DATA 622 # hw2

Assigned on September 27, 2018
Due on October 17, 2018 11:59 PM EST
15 points possible, worth 15% of your final grade

Required Reading

Read Chapter 5 of the Deep Learning Book
Read Chapter 1 of the Agile Data Science 2.0 textbook

Data Pipeline using Python (13 points total)

Build a data pipeline in Python that downloads data using the urls given below, trains a random forest model on the training dataset using sklearn and scores the model on the test dataset.

Scoring Rubric

The homework will be scored based on code efficiency (hint: use functions, not stream of consciousness coding), code cleaniless, code reproducibility, and critical thinking (hint: commenting lets me know what you are thinking!)
Instructions:

Submit the following 5 items on github.

ReadMe.md (see "Critical Thinking")
requirements.txt
pull_data.py
train_model.py
score_model.py

More details:
requirements.txt (1 point)
This file documents all dependencies needed on top of the existing packages in the Docker Dataquest image from HW1. When called upon using pip install -r requirements.txt , this will install all python packages needed to run the .py files. (hint: use pip freeze to generate the .txt file)

pull_data.py (5 points)
When this is called using python pull_data.py in the command line, this will go to the 2 Kaggle urls provided below, authenticate using your own Kaggle sign on, pull the two datasets, and save as .csv files in the current local directory. The authentication login details (aka secrets) need to be in a hidden folder (hint: use .gitignore). There must be a data check step to ensure the data has been pulled correctly and clear commenting and documentation for each step inside the .py file.
Training dataset url: https://www.kaggle.com/c/titanic/download/train.csv
Scoring dataset url: https://www.kaggle.com/c/titanic/download/test.csv

train_model.py (5 points)
When this is called using python train_model.py in the command line, this will take in the training dataset csv, perform the necessary data cleaning and imputation, and fit a classification model to the dependent Y. There must be data check steps and clear commenting for each step inside the .py file. The output for running this file is the random forest model saved as a .pkl file in the local directory. Remember that the thought process and decision for why you chose the final model must be clearly documented in this section.
eda.ipynb (0 points)

[Optional] This supplements the commenting inside train_model.py. This is the place to provide scratch work and plots to convince me why you did certain data imputations and manipulations inside the train_model.py file.

score_model.py (2 points)
When this is called using python score_model.py in the command line, this will ingest the .pkl random forest file and apply the model to the locally saved scoring dataset csv. There must be data check steps and clear commenting for each step inside the .py file. The output for running this file is a csv file with the predicted score, as well as a png or text file output that contains the model

accuracy report (e.g. sklearn's classification report or any other way of model evaluation).

Critical Thinking (2 points total)
Modify this ReadMe file to answer the following questions directly in place.
I would put in checks to make sure the number of columns and the data types match the format so that the processes that wrangles the data later on before running ml part does not break. Maybe add a check of file to make sure it is not corrupt and is readable after the merge(Not sure how much of a problem this actually is, but this would not hurt). Also the way my scripts work right now, is I have statically specified where the NaN values are and I am handling them per case, since we do not know what data we will get, a more robust method of detecting and dealing with NaN values will be needed.

Kaggle changes links/ file locations/login process/ file content
We run out of space on HD / local permissions issue - can't save files

Someone updated python packages and there is unintended effect (functions retired or act differently)
Docker issues - lost internet within docker due to some ip binding to vm or local routing issues(I guess this falls under lost internet, but I am talking more if docker is the cause rather then ISP)

We would need some kind of back engine like Hadoop or Spark to handle the data. Also we will need ability to process data in chunks, data pull would need to be done in chunks, with further checks, processing in chunks and so on, also I would probably separate pipeline into more steps(scripts), for example I would take all the data wrangling parts into a separate scripts. If we have too much data to process we can't have multiple steps run at the same time as each one of those steps will be cpu and time consuming, so each step will have to be separated with more reporting/logging added in between each step. Trillion records sounds like a lot, so I would probably put some effort into monitoring of resources during the time my jobs run, With trillion records something will break sooner or later and if we have good overview of systems performance it will help with troubleshooting a lot.

I would add a cron job to check for differences daily, something like an rsync should get the job done, by checking if there is a change and pulling only the differences. Also if there is daily processing we will preferably need a more complex dependency based system where you check if last days jobs have completed successfully before moving on to the next day Same goes for steps within a day each one depends on completion of the other not just time based cron jobs. Alerts (email/texts) will be critical to have in place, they will be notifying successful completion of jobs or alert about a problem.

Please make sure you press enter when running last script, The output was pretty odd inside my docker container so I show part of output then, you hit enter then script will output the rest.

Also, if you use the wrong login/pass, you will have to delete login file. If file exists, it will just use whatever is in the file.