Artificial Intelligence (Machine Learning & Deep Learning) [Course]

Week 18 - AI Deployment

[See examples / code in GitHub code repository]

It is not about Theory, it is 20% Theory and 80% Practical – Technical/Development/Programming [Mostly Python based]

AI Deployment

Deploying machine learning models enables your applications to make real-time predictions and decisions.

Reference:

https://medium.com/@emyasenc/deploying-machine-learning-models-with-flask-fastapi-or-streamlit-an-in-depth-guide-30c2e1f2ee44

AI Deployment Via Flask

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Poocco. Flask is based on the Werkzeg WSGI toolkit and the Jinja2 template engine.Both are Pocco projects.

Reference:

https://www.geeksforgeeks.org/python/flask-tutorial/
https://pythonbasics.org/what-is-flask-python/
https://flask.palletsprojects.com/en/stable/

Practical Development Case Study

Reference:
https://medium.com/@emyasenc/deploying-machine-learning-models-with-flask-fastapi-or-streamlit-an-in-depth-guide-30c2e1f2ee44

Model Saving:

https://github.com/ShahzadSarwar10/FullStackAI-B-4-SAT-SUN-10AM-TO-12PM/blob/main/Week18/Case17-2A-LinearRegressionViaSciKitLearn-SaveModel.py

Model Calling via Flask

https://github.com/ShahzadSarwar10/FullStackAI-B-4-SAT-SUN-10AM-TO-12PM/blob/main/Week18/Case17-2C-LinearRegressionViaSciKitLearn-ModelDeploymentWithFlask.py

AI Deployment Via FastAPI

FastAPI is a modern, high-performance web framework for building APIs with Python based on standard type hints. It has the following key features:

Fast to run: It offers very high performance, on par with NodeJS and Go, thanks to Starlette and pydantic.
Fast to code: It allows for significant increases in development speed.
Reduced number of bugs: It reduces the possibility for human-induced errors.
Intuitive: It offers great editor support, with completion everywhere and less time debugging.
Straightforward: It's designed to be uncomplicated to use and learn, so you can spend less time reading documentation.
Short: It minimizes code duplication.
Robust: It provides production-ready code with automatic interactive documentation.
Standards-based: It's based on the open standards for APIs, OpenAPI and JSON Schema.

Practical Development Case Study

Reference:
https://medium.com/@emyasenc/deploying-machine-learning-models-with-flask-fastapi-or-streamlit-an-in-depth-guide-30c2e1f2ee44

Model Saving:

https://github.com/ShahzadSarwar10/FullStackAI-B-4-SAT-SUN-10AM-TO-12PM/blob/main/Week18/Case17-2A-LinearRegressionViaSciKitLearn-SaveModel.py

Model Calling via FASTAPI

https://github.com/ShahzadSarwar10/FullStackAI-B-4-SAT-SUN-10AM-TO-12PM/blob/main/Week18/Case17-2D-LinearRegressionViaSciKitLearn-ModelDeploymentWithStreamlit.py

AI Deployment Via Streamlit

Streamlit is an open-source Python library that simplifies the creation and sharing of custom web applications, particularly for data science, machine learning, and AI applications. It enables developers to build interactive user interfaces and dashboards using only Python code, eliminating the need for traditional front-end technologies like HTML, CSS, or JavaScript.

How it works:
Streamlit works by evaluating a Python script from top to bottom and rendering the corresponding UI elements in a web browser. User interactions with widgets trigger a rerun of the script, allowing for dynamic updates and maintaining the application's state. This reactive programming model simplifies the development of interactive data applications without complex callbacks or event handling.

Reference:
https://streamlit.io/
https://www.geeksforgeeks.org/python/a-beginners-guide-to-streamlit/
https://www.datacamp.com/tutorial/streamlit

Practical Development Case Study

Reference:
https://medium.com/@emyasenc/deploying-machine-learning-models-with-flask-fastapi-or-streamlit-an-in-depth-guide-30c2e1f2ee44

Model Saving:

https://github.com/ShahzadSarwar10/FullStackAI-B-4-SAT-SUN-10AM-TO-12PM/blob/main/Week18/Case17-2A-LinearRegressionViaSciKitLearn-SaveModel.py

Model Calling via Streamlit

https://github.com/ShahzadSarwar10/Fullstack-WITH-AI-B-3-SAT-SUN-6Months-Explorer/blob/main/Week16/Case16-2D-LinearRegressionViaSciKitLearn-ModelDeploymentWithStreamlit.py

AI Deployment - MLOps – Model Pipeline - CI/CD

Understanding TFX Pipelines

MLOps is the practice of applying DevOps practices to help automate, manage, and audit machine learning (ML) workflows. ML workflows include steps to:

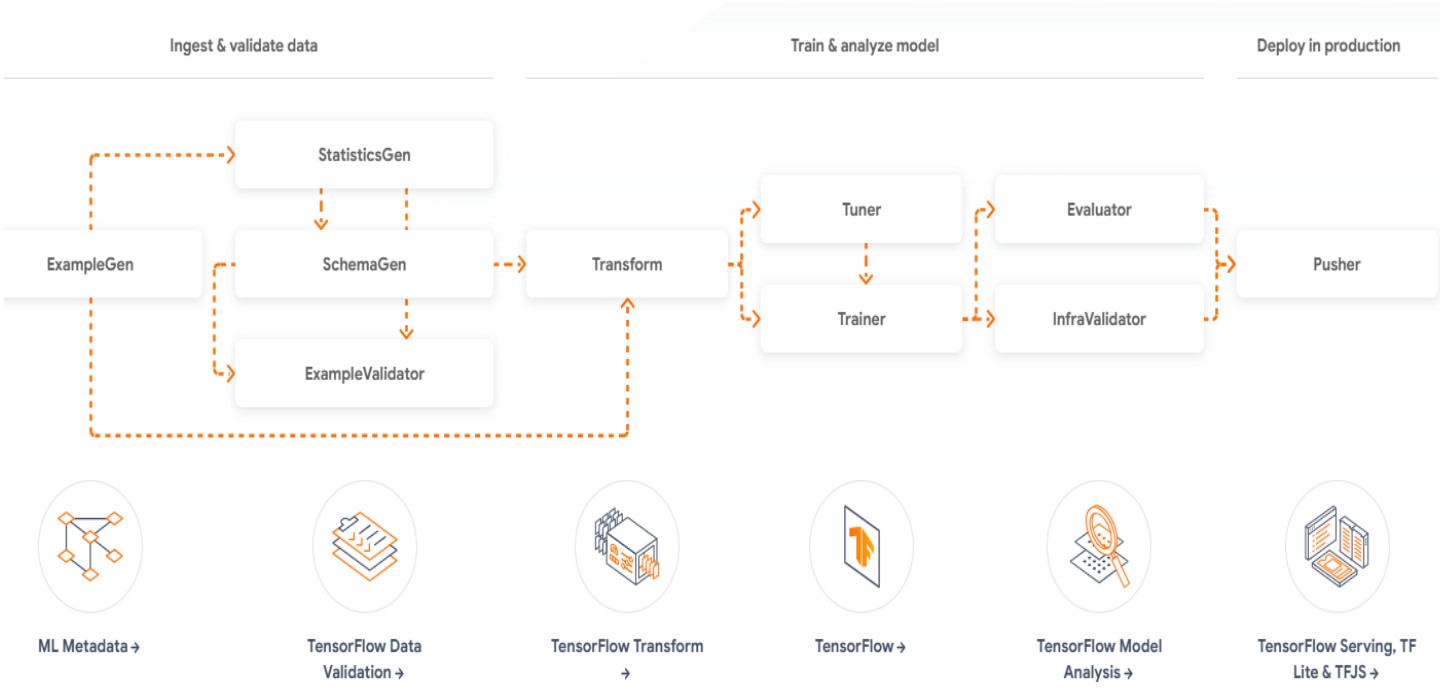
- Prepare, analyze, and transform data.
- Train and evaluate a model.
- Deploy trained models to production.
- Track ML artifacts and understand their dependencies.

Managing these steps in an ad-hoc manner can be difficult and time-consuming.

TFX makes it easier to implement MLOps by providing a toolkit that helps you orchestrate your ML process on various orchestrators, such as: Apache Airflow, Apache Beam, and Kubeflow Pipelines.

By implementing your workflow as a TFX pipeline, you can:

- Automate your ML process, which lets you regularly retrain, evaluate, and deploy your mdel.
- Utilize distributed compute resources for processing large datasets and workloads.
- Increase the velocity of experimentation by running a pipeline with different sets of hyperparameters.



Reference:

https://www.youtube.com/watch?v=I3MjuFGmJrg

Practical Case Study

Reference:

https://www.tensorflow.org/tfx/tutorials/tfx/components_keras

https://colab.research.google.com/github/tensorflow/tfx/blob/master/docs/tutorials/tfx/penguin_simple.ipynb#scrollTo=Gnc67uQNTDfW

Thank you - for listening and participating

- ❑**Questions / Queries**
- ❑**Suggestions/Recommendation**
- ❑**Ideas…..?**

Shahzad Sarwar
Cognitive Convergence
https://cognitiveconvergence.com
shahzad@cognitiveconvergence.com
voice: +1 4242530744 (USA) +92-3004762901 (Pak)