Machine Learning Lab Assignment

#### (Provide your code snippets. You may answer the questions with screenshots if applicable.)

In this lab, we're going to have a little competition between ourselves and the automated machine learning system. You’ll be creating a machine learning model from scratch to classify patients and predict whether they have diabetes or not. At the same time, you’ll also have Azure Machine Learning’s AutoML system create a similar model. The goal is to see if you can beat the AutoML system!

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PatientID** | **Pregnancies** | **Plasma**  **Glucose** | **DiastolicBlood**  **Pressure** | **Triceps**  **Thickness** | **Serum**  **Insulin** | **BMI** | **Diabetes**  **Pedigree** | **Age** | **Diabetic** |
| 1354778 | 0 | 171 | 80 | 34 | 23 | 43.509 | 1.21319135 | 21 | 0 |
| 1147438 | 8 | 92 | 93 | 47 | 36 | 21.240 | 0.15836498 | 23 | 0 |
| 1640031 | 7 | 115 | 47 | 52 | 35 | 41.511 | 0.07901857 | 23 | 1 |

At the end, you’ll deploy the best model as an API and I will score a validation dataset through your model and see who can make the best overall model (either manually or using AutoML).

### A couple notes:

* Each person should complete the ML model training and submit this lab individually using the share course Azure Machine Learning workspace. (Feel free to work together on the dataset upload step and help each other out, of course.)
* Learning to use Azure Machine Learning will prepare you for the [Microsoft DP-100](https://docs.microsoft.com/en-us/learn/certifications/exams/dp-100) exam. Thus, you can reference the following training material (though these are NOT the lab instructions): <https://microsoftlearning.github.io/mslearn-dp100/>
* Though Azure Machine Learning has an R SDK, it’s currently being rewritten. So, I would suggest you use the Python SDK. You can use the workspace UI for the AutoML modeling, but please use the SDK for your manual model creation.
* I don’t provide all the necessary code to complete this lab, but here’s an excellent repo for sample code: <https://github.com/MicrosoftLearning/mslearn-dp100>

## Create a Datastore and Data Asset

1. Download the diabetes dataset from [here](https://github.com/colbyford/DSBA6190-CloudComputing/raw/master/3%20-%20Machine%20Learning/lab/diabetes_train.csv) and upload the CSV file to your data lake.
2. Using the workspace UI, create a Datastore connection from Azure Machine Learning to your data lake. (Only needs to be done once per group
   * Datastore name: *dsba6190storage<group>*
   * Datastore type: *Azure Blob Storage*

Graphical user interface, text, application, email

Description automatically generated

* + Account key: (From the Access Keys menu in your Azure Storage Account)

Graphical user interface, text, application, email

Description automatically generated

1. Using the workspace UI, create a Data Asset of the diabetes training data (from the Datastore you just created).
   * Basic info
     1. Name: *diabetes\_<group>*
   * Datastore selection
     1. Select or create a datastore: *dsba6190storage<group>*
     2. Path: *diabetes\_train.csv* (or wherever you uploaded the file)
   * Settings and preview
     1. (Leave the default settings unless you see an error.)
   * Schema
     1. Check the column types to make sure the automatic schema determination worked correctly.

## Create a Compute Instance

1. Create a Compute Instance for yourself using the following settings:
   * + Name: *<group>-<name>*
     + Region: *East US*
     + Virtual machine type: *CPU*
     + Virtual machine size: *Standard\_D4s\_v3*
     + Don’t worry about Advanced Settings but have a look at the options anyway.
2. Once the instance has finished spinning up, open JupyterLab to begin coding.

## Train Your Own Diabetes Model

1. From JupyterLab inside your Compute Instance, be sure to navigate into your user folder.

Graphical user interface, application

Description automatically generated

1. Create a new “Python 3.8 – Azure ML” notebook in JupterLab
2. Train the best classification machine learning model you can using any standard ML library (such as Scikit-learn or XGBoost) to predict the “Diabetic” column in the training dataset.
   * Name your Experiment: <group>-<name>-<description>
   * Log any metrics associated with your model training to the Experiment.
3. Use the Azure Machine Learning functionality to track the model performance.

**Sample Code:** <https://github.com/MicrosoftLearning/mslearn-dp100/blob/main/05%20-%20Train%20Models.ipynb>

## Have AutoML Train a Diabetes Model

1. Create another “Python 3.8 – Azure ML” notebook in JupterLab (or you can use the AutoML UI)
2. Have the AutoML system train a classification model on the “Diabetic” column in the training dataset.

**Sample Code:** <https://github.com/MicrosoftLearning/mslearn-dp100/blob/main/12%20-%20Use%20Automated%20Machine%20Learning.ipynb>

## Interpret the Best AutoML Model

1. Once AutoML has finished its process of training multiple models, use the Model Explainability tool to explain the interpret the model and tell you which features were the most important in the diabetes classification. (This can either be performed in the UI or as a setting in the SDK code.)

## Deploy the Best Model as a Webservice

1. After comparing the performance of your best model versus the best model from the AutoML system, deploy the model as a webservice endpoint.

* If your winning model is your own, here’s some sample code: <https://github.com/MicrosoftLearning/mslearn-dp100/blob/main/09%20-%20Create%20a%20Real-time%20Inferencing%20Service.ipynb>
* If your winning model is an AutoML model, you can deploy from the UI quite easily.

## Questions

1. Provide a screenshot and compare the performance of your best AutoML model(s) vs. the model(s) you trained yourself.
2. Which model did you select as your best model? Why?
3. For the AutoML training, which primary metric did you choose? Why?
4. For the best AutoML model, which features did it select as the most important? (Either list the features or provide a screenshot.)
5. Provide the endpoint URL and key for your deployed model.