Regularized Regression

Ridge and The Lasso

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

A Note

- **Not just for continuous targets** – Ridge and Lasso have extensions for logistic (and other) regression models.

- In the corresponding code from R glmnet package, simply **add the option family="binomial" for a classification task.**

Regularization and Overfitting

• • •

The bias-variance tradeoff revisited

Overfitting

- Models with too many variables will overfit the training data.

- IF you want a linear model, but:
  - Leaving variables out is not an option.
  - You find it too difficult to determine which variables to leave out.
  - You need many variables to model a significant portion of signal.
  - You have more variables than observations.
  - You want superior predictions on out-of-sample data.

  ==> THEN Regularized Regression is your best bet.

Bias-Variance Tradeoff

The mean-squared error of a model on out-of-sample test data can be decomposed into three terms:

$$E(y - \hat{y})^2 = Var(\hat{y}) + [Bias(\hat{y})]^2 + Var(\epsilon)$$

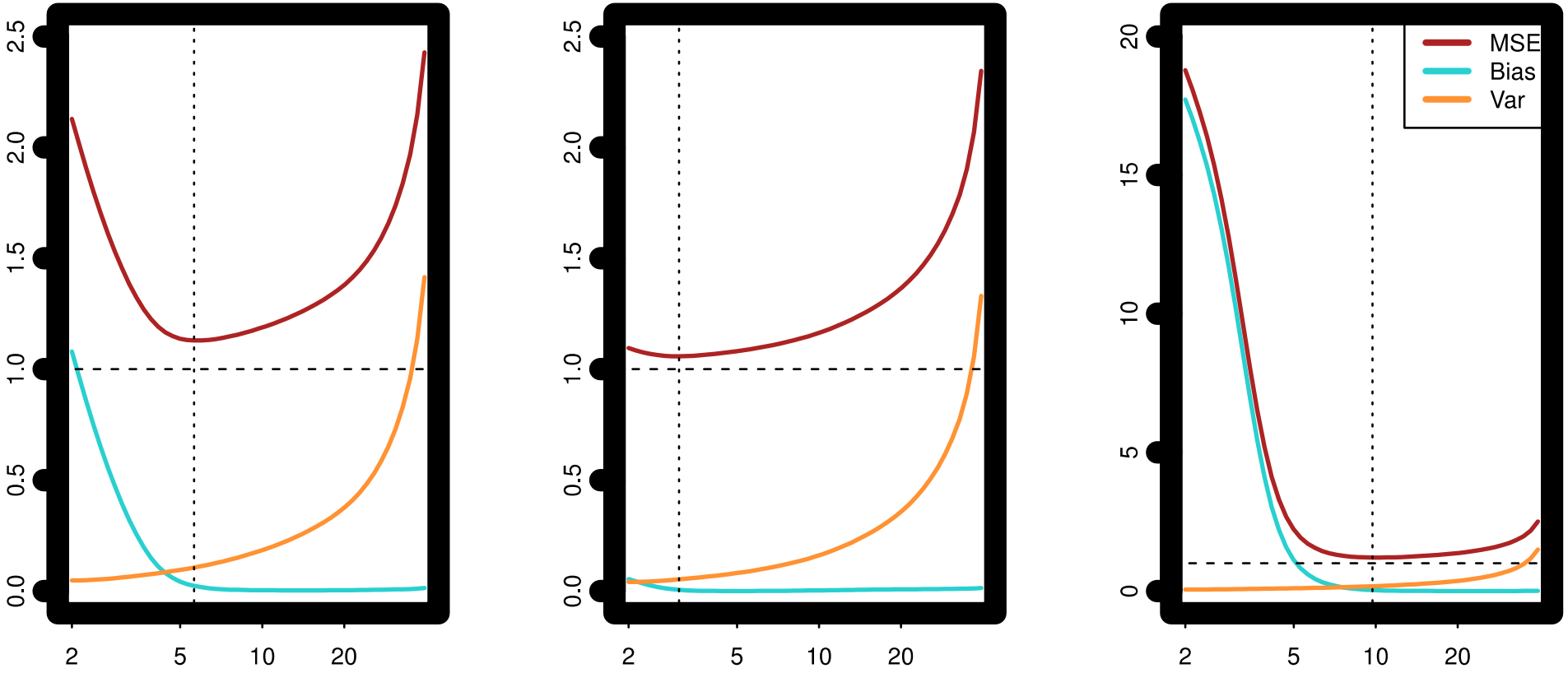
The **variance** of the estimates when the model is created on different training sets

The squared **bias** of the estimates (The squared difference between the average estimate over different training sets and the actual target value)

The irreducible error (can't model this)

Bias-Variance Tradeoff Illustrated

(For 3 different data sets)



(x-axis represents model complexity)

Regularization

- In Machine Learning, **regularization is a common tool to control the complexity/flexibility of a model**.
- Regularization adds a penalty term to the objective function of a model that penalizes model complexity.
- Also called **parameter shrinkage**
- Regularization has been shown to trade the introduction of small amounts of bias for a reduction in large amounts of variance.
- Regularization thus creates a **biased model**.

Strong Feelings...

"If you're using regression without regularization, you have to be very special..."

– Owen Zhang (Kaggle rank 3, Previously Chief Product Officer at Data Robot, now Hedge Fund Quant)

What if I told you...

- You could just go ahead and keep ALL of your variables in the model
- Without overfitting the training data
- Resulting in a complex yet generalizable model

Ridge Regression

• • •

a.k.a.

Tikhonov Regularization

L_2 Regularization

Weight-decay

(Tikhonov **1943**)

Ridge Regression

- Ridge regression is a biased regression technique (like PCR)
- Parameter estimates tend to have lower variance than OLS estimates, but are biased
- Often proposed as a 'solution' for multicollinearity when estimating parameters.

- Theoretically shown to trade large amounts of variance for minimal amounts of bias

Ridge Regression

- OLS minimizes the sum of squared error:

OLS Objective function: $f_{OLS}(x) = \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2$

- Ridge regression adds a penalty for the parameters in the model:

Ridge Objective function: $f_{ridge}(x) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$

$= SSE + \lambda \|\boldsymbol{\beta}\|_2^2$

- FIRST STEP IS TO STANDARDIZE YOUR DATA!
- Most software will do this FOR YOU (SAS - glmselect, R - glmnet) but worth checking!

λ - The Regularization Parameter

- **The larger the value of λ, the more bias** is introduced into model.
- At very large values, all parameters would be forced to zero.
- At very small values, the penalty term would have no effect.

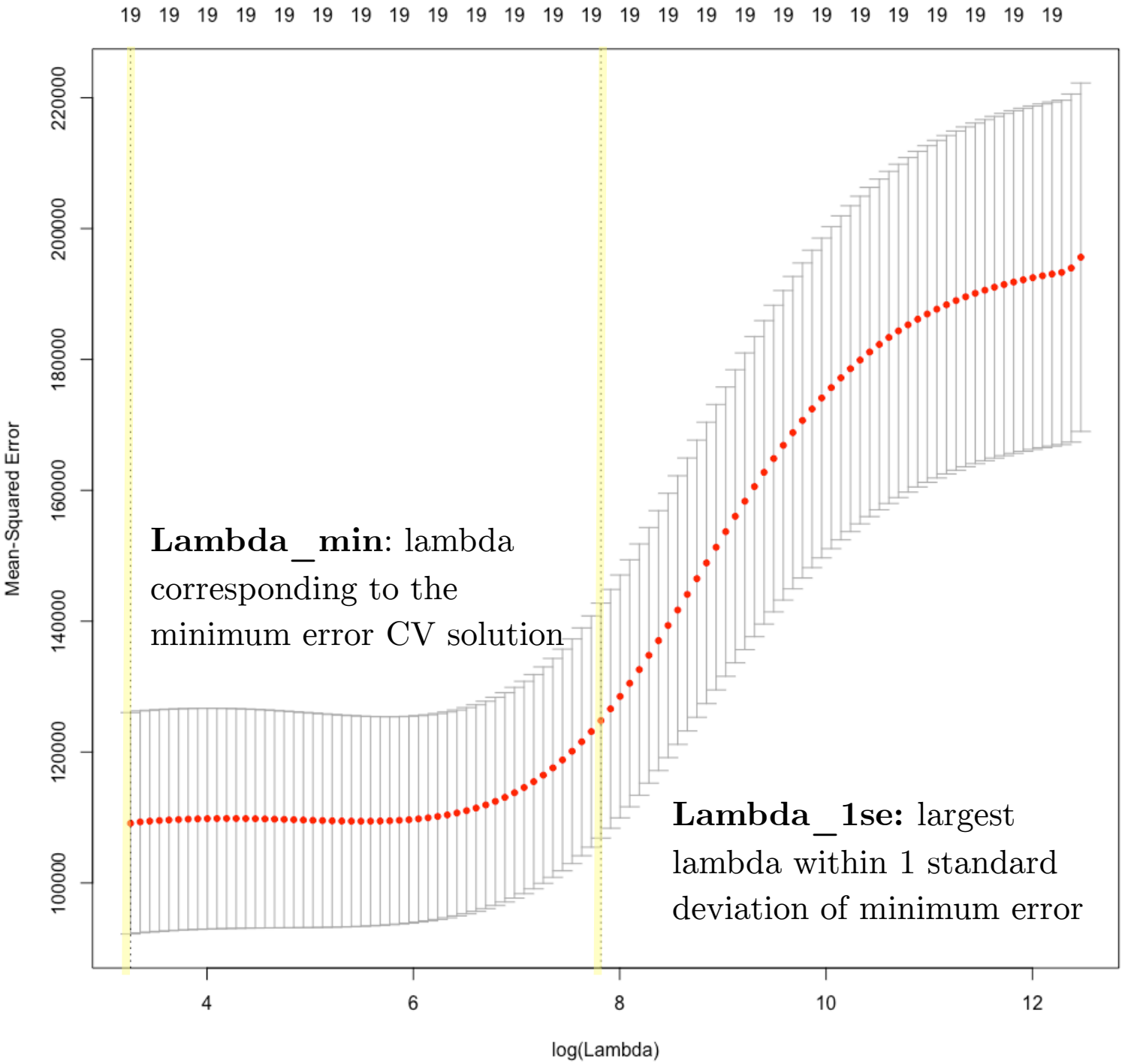
- Many ways to set/tune this parameter have been proposed through the years, but validation/CV is preferred.

λ - The Regularization Parameter

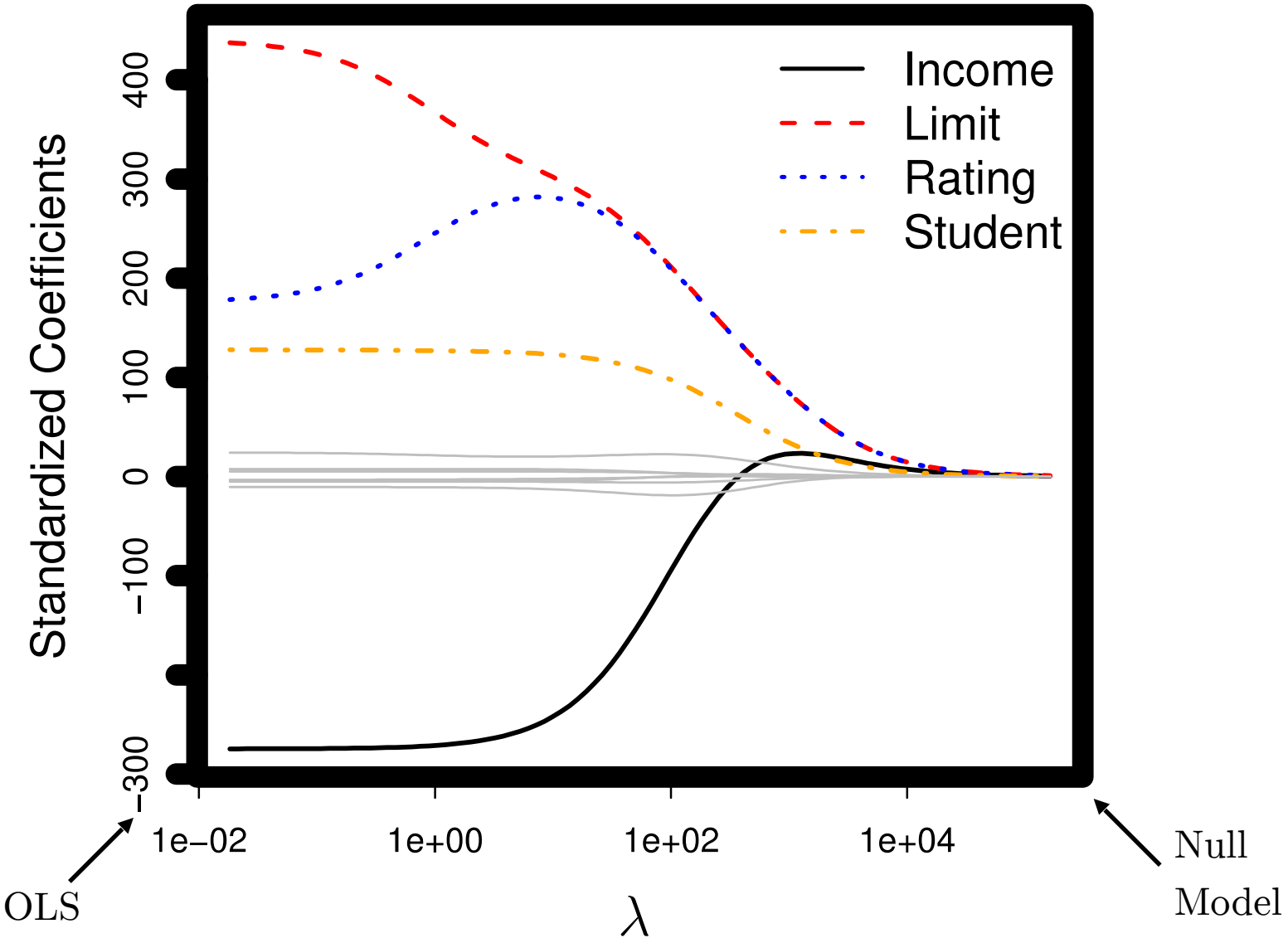
Optimization using cross validation:

Option 1: Chose λ that provides the minimum average error on K-fold cross validation. ($= lambda\_min$)

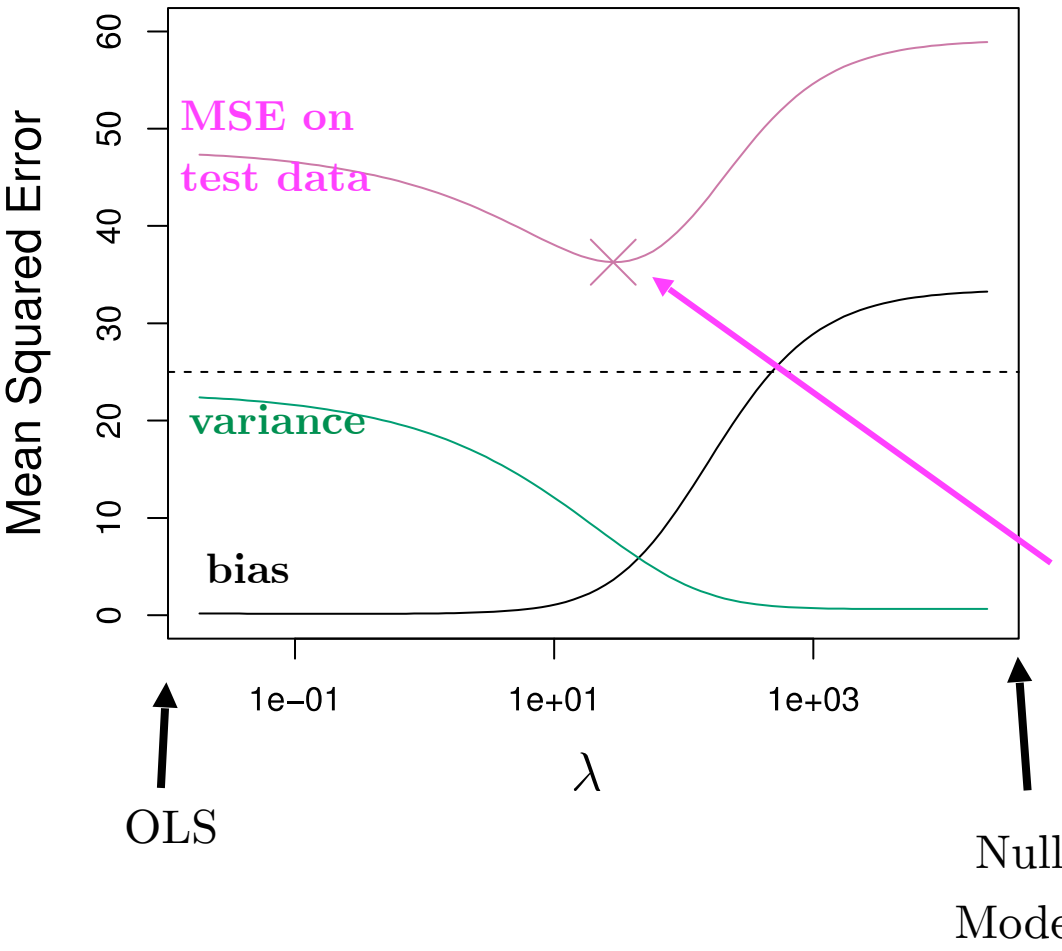
Option 2: (recommended) Chose the largest value of λ that provides an average error within 1 standard deviation of the minimum average error. ($= lambda\_1se$)


19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
Mean-Squared Error
100000
120000
140000
160000
180000
200000
220000
4
6
8
10
12
log(Lambda)
Lambda_min: lambda corresponding to the minimum error CV solution
Lambda_1se: largest lambda within 1 standard deviation of minimum error

How λ affects parameters



How λ affects bias/variance/MSE



When λ small, no penalty

- high variance
- no bias
- high MSE on test data

When λ big, null model

- high bias
- no variance
- high MSE on test data

Sweet spot

- minimizes MSE on test data
- introduces small bias
- substantially reduces variance

Ridge Regression Summary

<u>Advantages</u>

- **Super fast. Simple closed form solution, similar to OLS.**
- Sidestep overfitting concerns without leaving variables out of the model (**no variable selection required**!)
- Works well in situations where least squares estimates have high variance (**solution to severe multicollinearity**)

<u>Disadvantages</u>

- Will not create many zero parameter estimates, so **all of the input variables likely to stay in the model**.
- **No statistical hypothesis tests** for beta coefficients

The LASSO

• • •

a.k.a

L_1 Regularization

(Tibshirani **1996**)

Penalties for Model Selection

- In recent years, stepwise selection techniques for variable selection have come under fire.
- Alternative methods, such as "The LASSO" have been proposed and have soared in popularity.

Drawbacks to Stepwise Selection

- Bias in parameter estimation
  - Standard errors biased toward zero
  - p-values biased toward zero
  - Parameter estimates biased away from zero
  - R-Squared biased upwards
- F and Chi-Square tests don't have the desired distribution
- Resulting models are complex with exacerbated collinearity problems
- Inconsistencies among model selection algorithms
- An inherent problem with multiple hypothesis testing
- An inappropriate focus or reliance on a single best model

(MJ Whittingham et al – 2006, Harrell – 2010, Flom & Cassell – 2007)

Analogy for Stepwise

Flom and Cassell (2007) write:

> *"In Stepwise Regression, this assumption [of independent hypothesis tests] is grossly violated in ways that are difficult to determine.*
>
> *For example, if you toss a coin ten times and get ten heads, then you are pretty sure that something weird is going on. You can quantify exactly how unlikely such an event is...*
>
> *If you have 10 people each toss a coin ten times, and one of them gets 10 heads, you are less suspicious, but you can still quantify the likelihood.*
>
> *But if you have a bunch of friends (you don't count them) toss coins some number of times (they don't tell you how many) and someone gets 10 heads in a row, you don't even know how suspicious to be. That's stepwise."*

LASSO Regression

- OLS minimizes the sum of squared error:

OLS Objective function: $f_{OLS}(x) = \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2$

- LASSO regression adds a penalty for the parameters in the model:

LASSO Objective function: $f_{LASSO}(x) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$

$= SSE + \lambda \|\boldsymbol{\beta}\|_1$

- FIRST STEP IS TO STANDARDIZE YOUR DATA!
- Most software will do this FOR YOU (SAS - glmselect, R - glmnet) but worth checking!

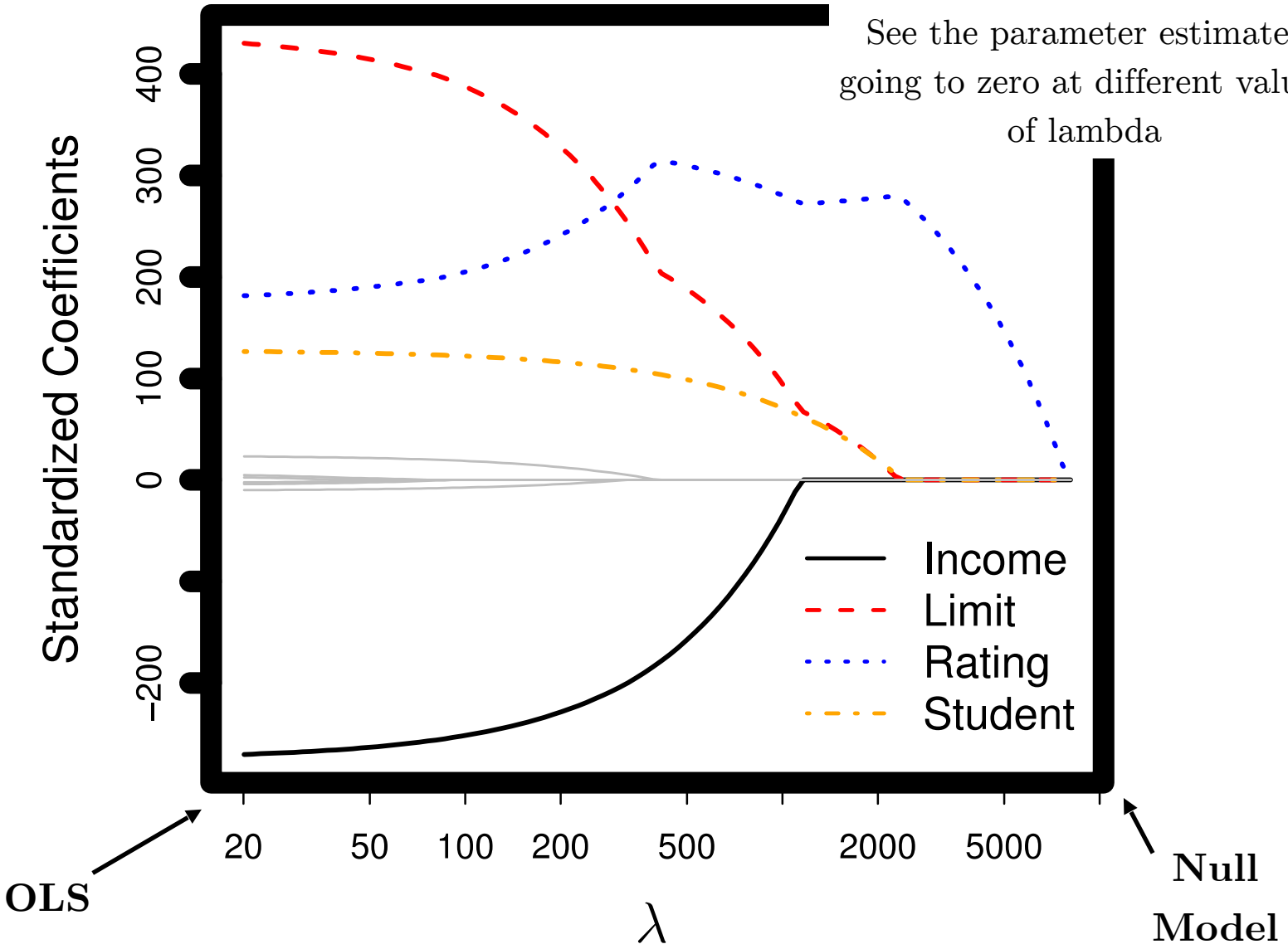
LASSO = Variable Selection

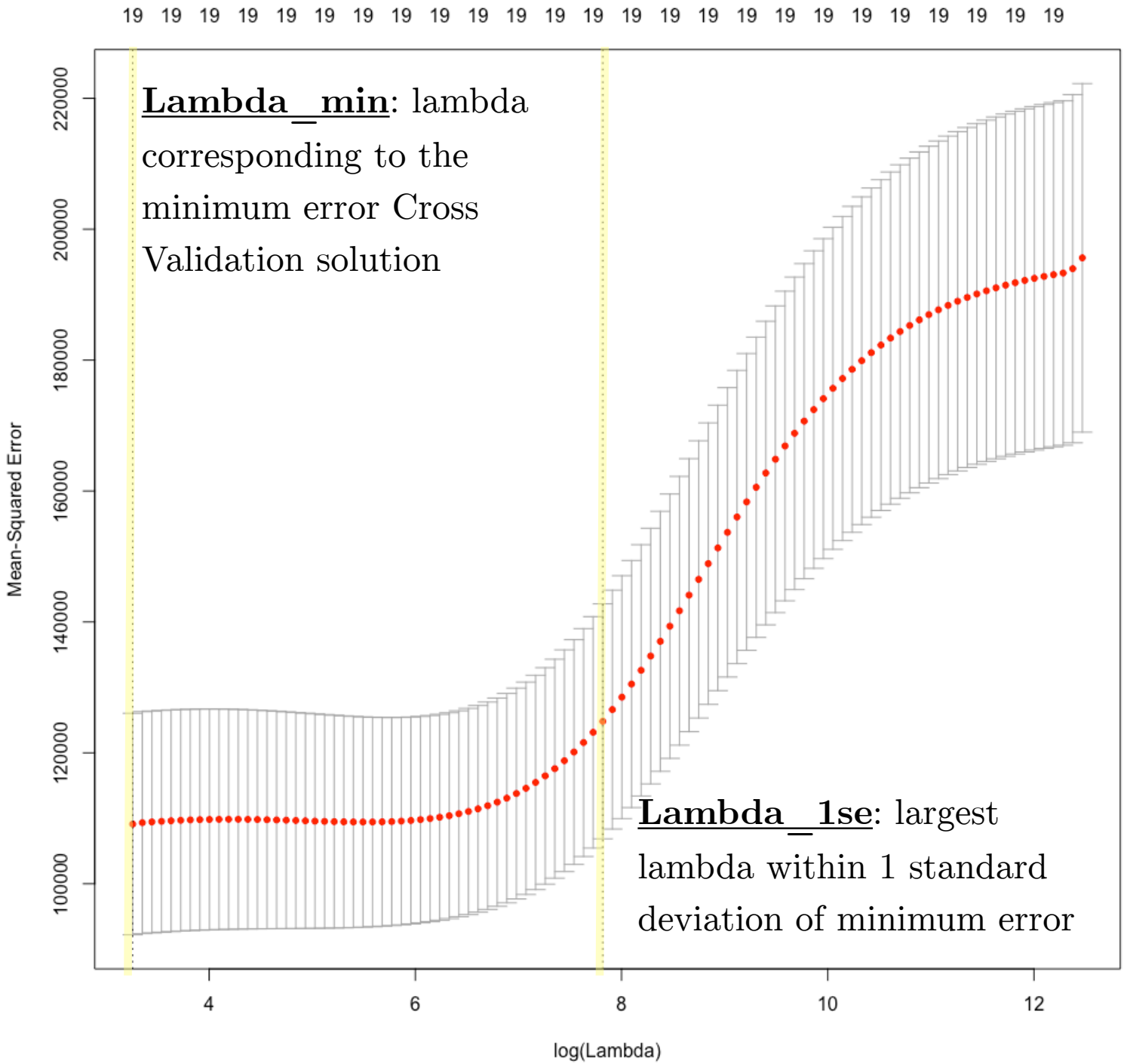
The LASSO penalty has the added benefit that it causes **many of the parameter estimates to tend toward zero**.

=> The LASSO produces **sparse solutions**

This implies **automated variable selection**

How λ affects Parameters




19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
Lambda_min: lambda corresponding to the minimum error Cross Validation solution
Lambda_1se: largest lambda within 1 standard deviation of minimum error
Mean-Squared Error
100000
120000
140000
160000
180000
200000
220000
4
6
8
10
12
log(Lambda)

LASSO Regression

- Very common when the number of variables is overwhelming for stepwise selection (particularly in text)

- Generally implemented through Least Angle Regression (LARS) algorithm (Efron et al 2004)
  - glmnet package in R
  - lars package in R
  - LARS node in SAS EM
  - proc glmselect option selection=LASSO

Predicting Salary of Baseball Players

• • •

An Example in R

Stepwise Selection the ML way

• • •

The number of variables in the model, p, is a hyperparameter to be tuned

Stepwise Selection Using Validation

The purpose here is to choose a level of model complexity, p= the number of parameters.

1. Run the stepwise selection algorithm on training data. For all possible number of variables, p, find the chosen model.
2. Compare the p models found in step 1 on validation data and record the MSE.
3. Pick the "optimal" number of parameters p* as the one that minimized the MSE on validation data.
4. Now you've validated your modelling process. Re-run the stepwise selection on the entire data to choose p* parameters.

   Yes, They may be different when you use all the data! That's ok. You've validated the procedure!

Forward Selection Using Validation: Tuning p

Build models using forward selection on **Training Data**

Forward Selection choosing 1 variable

Forward Selection choosing 2 variables

Forward Selection choosing 3 variables

Forward Selection choosing 4 variables

Forward Selection choosing 5 variables

Forward Selection choosing 6 variables

Forward Selection choosing 7 variables

Get predictions for **Validation Data** and compute error

Validation Error:

25

20

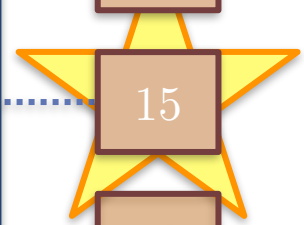
18

15

16

19

21



Forward Selection Using Validation: Finalize Model

Training Data

Validation Data

Forward Selection choosing 4 variables

Final Model $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4$

Stepwise Selection on Hitters Data

```
> regfit.fwd$vorder
```

```
[1]  1 13  3 17 16  2  7 14 12  9 18 15  5 19  4 10  6 20  8 11
```

```
> regfit.bwd$vorder
```

```
[1]  1 12  3 17  2  7 16 14 13  9 18 15  5 19  4 10  6 20  8 11
```

When backward selection arrived at a 4 variable model, it contained columns {1, 12, 3, 17}

Stepwise Selection on Hitters Data

```
> regfit.fwd$vorder
```

```
 [1]  1 13  3 17 16  2  7 14 12  9 18 15  5 19  4 10  6 20  8 11
```

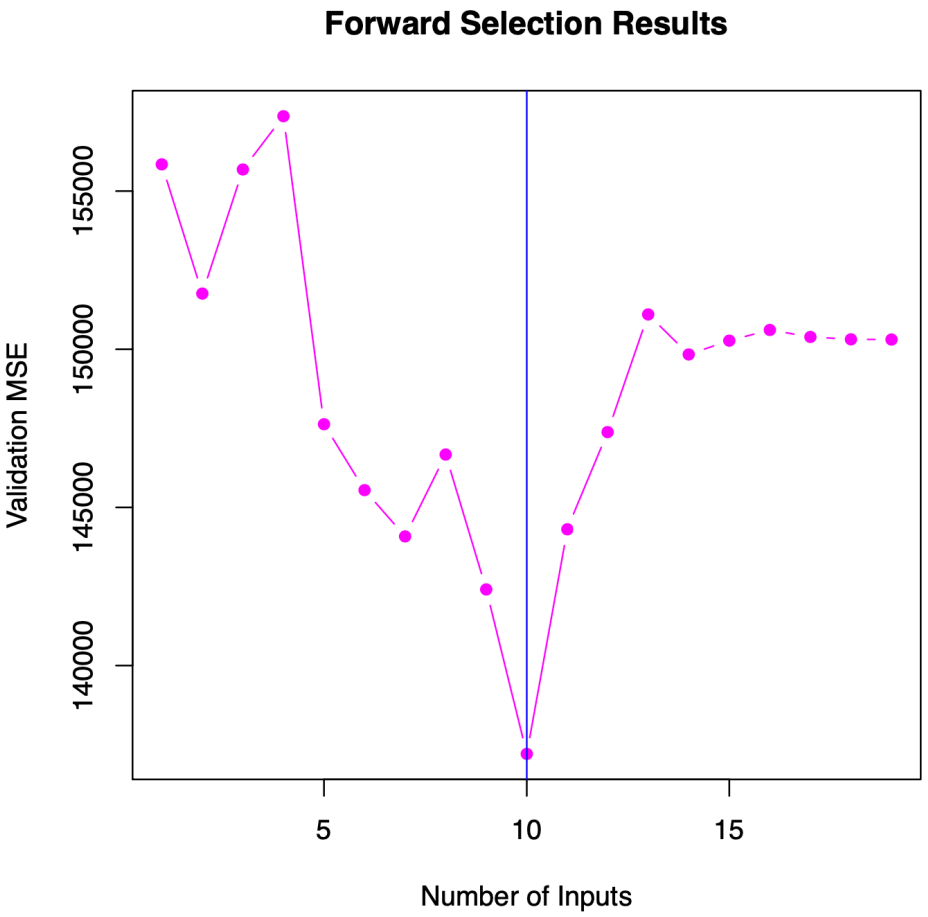
```
> regfit.bwd$vorder
```

```
 [1]  1 12  3 17  2  7 16 14 13  9 18 15  5 19  4 10  6 20  8 11
```

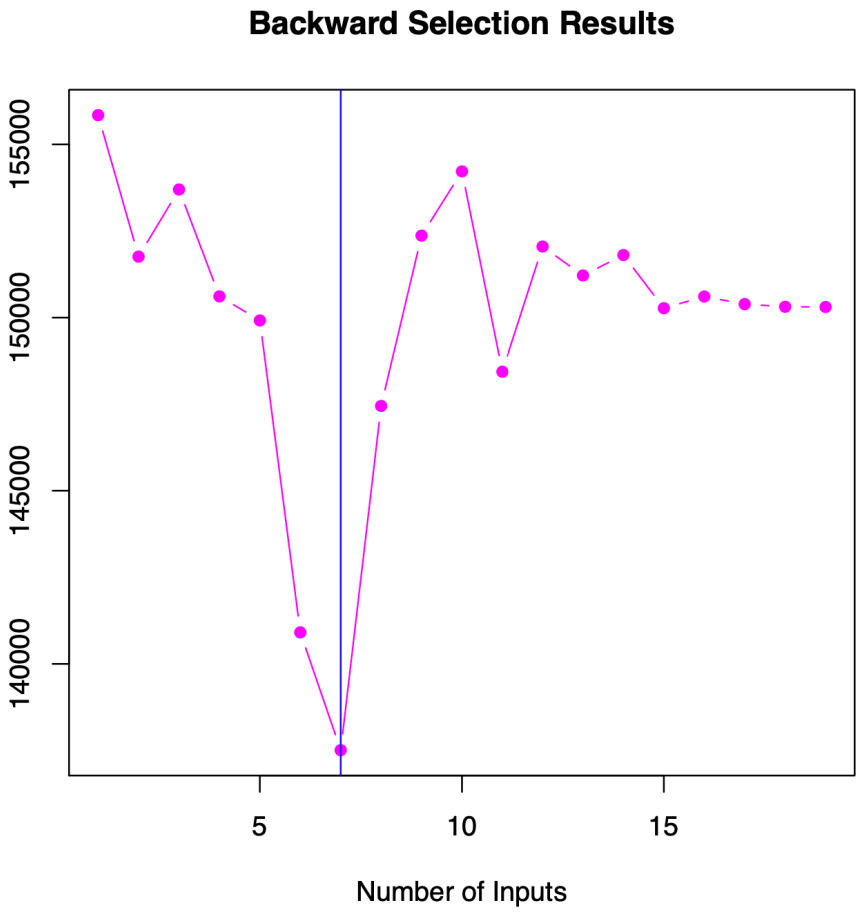
This agreed with forward selection, which added that variable to the model *last*

The *first* variable to leave in backward selection was column 11.

Stepwise Selection on Hitters Data

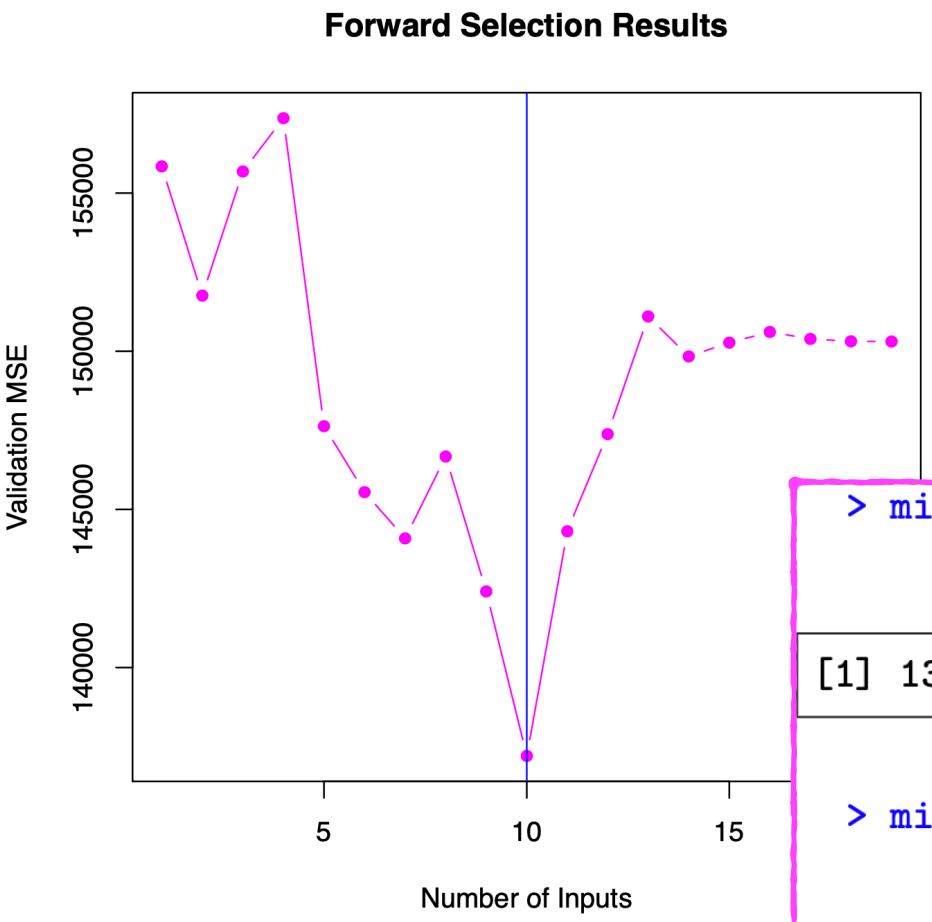


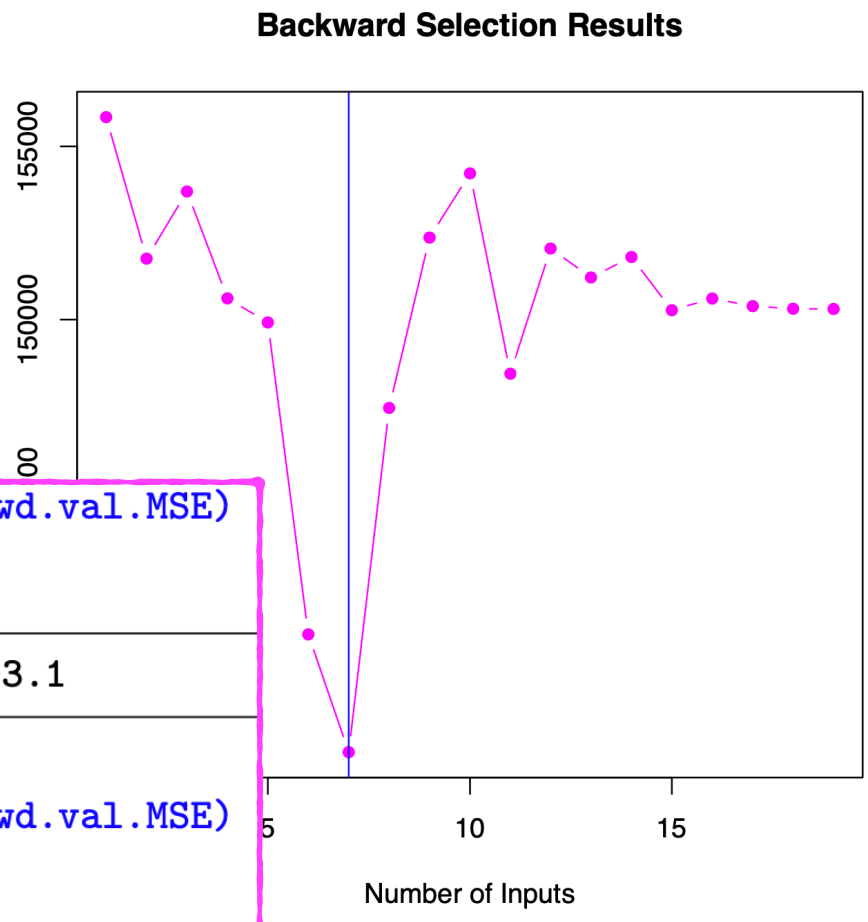
$p^* = 10$



$p^* = 7$

Stepwise Selection on Hitters Data






```
> min(fwd.val.MSE)

[1] 137203.1

> min(bwd.val.MSE)

[1] 137510.2
```

$p^* = 10$

$p^* = 7$

Stepwise Selection on Hitters Data: Conclusion

- In the Hitters example, we found that forward selection chose a model with 10 parameters and backward selection chose a model with 7 parameters.
- The validation MSE on both models was ~ the same.
- Choose the simpler model.
- To obtain final model, run backward selection choosing 7 variables on the *whole* data set (training + validation).

Ridge Regression

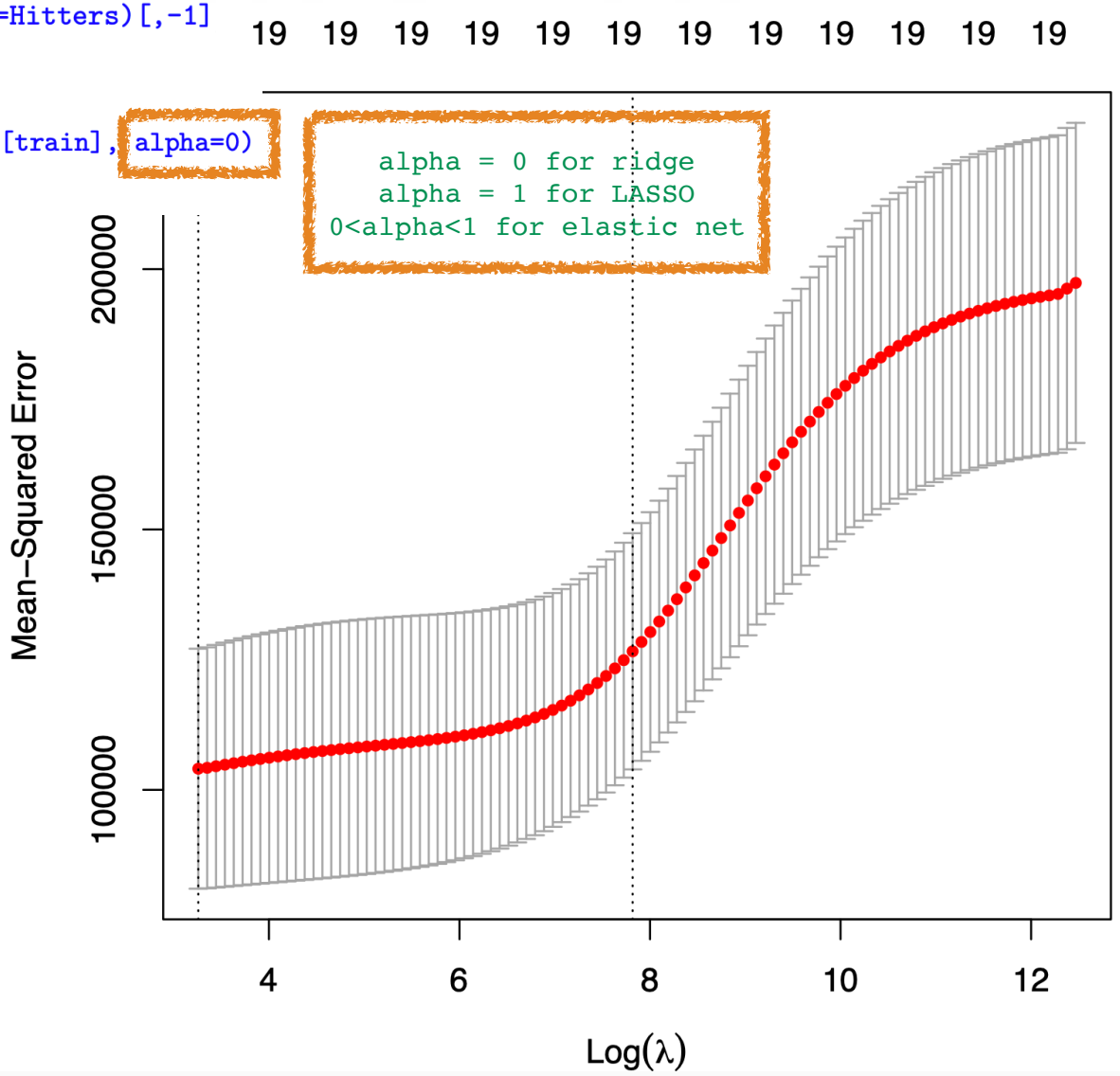
- The number of parameters used in the stepwise selection models didn't agree, nor did the actual variables used.
- If simplicity of the model is not as important as generalizability, we can consider ridge regression.
- Why?
  - Skip the agony of choosing predictors. Use them all, and shrink parameters to control for overfitting.
  - Ridge regression generally yields better predictions than OLS through a better bias-variance compromise.
  - Works especially well in the presence of severe multicollinearity in predictor variables.

Ridge Regression on Hitters Data

```
> X=model.matrix(Salary~. ,data=Hitters)[,-1]
> y = Hitters$Salary
```

```
> set.seed(1)
> cv.out = cv.glmnet(X[train,], y[train], alpha=0)
> plot(cv.out)
```

alpha = 0 for ridge
alpha = 1 for LASSO
0<alpha<1 for elastic net



Ridge Regression on Hitters Data

```
> bestlambda=cv.out$lambda.min
> bestlambda
```

```
[1] 26.01949
```

```
> ridge.mod = glmnet(X[train,], y[train], alpha=0, lambda=bestlambda)
> pred.ridge = predict(ridge.mod, newx=X[test,])
> val.MSE.ridge = mean((pred.ridge - y[test])^2)
> val.MSE.ridge
```

```
[1] 131187.9
```

```
> min(fwd.val.MSE)
```

```
[1] 137203.1
```

```
> min(bwd.val.MSE)
```

```
[1] 137510.2
```

Lambda.min vs Lambda.1se

Lambda.min

The value of lambda that provides the minimum average MSE on cross validation

Lambda.1se

The value of lambda that provides the simplest model but still provides MSE within 1 standard error of the minimum of cross validation

Sometimes lambda.min provides a model that still has too much variance and lambda.1se provides a slightly more stable model with less variance

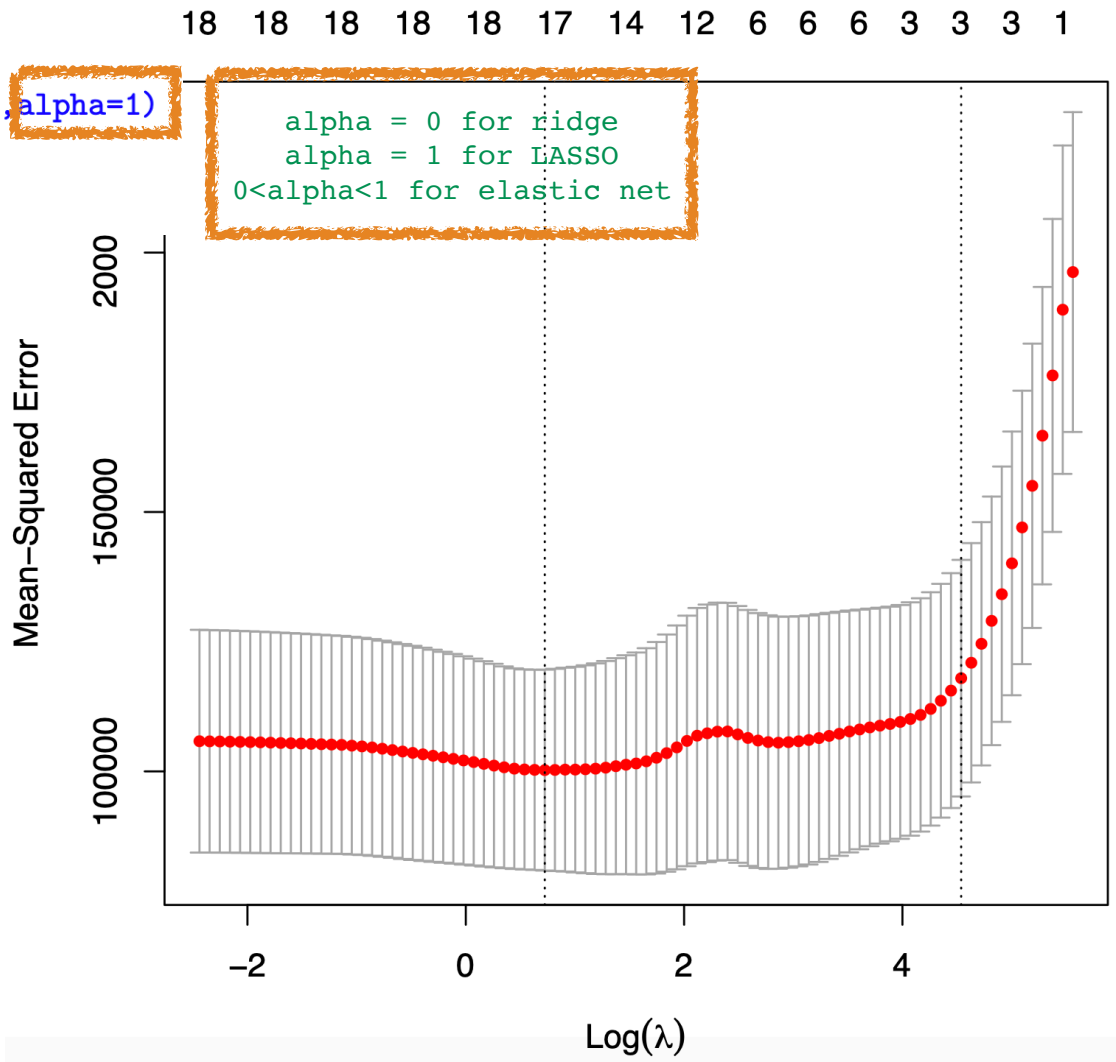
The LASSO

- If simplicity of the model IS desired, but we decide that we're not comfortable with the stepwise selection procedure...
    - (In the case of many variables (~50-100) we probably *shouldn't* be comfortable with such a procedure)

- Parameter shrinkage methods like the LASSO have proven to work better in light of the bias-variance trade-off.

The LASSO on Hitters Data

```
> set.seed(1)
> cv.out=cv.glmnet(X[train,],y[train],alpha=1)
> plot(cv.out)
```



The LASSO on Hitters Data

```
> bestlambda=cv.out$lambda.min
> pred.lasso = predict(cv.out, s=bestlambda, newx=X[test,])
> val.MSE.lasso = mean((pred.lasso-y[test])^2)
> val.MSE.lasso

[1] 137038
```

```
> val.MSE.ridge

[1] 131187.9
```

```
> min(fwd.val.MSE)

[1] 137203.1

> min(bwd.val.MSE)

[1] 137510.2
```

The LASSO on Hitters Data

```
> out=glmnet(X,y,alpha=1,lambda=bestlambda)
> lasso.coef=predict(out, type="coefficients")
> lasso.coef
```

(Intercept)	1.379844e+02
AtBat	-1.729667e+00
Hits	6.053342e+00
HmRun	1.586088e-01
Runs	.
RBI	.
Walks	5.054177e+00
Years	-1.033180e+01
CAtBat	-7.831918e-04
CHits	.
CHmRun	5.607890e-01
CRuns	7.199190e-01
CRBI	3.905694e-01
CWalks	-6.019104e-01
LeagueN	3.330146e+01
DivisionW	-1.193222e+02
PutOuts	2.769324e-01
Assists	2.084560e-01
Errors	-2.336031e+00
NewLeagueN	.

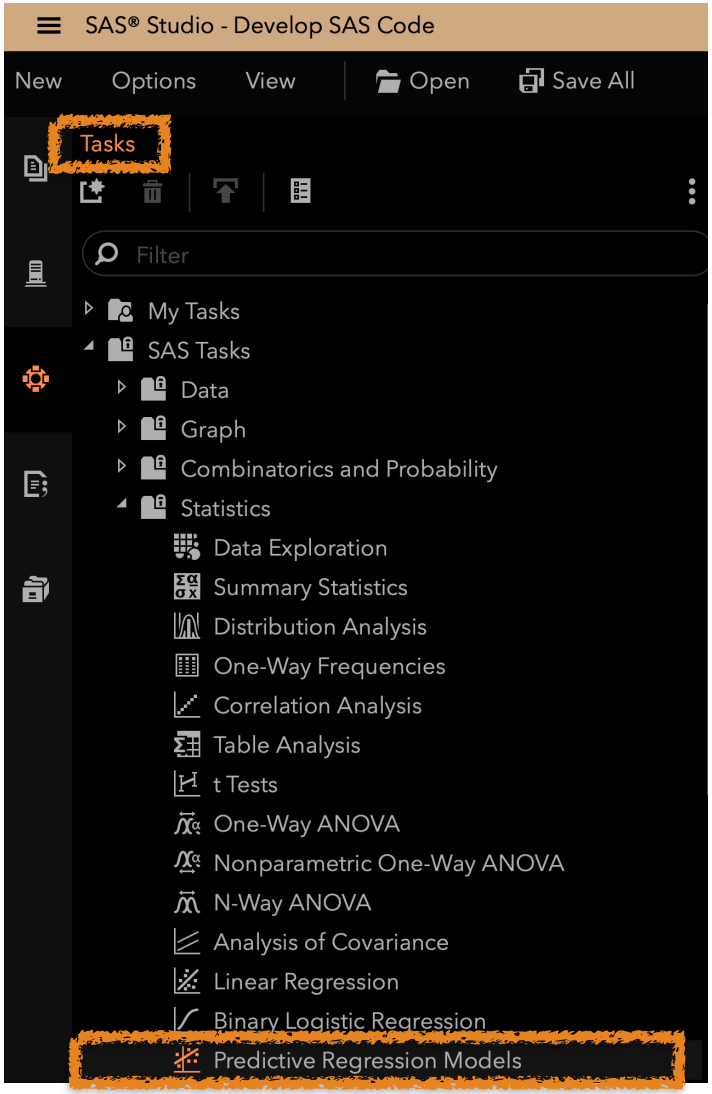
LASSO automatically creates *sparse* solutions.

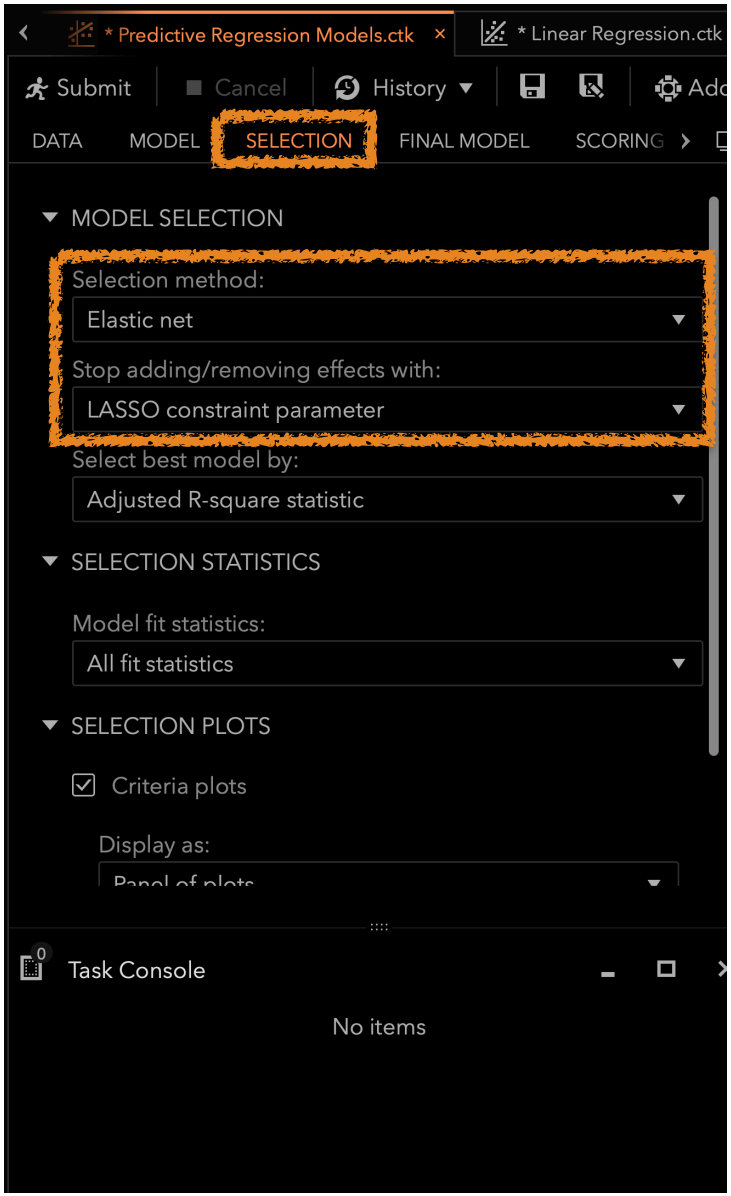
=> Variable Selection

SAS Viya

• • •

LASSO and ELASTIC NET in Viya Studio


SAS® Studio - Develop SAS Code
New
Options
View
Open
Save All
Tasks
Filter
My Tasks
SAS Tasks
Data
Graph
Combinatorics and Probability
Statistics
Data Exploration
Summary Statistics
Distribution Analysis
One-Way Frequencies
Correlation Analysis
Table Analysis
t Tests
One-Way ANOVA
Nonparametric One-Way ANOVA
N-Way ANOVA
Analysis of Covariance
Linear Regression
Binary Logistic Regression
Predictive Regression Models


* Predictive Regression Models.ctk
* Linear Regression.ctk
Submit
Cancel
History
DATA
MODEL
SELECTION
FINAL MODEL
SCORING
MODEL SELECTION
Selection method:
Elastic net
Stop adding/removing effects with:
LASSO constraint parameter
Select best model by:
Adjusted R-square statistic
SELECTION STATISTICS
Model fit statistics:
All fit statistics
SELECTION PLOTS
Criteria plots
Display as:
Task Console
No items

The Elastic Net

• • •

Combining the L1 and L2 penalties

ElasticNet Criteria

(Zou & Hastie 2005)

The ElasticNet Criteria **combines the L_1 penalty and the L_2 penalty** to achieve both the parameter shrinkage of ridge regression and the sparsity feature of the LASSO.

$$f_{ELASTIC}(x) = \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2$$

Works better than LASSO when some of your input variables are highly correlated

In R glmnet package, simply set 0<alpha<1.

Alpha closer to 0 emphasizes ridge regression

Alpha closer to 1 emphasizes LASSO.