treated with various and often undisclosed chemicals that allow for the cracks in the shale rock to stay open for longer and further increasing the gas flow. This water can contaminate local groundwater if not properly contained. As natural gas development continues to expand, energy companies will eventually need the approval of a wider demographic population. Conflicts between local residents and energy companies have already begun to surface in the form of bans and moratoriums on fracking in Colorado, New Jersey, and New York.

We estimate the impacts of fracking on house prices in Pennsylvania, which sits on the natural gas rich Marcellus Formation. Our results show that homes within 1 km to fracking wells had net zero effects on home prices, while those within 1-2.5 km of fracking wells experienced 5% increases in home prices. Without data on lump sum and royalty payments, these results are interpreted as the net effects of fracking on house prices. We suspect that homes in the immediate vicinity of a fracking well are compensated exactly for the negative environmental harm while those farther away are also compensated but suffer from less environmental disamenities resulting in increases in house prices. We use a hedonic model to measure the impact of fracking wells on property values while controlling for house features and demographic characteristics. Homebuyers factor in a high volume of property details before making a purchase. These include the standard structured variables such as number of rooms, bathrooms, square footage, etc. but also curbside appeal and local amenities such as nearby stores and parks. A majority of hedonic analyses done in the past include only a limited set features and suffer from omitted variable bias when information that buyers view as critical are not accounted for.

This paper more accurately models homebuyers' decisions by controlling for a rich set of features that include descriptions and images of homes that buyers view before purchasing. We also demonstrate a method to turn highly unstructured image features into interpretable structured variables. We apply this method to estimate the curb appeal of a home, but the method can be applied to a broader genre of extracting qualitative features from images. Text and image processing require machine learning methods that allows for high dimensional features and multicollinearity without loss of prediction accuracy. It is difficult to know which variables are important when using thousands of covariates. Thus, we have selected several machine learning methods that are robust to the inclusion of irrelevant variables in order to reduce the burden of searching for optimal regression specifications when the number of variables is high. These methods include LASSO, gradient boosting, and random forest. The tree-based models, gradient boosting and random forest, outperform linear models by over 10% in accuracy when predicting home prices.

The paper is organized as follows. Section 3.2 reviews the literature on hedonic price models and how hedonic models are used to estimate the marginal price of environmental amenities. Section 3.3 discuss the econometric models. Section 3.4 provides details on the data set, as well as how we process raw text and image as input variables. Section 3.5 demonstrate how we construct the curbside appeal score from images. Section 3.6 has results, and section 3.7 is the conclusion.

3.2 Literature

3.2.1 Hedonics

The hedonic price model is a widely accepted methods for estimating the demand of goods and environmental amenities using their attributes. This framework proposes that the prices of heterogeneous goods can be broken down into their individual attributes. It has enjoyed wide acceptance in environmental applications by treating public goods as attributes.

Applications of the hedonic method in environmental issues are diverse. The early ap-

plications involved the impacts of air quality on housing values. More recently, hedonic studies have considered the impact of a range of land uses including landfills with hazardous substances, past hurricanes as signals of future risks, and fracking with groundwater contamination. For example, Muehlenbachs et al. [2015] and Muehlenbachs et al. [2013] studied the housing market impacts of fracking.

While the hedonic theories are well established, Klaiber and Smith [2011] argued that there are several econometric challenges. First, the lack of an explicit functional form of the hedonic price function led to various functional forms being used, including non-parametric approaches to estimating the price function. A second challenge involved recovering underlying demand curves for specific characteristics. Without structural restrictions or the ability to observe the same households in different markets or at different times, James N. Brown [1982] argued that the hedonic model is unable to evaluate how non-marginal changes in house attributes influence prices in a specific application.

In the estimation of the hedonic house price function, it is tricky to identify the unobserved locational attributes. We cannot model it explicitly, and therefore, cannot eliminate omitted variable bias. While a more complex model specification can alleviate omitted variable bias, Maureen L. Cropper [1988] found that omitted variable bias is actually greater in complex functional forms such as the quadratic Box-Cox compared to simpler linear approximations. They solved for a hedonic price equilibrium by solving an assignment problem using a sample of homes from Baltimore and a set of different specifications for the 'true' unknown preference function. As a result of this finding, the majority of subsequent hedonic papers rely on relatively simple linear models.

Since the omitted variable bias cannot be solved directly by complex parametric model specifications, a common way to get around the omitted variable bias is by estimating treatment effects via a regression discontinuity design (RDD). RDD is a quasi-experimental design that isolates treatment effects by comparing groups immediately above and below a threshold. Black [1999] used RDD to analyze the marginal effects of school zones on house prices by comparing houses with similar locational features at the borders of different school zones.

Another way to estimate treatment effects is through machine learning methods, the approach that we take. Varian [2014] reviewed a list of machine learning models and proposed applying them to economics. As Varian pointed out, these models are good at prediction and can be used to estimate treatment effects when the treatment is not perfectly randomized. Given a control and treatment group, if the model is predictive, a way to simulate the counterfactual is to predict the outcome for the treated group using a model trained on the control group. This yields a model that is predictive in the counterfactual world, one where the treatment group is not treated. The difference in the predicted and the actual can be interpreted as the treatment effect.

3.2.2 Fracking

Muehlenbachs et al. [2015] studied the housing market impacts of fracking using data from Pennsylvania. They used a triple-difference estimator with fixed effects to estimate the effects of adjacency to wells and groundwater contamination risk. They found large negative impacts of fracking on nearby homes with groundwater and smaller positive impacts on piped-water homes, which indicates benefits from royalty payments.

3.3 Model

3.3.1 Hedonic Pricing Model

For a hedonic model in a differentiated product market, the household maximizes utility subject to a budget constraint that is determined by the price function. In the housing market, the price function is the house price function with housing attributes and locational features as inputs. Suppose there are $i = 1, ..., I$ households, $j = 1, ..., J$ different types of houses, and $t = 1, ..., T$ time periods in a differentiated house market. Formally, utility is a function of housing characteristics, g_{jt}, locational public goods, q_{jt}, a numeraire good, c_{jt},

and unobserved household preference parameters, β_{it}, or

$$U_{it} = U_{it}(g_{jt}, q_{jt}, c_{jt}; \beta_{it}) \tag{3.1}$$

By maximizing (3.1) subject to the budget constraint in (3.2), where m_{it} is the income, $p_t(g_j t, q_j t, \psi_j t)$ is the hedonic price function for a house, and $\psi_j t$ is an indicator for time unobserved locational attributes. The price of c_t is normalized to unity.

$$m_{it} = p_t(g_{jt}, q_{jt}, \psi_{jt}) + c_t \tag{3.2}$$

Hedonic models assume a continuous price function where the model parameters can be estimated either through the structure of the utility function or by variations in the price function across markets. In order to distinguish the preferences from the equilibrium price function, there are two ways to add structure to the problem and identify the parameters. The first is assuming a specific functional form for preferences. The second assumes that there is a continuous choice set in order to satisfy the continuous price function assumption.

Benkard and Bajari [2004]'s identification strategy is to assume the preference parameters, β, vary with each household and that the choice set is continuous. Each household selects a different choice from the continuous choice set. Assuming well behaved preferences, the unobserved β is estimated as the value that rationalized the observed choice.

3.3.2 Empirical Specification

The ability of the hedonic model to measure non-market benefits arises through the inclusion of measures for the non-market services as elements in c. We highlighted this group by labeling them as q in equation (3.2). Suppose there are $i = 1, ..., I$ households and $j = 1, ..., J$ different types of houses and $t = 1, ..., T$ time periods in a differentiated house market. There are k housing characteristics $(g_1, \cdots, g_k)$ and n locational public goods $(q_1, \cdots, q_n)$. The hedonic regression of this problem can be formulated as follows assuming Cobb-Douglas

utility

$$U_{it} = \beta_{i1} log(g_{j1}) + \cdots + \beta_{ik} log(g_{jk}) + \gamma_{i1} log(q_{j1}) + \cdots + \gamma_{ik} log(q_{jn}) + \beta_{i\chi} log(c_t) \tag{3.3}$$

subject to the budget constraint in (3.2).

Using the first order condition and normalizing the parameter for fracking γ_1 with respect to the numeraire, the marginal willingness-to-pay for fracking is

$$\bar{\gamma}_{i1} = (\frac{q_{j1}}{c_i}) \frac{\partial p}{\partial q_{j1}}, \tag{3.4}$$

which is the partial effects of q_{j1} on price multiplied by a constant. This is a closed form solution and straightforward to estimate.

Therefore our empirical strategy is to estimate the following price function

$$log(p_{ijt}) = \sum_{k'=1}^{k} \beta_{ik't} g_{jk't} + \sum_{n'=1}^{n} \gamma_{in't} q_{jn't} + \theta_{ijt} \psi_{jt} + u_{ijt}, \tag{3.5}$$

where g_{jt} is the housing characteristics, q_{jt} is the locational public goods, ψ_{jt} is the unobserved locational attributes that vary with time.

In machine learning models, there is no clear way to estimate treatment effects in the absence of coefficients on predictors. While these models provide rankings of variable importance, they do not estimate direct effects of variables on the outcome. Our estimate of $\bar{\gamma}_{i1}$ is as mentioned earlier, the residual from a model trained on the control group.

A similar approach to dealing with estimating treatment effects with repeated cross sections is propensity score matching where treated observations are matched to a control observation based on a score measuring similarity based on covariates. Tree models differ in the sense that the matching is done directly on the covariates. Covariate are matched to regions within the tree model and assigned the average outcome of observations in that region.

3.4 Data

We collect house, well and demographic data from various sources and process them using different tools. Especially for the raw text description of houses and the curbside images, we utilize machine learning models to convert them into simpler variables.

3.4.1 Summary of Data

Well data was collected from Pennsylvania Department of Environmental Protection (PADEP) for the years 2000 - 2012. The reported data has the location of each well and the gas and oil production for each year. A majority of residential areas in Pennsylvania are serviced by piped water while other parts rely on groundwater. This geographic information of household source of water was also obtained from PADEP.

Census data was collected for 2000 and 2010. Data on the income, educational attainment, population size, and various other neighborhood characteristics were collected for each census tract in Pennsylvania.

Property information was collected from Zillow. We included properties from Pennsylvania that were listed as Single Family Houses with last sold dates between 2000-2012. Figure 3.1 shows the location of the properties in the dataset relative to the Marcellus Shale formation.

Property related variables collected include the structured and unstructured characteristics from the home details page on Zillow for each property. The structured characteristics are physical attributes of the property such as number of bedrooms, number of bathrooms, square footage, lot square footage, basement size, etc. Other structured variables such as external house material, type of cooling, floor covering materials, extra features, etc. are treated as categorical variables in our analysis. The unstructured characteristics are the paragraphs describing each home and the curbside views of the homes posted on the website.

Each property was mapped to its closest schools (elementary, middle, and high school),

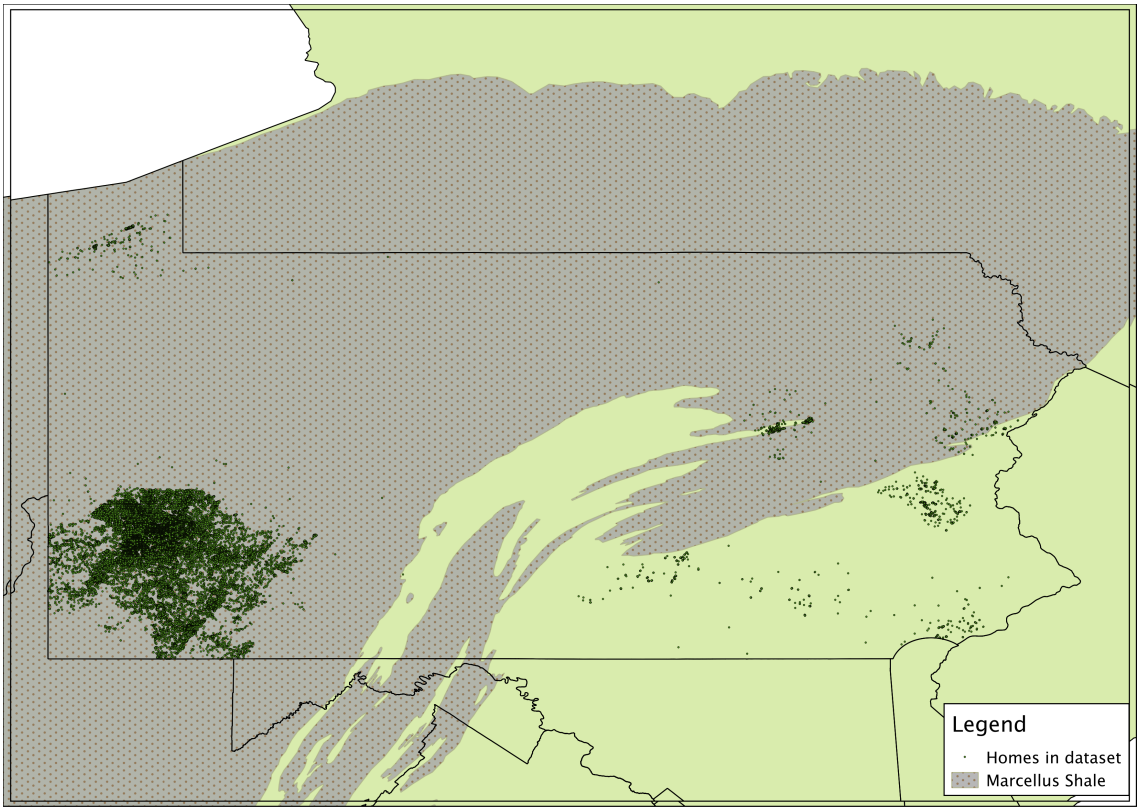

Figure 3.1: Map of Pennsylvania Homes in Dataset

and data on the quality of those schools on a scale of 1-10 were collected from GreatSchools. Separately, properties were mapped to census tracts and the corresponding neighborhood characteristics. The distance to the closest well pad for a property was based on Euclidean distances using the longitudes and latitudes of the properties and the well pads.

Table 3.1 is a summary of the structured home variables and the neighborhood characteristics grouped by distance to the closest fracking well. The average house prices differ in the magnitude of $6,000 among the three distance groups. However, comparisons of the home characteristics of the three groups show that they do not vary significantly. Thus, we found no clear evidence that treatment is endogenous. The demographics of the households show a similar pattern as the house prices, where the 3-5km group has the highest median household income, percentage with bachelors and bachelors or highers, as well as house prices.

Table 3.1: Mean Summary Statistics of Homes

	<3km of well	3-5km of well	>5 km of a well
price	118,041	124,755	122,417
	(229,288)	(159,568)	(230,328)
bedrooms	3.1	3.2	3.1
	(0.8)	(0.9)	(0.8)
bathrooms	2.1	2.1	2.1
	(0.7)	(0.7)	(0.7)
sqft	1,620	1,619	1,598
	(819)	(676)	(700)
basement_sqft	495	500	447
	(235)	(190)	(217)
age at sale in years	62.8	71.9	69.1
	(33.6)	(24.4)	(25.8)
years since last remodel	48.7	55.1	52.1
	(29.8)	(29.9)	(28.3)
number of photos on zillow	10.7	10.1	10.4
	(9.0)	(8.7)	(8.4)
school rating (1-10)	5.9	5.1	5.0
	(2.6)	(2.9)	(2.7)
avg distance to school (km)	2.1	1.6	1.6
	(1.4)	(0.7)	(0.9)
median income	52,612	54,028	50,770
	(23,266)	(25,209)	(26,096)
% with bachelors	18.3	20.3	18.2
	(9.0)	(9.4)	(8.8)
% with bachelors or higher	29.5	36.1	31.0
	(17.2)	(20.7)	(20.1)
observations	3,476	3,991	166,093

* The data contains total 173,560 homes in PA.
* Standard deviation in parenthesis.

3.4.2 Text Processing

After we collected all the descriptive paragraphs of homes for the houses in Pennsylvania, we convert the raw texts into more meaningful variables for analysis. This section explains how we extract all the words from each house description paragraph and process them. Aside from structured features, the hedonic attributes of a house can be defined by unstructured texts that an owner posts to describe a house. We apply a bag-of-words model where

each paragraph is parsed into words. Each distinct word is a count variable indicating the frequency of that word. We eliminated words with three or less letters and words with a frequency of less than 50. We also stemmed the words to group singular and plural forms of words together. In the final dataset, each home has a set of words, or a bag-of-words, associated with it. An example can be seem in table 3.2 where a sentence describing a home is represented as a bag-of-words with a column for each unique word. All words occur once except for "basement" which occurs twice.

Table 3.2: Bag-of-words Representation

"Beautiful wooded lot on quiet street, plumbed for bath in basement. Large high basement."

basement	bath	beautiful	high	large	lot	plumbed	wooded	quiet	street
2	1	1	1	1	1	1	1	1	1

In a high dimensional learning problem, only some parts of an observation are important for prediction. For example, the information to correctly categorizing a house may lie in a handful of its words. The extraneous words could prove computational burdensome, so word regularization may be helpful. Possible methods for regularization include LASSO and other shrinkage models. Therefore it's natural to combine the technique of bag-of-words with our models. In table 3.8, the model results combining bag-of-words and the main machine learning models is reported. We will discuss the details of it in the Section 3.6.

3.4.3 Image Processing

There are different types of house images we collect - curbside, exterior, interior, view, etc.. After experimenting with all of the types, we focused on curbside images as they are more homogeneous to extract useful features for comparison. We use the entire next section to demonstrate how we estimate curbside appeal from curbside images.

3.5 Estimating Curbside Appeal from Images

This section explains how we extract features from each image to model curbside appeal. In our dataset, 23,733 homes posted curbside images, accounting for 14% of the data. We submit a small sample of these images to Mechanical Turk to obtain human rated curb appeal for each image. A model is built on this training set and used to predict the remaining dataset. We start with an introduction on how an image can be transformed into a data array. Then, we describe the features that we extracted and the results of using a gradient boosting model to predict curb appeal.

An image is an array of numbers that reflect the colors of the pixels. For a grayscale image with $X \times Y$ pixels, the corresponding array is $X \times Y$ where the value in each cell ranges from 0 - 255, or black to white, to indicate the color of the pixel. An example can be seen in figure 3.2. For a color image, the array is $X \times Y \times 3$ to reflect different values for each red, green, and blue (RGB) spectrum. Each pixel has coordinates (x, y) and an $I(x, y)$ function that maps the pixel coordinates to the color value of that pixel. For a grayscale image, $I(\cdot)$ returns one value while an RGB image has 3 values for each pixel - one for each color component: $I_R(x, y)$ $I_G(x, y)$, and $I_B(x, y)$. We will be using a variety of cylindrical coordinate representations such as hue-saturation-value (HSV).

We follow Datta et al. [2006] approach on estimating the aesthetic appeal of an image but model the curbside appeal of a house instead. We collect 43 of the 56 image features introduced in their paper depending on the relevance to curbside appeal. The bulk of their features represent the textures and colors within segmented parts of an image. Table 3.3 summarizes all the features. We first offer a brief introduction on wavelet analysis and segmentation.

Wavelets can be used to analyze textures within images to indicate what types of materials are present in the image (i.e. brick vs. concrete homes). Wavelets smooth and capture information about signals, which for images correspond to the magnitude of color contrasts across pixels. They are used to compress the size of images while maintaining the most

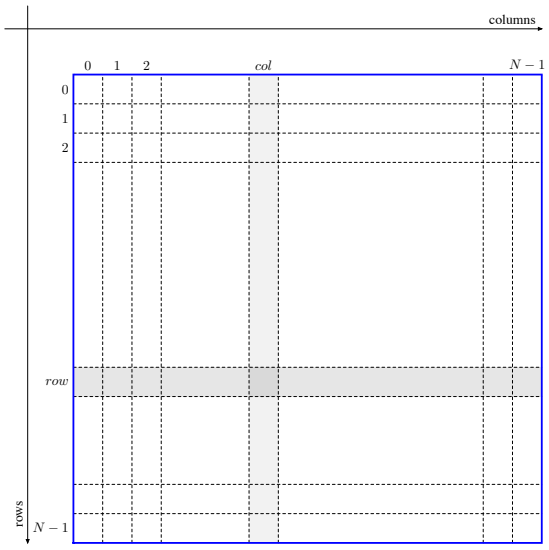

Figure 3.2: Representing an Image as a Matrix

important details. For our analysis, we calculate the average changes in groups of pixels, also called wavelet coefficients, to capture the intensity of color changes in an image. The pattern of these changes represent a texture. We summarize the textures within each image averaging the wavelet coefficients to use as image features. A more detailed and mathematical explanation on how wavelets are used in this context is provided in appendix C. Livens et al. [1997] also discuss this technique in detail.

Segmentation is the process of grouping or clustering similar pixels together, often to segregate objects within an image. It is implemented using the K-Means algorithm, which is formally defined in appendix B. The ability for clusters to accurately summarize the data depends heavily on the number of clusters chosen. For example, if the true data have 3 clusters but K is chosen to be 2 then the final clusters would not be representative of the true structure of the data. The same is true if there are no true clusters in the data but K-means is still performed with K>1. To provide a more flexible analysis, we created an RGB histogram with 64 bins for each image and then choose K to be the number of peaks that surpass 500 pixels. Thus, the chosen K is image dependent. Next, we ran K-Means

on the pixels based on their colors using the chosen K and kept only the 5 largest segments of each image to analyze. An example of this is shown in figure 3.3 and 3.4. The size (or number of pixels) and the average H, S, and V values for each segment define features f21-f43.

Table 3.3: Summary of Image Features

Features	Description
From Datta et al	
f1	average pixel intensity
f2	colorfulness
f3	average saturation
f4	average hue
f5-f7	average hue, saturation, and value for the center 1/3 of the image
f8-f18	Daubechies wavelet transform values to measure textures
f19	$X + Y$ size of image
f20	X/Y aspect ratio
f21-f43	segmentation

Figure 3.3: Original image

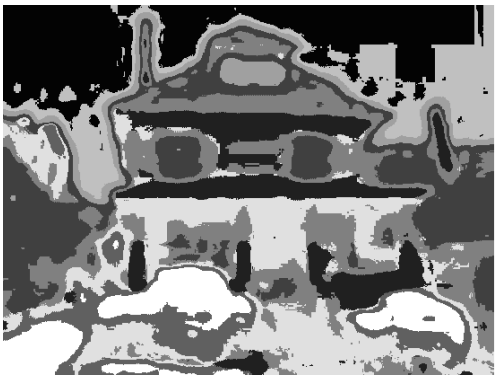

Figure 3.4: K-means on image

Figure 3.5: K was selected to be 9 based on the number of peaks in a RGB color histogram. On the left, figure 3.3, is the original image. On the right, figure 3.4 is the segmented image after K-means on color.

Next, we will discuss how we modeled curb appeal. We recruited 44 Mechanical Turk workers to rate the curbside appeal of 2,774 house images on a scale of 1 to 10 with 10 being the most appealing. Each photo was rated by 2 unique workers the average of the two was used as the final rating. To avoid having to obtain a large set of human-rated images, we submitted a sample of the images to Mechanical Turk to obtain a training set. Then we trained that sample using a gradient boosting model with 43 extracted image features and used the final model to predict the curb appeal for all the images.

Datta et al. [2006] classify the images into binary groups: low and high aesthetic appeal and then use a support vector machine model to classify the images. We select to use a gradient boosting regression model to predict curb appeal. While support vector machines are strong classification models, they do not work well with high dimensional features. Furthermore, binary classification would eliminate a lot of possibly explanatory variation in curb appeal. Since curb appeal ratings are ordinal, we use a regression model instead of a multiclass classification model.

A summary of the curb appeal rating distribution of the training set and the predicted set are in table 3.4. The predicted curb appeals are much more conservative than the training set since machine learning models often have trouble predicting outliers. Nevertheless, the predictions reasonably separate images with low curb appeal from those with high as seen in figure 3.6 and 3.7 as neither of these images were in the training dataset. We use the predicted curb appeal ratings, including the predicted values on the training images for easy comparison, as a structured variable in our hedonic analysis.

Including the curb appeal estimate is our hedonic model resulted in reasonable results. In the gradient boosting model, curb appeal was the 34th most important variable out of 1,848 variables.

One way to interpret how curb appeal affects the predicted price is to look at a partial dependence plot. This is a graph of $\hat{F}(x)$ as a function one of the inputs conditioned on the other inputs. In other words, a partial plot for input variable x_i would be a plot of $\hat{F}_{x_{-i}}(x_i) = \hat{F}(x_i|x_{-i})$. This is estimated from the training data by averaging over the con-

Figure 3.6: Predicted rating of 3.7

Figure 3.7: Predicted rating of 7.4

Table 3.4: Summary of Curb Appeal Ratings

Images	Mean	Min	Max	N
Mechanical Turk	6.379	1.000	10.000	2,774
	(1.516)			
Predicted data	6.332	3.662	7.730	23,733
	(0.475)			

Notes: Standard deviations in parenthesis

ditioned variables $\bar{F}(x_i) = \frac{1}{N}\sum_{i=1}^{N}\hat{F}(x_i, x_{-i})$. In general, partial dependence plots are the most useful when the output variable is strongly dependent on x_i. Nevertheless, these plots can still give useful insights by showing how a variable was split in the tree.

We use a partial dependence plot of curb appeal to show an estimate of how log prices vary with curb appeal. Each jump in the graph corresponds to a split in the tree, and the predicted value is the average of the terminal nodes following that split. Figure 3.8 shows that in the data set, 6.5 is the most common curb appeal scores. It also shows that the difference in prices between homes with curb appeal forms a step function.

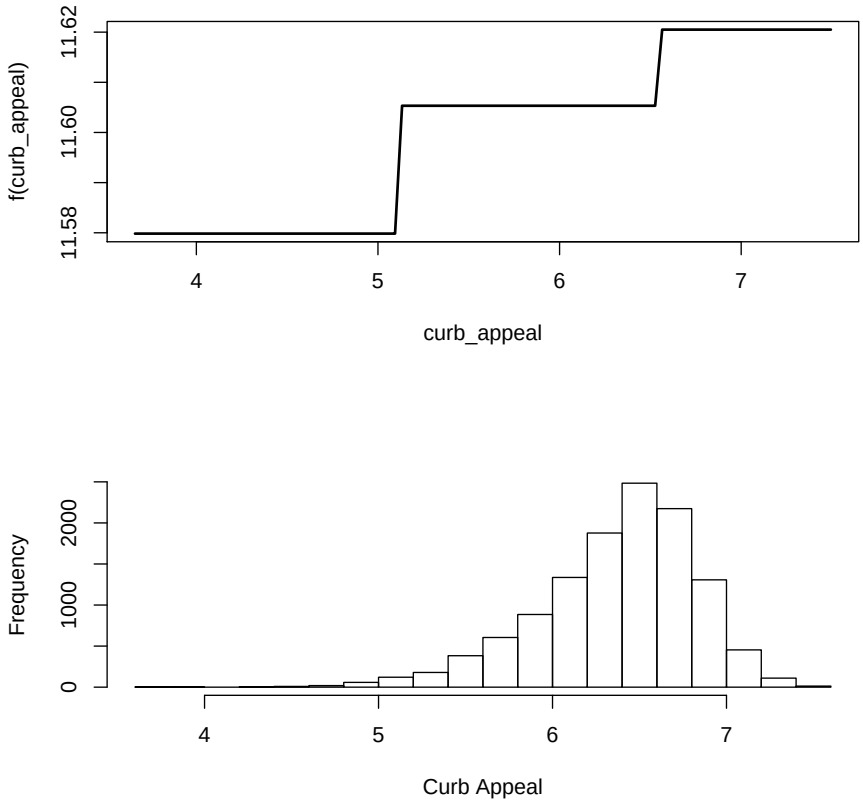

Figure 3.8: How Predicted Log Price Varies With Curb Appeal

3.6 Results

In the first part of this section, we carry out a similar exercise as did Muehlenbachs et al. [2015]. We make conclusion of how proximity to a well and water source may impact housing prices. We also demonstrate the effectiveness of including curb appeal in the regression. In the second part, we then have a competition among four different models using out-of-sample error as the comparison criteria. We show that the machine learning models out-perform traditional least squares regression. We then use the machine learning models to predict counterfactual outcome of the treatment group and compute treatment effects.

3.6.1 Curb Appeal

In the top panel of table 3.6 we present results from the regression of log price on the number of well pads within 1-5 km of a house, its interaction with water source, the same set of variables for 20 km, property features and census tract-year fixed effects. It shows when near a well, adding one more well in close proximity has a significant negative impact on the price, no matter how large the proximity (K) is. Moving to a larger proximity (from 1 km to 5 km), we see a pattern of a smaller negative impact from wells (likely because the negative effects of wells such as noise and light pollution are smaller from father wells). Having one more well while dependent on ground water has a negative impact on the house prices in almost all cases except when K is very small ($K \leq 1$ km), as we don't have any observations where homes are closely located next to wells and use ground water. On the other hand, adding one more well in 20 km only has a small positive impact and ground water has no significant impacts within 20 km distance.

In the bottom panel of table 3.6 we have the results very similar to th top panel with the addition of curb appeal variable. By adding curb appeal to the pool of right hand side variables, we improved the overall fit of models in all cases while maintaining a similar set of signs. Thus we show the effects of curb appeal on log price while controlling for the number of wells within certain distances of the home, water source, property features and location-time fixed effects.

Table 3.5: Summary Statistics of Sample

	Mean	Std. Dev.
Log Price	11.310	0.967
Curb Appeal	6.335	0.471
Observations	21691	

Table 3.5 gives a quick summary of the log price and curb appeal in the dataset. A quick interpretation from table 3.5 and coefficients of curb appeal in table 3.6 shows that

the effects of curb appeal, translating to dollar terms, is $15127.

3.6.2 Treatment Effects of Fracking

To estimate the treatment effects of fracking on house prices, we train a model on a control group defined as a home that is more than 15km away from the closest well. 25% of this control group is further split into a test set with the remaining 75% used as the training sample yielding 80,509 observations. Hence the treatment group contains homes that are within 15km distance from any well.

Model Comparison

We tested four specification of models: OLS, LASSO - OLS, random forest, and gradient boosting. LASSO - OLS processes all the covariates in LASSO to extract the important ones and drop irrelevant ones and then feeds the kept covariates to a linear regression. The rationale for using LASSO-OLS instead of LASSO is that the former produces a smaller bias compared to the latter (Chernozhukov [2013]). Gradient boosting and random forest simply takes all the covariates, extracts important features. [1]

We first present a baseline result from ordinary least square regression of log price on the set of structural variables in table 3.7 as a summary of how the variables are related to price. In order to avoid rank deficiency caused by sparse data where there are too many zeros in the dependent variables, we run all the covariates through a function to remove collinear variables. The function recursively calculates a variance inflation factor (VIF) for all the variables and removes the variable with the highest VIF at each iteration.[2] Using

[1] In appendix A we give an introduction to the machine learning techniques we used in this paper. Running models with this many covariates requires significant computing power. In appendix B, we give a brief introduction to the tools and algorithms to speed up the computational tasks presented in this paper.

[2] Suppose we have k explanatory variables, run each X_i as an ordinary least square of all the other variables and calculate the VIF factor with the following formula:

$$VIF_i = \frac{1}{1 - R_i^2}.$$

Table 3.6: Log Price on Well Pads

	$K \leq 1$ km	$K \leq 2$ km	$K \leq 3$ km	$K \leq 4$ km	$K \leq 5$ km
Panel A. Census tract-year fixed effects					
Pads in K km	-0.058***	-0.007***	-0.028***	-0.021***	-0.016***
	(0.004)	(0.001)	(0.002)	(0.002)	(0.003)
(Pads in K km)× GW	.	-0.198	-0.182*	-0.126*	-0.068
	.	(0.124)	(0.089)	(0.061)	(0.059)
Pads in 20 km	0.002	0.002	0.003**	0.004**	0.004***
	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)
(Pads in 20 km)× GW	-0.048	-0.031	-0.024	-0.026	-0.033
	(0.031)	(0.033)	(0.033)	(0.032)	(0.029)
Property attributes	Yes	Yes	Yes	Yes	Yes
Census tract-year effects	Yes	Yes	Yes	Yes	Yes
Observations	21691	21691	21691	21691	21691
R^2	0.580	0.580	0.580	0.580	0.580
Avg. pads in K km	0.009	0.044	0.085	0.135	0.210
Avg. pads in 20 km	1.922	1.922	1.922	1.922	1.922
Panel B. Adding Curb Appeal					
Pads in K km	-0.059***	-0.009***	-0.028***	-0.022***	-0.017***
	(0.002)	(0.000)	(0.001)	(0.002)	(0.002)
(Pads in K km)× GW	.	-0.180	-0.175*	-0.120**	-0.056
	.	(0.113)	(0.079)	(0.053)	(0.065)
Pads in 20 km	0.003**	0.003**	0.004***	0.004***	0.005***
	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)
(Pads in 20 km)× GW	-0.035	-0.019	-0.011	-0.014	-0.021
	(0.031)	(0.033)	(0.032)	(0.031)	(0.027)
Curb Appeal	0.170***	0.170***	0.170***	0.170***	0.170***
	(0.003)	(0.003)	(0.003)	(0.003)	(0.003)
Property attributes	Yes	Yes	Yes	Yes	Yes
Census tract-year effects	Yes	Yes	Yes	Yes	Yes
Observations	21691	21691	21691	21691	21691
R^2	0.589	0.589	0.589	0.589	0.589
Avg. pads in K km	0.009	0.044	0.085	0.135	0.210
Avg. pads in 20 km	1.922	1.922	1.922	1.922	1.922

Robust standard errors are clustered by census tract.
***Significant at the 1 percent level.
**Significant at the 5 percent level.
*Significant at the 10 percent level.

the training sample of 80,509 observations and 172 explanatory variables (154 variables after removing collinear ones), the regression results of log price on a set of covariates pruned by removing high VIFs are presented in table 3.7. We use VIF > 5 as a rule to remove covariates. Table 3.7 only displays a subset of the covariate estimates where the variables are ordered by absolute t values. The estimates of the variables are interpretable as the percent marginal effect of the house feature or demographic characteristics on price. Roughly speaking, the signs of the variables are intuitive. We use VIF only for interpretation. In the final comparisons, we run a standard OLS including all the variables.

Table 3.7: OLS Regression Results (after removing collinear covariates)

Log Price	Estimate	Std Err	t Value	$\Pr(>\|t\|)$	
(Intercept)	9.8940	0.0780	127.0120	0.0000	***
Number of Bath	0.1410	0.0050	31.1670	0.0000	***
Square Footage	0.0000	0.0000	20.2420	0.0000	***
% Black	-0.0060	0.0000	-13.7190	0.0000	***
Age at Sale	-0.0010	0.0000	-13.4980	0.0000	***
Fireplace	0.1110	0.0090	12.6090	0.0000	***
% Graduate Education	0.0090	0.0010	11.5780	0.0000	***
Curb Appeal	0.1000	0.0090	11.1700	0.0000	***
External Material: Wood	-0.1140	0.0110	-9.9560	0.0000	***
% Bachelor Education	0.0080	0.0010	8.8290	0.0000	***
School Rating	0.0190	0.0020	8.4540	0.0000	***
...					
Adjusted R-squared	0.6479				
Number of Variables Used	171/197				
Observation	11845				

Machine learning methods tend to overfit on the training data so looking at the root mean-squared error of a test set or out-of-sample set is useful to compare the prediction accuracy of each model. The root mean squared error to the OLS, LASSO-OLS, gradient

Do the same for all k variables and remove the variable with highest VIF. Repeat the process for the rest $k-1$ variables until the highest VIF is less than the threshold (VIF< 5 in this paper).

boosting, and random forests are shown in table 3.8. Each model is done with two different specifications: structured variables (includes curb appeal) and structured variables with bag-of-words indicators. The table also shows how many variables were selected to be used as LASSO and boosting are both effective shrinkage models with boosting eliminating a majority of the variables when words are included.

Random forest was run with 800 trees and 1/3 of the variables are possibly used to split each tree. We selected the minimum number of observations in a terminal node to be 50. Gradient boosting was run with 1000 trees, a shrinkage parameter of 0.3, minimum number of observations in a terminal node to be 100, and splits on 3 variables in each tree. The optimal number of trees to predict on is selected through a K-fold cross validation with K=10, a common choice in the machine learning literature. Among the four models, boosting obtains the lowest error with random forest having the second lowest. The two tree-based models perform approximately the same but outperform OLS and LASSO-OLS. The addition of the words improve all of the models with boosting being the least sensitive to which features were included in the model.

The results of the tree-based models are harder to interpret than those for OLS and LASSO-OLS. In an attempt the peer into these models, we show the variables that influence each model the most. For boosting models, this is called the relative influence of the variables and is measured by how much splitting on a variable improves the mean squared error. Figure 3.9 shows the 10 most influential variables. This bar chart shows that the number of bathrooms and age of the house are the most influential. The results are sparse in the sense that a select few variables drive a majority of the model. The quality of school has a tenth of the influence as number of bathrooms and age of the house but ranks yet ranks as the 8th more important variable. For random forest, the comparable statistic is variable importance.

Table 3.8: Model Comparisons

Model	# Variables selected*	RMSE-OOB sample	Standard Error of RMSE
OLS			
structured	172/172	0.493	0.002
+ bag-of-words	1848/1848	0.480	0.002
LASSO-OLS			
structured	156/172	0.493	0.001
+ bag-of-words	1251/1848	0.478	0.003
Random forest			
structured	172/172	0.441	0.001
+ bag-of-words	1848/1848	0.440	0.003
Boosting			
structured	77/172	0.435	0.003
+ bag-of-words	255/1848	0.431	0.003

*based on 172 variables including curb appeal in structured model and 1,848 variables in model with bag-of-words

Treatment Effects

Using best performing models, random forest and boosting, we estimate the counterfactual house prices of the treatment group. The average treatment effects from random forest and boosting for different distances are shown in table 3.9. The two models predict very similar treatment effects.

Homes within 1km of a well appear to have no treatment effect. We suspect that this is the net effect of negative environmental harm and royalty payments because homes 1-1.5km from a well have an increase of an estimated about 5% as estimated in the boosting and random forest models. And the perceived harm is likely to decrease with distance. The predicted residuals converge to zero as the distance increases, consistent with the impact of drilling decreasing in distance. At a distance of 10km, the effect of fracking goes to zero. We plot the boosting model results in figure 3.10.

We follow Bajari et al. [2015] to compute the standard error of the predicted treatment effects. The three prediction models in this paper will be asymptotically normal, and we

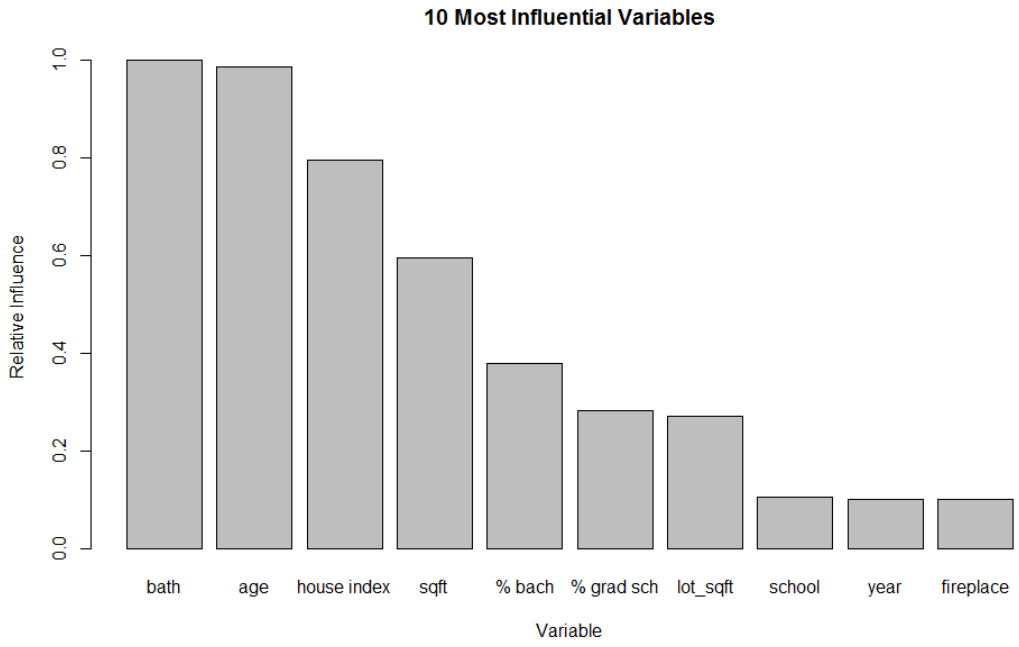

Figure 3.9: Variable Influence

subsample the data to compute standard errors. By holding the model parameter from the training data set fixed, we take 50% subsamples of the data and feed them into the fixed model 1,000 times. Then we calculate the standard error of the predicted treatment effects from this exercise. Both table 3.8 and table 3.9 uses this technique to compute standard errors.

3.7 Conclusion

In this paper, we apply hedonic price regression in a non-traditional way. We collect a very unique and comprehensive data set on houses in Pennsylvania and integrate the econometrics of hedonic models and machine learning to estimate the marginal price of fracking on houses. This paper demonstrates useful applications of machine learning techniques as well as tools to process raw text and images as input variables. We construct a new variable for curbside appeal and estimated the marginal price for curb appeal. Our results show how

Table 3.9: Average Treatment Effects by Distance From a Well

	Average Treatment Effects on Log Price							
Dist. in km	1	1-1.5	1.5-2	2-3	3-4	4-5	5-10	10-15
Random forest	-0.032	0.047	0.029	-0.008	-0.003	-0.014	-0.006	0.001
	(0.022)	(0.018)	(0.016)	(0.011)	(0.011)	(0.010)	(0.004)	(0.003)
Boosting	-0.014	0.045	0.034	0.013	0.005	-0.007	-0.001	0.001
	(0.022)	(0.017)	(0.015)	(0.012)	(0.010)	(0.010)	(0.004)	(0.003)
N	345	543	719	1,442	1,554	1,817	12,679	16,938

Note: Standard errors in parenthesis.

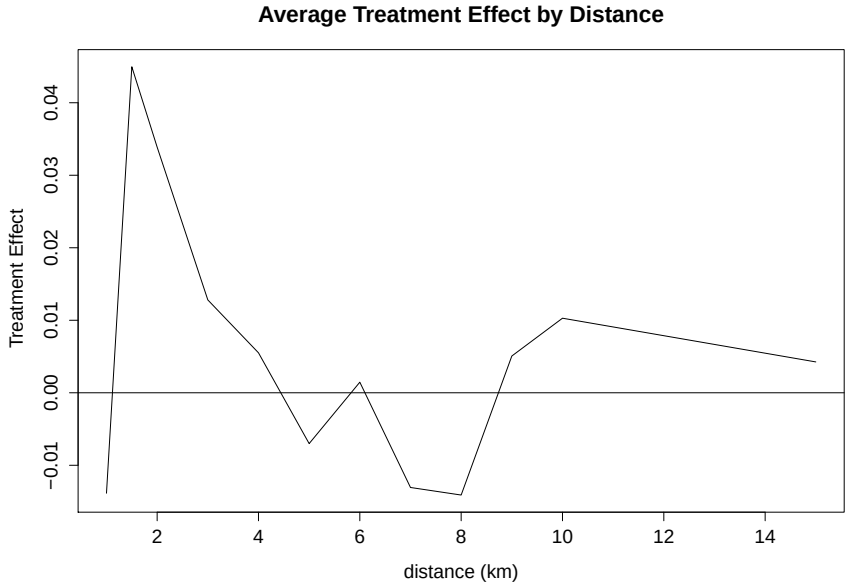

Figure 3.10: Average Treatment Effect

prediction accuracy can be significantly improved when flexible nonlinear tree models and feature selection are used.

We estimate the changes in house prices due to fracking and find that houses very close to wells are likely to be monetarily compensated for any environmental harm while homes farther away benefit from price increases of up to 5%. These increases suggest that envi-

ronmental harm decreases the farther away from a well a home is. Since these estimates are generated from prediction models, they can also be interpreted as predictions for future impacts of fracking on house prices.

This paper shows how to extract features from unstructured data like words and images for use in hedonic regressions. The main tools we use are from the machine learning literature. We reduce dimension of the raw data (words and images) when they are too big to handle and still achieve improvements in model performance. The methods are straightforward yet powerful.This is a demonstration of how machine learning techniques could benefit empirical work in the social sciences.

Appendix A

MACHINE LEARNING TECHNIQUES

In this section, we provide a brief introduction to each of the machine learning techniques that we use to model house prices as a function of house features. These methods were chosen because they are effective feature selection methods and predict out of sample well despite the presence of irrelevant variables and multicollinearity. The first method, LASSO, can be used within an ordinary least squares framework. The other two methods, gradient boosting and random forest, involve regression trees. Readers familiar with machine learning models could skip this section.

A.1 LASSO

The LASSO (least absolute shrinkage and selection operator) is a regularization method that induces subset selection and shrinkage by setting some covariate coefficients to zero and thus prevents overfitting, according to Tibshirani [1994]. It improves an ordinary least squares model in two ways: reducing the variance of predictions and simplifying the interpretation of the model by eliminating less influential variables.Consider the standard linear equation

$$y_i = \beta_0 + \sum_{j=1}^{p} x_{ji}\beta_{ji} + \epsilon_i \text{ for } i = 1, 2, ..., N \tag{A.1}$$

where y_i is the outcome variable, x_{ij} is variable j for observation i, β_{ji} is the unknown parameter, and ϵ_i is the error term. With an ordinary least squares model, $\hat{\beta}$ minimizes the residual sum of squares error. Under LASSO, the penalty is added in the form of a constraint

and the optimal β becomes

$$\hat{\beta}^{lasso} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \text{ subject to } \sum_{j=1}^{p} |\beta_j| \leq t \tag{A.2}$$

The threshold t depends on the coefficient on the constraint which is a tuning parameter usually chosen by five-fold cross-validation. Since LASSO estimates are nonlinear and non-differentiable functions, accurate standard errors are difficult to compute.

LASSO biases the estimated coefficients towards zero. We follow Chernozhukov [2013] approach and run OLS using only the regressors with nonzero coefficients from the LASSO estimates to mitigate the biasness towards zero. Standard errors can easily be computed and estimated coefficients are less bias.

A.2 Regression Tree Models

Both gradient boosting trees and random forecast are ensemble tree models. A tree model recursively divides an input space into a set of regions and then fit a constant (usually an average) to each one. Assuming a continuous outcome, y_i, for observations $i = 1, ..., N$ with p inputs, $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$, a tree is defined as:

$$f(x) = \sum_{j=1}^{J} \gamma_j I(x \in R_j) \tag{A.3}$$

where $f(x)$ is the predicted outcome for an observation x. The input space is divided into J regions $R_j, j = 1, 2, ..., J$ with I as an indicator function for whether x is in a particular region j. γ_j is a constant and chosen such that the sum of squares, $\sum(y_i - f(x_i))^2$, is minimized. Thus, the optimal γ_j that minimizes this error is the average of the outcomes of all the observations within region j.

The tree model needs to select optimal cutoffs to split the input space. Splits are done with a single variable at a chosen threshold. For example, if variable k is selected as the

splitting variable at a threshold s then a region would be split into two half-planes:

$$R_1(k,s) = \{x|x_k < s\} \text{ and } R_2(k,s) = \{x|x_k >= s\} \tag{A.4}$$

The variable and threshold are chosen such that the sum of squares in the two regions combined are minimized. Then splits are chosen on each of the separate regions and all the subsequent regions. The algorithm follows a greedy approach in the sense that once a split is made, all following splits do not alter previous splits. The final regions are called leaves or terminal nodes. A chosen limit on the number of nodes or minimum number of observations remaining in each region are usually set as stopping criteria. In general, the more nodes and fewer observations left in each region, the more accurate the fit on the training set.

An example of a tree model for house price (in \$1,000) as a function of its attributes with 1,000 observations is shown in figure A.1. The depth of the tree here is 3, and the total number of nodes is 11. The house attributes that split the data into two regions are listed with the numbers of observation in the regions as well as the proportion of the data. Notice that regions can be split on the same attribute multiple times. The algorithm splits until a specified number of observations are left in each terminal node. The terminal nodes in this tree define the final regions and the average house price for homes in that region is given in each terminal node.

While trees are easy to interpret and flexible, they tend to overfit, have big biases, and produce inaccurate out of sample predictions. To mitigate these problems, we explore two ensemble tree models in the following subsections: gradient boosting and random forest.

A.3 Gradient Boosting

Friedman [2001] introduced gradient boosting as an additive ensemble model where a weighted sum of base models are added together. Using tree models as base models, a new tree is added to a weighted sum of all previous trees at each iteration. We use a sum of squares loss function $L(y, F(x)) = -\frac{1}{2}(y - F(x))^2$, but a variety of different loss functions can be used.

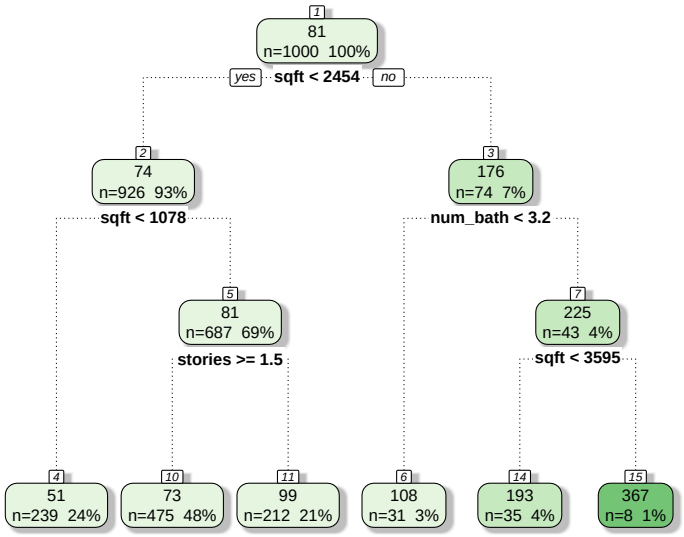

Figure A.1: An example of a tree with 1,000 observations

We will present the basic idea of the gradient boosting algorithm and then introduce some methods used in practice that further improve performance. The model is initialized to be the average of the outcomes, $\bar{y}$ for N observations. At each subsequent iteration, the model parameters are chosen to correct the error from the previous iteration. Suppose there are M iterations, then the model is defined as:

$$F_M(x) = \sum_{m=0}^{M} f_m(x) \tag{A.5}$$

where

$$f_m(x) = -\rho_m h_m(x). \tag{A.6}$$

Equation (A.5) can be rewritten as

$$F_m(x) = F_{m-1}(x) + \rho_m h_m(x) \tag{A.7}$$

where $h_m(x)$ is a regression tree model fit to the residuals, $g_m(x)$, from the previous iteration which is also equal to the gradient of the loss function. ρ_m is the step size or weight on a model and chosen to minimize the loss at iteration m. In other words,

$$\rho_m = \arg\min_{\rho} \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \rho h_m(x_i)) \tag{A.8}$$

and

$$g_m(x) = y - F_m(x) = \left[\frac{\partial L(y, F(x))}{\partial F(x)}\right]_{F=F_{m-1}} \tag{A.9}$$

This is similar to the gradient descent algorithm for a minimization problem with $-g_m$ as the negative gradient and ρ_m as the "line search" along the gradient. This is why it is called a gradient (descent) boosting model. In practice, $h_m(x)$ is an approximation to the error, $g_m(x)$, and takes the form of a regression tree with J cuts as defined in the section A.2. The step by step algorithm is in table A.1. For prediction models, gradient boosting can be computed with a shrinkage parameter, ν, to prevent overfit. ν effectively shrinks the influence of each additional tree. Now, equation (A.7) is replaced with

$$F_m(x) = F_{m-1}(x) + \nu\rho_m h_m(x) \tag{A.10}$$

The gradient boosting model iteratively corrects the errors in the model and then applies a shrinkage parameter to the errors to prevent them from overfitting at the next iteration. This prevents the errors in one iteration from having too much influence on the final predictions. In practice, the optimal model for out-of-bag sample predictions will depend on both the choice of M and ν. Greater values of ν require lower M and vice versa. It is common to set $\nu = 0.1$. An additional way to mitigate overfit is to subsample the data at each iteration m.

This reduces the variance of the predicted outcome.

Table A.1: Gradient Boosting Algorithm

1. Initialize the model $F_0(x) = \underset{\gamma}{\operatorname{argmin}} L(y_i, \gamma)$
2. For m=1 to M do:
   (a) Compute
   $$g_m(x_i) = \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x)}\right]_{F=F_{m-1}} \text{ for } i = 1, ..., N$$
   (b) Fit a regression tree, $h_m(x)$, to the errors, $g_m(x)$, with J cuts
   (c) Compute
   $$\rho_m = \arg\min_\rho \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \rho h_m(x_i))$$
   (d) $F_m(x) = F_{m-1}(x) + \rho_m h_m(x)$
3. The predicted outcome for observation x is $F_M(x)$

A.4 Random Forest

The second tree-based model is Breiman [2001]'s random forest. Random forest averages the predictions from a set of trees formed from bootstrapped samples to decrease the variance of the predictions. A tree is represented as $f(x)$ in equation (A.3) and has the same form as discussed earlier with the same parameters. A random forest with B number of bootstrapped trees can be written as:

$$F(x) = \frac{1}{B}\sum_{b=1}^{B} f_b(x). \tag{A.11}$$

Random forest bootstraps the training data B number of times by randomly selecting m variables from p input variables each time before another split is made. Selecting only a subset of variables decorrelates the trees and prevents overfitting by decreasing the variance of the average, which is predicted value. With independently distributed variables, correlation

ρ, and variance σ^2, the variance of the average is

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \tag{A.12}$$

As B increases, the second term goes to zero, leaving the first term. Thus, as the correlation, ρ, decreases, the entire term goes to zero. In practice, m is usually set to $\frac{p}{3}$ for regression trees. Random forests can also reduce the variance of predictions by averaging bootstrapped samples of regression trees. Hastie et al. [2009] argue that random forest is computationally scalable and able to deal with irrelevant inputs. We offer the algorithm in table A.2.

Table A.2: Random Forest Algorithm

1. For b=1 to B:
   (a) Draw a bootstrapped sample of size N from the training sample.
   (b) Grow a regression tree f_b using the bootstrapped sample. Stop when the minimum node size is obtained. At each terminal node,
      (i) Randomly select m variables from p total variables.
      (ii) Split on the best variable from the m variables selected
2. Output the ensemble of trees $\left\{f_b\right\}_1^B$.
3. The predicted outcome for an observation x is $F(x) = \frac{1}{B}\sum_{b=1}^{B} f_b(x)$.

Appendix B

K-MEANS

K-means algorithm:

1. Initialize K centers at random $\mu_1, \mu_2, ..., \mu_K$

2. Repeat until convergence

   (a) Ror i in $1 : n$ (assign each point to the closest μ_k)

   $$k(i) = \underset{k}{\operatorname{argmin}} \|x_i - \mu_k\|$$

   (b) For $k = 1 : K$ (recalculate the mean of each cluster C_k)

   $$\mu_k = \tfrac{1}{|C_k|} \textstyle\sum_{i \in C_k} x_i$$

The k-means algorithm obtains a local minimum of a loss function defined as

$$L(\Delta) = \sum_{i=1}^{n} \sum_{i \in C_k} \|x_i - \mu_k\|^2$$

where x_i is an observation i, C_k is cluster k, and mu_k is the mean of the points in cluster C_k, and Δ is the set of all clusters $\{C_1, C_2, ...C_K\}$.

Appendix C

WAVELETS FOR TEXTURE ANALYSIS IN IMAGES

For a signal $x(u)$, where u is the pixel's x-coordinate, the Haar wavelet is defined by

$$\psi(u) = \begin{cases} -\frac{1}{2} & -1 \leq u < 0 \\ \frac{1}{2} & 0 \leq u < 1 \\ 0 & otherwise \end{cases}$$

The color intensity $x(u)$ across pixels is filtered by the wavelet since for all values less than -1 and 1, all the values are set to zero when the signal is multiplied, or convolved, by the wavelet. An example of this transformation is seen in figure C.1. The usefulness of the wavelet for texture analysis is seem in the integral of the convolved signal $\psi(u)x(u)$. The integral is defined as

$$\begin{aligned} \mathcal{W} &= \int_{-1}^{1} \psi(u)x(u)du \\ &= -\int_{-1}^{0} x(u)du + \int_{0}^{1} x(u)du \end{aligned}$$

W is called a wavelet coefficient. Given this equation, it is clear that the wavelet coefficient is the difference between the averages of the two intervals. If $x(u)$ for $u < 0$ and $x(u)$ for $u > 0$ differ greatly, then the wavelet coefficient will be very high. In contrast, if the two sets of signals are similar, then W would be close to zero.

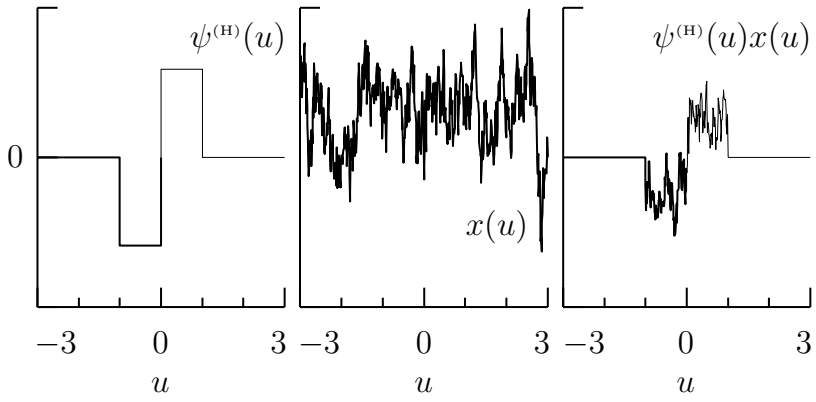

Figure C.1: How a Haar Wavelet Filters a Signal

BIBLIOGRAPHY

Alberto Abadie and Javier Gardeazabal. The economic costs of conflict: A case study of the basque country. *American economic review*, pages 113–132, 2003.

Donald WK Andrews. Asymptotic normality of series estimators for nonparametric and semiparametric regression models. *Econometrica: Journal of the Econometric Society*, pages 307–345, 1991.

Patrick Bajari, Denis Nekipelov, Stephen P. Ryan, and Miaoyu Yang. Demand estimation with machine learning and model combination. Technical report, Working Paper No. 20955, University of Texas at Austin, December 2015.

John M Bates and Clive WJ Granger. The combination of forecasts. *Or*, pages 451–468, 1969.

Alexandre Belloni and Victor Chernozhukov. L1-penalized quantile regression in high-dimensional sparse models. Technical report, MIT, November 2011.

Alexandre Belloni, Victor Chernozhukov, and Christian Hansen. Lasso methods for gaussian instrumental variables models. Technical report, MIT, February 2011a.

Alexandre Belloni, Victor Chernozhukov, and Lie Wang. Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806, 2011b.

Alexandre Belloni, Daniel Chen, Victor Chernozhukov, and Christian Hansen. Sparse models and methods for optimal instruments with an application to eminent domain. *Econometrica*, 80(6):2369–2429, 2012.

Alexandre Belloni, Victor Chernozhukov, and Christian Hansen. High-dimensional methods and inference on structural and treatment effects. *The Journal of Economic Perspectives*, 28(2):29–50, 2014.

C. Lanier Benkard and Patrick Bajari. Demand estimation with heterogeneous consumers and unobserved product characteristics: A hedonic approach. Working Paper 10278, National Bureau of Economic Research, February 2004. URL `http://www.nber.org/papers/w10278`.

Steven Berry. Estimating discrete choice models of product differentiation. *RAND Journal of Economics*, 25:242–62, 1994.

Steven Berry, James Levinsohn, and Ariel Pakes. Automobile prices in equilibrium. *Econometrica*, 63(4):841–90, 1995.

Sandra E. Black. Do better schools matter? parental valuation of elementary education. *The Quarterly Journal of Economics*, 114(2):577–599, 1999. ISSN 00335533, 15314650. URL `http://www.jstor.org/stable/2587017`.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Bart J Bronnenberg, Michael W Kruger, and Carl F Mela. Database paper-the iri marketing data set. *Marketing Science*, 27(4):745–748, 2008.

Xiaohong Chen. Large sample sieve estimation of semi-nonparametric models. *Handbook of econometrics*, 6:5549–5632, 2007a.

Xiaohong Chen. Large sample sieve estimation of semi-nonparametric models. *Handbook of Econometrics*, 6:5549–5632, 2007b.

V. Chernozhukov. Least squares after model selection in high-dimensional sparse models. *Bernoulli*, 19:521–547, 2013.

Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Studying aesthetics in photographic images using a computational approach. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part III*, ECCV'06, pages 288–301, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-33836-5, 978-3-540-33836-9. doi: 10.1007/11744078_23. URL http://dx.doi.org/10.1007/11744078_23.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Ann. Statist.*, 29(5):1189–1232, 10 2001. doi: 10.1214/aos/1013203451. URL http://dx.doi.org/10.1214/aos/1013203451.

M. Gentzkow and Jesse M. Shapiro. What drives media slant? evidence from u.s. daily newspapers. *Econometrica*, 78(1), 2010.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning*, volume 2. Springer, 2009.

J. Ho and M. Yang. Machine learning in a hedonic model: Estimating the impact of fracking on house prices. 2014.

Harvey S. Rosen James N. Brown. On the estimation of structural hedonic price models. *Econometrica*, 50(3):765–768, 1982. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1912614.

Jeankyung Kim and David Pollard. Cube root asymptotics. *The Annals of Statistics*, pages 191–219, 1990.

H. Allen Klaiber and V. Kerry Smith. *Preference Heterogeneity and Non-market Benefits: The Roles of Structural Hedonic and Sorting Models*. Edward Elgar Publishing, Inc., Cheltenham, UK, 2011. ISBN 9781848444256. URL http://www.elgaronline.com/9781848444256.00017.xml.

Tatiana Komarova. Binary choice models with discrete regressors: Identification and misspecification. *Journal of Econometrics*, 177(1):14–33, 2013.

S. Livens, P. Scheunders, G. van de Wouwer, and D. Van Dyck. Wavelets for texture analysis, an overview. In *Image Processing and Its Applications, 1997., Sixth International Conference on*, volume 2, pages 581–585 vol.2, Jul 1997. doi: 10.1049/cp:19970958.

Kenenth E. McConnell Maureen L. Cropper, Leland B. Deck. On the choice of funtional form for hedonic price functions. *The Review of Economics and Statistics*, 70(4):668–675, 1988. ISSN 00346535, 15309142. URL `http://www.jstor.org/stable/1935831`.

Lucija Muehlenbachs, Elisheba Spiller, and Christopher Timmins. The housing market impacts of shale gas development. *American Economic Review*, 105(12):3633–59, December 2015. doi: 10.1257/aer.20140079. URL `http://www.aeaweb.org/articles?id=10.1257/aer.20140079`.

Lucija Anna Muehlenbachs, Elisheba Beia Spiller, and Chris Timmins. Shale gas development and the costs of groundwater contamination risk. Technical report, 2016/07/18/ 2013. URL `http://www.rff.org/research/publications/shale-gas-development-and-costs-groundwater-contamination-risk`.

A. Nevo. Measuring market power in the ready-to-eat cereal industry. *Econometrica*, 69(2): 307–342, 2001.

Whitney K Newey. The asymptotic variance of semiparametric estimators. *Econometrica: Journal of the Econometric Society*, pages 1349–1382, 1994.

Whitney K Newey. Convergence rates and asymptotic normality for series estimators. *Journal of Econometrics*, 79(1):147–168, 1997.

Serena Ng. Variable selection in predictive regressions. Technical report, Columbia University, 2012.

A. Petrin and K. Train. A control function approach to endogeneity in consumer choice models. *Journal of Marketing Research*, 47(1):3–13, 2010.

Anand Rajaraman and Jeffrey D Ullman. Mining of massive datasets. *Lecture Notes for Stanford CS345A Web Mining*, 67(3):328, 2011. URL `http://infolab.stanford.edu/~ullman/mmds.html`.

Peter M Robinson. Semiparametric econometrics: A survey. *Journal of Applied Econometrics*, 3(1):35–51, 1988.

Erwan Scornet. On the asymptotics of random forests. *arXiv preprint arXiv:1409.2090*, 2014.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Vladimir Vapnik. *Statistical learning theory*, volume 2. Wiley New York, 1998a.

Vladimir Naumovich Vapnik. *Statistical learning theory*. Wiley New York, 1998b.

Hal R Varian. Big data: New tricks for econometrics. *The Journal of Economic Perspectives*, 28(2):3–27, 2014.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.