Predictive Modeling in SAS Viya Workbench for Learners

SAS Guided Demo

Purpose

This SAS Guided Demo provides participants a sneak preview of our newly launch SAS Viya Workbench course, *Modern Data Science with SAS Viya Workbench and Python*. The full course is much longer and covers the entire analytics lifecycle – from data wrangling to model deployment. However, this SAS Guided Demo focuses on predictive modeling in SAS and Python – aka the really fun parts of the course. So, please enjoy these modeling sections – while learning about the new coding platform from SAS!

The Scenario

Welcome to your first day at Styled and Sophisticated Enterprises (SASe), the premier online personal styling service in the world! SASe connects clients across the globe to a personal stylist and offers a premium clothing subscription box. No longer will busy professions need to search for clothing online. We'll give you a modern look, that will fit your style in both the boardroom and out on the town.

As a data scientist in our Customer Retention Department at SASe, you'll help us use machine learning models to help us analyze customer churn. Customer "churn" simply means that our client has canceled their premium clothing subscription. And since it often is more difficult to find a new customer than keep an existing one, you will help us identify which clients are likely on the cusp of churning, so that we can find ways to retain them.

Your onboarding will take several weeks, but we'll get you started today by examining the work of your predecessor. This work will expose you to the typical activities and challenges you'll face in your daily job.

For our lessons today, I will walk you through our customer retention project, which follows the classic analytics lifecycle. My goal here isn't to have you understand everything. But, instead, I want to expose you to the wider data engineering and data science scope of the project – so that you can follow-up with more detailed analysis later.

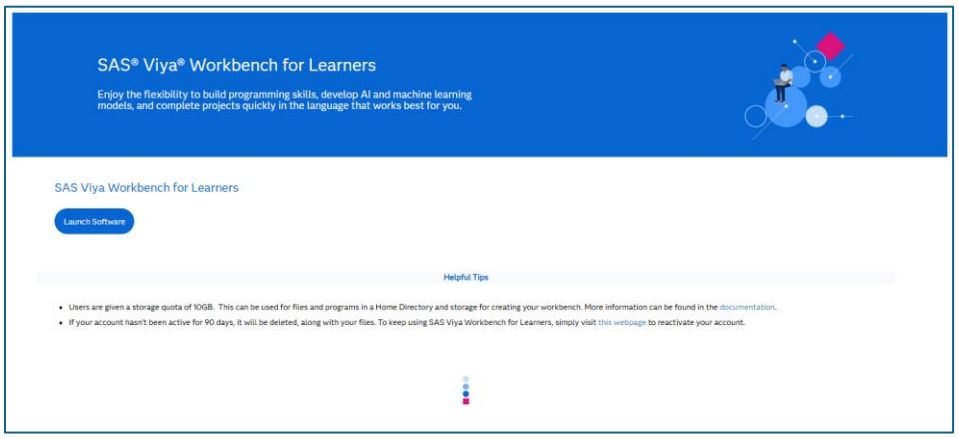
Additionally, we'll show you your new coding environment, SAS Viya Workbench. SAS Viya Workbench is a lightweight, standalone development, modeling and coding environment for data scientists like you and me.

I hope that you're as excited as I am… so let's get started!

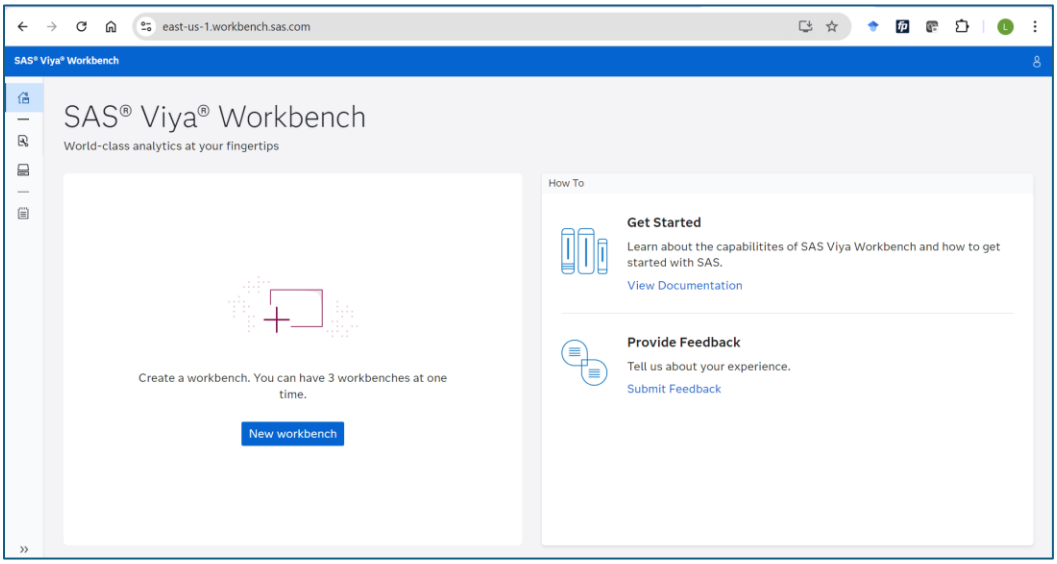
How to Get Started in SAS Viya Workbench for Learners

Step 1 is all about gaining access to the SAS Viya Workbench for Learners (WFL) software and then successfully getting you into the environment. So, this section is truly the beginning… and needs to be followed exactly as specified below. And then you are just a few clicks away from running all the programs required to predict customer churn using both SAS and Python, using the code left behind by your predecessor.

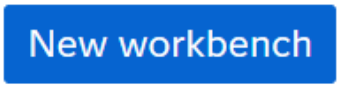
- Like any good analytics adventure, we start with software. To access SAS Viya Workbench for Learners, start here: www.sas.com/workbenchforlearners
- Once you have a SAS profile and accept the terms of use, you should be taken to the SAS Viya Workbench for Learners product page: Course: SAS Viya Workbench for Learners. It looks like this:



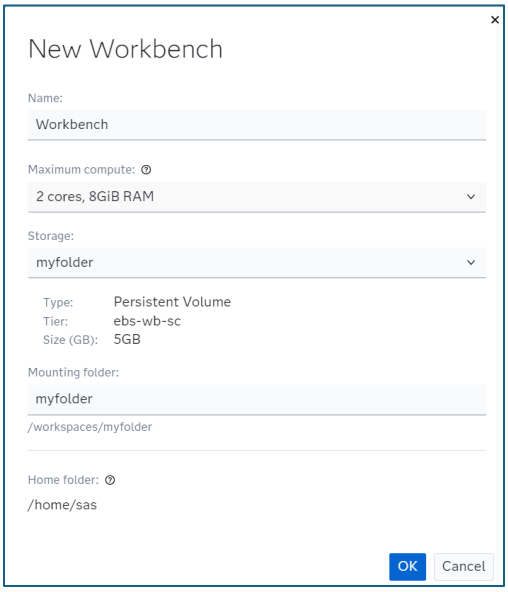
- Click the **Launch Software** button above to get started!
- If this is your first time in SAS Viya Workbench, you'll need to create a **New workbench**. And creation is just a few button clicks away:



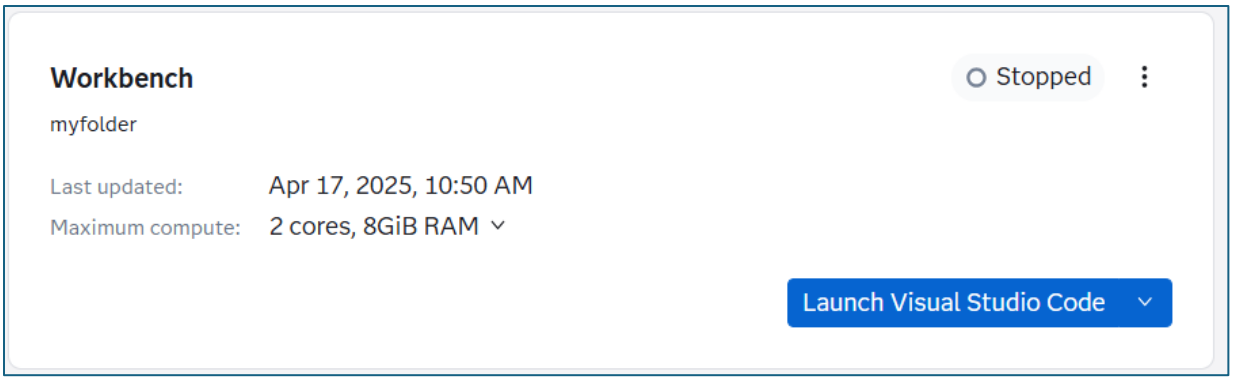
- But before getting right to it, spend a little time getting acquainted with the WFL landing page. For example, check out the **Get Started** resources, or those **Workbenches** and **Storage** icons on the left side. In other words, just become more and more comfortable with the resources available on the WFL landing page. And when you're satiated, click that **New workbench** button:



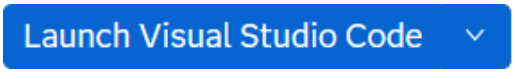
- The following appears:



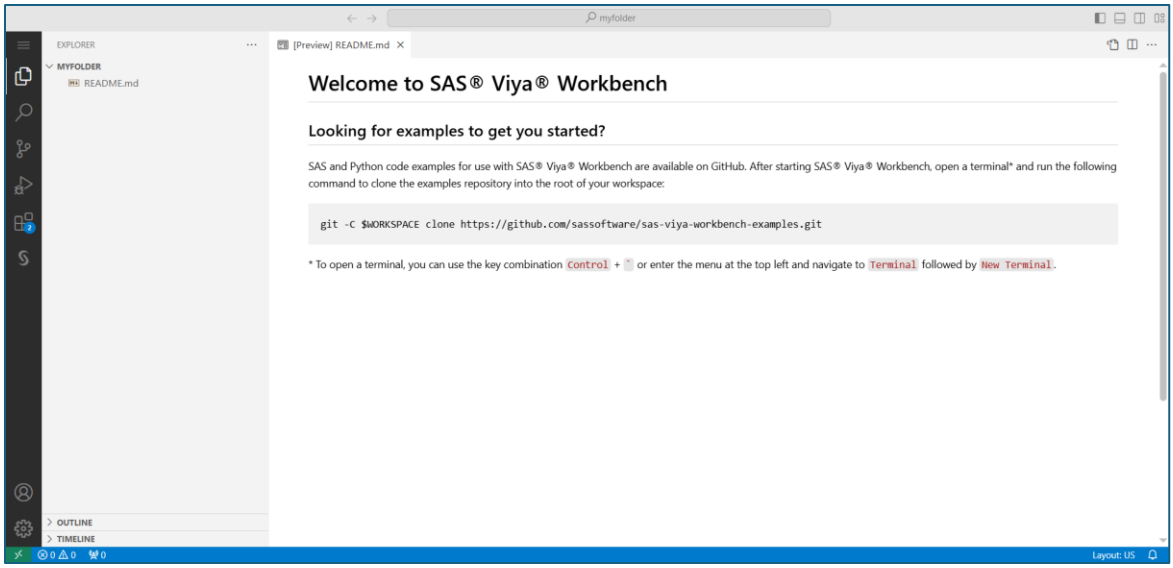
- For **Name**, let's just keep **Workbench** to simplify. And, in fact, just keep the other settings at their default and then click **OK**. Success looks like this:



- Next, in our new workbench, click the **Launch Visual Studio Code** button, here:

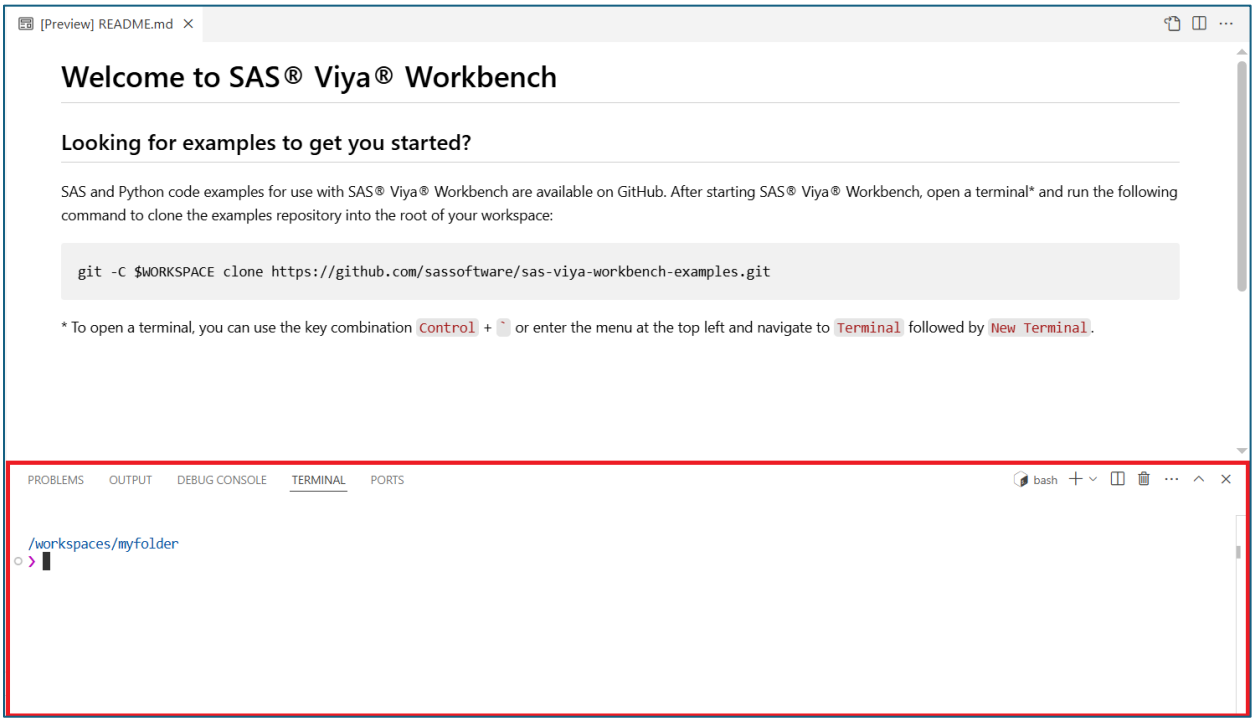


- Your workbench container loads! And welcome to **Visual Studio**!

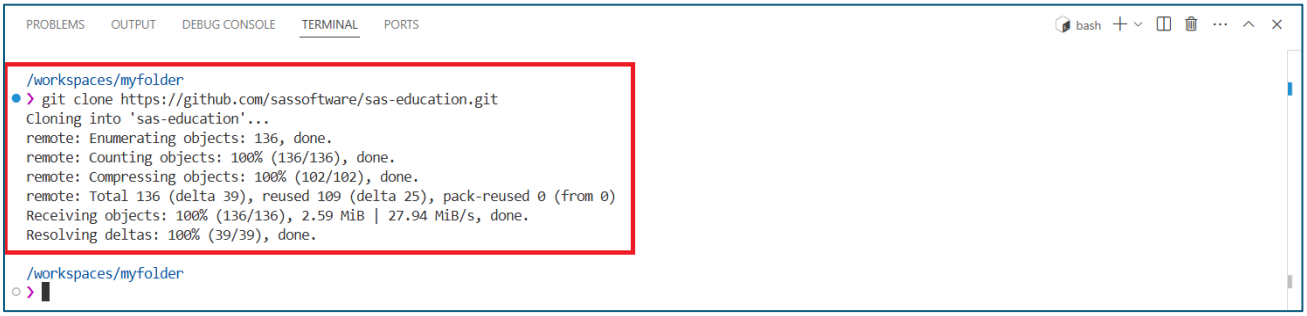


- If this is your first time in the environment, explore the Visual Studio environment for a bit. Because we're all about exploring and learning here.
- Once you're more comfortable with Visual Studio, the last step is to access the data for this SAS Guided Demo. You'll want to access the **Terminal** – which is likely hidden by default. No worries – it can be activated via the **Toggle Panel** if it's not already available. And that button is in the upper-right corner, here:

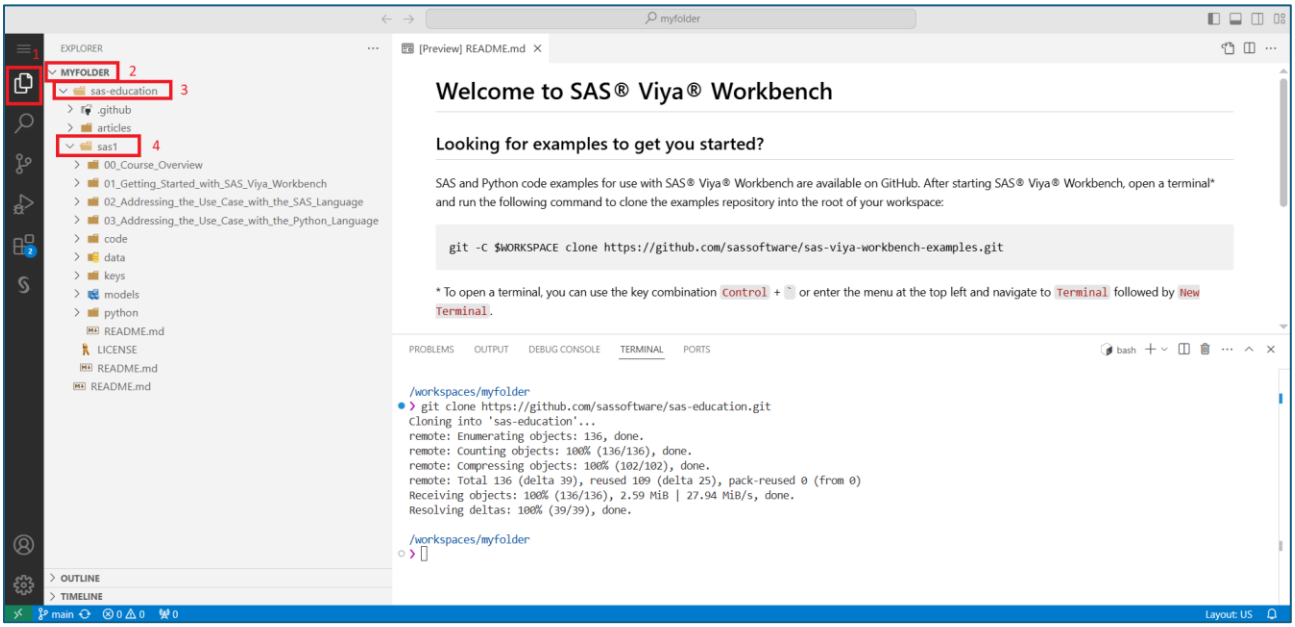
- And a happy terminal will look like the following:



- You're almost there! Simply type the following line of code into the Terminal:
  git clone https://github.com/sassoftware/sas-education.git
- Then hit **Enter**. Success looks like this:



- The files that we'll need for our day 1 adventures at SASe can be found in the **Explorer**. The files are a couple of layers down into the folder, so follow this path: **MYFOLDER** >> **sas-education** >> **sas1**. And here are some click checkpoints to guide you:

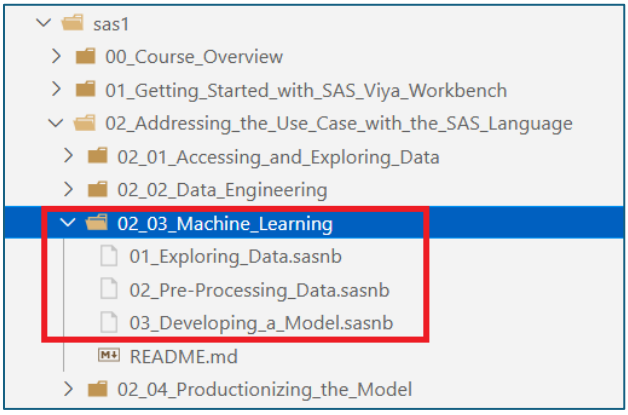


- That's all for this section! Software and data accessed!

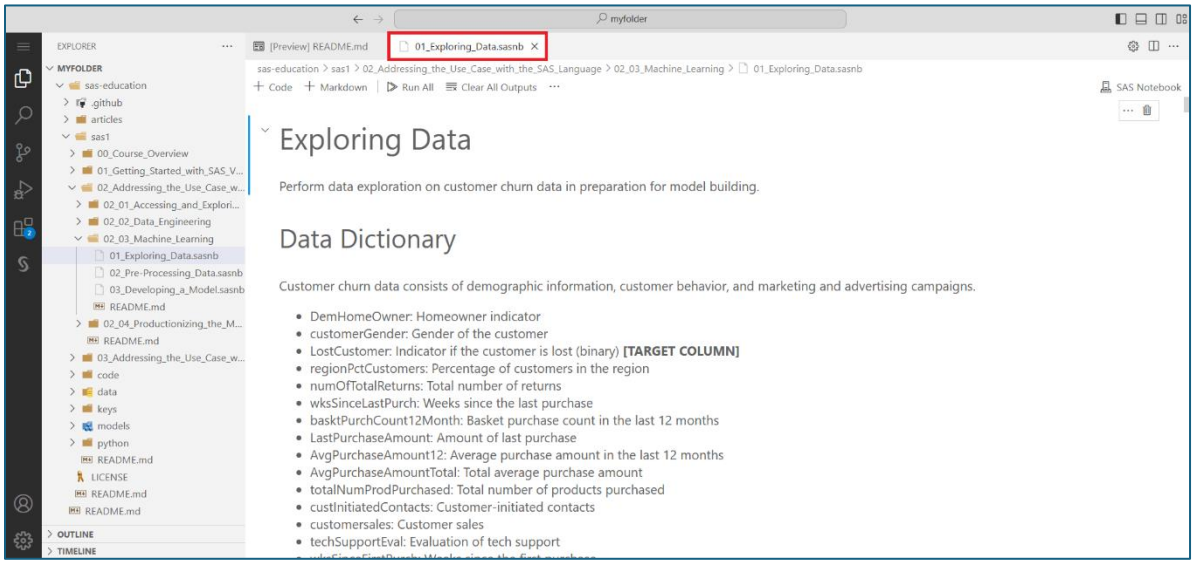
Predictive Modeling using SAS Code in Microsoft Visual Studio

You are mere clicks away from the awesomeness that is running SAS Code in Visual Studio. Without further ado, let's get at it!

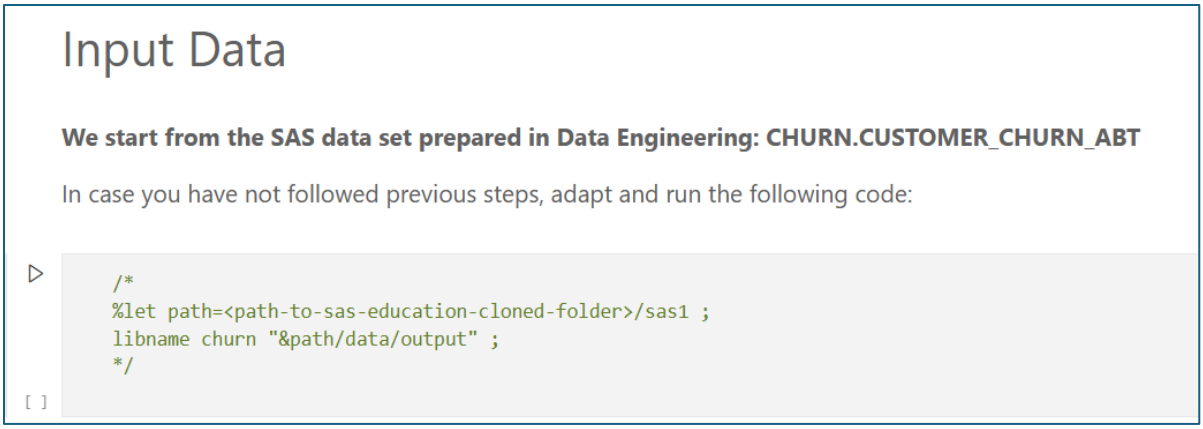
- Open the **02_Addressing_the_Use_Case_with_the_SAS_Language** folder under the **sas1** folder in SAS Viya Workbench. Then find – and expand – the **02_03_Machine_Learning** folder. Three notebooks await!



- And, yes, those numbers denote the order you'll run them in. So, start with *01_Exploring_Data.sasnb*... as in double-click it. You'll see your very first **SAS Notebook** in Visual Studio:



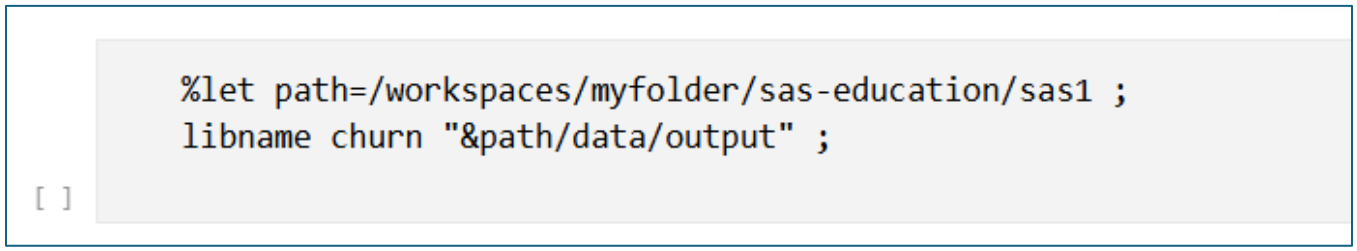
- Nice! Before we can run any of the code, we need to do one more thing. And that's to adjust the location of the **Input Data**, found in this cell:



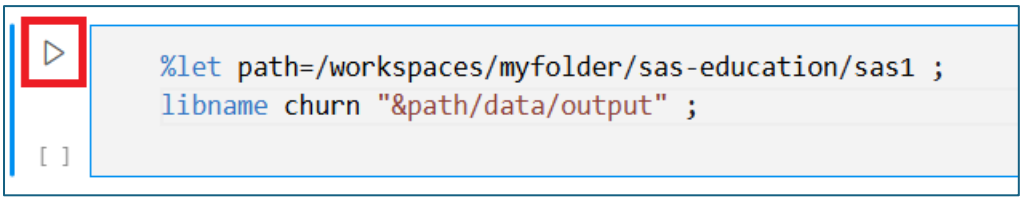
- As noted, we'll want to activate the code and identify the path to the SAS1 data. This path is provided below:

```
%let path=/workspaces/myfolder/sas-education/sas1 ;
```

- With some copy-pasting (and typing), and activating the SAS code, your cell should look akin to this:



- Execute that cell by clicking on the **Execute Cell** button in Visual Studio:

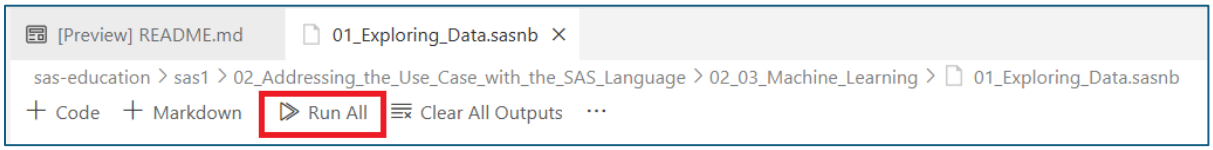


- And, hopefully, you'll see blue happiness:

```
%let path=/workspaces/myfolder/sas-education/sas1 ;
libname churn "&path/data/output" ;
```
[14] ✓ 0.2s SAS

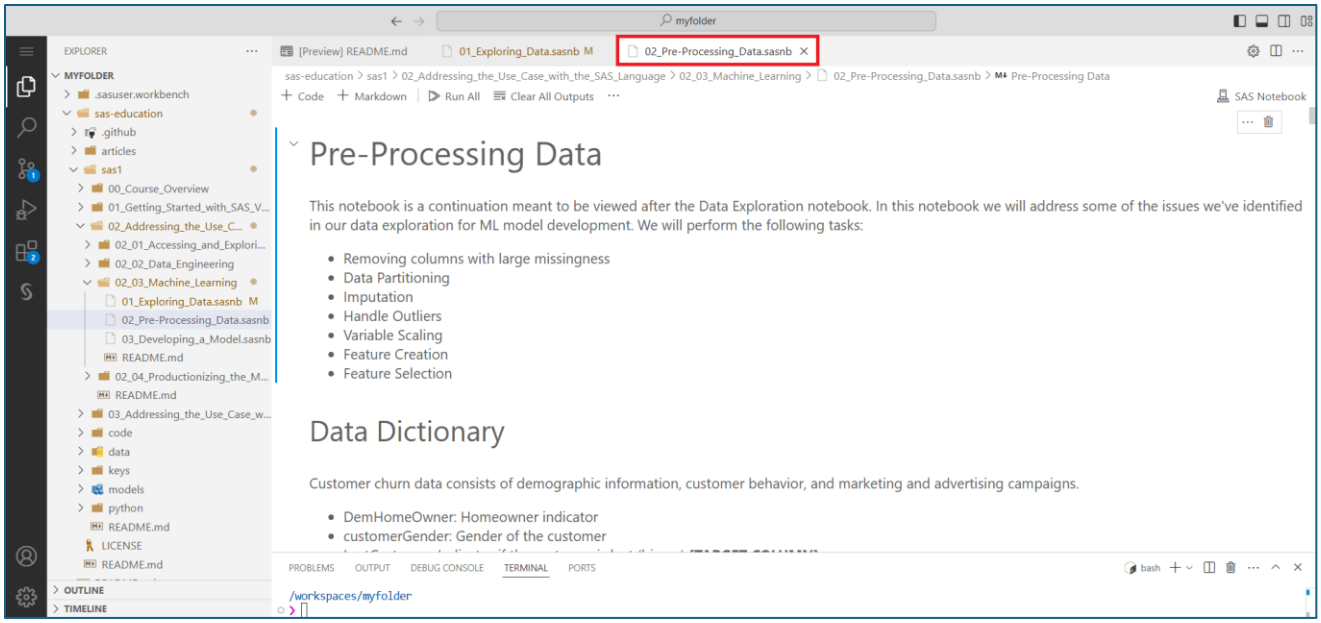
```
294  /** LOG_START_INDICATOR **/
295  title;footnote;ods _all_ close;
296  ods graphics on;
297  ods html5(id=vscode) style=HTMLEncore options(bitmap_mode='inline' svg_mode='inline');
NOTE: Writing HTML5(VSCODE) Body file: sashtml13.htm
298  %let _SASPROGRAMFILE =
298! %nrquote(%nrstr(/workspaces/myfolder/sas-education/sas1/02_Addressing_the_Use_Case_with_the_SAS_Language/02_03_Machine_Learning
298! /01_Exploring_Data.sasnb));
299  %let path=/workspaces/myfolder/sas-education/sas1 ;
300  libname churn "&path/data/output" ;
NOTE: Libref CHURN was successfully assigned as follows:
      Engine:        V9
      Physical Name: /workspaces/myfolder/sas-education/sas1/data/output
301  ;*';*";*/;run;quit;ods html5(id=vscode) close;
```

- Now, there are a couple of ways to continue this "preview" party. You can simply click **Run All**... and run all the code, scroll to absorb some random content, and then just move on to the next program. That magical **Run All** button is found here:

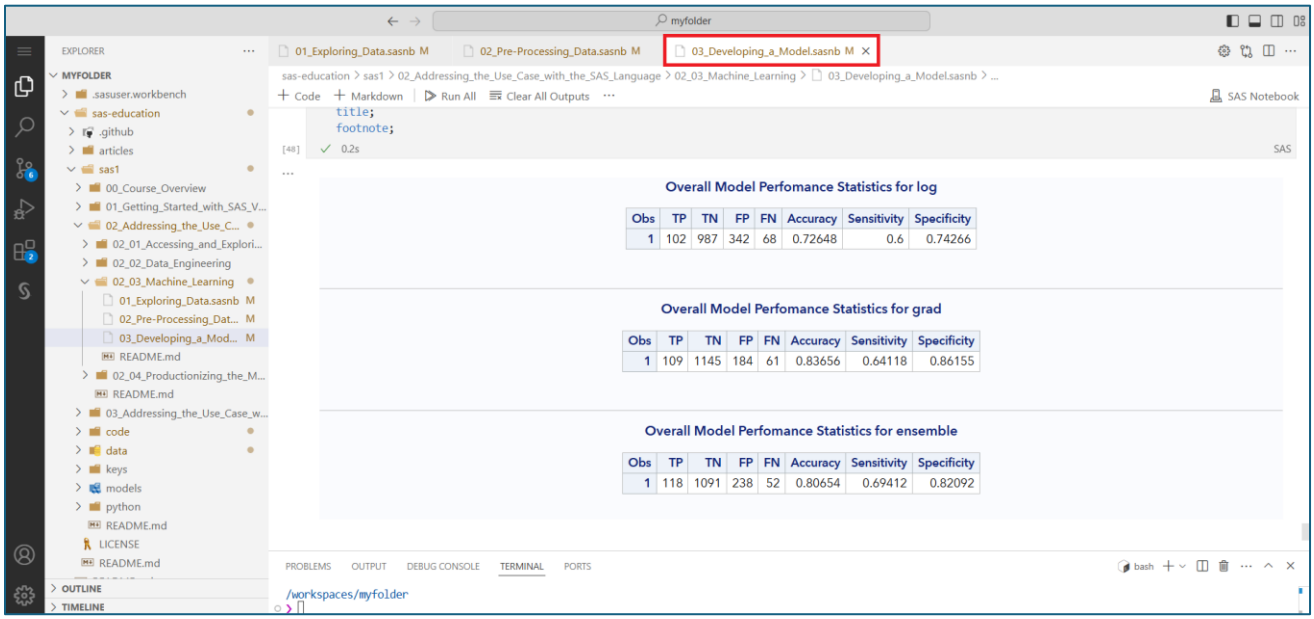


- Or... you can actually walk through the program as a "true" learner, which means running the program cell-by cell. This requires navigating to a cell, finding the little **Play** button, and mashing each and every one of those:

- For those not in the know, "mashing" is a southern verb meaning "to push". Local dialects aside, the goal here is to show you an end-to-end analysis data science project – done in two approaches – once in SAS and once in Python. And, as noted, we're leading with SAS.
- Once you're happy with exploring the data, know that we've got 5 more programs to explore: 2 SAS notebooks and 3 corresponding ones for Python. Because we're going to do (roughly) the same thing in both SAS and Python.
- After ***01_Exploring_Data***, the next step is ***02_Pre-Processing_Data***. Open that notebook... and – in some shape or form – run it all. And make sure not to skip over the description at the top and get right to the code, as that information helps you retain the global "why and what" of the notebook.



- And what is the next step after 02 is completed? All together now: ***to run 03_Developing_a_Model.sasnb***! Yay! Open and run it all. And keep drinking from that firehose… with a completed file appearing as:

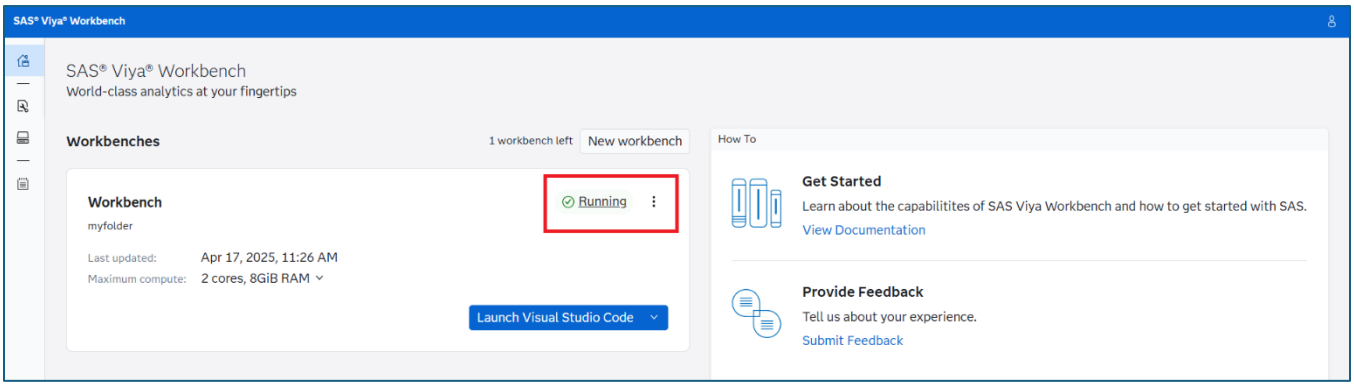


- Now… exhale. And think about what I just made you do. You just ran a TON of code in Visual Studio to estimate the probability of customer churn in our fictitious clothing company… at a breakneck pace. And the output was (hopefully) easy to follow in the VS notebook. Dang.
- Take a break. Because we're going to do it all again in Python.

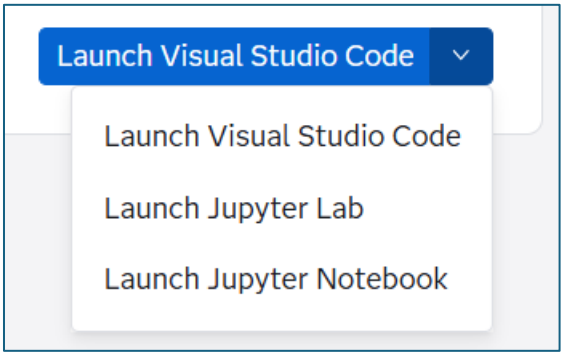
Predictive Modeling using Python Code in JupyterLab

While we could keep running our Notebooks in Visual Studio, I'm going to have you move into JupyterLab because a LOT of academics are more comfortable with that IDE (integrated development environment). Plus it's our default IDE in SAS Viya for Learners. Let's get rolling:

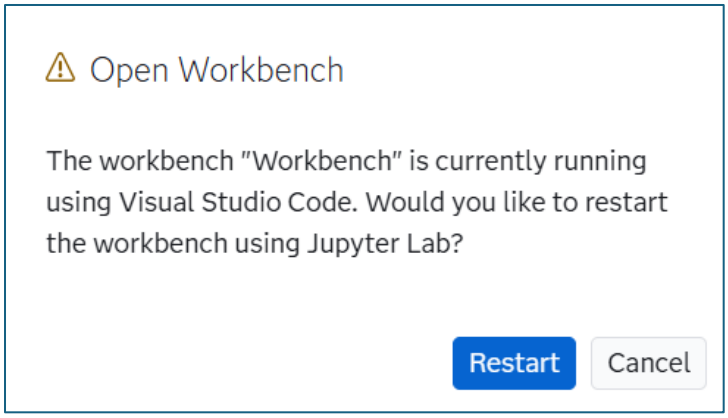
- You're likely still in Visual Studio… because that's where I left you 😊 Close the Visual Studio window and get back to the **SAS Viya Workbench** landing page. And confirm that you session is still **Running**… because running is a good thing. It means the session is still active:



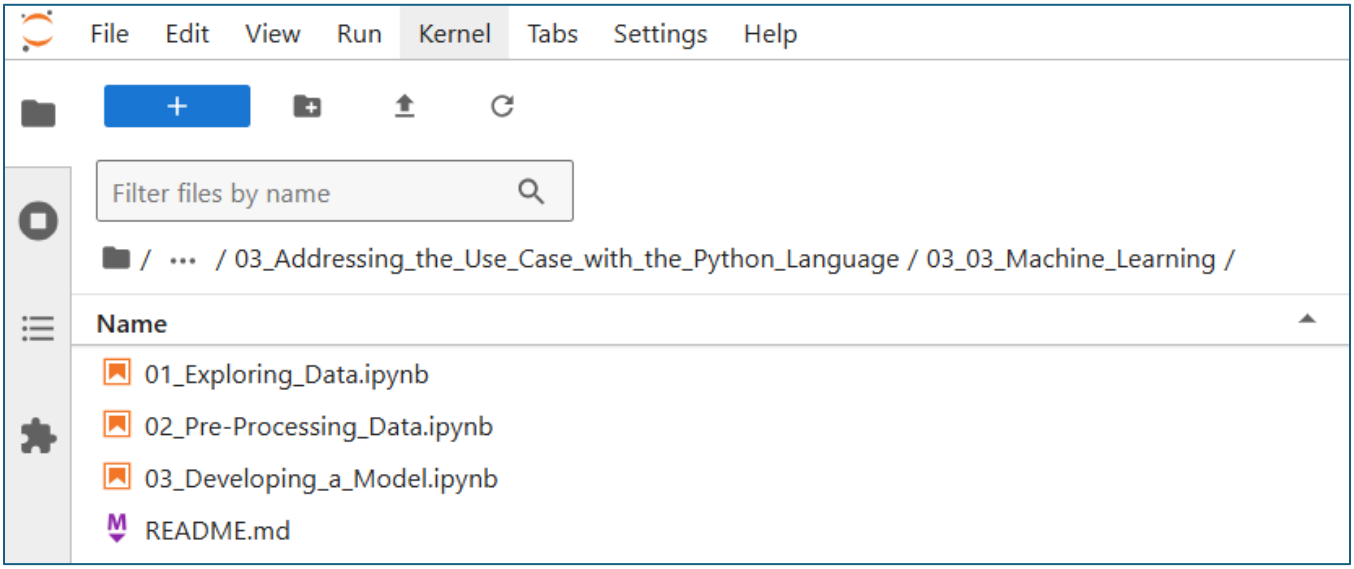
- Now examine the **Editor** options next to the **Launch Visual Studio Code** button:



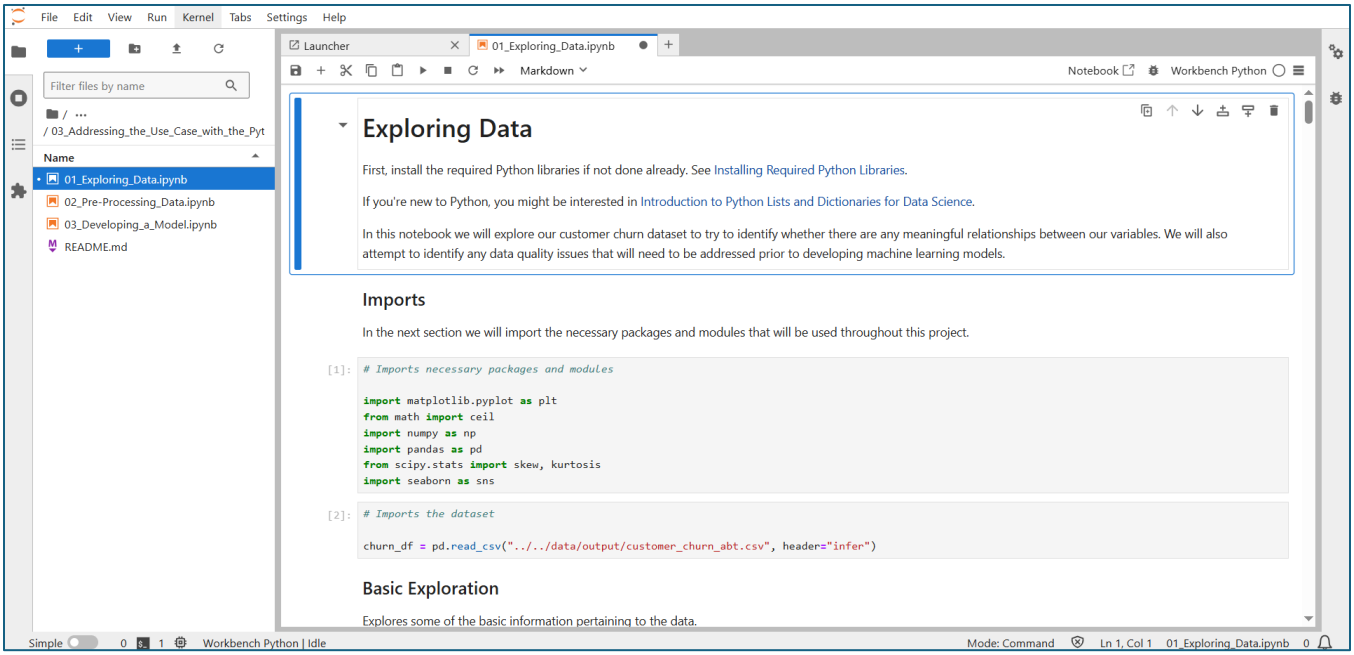
- Lookie what we have there! For those looking for a Jupyter experience closer to the one provided by default in SAS Viya, I recommend using **Jupyter Lab**.
    - And for those of you interested, my good friend ChatGPT has this to say about the difference between Jupyter Notebook and Jupyter Lab:
    - "Jupyter Notebook offers a simple, single-document interface where you work on one notebook at a time, making it ideal for focused tasks. JupyterLab provides a more flexible, IDE-like interface that allows for working on multiple documents and tools simultaneously, with customizable layouts for enhanced productivity."
- So, open **Jupyter Lab** for today, because we've got multiple programs to look at. And you'll likely get this message:



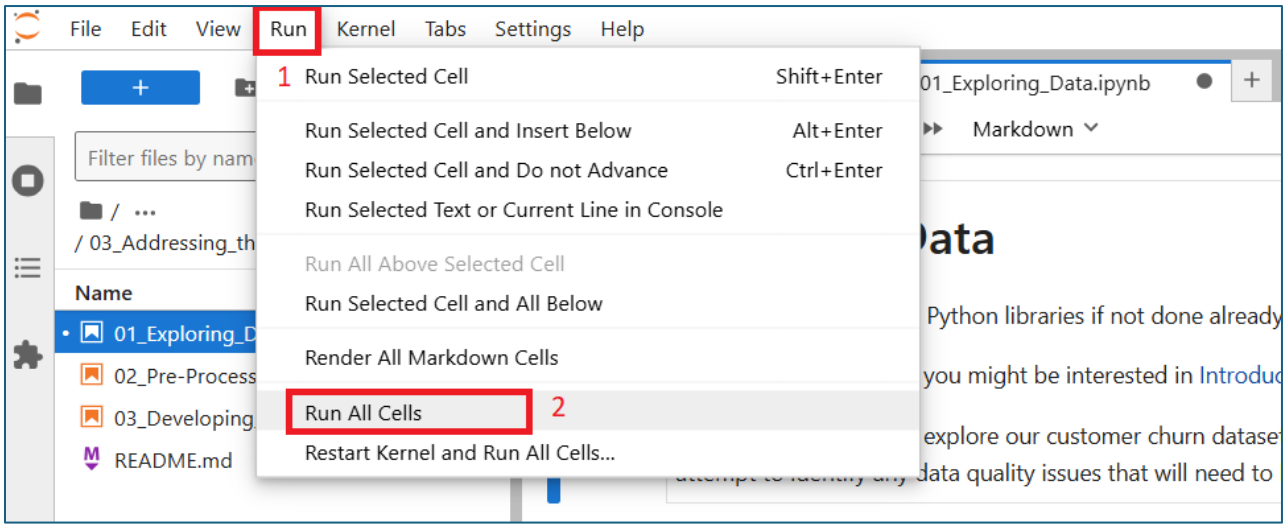
- All good, let's **Restart**! And then Welcome to Jupyter Lab!
- Now we already did the hard work of setting up the data and notebooks in Visual Studio. So, to find the Python programs using the **File Browser**, drill into **sas-education** >> **sas1** >> **03_Addressing_the_Use_Case_with_the_Python_Language >> 03_03_Machine_Learning**. Like so:



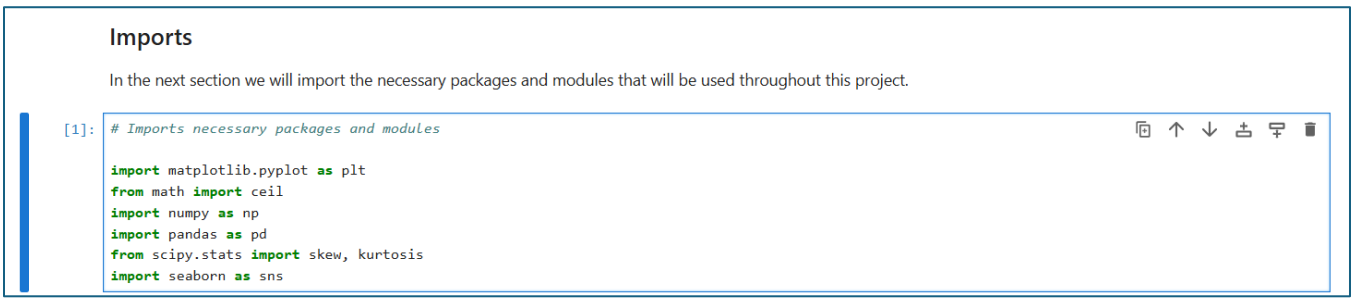
- And the corresponding notebooks await. Nice!
- Even though we're in a second IDE, the actions are the same:
  - Open the notebooks in order;
  - Run them… either as individual cells or as entire programs;
  - Digest as much of this Machine Learning section that you can handle in one sitting.
  - And, finally, more high fives – you're done.
- But, I know that not everyone is comfortable with Jupyter Lab yet. So, lemme guide you through the process for the first notebook. Open ***01_Exploring_Data.ipynb*** with a double-click. The following appears:



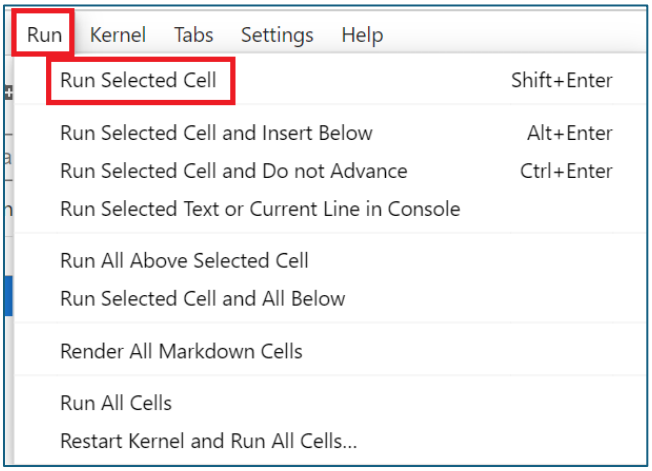
- Want to run everything all at once? Do it with these two clicks:



- Want to savor the moment… and actually digest the output? A couple of options on that front. And they all start with highlighting the cell you want to run, like so:



- You can then go to **Run** >> **Run Selected Cell**:



  - Or… like the helpful guide above notes, you can simply highlight a cell and then click **Shift + Enter**. It's very much your party now… so do whatever feels most comfortable!

- Please run the three programs and try to digest as much output as possible on your first day. And then share any insights you have!
  - Oh – and note that **03_Developing_a_Model.ipynb uses** some autotuning Python procedures – which is why it takes some time to run. Nice! Autotuning is not yet available in our SAS programs in WFL… so this is a nice example of where open source can move a bit faster than the (highly validated) SAS programming language.
- Oh – and finally: