REVOLUTION ANALYTICS

Modeling in Revolution R Enterprise Module 5: Stepwise Regression: Linear, GLM, and Logistic

Stepwise Regression

Sometimes models may contain extraneous predictor variables that unnecessarily complicate our formula without substantially increasing our ability to explain our response variable. In this case, Stepwise Regression can provide a viable method to reduce model complexity by deleting superfluous predictor terms.

In this module, we will consider inclusion and exclusion of response variables in our statistical models. Up to now, our examples have largely concerned only a small subset of potential response variables to predict an outcome, and it can be useful to expand the number of response variables sometimes.

Stepwise Regression

Some things to keep in mind when adding additional variables to our models:

- Adding variables will always either increase or, at worst, cause the R-squared value for the model to remain the same.
- At the same time, adding additional variables inherently increases the complexity of any statistical model.

Added Complexity

It may seem tempting to add as many variables as possible to any model in the hopes of raising R-squared. Indeed, wouldn't an increase in our model's ability to account for the variance in the observational data worth any added complexity?

- Not necessarily. When making predictions, the additional variables suddenly require the additional collection of data, which in application may be much more costly or even unfeasible.
  - Also, if you have a model explaining, to a high degree of accuracy, a predictor variable by a small number of response variables, sometimes that is more desirable than having a model explaining the same predictor variable with limited improvement in accuracy.

Adding Response Variables

Ultimately, the statistician must determine the attributes and detriments of adding or deleting variables. Usually, the debate surrounds two main points:

1. What is the added benefit, in terms of accuracy, of keeping the debated response variable?
2. Is it worth the added complexity?

After considering both of these points, an informed decision can be made on whether or not to keep the debated response variable.

Model Selection: Stepwise Regression

This process may be tedious to perform for a large number of variables, especially for a large data set. Statisticians have developed algorithms to help users with the process of determining which predictor variables to keep and which to discard, and these processes are referred to as Stepwise Regression.

There are three main choices one has when dealing with variable selection, and we will consider each of these in a greater detail:

- Forward Selection
- Backwards Selection
- Stepwise (a combination of forward and backward selection, bidirectional)

Model Selection: Stepwise Regression

In ScaleR, stepwise linear regression is implemented by the appropriate regression function and rxStepControl.

- Using the correct regression function, add the variableSelection argument to the function call for stepwise regression.
- Using rxStepControl, specify the method, the scope, and various control parameters. The default method, “stepwise”, specifies a bidirectional search, while the scope provides upper and lower formulas for the search.

Methods of Variable Selection

There are three methods of variable selection supported by ScaleR:

- “forward”: variables are added onto the minimal model one at a time until either no additional variable satisfies the selection criterion or until the maximal model is reached.
- “backward”: variables are removed from the maximal model one at a time until either the removal of another variable won’t satisfy the selection criterion or until the minimal model is reached.
- “stepwise”: combination of forward and backward variable selection; that is, variables are added to the minimal model, but at each step the model is reanalyzed to re-evaluate whether previously added variables should be deleted from the updated (current) model.

Example: Stepwise Regression

For this example, let’s construct a multivariate linear model predicting balance from all of the other variables in the Bank data set. We can anticipate that this model could be potentially too complex with little added capability in predictability. Consequently, we will implement stepwise regression, using a scope specifying the lower bound as the minimal model and an upper bound as the maximal model. Note that the SigLevel step criterion is identical to using AIC:

```
infile <- file.path("data", "BankXDF.xdf")
BankDS <- RxXdfData(file = infile)

subLinMod <- rxLinMod(balance ~ age, data = BankDS, variableSelection = rxStepControl(method = "s
    scope = ~age + job + marital + education + default + housing + loan + contact +
        day + month + duration + campaign + pdays + previous + poutcome + y,
    stepCriterion = "SigLevel"))
```

```
## Rows Read: 10000, Total Rows Processed: 10000, Total Chunk Time: 0.005 seconds
## Rows Read: 10000, Total Rows Processed: 20000, Total Chunk Time: 0.047 seconds
## Rows Read: 10000, Total Rows Processed: 30000, Total Chunk Time: 0.023 seconds
```

Example: Stepwise Regression

This is a long summary, so let's consider the output:

```
summary(subLinMod)
```

```
## Call:
## rxLinMod(formula = balance ~ age, data = BankDS, variableSelection = rxStepControl(method = "stepw
##     scope = ~age + job + marital + education + default + housing +
##         loan + contact + day + month + duration + campaign +
##         pdays + previous + poutcome + y, stepCriterion = "SigLevel"))
##
...
```

Example: Stepwise Regression

Notice that only some values of months are determined to be significant. For instance, January clients tend to have smaller account balances, whereas November clients tend to have larger balances. However, for values like March, the response variable is not significant enough to have an noticeable effect on our model.

```
summary(subLinMod)$balance$coefficients[4:14, ]
```

```
##              Estimate Std. Error t value  Pr(>|t|)
## month=jun    102.16       135.3  0.7549 4.503e-01
## month=jul   -604.51       131.5 -4.5985 4.267e-06
## month=aug   -392.52       131.1 -2.9936 2.759e-03
## month=oct    311.47       165.3  1.8841 5.955e-02
## month=nov    849.92       134.8  6.3071 2.870e-10
...
```

Example: Stepwise Regression

Similarly, clients owning a home tend to have lower account balances.

```
summary(subLinMod)$balance$coefficients[26:27, ]
```

```
##              Estimate Std. Error t value  Pr(>|t|)
## housing=yes    -181.1      33.75  -5.367 8.048e-08
## housing=no         NA         NA      NA        NA
```

Example: Stepwise Regression

Notice that the R-squared value equals only about 5 percent, so even after the inclusion of all possible predictor variables our model still only explain about 5-percent of the variance contained in the observable data.

```
summary(subLinMod)$balance$r.squared
```

```
## [1] 0.0502
```

This value can be improved by considering more relevant statistical models; for instance, we have already discussed how age and balance are not linearly related, and therefore a non-linear regression would be more applicable.

Exercise: Stepwise Regression

In this exercise, construct a similar model as the one above using all of the predictor variables in the Bank data set, except poutcome, only this time predict whether or not a client will subscribe to a term deposit (the y variable). Use a combination of forward and backward selection, and because of the binary response variable, remember to use logistic regression rather than linear regression.

. .

Exercise: Stepwise Regression

In this exercise, construct a similar model as the one above using all of the predictor variables in the Bank data set, except poutcome, only this time predict whether or not a client will subscribe to a term deposit (the y variable). Use a combination of forward and backward selection, and because of the binary response variable, remember to use logistic regression rather than linear regression.

- Hint: start with a minimal model, as above, and define your scope to be the maximum number of response variables in the data set, excluding poutcome.

Exercise: Solution

Constructing the logistic model,

```
subLogMod <- rxLogit(y ~ age, data = BankDS, variableSelection = rxStepControl(method = "stepwise
    scope = ~age + job + marital + education + default + housing + loan + contact +
        day + month + duration + campaign + pdays + previous + balance, stepCriterion = "SigLevel
```

```
## Rows Read: 10000, Total Rows Processed: 10000, Total Chunk Time: 0.004 seconds
## Rows Read: 10000, Total Rows Processed: 20000, Total Chunk Time: 0.006 seconds
## Rows Read: 10000, Total Rows Processed: 30000, Total Chunk Time: 0.005 seconds
## Rows Read: 10000, Total Rows Processed: 40000, Total Chunk Time: 0.005 seconds
## Rows Read: 5211, Total Rows Processed: 45211, Total Chunk Time: 0.004 seconds
##
...
```

And beginning the analysis for the summary:

```
summary(subLogMod)
```

```
## Call:
## rxLogit(formula = y ~ age, data = BankDS, variableSelection = rxStepControl(method = "stepwise",
##     scope = ~age + job + marital + education + default + housing +
##         loan + contact + day + month + duration + campaign +
```

Predictions

To finish our discussion on regression, let's lastly mention the rxPrediction function.

What if we want to calculate predictions for our fitted models? Luckily, ScaleR has us covered with the rxPredict function, which basically takes a model object an outputs predicted values and residuals:

```
rxPredict(modelObject, data = targetDataSet, outData = targetDataFileName, computeResiduals = TRU
```

Conceptual Example: Predictions

For some applications, we may wish to test the effectiveness of our model on a similar data set. One technique is pulling a sub-sample of data from the original data set, constructing a model, and testing the rigidity of that model on the entire data set. The rxPredict function is very useful for this.

Conceptual Example: Predictions

The model object is simply the name of the model we have defined in our ScaleR command. For instance, our last GLM model we created was named glmMod, and that would be used as the input to the rxPredict command. Other things to keep in mind are:

- Data should be defined as your target data set, or entire data set if continuing the example above.
- Computing the residuals allows you to conduct model evaluation, an important topic we will cover in a later module.

Recap

Let's review some of the concepts covered in this module:

- Is it possible to model too many variables?
- What is the benefit of excluding a variable from a model?
- What are the three methods compatible with Stepwise Regression in ScaleR?

REVOLUTION ANALYTICS

Thank you

Revolution Analytics is the leading commercial provider of software and support for the popular open source R statistics language.

www.revolutionanalytics.com, 1.855.GET.REVO, Twitter: @RevolutionR