STA 371G: Statistics and Modeling

Some Additional Topics and Model Selection

Mingyuan Zhou
McCombs School of Business
The University of Texas at Austin

http://mingyuanzhou.github.io/STA371G

Time Series: Moving Averages

Moving averages

- A simple and widely used extrapolation method
- A moving average is the average of the past n observations

$$F_{t+1} = \frac{Y_{t-n+1} + \cdots + Y_t}{n}$$

- The span n is the number of past observations used for smoothing
- Choose an appropriate span n to forecast future values

Time Series: Exponential Smoothing

Exponential smoothing forecasts future values using a weighted averages of past observations.

- It puts more weight on the more recent observations
- It requires much less data storage than moving averages
- $F_{t+k} = L_t$ for $k \geq 1$, where

$$L_t = \alpha Y_t + (1 - a) L_{t-1}$$

- Equivalent formula:

$$L_t = L_{t-1} + \alpha(Y_t - L_{t-1}) = \alpha \sum_{k=0}^{\infty} (1 - \alpha)^k Y_{t-k}$$

- Choosing a large α in exponential smoothing is similar to using a small span n in moving averages

Time Series: ARMA

Autoregressive-Moving-Average (ARMA) Model

$$Y_t = \beta_0 + e_t + \beta_1 Y_{t-1} + \cdots + \beta_p Y_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q}$$

- $q = 0$: Autoregressive model of order p
- $p = 0$: moving-average model of order q

Hypothesis Testing

	H_0 is true	H_0 is false
Reject H_0	Type I Error	Correct rejection
Fail to reject H_0	Correct fail	Type II Error

Type I Erorr: incorrect rejection of a TRUE null hypothesis
Type II Erorr: fail to reject a FALSE null hypothesis

Hypothesis Testing

	Not Guilty (Truth)	Guilty (Truth)
Guilty (Decision)	Type I Error	Correct decision
Not Guilty (Decision)	Correct decision	Type II Error

Type I Erorr: innocent people sent to prison

Type II Erorr: criminals set free

Hypothesis Testing for Regression Coefficients

Approximately we have $b_j \sim \mathcal{N}(\beta_j, s_{b_j}^2)$

- One-tailed test:
  Null Hypothesis $H_0 : \beta_j \leq \beta_j^0$
  versus
  Alternative Hypothesis $H_A : \beta_j > \beta_j^0$
- Two-tailed test:
  Null Hypothesis $H_0 : \beta_j = \beta_j^0$
  versus
  Alternative Hypothesis $H_A : \beta_j \neq \beta_j^0$

Which test should we use?

- H_0: stock R is no more risky than the market
- H_0: stock R is as risky as the market

Hypothesis Testing for Regression Coefficients

Approximately we have $b_j \sim \mathcal{N}(\beta_j, s_{b_j}^2)$ in multiple regression

- Hypothesis testing:
  Null Hypothesis $H_0 : \beta_j = \beta_j^0$
  versus
  Alternative Hypothesis $H_A : \beta_j \neq \beta_j^0$
- The probability of committing a Type I Error α is the significance level of the test
- The probability of committing a Type II Error depends on α and the true value of β_j

Hypothesis Testing for Regression Coefficients

- If $\alpha = 0.05$, then

$$P(\textit{Type I Error}) = 0.05$$

$$P(\textit{Type II Error}) \approx P(\beta_j^0 - 2s_{b_j} < b_j < \beta_j^0 + 2s_{b_j})$$

- The power of the test is the probability that we reject H_0 when H_A is true

$$\textit{Power of the Test} = 1 - P(\textit{Type II Error})$$

- Calculate the power of the test for
  - $\beta_j = 1$, $\beta_j^0 = 0$, $s_{b_j} = 1$
  - $\beta_j = 2$, $\beta_j^0 = 0$, $s_{b_j} = 1$
  - $\beta_j = 3$, $\beta_j^0 = 0$, $s_{b_j} = 1$

Hypothesis Testing for Regression Coefficients

The p-value is the probability of observing a test statistic at least as extreme as the observed value $|t| = \frac{|b_j - \beta_j|}{s_{b_j}}$, assuming the null hypothesis is true.

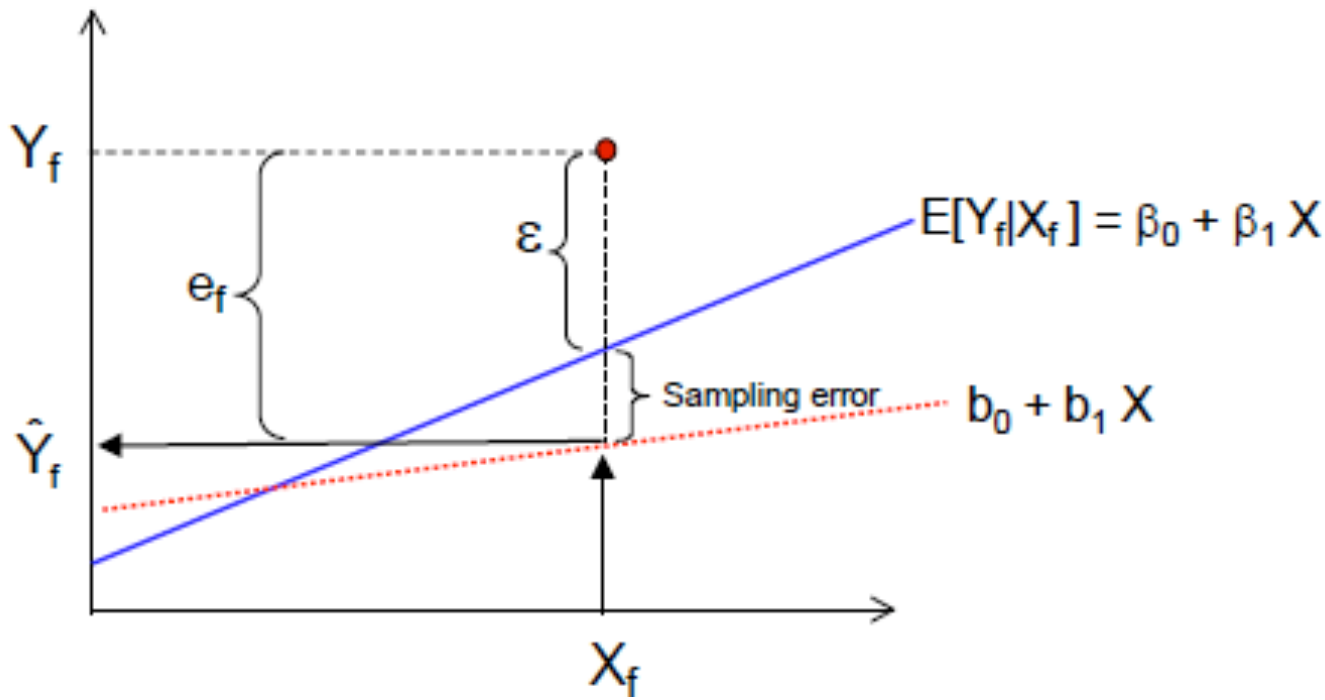
- p-value $= P(>|t| \mid H_0$ is true$)$
- We reject the null hypothesis if p-value $< \alpha$
- We reject the null hypothesis if t is large

Reject $H_0 : \beta_j = \beta_j^0$ if the hypothesized value β_j^0 is not within the $(1 - \alpha)$ confidence interval of β_j.

Prediction Errors

In simple linear regression, we predict Y_f with X_f, our prediction error is

$$\begin{aligned} e_f = Y_f - \hat{Y}_f &= Y_f - b_0 - b_1 X_f \\ &= (\beta_0 + \beta_1 X_f + \epsilon) - (b_0 + b_1 X_f) \\ &= (\beta_0 - b_0) + (\beta_1 - b_1) X_f + \epsilon \end{aligned}$$



Prediction Errors

The standard error of $\hat{Y}_f$ is

$$s\sqrt{1+\frac{1}{n}+\frac{(X_f-\bar{X})^2}{\sum_{i=1}^{n}(X_i-\bar{X})^2}} \approx s$$

The standard error of the mean of $\hat{Y}_f$ is

$$s\sqrt{\frac{1}{n}+\frac{(X_f-\bar{X})^2}{\sum_{i=1}^{n}(X_i-\bar{X})^2}} \approx \frac{s}{\sqrt{n}}$$

A large predictive error variance (high uncertainty) comes from

- Large s (i.e., large ε's).
- Small n (not enough data).
- Small s_x (not enough observed spread in covariates).
- Large difference between X_f and $\bar{X}$.

Prediction Errors

In multiple regression, we also have two sources of errors in $\hat{Y}_f = b_0 + b_1 X_{f1} + \cdots + b_p X_{fp}$

$$\begin{aligned} e_f &= Y_f - \hat{Y}_f \\ &= Y_f - b_0 - b_1 X_{f1} - \cdots - b_p X_{fp} \\ &= (\beta_0 - b_0) + (\beta_1 - b_1) X_{f1} + \cdots + (\beta_p - b_p) X_{fp} + \epsilon \end{aligned}$$

Prediction Errors

Given X_f, an example R code to calculate the 95% prediction interval for $\hat{Y}_f$ and the 95% confidence interval for $\mathbb{E}[\hat{Y}_f]$:

```
data = read.csv("Profits.csv", header=TRUE)
attach(data)
Fit = lm(PROFIT~ RISK+RD)
new = data.frame(RISK=7,RD=76)
predict(Fit, new, interval = "prediction")
predict(Fit, new, interval = "confidence")
```

Model Selection: Model Building Process

When building a regression model remember that simplicity is your friend... smaller models are easier to interpret and have fewer unknown parameters to be estimated.

Keep in mind that every additional parameter represents a cost!!

The first step of every model building exercise is the selection of the the universe of variables to be potentially used. This task is entirely solved through your experience and context specific knowledge...

- Think carefully about the problem
- Consult subject matter research and experts
- Avoid the mistake of selecting too many variables

Model Building Process

With a universe of variables in hand, the goal now is to select the model. Why not include all the variables in?

Big models tend to over-fit and find features that are specific to the data in hand... ie, not generalizable relationships.
The results are bad predictions and bad science!

In addition, bigger models have more parameters and potentially more uncertainty about everything we are trying to learn... (check the beer and weight example!)

We need a strategy to build a model in ways that accounts for the trade-off between fitting the data and the uncertainty associated with the model

Out-of-Sample Prediction

One idea is to focus on the model's ability to predict... How do we evaluate a forecasting model? Make predictions!

Basic Idea: We want to use the model to forecast outcomes for observations we have not seen before.

- Use the data to create a prediction problem.
- See how our candidate models perform.

We'll use most of the data for training the model,
and the left over part for validating the model.

Out-of-Sample Prediction

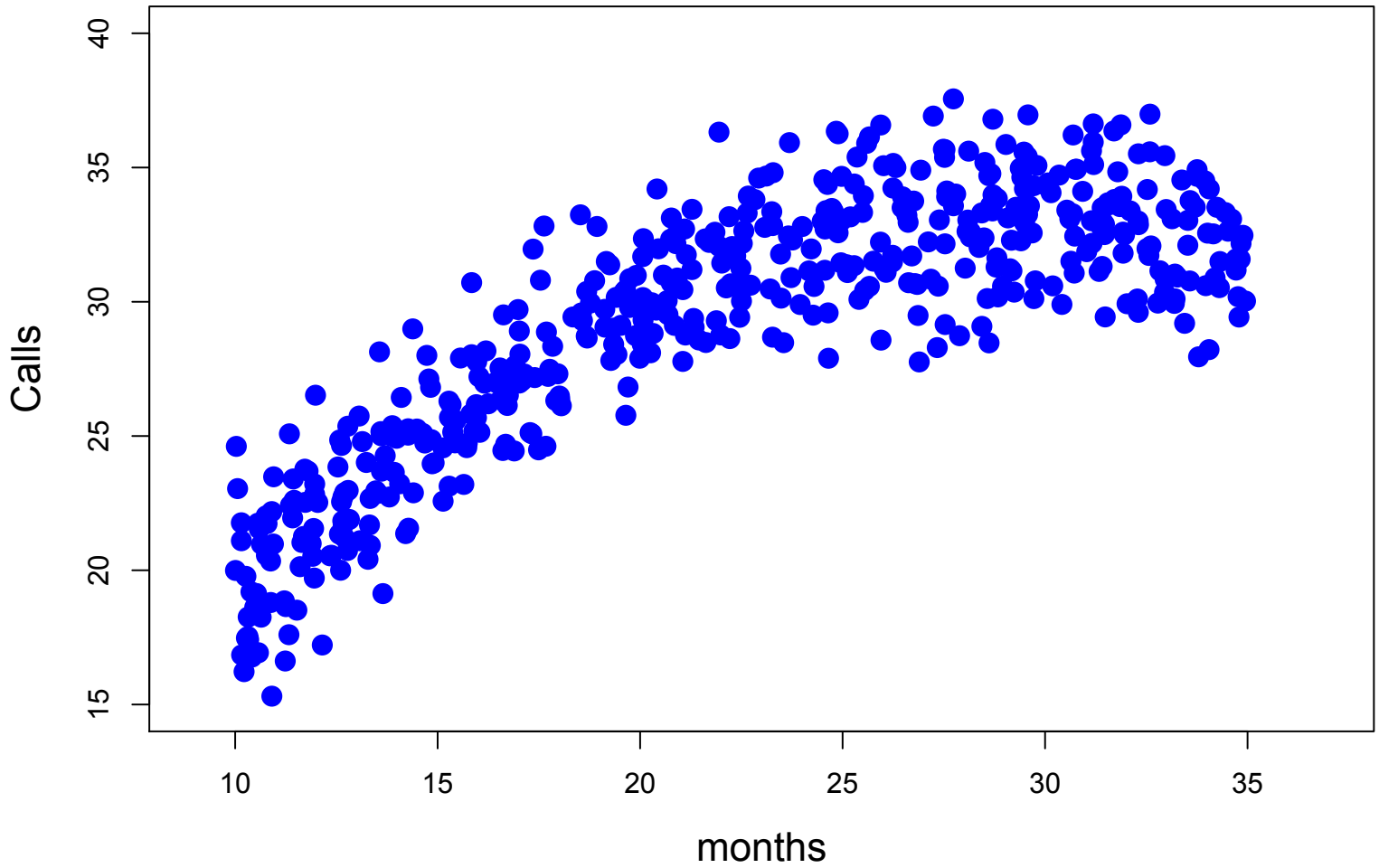
In a cross-validation scheme, you fit a bunch of models to most of the data (training sample) and choose the model that performed the best on the rest (left-out sample).

- Fit the model on the training data
- Use the model to predict $\hat{Y}_j$ values for all of the N_{LO} left-out data points
- Calculate the Mean Square Error for these predictions

$$MSE = \frac{1}{N_{LO}} \sum_{j=1}^{N_{LO}} (Y_j - \hat{Y}_j)^2$$

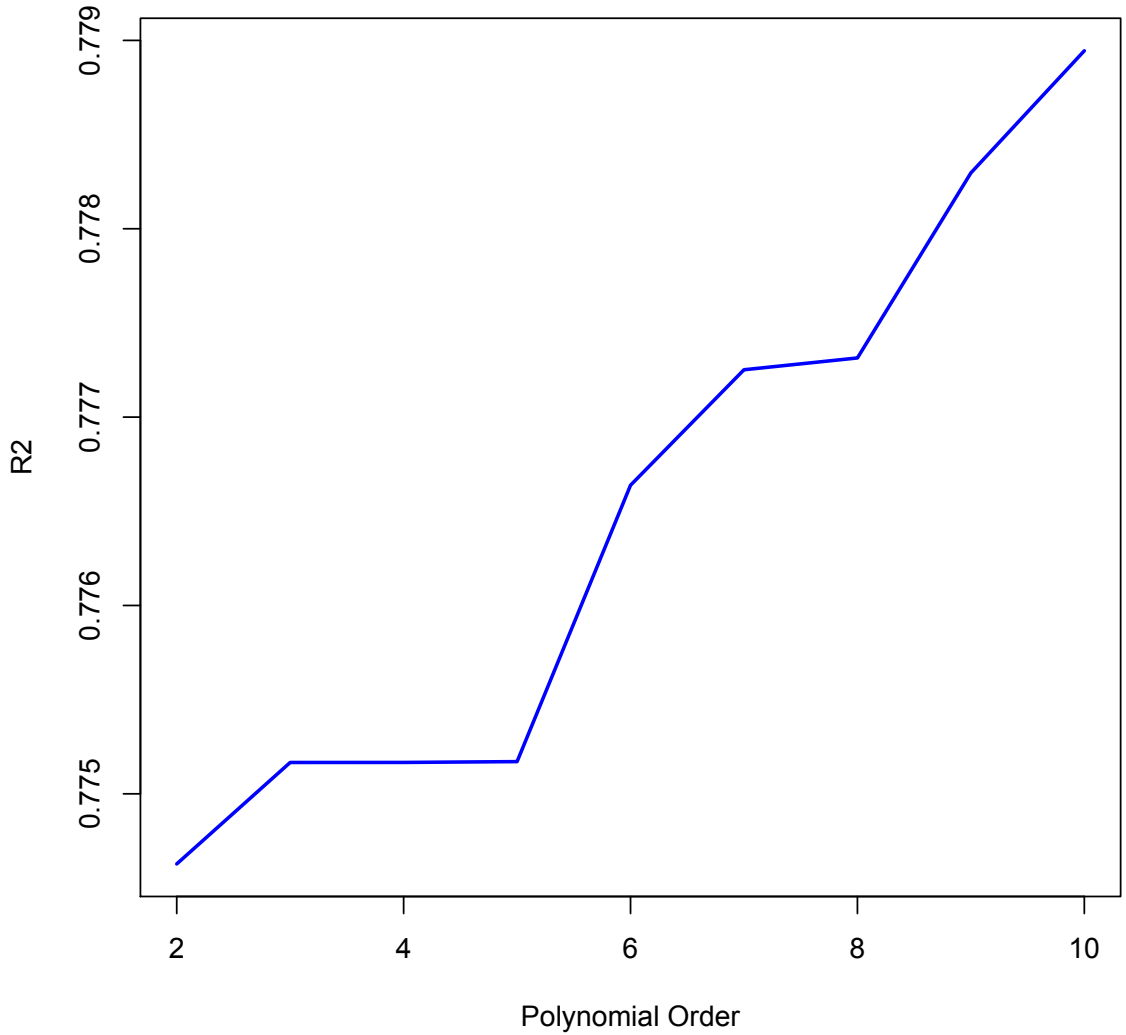
Example

To illustrate the potential problems of "over-fitting" the data, let's look again at the Telemarketing example... let's look at multiple polynomial terms...



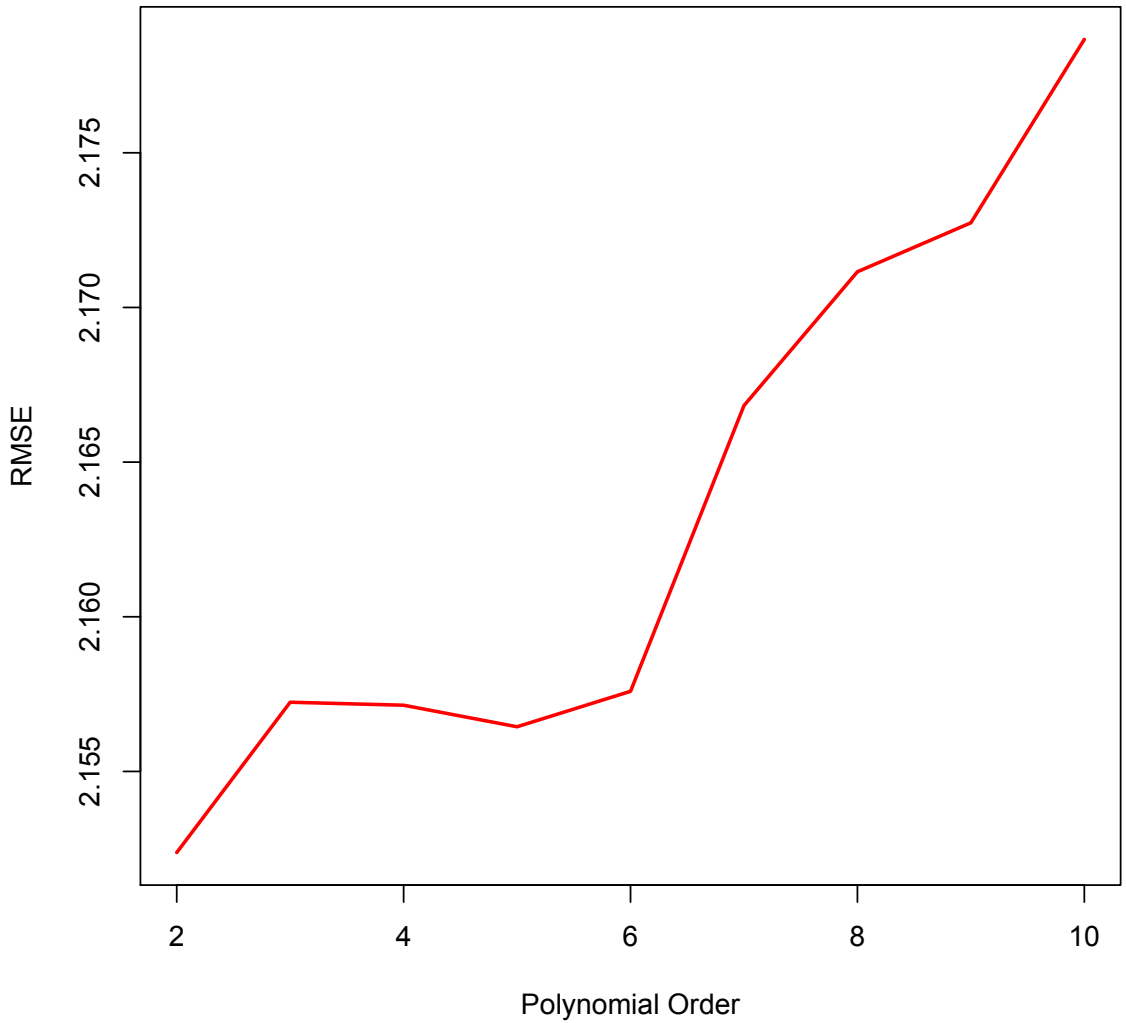
Example

Let's evaluate the fit of each model by their R^2 (on the training data)



Example

How about the MSE?? (on the left-out data)



Information Criteria for Model Selection

Another way to evaluate a model is to use Information Criteria metrics which attempt to quantify how well our model would have predicted the data (regardless of what you've estimated for the β_j's).

$$AIC = n \log(SSE/n) + 2p$$

A good alternative is the BIC: Bayes Information Criterion, which is based on a "Bayesian" philosophy of statistics.

$$BIC = n \log(SSE/n) + p \log(n)$$

You want to choose the model that leads to minimum BIC.

BIC for Model Selection

One (very!) nice thing about the BIC is that you can interpret it in terms of model probabilities.

Given a list of possible models $\{M_1, M_2, \ldots, M_R\}$, the probability that model i is correct is

$$P(M_i) \approx \frac{e^{-\frac{1}{2}BIC(M_i)}}{\sum_{r=1}^{R} e^{-\frac{1}{2}BIC(M_r)}} = \frac{e^{-\frac{1}{2}[BIC(M_i)-BIC_{min}]}}{\sum_{r=1}^{R} e^{-\frac{1}{2}[BIC(M_r)-BIC_{min}]}}$$

(Subtract $BIC_{min} = \min\{BIC(M_1) \ldots BIC(M_R)\}$ for numerical stability.)

BIC for Model Selection

Thus BIC is an alternative to testing for comparing models.

- It is easy to calculate.
- You are able to evaluate model probabilities.
- There are no "multiple testing" type worries.
- It generally leads to more simple models than F-tests.

As with testing, you need to narrow down your options before comparing models. What if there are too many possibilities?

If $p = 20$, we have $2^{20} = 1,048,576$ possible models! Choosing the best subset seems infeasible.

Adjusted R Square, R_a^2

The R^2 is a measure of the goodness of fit, but it always increases as we include more predictor variables.

The adjusted R^2, R_a^2, is also a measure of the goodness of fit.

- $$R_a^2 = 1 - \frac{SSE/(n-p-1)}{SST/(n-1)}$$
- R_a^2 can be used to compare models with different number of predictor variables

Stepwise Regression

One computational approach to build a regression model step-by-step is "stepwise regression" There are 3 options:

- Forward: adds one variable at the time until no remaining variable makes a significant contribution (or meet a certain criteria... could be out of sample prediction)
- Backward: starts will all possible variables and removes one at the time until further deletions would do more harm than good
- Stepwise: just like the forward procedure but allows for deletions at each step

Auto MPG Example

Initial model:
$MPG = \beta_0 + \beta_1 weight + \beta_2 horsepower + \beta_3 displacement +$
$\beta_4 acceleration + \beta_5 cylinders + \beta_6 year + \beta_7 year^2 + \beta_8 origin1 +$
$\beta_9 origin2 + \epsilon$

Backward elemination:

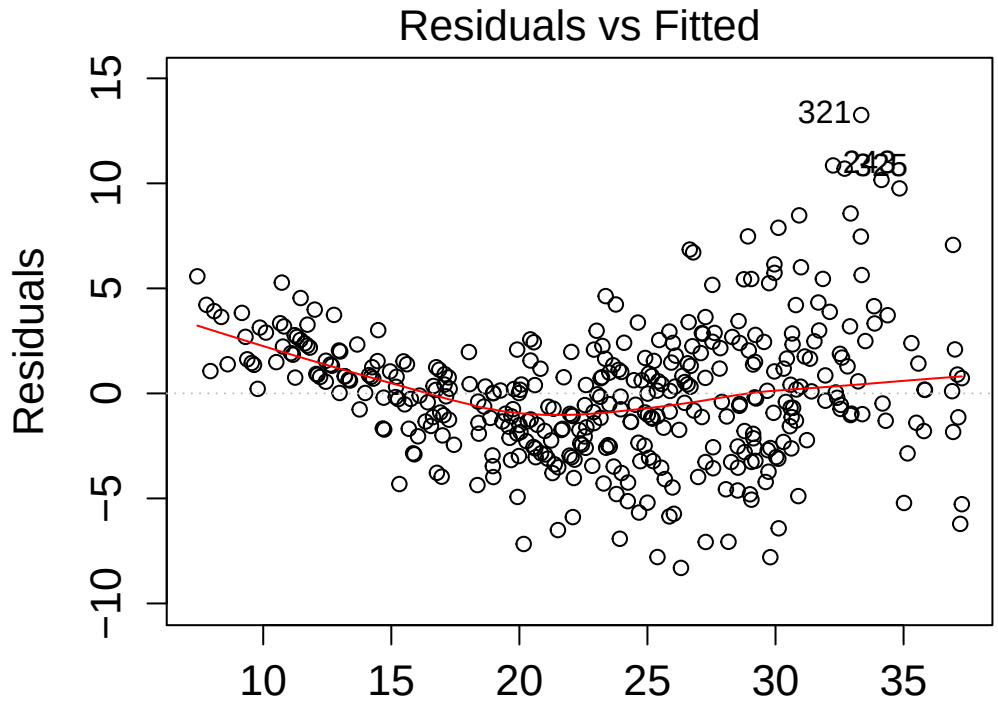
- Step1: delete *origin2*
- Step 2: delete *acceleration*
- Step 3: delete *cylinders*

Selected model:
$MPG = \beta_0 + \beta_1 weight + \beta_2 horsepower + \beta_3 displacement +$
$\beta_6 year + \beta_7 year^2 + \beta_8 origin1 + \epsilon$

Auto MPG Example

Residual plot for:

$MPG = \beta_0 + \beta_1 weight + \beta_2 horsepower + \beta_3 displacement + \beta_6 year + \beta_7 year^2 + \beta_8 origin1 + \epsilon$



Does this plot look good?

Auto MPG Example

Initial model:
$\log(MPG) = \beta_0 + \beta_1 + \log(weight) + \beta_2 \log(horsepower) +$
$\beta_3 \log(displacement) + \beta_4 \log(acceleration) + \beta_5 cylinders +$
$\beta_6 year + \beta_7 year^2 + \beta_8 origin1 + \beta_9 origin2 + \epsilon$

Backward elemination:

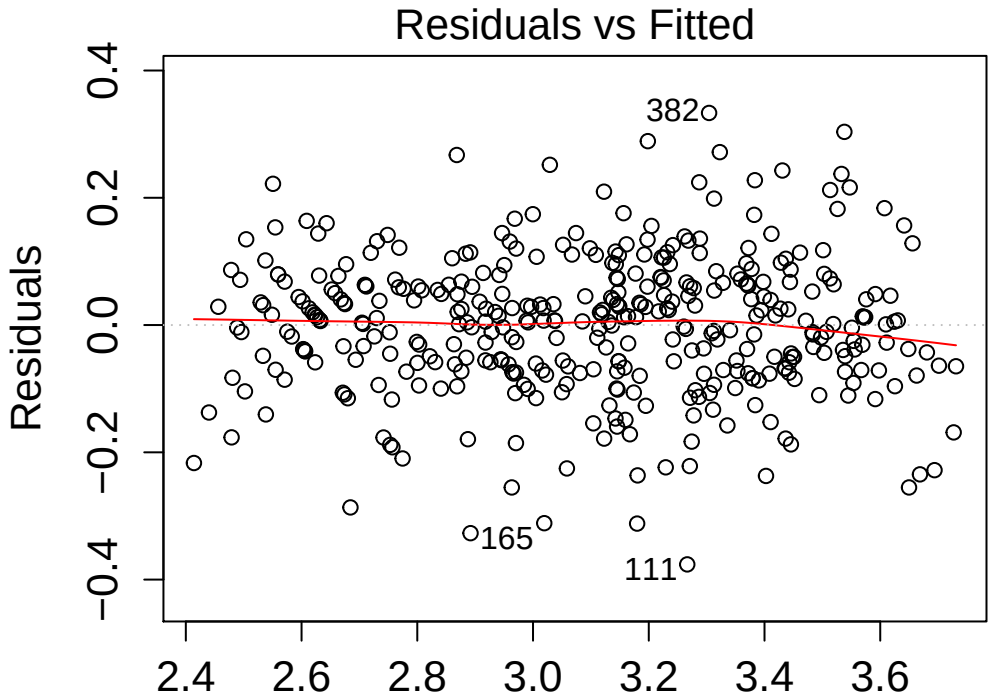
- Step1: delete $\log(displacement)$
- Step 2: delete $origin2$

Selected model:
$\log(MPG) = \beta_0 + \beta_1 + \log(weight) + \beta_2 \log(horsepower) +$
$\beta_4 \log(acceleration) + \beta_5 cylinders + \beta_6 year + \beta_7 year^2 + \beta_8 origin1 + \epsilon$

Auto MPG Example

Residual plot for:

$$\log(MPG) = \beta_0 + \beta_1 + \log(weight) + \beta_2 \log(horsepower) + \beta_4 \log(acceleration) + \beta_5 cylinders + \beta_6 year + \beta_7 year^2 + \beta_8 origin1 + \epsilon$$



How about this one?

Diagnose the residuals!

LASSO

The LASSO is a shrinkage method that performs automatic selection. Yet another alternative... has similar properties as stepwise regression but it is more automatic... R does it for you! The LASSO solves the following problem:

$$\arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(Y_i - X_i'\beta\right)^2 + \lambda|\beta| \right\}$$

- Coefficients can be set exactly to zero (automatic model selection)
- Very efficient computational method
- λ is often chosen via CV

One informal but very useful idea to put it all together...

I like to build models from the bottom, up...

- Set aside a set of points to be your validating set (if dataset large enought)
- Working on the training data, add one variable at the time deciding which one to add based on some criteria:
  1. larger increases in R^2 while significant
  2. larger reduction in MSE while significant
  3. BIC, etc...
- at every step, carefully analyze the output and check the residuals!
- Stop when no additional variable produces a "significant" improvement
- Always make sure you understand what the model is doing in the specific context of your problem