Statistical Learning Methods using R

Kelly McConville

Assistant Professor of Statistics
Swarthmore College

2-15-2018

Course Set-up

- If you want to follow along in your own RStudio session, please install the following packages:
  - `dplyr`, `ggplot2`, `knitr`, `glmnet`, `broom`, `caret`, `GGally`, `readr`, `splines`, `SemiPar`, `kernlab`, `rpart`, `partykit`

Goals of this Course

- Learn some statistical learning techniques.
- Further develop our R skills.
  - I am assuming at least a basic proficiency in R.
- Course format: For each topic,
  - Introductory lecture.
  - R demonstration.
- Feel free to ask questions at any time.
  - Because of our size and time constraints, some questions may need to be pushed to the break period or to later in the conference.
- Great resource: Introduction to Statistical Learning in R
  - James et al. (2013)

Outline of Topics

- **Reproducible Workflow**
  - RMarkdown
- **Penalized Linear Regression**
  - Cross-validation
  - Lasso, Ridge, Elastic Net
- **Non-parametric Regression**
  - Regression Splines
  - Smoothing Splines
- **Classification**
  - Logistic Regression
  - Logistic Elastic Net
  - Support Vector Machines
  - Trees

R Markdown: Quick intro

- What is it?
  - File format (Rmd) for creating reports/slides/web pages in R.
  - Uses markdown (a easy to use text format).
  - Can insert R code, in *R chunks* directly into the document.
  - Can insert Latex syntax too: μ.
- Why use it?
  - Allows for a fully reproducible workflow.
    - I can share my data, Rmd file, and supplementary files (css, bibliography images) with you and you will be able to reproduce my work.
  - Easy to update figures and tables.
  - Can compile, *knit*, document into many different formats.
    - Example: These are beamer slides.
- RMarkdown Resources

A Few Comments on the R Code in this Course

- `tidyverse` versus base R?
  - I will mostly use the `tidyverse`.
  - Data import: `readr`
  - Data viz: `ggplot2`
  - Data wrangling: `dplyr`
- For model fitting, there are often several competing packages.
  - It is a good idea to try several and see if you prefer the functionality of one over another.
  - I will use the packages I know. (But people are always posting new ones to the CRAN!)
  - We will use `caret` which allows you to fit many different predictive models with the same syntax.

Data: Consumer Expenditure Survey

- Will use the fourth quarter of the 2016 BLS Consumer Expenditure Interview Survey throughout the course for examples.
  - Public Micro Data
- CE uses a multistage design.
  - We are ignoring that structure here!
  - Not much variability in the sampling weights.
- Data Description file on course page.

Statistical Models

What is the role of statistical modeling?

> *The only useful function of a statistician is to make predictions. – Wallis (1980)*

> *The prediction of observables or potential observables is of much greater relevance than the estimate of what are often artificial constructs-parameters. – Geisser (1975)*

> *The two goals in analyzing data... I prefer to describe as "management" and "science". Management seeks profit... Science seeks truth. – Parzen (2001)*

> *[A question of] whether observed data should be used only to shed light on existing theories or also for the purpose of hypothesis seeking in order to develop new theories. – Feelders (2002)*

- Pulled quotes from a wonderful paper: "To explain or to predict?" (Shmueli 2010)

Statistical Modeling Goals

- So, is the goal to explain or predict?
- Goals will impact the form of the models you consider.
  - Most traditional statistical modeling courses focus on explanatory models.
- Statistical learning models tend to be used more often when the goal is prediction.
  - Therefore, we will focus more on prediction accuracy than on explanatory ability when comparing models.
  - But, building a flexible, data-driven model can help uncover new causal mechanisms, which will help you build better explanatory models!

Set-up

- Assume we have *tidy* data:
  - Each row is an observation.
  - Each column is a variable.
    - Variable of interest: Y
    - Predictor variables: X's
- Model:

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

- Want to estimate $f(\cdot)$ well.
  - Over the course of the next couple of hours, we will make assumptions about the form of $f(\cdot)$.

Form of the Model

- Parametric: Specify a specific form
  - EXs:

$$f(X) = \beta_o + \beta_1 X$$

$$f(X) = \beta_o + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2$$

  - Pros:
    - Need less data to accurately estimate $f(\cdot)$ because we have less parameters to estimate in the model.
    - Usually more interpretable.
    - Easy to conduct tests of significance.
  - Cons:
    - Makes strong assumptions about the form of the model
    - If true model is far from the assumed model, all results are fairly suspect.

Form of the Model

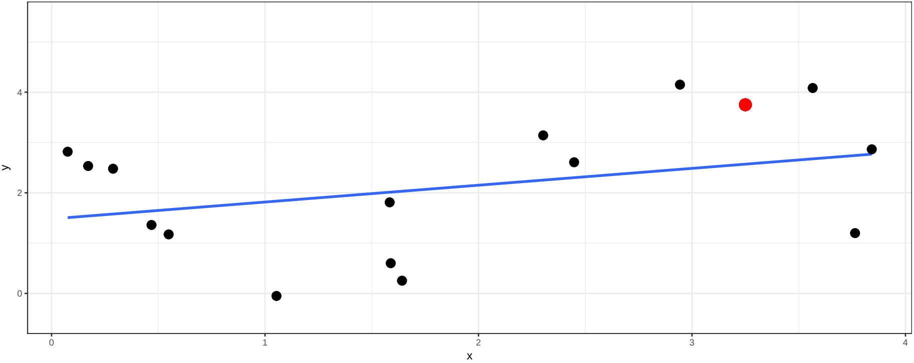
- Non-parametric: Specify a broad class of functions
  - EX:

$$f(X) = \text{All twice differentiable functions}$$

  - Data-driven: Get as close to the data as possible while still producing a smooth function.
  - Harder to explain the relationship between the predictors and the response
- Parametric methods will usually out-perform non-parametric methods when n/p is small.

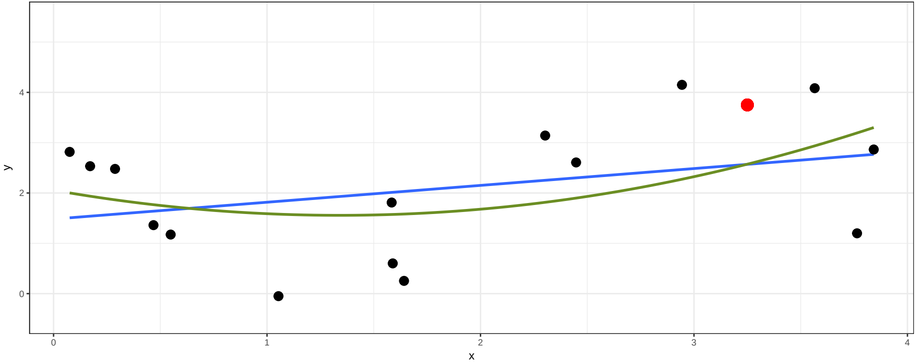
Form of the Model

- For both parametric and non-parametric models, one needs to balance
  - Fitting the data well
  - Generalizability to new data



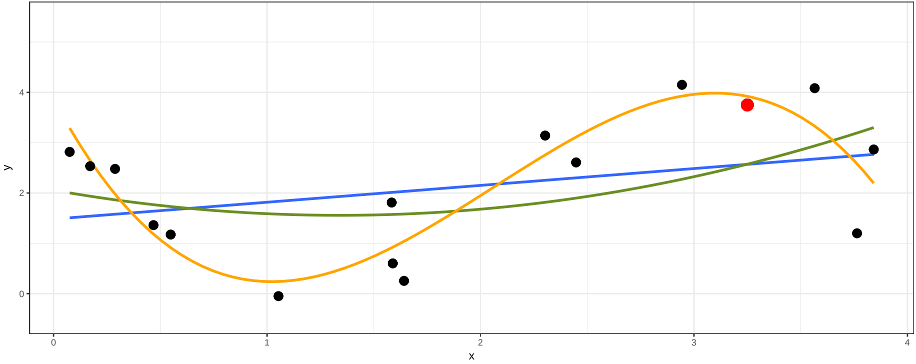
Form of the Model

- For both parametric and non-parametric models, one needs to balance
  - Fitting the data well
  - Generalizability to new data!



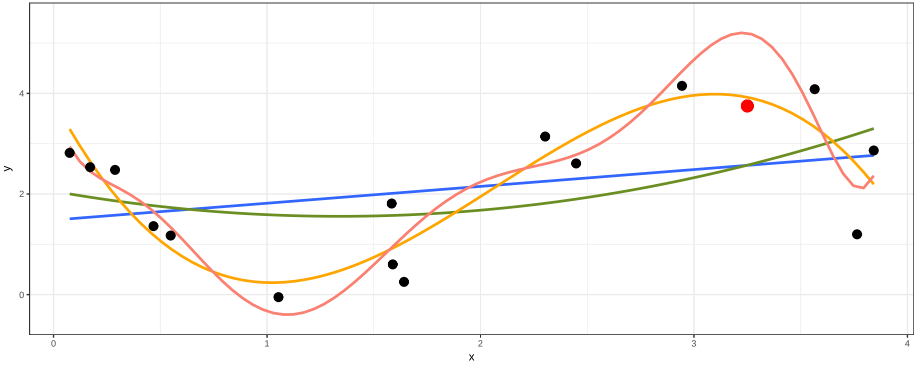
Form of the Model

- For both parametric and non-parametric models, one needs to balance
  - Fitting the data well
  - Generalizability to new data!



Form of the Model

- For both parametric and non-parametric models, one needs to balance
  - Fitting the data well
  - Generalizability to new data!



Model Selection

- For now assume Y is quantitative.
- **Question**: Suppose I have a set of models I am considering. How do I decide which one is the best?
- (One) **Answer**: Pick the model that is best at predicting new observations.
- **Question** How can we measure prediction accuracy?
- **Answer**:

$$\text{Test MSE} = \frac{1}{n_o} \sum_{i=1}^{n_o} (Y_i - \hat{y}_i)^2$$

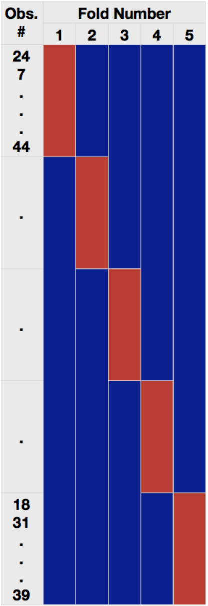
where

- n_o is the number of new observations
- Y_i is the i-th new observation for variable Y
- $\hat{y}_i$ is the predicted value for observation i based on the model fit to the original data.
- Pick the model with the smallest Test MSE.

Model Selection

- **Question**: How do we get "new" observations?
- **Answer**: Before fitting the model, split the data into two sets.
  - Training data: Fit model on this data.
  - Test data: Use estimated model to compute Test MSE.
- **Question**: What is a good partition? 50:50?
- **Answer**: It depends...
  - Need enough training data to get accurate parameter estimates.
  - Need enough test data to accurately estimate the Test MSE.
  - Common split: 80:20
  - Could run simulations to see how variability the parameters and Test MSE are under different partitions.

Cross-Validation



- Need very large dataset to justify partitioning since you are losing information.
- For smaller datasets use *k*-fold cross validation to estimate the Test MSE.
- **Idea of CV**: Create *k* training and test datasets. For each training dataset, fit the model and predict the observations in the test.

Cross-Validation

- Randomly partition data in k equally sized test datasets = folds.
- For each fold,
  - Fit model on the data not in the fold (training data).
  - Compute $\hat{y}_i$ for each i in the fold.
- Compute the Test MSE.
- Common $k = 10, n$.
- Leave one out cross-validation
  - $k = n$
  - For least squares regression,

$$\text{test MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{y}_i}{1 - h_{ii}}\right)^2$$

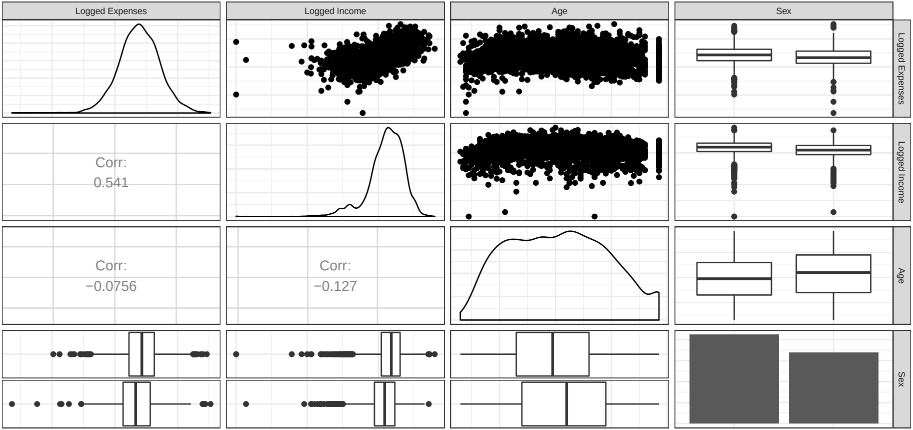
where h_{ii}, the i-th leverage value, is the i-th diagonal element of the Hat matrix:

$$\boldsymbol{H} = \left(\boldsymbol{X^t X}\right)^{-1}\boldsymbol{X^t Y}.$$

Example:

- Y = logged expenses
- X_1 = logged income, X_2 = age, X_3 = sex of principal earner

(Only looked at households with expenses and income)



Example

- Let's compare these three models:

$$f_1(X) = \beta_o + \beta_1 X_1 + \beta_2 X_1^2$$

$$f_2(X) = \beta_o + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3$$

$$f_3(X) = \beta_o + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_1 X_2 + \beta_5 X_2 X_3 + \beta_6 X_1 X_2 X_3$$

Example

```r
#Only use data without missing values or -inf
dat <-  ce %>%
  filter(TOTEXPCQ > 0, FINCBTAX > 0) %>%
  select(logTOTEXPCQ, logFINCBTAX, AGE, SEX)


#Fit models
mod1 <- lm(logTOTEXPCQ ~ poly(logFINCBTAX, degree = 2),
           data = dat)
mod2 <- lm(logTOTEXPCQ ~ logFINCBTAX + AGE + SEX, data = dat)
mod3 <- lm(logTOTEXPCQ ~ logFINCBTAX*SEX*AGE, data = dat)
```

Example

```r
# Predict
pred1 <- predict(object = mod1)
pred2 <- predict(object = mod2)
pred3 <- predict(object = mod3)
# Leverage values
h1 <- hat(model.matrix(mod1))
h2 <- hat(model.matrix(mod2))
h3 <- hat(model.matrix(mod3))
# Test MSE from LOOCV
test_mse1 <- mean(((dat$logTOTEXPCQ - pred1)/(1 - h1))^2)
test_mse2 <- mean(((dat$logTOTEXPCQ - pred2)/(1 - h2))^2)
test_mse3 <- mean(((dat$logTOTEXPCQ - pred3)/(1 - h3))^2)
```

Example

Table 1: Ratio of Test MSE to MSE of Model 1

Model 1	Model 2	Model 3
1	1.095692	1.097225

- Pick Model 1.
  - How might we do this differently if our focus was on finding the best explanatory model?
- If I have lots of models to choose from, I won't build all possible models and estimate the test MSE via cross-validation.
  - Time to learn the Lasso.

Model Selection

- Set-up:
  - Have variable of interest: Y
  - Have several possible predictor variables: $X_1, X_2, \ldots, X_p$.
  - Have n observations.
    - Possibly that $n < p$.
- Want to build a **linear regression** model:

$$
\begin{aligned}
Y =& \beta_o + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \epsilon \\
=& \boldsymbol{X}^t \boldsymbol{\beta} + \epsilon
\end{aligned}
$$

where $\boldsymbol{X} = (1, X_1, X_2, \ldots, X_p)^t$ and $\boldsymbol{\beta} = (\beta_o, \beta_1, \beta_2, \ldots, \beta_p)^t$

Model Selection

- All subsets: Fit all models and pick best based on some criteria.
  - Too difficult when p is large.
  - Can't use least squares to fit models when $p > n$.
- Step-wise: Sequentially add (or drop) predictors based on which adds the most value to the model.
  - Can end up with a local max, not a global max (i.e. miss the BEST model).

Least Absolute Shrinkage and Selection Operator

- Introduced by Tibshirani (1996) for a non-survey context.
  - McConville et al. (2017) extended to account for survey weights.
- Assume the linear regression working model but also assume some of the variables should not be in the model.
- Estimator for $\boldsymbol{\beta}$:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{j \in s} (Y_j - \boldsymbol{X}_j^T \beta)^2 + \lambda \sum_{i=1}^{p} |\beta_i|$$

for some positive constant λ.

- Some coefficient estimates will be exactly zero.
- Estimate λ with cross-validation.

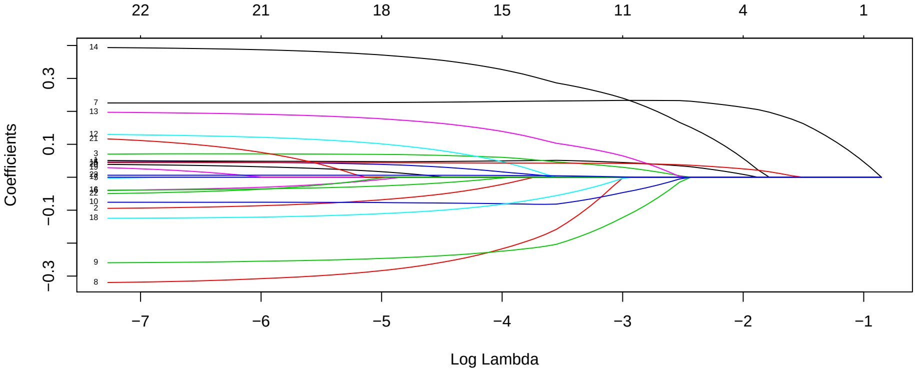
Lasso Example

```r
# Pick Y and X's
ds <- filter(ce, TOTEXPCQ > 0, FINCBTAX > 0) %>%
  dplyr::select(logTOTEXPCQ, FAM_SIZE, BLS_URBN, AS_COMP1,
                AS_COMP2, AS_COMP4, AS_COMP5, logFINCBTAX,
                HIGH_EDU, ROOMSQ, SEX, AGE,
                MEMBRACE, BEDROOMQ) %>%
  # Remove missing values
  na.omit()
```

```r
# Create design matrix
x <- model.matrix(logTOTEXPCQ ~ ., data = ds)[, -1]
# Define response variable
y <- ds$logTOTEXPCQ
# Fit Lasso model
fit <- glmnet(x, y, alpha = 1, standardize = TRUE,
    nlambda = 100)
```

Lasso Example

```r
# Look at coefficient paths
plot(fit, xvar = "lambda", label = TRUE)
```



- Which λ value produces the best model?

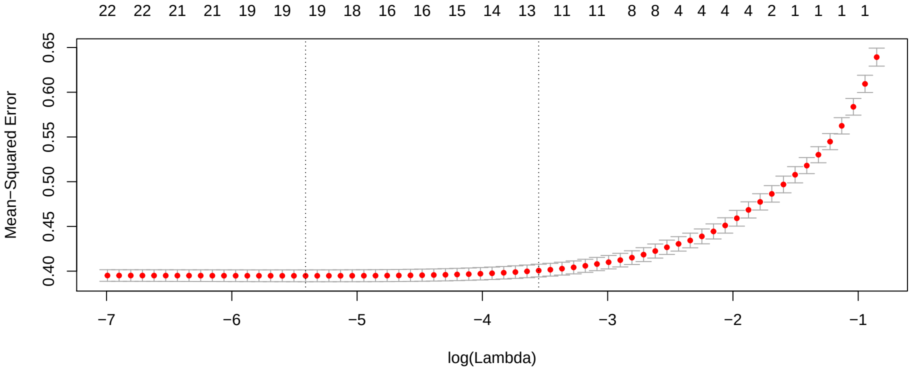
Lasso Example

```r
# Use cross-validation to pick lambda
cvfits <- cv.glmnet(x, y, alpha = 1, standardize = TRUE,
    nfolds = 10)
glance(cvfits)
```

```
##   lambda.min lambda.1se
## 1 0.00446644  0.0287106
```

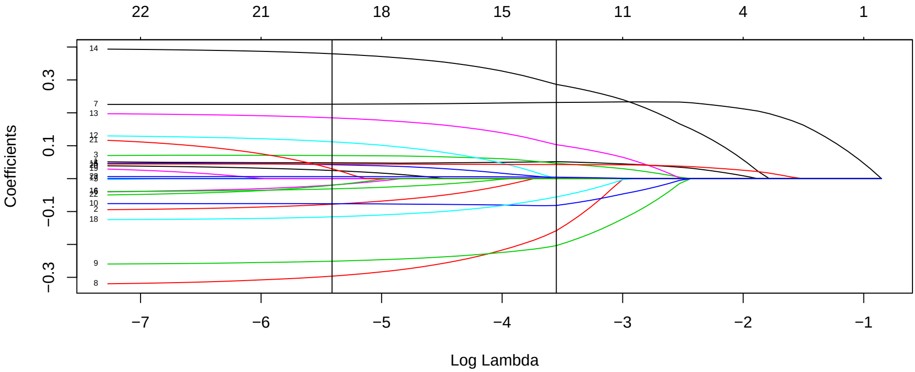
Lasso Example

```
plot(cvfits)
```



Lasso Example

```r
# Look at coefficient paths
plot(fit, xvar = "lambda", label = TRUE)
abline(v = log(cvfits$lambda.1se))
abline(v = log(cvfits$lambda.min))
```



- Which predictors should we keep?

Lasso Example

- Non-zero coefficients using minimum λ

```r
tidy(coef(object = cvfits, s = "lambda.min"))[, -2]
```

```
##             row       value
## 1  (Intercept)  5.528770925
## 2     FAM_SIZE  0.048060615
## 3    BLS_URBN2 -0.077868322
## 4     AS_COMP1  0.070170424
## 5     AS_COMP2  0.042287485
## 6     AS_COMP5 -0.019362364
## 7  logFINCBTAX  0.226253316
## 8   HIGH_EDU10 -0.296400487
## 9   HIGH_EDU11 -0.250989758
## 10  HIGH_EDU12 -0.076291240
## 11  HIGH_EDU14  0.111949131
## 12  HIGH_EDU15  0.184782458
## 13  HIGH_EDU16  0.379555756
## 14      ROOMSQ  0.044296291
## 15        SEX2 -0.031126759
## 16   MEMBRACE2 -0.116049421
## 17   MEMBRACE4  0.024737976
## 18   MEMBRACE5  0.030012470
## 19   MEMBRACE6 -0.020159464
## 20    BEDROOMQ  0.006159883
```

Lasso Example

- Non-zero coefficients using λ which is 1 away from the minimum λ

```
tidy(coef(object = cvfits, s = "lambda.1se"))[, -2]
```

```
##             row        value
## 1   (Intercept)  5.563038299
## 2      FAM_SIZE  0.051375451
## 3      AS_COMP1  0.046387574
## 4      AS_COMP2  0.001006932
## 5   logFINCBTAX  0.231545253
## 6    HIGH_EDU10 -0.158181034
## 7    HIGH_EDU11 -0.203536773
## 8    HIGH_EDU12 -0.081511551
## 9    HIGH_EDU15  0.103201066
## 10   HIGH_EDU16  0.286428937
## 11       ROOMSQ  0.042723830
## 12    MEMBRACE2 -0.055611375
## 13     BEDROOMQ  0.004523887
```

Ridge

- Lasso doesn't handle multicollinearity well.
- Ridge uses a different penalty:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{j \in s} (Y_j - \boldsymbol{X}_j^T \beta)^2 + \lambda \sum_{i=1}^{p} \beta_i^2$$

- But Ridge doesn't do model selection.

Elastic Net

- Elastic Net (Hui Zou and Hastie 2005) is a compromise between Lasso and Ridge:

$$\hat{\beta} = \arg\min_{\beta} \sum_{j \in s} (Y_j - \boldsymbol{X}_j^T \beta)^2 + \lambda \sum_{i=1}^{p} \left(\alpha |\beta_i| + (1-\alpha)\beta_i^2 \right)$$

- Now we also need to estimate α, the mixing parameter.

Elastic Net Example

- Will use the **caret** package to find λ and α

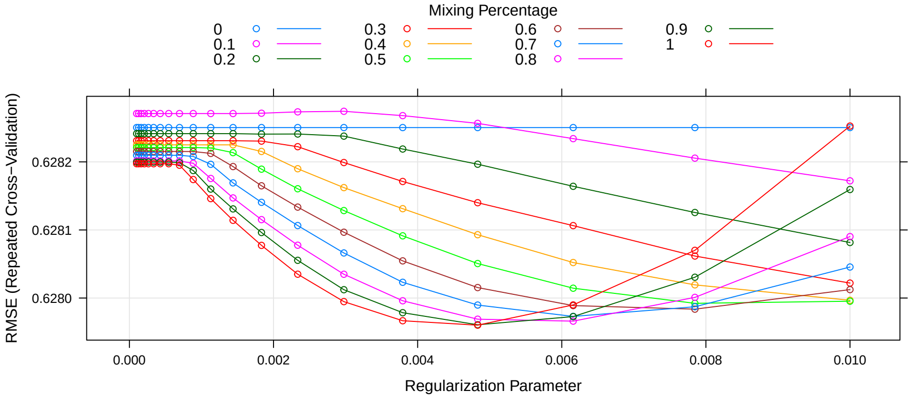
```
#Set-up grid of possible lambda and alpha values
lam <- 10^seq(-4, -2, length.out = 20)
alpha <- 0:10/10
grd <- expand.grid(lambda = lam, alpha = alpha)

#Set-up CV options
cv_opts <- trainControl(method = "repeatedcv", number = 10,
                        repeats = 5)

#Train glmnet model
cvfits2 <- train(logTOTEXPCQ ~ ., data = ds,
                 method = "glmnet", tuneGrid = grd,
                 trControl = cv_opts, standardize = TRUE)
```

Elastic Net Example

```
plot(cvfits2)
```




```
cvfits2$bestTune
```

```
##     alpha     lambda
## 217     1 0.00483293
```

Elastic Net Example

```r
bestfit2 <- cvfits2$finalModel
tidy(coef(bestfit2, s = cvfits2$bestTune$lambda))[,
    -2]
```

```
##             row       value
## 1   (Intercept)  5.52901842
## 2      FAM_SIZE  0.04795939
## 3     BLS_URBN2 -0.07631178
## 4      AS_COMP1  0.07002317
## 5      AS_COMP2  0.04173335
## 6      AS_COMP5 -0.01731301
## 7   logFINCBTAX  0.22634217
## 8    HIGH_EDU10 -0.29430811
## 9    HIGH_EDU11 -0.25026265
## 10   HIGH_EDU12 -0.07637131
## 11   HIGH_EDU14  0.11023492
## 12   HIGH_EDU15  0.18359929
## 13   HIGH_EDU16  0.37818364
## 14       ROOMSQ  0.04427220
## 15         SEX2 -0.03036356
## 16    MEMBRACE2 -0.11515514
## 17    MEMBRACE4  0.02345157
## 18    MEMBRACE5  0.02163652
## 19    MEMBRACE6 -0.01714617
## 20     BEDROOMQ  0.00614561
```

Modification of Elastic Net

- Weighting the penalty on the coefficients:
  - Adaptive Lasso (H. Zou 2006)

$$\hat{\boldsymbol{\beta}} = \arg\min_{\beta} \sum_{j \in s} (Y_j - \boldsymbol{X}_j^T \beta)^2 + \lambda \sum_{i=1}^{p} v_i \left(\alpha |\beta_i| + (1-\alpha)\beta_i^2 \right)$$

- Usually, the weights are the inverse of the absolute value of the least square estimates.

$$v_i = \left(|\hat{\beta}_i^{LS}| \right)^{-1}$$

Adaptive Elastic Net

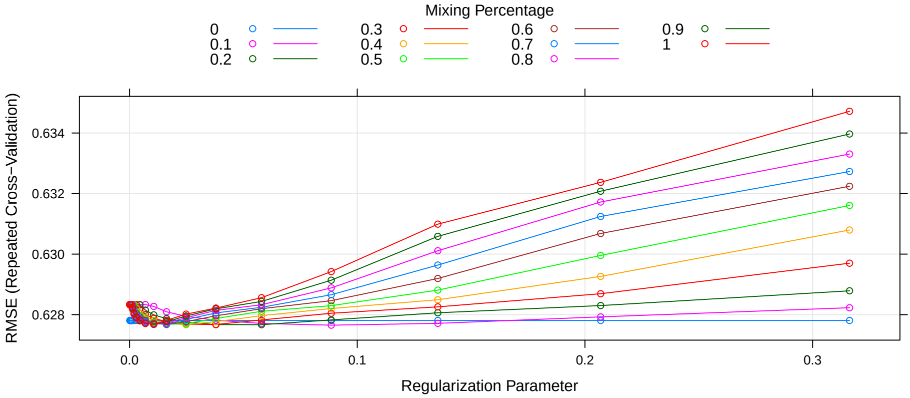
```r
# Penalty weights
pen <- abs(lm(y~scale(x))$coef[-1])^(-1)

#Set-up grid of possible lambda and alpha values
lam <- 10^seq(-4, -.5, length.out = 20)
alpha <- 0:10/10
grd <- expand.grid(lambda = lam, alpha = alpha)

#Train glmnet model with differing penalties
cvfits3 <- train(logTOTEXPCQ ~ ., data = ds,
                 method = "glmnet", tuneGrid = grd,
                 trControl = cv_opts, standardize = TRUE,
                 penalty.factor = pen)
```

Adaptive Elastic Net

```r
plot(cvfits3)
```




```r
cvfits3$bestTune
```

```
##    alpha     lambda
## 37   0.1 0.08858668
```

Adaptive Elastic Net

```r
bestfit3 <- cvfits3$finalModel
tidy(coef(bestfit3, s = cvfits3$bestTune$lambda))[,
    -2]
```

```
##             row          value
## 1  (Intercept)  5.44072008995
## 2     FAM_SIZE  0.05028440456
## 3    BLS_URBN2 -0.06619440018
## 4      AS_COMP1  0.07128374012
## 5      AS_COMP2  0.03595248568
## 6      AS_COMP5 -0.00004388746
## 7   logFINCBTAX  0.22646851880
## 8    HIGH_EDU10 -0.22833336599
## 9    HIGH_EDU11 -0.18111008330
## 10   HIGH_EDU12 -0.00048967566
## 11   HIGH_EDU13  0.07766592087
## 12   HIGH_EDU14  0.20769955850
## 13   HIGH_EDU15  0.27888858758
## 14   HIGH_EDU16  0.47536071604
## 15       ROOMSQ  0.04629865104
## 16         SEX2 -0.02193148866
## 17    MEMBRACE2 -0.11638118122
```

Elastic Net

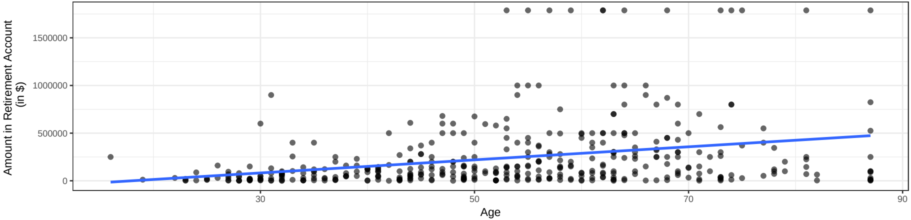
- Pros
  - Does model selection and parameter estimation simultaneously.
  - Can handle situations where you have more predictors than you have observations.
- Cons
  - Linear model
    - Can add polynomial terms and interactions.
  - Doesn't always pick the best model.
- What if we want to fit a non-parametric model?

Parametric Regression Example

- New Y: `IRAX`
- Question: "As of today, what is the total value of all retirement accounts, such as 401(k)s, IRAs, and Thrift Savings Plans that you own?"
- Let's model the relationship between `IRAX` and `AGE` with a simple linear regression model.
  - Only consider households with positive retirement funds.

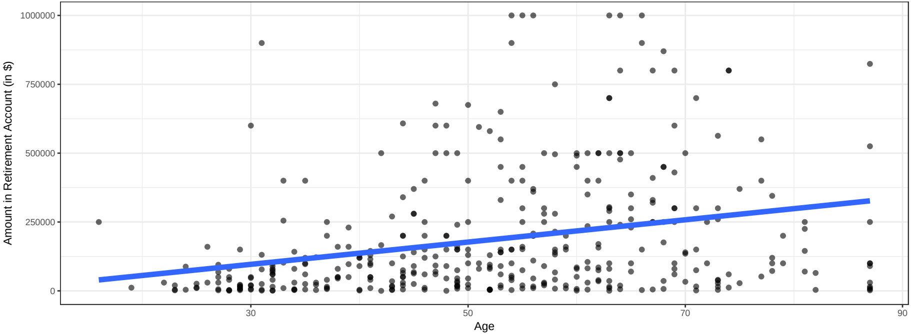
$$Y = \beta_o + \beta_1 X + \epsilon$$

Parametric Regression Example



- How will the top-coded individuals impact the fit?
  - Remove for the rest of our model building.

Parametric Regression Example

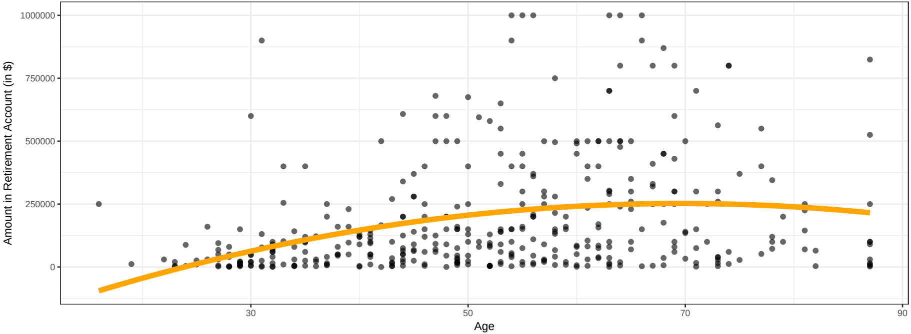


term	estimate	std.error	statistic	p.value
(Intercept)	-24982.318	37836.3473	-0.660273	0.5094684
AGE	4043.665	695.7692	5.811790	0.0000000

Parametric Regression Example

- Based on the plot, linear regression is reasonable.
- Have a significant p-value.
  - Does that imply this is the correct model?
- Easy to interpret predictor: For each one year increase in age, we expect the amount in their retirement account to increase by $ 4043.6645496.
- Based on prior knowledge, what would be a better parametric model to consider?

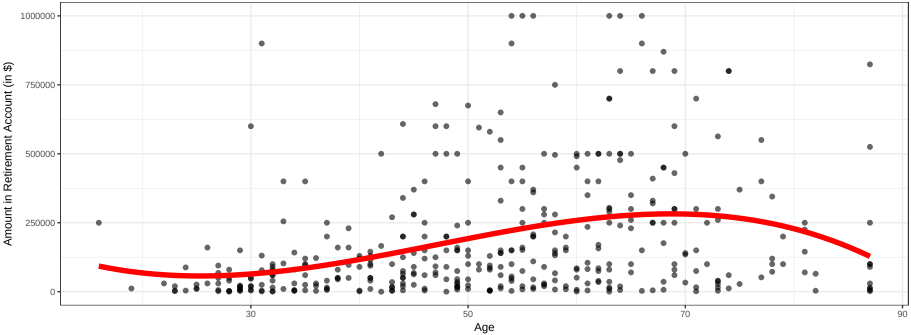
Parametric Regression Example



term	estimate	std.error	statistic	p.value
(Intercept)	185577.1	10782.27	17.211317	0.0000000
poly(AGE, 2)1	1255230.6	213478.06	5.879904	0.0000000
poly(AGE, 2)2	-681632.3	213478.06	-3.192985	0.0015227

- Problems fixed?

Parametric Regression Example



term	estimate	std.error	statistic	p.value
(Intercept)	185577.1	10695.57	17.350840	0.0000000
poly(AGE, 3)1	1255230.6	211761.42	5.927570	0.0000000
poly(AGE, 3)2	-681632.3	211761.42	-3.218869	0.0013953
poly(AGE, 3)3	-573416.2	211761.42	-2.707841	0.0070718

- Better?
- Are we sure we want to impose a global model?

Step Functions

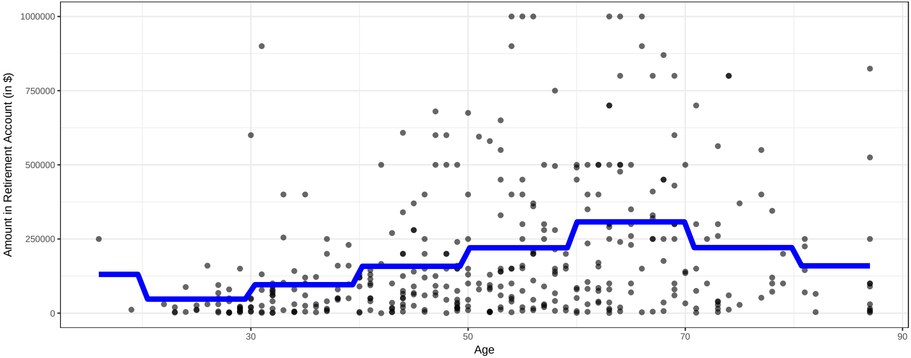
- Partition X value into K regions:

$$C_o(x) = I(x < \kappa_1) \qquad C_1(x) = I(\kappa_1 \leq x < \kappa_2)$$
$$C_2(x) = I(\kappa_2 \leq x < \kappa_3) \ldots C_K(x) = I(\kappa_K \leq x)$$

- Fit linear regression model based on these new predictor variables.
- Bridge between parametric and non-parametric models.
  - Still specifying a form for the model.
  - Allows a local fit: piece-wise constant functions.

Step Function Example

- Put breaks, called knots, at 20, 30, . . . , 90.



- Pro:
  - Local fits: Each constant function is driven by the data in that interval.
- Cons:
  - In each interval, the relationship between Y and X doesn't seem to be constant.
    - Would prefer to fit a linear or polynomial in each interval.

All about the Basis Functions

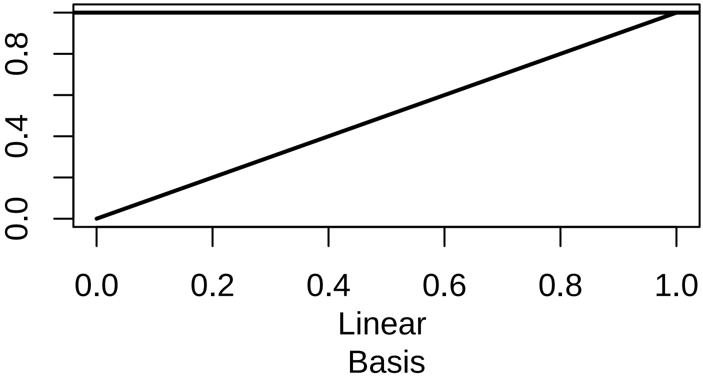
- To get a greater degree of flexibility, we will transform X, using basis functions: $h_0(X), h_1(X), h_2(X), \ldots, h_K(X)$.
- Then fit a linear regression model on the basis functions:

$$Y = \beta_o h_o(X) + \beta_1 h_1(X) + \cdots + \beta_K h_K(X) + \epsilon$$

- Already know some basis functions.

Regression Examples with Basis Functions

- Let $$h_o(X) = 1 \qquad h_1(X) = X$$
  then we get simple linear regression
  $$Y = \beta_o + \beta_1 X + \epsilon$$



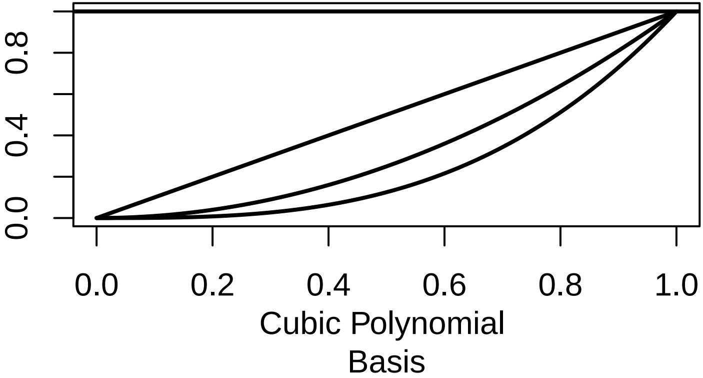
Regression Examples with Basis Functions

- Let

$$h_o(X) = 1 \qquad h_1(X) = X \qquad h_2(X) = X^2 \qquad h_3(X) = X^3$$

then we get polynomial regression

$$Y = \beta_o + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$$



Regression Examples with Basis Functions

- Let

$$h_0(X) = I(X < \kappa_1) \qquad h_1(X) = I(\kappa_1 \leq X < \kappa_2) \cdots h_K(X) = I(\kappa_K \leq X)$$

then we get regression with piece-wise constant functions

$$Y = \beta_o I(X < \kappa_1) + \beta_1 I(\kappa_1 \leq X < \kappa_2) + \cdots + \beta_K I(\kappa_K \leq X) + \epsilon$$

- So far, all of the basis examples have been parametric models.
- But we can use other basis functions to achieve more flexible approximations of $f(X)$.
  - These will make very large linear models. So why will we consider these non-parametric?

Regression Splines

Consider the following basis functions:

$$h_1(X) = I(X < \kappa_1) \quad h_2(X) = I(\kappa_1 \leq X < \kappa_2) \quad \cdots \quad h_{K+1}(X) = I(\kappa_K \leq X)$$

$$h_{K+2} = Xh_1(X) \quad h_{K+3} = Xh_2(X) \quad \cdots \quad h_{2(K+1)}(X) = Xh_K(X)$$

$$h_{2K+3} = X^2 h_1(X) \quad h_{2K+4} = X^2 h_2(X) \quad \cdots \quad h_{3(K+1)}(X) = X^2 h_K(X)$$

$$h_{3K+4} = X^3 h_1(X) \quad h_{3K+5} = X^3 h_2(X) \quad \cdots \quad h_{4(K+1)}(X) = X^3 h_K(X)$$

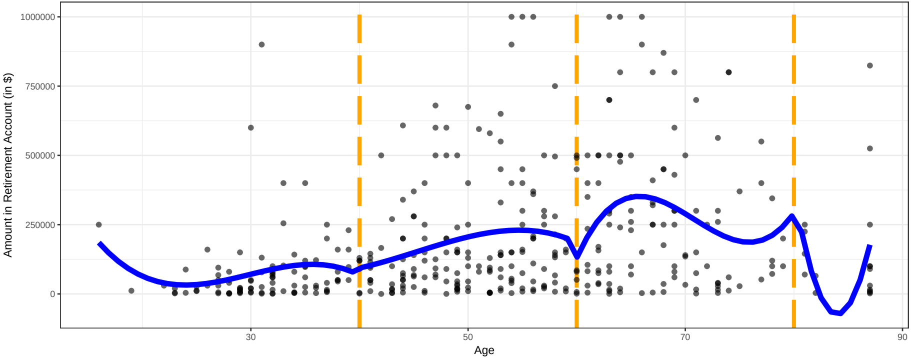
Then the regression equation is

$$Y = \sum_{j=1}^{4(K+1)} \beta_j h_j(X) + \epsilon$$

- What type of fit will we get from these bases?

Regression Splines

- Knots: 40, 60, 80



- What is wrong with the mean function from this set of bases?
  - Remember: Want a flexible class of functions which approximate $f(X)$ where all we are assuming is that $f(X)$ is a smooth function.

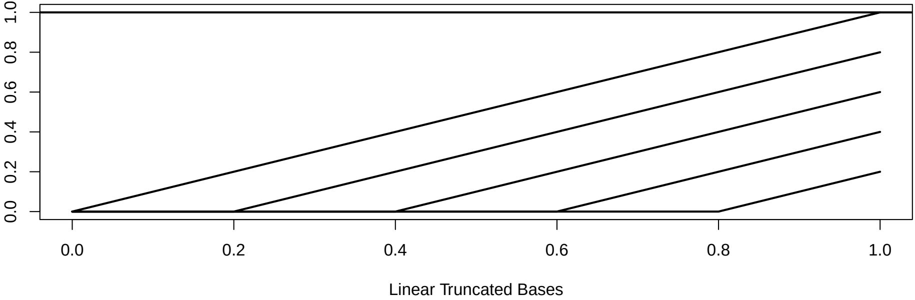
Regression Splines

- Add some restrictions to the class of functions
  - Want $f(X)$ to be continuous at the knots
  - May even want $f(X)$ to have up to p derivatives at the knots.
- Instead of incorporating the constraints into our estimation procedure, let's use a new set of basis functions which satisfy the constraints.

Truncated Splines Basis

$$h_1(X) = 1 \quad h_2(X) = X$$

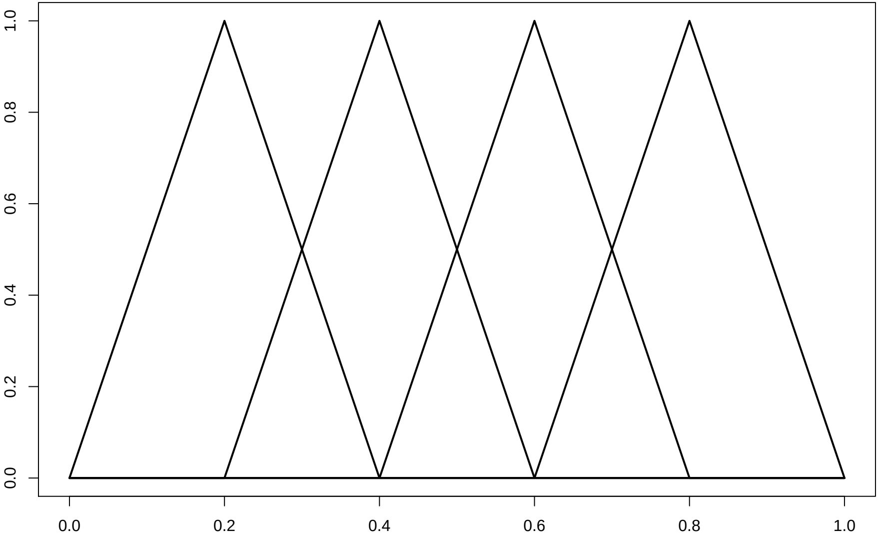
$$h_3 = (X - \kappa_1)_+ \quad h_4 = (X - \kappa_2)_+ \quad \cdots \quad h_{K+2}(X) = (X - \kappa_K)_+$$



- Can add terms $(X - \kappa_k)_+^p$ to get higher order terms.

B Splines

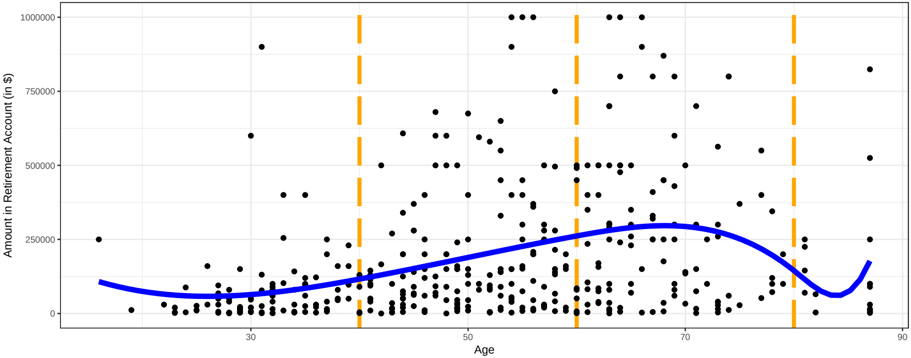
- Hard to find R functions for truncated spline basis because a more computationally feasible basis expansion are B-Splines.



B Splines

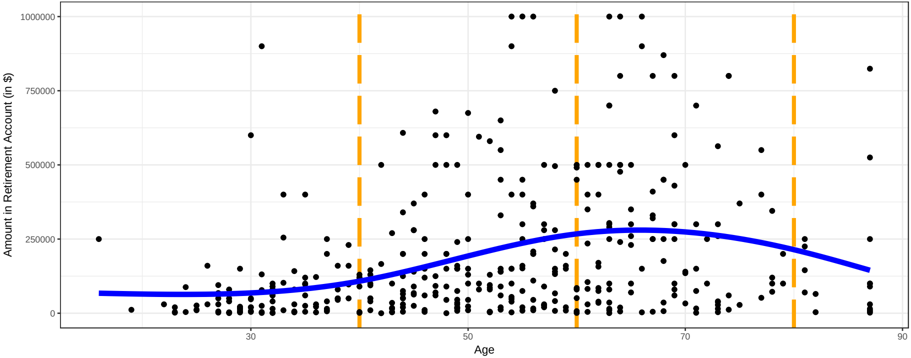
- Formulae for the B Spline functions are very complicated so I won't include them here.
- Like the truncated spline basis, the user chooses:
  - K = number of knots
  - Placement of the knots.
  - d = degree of polynomials.
- If one selects the same K, d, and knot placement, then the fit from the truncated spline basis will be equivalent to the fit from the B spline basis.

Regression Splines



- Now we have a smooth piece-wise cubic fit.
- **One issue**: The behavior at the ends can be erratic.
  - **Solution**: Natural cubic spline: adds constraint that solution is linear beyond the boundary knots.

Regression Splines



- Now we have a smooth piece-wise cubic fit.
  - Smooth = d - 1 derivatives everywhere.
- **One issue**: The behavior at the ends can be erratic.
  - **Solution**: Natural cubic spline: adds constraint that solution is linear beyond the boundary knots.

Regression Splines

- Degrees of freedom (df) = measure of the flexibility of a function
- For B-Splines: df = (# of regions) (parameters per region) - (# of knots)(# of constraints) = (K + 1)(d+1) - K(d) = K + d + 1
- For Natural Splines: df = K + d + 1 - 4
  - Can have more knots than B-Spline but get same measure of "flexibility"!

Regression Splines

- **Question**: How many knots should I have?
- **Question**: Where should I place the knots? Uniformly? Based on where I think the function will change more rapidly?
- **Question**: What degree of polynomial should I use?
- Could use cross-validation to determine the optimal df.
  - Let's learn about another type of splines where knot number and placement are handled for us.

Smoothing Splines

- Class of functions: All functions $f(X)$ with two continuous derivatives.
- Criterion: Find the function that minimizes the penalized residual sum of squares:

$$\sum_{i=1}^{N}(Y_i - f(X_i))^2 + \lambda \int \left[f''(t)\right]^2 dt$$

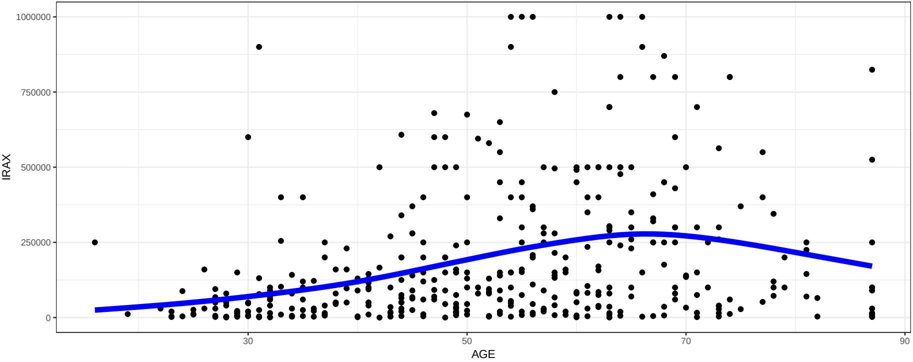
- $\lambda \geq 0$ is the penalty parameter, just like in Elastic net.
- Balance goodness of fit with smoothness of function.

Smoothing Splines

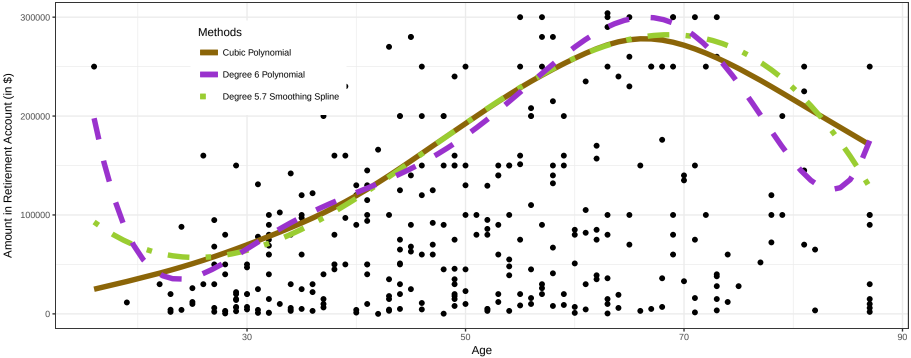
- Although this is an infinite dimensional function space, it can be shown that the minimizer of the criterion is unique and finite dimensional!
- In fact, the solution is a natural cubic spline with knots at the unique values of X!
- **Question**: But haven't we over-fit the data, if we have N knots/df?
  - The penalty term shrinks the spline coefficients towards a linear fit.
- **Question**: How to select the penalty term?
  - Cross-validation!

Smoothing Splines

```
mod <- smooth.spline(x = dat$AGE, y = dat$IRAX, cv = TRUE)
dat2 <- data.frame(x = mod$x, fits = mod$y)
ggplot(dat, aes(x = AGE, y = IRAX)) + geom_point() +
    geom_line(data = dat2, aes(x = x, y = fits), color = "blue
        size = 2)
```



Non-Parametric Fit Versus Parametric Fit



- Which to use?
- The ratio of test MSE for the cubic polynomial versus smoothing spline is 0.9991905.

Non-Parametric Versus Parametric

- Why use a non-parametric model?
  - Can have greater prediction accuracy.
  - Can help us see trends in the data that we would have missed by imposing a global model/specific shape.
- Why use a parametric model?
  - Don't have much data.
  - Care about inference and interpretability.
  - Can get a comparable fit and prediction accuracy as a non-parametric model.

Moving Beyond One Predictor

Generalized Additive Models:

- Extension of generalized linear regression
- For each X, allow for a flexible fit
- Assumes additivity: Don't include interactions
  - To avoid the "curse of dimensionality"
- Model:

$$Y = \beta_o + \sum_{i=1}^{p} f_i(X_i) + \epsilon$$

Moving from Regression to Classification

- The variable of interest, Y, is a binary categorical variable.
- The predictor variables, X's, are still a mix of categorical and quantitative variables.
- Two options for modeling Y:

1. Build a model that estimates $P(Y_i = 1) = f(\boldsymbol{X}_i)$
   - Then the classification rule is

$$\text{When } \hat{f}(\boldsymbol{X}_i) \geq 0.5 \Rightarrow \hat{Y}_i = 1$$
$$\text{When } \hat{f}(\boldsymbol{X}_i) < 0.5 \Rightarrow \hat{Y}_i = 0$$

2. Build a model which directly predicts the class label

$$\text{When } \hat{f}(\boldsymbol{X}_i) \geq c \Rightarrow \hat{Y}_i = 1$$
$$\text{When } \hat{f}(\boldsymbol{X}_i) < c \Rightarrow \hat{Y}_i = 0$$

- In the second case, $f(\cdot)$ is probably still a measure related to the likelihood of class assignment but is not a probability.

Classification

- Want to find a classifier $f(X)$ with decision rule such that you accurately predict the classes of Y for new observations.
- Test MSE is no longer a good performance measure.
- Performance measure:
  - Test Accuracy Rate: (# correct classifications in test dataset)$/n_o$
- Also useful: Confusion matrix:

	$\hat{Y} = 1$	$\hat{Y} = 0$
Y = 1	TP	FN
Y = 0	FP	TN

- There are other measures to use if...
  - You have extreme class imbalance.
  - One error is worse than the other.

Logistic Regression

- Assume Y_i are independent Bernoulli random variables with $P(Y_i = 1|\boldsymbol{X}_i) = f(\boldsymbol{X}_i)$.
- From generalized linear modeling theory, we can use the logit link function

$$
\begin{aligned}
\text{logit}\,(f(\boldsymbol{X}_i)) = \log\left(\frac{f(\boldsymbol{X}_i)}{1 - f(\boldsymbol{X}_i)}\right) &= \beta_o + \beta_1 X_1 + \cdots + \beta_p X_p \\
&= \boldsymbol{X}_i^T \beta
\end{aligned}
$$

- And then solve out for $f(\boldsymbol{X}_i)$:

$$
f(\boldsymbol{X}_i) = \exp\left(\boldsymbol{X}_i^T \beta\right)\left[1 + \exp\left(\boldsymbol{X}_i^T \beta\right)\right]^{-1}
$$

Logistic Regression

- The parameters are estimated by minimizing the negative log-likelihood based on the Bernoulli distribution:

$$\hat{\beta} = \arg\min_{\beta} \left[- \sum_{i \in s} \left\{ Y_i \boldsymbol{X}_i^T \beta - \log \left[1 + \exp(\boldsymbol{X}_i^T \beta) \right] \right\} \right]$$

- Then the estimated classifier is:

$$\hat{f}(\boldsymbol{X}_i) = \exp\left(\boldsymbol{X}_i^T \hat{\beta}\right) \left[1 + \exp\left(\boldsymbol{X}_i^T \hat{\beta}\right) \right]^{-1}$$

with rule:

$$\text{When } \hat{f}(\boldsymbol{X}_i) \geq 0.5 \Rightarrow \hat{Y}_i = 1$$
$$\text{When } \hat{f}(\boldsymbol{X}_i) < 0.5 \Rightarrow \hat{Y}_i = 0$$

Logistic Regression Example

- Question: "As of today, what is the total value of all retirement accounts, such as 401(k)s, IRAs, and Thrift Savings Plans that you own?"
- **New** Y = non-zero or zero balance in retirement accounts

Logistic Regression Example

```
# Create dataset
ds <- dplyr::select(ce, IRAX, TOTEXPCQ, FAM_SIZE, BLS_URBN,
    AS_COMP1, AS_COMP2, FINCBTAX, HIGH_EDU, ROOMSQ,
    AS_COMP4, AS_COMP5, SEX, AGE, BEDROOMQ) %>% # Create vari
# IRAX
mutate(IRAX_cat = ifelse(IRAX > 0, "Yes", "No")) %>%
    # Remove missing values
na.omit() %>% # Remove IRAX (so we don't accidentally use it a
# predictor!)
select(-IRAX)
```

Logistic Regression Example

```r
# Standard way of fitting logistic model
mod_log <- glm(as.factor(IRAX_cat) ~ ., data = ds,
    family = "binomial")
# But summary() doesn't have the CV Accuracy
# summary(mod_log)
```

Logistic Regression Example

- Instead let's use the caret library functions again:
  - Don't need a grid of parameter values still only fitting one model

```
# Set-up CV options
cv_opts <- trainControl(method = "repeatedcv", number = 10,
    repeats = 5)
# Build logistic regression model
mod_log <- train(IRAX_cat ~ ., data = ds, method = "glm",
    trControl = cv_opts, family = "binomial")
```

Logistic Regression Example

```
mod_log
```

```
## Generalized Linear Model
##
## 1041 samples
##   14 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 5 times)
## Summary of sample sizes: 937, 937, 937, 938, 937, 937, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7871434  0.5361791
```

Logistic Elastic Net

- Just like in linear regression, the kitchen sink model will not always have the best predictive abilities since we may have over fit the data.
- Therefore, again, we want to model selection.
- Luckily, there is a Logistic Elastic Net!
- Find β that minimize the negative log likelihood with the penalty term:

$$\begin{aligned}\hat{\beta} =& \arg\min_{\beta} \left[- \sum_{i \in s} \left\{ Y_i \boldsymbol{X}_i^T \beta - \log \left[1 + \exp(\boldsymbol{X}_i^T \beta) \right] \right\} \right. \\ & \left. + \lambda \sum_{i=1}^{p} \left(\alpha |\beta_i| + (1-\alpha) \beta_i^2 \right) \right]\end{aligned}$$

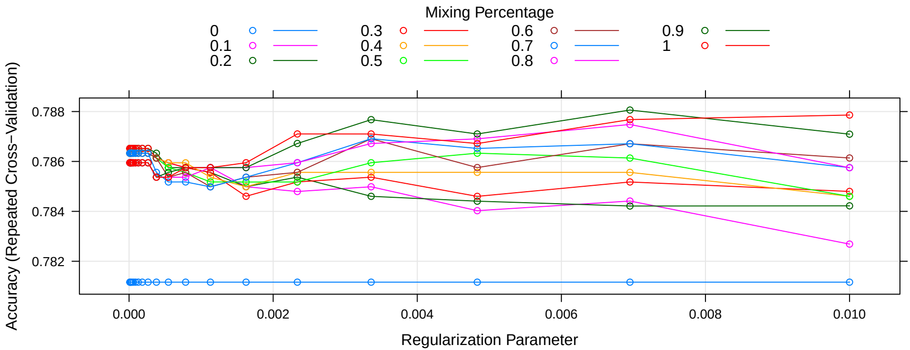
Elastic Net Example

```r
# Set-up grid of possible lambda and alpha values
lam <- 10^seq(-2, -5, length.out = 20)
alpha <- 0:10/10
grd <- expand.grid(lambda = lam, alpha = alpha)
# Train the Logistic Elastic Net Model
cv_en <- train(IRAX_cat ~ ., data = ds, method = "glmnet",
    tuneGrid = grd, trControl = cv_opts, family = "binomial",
    standardize = TRUE)
```

Elastic Net Example

- What does the blue curve represent?

```
plot(cv_en)
```




```
cv_en$bestTune
```

```
##     alpha      lambda
## 199   0.9 0.006951928
```

Elastic Net Example

```
bestfit_en <- cv_en$finalModel
tidy(coef(bestfit_en, s = cv_en$bestTune$lambda))[,
    -2]
```

```
##             row         value
## 1  (Intercept) -2.90278846267
## 2     TOTEXPCQ  0.00007184674
## 3     FAM_SIZE -0.11102709166
## 4     FINCBTAX  0.00001619891
## 5   HIGH_EDU10 -1.81575089431
## 6   HIGH_EDU11 -1.66852889329
## 7   HIGH_EDU12 -0.55097659328
## 8   HIGH_EDU14  0.30067725161
## 9   HIGH_EDU15  0.79713839165
## 10  HIGH_EDU16  0.78521074775
## 11      ROOMSQ  0.10543099160
## 12    AS_COMP4 -0.10032602227
## 13    AS_COMP5 -0.13993434459
## 14        SEX2 -0.09204564162
## 15         AGE  0.01802914592
## 16   MEMBRACE2 -0.52546019322
## 17   MEMBRACE4  0.00458822982
## 18   MEMBRACE5  0.80868679726
## 19   MEMBRACE6  0.35127504181
## 20    BEDROOMQ  0.00892619260
```

Adaptive Elastic Net Example

```r
# Create penalty weights
x <- scale(model.matrix(IRAX_cat ~ ., data = ds)[,
    -1])
pen_log <- abs(glm(as.factor(ds$IRAX_cat) ~ x, family = "binom
# Train the Logistic Adaptive Elastic Net Model
cv_ena <- train(IRAX_cat ~ ., data = ds, method = "glmnet",
    tuneGrid = grd, trControl = cv_opts, family = "binomial",
    penalty.factor = pen_log, standardize = TRUE)
```

Example: Compare Performance

- Test Accuracy Rates:

```r
# Logistic Regression
mod_log$results$Accuracy
```

```
## [1] 0.7871434
```

```r
# Elastic Net Logistic Regression
max(cv_en$results$Accuracy)
```

```
## [1] 0.7880604
```

```r
# Adaptive Elastic Net Logistic Regression
max(cv_ena$results$Accuracy)
```

```
## [1] 0.787108
```

Support Vector Classifier

- Now assume:

$$Y = \begin{cases} 1 & : \text{category 1 (e.g. yes)} \\ -1 & : \text{category 2 (e.g. no)} \end{cases}$$

Will work better than 0, 1.

- We want to create a decision boundary:

$$f(\boldsymbol{X}) = \beta_o + \boldsymbol{X}^T \beta$$

.

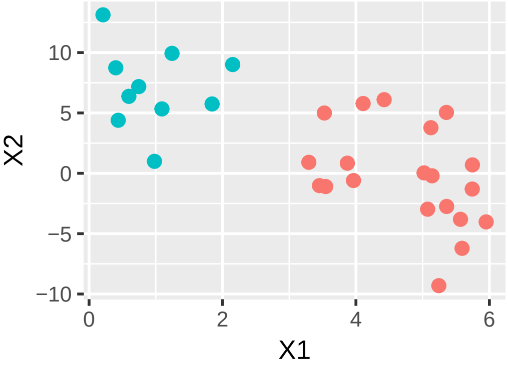
- Rule:

$$\text{When } \hat{f}(\boldsymbol{X}_i) \geq 0 \Rightarrow \hat{Y}_i = 1$$
$$\text{When } \hat{f}(\boldsymbol{X}_i) < 0 \Rightarrow \hat{Y}_i = -1$$

- How should we estimate the decision boundary?
  - How should we select the coefficient estimates?

Support Vector Classifier: Separable Case

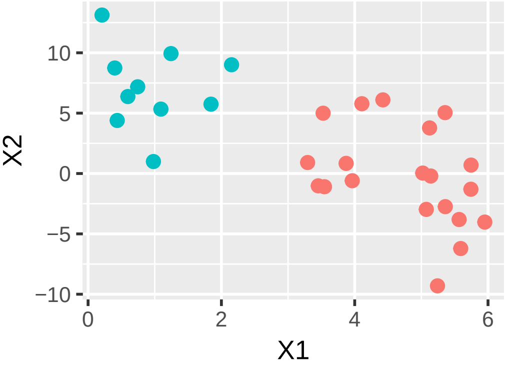
- Assume we have two predictor variables, X_1 and X_2, and that the data are perfectly separable.



- Want to insert a decision boundary.

Support Vector Classifier: Separable Case

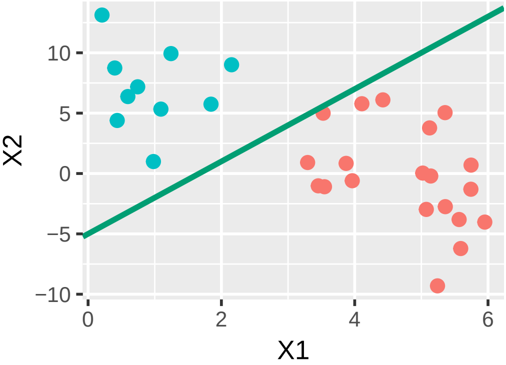
- Assume we have two predictor variables, X_1 and X_2, and that the data are perfectly separable.



- Want to insert a decision boundary.
  - Where is the **best** place to put the decision boundary?

Support Vector Classifier: Separable Case

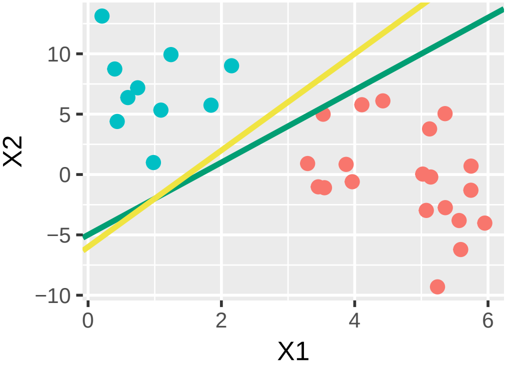
- Assume we have two predictor variables, X_1 and X_2, and that the data are perfectly separable.



- Want to insert a decision boundary.
  - Where is the **best** place to put the decision boundary?

Support Vector Classifier: Separable Case

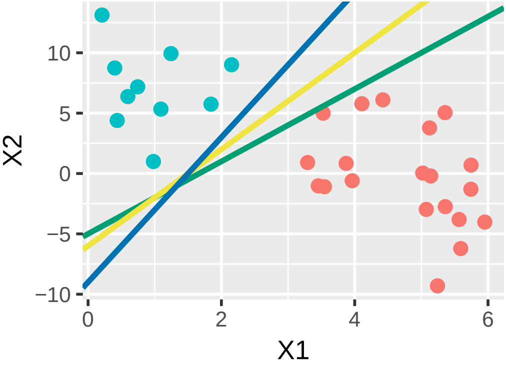
- Assume we have two predictor variables, X_1 and X_2, and that the data are perfectly separable.



- Want to insert a decision boundary.
  - Where is the **best** place to put the decision boundary?

Support Vector Classifier: Separable Case

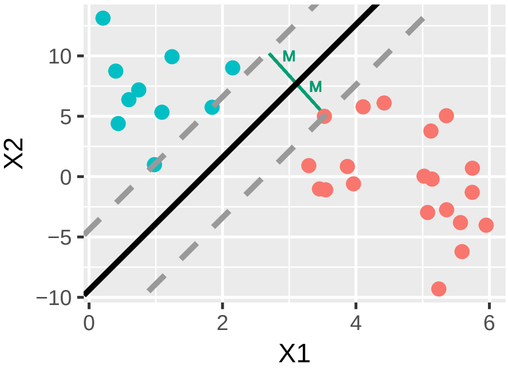
- Assume we have two predictor variables, X_1 and X_2, and that the data are perfectly separable.



- Want to insert a decision boundary.
  - Where is the **best** place to put the decision boundary?

Support Vector Classifier: Separable Case

- **Key Idea**: Put the decision boundary where you can maximize the margin between the boundary and the closest points.



- If we have a large margin, then we should have better classification rates for future observations.

Support Vector Classifier: Separable Case

- Goal: Separate the predictor space with a linear decision boundary:

$$f(\boldsymbol{X}) = \beta_o + \boldsymbol{X}^T\beta = 0$$

- Guiding principles:
  - Create groups that are as homogeneous as possible with respect to y.
  - Make the margin as large as possible.

Support Vector Classifier: Separable Case

- Notice:

$$Y_i(\beta_o + \boldsymbol{X}_i^T \beta) > 0 \Rightarrow \text{Correctly classified}$$
$$Y_i(\beta_o + \boldsymbol{X}_i^T \beta) < 0 \Rightarrow \text{In-correctly classified}$$

  - Then $Y_i(\beta_o + \boldsymbol{X}_i^T \beta)$ is a signed measure of how correct or in-correct a classification is.
  - Since data are perfectly separable, can find $\boldsymbol{\beta}$ and β_o such that $Y_i(\beta_o + \boldsymbol{X}_i^T \beta)$ is positive for all i.

- Pick values of $\boldsymbol{\beta}$ and β_o so that the distance between the decision boundary and the closest point is as large as possible.

Support Vector Classifier: Separable Case

- The decision boundary is defined by the following optimization:

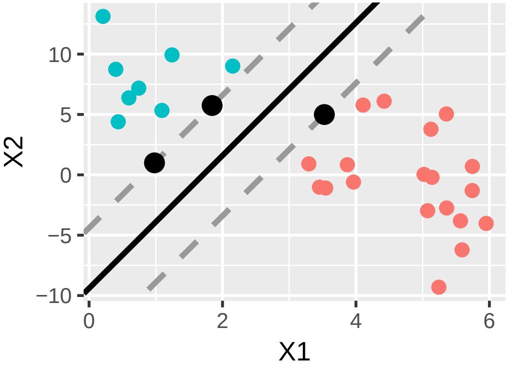
$$\max_{\boldsymbol{\beta}, \beta_o, ||\boldsymbol{\beta}||=1} M$$

$$\text{subject to } Y_i(\beta_o + \boldsymbol{X}_i^T \beta) \geq M, \text{ for all } i.$$

- Hard boundary

Support Vector Classifier: Separable Case

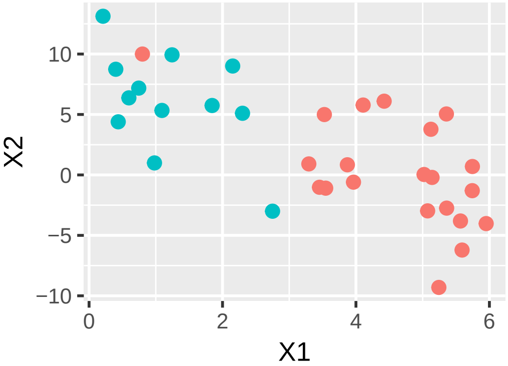
- **Key Idea**: Put the decision boundary where you can maximize the margin between the boundary and the closest points.



- The three black points are called support vectors (Hastie, Tibshirani, and Friedman 2001).
- **What about when the data aren't perfectly separable?**

Support Vector Classifier: Non-separable Case

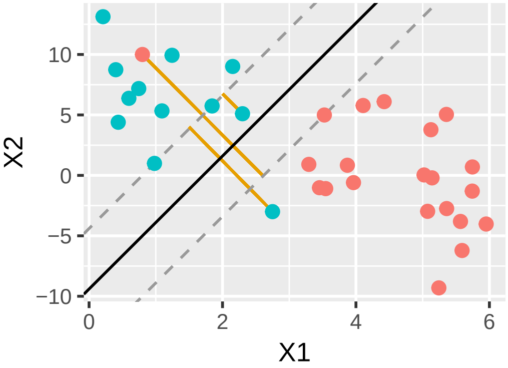
- Assume we have two predictor variables, X_1 and X_2, and that the data are not perfectly separable.



- Want to insert a decision boundary.
  - Where is the **best** place to put the decision boundary?
- Again try to maximize the margin but allow for some misclassifications.

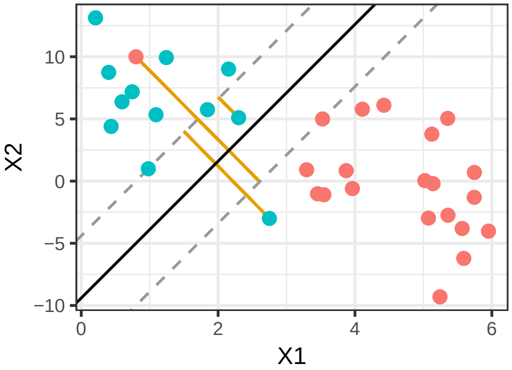
Support Vector Classifier: Non-separable Case

- We want to maximize the margin while still allowing some points to be on the wrong side of the margin boundary.
- Let $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_n)$ be a measure of how far each point is on the **wrong** side of the margin boundary.
  - Let $\xi_i = 0$ for correctly classified points on the correct side of the margin boundary.



Support Vector Classifier

- As $\sum \xi_i$ increases, allows for larger margin.
  - More stable classification $\Rightarrow$ less variance.
  - Greater number of misclassifications $\Rightarrow$ more bias.
- Therefore bounding $\sum \xi_i$ will be an important tuning parameter.



Support Vector Classifier: Non-separable Case

- The decision boundary is defined by the following optimization (Berk 2008):

$$\max_{\boldsymbol{\beta}, \beta_o, ||\boldsymbol{\beta}||=1} M$$

  - subject to:
    - $Y_i(\beta_o + \boldsymbol{X}_i^T \boldsymbol{\beta}) \geq M(1 - \xi_i)$ for all $\xi_i \geq 0$,
    - $\sum \xi_i \leq C$.
- C is a tuning parameter.
- Soft boundary.

Support Vector Classifier: Non-separable Case

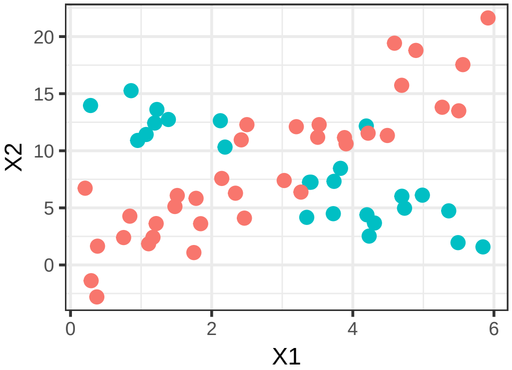
- Note:

$$\xi_i = |Y_i - f(\boldsymbol{X}_i)|$$

- Then
  - $0 < \xi_i < 1 \Rightarrow$ wrong side of the margin boundary but correctly classified.
  - $\xi_i > 1 \Rightarrow$ mis-classified.
- As C increases, there are more support vectors.
  - More stable solution $\Rightarrow$ less variance.
  - More misclassifications $\Rightarrow$ greater bias.

Support Vector Machines

- **What if we need a non-linear decision boundary?**



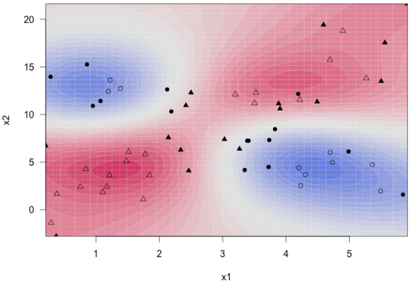
Support Vector Machines

- **What if we need a non-linear decision boundary?**

- Then we replace
$$f(\boldsymbol{X}) = \beta_o + \boldsymbol{X}^T\boldsymbol{\beta}$$
with
$$f(\boldsymbol{X}) = \beta_o + \boldsymbol{h}(\boldsymbol{X})^T\boldsymbol{\beta}$$
where $\boldsymbol{h}(\boldsymbol{X})$ are basis functions.
  - Solution will be linear in the enlarged feature space and non-linear in the predictor space.

Support Vector Machines

- **What if we need a non-linear decision boundary?**



SVM Fit

- The computation behind fitting the SVM is beyond this class but we can still talk about the structure of the fit.
- An equivalent formula for the fitting criterion is:

$$\min_{\beta,\beta_o} \sum_{i=1n} [1 - Y_i f(\boldsymbol{X}_i)]_+ + \lambda ||\beta||^2$$

where $\lambda \geq 0$ and $[r]_+ = \max(0, r)$ (Hastie, Tibshirani, and Friedman 2001).

  - Hinge-Loss + Penalty
  - Penalty controls width of the margin.

SVM Fit

- The minimizer can be shown to have the form

$$\hat{f}(\boldsymbol{X}) = \sum_{i=1}^{n} \hat{c}_i K(\boldsymbol{X}_i, \boldsymbol{X})$$

where $K(\cdot,\cdot)$ is a Kernel.

- Example Kernels:
  - Linear: $K(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{u}^t\boldsymbol{v}$
  - Polynomial: $K(\boldsymbol{u},\boldsymbol{v}) = (1 + \boldsymbol{u}^t\boldsymbol{v})^d$
  - Radial: $K(\boldsymbol{u},\boldsymbol{v}) = \exp[-\gamma(\boldsymbol{u}-\boldsymbol{v})^t(\boldsymbol{u}-\boldsymbol{v})]$
- Think of Kernels as a generalization of the inner product.
  - Different Kernels correspond to different basis functions.
- **Kernel Trick**: To fit the model, I don't need to know the basis functions. I only need to pick a Kernel.
  - But we will have additional parameters to estimate for some of the kernels!

SVM Example: Linear

```
# Create design matrix
# Selected subset of *useful* predictors
x <- model.matrix(IRAX_cat ~ AGE + FINCBTAX + TOTEXPCQ +
    HIGH_EDU, data = ds)[, -1]
# Set-up CV options
cv_opts <- trainControl(method = "CV", number = 5)
# Range of values for hyper-parameter
C <- c(0.1, 1, 2, 10, 20)
# Train Linear SVM model
cv_svm_l <- train(x, ds$IRAX_cat, method = "svmLinear",
    preProc = c("center", "scale"), tuneGrid = data.frame(C),
    trControl = cv_opts)
```

SVM Example: Linear

```
cv_svm_l
```

```
## Support Vector Machines with Linear Kernel
##
## 1041 samples
##   10 predictor
##    2 classes: 'No', 'Yes'
##
## Pre-processing: centered (10), scaled (10)
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 833, 832, 834, 832, 833
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##    0.1  0.7752096  0.5052824
##    1.0  0.7751911  0.5071442
##    2.0  0.7761573  0.5095491
##   10.0  0.7751957  0.5072047
##   20.0  0.7751957  0.5072047
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was C = 2.
```

SVM Example: Radial

- Another hyper-parameters:
  - `sig` = inverse kernel width

```r
# Hyper-parameter grid
sig <- sigest(x)
C <- c(0.1, 0.3, 0.5, 1, 2)
grd <- expand.grid(.sigma = sig, .C = C)
# Train Radial SVM Model
cv_svm_r <- train(x, ds$IRAX_cat, method = "svmRadial",
    preProc = c("center", "scale"), tuneLength = 10,
    trControl = cv_opts)
```

SVM Example: Radial

```
cv_svm_r
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 1041 samples
##   10 predictor
##    2 classes: 'No', 'Yes'
##
## Pre-processing: centered (10), scaled (10)
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 832, 833, 832, 833, 834
## Resampling results across tuning parameters:
##
##   sigma       C    Accuracy   Kappa
##   0.02602973  0.1  0.7530385  0.4594774
##   0.02602973  0.3  0.7703464  0.4953161
##   0.02602973  0.5  0.7837804  0.5206901
##   0.02602973  1.0  0.7847697  0.5229460
##   0.02602973  2.0  0.7722695  0.4992167
##   0.05716253  0.1  0.7530662  0.4572479
##   0.05716253  0.3  0.7799572  0.5115062
##   0.05716253  0.5  0.7790095  0.5113171
##   0.05716253  1.0  0.7732494  0.5008101
##   0.05716253  2.0  0.7713079  0.4976603
##   0.32430198  0.1  0.7473384  0.4266935
##   0.32430198  0.3  0.7732493  0.5004794
##   0.32430198  0.5  0.7713124  0.5016272
##   0.32430198  1.0  0.7799525  0.5217288
##   0.32430198  2.0  0.7809371  0.5249126
##
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were sigma = 0.02602973 and C = 1.
```

Support Vector Machines

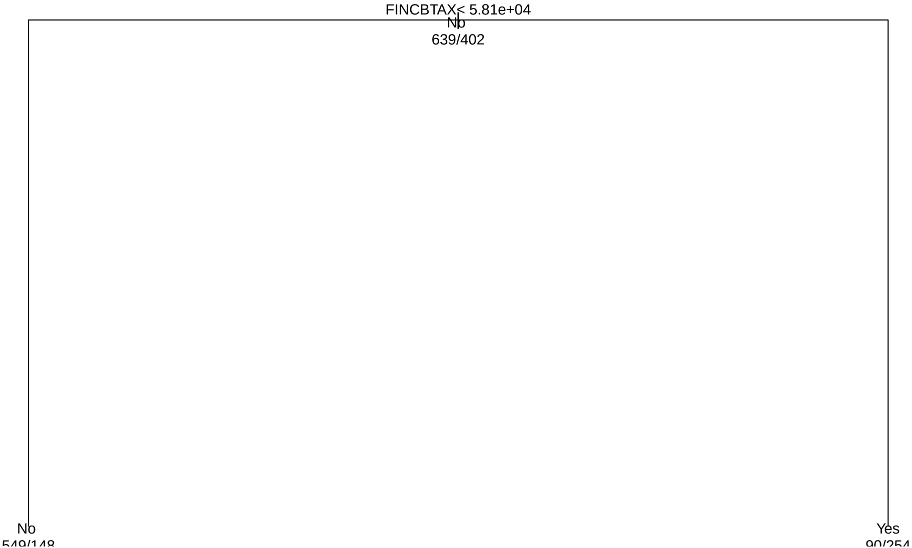
- Pros
  - Very flexible
  - Computationally fast because only a subset of the data impact the model fit.
  - Resistant to outliers
- Cons
  - Have to choose a basis representations
  - Hard to interpret
  - Have to estimate the hyper-parameters

Classification Trees

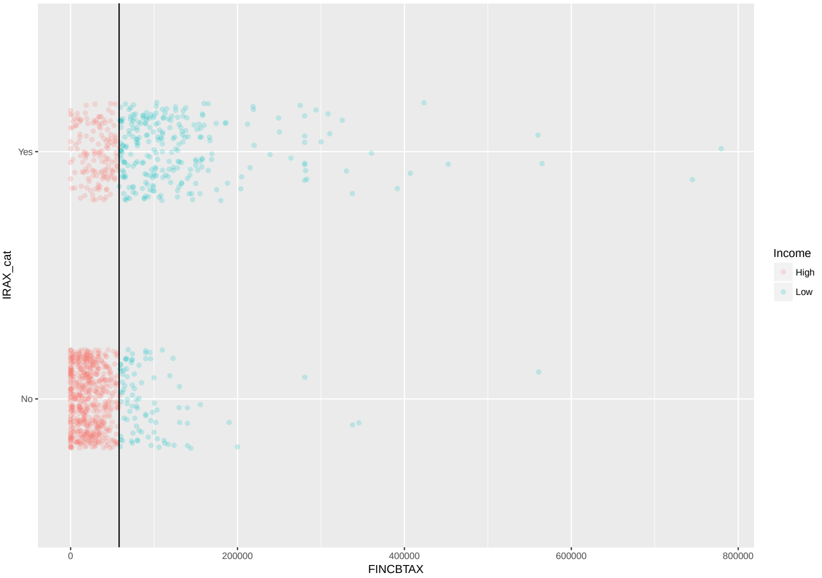
- Recursively splits sample into two disjoint groups based on one of the predictor variables.
  - Pick the split that leads to the greatest increase in group purity.
- Build a large tree and then *prune* back some splits.
  - Rule: Keep split if decreases the error by a factor of 1%.
- Rule: For a given end node,

$$\hat{Y}_i = \text{most common category}$$

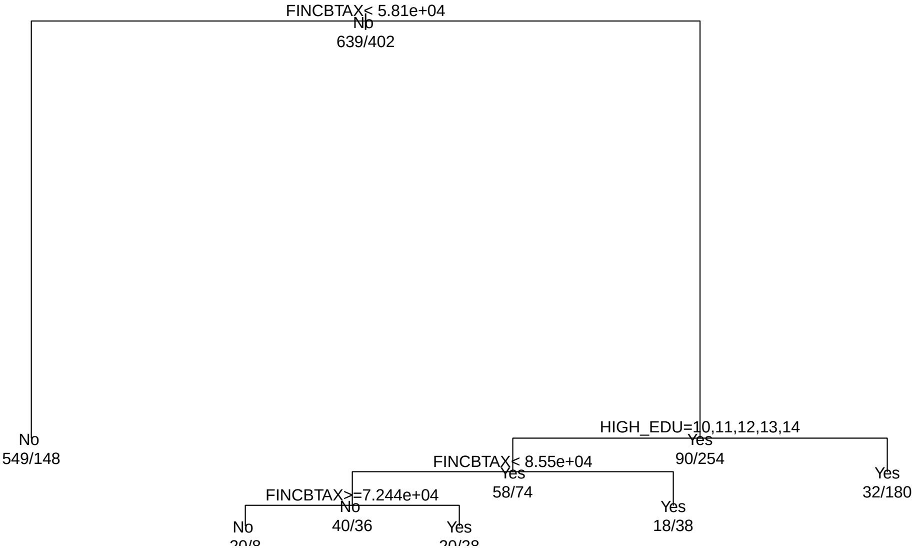
Classification Trees



Classification Trees



Classification Trees



Classification Trees

- At each node, the tree will split the observations into two child nodes based on which split increases the purity.
- What do we mean by purity?
  - The observations are separated more distinctly into the groups of the response variable.
- Several possible measures of purity. Default in `rpart`:

$$\text{Gini} = 1 - \sum_{i=1}^{2} p_i^2$$

where p_i = fraction of records in class i for node t.

Classification Trees with rpart

```r
# Fit the tree
mod_t <- rpart(IRAX_cat ~ ., data = ds)
```

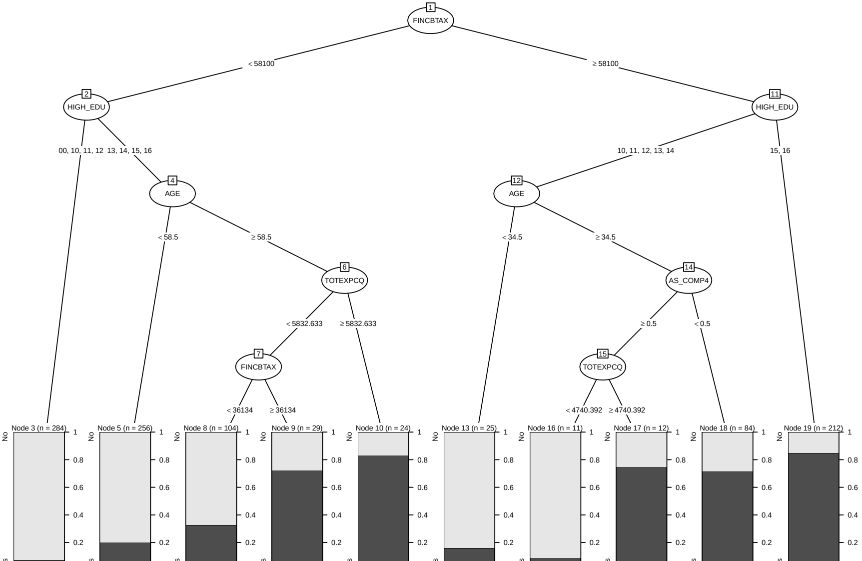
Classification Trees with `rpart`

```r
mod_t
```

```
## n= 1041
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 1041 402 No (0.61383285 0.38616715)
##    2) FINCBTAX< 58100 697 148 No (0.78766141 0.21233859)
##      4) HIGH_EDU=00,10,11,12 284  21 No (0.92605634 0.07394366) *
##      5) HIGH_EDU=13,14,15,16 413 127 No (0.69249395 0.30750605)
##       10) AGE< 58.5 256  52 No (0.79687500 0.20312500) *
##       11) AGE>=58.5 157  75 No (0.52229299 0.47770701)
##         22) TOTEXPCQ< 5832.633 133  55 No (0.58646617 0.41353383)
##           44) FINCBTAX< 36134 104  34 No (0.67307692 0.32692308) *
##           45) FINCBTAX>=36134 29   8 Yes (0.27586207 0.72413793) *
##         23) TOTEXPCQ>=5832.633 24   4 Yes (0.16666667 0.83333333) *
##    3) FINCBTAX>=58100 344  90 Yes (0.26162791 0.73837209)
##      6) HIGH_EDU=10,11,12,13,14 132  58 Yes (0.43939394 0.56060606)
##       12) AGE< 34.5 25   4 No (0.84000000 0.16000000) *
##       13) AGE>=34.5 107  37 Yes (0.34579439 0.65420561)
##         26) AS_COMP4>=0.5 23  10 No (0.56521739 0.43478261)
##           52) TOTEXPCQ< 4740.392 11   1 No (0.90909091 0.09090909) *
##           53) TOTEXPCQ>=4740.392 12   3 Yes (0.25000000 0.75000000) *
##         27) AS_COMP4< 0.5 84  24 Yes (0.28571429 0.71428571) *
##      7) HIGH_EDU=15,16 212  32 Yes (0.15094340 0.84905660) *
```

Visualizing Classification Trees with partykit

```r
plot(as.party(mod_t), gp = gpar(fontsize = 6))
```



Classification Trees with caret

- Tuning parameter?
  - *Complexity parameter*: Value for pruning decision.

```
# Hyper-parameter grid
grd <- data.frame(.cp = (1:50) * 0.01)
# Train Classification Tree
cv_tree <- train(IRAX_cat ~ ., data = ds, method = "rpart",
    tuneGrid = grd, trControl = cv_opts)
cv_tree$bestTune
```

```
##     cp
## 1 0.01
```

Classification Trees with caret

```
cv_tree
```

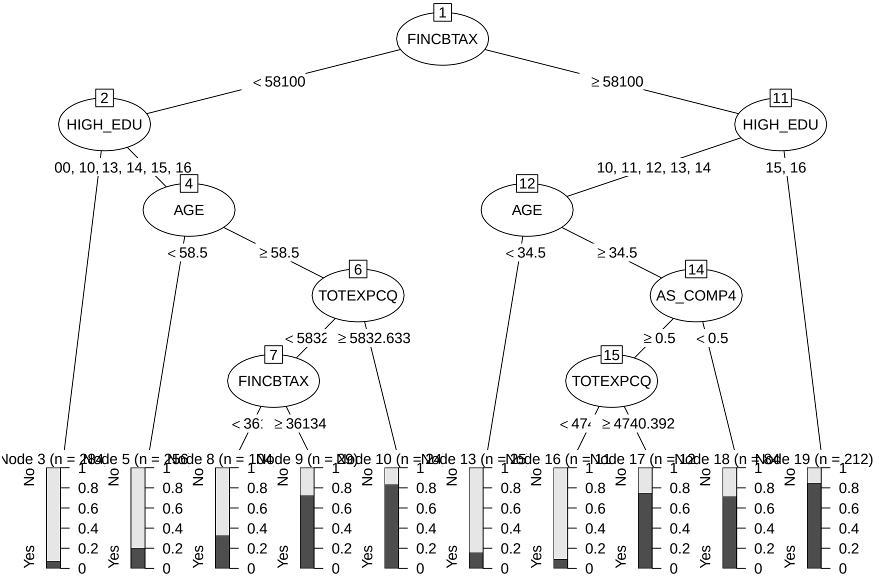
```
## CART
##
## 1041 samples
##   13 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 833, 832, 833, 833, 833
## Resampling results across tuning parameters:
##
##   cp    Accuracy   Kappa
##   0.01  0.7704315  0.5088727
##   0.02  0.7637146  0.4929280
##   0.03  0.7617915  0.4833510
##   0.04  0.7617915  0.4833510
##   0.05  0.7617915  0.4833510
##   0.06  0.7617915  0.4833510
##   0.07  0.7617915  0.4833510
##   0.08  0.7617915  0.4833510
##   0.09  0.7617915  0.4833510
##   0.10  0.7617915  0.4833510
##   0.11  0.7617915  0.4833510
##   0.12  0.7617915  0.4833510
##   0.13  0.7617915  0.4833510
##   0.14  0.7617915  0.4833510
##   0.15  0.7617915  0.4833510
##   0.16  0.7617915  0.4833510
##   0.17  0.7617915  0.4833510
##   0.18  0.7617915  0.4833510
##   0.19  0.7617915  0.4833510
##   0.20  0.7617915  0.4833510
```

Classification Trees

```r
#Fit the tree
mod_t <- rpart(IRAX_cat ~ ., data = ds, control =
                 rpart.control(cp = cv_tree$bestTune))

#Plot the tree
plot(as.party(mod_t), gp = gpar(fontsize = 6))
```

Classification Trees



Classification Final Thoughts

- Logistic and linear SVM often give similar answers.
- If you think a non-linear classification function would help, use the Kernel Trick.
- If you have extraneous predictors, incorporate model selection.
- Classification trees are so interpretable!
  - Often have a lower test accuracy rate than other methods when you have strong quantitative predictors.
  - Random Forests will increase the predictive power!

Final, Final Thoughts

- Many more statistical learning/predictive models that are available to us in the `caret` package.
  - Even more across all of the available R packages!
- Ensemble models tend to have greater predictive power.
- Important to consider your goals when thinking about model form.

Questions

- Congrats! We made it through the course!
- If you have any questions, let me know. I will be here through Saturday!
- I will be leading a food truck tour lunch excursion on Friday from 12:30-2pm.
- Email: kmcconv1@swarthmore.edu

References

Berk, R. A. 2008. *Statistical Learning from a Regression Perspective*. New York: Springer.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. New York: Springer.

James, G., D. Witten, T. Hastie, and R. Tibshirani. 2013. *An Introduction to Statistical Learning*. Vol. 6. Springer.

McConville, K. S., F. J. Breidt, T. C. M. Lee, and G. G. Moisen. 2017. "Model-Assisted Survey Regression Estimation with the Lasso." *Journal of Survey Statistics and Methodology* 5: 131–58.

Shmueli, G. 2010. "To Explain or to Predict?" *Statistical Science* 25 (3). Institute of Mathematical Statistics: 289–310.

Tibshirani, R. 1996. "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58: 267–88.

Zou, H. 2006. "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association* 101: 1418–29.

Zou, Hui, and Trevor Hastie. 2005. "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67 (2). Wiley Online Library: 301–20.