Implementing Predictive Analytics with Spark in Azure HDInsight

Lab 3 – Evaluating Supervised Learning Models

Overview

In this lab, you will use Spark to evaluate classification and regression models. You will then validate parameters to optimize the performance of your models.

What You'll Need

To complete the labs, you will need the following:

- A web browser
- A Microsoft account
- A Microsoft Azure subscription
- A Windows, Linux, or Mac OS X computer
- Azure Storage Explorer
- The lab files for this course
- A Spark 2.0 HDInsight cluster.

Note: If you have not already done so, set up the required environment for the lab by following the instructions in the Setup document for this course. Then follow the instructions in Lab 1 to provision an HDInsight cluster.

Evaluating a Classification Model

First, you will evaluate a classification model that predicts whether or not a flight will be late.

Upload Source Data to Azure Storage

Note: If you have already uploaded the flights.csv data file to your Azure storage container, you can skip this procedure.

In this lab, you will build a model based on data about flights. Before you can do this, you must store the flight data files in the shared storage used by your cluster. The instructions here assume you will use Azure Storage Explorer to do this, but you can use any Azure Storage tool you prefer.

1. In the folder where you extracted the lab files for this course on your local computer, in the **data** folder, verify that the **flights.csv** file exists. This file contains flight data that has been cleaned and prepared for modeling.
2. Start Azure Storage Explorer, and if you are not already signed in, sign into your Azure subscription.
3. Expand your storage account and the **Blob Containers** folder, and then double-click the blob container for your HDInsight cluster.
4. In the **Upload** drop-down list, click **Upload Files**. Then upload **flights.csv** as a block blob to a folder named **data** in root of the container.

Upload a Jupyter Notebook

You will use a Jupyter Notebook to create and evaluate your classification model. You can choose to work with Python or Scala.

1. In the Azure portal, in the blade for your HDInsight cluster, under **Quick Links**, click **Cluster Dashboards**.
2. Click **Jupyter Notebook**, and if prompted, log in using the cluster login name you specified when provisioning your cluster (be sure you use login name for HTTP connections and not the SSH user name.)
3. Click **Upload**, and browse to the **Lab03** folder in the folder where you extracted the lab files. Then select either **Python Classification Evaluation.ipynb** or **Scala Classification Evaluation.ipynb**, depending on your preferred choice of language, and upload it.

Evaluate a Classification Model

Now that you have uploaded the notebook, you can use the code it contains to evaluate your model.

1. Open the notebook you uploaded and then read the notes and run the code it contains to build and evaluate a classification model.

Evaluating a Regression Model

Having evaluated a classification model that predicts whether or not a flight will be late, you will now evaluate a regression model that predicts how late (or early) flights will arrive.

Evaluate a Regression Model

You will use a Jupyter Notebook to create and evaluate your regression model. You can choose to work with Python or Scala.

1. From the **Lab03** folder in the folder where you extracted the lab files, upload **Python Regression Evaluation.ipynb** or **Scala Regression Evaluation.ipynb**, depending on your preferred choice of language, to the Jupiter Dashboard for your cluster.
2. Open the notebook you uploaded and then read the notes and run the code it contains to build a regression model.

Tuning Parameters

You can optimize the performance of a model by tuning the parameters that you specify when training it. In this exercise, you will explore two common techniques for tuning parameters.

Tune Parameters using a Training / Validation Split

You will use a Jupyter Notebook to tune the parameters for your model. You can choose to work with Python or Scala.

1. From the **Lab03** folder in the folder where you extracted the lab files, upload **Python Parameter Tuning.ipynb** or **Scala Parameter Tuning.ipynb**, depending on your preferred choice of language, to the Jupiter Dashboard for your cluster.
2. Open the notebook you uploaded and then read the notes and run the code it contains to build a classification model and use a **TrainValidationSplit** class to tune the parameters.

Tune Parameters using Cross-Validation

You will use a Jupyter Notebook to create your regression model. You can choose to work with Python or Scala.

1. From the **Lab03** folder in the folder where you extracted the lab files, upload **Python Cross Validation.ipynb** or **Scala Cross Validation.ipynb**, depending on your preferred choice of language, to the Jupiter Dashboard for your cluster.
2. Open the notebook you uploaded and then read the notes and run the code it contains to build a classification model and use a **CrossValidation** class to tune the parameters.

Clean Up

If you intend to proceed straight to the next lab, skip this section. Otherwise, follow the steps below to delete your cluster and avoid being charged for cluster resources when you are not using them.

Delete the Resource Group

1. Close the browser tab containing the Jupyter Notebooks dashboard if it is open.
2. In the Azure portal, view your **Resource groups** and select the resource group you created for your cluster. This resource group contains your cluster and the associated storage account.
3. In the blade for your resource group, click **Delete**. When prompted to confirm the deletion, enter the resource group name and click **Delete**.
4. Wait for a notification that your resource group has been deleted.
5. Close the browser.