TAP Hands-On Workshop

Module 1: Performing Analytics on Your Data

Overview

This module uses the Analytics Toolkit (ATK) within the Trusted Analytics Platform (TAP) to execute a script that reads data from a TAP server and creates and outputs an amended dataset. The amended dataset becomes the input for Module 2: Visualizing Your Data with an App.

Steps	Time Required
Step 1	Varies based on your current environment
Step 2	15 minutes
Step 3	5 minutes
Step 4	5 minutes
Step 5	30 minutes
Step 6	5 minutes
Total Workshop Time	**60 minutes**

Setup

The data scientist needs to prepare his or her local environment/computer with the tools required to edit, assemble and deploy the module. **Important**: You need to have access to an instance or to set up an instance of ATK.

- The software you need includes:
  - **iPython Notebook** (also known as Jupyter) - a tool for exploratory computation and data analysis that stores inputs/outputs in notebook documents. http://ipython.org/install.html
  - **Python (2.7)** - a widely used general-purpose, high-level programming language. https://www.python.org/downloads/

Objectives

After completing this module, you will have:

- Accessed Marketplace and TAP.
- Demonstrated the ability to create an instance of ATK on your system.
- Used Python or iPython Notebook.
- Accessed the help functions for editing a script**.**
- Shown how to attach a TAP server to your project to input data.
- Executed an ATK script that exports data.
- Demonstrated the ability to install multiple software tools needed to build and operate domain-specific applications.

Module Steps

1. Install all necessary developer tools listed in Setup.
2. Sign into Marketplace, view the Data Catalog functions and add a file into TAP.
3. Access Marketplace to create a new instance of ATK on your system, view the running instance and copy the instance uri to use in the Netflow demo.
4. Netflow demo? Open the Netflow demo file in Python or iPython Notebook, review the help functions and edit the ATK.server.uri field in the file.
5. Execute the Netflow demo to deploy the ATK model. The demo first clears out old frames and graphs. Then it reads in the data, formats and prints the date/time data in columns, builds a Titan graph for computing statistics, computes bytes in and bytes out, computes weighted and unweighted degree counts, computes a summary frame and downloads it to Pandas, computes and plots histograms, makes scatter plots, creates and trains an Support Vector Machine (SVM) model and downloads a scatter frame to Pandas.
6. Export the data to Hadoop Distributed File System (HDFS) as a comma separated value (CSV) file.

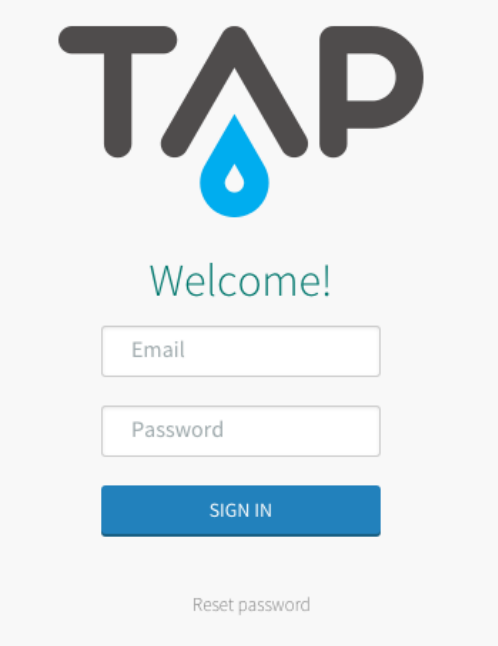
Important: The URLs and input commands provided in the workshop will change based on the instance used and the setup of your network. Check with the workshop instructor or consult with your development operations team (DevOps) to obtain the proper information.

Results

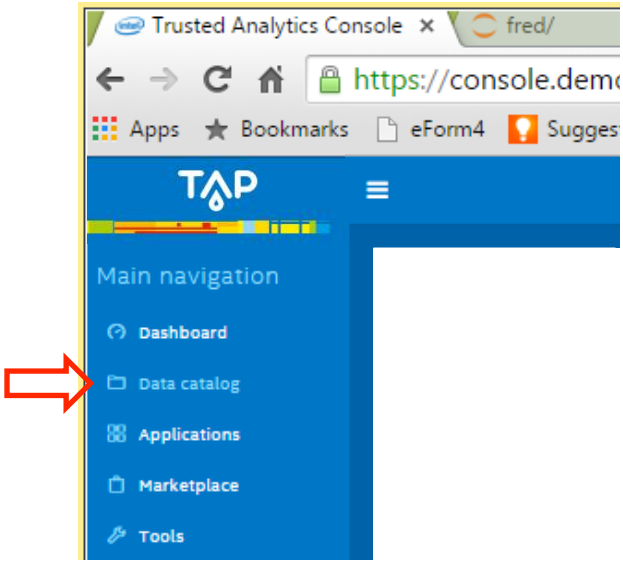
This section provides step-by-step instructions and their results.

1. **Install all necessary developer tools listed in Setup**
   1.1. No additional instructions are provided for this step.

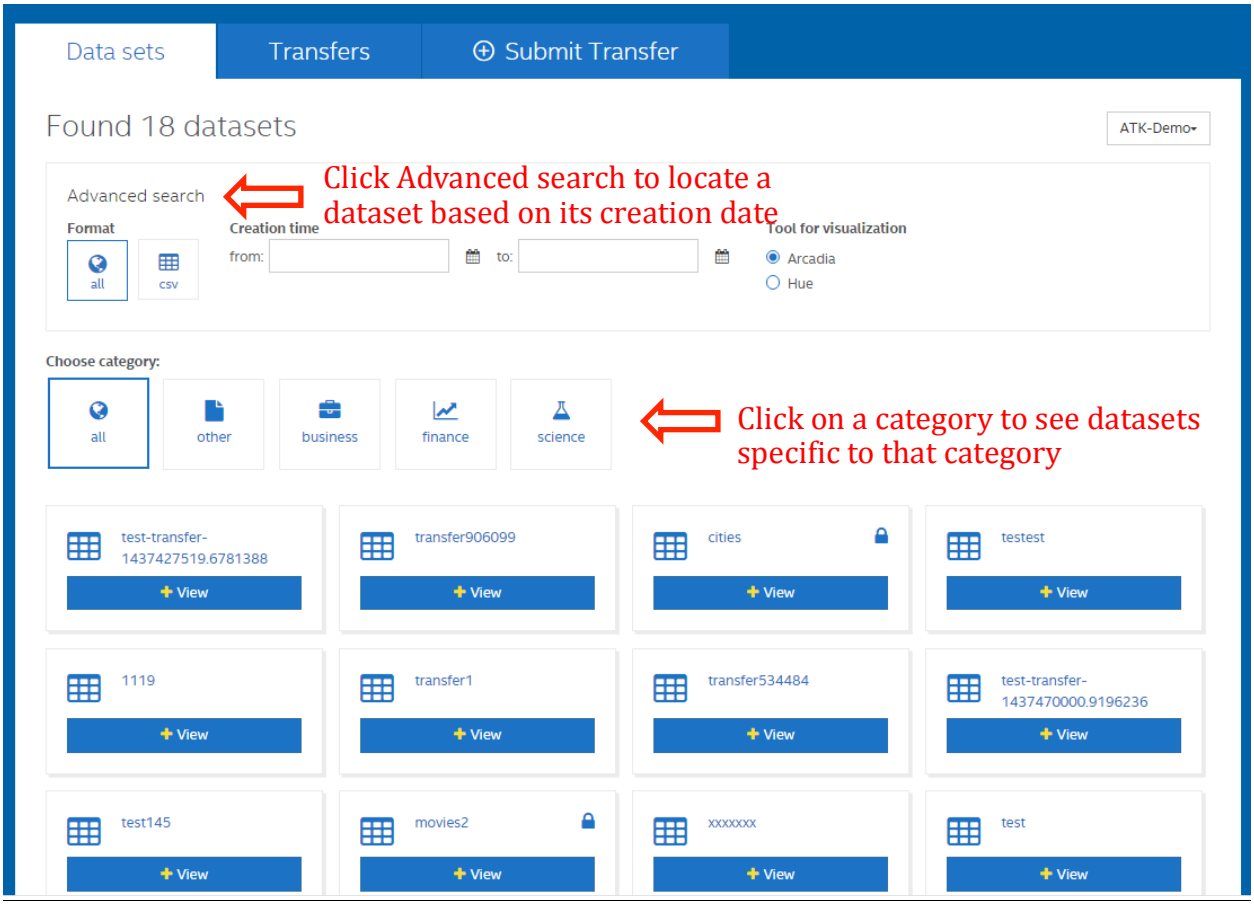
2. **TAP Marketplace**
   2.1. Sign into TAP with your User ID and Password:



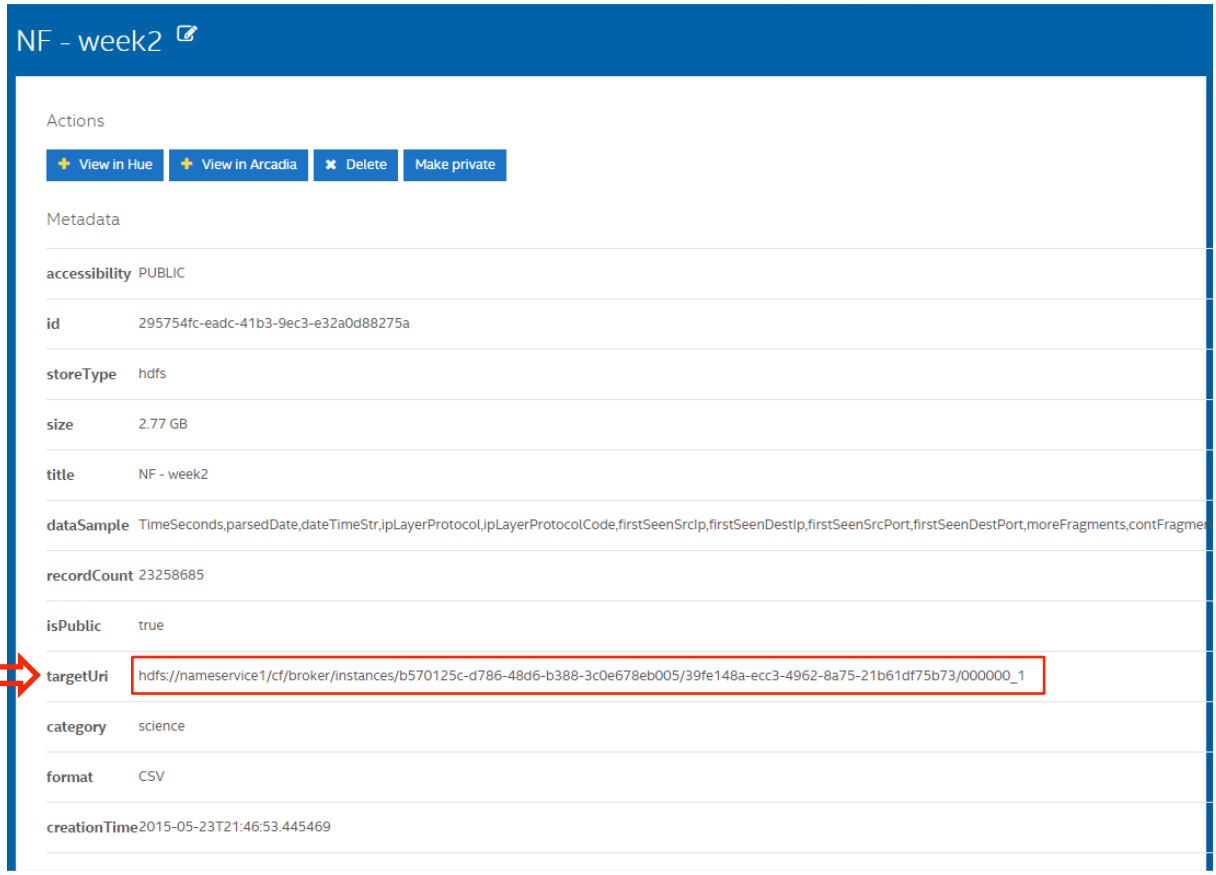
   2.2. View the Data Catalog functions:
   - Click on Data Catalog under Main navigation to access the three Data Catalog tabs – Data sets, Transfers and Submit Transfer.



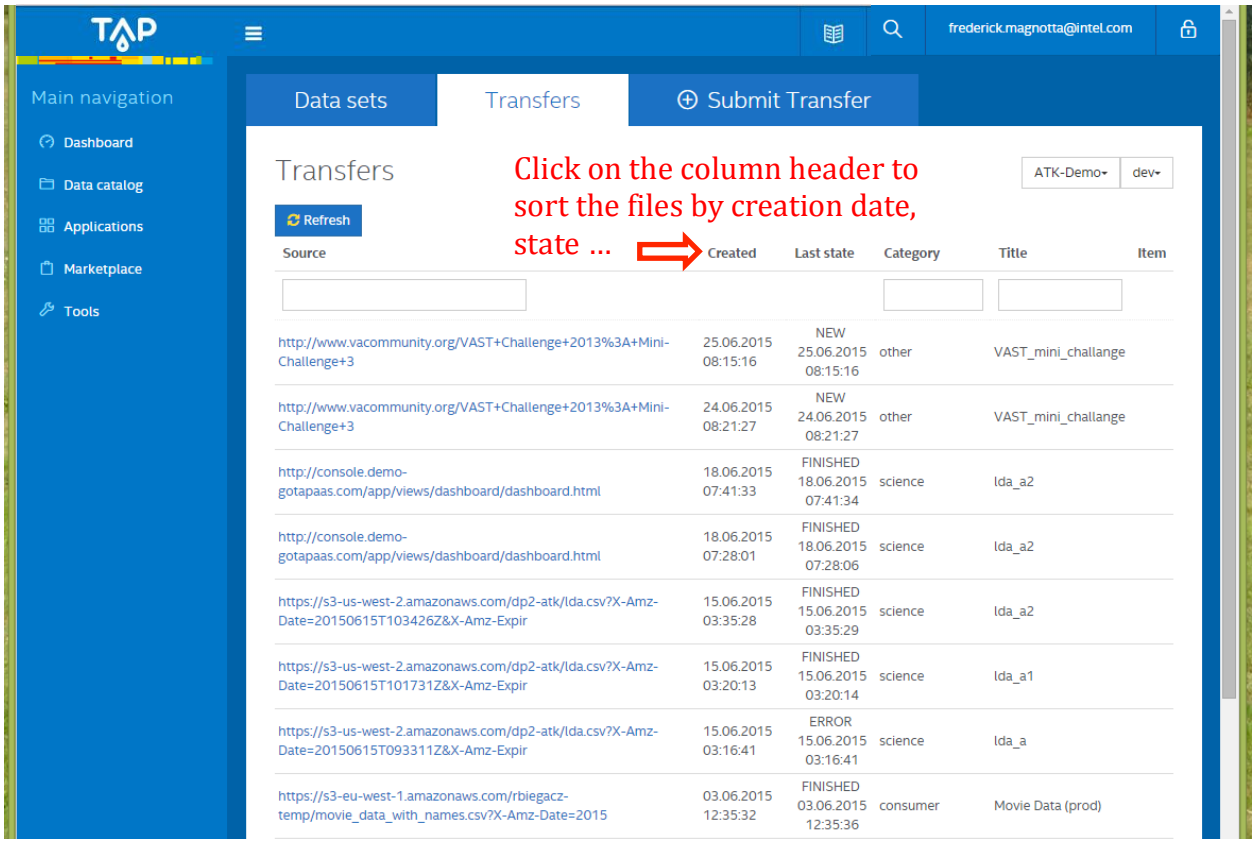
- Click on the **Data sets** tab to see the datasets on TAP:
  - You can see your datasets and all the datasets marked public.
  - In this example, we are in the ATK-Demo group in Marketplace. The ATK-Demo group is only available in the TAP instance. The group name may be different if you use your own instance.
  - You can limit the number of datasets you see by either using the Advanced search option or choosing a specific category.
  - **Note**: Make sure you select the correct group when you look for your dataset in Marketplace.



- Click the **View** button on the Data sets tab to display dataset information, such as:
  - Title, size, category …
  - **TargetUri** is a link to a dataset. This particular dataset link may not be the one needed in this exercise. You need to consult the workshop instructor or your DevOps team to obtain the proper information
  - You can copy the link to use in other apps.

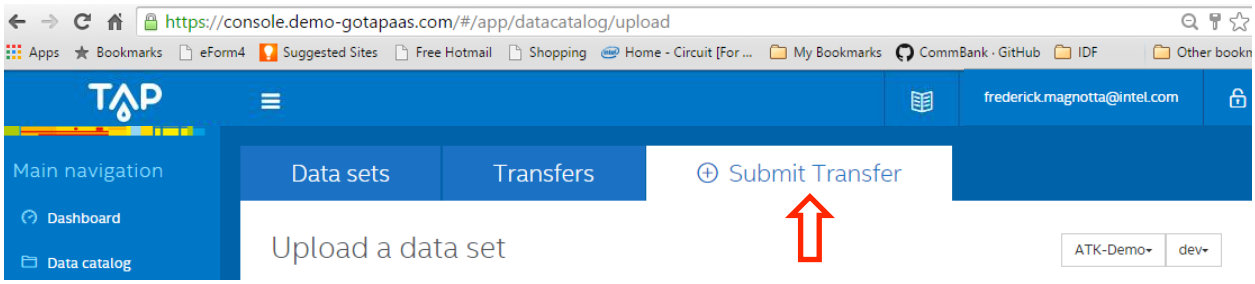


- Click on the **Transfers** tab to see what files have been uploaded to TAP and to get information on when a file was added to the system.

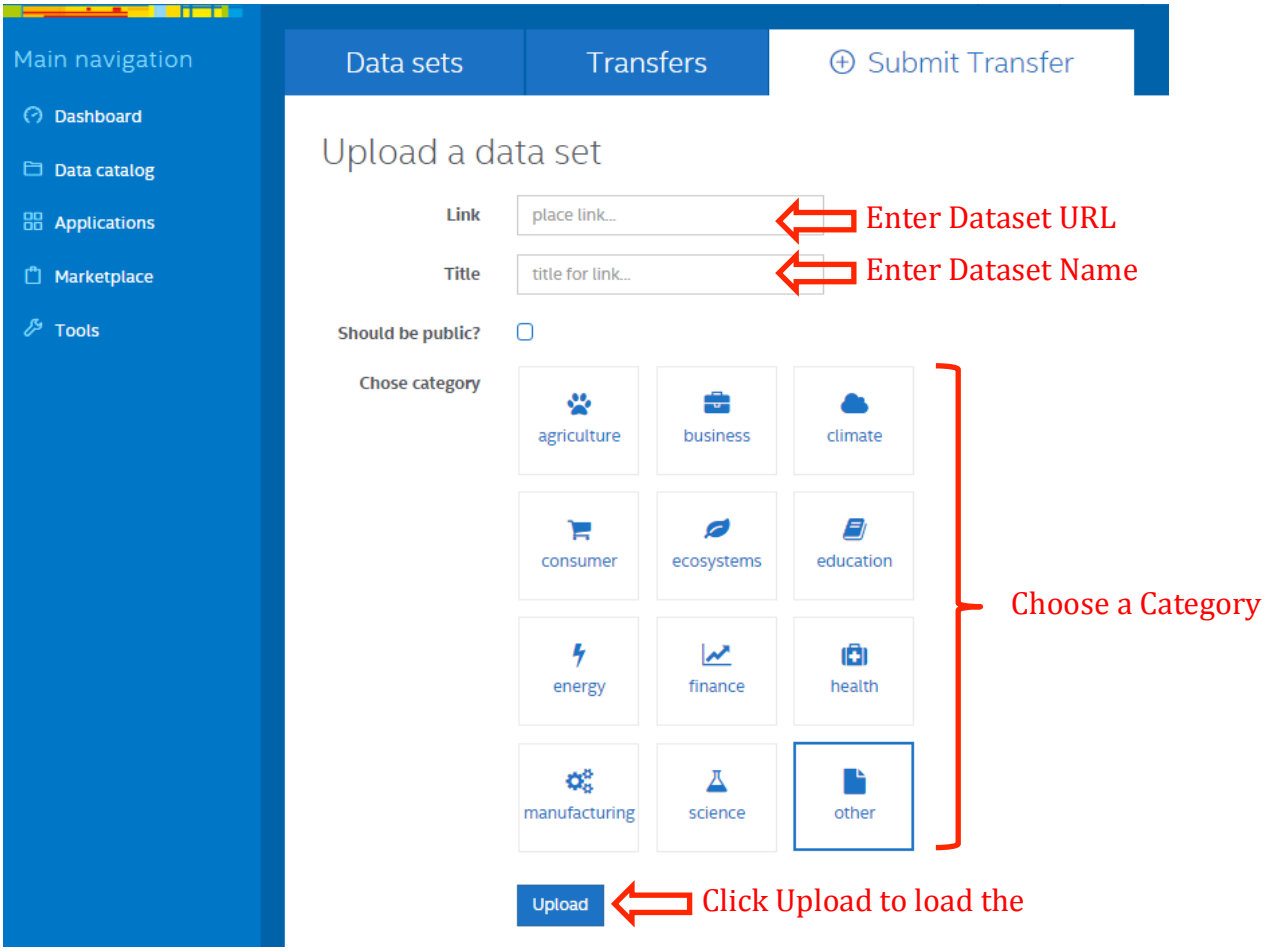


2.3. To add a file into TAP:

- Click the **Submit Transfer** tab.

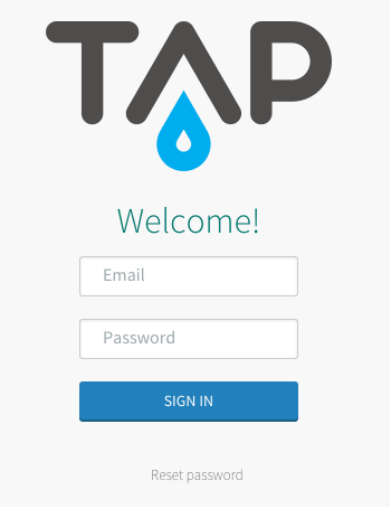


- Complete the Submit Transfer fields:
  - **Category** – Click on a category, based on your needs.
  - **Link field** – Enter the URL of the dataset you want to upload.
  - **Title field** – Enter a file name for your dataset.
- Click on Upload after you have entered the necessary information.



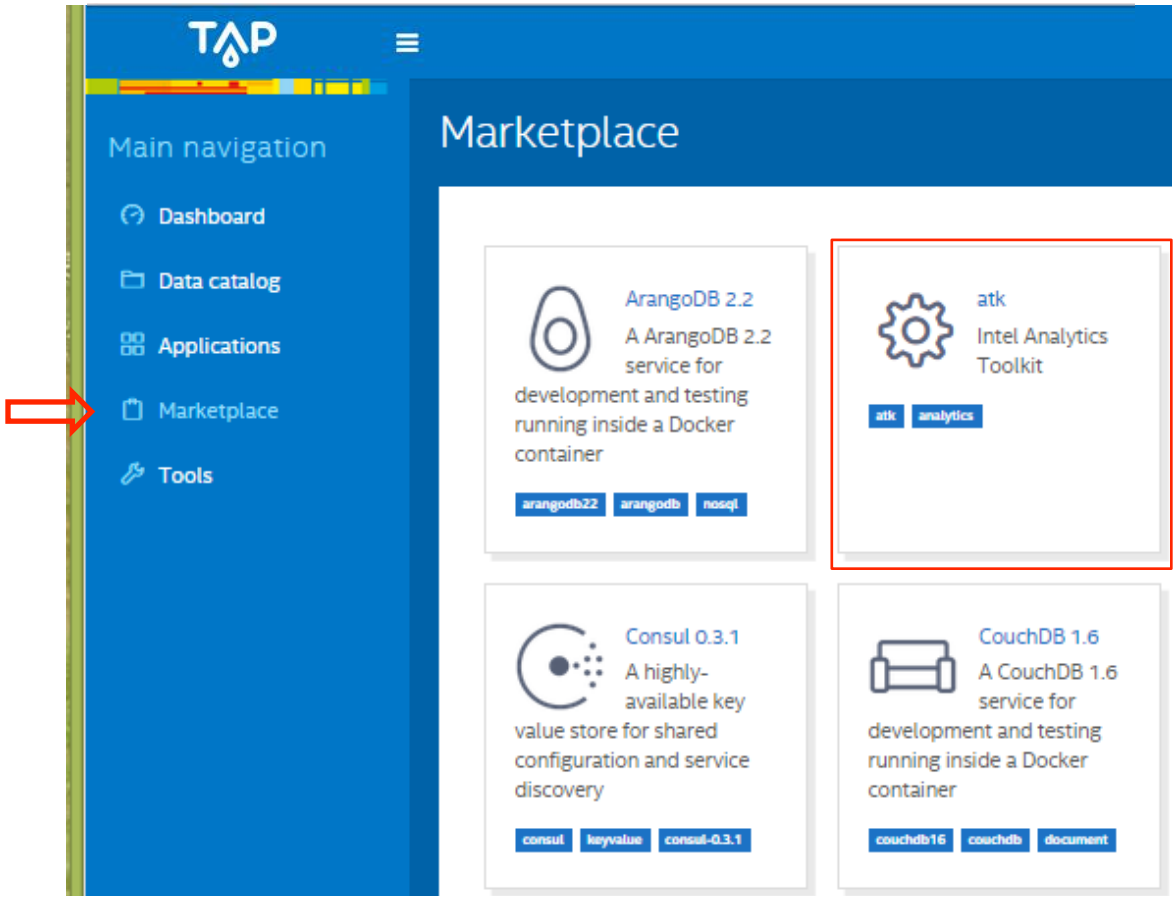
3. TAP Marketplace

- If you are not already logged into Marketplace, sign in with your username and password.

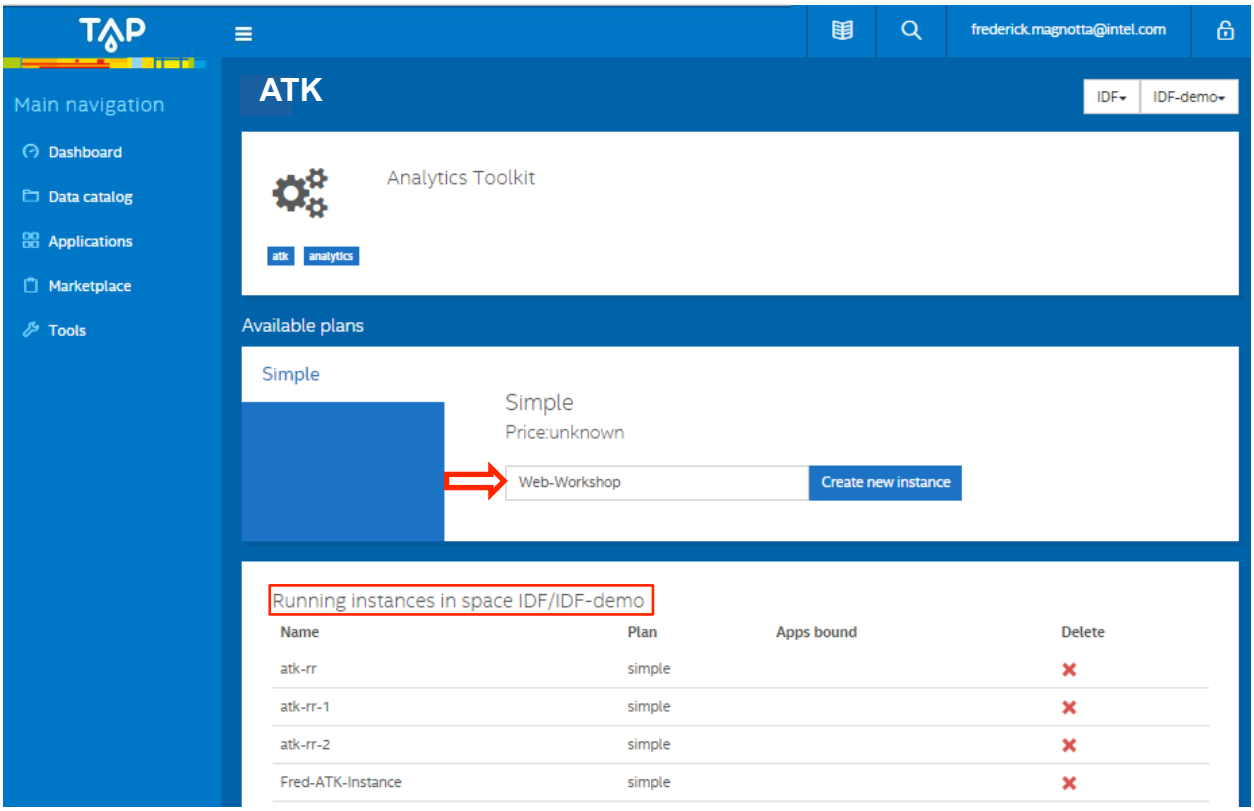


3.1. Create a new instance of ATK in TAP:

- Click on Marketplace under Main navigation.
- Click on ATK.

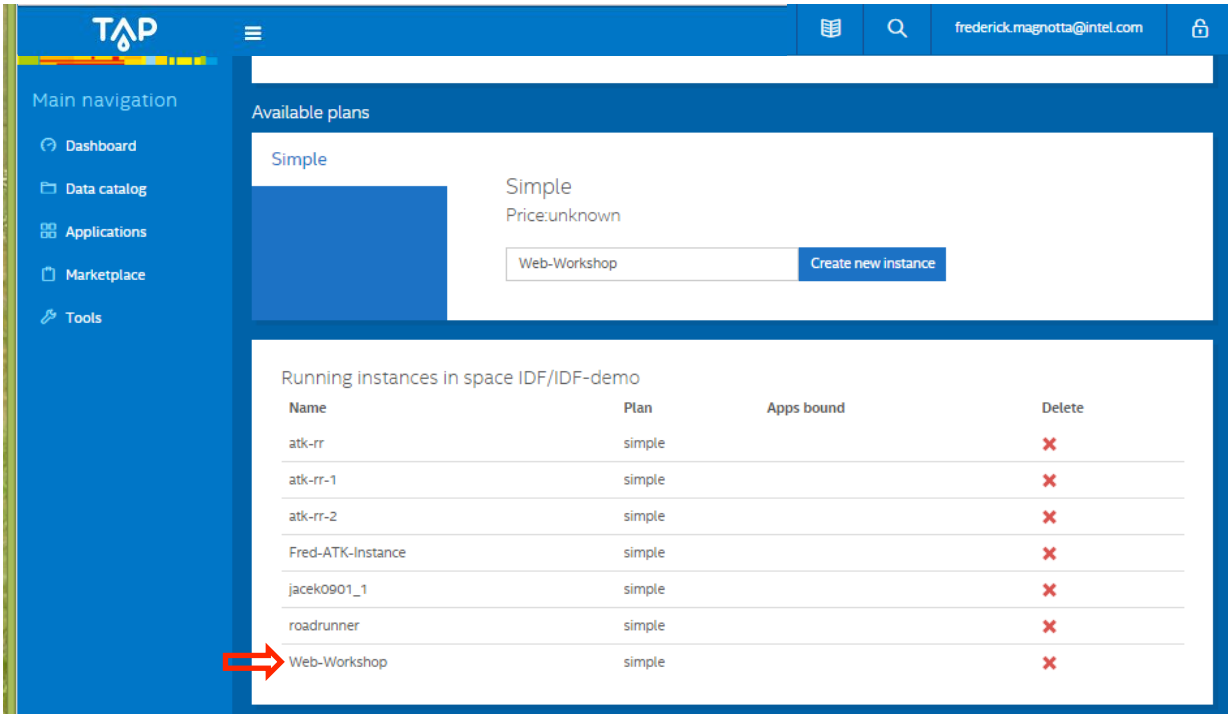


- In the ATK window, enter a name for your instance and click Create new instance. In this example, we created an instance called Web-Workshop.
- You can also delete existing ATK instances in this window.
- The ATK window displays all the running instances on TAP.



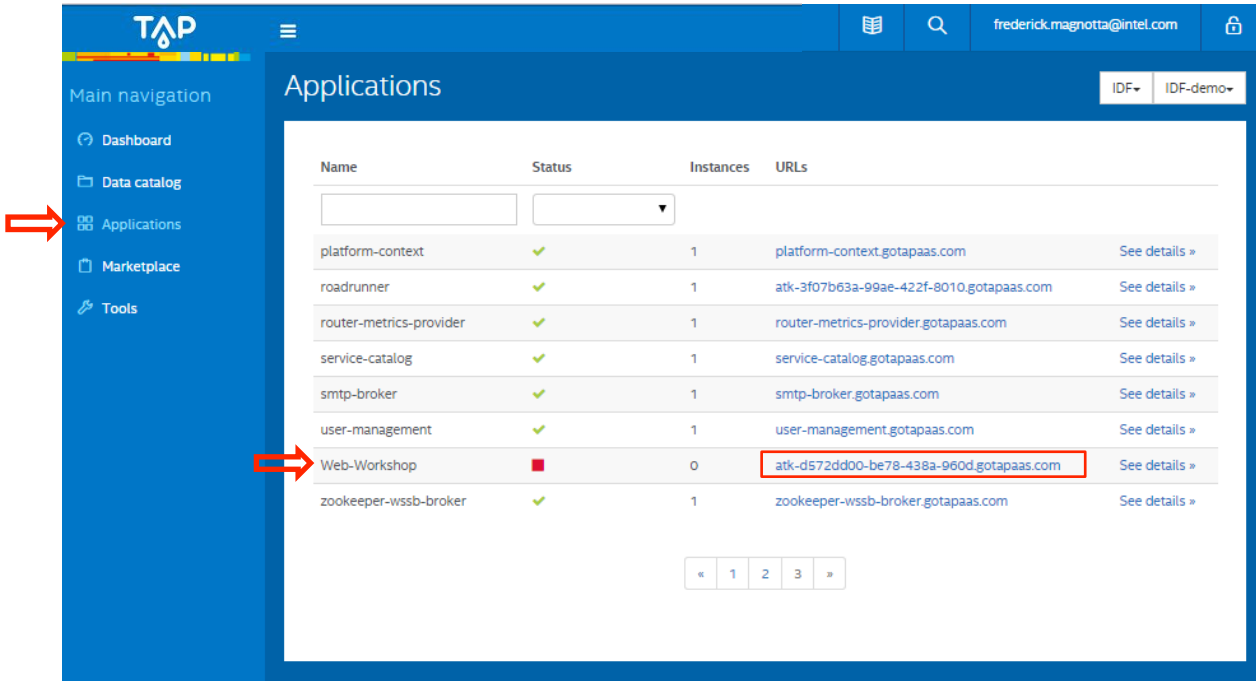
3.2. View the running instance:

- Refresh your window to see the instance you created.



3.3. Copy the instance uri to use in the Netflow demo:

- Click Applications under Main navigation to view the status of the new instance in the Applications window.
- Copy the uri of the instance. You will need the URL to run the Netflow demo in the next part of the workshop.
  - The red box indicates the instance is still initiating and not yet available. The red box is replaced by a green check when the instance is running.
  - Click on the instance's URL to see if the instance is up and running.



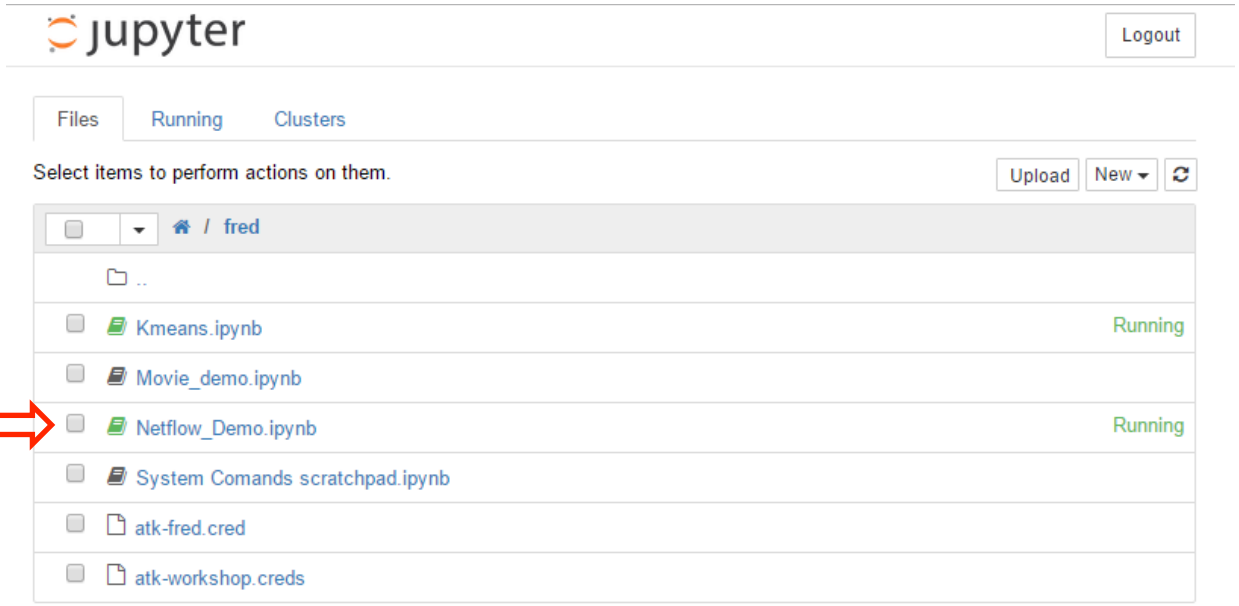
4. Netflow demo

4.1. What is the Netflow demo?

- The Netflow dataset contains information from web servers. The dataset contains information on tracked network traffic, which is incoming and outgoing internet traffic.
- The demo reads and trains the dataset. The demo then establishes a normal traffic pattern by tying the network edges and nodes together.
- It builds a graph showing all the connections based on the nodes' IP addresses.
- Finally, it produces a number of charts and graphs to show the network traffic flow and display outliers, which are outside the normal traffic pattern.

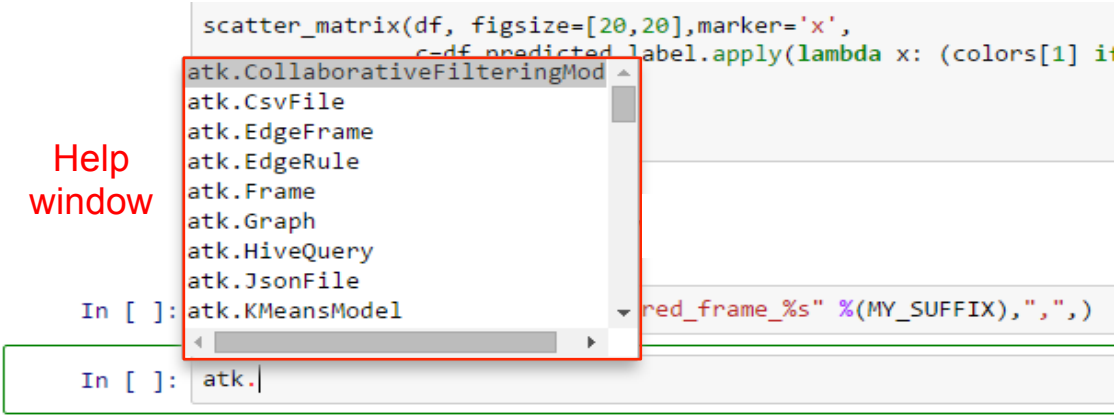
4.2. Open the Netflow demo file:

- Open the Netflow demo file in Python or iPython Notebook. iPython Notebook is also known as Jupyter.
- You can find the file at https://s3-us-west-2.amazonaws.com/analytics-tool-kit/public/datasets/latest/nf-hour.csv.

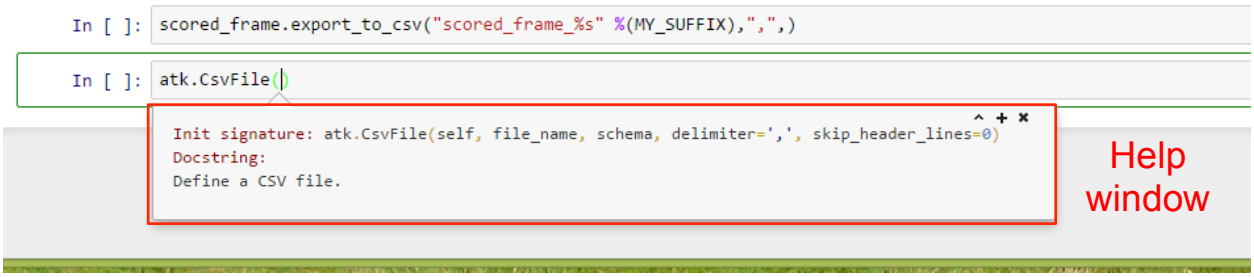


4.3. Review the help functions:

- If you need help knowing what commands are available:
  - Type the command into the In field; for example, "**ATK.**"
  - Then press the **tab key** to see the available command options.

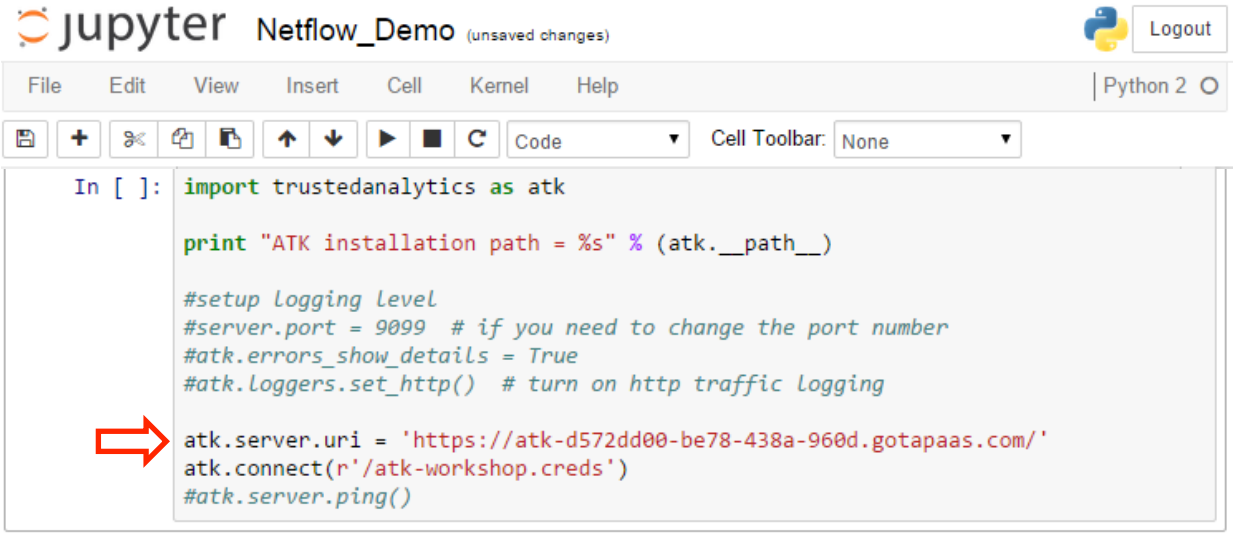


- If you need help knowing what the default parameters are for a command:
  - Type the command into the In field. For example, **"ATK.CsvFile( )".**
  - Then press the **shift** and **tab keys** to see the default parameters for that command.



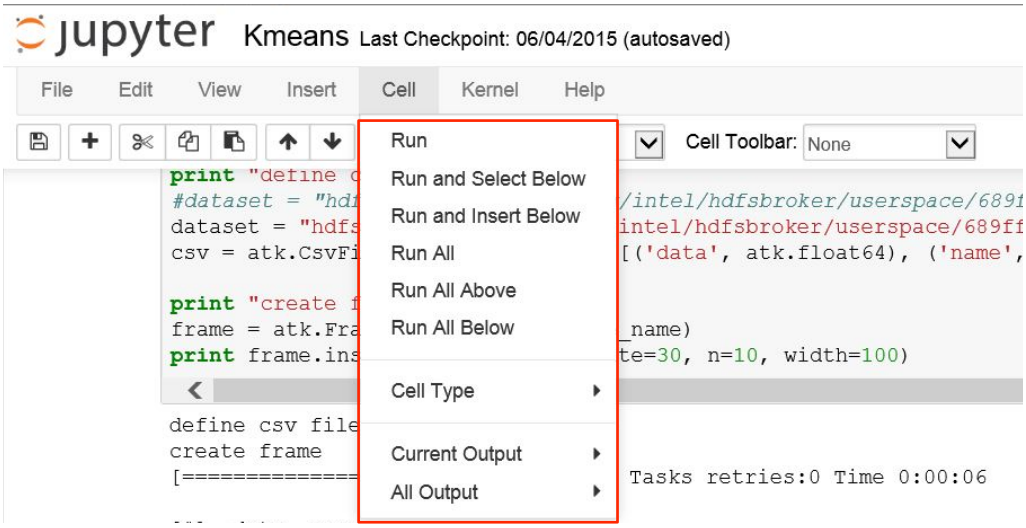
4.4. Edit the Netflow demo file:

- Change the ATK.server.uri field in the demo script to access the ATK instance you created in TAP.
- Copy the ATK.server.uri you created in TAP (Step 3.3).
- Paste the uri of the instance into the ATK.server.uri field of the Netflow demo script.
- **Important**: You must remove the “http://” at the beginning of the uri and the “/” at the end, before you enter the uri information.
- You should already have created a credentials file, when you installed Python or iPython.



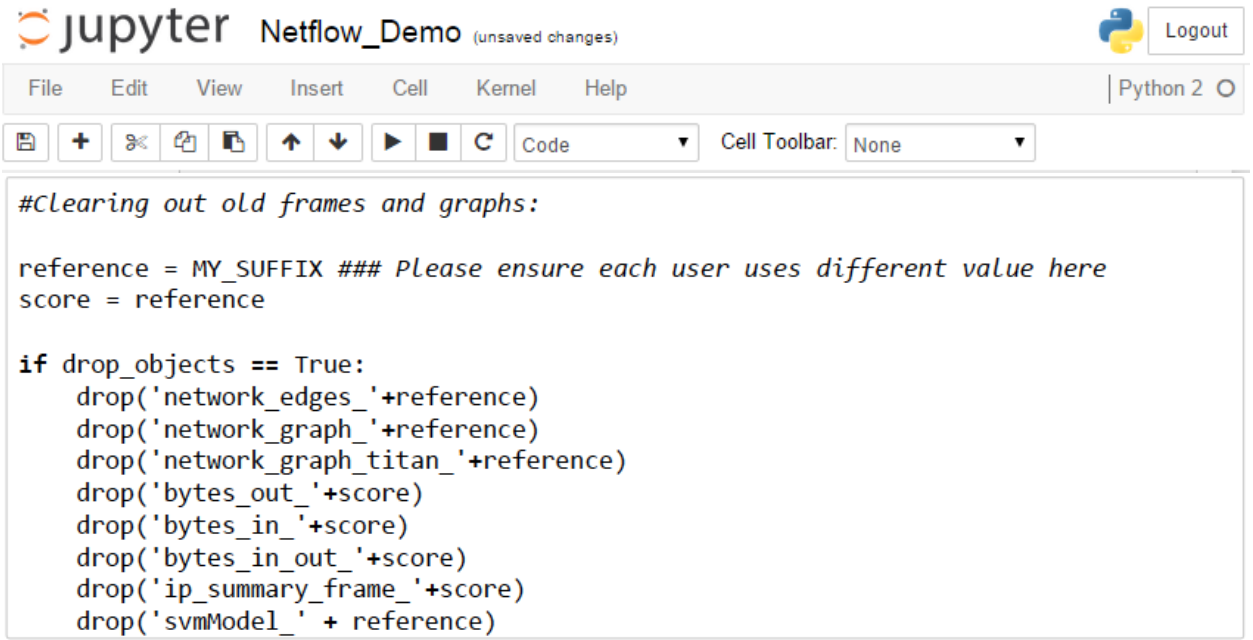
5. Deploy the ATK Model

- Execute the Netflow demo, which deploys the ATK.
- To begin, click Cell and select Run All. This runs all the scripts in the demo.
- If you need to rerun a single task, choose Run to execute a specific script in the demo.
- Or, if you need to run a group of scripts, choose Run All Below.



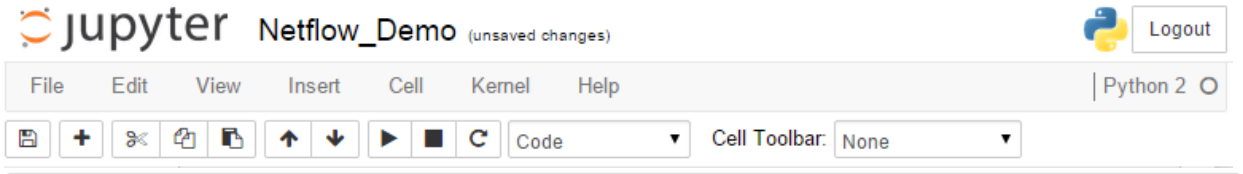
5.1. Clear out the old frames and graphs:

- The first step in the script cleans up the objects to ensure the code can be rerun multiple times.
- This script resets all the counters to zero.



5.2. Read in the data:

- This step reads in the data file that was loaded onto the ATK server.
- The Netflow dataset includes: date and time information, node IP addresses, protocols, bytes in and out, packets in and out …
- Confirm that the script completed 100% of the tasks and review the log for error messages. You may see several progress bars when multiple tasks run in the same section.




```
dataset = "hdfs://nameservice1/org/intel/hdfsbroker/userspace/28ff2403-eb1d-4ace
-b1d4-624c1d9f1f4f/6635acc0-71ff-40d5-9a2c-788dc3564349/000000_1"
week2_nf_schema=[('TimeSeconds', atk.float64),
                 ('tstart', str),
                 ('dateTimeStr', atk.float64),
                 ('protocol', str),
                 ('proto', str),
                 ('sip', str),
                 ('dip', str),
                 ('sport', atk.int64),
                 ('dport', atk.int64),
                 ('flag', atk.int64),
                 ('fwd', atk.int64),
                 ('tdur', atk.int64),
                 ('firstSeenSrcPayloadBytes', atk.int64),
                 ('firstSeenDestPayloadBytes', atk.int64),
                 ('ibyt', atk.int64),
                 ('obyt', atk.int64),
                 ('ipkt', atk.int64),
                 ('opkt', atk.int64),
                 ('recordForceOut', atk.int64)]

real_netflow = atk.Frame(atk.CsvFile(dataset, week2_nf_schema,  skip_header_line
s=1))
if drop_objects == True:
    drop('real_netflow_'+reference)
real_netflow.name='real_netflow_'+reference

real_netflow.inspect(wrap=10, round=4, width=100)
```

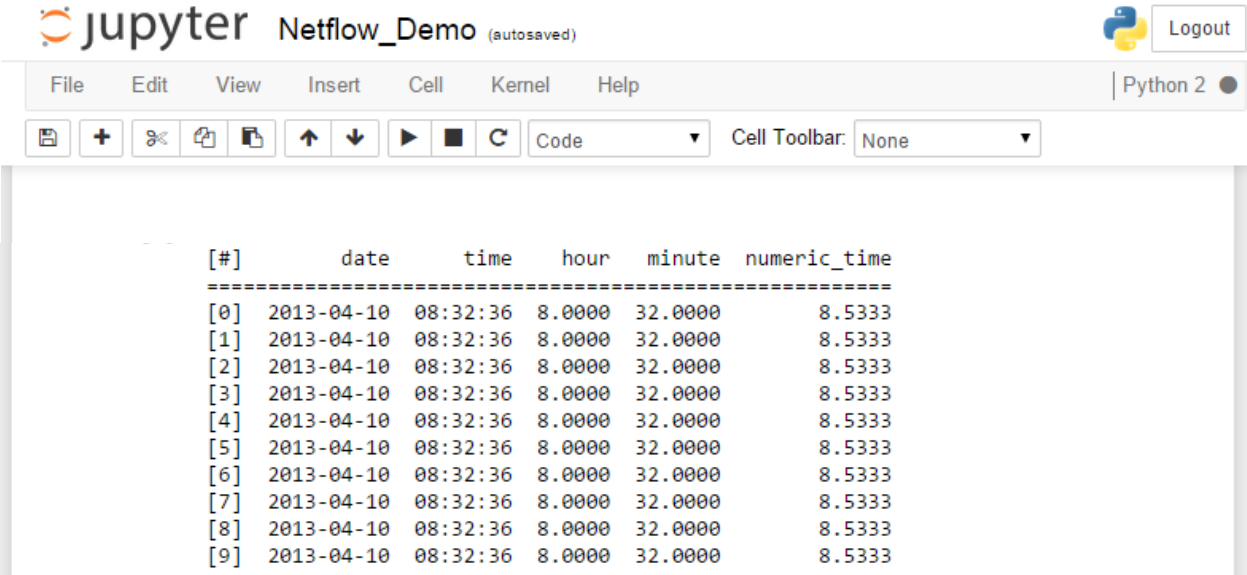
```
[==========================] 100.00% Tasks retries:0 Time 0:06:22
[==========================] 100.00%  Time 0:00:00
```

```
[#]      TimeSeconds              tstart         dateTimeStr  protocol  proto
sip
================================================================================
=============
[0]  1365582756.3800  2013-04-10 08:32:36  20130410083236.3828  17        UDP
172.20.2.19
[1]  1365582756.3800  2013-04-10 08:32:36  20130410083236.3828  17        UDP
172.20.2.18
[2]  1365582756.3800  2013-04-10 08:32:36  20130410083236.3828  17        UDP
172.20.2.17
[3]  1365582756.3800  2013-04-10 08:32:36  20130410083236.3828  17        UDP
```

5.3. Format and print the date/time data in columns:

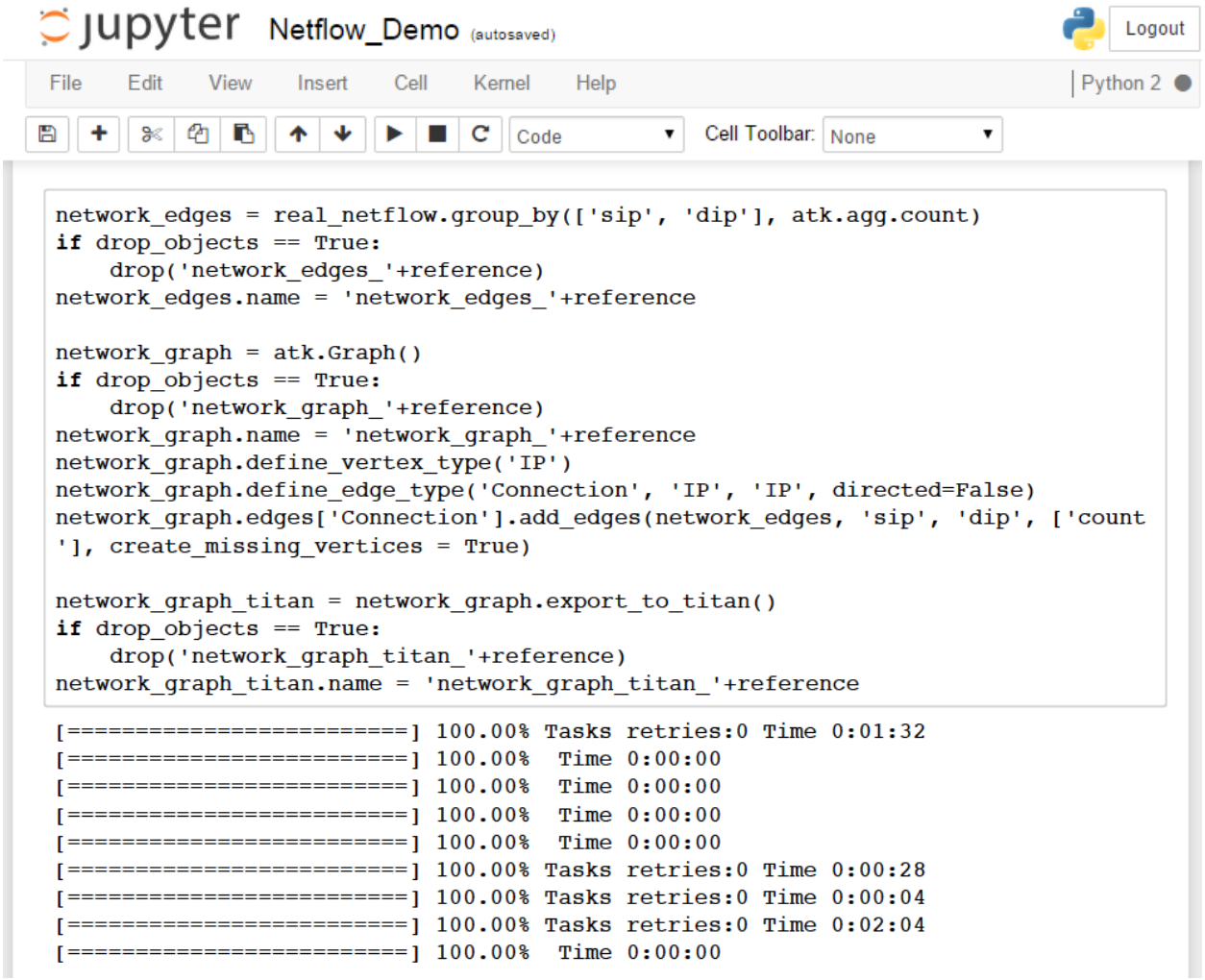
- This script formats the data into columns for date, time …

- The output from the date/time format script should look like this:



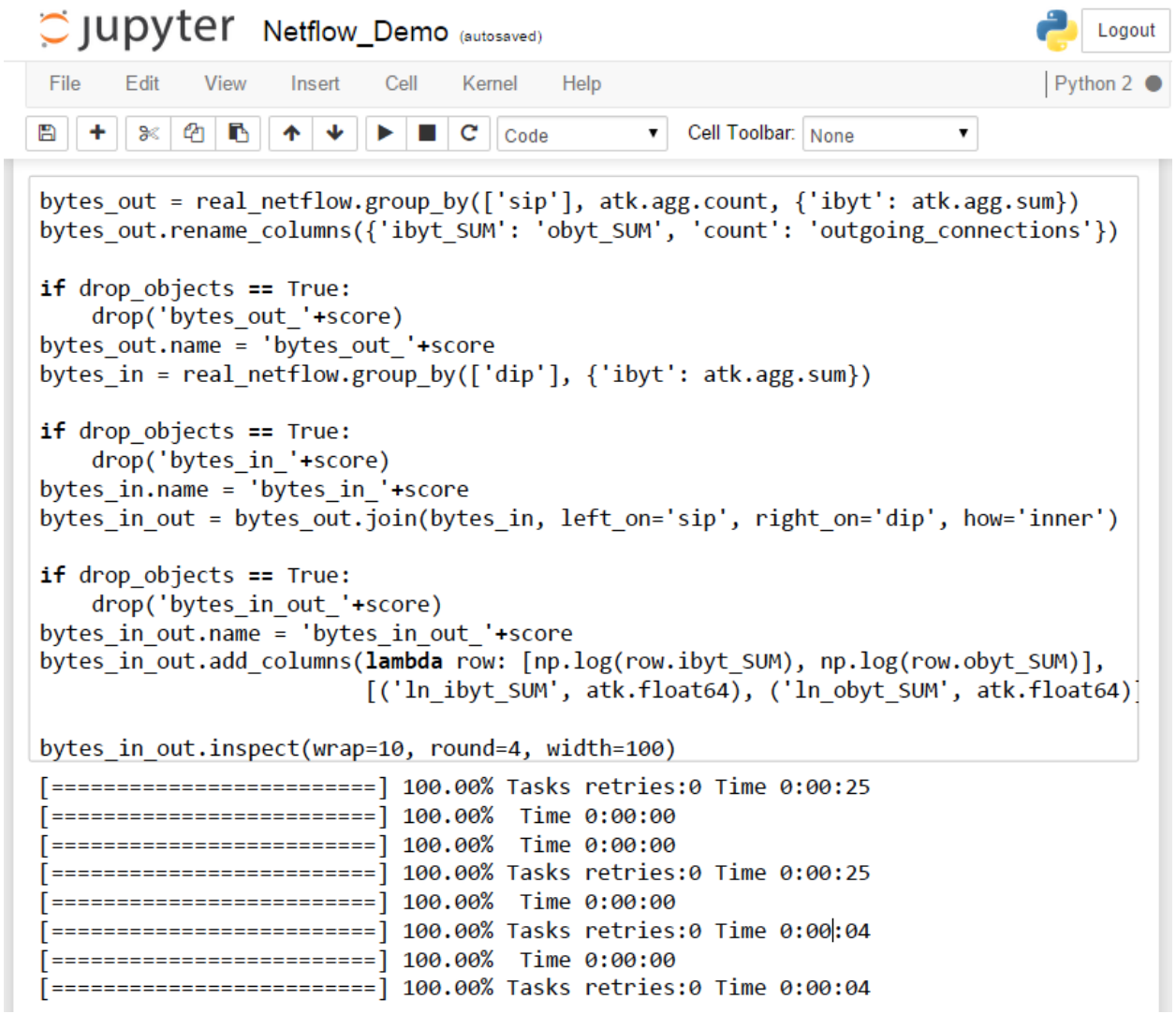
5.4. Build a Titan graph for computing statistics:

- This is the longest running script in the workshop. It may take four to five minutes to complete.
- Titan is a scalable graph database optimized for storing and querying graphs containing hundreds of billions of vertices and edges distributed across a multi-machine cluster.



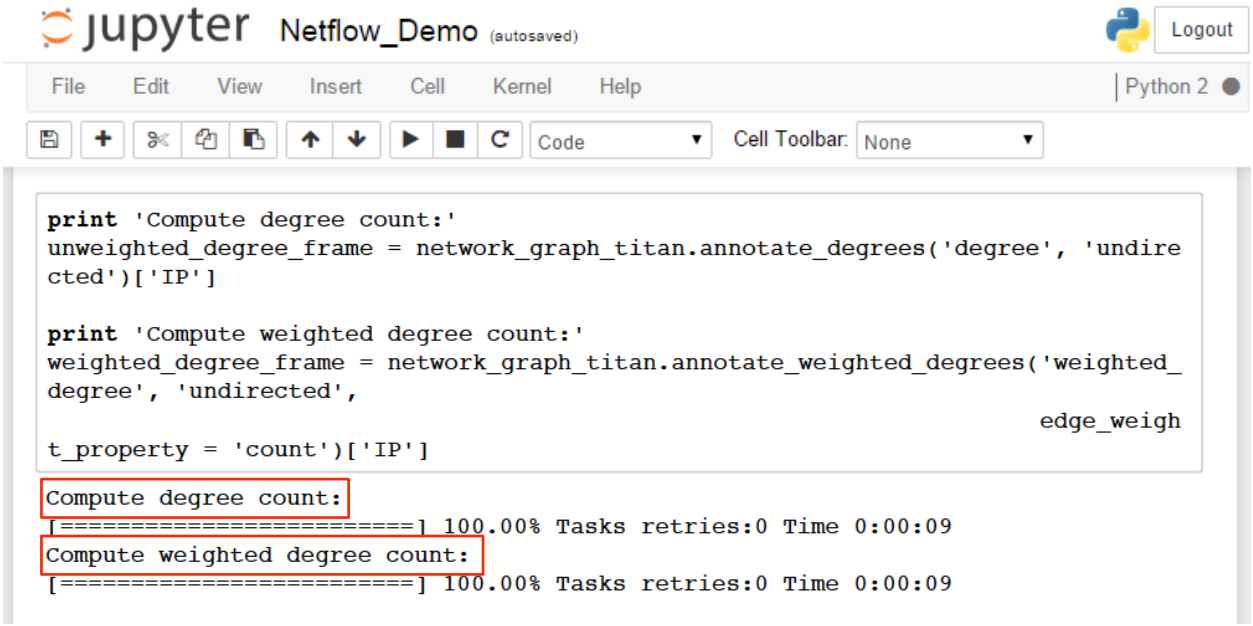
5.5. Computes bytes in and bytes out:

- This script computes the number of bytes in and bytes out for each node and creates total I/O counts for the nodes.



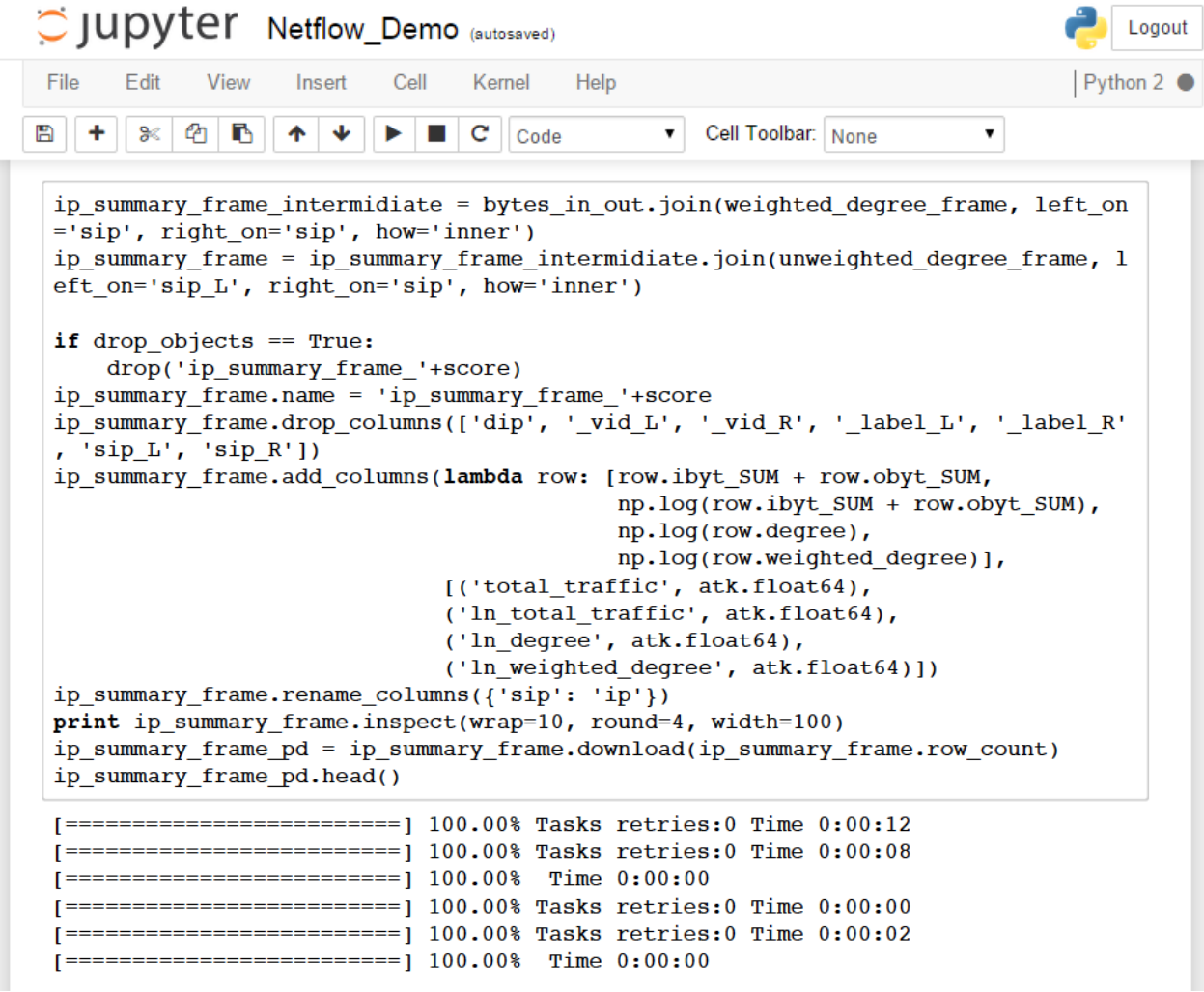
5.6. Compute weighted and unweighted degree counts:

- The degree of a node is the number of relations (edges) it has.
- Relations (edges) are borders between 2 IP nodes. If a node has a high degree, it means it has a lot of nodes as neighbors.
- The weighted degree of a node is like the degree. It's based on the number of edges for a node but is influenced by the weight of each edge.
- In addition to calculating the degree counts, this script identifies the 2 tasks it runs using print statements, e.g., print 'Compute degree count:' and print 'Compute weighted degree count:'. Notice the printout located just above each of the Tasks %.



5.7. Compute summary frame and download to Pandas:

- A data frame is a table, or two-dimensional array-like structure, in which each column contains measurements on one variable and each row contains one case. A summary frame is the sum of each row or column.
- This script prints the summary frame results in a tabular format.
- Additionally, this script downloads the data to your local machine. This allows you to execute commands and scripts against the sample data set.




```
ip_summary_frame_intermidiate = bytes_in_out.join(weighted_degree_frame, left_on
='sip', right_on='sip', how='inner')
ip_summary_frame = ip_summary_frame_intermidiate.join(unweighted_degree_frame, l
eft_on='sip_L', right_on='sip', how='inner')

if drop_objects == True:
    drop('ip_summary_frame_'+score)
ip_summary_frame.name = 'ip_summary_frame_'+score
ip_summary_frame.drop_columns(['dip', '_vid_L', '_vid_R', '_label_L', '_label_R'
, 'sip_L', 'sip_R'])
ip_summary_frame.add_columns(lambda row: [row.ibyt_SUM + row.obyt_SUM,
                                          np.log(row.ibyt_SUM + row.obyt_SUM),
                                          np.log(row.degree),
                                          np.log(row.weighted_degree)],
                             [('total_traffic', atk.float64),
                             ('ln_total_traffic', atk.float64),
                             ('ln_degree', atk.float64),
                             ('ln_weighted_degree', atk.float64)])
ip_summary_frame.rename_columns({'sip': 'ip'})
print ip_summary_frame.inspect(wrap=10, round=4, width=100)
ip_summary_frame_pd = ip_summary_frame.download(ip_summary_frame.row_count)
ip_summary_frame_pd.head()
```

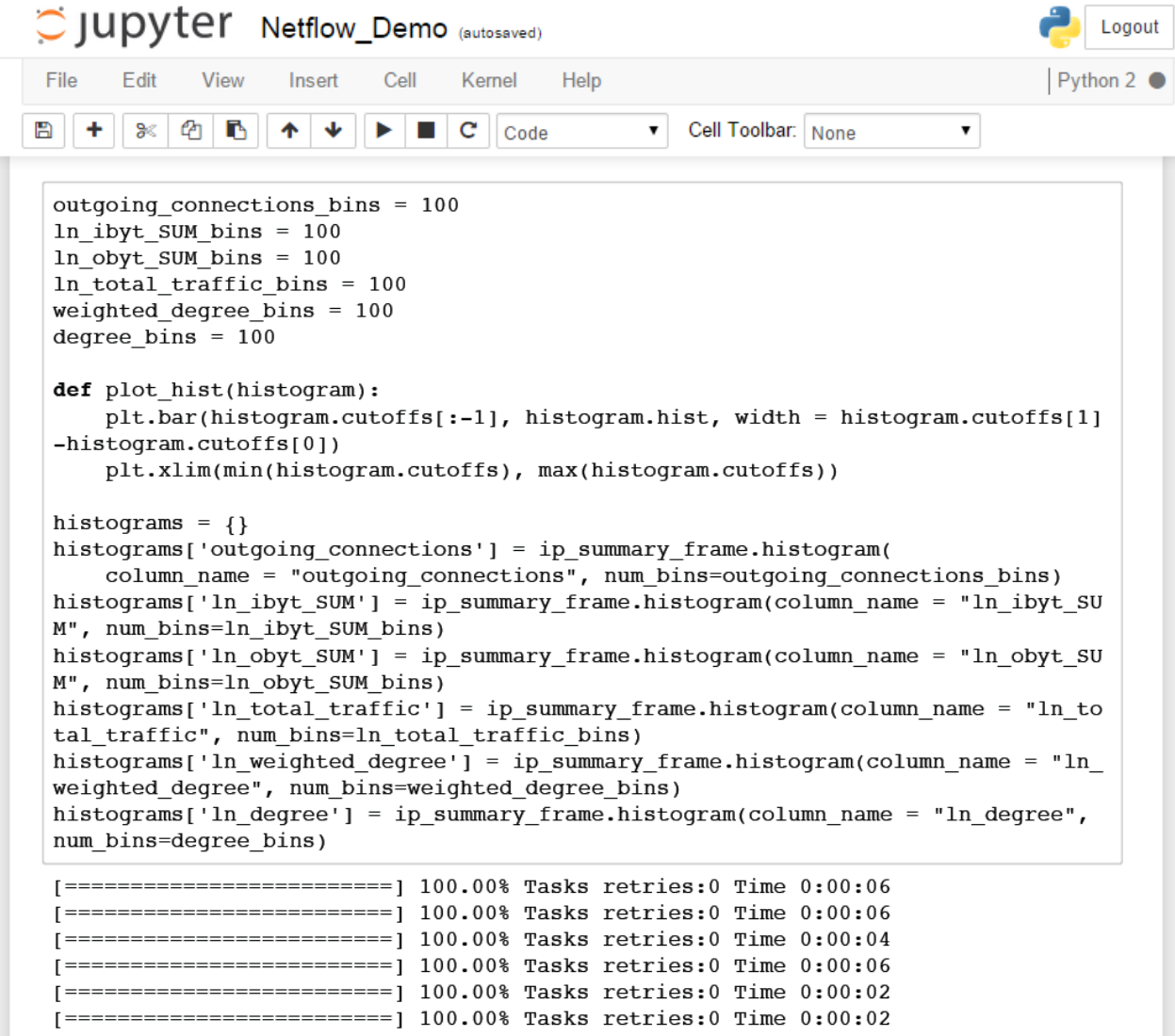
```
[=========================] 100.00% Tasks retries:0 Time 0:00:12
[=========================] 100.00% Tasks retries:0 Time 0:00:08
[=========================] 100.00%  Time 0:00:00
[=========================] 100.00% Tasks retries:0 Time 0:00:00
[=========================] 100.00% Tasks retries:0 Time 0:00:02
[=========================] 100.00%  Time 0:00:00
```

- This is only a sample of the summary frame table.

	outgoing_connections	obyt_SUM	ibyt_SUM	ln_ibyt_SUM	ln_obyt_SUM	weighted_
0	1176	17332354	22267710	16.918648	16.668085	1206
1	842	4819025	8652932	15.973409	15.388082	860
2	1105	9816363	23508230	16.972861	16.099561	1138
3	712	1578007	4087014	15.223325	14.271673	734
4	722	2469206	11834152	16.286500	14.719407	745

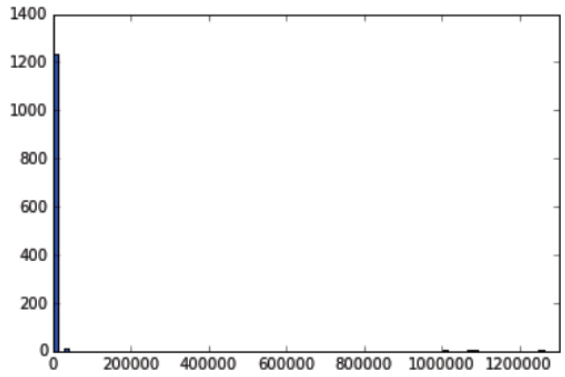
5.8. Compute and plot histogram:

- A histogram is a visual interpretation of numerical data indicating the number of data points that lie within a range of values, called a class or a bin. The frequency of the data in each class is depicted by the use of a bar.
- This script separates the data into six groups—outgoing connections, in byte sum, out byte sum, total traffic, weighted degree and degree.
- It displays the results in 6 histograms, seen on the next page.

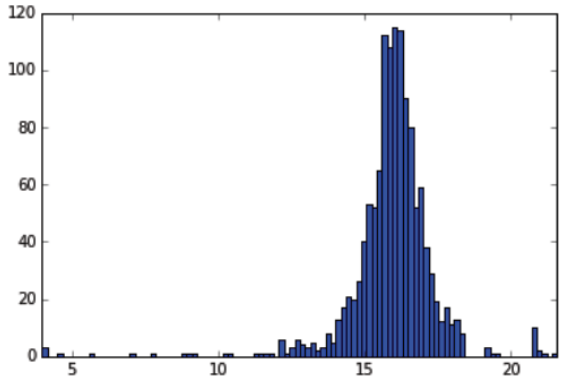


- Six histograms:

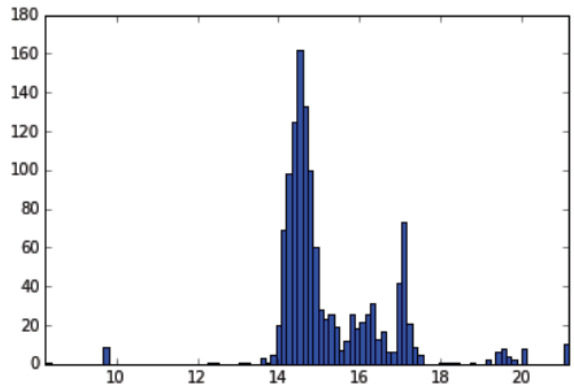
```
plot_hist(histograms['outgoing_connections'])
```



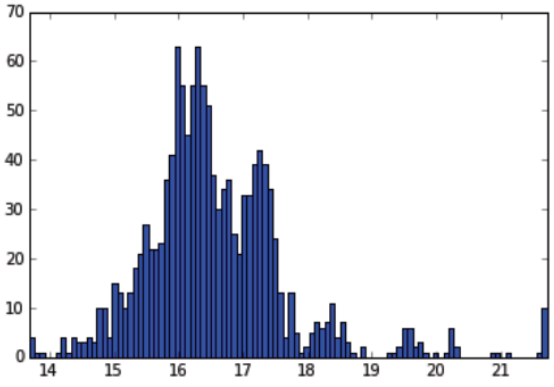

```
plot_hist(histograms['ln_ibyt_SUM'])
```



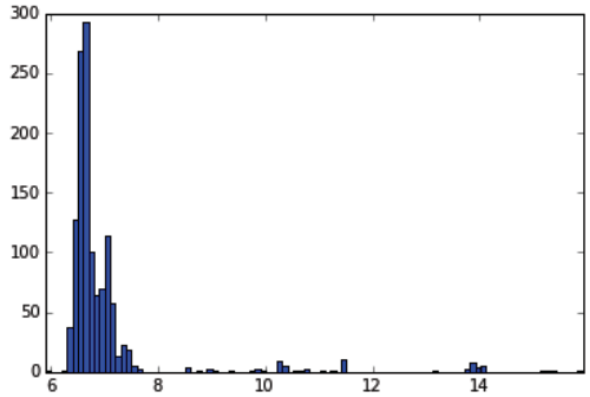

```
plot_hist(histograms['ln_obyt_SUM'])
```



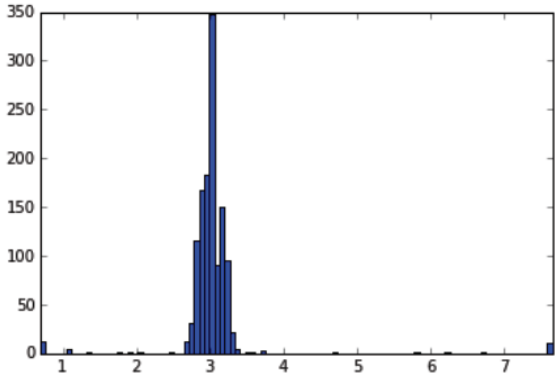

```
plot_hist(histograms['ln_total_traffic'])
```




```
plot_hist(histograms['ln_weighted_degree'])
```

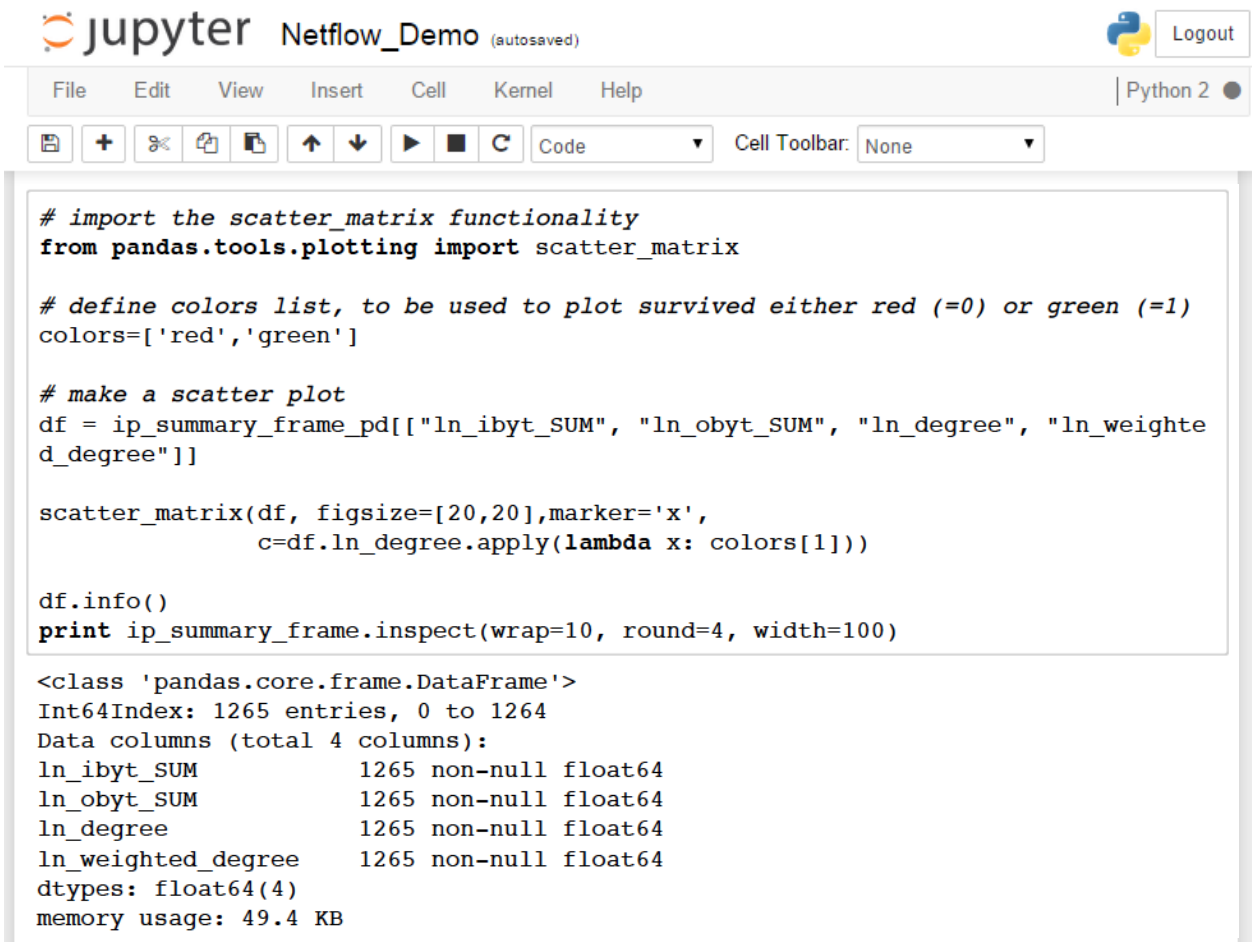



```
plot_hist(histograms['ln_degree'])
```



5.9. Create scatter plots:

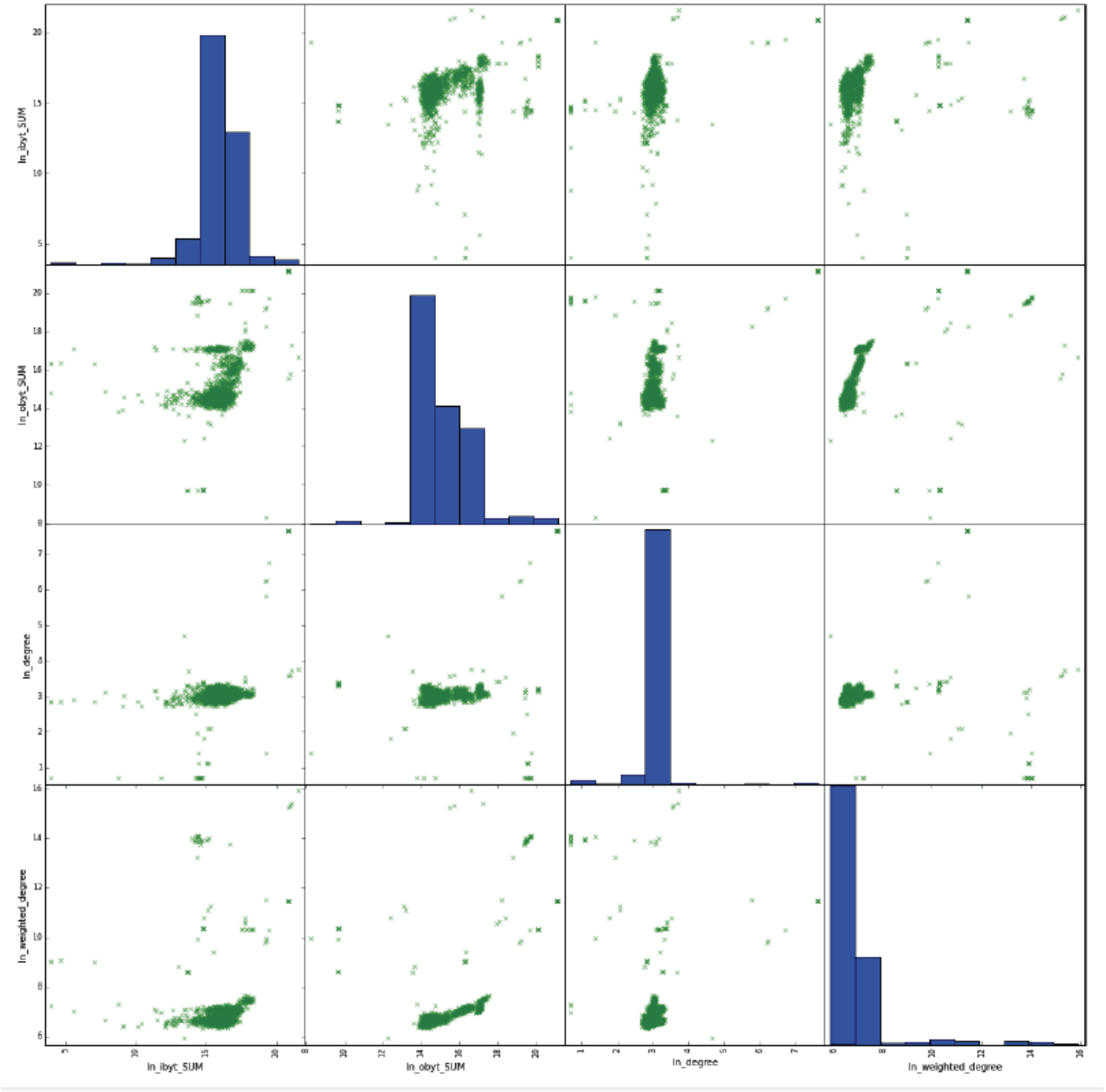
- A scatter (XY) plot has points that show the relationship between 2 sets of data. The relationship between two variables is called a correlation.
- This script uses 4 groups of data—in byte sum, out byte sum, degree, and weighted degree.
- It creates a four-by-four scatter-plot matrix.




```
# import the scatter_matrix functionality
from pandas.tools.plotting import scatter_matrix

# define colors list, to be used to plot survived either red (=0) or green (=1)
colors=['red','green']

# make a scatter plot
df = ip_summary_frame_pd[["ln_ibyt_SUM", "ln_obyt_SUM", "ln_degree", "ln_weighte
d_degree"]]

scatter_matrix(df, figsize=[20,20],marker='x',
               c=df.ln_degree.apply(lambda x: colors[1]))

df.info()
print ip_summary_frame.inspect(wrap=10, round=4, width=100)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1265 entries, 0 to 1264
Data columns (total 4 columns):
ln_ibyt_SUM           1265 non-null float64
ln_obyt_SUM           1265 non-null float64
ln_degree             1265 non-null float64
ln_weighted_degree    1265 non-null float64
dtypes: float64(4)
memory usage: 49.4 KB
```

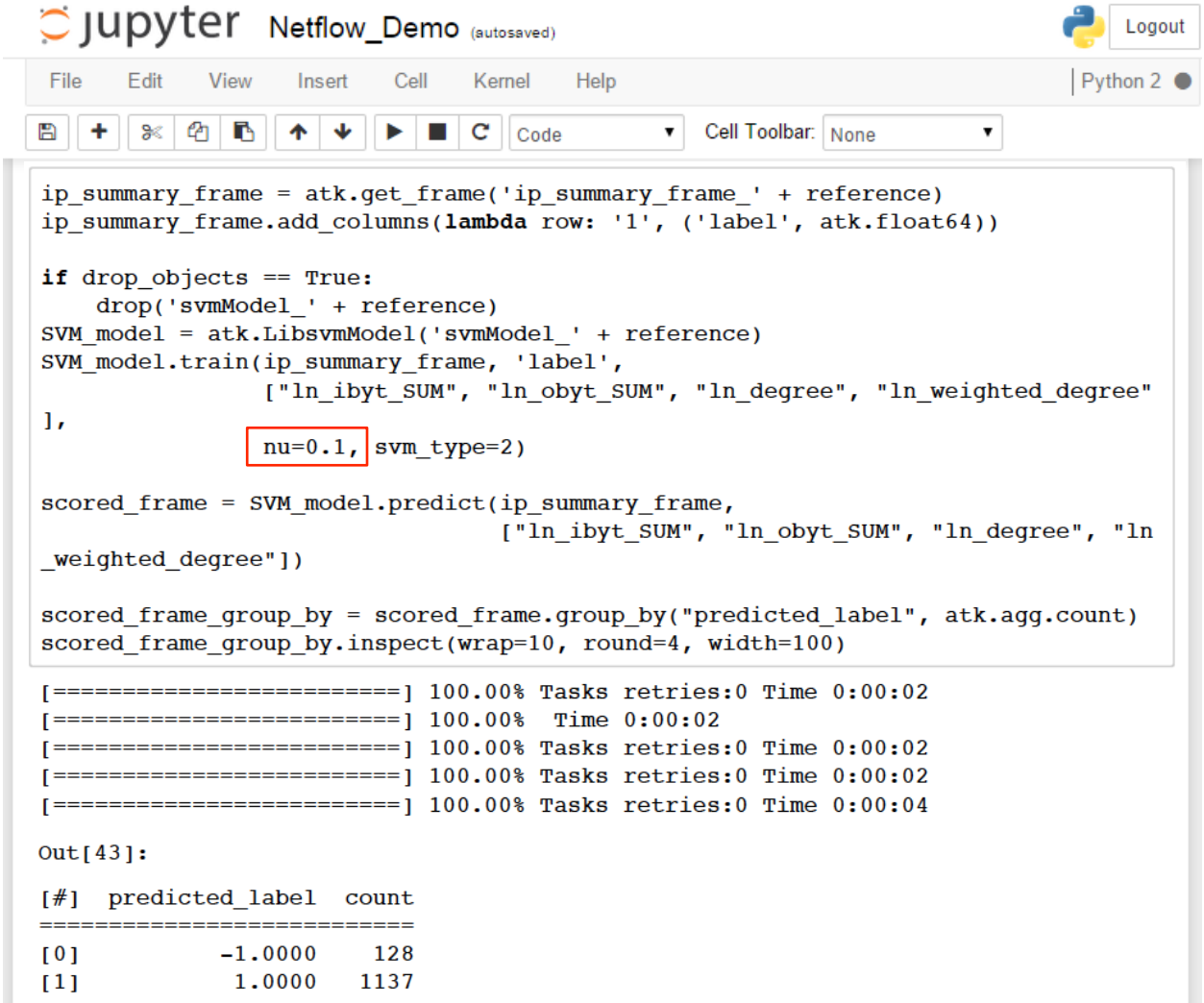
- Scatter-plot matrix:



- The scatter-plot matrix is replicated in Module 2: Visualizing Your Data with an App.

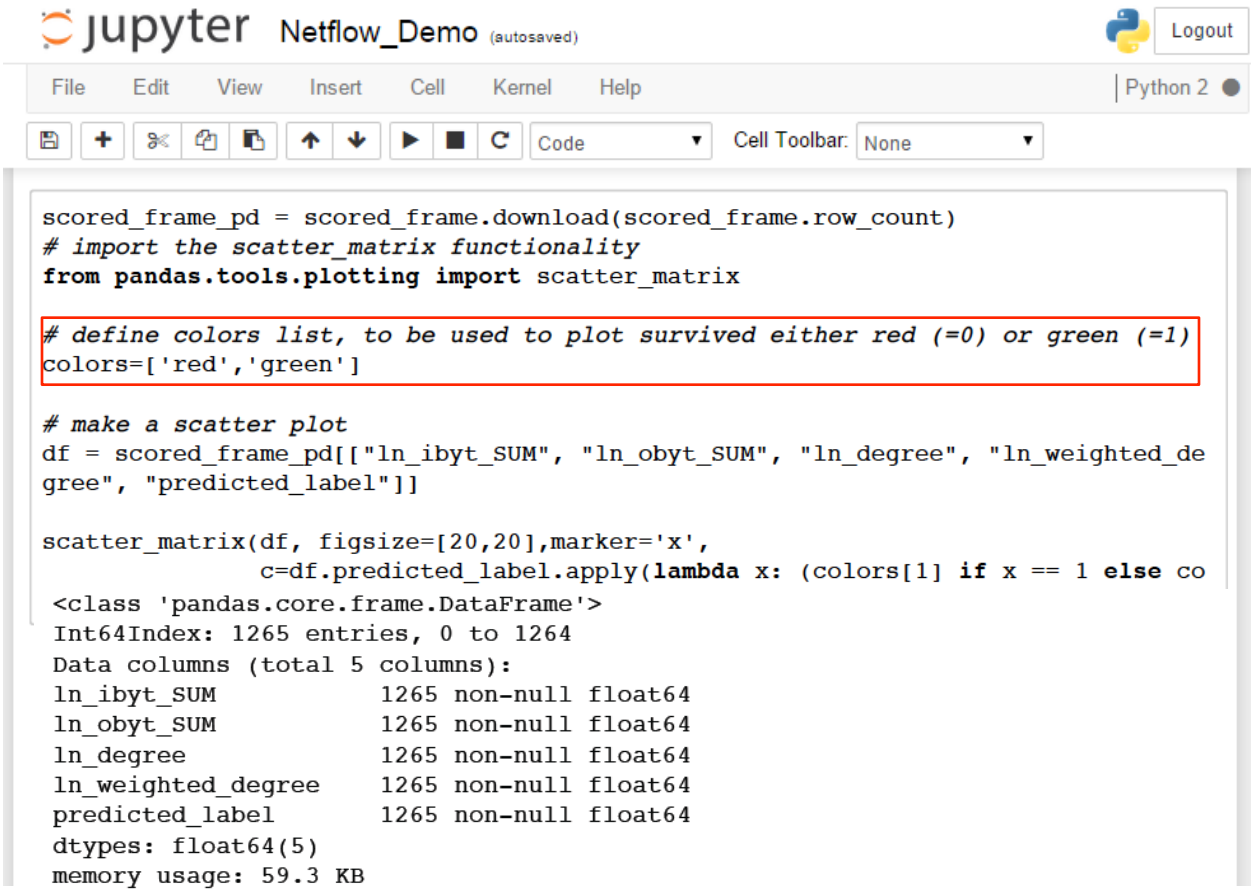
5.10. Create and train an SVM model:

- The script calculates the normal distribution of network traffic and the outliers. It detects anomalies on the network, such as DOS attach.
- The **nu** value determines the precision of our calculation. The value is the reciprocal of the dispersive power of a medium, known as constringency.
- After completing the remaining workshop steps, we encourage you to change the nu value in this script to see how it affects the data.
- Changing the nu value:
  - You **must** rerun step **5.7 "Compute summary frame and download to Pandas"** each time you change the nu value.
  - Then run this step and the last step in the workshop: **5.11 "Download the scatter frame to Pandas."**
  - Change the nu value to 0.2 or higher and notice how the graphs change.
  - Locate the upper limit of the nu value.



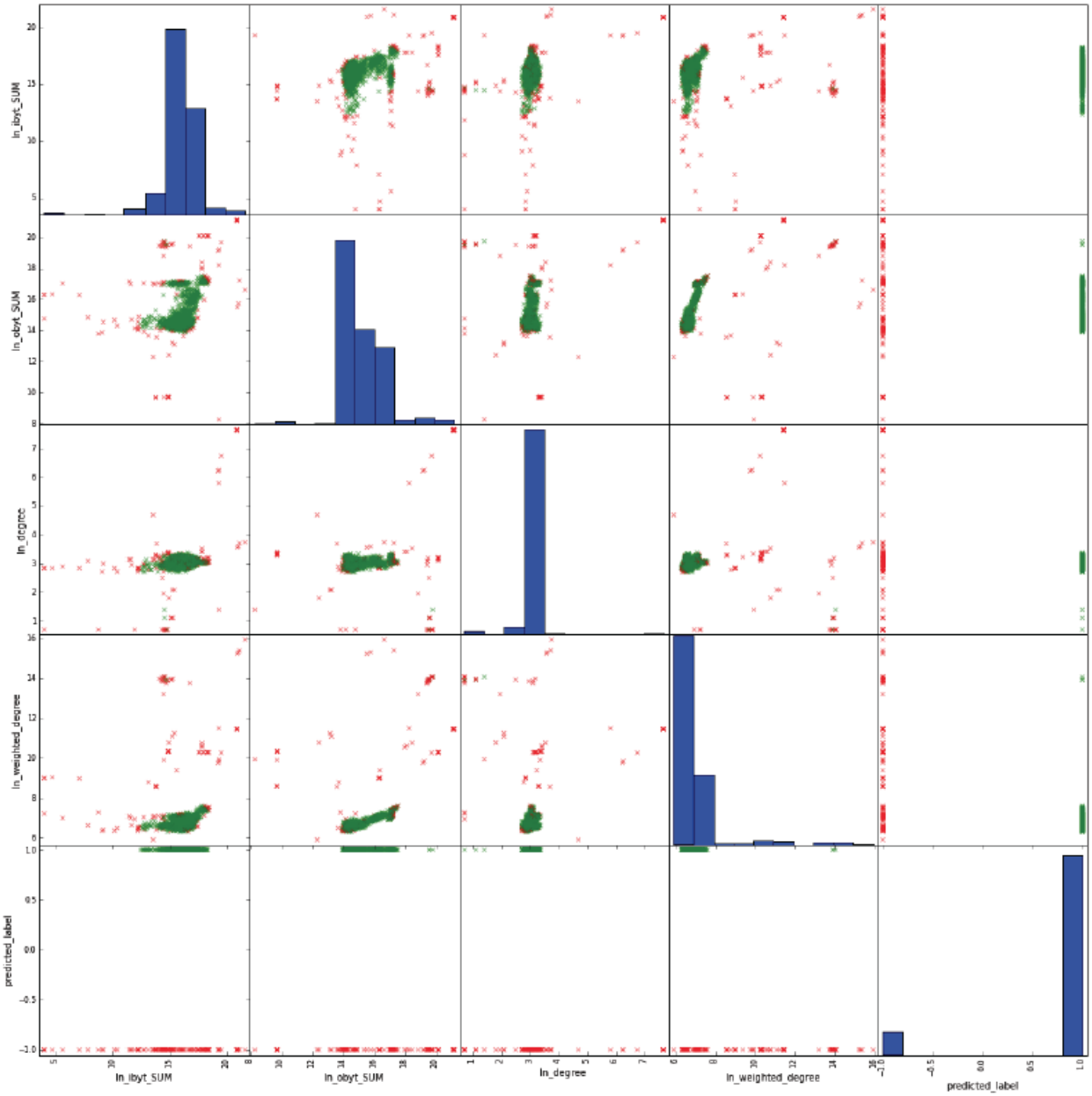
5.11. Download the scatter frame to Pandas:

- This script creates plot graphs from the results calculated in the “Create and train an SVM model” script in the previous step.
- The areas in green show the normal distribution. The points in red show the outliers.
- The points that are away from the “green cloud” in each frame should be investigated, as they fall outside the normal traffic patterns.
- To change the graph’s colors, modify the *# define colors list* in the script.



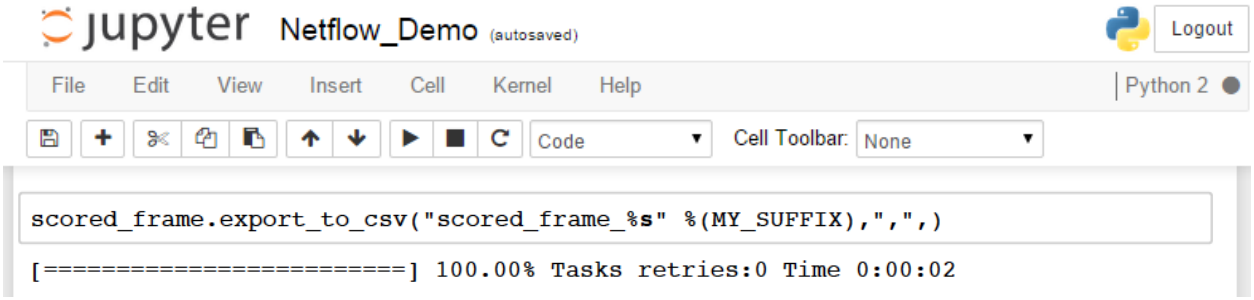
- See the scatter-plot graphs on the next page.

- Scatter-plot graphs:



6. Export the data to HDFS as a CSV file

- This script exports the data to CSV files, e.g., Microsoft Excel.



CONGRATULATIONS!

You have successfully completed the workshop.