

# Machine Learning



# Overview

- Machine Learning APIs
- Using REST APIs
  - API Hub
  - Using a REST
- Using in Code
  - First application in JS
  - Java
  - Python
- In Production
  - How to authenticate.
  - Deployment of a simple application.



# SAP API Hub

- Functional APIs are typically classification
  - Text
  - Images
  - (some exceptions)
- Business
  - Ticket
  - Finance
  - Solution
- Operation/customization
  - Customizable models or new models.

# Excercise: API Hub

- Pick an API
- Try out
- Enter parameters
- Hit execute

Note that parameters are typically JSON or Files (sometimes both).

API Environment [Configure Environments](#)

Sandbox

POST /lang-detect/ [Try out](#) [Code Snippet](#)

detectLang

Parameters

Name Description

body \* required body

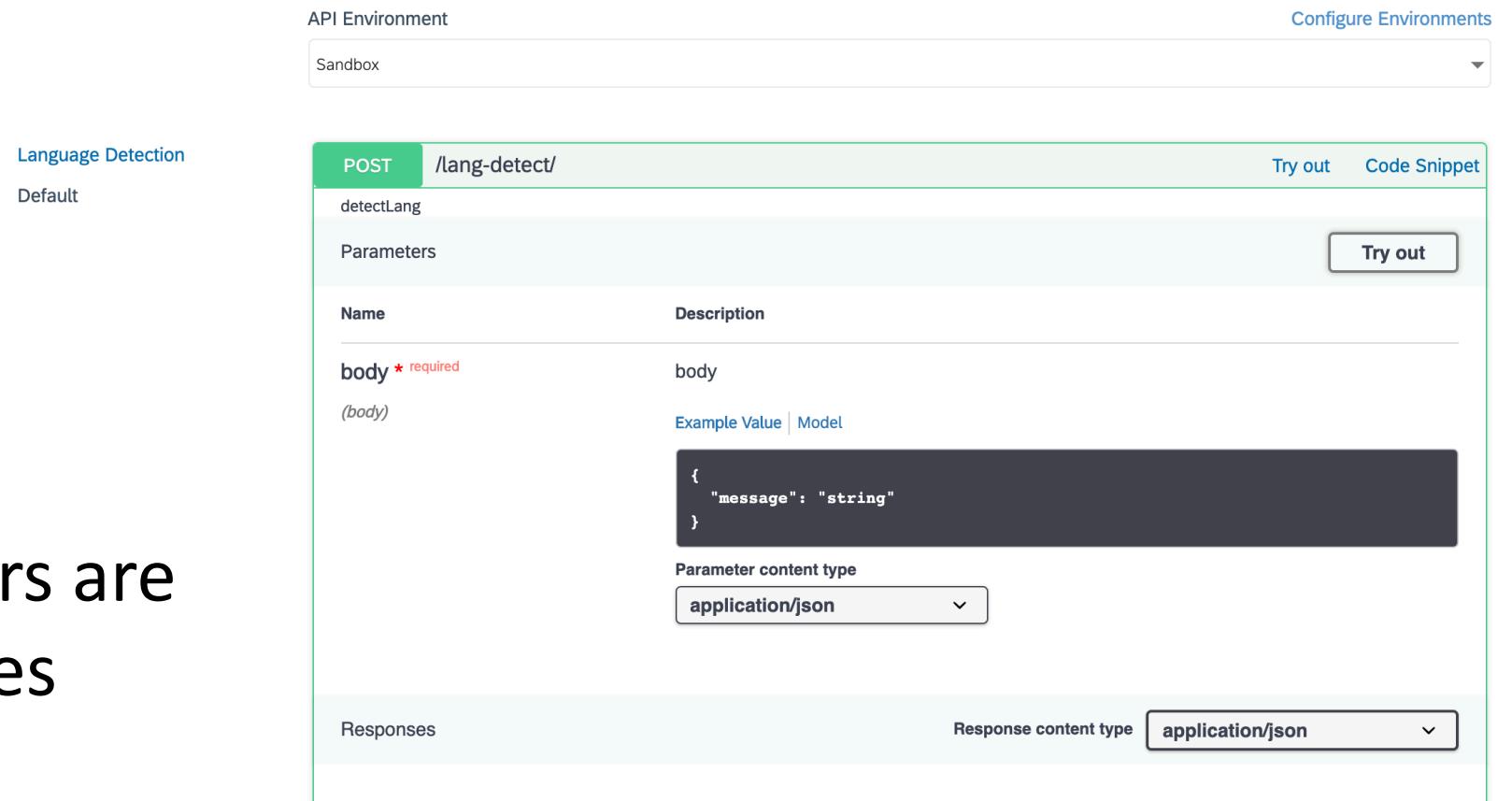
(body)

Example Value | Model

```
{  
  "message": "string"  
}
```

Parameter content type [application/json](#)

Responses Response content type [application/json](#)



# API Hub Preference

You API key can be seen in  
the code examples or under  
preferences.

Hello Peter!

You are logged in as I051802 (peter.snowdon@sap.com)

API Settings

An API sandbox has been created for you to try out APIs.

The use of these APIs with sandbox endpoints is protected by your personalized API key.

Show API Key

# Excercise: Postman

Note the environment

- Enter the API URL
- Enter the required headers
- Put the message as body
- Ensure you have the right method
- Hit send

The screenshot shows the Postman interface with a collection named 'LMLFoundation'. A 'Detect Language' request is selected. The 'Headers' tab is active, showing two headers: 'Content-Type: application/json' and 'APIKey: {{API\_KEY}}'. An arrow points from the 'Note the environment' text to the 'Send' button. Another arrow points from the 'Note the environment' text to the 'Body' tab, which is also highlighted in red.

Collection available in git:  
<https://github.com/drefter/ml-hackathon>

# Optional: Other Services

- You can try other services including image based services. You just need to read the doc and ensure you set the right settings:

The screenshot shows the Postman application interface. At the top, it displays a POST method and the URL <https://sandbox.api.sap.com/ml/api/v2alpha1/image/human-detection/>. Below the URL, there are tabs for Params, Authorization, Headers (2), Body (highlighted with a red underline), Pre-request Script, and Tests. Under the Body tab, there are five options: none, form-data (selected), x-www-form-urlencoded, raw, and binary. A table below shows two key-value pairs: 'file' with a checked checkbox and a 'Select Files' button, and 'Key' with a 'Value' field and a 'Description' column. At the bottom, there is a 'Response' section.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> file	Select Files	
Key	Value	Description

# Excercise: In Code (javascript)

## The basics:

- Make a POST call
- Ensure that you have a means of authenticating the call
- Send the right message.

```
/*
url: is the end point of the call
data: is the message to send (json assumed)
callback: to get the data backout
*/
function callLML(url, oData, callback){
  request.post({
    url: url,
    body: JSON.stringify(oData),
    headers: {
      "Content-Type": "application/json",
      "Accept": "application/json",
      "APIKey": APIKey
    }
  }, function optionalCallback(err, httpResponse, body) {
    if (err) {
      return console.error('upload failed:', err);
    }
    callback(body);
  });
}
```

# Git: <https://github.com/drefter/ml-hackathon>

Code Issues 0 Pull requests 0 Projects 0 Insights Settings

Branch: master [ml-hackathon](#) / [code-samples](#) / [node](#) / [langauge](#) /

Create new file Upload files Find file History

		Latest commit bed46d4 4 minutes ago
 drefter	oops	
..		
<a href="#">.env.example</a>	oops	4 minutes ago
<a href="#">index.html</a>	updating env and turning detect to index	8 minutes ago
<a href="#">jquery.js</a>	adding some files	7 days ago
<a href="#">language-mapping.csv</a>	updating everything that is relevant	18 hours ago
<a href="#">manifest.yaml</a>	updating everything that is relevant	18 hours ago
<a href="#">package.json</a>	updating everything that is relevant	18 hours ago
<a href="#">server.js</a>	oops	4 minutes ago

# Setup

- If you have git you can clone the repo or just download a zip.
- Install node: <https://nodejs.org/en/download/>
- Check npm: <https://www.npmjs.com/get-npm>
- Install required packages:

```
/Users/i051802/development/git/ml-hackathon/code-samples/node/langauge  
(base) C02YG2FHJHD3:langauge i051802$ npm install
```

# .env.example to .env

- Note that for the environment you will need to add some properties notable the APIKey above

```
1 PORT=1666
2 APIKey="xxxxxxxxxxxxxxxxxx"
3 clientID="xxxxxx-xxxxxx-xxxxxx-xxxxxx-"
4 clientSecret="xxxxxx-xxxxxx-"
5 authenticationURL="https://you.domain.url/oauth/token?grant_type=client_credentials"
```

# Launch the server.

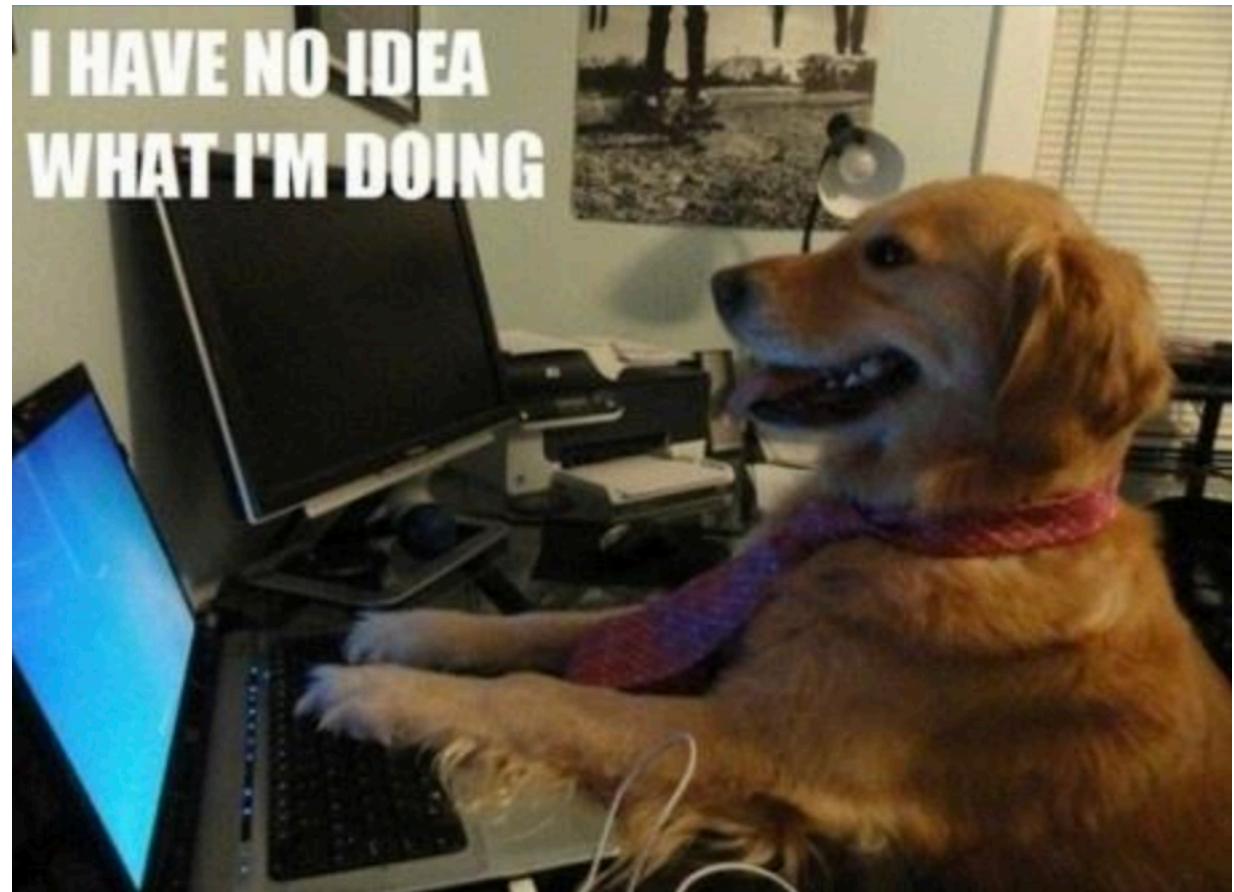
- **npm install:** installs required 3rd parties
- **node server.js (Or: npm start)**
  - Application runs on the port that you assigned in .env

```
[(base) C02YG2FHJHD3:ml-hackathon i051802$ cd code-samples/node/langauge/  
[(base) C02YG2FHJHD3:langauge i051802$ npm start  
  
> language-detection@1.0.0 start /Users/i051802/development/git/ml-hackathon/code-samples/node/langauge  
> node server.js  
  
Started dashboard dev server on port 1666
```

# The sample contains your challenge

If this isn't you then there are other code examples in the repo:

- Image comparison
- Text detection.



# Example: java

```
String url = "https://sandbox.api.sap.com/mlfs/api/v2/text/translation";

CloseableHttpClient httpClient = HttpClients.createDefault();
HttpPost uploadFile = new HttpPost(url);
uploadFile.addHeader("Accept", "application/json");
uploadFile.addHeader("APIKey", apiKey);

String data = createTranslationJSON("Can you translate this please?").toString();

StringEntity requestEntity = new StringEntity(data, ContentType.APPLICATION_JSON);
uploadFile.setEntity(requestEntity);

CloseableHttpResponse response = httpClient.execute(uploadFile);
HttpEntity responseEntity = response.getEntity();

JSONTokener tokener = new JSONTokener(responseEntity.getContent());
JSONObject results = new JSONObject(tokener);
```

# Example: python

```
import requests
import sys

if len(sys.argv) < 2:
    print("please pass in your apikey")
    exit();

# defining the api-endpoint
API_ENDPOINT = "https://sandbox.api.sap.com/ml/api/v2alpha1/text/lang-detect/"

# your API key here
APIKey = sys.argv[1];

# data to be sent to api
data = '{"message": "this is a string"}'

headers = {'APIKey': APIKey, "Content-Type" : "application/json"}

# sending post request and saving response as response object
r = requests.post(url = API_ENDPOINT, data = data, headers=headers)

# extracting response text
response = r.text
print("The response is:%s"%response)
```

# Using Production Accounts

- API hub gives you trial access to a subset of the LML APIs.
- To access everything you need a production account.
- And also API authentication.

# Authentication with Production

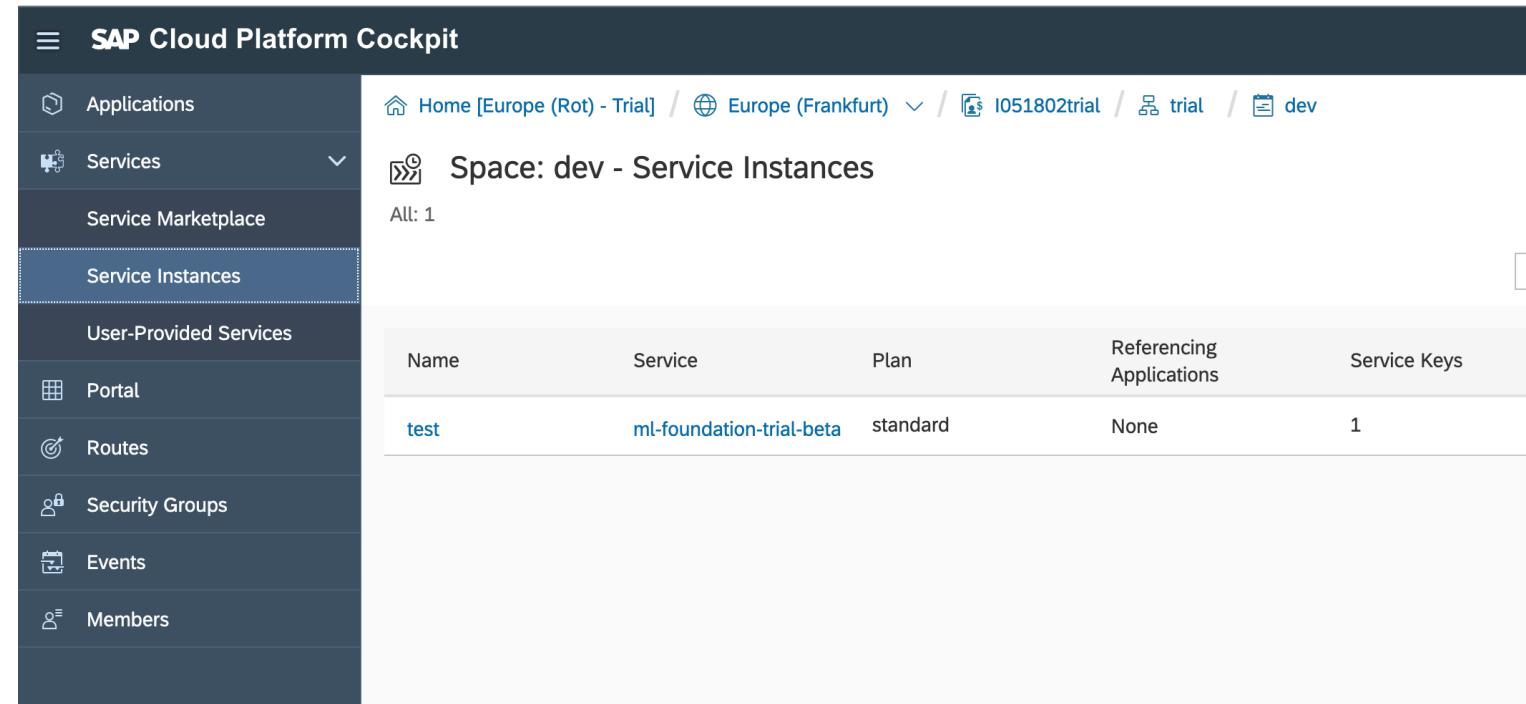
- With a production account you will need to authenticate using Oauth.
- You will need 3 values that can be found in your account.
  - clientID
  - clientSecret
  - authenticationURL

```
1 PORT=1666
2 APIKey="xxxxxxxxxxxxxxxxxx"
3 clientID="xxxxxx-xxxxxx-xxxxxx-xxxxxx-"
4 clientSecret="xxxxxx-xxxxxx-"
5 authenticationURL="https://you.domain.url/oauth/token?grant_type=client_credentials"
```

# CF Account

In your CF account you will either have or need to create a service instance.

Under that instance you will find your service variables like authentication and endpoints.



The screenshot shows the SAP Cloud Platform Cockpit interface. The left sidebar has a dark blue header with the SAP logo and the text "SAP Cloud Platform Cockpit". Below the header, the "Services" menu item is expanded, showing sub-options: "Service Marketplace", "Service Instances" (which is highlighted with a dotted border), "User-Provided Services", "Portal", "Routes", "Security Groups", "Events", and "Members". The main content area has a light gray background. At the top, there is a breadcrumb navigation: "Home [Europe (Rot) - Trial] / Europe (Frankfurt) / I051802trial / trial / dev". Below the breadcrumb, the title "Space: dev - Service Instances" is displayed, followed by "All: 1". A table lists one service instance:

Name	Service	Plan	Referencing Applications	Service Keys
test	ml-foundation-trial-beta	standard	None	1

# Cloud Foundry Deployment.

Returning to our Node example its possible to deploy to CF for example by creating a yaml file.

- cf login account
  - User/pass
- Select your space
- cf push

```
1  ---
2  applications:
3
4  - name: language-detection
5    host: language-detection
6    buildpack: nodejs_buildpack
7    memory: 512M
8    command: node server.js
9    instances: 1
10   random-route: true
11
```

# Hackathon Guidelines

- Theme: **Intelligent Enterprise**
- Scoring Guide:
  - Staying on topic.
  - Technical Merit.
  - Conceptual Brilliance.
- Note:
  - No presentations
  - ‘Working’ application
  - Using an ML API
  - More than one!!

**Strictly: 6 minutes per team.**

When you do nothing in the group project, but still you get +A.

#GameofThrones



#GameOfThronesFinale

