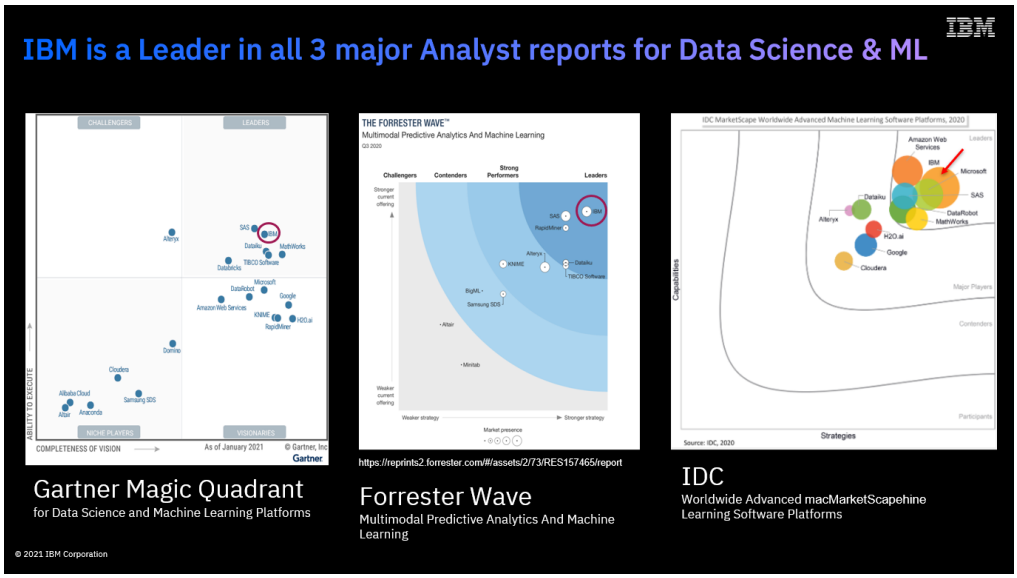
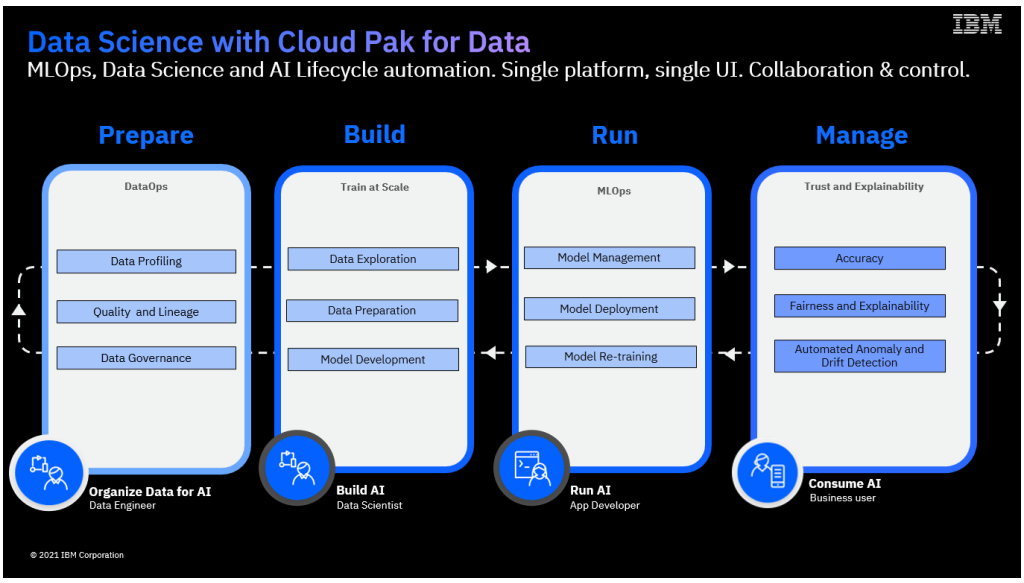
Use-Case

You are a data scientist employed by ABCOnlineRetailXYZ Ltd. The online retailer has not been doing that great lately and has been losing customers to competition. While it is clear that its customer churn rates have been growing, figuring out which customers are more likely to churn is proving hard.

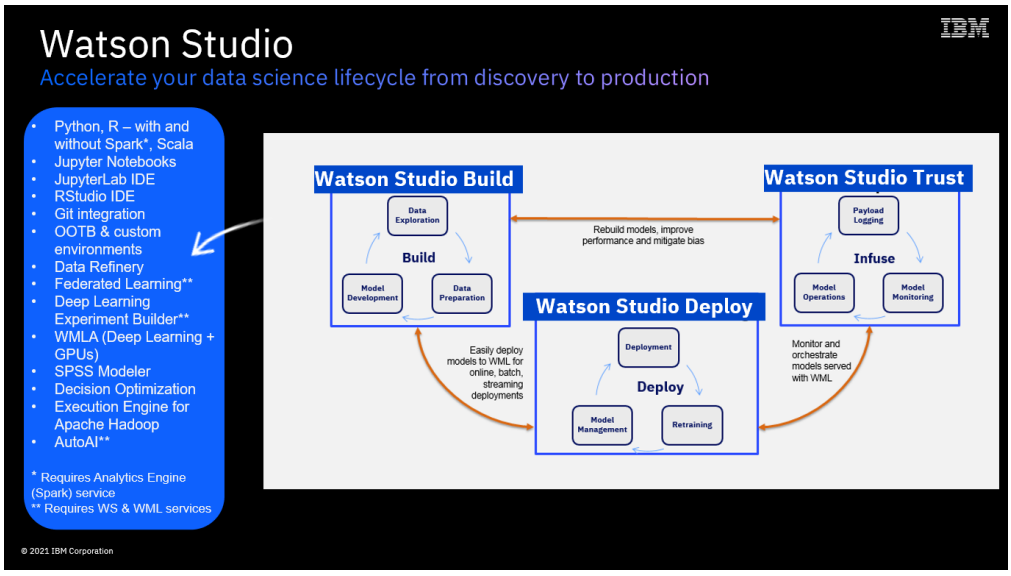
Your manager came by your desk today, gave you a couple of files containing some customer-related data and asked you to build a model helping the business understand which of its customers may churn soon, so that Marketing could proactively target them with personalized incentives and discount vouchers.
You have a lot of other work to do and are a short on time and are in need of inspiration – your Python skills are a bit rusty.
You are aware that IBM provides market-leading data science tooling and decide to use IBM's Watson Studio running on CPDaaS to build and test your model.



IBM's Cloud Pak for Data provides a range of capabilities catering for the whole data science life cycle – from data preparation to model build, deployment, training, and monitoring.



The platform's Data Science and ML component(s) sit under the Watson Studio branding umbrella. Cloud Pak for Data's Watson Studio services (Watson Studio / Watson Studio Build, Watson Machine Learning / Watson Studio Deploy, Watson Openscale / Watson Studio Trust) enable the Build, Run and Manage capabilities of the MLOps lifecycle.



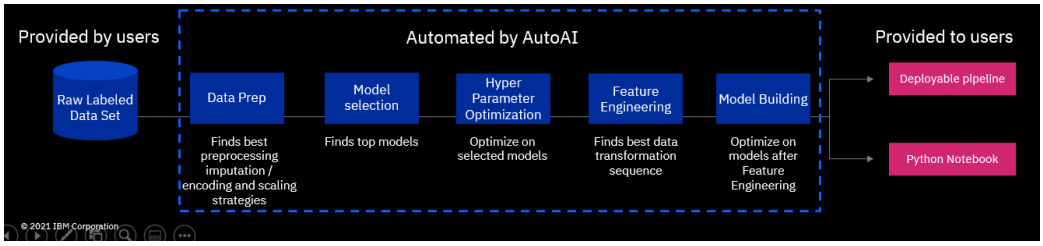
In this Lab, you will:

- Use AutoAI to automatically build the most optimal customer churn prediction model based on your data sets
- And, finally, deploy and test that model with Watson Machine Learning (Watson Studio Deploy capabilities).

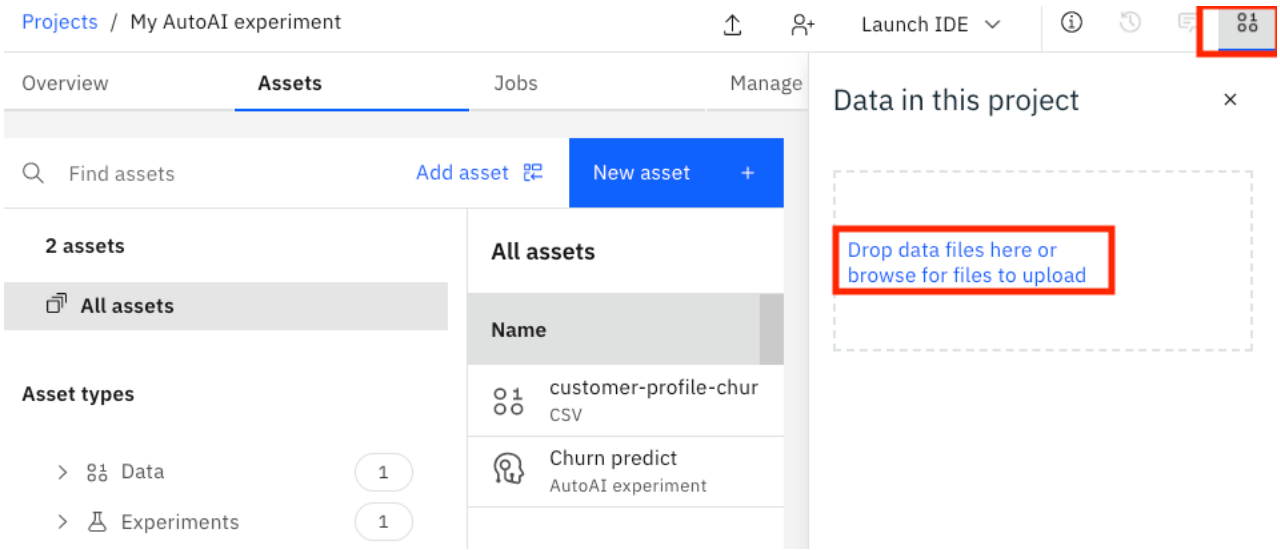
… and, your model will be built (for you!) and deployed in minutes - not hours or days!

Building Models with AutoAI

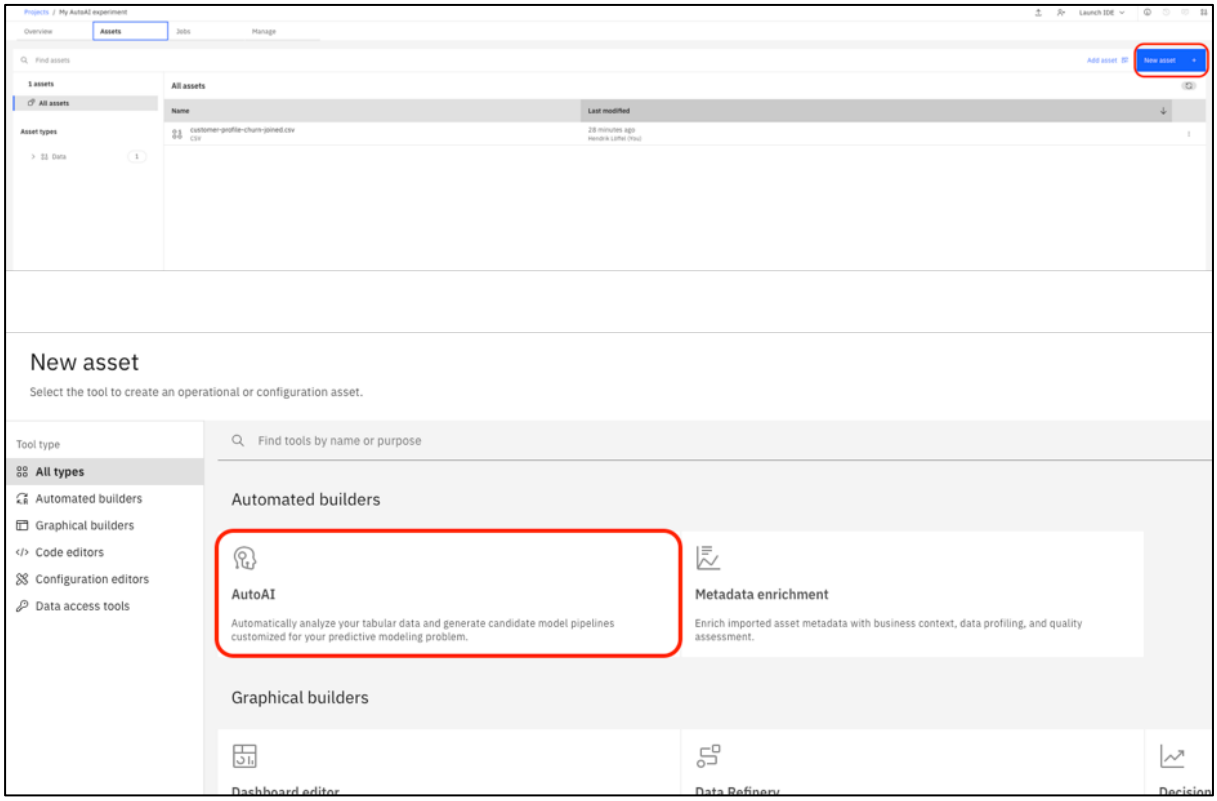
In this section, we are going to build a machine learning model using AutoAI. The AutoAI graphical tool in Watson Studio automatically analyses your data and generates candidate model pipelines customized for your predictive modelling problem.
These model pipelines are created iteratively as AutoAI analyses your dataset and discovers data transformations, algorithms, and parameter settings that work best for your problem setting. Results are displayed on a leader board, showing the automatically generated model pipelines ranked according to your problem optimisation objective.



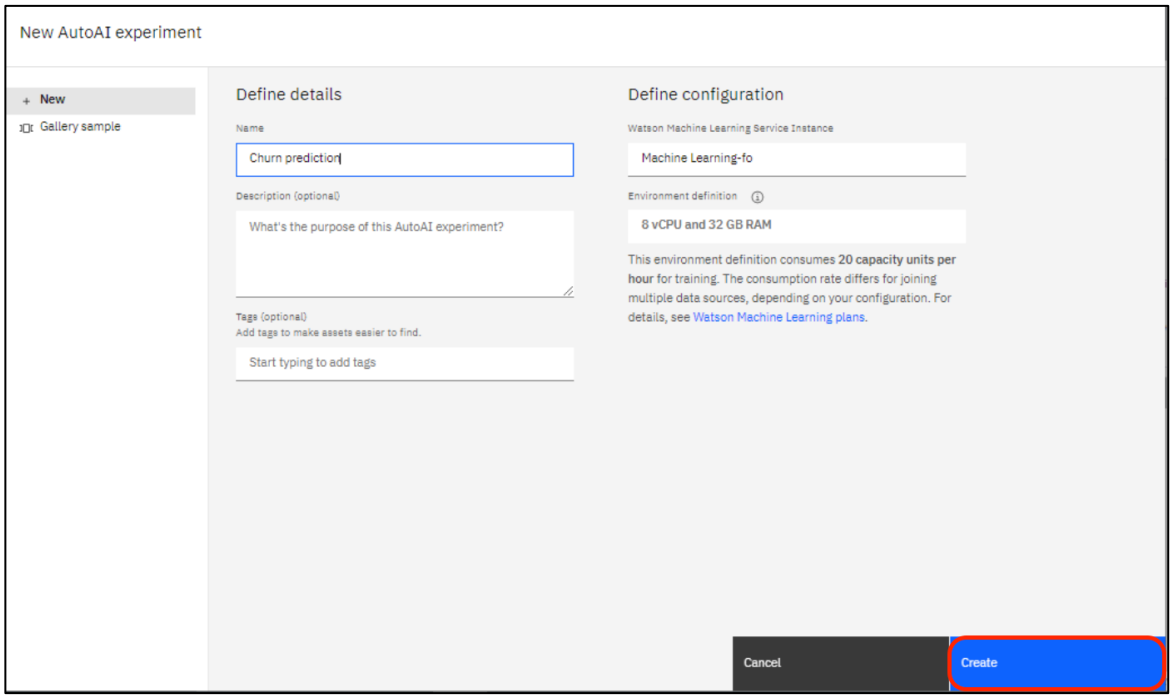
Before starting with AutoAI make sure the sample data has been added to the project - please download the .csv file from the github page called “customer-profile-churn-joined.csv” to your local machine and then to your project (click the “01” button to the upper right to add data to a project).



Click the Add to New Assets button and add a new “AutoAI” experiment.

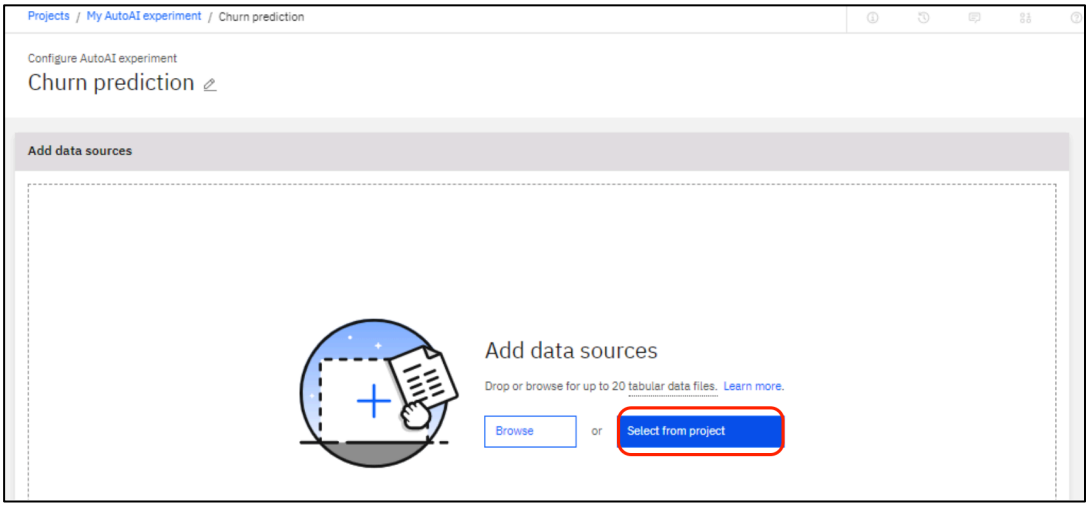


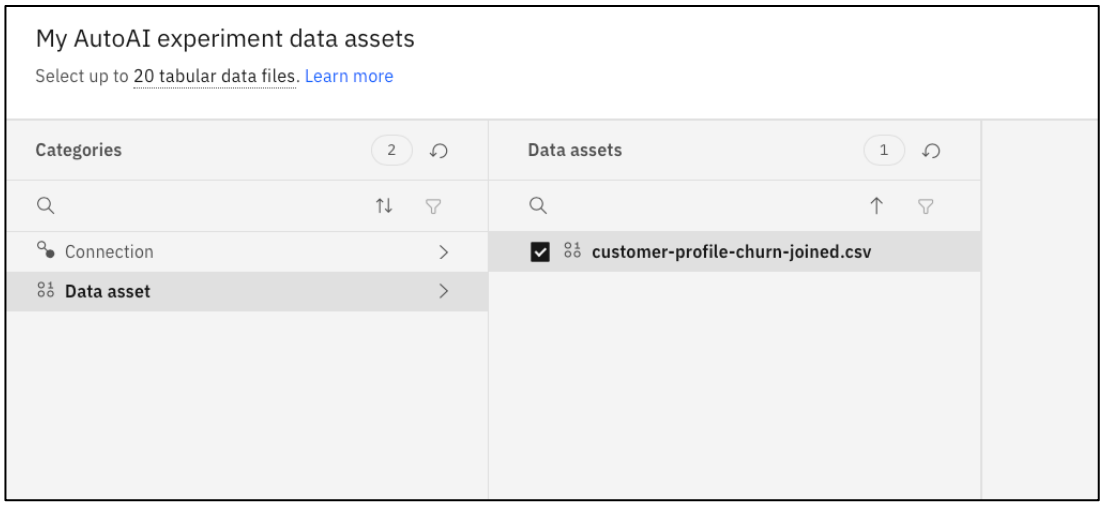
Give it a name – for instance “Churn prediction”, click “Create” to move to the next step.



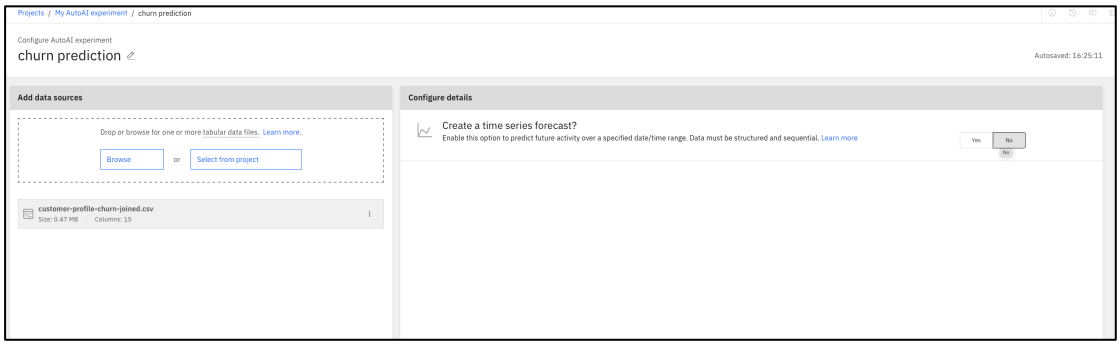
In the next step, we have to add the data source we want to use for training and assessing our AutoAI solution.

On the next screen, click on “browse” and add the csv file you downloaded in the previous step to the project.

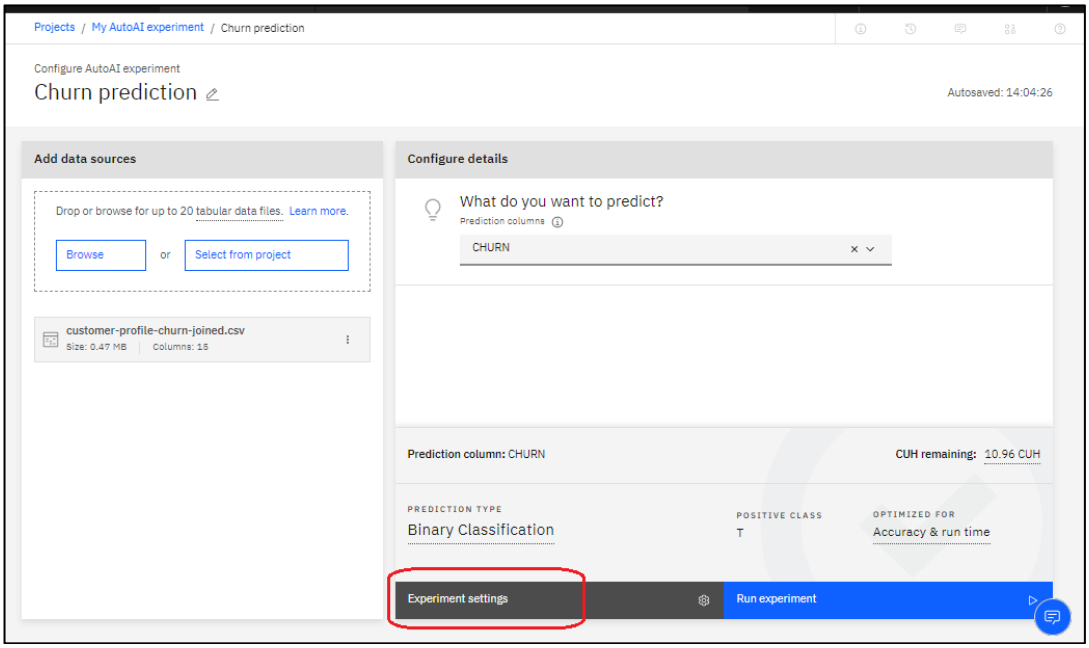




On the next screen, please click “No” when you are asked if you want to create a time series forecast.



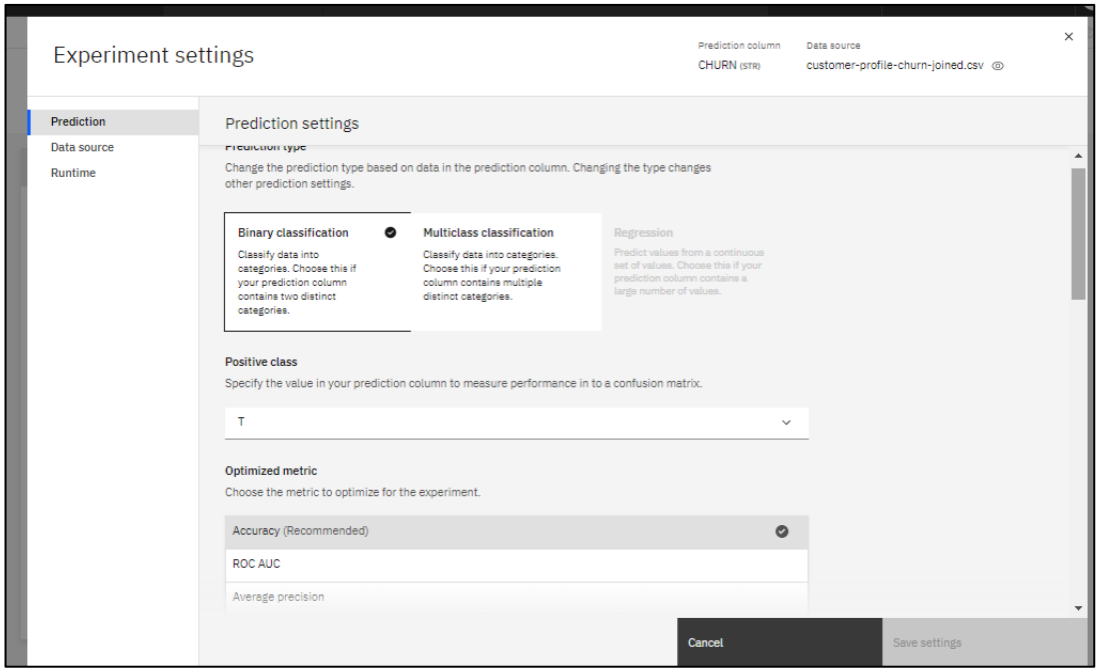
On the next screen we will need to let AutoAI know what we would like to predict (our target). In data science terminology the columns that we use to predict churn are called features, and the column that we are trying to predict is called target.



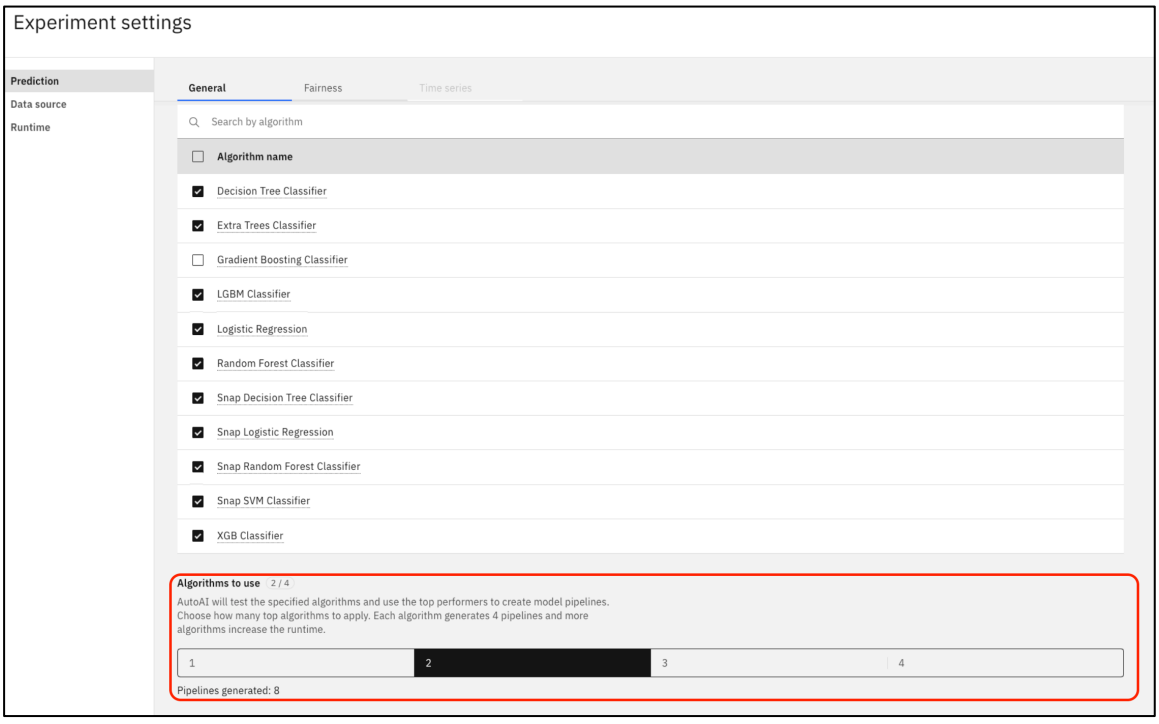
AutoAI will switch to Binary Classification (CHURN is a yes/no choice – or, more precisely True/False – the customer will either leave us or will not).

Let's explore additional settings – click Explore Settings. Feel free to explore the settings on the screen. However, due to time constraints, please don't change any of the default settings. When you have finished exploring the parameters, exit the "Experiment Settings Page" with "Cancel".

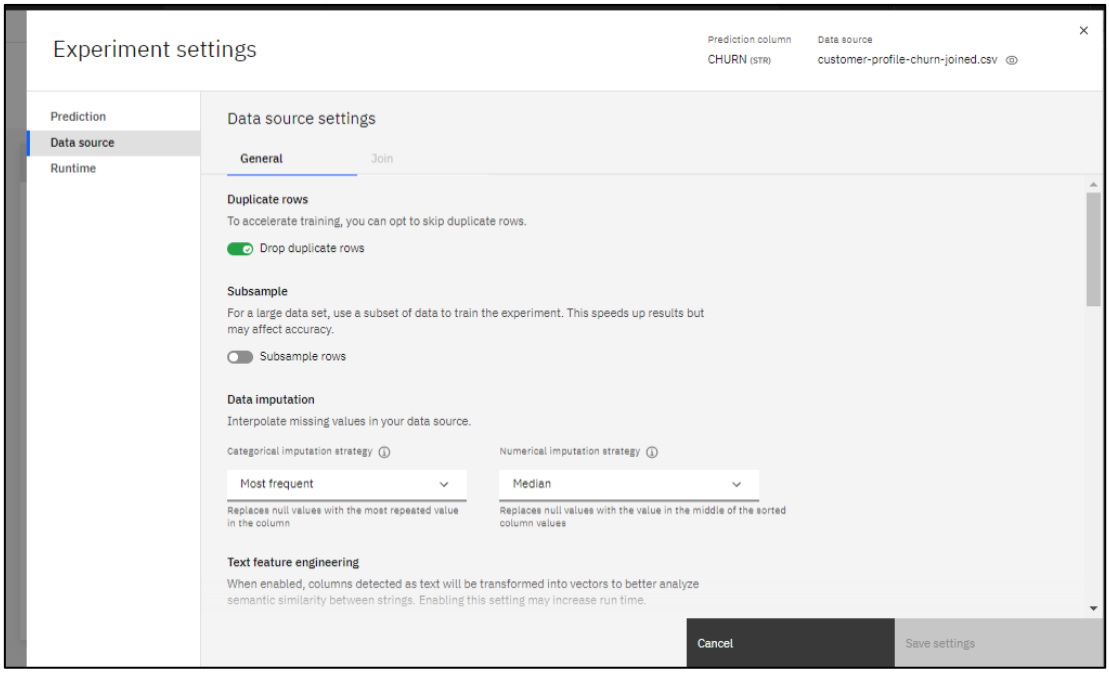
Positive class T (True) makes sense in our case as we are looking to predict which customers will actually churn.



The AutoAI set up allows you to specify how many pipelines (models) you would like to build. By default, 2 top performers out of the 7 available algorithms will be used to build a total of 8 pipelines (4 for each algorithm).



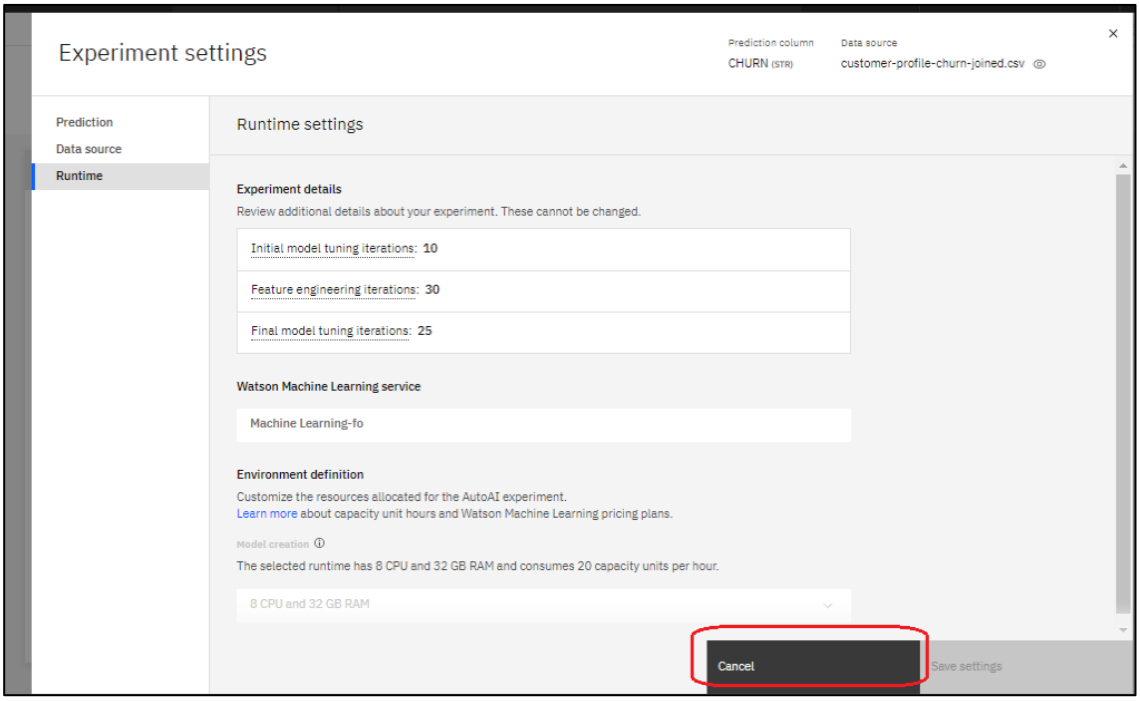
Next, switch to the Data Source tab from the menu on the left. Explore the settings. An explanation of the most important settings can be found subsequently to the following picture.



When creating a model, the data is split into training and testing data. Testing data is also called "holdout data". By default, 10% of data is used for testing. Let's leave the default value. *Folds* are parts of the input dataset that will alternatively be used for training and testing. By specifying 3 folds, we are creating 3 parts in the dataset. We will use the default value of folds in this lab.

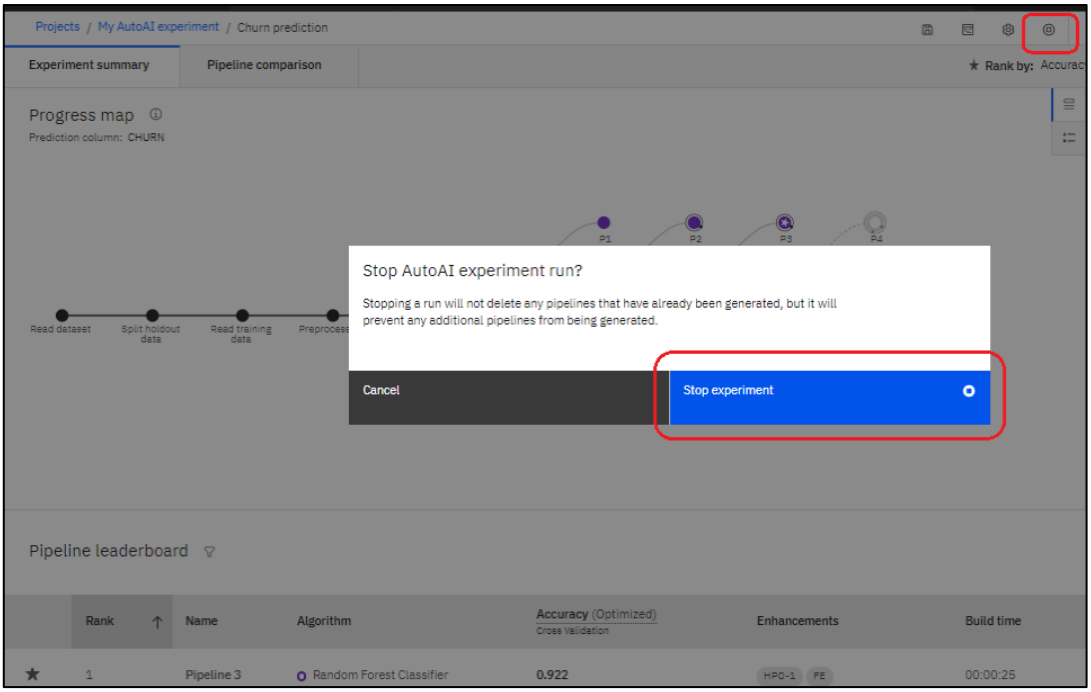
When needed, we can also deselect fields to be used for training. In our case, let's keep all of the fields.

Finally, switch to the Runtime tab and review the screen. Once done, click cancel (since you haven't changed any of the default settings) to quit the experiment settings.

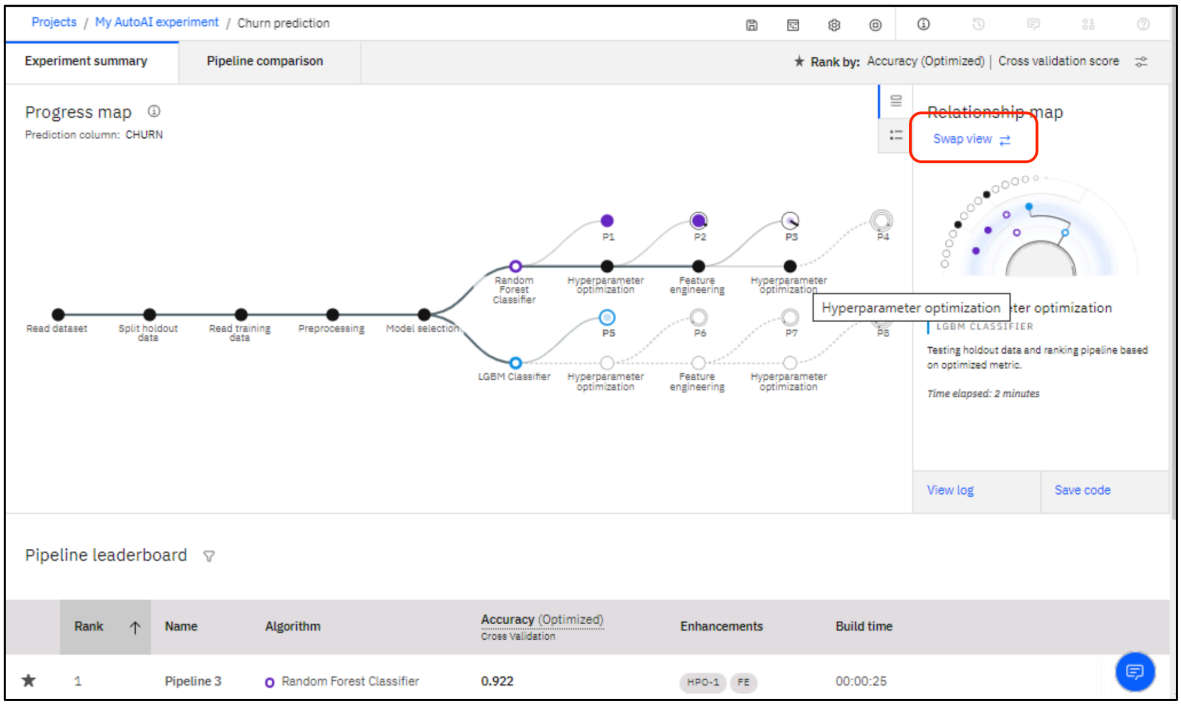


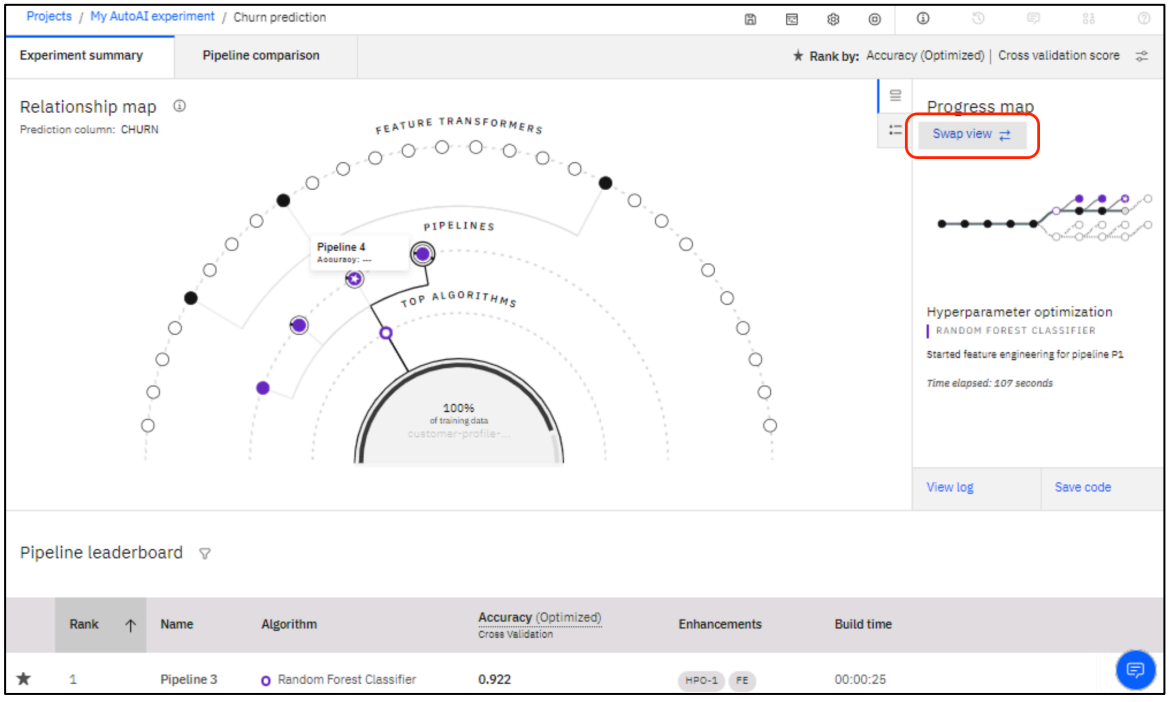
Finally, click “Run experiment”.

Watch AutoAI do its magic. This process will take approx. 2 minutes. If the experiment is taking more than 5 minutes to complete, in the interest of time, please stop the experiment manually. It will keep all the already built models but will not generate any new ones.



Once the process completes, click on “swap view” to see the following charts:





A *pipeline* (in the following abbreviated with *P*) is a term used to describe various steps in creating a model. A *P* always contains the generated model. It can also contain steps to generate new columns of data to use as input variables (*features*) and steps to tune the model (*hyperparameter optimization*).

As we can see in the example, *P1* and *P5* build a pipeline using the *estimators* (algorithms) which AutoAI has determined to be the best fit for the data, in this case *Snap Random Forest* and *XGB Classifier.*

P2 and *P6* perform hyperparameter optimisation (HPO). Hyperparameters are “settings” (parameters) that are specific to each algorithm. Hyperparameter optimisation means that we are building the model using different settings in the algorithm. *AutoAI* tries several combinations and determines the combination which will produce the best result.

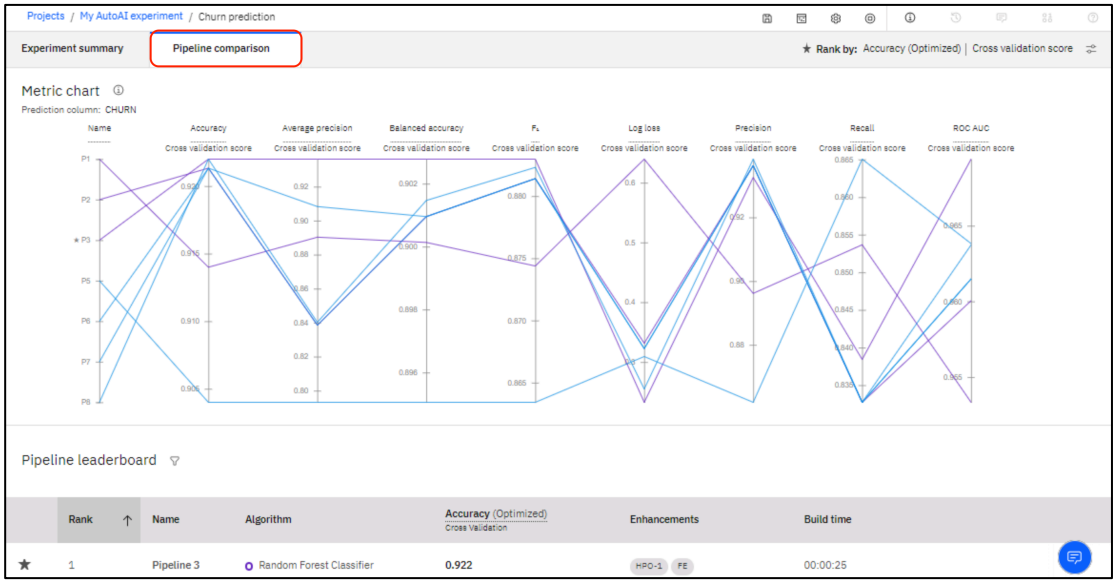
P3 and *P7* perform feature engineering (derives new features) and build a model with these features.

P4 and *P8* perform hyperparameter optimization for the model that uses the derived features.

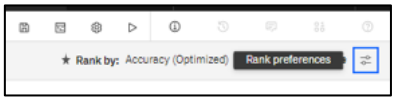
In summary, for each algorithm AutoAI creates 4 pipelines:

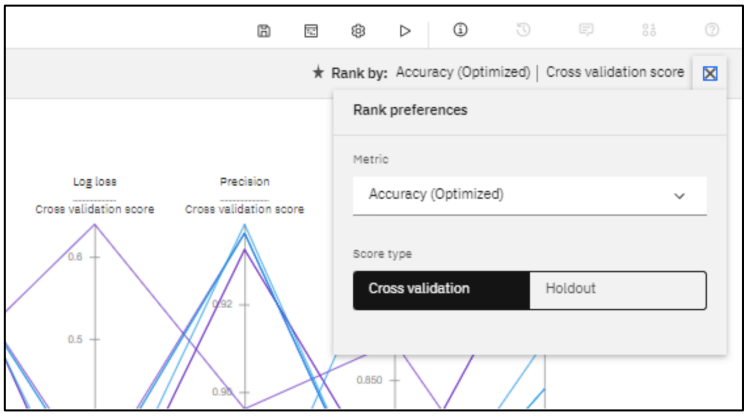
- *P* 1 contains just the model
- *P* 2 contains the algorithm and performs HPO
- *P* 3 generates additional features and builds a model that includes these features
- *P* 4 generates the same features as pipeline 3 and performs HPO.

Once the experiment has finished or has been stopped, review the built pipelines and explore the metrics the platform generated. Switch to the Pipeline comparison tab.



Use the Rank preference button (top right on the screen) to switch between different evaluations metrics that are used by data scientists.





Score types: The *Holdout Score* is the score for the *test* dataset that was not used for modeling (by default, 10 percent of the data was left out for testing). The *Cross Validation Score* is the score when the cross validation technique was used. Cross validation uses different parts of the same dataset first for training, then for testing.

An experienced data scientist may look at a combination of evaluation metrics before determining if the model is ready for production, as these measures provide information on how the model performs in difference areas of "good-ness".

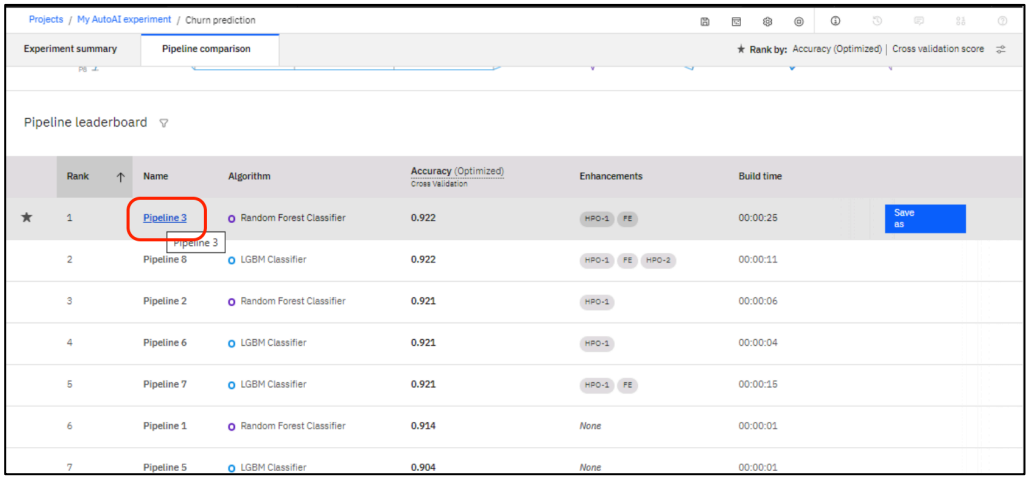
For example:

- Precision tells you: "Out of all the records in the hold-out sample that the model predicted to be Churners (T), what percentage actually churned…". For Pipeline 1, from all the records that the model predicted were churners, 0.896 (89.6%) were actual churners.
- Recall tells you: "Out of all the churners in the hold-out sample, what percentage can your model actually find?". For Pipeline 1, the model managed to correctly identify 0.853 (85.3%) of the total churners in the hold-out sample.

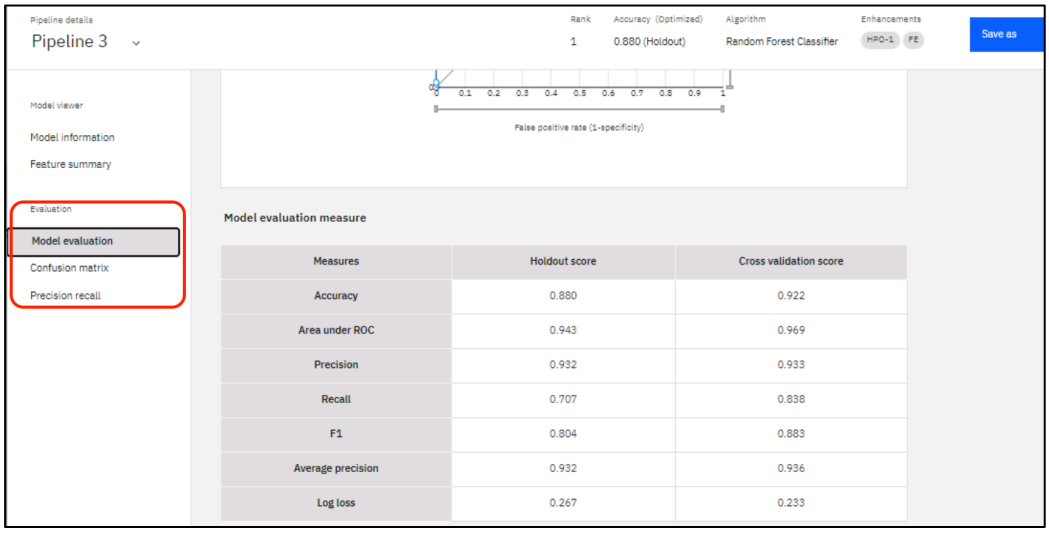
Many data scientists use *Area Under ROC Curve.* As a rough guide, you can use the following guidelines for *Area Under ROC Curve*:

- *.90-1 = excellent (A)*
- *.80-.90 = good (B)*
- *.70-.80 = fair (C)*
- *.60-.70 = poor (D)*
- *.50-.60 = fail (F)*

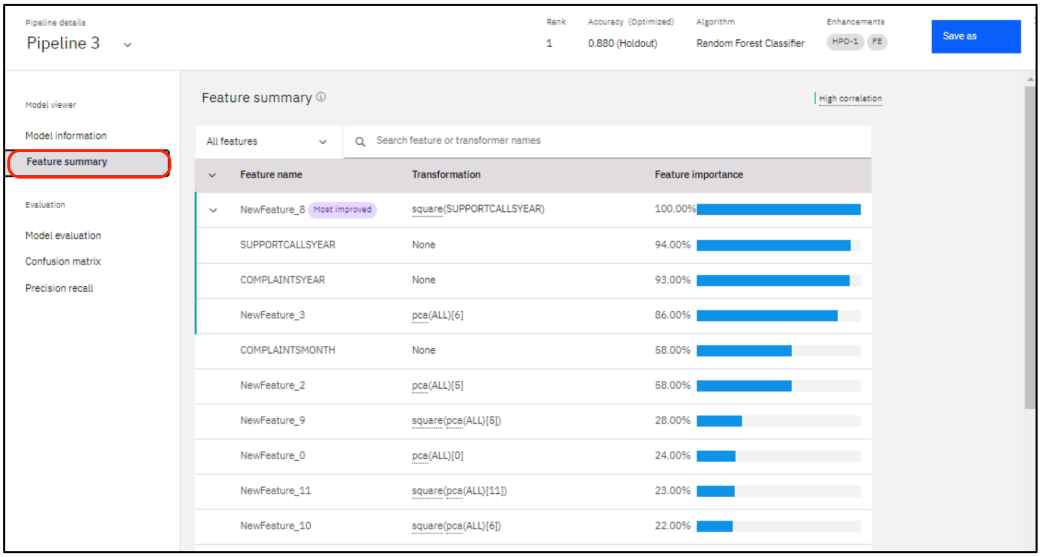
Now, scroll down and review individual pipelines. In our case, AutoAI flagged Pipeline 3 as the best performing based on Accuracy (Optimized). Click on the name of the pipeline to see further details. Note: Due to randomness, your results are likely to vary.



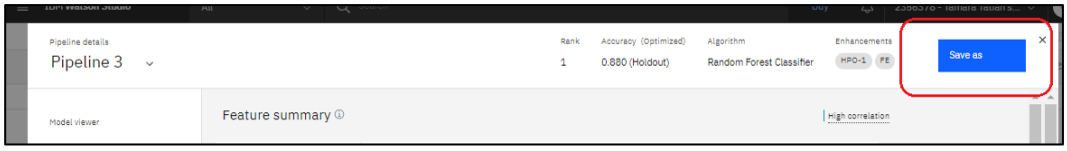
Review the details. Under “Evaluation”, you will find various metrics that are familiar to data scientists and help them to evaluate which pipeline has performed in the best way. In our case, we focus on the metric “accuracy” for the purpose of ease. Depending on the business context, other metrics (like precision or recall) can be more relevant.



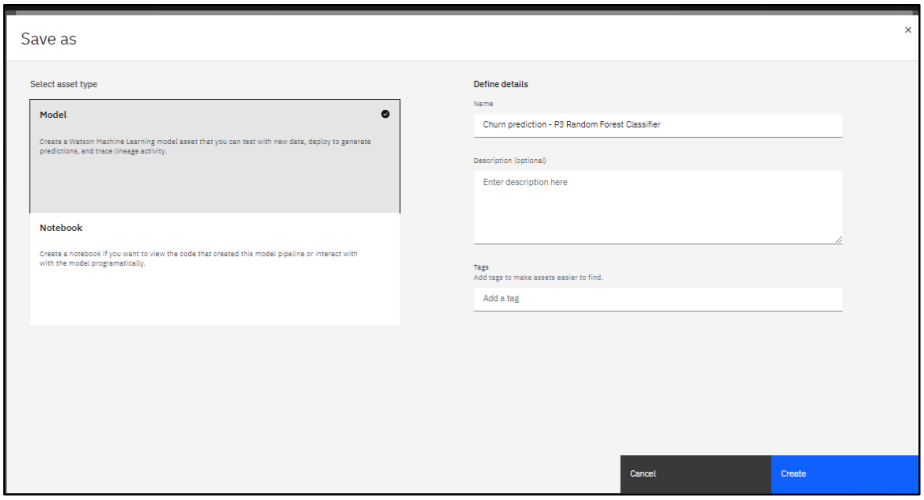
Another interesting aspect can be found under “feature summary”. Here, we can see which features make the most impact on the prediction.



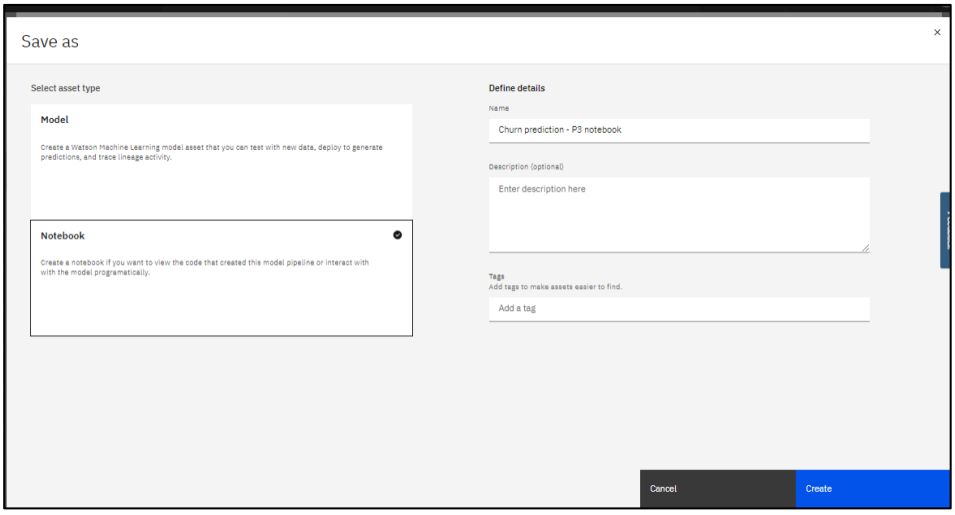
Let's now save the pipeline. Just click on one of the pipelines, and a "save as" button will appear – as shown below.



Please save the model as a "Model", not as a Notebook. You do not have to change the name- this is automatically generated for you.

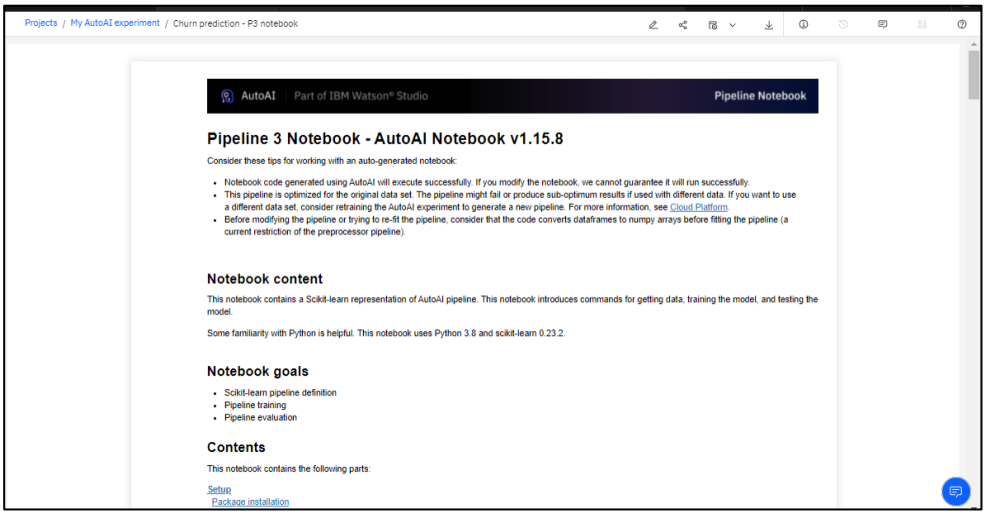


Repeat the same process. This time, please save it as a Notebook.

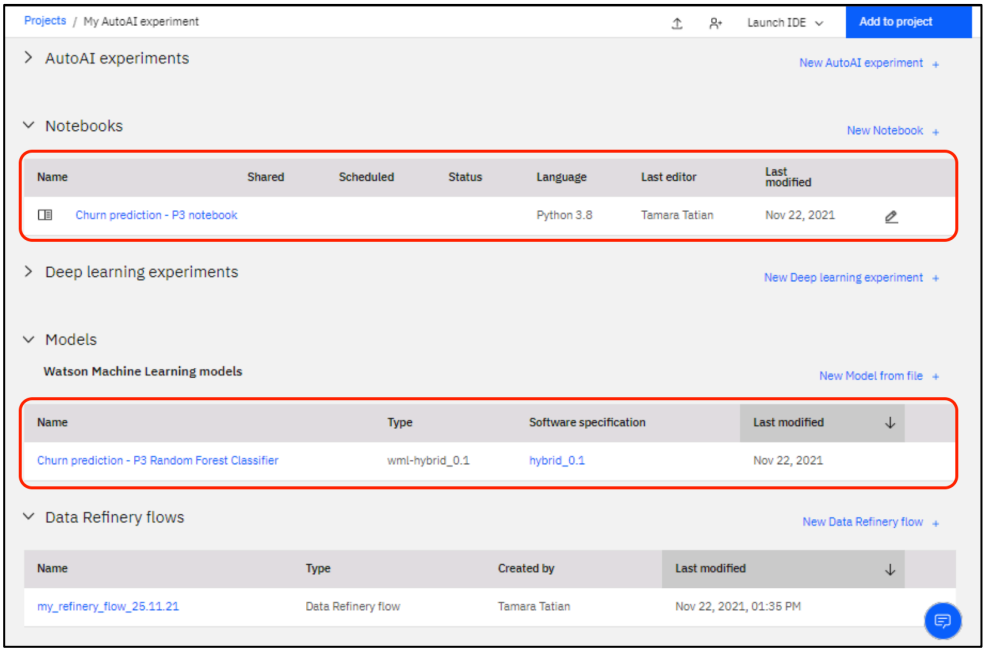


Click the “View in Project” link for the notebook you just saved.

Review the notebook – AutoAI is not a black box, it produces well annotated code that you can take as-is or choose to develop further.

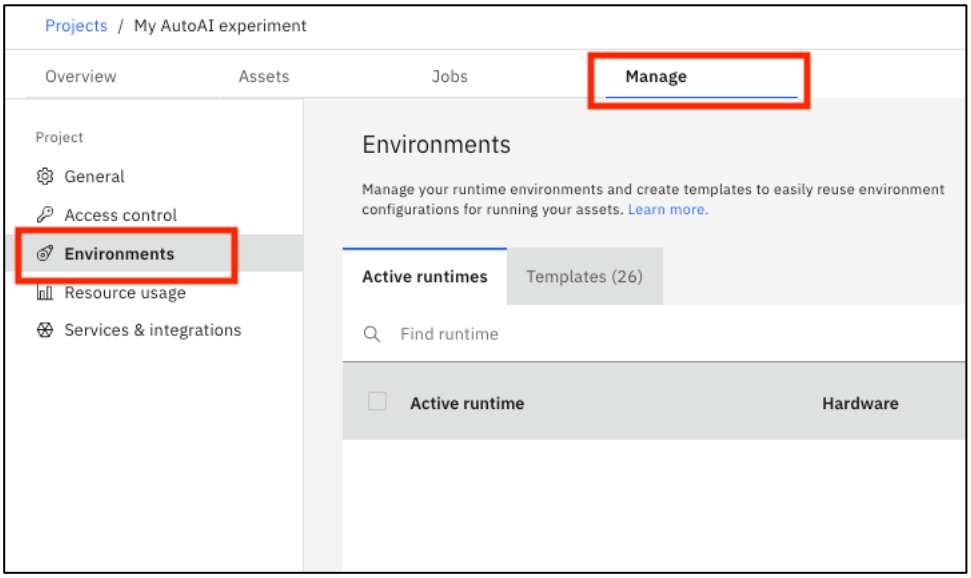


Next, navigate back to the project Assets view by clicking the breadcrumb with your project name (top left of the screen). Scroll down – note that you now have your saved model and notebook listed.



Before moving to model deployment, let's ensure 'good housekeeping' practices are followed and clean up any environments that may still be running against your project and account as they are no longer needed (this will also help minimize any potential charges and ensure you stay within the allotted CUH limits for your account type).

Please switch to the Manage tab and select Environments (see below) If there are any active runtimes still running there – please stop them at this point.



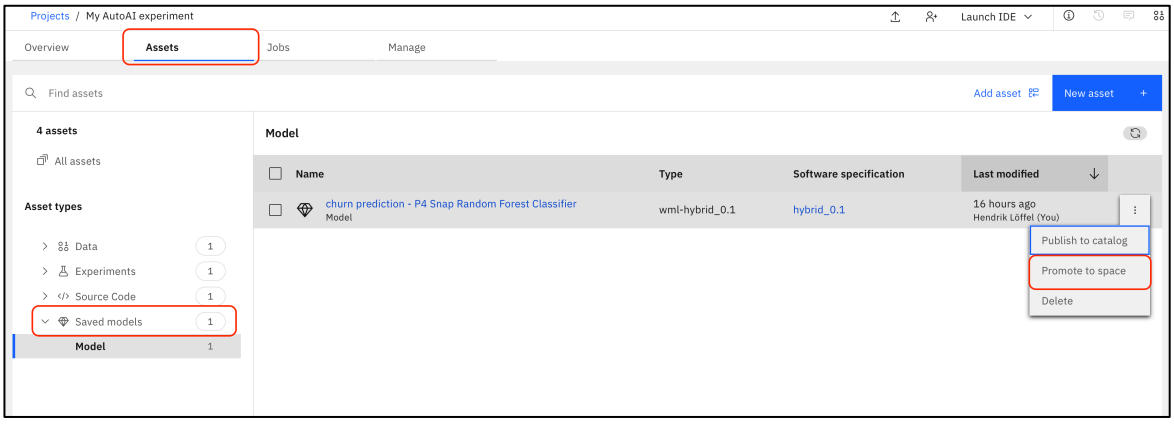
Deploying Your Model with Watson Machine Learning

Next, we are going to deploy our saved model. First, we will need to promote the model to a deployment space. Deployment spaces (another project/collaborative workspace type provided in CPDaaS) allow you to create deployments for machine learning models and functions and view and manage all of the activity and assets for the deployments, including data connections and connected data assets.

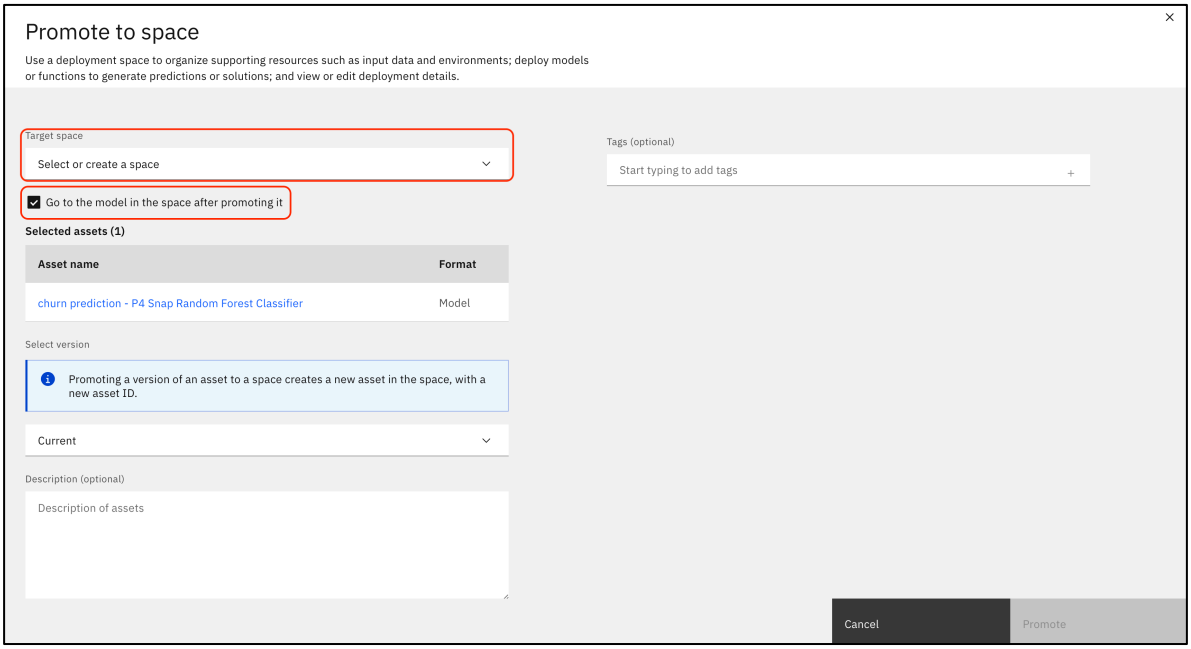
You can deploy assets from multiple projects to a space, and you can deploy assets to more than one space. For example, you might have a test space for evaluating deployments, and a production space for deployments you want to deploy in business applications.

Separate development and deployment environments provide better security and governance. Deployments are often configured by ModelOps team, and only authorised users are given access to *Deployment Spaces*.

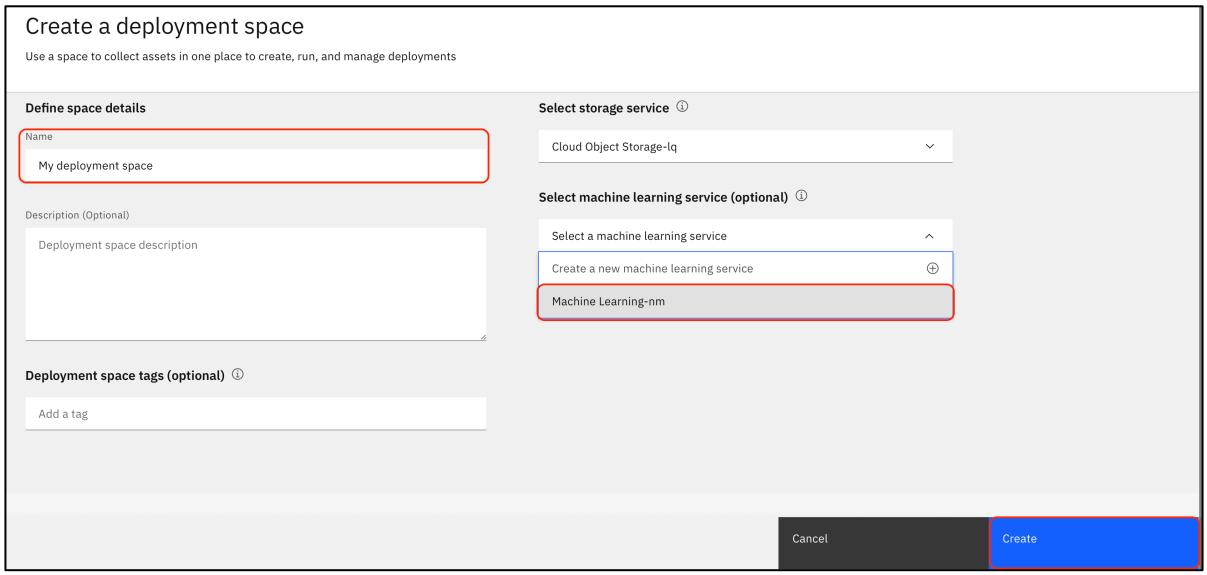
From the project menu, switch to assets and click on “Saved models”. From there, click “Promote to Space”.



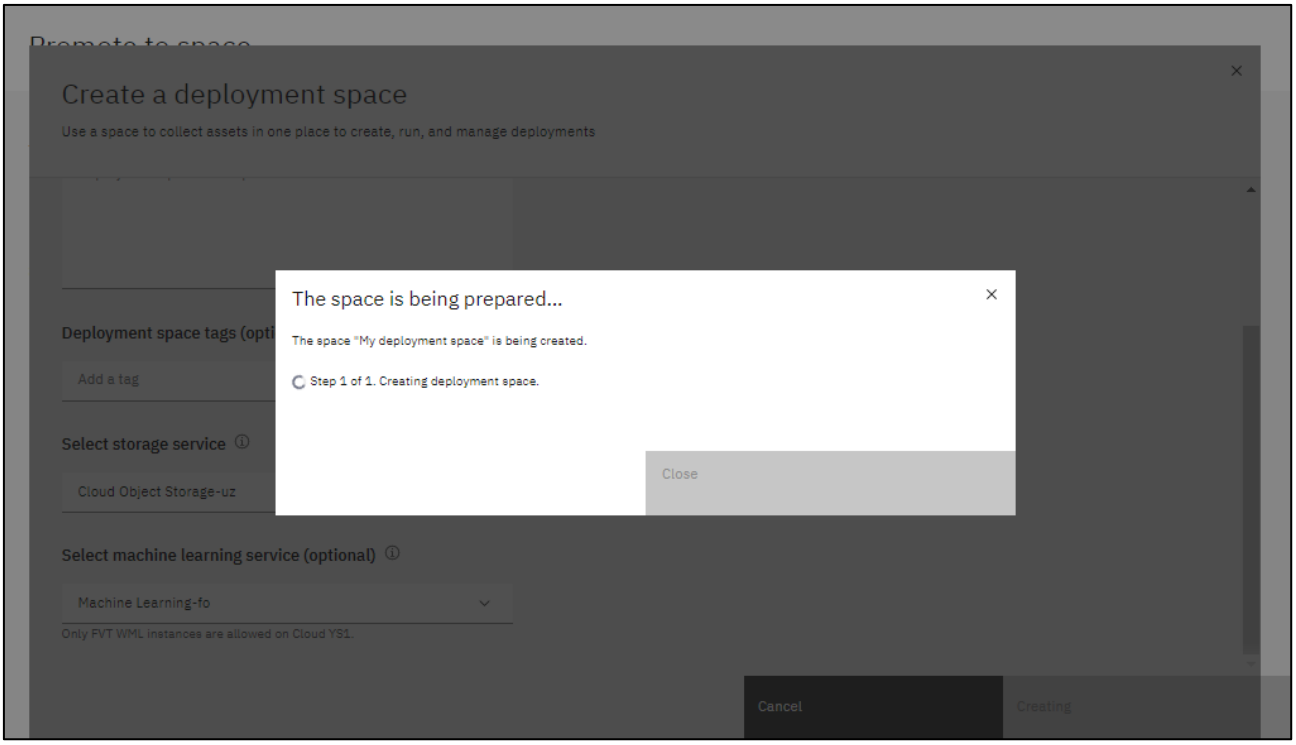
Check the box saying “Go to Model” on the next screen. Then, select from the drop down menu the option “Create a new deployment space”.



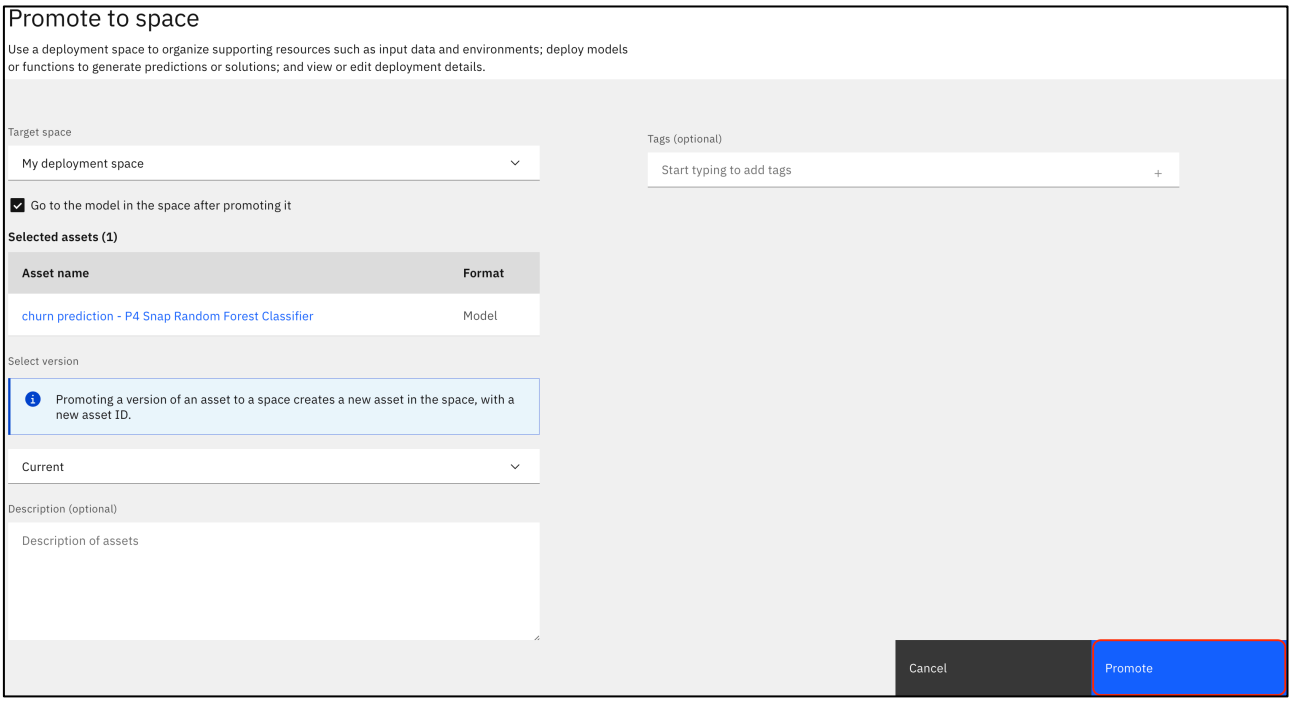
Create a new deployment space. Give it a random name - and make sure to select the machine learning service correctly.



By clicking create, you will see the following screen:

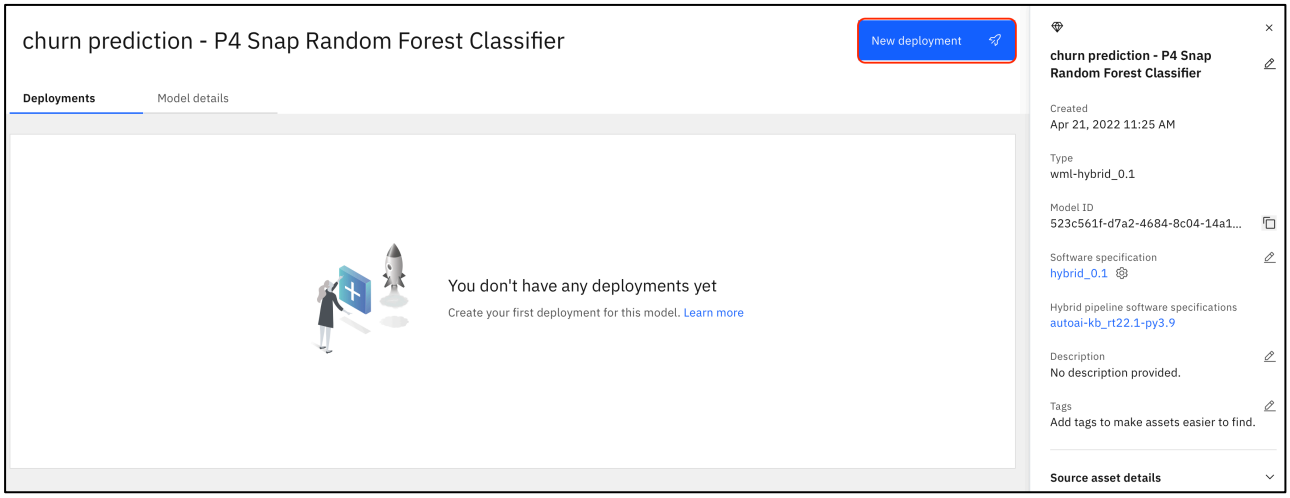


Once the space is ready, close the pane. Click “Promote” on the down-right corner.



Because we ticked the checkbox, the platform will take us straight to the promoted model.

Next, we will deploy the model. Click the “New Deployment” button.

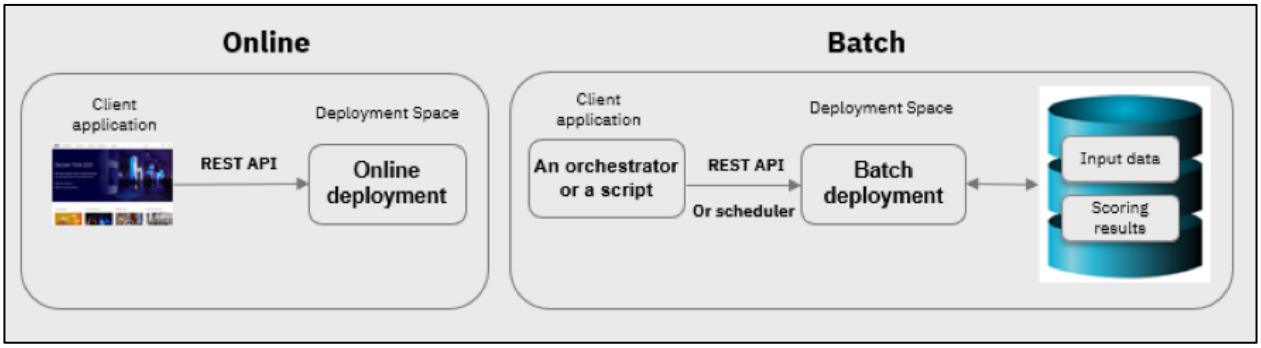


The types of deployment available for a model and the type of input supported depends on the model framework.

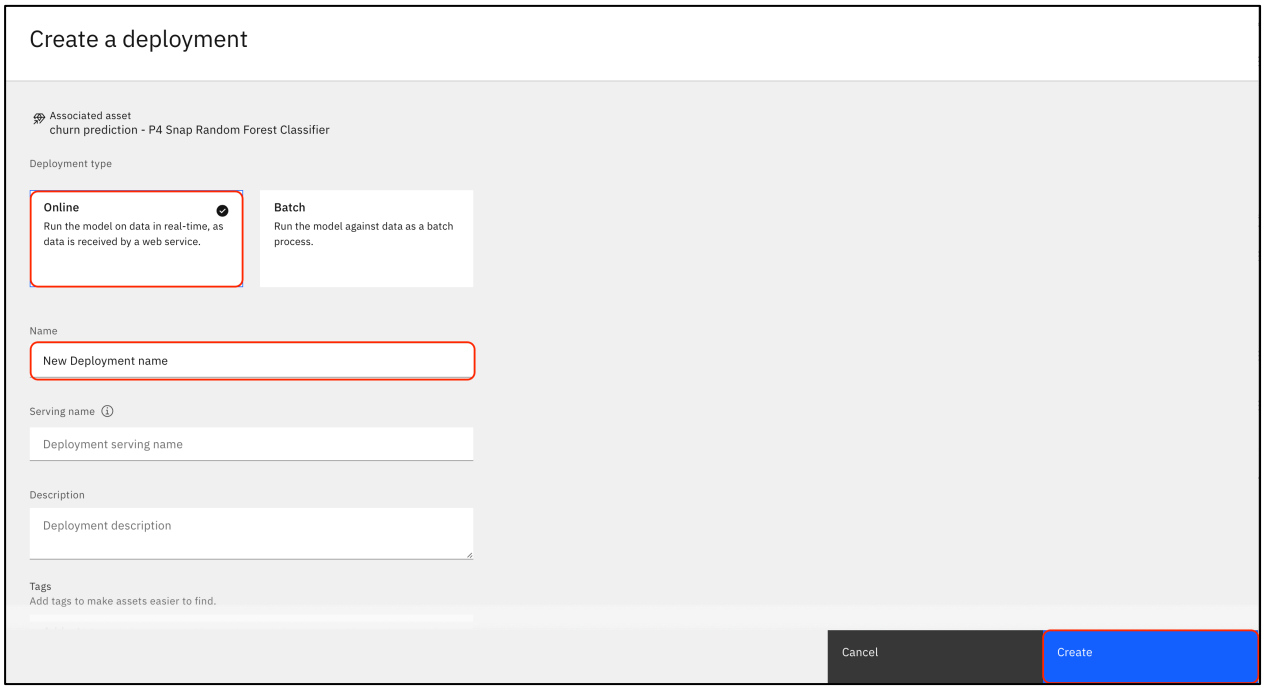
The types of deployments supported by Watson Studio are:

- Online - also called Web service, this deployment loads the model or Python code when the deployment is created to generate predictions online, in real time. *Online scoring* is used when the use case requires immediate scoring results. For example, a call center agent needs to know if a customer is likely to churn when answering a call. Online scoring is integrated with line of business applications using REST APIs.
- Batch - to process batches of input submitted from a data file, a connection to a data repository such as a database, or connected data in a storage repository such as a Cloud Object Storage bucket. Batch deployments can be run on demand or on a schedule. *Batch scoring* is used for a repeatable business process for all customers. For example, creating a weekly marketing campaign for customers who are likely to churn. Batch scoring is integrated via data layer: scoring results are written to a data source that's used by the line of business application.
- Core ML - downloads the code required to deploy on an iOS device.
- App - to make R Shiny apps available for use by users with a link to the endpoint.

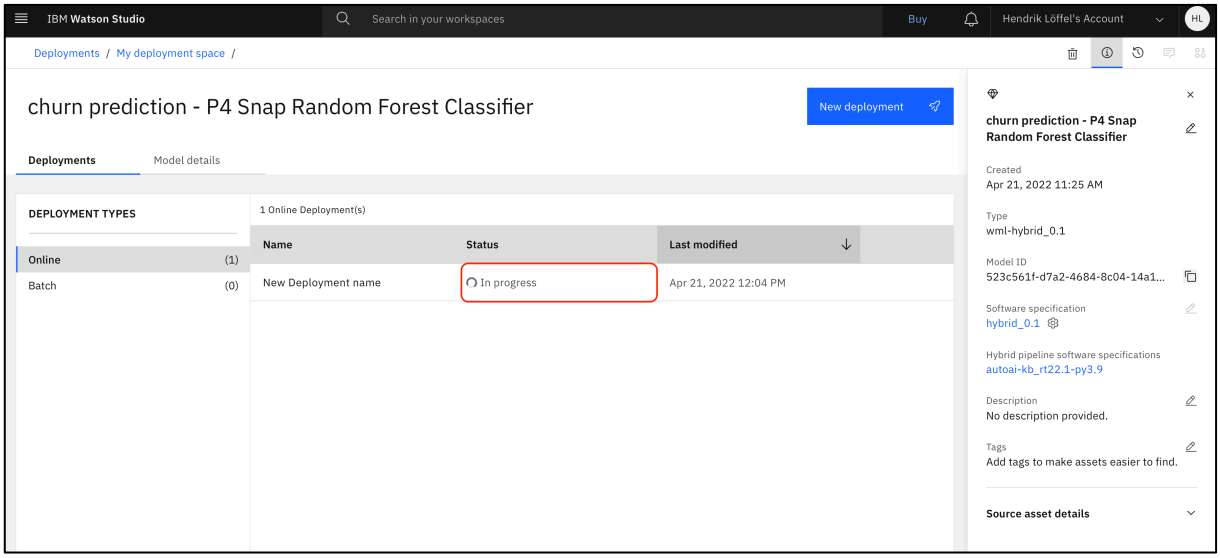
We can deploy our model for online or batch scoring.



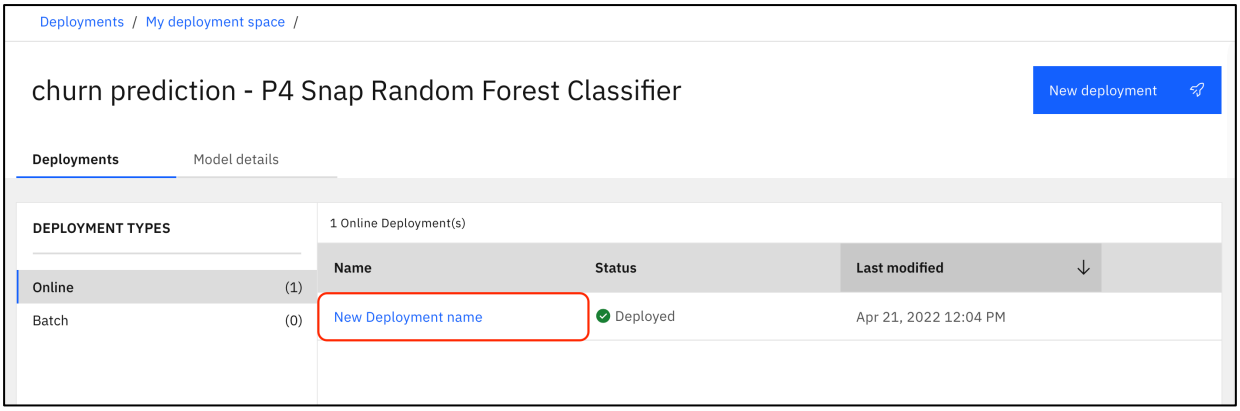
In this lab we will configure online scoring. Give your deployment a name, make sure Online mode is selected, then click “Create”.



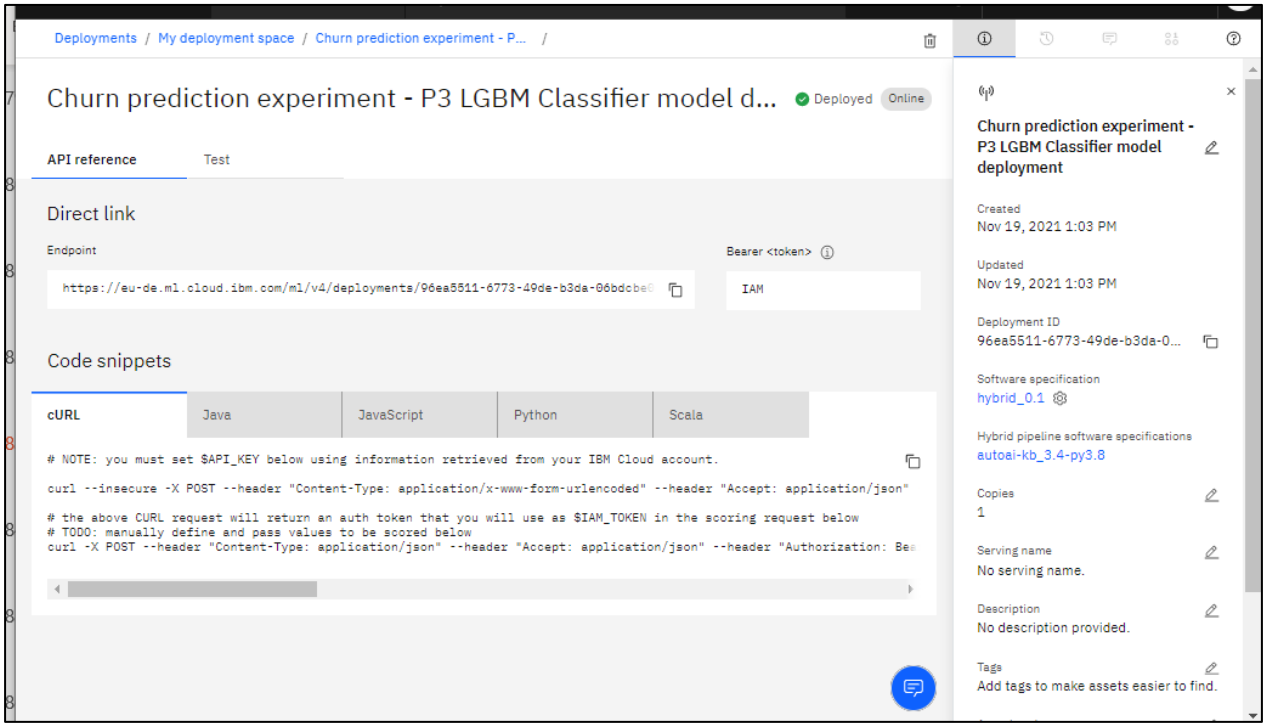
The model is now being deployed into a highly available container in Watson Machine Learning service. After a few minutes the *Status* will change from *In progress* to *Deployed*.



Once the deployment finishes, click the name of the deployed model to see further detail.

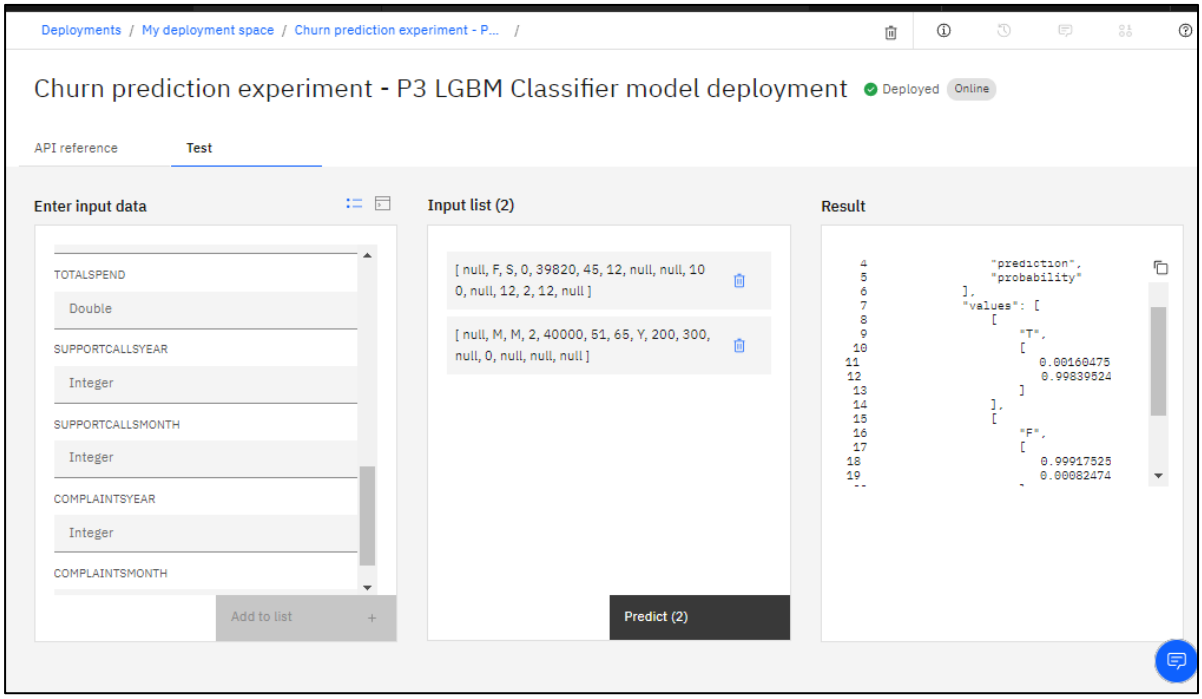


Review the code snippets and note that the system automatically generated a REST API endpoint for us.

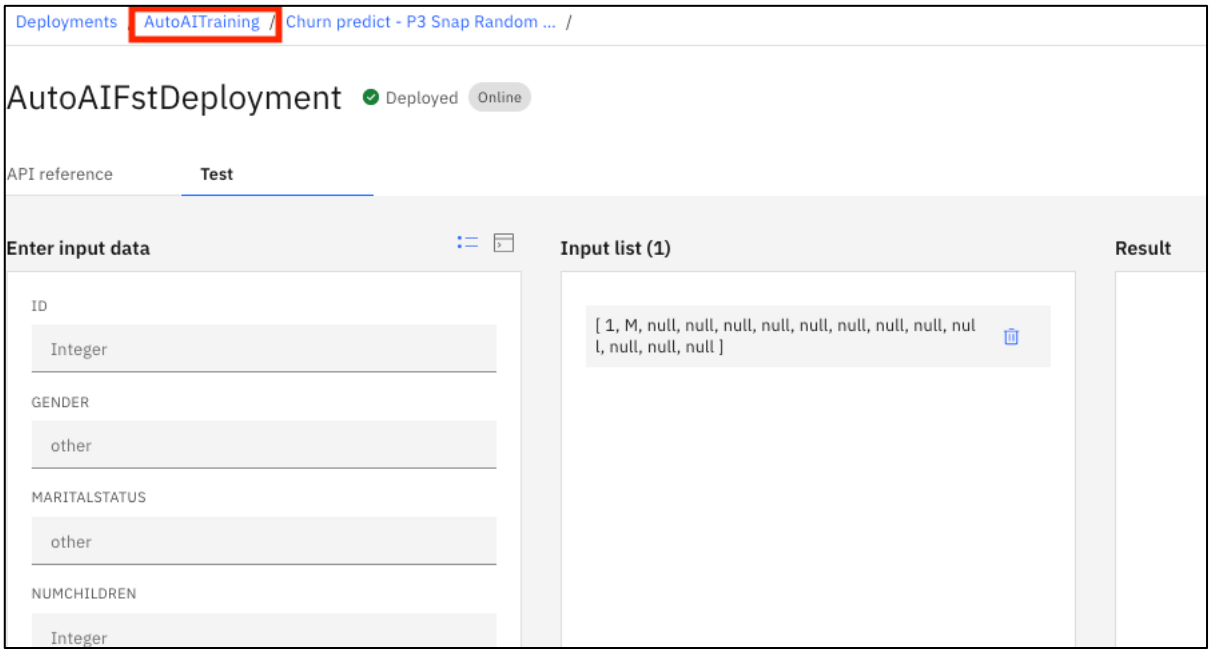


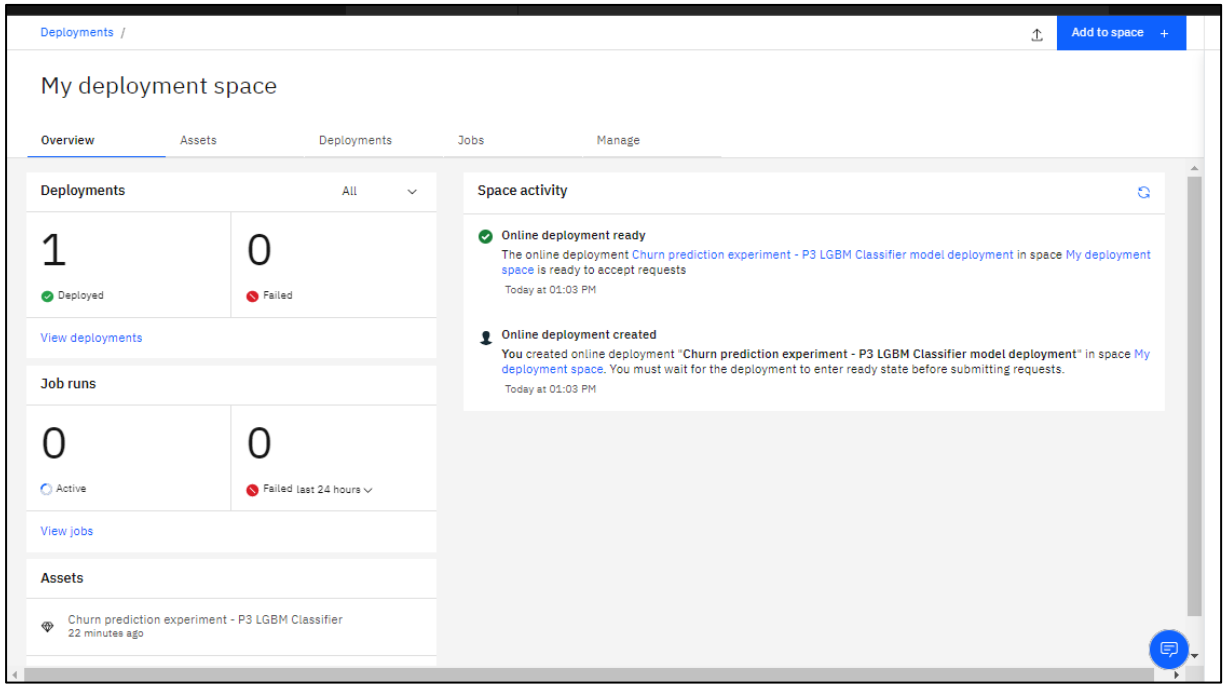
Navigate to the “Test” tab.

Enter some values into the input data fields on the left-hand side – note that you can add more than one test item. Click Predict – the API will return its prediction, along with the probability of a True outcome for the predicted CHURN metric.



Navigate to the deployment space using the main menu or the on-screen breadcrumbs (click the name of your space on the top left) to check metrics and statistics as well as general settings and access controls for your space.





This concludes the lab.

You have finished developing, deploying, and testing an AutoAI model. The model can now be integrated with other applications using REST API.