

Using Azure Machine Learning Studio

What Is Microsoft Azure Machine Learning Studio?

Microsoft Azure Machine Learning Studio (henceforth referred to as *MAML*) is an online collaborative, drag-and-drop tool for building machine learning models. Instead of implementing machine learning algorithms in languages like Python or R, MAML encapsulates the most-commonly used machine learning algorithms as modules, and it lets you build learning models visually using your dataset. This shields the beginning data science practitioners from the details of the algorithms, while at the same time offering the ability to fine-tune the hyperparameters of the algorithm for advanced users. Once the learning model is tested and evaluated, you can publish your learning models as web services so that your custom apps or BI tools, such as Excel, can consume it. What's more, MAML supports embedding your Python or R scripts within your learning models, giving advanced users the opportunity to write custom machine learning algorithms.

In this chapter, you will take a break from all of the coding that you have been doing in the previous few chapters. Instead of implementing machine learning using Python and Scikit-learn, you will take a look at how to use the MAML to perform machine learning visually using drag-and-drop.

An Example Using the Titanic Experiment

Now that you have a good sense of what machine learning is and what it can do, let's get started with an experiment using MAML. For this experiment, you will be using a classic example in machine learning—predicting the survival of a passenger on the Titanic.

In case you are not familiar with the Titanic, on April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1,502 out of 2,224 passengers and crew. While the main reason for the deaths was due to insufficient lifeboats, of those who survived, most of them were women, children, and the upper-class. As such, this presents a very interesting experiment in machine learning. If we are given a set of data points, containing the various profiles of passengers (such as gender, cabin class, age, and so forth) and whether they survived the sinking, it would be interesting for us to use machine learning to predict the survivability of a passenger based on his/her profile.

Interestingly, you can get the Titanic data from Kaggle (<https://www.kaggle.com/c/titanic/data>). Two sets of data are provided (see Figure 11.1):

- Training set
- Testing set

Getting Started Prediction Competition

Titanic: Machine Learning from Disaster

Start here! Predict survival on the Titanic and get familiar with ML basics

Kaggle · 6,946 teams · 3 years to go

Overview **Data** Kernels Discussion Leaderboard Rules Team My Submissions **Submit Predictions**

Training Data

gender_submission.csv	train.csv 59.76 KB	Download
test.csv		
train.csv		

Data Introduction

Overview

The data has been split into two groups:

- training set (train.csv)
- test set (test.csv)

The **training set** should be used to build your machine learning models. For the training set, we provide the outcome (also known as the "ground truth") for each passenger. Your model will be based on "features" like passengers' gender and class. You can also use [feature engineering](#) to create new features.

The **test set** should be used to see how well your model performs on unseen data. For the test set, we do not provide the ground truth for each passenger. It is your job to predict these outcomes. For each passenger in the test set, use the model you trained to predict whether or not they survived the sinking of the Titanic.

Figure 11.1: You can download the training and testing datasets from Kaggle

You use the training set to train your learning model so that you can use it to make predictions. Once your learning model is trained, you will make use of the testing set to predict the survivability of passengers.

Because the testing test does not contain a label specifying if a passenger survived, we will not use it for this experiment. Instead, we will only use the training set for training and testing our model.

Once the training set is downloaded, examine its contents (see Figure 11.2).

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr.	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mr	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, M	female	26	0	0	STON/O2. 31	7.925		S
4	1	1	Futrelle, Mrs	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. W	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr. J	male		0	0	330877	8.4583		Q
7	0	1	McCarthy, M	male	54	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Mas	male	2	3	1	349909	21.075		S
9	1	3	Johnson, M	female	27	0	2	347742	11.1333		S
10	1	2	Nasser, Mrs.	female	14	1	0	237736	30.0708		C
11	1	3	Sandstrom, f	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, Mis	female	58	0	0	113783	26.55	C103	S
13	0	3	Saunderscock	male	20	0	0	A/5. 2151	8.05		S
14	0	3	Andersson, A	male	39	1	5	347082	31.275		S
15	0	3	Vestrom, M	female	14	0	0	350406	7.8542		S
16	1	2	Hewlett, Mrs	female	55	0	0	248706	16		S
17	0	3	Rice, Master	male	2	4	1	382652	29.125		Q
18	1	2	Williams, M	male		0	0	244373	13		S
19	0	3	Vander Planck	female	31	1	0	345763	18		S
20	1	3	Masseimani, f	female		0	0	2649	7.225		C
21	0	2	Fynney, Mr.	male	35	0	0	239865	26		S
22	0	3	Beesley, M	male	34	0	0	348600	13.05		S

Figure 11.2: Examining the data in Excel

The training set should have the following fields:

PassengerId: A running number indicating the row of records.

Survived: If the particular passenger survived the sinking. This is the label of the dataset for our experiment.

Pclass: Ticket class that the passenger is holding.

Name: Name of the passenger.

Sex: Gender of the passenger.

Age: Age of the passenger.

SibSp: Number of siblings/spouses aboard the Titanic.

Parch: Number of parents/children aboard the Titanic.

Ticket: Ticket number.

Fare: Fare paid by the passenger.

Cabin: Cabin number of the passenger.

Embarked: Place of embarkation. Note that C = Cherbourg, Q = Queenstown, and S = Southampton.

Using Microsoft Azure Machine Learning Studio

We are now ready to load the data into MAML. Using your web browser, navigate to <http://studio.azureml.net>, and click the “Sign up here” link (see Figure 11.3).

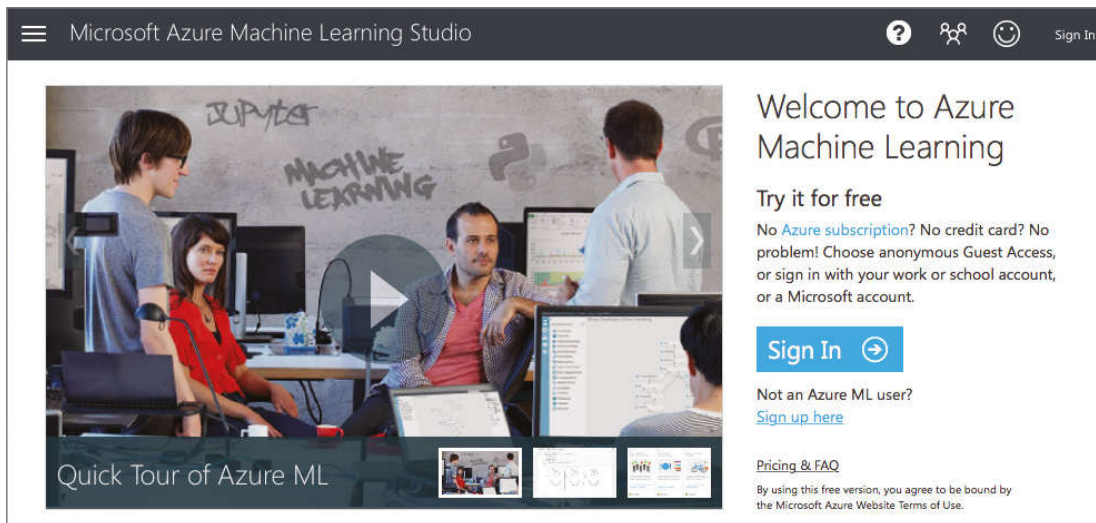


Figure 11.3: Click the “Sign up here” link for first-time Azure Machine Learning users

If you just want to experience MAML without any financial commitment, choose the Free Workspace option and click Sign In (see Figure 11.4).

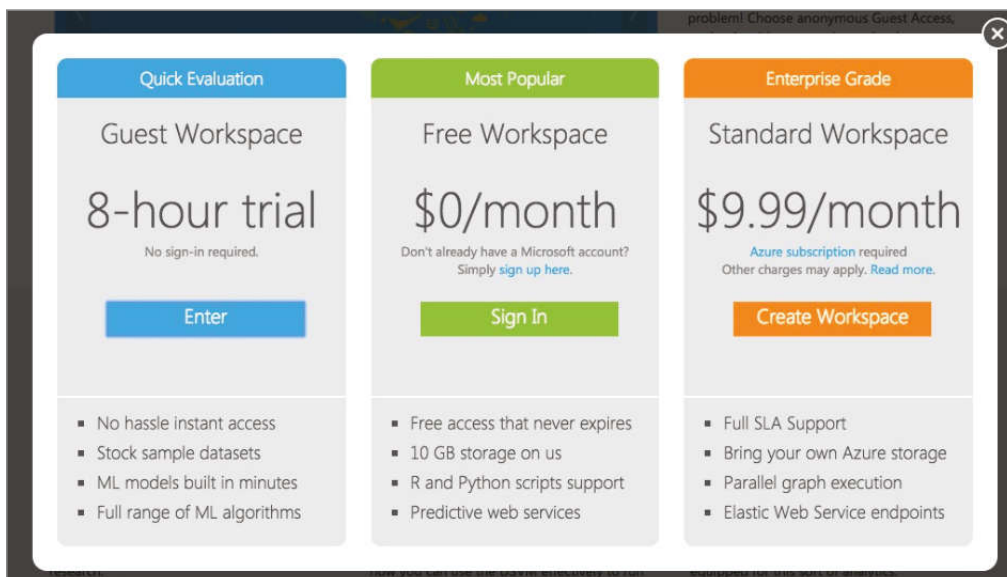


Figure 11.4: You can choose from the various options available to use MAML

Once you are signed in, you should see a list of items on the left side of the page (see Figure 11.5). I will highlight some of the items on this panel as we move along.

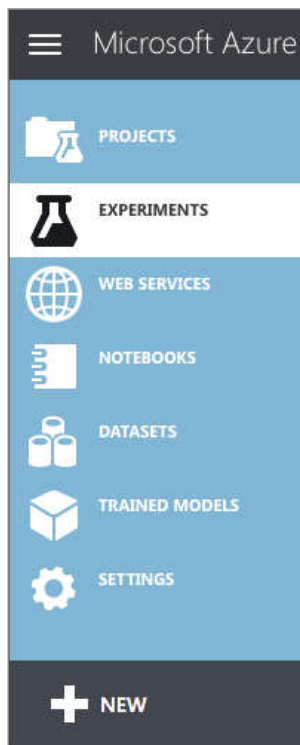


Figure 11.5: The left panel of MAML

Uploading Your Dataset

To create learning models, you need datasets. For this example, we will use the dataset that you have just downloaded.

Click the + NEW item located at the bottom-left of the page. Select DATASET on the left (see Figure 11.6), and then click the item on the right labeled FROM LOCAL FILE.

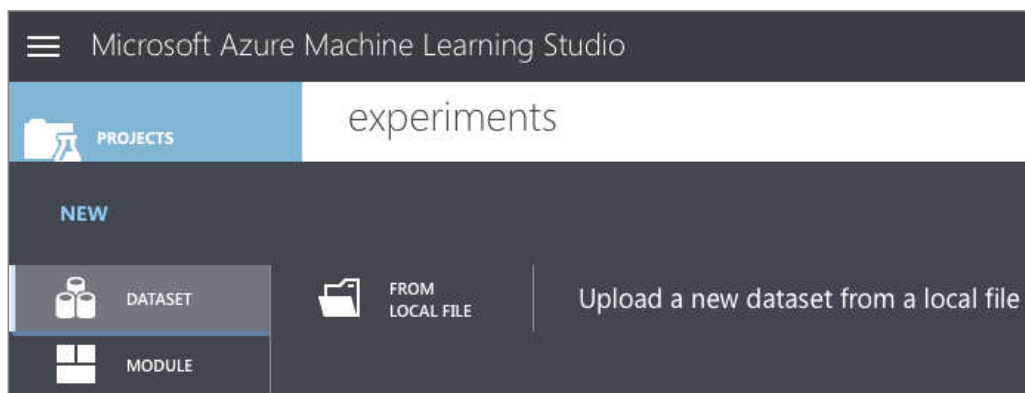
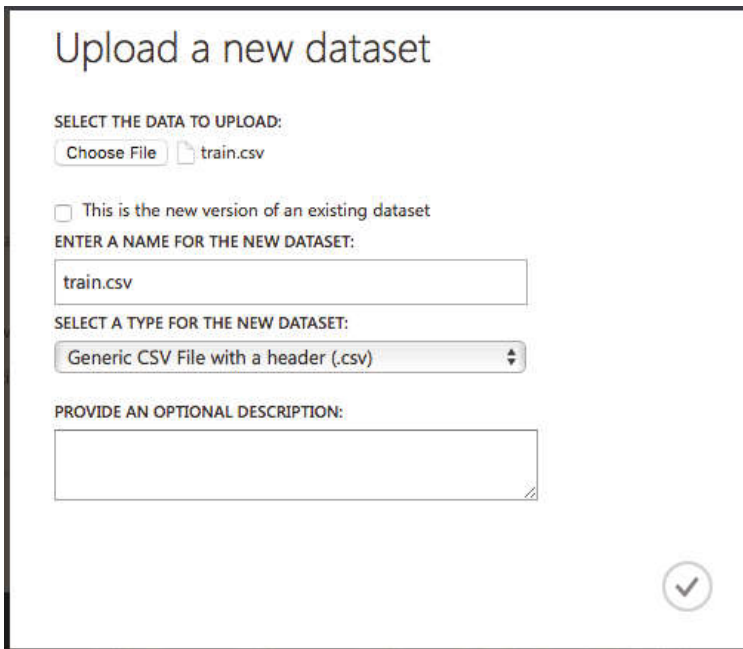


Figure 11.6: Uploading a dataset to the MAML

Click the Choose File button (see Figure 11.7) and locate the training set downloaded earlier. When finished, click the tick button to upload the dataset to the MAML.



The screenshot shows a web form titled "Upload a new dataset". It contains the following elements:

- SELECT THE DATA TO UPLOAD:** A "Choose File" button and a file icon next to "train.csv".
- ☐ This is the new version of an existing dataset
- ENTER A NAME FOR THE NEW DATASET:** A text input field containing "train.csv".
- SELECT A TYPE FOR THE NEW DATASET:** A dropdown menu showing "Generic CSV File with a header (.csv)".
- PROVIDE AN OPTIONAL DESCRIPTION:** A large empty text area.
- A circular button with a checkmark icon at the bottom right.

Figure 11.7: Choose a file to upload as a dataset

Creating an Experiment

You are now ready to create an experiment in MAML. Click the + NEW button at the bottom-left of the page and select Blank Experiment (see Figure 11.8).

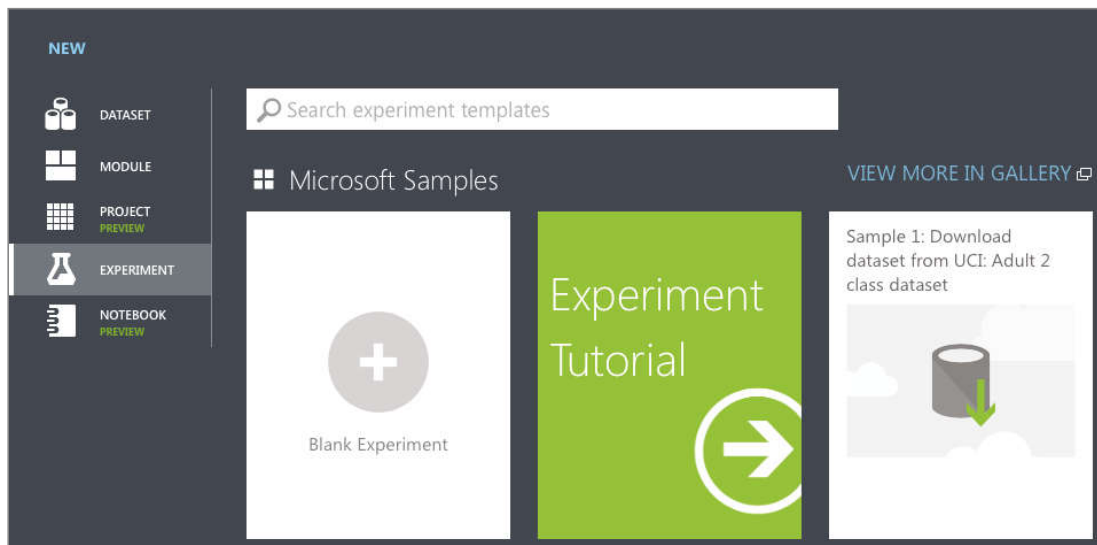


Figure 11.8: Creating a new blank experiment in MAML

You should now see the canvas, as shown in Figure 11.9.

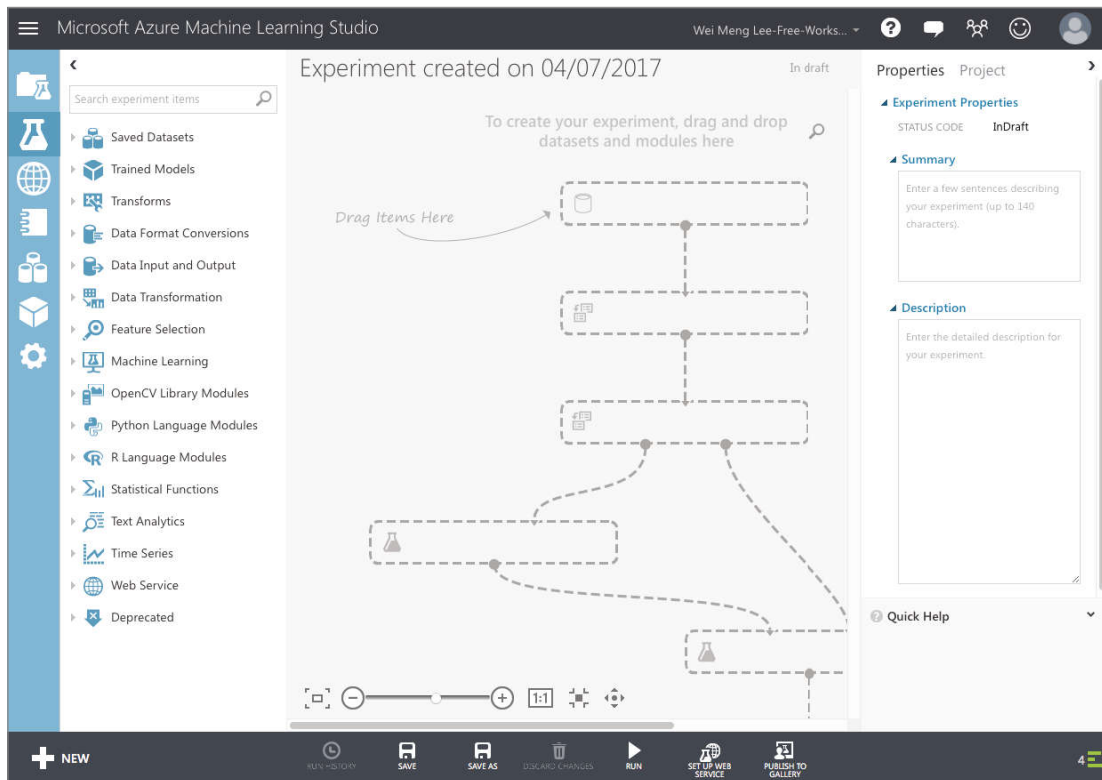


Figure 11.9: The canvas representing your experiment

You can give a name to your experiment by typing it over the default experiment name at the top (see Figure 11.10).

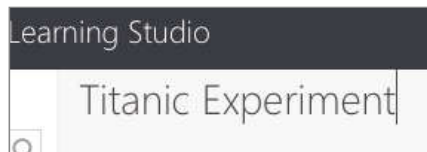


Figure 11.10: Naming your experiment

Once that is done, let's add our training dataset to the canvas. You can do so by typing the name of the training set in the search box on the left, and the matching dataset will now appear (see Figure 11.11).

Drag and drop the `train.csv` dataset onto the canvas (see Figure 11.12).

The `train.csv` dataset has an output port (represented by a circle with a 1 inside). Clicking it will reveal a context menu (see Figure 11.13).

Click Visualize to view the content of the dataset. The dataset is now displayed, as shown in Figure 11.14.

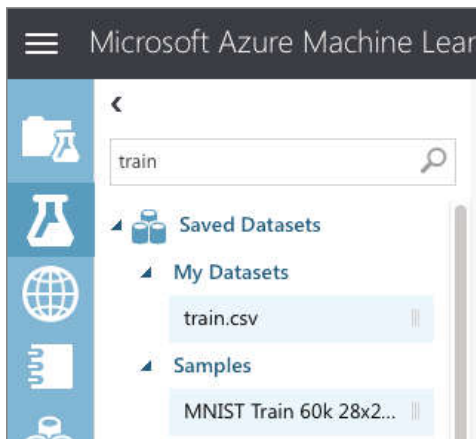


Figure 11.11: Using the dataset that you have uploaded

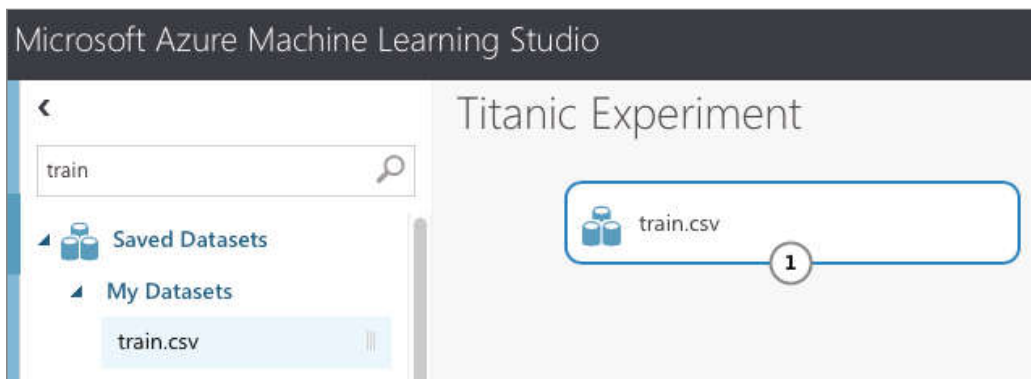


Figure 11.12: Dragging and dropping the dataset onto the canvas

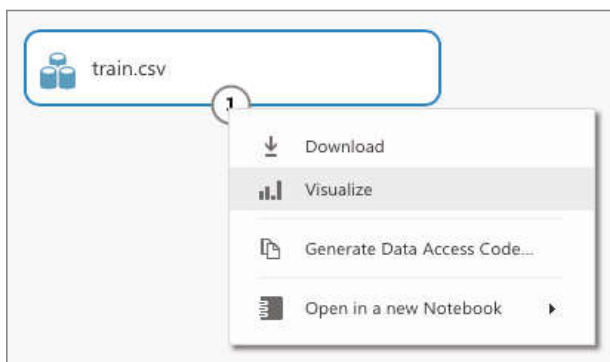


Figure 11.13: Visualizing the content of the dataset

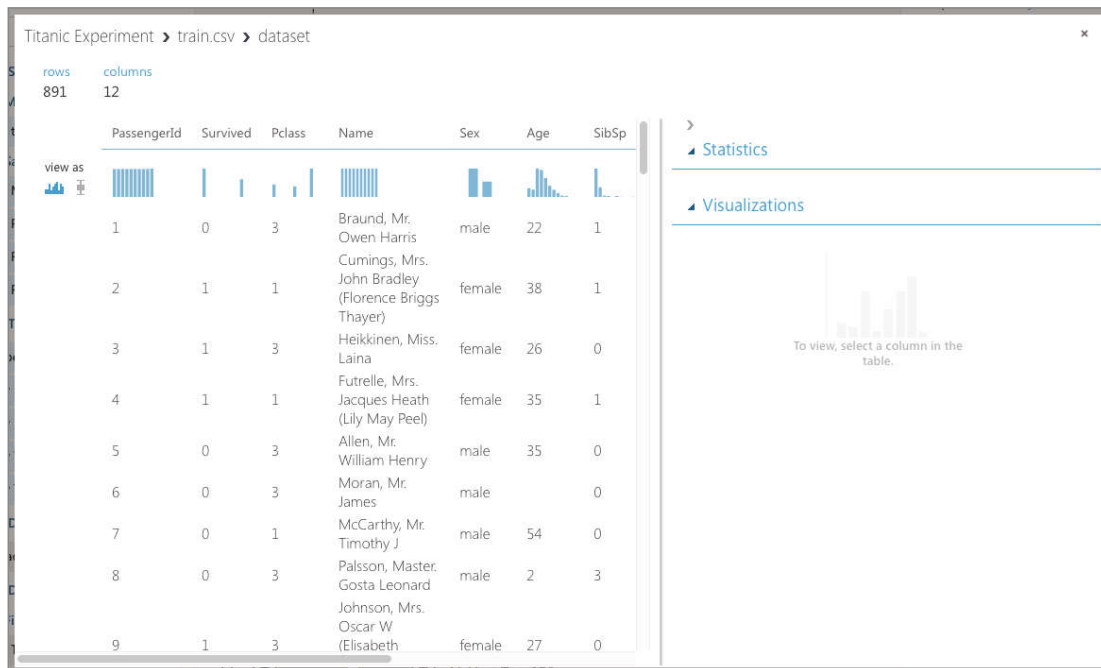


Figure 11.14: Viewing the dataset

Take a minute to scroll through the data. Observe the following:

- The **PassengerID** field is simply a running number, and it does not provide any information with regard to the passenger. This field should be discarded when training your model.
- The **Ticket** field contains the ticket number of the passengers. In this case, however, a lot of these numbers seem to be randomly generated. Thus, it is not very useful in helping us to predict the survivability of a passenger and hence should be discarded.
- The **Cabin** field contains a lot of missing data. Fields that have a lot of missing data do not provide insights to our learning model and hence should be discarded.
- If you select the **Survived** field, you will see the chart displayed on the bottom right of the window (see Figure 11.15). Because a passenger can either survive (represented by a 1) or die (represented by a 0), it does not make sense to have any values in between. However, since this value is represented as a numeric value, MAML would not be able to figure this

out unless you tell it. To fix this, you need to make this value a categorical value. A *categorical value* is a value that can take on one of a limited, and usually fixed, number of possible values.

- The **Pclass**, **SibSp**, and **Parch** fields should all be made categorical as well.

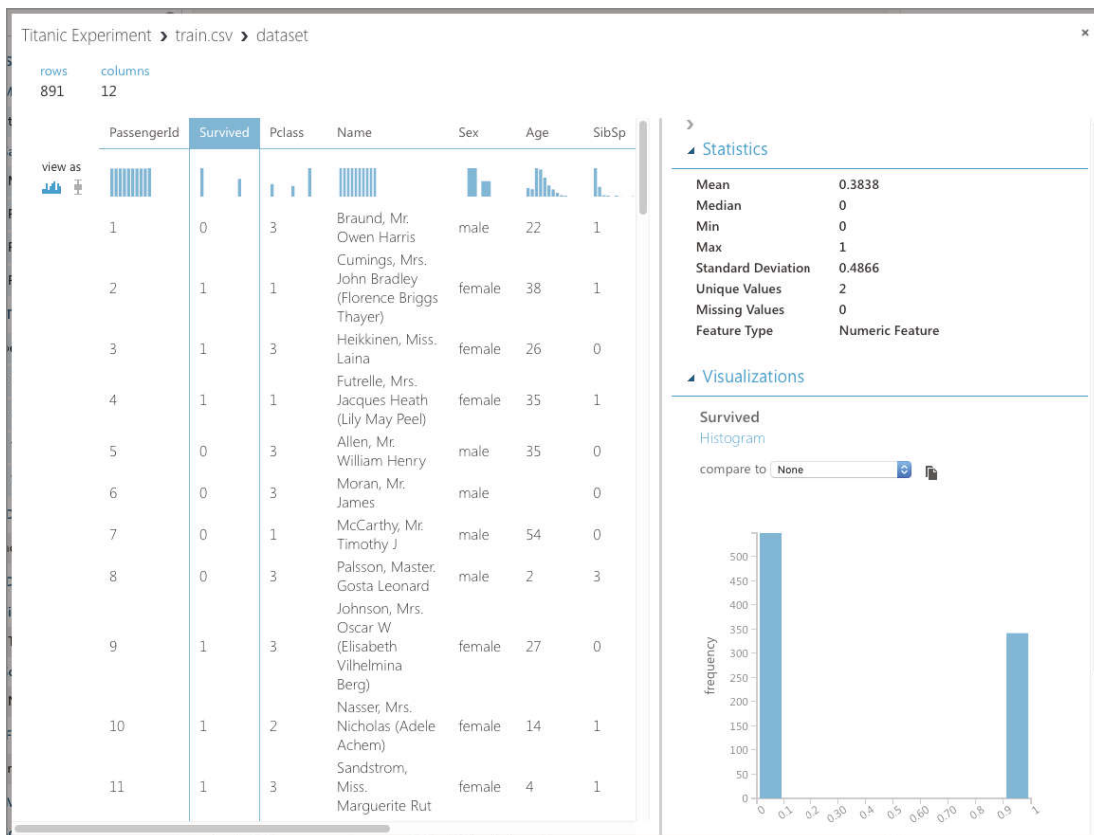


Figure 11.15: Viewing the Survived column

All of the fields that are not discarded are useful in helping us to create a learning model. These fields are known as *features*.

Filtering the Data and Making Fields Categorical

Now that we have identified the features we want, let's add the Select Columns in Dataset module to the canvas (see Figure 11.16).

In the Properties pane, click the Launch column selector and select the columns, as shown in Figure 11.17.

The Select Columns in Dataset module will reduce the dataset to the columns that you have specified. Next, we want to make some of the columns categorical. To do that, add the Edit Metadata module, as shown in Figure 11.18, and connect it as shown. Click the Launch column selector button, and select the **Survived**, **Pclass**, **SibSp**, and **Parch** fields. In the Categorical section of the properties pane, select "Make categorical."

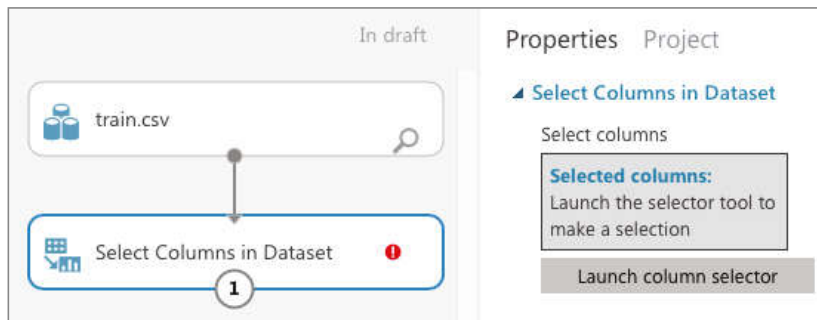


Figure 11.16: Use the Select Columns in Dataset module to filter columns

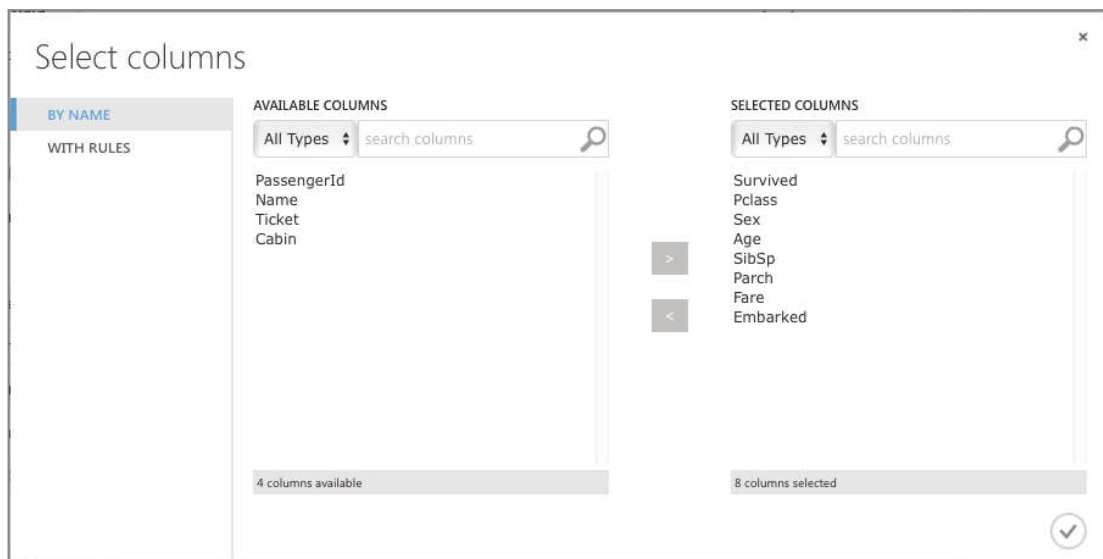


Figure 11.17: Selecting the fields that you want to use as features

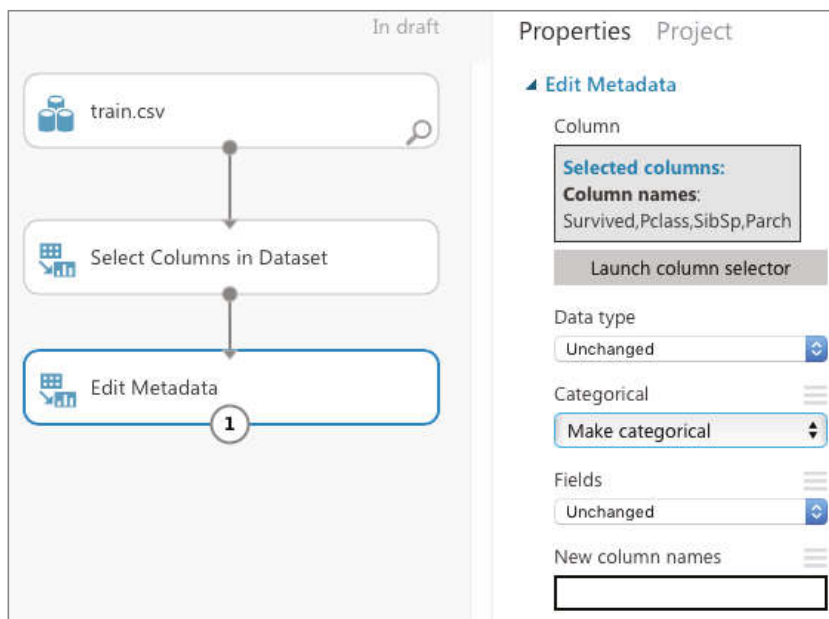


Figure 11.18: Making specific fields categorical

You can now run the experiment by clicking the RUN button located at the bottom of the MAML. Once the experiment is run, click the output port of the Edit Metadata module and select Visualize. Examine the dataset displayed.

Removing the Missing Data

If you examine the dataset returned by the Edit Metadata module carefully, you will see that the **Age** column has some missing values. It is always good to remove all those rows that have missing values so that those missing values will not affect the efficiency of the learning model. To do that, add a *Clean Missing Data* module to the canvas and connect it as shown in Figure 11.19. In the properties pane, set the “Cleaning mode” to “Remove entire row.”

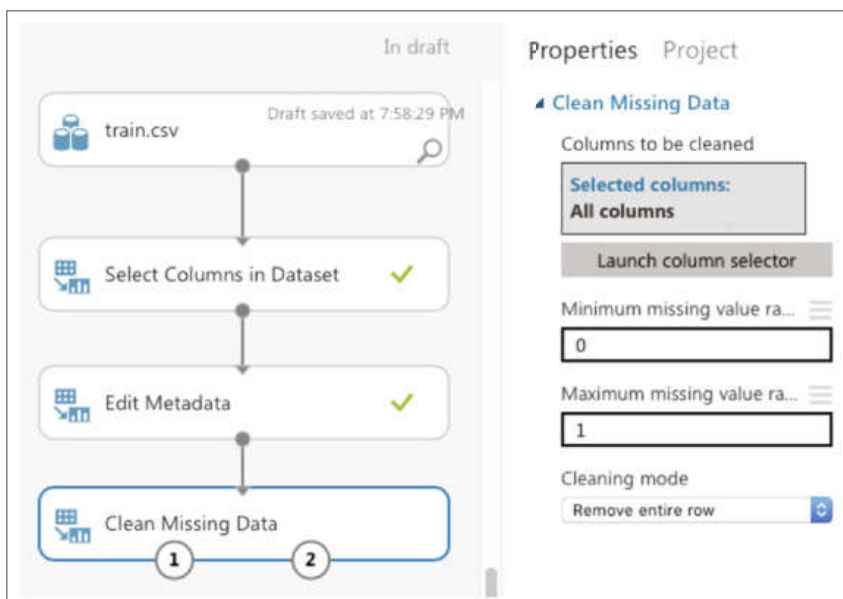


Figure 11.19: Removing rows that have missing values in the Age column

TIP You can also replace the missing values with the mean of the column, if you prefer.

Click RUN. The dataset should now have no more missing values. Also notice that the number of rows has been reduced to 712 (see Figure 11.20).

Splitting the Data for Training and Testing

When building your learning model, it is essential that you test it with sample data after the training is done. If you only have one single set of data, you can split it into two parts—one for training and one for testing. This is accomplished

by the Split Data module (see Figure 11.21). For this example, I am splitting 80 percent of the dataset for training and the remaining 20 percent for testing.

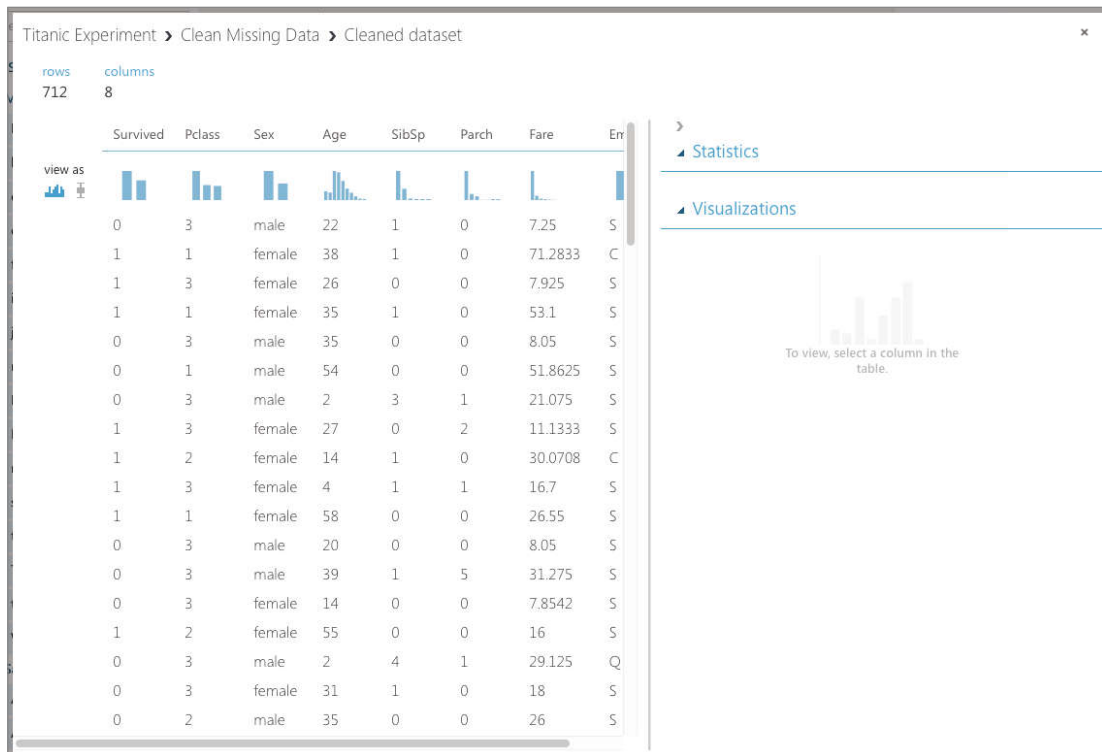


Figure 11.20: Viewing the cleaned and filtered dataset

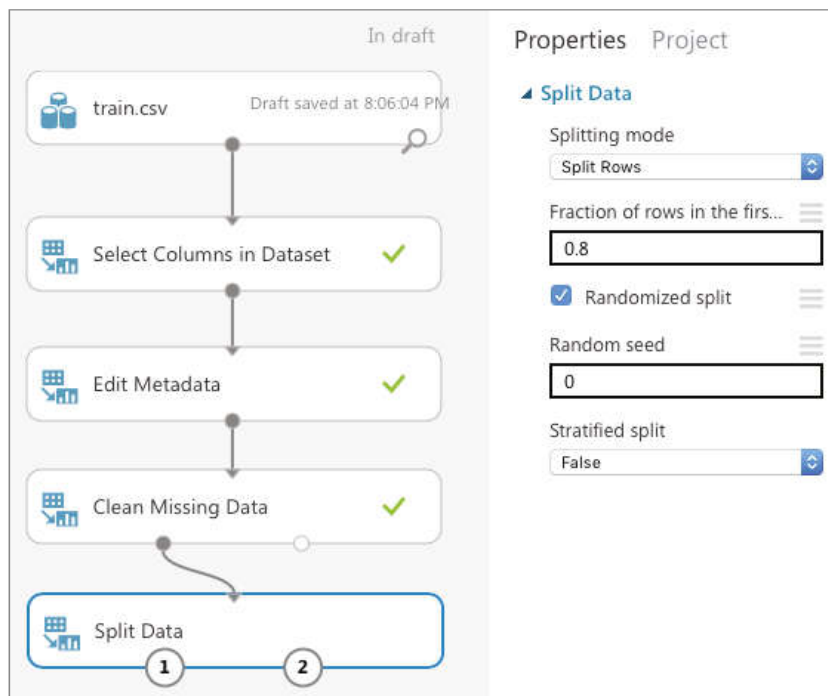


Figure 11.21: Splitting the data into training and testing datasets

The left output port of the Split Data module will return 80 percent of the dataset while the right output port will return the remaining 20 percent.

Training a Model

You are now ready to create the training model. Add the Two-Class Logistic Regression and Train Model modules to the canvas and connect them as shown in Figure 11.22. The Train Model module takes in a learning algorithm and a training dataset. You will also need to tell the Train Model module the label for which you are training it. In this case, it is the **Survived** column.

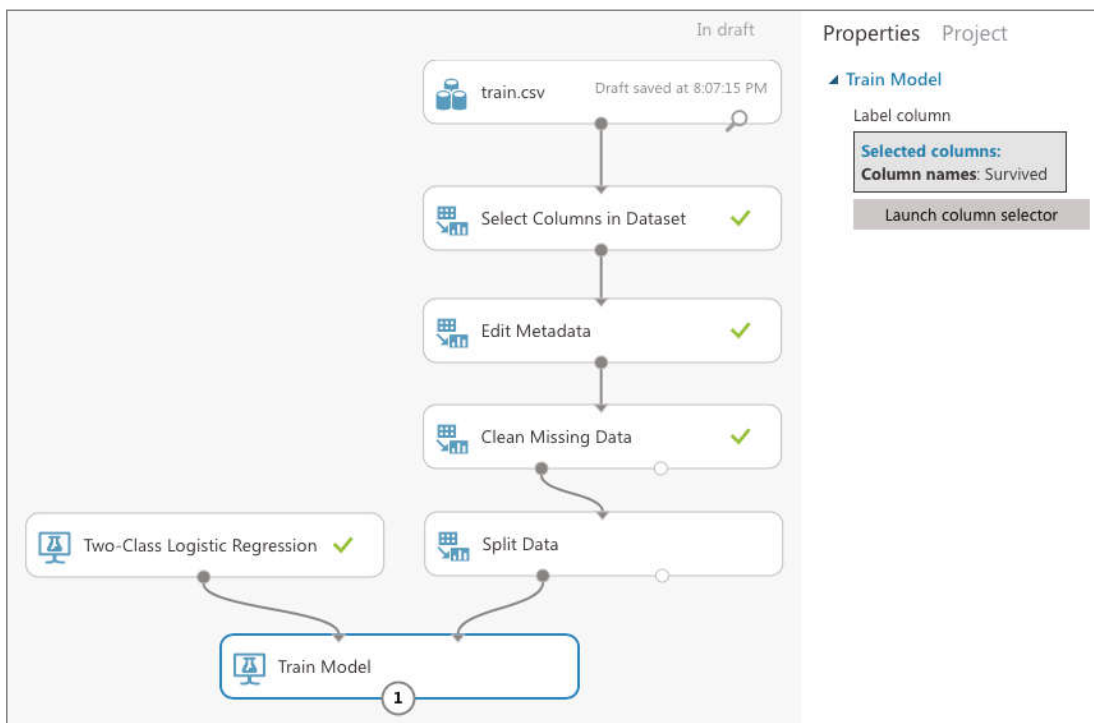


Figure 11.22: Training your model using the Two-Class Logistic Regression algorithm

Once you have trained the model, it is essential that you verify its effectiveness. To do so, use the Score Model module, as shown in Figure 11.23. The *Score Model* takes in a trained model (which is the output of the Train Model module) and a testing dataset.

You are now ready to run the experiment again. Click RUN. Once it is completed, select the **Scored Labels** column (see Figure 11.24). This column represents the results of applying the test dataset against the learning model. The column next to it, **Scored Probabilities**, indicates the confidence of the prediction. With the **Scored Labels** column selected, look at the right side of the screen and above the chart, select Survived for the item named “compare to.” This will plot the confusion matrix.

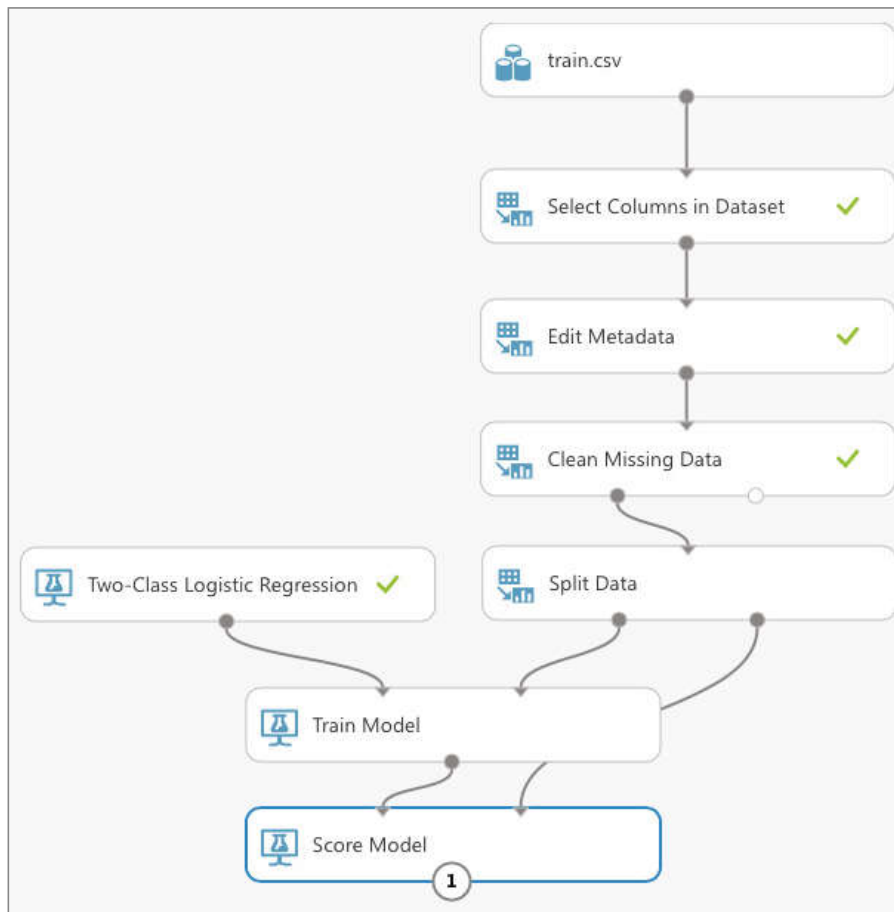


Figure 11.23: Scoring your model using the testing dataset and the trained model

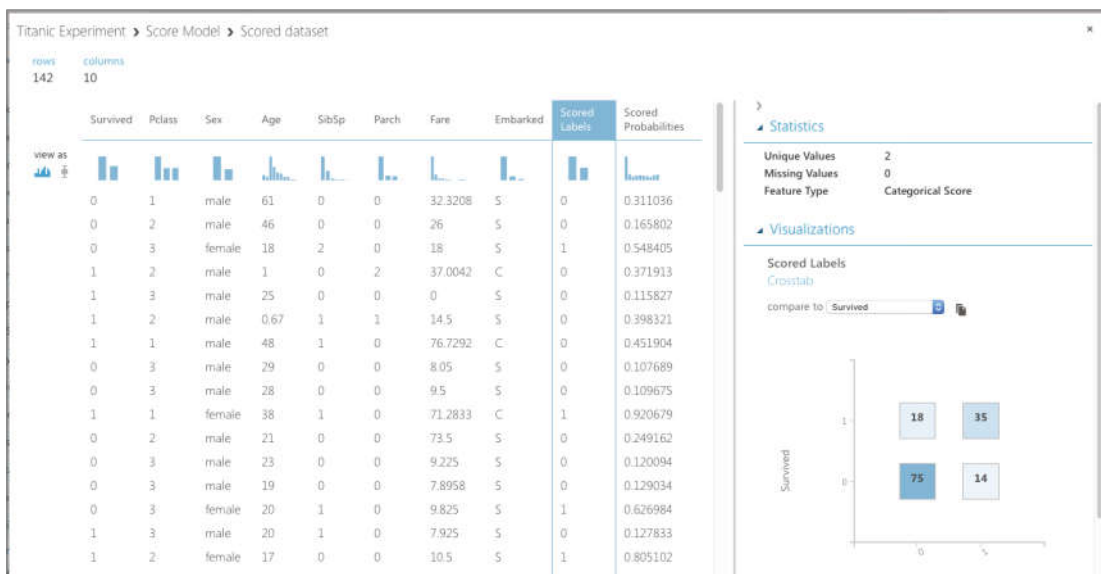


Figure 11.24: Viewing the confusion matrix for the learning model

The y-axis of the confusion matrix shows the actual survival information of passengers: 1 for survived and 0 for did not survive. The x-axis shows the prediction. As you can see, 75 were correctly predicted not to survive the disaster, and 35 were correctly predicted to survive the disaster. The two other boxes show the predictions that were incorrect.

Comparing Against Other Algorithms

While the numbers for the predictions look pretty decent, it is not sufficient to conclude at this moment that we have chosen the right algorithm for this problem. MAML comes with 25 machine learning algorithms for different types of problems. Now let's use another algorithm provided by MAML, Two-Class Decision Jungle, to train another model. Add the modules as shown in Figure 11.25.

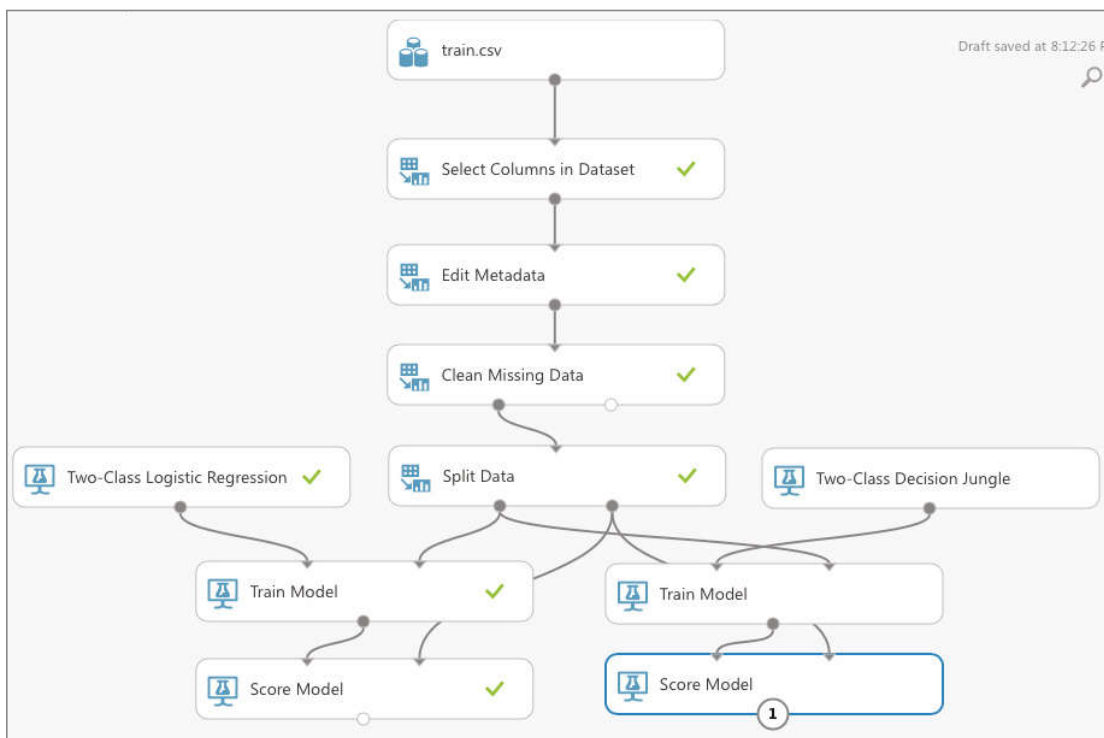


Figure 11.25: Using another algorithm for training the alternative model

TIP The Two-Class Decision Jungle algorithm is another machine learning algorithm that is based on decision trees. For this experiment, you can also use other algorithms provided by MAML, such as the Two-Class Logistic Regression and Two-Class Support Vector Machine.

Click Run. You can click the output port of the second Score Model module to view the result of the model, just like the previous learning model. However, it would be more useful to be able to compare them directly. You can accomplish this using the Evaluate Model module (see Figure 11.26).

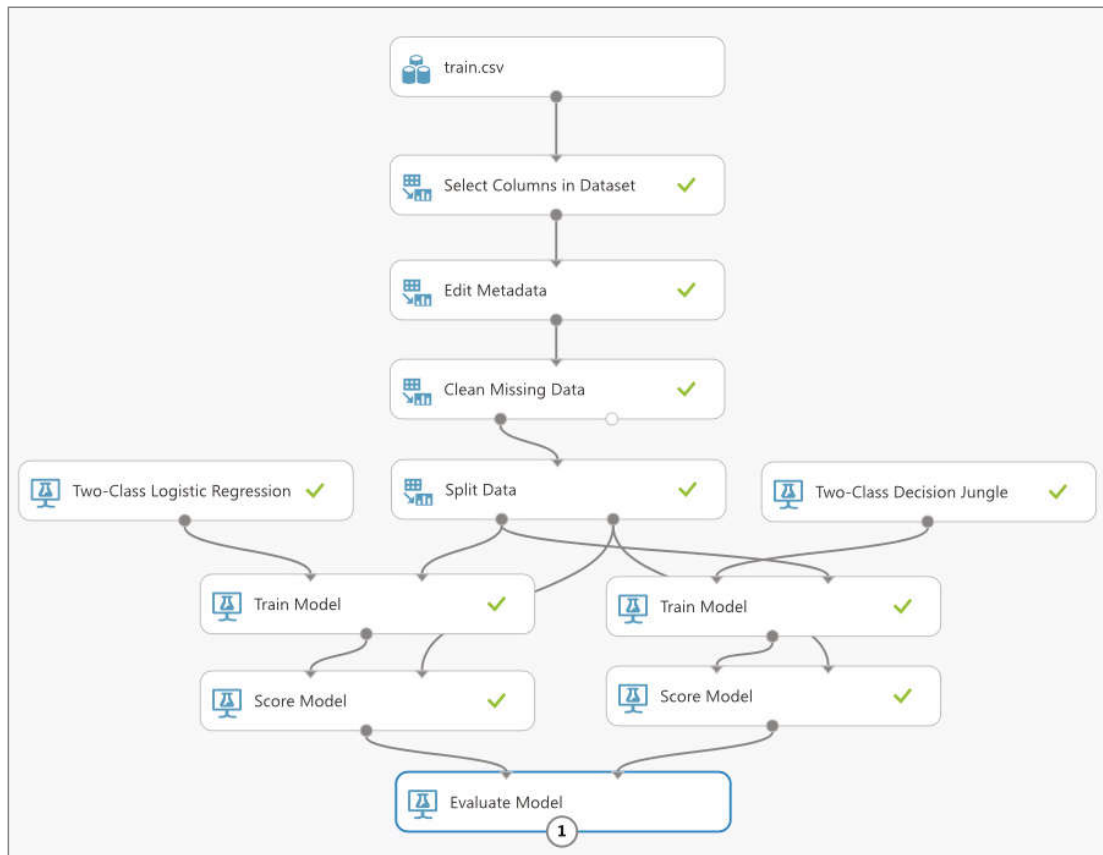


Figure 11.26: Evaluating the performance of the two models

Click RUN to run the experiment. When done, click the output port of the Evaluate Model module and you should see something like Figure 11.27.

The blue line represents the algorithm on the left input port of the Evaluate Model module (Two-Class Logistic Regression), while the red line represents the algorithm on the right (Two-Class Decision Jungle). When you click either the blue or red box, you will see the various metrics for each algorithm displayed below the chart.

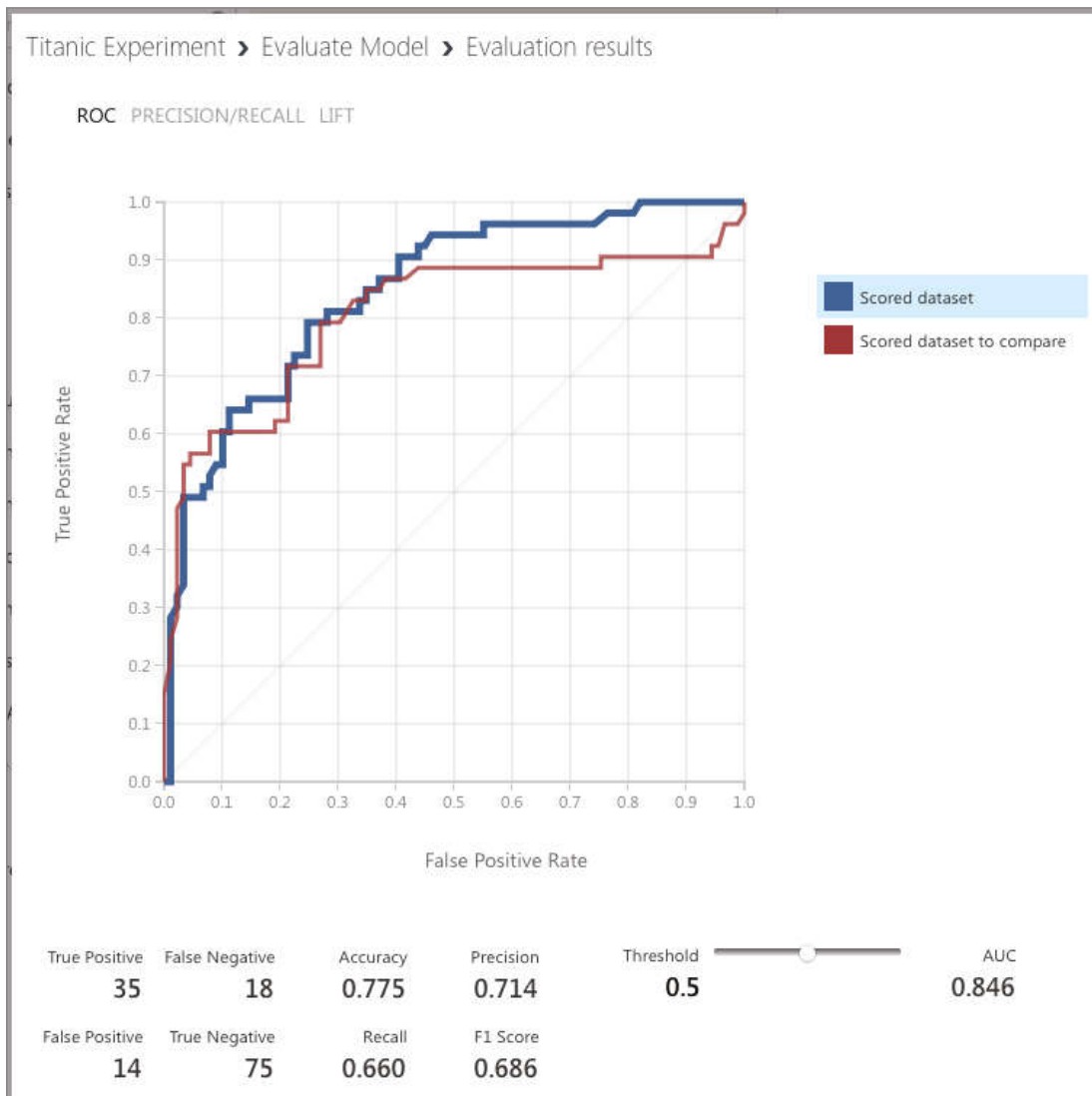


Figure 11.27: Viewing the metrics for the two learning algorithms

Evaluating Machine Learning Algorithms

Now that you have seen an experiment performed using two specific machine learning algorithms—Two-Class Logistic Regression and Two-Class Decision Jungle—let's step back a little and examine the various metrics that were generated by the *Evaluate Model* module. Specifically, let's define the meaning of the following terms:

True Positive (TP) The model correctly predicts the outcome as positive. In this case, the number of TP indicates the number of correct predictions that a passenger survived (positive) the disaster.

True Negative (TN) The model correctly predicts the outcome as negative (did not survive); that is, passengers were correctly predicted not to survive the disaster.

False Positive (FP) The model incorrectly predicted the outcome as positive, but the actual result is negative. In the Titanic example, it means that the passenger did not survive the disaster, but the model predicted the passenger to have survived.

False Negative (FN) The model incorrectly predicted the outcome as negative, but the actual result is positive. In this case, this means the model predicted that the passenger did not survive the disaster, but actually the passenger did.

This set of numbers is known as the *confusion matrix*. The confusion matrix is discussed in detail in Chapter 7, “Supervised Learning—Classification Using Logistic Regression.” So if you are not familiar with it, be sure to read up on Chapter 7.

Publishing the Learning Model as a Web Service

Once the most effective machine learning algorithm has been determined, you can publish the learning model as a web service. Doing so will allow you to build custom apps to consume the service. Imagine that you are building a learning model to help doctors diagnose breast cancer. Publishing as a web service would allow you to build apps to pass the various features to the learning model to make the prediction. Best of all, by using MAML, there is no need to handle the details of publishing the web service—MAML will host it for you on the Azure cloud.

Publishing the Experiment

To publish our experiment as a web service:

- Select the left Train Model module (since it has a better performance compared to the other).
- At the bottom of the page, hover your mouse over the item named SET UP WEB SERVICE, and click Predictive Web Service (Recommended).

TIP For this experiment, the best algorithm is the one that gives the highest AUC (Area Under the Curve) score.