Kolby Boyd

ITAI 1378

Prof. McManus

L03

Exploring Jupyter Notebooks

L03 (a)

Becoming familiar with Jupyter Notebooks is no easy task, until you jump in and click all the buttons. The provided references are also a large help, guiding the process to discovering just what the program can do in your development and programming journey. I was pleasantly surprised to learn that a single Jupyter Notebook can be used to run programs, develop the software, and even work on pieces or chunks of a live, large project while it is being used by others, perform tests and then deploy patches.

The Jupyter 101 Notebook taught me how to utilize a Jupyter Notebook to organize thoughts in a coherent and presentable manner. Markdown formatting is very lightweight in its codex and as user friendly as Microsoft Word, or perhaps more akin to the simplicity of WordPad (rip). Alongside the markdown boxes, code boxes can be run instantly with a press of (Shift + Enter). Super Easy.

A kernel is the engine that executes the code in a notebook document. Each notebook is associated with a separate and specific kernel, be it Python, R, Julia or other programming languages. I'm not sure if I had the wrong files, but my kernel stopped working after the pandas code was introduced. This part of the lab was most frustrating not knowing what I was doing wrong and where to look to troubleshoot. I received a FileNotFoundError when attempting to load the first seven rows of the Pandas Dataframe. It took some time, but eventually I figured to rename the file folder holding the Data Set to just "data" as in the example, then reloading the notebook and that worked. Good Lab!

L03 (b)

This portion of the lab was simple: download the raw file from GitHub, open in Jupyter Notebooks and go through the code process. The code is already written. Each step took a few minutes to complete, but that was easily visible by the [*] next to each code box. I waited until each step was completed before moving onto the next. I changed some of the code, including the file path and added double backslashes so that the file path could be read correctly. Watching the model predict the breed of dog in the image was cool to see.

The pre-trained model was able to make predictions based on data it has seen before. The model's predictions are based on the learned patterns and features extracted from the training data during the training process. Data preprocessing is a crucial step in preparing the training data for the pre-trained model. If this step is not completed correctly, the model may not make correct predictions.

For example, if the model is not trained on rotated images, the model will make incorrect predictions based on what it sees, but if it is trained on these rotated images, it will be able to make predictions based on the rotation in any direction. This also includes other parameters, such as: lighting, rotation, inversion, colors (grayscale or RGB, etc.), and noise. This extensive training will create a more robust model.