Eric Perkins

Exercise 2 Architecture

Directory and File Structure:

The exercise_2 repo holds the following files and directories:
Files:
- Plot.png (matching requirement in instructions, showing top 20 words in stream)
- README.txt (matching requirement in instructions)

Directories:
1) screenshots
   1) twitter_stream_running.png: Shows running twitter stream after appropriate topology, spout and bolt modifications were made, as well as the creation of a Twitter application with all credentials added to necessary files
   2) final_results_with_argument.png: Shows successfully running final results with ‘you’ argument
   3) final_results_no_argument.png: Shows successfully running final results file without arguments, producing alphabetically sorted tuples of words and counts.
   4) histogram_running.png: Shows successfully running histogram.py file with argument “3,9”
2) ex2files
   1) ex2_instructions: PDF file with instructions for exercise 2
   2) tweetwordcount: Project folder provided by w205 git repository
3) EX2Tweetwordcount: Project files, info below

Files which I edited and created:

Edited:
1) EX2Tweetwordcount/topologies/EX2tweetwordcount.clj
   1) Changed number of parallel processes to be 1 in each spout and bolt
2) EX2Tweetwordcount/src/spouts/tweets.py
   1) Added credentials where indicated
   2) Changed the “timeout” in “next_tuple” method from 0.1 to “None” to avoid empty queue exceptions
3) EX2Tweetwordcount/src/bolts/wordcount.py
   1) Used psycopg2 to create table in tcount database
   2) Added all communication related to Postgres
4) EX2Tweetwordcount/Twittercredentials.py
   1) Added credentials where indicated

Created (all in EX2Tweetwordcount):

1) finalresults.py
2) histogram.py
3) t20plot.py (Used to produce Plot.png)
4) Twittercredentials.pyc (Created automatically when running application)

Dependencies
Python 2.7, numpy, matplotlib, sys, os, tweepy, psycopg2, pip, virtualenv, streamparse, lein, EasyButton script

Application Decision Note

This application tracks the frequency of words in tweets. The definition of "words" here is up for interpretation, because the code as provided for us, even though it filters the tweets somewhat, leaves things like "!!!!" and "&d02l" to count as words. After playing with the idea of additional filtering, I decided to allow the words to be counted as-is. It is too bold, in my opinion, to claim that alphanumeric characters are all we should be tracking or all that would convey analysis-worth sentiment and content in the tweets. Therefore, if punctuation and random character strings are present, they will be recorded in the table. Alphabetically-sorted lists will return many non-alphabetical strings before words that start with 'a' for example.

Github Repo Info

Running this code relies on cloning my repo for the assignment,
https://github.com/eperkins1/exercise_2.git

Architecture Description:

This streaming application follows the directed architecture from the instructions. Data is generated via a Twitter application, which I created, which connects to a spout in my project. This spout receives data in the form of tweets from the application using tweepy, and this data fills and passes through a queue in the spout.

This tweets spout emits each tweet to the parse bolt, which filters and parses the tweet string into individual words before emitting these words to the wordcount bolt. The wordcount bolt first creates a table in a given Postgres database, and receives words from the parse bolt. Each word that passes through the bolt filters results in either a new row addition to the PG table, or an increment of the count associated with that word.

Step-by-Step Setup Instructions (Run all instructions as root where applicable)

1) Create instance with only 10GB root using ucbw205_complete_plus_postgres_PY2.7
Ports: 4040, 50070, 8080, 22, 10000, 8020, 8088
2) Follow Appendix instructions to install Python 2.7 (Bottom of page 10)
3) Follow Appendix instructions to install Streamparse (Page 9)
4) Run pip install psycopg2
5) Run pip install tweepy
6) Run wget https://s3.amazonaws.com/ucbdatasciencew205/setup_ucb_complete_plus_postgres.sh
7) Run bash setup_ucb_complete_plus_postgres.sh /dev/xvdb (or a different last argument if /dev/xvdb is not the name of your disk)
8) Clone my git repository:
SSH: git@github.com:eperkins1/exercise_2.git
OR
HTTPS: https://github.com/eperkins1/exercise_2.git

9) Create Postgres database “tcount”

Run the following commands to create Tcount database:

```
psql -U postgres
CREATE DATABASE Tcount;
\q
```

10) Run stream:

cd into exercise_2 repo

cd into EX2Tweetwordcount folder, and run “sparse run”

Keep it running for as long as you’d like to build up the table

11) Run either of the python files, with option argument for finalresults.py:

histogram.py

finalresults.py

12) To create the Plot.png, import matplotlib and numpy by running the following commands, in the EX2Tweetwordcount directory:

```
mkdir /usr/local/lib/temph/
mv /usr/local/lib/libpython2.7.a /usr/local/lib/temph/
pip install numpy
pip install matplotlib
mv /usr/local/lib/temph/libpython2.7.a /usr/local/lib/
python t20plot.py
```