Pset 3 - Cookiecutter

Due Mar 4 by 11:59pm **Points** 50 **Submitting** a website url

This component assignment is for your own cookiecutter fork. Accept the assignment here: **Link (https://classroom.github.com/a/NzNmp08z)**

We will customize our own cookiecutter repo, and using it to complete a Word2Vec project.

Note: This problem set will involve submitting two separate assignments:

(1) customized cookiecutter (here)
(2) pset 3 (released shortly)

Please submit both links to the Canvas assignments when you have completed!

Adapt cookiecutter-pylibrary for this course

In the last problem set, you used cookiecutter to create your pset-utils repository. If you recall, you made a bunch of changes to configurations (in .travis.yml, tox.ini, setup.py, etc.). However, if you were to use cookiecutter again for a new project, you would have to change all those configurations again!

Now we will work on modifying your own version of the cookiecutter repository, so that every time you create a project using the cookiecutter template, your preferred configurations are up-to-date.

Fork the cookiecutter-pylibrary repo into this classroom by accepting the assignment above, and clone it locally.

Be sure to push changes in the template to your fork. You may continue to develop that repository throughout the course. If you make any changes beyond those instructed, please ensure to mention them in the problem set you
are actively working on, and let us know if you add something awesome!

Note that the cloned repo has two directories with similar contents:

```
cookiecutter-csci-pset-GITHUBID/  <- repo root directory
    .travis.yml                      <- tests for the template itself!
    {{ cookiecutter.repo_name }}/ <- template directory
        .travis.yml                <- rendered into tests for new project
```

Settings

Just like you did in the previous problem set, inspect `cookiecutter.json`.

Change the defaults in this file to the ones listed below (from the previous problem set). NB: the first item in a list is the default. You can also remove other options if you'd like.

The value of this is that every time you render a new project for this class using this cookiecutter repo, you will not need to re-type these defaults. We are capturing our best practices in code!

Param	Value
name/email	Yours
github_username	`csci-e-29 (since the repo lives in the org)`
repo_name	`"2019sp-{{ cookiecutter.project_name|lower|replace (' ','-') }}-GITHUBID"`
test_runner	pytest
travis	yes
command line interface	argparse
codecov	no
requiresio	no

Misc changes

Note that this is similar to what you did for pset utils, but this time, it's in your cookiecutter template so that it is reproduced every time you make a new project.

Now, within the template folder `cookiecutter-csci-pset-<YOUR_GITHUB_ID>/{{cookiecutter.repo_name}}` (NOT in the root directory) **do the following**:

1. Clear the contents of the LICENSE file
2. In setup.py:
    1. Delete the whole `project_urls` object
    2. python_requires='>=3.6'
3. In setup.cfg:
    1. Add {{ cookiecutter.package_name }} to the testpaths
    2. Add following to addopts (ignore the leading 1 below):
        1. --cov={{ cookiecutter.package_name }}
           --cov-branch
           --pyarg
4. Delete .travis.yml
5. Add "exclude README.md" to MANIFEST.in

Travis config

Our template is designed for libraries, not apps. Let's update the `.travis.yml` to look more like what we expect for a pset application.

Replace the existing `.travis.yml` in the ***template*** with the one provided in Pset 1. However, you should replace all mentions of 'pset 1' with the appropriate cookiecutter template variable!

Modify the relevant lines under `stage: answers`:

```
# Delete line about s3 download
- python3 -m {{ cookeicutter.package_name }}
```

Your new travis builds should have a 'tests' and 'answers' stage just like Pset 1

Pipfile

The following will require you to setup your github token

Let's add a Pipfile to the template, similar to what we did with Pset 1:

```
[[source]]
name = "pypi"
url = "https://pypi.org/simple"
verify_ssl = true
```

```
[dev-packages]
pytest = "*"
pytest-cov = "*"

[packages]
{{ cookiecutter.package_name }} = {editable = true,path = "."}
pset-utils = {editable = true,git = "https://github.com/csci-e-29/2019sp-pset-utils-GITHUBID"}

[requires]
python_version = "3.7"
```

Note that we ***do not add the Pipfile.lock***. This should be generated when you actually use a rendered template later! Otherwise, we would be locking old versions.

You may manually add to this over time, or add conditional installs as you want.

Docker (optional)

You will need to ensure your docker image contains the github token as instructed in **Pset Utils**.

Follow the instructions in Piazza (**https://piazza.com/class/jqhidqternh3sq?cid=359 (https://piazza.com/class/jqhidqternh3sq?cid=359)**) to modify your Dockerfile from **Pset 1** and the setup.py in your template. Be sure to replace all references to pset_utils with the appropriate cookiecutter variable, eg `{{ cookiecutter.package_name }}`.

Note that you need to 'bootstrap' the Pipfile.lock in new rendered templates. To do so, after rendering the template, comment out all lines in your Dockerfile starting with `COPY Pipfile .`. Build the docker image, then run:

```
./drun_app pipenv install
```

Restore the commented lines in the Dockerfile, and you should be able to rebuild your docker image and proceed normally.

If you desire, you could take this lockfile, template out the part with the name of the rendered package, and commit it back to the template itself to avoid bootstrapping in the future.

Push and Test

Ensure your changes are pushed to your cookiecutter repo. The tests on the template are extensive - we have commented out most of the matrix in the root level travis file to avoid bottlenecks for this course. You may uncomment some of the tests if you desire to use a particular feature of the template.

You should try rendering your project a few times and building the resulting environment to ensure it works.

Note you can use Docker to ensure you have appropriately set up your github token, even before pushing a rendered template to Travis.

After completing all steps, with default settings, you should be able to run something like this (I added nameless/tests just to check test paths were working):

```
$ pipenv run pytest
============================================================ test session starts ============================
============================
platform darwin -- Python 3.7.2, pytest-4.3.0, py-1.8.0, pluggy-0.8.1
rootdir: /Users/n0311699/repositories/psets/2019sp-nameless-gorlins, inifile: setup.cfg
plugins: cov-2.6.1
collected 2 items

tests/test_nameless.py .
[ 50%]
src/nameless/tests.py .
[100%]

---------- coverage: platform darwin, python 3.7.2-final-0 -----------
Name                        Stmts   Miss Branch BrPart     Cover   Missing
-------------------------------------------------------------------------
src/nameless/__init__.py        1      0      0      0   100.00%
src/nameless/__main__.py        3      1      2      1    60.00%   14, 13->14
src/nameless/cli.py             6      0      0      0   100.00%
src/nameless/tests.py           4      0      0      0   100.00%
-------------------------------------------------------------------------
TOTAL                          14      1      2      1    87.50%
```

```
========================================================= 2 passed in 0.06 seconds ========================
============================
```

Rubric			
Criteria	**Ratings**		**Pts**
Cookiecutter template modifications	**50.0 pts** **Full Marks**	**0.0 pts** **No Marks**	50.0 pts
			Total Points: 50.0