Boosting for exploration and prediction in the social sciences

Christopher Boylan
Department of Political Science

Motivation

Boosting:

- Is a flexible and versatile method
- Has strong predictive performance (extremely popular in data mining competitions)
- Can help identify patterns in data

Outline

1. Machine Learning Basics
2. CART
3. Boosting
4. Application: Using Boosting to Predict Civil War Onsets

Machine Learning Basics

Data: (x_i, y_i) $i = 1, 2, ..., N$

- Predictors: x_i
- Response: y_i
- *Quantitative* response: *Regression* problem
- *Categorical* response: *Classification* problem
- Training Data: Used to fit model
- Validation Data: Unseen data used to tune the model
- Test Data: Unseen data used to evaluate model performance

Goal:

- Find function $\hat{f}$ such that error $L(y, \hat{f}(x))$ for test data is minimal

Machine Learning Basics

Loss function:

- A loss function L measures the discrepancy between a model's prediction and the value of the response

Why use test data?:

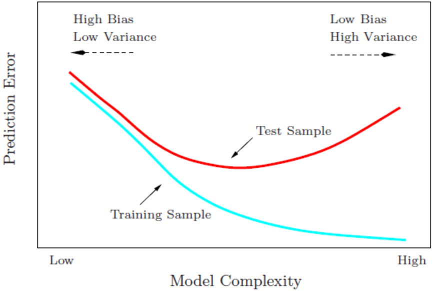
- A predictive model can be substantively more interesting
- Overfitting: Training error is small while test error is large. Model may be picking up patterns caused by random chance.

Machine Learning Basics

Bias-Variance Tradeoff

$$\text{Err}(x_0) = \text{Variance} + \text{Bias}^2 + \text{Irreducible Error}$$

- Bias: Error due to erroneous assumptions of model
- Variance: Error due to sensitivity to small changes in the training data

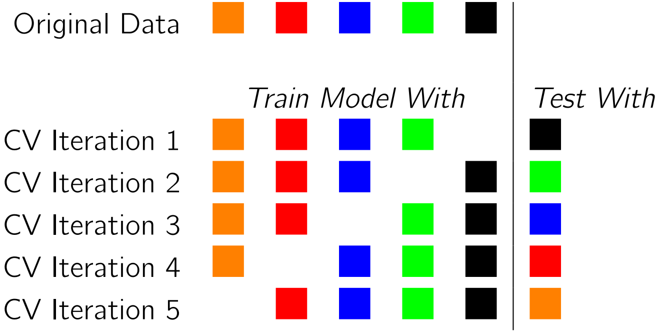


Source: Hastie *et al.*

Machine Learning Basics

k-Fold Cross Validation

5 fold cross validation:



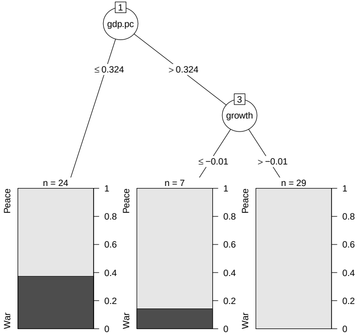
Classification and Regression Trees (CART)

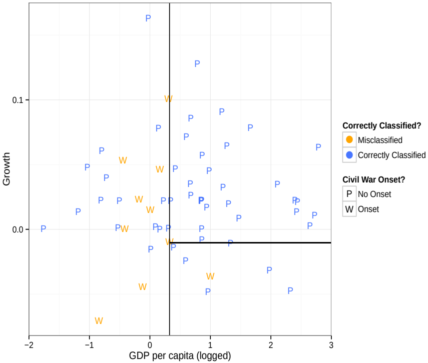
CART:

- Predictor space is partitioned into J non-overlapping $R_1, R_2, ..., R_J$ regions that are homogenous with respect to the response y
- Predictors and split points are chosen to minimize prediction errors
- Trees fit a prediction, a constant c_j, to each region R_j.

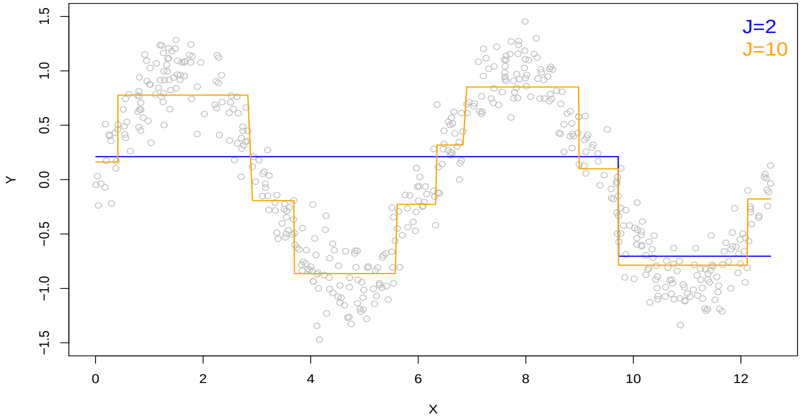
Classification and Regression Trees (CART)

Example: Civil War Onset and Economic Conditions





Classification and Regression Trees (CART)



Classification and Regression Trees (CART)

Advantages:

- Can easily handle mixed predictors
- Small trees are easy to interpret
- Can detect nonlinear relationships

Disadvantages:

- Can often have poor predictive performance

Ensemble Methods

Bagging (bootstrap aggregation):

- Draw a large number of bootstrapped samples
- Fit a tree to each bootstrapped sample
- Combine the predictions

Random Forests:

- Draw a large number of bootstrapped samples
- Fit a tree to each bootstrapped sample only considering a randomly selected subset of predictors at each split
- Combine the predictions

Ensemble Methods

- $y = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1$
- Ensemble with a majority vote:

Model	*Prediction*										*Accuracy*
Tree A	1	1	1	1	1	1	1	1	0	0	0.8
Tree B	0	1	1	1	0	1	1	1	0	1	0.7
Tree C	1	0	0	0	1	0	1	1	1	1	0.6
Ensemble	1	1	1	1	1	1	1	1	0	1	0.9

Boosting

Introduction

- Boosting also uses an ensemble of regression trees.
- Boosting works in a sequential manner where each tree tries to correct the errors of its predecessors.

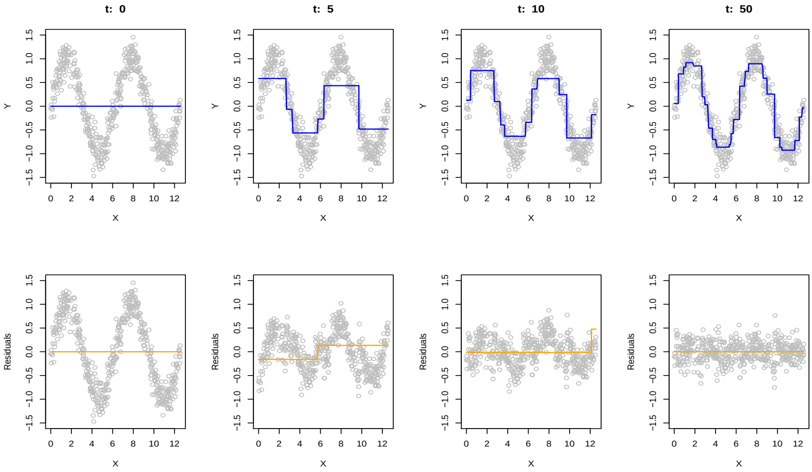
Boosting

Least Squares Boosting

1. Start with $f_0(x) = 0$, residuals $r = y$, $t = 0$
2. $t \leftarrow t + 1$
3. Fit a CART regression tree to r giving $g(x)$
4. Set $f_t(x) \leftarrow f_{t-1}(x) + g(x)$, $r \leftarrow r - g(x)$, and repeat steps 2-4 many times

Boosting

Intuition



Stochastic Gradient Boosting Algorithm

Select:

- A loss function (L)
- Number of trees (T)
- Number of regions in each tree (J)
- Shrinkage parameter (λ)
- Subsampling rate (p)

1. Initialize $f_0(x)$ to be a constant c, $f_0(x) = \text{argmin}_c \sum_{i=1}^{N} L(y_i, c)$
2. For $t = 1$ to T:
   (a) Randomly sample $p \times N = \tilde{N}$ cases from the data $(\left\{y_{\pi(i)}, x_{\pi(i)}\right\}_{i=1}^{\tilde{N}})$
   (b) For $i = 1, 2, ..., \tilde{N}$ compute
   $$r_{\pi(i)t} = -\left[\frac{\partial L(y_{\pi(i)}, f(x_{\pi(i)}))}{\partial f(x_{\pi(i)})}\right]_{f(x)=f_{t-1}(x)}$$
   (c) Fit a regression tree to the targets $r_{\pi(i)t}$ giving terminal regions R_{jt}, $b = 1, 2, ..., j_t$
   (d) Compute the optimal terminal node predictions
   $$c_{jt} = \underset{c}{\text{argmin}} \sum_{x_{\pi(i)} \in R_{jt}} L(y_{\pi(i)}, f_{t-1}(x_{\pi(i)}) + c)$$
   (e) Update $f_t(x) = f_{t-1}(x) + \lambda \cdot c_{jt} I(x \in R_{jt})$
3. Output $\hat{f}(x) = f_T(x)$

Response Types and Loss Functions

Boosting can be used to analyze continuous, categorical, count, and censored survival data.

Setting	Loss Function	$-\partial L(y_i, f(x_i))/\partial f(x_i)$
Regression	$\frac{1}{2}\left[y_i - f(x_i)\right]^2$	$y_i - f(x_i)$
Binary Classification	$-2\left[y_i f(x_i) - log(1 + exp(f(x_i)))\right]$	$y_i - p_i$
Count	$-2\left[y_i f(x_i) - exp(f(x_i))\right]$	$y_i - exp(f(x_i))$

Hyperparameters

- Shrinkage parameter (λ)- Slows down learning and helps to prevent overfitting. Increases computational costs.
- Number of trees (T)- Growing too many trees can lead to overfitting and poor predictive performance.
- Subsampling rate (p)- Introducing randomness into procedure can reduce the influence of individual observations and reduce computation time.
- Number of regions in each tree (J)- Restrict all trees to be the same size. Lower order interactions tend to perform best.

Use cross validation to select optimal hyperparameters

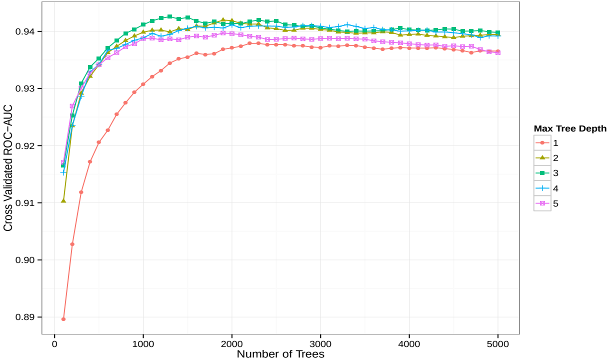
Application

Predicting Civil War Onset

- Response- Onset of Civil War (Yes/No)
- Predictors - 90 variables pertaining to the political institutions, development, natural resources, economic conditions, and demographic characteristics of a country

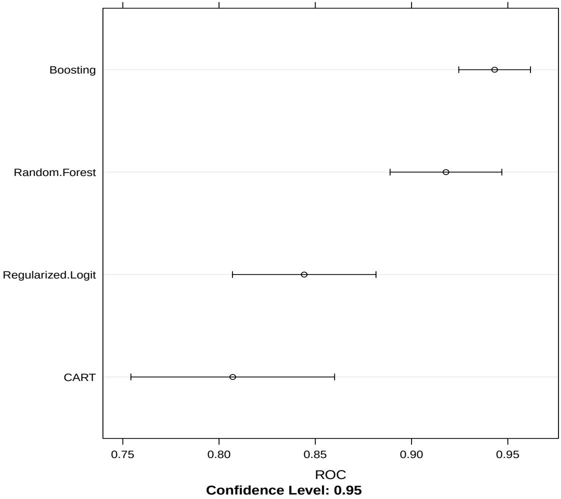
Hyperparameter Selection

- Subsampling rate $(p) = 0.5$, Shrinkage parameter $(\lambda) = 0.01$
- ROC-AUC maximized with Trees $(T) = 1300$, Regions$(J) = 7$



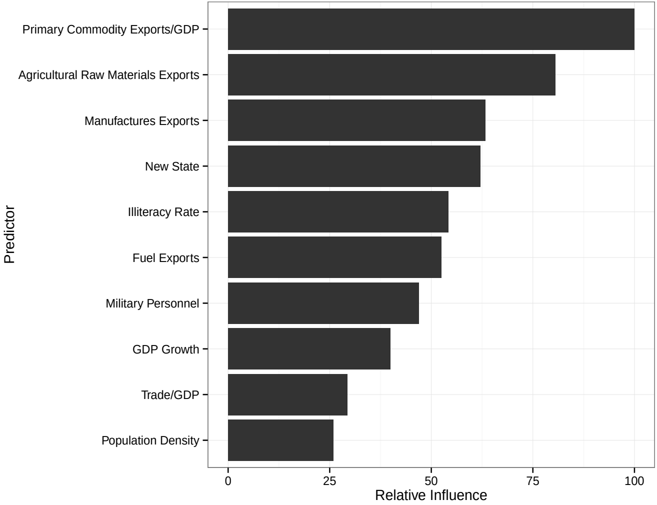
Prediction

Boosting versus Other Methods



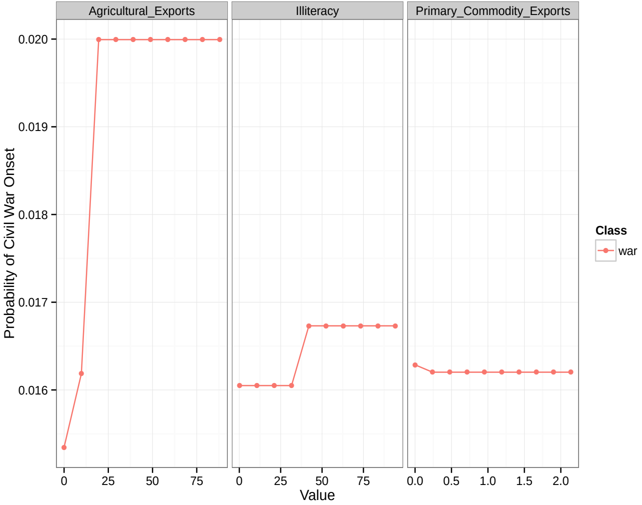
Exploration

Variable Importance



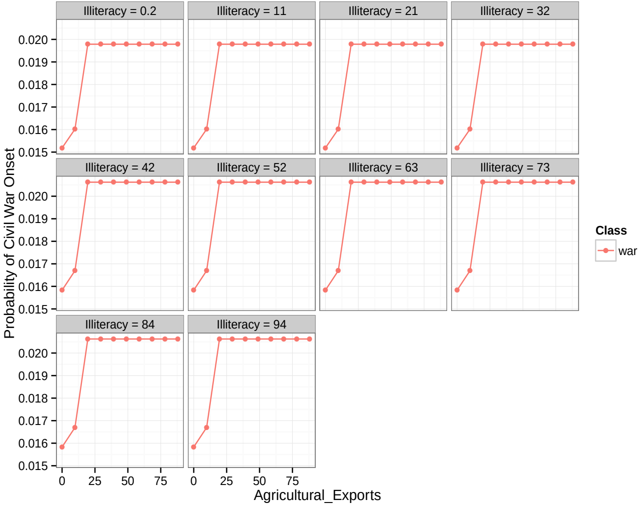
Exploration

Partial Dependence Plots



Exploration

Automatic Interaction Detection



Implementing Boosting

R

`gbm` package implements stochastic gradient boosting

- `gbm` for model training and prediction
- `plot.gbm` for partial dependence plots
- `relative.influence` for variable importance

`caret` acts as a wrapper for `gbm`

- `train` for easy model tuning and comparison
- `varImp` for variable importance

Implementing Boosting

R

`mlr` also acts as a wrapper for `gbm`

- `plotPartialPrediction` for partial dependence and interaction plots

`xgboost` implements a modified version of gradient boosting

- Extremely popular on kaggle
- Faster than `gbm` and can be used with extremely large datasets
- Accessible through both `caret` and `mlr`

Summary

- Boosting is a flexible technique that can be used with a variety of response types
- Boosting has strong predictive performance
- Boosting can help identify patterns in data