A Brief Overview of Machine Learning in Economics with an Application of Penalized Regression

Daniel Li

1 Introduction

In the last decades, the pure amount of information available to researchers has dramatically increased. Computing power has increased too. Innovations in the field of machine learning have enabled scientists and businesses alike to gain new insights into their data. For economics, much of the field's focus is on causal inference, aiming to explain the factors that influence GDP, wages, housing prices, etc. Machine learning is usually seen as a set of predictive tools, providing little explanatory power. Still, there are ways that machine learning can add to the world of economics, and prominent researchers have been advocating for machine learning's increased usage, especially in empirical research. In this paper, we will explore some of the ways machine learning can be applied to economics. I will also provide an application of machine learning to crime data.

2 Big Data, Machine Learning, and Economics

2.1 Big Data

The term "big data" has gotten popular in recent years. But that term in itself doesn't say much about the challenges and new avenues of analysis that are available to economists. What makes data "big"? Well, it's not just an issue of scale, and simply having many rows of data doesn't make it challenging to deal with. Mullainathan and Spiess (2017) describes big data as data that is too high dimensional for standard estimation methods, usually meaning data that has many more parameters than observations. This $n \ll p$ situation causes traditional statistical methods such as ordinary least squares to be not well behaved (Fan, Lv, & Qi, 2011). You will have issues such as collinearity and spurious correlations. Noise accumulation can make classification of features just as bad as random guessing. Methods have been developed to address high dimensionality, some of which will be discussed below. Big data arises from sources such as satellite imagery, network data, online posts and corporate financials (Mullainathan & Spiess, 2017). We are living in a world where new data is generated every second. Consider the millions of Instagram posts made each day. Each post contains an image which can be broken down at the pixel level. There are also captions and comments which can be analyzed word by word. One can easily imagine using location tagged Instagram posts to analyze sentiment and finding a relationship to housing prices.

2.2 Machine Learning

Machine learning focuses on developing algorithms to use on data sets for prediction, classification, and clustering (Athey, 2019). Machine learning algorithms are generally classified as supervised or unsupervised. Unsupervised methods are used mainly for grouping purposes. Athey (2019) sees unsupervised machine learning as a way to construct outcome variables or explanatory variables. Supervised machine learning, on the other hand, is used for prediction. You feed the algorithm a set of features and an outcome and aim to predict the outcomes based on the specification of features in a new set. The rest of this report will mainly address supervised machine learning.

2.3 Applications for Economics

Let's first differentiate between causation and prediction. Kleinberg, Ludwig, Mullainathan, and Obermeyer (2015) gives a framework for such differentiation. Consider an outcome variable Y which depends on a set of variables X_0 and X. Suppose a policy maker has to decide whether or not to implement policy X_0 to maximize a known payoff function, $F(X_0, Y)$. The derivative is

$$\frac{dF(X_0, Y)}{dX_0} = \frac{\partial F}{\partial X_0}(Y) + \frac{\partial F}{\partial Y}\frac{\partial Y}{\partial X_0}.$$

Kleinberg et al. (2015) states there are two unknowns, $\frac{\partial Y}{\partial X_0}$ and $\frac{\partial F}{\partial X_0}$. The first term, $\frac{\partial Y}{\partial X_0}$, requires causal inference to estimate. It's asking how much the policy effects the outcome. The second term, $\frac{\partial F}{\partial X_0}$, requires a prediction of Y to calculate. So, having a good prediction of Y will help you better understand the effect of the policy X_0 on the payoff. Kleinberg et al. (2015) argue that prediction problems are common in economics, with examples such as predicting which teachers with the greatest value add, predicting unemployment spell lengths to help workers optimize savings strategies, and predicting high risk youth who need interventions. As an example, their paper creates a model to predict the payoff to joint replacement surgery through predicting mortality rate. By looking at the high risk patients, they saw that the riskiest 1% of patients in their model accounted for 10% of futile surgeries. They believe that healthcare systems could use such models to avoid futile surgeries and better allocate money to beneficiaries that would have benefited from joint replacement (Kleinberg et al., 2015).

It's important to emphasize that usually predictions from a machine learning model, such as the one in the joint replacement example, doesn't tell us about the underlying relationship between variables. Mullainathan and Spiess (2017) writes that a large issue is the lack of standard errors on coefficients, meaning it's difficult to do inference. Also, even if one has standard errors, under certain circumstances, the errors are inconsistent. Another issue is that the way machine learning algorithms select models is unstable, resulting in changes to the coefficients used even if the model is changed slightly (Mullainathan & Spiess, 2017). As different models can provide the same level of predictive ability, there is no way to discern which covariates are actually informative. There are also concerns of whether models can be stable across time or environments due to the way machine learning algorithms perform exhaustive searches for the best specifications (Athey, 2019).

Still, prediction methods are a great way to inform policy decisions. Athey (2019) gives several examples of ways localities are already implementing machine learning into policy making; one interesting use case is predicting the likely hood of showing up to court in deciding whether to grant bail. A concern of real-world use of machine learning is models are subject to manipulation (Athey, 2019). For example, in a ranking system, if an malicious actor knows what factors are heavily weighted by the algorithm, they are able to adjust their behaviour to optimize their ranking.

3 Sparse Prediction Models

Now, we will turn to describing one of the machine learning methods used for prediction purposes. One of the issues with big data was that it has too many features compared to number of observations ($n < p$). A means to deal with high dimensionality is through sparse modeling, which assumes that many of the p components of the parameter vector are zero or negligibly small, leaving only important predictors with a nonzero component (Fan et al., 2011). We can also use sparse modeling when $n > p$ and we believe that only a few features are important to the model. A form of sparse modeling is called penalized least squares. To start, let's consider a linear model,

$$y = X\beta + \varepsilon,$$

where y is an n dimensional response vector and X is an $n \times p$ design matrix, and ε is an n dimensional noise vector. In ordinary least squares regression, we want to pick β such that we minimize the sum of the squared difference of the true value and predicted value,

$$\min_{\beta \in \mathbb{R}^p} ||y - X\beta||_2^2.$$

In penalized least squares, we add a penalty function, $p_\lambda(\cdot)$, to the minimization problem,

$$\min_{\beta \in \mathbb{R}^p} \left\{ ||y - X\beta||_2^2 + \sum_{j=1}^{p} p_\lambda(|\beta_j|) \right\}.$$

The use of the penalty function aims to simultaneously select important variables and estimate regression coefficients with sparse estimates Fan et al. (2011). A commonly used penalty function takes the form $p_\lambda(|\theta|) = \lambda(||\theta||_q)^{1/q}$, and if $q = 1$ or $q = 2$, we obtain LASSO or RIDGE estimators respectively (Athey & Imbens, 2019).

Why does adding a penalty function result in better predictions than OLS? Simply put, the penalty function introduces bias to the coefficients (Kleinberg et al., 2015). For example, LASSO usually brings many coefficients to zero. OLS gives nonbiased estimates of the coefficients. This may mean producing a coefficient with lots of variance. While the nonbiased property of OLS is good for causal inference, we are much more concerned with limiting noise in prediction.

4 Predicting Arsons with LASSO

This section will apply an LASSO regression to predicting arsons per 100k population given variables that describe a US county. This is of interest because it can inform policy makers on how to allocate resources in preventing crimes. Rather than focusing on actual predicted values, our goal is to compare how well LASSO performs versus standard OLS as we add more predictors.

4.1 The Data Set

We will be using the Communities and Crime Unnormalized Data Set from UC Irvine's Machine Learning Repository.[1] This data set was put together by combining data from the US Department of Commerce and the US Department of Justice; full attributions can be found on the UC Irvine website. The data set with column names in CSV format along with code for the sections below will be available on my GitHub.[2]

The data set contains 2215 observations and 147 columns. 125 of the variables are predictive such as percentage of population that is Asian, percentage of population that is 12-21 in age, and police officers per 100k population. There are 18 potential goal attributes such as total number of crimes, murders per 100k population, and arsons per 100k population. A full list of attributes will not be given for space reasons but can be found in either link in the footnotes.

4.2 Application

To compare LASSO and OLS, we created models with increasing number of features. The features were randomly select from the full "pool" of features available for prediction. Models with more features are built on top of the previous models. The code for this method can be seen in Listing 1. The models were trained on 60% of the full data, leaving 40% for test purposes. We calculated the root mean squared error of the respective model predictions on the test data. This produced Table 1. Figure 1 gives a visual interpretation of the table.

[1]https://archive.ics.uci.edu/ml/datasets/Communities+and+Crime+Unnormalized
[2]https://github.com/dx-li

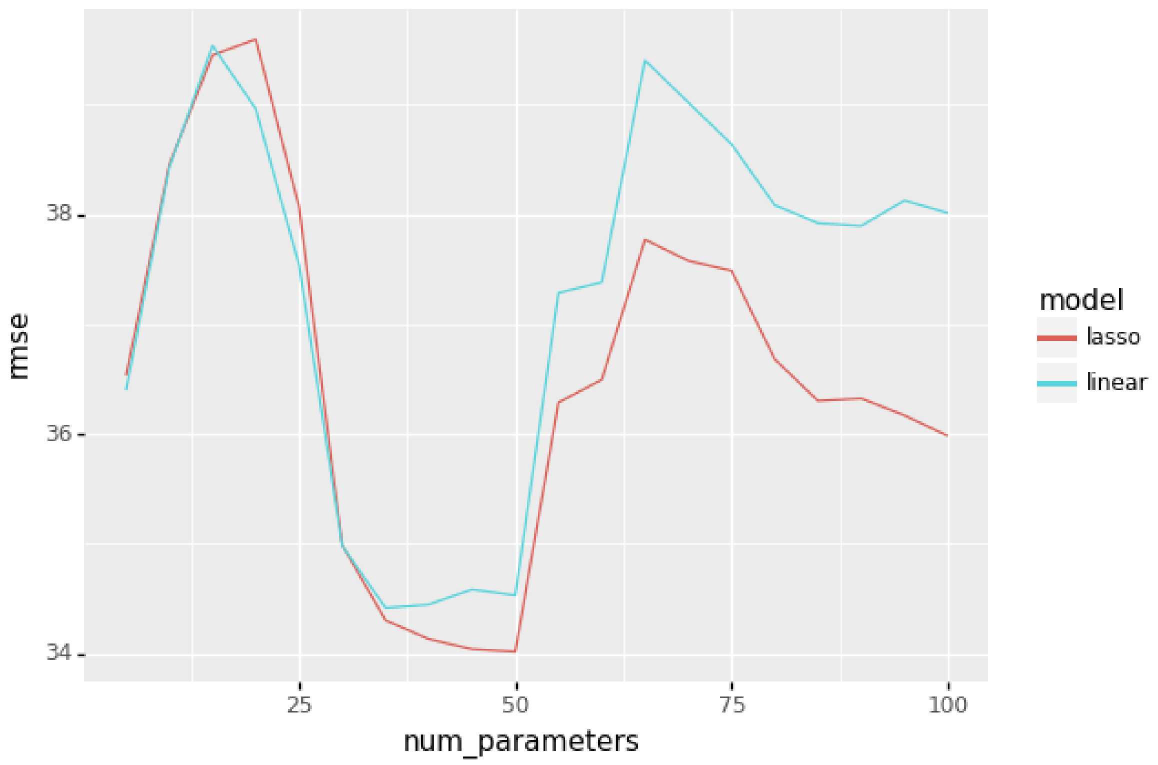

Figure 1: Graph of RMSE as we increase parameters

We can observe that LASSO didn't necessarily perform better than OLS. For lower number of parameters, OLS and LASSO were approximately even, with LASSO even performing worse at times. It's once we have around 30 parameters that LASSO is consistently predicting better. Even then, RMSE is minimized at around 50 parameters, after which, we are likely overfitting to the training data.

Through a similar procedure as above, I also created a comparison of LASSO and OLS when there are second order interaction terms. This was done to partially simulate high dimensionality and resulted in Figure 2. The full table of RMSE can be found in Table 2.

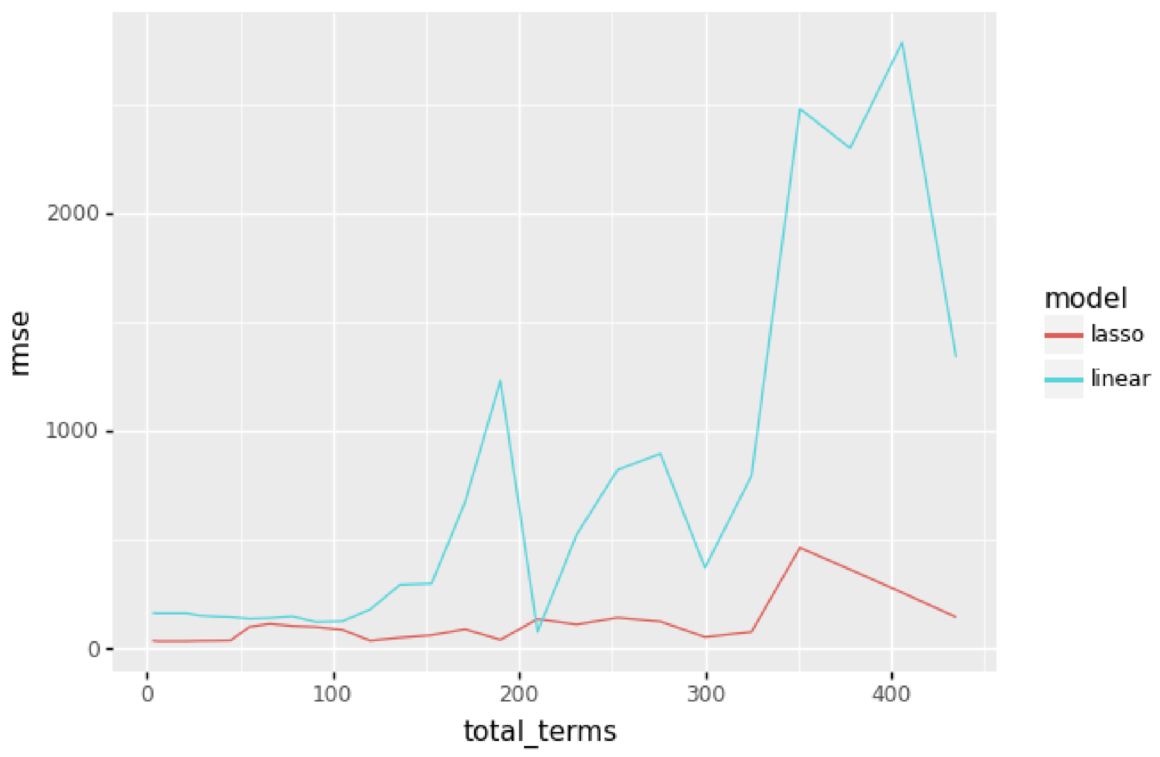

Figure 2: Graph of RMSE as we have more terms from interactions

As expected, LASSO does much better in this case. The interactions terms likely created high levels of multicollinearity, so the LASSO penalizer essentially zeroed out many coefficients, enabling the RMSE to stay relatively low compared to the linear model.

Overall, this section shows that we have to carefully choose our models. We also can't increase predictors in the hopes of a better fit–that may be good in-sample, but out-of-sample predictions, which are of interest, suffer from overfitting. Lastly, Figure 2 in a way shows how LASSO can minimize variance in return for being a biased.

5 Machine Learning for Causal Inference?

So far, we have been discussing prediction and how machine learning can help economists predict variables of interest. But a major focus of an economist's work is on finding causal relationships. That doesn't mean machine learning is unhelpful. There has been a growing body of work developing around machine learning for causal inference in recent years. One area that causal machine learning can address is heterogeneous treatment effects (Athey, 2019). Heterogeneous treatment effects analysis asks how a treatment or policy or drug's efficacy depends on an individuals' or subgroup's characteristics. Athey, Tibshirani, and Wager (2016) proposes Generalized Random Forests as a machine learning model for causal inference of treatment effects. Their model differs from Random Forests in that the algorithms splits the data to maximize heterogeneity and use, what they term, "honest trees", which can reduce bias and give valid confidence intervals (Athey et al., 2016). Athey and Wager (2019) gives an application of Generalized Random Forests to assessing education interventions. Other work has been done on effects estimation too. Syrgkanis et al. (2019) provides a method of estimating heterogenous treatment effects with instruments by reducing effects estimation to a square loss problem. Overall, there's no doubt that machine learning will help economists solve causal problems in the future.

6 Conclusion

The literature for machine learning in economics is thick, and this report has barely scraped the surface. I see a possibility in testing more models to the crime data set and finding the optimal one for prediction. A topic of interest is applying new models that have been developed, such as Generalized Random Forests, to data from old studies of treatment effects. Then, one would see if there are consistent results. Machine learning has opened many new doors for economists, and it will take creativity to use all of them.

References

Athey, S. (2019, May). The impact of machine learning on economics. *The Economics of Artificial Intelligence*, 507-552. Retrieved from `https://www.universitypressscholarship.com/view/10.7208/chicago/9780226613475.001.0001/upso-9780226613338-chapter-021` DOI: 10.7208/chicago/9780226613475.003.0021

Athey, S., & Imbens, G. (2019). Machine learning methods economists should know about.

Athey, S., Tibshirani, J., & Wager, S. (2016). Generalized random forests. Retrieved from `https://arxiv.org/abs/1610.01271` DOI: 10.48550/ARXIV.1610.01271

Athey, S., & Wager, S. (2019). Estimating treatment effects with causal forests: An application.

Fan, J., Lv, J., & Qi, L. (2011). Sparse high-dimensional models in economics. *Annual Review of Economics*, *3*(1), 291-317. Retrieved from https://doi.org/10.1146/annurev-economics-061109-080451 DOI: 10.1146/annurev-economics-061109-080451

Kleinberg, J., Ludwig, J., Mullainathan, S., & Obermeyer, Z. (2015, May). Prediction policy problems. *American Economic Review*, *105*(5), 491-495. Retrieved from https://www.aeaweb.org/articles?id=10.1257/aer.p20151023 DOI: 10.1257/aer.p20151023

Mullainathan, S., & Spiess, J. (2017, May). Machine learning: An applied econometric approach. *Journal of Economic Perspectives*, *31*(2), 87-106. Retrieved from https://www.aeaweb.org/articles?id=10.1257/jep.31.2.87 DOI: 10.1257/jep.31.2.87

Syrgkanis, V., Lei, V., Oprescu, M., Hei, M., Battocchi, K., & Lewis, G. (2019). Machine learning estimation of heterogeneous treatment effects with instruments. Retrieved from https://arxiv.org/abs/1905.10176 DOI: 10.48550/ARXIV.1905.10176

Appendices

```python
random.seed(34)
parameterpool = set(df.columns[5:-18])  # create parameter pool
num_parameters = np.arange(5, 101, 5)  # add five predictors each time
lasso_rmse = []
linear_rmse = []
parameters = []
for n in num_parameters:
    parameters = parameters + random.sample(parameterpool, n-len(parameters))
    parameterpool.difference_update(parameters)
    model = Lasso(random_state=3)
    model.fit(train[parameters], train['arsonsPerPop'])
    lasso_rmse.append(np.sqrt(mean_squared_error(
        test['arsonsPerPop'], model.predict(test[parameters]))))
    linmodel = LinearRegression()
    linmodel.fit(train[parameters], train['arsonsPerPop'])
    linear_rmse.append(np.sqrt(mean_squared_error(
        test['arsonsPerPop'], linmodel.predict(test[parameters]))))
```

Listing 1: Code Snippet for generating RMSE

num_parameters	lasso_rmse	linear_rmse
5	36.53	36.40
10	38.45	38.42
15	39.45	39.54
20	39.59	38.96
25	38.07	37.54
30	34.99	34.98
35	34.31	34.42
40	34.14	34.45
45	34.04	34.59
50	34.02	34.53
55	36.29	37.28
60	36.49	37.38
65	37.77	39.40
70	37.58	39.02
75	37.49	38.64
80	36.68	38.08
85	36.30	37.92
90	36.32	37.89
95	36.17	38.13
100	35.98	38.01

Table 1: RMSE as number of features increased

	num_parameters	lasso_rmse	linear_rmse	total_terms
0	2	37.66	164.04	3.00
1	3	34.68	162.42	6.00
2	4	34.71	162.77	10.00
3	5	34.83	162.63	15.00
4	6	34.82	162.58	21.00
5	7	36.49	152.11	28.00

6	8	36.92	147.75	36.00
7	9	38.25	145.21	45.00
8	10	99.07	137.03	55.00
9	11	113.93	140.42	66.00
10	12	103.27	148.47	78.00
11	13	98.57	121.88	91.00
12	14	86.46	125.41	105.00
13	15	36.92	179.95	120.00
14	16	50.45	292.86	136.00
15	17	61.94	298.43	153.00
16	18	88.92	672.44	171.00
17	19	40.72	1,229.65	190.00
18	20	135.53	77.06	210.00
19	21	110.65	523.69	231.00
20	22	141.73	820.68	253.00
21	23	124.14	894.08	276.00
22	24	53.09	371.45	300.00
23	25	76.49	793.11	325.00
24	26	462.97	2,477.73	351.00
25	27	362.85	2,296.80	378.00
26	28	257.14	2,782.31	406.00
27	29	144.65	1,336.84	435.00

Table 2: RMSE when we have interactions terms