Print

Demo: Scoring Data Using PROC PLM

Filename: **st106d02.sas**

In the previous demonstration, we built a predictive model and created an item store. Now, we'll use the item store to score data with two different methods, and then compare the results to show equivalence between the two methods. For demonstration purposes, we'll score ameshousing4, the validation data set from the previous demonstration. Remember that in a business environment, you would score data that was not used in either training or validation.

PROC PLM RESTORE=*item-store* <*options*>;
    SCORE DATA=*SAS-data-set* <OUT=*SAS-data-set*>;
    CODE <FILE=*file-name*>;
RUN;

DATA <*data-set-name***>;**
    SET *SAS-data-set* <(*data-set-options*)>;
    %INCLUDE *source*;
RUN;

PROC COMPARE BASE=*SAS-data-set* **COMPARE=***SAS-data-set* **CRITERION=***value*;
    VAR *variable(s)*;
    WITH *variable(s)*;
RUN;

1. Open program st106d02.sas.

```
/*st106d02.sas*/  /*Part A*/

proc plm restore=STAT1.amesstore;
    score data=STAT1.ameshousing4 out=scored;
    code file="&homefolder\scoring.sas";
run;

data scored2;
    set STAT1.ameshousing4;
    %include "&homefolder\scoring.sas";
run;

proc compare base=scored compare=scored2 criterion=0.0001;
    var Predicted;
    with P_SalePrice;
run;
```

In the PROC PLM step, the RESTORE= option specifies amesstore, the item store that contains the model information. The SCORE statement scores the ameshousing4 data and creates a new data set, scored. This data set contains the input data and the scored variable, Predicted. The CODE statement

writes the scoring instructions to scoring.sas, the file named in the FILE= option. The scoring instructions are SAS DATA step programming statements that create a scoring variable. By default, the name of this new variable is the original variable name prefixed with P_.

2. Submit the PROC PLM step.

3. Review the output.

   The Store Information table shows the model parameters. The log shows that the data set, WORK.SCORED, and the SAS program, scoring.sas, were created.

4. Let's go back to the code. Next, we need a DATA step to read the input data and execute the scoring code. This DATA step reads ameshousing4 and creates a temporary data set, scored2. The %INCLUDE statement copies the scoring code from scoring.sas. Remember that if we made any transformations to the original data set before building the model, we would need to perform those transformations in the DATA step before the %INCLUDE statement.

5. Submit the DATA step.

6. Check the SAS log to verify that the output data set, WORK.SCORED2, was successfully created.

7. Let's see whether the two methods scored the data the same way. We'll use the COMPARE procedure to compare the values of the scored variable in the two output data sets. There's no need to do any preliminary matching or sorting in this case, because the output data sets are based on the same input data set. They have the same number of variables and observations, in the same order.

   In the PROC COMPARE statement, the BASE= option specifies scored, the data set created by the SCORE statement. The COMPARE= option specifies scored2, the data set created by the DATA step. By default, the criterion for judging the equality of the numeric values is .00001, but you can use the CRITERION= option to change this. In this example, we'll use 0.0001, which is less stringent than the default. The VAR statement names the scored variable in the BASE= data set, Predicted, and the WITH statement specifies P_SalePrice, the scored variable in the COMPARE= data set.
8. I'll submit the PROC COMPARE step.

9. Review the output.

10. In the Compare Summary table, let's look at the Values Comparison Summary to see whether the two methods produced similar predictions. All the compared values are, in fact, equal. Some values aren't exactly equal due to rounding, but as the maximum difference criterion value indicates, the differences are too small to be important. Of course, if we used the more stringent default criterion, the results would likely show more differences. We built a predictive model on training data, chose a best fitting and generalizable model according to validation data, and now we've seen multiple ways to deploy our predictive model. We can now predict new cases after we measure the model inputs by passing the new data to PROC PLM or a DATA step using score code. That is, we can predict home prices after we measure the home attributes that are needed as inputs in our predictive model. After predicting sale prices of homes in Ames, Iowa, we'll have some idea of the future commission for our real estate firm.

Close