**Microsoft Azure Machine Learning Studio**

**Data Science**

Before you can start learning how to use data science techniques and build machine learning solutions, it might be useful to start by defining what we mean by "data science". In this course, we use the following definition:

*“ Data Science is the exploration and quantitative analysis of all available structured and unstructured data to develop understanding, extract knowledge, and formulate actionable results.”*

**Data Science Tools and Technologies**

Now that you know something about what data science is, and the iterative process that data scientists follow; it's time to consider the technologies that you can use to explore data.

There's a huge range of tools and technologists that you can use as a data scientist; ranging from spreadsheet tools like Microsoft Excel with through to large-scale data processing platforms like Apache Spark. This course focuses on a core set of tools that you can use to perform most data exploration, cleaning, and modeling tasks - and which you will use in the course labs. These tools are:

* Microsoft Azure Machine Learning
* R
* Python

**Data Science Essentials**

Data scientists use a range of technologies to work with data. From common productivity tools like Excel to highly specialized statistical tools and languages. Microsoft Azure Machine Learning Studio (Azure ML Studio) is a cloud-based service from Microsoft in which you can create and run data science experiments, and publish them as web services. It provides an easy to use, drag and drop interface for creating data science experiments, and can significantly reduce the time it takes to build a solution. Azure ML Studio experiments are based on data flows that include datasets, which define sources of data, and modules that operate on the data. In addition to tools like Azure ML Studio, most data scientists rely heavily on programming languages that enable them to ingest, manipulate, aggregate, and visualize data. Most data scientists have some familiarity with Structured Query Language (SQL), which is a standards-based language for querying databases and retrieving data. Additionally, most data scientists are proficient in R or Python (or both!)

**Machine Learning Studio (classic): Algorithm and module**

Machine Learning Studio (classic) is a cloud predictive analytics service that makes it possible to quickly create and deploy predictive models as analytics solutions. The machine learning tools are mostly cloud-based services, eliminating setup and installation concerns because you can work through your web browser on any internet-connected PC.

**Module:**

Each module in Machine Learning Studio (classic) represents a code set that can run independently and perform a machine learning task, given the required inputs. A module might contain a particular algorithm, or perform a task that is important in machine learning, such as missing value replacement, or statistical analysis.

In Studio (classic), modules are organized by functionality:

* ***Data input and output modules*** do the work of moving data from cloud sources into your experiment. You can write your results or intermediate data to Azure Storage, a SQL database, or Hive, while running an experiment, or use cloud storage to exchange data between experiments.
* ***Data transformation modules*** support operations on data that are unique to machine learning, such as normalizing or binning data, feature selection, and dimensionality reduction.
* ***Machine learning algorithms***, such as clustering, support vector machine, or neural networks, are available within individual modules that let you customize the machine learning task with appropriate parameters. For classification tasks, you can choose from binary or multiclass algorithms. After you've configured the model, use a training module to run data through the algorithm, and measure the accuracy of the trained model by using one of the evaluation modules. To get predictions from the model you've just trained, use one of the scoring modules.
* ***Anomaly detection:*** Machine Learning Studio (classic) includes multiple algorithms specialized for these tasks.
* ***Text analytics*** modules support various natural language processing tasks.
* ***Python and R language*** modules make it easy to run a custom function. You write the code, and embed it in a module, to integrate Python and R with an experiment service.
* ***OpenCV*** library provides modules to use in specific image recognition tasks.
* ***Time series analysis*** supports anomaly detection in time series.
* ***Statistical*** modules provide a wide variety of numerical methods related to data science. Look in this group for correlation methods, data summaries, and statistical and math operations.

**A-Z list of Machine Learning Studio (classic) modules:**

The modules cover a wide range of features and functions necessary for machine-learning tasks:

* Data conversion functions
* Data transformation functions
* Modules for executing R or Python script
* Algorithms, including:
* Decision trees
* Decision forests
* Clustering
* Time series
* Recommendation models
* Anomaly detection

**To find a module:**

* If you know the name of the module, use the alphabetical table as an index to quickly find a specific module or algorithm.
* For a list of the modules by functional category, see Module categories and descriptions.

**Alphabetical table of modules:**

|  |  |
| --- | --- |
| **Module name** | **Description** |
| * *Add Columns* | Adds a set of columns from one dataset to another. |
| * *Add Rows* | Appends a set of rows from an input dataset to the end of another dataset. |
| * *Apply Filter* | Applies a filter to specified columns of a dataset. |
| * *Apply Math Operation* | Applies a mathematical operation to column values. |
| * *Apply SQL Transformation* | Runs a SQLite query on input datasets to transform the data. |
| * *Apply Transformation* | Applies a well-specified data transformation to a dataset. |
| * *Assign Data to Clusters* | Assigns data to clusters by using an existing trained clustering model. |
| * *Bayesian Linear Regression* | Creates a Bayesian linear regression model. |
| * *Boosted Decision Tree Regression* | Creates a regression model by using the boosted decision tree algorithm. |
| * *Build Counting Transform* | Creates counts to use to build features. |
| * *Clean Missing Data* | Specifies how to handle values that are missing from a dataset. |
| * *Clip Values* | Detects outliers, and then clips or replaces their values. |
| * *Compute Elementary Statistics* | Calculates specified summary statistics for selected dataset columns. |
| * *Detect Languages* | Detects the language of each line in the input file. |
| * *Compute Linear Correlation* | Calculates the linear correlation between column values in a dataset. |
| * *Convert to ARFF* | Converts data input to the attribute relation file format that's used by the Weka toolset. |
| * *Convert to CSV* | Converts data input to a comma-separated values format. |
| * *Convert to Dataset* | Converts data input to the internal dataset format that's used by Machine Learning. |
| * *Convert to Indicator Values* | Converts categorical values in columns to indicator values. |
| * *Convert to SVMLight* | Converts data input to the format that's used by the SVMlight framework. |
| * *Convert to TSV* | Converts data input to the tab-delimited format. |
| * *Create R Model* | Creates an R model by using custom resources. |
| * *Cross-Validate Model* | Cross-validates parameter estimates for classification or regression models by partitioning the data. |
| * *Decision Forest Regression* | Creates a regression model by using the decision forest algorithm. |
| * *Detect Languages* | Detects the language of each line in the input file. |
| * *Edit Metadata* | Edits metadata that's associated with columns in a dataset. |
| * *Enter Data Manually* | Enables entering and editing small datasets by typing values. |
| * *Evaluate Model* | Evaluates a scored classification or regression model by using standard metrics. |
| * *Evaluate Probability Function* | Fits a specified probability distribution function to a dataset. |
| * *Evaluate Recommender* | Evaluates the accuracy of recommender model predictions. |
| * *Execute Python Script* | Executes a Python script from an Machine Learning experiment. |
| * *Execute R Script* | Executes an R script from an Machine Learning experiment. |
| * *Export Count Table* | Exports counts from a count transform. |
| * *Export Data* | Writes a dataset to web URLs or to various forms of cloud-based storage in Azure, such as tables, blobs, and Azure SQL databases. This module was formerly named Writer. |
| * *Extract Key Phrases from Text* | Extracts keywords and phrases from a text column. |
| * *Extract N-Gram Features from Text* | Creates N-Gram dictionary features, and then does feature selection on them. |
| * *Fast Forest Quantile Regression* | Creates a quantile regression model. |
| * *Feature Hashing* | Converts text data to integer-encoded features by using the Vowpal Wabbit library. |
| * *Filter Based Feature Selection* | Identifies the features in a dataset that have the greatest predictive power. |
| * *FIR Filter* | Creates a finite impulse response filter for signal processing. |
| * *Fisher Linear Discriminant Analysis* | Identifies the linear combination of feature variables that can best group data into separate classes. |
| * *Group Categorical Values* | Groups data from multiple categories into a new category. |
| * *Group Data into Bins* | Puts numerical data into bins. |
| * *IIR Filter* | Creates an infinite impulse response filter for signal processing. |
| * *Import Count Table* | Imports counts from an existing count table. |
| * *Import Data* | Loads data from external sources on the web or from various forms of cloud-based storage in Azure, such as tables, blobs, SQL databases, and Azure Cosmos DB. Can load data from an on-premises SQL Server database if a gateway has been configured. This module was formerly named Reader. |
| * *Import Images* | Loads images from Azure Blob storage into a dataset. |
| * *Join Data* | Joins two datasets. |
| * *K-Means Clustering* | Configures and initializes a K-means clustering model. |
| * *Latent Dirichlet Allocation* | Performs topic modeling by using the Vowpal Wabbit library for Latent Dirichlet Allocation (LDA). |
| * *Linear Regression* | Creates a linear regression model. |
| * *Load Trained Model* | Gets a trained model that you can use for scoring in an experiment. |
| * *Median Filter* | Creates a median filter that's used to smooth data for trend analysis. |
| * *Merge Count Transform* | Merges two sets of count tables. |
| * *Modify Count Table Parameters* | Builds a compact set of count-based features from count tables. |
| * *Moving Average Filter* | Creates a moving average filter that smooths data for trend analysis. |
| * *Multiclass Decision Forest* | Creates a multiclass classification model by using the decision forest algorithm. |
| * *Multiclass Decision Jungle* | Creates a multiclass classification model by using the decision jungle algorithm. |
| * *Multiclass Logistic Regression* | Creates a multiclass logistic regression classification model. |
| * *Multiclass Neural Network* | Creates a multiclass classification model by using a neural network algorithm. |
| * *Named Entity Recognition* | Recognizes named entities in a text column. |
| * *Neural Network Regression* | Creates a regression model by using a neural network algorithm. |
| * *Normalize Data* | Rescales numeric data to constrain dataset values to a standard range. |
| * *One-Class Support Vector Machine* | Creates a one-class support vector machine model for anomaly detection. |
| * *One-vs-All Multiclass* | Creates a multiclass classification model from an ensemble of binary classification models. |
| * *Ordinal Regression* | Creates an ordinal regression model. |
| * *Partition and Sample* | Creates multiple partitions of a dataset based on sampling. |
| * *Permutation Feature Importance* | Computes the permutation feature importance scores of feature variables in a trained model and a test dataset. |
| * *PCA-Based Anomaly Detection* | Creates an anomaly detection model by using Principal Component Analysis (PCA). |
| * *Poisson Regression* | Creates a regression model that assumes data has a Poisson distribution. |
| * *Preprocess Text* | Performs cleaning operations on text. |
| * *Pretrained Cascade Image Classification* | Creates a pretrained image classification model for frontal faces by using the OpenCV Library. |
| * *Principal Component Analysis* | Computes a set of features that have reduced dimensionality for more efficient learning. |
| * *Remove Duplicate Rows* | Removes duplicate rows from a dataset. |
| * *Replace Discrete Values* | Replaces discrete values from one column with numeric values based on another column. |
| * *Score Matchbox Recommender* | Scores predictions for a dataset by using the Matchbox recommender. |
| * *Score Model* | Scores predictions for a trained classification or regression model. |
| * *Score Vowpal Wabbit 7-4 Model* | Scores data by using the Vowpal Wabbit machinelearningsystem. Requires a trained model built by using Vowpal Wabbit versions 7-4 and 7-6. |
| * *Score Vowpal Wabbit 7-10 Model* | Scores data by using the Vowpal Wabbit machine learning system.  Requires a trained model built by using Vowpal Wabbit version 7-10. |
| * *Score Vowpal Wabbit 8 Model* | Scores data by using the Vowpal Wabbit machine learning system from the command-lineinterface. Requires a trained model built by using Vowpal Wabbit version 8. |
| * *Select Columns in Dataset* | Selects columns to include in or exclude from a dataset in an operation. |
| * *SMOTE* | Increases the number of low-incidence examples in a dataset by using synthetic minority oversampling. |
| * *Split Data* | Partitions the rows of a dataset into two distinct sets. |
| * *Summarize Data* | Generates a basic descriptive statistics report for the columns in a dataset. |
| * *Sweep Clustering* | Performs a parameter sweep on a clustering model to determine the optimum parameter settings. |
| * *Test Hypothesis Using T-Test* | Compares means from two datasets by using a t-test. |
| * *Threshold Filter* | Creates a threshold filter that constrains values. |
| * *Time Series Anomaly Detection* | Learns a trend in time series data, and then uses the trend to detect anomalies. |
| * *Train Anomaly Detection Model* | Trains an anomaly detector model, and then labels data from the training set. |
| * *Train Clustering Model* | Trains a clustering model, and then assigns data from the training set to clusters. |
| * *Train Matchbox Recommender* | Trains a Bayesian recommender by using the Matchbox algorithm. |
| * *Train Model* | Trains a classification or regression model in a supervised manner. |
| * *Train Vowpal Wabbit 7-4 Model* | Trains a model from the Vowpal Wabbit machine learning system.  This module is for compatibility with Vowpal Wabbit versions 7-4 and 7-6. |
| * *Train Vowpal Wabbit 7-10 Model* | Trains a model from the Vowpal Wabbit machine learning system.  This module is for Vowpal Wabbit version 7-10. |
| * *Train Vowpal Wabbit 8 Model* | Trains a model by using version 8 of the Vowpal Wabbit machine learning system.  This module is for Vowpal Wabbit version 8. |
| * *Tune Model Hyperparameters* | Performs a parameter sweep on a regression or classification model to determine the optimum parameter settings. |
| * *Two-Class Averaged Perceptron* | Creates an averaged perceptron binary classification model. |
| * *Two-Class Bayes Point Machine* | Creates a Bayes point machine binary classification model. |
| * *Two-Class Boosted Decision Tree* | Creates a binary classifier by using a boosted decision tree algorithm. |
| * *Two-Class Decision Forest* | Creates a two-class classification model by using the decision forest algorithm. |
| * *Two-Class Decision Jungle* | Creates a two-class classification model by using the decision jungle algorithm. |
| * *Two-Class Locally Deep Support Vector Machine* | Creates a binary classification model by using the locally deep support vector machine algorithm. |
| * *Two-Class Logistic Regression* | Creates a two-class logistic regression model. |
| * *Two-Class Neural Network* | Creates a binary classifier by using a neural network algorithm. |
| * *Two-Class Support Vector Machine* | Creates a binary classification model by using the support vector machine algorithm. |
| * *Unpack Zipped Datasets* | Unpacks datasets from a .zip package in user storage. |
| * *User-Defined Filter* | Creates a custom finite or infinite impulse response filter. |