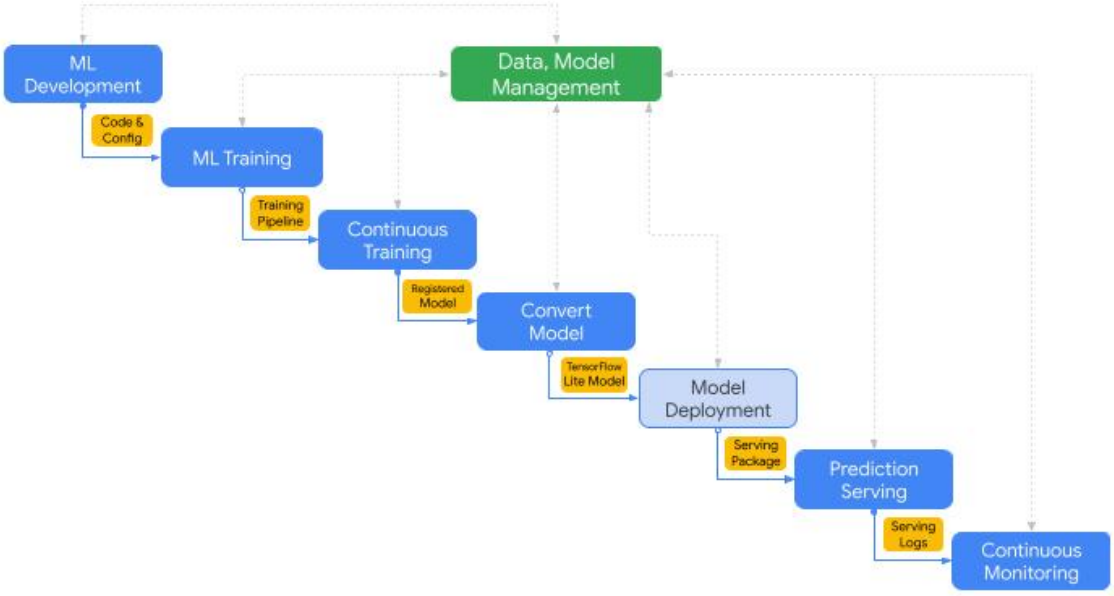
Overview of Model Deployment



Objectives

Once we have developed, trained, and validated our models, we want them to be implemented into our processes. This stage of implementation is what is often called **model deployment**. The deployment stage involves packaging the model, performing a suite of tests, and then deploying it to our target environment. For more traditional deployments, this could be a web server or Kubernetes cluster. Deployment to a production environment is usually more complex than just saving over our previous model, or adding a new one. Typically, it also involves continuous monitoring of the predictions, and deployment is often performed using one of a number of techniques which we will briefly discuss. For TinyML, this more so refers to deployment on a specific device and that is where several unique challenges arise. To this end, in this section we will cover the following topics to understand how to robustly deploy models:

- Containers and the role that containers play in easing ML deployments into the wild
- Challenges with the lack of containerization methods for embedded systems
- Tiny Machine Learning as a Service (tinyMLaaS) as a potential solution for the issue

Motivation

We have previously discussed deployment in the context of “ML Training” or “Training Operationalization” as it was also relevant to the issues of testing. More often than not it is important to test in production as it is impossible to know if you have covered all your bases despite having integration, smoke, as well as acceptance testing and so forth. So here we will do a quick recap on the deployment methods: basic, multi-service, rolling, canary, blue-green and shadow. As you can now see, there are many ways for us to introduce a new model into our production environment. Some of these are easy to implement, but relatively risky, while others are complex but relatively safe. The most suitable deployment strategy depends strongly on the application, but also things like cost, existing infrastructure, and technical expertise available.

In this module, we will be focusing on another part of model deployment known ‘**as a service**.’ This extends well beyond the Continuous Integration and Continuous Deployment requirements. Often when you have to deploy models into production, you have to have the right infrastructure in place. One can choose to build all of the components themselves or try and lease them from others who are providing such services for a fee—leasing alleviates the customer to focus on *their* problem at hand, rather than build all the components from scratch with little to no experience just to deploy their solution. For instance, you can set up a web server with a push of a button today, thanks in part to all the “LAMP” (Linux, Apache, MySQL and PHP) infrastructure components that have all matured well and been integrated well with one another. This allows you to focus on the website you are designing, rather than figuring out how to bolt together all these different major software components together just to get your website hosted.

Similarly, when we want to deploy ML models into production at scale, we need the right underlying software infrastructure in place. Arguably, many of these components are still missing and if not they are limited in their capabilities. To understand the challenges and opportunities, we will be introducing **Tiny Machine Learning as a Service (TinyMLaaS).**