IEOR E4650 Business Analytics

Session 6: Model Selection and Assessment

Spring 2018

Prof. Adam Elmachtoub

Outline

- Properties of good linear regression models
- Choosing the best model (Model Selection)
  - Traditional Approaches
  - Train-Validation-Test Approach
  - Cross-Validation
- Estimating the true error rate of a model (Model Assessment)

Properties of a Good Linear Regression

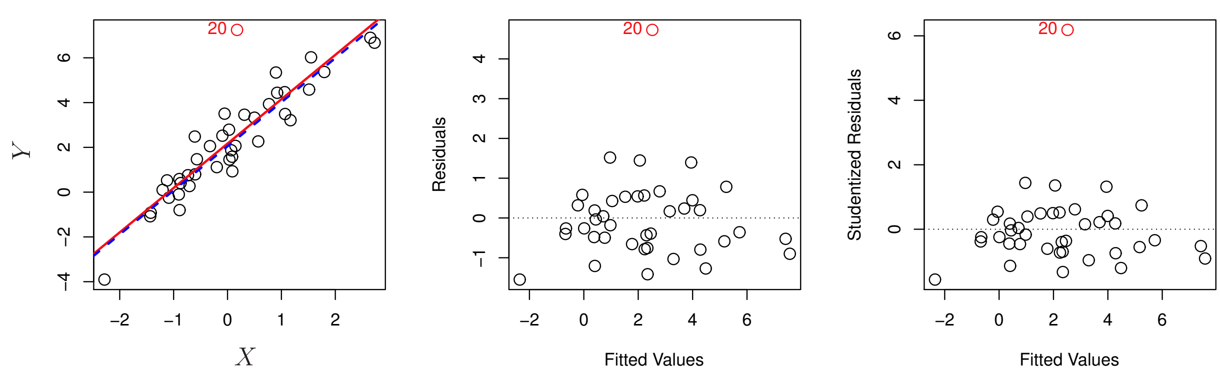
Actually, 6 potential problems to look for in a linear regression model:

- Outliers
- High-leverage points
- Collinearity
- Correlation of residuals
- Non-linearity in the residuals
- Non-constant variance in the residuals

Let's look at these problems and their solutions..

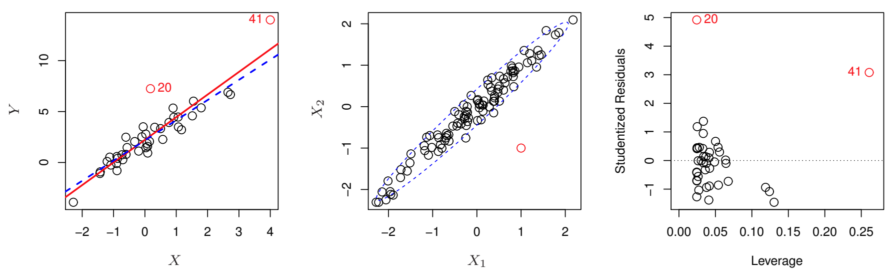
Outliers

- An outlier is a point where the prediction $\widehat{y}_i$ is very or unusually far from the observation y_i.
- Outliers occur due to inaccurate data collection. (Remove or fix!)
- Outliers distort model and skew the RSS and MSE
- Be warned.. some outliers may be true data points and require special consideration



High-leverage points

- A point is high-leverage if x_i takes on an unusual value, i.e., the features in x_i fall way outside the normal range
- Can distort model and skew the RSS and MSE
- Be warned.. high-leverage points may require special consideration

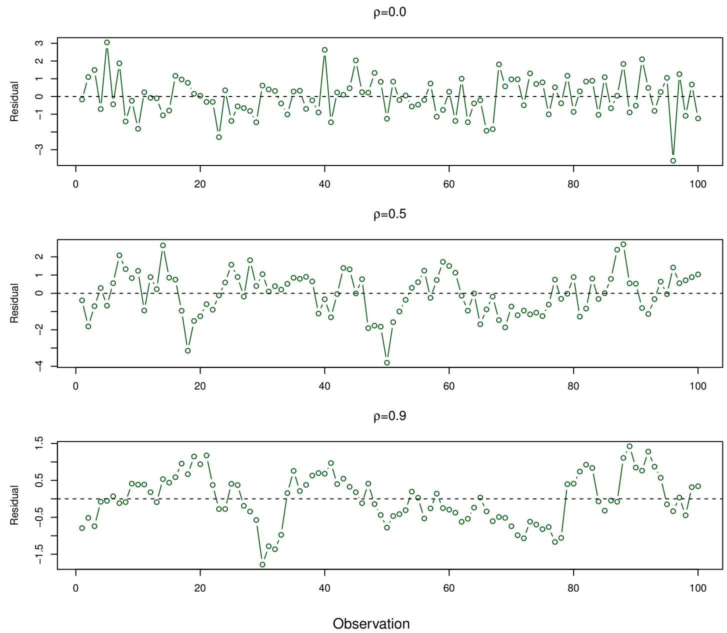


- In the middle graph, the red point has a normal value of X_1 and X_2, but together the point is high-leverage

Collinearity

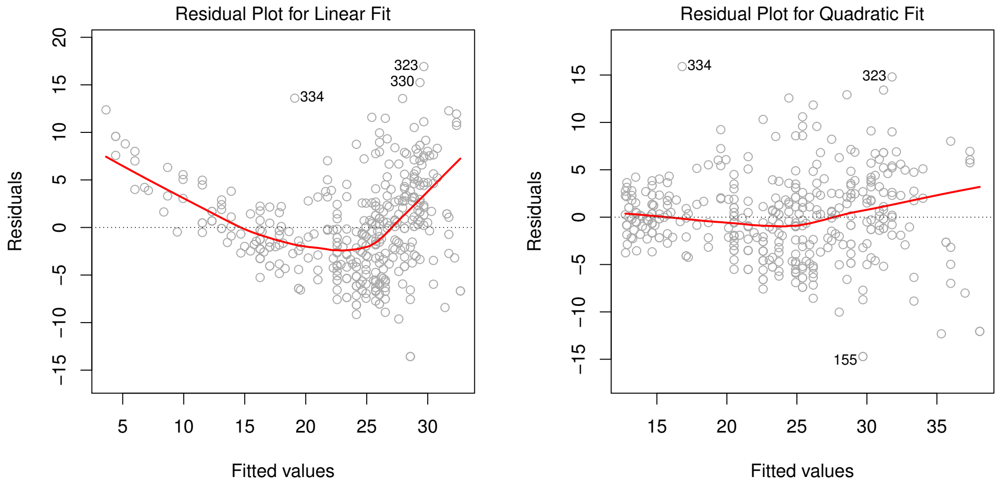
- Collinearity is when predictor variables are very correlated to one another.
- If two independent variables are highly correlated (positive or negative), then only one of them should be needed in the model.
- Combining correlated independent variables or using only one makes the model smaller and the coefficients more significant, without sacrificing performance.
- This is why we only use $K - 1$ dummy variables for a K category qualitative variable

Correlation of residuals



- Most likely not using an important time-related independent variable!

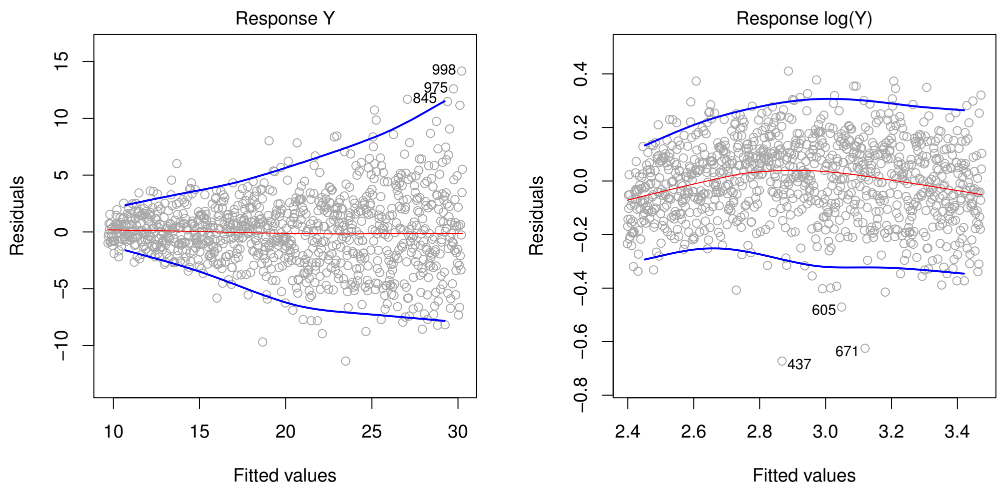
Non-linearity in the residuals



- Most likely not using an important non-linear independent variable, i.e., $\log X$, X^2, or $\sqrt{X}$

Non-constant variance in the residuals

- Also known as heteroscedasticity



- Most likely we should transform dependent variable, i.e., predict $\log Y, Y^2$, or $\sqrt{Y}$

Model Selection and Assessment

- **Model Selection**: Start with several candidate models or parameters for a model
  - Select best model (or parameter) using *training data*, $\approx 75\%$ of the data
  - We think it will have the lowest test error rate
- **Model Assessment**: Evaluating the true performance (error rate) of selected model
  - Measure performance on a *test data*,$\approx 25\%$ of the data
  - Compute the test error rate of the selected model on the test data
- Today we focus on using $MSE = \frac{1}{n}RSS = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ as the error rate we would like to minimize

Traditional Approach

- First, fit all candidate models on the training data.
- Second, compute one of RSE, $adjusted\ R^2$, (AIC, BIC, C_p) for all models on training data.
  - Each of these performance measures combines the number of predictors used with the MSE on the training dataset
- Third, *select* model with lowest performance measure.
- Fourth, *assess* MSE of selected model on test data.

Train-Validation-Test Approach

- First, fit different linear models on 50% of the data, i.e., $2/3$ of the training data.
- Second, compute MSE of each model on the remaining part of the training data, i.e. the *validation data*, which is 25% of the overall data
- Third, *select* model with lowest MSE.
- Fourth, retrain the selected model on the entire training dataset, i.e., 75% of the data.
- Fifth, *assess* MSE of final model on test data, i.e., 25% of the data.

Limitations of the Validation Approach

- The Validation Set Approach has two important limitations:
  - The models are originally only fit on half the data. Statistical methods perform worse with fewer observations.
  - The results are highly dependent on which part of the training data was used to fit the models.
- What can we do to mitigate the problem?

k-Fold Cross Validation Approach

- First, randomly divide training data into k folds (parts) of equal size.
- Second, for each model we do the following:
  - For *each* fold $j = 1, \ldots, k$
    - Fit the model on the all the training data except fold j.
    - Compute the MSE of the fitted model on fold j and call this MSE_j.
  - Use the average of these to estimate performance of the model, i.e., $MSE = \frac{1}{k} \sum_{j=1}^{k} MSE_j$.
- Third, *select* model with lowest MSE.
- Fourth, retrain the selected model on the entire training dataset.
- Fifth, *assess* MSE of final model on test data.

Choosing k

- Best practice is to choose $k = 5$ or $k = 10$
- If $k = |\text{training data}|$, this is called leave-one-out cross validation (LOOCV)
  - This requires fitting the model many times which is computationally expensive
  - For linear regression, it can actually be done efficiently
- Bias-variance tradeoff in choosing k
  - When k is large, the datasets we train on have a lot of overlap (more variance) but are larger (less bias)
  - When k is large, our estimates of the test error are good on average due to using a lot of data (less bias) but are highly correlated (more variance)
  - When k is small, the datasets do not overlap much (less variance) but are smaller (more bias)

Examples of CV

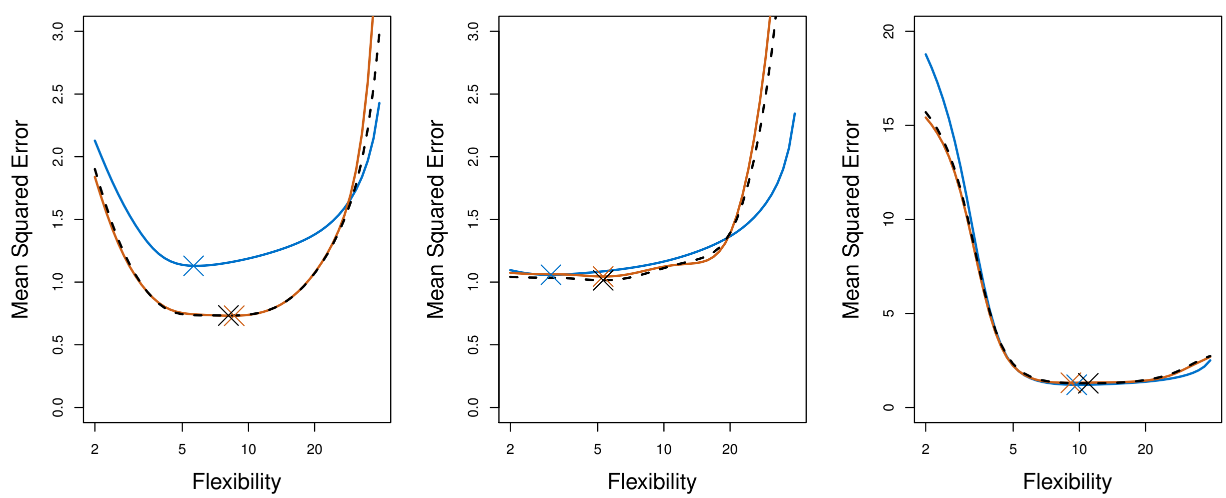


Figure: Blue: True test error. Orange: 10-fold CV error. Dotted black: LOOCV error

CV with Auto data

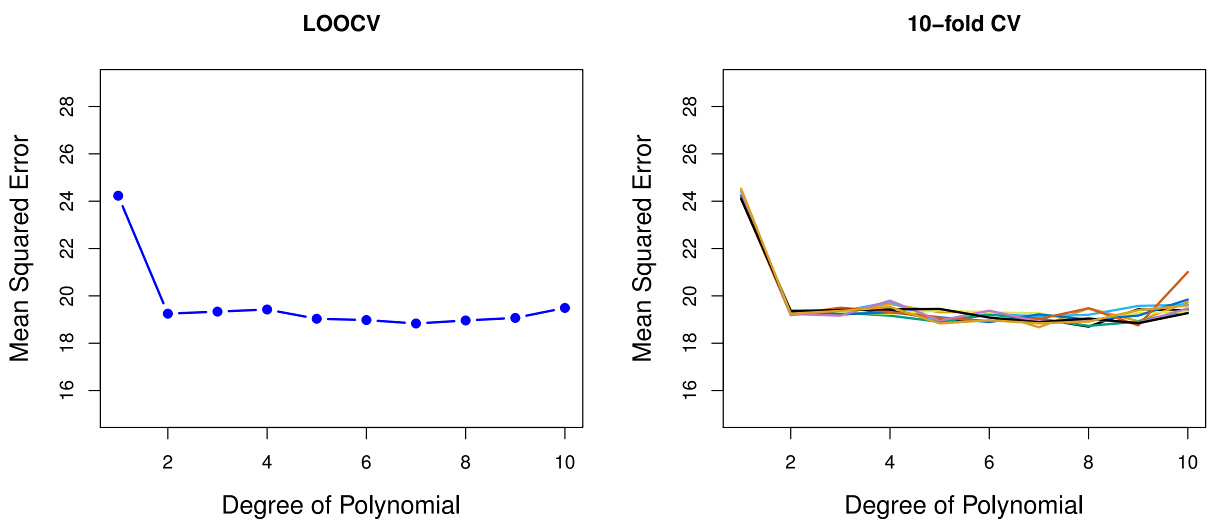


Figure: Left: LOOCV error. Right: 10-fold CV error, done with 9 different partitions of the data

LOOCV with Auto Data

- LOOCV for the Auto Data regressing mpg vs horsepower

```
library(boot)
glm.fit=glm(mpg~horsepower,data=Auto)
cv.err=cv.glm(Auto,glm.fit)
cv.err$delta[1]
[1] 24.23
```

- LOOC for polynomials up to degree 5

```
cv.error=rep(0,5)
for (i in 1:5){glm.fit=glm(mpg~poly(horsepower,i),
data=Auto)
cv.error[i]=cv.glm(Auto,glm.fit)$delta[1]}
cv.error
[1] 24.23 19.25 19.33 19.42 19.03
```

k-fold CV with Auto

- Use 10-fold CV for polynomials of up to degree 10.

```
set.seed(1)
> cv.error.10=rep(0,10)
> for (i in 1:10){
+   glm.fit=glm(mpg~poly(horsepower,i),data=Auto)
+   cv.error.10[i]=cv.glm(Auto,glm.fit,K=10)$delta[1]}
cv.error.10
 [1] 24.11 19.24 19.29 19.46 19.26 18.86 18.84 18.78 19.66 19.54
```

- Lets try it again with another seed.

```
set.seed(17)
cv.error.10=rep(0,10)
for (i in 1:10){
  glm.fit=glm(mpg~poly(horsepower,i),data=Auto)
  cv.error.10[i]=cv.glm(Auto,glm.fit,K=10)$delta[1]}
cv.error.10
[1] 24.21 19.19 19.31 19.34 18.88 19.02 18.90 19.71 18.95 19.50
```


Next time

- Automated ways to discover good linear regression models when number of independent variables, p, is large
- Stepwise regression, subset selection, ridge regression, LASSO

[1]Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani