CHAPTER 47

# Deploying an End-to-End Machine Learning Solution on Kubeflow Pipelines

A Kubeflow pipeline component is an implementation of a pipeline task. A component is a step in the workflow. Each task takes one or more artifacts as input and may produce one or more artifacts as output.

Each component usually includes two parts:

- Client code: The code that talks to endpoints to submit jobs, for example, code to connect with the Google Cloud Machine Learning Engine.
- Runtime code: The code that does the actual job and usually runs in the cluster, for example, the code that prepares the model for training on Cloud MLE.

A component consists of an interface (inputs/outputs), the implementation (a Docker container image and command-line arguments), and metadata (name, description).




E. Bisong, *Building Machine Learning and Deep Learning Models on Google Cloud Platform*,
https://doi.org/10.1007/978-1-4842-4470-8_47

# Overview of a Simple End-to-End Solution Pipeline

In this simple example, we will implement a deep neural regressor network to predict the closing prices of Bitcoin crypto-currency. The machine learning code itself is pretty basic as it is not the focus of this article. The goal here is to orchestrate a machine learning engineering solution using microservice architectures on Kubernetes with Kubeflow Pipelines. The code for this chapter is in the book code repository. Clone the repository from the GCP Cloud Shell.

The pipeline consists of the following components:

1. Move raw data hosted on GitHub to a storage bucket.
2. Transform the dataset using Google Dataflow.
3. Carry out hyper-parameter training on Cloud Machine Learning Engine.
4. Train the model with the optimized hyper-parameters.
5. Deploy the model for serving on Cloud MLE.

# Create a Container Image for Each Component

First, we'll package the client and runtime code into a Docker image. This image also contains the secure service account key to authenticate against GCP. For example, the component to transform the dataset using Dataflow has the following files built into its image:

- `__ Dockerfile`: Dockerfile to build the Docker image.
- `__ build.sh`: Script to initiate the container build and upload to Google Container Registry.
- `__ dataflow_transform.py`: Code to run the beam pipeline on Cloud Dataflow.
- `__ service_account.json`: Secure key to authenticate container on GCP.
- `__ local_test.sh`: Script to run the image pipeline component locally.