

Applied Regression Modeling: A Business Approach

Computer software help: SAS code

SAS (originally Statistical Analysis Software) is a commercial statistical software package based on a powerful programming interface. The following instructions provide SAS code to carry out all the procedures covered in the book. SAS also has an easy-to-use graphical user-interface in its Analyst Application—this is covered in a separate Help document. The following instructions are based on SAS 9.1 for Windows. Further information is available at www.sas.com.

Getting started and summarizing univariate data

- 1 If desired, change SAS's default **options** by selecting Tools > Options.

There are different ways to import data, for example use File > Import Data to open text data files or Excel spreadsheets (when you are prompted to "Choose the SAS destination" type a name for the dataset into the "Member" box).

Once you have successfully imported some data, you conduct analyses by writing lines of code in an "Editor" window and then submitting the code by selecting Run > Submit (or simply clicking the "running person" Submit button). You can save the code you write into a text ".sas" file (recommended).

- 2 You can access **help** by selecting Help > SAS Help and Documentation.

- 3 To **transform data** or compute a **new variable**, type, for example,

```
data mydata2;
set work.mydata;
logX = log(X);
Xsq = X**2;
run;
```

for the natural logarithm of X and X^2 respectively. If you get a "syntax error" message this means there is a syntax error in your expression—a common mistake is to forget the multiplication symbol (*) between a number and a variable (e.g., $2 * X$ represents $2X$).

To create **indicator (dummy) variables** from a qualitative variable, type,

```
data mydata2;
set work.mydata;
if X='level' then D1=1;
else D1=0;
run;
```

where X is the qualitative variable and "level" is the name of one of the categories in X . Repeat for other indicator variables (if necessary).

- 4 Calculate **descriptive statistics** for quantitative variables by typing

```
proc univariate data=mydata;
var Y;
run;
```

where Y is the quantitative variable.

Specify an `output` statement to calculate other statistics beyond those calculated by default (see SAS Help for specific details on how to do this).

- 5 Create **contingency tables** or **cross-tabulations** for qualitative variables by typing

```
proc freq data=mydata;  
tables X1*X2;  
run;
```

where X1 and X2 are the qualitative variables.

- 6 If you have a quantitative variable and a qualitative variable, you can calculate **descriptive statistics** for cases grouped in different categories by typing

```
proc univariate data=mydata;  
var Y;  
by X notsorted;  
run;
```

where Y is the quantitative variable and X is the qualitative variable.

Specify an `output` statement to calculate other statistics beyond those calculated by default (see SAS Help for specific details on how to do this).

- 7 To make a **stem-and-leaf plot** for a quantitative variable, type

```
proc univariate data=mydata plot;  
var Y;  
run;
```

where Y is the quantitative variable (for large sample sizes, SAS will create a horizontal bar chart instead of a stem-and-leaf plot).

To make a **histogram** for a quantitative variable, type

```
proc univariate data=mydata noprint;  
histogram Y / endpoints=1 to 4 by 1;  
run;
```

where Y is the quantitative variable and `endpoints` specifies how to construct the breakpoints.

- 8 To make a **scatterplot** with two quantitative variables, type

```
proc gplot data=mydata;  
plot Y*X;  
run;
```

where Y is the vertical axis variable and X is the horizontal axis variable.

All possible scatterplots for more than two variables can be drawn simultaneously (called a **scatterplot matrix**) by typing

```
ods html;  
ods graphics on;  
proc corr data=mydata plots=matrix;  
var Y X1 X2;  
run;
```

```
ods graphics off;
```

```
ods html close;
```

where Y, X1, and X2 are quantitative variables.

- 9 You can **mark or label cases** in a scatterplot with different colors/symbols according to categories in a qualitative variable. For example, in a dataset of 20 observations, suppose X2 contains values 1–4 to represent four categories, and Y and X1 are two quantitative variables. Then the following code produces a scatterplot with numbers (representing the value of X2) marking the points:

```
proc gplot data=mydata;
plot Y*X1 = X2;
run;
```

To change the colors/symbols used submit code like the following before drawing the plot:

```
symbol1 c=circle c=black;
symbol2 v=star c=red;
symbol3 v=square c=blue;
symbol4 v=plus c=green;
```

There does not appear to be an easy way to identify individual cases after drawing a scatterplot using SAS code.

- 10 To make a **bar chart** for cases in different categories, use either the `gbarline` or `chart` procedures. For frequency bar charts of one qualitative variable, type:

```
proc gbarline data=mydata;
bar X1;
run;
or:
```

```
proc chart data=mydata;
vbar X1;
run;
```

where X1 is a qualitative variable. For frequency bar charts of two qualitative variables, type:

```
proc chart data=mydata;
vbar X1 / group=X2;
run;
```

where X1 and X2 are qualitative variables (the `gbarline` procedure does not offer this option).

The bars can also represent various summary functions for a quantitative variable. For example, to produce a bar chart of means, type:

```
proc gbarline data=mydata;
bar X1 / sumvar=Y type=mean;
run;
or:
```

```
proc chart data=mydata;
vbar X1 / group=X2
sumvar=Y type=mean;
run;
```

where X1 and X2 are the qualitative variables and Y is a quantitative variable.

- 11 To make **boxplots** for cases in different categories, use the `boxplot` procedure.

For just one qualitative variable, type:

```
proc boxplot data=mydata;
plot Y*X1;
run;
```

where `Y` is a quantitative variable and `X1` is the qualitative variable. For two qualitative variables, first sort the dataset by the values of one of the qualitative variables, `X2` say:

```
proc sort data=mydata out=mydata2;
by X2;
run;
```

and then type:

```
proc boxplot data=mydata2;
plot Y*X1 (X2);
run;
```

- 12 To make a **QQ-plot** (also known as a **normal probability plot**) for a quantitative variable, type:

```
proc univariate data=mydata noprint;
qqplot Y / normal(mu=est sigma=est color=red);
run;
```

where `Y` is a quantitative variable.

- 13 To compute a **confidence interval** for a univariate population mean, type:

```
proc univariate data=mydata cibasic(alpha=0.05);
var Y;
run;
```

where `Y` is the variable for which you want to calculate the confidence interval, and `alpha` is the confidence level of the interval.

- 14 To do a **hypothesis test** for a univariate population mean, type:

```
proc univariate data=mydata mu0=0;
var Y;
run;
```

where `Y` is the variable for which you want to do the test and `mu0` is the (null) hypothesized value.

Simple linear regression

- 15 To fit a **simple linear regression model** (i.e., find a least squares line), type

```
proc reg data=mydata;
model Y=X;
run;
```

where `Y` is the response variable and `X` is the predictor variable.

- 16 To add a **regression line** or **least squares line** to a scatterplot, first use the option `i=r` when specifying `symbol1`, for example, `symbol1 v=circle c=black i=r`. Then construct the scatterplot using computer help #8.

- 17 To find **95% confidence intervals for the regression parameters** in a simple linear regression model, type:

```
proc reg data=mydata alpha=0.05;
model Y=X / clb;
run;
```

where Y is the response variable, X is the predictor variable, and alpha is the confidence level of the intervals.

This applies more generally to multiple linear regression also.

- 18 To find a **confidence interval for the mean of Y** at a particular value of X in a simple linear regression model, type:

```
proc reg data=mydata alpha=0.05;
model Y=X / clm;
run;
```

where Y is the response variable, X is the predictor variable, and alpha is the confidence level of the interval. The confidence intervals for the mean of Y at each of the X-values in the dataset are displayed as two columns headed CL Mean.

You can also obtain a confidence interval for the mean of Y at an X-value that is not in the dataset by doing the following. Before fitting the regression model, add the X-value to the dataset using code such as:

```
data mydata2;
input X;
datalines;
2
;
data mydata3;
set mydata mydata2;
run;
```

Then fit the regression model by following the steps above. SAS will ignore the X-value you added when fitting the model (since there is no corresponding Y-value), so all the regression output (such as the estimated regression parameters) will be the same. But SAS will calculate a confidence interval for the mean of Y at this new X-value based on the results of the regression.

This applies more generally to multiple linear regression also.

- 19 To find a **prediction interval** for an individual value of Y at a particular value of X in a simple linear regression model, type

```
proc reg data=mydata alpha=0.05;
model Y=X / cli;
run;
```

where Y is the response variable, X is the predictor variable, and alpha is the confidence level of the interval. The prediction intervals for an individual value of Y at each of the X-values in the dataset are displayed as two columns headed CL Predict.

You can also obtain a prediction interval at an X-value that is not in the dataset by doing the following. Before fitting the regression model, add the X-value to the dataset using code such as:

```
data mydata2;
input X;
datalines;
2
;
data mydata3;
set mydata mydata2;
run;
```

Then fit the regression model by following the steps above. SAS will ignore the X-value you added when fitting the model (since there is no corresponding Y-value), so all the regression output (such as the estimated regression parameters) will be the same. But SAS will calculate a prediction interval for an individual value of Y at this new X-value based on the results of the regression.

This applies more generally to multiple linear regression also.

Multiple linear regression

- 20 To fit a **multiple linear regression model**, type:

```
proc reg data=mydata;
model Y=X1 X2;
run;
```

where Y is the response variable and X1 and X2 are the predictor variables.

- 21 To add a **quadratic regression line** to a scatterplot, first use the option `i=rq` when specifying `symbol1`, for example, `symbol1 v=circle c=black i=rq`. Then construct the scatterplot using computer help #8.

- 22 Categories of a qualitative variable can be thought of as defining **subsets** of the sample. If there are also quantitative response and predictor variables in the dataset, a regression model can be fit to the data to represent separate regression lines for each subset. For example, suppose that X2 is a qualitative variable with four categories, and Y and X1 are two quantitative variables. Then the following code produces a scatterplot with different symbols and colors (representing the value of X2) marking the points, and four separate regression lines:

```
symbol1 c=black i=r;
symbol2 v=star c=red i=r;
symbol3 v=square c=blue i=r;
symbol4 v=plus c=green i=r;
proc gplot data=mydata;
plot Y*X1 = X2;
run;
```

- 23 To find the F-statistic and associated p-value for a **nested model F-test** in multiple linear regression, submit code such as the following:

```
proc reg data=mydata;
model Y={X1 X2} {X3 X4} / selection=forward
groupnames='X1 X2' 'X3 X4'
slentry=0.99;
run;
```

Here, X1 and X2 are in the reduced model, while X1, X2, X3, and X4 are in the complete model. The F-statistic is in the second row of the “Summary of Forward Selection Table” in the column headed F Value, while the associated p-value is in the column headed Pr > F.

- 24 To save **studentized residuals** in a multiple linear regression model type:

```
proc reg data=mydata;
model Y=X1 X2;
output out=work.res1 student=sresid;
run;
quit;
```

This code saves the studentized residuals to the SAS dataset *res1* (in the *work* library) as variable *sresid*, where they can now be used just like any other variable, for example, to construct residual plots (note that all the variables in the original dataset are included in the new dataset). (SAS also has a *rstudent* keyword, but that calculates “deleted studentized residuals.”)

- 25 It is possible to use the *loess* procedure to add a **loess fitted line** to a scatterplot (useful for checking the zero mean regression assumption in a residual plot). However, it is perhaps easier to use the following code to implement a smoothing spline interpolation routine. First use the option *i=sm50s* when specifying *symbol1*, for example, *symbol1 v=circle c=black i=sm50s*; (adjust the value “50” up or down to change the smoothness of the resulting line). Then construct the scatterplot using computer help #8.

- 26 To save **leverages** in a multiple linear regression model type:

```
proc reg data=mydata;
model Y=X1 X2;
output out=work.res1 h=lev;
run;
quit;
```

This code saves the leverages to the SAS dataset *res1* (in the *work* library) as variable *lev*, where they can now be used just like any other variable, for example, to construct scatterplots (note that all the variables in the original dataset are included in the new dataset).

- 27 To save **Cook’s distances** in a multiple linear regression model:

```
proc reg data=mydata;
model Y=X1 X2;
output out=work.res1 cookd=cooksd;
run;
quit;
```

This code saves the Cook's distances to the SAS dataset `res1` (in the `work` library) as variable `cooksd`, where they can now be used just like any other variable, for example, to construct scatterplots (note that all the variables in the original dataset are included in the new dataset).

- 28 To create **residual plots** automatically in a multiple linear regression model, type:

```
proc reg data=mydata;
model Y=X1 X2;
plot student.*predicted. student.*nqq. student.*cookd.;
run;
```

This produces a plot of studentized residuals versus fitted values, a QQ-plot of the studentized residuals, and a plot of studentized residuals versus Cook's distances.

To create manual residual plots, use computer help #24 to save studentized residuals and then create scatterplots with these studentized residuals on the vertical axis.

- 29 To create a **correlation matrix** of quantitative variables (useful for checking potential **multicollinearity** problems), type:

```
proc corr data=mydata;
var Y X1 X2;
run;
```

where `Y`, `X1`, and `X2` are quantitative variables.

- 30 To find **variance inflation factors** in multiple linear regression, type:

```
proc reg data=mydata;
model Y=X1 X2 / vif;
run;
```

where `Y` is the response variable and `X1` and `X2` are the predictor variables.

- 31 To draw a **predictor effect plot** for graphically displaying the effects of transformed quantitative predictors and/or interactions between quantitative and qualitative predictors in multiple linear regression, first create a variable representing the effect, say, "X1effect" (see computer help #3).

Next, sort the dataset by the values of `X1` (if necessary):

```
proc sort data=mydata out=mydata2;
by X1;
run;
```

If the "X1effect" variable just involves `X1` (e.g., $1+3X1+4X1^2$), type, for example:

```
symbol1 v=point c=black i=join;
proc gplot data=mydata2;
plot X1effect*X1;
run;
```

If the "X1effect" variable involves a qualitative variable (e.g., $1-2X1+3D2X1$, where `D2` is an indicator variable), type, for example:

```
symbol1 v=point c=black i=join;
symbol2 v=point c=red i=join;
proc gplot data=mydata2;
plot X1effect*X1=X2;
run;
```

See Section 5.4 for an example.