

# Develop & Deploy Non-Models in SAS Viya

## Using Model Studio and Open-Source IDE

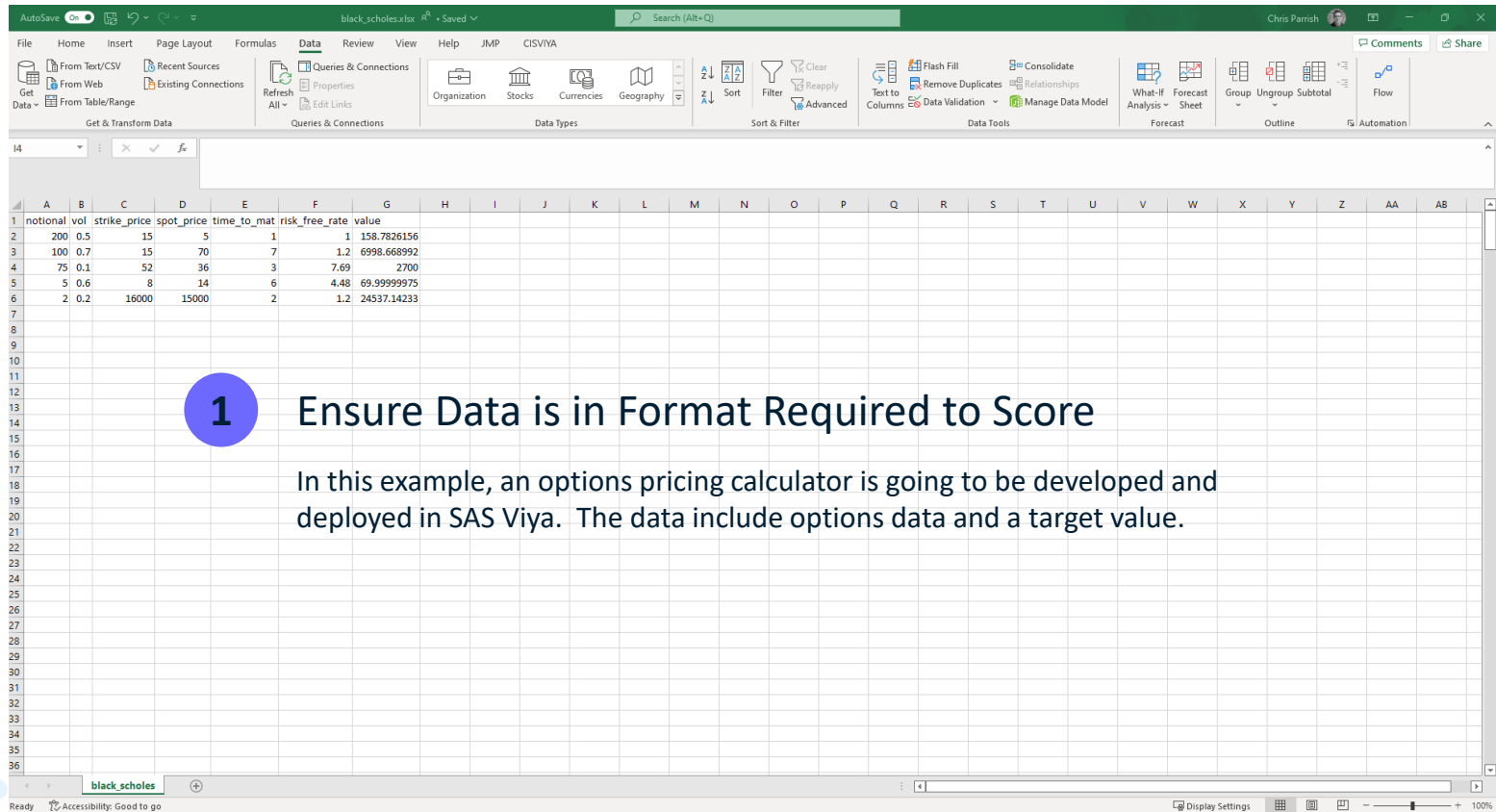
Chris Parrish, Sr. Data Scientist

[chris.parrish@sas.com](mailto:chris.parrish@sas.com)

May 2023

# Steps

- 1 Ensure Data is in Format Required to Score
- 2 Import Data into SAS Viya
- 3 Create New Model Studio Project / Create New OS Script
- 4 Create Score Code
- 5 Register Non-Model to Model Manager
- 6 Create Score Test to Validate Score Code
- 7 Publish Model
- 8 Create Score Test to Validate Publishing



The screenshot displays the SAS Viya user interface. At the top, a green header bar contains the 'AutoSave' toggle, the file name 'black\_scholes.xlsx', and a search bar. Below this is a ribbon menu with tabs for File, Home, Insert, Page Layout, Formulas, Data, Review, View, Help, JMP, and CISVIA. The 'Data' tab is active, showing various data management tools. The main workspace is a spreadsheet with columns A through AB and rows 1 through 36. The data is organized into a table with the following headers and values:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB
1	notional	vol	strike_price	spot_price	time_to_mat	risk_free_rate	value																					
2	200	0.5	15	5	1	1	158.7826156																					
3	100	0.7	15	70	7	1.2	6998.668992																					
4	75	0.1	52	36	3	7.69	2700																					
5	5	0.6	8	14	6	4.48	69.99999975																					
6	2	0.2	16000	15000	2	1.2	24537.14233																					

Below the spreadsheet, a status bar shows 'Ready' and 'Accessibility: Good to go'. The bottom right corner of the interface includes a 'Display Settings' icon and a zoom level of 100%.

# 1 Ensure Data is in Format Required to Score

In this example, an options pricing calculator is going to be developed and deployed in SAS Viya. The data include options data and a target value.

AvailableData SourcesImport (1)

Import (1) ?

black\_scholes.csvLocal file

black\_scholes.csv

The table was successfully imported on May 10, 2023 03:31:59 PM and is ready for use. [Actions](#)

Target table name: \*black\_scholes

Target location: \*cas-shared-default/Public

Find

☐ Save as an in-memory table only

If target table name exists:

☐ Cancel import

☒ Replace file

Label:Enter label

Format: sashdat

File SpecificationsAdvanced

Input file delimiter:Comma

Scanned rows: 20

Locale:Enter locale

Source encoding: \*UTF-8

☒ First row contains column names

☒ Convert character columns to variable size

2

## Import Data into SAS Viya

This can be accomplished several ways. In this example, data is being imported from a local csv file.

black\_scholes

Data Pipelines Pipeline Comparison Insights

Filter								
<input type="checkbox"/>	Variable Name	Label	Type	Role	Assess for Bias	Level	Number of Levels	Mean
<input type="checkbox"/>	notional		Numeric	Input		Interval	5	76.4000
<input type="checkbox"/>	risk_free_rate		Numeric	Input		Interval	4	3.1140
<input type="checkbox"/>	spot_price		Numeric	Input		Interval	5	3,025.0000
<input type="checkbox"/>	strike_price		Numeric	Input		Interval	4	3,218.0000
<input type="checkbox"/>	time_to_mat		Numeric	Input		Interval	5	3.8000
<input type="checkbox"/>	value		Numeric	Target		Interval	5	6,892.9188
<input type="checkbox"/>	vol		Numeric	Input		Interval	5	0.4200

&gt;&gt; BLACK\_SCHOLES

Columns:

7

Rows:

5

Label:

(not available)

Location:

cas-shared-default/Public

3

## Create New Model Studio Project

Add data and assign target.

## black\_scholes

Data Pipelines Pipeline Comparison Insights

## Nodes

Filter

&gt; Data Mining Preprocessing

&gt; Supervised Learning

&gt; Postprocessing

v Miscellaneous

Create Scored Table

Data Exploration

Open Source Code

SAS Code

Save Data

Score Data

Scorecard

Segment Profile

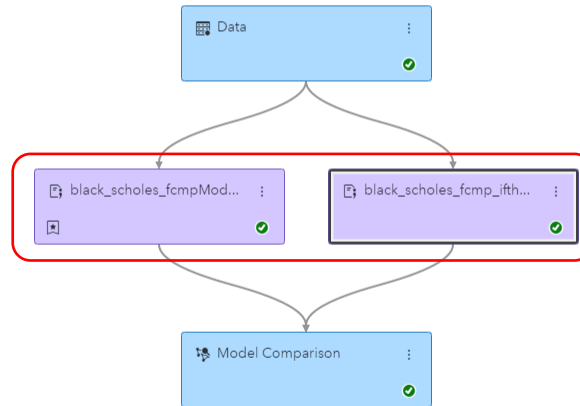
Pipeline 1 +

Run pipeline

4

## Create Score Code

Create a pipeline and include a SAS Code node.  
Open the editor.



black\_scholes\_fc... ⌕ ▶ ⚙ ?

Description:

Runs SAS code.

Open code editor

☐ Training data only☒ Use the exact percentile method for lift calculations

## Post-training Properties

Changing these properties will not retrain the model.

v Model Interpretability

&gt; Global Interpretability

&gt; Local Interpretability

&gt; PD/ICE Options

Seed:

12,345

black\_scholes &gt; black\_scholes\_fcmp\_ifthenModelStudio

## Macros

Filter

## DATA: VARIABLES

dm\_interval\_input  
dm\_binary\_input  
dm\_nominal\_input  
dm\_ordinal\_input  
dm\_unary\_input  
dm\_id  
dm\_segment  
dm\_text  
dm\_dec\_target  
dm\_offset  
dm\_key  
dm\_predicted\_var  
dm\_into\_var  
dm\_freq

## UTILITIES

dmcas\_varmacro  
dmcasregister  
dmcasreport  
dmcasmetachange  
dmcascopyfile  
dmcascopybinaryfile  
dmcascheckMacroVar

## Training Code

```
1 /* SAS code */
2
3 proc fcmp;
4   function black_scholes(notional, vol, strike_price, spot_price, time_to_mat, risk_free_rate);
5     d1 = (log(spot_price/strike_price) + (risk_free_rate+vol**2/2)*time_to_mat)/(vol*sqrt(time_to_mat));
6     d2 = d1 - vol*sqrt(time_to_mat);
7     option_price = cdf('NORMAL', d1)*spot_price-cdf('NORMAL', d2)*strike_price*exp(-risk_free_rate*time_to_mat);
8     if vol > 0.1 then option_val = notional * option_price;
9     else option_val = spot_price/risk_free_rate;
10    return (option_val);
11  endfunc;
12 run;
```

4

## Create Score Code

The training code is not relevant, but it is included to show how to this would run in a SAS script. The DS1 score code should include calculations, if/then, do-loop, etc. statements only. There is no need to use the macro variables. In this example, “option\_val” is the desired calculation value.

## Scoring Code

```
1 /* SAS ds1 score code */
2
3 d1 = (log(spot_price/strike_price) + (risk_free_rate+vol**2/2)*time_to_mat)/(vol*sqrt(time_to_mat));
4 d2 = d1 - vol*sqrt(time_to_mat);
5 option_price = cdf('NORMAL', d1)*spot_price-cdf('NORMAL', d2)*strike_price*exp(-risk_free_rate*time_to_mat);
6 if vol > 0.1 then option_val = notional * option_price;
7 else option_val = spot_price/risk_free_rate;
8
9
```

black\_scholes

Data Pipelines Pipeline Comparison Insights

Filter

<input type="checkbox"/>	Champion	Challenger	Registered	Name	Algorithm Name	Pipeline Name	Average Squared Error	
<input type="checkbox"/>			<input checked="" type="checkbox"/>	black_scholes_fcmpModelStudio	SAS Code	Pipeline 1	84,162,237.250	
<input checked="" type="checkbox"/>			<input checked="" type="checkbox"/>	black_scholes_fcmp_ifthenMod...	SAS Code	Pipeline 1	84,162,237.250	

- Set as champion
- Remove challenger models
- Register models
- Publish models
- Score holdout data
- Download score API
- Download score code
- Manage Models

5

## Register Non-Model to Model Manager

Under Pipeline Comparison, select 'Register Model' for the non-model to be registered. These models have already been registered, so the option is greyed out.

SAS Code

```

1  /* SAS code */
2
3  proc fcmp;
4      function black_scholes(notional, vol, strike_price, spot_price, time_to_mat, risk_free_r
5          d1 = (log(spot_price/strike_price) + (risk_free_rate+vol**2/2)*time_to_mat)/(vol*sqrt
6          d2 = d1 - vol*sqrt(time_to_mat);
7          option_price = cdf('NORMAL', d1)*spot_price-cdf('NORMAL', d2)*strike_price*exp(-risk
8          if vol > 0.1 then option_val = notional * option_price;
9              else option_val = spot_price/risk_free_rate;
10             return (option_val);
11         endfunc;
12 run;
13
14

```

Node Score Code

```

1  /* SAS ds1 score code */
2
3  d1 = (log(spot_price/strike_price) + (risk_free_rate+vol**2/2)*time_to_mat)/(vol*sqrt(time_t
4  d2 = d1 - vol*sqrt(time_to_mat);
5  option_price = cdf('NORMAL', d1)*spot_price-cdf('NORMAL', d2)*strike_price*exp(-risk_free_ra
6  if vol > 0.1 then option_val = notional * option_price;
7      else option_val = spot_price/risk_free_rate;
8
9  *-----*;
10 * Accounting for missing predicted variable;
11 *-----*;
12 label "P_value"n='Predicted: value';
13 if "P_value"n = . then "P_value"n =6892.9187875;
14

```

Path Score Code

```

1  /*-----*/
2  /* Product:      Visual Data Mining and Machine Learning */
3  /* Release Version: V2022.09 */
4  /* Component Version: V2022.09 */
5  /* CAS Version:    V.04.00M0P09192022 */
6  /* SAS Version:    V.04.00M0P091922 */
7  /* Site Number:    70180938 */
8  /* Host:          sas-cas-server-default-client */
9  /* Encoding:      utf-8 */
10 /* Java Encoding:  UTF8 */
11

```

DS2 Package Code

```

1  /*-----*/
2  /* Product:      Visual Data Mining and Machine Learning */
3  /* Release Version: V2022.09 */
4  /* Component Version: V2022.09 */
5  /* CAS Version:    V.04.00M0P09192022 */
6  /* SAS Version:    V.04.00M0P091922 */
7  /* Site Number:    70180938 */
8  /* Host:          sas-cas-server-default-client */
9  /* Encoding:      utf-8 */
10 /* Java Encoding:  UTF8 */
11

```



```

202 with open(str(output_path)+str('/ModelProperties.json'), 'w') as outfile:
203     outfile.write(ModelPropertiesObj)
204
205 requirementsObj = json.dumps(requirements, indent = 4)
206 with open(str(output_path)+str('/requirements.json'), 'w') as outfile:
207     outfile.write(requirementsObj)
208
209 #####
210 ### Create Score Code ###
211 #####
212
213 score_script = """
214
215 import numpy as np
216 from scipy.stats import norm
217
218 def option_value(notional, vol, strike_price, spot_price, time_to_mat, risk_free_rate):
219     "Output: option_val"
220     d1 = (np.log(spot_price/strike_price) + (risk_free_rate+vol**2/2)*time_to_mat)/(vol*np.sqrt(time_to_mat))
221     d2 = d1 - vol*np.sqrt(time_to_mat)
222     option_price = norm.cdf(d1)*spot_price-norm.cdf(d2)*strike_price*np.exp(-risk_free_rate*time_to_mat)
223     option_val = notional * option_price
224     return float(option_val)
225
226
227 with open((output_path / score_code_name), 'w') as scorecode:
228     scorecode.write(score_script)
229
230 #####
231 ### Create Table to Score Model ###
232 #####
233
234 data_x = [[200, .5, 15, 5, 1, 1]]
235 columns_x = ['notional', 'vol', 'strike_price', 'spot_price', 'time_to_mat', 'risk_free_rate']
236 X = pd.DataFrame(data_x, columns=columns_x)
237 if conn.tableExists(caslib=caslib, name=in_mem_tbl).exists<=0:
238     conn.upload(data=X, casOut={"caslib":caslib, "name":in_mem_tbl, "promote":True})
239
240 #####
241 ### Zip Files & Send to Model Manager ###
242 #####
243
244 from sasctl import Session
245 import sasctl.pzmm as pzmm
246 from sasctl_services.model_repository import ModelRepository as mr
247
248 zip_file = pzmm.ZipModel.zipFiles(fileDir=output_path, modelPrefix=model_name, isViya4=True)
249 sess=Session(hostname, username=username, password=password, verify_ssl=False, protocol="http")
250 with sess:
251     try:
252         mr.get_project(project_name).name
253     except:
254         mr.create_project(project_name, mr.default_repository())
255     mr.import_model_from_zip(model_name, project_name, zip_file, version='latest')
256

```

## 3 Create New OS Script

## 4 Create Score Code

## 5 Register Non-Model to Model Manager

Name	Date Modified
black_scholes_python	4/19/2022 1:39 PM
black_scholes_python.zip	4/19/2022 1:39 PM
black_scholes_pythonScore.py	4/19/2022 1:38 PM
fileMetadata.json	4/19/2022 1:38 PM
inputVar.json	4/19/2022 1:38 PM
ModelProperties.json	4/19/2022 1:38 PM
outputVar.json	4/19/2022 1:38 PM

```

import keyring
import runpy
import os

## run script that contains username, password, hostname, working directory, and output directory
## ...OR define directly in this script
from password import hostname, port, protocol, wd, output_dir, hostname_dev, port_dev, protocol_dev

runpy.run_path(path_name='password.py')
username = keyring.get_password('cas', 'username')
password = keyring.get_password('cas', username)
metadata_output_dir = 'outputs'

#####
### Environment ###
#####

import swat
import pandas as pd

conn = swat.CAS(hostname=hostname, port=port, username=username, password=password, protocol=protocol)
print(conn.serverstatus())

#####
### Identify Table in CAS ###
#####

## caslib and table to use in modeling
caslib = 'Public'
in_mem_tbl = 'BLACK_SCHOLES'

## load table in-memory if not already exists in-memory
if conn.table.tableExists(caslib=caslib, name=in_mem_tbl).exists<=0:
    conn.table.loadTable(caslib=caslib, path=str(in_mem_tbl+str('.sashdat')),
                        casout={'name':in_mem_tbl, 'caslib':caslib, 'promote':True})

## show table to verify
conn.table.tableInfo(caslib=caslib, wildignore=False, name=in_mem_tbl)

```

<input type="checkbox"/>	Name	Role	Model Function	Project Version	Score Code Type	Algorithm	Date Modified
<input type="checkbox"/>	black_scholes_fcmp_ifthenModelStudio(Pipeline 1)		Prediction	Version 3 (3.0)	DATA step	sascode	May 10, 2023 05:16 PM
<input type="checkbox"/>	black_scholes_fcmpModelStudio(Pipeline 1)		Prediction	Version 3 (3.0)	DATA step	sascode	May 10, 2023 03:52 PM
<input type="checkbox"/>	black_scholes_python		Prediction	Version 1 (1.0)	Python	UDF	May 10, 2023 03:58 PM
<input type="checkbox"/>	black_scholes_r		Prediction	Version 1 (1.0)	R	UDF	Mar 29, 2023 06:49 AM

Home  
 Models  
 Projects  
 Deployments  
 Tasks

black\_scholes

Models Variables Properties Files **Scoring** Performance Workflow History

Tests

Publishing Validation

<input type="checkbox"/>	Name	Results	Status
<input type="checkbox"/>	black_scholes_fcmp		
<input type="checkbox"/>	black_scholes_fcmp_ifthen		

Test Results				
d1	d2	option_price	option_val	EM_PREDICTION
6.2933487751	4.4413228574	69.996626994	6999.6626994	6892.9187875
131.15824963	130.98504455	35.999999995	4.6814044213	6892.9187875
19.40514065	17.935446805	14	70	6892.9187875
8.3985246008	8.1156818883	13548.512747	27097.025495	6892.9187875
0.0527754227	-0.447224577	0.7988072228	159.76144456	6892.9187875

6

## Create Score Test to Validate Score Code

Under the Scoring tab, create a new scoring test with data imported into SAS Viya. The results show that “option\_val” was calculated correctly, so the score code is working as expected. There are “out-of-the-box” SAS Viya columns calculated, including P\_(target) and EM\_PREDICTION, but those can be ignored. They are just the average of the target column.

The screenshot shows the SAS Model Manager interface for a project named 'black\_scholes'. The left sidebar contains navigation links: Home, Models, Projects (selected), Deployments, and Tasks. The main area displays a table of models under the 'Models' tab. The table has columns: Name, Role, Model Function, Project Version, Score Code Type, Algorithm, and Date Modified. Two models are listed: 'black\_scholes\_fcmpModelStudio(Pipeline 1)' (Version 1 (1.0)) and 'black\_scholes\_fcmpModelStudio(Pipeline 1)' (Version 2 (2.0)). The second model is selected. A 'Publish' button is visible in the top right corner of the main area. A 'Publishing Results' dialog box is open in the foreground, showing the results of the publish action.

Name	Published Name	Status	Output	Micro Analytic Module
black_scholes_fcmpModelStudio(Pipeline 1)	black_scholes_fcmpModelStudio__Pipeline_1_	✓ Published succes...	Log	/microanalyticScore/modules/black_scholes_fcmpmodelstudio__p

7

## Publish Model

Under the Models tab, select the model and click on “Publish.” The available deployment destinations will be listed. In this example, the model was published SAS Viya Micro Analytic Services.

<input type="checkbox"/>	Name	Results	Status	Published Name	Model Name	Project Version	Target Destination	Date Last Run	Date Modified	Modified By	
<input type="checkbox"/>	<a href="#">black_scholes_fcmp</a>		✓	black_scholes_fcmpModelStudio__Pipeline_1_	black_scholes_fcmpModelStudio (Pipeline 1) (1.0)	Version 3 (3.0)	maslocal	May 10, 2023 03:53 PM	May 10, 2023 03:53 PM		
<input type="checkbox"/>	<a href="#">black_scholes_fcmp_ifthen</a>		✓	black_scholes_fcmp_ifthenModelStudio__Pipeline_1_	black_scholes_fcmp_ifthenModelStudio (Pipeline 1) (1.0)	Version 3 (3.0)	maslocal	May 10, 2023 05:17 PM	May 10, 2023 05:17 PM		
<input type="checkbox"/>	<a href="#">black_scholes_python</a>		✓	black_scholes_python	black_scholes_python (2.0)	Version 1 (1.0)	maslocal	May 10, 2023 03:59 PM	May 10, 2023 03:59 PM		

SAS® Model Manager - Manage Models

black\_scholes > black\_scholes\_fcmp\_ifthen

Test Results

P_VALUE	EM_PREDICTION	D1	D2	OPTION_PRICE	OPTION_VAL	dcm_error_response	dcm_http_response_c...
6.2933487751	4.4413228574	69.996626994	6999.6626994			201	
131.15824963	130.98504455	35.999999995	4.6814044213			201	
19.40514065	17.935446805	14	70			201	
8.3985246008	8.1156818883	13548.512747	27097.025495			201	
0.0527754227	-0.447224577	0.7988072228	159.76144456			201	

8

## Create Score Test to Validate Publishing

Going back to the Scoring tab, all published models will surface on this page. To test the publishing destination, follow the same steps as score code validation. The publishing test indicated that the calculation is the same as the scoring test. At this point, new data can be fed into the deployment endpoint to score.