ST559 HW 9

Nick Sun

March 12, 2019

Faraway 10.6

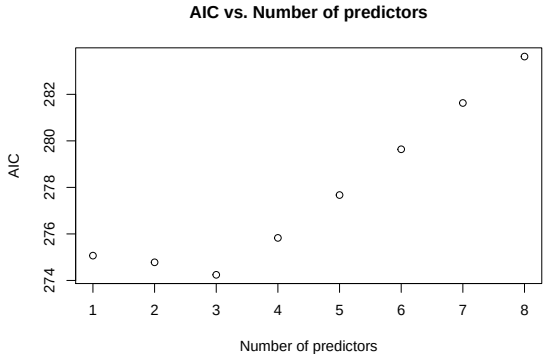
Here we want to model `hipcenter` in the `seatpos` data using all of the other predictors in the dataset. Our full model looks like this:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	436.4	166.6	2.62	0.01384
Age	0.7757	0.5703	1.36	0.1843
Weight	0.02631	0.331	0.0795	0.9372
HtShoes	-2.692	9.753	-0.2761	0.7845
Ht	0.6013	10.13	0.05936	0.9531
Seated	0.5338	3.762	0.1419	0.8882
Arm	-1.328	3.9	-0.3405	0.7359
Thigh	-1.143	2.66	-0.4297	0.6706
Leg	-6.439	4.714	-1.366	0.1824

Leg length appears to have a negative effect on seated height. A cm increase in leg length decreases the mean seated height by .242 cm.

For a person whose value are the means of all the predictor variables, the prediction interval is computed as:

fit	lwr	upr
-164.9	-243	-86.73

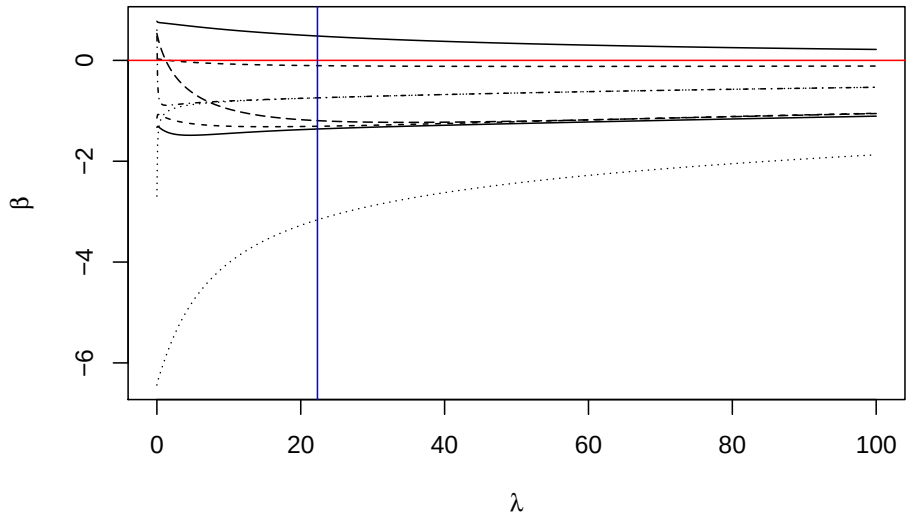


AIC is minimized at 3 predictors, so our model selected by this method is `hipcenter ~ Intercept + Age + Height`.

fit	lwr	upr
88.95	85.17	92.74

Faraway 11.3

Using the same `seatpos` data from above, we will fit a ridge regression model.



Our optimal λ which minimizes the generalized cross validation score is around 22.3 Using this ridge model, we can calculate a prediction for the given data. Our predicted value comes out to:

```
##           [,1]
## [1,] -194.4909
```

Faraway 11.4 (a, b, e)

Taking another look at good old `fat` data. We will be doing linear regression with all predictors, linear regression with variable selection done by AIC, and ridge regression.

We have removed every tenth observation to create a test set with a nice `seq` function. We will test each model on this test set and calculate the RMSE.

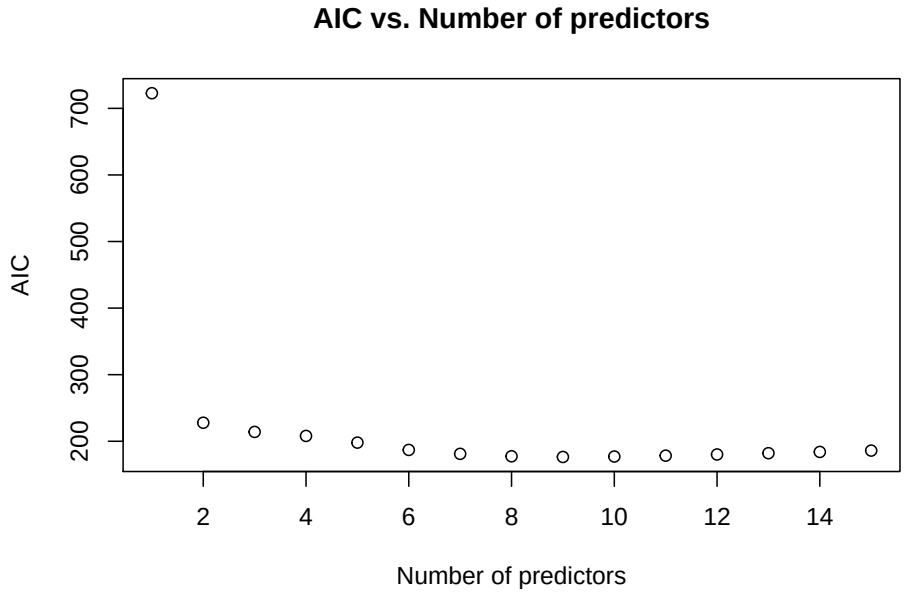
Linear model

For the linear model, we simply fit a model with all predictors. The coefficients for this model look like this:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-6.612	6.409	-1.032	0.3034
age	0.004228	0.01142	0.3702	0.7116
weight	0.3879	0.02359	16.44	1.34e-39
height	0.03349	0.03822	0.8763	0.3818
adipos	-0.4708	0.1059	-4.444	1.428e-05
free	-0.5736	0.01439	-39.86	6.561e-100
neck	-0.02331	0.08403	-0.2774	0.7817

	Estimate	Std. Error	t value	Pr(>|t|)
chest	0.1229	0.03721	3.304	0.001119
abdom	0.1058	0.03844	2.751	0.006455
hip	-0.004548	0.05427	-0.08381	0.9333
thigh	0.1763	0.05107	3.452	0.0006725
knee	0.02536	0.09073	0.2795	0.7802
ankle	0.111	0.09534	1.164	0.2458
biceps	0.1382	0.06158	2.244	0.02586
forearm	0.2048	0.0695	2.947	0.003572
wrist	0.165	0.2031	0.8121	0.4176

AIC selected linear model



The 9 parameter model has the lowest AIC so we will use that as our final model. The coefficients for that model are given here:

Table 5: Table continues below

(Intercept)	weight	adipos	free	chest	abdom	thigh	ankle
-2.919	0.3925	-0.5277	-0.5698	0.1246	0.1179	0.1561	0.1475

biceps	forearm
0.149	0.2146

Ridge regression model

Table 7: Table continues below

	age	weight	height	adipos	free	neck	chest
-7.252	0.004087	0.3844	0.03527	-0.4666	-0.572	-0.02201	0.1238

abdom	hip	thigh	knee	ankle	biceps	forearm	wrist
0.1085	-0.002021	0.1765	0.02743	0.1136	0.1394	0.2056	0.1627

After creating these three models and calculating the MSE using the same test data set, we get the following table:

Linear	AIC	Ridge
1.946	1.989	1.937

The model with the least RMSE is actually ridge regression.

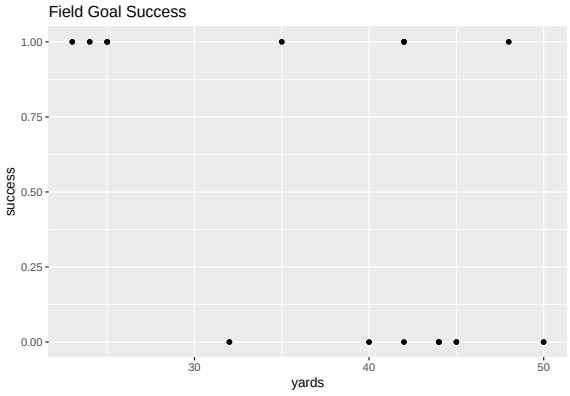
Logistic Regression

In regular OLS regression, we have the assumption that the errors are normally distributed. For some data, this isn't going to work. For example, say we have data where the response is binary i.e. either a 0 or a 1. We want to make a model that will give us predicted probabilities of getting a 1 for a certain values of our covariates. Since our response is binary, the errors are definitely not going to be normally distributed! To make a model for this data, we can use *logistic regression* which basically allows us to model probabilities using a linear equation. The way we can do this is by transforming our response with the logit function which looks like:

$$logit(p) = log\left(\frac{p}{1-p}\right)$$

Example

Let's say we have field goal data from the Oregon State Football team. We have one predictor, the distance of the try. Our response is whether or not the kick is successful.



We can fit a logistic model using the `glm()` function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	5.778	3.329	1.736	0.08262
yards	-0.1474	0.08281	-1.78	0.07508

Our coefficient estimate for the `yards` is the effect of a unit increase in a single yard on the log odds of making the field goal. According to this model, as we increase the yardage, the field goal success percentage on average goes down.

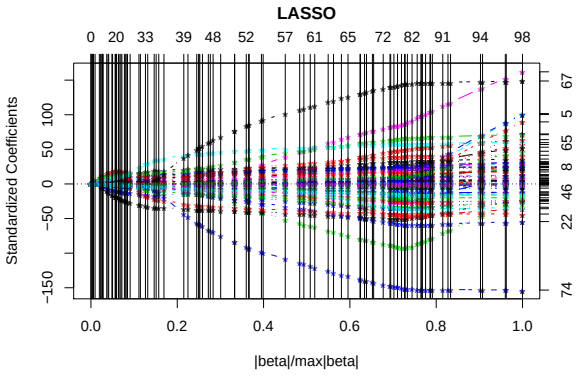
Lasso Regression

Lasso Regression is based around adding a *penalty term* to our OLS objective function that we are trying to minimize. This penalty term is a prespecified value λ and the sum of the absolute values of our coefficient estimates. The gist is that this penalty discourages the model from having large coefficient estimate - it shrinks our $\hat{\beta}$. This is useful because Lasso can shrink some coefficients to zero, meaning that it does simultaneous model fitting and variable selection! Finding a good value of λ is key and in practice we usually try a bunch of candidate values and calculate the cross-validation mean squared error for each. The λ with the lowest MSE is our best guess for the optimal penalty constant.

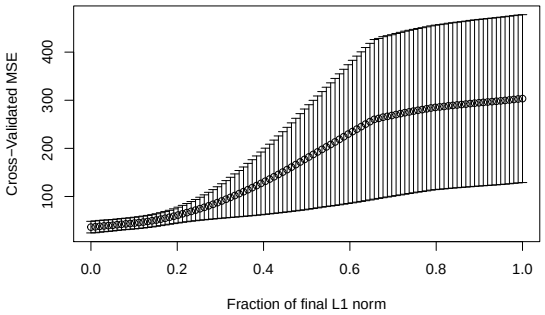
Example

Here's an example with metagenomics data. Each column in our matrix represents the relative proportion of a certain phylotype of algae in a sample of coral. We want to make a model that predicts the amount of bleaching in the coral (for our purposes, we will treat this as a continuous variable). Lasso is a great tool here because we have lots of potential predictors, and definitely want to have some form of variable selection.

Here we will use the `lars` package to create a Lasso regression model.



This plot is a fairly cluttered, but the main idea is that as we increase the penalty parameter (go from right to left), a lot of our parameters represented by the various colored lines shrink to 0!



Sadly, Lasso might not be the best technique to use here since as we can see the MSE is actually minimized when $\lambda = 0$. Ah well, that's messy genomics data for ya.