



Microsoft Data Science Workshop

Lab Setup and Instruction Guide

Overview

In this lab, you will learn how to build a **Human Activity Classifier** with Azure Machine Learning. This classifier predicts somebody's activity class (sitting, standing up, standing, sitting down, walking) based on the use of wearable sensors. The point of this lab is to introduce you to the basics of creating and deploying a machine learning model in Azure ML, it is not intended to be a deep-dive into model design, validation and improvement.

This lab environment contains the following tasks:

1. Setup your Azure ML environment
2. Get the data
3. Build your model
4. Publish your model

What You'll Need

To perform the tasks, you will need the following:

- A Windows, Linux, or Mac OSX computer.
- A web browser and Internet connection.

1. Setup your Azure ML environment

There are several options to start with Azure ML: <https://azure.microsoft.com/en-us/services/machine-learning/>

Quick Evaluation	Most Popular	Enterprise Grade
Guest Workspace	Free Workspace	Standard Workspace
8-hour trial	\$0/month	\$9.99/month
No sign-in required.	Don't already have a Microsoft account? Simply sign up here .	Azure subscription required Other charges may apply. Read more .
Enter	Sign In	Create Workspace
<ul style="list-style-type: none">▪ No hassle instant access▪ Stock sample datasets▪ ML models built in minutes▪ Full range of ML algorithms	<ul style="list-style-type: none">▪ Free access that never expires▪ 10 GB storage on us▪ R and Python scripts support▪ Predictive web services	<ul style="list-style-type: none">▪ Full SLA Support▪ Bring your own Azure storage▪ Parallel graph execution▪ Elastic Web Service endpoints

If you don't have an Azure account already, we recommend you to use the **Free Workspace** option. Therefore, you would have to sign up for a Microsoft account. If you don't have one already, you can sign up for one at <https://signup.live.com/>.

2. Get the data

This classifier predicts somebody's activity class (sitting, standing up, standing, sitting down, walking). It is based on the **Human Activity Recognition** dataset. Human Activity Recognition (HAR) is an active research area, results of which have the potential to benefit the development of assistive technologies in order to support care of the elderly, the chronically ill and people with special needs. Activity recognition can be used to provide information about patients' routines to support the development of e-health systems. Two approaches are commonly used for HAR: image processing and use of wearable sensors. In this case we will use information generated by wearable sensors (Ugulino et al, 2012).

Understand the data source

In this lab we use the Human Activity Recognition Data from its source: <http://groupware.les.inf.puc-rio.br/har#ixzz2PyRdbAfA>. More info can also be found on the [UCI repository](#).

You can download the data from <http://groupware.les.inf.puc-rio.br/static/har/dataset-har-PUC-Rio-ugulino.zip> and extract the downloaded zip file to a convenient folder on your local computer.

The data has been collected during 8 hours of activities, 2 hours with each of the 2 men and 2 women, all adults and healthy. These people were wearing 4 accelerometers from [LiliPad Arduino](#), respectively positioned in the waist, left thigh, right ankle, and right arm. This resulted in a dataset with 165634 rows and 19 columns.

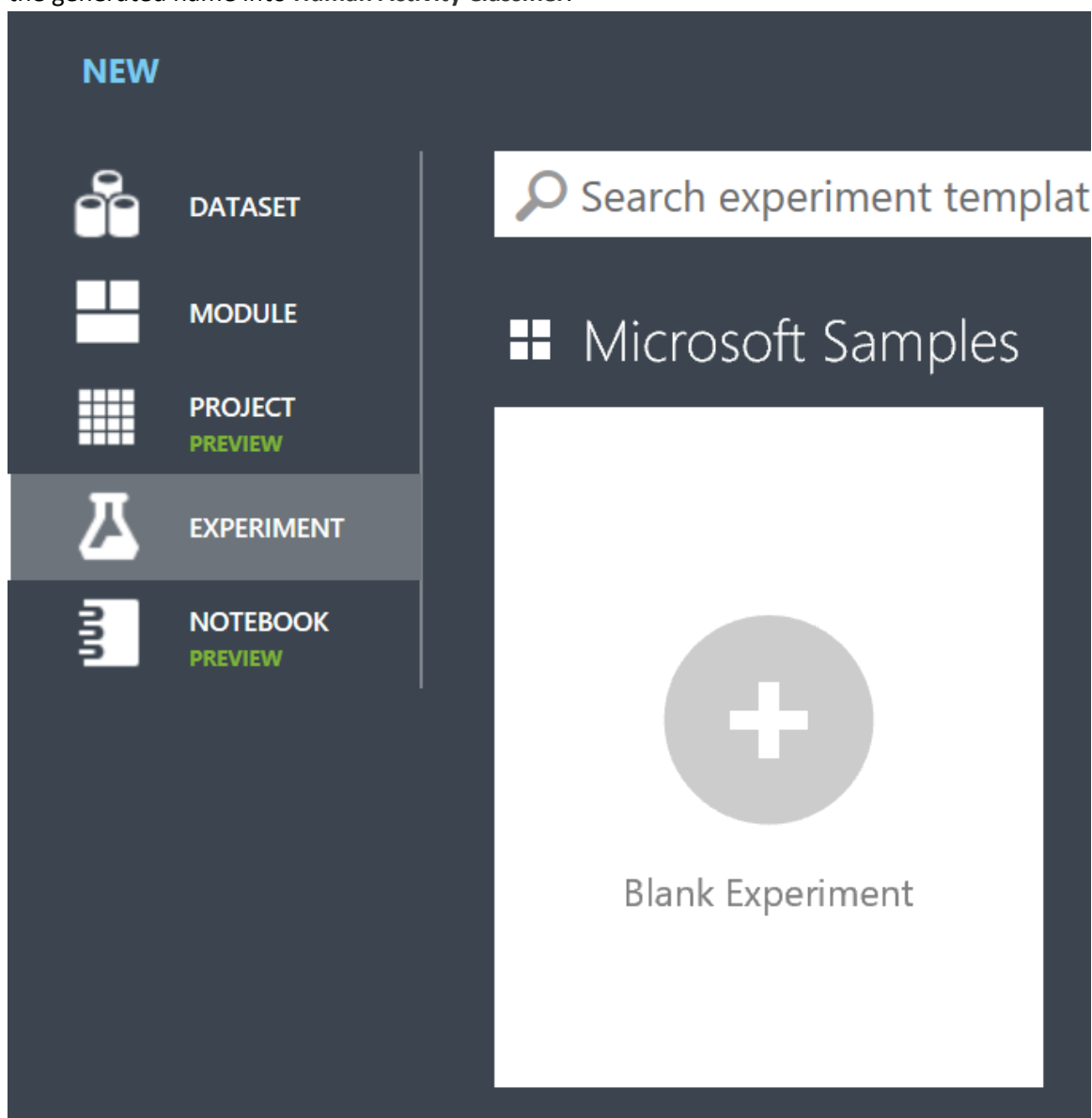
- user (text)
- gender (text)
- age (integer)
- how_tall_in_meters (real)
- weight (int)
- body_mass_index (real)
- x1 (type int, value of the axis 'x' of the 1st accelerometer, mounted on waist)
- y1 (type int, value of the axis 'y' of the 1st accelerometer, mounted on waist)
- z1 (type int, value of the axis 'z' of the 1st accelerometer, mounted on waist)
- x2 (type int, value of the axis 'x' of the 2nd accelerometer, mounted on the left thigh)
- y2 (type int, value of the axis 'y' of the 2nd accelerometer, mounted on the left thigh)
- z2 (type int, value of the axis 'z' of the 2nd accelerometer, mounted on the left thigh)
- x3 (type int, value of the axis 'x' of the 3rd accelerometer, mounted on the right ankle)
- y3 (type int, value of the axis 'y' of the 3rd accelerometer, mounted on the right ankle)
- z3 (type int, value of the axis 'z' of the 3rd accelerometer, mounted on the right ankle)
- x4 (type int, value of the axis 'x' of the 4th accelerometer, mounted on the right upper-arm)
- y4 (type int, value of the axis 'y' of the 4th accelerometer, mounted on the right upper-arm)
- z4 (type int, value of the axis 'z' of the 4th accelerometer, mounted on the right upper-arm)
- class (text, 'sitting-down', 'standing-up', 'standing', 'walking', and 'sitting')

2. Build the Human Activity Classifier

Prepare the data

Before you can use it to train a classification model you must prepare and upload the data:

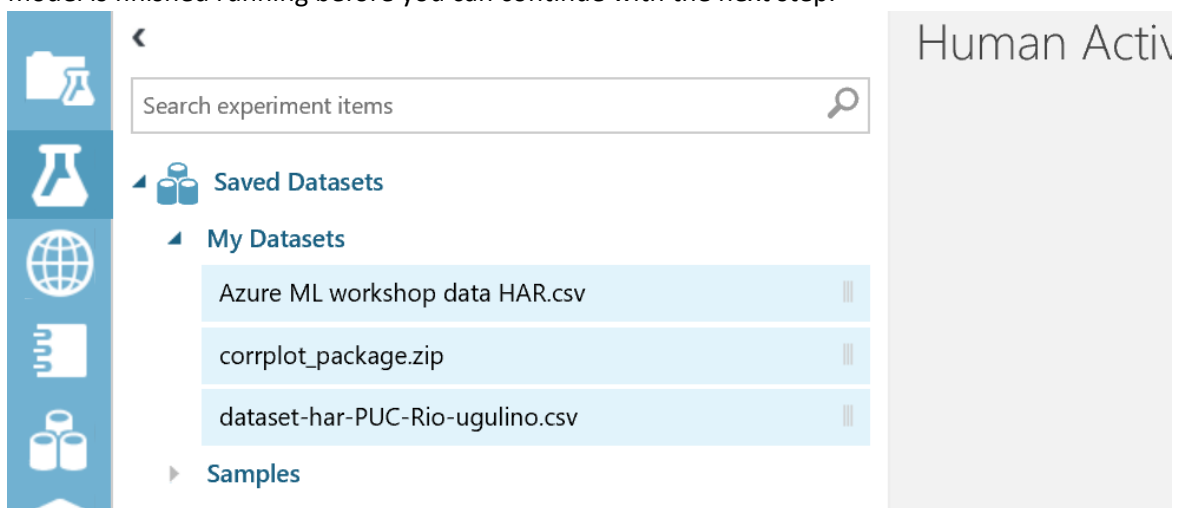
1. Azure ML works with comma separated files. The original data file contains “;” as separator and will therefore be not suitable for uploading. We first have to open the downloaded csv file and convert it to a csv with “,” as a separator. Make sure you also have “.” for your decimals. If you have trouble creating such file, you can start with this starting experiment: <https://gallery.cortanaintelligence.com/Experiment/Human-Activity-Classifier-Step-1-Load-data>. This will open a window where you have to sign in into Azure ML.
2. Open a browser and browse to <https://studio.azureml.net>. Then sign in using the Microsoft account associated with your Azure ML account.
3. Create a new blank experiment by clicking on the + NEW button in the left of your browser, and select EXPERIMENT, and subsequently BLANK EXPERIMENT. You can change the generated name into Human Activity Classifier.



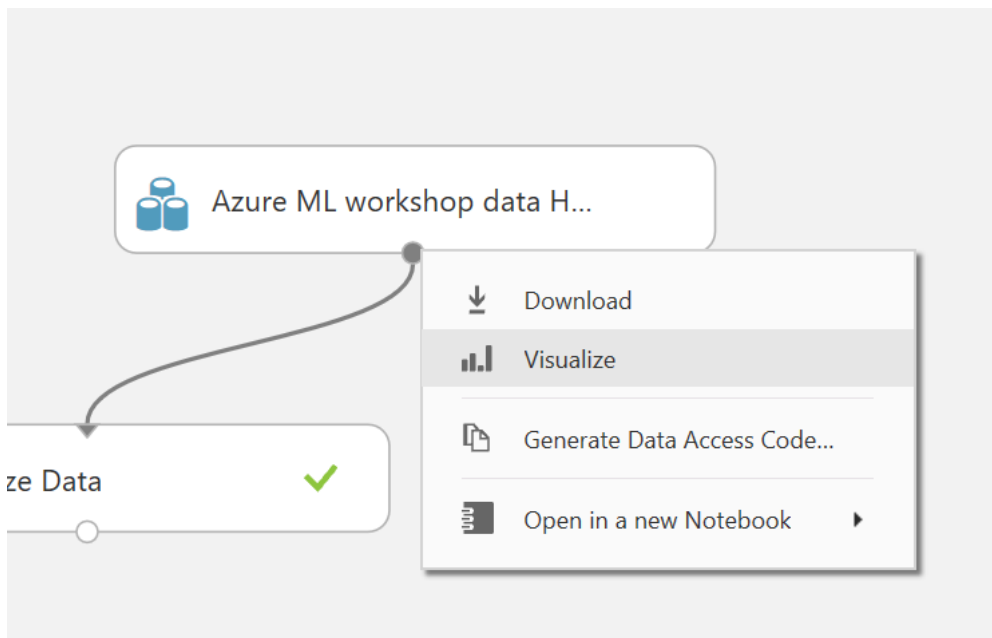
Upload the csv file to Azure ML and name it **HAR dataset**. To do this, you have to click on the + **NEW** button in the left lower corner of your browser, and select **DATASET**, and subsequently **FROM LOCAL FILE**.



4. In the **Human Activity Classifier** experiment, go to **My Datasets** under **Saved datasets**, and drag the **HAR dataset** on the canvas, and click on **RUN** (menu below). You will have to wait until the model is finished running before you can continue with the next step.



5. To visualize the output of the dataset, **right-click** on the output port of the data module and select **Visualize**.



Now you can review the data it contains. Note that the dataset contains the following variables:

- user (string)
- gender (string)
- age (numeric)
- how_tall_in_meters (numeric)
- weight (numeric)
- body_mass_index (numeric)
- x1 (numeric)
- y1 (numeric)
- z1 (numeric)
- x2 (numeric)
- y2 (numeric)
- z2 numeric)
- x3 (numeric)
- y3 (numeric)
- z3 (numeric)
- x4 (numeric)
- y4 (numeric)
- z4 (string) !!!
- class (string)

6. Ups, something went wrong: “z4” has been processed as a “string” instead of an “integer”. You can change this by using a few lines of R with the **Execute R Script** module. Drag the **Execute R Script** module on the canvas and use this code to convert “z4” to a numeric:

```
# Map 1-based optional input ports to variables
df <- maml.mapInputPort(1) # class: data.frame
```

```
df$z4 <- as.numeric(df$z4)
```

```
# Select data.frame to be sent to the output Dataset port
```

```
maml.mapOutputPort("df");
```

You might ask yourself: why don't you use the Edit Metadata module for that. Well, if we try that we will get an error that Azure cannot convert specific strings to an integer.

7. After converting “z4” to an integer, we have to inspect the data if we miss any. Therefore, **right-click** on the **Results dataset1** (left) output of the **Execute R Script** module. Although the UCI repository states that there are no missing values, we find that the “z4” column has 1 missing value.
8. We will delete this row with the **Clean Missing Data** Module. Set the properties as follows:
- Columns to be cleaned: all
 - Minimum missing value ratio: 0

- Maximum missing value ratio: 1
 - Cleaning mode: entire row
9. After cleaning the data, we can inspect the data. We start with some descriptive statistics using the **Summarize Data** module.
10. Besides, we can inspect the correlation between the numeric columns using the using the **Select Columns in Dataset** module. Drag this module on the canvas and connect the output port of the **Cleaning Missing Data** module to the input port of the **Select Columns in Dataset** module. Now we have to select the numeric columns, using the WITH RULES, and starting with **NO COLUMNS**, and subsequently select **Include, column types,** **Numeric:**

Select columns

BY NAME

WITH RULES

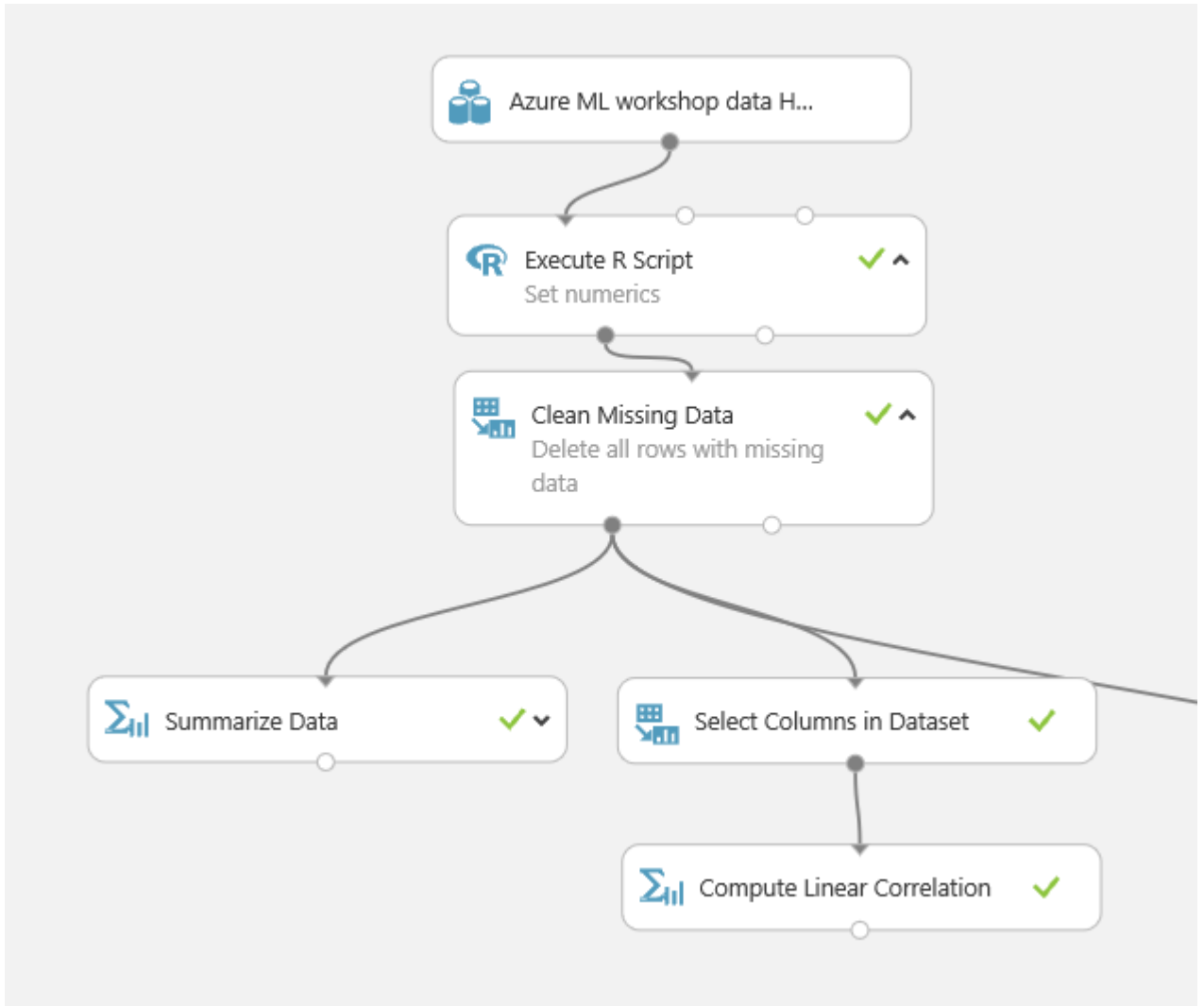
Allow duplicates and preserve column order in selection

Begin With

ALL COLUMNS NO COLUMNS

Include column type Numeric

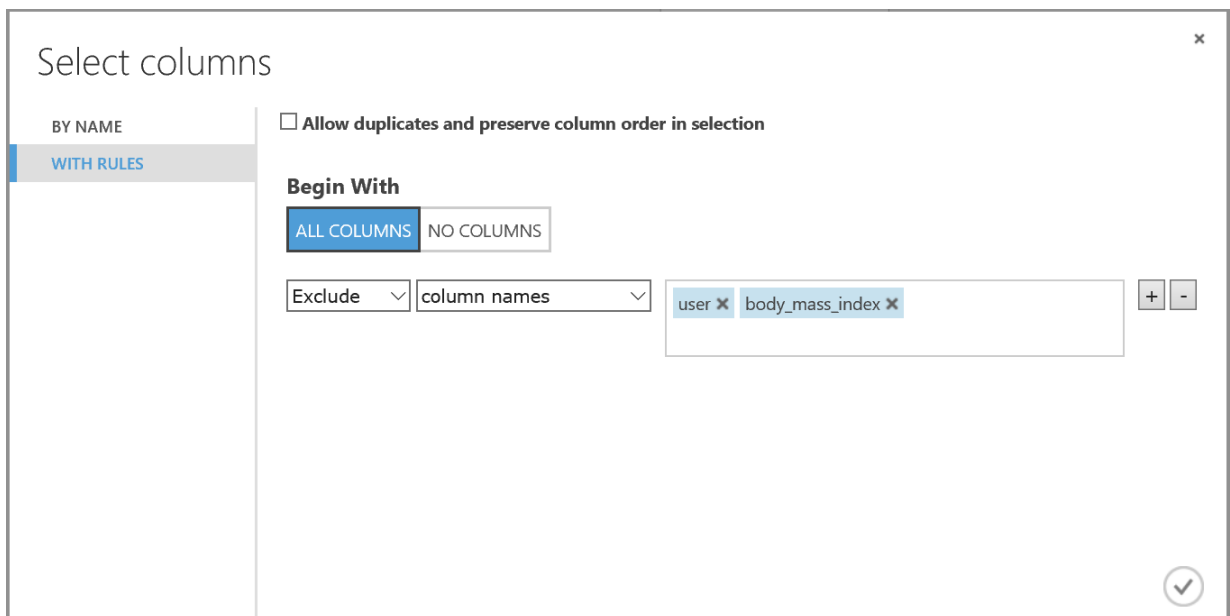
11. Now we can add the Compute Linear Correlation module to calculate the (Pearson's) correlation. Observe that there is a strong correlation between length (how_tall_in_meters), weight (weight) and b.m.i. (body_mass_index). This is not surprising as b.m.i is calculated based on length and weight.



12. Based on prior logic, we will remove 'body_mass_index' using the **Select Columns in Dataset** module. Here we also exclude 'user', as we don't need this identifier later on in our model. Select the **Select Columns in Dataset** module, and in the Properties pane launch the column selector. Then use the column selector to exclude the following columns:

- o user
- o body_mass_index

You can use the With Rules page of the column selector to accomplish this as shown here:



13. Now we transform gender to be a categorical variable by adding an **Edit Metadata** module to the experiment, and connect the **Select Columns** in Dataset output to its input. Set the properties of the **Edit Metadata** module as follows:
 - Column: gender
 - Data type: Unchanged
 - Categorical: Make categorical
 - Fields: Features
 - New column names: Leave blank

14. We will do a likewise transformation with our dependent variable “class”, and set it to a categorical variable and define it as our label. Add an **Edit Metadata** module to the experiment, and connect the Edit Metadata output to its input. Set the properties of the Edit Metadata module as follows:
 - Column: **Edit Metadata** class
 - Data type: Unchanged
 - Categorical: Make categorical
 - Fields: Label
 - New column names: Leave blank

15. When the experiment has finished running, visualize the output of the **Edit Metadata** module and verify that:
 - The columns you specified have been removed.
 - All numeric columns now have a Feature Type of Numeric Feature.
 - All string columns now have a Feature Type of Categorical Feature.

Create and Evaluate a Classification Model

Now that you have prepared the data, you will construct and evaluate a classification model. The goal of this model is to identify a human activity and to find out if somebody is 'sitting-down', 'standing-up', 'standing', 'walking', or 'sitting'.

16. We are now ready to split the data into separate training and test datasets. We will train the model with the training dataset, and test the model with the test dataset. Therefore, add a **Split Data** module to the **Human Activity Classifier** experiment, and connect the output of the **Edit Metadata** module to the input of the **Split Data** module. Set the properties of the **Split Data** module as follows:

- Splitting mode: Split Rows
- Fraction of rows in the first output dataset: 0.7
- Randomized split: Checked
- Random seed: 123
- Stratified split: False

17. Add a **Train Model** module to the experiment, and connect the **Results dataset1** (left) output of the **Split Data** module to the **Dataset** (right) input of the **Train Model** module. In the **Properties** pane for the **Train Model** module, use the column selector to select the **class** column. This sets the label column that the classification model will be trained to predict.

18. Add a **Multiclass Decision Forest** module to the experiment, and connect the output of the **Multiclass Decision Forest** module to the **Untrained model** (left) input of the **Train Model** module. This specifies that the classification model will be trained using the multiclass decision forest algorithm.

19. Set the properties of the **Multiclass Decision Forest** module as follows:

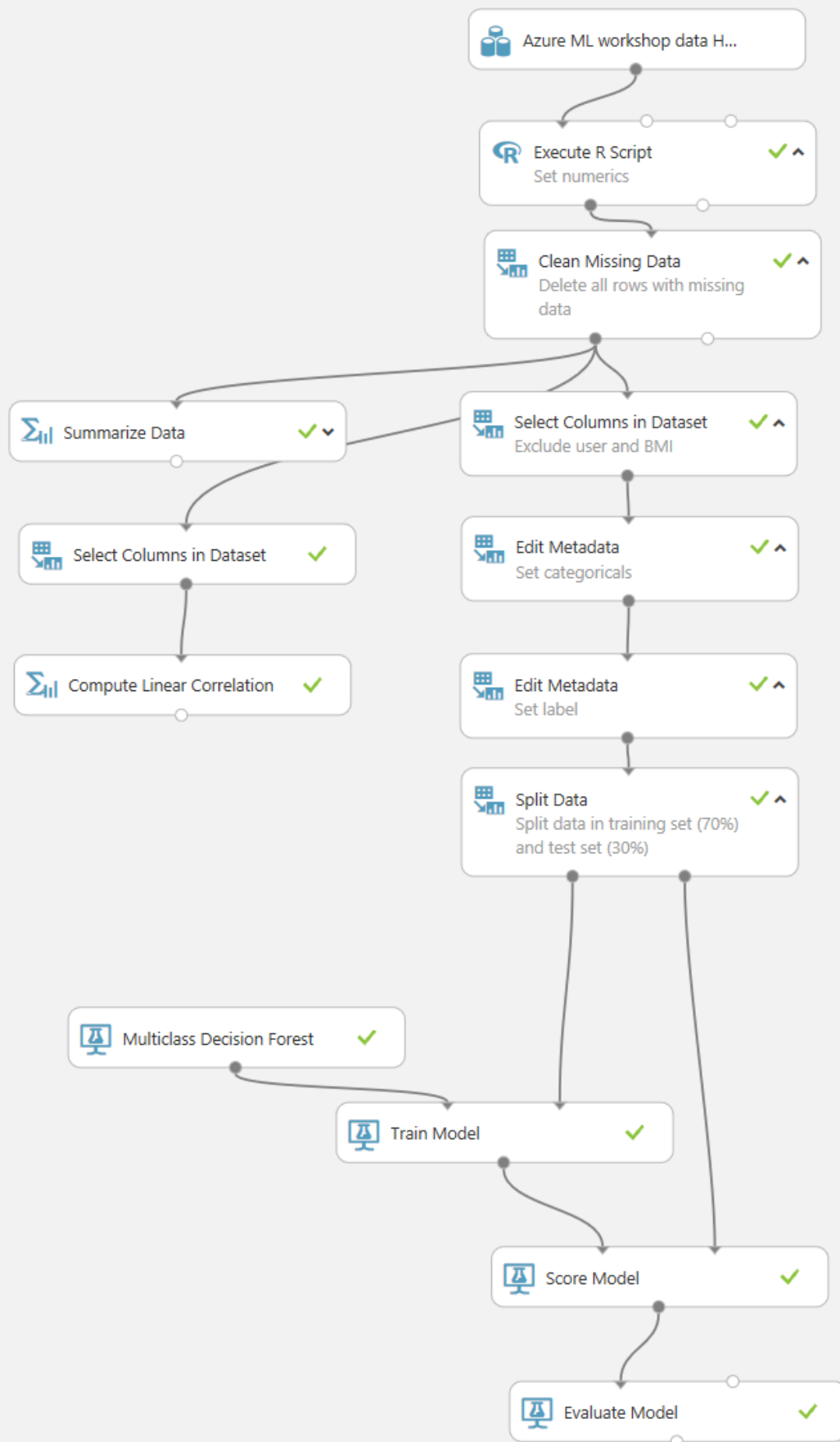
- **Resampling method**: Bagging
- **Create trainer mode**: Single Parameter
- **Number of decision trees**: 8
- **Maximum depth of decision trees**: 32
- **Number of random splits per node**: 128
- **Minimum number of samples per leaf**: 1
- **Allow unknown categorical levels**: Checked

20. Add a **Score Model** module to the experiment. Then connect the output of the **Train Model** module to the **Trained model** (left) input of the **Score Model** module, and connect the **Results dataset2** (right) output of the **Split Data** module to the **Dataset** (right) input of the **Score Model** module.

21. On the **Properties** pane for the **Score Model** module, ensure that the **Append score columns to output** checkbox is selected.

22. Add an **Evaluate Model** module to the experiment, and connect the output of the **Score model** module to the **Scored dataset** (left) input of the **Evaluate Model** module.

23. Verify that your experiment resembles the figure below, then save and run the experiment.

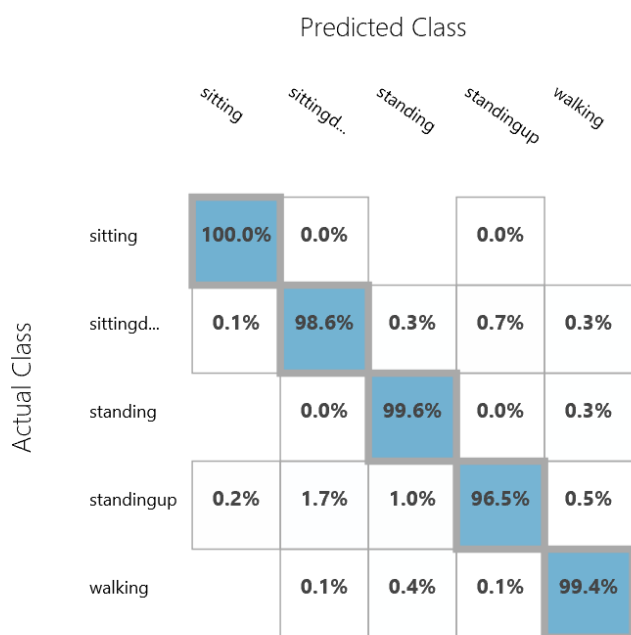


24. When the experiment has finished running, visualize the output of the **Score Model** module, and compare the predicted values in the **Scored Labels** column with the actual values from the test data set in the **class** column.
25. Visualize the output of the **Evaluate Model** module, and review the results (shown below). We see the score per class. Then review the **Overall Accuracy** figure for the model, which should be around **0.994**. This indicates that the classifier model is correct 99% of the time, which is a good figure for an initial model, keeping in mind the original distribution of the classification (see below).

Metrics

Overall accuracy	0.99358
Average accuracy	0.997432
Micro-averaged precision	0.99358
Macro-averaged precision	0.989947
Micro-averaged recall	0.99358
Macro-averaged recall	0.988234

Confusion Matrix



Detailed Accuracy from the original paper

Correctly Classified Instances	164662	99.4144 %
Incorrectly Classified Instances	970	0.5856 %
Root mean squared error	0.0463	
Relative absolute error	0.7938	%
Relative absolute error	0.7938	%

Detailed Accuracy by Class

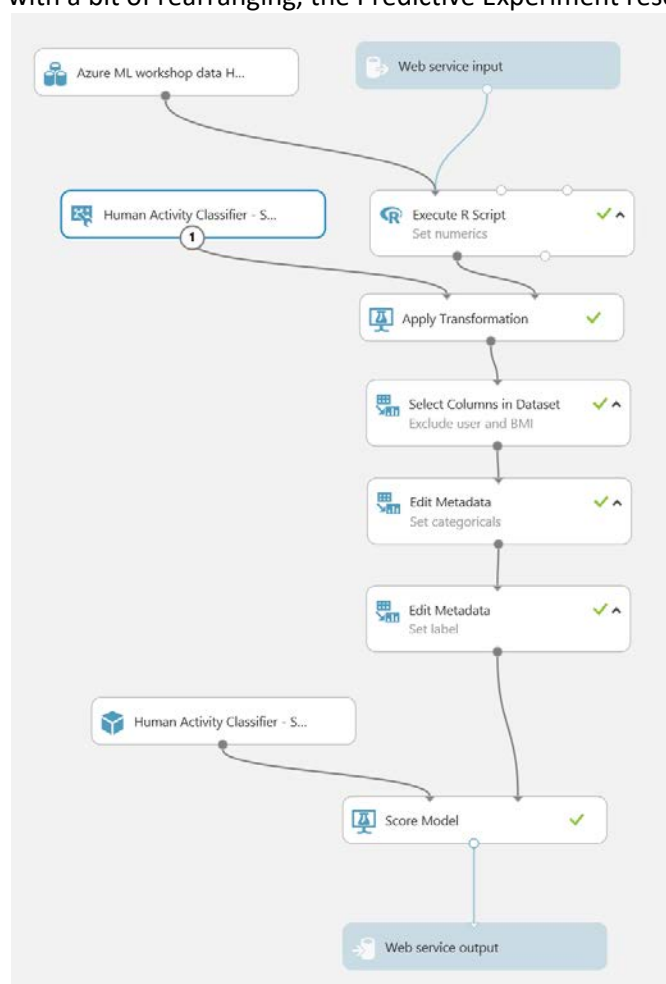
TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.999	0	1	0.999	0.999	1	Sitting
0.971	0.002	0.969	0.971	0.970	0.999	Sitting down
0.999	0.001	0.998	0.999	0.999	1	Standing
0.962	0.003	0.969	0.962	0.965	0.999	Standing up
0.998	0.001	0.998	0.998	0.998	1	Walking
0.994	0.001	0.994	0.994	0.994	1	Weighted Avg.

3. Publish your Human Activity Classifier

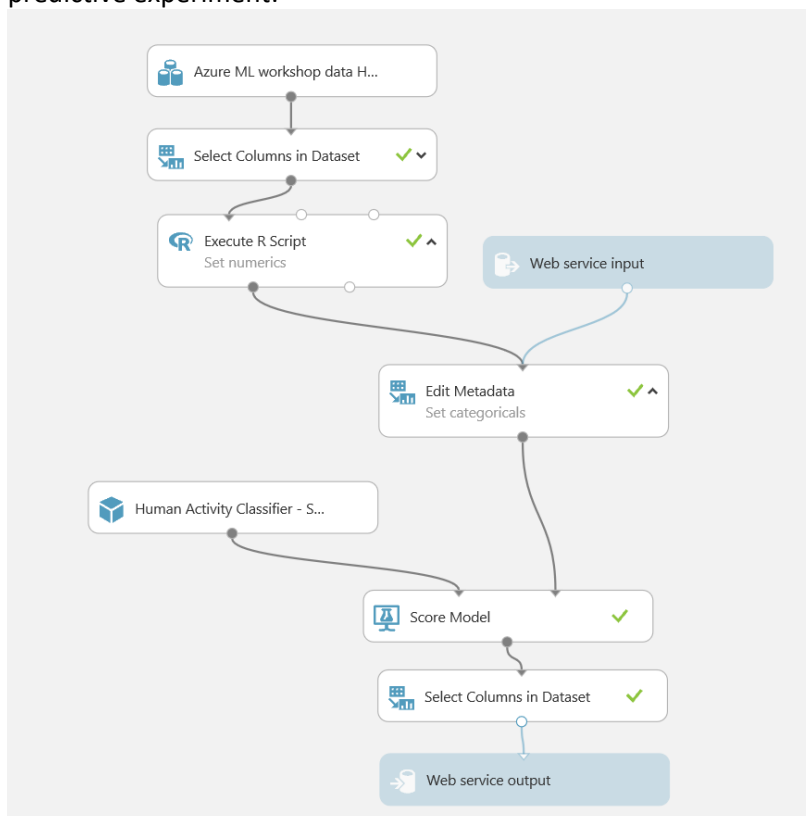
Publish the Model as a Web Service

26. Make sure you have saved and ran the experiment. With the **Human Activity Classifier** experiment open, click the **SET UP WEB SERVICE** icon at the bottom of the Azure ML Studio page and click **Predictive Web Service [Recommended]**. A new Predictive Experiment tab will be automatically created.

27. Verify that, with a bit of rearranging, the Predictive Experiment resembles this figure:



28. We can now start to remove variables we don't need for prediction. Besides eliminating "user", and "bmi" we can now also remove "class", as we want that as output from the model. Therefore, you can drag the **Select Columns in Dataset** module up, add "class" to be removed, and connect it to the original dataset and the output to the **Execute R Script** module.
29. Besides, we will make sure to use a numeric value for "z4", so we can move the **Webservice input** and connect it directly to the **Edit Metadata** module where we make "gender" categorical.
30. For this experiment, we will also make sure to send complete records, so we remove the **Clean Missing Data** module.
31. Delete the connection between the **Score Model** module and the **Web service output** module.
32. Add a **Select Columns in Dataset** module to the experiment, and connect the output of the **Score Model** module to its input. Then connect the output of the **Select Columns in Dataset** module to the input of the **Web service output** module.
33. Select the **Select Columns in Dataset** module, and use the column selector to select only the **Scored Labels** column. This ensures that when the web service is called, only the predicted value is returned.
34. Ensure that the predictive experiment now looks like the following, and then save and run the predictive experiment:



35. When the experiment has finished running, visualize the output of the **last Select Columns in**

Dataset module and verify that only the **Scored Labels** column is returned.

Deploy and Use the Web Service

36. In the **Human Activity Classifier [Predictive Exp.]** experiment, click the **Deploy Web Service** icon at the bottom of the Azure ML Studio window.

37. Wait a few seconds for the dashboard page to appear, and note the **API key** and **Request/Response** link. You will use these to connect to the web service from a client application.

human activity classifier - step 3: publish the model [predictive exp.]

The screenshot shows the 'New Web Services Experience' dashboard. At the top, there are tabs for 'DASHBOARD' and 'CONFIGURATION'. Below that, there are links for 'General' and 'New Web Services Experience preview'. The 'Published experiment' section includes 'View snapshot' and 'View latest'. The 'Description' section states 'No description provided for this web service.' The 'API key' is displayed in a text box: 'GEJlm6jdl6c3JLch2BqHTI339EHyqh8ifMMYXyCCNdei4oTa/lkwAXF+zNi2WKeZ+KJxaeH08mNc3IOWRaT1sA=='. Below this, there are links for 'API HELP PAGE', 'TEST', and 'APPS'. The 'TEST' section has a 'Test' button and a 'Test preview' link. The 'APPS' section has links for 'Excel 2013 or later' and 'Excel 2010 or earlier workbook'. The 'REQUEST/RESPONSE' section has a 'Test' button and a 'Test preview' link. The 'BATCH EXECUTION' section has a 'Test preview' link.

38. You have several options to connect to the webservice. To test this webservice, you can click on **New Web Services Experience (preview)**. This will open a new browser.

39. Here you have the option to test your model (**Test endpoint** option under BASICS):

The screenshot shows the 'Human Activity Classifier - Step 3: Publish the model [Predictive Exp.]' dashboard. The title is 'default'. There is a 'View in Studio' button. The dashboard is divided into three main sections: 'BASICS', 'MANAGE & MONITOR', and 'DEVELOP'. The 'BASICS' section has an icon of three nodes connected by lines and a list of links: 'Test endpoint', 'Configure endpoint', 'Use endpoint', and 'Launch in Excel'. The 'MANAGE & MONITOR' section has an icon of a bar chart and a link: 'Tutorial: Retrain web service'. The 'DEVELOP' section has an icon of code tags and a list of links: 'Tutorial: How to build apps', 'Tutorial: Debug web service', and 'Tutorial: Best practices'.

40. When clicking on Test endpoint, you have the option to enable the usage of sample data, which will generate a sample record to test your model with:

Request-Response Batch

Sample Data ✕

Sample Data is a feature for your web service users to get started with using your web service. Sample data will make a small sample from your training data set available, so we can populate this test dialog. Do you want to enable it?

[Enable](#)

41. After enabling this sample data, you will see the generated sample data:

input1
output1

gender

age

how_tall_in_meters

weight

x1

y1

z1

x2

y2

z2

x3

y3

z3

x4

y4

z4

Your prediction results will display here.

[Test Request-Response](#)

42. The final step would be pressing the **Test Request-Response** button: what kind of activity is this woman doing according to your model?

43. Another option is to click on the blue TEST button.

Default Endpoint

API HELP PAGE
TEST

REQUEST/RESPONSE
Test

44. This will open a pop-up window, where you can fill out some test values:

Test Human Activity Classifier - Step 3: Publish the model [Predictive Exp.] Service

Enter data to predict

GENDER

AGE

HOW_TALL_IN_METERS

WEIGHT

X1

✓

45. The last option is to open an Excel file, which will automatically create sample data. Opening this file will add the Azure Machine Learning add-in to the workbook. If that doesn't work, or you don't have Excel on your laptop, you could follow the next steps to make a workbook online:
46. Open a new browser tab.
47. In the new browser tab, navigate to <https://office.live.com/start/Excel.aspx>. If prompted, sign in with your Microsoft account (use the same credentials you use to access Azure MLStudio.)
48. In Excel Online, create a new blank workbook.
49. On the **Insert** tab, click **Office Add-ins**. Then in the **Office Add-ins** dialog box, select **Store**, search for *Azure Machine Learning*, and add the **Azure Machine Learning** add-in as shown below:



Office Add-ins

MY ADD-INS | **STORE**

Add-ins may access personal information. By turning an add-in on, you agree to its License Terms and Privacy Policy.

Machine Learning

Suggested for you

Category		
All		Azure Machine Learning Use Azure ML web services ★★★★☆
CRM		<input type="button" value="Add"/>
Data Analytics		
Visualization		ReportWorX 365 ReportWorX 365 delivers insightful reports from manufacturing and building energy data. May require additional purchase ★★★★★
Document Review		<input type="button" value="Add"/>
Editor's Picks		
Education		
Financial Management		

50. After the add-in is installed, in the **Azure Machine Learning** pane on the right of the Excel workbook, click **Add Web Service**. Boxes for the URL and API key of the web service will appear.
51. On the browser tab containing the dashboard page for your Azure ML web service, right-click the **Request/Response** link you noted earlier and copy the web service URL to the clipboard. Then return to the browser tab containing the Excel Online workbook and paste the URL into the URL box.
52. On the browser tab containing the dashboard page for your Azure ML web service, click the **Copy** button for the **API key** you noted earlier to copy the key to the clipboard. Then return to the browser tab containing the Excel Online workbook and paste it into the **API key** box.
53. Verify that the **Azure Machine Learning** pane in your workbook now resembles this, and click **Add**:

The screenshot shows the Azure Machine Learning pane in Excel. It has a title bar 'Azure Machine Learning' and a sub-header 'Web Services'. Below this, there are two service entries, each with a globe icon and a truncated name. Below the services, there are two input fields: 'URL' and 'API key', both with a question mark icon. The URL field contains a long URL: 'https://studio.azureml.net/apihelp/workspaces/b2101c3182ae42c58c2466ab40607479/webservice/s/4d98abd657a449a895374b16fa2722aa/endpoints/66ce37974550469ba9b9c3d90eb25814/score'. The API key field contains a long alphanumeric string: 'Tv7LH8SUFERGOZT0Tched/xBDGb+V9/QHgMP.Tcd/rDH57OWrMLjgdPmgLF3+rO1/pUe9vXevJlU7dHjU8TPyg=='. At the bottom of the pane, there are 'Cancel' and 'Add' buttons. Below the pane, there is an 'Auto-predict' checkbox (unchecked) and a 'Predict All' button.

54. After the web service has been added, in the **Azure Machine Learning** pane, it is opened on 2. Predict. Here you have the option to generate sample data by clicking on **Use sample data**. This enters some sample input values in the worksheet.
55. Select the cells containing the input data (cells A1 to P6), and in the **Azure Machine Learning** pane, click the button to select the input range and confirm that it is '**Sheet1**!A1:P6
56. Ensure that the **My data has headers** box is checked.
57. In the **Output** box type **Q1**, and ensure the **Include headers** box is checked.
58. Click the **Predict** button, and after a few seconds, view the predicted label in cell Q2.

gender	age	bmi	weight	x1	y1	z1	x2	y2	z2	x3	y3	z3	x4	y4	z4	Scored Labels
Woman	46	1.62	75	-3	92	-63	-23	18	-19	5	104	-92	-150	-103	-147	sitting
Woman	46	1.62	75	-3	94	-64	-21	18	-18	-14	104	-90	-149	-104	-145	sitting
Woman	46	1.62	75	-1	97	-61	-12	20	-15	-13	104	-90	-151	-104	-144	sitting
Woman	46	1.62	75	-2	96	-57	15	21	-16	-13	104	-89	-153	-103	-142	sitting
Woman	46	1.62	75	-1	96	-61	-13	20	-15	-13	104	-89	-153	-104	-143	sitting

59. Change some values of row 2 and click **Predict** again. Then view the updated label that is predicted by the web service.
60. Try changing a few of the input variables and predicting the human activity class. You can add multiple rows to the input range and try various combinations at once.

Summary

By completing this lab, you have prepared your environment and data, and built and deployed your own Azure ML model. We hope you enjoyed this introductory lab and that you will build many more machine learning solutions!