Solution Brief

Cross-Industry Architecture
Intel® Optimized Cloud Modules

intel software

Intel® Optimized Cloud Modules for Microsoft Azure*: XGBoost* Kubeflow* Pipeline

Author: Kelli Belcher

August 16, 2023

Build an accelerated Kubeflow Pipeline with the Intel® Optimization for XGBoost* on an Azure confidential computing cluster

The **Intel® Optimized Cloud Modules for Microsoft Azure*: XGBoost* Kubeflow* Pipeline** is designed to facilitate building and deploying accelerated AI applications on **Kubeflow***. Kubeflow is the simple, portable, and scalable machine learning toolkit for Kubernetes* services. This reference solution provides an optimized training and inference Kubeflow pipeline using XGBoost to predict the probability of a loan default. The module enables the use of **Intel® Optimization for XGBoost***, **Intel® oneAPI Data Analytics Library (oneDAL)**, and **Intel® Extension for Scikit-Learn*** in a full end-to-end reference architecture. It also leverages secure and confidential computing using the **Intel® Software Guard Extensions (Intel® SGX)** virtual machines on an **Azure Kubernetes Services (AKS)** cluster.

Use it as a reference solution for:

- Data preprocessing
- Machine learning for binary classification tasks
- Model inference and performance analytics
- Kubernetes applications and Kubeflow Pipelines
- Confidential computing applications

Who needs it?

- Developers aiming to maximize performance and productivity with industry-leading, CPU-accelerated Python* libraries for machine learning
- Developers with access to new or existing Azure Kubernetes applications and Kubeflow pipeline code who want to take advantage of secure and confidential computing virtual machines leveraging Intel® Software Guard Extensions

What it does

This module demonstrates how to build a scalable, accelerated machine learning pipeline on a secure and confidential computing cluster. The module serves as a guide for the following:

- Configuring Microsoft Azure* Cloud Services, including an **Azure resource group**, an **Azure container registry (ACR)**, and an **Azure Kubernetes Services (AKS)** cluster with a confidential computing node pool.
- Installing the Kubeflow software layer on the Azure Kubernetes cluster using a secure transport layer security (TLS) protocol.
- Implementing a full, end-to-end machine learning pipeline to predict credit risk using an XGBoost model, from data preprocessing to model inference with Intel oneAPI software optimizations.
- Leveraging the security of confidential computing nodes on an Azure Kubernetes Services cluster using the Intel® SGX, powered by 3rd Generation **Intel® Xeon® Scalable processors** and use **Intel® Turbo Boost Max Technology 3.0** to reach 3.5 GHz for AI acceleration.

Cloud Solution Architecture

This cloud solution utilizes an AKS cluster, with the central components being Intel® SGX virtual machines (VMs). Intel SGX VMs allow you to run sensitive workloads and containers within a hardware-based trusted execution environment (TEE). TEEs enable user-level code from containers to allocate enclaves, which are private regions of memory to execute the code directly on the CPU. Enclaves help

protect data confidentiality, data integrity and code integrity from other processes running on the same node. An Azure Container Registry is attached to the AKS cluster so that the Kubeflow pipeline can build the containerized Python components. These Azure resources are managed in an Azure Resource Group. The Kubeflow software layer is then installed on the AKS cluster. When the pipeline is run, each pipeline pod will be assigned to an Intel SGX node. Figure 1 shows the cloud solution architecture.

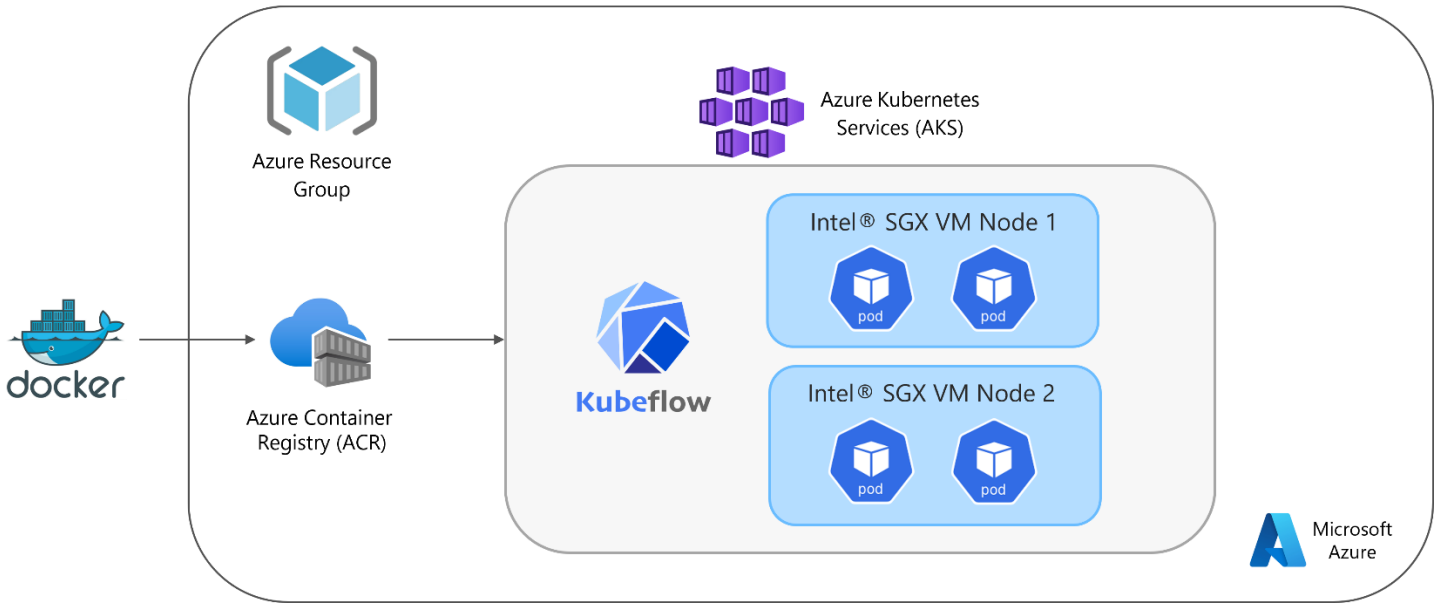

Figure 1: XGBoost Kubeflow Pipeline Architecture Diagram. Image from author.

Solution Component Overview

This solution is derived from the **Loan Default Risk Prediction AI Reference Kit**, built by Intel in collaboration with Accenture*. The code has been enhanced through refactoring to achieve better modularity and suitability for Kubeflow Pipelines. The credit risk data set used in the pipeline is obtained from **Kaggle*** and synthetically augmented for testing and benchmarking purposes. Figure 2 shows the full XGBoost daal4py Kubeflow pipeline.

The XGBoost Daal4py Kubeflow pipeline consists of the following seven components:

1. **Load Data**
   This component loads the dataset (credit_risk_dataset.csv) from the URL specified in the pipeline run parameters and performs synthetic data augmentation. **Download** the Kaggle Credit Risk Dataset.
2. **Create training and test sets**
   This component splits the data into training and test sets of an approximately 75:25 split for model evaluation.
3. **Preprocess features**
   This component transforms the categorical features of the training and test sets by using one-hot encoding, imputes missing values and power-transforms numerical features.
4. **Train XGBoost model**
   This component trains an XGBoost model using the accelerations provided by the Intel optimizations for XGBoost.
5. **Convert XGBoost model to daal4py**
   This component converts the XGBoost model to an inference-optimized daal4py classifier.
6. **daal4py Inference**
   This component computes predictions using the inference-optimized daal4py classifier and evaluates model performance. It returns an output summary of the precision, recall, and F1 score for each class, as well as the area under the curve (AUC) and accuracy score of the model.
7. **Plot ROC Curve**
   This component performs model validation on the test data and generates a graph of the receiver operating characteristic (ROC) curve.

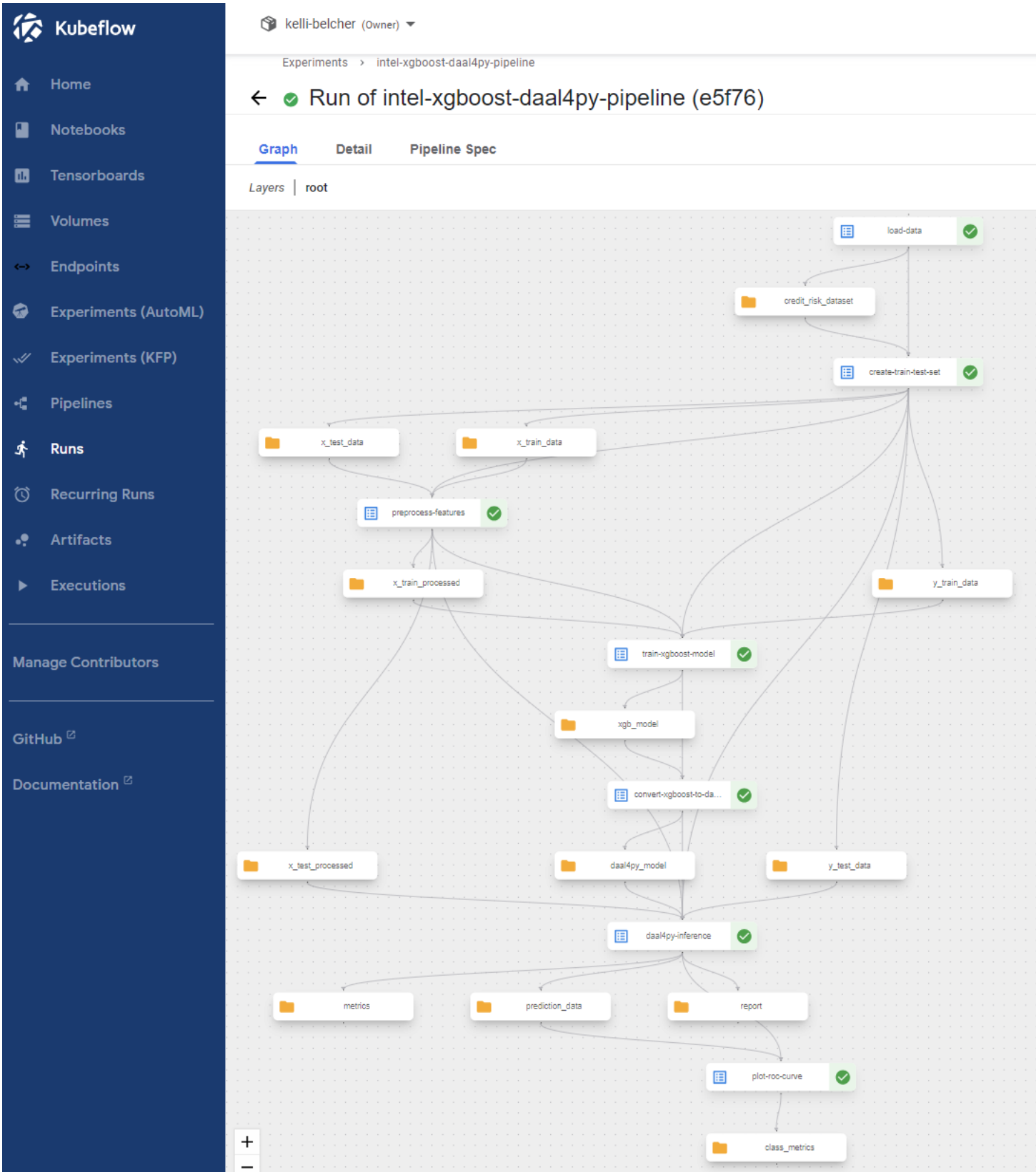

Figure 2: XGBoost Kubeflow Pipeline Graph. Image from author.

Code Highlights

Enable the Intel Optimizations for XGBoost

The **XGBoost optimizations** for training and inference on CPUs are upstreamed into the open source XGBoost framework. Ensure you are using the latest version of XGBoost to have access to the most Intel optimizations. The following code is implemented in the **`train_xgboost_model`** component.

```
# Create the XGBoost DMatrix, which is optimized for memory efficiency and training speed
dtrain = xgb.DMatrix(X_train.values, y_train.values)

# Define model parameters
params = {"objective": "binary:logistic",
     "eval_metric": "logloss",
     "nthread": 4,  # num_cpu
     "tree_method": "hist",
     "learning_rate": 0.02,
     "max_depth": 10,
     "min_child_weight": 6,
     "n_jobs": 4,  # num_cpu,
     "verbosity": 1}

# Train initial XGBoost model
clf = xgb.train(params = params, dtrain = dtrain, num_boost_round = 500)
```

Convert the Trained XGBoost Model to daal4py

daal4py is the Python API of the oneAPI Data Analytics Library, oneDAL. daal4py helps to further optimize model prediction, or inference, on CPUs. The following code is implemented in the **`convert_xgboost_to_daal4py`** and the **`daal4py_inference`** component.

```
# Convert XGBoost model to daal4py
d4p_model = d4p.mb.convert_model(clf)

# For optimized inference on CPU
d4p_predictions = d4p_model.predict(X_test)
```

Enable Intel Extension for Scikit-Learn

Intel Extension for Scikit-Learn* provides CPU accelerations for many scikit-learn libraries. Below is an example using the scikit-learn extension to accelerate the computation of the ROC curve. The following code is implemented in the **`plot_roc_curve`** component.

```
# Call patch_sklearn() before importing scikit-learn libraries
from sklearnex import patch_sklearn
patch_sklearn()
from sklearn.metrics import roc_curve

# Calculate the ROC curve using the CPU-accelerated version
fpr, tpr, thresholds = roc_curve(
     y_true = prediction_data['y_test'],
     y_score = prediction_data['probability_score'],
     pos_label = 1)
```

Next Steps

Download the module from GitHub ›

Register for office hours to get help on your implementation ›

Check out the full suite of Intel® Optimized Cloud Modules ›

Come chat with us on our DevHub Discord* server to keep interacting with other developers ›

Intel® technologies may require enabled hardware, software, or service activation. Learn more at intel.com or from the OEM or retailer. Your costs and results may vary.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

Optimization notice: Intel® compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel® microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel® microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product user and reference guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804. https://software.intel.com/en-us/articles/optimization-notice

Software and workloads used in performance tests may have been optimized for performance only on Intel® microprocessors.

Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations, and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. See backup for configuration details. For more complete information about performance and benchmark results, visit intel.com/benchmarks.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See configuration disclosure for details. No product or component can be absolutely secure.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and noninfringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

*Other names and brands may be claimed as the property of others.

1121/SS/CMD/PDF