Machine Learning with LIS

Pedro Salas-Rojo, International Inequalities Institute (LSE)

Gdansk, September, 2023

Basic ideas about ML (more in Hastie et al. (2009))

- Algorithms performing tasks using statistics.
- Data-driven, but allow for theoretical restrictions.

Two big families:

- Supervised Learning: use information on regressors (X) to approximate a data-generating process (Y).
- Unsupervised Learning: cluster, PCA's, find patterns, detect outliers...

Set up

- Basic understanding of the bias-variance trade-off.
- Parametric Machine Learning: Regularized regression (LASSO).
    - Theoretical introduction: what is LASSO?
    - Use LISSY: Compare LASSO performance vs OLS.
- Break.
- Non-Parametric Machine Learning: Trees and Random Forests.
    - Theoretical introduction: what are trees and random forests?
    - Use LISSY: Explore predictors of financial behavior.

Supervised Machine Learning

Many settings in social sciences are based on prediction.

- Prediction in-sample has trivial solutions.
- Ideally, we should aim for out-of-sample prediction.
- Surveys are usually representative, but do not comprehend the complete population.
- Supervised learning elaborates on: **What is the ability of a set X to predict Y out of sample?**

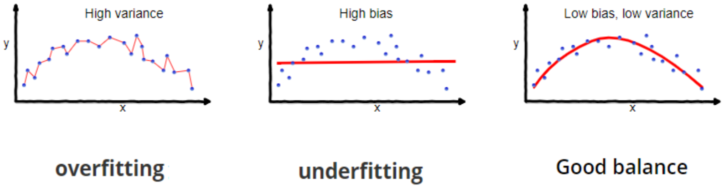
Prediction problem (more in Hastie et al. (2009))

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

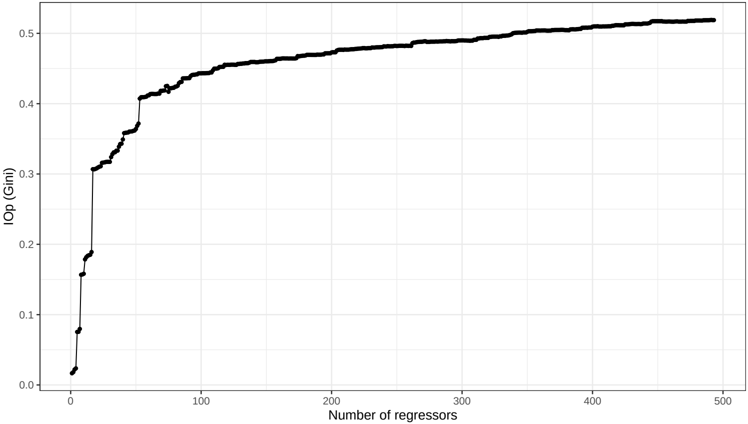
$$\hat{Y} = \hat{f}(X) \quad (2)$$

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

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



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.

Some properties of LASSO

Uncertain about which variables you should use? Let LASSO decide.

Coefficients (β's) cannot be interpreted as "marginal effects" (but you can use a "post-LASSO").

You can include weights and other features from standard OLS.

You can exclude variables from regularization.

Example with LISSY: Setup

Data: 'pl20'

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

$$\text{pilabour} = (\text{sex}) + \text{factor}(\text{marital}) + \text{factor}(\text{educlev}) + \text{factor}(\text{status1}) + \text{factor}(\text{ind1}_c) + factor(occ1_c) + factor(age5num) + (disabled)$$

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

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

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

- OLS output:

```
Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)            44469.1    11750.5   3.784 0.000157 ***
sex                    -5132.1      674.1  -7.613 3.62e-14 ***
factor(marital)210     -1141.3      740.0  -1.542 0.123109
factor(marital)221      -762.5     4029.1  -0.189 0.849908
factor(marital)222      1176.0     1017.1   1.156 0.247665
factor(marital)223      1547.7     1862.3   0.831 0.406007
factor(educlev)120     -2113.4     9795.1  -0.216 0.829187
factor(educlev)130     -6266.9    10667.6  -0.587 0.556933
factor(educlev)210     -1156.0     9655.6  -0.120 0.904709
factor(educlev)220      -333.3     9789.4  -0.034 0.972839
factor(educlev)311     -8725.2    12548.6  -0.695 0.486917
factor(educlev)312      -158.9     9714.8  -0.016 0.986952
factor(educlev)313      2486.8     9679.1   0.257 0.797256
factor(educlev)320      8857.3     9959.5   0.889 0.373896
```

- LASSO output:

```
sex                 -4928.87805
factor(marital)210   -717.10616
factor(marital)221            .
factor(marital)222            .
factor(marital)223            .
factor(educlev)120  -1523.06192
factor(educlev)130  -2418.95263
factor(educlev)210  -1371.22538
factor(educlev)220            .
factor(educlev)311            .
factor(educlev)312     85.70867
factor(educlev)313   3953.30783
factor(educlev)320   8406.47071
```

LASSO with several λ

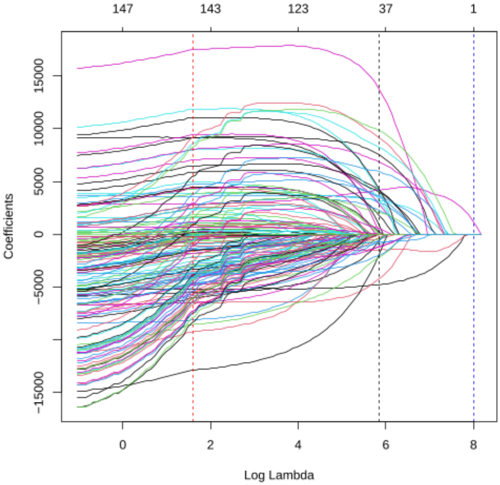
- LASSO with low λ ($\lambda = 5$):

```
sex                    -5168.711736
factor(marital)210     -1161.955387
factor(marital)221      -663.499983
factor(marital)222      1110.179540
factor(marital)223      1566.278014
factor(educlev)120     -1994.005394
factor(educlev)130     -5713.923938
factor(educlev)210     -1049.209486
factor(educlev)220      -207.364513
factor(educlev)311     -8333.376437
factor(educlev)312          .
factor(educlev)313      2650.516356
factor(educlev)320      8960.507379
```

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

```
sex                         .
factor(marital)210          .
factor(marital)221          .
factor(marital)222          .
factor(marital)223          .
factor(educlev)120          .
factor(educlev)130          .
factor(educlev)210          .
factor(educlev)220          .
factor(educlev)311          .
factor(educlev)312          .
factor(educlev)313  1148.819
factor(educlev)320          .
```

LASSO Plot with several (log)λ



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.

Exercise 1: Minimize out of sample RMSE

What is the best (out of sample) set of predictors of "pilabour"? Define manually the model and lambda:

```
model <- pilabour ~ (sex) + factor(marital)

lambda_try <- 225

lasso_tr <- caret::train(model,
                         data = data,
                         method = "glmnet",
                         trControl = trainControl(method = "cv", number = 2,
                                                  verboseIter = TRUE,   savePredictions = "all"),
                         tuneGrid = expand.grid(alpha = 1,
                                                lambda = lambda_try))
```

You will get a response in the script. In this case, "The RMSE of this model, with a lambda of 225, is 14,799" Can you improve it?

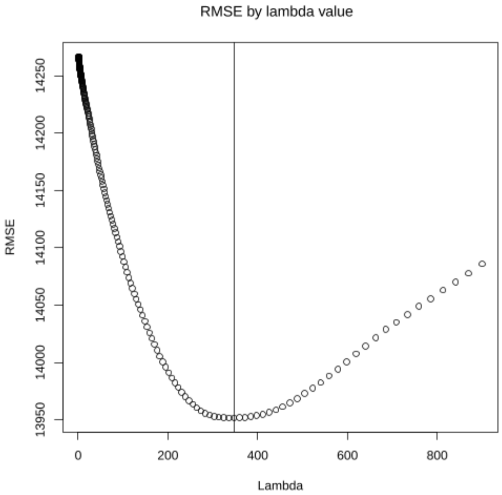
What λ? K-fold Cross-Validation

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

- Divide the sample in k folds.
- Define a set of λ to search in.
- Take k-1 folds (training sample) and run the model. Use one λ and run the LASSO regression.
- Get prediction 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.

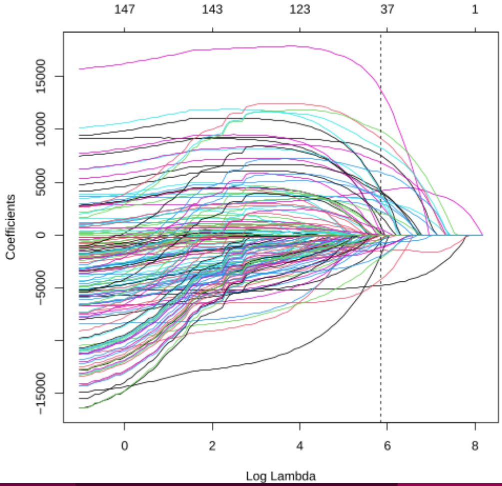
Example with LISSY: Cross-Validation

- Grid: $30^{seq(0.01,2,0.01)}$.



Example with LISSY: Cross-Validation

- $\lambda = 347$, $\log(\lambda)$=5.85.



Your turn: Tune the LASSO

- Change λ grid:

```
expand.grid(alpha = 1,
            lambda = 15^seq(0.01,2,0.01)))
```

- 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 2: Compare OLS vs LASSO with hilabour

- Select a dependent and regressors of your choice.
- 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=14122 and LASSO=13952, an improvement of 1.2%. Can you enlarge it?

Other regularization terms

LASSO is not the only regularizer. In fact, there are many!

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

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

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

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

Also: relaxed LASSO, post-regularizers,...

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.

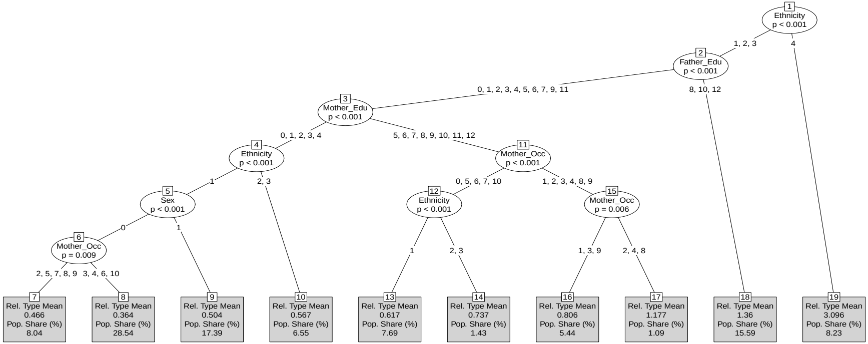
- Can circumstances predict incomes?
- LASSO selects without overfitting.
- See Hufe et al. (2021) or Bloise et al. (2021).

Use for instrument selection, missing imputation, cross-survey imputation, matching,...

Why Regression Trees are cool

- Regularizers work nicely to select variables when X is big.
- However, they are not ideal for exploring non-linearities.
- Trees perform binary splits in the sample leading to exhaustive and mutually exclusive groups.
- Binary splits have limitations but allow exploring non-linearities.
- After groups are defined, trees assign the expectation to each terminal node.

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



Example from Brunori et al. (2023a).

How does a CIT grow?

- 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.

We are focusing on the α, but please note that in your own applications you should go deeper.

Example with LISSY: Explore predictors of financial behavior

Data: 'es17'

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

$$\text{saves} = \text{age} + \text{sex} + \text{factor}(\text{marital}) + \text{factor}(\text{health}_c) + factor(educlev) + factor(status1) + factor(ind1_c) + factor(occ1_c)$$

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))
```

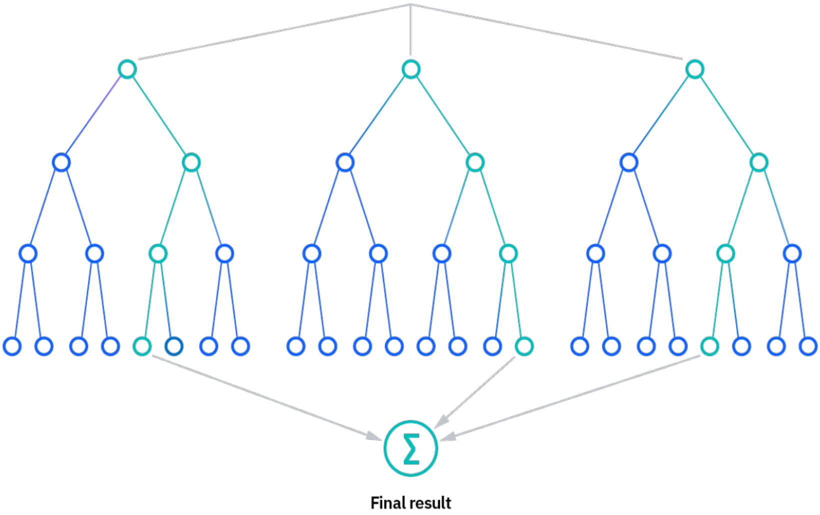
Tree Plot

PLOT SCHEME ON HOW COVARIATES PREDICT FINANCIAL BEHAVIOR

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.
- 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))

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

```
factor(educlev)   factor(occ1_c)   factor(ind1_c)  factor(status1)
         100.00            51.39            36.25            30.47
            sex              age  factor(marital)  factor(health_c)
          18.25            15.46             6.58             2.75
```

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 financial behaviour, climate impact, forecast weather, 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

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.

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.

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.

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.