Utilização dos Serviços Watson no Bluemix

Franklin Lindemberg Guimarães
flcca@br.ibm.com

1. Serviços do Watson

Todas as informações dos serviços Watson podem ser encontradas no site do Watson Developer Cloud (https://www.ibm.com/watson/developercloud/). As informações estão disponíveis apenas em inglês. Abaixo seguem os links úteis e as referências utilizando nodeJS das API que foram apresentadas no treinamento.

1.1. Personality Insights

Website: http://www.ibm.com/watson/developercloud/personality-insights.html
Documentação: http://www.ibm.com/watson/developercloud/doc/personality-insights/
Referência da API: http://www.ibm.com/watson/developercloud/personality-insights/api/v2/#
Demonstração: https://personality-insights-livedemo.mybluemix.net
Código de demonstração no Github: https://github.com/watson-developer-cloud/personality-insights-nodejs
API no Bluemix: https://console.ng.bluemix.net/catalog/services/personality-insights/

1.2. Conversation

Website: https://www.ibm.com/watson/developercloud/conversation.html
Documentação: https://www.ibm.com/watson/developercloud/doc/conversation/
Referência da API: http://www.ibm.com/watson/developercloud/conversation/api/v1/#
Demonstração: https://conversation-demo.mybluemix.net
Código de demonstração no Github: https://github.com/watson-developer-cloud/conversation-simple
API no Bluemix: https://console.ng.bluemix.net/catalog/services/conversation/

1.3. Natural Language Classifier (NLC)

Website: https://www.ibm.com/watson/developercloud/nl-classifier.html
Documentação: https://www.ibm.com/watson/developercloud/doc/nl-classifier/
Referência da API: https://www.ibm.com/watson/developercloud/natural-language-classifier/api/v1/?node#
Demonstração: http://natural-language-classifier-demo.mybluemix.net
Código de demonstração no Github: https://github.com/watson-developer-cloud/natural-language-classifier-nodejs

API no Bluemix: https://console.ng.bluemix.net/catalog/services/natural-language-classifier/

1.4. Tone Analyzer

Website: https://www.ibm.com/watson/developercloud/tone-analyzer.html
Documentação: https://www.ibm.com/watson/developercloud/doc/tone-analyzer/
Referência da API: https://www.ibm.com/watson/developercloud/tone-analyzer/api/v3/#
Demonstração: https://tone-analyzer-demo.mybluemix.net
Código de demonstração no Github: https://github.com/watson-developer-cloud/tone-analyzer-nodejs
API no Bluemix: https://console.ng.bluemix.net/catalog/services/tone-analyzer/

1.5. Test to Speech

Website: https://www.ibm.com/watson/developercloud/text-to-speech.html
Documentação: https://www.ibm.com/watson/developercloud/doc/text-to-speech/
Referência da API: https://www.ibm.com/watson/developercloud/text-to-speech/api/v1/#
Demonstração: https://text-to-speech-demo.mybluemix.net
Código de demonstração no Github: https://github.com/watson-developer-cloud/text-to-speech-nodejs
API no Bluemix: https://console.ng.bluemix.net/catalog/services/text-to-speech/

1.6. Visual Recognition

Website: https://www.ibm.com/watson/developercloud/visual-recognition.html
Documentação: https://www.ibm.com/watson/developercloud/doc/visual-recognition/
Referência da API: https://www.ibm.com/watson/developercloud/visual-recognition/api/v3/
Demonstração: http://visual-recognition-demo.mybluemix.net
Código de demonstração no Github: https://github.com/watson-developer-cloud/visual-recognition-nodejs
API no Bluemix: https://console.ng.bluemix.net/catalog/services/visual-recognition/

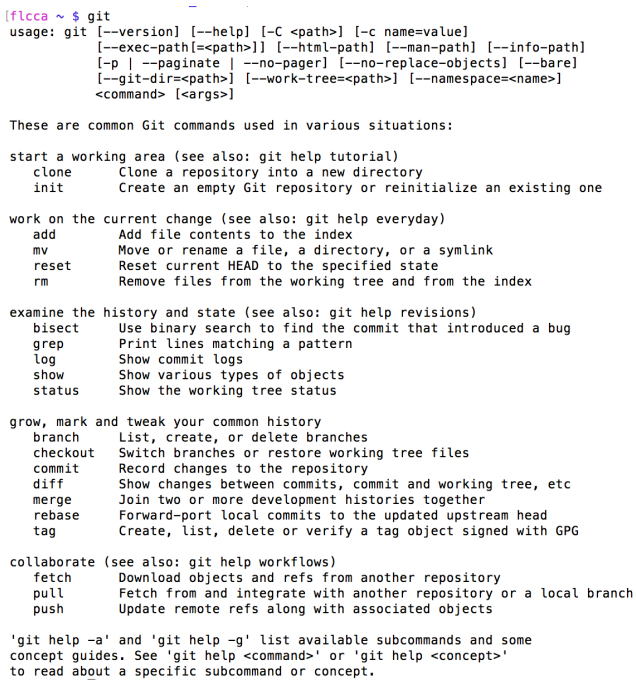
2. Passo a passo – Primeira atividade

Na primeira atividade do treinamento nós baixamos o código de exemplo do Personality Insights do GitHub e rodamos tanto local, quanto no bluemix, bem como integrando com o serviço do Personality Insights.

2.1. Instalação do git

Inicialmente foi necessário instalar o git nas máquinas. Isso foi feito baixando-se o instalador do site https://git-scm.com/downloads (selecionando-se o sistema operacional correspondente).

Após a instalação do git, deve-se abrir um terminal (ou fechar e reabrir caso já tenha um aberto) e, ao se digitar git, se aparecer uma tela igual a abaixo é porque ele foi instalado corretamente.



2.2. Download do código de exemplo

Para esta etapa, nós acessamos a pasta pelo terminal na qual queremos baixar o código (Exemplo: C:/git_exemplos). Em seguida foi utilizado o comando:

git clone https://github.com/watson-developer-cloud/personality-insights-nodejs

Este comando faz com que o código seja baixado do github para a pasta em questão no terminal. Após a finalização do comando é criada uma pasta nova na qual contém todo o código necessário. Deve entao acessar esta pasta (Ex: C:/git_exemplos/personality_insights_nodejs).

2.3. Criação do serviço no bluemix

Deve-se em seguida acessar o bluemix, na página do serviço do personality insights, e criar uma instância do mesmo (https://console.ng.bluemix.net/catalog/services/personality-insights/). Após a criação deve-se guardar os dados de username e password que podem ser obtidos na opção “credenciais do serviço”.

2.4. Alterar arquivo de credenciais

Deve-se em seguida adicionar as credenciais obtidas ao criar o serviço do personality insights no arquivo credentials.json (o qual fica na raiz da pasta da aplicação). Desta forma nós estamos dando acesso para a aplicação acessar o serviço. Obs: deve-se substituir tudo em amarelo (inclusive os < e >) pelo password e username.

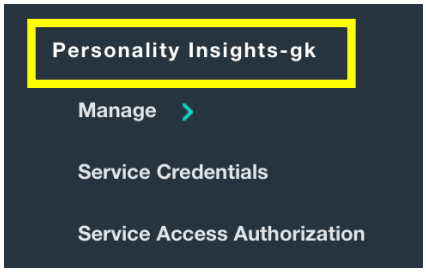
```
"personality_insights" : {
  "password": "<password>",
  "username": "<username>",
  "version": "v2",
```

```
  "headers": {
    "X-Watson-Learning-Opt-Out": 1
  }},
```

2.5. Alterar arquivo manifest

Deve-se também alterar o arquivo manifest para refletir a utilização do serviço criadono bluemix. Deve-se alterar seguindo a regra abaixo:

- Trocar o que está em amarelo pelo nome do serviço do personality insights que foi criado. Este nome pode ser encontrado um pouco acima de onde se encontra o botão das credenciais do serviço.
- Adicionar a linha que esta em verde, escolhendo um nome unico para o host, o qual corresponderá ao endereço da url. Desta forma não é necessário passar o parâmetro –n ai fazer o comando cf push. Caso durante o cf push haja algum problema que diga que o host escolhido já está em uso, deve-se escolher um nome nome para o host e tentar novamente.




```
---
declared-services:
  Personality Insights-ha:
    label: personality_insights
    plan: tiered

applications:
- name: personality-insights-livedemo
  host: <nome unico da aplicacao (será a url>
  command: npm start
  path: .
  memory: 512M
  instances: 1
  services:
  - Personality Insights-ha
  env:
    NODE_ENV: production
```

2.6. Deploy local

Para realizar o deploy local deve, a partir do terminal, ir na pasta raiz da aplicação e aplicar os seguintes comandos:

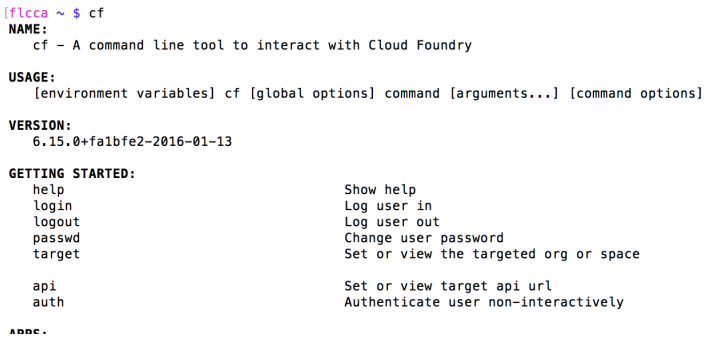
- npm install (o qual instalará todas as dependências necessárias para a aplicação funcionar)
- npm start (o qual iniciará a aplicação)

Após o comando de npm start deve-se abrir o browser e digitar localhost:3000 e a aplicação deve estar funcionando corretamente.

2.7. Deploy no bluemix

Antes de realizar o deploy no bluemix deve-se instalar aplicação do cloudfoundry (a qual permitirá o uso do comando cf pelo terminal/prompt). O arquivo de instalação pode ser obtido no link https://github.com/cloudfoundry/cli/releases, selecionando-se o respectivo sistema peracional)

Após a instalação, pode-se testar se foi instalado corretamente abrindo-se o terminal/prompt (ou fechando e abrindo caso haja algum aberto) e digitando-se o comando cf. Deve ser mostrado o mesmo que a foto abaixo (na foto contem-se apenas o inicio do texto pois é muito longo).



Caso esteja tudo funcionando corretamente, deve-se efetuar os seguintes comandos:

- cf login (e então inserir o email e senha do bluemix quando solicitado. Escolher também a organização e o espaço que deseja-se logar, geralmente há apenas uma opção para cada)
- cf push (este comando irá enviar o código da aplicação para o bluemix e executá-la).

Após o comando cf push ser finalizado a aplicação poderá ser acessar a partir do site <nome único da aplicação escolhido no parâmetro host do manifest>.mybluemix.net.

Caso ocorra algum problema, pode-se consultar o log do bluemix executando-se o comando abaixo:

- cf logs <valor do parâmetro name do manifest> --recent (no nosso caso pode-se substituir a parte com fundo amarelo por personality-insights-livedemo

3. Passo a passo – Segunda atividade

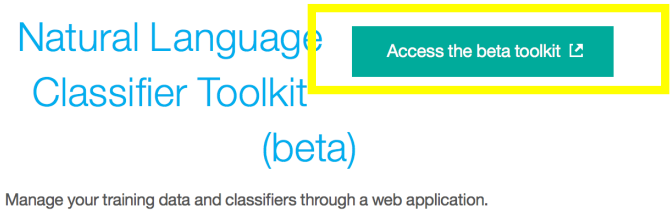
Na segunda atividade nós utilizamos o serviço do Natural Language Classifier (NLC) para criar um classificador e acessá-lo a partir de uma aplicação criada a partir de um template do bluemix.

3.1. Criar serviço NLC no bluemix

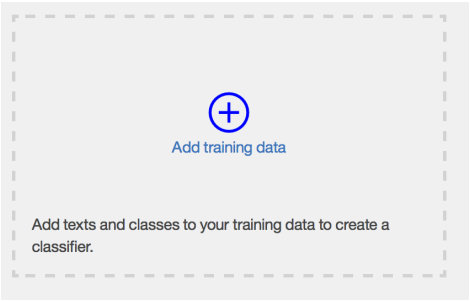
Deve-se acessar a página https://console.ng.bluemix.net/catalog/services/natural-language-classifier/ e criar o serviço do Natural Language Classifier.

3.2. Configurar o classificador

Após a criação, deve acessar o toolkit do NLC. Pode ser necessário clicar num botão que solicita logar no bluemix e também dar autorização do toolkit acessar o NLC.



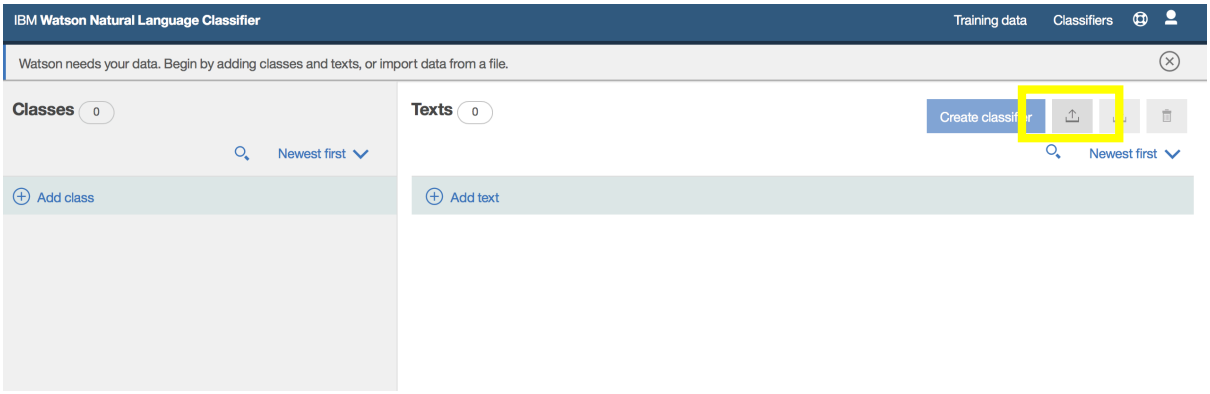
Deve-se clicar em adicionar training data



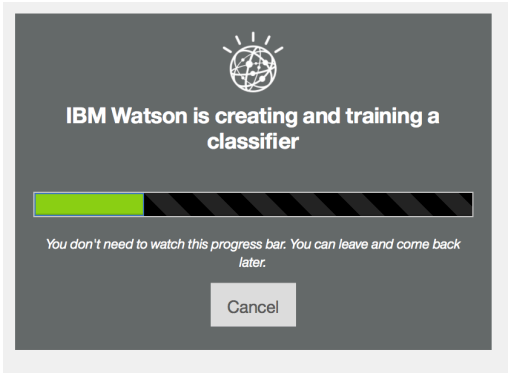
Em seguida deve-se inserir as classes desejadas e os respectivos exemplos. Os dados utilizados durante o treinamento encontram-se abaixo:

```
vai esfriar hoje?,temperatura
devo usar roupa de frio ou calor?,temperatura
yoga amanha talvez,mastertech
estou com calor,temperatura
estou com frio,temperatura
qual a temperatura hoje?,temperatura
yoga hj nao,mastertech
watson,mastertech
aprenderam iot?,mastertech
eles veem css em tudo,mastertech
demoday cafe da madalena,mastertech
sabem programar em nodejs?,mastertech
estou congelando,temperatura
quantos graus hoje?,temperatura
esta calor la fora?,temperatura
estou tremendo de frio,temperatura
estudaram javascript?,mastertech
eu devo usar uma malha?,temperatura
ninjas html,mastertech
foca no codigo,mastertech
```

Caso queiram, pode ser importado diretamente o arquivo export.csv que foi enviado pelo slack. Deve-se apenas clicar no botão de import conforme foto abaixo:



Após todos os dados serem adicionados nos dados de treinamento, deve-se clicar em criar classificador, escolher um nome para o mesmo e a lingua que se deseja utilizar. Deve-se então aguardar o treinamento. Demora entre 10 e 20 minutos para finalizar.

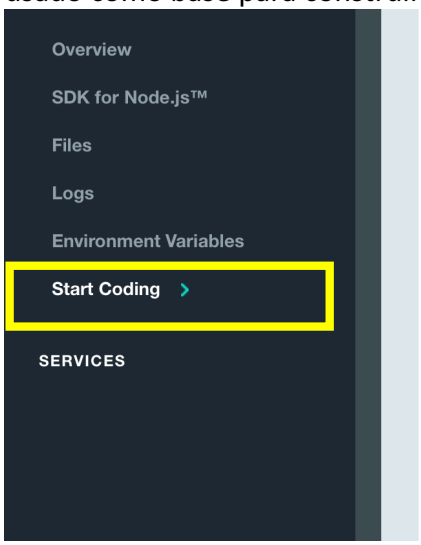


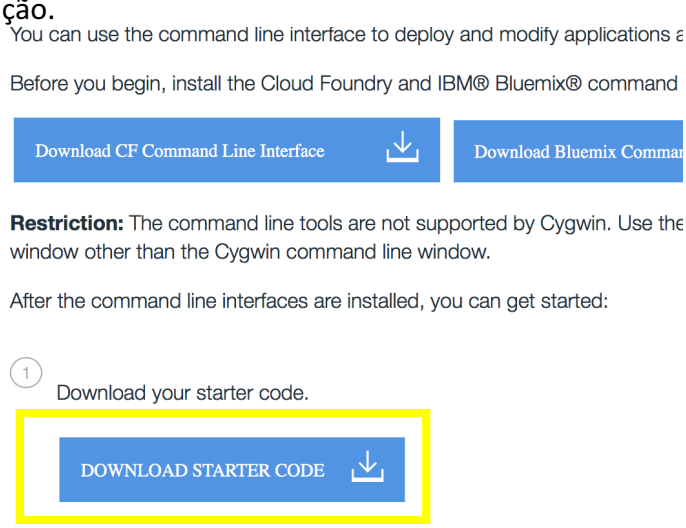
Aguardando o treinamento

3.3. Criar aplicação no bluemix

Enquanto se aguarda o treinamento, podemos ir criando o aplicativo no Bluemix. Deve-se acessar o bluemix, ir em painel, em seguida criar aplicação, escolhar a opção WEB, e escolher a opção SDK for Node.js. Em seguida deve-se escolher o nome da aplicação (deve ser um nome único pois será o mesmo utilizado na URL).

Em seguida deve-se ir em iniciar codificação e baixar o código de início, o qual foi usado como base para construir a aplicação.





3.4. Realizar modificações no código da aplicação

A partir do código baixado do bluemix, devemos fazer as seguintes alterações

3.4.1. Arquivo package.json

Deve-se adicionar a dependência do watson-developer-cloud, que é a SDK para NodeJS que permite utilizar os serviços do Watson.

```
"dependencies": {
                "express": "4.13.x",
                "cfenv": "1.0.x",
                "watson-developer-cloud": "*"
        },
```

3.4.2. Arquivo app.js

Deve-se adicionar as partes representadas em amarelo no código abaixo. Obs: deve-se obter o username e password do serviço do NLC a partir das credencias do serviço no bluemix. O classifier ID pode ser obtido após o treinamento do classificador.

```
/*eslint-env node*/

//------------------------------------------------------------------------------
// node.js starter application for Bluemix
//------------------------------------------------------------------------------

// This application uses express as its web server
// for more info, see: http://expressjs.com
var express = require('express');

// cfenv provides access to your Cloud Foundry environment
// for more info, see: https://www.npmjs.com/package/cfenv
var cfenv = require('cfenv');

// create a new express server
var app = express();

// serve the files out of ./public as our main files
app.use(express.static(__dirname + '/public'));

// get the app environment from Cloud Foundry
var appEnv = cfenv.getAppEnv();

var watson = require('watson-developer-cloud');

// start server on the specified port and binding host
app.listen(appEnv.port, '0.0.0.0', function() {
  // print a message when the server starts listening
  console.log("server starting on " + appEnv.url);

  var nlc = watson.natural_language_classifier({
    username: '<username do servico do NLC>',
    password: '<password do serviço do NLC>',
    version: 'v1'
  });

  nlc.classify({
        text: 'How hot will it be today?',
        classifier_id: '<id do classificador que foi treinado>'
    },
    function (err, response) {
      if (err)
        console.log('error:', err);
      else
```

```
            console.log(JSON.stringify(response, null, 2));
    });
});
```

3.5. Execução da aplicação.

A aplicação pode ser executada tanto localmente quando no bluemix, seguindo-se os mesmo passos explicados no tutorial da primeira atividade.

Obs: como não há ainda a interface gráfica (front end) o retorno do serviço do nlc será apresentado no terminal/prompt. Caso a aplicaçãoe esteja sendo testada no bluemix, deve utilizar o comando cf logs (conforme explicado na primeira atividade) para que se possa observar o retorno do nlc.