|  |  |  |  |
| --- | --- | --- | --- |
| **Product** | **What it is** | **What you can do with it** | **Features** |
| Azure Machine Learning Service | Managed cloud service for Machine Learning. Can be used for Data Engineering, Data Science and Development | Train, deploy, and manage models, in Azure using Python, the Azure CLI, and the portal  **See below for details** | Azure ML Service SDK  Python  Sklearn, TF, pytorch, CNTK, MXNet  Jupyter Notebook, Visual Studio, PyCharm other IDEs  Container Services  Kubernetes  IoT Edge  Field programmable gate array (FPGA) |
| Azure Machine Learning Studio | Drag-and-drop visual interface for Machine Learning | Build, experiment, and deploy models using preconfigured algorithms. Ideal for learning about machine learning | Data Vis  Drag and drop  Azure Notebooks  Project Samples |
| Azure Databricks | Spark-based analytics platform with an integrated notebook interface that seamlessly integrates with Azure AD and data services | Build and deploy models and data workflows with Big Data. To use Azure Databricks, you provision an Azure Databricks Workspace using the portal. From there, you can add users you want to have access to the workspace and refine the level of access from with Azure Databricks. Switch to another language if you need to. Select Datbricks from services, under ‘common tasks’, option to start new cluster. Then you can run a notebook, import data from different sources and perform ML processes | Databricks notebooks support SQL, Python, R Scala and Java  Active Directory Integration  Job Scheduler  Local built-in blob storage  Azure Data Services (Blob, Azure SQLDB, Azure DW, CosmosDB)  GUI to create clusters  Spark optimizations  Workspaces  Granular Security Control |
| Azure Data Science Virtual Machine | A virtual machine with pre-installed data science tools | Develop machine learning solutions in a pre-configured data science environment. To use a Windows based DSVM, you can Remote Desktop into it using the administrator credentials. Types include: OS supported, DL virtual machine, GeoAI DVSM | SQL Server / ML Server  Microsoft R Open  Jupyter Notebooks  Anaconda Python  Sample code |
| SQL Server / Machine Learning Server | Integrated with SQL Server, this scalable analytics server supports the Python and R language. | Build and develop models inside on-premises SQL server that scale to match the SQL Server engine. Note: Machine Learning Server is also available as a cluster type in HDInsight. | Commercial High Scale Python and R modules |

**HDINsight**

HDInsight is an Azure Platform as a Service (PaaS) Apache Hadoop offering. Spark on HDInsight support massive scale for data querying, wrangling, and machine learning model training. A cluster has a master server that controls the other servers of the cluster. By working on a problem split among multiple machines, performance is greatly enhanced. For HDInsight to run, it needs Azure Blob storage because it needs a place to store data. A version of HDInsight Spark is offered that includes R Server as a processing front end. Performing data science using HDInsight Spark is simple because once you spin up a Spark cluster, you can run everyday tools such as Jupyter Notebook and Zeppelin Notebook

**Azure ML Service**

Create Workspace in Portal

1. Login in to the Azure portal, and find the search bar on top of the dashboard
2. In the search bar, enter machine learning service workspaces, you will find the item under services, click on it
3. Click Add button on top left then enter the information needed to create the workspace. Below are descriptions for each field.

|  |  |
| --- | --- |
| **Field** | **Description** |
| Workspace name | Enter a unique name for your workspace. In this example, you use **docs-ws**. The names must be unique across the resource group. Use a name that's easy to remember and different from workspaces created by others. |
| Subscription | Select the Azure subscription that you want to use. |
| Resource group | Use an existing resource group in your subscription or enter a name to create a new resource group. A resource group is a container that holds related resources for an Azure solution. |
| Location | Select the location closest to your users and the data resources. This location is where the workspace is created. |

1. It may take a few minutes for the workspace to be created. The bell icon in the Azure portal will show a moving line under it while Azure is creating the workspace. Once the line disappears, click on the bell and select ‘Go to resource’.

Workspace resources

* A storage account - used to store files used by the workspace as well as data for experiments and model training.
* An Application Insights instance, used to monitor predictive services in the workspace.
* An Azure Key Vault instance, used to manage secrets such as authentication keys and credentials used by the workspace.
* Virtual Machines, and their associated virtual hardware resources, used to provide compute for notebook development in the workspace.
* A container registry, used to manage containers for deployed models.
* Role Based Access Control - You can assign role-based authorization policies to a workspace, enabling you to manage permissions that restrict what actions specific Azure Active Directory (AAD) principals can perform.

## **Types of datastore**

Azure Machine Learning supports the creation of datastores for multiple kinds of Azure data source, including:

* Azure Storage (blob and file containers) - included in workspace
* Azure Data Lake Storage
* Azure SQL Database
* Azure Databricks file system (DBFS)

## **Types of dataset**

Datasets are typically based on files in a datastore, though they can also be based on URLs and other sources. You can create the following types of dataset:

* **Tabular**: The data is read from the dataset as a table. You should use this type of dataset when your data is consistently structured and you want to work with it in common tabular data structures, such as Pandas dataframes.
* **File**: The dataset presents a list of file paths that can be read as though from the file system. Use this type of dataset when your data is unstructured, or when you need to process the data at the file level (for example, to train a convolutional neural network from a set of image files).

**Compute Targets**

In Azure Machine Learning, *Compute Targets* are physical or virtual computers on which experiments are run.

**Types of Compute**

### **Local compute**

You can specify a *local* compute target for most processing tasks in Azure Machine Learning. This runs the experiment on the same compute target as the code used to initiate the experiment, which may be your physical workstation or a virtual machine such as an Azure Machine Learning compute instance on which you are running a notebook.

Local compute is generally a great choice during development and testing with low to moderate volumes of data.

### **Training clusters**

For training workloads with high scalability requirements, you can use Azure Machine Learning training clusters; which are multi-node clusters of Virtual Machines that automatically scale up or down to meet demand. This is a cost-effective way to run experiments that need to handle large volumes of data or use parallel processing to distribute the workload and reduce the time it takes to run.

### **Inference clusters**

To deploy trained models as production services, you can use Azure Machine Learning inference clusters, which use containerization technologies to enable rapid initialization of compute for on-demand inferencing.

### **Attached compute**

If you already use an Azure-based compute environment for data science, such as a virtual machine or an Azure Databricks cluster, you can attach it to your Azure Machine Learning workspace and use it as a compute target for certain types of workload.

### **Types of Pipeline Steps**

Common kinds of step in an Azure Machine Learning pipeline include:

* **PythonScriptStep**: Runs a specified Python script.
* **EstimatorStep**: Runs an estimator.
* **DataTransferStep**: Uses Azure Data Factory to copy data between data stores.
* **DatabricksStep**: Runs a notebook, script, or compiled JAR on a databricks cluster.
* **AdlaStep**: Runs a U-SQL job in Azure Data Lake Analytics.

## **The OutputDatasetConfig object**

The **OutputDatasetConfig** object is a special kind of **DataReference** that:

* References a location in a datastore.
* Creates a data dependency between pipeline steps.

## **OutputDatasetConfig step inputs and outputs**

To use a **OutputDatasetConfig** object to pass data between steps, you must:

1. Define a named **OutputDatasetConfig** object that references a location in a datastore.
2. Specify the **OutputDatasetConfig** object as an *input* or *output* for the steps that use it.
3. Pass the **OutputDatasetConfig** object as a script parameter in steps that run scripts (and include code in those scripts to read or write data)

After you have created a pipeline, you can publish it to create a REST endpoint through which the pipeline can be run on demand.

You can increase the flexibility of a pipeline by defining parameters.

After you have published a pipeline, you can initiate it on demand through its REST endpoint, or you can have the pipeline run automatically based on a periodic schedule or in response to data updates.

**Deploying a Model**

You can deploy a model as a real-time web service to several kinds of compute target, including local compute, an Azure Machine Learning compute instance, an Azure Container Instance (ACI), an Azure Kubernetes Service (AKS) cluster, an Azure Function, or an Internet of Things (IoT) module. Azure Machine Learning uses *containers* as a deployment mechanism, packaging the model and the code to use it as an image that can be deployed to a container in your chosen compute target.

Steps:

## 1. Register a trained model to Azure ML workspace

## 2. Define an Inference Configuration

* + Define entry script to load model
    - init()
    - run(raw\_data)
  + Define environment to run

## 3. Define a Deployment Configuration

* Define compute target

## 4. Deploy the Model

After deploying a real-time service, you can consume it from client applications to predict labels for new data cases.

### **Authentication**

In production, you will likely want to restrict access to your services by applying authentication. There are two kinds of authentication you can use:

* **Key**: Requests are authenticated by specifying the key associated with the service.
* **Token**: Requests are authenticated by providing a JSON Web Token (JWT).

By default, authentication is disabled for ACI services, and set to key-based authentication for AKS services (for which primary and secondary keys are automatically generated). You can optionally configure an AKS service to use token-based authentication (which is not supported for ACI services).

**Troubleshooting**

There are a lot of elements to a real-time service deployment, including the trained model, the runtime environment configuration, the scoring script, the container image, and the container host. Troubleshooting a failed deployment, or an error when consuming a deployed service can be complex.

**AutoML**

You can use automated machine learning in Azure Machine Learning to train models for the following types of machine learning task:

* Classification
* Regression
* Time Series Forecasting

By default, automated machine learning will randomly select from the full range of algorithms for the specified task. You can choose to block individual algorithms from being selected; which can be useful if you know that your data is not suited to a particular type of algorithm.

## **Scaling and Normalization**

Automated machine learning applies scaling and normalization to numeric data automatically, helping prevent any large-scale features from dominating training. During an automated machine learning experiment, multiple scaling or normalization techniques will be applied.

## **Optional Featurization**

You can choose to have automated machine learning apply preprocessing transformations such as:

* Missing value imputation to eliminate nulls in the training dataset.
* Categorical encoding to convert categorical features to numeric indicators.
* Dropping high-cardinality features, such as record IDs.
* Feature engineering (for example, deriving individual date parts from DateTime features)
* Others...

### **Specifying Data for Training**

Automated machine learning is designed to enable you to simply bring your data, and have Azure Machine Learning figure out how best to train a model from it.

When using the Automated Machine Learning user interface in Azure Machine Learning studio, you can create or select an Azure Machine Learning [*dataset*](https://aka.ms/AA6zxeb) *to be used as the input for your automated machine learning experiment.*

When using the SDK to run an automated machine learning experiment, you can submit the data in the following ways:

* Specify a dataset or dataframe of *training data* that includes features and the label to be predicted.
* Optionally, specify a second *validation data* dataset or dataframe that will be used to validate the trained model. if this is not provided, Azure Machine Learning will apply cross-validation using the training data.

Alternatively:

* Specify a dataset, dataframe, or numpy array of *X* values containing the training features, with a corresponding *y* array of label values.
* Optionally, specify *X\_valid* and *y\_valid* datasets, dataframes, or numpy arrays of *X\_valid* values to be used for validation.

### **Specifying the Primary Metric**

One of the most important settings you must specify is the primary\_metric. This is the target performance metric for which the optimal model will be determined.

**Azure DSVM**

Data scientists typically need to install and configure a variety of software for machine learning development. Once created, these environments frequently need to be reconfigured for additional data science projects. The Azure Data Science Virtual Machines (Azure DSVM) makes it easy to maintain consistency in the environments in which you are working.

Additionally, each DSVM provides code samples in the form of Jupyter notebooks and scripts in languages such as Python and R to help you learn about Microsoft on-premises and Azure-based machine learning services.

Currently there are **two different types of Azure DSVM: Windows DSVMs and Linux DSVMs**. Azure offers a Windows Server 2016 and Windows Server 2012 version. Linux DSVMs offer Ubuntu 16.04 LTS and CentOS 7.4 versions.

The Deep Learning DSVM comes preconfigured and preinstalled with many deep learning tools. Training deep learning models is computationally intense. For better model training performance select a high-speed GPU-based machine, which will significantly speed up machine model training. The Windows Server 2016 edition of DSVM comes preinstalled with GPU drivers, frameworks, and GPU versions of deep learning frameworks. The Linux edition has deep learning on GPU enabled on both the CentOS and Ubuntu DSVMs. You can deploy the Ubuntu, CentOS, or Windows 2016 edition of DSVM to a non-GPU–based Azure VM, in which case all of the deep learning frameworks will fall back to the CPU mode.

The Geo AI DSVM has analytic capabilities optimized for geospatial and location data. It has the ArcGIS Pro geographic information system integrated into the VM to support advanced geographic-based AI questions. It also has other conventional data science tools preconfigured and preinstalled for ease-of-use.

## **DVSM Use Cases**

## **Collaborate as a team using DSVMs**

By replacing physical machines with cloud-based VMs, the data science team can easily and quickly create and modify a machine learning environment with the same baseline configuration. This ensures that all the data science team members have a consistent development environment.

## **Address issues with DSVMs**

A data science training class requires students to have an environment with the same software installed and configured so examples and labs work consistently. With the DSVM, the problems caused by package incompatibility and language versions are reduced. The students focus on the data science content, and the trainers don't have to spend class time debugging issues. The VMs can be paused or deleted, so you only pay while the machines are in use. With the DSVM, the users can even train themselves using the preloaded code samples.

## **Use on-demand elastic capacity for large-scale projects**

Data science hackathons (or *competitions*), or large-scale data modeling and exploration require scaled out hardware capacity, typically for a short duration. The DSVM can help replicate the data science environment on demand, allowing experiments requiring high-powered computing resources to be run.

## **Experiment and evaluate on a DSVM**

You can use the DSVM to learn about tools and topics such as:

* Microsoft Machine Learning Server
* SQL Server
* Microsoft Visual Studio tools
* Jupyter Notebooks
* Deep learning tools
* Popular Machine Learning toolkits
* Other new tools popular in the community

Because the DSVM can be set up quickly, you can apply it in short-term usage scenarios such as replicating published experiments, executing demonstrations, following walkthroughs in online sessions, and doing tutorials. Many sample scripts and data are provided to demonstrate how to use various Microsoft products that support data science and machine learning such as SQL Server, Machine Learning Server, Azure Machine Learning service, and Microsoft Cognitive Services.

# **Windows-Based Data Science VMs**

The tools installed on the Windows-based DSVM include the most popular data science software, frameworks, and tools to support integration of Azure services. You can use the Windows-based DSVM to jump-start your data science projects.

There are no software charges for the DSVM image. You pay only the Azure usage fees, and those depend on the size of the virtual machine you provision.

## **Features of the Windows Data Science Virtual Machine**

The Windows-based DSVM includes some features that are not available on the Linux-based DSVM.

* Tutorials available for training purposes.
* Support for Microsoft Office.
* SQL Server integrated with Machine Learning Services to support Python/R execution with SQL Server.

In addition, the Windows-based DSVM includes several preinstalled programming languages, including R, Python, SQL, and C#. You can develop, test, and run your code with Microsoft Visual Studio, or Visual Studio Code. The DSVM also comes with several other development and data access tools:

* Azure SDK to integrate Microsoft Azure cloud services into your apps
* Power BI Desktop
* Azure PowerShell
* AzCopy
* Azure Data Lake Storage
* Migration tools for various databases (DocumentDB, Azure Cosmos DB, etc.)
* Data Management Gateway

## **Data science tools**

The Windows DSVM has many preconfigured data science tools for data visualization, model training, deployment, and more. Some of these tools are:

* Azure Machine Learning SDK for Python
* Anaconda Python distribution
* Jupyter Notebook with R, Python, and Apache Spark Python (PySpark) kernels
* Microsoft Visual Studio Community
* Microsoft Power BI Desktop
* Microsoft SQL Server + Microsoft Machine Learning Services
* Apache Spark instance for local development and testing
* Julia Pro by Julia Computing
* Python/R
* Git

## **Machine Learning tools**

Available Machine Learning and Deep learning tools include:

* Azure Cognitive Services support
* H2O - Open-source AI platform that supports in-memory, distributed, fast, and scalable ML.
* TensorFlow - Python based deep learning framework.
* Chainer - Python based deep learning framework.
* Apache MXNet - Deep learning framework with support for multiple languages including C++, Python, R, and Perl.
* Keras - High-level neural networks API, written in Python.
* Vowpal Wabbit ("VW") - open-source, fast, out-of-core learning system library.
* XGBoost - distributed gradient boosting (GBDT, GBRT, or GBM) library for Python, R, Java, Scala.
* Rattle - graphical user interface for data mining by using R.
* Weka - a collection of Java-based ML algorithms for data mining tasks.
* Apache Drill - an open source SQL query engine for big data; allows data exploration without export.

# **Creating a DSVM in Azure Portal**

***Prerequisite***

To create a Windows Data Science Virtual Machine, you must have an Azure subscription. [Try Azure for free](https://azure.com/free). Please note Azure free accounts do not support GPU enabled virtual machine SKUs.

## **Create your DSVM**

1. To create a DSVM instance:
2. Go to the [Azure portal](https://portal.azure.com/) You might be prompted to sign in to your Azure account if you're not already signed in.
3. Find the virtual machine listing by typing in "data science virtual machine" and selecting "Data Science Virtual Machine - Windows 2019."
4. Select the Create button at the bottom.
5. You should be redirected to the "Create a virtual machine" blade.
6. Fill in the Basics tab:
   1. Subscription: If you have more than one subscription, select the one on which the machine will be created and billed. You must have resource creation privileges for this subscription.
   2. Resource group: Create a new group or use an existing one.
   3. Virtual machine name: Enter the name of the virtual machine. This is how it will appear in your Azure portal.
   4. Location: Select the datacenter that's most appropriate. For fastest network access, it's the datacenter that has most of your data or is closest to your physical location. Learn more about [Azure Regions](https://azure.microsoft.com/global-infrastructure/regions/).
   5. Image: Leave the default value.
   6. Size: This should auto-populate with a size that is appropriate for general workloads. Read more about [Windows VM sizes in Azure](https://docs.microsoft.com/en-us/azure/virtual-machines/windows/sizes).
   7. Username: Enter the administrator username. This is the username you will use to log into your virtual machine, and need not be the same as your Azure username.
   8. Password: Enter the password you will use to log into your virtual machine.
7. Select Review + create.
8. Review+create
   1. Verify that all the information you entered is correct.
   2. Select Create.

**Note**

You do not pay licensing fees for the software that comes pre-loaded on the virtual machine. You do pay the compute cost for the server size that you chose in the Size step.

Provisioning takes 10 to 20 minutes. You can view the status of your VM on the Azure portal.

## **Access the DSVM**

1. After the VM is created and provisioned, follow the steps listed to [connect to your Azure-based virtual machine](https://docs.microsoft.com/en-us/azure/marketplace/cloud-partner-portal/virtual-machine/cpp-connect-vm). Use the admin account credentials that you configured in the Basics step of creating a virtual machine.
2. You're ready to start using the tools that are installed and configured on the VM. Many of the tools can be accessed through Start menu tiles and desktop icons.
3. You can also attach a DSVM to Azure Notebooks to run Jupyter notebooks on the VM and bypass the limitations of the free service tier. For more information, see [Manage and configure Notebooks projects](https://docs.microsoft.com/en-us/azure/notebooks/configure-manage-azure-notebooks-projects#manage-and-configure-projects).

The DSVM provides a wide selection of languages, development, machine learning, and data visualization tools, and data platforms that enable you to accomplish most data science tasks. Some examples of those tasks are as follows:

TABLE 1

|  |  |
| --- | --- |
| **Task** | **Provided tools** |
| Find, load, and preprocess data. | Use the preinstalled data platform and ingestion tools, such as Azure Blob Storage, Azure Data Lake, Azure HDInsight Hadoop, Azure Cosmos DB, Azure SQL Data Warehouse, and databases. |
| Train and test models. | Use Jupyter Notebook, Microsoft Visual Studio, and Visual Studio Code as development tools. Select your preferred language (R or Python) to leverage the powerful tools included in the Microsoft Machine Learning Server (for example, Microsoft R Server as you saw in the earlier section). |
| Operationalize the model. | Use R and Python Azure Machine Learning to deploy your model. Then client applications can access your models using a simple web service interface. |
| Train deep learning models. | Use deep learning platforms such as PyTorch, scikit-learn, and TensorFlow, which supports GPU computation to accelerate the model training. |
| Visualize data. | Use the Microsoft Power BI desktop to build reports and dashboards, and use the one-click feature to publish to the cloud. |
| Share files and code. | Create Azure file storage as a mountable drive on your DSVM to share large-scale datasets/code with your team. Use Git clients such as Git Bash and Git GUI to access your repository and share code with your team. |
| Perform cloud administration. | Use the Azure portal or PowerShell to manage your Azure resources. Options include restarting or stopping your VM, dynamically scaling your DSVM to meet your project needs, or extending your storage space. |

**Databricks**

Azure Databricks is a fully-managed version of the open-source [Apache Spark](https://spark.apache.org/) analytics and data processing engine. Azure Databricks is an enterprise-grade and secure cloud-based big data and machine learning platform.

**Deploy an Azure Databricks workspace**

1. Open the Azure portal.
2. Click **Create a Resource** in the top left
3. Search for "Databricks"
4. Select *Azure Databricks*
5. On the Azure Databricks page select *Create*
6. Provide the required values to create your Azure Databricks workspace:
   * **Workspace Name**: Provide a name for your workspace.
   * **Subscription**: Choose the Azure subscription in which to deploy the workspace.
   * **Resource Group**: Use **Create new** and provide a name for the new resource group.
   * **Location**: Select a location near you for deployment. For the list of regions that are supported by Azure Databricks, see [Azure services available by region](https://azure.microsoft.com/regions/services/).
   * **Pricing Tier**: **Trial (Premium - 14 days Free DBUs)**. You must select this option when creating your workspace or you will be charged. The workspace will suspend automatically after 14 days. When the trial is over you can convert the workspace to **Premium** but then you will be charged for your usage.
7. Accept the terms and conditions.
8. Select **Create**

# **Apache Spark notebooks**

After workspace creation, need to setup a cluster and notebook

## **Create a cluster**

1. In the Azure portal, click **All resources** menu on the left side navigation and select the Databricks workspace you created in the last unit.
2. Select **Launch Workspace** to open your Databricks workspace in a new tab.
3. In the left-hand menu of your Databricks workspace, select **Clusters**.
4. Select **Create Cluster** to add a new cluster.
5. Enter a name for your cluster. Use your name or initials to easily differentiate your cluster from your coworkers.
6. Select the **Databricks RuntimeVersion**. We recommend the latest runtime and **Scala 2.11**.
7. Specify your cluster configuration. While on the 14 day free trial, the defaults will be sufficient. When the trial is ended, you may prefer to change Min Workers to zero. That will allow the compute resources to shut down when you are not in a coding exercise and reduce your charges.
8. Select **Create Cluster**.

## **Create a notebook**

1. On the left-hand menu of your Databricks workspace, select **Home**.
2. Right-click on your home folder.
3. Select **Create**.
4. Select **Notebook**.
5. Name your notebook **First Notebook**.
6. Set the **Language** to **Python**.
7. Select the cluster to which to attach this notebook.
8. Select Create

## **Attach and detach your notebook**

To use your notebook to run a code, you must attach it to a cluster. You can also detach your notebook from a cluster and attach it to another depending upon your organization's requirements.

If your notebook is attached to a cluster, you can:

* Detach your notebook from the cluster
* Restart the cluster
* Attach to another cluster
* Open the Spark UI
* View the log files of the driver

You can use Apache Spark notebooks to:

* Read and process huge files and data sets
* Query, explore, and visualize data sets
* Join disparate data sets found in data lakes
* Train and evaluate machine learning models
* Process live streams of data
* Perform analysis on large graph data sets and social networks

### **Delete the Azure Databricks instance**

1. Navigate to the Azure portal.
2. Navigate to the resource group that contains your Azure Databricks instance.
3. Select **Delete resource group**.
4. Type the name of the resource group in the confirmation text box.
5. Select **Delete**.

**Machine Learning Studio**

## **Create a Machine Learning Studio Workspace**

1. In a web browser, go to [Machine Learning Studio](https://studio.azureml.net/) .
2. Under the **Sign In** button, select the **Sign up here** link if you're not a current Machine Learning Studio user.
3. Select one of the free options. If you have a Microsoft account, we recommend using the **Free Workspace** option. You'll need this to complete the hands-on parts of the units that follow.

Use Azure Account to create ML Studio workspace

1. Sign in to the [Azure portal](https://portal.azure.com/) .
2. On the dashboard, select **Create a resource**.
3. In the search bar, enter *Machine Learning Studio Workspace*.
4. At the bottom of the window, select **Create**.
5. Enter your workspace information
6. Select **Create** and wait until the resource is created and deployed.

Now you can access your new Machine Learning Studio Workspace in one of two ways:

* Go to the resource in your Azure portal. On the Overview page, select **Launch Machine Learning Studio** to open the workspace in Machine Learning Studio.
* Sign in to the [Machine Learning Studio portal](https://studio.azureml.net/) .
  + In the upper-right corner, select your workspace.
  + Select **my experiments**.

**Create an Experiment**

1. Download and extract the [Building Data.zip](https://github.com/MicrosoftDocs/mslearn-work-with-vision-cognitive-services/blob/master/Building%20Data.zip?raw=true) file. This is the dataset.
2. Go to the [Machine Learning Studio portal](https://studio.azureml.net/) .
3. Sign in by using the Microsoft account that's associated with your Azure account.
4. On the left navigation bar of Machine Learning Studio, select the **Datasets** icon.
5. In the lower-left corner, select the **New** button to upload the dataset.
6. Select **DataSet** > **From Local File**.
7. Select **Choose File** and then select the file you just downloaded.
8. In the lower-right corner, select the check mark to upload the dataset.
9. Verify that **Building Data.csv** is listed as a dataset.
10. In Machine Learning Studio, select the **Experiments** icon.
11. Select **New** > **Blank Experiment**.
12. Select **Saved Datasets** > **My Datasets** > **Building Data.csv**.
13. Drag this dataset onto the designer surface.

# **Build the experiment**

1. Expand **Data Transformation** > **Sample and Split**.
2. Drag a **Split Data** operation onto the designer surface.
3. On the right, set the fraction of rows in the first output dataset to **0.7** to split the data at 70/30.
4. Connect the dataset to the Split Data operation by dragging the dataset handle to the Split Data operation.
5. On the left, expand **Machine Learning** > **Initialize Model** > **Regression**.
6. Drag the **Decision Forest Regression** entry onto the designer surface.
7. On the left, expand the **Score** subcategory and locate **Score Model**.
8. Drag the **Score Model** operation onto the designer.
9. On the left, expand **Train** and drag a **Train Model** operation onto the designer.
10. On the left, expand the **Evaluate** subcategory of Machine Learning and drag an **Evaluate Model** operation onto the designer.
11. Connect output 1 of the **Split Data** operation to the **Train Model** operation.
12. Select the **Train Model** operation.
13. On the right, select **Launch Column Selector**.
14. Select the **With Rules** entry. Then click inside the empty column list and select the **Wall Area** column name.
15. Select the check mark to accept the selection.
16. Connect the **Decision Forest Regression** operation to the remaining input of the **Train Model** operation.
17. Connect the second output of the **Split Data** operation to the **Score Model** operation.
18. Connect the output of the **Score Model** operation to the left input of the **Evaluate Model** operation.
19. Connect the output of the **Train Model** operation to the remaining **Score Model** input. Your experiment should now look like this:
20. At the bottom of the designer window, run the experiment by selecting the **Run** button.
21. Make sure all of the operation boxes show green check marks and that the upper-right corner of the designer displays **Finished running** with a green check mark.
22. Save your experiment as **Energy Efficiency Regression**.

# **Prepare and deploy the experiment**

1. On the left navigation bar of Machine Learning Studio, select the **Experiments** icon.
2. Select your completed **Energy Efficiency Regression** experiment to open it.
3. Depending on the state of the web service, you might not have to run your experiment again. If the upper-right corner of the designer displays **Finished running**, and your stages show green check marks, you're ready for the next step. If not, select the **Run** button.

## **Set up and deploy the web service**

1. After the energy-efficiency experiment has run, at the bottom of the designer, hover over the **Set Up Web Service** button.
2. Select **Predictive Web Service** [Recommended].
3. Verify that the **Predictive experiment** tab opens. The designer should show a web service input and web service output operation.
4. Select the **Run** button.
5. Verify successful completion by looking for a green check mark in the **Score Model** operation. Also look for a **Finished running** note in the upper-right corner.
6. Select **Deploy Web Service**

## **Test the web service**

Now that the web service is deployed, let’s test it to see the output results.

1. Next to the **Request/Response** option, select the **Test** button to open the data entry dialog box.
2. In the data entry dialog box, enter the following values from the training model data:
   * Wall area: **296**
   * Roof area: **110.25**
   * Overall height: **7**
   * Glazing area: **0**
   * Heating load: **15.55**
3. In the lower-right corner, select the check mark.

To see an example of the returned JSON code, select the **Details** button

## **View the request and response headers**

1. Under **API Help Page for the Default Endpoint**, select the **Request/Response** link. Machine Learning Studio opens another browser tab or window to show the API documentation for this web service.
2. Scroll down to **Request Headers** and **Request Body** to get an idea of how the request will be formed. This request will be sent to the web service for evaluation. It's in JSON format.
3. Scroll down more and evaluate the **Response Headers** and **Response Body** sample. This section shows the format of the data that will be returned along with a sample JSON response. Use this sample to understand how to parse the JSON code for the data you want to use in your application.
4. Scroll down more to find information about the input and output parameters. These parameters include the expected names and data types. If you create a class to represent the returned information, you can use these parameters to create the proper data types for your member variables. The last two returned items map a **Scored Label Mean** value to a **Scored Label Standard Deviation** value.

## **Evaluate the sample code**

1. Scroll all the way to the bottom of the web service API documentation window.
2. Select the tab of the language you're most familiar with. Then review the code to see what it's doing.
3. Notice the placeholder for the API key. You'll need to replace the value **abc123** with a valid API key.