**LIVING PROGRESS - CROWDANALYTICS - PYTHON TEST HARNESS**

**Deployment Guide**

Revision History

|  |  |  |
| --- | --- | --- |
| **Author** | **Revision Number** | **Date** |
| TCSCODER | 1.0 | 05/30/2016 |

[1. Organization of Submission 3](#_Toc452401609)

[2. Prerequisites 3](#_Toc452401610)

[3. Run Mock API Python Script 3](#_Toc452401611)

[4. Run Test Harness Python Script 3](#_Toc452401612)

[5. Verification 4](#_Toc452401613)

[5.1 Test with primary\_test\_data 4](#_Toc452401619)

[5.2 Test with secondary\_test\_data 5](#_Toc452401620)

[5.3 Test with combined\_test\_data 6](#_Toc452401621)

[5.4 Test with batch\_test\_data 7](#_Toc452401622)

# 

# Organization of Submission

|  |  |
| --- | --- |
| **Directory / File Name** | **Description** |
| **doc** | It contains the deployment guide |
| **mock-api** | It contains the mock REST api python script. |
| **test-harness** | It contains the test harness python script. |

# Prerequisites

Make sure Python 2.7 and [Virtualenv](https://virtualenv.pypa.io/en/stable/) are installed.

# Run Mock API Python Script

Go to the mock-api sub-directory, and execute the following commands:

# create a virtual environment first

virtualenv .venv

# activate the virtual env

. .venv/bin/activate

# install dependencies to virtual env

pip install -r requirements.txt

# then start the REST server as below

python app.py dataFile [--port 5000] [--dataType paloalto]

Note that the script takes the following command line arguments:

|  |  |
| --- | --- |
| **Name** | **Description** |
| dataFile | The xlsx file containing the data that the mock api will reply in response. Required. It should be replaced with valid file name like: test\_files/primary\_test\_data.xlsx |
| --port | The listening port number. Default to 5000 if not specified. |
| --dataType | The endpoint data type. It can be either paloalto or chile. Default to paloalto. |

# Run Test Harness Python Script

Go to the test-harness sub-directory, and execute the following commands:

# create a virtual environment first

virtualenv .venv

# activate the virtual env

. .venv/bin/activate

# install dependencies to virtual env

pip install -r requirements.txt

# then run it as below

python score.py dataFile submissionsFile [--timeout 60] [--threads 5]

Note that the script takes the following command line arguments:

|  |  |
| --- | --- |
| **Name** | **Description** |
| dataFile | The xlsx file containing the data to send to the submissions’ API endpoints to categorize, as well as the expected results from the API endpoints. Required. It should be replaced with valid file name like: test\_files/primary\_test\_data.xlsx |
| submissionsFile | The csv file containing all the submission records. Required. It should be replaced with valid file name like: test\_files/submissions.csv. |
| --timeout | The timeout (in seconds) to receive the response from the submission’s API endpoint. Optional, it is default to 60 seconds. |
| --threads | The number of worker threads to calculate the scores for the submissions. Optional, it is default to 5 threads. |

# Verification

The mock-api directory contains the following files in the test\_files sub-directory:

* + - primary\_test\_data.xlsx
    - secondary\_test\_data.xlsx
    - combined\_test\_data.xlsx
    - batch\_test\_data.xlsx

The test-harness directory contains the following files in the test\_files sub-directory:

* + - primary\_test\_data.xlsx
    - secondary\_test\_data.xlsx
    - combined\_test\_data.xlsx
    - batch\_test\_data.xlsx
    - submissions.csv

The usage of these files will be explained in sections below.

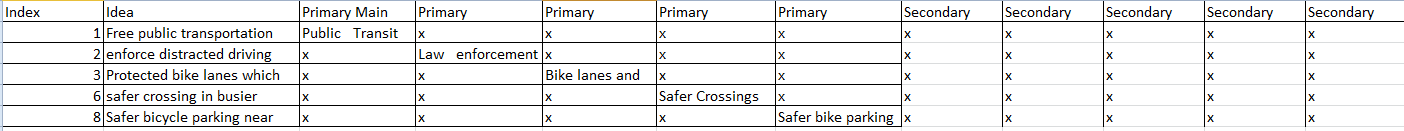


## Test with primary\_test\_data

In **mock-api** directory, run the script as below (the mock api will run locally on port number 5000):

python app.py test\_files/primary\_test\_data.xlsx

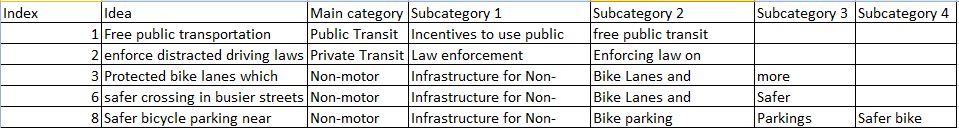
And the test data in the test\_files/primary\_test\_data.xlsx file is as below:



In **test-harness** directory, run the script as below to test the submisisons:

python score.py test\_files/primary\_test\_data.xlsx test\_files/submissions.csv

And the test data in the test\_files/primary\_test\_data.xlsx is as below:



The test data in the test\_files/submissions.csv is as below:



After the test is completed, the test\_files/submissions.csv will be changed as below:



Compare the two primary\_test\_data.xlsx files in mock-api and test-harness directories, you can see the category columns with ‘x’ won’t match, and the other category columns will match.

So for the submission (id = 11), it will get the following score:

* + - 1 point – for the primary main category match for record with index = 1
    - 1 point – for the primary subcategory 1 match for record with index = 2
    - 0.5 point – for the primary subcategory 2 match for record with index = 3
    - 0.25 point – for the primary subcategory 3 match for record with index = 6
    - 0.25 point – for the primary subcategory 4 match for record with index = 8

So in total, this submission will get 3 points for score. And as each primary category is correct once, so the accuracy is 20.00 percent for all of them.

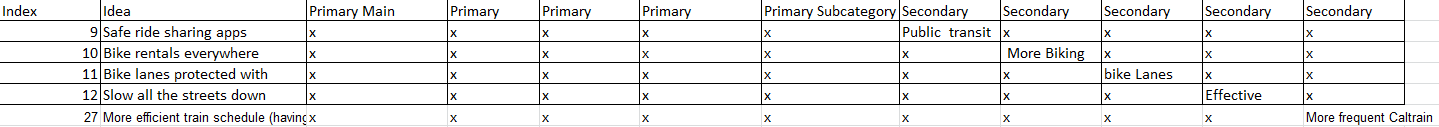
The endpoint of submission (id = 22) is not accessible, so it will get 0s for all them.

## Test with secondary\_test\_data

In mock-api directory, run the script as below (the mock api will run locally on port number 5000):

python app.py test\_files/secondary\_test\_data.xlsx

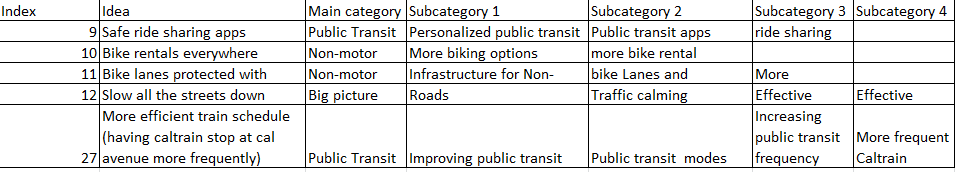
And the test data in the test\_files/secondary\_test\_data.xlsx file is as below:



In test-harness directory, run the script as below to test the submisisons:

python score.py test\_files/secondary\_test\_data.xlsx test\_files/submissions.csv

And the test data in the test\_files/secondary\_test\_data.xlsx is as below:



The test data in the test\_files/submissions.csv is as below:



After the test is completed, the test\_files/submissions.csv will be changed as below:



Compare the two secondary\_test\_data.xlsx files in mock-api and test-harness directories, you can see the category columns with ‘x’ won’t match, and the other category columns will match.

So for the submission (id = 11), it will get the following score:

* + - 0.5 point – for the secondary main category match for record with index = 9
    - 0.5 point – for the secondary subcategory 1 match for record with index = 10
    - 0.25 point – for the secondary subcategory 2 match for record with index = 11
    - 0.125 point – for the secondary subcategory 3 match for record with index = 12
    - 0.125 point – for the secondary subcategory 4 match for record with index = 27

So in total, this submission will get 1.5 points for score. And as no primary category is correct, so the accuracy is 0.00 for all of them.

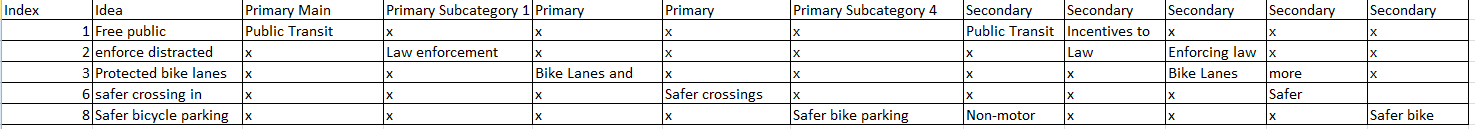
The endpoint of submission (id = 22) is not accessible, so it will get 0s for all them.

## Test with combined\_test\_data

In mock-api directory, run the script as below (the mock api will run locally on port number 5000):

python app.py test\_files/combined\_test\_data.xlsx

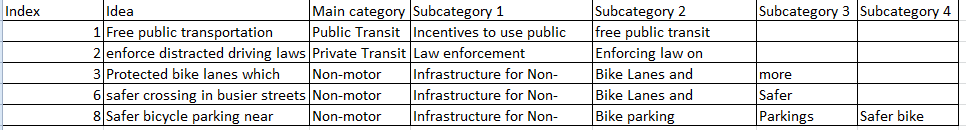
And the test data in the test\_files/combined\_test\_data.xlsx file is as below:



In test-harness directory, run the script as below to test the submisisons:

python score.py test\_files/combined\_test\_data.xlsx test\_files/submissions.csv

And the test data in the test\_files/combined\_test\_data.xlsx is as below:



The test data in the test\_files/submissions.csv is as below:



After the test is completed, the test\_files/submissions.csv will be changed as below:



Compare the two combined\_test\_data.xlsx files in mock-api and test-harness directories, you can see the category columns with ‘x’ won’t match, and the other category columns will match.

So for the submission (id = 11), it will get the following score:

* + - for record (index = 1): 1 (primary main category) + 0.5 (secondary subcategory 1)
    - for record (index = 2): 1 (primary subcategory 1) + 0.25 (secondary subcategory 2)
    - for record (index = 3): 0.5 (primary subcategory 2) + 0.125 (secondary subcategory 3)
    - for record (index = 6): 0.25 (primary subcategory 3) + 0.125 (secondary subcategory 4)
    - for record (index = 8): 0.25 (primary subcategory 4) + 0.5 (secondary main category)

So in total, this submission will get 4.5 points for score. And as each primary category is correct once, so the accuracy is 20.00 percent for all of them.

The endpoint of submission (id = 22) is not accessible, so it will get 0s for all them.

## Test with batch\_test\_data

In mock-api directory, run the script as below (the mock api will run locally on port number 5000):

python app.py test\_files/batch\_test\_data.xlsx

In test-harness directory, run the script as below to test the submisisons:

python score.py test\_files/batch\_test\_data.xlsx test\_files/submissions.csv

The batch\_test\_data.xlsx file contains 16 records, so the data will be sent in two batches to each submission’s api endpoint.