data-loader-plugin

Release 0.0.1

Stanislav Khrapov

Jan 19, 2022

CONTENTS

CHAPTER

ONE

TABLE OF CONTENT (TOC)

Table of contents generated with markdown-toc

CHAPTER

TWO

OVERVIEW

The data loader plugin, aims at supporting running programs (*e.g.*, API service backends) when downloading data from cloud services such as AWS S3. It provides a base Python library, namely `data-loader-plugin`, offering a few methods to download data files from AWS S3.

CHAPTER

THREE

REFERENCES

3.1 Python module

- GitHub: https://github.com/cloud-helpers/python-plugin-data-loader/tree/master/data_loader_plugin
- PyPi: https://pypi.org/project/data-loader-plugin/
- Read the Docs (RTD): https://readthedocs.org/projects/data-loader-plugin/

3.2 Python virtual environments

- Pyenv and pipenv: http://github.com/machine-learning-helpers/induction-python/tree/master/installation/virtual-env

CHAPTER

FOUR

INSTALLATION

4.1 Clone this Git repository

```
$ mkdir -p ~/dev/infra && \
  git clone git@github.com:cloud-helpers/python-plugin-data-loader.git ~/dev/infra/
↪python-plugin-data-loader
$ cd ~/dev/infra/python-plugin-data-loader
```

4.2 Python environment

- If not already done so, install `pyenv`, Python 3.9 and, `pip` and `pipenv`
  - PyEnv:

```
$ git clone https://github.com/pyenv/pyenv.git ${HOME}/.pyenv
$ cat >> ~/.profile2 << _EOF

# Python
eval "\$(pyenv init --path)"

_EOF
$ cat >> ~/.bashrc << _EOF

# Python
export PYENV_ROOT="\${HOME}/.pyenv"
export PATH="\${PYENV_ROOT}/bin:\${PATH}"
. ~/.profile2
if command -v pyenv 1>/dev/null 2>&1
then
        eval "\$(pyenv init -)"
fi
if command -v pipenv 1>/dev/null 2>&1
then
        eval "\$(pipenv --completion)"
fi

_EOF
$ . ~/.bashrc
```

- Python 3.9:

```
$ pyenv install 3.9.8 && pyenv local 3.9.8
```

- pip:

```
$ python -mpip install -U pip
```

- pipenv:

```
$ python -mpip install -U pipenv
```

CHAPTER

FIVE

USAGE

5.1 Install the `data-loader-plugin` module

- There are at least two ways to install the `data-loader-plugin` module, in the Python user space with `pip` and in a dedicated virtual environment with `pipenv`.
  - Both options may be installed in parallel
  - The Python user space (typically, `/usr/local/opt/python@3.9` on MacOS or `~/.pyenv/versions/3.9.8` on Linux) may already have many other modules installed, parasiting a fine-grained control over the versions of every Python dependency. If all the versions are compatible, then that option is convenient as it is available from the whole user space, not just from this sub-directory
- In the remainder of that *Usage section*, it will be assumed that the `data-loader-plugin` module has been installed and readily available from the environment, whether that environment is virtual or not. In other words, to adapt the documentation for the case where `pipenv` is used, just add `pipenv run` in front of every Python-related command.

5.1.1 Install in the Python user space

- Install and use the `data-loader-plugin` module in the user space (with `pip`):

```
$ python -mpip uninstall data-loader-plugin
$ python -mpip install -U data-loader-plugin
```

5.1.2 Installation in a dedicated Python virtual environment

- Install and use the `data-loader-plugin` module in a virtual environment:

```
$ pipenv shell
(python-...-JwpAHotb) ✓ python -mpip install -U data-loader-plugin
(python-...-JwpAHotb) ✓ python -mpip install -U data-loader-plugin
(python-...-JwpAHotb) ✓ exit
```

5.2 Use `data-loader-plugin` as a module from another Python program

- Check the data file with the AWS command-line (CLI):

```
$ aws s3 ls --human s3://nyc-tlc/trip\ data/yellow_tripdata_2021-07.csv --no-sign-request
2021-10-29 20:44:34  249.3 MiB yellow_tripdata_2021-07.csv
```

- Module import statements:

```
>>> import importlib
>>> from types import ModuleType
>>> from data_loader_plugin.base import DataLoaderBase
```

- Create an instance of the DataLoaderBase Python class:

```
>>> plugin: ModuleType = importlib.import_module("data_loader_plugin.copyfile")
>>> data_loader: DataLoaderBase = plugin.DataLoader(
        local_path='/tmp/yellow_tripdata_2021-07.csv',
        external_url='s3://nyc-tlc/trip\ data/yellow_tripdata_2021-07.csv',
    )
>>> data_load_success, message = data_loader.load()
```

CHAPTER

SIX

DEVELOPMENT / CONTRIBUTION

- Build the source distribution and Python artifacts (wheels):

```
$ rm -rf _skbuild/ build/ dist/ .tox/ __pycache__/ .pytest_cache/ MANIFEST *.egg-info/
$ pipenv run python setup.py sdist bdist_wheel
```

- Upload to Test PyPi (no Linux binary wheel can be uploaded on PyPi):

```
$ PYPIURL="https://test.pypi.org"
$ pipenv run twine upload -u __token__ --repository-url ${PYPIURL}/legacy/ dist/*
Uploading distributions to https://test.pypi.org/legacy/
Uploading data_loader_plugin-0.0.1-py3-none-any.whl
100%|| 23.1k/23.1k [00:02<00:00, 5.84kB/s]
Uploading data-loader-plugin-0.0.1.tar.gz
100%|| 23.0k/23.0k [00:01<00:00, 15.8kB/s]

View at:
https://test.pypi.org/project/data-loader-plugin/0.0.1/
```

- Upload/release the Python packages onto the PyPi repository:
  - Register the authentication token for access to PyPi:

```
$ PYPIURL="https://upload.pypi.org"
$ pipenv run keyring set ${PYPIURL}/ __token__
Password for '__token__' in '${PYPIURL}/':
```

- Register the authentication token for access to PyPi:

```
$ pipenv run twine upload -u __token__ --repository-url ${PYPIURL}/legacy/ dist/*
Uploading distributions to https://upload.pypi.org/legacy/
Uploading data_loader_plugin-0.0.1-py3-none-any.whl
100%|| 23.1k/23.1k [00:02<00:00, 5.84kB/s]
Uploading data-loader-plugin-0.0.1.tar.gz
100%|| 23.0k/23.0k [00:01<00:00, 15.8kB/s]

View at:
https://pypi.org/project/data-loader-plugin/0.0.1/
```

- Note that the documentation is built automatically by ReadTheDocs (RTD)
  - The documentation is available from https://data-loader-plugin.readthedocs.io/en/latest/
  - The RTD project is setup on https://readthedocs.org/projects/data-loader-plugin/

- Build the documentation manually (with Sphinx):

```
$ pipenv run python setup.py build_sphinx
running build_sphinx
Running Sphinx v4.3.0
[autosummary] generating autosummary for: README.md
myst v0.15.2: ..., words_per_minute=200)
building [mo]: targets for 0 po files that are out of date
building [html]: targets for 1 source files that are out of date
updating environment: [new config] 1 added, 0 changed, 0 removed
reading sources... [100%] README
...
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
writing output... [100%] README
...
build succeeded.

The HTML pages are in build/sphinx/html.
```

- Re-generate the Python dependency files (`requirements.txt`) for the CI/CD pipeline (currently Travis CI):

```
$ pipenv --rm; rm -f Pipfile.lock; pipenv install; pipenv install --dev
$ git add Pipfile.lock
$ pipenv lock -r > ci/requirements.txt
$ pipenv lock --dev -r > ci/requirements-dev.txt
$ git add ci/requirements.txt ci/requirements-dev.txt
$ git commit -m "[CI] Upgraded the Python dependencies for the Travis CI pipeline"
```

6.1 Test the data loader plugin Python module

- Enter into the `pipenv` Shell:

```
$ pipenv shell
(python-...-iVzKEypY) ✓ python -V
Python 3.9.8
```

- Uninstall any previously installed `data-loader-plugin` module/library:

```
(python-...-iVzKEypY) ✓ python -mpip uninstall data-loader-plugin
```

- Launch a simple test with `pytest`

```
(python-iVzKEypY) ✓ python -mpytest tests
==================== test session starts ====================
platform darwin -- Python 3.9.8, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: ~/dev/infra/python-plugin-data-loader
plugins: cov-3.0.0
collected 3 items

```

(continues on next page)

(continued from previous page)

```
tests/test_copyfile.py .                                    [ 33%]
tests/test_s3.py ..                                         [100%]
======================== 3 passed in 1.22s ====================
```

- Exit the `pipenv` Shell:

```
(python-...-iVzKEypY) ✓ exit
```