01:45 Friday 4th December, 2015
See updates and corrections at http://www.stat.cmu.edu/~cshalizi/mreg/

Lecture 26: Variable Selection

36-401, Fall 2015, Section B

1 December 2015

Contents

1 What Variable Selection Is

"Variable selection" means selecting which variables to include in our model (rather than some sort of selection which is itself variable). As such, it is a special case of model selection. People tend to use the phrase "variable selection" when the competing models differ on which variables should be included, but agree on the mathematical form that will be used for each variable — e.g., temperature might or might not be included as a predictor, but there is no question about whether, if it is, we'd use temperature or temperature^2 or $\log$ temperature.

Since variable selection is a special case of model selection, and we've talked extensively about model selection already (see especially lecture 21), these notes can be briefer than usual.

2 Why Variable Selection Using p-Values Is a Bad Idea

When we assume the linear, constant-variance, independent-Gaussian-noise model is completely correct, it is easy to test the hypothesis that any particular coefficient is zero. The (Wald) test statistic is

$$\frac{\hat{\beta}_i}{\widehat{\mathrm{se}}\left[\hat{\beta}_i\right]}$$

and, under the null hypothesis that $\beta_i = 0$, this has a $t_{n-(p+1)}$ distribution, therefore tending to a z (standard-Gaussian) distribution as $n \rightarrow \infty$.

It is very, very tempting, and common, to use the p-values which come from this test to select variables: significant variables get included, insignificant ones do not, ones with smaller p-values (hence larger test statistics) are higher priorities to include than ones with smaller test statistics. This pattern of reasoning shows up over and over again among users of regression, including, I am ashamed to say, not a few statisticians.

The reasons why this is a bad idea were already gone over in lecture 15, so, again, I will be brief. Let us think about what will tend to make the test statistic larger or smaller, by being more explicit about the denominator:

$$\frac{\hat{\beta}_i}{\frac{\hat{\sigma}}{\sqrt{n\widehat{\mathrm{Var}}[X_i]}}\sqrt{VIF_i}}$$

where $\widehat{\mathrm{Var}}\left[X_i\right]$ is the sample variance of the i^{th} predictor variable, and VIF_i is that variables variance-inflation factor (see Lecture 17). What follows from this?

1. Larger coefficients will, all else being equal, have larger test statistics and be more significant ($\hat{\beta}_i$ in the numerator).

2. Reducing the noise around the regression line will increase all the test statistics, and make every variable more significant ($\hat{\sigma}$ in the denominator).

3. Increasing the sample size will increase all the test statistics, and make every variable more significant ($\sqrt{n}$ in the denominator).

4. More variance in a predictor variable will, all else being equal, increase the test statistic and make the variable more significant ($\widehat{\mathrm{Var}}\left[X_i\right]$ in the denominator).

5. More correlation between X_i and the other predictors will, all else being equal, decrease the test statistic and make the variable less significant (VIF_i in the denominator).

The test statistic, and thus the p-value, runs together an estimate of the actual size of the coefficient with how well we can measure that particular coefficient. This is exactly the right thing to do if our question is "Can we reliably detect that this coefficient isn't exactly zero?" That is a very, very different question from "Is this variable truly relevant to the response?", or even from "Does including this variable help us predict the response?" Utterly trivial variables can show up as having highly significant coefficients, if the predictor has lots of variance and isn't very correlated with the other predictors. Very important (large-coefficient) variables can be insignificant, when their coefficients can't be measured precisely with our data. *Every* variable whose coefficient isn't exactly zero will eventually (as $n \rightarrow \infty$) have an arbitrarily large test statistic, and an arbitrarily small p-value[1]

None of this is even much help in answering the question "Which variables help us predict the response?", let alone "Which variables help us explain the response?"

None of this is fixed by using F-tests on groups of coefficients, rather than t-tests on individual coefficients.

3 Cross-Validation Instead

If we want to use our models to make predictions, then what we want to know is how well the model will predict new data. The C_p statistic and AIC attempt to estimate this, using how well the model predicted the old data, plus adjustments based on theory. Cross-validation estimates how well the model will predict new data by predicting new data. This is, unsurprisingly, a very good way of estimating how well the model will predict.

The two main forms of cross-validation are leave-one-out, which we have already discussed in detail, and k-fold cross-validation, which we have spent less time on in class but is described in Lecture 21. They each have their strengths and weaknesses (which is why we have both).

- k-fold CV is fast (the model gets fit only k times, and typically k is 5 or 10); it is also "consistent for variable selection", meaning that if one of the models presented to it contains all the relevant predictors, and *only* the relevant predictors, then the probability of picking that right model $\rightarrow 1$ as $n \rightarrow \infty$. On the other hand, it tends to give somewhat worse predictions than leave-one-out, especially when all the models are wrong.

- Leave-one-out can be slow (because the model must be fit n times), *except* for linear regression where there is a short-cut formula. LOOCV is *in*-consistent for variable selection: even with unlimited amounts of data, it tends to include more variables than are necessary, though it will tend to include all the relevant variables. The model it picks tends to have lower prediction errors on new data than those picked by k-fold CV.

[1] "Oh my God, it's full of stars." — David Bowman, on increasing his sample size to 2001.

As discussed in Lecture 21, C_p and AIC are best seen as approximations to leave-one-out, which avoid the step of re-fitting the model, or even of calculating the short-cut formula (which still involves summing over every data point).

4 Stepwise Variable Selection

"Stepwise" or "stagewise" variable selection is a family of methods for adding or removing variables from a model sequentially.

Forward stepwise regression starts with a small model (perhaps just an intercept), considers all one-variable expansions of the model, and adds the variable which is best according to some criterion. This criterion might be "lowest p-value", "highest adjusted R^2", "lowest Mallow's C_p", "lowest AIC", "lowest score under cross-validation", etc. This process is then repeated, always adding one variable at a time, until the criterion stops improving. In **backwards** stepwise regression, we start on the contrary with the largest model we're willing ton contemplate, and keep eliminating variables until we no longer improve. The obvious **forward-backward** or **mixed** stepwise variable selection procedure will contemplating both adding and removing one variable at each step, and take the best step.

In a forward-backward algorithm we could easily add one variable, then add or remove another, and then remove the first variable we'd added. This is because these stepwise algorithms only look at models which are close (one variable away from) the variable we started with[2] In principle, we could just look at all possible linear models based on a given set of variables, and compute our criterion (adjusted R^2, C_p, AIC, LOOCV, etc.) for each one of them; this is called **all-subsets** variable selection, because each model corresponds to a subset of the variables. With p variables there are 2^p possible models, so all-subsets regression becomes, literally, exponentially more time-consuming with more variables; this is the only real justification for the stepwise procedures.

4.1 Stepwise Selection in R

The simplest function for stepwise model selection is the `step` function, which is built in to R. It can do forward or backward selection, or both, and you can specify both the smallest model to consider (so those variables are always included), and the largest. It can, however, only use AIC or BIC as the selection criteria.

Here's an example of how it works[3], for the real estate data set from homework 8[4].

[2]As search algorithms, they are "greedy".

[3]The `trace` argument controls how much `step` prints out as it tries various models. Larger values print out more information; the default, `trace=1`, is already a lot. Setting it to zero suppresses this. I urge you to re-run these examples with `trace=1`, but I doing so would substantially lengthen these notes.

[4]But without any attempt at cleaning the data by removing outliers, etc.; this is just to illustrate the syntax, not as a full-scale data analysis.

```
real.estate <- read.csv("http://www.stat.cmu.edu/~cshalizi/mreg/15/hw/08/real-estate.csv")
# Fit a "kitchen sink" model
  # But don't try to use the ID numbers as a predictor variable!
(realty.lm.all <- lm(Price ~ . -ID, data=real.estate))

##
## Call:
## lm(formula = Price ~ . - ID, data = real.estate)
##
## Coefficients:
##     (Intercept)             Sqft          Bedroom         Bathroom
##      -2.390e+06        1.075e+02       -9.712e+03       -1.067e+02
## Airconditioning           Garage             Pool        YearBuild
##      -1.222e+04        1.732e+04        1.249e+04        1.279e+03
##         Quality              Lot       AdjHighway
##      -5.390e+04        1.422e+00       -2.717e+04

step(realty.lm.all, direction="backward", trace=0)

##
## Call:
## lm(formula = Price ~ Sqft + Bedroom + Garage + YearBuild + Quality +
##     Lot, data = real.estate)
##
## Coefficients:
## (Intercept)         Sqft      Bedroom       Garage    YearBuild
##  -2.233e+06    1.093e+02   -1.007e+04    1.665e+04    1.191e+03
##     Quality          Lot
##  -5.223e+04    1.415e+00
```

By comparison to the kitchen sink model, this drops bathrooms, air-conditioning, the pool, and adjacency to highways.

Of course, we could start with a very simple model and expand:

```
(realty.lm.minimal <- lm(Price ~ 1, data=real.estate))

##
## Call:
## lm(formula = Price ~ 1, data = real.estate)
##
## Coefficients:
## (Intercept)
##      277894

step(realty.lm.minimal, scope=list(upper = realty.lm.all,
                            lower= realty.lm.minimal), direction="forward",
     trace=0)

##
## Call:
```

```
## lm(formula = Price ~ Sqft + Quality + YearBuild + Lot + Garage +
##     Bedroom, data = real.estate)
##
## Coefficients:
## (Intercept)         Sqft      Quality    YearBuild          Lot
##  -2.233e+06    1.093e+02   -5.223e+04    1.191e+03    1.415e+00
##      Garage      Bedroom
##   1.665e+04   -1.007e+04
```

This begins with an intercept-only model, and then adds variables. Here giving a lower limit to the scope is pretty much superfluous, we could just give it an upper limit, but it doesn't hurt.

Of course, we can also ask `step` to consider both adding and subtracting variables:

```
step(realty.lm.minimal, scope=list(upper = realty.lm.all,
                            lower= realty.lm.minimal),
     direction="both", trace=0)
```

```
##
## Call:
## lm(formula = Price ~ Sqft + Quality + YearBuild + Lot + Garage +
##     Bedroom, data = real.estate)
##
## Coefficients:
## (Intercept)         Sqft      Quality    YearBuild          Lot
##  -2.233e+06    1.093e+02   -5.223e+04    1.191e+03    1.415e+00
##      Garage      Bedroom
##   1.665e+04   -1.007e+04
```

(This just so happens to reach the same answer as only doing forward selection.)

If we want to only consider models which include certain terms, we can do that through changing the lower limit of the `scope`:

```
(realty.lm.comforts <- lm(Price ~ Pool+Airconditioning, data=real.estate))
```

```
##
## Call:
## lm(formula = Price ~ Pool + Airconditioning, data = real.estate)
##
## Coefficients:
##     (Intercept)             Pool  Airconditioning
##          188852            64335           101760
```

```
step(realty.lm.comforts, scope=list(upper=realty.lm.all,
                             lower=realty.lm.comforts), direction="both",
     trace=0)
```

```
##
## Call:
## lm(formula = Price ~ Pool + Airconditioning + Sqft + Quality +
##     YearBuild + Lot + Garage + Bedroom, data = real.estate)
##
## Coefficients:
##     (Intercept)             Pool  Airconditioning             Sqft
##      -2.346e+06        1.279e+04       -1.176e+04        1.080e+02
##         Quality        YearBuild              Lot           Garage
##      -5.388e+04        1.256e+03        1.389e+00        1.724e+04
##         Bedroom
##      -9.756e+03
```

The `step` function is a simplified version of the function `stepAIC` in the MASS package, which works very similarly but is more flexible. The `leaps` package contains an even more flexible function, `subsetreg`, which tries to determine the lowest-MSE model at any given number of variables, and then lets you chose how to trade the number of parameters against MSE.

5 Inference after Selection, Again

The standard inferential statistics (like the p-values on individual coefficients) are only valid if the model is chosen independent of the data being used to calculate them. If there is any sort of data-dependent model selection, whether stepwise variable selection or something else, they are no longer valid. This applies even to eliminating variables because their coefficients are insignificant. If we do go ahead and use the same data twice, once to pick a model and once to test hypotheses about that model, we will get confidence intervals which are systematically too narrow, p-values which are systematically too small, etc. (See Lecture 21 for more discussion, and an example of how doing model selection on pure noise can lead to apparently highly-significant results.)

The easy cure, as discussed in Lecture 21, is to split the data in half at random, and use one part to do model selection and the other half to do inference for your selected model. Again, there is nothing about variable selection which makes this any different.

6 Further Reading

In general, all the references for Lecture 21 are relevant again.

For a vivid example of just how badly misleading selecting variables based on statistical significance can be, see **?**.