**Boston University** Metropolitan College

# Assignment 5
# Big Data Analytics
MET CS777

Faculty - Farshid Alizadeh-Shabdiz, PhD, MBA

## 1 Description

In this assignment you will implement Batch Gradient Descent to fit a line into a two dimensional data set. You will implement a set of Spark jobs that will learn parameters of a line model using the New York City Taxi trip in the Year 2013 dataset. The dataset was released under the FOIL (The Freedom of Information Law) and made public by Chris Whong (https://chriswhong.com/open- data/foil nyc taxi/). See the Assignment 1 for details of the dataset.

In this assignment you will implement a linear model using travel distance in miles to predict the fare amount, the money that is paid to the taxis.

Please note that in this assignment you **CANNOT use Spark ML and MLlib** libraries.

## 2 Taxi Data Set - Same data set as Assignment 1

This is the same dataset as Assignment 1. Please refer to there for the data description.

The dataset is in Comma Separated Volume Format (CSV). When you read a line and split by comma sign (","), you will get a string array of length 17. Index number startes from zero. For the assignment, you need to get trip distance in miles (index 5) and fare amount in dollars (index 11) as stated in the following table.

| | | |
|---|---|---|
| index 5 (X-axis) | trip distance | trip distance in miles |
| index 11 (Y-axis) | fare amount | fare amount in dollars |

Table 1: Taxi Data Set fields

**Data Clean-up Step**

- Remove all taxi rides that are less than 2 mins or more than 1 hour.
- Remove all taxi rides that have "fare amount" less than 3 dollars or more than 200 dollars

- Remove all taxi rides that have ”trip distance” less than 1 mile or more than 50 miles
- Remove all taxi rides that have ”tolls amount” less than 3 dollars.

Note: Preprocess the data and store the results in your cluster storage.

# 3 Obtaining the Dataset

You can download or access the datasets using following internal URLs:

| Google Cloud | |
|---|---|
| Small Data Set | gs://metcs777-fa/taxi-data-sorted-small.csv.bz2 |
| Large Data Set | gs://metcs777-fa/taxi-data-sorted-large.csv.bz2 |

Table 2: Data set on Google Cloud Storage - URLs

Small dataset (93 MB compressed, uncompressed 384 MB) is for code development, implementation and testing, and it roughly consists of 2 million taxi trips.

# 4 Assignment Tasks

## 4.1 Task 1 : Simple Linear Regression (20 points)

Find a simple line regression model to predict ”fare amount” from the travel distance.

Consider a Simple Linear Regression model given in equation (1). The solutions for slope of the line, m, and y-intercept, b, are calculated in the equations (2) and (3).

$$Y = mX + b \tag{1}$$

$$\hat{m} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i{}^2 - (\sum_{i=1}^{n} x_i)^2} \tag{2}$$

$$\hat{b} = \frac{\sum_{i=1}^{n} x_i{}^2 \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i y_i}{n\sum_{i=1}^{n} x_i{}^2 - (\sum_{i=1}^{n} x_i)^2} \tag{3}$$

Implement a PySpark Job that calculates the exact answers for the parameters *m and b*.

Run your implementation on the large data set and report the computation time for your Spark Job for this task. You can get the execution time for the cloud services that you use.

Note on Task 1: Execution of this task on the large dataset depends on your implementation and can take a long time. For example, on a cluster with 12 cores in total, it takes more than 40 min computation time.

## 4.2 Task 2 - Find the Parameters using Gradient Descent (40 Points)

In this task, you should implement the batch gradient descent to find the optimal parameters for the Simple Linear Regression model.

- Load the data into spark cluster memory as RDD or Dataframe
- Start with all parameters initial values equal to 0.1
- Run for gradient descent for 100 iterations.

Cost function will be

$$L(m,b) = \sum_{i=1}^{n} [y_i - (mx_i + b)]^2$$

Partial Derivatives to update the parameters m and b are as follows:

$$\frac{\partial}{\partial m} = \frac{2}{N} \sum_{i=1}^{n} -x_i(y_i - (mx_i + b))$$

$$\frac{\partial}{\partial b} = \frac{2}{N} \sum_{i=1}^{n} -(y_i - (mx_i + b))$$

Here is a list of important setup parameters:

• Set your initial learning rate to be learningRate=0.0001 and change it if needed.

• You can implement the bold driver to improve your learning rate.

• Maximum number of iterations is 100, numIteration=100

Run your implementation on the large data set and report the computation time for your Spark Job for this task. Compare the computation time with the previous task.

• Print out the cost for each iteration
• Print out the model parameters, (m, b) in each iteration

Note: You might write some code for the gradient descent in PySpark that can work perfectly on your laptop but does not run on a cluster (AWS/Google Cloud). The main reason is that on your laptop it is running in a single process while on a cluster it runs on multiple processes (shared-nothing processes). You need to be careful to reduce all of jobs/processes to be able to update the variables. Otherwise, each processes will have its own variables.

## 4.3 Task 3 - Fit Multiple Linear Regression using Gradient Descent (40 Points)

We would like to find a linear model with 4 variables to predict total paid amounts of Taxi rides. The following table describes the variables that we want to use.

| | | |
|---|---|---|
| index 4 (1st independent variable) | trip_time_in_secs | duration of the trip |
| index 5 (2nd independent variable) | trip_distance | trip distance in miles |
| index 11 (3rd independent variable) | fare_amount | fare amount in dollars |
| index 12 (4th independent variable) | tolls_amount | bridge and tunnel tolls in dollars |
| index 16 (y-axis, dependent variable) | total_amount | total paid amount in dollars |

Table 4: Taxi Data Set fields

- Initialize all parameters to 0.1
- Set your learning rate to learningRate=0.001
- Maximum number of iterations to 100, numIteration=100
- Use Vectorizatoin for this task. We will not accept your solution when you write duplicated code. It should include vectorization.
- Implement "Bold Driver" technique to dynamically change the learning rate. (10 points of 40 points)

- Print out the costs in each iteration
- Print out the model parameters in each iteration

# 5 Important Considerations

## 5.1. Machines to Use

Be aware that you can choose virtually any configuration for your cluster - you can choose different number of machines, and different configurations of those machines. Note that each setting is going to cost you differently.

Pricing information is available at: http://aws.amazon.com/elasticmapreduce/pricing/

Since this is real money, it makes sense to develop your code and run your jobs locally on your laptop, using the small data set. Once things are working, then you'll move to a cluster.

Run your Spark jobs over the "large" data using 4 workers machines with 4 cores and 8GB RAM each.

- As you can see on the list price, costs per hour is not much, but IT WILL ADD UP QUICKLY IF YOU FORGET TO SHUT OFF YOUR MACHINES. Be very careful, and stop your machine as soon as you are done working. You can always come back and start your machine or create a new one easily when you begin your work again.
- Another thing to be aware of is that cloud services charge you when you move data around. To avoid such charges, do everything in the north east region, where the data is.
- You should document your code very well and as much as possible.

## Submission Format

Create a single document that has results for all three tasks. For each task, copy and paste the result that your last Spark job saved in the bucket. Also for each task and each Spark job include a screen shot of the Spark History.

Please zip your code and your document (use .zip only, please!), or else attach each piece of code as well as your document to your submission individually.

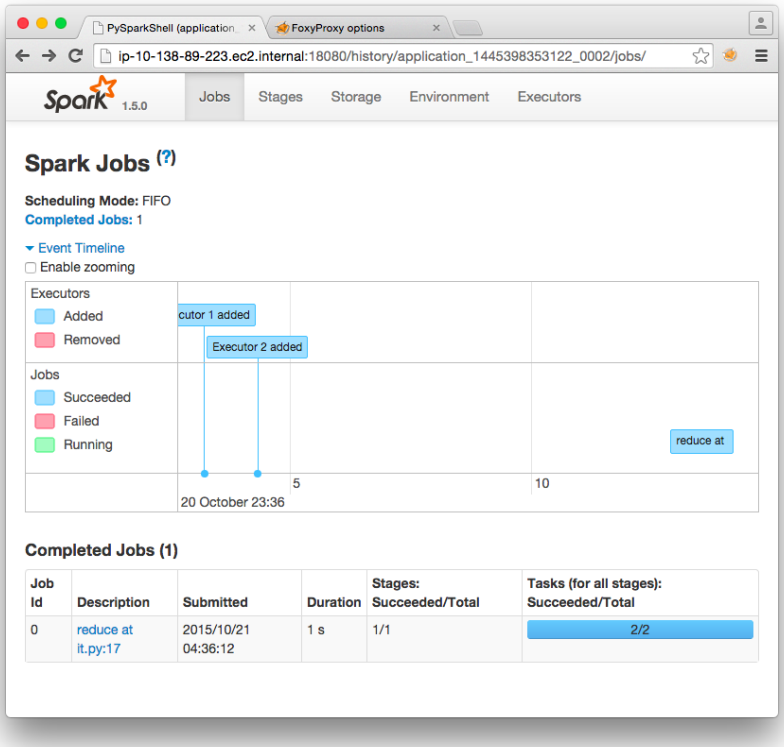


Figure 1: Screenshot of Spark History

## Academic Misconduct Regarding Programming

In a programming class like our class, there is sometimes a very fine line between ”cheating” and acceptable and beneficial interaction between peers. Thus, it is very important that you fully understand what is and what is not allowed in terms of collaboration with your classmates. We want to be 100% precise, so that there can be no confusion.

The rule on collaboration and communication with your classmates is very simple: you cannot transmit or receive code from or to anyone in the class in any way—visually (by showing someone your code), electronically (by emailing, posting, or otherwise sending someone your code), verbally (by reading code to someone) or in any other way we have not yet imagined. Any other collaboration is acceptable.

The rule on collaboration and communication with people who are not your classmates (or your TAs or instructor) is also very simple: it is not allowed in any way, period. This disallows (for example) posting any questions of any nature to programming forums such as StackOverflow. As far as going to the web and using Google, we will apply the ”two line rule”. Go to any web page you like and do any search that you like. But you cannot take more than two lines of code from an external resource and actually include it in your assignment in any form. Note that changing variable names or otherwise transforming or obfuscating code you found on the web does not render the ”two line rule” inapplicable. It is still a violation to obtain more than two lines of code from an external resource and turn it in, whatever you do to those two lines after you first obtain them.

Furthermore, you should cite your sources. Add a comment to your code that includes the URL(s) that you consulted when constructing your solution. This turns out to be very helpful when you’re looking at something you wrote a while ago and you need to remind yourself what you were thinking.