



Using SAS[®] Viya[™] Models with SAS[®] Model Manager 14.2

Overview

This document shows how to use models that were created with SAS Visual Data Mining and Machine Learning procedures. First, you see how the procedures can be used to generate the SAS score code, input variables, and output variables that you need for a model. The document also explains how to remotely submit code from SAS Viya to SAS 9.4 using SAS/CONNECT. The result is a SAS model package (SPK) file. Finally, you see how to register the model in the SAS Model Manager 14.2 model repository, and then you can see the models in the SAS Model Manager web application.

Note that these procedures run on SAS Viya. When these procedures create SAS DATA step code, the code can be used with SAS Model Manager 14.2. For more information, see the [SAS Viya Documentation](#) page on support.sas.com.

Prerequisites for Running Code Examples

The code examples in this document are run on SAS Viya in a UNIX environment. Directory paths, servers, ports, and data set location depend on your environment and might need to be modified. To run the examples:

- Install both SAS/CONNECT and SAS Studio on your SAS Viya machine.
- Install both SAS/CONNECT and SAS Model Manager 14.2 on your SAS 9.4 machine.
- Ensure that your SAS 9.4 machine can access HMEQ project and training data sets or your own data sets in a directory such as `/home/SAS94`.
- Ensure that the SAS Viya machine can access the HMEQ training data set or your own training data set in a directory such as `/home/SASViya`.

The HMEQ_TRAIN, HMEQ_PROJECT_INPUT, and HMEQ_PROJECT_OUTPUT data sets from the Quick Start Tutorial ZIP file are used in the code examples. The ZIP file is available on the [SAS Model Manager Product Documentation](#) page. Note that the CIMPORT procedure is not available in SAS Viya. Therefore, if your SAS Viya and SAS 9.4 machines are different and you are in a UNIX environment, you must use the CPORT procedure on the SAS 9.4 machine and place the HMEQ data sets in a location that is accessible by both the SAS Viya machine and the SAS 9.4 machine. You can also copy the data sets from the SAS 9.4 machine to the SAS Viya machine. If you are in a Windows environment, the data sets are in the **Data** folder within the Quick Start Tutorial ZIP file. For more information, see [“Make the Tutorial Files Available” in SAS Model Manager: User’s Guide](#).

Create a Model in SAS Viya

You can use SAS Studio in SAS Viya to run the code that creates a model and its score code. The training data set must be in a CAS library, and users must be authorized to connect to a session on the SAS Cloud Analytic Services (CAS) server. For more information, see [SAS Cloud Analytic Services: Fundamentals](#) and [Getting Started with SAS Visual Data Mining and Machine Learning 8.1](#).

Create Macro Variables

This example uses macro variables in the SAS code; it is unnecessary to specify literals multiple times:

```
%let training_dataset=%quote(hmeq_train);
%let model_name1=logselect_badloans;
```

Generate SAS Score Code for a Model

The LOGSELECT procedure fits binary and binomial response models in SAS Viya. The procedure creates a SAS code file that can be used for scoring new data. This example demonstrates how to build a model, generate the SAS score code file, and create a sample of the training data set that is used when registering the model.

Note: For information about starting a CAS session and loading data, see [SAS Viya 3.1 Quick Start](#).

```
/* Uncomment and run this next statement to start a session, */
/* if you do not already have one. */
/*cas casauto; */

/* Create a libref that points to where your training data set is stored. */
libname home '/home/sasdemo/HMEQ';

/* Load the training data set into the Caslib CASUSER in session memory. */
proc casutil;
  load data=home.&training_dataset replace;
  list tables;
quit;

/* Assign a libref with the CAS LIBNAME engine to access the in-memory table. */
libname mycas cas;

/*Create the model file and the model output table. */
proc logselect data=mycaslib.&training_dataset;
  class job reason;
  model bad=delinq derog ninq clage clno debttinc loan mortdue value yoj job reason;
  partition frac(valid=0.2 seed=78900);
  code file="~/SASViya/&model_name1..sas" pcatall replace=true;
  output out=mycas.&model_name1._output copyvars=( _all_ ) role;
```

```
run;

/* Create a sample of the training data set. */
libname SASViya '~/SASViya';
data SASViya.training_data_subset;
  set mycaslib.&training_dataset(obs=10);
run;
```

Generated SAS Scoring Code

```
/*-----
Generated SAS Scoring Code
Date: 14NOV2016:16:41:53
-----*/

drop _badval_ _linp_ _temp_ _i_ _j_;
_badval_ = 0;
_linp_ = 0;
_temp_ = 0;
_i_ = 0;
_j_ = 0;

array _xrow_0_0_{19} _temporary_;
array _beta_0_0_{19} _temporary_ ( 4.84117022020336
-0.75527303150393
-0.75489831554785
-0.08378266495349
0.00551538465093
0.02704356320614
-0.10305346531347
0.00001254508488
-3.182502550997E-7
-1.1927173270401E-6
-0.00122031322848
0.78216672760293
1.05461905358194
0.80400513781889
0.27217114259894
-0.75976952374211
0
-0.45542539133306
0);

length _JOB_ $7; drop _JOB_;
_JOB_ = left(trim(put(_JOB_,$7.)));
length _REASON_ $7; drop _REASON_;
_REASON_ = left(trim(put(_REASON_,$7.)));
if missing(CLNO)
or missing(NINQ)
or missing(MORTDUE)
or missing(DEROG)
or missing(VALUE)
or missing(DELIQ)
or missing(LOAN)
or missing(YOJ)
or missing(DEBTINC)
or missing(CLAGE)
then do;
  _badval_ = 1;
  goto skip_0_0;
end;
```

```

do _i=1 to 19; _xrow_0_0_{_i_} = 0; end;

_xrow_0_0_[1] = 1;

_xrow_0_0_[2] = DELINQ;

_xrow_0_0_[3] = DEROG;

_xrow_0_0_[4] = NINQ;

_xrow_0_0_[5] = CLAGE;

_xrow_0_0_[6] = CLNO;

_xrow_0_0_[7] = DEBTINC;

_xrow_0_0_[8] = LOAN;

_xrow_0_0_[9] = MORTDUE;

_xrow_0_0_[10] = VALUE;

_xrow_0_0_[11] = YOJ;

_temp_ = 1;
select (_JOB_);
  when ('Mgr') _xrow_0_0_[12] = _temp_;
  when ('Office') _xrow_0_0_[13] = _temp_;
  when ('Other') _xrow_0_0_[14] = _temp_;
  when ('ProfExe') _xrow_0_0_[15] = _temp_;
  when ('Sales') _xrow_0_0_[16] = _temp_;
  when ('Self') _xrow_0_0_[17] = _temp_;
  otherwise do; _badval_ = 1; goto skip_0_0; end;
end;

_temp_ = 1;
select (_REASON_);
  when ('DebtCon') _xrow_0_0_[18] = _temp_;
  when ('HomeImp') _xrow_0_0_[19] = _temp_;
  otherwise do; _badval_ = 1; goto skip_0_0; end;
end;

do _i=1 to 19;
  _linp_ + _xrow_0_0_{_i_} * _beta_0_0_{_i_};
end;

skip_0_0:
if (_badval_ eq 0) and not missing(_linp_) then do;
  if (_linp_ > 0) then do;
    P_BAD0 = 1 / (1+exp(-_linp_));
  end; else do;
    P_BAD0 = exp(_linp_) / (1+exp(_linp_));
  end;
  P_BAD1 = 1 - P_BAD0;
end; else do;
  _linp_ = .;
  P_BAD0 = .;
  P_BAD1 = .;
end;

```

The LOGSELECT Procedure – Results

The LOGSELECT Procedure

Model Information	
Data Source	HMEQ_TRAIN
Response Variable	BAD
Distribution	Binary
Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging
Seed	78900

Number of Observations			
Description	Total	Training	Validation
Number of Observations Read	2381	1880	501
Number of Observations Used	1364	1077	287

Response Profile				
Ordered Value	BAD	Total Frequency	Training	Validation
1	0	1243	982	261
2	1	121	95	26

Probability modeled is BAD = 0.

Class Level Information		
Class	Levels	Values
JOB	6	Mgr Office Other ProfExe Sales Self
REASON	2	DebtCon HomeImp

Convergence criterion (GCONV=1E-8) satisfied.

Dimensions	
Columns in Design	19
Number of Effects	13
Max Effect Columns	6
Rank of Design	17
Parameters in Optimization	17

Testing Global Null Hypothesis: BETA=0			
Test	DF	Chi-Square	Pr > Chi Sq
Likelihood Ratio	16	149.8225	<.0001

Fit Statistics		
Description	Training	Validation
-2 Log Likelihood	492.87085	148.44390
AIC (smaller is better)	526.87085	182.44390
AICC (smaller is better)	527.44875	184.71899
SBC (smaller is better)	611.56374	244.65509
Average Square Error	0.06038	0.06646
-2 Log L (Intercept-only)	642.69336	174.44212
R-Square	0.12987	0.08660
Max-rescaled R-Square	0.28898	0.19015
McFadden's R-Square	0.23312	0.14904
Misclassification Rate	0.07242	0.08014
Difference of Means	0.23601	0.16968

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	4.841170	1.130415	18.3411	<.0001
DELINQ	1	-0.755273	0.117179	41.5440	<.0001
DEROG	1	-0.754898	0.178548	17.8782	<.0001
NINQ	1	-0.083783	0.065879	1.6174	0.2035
CLAGE	1	0.005515	0.001981	7.7543	0.0054
CLNO	1	0.027044	0.014963	3.2667	0.0707
DEBTINC	1	-0.103053	0.019637	27.5418	<.0001
LOAN	1	0.000012545	0.000014825	0.7161	0.3974
MORTDUE	1	-0.000000318	0.000005857	0.0030	0.9567
VALUE	1	-0.000001193	0.000004963	0.0577	0.8101
YOJ	1	-0.001220	0.017692	0.0048	0.9450
JOB Mgr	1	0.782167	0.775211	1.0180	0.3130
JOB Office	1	1.054619	0.777071	1.8419	0.1747
JOB Other	1	0.804005	0.730498	1.2114	0.2711
JOB ProfExe	1	0.272171	0.734760	0.1372	0.7111
JOB Sales	1	-0.759770	1.011778	0.5639	0.4527
JOB Self	0	0	.	.	.
REASON DebtCon	1	-0.455425	0.299378	2.3142	0.1282
REASON Homelmp	0	0	.	.	.

Task Timing		
Task	Seconds	Percent
Setup and Parsing	0.00	2.23%
Levelization	0.00	0.88%
Model Initialization	0.00	0.06%
SSCP Computation	0.00	0.70%
Model Fitting	0.03	18.69%
Creating Output Data	0.00	1.19%
Producing Score Code	0.10	74.25%
Cleanup	0.00	1.81%
Total	0.14	100.00%

Output CAS Tables			
CAS Library	Name	Number of Rows	Number of Columns
CASUSER(sas)	LOGSELECT_BADLOANS_OUTPUT	2381	14

Transfer a Model from SAS Viya to SAS 9.4

This example uses a remote submit operation with SAS/CONNECT to transfer the model score code and a subset of the training data from SAS Viya to SAS 9.4. The example code is run in a UNIX environment.

For more information, see [SAS/CONNECT for SAS Viya 3.1 User's Guide](#).

```
/* Set up macro variables. */
%let usr=yourUserID;
%let passwd=yourPassword;
```

```

%let conn94=sas94connect.server.com connectPortNumber;
%let model_name1=logselect_badloans;

/* Reference the library where the training_data_subset is stored. */
libname SASViya '~/SASViya';

/* Reference the location where the score code is stored in SAS Viya. */
filename input1 "~/SASViya/&model_name1..sas";

/* Initialize SAS/CONNECT. */
signon conn94 user=&usr pwd="&passwd";

/* Add needed macro variables to the remote session. */
%syslput model_name1=&model_name1;

/* Remotely submit code from SAS Viya to SAS 9.4. */
rsubmit;

/* Upload the training data set. */
libname SAS94lib '~/SAS94';
proc upload inlib=SASViya outlib=SAS94lib;
  select training_data_subset;
run;

/* Transfer the score code files. */
filename output1 "~/SAS94/&model_name1..sas";
proc upload infile=input1 outfile=output1;
run;

/* Sign out of the SAS 9.4 machine. */
endrsubmit;
signoff;

```

Registering Models in SAS Model Manager

You can use the %AA_MM_Register macro to register models to an existing project version in the SAS Model Manager 14.2 model repository, or you can import the models using the SAS Model Manager 14.2 web application. The %AA_MM_Register macro is available on SAS 9.4 where SAS Model Manager 14.2 is installed.

Run the %AA_Model_Register Macro

After you transfer the logselect_badloans.sas code file from SAS Viya to SAS 9.4 using the RSUBMIT statement, you can then create a SAS model package (SPK) file using the %AA_Model_Register macro. The macro creates a folder for the model location that is specified for the SPKFOLDER= option. The folder name is the UUID of the model. The name of the SPK file is miningResults.spk.

For more information, see [“%AA_Model_Register Autocall Macro” in SAS Model Manager: Macro Reference](#).

Here is an example of the %AA_Model_Register macro:

Note: You can run the %AA_Model_Register macro code on SAS Viya using the RSUBMIT statement, or you can run the code on SAS 9.4.

```

/* Set up macro variables. */
%let usr=yourUserID;
%let passwd=yourPassword;

```



```

%let conn94=sas94connect.server.com connectPortNumber;
%let meta94=sas94metadata.server.com;
%let metaport=sas94metadataServerPortNumber;
%let model_name1=logselect_badloans;

/* Initialize SAS/CONNECT. */
signon conn94 user=&usr pwd="&passwd";

/* Add needed macro variables to the remote session. */
%syslput usr=&usr;
%syslput passwd=&passwd;
%syslput meta94=&meta94;
%syslput metaport=&metaport;
%syslput model_name1=&model_name1;

/* Remotely submit code from SAS Viya to SAS 9.4. */
rsubmit;

/* Set up the metadata connection system options. */
options metaPort=&metaport
        metaServer=&meta94
        metaRepository=Foundation
        metaUser=&usr
        metaPass=&passwd;

/* Specify the libname. */
libname SAS94lib '~/SAS94';

/* Automatically call the SAS 9.4 macro for generating the model SPK file. */
%aamodel;

/* Create a logselect SAS model package (SPK) file, with input variables and output variables metadata. */
/* Do not register the model in the SAS Metadata Repository. */
%aa_model_register(
    modelname=&model_name1,
    modeldesc=%bquote(logselect_badloans model from CAS),
    data=SAS94lib.training_data_subset,
    target=BAD,
    level=binary,
    scorecodefile=~/SAS94/&model_name1..sas,
    scorecodeformat=DATASTEP,
    register=N,
    spk=Y,
    spkfolder=%str(~),
    miningfunction=classification,
    debug=N);

/* Get the location of the SPK file. */
%let spkdir1 = &AA_SPK_FOLDERID;

/* Note: Stay signed in to your SAS/CONNECT session on SAS 9.4 to use the SAS Model Manager macros
to register your model in SAS Model Manager. */

```

Use Macros to Register Models into SAS Model Manager

SAS Model Manager 14.2 provides macros that you can use in a SAS program to add folders, projects, and versions, and to set properties, as well as register models. These macros are available on SAS 9.4 where SAS Model Manager 14.2 is installed. For more information, see [“Adding Folders, Projects, Versions, and Properties Using Macros” in SAS Model Manager: Macro Reference](#) and [“%MM_Register Macro” in SAS Model Manager: Macro Reference](#).

Here is an example of using the macros to create a folder, project, and version, and then import the SAS model package file that you previously created.

Note: This example assumes that you are still signed in to the SAS/CONNECT session on SAS 9.4 from the previous example.

```

/* Initialize the SAS Model Manager 14.2 macros and variables. */
%let _MM_user=&usr;
%let _MM_Password=&passwd;

/* Specify the libname for the SAS Model Manager tutorial data sets. */
libname mmtut '~/SAS94';

/* Include the SAS Model Manager macros that are needed. */
filename aaa catalog "sashelp.modelmgr.accessmacros.source"; %include aaa;
filename aaa catalog 'sashelp.modelmgr.mdlmgr_addfolder.source'; %include aaa;
filename aaa catalog 'sashelp.modelmgr.mdlmgr_addproject.source'; %include aaa;
filename aaa catalog 'sashelp.modelmgr.mdlmgr_setproperty.source'; %include aaa;
filename aaa catalog 'sashelp.modelmgr.mdlmgr_addversion.source'; %include aaa;
filename aaa catalog 'sashelp.modelmgr.mdlmgr_ep_publish.source'; %include aaa;
filename aaa catalog 'sashelp.modelmgr.mdlmgr_hadoop_publish.source'; %include aaa; filename aaa;
%global topLvlFolder project1 version1;

/* Add a SAS Model Manager folder, project, and version to contain the models. */
%mdlmgr_AddFolder(parentId=//ModelManagerDefaultRepo/MMRoot,
                 Name=SASViya,
                 desc=%bquote(Only models imported from SAS Viya allowed.),
                 NewFolderId=topLvlFolder);
%mdlmgr_AddProject(parentId=&topLvlFolder.,
                  name=Badloans, desc=none,
                  modelFunction=classification,
                  inputVarTable=mmtut.hmeq_project_input,
                  outputVarTable=mmtut.hmeq_project_output,
                  newProjectId=project1);

%mdlmgr_AddVersion(parentId=&project1.,
                  newVersionId=version1);

/* Register the model into SAS Model Manager 14.2. */
filename mdlspk1 "&spkdir1./miningResult.spk";

%MM_Register(
  VersionId=&version1.,
  EMModelPackage=mdlspk1,
  ModelName=&model_name1,
  Description=none
);

```

```

/* Clean up files and data. */
filename score1 "~/SAS94/&model_name1..sas";
data _NULL_;
  rc = fdelete("score1");
  rc = fdelete("mdlspk1");
  rc = filename("subdir1", "&spkdir1");
  rc = fdelete("subdir1");

/* Sign out of the SAS 9.4 machine. */
endrsubmit;
signoff;

```

View SAS Viya Models in SAS Model Manager 14.2

You can now sign in to the SAS Model Manager 14.2 web application to view the SAS Viya models that were registered. The retraining of DATA step score code models that have been created with SAS Visual Data Mining and Machine Learning procedures is not currently supported in SAS Model Manager 14.2. For more information about creating a project, importing models, and what tasks can be performed, see [SAS Model Manager: User's Guide](#).

SAS Viya Folder with Badloans Project

Name	Model Function	Type	Location	Date Modified	Created By	Operation Status
▶ Credit				Sep 29, 2016 04:08 PM	sasdemo	
▶ HMEQ				Oct 6, 2016 12:51 PM	sasdemo	
▼ SASViya				Oct 6, 2016 12:56 PM	sasdemo	
▶ Badloans	Classification	Project	SASViya	Oct 6, 2016 12:56 PM	sasdemo	Under Development
▶ SASViya2				Oct 4, 2016 03:55 PM	sasdemo	
▶ Tutorials				Oct 6, 2016 12:52 PM	sasdemo	

List of Registered Models

Name	Role	Version	Description	Model Function	Date Published	Date Modified	Created By
logselect_badloans		1.0	none	Classification		Oct 6, 2016 12:56 PM	sasdemo

