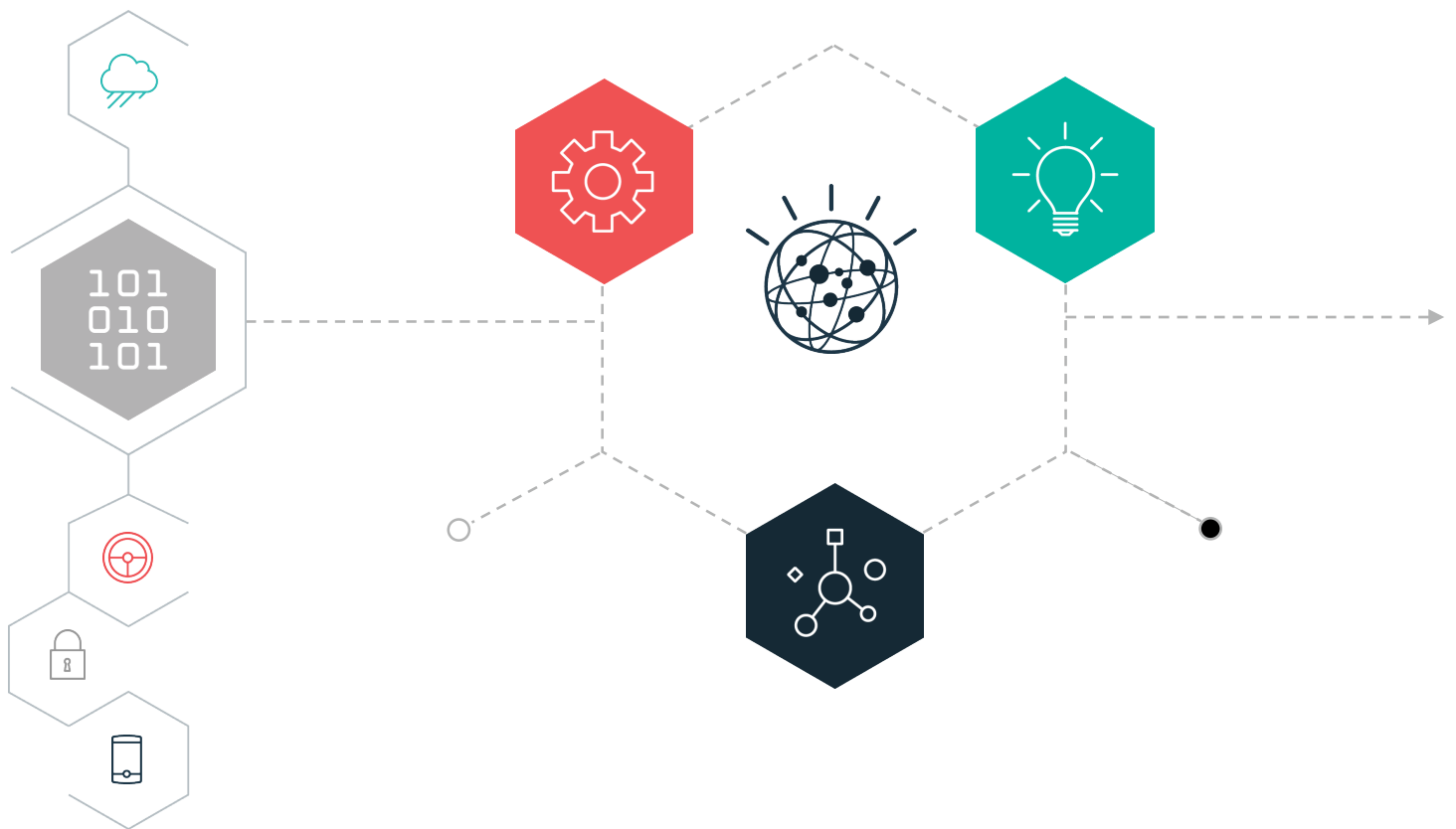


IBM Data Science Experience Overview

Student Lab Manual – Draft v1.0



Contents

	OVERVIEW	4
LAB 1	DATA SCIENCE EXPERIENCE (DSX)	5
	1.1 EXPLORE THE COMMUNITY CONTENT	5
	1.2 CREATE A PROJECT	9
	1.3 ADD COLLABORATORS TO A PROJECT	11
	1.4 ADD A DATA FILE TO A PROJECT	12
LAB 2	JUPYTER NOTEBOOKS LAB.....	14
	2.1 CREATING A BLANK NOTEBOOK	14
	2.2 CREATING A NOTEBOOK FROM A FILE.....	18
	2.3 CHANGING YOUR KERNEL	20
	2.4 CREATING A NOTEBOOK FROM A URL	22
	2.5 SAVING YOUR NOTEBOOK.....	23
LAB 3	RSTUDIO.....	25
	3.1 LAUNCH RSTUDIO IN DSX	25
	3.2 CONNECT TO SPARK FROM RSTUDIO	25

Overview

IoT is driving digital disruption of the physical world, accelerating advances in technology, transforming every part of our lives. Cognitive IoT infuses new intelligence into products, services and processes.

Start by deploying a highly secure, scalable, and open platform that lets you start small, and grow quickly.

Introduction

The IBM IoT Watson Platform is available in the Cloud. It has extensive capability to facilitate agile application development for IoT ecosystems. In this Proof-of-Technology, you will learn how to create an IoT ecosystem using containers, Node-RED, Cloudant, dashDB and Spark. Let us explore this innovative offering.

Lab 1 Data Science Experience (DSX)

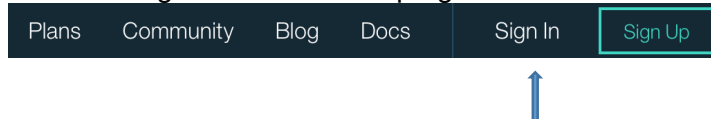
Before starting the real work of these labs, the student needs to register for the Data Science Experience platform. Navigate to <http://datascience.ibm.com> and create an account using the “Sign Up for a Free Trial” link.

This lab will go over navigating the Community content in DSX as well as project management. You will add collaborators and data assets to a project.

1.1 Explore the Community Content

__1. The first step to using DSX is to navigate to <http://datascience.ibm.com/>

__2. Click the “Sign In” link in the top right corner of the window.



__3. Enter your credentials when prompted.

- ___4. You should arrive at the DSX home page, with recently published community content at the top, recently updated projects in the middle, and helpful links at the bottom.

IBM Data Science Experience

Projects Tools Data Services Community US South Docs DM

Get started Create new

Recently published

Explore

ARTICLE

Challenges in Deep Learning

AUTHOR: ParallelDots DATE: Jul 20, 2017

TOPIC: Deep Learning FORMAT: Web page

0

DATA SET

Breast Cancer Wisconsin (Diagnostic) Data Set

AUTHOR: IBM DATE: Jun 28, 2017

TOPIC: Health

1

NOTEBOOK

Analyze Facebook Data Using IBM Watson and...

AUTHOR: IBM DATE: Jul 18, 2017

TOPIC: Economy & Business

9

TUTORIAL

Using Machine Learning to Predict Value of...

AUTHOR: Airbnb Engineering & Data Science DATE: Jul 21, 2017

LEVEL: Beginner TOPIC: Machine Learning

0

Recently updated projects

View all (1) New project

NAME	ROLE	COLLABORATORS	DATE CREATED	LAST UPDATED
Default Project	Admin	DM	Jul 20, 2017	Jul 21, 2017

Helpful links

Docs

Find the information you need. Watch videos of key tasks.

Discussion forum

Stack Overflow is a community of 6.9 million programmers just like you, helping each other. Join the conversation on Data Science Experience.

Blog

Read and follow our blog to keep up with the latest updates about Data Science Experience.

Got ideas?

Have feedback on Data Science Experience? Submit your ideas in our product forum or vote on ideas submitted by others.

Blog Documentation Contact Privacy Terms of Use

- ___5. Click the Explore link by the Recently published community content to review some of the entries. Note that there are the following four types of entries:

Articles	Links to data science related articles from across the Internet
Data Set	Links to publicly available data sets that you can experiment with in your projects
Notebook	Links to publicly available notebooks that demonstrate some data science process or methodology
Tutorial	A walkthrough of data science techniques

___6. Select one of each type of entry to investigate. Some of them open in a new browser tab while others open in the same tab.

___	Article	___	Data Set
___	Notebook	___	Tutorial


___7. You can filter the community content by clicking the links above the list, selecting to display only one of the types of content. Try a few now. You can view all content again by clicking All.

___8. Type in a word or phrase in the Search box above the content list. DSX automatically filters the content according to the keywords that you enter. Try “Airbnb”.

___9. Note that in the bottom right hand corner of each of the community “tiles” there are three icons.

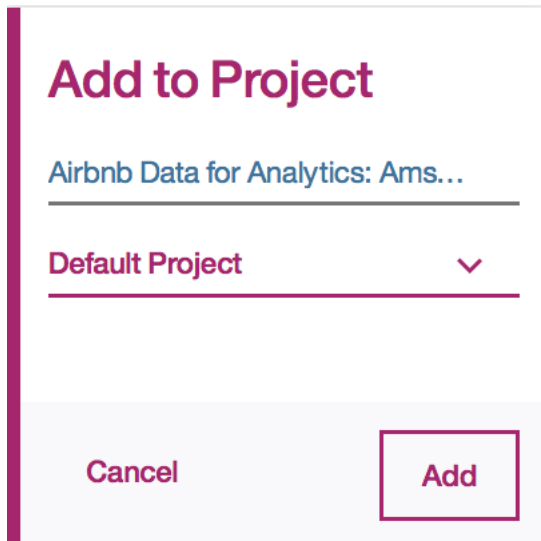


The heart  indicates how many users have “liked” that post.

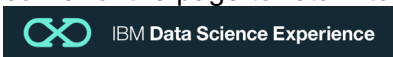
The bookmark  allows you to capture a link to the content as a Bookmark in a project.

The plus sign  is displayed only for Data Set tiles, and allows you to add that data to a project.

- ___10. Pick one Data Set and click the plus sign to add it to your “Default Project”. When you click the icon, the tile changes to a dialog tile where you can select a project to add the data set to. Every DSX user should initially have a “Default Project” automatically.

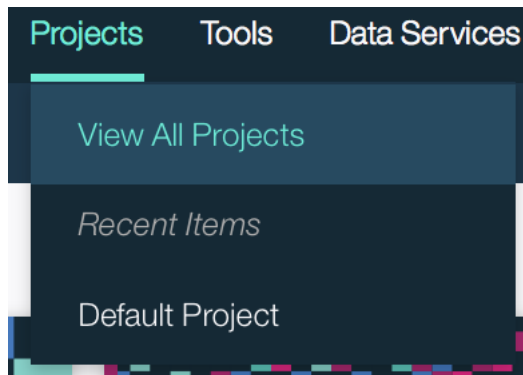


- ___11. Pick one other type of tile and click the bookmark to add a link to it to your “Default Project”. Note that you could also click the “Cancel” link to avoid actually adding the bookmark.
- ___12. At any point in your navigation, you can click the Data Science Experience logo at the top left corner of the page to return to the DSX home page. Click it now to return.

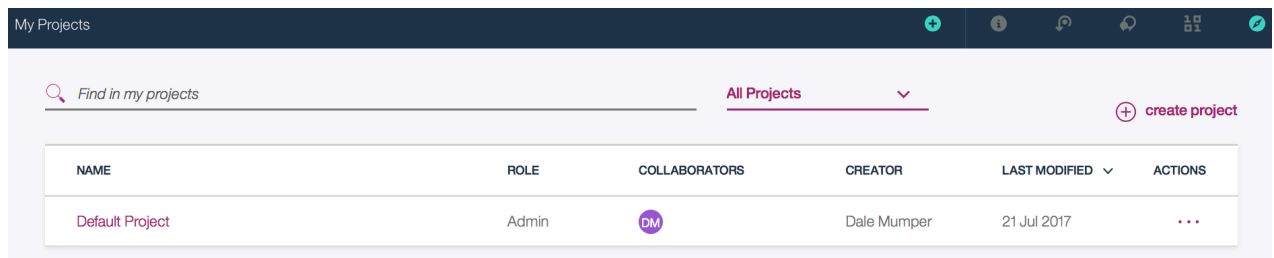


1.2 Create a project

- ___13. From the top DSX menu bar, hover over Projects and then select “All Projects”



- ___14. You should see at least the “Default Project” listed in the list of your projects. The projects that are in the list are those that you either created yourself or were granted access to by others.



- ___15. If you click on the three dots (ellipsis) in the Actions column for a project, you’ll find two options. Clicking Delete will prompt you to confirm that you do actually want to delete the project. Clicking Edit (or clicking on the hyperlink name of the project) will bring you to the Overview page for the project.

__16. Click the “create project” link to create your own project now.

 [create project](#)

__17. In the “Create new project” page, give your project a distinct name. Something like “Johns New Project” should suffice here, where you replace “John” with your own name.

__18. You can enter a description if desired, but it is not necessary.

__19. Leave the Spark Service, Storage Type, Target Object Storage instance, and Target Container items at their default values.

__20. Click Create to create your project.

__21. At the Overview page of the project, click each of the page headers in turn to see the kinds of assets that are visible from each. Analytics Assets, Data Assets, Bookmarks, Collaborators, and Settings.

[Overview](#) [Analytics Assets](#) [Data Assets](#) [Bookmarks](#) [Collaborators](#) [Settings](#)

Notebooks

[view all \(0\)](#) [+ add notebooks](#)

NAME	SHARED	STATUS	LANGUAGE	LAST EDITOR	LAST MODIFIED	ACTIONS
you currently have no notebooks						

Data Assets

[view all \(0\)](#) [+ add data assets](#)

NAME	TYPE	SERVICE	LAST MODIFIED	ACTIONS
you currently have no data assets				

Bookmarks

[view all \(0\)](#) [+ explore community](#)

you currently have 0 bookmarks

- __22. Click on the Settings header to review the Storage, Associated Services, Access Tokens, and GitHub Repository options. DSX comes with an instance of Spark-as-a-Service, and it should be indicated here. If you have access to other Spark instances you could switch the project to using that one on this page.

1.3 Add collaborators to a project

- __23. Click on the Collaborators header. On this page, you can invite coworkers or external people to access your project. Your own name should be listed as an Admin.

- __24. Click on the “Add collaborators” link to invite someone to your project.

Dales New Project


Add Collaborator

Collaborators

Admin (1) ▾

demo01@mumpermachine.com

Invite

Search for name or enter email address 

Access Level

Select an option ▴

Viewer

Editor

Admin

Add

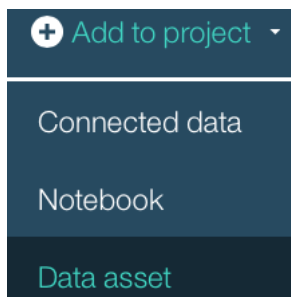
Cancel

Invite

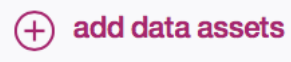
- ___25. In the Invite text area, type in the name or email address of the person you'd like to invite to your project. If the person is defined in your Enterprise DSX organization, it should offer you matching names as you type. If they are outside of your organization, you can still just type their email address and they will be emailed an invitation. If they do not have a DSX account, they'll be prompted to create one.
- ___26. Select the Access Level to invite the user as, either Viewer, Editor, or Admin.
- ___27. Click the Invite link to complete the addition of the collaborator.
- ___28. Once the invited people have accepted their invitations, they should appear in the list of Collaborators.
- ___29. In the DSX menu, hover over Projects and click "All projects". If anyone invited you to their projects you'll see them listed in the project list. Note that if we had all invited each other to our own "Default Project", we'd have that many projects with the same name in our list! That's why it's a good practice to give your projects meaningful names.

1.4 Add a data file to a project

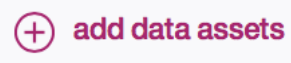
- ___30. From your list of projects, click on the name of "Default Project".
- ___31. There are couple different ways to add a data file to your project.
- From the Action Bar, click "Add to project" and then "Data Asset"



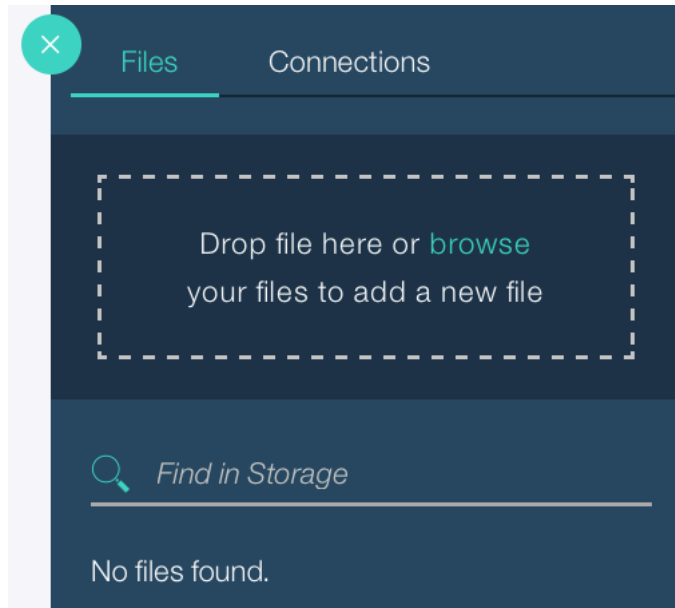
- From the Overview page, click "add data assets" from above the list of data assets



- From the Data Assets page, click "add data assets"



- __32. Note that all three methods seem equally ineffective. By default, there is a side window opened to allow you to easily add Data Assets. But if you had closed that by clicking the X, the three methods listed above would have opened it again!



- __33. Click the browse button to open up a dialog window to select the file to add. If you don't have one handy, you can use this one from a public box folder I set up. Just download the file to your local system first.
<https://github.com/dale-mumper-ibm/dsx-overview-class/blob/master/wisc-diag-breast-cancer-shuffled.csv>
- __34. Once the file is uploaded, an Apply button will be enabled for you to confirm the addition of the Data Asset to the project. Click the Apply button when it lights up.
- __35. The file is now available for use in your project.

Lab 2 Jupyter Notebooks lab

This lab introduces you to the Jupyter notebooks capabilities in DSX. You will learn how to create notebooks from several different sources, how to manage your kernel, how to add Data Asset references into your notebook, and how to save your notebook. While Jupyter notebooks are intended to be created by coding data scientists, you don't need any coding experience to complete this lab.

2.1 Creating a blank notebook

- ___1. From the DSX menu bar, hover over Projects and click on "Default Project" to open it. If it is not listed, click "All projects" and select it from the resulting list.
- ___2. A notebook counts as an "Analytic Asset", so you can look at that header for any notebooks in the project, or just in the "Analytics Assets" section of the Overview page.
- ___3. Chances are you have none at this point, so let's create a new one! Click the "add notebooks" link.

- ___4. The “Create Notebook” window appears. You have the option of creating a Blank notebook, importing a notebook “From File”, or importing a notebook “From URL”. Leave the selection at the default of Blank.

Create Notebook

Blank

From File

From URL

Name*

Type Notebook Name here

Description

Type your Description here

Language*

☒ Python 2 ☐ R ☐ Scala ☐ Python 3.5

Experimental

Spark version*

☐ 2.1 ☒ 2.0 ☐ 1.6

Spark Service*

DSX-Spark



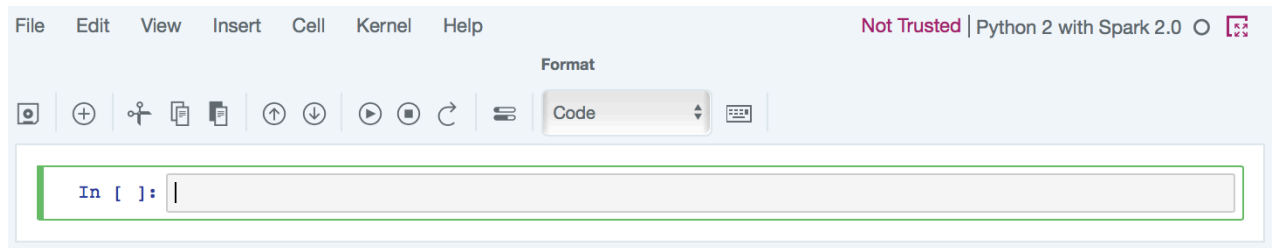
Associate this notebook with the Spark Service of your choice.


Cancel

Create Notebook

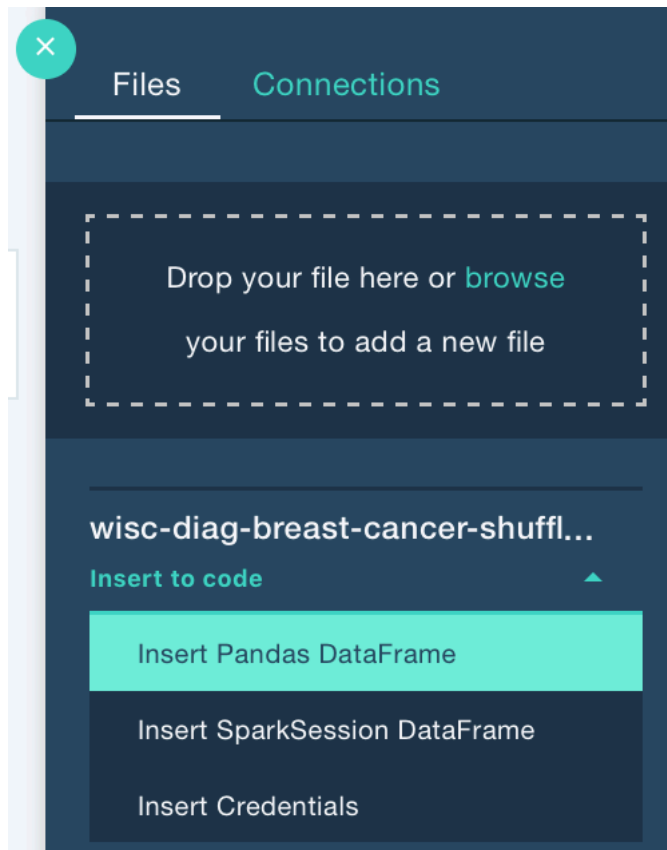
- ___5. Enter a descriptive notebook name. Something like “My very first DSX notebook” will suffice.
- ___6. You have the option of setting the language to Python (version 2 or 3.5), R, or Scala. For this lab, select Python 2.

- ___7. Different versions of Spark kernels are available, each with slightly different capabilities. Select 2.0 here.
- ___8. Click the “Create Notebook” button. You’re delivered into the Jupyter notebook interface, with an empty cell waiting for you.



- ___9. Let’s add our data file into our notebook. If your Data Assets sidebar isn’t visible, click the icon for it in the Action Bar  for it in the Action Bar

- ___10. The list of data assets and connections available to the project are listed in the sidebar, including the one that we added earlier. Click the “Insert to code” link and select “Insert Pandas DataFrame”



- ___11. A snippet of code will be inserted with all of the I/O plumbing needed to access the file in the object store and read it into the notebook as a Pandas DataFrame. It's a lot of code, so I'm glad it's inserted for me.

- ___12. Make sure your cursor insertion point is in that code cell and click the “Run Cell” toolbar button

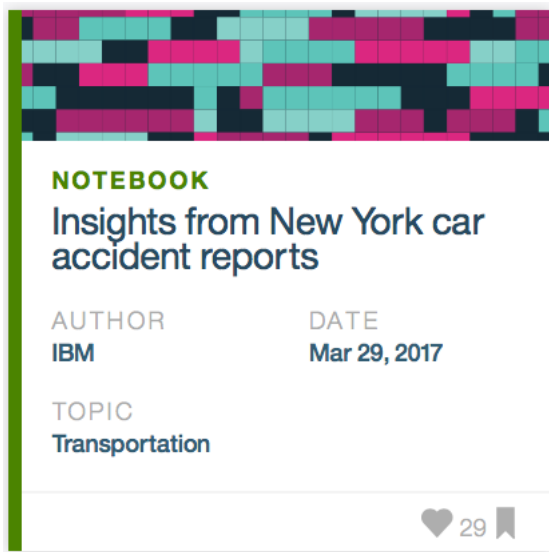


- ___13. The code cells in a Jupyter notebook each have an In segment associated with them, and after execution an order of execution number is placed in the square brackets next to it. If there is some output then an additional Out segment is created. During execution there is an asterisk (*) in the square brackets. Some long running commands can be an asterisk for a while.

- ___14. This block of code ends with a `df_data_1.head()` command, returning the top rows from the dataframe.

2.2 Creating a notebook from a file

- ___15. Now let's create a notebook in our project that is based on a pre-existing notebook. This often happens when you work through a notebook tutorial or want to review someone else's work.
- ___16. The first step is to get a copy of an ipynb file. We can grab one from the Community content in DSX. Click on Community in the DSX menu bar.
- ___17. Click the Notebooks link so the search results will be filtered to just notebooks.
- ___18. In the search box, enter "New York car accidents". One of the results will be "Insights from New York car accident reports".



- ___19. Click on the title of the tile to go to the details page for the notebook.
- ___20. In the top right corner of the window, click the download button from the Action Bar. Download the file to your local system.



- __21. After the download has completed, return to your project by hovering over “Projects” and selecting “Default Project”
- __22. From the Overview page of your “Default Project”, click “add notebooks”
- __23. At the top of the “Create a notebook” page, click the “From File” link.
- __24. Enter a name for your copy of the notebook. Something like “My copied notebook” will work.
- __25. Click the “Choose file” button and select the ipynb file you downloaded above.
- __26. Click “Create Notebook” to upload the ipynb file, have it added to your project, and enter edit mode on the copied notebook.
- __27. While you’re in the “Insights from New York car accident reports” notebook, scroll down and review some of the code. You’ll see lines like these that show the extensibility of Jupyter in the DSX environment:

<code>import numpy as np</code>	Imports a standard Python library for use in your notebook execution
<code>!pip install --user seaborn</code>	Installs a 3 rd party Python library for use
<code>%matplotlib inline</code> <code>import matplotlib.pyplot as plt</code>	Tells matplotlib to display charts inline in the notebook and then imports a pyplot library for visualizations

- __28. Stay in this notebook for the next segment of the lab

2.3 Changing your kernel

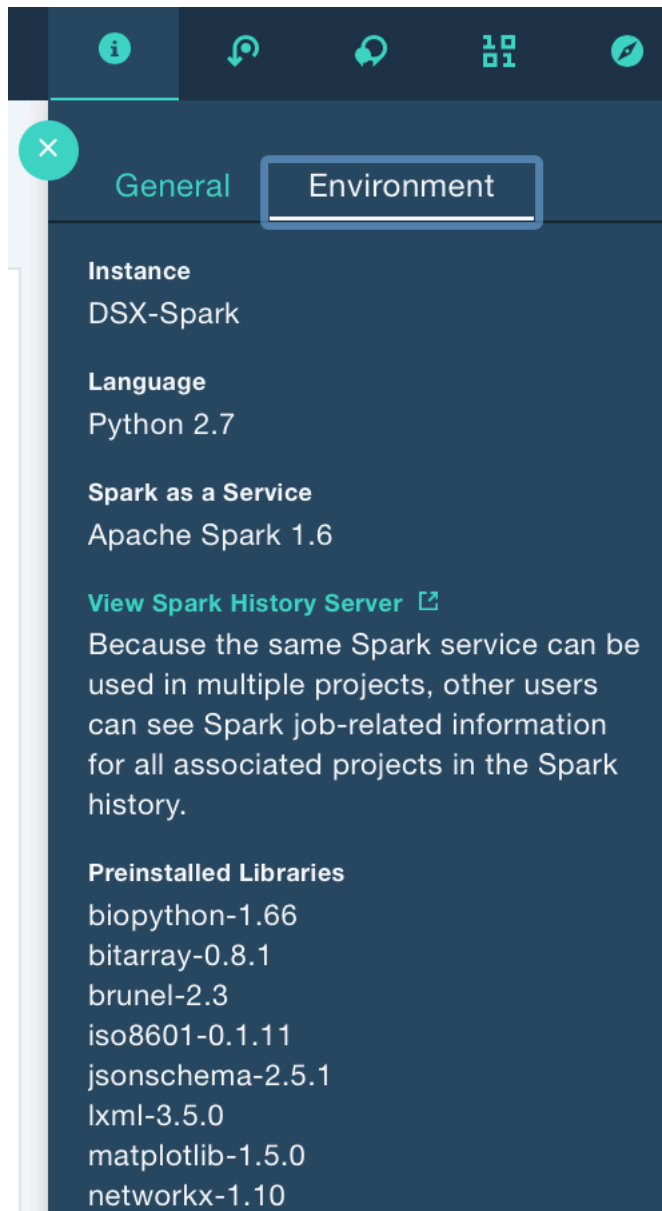
- ___29. You can review the kernel properties that your notebook is executing in by looking at the top right of the Jupyter window. An indication of the language and Spark version will be shown there.

Not Trusted | Python 2 with Spark 1.6 ○ 

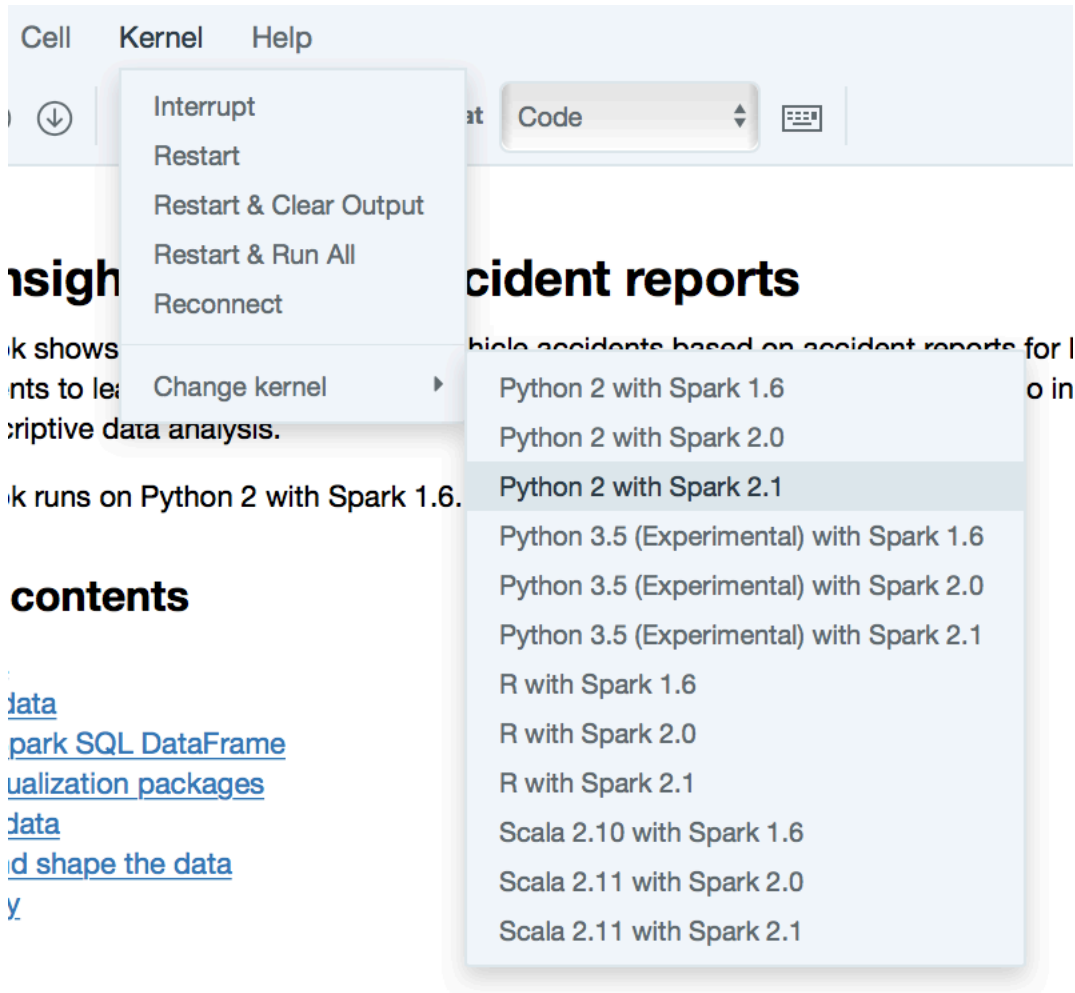
- ___30. Additionally, you can click on the notebook information icon in the Action Bar to see even more details on the environment.



__31. Information on the kernel and default libraries are shown.



- ___32. If you wanted to change the kernel to something else, you can do so from the Jupyter menu. Click Kernel -> Change kernel and then select the desired new kernel. This would be an option for when you wanted to test your Spark 1.6 code against a Spark 2.1 kernel to confirm that it works in that configuration too.



2.4 Creating a notebook from a URL

- ___33. There's not too much different in creating a Jupyter notebook from a URL than the other methods. Try it here with a notebook referenced from github.

<https://github.com/dale-mumper-ibm/dsx-overview-class/blob/master/Introduction%20to%20Spark%20-%20Student%20Lab.ipynb>

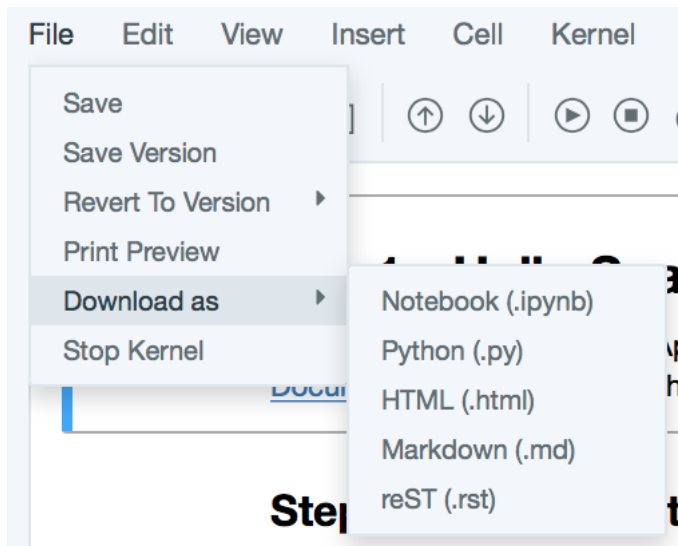
- ___34. Navigate to your Default Project by hovering over Projects and selecting "Default Project"

- __35. Click “add notebooks”
- __36. On the “Create a notebook” page, click the “From URL” option.
- __37. Specify a name for the notebook, such as “My url notebook”
- __38. Enter the url for the notebook from above in the “Notebook URL” area
- __39. Click “Create notebook”

2.5 Saving your notebook

- __40. DSX automatically saves your notebook every few seconds, so there is no need to click the Save button in the Jupyter interface. You can if you want, though.
- __41. If you want to specifically label a notebook as a given “version”, there is the option to do so from the Jupyter menu. Click File -> Save Version. Later you can revert to that saved version in case your changes are undesired.

- __42. Another option for saving notebooks is as a different format entirely, for consumption outside of DSX and Jupyter. Click File on the Jupyter menu and hover over “Download as”. You can download your notebook in several formats.



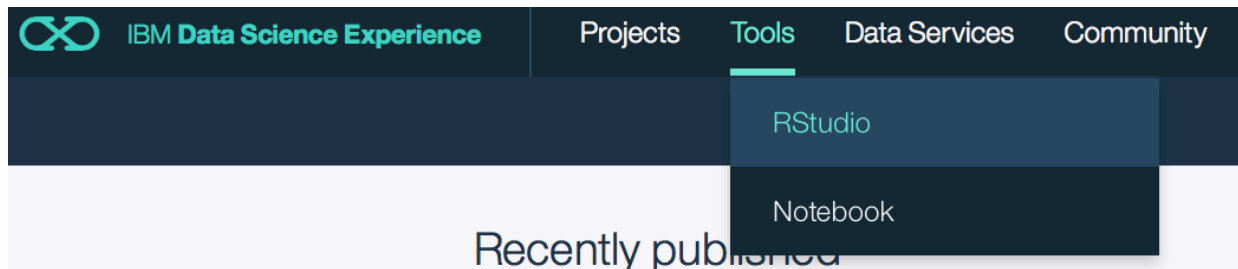
- __43. Select File -> Download as -> HTML (.html) to save your notebook as an html page. This will export all of your markdown cells, code cells, and visualizations.
- __44. When prompted, enter a filename and location to save your rendered html file to.
- __45. After the html file is downloaded, open the file to review how the output looks. Any visualizations that existed in the notebook are embedded into the html file.

Lab 3 RStudio

In this lab the RStudio tool is explored in DSX. Note that this lab is a bit more challenging than the others. We will be referencing the data asset that we stored in our project from inside of RStudio, linking our script to our Spark instance, and loading a dataframe for manipulation. You can just explore the RStudio environment instead of all of this code if you want an easier approach.

3.1 Launch RStudio in DSX

- __1. From the DSX menu, hover over Tools and click on RStudio



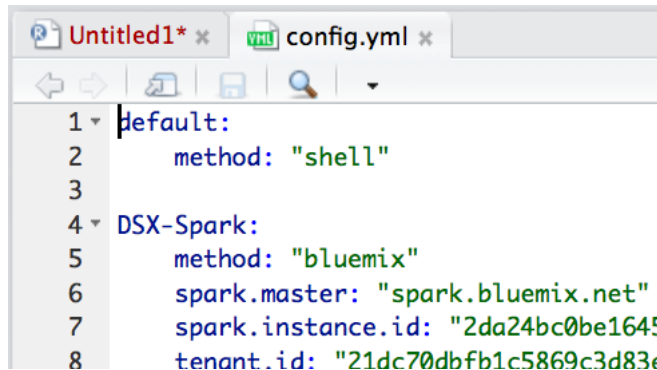
- __2. After a few seconds, a full RStudio development environment is loaded in the browser.
- __3. If you have previous experience with RStudio you should notice that the environment looks just like a locally installed version. The main difference is that it is running on a server within the DSX environment and from inside of a browser.

3.2 Connect to Spark from RStudio

[Based in part on the fine blog post at <https://datascience.ibm.com/blog/read-and-write-data-to-and-from-bluemix-object-storage-in-rstudio-2/>]

- __4. When RStudio is launched for the first time, a new project is created in it, including a config.yml file. This file contains information on the Spark instances accessible through DSX. Click on this file in the Files window in the lower-right hand corner.

- ___5. One of the Spark instances listed should have a method of “bluemix”. Note the name of that entry. It is the value before the colon and indented entries. For example, this file indicates the name of the instance is “DSX-Spark”



- ___6. You can also list the Spark kernels by issuing the list_spark_kernels function and storing the result in a kernels variable for reference.

```
kernels <- list_spark_kernels()
```

```
kernels
```

```
> kernels <- list_spark_kernels()
> kernels
[1] "DSX-Spark"
```

- ___7. In the console window in the lower right-hand corner of RStudio, enter the following code. It loads the required libraries for the rest of the script.

```
#connect to spark
```

```
library(sparklyr)
```

```
library(dplyr)
```

- ___8. Instantiate a Spark context variable, passing in the name of the Spark instance you found above.

```
sc <- spark_connect(config = "DSX-Spark")
```

- ___9. Load the devtools library and then install the sparklyr package

```
library(devtools)
```

```
devtools::install_url("https://github.com/ibm-cds-labs/ibmos2spark/archive/0.0.7.zip", subdir=
"r/sparklyr/", dependencies = FALSE)
```

__10. With the sparklyr package installed, load it

```
library(ibmos2sparklyr)
```

__11. This next step requires that you open a new browser tab and log in to DSX in it. We need to get some of the credential information that DSX inserts for us for connecting to data stored in object storage.

__12. In this new DSX tab, create a new python notebook in your “Default Project”

__13. In the Jupyter notebook editor, click the “Insert to code” link by your data asset and select “Insert credentials”

__14. The inserted text provides the values you need to enter in to the R script in RStudio. Switch back to the RStudio tab now.

__15. Enter the following code, replacing the values in double angle brackets with the values from your “Insert credentials” code.

```
creds <-list(auth_url = "https://identity.open.softlayer.com",  
project = "<<project>>",  
project_id = "<<project_id>>",  
region = "dallas",  
user_id = "<<user_id>>",  
domain_id = "<<domain_id>>",  
domain_name = "<<domain_name>>",  
username = "<<username>>",  
password = "<<password>>",  
container = "<<container>>")
```

__16. Now that we have all of the credentials set, we can connect the spark context with the Bluemix object store, passing in the filename that we want to read from the container.

```
configurationname = "keystone"  
bmconfig = bluemix(sparkcontext=sc, name=configurationname, credentials = creds)
```

```
#change the container name if you want to store it to
#another existing container.
container = creds['container']
objecttoread = "wisc-diag-breast-cancer-shuffled.csv"
sparkobject_name = "dataFromSwift"
```

___17. Finally, we issue the command to have Spark read in the text file, storing it in a variable called data.

```
data = sparklyr::spark_read_csv(sc, sparkobject_name, bmconfig$url(container, objecttoread))
src_tbls(sc)
```

___18. We can display the data with the head function.

```
head(data,4)
```

```
> head(data,4)
```

```
Source: query [4 x 32]
```

```
Database: spark connection master=spark.bluemix.net app=sparklyr local=FALSE
```

```
# A tibble: 4 x 32
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
	<int>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	871001502	B	8.219	20.70	53.27	203.9	0.09405	0.13050
2	8810528	B	11.840	18.94	75.51	428.0	0.08871	0.06900
3	89511501	B	12.200	15.21	78.01	457.9	0.08673	0.06545
4	91594602	M	15.050	19.07	97.26	701.9	0.09215	0.08597

```
# ... with 24 more variables: concavity_mean <dbl>, concave_points_mean <dbl>, symmetry_mean <dbl>,
```

```
# fractal_dimension_mean <dbl>, radius_se <dbl>, texture_se <dbl>, perimeter_se <dbl>, area_se <dbl>,
```