Modern Statistics and Machine Learning: Regularized Regression

37105 Data Science for Marketing Decision Making
Günter J. Hitsch
Chicago Booth

Winter 2018

Overview

1. The bias-variance tradeoff
2. Ridge regression — regularization and shrinkage
3. LASSO — variable selection
4. Cross-validation
5. Elastic nets
6. Classification

The bias-variance trade-off

The data set: $\mathcal{D} = ((y_1, \boldsymbol{x}_1), \ldots, (y_n, \boldsymbol{x}_n))$

To simplify the notation, $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})$ contains the values for all inputs

$r(\boldsymbol{x})$ is the regression function,

$$r(\boldsymbol{x}) = \mathbb{E}[Y|\boldsymbol{x}]$$

We often propose some model $f(\boldsymbol{x}, \boldsymbol{\beta})$ for the regression function and use the data $\mathcal{D}$ to estimate the model parameters, $\boldsymbol{\beta}$.

$\hat{r}(\boldsymbol{x})$ is the estimated regression function, given the *estimated* model parameters $\hat{\boldsymbol{\beta}}$:

$$\hat{r}(\boldsymbol{x}) = f(\boldsymbol{x}, \hat{\boldsymbol{\beta}})$$

In the linear regression model:

$$r(\boldsymbol{x}) = \beta_0 + \sum_{k=1}^{p} x_k \beta_k$$

The **variance** of the estimated regression function at a specific value of the inputs, $\boldsymbol{x}$, is

$$\begin{aligned}
\text{var}(\hat{r}(\boldsymbol{x})) &= \mathbb{E}\left[(\hat{r}(\boldsymbol{x}) - \mathbb{E}[\hat{r}(\boldsymbol{x})])^2\right] \\
&= \mathbb{E}\left[\hat{r}(\boldsymbol{x})^2 - 2\hat{r}(\boldsymbol{x})\mathbb{E}[\hat{r}(\boldsymbol{x})] + \mathbb{E}[\hat{r}(\boldsymbol{x})]^2\right] \\
&= \mathbb{E}[\hat{r}(\boldsymbol{x})^2] - \mathbb{E}[\hat{r}(\boldsymbol{x})]^2
\end{aligned}$$

The **bias** of the prediction is

$$\begin{aligned}
\text{bias}(\hat{r}(\boldsymbol{x})) &= \mathbb{E}[\hat{r}(\boldsymbol{x}) - r(\boldsymbol{x})] \\
&= \mathbb{E}[\hat{r}(\boldsymbol{x})] - r(\boldsymbol{x})
\end{aligned}$$

The estimated regression function allows us to predict the output $\hat{Y} = \hat{r}(\boldsymbol{x})$ for the input values $\boldsymbol{x}$. The accuracy of this prediction can be measured using the mean-squared error (MSE)

$$\begin{aligned}
\mathbb{E}\left[(r(\boldsymbol{x}) - \hat{r}(\boldsymbol{x}))^2\right] &= \mathbb{E}\left[r(\boldsymbol{x})^2 - 2r(\boldsymbol{x})\hat{r}(\boldsymbol{x}) + \hat{r}(\boldsymbol{x})^2\right] \\
&= \mathbb{E}\left[r(\boldsymbol{x})^2 - 2r(\boldsymbol{x})\hat{r}(\boldsymbol{x}) + \hat{r}(\boldsymbol{x})^2 + \mathbb{E}[\hat{r}(\boldsymbol{x})]^2 - \mathbb{E}[\hat{r}(\boldsymbol{x})]^2\right] \\
&= \mathbb{E}[\hat{r}(\boldsymbol{x})^2] - \mathbb{E}[\hat{r}(\boldsymbol{x})]^2 + \left(r(\boldsymbol{x})^2 - 2r(\boldsymbol{x})\mathbb{E}[\hat{r}(\boldsymbol{x})] + \mathbb{E}[\hat{r}(\boldsymbol{x})]^2\right) \\
&= \left(\mathbb{E}[\hat{r}(\boldsymbol{x})^2] - \mathbb{E}[\hat{r}(\boldsymbol{x})]^2\right) + \left(\mathbb{E}[\hat{r}(\boldsymbol{x})] - r(\boldsymbol{x})\right)^2
\end{aligned}$$

Hence, we see that

$$\mathbb{E}\left[(r(\boldsymbol{x}) - \hat{r}(\boldsymbol{x}))^2\right] = \mathsf{var}(\hat{r}(\boldsymbol{x})) + \mathsf{bias}(\hat{r}(\boldsymbol{x}))^2$$

Furthermore, the realized outcome can be written as

$$Y = r(\boldsymbol{x}) + \epsilon,$$

where $\mathbb{E}[\epsilon] = 0$ and $\mathbb{E}[h(\boldsymbol{x})\epsilon] = 0$ for any function h of $\boldsymbol{x}$.

Hence, the mean-squared error between the output Y and the prediction $\hat{Y} = \hat{r}(\boldsymbol{x})$ is

$$\begin{aligned}
\mathsf{MSE} &= \mathbb{E}\left[(Y - \hat{r}(\boldsymbol{x}))^2\right] \\
&= \mathbb{E}\left[(r(\boldsymbol{x}) + \epsilon - \hat{r}(\boldsymbol{x}))^2\right] \\
&= \mathbb{E}\left[(r(\boldsymbol{x}) - \hat{r}(\boldsymbol{x}))^2\right] + \mathbb{E}[\epsilon^2]
\end{aligned}$$

Note that $\mathbb{E}[\epsilon^2] = \mathsf{var}(\epsilon^2)$ because ϵ has mean 0.

Hence, we see that

$$\mathsf{MSE} = \mathsf{var}(\epsilon^2) + \mathsf{var}(\hat{r}(\boldsymbol{x})) + \mathsf{bias}(\hat{r}(\boldsymbol{x}))^2$$

$$\text{MSE} = \text{var}(\epsilon^2) + \text{var}(\hat{r}(\boldsymbol{x})) + \text{bias}(\hat{r}(\boldsymbol{x}))^2$$

In words, the MSE is given by the variance of the error term plus the variance of the prediction plus the squared bias of the prediction.

The accuracy of the prediction is worse if

(i) The inherent variance in Y through ϵ (the irreducible error) increases

(ii) The sampling variability of the estimated regression function $\hat{r}(\boldsymbol{x})$ increases

(iii) The systematic bias in the prediction of $\hat{Y} = \hat{r}(\boldsymbol{x})$ increases

Note: We can do nothing about the variance of the error term ϵ. Even if we knew the regression function $r(\boldsymbol{x})$, we would still have a prediction error through the variance in Y through ϵ.

To be perfectly clear what the expectation used to characterize the mean-squared error means, we should write

$$\text{MSE} = \mathbb{E}_{\mathcal{D},Y}\left[\left(Y - \hat{r}(\boldsymbol{x})\right)^2\right]$$

The expectation is taken over realized training data sets, $\mathcal{D}$, and realized prediction cases, Y.

Hence, the MSE is an average (expected) measure of the prediction accuracy over many data sets and prediction problems = applications of the estimation method.

Special case: Linear regression model

Suppose the key condition, mean-independence of the error term is satisfied:

$$\mathbb{E}[\epsilon|\boldsymbol{x}] = 0$$

We already know that in this case the OLS estimator is unbiased:

$$\mathbb{E}[\hat{\boldsymbol{\beta}}] = \boldsymbol{\beta}$$
$$\Rightarrow \text{bias}(\hat{r}(\boldsymbol{x})) = \mathbb{E}[\hat{r}(\boldsymbol{x})] - r(\boldsymbol{x}) = 0$$

Also, according to the *Gauss-Markov Theorem*, the OLS estimator is BLUE = the **best linear unbiased estimator**, i.e. the estimator with the smallest variance among all linear unbiased estimators.

Implication: If we require our estimator to be unbiased, then the *OLS estimator is the best linear estimator*

The "old school" thinking was that unbiasedness was the most important quality measure of an estimator. Non-linear estimation methods were hard to work with, and hence the OLS estimator was it.

But modern statistics recognizes that the price for an unbiased estimator can be high. If we tolerate some (hopefully small) amount of bias, we can find alternative estimators with lower variance, and the overall quality of the prediction will increase.

Regularization is a key technique to construct such estimators.

Ridge regression

In a ridge regression the estimated coefficients, $\boldsymbol{\beta}_{\text{ridge}}$, are found by minimizing the objective:

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{k=1}^{p} x_{ik}\beta_k \right)^2 + \lambda \sum_{k=1}^{p} \beta_k^2$$

The first term is the sum of squared residuals (RSS) — well known from the linear regression model.

The second term is a **penalty term**, which depends on the magnitude of the coefficients (β_k^2) and the tuning parameter $\lambda \geq 0$. The penalty allows for **regularization**.

Note: There is no penalty for the intercept.

If $\lambda = 0$, the penalty is zero, and minimization of the RSS yields the OLS (ordinary least-squares) estimator. To minimize the objective, the regression coefficients, $\boldsymbol{\beta}_{\text{OLS}}$, are chosen to make the squared residuals as small as possible.

If $\lambda > 0$, then the penalty term increases. Therefore, increasing the magnitude of a coefficient (measured by the square β_k^2) away from 0 has a cost. Minimizing the squared residuals needs to be balanced with minimizing the cost of the coefficient values.

Note: A ridge regression should be performed based on *standardized (scaled) inputs*:

$$x'_{ik} = \frac{x_{ik}}{\text{sd}(x_k)}$$

This makes the magnitudes and hence the cost of all inputs comparable.

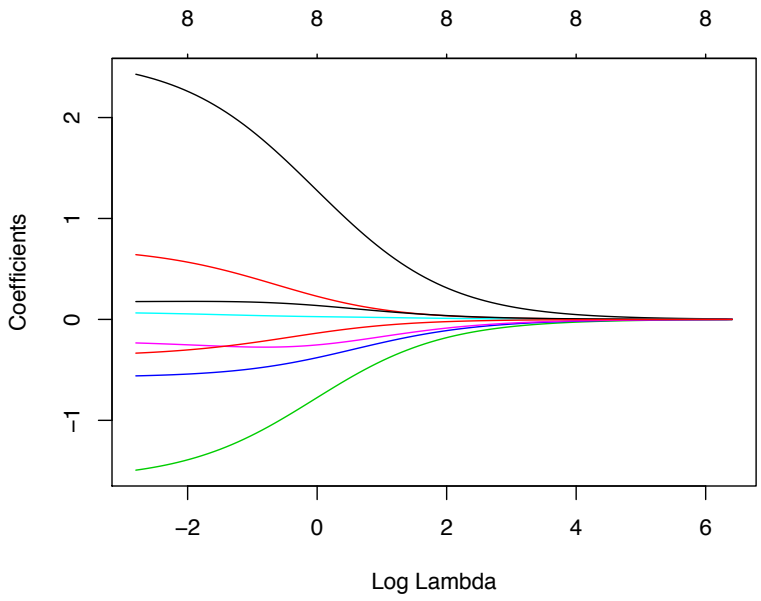
Regularization and shrinkage methods

Ridge estimators are **shrinkage estimators**. If λ increases, the regression coefficients β_k are shrunk towards zero.

We thus achieve **regularization**, in the sense that we constrain or stabilize the β_k values.

At the limit, $\lambda = \infty$, the price to be paid for the coefficients is so high that they will all be set to zero.

Example: Regression with 8 inputs. As (log) λ increases, the coefficients are shrunk towards zero.



Model exploration

Explore how the model works using the examples in the `Regularization-Variable-Selection.Rmd` R Markdown file

Compare the OLS estimates with the ridge regression estimates: Even parameters that do not influence the output, $\beta_k = 0$, will sometimes have large magnitudes and be statistically different from zero. This phenomenon is called **overfitting**. Overfitting is more likely to occur if the model is very complex, i.e. includes a large number of inputs that have no (or very little) impact on the output. Regularization reduces the model complexity and ameliorates overfitting.

Compare the predicted MSE (mean-squares error) of OLS with the predict MSE of the ridge regression.

Compare the results under different degrees of model complexity (change number of inputs).

Selection of tuning parameter λ

Can we use the training data to directly estimate both the regression coefficients $\boldsymbol{\beta}$ and the tuning parameter $\lambda \geq 0$?

$$\min_{\boldsymbol{\beta}, \lambda \geq 0} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{k=1}^{p} x_{ik} \beta_k \right)^2 + \lambda \sum_{k=1}^{p} \beta_k^2$$

What estimate of λ do you expect if you try this approach?

Cross-validation

Take the training data and *randomly* split them into K **folds**.

Most commonly used: $K = 10$

Choose some value for the tuning parameter λ.

Pick one of the folds, k, and set it aside as a validation data set. Estimate the model only using the data in the other folds, $1, \ldots, k-1, k+1, \ldots, 10$

Then predict the output y_i in fold k based on the model estimates, and record the MSE (mean-squared error).

Repeat for all folds k, and compute the average MSE over all folds. We thus obtain an **out-of-sample prediction error** for the chosen tuning parameter λ.

Selection of λ

Select a range of tuning parameter λ values, $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_L)$, and split the training data into K folds.

Iterate over all folds $k = 1, \ldots, K$

- For each value of λ_l in $\boldsymbol{\lambda}$:
  1. Train (estimate) the model using all folds but fold k
  2. Predict the output, $y_{\lambda_l,-k}(\boldsymbol{x}_i)$, for all observations $\boldsymbol{x}_i$ in fold k. Then calculate the total prediction error in fold k,

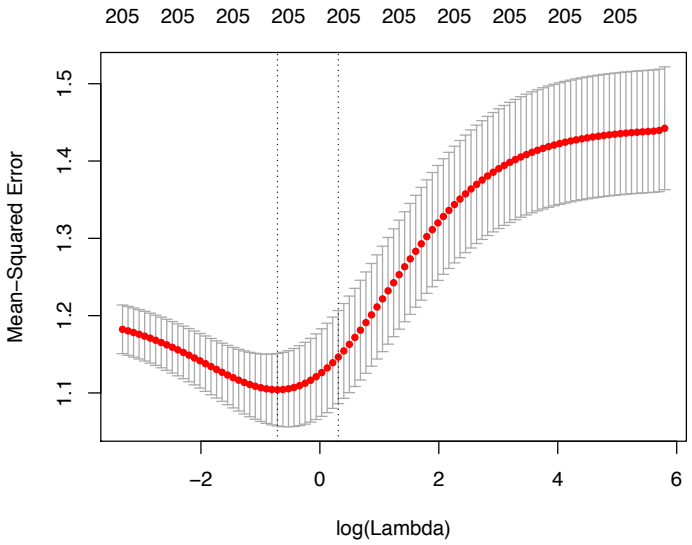
$$e_k(\lambda_l) = \sum_{i \in \text{fold}(k)} \left(y_i - y_{\lambda_l,-k}(\boldsymbol{x}_i)\right)^2$$

Calculate the cross-validation error for each λ_l,

$$CV(\lambda_l) = \frac{1}{n} \sum_{k=1}^{K} e_k(\lambda_l)$$

$CV(\lambda_l)$ is the out-of-sample MSE given the tuning parameter λ_l

Select tuning parameter λ based on $CV(\lambda)$, the **cross-validation error curve**



Plot indicates λ_{min} that minimizes $CV(\lambda)$ and λ_{1se} for most regularized model such that $CV(\lambda_{\text{1se}})$ is within one standard error of the minimum.

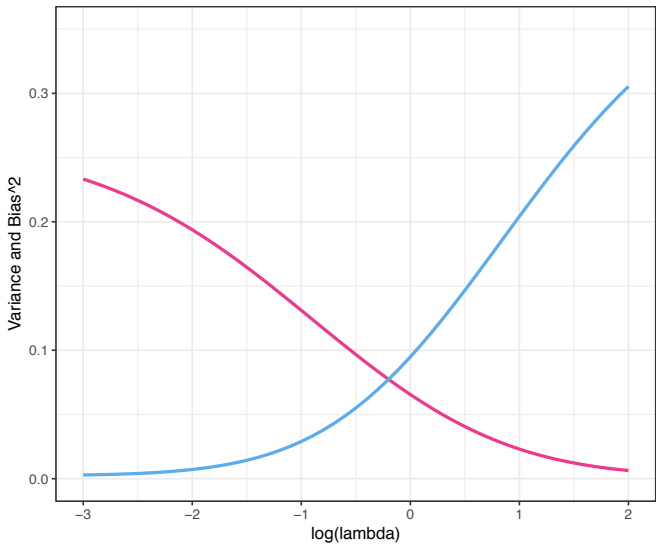
The ridge regression is able to achieve a smaller out-of-sample prediction error by imposing a penalty on the regression coefficients through the tuning parameter λ — regularization.

Hence, a ridge regression is able to predict better than the OLS estimator, which is unbiased under sufficient conditions.

Why can a biased estimator predict better than a biased estimator?

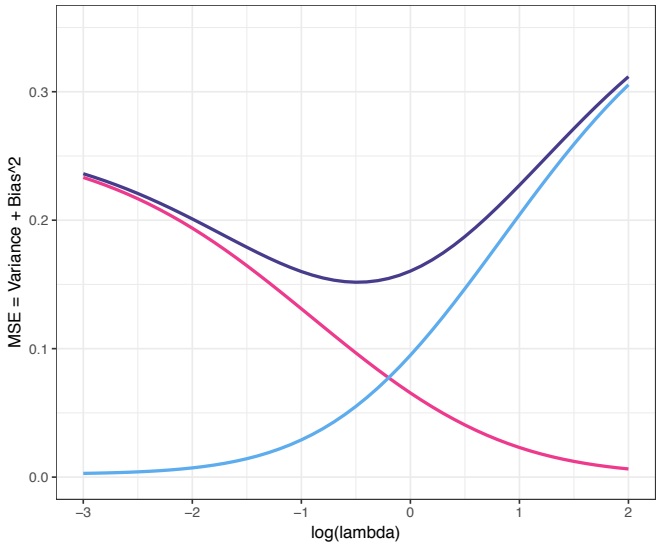
Bias-variance tradeoff

Plot shows the squared bias, $\text{bias}(\hat{r}(\boldsymbol{x}))^2$, and variance, $\text{var}(\hat{r}(\boldsymbol{x}))$, conditional on (log) λ (averaged over many data sets)



Pink: $\text{var}(\hat{r}(\boldsymbol{x}))$, blue: $\text{bias}(\hat{r}(\boldsymbol{x}))^2$

Ignoring the irreducible error variance $\text{var}(\epsilon)$, the mean-squared error is $\text{MSE} = \text{var}(\hat{r}(\boldsymbol{x})) + \text{bias}(\hat{r}(\boldsymbol{x}))^2$. By trading off an increase in the bias in exchange for a reduction in the variance of the prediction we can lower the prediction error.



Dark blue: MSE

Variable selection

Especially when p is large compared to n, or if there are good reasons to believe that many of the coefficients are zero, $\beta_k = 0$, we are interested in variable selection methods.

Variable selection: Estimator sets some of the estimated coefficients to zero.

Can a ridge regression perform variable selection? — Examine the marginal cost of increasing β_k in the ridge regression penalty factor:

$$\frac{d}{d\beta_k}\lambda\sum_{k=1}^{p}\beta_k^2 = 2\lambda\beta_k$$

Hence, the marginal cost of increasing β_k at $\beta_k = 0$ is $2\lambda \cdot 0 = 0$. Therefore, even though a ridge regression shrinks the regression coefficients, it will never set all coefficients to exactly zero.

This can be a disadvantage if many coefficient are in fact zero. Also, the regression results can be harder to **interpret** if no variable selection is performed.

The LASSO

The LASSO estimator, $\boldsymbol{\beta}_{\text{LASSO}}$, minimizes the objective:

$$\min_{\boldsymbol{\beta}}\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{k=1}^{p}x_{ik}\beta_k\right)^2 + \lambda\sum_{k=1}^{p}|\beta_k|$$

This appears very similar to a ridge regression. Ridge regression uses the penalty $\sum\beta_k^2$, called an ℓ_2 norm, while the LASSO uses the ℓ_1 norm $\sum_k|\beta_k|$.

LASSO stands for *Least Absolute Selection and Shrinkage Operator.*

Notes:

- The intercept is not penalized
- If the tuning parameter $\lambda = 0$ we obtain the OLS estimator
- To make the inputs comparable they need to be *standardized (scaled)*:

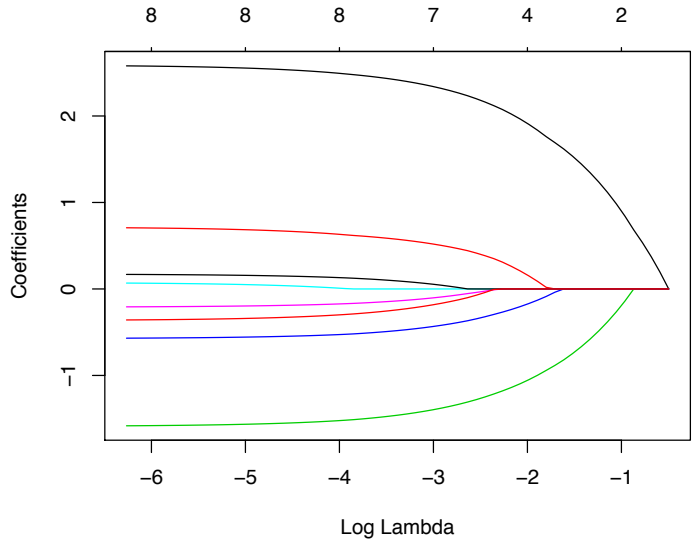
$$x'_{ik} = \frac{x_{ik}}{\text{sd}(x_k)}$$

The LASSO is qualitatively different from a ridge regression. In particular, the marginal cost of increasing β_k (in absolute value) is λ and does not vary with the level of β_k. Hence, even at $\beta_k = 0$ the marginal cost is λ.

Like the ridge regression estimator, the LASSO is a shrinkage estimator: As λ increases, the regression coefficients β_k are shrunk towards zero (**regularization**).

However, because the cost of the coefficient values is always constant at λ (even for tiny values of β_k) the LASSO will set the coefficients exactly to zero if λ is large enough.

Example: LASSO regression with 8 inputs. Once (log) λ is large enough, all coefficients are set to exactly zero.



The variable-selection property of the LASSO can be useful to **interpret** the regression results.

Model exploration

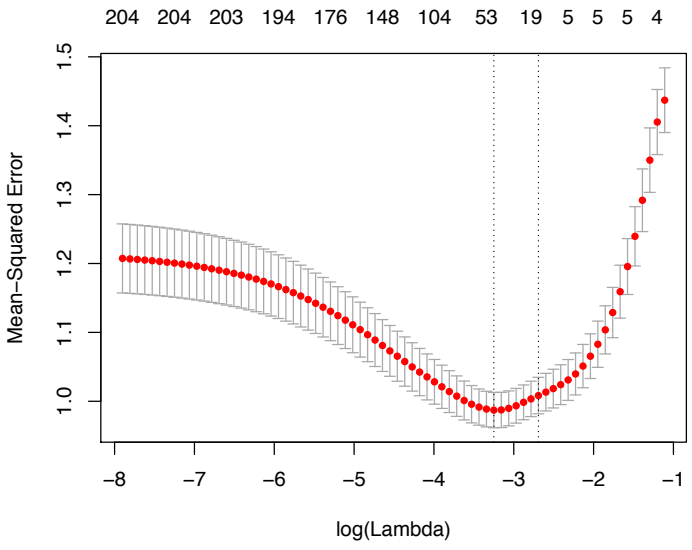
Explore how the model works using the examples in `Regularization-Variable-Selection.Rmd`.

Compare the LASSO with the ridge regression estimates.

Compare the prediction MSE (mean square error) of OLS, ridge regression, and the LASSO.

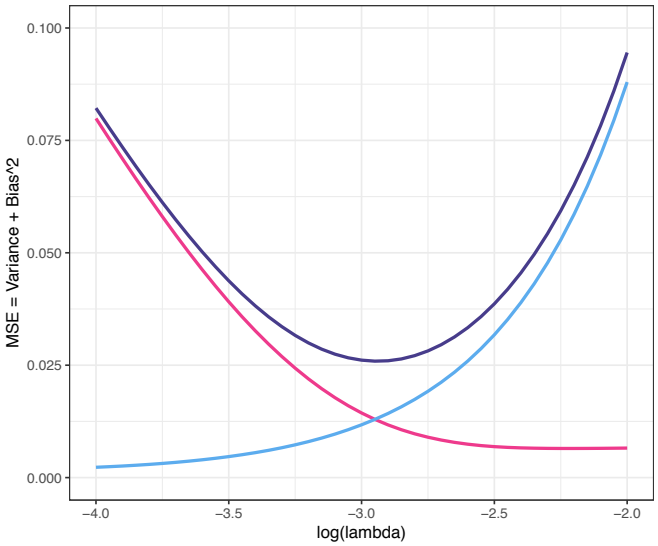
Compare the results under different degrees of model complexity (number of inputs) and different parameter vectors.

Tuning λ using the cross-validation error curve



Plot indicates λ_{min} that minimizes $CV(\lambda)$ and λ_{1se} for most regularized model such that $CV(\lambda_{\text{1se}})$ is within one standard error of the minimum.

Bias-variance tradeoff



Other variable selection methods: Best subset selection

Algorithm:

1. Fit the null model $\mathcal{M}_0$ that contains no inputs—fit based on sample mean of all observations
2. For all values $k = 1, \ldots, p$:
   (i) Fit all models that contain exactly k inputs
   (ii) Find the model with k inputs that has the smallest RSS, and call it $\mathcal{M}_k$
3. Compare all models $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ and choose the one that is "best" based on one of the following:
   - Cross-validated error
   - AIC (Akaike information criterion) or BIC (Bayesian information criterion)

Limitations: Need to evaluate fit for 2^p subsets

With $p = 25$ inputs, $2^{25} = 33,554,432$ (OK)

$p = 50$: $2^{50} = 1,125,899,906,842,624$ (more than 1,125 trillion) subsets

As p becomes large, best subset selection becomes computationally infeasible

Alternatives to best subset selection

Forward stepwise selection

- Start with the null model $\mathcal{M}_0$ that contains no inputs
- Successively add predictors to improve model fit

Backward stepwise selection

- Start with the full model that contains all inputs, $\mathcal{M}_p$
- Successively remove predictors

These approaches are computationally much less intensive, but they are not guaranteed to find the best subset of predictors.

For details, consult *An Introduction to Statistical Learning* (James et al.) or *The Elements of Statistical Learning* (Hastie et al.)

Variable selection based on p-values

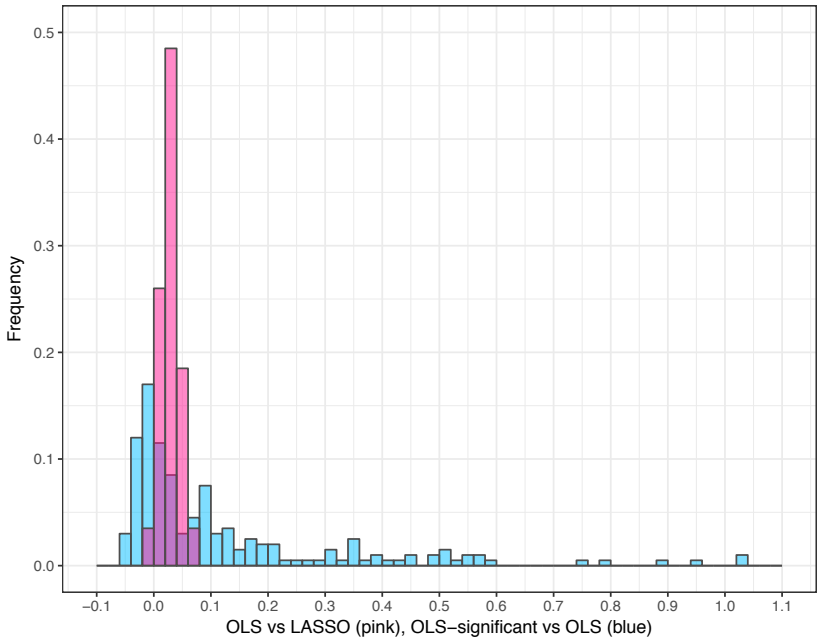
Why not select variables based on p-values, i.e. if an input is statistically different from 0 at the level α (e.g. $\alpha = 0.05$)?

This is a very tempting approach.

However, variable selection based on p-values is a bad idea:

- A hypothesis test, $H_0 : \beta_k = 0$, is not intended to directly assess how well input x_k predicts the output.
- We select variables based on an arbitrary significance level α, not a tuning parameter λ that is chosen to minimize the out-of-sample prediction error.
- We include inputs without predictive power if the estimated value happens to be large (i.e. when the inclusion is particularly damaging) and we exclude inputs if the estimated value happens to be small, even though out-of-sample the input explains much of the variation in the data.

Example in `Regularization-Variable-Selection.Rmd`: Compare difference in MSE between (i) OLS and LASSO and (ii) OLS with p-value based selection and OLS across 200 simulated data sets



The LASSO does best, and often the p-value selection method does very poorly compared to OLS:

	MSE
OLS	1.107
OLS/p-value sel.	1.229
LASSO	1.079

Elastic net

An elastic net can be thought of as a combination of a ridge regression and a LASSO. The penalty factor in an elastic net has the form:

$$\lambda \left(\sum_{k=1}^{p} \left[\alpha |\beta_k| + (1-\alpha)\beta_k^2 \right] \right)$$

for a parameter α such that $0 \leq \alpha \leq 1$.

Special cases: $\alpha = 1$ (LASSO), $\alpha = 0$ (ridge regression). The prediction quality of the model may be improved if $0 < \alpha < 1$ — test this in `Regularization-Variable-Selection.Rmd`.

The α parameter can be supplied using the `alpha` option in `glmnet` or in `cv.glmnet`.

Regularized regression for classification

Regularized regression models can easily be applied to classification problems.

To estimate a regularized logistic regression model in `glmnet` (LASSO, ridge regression, or an elastic net) supply the option `family = "binomial"`

Summary

- Regularization techniques stabilize or constrain the parameter estimates
  - Introduces more bias in prediction
- Bias-variance tradeoff: Allow for some bias if the reduction in the variance of the prediction is sufficiently high — improve overall MSE of prediction
- Key techniques:
  - LASSO — variable selection
  - Ridge regression
  - Elastic net
- Selection of tuning parameters using cross-validation