Machine Learning Approach for Targeting the Poverty in Peru

RACHID BEN MAATOUG *rachid.benmaatoug@uzh.ch*
NATAELL CORNU *nataell.cornu@uzh.ch*
MINGMIN FENG *mingmin.feng@uzh.ch*
JINGYAN YANG *jingyan.yang@uzh.ch*
YU PAN *yu.pan@uzh.ch*

This report is a replication and extension of Hanna and Olken's study about predicting per-capita consumption by using information of observable assets for per household in Peru. We add new assets features which are not used by Hanna and Olken and also create more than four thousands interactions specifically for Lasso. We use other machine learning techniques such as K-nearest Neighbors, Support Vector Machine, Light Gradient Boosting Machine and Neural Network to predict the poverty status of households on the testing sample. We find that Light Gradient Boosting Machine performs better than other models using the metric of MSE. In order to find out whether the new predictions can better target the poverty, we also replicate the welfare analysis with the new proxy-means test and compare it to the original one. Among all these methods, predictions from Light GBM help Peru achieves a highest maximum of social utility.

Word Count: 3161

1 INTRODUCTION

Fighting poverty is a major priority for many developing countries and international organizations. One method for the government to obtain a signal about households' economic status is termed a proxy-mean test. Generally, the government uses information of easily observable and verifiable assets to predict income, per-capita consumption or other measurements of poverty of the household. Hanna and Olken, using the data in Peru from Peruvian Encuesta Nacional de Hogares (ENAHO), estimate per-captia consumption for each household with a welfare analysis by using the predicted outcomes. In the data acquisition process, we add new assets features which are not used by Hanna and Olken and also we create more than four thousands interactions as what Kozbur

(2020) does. Thereafter, we use multiple machine learning techniques and create new outcomes of proxy-means on the testing sample.

In the second section, we will describe the randomized controlled trial (RCT) study conducted in Peru that used by Hanna and Olken and also describe our data cleaning process. In the third section, we use ordinary least squares (OLS) regression to replicate their estimation for the testing sample. Then we explore different machine learning methods such as LASSO, K-nearest Neighbors (KNN), Support Vector Regression (SVR), Light Gradient Boosting Machine(Light GBM), Neural Network(NN) to create a more precise new proxy-means on the testing sample which is the same as the original study used. We compare the mean-square error[1] (MSE) on the hold-out sample among different methods. In the forth section, we replicate the welfare analysis (Figure 5 in the Hanna and Olken reference) with our new predictions.

2 PROJECT OVERVIEW

2.1 Program Background

The original Peru study focused on the Juntos program, which provides conditional cash transfers to mothers designed to subsidize child health and education. Beneficiary households receive a monthly transfer of 100 soles (approximately $30). This program targets roughly one-third of the population. They obtain household-level data from the Peruvian National Household Survey(ENAHO) for the years 2010 and 2011. The whole sample is 46305 households and the surveys contain the complete set of asset variables used in targeting with a measure of actual per-capita consumption (which is the outcome variable we aim to predict). They randomly divide the whole dataset into roughly equally sized "training" (23,153 observations) and "test" sets (23,152 observations). They predict monthly per-capita consumption for each household in the test set using the coefficients of the OLS regression from the training dataset.

However, this prediction will lead to predictions errors: exclusion errors of excluding those who should have been subsidized (households with actual per-capita consumption below the poverty line)

[1] $MSE = \frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$

and inclusion errors of including those who should not have been subsidized (households with actual per-capita consumption above the poverty line). The government then will face the tradeoffs between inclusion error and exclusion error. Better balancing the tradeoffs requires the government evaluating the total social welfare from these different cutoff decision holding the total transfer budget constant. This paper uses CRRA utility function to find the best cutoff at which the utility will be maximized.

2.2 Data Processing

Our task is to predict the monthly (log) per capita consumption (continuous target variable) based on a set of qualitative variables, where the original paper runs an OLS regression by using them as showed in Table 2 and Table 3. These variables describe fuel type, water source, drainage source, wall type, roof type, floor type, electricity, telephone, head's education level, maximum education level, insurance coverage, household crowding, number of luxury. All the variables have been transformed into 72 dummies by using one-hot encoding.

In addition, we add 5 more variables representing whether a household owns a specific asset: internet, TV cable, computer, refrigerator, washer. In the original paper, Hanna and Olken keep $luxury$ only to represent the total number of the above mentioned assets, which we think this might lose some information. Therefore, we keep the $luxury$ variable and these five durable asset variables.

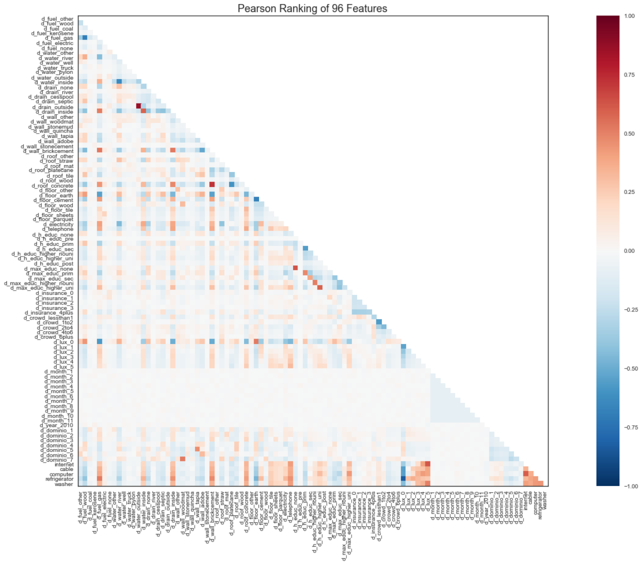
In order to capture the time trend variation (though there are only two years in our sample), we add one $year$ dummy about whether the $year$ is 2010 (equals to 1 if the year is 2010; otherwise, the dummy is 0) in our dataset. Since our prediction is monthly per-capita consumption for a household, it is reasonable to believe that people's consumption behavior might be different among different months. Therefore, we use one-hot encoding to transform the $month$ into 11 dummies [2]. Thus, we can capture this effect. As Figure 1 shows, the $month$ and $year$ dummies are almost uncorrelated with other explanatory variables since whether the household owning one kind of assets does not change over time.

Moreover, we find the original paper does not include the variables $dominio$. $dominio$ in some

[2] $Note$: The dummy whether month is equal to December is omitted due to the multicollinearity

sense represents the location of the household and it contains seven different types [3]. Taking the different development level among different region into consideration, *dominio* could be an important feature for our prediction. We also check that the mean of monthly per-capita consumption varies a lot among different region as showed in Table 4. We then use one-hot encoding to add the 7 corresponding dummies.

FIGURE 1. Correlation Matrix



Looking into the data, we find that there are 927 households with missing value, 479 observations from training dataset and 448 observations from test dataset which represented by the blank rows in the Figure 7. The frequently used method for imputing missing values are using the statistics (constant, mean, median or most frequent) of each features. As the sample size with missing value only accounts for 2% of the total sample which will not result in a great loss of information, we drop these records for the training dataset. For the test dataset, we impute the missing data by the median since all of our predictors are dummy variables and it makes no sense to impute the missing values by mean.

[3] *Note*: The seven types are central coast, northern coast, southern coast, metropolitan Lima, rain forest, central highlands, northern highland, southern highland, where metropolitan Lima is omitted.

3 MODEL IMPLEMENTATION

In this section we explore how well machine learning techniques perform in our setting. Since we are dealing with the regression problem, the metric we use for all the machine learning algorithms is MSE. And for each algorithm, we test both $percapita\ consumption$ and $lnpercapita\ consumption$.

3.1 Baseline: OLS Regression

First we replicate the Ordinary Least Squares Regression (OLS) from the original paper as our baseline. The OLS regression makes 22704 predictions with MSE of 89689.4786. For $lnpercapita\ consumption$, MSE is 0.1908. Though OLS regression only depicts the linear relationship between the input and the output, the authors already binned/discretized continuous variables, such as head's education, which introduced nonlinearity into the linear regression model. Yet, one shortage that can be seen is that STATA ignores any observations of which dependent or independent variables are missing both in training and test datasets, and cannot make predictions for every observation in the test dataset.

3.2 LASSO

The "least absolute shrinkage and selection operator"(LASSO) was proposed by Tibshirani (1996), minimizing the residual sum of squares under a constraint on the $l^1 - norm$ of the coefficient vector. Thus, the optimization problem was solved by the LASSO estimator for some time $t > 0$.

$$\min_{\beta_1,\ldots,\beta_m} \frac{1}{2}\sum_{i=1}^{n}(y_i - \sum_{j=1}^{m} x_{ij}\beta_j)^2$$

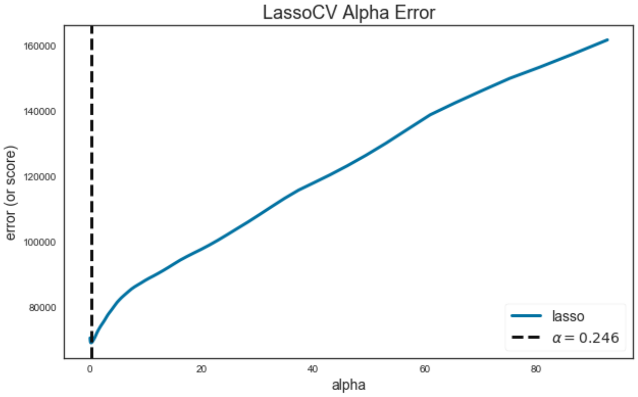
$$s.t. \sum_{j=1}^{m} |\beta_j| \leq t$$

For smaller values of t, the LASSO shirks the estimated coefficient vector towards the origin, typically setting some of the coefficient equal to zero. As a result, the characteristics of ridge regression (regularization) and subset selection (variable selection) were combined by LASSO, performing to be a useful tool for regression analysis.

Benefited from the advantages, we would like to perform the regularized linear regression, LASSO, based on a dataset with more additional features. The common method of creating new features is feature interaction, including sum of features, difference between features, product of features and quotient of features. Since all of our inputs are boolean values, taking the sum, difference or quotient does not make sense. Therefore, we follow what ATBFMS (Kozbur 2020) does, making interactions between each of two different variables in our dataset, and generate 4560 interaction terms in the end. Therefore, compared with 96 features at the beginning, we now have 4656 features in total.

After applying the proper alpha, MSE for $percapita\ consumption$ from Lasso is 80288.0317, and MSE for $lnpercapita\ consumption$ is 0.1805. It is clear that Lasso perform better than our baseline model (OLS) in this dataset.

FIGURE 2. LassoCV Alpha Error with 4656 features



3.3 K-Nearest Neighbors(KNN)

Cover and Hart (2006) proposed K-nearest neighbors (KNN), a non-parametric method computing the distances or similarities between all training points and then assign a value to the new point based on how closely it resembles the point in the training set. There are different ways of measuring the distance of the nearest neighbors. Here we use the Hamming Distance since all of our inputs are categorical variables. Also for this reason, there is no need to normalize our data.

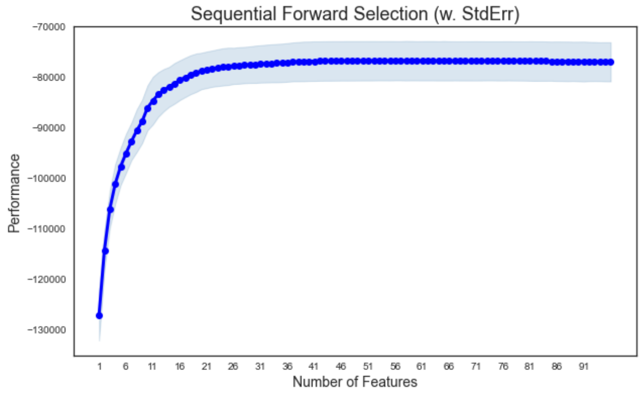
$$D_H = \sum_{i=1}^{k} |x_i - y_i|, \quad where\ both\ x_i\ and\ y_i\ are\ inputs.$$

In the KNN algorithm, the other important parameter needed to be considered is the number of neighbors K. However, there is often a trade-off between a small K and a large K. In general, a small K may cause overfitting problem as it adds the overall noise to the model, while a large K may make imprecise prediction since the dissimilar neighbor is taken into account as well. To handle this problem, we use cross validation to find the optimal value of K, which is 37. The MSE of $percapitaconsumption$ for the training dataset is 4480.9586 while the MSE for the testing dataset is 91224.3583, indicating the existence of overfitting problem.

As the number of neighbors we choose is not too small, this should not be the driver of the overfitting problem. Hence, it may be necessary to perform dimensionality reduction to more efficiently implement KNN. We use the forward model selection to find out a set of the most important explanatory variables as shown in Figure 3, coming up with 65 variables in the end.

However, the dimensionality reduction does not result in an obvious improvement on the prediction performance with the MSE of 90780.4793 for $percapita\ consumption$ and the MSE of 0.2099 for $lnpercapita\ consumption$. We think this is mostly because that all the features are of categorical type, in this sense it would be highly difficult to accurately define and measure the distance in the KNN algorithm.

FIGURE 3. KNN Sequential Forward Selection



3.4 Support Vector Machine (SVM Regression)

The Support Vector Machine (SVM) is a novel learning machine introduced first by Cortes and Vapnik (1995). It performs regression by transforming the data into a higher dimensional space and then finding the fitting hyperplane in the space, where the hyperplane can be represented by $w^T x + b = 0$. The optimaL hyperplane is found by solving a Lagrangian dual quadratic programming problem.

$$\begin{aligned} &min \quad \frac{1}{2}\|w\|^2, \quad where\|w\| = \sqrt{w_1^2 + ... + w_n^2} \\ &s.t. \quad y_i(w^T x_i + b), \quad fori = 1, ..., l \leq 1 \end{aligned}$$

One clear merit of SVM is to handle high-dimensional model the non-linear relationship by the use of "kernel trick". In our case, we use the radial basis (RBF) function as our kernel function which helps to map the data to an infinite-dimensional space. Nevertheless, this also causes high computational expense, especially when training a huge dataset.

3.5 Light Gradient Boosting Machine (Light GBM)

Ke et al. (2017) introduced light Gradient Boosting Machine (Light GBM) based on Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). Light GBM combines multiple weak learners and thus obtains better predictive performance, providing the gradient boosting framework based on decision tree algorithms.

Compared with the level-wise algorithm for the same number of splits, Light GBM splits one leaf at a time with the largest splitting gain (which yields the largest decrease in loss) from all the current leaves and then the process repeats, in this approach obtaining better accuracy with faster training speed.

3.6 Neural Network: Multi-Layer Perception (MLP)

Multi-layer Perception (MLP) introduced by (Mhaskar and Micchelli 1992) , as the most basic and a feed-forward neural network, learns to re-express and re-present the training data and relates it to the output variable passing through multiple layers. Between layers the activation function will be used transferring the weighted sum of inputs from one layer to the next. During this process, the use of non-linear activation functions combines the inputs in a more complex way and hence provides a better-performed model.

Though Neural Network is computationally expensive to train and has many parameters requires to be tuned, it performs well when modeling with the nonlinear data with many features and a large set of inputs.

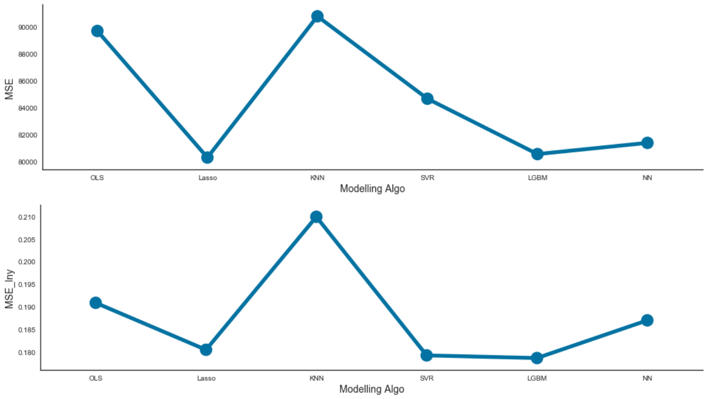
3.7 Comparison among Models

In this section we report the results of the six machine learning algorithms applied to the Peru dataset which are shown in Table 4. As it can be seen, the ensemble learning method, Light GBM outperforms other models with MSE of 80546.3523 for *percapita consumption* and 0.1786 for *lncapita consumption*. We think this is because of its compatibility with large datasets and the leaf-wise splitting approach which can produce more complex trees.

TABLE 1. Summary Statistics: per-capita consumption by different dominio

Model	MSE.y	MSE.lny
OLS:	89689.4794	0.1909
Lasso:	80288.0317	0.1805
KNN:	90780.4793	0.2099
SVR:	84657.0160	0.1792
Light GBM:	80546.3523	0.1786
NN:	81377.0992	0.1870

FIGURE 4. Comparison of MSE: percapita and Inpercapita



4 SOCIAL WELFARE IMPROVEMENT

As we described above, the government has to consider the tradeoffs between exclusion errors and inclusion errors, thus needs to evaluate the total social welfare from these different cutoff decision holding the total transfer budget constant. Our report follows the widely-used measurement that Hanna and Olken uses to evaluate the total social welfare by using a CRRA-utility function:

$$U = \frac{\sum (y_i + b_1)^{1-\rho}}{1-\rho}$$

where y_i is household i's pre-tax per-capita income, b_i is the per capita benefits assigned to the household i, and ρ is a coefficient of relative risk-aversion which is assigned as 3 in their study. Here a higher ρ means a higher weights on transfers received by the very poor. The socially optimal program in Peru targets about 18 percent of the population with inclusion error of 6.4 percent and exclusion error of 52.4 percent. Here we predict the $lnpercapita\ consumption$ by training the X training dataset and the actual $lnpercapita$ training dataset using the aforementioned models rather than simply transforming the predicted $percapita\ consumption$ into the logarithm form.

The results of the corresponding of machine learning approaches are as shown below:

- LASSO: The CRRA utility is -0.2362. The social utility is maximized when targeting about 21 percent of the population with inclusion error of 8.0 percent and exclusion error of 45.5 percent.
- KNN: The CRRA utility is -0.2392. The social utility is maximized when targeting about 19

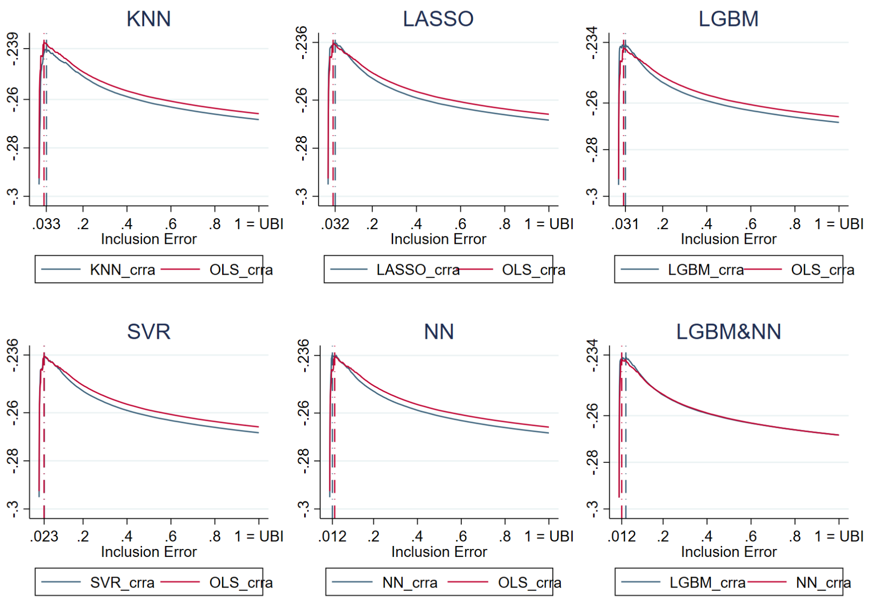
percent of the population with inclusion error of 7.7 percent and exclusion error of 51.9 percent.

- Light GBM: The CRRA utility is -0.2348. The social utility is maximized when targeting about 21 percent of the population with inclusion error of 7.8 percent and exclusion error of 45.0 percent.
- SVR: The CRRA utility is -0.2364. The social utility is maximized when targeting about 18 percent of the population with inclusion error of 6.2 percent and exclusion error of 51.7 percent.
- NN: The CRRA utility is -0.2358. The social utility is maximized when targeting about 13 percent of the population with inclusion error of 3.8 percent and exclusion error of 63 percent.

We draw the relevant Utility-inclusion error figures and compare them to the original utility line with OLS estimation. Light GBM performs the best as it has the highest utility peak with CRRA utility of -0.2348, and Neural Network ranked the second with CRRA utility of -0.2358.

With CRRA utility, a higher ρ means a higher weights on transfers received by the very poor. We also change ρ from 3 to 5 as our robustness check and display them in Figure 8 in our Appendix.

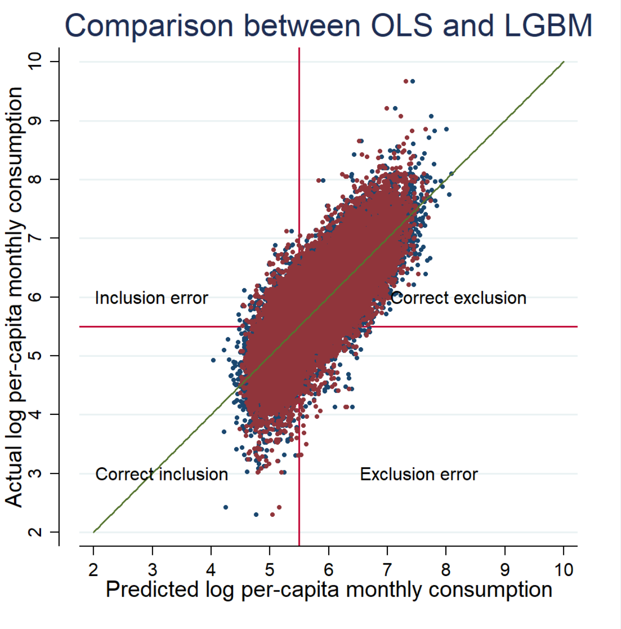
FIGURE 5. Comparison among models for socially optimal program($\rho = 3$)



From Figure 6, it can be shown that the best model from our machine learning approaches, Light

GBM[4], performs almost the same as the baseline model, OLS[5], when considering the inclusion and exclusion error. Predictions with lower MSE cannot target the poverty better and hence improve the CRRA utility (the total social welfare) all the time. For instance, since generally the decision makers on targeted transfers will not set a cutoff line of a very high value, if a model has extremely imprecise and unreliable prediction for the rich but accurately estimates the economic status of people or households around the poverty line, it still correctly includes the people or households those who should be targeted and correctly exclude the people or household those who should not be benefited, in which way the model with high MSE but still effectively targets the poverty.

FIGURE 6. The Inclusion and Exclusion Error of OLS and Light GBM



5 CONCLUSION AND DISCUSSION

In this study, we construct new proxy-mean tools by exploiting several machine learning methods to solve the poverty prediction problem, in which LASSO, SVR and Light GBM make more accurate predictions in the $lnpercapita\ consumption$ context, while LASSO, Light GBM and Neural Network

[4] *Note*: The prediction from Light GBM represented by the blue dots in the figure.

[5] *Note*: The prediction from OLS represented by the red dots in the figure.

have better performance in the *percapita consumption* context.

As discussed above, on average, improving the accuracy of poverty measurement helps to increase the total social welfare.However, there is still much work to be done on making a proper targeted transfer.

We only perform some basic machine learning algorithms on our dataset. Due to the computational expense, we are not able to find out the actual best parameter for each of the model, thereby cannot give perfect predictions.

Another limitation of this report is that we simply use MSE as the metric for evaluating the prediction accuracy. However, MSE can only measure an overall prediction error of the whole sample, but cannot give us explicit information of some specific subsets of the sample (like whether the households near the poverty line are predicted correctly or not), which is more helpful for the poverty targeting problem.

Further work worthy to try with machine learning approaches is to use other datasets containing more information, such as the satellite imagery data, which is more transparent and hard to counterfeited. In this way, it may well explain the local economic status and therefore facilitates to target poverty in developing countries.

REFERENCES

Cortes, Corinna and Vladimir Vapnik (1995, September). Support-vector networks. *Mach. Learn. 20*(3), 273–297.

Cover, T. and P. Hart (2006, September). Nearest neighbor pattern classification. *IEEE Trans. Inf. Theor. 13*(1), 21–27.

Ke, Guolin , Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Red Hook, NY, USA, pp. 3149–3157. Curran Associates Inc.

Kozbur, Damian (2020). Analysis of Testing-Based Forward Model Selection. *Econometrica 88*(5), 2147–2173.

Mhaskar, H. N. and Charles A. Micchelli (1992). Approximation by superposition of sigmoidal and radial basis functions. *Advances in Applied Mathematics 13*(3), 350 – 373.

Tibshirani, Robert (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological) 58*(1), 267–288.

APPENDIX

FIGURE 7. Missing Value Visualization

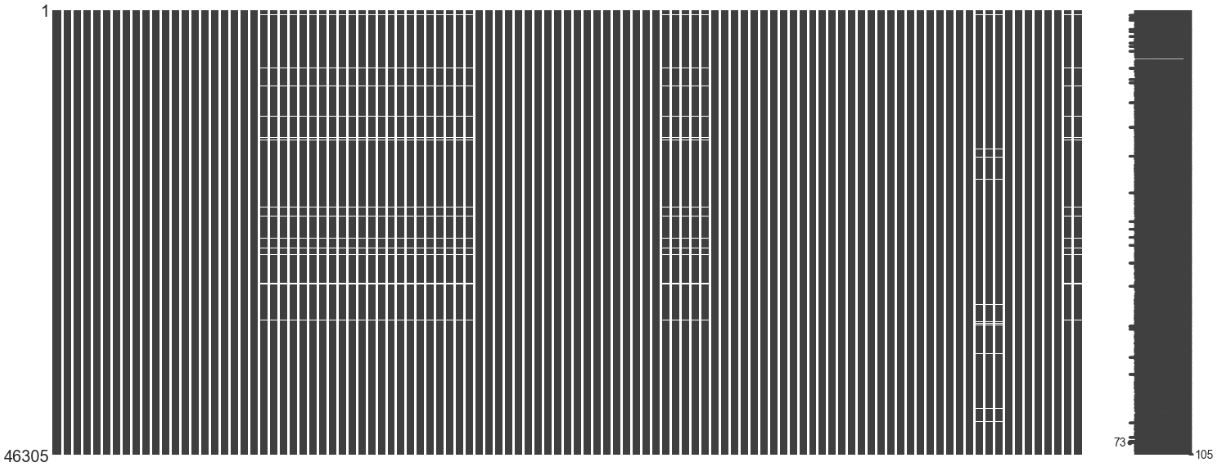


FIGURE 8. Comparison models for socially optimal program($\rho = 5$)

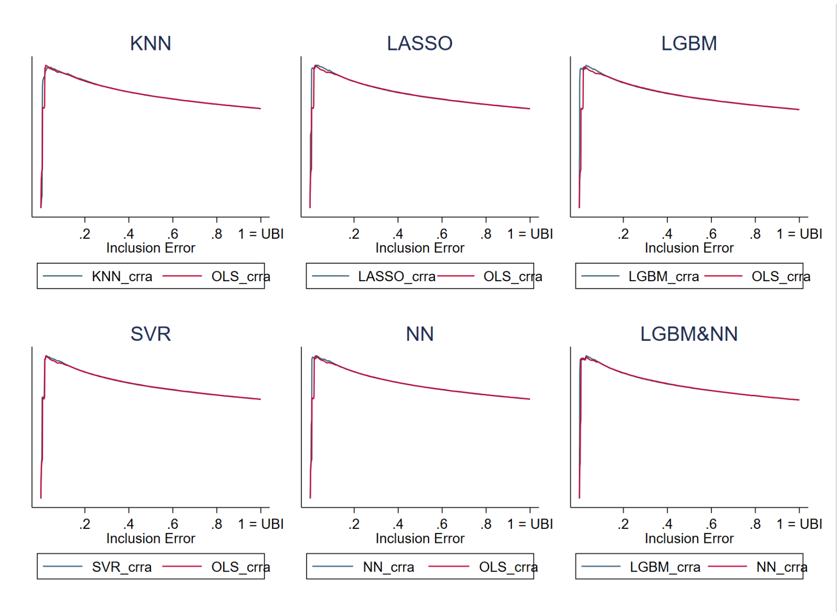


TABLE 2. Summary Statistics: Training Data with 72 Variables

Variable	Mean	Std. Dev.	Min.	Max.	N
d_fuel_other	0.105	0.307	0	1	23153
d_fuel_wood	0.307	0.461	0	1	23153
d_fuel_coal	0.026	0.161	0	1	23153
d_fuel_kerosene	0.004	0.067	0	1	23153
d_fuel_gas	0.519	0.5	0	1	23153
d_fuel_electric	0.008	0.087	0	1	23153
d_fuel_none	0.031	0.173	0	1	23153
d_water_other	0.04	0.196	0	1	23153
d_water_river	0.209	0.406	0	1	23153
d_water_well	0.041	0.199	0	1	23153
d_water_truck	0.017	0.129	0	1	23153
d_water_pylon	0.02	0.14	0	1	23153
d_water_outside	0.064	0.245	0	1	23153
d_water_inside	0.608	0.488	0	1	23153
d_drain_none	0.149	0.356	0	1	23153
d_drain_river	0.021	0.142	0	1	23153
d_drain_cesspool	0.14	0.347	0	1	23153
d_drain_septic	0.155	0.362	0	1	23153
d_drain_outside	0.06	0.238	0	1	23153
d_drain_inside	0.47	0.499	0	1	23153
d_wall_other	0.049	0.216	0	1	22675
d_wall_woodmat	0.112	0.316	0	1	22675
d_wall_stonemud	0.014	0.118	0	1	22675
d_wall_quincha	0.024	0.152	0	1	22675
d_wall_tapia	0.101	0.301	0	1	22675
d_wall_adobe	0.301	0.458	0	1	22675
d_wall_stonecement	0.006	0.079	0	1	22675
d_wall_brickcement	0.393	0.489	0	1	22675
d_roof_other	0.012	0.108	0	1	22675
d_roof_straw	0.073	0.26	0	1	22675
d_roof_mat	0.012	0.111	0	1	22675
d_roof_platecane	0.5	0.5	0	1	22675
d_roof_tile	0.118	0.323	0	1	22675
d_roof_wood	0.013	0.115	0	1	22675
d_roof_concrete	0.271	0.444	0	1	22675
d_floor_other	0.013	0.112	0	1	22675
d_floor_earth	0.393	0.488	0	1	22675
d_floor_cement	0.406	0.491	0	1	22675
d_floor_wood	0.078	0.268	0	1	22675
d_floor_tile	0.058	0.233	0	1	22675
d_floor_sheets	0.03	0.172	0	1	22675
d_floor_parquet	0.022	0.147	0	1	22675
d_electricity	0.837	0.369	0	1	23153
d_telephone	0.203	0.402	0	1	23153
d_h_educ_none	0.08	0.271	0	1	23152
d_h_educ_pre	0	0.017	0	1	23152
d_h_educ_prim	0.384	0.486	0	1	23152
d_h_educ_sec	0.331	0.471	0	1	23152
d_h_educ_higher_nouni	0.104	0.305	0	1	23152
d_h_educ_higher_uni	0.087	0.282	0	1	23152
d_h_educ_post	0.013	0.114	0	1	23152
d_max_educ_none	0.033	0.178	0	1	23153
d_max_educ_prim	0.171	0.376	0	1	23153
d_max_educ_sec	0.402	0.49	0	1	23153
d_max_educ_higher_nouni	0.173	0.379	0	1	23153
d_max_educ_higher_uni	0.198	0.399	0	1	23153
d_insurance_0	0.176	0.381	0	1	23153
d_insurance_1	0.164	0.37	0	1	23153
d_insurance_2	0.164	0.371	0	1	23153
d_insurance_3	0.154	0.361	0	1	23153
d_insurance_4plus	0.341	0.474	0	1	23153
d_crowd_lessthan1	0.301	0.458	0	1	22675
d_crowd_1to2	0.431	0.495	0	1	22675
d_crowd_2to4	0.202	0.402	0	1	22675
d_crowd_4to6	0.047	0.211	0	1	22675
d_crowd_6plus	0.019	0.137	0	1	22675
d_lux_0	0.542	0.498	0	1	23153
d_lux_1	0.196	0.397	0	1	23153
d_lux_2	0.114	0.318	0	1	23153
d_lux_3	0.068	0.252	0	1	23153
d_lux_4	0.044	0.204	0	1	23153
d_lux_5	0.036	0.186	0	1	23153
d_month_1	0.083	0.275	0	1	23153
d_month_2	0.085	0.279	0	1	23153
percapitaconsumption	459.016	401.572	8.25	7030.769	23153
lnpercapitaconsumption	5.854	0.742	2.11	8.858	23153
percapitahat_OLS	458.472	291.706	-57.498	2132.654	22674
lncaphat_OLS	5.849	0.604	4.434	7.738	22674

TABLE 3. Summary Statistics: Test Data with 72 Variables

Variable	Mean	Std. Dev.	Min.	Max.	N
d_fuel_other	0.103	0.305	0	1	23152
d_fuel_wood	0.307	0.461	0	1	23152
d_fuel_coal	0.025	0.156	0	1	23152
d_fuel_kerosene	0.005	0.068	0	1	23152
d_fuel_gas	0.519	0.5	0	1	23152
d_fuel_electric	0.008	0.089	0	1	23152
d_fuel_none	0.033	0.178	0	1	23152
d_water_other	0.043	0.202	0	1	23152
d_water_river	0.207	0.405	0	1	23152
d_water_well	0.04	0.195	0	1	23152
d_water_truck	0.018	0.131	0	1	23152
d_water_pylon	0.019	0.135	0	1	23152
d_water_outside	0.064	0.245	0	1	23152
d_water_inside	0.61	0.488	0	1	23152
d_drain_none	0.15	0.357	0	1	23152
d_drain_river	0.023	0.149	0	1	23152
d_drain_cesspool	0.132	0.338	0	1	23152
d_drain_septic	0.157	0.363	0	1	23152
d_drain_outside	0.059	0.235	0	1	23152
d_drain_inside	0.475	0.499	0	1	23152
d_wall_other	0.052	0.222	0	1	22707
d_wall_woodmat	0.115	0.319	0	1	22707
d_wall_stonemud	0.012	0.109	0	1	22707
d_wall_quincha	0.023	0.151	0	1	22707
d_wall_tapia	0.1	0.3	0	1	22707
d_wall_adobe	0.298	0.457	0	1	22707
d_wall_stonecement	0.006	0.076	0	1	22707
d_wall_brickcement	0.394	0.489	0	1	22707
d_roof_other	0.012	0.109	0	1	22707
d_roof_straw	0.07	0.255	0	1	22707
d_roof_mat	0.014	0.118	0	1	22707
d_roof_platecane	0.499	0.5	0	1	22707
d_roof_tile	0.119	0.324	0	1	22707
d_roof_wood	0.014	0.118	0	1	22707
d_roof_concrete	0.272	0.445	0	1	22707
d_floor_other	0.011	0.106	0	1	22707
d_floor_earth	0.395	0.489	0	1	22707
d_floor_cement	0.402	0.49	0	1	22707
d_floor_wood	0.081	0.273	0	1	22707
d_floor_tile	0.057	0.231	0	1	22707
d_floor_sheets	0.03	0.172	0	1	22707
d_floor_parquet	0.023	0.148	0	1	22707
d_electricity	0.839	0.368	0	1	23152
d_telephone	0.206	0.405	0	1	23152
d_h_educ_none	0.082	0.274	0	1	23149
d_h_educ_pre	0	0.011	0	1	23149
d_h_educ_prim	0.381	0.486	0	1	23149
d_h_educ_sec	0.332	0.471	0	1	23149
d_h_educ_higher_nouni	0.102	0.302	0	1	23149
d_h_educ_higher_uni	0.089	0.285	0	1	23149
d_h_educ_post	0.014	0.118	0	1	23149
d_max_educ_none	0.032	0.176	0	1	23152
d_max_educ_prim	0.172	0.377	0	1	23152
d_max_educ_sec	0.409	0.492	0	1	23152
d_max_educ_higher_nouni	0.166	0.372	0	1	23152
d_max_educ_higher_uni	0.196	0.397	0	1	23152
d_insurance_0	0.182	0.386	0	1	23152
d_insurance_1	0.163	0.37	0	1	23152
d_insurance_2	0.163	0.369	0	1	23152
d_insurance_3	0.147	0.354	0	1	23152
d_insurance_4plus	0.344	0.475	0	1	23152
d_crowd_lessthan1	0.3	0.458	0	1	22707
d_crowd_1to2	0.423	0.494	0	1	22707
d_crowd_2to4	0.209	0.407	0	1	22707
d_crowd_4to6	0.049	0.216	0	1	22707
d_crowd_6plus	0.019	0.137	0	1	22707
d_lux_0	0.547	0.498	0	1	23152
d_lux_1	0.192	0.394	0	1	23152
d_lux_2	0.113	0.317	0	1	23152
d_lux_3	0.066	0.249	0	1	23152
d_lux_4	0.044	0.205	0	1	23152
d_lux_5	0.037	0.188	0	1	23152
d_month_1	0.084	0.277	0	1	23152
d_month_2	0.085	0.279	0	1	23152
percapitaconsumption	463.472	420.982	9.979	15737.628	23152
lnpercapitaconsumption	5.859	0.746	2.3	9.664	23152
percapitahat_OLS	459.77	295.603	-13.181	2132.654	22704
lncaphat_OLS	5.851	0.607	4.492	7.738	22704

TABLE 4. Summary Statistics: per-capita consumption by different dominio

Variable	Mean	Std. Dev.	Min.	Max.	N
dominio1: percapitaconsumption	489.379	355.214	40.061	6813.354	6255
dominio2: percapitaconsumption	542.672	316.656	61.19	3615.231	3582
dominio3: percapitaconsumption	610.139	412.159	28	5203.798	2638
dominio4: percapitaconsumption	264.995	271.653	20.514	4063.173	2931
dominio5: percapitaconsumption	335.942	305.722	8.25	9958.103	8908
dominio6: percapitaconsumption	382.114	376.117	23.583	6938.554	6811
dominio7: percapitaconsumption	441.862	403.942	9.979	15737.628	9715
dominio8: percapitaconsumption	746.365	576.014	80.947	8678.030	5465