Using built-in Azure ML models (Automated ML)

PURPOSE: To learn how to build and train Azure ML model without using any coding

TIME REQUIRED: 45…60 mins

INTRO: Azure offers several built-in ML models, such as Classification, Regression, Time series, Natural language processing, and Computer vision. In this demo, we will learn how to build and train one of these models. The benefit of such approach is that you do not need to know programming to build these models. The downside is lack of customisation – however, for simple tasks this could be a very good solution

---

Start Azure environment, make sure Azure Machine Learning workspace is created, and launch Azure Machine Learning Studio. For this demo, we do not need Compute – so click **Compute** under the **Manage** section, and make sure there are no Compute instances or clusters created. If there are any – stop and delete them

Place the **diabetes-data.zip** file to the Desktop – either from the resources provided, or download it from https://github.com/MicrosoftLearning/mslearn-azure-ml/raw/refs/heads/main/Labs/02/diabetes-data.zip

Make sure you saved the file to the same environment, which runs the browser you access Azure from (i.e. do not confuse Virtual Machine and your local PC). Extract the archive (right-click – Extract All)

Select **Automated ML** page, under the **Authoring** section. Click **+New Automated ML job**

In the **Basic settings** step, give a unique name to your training job and experiment (or just keep the default values assigned). Click **Next**

In the **Task type & data** step, select **Classification** as the task type. Click **+Create** to add your training data

On the **Create data asset** page, in the **Data type** step, give a name to your data asset (e.g. **data**). Keep Type as **Table (mltable)** and click **Next**

In the **Data source** step, select **From local files** to upload the training data you download previously. Click **Next**

In the **Destination storage type** step, verify that **Azure Blob Storage** is selected as the datastore type, and verify that **workspaceblobstore** is the datastore selected. Click **Next**

In the **MLTable selection** step, click **Upload folder** and browse to the folder you extracted from the file downloaded earlier. Click **Upload**, confirm **Upload**, and wait for two files to upload. Click **Next**

Review the settings and click **Create**

Back in the **Task type & data** step, verify that **Select task type** is still set to **Classification**. Select the data you just uploaded by ticking checkbox next to it and click **Next**

In the **Task settings** step, select **Diabetic (Boolean)** as your **Target column**. Click **View additional configuration settings**

In the **Additional configuration** pane, change the primary metric to **Accuracy** and click **Save**

Expand the Limits section and set the following properties:

- Max trials: **10**
- Experiment timeout (minutes): **60**
- Iteration timeout (minutes): **15**
- Tick **Enable early termination**
- Test data: **Train-test split**
- Verify that the Percentage test of data is set to **10**

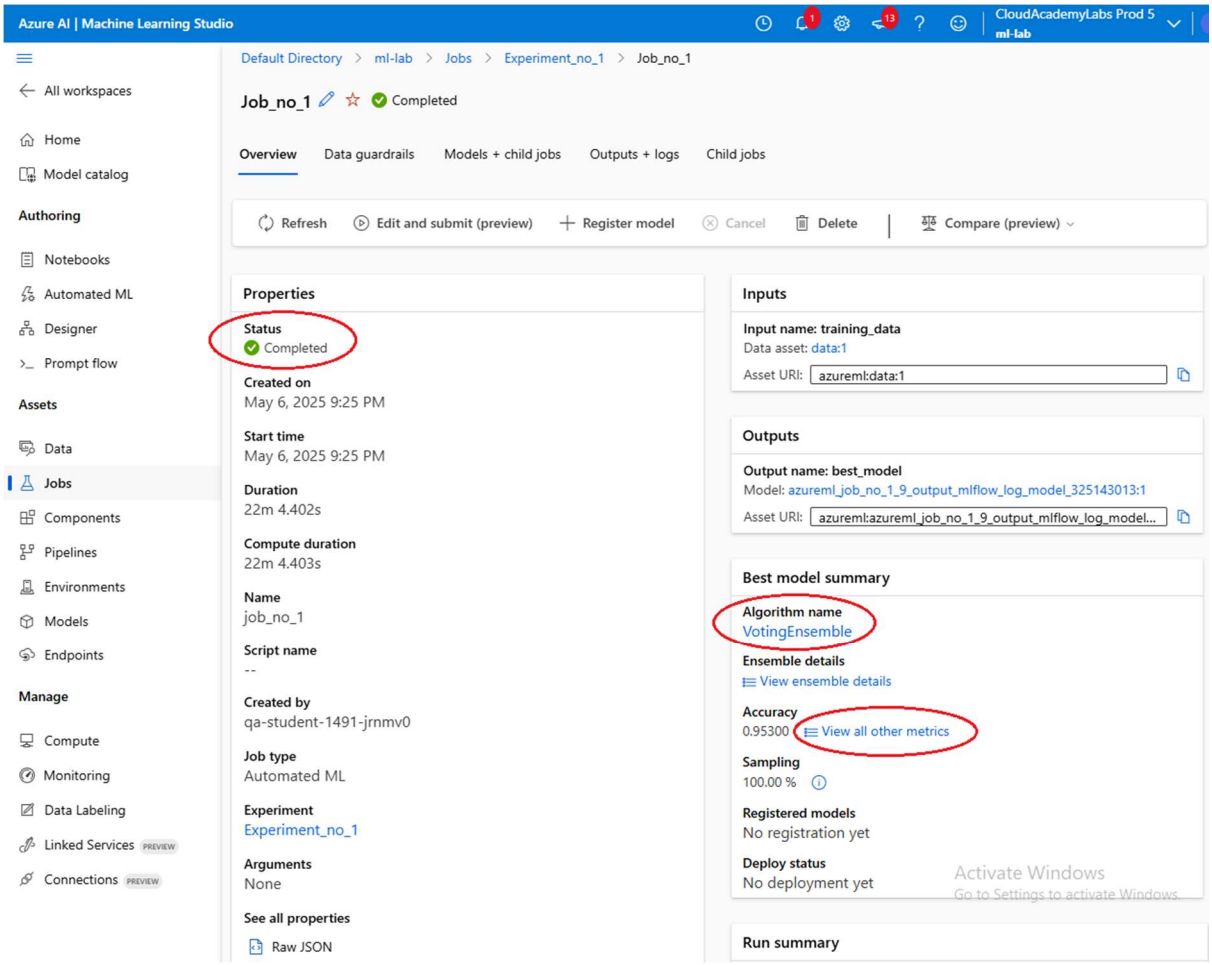
Click **Next**

In the **Compute** step, verify that the compute type is set to **Serverless**, and the virtual machine size is set to **Standard_DS3_v2**. Bear in mind that a larger image may incur higher cost and a smaller image may not be sufficient to complete the tasks. Click **Next**, and then **Submit training job**

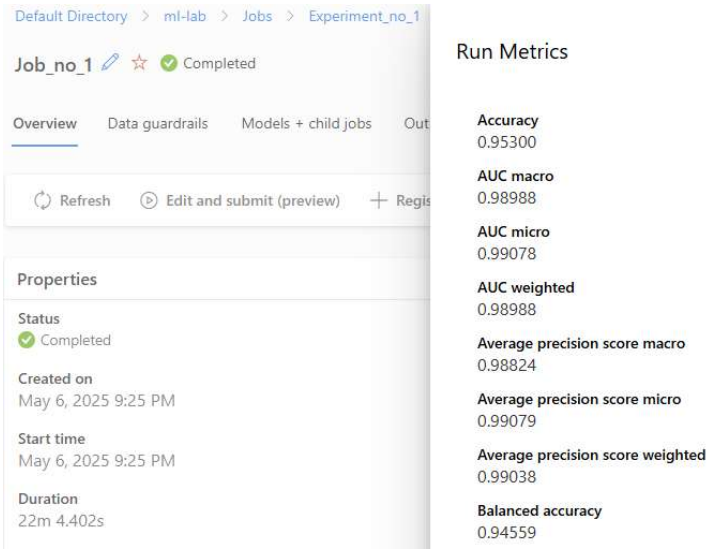
You will be redirected to the **Jobs** page, where you can check its status by clicking **Refresh** button. If you get error message advising that job is not found or has been deleted, simply refresh the page, and then navigate manually by clicking **Jobs** under the **Assets** section, and then click job name

Jobs allow you to keep track of the workloads you ran and compare them with each other. Jobs belong to an experiment, which allows you to group job runs together

Wait for this job to complete – it will take 10…20 minutes. During this time, job status will go through the following stages: Not started; Starting; Setting up the run; Running featurization (misspelled, I know); Model training; and Completed.



Check the results by clicking the following links: **Data guardrails** (shows if any of the steps failed), **Models + child jobs** (lists all the models it has built, sorted by the accuracy), **Output + logs** (shows the code that was used to run the job), and others.



The best model will be displayed in the **Best model summary** section under the **Overview** tab. Under the **Tags** section, label **run_algorithm_000** lists all the other models that were tried. The best model for this particular dataset and task probably was **VotingEnsemble** – you can see all of its metrics by clicking **View all other metrics** under the **Best model summary** section.

As you can see, Azure allows you to create and train Machine Learning model without requiring any code or programming skills.