JAVA

import java.io.\*;

import java.net.MalformedURLException;

import java.util.Base64;

import java.util.HashMap;

import java.util.Map;

import java.net.HttpURLConnection;

import java.net.URL;

import java.nio.charset.StandardCharsets;

public class HttpClientTest {

public static void main(String[] args) throws IOException {

// NOTE: you must manually set API\_KEY below using information retrieved from your IBM Cloud account. (https://eu-gb.dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-authentication.html?context=cpdaas)

String API\_KEY = "<your API key>";

HttpURLConnection tokenConnection = null;

HttpURLConnection scoringConnection = null;

BufferedReader tokenBuffer = null;

BufferedReader scoringBuffer = null;

try {

// Getting IAM token

URL tokenUrl = new URL("https://iam.cloud.ibm.com/identity/token?grant\_type=urn:ibm:params:oauth:grant-type:apikey&apikey=" + API\_KEY);

tokenConnection = (HttpURLConnection) tokenUrl.openConnection();

tokenConnection.setDoInput(true);

tokenConnection.setDoOutput(true);

tokenConnection.setRequestMethod("POST");

tokenConnection.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");

tokenConnection.setRequestProperty("Accept", "application/json");

if (tokenConnection.getResponseCode() == 200) { // Successful response

tokenBuffer = new BufferedReader(new InputStreamReader(tokenConnection.getInputStream()));

} else { // Error response

tokenBuffer = new BufferedReader(new InputStreamReader(tokenConnection.getErrorStream()));

}

String line;

StringBuffer jsonString = new StringBuffer();

while ((line = tokenBuffer.readLine()) != null) {

jsonString.append(line);

}

System.out.println("Token response body:\n" + jsonString);

// Scoring request

URL scoringUrl = new URL("https://private.eu-gb.ml.cloud.ibm.com/ml/v4/deployments/b7d8b5a3-c153-4f81-8046-408794325adc/predictions?version=2021-05-01");

String iam\_token = "Bearer " + jsonString.toString().split(":")[1].split("\"")[1];

scoringConnection = (HttpURLConnection) scoringUrl.openConnection();

scoringConnection.setDoInput(true);

scoringConnection.setDoOutput(true);

scoringConnection.setRequestMethod("POST");

scoringConnection.setRequestProperty("Accept", "application/json");

scoringConnection.setRequestProperty("Authorization", iam\_token);

scoringConnection.setRequestProperty("Content-Type", "application/json; charset=UTF-8");

OutputStreamWriter writer = new OutputStreamWriter(scoringConnection.getOutputStream(), "UTF-8");

// NOTE: manually define and pass the array(s) of values to be scored in the next line

String payload = """

{\"input\_data\": [

{

\"fields\": [array\_of\_input\_fields],

\"values\": [array\_of\_values\_to\_be\_scored, another\_array\_of\_values\_to\_be\_scored]

}

]}""";

writer.write(payload);

writer.close();

if (scoringConnection.getResponseCode() == 200) { // Successful response

scoringBuffer = new BufferedReader(new InputStreamReader(scoringConnection.getInputStream()));

} else { // Error response

scoringBuffer = new BufferedReader(new InputStreamReader(scoringConnection.getErrorStream()));

}

String lineScoring;

StringBuffer jsonStringScoring = new StringBuffer();

while ((lineScoring = scoringBuffer.readLine()) != null) {

jsonStringScoring.append(lineScoring);

}

System.out.println("Scoring response body:\n" + jsonStringScoring);

} catch (IOException e) {

System.out.println("The request was not valid.");

System.out.println(e.getMessage());

}

finally {

if (tokenConnection != null) {

tokenConnection.disconnect();

}

if (tokenBuffer != null) {

tokenBuffer.close();

}

if (scoringConnection != null) {

scoringConnection.disconnect();

}

if (scoringBuffer != null) {

scoringBuffer.close();

}

}

}

}

JAVASCRIPT

const XMLHttpRequest = require("xmlhttprequest").XMLHttpRequest;

// NOTE: you must manually enter your API\_KEY below using information retrieved from your IBM Cloud account (https://eu-gb.dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-authentication.html?context=cpdaas)

const API\_KEY = "<your API key>";

function getToken(errorCallback, loadCallback) {

const req = new XMLHttpRequest();

req.addEventListener("load", loadCallback);

req.addEventListener("error", errorCallback);

req.open("POST", "https://iam.cloud.ibm.com/identity/token");

req.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");

req.setRequestHeader("Accept", "application/json");

req.send("grant\_type=urn:ibm:params:oauth:grant-type:apikey&apikey=" + API\_KEY);

}

function apiPost(scoring\_url, token, payload, loadCallback, errorCallback){

const oReq = new XMLHttpRequest();

oReq.addEventListener("load", loadCallback);

oReq.addEventListener("error", errorCallback);

oReq.open("POST", scoring\_url);

oReq.setRequestHeader("Accept", "application/json");

oReq.setRequestHeader("Authorization", "Bearer " + token);

oReq.setRequestHeader("Content-Type", "application/json;charset=UTF-8");

oReq.send(payload);

}

getToken((err) => console.log("An error occurred submitting the request."), () => {

let tokenResponse;

try {

tokenResponse = JSON.parse(this.responseText);

} catch(ex) {

// TODO: handle parsing exception

}

// NOTE: manually define and pass the array(s) of values to be scored in the next line

const payload = `{"input\_data": [

{

"fields": [array\_of\_input\_fields],

"values": [array\_of\_values\_to\_be\_scored, another\_array\_of\_values\_to\_be\_scored]

}

]}`;

const scoring\_url = "https://private.eu-gb.ml.cloud.ibm.com/ml/v4/deployments/b7d8b5a3-c153-4f81-8046-408794325adc/predictions?version=2021-05-01";

apiPost(scoring\_url, tokenResponse.access\_token, payload, function (resp) {

let parsedPostResponse;

try {

parsedPostResponse = JSON.parse(this.responseText);

} catch (ex) {

// TODO: handle parsing exception

}

console.log("Scoring response");

console.log(parsedPostResponse);

}, function (error) {

console.log(error);

});

});

PYTHON

import requests

# NOTE: you must manually set API\_KEY below using information retrieved from your IBM Cloud account (https://eu-gb.dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-authentication.html?context=cpdaas)

API\_KEY = "<your API key>"

token\_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey": API\_KEY, "grant\_type": 'urn:ibm:params:oauth:grant-type:apikey'})

mltoken = token\_response.json()["access\_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line

payload\_scoring = {"input\_data": [

{

"fields": [array\_of\_input\_fields],

"values": [array\_of\_values\_to\_be\_scored, another\_array\_of\_values\_to\_be\_scored]

}

]}

response\_scoring = requests.post('https://private.eu-gb.ml.cloud.ibm.com/ml/v4/deployments/b7d8b5a3-c153-4f81-8046-408794325adc/predictions?version=2021-05-01', json=payload\_scoring,

headers={'Authorization': 'Bearer ' + mltoken})

print("Scoring response")

try:

print(response\_scoring.json())

except ValueError:

print(response\_scoring.text)

except Exception as e:

print(f"An unexpected error occurred: {e}")

SCALA

import scalaj.http.{Http, HttpOptions}

import scala.util.{Success, Failure}

import java.util.Base64

import java.nio.charset.StandardCharsets

import play.api.libs.json.\_

// NOTE: you must manually set API\_KEY below using information retrieved from your IBM Cloud account (https://eu-gb.dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/ml-authentication.html?context=cpdaas)

val API\_KEY = "<your API key>"

// Get IAM service token

val iam\_url = "https://iam.cloud.ibm.com/identity/token"

val iam\_response = Http(iam\_url).header("Content-Type", "application/x-www-form-urlencoded").header("Accept",

"application/json").postForm(Seq("grant\_type" -> "urn:ibm:params:oauth:grant-type:apikey",

"apikey" -> API\_KEY)).asString

val iamtoken\_json: JsValue = Json.parse(iam\_response.body)

val iamtoken = (iamtoken\_json \ "access\_token").asOpt[String] match {

case Some(x) => x

case None => ""

}

// TODO: manually define and pass list of values to be scored

val payload\_scoring = Json.stringify(Json.toJson(Map("input\_data" -> List(Map(

"fields" -> Json.toJson(List(list\_of\_input\_fields)),

"values" -> Json.toJson(list\_of\_values\_to\_be\_scored)

)))))

val scoring\_url = "https://private.eu-gb.ml.cloud.ibm.com/ml/v4/deployments/b7d8b5a3-c153-4f81-8046-408794325adc/predictions?version=2021-05-01"

val response\_scoring = Http(scoring\_url).postData(payload\_scoring).header("Content-Type",

"application/json").header("Authorization", "Bearer " + iamtoken).option(HttpOptions.

method("POST")).option(HttpOptions.connTimeout(10000)).option(HttpOptions.readTimeout(50000)).asString

println("scoring response")

println(response\_scoring)