Microsoft

Data Science and Machine Learning Essentials

Module 2 Key Points

Chapter 9: Data Acquisition and Flow

Key Points

- An Azure ML experiment is usually published as a Web service, to which real-time data can be passed as input and results are returned as output. In addition to the real-time data passed to the web service, the experiment may reference static data.
- Real-time data is passed to a published experiment through the **Web Service Input** module. Results are passed to the client that called the experiment web service through the **Web Service Output** module.
- You can use the following Azure ML modules to read and write static data:
- **Dataset** (read data from sample data or manually uploaded data files)
- **Enter Data** (enter data values directly into the experiment)
- **Reader** (read data from Azure storage, Azure SQL Database, HTTP URLs, data feeds, or Hive queries)
- **Writer** (write data to Azure storage, Azure SQL Database, or HTTP URLs)
- You can use the **Join** module to combine two datasets based on common key fields.
- You can use the **Add Rows** module to concatenate (or *union*) two datasets with similar schema.

Further Reading

- Import data: https://azure.microsoft.com/en-gb/documentation/articles/machine-learning-importdata/
- Data Input and Output: https://msdn.microsoft.com/en-us/library/azure/dn906024.aspx
- The **Join** module: https://msdn.microsoft.com/en-us/library/azure/dn905836.aspx
- The **Add Rows** module: https://msdn.microsoft.com/en-us/library/azure/dn905871.aspx

Chapter 10: R and Python for Data Science

Key Points

- You can extend Azure ML experiments by using custom R or Python code. Use the **Execute R Script** module to run R code, or the **Execute Python Script** module to run Python code.
- The **Execute R Script** module requires an R script that uses the **maml.mapInputPort** function to read data frames passed to it.

- The **Execute Python Script** module requires a function named **azureml_main** that accepts a data frame as a parameter.
- To test R code locally, install R and R Studio. To test Python code locally, install a Python editor such as Spyder.
- When testing code, add conditional logic to load data from a local file when running locally, or the data frame passed to the module when running in Azure ML.
- You can encapsulate R or Python functions that return data frames as a dataset by uploading zipped scripts.

Further Reading

You can use the following resources as references when working with R or Python:

Resources for R

- Quick R: http://statmethods.net
- CRAN R Manuals: https://cran.r-project.org/manuals.html
- Extend Your Experiment with R: https://azure.microsoft.com/enus/documentation/articles/machine-learning-extend-your-experiment-with-r/

Resources for Python

- Beginner's Guide to Python: https://wiki.python.org/moin/BeginnersGuide
- Python 2.7 Documentation: https://docs.python.org/2.7/
- Execute Python Scripts: https://azure.microsoft.com/en-us/documentation/articles/machinelearning-execute-python-scripts/

Chapter 11: Data Sampling and Quantization

Key Points

- Data types for variables (column values for features and labels) include integer and floating point numbers, Boolean values, strings, date/time values, timespans, and images.
- Some values can be *continuous* (a value on a numeric scale) or *categorical* (a discrete value that represents a category).
- Metadata describes attributes of the data. Examples include column names, data types, and whether a numeric field is categorical. You can modify metadata in an Azure ML experiment by using the **Metadata Editor** module.
- You can convert, or *quantize*, continuous numeric variables into categorical variables by using the **Quantize Data** module. This allocates each value into one of a specified number of categories, or *bins*.
- You can use custom R or Python code to quantize variables.

Further Reading

- **Metadata Editor** module documentation: https://msdn.microsoft.com/enus/library/azure/dn905986.aspx
- **Quantize Data** module documentation: https://msdn.microsoft.com/enus/library/azure/dn913065.aspx

Chapter 12: Data Cleansing and Transformation

Key Points

- You can use the **Clean Missing Data** module to handle missing values in your data.
- You can use the **Remove Duplicate Rows** module to delete duplicate rows in your data.
- You can use the **Clip Values** module to handle outliers in your data.
- You can use the **Normalize Data** module to scale numeric values in your data.
- You can use R or Python to implement custom logic that cleans your data. This is particularly useful when you need to visualize and handle outliers in datasets that include many features.

Further Reading

- **Clean Missing Data** module documentation: https://msdn.microsoft.com/enus/library/azure/dn906028.aspx
- **Remove Duplicate Rows** module documentation: https://msdn.microsoft.com/enus/library/azure/dn905805.aspx
- **Clip Values** module documentation: https://msdn.microsoft.com/enus/library/azure/dn905918.aspx
- **Normalize Data** module documentation: https://msdn.microsoft.com/enus/library/azure/dn905838.aspx