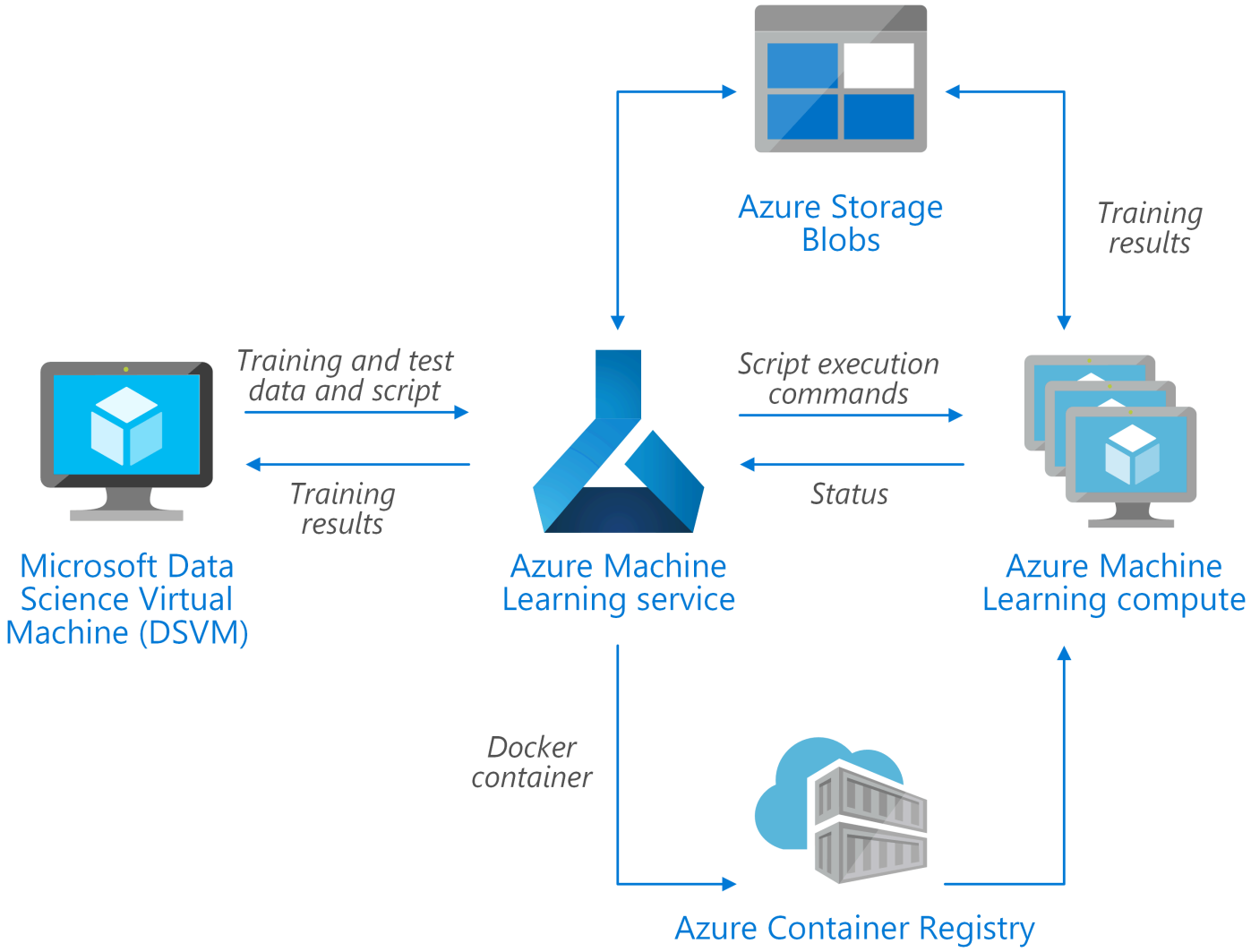
Training of Python scikit-learn models on Azure

06/01/2019 • 5 minutes to read • Contributors

In this article

This reference architecture shows recommended practices for tuning the hyperparameters (training parameters) of a scikit-learn Python model. A reference implementation for this architecture is available on GitHub.



Scenario: The problem addressed here is Frequently Asked Question (FAQ) matching. This scenario uses a subset of Stack Overflow question data that includes original questions tagged as JavaScript, their duplicate questions, and their answers. It tunes a scikit-learn pipeline to predict the probability that a duplicate question matches one of the original questions.

Processing in this scenario involves the following steps:

1. The training Python script is submitted to the Azure Machine Learning service.
2. The script runs in Docker containers that are created on each node.
3. That script retrieves training and testing data from Azure Storage.
4. The script learns a model from the training data using its combination of training parameters.
5. The model's performance is evaluated on the testing data, and is written to Azure Storage.
6. The best performing model is registered with the Azure Machine Learning service.

See also considerations for training deep learning models with GPUs.

Architecture

This architecture consists of several Azure cloud services that scale resources according to need. The services and their roles in this solution are described below.

Microsoft Data Science Virtual Machine (DSVM) is a VM image configured with tools used for data analytics and machine learning. Both Windows Server and Linux versions are available. This deployment uses the Linux editions of the DSVM on Ubuntu 16.04 LTS.

Azure Machine Learning service is used to train, deploy, automate, and manage machine learning models at cloud scale. It's used in this architecture to manage the allocation and use of the Azure resources described below.

Azure Machine Learning Compute is the resource used to train and test machine learning and AI models at scale in Azure. The compute target in this scenario is a cluster of nodes that are allocated on demand based on an automatic scaling option. Each node is a VM that runs a training job for a particular hyperparameter set.

Azure Container Registry stores images for all types of Docker container deployments. These containers are created on each node and used to run the training Python script. The image used in the Machine Learning Compute cluster is created by the Machine Learning service in the local run and hyperparameter tuning notebooks, and then is pushed to Container Registry.

Azure Blob storage receives the training and test data sets from the Machine Learning service that are used by the training Python script. Storage is mounted as a virtual drive onto each node of a Machine Learning Compute cluster.

Performance considerations

Each set of hyperparameters runs on one node of the Machine Learning compute target. For this architecture, each node is a Standard D4 v2 VM, which has four cores. This architecture uses a LightGBM classifier for machine learning, a gradient boosting framework. This software can run on all four cores at the same time, speeding up each run by a factor of up to four. That way, the whole hyperparameter tuning run takes up to one-quarter of the time it would take had it been run on a Machine Learning Compute target based on Standard D1 v2 VMs, which have only one core each.

The maximum number of Machine Learning Compute nodes affects the total run time. The recommended minimum number of nodes is zero. With this setting, the time it takes for a job to start up includes some minutes for auto-scaling at least one node into the cluster. If the hyperparameter tuning runs for a short time, however, scaling up the job adds to the overhead. For example, a job can run in under five minutes, but scaling up to one node might take another five minutes. In this case, setting the minimum to one node saves time but adds to the cost.

Monitoring and logging considerations

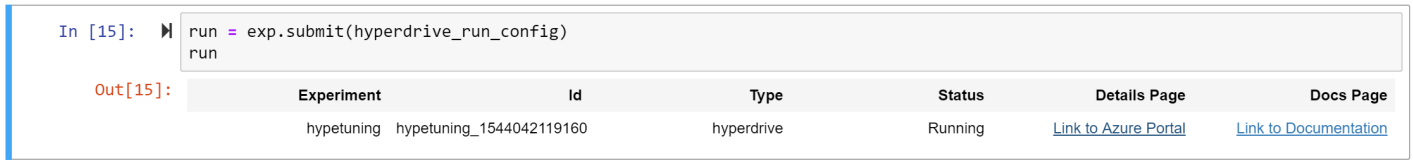
Submit a HyperDrive run configuration to return a Run object for use in monitoring the run's progress.

RunDetails Jupyter Widget

Use the Run object with the RunDetails Jupyter widget to conveniently monitor its progress at queuing and when running its children jobs. It also shows the values of the primary statistic that they log in real time.

Azure portal

Print a Run object to display a link to the run's page in Azure portal like this:



Use this page to monitor the status of the run and its children runs. Each child run has a driver log containing the output of the training script it has run.

Cost considerations

The cost of a hyperparameter tuning run depends linearly on the choice of Machine Learning Compute VM size, whether low-priority nodes are used, and the maximum number of nodes allowed in the cluster.

Ongoing costs when the cluster is not in use depend on the minimum number of nodes required by the cluster. With cluster autoscaling, the system automatically adds nodes up to the allowed maximum to match the number of jobs, and then removes nodes down to the requested minimum as they are no longer needed. If the cluster can autoscale down to zero nodes, it does not cost anything when it is not in use.

Security considerations

Restrict access to Azure Blob Storage

This architecture uses storage account keys to access the Blob storage. For further control and protection, consider using a shared access signature (SAS) instead. This grants limited access to objects in storage, without needing to hard-code the account keys or save them in plaintext. Using a SAS also helps to ensure that the storage account has proper governance, and that access is granted only to the people intended to have it.

For scenarios with more sensitive data, make sure that all of your storage keys are protected, because these keys grant full access to all input and output data from the workload.

Encrypt data at rest and in motion

In scenarios that use sensitive data, encrypt the data at rest—that is, the data in storage. Each time data moves from one location to the next, use SSL to secure the data transfer. For more information, see the Azure Storage security guide.

Secure data in a virtual network

For production deployments, consider deploying the cluster into a subnet of a virtual network that you specify. This allows the compute nodes in the cluster to communicate securely with other virtual machines or with an on-premises network. You can also use service endpoints with blob storage to grant access from a virtual network or use a single-node NFS inside the virtual network with Azure Machine Learning service.

Deployment

To deploy the reference implementation for this architecture, follow the steps in the GitHub repo.

Next steps

To operationalize the model, see Machine learning operationalization (MLOps) for Python models using Azure Machine Learning.