Assignment 4

MET CS 777 - Big Data Analytics Classification
- Logistic Regression with Gradient Descent
(20 points)

GitHub Classroom Invitation Link

https://classroom.github.com/a/MSGh5Ck5

1 Description

In this assignment, you will be implementing a regularized logistic regression to classify text documents.

You will be asked to perform three subtasks:

1. data preparation,
2. learning (which will be done via gradient descent) and
3. evaluation of the learned model.

2 Data

You will be dealing with a data set consisting of around 170,000 text documents (this is 7.6 million lines of text in all) and a test/evaluation data set consisting of 18,700 text documents (almost exactly one million lines of text in all).

Your task is to build a classifier that can automatically determine whether a text document is an Australian court case. We have prepared four data sets for your use.

1. The Training Data Set (1.9 GB of text - 520MB compressed). This is the set you will use to train your logistic regression model.
2. The Testing Data Set (200 MB of text, 57MB compressed). This is the set you will use to evaluate your model.
3. The Small Data Set (37.5 MB of text 10MB compressed). This is for you to use for training and testing your model locally before you try to do anything in the cloud.
4. The Small Testing Data Set to use when testing your model.

Public links

- https://storage.googleapis.com/met-cs-777-data/TestingData.txt.bz2
- https://storage.googleapis.com/met-cs-777-data/TrainingData.txt.bz2
- https://storage.googleapis.com/met-cs-777-data/SmallTestingData.txt.bz2
- https://storage.googleapis.com/met-cs-777-data/SmallTrainingData.txt.bz2

Google Cloud Storage

- gs://met-cs-777-data/TestingData.txt.bz2
- gs://met-cs-777-data/TrainingData.txt.bz2
- gs://met-cs-777-data/SmallTestingData.txt.bz2
- gs://met-cs-777-data/SmallTrainingData.txt.bz2

Data Details: You should download and look at the SmallTrainingData.txt.bz2 file before beginning. You'll see that the contents are sort of a pseudo-XML, where each text document begins with a < *doc id* **=** ... > tag, and ends with < */doc* >.

Note that all Australian legal cases begin with something like < doc id = "AU1222" ... > that is, the doc id for an Australian legal case always starts with AU. You will be trying to figure out if the document is an Australian legal case by looking only at the document's contents.

3 Assignment Tasks

3.1 f

First, you need to write Spark code that builds a dictionary that includes the 10,000 most frequent words in the training corpus. This dictionary is essentially an RDD with the word as the key and the relative frequency position of the word as the value. For example, the value is zero for the most frequent word and 9,999 for the least frequent word in the dictionary.

To get credit for this task, give us the frequency position of the words "applicant", "and", "attack", "protein", and "car". You need to have a printout in your code for these outputs. These should be values from 0 to 9,999, or -1 if the word is not in the dictionary.

3.2 Task 2 - Learning the Model (10 Points)

Next, you will convert each document in the training set to a TF ("term frequency") vector with 10,000 entries. For a particular document, the entry in the 177th value in this vector is double that tells us the frequency, in this document, of the 177th most common word in the corpus. Likewise, the first entry in the vector is a double that tells us the frequency, in this document, of the most common word in the corpus. It might make sense to store only the non-zero entries in this vector to keep the RAM requirements low.

You will then use a gradient descent algorithm to learn a logistic regression model that can decide whether a document is describing an Australian court case or not.

Your model should use l2 regularization; you can play a bit to determine the right parameter

for controlling the extent of the regularization. We have enough data that you might find that the regularization may not be too important. Please comment on this.

Once you have completed this task, you will get credit by printing out the five words with the largest regression coefficients. What are those five words that are most strongly related to an Australian court case?

- Remember that the problem is a classification problem, 1 for Australian court cases, 0 otherwise.
- Cache the important RDDs. You do not want to re-compute the RDDs each time you make a pass through the data during learning.
- When you debug your code, the first debugging step is to verify that the loss function is correctly decreasing as the learning progresses, as expected.
- First, implement your logistic regression without Regularization. Then, implement it with regularization, do some test runs and find a good regularization value.
- Feel free to play with the parameters. Learning rate can be any value that leads to algorithm convergence. There are more effective optimizers than Stochastic Gradient Descent that will make convergence faster.
- You need to find parameters that will lead to convergence of the optimization algorithm, and then it is enough to run the algorithm for 50 iterations.

3.3 Task 3 - Evaluation of the learned Model (8 Points)

Now that you have trained your model, it is time to evaluate it. Here, you will use your model to predict whether or not each of the testing points corresponds to Australian court cases. To get credit for this task, you need to compute for us the F1 score obtained by your classifier.

Look at the test samples for three of the false positives that your model produced. Write a paragraph describing why you think your model was fooled. Describe what about are the false-positive documents and why the model failed. If you don't have three false positives, just use the ones that you had (if any)

Important Considerations

4.1 Machines to Use

One thing to be aware of is that you can choose virtually any configuration for your Cloud Cluster - you can choose different numbers of machines and different configurations of those machines. And each is going to cost you differently! 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, you'll then move to Cloud.

As a proposal for this assignment, you can use the **n1-standard-4** or **e2-standard-4** machines on the Google Cloud, one for the Master node and two for worker nodes.

Remember to delete your cluster after the calculation is finished!!!

More information regarding Google Cloud Pricing can be found here https://cloud.google.com/products/calculator. As you can see average server costs around 50 cents per hour. That 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 Google and Amazon charge you when you move data around. To avoid such charges, do everything in the ”Iowa (us-cental1)” region. That’s where data is, and that’s where you should put your data and machines.

- You should document your code very well and as much as possible.
- Your code should be compilable on a Unix-based operating system like Linux or macOS.

4.2 Academic Misconduct Regarding Programming

In a programming class like ours, there is sometimes a very fine line between ”cheating” and acceptable and beneficial interaction between peers. Thus, it is essential that you fully understand what is and what is not allowed in collaboration with your classmates. We want to be 100% precise, so 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.

4.3 Turnin

Create a single document that has results for all three tasks.

Also, for each task, for each Spark job you ran, include a screenshot of the Spark History.

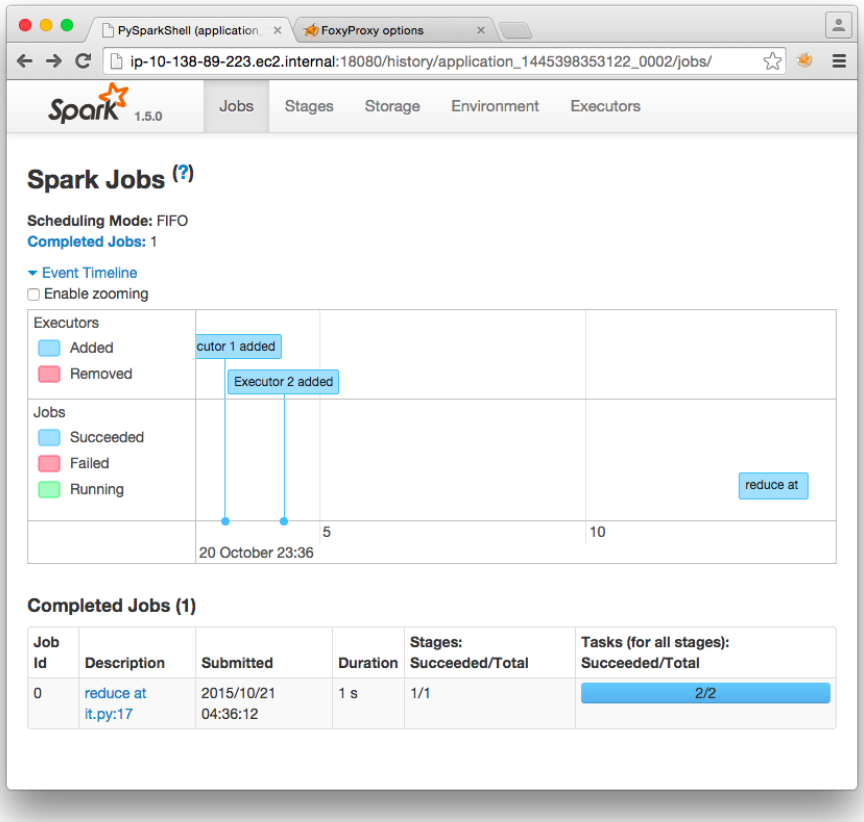

Figure 1: Screenshot of Spark History

Please zip up all of 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.

Please have the latest version of your code on GitHub. Zip the GitHub files and submit your latest version of assignment work to Blackboard. We will consider the latest version of the Blackboard