upGrad

Session - 5: Convert Notebook to Scripts

In this session, the problem statement was to build an ML-based system that can classify whether a person has heart disease based on certain parameters.

You started out by training different ML models in a jupyter notebook. Once you finalized your model, you created an inference notebook, which was used to get the model's prediction on live data. Recall that you used the same data pre-processing steps as we did while training for the inference process.

After successfully creating the inference notebook, you modularized our code by abstracting out the code as a function. You stored all the utility functions in our utils.py file and stored all the constants in the constants.py file so we can easily configure our constants later on.

Once this was done, you added documentation to your code and ensured it followed pep8 guidelines. This ensures that your code is readable by others, thus facilitating smooth collaboration.

After this, you performed unit testing to ensure that the code works as intended and that there are no bugs present in the code.

Once you were sure there were no bugs in your code, you created the main.py file and included exception handling and logging.
With this, you were able to create a production-ready inference process code.

Now let's understand why python scripts are preferred instead of Jupyter notebooks in production ML Systems.

- Difficult to implement proper code versioning and track changes
- Difficult to establish dependency management for notebooks
- Difficult for peer reviewing and working on the same notebook by multiple developers
- Challenges in automating continuous integration and continuous deployment of code (CI/CD)