**Final: Boston Air Quality Analysis (Hongwei’s Part)**

Team 3 in INFO 7390

Team Members: Hongwei Hu, Guangnan Liang

Note: Red lines mean Very Important information.

Hongwei’s Work:

* Data Fetching & Upload Automation
* Azure Machine Learning Model Building & Publishing
* Pipelines for Regular Fetching & Retraining
* Deployment: Docker, Flask API…

GitHub: <https://github.com/Besimilar/Air-Quality-Analysis>

Deployment:

1. Data Ingestion & Wrangling
   * Docker Hub: besimilar/advanced-data-analysis:aqi
   * Download Link: <https://s3-us-west-2.amazonaws.com/team3assignmentfinal/boston-aqi.tar>
2. EDA & Geospatical Analysis
   * Docker Hub: besimilar/advanced-data-analysis:aqi
   * Download Link: <https://s3-us-west-2.amazonaws.com/team3assignmentfinal/boston-aqi-eda.zip>
   * My teammate has not sent all Jupyter notebooks to me, I cannot package them into docker images or upload to S3. The link above won’t work.
3. Pipelines
   * Docker Hub: besimilar/advanced-data-analysis:pipeline
   * Download Link: <https://s3-us-west-2.amazonaws.com/team3assignmentfinal/boston-aqi-pipeline.tar>
4. Model
   * Azure Gallery: <https://gallery.cortanaintelligence.com/Experiment/Boosted-Decision-Tree>
5. FLASK API
   * Pythonanywhere: <http://besimilar.pythonanywhere.com/prediction>
   * My teammate has not finished FLASK API part, so I have to deploy previous FLASK API as an example of deployment.

Resources:

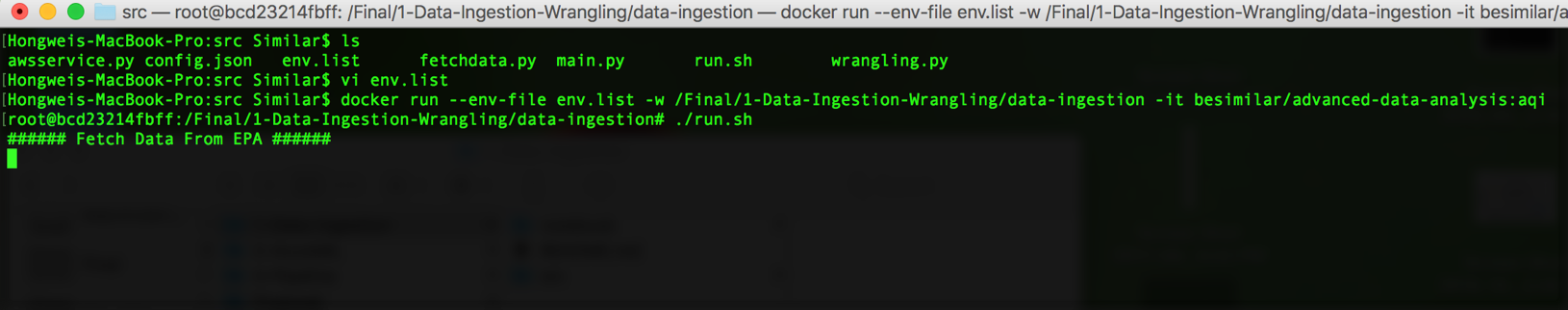
1. Data source: EPA (United States Environmental Protection Agency)
   * Outdoor Air Quality Data:

* url: <https://www.epa.gov/outdoor-air-quality-data/download-daily-data>
* api: <https://aqs.epa.gov/api>

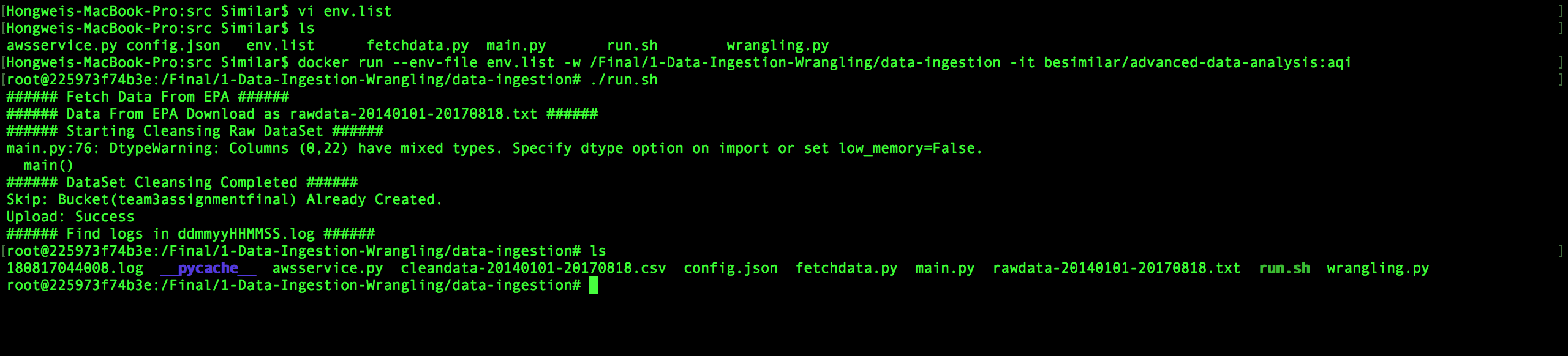
1. Model: Azure Machine Learning, Azure Gallery
2. Pipeline: Celery
3. Flask API deployment: <https://www.pythonanywhere.com/>

**Part 1: Data Ingestion & Wrangling Automation**

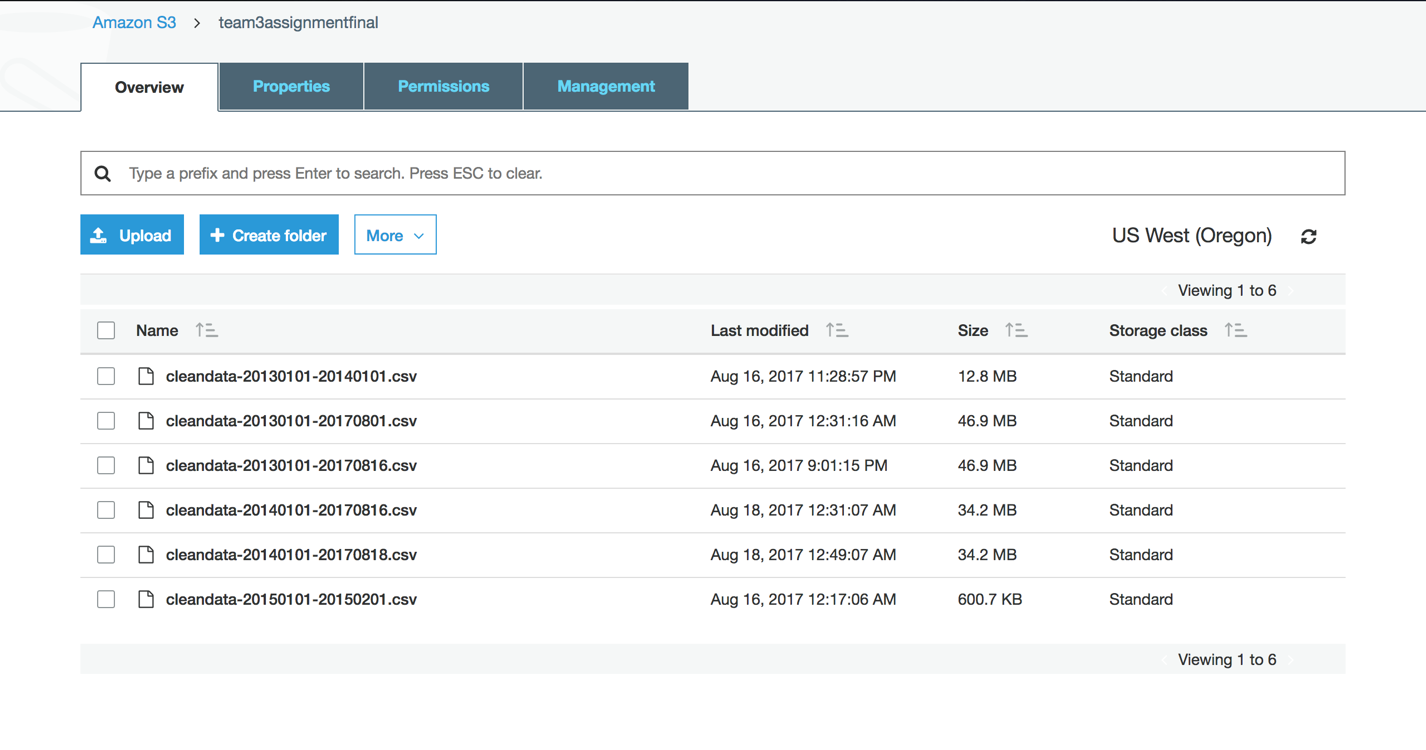
1. Source:
   1. Docker Hub: besimilar/advanced-data-analysis:aqi
   2. GitHub: <https://github.com/Besimilar/Air-Quality-Analysis/tree/master/1-Data-Ingestion>
2. What I did in this part:
   1. Fetch rawdata from EPA API ("src/fetchdata.py"):



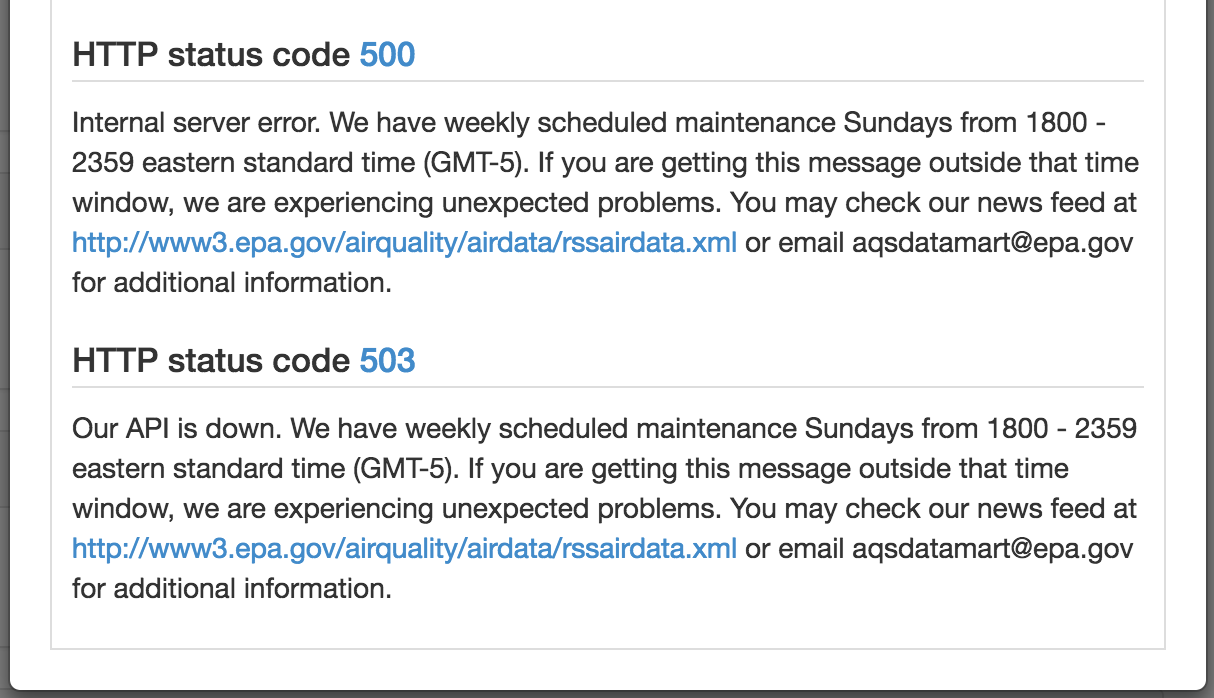
* 1. Clean rawdata and save both rawdata and cleandata to local ("src/wrangling.py")



* 1. Upload cleandata to AWS S3 ("src/awsservice.py"):



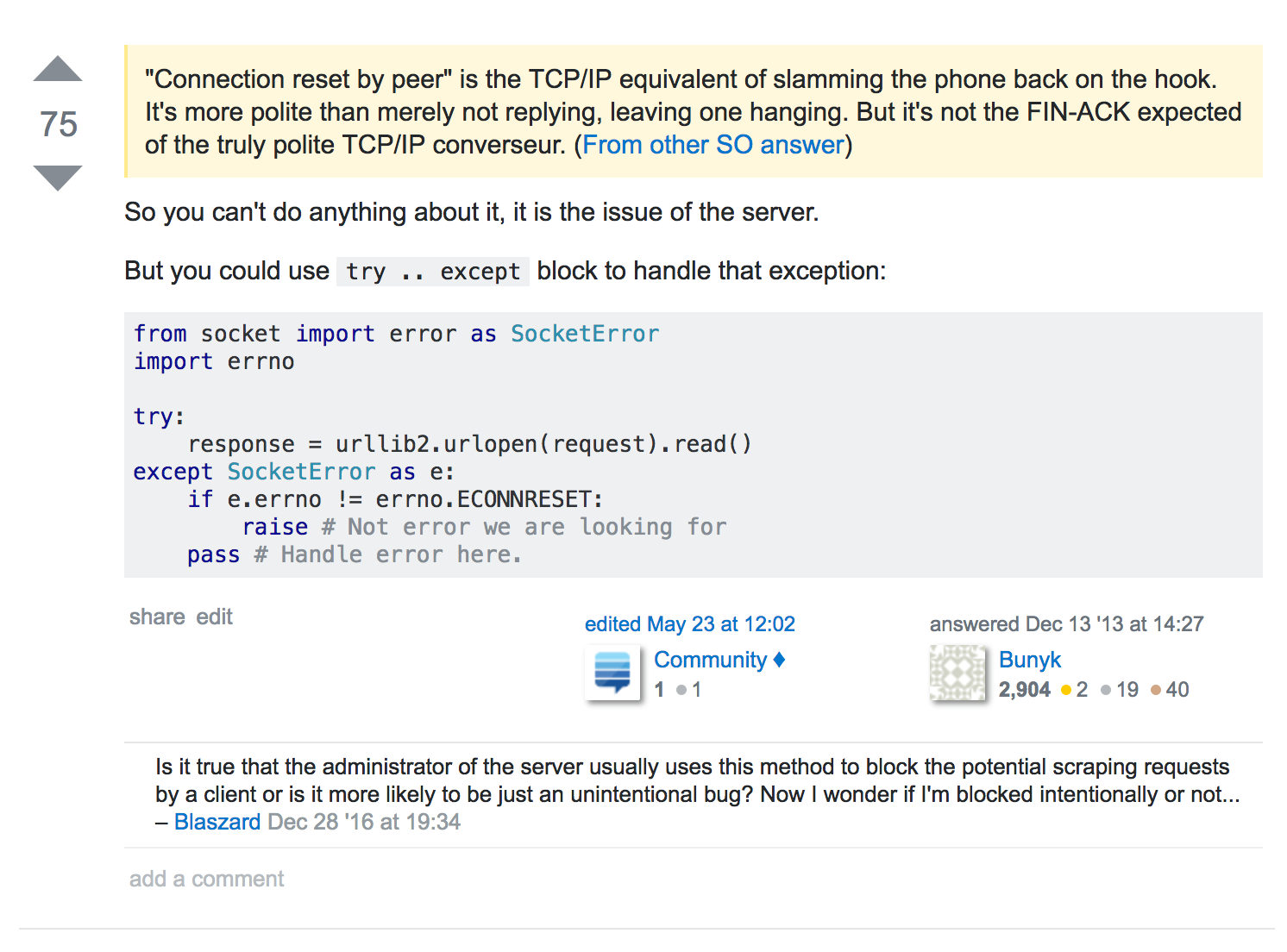
1. **Instruction**:
2. Maintenance Issue:
   1. The EPA API won't work on Sunday from 1800 to 2359. So avoid running this project during that time.
   2. refer to <https://aqs.epa.gov/aqsweb/documents/ramltohtml.html>



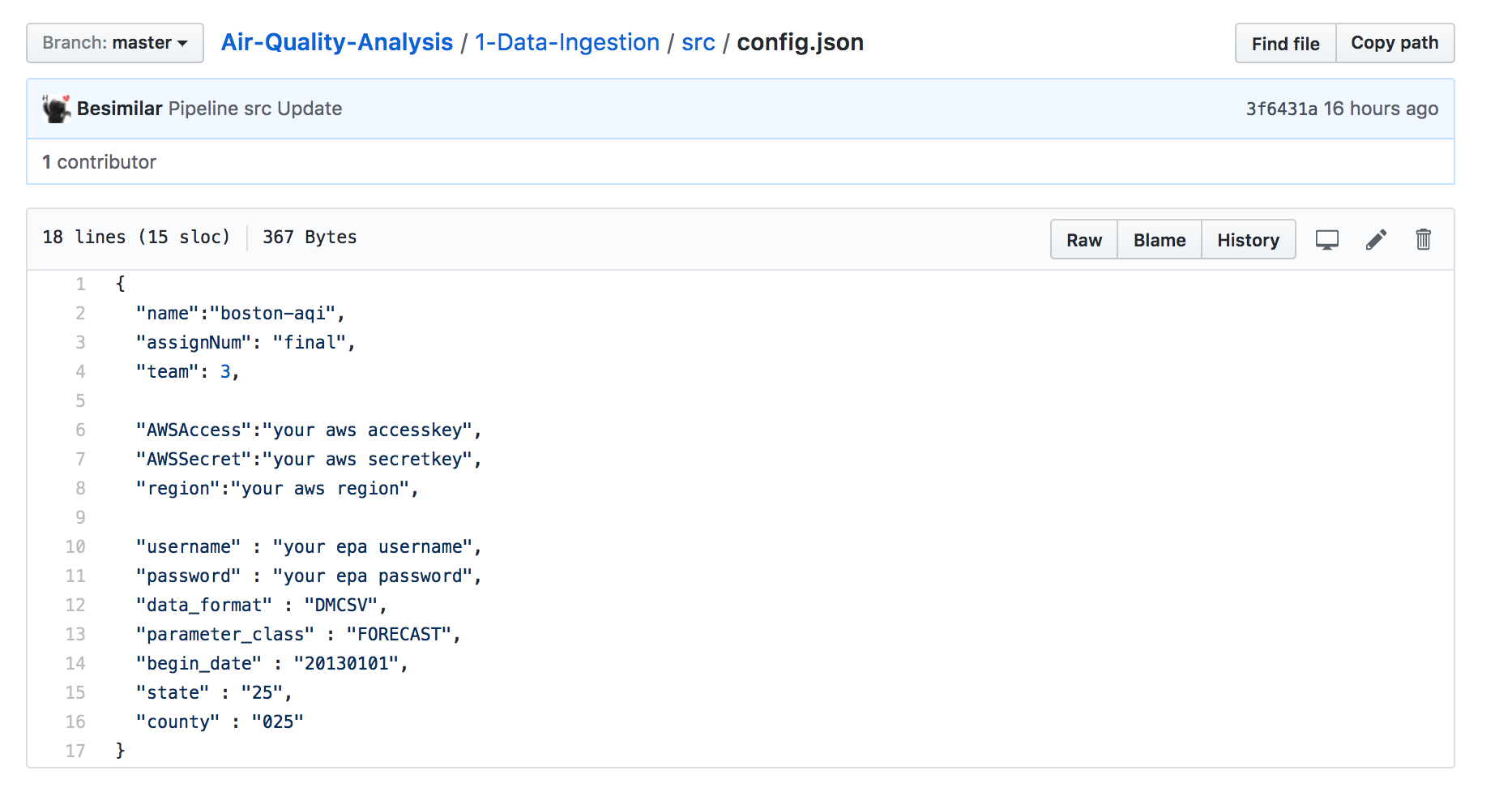
1. **Most Import Step Before starting your container**
   1. prepare a env.list file for env variables in container
   2. refer to "env.list"



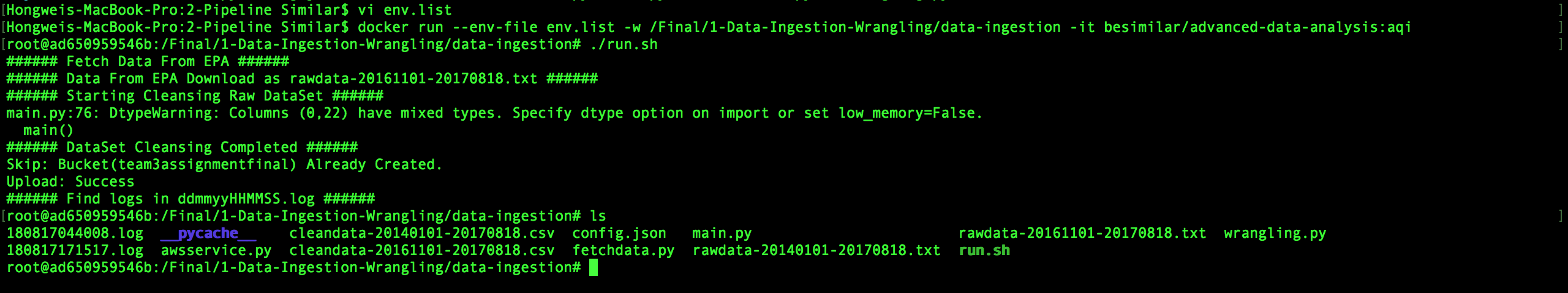
* + 1. AWS params:
       1. AWSACCESS: aws access key
       2. AWSSECRET: aws private key
       3. REGION: aws region
    2. EPA API params:
       1. EPAUSERNAME: EPA API username
       2. EPAPASSWORD: EPA API password
       3. BDATE: the first day of dataset (better within 3 years, try to set around 20140101, its a bug-free choice)
          1. You might get error, if you fetch large data from EPA api, because they seem to limit my account, refer to "Notes/1-Bug-EPA"



* + - * 1. You also might get error, if you fetch small data, because the data might not contain some predictive features, either wrangling or model retraining could produce "no features" errors.
      1. EDATE: the last day of dataset: (default: current date)
    1. Azure params:
       1. ACCOUNTNAME: azure storage account name
       2. ACCOUNTKEY: azure storage primary key
       3. CONTAINERNAME: azure storage container name
    2. Azure ML params:
       1. APIKEY: predictive service key
       2. APIURL: predictive service url
    3. Other default params:
       1. change them in "config.json" After starting container
       2. You dont't need to change them to do a demo



1. Start Container and set env:
   1. $ docker run --env-file env.list -w /Final/1-Data-Ingestion-Wrangling/data-ingestion -it besimilar/advanced-data-analysis:aqi
2. Run job in container:
   1. $ ./run.sh
3. **Screenshot for all steps above: You could do the same as below to do a demo for this part.**

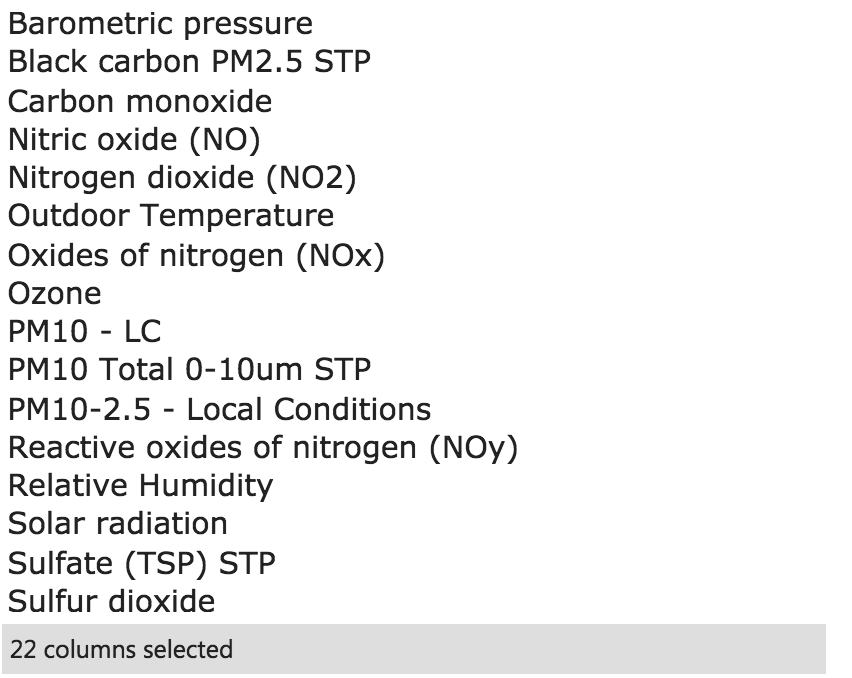


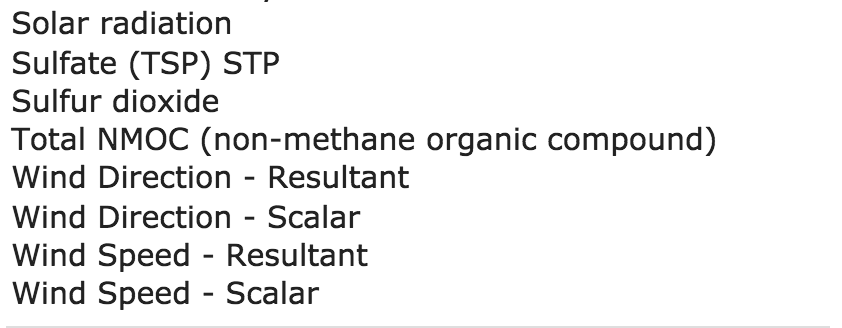
**Part 2: Azure Machine Learning Model Building & Publishing**

1. Azure Gallery: <https://gallery.cortanaintelligence.com/Experiment/Boosted-Decision-Tree>
2. PM2.5 AQI Prediction:
   1. Y value:

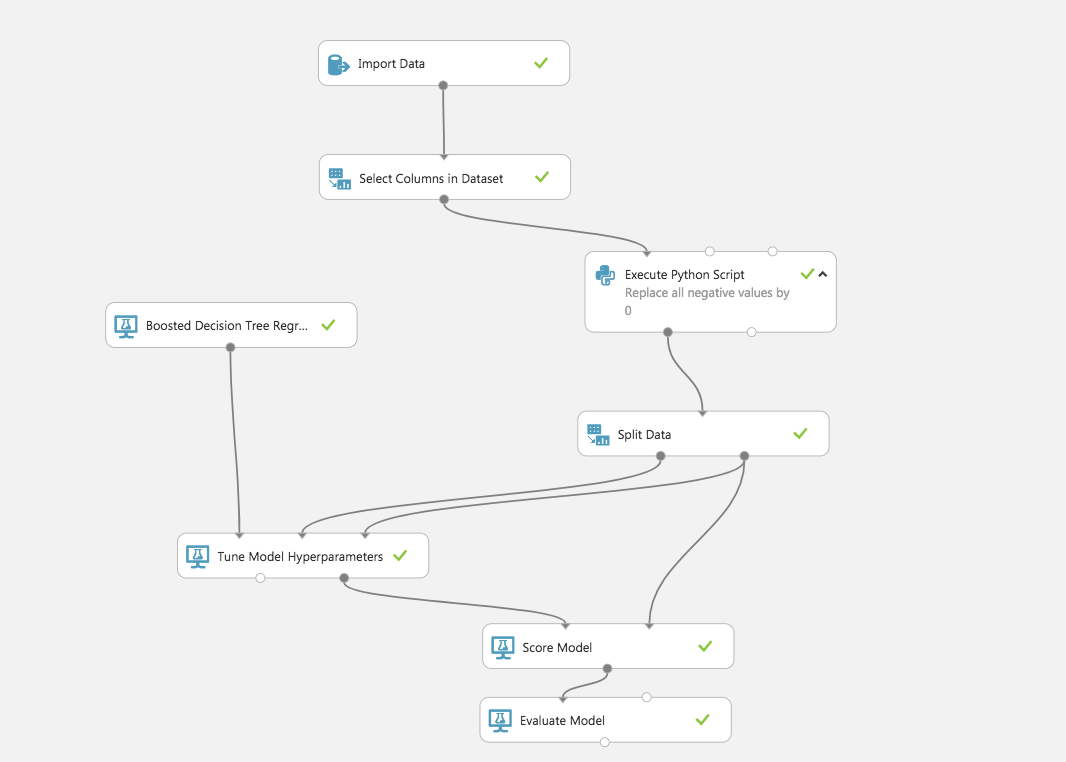


* 1. X values:

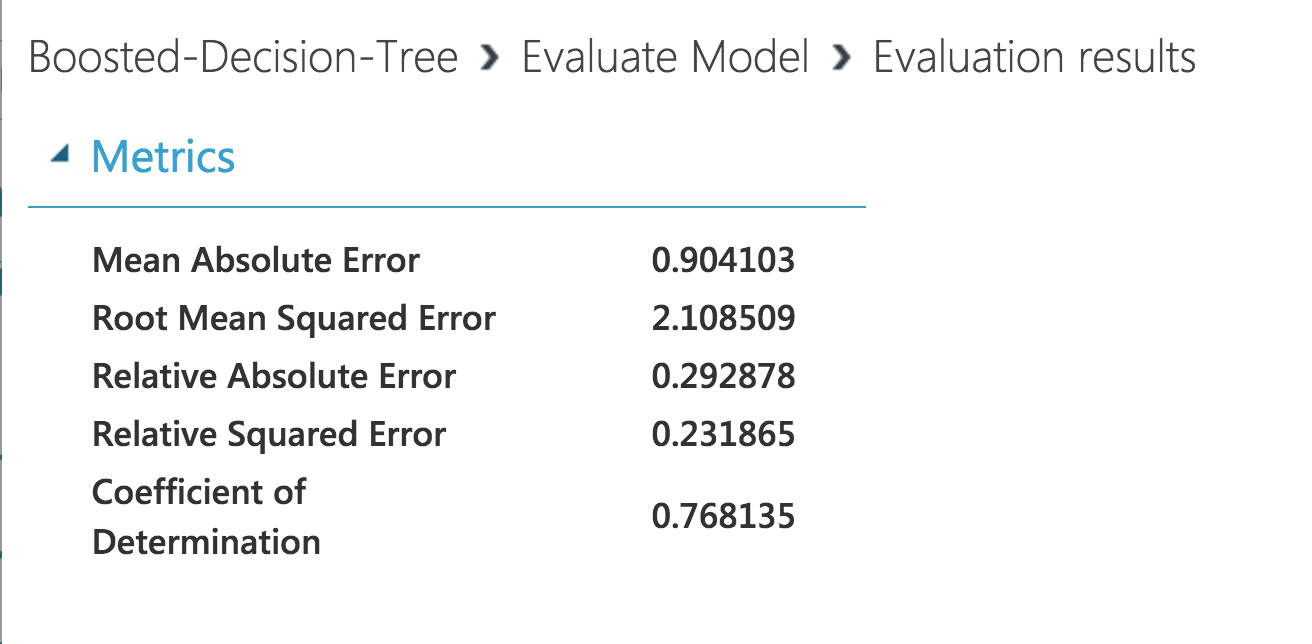




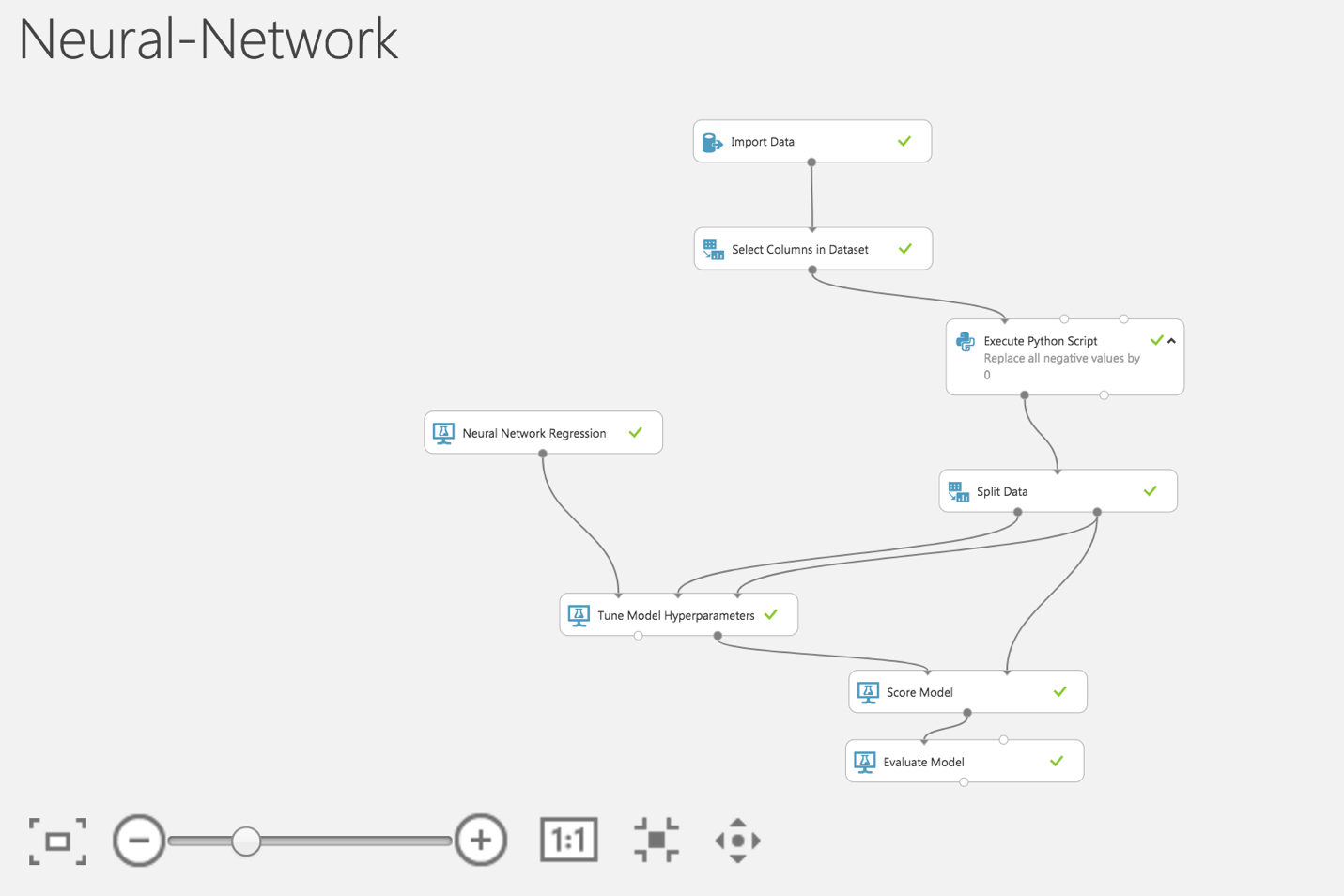
1. Algorithms:
   1. Boosted Decision Tree
      1. Model:



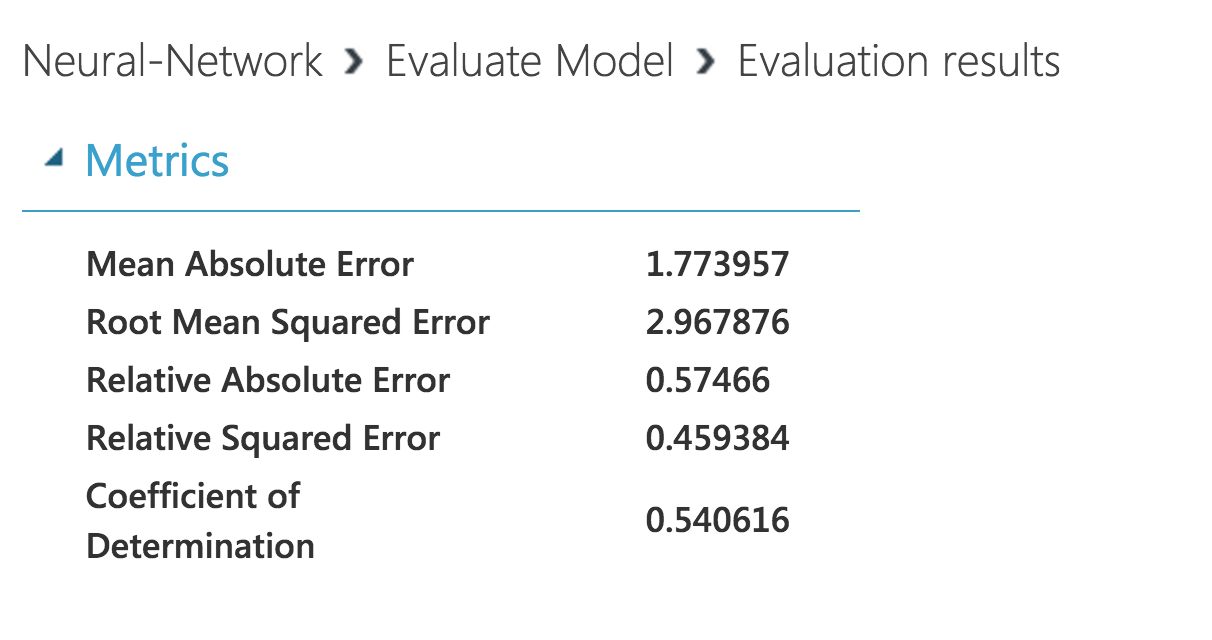
* + 1. Result:



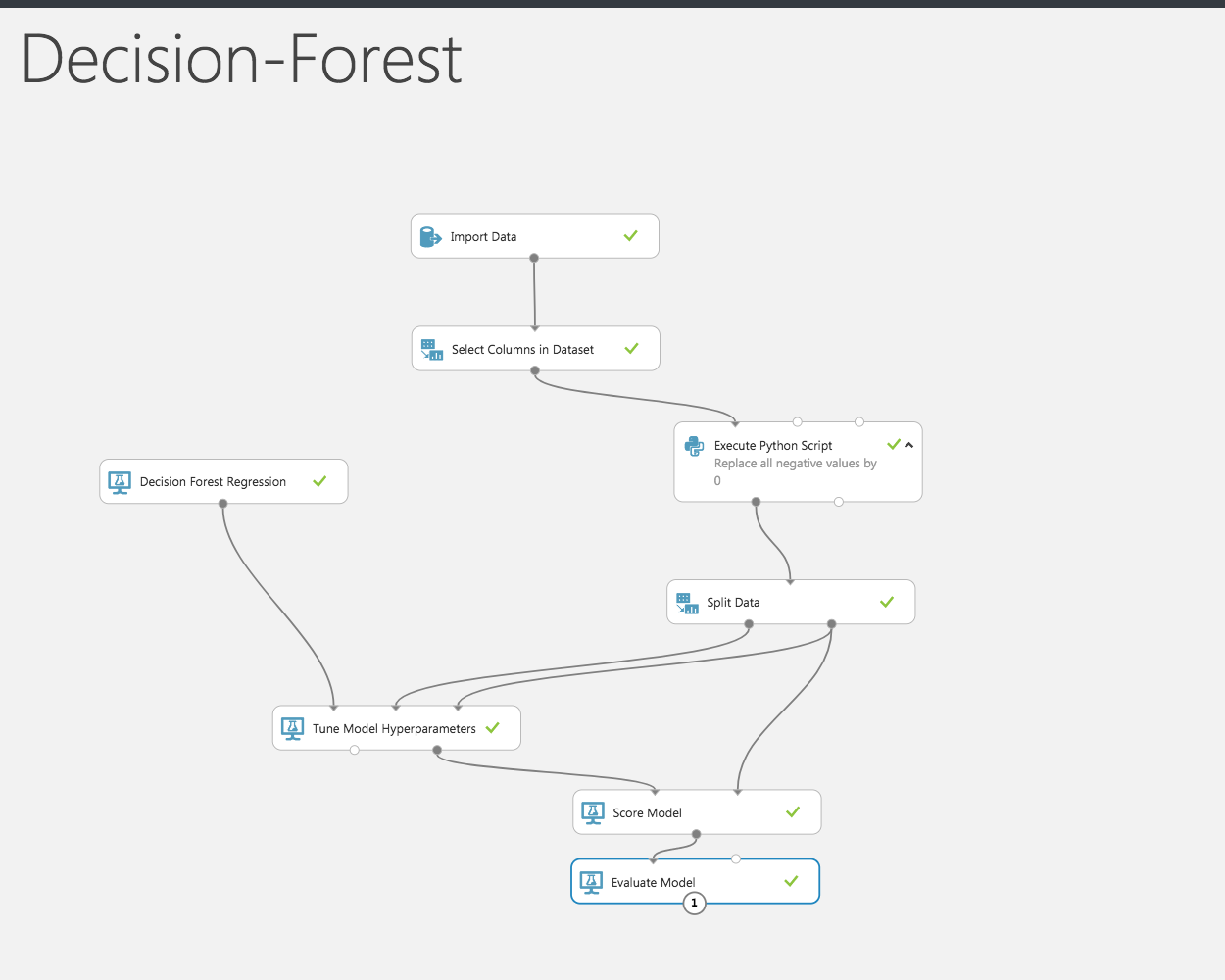
* 1. Neural Network
     1. Model



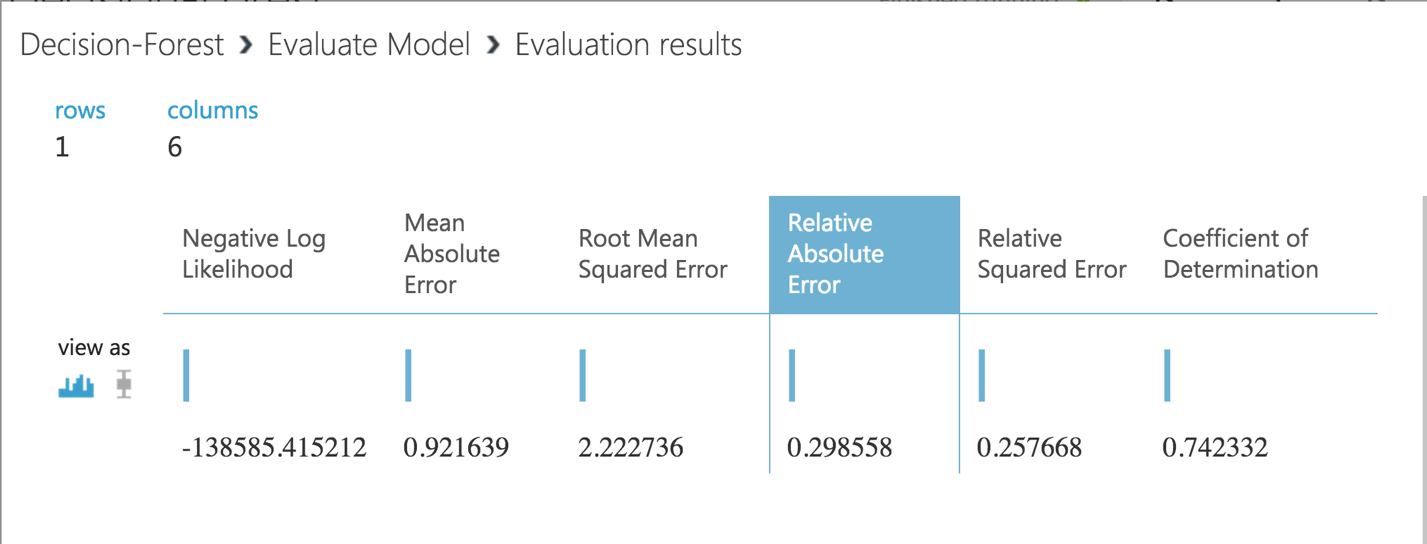
* + 1. Result



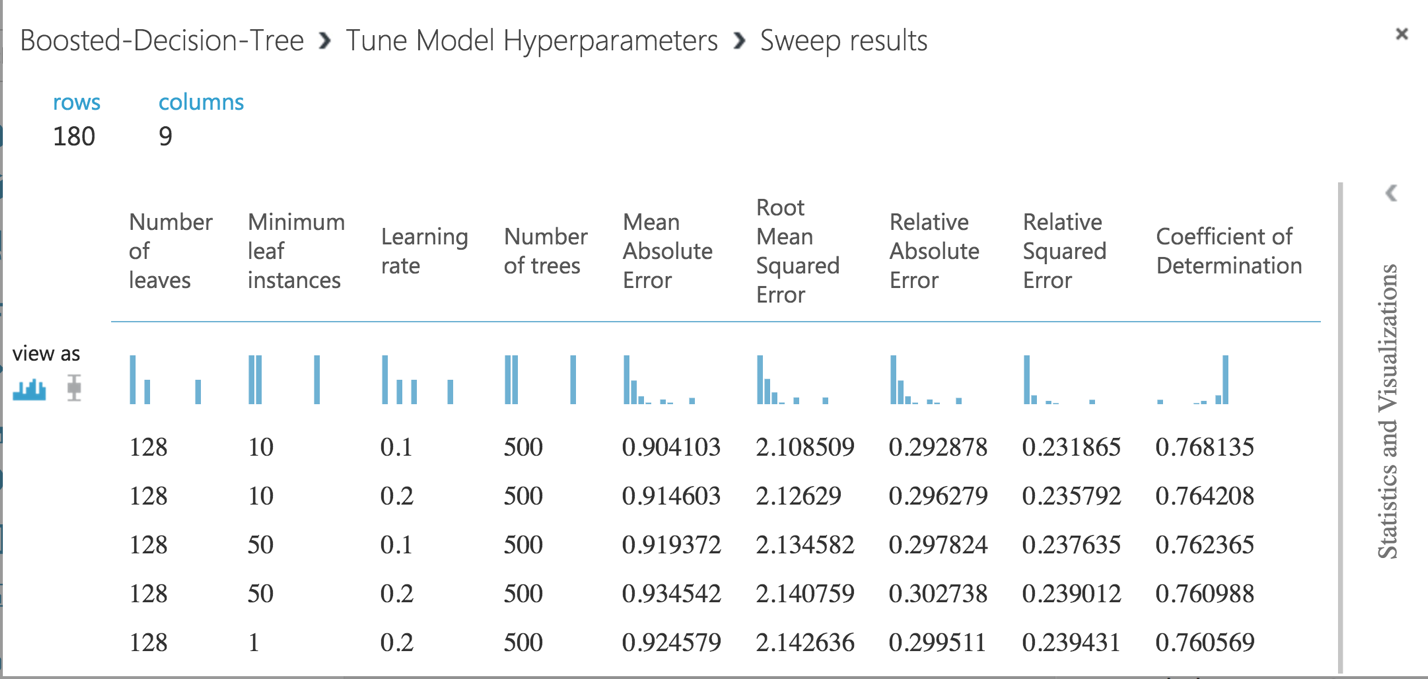
* 1. Decision Forest
     1. Model



* + 1. Result

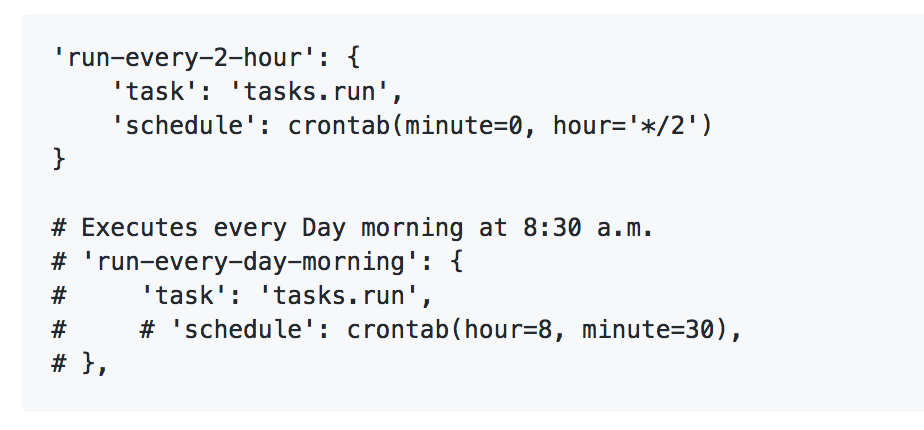


1. Best Model After Tuning Hyperparameters: Boosted Decision Tree

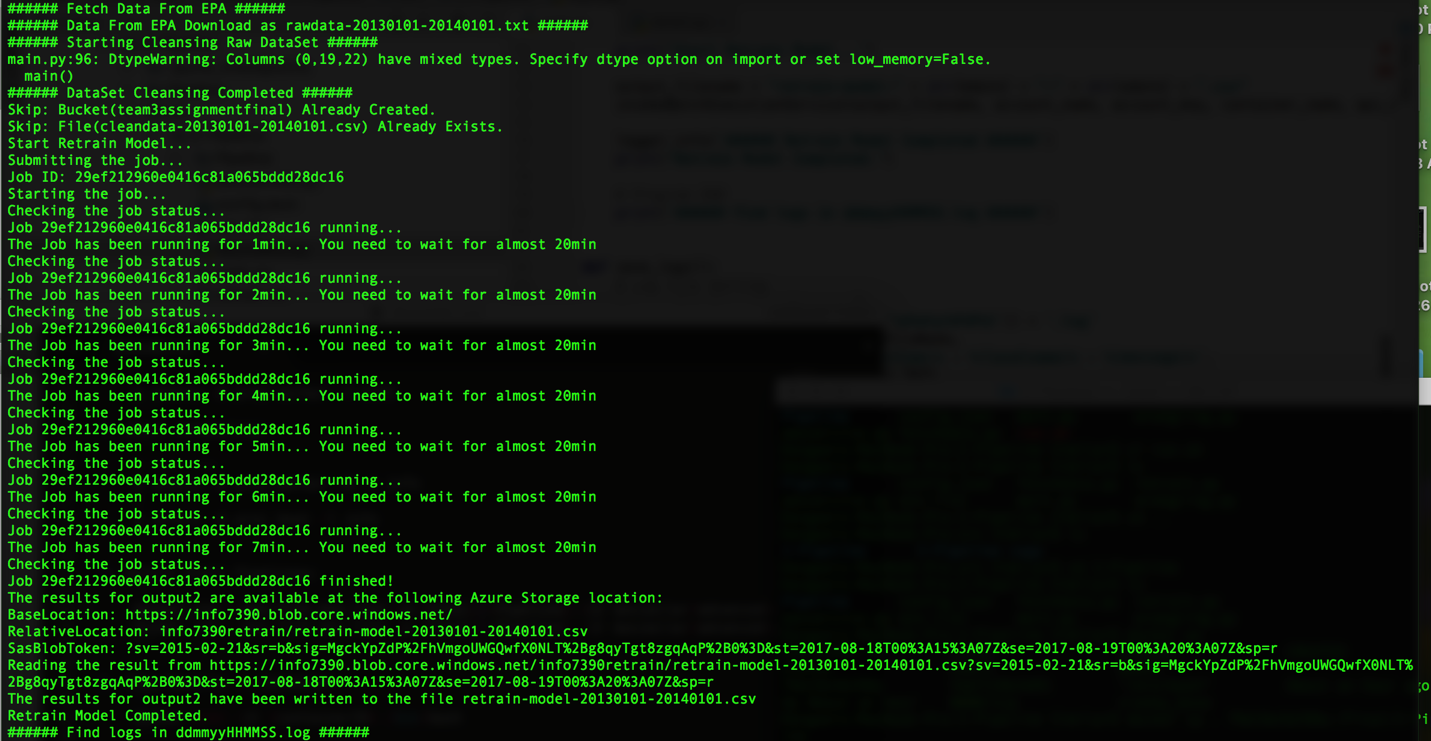


**Part 3: Pipelines for Regular Fetching & Retraining**

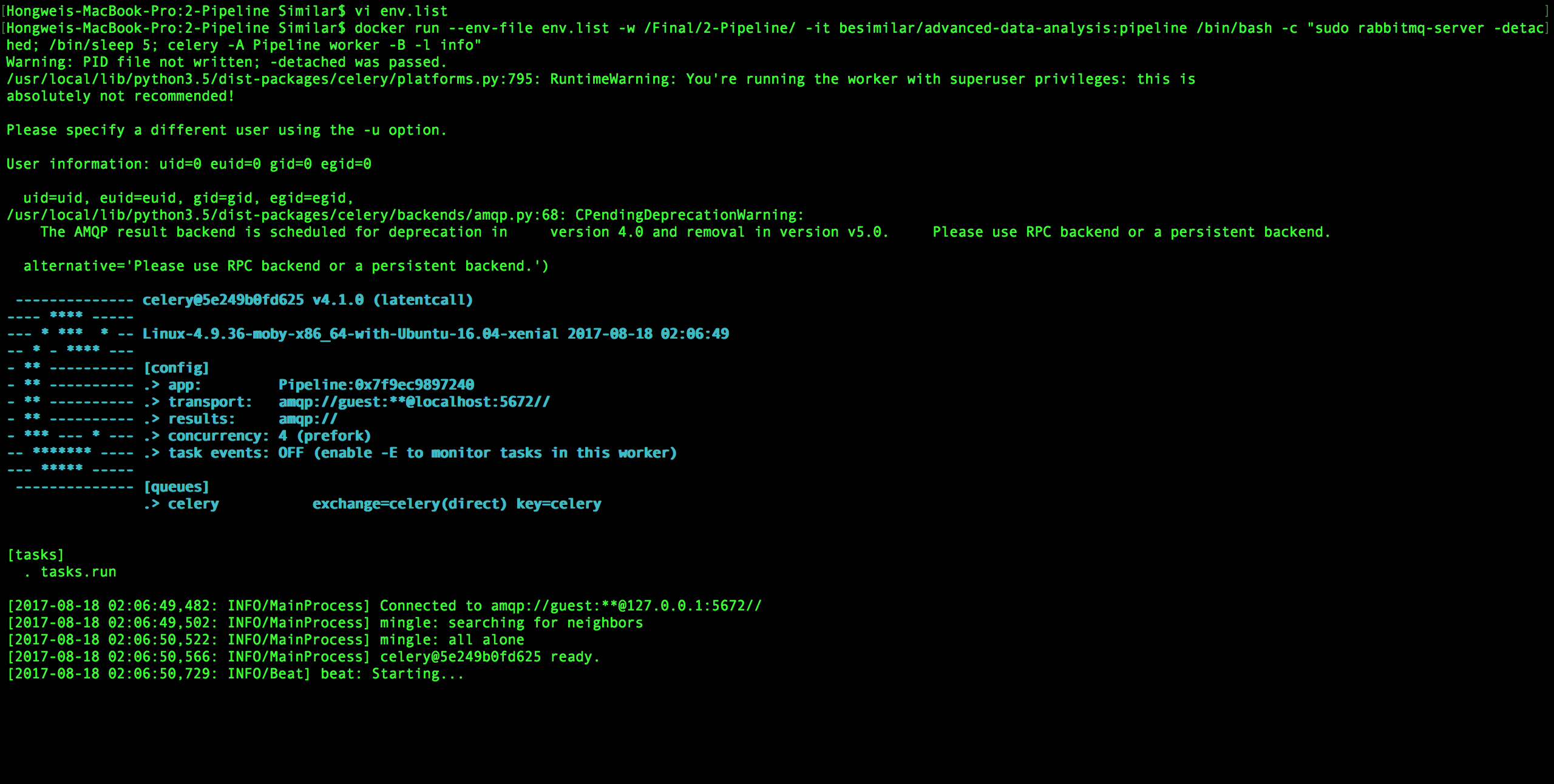
1. Source:
   1. Docker Hub: besimilar/advanced-data-analysis:pipeline
   2. GitHub: <https://github.com/Besimilar/Air-Quality-Analysis/tree/master/4-Pipeline>
2. What I did in this part:
   1. For demo, the job will be run every 2 hour. If you need to run every day, please uncomment code in "src/Pipeline/celery.py"



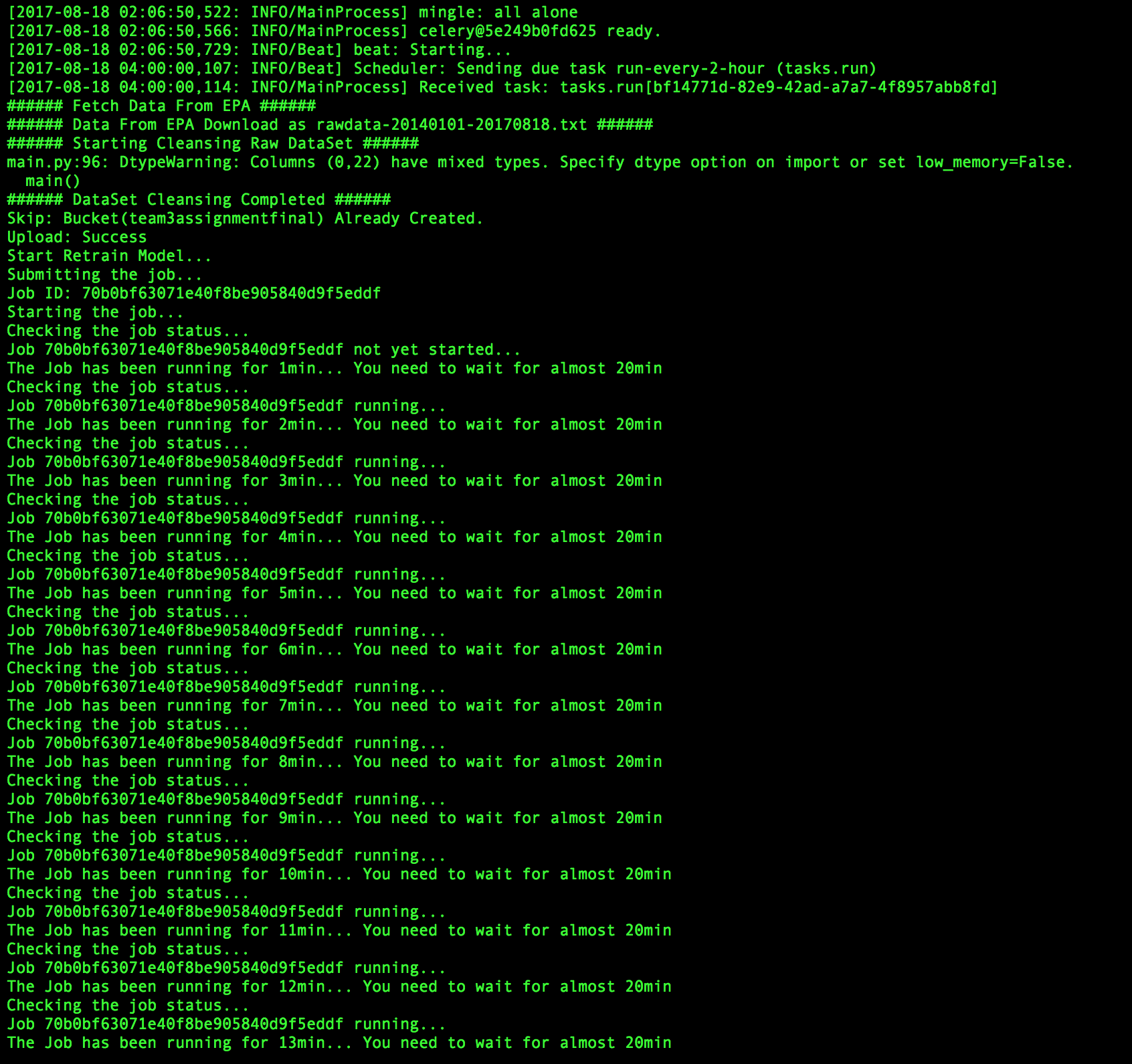
* 1. Fetch rawdata from EPA API ("src/fetchdata.py"):
  2. Clean rawdata and save both rawdata and cleandata to local ("src/wrangling.py")
  3. Upload cleandata to AWS S3 ("src/awsservice.py")
  4. Retrain Model in Azure Machine learning ("src/retrain.py"):
     1. Using Azure Batch Execution Service to retrain the model. For details, you can also see notebook in "notebook/retrain.ipynb"
     2. It will download retrain result to container as "retrain-model-bdate-edate.csv"
     3. You can also download retrain result from the link shown in terminal

****

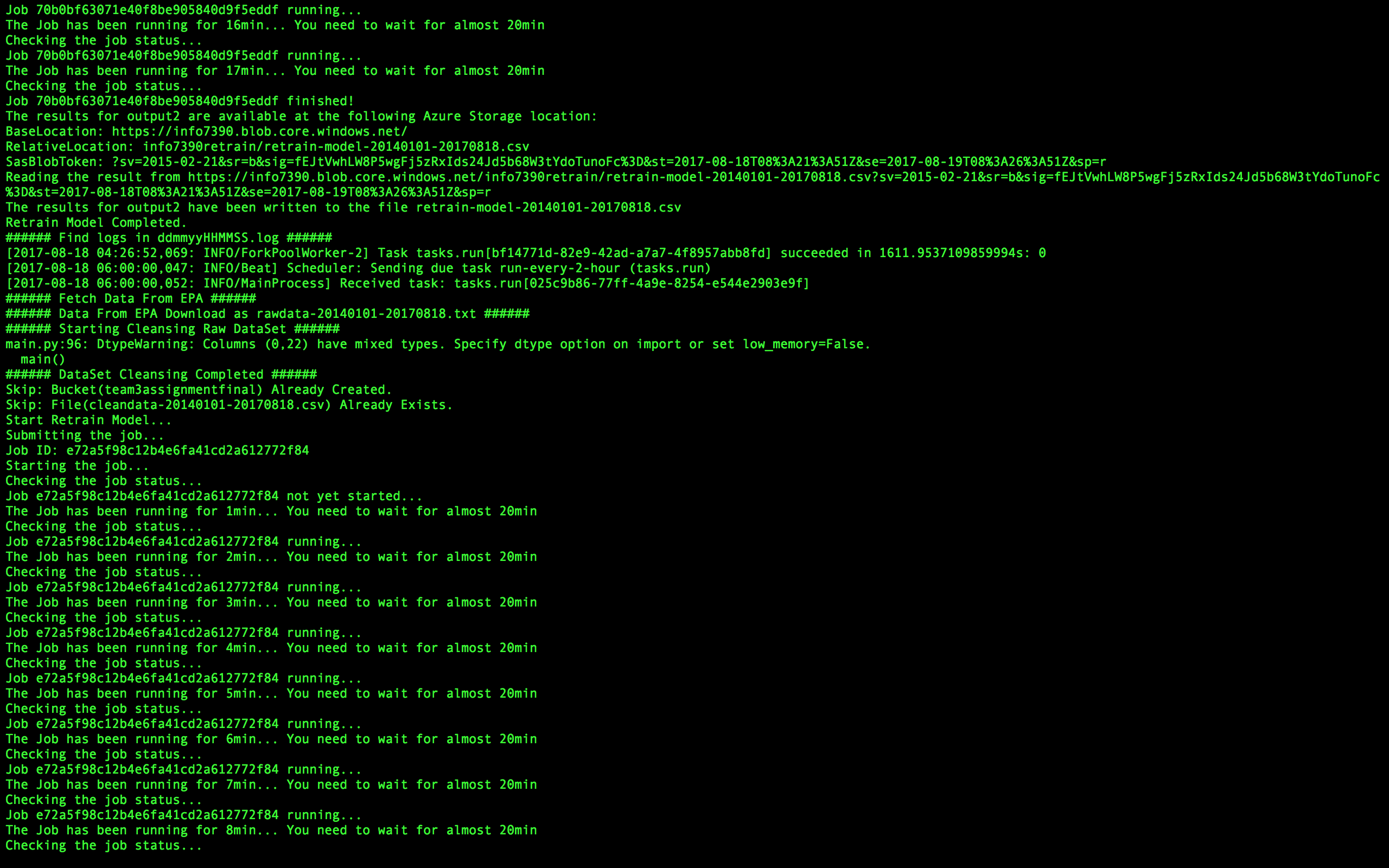
1. Instruction:
   1. **Most Import Step Before starting your container (same as Part 1):**
      1. refer to “src/env.list”, create your own env.list for container
      2. Set all parameters in env.list:
         1. AWS params:
            1. AWSACCESS: aws access key
            2. AWSSECRET: aws private key
            3. REGION: aws region
         2. EPA API params:
            1. EPAUSERNAME: EPA API username
            2. EPAPASSWORD: EPA API password
            3. BDATE: the first day of dataset (better within 3 years, try to set around 20140101, it’s a bug-free choice)
            4. EDATE: the last day of dataset: (default: current date)
         3. Azure params:
            1. ACCOUNTNAME: azure storage account name
            2. ACCOUNTKEY: azure storage primary key
            3. CONTAINERNAME: azure storage container name
         4. Azure ML params:
            1. APIKEY: predictive service key
            2. APIURL: predictive service url
         5. Other default params:
            1. change them in "config.json" After starting container
            2. You dont't need to change them to do a demo
   2. Start Pipeline Job:
      1. $ docker run --env-file env.list -w /Final/2-Pipeline/ -it besimilar/advanced-data-analysis:pipeline /bin/bash -c "sudo rabbitmq-server -detached; /bin/sleep 5; celery -A Pipeline worker -B -l info"
   3. View Results in Container
      1. $ docker start <containerID>
      2. $ docker exec -it <containerID> /bin/bash
   4. **Screenshot for all steps above: You could do the same as below to do a demo for this part (every 2 hour).**
      1. Start Job (notice the time: 2:06)



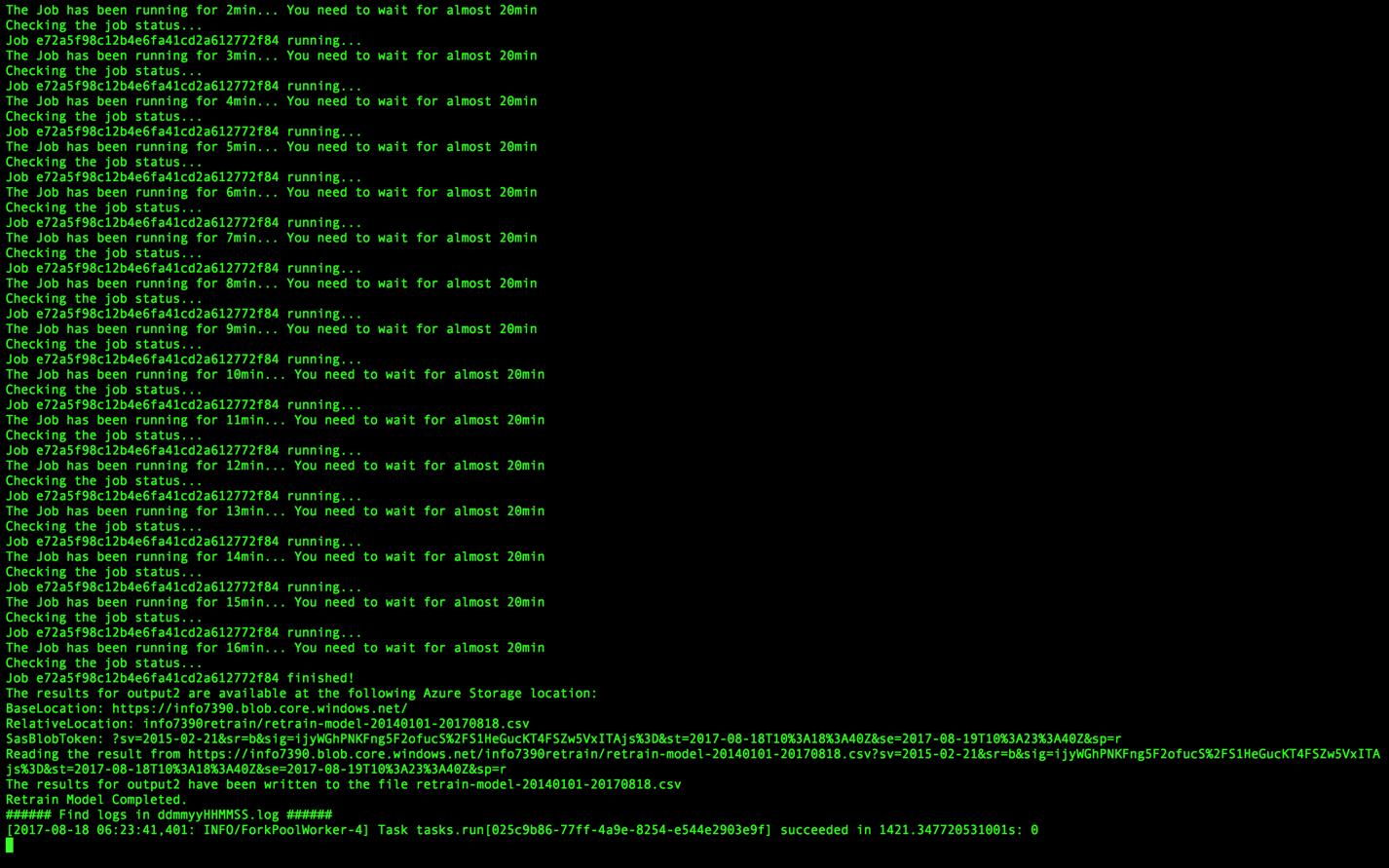
* + 1. 1st Job Running (4:00)



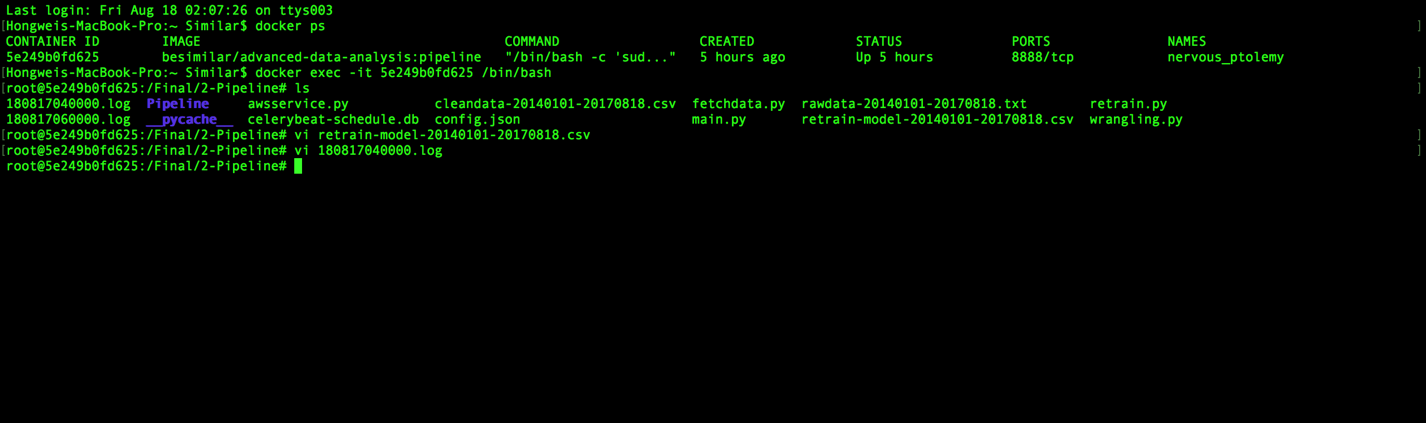
* + 1. 2nd Job Running (6:00)

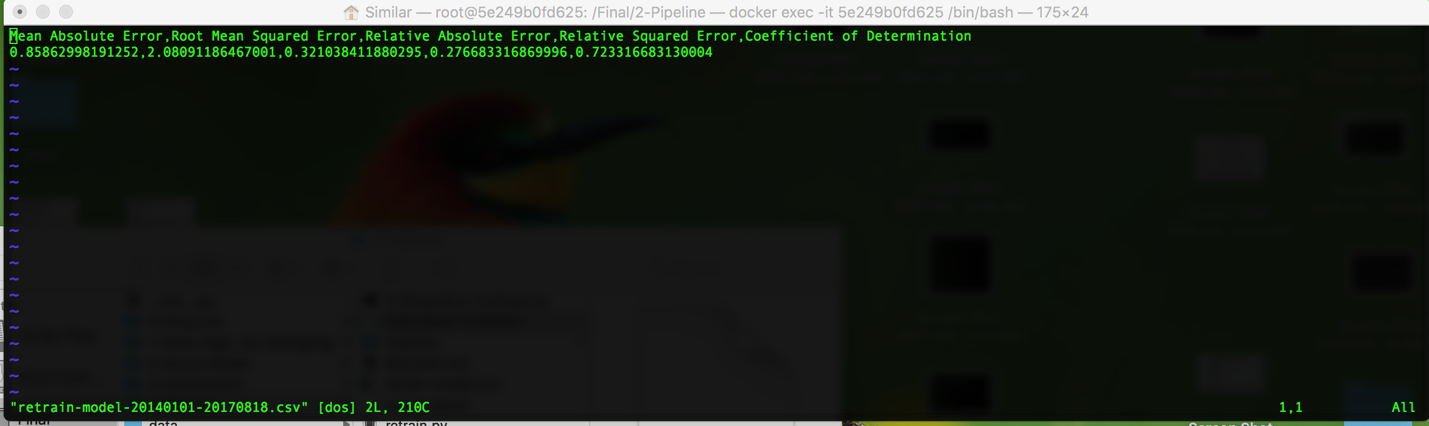


* + 1. Job End

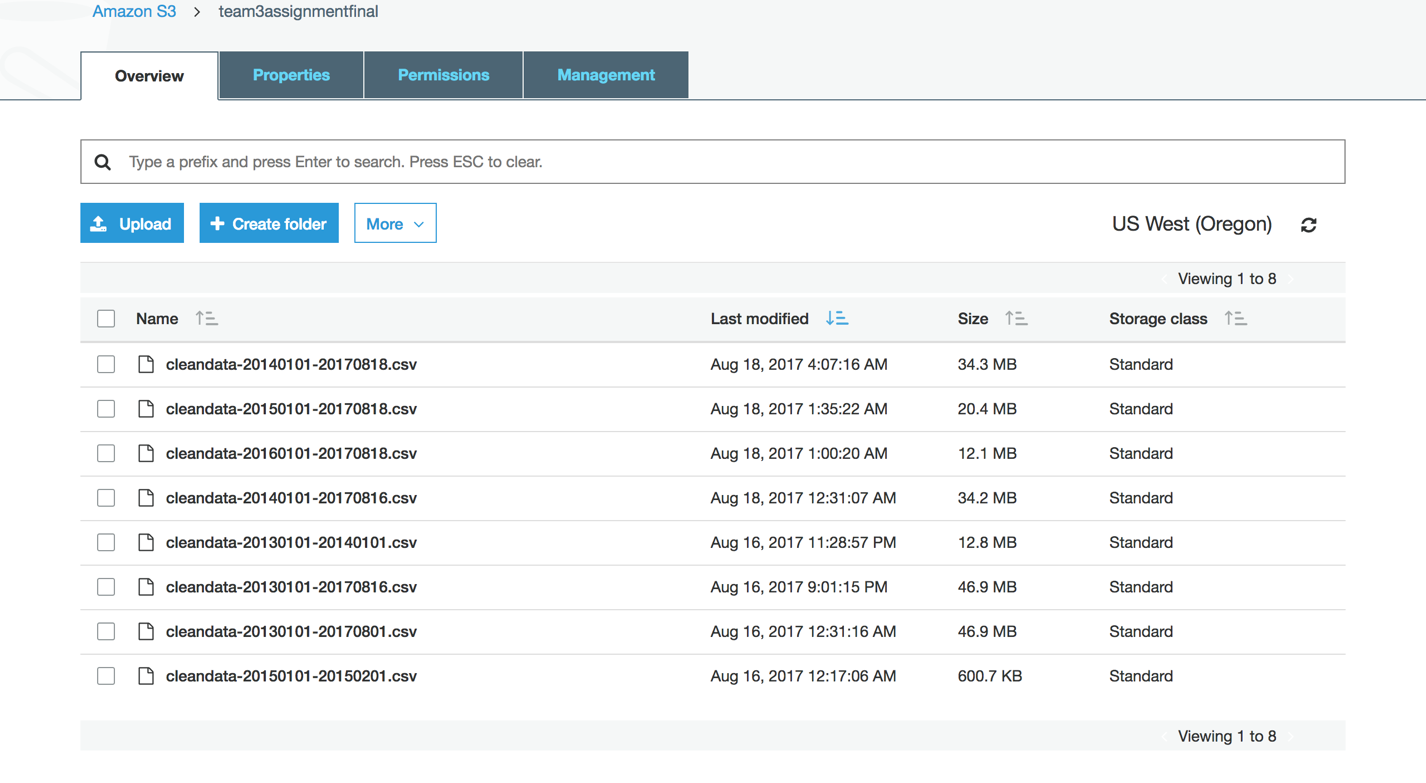


* + 1. View Retrain Results

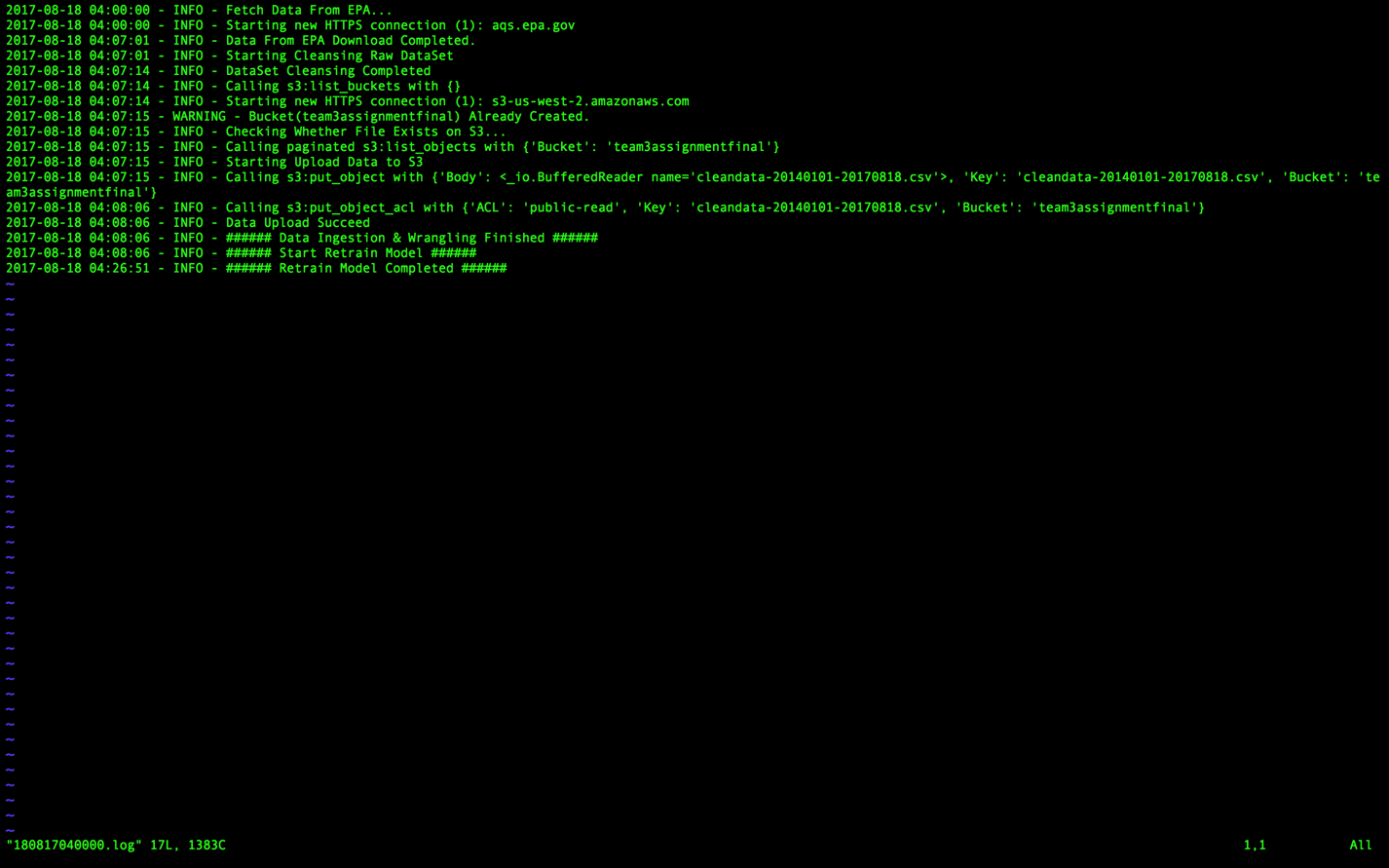


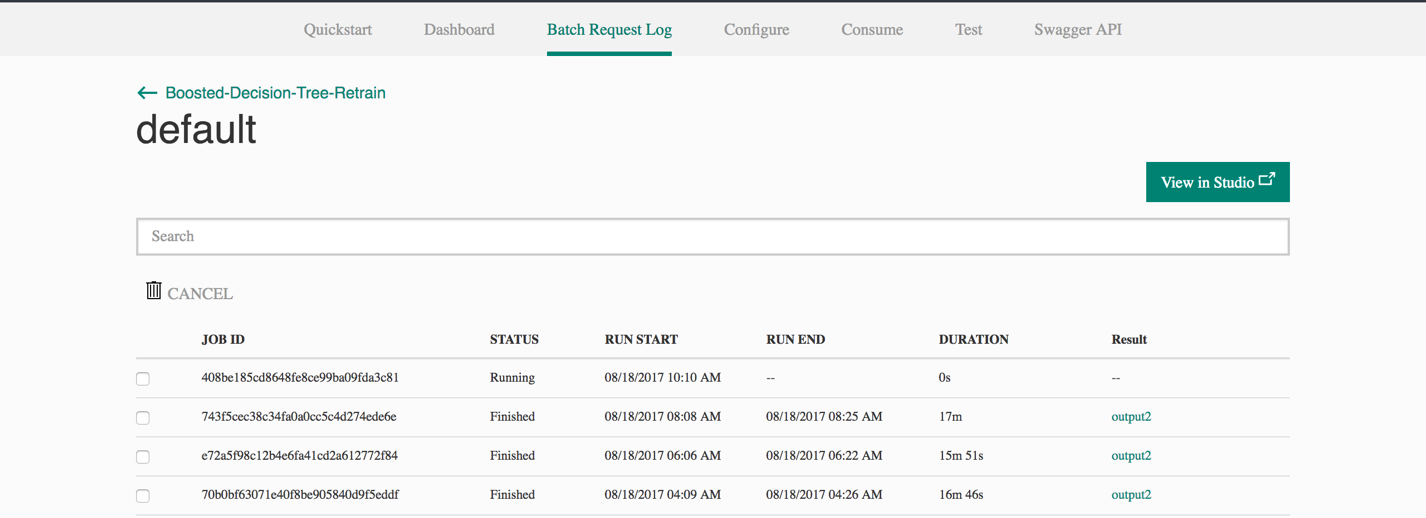


* + 1. S3



* + 1. Logs





**Part 4: Deployment**

1. Data Ingestion & Wrangling

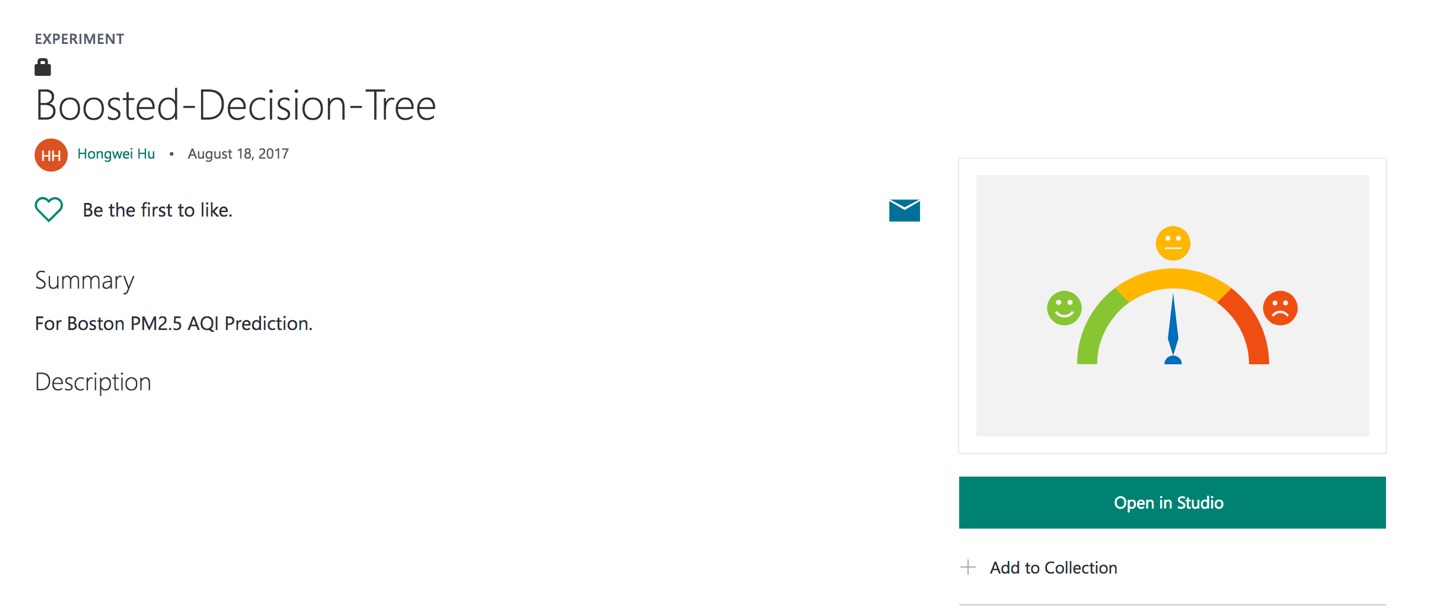
* Docker Hub: besimilar/advanced-data-analysis:aqi
* Download Link: <https://s3-us-west-2.amazonaws.com/team3assignmentfinal/boston-aqi.tar>

1. Pipelines

* Docker Hub: besimilar/advanced-data-analysis:pipeline
* Download Link: <https://s3-us-west-2.amazonaws.com/team3assignmentfinal/boston-aqi-pipeline.tar>

1. Model

* Azure Gallery: <https://gallery.cortanaintelligence.com/Experiment/Boosted-Decision-Tree>



1. FLASK API

* Pythonanywhere: <http://besimilar.pythonanywhere.com/prediction>

1. For Docker offline images

* You can download from links, and run your own demos
* Links:
  + Data Ingestion & Wrangling: <https://s3-us-west-2.amazonaws.com/team3assignmentfinal/boston-aqi.tar>
  + Pipelines: <https://s3-us-west-2.amazonaws.com/team3assignmentfinal/boston-aqi-pipeline.tar>
* Command lines: $ docker load --input <dockerimages>.tar