Introduction to AutoAI, Data Refinery and Watson Studio Deploy (WML) : Hands-on Lab

Tamara Tatian, IBM Cloud Pak for Data Technical Sales EMEA

Pre-work

To complete the Introduction to AutoAI, Data Refinery and Watson Studio Deploy (WML) Hands-on Lab, you will be using IBM Cloud Pak for Data as-a-Service (CPDaaS).

Pre-requisites: IBM ID, IBM Cloud Account, CPDaaS account – provisioned services: Watson Studio, Watson Machine Learning, Cloud Object Storage.

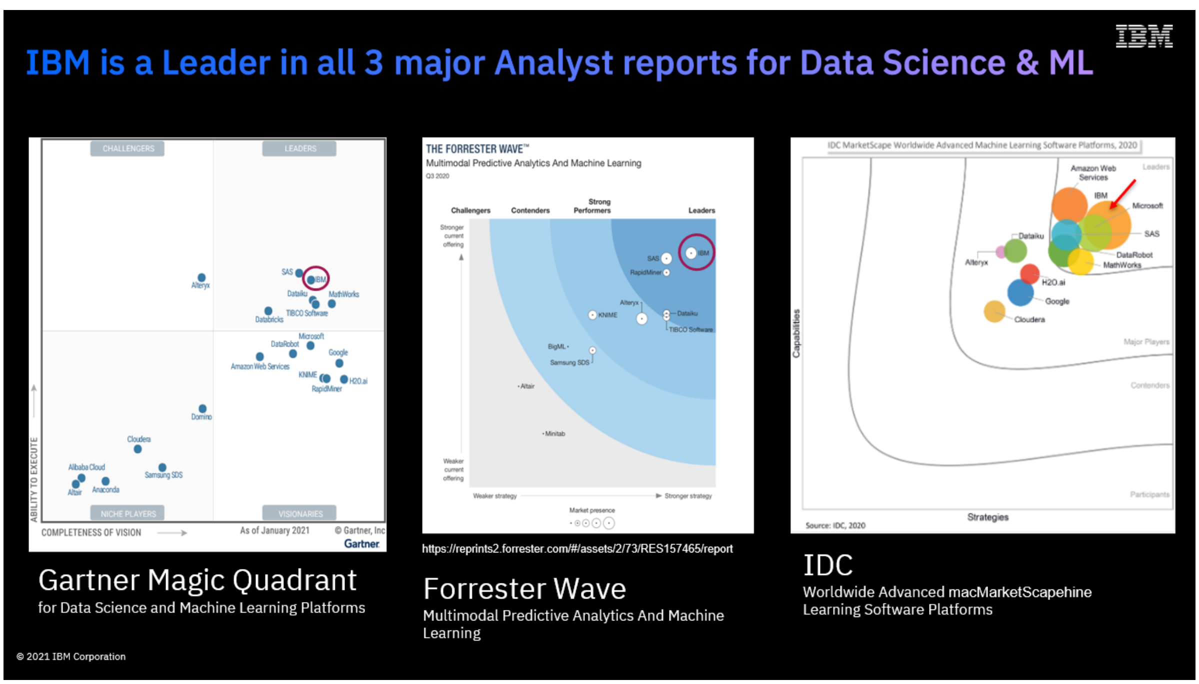
Please refer to the Pre-work lab companion guide and complete the required set up steps before attempting the lab.

Overview

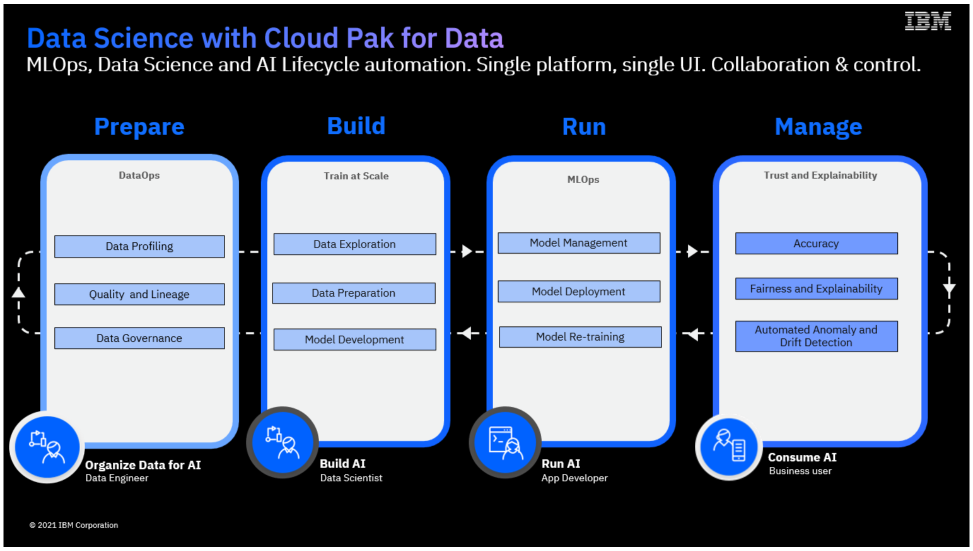
USE CASE

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 in the near future, so that Marketing could proactively target them with personalized incentives and discount vouchers.

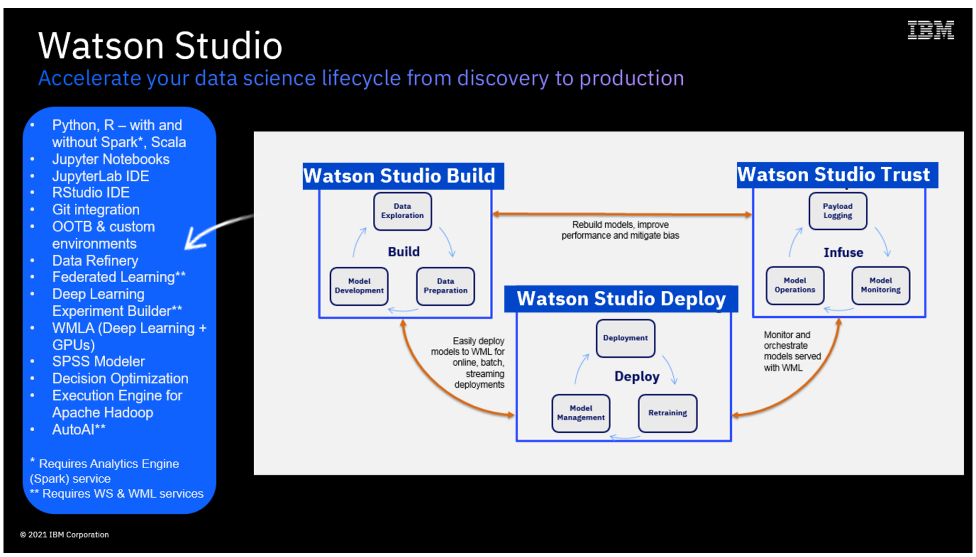
You have a lot of other work to do and are a short on time, and are in need of inspiration – your Python skills are a bit rusty.
You are aware that IBM provides market-leading data science tooling and decide to use IBM's Watson Studio running on CPDaaS to build and test your model.



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:

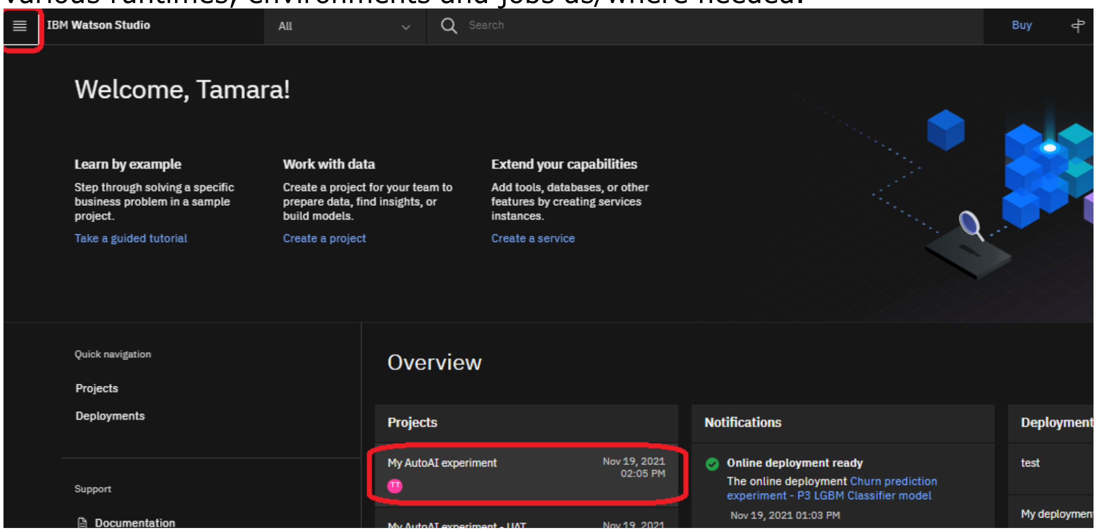
- Use Watson Studio's Data Refinery tooling to explore, prepare and shape the data you were given
- Use AutoAI to automatically build the most optimal customer churn prediction model based on your data sets
- And, finally, deploy and test that model with Watson Machine Learning (Watson Studio Deploy capabilities).

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

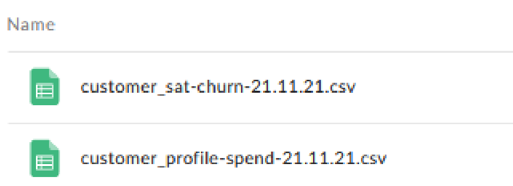
Exploring and preparing data with Data Refinery

Log in to your CPDaaS account: https://eu-de.dataplatform.cloud.ibm.com/
Navigate to your My AutoAI Experiment project you created as part of the pre-work – you can do that from the home page tile, or, alternatively – by going through the main menu (top left of the screen).

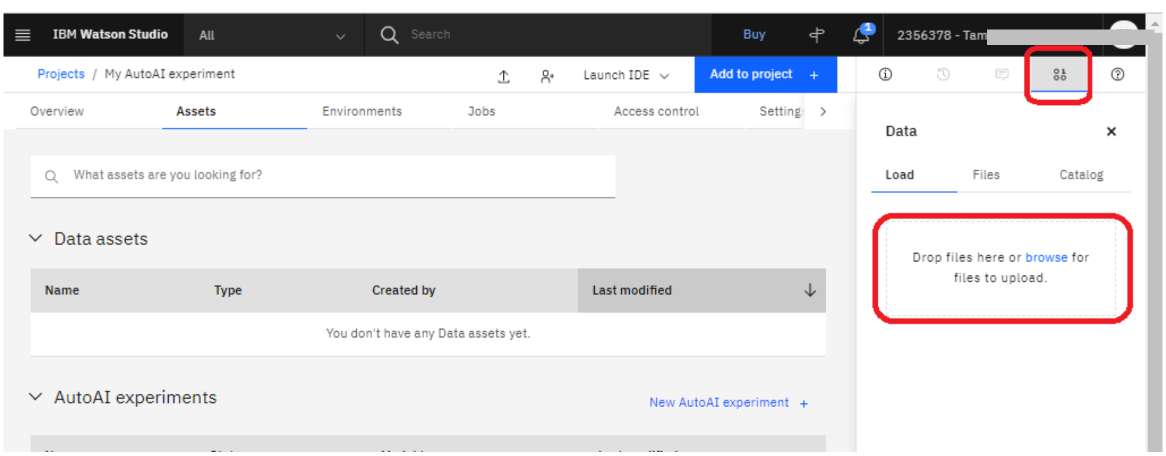
A Project is a collaborative workspace where you work with data and other assets to achieve a particular goal. Your project resources can include data, collaborators, tools, and operational assets that run code, like notebooks and models. Projects allow you to work with different analytical tools and IDEs built into the platform, and will spin up and run various runtimes, environments and jobs as/where needed.



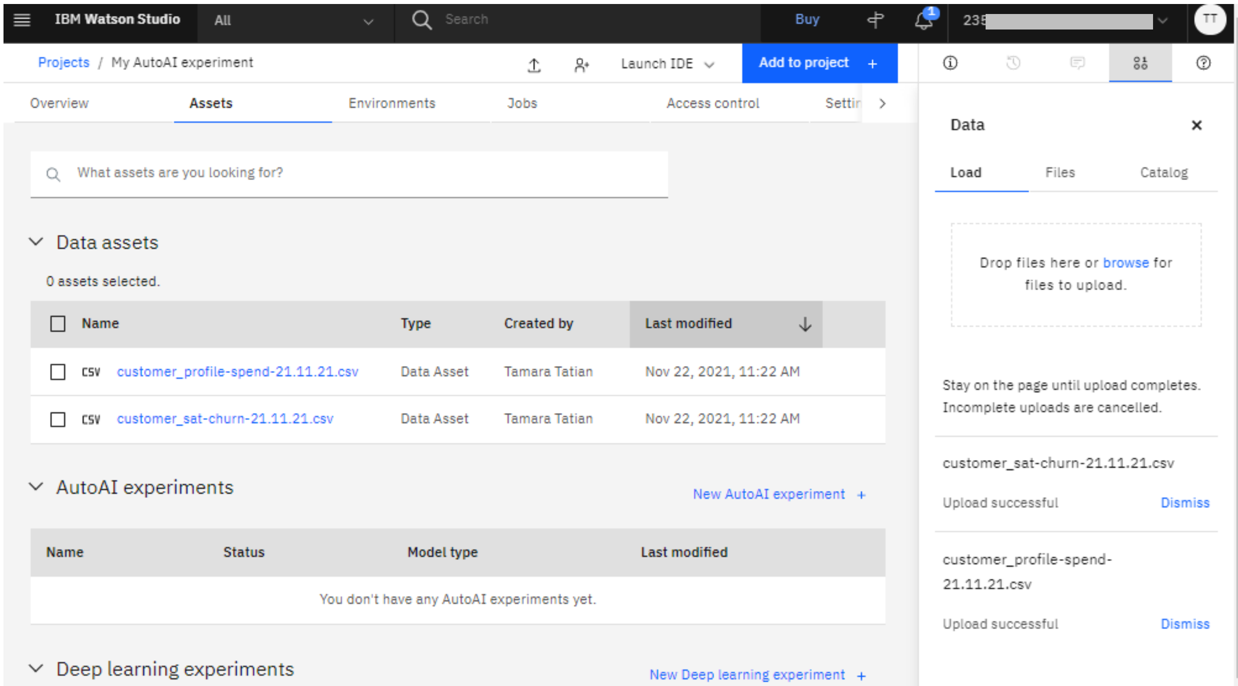
Download both files from the following folder: https://ibm.box.com/v/autoai-lab-2021-files



Drag and drop both the csv files into the "Drop files here" box under Data – Load

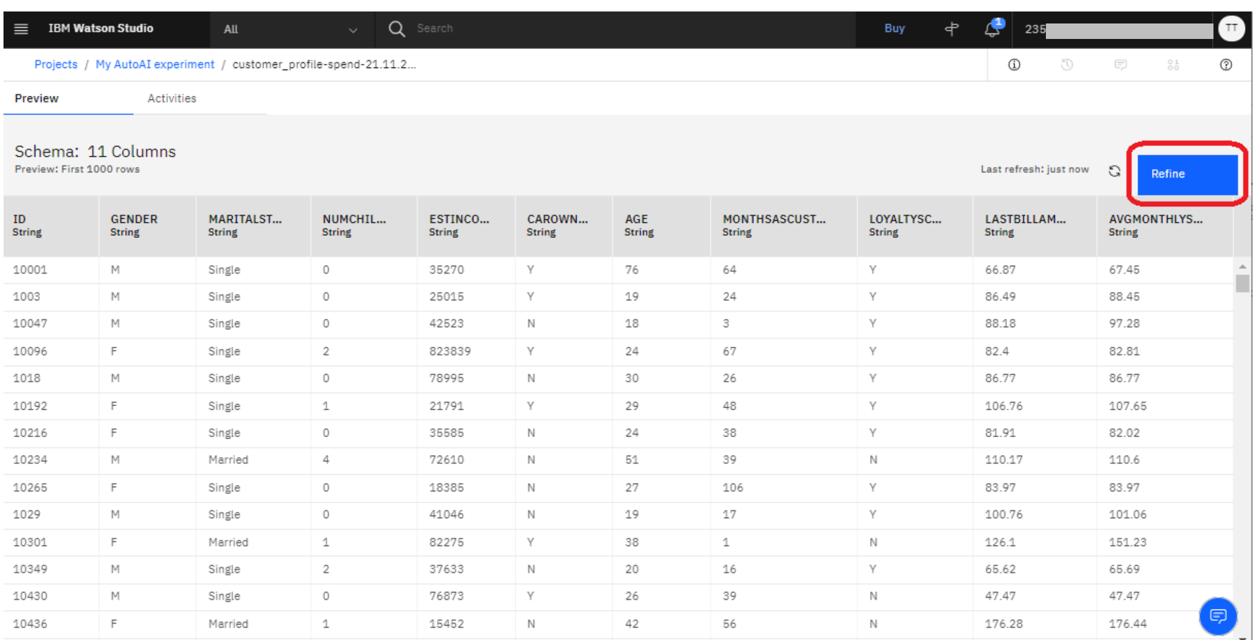


Once the upload finishes, navigate to the customer_profile-spend-21.11.21.csv by clicking the corresponding link in the Data Assets section



Cloud Pak for Data allows you to preview the data in your data asset. This applies to both files that you physically load to projects and catalogues, and to "Connected assets" – files and tables residing in remote data sources (that you can connect to Cloud Pak for Data using a wide range of standard connectors through "Connections"). In this lab, we will be working with uploaded project files only.

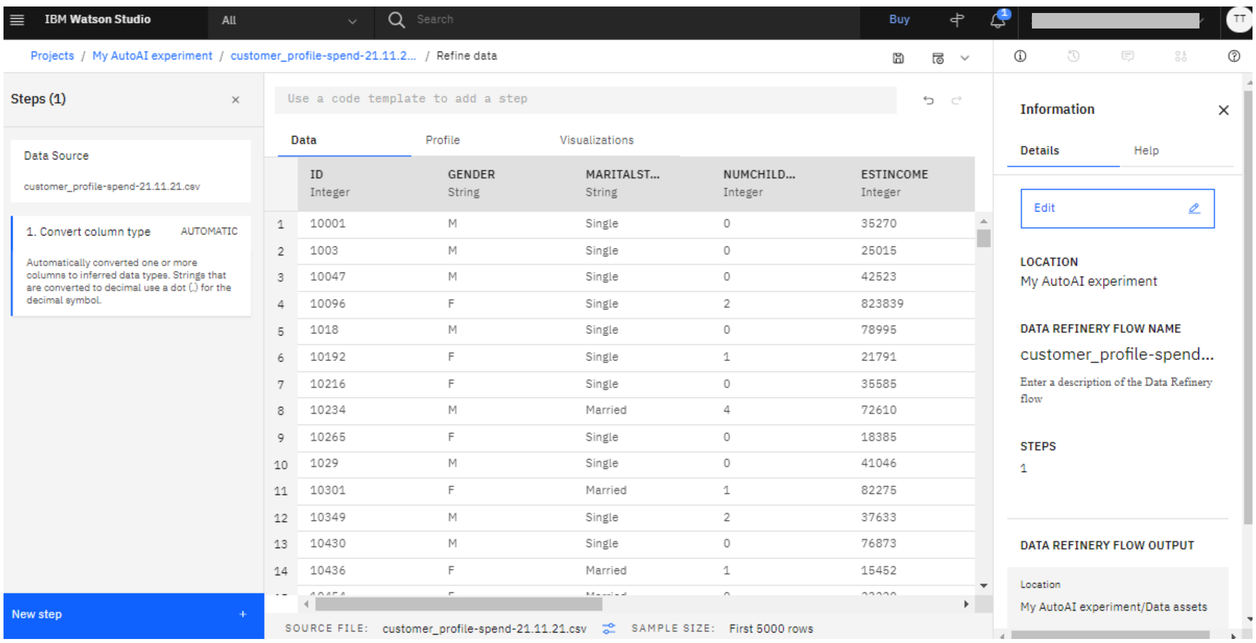
Let's explore the data further and do some data wrangling. Click the Refine button



The platform will fire up a Data Refinery instance (sandbox) for your file.

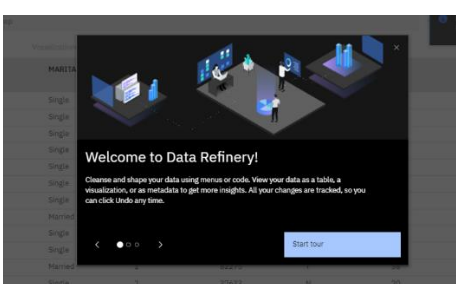
[Data Refinery](#) is a built-in data wrangling and data preparation tool available in Watson Studio. It helps reduce the amount of time it takes to prepare data for analysis and data science and allows you to cleanse and shape tabular data with a graphical flow editor. You can also use interactive templates to code operations, functions, and logical operators (R code is used). With Data Refinery, you can:

- Interactively discover, cleanse, and transform your data with over 100 built-in operations. No coding is required.
- Understand the quality and distribution of your data using dozens of built-in charts, graphs, and statistics.
- Automatically detect data types and business classifications.
- Schedule data flow executions for repeatable outcomes.



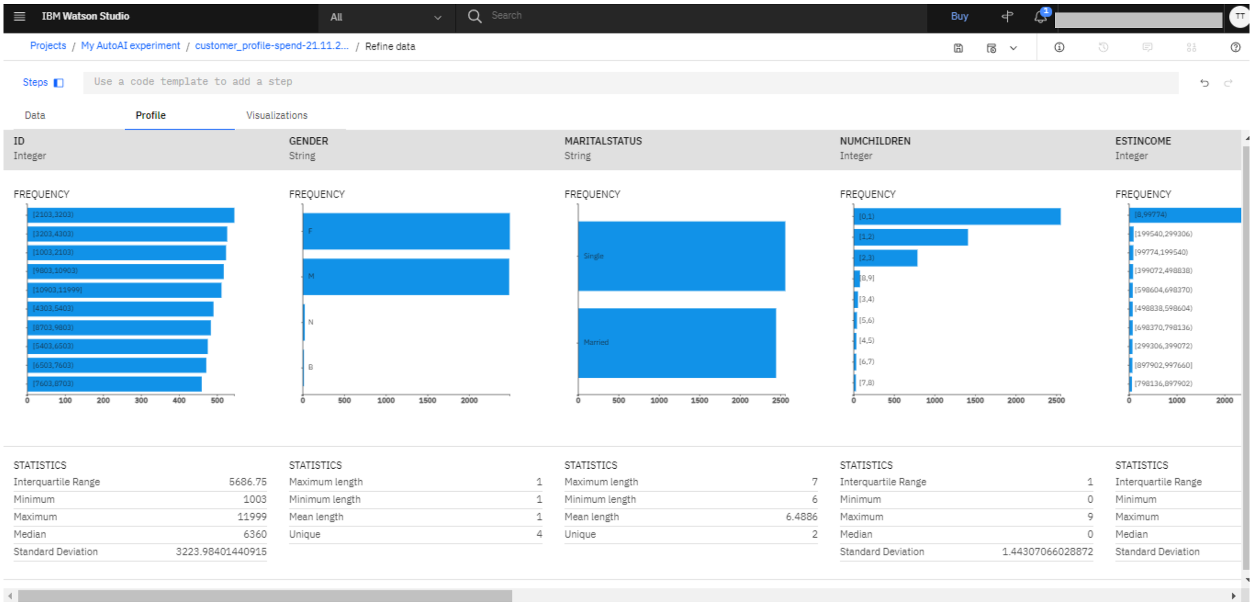
In your Data Refinery sandbox you can explore your data, design data wrangling and cleansing 'recipes' (flows) that you can further save and execute as jobs. While you are building your data wrangling recipes, all the transformations and changes are effectively performed in-memory and actual data is not touched at that point. It is only once you choose to execute your flow by running a Data Refinery job that the actual data will be transformed and changed.

On first load, the system may offer you to take a Tour – please feel free to explore or skip it.



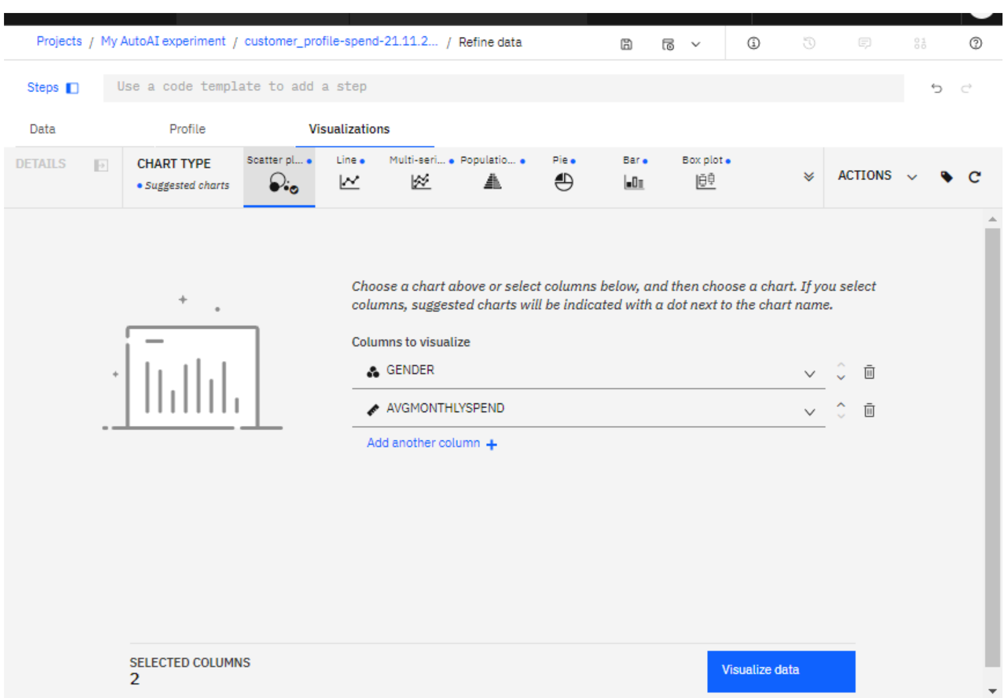
First, let's explore our data. Minimise the Information and Steps panes on the screen (click X in those sections). Click on the Profile tab above the Gender column title.

The Profile tab shows you statistics and frequency analysis for each of the columns in your data set.

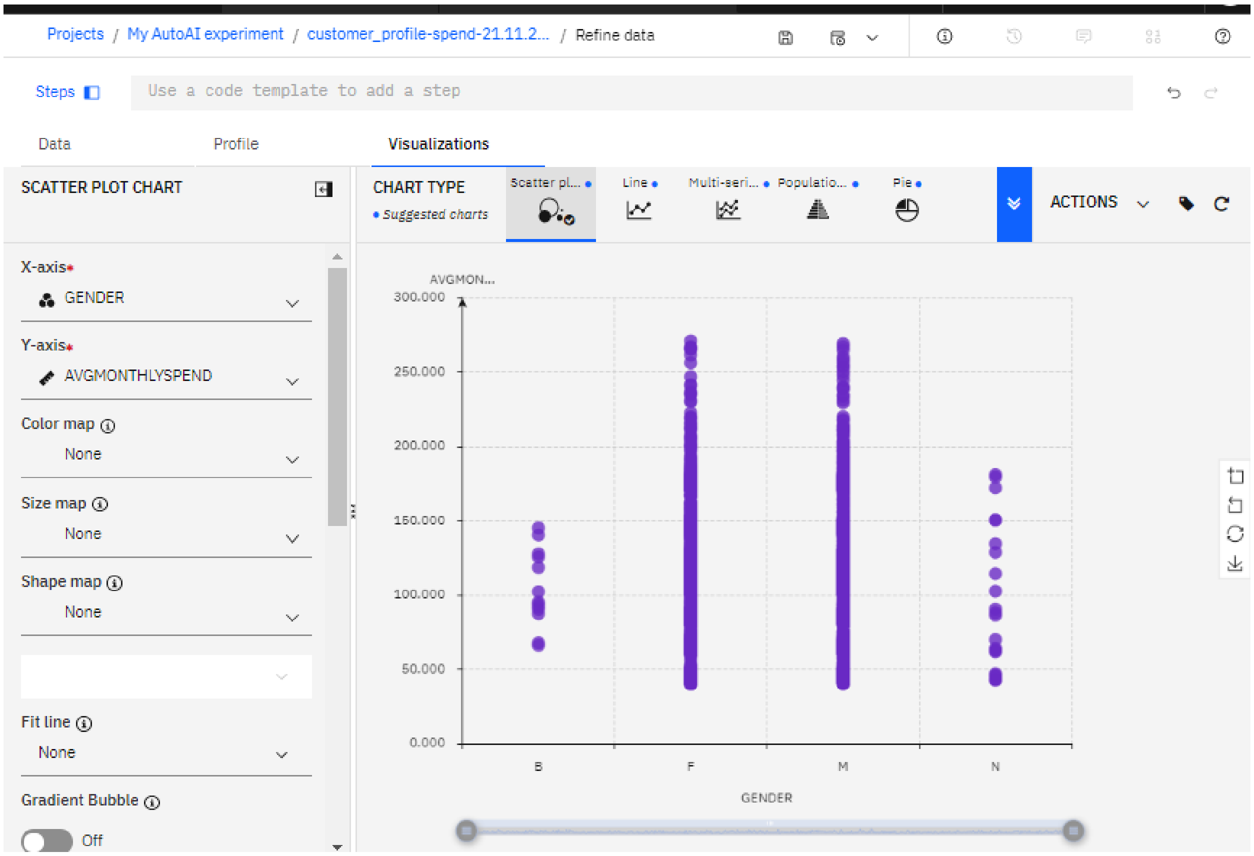


Next, let's visualize the data to try to explore and understand it a bit more. Navigate to the Visualizations tab above the Gender column title. Select GENDER as your first column, and add AVGMONTHLYSPEND as your second one, then click the Visualize Data button.

Refinery will automatically suggest the best fit type of visualization based on the data and number of columns that you choose.

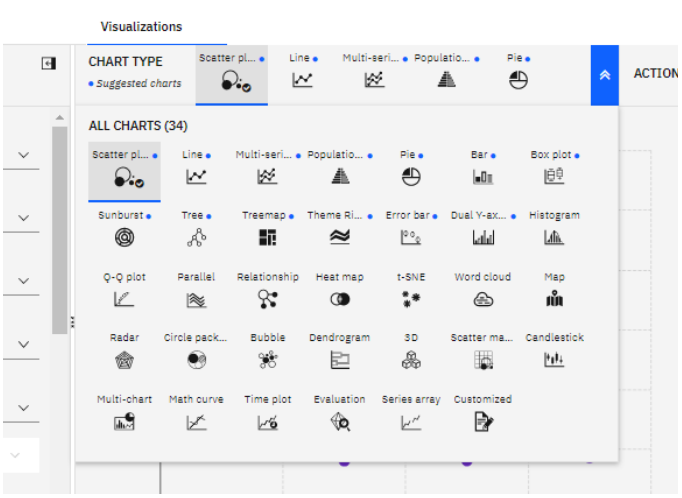


In our case, it picked Scatter plot.



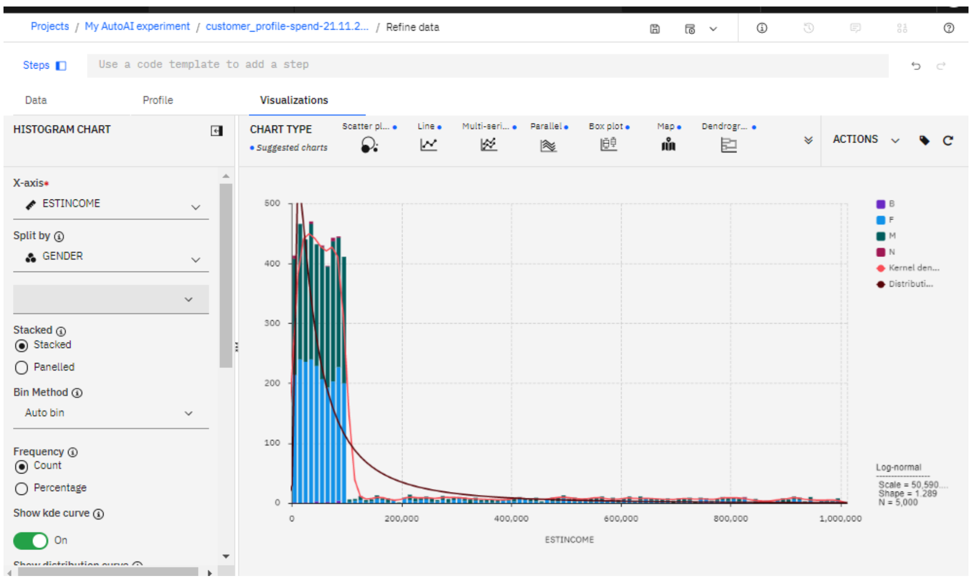
Note that the visualization highlighted to us that in our data set, our customers' gender data contains not only the more typical F and M values, but also B and N. This may warrant further investigation – there may be issue with data quality, or those could be legitimately valid values, depending on our company's data governance and capture policies and rules.

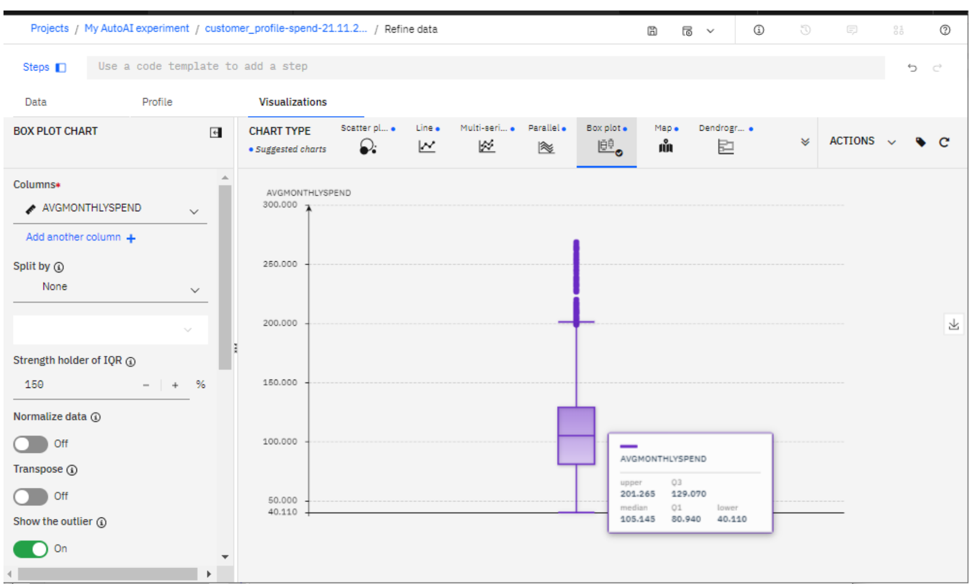
Expand the Chart Type menu by clicking on the chevron button. Note that the most suitable chart types are marked with a blue dot next to them. Feel free to switch between them and explore.



If you wanted to reset the visualization and start over, reset the chart by clicking the Start Over button. Note that you can also save your visualizations as an image.

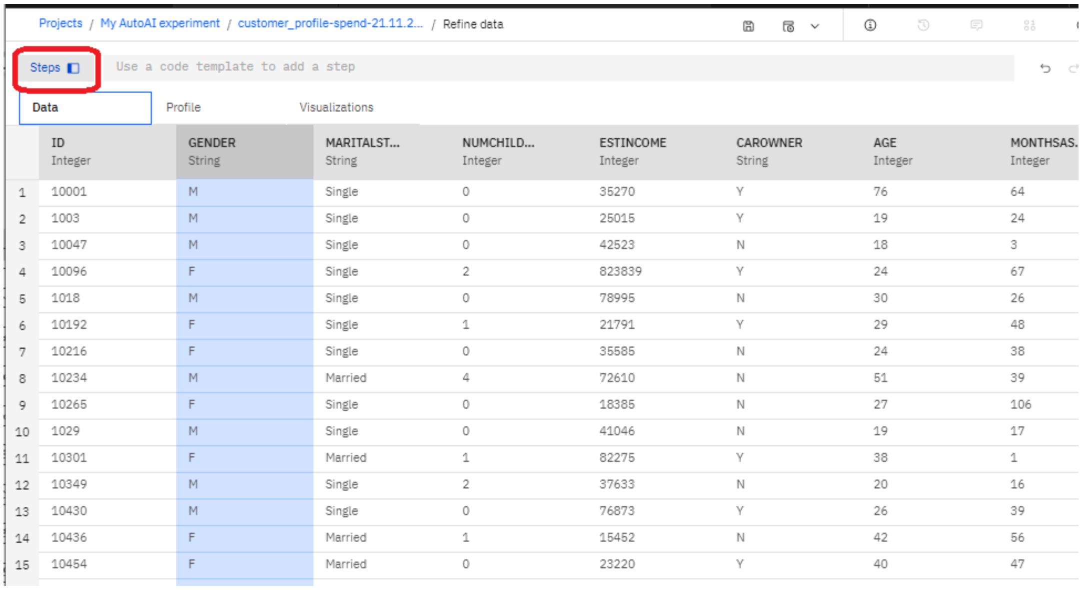
Some more examples of visualizations built using our data set:





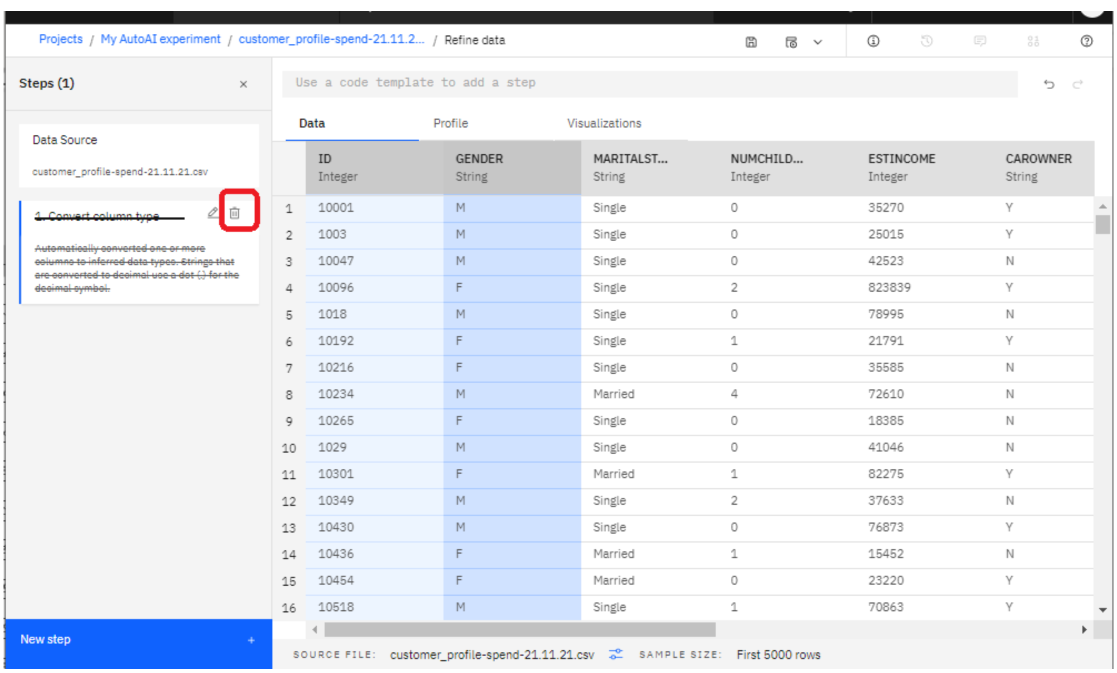
Switch back to the Data tab. We are going to address the GENDER data quality issue, data type/quality issues with some of the columns, create a new feature in our data called TOTALSPEND, and join our data set to the other csv from our project.

Expand the Steps pane.



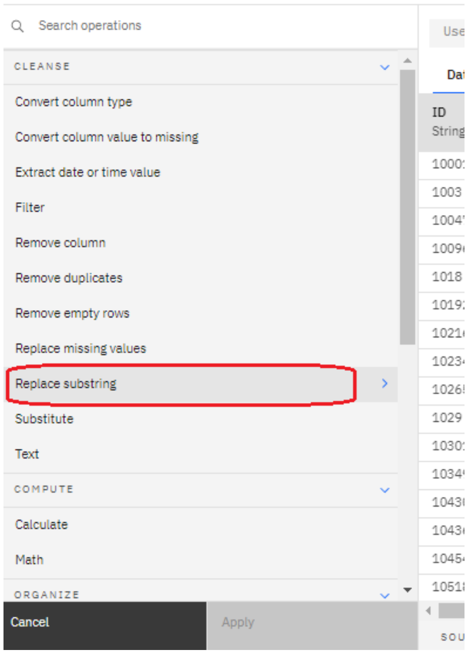
	ID Integer	GENDER String	MARITALST... String	NUMCHILD... Integer	ESTINCOME Integer	CAROWNER String	AGE Integer	MONTHSAS. Integer
1	10001	M	Single	0	35270	Y	76	64
2	1003	M	Single	0	25015	Y	19	24
3	10047	M	Single	0	42523	N	18	3
4	10096	F	Single	2	823839	Y	24	67
5	1018	M	Single	0	78995	N	30	26
6	10192	F	Single	1	21791	Y	29	48
7	10216	F	Single	0	35585	N	24	38
8	10234	M	Married	4	72610	N	51	39
9	10265	F	Single	0	18385	N	27	106
10	1029	M	Single	0	41046	N	19	17
11	10301	F	Married	1	82275	Y	38	1
12	10349	M	Single	2	37633	N	20	16
13	10430	M	Single	0	76873	Y	26	39
14	10436	F	Married	1	15452	N	42	56
15	10454	F	Married	0	23220	Y	40	47

First, let's get rid of the automated transformation step the tool did for us – Data Refinery can autoconvert column types to the best fit/most suitable ones based on the data they contain. It can prove useful, but because in our case we wanted to join the data set to another one by the ID feature later on, we need to make sure the data types of the ID column match in both of our CSVs – so the original type String would work best for us. Click on the Bin icon next to step – this will remove it. Please note that you can remove any steps we build later on the same way – e.g. if you make a mistake or decide you no longer need them.



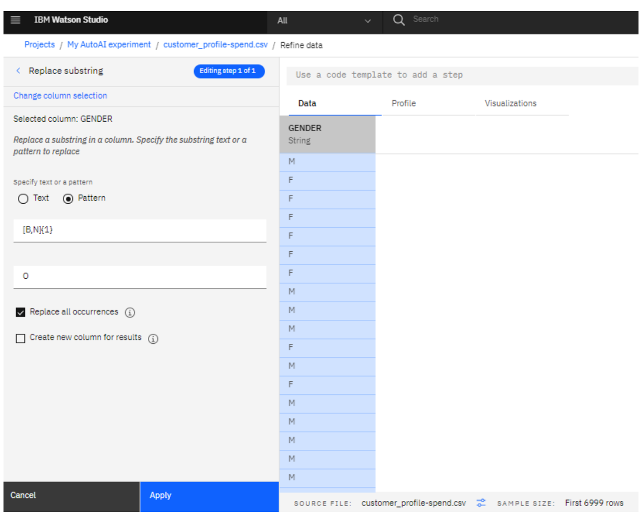
	ID Integer	GENDER String	MARITALST... String	NUMCHILD... Integer	ESTINCOME Integer	CAROWNER String
1	10001	M	Single	0	35270	Y
2	1003	M	Single	0	25015	Y
3	10047	M	Single	0	42523	N
4	10096	F	Single	2	823839	Y
5	1018	M	Single	0	78995	N
6	10192	F	Single	1	21791	Y
7	10216	F	Single	0	35585	N
8	10234	M	Married	4	72610	N
9	10265	F	Single	0	18385	N
10	1029	M	Single	0	41046	N
11	10301	F	Married	1	82275	Y
12	10349	M	Single	2	37633	N
13	10430	M	Single	0	76873	Y
14	10436	F	Married	1	15452	N
15	10454	F	Married	0	23220	Y
16	10518	M	Single	1	70863	Y

Select Replace Substring from the menu – we are going to replace our B and N entries in the GENDER column with a single new gender type of O, as we happen to know that our company’s data capture rules allow for “Other/Prefer Not to Say” option in addition to F (female) and M (male).

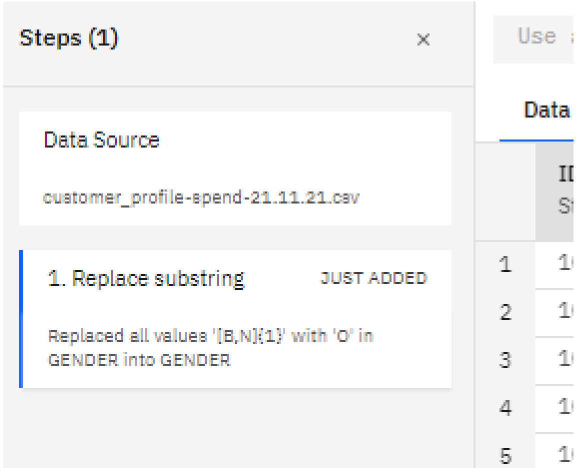


Select GENDER, click Next, switch to Pattern recognition on the next screen and enter the following values:

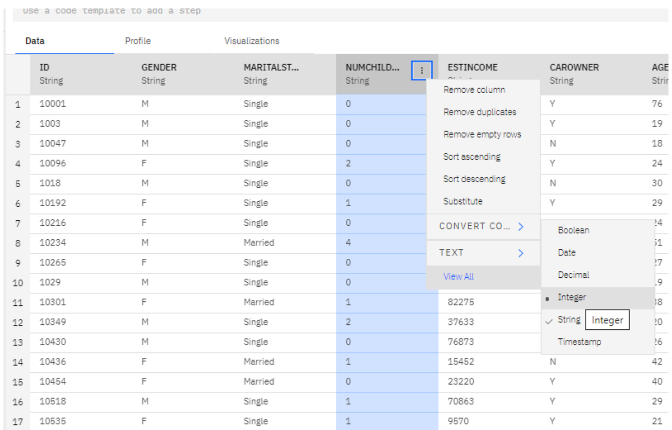
Regular expression: [B,N]{1}
Replacement string: O



Click Apply – you now have a new step in your data preparation flow.

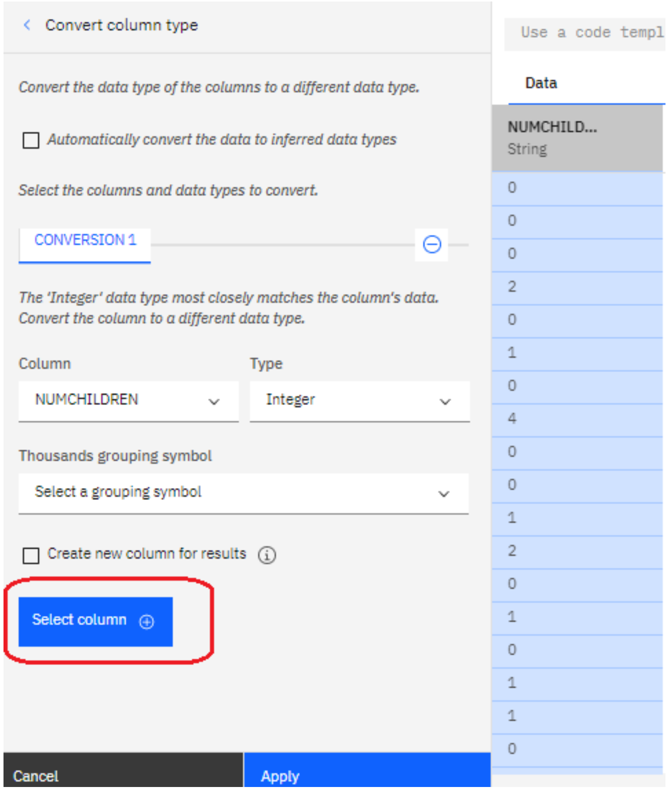


Next, we are going to convert several columns with numerical data to more appropriate data formats. Click on the three dots icon next to NUMCHILDREN column's title, select Convert Column Type > Integer.



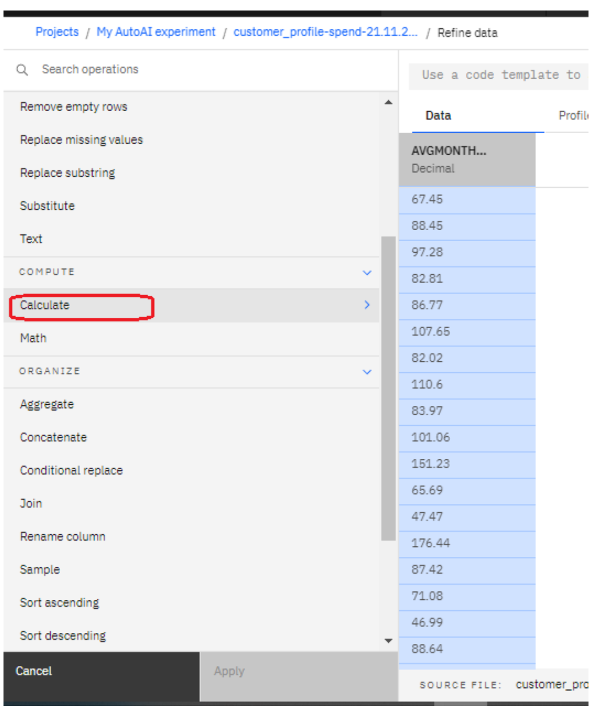
Add more columns on the next screen - click Select Column and add the following conversions:

- ESTINCOME – Integer
- AGE – Integer
- MONTHSASCUSTOMER – Integer
- LASTBILLAMOUNT – Decimal
- AVGMONTHLYSPEND - Decimal



Click Apply. Note how you can immediately see the changes in the preview, based on the transformation steps you are performing.

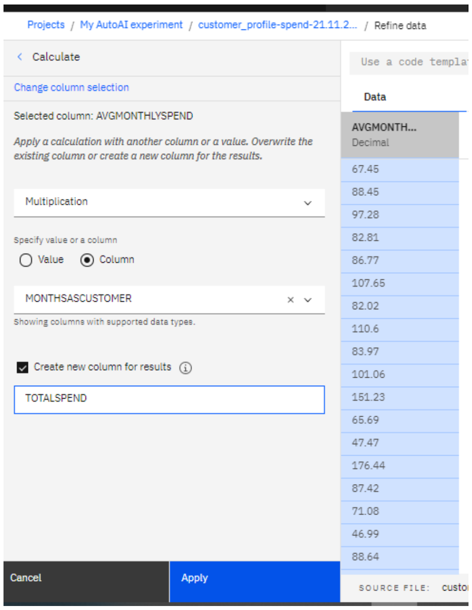
Next, we are going to create a new feature. Select the AVGMONTHLYSPEND column by clicking on it, then select Next Step – Calculate



Enter the following on the next screen:

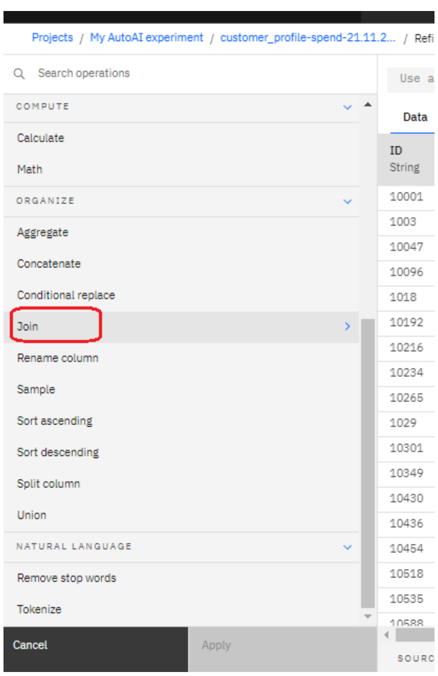
- Calculation type: Multiplication
- By: Column – MONTHSASCUSTOMER
- Create new column for results checkbox ticked (yes)
- Column name: TOTALSPEND

Click Apply.

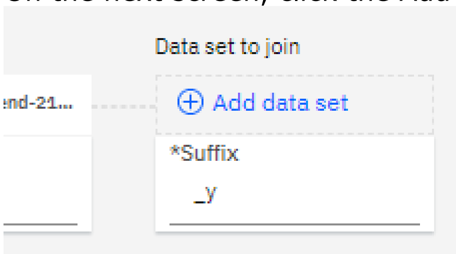


Preview now shows:

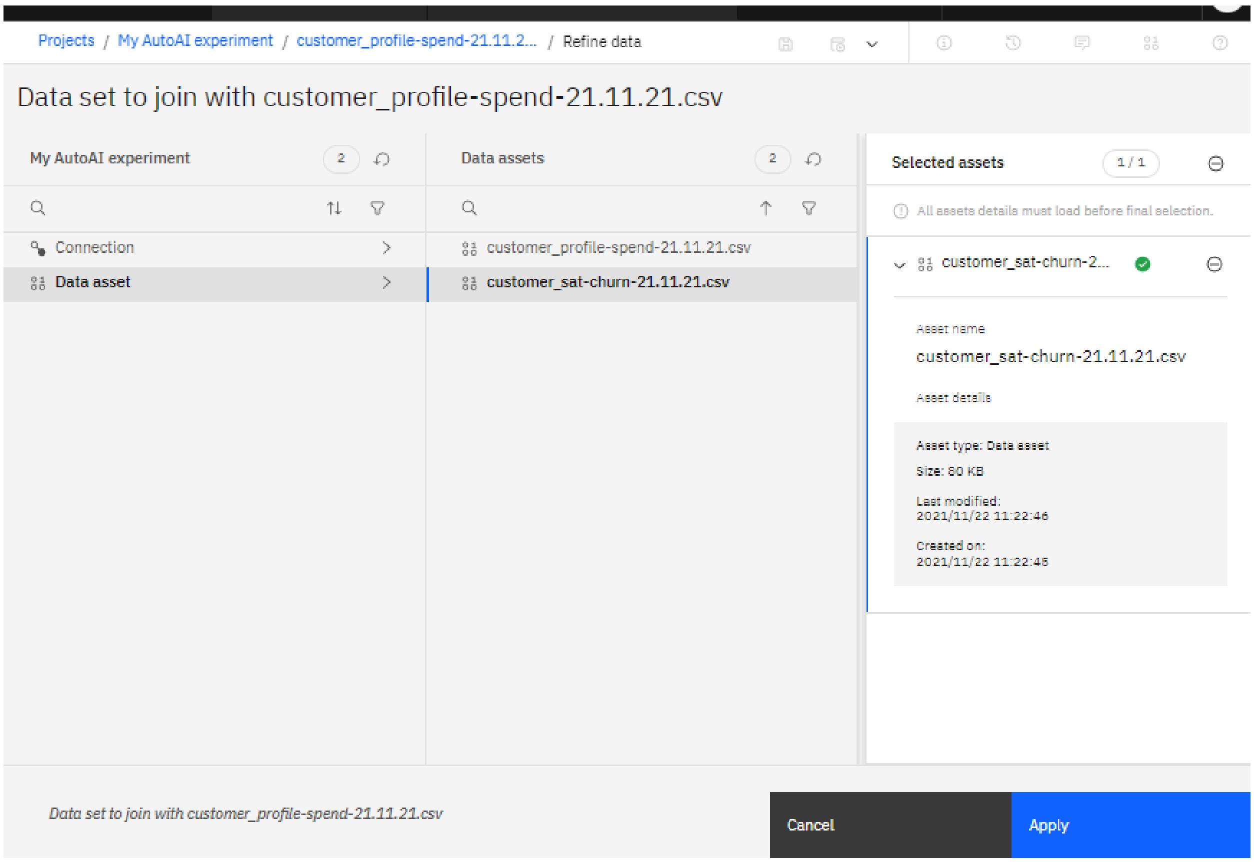
Finally, let's join our two csvs. We are going to be predicting customer CHURN and building a model for it later on – at the moment, our current data set does not include CHURN data. However, the other csv file we loaded does have it – so let's join them together.
Click Next Step – select Join from the menu.



On the next screen, click the Add Data Set link

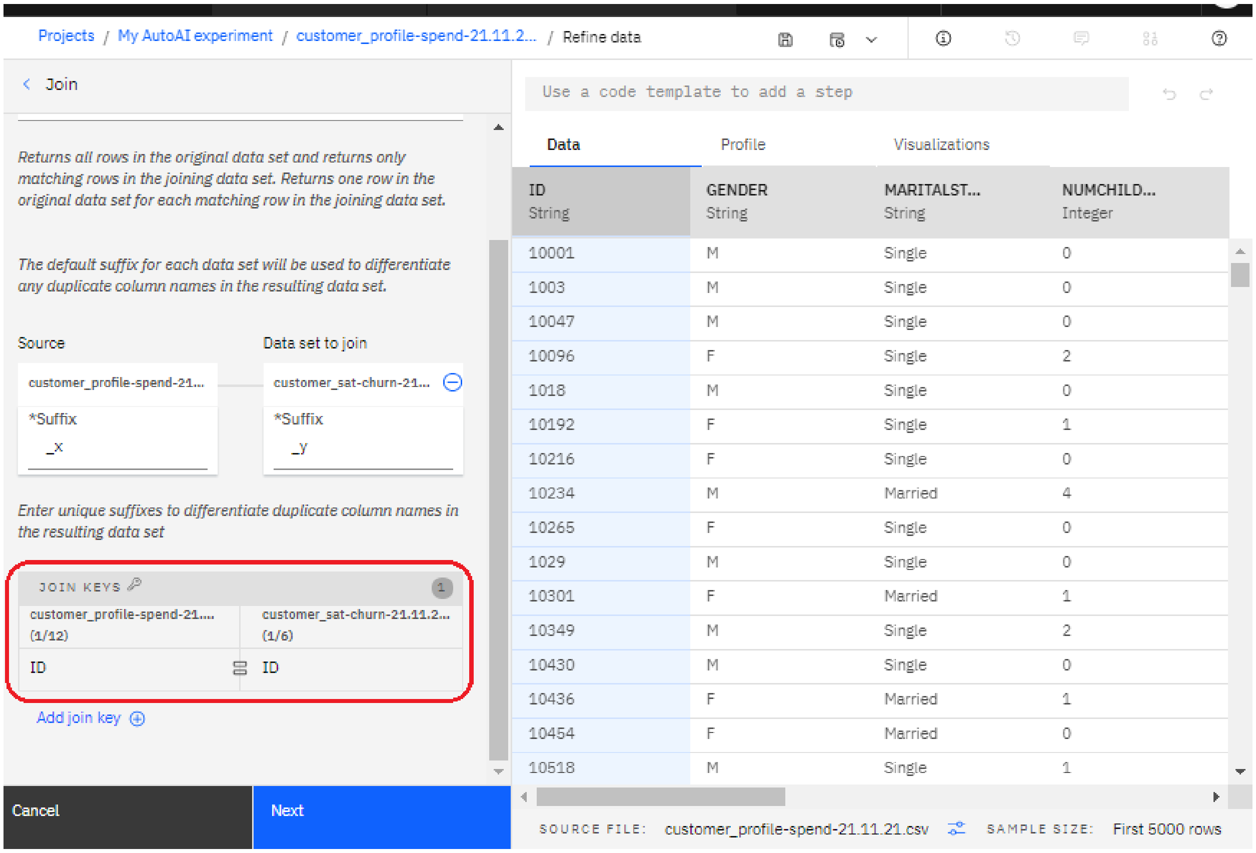


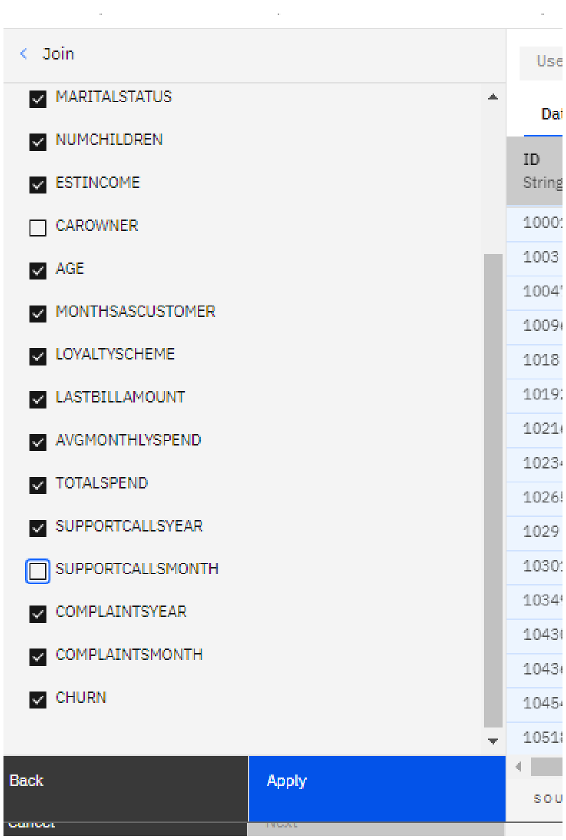
Select customer_sat-churn-21.11.21.csv (Data asset > customer_sat-churn-21.11.21.csv), then Apply.



Select ID as join keys for both the files (note that for the join to work both the columns you are using as join keys need to be of the same type – e.g. String and String, or Integer and Integer etc. – the tool will only let you pick columns of the same type once you specify your join key for the first data set). Click Next. On the next screen exclude CAROWNER and SUPPORTCALLSMONTH columns, then click Apply.

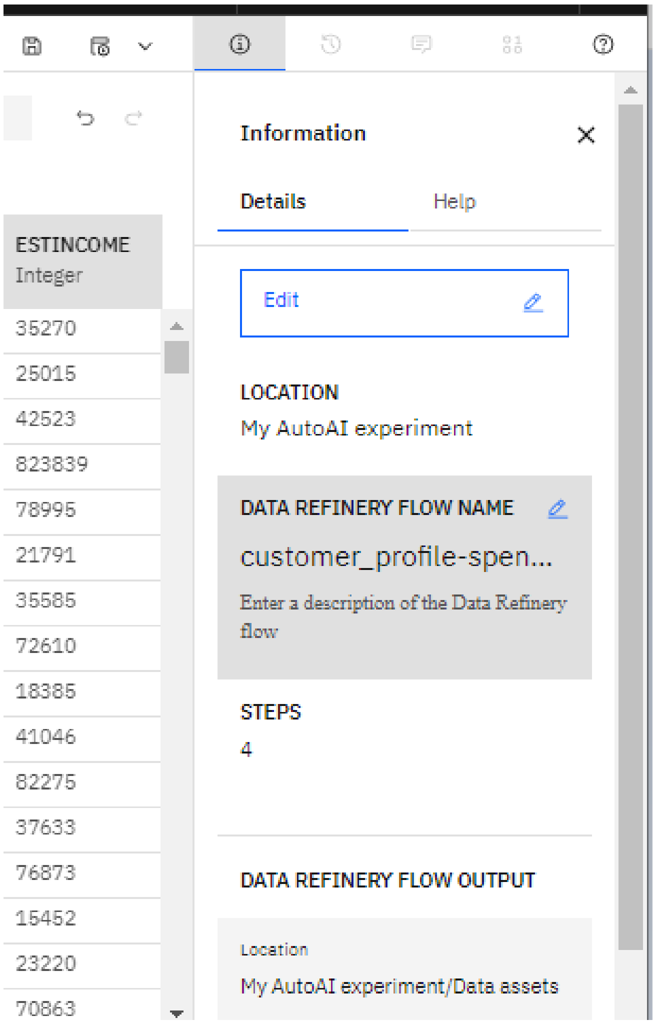
IBM

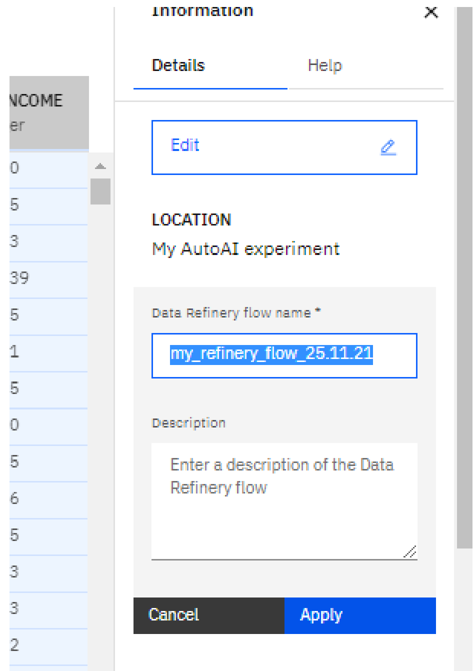

Projects / My AutoAI experiment / customer_profile-spend-21.11.2... / Refine data
Join
Use a code template to add a step
Data
Profile
Visualizations
Returns all rows in the original data set and returns only matching rows in the joining data set. Returns one row in the original data set for each matching row in the joining data set.
The default suffix for each data set will be used to differentiate any duplicate column names in the resulting data set.
Source
Data set to join
customer_profile-spend-21...
customer_sat-churn-21...
*Suffix
_x
*Suffix
_y
Enter unique suffixes to differentiate duplicate column names in the resulting data set
JOIN KEYS
customer_profile-spend-21.... (1/12)
customer_sat-churn-21.11.2... (1/6)
ID
ID
Add join key
Cancel
Next
ID String
GENDER String
MARITALST... String
NUMCHILD... Integer
10001 M Single 0
1003 M Single 0
10047 M Single 0
10096 F Single 2
1018 M Single 0
10192 F Single 1
10216 F Single 0
10234 M Married 4
10265 F Single 0
1029 M Single 0
10301 F Married 1
10349 M Single 2
10430 M Single 0
10436 F Married 1
10454 F Married 0
10518 M Single 1
SOURCE FILE: customer_profile-spend-21.11.21.csv
SAMPLE SIZE: First 5000 rows


Join
MARITALSTATUS
NUMCHILDREN
ESTINCOME
CAROWNER
AGE
MONTHSASCUSTOMER
LOYALTYSCHEME
LASTBILLAMOUNT
AVGMONTHLYSPEND
TOTALSPEND
SUPPORTCALLSYEAR
SUPPORTCALLSMONTH
COMPLAINTSYEAR
COMPLAINTSMONTH
CHURN
Back
Apply

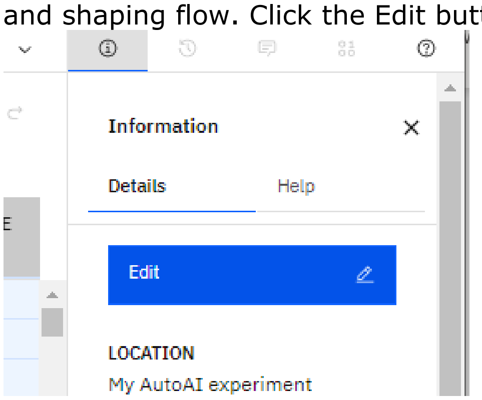
We finished building the data preparation and cleansing flow – let's now save it and run our shaping job.

First, let's give it a name and decide how and where we would want to save the output of our shaping flow. Click the i icon to expand the Information pane, then click on the pencil icon next to the data refinery flow name to edit it. Name your flow my_refinery_flow_25.11.21, then click Apply.



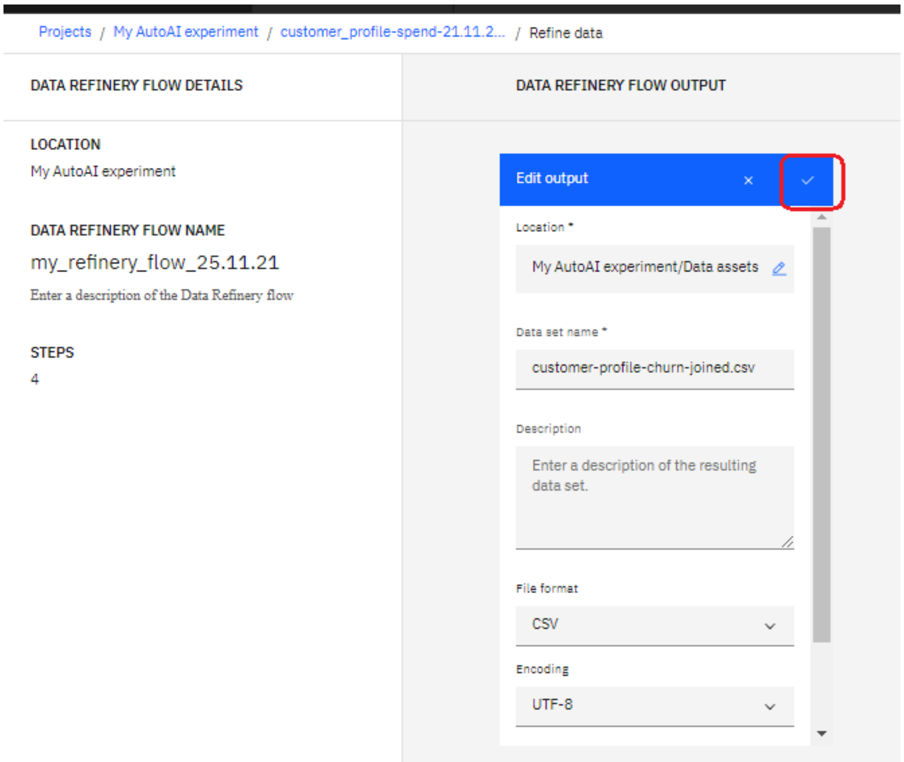


Next, let's check where and how Refinery is going to output the results of our data cleansing and shaping flow. Click the Edit button, then Edit Output on the next screen

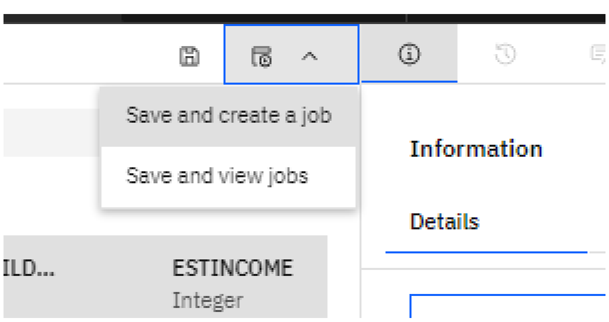


Name your data set customer-profile-churn-joined.csv

By default, Refinery will create a new csv file within your project (a new local data asset in the project). Please note that you can choose other file and format types if/where relevant, and choose to write the results to a connection (remote data source) as well – if you have any defined in the project. We are going to go with defaults - click the tickbox icon once you finish renaming the file, then click the Done button (bottom right of the screen)

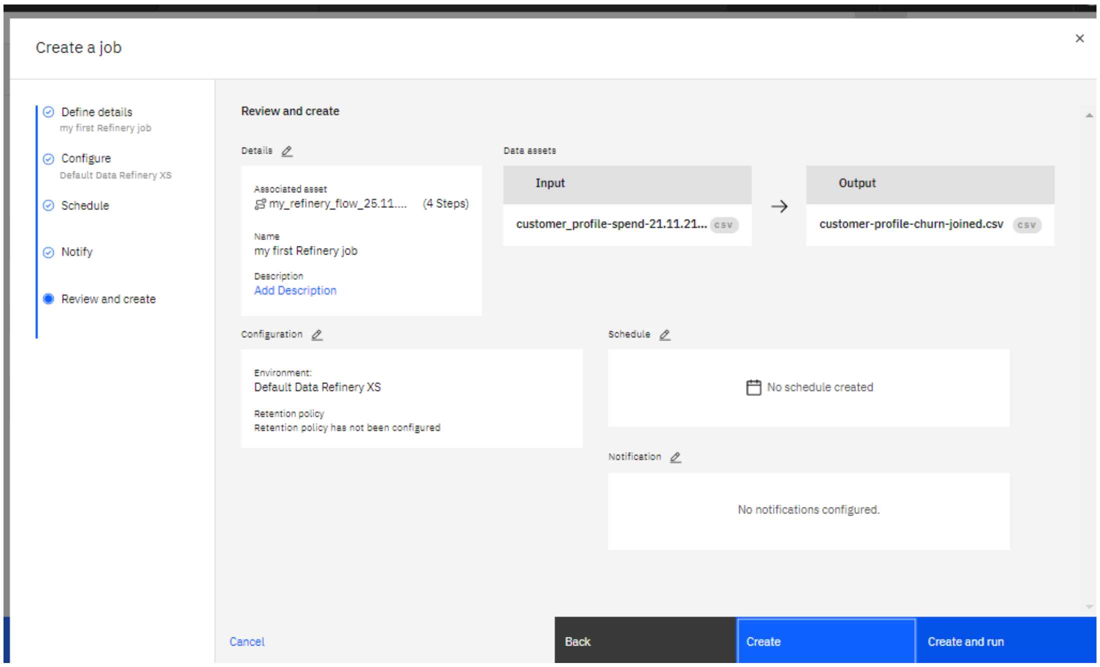


Now, let's create and run a job that will execute our flow. From the menu on the top right hand side, select Save and create a job.

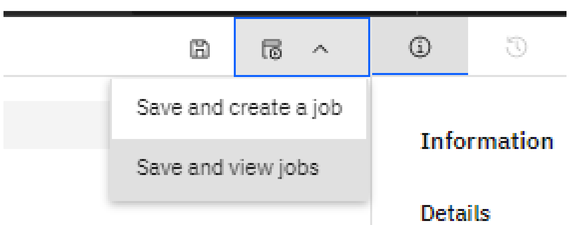


Give your job a name – e.g. "my first Refinery job", click Next. Review and go with the defaults on the next screen. Note that if you have the Analytics Engine for Spark service deployed as part of your CPDaaS, you can select different environment / runtime configurations and sizes for your Refinery jobs (e.g. create and use larger specs for more complex and data/compute intensive jobs).

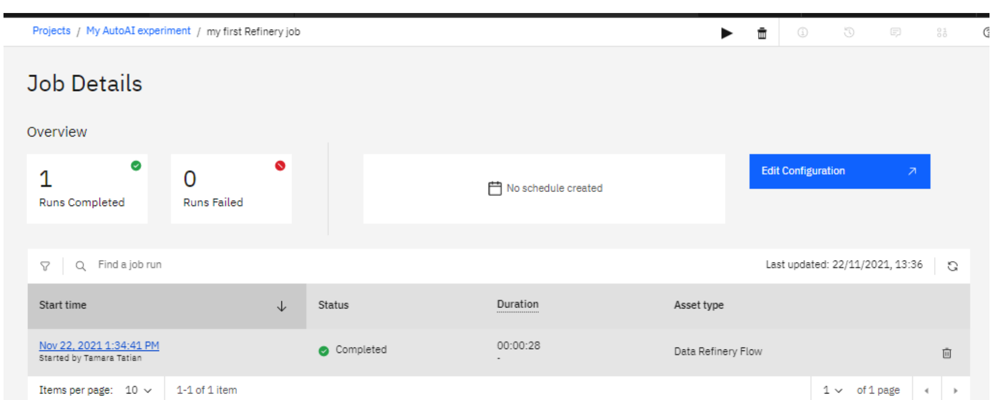
Next screen allows you to specify a schedule for your job. We are going to run it as a one-off, but feel free to explore the scheduling options. Disable the schedule before you move to the next screen. On the Notifications screen click next, and then Create and Run on the last screen.

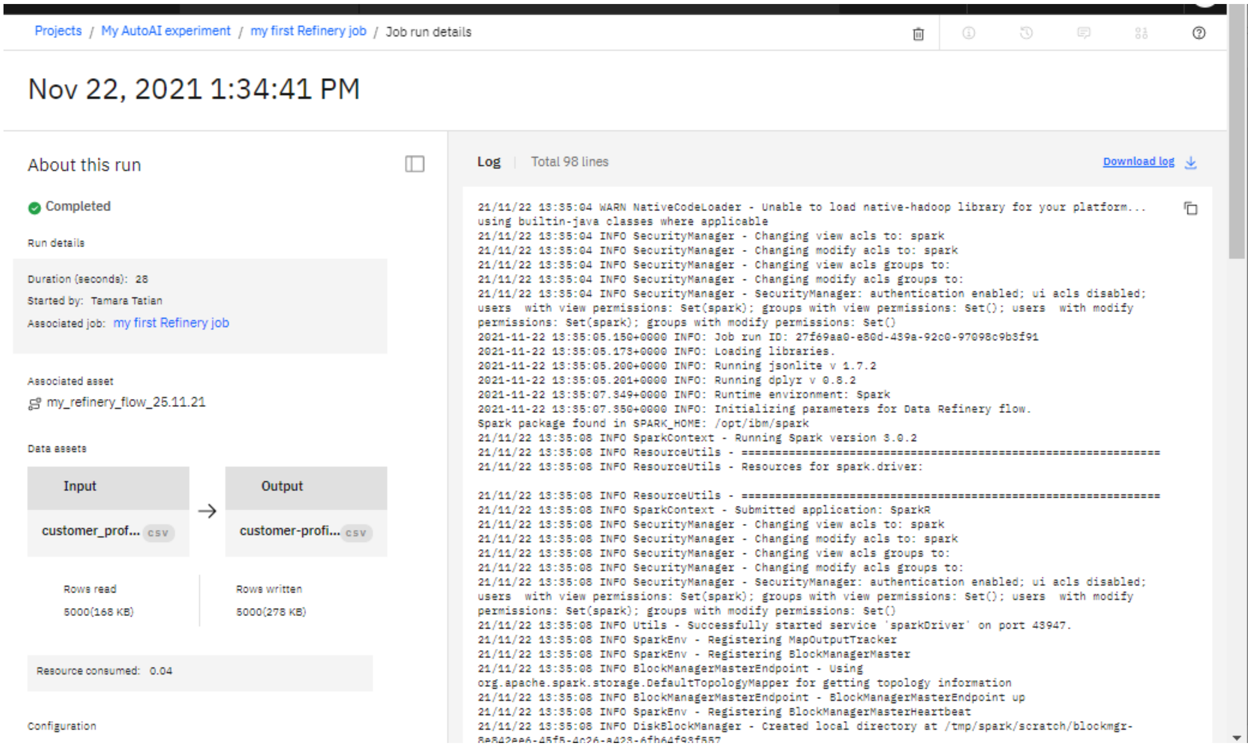


Select save and view jobs – then navigate to your newly created job

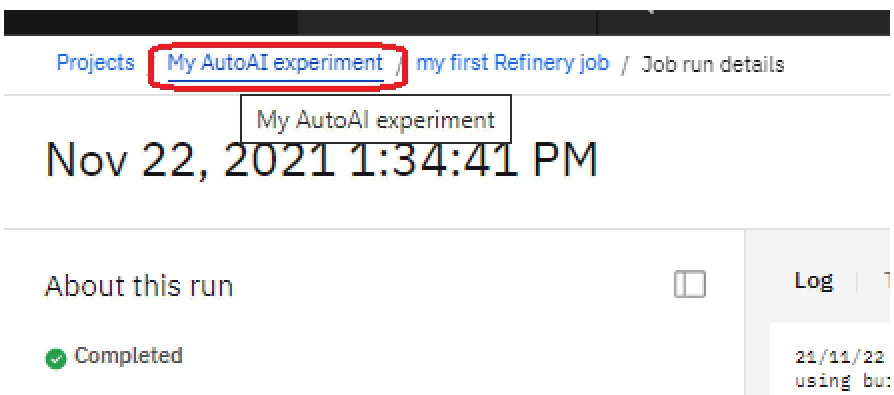


You can monitor its progress, as well as see the logs and execution details.

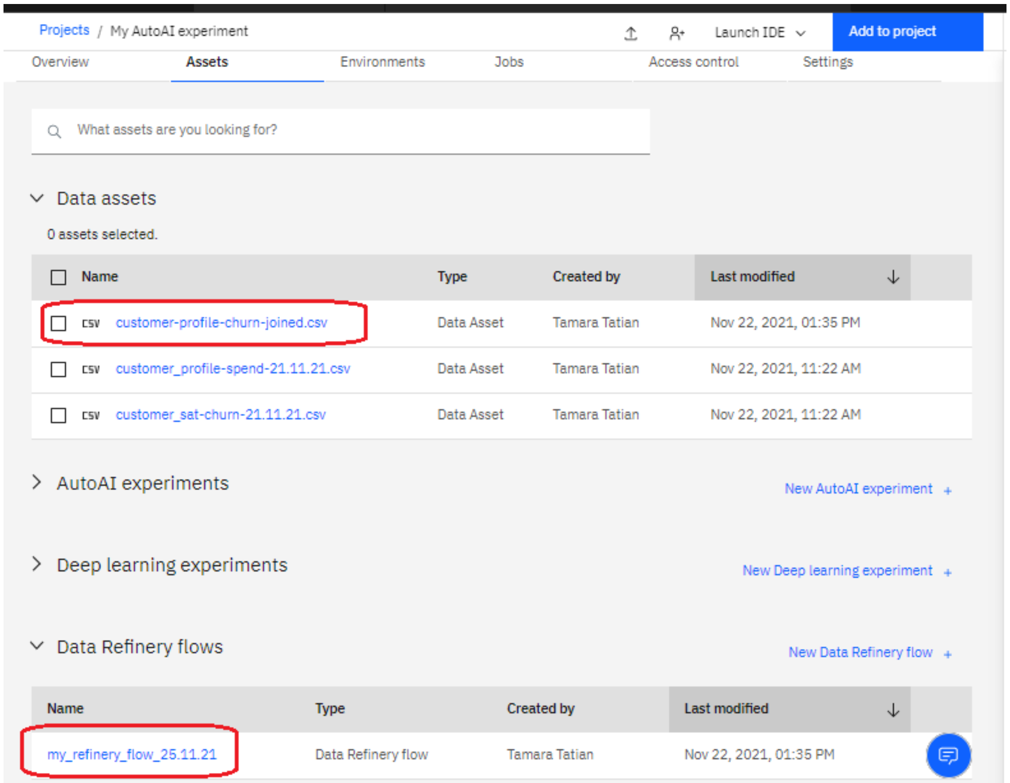


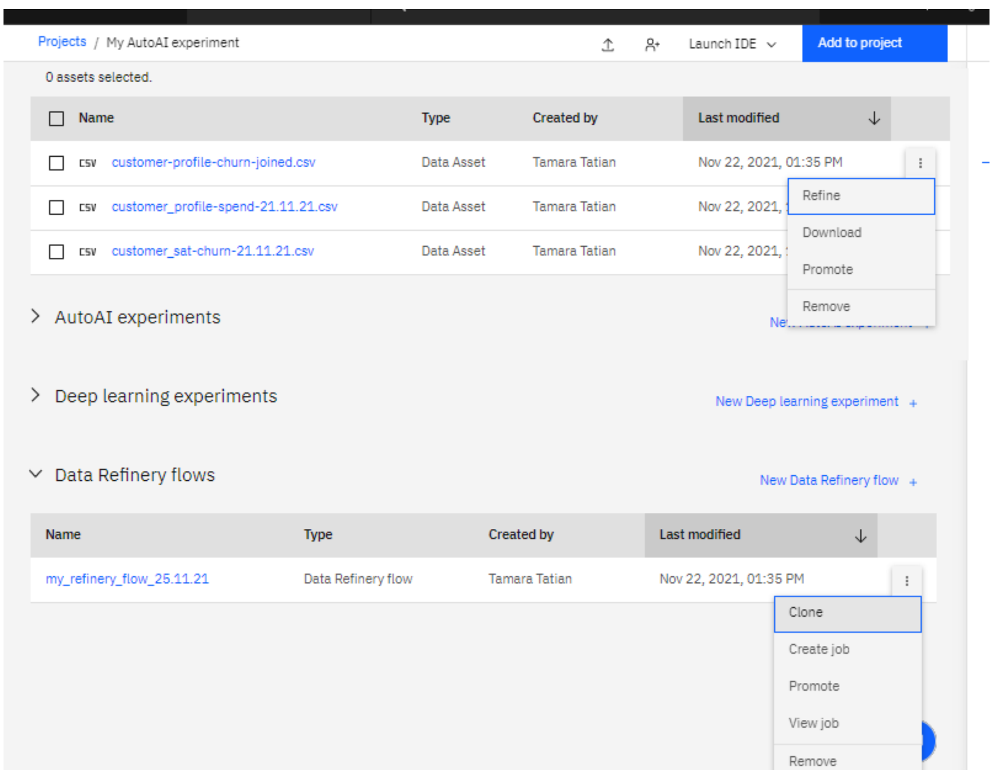


Navigate back to your project Assets view by clicking on the project name



You new data set and flow are now available – please feel free to preview the joined data set.

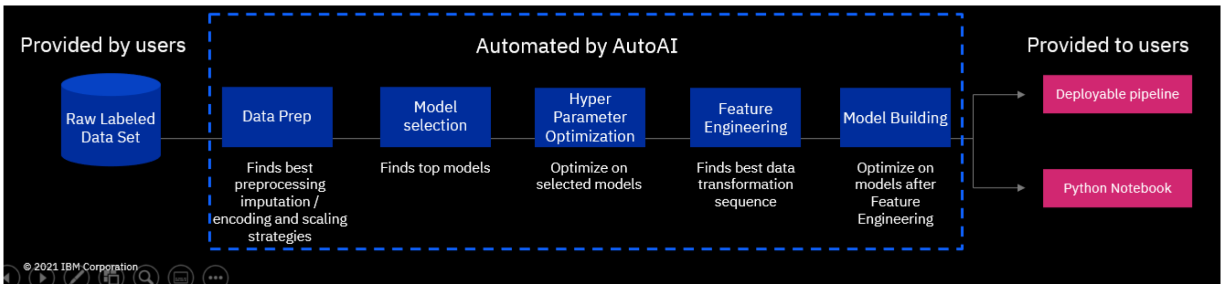

Projects / My AutoAI experiment
Launch IDE
Add to project
Overview
Assets
Environments
Jobs
Access control
Settings
What assets are you looking for?
Data assets
0 assets selected.
Name
Type
Created by
Last modified
CSV customer-profile-churn-joined.csv Data Asset Tamara Tatian Nov 22, 2021, 01:35 PM
CSV customer_profile-spend-21.11.21.csv Data Asset Tamara Tatian Nov 22, 2021, 11:22 AM
CSV customer_sat-churn-21.11.21.csv Data Asset Tamara Tatian Nov 22, 2021, 11:22 AM
AutoAI experiments
New AutoAI experiment +
Deep learning experiments
New Deep learning experiment +
Data Refinery flows
New Data Refinery flow +
Name
Type
Created by
Last modified
my_refinery_flow_25.11.21 Data Refinery flow Tamara Tatian Nov 22, 2021, 01:35 PM


Projects / My AutoAI experiment
Launch IDE
Add to project
0 assets selected.
Name
Type
Created by
Last modified
CSV customer-profile-churn-joined.csv Data Asset Tamara Tatian Nov 22, 2021, 01:35 PM
CSV customer_profile-spend-21.11.21.csv Data Asset Tamara Tatian
CSV customer_sat-churn-21.11.21.csv Data Asset Tamara Tatian
Refine
Download
Promote
Remove
AutoAI experiments
Deep learning experiments
New Deep learning experiment +
Data Refinery flows
New Data Refinery flow +
Name
Type
Created by
Last modified
my_refinery_flow_25.11.21 Data Refinery flow Tamara Tatian Nov 22, 2021, 01:35 PM
Clone
Create job
Promote
View job
Remove

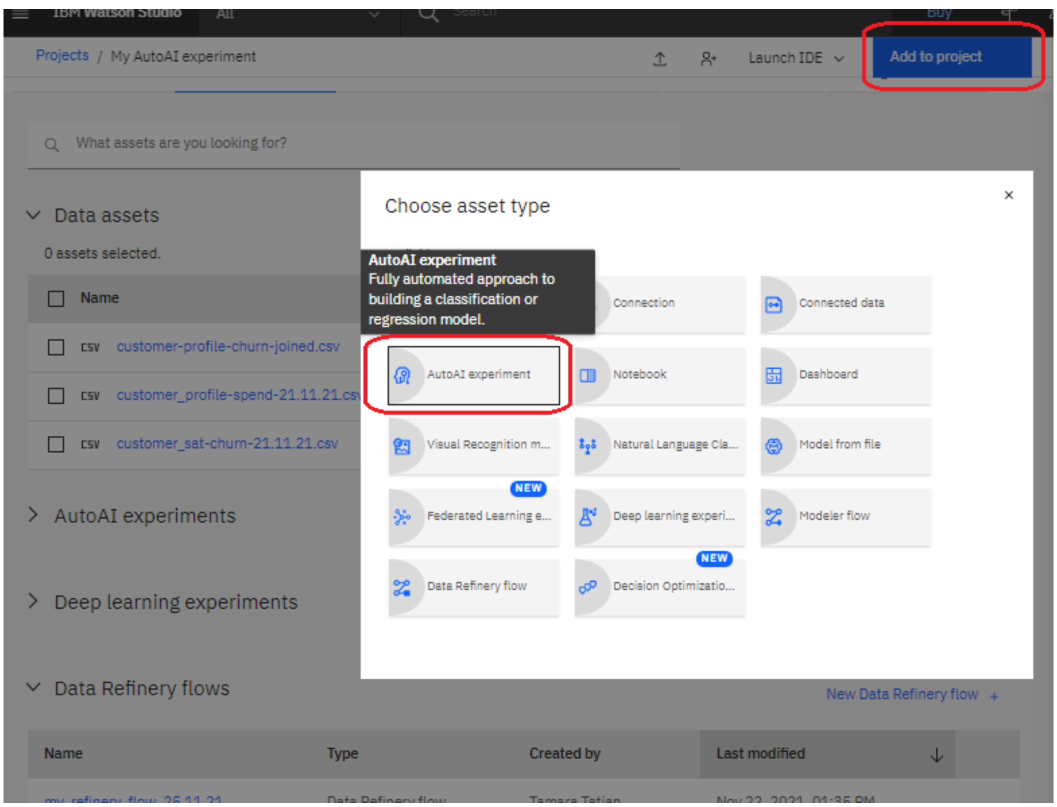
Building models with AutoAI

Next, we are going to build a model using AutoAI.

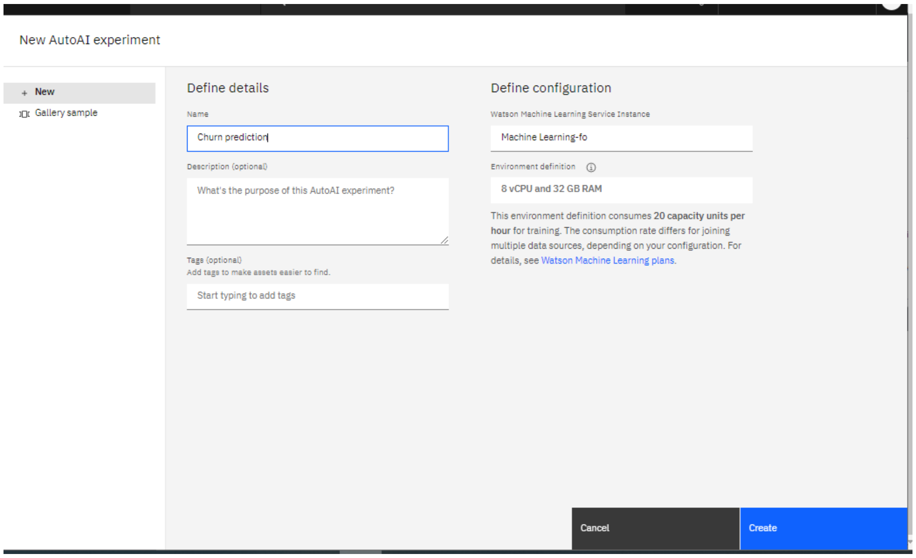
The AutoAI graphical tool in Watson Studio automatically analyses your data and generates candidate model pipelines customized for your predictive modelling problem. These model pipelines are created iteratively as AutoAI analyses your dataset and discovers data transformations, algorithms, and parameter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



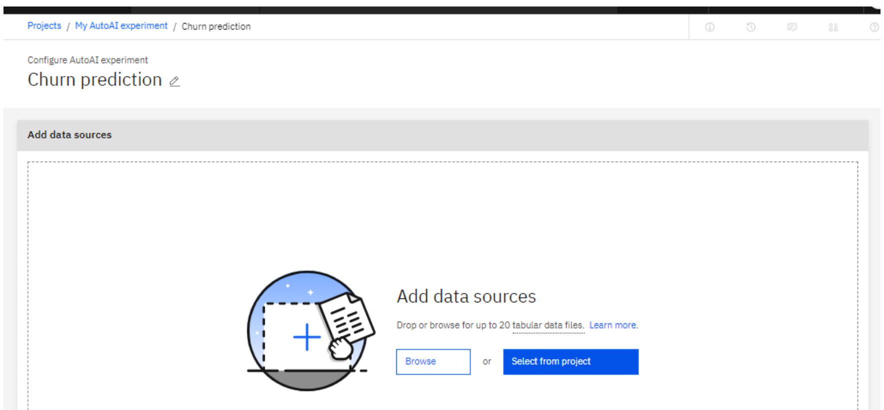
Click the Add to Project button and add a new AutoAI experiment



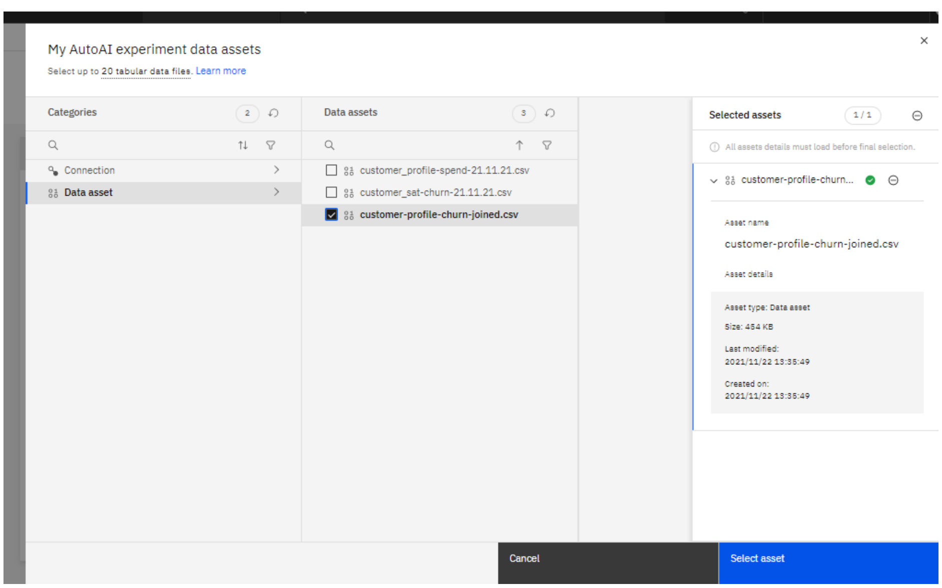
Give it a name – e.g. Churn prediction, click Create to move to the next step



Choose Select from Project on the next screen



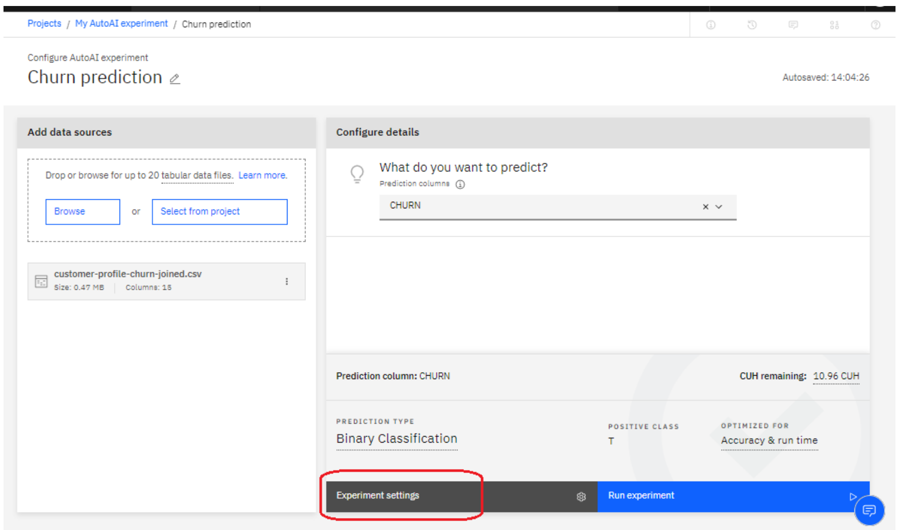
AutoAI allows you to join multiple data sets as part of the experiment set up, but since we have prepared our data already we are not going to explore that feature today. Select your newly joined data set only.



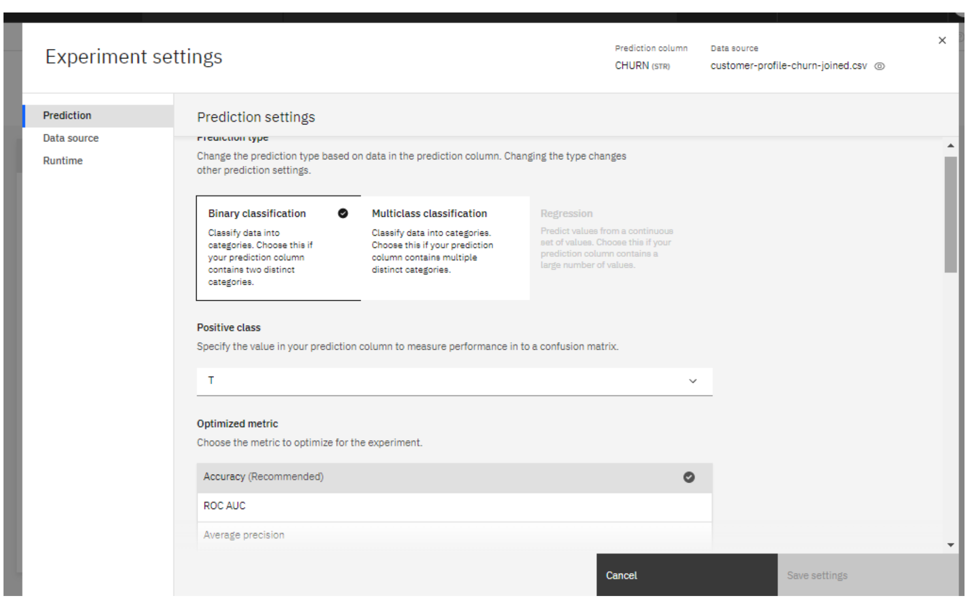
On the next screen we will need to let AutoAI know what we would like to predict (our target). In data science terminology the columns that we use to predict churn are called features, and the column that we are trying to predict is called target.

First, choose TOTALSPEND – note how the tool automatically picks Regression as the most appropriate prediction type, based on the type of data in that column (Decimal number). Next, replace your TOTALSPEND selection with GENDER (click X to clear the selection). The column has 3 possible values (we know that from our Data Refinery explorations and shaping/cleansing activities). Note how AutoAI switches prediction type to Multiclass Classification, which makes perfect sense. Finally, replace GENDER with CHURN – which we are actually looking to predict in our case.

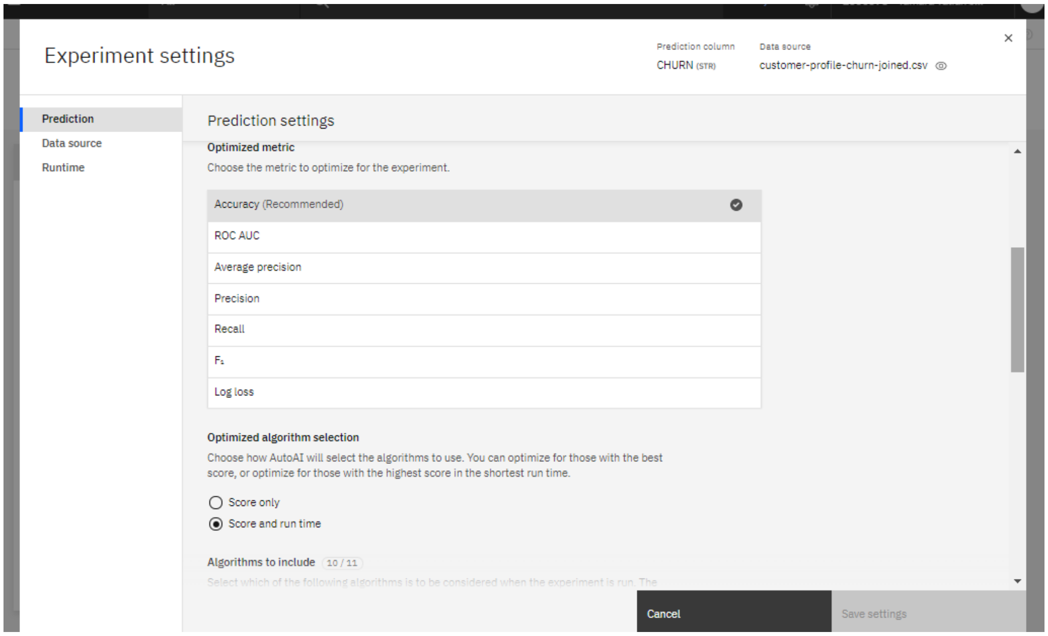
AutoAI will switch to Binary Classification (CHURN is a yes/no choice – or, more precisely True/False – customer will either leave us or will not). Let's explore additional settings – click Explore Settings



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

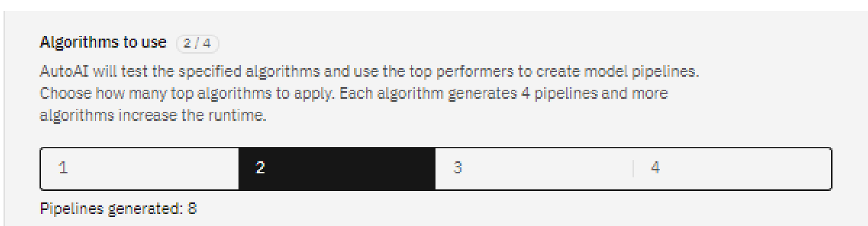


Explore settings on the screen.



Feel free to change default selections but please keep the Algorithms to use setting at 2 for your first AutoAI experiment run to keep to time.

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).

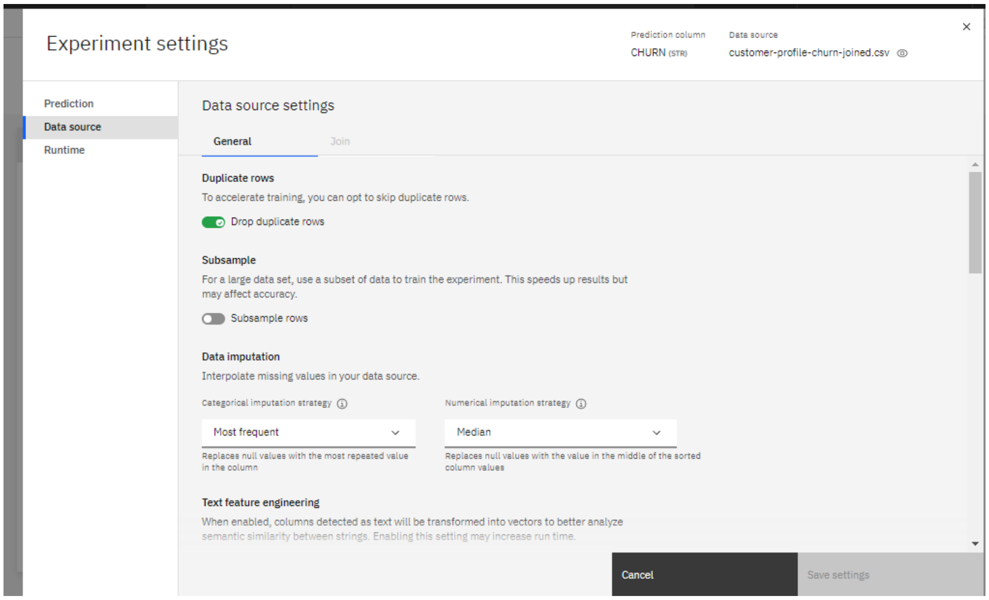


Next, switch to the Data Source tab from the menu on the left. Explore the settings.

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.

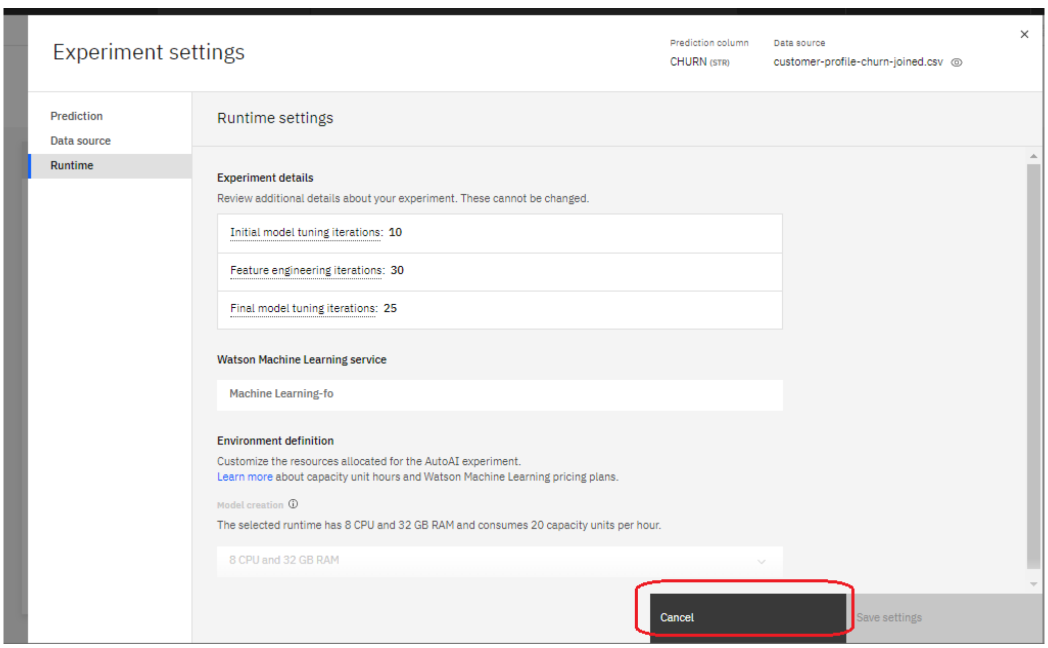
Folds are parts of the input dataset that will alternatively be used for training and testing. By specifying 3 folds, we are creating 3 parts in the dataset. We will use the default value of folds in this lab.

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

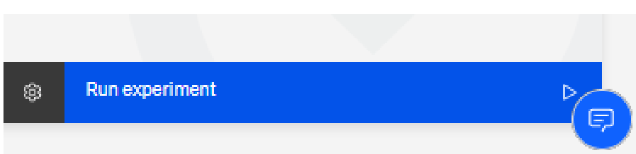


Finally, switch to the Runtime tab and review the screen.

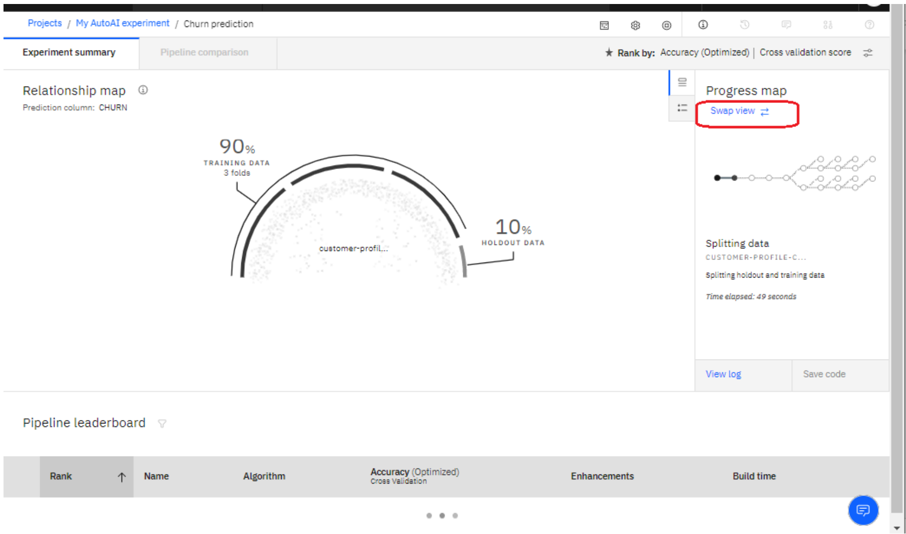
If you made any changes on the previous screens – click Save settings, if you did not – click Cancel.

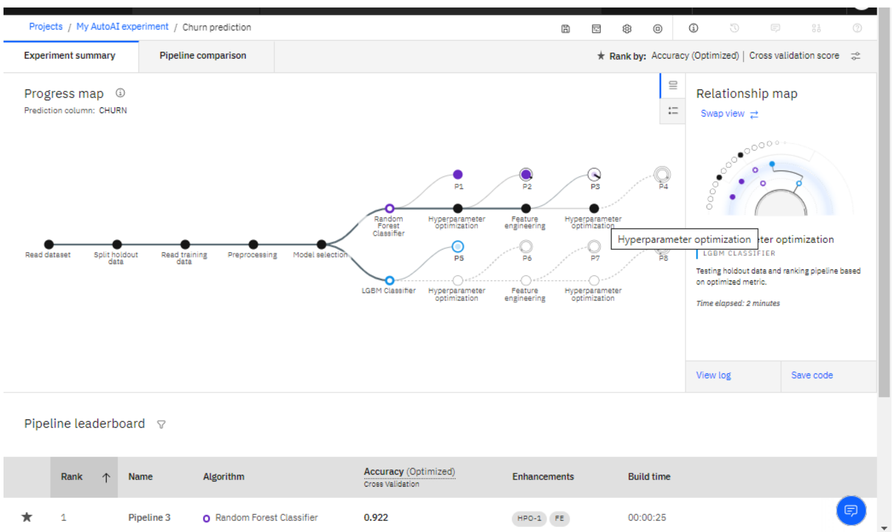


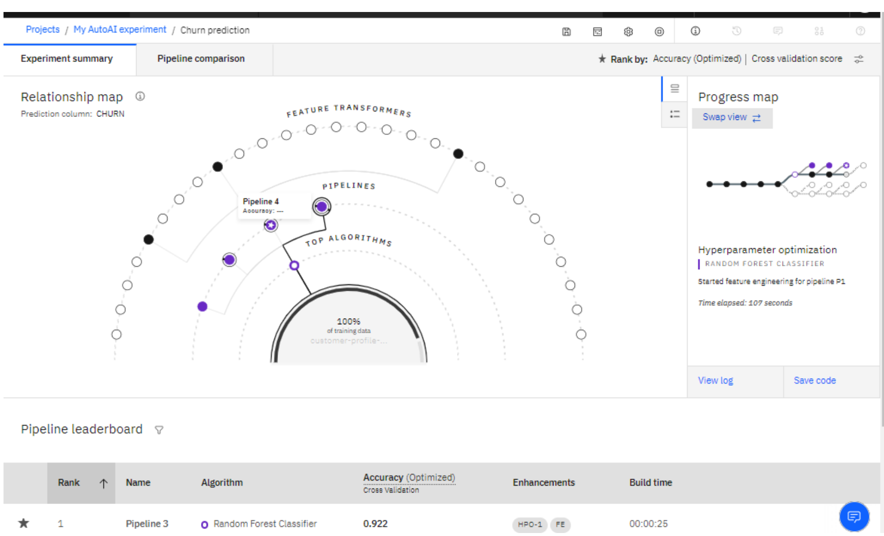
Finally, click Run Experiment



Watch AutoAI do its magic – Swap views as needed.


Projects / My AutoAI experiment / Churn prediction
Experiment summary
Pipeline comparison
Rank by: Accuracy (Optimized) | Cross validation score
Relationship map
Prediction column: CHURN
90%
TRAINING DATA
3 folds
10%
HOLDOUT DATA
customer-profil...
Progress map
Swap view
Splitting data
CUSTOMER-PROFILE-C...
Splitting holdout and training data
Time elapsed: 49 seconds
View log
Save code
Pipeline leaderboard
Rank
Name
Algorithm
Accuracy (Optimized)
Cross Validation
Enhancements
Build time


Projects / My AutoAI experiment / Churn prediction
Experiment summary
Pipeline comparison
Rank by: Accuracy (Optimized) | Cross validation score
Progress map
Prediction column: CHURN
Read dataset
Split holdout data
Read training data
Preprocessing
Model selection
Random Forest Classifier
LGBM Classifier
Hyperparameter optimization
Feature engineering
P1
P2
P3
P4
P5
P6
P7
P8
Relationship map
Swap view
Hyperparameter optimization
LGBM CLASSIFIER
Testing holdout data and ranking pipeline based on optimized metric.
Time elapsed: 2 minutes
View log
Save code
Pipeline leaderboard
Rank
Name
Algorithm
Accuracy (Optimized)
Cross Validation
Enhancements
Build time
1
Pipeline 3
Random Forest Classifier
0.922
HPO-1
FE
00:00:25


Projects / My AutoAI experiment / Churn prediction
Experiment summary
Pipeline comparison
Rank by: Accuracy (Optimized) | Cross validation score
Relationship map
Prediction column: CHURN
FEATURE TRANSFORMERS
PIPELINES
Pipeline 4
TOP ALGORITHMS
100%
of training data
Progress map
Swap view
Hyperparameter optimization
RANDOM FOREST CLASSIFIER
Started feature engineering for pipeline P1
Time elapsed: 107 seconds
View log
Save code
Pipeline leaderboard
Rank
Name
Algorithm
Accuracy (Optimized)
Cross Validation
Enhancements
Build time
1
Pipeline 3
Random Forest Classifier
0.922
HPO-1
FE
00:00:25

A *pipeline* is a term used to describe various steps in creating a model. A pipeline 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*).

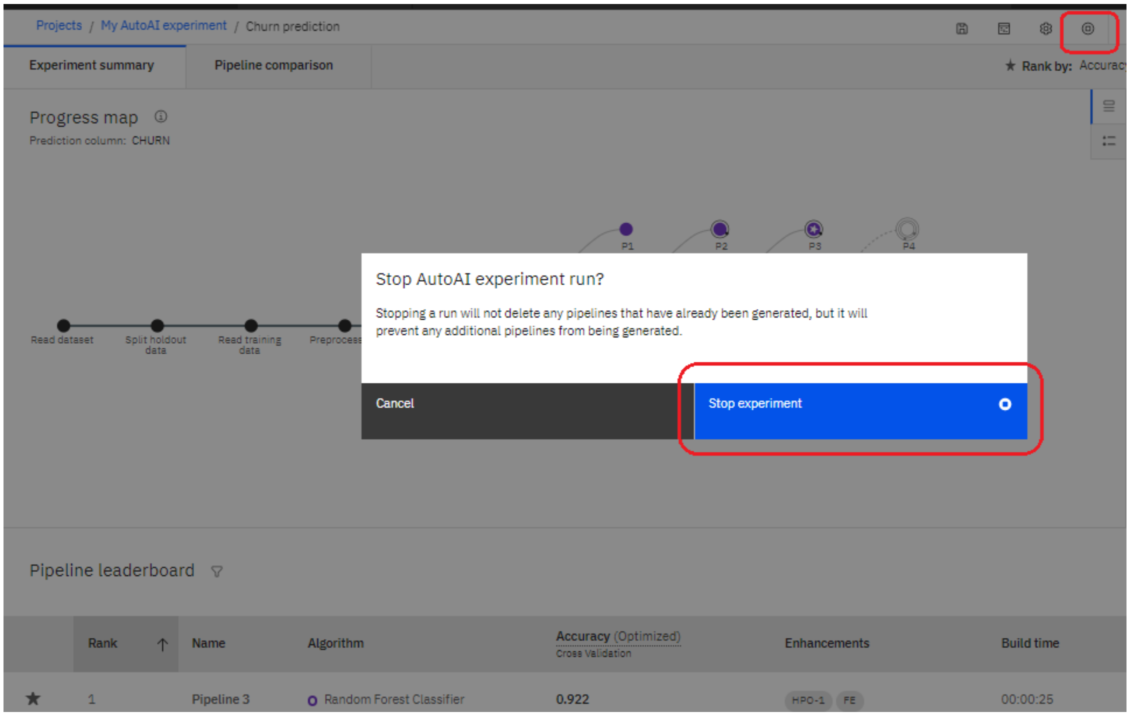
- As we can see in the graph, *P1* and *P5* build a pipeline using the *estimators* (algorithms) which AutoAI has determined to be the best fit for the data, in this case *LGBM classifier* and *Random Forest*
- *P2* and *P6* perform hyperparameter optimization (HPO). Hyperparameters are "settings" (parameters) that are specific to each algorithm. Hyperparameter optimization 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:

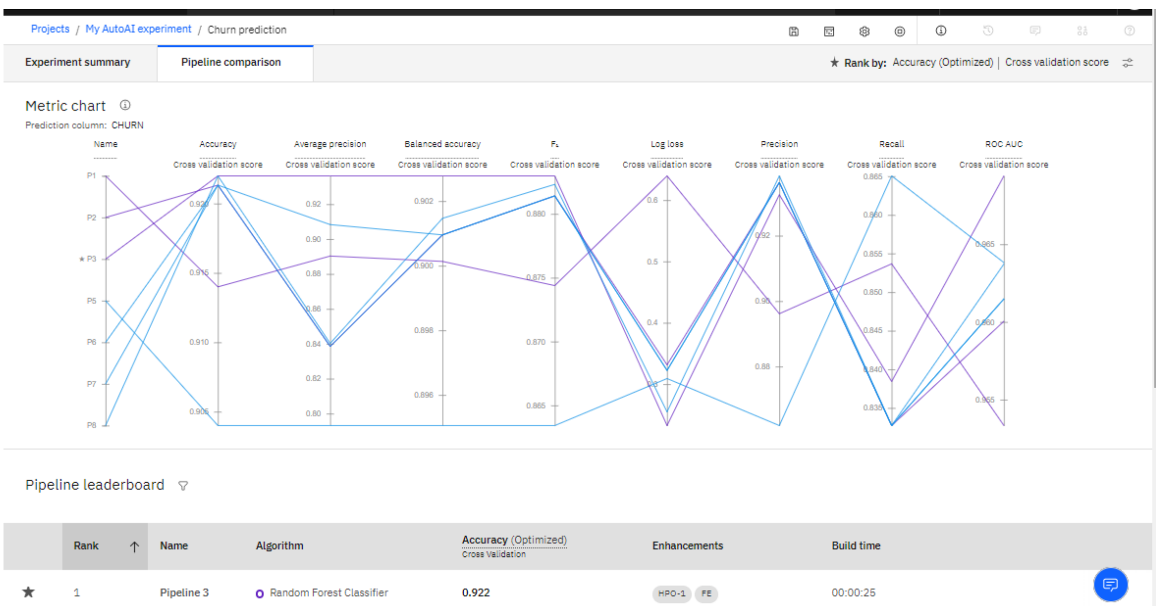
- Pipeline 1 contains just the model
- Pipeline 2 contains the algorithm and performs HPO
- Pipeline 3 generates additional features and builds a model that includes these features
- Pipeline 4 generates the same features as pipeline 3 and performs HPO.

For our experiment, all the 8 pipelines should be finished in about 3-5 minutes.

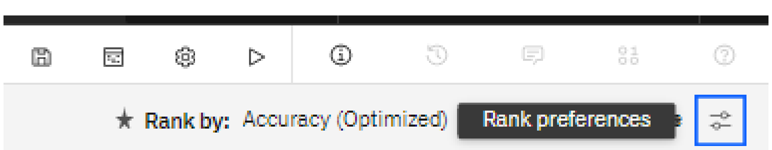
If the experiment is taking more than 5 minutes to complete, in the interest of time, please stop the experiment manually. It will keep all the already built pipelines but will not generate any new ones.

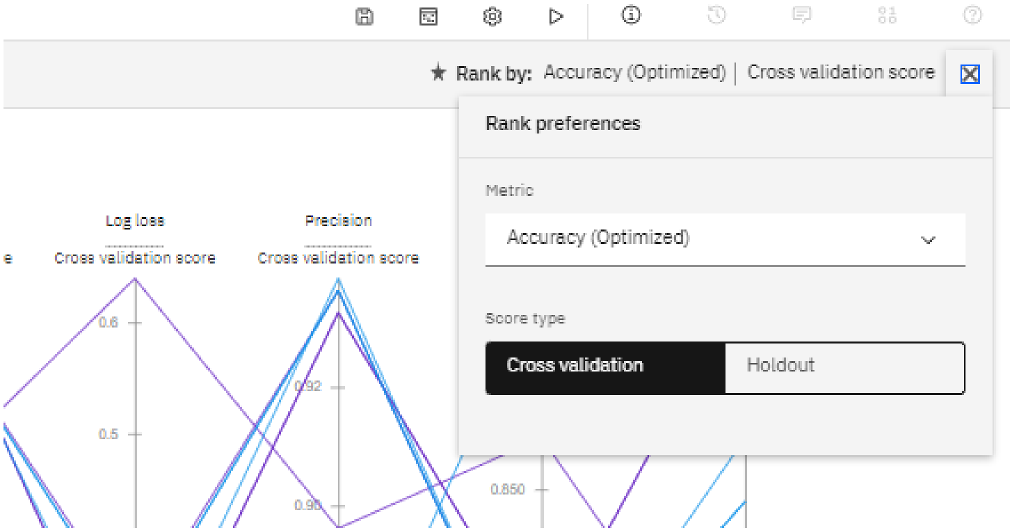


Once the experiment has finished or has been stopped, review the built pipelines and explore the metrics the platform generated. Switch to the Pipeline comparison tab.



Use the Rank preference button 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.

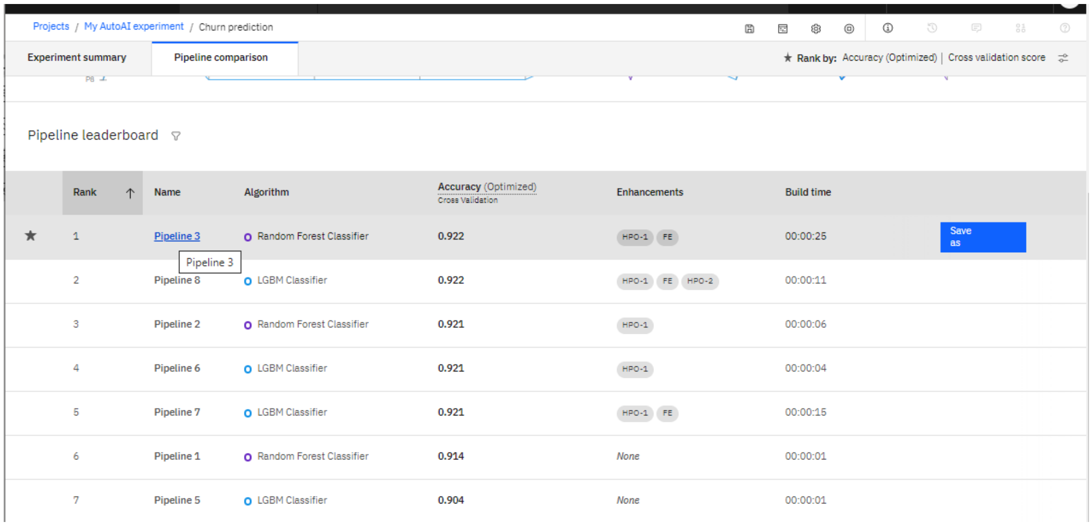
Many data scientists use *Area Under ROC Curve.* As a rough guide, you can use the following guidelines for *Area Under ROC Curve*:

.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.

AutoAI flagged Pipeline 3 as the best performing based on Accuracy (Optimized).

Click on the name of the pipeline to see further detail.



Projects / My AutoAI experiment / Churn prediction

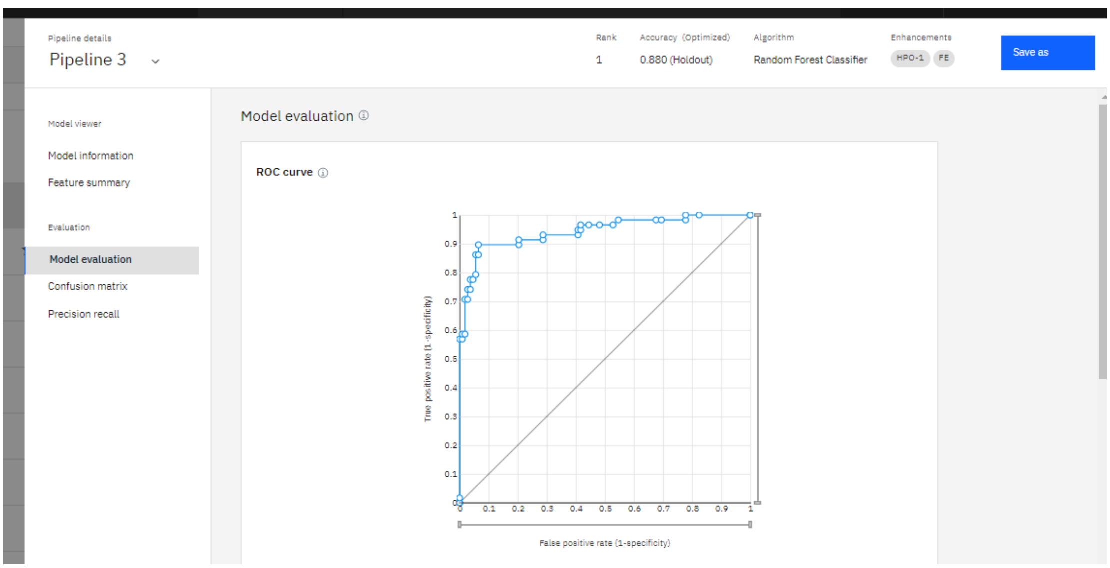
Experiment summary | Pipeline comparison

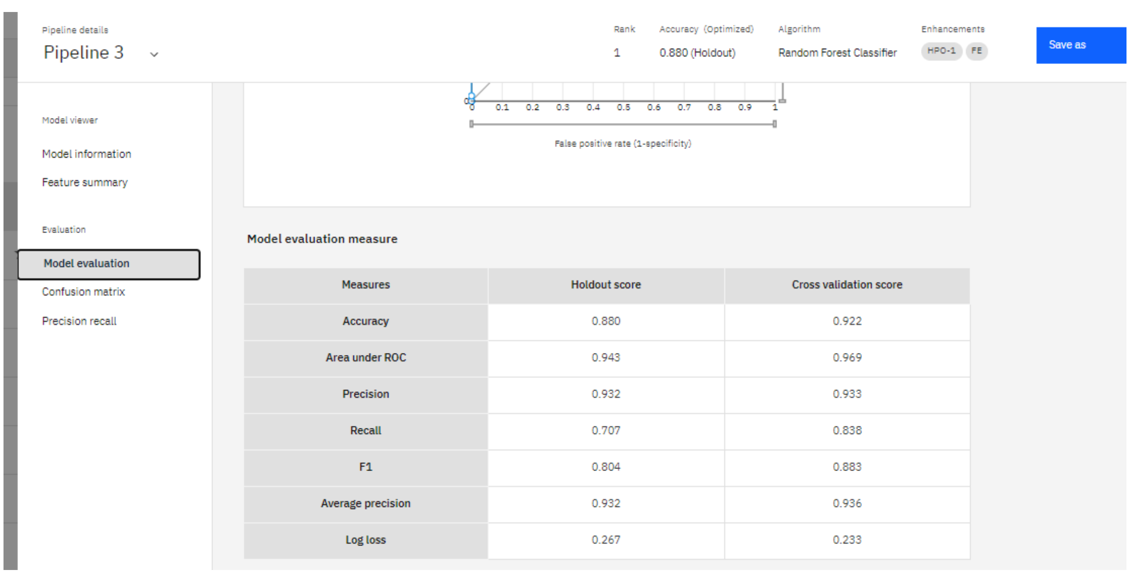
Rank by: Accuracy (Optimized) | Cross validation score

Pipeline leaderboard

Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time
★ 1	Pipeline 3	Random Forest Classifier	0.922	HPO-1 FE	00:00:25
2	Pipeline 8	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.





Pipeline details

Pipeline 3

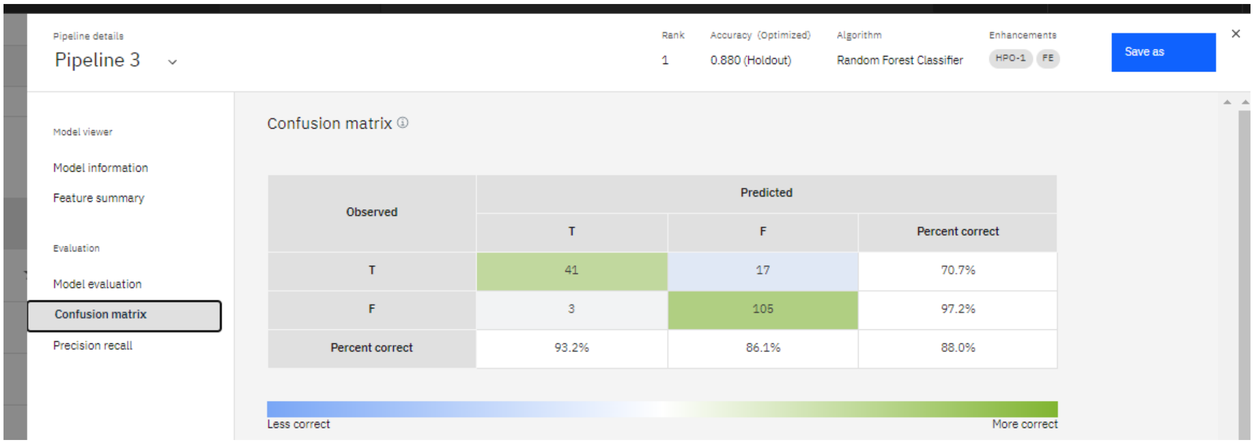
Rank 1 | Accuracy (Optimized) 0.880 (Holdout) | Algorithm Random Forest Classifier | Enhancements HPO-1 FE | Save as

Model viewer: Model information, Feature summary

Evaluation: Model evaluation, Confusion matrix, Precision recall

Model evaluation measure

Measures	Holdout score	Cross validation score
Accuracy	0.880	0.922
Area under ROC	0.943	0.969
Precision	0.932	0.933
Recall	0.707	0.838
F1	0.804	0.883
Average precision	0.932	0.936
Log loss	0.267	0.233

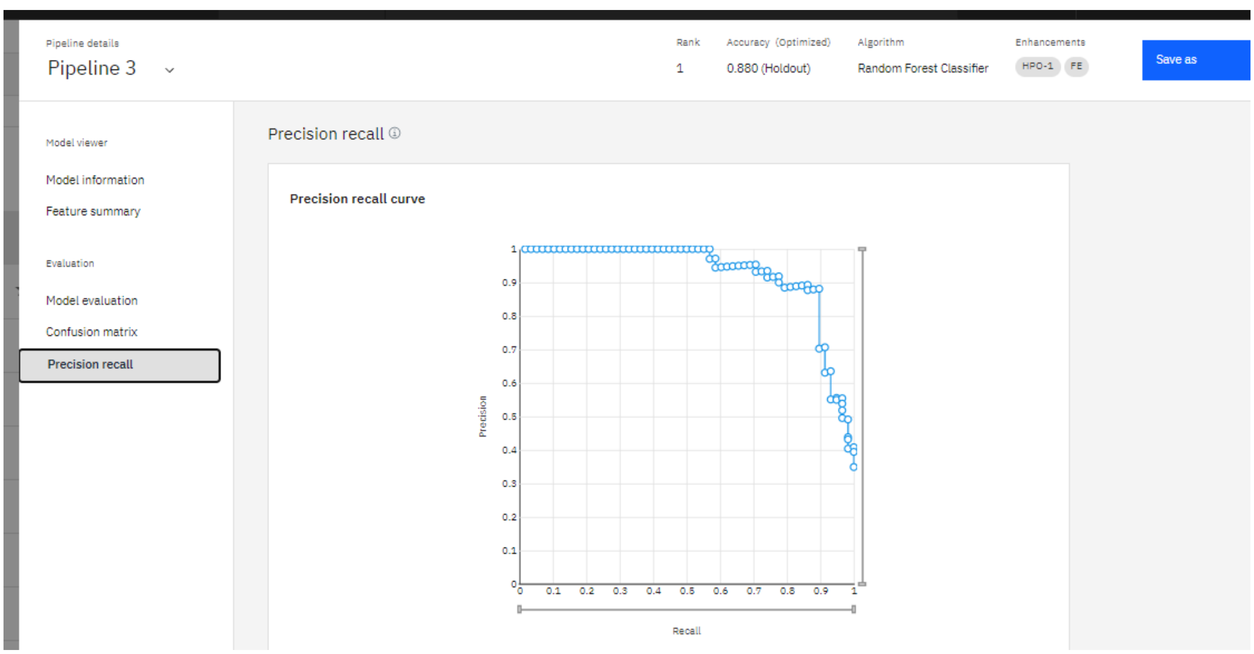


Confusion matrix is another typical metric for model evaluation. It shows *true positives, false positives, true negatives, and false negatives*.

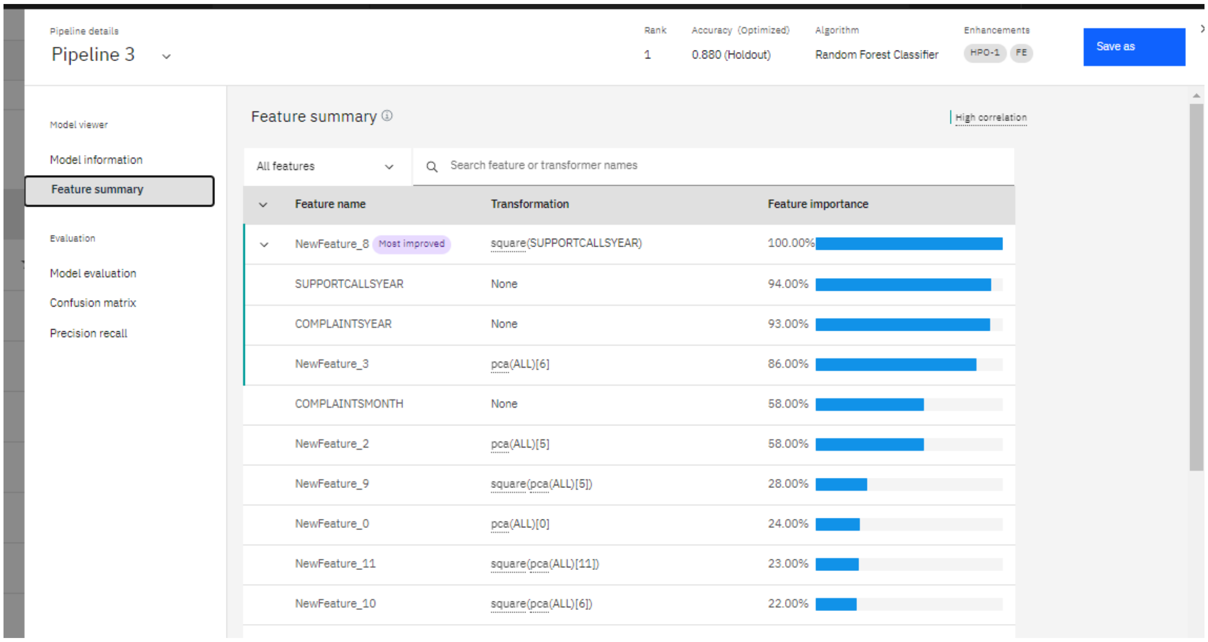
For example, the model predicted 105 of *F* (*False* – customer did not churn) records as *F* (correct prediction). At the same time, it predicted 3 records as T (the customer will churn), but the actual value was *F* (false positives). Overall, for the *False* prediction the accuracy is 97.2%. The accuracy for *True* prediction is only 70.7 percent.

If we accept that very few models are 100% accurate, and therefore will have errors in their predictions, often the goal of the confusion matrix is to help the business user under the "cost" of a misclassification (the 17 and 3 in the blue and grey cells) as there is often a "preferred" type of error for your specific use-case.

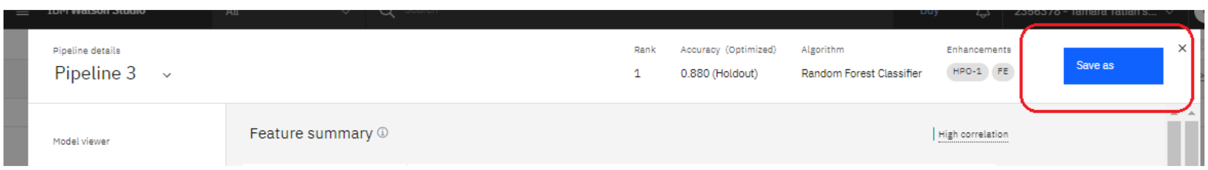
It's usually a business decision to determine what's more important – better accuracy in identifying true positives or reducing the number of false positives. For example, in a customer churn scenario identifying a false positive may mean that a customer will receive a promotion when they should not have received it. If there is no business impact of this prediction, then the model with the highest accuracy of true positives should be selected.

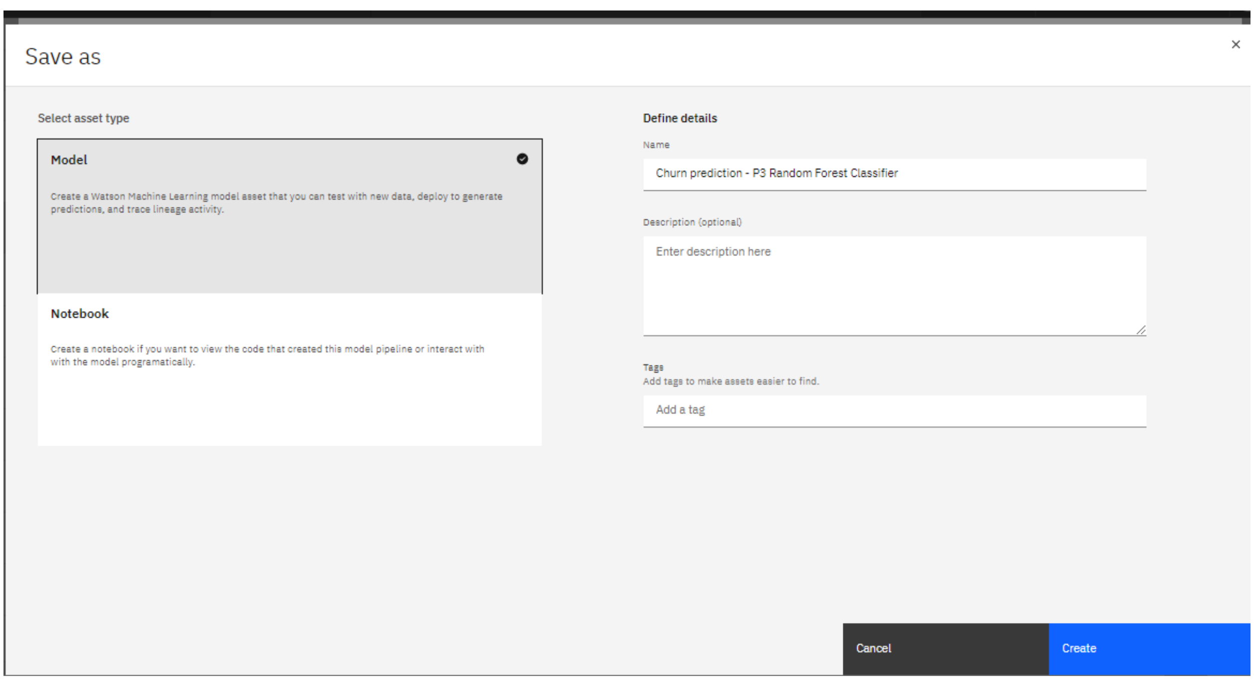


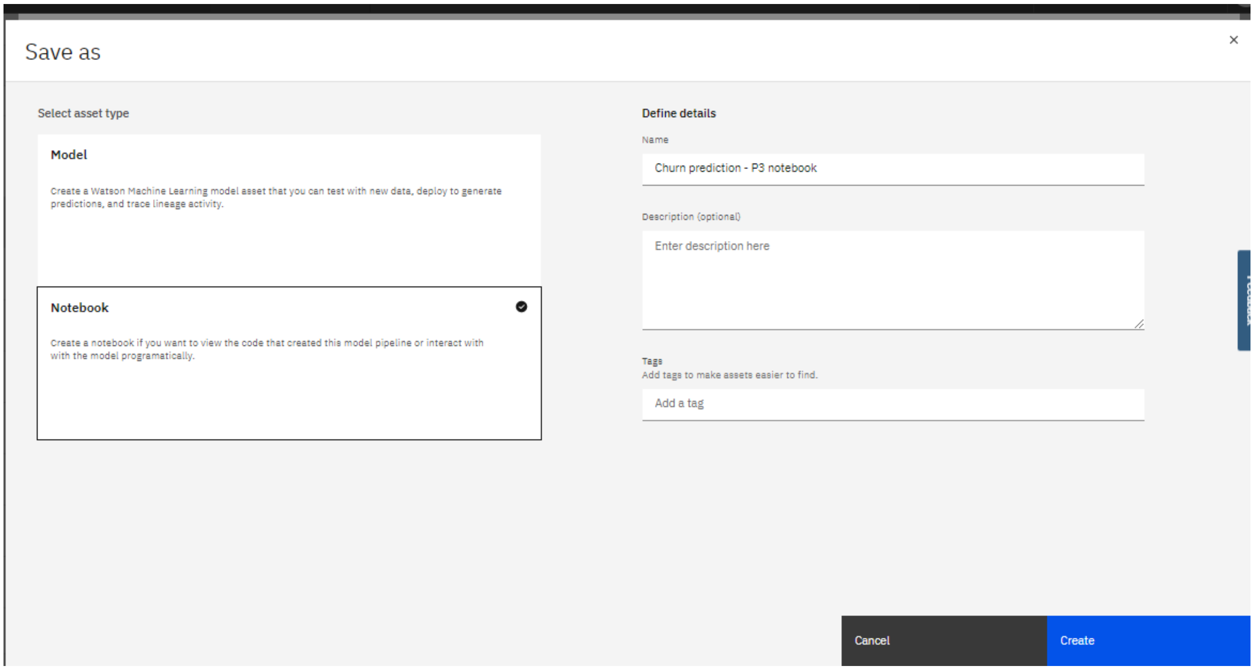
Switch to the Feature importance tab. Here we can see which features make the most impact on prediction. Notice that the model uses both features that were provided with the input data set and the features that were generated by AutoAI (start with *NewFeature*)



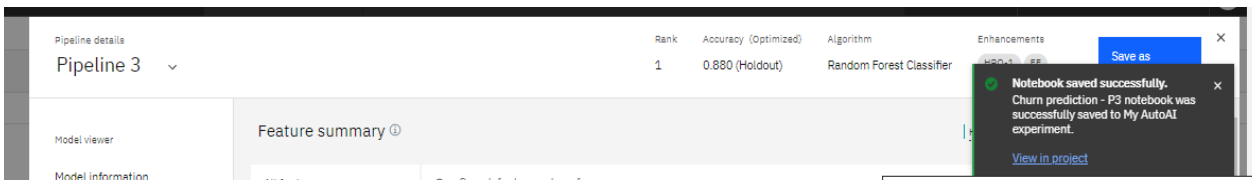
Let’s now save the pipeline – first as a model, then as a notebook.



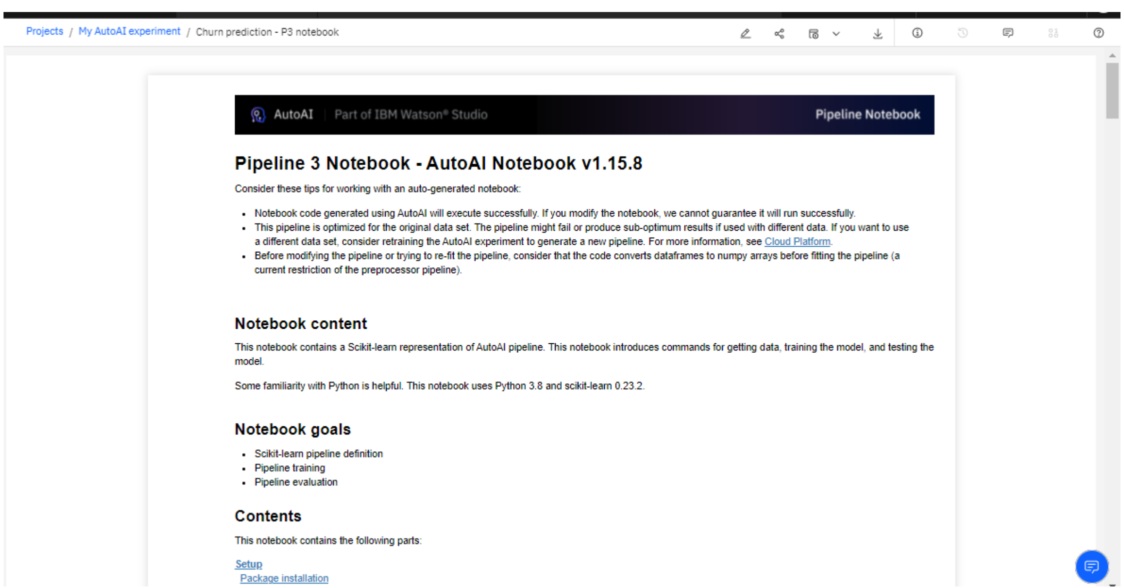




Click View in Project link for the notebook you just saved.

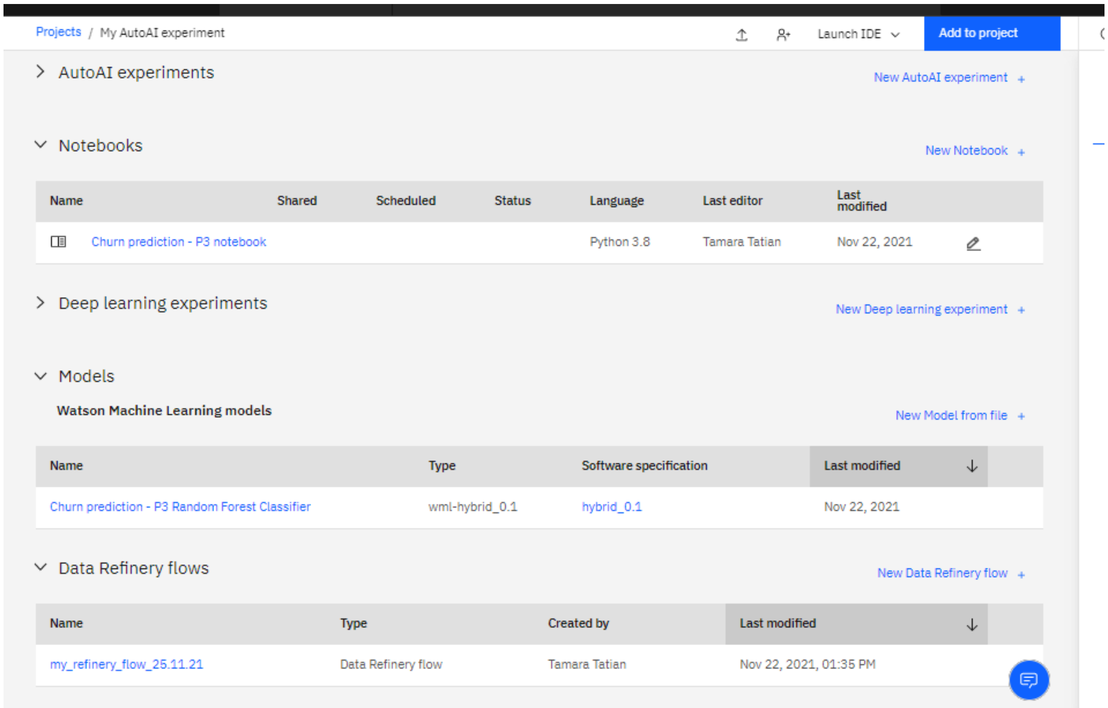


Review the notebook – AutoAI is not a black box, it produces well annotated code that you can take as-is or choose to develop further. You can save individual pipelines or the whole experiment as code (create a notebook of the full experiment to train all pipelines programatically)



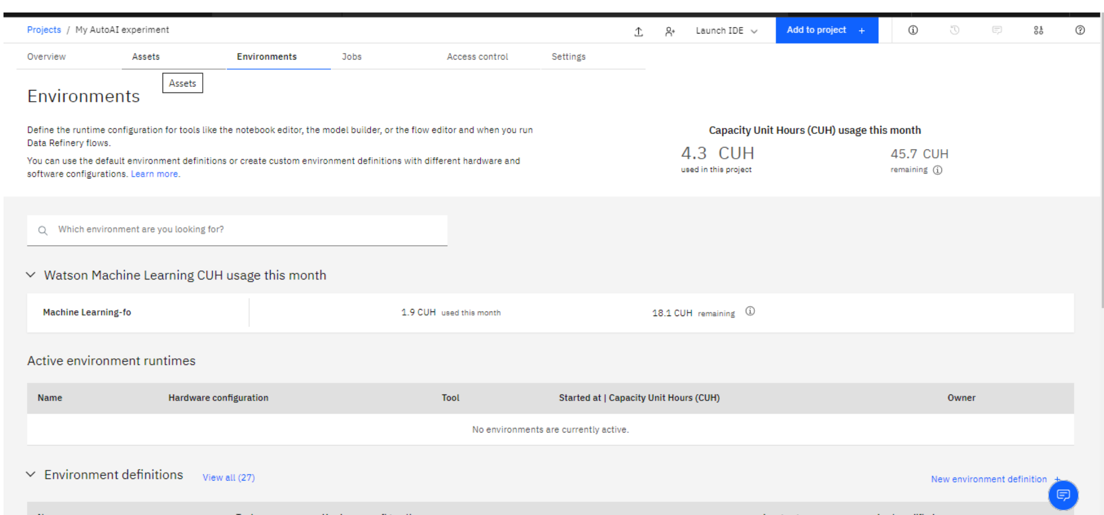
Next, navigate back to the project Assets view by clicking the breadcrumb with your project name (top left of the screen)

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).

Switch to the Environments tab. If there are any active runtimes still running there – please stop them at this point. You can also monitor your per-project and overall CUH spend on this screen. CUH (capacity unit hours) is a resource consumption and utilization metric used by Watson Studio on CPDaaS. Once done, please switch back to the Assets tab.



Deploying your model with Watson Machine Learning (Watson Studio Deploy)

Next, we are going to deploy our saved model.

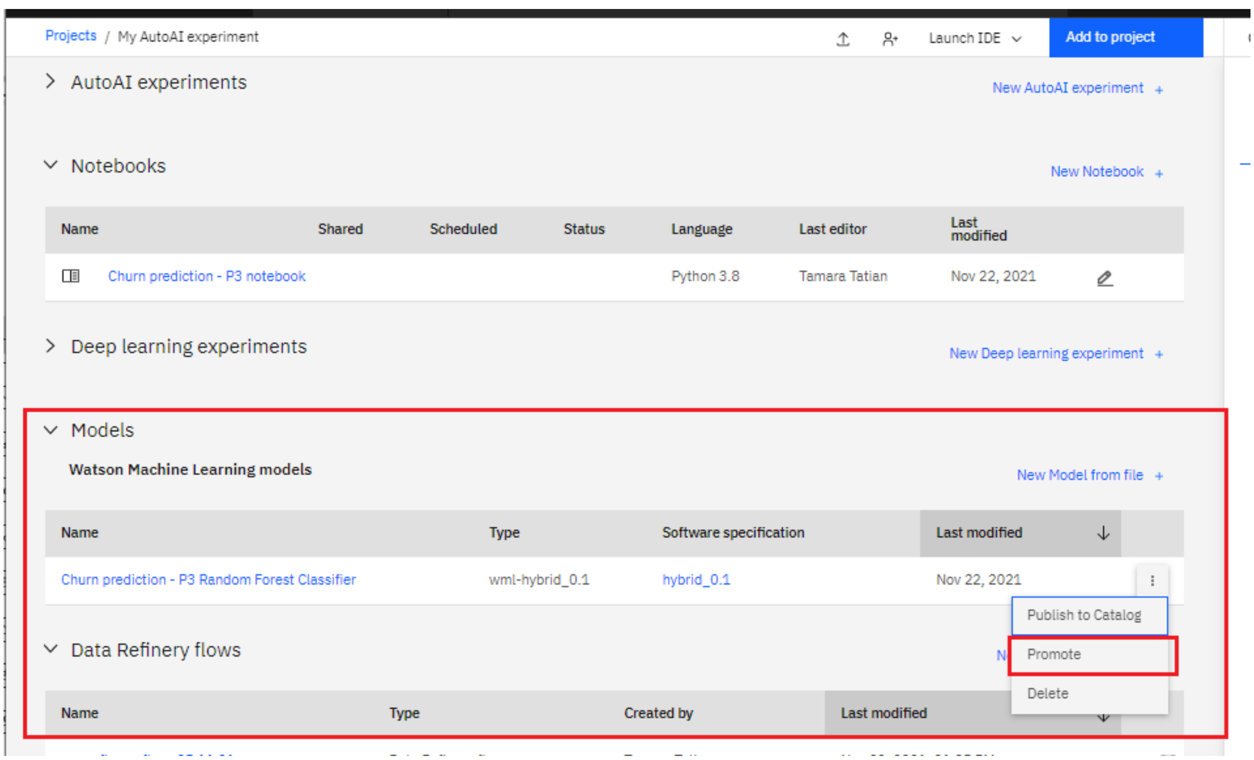
First, we will need to promote the model to a deployment space. Deployment spaces (another project/collaborative workspace type provided in CPDaaS) allow you to create deployments for machine learning models and functions and view and manage all of the activity and assets for the deployments, including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

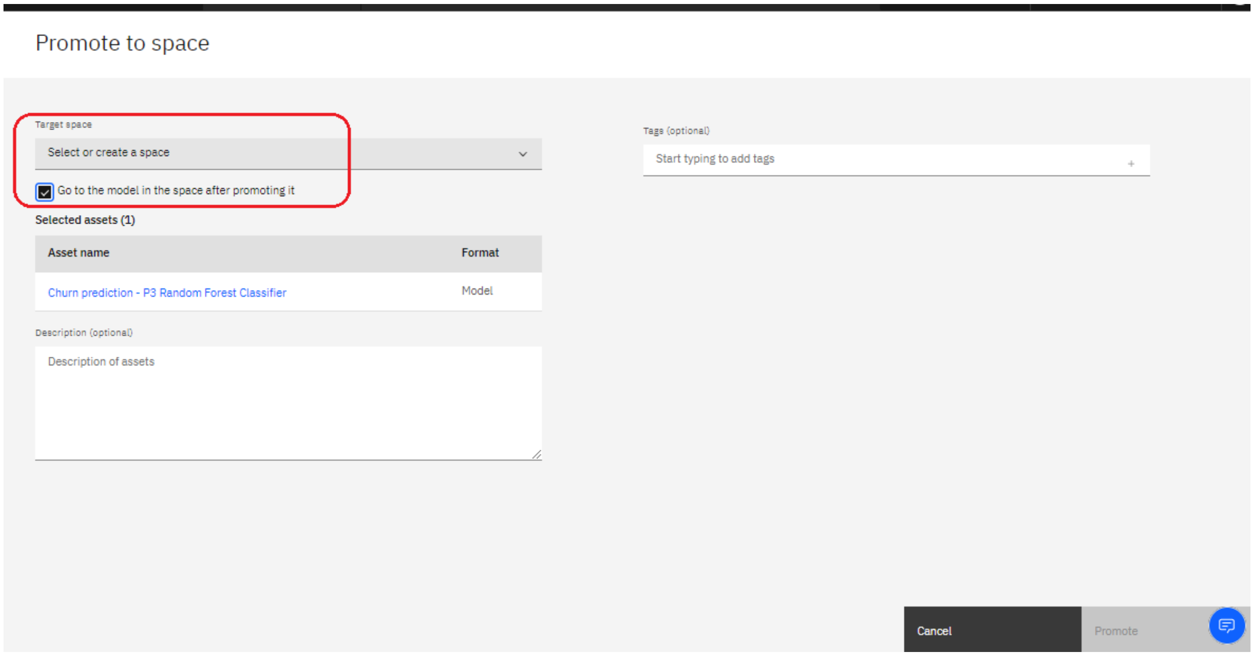
Separate development and deployment environments provide better security and governance. Deployments are often configured by ModelOps team, and only authorized users are given access to *Deployment Spaces*.

In the lab environment you have the necessary permissions to deploy models to a deployment space.

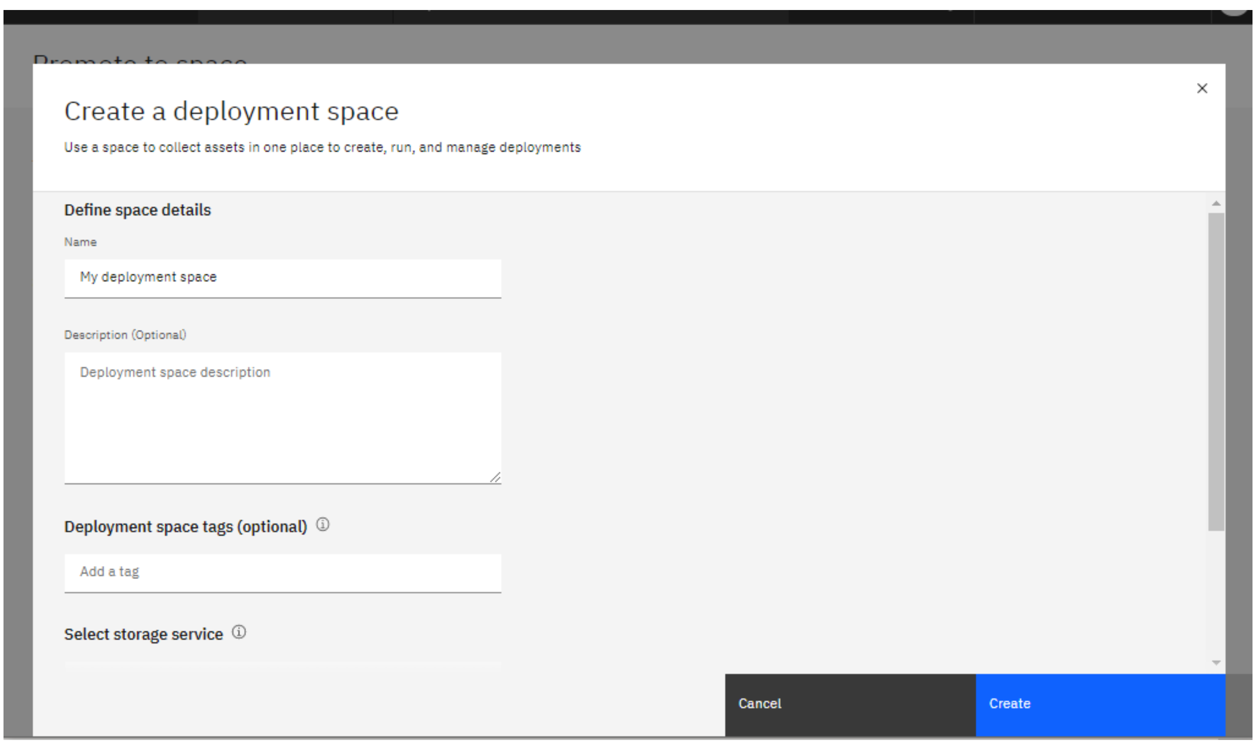
Expand the menu options for your model and select Promote.



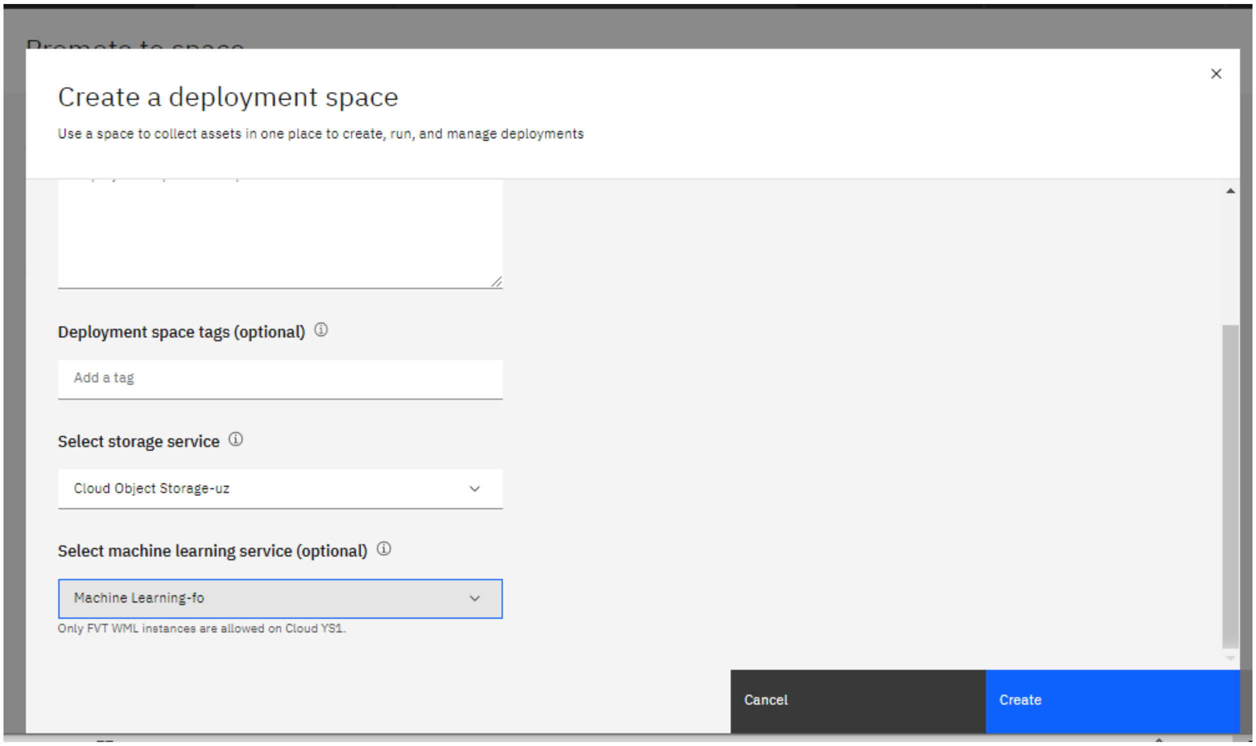
Check the box saying "Go to Model" on the next screen.

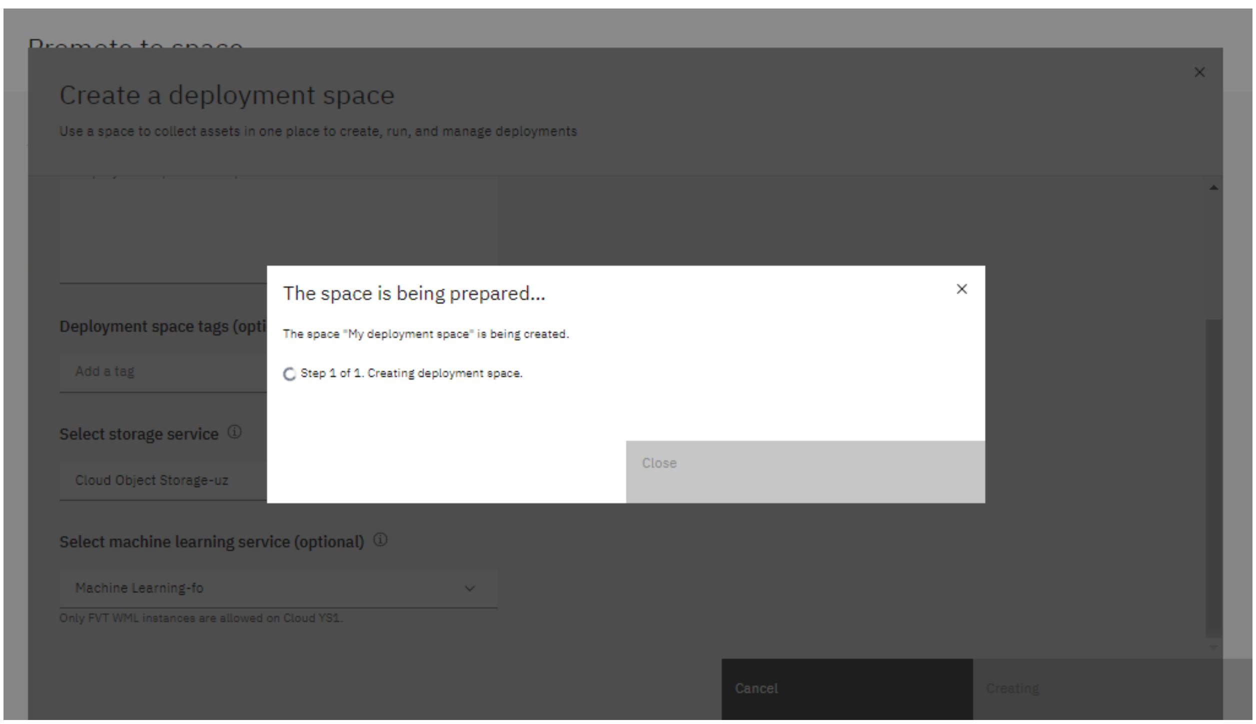


Create a new deployment space.

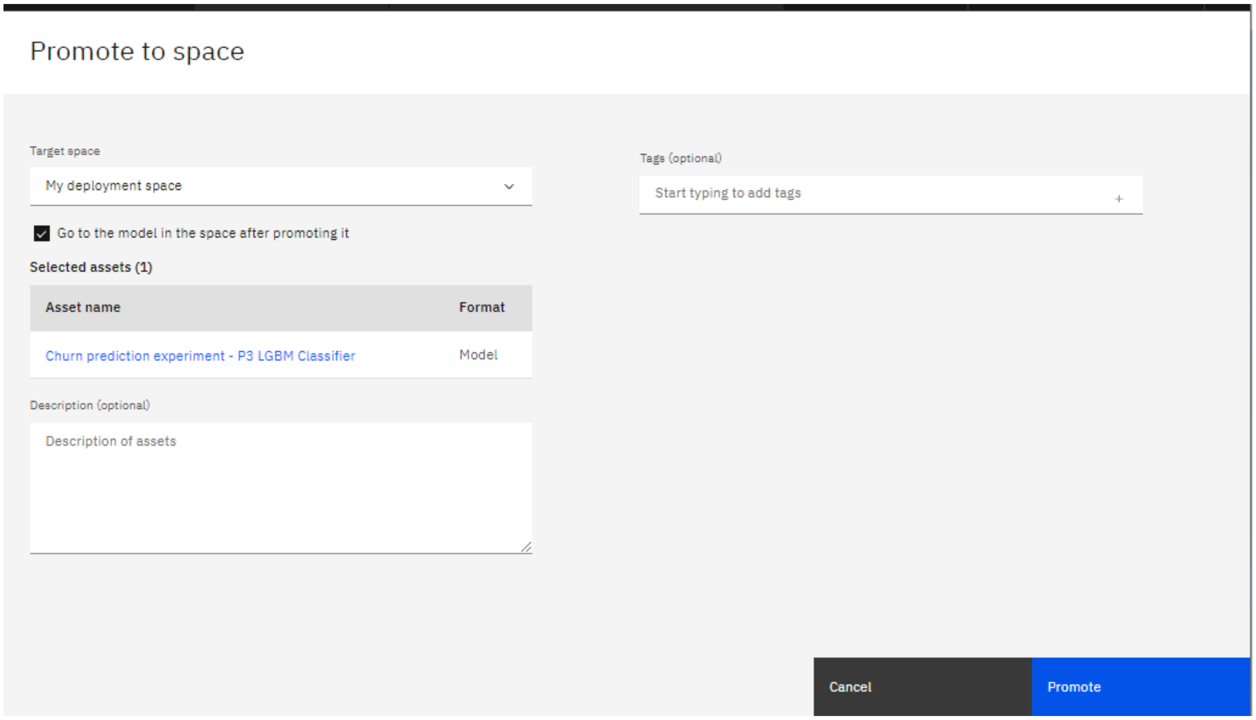


Make sure your Machine Learning instance is associated correctly



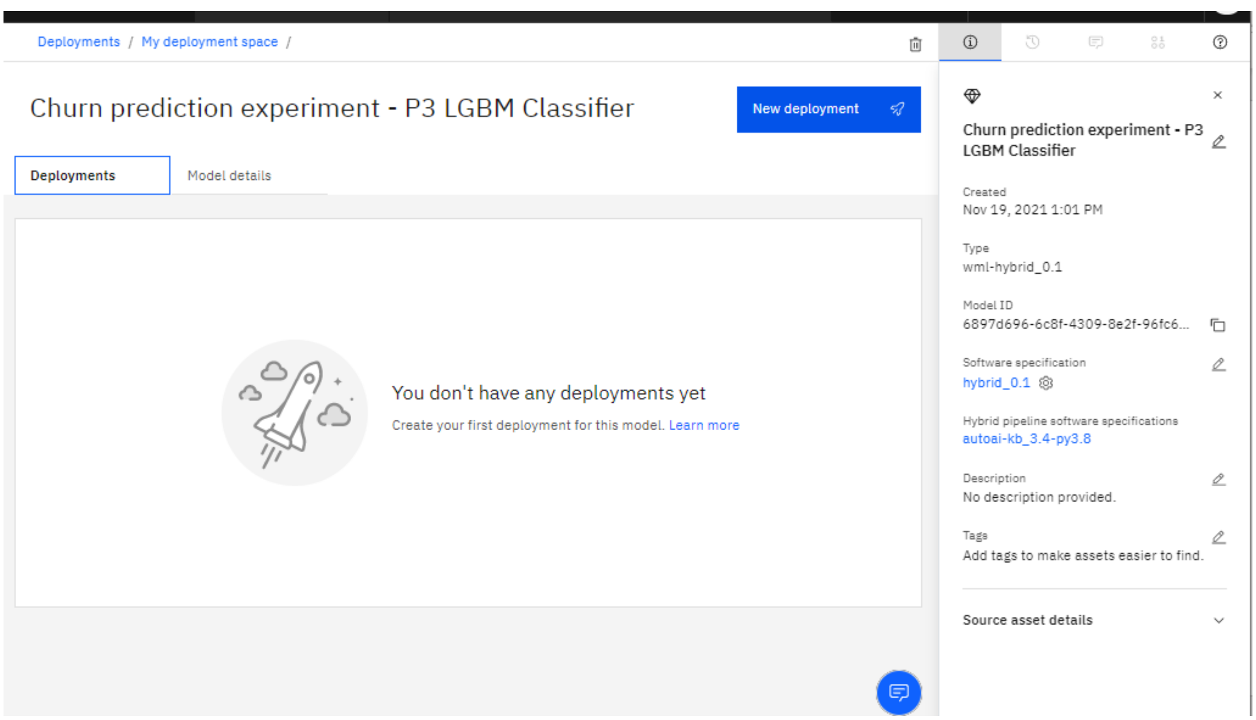


Finally, once the space has been created and picked up, click Promote



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

You can also navigate to your space from the main menu.



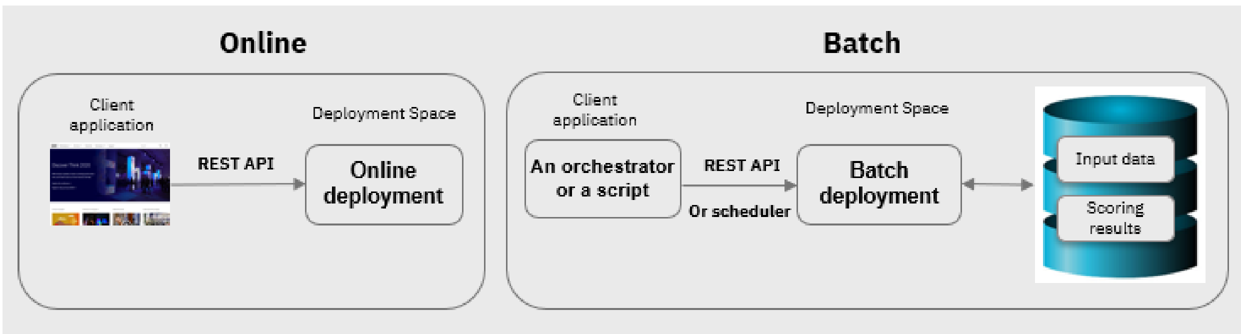
Next, we will deploy the model. Click the New Deployment button.

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

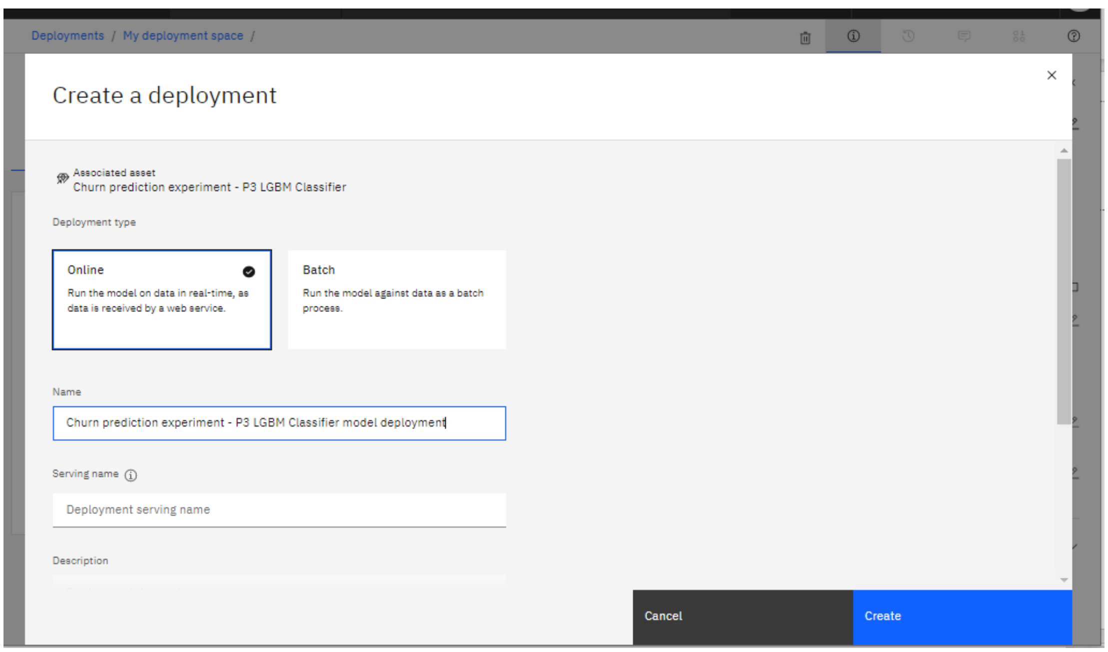
The types of deployments supported by Watson Studio are:

- 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.

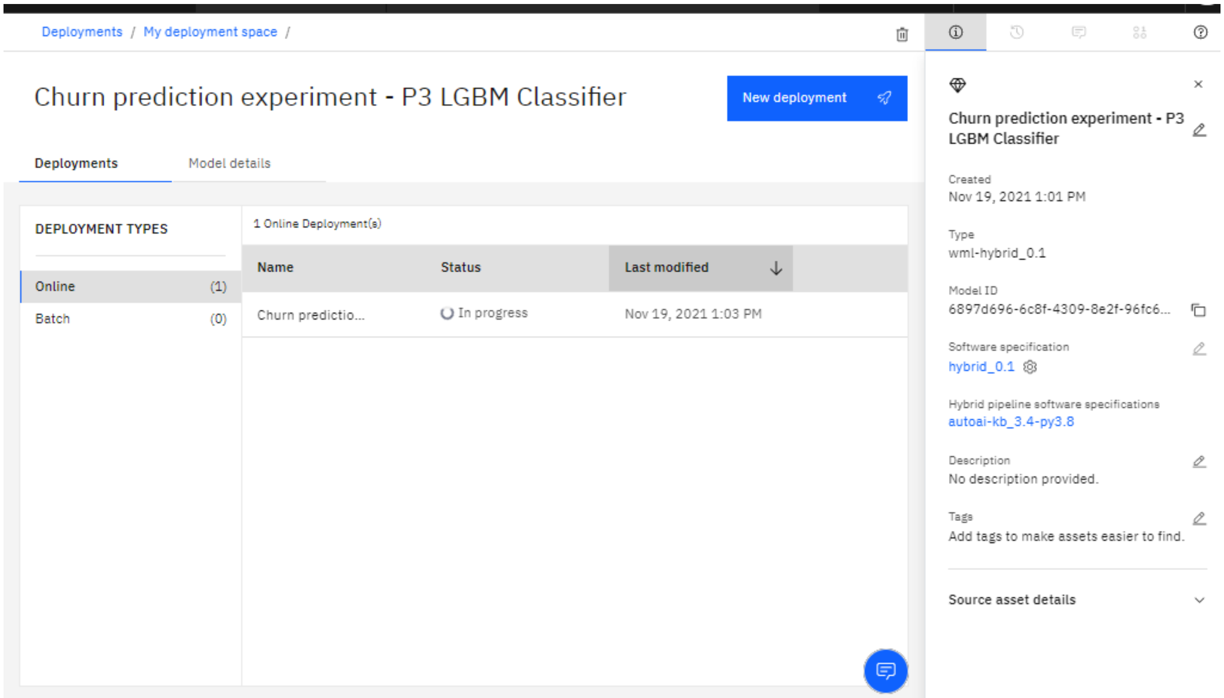
We can deploy our model for online or batch scoring.



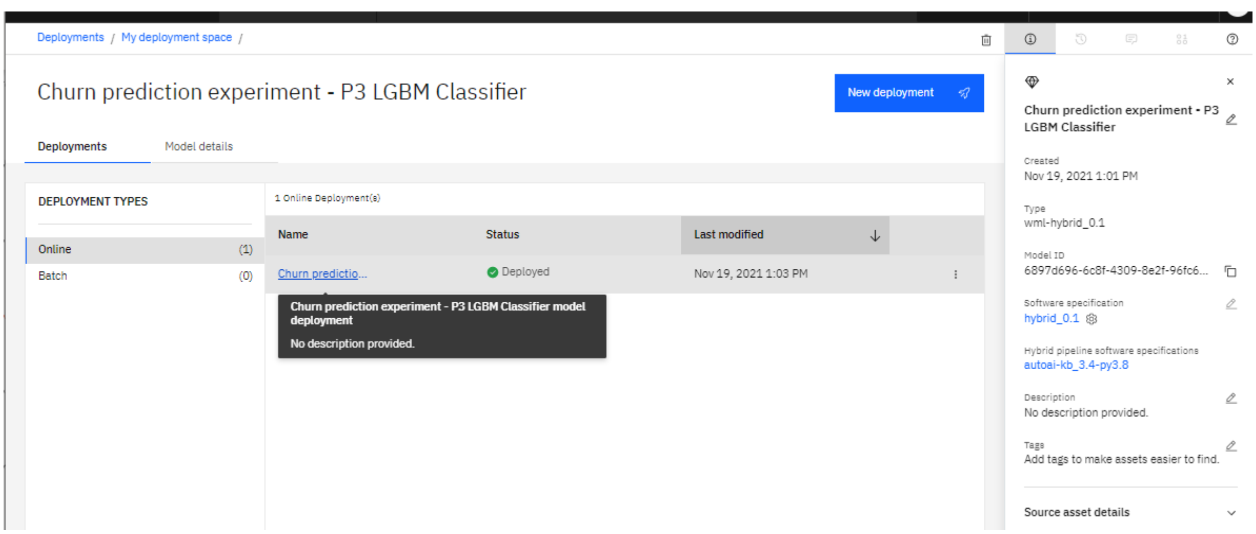
In this lab we will configure online scoring. Give your deployment a name, make sure Online mode is selected, then click Create.



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*.

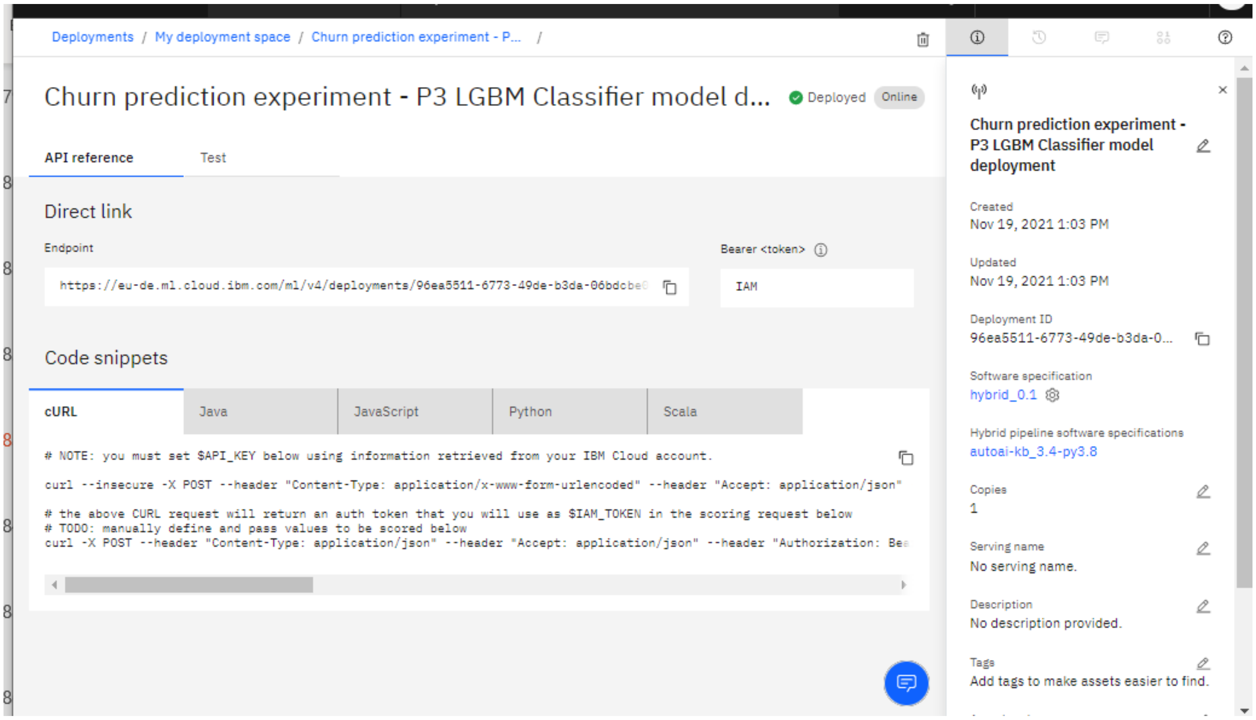


Once the deployment finishes, click the name of the deployed model to see further detail



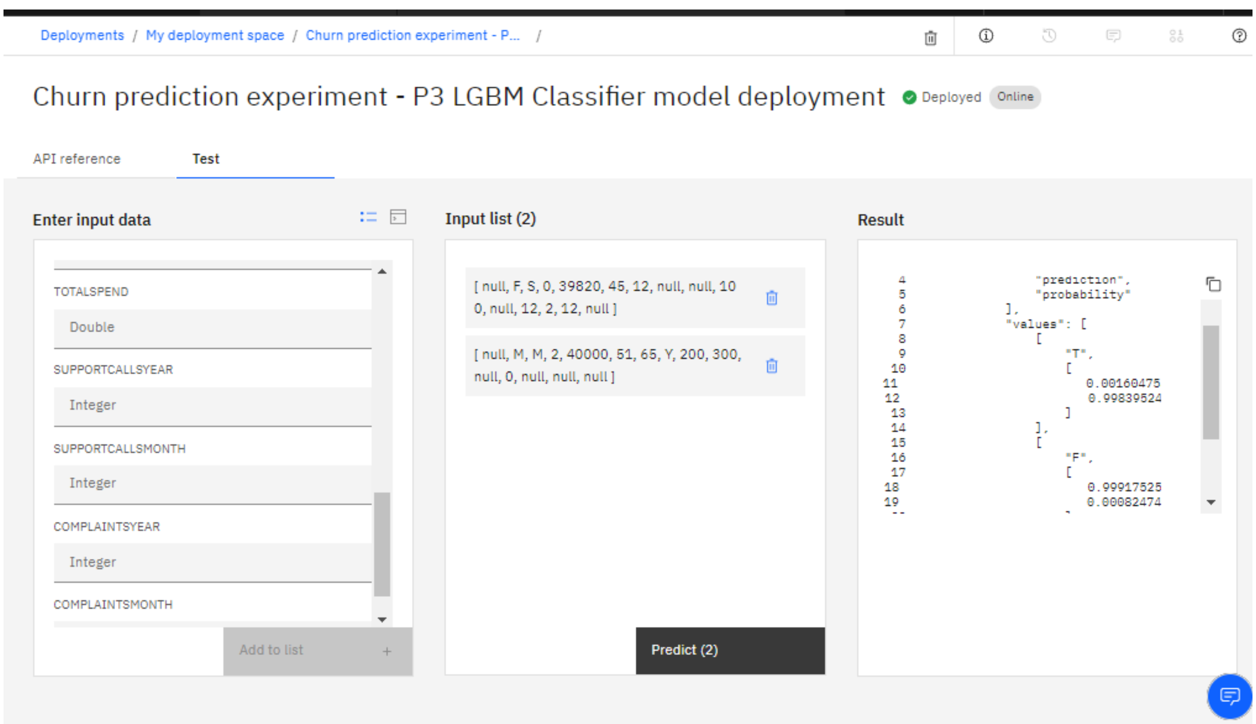
Note that you can scale the deployment and make it even more highly available by increasing the number of its copies (settings on the i tab).

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

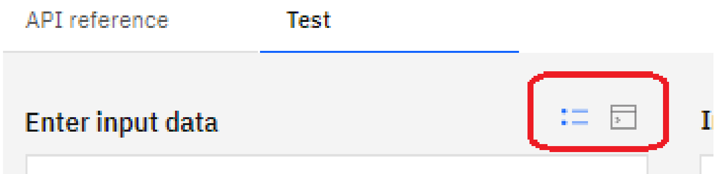


Navigate to the Test tab.

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.



Note that you can also provide test data in JSON format – you can switch between user-friendly fields view and JSON code entry through these buttons

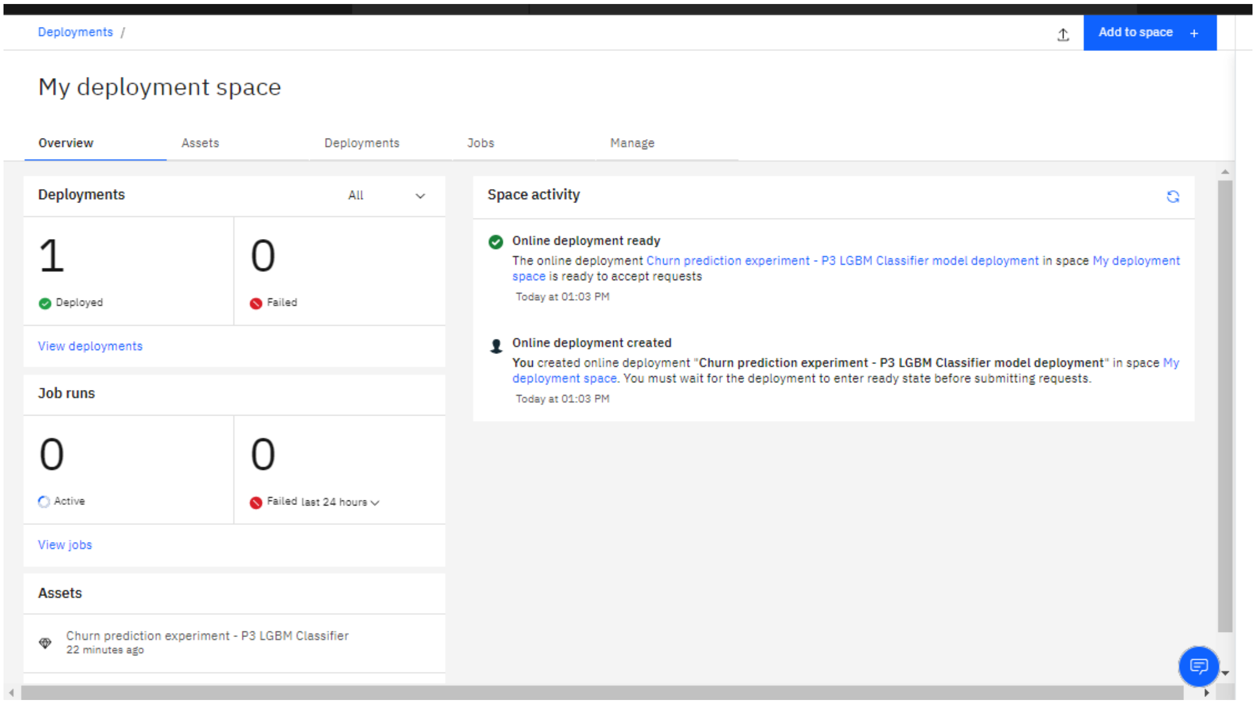


Feel free to test the model using JSON with these sample values:


```
{
        "input_data": [
                {
                        "fields": [
                                "ID",
                                "GENDER",
                                "MARITALSTATUS",
                                "NUMCHILDREN",
                                "ESTINCOME",
                                "AGE",
                                "MONTHSASCUSTOMER",
                                "LOYALTYSCHEME",
                                "LASTBILLAMOUNT",
                                "AVGMONTHLYSPEND",
                                "TOTALSPEND",
                                "SUPPORTCALLSYEAR",
                                "SUPPORTCALLSMONTH",
                                "COMPLAINTSYEAR",
                                "COMPLAINTSMONTH"
                        ],
                        "values": [
                                [
                                        null,
                                        "F",
                                        "S",
                                        null,
                                        39820,
                                        45,
                                        12,
                                        null,
                                        null,
                                        100,
                                        null,
                                        12,
                                        2,
                                        12,
                                        null
                                ],
                                [
                                        null,
                                        "M",
                                        "M",
                                        2,
```

```
                                        40000,
                                        51,
                                        65,
                                        "Y",
                                        200,
                                        300,
                                        null,
                                        null,
                                        null,
                                        null,
                                        null
                                   ]
                              ]
                         }
                    ]
}
```


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.

Optional tasks:

- Run another AutoAI experiment and build a regression model predicting customer TOTALSPEND. Deploy the model in Online mode and test it.
- Create and run an AutoAI experiment based on a Gallery sample