Guided project for Information Retrieval:

Classifying Tweets Based on sentiment

# Overview

In this lab, you will use Azure Machine Learning studio to work with text data. Specifically, you will use code R to clean text, remove stop words, and apply Porter stemming to the remaining words. You will then build a machine learning model to predict tweet class as positive or negative class. Finally, you will create an Azure ML web service to which will take free text and predict its label and score based on sentiment analysis.

# What You’ll Need

To complete this lab, you will need the following:

* An Azure ML account
* The files for this lab

To perform the setup tasks, you will need the following:

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

**Note**: To set up the required environment for the lab, follow below instructions.

# Create an Azure ML Account

Azure ML offers a free-tier account, which you can use to complete the tasks in this Assignment.

**Note**: A free Azure ML workspace is not the same as a Microsoft Azure trial subscription!

## Sign Up for a Microsoft Account and a Free Azure ML Workspace

1. If you do not already have a Microsoft account, sign up for one at <https://signup.live.com/>.
2. If you do not already have an Azure ML workspace, browse to <https://aka.ms/edx-dat203.3x-aml> and click **Get Started Now**. Then follow the instructions to sign up for a free Azure ML workspace. If prompted, sign in with your Microsoft account credentials.

**Note**: Your free-tier Azure ML workspace allows you unlimited access, with some reduced capabilities compared to a full Microsoft Azure subscription. Your experiments will only run at low priority on a single processor core. As a result, you will experience some longer wait times. However, you have full access to all features of Azure ML.

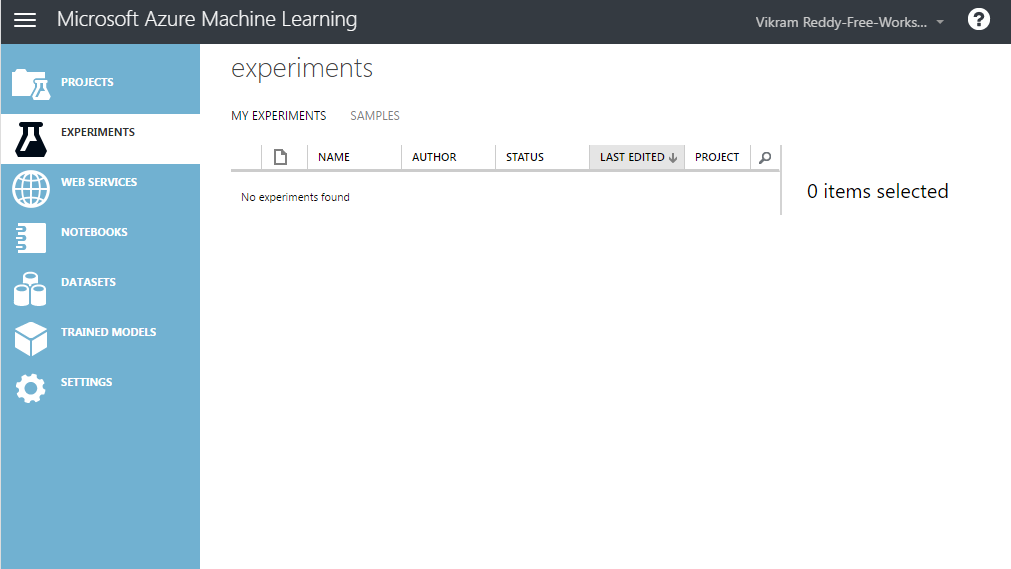
# Creating an Azure ML Experiment

## Azure ML enables you to create experiments in which you can manipulate data, create predictive models, and visualize the results.

## Sign into Azure ML Studio

Browse to <https://studio.azureml.net> and sign in using the Microsoft account associated with your free Azure ML account.

You should now be in Azure ML Studio with the **Experiments** page selected, which looks like the following image (if not, click the **Studio** tab at the top of the page).

Now you are ready to work in Azure Machine Learning studio by following below step by step guidelines:

## Create an Experiment

1. In the studio, at the bottom left, click **NEW**. Then in the **Experiment** category, in the collection of Microsoft samples, select **Blank Experiment**. This creates a blank experiment, which looks similar to the following image.

# 

# Change the title of your experiment from “Experiment created on today’s date” to “Tweet Sentiment”

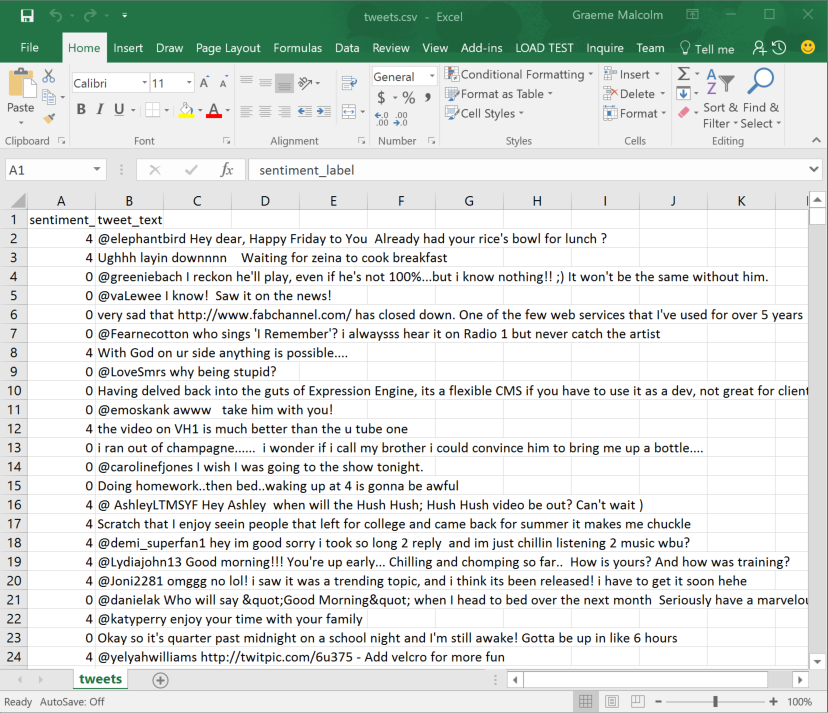
# Exploring and Uploading the Dataset

# Exploring Text Data

The text data used in this assignment consists of a collection of tweets that have been categorized as positive or negative.

## Explore the Tweets Dataset

1. In the folder where you got this assignment, open the **tweets.csv** file, using either a spreadsheet application such as Microsoft Excel, or a text editor such as Microsoft Windows Notepad.
2. View the contents of the **tweets.csv** file, noting that it contains tweets with a numeric indication of sentiment in which 4 indicates a positive tweet, and 0 indicates a negative tweet:



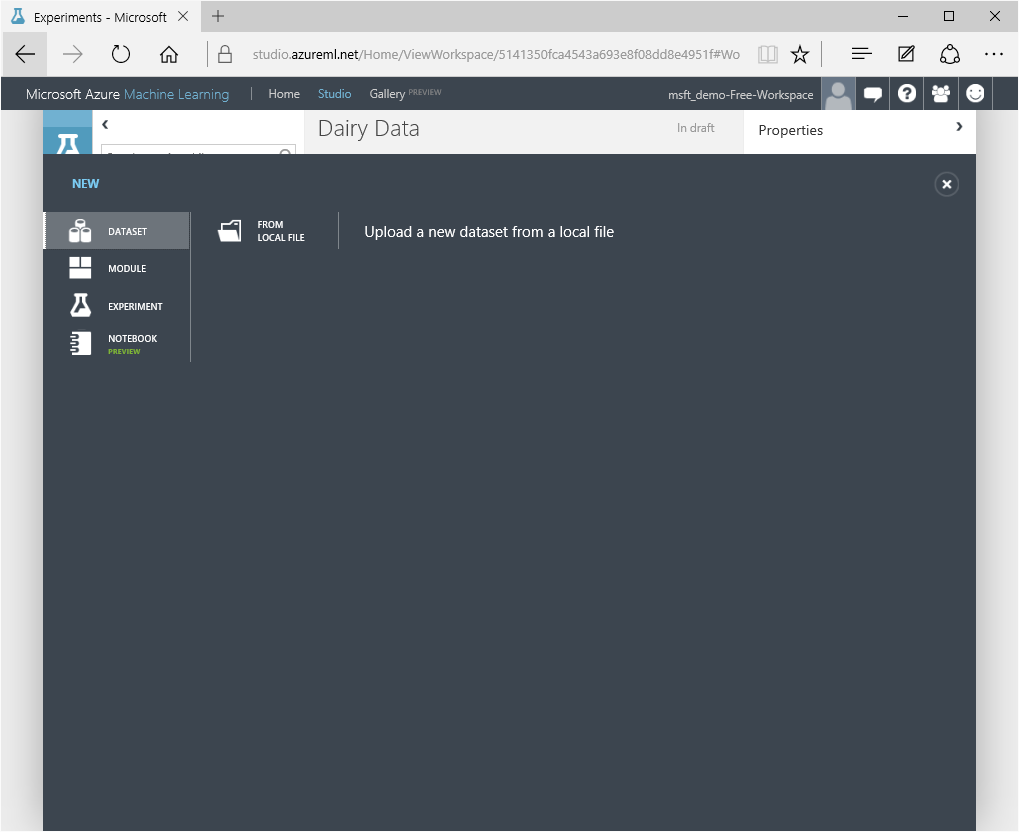
1. Close the data file without saving any changes.

## Explore the Stopwords Dataset

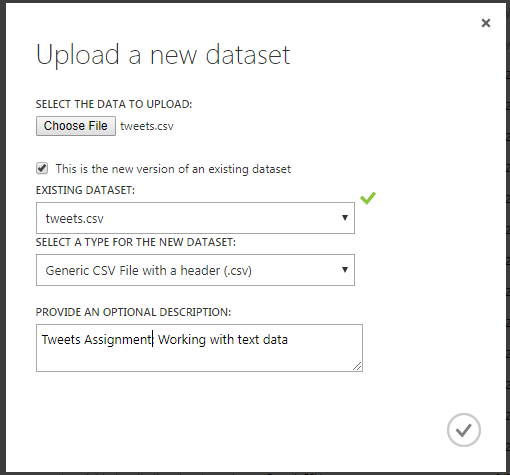
1. Open the **stopwords.csv** file and review its contents. Note that this file contains a list of common words such as “a”, “the”, “it”, and so on, which are generally not helpful in determining the meaning or sentiment of a sentence or paragraph.
2. Close the file without saving any changes.

## Upload the Datasets to Azure Machine Learning

1. Return to your browser where your experiment is displayed. At the bottom left, click **NEW**. Then in the **NEW** dialog box, click the **DATASET** tab as shown in the following image.



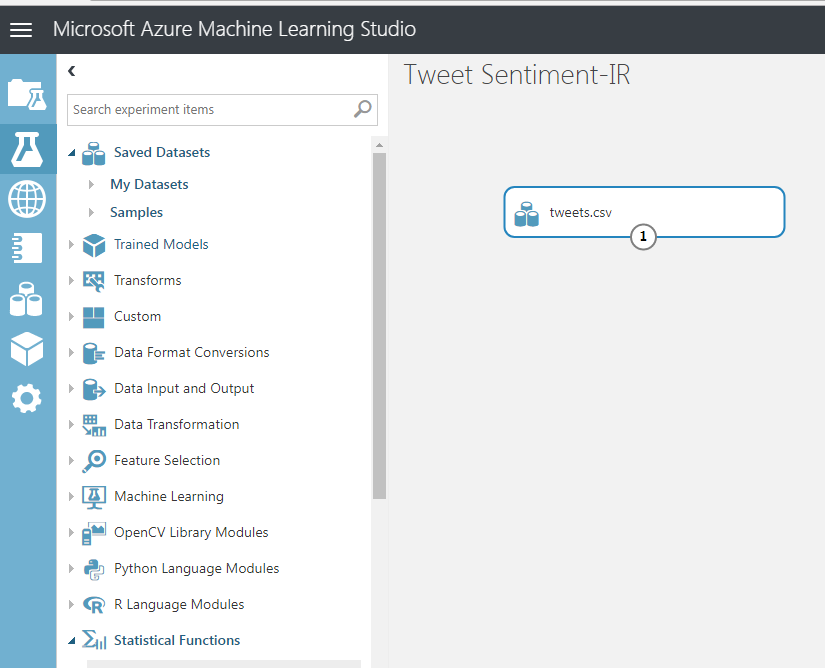
1. In the **DATASET** tab, at the bottom of page left click **NEW**; click **FROM LOCAL FILE**.
2. Then in the **Upload a new dataset** dialog box, browse to select the **tweets.csv** file from the folder where you extracted the lab files on your local computer.
3. Enter the following details, and then click the **✓**icon.
   * **This is a new version of an existing dataset**: Unselected
   * **Enter a name for the new dataset**: tweets.csv
   * **Select a type for the new dataset**: Generic CSV file with a header (.csv)
   * **Provide an optional description**: Tweets.



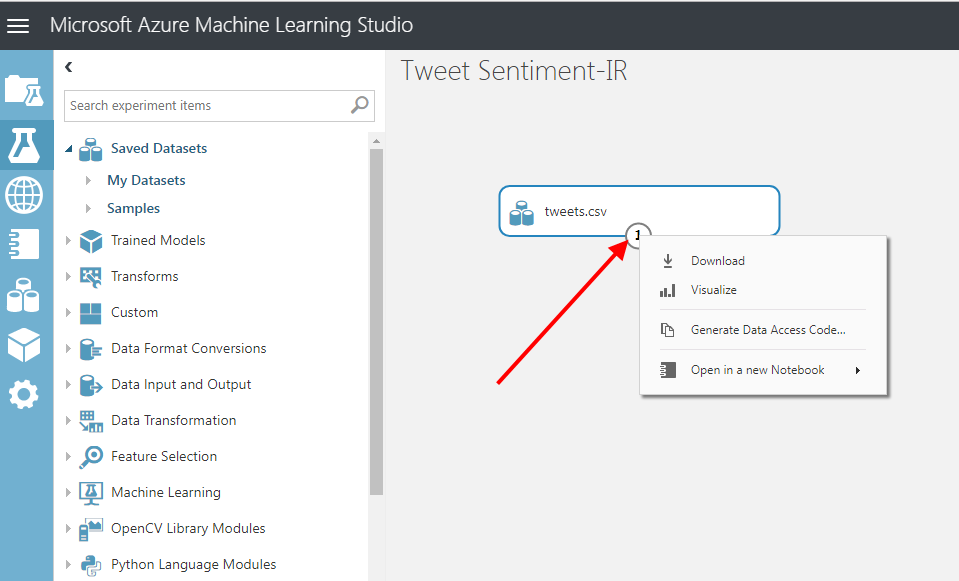
1. Wait for the upload of the dataset to complete, then click **OK** on the status bar at the bottom of the Azure ML Studio page.
2. Repeat the previous steps to upload the stopwords.csv file as a new dataset with the following properties:
   * **This is a new version of an existing dataset**: Unselected
   * **Enter a name for the new dataset**: stopwords.csv
   * **Select a type for the new dataset**: Generic CSV file with a header (.csv)
   * **Provide an optional description**: Stopwords.
3. On the experiment items pane, expand **Saved Datasets** > **My Datasets** to verify that the **Tweets** and **stopwords** dataset is listed.

## Create an Azure ML Experiment

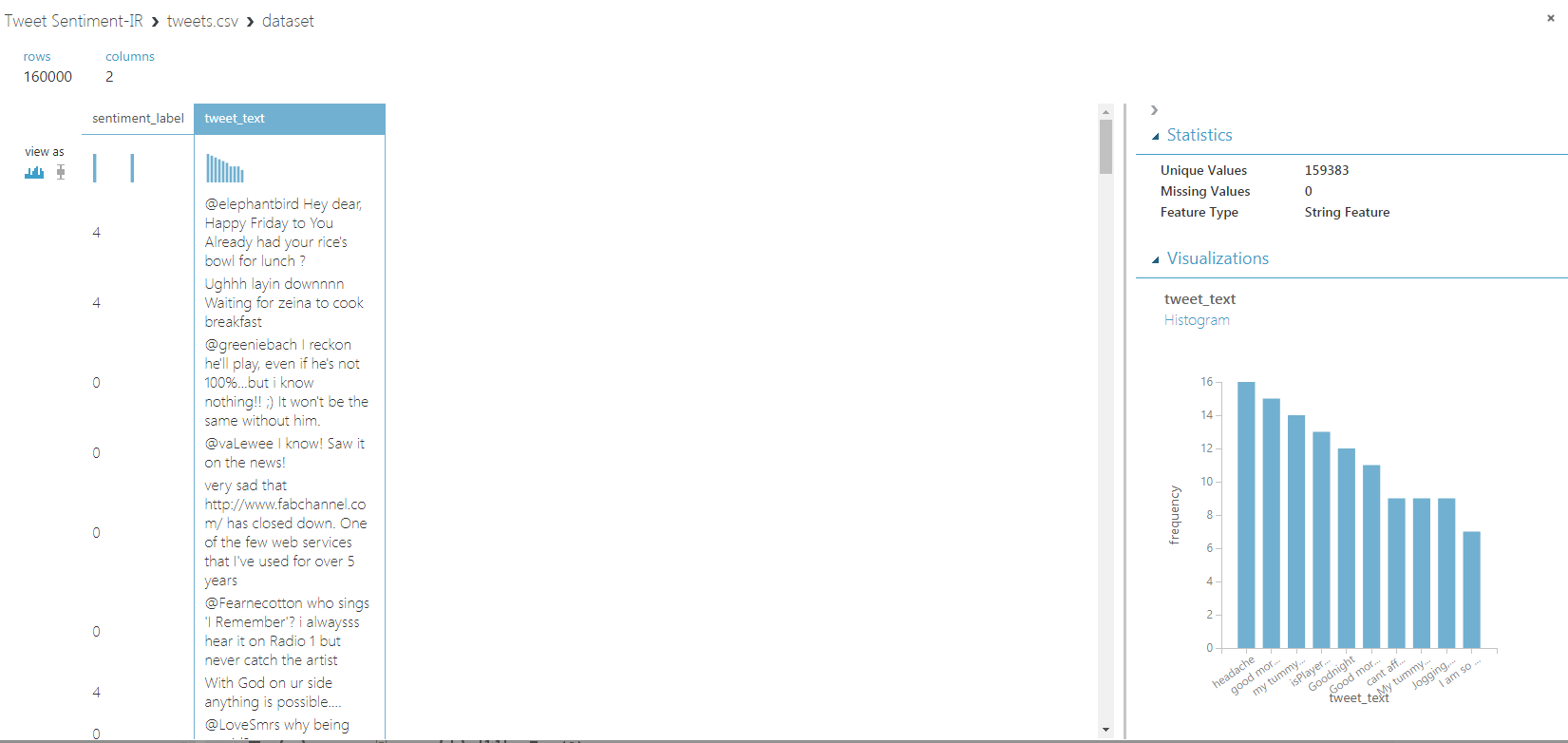
1. Drag the **Tweet** dataset to the canvas for the **Tweet Sentiment** experiment.
2. Verify that the Azure ML screen, which shows your experiment, now looks like the figure shown here:



1. Right Click the output port for the **tweet** dataset on the canvas and click **Visualize** to view the data in the dataset as shown in the figure:



1. Click on the second column labeled **tweet\_text**, which will display some properties of that feature (data column) on the right side of the display. These properties include summary statistics and the data type, as shown here:



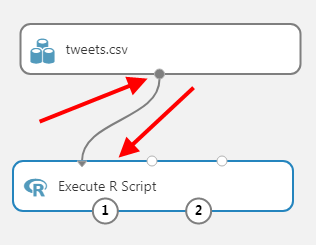
1. Verify that the dataset contains the data you viewed in the source file.

# Using Custom Code in Azure ML to Preprocess Text Data

Note that the experiment contains a number of **Execute R Script** modules to prepare the text data by removing stopwords and stemming the remaining words. Review the code in these modules.

1. Search for the **Execute R Script** module, which is under the **R Language Modules**, and drag it onto the canvas.
2. Connect the **Results Dataset** output of the **tweet Dataset** module to the **Dataset1** (left most) input of the **Execute R Script** module as shown here:

Note: To connect two ports, left click on dataset port of tweet dataset, drag and leave the mouse at leftmost port of execute R script



1. Select the **Execute R Script** module, set the **R Version** to the latest available version of **Microsoft R Open**, and then replace the existing R code in the code editor window of the **Execute R Script** module with the following code.

dataset <- maml.mapInputPort(1)

library(tm) ## Text mining library

## Set the comlumn names

colnames(dataset) <- c("sentiment", "tweets")

## Extract text data and coerce the vector to a tm corpus

tweet.text <- Corpus(VectorSource(dataset['tweets']))

## Apply transformations to the corpus

tweet.text <- tm\_map(tweet.text, content\_transformer(removeNumbers))

tweet.text <- tm\_map(tweet.text, content\_transformer(removePunctuation))

tweet.text <- tm\_map(tweet.text, content\_transformer(stripWhitespace))

tweet.text <- tm\_map(tweet.text, content\_transformer(tolower))

## Transform the processed corpus back to a vector of

## character strings in a dataframe

tweet\_content <- unlist(sapply(tweet.text, '[', "content"))

outframe <- data.frame(tweets = enc2utf8(tweet\_content),

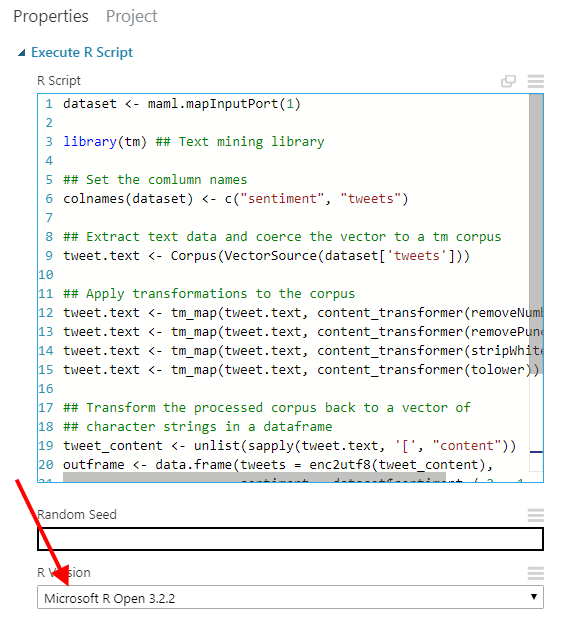
sentiment = dataset$sentiment / 2 - 1,

stringsAsFactors = F,

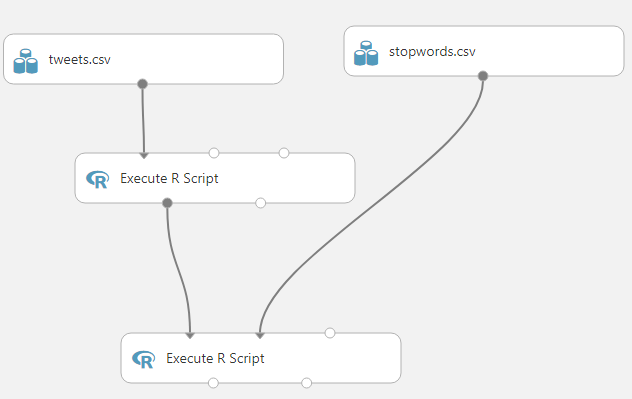
row.names = NULL)

## Output the result

maml.mapOutputPort("outframe")



1. Add second **Execute R Script** module, and connect the **Results Dataset** output of the first **Execute R Script** module to the **Dataset1** (left most) input of the second **Execute R Script** module
2. Drag **stopword.csv** from **Saved Datasets** > **My Datasets**
3. Connect the **Results Dataset** output of the **stopsords.csv** module to the **Dataset2** (middle) input of the second **Execute R Script** module as shown here:



1. Select the **Execute R Script** module, set the **R Version** to the latest available version of **Microsoft R Open**, and then replace the existing R code in the code editor window of the **Execute R Script** module with the following code.

dataset <- maml.mapInputPort(1)

stop.words <- maml.mapInputPort(2)

library(tm) ## Text mining library

## Extract text data and coerce the vector to a tm corpus

tweet.text <- Corpus(VectorSource(dataset['tweets']))

## Remove the stopwords

stop.words['words'] <- unique(stop.words['words'])

tweet.text <- tm\_map(tweet.text, removeWords, stop.words[, 'words'])

## Transform the processed corpus back to a vector of

## character strings in a dataframe

dataset['tweets'] <- data.frame(text = enc2utf8(unlist(sapply(tweet.text, `[`, "content"))),

stringsAsFactors=F)

## Output the result

maml.mapOutputPort("dataset")

1. Add third **Execute R Script** module, and connect the **Results Dataset** output of the second **Execute R Script** module to the **Dataset1** (left most) input of the third **Execute R Script** module
2. Select the **Execute R Script** module, set the **R Version** to the latest available version of **Microsoft R Open**, and then replace the existing R code in the code editor window of the **Execute R Script** module with the following code.

dataset <- maml.mapInputPort(1)

library(tm) ## Text mining library

library(SnowballC) ## For stemming words

## Extract text data and create a tm corpus

tweet.text <- Corpus(VectorSource(dataset['tweets']))

## Stem the words in the tweets

tweet.text <- tm\_map(tweet.text, stemDocument)

## Transform the processed corpus back to a vector of

## character strings in a dataframe

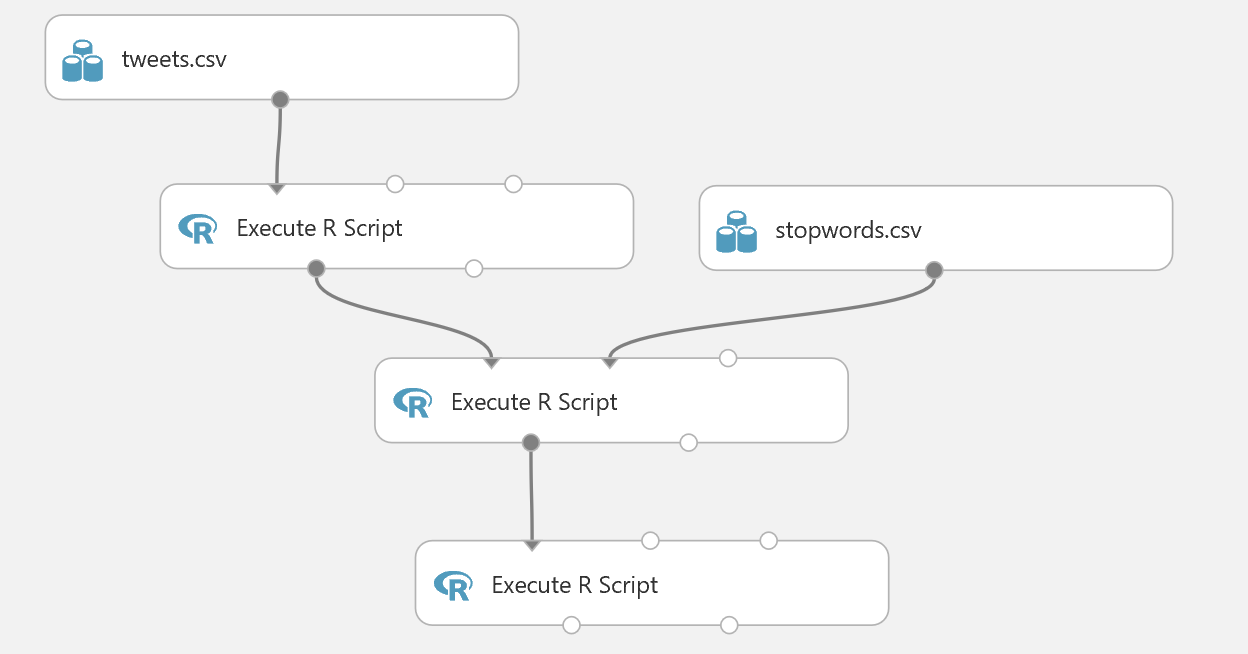
dataset['tweets'] <- data.frame(text = enc2utf8(unlist(sapply(tweet.text, `[`, "content"))),

stringsAsFactors=F)

## Output the result

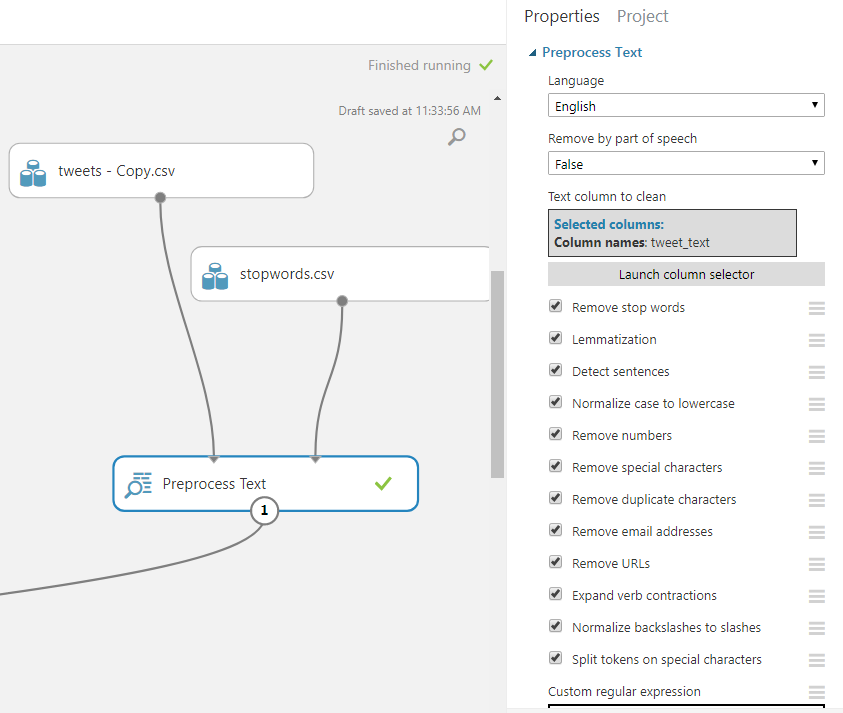
maml.mapOutputPort("dataset")

1. The experiment with three **Execute R Script module** should look like this:



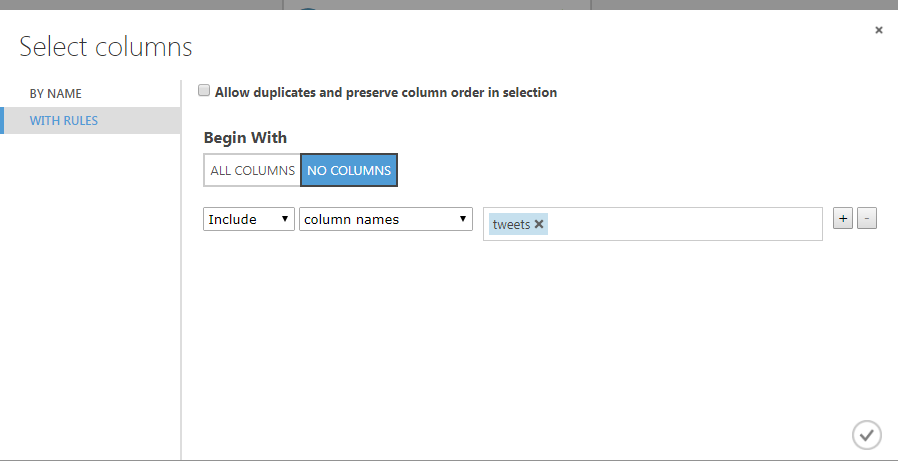
1. Note that the experiment contains a number of **Execute R Script** modules to prepare the text data by removing stopwords and stemming the remaining words. Review the code in these modules.
2. Save and run the experiment, and visualize the **Results Dataset** (left) output of the final **Execute R Script** module to see the cleaned text.

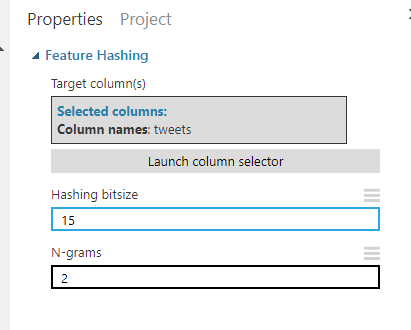
**NOTE:** you can use **Preprocess Text Module** instead of using R code to preprocess data.Add it in experiment and observe parameter in property pane, similar to following image:



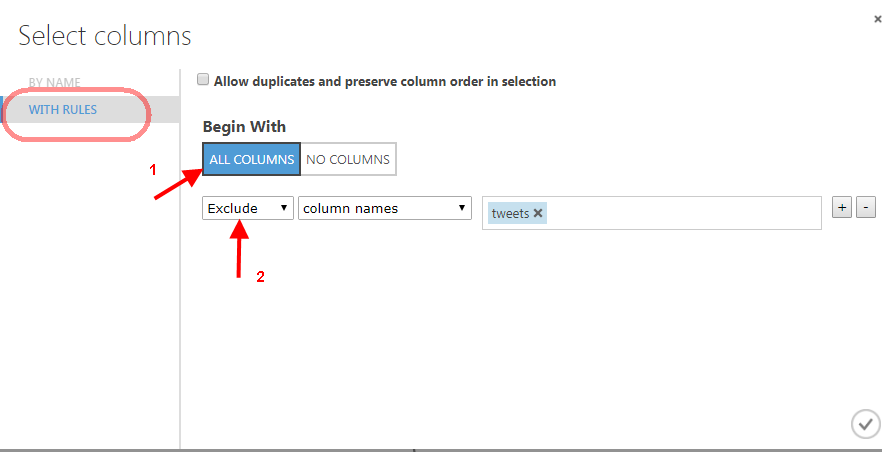
## Create Features for Classification

1. Add **Feature Hashing** module to the experiment and connect the **Results Dataset** (left) output of the final **Execute R Script** module to its input.
2. On the properties pane of the **Feature Hashing** module set the parameters as follows:
3. On the properties with the **Feature Hashing** module selected, click **Launch Column** selector.
   * **Target column(s**): tweets
   * **Hashing bitsize:** 15
   * **N-grams:** 2





1. Save and Run the experiment.
2. Visualize the output of the **Feature Hashing** module, noting that there are now around 33,000 features. The hash has compressed the approximately 135,000 unique words into a smaller number of features.
3. Add a **Select Columns in Dataset** module to the experiment and connect the output of the **Feature Hashing** module to its input. Then configure the **Select Columns in Dataset** module to exclude the **tweets** column, which is not required for classification now that you have generated features (note that the column selector might take some time to open due to the large number of columns generated by feature hashing).
4. On the properties pane of the **Select Columns in Dataset** module click **Launch column selector**



1. Save and run the experiment. Then visualize the output of the **Select Columns in Dataset** module and verify that the tweets column is no longer included.

## Train and Evaluate a Classifier Model

You have created a feature set for classifying the sentiment of the tweets. Perform the following steps to construct and evaluate a classification model for tweet sentiment:

1. Add a **Split Data** module to the experiment and connect the output of the **Select Columns in Dataset** module to its input.
2. On the properties pane of the **Split Data** module set the following parameters:

* **Splitting mode**: Split Rows
* **Fraction of the rows in the first output dataset**: 0.8
* **Randomized split**: Checked
* **Random seed**: 1234
* **Stratified split**: false

1. Add a second **Split Data** module and connect the **Results Dataset1** (left) output of the first **Split Data** module to its input.
2. Set the properties of the second **Split Data** module as follows:

* **Splitting mode**: Split Rows
* **Fraction of the rows in the first output dataset**: 0.7
* **Randomized split**: Checked
* **Random seed**: 1234
* **Stratified split**: false

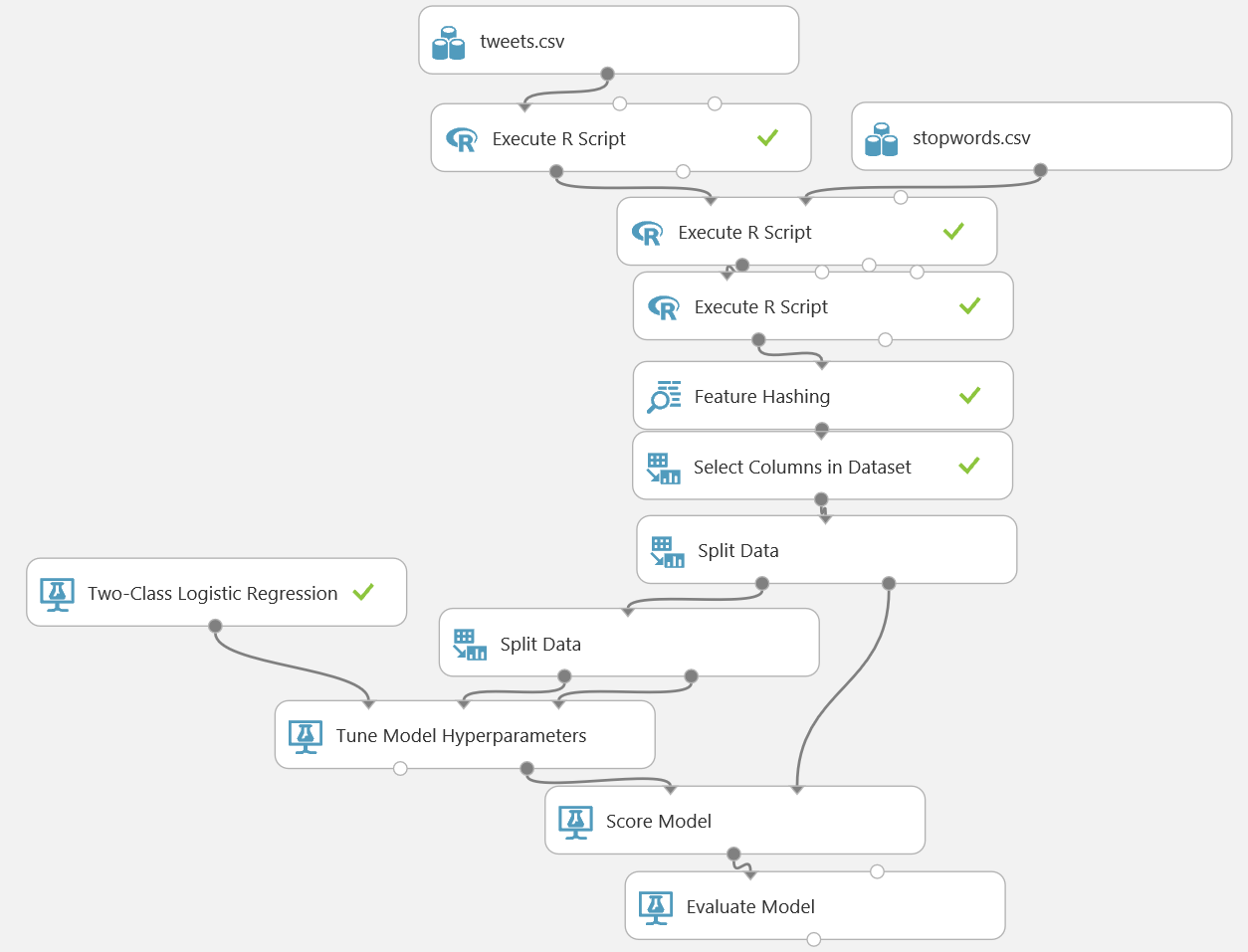
1. Add a **Tune Model Hyperparameters** module and connect the **Results Dataset1** (left) output of the second **Split Data** module to its **Training dataset** (middle) input. Then connect the **Results Dataset2** (right) output of the second **Split Data** module to its Optional validation dataset (right) input.
2. Add a **Two-Class Logistic Regression** module and connect its output to the **Untrained Module** (left) input of the **Tune Model Hyperparameters** module.
3. On the properties pane for the **Two-Class Logistic Regression** module set the following parameters:

* **Create trainer mode**: Parameter Range
* **Optimization tolerance**: **Use Range Builder**; Unchecked
* **Optimization tolerance**: 0.0001, 0.0000001
* **L1 regularization weight**: **Use Range Builder**; Unchecked
* **L1 regularization weight**: 0.0, 0.01, 0.1, 1.0
* **L2 regularization weight**: **Use Range Builder**; Unchecked
* **L2 regularization weight**: 0.01, 0.1, 1.0
* **Memory size for L-BFGS**: **Use Range Builder**; Unchecked
* **Memory size for L-BFGS**: 2, 20, 50
* **Random seed**: 1234
* **Allow unknown levels in categorical features**: Checked

1. On the properties pane for the **Tune Model Hyperparameters** module set the following parameters:

* **Specify parameter sweeping mode**: Random sweep
* **Maximum number of runs on random sweep**: 50
* **Random seed**: 1234
* **Label column**: sentiment
* **Metric for measuring performance for classification**: Accuracy
* **Metric for measuring performance for regression**: Mean absolute error

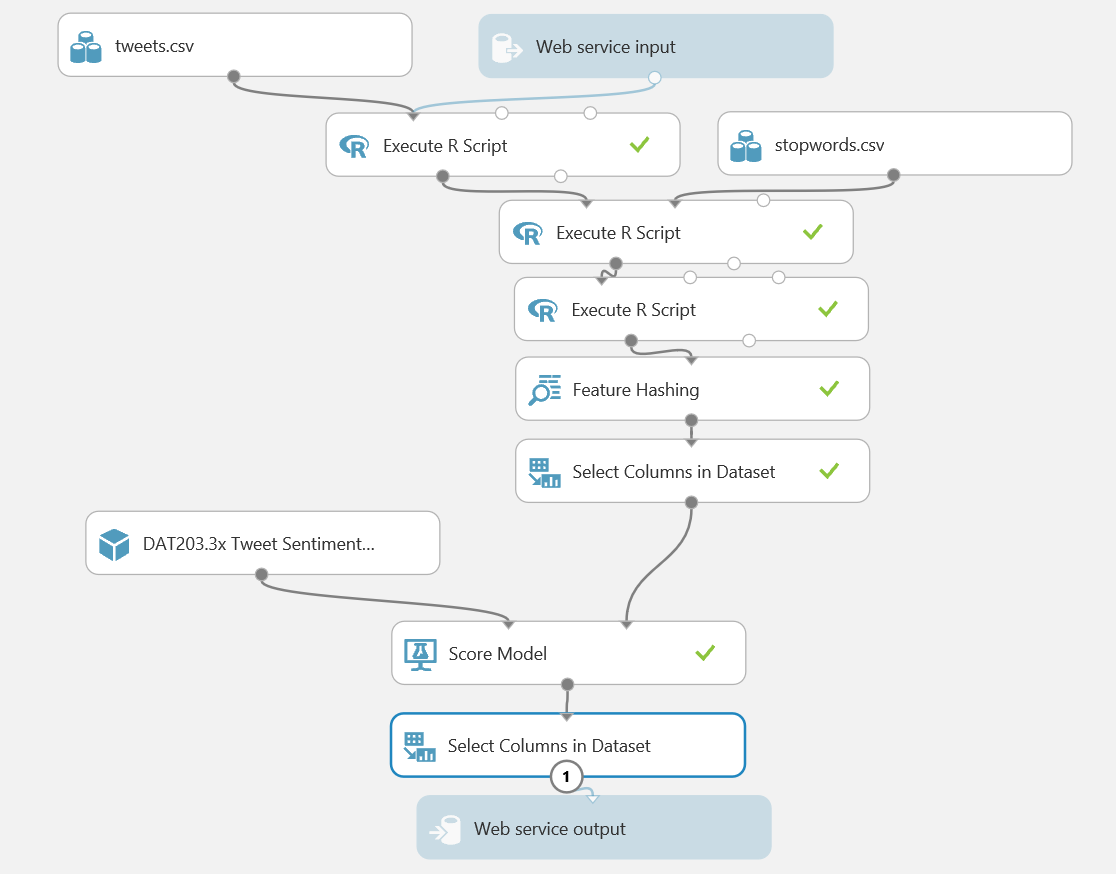
1. Add a **Score Model** module and connect the **Results Dataset2** (right) output of the first **Split Data** module to its Dataset (right) input. Then connect the **Trained best model** (right) output of the **Tune Model Hyperparameters** module to its **Trained model** (left) input.
2. Add an **Evaluate Model** module and connect the output of the **Score Model** module to its **Scored dataset** (left) input.
3. Verify that your experiment resembles the following:



1. Save and run the experiment.
2. Visualize the output of the **Evaluate Model** module. Scroll down until you see the performance statistics and verify that the model is performing at least better than a random guess.

## Create a Predictive Web Service

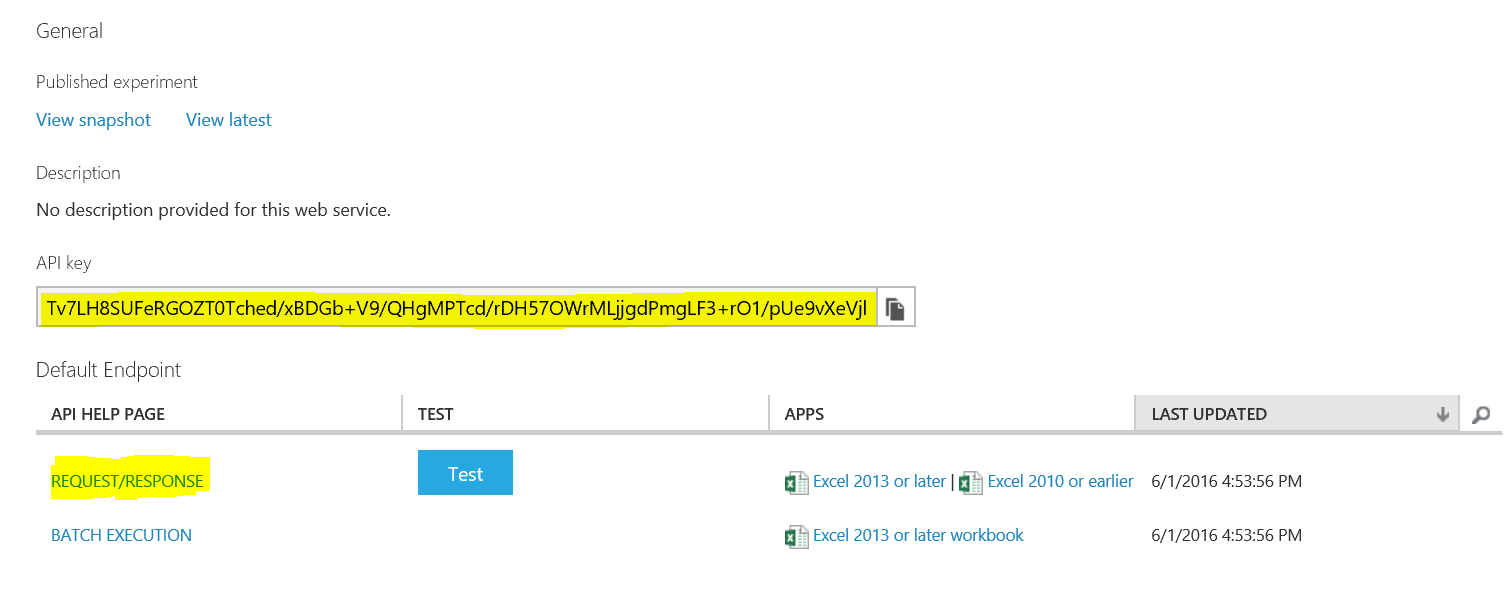
1. With the **Tweet Sentiment** experiment still open, click **Set Up Web Service (at the bottom of experiment page)**, and then click **Predictive Web Service (Recommended)**. When a banner at the bottom of the screen notifies you that the experiment has been created, click **Close** to remove it.
2. Save and run the experiment to read the data and pass it through the workflow. Note a new tab is added to your experiment named “predictive experiment”.
3. Next visualize the output of the **Score Model** module and note that the web service returns all of the feature columns
4. Add a **Select Columns in Dataset** module to the predictive experiment, and connect the output from the **Score Model** module to its input. Then connect its output to the **Web service output**.
5. In the properties for the **Select Columns in Dataset** module, use the column selector to select only the **Scored Labels** and **Scored Probability** columns.
6. Verify that your predictive experiment now looks like this:



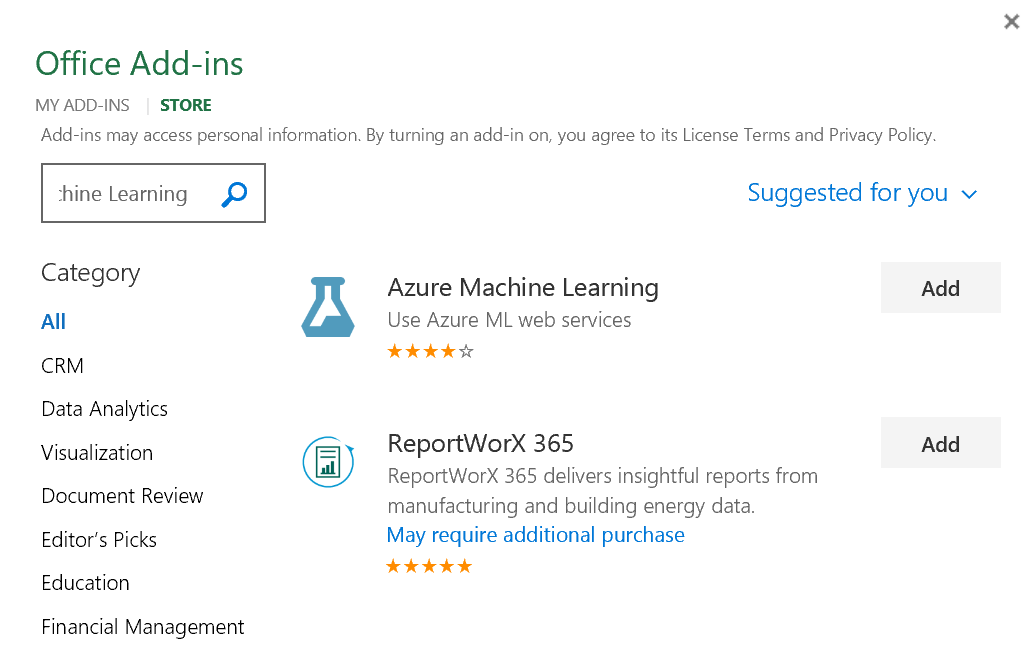
1. Save and run the experiment, and visualize the output of the **Select Columns in Dataset** module to verify that only the **Scored Labels** and **Scored Probability** columns are returned by the web service. Positive tweets are indicated by a value of 1, and negative tweets are indicated by a value of -1.

## Deploy and Use the Web Service

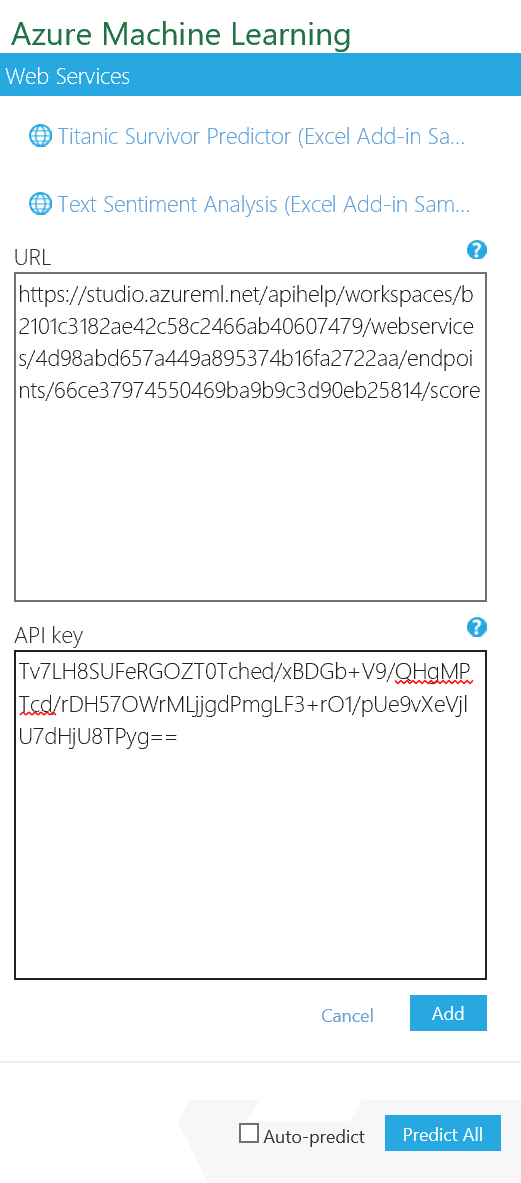
1. In the **Tweet Sentiment [Predictive Exp.]** experiment, click the **Deploy Web Service** icon at the bottom of the Azure ML Studio window.
2. Wait a few seconds for the dashboard page to appear, and note the **API key** and **Request/Response** link. You will use these to connect to the web service from a client application.



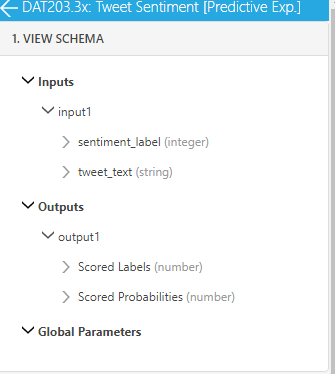
1. Leave the dashboard page open in your web browser, and open a new browser tab.
2. In the new browser tab, navigate to <https://office.live.com/start/Excel.aspx>. If prompted, sign in with your Microsoft account (use the same credentials you use to access Azure ML Studio.)
3. In Excel Online, create a new blank workbook.
4. 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:



1. 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.
2. 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.
3. 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.
4. Verify that the **Azure Machine Learning** pane in your workbook now resembles this, and click **Add**:



1. After the web service has been added, in the **Azure Machine Learning** pane, click **1. View Schema** and note the *inputs* expected by the web service (**sentiment\_label** and **tweet\_text**) and the *outputs* returned by the web service (**Scored Labels** and **Scored Probability**). In your workbook it will resemble like this. Note input required for this web service.



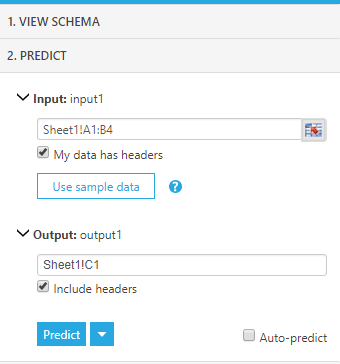
1. In the Excel worksheet select cell A1. Then in the **Azure Machine Learning** pane, collapse the **1. View Schema** section and in the **2. Predict** section, click **Use sample data**. this enters some sample input values in the worksheet.
2. Modify the sample data as follows:

**Tip**: Copy and paste the text from the table below to avoid typographical errors!

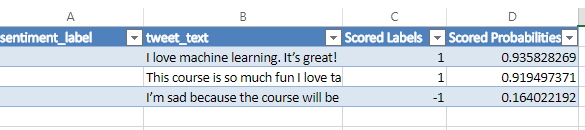
|  |  |
| --- | --- |
| sentiment\_label | tweet\_text |
|  | I love machine learning. It’s great! |
|  | This course is so much fun I love taking classes that are great it's my favorite thing to do |
|  | I’m sad because the course will be over soon |

1. Select the cells containing the input data (cells A1 to B4), and in the **Azure Machine Learning** pane, click the button to select the input range and confirm that it is **‘Sheet1’!A1:B4**.
2. Ensure that the **My data has headers** box is checked.
3. In the **Output** box type **C1**, and ensure the **Include headers** box is checked.

In your worksheet, it should resemble like this:



1. Click the **Predict** button, and after a few seconds, view the predicted sentiment(**Scored labels**) and the associated confidence (**Scored Probability**) for each tweet. The final output should be like this:



# Summary

In this hands-on experiment, you have used R with text data. You then created an Azure ML web service to classify tweets based on sentiment.