Project 2 Overview

CS 143 Project 2

This project is NEW to CS 143, so please bear with me. There may be technical issues, or data issues, and I am happy to fix them.

The point of this project is to teach you something useful, and to perhaps have some fun with it.

As part of class assignments, students will have to finish two class projects that are split into multiple submissions. In this second project, students will have to implement a Web site supported by both Spark, a very important big data processing system, and a DBMS.

You should find part of Project 2 to be similar to Project 1 except with Spark, and a bit of machine learning, but the machine learning will be gentle.

Prerequisite

We will assume that students are sufficiently proficient already with UNIX and have enough programming experience to learn the PHP programming language quickly. We expect the resources and links on the project pages will provide you with enough information to get you started even in case you are not familiar with either one.

We also expect students to be able to learn either Python (preferred) or Scala, as both languages can be used to interact with Spark.

System Setup

To help students set up the uniform environment for the class project, we will be using VirtualBox to run the Linux operating system in a virtual machine. VirtualBox allows a single machine to share resources and run multiple operating systems simultaneously. You will need to download the following files

- VirtualBox binary file for your host Operating System
- VirtualBox Image: CS143-P2.ova (requires UCLA BOL login) - This is a very large file (~3-4GB), so it may take a while to download.

and follow our VirtualBox setup instruction to install VirtualBox on your own machine.

The provided virtual machine image is based on Ubuntu 17.10 (18.04 does not seem to work yet), MySQL 5.7, Apache 2.4.27, PHP 7.1, Python 3.6.3, Scala 2.11.8 and Spark 2.3.0.

If you have access to an equivalent machine that has these packages, you may use it instead of the virtual machine image. However, please note that we cannot provide support for systems other than the virtual machine image, and that **your project MUST be runnable on the provided virtual machine**. We will be using the virtual machine image for grading purposes, and if your submission does not work within this setup, you may get zero points. We cannot make any exceptions to your project schedule for problems incurred by using your own computing facilities.

I have minimized the size of the data for this project. If you are using your own system, and are interested, you can download the full data for the November 2016-February 2018 period.

Project

Reddit (reddit.com) is a hybrid URL bookmark and/or news site, and a forum. Redditors post articles, links, or text on specific topics and other Redditors vote the submission up (meaning "like") or down (meaning "don't like"), and can also write comments in response to the submission.

Politics is the subject that everyone either loves, hates or loves to hate. But, it is also a very polarizing and emotional subject, so it provides a good basis for doing some machine learning. The midterms are coming up, so we will use this Reddit data to find the sentiment across time, across topics, and across states regarding President Trump, Republicans, and Democrats. **I do not know what results we will get. It may be**

that this data is useless, but we do not know until we try to use it!

__Caveat:__ **/r/politics, and Reddit in general, is known to bias heavily towards young (18-29, 59%) males (71%) that lean liberal / progressive / left (47% of overall users, but much higher *anecdotally* in /r/politics), so we are already starting with biased data, but let's see what we can get from it.**

Jason Baumgartner has collected every Reddit submission and every Reddit comment posted on the site since 2005. This data is available on his website in BZ2 or XZ format.

I have taken this data and restricted it to November 2016-February 2018, only kept comments between 10-50 words, and restricted comments submissions to only the /r/politics subreddit. Additionally, it contains *all* submissions and comments by *anybody* that has posted in /r/politics, as this may come in useful in Part B.

When decompressed, we get a JSON file containing data about each submission or comment. Your VM contains two files in the data directory:

1. `comments-minimal.json.bz2` containing a subset of comments to /r/politics from November 1, 2016 (before election day), to February 28, 2018. Because the original data was about 15GB in size, I restricted this dataset to comments containing between 10 and 50 words.
2. `submissions.json.bz2` contains all submissions made to /r/politics from November 1, 2016 to February 28, 2018.

Your project is to parse the comments (Python or Scala), use what you have learned in 143 about joins etc. to massage the data into the proper format (Spark and/or SQL in Spark) for training a *sentiment classifier* (Spark MLLib). Then, if time allows, create some type of visualization dashboard to present your findings in PHP, or JavaScript. We will see how it goes.

Part A: Text Parsing

In Part A, you will write a function, preferably in Python (though Scala will work as well) to take horribly messy text from Reddit comments, and parse them into a smooth format that we can eventually use to train a classifier.

Due Date: **Monday, May 21, 2018, 11:59pm**

Part B: Transforming Data and Training Classifier

In this part, you will use the function you wrote for Part A to parse the text (if you didn't already do it in Part A), into a data frame containing not only text, but several other features. You will then train a classifier using a Spark package called mllib. I may also provide the classifier to you, TBD.

Due Date: **Tuesday, May 28, 11:59pm**

Part C: Dashboard

Depending on time, you will then create a dashboard with a few visualizations about sentiment has changed over time, over location, and other interesting findings.

Due Date: **Wednesday, June 6, 11:59pm**

Groups

Students may implement the project individually or in **teams of up to four**. We recommend that two partnerships team up, or a partnership and a solo, or 3-4 solos, whatever. **The Zaniolo "No Remarriage" rule still applies, but not it's polyamorous.** The choice is up to each student, but please keep the following rules in mind when you select your project partner:

An identical amount of work is expected and the same grading scale is used for individual and team projects. Faculty experience indicates that in general it is not necessarily easier or more productive to work in teams of two - it's largely a matter of personal preference and working style. If you choose to work as a team, you are encouraged to make use of collaborative authoring tools for synchronizing your work and ideas, such as version control software (e.g. CVS, SVN, Perforce, **Private** git) and online document tools (e.g. Adobe Share, Buzzword, Google Docs, Dropbox Paper).

If you work in a team, choose your partner carefully. Teams are permitted to "divorce" at any time during the course (due to incompatibility, one partner dropping the course, or any other reason), and individual students may choose to team up as the project progresses, however students from divorced teams may not form new teams or join other teams. Put another way, if a student turns in any part of the project as part of a team, every later part of the project must be turned in individually or as part of the same team.

Both partners in a team will receive exactly the same grade for each project part turned in jointly. We will not entertain any complaints of the form "I did all the work and my partner did nothing." Choose your partner carefully!

If you work in a team, your work must be turned in jointly, as ONE submission. That is to say, only ONE of you two should submit your work as a team. Your team will get 10 points off as penalty if you violate this rule. Note that teamwork turned in as individual work will be considered as plagiarism and handled through official University channels.

Late Submission Policy

To accommodate the emergencies that a student may encounter, each student (or team) has a **4-day grace period** for late submission to use throughout the quarter. The grace period can be used for any part of the project in the unit of one day. For example, a student may use 1-day grace period for part 1B and 2-day grace period for part 2A. **Any single project part may not be more than 2 days late.** Note that the grace period can be used in the unit of one day. even if a student submits a project 12 hours late, he/she needs to use a full day grace period to avoid late penalty.

Electronic submission of Projects

All project submission should be done electronically. Steps for submitting your project electronically is as follows:

1. Visit the online submission page for the particular project, linked from the corresponding assignment page.
2. Fill in the necessary information and click Submit. You should receive a confirmation page and email with the timestamp and an MD5 hash for each of the files received.
3. If you need to resubmit something, just redo these directions. The submission page will notice if you are attempting to resubmit and overwrite your previous entry. Remember that only the very **last** submission will be graded.

Project References

Unix & VirtualBox

- Unix tutorial
- VirtualBox overview

PHP references

- W3Schools PHP tutorial
- Input handling in PHP
- MySQL access in PHP
- PHP Regular Expression Tutorial
- PHP Perl Compatible Regular Expressions API
- PHP-MySQL API reference
- PHP function reference

MySQL references

- Introduction to MySQL
- Constraints in MySQL
- Accessing MySQL from PHP
- Date and Time types in MySQL
- MySQL documentation
- PHP-MySQL API reference

Python references

- Python 3 Reference
- W3Schools Python Tutorial
- "The" Python Tutorial
- Python for Beginners
- The Python Guru

Spark references

- UCLA Brief Tutorial on Spark MLLib (Go Bruins!)
- Spark Quickstart
- Hands On Tutorial of Apache Spark in 10 Minutes
- Machine Learning in Spark (includes info about PySpark)

Last modified: Thursday, 17 May 2018, 3:01 AM PDT