Machine Learning with LIS

Pedro Salas-Rojo, International Inequalities Institute (LSE)

Gdansk, September, 2023

Welcome!

Machine Learning (ML) has transformed the social sciences profession (Varian, 2014; Athey and Imbens, 2019).

Applications range from variable selection to causal inference. Practically *everything* you have learned in econometrics has a ML counterpart.

Today we will introduce a set of tools that you can apply in your research.

WARNING: This is an **introductory** lecture. Use the results carefully.

What is ML? (More in Hastie et al. (2009))

- Algorithms that perform tasks using statistical methods.
- Data-driven, while allowing for theoretical restrictions.

Two main families:

- Supervised Learning: use information on regressors (X) to approximate a data-generating process (Y).
- Unsupervised Learning: clustering, PCA, text analysis, pattern detection, outlier identification...

Structure of the Lecture

- Basic understanding of the bias-variance trade-off.
- Parametric Machine Learning: Regularized regression (LASSO).
    - Theoretical introduction: What is LASSO?
    - Application using LISSY: Compare LASSO performance vs OLS.
    - Other parametric tools and applications
- Non-Parametric Machine Learning: Trees and Random Forests.
    - Theoretical introduction: What are trees and random forests?
    - Application using LISSY: Explore predictors of financial behavior.
    - Other non-parametric tools and applications.

Supervised Machine Learning

Many settings in social sciences are based on prediction, $Y = f(X) + \epsilon$.

- Prediction in-sample has trivial solutions (you can always add a regressor to rise the R^2).
- Surveys are usually representative, but do not comprehend the complete population.
- We should aim for out-of-sample prediction.
- Supervised learning elaborates on: **What is the best model -given X- to predict Y out of sample?**

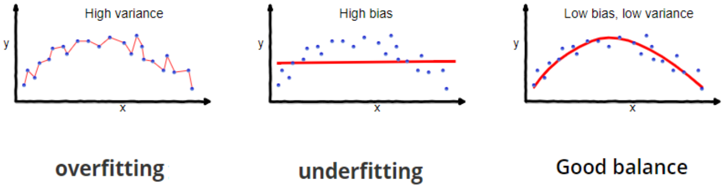
Prediction problem (more in Hastie et al. (2009))

$$Y = f(X) + \epsilon \quad (1)$$

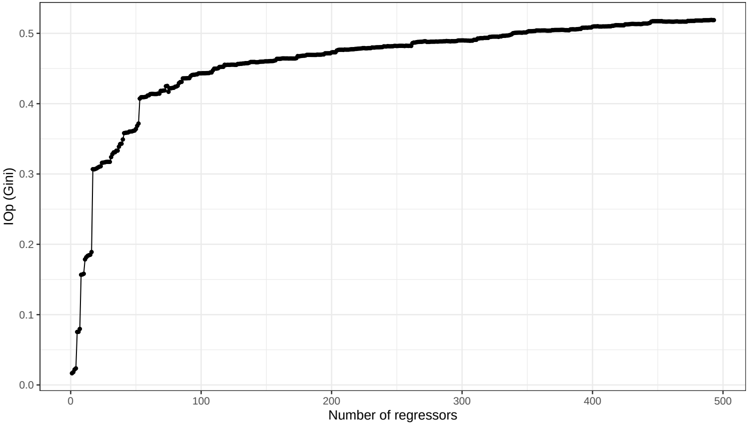
$$\hat{Y} = \hat{f}(X) \quad (2)$$

$$E[(Y - \hat{f}(X))^2] = \text{var}(\hat{f}(X)) + \text{bias}[\hat{f}(X)]^2 + \sigma_\epsilon^2 \quad (3)$$

- $\text{var}(\hat{f}(X))$: If X is too large, prediction is overfitted.
- $\text{bias}[\hat{f}(X)]^2$: If X is too restricted, prediction is underfitted.



Why it this important? (from Brunori et al. (2023a))



OLS vs LASSO

An OLS finds a set of β assigned to X minimizing:

$$\sum_{i=1}^{N}(Y_i - f(\beta X_i))^2 \tag{4}$$

A LASSO regression (Tibshirani, 1996) includes a penalization term:

$$\sum_{i=1}^{N}(Y_i - f(\beta X_i))^2 + \lambda \sum_{x=1}^{X} |\beta_x| \tag{5}$$

Some β will be shrunk to zero. LASSO selects variables that minimize the sum of squared errors.

Example with LISSY: Setup

Data: 'de20' from LIS data.

Basic data arrangement: age between 30 and 60, only those with positive incomes (USD2017, PPP adjusted), a random sample of 3,000 individuals.

pilabour = sex + factor(marital) + factor(educlev) + factor(age5num) + factor(status1) + factor(ind1_c) + factor(occ1_c) + disabled

Answer a simple question: What is the best (out of sample) set of predictors of "pilabour"?

$$\sum_{i=1}^{N}(Y_i - f(\beta X_i))^2 + \lambda \sum_{x=1}^{X} |\beta_x| \tag{6}$$

Example with LISSY: OLS vs LASSO (λ=225)

- OLS output:

```
Coefficients: (2 not defined because of singularities)
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          75787.9    49369.1   1.535 0.124842
sex                 -27485.9     2680.3 -10.255  < 2e-16 ***
factor(marital)120   -3773.8    11845.2  -0.319 0.750056
factor(marital)210   -3194.2     2829.7  -1.129 0.259056
factor(marital)221    3972.4     6246.9   0.636 0.524878
factor(marital)222    1186.9     3622.1   0.328 0.743167
factor(marital)223    4333.2    11328.0   0.383 0.702098
factor(educlev)130   10243.4     8405.6   1.219 0.223061
factor(educlev)210   12954.9     7446.5   1.740 0.081992 .
factor(educlev)220   14164.9     9653.2   1.467 0.142362
factor(educlev)311   19686.2     8773.4   2.244 0.024904 *
factor(educlev)312   26186.5     7933.4   3.301 0.000974 ***
```

- LASSO output:

```
462 x 1 sparse Matrix of class "dgCMatrix"
                             s0
(Intercept)                   .
sex                -25420.10036
factor(marital)120  -1495.75340
factor(marital)210  -3679.47973
factor(marital)221   2256.21696
factor(marital)222    368.32003
factor(marital)223   2449.37314
factor(ind1_c)2               .
factor(ind1_c)3               .
factor(ind1_c)5               .
factor(ind1_c)6               .
factor(ind1_c)8               .
factor(ind1_c)10    -4088.25633
factor(ind1_c)11    -1423.21717
factor(ind1_c)13     3791.99901
factor(ind1_c)14   -53574.89051
factor(ind1_c)15              .
factor(ind1_c)16     5237.07091
```

LASSO with different λ

- LASSO with low λ ($\lambda = 5$):

```
462 x 1 sparse Matrix of class "dgCMatrix"
                              s0
(Intercept)                    .
sex                  -27488.277540
factor(marital)120    -3809.524050
factor(marital)210    -3172.092038
factor(marital)221     3886.208889
factor(marital)222     1190.832368
factor(marital)223     4414.877892
factor(ind1_c)2       -7809.972543
factor(ind1_c)3       11777.459452
factor(ind1_c)5        2003.702963
factor(ind1_c)6          62.900050
factor(ind1_c)8      -13695.019525
factor(ind1_c)10      -1303.631565
factor(ind1_c)11      -6335.100605
```

- LASSO with high λ ($\lambda = 3000$):

```
>   print(coeff_lasso)
462 x 1 sparse Matrix of class "dgCMatrix"
                            s0
(Intercept)                  .
sex                 -21846.96750
factor(marital)120           .
factor(marital)210   -1011.52180
factor(marital)221           .
factor(marital)222           .
factor(marital)223           .
factor(ind1_c)2              .
factor(ind1_c)3              .
factor(ind1_c)5              .
factor(ind1_c)6              .
factor(ind1_c)8              .
factor(ind1_c)10             .
factor(ind1_c)11             .
```

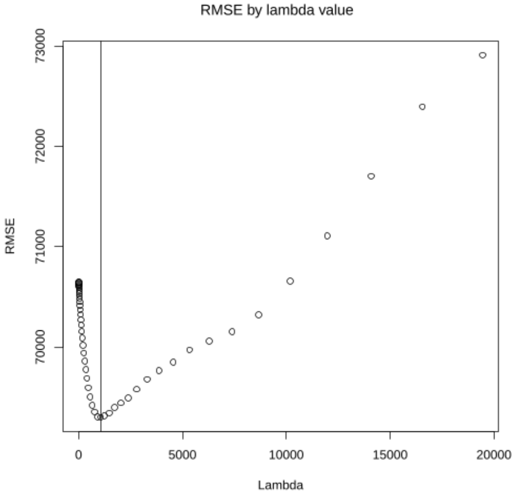
What λ? K-fold Cross-Validation

Select λ that optimizes the out-of-sample prediction (RMSE).

- Divide the sample in k folds.
- Define a grid of λ values to search in.
- Take k-1 folds (training sample) and run the model. Use one λ and run the LASSO regression.
- Predict in fold k (test sample). Estimate RMSE.
- Repeat leaving other fold k out.
- After all k have been used as test samples, average RMSE.
- Repeat all other λ candidates.
- Select λ* associated with the smallest averaged RMSE.

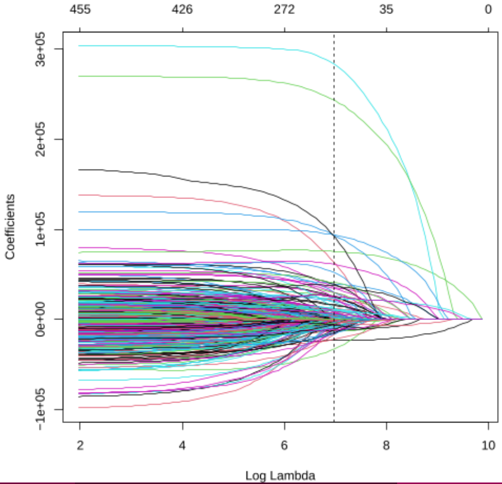
Example with LISSY: Cross-Validation

- Grid: Set endogenously by function glmnet.



Example with LISSY: Cross-Validation

- $\lambda = 1065$, $\log(\lambda)$=6.97.



How can you run a LASSO with LISSY?

Package 1: "glmnet" Friedman et al. (2021). Estimate LASSO (and other similar parametric) regression.

Package 2: "caret" Kuhn (2015). Tune and obtain optimum parameters, as well as out-of-sample RMSE.

Both installed in LISSY. Most functions and plug-ins are similar to those in standard regressions.

Your turn: Tune the LASSO

- Get λ grid:

```
exploremodel <- glmnet::cv.glmnet(x = vec, y = dep, alpha = 1)
range(exploremodel$lambda)
lambda_range <- exp(seq(log(min(exploremodel$lambda)),
                        log(max(exploremodel$lambda)),
                        length.out = 50))
print(lambda_range)
```

- Change the number of folds used in the tuning:

```
method = "cv", number = 3,
verboseIter = TRUE,   savePredictions = "all"),
```

Your turn: play LASSO with several λ

- Change the model, including or excluding variables you want to use.

```
dep <- data$pilabour
vec <- model.matrix( ~ sex + factor(marital) + factor(educlev) +
                       factor(status1) + factor(ind1_c) + factor(occ1_c) +
                       factor(age5num) + disabled, data)
```

- Update λ.

```
lasso <- glmnet(vec, dep, alpha=1, lambda = lambda)
coeff2 <- lasso$beta
```

Exercise: Compare OLS vs LASSO with pilabour

If you want to check that you learned how this works.

- Select a dependent and regressors of your choice. Use as many X as possible!
- Tune λ: Define λ grid and the number of folds.
- Check with the tune-plot that this tunning is appropriate (is it the minimum of the curve?).
- Run LASSO and OLS. Check coefficients.
- Check both RMSE's.

In the example script RMSE's are, OLS=70,612 and LASSO=69,302, an improvement of 1.86%. Try to beat it!

Some properties of LASSO

Imagine you want to approximate $Y = f(\chi) + \epsilon$. You have a few thousands of observations, and many regressors in χ, that you want to interact.

Which $X \in \chi$ you should use? Let LASSO decide.

Coefficients (β's) **cannot** be interpreted as "marginal effects", but you can use a "post-LASSO" (Hufe et al., 2021).

You can include weights and other features from standard OLS, or exclude variables from regularization.

Some other regularizers

A RIDGE regression (Tikhonov, 1963) includes a different penalization term:

$$\sum_{i=1}^{N}(Y_i - f(\beta X_i))^2 + \lambda \sum_{x=1}^{X} \beta_x^2 \quad (7)$$

An ELASTIC NET regression (Zou and Hastie, 2005) combines both:

$$\sum_{i=1}^{N}(Y_i - f(\beta X_i))^2 + \lambda \sum_{x=1}^{X} \beta_x^2 + \theta \sum_{x=1}^{X} |\beta_x| \quad (8)$$

Also: relaxed LASSO, post-regularizers,... They are all in LISSY.

Some applications:

Oaxaca-Blinder decomposition of the gender gap.

- Many covariates and interactions can explain the gender gap.
- LASSO selects the most relevant.
- See Böheim and Stöllinger (2021).

Inequality of opportunity and income mobility.

- Can circumstances predict incomes?
- LASSO selects without overfitting.
- See Hufe et al. (2021) or Bloise et al. (2021).

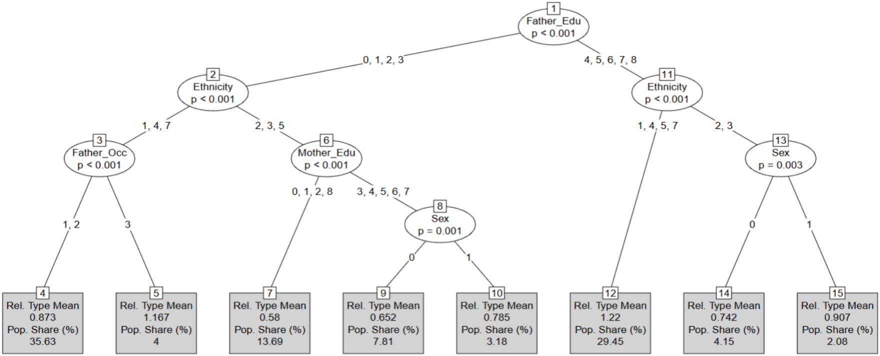
Used for instrument selection (Belloni et al., 2010), predicting financial markets behavior (Lee et al., 2022),...

LASSO works, but:

- Often not easy to interpret.
- Changes in the data affects the model selection. You can "bootstrap", but sometimes a more robust version is needed.
- They are not great to detect non-nonlinearities in the data generating process. You can interact as many regressors as you want, but it takes time to fit...

If you are not interested in variable selection and/or you suspect your $f(X)$ is very non-linear, you should consider trees.

Conditional Inference Trees (CIT, Hothorn et al. (2006))



Example (USA, 1980) from the Global Estimates of Opportunity and Mobility (GEOM) Database.

How does a CIT grow?

In the end, they are a regression $Y = f(X)$. Their structure follows these steps:

- Set an α,
- Search for the most correlated regressor running an independence test. If the (Bonferroni) p-value is bigger than α, stop the algorithm. Otherwise, continue,
- Search for binary splits. Compare means across resulting nodes (use a t-test) and select the one associated with the smallest p-value,
- Repeat in each resulting node until the algorithm stops everywhere.

How deep does a CIT grow?

- α: stops the algorithm.
- minbucket: minimum number of observations in each terminal node.
- minsplit: minimum number of observations to be considered as a splitting node.
- maxdepth: maximum depth of the tree

All of them can be tuned with k-fold cross-validation! However, they can also be set theoretically (i.e., $\alpha = 0.01$).

We are focusing on α, but note that in your own applications you should consider all parameters.

Example with LISSY: Explore predictors of financial behavior

Data: 'es21'

Basic data arrangement: age between 25 and 75, focus on first imputation set.

$$\text{saves} = \text{age} + \text{sex} + \text{factor(marital)} + \text{factor(health\_c)} + \text{factor(educlev)} + \text{factor(status1)} + \text{factor(ind1\_c)} + \text{factor(occ1\_c)}$$

Simple question: what is the best set of predictors of saving capacity at the end of the year? (basb=saves, 1 = saves, 0 = does not save).

Ctree with LISSY

Package 1: "partykit" Hothorn and Zeileis (2015). Estimate Ctree (and random forest, see later).

Package 2: "caret" Kuhn (2015). Tune and obtain the optimum α, as well as out-of-sample RMSE.

Both are installed in LISSY.

There is a previous version of "partykit" called "party". Caret uses party. Some functions are not compatible!

Your Turn: Tune a Tree

Since the dependent is binary, we maximize accuracy! We cannot use RMSE.

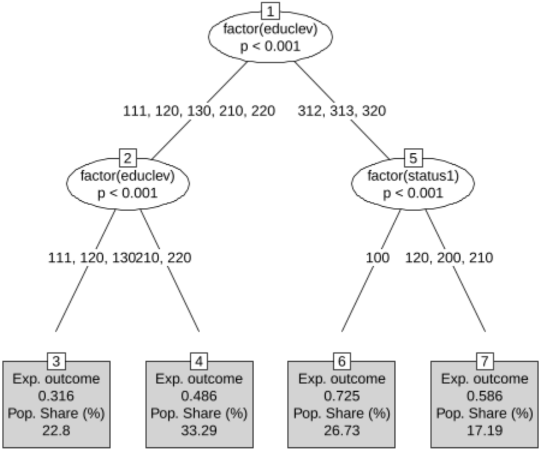
```
# Set model
model <- factor(saves) ~ sex + factor(educlev)

# Set cross-validation method and number of folds
cv <- trainControl(method = "cv", number = 5,
                    verboseIter = FALSE)

# Define grid of (1-alpha) used to tune the algorithm.
grid <- expand.grid(mincriterion = seq(0.9, 0.995, 0.005))

tr_train <- caret::train(model,
                         data = data,
                         method = "ctree",
                         trControl = cv,
                         tuneGrid = grid,
                         controls = ctree_control(minbucket = 100))
```

Your Turn: Play with model and parameters

- Change the model.
  - Include as many regressors as you want.
  - Note that for binary regressions, the dependent has to be a "factor".
  - You do not have to specify interactions, the tree searches for them!

```
tree <- partykit::ctree(model,
                        data = data,
                        control = ctree_control(testtype = "Bonferroni",
                                                teststat = "quad",
                                                alpha = 0.01,
                                                minbucket = 100,
                                                minsplit = 300,
                                                maxdepth = 6))
```

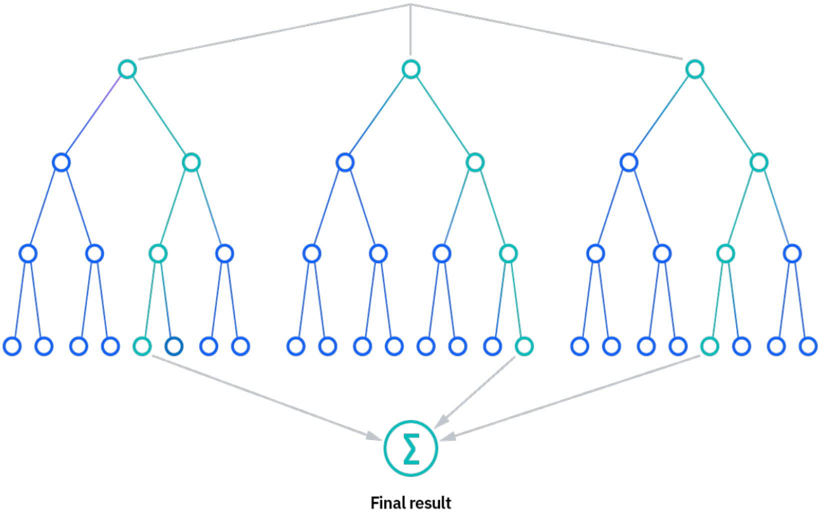
Tree Plot ($\alpha = 0.05$)



EXERCISE 3: Use a tree to explore financial behavior

- Get the deepest possible tree. How many terminal nodes do you get?
- Search for a model that maximizes the out-of-sample accuracy. Use caret!
- Explore the structure of the tree with other dependent variables: basp1, basp2, basp3. Are they different? Does the prediction capacity of your model improve or worsen?
- How stable is the structure of trees when you change the regressors?

Solution: Random Forest



Scheme: Random Forest

- Get a subsample (no replacement) from the data,
- Run a tree (usually set $\alpha = 1$). In each node, select a subset of regressors to test independence
- Store prediction,
- Repeat N times,
- Average across all predictions.

Averaging across many "bad" predictions leads to very good predictions (See Rubin (1996) and the literature on multiple imputation!)

Variable importance (Strobl et al. (2008))

- Each tree grows from a subset of regressors,
- Store the fall in accuracy or prediction capacity after dropping one regressor,
- After many trees, obtain a score of the average change in prediction capacity associated with each regressor,
- Set the maximum value of the score to 100, and index the rest accordingly.

The idea is quite close to a Shapley value decomposition (Shorrocks (2013); Brunori et al. (2023a))

Your turn: Random Forest and variable importance

- You can easily modify a random forest object

```
forest <- partykit::cforest(model,
                            data = data,
                            ntree = 100,
                            mtry = 5,
                            trace = FALSE,
                            control = ctree_control(testtype = "Bonferroni",
                                                    teststat = "quad",
                                                    mincriterion = 0,
                                                    minbucket = 10))
```

- and get variable importance

```
>   relimp <- relimp[order(-relimp)]
>   print(relimp)
 factor(educlev)   factor(occ1_c)    factor(ind1_c)  factor(status1)
          100.00            43.18             26.21            25.85
             age   factor(marital)              sex factor(health_c)
           13.80             7.32              6.41            -0.99
>
>
```

EXERCISE 4: Use a random forest to explore financial behavior

- What is the relative importance of regressors in your model?
- What is the relative importance of regressors when explaining other dependent variables: basp1, basp2, basp3. Are they different?
- How stable is the variable importance when you drop regressors?

Some applications:

Trees and Random Forests are widely popular now:

- Estimate Inequality of Opportunity (Brunori et al. (2023b))
- Estimate relation between inheritances and wealth inequality (Salas-Rojo and Rodríguez (2022))
- Identify heterogeneous causal effects on treatment assignments (Wager and Athey (2018))
- Address missingness in data (Tang and Ishwaran (2017))
- Explore poverty and vulnerability (Taye and d'Ambrosio (2021))
- Explore financial behaviour, climate impact on socioeconomic factors, forecast labor market fluctuations,...

Summing up

- LASSO is quite good for selecting regressors.
  - Not the best to detect non-linearities.
  - There are many regularizers to explore.

- Trees show the basic structure of the data.
  - Can be unstable.
  - Dozens of types of trees.

- Random Forest are very good for prediction, and provide hints about variable importance.
  - Hard to explore inside.
  - Quite flexible, and performs well in many different settings.
  - Combinations of trees and forests are used in all sorts of settings.

Many thanks!

Happy to chat anytime, drop a line to p.salas-rojo@lse.ac.uk

Athey, S. and Imbens, G. W. (2019). Machine learning methods that economists should know about. *Annual Review of Economics*, 11(1):685–725.

Belloni, A., Chernozhukov, V., and Hansen, C. (2010). Lasso methods for gaussian instrumental variables models. *arXiv preprint arXiv:1012.1297*.

Bloise, F., Brunori, P., and Piraino, P. (2021). Estimating intergenerational income mobility on sub-optimal data: a machine learning approach. *The Journal of Economic Inequality*, 19(4):643–665.

Böheim, R. and Stöllinger, P. (2021). Decomposition of the gender wage gap using the lasso estimator. *Applied Economics Letters*, 28(10):817–828.

Brunori, P., Ferreira, F., and Salas-Rojo, P. (2023a). Inherited inequalities. *mimeo*.

Brunori, P., Hufe, P., and Mahler, D. (2023b). The roots of inequality: Estimating inequality of opportunity from regression trees and forests. *Scandinavian Journal of Economics, forthcoming*.

Friedman, J., Hastie, T., Tibshirani, R., Narasimhan, B., Tay, K., Simon, N., and Qian, J. (2021). Package 'glmnet'. *CRAN R Repositary*.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Hothorn, T., Hornik, K., and Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical statistics*, 15(3):651–674.

Hothorn, T. and Zeileis, A. (2015). partykit: A modular toolkit for recursive partytioning in r. *The Journal of Machine Learning Research*, 16(1):3905–3909.

Hufe, P., Peichl, A., and Weishaar, D. (2021). Lower and upper bound estimates of inequality of opportunity for emerging economies. *Social Choice and Welfare*, pages 1–33.

Kuhn, M. (2015). A short introduction to the caret package. *R Found Stat Comput*, 1:1–10.

Lee, J. H., Shi, Z., and Gao, Z. (2022). On lasso for predictive regression. *Journal of Econometrics*, 229(2):322–349.

Rubin, D. B. (1996). Multiple imputation after 18+ years. *Journal of the American statistical Association*, 91(434):473–489.

Salas-Rojo, P. and Rodríguez, J. G. (2022). Inheritances and wealth inequality: a machine learning approach. *The Journal of Economic Inequality*, 20(1):27–51.

Shorrocks, A. F. (2013). Decomposition procedures for distributional analysis: a unified framework based on the shapley value. Technical Report 1.

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC bioinformatics*, 9:1–11.

Tang, F. and Ishwaran, H. (2017). Random forest missing data algorithms. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 10(6):363–377.

Taye, A. and d'Ambrosio, C. (2021). Predicting vulnerability to poverty with machine learning. In *DTU DRIVEN Colloquium*.

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

Tikhonov, A. N. (1963). Solution of incorrectly formulated problems and the regularization method. *Sov Dok*, 4:1035–1038.

Varian, H. R. (2014). Big data: New tricks for econometrics. *Journal of economic perspectives*, 28(2):3–28.

Wager, S. and Athey, S. (2018). Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228–1242.

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