Audio file

Your Recording 25.wav

Transcript

00:00:04 Speaker 2

So what ways could I run Jupiter? I could deploy it locally on my own computer. Now, if I already have Python installed on my machine, I'm likely to have the Python package manager pip, and I could type pip, install jupyter and I would get Jupyter installed. Alternatively, I could use Anaconda. Now Anaconda is an open source distribution used in data science that includes Jupyter and a whole set of data science libraries that are commonly used as well as an Anaconda package manager. But in either case, I would have Python And the Jupyter server running locally on my machine now in order to actually use.

00:00:33 Speaker 2

A Jupiter notebook. I would be doing that via a browser. So in a way the Jupyter server is kind of like a web server and I would need to point my browser back at my own computer to get the Jupyter user experience to get the Jupyter lab interface. Then I could just use the menu to do a file, open and open an ipymb file and run and edit that document. Another way I could do this is with a container image. Maybe on my machine I already run something like docker so I could obtain a ready made image from a docker image repository.

00:01:02 Speaker 2

He is a very common one or docker hub and I could.

00:01:04 Speaker 2

Pool a docker image, a container image from one of those registries, and then do a docker run to launch a container locally in my machine. The advantage there is you've encapsulated Jupyter into this container image and therefore it's not directly installed into the local OS environment of your machine. But yet it would be consumed the same way once you've run your Jupyter lab container. Then again it would be exposed locally so you would connect back to yourself using your browser in order to consume the Jupyter lab user interface. Now if you don't want to install Jupyter server.

00:01:34 Speaker 2

Could I use an IDE? And the short answer is yes. A number of integrated development environments such as Visual Studio Code support the IPYMB Jupyter notebook format and therefore you could open them natively in your existing IDE. Now, different Ides will have different levels of full support for what Jupyter can do. Now Visual Studio Code, it's OK, it doesn't offer me everything, but it might be just enough to perform what I need to do. But just be aware that not every IDE offers the full capability that Jupyter does.

00:02:03 Speaker 2

OK. And many data scientists prefer using Jupiter specifically for data science tasks, but we should be aware that that is an option. And lastly, we've got the opportunity to run Jupyter in the cloud as a hosted managed service, and that's where Sagemaker comes in for our first use case. What we can do with Sagemaker is that we can run a Jupyter or Jupyter lab server environment that is fully managed. And again, because we use our browser to connect to it, instead of pointing our browser at our own workstation name or IP address, we are pointing a browser at somewhere in the cloud.

00:02:32 Speaker 2

You're still seeing the same user interface, but now it is secured and managed in the cloud, and that's maybe where your data set resides. If your tabular data set of million rows of data 200 features wide exists as a set of CSV files in an S3 bucket, then having that Jupiter environment closer to that, and within a secured environment so that only your Jupyter server can access that bucket might be a more secure security posture to have than trying to use some kind of Jupyter server locally in your computer.

00:02:58 Speaker 2

So Sagemaker will provide you with a hosted Jupiter or Jupiter Lab environment so that you can run out of your AWS account. Let's now look at Jupiter in Sagemaker AI. When Sagemaker launched back in 2017, it launched with support for running hosted Jupyter notebooks. In other words, managed service for Jupyter.

00:03:16 Speaker 2

Then in 2019 AWS added support for Jupiter Lab where we could have multiple tabs, therefore multiple files open at the same time.

00:03:25 Speaker 2

Now, even today, if you use Sagemaker, you have the option of opening up a Jupyter notebook, and you get the option of whether you want to use the simple original Jupyter interface or the newer Jupyter Lab interface with multiple tabs.

00:03:37 Speaker 2

Most customers would generally use Jupiter Lab due to the flexibility and the support for having extensions to enrich that IDE.

00:03:45 Speaker 2

Now later in 2019, AWS launched Sagemaker Studio. Now Sagemaker Studio was intended to be a full machine learning focused IDE that was built upon the foundation of Jupyterlab so that you would get Jupyterlab plus access to a range of other tools and services that a data scientist or Mlops engineer would need to perform the tasks of the ML pipeline.

00:04:07 Speaker 2

So you still have that choice today of using the original hosted Jupyter notebooks or Jupyterlab.

00:04:13 Speaker 2

But really, since late 2019 when we want to do Jupiter, most customers have been using Sagemaker Studio because it goes even further in terms of delivering a rich environment with the tools you need for data scientists. So that means that we have two environments in Sagemaker where we can run Jupyter notebooks. We have the original notebook instances. So if you've explored an AWS management console in the Sagemaker AI product, you may have seen this in the left hand side menu of notebook instances. Now, if I create a notebook instance, I'm going to be creating a managed EC2.

00:04:43 Speaker 2

Instance AWS take care of for me, patch and maintain and while it's running I will pay for it. But that notebook instance is running a hosted version of Jupiter. You get to choose if you want to be hosting A Jupyter original notebook interface or the Jupyter Lab interface that gives you multiple tabs. And generally we're going to be using the Jupyter Lab interface because quite simply, it's better and more flexible. But remember from late 2019, AWS made available Sagemaker Studio and Sagemaker Studio. Ah, that gives me a much more rich editing environment for developing my models.

00:05:13 Speaker 2

It yet is built upon Jupiter Lab, but it gives me access to many more tools that will ease my ability to solve the problems of the machine learning pipeline.

00:05:21 Speaker 2

We are going to spend most of our time in this course on using Jupyter Lab in Sagemaker Studio. For the moment, Amazon is still supporting the original notebook instances, but we would not be surprised if at some stage they were marked end of support. So I would encourage you to spend more time in Sagemaker Studio as that's where the future lies. When we run Jupyter in Sagemaker, we need to think about how we consume it versus how it is hosted. Jupyter is a web-based application, so our consumption interface is always going to be the browser.

00:05:48 Speaker 2

We point the browser at the URL of where the application is hosted.

00:05:51 Speaker 2

So if I was running Jupiter locally on my computer, I would be pointing my browser at localhost and I'm running Jupyter lab in the cloud then the URL will be pointing at a cloud based resource. Now in Sagemaker if I want to host Jupyter lab Jupyter server then I need to specify a size of instance that will perform that hosting. Now when I get to create a Jupyter server Jupyter notebook server, I would specify size and the sizing looks similar to what you may have seen if you've used AWS EC2 service before where we have this idea of instance families generations.

00:06:21 Speaker 2

And T-shirt sizes. Now I can see here I've got something called MLM five large. Now what that means is there is a size of virtual machine in the machine learning family. The first ML stands for machine learning family, then M5. Well the M means kind of main application. It's a general purpose instance. But if I had more compute intensive tasks to do, I might use a Cfamily. I might use a C5 instance rather than an M5. Or if I wanted accelerated computing that included like an NVIDIA GPU, I might use a key instance like AP 3. So you can investigate what options there are.

00:06:51 Speaker 2

For what different instance sizes there are and what instance families depending on what you're trying to do. The number 5 just indicates a generation. So an M5 will run a newer hardware than an M4. An M6 instance will run a newer hardware than an M5. It's just a way that you can control where your instance ends up running. Does it end up running on a brand new server with the latest CPUs or does it run in maybe a previous generation? The last part that says dot large, we call that the T-shirt size. If you walk into a clothing store to buy a T-shirt, it will come in different sizes, Small, medium, large, extra large, and that last piece of the name there.

00:07:22 Speaker 2

Indicate size. So large might mean 2 CPUs and 4 gigabytes of RAM, but 2X large would be 4 CPUs, 8 gigabytes of RAM. So you have these different sizes like large, 2X large, 4X large, 8X large, and different families like the M family or the C family might have different size ranges that they support all the way up to. I think 24X or 48X large is some of the largest instance sizes, but that's not available in every single family. My point here is that when you're choosing an instance, in other words, a managed virtual machine to run your Jupyter.

00:07:52 Speaker 2

Server you get to decide the resources that are allocated to it. Now, that would mean that I could have multiple Jupyter Lab instances. I could have multiple environments and each one allocated sufficient resource to provide what that particular project needs. So if I was running traditional Jupyter Notebooks and Sagemaker, I might have one Jupyter notebook server that's an M5 large for project one. But then I might have another notebook instance that is running on an M5 extra large because it's going to be running Jupyter Lab rather than the original Jupyter server.

00:08:22 Speaker 2

And is going to have maybe more tabs open at the same time. So we need to understand that you can have more than one hosted environment in Sagemaker AI. Each one is defined by you and given the CPU and memory resources that you deem appropriate for the project use case. Remember that we are using Jupyter not just so that we can run code cells and capture their standard output, but we can also capture visualizations. Now visualization libraries such as seaborn and matplotlib allow us to produce very rich visualizations that help the.

00:08:52 Speaker 2

Scientist for example here I'm looking at correlation heat map. Very important when I'm doing linear regression that I can understand if I've got positive and negative correlation between numeric features. For example, if one feature is increasing in value, does that mean another feature also

increases in line with it? Or as one numeric value increases, does another numeric value decrease by the same amount? Are those features highly correlated with one another? If they are, I probably don't want to go forward with both of them. I would just pick one of them. So being able to visualize that in a correlation heat map can help me quickly identify.

00:09:22 Speaker 2

Which features I maybe need to decide to keep and which to drop. I can also look at the distribution of my data. Here I'm looking at the distribution of house size versus price and start to get a better idea of the scatter of values. And that's again going to help me understand how well distributed my data is. It's going to help me understand outliers. It's going to help me understand my data distribution far, far better. So these visualizations are produced in code, but there's standard output. The actual rendering of it is done directly in line to the Jupyter notebook. And therefore, if I save the notebook and saving those visualizations for.

00:09:52 Speaker 2

Else to inspect. By using Jupyter notebooks, I get some great results. I get inline visualizations. I've got that ability to get instant feedback after running a code cell, and I'm able to leverage those rich visualizations from seaborn, matplotlib, or indeed any other visualization library. I'm able to integrate with many different cloud environments. Jupyter is not specific to AWS. Jupyter is open source and as we've seen, we could run that locally or we could run it in cloud and it will work with Google Colab, it will work with Sagemaker, it will work with Databricks. It is a well known industry standard way of running.

00:10:22 Speaker 2

Science projects, and crucially it's self documenting because we are combining our code with markdown. Cells are describing what we're doing, why we're doing it, the hypothesis of our experimentation, and then we can see the results in line saved as one document. Then I can simply store that document in a git compatible repository and then another data scientist in my team can open it and review it.

00:10:42 Speaker 2

Using Jupyter Notebooks is absolutely the industry standard for data science projects. Some of the industry leaders, such as Airbnb, use Jupyter Notebooks in their ML projects around optimizing pricing and search ranking fraud detection. NASA users in terms of analyzing large datasets including that of the Mars Rover and Hubble Telescope. And Netflix also use Jupyter Notebooks for data exploration hypothesis testing. If the industry leaders are using this for their data science, we can take confidence from that that it will be good for our data projects too. So what have we seen in this lesson? We've seen that Jupyter Notebooks can be.

00:11:12 Speaker 2

Adopt by Jupyter Lab and Jupyterlab server can run either locally in your computer, maybe installed via Piper Anaconda, or can run remotely in the cloud environment as is the case with Sagemaker AI. Your user interface to Jupyter is always via a browser. It is a web application. We have seen that

within a Jupyter notebook we can have cells. They are either a code cell or a markdown cell. Code cells for Python code markdown for annotation documentation.

00:11:37 Speaker 2

Remember, when we run a cord sale, it's standard output will be captured in the document. Also, we have seen that Jupiter Lab as different from Jupiter Server means that it's a newer interface that supports multiple tabs and supports plugin for extensibility. So generally we're mostly using Jupyter Lab nowadays. We've seen that Sagemaker provides hosted version of Jupyter Notebook and Jupyter Lab. You've seen that we originally launched Sagemaker with these notebook instances.

00:12:04 Speaker 2

But late 2019, that's when we launched this Sagemaker Studio environment that's built on Jupyterlab but provides a raft of new tools that will support a data scientist in their ML project.

00:12:14 Speaker 2

So remember, when you first start looking at Sagemaker, there are two different ways we can run Jupyter, either as a notebook instance, which we would say is that legacy way of doing things, or in the new Sagemaker Studio. And that's most definitely the way we would encourage you to run your projects in the Sagemaker Studio. In our next lesson, we're going to look at Jupyter Lab in real life in our demo on how to use Jupyter. See you there.