Paper SAS0747-2017

Open Your Mind: Use Cases for SAS® and Open-Source Analytics

Tuba Islam, SAS Institute Inc.

ABSTRACT

Data scientists need analytical tools and algorithms, whether commercial or open source, and will always have some favourites. But how do you decide when to use what? And how can you integrate their use to your maximum advantage? This paper provides best practices for deploying both SAS® and open-source analytical tools to increase productivity and efficiency in your enterprise ecosystem. Further, an example is provided of an analytical business flow for marketing using SAS and R algorithms in SAS® Enterprise Miner™ for developing a predictive model, and then operationalizing and automating that model for scoring, performance monitoring and retraining. There are also suggestions for using Python and SAS integration in a Jupyter Notebook environment. Seeing these examples will help you decide how to improve your analytics with similar integration of SAS and open source.

INTRODUCTION

This paper provides you with some examples to show the goodness of using open source and SAS in an enterprise ecosystem. Working in an open environment with various tools brings some challenges. It sometimes becomes more difficult to execute models in production quickly without extensive recoding. It can also become more challenging to automate the complete business flow for model management and to enable governance and lineage for analytics. SAS provides the analytical platform that connects the dots in the model lifecycle and helps "productionize" open-source models.

Some of the main benefits of SAS and open source working together:

- Freedom to use your language of choice as a data scientist, work collaboratively, and be able to exchange the resources and the analytical assets within the organization.
- Operationalize and automate the execution of your open source and SAS models in production quickly and increase the value gained from analytics.
- A transparent and easy-to-monitor analytical process and the governance and lineage for your models.

You will find an example of a campaign propensity use case in SAS® 9.4 using SAS® Enterprise Miner™ and SAS® Decision Manager for operationalizing open source R models and also the use of Jupyter notebook for accessing these models via REST and running post analysis in Python.

PREREQUISITES

For R integration on SAS 9.4, you will need to install R and the required packages on the server where SAS Enterprise Miner resides and set the required environment parameters such as R_HOME and RLANG. The details for the installation and configuration can be found on SAS Communities portal when you search for R integration.

For Python integration on SAS 9.4, you will need to install Python and SASPy package to execute SAS code in Jupyter Notebook. The package and the instructions can be found under SASPy repository on GitHub. For Python integration on SAS® Viya™, you would install the package in python-swat repository on GitHub. This paper will only include examples for SAS 9.4.

STEP-BY-STEP ANALYTICS WITH SAS AND OPEN SOURCE

You will find the analytical steps of a marketing use case that builds a campaign propensity model for buying an organic product using SAS and open-source analytics. Open Source Integration node in SAS Enterprise Miner will be used for creating R models and making them production-ready. The use case will include one-click registration of models into SAS Decision Manager with the unique model ID that will enable the business users to track the performance of campaign models in production.

The marketing data includes customer demographics and historical purchase information about different products. The purchase event is flagged as 1/0 and stored in the variable called “Target”. The sample dataset and the code snippets shared in the paper are available on GitHub under enlighten-integration repository.

STEP 1. BUILD PRODUCTION MODELS IN SAS® ENTERPRISE MINER™

The Open Source Integration node enables the execution of R code within SAS Enterprise Miner. This node transfers the data, the metadata, and the results automatically between two systems. It facilitates multitasking in R by running models in parallel. The Open Source Integration node enables certain R models to be translated into SAS DATA step code and deployed in production.

The default setting for the output mode in the Open Source Integration node is “PMML”, which is an open standard. The R models that are tested and supported for PMML mode are “lm”, “rpart”, “glm”, “multinom”, “nnet” and “kmeans”. You can type in any R code in the code editor of Open Source Integration node when you set the output mode to “Merge” or “None” in the parameter panel. In this paper, the PMML mode is used as it provides a strong integration for model management and automation.

There are some built-in macros in Open Source Integration node such as &EMR_MODEL, &EMR_CLASS_TARGET, &EMR_NUM_INPUT that make the coding in R easier for the data scientists. Instead of hardcoding the variable names, the metadata for the input variables can be managed by SAS nodes, which makes the modeling process more transparent and easier to follow.

A simple example for calling R Regression model (glm) in Open Source Integration node:

```
# Regression model
library(glm2)
library(pmml)
&EMR_MODEL <- glm(&EMR_CLASS_TARGET ~ &EMR_CLASS_INPUT + &EMR_NUM_INPUT,
data= &EMR_IMPORT_DATA, family= binomial())
```

A simple example for calling R Decision Tree model (rpart) in Open Source Integration node:

```
# Decision Tree model
library(rpart)
library(pmml)
&EMR_MODEL <- rpart(&EMR_CLASS_TARGET ~ &EMR_CLASS_INPUT + &EMR_NUM_INPUT,
data= &EMR_IMPORT_DATA, method = "class")
```

You can create Ensemble models by joining SAS and R models in SAS Enterprise Miner while searching for better accuracy. In the example below, SAS Decision Tree and R Regression models are joined in the Ensemble node:

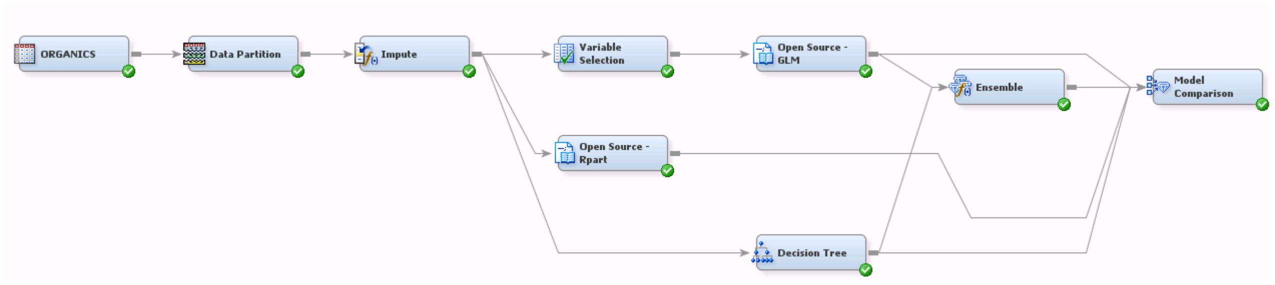


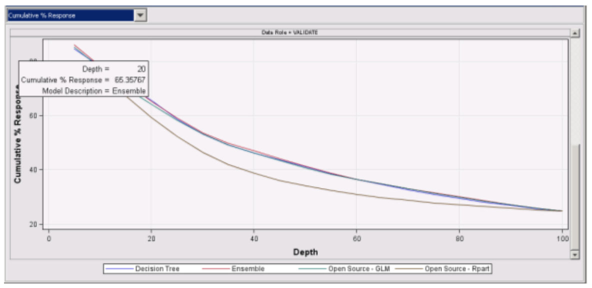
Figure 1. Developing R and SAS Models in SAS® Enterprise Miner™

As a result of the model build process with PMML, a DATA step score code is generated automatically, which is ready to be deployed in production.

A snippet of the generated code in the Open Source Integration node:

```
/*********************************************/;
* PSCORE TIMESTAMP: 2017-23-1 19:3:27.43 ;
* SAS VERSION: 9.04.01M4P110916 ;
* SAS HOSTNAME: sasva ;
* SAS ENCODING: wlatin1 ;
* SAS USER: sasdemo ;
* SAS LOCALE: EN_US ;
* PMML Path: D:\opt\sasinside\SASWORK\_TD30744_SASVA_\emRPMML.xml ;
* PMML SOURCE: Rattle/PMML;
* PMML SOURCE VERSION: 1.4;
* PMML TIMESTAMP: 2017-02-23 19:03:27 ;
* MODEL TYPE: GeneralRegressionModel ;
* MODEL FUNCTION NAME: Classification ;
/*********************************************/;

if missing("DemGender"n) then do;
    PSCR_WARN = 1;
end;
else do;
    "PSCR_AP0"n = "DemGender"n;
    if ( "PSCR_AP0"n not in ( 'F', 'M', 'U'  )  ) then do;

        PSCR_WARN = 1;
    end;
end;
...
```

The final step is the comparison of all models and selection of the champion to be deployed in production:



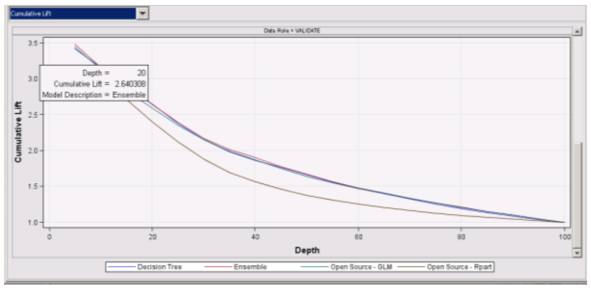


Figure 2. The Comparison of R and SAS Models in SAS® Enterprise Miner™

One of the reports generated by Model Comparison node is the Cumulative Captured Response chart shown on the left. It gives you the response rate for each decile ordered by the probability score. If you want to target top 20% of your customer base, you can explore the response rate for that percentile, which is around 65.4% in this example. The Lift chart on the right shows how better this response is compared to a random selection.

The results for the Ensemble model and SAS Decision Tree model are very close. You might set the Ensemble model as the champion to see how the automation will be handled for the combination of R and SAS model. And the SAS Decision Tree model can be set as the challenger in SAS Decision Manager.

The modeling process is now completed. In the next step, you will execute the key components in SAS Enterprise Miner to enable the lineage and automation for R and SAS models.

STEP 2. ONE-CLICK AUTOMATION AND LINEAGE

SAS automatically generates and stores a universally unique identifier (modelUUID) for all models including the R models. Using this Model ID, when you want to monitor the performance of your campaign and evaluate the impact of the analytical model on the response rate, you will be able to trace it back even in a very complex and large-scale marketing environment where you execute hundreds of models with different versions. This feature would also be beneficial for regulated industries like banking where you need to explain the reasons for your decisions.

You can export your model into SAS Decision Manager with one-click using the built-in SAS macros for model development and monitoring.

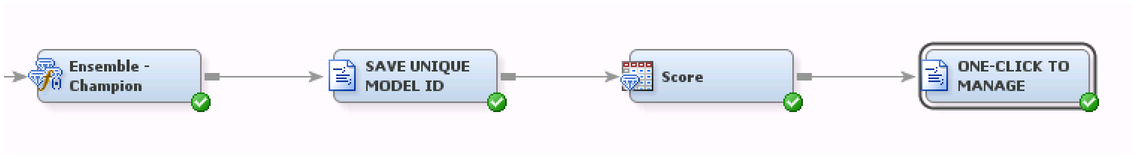
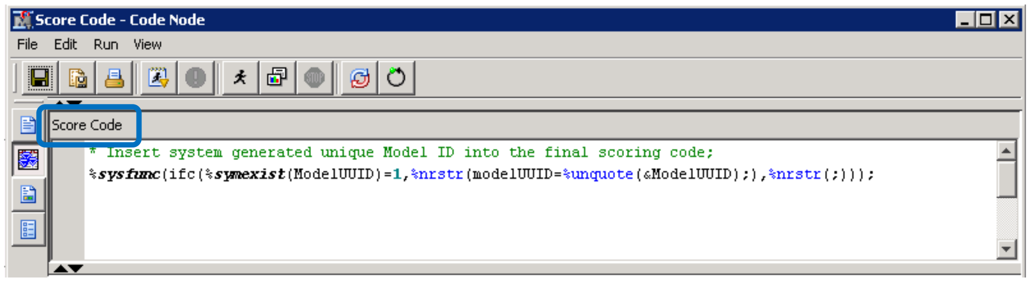


Figure 3. The Simple Flow Providing Model Lineage and Automation in SAS Enterprise Miner

The unique model ID is automatically extracted from SAS Metadata when you execute your score code in SAS.

If you run in-database scoring, you can create the model ID in your output table by including the following code in the *scoring editor* of the SAS Code node that is connected to the champion model:

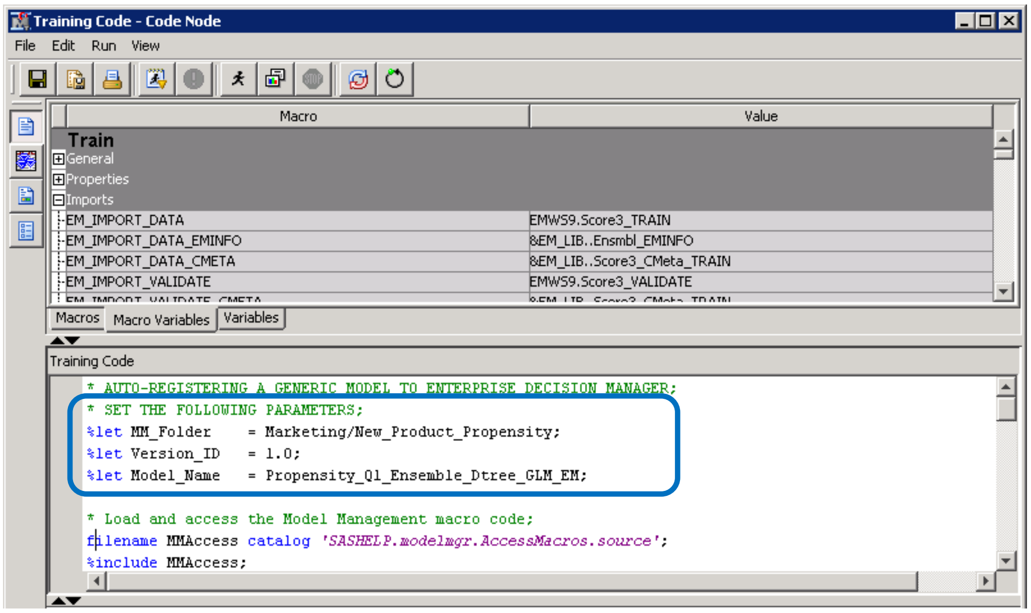


```
* Insert system generated unique Model ID into the final scoring code;
%sysfunc(ifc(%symexist(ModelUUID)=1,%nrstr(modelUUID=%unquote(&ModelUUID);),%nrstr(;)));
```

Figure 4. The SAS Code to Save the Unique Model ID for In-Database Scoring

The final node in the diagram above (One-Click to Manage) exports and registers your model into SAS Decision Manager.

The three parameters that will be used by %MM_Register macro are assigned at the beginning of the code in SAS Code node:

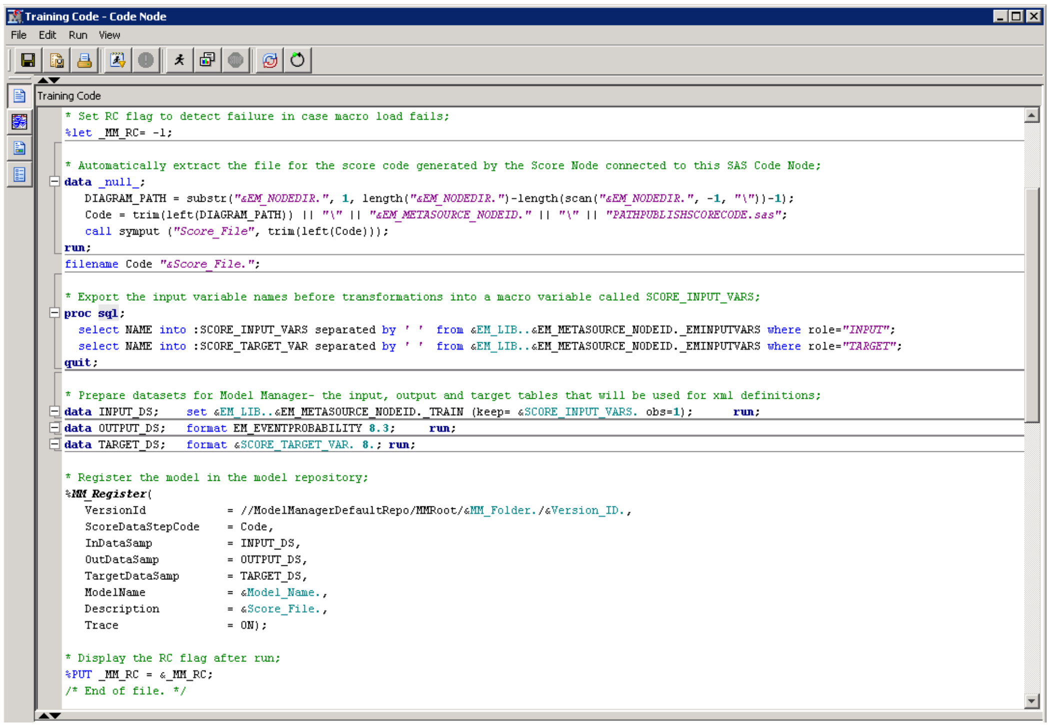


Macro	Value
EM_IMPORT_DATA	EMWS9.Score3_TRAIN
EM_IMPORT_DATA_EMINFO	&EM_LIB..Ensmbl_EMINFO
EM_IMPORT_DATA_CMETA	&EM_LIB..Score3_CMeta_TRAIN
EM_IMPORT_VALIDATE	EMWS9.Score3_VALIDATE

```
* AUTO-REGISTERING A GENERIC MODEL TO ENTERPRISE DECISION MANAGER;
* SET THE FOLLOWING PARAMETERS;
%let MM_Folder     = Marketing/New_Product_Propensity;
%let Version_ID    = 1.0;
%let Model_Name    = Propensity_Q1_Ensemble_Dtree_GLM_EM;

* Load and access the Model Management macro code;
filename MMAccess catalog 'SASHELP.modelmgr.AccessMacros.source';
%include MMAccess;
```

Figure 5. The One-Click Export SAS Code (Part-1) for Assigning the Parameters

The last part of the code in SAS Code node is a generic code that is parametric and can be used for exporting any model directly from SAS Enterprise Miner into SAS Decision Manager:



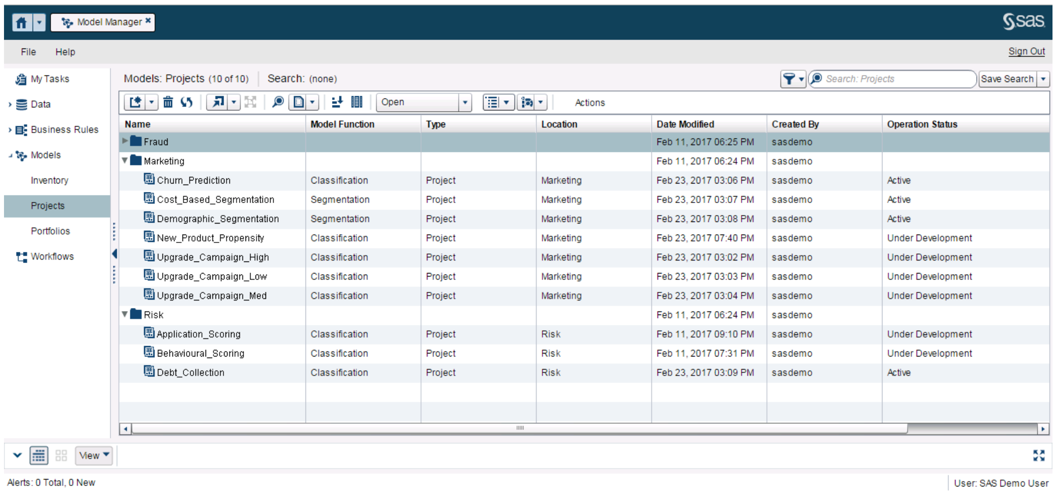
```
* Set RC flag to detect failure in case macro load fails;
%let _MM_RC= -1;

* Automatically extract the file for the score code generated by the Score Node connected to this SAS Code Node;
data _null_;
   DIAGRAM_PATH = substr("&EM_NODEDIR.", 1, length("&EM_NODEDIR.")-length(scan("&EM_NODEDIR.", -1, "\"))-1);
   Code = trim(left(DIAGRAM_PATH)) || "\" || "&EM_METASOURCE_NODEID." || "\" || "PATHPUBLISHSCORECODE.sas";
   call symput ("Score_File", trim(left(Code)));
run;
filename Code "&Score_File.";

* Export the input variable names before transformations into a macro variable called SCORE_INPUT_VARS;
proc sql;
  select NAME into :SCORE_INPUT_VARS separated by ' '  from &EM_LIB..&EM_METASOURCE_NODEID._EMINPUTVARS where role="INPUT";
  select NAME into :SCORE_TARGET_VAR separated by ' '  from &EM_LIB..&EM_METASOURCE_NODEID._EMINPUTVARS where role="TARGET";
quit;

* Prepare datasets for Model Manager- the input, output and target tables that will be used for xml definitions;
data INPUT_DS;    set &EM_LIB..&EM_METASOURCE_NODEID._TRAIN (keep= &SCORE_INPUT_VARS. obs=1);      run;
data OUTPUT_DS;   format EM_EVENTPROBABILITY 8.3;      run;
data TARGET_DS;   format &SCORE_TARGET_VAR. 8.; run;

* Register the model in the model repository;
%MM_Register(
   VersionId              = //ModelManagerDefaultRepo/MMRoot/&MM_Folder./&Version_ID.,
   ScoreDataStepCode      = Code,
   InDataSamp             = INPUT_DS,
   OutDataSamp            = OUTPUT_DS,
   TargetDataSamp         = TARGET_DS,
   ModelName              = &Model_Name.,
   Description            = &Score_File.,
   Trace                  = ON);

* Display the RC flag after run;
%PUT _MM_RC = &_MM_RC;
/* End of file. */
```

Figure 6. The One-Click Export SAS Code (Part 2) for Exporting and Registering

This champion model is now stored in SAS Decision Manager, ready to be published and monitored.

STEP 3. MONITOR PERFORMANCE OF OPEN SOURCE MODELS

SAS Decision Manager is the centralized management environment for all the models that could be developed within different departments of your organization.

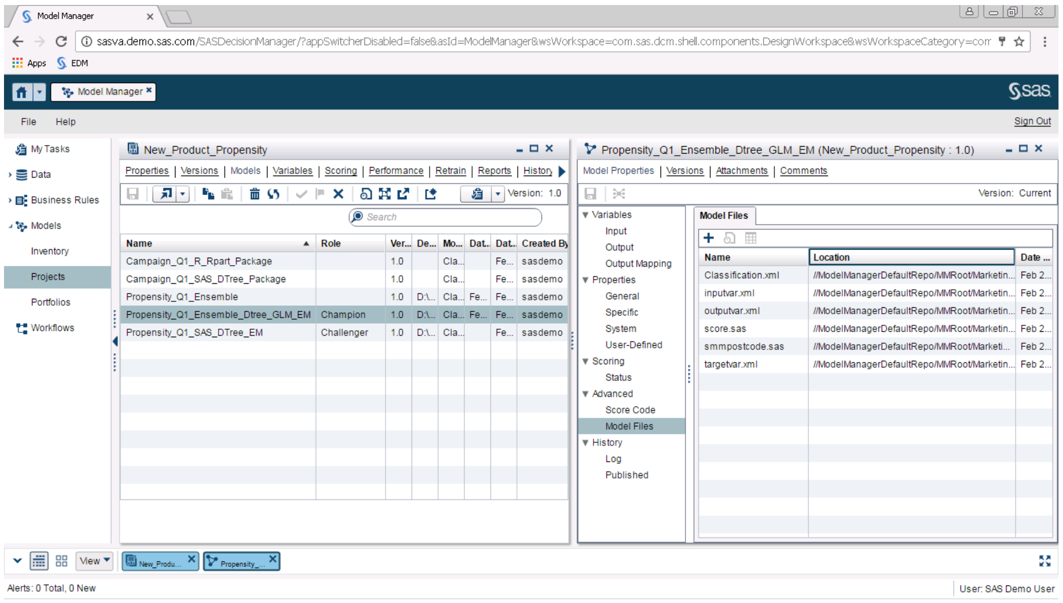


Models: Projects (10 of 10) | Search: (none)

Name	Model Function	Type	Location	Date Modified	Created By	Operation Status
Fraud				Feb 11, 2017 06:25 PM	sasdemo	
Marketing				Feb 11, 2017 06:24 PM	sasdemo	
Churn_Prediction	Classification	Project	Marketing	Feb 23, 2017 03:06 PM	sasdemo	Active
Cost_Based_Segmentation	Segmentation	Project	Marketing	Feb 23, 2017 03:07 PM	sasdemo	Active
Demographic_Segmentation	Segmentation	Project	Marketing	Feb 23, 2017 03:08 PM	sasdemo	Active
New_Product_Propensity	Classification	Project	Marketing	Feb 23, 2017 07:40 PM	sasdemo	Under Development
Upgrade_Campaign_High	Classification	Project	Marketing	Feb 23, 2017 03:02 PM	sasdemo	Under Development
Upgrade_Campaign_Low	Classification	Project	Marketing	Feb 23, 2017 03:03 PM	sasdemo	Under Development
Upgrade_Campaign_Med	Classification	Project	Marketing	Feb 23, 2017 03:04 PM	sasdemo	Under Development
Risk				Feb 11, 2017 06:24 PM	sasdemo	
Application_Scoring	Classification	Project	Risk	Feb 11, 2017 09:10 PM	sasdemo	Under Development
Behavioural_Scoring	Classification	Project	Risk	Feb 11, 2017 07:31 PM	sasdemo	Under Development
Debt_Collection	Classification	Project	Risk	Feb 23, 2017 03:09 PM	sasdemo	Active

Figure 7. The Enterprise-Level Management of SAS and R Models in SAS

You can view the models that you exported to your &MM_Folder, which is set as a parameter in the SAS Code node in SAS Enterprise Miner. You can also import models from SPK files (SAS package) which enables the retraining functionality of R models in SAS Decision Manager:



New_Product_Propensity

Name	Role	Ver...	De...	Mo...	Dat...	Dat...	Created By
Campaign_Q1_R_Rpart_Package		1.0		Cla...		Fe...	sasdemo
Campaign_Q1_SAS_DTree_Package		1.0		Cla...		Fe...	sasdemo
Propensity_Q1_Ensemble		1.0	D:\...	Cla...	Fe...	Fe...	sasdemo
Propensity_Q1_Ensemble_Dtree_GLM_EM	Champion	1.0	D:\...	Cla...	Fe...	Fe...	sasdemo
Propensity_Q1_SAS_DTree_EM	Challenger	1.0	D:\...	Cla...		Fe...	sasdemo

Propensity_Q1_Ensemble_Dtree_GLM_EM (New_Product_Propensity : 1.0)

Model Files

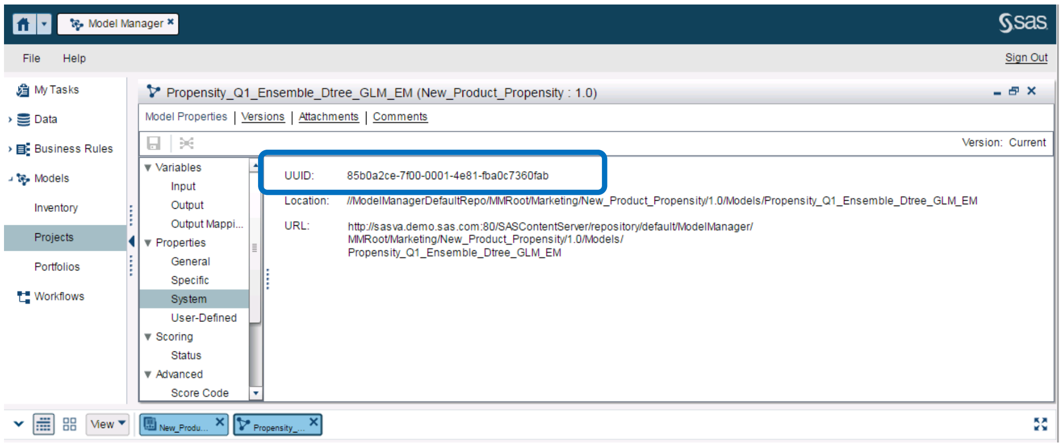
Name	Location	Date ...
Classification.xml	//ModelManagerDefaultRepo/MMRoot/Marketin...	Feb 2...
inputvar.xml	//ModelManagerDefaultRepo/MMRoot/Marketin...	Feb 2...
outputvar.xml	//ModelManagerDefaultRepo/MMRoot/Marketin...	Feb 2...
score.sas	//ModelManagerDefaultRepo/MMRoot/Marketin...	Feb 2...
smmpostcode.sas	//ModelManagerDefaultRepo/MMRoot/Marketi...	Feb 2...
targetvar.xml	//ModelManagerDefaultRepo/MMRoot/Marketin...	Feb 2...

Alerts: 0 Total, 0 New

User: SAS Demo User

Figure 8. Auto-Registered R and SAS Models for New Product Propensity Analysis

You find the unique model ID (UUID) in the System tab of the model in SAS Decision Manager:



Propensity_Q1_Ensemble_Dtree_GLM_EM (New_Product_Propensity : 1.0)

Version: Current

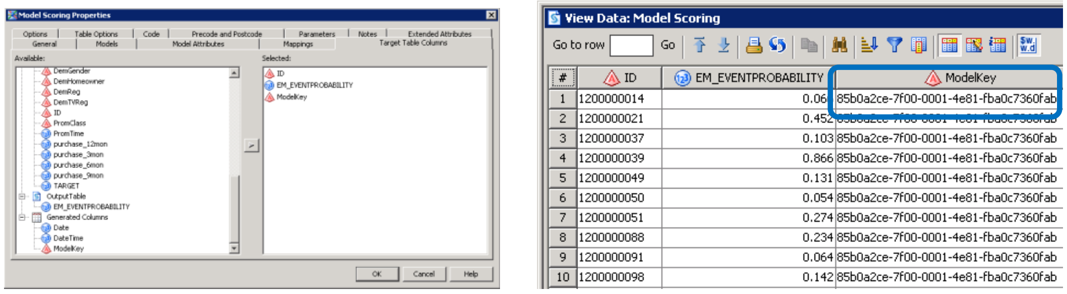
UUID: 85b0a2ce-7f00-0001-4e81-fba0c7360fab

Location: //ModelManagerDefaultRepo/MMRoot/Marketing/New_Product_Propensity/1.0/Models/Propensity_Q1_Ensemble_Dtree_GLM_EM

URL: http://sasva.demo.sas.com:80/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/New_Product_Propensity/1.0/Models/Propensity_Q1_Ensemble_Dtree_GLM_EM

Figure 9. Unique Model ID Automatically Generated for the Ensemble Model

The same ID is generated in the output table when you execute the score code:



Model Scoring Properties — Target Table Columns

Available: DemGender, DemHomeowner, DemReg, DemTVReg, ID, PromClass, PromTime, purchase_12mon, purchase_3mon, purchase_6mon, purchase_9mon, TARGET; OutputTable: EM_EVENTPROBABILITY; Generated Columns: Date, DateTime, ModelKey

Selected: ID, EM_EVENTPROBABILITY, ModelKey

View Data: Model Scoring

#	ID	EM_EVENTPROBABILITY	ModelKey
1	1200000014	0.06	85b0a2ce-7f00-0001-4e81-fba0c7360fab
2	1200000021	0.452	85b0a2ce-7f00-0001-4e81-fba0c7360fab
3	1200000037	0.103	85b0a2ce-7f00-0001-4e81-fba0c7360fab
4	1200000039	0.866	85b0a2ce-7f00-0001-4e81-fba0c7360fab
5	1200000049	0.131	85b0a2ce-7f00-0001-4e81-fba0c7360fab
6	1200000050	0.054	85b0a2ce-7f00-0001-4e81-fba0c7360fab
7	1200000051	0.274	85b0a2ce-7f00-0001-4e81-fba0c7360fab
8	1200000088	0.234	85b0a2ce-7f00-0001-4e81-fba0c7360fab
9	1200000091	0.064	85b0a2ce-7f00-0001-4e81-fba0c7360fab
10	1200000098	0.142	85b0a2ce-7f00-0001-4e81-fba0c7360fab

Figure 10. Unique Model ID Saved in the Score Dataset in SAS® Data Integration Studio

You can now build the monitoring jobs and start tracking the performance of R and SAS models. You can either use the GUI or MM macros to accomplish this task.

Monitoring the deviations in input characteristics and performance changes in time:

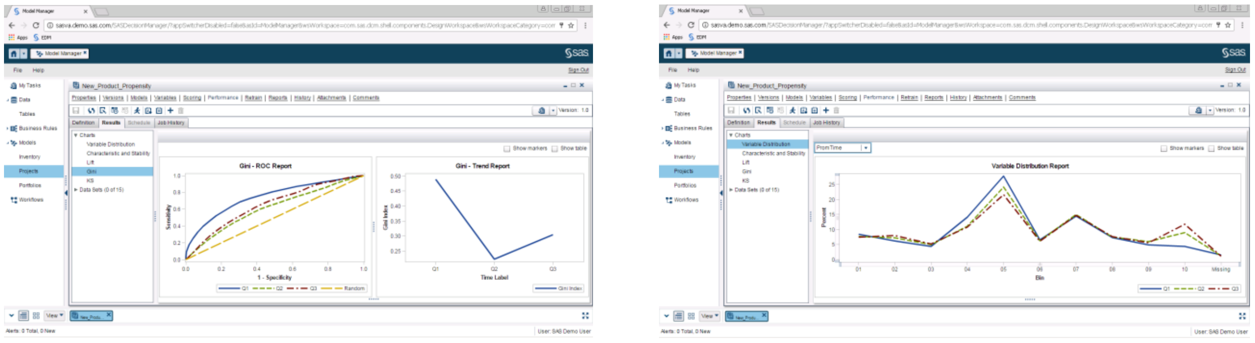


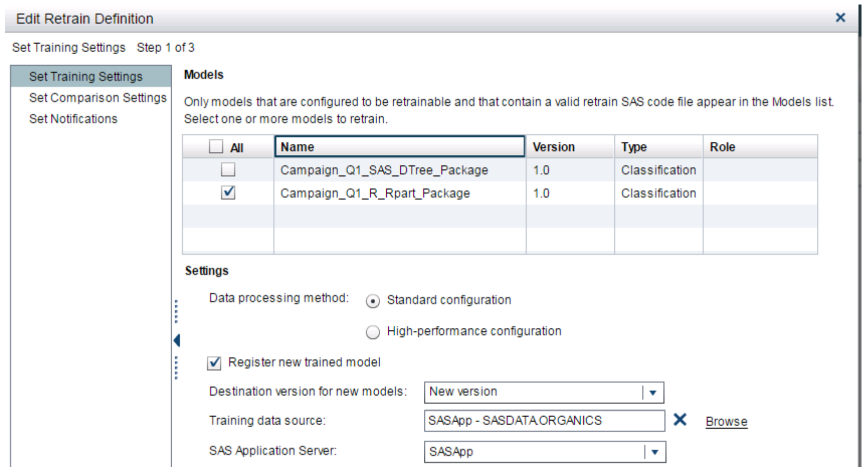
Figure 11. Model Performance Reports Generated for SAS® and R Models

There is a degradation in the performance of the Ensemble model when you look at the difference in Gini between the Q1 and Q3. SAS Decision Manager creates alerts when the models degenerate and sends notifications to pre-assigned users so that they can execute the retraining process or retire the model. The retraining can also be triggered automatically.

The next step will show the retraining of R models in SAS.

STEP 4. RETRAIN OPEN SOURCE MODELS

After importing the R model from SPK file, you define the retraining job in SAS Decision Manager and decide if the retrained model will be added to the existing version or to a new version.



Edit Retrain Definition

Set Training Settings Step 1 of 3

Set Training Settings
Set Comparison Settings
Set Notifications

Models

Only models that are configured to be retrainable and that contain a valid retrain SAS code file appear in the Models list. Select one or more models to retrain.

All	Name	Version	Type	Role
☐	Campaign_Q1_SAS_DTree_Package	1.0	Classification	
✓	Campaign_Q1_R_Rpart_Package	1.0	Classification	

Settings

Data processing method: [✓ Standard configuration] High-performance configuration

✓ Register new trained model

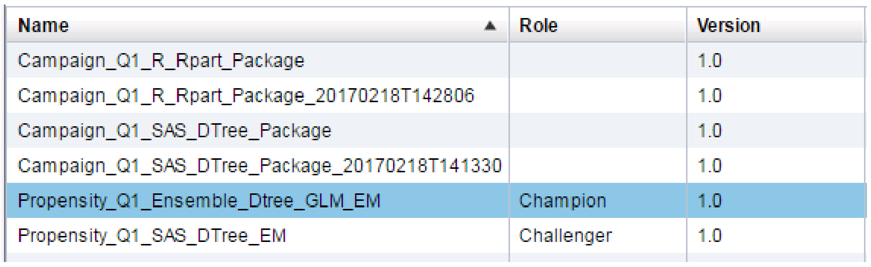
Destination version for new models: New version

Training data source: SASApp - SASDATA.ORGANICS Browse

SAS Application Server: SASApp

Figure 12. Set up a Retrain Job for R Rpart Model

After the execution, the final list of SAS and R models in SAS Decision Manager are updated to include the retrained models which are automatically renamed with the time stamp:



Name	Role	Version
Campaign_Q1_R_Rpart_Package		1.0
Campaign_Q1_R_Rpart_Package_20170218T142806		1.0
Campaign_Q1_SAS_DTree_Package		1.0
Campaign_Q1_SAS_DTree_Package_20170218T141330		1.0
Propensity_Q1_Ensemble_Dtree_GLM_EM	Champion	1.0
Propensity_Q1_SAS_DTree_EM	Challenger	1.0

Figure 13. The List in SAS Decision Manager Including Retrained R and SAS Models

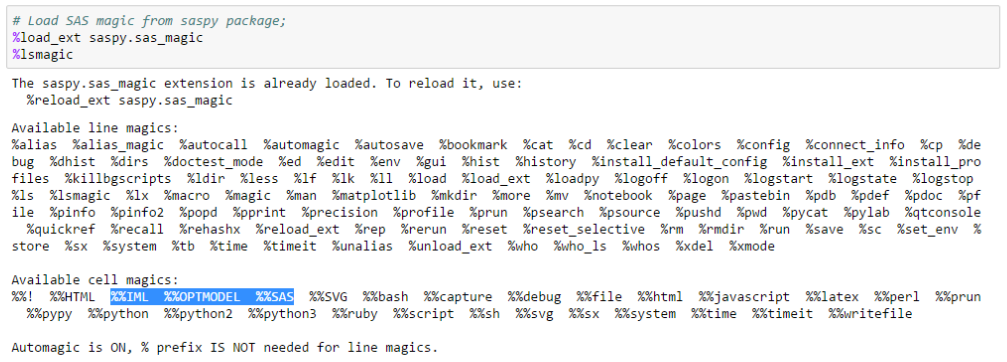
You can now compare the existing and retrained model and publish the one that performs better into production (in SAS or in-database) using the publish task in SAS Decision Manager or run the model in real-time.

In the next step, an example for an ad hoc analysis on model scores for Python developers is given.

STEP 5. AD HOC ANALYSIS OF PRODUCTION MODELS FROM PYTHON

You can download the module called SASPy from GitHub to connect to SAS 9.4 from Python. SASPy is a package that provides Python APIs to SAS and opens up all the data manipulation and analytical capabilities of your SAS System to the Python interface. It also supports the distributed computing. You can start a session and run SAS analytics from Python through the object-oriented methods. You can also use Jupyter magics which are available with the SASPy package and enable you to submit your SAS code to your SAS session from Python kernel. SASPy allows you to work in collaboration and utilize both SAS and Python skills for analytics.

In this example, you will see the usage of “sas_magic” to execute SAS code and Python together in Jupyter notebook.

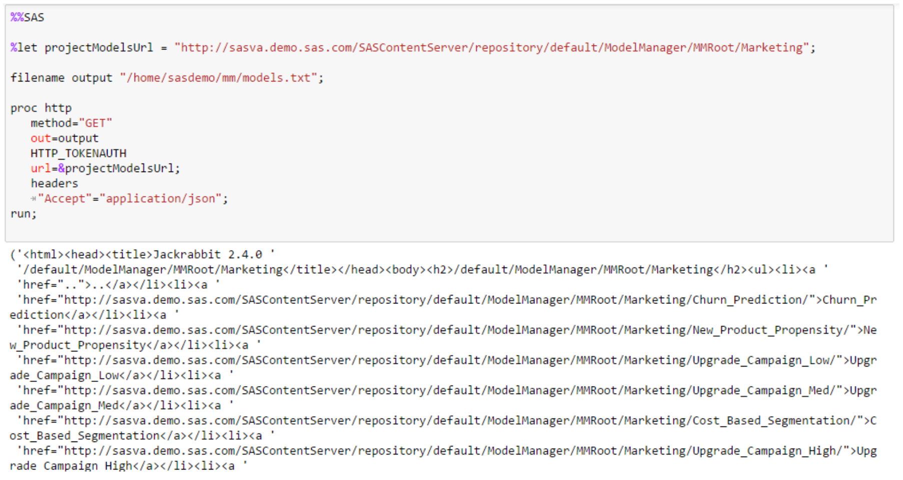


```
# Load SAS magic from saspy package;
%load_ext saspy.sas_magic
%lsmagic
```

```
The saspy.sas_magic extension is already loaded. To reload it, use:
  %reload_ext saspy.sas_magic

Available line magics:
%alias  %alias_magic  %autocall  %automagic  %autosave  %bookmark  %cat  %cd  %clear  %colors  %config  %connect_info  %cp  %de
bug  %dhist  %dirs  %doctest_mode  %ed  %edit  %env  %gui  %hist  %history  %install_default_config  %install_ext  %install_pro
files  %killbgscripts  %ldir  %less  %lf  %lk  %ll  %load  %load_ext  %loadpy  %logoff  %logon  %logstart  %logstate  %logstop
%ls  %lsmagic  %lx  %macro  %magic  %man  %matplotlib  %mkdir  %more  %mv  %notebook  %page  %pastebin  %pdb  %pdef  %pdoc  %pf
ile  %pinfo  %pinfo2  %popd  %pprint  %precision  %profile  %prun  %psearch  %psource  %pushd  %pwd  %pycat  %pylab  %qtconsole
  %quickref  %recall  %rehashx  %reload_ext  %rep  %rerun  %reset  %reset_selective  %rm  %rmdir  %run  %save  %sc  %set_env  %
store  %sx  %system  %tb  %time  %timeit  %unalias  %unload_ext  %who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%IML  %%OPTMODEL  %%SAS  %%SVG  %%bash  %%capture  %%debug  %%file  %%html  %%javascript  %%latex  %%perl  %%prun
  %%pypy  %%python  %%python2  %%python3  %%ruby  %%script  %%sh  %%svg  %%sx  %%system  %%time  %%timeit  %%writefile

Automagic is ON, % prefix IS NOT needed for line magics.
```

Figure 15. Import SAS Package for Calling SAS® Code in Python from Jupyter Notebook

When you include %%SAS magic at the top of the cell in your Jupyter notebook, you can execute any SAS code within that cell. In the example below, you extract the list of all projects in the repository under the Marketing folder, which also includes “New_Product_Propensity” project.



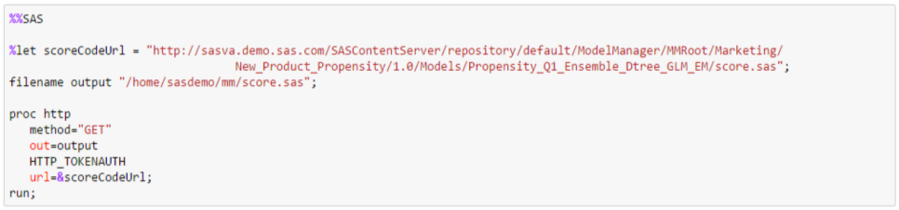
```
%%SAS

%let projectModelsUrl = "http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing";

filename output "/home/sasdemo/mm/models.txt";

proc http
   method="GET"
   out=output
   HTTP_TOKENAUTH
   url=&projectModelsUrl;
   headers
   "Accept"="application/json";
run;
```

```
('<html><head><title>Jackrabbit 2.4.0 '
 '/default/ModelManager/MMRoot/Marketing</title></head><body><h2>/default/ModelManager/MMRoot/Marketing</h2><ul><li><a '
 'href="..">..</a></li><li><a '
 'href="http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/Churn_Prediction/">Churn_Pr
ediction</a></li><li><a '
 'href="http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/New_Product_Propensity/">Ne
w_Product_Propensity</a></li><li><a '
 'href="http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/Upgrade_Campaign_Low/">Upgr
ade_Campaign_Low</a></li><li><a '
 'href="http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/Upgrade_Campaign_Med/">Upgr
ade_Campaign_Med</a></li><li><a '
 'href="http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/Cost_Based_Segmentation/">C
ost_Based_Segmentation</a></li><li><a '
 'href="http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/Upgrade_Campaign_High/">Upg
rade Campaign High</a></li><li><a '
```

Figure 16. The REST Call to Extract List of Analytical Projects in Marketing Repository

When you find the name of your project on that list, you can rerun the same code by updating the path in the “projectsModelsURL” until you extract the name of the full path for the score code.

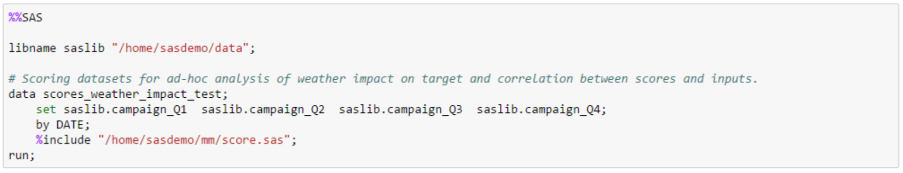
You then import the score code on your local drive via the following REST call:



```
%%SAS

%let scoreCodeUrl = "http://sasva.demo.sas.com/SASContentServer/repository/default/ModelManager/MMRoot/Marketing/
                                 New_Product_Propensity/1.0/Models/Propensity_Q1_Ensemble_Dtree_GLM_EM/score.sas";
filename output "/home/sasdemo/mm/score.sas";

proc http
   method="GET"
   out=output
   HTTP_TOKENAUTH
   url=&scoreCodeUrl;
run;
```

Figure 17. The REST Call to Extract the Score Code for a Specific Model

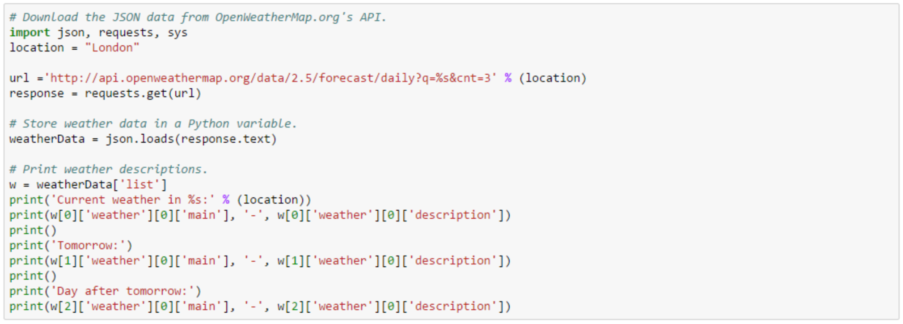
In the next step, you run an ad hoc analysis on your historical campaign data sets. You join data from different time periods and score all of them to run a seasonality analysis:



```
%%SAS

libname saslib "/home/sasdemo/data";

# Scoring datasets for ad-hoc analysis of weather impact on target and correlation between scores and inputs.
data scores_weather_impact_test;
    set saslib.campaign_Q1  saslib.campaign_Q2  saslib.campaign_Q3  saslib.campaign_Q4;
    by DATE;
    %include "/home/sasdemo/mm/score.sas";
run;
```

Figure 18. The SAS Code to Extract List of Analytical Projects in Marketing Repository

Using Python, you can experiment new data from web or other external sources to search for new correlations with campaign responses. As an example, you can extract the weather information from web and analyze the temperature impact on your target:



```
# Download the JSON data from OpenWeatherMap.org's API.
import json, requests, sys
location = "London"

url ='http://api.openweathermap.org/data/2.5/forecast/daily?q=%s&cnt=3' % (location)
response = requests.get(url)

# Store weather data in a Python variable.
weatherData = json.loads(response.text)

# Print weather descriptions.
w = weatherData['list']
print('Current weather in %s:' % (location))
print(w[0]['weather'][0]['main'], '-', w[0]['weather'][0]['description'])
print()
print('Tomorrow:')
print(w[1]['weather'][0]['main'], '-', w[1]['weather'][0]['description'])
print()
print('Day after tomorrow:')
print(w[2]['weather'][0]['main'], '-', w[2]['weather'][0]['description'])
```

Figure 19. The Python Code to Extract Weather Data

You might run further analysis to investigate your temperature-sensitive products, such as record players (might be preferred more in cold weather companied by wine) or bicycles (less tempting in rainy seasons) or dehumidifiers (more demand in summer when the laundry doesn't dry indoors) and so on.

As a Python coder, you can extract this information and run quick tests. If you decide that some of the new variables will add value to your production model, you can then automate the extraction process. The utilization of different tools and skills in the organization will bring flexibility that promotes collaboration and innovation which improves your analytical efficiency.

CONCLUSION

This paper provides examples to demonstrate the integration of open source analytics and SAS 9.4 using a marketing use case. SAS serves as the enterprise platform that enables and integrates different tools and skillsets within the organization including open source, provides a collaborative working environment and automates the analytical life cycle which would ultimately increase the value you get from analytics.

REFERENCES

SAS Communities Library for R Installation. Available https://communities.sas.com/t5/SAS-Communities-Library/The-Open-Source-Integration-node-installation-cheat-sheet/ta-p/223470.

SAS Institute Inc. 2016. SAS Institute white paper. “SAS in the Open Ecosystem.” Available https://www.sas.com/content/dam/SAS/en_us/doc/whitepaper1/sas-in-open-ecosystem-108574.pdf.

SAS Institute Inc. 2017. *SAS Model Manager 14.2: Macro Reference.* Cary, NC: SAS Institute Inc. Available http://support.sas.com/documentation/cdl/en/mdlmgrmacro/69923/PDF/default/mdlmgrmacro.pdf.

SAS repository on GitHub with code snippets. Available https://github.com/sassoftware/enlighten-integration.

SAS repository on GitHub for Python interface module to the SAS System. Available https://github.com/sassoftware/saspy.

SAS repository on GitHub for SAS Kernel for Jupyter Notebook. Available https://github.com/sassoftware/sas_kernel.

ACKNOWLEDGMENTS

Thanks to Andrew Pease, Larry Orimoloye, Cristina Conti, Sascha Schubert and Douglas Liming for their contributions to this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Tuba Islam
SAS Software Inc.
tuba.islam@sas.com
www.linkedin.com/in/tubaislam