McMaster University

Department of Mathematics and Statistics

STATISTICS 3A03: Applied Regression Analysis with SAS

Fall 2017

SAS Lab 3. September 26-29, 2017

Topics Covered in this Lab

1. The Data step
2. Using SAS to get p-values or critical values
3. Multiple Regression
4. Prediction

1. The Data Step

The `Data` step can be used to directly input the values for a dataset.

```
Data Mydata;
Input Y X1 X2 X3;
Datalines
6.2 0 1 2
5.1 0 2 1
4.6 0 1 1
7.3 1 1 0
8.1 1 1 1
;
```

There are some cases where this is useful (see Section 3 below) but it is uncommon to directly input a dataset of any realistic size since it is very error-prone and usually we get the data as some sort of file.

More commonly the `Data` step is used to create new datasets that add extra columns to existing datasets. These extra columns are often found by calculations based on columns in an existing dataset. There are a large number of functions that can be used in these calculations. Most have reasonably obvious names (such as `Log Sqrt Sin Cos` etc.). Note that `Log` is the natural log (base e), for log base 10 or 2 use the `Log10` or `Log2` functions respectively. All standard mathematical operations (`+ - * / **`) can also be included in the calculations.

Table 6.9 in your textbook gives a dataset relating the number of supervisors (Y) and the number of workers supervised (X) in 27 industrial establishments. The data are on the website in the file `Table6_9.txt`. For reasons that we will examine later in the course it is required to fit the multiple regression model

$$\log(Y) \;=\; \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$$

but the datafile only gives us X and Y so we need to construct $\log(Y)$ and X^2. I assume we have already used `PROC IMPORT` to read in the file to the dataset `S3A3.Tab6_9`.

```
Data S3A3.Tab6_9a;
   Set S3A3.Tab6_9;
   Logy=log(Y);
   X2= X*X;
run;
```

The `Set` Statement specifies the original dataset which will be used. The subsequent two statements create the two new columns `Logy` and `X2`. We will see further uses of the `Data` step later in this lab.

2. Using the data step to get p-values or critical values
As we have seen, SAS automatically gives the p-values to test if a given coefficient (intercept or slope) is equal to zero. If we want to do any other test, however, SAS does not give the p-value. Since p-values also cannot be calculated accurately from tables we would like to use SAS to give us those values. We can do that in a data step in SAS. First note that we can actually save the table of coefficient estimates and standard errors output by SAS (as well as other parts of the output) as a SAS dataset. Let us consider Pearson's Height data as an example. Here I assume that this dataset is called `S3A3.Heights`. The `ODS OUTPUT` statement can be used to save portions of the usual output into a new SAS dataset.

```
PROC REG Data=S3A3.Heights plots=none;
   Model DHeight=MHeight;
   ODS OUTPUT ParameterEstimates=S3A3.Heights_Parests
run;
quit;
```

This code will create a new dataset S3A3.Heights_Parests with two rows and a number of columns most of which correspond to what you see when SAS prints the results of a regression to the screen. Let us now suppose that we wish to test if the coefficients are 1 rather than 0. We need to calculate the new test statistic and the associated p-value which we can do in a DATA step using the `CDF` function to calculate the required probabilities.

```
DATA S3A3.Heights_Parests;
   Set S3A3.Heights_Parests;
   tValue1 = (Estimate-1)/StdErr;
   pvalue1 = 2*CDF('t', -abs(tValue1), 1373);
run;
```

Note that it is best to define the p-value as $p = 2\Pr(T \leq -|t|)$ since the `CDF` function calculates probabilities to the left of a point.

We can then print this new dataset as usual

```
PROC Print Data=S3A3.Heights_Parests;
run;
```

In this example we see that the p-values come out to be extremely small and so we would reject the null hypotheses.

A related question may be how to get the critical values (also called quantiles) of a t distribution to calculate confidence intervals. We can do that in a similar way using the `quantile` function to get critical values. Note that this should be calculated using $1 - \alpha/2$ since quantiles are calculated as the value that cuts off a certain probability to the left rather than the right.

```
DATA S3A3.Heights_Parests;
   Set S3A3.Heights_Parests;
   critValue = quantile('t', 0.975, 1373);
   CI_lower = Estimate-critValue*StdErr;
   CI_upper = Estimate+critValue*StdErr;
run;
```

Printing this dataset shows us that the critical value here is 1.9617 (very close to the standard normal value of 1.9600 since the degrees of freedom are so high) and gives us the two confidence intervals.

3. Multiple Regression

Multiple regression using `PROC REG` is done exactly as simple regression. The only difference is in the Model Statement which now becomes something like

```
Model Y=X1 X2 X3;
```

It is often useful in multiple regression to plot each of the variables against each other in a scatterplot matrix. This can be done using `PROC SGSCATTER`. In the following example we look at the first year university GPA, high school math and SAT math for the computer science students referred to in the third set of lecture notes. I assume the data have been imported into the SAS dataset `S3A3.csdata`.

```
PROC SGSCATTER Data=S3A3.csdata;
    Matrix GPA HSM SATM;
run;
```

The plots in the scatterplot matrix are useful but they do not tell us much about the model since all covariates in the model are adjusted for the other covariates. To examine the fit of the model it is common to plot the dependent variable against the fitted values from the model. This compares the observed response to the predicted response if the linear model holds and so will tell us similar things to the plot of dependent variable against covariate in simple regression. It is more useful than the individual plots since the fitted values accounts for all of the covariates together. SAS uses the keyword `Predicted.` (**NB: The period at the end is essential!**) to refer to these fitted values within the call to `PROC REG`. Including the statement

```
PLOT Y*Predicted.;
```

after the `Model` statement in the call to `PROC REG` will produce this plot.

```
PROC REG Data=S3A3.csdata plots=none;
    Model GPA=HSM SATM;
    Plot GPA*Predicted.;
run;
quit;
```

4. Prediction

In both simple and multiple linear regression, we often wish to predict the dependent variable for certain values of the covariate(s). We can this quite easily for the actual covariate values used in the fitting using the `OUTPUT` statement in `PROC REG`. We can also add lower and upper prediction limits using `LCL` and `UCL` and lower and upper confidence limits for the predicted mean using `LCLM` and `UCLM`. The default is for 95% intervals but this can be changed using the `alpha=` option in the `PROC REG` statement. Note that the confidence level for intervals is $100(1-\alpha)$ where α is the value given in the `alpha=` option so the default is `alpha=0.05`.

For the CSData example we could do

```
PROC REG Data=S3A3.csdata plots=none;
    Model GPA=HSM SATM;
    OUTPUT OUT=csdata_out
        Predicted=Fitted
        LCLM=Lower_Est
        UCLM=Upper_Est
        LCL=Lower_Pred
        UCL=Upper_Pred;
run;
quit;
```

In many situations, however, we want to estimate the mean and/or predict an individual value for covariates that are not part of the original dataset. In doing this it is important that the covariate values being used are within the range of those used in the model estimation since we have no information if the model is even valid outside of that range.

The easiest way to get SAS to do this prediction or estimation is to create a new dataset with the new values of the covariates, then put the two datasets together. If the new (extra) values are missing the dependent variable they will be ignored in the analysis. However, if we ask to save the predicted values in an output dataset, they will have predicted values as long as they have values for all of the covariates. We can specify that the output dataset only includes rows for those observations missing the dependent variable so we can easily see just the required intervals.

Here is an example using the dataset `FtCollinsSnow.txt` which was on your first assignment. Suppose that we wish to predict late snowfall when early snowfall is 1, 3, 6, 12, 18, 24, 30, 36, 42, or 48 inches.

```
Data NewSnow;
Input Early;
Datalines;
1
3
6
12
18
24
30
36
42
48
;

Data NewSnow;
Set NewSnow S3A3.Snow;

PROC REG Data=Snow plots=none noprint;
Model Late=Early;
OUTPUT Out=SnowPred(where=(Late=.))
   Predicted=Pred
   LCLM=LowerCI
   UCLM=UpperCI
   LCL=LowerPI
   UCL=UpperPI;
run;
quit;

PROC PRINT Data=SnowPred;
run;
```

Exercises

1. The dataset `Fuel2001.txt` has data on the 50 US states and the District of Columbia from 2001. The variables are

Drivers	Number of Licensed Drivers
FuelC	Total amount of gasoline sold for road use (thousands of gallons).
Income	Per person income (thousands of $).
Miles	Miles of Federal-aid highways in the state.
Pop	State population
Tax	State gasoline tax rate (cents per gallon)
State	State Name.

   (a) Create new variables Fuel, DLic and LogMiles defined as

$$\begin{aligned} \texttt{Fuel} &= \frac{1000 * \texttt{FuelC}}{\texttt{Pop}} \\ \texttt{DLic} &= \frac{1000 * \texttt{Drivers}}{\texttt{Pop}} \\ \texttt{LogMiles} &= \log_2(\texttt{Miles}) \end{aligned}$$

   (b) Construct a matrix of pairwise scatter-plots of the variables `Fuel`, `Tax`, `DLic`, `Income` and `LogMiles`.

   (c) Find the correlations between each pair of variables in the previous part.

   (d) Fit a linear model to predict `Fuel` using the other four variables from parts (b) and (c) and construct a plot of the observed `Fuel` values against the fitted values from your model.

2. Using the `Heights.txt` dataset, predict the daughters height for mothers with heights equal to 58,59,60,61,62,63 and 64 inches. Give 99% prediction intervals as well as 99% confidence intervals for the mean heights of daughters born to mothers with each of these heights.