MSBA Capstone Class (BMKT 699) – Code Submissions

Your Name: Adam Hunter

GitHub Repository Link: <https://github.com/hunterad93/firenet>

Link to Your Style Guide of Choice: [PEP 8](https://peps.python.org/pep-0008/)

Directions:

1. Please choose five sections of code to demonstrate good coding principles and good documentation practices that are consistent with your code style guide of choice.
2. Try to keep your submitted code sections 25 lines or shorter, unless you have a very good reason for submitting a longer section.
3. Each code section should illustrate a different aspect of good coding principles and/or good documentation practice.
4. Include the information and explanation for each code section in the fields below.

There are a lot of folders in my github, and a bunch of things I called main.py because it made it more simple for the google cloud function system to recognize the ‘entry point’ for each app. But to make things more clear I am including a link to each script with the code section im referencing.

**Code Sections**

(1)

* File name:
  + [viirs\_upload\_cloud\_func/main.py](https://github.com/hunterad93/firenet/blob/main/src/prediction_instance/viirs_upload_cloud_func/main.py)
* Starting code line:
  + 194
* Ending code line:
  + 200
* What coding principle or documentation practice is this section intended to illustrate?
  + This section illustrates separation of concerns, security, and reusability. By passing in a ‘bbox’ (bounding box) argument from the google cloud scheduler cron job that calls this cloud function, it is easy to alter the geographic bounding box for this function without needing to touch the codebase. I think this is especially useful for working with cloud functions where updating them in even a small way can sometimes take a while. By similarly passing in the API key this way, the API key is kept out of the code base.

(2)

* File name:
  + [viirs\_upload\_cloud\_func/main.py](https://github.com/hunterad93/firenet/blob/main/src/prediction_instance/viirs_upload_cloud_func/main.py)
* Starting code line:
  + 115
* Ending code line:
  + 119
* What coding principle or documentation practice is this section intended to illustrate?
  + This section of code illustrates clear documentation using ‘docstring conventions’ which I used to document most of my functions. I think the commenting of this function and other functions in this script are very clear. I believe I would be able to come back to this long after working on it and understand how these functions fit together.

(3)

* File name:
  + [unet\_upload\_cloud\_func/main.py](https://github.com/hunterad93/firenet/blob/main/src/prediction_instance/unet_upload_cloud_func/main.py)
* Starting code line:
  + 1
* Ending code line:
  + 3
* What coding principle or documentation practice is this section intended to illustrate?
  + This section illustrates modularity of code. In this section I am importing functions from separate .py files where I defined them to help keep things modular and clearly organized. These functions were for the cloud function that ran the neural net for fire classification. I created three sets of functions in ‘preprocessing.py’, ‘neuralnet\_processing.py’, and ‘post-processing.py’ which were all used in sequence.

(4)

* File name:
  + [goes\_upload\_cloud\_func/main.py](https://github.com/hunterad93/firenet/blob/main/src/prediction_instance/goes_upload_cloud_func/main.py)
* Starting code line:
  + 1
* Ending code line:
  + 25
* What coding principle or documentation practice is this section intended to illustrate?
  + Importing modules on separate lines, and consistently using lowercase\_with\_underscores for function names and variable names all improves readability.

(5)

* File name:
  + [firenet\_flask\_server/blob/main/main.py](https://github.com/hunterad93/firenet_flask_server/blob/main/main.py)
* Starting code line:
  + 18
* Ending code line:
  + 22
* What coding principle or documentation practice is this section intended to illustrate?
  + In this code section I define a dictionary where the keys are the bigquery table names, and the values are the field names that hold the geojson to upload to the leaflet map. Setting up things this way with a function that could utilize this dictionary made it easy to scale my project. This really helped when I added the ‘MODIS’ classifications after the conversation with Corinne. I was mostly able to reuse code, create a new table in bigquery, and update this dictionary to reflect the new table name and it displayed the classifications in the flask server.