CSE4/587 Data-intensive Computing Spring 2017

LAB 3: DATA ANALYTICS PIPELINE USING APACHE SPARK: B. RAMAMURTHY

OVERVIEW:

The hands-on practical learning components of the course comprises two types of activities: labs covering one or two knowledge units (skills, competencies) of data-intensive computing and a single term project serving as a capstone covering the entire data pipeline. In the first half of the course we learned data analytics and quantitative reasoning using R language. In the second half we focused on big data approaches for data analytics with Hadoop MapReduce and Apache Spark. In this lab, we will work on understanding concepts related to data analysis using Apache Spark [1].

In Lab1 we learned to analyze data using R language and R Studio. In lab2, we explored approaches that deal with big data, especially text data, using the Google's MapReduce algorithm. In Lab 3 we will explore processing graph data using Spark [1]. Here is a chance to show case your knowledge in Apache Spark data pipelines and big-data analytics. You can apply the model you build here, to numerous applications. Apache Spark is fantastically suited for this application we will describe here. And the application has uses in many vertical domains.

LEARNING OUTCOMES:

In this lab work you will be able to

1. **Explore** the Apache Spark framework and programming: spark context (sc), dataflow operations in **transformations, actions, pipelines and MLib.**
2. **Apply** your data analytics knowledge (word frequency, word-co-occurrence) and machine learning skills to perform multi-class classification of text data using Apache Spark.
3. **Build a data pipeline** using
    a. Data from sources such as NY Times articles using the APIs provided by the data sources.
    b. Split the data into training and test data set.
    c. Extract features that will determine the class or category of the article {politics, sports, business, one of your choice}
    d. Build a model for classification using any two of the several classification algorithms.
    e. **Assess** the accuracy using a query text or new article for each news "category"
    f. **Compare** the classification accuracy of at least two well-known classification algorithms, for a given text data set.
4. **Document** the design and implementation of the project using either markup or markdown [ref] language.
5. **Apply the knowledge and skills learned to solve classification problems in other domains.**

LAB DESCRIPTION:

Introduction: In this age of analytics, data science process plays a critical role for many organizations. Several organizations including the federal government (data.gov) have their data available to the public for various purposes. Social network applications such as Twitter and Facebook collect enormous amount of data contributed by their numerous and prolific user. For other businesses such as Amazon and NYTimes data is a significant and valuable byproduct of their main business. Nowadays everybody has data. Most of these data generator businesses make subset of their data available for use by registered users for free. Some of them as downloadable data files (.csv, .xlsx) as a database (.db, .db3). Sometimes the data that needs to be collected is not in a specific format but is available as a web page content. In this case, typically a web crawler is used to crawl the web (pages) and scrap the data from these web pages and extract the information needed. Data generating organizations have realized the need to share at least a subset of their data with users interested in developing applications. Entire data sets are sold as products. Very often data collected is not in the format required for the downstream processes such as EDA, analysis, prediction and visualization. The data needs to be cleaned, curated and munged before modeling and algorithmic processing. **All the data pre-processing will be done in the MR or Spark framework and not outside.**

Also we will follow the pedagogical pattern

- Preparation before lab (pre-lab)
- Learn from working on some of the solved problems.
- Apply the knowledge to meet the goals of this project to study scalability.

PREPARATION: Here are the preliminary requirements for the lab. **Time needed: 2 to 3 hours (Day 1)**

1. Work in groups: You will work in groups of 1 or 2. Find partner who will share the work with you equally and also have complementary resources and skills so that you can learn from each other. No more than two per group is allowed.
2. Study the architecture, APIs, and operations of Apache Spark.
3. Choose an Apache Spark environment: VM[4], docker, Jupyter, Windows/mac/linux native installations, google app engine [5], amazon EMR[6].
4. Make sure the environment you have chosen works well with Apache-Spark sample programs such as word count. Try it out and get familiarized with the dataflow, commands and the concept of Spark context.

Lab 3: What to do?

1. **(15 points) Understand Apache Spark with Titanic data analysis. (Day 2,3: Time Needed: 3-4 hours per day)**
   This activity is similar to an R-vignette. In this case you learn features of Apache Spark. On Jupyter's [9] start page the "Try it in your browser" and once you are inside the notebook, click on the Python or Scala version of the Spark notebook. Run the same analysis on the environment you created in the Pre-lab and make sure it works.
2. **(10 points) Collect and clean data:** Use your Lab 2 experience to collect news articles for known categories in a directory and call this directory. Do not combine the articles into one. Collect

enough data to split the set into trainset and testSet. You will collect data in all the categories listed in the Learning Outcomes section. You can also collect the query data files. You can use Hadoop MR or Apache Spark to tokenize and clean the

3. (**25 points) Feature Engineering**: Extract the "words" or the "features" characterizing the category. Use only top N frequently occurring words of each category to be the features. Unlike Lab2 instead raw count you will compute the probability of the word frequency to the total words in the article (file). You will use the training set data articles to do accomplish this step. Other methods for feature extraction is provided here [3]. You can use Java, Scala or Python language. Choose one and stick with it. Do not switch languages for the various the steps. By the end of this step you have characterization of the category.
4. **(30 points) Multi-class Classification:** In this step you can use several different methods: Naïve Bayes, support vector machine, neural network, etc. Choose any two and classify the test set and study the accuracy.
5. **(10 points) Testing:** Try the model you built with unknown set of articles (not test set) and see how it performs for the two classification methods. Based on the test results you may have to redo the step 3 with more data or different algorithm.
6. **(10 points) Documentation:** Document the whole process using markup or markdown. Since Spark is supported by Jupyter you should be able to create a very nice notebook to share at technical interviews. You don't need to use Jupyter. Hint: Provide a block diagram of your data pipeline and explain each of the elements of that pipeline.

SUBMISSION DETAILS:

1. Readme file containing your name (First and last) and your partner's name at the top with all the other details such as environment you have chosen etc. how to run your program and explanation of the output.
2. You may need to submit the "typescript" and also screen shots of your program working.
3. Other details of format of data outputs will be given to you later.
4. Online submission: do not email me: (-10 points) if you email me the code.

DUE DATE: 5/11/2018 BY 5.00 PM. ONLINE SUBMISSION.

REFERENCES:

[1] Apache Spark. http://spark.apache.org/, last viewed 2018.

[2] Titanic, Kaggle Competition Solution. https://benfradet.github.io/blog/2015/12/16/Exploring-spark.ml-with-the-Titanic-Kaggle-competition , last viewed 2018.

[3] G. Kaur and K. Bajaj. News Classification and Its Techniques: A Review, http://www.iosrjournals.org/iosr-jce/papers/Vol18-issue1/Version-3/D018132226.pdf, viewed 2018.

[4] Virtual Machine for Hadoop MapReduce on UBbox: https://buffalo.box.com/s/52did77hn2vjoje7iguf19btgs6vhvsc, last viewed 2017.

[5] Amazon AWS Elastic MapReduce. https://aws.amazon.com/emr/ , Last Viewed April 2017.

[6] MapReduce on Google App Engine. https://cloud.google.com/appengine/docs/standard/python/dataprocessing/, last viewed April 2017.