

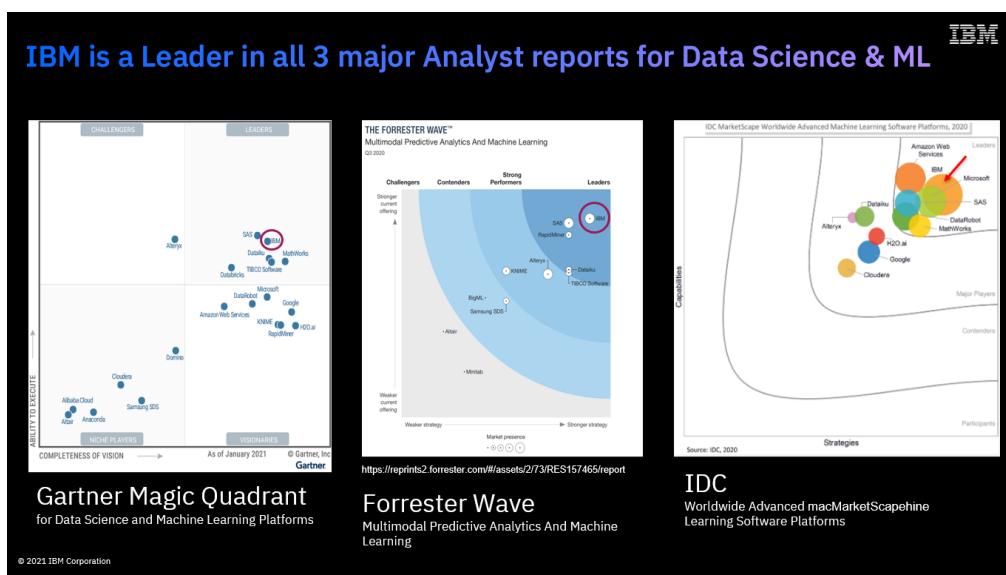
Use-Case

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.

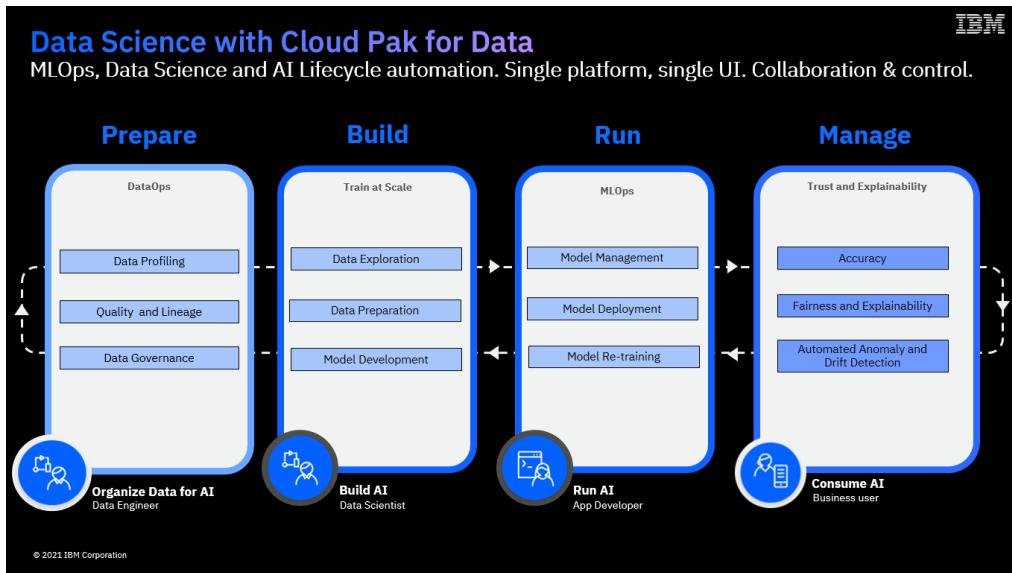
Your manager came by your desk today, gave you a couple of files containing some customer-related data and asked you to build a model helping the business understand which of its customers may churn soon, so that Marketing could proactively target them with personalized incentives and discount vouchers.

You have a lot of other work to do and are short on time and are in need of inspiration – your Python skills are a bit rusty.

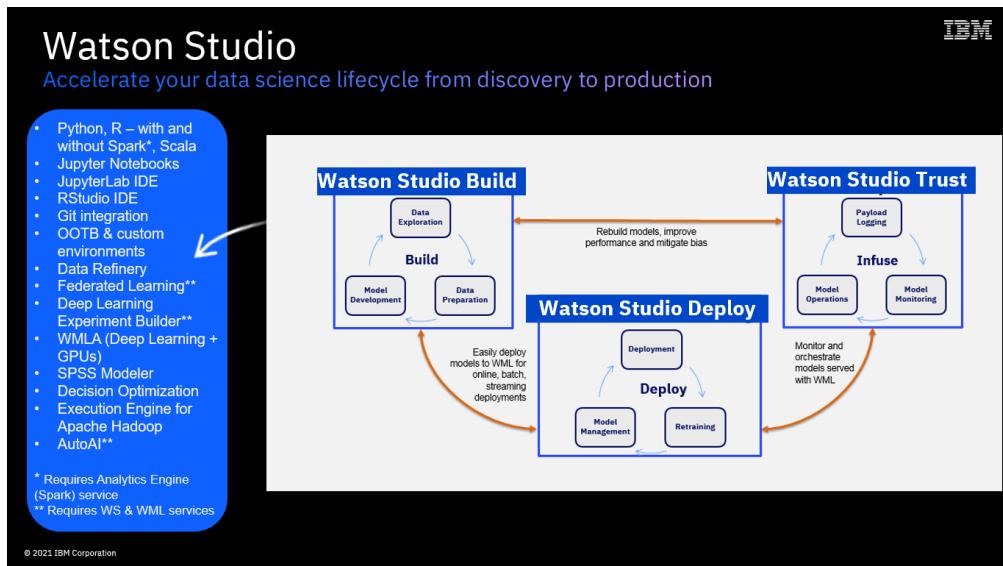
You are aware that IBM provides market-leading data science tooling and decide to use IBM's Watson Studio running on CPDaaS to build and test your model.



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:

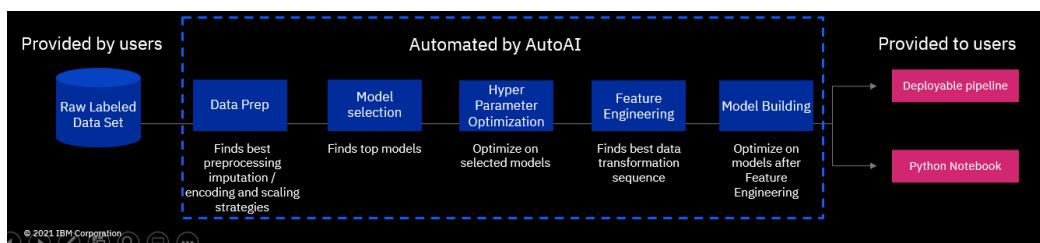
- Use AutoAI to automatically build the most optimal customer churn prediction model based on your data sets
- And, finally, deploy and test that model with Watson Machine Learning (Watson Studio Deploy capabilities).

... 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).

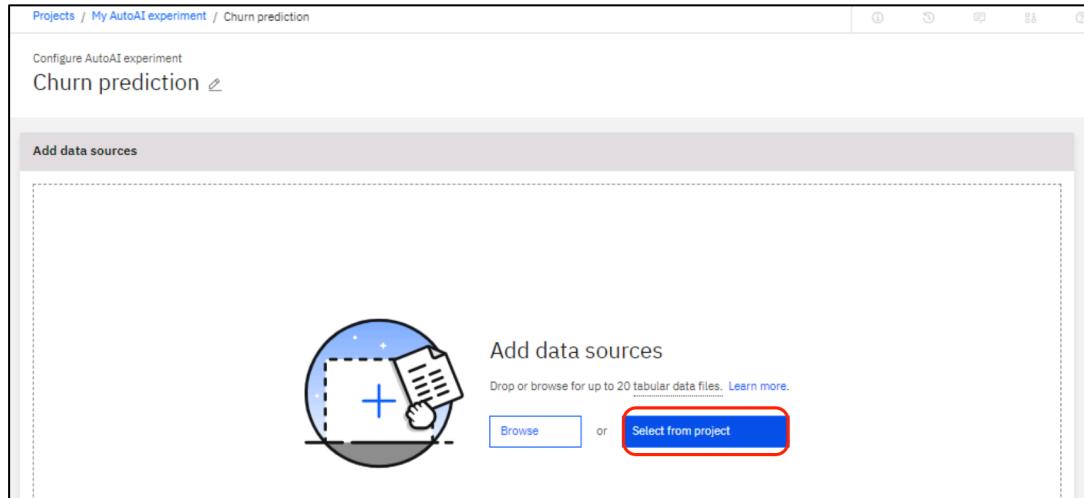
The screenshot shows the Watson Studio interface for an 'AutoAI experiment'. The top navigation bar includes 'Projects / My AutoAI experiment', 'Launch IDE', and a '01' button. The main area is titled 'Assets' with tabs for 'Overview', 'Jobs', and 'Manage'. A search bar 'Find assets' and a 'New asset' button are present. On the left, 'Asset types' show 1 'Data' asset and 1 'Experiments' asset. The central panel displays 'All assets' with a list including 'customer-profile-chur CSV' and 'Churn predict AutoAI experiment'. To the right, a sidebar titled 'Data in this project' features a red-bordered 'Drop data files here or browse for files to upload' area.

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.

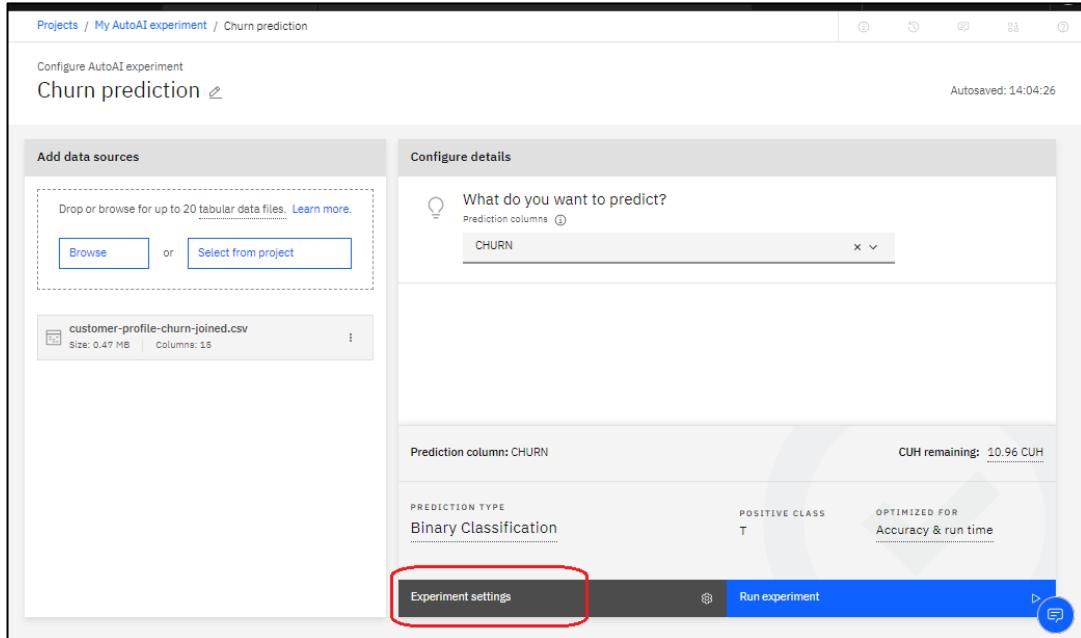


The screenshot shows the 'My AutoAI experiment data assets' page. It lists categories on the left and data assets on the right. Under 'Categories', 'Connection' and 'Data asset' are listed. Under 'Data assets', 'customer-profile-churn-joined.csv' is listed with a checked checkbox. A red box highlights this selected asset.

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

The screenshot shows the 'Configure details' page. On the left, the 'Add data sources' section is shown again. On the right, under 'Configure details', there is a checkbox labeled 'Create a time series forecast?' with the sub-instruction 'Enable this option to predict future activity over a specified date/time range. Data must be structured and sequential.' Below the checkbox is a 'Yes' button and a red box highlighting the 'No' button.

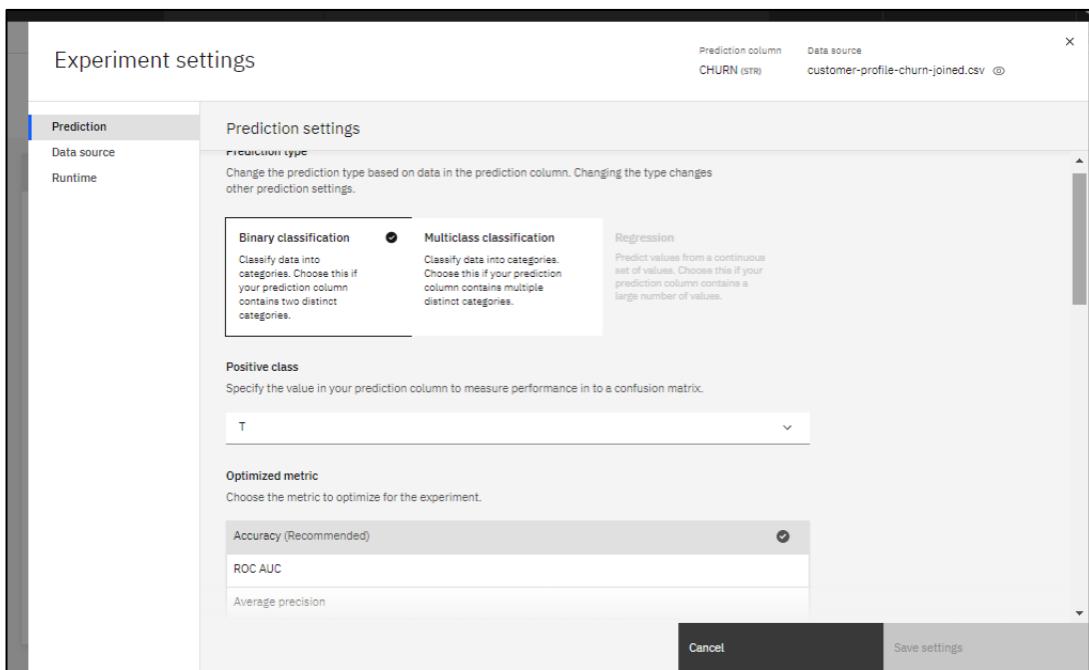
On the next screen we will need to let AutoAI know what we would like to predict (our target). In data science terminology the columns that we use to predict churn are called features, and the column that we are trying to predict is called target.



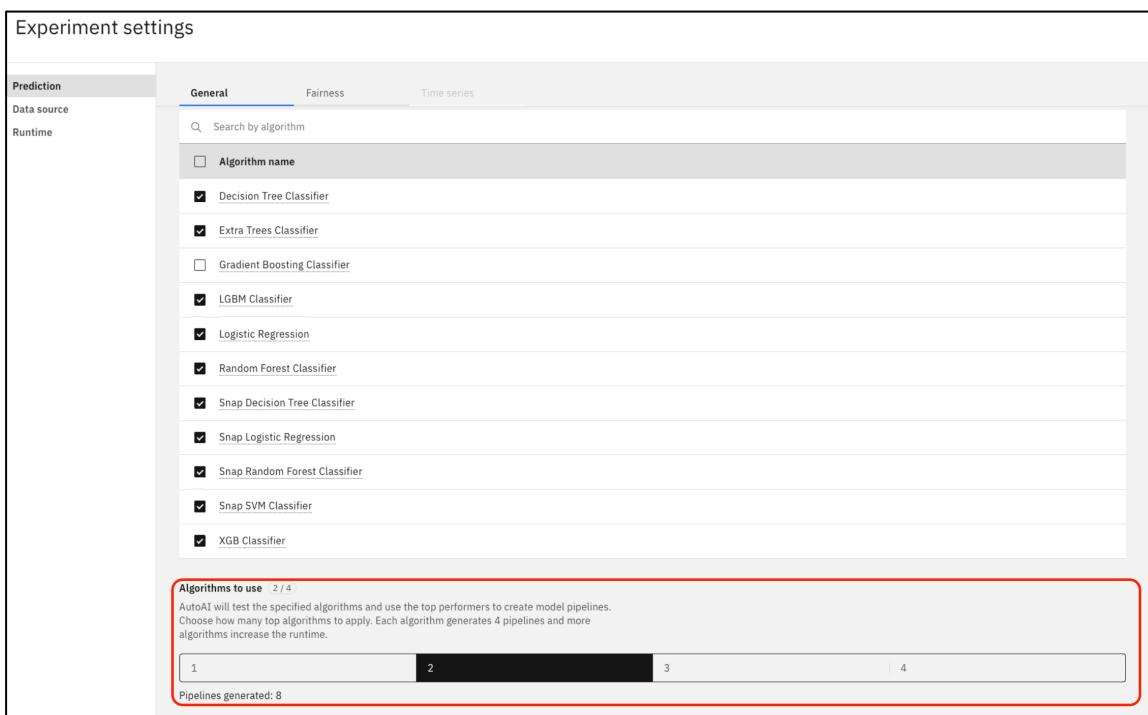
AutoAI will switch to Binary Classification (CHURN is a yes/no choice – or, more precisely True/False – the customer will either leave us or will not).

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".

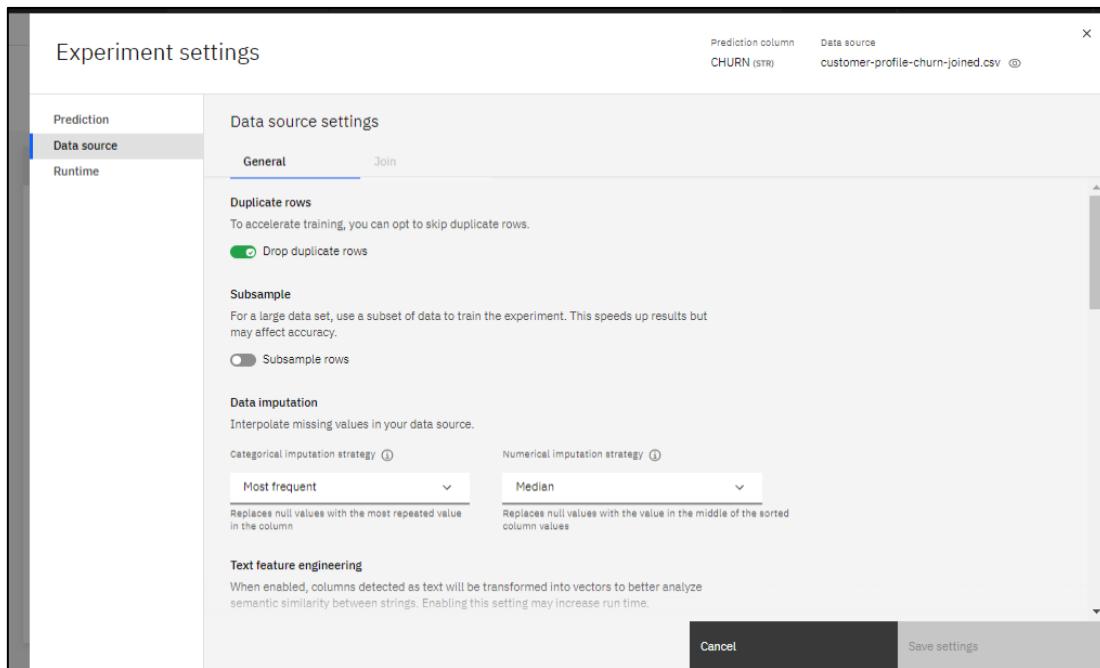
Positive class T (True) makes sense in our case as we are looking to predict which customers will actually churn.



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).



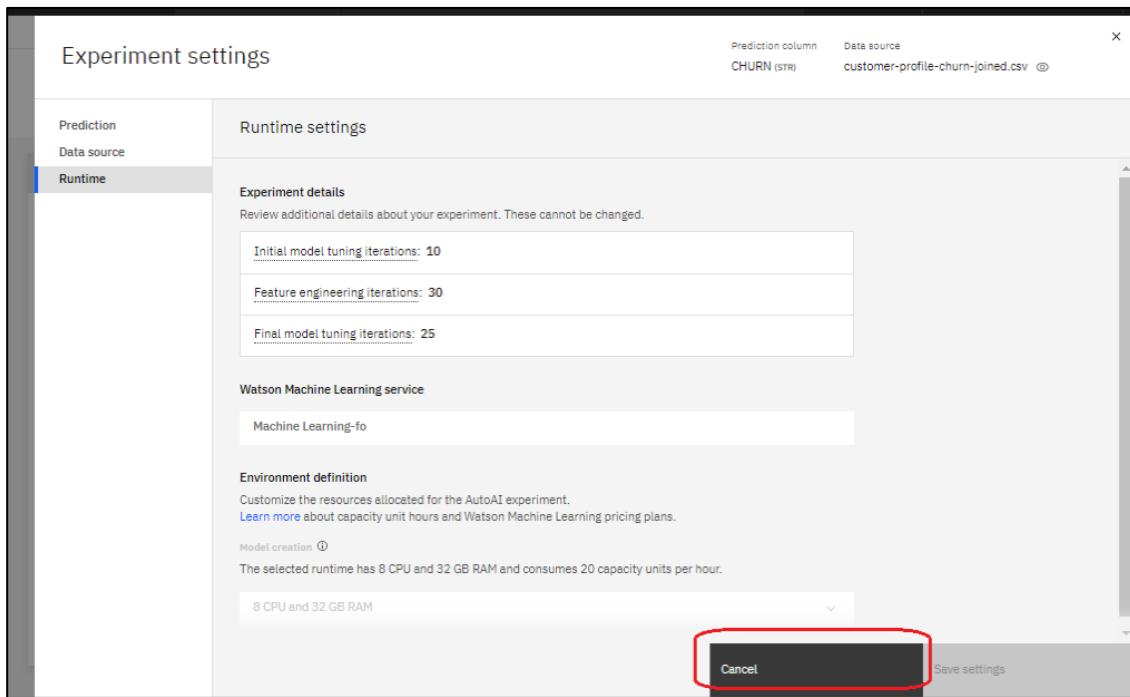
Next, switch to the Data Source tab from the menu on the left. Explore the settings. An explanation of the most important settings can be found subsequently to the following picture.



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.

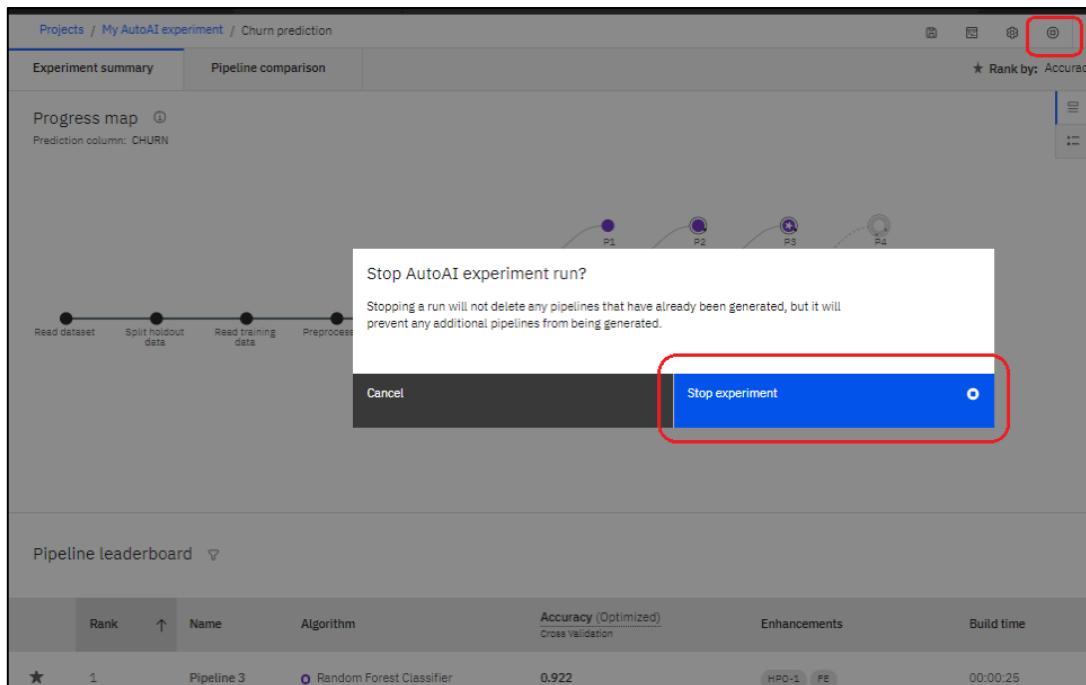
Finally, switch to the Runtime tab and review the screen. Once done, click cancel (since you haven’t changed any of the default settings) to quit the experiment settings.



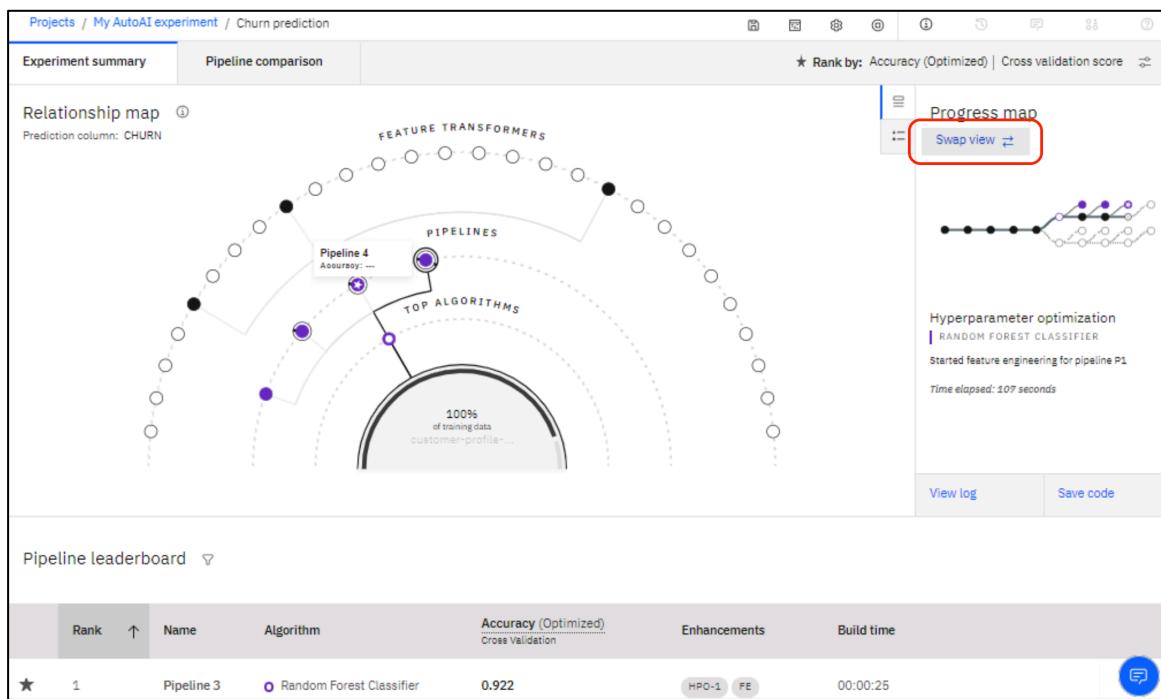
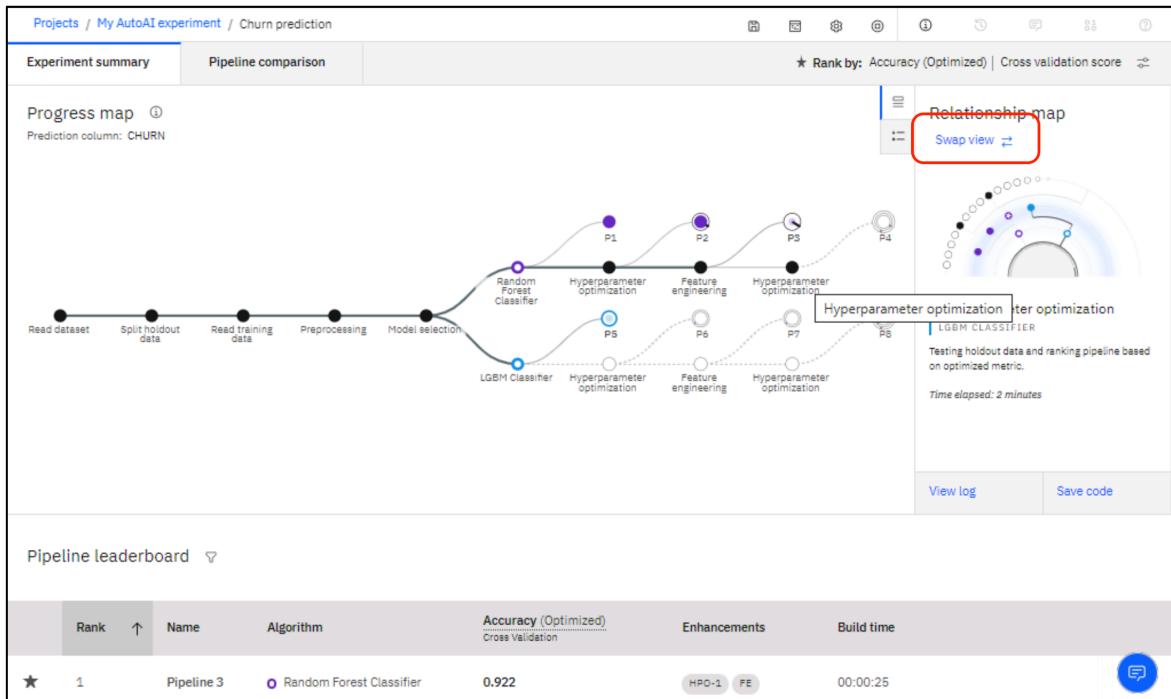
Finally, click “Run experiment”.



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.

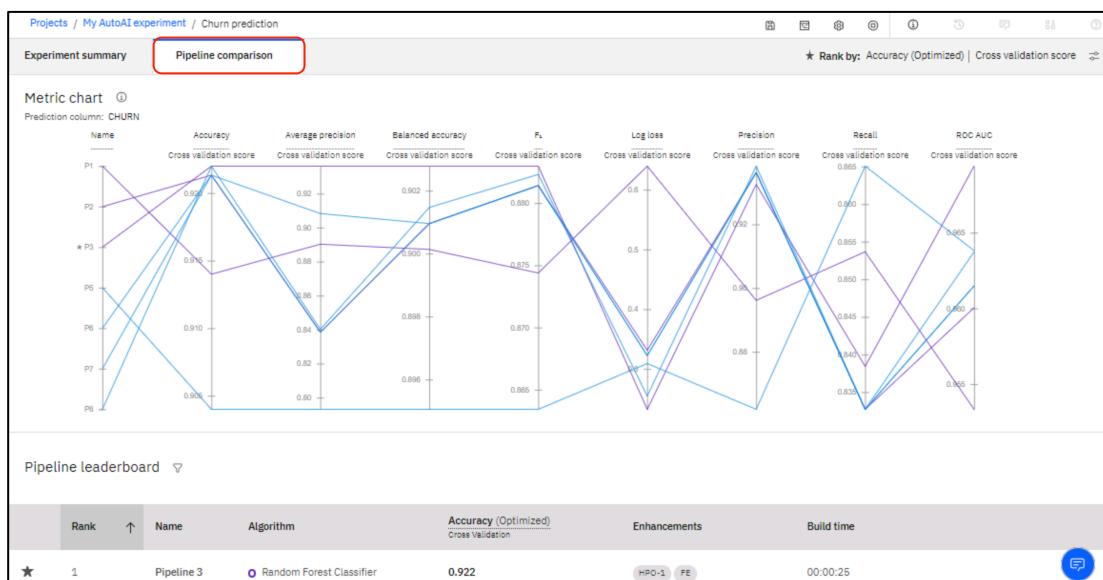
$P3$ and $P7$ perform feature engineering (derives new features) and build a model with these features.

P_4 and P_8 perform hyperparameter optimization for the model that uses the derived features.

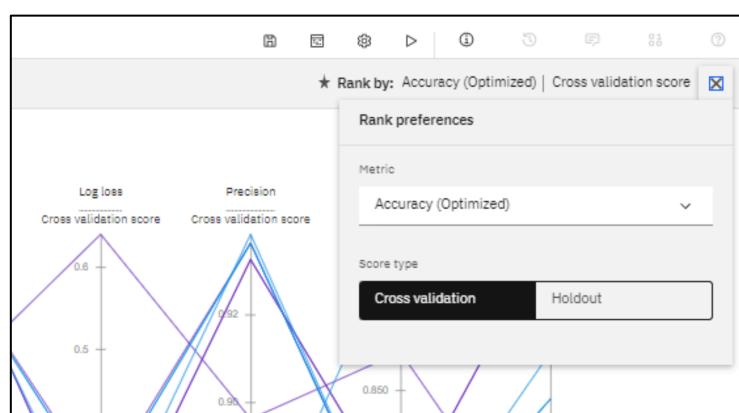
In summary, for each algorithm AutoAI creates 4 pipelines:

- 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*:

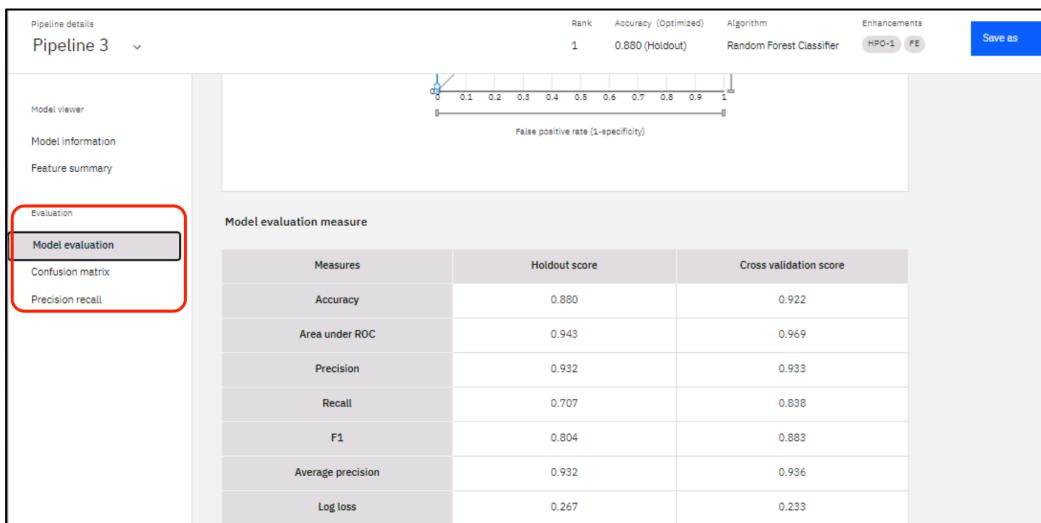
- $.90-1 = \text{excellent (A)}$
- $.80-.90 = \text{good (B)}$
- $.70-.80 = \text{fair (C)}$
- $.60-.70 = \text{poor (D)}$
- $.50-.60 = \text{fail (F)}$

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.

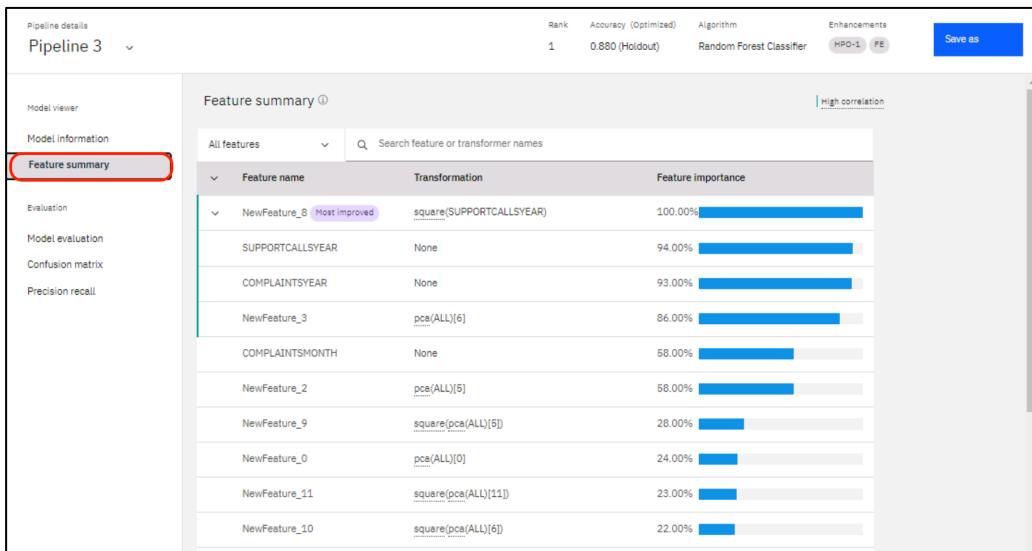
The screenshot shows the 'Pipeline comparison' tab in the AutoAI interface. A table lists seven pipelines based on accuracy (optimized) and cross-validation score. Pipeline 3 is highlighted with a red box.

Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time	Save as
1	Pipeline 3	Random Forest Classifier	0.922	HPO-1 FE	00:00:25	
2	Pipeline 6	LGBM Classifier	0.922	HPO-1 FE HPO-2	00:00:11	
3	Pipeline 2	Random Forest Classifier	0.921	HPO-1	00:00:06	
4	Pipeline 6	LGBM Classifier	0.921	HPO-1	00:00:04	
5	Pipeline 7	LGBM Classifier	0.921	HPO-1 FE	00:00:15	
6	Pipeline 1	Random Forest Classifier	0.914	None	00:00:01	
7	Pipeline 5	LGBM Classifier	0.904	None	00:00:01	

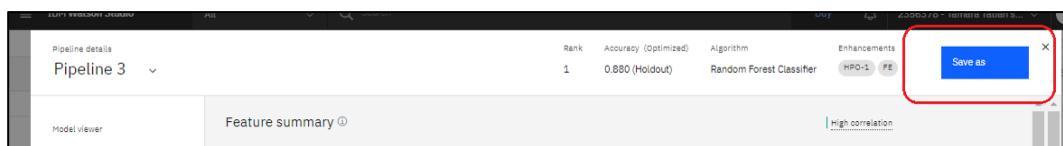
Review the details. Under “Evaluation”, you will find various metrics that are familiar to data scientists and help them to evaluate which pipeline has performed in the best way. In our case, we focus on the metric “accuracy” for the purpose of ease. Depending on the business context, other metrics (like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



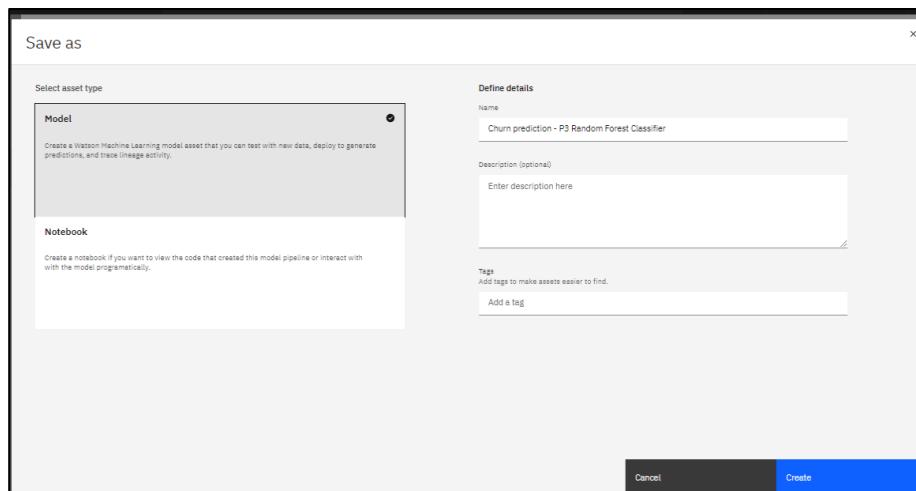
Another interesting aspect can be found under “feature summary”. Here, we can see which features make the most impact on the prediction.



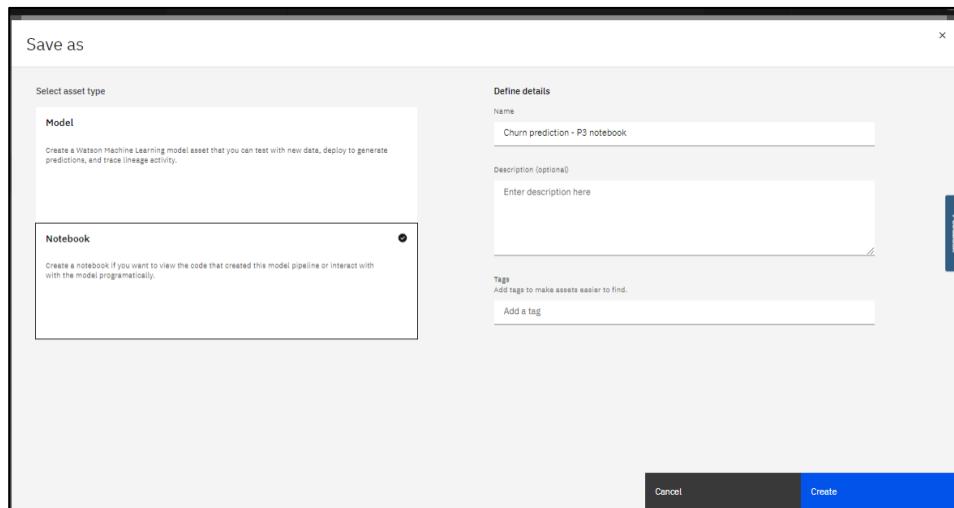
Let's now save the pipeline. Just click on one of the pipelines, and a "save as" button will appear – as shown below.



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.



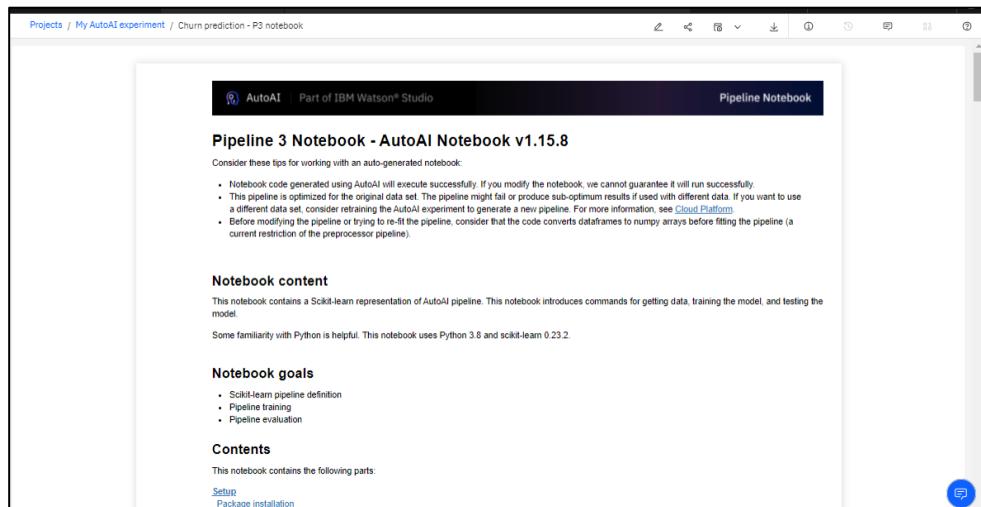
Repeat the same process. This time, please save it as a Notebook.



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.

Projects / My AutoAI experiment

> AutoAI experiments

> Notebooks

Name	Shared	Scheduled	Status	Language	Last editor	Last modified
Churn prediction - P3 notebook				Python 3.8	Tamara Tatian	Nov 22, 2021

> Deep learning experiments

> Models

Watson Machine Learning models

Name	Type	Software specification	Last modified
Churn prediction - P3 Random Forest Classifier	wml-hybrid_0.1	hybrid_0.1	Nov 22, 2021

> Data Refinery flows

Name	Type	Created by	Last modified
my_refinery_flow_25.11.21	Data Refinery flow	Tamara Tatian	Nov 22, 2021, 01:35 PM

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.

Projects / My AutoAI experiment

Overview Assets Jobs Manage

Project

General

Access control

Environments

Resource usage

Services & integrations

Environments

Manage your runtime environments and create templates to easily reuse environment configurations for running your assets. [Learn more](#).

Active runtimes Templates (26)

Find runtime

Active runtime	Hardware
----------------	----------

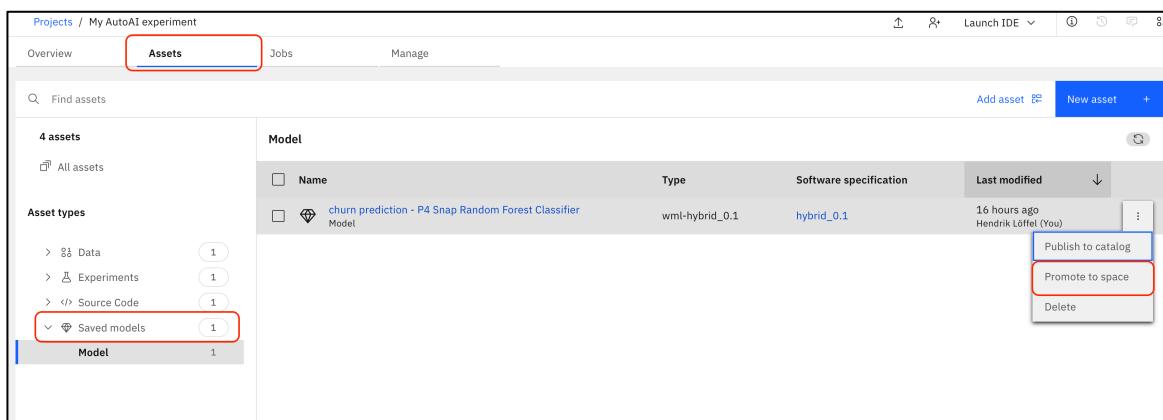
Deploying Your Model with Watson Machine Learning

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.

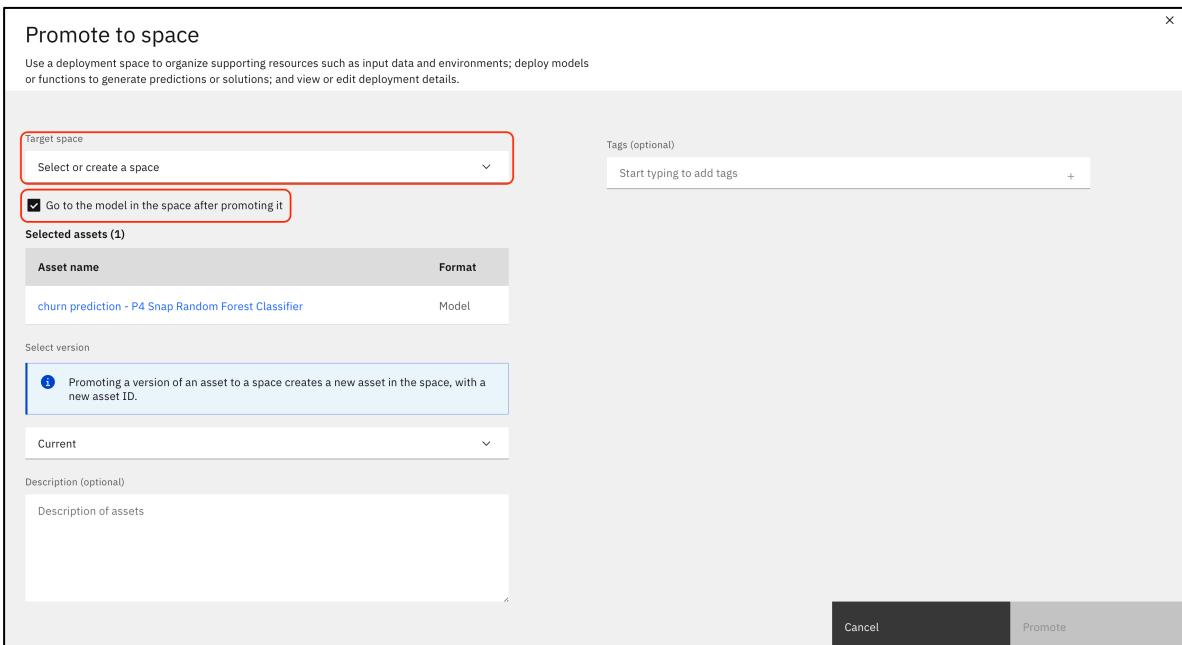
You can deploy assets from multiple projects to a space, and you can deploy assets to more than one space. For example, you might have a test space for evaluating deployments, and a production space for deployments you want to deploy in business applications.

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*.

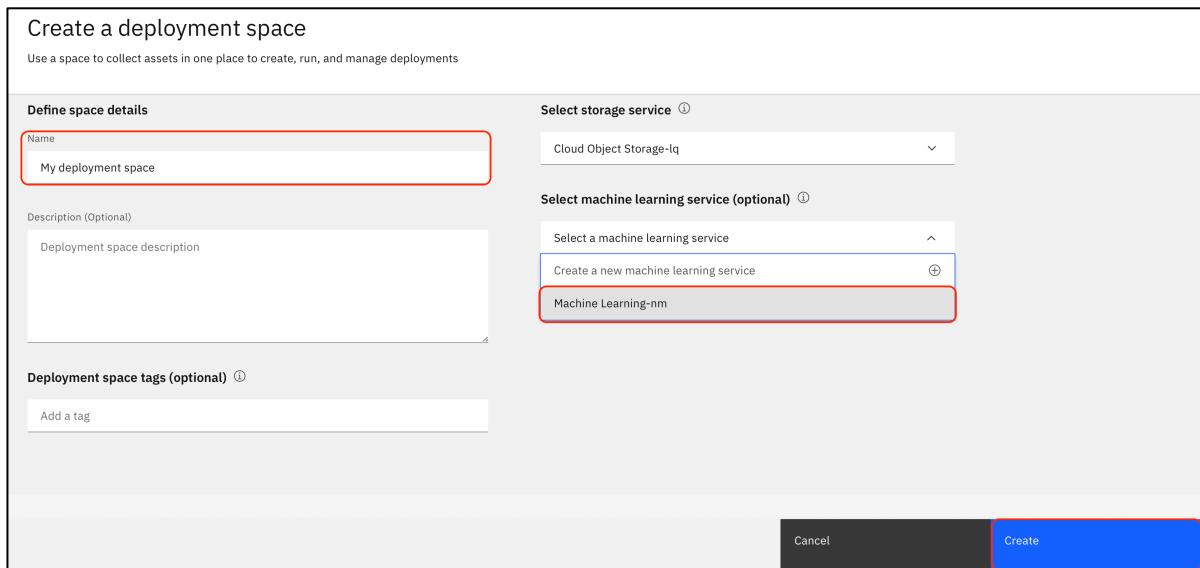
From the project menu, switch to assets and click on “Saved models”. From there, click “Promote to Space”.



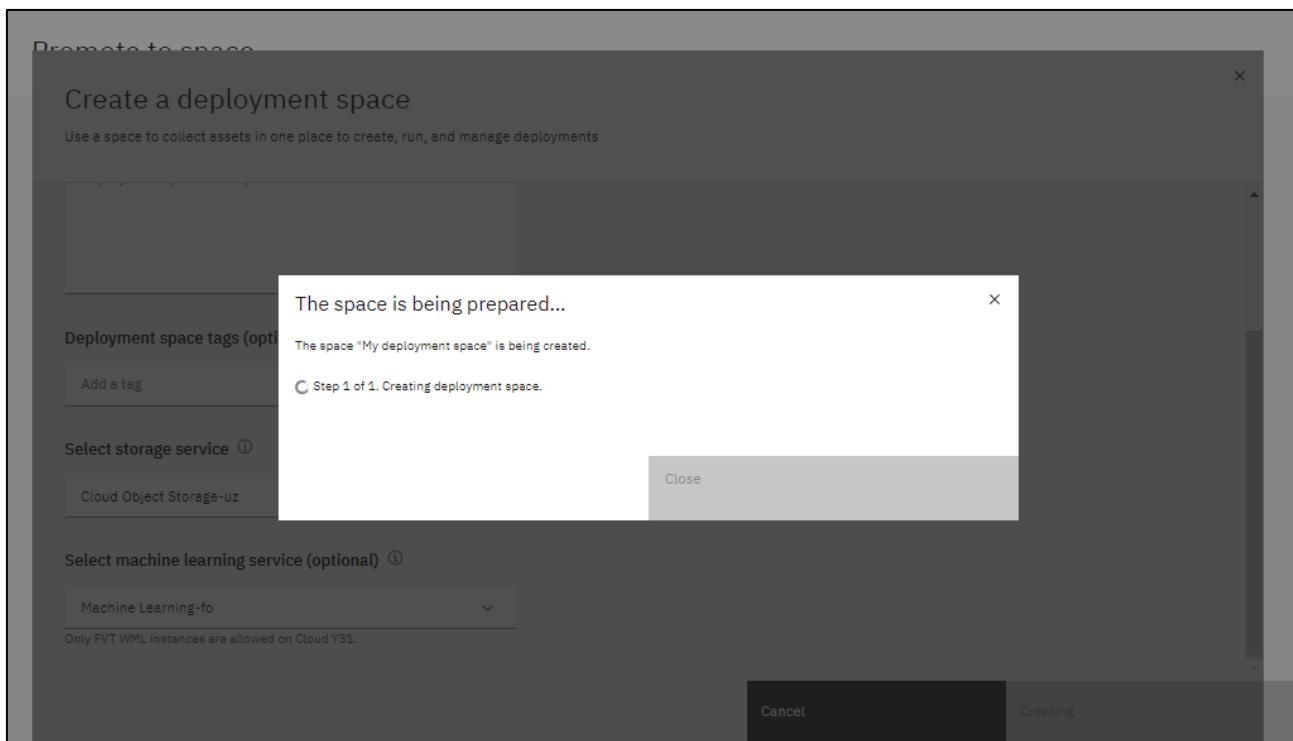
Check the box saying “Go to Model” on the next screen. Then, select from the drop down menu the option “Create a new deployment space”.



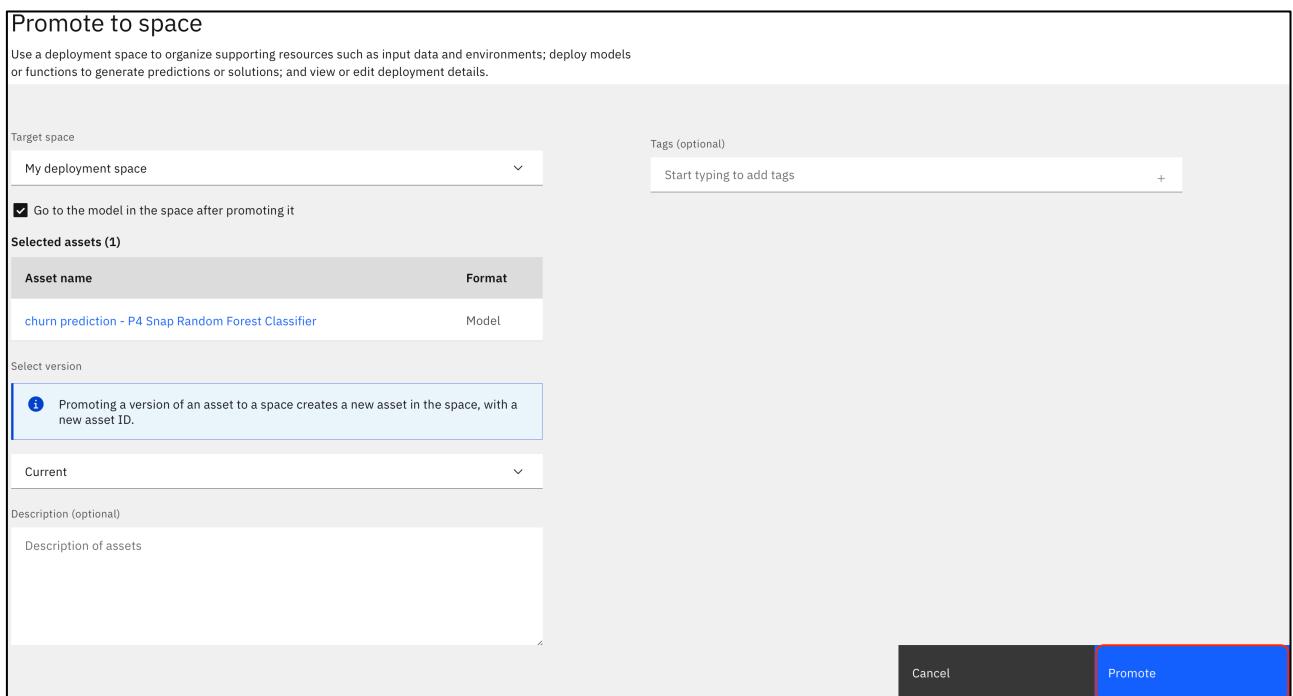
Create a new deployment space. Give it a random name - and make sure to select the machine learning service correctly.



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.

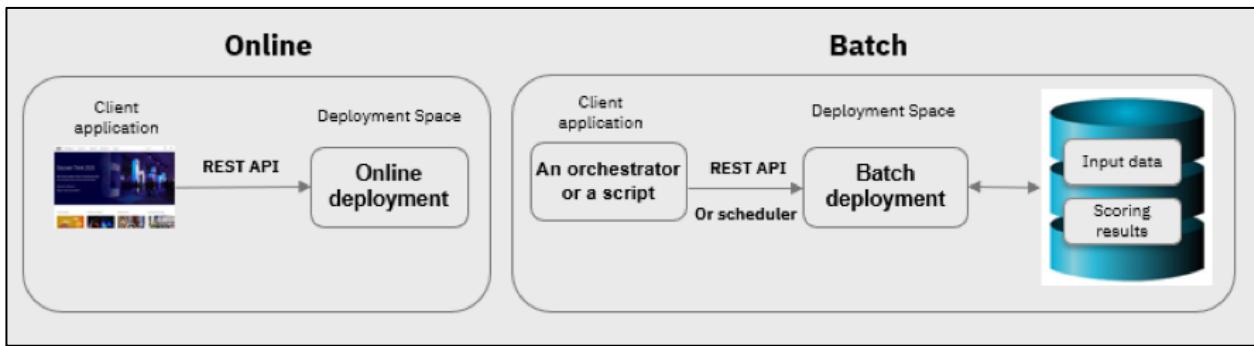
Next, we will deploy the model. Click the “New Deployment” button.

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”.

Create a deployment

Associated asset
churn prediction - P4 Snap Random Forest Classifier

Deployment type

- Online** (selected) Run the model on data in real-time, as data is received by a web service.
- Batch** Run the model against data as a batch process.

Name

Serving name ⓘ

Description

Tags
Add tags to make assets easier to find.

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*.

The screenshot shows the IBM Watson Studio interface. At the top, there's a navigation bar with 'IBM Watson Studio', a search bar, and account information for 'Hendrik Löffel's Account'. Below the navigation is a breadcrumb trail: 'Deployments / My deployment space /'. The main area displays a deployment named 'churn prediction - P4 Snap Random Forest Classifier'. A blue button labeled 'New deployment' is visible. On the left, a sidebar titled 'Deployments' is selected, showing a table of deployment types: 'Online' (1) and 'Batch' (0). The 'Online' row has a sub-table under 'Name' with a single entry 'New Deployment name' which is highlighted with a red box. The 'Status' column for this entry shows a circular progress icon followed by 'In progress'. The 'Last modified' column shows 'Apr 21, 2022 12:04 PM'. To the right of the deployment table is a detailed view of the deployment, including its creation date ('Apr 21, 2022 11:25 AM'), type ('wml-hybrid_0.1'), model ID ('523c561f-d7a2-4684-8c04-14a1...'), software specification ('hybrid_0.1'), hybrid pipeline software specifications ('autoai-kb_rt22.1-py3.9'), description ('No description provided.'), and tags ('Add tags to make assets easier to find').

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

This screenshot shows the same deployment page after the deployment has completed. The 'Status' column for the 'New Deployment name' entry now shows a green checkmark icon followed by 'Deployed'. The rest of the interface remains the same, including the sidebar, the deployment details on the right, and the overall layout of the Watson Studio dashboard.

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

The screenshot shows the IBM Cloud Deployment Center interface. On the left, there's a navigation bar with 'Deployments / My deployment space / Churn prediction experiment - P...'. The main area has tabs for 'API reference' (selected) and 'Test'. Under 'API reference', there's a 'Direct link' section with an endpoint URL: <https://eu-de.ml.cloud.ibm.com/ml/v4/deployments/96ea5511-6773-49de-b3da-06bdobe0>. To the right of the URL are fields for 'Bearer <token>' and 'IAM'. Below this is a 'Code snippets' section with tabs for 'cURL', 'Java', 'JavaScript', 'Python', and 'Scala'. The 'cURL' tab is selected, displaying a sample curl command. On the far right, there's a sidebar with deployment details: Created Nov 19, 2021 1:03 PM, Updated Nov 19, 2021 1:03 PM, Deployment ID 96ea5511-6773-49de-b3da-0..., Software specification hybrid_0.1, Hybrid pipeline software specifications autoai-kb_3.4-py3.8, Copies 1, Serving name No serving name, Description No description provided, and Tags Add tags to make assets easier to find.

Navigate to the “Test” tab.

The screenshot shows the 'Test' tab for a deployment. At the top, it says 'New Deployment name' with a status of 'Deployed Online'. Below this are tabs for 'API reference' (selected) and 'Test'. The 'Test' tab has a 'Direct link' section with an endpoint URL: <https://eu-de.ml.cloud.ibm.com/ml/v4/deployments/ae545795-2054-4cf7-a983-fc51d356ad95/predictions?version=1>. To the right of the URL are fields for 'Bearer <token>' and 'IAM'. Below this is a 'Code snippets' section with tabs for 'cURL', 'Java', 'JavaScript', 'Python', and 'Scala'. The 'cURL' tab is selected, displaying a sample curl command. There are also input fields for 'Input data' and a 'Predict' button.

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.

The screenshot shows the 'My deployment space' dashboard. At the top, there are tabs for Overview, Assets, Deployments (selected), Jobs, and Manage. Below the tabs, there's a summary section with counts for Deployments (1 Deployed, 0 Failed) and Job runs (0 Active, 0 Failed last 24 hours). An 'Assets' section lists a single item: 'Churn prediction experiment - P3 LGBM Classifier' created 22 minutes ago. To the right, a 'Space activity' panel displays two entries: 'Online deployment ready' (green checkmark) and 'Online deployment created' (black person icon). Both entries link to the deployment details page.

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.