LSIRM Statistical/Machine Learning Regression Models with Selection - MARS and Polywog

Dave Armstrong

University of Western Ontario
Department of Political Science

e: dave.armstrong@uwo.ca
w: www.quantoid.net/teachwlu/

What do we Mean by ‘Model Selection’

- Testing competing models against each other (i.e., relative fit).
  - Nested model tests
  - Non-nested model tests
- Feature Selection
  - Which variables (features) of the data are important to predict the outcome?
  - Focus here is often on parsimony
- Multi-model inference
  - How to deal with model selection uncertainty in a principled way.

Options for Comparative Model Fit

- Direct tests of nested models - F (ANOVA), χ^2 (Analysis of Deviance, LR-Test)
- Information Criteria measures (e.g., AIC and BIC)
- Tests for Non-nested Models (e.g., Clarke and Vuong)

Nested Model Tests

Tests like the LR test and F-test require nested models because,

- They are considering the different between two statistics (RSS or LR)
- This difference follows an F or χ^2 distribution under the null (neither distribution permits negative values).
- So, the model with more parameters *must* provide a fit not worse than the model with fewer parameters.
  - The only way to ensure this is the case is to ensure that the models are nested

Likelihood Ratio Test

The LR Test uses the statistic defined by the difference in the log-likelihoods of the models.

$$LR = -2\left(ll_{\text{restricted}} - ll_{\text{unrestricted}}\right) \sim \chi^2_{p-q} \tag{1}$$

where there are p parameters in the unrestricted model and q parameters in the restricted model.

- The distribution is asymptotically right, but will not be exactly χ^2 in finite samples.
- Deviance is often taken as $-2ll_{\text{model}}$, though this is not always the case (take, for example, the linear model case).

Information Theory

- Information theorists believe in reality, but not in the notion of "true" models.
  - Models are necessarily simplified constructions that try to approximate reality.
- There is more information in large datasets than small.
  - Information amounts to the ability to identify interesting, though substantively small effects

Three Principles guiding Model-based Inference

1. Parsimony
   - Encapsulates the bias-variance tradeoff.
2. Multiple Working Hypotheses
   - There is no single null hypothesis against which an alternative is to be tested.
   - rather, there is a (small-ish) set, well-specified and theoretically derived working hypotheses.
3. Strength of Evidence
   - We must be able to quantify the "strength of evidence" supporting various working hypotheses if science is to progress in the usual way.

K-L Information

Kullback and Leibler (1951) quantified the meaning of "information".

$$I(f,g) = \int f(x) log\left(\frac{f(x)}{g(x|\theta)}\right) dx$$

where:

- f denotes a fixed (i.e., constant) reality (reality is non-parametric [i.e., it has no parameters])
- g is a model approximating f with parameters θ.
- $I(f,g)$ is the information lost when using g to approximate f.

There is no assumption that a true model exists (much less that the true model is in our candidate set of models) nor is there an assumption that the models are nested.

Expected Information

We cannot use $I(f,g)$ in model selection because it requires knowledge of f and θ the parameters in g.

$$\begin{aligned} I(f,g) &= E_f\left[log(f(x))\right] - E_f\left[log(g(x|\theta))\right] \\ &= C - E_f\left[log(g(x|\theta))\right] \end{aligned}$$

Estimating relative information for each model in the set results in our ability to compare across models (since C is constant for all model comparisons).

Akaike's Information Criterion (AIC)

The goal was to estimate: $E_y E_x\left[log(g(x|\hat{\theta}(y)))\right]$, essentially the relative information with θ replaced with the MLE estimates $\hat{\theta}$.

- Akaike found that $log(\mathcal{L}(\hat{\theta}|\text{data}))$ was a biased estimator of $E_y E_x\left[log(g(x|\hat{\theta}(y)))\right]$, but that asymptotically the bias is approximately equal to K, the number of parameters in $\hat{\theta}$. Thus,

$$log(\mathcal{L}(\hat{\theta}|\text{data})) - K = C - \hat{E}_{\hat{g}}\left[I(f,\hat{g})\right]$$

K is not arbitrary, but chosen to minimize bias in the estimated expected information.

$$\begin{aligned} AIC &= -2(log(\mathcal{L}(\hat{\theta}|\text{data})) - K) \\ &= -2log(\mathcal{L}(\hat{\theta}|\text{data})) + 2K \end{aligned}$$

Small-sample Correction

When K is large relative to n or for any value of K for small-n, there is a correction to AIC.

$$AIC_c = -2log(\mathcal{L}(\hat{\theta}|\text{data})) + 2K + \frac{2K(K+1)}{n-K-1}$$

- This should be used probably always, but especially if $n/K \leq 40$ for the largest K in the model set.
- AIC_c converges to AIC as $n \rightarrow \infty$.

Δ_i values

Often, for AIC_c or AIC to be interpretable, Δ_i should be calculated such that for each model i in the model set,

$$\Delta_i = AIC_i - AIC_{\text{min}}$$

This gives the "best" model $\Delta_i = 0$

- This captures the information loss due to using model g_i rather than the best model, g_{min}.
- The large Δ_i, the less likely model i is the best approximation of reality f.

Conventional cut-off values for Δ_i are:

- $\Delta_i \leq 2$ indicates substantial support,
- $4 \leq \Delta_i \leq 7$ indicates less support,
- $\Delta_i \geq 10$ indicates essentially no support.

BIC

The BIC is defined as:

$$BIC = -2\log(\mathcal{L}) + K\log(n)$$

- BIC is not technically based in "information theory" and as such is not an information criterion measure.
- The BIC is meant to approximate the Bayes Factor (or rather its log):

$$\frac{\Pr(D|M_1)}{\Pr(D|M_2)} = \frac{\int \Pr(\theta_1|M_1)\Pr(D|\theta_1, M_1)d\theta_1}{\int \Pr(\theta_2|M_2)\Pr(D|\theta_2, M_2)d\theta_2}$$

- Models need not be nested and we need not appeal to the idea that there exists a "true" model, much less that the true model is in our set of candidate models.

AIC or BIC

The question of whether to use AIC or BIC is often left to how much you want to penalize additional model parameters. In actuality, the question is one of performance in picking the K-L best model.

- When there are "tapering effects", AIC is better
- When reality is simple with a few big effects captured by the highest posterior probability models, then BIC is often better.

Likelihood-based Tests

There are a number of tests that are based on the Likelihoods of the two models.

- Vuong Test
- Clarke Test

Vuong Test

The Vuong test is a likelihood ratio test specified as follows:

$$\tilde{LR}_n(\hat{\beta}_n, \hat{\gamma}_n) = \log(\mathcal{L}_1) - \log(\mathcal{L}_2) - \frac{k_1 - k_2}{2}\log n$$

This statistic has a standard normal distribution under the null hypothesis that the two models are not different from each other.

Distribution Free Test

Clarke (2003) puts forth a distribution-free test that is really a "paired sign test". The statistic is calculated as:

$$d_i = \log(\mathcal{L}_{\beta, x_i}) - \log(\mathcal{L}_{\gamma, z_i}) + (p - q)\left(\frac{log(n)}{2n}\right)$$

$$B = \sum_{i=1}^{n} I_{0,+\infty}(d_i)$$

- The d_i are the difference in individual log-likelihoods for the two models
- The second equation above counts up the number of positive d_i values.
- We are testing to see whether B is significantly bigger than a random binomial variable that has a $p = .5$ and n the same as the number of rows in $\boldsymbol{X}$ and $\boldsymbol{Z}$.

Examples in R

You can produce AIC, AICc and BIC in the following ways:

```
library(car)
data(Prestige)
mod1 <- lm(prestige ~ income + women,
  data=na.omit(Prestige), y=T)
mod2 <- lm(prestige ~ education + type + women,
  data=na.omit(Prestige), y=T)
AIC(mod1)

## [1] 763.8879

library(AICcmodavg)
AICc(mod1)

## [1] 764.318

BIC(mod1)

## [1] 774.2278
```

Vuong and Clarke Tests in R

```
library(games)
vuong(mod1, mod2)

##
## Vuong test for non-nested models
##
## Model 1 log-likelihood: -378
## Model 2 log-likelihood: -336
## Observations: 98
## Test statistic: -3.6
##
## Model 2 is preferred (p = 0.00034)

clarke(mod1, mod2)

##
## Clarke test for non-nested models
##
## Model 1 log-likelihood: -378
## Model 2 log-likelihood: -336
## Observations: 98
## Test statistic: 24 (24%)
##
## Model 2 is preferred (p = 4.2e-07)
```

Shrinkage Estimators

Shrinkage estimators can reduce sampling variability and sometimes improve model fit (particularly in the presence of collinearity).

- Shrinkage estimators impose constraints on the fitted model (particularly on the size of the coefficients).
- The result of these constraints is to shrink the estimates toward zero.
- Ridge Regression and the LASSO are the two most prominent shrinkage estimators.

NB: these are *biased* estimators, so they might be good for stabilizing predictions, but they won't be particularly good for more conventional theory testing.

Ridge Regression

Ridge Regression minimizes the following function:

$$\sum_{i=1}^{N}\left(y_i - \beta_0 + \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

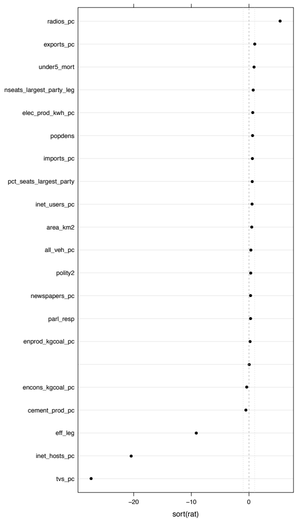
- λ is a tuning parameter that governs the relative impact on RSS and the penalty on the regression model.
- As $\lambda \rightarrow 0$, the estimates get increasingly close to the OLS estimates.
- As $\lambda \rightarrow \infty$, the estimates get increasingly close to zero.

The choice of λ is important and is often done with cross-validation.

CV with Ridge Regression

```
library(DAMisc)
library(parcor)
library(readstata13)
banks99 <- read.dta13(
  "http://quantoid.net/files/reg3/banks99.dta")
banks99s <- scaleDataFrame(banks99[,-c(1,2,4)])
X <- model.matrix(gdppc_mp ~. , data=banks99s)[,-1]
y <- model.response(model.frame(gdppc_mp ~. , data=banks99s))
rcv <- ridge.cv(X,y)
mod <- lm(y ~ X)
rat <- with(rcv, c(intercept, coefficients))/coef(mod)
names(rat) <- gsub("X", "", names(rat))
library(lattice)
dotplot(sort(rat), col="black")
trellis.focus("panel", 1, 1)
panel.abline(v=0, lty=2, col="gray65")
panel.abline(v=c(-1,1), lty=3, col="gray75")
trellis.unfocus()
```

Plot



LASSO (the L1 norm)

The LASSO (Least Absolute Shrinkage and Selection Operator) is another regularization method for estimating regression.

- Uses a different penalty than ridge regression:

$$\sum_{i=1}^{N}\left(y_i - \beta_0 + \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| \tag{2}$$

- Doesn't necessarily use all of the variables (i.e., some coefficients could be zero)
- Since not all variables are used in each fit, bootstrapping is more problematic here (though not impossible).

The LASSO in R

```
library(glmnet)
g <- glmnet(X, y)
cvg <- cv.glmnet(X,y)
round(cbind(coef(cvg), coef(mod)), 4)
```

```
## 21 x 2 sparse Matrix of class "dgCMatrix"
##                                1
## (Intercept)              0.0000  0.0000
## under5_mort              .       0.0214
## area_km2                 .       0.1365
## inet_hosts_pc            0.0558 -0.0032
## inet_users_pc            0.0956  0.1813
## enprod_kgcoal_pc         .       0.2801
## encons_kgcoal_pc         0.0457 -0.2730
## elec_prod_kwh_pc         .       0.1422
## cement_prod_pc           .       0.0073
## nseats_largest_party_leg 0.0018  0.1520
## eff_leg                  .      -0.0026
## pct_seats_largest_party  .       0.0250
## radios_pc                .       0.0140
## tvs_pc                   .      -0.0025
## newspapers_pc            .      -0.0930
## polity2                  .       0.0765
## parl_resp                .      -0.0853
## popdens                  .       0.0607
## imports_pc               0.1874  0.2825
## exports_pc               0.0714  0.1673
## all_veh_pc               0.4862  0.5060
```

Correlation of Predictions

```
tmp <- data.frame(
  ridge = c(cbind(1, X) %*% with(rcv,
    c(intercept, coefficients))),
  lasso = c(cbind(1, X) %*% as.matrix(coef(cvg))),
  ols = fitted(mod)
)
round(cor(tmp), 4)
```

```
##        ridge  lasso    ols
## ridge 1.0000 0.9862 0.9906
## lasso 0.9862 1.0000 0.9789
## ols   0.9906 0.9789 1.0000
```

Adaptive Lasso

The lasso gives all variables the same penalty (λ). The adaptive lasso relaxes this assumption by allowing each parameter to have a different weight:

$$\arg\min_{\boldsymbol{\beta}} \left\| y - \sum_{j-1}^{p} \boldsymbol{x}_j \beta_j \right\|^2 + \lambda \sum_{j=1}^{p} w_j |\beta_j|$$

Where we use results from an auxiliary regression (OLS, Ridge or LASSO) to make the weights:

$$\hat{w}_j = \frac{1}{|\hat{\beta}_j|^\gamma}$$

γ is not usually estimated, but values 0.5, 1, and 2 are tried to evaluate sensitivity. The only technical constraint is that $\gamma > 0$.

Oracle Property

The Adaptive Lasso has been shown to have the Oracle property, that the selection procedure asymptotically chooses the right model:

- True 0 coefficients are estimated as 0 with probability that tends toward 1
- True non-zero coefficients are estimated as if the true sub-model were known.

Steps for Adaptive LASSO

1. Estimate the initial coefficients via regression model (OLS, Ridge or LASSO).
2. Calculate the weights $w_j = \frac{1}{|\beta_j|^\gamma} \quad \gamma = \{0.5, 1, 2\}$.
3. Use the weights as input to the LASSO routine.

Adaptive LASSO example

```
# estimate initial ridge regression and save coefficients
b.ridge <- coef(ridge.cv(X,y))
# calculate weights
gamma <- 1
w <- 1/(abs(b.ridge)^gamma)
# estimate the LASSO with the weights
cvg <- cv.glmnet(X,y, penalty.factor=w)
coef(cvg)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                                       1
## (Intercept)              -4.321919e-17
## under5_mort                .
## area_km2                   .
## inet_hosts_pc              .
## inet_users_pc              .
## enprod_kgcoal_pc           .
## encons_kgcoal_pc           .
## elec_prod_kwh_pc           .
## cement_prod_pc             .
## nseats_largest_party_leg   .
## eff_leg                    .
## pct_seats_largest_party    .
## radios_pc                  .
## tvs_pc                     .
## newspapers_pc              .
## polity2                    .
## parl_resp                  .
## popdens                    .
## imports_pc                 2.078042e-01
## exports_pc                 2.980630e-03
## all_veh_pc                 5.881630e-01
```

Four Most Important Variables

Variable	Ridge	Lasso	Adaptive Lasso
Internet Users/capita	0.100	0.111	0.000
Energy Consumption	0.099	0.024	0.000
Imports/capita	0.169	0.199	0.201
Vehicles/capita	0.172	0.485	0.556

Inference After Selection

Inference gets much more complicated after model selection, given that variables are often selected *because* they are significant predictors. There are a few options for post-selection inference.

- Data Splitting - Split the sample into two halves - select on one set, test on the other. Most conserative (loss of power due to lower N).
- Data Carving - A small proportion of the sample is witheld from training and then the entire sample is used for testing Fithian, Sun and Taylor (2014).
- Exact post-selection inference possible for Forward Selection Regression and LASSO with fixed λ (Tibshirani et al. 2014, SelectiveInferecen package in R).
- Valid post-selection inference for Linear LS Models (Berk et al. 2013, implemented in the PoSI package in R).

Variable Selection Methods: Cautions (1)

- If we have a very large number of predictors and we simply want a parsimonious predictive model, subset methods and the lasso could be really useful.
- When tackling collinearity, however, variable selection may results in a re-specified model that does not address the original research question (ridge regression could help).
    - If the original model is correctly specified, then coefficient estimates following variable selection are *biased*. However, the bias may not be overwhelming if you started off with a severe collinearity problem

Variable Selection Methods: Cautions (2)

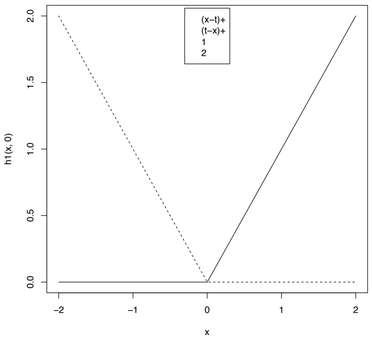
- If our goal is to assess the individual predictors (or their relative impacts), variable selection models have serious implications
    - Standard errors calculated following variable selection overstate the precision of results - they do not control for relevant predictors and they do not account for model selection uertainty.
    - A new sample may give different results, leading to inconsistent interpretation of "effects"
- These models, again, are really about *prediction* not hypothesis testing, though the can still be quite valuable.

Multivariate Adaptive Regression Splines (MARS)

The main component of MARS is a pair of piecewise linear (hinge) splines.

$$(x-t)_+ = \begin{cases} x-t & \text{if } x > t \\ 0 & \text{otherwise.} \end{cases}$$

$$(t-x)_+ = \begin{cases} t-x & \text{if } x < t \\ 0 & \text{otherwise.} \end{cases}$$



MARS Notation

MARS takes the form:

$$f(x) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(x)$$

where h_m is the pair of hinge functions.
Computationally:

1. Forward pass - add pairs of hinge functions by reduction in SSRes until all pairs are in.
2. Backward pass - take individual functions out by min increase in SSRes until GCV criterion is satisfied.

Interactions

- The `degree` parameter in the R algorithm controls the degree of interaction you want to allow.
  - This can make the model really complicated because it's expanding all possible interactions among hinge functions and then pulling them out on the backward pass step.
  - This model is more easily constrained (particular w.r.t additivity) than the other models we talked about before.
- You can also identify variables that will enter the model linearly *if they enter the model at all* .

MARS in R

The MARS algorithm is licensed by Salford Systems, so to avoid trademark infringements, other implementations of the MARS algorithm are called “Earth”.

```
set.seed(11)
n  = 200
p = 5
X = data.frame(matrix(runif(n * p), ncol = p))
y = 10 * sin(pi* X[ ,1] * X[,2]) +20 *
  (X[,3] -.5)^2 + 10 * X[ ,4] + 5 * X[,5] + rnorm(n)
df <- as.data.frame(cbind(X,y))
library(earth)
e1 <- earth(X,y, nfold=10, ncross=10, pmethod="cv", degree=2)
```

Earth Summary

```
summary(e1)


## Call: earth(x=X, y=y, pmethod="cv", degree=2, nfold=10, ncross=10)
##
##                                  coefficients
## (Intercept)                         20.522999
## h(0.502856-X1)                     -20.213453
## h(X1-0.502856)                      23.381319
## h(0.761508-X2)                     -19.661064
## h(X2-0.761508)                       6.144417
## h(0.40403-X3)                       12.491871
## h(X3-0.40403)                        3.818608
## h(X3-0.799209)                      11.158226
## h(0.932184-X4)                     -10.570805
## h(0.218507-X5)                      -6.047004
## h(X5-0.218507)                       5.190464
## h(X1-0.502856) * h(X2-0.419717)   -78.008484
## h(0.764608-X1) * h(0.761508-X2)    25.825215
## h(X1-0.764608) * h(0.761508-X2)   -42.362823
## h(X3-0.48401) * h(0.932184-X4)      4.023197
##
## Selected 15 of 19 terms, and 5 of 5 predictors using pmethod="cv"
## Termination condition: Reached nk 21
## Importance: X4, X1, X2, X3, X5
## Number of terms at each degree of interaction: 1 10 4
## GRSq 0.9389959  RSq 0.9585675  mean.oof.RSq 0.9316501 (sd 0.0258)
##
## pmethod="backward" would have selected the same model:
##     15 terms 5 preds,  GRSq 0.9389959  RSq 0.9585675  mean.oof.RSq 0.9316501
```

Visualizing Partial Effects: Partial Dependence Plot

The PDP plots the change in the average predicted value for a subset of features S, averaged over the subset of features C, where C is the complement of S. Formally:

$$f_S = \mathbb{E}_{x_C}\left[f(\boldsymbol{x}_S, \boldsymbol{x}_C)\right] = \int f(\boldsymbol{x}_S, \boldsymbol{x}_C) dP(\boldsymbol{x}_C)$$

In words: we are predicting $f()$ with the variables in S averaged over all of the variables in C.

Visualizing Partial Effects: Individual Conditional Expectation Plots

ICE disaggregates the PDP.

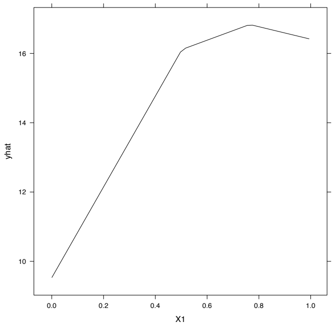
- The PDP is obtained by averaging over all of the ICE curves.
- Plots N different curves to enable evaluation of effect heterogeneity.
- Heterogeneity essentially means interactions with variables in C.

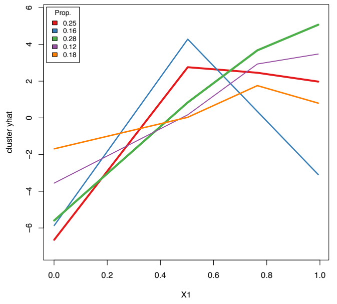
$$f_{S_i} = \mathbb{E}_{x_{C_i}}\left[f(\boldsymbol{x}_S, \boldsymbol{x}_{C_i})\right]$$

Dependence Plots

```
library(RColorBrewer)
cols <- brewer.pal(5, "Set1")
library(pdp)
ep1 <- partial(e1, train=X, pred.var="X1")
plotPartial(ep1)
```

```
library(ICEbox)
ep2 <- ice(e1, X=X, y=y, predictor="X1")
clusterICE(ep2, nClusters=5, plot_legend=TRUE,
  colorvec=cols)
```





Variance Models

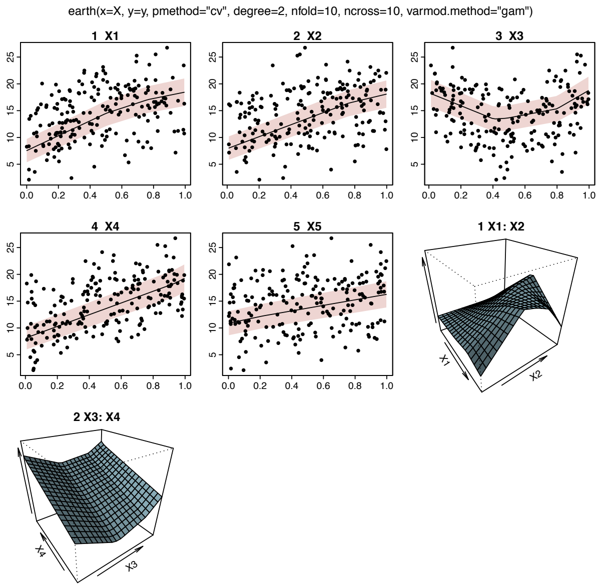
- You can't get confidence intervals from these models because they don't take into account the selection mechanism.
  - MARS picks values essentially because they are good predictors, so the items in the model will necessarily have small p-values.
- You can get prediction intervals for the - essentially the variability in future observations predicted by the model.
  - The `varmod.method` allows you to model the residual variance by modeling the absolute value of the residuals as a function of the fitted values.
- Prediction variance is:

$$\epsilon^2_{i,future} = \frac{(y_i - \hat{y}_i)^2}{(1 - h_{ii})} + \texttt{modvar}_i$$

Prediction Variances in earth

```
library(mgcv)
e2 <- earth(X,y, nfold=10, ncross=10, pmethod="cv",
  degree=2, varmod.meth="gam")
plotmo(e2, pt.col=1, level=.95)
```

```
##  plotmo grid:    X1        X2        X3        X4       X5
##             0.4392563 0.5140201 0.4955242 0.5069235 0.489479
```



Polywog

Polywog is a method developed by Kenkel and Signorino which puts two pieces we've already considered together:

- Polynomial expansion: If the degree = 3 and we have variablex $\{x_1, x_2\}$ in our model, then the following terms would be included in the expansion: $x_1, x_2, x_1^2, x_2^2, x_1^3, x_2^3, x_1x_2, x_1^2x_2, x_2^2x_1$.
- Adaptive Lasso: We use the adaptive LASSO to figure out which of the polynomial expansion terms to keep in the model.

Polywog Example

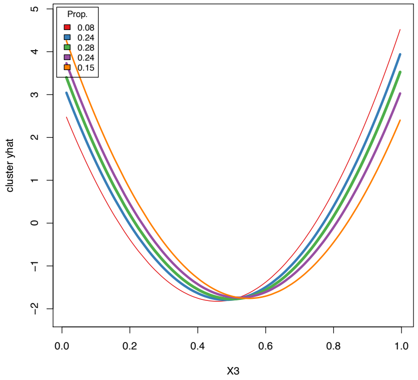
```
library(polywog)

p1 <- polywog(y ~ ., data=df)
sort(coef(p1)[-which(coef(p1) == 0)])


##     X1^2.X2     X1.X2^2          X3          X1          X2        X1^3
## -47.0104592 -40.0255071 -24.9238963 -17.1414326 -10.3280139  -8.1871101
##    X1.X3.X5     X2.X5^2    X2.X4.X5     X4^2.X5     X3^2.X4     X3.X4^2
##  -7.3665685  -4.3513901  -3.4855783  -2.7964654  -2.1900846  -0.7751271
##       X1.X4     X3^2.X5        X5^3       X1.X5       X3.X4    X1.X4.X5
##  -0.3598642  -0.2350903   0.9536742   1.2582594   1.6079199   1.7908432
##       X3.X5        X5^2       X4.X5       X2.X5    X2.X3.X5       X1.X3
##   2.7541044   2.9716099   3.4330804   3.9552448   4.0553543   4.2489761
##        X2^2 (Intercept)          X4        X3^2        X1^2       X1.X2
##   8.1053866   8.9033528  10.6006623  22.7082379  23.4724665  87.4183701
```

ICEPlot for Polywog

```
pice <- ice(p1, X, y, predictor="X3")
clusterICE(pice, nClusters=5, plot_legend=T, colorvec=cols)
```

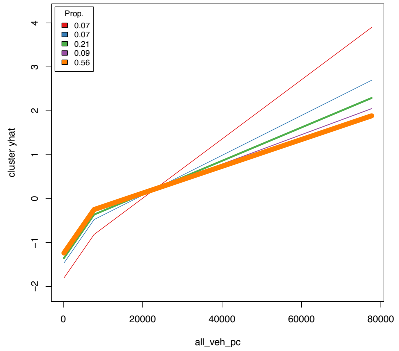


```
library(readstata13)
banks <- read.dta13("http://quantoid.net/files/reg3/banks99.dta")
banks.dat <- banks[,-c(1,2,4,5)]
banks.X <- model.matrix(gdppc_mp ~ ., data=banks.dat)[,-1]
banks.y <- log(model.response(model.frame(gdppc_mp~ ., data=banks.dat)))
e3 <- earth(banks.X, banks.y, nfold=10, ncross=5,
   degree=2, pmethod="cv")
summary(e3)
```

```
## Call: earth(x=banks.X, y=banks.y, pmethod="cv", degree=2, nfold=10,
##             ncross=5)
##
##                                  coefficients
## (Intercept)                         7.6896487
## h(3902-cement_prod_pc)             -0.0003636
## h(7751-all_veh_pc)                 -0.0001305
## h(all_veh_pc-7751)                  0.0000299
## exports_pc * h(all_veh_pc-7751)     0.0000000
##
## Selected 5 of 16 terms, and 3 of 19 predictors using pmethod="cv"
## Termination condition: GRSq -Inf at 16 terms
## Importance: all_veh_pc, cement_prod_pc, exports_pc, ...
## Number of terms at each degree of interaction: 1 3 1
## GRSq 0.9227393  RSq 0.9551502  mean.oof.RSq 0.7672082 (sd 0.41)
##
## pmethod="backward" would have selected:
##     8 terms 7 preds,  GRSq 0.9337144  RSq 0.9774445  mean.oof.RSq 0.4573622
```

ICE Plot

```
i3 <- ice(e3, X=banks.X, y=banks.y, predictor="all_veh_pc")
clusterICE(i3, nClusters=5, plot_legend=TRUE,
  colorvec=cols)
```



Example: GDP Data

What do we want to know?

- Earlier we saw that polity2 had a quadratic relationship with one *log*(gdp/capita).
- Is that “robust”? Does the additive, quadratic form really represent that relationship well?
- We can use the tools we developed today to figure that out.
    - Note, we are not using these tools to their greatest advantage because we have small data (both in n and k).
    - Puts us in a less good position than we might otherwise be regarding inference. In truly BIG data, inference is unnecessary (everything would be significant).

Models

```
library(earth)
library(polywog)
library(foreign)
dat <- read.dta("http://quantoid.net/files/reg3/gdp_data_2000.dta")
Xm <- model.matrix(log(rgdpna_pc) ~ ., data=dat)[,-1]
X <- as.data.frame(Xm)
y <- model.response(model.frame(log(rgdpna_pc) ~ ., data=dat))
m5 <- earth(log(rgdpna_pc) ~ ., data=dat, pmethod="cv", ncross=10, nfold=10, degree=3)
m6 <- polywog(log(rgdpna_pc) ~ primsch_enroll_pc + polity2 +
  pop_c100k_pc, data=dat, degree=3)
```

In-sample Predictive Accuracy

```
preds <- cbind(
  c(predict(m5, newdata=X)),
  predict(m6, newdata=X)
)
colnames(preds) <- c("MARS", "PWOG")
cor(preds, y)^2
```

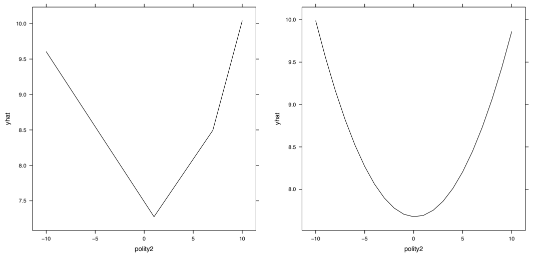
```
##           [,1]
## MARS 0.6970951
## PWOG 0.6011428
```

Partial Dependence Plots

```
plotPartial(partial(m5, train=X, pred.var="polity2"))
```

```
plotPartial(partial(m6,  X=X, pred.var="polity2",
  type="regression"))
```

Plots

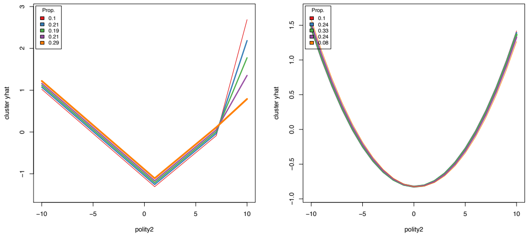


ICE Plots

```
clusterICE(ice(m5, X=X, y=y, predictor="polity2"),
  plot_legend=T, colorvec=cols, nClusters=5)
```

```
clusterICE(ice(m6, X=X, y=y, predictor="polity2"),
  plot_legend=T, colorvec=cols, nClusters=5)
```

Plots

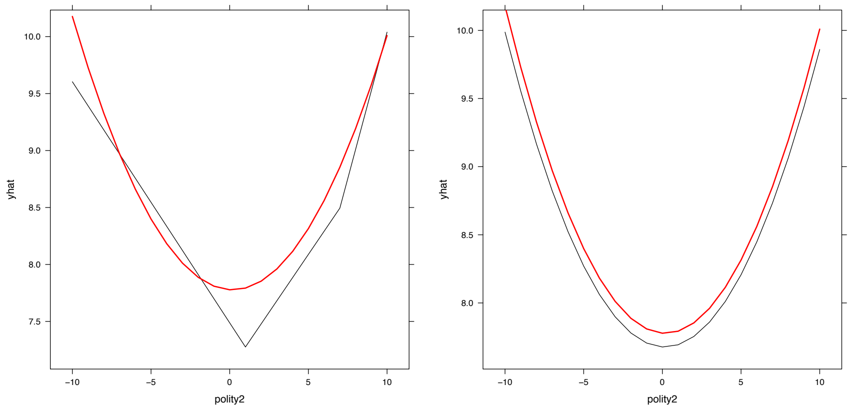


Compatability with Quadratic Form

```r
library(splines)
library(effects)
lm.mod <- lm(log(rgdpna_pc) ~ poly(polity2, 2, raw=TRUE) +
  bs(pop_c100k_pc, df=8) + primsch_enroll_pc, data=dat)
eff <- effect("poly(polity2, 2, raw=TRUE)", lm.mod, xlevels=21)
plotPartial(partial(m5, train=X, pred.var="polity2"))
trellis.focus("panel", 1, 1)
panel.lines(eff$x$polity2, eff$fit, col="red", lwd=2)
trellis.unfocus()
```

```r
plotPartial(partial(m6,  X=X, pred.var="polity2",
  type="regression"))
trellis.focus("panel", 1, 1)
panel.lines(eff$x$polity2, eff$fit, col="red", lwd=2)
trellis.unfocus()
```

Plots



Berk, Richard, Lawrence Brown, Andreas Buja, Kai Zhang and Linda Zhao. 2013. “Valid Post-selection Inference.” *The Annals of Statistics* 41:802–837.
URL: *https://www.jstor.org/stable/23566582*

Fithian, William, Dennis Sun and Jonathan Taylor. 2014. “Optimal Inference After Model Selection.”.
URL: *http://arxiv.org/abs/1410.2597*

Tibshirani, Ryan J., Jonathan Taylor, Richard Lockhart and Robert Tibshirani. 2014. “Exact Post-Selection Inference for Sequential Regression Procedures.”.
URL: *http://arxiv.org/abs/1401.3889*