cloudmesh-analytics

Qiwei Liu, Yanting Wan

Nov 05, 2019

CONTENTS:

CHAPTER

ONE

CLOUDMESH PACKAGE

1.1 Subpackages

1.1.1 cloudmesh.analytics package

Subpackages

cloudmesh.analytics.api package

Submodules

cloudmesh.analytics.api.manager module

class cloudmesh.analytics.api.manager.**Manager**
Bases: object

list(*parameter*)

Module contents

cloudmesh.analytics.command package

Submodules

cloudmesh.analytics.command.analytics module

class cloudmesh.analytics.command.analytics.**AnalyticsCommand**
Bases: cloudmesh.shell.command.PluginCommand

do_analytics(*args*)

```
Usage:
      analytics server start [--cloud=CLOUD]
      analytics server stop [--cloud=CLOUD]

This command manages the cloudmesh analytics server on the given cloud.
If the cloud is not spified it is run on localhost
```

(continues on next page)

(continued from previous page)

```
Options:
    --clout=CLOUD  The name of the cloud as specified in the
                   cloudmesh.yaml file
```

Module contents

cloudmesh.analytics.server package

Submodules

cloudmesh.analytics.server.db module

cloudmesh.analytics.server.db.**close_db**(*e=None*)

cloudmesh.analytics.server.db.**get_db**()

cloudmesh.analytics.server.db.**init_app**(*app*)

cloudmesh.analytics.server.db.**init_db**()

cloudmesh.analytics.server.server module

To create a flask app

The method definition to create a flask app by call ing the create_app function

Example: create_app(test_config)

cloudmesh.analytics.server.server.**create_app**(*config=None*)

To create a flask app

Parameters **config** – A dictionary contains the configurations for the flask app

Returns A flask app object

Module contents

Submodules

cloudmesh.analytics.analytics module

The analytic functions The module include analytic functions, and are also the endpoints of the flask app. Those functions are referred by the OpenAPI specification by operationIDs

cloudmesh.analytics.analytics.**kmeans_fit**(*file_name*, *body*)

cloudmesh.analytics.analytics.**linear_regression**(*file_name*, *body*)

Linear regression.

Parameters

- **file_name** (*str*) – The file name that has the input data.
- **body** (*dict*) – The request body, which is a dictionary mapped by the connexion.

Returns Return an json objects.

> **Warning:** The input format should be specified

cloudmesh.analytics.analytics.**pca**()

cloudmesh.analytics.file module

File operations The module include file operations

cloudmesh.analytics.file.**list**()
List all uploaded files

cloudmesh.analytics.file.**read**(*file_name*)
Read files given a file name.

Parameters **file_names** – The input data source.

Returns Return a json response.

cloudmesh.analytics.file.**upload**(*file=None*)
Upload files to the server :param file: A file stream

Returns Return the file name if it success

> **Attention:** Only support the csv format now.

Raises **Raise an error message if the file format is not supported** –

cloudmesh.analytics.file_helpers module

The helper function isolates non-endpoint function from the file module

cloudmesh.analytics.file_helpers.**allowed**(*file_name*, *allowed_extentions*)
The allowed file extensions

Parameters

- **file_name** – The file name to check
- **allowed_extensions** – The allowed file extensions

Returns Return true or false

cloudmesh.analytics.file_helpers.**read_csv**(*file_name*)
Read csv using panda. The source path is relative and set when initializing flask app.

Parameters **file_name** – The file name to read

Returns A numpy array

cloudmesh.analytics.file_helpers.**save**(*file*)
Save file after securing the file name

Parameters **file** – the input data source

Returns Return a json response

1.2 Module contents

CHAPTER

TWO

TESTS PACKAGE

2.1 Submodules

2.2 tests.conftest module

The configuration for tests The config is required by the pytest. The pytest will run this file at first.

tests.conftest.**app**()
Configure the flask app for testing

This is a pytest fixture

> **Attention:** The the database is in progress, and not used. All files are saved in the folder defined in the 'UPLOAD_FOLDER' in the app configurations

> **Warning:** The uploaded folder is relative to where the pytest is called. Calling pytest in other folder will result a mis- placed uploaded folder. The uploaded folder should be kept under test directory

tests.conftest.**client**(*app*)
The test client for simulating requests

Returns Return a test client

tests.conftest.**runner**(*app*)

> **Attention:** Not used now

2.3 tests.test_cloudmesh module

Test the functions in *cloudmesh.analytics.analytics*

Tip: Running the test under the cloudmesh-analytics directory

```
`> ./cloudmesh-analytics$ pytest`
```

class tests.test_cloudmesh.**TestFileOperations**
Bases: object

Test file operations

Attention:

1. The function will be ran first and the files uploaded will be used for other tests
2. The uploaded file is insulated and saved in the testing_files directory as indicated in *tests.conftest*

post_file(*client*, *path*, *name*)
A helper function to make post request

Parameters

- **client** – The pytest fixture defined in *tests.conftest*
- **path** – The rest api defined in the yaml file
- **name** – the file name to post

Returns The data attribute of the flask response object

pytestmark = [Mark(name='first', args=(), kwargs={})]

test_format_error(*client*)
The upload will failed due to the txt file format. An error message will return

Parameters **client** – The pytest fixture defined in *tests.conftest*

Returns The data attribute of the flask response object, which is a binary string that includes a list of uploaded file names

test_read(*client*)
Test read uploaded file using the rest api

test_success_upload(*client*)
Test upload. The file will be uploaded in to the current directory named files

The test sample will use a empty csv file called test upload

Parameters **client** – The pytest fixture defined in *tests.conftest*

Returns The data attribute of the flask response object, which is a binary string that includes a list of uploaded file names

test_success_upload_dabetes(*client*)
Test upload. The file will be uploaded in to the current directory named files The test sample will use a empty csv file called test upload

Parameters **client** – The pytest fixture defined in *tests.conftest*

Returns The data attribute of the flask response object, which is a binary string that includes a list of uploaded file names

test_success_upload_sample(*client*)
Test upload. The file will be uploaded in to the current directory named files

The test sample will use a empty csv file called test upload

Parameters **client** – The pytest fixture defined in *tests.conftest*

Returns The data attribute of the flask response object, which is a binary string that includes a list of uploaded file names

class tests.test_cloudmesh.**TestKMeans**
Bases: object

test_errors(*client*)
Testing error arguments. The exception raised by the sci-kit learn will be returned in the error message. The exception also raised by the filename doesn't exist in app.config['UPLOAD_FOLDER'] :param client: :return:

test_kmeans_fit(*client*)

class tests.test_cloudmesh.**TestLinearRegression**
Bases: object

test_errors(*client*)
Testing error arguments. The exception raised by the sci-kit learn will be returned in the error message

Parameters **client** – The pytest fixture defined in *tests.conftest*

Returns The data attribute of the flask response object, which is a binary string that includes a list of uploaded file names

Note: The server will return the error message raised by the sci-kit learn

test_linear_regression(*client*)
Testing error arguments. The exception raised by the sci-kit learn will be returned

The data is taken from the sci-kit learn built in samples.

Parameters **client** – The pytest fixture defined in *tests.conftest*

Returns The data attribute of the flask response object, which is a binary string that includes a list of uploaded file names

Warning: Todo: The assertion may be false due to the floating number representaion in different word-size systems

tests.test_cloudmesh.**teardown_module**()
Teardown any state that was previously setup Remove the test_upload_folder by the end of tests

tests.test_cloudmesh.**test_run_pca**(*client*)

2.4 Module contents

CHAPTER

THREE

REQUIREMENTS

3.1 Local Cloudmesh Command

1. using cloumesh command to start and stop the remote server?
2. how to deploy the project?

3.2 TODO

1. A cloudmesh client will communicate with the server
2. A cloudmesh client is required
3. 12 functions are needed functionality
4. Exposing more service
5. Stop watch for testing
6. Set up docker and migrate to the cloud

2019.11.01. 11:19:18

1. use the command generate tool?
2. ![image-20191101112532678](/Users/qiweiliu/Library/Application Support/typora-user-images/image-20191101112532678.png)
3. migrate the analytics folder to somewhere else
    1. The folder structure

The goal is to generate open api and

```
cms analytics run linear-regression -data = file.csv -intercept=true
```

1. Match the function name and parameters
2. Call the function matching the user input
    1. Return error raised by sklearn

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

c

t

INDEX

U