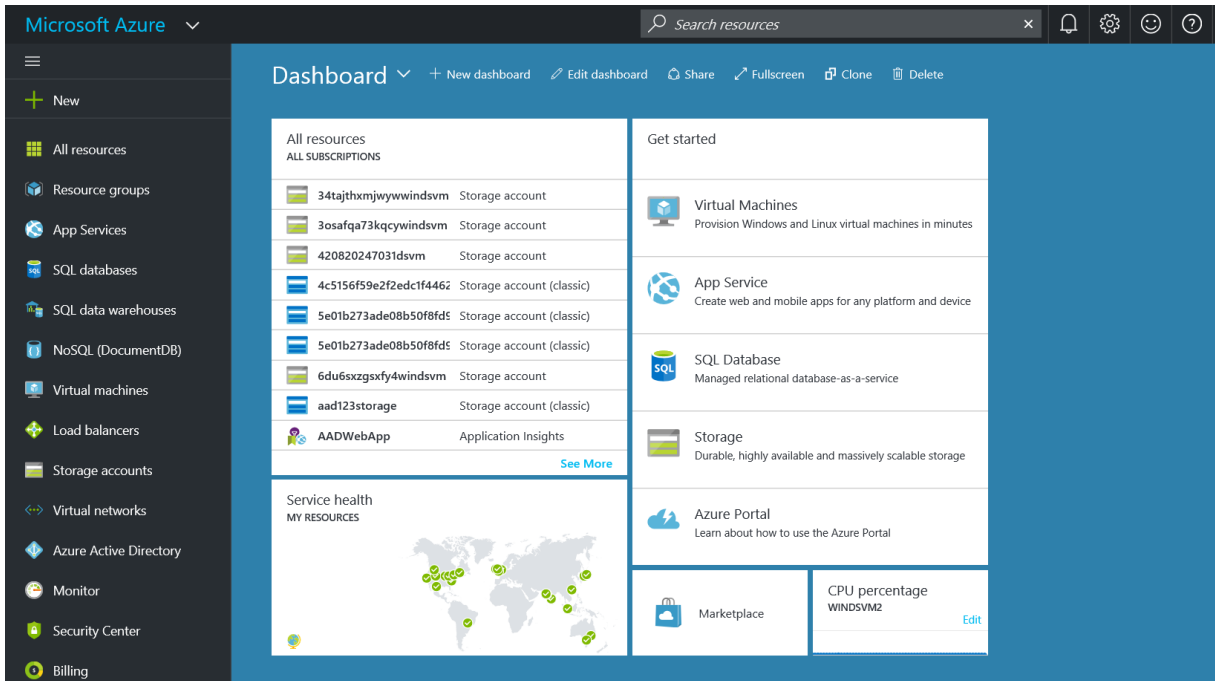

Data Science Virtual Machine – Introductory Hands-On Workshop

Contents

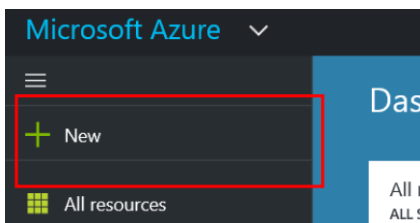
Deploying a Windows Data Science Virtual Machine – Using the Azure portal	2
Connecting to a Data Science Virtual Machine – Using Remote Desktop & Exploring the VM and some of the installed tools.....	12
Creating an Azure Storage Account and a 5TB Shared File store and Mounting the Shared Storage as a drive on the VM	20
Typical Data Science Walkthrough using the DSVM – From Getting Data and Creating Models to publishing Trained models as web services and consuming them for scoring – (R, Jupyter Notebooks, RTVS, Azure ML)	26
Conclusion + Next Steps.....	40

Deploying a Windows Data Science Virtual Machine – Using the Azure portal

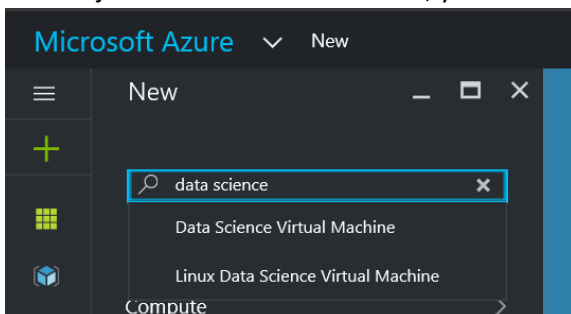
1. Go to <http://portal.azure.com>
Sign in with your Azure subscription credentials and you will see the portal dashboard.



2. Click on New







3. Type in data science in the search bar and then press Enter. You may also click on the suggestions, but if u just search for Data Science, you will see all options in the results.




4. The search results should look like the figure below. Choose the version of the Data Science Virtual Machine (DSVM) that you wish to use, i.e. Windows or Linux (This walkthrough is for the Windows version, but the Linux version is very similar). The third option is the 'Deep Learning Toolkit for the DSVM', this is designed to be deployed on the NC class hardware instances in Azure that offer GPU based parallel processing. There are other specific walkthroughs for this.

Results







NAME	PUBLISHER	CATEGORY
 Data Science Virtual Machine	Microsoft	Compute
 Linux Data Science Virtual Machine	Microsoft	Compute
 Algebraix Analytics Enterprise	Algebraix Data	Compute
 Deep Learning toolkit for the DSVM	Microsoft	Compute

5. Once you close the VM version, you will see the next frame that has some specific information about the image you chose. You can click 'Create' to start the creation process. The deployment model for DSVM is pre-set to be Azure Resource Manager (ARM).

**Data Science Virtual Machine**
Microsoft


The Data Science Virtual Machine running on a Windows Server 2012 contains popular tools for data exploration, modeling and development activities. The main tools include Microsoft R Server Developer Edition (An enterprise ready scalable R framework) , Anaconda Python distribution, Jupyter notebooks for Python and R, Visual Studio Community Edition with Python, R and node.js tools, Power BI desktop, SQL Server 2016 Developer edition including support In-Database analytics using Microsoft R Server. It also includes open source deep learning tools like Microsoft Cognitive Toolkit (CNTK 2.0) and mxnet; ML algorithms like xgboost, Vowpal Wabbit. The Azure SDK and libraries on the VM allows you to build your applications using various services in the cloud that are part of the Cortana Analytics Suite which includes Azure Machine Learning, Azure data factory, Stream Analytics and SQL Datawarehouse, Hadoop, Data Lake, Spark and more.

Jump start modeling and development for your data science project using software commonly used for analytics and machine learning tasks in a variety of languages including R, Python, SQL, C# all pre-installed. Visual Studio provides an easy to use IDE to develop and test your code. Jupyter notebooks offers a browser based experimentation and development environment for both Python and R. Microsoft R Services Developer edition included in the VM comes with Microsoft's ScaleR package in R that enables high-performance, scalable, parallelized, and distributed "Big Data Analytics." SQL Server 2016 Developer edition includes SQL Server R Services that supports scaleable In-Database analytics using Microsoft R enabling you to run analytics close to the data within the database server efficiently. Deep learning tools on the VM like CNTK2.0 (with Python interfaces) and mxNet (with Python, R interfaces) help you work on AI domains like Vision, Text, Language understanding.



PUBLISHERMicrosoft

USEFUL LINKS[Learn More](#)
[Data Science Process](#)
[How-To Guide to the Data Science Virtual Machine](#)

Select a deployment model 

Resource Manager

Create

Want to deploy programmatically? Get started →

6. Once you press 'Create' you will arrive at the 'Create Virtual Machine' frame. Here, you will be going through 5 stages of creation: Basics >> Size >> Settings >> Summary >> Buy.

The screenshot shows the Microsoft Azure portal interface for creating a virtual machine. The breadcrumb navigation at the top reads: Microsoft Azure > New > Marketplace > Everything > Data Science Virtual Machine > Create. The main window is titled 'Create virtual machine' and has a sub-header 'Basics'. On the left, a vertical sidebar contains icons for various Azure services. The main content area is divided into two parts. The left part shows a progress bar with five steps: 1. Basics (Configure basic settings), 2. Size (Choose virtual machine size), 3. Settings (Configure optional features), 4. Summary (Data Science Virtual Machine), and 5. Buy. The 'Basics' step is currently selected and highlighted in light blue. The right part of the main content area contains the configuration fields for the 'Basics' step. These fields include: 'Name' (a text input field with the value 'NameYourDSVM' and a green checkmark), 'VM disk type' (a dropdown menu with 'SSD' selected), 'User name' (a text input field), 'Password' (a text input field), 'Confirm password' (a text input field), 'Subscription' (a dropdown menu with 'Microsoft Azure Internal Consumption (1a)' selected), 'Resource group' (radio buttons for 'Create new' and 'Use existing', with 'Create new' selected), and 'Location' (a dropdown menu with 'East US' selected). At the bottom right of the main content area, there is a blue button labeled 'OK'.

Microsoft Azure > New > Marketplace > Everything > Data Science Virtual Machine > Create

Create virtual machine Basics

1 Basics
Configure basic settings >

2 Size
Choose virtual machine size >

3 Settings
Configure optional features >

4 Summary
Data Science Virtual Machine >

5 Buy >

* Name
NameYourDSVM ✓

VM disk type ⓘ
SSD ▼

* User name
[Text Input]

* Password
[Text Input]

* Confirm password
[Text Input]

Subscription
Microsoft Azure Internal Consumption (1a) ▼

* Resource group ⓘ
☒ Create new ☐ Use existing
[Text Input]

Location
East US ▼

OK

7. Fill out the fields in the 'Basics' tab, and chose the subscription if you have more than one and choose the data center region/location where you want your VM to be deployed to. Typically, geographically closest is the most common choice. Press 'OK' when you are done.

Microsoft Azure

Create virtual machine

Basics

1 Basics
Configure basic settings

2 Size
Choose virtual machine size

3 Settings
Configure optional features

4 Summary
Data Science Virtual Machine

5 Buy

* Name
NameYourDSVM

VM disk type
SSD

* User name
yourusername

* Password
.....

* Confirm password
.....

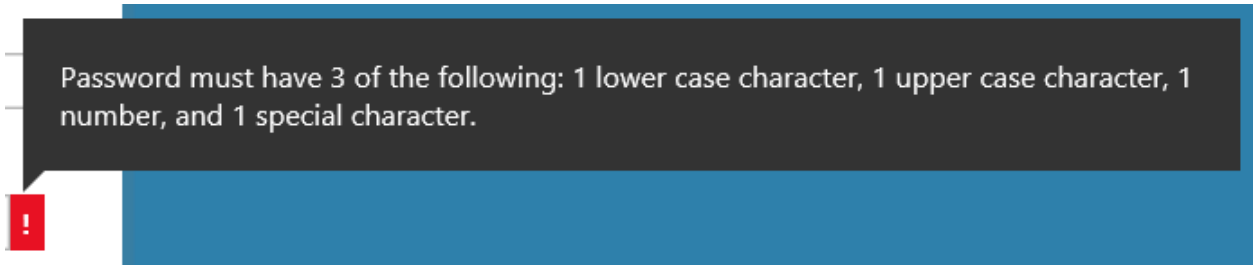
Subscription
Microsoft Azure Internal Consumption (1a)

* Resource group
☒ Create new ☐ Use existing
rg-NameYourDSVM

Location
East US
East US
East US 2
Japan East
Japan West
North Central US
North Europe
South Central US

OK

-
8. You may see a warning/error when entering your password as it would need to adhere to some complexity requirements, for security reasons, refer to the tooltip for exact requirements.



9. Additionally, you may either chose to create a new resource group to encapsulate all the related resources for your VM. This is a logical reference to collectively refer to related resources. You may also choose an existing resource group. Be sure to name resource groups in a manner where they are self-explanatory of the resources they encapsulate or their purpose. Click 'OK' when done.

*** Resource group** ⓘ

☒ Create new ☐ Use existing

*** Resource group** ⓘ

☒ Create new ☐ Use existing

Location

11. On the 'Size' tab, select the desired configuration of the machine that you want. Since the purpose of this instance is 'Data Science', we recommend choosing an instance size with 4 or more cores and 14GB or more of RAM. You can scale the instances up or down as per your requirement. Additionally, you can also shut down and deallocate the instances when not in use. This may be done programmatically as well. Click 'Select' to proceed to the 'Settings' tab.

Microsoft Azure Marketplace > Everything > Data Science Virtual Machine > Create virtual machine > Choose a size

Create virtual machine

Choose a size
Browse the available sizes and their features

1 Basics Done ✓

2 Size Choose virtual machine size >

3 Settings Configure optional features >

4 Summary Data Science Virtual Machine >

5 Buy >

DS1_V2 Standard	DS2_V2 Standard	DS3_V2 Standard
1 Cores	2 Cores	4 Cores
3.5 GB	7 GB	14 GB
2 Data disks	4 Data disks	8 Data disks
3200 Max IOPS	6400 Max IOPS	12800 Max IOPS
7 GB Local SSD	14 GB Local SSD	28 GB Local SSD
Load balancing	Load balancing	Load balancing
Premium disk support	Premium disk support	Premium disk support
104.16 USD/MONTH (ESTIMATED)	208.32 USD/MONTH (ESTIMATED)	416.64 USD/MONTH (ESTIMATED)

DS4_V2 Standard	DS5_V2 Standard	DS11_V2 Standard
8 Cores	16 Cores	2 Cores
28 GB	56 GB	14 GB
16 Data disks	32 Data disks	4 Data disks
25600 Max IOPS	51200 Max IOPS	6400 Max IOPS
56 GB Local SSD	112 GB Local SSD	28 GB Local SSD
Load balancing	Load balancing	Load balancing
Premium disk support	Premium disk support	Premium disk support
833.28 USD/MONTH (ESTIMATED)	1,666.56 USD/MONTH (ESTIMATED)	245.52 USD/MONTH (ESTIMATED)

DS12_V2 Standard	DS13_V2 Standard	DS14_V2 Standard
4 Cores	8 Cores	16 Cores
28 GB	56 GB	112 GB
8 Data disks	16 Data disks	32 Data disks
12800 Max IOPS	25600 Max IOPS	50000 Max IOPS
56 GB Local SSD	112 GB Local SSD	224 GB Local SSD

Select

13. On the 'Settings' tab you can select or specify different locations or names etc. for some resources like storage and network. The form is pre-populated with some generated values that are automatically built from the initial names etc. that you specified in the 'Basics' tab. You can choose to change these or use these generated values. Additionally, there are a few settings to note. Here is where you can click on 'Extensions' and add a post provisioning extension like an Anti-Malware solution etc. or bring in your custom scripts and extensions. You can also enable 'Guest OS diagnostics' which is off by default. This gives you metrics every minute for the VM and can use them to create alerts and stay informed on your applications etc. A service like Operations Management Suite can ingest these feeds, they are stored in the specified diagnostic storage account. Press 'OK' when done.

The screenshot shows the 'Create virtual machine' wizard in the Azure portal, specifically the 'Settings' tab. The left sidebar shows the progression: 1 Basics (Done), 2 Size (Done), 3 Settings (Configure optional features), 4 Summary (Data Science Virtual Machine), and 5 Buy. The main content area is divided into sections: Storage, Network, Extensions, High availability, and Monitoring. The 'Storage' section shows a storage account '(new) rgnameyoursvmdisks992'. The 'Network' section shows a virtual network '(new) rg-NameYourDSVM-vnet', a subnet 'default (10.52.4.0/24)', a public IP address '(new) NameYourDSVM-ip', and a network security group '(new) NameYourDSVM-nsg'. The 'Extensions' section shows 'No extensions'. The 'High availability' section shows 'None'. The 'Monitoring' section shows 'Boot diagnostics' as 'Enabled' and 'Guest OS diagnostics' as 'Disabled' (highlighted with a red box). Below 'Guest OS diagnostics' is a 'Diagnostics storage account' '(new) rgnameyoursvmdiag608'. At the bottom, there is an 'OK' button (highlighted with a red box).

14. The Summary tab is mostly for review. Do verify the details that you have specified in the previous steps. This step also allows you to click on the link to “Download template and parameters”. This option allows you to procure the Azure Resource Manager (ARM) templates for your VM specification so you can deploy the same configuration programmatically without the use of the portal. Press ‘OK’ to continue to the ‘Buy’ tab once you have verified the information.

Microsoft Azure Data Science Virtual Machine > Create virtual machine > Summary

Search resources

Create virtual machine

Summary

Validation passed

Basics

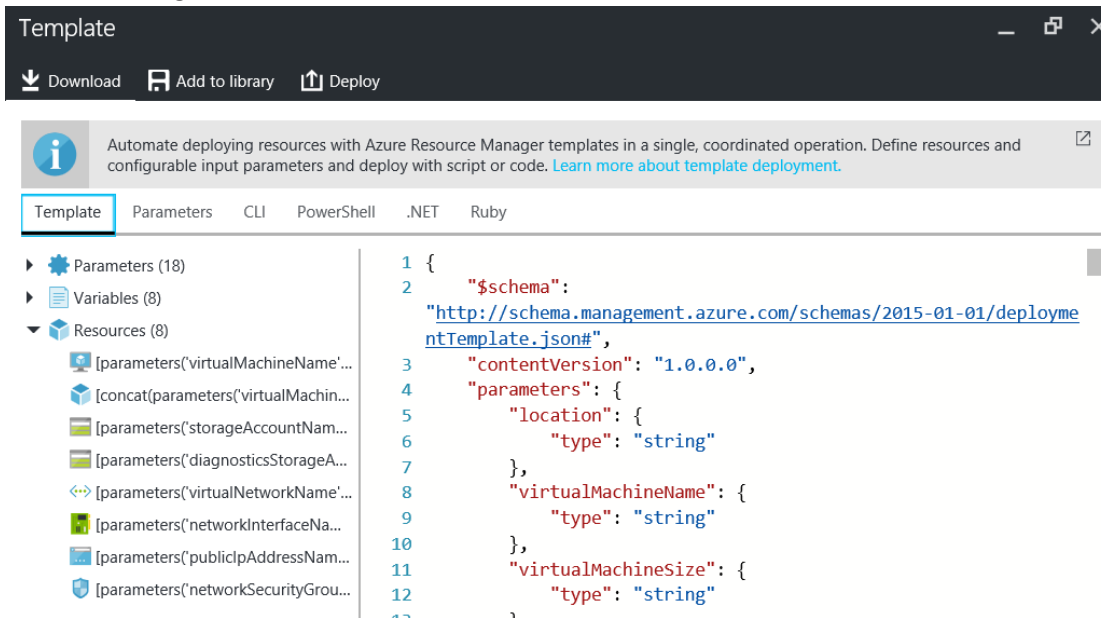
Subscription	Microsoft Azure Internal Consumption
Resource group	(new) rg-NameYourDSVM
Location	East US

Settings

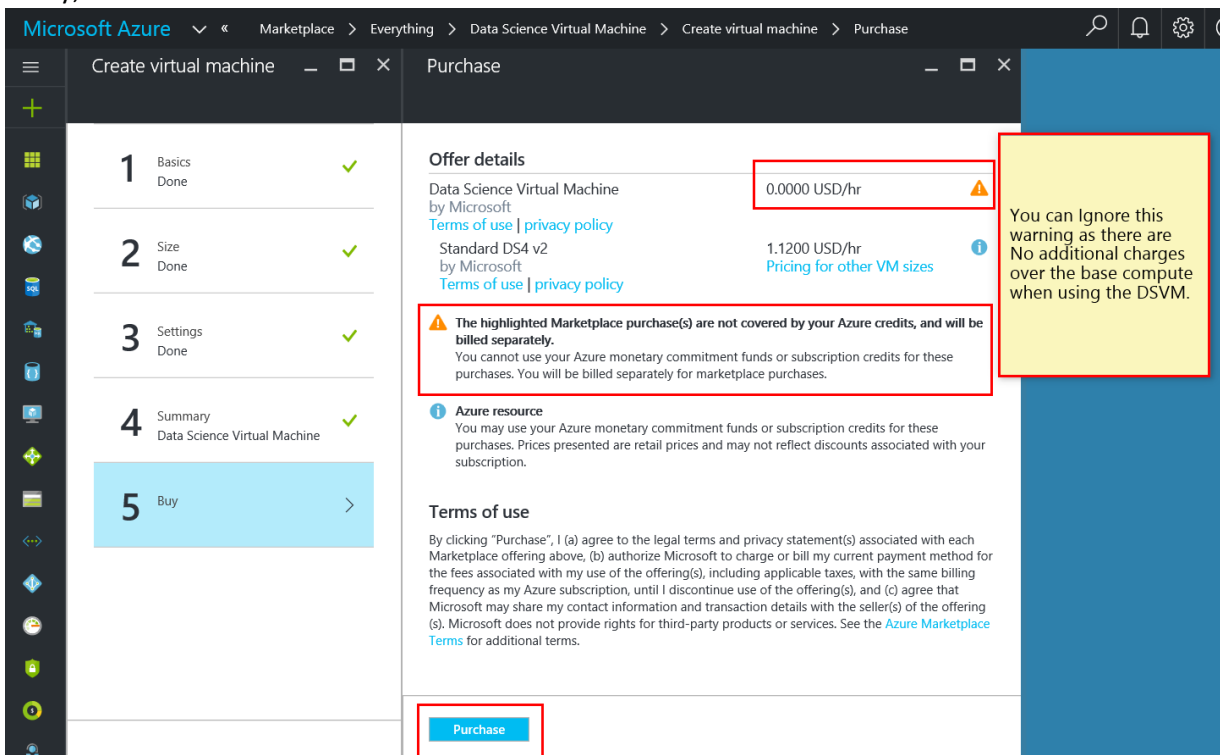
Computer name	NameYourDSVM
Disk type	SSD
User name	yourusername
Size	Standard DS4 v2
Storage account	(new) rgnameyourdsvm disks992
Virtual network	(new) rg-NameYourDSVM-vnet
Subnet	(new) default
Public IP address	(new) NameYourDSVM-ip
Network security group (firewall)	(new) NameYourDSVM-nsg
Availability set	None
Guest OS diagnostics	Enabled
Boot diagnostics	Enabled
Diagnostics storage account	(new) rgnameyourdsvm diag608

OK Download template and parameters

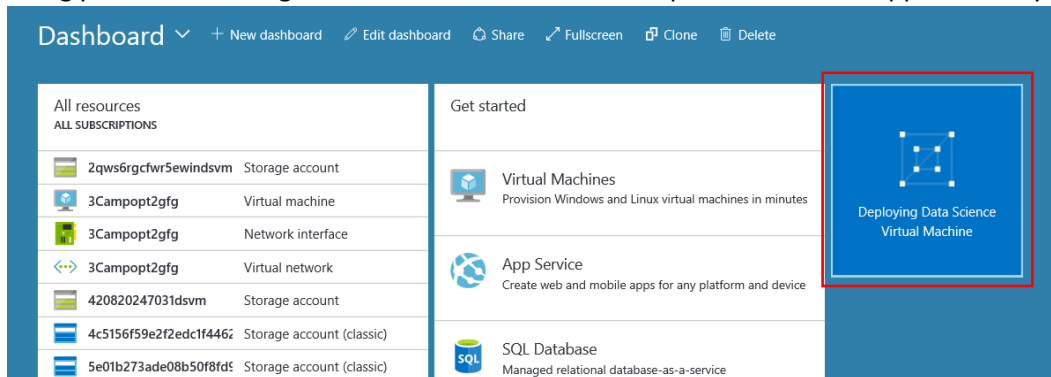
16. (OPTIONAL) If you do click on “Download Template and parameters”, the download interface will look like the figure below.



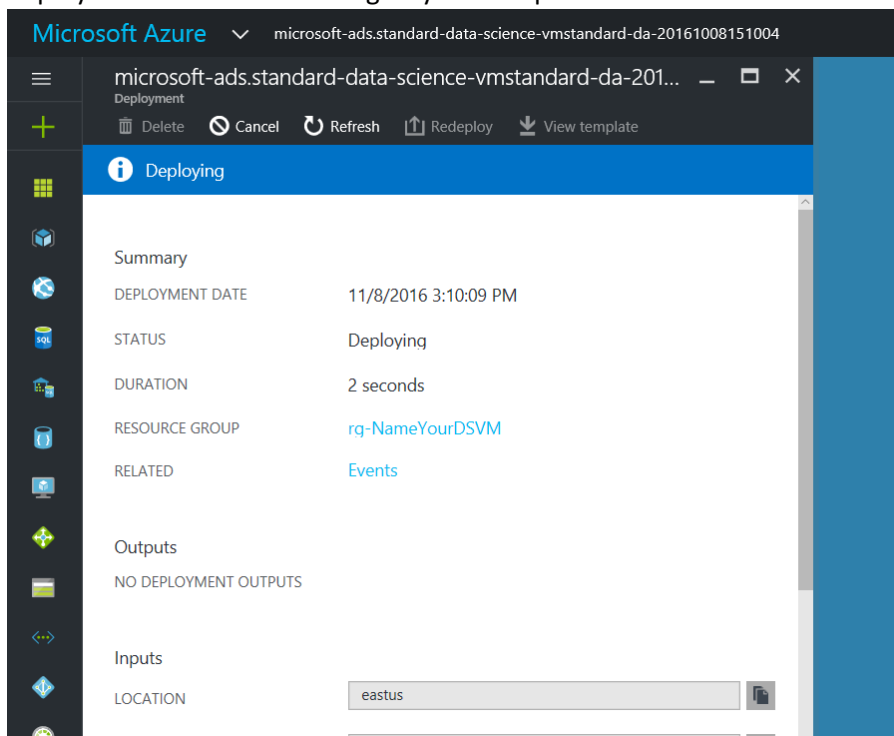
17. Once on the ‘Buy’ tab, you will see similar to the figure below. You will also see a ‘Marketplace Purchase’ charge warning. You can ignore this as this is a Zero-dollar warning and your actual per minute charges are only for the base compute as reflected in the compute charge line item. When ready, click ‘Purchase’.



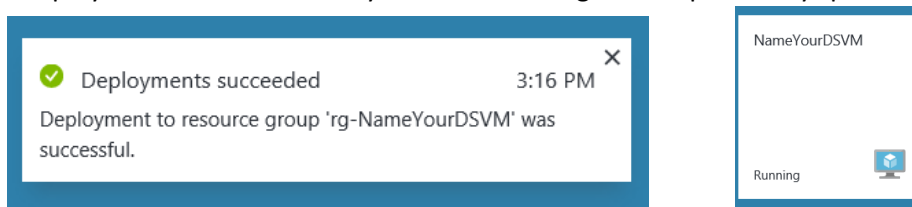
18. Once you click 'Purchase', you will be taken back to the portal dashboard and a new tile will show up like the figure below. This will read, "Deploying Data Science Virtual Machine". Your DSVM is now being provisioned along with its related resources. This process will take approximately 5 minutes.



19. If you click on the "Deploying Data Science Virtual Machine" tile, you will see some of the deployment details and it will give you the options to 'Cancel' or 'Refresh'.

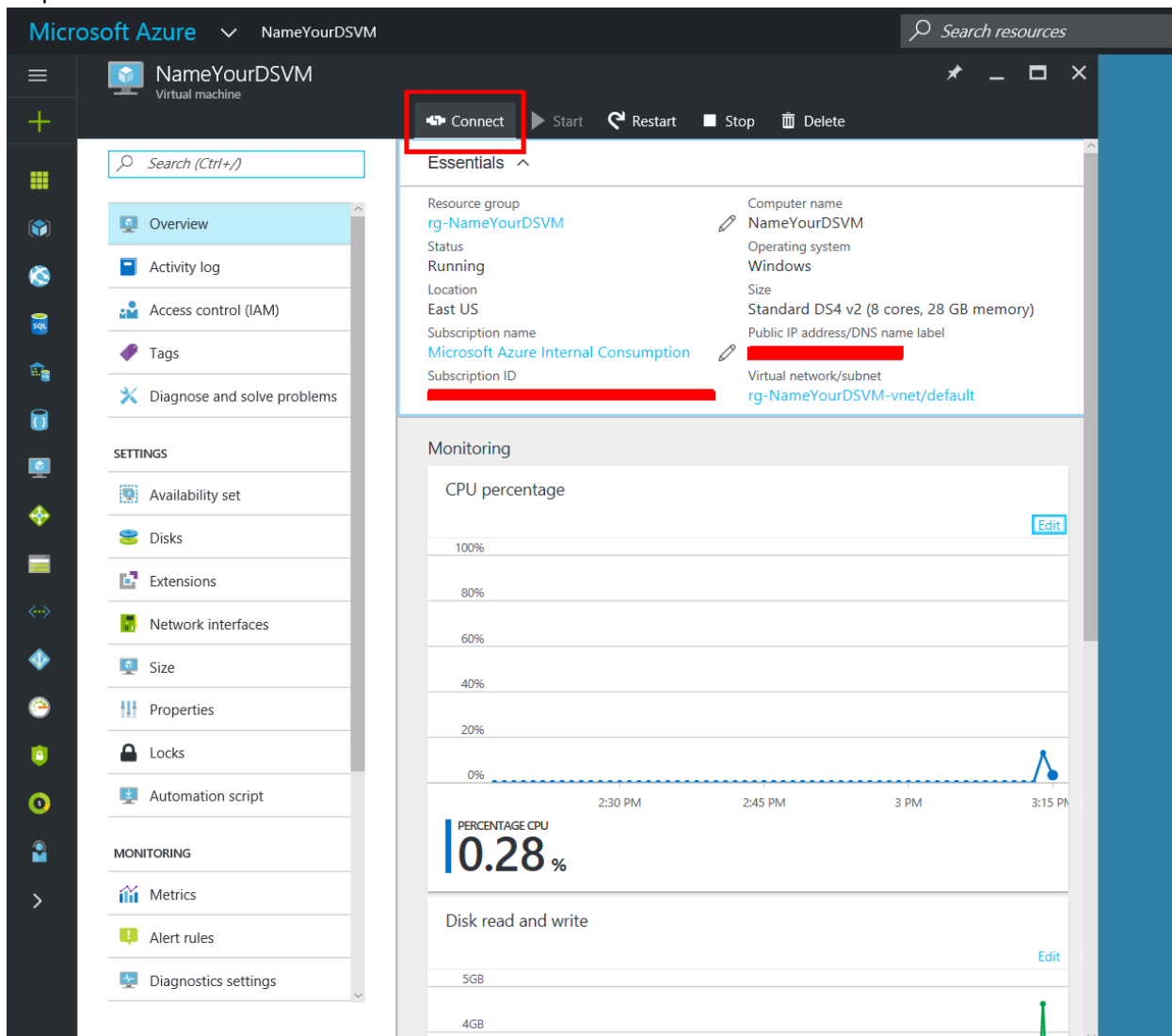


20. Once complete (after about 5 minutes) you will see a notification on the top right that reads "Deployment Successful" and your tile will change to the previously specified name for your DSVM.



Connecting to a Data Science Virtual Machine – Using Remote Desktop & Exploring the VM and some of the installed tools

21. You can now Click on the Tile for your DSVM. This will open the frame for your VM and give you numerous options, links and charts with dashboards showing the per minute diagnostics etc. if you had turned them on. You can now Click on the “Connect” button to download the RDP (remote desktop) file which is pre-configured with your VM’s details. You can also use this frame to restart, stop or delete the VM.

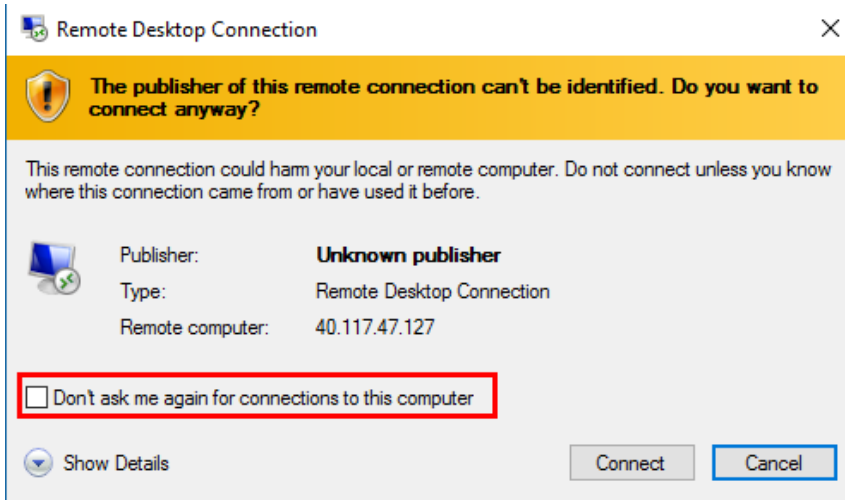


Save the RDP file to a location of your choice on your Local Machine.

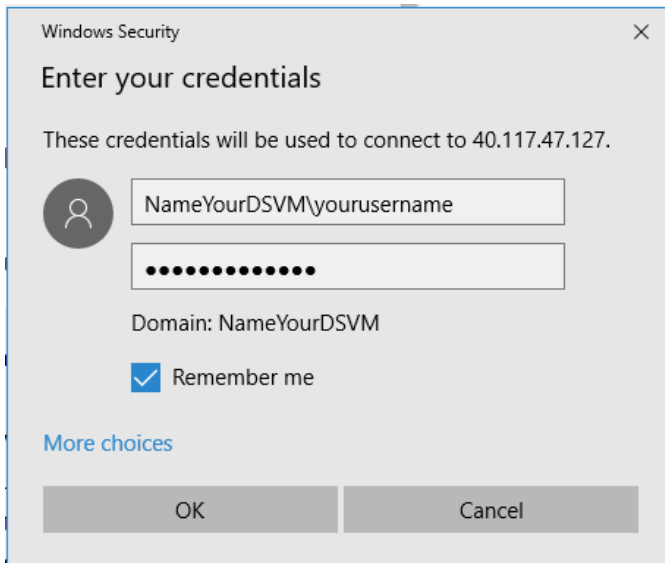


Alternatively you can also connect to the VM using the public IP on the Windows Remote Desktop Connection Client.

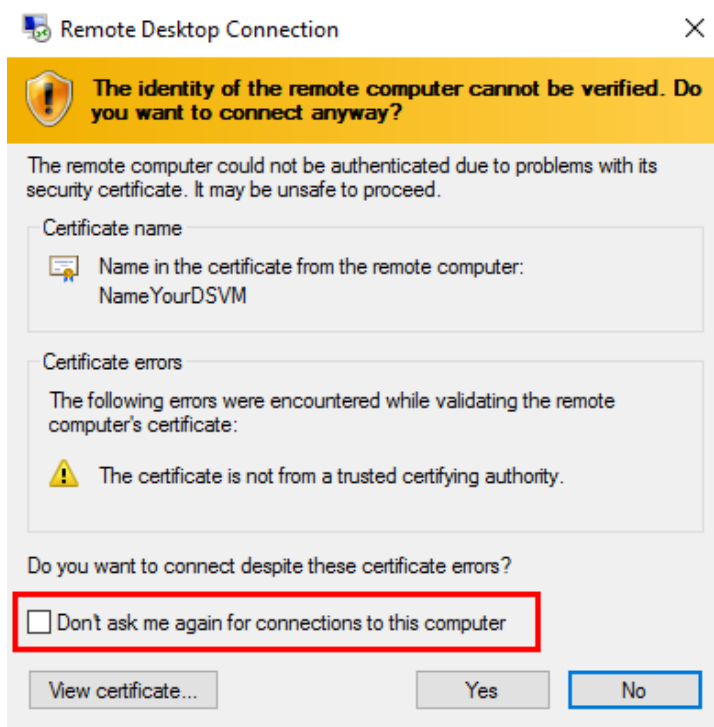
22. The first time you connect using this RDP connection to your created VM, you will see a publisher warning. Please ignore this and tick the 'Don't ask me again for connections to this computer' tick box to stop seeing this.



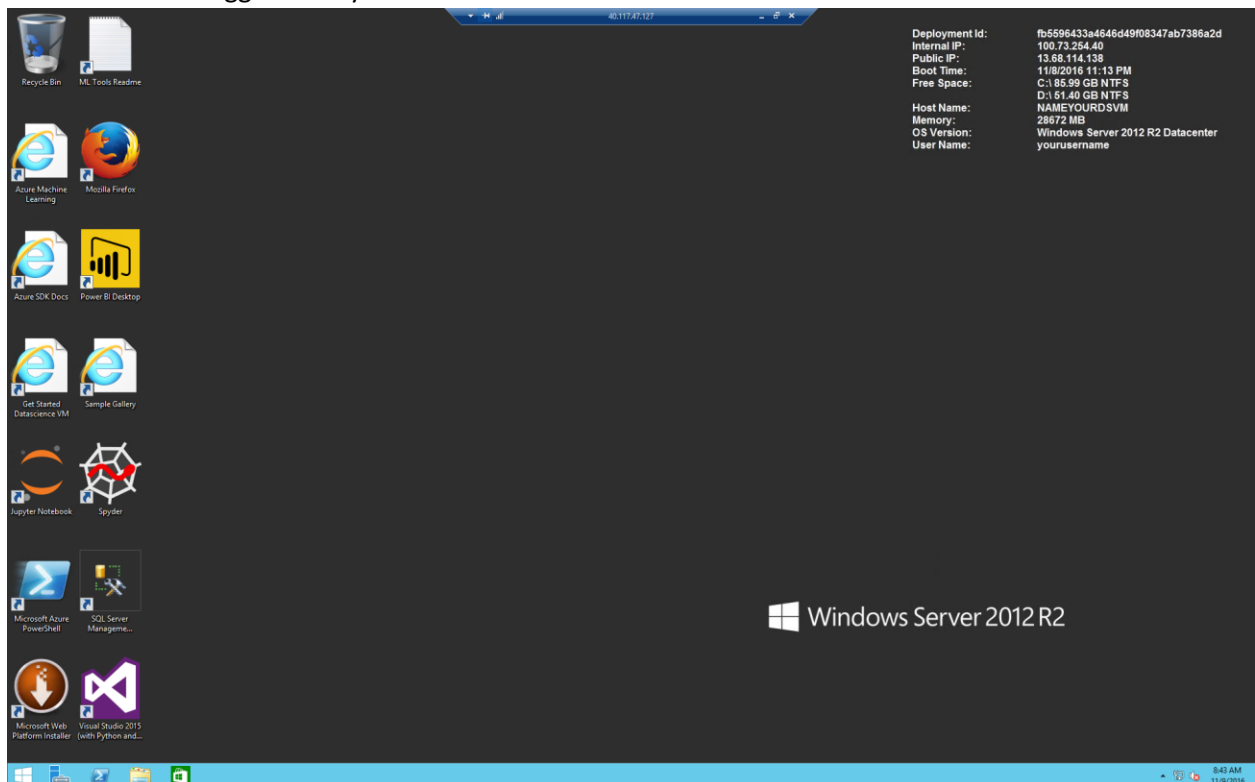
23. Enter your credentials.



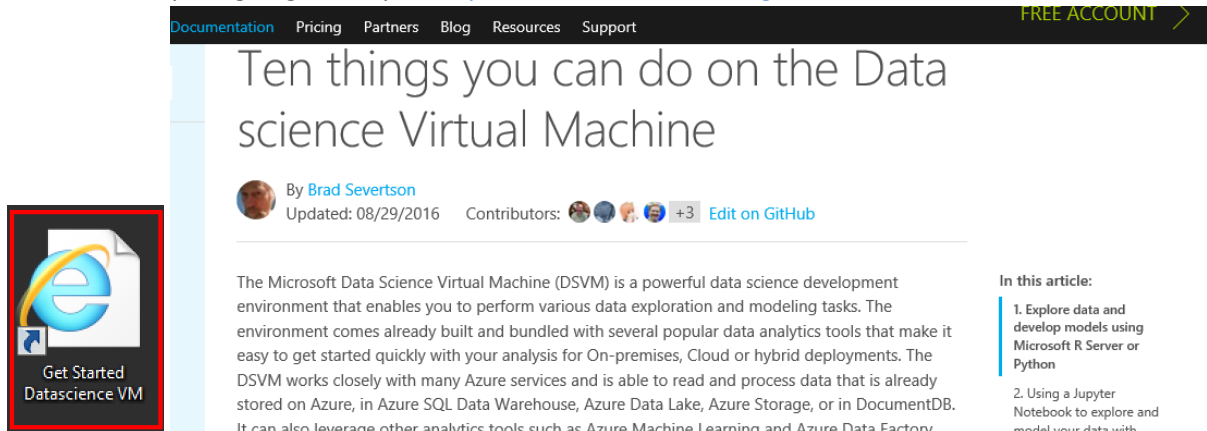
24. You will get another warning is because the VM is not in a domain yet where there is a trusted certificate provider. Please ignore this and tick the 'Don't ask me again for connections to this computer' tick box to stop seeing this.



25. You will now be logged in to your DSVM:




26. Check out the “Getting Started - 10 things You can do on DSVM” article by opening the shortcut link on the desktop, or going directly to <http://aka.ms/dsvmtenthings>.



Documentation Pricing Partners Blog Resources Support FREE ACCOUNT >

Ten things you can do on the Data science Virtual Machine

By Brad Severtson
Updated: 08/29/2016 Contributors:  +3 [Edit on GitHub](#)

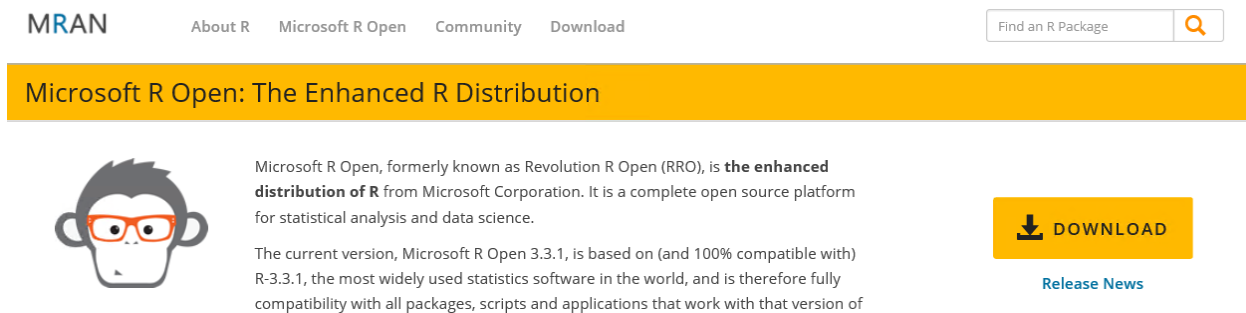
The Microsoft Data Science Virtual Machine (DSVM) is a powerful data science development environment that enables you to perform various data exploration and modeling tasks. The environment comes already built and bundled with several popular data analytics tools that make it easy to get started quickly with your analysis for On-premises, Cloud or hybrid deployments. The DSVM works closely with many Azure services and is able to read and process data that is already stored on Azure, in Azure SQL Data Warehouse, Azure Data Lake, Azure Storage, or in DocumentDB. It can also leverage other analytics tools such as Azure Machine Learning and Azure Data Factory.


Get Started Datascience VM

In this article:


1. Explore data and develop models using Microsoft R Server or Python
2. Using a Jupyter Notebook to explore and model your data with

27. Update your underlying Microsoft R Open environment if you wish from: <https://mran.microsoft.com/open>




MРАН About R Microsoft R Open Community Download 

Microsoft R Open: The Enhanced R Distribution



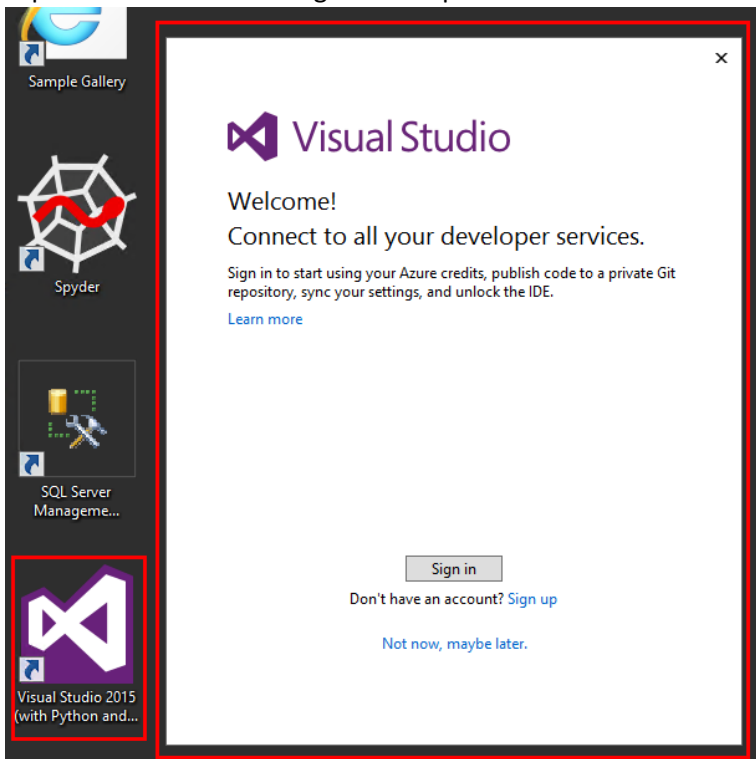
Microsoft R Open, formerly known as Revolution R Open (RRO), is **the enhanced distribution of R** from Microsoft Corporation. It is a complete open source platform for statistical analysis and data science.

The current version, Microsoft R Open 3.3.1, is based on (and 100% compatible with) R-3.3.1, the most widely used statistics software in the world, and is therefore fully compatibility with all packages, scripts and applications that work with that version of

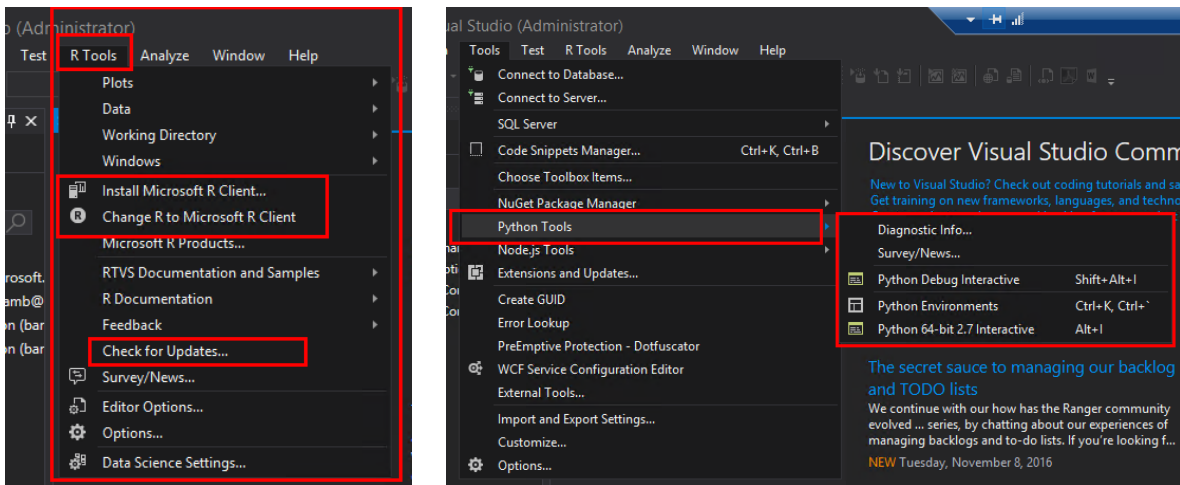
 **DOWNLOAD**

[Release News](#)

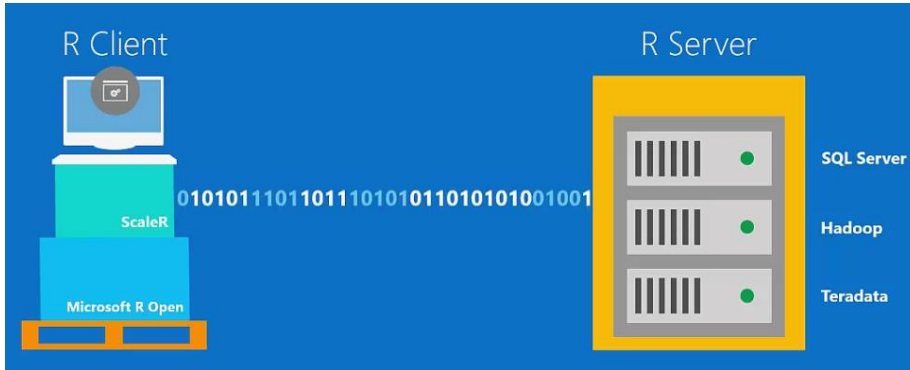
28. Open Visual Studio and sign in or skip.



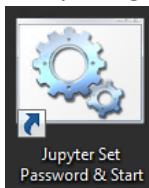
29. As you can see R & Python Tools for Visual Studio are pre-installed (RTVS & PTVS), you may check for updates. Please click Install Microsoft R Client and run through the installation to ensure you have the latest version.



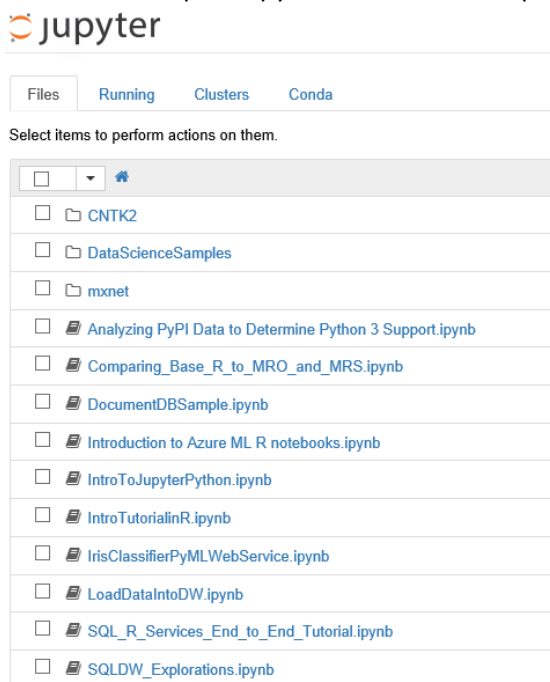
30. You may also install an IDE of choice like RStudio if you wish to use it. You may also wish to update the Version of Microsoft R client by going to this link: <http://aka.ms/rclient/download>. Some customers also prefer to use Microsoft R Server & R Services on SQL Server 2016. This introduction video for Microsoft R client should automatically start playing, but you will not get Audio from the VM so get the introductory video on your local machine from: <http://aka.ms/rclient/intro>.



31. The DSVM is also a fully configured Jupyter Notebooks Server (Disabled By default for security, you have to first set up your own password). Please make sure you set up jupyter security after first log-on by using the utility on the desktop.



You can then open Jupyter from the desktop or browse to “<https://localhost:9999/>”.

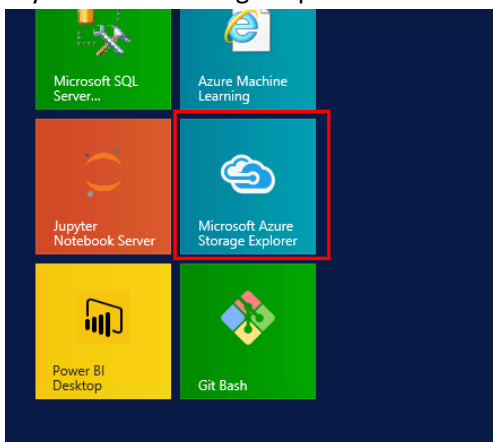


The DSVM comes Loaded with multiple examples.

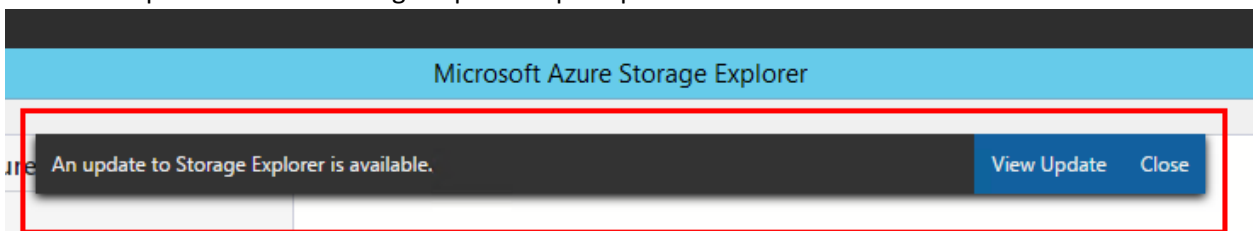
32. DSVN also has WEKA (Waikato Environment for Knowledge Analysis - popular data science tool) pre-installed. Please refer to <http://www.cs.waikato.ac.nz/ml/weka/documentation.html> for tutorial and resources.



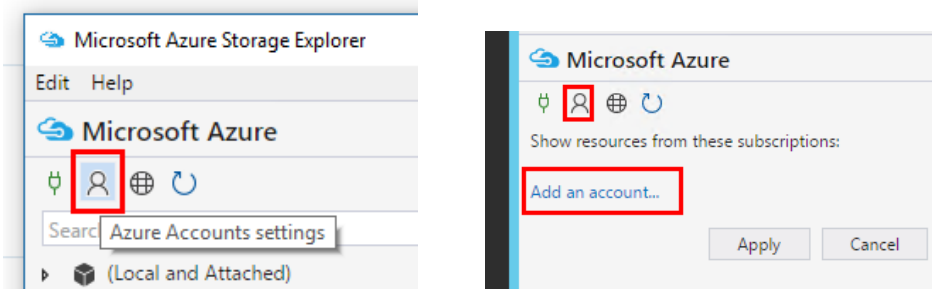
33. Try out Azure Storage Explorer:



Install the update to Azure Storage Explorer if prompted:



34. Add your Azure Account by signing in to see your subscription(s) and associated storage accounts:



Fill out your credentials:

Microsoft Azure

Work or school, or personal Microsoft account

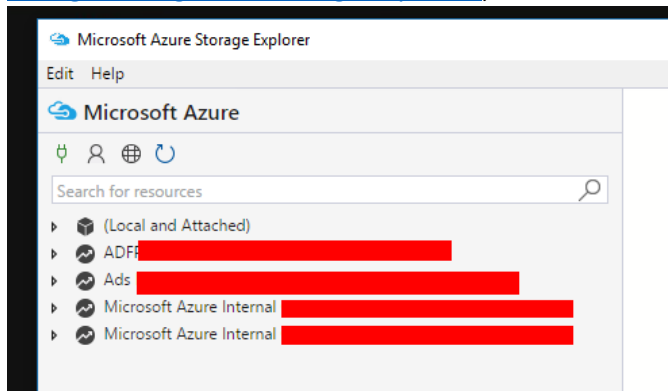
Email or phone

Password

Sign in Back

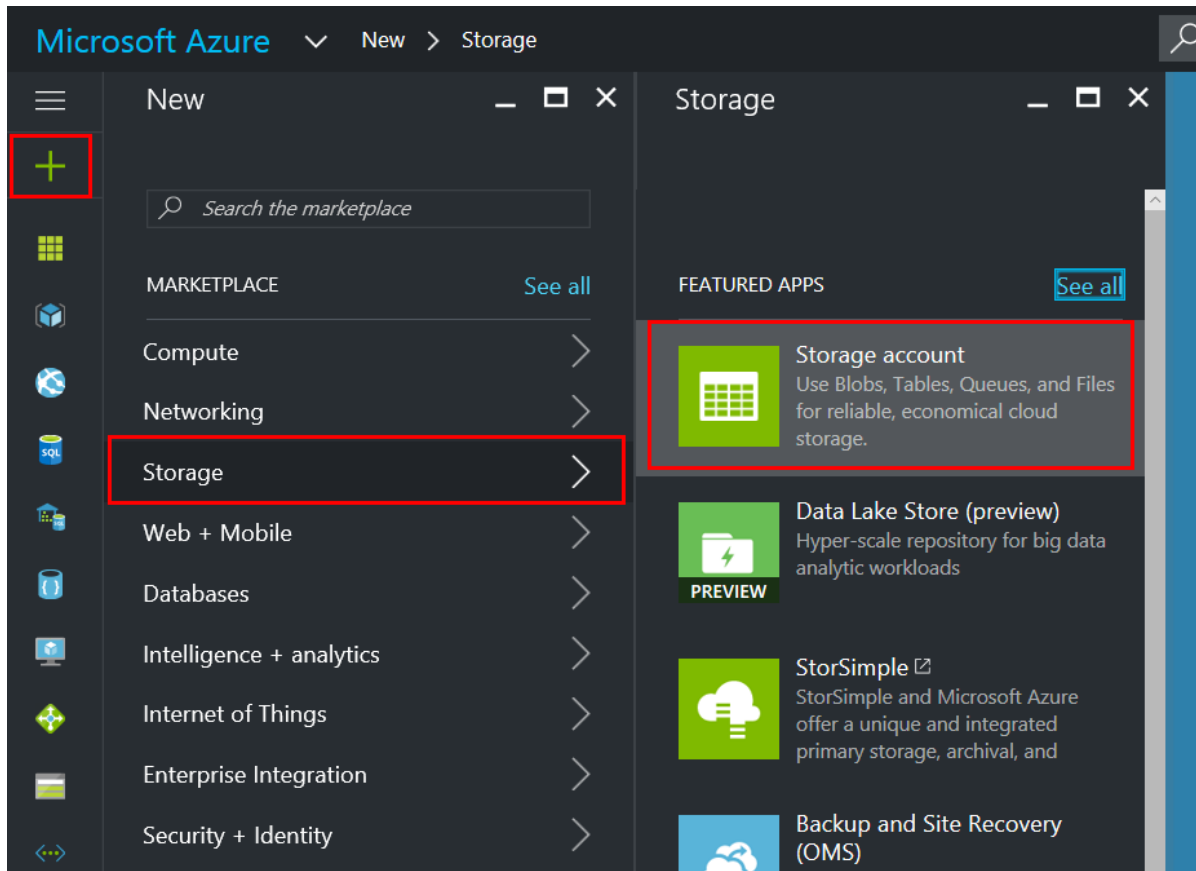
[Can't access your account?](#)

35. Once Signed in your view should look like the figure with the subscription(s) listed as dropdowns. When you expand the drop downs, you will see a list of all your storage accounts, which further expand to Blobs, Files, Queues and Tables. For a detailed introduction, refer to “Getting Started with Storage Explorer” (<https://azure.microsoft.com/en-us/documentation/articles/vs-azure-tools-storage-manage-with-storage-explorer/>).



Creating an Azure Storage Account and a 5TB Shared File store and Mounting the Shared Storage as a drive on the VM

36. Confirm that you have an existing storage account. If not, you can go and create one using the Azure Portal: <https://portal.azure.com>. On the portal navigate to New >> Storage >> Storage Account



37. On the Create Storage account pane, enter your desired name for your storage account and change the default values if you wish to do so. You can change settings like Performance, Encryption, Replication etc. You may choose the same resource group as your VM that you created earlier or may choose another group or create a new one. It is best to have your storage in the same Azure location as your Data Science VM to avoid latency and data movement charges. You may also choose to pin it to your dashboard. Click Create when ready.

Microsoft Azure New > Storage > Create storage account

Create storage account

The cost of your storage account depends on the usage and the options [Learn more](#)

* Name ⁱ
yourdsmvstorage ✓
.core.windows.net

Deployment model ⁱ
Resource manager Classic

Account kind ⁱ
General purpose
Blob storage

Performance ⁱ
Standard Premium

Replication ⁱ
Read-access geo-redundant storage (RA...) ▼

* Storage service encryption ⁱ
Disabled Enabled

* Subscription
Microsoft Azure Internal Consumption ▼

* Resource group ⁱ
☐ Create new ☒ Use existing
rg-NameYourDSVM ▼

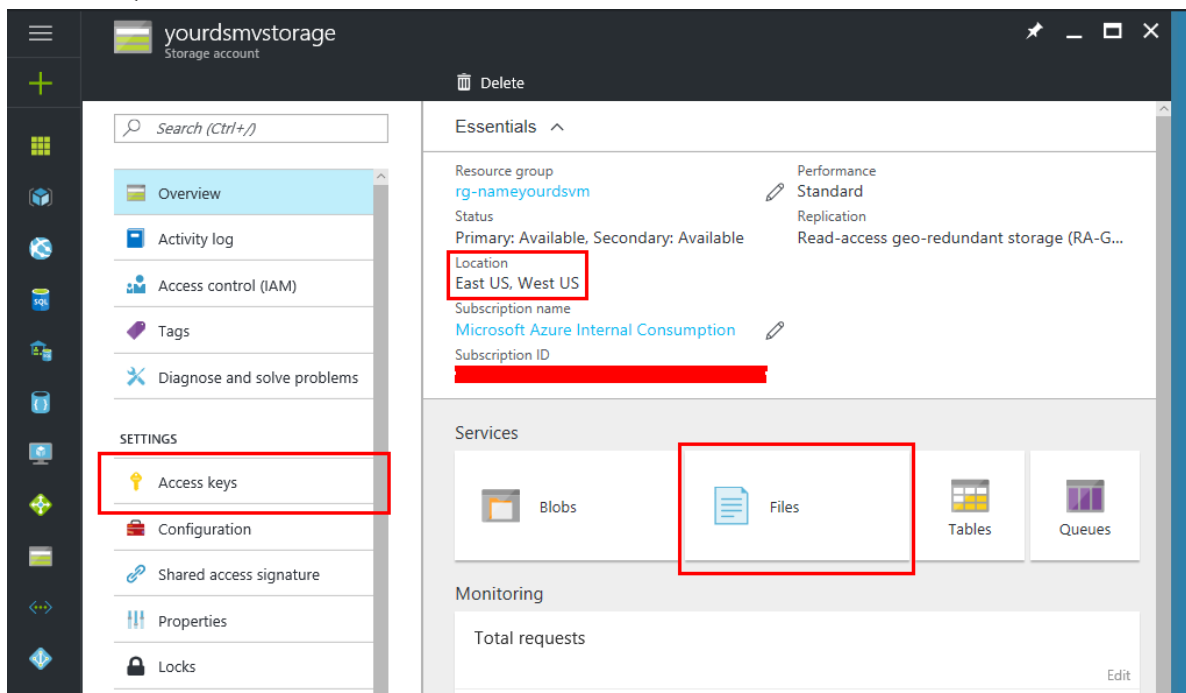
* Location
East US ▼

☒ Pin to dashboard

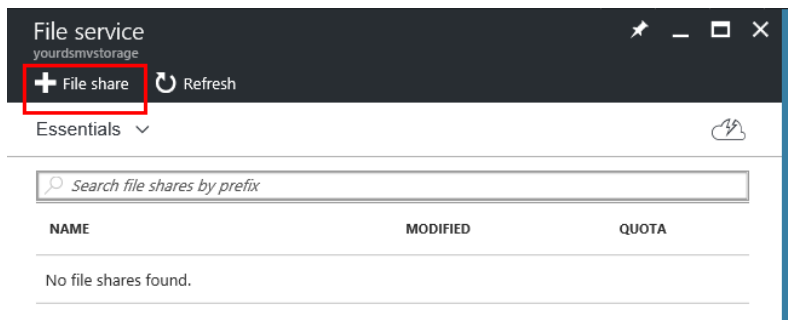
[Automation options](#)

[Create](#)

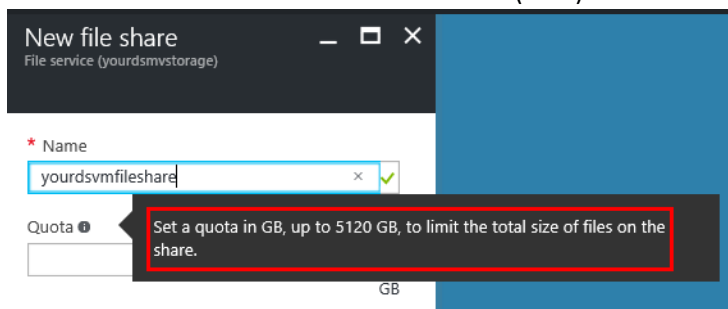
38. Once created you will see the storage account screen which should look like the figure below. If you select geo redundant storage, note that there will be a secondary location that will also be displayed. Since this is a general-purpose account you may create storage types of Blobs, Files, Tables & Queues.



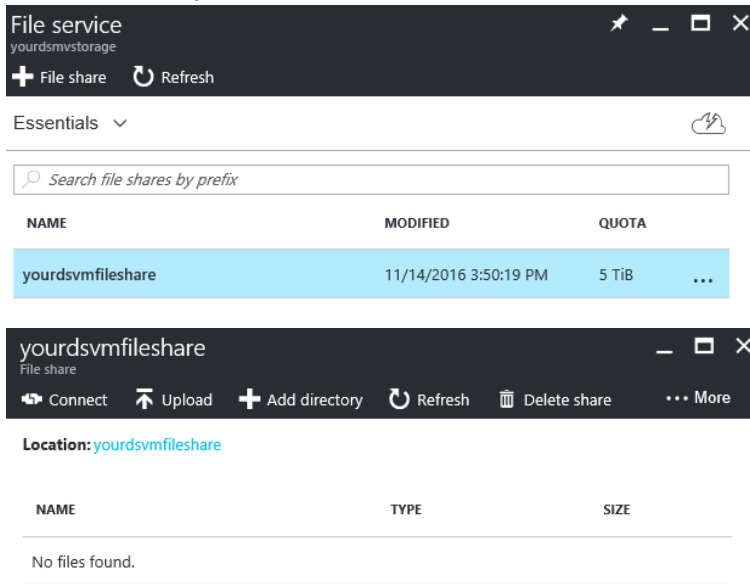
39. Click on 'Files' to show the File service pane and then click on '+ File share' to create a new File share.



Specify the 'Name' of your file share and optionally specify the maximum quota for size. By default, it chooses the maximum value of 5120 GB (5 TB). Press 'Create' to complete creation.



You should see your new file share.

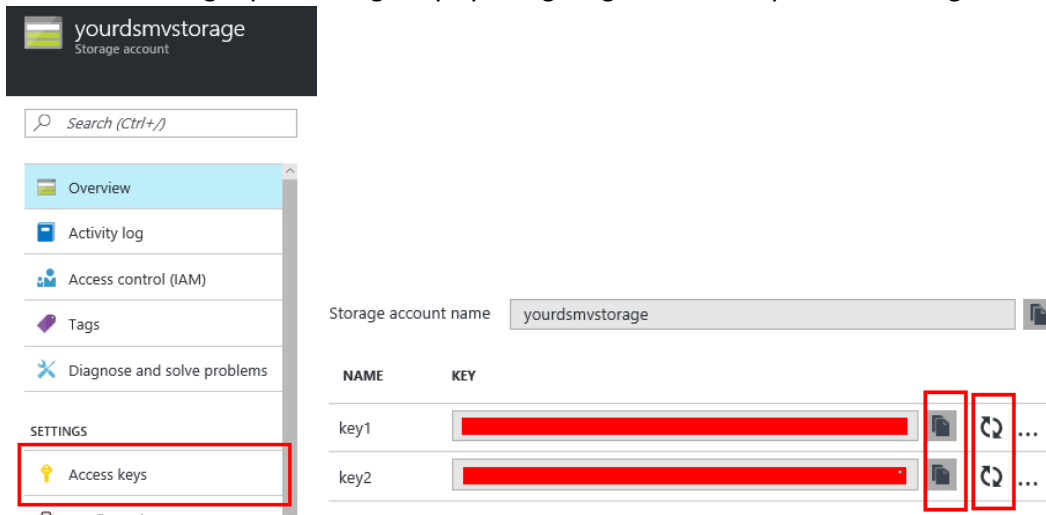


40. If you wish to create File Shares programmatically you can use the following PowerShell script on the pre-installed PowerShell executable.

```
# Authenticate to Azure.
Login-AzureRmAccount
# Select your subscription
Get-AzureRmSubscription -SubscriptionName "<your subscription name>" | Select-
AzureRmSubscription
# Create a new resource group.
New-AzureRmResourceGroup -Name <rg-yourdsvmdata>
# Create a new storage account. You can reuse existing storage account if you wish.
New-AzureRmStorageAccount -Name <mydatadisk> -ResourceGroupName <rg-yourdsvmdata> -
Location "<Azure Data Center Name For eg. East US>" -Type "Standard_LRS"
# Set your current working storage account
Set-AzureRmCurrentStorageAccount -ResourceGroupName "< rg-yourdsvmdata >" -
StorageAccountName <mydatadisk>

# Create a Azure File Service Share
$S = New-AzureStorageShare <<teamsharename>>
# Create a directory under the File share. You can give it any name
New-AzureStorageDirectory -Share $S -Path <directory name>
# List the share to confirm that everything worked
Get-AzureStorageFile -Share $S
```

41. This 5TB File share can be easily mounted as a shared drive on the Data Science Virtual Machine. This is a very convenient tool for sharing moderately sized data assets across team members as they can be mounted to multiple machines at the same time. Additionally, you can also create multiple File shares and mount them simultaneously on the same machine. As a best practice guidance, we recommend starting from 'Z' and then coming backwards when mounting File Shares with Drive Letters. You can get your storage key by Navigating to Access Keys on the Storage account pane:



Use the copy buttons to copy one of the keys or the refresh button to generate new keys. Here are the commands to mount the drive on the DSVM that you can run on Azure PowerShell.

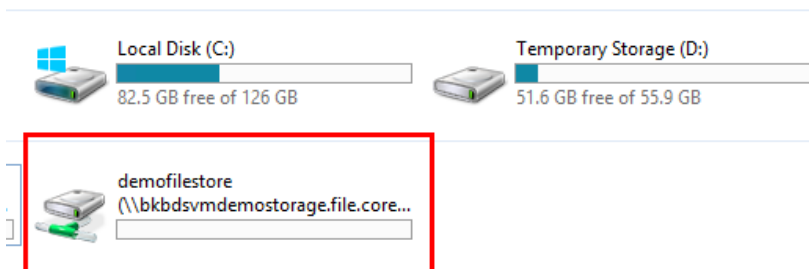
Get storage key of the storage account that has the Azure file share from Azure portal. Store it securely on the VM to avoid prompted in next command.

cmdkey /add:<<mydatadisk>>.file.core.windows.net /user:<<mydatadisk>> /pass:<storage key>

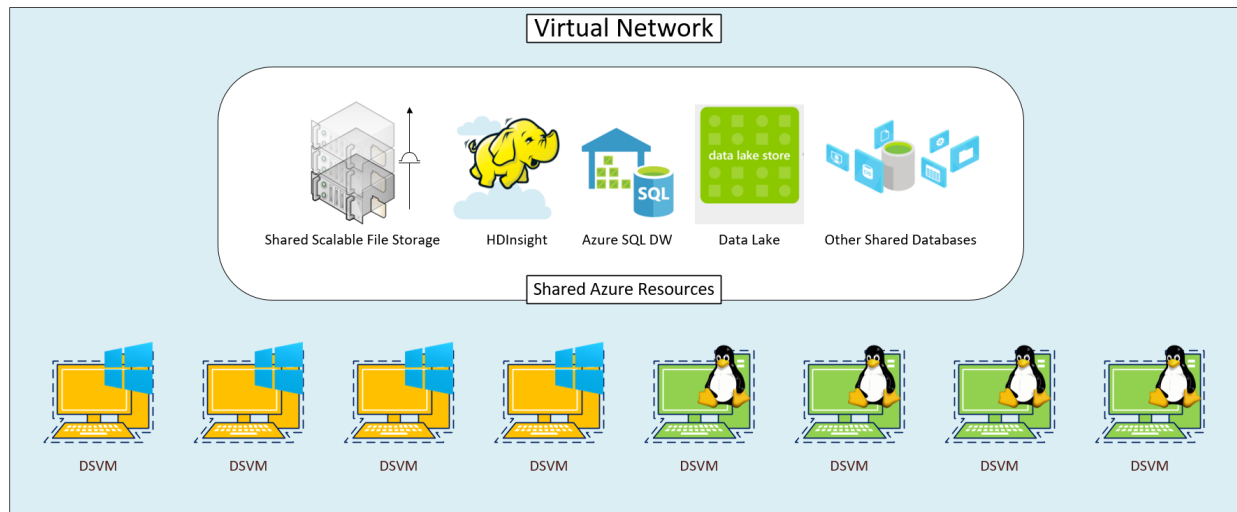
Mount the Azure file share as Z: drive on the VM. You can chose another drive letter if you wish
net use z: \\<mydatadisk>.file.core.windows.net\\<<teamsharename>>

```
PS C:\> cmdkey /add:bkbdsvmdestorage.file.core.windows.net /user:bkbdsvmdestorage /pass:
CMDKEY: Credential added successfully.
PS C:\> net use x: \\bkbdsvmdestorage.file.core.windows.net\demofilestore
The command completed successfully.
```

The Mounted Drive should now show in your explorer:



42. The Shared resource model is key to success and collaboration:



43. To copy files back and forth from Azure Blobs you may also use the pre-installed AzCopy utility:

Copy *.sql from local machine to a Azure Blob

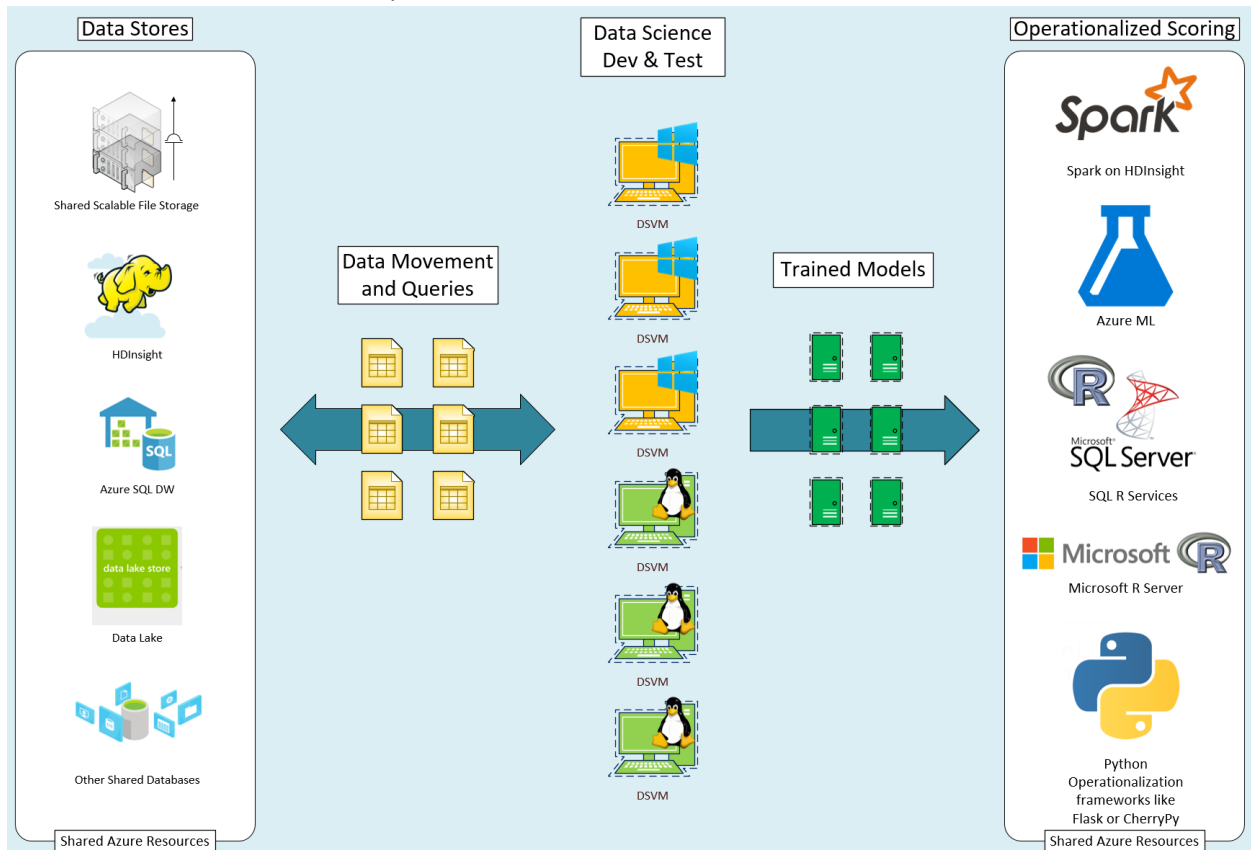
```
"C:\Program Files (x86)\Microsoft SDKs\Azure\AzCopy\azcopy" /Source:"c:\Aaqs\Data Science Scripts" /Dest:https://[ENTER STORAGE ACCOUNT].blob.core.windows.net/[ENTER CONTAINER] /DestKey:[ENTER STORAGE KEY] /S /Pattern:*.sql
```

Copy back all files from Azure Blob container to Local machine

```
"C:\Program Files (x86)\Microsoft SDKs\Azure\AzCopy\azcopy" /Dest:"c:\Aaqs\Data Science Scripts\temp" /Source:https://[ENTER STORAGE ACCOUNT].blob.core.windows.net/[ENTER CONTAINER] /SourceKey:[ENTER STORAGE KEY] /S
```

Typical Data Science Walkthrough using the DSVM – From Getting Data and Creating Models to publishing Trained models as web services and consuming them for scoring – (R, Jupyter Notebooks, RTVS, Azure ML)

44. The DSVM Dev Test to Model Operationalization Workflow Looks like:

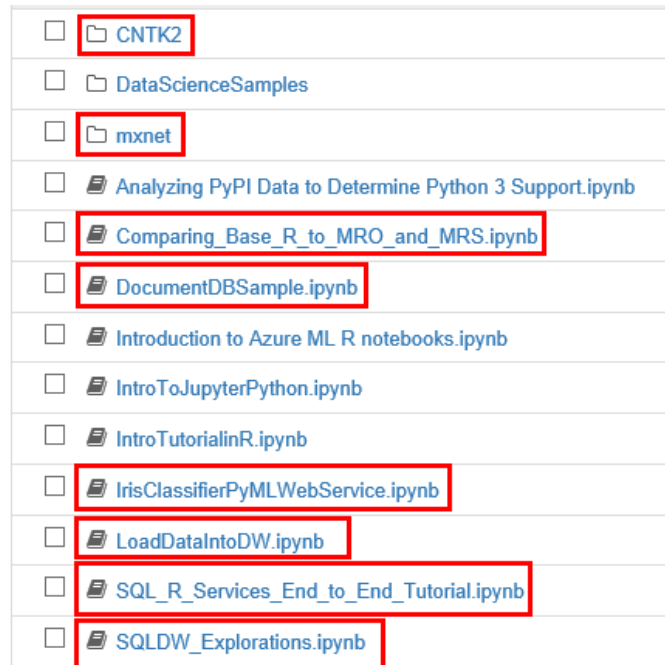


45. To better understand this workflow, we can run through one of the examples already provided on the DSVM's Jupyter Notebook Server. The same can be done in R Tools for Visual Studio by copying the code over.

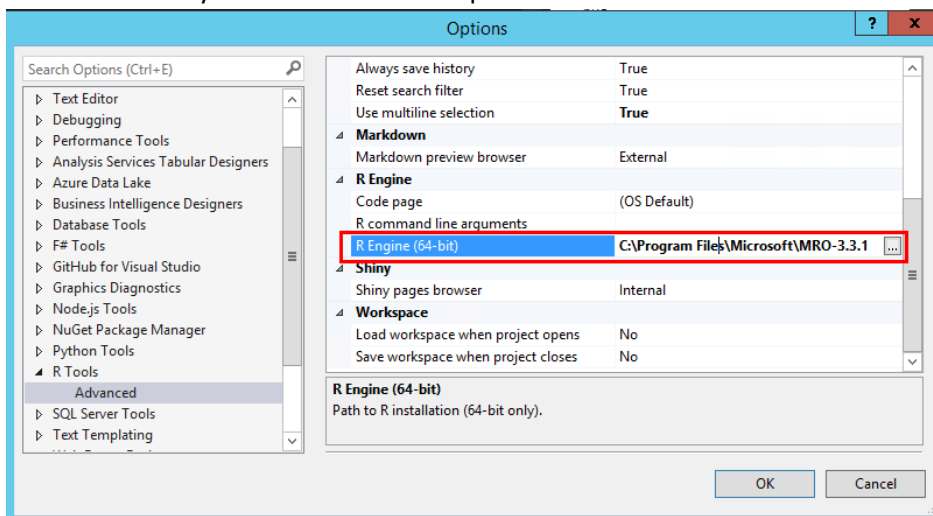
We will go through the "Introduction to Azure ML R notebooks.ipynb" notebook.

- ☐ [Analyzing PyPI Data to Determine Python 3 Support.ipynb](#)
- ☐ [Comparing_Base_R_to_MRO_and_MRS.ipynb](#)
- ☐ [DocumentDBSample.ipynb](#)
- ☒ [Introduction to Azure ML R notebooks.ipynb](#)
- ☐ [IntroToJupyterPython.ipynb](#)
- ☐ [IntroTutorialinR.ipynb](#)

46. There Are Multiple Notebooks that show how to connect to several Azure and other Data Stores Like Document DB, SQL Server, Azure SQL Data Warehouse etc. Additionally, there are examples for doing Deep Learning tasks using the Deep Neural Network and Cognitive Frameworks like mxnet and CNTK2 (Also known as the Microsoft Cognitive Toolkit). These are already pre-installed on the DSVM.



47. Open the “Introduction to Azure ML R notebooks.ipynb” notebook. This Notebook fits a model using R, then publishes the model as a web service to Azure ML Studio. If you are unfamiliar with Azure ML, Section 2 of the Data Scientists' Guide provides enough information for you to follow this tutorial: <https://gallery.cortanaintelligence.com/Experiment/Tutorial-for-Data-Scientists-3>.
48. (Optional) If you do the example from R Tools for Visual Studio, make sure to specify the R environment of your choice from the options window for R Tools.



49. In this example, you use the Boston housing data from the R package MASS. The Boston dataset contains 506 rows and 14 columns. Available information includes median home price, average number of rooms per dwelling, crime rate by town, etc.

More information about this dataset can be found by typing `?Boston` or `help(Boston)` in an R terminal, or at this UCI page <https://archive.ics.uci.edu/ml/datasets/Housing>.

To run a cell, select it and press CTRL+ENTER or SHIFT+ENTER.

The window should look like the figure below. Now you can proceed to running each Cell by placing the cursor in the cell and the Pressing CTRL+ENTER or SHIFT+ENTER and following the instructions.

Introduction to Azure ML R notebooks

Summary

Introduce situations where Azure Machine Learning (ML) R notebooks can be used. Fit a model using R, then publish the model as a web service to Azure ML Studio.

Description

The purpose of this notebook is to demonstrate how to use Jupyter notebooks on the Azure Machine Learning Studio to develop a model in R and publish a web service based on the model.

Using Azure ML notebooks

For data scientists new to Azure ML and accustomed to doing all analytical work using R on local computers, Azure Machine Learning makes it possible to write R notebooks on the cloud. So anyone with internet access can work with R from a web browser.

If you use R and understand the basics of Azure ML, notebooks make it possible to develop your models in R and then operationalize them easily.

Data scientists who are comfortable with both R and Azure ML Experiments, can use these together in different ways:

- To explore data from an Azure ML Experiment. For example, you can use a notebook to visualize your data in different ways.
- To fit models and use techniques that are not available in Azure ML Experiments yet. For example, R offers more options in terms of variable selection techniques and a wider variety of models. You can also use it for time series analysis.
- To test code before using it in the "Execute R Script" module of Azure ML Experiments.

Target audience

The target audience of this notebook are R users who have a basic understanding of Azure ML. If you are new to Azure ML, Section 2 of the [Data Scientists' Guide](#) provides enough information for you to follow this tutorial.

Data

In this example, you use the Boston housing data from the R package MASS. The Boston dataset contains 506 rows and 14 columns. Available information includes median home price, average number of rooms per dwelling, crime rate by town, etc. More information about this dataset can be found by typing `?Boston` or `help(Boston)` in an R terminal, or at this [UCI page](#).

To run a cell, select it and press CTRL+ENTER or SHIFT+ENTER.

```
In [5]: # load the library to use the Boston dataset
library(MASS)
?Boston
```

Out[5]: Boston {MASS} [R Documentation](#)

Housing Values in Suburbs of Boston

50. The code in the first cell is to get an output that describes the Dataset which is a data frame with 506 rows and 14 columns. Click and make sure your cursor is in the cell and then press SHIFT + ENTER if you want to re run the code.

```
In [1]: # load the library to use the Boston dataset
library(MASS)
?Boston
```

Even though for the sake of this example we are just using data from the MASS Library, this data can be from any of the typical data sources in our organization like Hadoop or SQL or Data Warehouse or just files on disk.

The Output on Jupyter should look like:

Out[1]:

Boston {MASS}

R Documentation

Housing Values in Suburbs of Boston

Description

The `Boston` data frame has 506 rows and 14 columns.

Usage

```
Boston
```

Format

This data frame contains the following columns:

- `crim`
per capita crime rate by town.
- `zn`
proportion of residential land zoned for lots over 25,000 sq.ft.
- `indus`
proportion of non-retail business acres per town.
- `chas`
Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).
- `nox`
nitrogen oxides concentration (parts per 10 million).
- `rm`
average number of rooms per dwelling.
- `age`
proportion of owner-occupied units built prior to 1940.
- `dis`
weighted mean of distances to five Boston employment centres.
- `rad`
index of accessibility to radial highways.
- `tax`
full-value property-tax rate per \$10,000.
- `ptratio`
pupil-teacher ratio by town.
- `black`
 $1000(Bk - 0.63)^2$ where Bk is the proportion of blacks by town.
- `lstat`
lower status of the population (percent).
- `medv`
median value of owner-occupied homes in \$1000s.

Source

Harrison, D. and Rubinfeld, D.L. (1978) Hedonic prices and the demand for clean air. *J. Environ. Economics and Management* 5, 81–102.

Belsey D.A., Kuh, E. and Welsch, R.E. (1980) *Regression Diagnostics. Identifying Influential Data and Sources of Collinearity*. New York: Wiley.

[Package MASS version 7.3-43]

On R tools for Visual Studio, use the R Interactive window to get an output like:

The screenshot shows the R Interactive window in Visual Studio. The top pane displays the R console output, which includes the R startup message, the loading of the MASS package, and the execution of the `?Boston` command. The bottom pane shows the R Help window for the `Boston` dataset, titled "Housing Values in Suburbs of Boston".

R Console Output:

```
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Microsoft R Open 3.3.1
The enhanced R distribution from Microsoft
Microsoft packages Copyright (C) 2016 Microsoft Corporation

Using the Intel MKL for parallel mathematical computing(using 4 cores).

Default CRAN mirror snapshot taken on 2016-07-01. See: https://mran.microsoft.com/.
> # load the library to use the Boston dataset
+ library(MASS)
+ ? Boston
starting httpd help server ... done
>
```

R Help Window: Boston [MASS]

Housing Values in Suburbs of Boston

Description

The Boston data frame has 506 rows and 14 columns.

Usage

```
Boston
```

Format

This data frame contains the following columns:

- crim**
per capita crime rate by town.
- zn**
proportion of residential land zoned for lots over 25,000 sq.ft.
- indus**
proportion of non-retail business acres per town.
- chas**
Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).
- nox**
nitrogen oxides concentration (parts per 10 million).
- rm**
average number of rooms per dwelling.
- age**
proportion of owner-occupied units built prior to 1940.
- dis**
weighted mean of distances to five Boston employment centres.
- rad**
index of accessibility to radial highways.
- tax**
full-value property-tax rate per \$10,000.
- ptratio**
pupil-teacher ratio by town.
- black**
 $1000(Bk - 0.63)^2$ where Bk is the proportion of blacks by town.
- lstat**
lower status of the population (percent).
- medv**
median value of owner-occupied homes in \$1000s.

Source

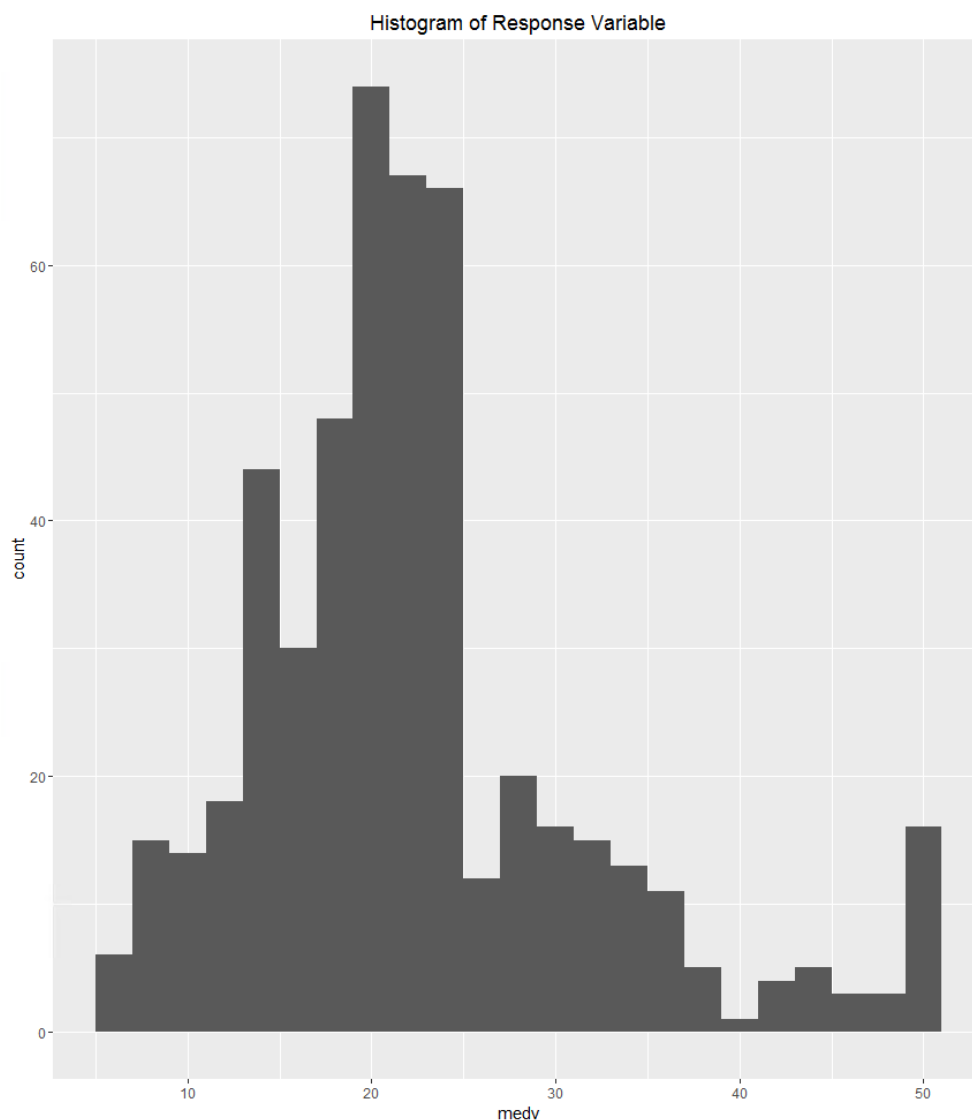
Harrison, D. and Rubinfeld, D.L. (1978) Hedonic prices and the demand for clean air. *J. Environ. Economics and Management* 5, 81–102.
Belsley D.A., Kuh, E. and Welsch, R.E. (1980) *Regression Diagnostics. Identifying Influential Data and Sources of Collinearity*. New York: Wiley.

[Package MASS version 7.3-45 [index](#)]

51. In the previous step, we got a good verbal description of the data. For the purposes of illustration, we will be using the 'medv' variable (Median Value of Owner-Occupied homes in \$1000s) as the target/response variable and the other 13 as the predictors. So, let us first see what the distribution of our response variable looks like. We will use the 'ggplot2' library for this:

```
In [6]: # library for plotting
library(ggplot2)
# plot distribution of the response variable
ggplot(Boston, aes(x=medv)) +
  geom_histogram(binwidth=2) +
  ggtitle("Histogram of Response Variable")
```

Run the next cell on your notebook or Run these lines of code in RTVS. You should see a plot resembling the figure that follows:



52. Now let us fit a simple Linear Regression model to predict 'medv' using all the other 13 variables. You may use other models if you feel like, this is an illustration only to showcase the functionality and the typical workflow.

```
In [7]: # fit a model using all variables except medv as predictors
lm1 <- lm(medv ~ ., data = Boston)

# check model performance
summary(lm1)
```

Your Output should look like either of the 2 following figures depending on whether you are running Jupyter or RTVS:

```
Out[7]:
Call:
lm(formula = medv ~ ., data = Boston)

Residuals:
    Min       1Q   Median       3Q      Max
-15.595  -2.730  -0.518   1.777  26.199

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
zn          4.642e-02  1.373e-02   3.382 0.000778 ***
indus       2.056e-02  6.150e-02   0.334 0.738288
chas       2.687e+00  8.616e-01   3.118 0.001925 **
nox       -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
rm         3.810e+00  4.179e-01   9.116 < 2e-16 ***
age         6.922e-04  1.321e-02   0.052 0.958229
dis       -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
rad         3.060e-01  6.635e-02   4.613 5.07e-06 ***
tax       -1.233e-02  3.760e-03  -3.280 0.001112 **
ptratio    -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
black       9.312e-03  2.686e-03   3.467 0.000573 ***
lstat     -5.248e-01  5.072e-02  -10.347 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.745 on 492 degrees of freedom
Multiple R-squared:  0.7406,    Adjusted R-squared:  0.7338
F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

```
Call:
lm(formula = medv ~ ., data = Boston)

Residuals:
    Min       1Q   Median       3Q      Max
-15.595  -2.730  -0.518   1.777  26.199

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
zn          4.642e-02  1.373e-02   3.382 0.000778 ***
indus       2.056e-02  6.150e-02   0.334 0.738288
chas       2.687e+00  8.616e-01   3.118 0.001925 **
nox       -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
rm         3.810e+00  4.179e-01   9.116 < 2e-16 ***
age         6.922e-04  1.321e-02   0.052 0.958229
dis       -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
rad         3.060e-01  6.635e-02   4.613 5.07e-06 ***
tax       -1.233e-02  3.760e-03  -3.280 0.001112 **
ptratio    -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
black       9.312e-03  2.686e-03   3.467 0.000573 ***
lstat     -5.248e-01  5.072e-02  -10.347 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.745 on 492 degrees of freedom
Multiple R-squared:  0.7406,    Adjusted R-squared:  0.7338
F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

53. Now, to calculate some error measure for the model, run the next cell/block of code.

```
In [8]: pred <- predict(lm1)
error <- pred - Boston$medv
mae <- mean(abs(error))
rmse <- sqrt(mean((error)^2))
rae <- mean(abs(error)) / mean(abs(Boston$medv - mean(Boston$medv)))
rse <- mean((error)^2) / mean((Boston$medv - mean(Boston$medv))^2)

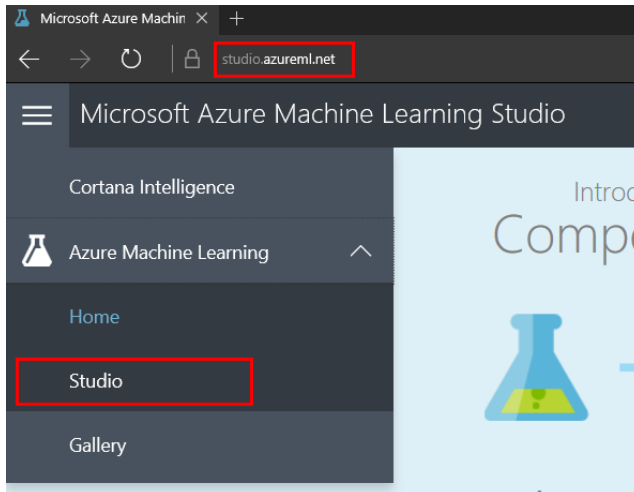
cat("Mean Absolute Error:", round(mae, 6), "\n")
cat("Root Mean Squared Error:", round(rmse, 6), "\n")
cat("Relative Absolute Error:", round(rae, 6), "\n")
cat("Relative Squared Error:", round(rse, 6), "\n")
```

The Output should look like one of the following figures depending on the tool:

```
Mean Absolute Error: 3.270863
Root Mean Squared Error: 4.679191
Relative Absolute Error: 0.492066
Relative Squared Error: 0.259357
```

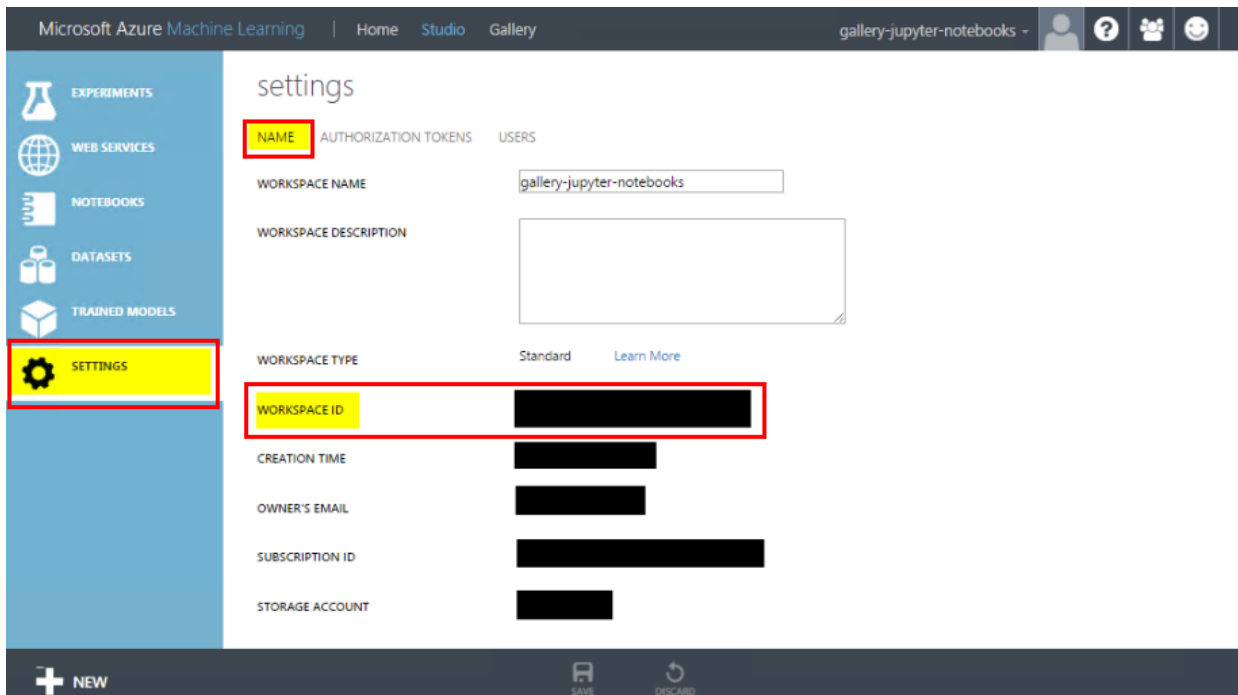
```
Mean Absolute Error: 3.270863
Root Mean Squared Error: 4.679191
Relative Absolute Error: 0.492066
Relative Squared Error: 0.259357
> |
```

54. So, now we have a trained linear regression model that can predict one value based on 13 inputs. One of the easiest ways to operationalize this trained model so that one can use it programmatically for scoring workflows, is to deploy it as a web service in Azure Machine Learning (Azure ML). The AzureML R package will be used for this purpose. Please open a browser window and log in to Azure ML Studio <https://studio.azureml.net/> :

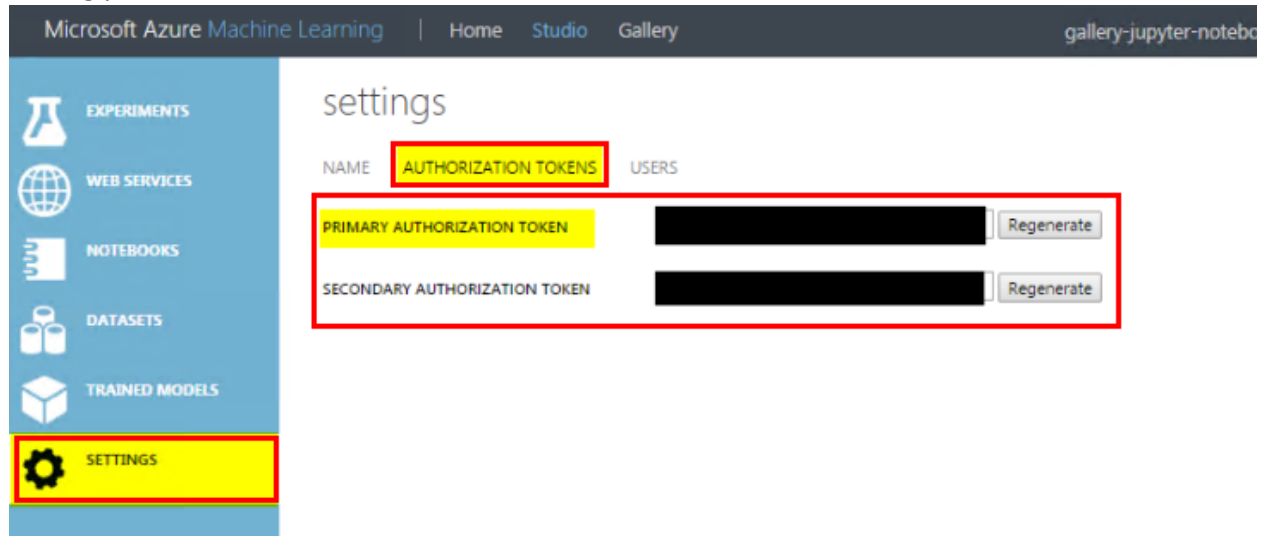


Important: Publishing and consuming a web service requires you to have valid Azure ML credentials. Concretely, if you logged in using a Guest account, you will get an error. If this is the case, please log in using your Microsoft or Azure account and try again.

You'll need to provide the **workspace ID** and **authorization token** for an Azure ML workspace. Finding your workspace ID:



Finding your authorization token:



55. If you are using RTVS, make sure you install the 'AzureML' package first:

```
> install.packages("AzureML")  
also installing the dependencies 'base64enc', 'miniCRAN', 'uuid'
```

56. We use the function `publishWebservice()` to deploy the model. Run the next code cell:

```
In [10]: # load the library  
library(AzureML)  
  
# If you use workspace() in a Jupyter notebook, you don't need to specify credentials,  
# since the settings are stored for you in a local file.  
# If you use this function on your own machine, specify your credentials. See ?workspace.  
  
if(file.exists("~/azureml/settings.json")){  
  ws <- workspace()  
} else {  
  workspace_id <- "ENTER AZURE ML WORKSPACE ID (SEE ABOVE)"  
  authorization_token <- "ENTER AZURE ML WORKSPACE AUTH TOKEN (SEE ABOVE)"  
  ws <- workspace(workspace_id, authorization_token)  
}  
  
# define predict function  
mypredict <- function(newdata){  
  predict(lm1, newdata)  
}  
  
# a sample with predictor information  
newdata <- Boston[1:5, ]  
  
# test the prediction function  
data.frame(  
  actual = newdata$medv,  
  prediction = mypredict(newdata))
```

```

> library(AzureML)
> # If you use workspace() in a Jupyter notebook, you don't need to specify credentials,
+ # since the settings are stored for you in a local file.
+ # If you use this function on your own machine, specify your credentials. See ?workspace.
+
+ if (file.exists("~/azureml/settings.json")) {
+   ws <- workspace()
+ } else {
+   workspace_id <- "XXXXXXXXXXXX"
+   authorization_token <- "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
+   ws <- workspace(workspace_id, authorization_token)
+ }
+
+ # define predict function
+ mypredict <- function(newdata) {
+   predict(lm1, newdata)
+ }
+
+ # a sample with predictor information
+ newdata <- Boston[1:5,]
+
+ # test the prediction function
+ data.frame(
+   actual = newdata$medv,
+   prediction = mypredict(newdata))

```

The Output should look like one of the 2 figures that follow:

Out[10]:	<table border="1"> <thead> <tr> <th></th><th>actual</th><th>prediction</th></tr> </thead> <tbody> <tr> <td>1</td><td>24</td><td>30.00384</td></tr> <tr> <td>2</td><td>21.6</td><td>25.02556</td></tr> <tr> <td>3</td><td>34.7</td><td>30.5676</td></tr> <tr> <td>4</td><td>33.4</td><td>28.60704</td></tr> <tr> <td>5</td><td>36.2</td><td>27.94352</td></tr> </tbody> </table>		actual	prediction	1	24	30.00384	2	21.6	25.02556	3	34.7	30.5676	4	33.4	28.60704	5	36.2	27.94352	<pre> actual prediction 1 24.0 30.00384 2 21.6 25.02556 3 34.7 30.56760 4 33.4 28.60704 5 36.2 27.94352 > </pre>
	actual	prediction																		
1	24	30.00384																		
2	21.6	25.02556																		
3	34.7	30.5676																		
4	33.4	28.60704																		
5	36.2	27.94352																		

Now use the `publishwebService()` function to deploy the web service to the workspace in Azure ML. Run the next cell of code:

```

In [11]: # publish the service
          ep <- publishWebService(ws = ws,
                                fun = mypredict,
                                name = "HousePricePrediction",
                                inputSchema = newdata)

          # str(ep)
          converting `inputSchema` to data frame

> # publish the service
+ ep <- publishWebService(ws = ws,
+   fun = mypredict,
+   name = "HousePricePrediction",
+   inputSchema = newdata)
+ # str(ep)
+

```

57. The Web Service has now been published and is now ready to be consumed by scoring requests. After setting up a web service, you can use R to consume the model, either in the same session, or by saving some data about the web service, in any other (future) R session. We use the 'consume()' function to consume a web service from R and the arguments are 'endpoint' and 'newdata'.

58. In-session consumption:

If you are consuming the web service in the same session that the web service was set up, you can refer to the endpoint directly.

Important: Publishing and consuming a web service requires you to have valid Azure ML credentials. Concretely, if you logged in using a Guest account, you will get an error. If this is the case, please log in using your Microsoft or Azure account and try again.

Run the next 2 code cells to request a prediction from the web service:

```
In [12]: pred <- consume(ep, newdata)$ans
# Note that this operation may take several seconds to complete.
# You may see several retry attempts, before the service returns with a value

In [13]: data.frame(actual = newdata$medv, prediction = pred)
```

You should see a response like the 2 following figures depending on your choice of Jupyter or RTVS:

Out[13]:

	actual	prediction
1	24	30.00384
2	21.6	25.02556
3	34.7	30.5676
4	33.4	28.60704
5	36.2	27.94352

```
> pred <- consume(ep, newdata)$ans
+ # Note that this operation may take several seconds to complete.
+ # You may see several retry attempts, before the service returns with a value
> data.frame(actual = newdata$medv, prediction = pred)
  actual prediction
1  24.0    30.00384
2  21.6    25.02556
3  34.7    30.56760
4  33.4    28.60704
5  36.2    27.94352
>
```

59. Out-of-session consumption:

If you consume the web service in a new session, you need to save the workspace information - workspace id and authorization token - and web service ID. Such information can then be used by the `consume()` function:

Important: Publishing and consuming a web service requires you to have valid Azure ML credentials. Concretely, if you logged in using a Guest account, you will get an error. If this is the case, please log in using your Microsoft or Azure account and try again.

In the next section of the code, we will store the workspace information in a temporary file, then read it back into a variable and then consume the predictions from the web service using the stored endpoint data. Now, run the next cell of code to store the data into the file and read it back.

```
# Option 1: Create a temporary file to store the endpoint data
tf <- tempfile(fileext = ".rds")
saveRDS(ep, file = tf)

# Read the endpoint data from file
endpoint <- readRDS(tf)

# # Option 2: save workspace ID, authorization token and service ID
# ws_id <- ws$id
# ws_auth <- ws$.auth
# # save web service ID
# service_id <- ep$WebServiceId
# # define workspace, this is necessary if you are running outside of the service deployment session
# ws <- workspace(
#   id = ws_id,
#   auth = ws_auth
# )
# # definend endpoint based on workspace and service ID information
# endpoint <- endpoints(ws, service_id)
```

We can again consume the service by running the code in the next cell:

```
# consume
pred <- consume(endpoint, newdata)$ans
# check predictions
data.frame(
  actual = newdata$medv,
  prediction = mypredict(newdata))
```

You should see a response like the 2 following figures depending on your choice of Jupyter or RTVS:

	actual	prediction
1	24	30.00384
2	21.6	25.02556
3	34.7	30.5676
4	33.4	28.60704
5	36.2	27.94352

```
actual prediction
1 24.0 30.00384
2 21.6 25.02556
3 34.7 30.56760
4 33.4 28.60704
5 36.2 27.94352
>
```

-
60. Now we know how to publish and consume models from Azure as Web Service Endpoints. So if you now make improvements to a model, you can update the existing web service. For this purpose you can use the `updateWebService()` function by specifying the web service ID.

The Next block has some sample code that you may chose to run to update the existing web service with a newer model.

```
In [11]: # To update a web service, you need the WebServiceId

# define test function
mypredictnew <- function(newdata){
  predict(lm1, newdata) + 100 # Add 100 for illustration purpose only
}














# update service with the new function
ep_update <- updateWebService(
  ws = ws,
  fun = mypredictnew,
  name = "not necessary", # this does not matter since serviceId is provided
  inputSchema = newdata,
  serviceId = ep$WebServiceId
)

# str(ep_update)
```

61. We just completed a simple example for successfully building a model, publishing it as a web service and scoring using this web service all using R and Azure ML enabled on the DSVM. You may repeat this with Python and consume using other application programming frameworks, with the model deployed to numerous other operationalization architectures like R Services on SQL Server, Microsoft R Server, Spark, Python on Flask or CherryPy etc.

Conclusion + Next Steps

This guide was designed to get you introduced to the DSVM and how to set one up. Additionally, we wanted to get you familiar with the installed tools, SDKs etc. This was intended to walk you through a typical Data Science Workflow using the Data Science Virtual Machine in Azure and other related technologies. Please do go through the numerous other examples that we have already provided on the VM to get familiar with other combinations of resources.

- ☐  [CNTK2](#)
- ☐  [DataScienceSamples](#)
- ☐  [mxnet](#)
- ☐  [Analyzing PyPI Data to Determine Python 3 Support.ipynb](#)
- ☐  [Comparing_Base_R_to_MRO_and_MRS.ipynb](#)
- ☐  [DocumentDBSample.ipynb](#)
- ☐  [Introduction to Azure ML R notebooks.ipynb](#)
- ☐  [IntroToJupyterPython.ipynb](#)
- ☐  [IntroTutorialinR.ipynb](#)
- ☐  [IrisClassifierPyMLWebService.ipynb](#)
- ☐  [LoadDataIntoDW.ipynb](#)
- ☐  [SQL_R_Services_End_to_End_Tutorial.ipynb](#)
- ☐  [SQLDW_Explorations.ipynb](#)

If you have questions and or want to discuss, please visit the Data Science Virtual Machine Forum at:

<http://aka.ms/dsvmforum>

*- Barnam Bora
Program Manager - Microsoft*