Audio file

Your Recording 17.wav

Transcript

00:00:06 Speaker 2

Let's now consider the data scientist. We know that the data scientist will start with data exploration. They want to get to know their data, to understand it, to understand the features, how they impact the target variable. They want to understand if there are correlations in that data, and they want to understand if there are unnecessary features.

00:00:24 Speaker 2

So the data scientist is going to want to analyze that data set. They're typically going to be using a Jupyter notebook at this point. Remember, Jupyter is our interactive Python environment that allows us to run a few lines of Python at a time, examine its output, and annotate what we're doing using Markdown as we go. And this really helps us kind of formulate experiments and show how we can repeat our work. But definitely the data scientist wants to explore the data set. They'll want to visualize that data. Now, typically they want to load that data into a pandas data frame. Remember, we saw pandas.

00:00:54 Speaker 2

When we're looking at the math module and we saw that that was a great Python package, we could import and then manipulate multidimensional data. Once it's in a pandas data frame, though, you have this range of other packages that we could choose to import that will help us visualize that data.

00:01:09 Speaker 2

The data scientists trying to find the useful features. The data scientists could be faced with a tabular data set that's maybe 200 columns wide. It's 200 features too many, quite possibly. So which ones actually influence the target variable?

00:01:22 Speaker 2

Ones that don't, let's drop them because if I've got too many features, it might become too much for the model to try and workout the patterns and then the model during the training process might not convert.

00:01:32 Speaker 2

So the data scientist working in a Jupyter notebook, working in a code first approach using Python is going to want to use handas #1 get it into a data frame and then we can start to manipulate it. We can use native features of pandas native methods, but we can also use numpy for example if we want to do particular mathematical operations.

00:01:50 Speaker 2

So numpy is a great tool and I think we used it earlier for clipping values when we were looking at our outliers.

00:01:56 Speaker 2

If I want to visualize data, matplotlib is a superb package. I can utilize and I can do all sorts of visualizations with matplotlib. I can look at what we call a confusion matrix, or I can simply look at a distribution chart. For particular input variables, I can look at correlation charts.

00:02:11 Speaker 2

And I'm likely to want to import scikit learn. Remember, scikit learn is like a Swiss army knife of tools and methods that are commonly used during machine learning. Now, scikit learn has sufficient methods in there to actually do training itself, but we often use it for gaining access to features like data imputation tooling or tooling to perform scaling. Remember we talked about min Max scaling, We talked about standardization and how we could utilize those features. Well, they are all just methods in scikit learn. We don't need to know the formulas, we simply need to know which feature.

00:02:41 Speaker 2

You're applying that scaling.

00:02:43 Speaker 2

Once the data scientist understands their data, they can start the process of feature engineering.

00:02:49 Speaker 2

Now, feature engineering sounds a little scary, but really all we're doing is making sure that we only go forward into the training process with those data features that actually contribute something meaningful towards a target variable.

00:03:01 Speaker 2

If we have an input feature that has no influence whatsoever on the target variable, then let's drop the feature. That's a simplest form of feature engineering, selecting the relevant features so that we don't have too much dimensionality, too many features. That would mean that the training process would never converge. It would never be able to generalize against that training data set.

00:03:20 Speaker 2

But there's more to it than that because there might be changes to the data we might need to make. For example, the data might need to be transformed in a way that the model expects. Now that could mean like categorical data, text data. The model works on numerical data. So if I want that to have an influence, then I probably want to use some kind of technique like 1 hot encoding to turn the categorical data into numerical data. I can then drop the categorical feature and just go forward with the new numerical features that the algorithm in the training process can actually use.

00:03:48 Speaker 2

So this is where the data scientist earns their money by understanding the data, by understanding what transformations are needed so that the algorithm that they've selected to solve the problem will be able to make best use of that data.

00:04:02 Speaker 2

The data scientist can then trigger the training job. Now I say training job, but it is likely that the data scientist will conduct a number of training experiments. So they might be considering, well, I'm going to think about using the XG Boost algorithm for this, but I'm also going to try the LGBM algorithm as well and see which one gives me the better results. So as the data scientist has determined their data is in the right shape, they then determine which algorithm and which algorithm versions they're going to attempt to train their model on.

00:04:29 Speaker 2

They will choose an appropriate algorithm for the problem that they are trying to solve.

00:04:33 Speaker 2

Then the data scientist will invoke a number of training processes. Now, for each training process, using different algorithms or different versions, they'll get a model artifact that they can evaluate against the data that we held back from our training set. If you remember, we talked about this in a previous lesson, that if I had 100,000 rows of data that I could train with, I would only train my model on 70% of that data set and I would hold back 30 of it.

00:04:56 Speaker 2

Maybe 20 for evaluation and 10 for a final Test so that I could present data to my model and measure its performance. Because in your training data set, you have the target variable so you can compare what the model produced versus what the actual ground truth was in the source data set. So the data scientist will typically use the Sagemaker SDK to invoke a number of model training processes. Now each time you train a model, you have a number of levers, a number of controls. I like to think of this like in a recording studio, you might see an audio mixing desk with.

00:05:26 Speaker 2

You know, 40 different sliders were adjusting different channels and compression and all sorts of different settings. I kind of think of a data scientist like that, adjusting all of those sliding values to say, OK, I'm going to increase this one and decrease that one. How does it sound? I know it's not a great analogy, but if you imagine you as a data scientist, similar to an audio mixer, adjusting the different sliders, you are adjusting the hyperparameters of the model that will adjust how quickly it's able to converge. How big a step can it make? How many times does it iterate over the data? These are controls in the data scientists realm.

00:05:56 Speaker 2

So again, the data scientist might choose to maybe pick one algorithm, but they're going to run a training job with one set of hyperparameters. And then they might run the same job again, but with a slightly different hyperparameters, and they're tuning their model to get the best possible result from it.

00:06:10 Speaker 2

So this is an iterative process. When they look at the results, they get got poor accuracy on that. I wonder why it looks like the model didn't converge. OK, I'll go back and adjust the learning rate or the number of times I iterate over the data in the next training job and then I can compare and contrast the accuracy results of my model after I've adjusted the values.

00:06:30 Speaker 2

So how does the data scientists activities map to Sagemaker features?

00:06:34 Speaker 2

While we know the data scientist is going to need an interactive Python environment in order to perform the feature engineering and training activities that they are responsible for. Now Sagemaker Studio is a hosted Jupyter Lab environment. So in other words, we host a Jupyter Lab server that you can connect to through your browser. This is launched from the AWS Management console and opens up as a new browser tab. So that's our first feature. Now we also might make use of Sagemaker Jumpstart. This would be if the data scientists rather than completely.

00:07:04 Speaker 2

Training a new model from scratch wants to leverage pre trained models. We call these foundation models and if they already exist as they do in Sagemaker product, we are able to use Jumpstart to say well I want to use that readily trained model and I'm going to import it and fine tune it within my Jupyter lab environment. Then jump start enables me to do that.

00:07:23 Speaker 2

But the data scientist, as they train their model, they are going to want some compute platform to do that.

00:07:29 Speaker 2

Now, when I worked with data scientists who only worked on laptops, they were training their models using the CPU and memory capabilities of their laptops. So that was very restrictive in terms of the data set sizes they could work with and the length of time it took to generate a model.

00:07:42 Speaker 2

If I use Sagemaker training jobs, then although I invoke the training job in Python from my Jupiter notebook, it's not the CPU and memory of the Jupiter notebook that will be used. Instead, we delegate that and it spins up another instance, A managed Sagemaker instance, that has whatever CPU and memory requirement that you want, however many GPUs you want, and that delegated job will run and produce the model artifact for you in a more timely way.

00:08:07 Speaker 2

Now we know that the data scientists expertise will be about selecting the features, choosing the right algorithm for the job, and selecting the right combination of hyperparameters. I mentioned a moment ago about hyperparameters being the fine-tuned controls that we have, like a audio engineer at a mixing desk. But there is also automation available to us. So we're going to look later on at something called hyperparameter optimization and this is going to allow the data scientist to lean upon a Sagemaker feature to do some of their job for them and find the best.

00:08:36 Speaker 2

Hyperparameter permutations available to me. You find it out for me and then I'll train the model.

00:08:42 Speaker 2

And lastly, the data scientist is likely to be making use of the Sagemaker debugger so that we can profile our training jobs and get a better feedback on understanding what's happening during training so that we can come up with better trained models.

00:08:54 Speaker 2

Let's consider the key responsibilities of our personas.

00:08:59 Speaker 2

So we have our three personas. What are the primary focus areas? We know data engineer is focused on that data extraction and transformation from the data source, the golden source for our data project. And in large organizations, we want that to be a repeatable process. Mlops engineers are all about getting the model productionized, get the model for development out into production so you're actually generating predictions from it.

00:09:22 Speaker 2

The data scientist is responsible for that feature engineering and model development.

00:09:28 Speaker 2

The output of each phase will The data engineer's output should be clean, accessible data ready for exploration and feature engineering. So remember, the data engineer may have started with structured data in a database that they needed to extract and transform into CSV format. For us, the Mlops engineer's output would be production-ready models that are hosted ready to receive their first inference request.

00:09:48 Speaker 2

And our data scientist output is going to be based on their analytical insights and the training process of training the best model possible given the data selected algorithm and feature engineering process.

00:10:00 Speaker 2

The dig engineer is going to be responsible for working with the data scientists. There is most definitely a handoff between the data engineer and the scientist so that the scientist gets the data in the format they need. The ML OPS engineer is going to collaborate with data engineers and scientists. Just in terms of thinking about the authoring of automation, are we going to be authoring Sagemaker pipelines to be automatically triggered based on committal of git code? So if the data engineer is written transformation code or the data scientist is written training code as they commit them to get repositories?

00:10:29 Speaker 2

Then what then happens? The Mlops engineer would typically want to link those git commit actions to their pipelines.

00:10:36 Speaker 2

The data scientist is going to work not only with the Mlops engineer and the data engineer, but will also have a face off to the business and typically a subject matter expert in the data domain that we are dealing with. So that's particularly during that exploratory data analysis period, we can better understand the features and what they actually mean in the context of the business problem ML is trying to solve.

00:10:57 Speaker 2

And the deliverables that we would expect from each persona or with the data engineer is going to be a data pipeline, something that is repeatable and is new data come in, it can be extracted, transformed and made available for training. For the Mlops engineer, it's going to be automation, typically Sagemaker pipelines that perform a sequence of actions from a common trigger.

00:11:15 Speaker 2

And for the data scientists, their deliverables will be the models that they have trained.

00:11:20 Speaker 2

So what have we seen in this lesson? We've seen that the data engineer is responsible for the data source extraction, transformation, and delivery. To the data scientist, that might mean extracting from a relational database source, and it might mean further filling in the values that are missing or other transformations. If the data format was maybe in Jason format and needed to be converted to CSV, that would very much be with a data engineer.

00:11:43 Speaker 2

Remember, in larger organizations a data engineer will not just be thinking about this as a one time activity, but instead about creating a repeatable data extraction and transform and load pipeline.

00:11:55 Speaker 2

The data scientists we know are going to do that exploratory data analysis. They're going to get to know which features have an impact on the target variable and drop the ones that don't. They might also need to transform the data further to generate new features or to change how that feature is presented, like categorical data into numerical data. They're going to choose the hyperparameters, those fine grained controls that we have on the training process, and they might train a model over and over and over using different hyperparameters to get the best results.

00:12:23 Speaker 2

The Mlops engineer we know is focused on building automation. Automation that empowers the entire ML pipeline so that when we as a data scientist commit our code into a git compatible repository, that triggers a process, that triggers the training, that triggers the registration of the model artifact into model registry. When a model approver approves a model for production, the automation that will be triggered by that approval step should then deploy the model.

00:12:49 Speaker 2

The Mlops engineer will use tools such as Sagemaker pipelines.

00:12:53 Speaker 2

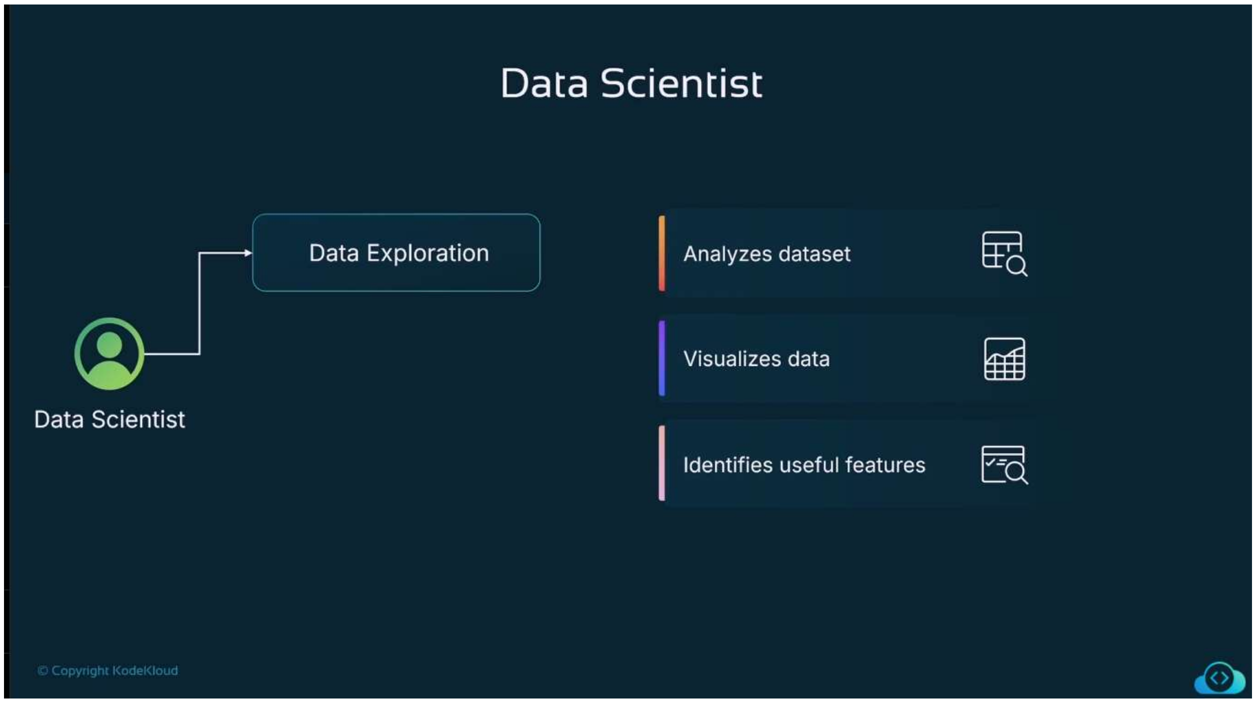
To build those sequences of automation.

00:12:56 Speaker 2

So the Sagemaker components that we've talked about so far, we've seen how they align not only to specific steps in the ML pipeline, but to the specific personas who will make use of those features.

00:13:07 Speaker 2

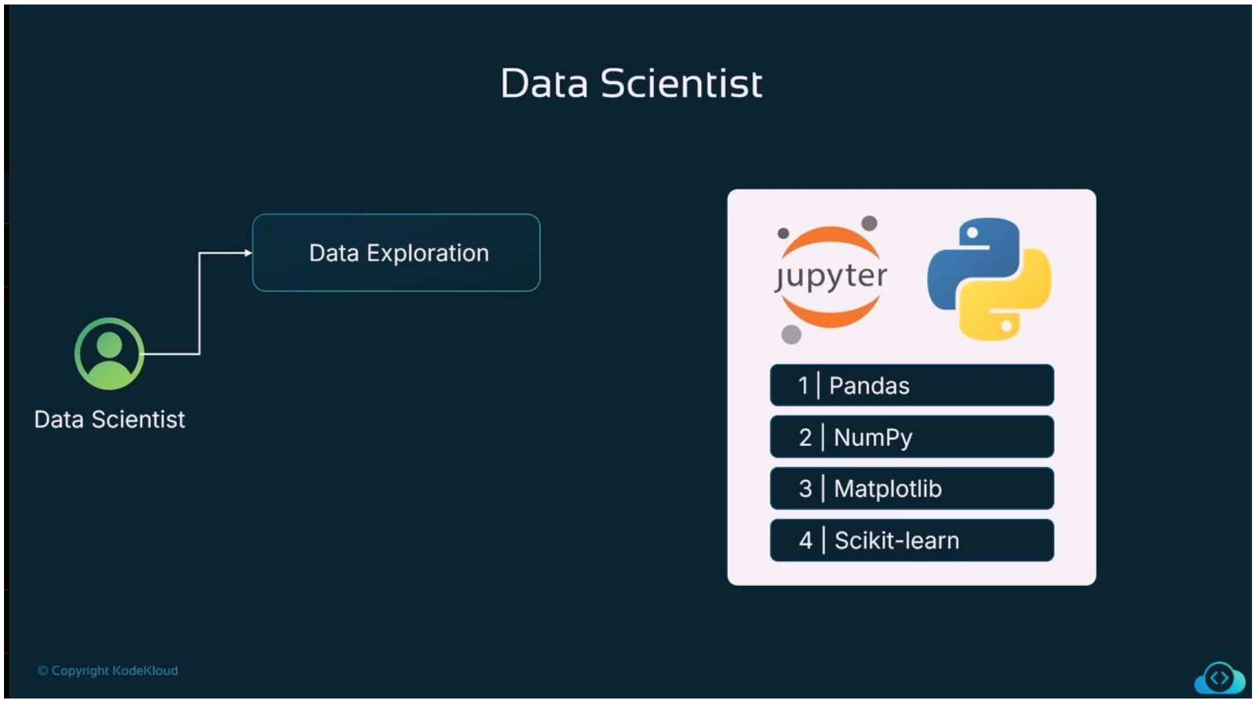
So with that, we're now better understanding what Sagemaker has to offer us. So now we have a better understanding of which Sagemaker features will be used by which persona. In our next lesson, we're going to see what a managed service is because Sagemaker is a managed service. See you there.



Data Scientist

Data Scientist → Data Exploration

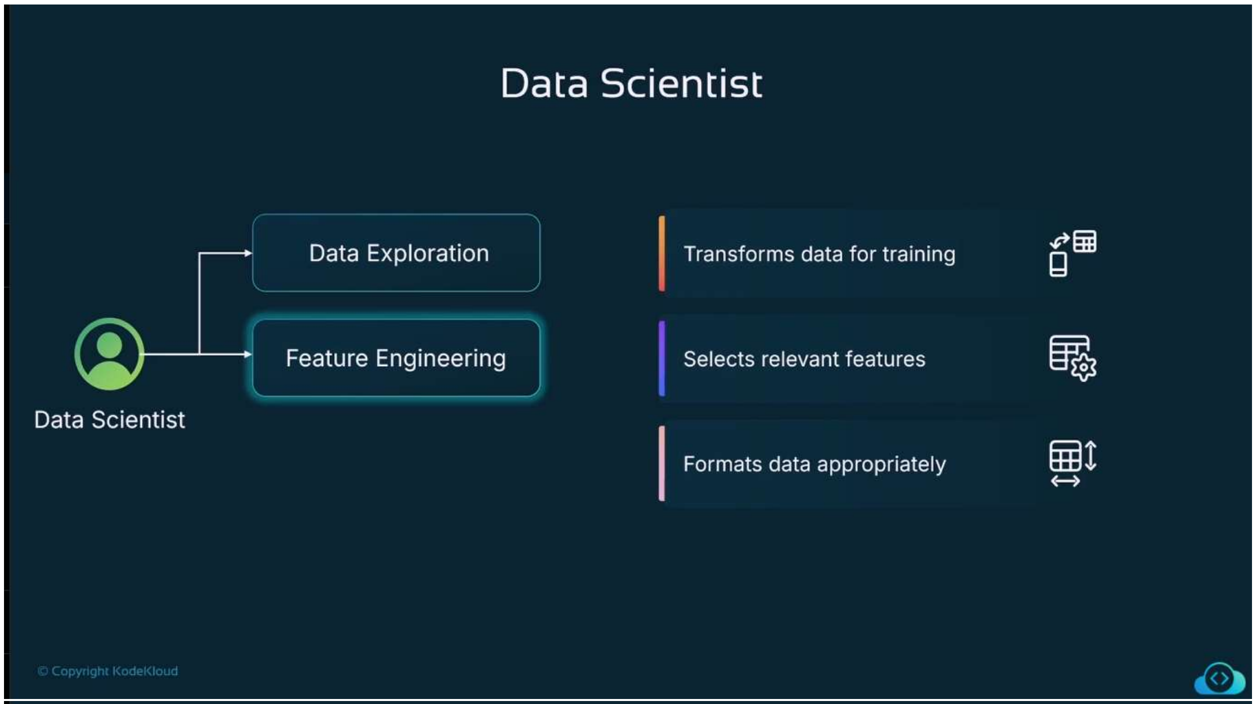
- Analyzes dataset
- Visualizes data
- Identifies useful features



Data Scientist

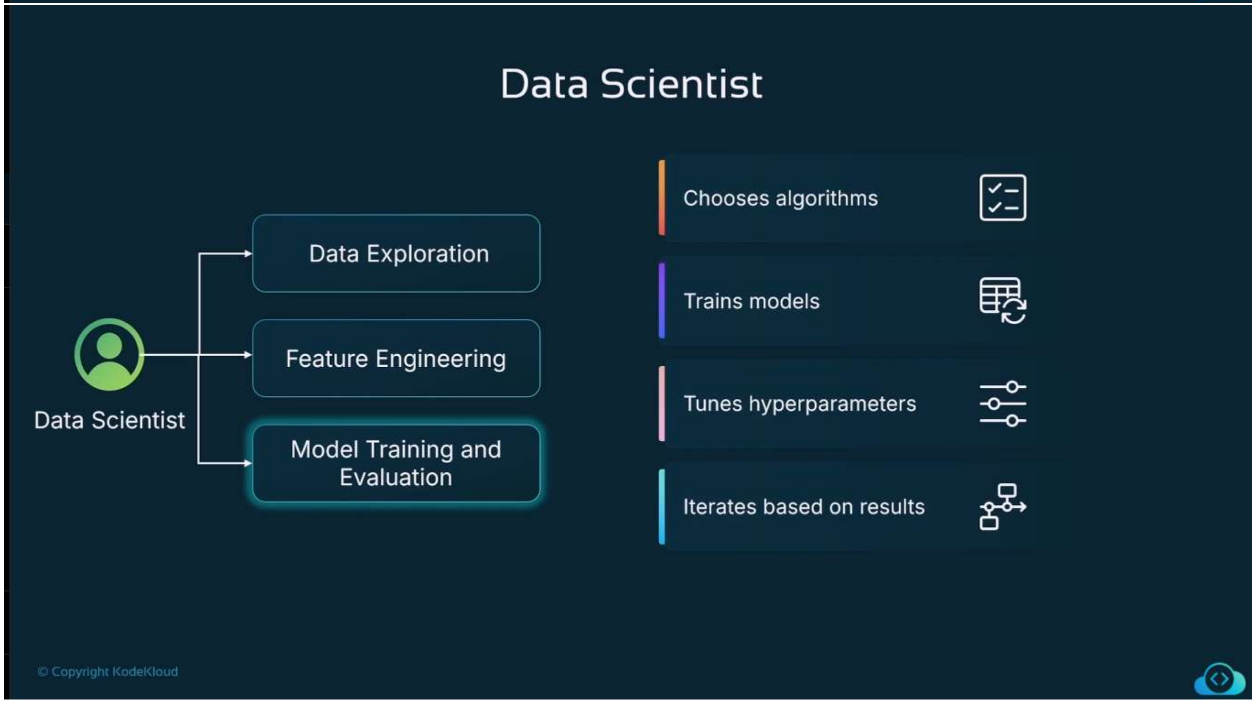
Data Scientist → Data Exploration

1 | Pandas
2 | NumPy
3 | Matplotlib
4 | Scikit-learn

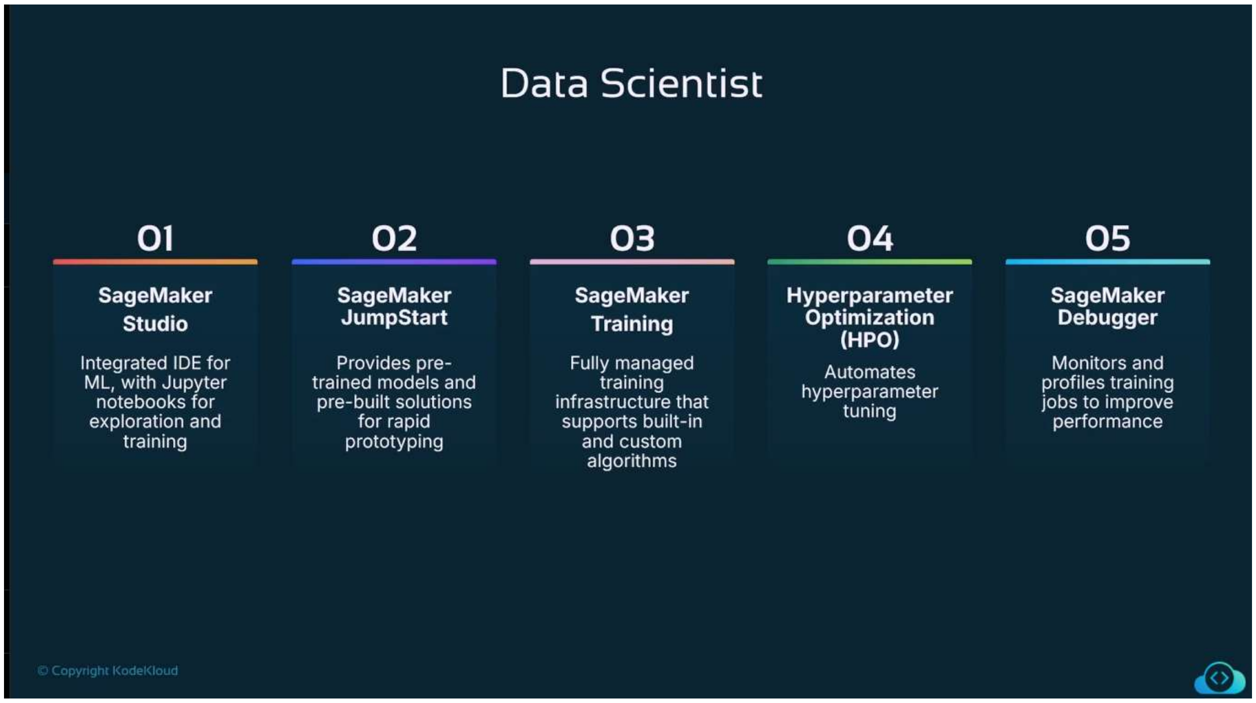


Data Scientist

Data Scientist
Data Exploration
Feature Engineering
Transforms data for training
Selects relevant features
Formats data appropriately



Data Scientist



Data Scientist

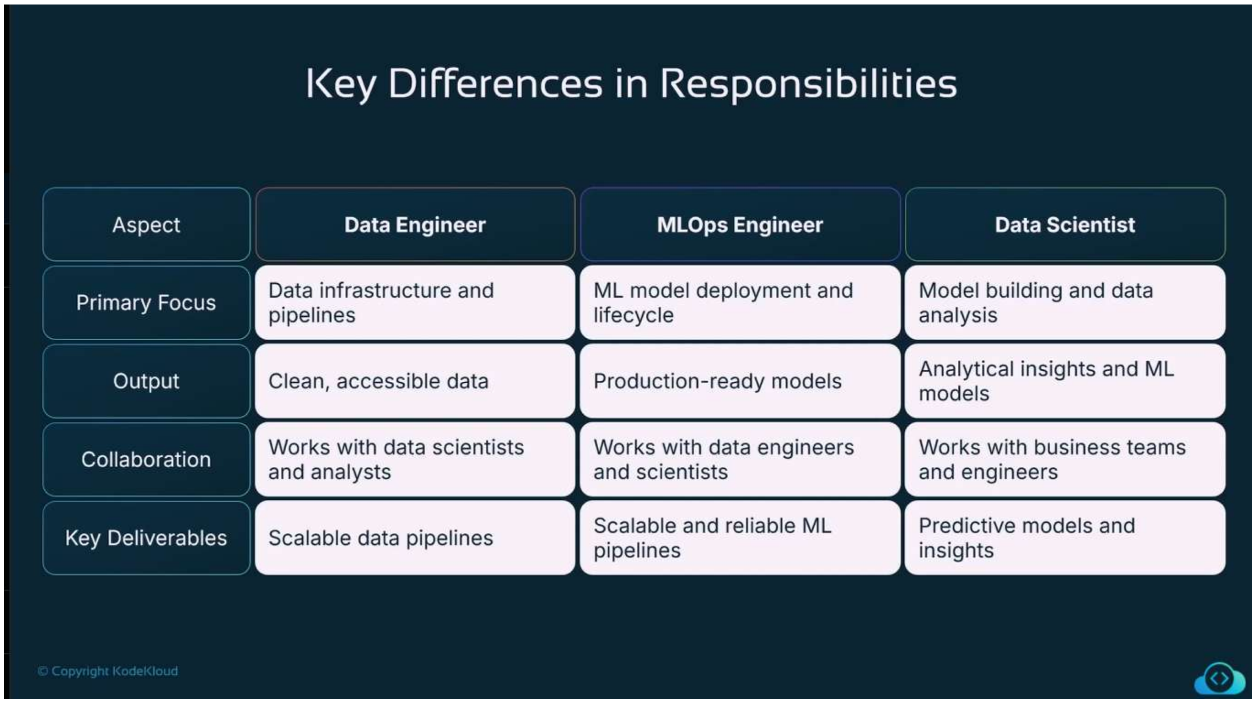
01
SageMaker Studio
Integrated IDE for ML, with Jupyter notebooks for exploration and training

02
SageMaker JumpStart
Provides pre-trained models and pre-built solutions for rapid prototyping

03
SageMaker Training
Fully managed training infrastructure that supports built-in and custom algorithms

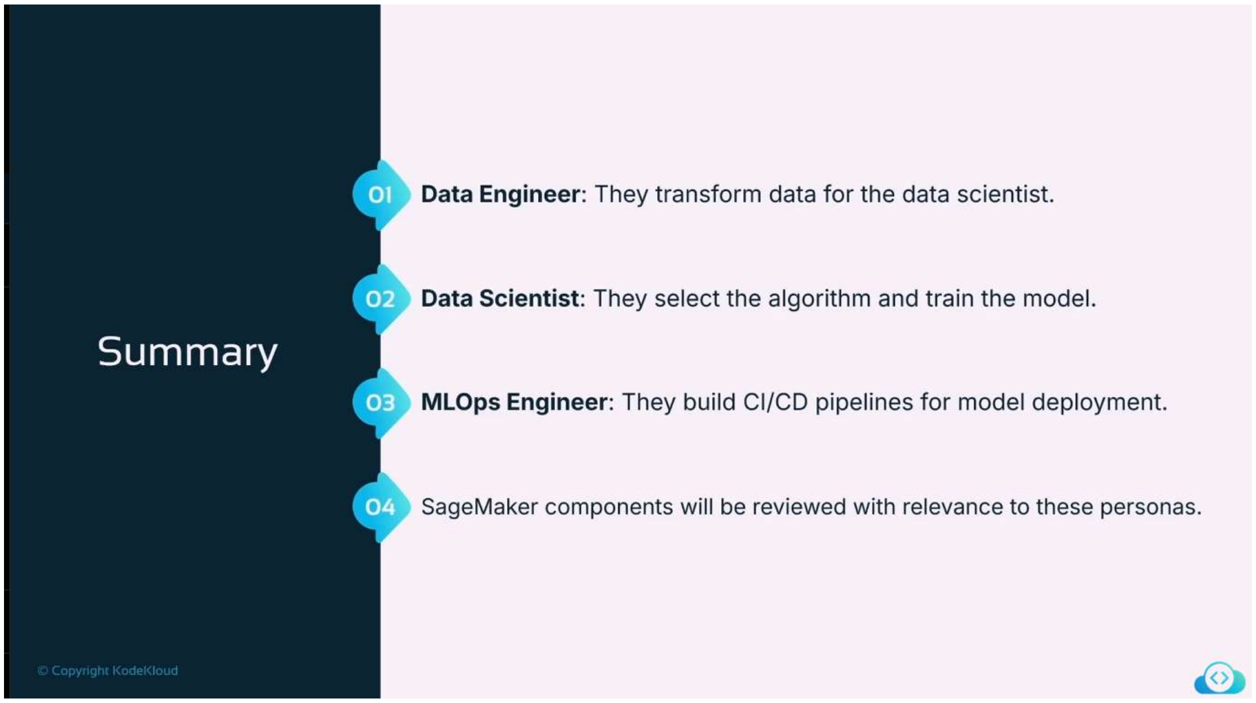
04
Hyperparameter Optimization (HPO)
Automates hyperparameter tuning

05
SageMaker Debugger
Monitors and profiles training jobs to improve performance



Key Differences in Responsibilities

Aspect	**Data Engineer**	**MLOps Engineer**	**Data Scientist**
Primary Focus	Data infrastructure and pipelines	ML model deployment and lifecycle	Model building and data analysis
Output	Clean, accessible data	Production-ready models	Analytical insights and ML models
Collaboration	Works with data scientists and analysts	Works with data engineers and scientists	Works with business teams and engineers
Key Deliverables	Scalable data pipelines	Scalable and reliable ML pipelines	Predictive models and insights



Summary

01 **Data Engineer**: They transform data for the data scientist.

02 **Data Scientist**: They select the algorithm and train the model.

03 **MLOps Engineer**: They build CI/CD pipelines for model deployment.

04 SageMaker components will be reviewed with relevance to these personas.