Using Python with SAS

Using SAS Methods on SAS Data

When programming in Python be aware that variable names are case sensitive, indenting code is important and effects the syntax of Python. Python starts with 0 not 1 as SAS does.

Setting up a session and gaining information about it

Import saspy python library to enable a SAS session	Import saspy
Import pandas python library to enable tabular data in Python For more information- https://blog.dominodatalab.com/pandas-for-sas-users-part-1/	Import pandas
Start a SAS Session	sas = saspy.SASsession()
Provides details of the SAS session	sas
List assigned libraries	sas.assigned_librefs()
List data sets in a library	sas.datasets('sasuser') sas.list_tables('sasuser')
Descriptor/Metadata information about the data set.	cars=sas.sasdata('cars,'sashelp') cars.columnInfo() cars.contents()
End the SAS Session	sas.endsas()

Getting Help

To understand what SAS code is being generated. When set to True, no output is generated. Make sure it is set to false when finished.	sas.teach_me_SAS(True) cars.top("make",order="freq") sas.teach_me_SAS(False)
To see what methods are available, type the class name and then press the TAB key.	sas.
To get help about a particular method, add a ?. The example provides information about what is required to produce a Histogram.	Cars.hist?

Accessing Data, creating Libraries and manipulate data

Accessing a SAS Data Set Data Set options can be added. Drop, keep, formats, obs, firstobs	cars=sas.sasdata('cars,'sashelp') cl2=sas.sasdata('class', 'sashelp',dsopts={'where':'sex = "M"', 'keep':'name age sex'})
Create a SAS library	sas.saslib('pg1', " 'c:\data\user' ")
Read in a CSV file. By default a data set called _csv is created in the Work library. Assigning a variable allows it to be used in analysis (see below).	sas.read_csv(file="c:\\useful data\\class.csv") ab=sas.read_csv(file="c:\\useful data\\class.csv") ab.head()

A named data set can be created.	ab=sas.read_csv(file="c:\\useful data\\class.csv",libref="work", table="class")
Check to see **if a data set exists**. A Boolean result ('True' or 'False') is returned.	sas.exist("class", "work")
Sort the data set. The variable created can then be used for analysis. Note, if you don't create an output data set, the original data is modified.	Asc=cars.sort('invoice') Desc=cars.sort('descending invoice') Desc_orig=cars.sort(' origin descending invoice') cs=cars.sort('descending invoice', out='cars_sorted') cs.head()
Enables a **random sample to be created**. A new variable (_PartInd_) with a value of 1 or 0 is created. Default split is 70/30. Values assigned randomly. The sample can be stratified on a variable and the split can also be changed. The 2nd and 3rd examples produce approx. 43 observations. The 3rd example ensures that 10% of each 'type' is selected.	cp=cars.partition() cp10pct=cars.partition(fraction=.1) cp10pcttype=cars.partition('type',fraction=.1)
Filter the data. The example uses the _Partind_ column created in the previous example.	cars_train = cars.where('_PartInd_=1') cars_test = cars.where('_PartInd_=0')
This method **imputes missing values** for a particular variable or variables. The impute method can be Mean or Median or any other statistic.	cor=cl_miss.impute({'median': ['height']}) cor=cl_miss.impute({'mean': ['height','age']})
Pandas. Are there any missing values	df.isnull().any()
Change all missing values to 0	df0 = df.fillna(0)
replaces missing values with their mean	df.fillna(df.mean())
replaces 1 column of missing values with its median	df.fillna(df.median()[['Weight']])
Symput creates macro variables **Symget** retrieves macro variables which can then be passed to Python variables. This method can be used to transfer values from sas.submit session to Python.	sas.symput('macvar',2019) x=sas.symget('macvar') print(x)

Simple Data Analysis

Print the first 5 rows	cars.head()
Print the last 5 rows	cars.tail()
Calculate statistics	cars.means() or cars.describe()
Top 10 /Top n	cars.top("make",order="freq") cars.top("make",5,order="freq")
Generate Heat Map	cars.heatmap('origin','type') cars.heatmap('origin','type',options='colorstat=pct colormodel=(pink lilac purple)')
Create Histograms	cars.hist('invoice')
Create Scatter Plots	cars.scatter('invoice','mpg_city') cars.scatter('invoice','mpg_city/ Group=type')
Create Bar Charts	cars.Bar('type') cars.bar('type/ group =origin ') cars.bar('type/ group =origin response= invoice ')
Create Series Plots	stock=sas.sasdata('stocks','sashelp') stock.series('date','adjclose/groupby = stock')

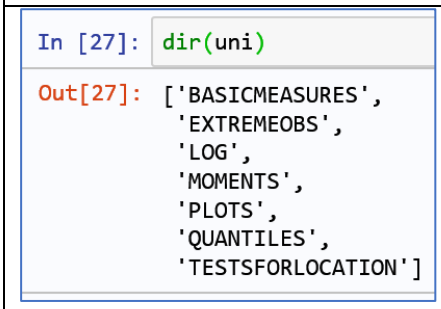
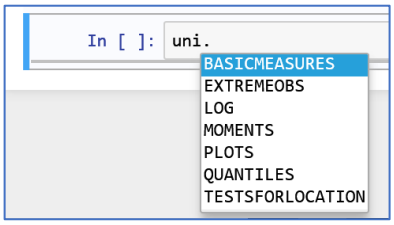
Additional Objects

Other objects are available that provide different SAS Procedures. They are grouped into the following groups – Utility, Machine Learning, Statistics, Econometric and Time Series, Quality Control and SAS VIYA VDML For full information https://sassoftware.github.io/saspy/api.html#procedure-syntax-statements.

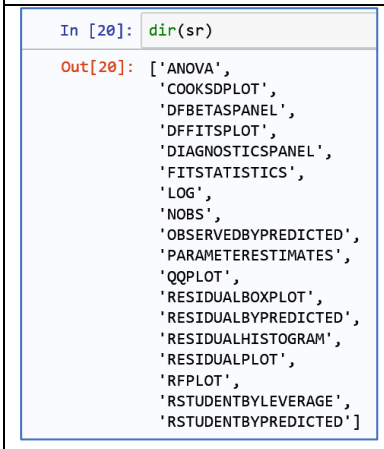
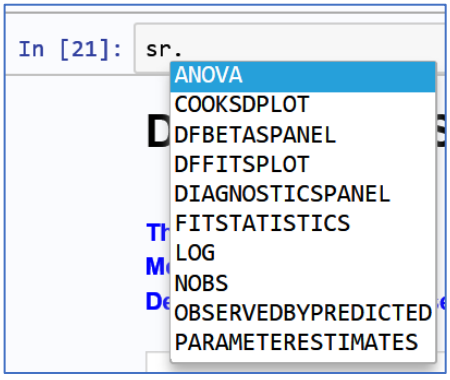
What follows are examples from the Utility and Statistics groups. Accessing the procedures is accessed and executed in a slightly different way.

Using the Univariate Method from the Util object.

To access the Univariate() method, firstly assign a variable to the sas.sasutil library of procedures.	util=sas.sasutil()
Create an object which contains the output from the Univariate procedure	uni=util.univariate(var='invoice',data="sashelp.cars")

There are 2 ways to explore what results have been created	
	uni. <Hit tab key> 
To access 1 of the results	Uni.moments
To access all of the results	uni.ALL()
Other procedures available include HPSAMPLE, HPBIN and HPIMPUTE	

Using Proc Reg from the SASstat object

To access the reg() method, firstly assign a variable to the sas.SASstat library of procedures.	stat=sas.sasstat()
Create a SASresults object which contains the output from the Reg procedure	cars=sas.sasdata('cars,'sashelp') sr=stat.reg(model='horsepower=Cylinders Enginesize',data=cars)
There are 2 ways to explore what results have been created	
	sr. <Hit tab key> 
To access 1 of the results	sr.diagnosticspanel
To access all of the results	sr.ALL()
Several other procedures are available. Enter stat. and then hit the Tab key to view a list.