By the end of this activity, you will be able to:

1. Read streaming data into Spark
2. Create and apply computations over a sliding window of data

Step 1. **Open Jupyter Python Notebook for Spark Streaming.** Open a web browser by clicking on the web browser icon at the top of the toolbar:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/RCneZE7PEeaqTxIkdCEfsw_c491f272226b35805e44abef7a7a22a9_browser-icon.png?expiry=1576108800000&hmac=LxE5WF1f27_DiO3b09IplBs1c9rWUs165BjzNgIfBkc

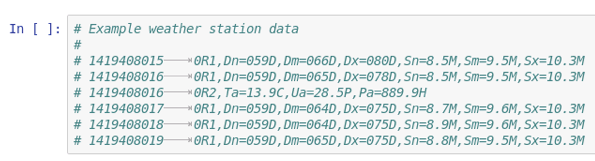
Navigate to *localhost:8889/tree/Downloads/big-data-3/spark-streaming*:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/Nt1zVlIHEeaWRg7WBfqodw_92380f2ed68f64b711346b8457d4db9f_browser.png?expiry=1576108800000&hmac=mSXuMwJrgVAWd0PFKAC1wjyDcbSmYUCYeQaXya2XAok

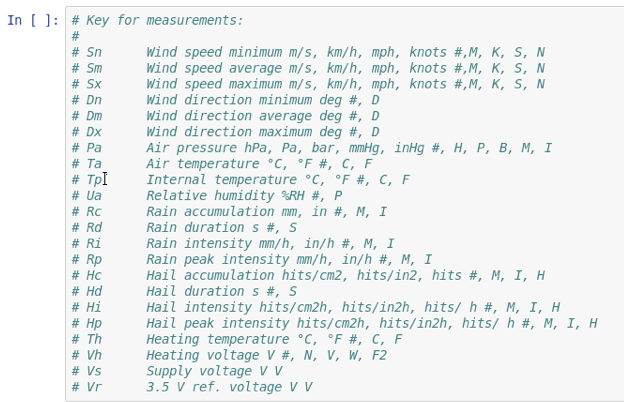
Open the Spark Streaming Notebook by clicking on *Spark-Streaming.ipynb:*

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/Pv8NsVIHEeaqTxIkdCEfsw_005004e59c9dd34f7bbcb4d92b508244_notebook.png?expiry=1576108800000&hmac=OP0J6L5dt9SOcl5XBqwQDepOx_l0jSNKywtRmgzfIrE

Step 2. **Look at sensor format and measurement types**. The first cell in the notebook gives an example of the streaming measurements coming from the weather station:



Each line contains a timestamp and a set of measurements. Each measurement has an abbreviation, and for this exercise, we are interested in the average wind direction, which is *Dm.* The next cell lists the abbreviations used for each type of measurement:



The third cell defines a function that parses each line and returns the average wind direction (*Dm*). Run this cell:



Step 3. **Import and create streaming context.** Next, we will import and create a new instance of Spark's StreamingContext:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/tCNNvlIHEeaX4QpLJOK7gQ_ad0d165c2121c43552275659b2a564e4_streaming-context.png?expiry=1576108800000&hmac=O5R8jhzngkpyoixYvimZf6Gpya3eAHo4IJ5C5pvTyyY

Similar to the SparkContext, the StreamingContext provides an interface to Spark's streaming capabilities. The argument *sc* is the SparkContext, and 1 specifies a batch interval of one second.

Step 4. **Create DStream of weather data**. Let's open a connection to the streaming weather data:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/bEQIp1IJEeaztg6Pg6w09w_617f5231b495159a5ed4cf09045e0abb_lines.png?expiry=1576108800000&hmac=1JgbMMHzaYRBdFT5X_TWgvaVMDKVe0-yx04LHRuDuiA

Instead of 12028, you may find that port 12020 works instead. This create a new variable *lines* to be a Spark DStream that streams the lines of output from the weather station.

Step 5. **Read measurement**. Next, let's read the average wind speed from each line and store it in a new DStream *vals:*

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/dsTv9VIJEeaqTxIkdCEfsw_ce64e4834676aee3ee6877ae8f635172_vals.png?expiry=1576108800000&hmac=rb3xHrqCJFKsQaCPX5kV3A6EoSaBHOux-TdK3jR7WgY

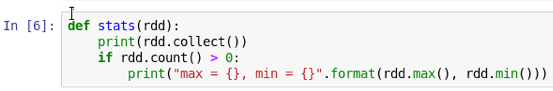
This line uses *flatMap()*to iterate over the *lines* DStream, and calls the *parse()*function we defined above to get the average wind speed.

Step 6. **Create sliding window of data**. We can create a sliding window over the measurements by calling the *window()* method:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/kT7kmFIJEeaWRg7WBfqodw_6f28cbf76b5b3a20bd52adb243c13649_window.png?expiry=1576108800000&hmac=oZ8hs_5J6yBtKRQcZgaHBg7QXpmUrNoqE2FoQO7ixOI

This create a new DStream called *window* that combines the ten seconds worth of data and moves by five seconds.

Step 7. **Define and call analysis function**. We would like to find the minimum and maximum values in our window. Let's define a function that prints these values for an RDD:



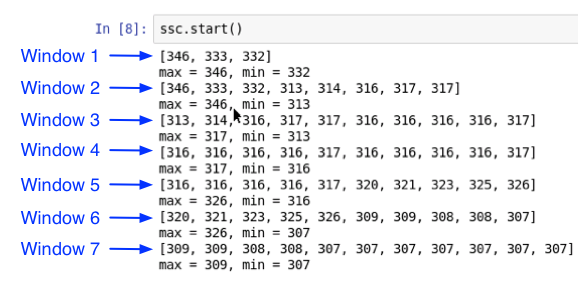
This function first prints the entire contents of the RDD by calling the *collect()*method. This is done to demonstrate the sliding window and would not be practical if the RDD was containing a large amount of data. Next, we check if the size of the RDD is greater than zero before printing the maximum and minimum values.

Next, we call the *stats()* function for each RDD in our sliding window:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/1S-CyFIKEeaqTxIkdCEfsw_26c0a5f92dc3737122f535fca531857d_window-stats.png?expiry=1576108800000&hmac=zKpDRNOdLOWBj6eiZ3td4LzFu2hp8tqDyYxPV8vaOVY

This line calls the *stats()* function defined above for each RDD in the DStream window.

Step 8. **Start the stream processing**. We call *start()* on the StreamingContext to begin the processing:



The sliding window contains ten seconds worth of data and slides every five seconds. In the beginning, the number of values in the windows are increasing as the data accumulates, and after Window 3, the size stays (approximately) the same. Since the window slides half as often as the size of the window, the second half of a window becomes the first half of the next window. For example, the second half of Window 5 is 310, 321, 323, 325, 326, which becomes the first half of Window 6.

When we are done, call *stop()* on the StreamingContext:

https://d3c33hcgiwev3.cloudfront.net/imageAssetProxy.v1/qNYOSlIKEeaqTxIkdCEfsw_840cc244c5958aa370c8d2c278bae7e2_ssc-stop.png?expiry=1576108800000&hmac=6yL9gEx1rzoUr_Ra8v5eXaaBcVYRaPo_J8YLZeGkoWs