MS&E 226: “Small” Data
Lecture 7: Model selection (v3)

Ramesh Johari
ramesh.johari@stanford.edu
Model selection
Overview

*Model selection* refers to the process of comparing a variety of models (using, e.g., model complexity scores, cross validation, or validation set error.

In this lecture we describe a few strategies for model selection, then compare them in the context of a couple of real datasets.

Throughout, *our goal is prediction*. Therefore we compare models through estimates of their generalization error ("model scores"): e.g., training error (sum of squared residuals), $R^2$, $C_p$, AIC, BIC, cross validation, validation set error, etc.
Model selection: Goals

There are two types of qualitative goals in model selection:

- **Minimize prediction error.** This is our primary goal in this lecture.
- **Interpretability.** We will have more to say about this in the next unit of the class.

Both goals often lead to a desire for “parsimony”: roughly, a desire for smaller models over more complex models.
Subset selection

Suppose we have $p$ covariates available, and we want to find which subset to include in a linear regression fit by OLS.

One approach is:

- For each subset $S \subset \{1, \ldots, p\}$, compute the OLS solution with just the subset of covariates in $S$.
- Select the subset that minimizes the chosen model score.

Implemented in R via the `leaps` package (with $C_p$ or $R^2$ as model score).

*Problem:* Computational complexity scales exponentially with number of covariates.
Forward stepwise selection

Another approach:

1. Start with $S = \emptyset$.
2. Add the single covariate to $S$ that leads to greatest reduction in model score.
3. Repeat steps 1-2.

Implemented in R via the `step` function (with AIC or related model scores).

The computational complexity of this is only quadratic in the number of covariates (and often much less).
Backward stepwise selection

Another approach:

1. Start with $S = \{1, \ldots, p\}$.

2. Delete the single covariate from $S$ that leads to greatest reduction in model score.

3. Repeat steps 1-2.

Also implemented via `step` in R.

Also quadratic computational complexity, though it can be worse than forward stepwise selection when there are many covariates. (In fact, backward stepwise selection can’t be used when $n \leq p$ — why?)
Stepwise selection: A warning

When applying stepwise regression, you are vulnerable to the same issues discussed earlier:

► The same data is being used repeatedly to make selection decisions.
► In general, this will lead to downward biased estimates of your prediction error.

The train-validate-test methodology can mitigate this somewhat, by providing an objective comparison.

To reiterate: Practitioners often fail to properly isolate test data during the model building phase!
Regularization

Lasso minimizes:

\[ \text{SSE} + \lambda \sum_{j=1}^{p} |\hat{\beta}_j| \]

where \( \lambda > 0 \).

Ridge regression minimizes:

\[ \text{SSE} + \lambda \sum_{j=1}^{p} |\hat{\beta}_j|^2. \]

where \( \lambda > 0 \).

[LO regularization: minimize \( \text{SSE} + \lambda \times \# \text{ of nonzero coefficients} \) ]
Both lasso and ridge regression are “shrinkage” methods for model selection:

- Relative to OLS, both lasso and ridge regression will yield coefficients $\hat{\beta}$ that have “shrunk” towards zero.
- The most explanatory covariates are the ones that will be retained.
- Lasso typically yields a much smaller subset of nonzero coefficients than ridge regression or OLS (i.e., fewer nonzero entries in $\hat{\beta}$).

To use these for model selection, tune $\lambda$ to minimize the desired model score.
Intuition for lasso

Why does lasso tend to “truncate” more coefficients at zero than ridge?

\[
Y = \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \quad X = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad n = 2, \quad p = 2
\]

OLS:

\[
\hat{Y} = \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2
\]

\[
\text{minimize} \quad (Y_1 - \hat{\beta}_1 X_1)^2 + (Y_2 - \hat{\beta}_2 X_2)^2
\]

\[
\hat{\beta}_1 = 2, \quad \hat{\beta}_2 = 3
\]

Lasso(\(\lambda\)):

\[
\text{minimize} \quad (Y_1 - \hat{\beta}_1 X_1)^2 + (Y_2 - \hat{\beta}_2 X_2)^2 + \lambda \hat{\beta}_1 + \lambda \hat{\beta}_2
\]

\[
\Rightarrow \quad \hat{\beta}_1 = \left[ 2 - \frac{\lambda}{2} \right]^+, \quad \hat{\beta}_2 = \left[ 3 - \frac{\lambda}{2} \right]^+
\]

MSE (\(\lambda\)):

\[
\text{minimize} \quad (Y_1 - \hat{\beta}_1 X_1)^2 + (Y_2 - \hat{\beta}_2 X_2)^2 + \lambda \hat{\beta}_1^2 + \lambda \hat{\beta}_2^2
\]

\[
\hat{\beta}_1 = \frac{4}{2\lambda + 2}, \quad \hat{\beta}_2 = \frac{6}{2\lambda + 2}
\]
Intuition for lasso

- OLS

\[ \beta \to \gamma \]

\[ \hat{\beta} \]

\( (0, 1) \)

\( \lambda = 0 \)

\( \lambda = 4 \)

\( \hat{\beta}^2 + \hat{\beta}_2^2 = 4 \)

\( \hat{\beta}_1 + \hat{\beta}_2 = 1 \)

\( \hat{\beta}_1 + \hat{\beta}_2 = 2 \)

\( \text{LASSO PATH AS } \lambda \to 0 \)

\( \text{RIDGE PATH AS } \lambda \to 0 \)
Example: Crime dataset
Crime dataset

From:
http://lib.stat.cmu.edu/DASL/Datafiles/USCrime.html

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Crime rate: # of offenses reported to police per million population</td>
</tr>
<tr>
<td>Age</td>
<td>The number of males of age 14-24 per 1000 population</td>
</tr>
<tr>
<td>S</td>
<td>Indicator variable for Southern states (0 = No, 1 = Yes)</td>
</tr>
<tr>
<td>Ed</td>
<td>Mean # of years of schooling x 10 for persons of age 25 or older</td>
</tr>
<tr>
<td>Ex0</td>
<td>1960 per capita expenditure on police by state and local government</td>
</tr>
<tr>
<td>Ex1</td>
<td>1959 per capita expenditure on police by state and local government</td>
</tr>
<tr>
<td>LF</td>
<td>Labor force participation rate per 1000 civilian urban males age 14-24</td>
</tr>
</tbody>
</table>
# Crime dataset

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>The number of males per 1000 females</td>
</tr>
<tr>
<td>N</td>
<td>State population size in hundred thousands</td>
</tr>
<tr>
<td>NW</td>
<td>The number of non-whites per 1000 population</td>
</tr>
<tr>
<td>U1</td>
<td>Unemployment rate of urban males per 1000 of age 14-24</td>
</tr>
<tr>
<td>U2</td>
<td>Unemployment rate of urban males per 1000 of age 35-39</td>
</tr>
<tr>
<td>W</td>
<td>Median value of transferable goods and assets or family income in tens of $</td>
</tr>
<tr>
<td>X</td>
<td>The number of families per 1000 earning below 1/2 the median income</td>
</tr>
</tbody>
</table>
Forward stepwise regression

```r
> fm.lower = lm(data = crime.df, R ~ 1)
> fm.upper = lm(data = crime.df, R ~ .)
> step(fm.lower,
      scope = list(lower = fm.lower,
                   upper = fm.upper),
      direction = "forward")
```
## Forward stepwise regression: Step 1

Start: AIC=344.58

\[ R \sim 1 \]

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum of Sq</th>
<th>RSS</th>
<th>AIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Ex0</td>
<td>1</td>
<td>32533</td>
<td>36276</td>
<td>316.49</td>
</tr>
<tr>
<td>+ Ex1</td>
<td>1</td>
<td>30586</td>
<td>38223</td>
<td>318.95</td>
</tr>
<tr>
<td>+ W</td>
<td>1</td>
<td>13402</td>
<td>55408</td>
<td>336.40</td>
</tr>
<tr>
<td>+ N</td>
<td>1</td>
<td>7837</td>
<td>60973</td>
<td>340.90</td>
</tr>
<tr>
<td>+ Ed</td>
<td>1</td>
<td>7171</td>
<td>61638</td>
<td>341.41</td>
</tr>
<tr>
<td>+ M</td>
<td>1</td>
<td>3149</td>
<td>65661</td>
<td>344.38</td>
</tr>
<tr>
<td>&lt;none&gt;</td>
<td></td>
<td>68809</td>
<td></td>
<td>344.58</td>
</tr>
<tr>
<td>+ LF</td>
<td>1</td>
<td>2454</td>
<td>66355</td>
<td>344.87</td>
</tr>
<tr>
<td>+ X</td>
<td>1</td>
<td>2205</td>
<td>66604</td>
<td>345.05</td>
</tr>
<tr>
<td>+ U2</td>
<td>1</td>
<td>2164</td>
<td>66646</td>
<td>345.08</td>
</tr>
<tr>
<td>+ S</td>
<td>1</td>
<td>565</td>
<td>68244</td>
<td>346.19</td>
</tr>
<tr>
<td>+ Age</td>
<td>1</td>
<td>551</td>
<td>68258</td>
<td>346.20</td>
</tr>
<tr>
<td>+ U1</td>
<td>1</td>
<td>175</td>
<td>68634</td>
<td>346.46</td>
</tr>
<tr>
<td>+ NW</td>
<td>1</td>
<td>73</td>
<td>68736</td>
<td>346.53</td>
</tr>
</tbody>
</table>
Forward stepwise regression: Step 2

Step: AIC=316.49
R ~ Ex0

<table>
<thead>
<tr>
<th>Df</th>
<th>Sum of Sq</th>
<th>RSS</th>
<th>AIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ X</td>
<td>1</td>
<td>7398.2</td>
<td>28878</td>
</tr>
<tr>
<td>+ Age</td>
<td>1</td>
<td>6167.4</td>
<td>30109</td>
</tr>
<tr>
<td>+ M</td>
<td>1</td>
<td>2505.2</td>
<td>33771</td>
</tr>
<tr>
<td>+ NW</td>
<td>1</td>
<td>2324.3</td>
<td>33952</td>
</tr>
<tr>
<td>+ S</td>
<td>1</td>
<td>2191.0</td>
<td>34085</td>
</tr>
<tr>
<td>+ W</td>
<td>1</td>
<td>1808.7</td>
<td>34468</td>
</tr>
<tr>
<td>&lt;none&gt;</td>
<td></td>
<td>36276</td>
<td>316.49</td>
</tr>
<tr>
<td>+ Ex1</td>
<td>1</td>
<td>1461.7</td>
<td>34815</td>
</tr>
<tr>
<td>+ LF</td>
<td>1</td>
<td>774.8</td>
<td>35501</td>
</tr>
<tr>
<td>+ U2</td>
<td>1</td>
<td>178.5</td>
<td>36098</td>
</tr>
<tr>
<td>+ N</td>
<td>1</td>
<td>56.7</td>
<td>36220</td>
</tr>
<tr>
<td>+ U1</td>
<td>1</td>
<td>28.8</td>
<td>36247</td>
</tr>
<tr>
<td>+ Ed</td>
<td>1</td>
<td>7.7</td>
<td>36269</td>
</tr>
</tbody>
</table>
Forward stepwise regression: Final output

Call:
`lm(formula = R ~ Ex0 + X +
    Ed + Age + U2 + W, data = crime.df)`

Coefficients:
(Intercept)    Ex0     X    Ed    Age    U2
 -618.5028   1.0507  0.8236  1.8179  1.1252  0.8282
   W
  0.1596

Backward stepwise regression yields the same result. Is this an interpretable model?
Example: Baseball hitters
Baseball hitters

Data taken from *An Introduction to Statistical Learning*.

Consists of statistics and salaries for 263 Major League Baseball players.

We use this dataset to:

- Develop the train-test method
- Apply lasso and ridge regression
- Compare and interpret the results

We’ll use the `glmnet` package for this example.
glmnet uses matrices rather than data frames for model building:

```r
> library(ISLR)
> library(glmnet)

> data(Hitters)
> hitters.df = subset(na.omit(Hitters))

> X = model.matrix(Salary ~ 0 , hitters.df)
> Y = hitters.df$Salary
```
Training vs. test set

Here is a simple way to construct training and test sets from the single dataset:

```r
train.ind = sample(nrow(X), round(nrow(X)/2))
X.train = X[train.ind,]
X.test = X[-train.ind,]
Y.train = Y[train.ind]
Y.test = Y[-train.ind]
```
Ridge and lasso

Building a lasso model:

```r
> lambdas = 10^seq(-2, 3.4, 0.1)
> fm.lasso = glmnet(X.train,  
    Y.train, alpha = 1,  
    lambda = lambdas, thresh = 1e-12)
```

Setting alpha = 0 gives ridge regression. Make predictions as follows at \( \lambda = \text{lam} \):

```r
> mean( (Y.test -  
    predict(fm.lasso, s = lam, newx = X.test))^2 )
```
Results

What is happening to lasso?
Lasso coefficients

Using `plot(fm.lasso, xvar="lambda")`: