

Machine Learning for Microeconometrics

Part 1: Selection and cross validation

A. Colin Cameron
Univ.of California - Davis

April 2024

Course Outline

- **Part 1: Variable selection and cross validation**
- **2.** Shrinkage methods
 - ▶ ridge, lasso, elastic net
- **3.** ML for causal inference using lasso
 - ▶ OLS with many controls, IV with many instruments
- **4.** Other methods for prediction
 - ▶ nonparametric regression, principal components, splines
 - ▶ neural networks
 - ▶ regression trees, random forests, bagging, boosting
- **5.** More ML for causal inference
 - ▶ ATE with heterogeneous effects and many controls.
- **6.** Classification and unsupervised learning
 - ▶ classification (categorical y) and unsupervised learning (no y).

1. Introduction

- These slides consider how to determine how well a model predicts with a penalty for model complexity
 - ▶ subsequent slides present various ML methods for prediction.
- MUS2: Chapter 28.2 “Machine Learning for Prediction and Inference” in A. Colin Cameron and Pravin K. Trivedi (2022), *Microeconometrics using Stata*, Second edition, forthcoming.
- For more detail on machine learning I used the two books given in the references
 - ▶ ISL2: *Introduction to Statistical Learning, R and Python editions*.
 - ▶ ESL: *Elements of Statistical Learning*.
- While most ML code is in R and Python, these slides use Stata.
 - ▶ Stata 16 plus some add-ons covers the material in these slides.

Overview

- 1 Introduction
- 2 Model Selection using Predictive Ability
 - 1 Generated data example
 - 2 Mean squared error
 - 3 Information criteria and related penalty measures
 - 4 Cross validation overview
 - 5 Single-split cross validation
 - 6 K-fold cross-validation
 - 7 Leave-one-out cross validation
 - 8 Stepwise selection and best subsets selection
 - 9 Selection using statistical significance

Terminology

- We consider two types of data sets.
- 1. **training data set** (or **estimation sample**)
 - ▶ used to fit a model.
- 2. **test data set** (or **hold-out sample** or **validation set**)
 - ▶ additional data used to determine how good is the model fit
 - ▶ a test observation (\mathbf{x}_0, y_0) is a previously unseen observation.

2.1 Generated Data Example

- We consider just OLS
 - ▶ but the model selection methods carry over to any other machine learner.
- These slides use Stata
 - ▶ most machine learning code is initially done in Python or R.
- Generated data: $n = 40$
- Three correlated regressors.
 - ▶
$$\begin{bmatrix} x_{1i} \\ x_{2i} \\ x_{3i} \end{bmatrix} \sim N \left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{bmatrix} \right)$$
- But only x_1 determines y
 - ▶ $y = 2 + x_{1i} + u_i$ where $u_i \sim N(0, 3^2)$.

```
* Generate three correlated variables (rho = 0.5) and y linear only in x1  
clear  
quietly set obs 40  
set seed 12345  
matrix MU = (0,0,0)  
scalar rho = 0.5  
matrix SIGMA = (1,rho,rho \ rho,1,rho \ rho,rho,1)  
drawnorm x1 x2 x3, means(MU) cov(SIGMA)  
generate y = 2 + 1*x1 + rnormal(0,3)
```

```
. * Summarize data
. summarize
```

| Variable | Obs | Mean | Std. Dev. | Min | Max |
|----------|-----|----------|-----------|-----------|----------|
| x1 | 40 | .3337951 | .8986718 | -1.099225 | 2.754746 |
| x2 | 40 | .1257017 | .9422221 | -2.081086 | 2.770161 |
| x3 | 40 | .0712341 | 1.034616 | -1.676141 | 2.931045 |
| y | 40 | 3.107987 | 3.400129 | -3.542646 | 10.60979 |

```
. correlate
(obs=40)
```

| | x1 | x2 | x3 | y |
|----|--------|--------|--------|--------|
| x1 | 1.0000 | | | |
| x2 | 0.5077 | 1.0000 | | |
| x3 | 0.4281 | 0.2786 | 1.0000 | |
| y | 0.4740 | 0.3370 | 0.2046 | 1.0000 |

Fitted OLS regression

- As expected only x_1 is statistically significant at 5%
 - though due to randomness this is not guaranteed.

```
. * OLS regression of y on x1-x3
. regress y x1 x2 x3, vce(robust)
```

```
Linear regression                               Number of obs   =           40
                                                F(3, 36)        =           4.91
                                                Prob > F        =          0.0058
                                                R-squared       =          0.2373
                                                Root MSE      =          3.0907
```

| | Coef. | Robust Std. Err. | t | P> t | [95% Conf. Interval] | |
|-------|-----------|---------------------|-------|-------|----------------------|----------|
| x1 | 1.555582 | .5006152 | 3.11 | 0.004 | .5402873 | 2.570877 |
| x2 | .4707111 | .5251826 | 0.90 | 0.376 | -.5944086 | 1.535831 |
| x3 | -.0256025 | .6009393 | -0.04 | 0.966 | -1.244364 | 1.193159 |
| _cons | 2.531396 | .5377607 | 4.71 | 0.000 | 1.440766 | 3.622025 |

2.2 Mean squared error

- Assume (y, \mathbf{x}) have common c.d.f. $F(\cdot)$ in both training and test data.
- We wish to predict y given $\mathbf{x} = (x_1, \dots, x_p)$
 - ▶ define $f(\mathbf{x}) = E[y|\mathbf{x}]$ for a single observation
- A **training data set** d yields prediction rule $\hat{f}(\mathbf{x})$
 - ▶ e.g. for OLS $\hat{y} = \hat{f}(\mathbf{x}) = \mathbf{x}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$ (where \mathbf{X} is $N \times p$).
- Using **test data** we predict y at point \mathbf{x}_0 using $\hat{y}_0 = \hat{f}(\mathbf{x}_0)$.
 - ▶ e.g. for OLS $\hat{y}_0 = \mathbf{x}_0(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$.
- For regression consider **squared error loss** $(y - \hat{y})^2$.
- We wish to estimate the **true prediction error**
 - ▶ $E_F[(y_0 - \hat{y}_0)^2]$ for **test data set** point $(\mathbf{x}_0, y_0) \sim F$.
- Some methods adapt to other loss functions
 - ▶ e.g. absolute error loss and log-likelihood loss
 - ▶ e.g. loss function for classification is $\mathbf{1}(y \neq \hat{y})$.

Models overfit in sample

- We want to estimate the **true prediction error**
 - ▶ $E_F[(y_0 - \hat{y}_0)^2]$ for **test data set** point $(\mathbf{x}_0, y_0) \sim F$.
- The obvious criterion is in-sample **mean squared error**
 - ▶ $\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$ where MSE = mean squared error.
- Problem: in-sample MSE **under-estimates** the true prediction error
 - ▶ Intuitively models “overfit” within sample.
- Example: suppose $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{u}$
 - ▶ then $\hat{\mathbf{u}} = (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_{OLS}) = (\mathbf{I} - \mathbf{M})\mathbf{u}$ where $\mathbf{M} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$
 - ★ so on average $|\hat{u}_i| < |u_i|$ (OLS residual < true unknown error)
 - ▶ and use $\hat{\sigma}^2 = s^2 = \frac{1}{n-k} \sum_{i=1}^n (y_i - \hat{y}_i)^2$ and not $\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$
- Two solutions:
 - ▶ penalize for overfitting e.g. \bar{R}^2 , AIC, BIC, C_p
 - ▶ use out-of-estimation sample prediction (cross-validation).

2.3 Information criteria and related penalty measures

- Two standard measures for general parametric model are
 - ▶ Akaike's information criterion
 - ★ $AIC = -2 \ln L + 2k$
 - ▶ BIC: Bayesian information criterion
 - ★ $BIC = -2 \ln L + (\ln n) \times k$
- Models with smaller AIC and BIC are preferred.
- AIC has a small penalty for larger model size
 - ▶ for nested models selects larger model if $-2\Delta \ln L > 2\Delta k$
 - ★ whereas LR test = $-2\Delta \ln L$ of size α requires $-2\Delta \ln L > \chi_{\alpha}^2(k)$.
- BIC has a larger penalty.

Details: AIC and BIC for OLS

- For classical regression assuming i.i.d. normal errors
 - ▶ $\ln L = -\frac{n}{2} \ln 2\pi - \frac{n}{2} \ln \sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mathbf{x}'_i \boldsymbol{\beta})^2$
- Different programs can calculate a different AIC and BIC
 - ▶ using the following variations.
- Constants such as $-\frac{n}{2} \ln 2\pi$ are often dropped.
- The sign sign can be reversed (so AIC and BIC are positive).
- Econometricians use $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2 = \text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \mathbf{x}'_i \hat{\boldsymbol{\beta}})^2$
 - ▶ then $\text{AIC} = -\frac{n}{2} \ln 2\pi - \frac{n}{2} \ln \hat{\sigma}^2 - \frac{n}{2} + 2k$.
- Machine learners use $\hat{\boldsymbol{\beta}}$ and $\tilde{\sigma}^2 = \frac{1}{n-p} \sum_{i=1}^n (y_i - \mathbf{x}'_i \tilde{\boldsymbol{\beta}}_p)^2$
 - ▶ where $\tilde{\boldsymbol{\beta}}_p$ is obtained from OLS in the largest model under consideration that has p regressors including intercept
- Also a finite sample correction is
 - ▶ $\text{AICC} = \text{AIC} + 2(K+1)(K+2)/(N-K-2)$.

Details: More measures for OLS

- For OLS a standard measure is \bar{R}^2 (adjusted R^2)
 - ▶ $\bar{R}^2 = 1 - \frac{\frac{1}{n-k} \sum_{i=1}^n (y_i - \hat{y}_i)^2}{\frac{1}{n-1} \sum_{i=1}^n (y_i - \bar{y})^2}$ (whereas $R^2 = 1 - \frac{\frac{1}{n-1} \sum_{i=1}^n (y_i - \hat{y}_i)^2}{\frac{1}{n-1} \sum_{i=1}^n (y_i - \bar{y})^2}$)
 - ▶ \bar{R}^2 has a small penalty for model complexity
 - ★ \bar{R}^2 favors the larger nested model if the subset test $F > 1$.
- Some machine learners also use Mallows C_p measure
 - ▶ $C_p = (n \times \text{MSE} / \tilde{\sigma}^2) - n + 2k$
 - ★ $\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \mathbf{x}'_i \hat{\boldsymbol{\beta}})^2$ and $\tilde{\sigma}^2 = \frac{1}{n-p} \sum_{i=1}^n (y_i - \tilde{y}_i)^2$
 - ▶ and some replace p with “effective degrees of freedom”

$$p = \frac{1}{\tilde{\sigma}^2} \sum_{i=1}^n \widehat{\text{Cov}}(\hat{\mu}_i, y_i).$$
- Note that for linear regression AIC, BIC, AICC and C_p are designed for models with homoskedastic errors
 - ▶ more generally AIC and BIC are for correctly specified likelihood function.

Penalty measure measures for OLS

| Measure | Penalty for model size |
|---|------------------------|
| $MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$ | None |
| $R^2 = 1 - \frac{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}{\frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})^2}$ | None |
| $\bar{R}^2 = 1 - \frac{\frac{1}{n-k} \sum_{i=1}^n (y_i - \hat{y}_i)^2}{\frac{1}{n-1} \sum_{i=1}^n (y_i - \bar{y})^2}$ | Too small |
| $AIC = \text{const} - \frac{n}{2} \ln MSE - 2k$ | Too small |
| $BIC = \text{const} - \frac{n}{2} \ln MSE - \ln nk$ | Possibly too large |

Example of penalty measures

- We will consider all 8 possible models based on x_1 , x_2 and x_3 .

```
. * Regressor lists for all possible models
. global xlist1

. global xlist2 x1

. global xlist3 x2

. global xlist4 x3

. global xlist5 x1 x2

. global xlist6 x2 x3

. global xlist7 x1 x3

. global xlist8 x1 x2 x3

.

. * Full sample estimates with AIC, BIC, Cp, R2adj penalties
. quietly regress y $xlist8

. scalar s2full = e(rmse)^2 // Needed for Mallows Cp
```


Example of penalty measures (continued)

- Manually get various measures. All but MSE favor model with just x_1
 - Note that MSE decreases as x_2 and x_3 added to model with x_1 .

```
. forvalues k = 1/8 {
2.   quietly regress y ${xlist`k'}
3.   scalar mse`k' = e(rss)/e(N)
4.   scalar r2adj`k' = e(r2_a)
5.   scalar aic`k' = -2*e(ll) + 2*e(rank)
6.   scalar bic`k' = -2*e(ll) + e(rank)*ln(e(N))
7.   scalar cp`k' = e(rss)/s2full - e(N) + 2*e(rank)
8.   display "Model " "${xlist`k'}" "_col(15) " MSE=" %8.5f mse`k'   ///
>     " R2adj=" %6.3f r2adj`k' " AIC=" %7.2f aic`k'   ///
>     " BIC=" %7.2f bic`k' " Cp=" %6.3f cp`k'
9. }
Model           MSE=11.27186 R2adj= 0.000 AIC= 212.41 BIC= 214.10 Cp= 9.199
Model x1        MSE= 8.73891 R2adj= 0.204 AIC= 204.23 BIC= 207.60 Cp= 0.593
Model x2        MSE= 9.99158 R2adj= 0.090 AIC= 209.58 BIC= 212.96 Cp= 5.838
Model x3        MSE=10.80016 R2adj= 0.017 AIC= 212.70 BIC= 216.08 Cp= 9.224
Model x1 x2     MSE= 8.59796 R2adj= 0.196 AIC= 205.58 BIC= 210.64 Cp= 2.002
Model x2 x3     MSE= 9.84189 R2adj= 0.080 AIC= 210.98 BIC= 216.05 Cp= 7.211
Model x1 x3     MSE= 8.73887 R2adj= 0.183 AIC= 206.23 BIC= 211.29 Cp= 2.592
Model x1 x2 x3  MSE= 8.59740 R2adj= 0.174 AIC= 207.57 BIC= 214.33 Cp= 4.000
```

2.4 Cross-validation overview

- Basic idea: assess models on basis of out-of-sample fit.
- Begin with single-split validation for pedagogical reasons.
- Then present K-fold cross-validation
 - ▶ used extensively in machine learning
 - ▶ generalizes to loss functions other than MSE such as $\frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$
 - ▶ though more computation than e.g. AIC, BIC.
- And present leave-one-out cross validation
 - ▶ widely used for local fit in nonparametric regression.
- Given a selected model the final estimation is on the full dataset
 - ▶ usual inference ignores the data-mining.

2.5 Single split cross validation

- Randomly **divide available data into two parts**
 - ▶ 1. model is fit on training set
 - ▶ 2. MSE is computed for predictions in validation set.
- Example: estimate all 8 possible models with x_1 , x_2 and x_3
 - ▶ for each model estimate on the training set to get $\hat{\beta}'$ s, predict on the validation set and compute MSE in the validation set.
 - ▶ choose the model with the lowest validation set MSE.
- Problems with this single-split validation
 - ▶ 1. Lose precision due to smaller training set
 - ★ so may actually overestimate the test error rate (MSE) of the model.
 - ▶ 2. Results depend a lot on the particular single split.

Single split cross validation example

- Use `splitsample` command

- ▶ training sample ($n = 32$) and test sample ($n = 8$)

```
. * splitsample command for single split validation
. * training data (80% of sample) and test data (20%)
. splitsample, split(1 4) values(0 1) generate(dtrain) rseed(10101)

. tabulate dtrain
```

| dtrain | Freq. | Percent | Cum. |
|--------|-------|---------|--------|
| 0 | 8 | 20.00 | 20.00 |
| 1 | 32 | 80.00 | 100.00 |
| Total | 40 | 100.00 | |

- In following code note that Stata `predict` command gives prediction for all possible observations
 - ▶ not just the subsample used in estimation.

Single split validation example (continued)

- In-sample (training sample) MSE minimized with x_1, x_2, x_3 .
- Out-of-sample (test sample) MSE minimized with only x_1 .

```

. * Single split validation - training and test MSE for the 8 possible models
. forvalues k = 1/8 {
2.   qui reg y ${xlist`k'} if dtrain==1
3.   qui predict y`k'hat
4.   qui gen y`k'errorsq = (y`k'hat - y)^2
5.   qui sum y`k'errorsq if dtrain == 1
6.   scalar mse`k'train = r(mean)
7.   qui sum y`k'errorsq if dtrain == 0
8.   qui scalar mse`k'test = r(mean)
9.   display "Model " "${xlist`k'}" _col(16) ///
>     " Training MSE = " %7.3f mse`k'train " Test MSE = " %7.3f mse`k'test
10. }
Model           Training MSE = 10.124 Test MSE = 16.280
Model x1        Training MSE = 7.478 Test MSE = 13.871
Model x2        Training MSE = 8.840 Test MSE = 14.803
Model x3        Training MSE = 9.658 Test MSE = 15.565
Model x1 x2     Training MSE = 7.288 Test MSE = 13.973
Model x2 x3     Training MSE = 8.668 Test MSE = 14.674
Model x1 x3     Training MSE = 7.474 Test MSE = 13.892
Model x1 x2 x3  Training MSE = 7.288 Test MSE = 13.980

```

2.6 K-fold cross validation

- K -fold cross-validation
 - ▶ splits data into K mutually exclusive folds of roughly equal size
 - ▶ for $j = 1, \dots, K$ fit using all folds but fold j and predict on fold j
 - ▶ standard choices are $K = 5$ and $K = 10$.
- The following shows case $K = 5$

| | Fit on folds | Test on fold |
|---------|--------------|--------------|
| $j = 1$ | 2,3,4,5 | 1 |
| $j = 2$ | 1,3,4,5 | 2 |
| $j = 3$ | 1,2,4,5 | 3 |
| $j = 4$ | 1,2,3,5 | 4 |
| $j = 5$ | 1,2,3,4 | 5 |

- The K -fold CV estimate is

$$CV_K = \frac{1}{K} \sum_{j=1}^K \text{MSE}_{(j)}, \text{ where } \text{MSE}_{(j)} \text{ is the MSE for fold } j.$$

splitsample command

- Create mutually exclusive subsamples of pre-specified size
 - ▶ `splitsample` command
- Example is five equal-sized subsamples
 - ▶ always set the seed
 - ▶ default numbers the subsamples 1, 2, 3, ...

```
. * Split sample into five equal size parts using splitsample commands
. splitsample, nsplit(5) generate(snum) rseed(10101)

. tabulate snum
```

| snum | Freq. | Percent | Cum. |
|-------|-------|---------|--------|
| 1 | 8 | 20.00 | 20.00 |
| 2 | 8 | 20.00 | 40.00 |
| 3 | 8 | 20.00 | 60.00 |
| 4 | 8 | 20.00 | 80.00 |
| 5 | 8 | 20.00 | 100.00 |
| Total | 40 | 100.00 | |

K-fold cross validation example

- MSE in each test fold (of 8 observations) for model with all three regressors.

```
. * Five-fold cross validation example for model with all regressors
. splitsample, nsplit(5) generate(foldnum) rseed(10101)

. matrix allmses = J(5,1,.)

. capture drop y*hat y*errorsq

. forvalues i = 1/5 {
2.   qui reg y x1 x2 x3 if foldnum != `i'
3.   qui predict y`i'hat
4.   qui gen y`i'errorsq = (y`i'hat - y)^2
5.   qui sum y`i'errorsq if foldnum == `i'
6.   matrix allmses[`i',1] = r(mean)
7. }
```

```
. matrix list allmses

allmses[5,1]
      c1
r1  13.980321
r2   6.4997357
r3   9.3623792
r4   6.413401
r5   12.23958
```


Average MSE and standard deviation across five folds

- Get mean and standard deviation of the preceding five MSEs.

```
. * Compute the average MSE over the five folds and standard deviation
. svmat allmses, names(vallmses)

. qui sum vallmses

. display "CV5 = " %5.3f r(mean) " with st. dev. = " %5.3f r(sd)
CV5 = 9.699 with st. dev. = 3.389
```

K-fold CV using user-written crossfold command

- User-written crossfold command (Daniels 2012)
 - ▶ gives **square root** of MSE (RMSE) in each fold
- Here 5-fold CV for just one model - the full model
 - ▶ $CV_5 = (3.739^2 + 2.549^2 + 3.060^2 + 2.532^2 + 3.499^2)/5 = 3.699$.

```
. * Five-fold cross validation measure for one model with 11 regressors
. set seed 10101

. crossfold regress y x1 x2 x3, k(5)
```

| | RMSE |
|------|----------|
| est1 | 3.739027 |
| est2 | 2.549458 |
| est3 | 3.059801 |
| est4 | 2.532469 |
| est5 | 3.498511 |

```
. drop _est* // Drop variables created
```

K-fold Cross Validation example (continued)

- Now do 5-fold CV for all eight models with $K = 5$
 - ▶ model with only x_1 has lowest CV_5

```
. * Five-fold cross validation measure for all possible models
. forvalues k = 1/8 {
2.   set seed 10101
3.   qui crossfold regress y ${xlist`k'}, k(5)
4.   matrix RMSEs`k' = r(est)
5.   svmat RMSEs`k', names(rmse`k')
6.   qui generate mse`k' = rmse`k'^2
7.   qui sum mse`k'
8.   scalar cv`k' = r(mean)
9.   scalar sdcv`k' = r(sd)
10.  display "Model " "${xlist`k'}" _col(16) " CV5 = " %7.3f cv`k' ///
>    " with st. dev. = " %7.3f sdcv`k'
11. }
```

| | |
|----------------|------------------------------------|
| Model | CV5 = 11.960 with st. dev. = 3.561 |
| Model x1 | CV5 = 9.138 with st. dev. = 3.069 |
| Model x2 | CV5 = 10.407 with st. dev. = 4.139 |
| Model x3 | CV5 = 11.776 with st. dev. = 3.272 |
| Model x1 x2 | CV5 = 9.173 with st. dev. = 3.367 |
| Model x2 x3 | CV5 = 10.872 with st. dev. = 4.221 |
| Model x1 x3 | CV5 = 9.639 with st. dev. = 2.985 |
| Model x1 x2 x3 | CV5 = 9.699 with st. dev. = 3.389 |

One standard error “rule” for K-fold cross-validation

- K folds gives K estimates $MSE_{(1)}, \dots, MSE_{(K)}$
 - ▶ so we can obtain a standard error of CV_K

$$se(CV_K) = \sqrt{\frac{1}{K-1} \sum_{j=1}^K (MSE_{(j)} - CV_{(K)})^2}.$$

- A further guard against overfitting that is sometimes used (ISL2)
 - ▶ don't simply choose model with minimum $CV_{(K)}$
 - ▶ instead choose the smallest model for which CV is within one $se(CV)$ of minimum CV
 - ▶ clearly could instead use e.g. a 0.5 standard error rule.
- Example is determining degree p of a high order polynomial in x
 - ▶ if CV_K is minimized at $p = 7$ but is only slightly higher for $p = 3$ we would favor $p = 3$.

2.7 Leave-one-out Cross Validation (LOOCV)

- Use a **single observation for validation** and $(n - 1)$ for training
 - ▶ $\hat{y}_{(-i)}$ is \hat{y}_i prediction after OLS on observations $1, \dots, i - 1, i + 1, \dots, n$.
 - ▶ Cycle through all n observations doing this.

- Then LOOCV measure is

$$CV_n = \frac{1}{n} \sum_{i=1}^n MSE_{(-i)} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_{(-i)})^2$$

- Requires n regressions in general

- ▶ except for OLS can show $CV_n = \frac{1}{n} \sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{1 - h_{ii}} \right)^2$
 - ★ where \hat{y}_i is fitted value from OLS on the full training sample
 - ★ and h_{ii} is i^{th} diagonal entry in the hat matrix $\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}$.

- Used for bandwidth choice in **local** nonparametric regression
 - ▶ such as k-nearest neighbors, kernel and local linear regression
 - ▶ but not used for (global) machine learning (see below).

Leave-one-out cross validation example

- User-written command `loocv` (Barron 2014)
 - ▶ slow as written for any command, not just OLS.

```
. * Leave-one-out cross validation
. loocv regress y x1
```

Leave-One-Out Cross-Validation Results

| Method | Value |
|--------------------------|-----------|
| Root Mean Squared Errors | 3.0989007 |
| Mean Absolute Errors | 2.5242994 |
| Pseudo-R2 | .15585569 |

```
. display "LOOCV MSE = " r(rmse)^2
LOOCV MSE = 9.6031853
```

Cross validation and AIC

- LOOCV is the special case of K -fold CV with $K = N$
 - ▶ it has little bias as all but one observation is used to fit.
 - ▶ but large variance as n predicted $\hat{y}_{(-i)}$ based on very similar samples.
- For K -fold CV, $K = 5$ or $K = 10$ is found to be a good compromise
 - ▶ then neither high bias nor high variance.
- K -fold usually selects fewer variables than AIC.
- K -fold requires few model assumptions and has wide applicability.
 - ▶ Remember: For replicability set the seed as this determines the folds.

Cross validation and AIC and BIC

- Jun Shao (1997) *Statistica Sinica* 221-264
 - ▶ considers linear regression with i.i.d. $(0, \sigma^2)$ errors
 - ▶ n observations and p_n potential regressors
 - ▶ $\alpha \in A_n$ denotes a subset with $p_n(\alpha)$ of the p regressors
 - ▶ $\mu_n = \mathbf{x}_n(\alpha)\boldsymbol{\beta}(\alpha_n)$ is the conditional mean using $\alpha \in A_n$
 - ▶ GIC(λ_n) methods minimize $\text{MSE}(\alpha) + \lambda_n \widehat{\sigma}_n^2 p_n(\alpha) / n$
 - ★ where λ_n is the penalty
 - ★ and $\widehat{\sigma}_n^2$ is a consistent estimate of σ^2 .
- Case 1: $\lambda_n = 2$ asymptotically is AIC, LOOCV (delete-1 CV), CP
 - ▶ consistent model selection if there is no fixed dimension true model.
- Case 2: $\lambda_n \rightarrow \infty$ asymptotically is BIC and delete- d CV with $d/n \rightarrow 1$
 - ▶ consistent model selection when there is a fixed dimension true model.
- Case 3: $\lambda_n \geq 2$ and fixed asymptotically is delete- d CV with $d \in (0, 1)$ and K -fold CV with K fixed
 - ▶ compromise between 1 and 2.

2.8 Model selection: Forwards, backwards and best subsets

- Forwards selection (or specific to general)
 - ▶ start with simplest model (intercept-only) and in turn include the variable that is most statistically significant or most improves fit.
 - ▶ requires up to $p + (p - 1) + \dots + 1 = p(p + 1)/2$ regressions where p is number of regressors
- Backwards selection (or general to specific)
 - ▶ start with most general model and in drop the variable that is least statistically significant or least improves fit.
 - ▶ requires up to $p(p + 1)/2$ regressions
- Best subsets
 - ▶ for $k = 1, \dots, p$ find the best fitting model with k regressors
 - ▶ in theory requires $\binom{p}{0} + \binom{p}{1} + \dots + \binom{p}{p} = 2^p$ regressions
 - ▶ but leaps and bounds procedure makes this much quicker
 - ▶ $p < 40$ manageable though recent work suggests p in thousands.
- Hybrid
 - ▶ forward selection but after new model found drop variables that do not improve fit.

Best subsets selection and stepwise selection

- User-written `vselect` command (Lindsey and Sheather 2010)
 - ▶ this does not use CV but uses AIC, BIC, ...
 - ▶ best subsets gives best fitting model (lowest MSE) with one, two and three regressors
 - ▶ and for each of these best fitting models gives various penalty measures
 - ▶ all measures favor model with just x_1 .

```
. * Best subset selection with user-written add-on vselect
. vselect y x1 x2 x3, best
```

```
Response :      y
Selected predictors:  x1 x2 x3
```

Optimal models:

| # Preds | R2ADJ | C | AIC | AICC | BIC |
|---------|----------|----------|----------|----------|----------|
| 1 | .2043123 | .5925225 | 204.2265 | 204.8932 | 207.6042 |
| 2 | .1959877 | 2.002325 | 205.5761 | 206.7189 | 210.6427 |
| 3 | .1737073 | 4 | 207.5735 | 209.3382 | 214.329 |

predictors for each model:

```
1 : x1
2 : x1 x2
3 : x1 x2 x3
```

Example of penalty measures (continued)

- `vselect` also does forward selection and backward selection
 - ▶ then need to specify whether use R^2_{adj} , AIC, BIC or AICC
 - ▶ e.g. `vselect y x1 c2 c3, forward aic`
 - ▶ e.g. `vselect y x1 c2 c3, backward bic`
- And can specify that some regressors always be included
 - ▶ e.g. `vselect y x2 x3, fix(x1) best`
- User-written `gvselect` command (Lindsey and Sheather 2015) implements best subsets selection for any Stata command that reports $\ln L$
 - ▶ then best model of any size has highest $\ln L$
 - ▶ and best model size has lowest AIC or BIC.

2.9 Selection using Statistical Significance

- **In past not recommended**

- ▶ pre-testing changes the distribution of $\hat{\beta}$
- ▶ and with fixed size α (such as $\alpha = 0.05$) all potential regressors likely to be found significant as $n \rightarrow \infty$.

- But recent papers say okay if critical value increases with number of potential regressors at an appropriate rate.

- **Stepwise forward** based on $p < 0.05$

- ▶ Stata add-on command `stepwise, pe(.05)`

- ★ chooses model with only intercept and x_1

```
. * Stepwise forward using statistical significance at five percent
. stepwise, pe(.05): regress y x1 x2 x3
      begin with empty model
p = 0.0020 < 0.0500  adding  x1
```

| Source | SS | df | MS | Number of obs | = | 40 |
|----------|------------|----|------------|---------------|---|--------|
| Model | 101.318018 | 1 | 101.318018 | F(1, 38) | = | 11.01 |
| Residual | 349.556297 | 38 | 9.19884993 | Prob > F | = | 0.0020 |
| | | | | R-squared | = | 0.2247 |
| | | | | Adj R-squared | = | 0.2043 |
| Total | 450.874315 | 39 | 11.5608799 | Root MSE | = | 3.033 |

| y | Coef. | Std. Err. | t | P> t | [95% Conf. Interval] | |
|-------|----------|-----------|------|-------|----------------------|----------|
| x1 | 1.793535 | .5404224 | 3.32 | 0.002 | .6995073 | 2.887563 |
| _cons | 2.509313 | .5123592 | 4.90 | 0.000 | 1.472097 | 3.54653 |

- **Stepwise backward** based on $p < 0.05$

- ▶ Stata add-on command `stepwise, pr(.05)`

- ★ chooses model with only intercept and x_1

```
. * stepwise backward using statistical significance at five percent
. stepwise, pr(.05): regress y x1 x2 x3
                        begin with full model
p = 0.9618 >= 0.0500   removing x3
p = 0.4410 >= 0.0500   removing x2
```

| Source | SS | df | MS | Number of obs | = | 40 |
|----------|------------|----|------------|---------------|---|--------|
| Model | 101.318018 | 1 | 101.318018 | F(1, 38) | = | 11.01 |
| Residual | 349.556297 | 38 | 9.19884993 | Prob > F | = | 0.0020 |
| Total | 450.874315 | 39 | 11.5608799 | R-squared | = | 0.2247 |
| | | | | Adj R-squared | = | 0.2043 |
| | | | | Root MSE | = | 3.033 |

| y | Coef. | Std. Err. | t | P> t | [95% Conf. Interval] | |
|-------|----------|-----------|------|-------|----------------------|----------|
| x1 | 1.793535 | .5404224 | 3.32 | 0.002 | .6995073 | 2.887563 |
| _cons | 2.509313 | .5123592 | 4.90 | 0.000 | 1.472097 | 3.54653 |

- Option `hierarchical` allows selection in order of the specified regressors.

Testing-based Forward Model Selection

- Kozbur (2020), “Analysis of testing-based forward model selection”, *Econometrica*, vol. 88(5), 2147-2173.
- Assumptions similar to Belloni et al. (Ecta 2012) for LASSO
 - ▶ including sparsity (only a fraction $s_0 = o(n)$ of the p potential regressors in model belong).
- Normalize regressors so $\frac{1}{n} \sum_{i=1}^n x_{ij}^2 = 1$ for all regressors $j \leq p$.
- Then in case of independent heteroskedastic errors
 - ▶ use stepwise forward OLS regression of y_i on the x'_{ik} s
 - ▶ sequentially select regressors where at any given round
 - ★ choose regressor j if $W_j \geq W_k$ for all regressors j, k not already selected and (definition 1) $W_j \geq$ a complicated critical value.

Testing-based Forward Model Selection (continued)

- Simpler procedures are
 - ▶ Definition 2: $W_j = W_j^{het} = \hat{\theta} / se_{hetrob}(\hat{\theta}) \geq \Phi^{-1}(1 - \alpha/p)$
 - ▶ Definition 3: $W_j = \Delta_j MSE \geq \Phi^{-1}(1 - \alpha/p)$ where $\Delta_j MSE$ is the increase in MSE when x_j is dropped.
- $\Phi^{-1}(1 - \alpha/p)$ is like a Bonferroni correction for multiple testing
 - ▶ e.g. $\alpha = 0.05$ and $p = 100$ then $\Phi^{-1}(1 - 0.05/100) = \Phi^{-1}(0.9995) = 3.29$.

One Covariate at a Time Testing Model Selection

- Chudik, Kapetanios and Pesaran (2018), “A One Covariate at a time, Multiple Testing Approach to Variable Selection in High-Dimensional Linear Regression Models”, *Econometrica*, vol. 86(4), 1479-1512.
- Proposes individual testing for each potential regressor
 - ▶ first stage: accept all those regressors j with $|W_j| > \text{threshold}$
 - ▶ further stages: see if some variables missed first time around by adding additional regressors if $|W_j| > \text{higher threshold}$
 - ▶ repeat previous step if necessary (few such additional steps are needed).
- Assumptions include
 - ▶ errors are a martingale difference sequence
 - ▶ W_j is Wald test statistic based on homoskedastic errors.

One Covariate at a Time Testing Model Selection (continued)

- The key threshold for p potential regressors and size α tests
 - ▶ $|W_j| > \Phi^{-1}(1 - \alpha/(2p^\delta))$
 - ▶ $\delta = 1$ corresponds to usual Bonferroni for multiple testing
- Simulations: $\alpha = 0.01$, $\delta = 1$ at first stage, $\delta = 2$ subsequently
 - ▶ $p = 100$: $\Phi^{-1}(1 - 0.01/(2 \times 100^1)) = \Phi^{-1}(0.9998) = 3.54$
 - ▶ $p = 100$: $\Phi^{-1}(1 - 0.01/(2 \times 100^2)) = \Phi^{-1}(0.999998) = 4.61$.

References

- ISLR2: Gareth James, Daniela Witten, Trevor Hastie and Robert Tibsharani (2021), *An Introduction to Statistical Learning: with Applications in R*, 2nd Ed., Springer.
 - ▶ PDF and \$40 softcover book at <https://link.springer.com/book/10.1007/978-1-0716-1418-1>
- ISLP: Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani and Jonathan Taylor (2023), *An Introduction to Statistical Learning: with Applications in Python*, Springer.
 - ▶ Masters level book. Free PDF from <https://www.statlearning.com/> and \$40 softcover book via Springer mycopy.
- Chapter 28 “Machine Learning for Prediction and Inference” in A. Colin Cameron and Pravin K. Trivedi (2022), *Microeconometrics using Stata*, Second edition.