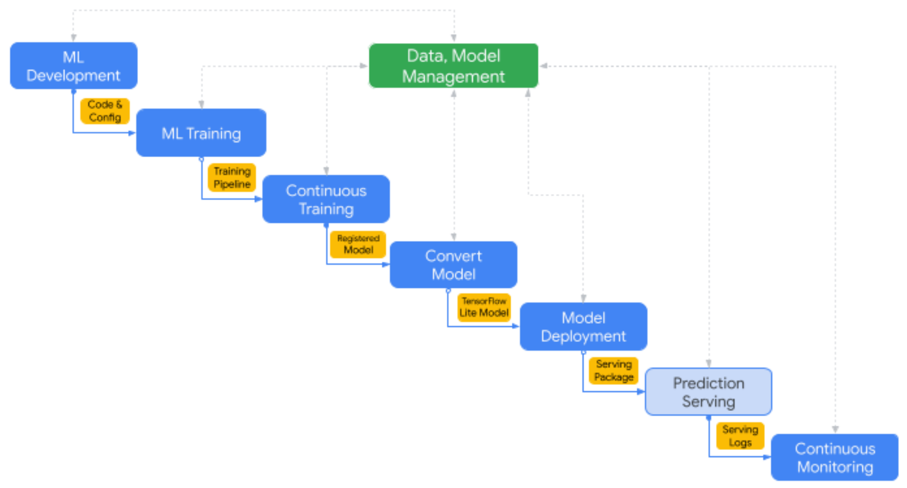
Overview of Prediction Serving



Objective

While much of the attention in machine learning research has been focused on the process of training models (i.e., learning), there is a distinct set of issues associated with the process of serving and updating those models. In this section, we will look at the broader machine learning life-cycle and address the difficulties associated with giving predictions to customers. To this end, we will learn the following concepts and ideas in this section:

- Different serving scenarios: Batch, Offline, Streaming and Embedded
- Serving architectures: Blackbox vs. Whitebox methods
- Benchmarking embedded inference serving scenarios

Motivation

Once we have our production environment ready, this is where the fun really begins. Let us imagine that our production environment is a restaurant. Our environment might have many different services that it can perform, in the same way that a chef in a restaurant can (hopefully) cook more than one dish. Depending on what our customer orders, we may have to serve up different results. The idea of a “customer” and an “order” is slightly more subtle in the machine learning sense, since often the inferences performed by machine learning models are done on-the-fly or without knowledge from the customer. An example of this could be fraud prevention in a bank. Oftentimes, when you make a transaction, a machine learning model will decide whether the behavior from your account looks “normal” or “suspicious”. If you buy a coffee from a local store every day, the model will likely know this is typical behavior of you, so has no reason to be concerned. However, if your card is suddenly used to purchase a piano in Costa Rica, the model might view this more suspiciously (unless you mentioned this trip or purchase to the bank beforehand). Many of the models that are in production today are somewhat hidden from the customer, but the models are still serving up predictions.

Prediction Serving Pipeline

Prediction serving starts with the user, whereby a request is made, knowingly or unknowingly, to the service provider. This is typically done through a REST API, which will send a JSON query to the server. The server will then, based on the information in the query, output a prediction and respond to the user via the REST API. This procedure is common for singleton requests to provide online predictions in real time.

In some situations, the model may require additional information that is not provided by the user. For a model that is meant to predict the likelihood that a customer will purchase a particular product, details about the customer, as well as the product, will likely be needed. The model will then have to take the product identifier and use this to interact with a product database to obtain relevant information about the product, which it will use in its prediction. Depending on how many external features might need to be queried to make a prediction, this can make the prediction serving pipeline quite complex.

Interpretability is also important in prediction serving (especially in Europe due to the "Right to Explanation") to provide rationale for why a particular prediction was provided. For example, why the price of a quote was the price provided, or why an individual was rejected from receiving a loan or a new credit card. These rationales are often saved alongside predictions in a serving log, so that the service provider can keep track of all of the predictions given to customers that it has served. In the remainder of this section, we will delve deeper into the nuances of prediction serving and how it is relevant to TinyML.