ISSN 1936-5330

kc FED Research Working Papers

Explaining Machine Learning by Bootstrapping Partial Dependence Functions and Shapley Values

Thomas R. Cook, Greg Gupton, Zach Modig,
and Nathan M. Palmer

November 2021
RWP 21-12
http://doi.org/10.18651/RWP2021-12

Federal Reserve Bank of Kansas City

Explaining Machine Learning by Bootstrapping Partial Dependence Functions and Shapley Values

Thomas R. Cook*† Greg Gupton†‡ Zach Modig†‡
Nathan M Palmer†§

October 28, 2021

Abstract

Machine learning and artificial intelligence methods are often referred to as "black boxes" when compared to traditional regression-based approaches. However both traditional and machine learning methods are concerned with modeling the joint distribution between endogenous (target) and exogenous (input) variables. The fitted models are themselves functionals of the data, about which we can do statistical inference using computationally intensive methods such as the bootstrap. Where linear models describe the fitted relationship between the target and input variables via the slope of that relationship (coefficient estimates), the same fitted relationship can be described rigorously for any machine learning model by first-differencing the partial dependence functions. Bootstrapping these first-differenced functionals provides standard errors and confidence intervals for the estimated relationships. We show that this approach replicates the point estimates of coefficients attained in a linear OLS models, and demonstrate how this generalizes to marginal relationships in ML/AI models. This paper extends the partial dependence plot

*Federal Reserve Bank of Kansas City Email: thomas.cook@kc.frb.org

†The views expressed in this article are those of the authors and do not necessarily reflect the views of the Federal Reserve Board, the Federal Reserve Bank of Kansas City or the Federal Reserve System.

‡Federal Reserve Board of Governors

§Federal Reserve Board of Governors Email: nathan.m.palmer@frb.gov

described in Friedman (2001), and visualizes the marginal distribution used to construct the PDP as described in Goldstein et al. (2015) before applying the steps described above. We further discuss the extension of PDP into a Shapley value decomposition and explore how it can be used to further explain model outputs. We conclude with a hedonic house pricing example, which illustrates how machine learning methods such as random forests, deep neural net, and support vector regression automatically capture nonlinearities and shed light on inconsistencies revealed by meta-studies of hedonic house pricing.
JEL Classifications: C14, C18, C15

1 Introduction

Machine learning (ML) methods are often regarded as a black box: they may capture useful interactions and nonlinearities in data, but the shape and nature of the relationships are not easy to ascertain. At the same time, there is a growing appetite to use ML models in finance and economics for purposes ranging from academic study to credit underwriting. As such, it is important that we can adequately interpret ML models.

At a fundamental level, when we go about the exercise of 'interpreting' a model, we are going about the exercise of trying to ascertain the marginal effects of the input variables – the effect of a change in an input variable on the model's prediction. For linear models, this marginal effect can be easily ascertained in terms of point-estimates and variances of the model's parameter estimates (i.e. estimated coefficients of the model). Further, these effects are easily summarized and communicated in tabular form (a so-called regression table). For machine learning models, there are no analogues to such tables. The reason is straightforward. For linear models, the marginal effects of the variables of interest are constant and are described entirely by the values of the estimated parameters. By contrast, machine learning models learn complex relationships wherein the marginal effect of the variables of interest are contingent on various estimated aspects of the model along with the values of model covariates. For neural networks, for example, there are *many* underlying parameters that correspond to the marginal effect of a given RHS variable. For a support vector regression, the regularization parameter is important but knowing the optimal regularization value

for a given dataset, and even knowing a (5%, 95%) confidence interval around the optimal regularization parameter, communicates very little about the marginal relationships of interest. What is needed is a model-agnostic way to describe the marginal effects between the model target and input variables of interest, ideally which reproduces the coefficients for a linear regression.

Fortunately, Friedman (2001) constructed a functional that generalizes OLS coefficient estimates and is model-agnostic: the partial dependency function, more widely known as the partial dependency plot (PDP). As the name indicates, this function is often displayed as a plot of a partial relationship.[1] Traditionally in the ML literature the PDP is displayed in levels, and only the point estimate of the PDP is calculated. However it is straightforward to demonstrate that when applied to linear regression, the first difference of the PDP directly replicates the coefficients of that regression, and bootstrapping the PDP produces standard errors and p-values comparable to the analytical results of running OLS. When applied to non-linear ML models, this approach generalizes the concept of the coefficient in a model-agnostic way. Applying the PDP to non-linear models, we can ascertain the nature of the fitted relationship in a way that is familiar to our understanding of the interpretation of a linear model. We can further extend the logic of the PDP to understand variable importance as well.

This paper contributes to the extensive and fast-growing interpretability literature in machine learning. Breiman (2001) provides an introduction to interpretability vs prediction in machine learning. Semenova, Rudin, and Parr (2019) and Molnar (2021) are two modern overviews of ML interpretability that provide a wide survey of the field. This paper in particular extends the partial dependence plot (PDP) described in Friedman (2001), and visualizes the marginal distribution used to construct the PDP as described in Goldstein et al. (2015)[2]. Section 3 provides similar discussion with respect to Shapley values (Shapley, 1953) and in response to two additional aspects of model interpretation: the effect of input inclusion and feature importance. Discussion of both the PDP and Shapley values demonstrate their equivalence to

[1]The marginal relationship between a target variable and a given input variable are non-linear over the range of the selected RHS variable for the general machine learning model, whereas the same relationship is linear for a simple OLS regression.

[2]The PDP is closely related to the "observed-value" approach described in Hanmer and Ozan Kalkan (2013) as well.

parameter estimates in the context of a linear model.

The illustrative hedonic house pricing model employed is inspired by the meta-analysis in Sirmans, Macpherson, and Zietz (2005) and Zietz, Zietz, and Sirmans (2008), and the data used is described in De Cock (2011).

The rest of this paper is organized as follows: Section 2 outlines how PDP can be constructed as a generalization of linear model coefficients. Section 3 provides a similar discussion with regard to Shapley values. Section 4 applies our results to the hedonic house-pricing exercise, and Section 5 concludes.

2 Model Agnostic Approaches to Inference

There are two commons purposes for constructing statistical models that relate left-hand side (target) variable, to a right-hand side (input) variable (in machine learning parlance, a supervised learning problem).

The first purpose is **prediction**: given a new observation of inputs, predict the associated target. This is a common use case, and improved predictive performance is an often-cited reason for employing ML and AI models in place of traditional models.[3]

The second purpose is **inference**: rather than predict the model target, the emphasis is on describing the world by examining the relationship between the target and the inputs captured by the model for the data set the model has been fit to, and describing the statistical properties of that relationship. For example, does a particular variable in the inputs have a positive or negative relationship with the target? Is that relationship statistically stable? That is, if this model was estimated on a different draw from the data generating process,would the relationship still be approximately the same? Or is the relationship so noisy that it is indiscernible?

Inference is one of economists' primary use cases for statistical models, and an extensive history of statisticians and economists have developed theoretical foundations for inference in econometrics. The lack of inference tools for ML and AI models

[3]See chapter 2 of James et al. (2013) for a discussion of prediction and inference in ML and traditional models, and discussion of ML models' improved forecast accuracy versus traditional models.

reveals itself in slow adoption rates in economics[4]. In the ML and AI literatures, the lack of inference tools reveals itself in the rapidly growing literature on explainability and interpretability.

Fortunately there is a promising path forward for inference, driven by two observations: *First*, the marginal relationships between the target and model inputs that are captured by coefficients in a traditional linear regression model can be estimated in a more general way that applies to any model. The approach we focus on in this paper is described in Friedman (2001) as the partial dependency function or "partial dependency plot" (PDP).[5] As described in the following section, the slope of the PDP is exactly the coefficient in a traditional linear regression[6]. This is due to the fact that Friedman (ibid.) constructed the PDP to be a generalization of the "ceteris paribus" reasoning that is taught regarding regression coefficients in introductory statistics courses.

Of course the coefficients in a traditional model (and the PDPs for a generalized model) are only a point estimate of the marginal relationship. We also care about the variance in this relationship, which brings us to the *second* observation: as described in Efron and Hastie (2016), the bootstrap and related methods can provide a straightforward if computationally intense way to calculate variance of a wide range of functions of data. We employ the bootstrap to find the variance in the slope of the marginal relationships captured by the PDP.

When we apply the PDP to a traditional linear regression model and bootstrap to obtain the variance, we replicate the traditional point estimates and variances of the coefficients that one obtains in a standard regression table. When applied to a general nonlinear ML model, we obtain a generalization of the regression table, which allows us to conduct inference with the ML model in the same way we conduct inference in a traditional econometric model via a regression table.

[4]Although see Athey and Imbens (2019) and Coulombe (2021) for examples of inference on some ML models and applications in economics.

[5]As we will describe in more detail later, there are a number of ways to generalize the marginal relationships that OLS coefficients embody. We focus on the two most popular, PDP and ALE, but Section 2.5 discusses a number of variations which may have better statistical properties.

[6]see Appendix C for proofs

2.1 PDP and Regression

This section uses two analogies to provide an intuitive description of what the PDP captures, before turning to the mathematical details. A key insight is that the PDP helps an economist understand the properties of a fitted model itself.

For the first analogy, suppose that we have a fitted model. We can think of the PDP as providing a summary statistic about the distribution of results for the following experiment:

> Take an observation and plug it into the fitted model and get a prediction. Change nothing else about this observation except for a single characteristic (for example change square footage for a house, but leave number of rooms, lot size, etc, unchanged). How much does the model output change? What if we did this with many observations? What is the average of conducting this experiment over the domain of the variable in question?

In this sense, the PDP is communicating something about what a fitted model would in fact predict, if it were asked to predict an observation where only one characteristic was changed. We are learning something about the fitted model itself through this process.

Alternatively, we can think of the PDP as analogous to conducting a type of field experiment has been done in experimental economics, only applied to a model that makes predictions. For example, in Bertrand and Mullainathan (2004), the authors submitted a number of resumes to a hiring process, and then change a single characteristic of the resumes (the name) and examine what changes in the outcomes (number of call-backs). The PDP essentially implements this experiment on a fitted model.

If these sounds like the interpretation of coefficients in multiple regression, that's because it is. Friedman (2001) constructed PDPs to implement and generalize the "ceteris paribus" reasoning for interpreting multiple regression coefficients as taught in most introductory econometrics courses. The main difference is that Friedman (ibid.), and almost all subsequent ML and AI literature, discusses the PDP in terms of the

level of the relationship, not the slope of the relationship, which is what traditional multiple regression coefficients capture.[7]

Consider a typical description of multiple regression coefficients, drawn from Abdi (2004) in the Encyclopedia of Social Sciences Research Methods:

> "[A] regression coefficient... gives the amount by which the dependent variable (DV) increases when one independent variable (IV) is increased by one unit and all the other independent variables are held constant."

The partial dependency function is defined by Friedman (2001) so as to directly capture this reasoning for any fitted model. This is useful in part because it means a student who has internalized the intuition of multiple regression coefficients can employ the same reasoning (and the same caveats!) to understand the relationships in data represented by an ML or AI model.

We can see this ceteris paribus reasoning implemented in the mathematics of the PDP applied to a fitted model $\hat{f}$.

Write the fitted model as:

$$\hat{f}(x) = \hat{f}(x^{(k)}, x^{(\neg k)}) \tag{1}$$

where $\hat{f}$ is the fitted model, x is the vector of input variables, and $x^{(k)}, x^{(\neg k)}$ simply separate out the single input variable $x^{(k)}$ from all other input variables, $x^{(\neg k)}$. For example, in the hedonic house pricing model to be described, k might represent square footage; then $x^{(\neg k)}$ would represent all other variables which are not square footage, such as number of bedrooms, age, neighborhood, etc..

The partial dependence function (PDP) for input variable denoted k for fitted model $\hat{f}$ is defined as follows:

[7]The reason for this is likely that tree-based models (what Friedman (2001) invented the PDP to describe) do not have smooth slopes in their PDP, unlike other methods like support vector machines, deep neural nets, or kernel ridge regressions (SVMs/SVRs, DNNs, KRRs, respectively), and looking at the PDP level is natural for a tree. However even for tree-based methods, approximations of the slope can be examined and provide insights similar to those of smooth methods.

$$\nu_k(q) = E_{x_{\neg k}}[\hat{f}(q, x^{(\neg k)})|q] \tag{2}$$

$$= \int_{x^{(\neg k)}} \hat{f}(q, x^{(\neg k)}) \mathbb{P}(x^{(\neg k)}) dx^{(\neg k)} \tag{3}$$

where $\mathbb{P}(x^{(\neg k)})$ in equation 3 is the marginal distribution over the input data. Thus the PDP "holds all other independent variables constant" by employing the marginal distribution directly. If the conditional distribution was used, for example, it would no longer be true that "all other independent variables are held constant"[8] for further discussion. This is a direct construction of the *ceteris paribus* assumption employed in the traditional interpretation of regression coefficients.

The PDP can be estimated using the fitted model and data via Monte Carlo estimate of $\hat{\nu}(q)$ for each value q in a range $[q_{low}, \ \ q_{high}]$:

$$\hat{\nu}_k(q) = \frac{1}{N} \sum_{n=1}^{N} \hat{f}(q, x_n^{(\neg k)}) \tag{4}$$

where N is number of observations used and the above is calculated for each q in the desired domain of the variable k: $q \in \left[x_{min}^{(k)}, \ \ x_{max}^{(k)}\right]$.

That is, for each $q \in \left[x_{low}^{(k)}, \ \ x_{high}^{(k)}\right]$, the above mean is taken over the empirical marginal distribution of the fitted values $\hat{f}(q, x_n^{(\neg k)})$, where the $x^{(\neg k)}$ values in the data are literally held constant. The resulting PDP function is in levels of the predicted $\hat{y}$ variable; one can obtain the slope of this marginal relationship using a simple first difference approximation. Appendix C proves that for a linear model, this slope is equivalent to the traditional multiple regressions coefficient.

This partial dependence function of course is a point estimate of this relationship. If we want the variance in the estimate, then we can employ the non-parametric (pairs) bootstrap to the entire process as described in the following algorithm:

This is computationally expensive but also an embarrassingly parallel problem.

When this process is applied to a linear model, it directly replicates the traditional

[8] see Section 2.5

Algorithm 1 Bootstrap PDP

Preallocate output matrix Z with dimensions $(J \times B)$
Allocate vector Q as a J-length vector of equally spaced values from $\left[x_{low}^{(k)},\ \ x_{high}^{(k)}\right]$
for b in 1 to B: **do**:
    X_b, y_b =bootstrap(X)
    Estimate f_b s.t. $\hat{y_b} = f_b(X_b)$
    for j in [1,J]: **do**
        $q = Q_j$
        $Z_j^{(b)} = f_b(q, X_b^{(\neg k)})$
Return Z

regression table for point estimates. Bootstrapping this process provides standard errors around these point estimates and allows for model inference.

2.2 PDP Simulation Exercise

Consider a simple data generating process $y = XB' + u$ where $X = (x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}$ is a set of independent covariates and u is an IID random disturbance term, and fix parameters $B = (1,\ -2,\ 10,\ 0)$. We simulate a sample of 3000 observations from this dataset and estimate B via OLS regression. The results are presented in the first column of Table 1 and they have the conventional interpretation.

When we calculate the slope of the PDP for this linear model, we directly replicate the slope estimates of the OLS regression in 1, and bootstrapping provides nearly equivalent standard errors (we should not expect bootstrapping to replicate the standard errors to machine precision, as is the case with the point estimates)..

The PDP for the linear model, with bootstrapped confidence interval, is displayed in Figure 1, and the PDP slope is presented in Figure 2. The average slope of these PDP lines and average standard deviation are displayed in the second column of Table 1; note that the OLS B point estimates are equal to the values of the PDP slope to machine precision.[9]

[9]Because the PDP slope technically changes over the range of the reference variable, we only display the average of the slope and average of the standard deviation in the second column of this this table. Under the special case of a linear model all PDP slope values are the same over the entire range. This will not be true of a nonlinear ML model as will be demonstrated.

We now estimate an ML model – a gradient boosting machine (GBM) – on the same simulated dataset. Figure 3 displays the PDP in levels for the GBM, while Figure 4 displays the PDP slope, both with (5%, 95%) bootstrap confidence intervals. The GBM is nonlinear in its PDP, and there is no single number which can summarize this slope for each variable. For completeness we have included the average PDP slope and average PDP bootstrapped variance in the final column of Table 1, but Figures 3 and 4 demonstrate that these averages do not tell the whole story. The nonlinear GBM, a tree-based method, finds some slight nonlinearities in the edges of the input variable ranges, and these lower slopes bring down the average slope. Figure 4 displays the PDP slopes, and the (5%, 95%) confidence intervals demonstrate that all but the $x^{(4)}$ variable are statistically significant – the same as in the OLS results.

Table 1: Parameter estimates on simulated dataset from linear DGP

	Parameter	OLS	Δ OLS PDP	Δ GBM PDP
$x^{(1)}$	1	1.0154	1.0154	0.6242
		(0.013)	(0.012)	(0.281)
$x^{(2)}$	-2	-2.0108	-2.0108	-1.5453
		(0.013)	(0.012)	(0.505)
$x^{(3)}$	10	10.0188	10.0188	8.6805
		(0.013)	(0.012)	(1.447)
$x^{(4)}$	0	0.0013	0.0013	-0.0128
		(0.013)	(0.012)	(0.101)
N		3000	3000	3000

Now, consider a dataset with a nonlinearity in the relationship between target and input variables. Consider the data generating process described above, but allow for the presence of a nonlinearity in the relationship of $x^{(1)}$ to the model output y, so that $y = (x^{(1)})^2\gamma + XB + u$ with $B = (0, 5, -5, \epsilon^+, \epsilon^-)$, $\gamma = 10$, and (ϵ^+, ϵ^-) being positive and negative values near zero. In this example, an OLS model will only properly fit the data if the nonlinearity in $x^{(1)}$ is explicitly specified. Most machine learning models, by contrast, will learn this nonlinear relationship implicitly. Either way, the presence of a nonlinearity makes direct interpretation of the model more difficult.

The PDP for the OLS model[10] is presented in Figure 5. As before, since this is a linear model, differencing the PDP yields the original OLS estimates and the

[10]see Appendix A for OLS estimates.

Figure 1: PDP for OLS model fit to simulated data from linear DGP

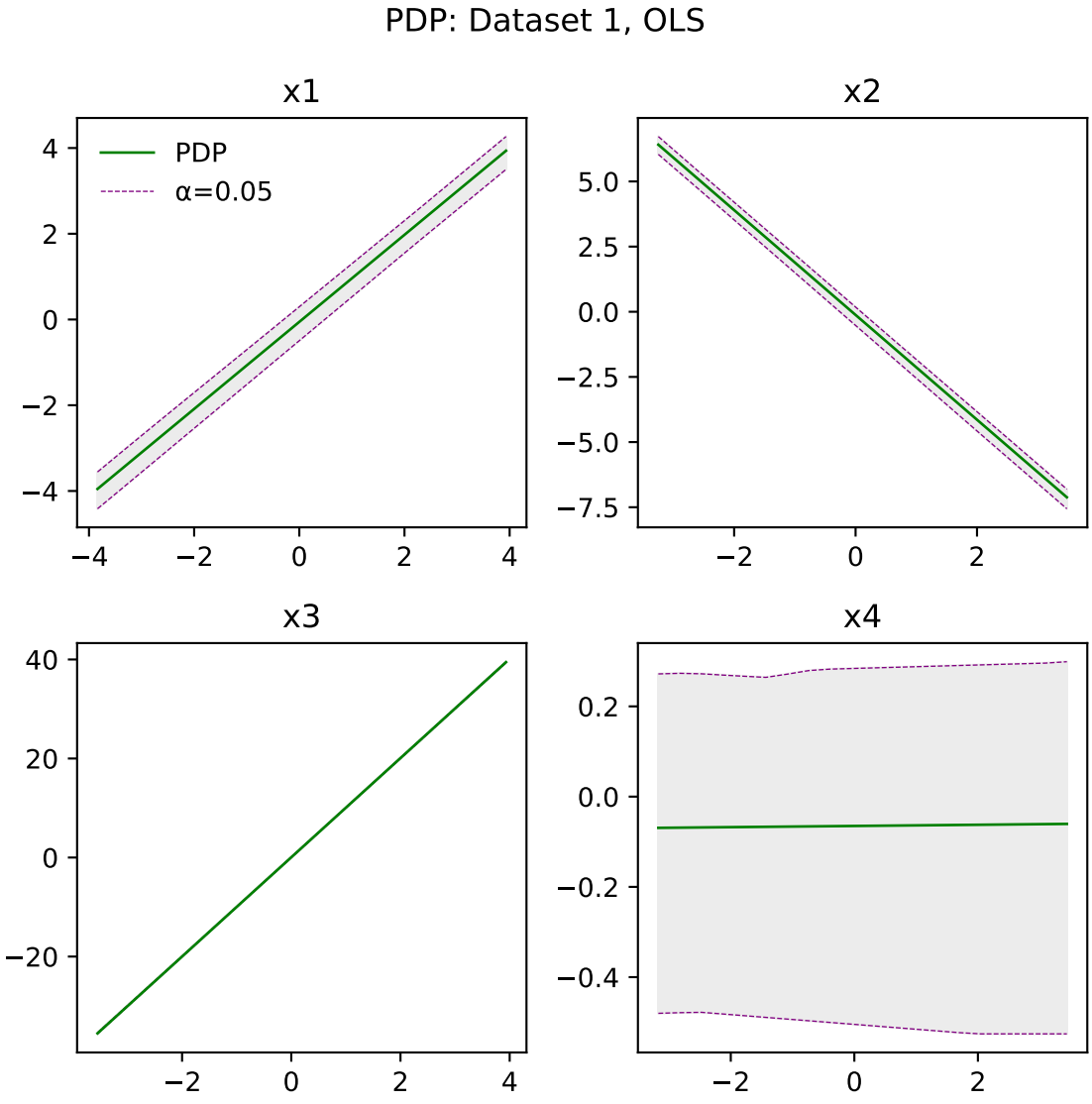


Figure 2: PDP slope for OLS model fit to simulated data from linear DGP

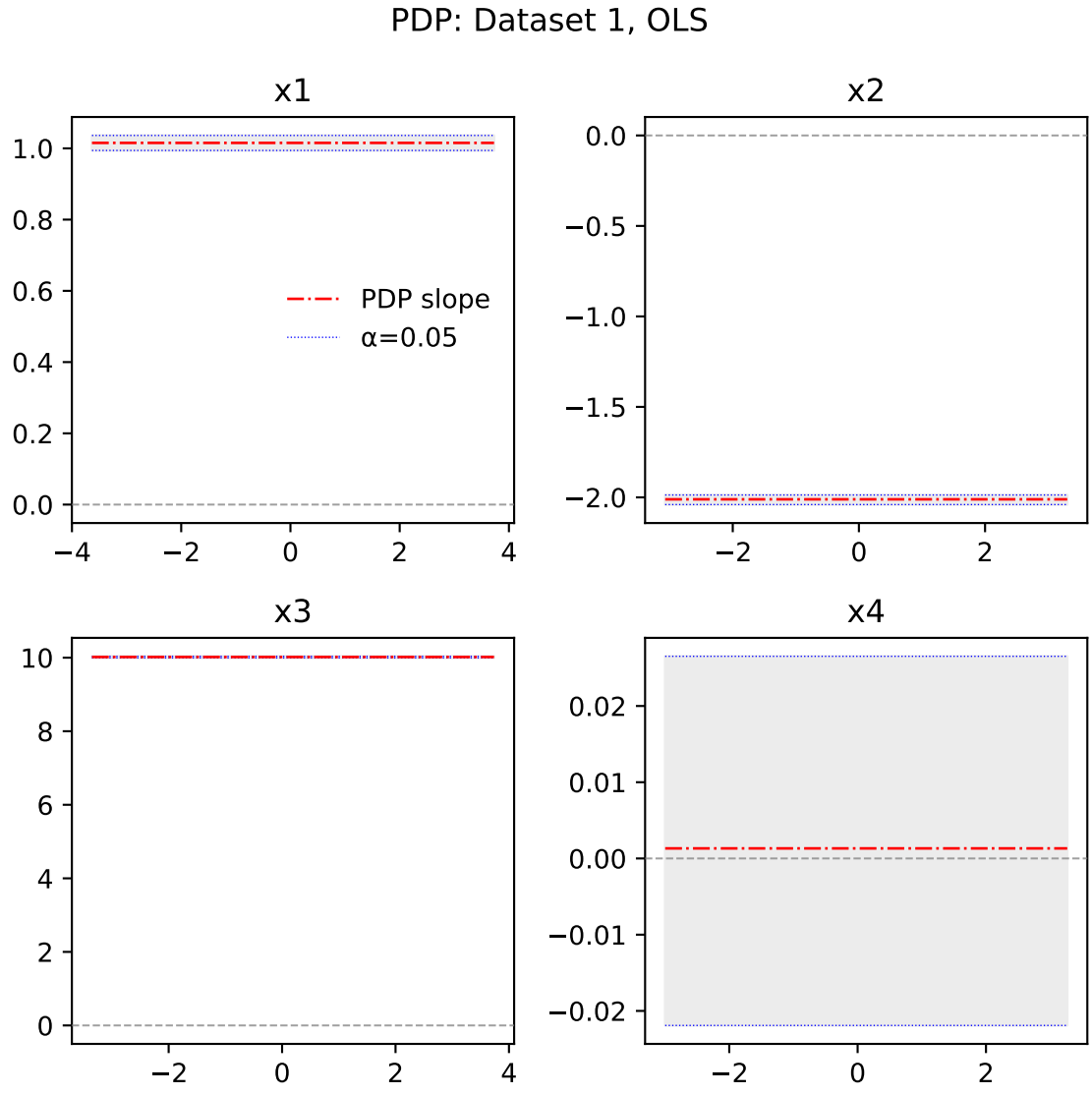


Figure 3: PDP for GBM model fit to simulated data from linear DGP

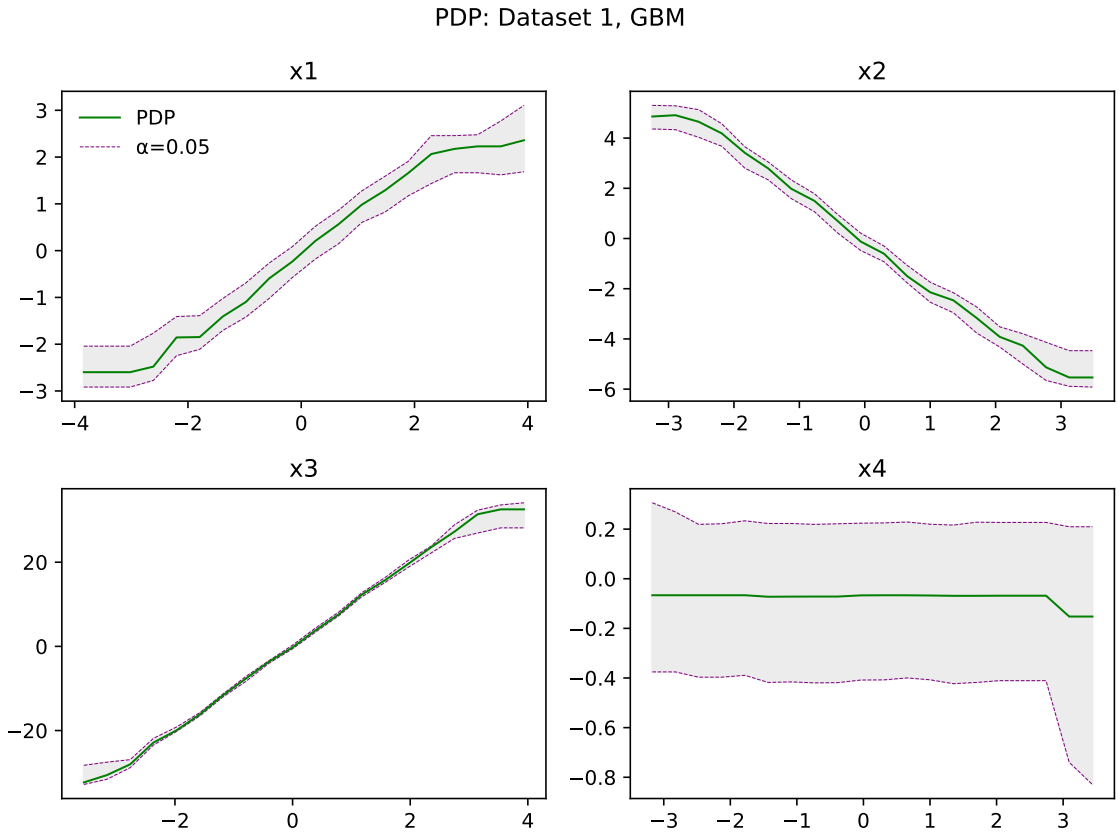
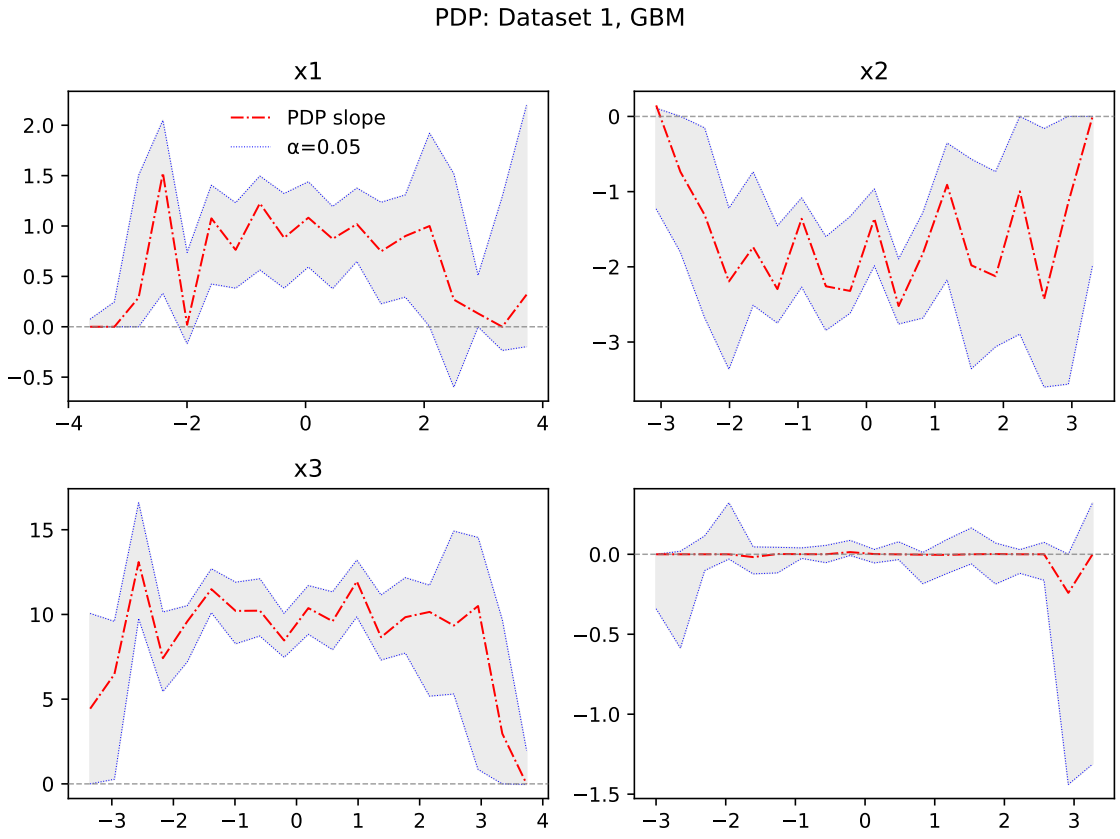


Figure 4: PDP slope for GBM model fit to simulated data from linear DGP



application of algorithm 1 allows us to closely approximate the standard error of the OLS estimates. These PDP functions reveal the nonlinear relationship in the data, but only to the extent that the nonlinearity was explicitly specified[11]. The PDP functions confirm what we already know: the linear model does not learn any relationships between inputs and outputs that are not explicitly linear.

By contrast, consider the PDP for the GBM model as presented in Figure 6. The GBM will learn a nonlinear relationship between the inputs and target; we do not need to supply the GBM with an explicit $(x^{(1)})^2$ variable. Looking at the PDP helps us understand the nature of the relationship that the GBM model has learned. Examining Figure 6, we see that for $x^{(2)}$ and $x^{(3)}$, the mdoel has learned an approximately linear relationship, a flat (or null) relationship for $x^{(5)}$, and most importantly, a distinctly quadratic relationship for $x^{(1)}$.

This ability to use PDPs to infer nonlinear relationships learned by ML models hints at one approach for fruitfully using ML and OLS models together. First, estimate a fast ML model (such as a GBM) on the data in question and then use the PDPs of those models to identify nonlinearities. Add these nonlinearities to the specification of a corresponding OLS model.[12]

[11] i.e. the quantity $(x^{(1)})^2$ is supplied to the model as a precalculated variable.

[12] An important consideration is how to think about this strategy with respect to the multiple comparisons problem. Preselecting variable transformations from a ML model and embedding them in a parametric model may imply that the resulting significance levels in the parametric model are too narrow. More broadly, think of decompose the estimating of a model into two steps: preliminary model choice (which transformations and interactions to include in X), and model estimation given the model choice. If only the second step is included in the variance calculation there may be a multiple comparisons problem. In the current context, ML models which automatically discover interactions and nonlinearities as part of the estimation process side-step the multiple comparisons problem, as the variance calculation applied via bootstrap encompasses both the model choice (with respect to variable transformations and interactions) and the model estimation steps. If one were to instead first estimate an ML model, observe the interactions and nonlinearities, add those to a linear model, estimate the linear model, and then simply accept the default errors for the estimated linear model, those errors will likely need to be adjusted to account for multiple comparisons. Doing this appropriately requires more careful thought. See also helpful discussion in Chapter 20, "Inference After Model Selection," in Efron and Hastie (2016).

Figure 5: PDP for OLS model fit to simulated data from linear DGP

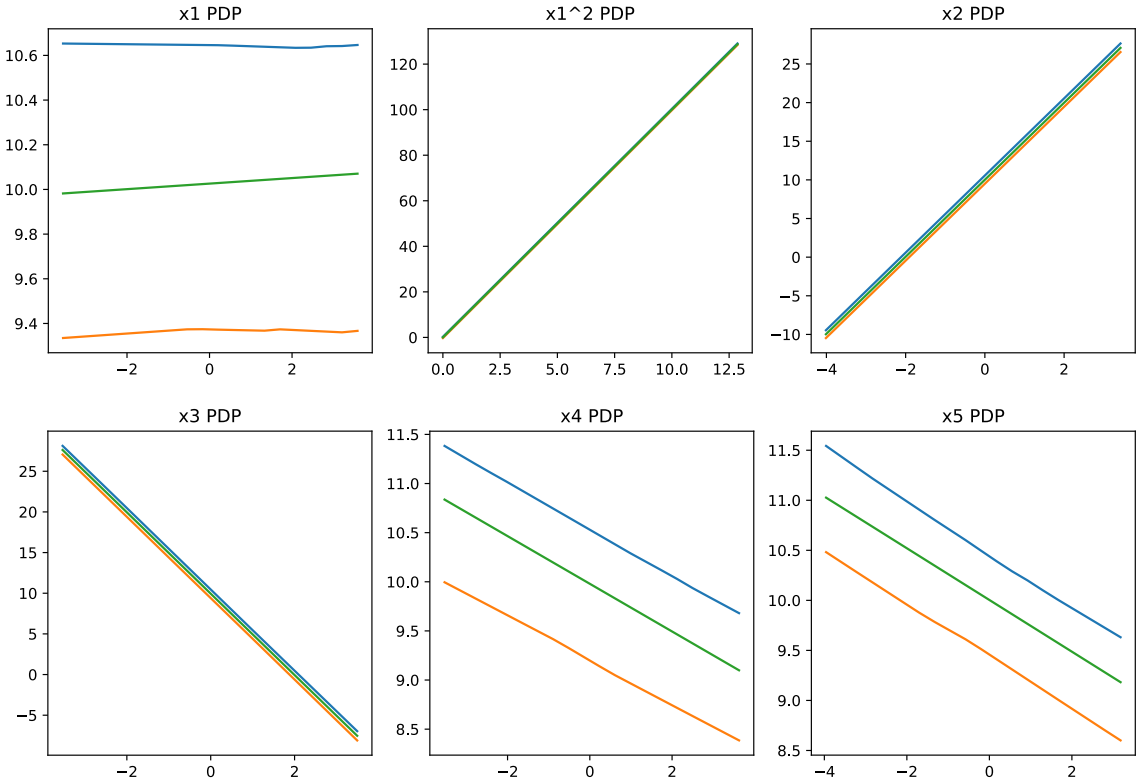
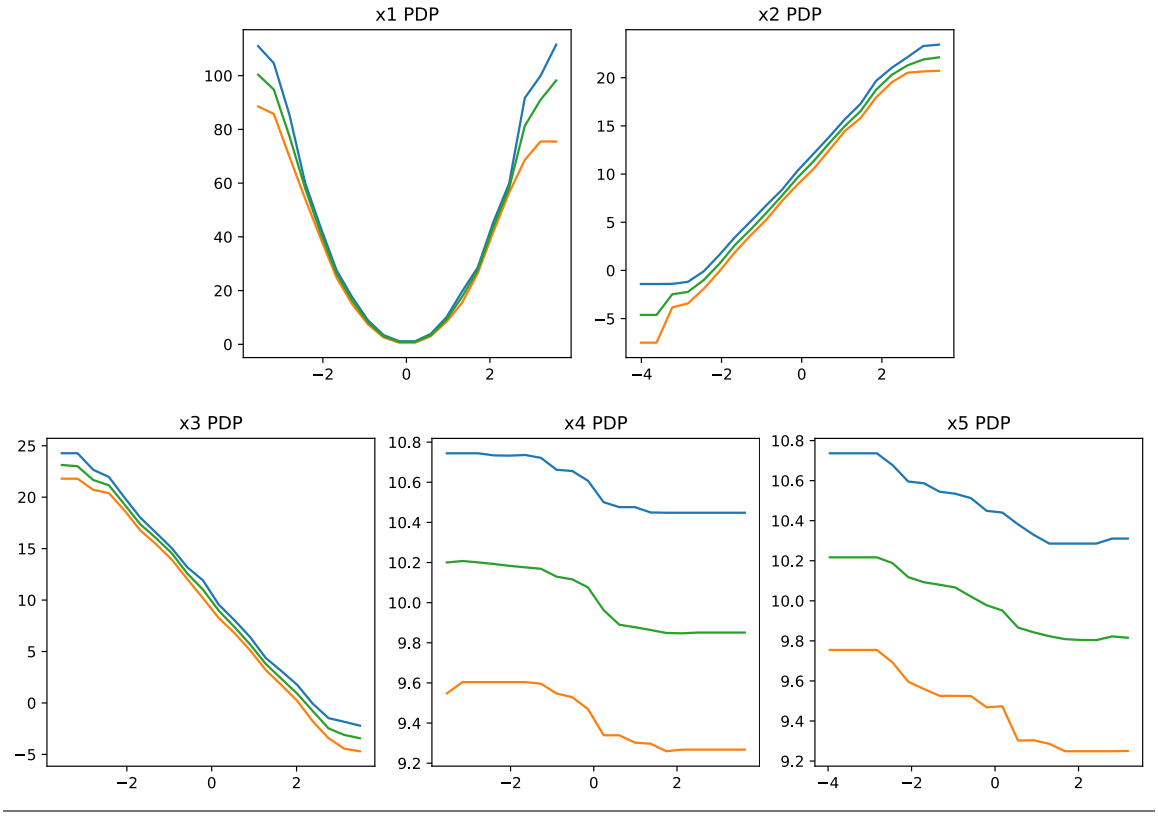


Figure 6: PDP for GBM model fit to simulated data from linear DGP



2.3 Disadvantages of PDP

This section discusses disadvantages of PDP, and the following section compares PDP with ALE, the main alternative method.

The two main disadvantages of using PDP are 1) speed and 2) the estimand.

Regarding **speed**, PDP can be slow to calculate. For a single input variable, with m points in the input variable range and n data points used to evaluate the PDP, there are $n * m$ evaluations of the fitted model needed to construct the PDP. Bootstrapping this b times means that both b models are required to be fitted and a total of $b * m * n$ model evaluations are needed. This can be computationally costly, although this can also be embarrassingly parallelized for the bootstrap steps. Regardless, this number of evaluations can be slow.

Regarding the **estimand**: PDP is a generalization of the "ceteris paribus" reasoning described in the quote from Abdi (2004) in Section 2.1. Like coefficients in a linear regression, it suffers from the usual downsides of reasoning about global properties of the model when input variables are not independent from one another. That is, if input variables are correlated, interact, or have nonlinearities, then the ceteris paribus assumption doesn't hold.

To build intuition, in the house-pricing example to be discussed below, consider the effect of square footage (input) on house prices (target), and think of the interaction that almost certainly occurs between square footage and number of bedrooms: many bedrooms is almost certainly associated with higher square footage. Nonetheless, the ceteris paribus assumption means that when we think about the effect of going from 5000 to 5001, we treat all data as equally likely to experience that change. Consider a datum that has 1 bedroom and is 500 square feet. The ceteris paribus assumption implies that when we consider the effect of going from 5000 to 5001 square feet for this datum, we hold constant the number of bedrooms at 1. This is implicit in the mathematics of multiple regression, but explicit in the estimation of the PDP for a linear model – we literally replace 500 square feet with 5000 square feet, predict a price with the fitted model, do it again with 5001 square feet, and use these values to extract the slope. This observation under these conditions should likely be very low-probability, but in the PDP (and in standard multiple regression calculation) they

are equally weighted with all other observations.[13]

This can be partially addressed by adding nonlinear manipulations and interactions to a linear model, or in a ML model some of these will emerge from the fitting process, as was seen above. In this way ML models may be viewed as automatically addressing *some* specification problems.

None the less, it is still the case that the PDP uses the marginal distribution of the data to create the marginal relationship between the input and target variables of the fitted model. While the PDP as written in Equation 3 generalizes the ceteris paribus of multiple regression, one might instead prefer to replace the marginal distribution $\mathbb{P}(x^{(\neg k)})$ with the conditional distribution $\mathbb{P}(x^{(\neg k)} \mid q)$ to get a "conditional partial dependence function" (cPDP):

$$c\nu_k(q) = E_{x_{\neg k}|q}[\hat{f}(q, x^{(\neg k)})|q] \tag{5}$$

$$= \int_{x^{(\neg k)}} \hat{f}(q, x^{(\neg k)})\mathbb{P}(x^{(\neg k)} \mid q)dx^{(\neg k)} \tag{6}$$

This could be estimated by modifying the Monte Carlo estimator of the PDP described above, replacing the $\frac{1}{N}$ probability with probabilities $p(x_n^{(\neg k)} \mid q)$ taken from a joint kernel density over the entire input dataset, and using the conditional kernel density implied by that for calculating the probabilities:

$$\hat{c\nu}_k(q) = \sum_{n=1}^{N} \hat{f}(q, x_n^{(\neg k)})p(x_n^{(\neg k)} \mid q) \tag{7}$$

As with the PDP, the slope can be obtained via a difference approximation. Unfortunately, this conditional PDP is much slower to estimate practically than the typical PDP.[14] Alternatively, the Accumulated Local Effect (ALE) of Apley and Zhu (2020)

[13]To reiterate, this process directly implements the ceteris paribus assumption of basic linear model coefficient interpretation – only it makes the improbability of this assumption for corrected DGPs much more obvious.

[14]There is almost certainly a trick that can be used with kernel density estimates over the input dataset that can speed this up, but the naive implementation we explored is too slow for use currently. Apley and Zhu (2020) describes an alternative estimator, ALE, with a similar but not identical approach to incorporating the conditional PDP. Their implementation does not use the full

attempts to capture a similar conditional relationship, and seeks to address both the "estimand" and the "speed" problems of PDP.[15] This is the primary model-agnostic alternative to PDP in the literature.

2.4 Accumulated Local Effect (ALE)

Accumulated Local Effect (ALE) was introduced by Apley and Zhu (2020) and seeks to address both the "estimand" and "speed" problems described above for PDP.

Where PDP uses the marginal distribution of an explanatory variable to generate the expected marginal relationship over the range of the selected variable, ALE uses the conditional distribution as described above.

However the order of integration and differentiation is reversed. The PDP and cPDP described in the previous sections are first integrated across all observations to get the expected marginal fitted relationship in levels, and then differentiated to get the slope of that expected relationship.

ALE reverses this order: a slope estimate for the fitted relationship is first calculated for each relevant observation[16] at each point along the ALE domain, and then the expectation is taken over these slopes. Thus differentiation is applied to the fitted relationship first, and integration applied second. This process produces an expected slope of the fitted relationship between the target and the input variables, which Apley and Zhu (ibid.) refer to as "local effects." To get a levels series comparable to PDP, the authors choose a starting point in the domain of each input variable, and use the estimates slopes to build a levels series. That is, the local effects are accumulated over the domain of the relevant input variable, hence the name, "Accumulated Local Effects." From an econometric perspective, the "local effects" are comparable to the coefficients in a typical regression table, and we examine these as well.

Where the PDP uses the full information set of the data over the full domain

information set of the input dataset but rather a local approximation; this alternative also switches the order of integration and differentiation. ALE will be discussed in depth in Section 2.4.

[15] Note that the "estimand problem" is not a problem if one wants to construct an analogue to the "ceteris paribus", coefficient interpretation described thus far.

[16] As will be described, ALE uses a subset of the data to as the relevant observations for each point.

of the function, ALE only uses datapoints that occur within a local region around each point in the domain of the input variable. Specifically as described in Apley and Zhu (2020), the domain of the input variable under consideration is divided into quantiles to ensure that the same number of data points are used for each estimate of the slope. Thus very low-likelihood areas of the state-space are unlikely to appear in the calculation of the slope, and the ceteris paribus assumption does not need to be made. This comes at the cost of using less information than the PDP, and this reveals itself as wider confidence intervals and "more jagged" ALE lines when compared to PDP lines calculated on the same fitted model, as can be seen in the appendix.

The mathematical description of ALE from Apley and Zhu (ibid.) is:

$$ALE_k(x_i^{(k)}) = \int_{x_{min}^{(k)}}^{x_i^{(k)}} \mathbb{E}_{x^{(\neg k)}|z^{(k)}} \left[\hat{f}^{(k)}(z^{(k)}, x^{(\neg k)}) \mid z^{(k)} \right] dz^{(k)} - \mathrm{C} \tag{8}$$

$$= \int_{x_{min}^{(k)}}^{x_i^{(k)}} \int \hat{f}^{(k)}(z^{(k)}, x^{(\neg k)}) \mathbb{P}(x^{(\neg k)} \mid z^{(k)}) dx^{(\neg k)} dz^{(k)} - C \tag{9}$$

where C is a constant, and

$$\hat{f}_i^{(k)}(x_i^{(k)}, x_i^{(\neg k)}) = \frac{\partial \hat{f}(x_i^{(k)}, x_i^{(\neg k)})}{\partial x_i^{(k)}} \tag{10}$$

is the individual-level "local effect" of $x_i^{(k)}$ on the fitted model $\hat{f}$ at the point $(x_i^{(k)}, x_i^{(\neg k)})$. As noted previously this process first takes the partial derivative and then integrates that to get the expectation of the effect. The second, outer integration is the "accumulation" from some initial starting point to obtain a levels series. The constant centers the ALE function in its range.[17]

[17]Both the ALE and PDP functions as described are the "main effects" functions; interaction effects can also be constructed but we do not discuss these here.

2.4.1 Choosing ALE or PDP

As PDP and ALE estimate similar but not identical relationships, one might ask which to choose. We have chosen to use PDP largely because it reduces the variance in our final estimate of the marginal relationship we care about, and because it provides a conceptual bridge between interpreting regressions and interpreting AI/ML models. However this is not a firm conclusion and may change for any number of reasons. One could also imagine using both. Recall that each PDP or ALE function must be constructed individually for each input variable, a costly process. One could imagine using ALE to quickly examine many relationships, and PDP to examine specific relationships in depth.

Regardless, this is an appropriate place to compare ALE and PDP. Here are a few considerations:

Properties that recommend ALE include:

- Speed: less evaluations of the fitted model necessarily imply faster code
- Estimand: PDP implements the ceteris paribus reasoning, but we care about this in part because this is the reasoning everyone is taught for interpreting OLS coefficients, providing a conceptual bridge from OLS to ML/AI interpretability. However we may actually prefer the conditional relationship that ALE describes.
- Direct construction of slope: with PDP, we first construct the levels function and the first-difference it to get the slope. With ALE, we construct the slope in the first step.

Properties that recommend PDP include:

- Estimand:
  - PDP directly implements the ceteris paribus reasoning, providing a conceptual bridge from OLS interpretation to ML/AI interpretation
  - PDP implements the block-recursive causal structure of the Pearl (2009) approach to causality, as described in Zhao and Hastie (2021)

  - PDP provides a summary statistic about the distribution of results for the following experiment: "take an observation and plug it into the model and get a prediction. Change nothing else about this observation except for a single characteristic. What changes about the model output? What if we did this with many observations?". This type of field experiment has been done in experimental economics, for example as in Bertrand and Mullainathan (2004) and others, and may be intuitive for a wide range of audiences.

- Variance: by using more information PDP appears to be less variable, which appears in both the visual noise of the function and in tighter confidence bands, both of which can aid interpretation

The question of whether to use ALE or PDP is essentially a question of which tradeoffs to choose: slower PDP calculation with a traditional ceteris paribus interpretation and lower-variance output, or faster ALE calculation with a somewhat less tradition interpretation and higher variance in the marginal relationship.

We have coded up both a bootstrapped version of PDP and ALE, and compare the results of both PDP and ALE for the hedonic house pricing exercise in the appendix. Aside from higher variance in the ALE estimates, we find that both the PDP and ALE approaches perform similarly.

2.5 Additional Generalizations of Marginal Relationships

The structure of ALE and PDP calculation suggest that there are in fact a small number of estimators of the slopes of the marginal relationship of interest, which may have different advantages either theoretically or in practice. We suggest a few here but otherwise simply note these for future work.

For example, one potential extension of ALE, not pursued in our current paper, is to apply the "derivate first, then integrate" reasoning of ALE to the conditional PDP described earlier. This would combine the "full information" benefit of PDP with the "local information" benefit of ALE, and may produce a smoother, more intuitive descriptive statistic. We suggest "conditional local effects" (CLE) as a descriptive

name. Under such a setup the ICE lines and $\hat{y}$ points would still be useful and appropriate to display to potentially further aiding interpretation.

A simpler extension would be to use the "derivate first, then integrate" reasoning of ALE with the marginal distribution of PDP, resulting in something like a "partial local effects" (PLE) model which retains the ceteris paribus reasoning of PDP as well as some of the controls for correlations between variables that Apley and Zhu (2020) note is an advantage of the "derivate first" approach of ALE.

In fact, we can describe several different estimators for the marginal effect as a profile of three distinct choices:

1. Order of operation of calculating expected slope: (1) integrate then derivate: $\frac{\partial \mathbb{E}\left[\hat{f}(.)\right]}{\partial x_i^{(k)}}$, vs (2) derivate then integrate: $\mathbb{E}\left[\frac{\partial \hat{f}(.)}{\partial x_i^{(k)}}\right]$
2. Using the (1) marginal distribution vs (2) conditional distribution for the expectation (using $\frac{1}{N}$ vs conditional kernel-density-based weights for the expectation over ICE lines vs slopes), and
3. Using (1) global or (2) local data to form the expectation (eg. kernel density vs local sample).

.

PDP is {1, 1, 1} while ALE is {2, 2, 2}. The cPDP suggested above is {1, 2, 1}, m-plots as described in Apley and Zhu (ibid.) (not discussed here) are {1, 2, 2}, the "conditional local effects" (CLE) suggested above would be {2, 2, 1}, and the "partial local effects" (PLE) would be {2, 1, 1}. Other permutations are possible but may not yield much practical advantage. There are likely tradeoffs between how quickly these converge under the bootstrap, how variable they are, and how efficiently these can be calculated taking advantage of computational tricks (for example, using kernel density estimates to accelerate calculation).

Future research should examine the speed, variance, and convergence properties of the different estimators of marginal slope described above. In particular, it is important to understand how quickly bootstrapping converges for the different methods listed here. The method that converges with the nicest properties (for example, in

the higher-order moments of the bootstrapped distributions) will allow for the most efficient bootstrapping estimate of of confidence intervals around the marginal slope relationships.

3 Shapley Values

While PDP and ALE provide useful insights into the marginal effect of a feature – the change in model prediction as $x_i^{(k)}$ increases or decreases, they do not provide as much insight about the more general questions such as whether a feature should be included in the model in the first place, or which features are most important to a model's prediction. We can, however, answer these types of questions with Shapley values (Shapley, 1953; Štrumbelj and Kononenko, 2014) and we can extend our understanding by combining Shapley values PDP and bootstrapping.

The Shapley value $\psi_k f(x_i^{(k)}) \equiv \psi_k f(x_i^{(k)}, x_i^{(\neg k)})$ is the marginal contribution of a variable, k, to a model's prediction, $f(x_i)$, for a particular observation, i averaged over all possible combinations with covariates in x_i. Write the demeaned PDP of $f(x^{(l)} = q)$ as $\tilde{\nu}(x^{(l)} = q) = E_{x^{(\neg l)}}[f(q, x^{(\neg l)}) - E_x[f(x)]]$, then we can write the Shapley value of $x_i^{(k)}$ as

$$\psi_k f(x_i) = \frac{1}{K} \sum_{s \subseteq x_i^{\neg k}} \binom{K-1}{|s|}^{-1} \big(\tilde{\nu}(x_i^{(k)} \cup s) - \tilde{\nu}(s)\big). \tag{11}$$

Where s is a subset of covariates at values observed in $x_i^{\neg k}$ and K is the total number of variables. The Shapley decomposition of a model's output, $\Psi(f(x_i)) = \{\psi_1 f(x_i), \dots \psi_K f(x_i)\}$, is a linear decomposition that can be described as an additive feature attribution(Lundberg and Lee, 2017) of f i.e. $\sum_k \psi_k f(x_i^{(k)}) = f(x_i) - E[f(x)]$. This property of the Shapley decomposition makes interpretation straightforward; $\psi_k f(x_i^{(k)})$ tells us the gain[18] in model output on observation i from including variable k *at its value for observation* i, marginalized over its inclusion with all possible combinations of covariates as observed for observation i.

[18]This gain is expressed relative to $E[f(x)]$ allowing for comparison between different models.

To better understand the intuition and interpretation of the Shapley value, consider a linear model, $f(x) = xB$. OLS estimates of B provide a summary interpretation of the effect of x on $f(x)$. That is, $\beta_k = \frac{\partial f(x)}{\partial x^{(k)}}$ tells us about the overall effect of $x^{(k)}$ on $f(x)$ regardless of its observed value. By contrast, $\psi_k f(x_i^{(k)})$ incorporates the value of x_i directly[19]. In the linear model case[20], $\psi_k f(x_i^{(k)}) = (x_i^{(k)} - E[x^{(k)}])\beta_k$.

In non-parametric models and machine learning models specifically, models are much more complex than linear models and generally explore various nonlinearities in the data. In these circumstances, the effect of $x_i^{(k)}$ on $f(x_i^{(k)})$ is not necessarily independent of $x_i^{(\neg k)}$. By producing the average contribution of including $x_i^{(k)}$ over all possible combinations of observed covariates in $x_i^{(\neg k)}$, Shapley values resolve the attribution of the effect of $x_i^{(k)}$ on $f(x_i^{(k)})$ in a way that is 'fair'[21]. As a consequence however, $\psi_k f(x_i^{(k)})$ is dependent on the specific covariate profile $x_i^{(\neg k)}$.

We can gain a more general understanding of the effect of including $x^{(k)}$ in the model by marginalizing out the influence of the covariate profile, i.e. by calculating $E_{x^{(\neg k)}}[\psi_k f(x_i^{(k)})|x_i^{(k)}]$. We can estimate this quantity as $SPDP_k(x) = \nu(\psi_k f(x))$ and we can produce bootstrap estimates of $SPDP_k(x)$ by applying the following algorithm:

Note that the quantities contained in $\tilde{Z}$ are just a normalized version of the quantities in Z, which gives us the Shapley feature importance partial dependency plot (SFIPDP). The SFIPDP values can be plotted against quantiles of k to give a sense of how much influence k would have at a specific value and how likely it would be to actually observe such a value.

[19] i.e. $|q - E[x^{(k)}]| > |p - E[x^{(k)}]|$ then $|\psi_k f(q, x^{(\neg k)})| > |\psi_k f(p, x^{(\neg k)})|$, assuming f is linear.

[20] see Appendix C for discussion

[21] The properties of fairness are discussed in Young (1985). Crucially, for a decomposition to fairly describe feature attribution, it must be accurate (i.e. the sum of feature attributions must sum to the model output) and it must exhibit coalitional monotonicity whereby if $f(x_i^{(k)}) - E[f(x)] \geq g(x_i^{(k)}) - E[f(x)]$ then $\psi_k f(x_i^{(k)}) \geq \psi_k g(x_i^{(k)})$. See discussion in Lundberg and Lee (2017) for application of fairness to feature attributions specifically and in which the authors establish that the only linear decomposition approaches to feature attribution that can be described as fair are those that are derived from Shapley values.

Algorithm 2 Estimate SPDP and SFIPDP

Preallocate output matrix Z with dimensions $(J \times B)$
Preallocate output matrix $\tilde{Z}$ with dimensions $(J \times B)$
Allocate vector Q as a J-length vector of equally spaced values from $\left[x_{low}^{(k)},\ \ x_{high}^{(k)}\right]$
for for b in 1 to B: **do**:
  x_b, y_b =bootstrap(x)
  Estimate f_b s.t. $\hat{y_b} = f_b(x_b)$
  for for j in [1,J]: **do**
    $q = Q_j$
    $Z_j^{(b)} = \frac{1}{N} \sum_i \psi_k f(q, x_{bi}^{(\neg k)})$
    $\tilde{Z}_j^{(b)} = \frac{1}{N} \sum_i \frac{|\psi_k f(q, x_{bi}^{(\neg k)})|}{\sum_{h \in \{k, \neg k\}} |\psi_h f(x_{bi}^{(h)})|}$
Return Z as SPDP, and $\tilde{Z}$ as SFIPDP

3.1 Illustrating SPDP and SFIPDP via Simulation

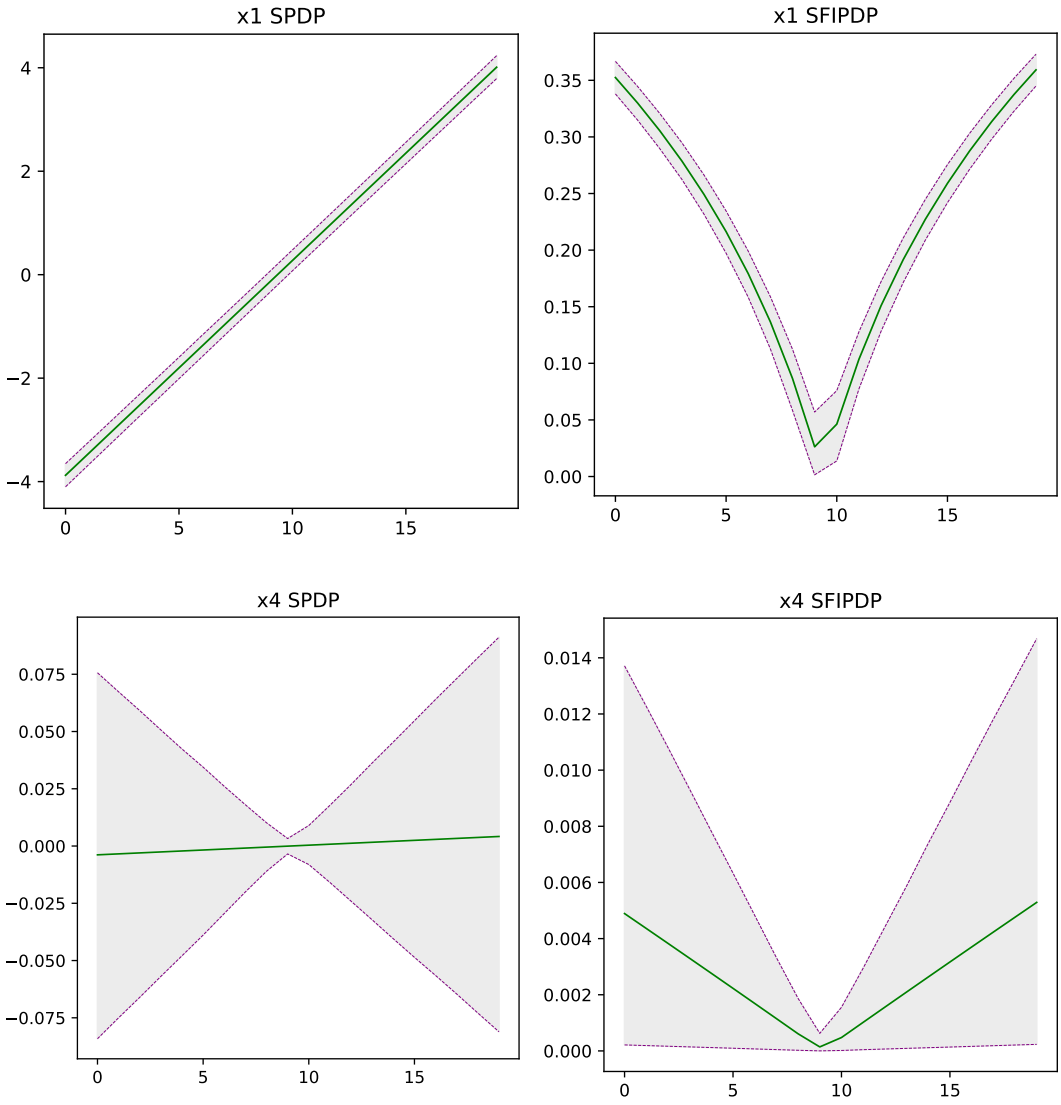
To help better understand the intuition behind SPDP and SFIPDP, we return to the simulated data discussed earlier. To begin, recall our initial dataset simulated from a simple linear DGP, $y = xB + u$ and with coefficients $B = (1, -2, 10, 0)$. Using the OLS model discussed in section 2.2 we can apply Algorithm 2 to generate SPDP and SFIPDP bootstrap estimates[22]. Figure 7 shows the SPDP and SFIPDP for $x^{(1)}$ and $x^{(4)}$. Recall from the previous section that $\psi_k f(x_i^{(k)}) = (x_i^{(k)} - E[x])\beta$ when f is linear. Accordingly, the SPDP for the variables shown in figure 7 are linear and exhibit slopes that are effectively 1 and 0 (equivalent to the OLS coefficient estimates). This is similar to the special case presented when the PDP is taken of an OLS model as in Section 2.1. For a model that is not a simple linear combination of covariates, however, this special case does not hold and SPDP is not necessarily equivalent to the demeaned PDP.

Figure 8 shows the SPDP and SFIPDP for the GBM model estimated on the simulated data from the linear DGP. As with the PDP, the SPDP is generally consistent with the OLS model – it portrays the marginal effect of including a variable as linear in the value of that variable and adhering, roughly, to the slope equiva-

[22]Though it is possible to generate $\Psi_k(f(x_i))$ for this simulated dataset by directly implementing equation 11, it is computationally costly to do so and the computational cost of generating $\Psi_k(f(x_i))$ grows exponentially in the number of features. Accordingly, and throughout the rest of this paper, we estimate $\Psi_k(f(x_i))$ using methods described in Lundberg and Lee (2017).

lent to the corresponding OLS parameter estimate. As with the PDP, the SPDP is considerably less smooth for GBM than OLS, but this is attributable largely to the binary-choice nature of the decision trees that comprise the GBM. We can observe a similar correspondence between SFIPDP for the GBM and OLS models.

Figure 7: SPDP and SFIPDP values for OLS model on linear simulated data



SPDP remains useful and intuitive to interpret when applied to more complex situations. Recall the nonlinear DGP discussed above where $y = (x^{(1)})^2\gamma + XB + u$ with $B = (0, 5, -5, \epsilon^+, \epsilon^-)$, $\gamma = 10$, and with (ϵ^+, ϵ^-) being positive and negative

values near zero. We estimate SPDP and SFIPDP for an OLS model specified to explicitly account for the nonlinearity in the DGP. Figure 9 shows the SPDP and SFIPDP for $x^{(1)}$, which has no effect in the DGP, its nonlinear transformation, which has a large effect, and an unrelated variable, and $x^{(5)}$, which has a trivial effect on the DGP. As just discussed, we can interpret the SPDP we would the demeaned PDP since they are equivalent for linear models.

The SPDP and SFIPDP for the GBM model are estimated and shown in Figure 10. Unlike the OLS model, the GBM model is capable of learning the nonlinear relationship between y and $x^{(1)}$ without the nonlinearity being explicitly specified. Thus, while the SPDP of $x^{(1)}$ for the OLS model shows no marginal effect from including (the untransformed value of) $x^{(1)}$, the SPDP of $x^{(1)}$ for the GBM model portrays the substantial, nonlinear effect of including $x^{(1)}$ that is learned by the model.

Under the circumstances of this exercise, where variables are largely independent of one another and our models are setup to estimate $E[y|x]$, then the $SFIPDP(x^{(k)})$ will tend towards a local minimum as $g(x^{(k)}) \to E[g(x^{(k)})]$, where g is the representation of the data learned by the model. In the case of the OLS model, $g(x^{(k)}) = x^{(k)}$, and thus $SFIPDP(x^{(k)})$ approaches its lowest point as $x^{(k)} \to E[x^{(k)}] = 0$. The SFIPDP results for the GBM model are similar, except with regard to $x^{(1)}$. Because the GBM model implicitly learns $g(x^{(1)}) \approx (x^{(1)})^2$, $SFIPDP(x^{(1)})$ for the GBM model approaches its lowest values as $g(x^{(1)}) \to E[g(x^{(1)})] = 1$. When plotted against $x^{(k)}$, then, $SFIPDP(x^{(k)})$ approaches its lowest as $x^{(k)} \to g^{-1}(E[g(x^{(k)})]) = \pm 1$.

Figure 8: SPDP and SFIPDP estimates from GBM model on linear simulated data

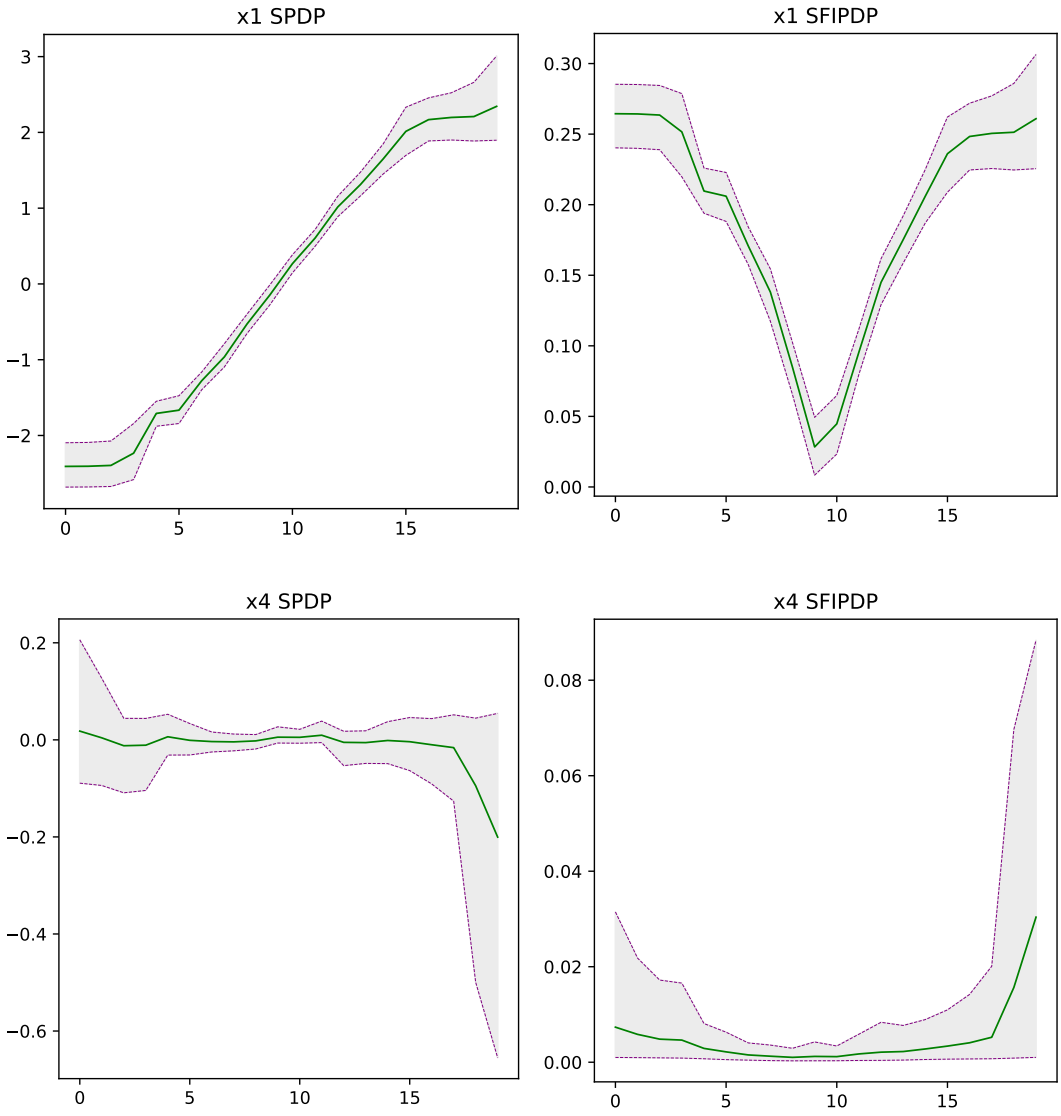


Figure 9: SPDP and SFIPDP estimates from OLS model on nonlinear simulated data

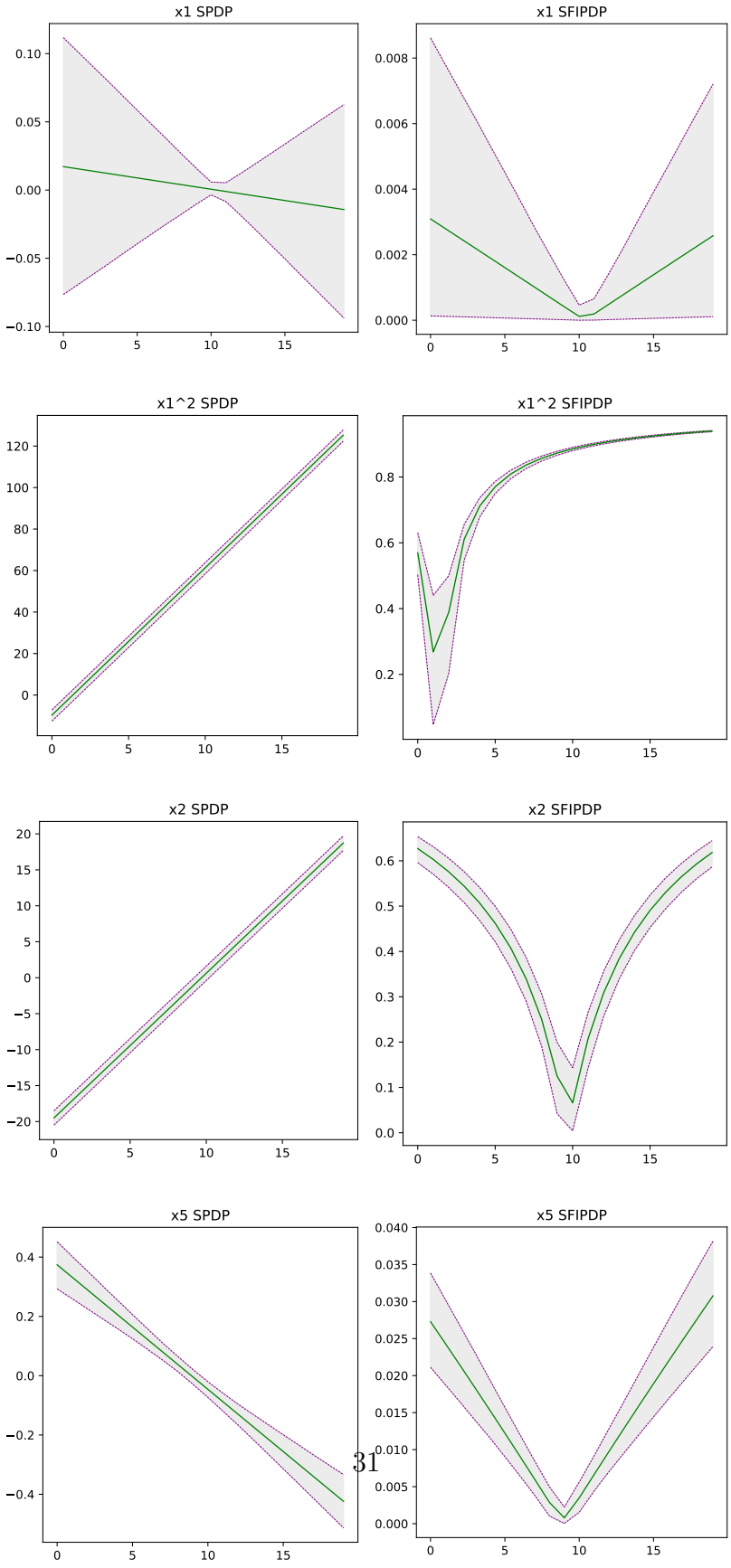
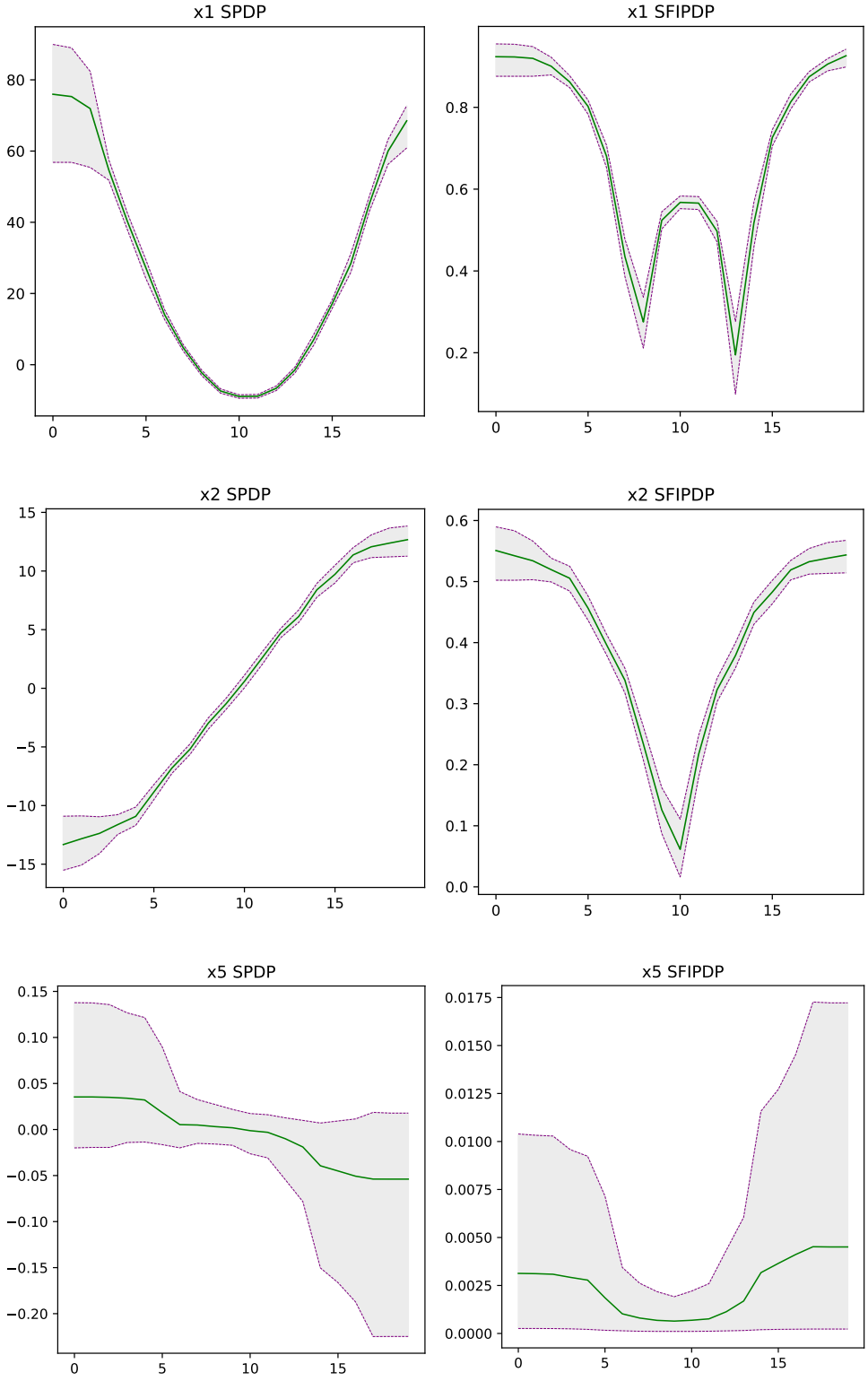


Figure 10: SPDP and SFIPDP estimates from GBM model on nonlinear simulated data



4 Applied Exercise: Ames Housing Data

To concretely illustrate the PDP, SPDP, and SFIPDP methods discussed in sections 2 and 3 we apply them to a hedonic house pricing model exercise run on tax assessor data. The data, described in De Cock (2011), is comprised of houses sold from 2006 through 2010 in Ames, Iowa. The data contains approximately 3000 observations with models spanning roughly 90 different variables, including square footage, number of beds and baths, number of fireplaces, neighborhood and amenities information.

Our hedonic model measures the response of the log of house prices to the most included dependent variables across the two meta-studies on hedonic house pricing, Sirmans, Macpherson, and Zietz (2005) and Zietz, Zietz, and Sirmans (2008).

Table 2: Model Specification

Target	Input Features	Additional Controls
Log(sale price)	Square Footage	Neighborhood
	Age	Sale Condition
	Lot Area	Central Air
	Garage Area	Condition 1
	Bathrooms	
	Bedrooms	
	Bathrooms:Bedrooms	
	Fireplaces	
	Time Trend	

The estimated model, for both OLS and the ML models[23], is presented in Table 2. In this discussion, we will focus on the results from five models. The simplest model is a linear model estimated via OLS. Two of the models are ensemble models based on decision-trees: Random Forest (RF) and Gradient Boosting Regression (GBR). These ensemble models are notable in that they essentially learn complex piecewise functions to fit the data. By contrast, the other two models we will discuss, Support Vector Regression (SVR) and Deep Neural Networks (DNN) learn 'smooth', continuous functions that fit the data. We will further discuss the difference between the smooth and non-smooth models shortly.

[23]Machine learning models have been widely explored for house pricing models. However, these studies either focus on the accuracy of the ML model (Limsombunchai, 2004; McCluskey et al., 2013, See) or focus on model-specific interpretation of less-complex ML type models (See, for example, Čeh et al., 2018; McMillen and Redfearn, 2010).

Generally, each of the five models performed well. In-sample and out of sample R^2 scores are presented in Table 3. The linear model exhibits inferior fit to the other models while the Random Forest model exhibits near perfect fit. The difference between the linear model and the ML models is less striking with regard to out of sample fit (R^2 measured via 10-fold cross-validation), where the OLS model performs better than the DNN and only slightly worse than other models.

Table 3: OLS and ML model performance

	OLS	GBR	RF	SVR	DNN
R^2	0.847	0.897	0.979	0.914	0.887
CV R^2	0.837	0.842	0.841	0.839	0.808

4.1 PDP of Linear Model and OLS Estimates

Turning to the OLS model results, we can interpret the effect of each of the variables in Table 2 by examining the OLS coefficient estimates. These are presented in Table 4. As with the exercise in section 2.2, we can also recover these estimates from the PDP. The PDP based coefficient estimates are presented in the second column of Table 4 along with bootstrap estimates of the standard error. The PDP-based coefficient estimates are equal to the OLS based estimates (to machine precision) and the corresponding estimates of standard errors are only slightly different.

The OLS estimates show effects that are generally consistent with findings documented in the meta analyses in Sirmans, Macpherson, and Zietz (2005) and Zietz, Zietz, and Sirmans (2008). That is, it finds square footage, lot size, number of bathrooms, garage area and number of fireplaces as statistically significant with a positive effect on house price while Age is statistically significant with a negative effect. The time trend is not estimated to be statistically significant, which fits with the mixed findings regarding the time trend in Sirmans, Macpherson, and Zietz (2005). The total effect of bedrooms is estimated as negative as long as the home has at least 1 bathroom, but not necessarily always statistically significant. This is likewise consistent with the mixed findings in Sirmans, Macpherson, and Zietz (ibid.).

Table 4: OLS and PDP-derived model estimates

	OLS	Δ PDP
Square Footage	0.168*	0.168*
	(0.007)	(0.005)
Age	-0.073*	-0.073*
	(0.009)	(0.009)
Lot Area	0.018*	0.018*
	(0.005)	(0.006)
Bedrooms	0.012	0.012
	(0.009)	(0.008)
Bathrooms	0.058*	0.058*
	(0.012)	(0.012)
Bed x Bath	-0.043*	-0.043*
	(0.015)	(0.014)
Garage Area	0.043*	0.043*
	(0.006)	(0.006)
Fireplaces	0.038*	0.038*
	(0.004)	(0.003)
Time Trend	-0.001	-0.001
	(0.003)	(0.003)
Adj. R^2	0.845	0.845
N	2874	2874

Robust standard errors presented in parentheses for OLS model. Bootstrap estimated standard errors presented for Δ PDP model.

4.2 ML Models and PDPs

Turning to the ML models, recall that the nonlinear nature of the ML models makes it difficult to present the marginal effects of the ML models in tabular form. Instead, we turn to an examination of the PDP for these models. Figures 11 and 12 provide a comparison of the PDP of the linear model to the smooth (SVM, DNN) and non-smooth (GBM, RF) models for select variables[24]. These figures show distinct non-linear effects for each of the variables shown. Notably, for a number of the variables, we see diminishing effects for a number of variables as they approach values far outside of the mean.

[24]See the appendix for the PDP of other variables

For age in particular, the PDP suggests substantial differences in the nature of the relationship captured from one model to the next. As a reference point, the linear model portrays a constant negative effect whereby a house's sale price declines at a consistent rate as it ages[25]. The RF and GBM models pictured in Figure 12 however, suggest that this effect is not constant; the PDP shows steep declines in house price each year for its first 30 or so years. After that, however, the effect of age tapers substantially. For the SVM (pictured in Figure 11), the relationship suggests that age has a consistent negative effect on the price of a house for about its first 50 years but this effect then wanes and even reverses course as the age of the house increases past 100 years. A similar nonlinearity is revealed for lot size. Whereas the PDP of the linear model portrays a constant, weak effect, the PDP of ML models suggest a strong effect for lot sizes at or below the mean (about 10000 square feet, or 1/4 of an acre).

[25]This nonlinearity in the effect of age on house price is well known. See Goodman and Thibodeau (1995). Our purpose in discussing it here is to highlight (1) that the ML models account for the nonlinearity without prior specification and (2) that PDPs allow us to depict the nonlinear relationship that the ML models have learned.

Figure 11: PDP of linear model and smooth ML models

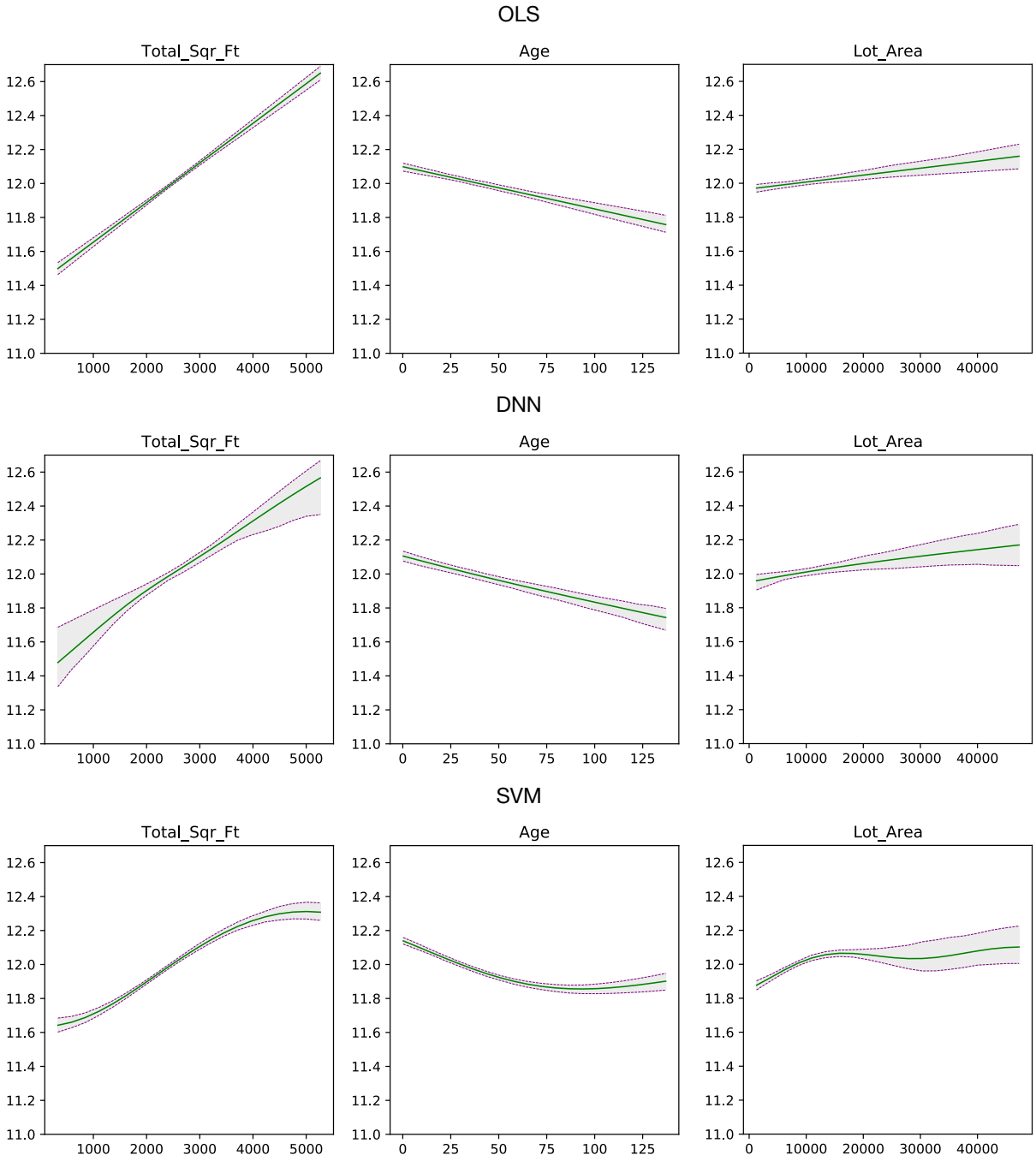


Figure 12: PDP of linear model and non-smooth ML models

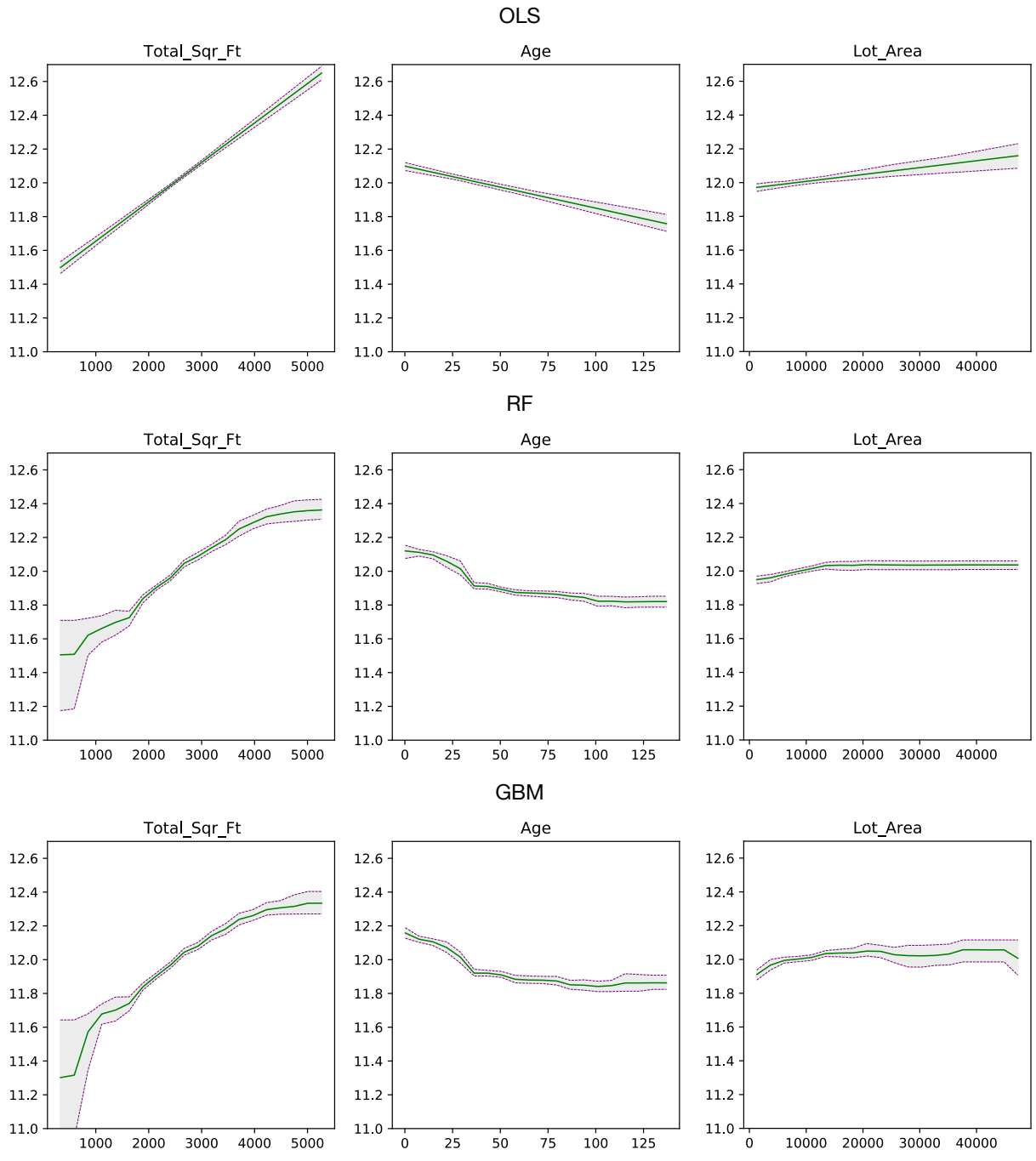
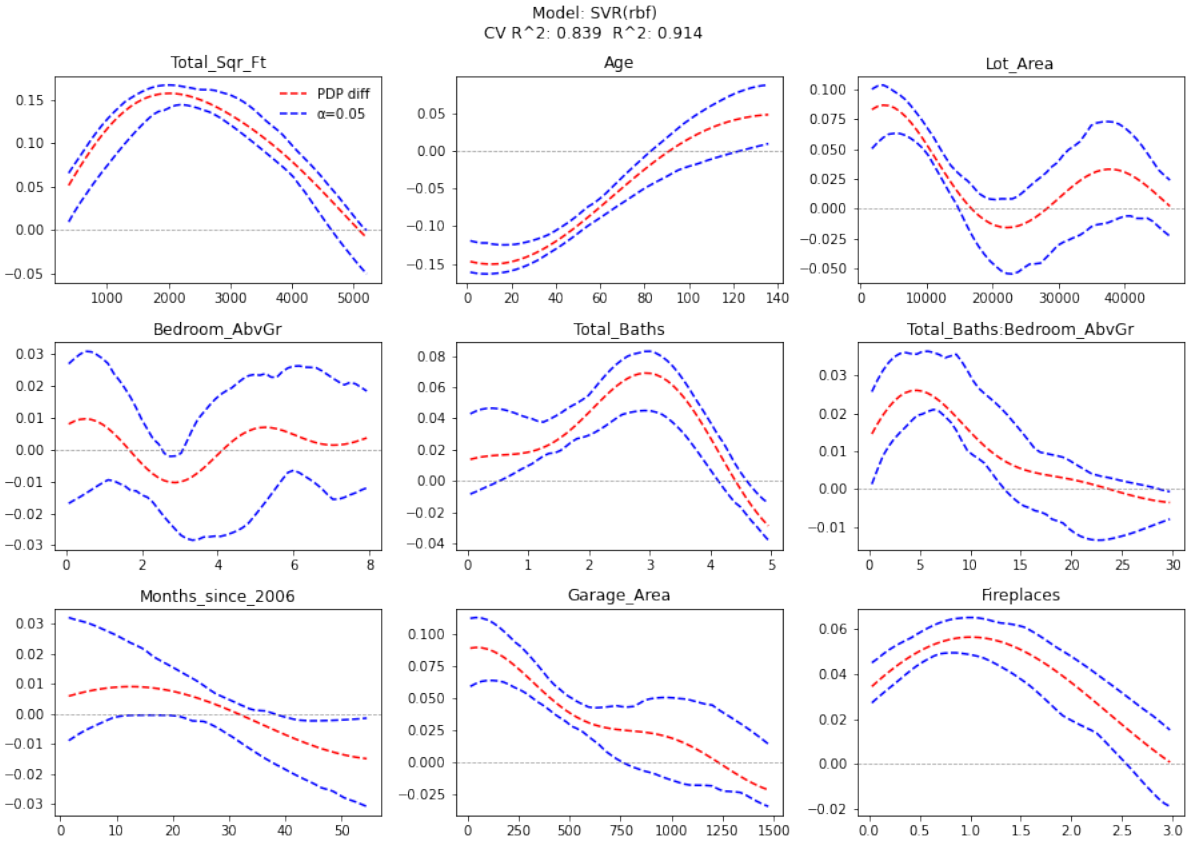


Figure 13: SVM PDP-Differenced Visual Regression Table



If we look at the PDP in first-differences, the nonlinearities are placed in higher relief. Figure 15 shows the PDP in first differences for the linear, SVM and DNN models for selected variables. If we look at Figure 11, the effect of age on house price for the DNN appears linear. If, however, we examine the first differences as shown in Figure 15, we see that the effect of age tapers gradually over its entire range. If we consider the bootstrapped confidence intervals, the effect becomes indistinguishable from zero for houses older than about 80 years. These first difference plots similarly reveal slight but discernable nonlinearities for lot area and total square footage. These first difference plots are useful for interpreting the SVM as well. For example, they more clearly suggest, when taking the bootstrapped confidence intervals into account, that the effect of lot area goes to zero as a lot exceeds about 15,000 square feet.

However, for tree-based models the PDP-difference slope is likely not the best tool due to the fact that they yield piecewise, 'non-smooth', model functions. Figure 16 shows the PDP in first differences for the linear, RF, and GBM models. We can see that these plots are rather volatile; non-zero effects are only observed in the differences between points that straddle 'jumps' in the piecewise model function[26]. The first-difference of the PDP can be made more legible by evaluating it on fewer points over the range of the input variable, but this comes with the risk of imprecision in interpretation. Instead, for tree based models, it may be somewhat preferable to interpret the PDP directly.

4.3 Feature importance and marginal effect of inclusion via SPDP and SFIPDP

Recall that SPDP and SFIPDP are complimentary measures to PDP. In general, the PDP values can be interpreted as β-coefficient estimates in the spirit of statistical regression pedagogy. That is, we can think of the PDP as telling us about the effect of a variable as it increases or decreases and taking as a given that the variable will be included in the model. The SPDP is subtly different in that it tells us about the expected effect of including a variable in a model at a given value. The SFIPDP is

[26]To think of this another way, consider that tree-based methods will produce the same prediction for two observations that are sufficiently close together in the input feature space. As such, the PDP will be the same for two values that are sufficiently close together and thus produce a first-difference of zero

Figure 14: OLS PDP-Differenced Visual Regression Table

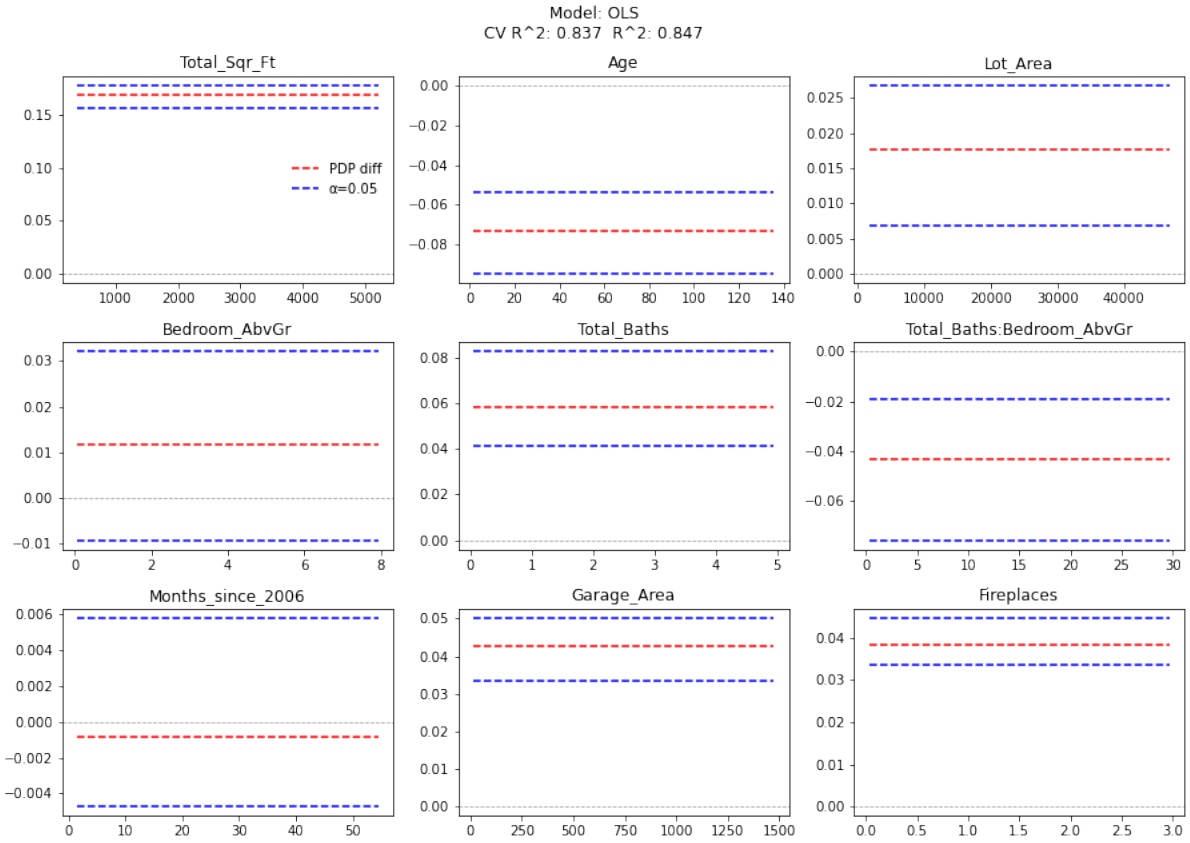
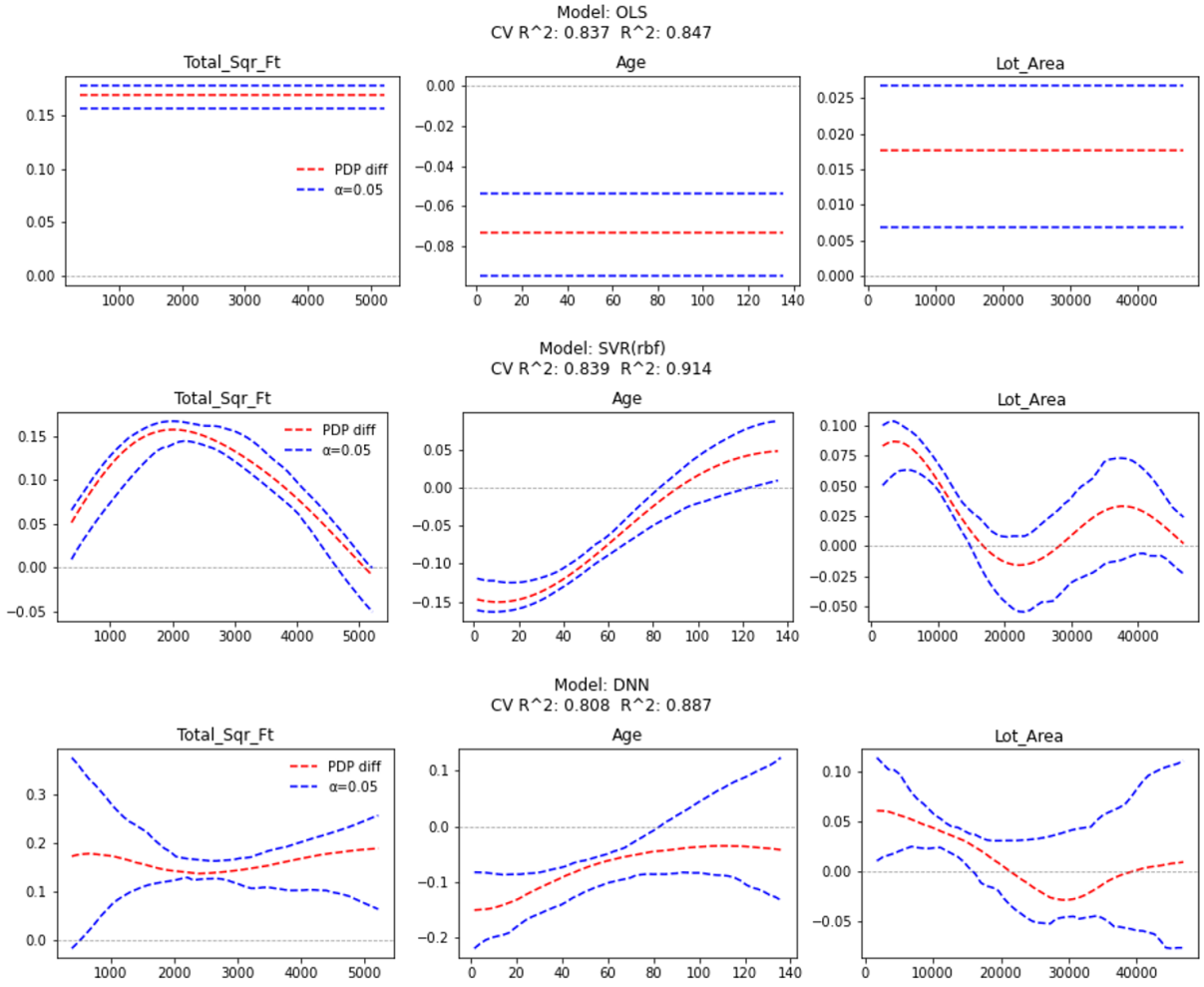


Figure 15: PDP First Differences for smooth models (SVM, DNN)

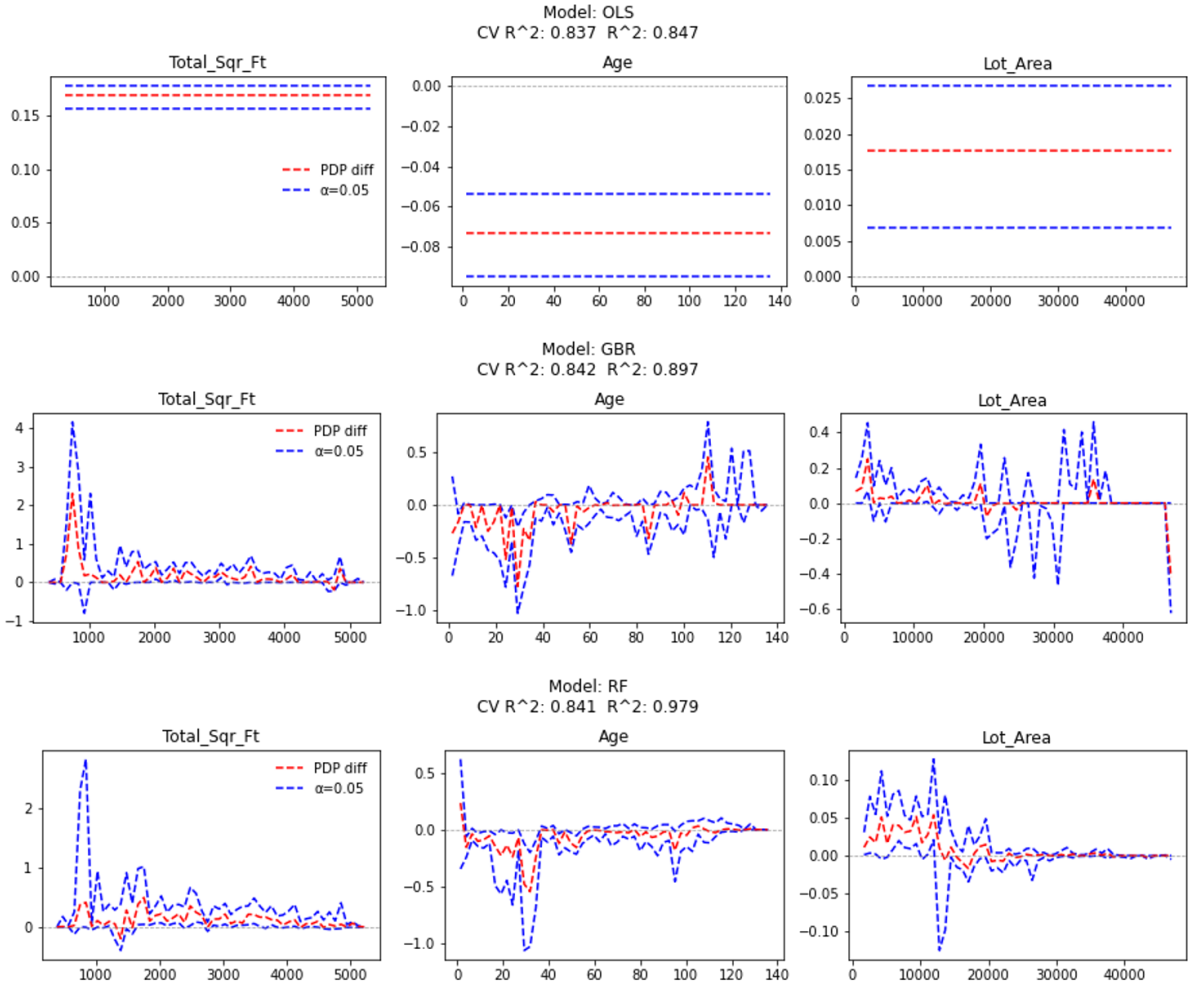


more distinct from the PDP and can be thought of as a measure of feature importance. As such, the SFIPDP enables us to compare input features to one another in terms of their overall impact on model output.

Figures 17 and 17 plot SPDP and SFIPDP results for Total Square Feet, Year Built, and Lot Area as constructed for OLS, SVR, and GBM modelling approaches. As discussed in Section 3, the SPDP for the linear model is linear with a slope equal to the estimated OLS coefficients.

For the SVM model, the nonlinearities are less pronounced for some variables (square footage and age) but are more readily apparent for others (lot area). It is

Figure 16: PDP First Differences for non-smooth models (GBM, RF)



also notable that for age, the SPDP of the SVM model is nearly flat and close to zero. This suggests that the inclusion of age has a relatively limited impact on the model output. The corresponding panel from Figure 18 supports this, showing that, on average, the inclusion of age in the SVM model accounts for between 5% and 10% of the model output[27].

For the GBM model, the SPDP follow the same general direction of the linear model, but are nonlinear, reflecting the nonlinearities captured by these models. Un-

[27]It is additionally notable that both the SPDP and SFIPDP for the SVM exhibit rather wide bootstrap confidence intervals (for age and other variables, when compared to OLS and GBM models). This suggests sensitivity to the composition of the training data that might otherwise be corrected through more careful hyperparameter tuning.

like the SVM model, the SPDP of age for the GBM model in figure 17 is quite sizable. Here, the the inclusion of the age of a home variable will, on average, reduce the estimated price by about 20% for houses more than 80 years old. For houses under 30 years old, however, the inclusion of the age variable would, on average increase the estimated home price, with the size of the effect increasing sharply, resulting in an average increase of the estimated log price by approximately 30% for the newest homes. In terms of relative importance, the GBM model appears to place more weight on the age variable than the SVM model. This is highlighted in Figure 18 where we can see the SFIPDP of the age variable for the GBM model accounts for as much as 50% of the model output[28]

5 Conclusion

This paper has explored the use of model-agnostic tools to aid in the interpretation of machine learning models in a way that is familiar to our interpretation of standard econometric models. It identifies PDPs as a viable route towards understanding the magnitude of effect and provides extensions to shapley values to understand the general impact of including a variable on a model's output. Further, it demonstrates that PDPs, Shapley Values, and SPDP, all replicate the OLS point estimate (β) in the linear case and that, for the PDP in particular, the nonparametric bootstrap produces variances in those point estimates comparable to OLS standard errors. Understanding these tools in the linear context, we expect that researchers will feel more comfortable using them to interpreting ML models in the future. The hedonic house-pricing exercise demonstrates such an application and shows that ML models can both replicate the results of meta-analysis of the literature, and add new insights about non-linear behavior of variables in the house-pricing models.

Despite the promise of PDPs, shapley values and the SPDP, their use in interpreting 'black-box' models is not without caveats. Perhaps most crucially, these techniques require nuanced interpretation where interactions between variables are concerned. Indeed a crucial avenue for further development will be the extension of these techniques to more fully illustrate interactions captured by a model.

[28]There is a notable decline in the SFIPDP as the age variable approaches the average age of homes in the dataset. As discussed in Section 3, this is expected behavior for the SFIPDP.

References

Abdi, Hervé (2004). "Partial regression coefficients". *Encyclopedia of social sciences research methods*, pp. 1–4.

Apley, Daniel W and Jingyu Zhu (2020). "Visualizing the effects of predictor variables in black box supervised learning models". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 82.4, pp. 1059–1086.

Athey, Susan and Guido W Imbens (2019). "Machine learning methods that economists should know about". *Annual Review of Economics* 11, pp. 685–725.

Bertrand, Marianne and Sendhil Mullainathan (2004). "Are Emily and Greg more employable than Lakisha and Jamal? A field experiment on labor market discrimination". *American economic review* 94.4, pp. 991–1013.

Breiman, Leo (2001). "Statistical modeling: The two cultures". *Statistical science* 16.3, pp. 199–231.

Čeh, Marjan et al. (2018). "Estimating the performance of random forest versus multiple regression for predicting prices of the apartments". *ISPRS international journal of geo-information* 7.5, p. 168.

Coulombe, Philippe Goulet (2021). *The Macroeconomy as a Random Forest*. arXiv: 2006.12724 [econ.EM].

De Cock, Dean (2011). "Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project". *Journal of Statistics Education* 19.3.

Efron, Bradley and Trevor Hastie (2016). *Computer age statistical inference*. Vol. 5. Cambridge University Press.

Friedman, Jerome H (2001). "Greedy function approximation: a gradient boosting machine". *Annals of statistics*, pp. 1189–1232.

Goldstein, Alex et al. (2015). "Peeking inside the black box". *Journal of Computational and Graphical Statistics* 24.1, pp. 44–65.

Goodman, Allen C and Thomas G Thibodeau (1995). "Age-related heteroskedasticity in hedonic house price equations". *Journal of Housing Research*, pp. 25–42.

Hanmer, Michael J and Kerem Ozan Kalkan (2013). "Behind the curve: Clarifying the best approach to calculating predicted probabilities and marginal effects from limited dependent variable models". *American Journal of Political Science* 57.1, pp. 263–277.

James, Gareth et al. (2013). *An introduction to statistical learning*. Springer Science & Business Media.

Limsombunchai, Visit (2004). "House price prediction: hedonic price model vs. artificial neural network". In: *New Zealand agricultural and resource economics society conference*, pp. 25–26.

Lundberg, Scott and Su-In Lee (2017). "A unified approach to interpreting model predictions". *CoRR* abs/1705.07874. arXiv: 1705.07874. URL: http://arxiv.org/abs/1705.07874.

McCluskey, William J et al. (2013). "Prediction accuracy in mass appraisal: a comparison of modern approaches". *Journal of Property Research* 30.4, pp. 239–265.

McMillen, Daniel P and Christian L Redfearn (2010). "Estimation and hypothesis testing for nonparametric hedonic house price functions". *Journal of Regional Science* 50.3, pp. 712–733.

Molnar, Christoph (2021). *Interpretable machine learning*. mimeo.

Pearl, Judea (2009). "Causal inference in statistics: An overview". *Statistics surveys* 3, pp. 96–146.

Semenova, Lesia, Cynthia Rudin, and Ronald Parr (2019). "A study in Rashomon curves and volumes". *arXiv preprint 1908.01755*.

Shapley, L. S. (1953). "Stochastic Games". *Proceedings of the National Academy of Sciences* 39.10, pp. 1095–1100. ISSN: 0027-8424. DOI: 10.1073/pnas.39.10.1095. eprint: https://www.pnas.org/content/39/10/1095.full.pdf. URL: https://www.pnas.org/content/39/10/1095.

Sirmans, Stacy, David Macpherson, and Emily Zietz (2005). "The composition of hedonic pricing models". *Journal of real estate literature* 13.1, pp. 1–44.

Štrumbelj, Erik and Igor Kononenko (2014). "Explaining prediction models and individual predictions with feature contributions". *Knowledge and information systems* 41.3, pp. 647–665.

Young, H Peyton (1985). "Monotonic solutions of cooperative games". *International Journal of Game Theory* 14.2, pp. 65–72.

Zhao, Qingyuan and Trevor Hastie (2021). "Causal interpretations of black-box models". *Journal of Business & Economic Statistics* 39.1, pp. 272–281.

Zietz, Joachim, Emily Norman Zietz, and G Stacy Sirmans (2008). "Determinants of house prices: a quantile regression approach". *The Journal of Real Estate Finance and Economics* 37.4, pp. 317–333.

Figure 17: SPDP for Linear, SVM and GBM models for selected variables

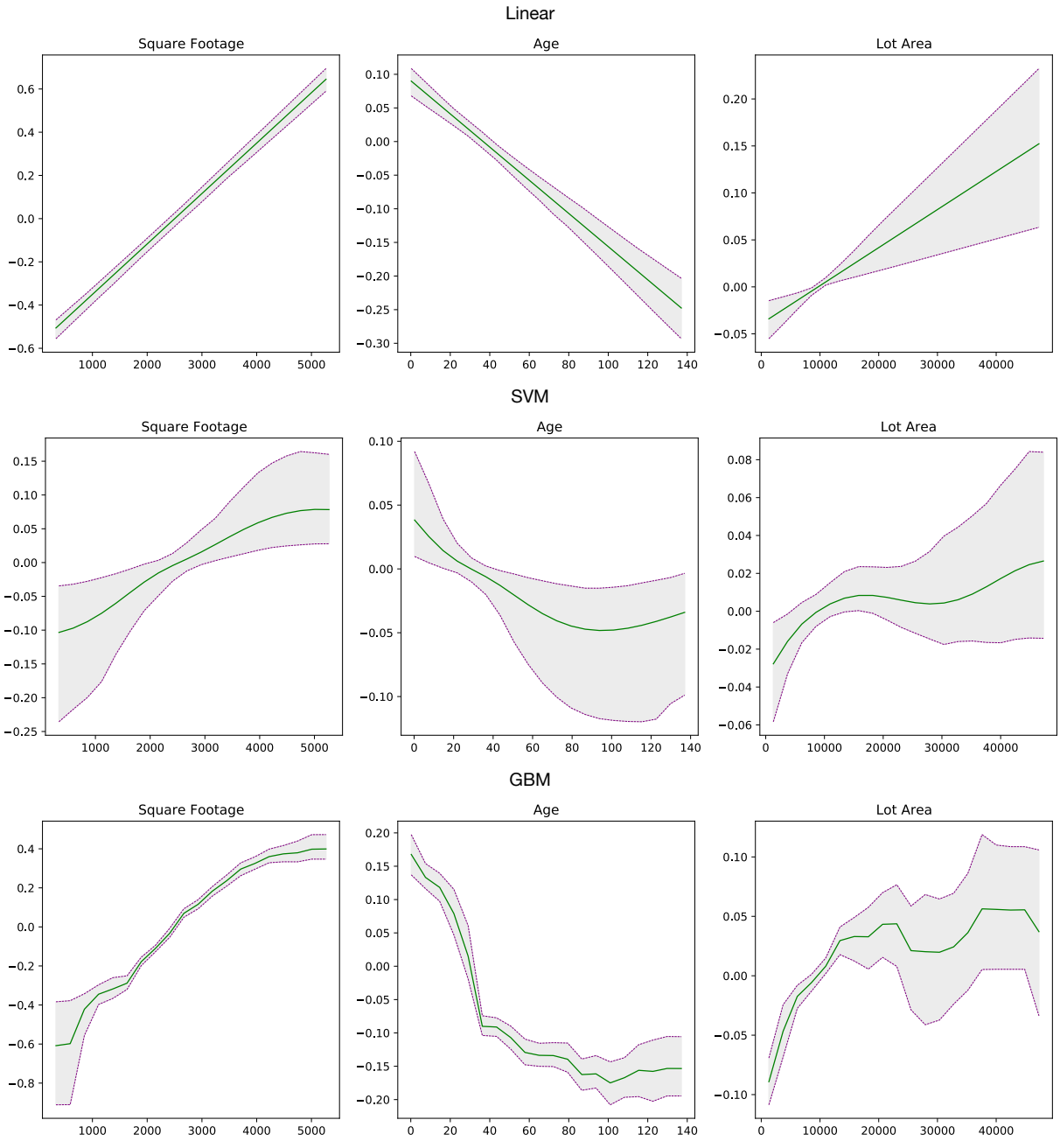
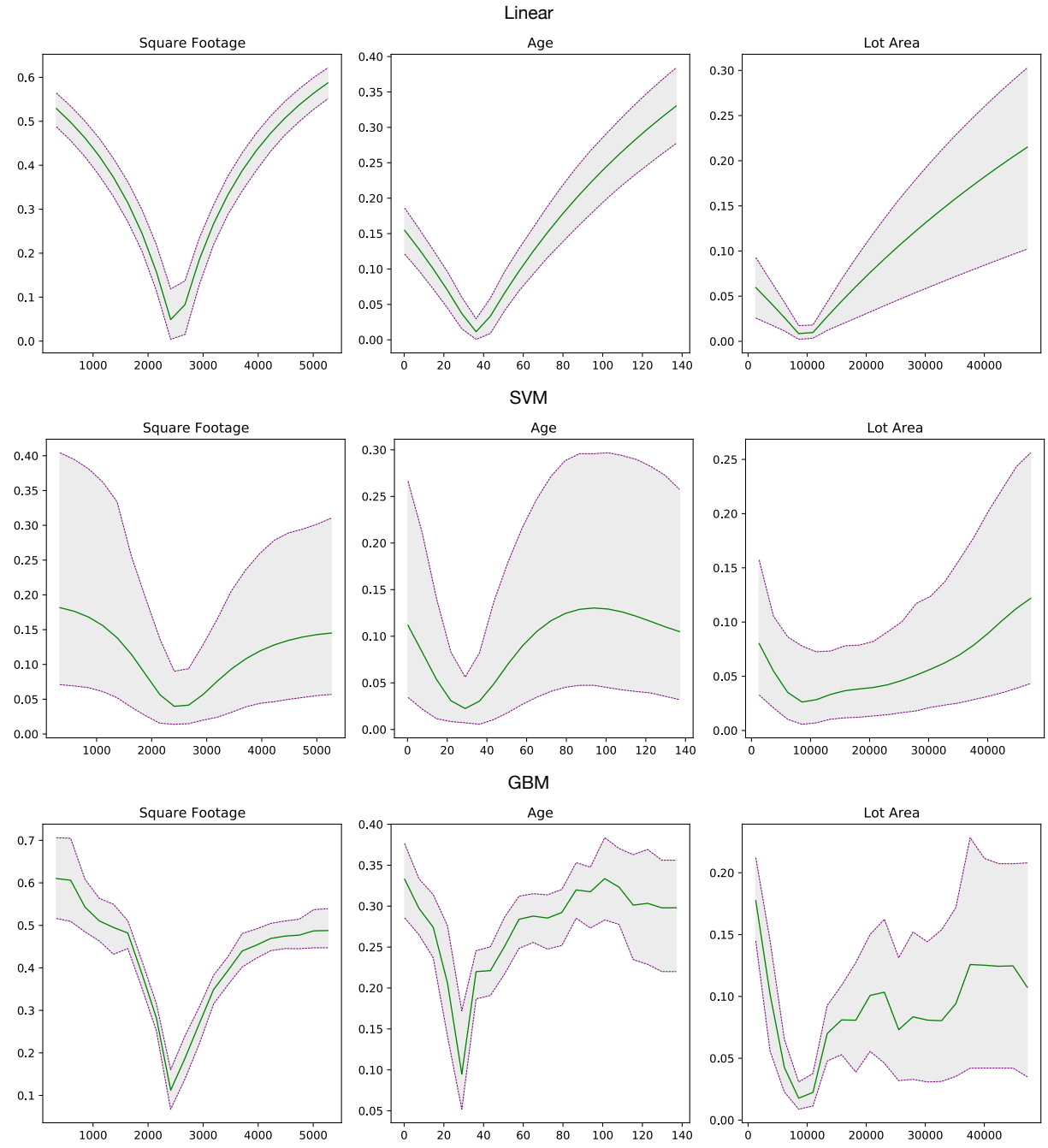


Figure 18: SFIPDP for Linear, SVM and GBM models for selected variables



A Nonlinear DGP Simulation OLS Fit

Table 5: Parameter estimates on simulated dataset from nonlinear DGP

Dep. Variable:	y	**R-squared (uncentered):**	0.999
Model:	OLS	**Adj. R-squared (uncentered):**	0.998
No. Observations:	3000	**AIC:**	6379.
Df Residuals:	2994	**BIC:**	6416.
Df Model:	6		

	coef	**std err**	**z**	**P> |z|**	**[0.025**	**0.975]**
$x^{(1)}$	-0.0050	0.013	-0.378	0.706	-0.031	0.021
$x^{(2)}$	4.9937	0.013	384.767	0.000	4.968	5.019
$x^{(3)}$	-4.9939	0.013	-378.003	0.000	-5.020	-4.968
$x^{(4)}$	0.0484	0.012	3.965	0.000	0.024	0.072
$x^{(5)}$	-0.1209	0.013	-9.503	0.000	-0.146	-0.096
$(x^{(1)})^2$	10.0002	0.008	1273.826	0.000	9.985	10.016

Omnibus:	0.113	**Durbin-Watson:**	2.008
Prob(Omnibus):	0.945	**Jarque-Bera (JB):**	0.117
Skew:	-0.015	**Prob(JB):**	0.943
Kurtosis:	2.994	**Cond. No.**	1.71

B PDP Plots for Housing Exercise

Figure 19: PDP for GBM model

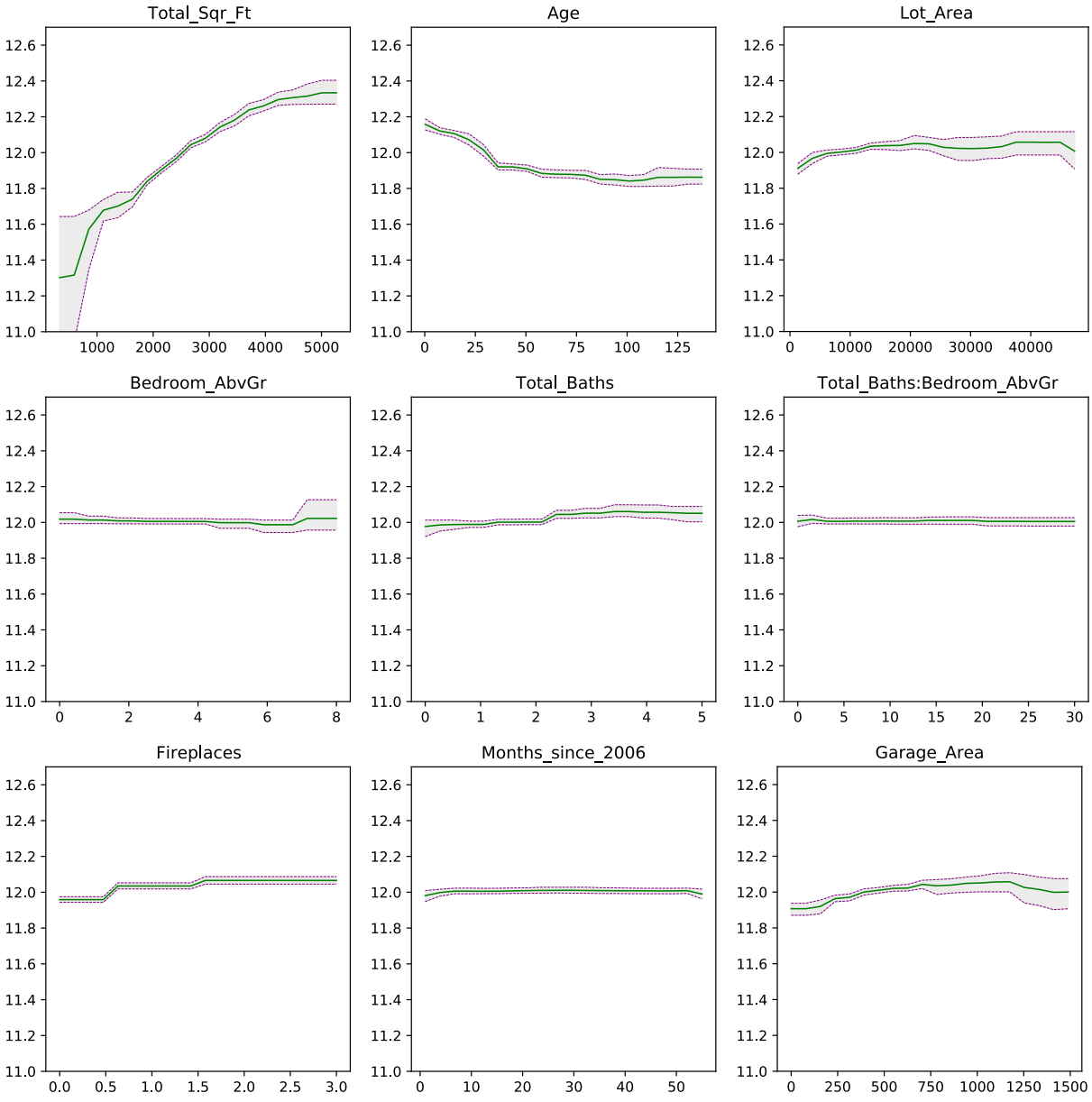


Figure 20: PDP for RF model

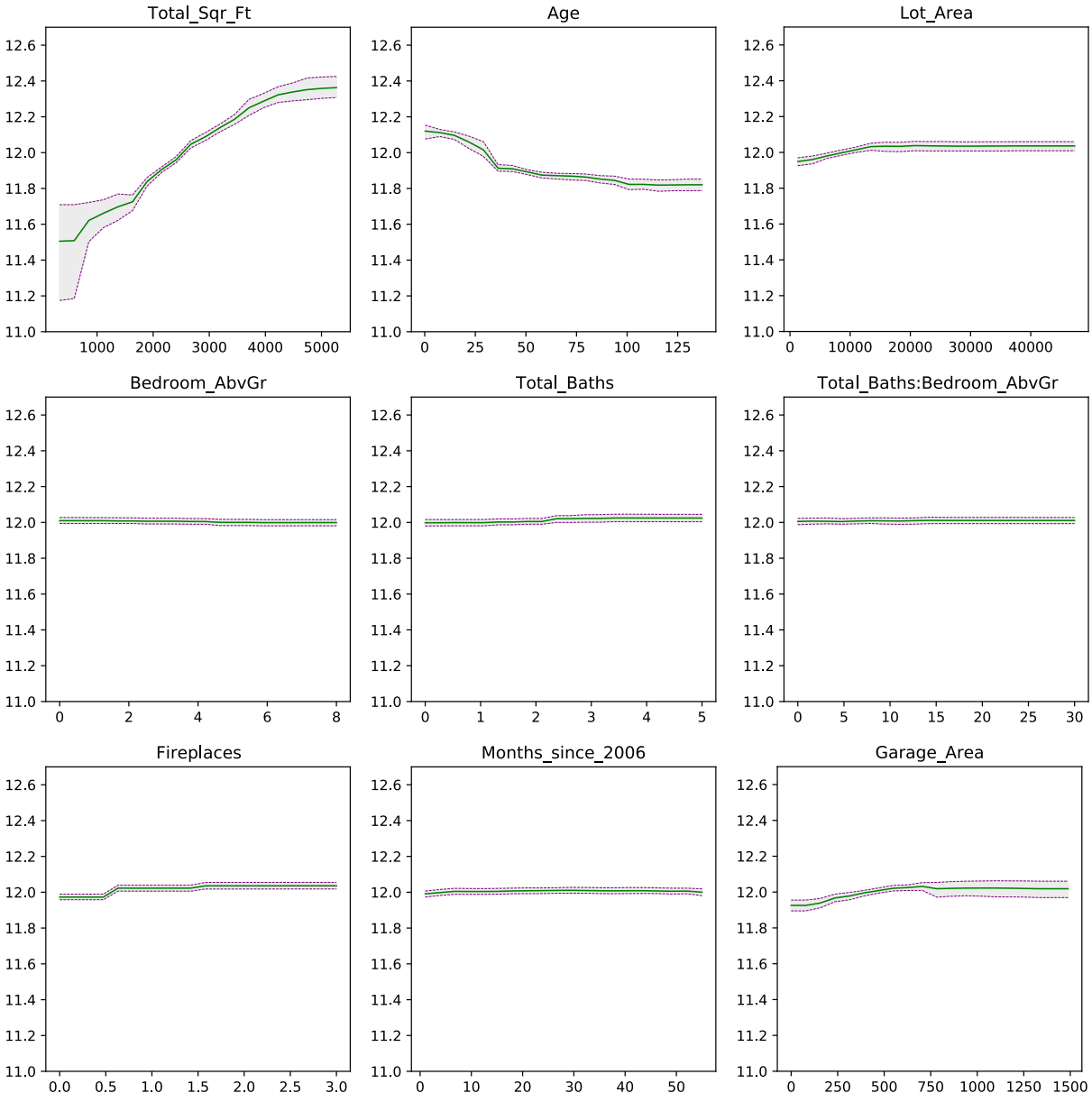


Figure 21: PDP for SVM model

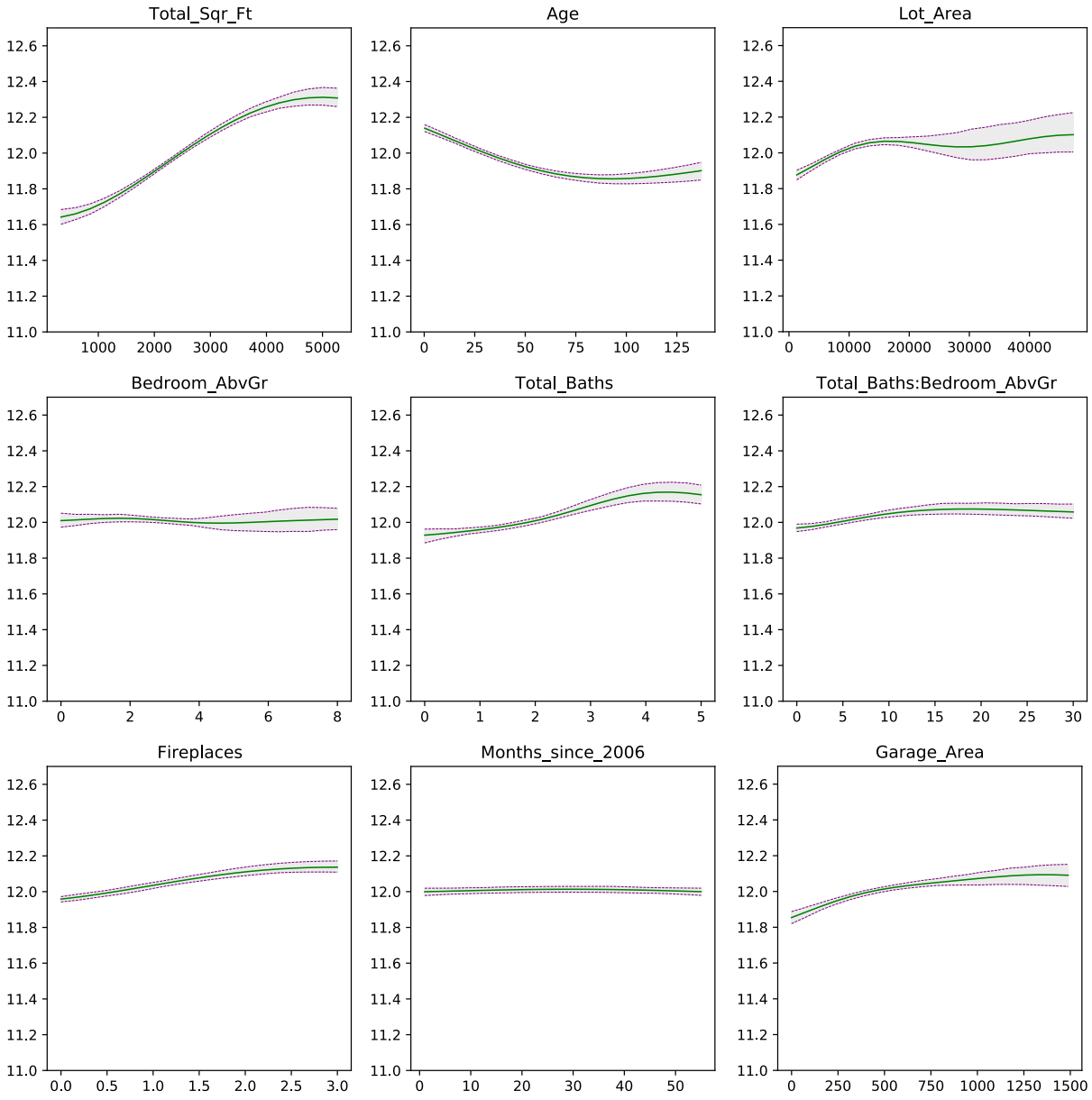


Figure 22: PDP for OLS model

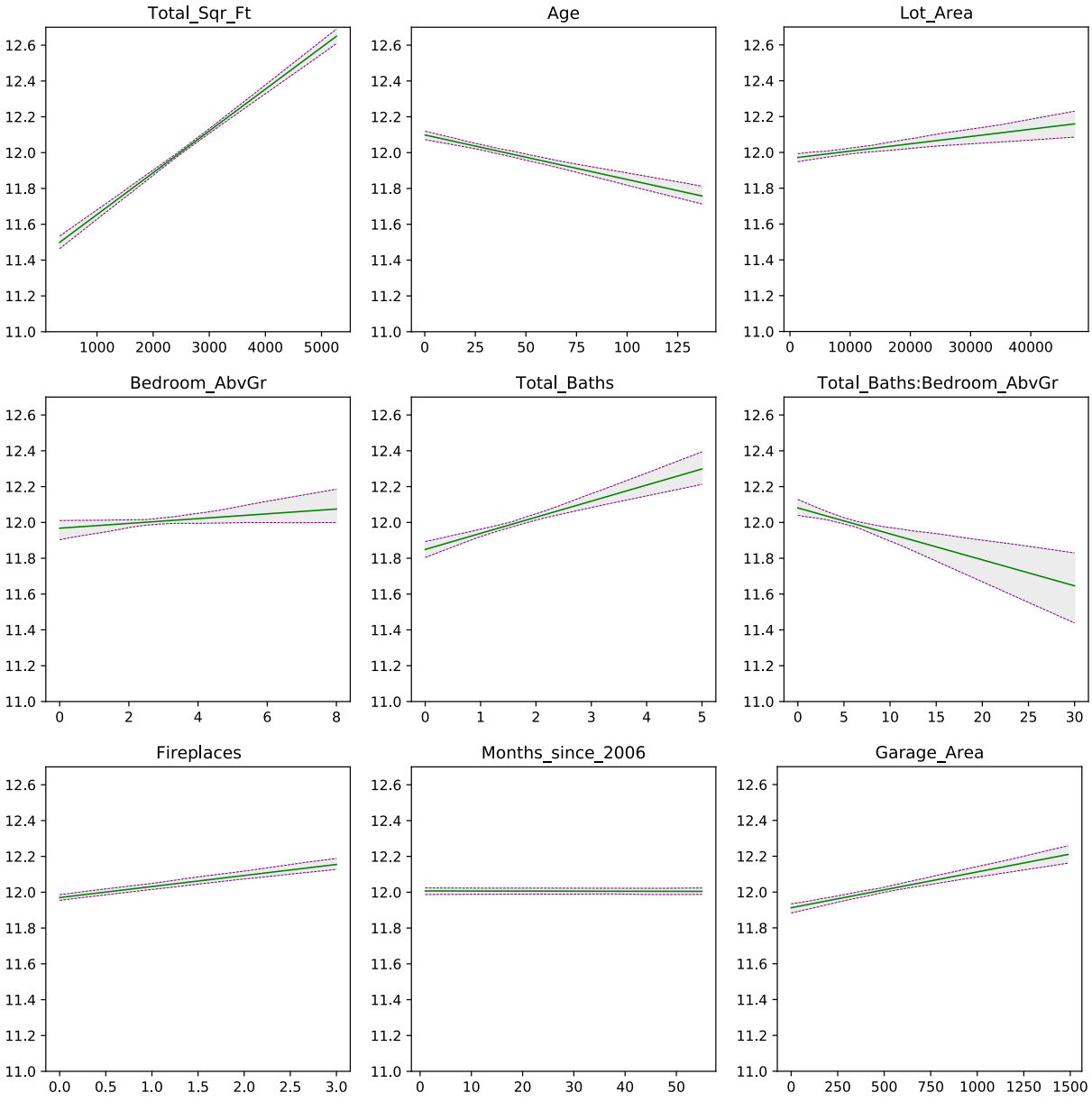
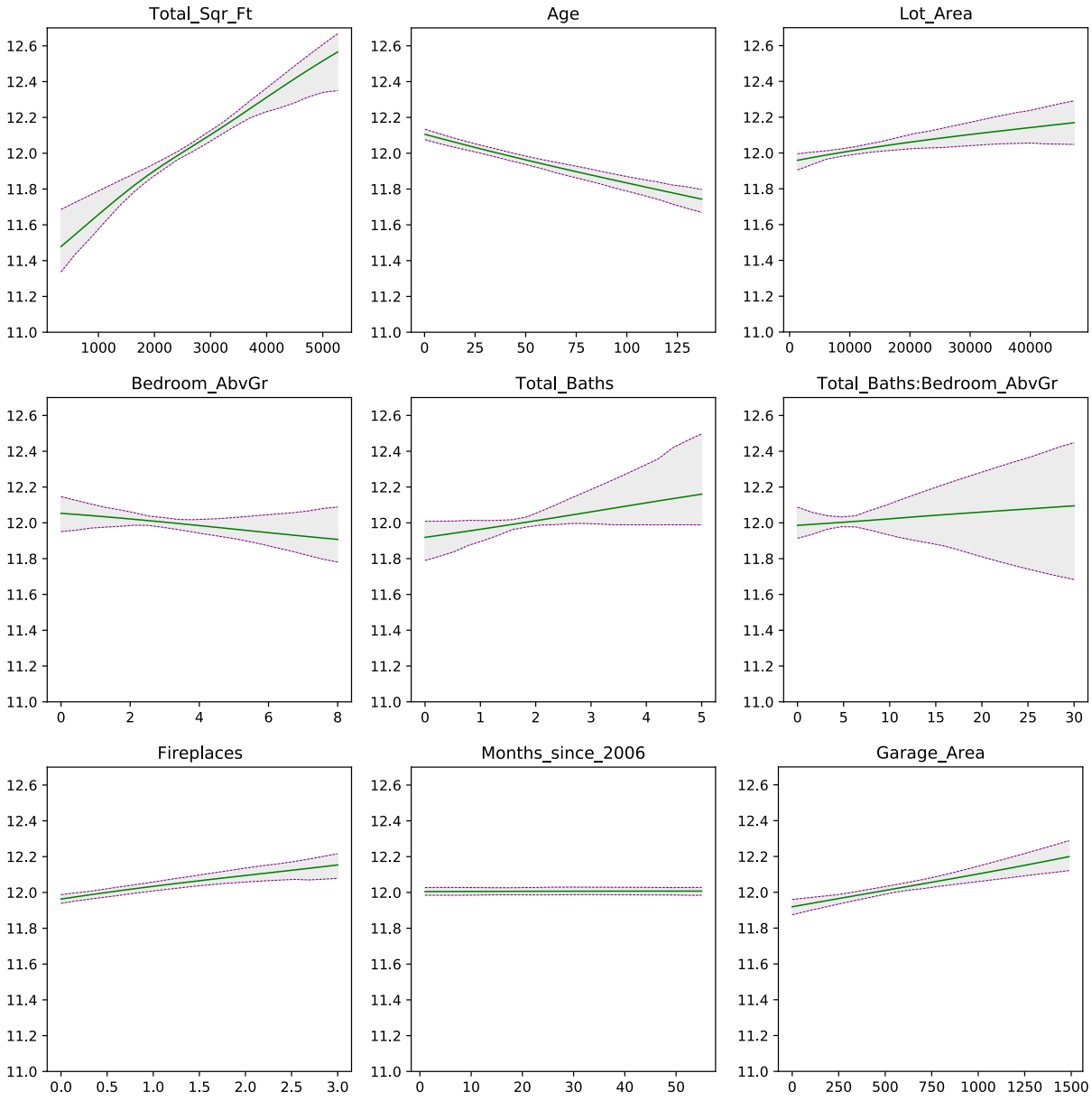


Figure 23: PDP for DNN model



B.1 Differenced PDP Plots

Figure 24: Differenced PDP for Linear Model

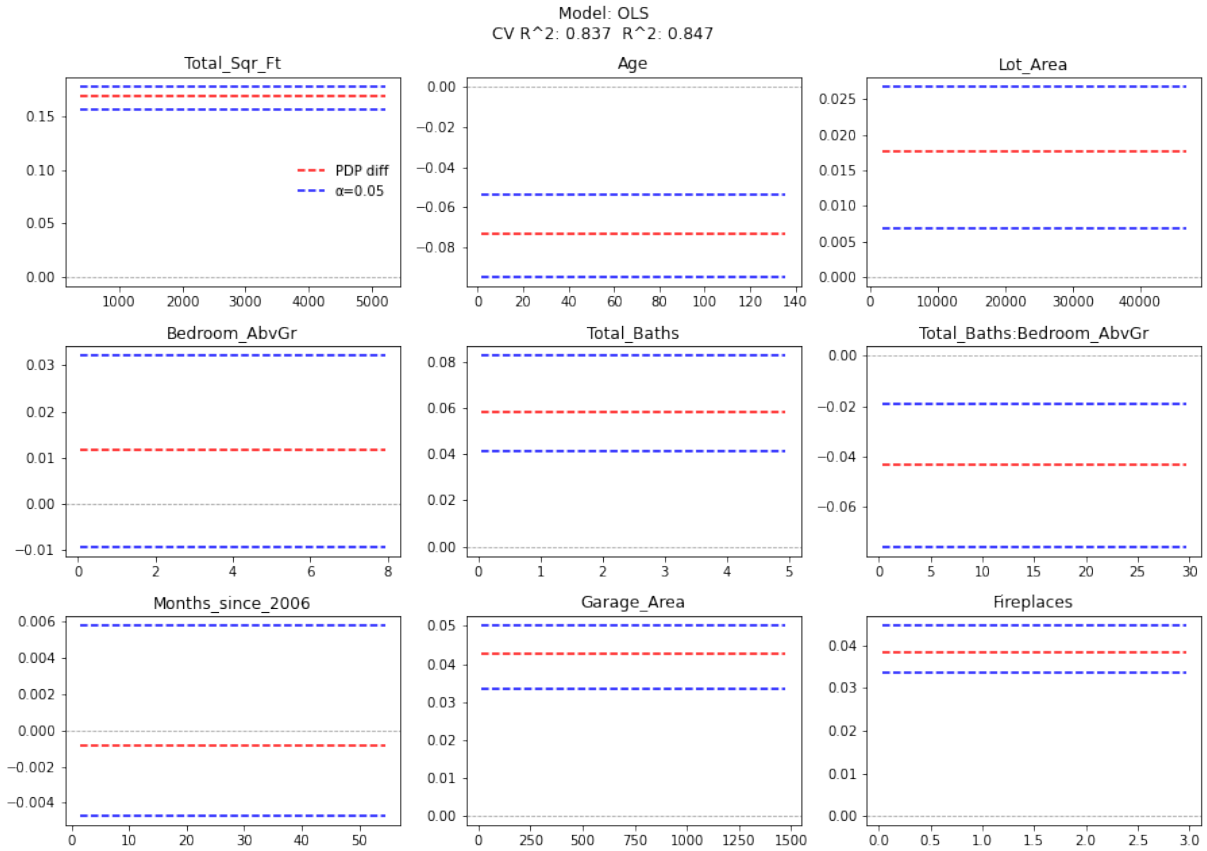
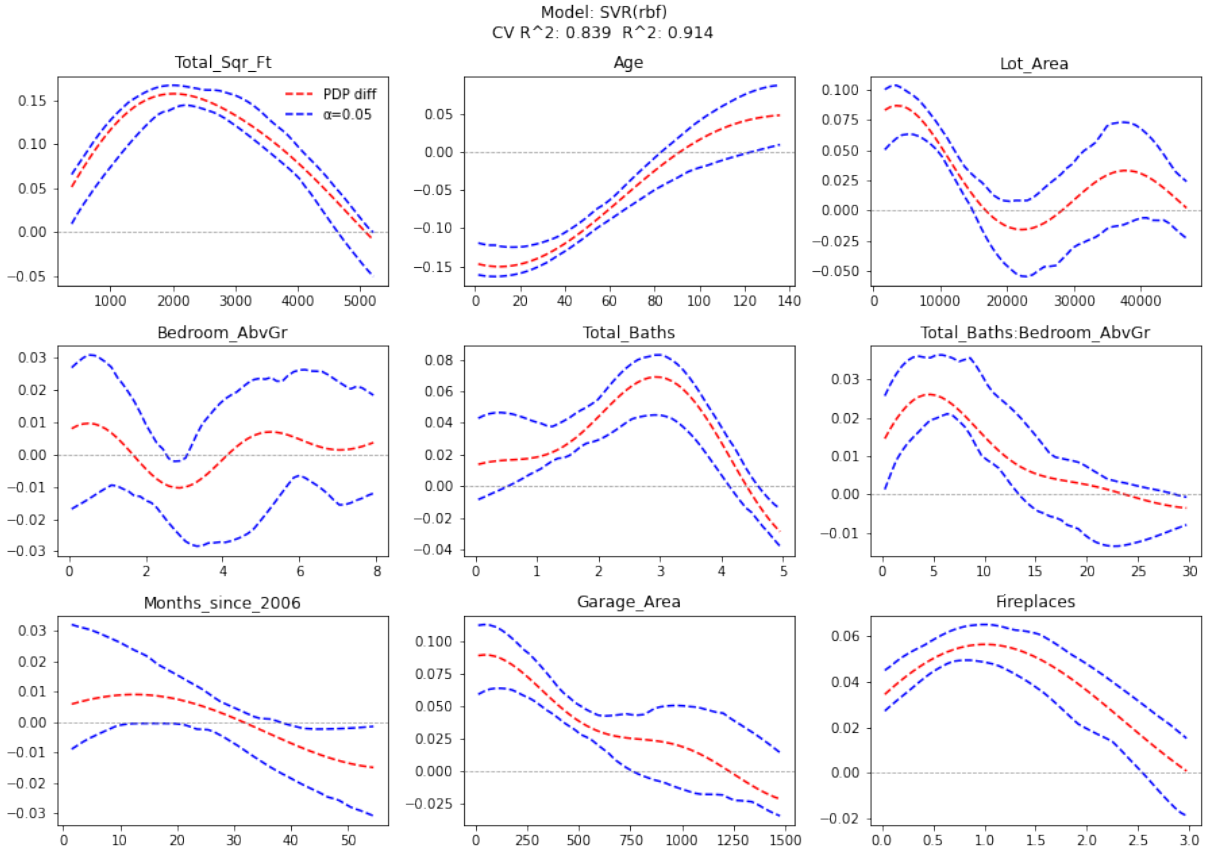


Figure 25: Differenced PDP for SVM



B.2 Additional SPDP and SFIPDP Results

Figure 26: SPDP results for SVM model

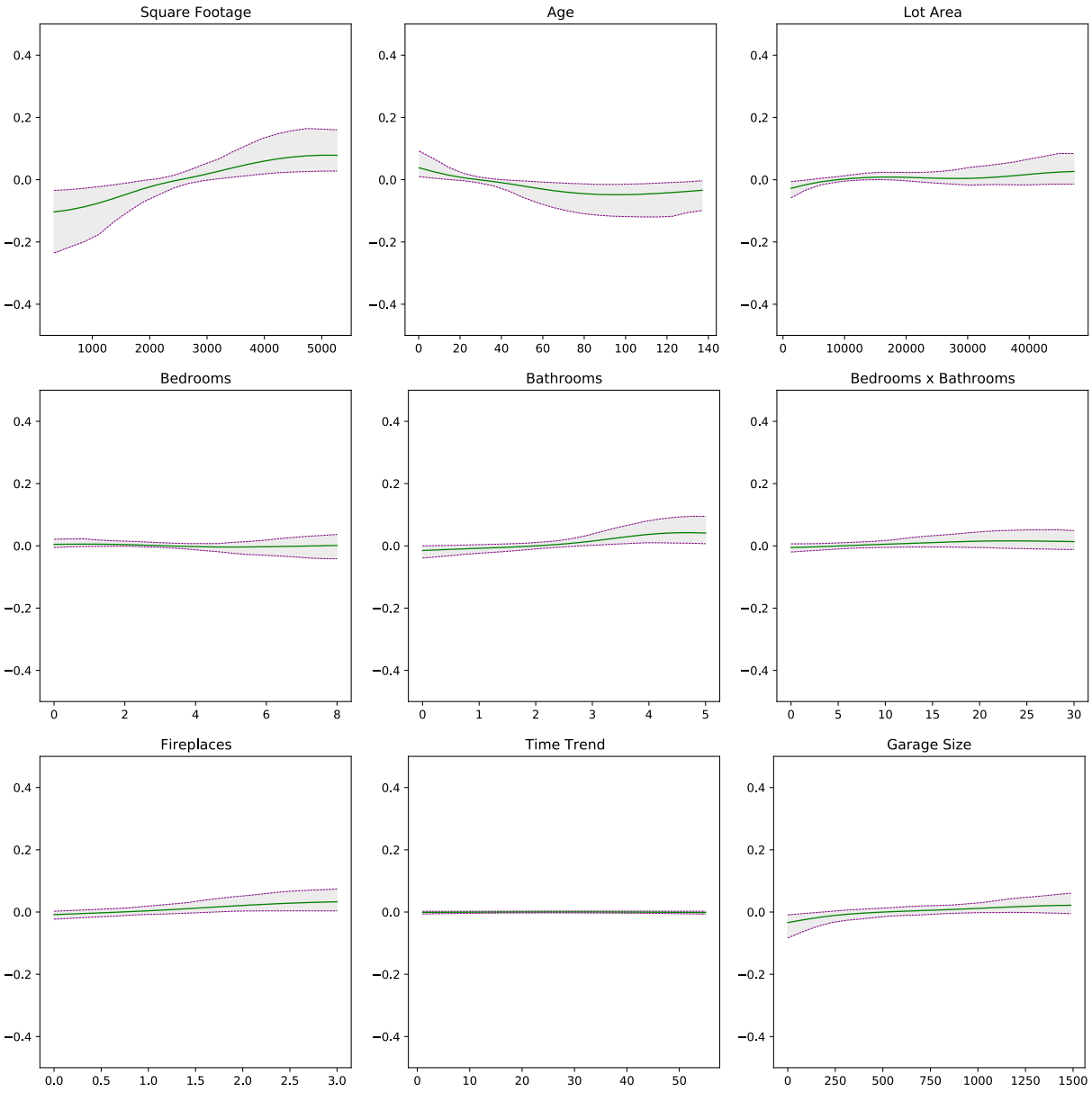


Figure 27: SFIPDP results for SVM model

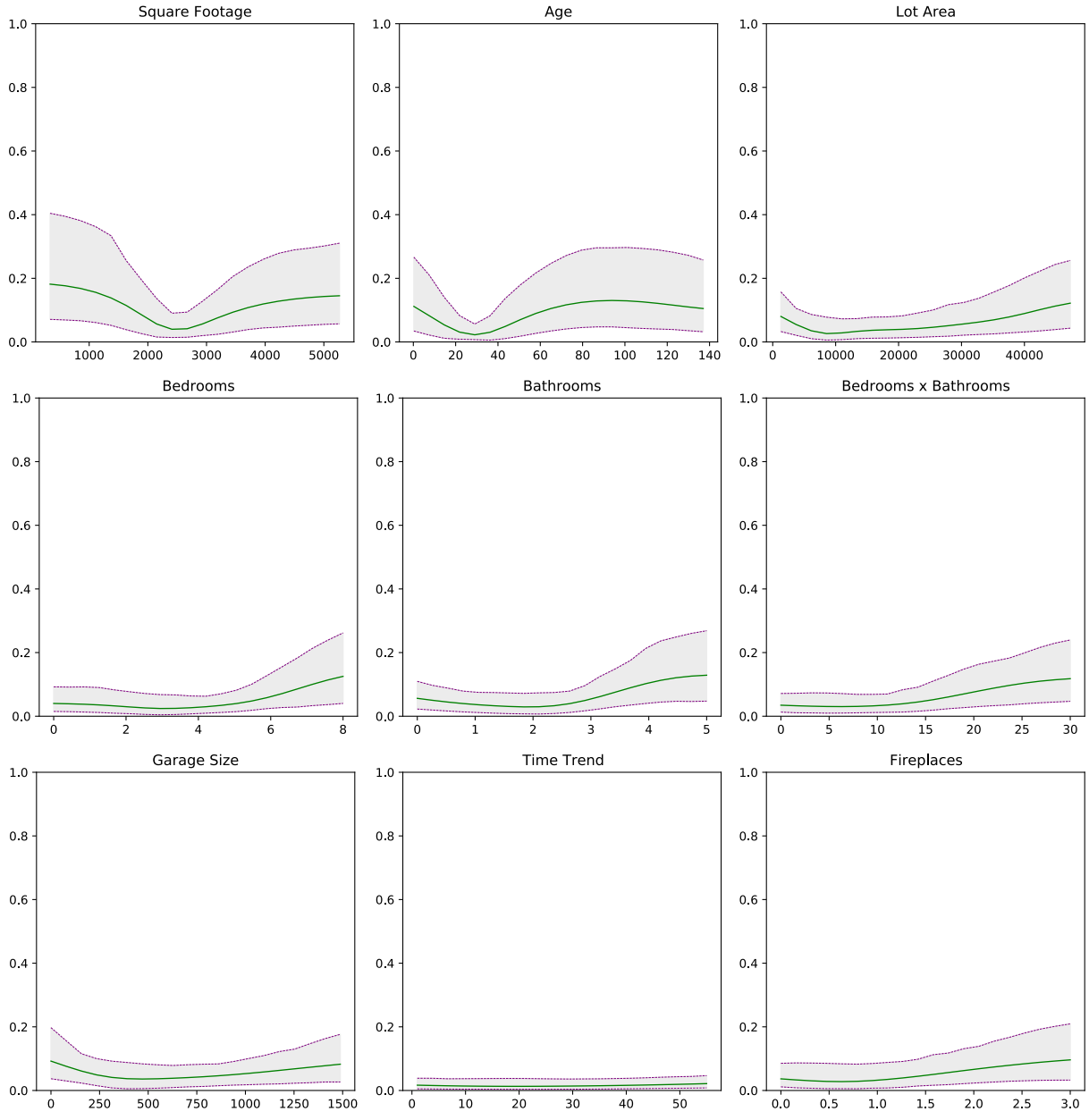


Figure 28: SPDP results for GBM model

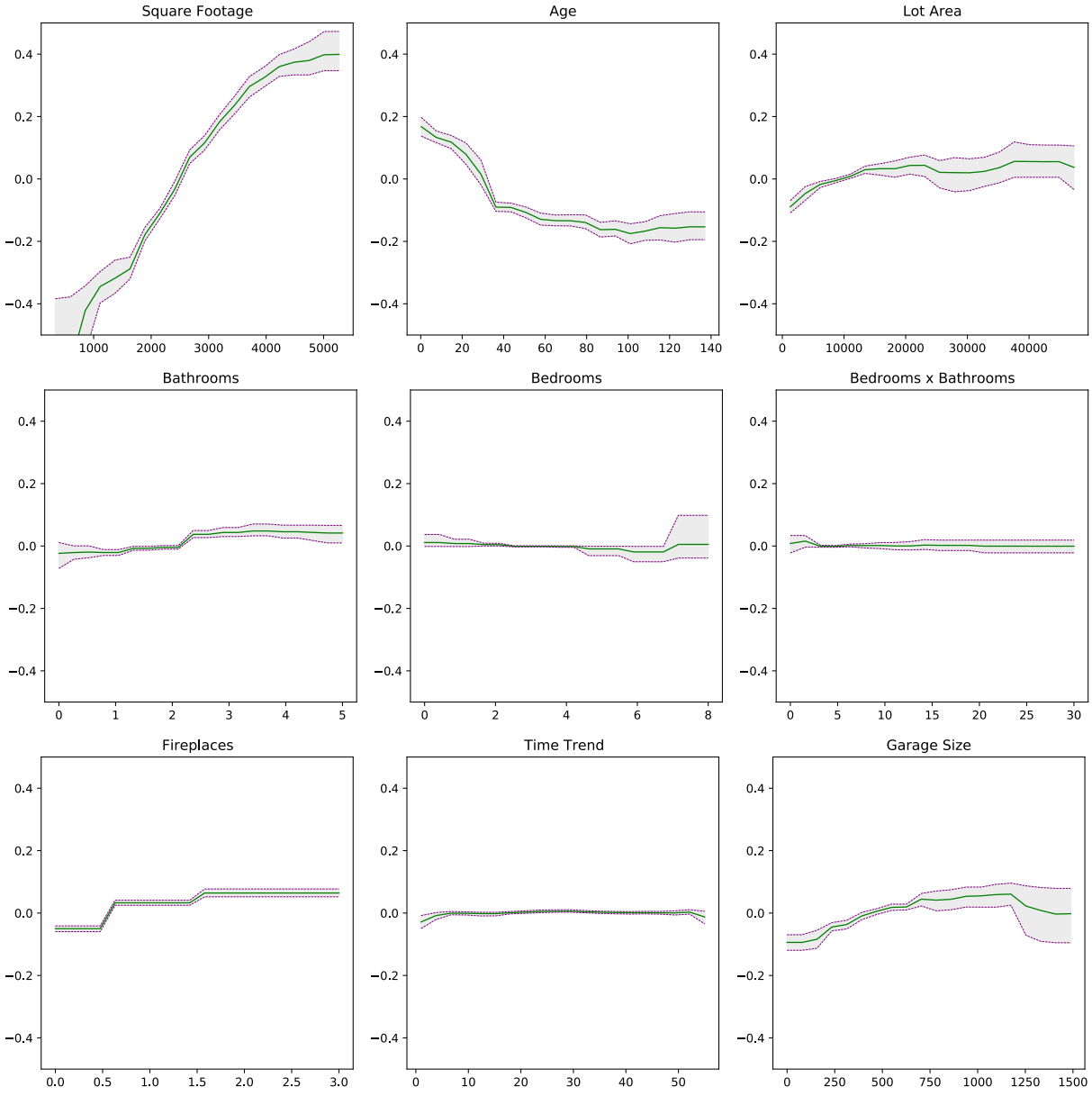


Figure 29: SFIPDP results for GBM model

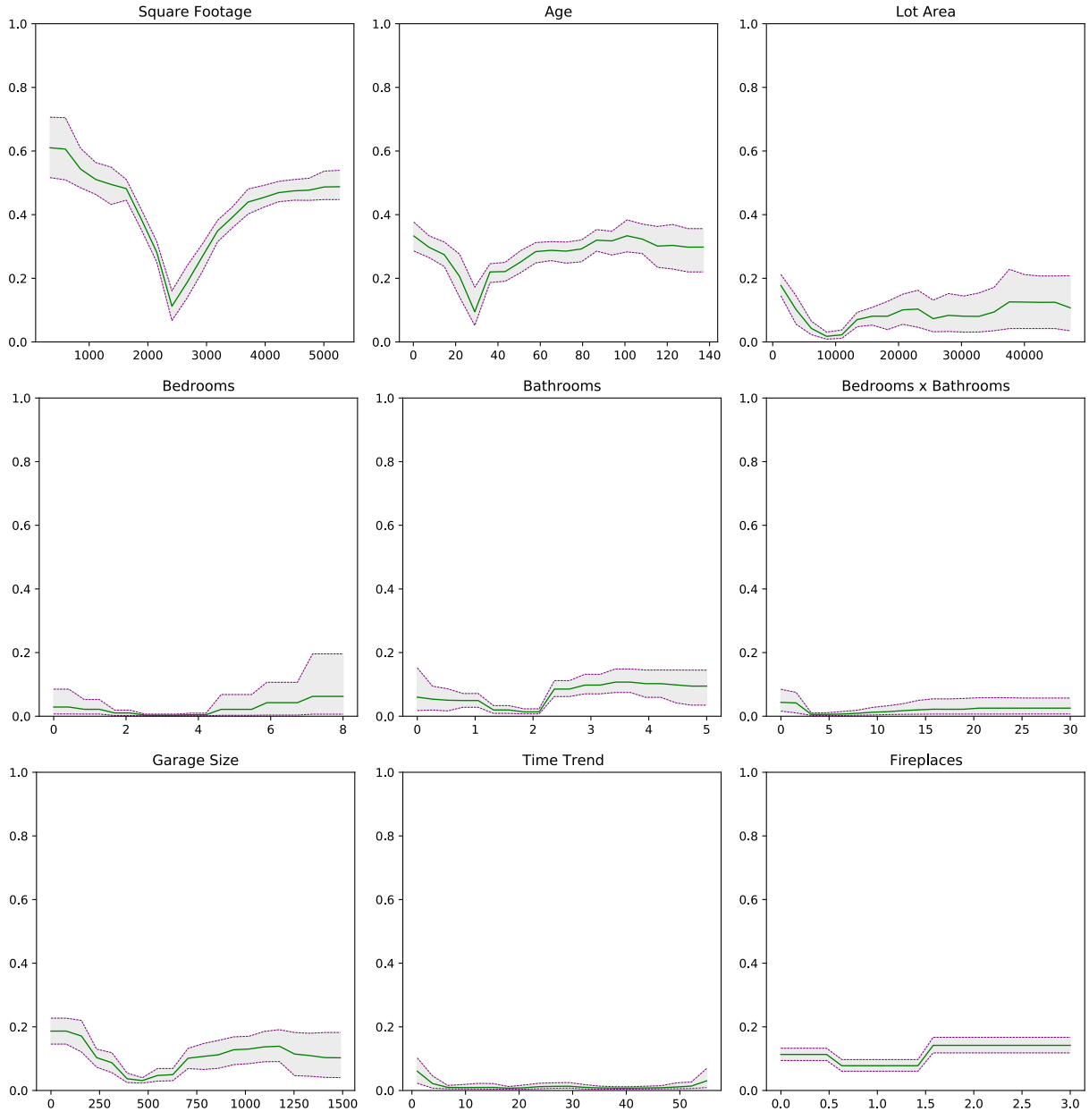


Figure 30: SPDP results for Linear model

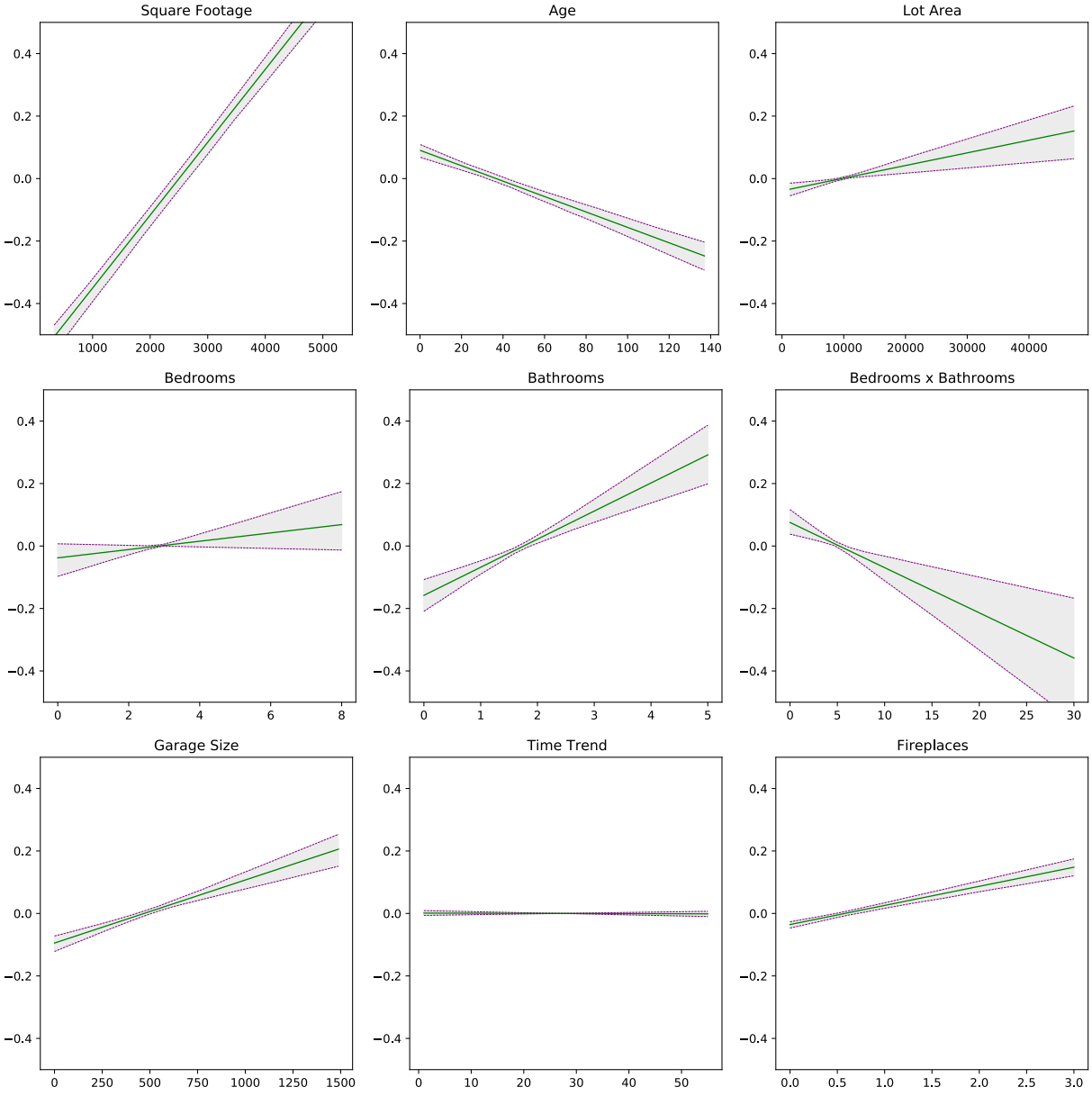
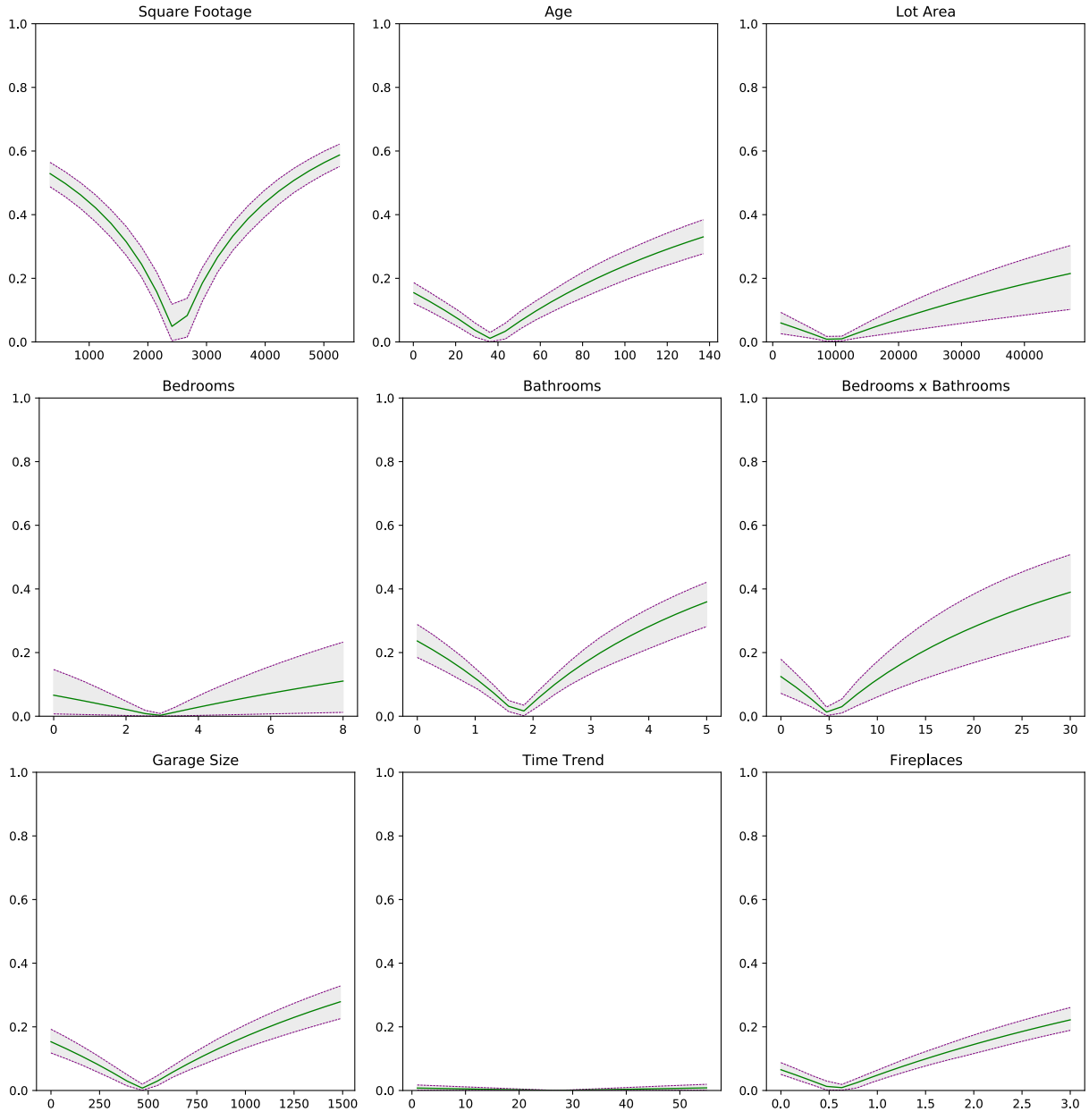


Figure 31: SFIPDP results for Linear model



C Claims about PDP, SPDP in linear models

The PDP of a function f is written $\nu_k(f(x_k = q)) \equiv \nu_k(q) = E_{x_{\neg k}}[f(q, x_{\neg k})|q] = \int_{x_{\neg k}} f(q, x_{\neg k})\mathbb{P}(x_{\neg k})dx_{\neg k}$.

Claim 1: $\nu_k(q) = q\beta_k + E(X_{\neg k})\beta_{\neg k}$ if f is linear and $\partial\nu_k/\partial q = \beta_k$.

Proof: If f is linear, it can be written as $XB = \sum_j x_j\beta_j$. Accordingly, we can write $f(q)$ as $q\beta_k + X_{\neg k}\beta_{\neg k}$ and

$$\begin{aligned}\nu_k(q) &= E[q\beta_k + X_{\neg k}\beta_{\neg k}|q] \\ &= q\beta_k + E[X_{\neg k}]\beta_{\neg k}\end{aligned}$$

From this it follows that $\partial\nu_k/\partial q = \beta_k$ ■

Write the de-meaned PDF as $\tilde{\nu}_k = E_{x_{\neg k}}[f(q, x_{\neg k}) - E_x[f(x)]|q]$

Claim 2: If f is linear, then $\partial\tilde{\nu}_k(q)/\partial q = \beta_k$

Proof: If f is linear, then we can write

$$\begin{aligned}\tilde{\nu}_k(q) &= q\beta_k + E_{x_{\neg k}}[x_{\neg k}\beta_{\neg k}] - E_{x_k}[x_k] - E_{x_{\neg k}}[x_{\neg k}]\beta_{\neg k} \\ &= (q - E_{x_k}[x_k])\beta_k.\end{aligned}$$

From this it follows $\partial\tilde{\nu}_k(q)/\partial q = \beta_k$ ■

Claim 3: If f is linear, then $\psi_k f(q) = (q - E_{x_k}[f(x_k)])\beta_k$

Proof: Recall that we write the shapley value as

$$\psi_k f(q) = \frac{1}{K}\sum_{s\subset S(x)\setminus k}\binom{K}{|s|}^{-1}\tilde{\nu}_k(q \cup s) - \tilde{\nu}_k(s)$$

From the proof of Claim 2, it follows that if f is linear,

$$\begin{aligned}\nu_k^e(q \cup s) - \nu_k^e(s) &= (q - E_{x_k}[x_k])\beta_k - (s - E[x_s])\beta_s - (s - E_{x_s}[x_s])\beta_s \\ &= (q - E_{x_k}[x_k])\beta_k \ \forall s \subset S(x)\end{aligned}$$

Further, denote $S(x, i)$ as the set of all subsets of covariates in x of size i, then we can write

$$\begin{aligned}\psi_k f(q) = \frac{1}{K}\sum_i^K (E_{s \subset S(x,i)\setminus k}[q] - E[x_k])\beta_k \\ = (q - E[x_k])\beta_k \blacksquare\end{aligned}$$