

Iris Classification Problem

COMP 4211 - Tutorial 02

Chun-Kit Yeung

Hong Kong University of Science and Technology

2018-02-23

In this tutorial, I am going to cover the general work-flow in machine learning project. Azure Machine Learning Studio is used to let you have a first taste using machine learning algorithm to solve a classification problem.

If time is available, I will introduce *Numpy* which is a core scientific library in Python. It provides high-efficiency in computation of array (or so-called tensor) object.

Create an experiment

Create a model

- 1 Get data
- 2 Prepare the data (Not required this time.)
- 3 Define the columns used

Train the model

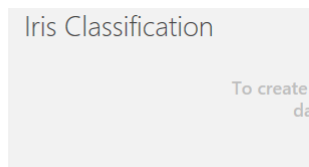
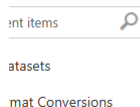
- 4 Choose and apply a learning algorithm

Score and test the model

- 5 Predict new iris labels

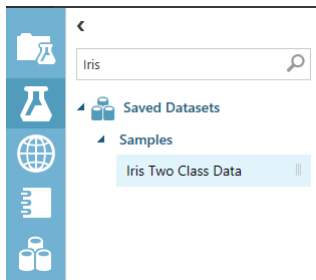
Step 1: Get Data

- 1 Create a new experiment by clicking **+NEW** at the bottom of the Machine Learning Studio window, select **EXPERIMENT**, and then select **Blank Experiment**.
- 2 The experiment is given a default name that you can see at the top of the canvas. Select this text and rename it to something meaningful, for example, **Iris Classification**. The name doesn't need to be unique.



Step 1: Get Data

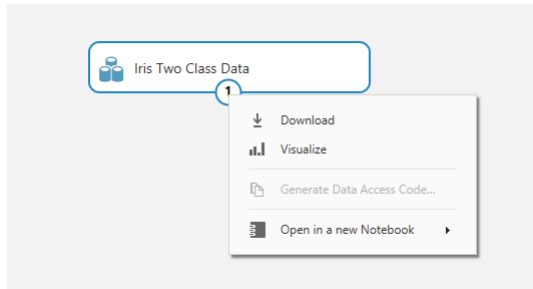
- 8 To the left of the experiment canvas is a palette of datasets and modules. Type **Iris** in the Search box at the top of this palette to find the dataset labeled **Iris Two Class Data**. Drag this dataset to the experiment canvas.



Step 2: Prepare Data

Dataset usually requires some preprocessing before it can be analyzed, as the data may be corrupted, inaccurate, or even missing in the database. Fortunately, in the iris dataset, the data is clean and do not require preprocessing.

Let's see what the data looks like using Azure. Click the output port at the bottom of the automobile dataset, and then select **Visualize**.



Step 3: Define the columns used

The goal of the iris classification problem is to find a machine learning model $h(\cdot)$ to predict whether the given sample \mathbf{x} is an iris or not, i.e. $y \in \{0, 1\}$, where 1 indicates the sample is an iris.

Specifically, we are going to select a supervised machine learning model $h(\cdot)$ which maps an input sample $\mathbf{x} = (x_1, x_2, \dots, x_n)$ to an output y . That is $y = h(\mathbf{x})$ (you can view it as a math function.) Different machine learning algorithms have different approaches in finding such the model $h(\cdot)$.

Before to train a model, one thing has to be decided. That is what should the \mathbf{x} be to feed in the algorithm.

Step 3: Define the columns used

In Iris dataset, there are 5 columns: *class*, *sepal-length*, *sepal-width*, *petal-length*, and *petal-width*.

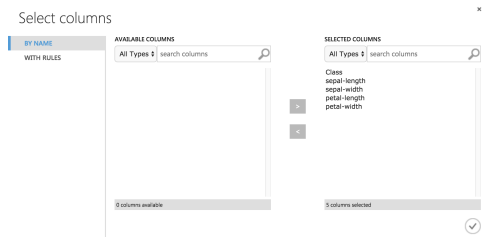
Obviously, the class indicates whether a sample is an iris or not, while the sepal-length, sepal-width, petal-length, and petal-width are the information about the sample.

Since they might all be relevant to classify whether a given sample is an iris, we will use them all to form the input \mathbf{x} . We also call \mathbf{x} to be “*features*”, as they are the features characterizing the sample.

Step 3: Define the columns used

Let's do it in Azure.

- 1 Drag **Select Columns in Dataset** module to the experiment canvas. Connect the left output port of the iris dataset to the input of the “Select Columns in Dataset” module.
- 2 Click **Launch column selector** in the **Properties** pane.
- 3 Select the columns used (both the features and label) as follow:



- 4 Click the check mark (OK) button.

Step 4: Choose and apply a learning algorithm

Now that the data is ready, constructing a predictive model consists of training and testing. We'll use our data to train the model, and then we'll test the model to see how closely it is able to predict the label.

- 1 Select and drag the “Split Data” module to the experiment canvas and connect it to the last “Select Columns in Dataset” module.
- 2 Click the “Split Data” module to select it. Find the **Fraction of rows in the first output dataset** (in the **Properties** pane to the right of the canvas) and set it to 0.75. This way, we'll use 75 percent of the data to train the model, and hold back 25 percent for testing (later, you can experiment with using different percentages).
- 3 Visualize the two output ports to see the split result.

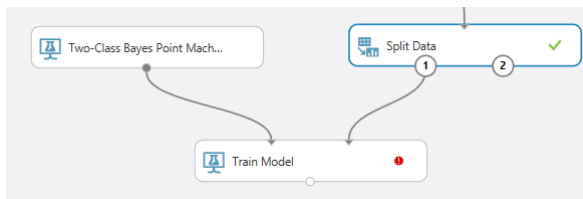
Step 4: Choose and apply a learning algorithm

- 4 To select the learning algorithm, expand the Machine Learning category in the module palette to the left of the canvas, and then expand Initialize Model. This displays several categories of modules that can be used to initialize machine learning algorithms.

For this experiment, select the “Two-Class Bayes Point Machine” which is the naive Bayes classifier, and drag it to the experiment canvas.

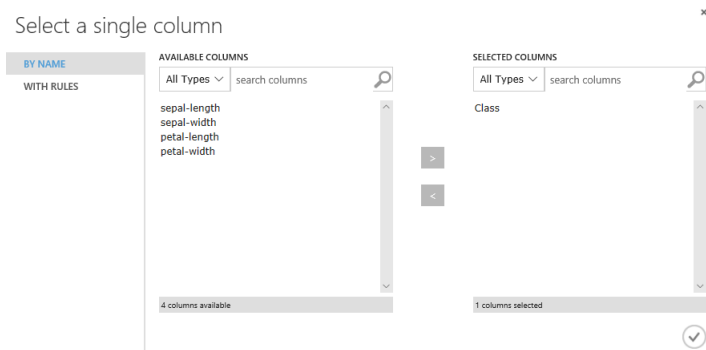
Step 4: Choose and apply a learning algorithm

- 5 Find and drag the “Train Model” module to the experiment canvas. Connect the output of the “Linear Regression” module to the left input of the “Train Model” module, and connect the training data output (left port) of the “Split Data” module to the right input of the “Train Model” module.



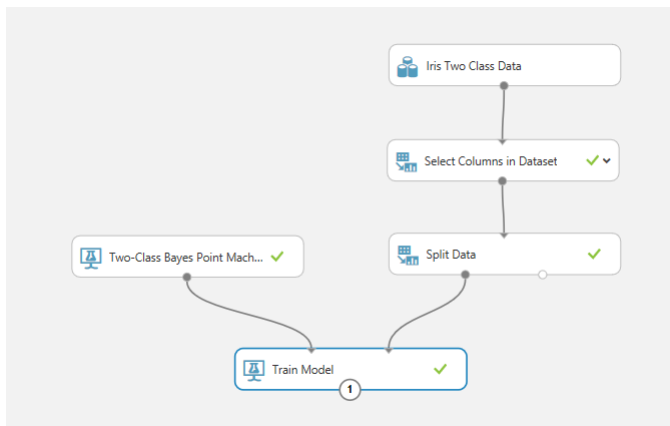
Step 4: Choose and apply a learning algorithm

- 6 Click the “Train Model” module, click **Launch column selector** in the **Properties** pane, and then select the **class** column. This is the value that our model is going to predict.



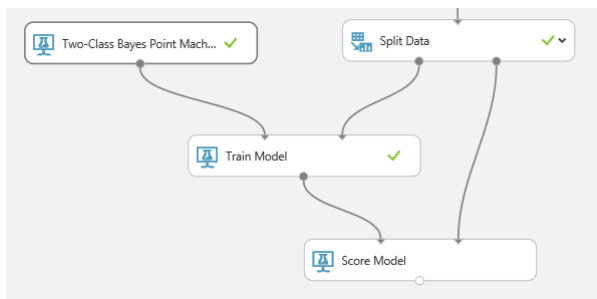
- 7 Run the experiment.

The computation graph up to this moment:



Step 5: Predict new iris label

- 1 Find and drag the Score Model module to the experiment canvas. Connect the output of the “Train Model” module to the left input port of “Score Model”. Connect the test data output (right port) of the Split Data module to the right input port of “Score Model”.





Step 5: Predict new iris label

- Run the experiment and view the output from the “Score Model” module (click the output port of Score Model and select Visualize). The output shows the predicted values and the known values from the test data.

Iris Classification > Score Model > Scored dataset

rows 25 columns 7

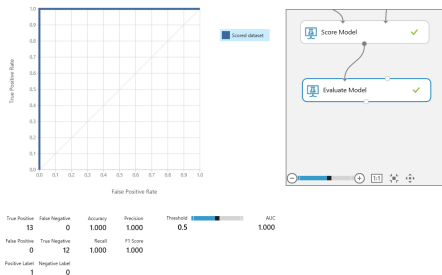
view as  

	Class	sepal-length	sepal-width	petal-length	petal-width	Known value Scored Labels	Predicted value Scored Probabilities
	0	5.1	3.8	1.6	0.2	0	0.015517
	0	4.6	3.6	1	0.2	0	0.016605
	0	5.2	3.5	1.5	0.2	0	0.019403
	1	6	3	4.8	1.8	1	0.968053
	1	6.3	2.8	5.1	1.5	1	0.914737
	0	5.4	3.4	1.7	0.2	0	0.021879
	0	5	3.2	1.2	0.2	0	0.023837
	0	4.7	3.2	1.6	0.2	0	0.027099
	1	6.3	2.9	5.6	1.8	1	0.969784
	1	6.7	3.1	5.6	2.4	1	0.996486
	0	4.7	3.2	1.3	0.2	0	0.025158

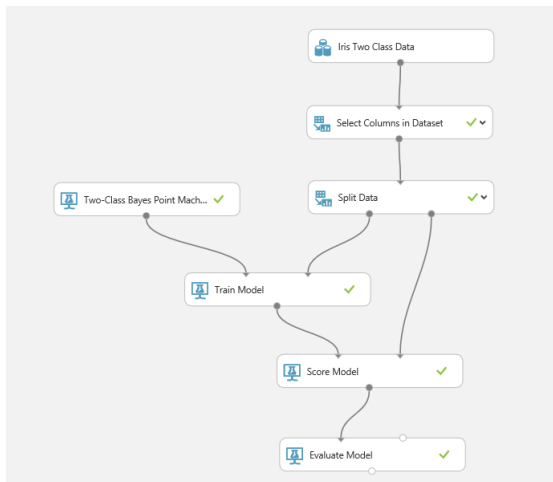
Step 5: Predict new iris label

- ③ Finally, we test the quality of the results. Select and drag the “Evaluate Model” module to the experiment canvas, and connect the output of the “Score Model” module to the left input of “Evaluate Model”.
- ④ Run the experiment.
- ⑤ Visualize the “Evaluate Model”

Iris Classification > Evaluate Model > Evaluation results



The final computation graph



Cross-Validation

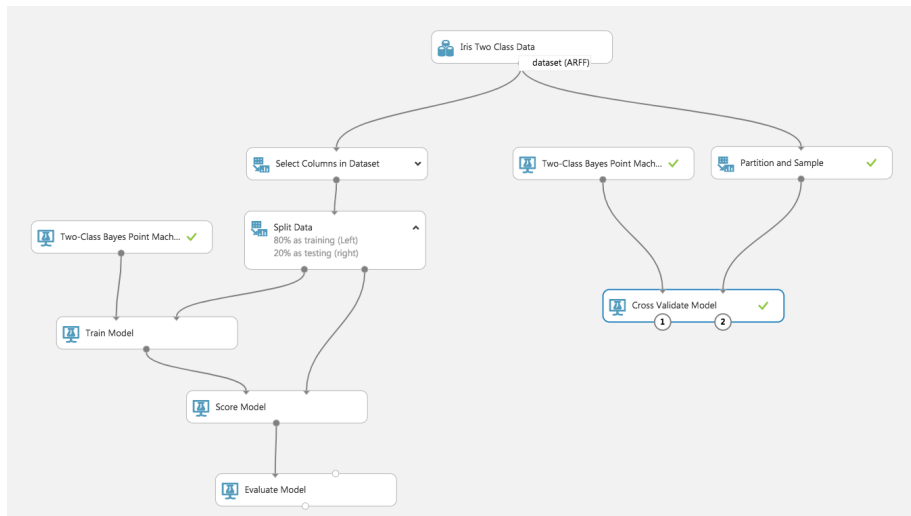
Apart from the training set and the test set, there is another dataset called validation set to assess both the variability of a dataset and the reliability of any model trained using that data.

The best practice of doing the validation in machine learning is *cross-validation* which randomly divides the training data into a number of partitions, also called *folds*.

Question: What are the pros and cons of using cross-validation different instead of testing?

Using Cross-Validation in Azure

First, build the following computation graph:



Using Cross-Validation in Azure

Modify the setting with the following steps:

- 1 In the property of **Partition and Sample**, follow the settings in the image on the right to enable the 5-fold cross validation.
- 2 In the property of the **Cross Validate Model**, select “Class” label in “label column” by launching the column selector.

Partition and Sample

Partition or sample mode

Assign to Folds

☐ Use replacement in...

☒ Randomized split

Random seed

0

Specify the partitioner method...

Partition evenly

Specify number of folds...

5

Stratified split

True

Stratification key column

Selected columns:
Column names: Class

Launch column selector












Using Cross-Validation in Azure

Finally, run the experiment and visualize the result in **Cross Validate Model**. You can see the evaluation result in each fold.

Iris Classification > Cross Validate Model > Evaluation results by fold

rows
7

columns
10

	Fold Number	Number of examples in fold	Model	Accuracy	Precision	Recall	F-Score	AUC	Average Log Loss	Training Log Loss
view as 										
	0	20	Microsoft.Analytics.Modules.BayesPointMachineClassifiers.Dll.BackwardCompatibleBinaryBayesPointMachineClassifier	1	1	1	1	1	0.033452	95.173909
	1	20	Microsoft.Analytics.Modules.BayesPointMachineClassifiers.Dll.BackwardCompatibleBinaryBayesPointMachineClassifier	1	1	1	1	1	0.026106	96.23363
	2	20	Microsoft.Analytics.Modules.BayesPointMachineClassifiers.Dll.BackwardCompatibleBinaryBayesPointMachineClassifier	1	1	1	1	1	0.018057	97.394934
	3	20	Microsoft.Analytics.Modules.BayesPointMachineClassifiers.Dll.BackwardCompatibleBinaryBayesPointMachineClassifier	1	1	1	1	1	0.024413	96.477919
			Microsoft.Analytics.Modules.BayesPointMachineClassifiers.Dll.BackwardCompatibleBinaryBayesPointMachineClassifier							

Statistics and Visualizations

Optional Exercises

- What is the “Random seed” in “Split data”? Why is it useful?
- What would be the potential implications if we increase or decrease the fraction of rows split in “Split Data”.
- Try out other classification algorithms. (Further question: How can you compare the performances of two different models? (Refer to the demo of breast cancer detection in “reference and further reading”))
- Try out the Machine learning tutorial (in “reference and further reading”). See what regression problem is and how the data preprocessing is done.

Reference and Further Reading

- Machine learning tutorial: Create your first data science experiment in Azure Machine Learning Studio: available in <https://docs.microsoft.com/en-us/azure/machine-learning/studio/create-experiment>
- Demo of breast cancer detection using two-class naive Bayes classifier in Azure ML Studio: available in <https://gallery.cortanaintelligence.com/Experiment/Breast-cancer-detection>