An Introduction to Machine Learning for Social Scientists

Tyler Ransom

Duke University, Social Science Research Institute

March 30, 2016

Outline

What is machine learning?

- A branch of statistics devoted to accurate prediction
- Maximize both in- and out-of-sample prediction
- Systematically combine estimation and model selection
- Becoming more popular in “big data” era
- These slides based heavily on Varian (2014)

Motivating example

- Suppose you want to predict mortgage loan default (0/1 outcome)
- You have a large number (over 5,000) of relevant variables
- What would you do?
- There are better methods of prediction than logit:
  - Methods to help you determine which of the 5,000 variables are most important
  - Methods to automatically detect interactions among variables
  - Methods that will do a better job of predicting out-of-sample than logit

Measuring fit

- What is meant by “fit”? Prediction “loss” (or “cost”)
- For continuous outcomes, it’s minimizing the sum of squared residuals:

$$\text{loss} = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

- For discrete outcomes, it’s minimizing the (negative) log likelihood analog of a multinomial distribution:

$$\text{loss} = -\sum_{i=1}^{N} \sum_{j=1}^{J} y_{ij} \log (\hat{p}_{ij})$$

Trading off in- and out-of-sample fit

- Suppose you have cross-sectional data on n individual earnings
- One way to predict earnings is to use OLS and estimate n dummy variable coefficients (no constant)
- R^2 will be very close to 1, indicating very good in-sample fit
- But if I gave you a separate sample of this data with m different individuals, how would you predict them? Which dummy coefficients would you assign to the new individuals?
- This is a case of *overfitting*: estimating a model that performs well in-sample but poorly out of sample

Solution to overfitting

1. Penalizing parameter complexity
2. Testing a variety of models out-of-sample
3. Using cross-validation to find the best level of penalty

How cross-validation works

Typical steps used to cross-validate and test predictions:

1. Randomly divide up your data into three parts: training set (60%), cross-validation set (20%), and test set (20%)
2. Estimate your model parameters in the training set
3. Compute the prediction error in both the cross-validation and test sets
4. Repeat this for various levels of penalty
5. Pick the penalty level that minimizes error in the cross-validation set
   - Test set should only be used for out-of-sample prediction; some people lump test/CV together

Outline

Examples of machine learning algorithms

- Continuous dependent variable:
  - Ordinary least squares
  - Regression trees / random forests
  - Penalized regression (LASSO, Ridge, Elastic net)
- Categorical dependent variable:
  - Logistic regression
  - Classification trees / random forests
  - Support vector machines
  - Neural networks

Penalized regression (regularized regression)

- Recall the loss function associated with linear regression:

$$\text{loss} = \sum_{i=1}^{N} \left(y_i - x_i \hat{\beta} \right)^2$$

- Now add on a penalty (regularization) term with weight λ:

$$\text{loss} = \sum_{i=1}^{N} \left(y_i - x_i \hat{\beta} \right)^2 + \lambda \sum_{k=1}^{K} \left[(1 - \alpha) \left| \hat{\beta}_k \right| + \alpha |\hat{\beta}_k|^2 \right]$$

- where λ is the level of penalty ($\lambda = 0 \Rightarrow$ OLS)
- $\alpha = 0 \Rightarrow$ LASSO; $\alpha = 1 \Rightarrow$ ridge regression; $\alpha \in (0, 1) \Rightarrow$ elastic net

Penalized (regularized) logistic regression

- Regularization can also be applied to binary dependent variables:
- Recall the loss function associated with logistic regression:

$$\text{loss} = -\sum_{i=1}^{N} y_i \log(\hat{p}_i) + (1 - y_i) \log(1 - \hat{p}_i)$$

where $\hat{p}_i = \frac{1}{1+\exp(-x_i\beta)}$

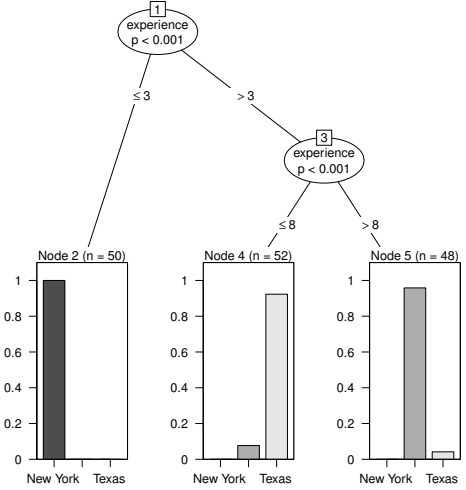
- The elastic net for this case becomes

$$\text{loss} = -\sum_{i=1}^{N} y_i \log(\hat{p}_i) + (1 - y_i) \log(1 - \hat{p}_i) + \lambda \sum_{k=1}^{K} \left[(1 - \alpha) |\hat{\beta}_k| + \alpha |\hat{\beta}_k|^2 \right]$$

Classification and Regression Trees (CART)

- Build a decision tree to classify observations (e.g. $1, \ldots, J$ for categorical outcomes, group means for continuous outcomes)
- Algorithm first selects variables that are most strongly associated with outcome
- It then finds splits of the variables that lead to the most distinct predictions
- Prediction is the mean of the terminal node (“leaf”)
- Automatically detects non-linearities
- Handles missing data well

Tree example



Pruning trees

- Tree algorithms overfit by creating too many branches
- “pruning” is equivalent to penalization
- e.g. choose max number of terminal nodes (“leaves”)

Random Forests

Technique that uses multiple trees

1. Choose a bootstrap sample of the observations and start to grow a tree
2. At each node of the tree, choose a random sample of the predictors to make the next decision. Do not prune the trees
3. Repeat this process many times to grow a forest of trees
4. Each tree makes a classification prediction
5. Use a majority vote from all trees for the final prediction

Support Vector Machine (Bajari et al., 2015)

- Similar to penalized regression
- Instead of adding penalty term, the tuning parameter governs which errors are included in the model
- Errors of sufficiently small size are treated as 0
- Typically, some parameters will be forced to 0

Neural Networks

- Used for prediction of categorical dependent variables
- Employ a structure of axons and dendrites found in brain cells
- Can approximate any arbitrary nonlinear function of covariates
- Use parameter regularization to avoid overfitting

Ensemble prediction

- Often times, you will obtain better prediction by averaging across models (e.g. forests vs. trees; Bajari et al. (2015))
- e.g. obtain predictions from Penalized logistic regression, classification tree, and support vector machine
- Create a meta-prediction by regressing (in the cross-validation set) the outcome on the predictions from each model
- The meta-prediction will usually perform better in the test set than any single prediction

Software to estimate ML models

- `R` is the home of machine learning development
- Matlab has a ML toolbox, but lacks customizability
- Limited availability in Stata (one type of tree algorithm, user-written)
- Limited availability in Julia

Unsupervised learning

- Up to now, we've only discussed *supervised* learning
- *Unsupervised* learning $\Rightarrow$ no dependent variable
- Used primarily to reduce large datasets
- e.g. detect partitions in data (k-means clustering, EM algorithm)
- Reduce dimensionality of data (PCA)

Outline

Limitations of machine learning

- Machine learning is all about *prediction*
- But social science is primarily motivated by *causality* (i.e. prediction in a counterfactual environment)
- Attempts currently being made to re-frame machine learning in terms of causal inference (Varian, 2014; Athey and Imbens, 2015; Bajari et al., 2015)
- These are (currently) largely application-specific

When should I use machine learning?

- If you are mainly interested in prediction
- If you have an intermediate step of your model estimation that requires making predictions
- If you need to compress a prohibitively large data set

References

Athey, Susan and Guido W. Imbens. 2015. “Machine Learning Methods for Estimating Heterogeneous Causal Effects.” Working paper, Stanford University.

Bajari, Patrick, Denis Nekipelov, Stephen P. Ryan, and Miaoyu Yang. 2015. “Demand Estimation with Machine Learning and Model Combination.” Working Paper 20955, National Bureau of Economic Research.

Varian, Hal R. 2014. “Big Data: New Tricks for Econometrics.” *Journal of Economic Perspectives* 28 (2):3–28.