Hands-on Lab: Custom Model Import

General Info

We can import our custom-built models into DataRobot for deployment. DataRobot streamlines the model deployment process and provides a REST API endpoint and monitoring for each model/deployment. The **Custom Model Workshop** additionally provides testing, governance, and CI/CD integration for our imported models. The following lab provides instructions on how to import a custom model.

Lab Assets

- Access to DataRobot MLOps
- The following assets to be uploaded as part of the custom model:
    - model.pkl - The pickled model artifact
    - custom.py - A python script containing customized code hooks and any additional functions
    - requirements.txt - A requirements text file containing any required package versions
    - A training dataset: “readmissions_train.csv”
- A predictions dataset: “readmissions_test - with Actuals.csv”

Instructions

Import your custom model assets

1. In Classic or NextGen, go to the Model Registry. Navigate to the Custom Model Workshop, and you should automatically be in the Models section.
2. Click “Add new model”
3. For our example model (a binary classification model predicting patient readmissions), configure the following settings:
    a. Model name: [Your choice. E.g. “New Readmission Classifier”]
    b. Target type: Binary
    c. Target name: readmitted
    d. Positive class label: True
    e. Negative class label: False
4. Click “Add custom model”
5. In the Model Environment > Base Environment settings, choose “[DataRobot] Python 3.9 Scikit-Learn Drop-In”. This is a default environment with common libraries required for running sci-kit learn based models.

6. Drag and drop your model files (model.pkl, custom.py, and requirements.txt) into the Model section.
7. In the Dataset section, click “Assign” to upload the training dataset used for this model. Note that you can also use the AI Catalog / Data Registry, which allows you to use a dataset snapshotted from one of your data source connections.
8. Drag and drop the training dataset (“readmissions_train.csv”) into the Add Training Data window.
9. Click “Add training data” once the file is uploaded.
10. Click “Build environment” in the Model Environment section.

Test your custom model

1. Click the “Test Model” button/link.
2. In the Prediction test data, choose file. You can either reupload your training dataset or find the dataset in your AI Catalog / Data Registry if you uploaded there. In general, you only need a few rows to test the model though this lab uses the training dataset itself.
3. We will run the null imputation check and side effects check tests - make sure those two tests are toggled on. (You can review the other tests available in the documentation.)
4. Click “Start test”.

Register your custom model

1. Once testing is complete, click the “Register to deploy” button. will need to navigate You can also navigate back to the “Assemble” tab and click “Register to deploy” there.If you are in NextGen, the “Register model” button will be located in the top right.
2. Register the model into the directory.

Deploy your custom model

Note: The following instructions use DataRobot NextGen:

1. Find and go to your model in the Model Registry > Model Directory (NextGen).
2. After clicking the “Deploy” button, we will set up the monitoring configuration. In the Configure the following:
    a. Under “Show advanced options”:
        i. Data Drift - Enable target monitoring: On by default (because we have the training data assigned to the model)
        ii. Data Drift - Enable feature drift tracking: On by default (because we have the training data assigned to the model)
        iii. Accuracy: Association ID: patient_id
3. Click the “Deploy model” button at the top right to set up the deployment. Wait for DataRobot to finish creating the deployment - note that this will take longer than deploying DataRobot-built models.

Make a new prediction with your deployed custom model

Note: The following instructions use DataRobot NextGen:

1. Access the deployment - you can always get to the deployment again by finding and selecting it from the Deployments tab.
2. In the deployment, go to the Predictions tab. We will use the UI drag-and-drop to test our model in this exercise. You can also access the deployment via the API to make both batch and real-time predictions as well as set up scheduled batch prediction jobs from/to your data sources.
3. In the “Make Predictions” page, drag and drop the prediction dataset (“readmissions_test - with Actuals.csv”) as the Prediction Dataset.
4. Optional: Toggle on Prediction Explanations to return the prediction explanations for each new prediction.
5. Click “Compute and download predictions”.
6. When the predictions are complete, you should be able to download the prediction results. Note that you can also set up scheduled batch prediction jobs that read/write back to your data source or use the API endpoint in your own scoring scripts.

Notes

- For a custom model, the only requirement is a serialized model artifact. As part of the custom model containerization, DataRobot will run default code hooks to load and score the model, so a simple model can be executed without any additional assets.

- The custom.py file allows flexibility for you to include your own functions in addition to modifying the code hooks used by DataRobot. You can also include any number of additional files (e.g. other pickled assets, other .py files, etc) that can be loaded and used in custom.py. For more information, refer to the **custom model components documentation**.

- DataRobot automatically sets up model monitoring for **Structured** custom models. Structured models have an expected output according to the type of model (e.g. regression, classification, etc). You can also import models as **Unstructured** custom models that provide much more flexibility for model input and output.
    - Unstructured custom models can also be monitored by using the DataRobot MLOps library within custom.py. Please ask your DataRobot team for more information to discuss your use case.

- The DataRobot github provides additional templates in the public DataRobot github. The github also includes additional instructions on preparing custom models for both structured and unstructured models.