Week 3

#Data Science/2-Data Science Tools#

Watson Studio

What is IBM Watson Studio?

Every business wants to work smarter, and to do that you need to tap into your company's greatest resource, your data. But extracting the full value out of your data isn't always an easy process. First, you end up juggling an incredibly large and complex collection of tools that are used for finding and cleaning data, analyzing and generating visualizations of that data, and using the data to build and deploy machine learning models. And to make matters worse these tools are often a time drain to individually manage, and can be difficult to integrate into your system, which can really slow down the workflow.
Using Watson Studio you can simplify your data projects with a streamlined process, that allows you to extract value and insights from your data to help your business get smarter, faster. It delivers an easy-to-use collaborative data science and machine learning environment for building and training models, preparing and analyzing data, and sharing insights, all in one place.
Watson Studios easy to create visualizations and drag-and-drop code put the power of database decision-making into the hands of any member of your organization with no need for IT assistance. And if you need access to open source tools, the environment offers some of the most popular and powerful ones available. Watson Studio single environment also creates a **workflow** that's incredibly efficient so data scientists can share assets and work to solve problems within the system rather than starting from scratch every time a new issue arises. And developers can use that efficiency to quickly dive into **building machine learning** and **deep learning algorithms**. In fact, in the area of deep learning, Watson Studio supports some of the most popular frameworks and can deploy that deep learning on to the latest GPUs to help accelerate modeling by making it easier to use. The environments built-in **neural network modeler** also helps you build models with a simplified graphical interface even if you don't have the dedicated resources to build a model from scratch, Watson's Studio can help you get started with modeling templates for areas such as visual recognition, language classification, and other tools from IBM Watson services.
Because Watson Studio is seamlessly integrated with the IBM **Watson Knowledge Catalog**, an intelligent asset discovery tool, you can transform data and models into trusted enterprise resources and collaborate with confidence, without compromising **compliance**, security or access control.
Watson Studio provides many benefits for organizations helping to infuse **AI** into the business

and drive innovation. You can train Watson Studio with embedded AI services including watson visual recognition. You can customize your models and deploy them as APIs or Core ML by using open source tools like Jupyter, Notebook, Anaconda and RStudio.
Watson Studio supports most popular code libraries as well as no code visual modeling with neural network modeler for designing neural architectures using the most popular deep learning frameworks. In Watson Studio you can interactively **discover**, **cleanse**, and **transform** your data using **data refinery**. It helps you understand the quality and distribution of your data with built-in charts and statistics, and provides visualized results through interactive dashboards. Watson Studio includes an intuitive drag-and-drop interface that enables a non programmer to speed up the bottle building process by visually selecting, configuring, designing and auto coding neural networks. From development and training to production and evaluation, Watson Studio tracks your models over time to ensure you have the best performance for any given task using the best solutions across the entire lifecycle of your machine learning models.

Watson Studio Introduction

Watson Studio is an integrated platform of tools, services, and data that helps companies accelerate their shift to become data-driven organizations. You can start with a free account to explore its capabilities. Data science is a team sport; we have different types of people interested in the insights that data science can provide. This includes business analysts, data engineers, data stewards, data scientists, and developers. Data needs to be located and cleansed, models have to be created, tested, monitored, and updated. All this requires teamwork.
For this reason Watson Studio was built as a collaborative platform a community of like-minded people. There is a lot to cover in this introduction and will only scratch the surface. You can find more information on the digital technical engagement site at ibm.com/demos. Once you are logged in you may see the Get Started Welcome screen. You can minimize the screen by clicking on the get started button in the upper right.
One important item that is easy to miss is the hamburger button in the upper left it gives you direct access to projects, catalogs, and services among other things.

- The **gallery** is particularly interesting. It is a collection of assets including tutorials, notebooks, data sets, articles, and papers from multiple sources. New assets are constantly added. Assets can be searched using filters for type, language, technology, topics and so on. The results can be sorted by features or by date.
- **Manage** gives you quick access to specific areas to manage-- finally we have integrated support and documentation in the Watson environment.

As mentioned earlier, the **project** is the center of the collaboration. It is very simple to create a project. You click "create a project" in the welcome screen or "new project" and either create an empty project or one from an existing one. Then you give it a name, possibly add a description,

and you're ready to go. At the project level we also have a menu of options. It starts with the **overview** where you can see basic information on the project. This tab also includes a README section where you can get more details on what the project is about. The next one is **assets** where you can see the data assets, models, notebooks, and other assets that are part of the project. You can go to add specific assets using the "Add to project" drop down menu at the top of the screen. We won't go into all of those menu items but one important one to know is "**connection**." This allows you to access data that comes from outside Watson Studio as you can see it includes a lot of data services from IBM but also quite a few from third parties such as Amazon and Microsoft. Going back to our project I'd like to point out the **environment** section. One important tool for that exploration, data manipulation, and model creation is the **notebook**. Depending on the amount of work that needs to be done we have a choice of resource allocation. We can also tailor the environment to include additional libraries so we have a complete environment from the start. I want to point out two more selections from the top menu: "**Access Control**" and "**Settings**." The access control allows you to control collaborators and their permissions and more. In the settings section you can among other things, add services. For example you click on the "Add service" drop-down menu, select "Watson" and add a "Machine-learning" service. You have the choice to add an existing service you may have created earlier in another project or create a new one. Note that most services include a light free version. This means that you can experiment with all sorts of capabilities for free.

Jupyter Notebook in Watson Studio - Part 1

This video covers the basics for working with Jupiter notebooks in Watson studio start in a Watson studio project and add to the project a notebook just provide a name in a description and create the notebook. Let's first load a file so you have some data to work with from the files slide-out panel browse to select the file after the file is added to the project its available to work with in this notebook just click insert to code and insert a panda's data frame for running the notebook.
It's a best practice to insert a cell at the top to describe what the notebook does change the cell type to **markdown** so this cell will not be treated as code and then add the description now you're ready to run the notebook the inserted code loads the data set into a data frame using your credentials for your cloud object storage instance and then displays the first five rows of the data set before returning to the project **save** the notebook.
In the assets tab you'll find the notebook if you open the notebook it will be in read-only mode but you can edit the notebook and make changes for example you can access the info panel and change the name of the notebook and on the environment tab you could change the environment used to run the notebook as well as stop or restart the runtime environment.
If you'd like to share a read-only version of the notebook you can do that from here you can select how much of the content you'd like to share and how you want to share the notebook

either through a link or social media.
If you'd like to **schedule** the notebook to run at a different time you can create a job just provide a name for the job and select the scheduling options like specifying a date for the job to run and whether you'd like the job run to repeat after you create and run the job you can see the status on the jobs tab in the project.

Jupyter Notebook in Watson Studio - Part 2

This video shows you how to create a jupiter notebook let's start by adding a data asset to the project you can either browse to select files or drag files into the panel great now the data file is uploaded to object storage and available as a data asset. In this project next create a notebook provide a name and a description and then select the runtime to use when running this notebook here you see the environments you could use.
You'll learn more about environments later so for now just select the default spark Python environment and verify the language and spark version when you ready create the notebook now wait while the runtime environment is instantiated once the environment is ready.
In the notebook access the data sources and locate the file click insert to code and choose how you want to insert the data the choices in this drop-down box are dependent upon the language used in this notebook and the file type. Notice that the inserted code includes the credentials you'll need to read the data file from the object storage instance when you run the code the first five rows display.
Now let's take a closer look at environments on the environments tab you can define the hardware size and software configuration for the runtime associated with Watson studio tools such as notebooks you can see that there is one active environment runtime namely the runtime being used by the notebook you just created and here are the other default environments you can view any of the default environments to see a summary of the configuration and also create a new environment definition.
First provide a name in a description if you select spark for the type you'll see some additional configuration options in this case just accept the defaults and choose Scala for the software version when you ready create the new environment the environment is ready for you to use with a notebook to switch a notebook to use a different environment. You need to first stop the colonel then you can change the environment and select the custom environment you just created and associate that with the notebook now open the notebook in edit mode and wait for the new environment to be instantiated.
Since this notebook was last saved using a different kernel you need to set the new kernel let's delete the existing cell locate the source data file and insert a spark session data frame when you run the code the first five rows display now you're ready to explore the community and find sample notebooks and datasets to get started analyzing data.

Lab: Creating a Watson Studio Project with Jupyter Notebooks

Objective(s):

After completing this lab, you will be able to:

- Use Watson Studio service
- Create project in Watson Studio
- Add an interactive python notebook to a project in Watson Studio

Pre-requisite

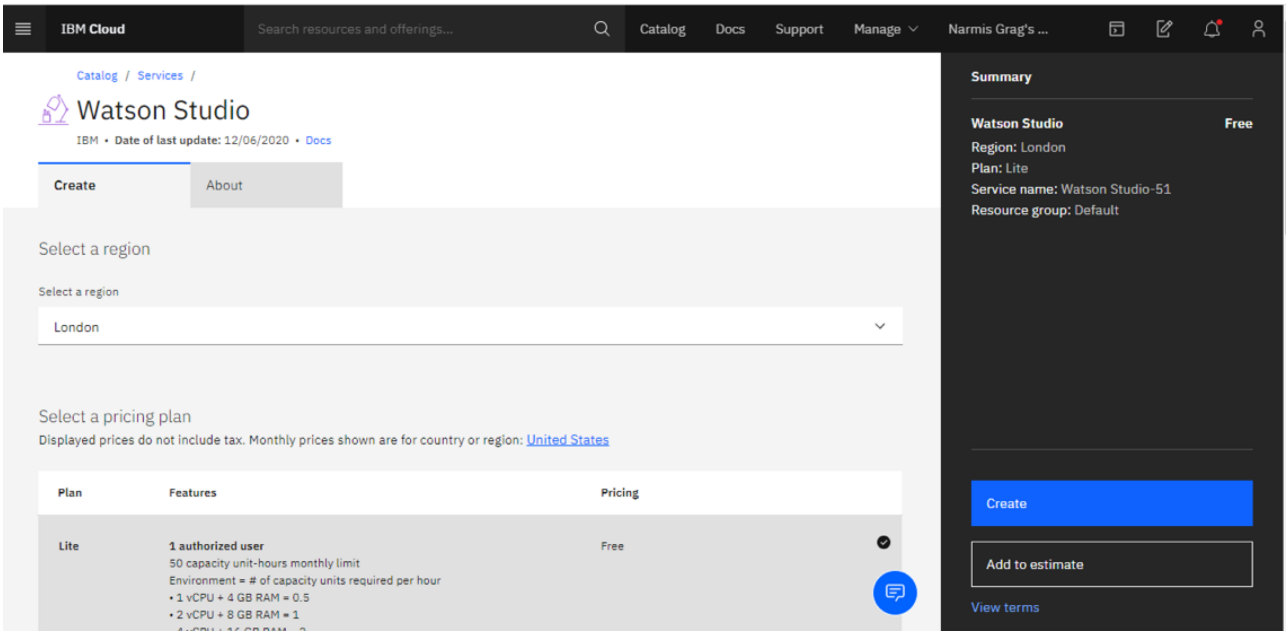
You need an IBM Cloud account to create a project in Watson Studio. If you don't have an account created already, click and open this link and follow the instructions, to create an IBM Cloud account.

Exercise - Create a project on Watson Studio

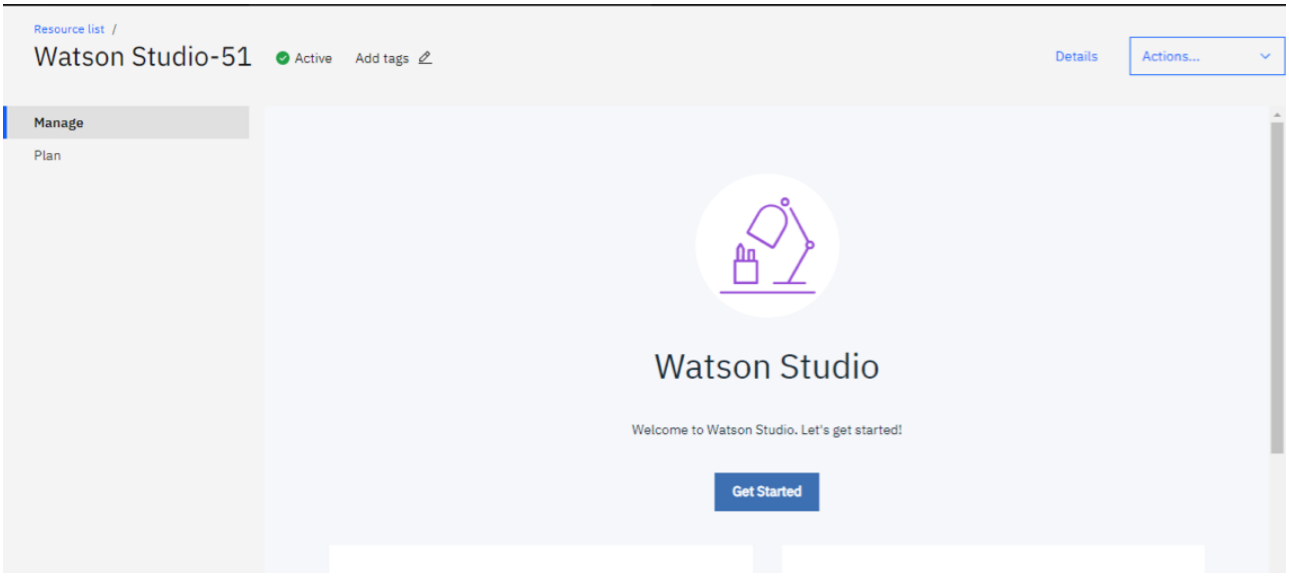
If you have not created a Watson service before proceed with Task 1, otherwise go to Task 2

Task 1: Create Watson Studio Service:

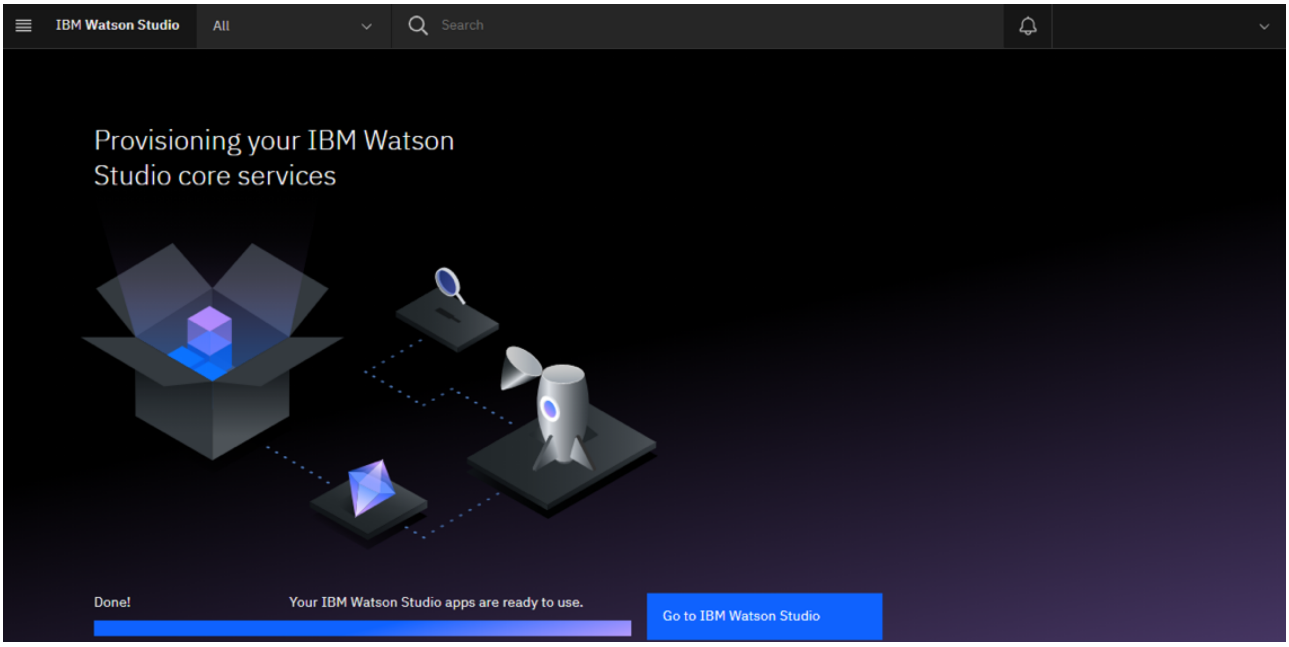
1. Click here to go to the IBM Cloud Watson Studio page. You will see the screen in the figure below. Click on the **Create** button.



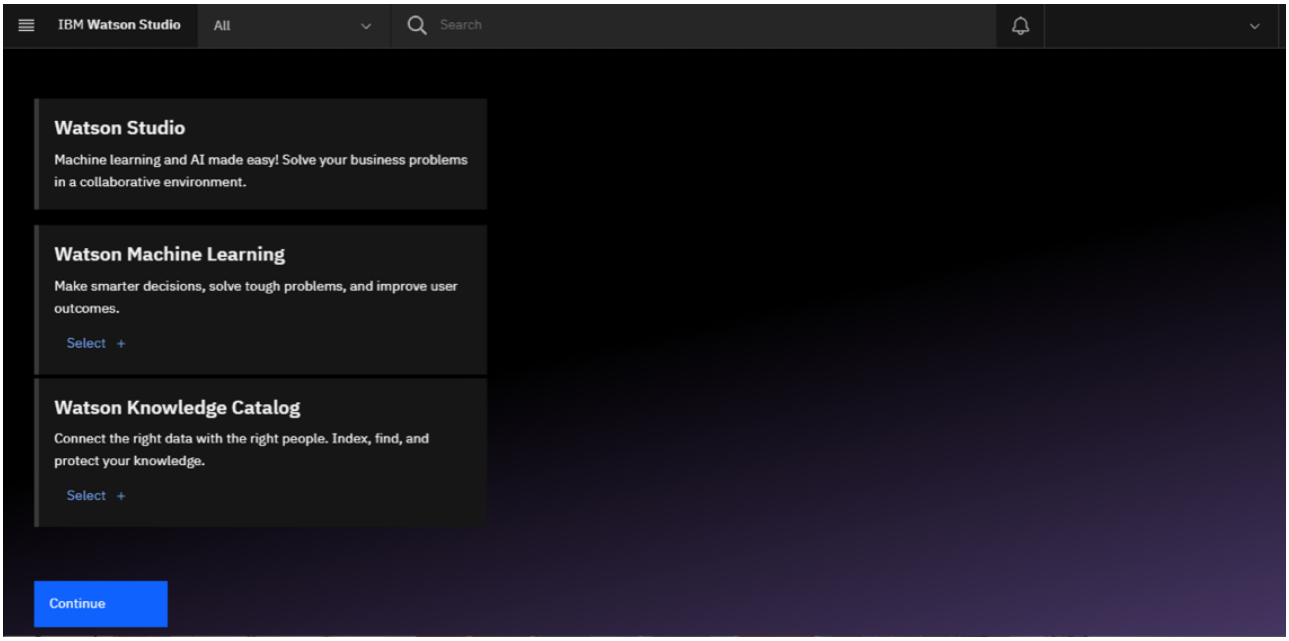
2. Now click **Get Started**.



3. Then click **Go to IBM Watson Studio**.

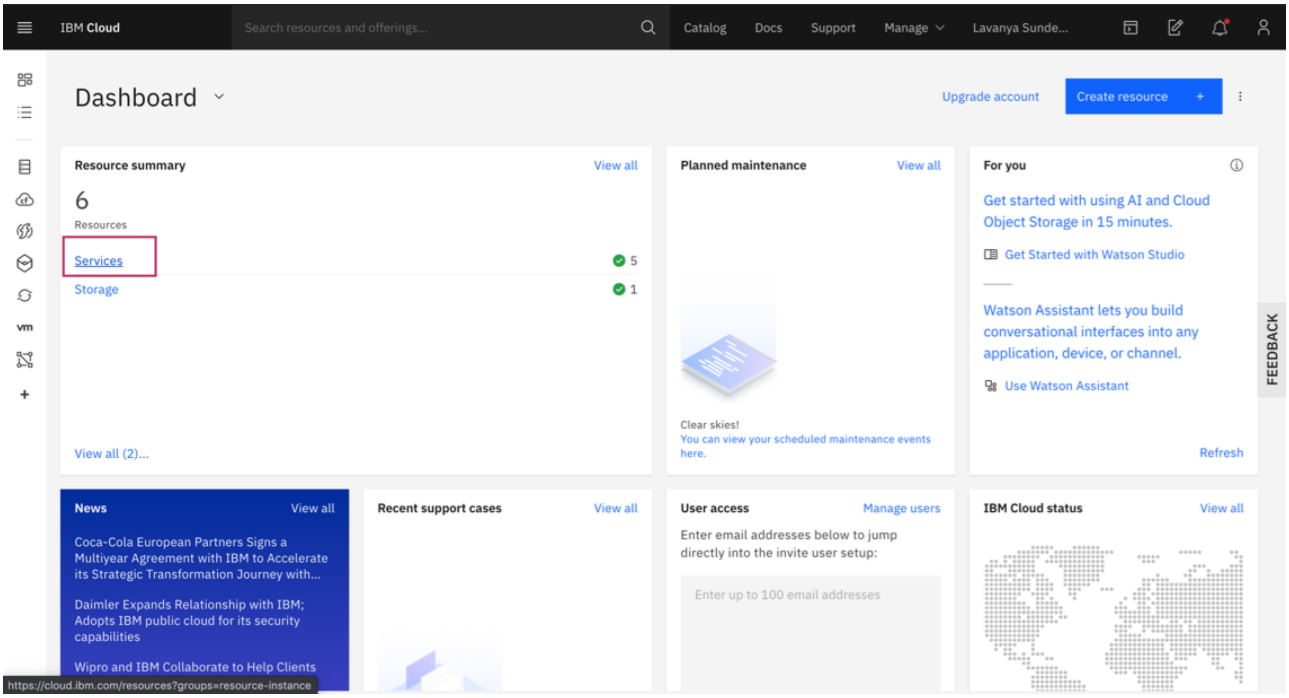


4. Then click **Continue**.

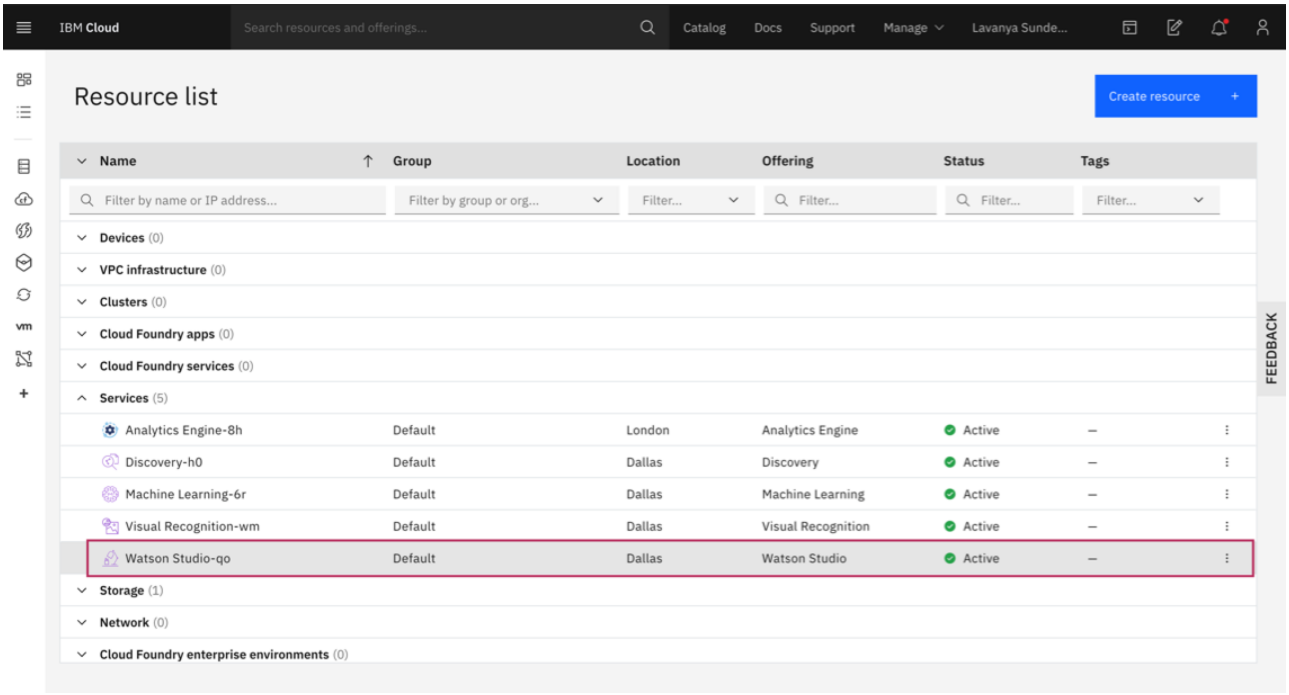


Task 2: Open Watson Studio

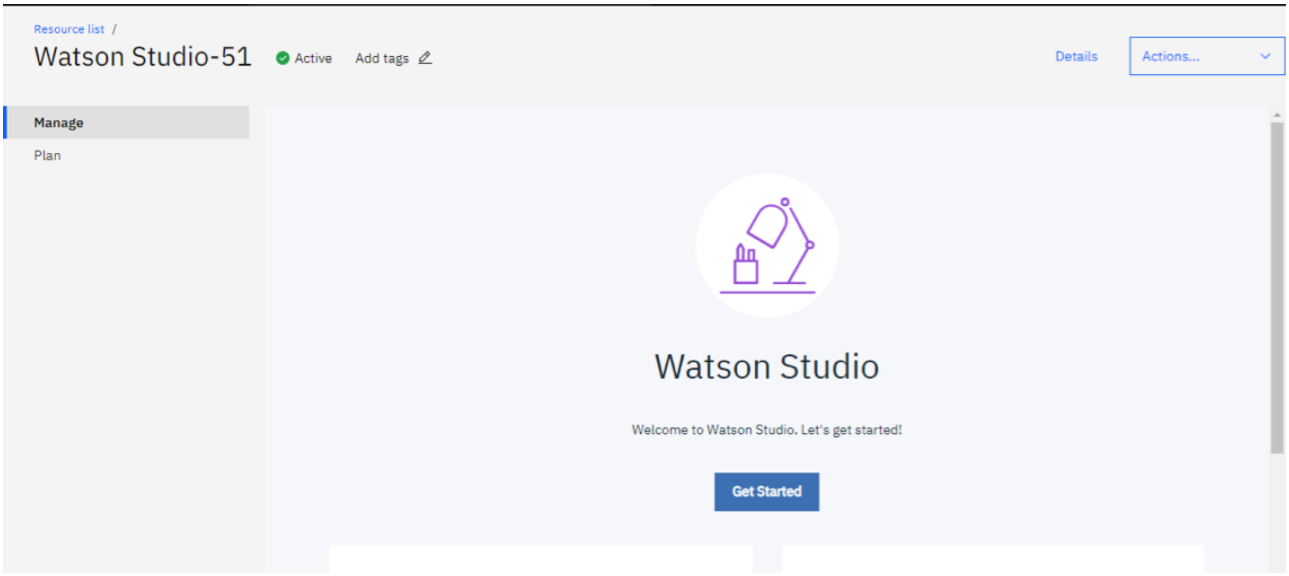
1. Go to the IBM Cloud Dashboard and click **Services**.



2. When you click on Services, all your existing services will be shown in the list. Click the Watson Studio service you created:

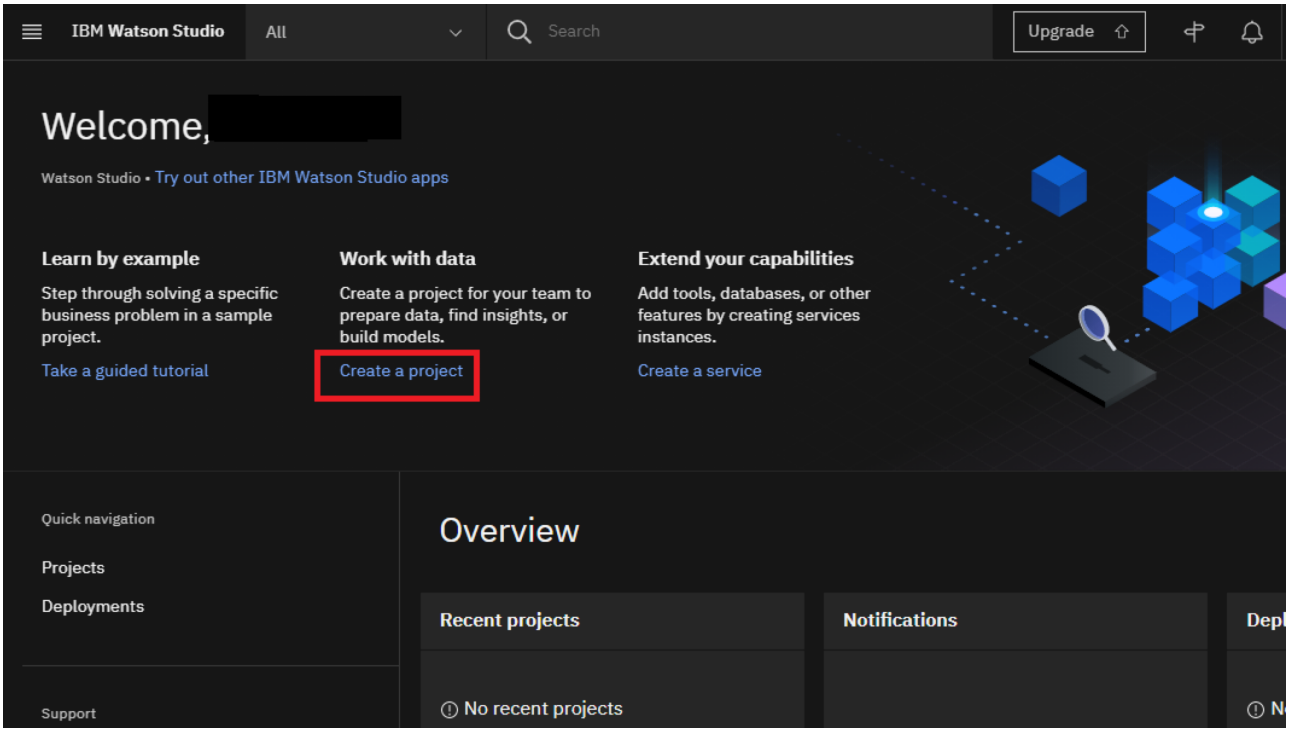


3. Then click **Get Started**.

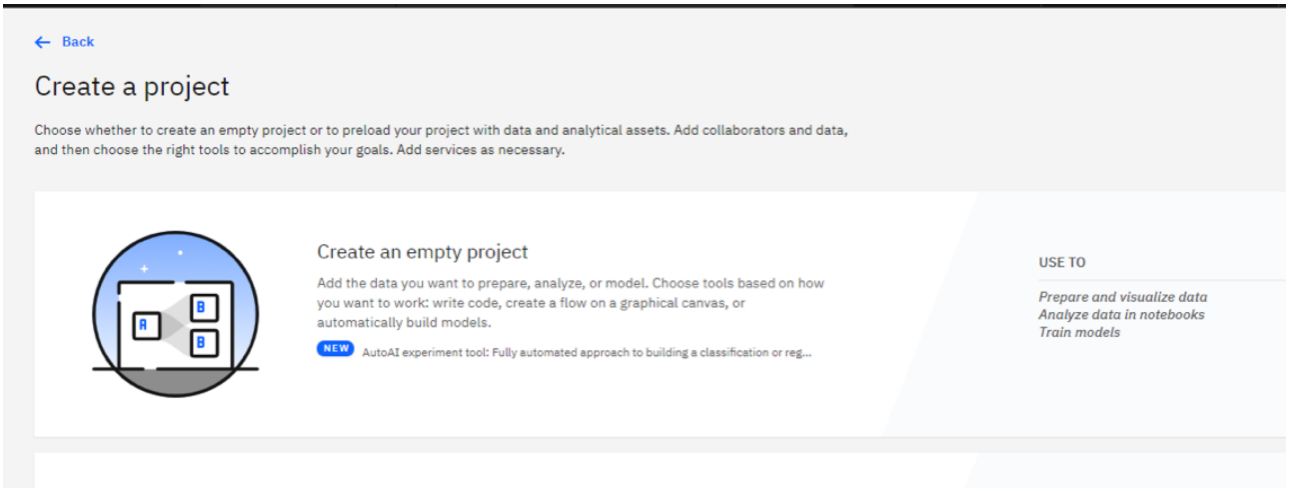


Task 3: Create a Project

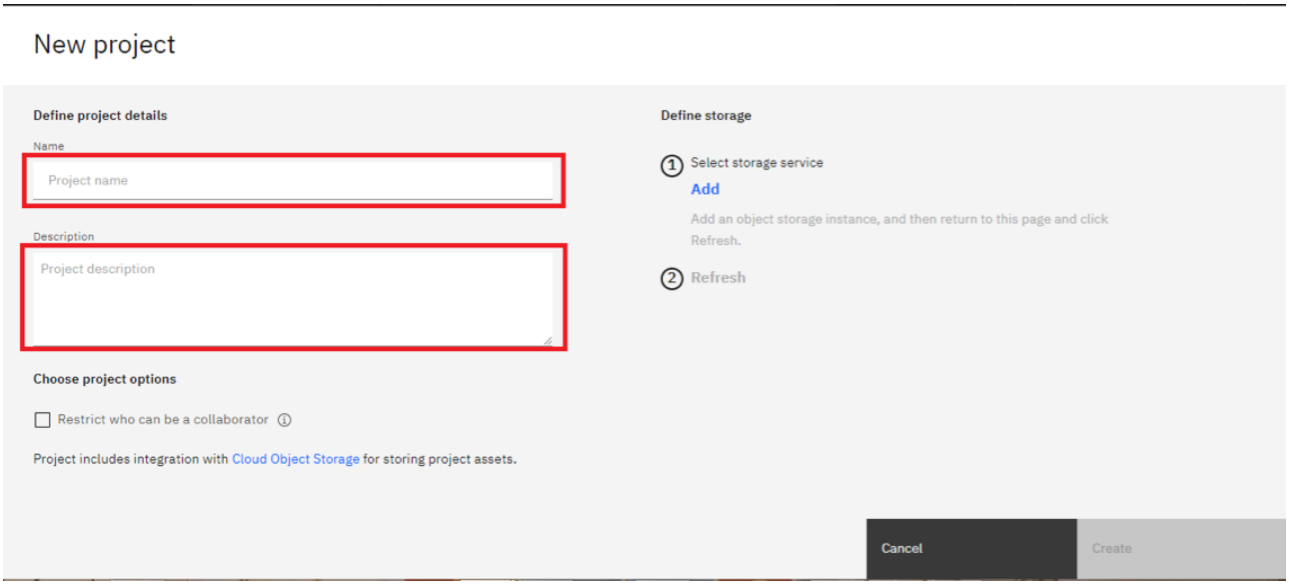
1. Click on **Create a project**.



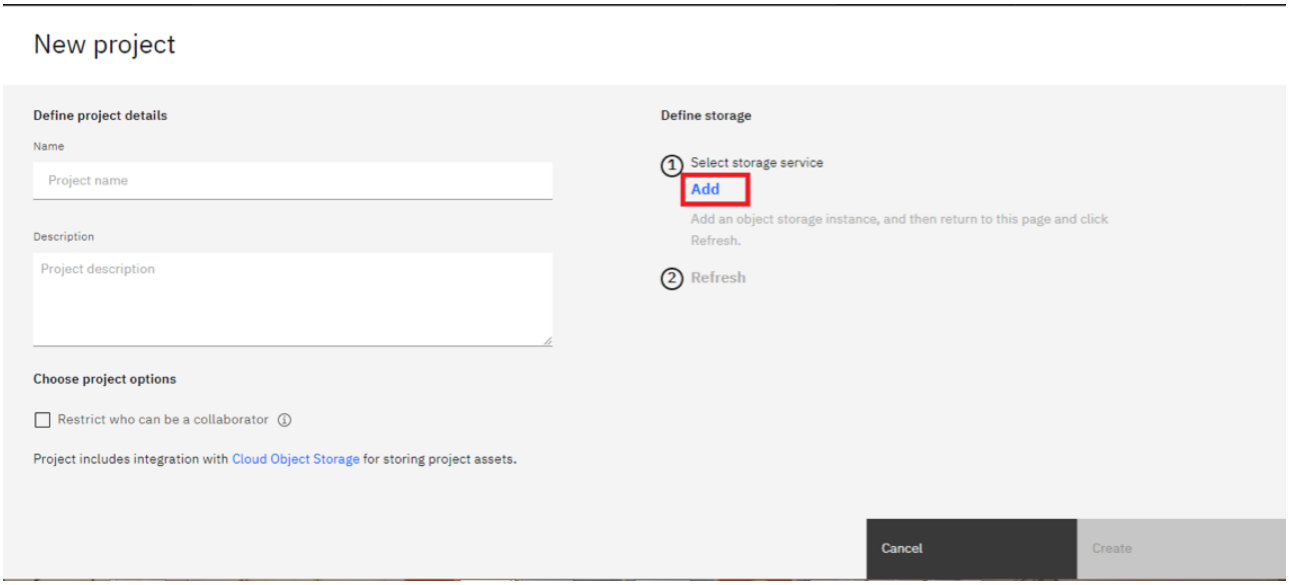
2. On the **Create a project page**, click **Create an empty project**.



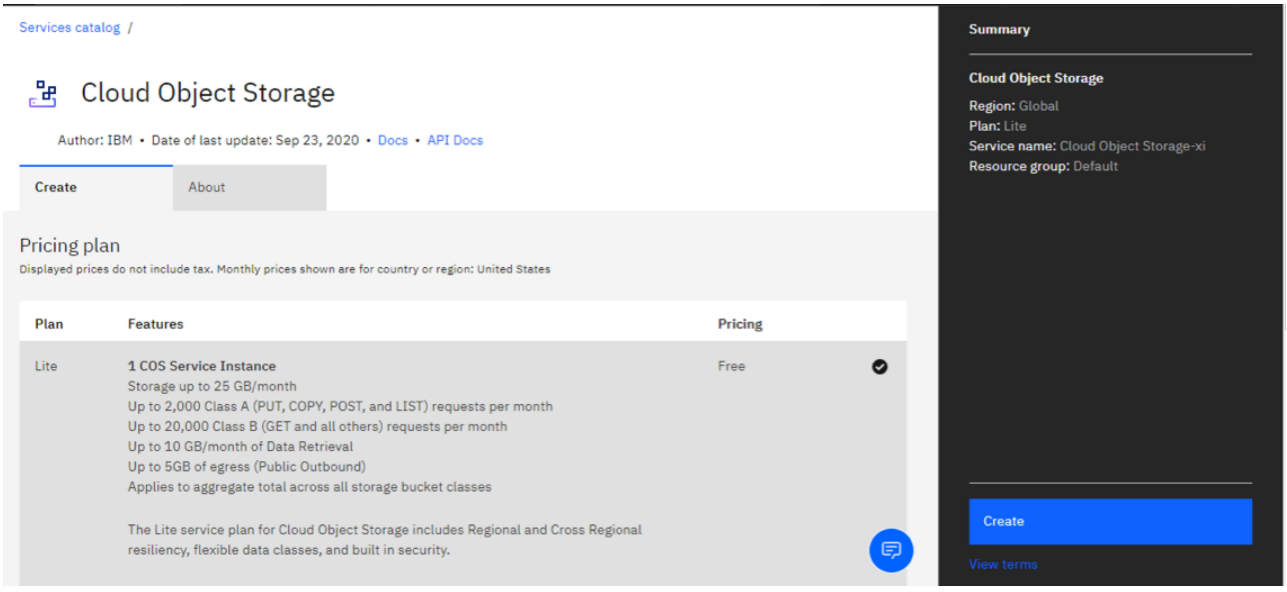
3. Provide a **Project Name** and **Description**.



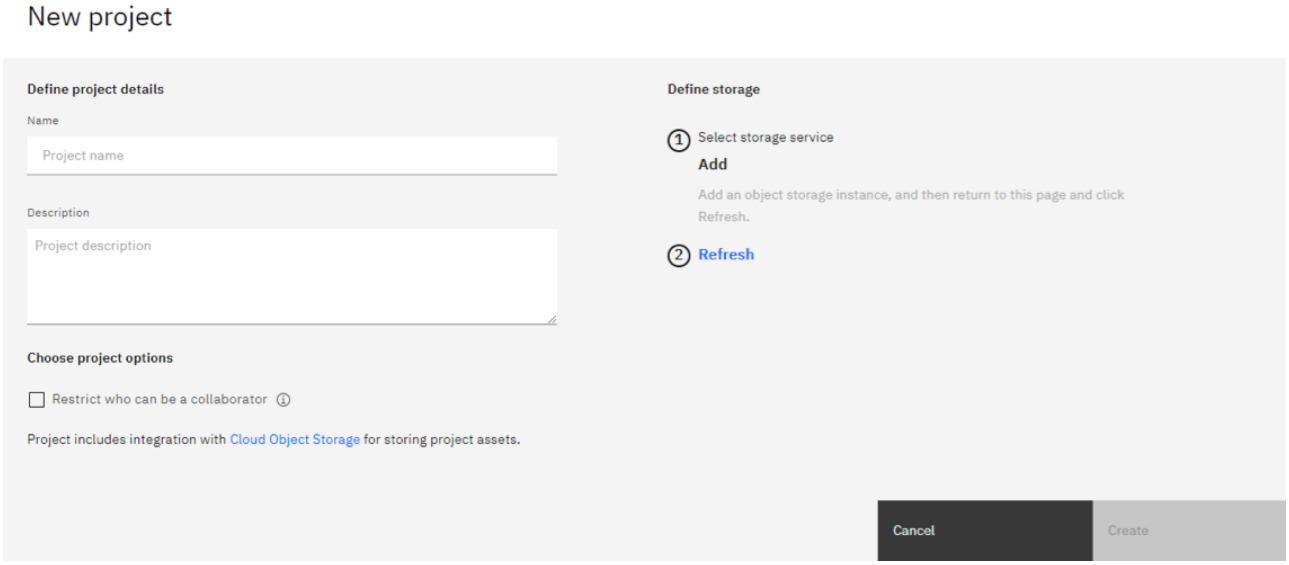
4. You must also create storage for the project. Click **Add**



5. On the Cloud Object Storage page, click **Create**.

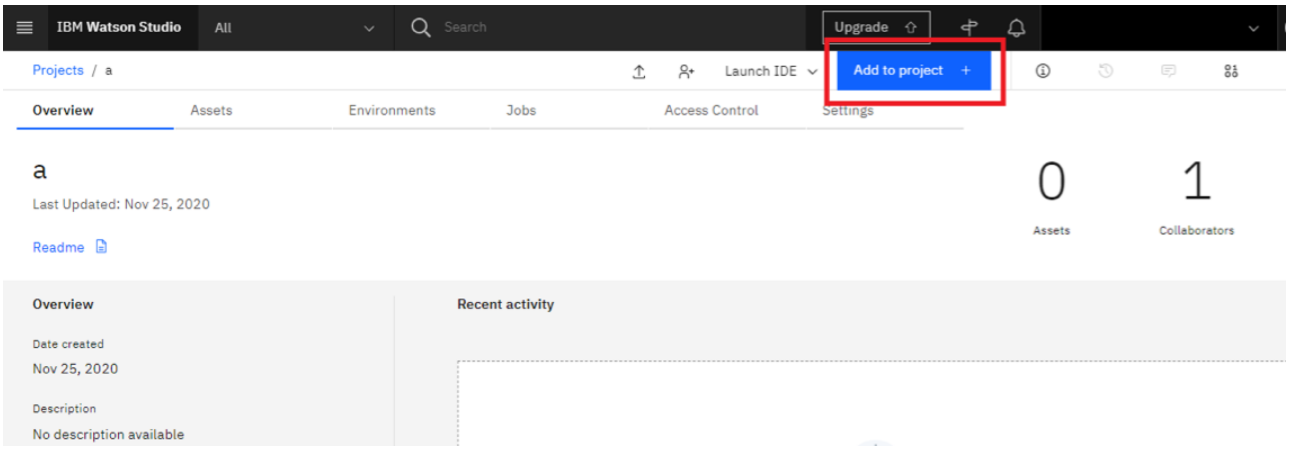


6. On the New project page, note that the storage has been added, click **Refresh** and then click **Create**.

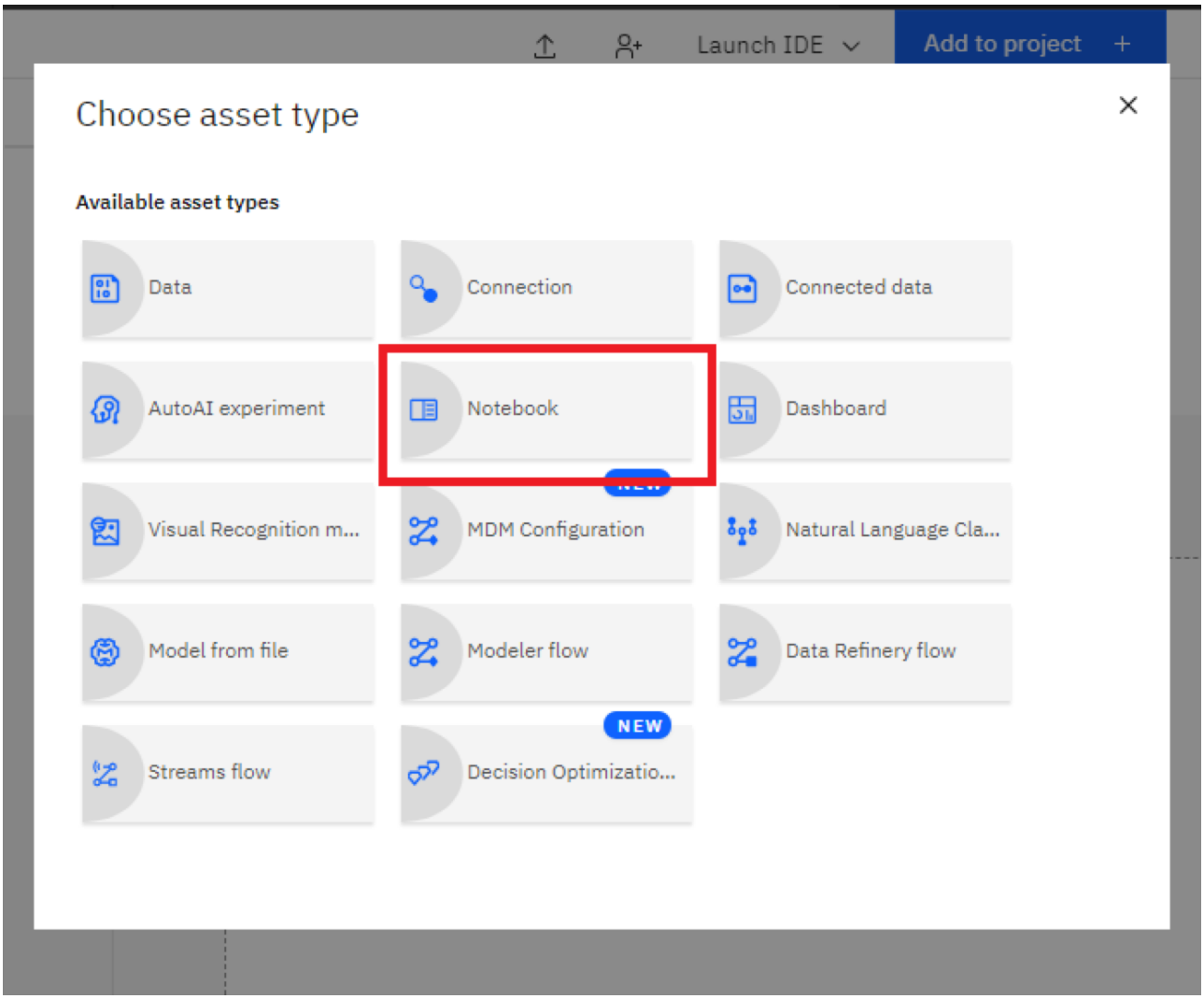


Task 4: Adding a Notebook to the Project:

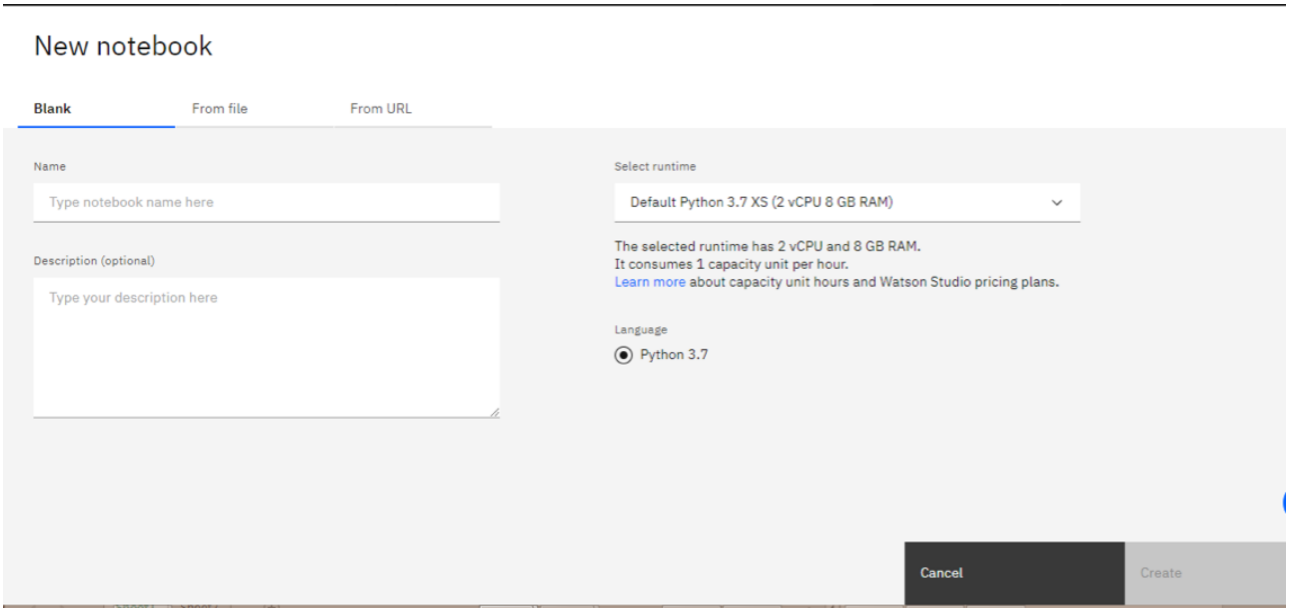
1. Click **Add to project**.



2. In the list of asset types, click **Notebook**.

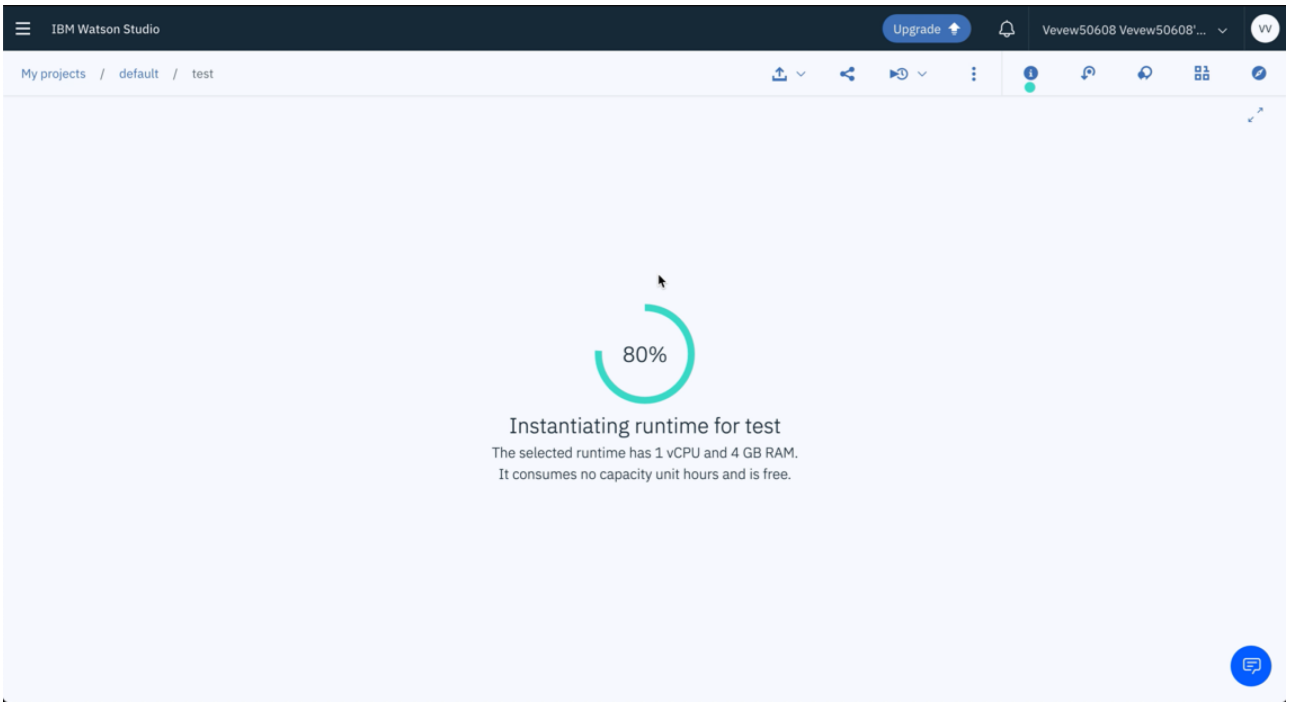


3. On the New notebook page, click **Blank** and then add a name and optional description for the notebook. Specify the language as Python and runtime environment. Click **Create**.

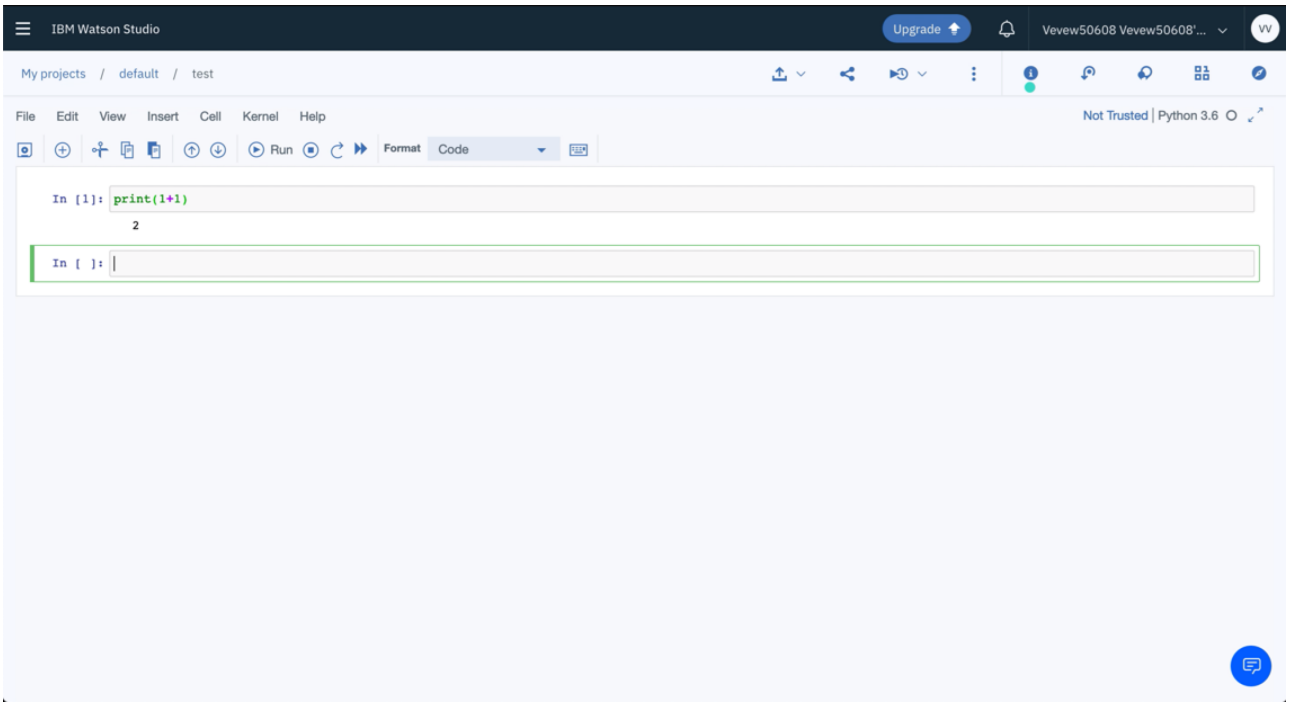


Just wait until the notebook appears. In case you are interested. The jupyter enterprise gateway has requested resources on the Kubernetes cluster IBM hosts for serving the jupyter kernel

backing your notebook.



Now you're ready to code!



This concludes this tutorial.

Other IBM Tools

Other IBM Tools for Data Science

In this video, we will look at several other IBM tools that help data scientists in their day to day work.

- **Watson Knowledge Catalog** helps data scientists to catalog and manage all their data resources.
- **Data refinery** provides graphical tools for analyzing and preparing data.
- **SPSS** based products include easy to use graphical interfaces for wide varieties of statistical and machine learning algorithms and data transformations.
- We will talk about approaches to model deployment, including open standards and **Watson Machine Learning**
- Newer features of Watson Studio include **AutoAI** that automatically computes the best data pipeline and
- **Watson OpenScale** which helps to ensure fairness and explainability of the models.

IBM Watson Knowledge Catalog

Most organizations have huge amounts of data stored in many forms in various locations. Finding relevant data quickly and connecting disparate data sources can be challenging and time-consuming. **Watson Knowledge Catalog unites all information assets into a single metadata-rich catalog**, based on Watson's understanding of relationships between assets and how they're being used and socialized among users in existing projects. Let's have a look at the overview of different tool categories that we've previously discussed.

Watson Knowledge Catalog corresponds to the **Data Asset Management**, **Code Asset Management**, **Data Management**, and **Data Integration** and **Transformation**. Watson Knowledge Catalog is a data catalog that is integrated with an enterprise data governance platform. It also merges the analytics capabilities of Watson Studio. The data catalog assists data scientists to easily find, prepare, understand, and use the data as needed. Watson Knowledge Catalog protects data from misuse and enables the sharing of assets with automated, dynamic masking of sensitive data elements.

Data-profile visualizations, built-in charts and statistics help users to understand data assets. Seamless integration with Watson Studio helps data citizens to drive production of their data in a suite of powerful data science, AI, machine-learning and deep-learning tools. Joining with Watson Studio directs the building, training, and deploying of models. Users can interactively discover, cleanse, and prepare data with a built-in **data refinery**.

Possible connections to more than 30 IBM and third-party data sources help to catalog and use your data in the original locations. IBM Watson Knowledge Catalog has various deployment choices on IBM Cloud™ and can be run anywhere with **IBM Cloud Pak™ for Data**. The latter is a f**ully-integrated data and AI platform** built on Red Hat® OpenShift® Container base. It can be deployed easily into any public or private cloud or other enterprise platforms.

A catalog contains **metadata** about the contents of assets and how to access them. And a set of collaborators who need to use the assets for data analysis. The metadata is stored in an encrypted IBM Cloud object storage instance. Any data that you want to store in the Cloud, you

can upload to the cloud object storage of your choice, and then specify that object storage when you create the catalog.

This s**plit between** where the data's **metadata** is stored and the actual location of the **data** is important. It means that you can keep your data where ever it is. You don't need to move it into the catalog because the catalog only contains metadata. You can have the data in unpremises data repositories in other IBM cloud services like Cloudant or Db2 on Cloud and in non-IBM cloud services like Amazon or Azure, in streaming data services or even dark data sources like PDFs. Included in the metadata is how to access the data asset. In other words, the location and credentials. That means that anyone who is a member of the catalog and has sufficient permissions can get to the data without knowing the credentials or having to create their own connection to the data.

Since the new catalog is empty, let's take a look at an existing catalog. On the Browse **Assets** tab you can see "recommendations", "highly rated assets", and "recently created assets", as well as a list of all the assets. You can type a search term to find assets, and you can filter by asset type, such as Data Asset or Notebook. Or filter by tags that were assigned to the asset when it was added to the catalog. When you view an asset, you get a preview of the data and other information like a description, ratings, tags, where the source is located, and any classifications. On the **Access** tab, those with permission can add members to view this particular asset. And the Review tab shows reviews and lets you contribute a review. When assets are added to a catalog with Data Policies enabled, Watson Knowledge Catalog automatically profiles and classifies the content of the asset based on the values in those columns.

The **Profile** tab contains more detailed information about the inferred classifications. You can see the other possibilities for classifying each column and the confidence scores for those other possibilities. On the **Lineage** tab, you'll see the various events that Watson Knowledge Catalog has captured that occurred in the lifecycle of this data asset, allowing you to trace what's happened to the asset since it was created. On the **Access Control** tab, you can see the current list of catalog members. you can also add members which is pretty similar to adding collaborators in a project.

Most catalog members will likely have the editor role. The viewer role is intentionally restricted and only a select few users will have the admin role. Watson Knowledge Catalog includes capabilities to automatically mask sensitive data according to your organization's governance policies. For example, you can see in the diagram that the first name, last name, and gender data in the data set have been masked. You've learned how IBM Watson Knowledge Catalog can help organizations deal with their numerous data and other assets. In the next video we'll look at Data Refinery, a powerful tool for analyzing and preparing data.

Data Refinery

IBM Data Refinery addresses this issue and simplifies the task of refining data and its

workflows. It provides a self-service data preparation environment where you can quickly analyze, cleanse and prepare data sets. Data refinery is available with Watson Studio on public cloud, private cloud and desktop.
In the rest of the video we will walk through a scenario and see Data Refinery in action. In this scenario we will use Data Refinery to find the best deals using data about discounts offered over time. We will then **automate** the analysis to run on a regular schedule. Before the Data Scientist starts, she looks at the data distribution and notices that the inSale column is missing data. She visualizes the offer column and notices that it contains valuable information about discounts. Many fields contain the percent of information, some contain references to previous price indicating a new reduced price being available. She decides to derive sale from offer. She uses a conditional decrease operation to derive if the product is on sale. Next she uses a filter operation to eliminate deals that are not on sale She then wants to pick up the bargains. She uses the replace substring operation and provides a pattern that extracts the discounts from the offer. After converting the discount values to a decimal she can visually see the discounts that were available. She needs to find the months that offered the best deals. She visualizes the date Updated and notices that the date field has a variety of formats, some with dashes some with slashes and some with months as text. She hopes that Data Refinery can normalize the data and extract a month. She uses the convert column operation to convert to date and selects ymd. Next she extracts month and creates a derived column called discountMonth.
The data now represents all brands and products providing sales and the month the offer was available. The data scientist is only interested in her preferred brands. Over time she has built a list of preferred brands and has imported the data in her project. Data Refinery provides **relational transformations** such as left, inner right, full, semi and anti-join. To ensure that the data only contains her preferred brand she uses a semi-join operation which narrows the brands to match her preferences. She then selects the keys for the join and the resulting fields. The visual results now confirms that the brands match the preferences.
To find the best possible deals she needs to perform some aggregations. Several features determine a good deal. She is interested in the best offer and duration when the discounts are active. Aggregating the sale data will help understand the deals. She groups the columns by brand and discountMonth and calculates the maximum discount. Finally she sorts the result in descending order Data refinery is now displaying the best deals by brand preferences and the duration which the offer is available.
The last step is to execute the analysis on the full dataset. She starts the full analysis, which she can monitor for the completion status. It's time to automate the analysis which runs on a regular basis. The data in the database can grow over time. She uses a personalized runtime to match the larger data volumes and sets a schedule for automation. The hourly schedule reads from updated data from the database and writes to the target table. Data Refinery has helped her uncover deals in the raw data through a small set of operations and transformations with the bulk of the work done for her.

SPSS Modeler Flows in Watson Studio

In this video, we will take a look at an easy-to-use, **graphical way to build machine learning models** and pipelines. SPSS Modeler Flows is a part of Watson Studio, which was inspired by another product, IBM SPSS Modeler. We'll discuss that product in a later unit. Let's have a look again at the overview of different tool categories. Modeler flows include some **data management capabilities**, as well as tools for data preparation, visualization, and model building.

All flows are created using a drag-and-drop editor and consist of "nodes" of various types, with data "flowing" from one node to the next according to their connections. A sample Modeler flow shown here includes two data **source nodes** shown in purple on the left; **type**, **aggregate**, **filter**, **merge**, **filler**, and **partition** nodes in the middle; 2 **model building nodes** shown in pentagons. Once a flow is executed and the models are built, the upside-down pentagon "model nuggets" are created. They can be used to see information about the models and to get predictions for new data. And the three green square nodes on the right provide model evaluation information in the form of tables and charts.

You can build your SPSS Modeler flows by dragging different types of nodes from the left, the part of the screen called the "palette," to the "canvas," the main part of the screen. Each flow starts with one or more data sources located in the "Import" group, and can include some or all other types of nodes. Watson Studio provides some sample flows to help new users.

In the Drug Study example shown here, we are using a small artificial data set. The target variable is a categorical field, "Drug," that has five categories, and there are several predictor variables. This flow creates a new "derived" field by dividing the values of one of the predictors by values of another one, and at the end builds a small neural network model and a decision tree model. When a user clicks the "Run" button on the top panel, denoted by a triangle, the flow is executed and the models build. This is reflected in the new pentagon nodes, called "model nuggets," that display under each model node. If you click on the three dots in the upper right corner of one of those nodes and select "View Model", you will see various types of model information. By connecting new data sources to the model nugget, you can get predictions on new data. The first window in the model viewer shows model accuracy and related measures, such as precision and recall. This toy data example enabled us to get perfect accuracy, which is normally not the case with real life data.

The Confusion Matrix view shows how model predictions on the training data matched the observed target values. Once again, in this toy example all cases were classified correctly. We can also look at Model Information, which displays a table that tells us more about the details of the model. Feature Importance displays a diagram that indicates the relative predictive strength of various model inputs.

Finally, the Network Diagram gives a visual representation of the neural network model we built.

On the left is the input layer, with units corresponding to each continuous predictor and each category of the categorical predictors, plus a bias unit that is usually present in each layer of a neural network. In the middle, we see a “hidden layer” with 7 units, or neurons, and a bias unit. On the right is the output layer with 5 units corresponding to the five target categories. Controls on the right and bottom of the diagram enable some interactive exploration of the model. The colors of the connections between units indicate the values of the weights on those connections. We can also look at the decision tree model built using the C5 algorithm. A Model Information table and Feature Importance chart appear as before. Additionally, a Top Decision Rules table is displayed.

Decision tree models are popular because they have a special structure that makes it easy to explain predictions or extract decision rules. The tree diagram is also displayed. On the left side of the canvas, we see a part of the model palette that can be used in the flows. At the top are “Auto Classifier” and “Auto Numeric” nodes that can be used for categorical and continuous targets, respectively. Those nodes will build several kinds of models and pick the best one based on a certain criterion.

Later, we will talk about the AutoAI feature of Watson Studio; AutoAI takes this capability to the next level by automatically finding not only the best model, but an entire data pipeline, which includes various data transformations. In this video, you've learned how Modeler Flows in Watson Studio can help analysts to create powerful machine learning pipelines using a graphical interface without the need to write any code.

This feature was based on IBM SPSS Modeler. Next, after completing a lab to give you hands-on experience with this powerful technology, we will take a look at two other IBM products that can be used for Data Science: IBM SPSS Modeler and IBM SPSS Statistics.

IBM SPSS Modeler

In this lesson we will discuss two products that are very helpful for data scientists. Both came to IBM with the SPSS acquisition in 2009. First is IBM SPSS Modeler. Let's review the different tool categories we discussed previously. **IBM SPSS Modeler** includes **data management** capabilities and tools for **data preparation**, **visualization**, **model building** and model **deployment**. The product was created by Integral Solutions Limited in the United Kingdom in 1994 and was originally called Clementine. It was acquired by a company called SPSS in 1998 and SPSS was in turn acquired by IBM in 2009.

SPSS Modeler is a **data mining** and **text analytics** software application. It's used to build predictive models and conduct other analytics tasks. It has a visual interface that enables users to leverage statistical and data mining algorithms without programming. One of its main goals from the beginning was to **create complex predictive modeling pipelines that are easily accessible**.

A sample modeler stream shown here includes one round data source node, three triangular

graph nodes, one hexagonal node for computing, a new variable, and a square node for an output table. Below the canvas, we can see the rich node palette with separate tabs for data sources, record in field operations, graphs, models, output and so on. Nodes and different tabs have different shapes with Pentagon's used for modeling nodes.

Let's examine the sample stream that comes as an example with the product. It starts with a data set of telecommunications records and the goal is to build a model to predict which customers are about to leave the service otherwise known as **churn**. The data source is shown by the round node on the left side, a hexagon type node typically follows a data source node and it enables us to specify roles, target predictor or none. And measurement levels such as continuous nominal or flag for all variables. The term **flag** is used to denote a **variable with two categories** one of which can be considered positive and the other negative.

In this example the measurement level for the churn field is set to flag and the role is set to target. All others are set as predictors and inputs. The original data set has many fields and some of them are not relevant to the target variable, so we first need to decide which fields are more useful as predictors. There is a feature selection modeling node that helps to do this. After the stream with the feature selection node is executed a yellow model nugget gets created below it in the flow diagram.Using that nugget we can generate a filter node that filters out the variables that are not good predictors for the target.

The data audit node located below the filtering node shows various properties of the data such as numbers of outliers in each variable and the percentage of valid values. It can also help to create a special node for missing value imputation that is replacing missing values of a variable with some valid values that can be selected based on domain knowledge.

Here variable log toll has greater than 50% missing values and we will specify a value the mean to replace them. A super node in modeler is a special node that is not found in the palette but is created by the user with special functions included in it. The data audit node enables us to create a super node for imputing missing values. It is shaped as a star and shown on the right of the screen.

Finally we attach the logistic regression model node to the stream and click run. Another model nugget appears and by clicking it we can see various model information and other output. in the output window that opens when we click on the model nugget the summary tab shows the target inputs and some model building settings. Based on certain advanced output settings that were specified before the model was built we can also see a classification table, accuracy, and some other generated outputs for the model. Note that these results are based on training data only.

To assess how well the model generates two other real-world data you should always use a partition node to hold out a subset of records for the purposes of testing and validation. Then, in the model setup screen select the use partitioned data check box. This will help detect and avoid model overfitting. Overfitting is defined as having significantly higher accuracy on the training data. Data used for training the model then on tests or unseen data. The yellow model

nugget added earlier can also be used to compute predictions, also called scores on the original data or on a new data source. All we need to do is to connect the data source in question to the nugget, make sure it has the predictor variables used in the model, and create an output to a table or other structure for storing the scores. We can also specify settings for scoring inside the model nugget. Note that if the model was built on transformed predictor data, the same data transformation steps would be applied to the new data before it can be scored by the model.

The analysis node is the final node in the stream. It attaches to a model nugget and when executed it will compute some model evaluation metrics, auch as a confusion matrix and accuracy. In this example we've only looked at a logistic regression model. IBM SPSS Modeler offers a rich modeling palette that includes many classification, regression clustering, Association rules and other models. It also contains large selections of data source types, data transformations, graphs, and output notes. And we haven't even talked about text analytics, entity resolution and many other features of the product that can be extremely helpful to data scientists. We could create an entire course on IBM SPSS Modeler alone. You've learned how IBM SPSS Modeler helps analysts to create powerful machine learning pipelines using graphical interface. Next, we will talk about the original SPSS product now called IBM SPSS Statistics.

SPSS Statistics

IBM SPSS Statistics evolved from an original product that was released in 1968. That product was called "**Statistical Package for Social Sciences**," or "SPSS." IBM SPSS Statistics is a statistical and machine learning software application and is widely used in academia, government agencies, and large enterprises. It's used to build predictive models, perform statistical analysis of data, and conduct other analytic tasks. It has a visual interface, which enables users to leverage statistical and data mining algorithms without programming, although the interface is very different from Modeler.

As you can see, the main section of the screen looks very much like a spreadsheet; it displays data and allows manual editing. This particular small data set, called "Employee Data", was created some time ago and does not represent real people. It is shipped with the product for use in demos and tutorials. At the bottom of the screen, we can see two tabs:

- Data View
- Variable View.

In the Variable View, we can see and edit the information about all variables, including names, labels, data types, and measurement levels. We can also specify labels for values of categorical variables, and missing values. At the top of the data window is a menu.

Under File, if you select "Import Data," you will see a list of a wide variety of data formats that you can import. The product uses its own data file format with the extension ".sav" that saves all the information about the variables we just saw in Variable view. The menu enables importing

from and exporting to many other formats. Under "Data," you'll find an extensive menu of possible data operations. Note that Data Validation can be performed using user-defined rules that specify the expected behavior of variable values. For example, if the date and month are kept in separate columns, the date cannot exceed "31," but for February, the date can't exceed "29." A special rule can therefore be created and applied during data validation. Additionally, you can enable some checks, such as percentage of missing values in a record or in the field. When you click the "Transform" menu item, you'll find a variety of available data transformations. Under "Compute Variable…" you can write a formula for a new variable based on existing variables. You can use any of the many mathematical and statistical functions available in the product. You also have the option to use automatic data preparation, similar to Modeler. In the "Analyze" menu, you will see many types of statistical and machine learning analysis. Under "Regression," there are a variety of regression-related models. There are other kinds of regressions that appear separately on the Analyze menu, including General Linear Model, Generalized Linear Models, Mixed Models, and Loglinear.

Now let's build a decision-tree model on the data. For this exercise we'll try to predict the "Employment category" field based on other fields. In the "Analyze" menu, select "Classify" and then "Tree". In the Decision Tree window, we can specify the dependent variable "Employment Category," and use most other fields -- except id and bdate -- as predictors, or independent variables. Usually the ID variable should not be used as a predictor, because it will not help with new cases, and the birthdate does not seem to be a useful predictor in this example either. We'll select "Exhaustive CHAID" as our Growing Method, although there are also three other options available.

Data scientists often try many different models to see which one works best for their data. Here we are just looking at one example model in order to illustrate how the product works. Click the "Validation" button to open the Decision Tree Validation window. Here, we select "Split-sample validation" to make sure we test the model on new data. Click "OK" in the Decision Tree window, to generate the output, including the tree diagram shown here. A Classification table is also displayed that shows how well the model works on training and test data. In this case, the accuracy is 91.2% on training data and only 85.6% on test data, which means the model does not generalize to new data very well. It's possible that by using different models, we can get better results. Let's move to the next menu item. When you click "Graphs," you'll open a versatile Chart Builder, in addition to several other options.

The Chart Builder enables us to choose a style from the gallery and to drag required fields onto the canvas, select colors, and choose from other options. Here's an example after we drag the "Previous Experience," "Current Salary," and Gender variables to the corresponding slots to define the axis and colors for the dots on the chart. The plot in the canvas is not based on real data, this example simply gives you an idea of what to expect. Here is the real plot obtained from the data that we've been using. It shows different colored dots for gender, and regression lines that show the relationship of the current salary to previous experience for each gender.

Throughout IBM SPSS Statistics, you'll see a "Paste" button. When you click the "Paste" button, instead of executing the task right away the application will open another window, called the Syntax editor. Here, you can see the code called "syntax" pasted for you. SPSS syntax is a special programming language. For example, here is the code for the decision tree we just built. Once we have the syntax, we can execute it, manually edit it, store it for later use, or send it to other users of IBM SPSS Statistics.

Experienced SPSS users can write the code from scratch, while others might prefer to have it generated by the graphical interface. Remember, the option to paste syntax is available in throughout the program. If the syntax is generated by all the steps in a data analytics process -- opening the data set, applying any data transformations, building models -- and then saved as a syntax file with the extension ".sps", it's similar to saving a stream in IBM SPSS Modeler. However, one important difference is that it does not allow for an easy way of scoring new records with the model.

We'll talk about different ways to deploy models in the next section. You've learned how IBM SPSS Statistics helps data scientists to analyze their data using many statistical and machine learning techniques. Using a graphical user interface, we can create complicated analysis that can be saved in the form of syntax and reused later. Next, we will talk about predictive model deployment, an important part of the overall data science lifecycle.

Model Deployment with Watson Machine Learning

So far, we've talked about building machine learning models and pipelines. In most practical applications, the return on investment is obtained when the model or pipeline is put into production, where it is used to get predictions, or scores, for the new cases.

Let's look back at our overview of different tool categories. In this unit, **Model Deployment** is our focus. Suppose you worked hard to create the best possible machine learning model and the data preparation pipeline for it. How will you deploy your models? In many practical scenarios, models are built and deployed by different teams, using different programming, and perhaps human languages. The teams will use different computing and data storage environments, and It might prove difficult to translate your program and the associated data preparation and post-processing steps from one environment to the other.

Currently there are several approaches you can use to solve this problem, some commercial, some open source. Yet each one typically supports only a subset of all possible models, from building them to deploying, so a user gets locked into a specific framework. Open standards for model deployment are designed to support model exchange between a wider variety of proprietary and open source models.

Predictive Model Markup Language, or "**PMML**," was the first such standard, based on XML. It was created in the 1990s by the Data Mining Group, a group of companies working together on the open standards for predictive model deployment. IBM and SPSS were among the founding

members of the Data Mining Group. PMML 4.4 was recently released. It includes 17 statistical and machine learning models and many data transformations, built-in functions, ways to combine multiple models together, and other features. This standard is widely known and used. The products we looked at earlier -- Watson Studio, IBM SPSS Statistics, IBM SPSS Modeler -- enable users to export most models in PMML. In 2013, a demand for a new standard grew, one that did not describe models and their features, but rather the scoring procedure directly, and one that was based on JSON rather than XML. This led to the creation of **Portable Format for Analytics**, or **PFA**. PFA is now used by a number of companies and open source packages. After 2012, deep learning models became widely popular. Yet PMML and PFA did not react quickly enough to their proliferation.

The need for a standard intermediate representation was amplified by the wide variety of emerging deep learning frameworks and specialized hardware. In 2017, Microsoft and Facebook created and open-sourced Open Neural Network Exchange , or "ONNX." Originally created for neural networks, this format was later extended to support "traditional machine learning" as well. There are currently many companies working together to further develop and expand ONNX, and a wide range of products and open source packages are adding support for it.

Watson Machine Learning is IBM's commercial offering designed for model deployment. It supports deployment of models built with most open source packages, as well as those expressed in PMML or ONNX. It also supports deployment of IBM SPSS Modeler streams and Modeler flows from Watson Studio. Deployment can be done using a graphical interface or Python code, and can be for online scoring through a REST API or batch scoring. Watson Machine Learning helps integrate a deployed model into applications in the form of code snippets in several programming languages. In this video, you've learned how open standards and Watson Machine Learning can help users to deploy their models into various application. Next we'll talk about AutoAI and OpenScale, two advanced Watson Studio features that help to further simplify a data scientist's work.

Auto AI in Watson Studio

In earlier sections we saw how IBM SPSS Modeler and Watson Studio Modeler flows allow you to graphically create a stream or flow that includes data transformation steps and machine learning models. Such sequences of steps are called data pipelines or ML pipelines. This section examines a feature of Watson Studio that helps to **automate** the **creation of machine learning pipelines**. This allows data scientists to produce results much faster and to focus on more creative work. There is currently a shortage of qualified data scientists. Many operations that a data scientist typically performs are repetitive and time-consuming. Therefore, automating some of that repetitive work will help free up both new and experienced data scientists to do the important work that they are trained to do.

The **AutoAI** system was developed by IBM Research experts in collaboration with IBM Distinguished Engineer and two-time Kaggle Grandmaster Jean-Francois Puget. It **provides** a **graphical interface to create and deploy machine learning models with real time visualizations**.

AutoAI automatically performs typical machine learning steps, such as: Data preparation Model selection Feature engineering Hyper-parameter optimization Users can view the progress on the graphical interface.

This example shows the training of a model to predict whether or not a customer is likely to buy a tent from an outdoor equipment store. We start with structured data. In this historical data, there are four feature, or "predictor," columns:

- GENDER: The customer's gender
- AGE: The customer's age
- MARITAL_STATUS: "Married", "Single", or "Unspecified"
- PROFESSION: The general category of the customer's profession, such "Hospitality" or "Sales", or simply "Other."

The model will learn to predict the value for the IS*TENT column; that is, whether or not the customer bought a tent. After we choose IS*TENT as the column to predict, AutoAI analyzes the data and determines that the IS_TENT column contains True/False information, making this data suitable for a binary classification model.

The default metric for a binary classification is ROC/AUC. After we click Run experiment, an infographic shows the process of building the pipelines as the model trains. Once the pipeline creation is complete, we can view and compare the ranked pipelines in a leaderboard. The pipelines for the sample binary classification model are quite uniform because of the underlying sample data. To see pipelines in action, re-run the experiment as a regression experiment to predict purchase amount.

That experiment gives better variation in the resulting pipelines. After clicking "Pipeline comparison," we can see how the pipelines differ on various measures of model quality. The pipelines can be saved as Machine Learning assets in the Watson Studio project. Then they can be deployed and tested. Currently AutoAI is available only for classification and regression models; there is a plan to add time series model support in the future.

In this unit, you have learned how AutoAI automates typical data science tasks and helps get better performing data pipelines more quickly, while also simplifying pipeline deployment into production in Watson Machine Learning. In the next section, we will discuss Watson OpenScale, which helps to ensure that your models are fair, explainable, and up to date.

IBM Watson OpenScale

IBM Watson Openscale is a product that includes several important features. It can **test the model and its predictions for fairness and apply ways to overcome bias**. It can also help to

provide explanations for model predictions that are often hard to get but are necessary for compliance in some application areas. It monitors the model performance and can detect its deterioration or model drift over time. It can alert the users when drift is detected and explain which predictors are causing it.
We can specify criteria under which the model gets automatically retrained on fresh data; it also helps to measure how the model helps the business. The attributes to monitor for bias are automatically recommended based on prior experience. They can be edited as needed. Openscale then keeps track of model predictions for the specified groups and checks the bias in the predictions. Users need to know that their AI models are fair but the date of their models were trained on and include unwanted **biases**\a which may unintentionally be included in the resulting models. IBM Watson Openscale can detect bias when a model is in production and not just when it's being built.
In this demo of Watson Openscale we'll monitor a credit risk model which has been trained to determine whether or not someone is eligible for a loan, based on a variety of different features, such as their credit history age and their number of dependents. After launching Openscale we can see a few highlighted metrics for the monitored model, such as its quality and a fairness score.
What Openscale does is **measure** a model's **fairness** by calculating the difference between the rates at which different groups, for example, women versus men, received the same outcome. A fairness value below 100% means that the monitored group receives an unfavorable outcome more often than the reference group. In this case, we see that women are receiving the no-risk outcome, or getting approved for loans, at a lower rate than men.
Openscale enables the inspection of each model's training data and this reveals that there was more training data for men than women. This can give some insight as to why the model exhibits bias against women who apply for loans. Data scientists can use this information to approve the model. Now, detecting bias is one thing-- Openscale can also mitigate it by creating a D bias model that runs alongside the monitored one. In this case the D bias model is 12% more fair than the production model. The D bias model has been trained to detect when your production model will make a bias prediction so that you can isolate the specific transactions that result in the bias. For each of these transactions Watson Openscale will flip the monitored value in a record to the reference value, in this case from female to male, and leave all other data points in that record the same. If this changes the prediction from risk to no-risk then the D biased model will surface the no-risk outcome as the D biased result. This is just one of the ways that Watson open scale helps you ensure that your models are fair explainable and compliant wherever your model was built or is running.
Insurance underwriters can use machine learning and Openscale to more consistently and accurately assess claims risk, ensure fair outcomes for customers, and explain AI recommendations for regulatory and business intelligence purposes. Why does an AI model arrive at a given recommendation or prediction? Users and customers want an explanation and

with most models providing this information is not an easy task.
IBM Watson Openscale explains predictions in business friendly language. This credit application, for instance, was predicted to be a risk. Openscale determines the features which contributed positively or negatively to that prediction and spells them out. The explanation is presented visually, as well as in a sentence-based text summary in order to ensure maximum clarity.
Using proprietary IBM research technology, Openscale also generates a contrast of explanations. Here we see the minimum changes to this input record which would produce a different output, changing the prediction from risk to no-risk. The explanations provided by Watson Openscale can help organizations comply with regulations such as the Fair Credit Reporting Act and GDPR which give customers the right to ask for reasons why their applications were denied.
Before an AI model is put into production it must prove it can make accurate predictions on test data, a subset of its training data; however, over time, production data can begin to look different than training data, causing the model to start making less accurate predictions. This is called drift. IBM Watson Openscale monitors a model's accuracy on production data and compares it to accuracy on its training data. When a difference in accuracy exceeds a chosen threshold Openscale generates an alert. Watson Openscale reveals which transactions caused drift and identifies the top transaction features responsible. For instance, 25% of a transactions causing drift in this loan approval model were problematic because of these features, which contained data crucially different from the training data.
The transactions causing drift can be sent for manual labeling and use to retrain the model so that its predictive accuracy does not drop at run time. Watson Openscale not only helps identify drift but also highlights its root cause and provides transactions which can be turned into training data useful at fixing drift. It gives you the insight you need to ensure that your models will consistently deliver the results you want over time. For instance, the retrain version of the model, but based on the recommendations made by Watson Openscale, started making accurate recommendations alleviating the drift.
This is just one of the ways that Watson Openscale helps you ensure your models are fair explainable and compliant wherever your model was built or is running. In this video you have learned how Openscale ensures fairness and explain ability of models and monitors for model drift in production. This completes the model on IBM products for data scientists. Good luck on the quizzes!