Natural Language Classifier in Node.js

Hands-On Lab

JeanCarl Bisson | jbisson@us.ibm.com | @dothewww


```
$ node classify.js
{
  "classifier_id": "0a1b23c456-nlc-789",
  "url": "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/0a1b23c456-nlc-
789",
  "text": "Is it sunny?",
  "top_class": "conditions",
  "classes": [
    {
      "class_name": "conditions",
      "confidence": 0.9930312851542132
    },
    {
      "class_name": "temperature",
      "confidence": 0.006968714845786802
    }
  ]
}
```

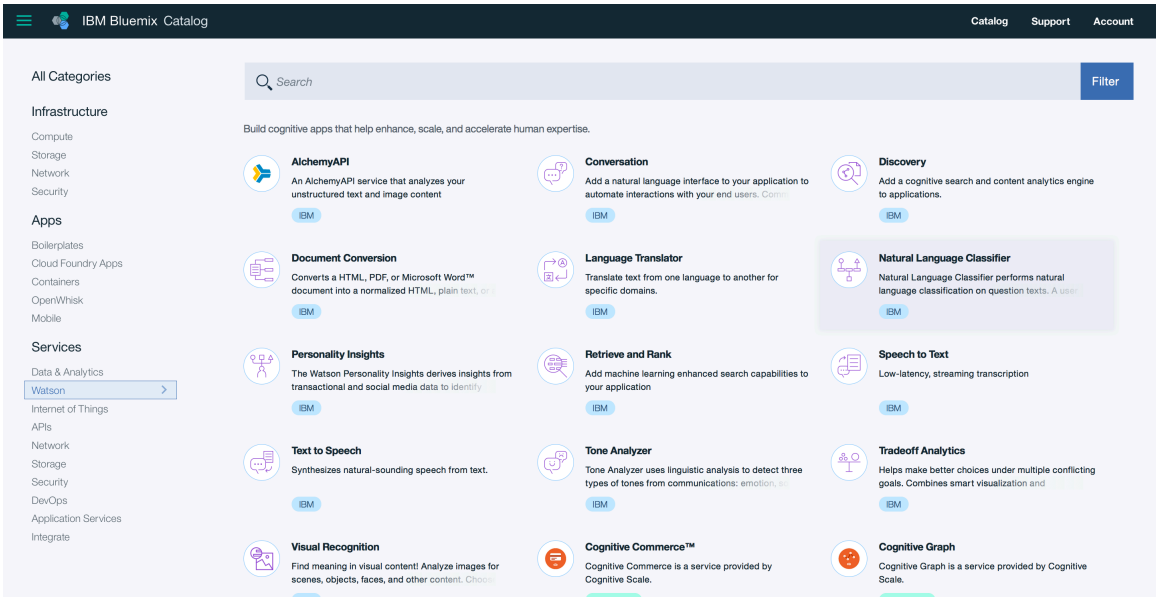
Create custom classifiers (pg.2) to use to classify text into classes with confidence scores (pg.5)

A digital copy of this lab and code snippets can be found at:
http://ibm.biz/nodejs-natural-language-classifier

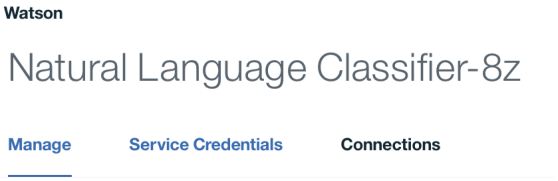
Create a Natural Language Classifier Service

To use the IBM Watson Natural Language Classifier service, we first need to create the Watson service in the IBM Bluemix console.

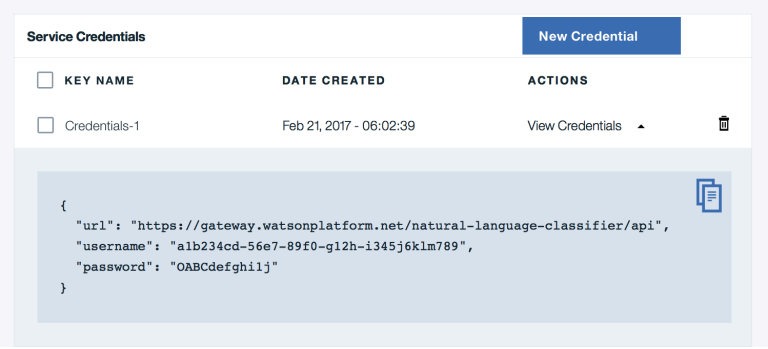
1. Sign up for an IBM Bluemix account at `bluemix.net`. If you already have a Bluemix account, login.
2. Click on the **Catalog** link in the top-right corner.
3. Click on the **Natural Language Classifier** tile under the Watson section. You can leave the fields as the default values. Click on **Create**.



4. Click on the **Service Credentials** tab.



5. Click on **View Credentials**. Copy the username and password credentials. You will use these credentials in the next two sections.



Training a Custom Classifier

In this section, we will train Watson with a couple of classes about weather questions. Refer to the CSV file that contains two columns, a question and a class name. A portion of the file is displayed below.

Is it hot outside?	temperature
Will it be uncomfortably hot?	temperature
Will it be sweltering?	temperature
How cold is it today?	temperature
Is it cold outside?	temperature
Will it be uncomfortably cold?	temperature
Is it windy?	conditions
Will it rain today?	conditions
What are the chances for rain?	conditions
Will we get snow?	conditions
Are we expecting sunny conditions?	conditions
Is it overcast?	conditions

Copy and paste the contents at:

ibm.biz/nodejs-nlc-weathercsv

1. Training Watson is straightforward once you have a list of example strings and their classes. Save this list into a CSV formatted file named `weather_training.csv`.

2. Create a new JavaScript file named `train.js`. Use the code shown below.

```
1.  "use strict";
2.  const NaturalLanguageClassifierV1 = require("watson-developer-cloud/natural-language-classifier/v1");
3.  const fs = require("fs");
4.
5.  const natural_language_classifier = new NaturalLanguageClassifierV1({
6.    username: "<USERNAME>",
7.    password: "<PASSWORD>",
8.    version: "v1"
9.  });
10.
11. // Creating a classifier
12. const params = {
13.   language: "en",
14.   name: "<CLASSIFIER NAME>",
15.   training_data: fs.createReadStream("<FILE PATH AND NAME TO CSV FILE>")
16. };
17.
18. natural_language_classifier.create(params, function(err, response) {
19.   if (err) {
20.     console.log(err);
21.   } else {
22.     console.log(JSON.stringify(response, null, 2));
23.   }
24. });
```

Copy and paste the code at:

ibm.biz/nodejs-nlc-train

Change the following:

- replace <USERNAME> and <PASSWORD> (lines 6 and 7) with values from the service credentials from Step #5 on page 2
- replace <CLASSIFIER NAME> (line 14) with a custom name for the classifier. For this example, let's choose `weather-questions`
- replace <FILENAME AND PATH TO CSV FILE> (line 15) with the file path and name where the CSV file is located. For this example, enter `./weather_training.csv`

There are also a couple of things to take note of in the code:

- the language parameter (line 10) of the training data is a 2-letter primary language code as assigned in ISO standard 639. Supported languages are English (en), Arabic (ar), French (fr), German, (de), Italian (it), Japanese (ja), Portuguese (pt), and Spanish (es)
- the classifier name (line 11), <CLASSIFIER NAME>, is a label for your convenience

3. Run the Node.js application by using the command `node app.js` . You should see output like the following. However, some values will be different.

```
$ node app.js
{
  "classifier_id": "0a1b23c456-nlc-789",
  "name": "weather-questions",
  "language": "en",
  "created": "2017-02-21T18:26:02.327Z",
  "url": "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/0a1b23c456-
nlc-789",
  "status": "Training",
  "status_description": "The classifier instance is in its training phase, not yet ready to accept
classify requests"
}
```

There are a couple of things to take note of in the response:

- take note of the `classifier_id` value. This will be needed when we call the Natural Language Classifier service to classify new phrases
- the `status` property has a value of `Training` while Watson is ingesting the data and training. When training is complete, this status should become `Available`. Other values are `Non Existent`, `Failed` or `Unavailable`

Classify Phrase Using the Custom Classifier

In the last section, we created a custom classifier of weather questions using the IBM Watson Natural Language Classifier service. In this section, we will use this classifier to classify questions such as "Is it sunny?"

1. Create a new JavaScript file named `classify.js`. Use the code shown below.

```
1.  "use strict";
2.  const NaturalLanguageClassifierV1 = require("watson-developer-cloud/natural-language-classifier/v1");
3.  const fs = require("fs");
4.
5.  const natural_language_classifier = new NaturalLanguageClassifierV1({
6.    username: "<USERNAME>",
7.    password: "<PASSWORD>",
8.    version: "v1"
9.  });
10.
11. natural_language_classifier.classify(
12.   {
13.     text: "Is it sunny?",
14.     classifier_id: "<CLASSIFIER ID>"
15.   },
16.   function(err, response) {
17.     if (err) {
18.       console.log("error:", err);
19.     } else {
20.       console.log(JSON.stringify(response, null, 2));
21.     }
22.   }
23. );
```

Copy and paste the code at:

ibm.biz/nodejs-nlc-classify

Change the following:

- replace <USERNAME> and <PASSWORD> (lines 6 and 7) with values from the service credentials from Step #5 on page 2
- replace <CLASSIFIER ID> (line 14) with the classifier ID created from the previous section

2. Run the Node.js application with the command `node classify.js`. You should see output like the following. However, some values will be different.

```
$ node classify.js
{
  "classifier_id": "0a1b23c456-nlc-789",
  "url": "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/0a1b23c456-
nlc-789",
  "text": "Is it sunny?",
  "top_class": "conditions",
  "classes": [
    {
      "class_name": "conditions",
      "confidence": 0.9930312851542132
    },
    {
      "class_name": "temperature",
      "confidence": 0.006968714845786802
    }
  ]
}
```

There are a couple of things to take note of in the JSON response:

- the `classes` property contains an array of up to ten `class_name/confidence` pairs that are sorted in descending order of confidence. If there are fewer than 10 classes, the sum of the confidence values is 100%
- the `confidence` score is a decimal percentage that represents the confidence that Watson has in this class. A higher value represent a higher confidence
- the property named `top_class` returns the class with the highest confidence

3. Experiment by changing the question on line 13 and rerun the `node classify.js` command to see how values change.

Listing Classifiers

To get a list of classifiers that have been created with this Watson Natural Language Classifier service, use the following code:

```
1.  "use strict";
2.  const NaturalLanguageClassifierV1 = require("watson-developer-cloud/natural-language-classifier/v1");
3.
4.  const natural_language_classifier = new NaturalLanguageClassifierV1({
5.    username: "<USERNAME>",
6.    password: "<PASSWORD>",
7.    version: "v1"
8.  });
9.
10. natural_language_classifier.list({},
11.   function(err, response) {
12.     if (err)
13.       console.log("error:", err);
14.     else
15.       console.log(JSON.stringify(response, null, 2));
16. });
```

Copy and paste the code at:

ibm.biz/nodejs-nlc-list

```
$ node list.js
{
  "classifiers": [
    {
      "classifier_id": "0a1b23c456-nlc-789",
      "url": "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/0a1b23c456-
nlc-789",
      "name": "weather-questions",
      "language": "en",
      "created": "2017-02-21T18:26:02.327Z"
    }
  ]
}
```

Getting Status of a Classifier

To get a list of classifiers that have been created with this Watson Natural Language Classifier service, use the following code:

```
1.  "use strict";
2.  const NaturalLanguageClassifierV1 = require("watson-developer-cloud/natural-language-classifier/v1");
3.
4.  const natural_language_classifier = new NaturalLanguageClassifierV1({
5.    username: "<USERNAME>",
6.    password: "<PASSWORD>",
7.    version: "v1"
8.  });
9.
10. natural_language_classifier.status({
11.     classifier_id: "<CLASSIFER ID>"
12.   },
13.   function(err, response) {
14.     if (err)
15.       console.log("error:", err);
16.     else
17.       console.log(JSON.stringify(response, null, 2));
18. });
```

Copy and paste the code at:

ibm.biz/nodejs-nlc-status

```
$ node status.js
{
  "classifier_id": "0a1b23c456-nlc-789",
  "name": "weather-questions",
  "language": "en",
  "created": "2017-02-21T18:26:02.327Z",
  "url": "https://gateway.watsonplatform.net/natural-language-classifier/api/v1/classifiers/0a1b23c456-nlc-
789",
  "status": "Available",
  "status_description": "The classifier instance is now available and is ready to take classifier requests."
}
```