# AzureML Overview Demo

The demos below will walk through AzureML capabilities across our Hero Scenarios of Setup, Inner Loop, Middle Loop, and Outer Loop using different approaches and tools. This will show you how the field and analysts see the product. There are also advanced exercises for those already familiar with Azure ML basics. Future workshops will go more in depth on different data scientists challenges.

The business question you are being asked to look at as ML Pros is predicting if an airline flight will be delayed or not based up on historical data.

Delays are often considered as an inevitable part of airline travel. Their impact ranges from a minor irritant through to tangible financial loss through missed connections and work opportunities. For a large business, this can result in a material impact to their bottom line as well as having a negative impact on the morale of their employees.

Today, we’ll examine a solution to this challenge by using data to predict the risk of delays and thus provide an organization the opportunity to optimize their travel. We’ll see how the Azure Platform, and in particular Azure Machine Learning, can be used to ingest & prepare data, train & deploy a model, and operationalize an end-to-end solution to this ever-challenging problem.

For the purposes of our analysis, a flight delay will be defined as a flight that arrives more than fifteen minutes late.

## Pre-requisites

Azure subscription or owner of a resource group in a shared subscription so you can create a workspace and the necessary Azure services.

Clone or download this repo to your local machine [Azure-Samples/azureml-flight-delay: Azure Machine Learning demo on flight delay prediction (work in progress) (github.com)](https://github.com/Azure-Samples/azureml-flight-delay).

Ensure sufficient CPU quota in the subscription and region that you will work in.

TODO: Brandon on subscription setup, region recommendation and quota

## Setup Scenario

To run the demo, we first need to create a workspace and configure the required Azure infrastructure services. This is done with an ARM template deployed through the Azure portal. Templates like these are typically created by IT admin teams to provision AML workspaces consistent with organization standards for networking, security, and naming.

The template provisions an AML Workspace, Blob and File Storage, Container Registry, Key Vault, and Application Insights.

|  |  |
| --- | --- |
| **Screenshot** | **Steps** |
| Graphical user interface, text, application, email  Description automatically generated | 1. From the Azure Portal, click on the **Create a resource** button. 2. Search for **Template deployment** and click **Create**. 3. Click on **Build your own template in the editor**. |
| Graphical user interface, text, application, email  Description automatically generated | 1. Click on **Load file**. 2. Locate the ARM Template **aml.json** in the **armTemplates** folder. 3. Click **Save**. |
| Graphical user interface, text, application, email  Description automatically generated | 1. Select your **Subscription** and **Resource Group** and **Region.** 2. Click **Review + create**. 3. Click **Create**. 4. Note the subscription, resource group and region as you will need them later to setup the notebooks.   Use the resource group created for you as a prerequisite, or create a new one if you are a subscription owner. |

Once created, you can open the Workspace you created in [Azure ML Studio](https://ml.azure.com/). You may need to view all workspaces and filter down to the subscription you are using. Hint: look for the resource group you specified in the Azure Portal. The default workspace name is **gartnerdemoltihmrml**.

Compute Instances are restricted to a single user for security. This next step will walk you though creating a new Compute Instance from the Studio. Admins can also provision and configure Compute Instances programmatically for users.

|  |  |
| --- | --- |
| **Screenshot** | **Steps** |
|  | 1. From inside the portal, click **Compute**. |
|  | 1. Within the **Compute Instances** tab. 2. Click on the **+ New** button. |
| Graphical user interface, text, application, email  Description automatically generated | 1. Select a name and size for your new notebook. The notebook name should be unique within the region. A **STANDARD\_DS3\_V2** size is sufficient for this demo. 2. Specify a name (e.g. **aml-notebook-<username>**). 3. Click **Create.** |
|  | 1. The provisioning of the machine usually takes a couple of minutes. The **Status** will remain as **Creating** during the provisioning process. |
|  | 1. Once the Compute Instance is in Running state, select the **Notebooks** tab. 2. Click on the Quick Action to use terminal to cline from git repo 3. In the terminal, enter git clone https://github.com/Azure-Samples/azureml-flight-delay.git 4. Refresh the file viewer in AML Studio and you should see the project cloned to your file share. 5. Close the terminal tab   TODO: git LFS still required if using PowerBI |
| Graphical user interface, text, application  Description automatically generated | 1. Go to the **Notebooks** tab. 2. Navigate to **/azureml-flight-delay/notebooks/setup/setup.ipynb** 3. Update the **workspace** details (these details are available from the Azure Portal):    1. **Subscription ID**    2. **Resource Group**    3. **Workspace Name** 4. Execute the notebook until the end. |

## Forecast Flight Delays with AutoML

The first step for the ML Pro is to understand the business question they are being asked to answer and the data to build a model for prediction or classification. We will create a dataset, profile and view its properties, and use AutoML to train a classification model. We consider these steps part of the Inner Loop scenario using AzureML Studio.

### Preparing data

|  |  |  |
| --- | --- | --- |
| **Screenshot** | **Steps** | **Notes** |
|  | 1. Go to the Datasets tab under assets and create dataset from datastore 2. Name your dataset **flightdelayweather\_ds** | First we will create a dataset from a tabular data file in your workspace storage |
|  | 1. Select the **workspaceworkingdirectory** datastore 2. Browse to the folder for your account and the **azureStorageFiles** directory in the flightdelay folder 3. Select the file flightdelayweather\_ds\_clean.csv and hit **save** |  |
|  | 1. Preview the dataset and schema, accepting the defaults 2. Select Profile this dataset after creation in the final step, using the Compute Instance you created earlier. |  |
|  | 1. Click on the dataset name to view its properties |  |
|  | 1. Select Explore to look at a preview and profile of the data | The profile helps you visualize the data and distribution, identifying need to potentially clean or augment it. |

### Training a forecasting model with AutoML

|  |  |  |
| --- | --- | --- |
| **Screenshot** | **Steps** | **Notes** |
|  | 1. Switch to Automated ML in the Studio 2. Click New Automated ML run |  |
|  | 1. Choose the flightdelayweather\_ds dataset you created above and click “Next” |  |
|  | 1. Create a new Experiment and name it “flight-delay-exp”, select “ArrDelay15” as the Target column, and your Compute Instance as the compute type and target 2. Click Next | AutoML training can be run using a compute cluster or compute instance. |
|  | 1. Select “Classification” 2. Click “View featurization settings”, explore the page then cancel 3. Click “View additional configuration settings”, explore the page then cancel 4. Click “Next” and “Finish” to start the run | Automated ML has correctly detected the features within the dataset and will intelligently transform them before training starts. As a more sophisticated user, you can override this behavior with custom types and imputation approaches. For text inputs, we can even apply deep learning techniques, like BERT, to improve featurization for inputs like text. |
|  | 1. You can monitor progress of the run from the AutoML tab, look at child runs, logs, models created by AutoML, etc 2. Because AutoML is training many models, the overall run can take 45 min to complete | Look at details like AutoML Data guardrails, child runs to see the different models trained, and explainability for an ensemble. |

AutoML will train many models which can take time. Let’s move on and a variation on the flow using Designer.

TODO: Deploy to PowerBI

## Azure ML Designer

Now that we’ve established the potential for ML, other members of the team might want to bring their domain knowledge to build a more sophisticated model. As beginners, they’ll need a visual composition and no-code environment. Let’s see how the Azure ML platform supports us in building a flight delay prediction model.

Designer, the next iteration of Azure ML Studio, retains all of the key visual composition capabilities but leverages the common data (through datasets), experiment, training, deployment (through AKS), and management (through IAM) platform. It’s a great way to get started and to learn.

### Create Pipeline to prepare data and train a model

TODO: Add notes explaining what each step does, or is it self-evident enough?

|  |  |
| --- | --- |
| **Screenshot** | **Steps** |
|  | 1. From inside Studio, click **Designer**. 2. Click the **Easy-to-use prebuilt modules** option. |
|  | 1. In the right panel click the **Select compute target** option. 2. Select the compute instance you created earlier. 3. Click the **Save** button. |
|  | 1. Drag the **flightdelayweather\_ds** dataset from the components panel to the canvas. |
|  | 1. In the components panel, enter **Clean Missing Data** into the search input field. 2. Drag to the canvas the **Clean Missing Data** component. 3. Click the component in the canvas to show the configuration in the right panel. 4. Click the **Edit column** for the **Columns to be cleaned** option. 5. Leave the first option as **Column names**. 6. Enter the value **ArrDelay15**. 7. Click the **Save** button. 8. Connect the two components. |
|  | 1. In the components panel enter **Select Columns in Dataset** into the search input field. 2. Drag to the canvas the component and click it. 3. In the right panel click the **Edit column** for the **Select columns** option. 4. Select the **All columns**. 5. Click the **Save** button. 6. Connect the component with the Clean Missing Data one. |
|  | 1. In the components panel enter **Apply SQL Transformation** into the search input field. 2. Drag to the canvas the component and click it. 3. In the right panel enter the following query to the SQL query script space:   select Month,DayofMonth,DayOfWeek,CRSDepTime,CRSArrTime,UniqueCarrier,CRSElapsedTime,Origin,Dest,Distance,ArrDelay15,Origin\_Lat,Origin\_Lon,Origin\_State,Dest\_Lat,Dest\_Lon,Dest\_State,CAST(Origin\_dayl AS float) AS Origin\_dayl,CAST(Dest\_dayl AS float) AS Dest\_dayl,CAST(Origin\_prcp AS float) AS Origin\_prcp,CAST(Dest\_prcp AS float) AS Dest\_prcp,CAST(Origin\_srad AS float) AS Origin\_srad,CAST(Dest\_srad AS float) AS Dest\_srad,CAST(Origin\_swe AS float) AS Origin\_swe,CAST(Dest\_swe AS float) AS Dest\_swe,CAST(Origin\_tmax AS float) AS Origin\_tmax,CAST(Dest\_tmax AS float) AS Dest\_tmax,CAST(Origin\_tmin AS float) AS Origin\_tmin,CAST(Dest\_tmin AS float) AS Dest\_tmin,CAST(Origin\_vp AS float) AS Origin\_vp,CAST(Dest\_vp AS float) AS Dest\_vp  from t1  where Dest\_dayl != '--' and Dest\_prcp != '--' and Dest\_srad != '--' and Dest\_swe != '--' and Dest\_tmax != '--' and Dest\_tmin != '--' and Dest\_vp != '--'   1. Connect the component with the previous one. |
|  | 1. In the components panel enter **Summarize Data** into the search input field. 2. Drag to the canvas. 3. Connect with the previous component.   Optional: Submit the run to see the summarized data. Create a new experiment name like “Flight-Delay-Designer”. Click on “View run overview” to see progress. This takes about 6 min to complete. Right-click on the component to preview the results dataset. |
|  | 1. In the components panel enter **Filter Based Feature Selection** to the search input field. 2. Drag to the canvas and click it. 3. In the right panel for the **Target column** option click the **Edit column**. 4. Leave **Column names** option selected and select the **ArrDelay15** column. 5. Click the **Save** button. 6. Connect the component with the **Apply SQL Transformation** component. |
|  | 1. In the components panel enter **Two-Class Logistic Regression in**to the search input field. 2. Drag to the canvas. 3. Do not connect this component with previous components. |
|  | 1. In the components panel enter **Split Data** into the search input field. 2. Drag to the canvas. 3. Connect the **Filter Based Feature Selection** component with this one. |
|  | 1. In the components panel enter **Two-Class Averaged Perceptron** into the search input field. 2. Drag to the canvas. 3. Do not connect this component with previous components. |
|  | 1. In the components panel enter **Train Model** into the search input field. 2. Drag to the canvas and click it. 3. In the right panel for the **Label column** option click the **Edit column**. 4. Leave **Column names** option selected and select the **ArrDelay15** column. 5. Click the **Save** button. 6. Connect the component with the previous **Two-Class Logistic Regression**. 7. Also connect the component with the previous **Split Data** component. |
|  | 1. In the components panel enter **Train Model** into the search input field. 2. Drag to the canvas and click it. 3. In the right panel for the **Label column** option click the **Edit column**. 4. Leave **Column names** option selected and select the **ArrDelay15** column. 5. Click the **Save** button. 6. Connect the component with the previous **Two-Class Averaged Perceptron**. 7. Also connect the component with the previous **Split Data** component. |
|  | 1. In the components panel enter **Score Model** to the search input field. 2. Drag to the canvas. 3. Connect the component with the previous **Train Model** component that is connected with the **Two-Class Logistic Regression**. 4. Also connect the component with the previous **Split Data** component. |
|  | 1. In the components panel enter **Score Model** to the search input field. 2. Drag to the canvas. 3. Connect the component with the previous **Train Model** component that is connected with the **Two-Class Averaged Perceptron**. 4. Also connect the component with the previous **Split Data** component. |
|  | 1. In the components panel enter **Evaluate Model** to the search input field. 2. Drag to the canvas. 3. Connect the component with the 2 previous **Score Model** components. |
|  | 1. The final design should look something like this. |
|  | 1. Click “Submit” button to run the designer experiment. |
|  | 1. Select “Create new” 2. Enter an experiment name. 3. Click the “Submit” button. The whole pipeline can take around 15 min to run. 4. When done, right-click on “Evaluate Model” and preview the evaluation results for the two models you trained |

### Evaluate the model

TODO: Add commentary/notes on the components?

Add data to test model with

Managed endpoint instead of ACI, or not available yet?

Advanced: export to code and use with CLI

|  |  |
| --- | --- |
| **Screenshot** | **Steps** |
|  | 1. Once the designer experiment has finished. 2. Click on one of the “Train Model” 3. Click the “Create inference pipeline” dropdown at the top. 4. Select “Real-time inference pipeline”. |
|  | 1. Remove the “Evaluate Model” item from the bottom of the Designer. 2. Click the “Submit” button. This takes around 7 min. |
|  | 1. Once the designer experiment has finished. 2. Click on one of the “Deploy”. |
|  | 1. Fill in the endpoint name. 2. Select a Compute Type: ACI 3. Click “Deploy”. You can then view the new endpoint on the “Endpoints” tab and test the scoring. The model can be seen on the “Models” tab. |

## Notebook with AutoML Responsible AI

TODO

Remove PowerBI

Troubleshoot AutoML errors—show best model type,

Add/update Responsible AI (is this best shown here or next notebook with scikit-learn?)

Update for managed endpoints

Next we will follow the workflow using a Jupyter Notebook and Python scripts. We will extend the demo to include Responsible AI analysis.

|  |  |
| --- | --- |
| **Screenshot** | **Steps** |
| Graphical user interface, text, application, email  Description automatically generated | In AzureML Studio, click on Notebooks  Click “flight-delay.ipynb” under “notebooks/flight-delay-automl”.  Use Python 3.8 kernel |
| Graphical user interface, text, application, email  Description automatically generated | Run the cells under:   * Install prerequisites * Setup working directory * Write helper file |
| Graphical user interface, text, application  Description automatically generated | Scroll through to “Define the training scripts” to see example of using the AzureML SDK  Run the cells in this section |
| Graphical user interface, text, application, email  Description automatically generated | Scroll through to the Getting Started section  Run the cell to import the AzureML SDK |
| Graphical user interface, text, application, email  Description automatically generated | Skip the PowerBI cells for now |
| Graphical user interface, text, application  Description automatically generated | Scroll down to Connect to Workspace  Edit the cell with your subscription id, resource group and workspace name (these can be found in the Azure Portal). Then run the cell |
| Graphical user interface, text, application  Description automatically generated | Next load the dataset and then create a CPU compute cluster (or connect if you already created it) |
| Graphical user interface, text, application, email  Description automatically generated | Scroll down to Instantiate an Automated ML Config and run the next cells to start an experiment on the compute cluster.  Confirm in the output that the remote run was submitted. |
| Graphical user interface, text, application, email  Description automatically generated | Run the cell for Display Automated ML Run Details to monitor the results.  You can also monitor this in the AzureML Studio Experiments view. The run takes about 7 minutes to complete. |
| Graphical user interface  Description automatically generated with low confidence | When AutoML job is complete, run the cell Show best run |
| Graphical user interface, text, application, email  Description automatically generated  A picture containing application  Description automatically generated | Skip Show best run and Show best model type as they are not working right now |
| Graphical user interface, text, application, email  Description automatically generated | This next section shows you the data and feature engineering capabilities in Automated ML  Class balancing detection will look for imbalanced training data |
| Graphical user interface, text, application  Description automatically generated | Feature engineering has a pandas error |
| Graphical user interface, text, application  Description automatically generated | Engineered features importance has an error |
| Graphical user interface, text, application, email  Description automatically generated | This explains the importance of the raw features in training the classification model |
| Graphical user interface, text  Description automatically generated | This next section shows Responsible ML capabilities  And may need to be updated  First is InterpretML    Does not work as no model pkl file from best run |
| Graphical user interface, text, application, email  Description automatically generated | Need to configure Dashboard |
|  | Need to complete this |
|  | Stop at ML Pipelines section |

## MLOps and MLFlow

TODO: Remove AKS and ACI, update MIR and run through rest of steps

|  |  |
| --- | --- |
| **Screenshot** | **Steps** |
|  | Click on Notebook “flight-delay-mlflow.ipynb” in \flight-delay-mlops\flight-delay-mlflow  Run the first 4 cells to set things up |
|  | The next cell write the training script to a working directory |
|  | The next cell configures your workspace connection. You need the subscription ID, resource group, and workspace name from the Portal.  Edit the cell and then run it. |
|  | Next step is to load the training dataset.  The output shows you a preview of the data |
|  | And then configure a CPU compute cluster if not already created |
|  | Next create a baseline for Data Drift Monitoring and create a Data Drift Monitor.  This can take over 20 min to complete. |
|  | Then compare the dataset to target for a specificized time period |
|  | Now let’s train and register a model using scikit-learn  Dataset name should be flightdelayweathger\_ds not \_flightdelayweather\_ds\_clean  Start the training run and fetch the latest model |
|  | Next let’s deploy the model as a realtime endpoint with MLflow Tracking  Change to MIR instead of AKS and ACI?  Error in AKS deploy |
|  | Change to MIR instead of AKS and ACI?  Error in ACI Deploy—error in entry script |
|  | Need to use correct AKS service name from above in the cell |
|  | Managed endpoint deploy  Create directory  Create scoring file  Create environment definition  Define endpoint configuration—update this for errors in next cell |
|  | Deploy the endpoint  Edit the cell with workspace and resource group name  Change command to batch-endpoint  Update the config file |
|  | Need to validate and capture the remaining steps… |
|  |  |

## Local Visual Studio Code to Cloud

Propose removing this demo of VS Code locally then job in cloud